Christoph Lohrmann

HEURISTIC SIMILARITY- AND DISTANCE-BASED SUPERVISED FEATURE SELECTION METHODS

ACTA UNIVERSITATIS LAPPEENRANTAENSIS 890

Christoph Lohrmann

HEURISTIC SIMILARITY- AND DISTANCE-BASED SUPERVISED FEATURE SELECTION METHODS

Dissertation for the degree Doctor of Philosophy (Computational Engineering) to be presented with due permission for public examination and criticism in the Auditorium 1316 at Lappeenranta-Lahti University of Technology LUT, Lappeenranta, Finland on the 16th of December, 2019, at noon.

Acta Universitatis
Lappeenrantaensis 890

Supervisors Professor Pasi Luukka
LUT School of Business and Management
Lappeenranta-Lahti University of Technology LUT
Finland

Professor Mikael Collan
LUT School of Business and Management
Lappeenranta-Lahti University of Technology LUT
Finland

Post-Doctoral Researcher Matylda Jabłońska-Sabuka
LUT School of Engineering Science
Lappeenranta-Lahti University of Technology LUT
Finland

Reviewers Associate Professor Yuri Lawryshyn
Faculty of Applied Science and Engineering
University of Toronto
Canada

Professor Frank Chung-Hoon Rhee
Department of Electronics and Computer Engineering
Hanyang University
South Korea

Opponent Associate Professor Yuri Lawryshyn
Faculty of Applied Science and Engineering
University of Toronto
Canada

ISBN 978-952-335-472-2
ISBN 978-952-335-473-9 (PDF)
ISSN-L 1456-4491
ISSN 1456-4491

Lappeenranta-Lahti University of Technology LUT
LUT University Press 2019

Abstract

Christoph Lohrmann
Heuristic similarity- and distance-based supervised feature selection methods
Lappeenranta 2019
128 pages
Acta Universitatis Lappeenrantaensis 890
Diss. Lappeenranta-Lahti University of Technology LUT
ISBN 978-952-335-472-2, ISBN 978-952-335-473-9 (PDF), ISSN-L 1456-4491, ISSN 1456-4491

In the field of machine learning, the available data often contain many features to describe phenomena. This can pose a problem since only those features that are relevant to characterize the target concept are needed, whereas additional features can make it even more complicated to determine the underlying association between the features and the phenomenon. Therefore, an essential task for data analysis is feature selection, which means to reduce the number of features in the data to a set of relevant features. The focus in this thesis is on supervised feature selection methods used in the context of classification tasks. In particular, the emphasis is on heuristic filter methods, which do not guarantee an optimal solution but are considerably faster and are deployed as a pre-processing step for the data before a classification algorithm is applied.

The first approach presented is the 'fuzzy similarity and entropy' (FSAE) feature selection method, which is a modification of the approach by Luukka (2011). It is demonstrated that this approach, which evaluates each feature by itself (a univariate approach), accomplishes at least comparable classification results to the original approach, often with a considerably smaller feature subset. The results were competitive to those of several other distance- and information-based filter methods. In addition to several artificial examples and real-world medical datasets, the FSAE was deployed together with a random forest to construct a classification model for the prediction of the S&P500 intraday return. Several trading strategies derived from the classification model demonstrated the ability to outperform a buy-and-hold strategy with small to moderate transaction costs. In the context of classification, the similarity classifier, which as the FSAE feature selection method works with a single representative point (ideal vector) for each class, was modified to allow for multiple ideal vectors per class using clustering. This classifier was able to outperform all single classifier models it was compared to in terms of classification accuracy, often by a significant margin. The same idea of using multiple class representatives was successfully applied in the context of feature selection with the proposed 'clustering one less dimension' (COLD) algorithm. In addition, the distance-based COLD filter algorithm is capable of accounting for dependencies among features (a multivariate approach). This ability was highlighted on several artificial examples. Lastly, it achieved at least competitive results compared to several other heuristic filter methods on real-world datasets.

Keywords: feature selection, feature ranking, dimensionality reduction, filter method, similarity classifier, FSAE, COLD, supervised learning, machine learning

Acknowledgements

The journey towards writing a dissertation is often depicted as challenging, laborious and, at times, frustrating. Even though I also faced setbacks during this time, I felt they were few in number and, at least retrospectively, relevant for my personal advancement. However, it is clear that I would not be able to look back at the past three years with a feeling of gratefulness if it were not for the many special people that accompanied me during this journey.

First and foremost, I would like to thank my three supervisors. I am deeply grateful for Professor Pasi Luukka's continuous support and encouragement to pursue ideas and engage in discussions and his sympathetic ear for problems of any kind I was facing during that time. Moreover, I would like to thank Professor Mikael Collan and Professor Pasi Luukka for their support in finding project works as well as funding and providing me with the opportunity to teach. In this context, I would also like to thank the other members of the business analytics team at LUT – Mariia, Jyrki, Jan, Sheraz and Azzurra. I have always felt supported, and all of them made me feel as part of the team, which I am very grateful for. In addition, I would like to thank Professor Ari Jantunen, who was very open and supportive right from the start of my employment in the School of Business and Management.

In terms of funding, I would like to express my appreciation for the financial support obtained by the Manufacturing 4.0 (MFG 4.0) project of the Finnish Strategic Research Council as well as two project-related personal grants obtained from the LUT Foundation. None of these could I have obtained without the invaluable support of Professor Mikael Collan.

I would like to thank my supervisor post-doctoral researcher Matylda Jabłońska-Sabuka for providing me with the opportunity to start as a doctoral student in the School of Engineering Science. It was she and Dr Tuomo Kauranne that believed in my ideas and made it possible to return to LUT after my exchange at LUT during my master studies. In addition, I would like to thank the staff at the Doctoral School, especially Sari Damsten and Saara Merritt, for their support in any matters related to doctoral studies.

Moreover, I would like to stress my appreciation for the friends that I shared an office and many emotions with: Mahinda Mailagaha Kumbure, Sebastian Springer, Ramona Maraia and Dipal Shah. Last but not at least, I would like to express my appreciation for the support from my family and friends. I would like to thank my wife, Alena, for her continuous belief in me and for having the right advice and words of encouragement at the right time. Finally, I would like to express my gratitude to my parents, my grandfather and my friends Arthur, Benedict and Fabian in my home country. I always felt that no matter where I am, I have your support and a place in your hearts, as you do in mine.

Christoph Lohrmann
August 2019
Lappeenranta, Finland

‘We are drowning in information and starving for knowledge’
– Rutherford D. Rogers

To my wife, Alena

Contents

List of Publications

This dissertation is based on the following papers. The rights have been granted by the publishers to include the papers in this dissertation.

I. Lohrmann, C., Luukka, P., Jabłońska-Sabuka, M., and Kauranne, T. (2018). A combination of fuzzy similarity measures and fuzzy entropy measures for supervised feature selection. *Expert Systems with Applications*, 110, pp. 216-236. **Publication Forum JUFO Level: 1**

II. Lohrmann, C., and Luukka, P. (2019). Classification of intraday S&P500 returns with a random forest. *International Journal of Forecasting*, 35, pp.390-407. **Publication Forum JUFO Level: 2**

III. Lohrmann, C., and Luukka, P. (2018). A novel similarity classifier with multiple ideal vectors based on k-means clustering. *Decision Support Systems*, 111, pp.27-37. **Publication Forum JUFO Level: 2**

IV. Lohrmann, C., and Luukka, P. (2019). Using clustering for supervised feature selection to detect relevant features. Accepted at the *Fifth International Conference on Machine Learning, Optimization, and Data Science (LOD 2019)*. **Publication Forum JUFO Level: 1**

Author's Contribution

For **Publication I**, the author analysed the vulnerabilities of the existing feature selection algorithm by Luukka (2011), proposed the modification, suggested artificial examples to illustrate the abilities of the modified algorithm, implemented it in MATLAB code, and is the main author of the paper.

For **Publication II**, the author selected and collected the financial data, proposed the trading strategies, implemented the MATLAB code and is the main author of the paper.

For **Publication III**, the author proposed the novel similarity classifier, suggested artificial examples to illustrate its abilities, implemented the MATLAB code and is the main author of the paper.

For **Publication IV**, the author proposed the multivariate feature selection method, suggested artificial examples to illustrate its abilities, implemented the MATLAB code and is the main author of the paper.

Nomenclature

Mathmematical Notations

α	ridge regularization parameter
D	number of features
H	entropy
m	generalized mean parameter
μ	mean value of a distribution
N	number of classes
n	number of observations/samples
p	Łukasiewicz parameter
r	ratio of distances (COLD algorithm)
S	similarity
SE	scaled entropy
SF	scaling factor
σ	variance
Σ	covariance matrix
v	ideal vector
X	dataset
x	observation of the dataset

Abbreviations

COLD	Clustering one less dimension
ETF	Exchange traded fund
FS	Feature selection
FSAE	Fuzzy similarity and entropy
KNN	K-nearest neighbour
LVF	Las Vegas filter
MAP	Minimum average partial (test)
MA	Moving average
MACD	Moving average convergence divergence
MVDE	Multiple vector differential evolution
OWA	Ordered weighted average
PA	Parallel analysis
PC	Principal component
PCA	Principal component analysis
RSI	Relative strength index
SVM	Support vector machine
VIX	Volatility index

1 Introduction

In machine learning and data analysis, it is common for there to be a ‘wealth’ of information and features available for real-world problems (Caruana and Freitag, 1994a; Blum and Langley, 1997). In this context the term ‘feature’ means a ‘measurable property’ (Chandrashekar and Sahin, 2014) and is interchangeable with the term ‘variable’ or ‘attribute’. For instance, in connection with medical data, a feature can be the height, age, blood pressure or information on the medical history of a patient. For a credit evaluation, a feature can be the balance, income, credit history or the profession of a customer. In many applications, the number of features available for use have increased considerably (Chandrashekar and Sahin, 2014). What may appear at first glance an unreserved benefit for machine learning algorithms poses problems for these algorithms that do not appear in simple textbook examples (Caruana and Freitag, 1994a). The main drawback of high-dimensional data, meaning data containing a large quantity of features and often also of observations, is its potentially adverse effect on the generalization. Features not relevant to or useful for a machine learning task can act as noise, interfering with actually useful features and making it harder for the algorithm to determine the actual signal or pattern(s) in the data (Caruana and Freitag, 1994a; Dessì and Pes, 2015). Hence, using irrelevant features in a classifier that has no feature selection embedded can lead to overfitting of the training data, which leads to a worse classification accuracy on the test data. For instance, Luukka and Lampinen (2011) investigate and show that adding additional irrelevant features that act as noise deteriorates the classification accuracy of their classifier on the test set. Hence, additional features can deteriorate the performance of an algorithm in addition to the higher computational complexity that naturally follows from higher dimensional data (Rhee and Lee, 1999; Dessì and Pes, 2015). Therefore, using, for instance, a suitable subset of features instead of all available data can improve the generalization of an algorithm (Caruana and Freitag, 1994a). Unfortunately, the common issue in real-world problems is that it is unknown which features are relevant (Almuallim and Dietterich, 1994; Piramuthu, 2004). As a consequence, practitioners may feel inclined to introduce and simply keep numerous features to address their task (Almuallim & Dietterich, 1991; Dash & Liu, 1997). However, the more complex and high-dimensional a task is, the more essential it becomes to focus on the relevant features (Blum and Langley, 1997). To decrease the number of features, meaning the number of dimensions under consideration, so-called dimensionality reduction techniques can be applied.

A categorization of dimensionality reduction methods and selected related techniques is outlined in Figure 1.1.

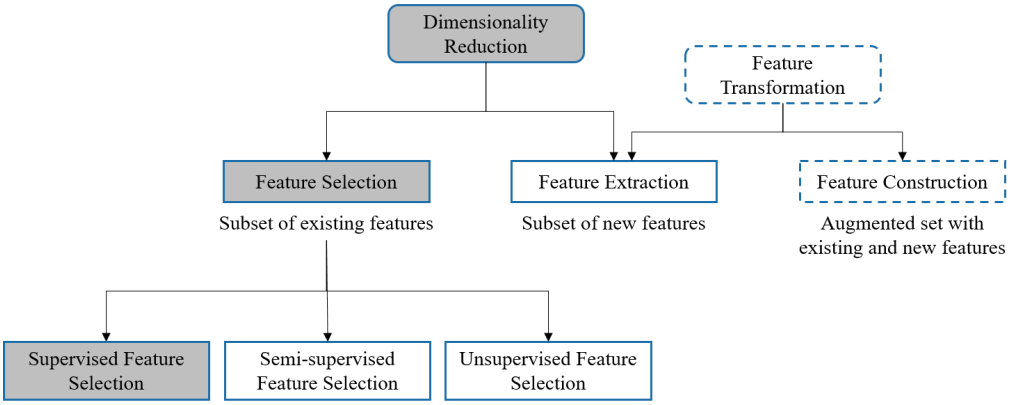


Figure 1.1: Taxonomy of dimensionality reduction techniques

Dimensionality reduction techniques are commonly divided into feature selection and feature extraction (Liu and Motoda, 2001; Li *et al.*, 2017). Feature extraction characterizes a process that uses the original set of features, projects it into a different lower-dimensional feature space and extracts a new set of features that is smaller than the original one (Liu and Motoda, 2001; Li *et al.*, 2017). A classic and popular representative of feature extraction is principal component analysis (PCA), where a smaller feature set is obtained by linearly transforming the original features (Mitra, Murthy and Pal, 2002; Motoda and Liu, 2002). Feature extraction belongs to the group of feature transformation methods which change the original features by applying some form of transformation. Feature transformation also includes the process of feature construction, which works inherently differently than feature extraction and is not considered a dimensionality reduction technique. The reason, therefore, is that feature construction augments the existing set of features by creating and adding new features to the feature set based on the existing features (Wnek and Michalski, 1994; Liu and Motoda, 2001; Piramuthu and Sikora, 2009). In this way, it attempts to improve the information and expressive power contained in the original features (Motoda and Liu, 2002). However, this method obviously increases the number of features, which is the opposite of dimensionality reduction.

Feature selection stands in stark contrast to feature transformation techniques such as feature extraction but also feature construction. Feature selection can be defined as the process of selecting a subset of the existing features measured according to some criterion (e.g. classification accuracy/error rate, distance measure, etc.) (Liu and Setiono, 1996; Bins and Draper, 2001; Liu and Motoda, 2001; Liu and Yu, 2005). Two aspects are essential in this definition – first, the existing features are unaltered, and, second, a subset is selected, meaning that the number of features and, hence, the dimension of the data, are reduced. In contrast to feature extraction, no transformation of the features is conducted, whereas in contrast to feature construction, no additional (transformed) features are added.

Feature selection has been successfully applied in various disciplines and applications. These applications include, but are not limited to, diagnosis models for business crises (Chen and Hsiao, 2008), credit scoring (Maldonado, Pérez and Bravo, 2017), image classification (Zhou *et al.*, 2017), diagnosis of Alzheimer's disease (Trambaiolli *et al.*, 2017), prediction of groundwater pollution and quality (Rodriguez-Galiano *et al.*, 2018) and astronomy (Zheng and Zhang, 2008).

One of the main advantages of feature selection is that it results in simpler classification models and benefits the generalization ability of these models (Mitra, Murthy and Pal, 2002; Saeys, Inza and Larranaga, 2007). This also positively impacts the classification accuracy achieved with the corresponding classification model (Guyon and Elisseeff, 2003; Liu and Yu, 2005; Chandrashekar and Sahin, 2014). Moreover, the feature subset obtained via feature selection has the advantage of containing unaltered features, which preserves the interpretability of the results (Saeys, Inza and Larranaga, 2007). This aspect and the lower dimensional feature set contribute to the ability to visualize and understand the features and the corresponding model more easily (Guyon and Elisseeff, 2003; Chandrashekar and Sahin, 2014). In addition, using fewer features for the model construction leads to improvements in the computational complexity and data storage requirements (Guyon and Elisseeff, 2003; Liu and Yu, 2005).

As highlighted in Figure 1.1, feature selection is commonly divided into supervised, semi-supervised and unsupervised forms (Ang *et al.*, 2016; Li *et al.*, 2017). In unsupervised feature selection, no class information (class labels) is available to indicate the group or class the observations belong to (Mitra, Murthy and Pal, 2002). A class label could, for instance, represent whether a credit applicant was paying his/her loan back ('0') or defaulted on the payments ('1'). Since the class labels are unknown, unsupervised feature selection methods evaluate features with respect to the inherent structure of the data (e.g. variance, distribution, separability, etc.) to determine a feature subset (Ang *et al.*, 2016; Li *et al.*, 2017). This type of feature selection is commonly used in the context of clustering, which is a form of unsupervised learning (Li *et al.*, 2017). Semi-supervised (or semi-unsupervised) feature selection is used when only for some proportion of the observations the class label is known, and for the remaining part it is unknown (Ang *et al.*, 2016). Finally, in supervised feature selection, the class label for all observations is known and used to select a subset of features (Ang *et al.*, 2016). Hence, supervised feature selection can be applied in the context of classification or regression (Li *et al.*, 2017), which are both forms of supervised learning. This dissertation centres on supervised feature selection for classification, where the labelled information of classes is available (highlighted in grey in Figure 1.1).

The aim of supervised feature selection is to find a set of relevant features. To be able to determine 'relevant' features, a definition of relevance is required (Molina, Belanche and Nebot, 2002). There is a variety of theoretical definitions of relevance in the context of feature selection, with some authors presenting multiple definitions (Caruana and Freitag, 1994b; John, Kohavi and Pfleger, 1994; Blum and Langley, 1997; Bell and Wang, 2000; Molina, Belanche and Nebot, 2002). As Blum & Langley (1997) point out, the diversity

of definitions is dependent on the context of relevance or, as they phrase it, in the form of the question ‘relevant to what?’. The focus in this research is on the practical aspect of feature selection as a means by which to reduce the number of features in the feature set and improve, or at a minimum not strongly deteriorate, the corresponding performance of a classifier compared to the complete feature set. Hence, the definition of relevance used in the context of this dissertation follows the fifth definition in Blum & Langley (1997) on ‘incremental usefulness’. According to that definition, for a sample *S*, a learning algorithm *L* and a set of features *D*, a feature *d* is incrementally useful to the classifier *L* with respect to the feature set *D* if the classification accuracy of the hypothesis produced by *L* using the features *D* and *d* together is higher than that of feature set *D* alone (Blum and Langley, 1997). In simple terms, if adding a feature *d* to a set of features *D* improves the mean accuracy of the classifier used on them, then feature *d* is ‘incrementally useful’, which means relevant. Hence, the features assumed to be relevant according to the feature selection methods in this research are those that have shown to have improved the classification accuracy for a classifier or are considered to contribute to the discrimination among classes and, hence, are assumed to improve the classification accuracy.

After having clarified the meaning of the term ‘relevance’ for feature selection in the context of this research, it is essential to discuss the distinction of supervised feature selection methods in order to set the scope of this dissertation. Supervised feature selection is commonly subdivided into the three forms (Figure 1.2) of filter, wrapper and embedded methods (Liu and Yu, 2005; Saeys, Inza and Larranaga, 2007; Chandrashekar and Sahin, 2014).

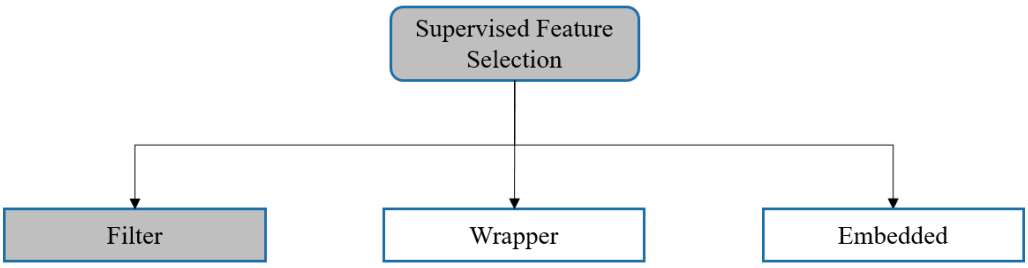


Figure 1.2: Types of supervised feature selection

These three types and their respective advantages and disadvantages can be summarized as follows:

Filter method: The filter method is part of the pre-processing of the data, meaning that it is used before a classification algorithm (= induction algorithm) is applied to the data (John, Kohavi and Pfleger, 1994; Blum and Langley, 1997). These methods evaluate the

relevance of a feature based on the characteristics of the features in the data and then rank the features according to an evaluation criterion (Liu and Yu, 2005; Saeys, Inza and Larranaga, 2007). Then, a user-specified number of the highest-ranked features can be retained, or a threshold can be used such that all features with a score exceeding the threshold are included in the feature subset (Saeys, Inza and Larranaga, 2007; Chandrashekar and Sahin, 2014). Since these methods 'filter' out features that are not considered relevant even before any classification algorithm is used, they are independent of a classifier and can be adjoined with any of them (Liu and Setiono, 1996; Blum and Langley, 1997). Additional advantages include that these methods are computationally inexpensive, fast and can also be easily applied for high-dimensional data (Saeys, Inza and Larranaga, 2007). Common disadvantages include that they usually evaluate features without accounting for feature interactions and without respect to the actual classification performance as well as their inability to detect redundant features (Kohavi and John, 1997; Saeys, Inza and Larranaga, 2007; Chandrashekar and Sahin, 2014).

Wrapper method: The wrapper method is not part of pre-processing and deploys the classification algorithm as part of feature selection, which is sometimes described as being 'wrapped around' the classifier (Liu and Setiono, 1996; Liu and Yu, 2005; Chandrashekar and Sahin, 2014). For this type of approach, multiple feature subsets are generated based on some measure, and their performance is determined via an evaluation criterion – commonly classification accuracy (Liu and Yu, 2005; Saeys, Inza and Larranaga, 2007). This follows the general idea that using the classifier with the feature subsets will provide a better indication of the classification accuracy on the feature subset than other measures and their corresponding biases (Blum and Langley, 1997). Wrapper methods are implemented as iterative processes that at each step evaluate a different feature subset (Li *et al.*, 2017). This can be achieved via (sequential) forward selection, (sequential) backward elimination or bi-directional selection (Liu and Motoda, 2001; Liu and Yu, 2005). Forward selection is an iterative procedure that starts with an empty feature set and adds in each iteration a feature, backward elimination starts with a complete feature set and iteratively removes a feature and the bi-directional selection simultaneously adds and removes features at each step (Blum and Langley, 1997; Liu and Yu, 2005). The wrapper process is terminated when a stopping criterion is met, for instance, based on classification accuracy. An example of a stopping criterion could be that the new feature subset in a step does not lead to better classification accuracy than the one in the previous step (Liu and Yu, 2005). Thus, the advantages of the wrapper approach are that it may lead to higher classification accuracies and may account for feature interactions (Saeys, Inza and Larranaga, 2007; Dessì and Pes, 2015). The main limitation of this approach is that the selection of the optimal feature subset and the corresponding performance are specific to the classifier (Saeys, Inza and Larranaga, 2007). Hence, the selected feature subset might not generally be the best feature subset for any classification algorithm (Liu and Setiono, 1996). This stands in contrast to filter methods, which produce a feature ranking independently of any classifier. Last, a common criticism of wrapper methods is their high computational complexity since they incorporate the classifier in each iteration of the algorithm (Blum and Langley, 1997; Guyon and Elisseeff, 2003; Saeys, Inza and Larranaga, 2007).

Embedded method: The term embedded method refers to learning algorithms that include feature selection in the training procedure (Guyon and Elisseeff, 2003). Thus, similar to wrapper methods, the optimal feature subset provided by such an algorithm is specific to the classifier (Saeys, Inza and Larranaga, 2007). Using such methods has the advantage that they can be more efficient and less computationally expensive than wrappers (Guyon and Elisseeff, 2003; Saeys, Inza and Larranaga, 2007). On the downside, embedded methods do not account for feature interactions, which can lead them to neglect features that are relevant in relation with other features but do not appear to be relevant just by themselves (Blum and Langley, 1997). Decision trees are a common example of embedded methods (Breiman *et al.*, 1984) since they select features to partition the feature space, implicitly determining their relevance.

For the sake of completeness it should be noted that, in addition to these three main types of supervised feature selection, there is also a hybrid form, which is commonly not mentioned as a separate type (Li *et al.*, 2017). Hybrid methods combine aspects of filter and wrapper methods to avoid part of the computational complexity of a wrapper but implement evaluation criteria from both approaches (Das, 2001; Liu and Yu, 2005). Besides that, it should be noted that the emphasis in this research is on filter methods.

In addition to the taxonomy of supervised feature selection methods into filter, wrapper and embedded methods, there exist further categorizations of these methods with respect to the process of how feature subsets are (1) generated and (2) evaluated. The first refers to the 'search strategy' (also referred to as 'generation procedure'), which is the way candidate feature subsets are generated (Dash & Liu, 1997; Liu & Yu, 2005). The search strategies are commonly distinguished into three types, as illustrated in Figure 1.3.

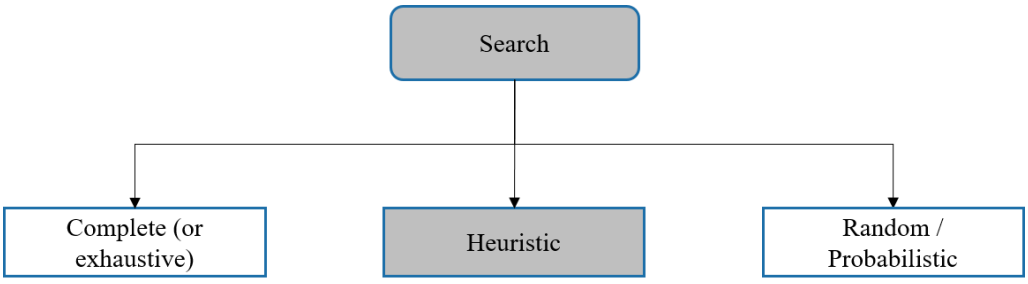


Figure 1.3: Search strategies for supervised feature selection methods

Complete (or exhaustive) search: In feature selection, there are 2^N possible feature subsets for a set of N features (Dash & Liu, 1997; Liu & Yu, 2005). Evaluating all of these feature subsets in order to find the optimal one is termed an exhaustive search. This form of search is only computationally feasible for a small number of features, is costly and, even with moderate feature set size, computationally intractable (Chandrashekar &

Sahin, 2014; Dash & Liu, 2003; Guyon & Elisseeff, 2003). Alternatively, exhaustive search guarantees the optimal feature subset (Guyon and Elisseeff, 2003; Liu and Yu, 2005). An example of an exhaustive search is the FOCUS algorithm (Almuallim and Dietterich, 1994). However, the search does not have to be exhaustive to guarantee the optimal feature subset since the search can be reduced to a certain extent to still conduct a 'complete search' that will find the optimal subset (Dash & Liu, 2003). An example of a complete but non-exhaustive search is the branch and bound algorithm (Narendra and Fukunaga, 1977).

Heuristic search: An alternative to a complete (or exhaustive) search is a heuristic search. The basic trade-off with this type of search is that the guarantee to obtain the optimal feature subset is exchanged for a (considerable) reduction in computational complexity (Bins & Draper, 2001; Dash & Liu, 2003; Liu & Yu, 2005). Hence, heuristic methods are able to provide a suggestion for a good feature subset faster (Dash & Liu, 1997). Many algorithms are heuristic, including ReliefF (Kononenko, Simec and Robnik-Sikonja, 1997) and the feature selection algorithm by Luukka (2011).

Random/Probabilistic search: The last search type is, compared to the other approaches, rather new and also embodies an attempt to reduce computational complexity by allowing the user to end up with a suboptimal feature subset (Dash & Liu, 1997; 2003). As the name suggests, it is premised on generating feature subsets randomly (according to some probability distribution) or on inserting randomness into a sequential feature subset generation approach (Liu and Yu, 2005). An example of the former is the Las Vegas filter (LVF) (Liu and Setiono, 1996) and of the latter the random-start-hill-climbing method (Doak, 1992). The advantage of random search methods is that the randomness can allow them to 'escape' from a local optimum (Liu and Yu, 2005).

The supervised feature selection methods discussed in the scope of this dissertation are limited to heuristic methods (as indicated in Figure 1.3). As mentioned previously, the search strategy determines which feature subsets are selected and evaluated. The next categorization to support the understanding of different supervised feature selection methods is according to the 'evaluation criterion' for feature subsets. Finding the (locally) optimal feature subset is always based on an evaluation criterion in relation to which the feature subset is (locally) optimal (Dash & Liu, 1997). Moreover, different criteria can lead to different feature subsets (Dash & Liu, 1997). The evaluation criteria according to Dash & Liu (1997; 2003) can be categorized into five groups (Figure 1.4).

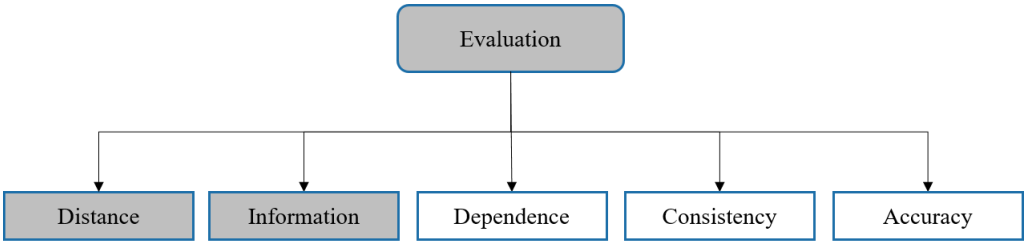


Figure 1.4: Evaluation criteria for supervised feature selection methods

Distance criteria: A distance measure (or separability-, divergence-, discrimination measure) was defined based on a simple two-class example by stating that if a feature leads to a greater difference in the class conditional probabilities than another, then it is preferred to the other feature (Dash & Liu, 1997; Liu & Yu, 2005). An example of a distance measure is the commonly used Euclidean distance (Dash & Liu, 1997).

Information criteria: An information criterion is a measure focusing on the information gain that a feature provides. This gain can be defined as the decrease in uncertainty by including a specific feature (expected posterior) compared to the case without this feature (prior). A feature that leads to a higher information gain, meaning to a higher (expected) reduction in uncertainty, is preferred to one with a lower information gain (Dash & Liu, 1997; Liu & Yu, 2005; Molina et al., 2002). An example of an information criterion is entropy (Dash & Liu, 2003).

Dependence criteria: A dependence criterion (or correlation criterion) quantifies the ability of a feature to predict the values taken by another feature or the class label (Dash & Liu, 1997; Liu & Yu, 2005; Molina et al., 2002). It is common to measure this dependence between a feature and the class and subsequently prefer features that have a stronger association with the class labels than other features (Liu and Yu, 2005). However, it is also possible to use the correlation among the features themselves to possibly determine redundant features or those features that are independent of others (Dash & Liu, 1997). A common example of a dependence criterion is correlation (Hall, 2000). Dash & Liu (1997; 2003) mention that dependence criteria could also be divided into distance and information criteria but are kept as a separate type. This aspect also indicates that evaluation criteria for a supervised feature selection method, such as the dependence criterion, do not necessarily only embody characteristics of one evaluation criterion.

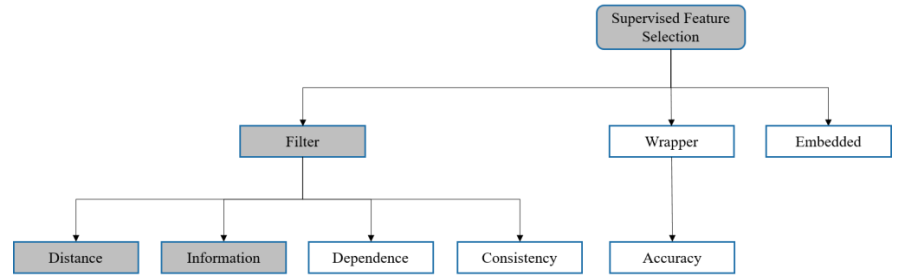
Consistency criteria: A consistency criterion uses the so-called ‘MIN-FEATURES bias’ (Almuallim and Dietterich, 1991, 1994). This bias states, ‘if two functions are consistent with the training examples, prefer the function that involves fewer input features’ (Almuallim and Dietterich, 1991). Essentially, a feature subset that is ‘consistent’ with the classes is selected that uses the smallest number of features. In this context, the term ‘consistency’ means that no two observations that share the same values for all features

belong to different classes (Arauzo-Azofra, Benitez and Castro, 2008). In simple terms, two observations cannot be the same in terms of all features but have different class labels. The latter is defined as an 'inconsistency' (Molina, Belanche and Nebot, 2002; Liu and Yu, 2005). Since zero inconsistencies may not be achievable in a certain dataset, a user-specified acceptable inconsistency rate can be applied, based on which the corresponding minimum feature subset is determined (Dash & Liu, 1997).

Accuracy criteria: An accuracy criterion (or error criterion) is simply the classification accuracy (or probability of error) that is used to evaluate feature subsets (Dash & Liu, 1997; Molina et al., 2002). It is obvious that a feature subset leading to a higher classification accuracy (or a lower error probability) will be preferred to other feature subsets.

In addition to the type of evaluation criterion, it is crucial to distinguish between an 'univariate' and a 'multivariate' evaluation of feature subsets. Univariate methods consider each feature separately, implicitly assuming the conditional independence of the features, whereas multivariate methods account for such dependencies when evaluating features (Martínez Sotoca and Pla, 2010; Dessì and Pes, 2015). Univariate feature selection methods have the advantage of often being intuitive, easily interpretable and computationally inexpensive compared to multivariate methods (Saeys, Inza and Larranaga, 2007). However, univariate methods neglect feature interactions, which can lead to worse classification accuracies for the corresponding feature subsets (Saeys, Inza and Larranaga, 2007; Martínez Sotoca and Pla, 2010). In contrast, multivariate feature selection methods account somehow for interactions among features, where univariate methods fail to incorporate such (Robnik-Šikonja and Kononenko, 2003; Saeys, Inza and Larranaga, 2007). However, multivariate feature selection methods are not always necessary. In the majority of real-world cases Kononenko, Simec and Robnik-Sikonja (1997) tested, there was only an insignificant difference in the results between multivariate and univariate (= myoptic) filter methods. Notwithstanding, they state that multivariate feature selection methods are useful if it is not known whether considerable conditional dependencies exist among features in the data (Kononenko, Simec and Robnik-Sikonja, 1997).

Combining the different categorizations of type, search strategy and evaluation criterion for supervised feature selection methods, a two-dimensional taxonomy for supervised feature selection methods can be obtained that includes selected examples of algorithms that fall into certain categories (Figure 1.5). It is noteworthy that this taxonomy is two-dimensional since the type of supervised feature selection and evaluation criterion are closely related. In particular, the first four types of evaluation criteria are exclusively found in filter methods, whereas accuracy is the preferred evaluation criterion for wrapper methods (Kohavi and John, 1997; Liu and Yu, 2005).



Complete	Branch & Bound (Narendra and Fukunaga, 1977) Best First Strategy (Xu, Yan and Chang, 1998)			FOCUS (Almuallim and Dietterich, 1994) ABB Branch & Bound (Liu, Motoda and Dash, 1998)	Beam Search (Doak, 1992) Smart Beam Search (Gupta, Doermann and DeMenthon, 2002)	
Heuristic	Relief (Kira and Rendell, 1992b) ReliefF (Kononenko, Simec and Robnik-Sikonja, 1997) **COLD** (Publication **IV**) Laplacian Score (He, Cai and Niyogi, 2005) Fisher Score (Duda, Hart and Stork, 2012)	FS Luukka (2011) **FSAE** (Publication **I**) **Nonspecificity** (Luukka & Lohrmann, 2019) Information Gain (Hall and Holmes, 2003)	Correlation-based Feature Selection (Hall, 2000) Mutual Information (Battiti, 1994) Symmetrical Uncertainty (Witten and Frank, 2005; Breiman *et al.*, 1984) Gain Ratio (Karegowda, Manjunath and Jayaram, 2010)	Set Cover (Dash, 1997)	BSE-SLASH (Caruana and Freitag, 1994a) Importance Score (Vafaie and Imam, 1994) Bidirectional Search (Doak, 1992)	Decision Tree (Breiman *et al.*, 1984) Random Forest (Breiman, 2001)
Random				Las Vegas Filter (Liu and Setiono, 1996)	Random-Start Hill-Climbing (Doak, 1992) Genetic Algorithm (Vafaie and Imam, 1994)	

Figure 1.5: Taxonomy of supervised feature selection, search strategy and evaluation criteria [modified from Liu & Yu (2005)]

As indicated in Figure 1.5 (in grey), the emphasis in this dissertation is on heuristic filter methods using distance or information criteria to conduct supervised feature selection. These two subcategories were selected since using distance and information criteria is comparably simple but often yields good results and is a popular approach to supervised filter methods. Moreover, the feature selection algorithm by Luukka (2011) was the starting point of this research in feature selection, which is an information-theoretic supervised filter method. This dissertation does not cover dependency- and consistency-based filter methods.

The research in this dissertation focuses on the development and improvement of feature selection methods (**Publications I, IV**) and the application of feature selection on real-world data (**Publication II**). However, it also covers a part on classification (**Publication III**) since the idea underlying the modification of the discussed classifier is subsequently deployed in one of the feature selection algorithms (**Publication IV**). An overview of the objectives and contents of the publications is presented in Table 1.1 for the reader's convenience.

Publication Number	I	II	III	IV
Publication Name	'A combination of fuzzy similarity measures and fuzzy entropy measures for supervised feature selection'	'Classification of intraday S&P500 returns with a Random Forest'	'A novel similarity classifier with multiple ideal vectors based on k-means clustering'	'Using clustering for supervised feature selection to detect relevant features'
Purpose/Objective	To highlight the vulnerability of the feature selection algorithm by Luukka (2011), to develop an improved version termed fuzzy similarity and entropy (FSAE) feature selection to address these vulnerabilities and to demonstrate the ability of the improved filter method to find subsets of relevant features more effectively than the original algorithm	To develop a machine learning approach using feature selection and classification to predict the S&P500 intraday return, to introduce a simple trading strategy based on the classification model and to demonstrate the ability of this strategy to outperform a buy-and-hold strategy after transaction cost	To develop an extension to the similarity classifier that deploys clustering to find multiple ideal vectors for classification and to demonstrate the ability of this algorithm to outperform the original similarity classifier and achieve competitive results to the benchmark classification algorithms	To develop a novel supervised feature selection algorithm termed 'clustering one less dimension' (COLD) and to demonstrate its ability to cope effectively with certain complex data structures and to find subsets of relevant features that lead to competitive results to the benchmark filter methods
Problem Type	Feature selection	Feature selection and classification	Classification	Feature selection
Problem Mode	Supervised	Supervised	Supervised	Supervised
Type of Algorithm	Univariate filter/wrapper (heuristic, information criterion)	Univariate filter and ensemble classifier	Single classifier	Multivariate filter (heuristic, distance criterion)
Data	Three simple artificial datasets, five real-world medical datasets	Real-world financial stock market dataset	Three artificial datasets, three real-world financial datasets	Four artificial datasets, two real-world medical datasets

Table 1.1: Overview of the publications and their objectives

The structure of the dissertation as well as the objective and content of each section are summarized in Table 1.2.

Section	Content	Related Publication(s)
1 Introduction	▪ Theoretical background ▪ Taxonomy of feature selection ▪ Scope of research ▪ Summary of objectives of each publication	
2 Related Methods	▪ Introduction to the similarity classifier ▪ Introduction to several distance- and information-based heuristic filter methods for feature selection (univariate and multivariate)	Publications I, III, IV
3 Fuzzy Similarity and Entropy (FSAE) Feature Selection	▪ Emphasis on vulnerabilities of the feature selection by Luukka (2011) [Research need] ▪ Introduction of the univariate filter FSAE ▪ Application of feature selection algorithms to artificial and real-world data	Publications I, II
4 Similarity Classifier with Multiple Ideal Vectors	▪ Emphasis on a common deficiency of distance-based classifiers [Research need] ▪ Introduction of the similarity classifier with multiple ideal vectors that deploys clustering and two different forms of pre-processing ▪ Application of classification algorithms to artificial and real-world data	Publication III
5 Clustering One Less Dimension (COLD) Feature Selection	▪ In-depth discussion of univariate vs multivariate filter methods ▪ Introduction of the multivariate distance-based heuristic filter method COLD using K-medoids clustering ▪ Application of feature selection algorithms to artificial and real-world data	Publication IV
6 Conclusion, Limitations and Future Work	▪ Summary of the contributions of each publication ▪ Limitations of the suggested algorithms and findings ▪ Suggestions for future work	Publications I–IV

Table 1.2: Structure of the dissertation

The subsequent section will focus on the introduction of the similarity classifer and the related filter methods for feature selection. The similarity classifier is included because it is linked to the feature selection approach by Luukka (2011), discussed afterwards. Selected filter methods are depicted since they belong to the same categories (distance- and information-based heuristic filter methods) as the filter methods presented in this dissertation. Hence, they function as suitable benchmark algorithms for the artificial and real-world examples deployed in this research.

2 Related Methods

2.1 Similarity Classifier

The similarity classifier is a classification method, which is a form of supervised learning. The term 'supervised' refers to the fact that the targets, meaning in classification the class labels of observations, are known (Webb, 2002; Bishop, 2006). Classification refers to machine learning problems in which observations (synonymic: samples) have to be assigned to classes (Bishop, 2006). The assignment is conducted based on the feature values that characterize an observation. A very simple example is presented in Duda, Hart and Stork (2012), where fishes should be assigned to one of two classes, 'Bass' or 'Salmon', premised on their length (first feature) and colour (second feature). The notion of similarity is rooted in fuzzy set theory. In contrast to crisp sets, fuzzy sets do not simply indicate whether a property is present or not but also allow the modelling of a membership degree (Zadeh, 1965; Klawonn and Castro, 1995). The strength of the membership is expressed as a real number in the compact interval $[0,1]$ (Zadeh, 1965, 1971). The similarity relation can be regarded as a generalization of the equivalence relation (Zadeh, 1971). So, the notion of the fuzzy equivalence relation can be used to formalize similarity (Klawonn and Castro, 1995). The similarity classifier is based on the equivalence relation of the generalized Łukasiewicz structure to define the membership or similarity of objects (Luukka, Saastamoinen, & Könönen, 2001). The equivalence of two objects *a* and *b* can be expressed in the generalized Łukasiewicz structure as follows:

$$a \leftrightarrow b = \sqrt[p]{1 - |a^p - b^p|}, \tag{2.1}$$

where the parameter p comes from the generalized Łukasiewicz structure (Luukka, Saastamoinen, & Könönen, 2001). The idea to use generalized Łukasiewicz-valued similarity takes an important role in the similarity classifier. Let us assume a dataset denoted X containing n observations of D features (from $d = 1, 2, \dots, D$). The observations belong to one of N classes (from $i = 1, 2, \dots, N$), and the class label for each observation is known. The setup of the classifier is based on the observations in the training set. The step-by-step algorithm underlying the similarity classifier is presented below.

The first step is the scaling of the data into the compact interval $[0,1]$ and the calculation of a so-called 'ideal vector' for each class. Such an ideal vector is supposed to represent a class well (Luukka et al., 2001). This means that for each feature it contains a value that is supposed to embody the values that this feature takes for a given class. The ideal vector for a class can, for instance, be calculated simply with an arithmetic mean, generalized mean, Bonferroni mean or OWA operator (Kurama, Luukka, & Collan, 2016; Luukka & Kurama, 2013; Luukka & Leppälampi, 2006). Using the generalized mean, the ideal vector element for class *i* for a feature *d* can be calculated as:

$$v_{i,d} = \left(\frac{1}{n_i} \sum_{x \epsilon X_i} {x_d}^m \right)^{\frac{1}{m}}, \tag{2.2}$$

where X_i is the subset of observations in X that belongs to class i, n_i is the number of observations in that class and m is the generalized mean parameter. If $m = 1$, then the result is simply the arithmetic mean. The ideal vector for class i is the vector of all ideal vector elements for all D features and can be denoted as $v_i = (v_{i,1}, v_{i,2}, \dots v_{i,D})$. At the end of this step, the ideal vector for each class in the dataset was calculated.

The second step is to calculate the similarity S of a new observation x_j with the ideal vectors of each of the N classes. The similarity values are generalized Łukasiewicz-valued and are computed for each feature separately and then aggregated by using a mean or weighted mean function. The similarity of observation x_j with ideal vector v_i can be expressed as (Luukka and Leppälampi, 2006):

$$S(x_j, v_i) = \left(\frac{1}{D} \sum_{d=1}^{D} \left(\sqrt[p]{\left(1 - \left| x_{j,d}^p - v_{i,d}^p \right|\right)} \right)^m \right)^{\frac{1}{m}} \tag{2.3}$$

The similarity expresses the strength of the membership of observation x_j to class i. As for the calculation of ideal vectors, different mean functions can also be deployed to aggregate the similarity values over all D features. Here, a simple arithmetic mean is applied.

The third step is the assignment of each observation to the class it is most similar to. Since each class is represented by an ideal vector, the observation is assigned to the class to whose ideal vector it is most similar or, in other words, has the highest membership degree (Luukka et al., 2001).

An advantage of the similarity classifier is that the steps underlying this algorithm are overall quite simple. Moreover, it requires only a few observations to accomplish high classification accuracies and is also computationally inexpensive compared to many other classification algorithms (Luukka, 2008). In addition, several applications on real-world datasets have demonstrated its ability to achieve high classification accuracies (Luukka, 2008; Luukka & Leppälampi, 2006; Luukka, 2010).

2.2 Selected Feature Selection Methods

2.2.1 ReliefF

The ReliefF algorithm is an extension of the popular filter feature selection (and feature weighting) approach called the Relief algorithm introduced by Kira and Rendell (1992a).

The Relief algorithms belong to the most successful filter methods (Dietterich, 1997). It is noteworthy that the original Relief algorithm was limited to binary class problems. Relief is premised on the idea that for each observation, the closest observation from the same class (near-hit) and from the other class (near-miss) can be used to determine each feature's relevance (Kira and Rendell, 1992b). In particular, for a certain feature, the difference in the distance of the near-hit from an observation to the distance of the near-miss with the same observation is calculated. This calculation is repeated for each considered observation (with at most all observations) and then averaged to constitute the weight of the feature, where higher weights indicate more important features (Kira and Rendell, 1992b). It is apparent that the feature weight following this approach will be higher if the average difference of the near-misses (other class) is higher than the near-hits (same class). Using this approach allows an accounting for conditional dependencies among features (Kononenko, Simec and Robnik-Sikonja, 1997). The reason for this is that the near-hits and near-misses are determined based on the distance between observations accounting for all features. Afterwards, the contribution to the feature weight is computed premised solely on the distance of the near-hits and near-misses to an observation. In this way, the selection of near-hits and near-misses accounts for conditional dependencies among features, even though the subsequent evaluation is conducted for only one feature. This makes Relief (and also the extension ReliefF) a multivariate, distance-based filter method. Hence, Relief (and ReliefF) can address data structures with conditional dependence among features (e.g. the XOR problem) that simple dependence-based methods cannot account for (e.g. Gini-index, gain ratio)[1] (Kononenko, Simec and Robnik-Sikonja, 1997). Therefore, already Relief is regarded as a powerful algorithm for weighting and ranking features (Kononenko, Simec and Robnik-Sikonja, 1997). One of the main drawbacks of Relief is that it is limited to two-class problems (Kononenko, 1994). To address this point and other weaknesses of Relief, Kononenko (1994) introduced ReliefF, which can cope with multiple classes and performs better in the presence of noisy and incomplete data. The idea behind the extension remains the same as in ReliefF but with certain additions that will be highlighted in the algorithm's step-by-step description below (Robnik-Šikonja and Kononenko, 2003). Let us assume a dataset denoted X containing n observations of D features (from $d = 1, 2, \dots, D$).

In the **first step** the weights for each feature, W, which represent each feature's relevance, are initialized to zero. The weight for the d-th feature is denoted by W_d. Moreover, it is decided for how many observations, denoted m, the algorithm is run (with $m \leq n$). More observations m will lead to a more reliable feature ranking. Hence, for datasets not too large, it is suggested to carry out ReliefF with all observations (Kononenko, Simec and Robnik-Sikonja, 1997).

[1] Kononenko, Simec and Robnik-Sikonja (1997) referred to these approaches as 'myoptic' (which is synonymic to the word 'shortsighted'), stating that they cannot take context into account such that (local) dependencies remain hidden to them.

In **the second step**, a single observation x_i is randomly selected and the near-hits (for the same class) h_j and near-misses (for each other class) m_j are calculated (with $j = 1\ to\ k$). In contrast to Relief, where only a single near-hit and near-miss are deployed, in ReliefF, multiple near-hits and near-misses are used to make the feature ranking more reliable when noisy data are present (Kononenko, Simec and Robnik-Sikonja, 1997). For the given observation x_i, the contribution to the weight of each of the features is calculated. For the *d*-th feature, the weight W_d is updated as (Robnik-Šikonja and Kononenko, 2003):

$$W_d = W_d - \sum_{j=1}^{k} \frac{diff(d, x_i, h_j)}{m * k} + \sum_{C \neq class(x_i)} \frac{\left[\frac{P(C)}{1 - P(class(x_i))} \sum_{j=1}^{k} diff(d, x_i, m_j(C))\right]}{m * k}, \tag{2.4}$$

where *k* is the number of near-hits and near-misses used, *m* is the number of observations considered for ReliefF, $P(class(x_i))$ is the probability of the class that observation x_i belongs to, *C* is the set of all other classes and $P(C)$ is the probability of each of these classes. In addition, $diff(d, x_i, h_j)$ is the difference between observation x_i and the near-hit h_j (from the same class), and $diff(d, x_i, m_j(C))$ is the difference between observation x_i and the near-miss m_j of one of the classes in *C*.

The difference $diff(d, x_i, h_j)$ and $diff(d, x_i, m_j)$ for an observation x_i with the near-hit h_j and near-miss m_j is calculated differently for categorical and numerical variables. For categorical features, the difference is calculated according to Equation (2.5) and for numerical features as in Equation (2.6) (Robnik-Šikonja and Kononenko, 2003).

$$diff(d, x_i, x_j) = \begin{cases} 0\ if\ x_{i,d} = x_{j,d} \\ 1\quad otherwise \end{cases} \tag{2.5}$$

$$diff(d, x_i, x_j) = \frac{|x_{i,d} - x_{j,d}|}{\max(x_d) - \min(x_d)} \tag{2.6}$$

The basic difference between these two approaches is that for categorical features, only the fact is measured whether the feature value is equal or unequal. For numerical features, it is measured how far the values of the *d*-th feature are for two instances x_i and x_j from each other, normalized to the unit interval [0,1]. According to Robnik-Šikonja and Kononenko (2003), the same normalized distance is used via averaging over all features *D* to calculate the near-hits and near-misses. In the original Relief algorithm, the Euclidean distance was used for this purpose (Kira and Rendell, 1992b), which is not normalized.

The formula in Equation (2.4) essentially adjusts the weights W_d for the near-hits and near-misses. The weight is reduced by the average difference to the near-hits. This means that if observations from the same class as the current observation are far from that observation, this is regarded as negative for feature importance. The intuition behind this is that observations of the same class being close is desirable and indicates a clear pattern for the class. In contrast, the weight is increased by the weighted average distance of the near-misses of the other classes *C*. Hence, it has a positive effect on the feature importance score if observations from other classes are distant from the current observation. A high distance from the observations of other classes indicates that the observation clearly belongs to its class, and it is apparent that it is different to representatives of the remaining classes. Overall, the weight is adjusted in favour of the weighted average difference to the near-misses from other classes and faces a decrease from the average difference of near-hits that belong to the same class. Thus, if the closest observations of other classes are more distant than the closest observations of the same class (for a specific feature), this represents a positive contribution overall to the weight of the feature. The differences to the near-misses m_j is weighted by the share of the probability of each of the classes *C*. This probability is divided by the probability of all other classes $(1 - P(class(x_i)))$ which the given observation x_i does not belong to. The calculation of W_d is conducted for all $d = 1\ to\ D$, meaning for all features in the data.

It is noteworthy that in Equation (2.4) the division by $m * k$, which leads to an averaging over all *m* and *k* for each weight, assumes that each observation *m* and each near-hit and near-miss for an observation x_i are equally important. For RRelief, the Relief algorithm extension for regression, and ReliefF, Robnik-Šikonja and Kononenko (2003) introduce the possibility to adjust the latter. This means that the rank of different near-misses and near-hits affects their influence on the weight for observation x_i. Closer near-misses and near-hits (meaning higher-ranked near-misses and near-hits) have a larger impact. In contrast, the influence of near-misses and near-hits decreases exponentially if they are further away. Hence, instead of using *k* in the denominator of Equation (2.4), which is equivalent to multiplying by $\frac{1}{k}$, the new multiplication factor can be stated in the following (simplified) formula (Robnik-Šikonja and Kononenko, 2003):

$$d_{i,j} = \frac{e^{-\left(\frac{rank(x_i,x_j)}{\sigma}\right)^2}}{\sum_{j=1}^{k} e^{-\left(\frac{rank(x_i,x_j)}{\sigma}\right)^2}} \tag{2.7}$$

The $rank(x_i, x_j)$ is the rank (or position) of x_j in the sequence of all instances that is ordered by their distance from x_i (Robnik-Šikonja and Kononenko, 2003). The parameter σ is the decay factor that affects the calculation of the distance.

When all near-hits and near-misses are equally important (so when in Equation (2.4) *k* is used in the denominator instead of the result from Equation (2.7)), a common choice for the number of near-hits and near-misses is 10 (Kononenko, 1994; Kononenko, Simec and

Robnik-Sikonja, 1997). Opposed to that, when different weights are given to near-misses and near-hits according to their rank (see Equation (2.7)), a suggested setup contains 70 nearest neighbours and $\sigma = 20$ (Robnik-Šikonja and Kononenko, 2003).

The **third step** encompasses the repetition of the second step until the contribution of all *m* observations to the weight for each feature is calculated. Subsequently, a weight for each feature will be obtained that is $\epsilon[-1,1]$, with higher weights indicating more relevant features (Kononenko, Simec and Robnik-Sikonja, 1997).

Relevant features are expected to have weights larger than zero, whereas irrelevant or even noisy features should end up with negative weights or a weight close to zero (Kira and Rendell, 1992b). After that, a user-specified number of features with the highest weights can be selected or a weight threshold can be used so that only features above that threshold are included in the feature subset (Kira and Rendell, 1992b; Robnik-Šikonja and Kononenko, 2003; Souza, Matwin and Japkowicz, 2006). The selection of the threshold value is one of the drawbacks of Relief(F) (Chandrashekar and Sahin, 2014) as well as its inability to find redundant features, which can result in a feature subset containing redundant variables (Liu and Setiono, 1996; Bins and Draper, 2001).

2.2.2 Fisher Score

The Fisher score is a simple supervised filter method for feature selection (Duda, Hart and Stork, 2012). The objective behind the Fisher score is to select features for which the distance between observations of different classes is large and the distance from observations in the same class is small (Gu, Li and Han, 2011; Li *et al.*, 2017). Since this approach considers features only in isolation and no dependencies among features, it is a univariate, distance-based feature selection method. Conceptually, the idea underlying the Fisher score is implemented by measuring how distant the class means for a feature are from the feature mean and dividing this value by the variation for the feature in that class. Accordingly, the Fisher score for a feature *d* is defined as:

$$Fisher\ Score_d = \frac{\sum_{i=1}^{N} n_i \left(\bar{x}_{i,d} - \bar{x}_d\right)^2}{\sum_{i=1}^{N} n_i \sigma_{i,d}^2}, \tag{2.8}$$

where n_i is the number of observations contained in class *i*, $\bar{x}_{i,d}$ is the mean value of the *d*-th feature for class *i*, $\sigma_{i,d}$ is the corresponding standard deviation for the values feature *d* takes for observations in class *i* and $\bar{x}_d$ is the mean of the *d*-th feature in the entire dataset. Following the objective of the Fisher score, high feature scores will be obtained if the class means for a feature *d* are far from the mean of the feature $\bar{x}_d$ while the variance of observations in each class for feature *d* is small. The mean differences and variances of the classes are weighted by the number of observations n_i in each of these classes. It is apparent that the difference between the class means to the feature mean is only large when the class means are on average distant from the feature mean, meaning that the feature mean does not represent the values for feature *d* in the classes well. This is

obviously positive for the discrimination achieved among the classes. In addition, the variance is included since features for which the classes are distant from each other and that have a low variance are likely far from each other and, hence, well discriminating the classes. In contrast, if the class means for a feature are on average distant from the feature mean, but there is high variance in each class, this may likely lead to an overlap of the classes, which results in a lower Fisher score. Hence, the Fisher score reflects the relevance of features. Finally, the *k* features with the highest Fisher score should be selected for the feature subset (Li *et al.*, 2017).

2.2.3 **Laplacian Score**

The Laplacian score is a filter method that can be deployed for supervised and unsupervised machine learning tasks (He, Cai and Niyogi, 2005). The explanation in this section will focus on the supervised version of the Laplacian score. This heuristic distance-based filter method aims to rank features according to their locality-preserving power (He, Cai and Niyogi, 2005). The underlying supervised method can be represented in the form of a step-by-step algorithm with the following three distinct steps:

The **first step** consists of the calculation of an affinity matrix *S* of dimension $n * n$, where *n* is the number of observations in the data. The affinity matrix *S* adheres to the following rule for two observations x_i and x_j (He, Cai and Niyogi, 2005; Li *et al.*, 2017):

$$S_{i,j} = \begin{cases} e^{-\frac{\|x_i - x_j\|^2}{t}} & if\ c_i = c_j \\ 0 & if\ c_i \neq c_j \end{cases}, \tag{2.9}$$

where c_i is the class of observation x_i and c_j the class of observation x_j, respectively. The value for parameter *t* is a 'suitable' constant (He, Cai and Niyogi, 2005). The affinity (or weight) matrix *S* represents the local structure of the data (He, Cai and Niyogi, 2005). If two examples belong to the same class, then a weight/affinity value that represents the similarity of these examples is calculated. If they do not belong to the same class, then the weight is set to zero.

In the **second step** the diagonal matrix *D* is computed, for which off-diagonal elements are zero, and the diagonal element for an observation x_i is determined as (Li *et al.*, 2017):

$$D_{i,i} = \sum_{j=1}^{n} S_{i,j} \tag{2.10}$$

This means that, for instance, the *i*-th element on the diagonal of *D* is the sum over all columns *j* for the weights in the *i*-th row in the affinity matrix *S*. Based on the affinity matrix *S* and the diagonal matrix *D*, the graph Laplacian (or Laplacian matrix) can be calculated as $L = D - S$ (He, Cai and Niyogi, 2005; Li *et al.*, 2017).

In the **third step** the Laplacian score for a feature d is calculated according to the following formula (He, Cai and Niyogi, 2005):

$$L_d = \frac{\tilde{x}_d^T L \tilde{x}_d}{\tilde{x}_d^T D \tilde{x}_d} \quad where \quad \tilde{x}_d = x_d - \frac{x_d^T D 1}{1^T D 1} 1\,, \tag{2.11}$$

where x_d denotes the d-th column of the dataset, T in the exponent the transpose and 1 a column vector of ones $[1,1,\dots,1]^T$. Even though it is not directly apparent from Equation (2.11), the Laplacian score L_d is related to the Fisher score and attempts to minimize the difference between feature values of observations of the same class, taking into account the multidimensional similarity of the observations and scaling the corresponding value by the variance (He, Cai and Niyogi, 2005). After the Laplacian score for each feature is determined, the user-specified k features with the smallest Laplacian scores are selected for the feature subset (Li *et al.*, 2017). Since the calculation of the affinity matrix S that is deployed in the Laplacian score uses the similarity between observations accounting for all features, the method can be regarded as a multivariate filter. In this aspect, it is related to ReliefF, where the features are evaluated by themselves, but since the distance to the near-hits and near-misses also accounts for all features, the method is overall multivariate.

2.2.4 Feature Selection by Luukka (2011)

The feature selection method introduced by Luukka (2011) is a heuristic method that uses similarity as well as fuzzy entropy measures. In the paper by Luukka (2011), the entropy measure developed by De Luca and Termini (1972) and the two entropy measures from Parkash, Sharma and Mahajan (2008) are deployed. De Luca and Termini (1972) depicted fuzzy entropy as a 'measure of the degree of fuzziness' that provides insight into the average information that is present in the data. The entropy measure by De Luca and Termini is defined as:

$$H(A) = -\sum_{j=1}^{n} \left[\mu_A(x_j)\log \mu_A(x_j) + \left(1-\mu_A(x_j)\right)\log\left(1-\mu_A(x_j)\right)\right], \tag{2.12}$$

where $\mu_A(x_j)$ stands for the membership degree of x_j to the fuzzy set A, which is within the compact interval $[0,1]$. The other two entropy measures were develped by Parkash, Sharma and Mahajan (2008) and are related to the idea of weighted entropy (Belis and Guiasu, 1968). They are defined as:

$$H^1(A) = \sum_{j=1}^{n} w_j \left[sin\frac{\pi\mu_A(x_j)}{2} + sin\frac{\pi(1-\mu_A(x_j))}{2} - 1\right] \tag{2.13}$$

$$H^2(A) = \sum_{j=1}^{n} w_j \left[cos\frac{\pi\mu_A(x_j)}{2} + cos\frac{\pi(1-\mu_A(x_j))}{2} - 1 \right] \quad (2.14)$$

Two key properties of entropy measures for inputs $\epsilon[0,1]$ are that the maximum entropy is reached for an input value of 0.5 and that input values of '0' and '1' both yield an entropy of zero (De Luca and Termini, 1972). This idea is visually represented in Figure 2.1.

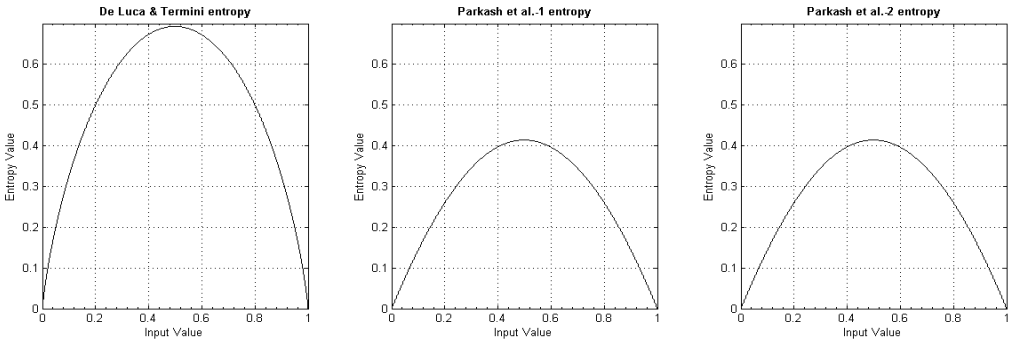


Figure 2.1: Different entropy measures [reproduced from Publication **II** with the permission of the publisher]

A high entropy value can indicate randomness and a low level of informativity in the data, whereas a small entropy value can signal structure and a high level of informativity (Yao, Wong and Butz, 1999). Luukka (2011) suggested combining a similarity measure as the input to an entropy measure. In general, to combine entropy with the similarity values for features is intuitive. A similarity close to '1' indicates that two objects are very similar, and a similarity of close to '0' highlights that the objects are very dissimilar. It is obvious to regard this as informative since there is no, or close to no, ambivalence in these similarity values. The objects are in the first case clearly highly similar and in the second case highly dissimilar. Therefore, the corresponding entropy value is close to zero and indicates informativity. In contrast, a similarity value close to 0.5 indicates ambivalence since the two objects are neither really dissimilar nor can they be considered very similar. On account of this, the corresponding entropy value is high and emphasizes a low informativity of such a similarity value. For each observation, the entropy is calculated for all similarity values between the feature values of an observation with the feature values of the ideal vectors. They are then summed for each feature. The outcome for each feature will be a sum of entropy values for that feature over all observations and all ideal vectors.

Being consistent in its meaning, a low sum of entropy values indicates that the feature values of observations are informative and that the similarities on average tend to be far from 0.5. Such a feature should be retained given that it is characterized on average with

low ambivalence and uncertainty. Opposed to that, a high sum of entropy values represents low informativity, and the similarity values used as input are on average closer to 0.5, indicating ambivalence of the observations' memberships. In other words, features with high summed entropy values are considered uninformative and potentially random. Hence, they constitute candidates for feature removal. Thus, the idea to combine an entropy and a similarity measure to determine relevant features in a dataset was developed (Luukka, 2011). In his paper, Luukka (2011) implemented this approach as a heuristic filter method. The step-by-step process underlying the approach is illustrated in Figure 2.2.

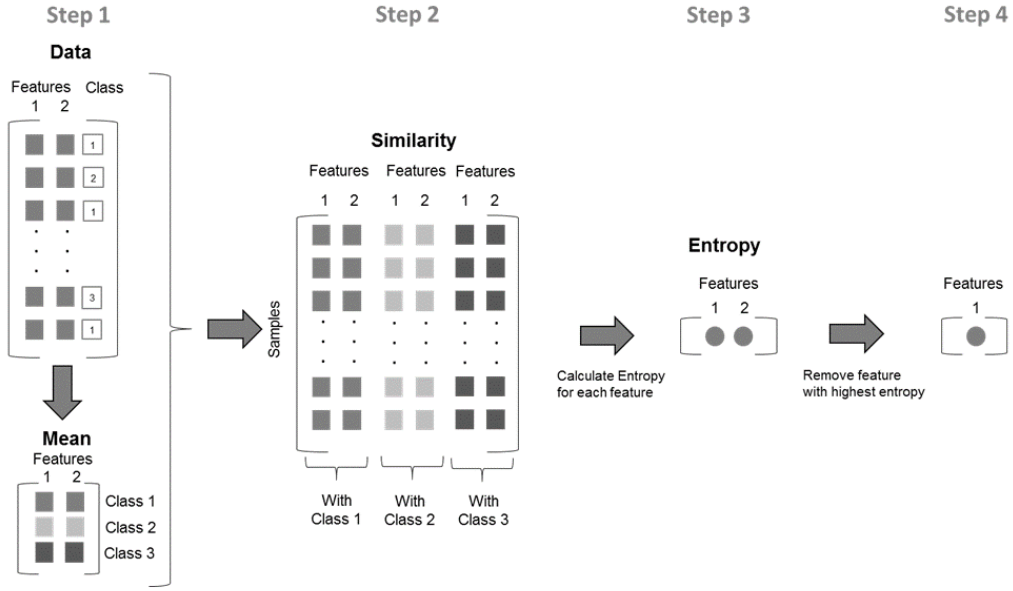


Figure 2.2: Step-by-step process underlying the feature selection method by Luukka (2011) for a two-dimensional example with three classes [reproduced from Publication **I** with the permission of the publisher]

The first step is centred on the calculation of the ideal vectors of each class. This can be conducted the same way as in the similarity classifier with a generalized mean (Luukka, 2011; Publication **I**).

The second step consists of the calculation of the similarity values *S* of the observations with the ideal vectors for each feature *d* (Luukka, 2011).

$$S\left(x_{j,d}, v_{i,d}\right) = \sqrt[p]{\left(1 - \left|x_{j,d}^{p} - v_{i,d}^{p}\right|\right)}, \tag{2.15}$$

where j denotes the observation (from $j = 1,2, \ldots n$), d the feature (from $d = 1,2, \ldots D$) and i the class (from $i = 1,2, \ldots N$). This equation is similar to Equation (2.3) but does not average over all D features. Since the similarity is calculated for each observation j for all features d with ideal vectors of all classes i, the resulting matrix of similarities is *n*(DN)* (Luukka, 2011; Publication **I**). This matrix is illustrated in Figure 2.2 under 'Step 2'.

The third step encompasses the calculation of the entropy values. An entropy value is calculated for each element in the *n*(DN)* matrix mentioned in the previous step and, subsequently, summed over all observations and classes.

$$H_d = \sum_{i=1}^{N} \sum_{j=1}^{n} H\left(S\left(x_{j,d}, v_{i,d}\right)\right) \tag{2.16}$$

The result is a summed entropy value for a feature d. This calculation is repeated for each feature in order to obtain D summed entropy values.

The fourth step is simply the selection of a user-specified number of features with the lowest entropy values to keep for the subsequent analysis.

The difficulty in following this procedure, as with many filter methods, is that there is no clear threshold value after which features can be removed without the chance that they are actually relevant. In contrast, removing too few features will leave irrelevant or even noisy features in the feature subset. One way to address this problem is by using the idea underlying the filter method by Luukka (2011) in the form of a wrapper approach, as presented in Publication **I**. The wrapper version of the approach by Luukka (2011) uses a similarity classifier as the evaluation criterion. In the first step, this requires dividing the data into training and test data. The first three steps are now conducted exclusively on the training data. After the feature ranking based on the training data is obtained, the feature with the lowest rank (and the highest summed entropy value) is suggested for removal from the feature set. Subsequently, the classification accuracy on the test set before and after the removal is evaluated with a classifier (such as the similarity classifier). The change in the classification accuracy can function as a 'stopping criterion' for this wrapper. If the classification accuracy after the removal of a feature remains unchanged or increases, the feature can be considered irrelevant. Features that are not relevant for the discrimination between the classes in the dataset are by definition irrelevant. If the mean accuracy after the feature removal did not deteriorate, the feature is removed, and all steps will be repeated another time. If, however, the 'stopping criterion' is met since the performance on the test set after the feature removal is worse than before the feature removal or has deteriorated more in performance than a user-specified threshold, the algorithm is stopped, and only the features up to the previous step are removed from the dataset.

The original pseudocode for this feature selection method is available in Luukka (2011).

3 Fuzzy Similarity and Entropy (FSAE) Feature Selection

3.1 Vulnerabilities of the Feature Selection Method by Luukka (2011)

The feature selection algorithm developed and introduced by Luukka (2011) has demonstrated on four medical datasets the ability to remove features that are not relevant and useful for classification and even improve the classification accuracy in the majority of them (Luukka, 2011). Hence, the algorithm has revealed the ability to successfully conduct feature selection on these real-world datasets. In the following, the assumptions underlying this algorithm, which characterize its ability to find relevant features, will be examined in more detail. In particular, these assumptions will be analysed with respect to vulnerabilities that the algorithm can be susceptible to.

One main assumption of the algorithm is that one ideal vector per class, so essentially a single point in the feature space, is sufficient to characterize a class well. This assumption is the same as in the similarity classifier (Luukka et al., 2001). However, it can pose a problem for more complex data structures, where classes are composed of groupings/clusters. It is clearly intuitive that multiple disjoint groups cannot be represented well by a single point. However, it requires additional steps and computations to determine a suitable number of class representatives. Moreover, for less complex data structures, a single ideal vector is likely sufficient to represent each class. Thus, using a single ideal vector per class is a trade-off between the ability to capture multiple decision regions and the computational complexity of the algorithm.

The second main assumption concerns the use of similarity as the input to the entropy measure. Recalling the shape of entropy functions (see Figure 2.1), inputs of close to 0 and 1 take low entropy values, whereas the closer the input value is to 0.5, the higher the entropy associated with it. Since low entropy values indicate informativity, input values of close to 0 and 1 are more informative than values closer to 0.5. Hence, if similarity values are the input to the entropy function, this assumes that similarity values close to 0 and 1 are regarded as informative. This appears plausible when the feature of an observation has a high similarity value to its own class's ideal vector element and a low similarity value to the competing class's ideal vector element. This way, both similarity values can be considered informative in the sense that they highlight that, for this observation, the feature clearly indicates its class membership. Thus, the feature is relevant to discriminate between these classes. However, this appears less plausible for observations *within* a class. Using entropy on these similarity values implicitly assumes that even within a class, observations that are highly similar to their ideal vector and those being highly dissimilar to it can be treated equally. However, observations of a feature that are highly dissimilar to their own class's ideal vector element are intuitively not a positive sign. This means that the ideal vector does not represent observations of the class well. Nevertheless, such feature values lead to low entropy values, indicating informativity. Likewise, feature values for observations that are close to any competing class's ideal vector element lead to low entropy values as well. Once more, this is not a

positive sign since a class's feature representative should not also represent feature values of other classes well.

One last assumption is that the features can be evaluated on a stand-alone basis (univariate). Hence, it is expected that no or only marginal feature interactions are present since they simply cannot be captured explicitly. It is noteworthy that this assumption is different from the first assumption about one ideal vector per class. The first assumption is related to the data structure and is independent of the number of features. In contrast, assumption three means that a feature is evaluated merely by itself, and no information of other features is incorporated into the calculation of the feature's relevance. Figure 3.1 illustrates four examples that clearly show that the first and third assumption refer to different properties of the data structure.

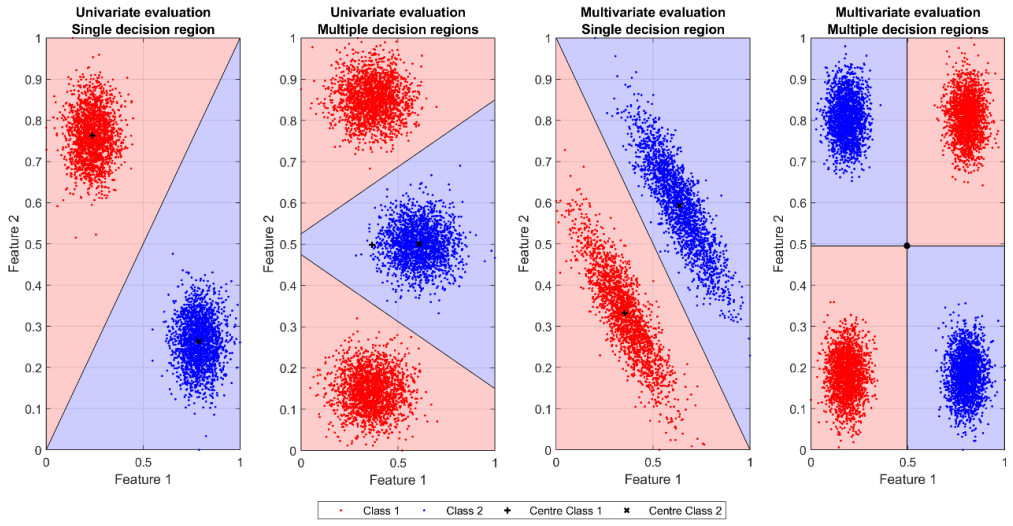


Figure 3.1: Comparison of the first and third assumption. Each graph shows a binary classification problem and the approximate decision regions for each of the classes.

The first example shows one data cluster per class and the corresponding approximate decision region for each of the classes. There is no (conditional) dependency between the first and second feature so that this example can be evaluated univariately.

The second example shows two distinct decision regions for the red class that are non-overlapping with the blue class. Once again, the features can be evaluated univariately and it is apparent that the second feature is more important than the first feature, which overlaps for both classes considerably. It is noteworthy that using a single ideal vector to

represent the red class is problematic. For the red as well as the blue class, the ideal vector elements for the second feature are both essentially at the same position.

The third example highlights two classes that have each a single decision region. In addition, the blue and red class have a (conditional) dependency between the first and second feature. This means that knowing what value one of the features takes, affects the probability of the values that the remaining feature has. Hence, for this example, a multivariate evaluation is necessary to capture that the first and second feature together can linearly separate the classes but are each alone highly overlapping.

The fourth example shows a numeric XOR problem, where both classes have two distinct decision regions. For both features the centre point of each class is essentially the same, which indicates that using one ideal vector per class will make it challenging, if not impossible, to capture the feature relevance for this data structure. Moreover, this problem needs to be addressed multivariately since the classes are for each feature overlapping entirely, whereas both features together enable a linear separation between the two classes.

In summary, the assumption of a single ideal vector and of a univariate evaluation of features address two different aspects of the data. However, even when following the assumptions that having one ideal vector per class and the univariate evaluation of features are acceptable, dealing equally with high and low similarity values using entropy (the second assumption) can pose a problem even with simple data structures.

Publication **I** and Publication **II** presented artificial examples to highlight the vulnerability of the feature selection method by Luukka (2011) for selected simple, low-dimensional two-class and three-class cases. In the following, a set of three binary classification problems will be presented, which can be seen as a simple extension of the binary problem presented in Publication **II**. The reason for presenting binary classification problems in a two-feature context is that it is easy to follow, and the desired removal decision is intuitive. The three examples are displayed in Figure 3.2. All of these examples contain a first feature that is basically completely overlapping for both classes and, therefore, irrelevant for the classification. In other words, knowing which value the first feature takes does not help in assigning an observation to one of the two classes. For almost any value this feature takes, it is equally probable that it will belong to either of the classes. The focus of these three examples is on the second feature, which differs in terms of the magnitude of the variance.

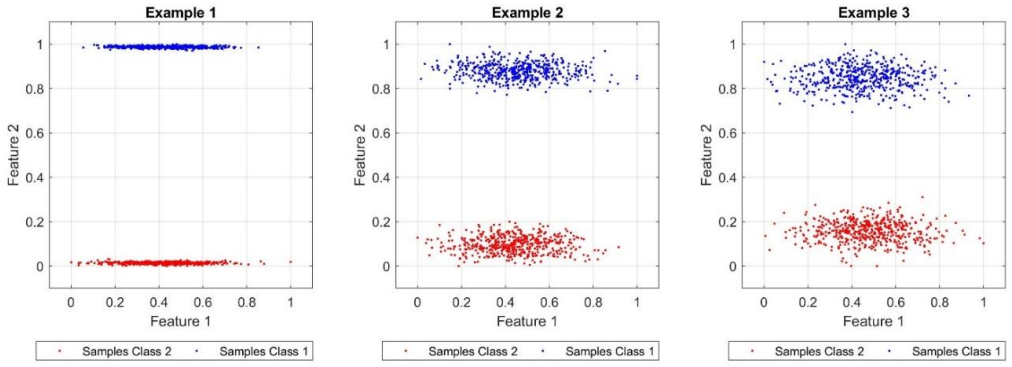


Figure 3.2: Examples of a binary classification problem with different magnitudes of variance. Each class consists of 500 observations generated from a multivariate normal distribution. Example 1 with means $\mu_{red} = (5{,}40), \mu_{blue} = (5{,}60)$ and covariance matrices $\Sigma_{red} = \Sigma_{blue} = \begin{bmatrix} 0.075 & 0 \\ 0 & 0.01 \end{bmatrix}$; example 2 with the same means and covariance matrices but with the variance for the second feature increased to 1; and example 3 with the same means and covariance matrices but with the variance of the second feature set to 2.

In the first example, the second feature is in principle an optimal feature for the algorithm by Luukka (2011) in the sense that the approach for this example will work exactly as intended, and the data structure does not allow the vulnerabilities of the algorithm to be revealed. For this feature, the two classes have mean values close to 0 and 1, respectively, as well as a very low variance. In contrast, the first feature is entirely overlapping and, hence, irrelevant. Thus, the first feature should be selected for removal.

Looking at Figure 3.2, the expectation for the first feature's similarity values is that for both classes a large share of observations of each class will be fairly close to the class's own ideal vector as well as the ideal vector of the competing class. The reason for this expectation is that these ideal vectors are extremely close in that dimension. The distribution of the observations' similarities with their own class's ideal vector and the competing class's ideal vector is displayed in Figure 3.3. Figure 3.3(a) highlights that, indeed, the distribution of similarities for the first feature shows that most similarities are very high. In particular, more than 50% of the similarity values are within the range of 0.9 to 1, and another 30% is between 0.8 to 0.9. However, given the large variance for the first feature, also more than 15% are between 0.4 and 0.7, which is a range that will result in high entropy values. Consequently, the entropy values for this feature are mainly in the middle range of possible entropy values, with few entropies being very small (0 to 0.1) and even fewer being very large (0.6 to 0.7) (see Figure 3.3(c)). For the second feature, however, all observations of both classes are highly similar to their ideal vector (similarity of 0.9 to 1) and highly dissimilar to the ideal vector of the other class (similarity of 0 to 0.1) (see Figure 3.3(b)). For this reason, analysing the corresponding entropy values for the second feature reveals very low (0 to 0.1) to low (0.1 to 0.2)

entropies for all similarity values. Consequently, it comes as no surprise that the summed entropy of the second feature is a fraction of that for the first feature (145 vs 628). For this very distinct structure in example 1, the algorithm by Luukka (2011) correctly suggests the first feature for removal.

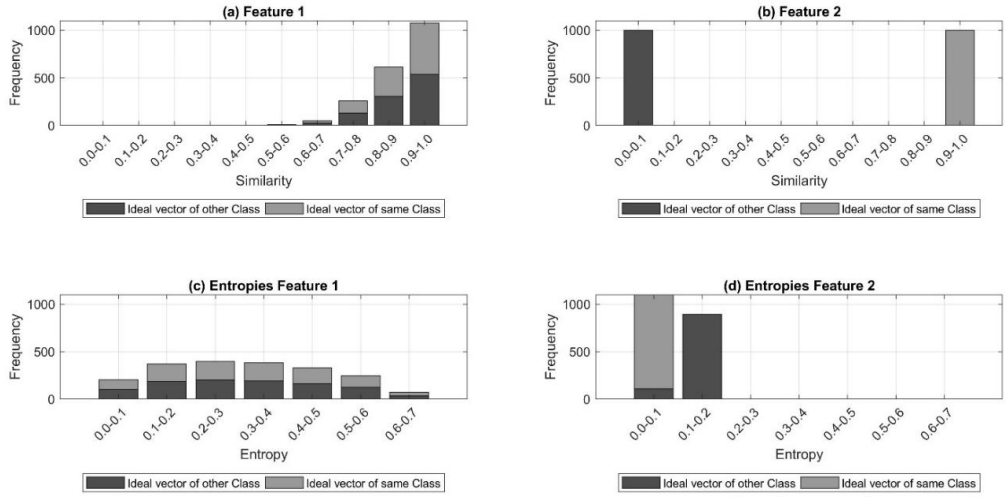


Figure 3.3: Similarity and entropy histograms for both features (first example)

Before presenting the similarity and entropy values for the second and third example, the effect that variance has on the scaled data and the corresponding similarity values will be elaborated on. The effect is twofold:

First, the opposing ideal vector elements at the upper and lower ends of the scale are now moved to the centre, which restricts the smallest similarity that an observation's feature value can possibly have to these ideal vector elements. This shift is premised on the scaling to the compact interval [0,1]. In comparison to a feature with lower variance, observations for a feature with higher variance are moved further away from their ideal vector. Since the observations on the higher end (at the top of the figure) and the lower end (bottom of the figure) have 'nowhere to go' due to the limits of 1 and 0, the ideal vectors as the representative points are pushed downwards and upwards, respectively. So, the ideal vectors are on average closer to each other. Hence, higher variance constrains observations from being highly dissimilar to the ideal vectors of competing classes.

Second, the impact of a higher variance affects the similarity values to the competing ideal vectors disproportionally. A simple example of this behaviour and the change in the ideal vector elements is presented in Figure 3.4.

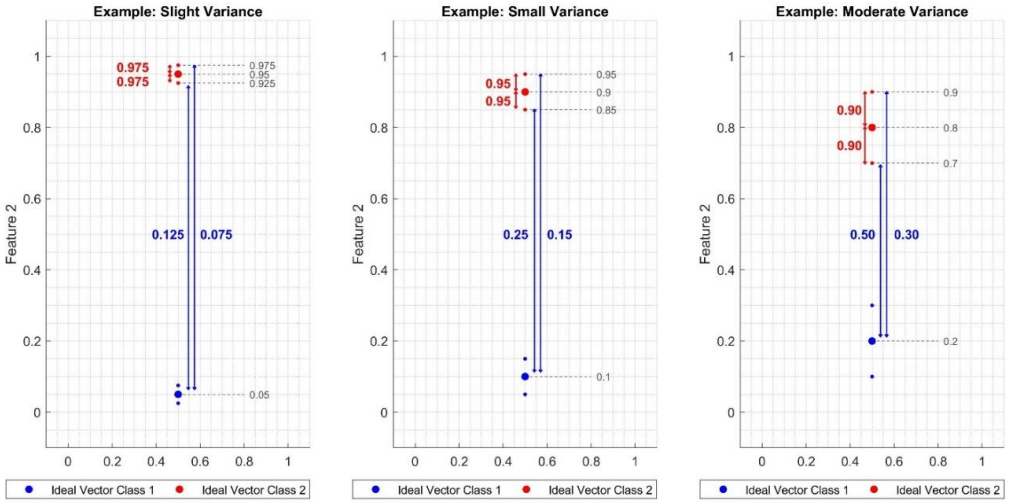


Figure 3.4: Change of similarity values for different levels of variance in a binary classification task

Figure 3.4 illustrates for the red class how the similarity of the feature values to its own ideal vector element does not account for whether the feature has a lower or higher value than the ideal vector element. In other words, whether the observation is above or below the ideal vector for that feature does not affect the similarity value. In contrast, for the competing ideal vector that is non-overlapping, it does matter whether an observation's feature value is below or above the ideal vector element of that class. Hence, for non-overlapping classes, an increase in a feature's variance will affect all similarity values, but this effect will be amplified for similarity values of observations with the competing ideal vectors.

The focus is now on the change in the second feature since the first feature remains unaltered throughout all three examples. The specific effect that the increased variance in the second example has on the similarity values is displayed in Figure 3.5.

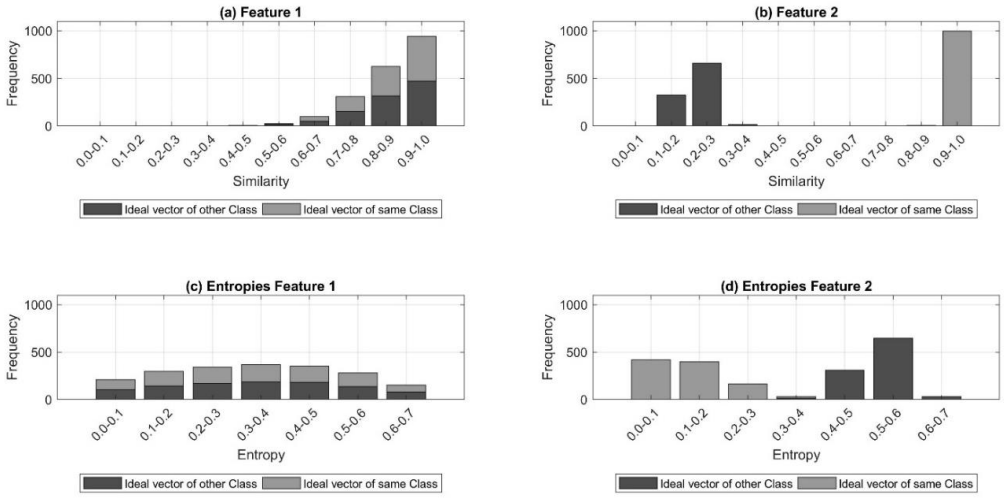


Figure 3.5: Similarity and entropy histograms for both features (second example)

For the second feature, the similarity values between the ideal vector element of a class and the observation in that same class are on average still very high (Figure 3.5(b)). However, a few observations start to become less similar to their own ideal vector element (0.8 to 0.9). The changes in the similarity values of the observations' feature value to the competing class's ideal vector element are more pronounced. There are no longer any similarity values between 0 and 0.1. Moreover, the similarity values are more spread out than those to the observation's own ideal vector.

The corresponding entropy values (Figure 3.5(d)) are now on average considerably higher than in the first example (Figure 3.3(d)). The entropy values for the similarities of observations to the competing ideal vector for a feature experience the largest increase. However, also, the decrease in the similarity of the observations' feature values to their own ideal vector element can be tracked to the elevated entropy values. Thus, the second feature's entropy is substantially increased. Nonetheless, this entropy value is still smaller than that of the first feature (646 vs 680), and the first feature remains the one selected for removal. Notwithstanding, in this example the features' entropy values indicate that the two features possess almost the same relevance, meaning a similar ability to discriminate between classes. This is obviously not representative of the actual data structure. A completely overlapping feature with high variance is certainly not as discriminating as a non-overlapping feature with small variance.

In the third example, this issue of capturing the actual ability of features to discriminate between classes is taken one step further. The second feature's mean values for the two classes are the same as before, but the in-class variance of the feature is increased one more time. After normalization, once more a shift of the ideal vector (represented by the centre of each class) towards the middle can be observed (see Figure 3.2). The

consequences of this shift and of the higher variance for the similarity and entropy values come into view in Figure 3.6(b) and (d).

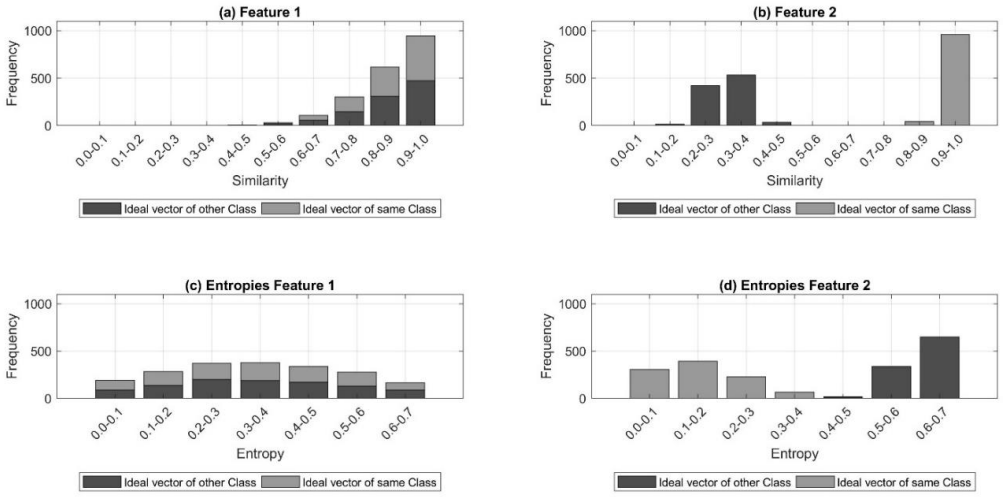


Figure 3.6: Similarity and entropy histograms for both features (third example)

Compared to Figure 3.5(b), the similarity values of the second feature have become more ambivalent – that is, being closer to 0.5 – especially for the similarity of observations with the competing ideal vector element. On account of this, the entropy values have likewise increased on average, especially those based on the similarity of observations to a competing ideal vector element. The summed entropy value for the second feature is now clearly higher than that of the first feature (767 vs 689). As a result, for the first time, the removal of the second feature is suggested. Nonetheless, the second feature in this third example is still far from having a high variance within any of the two classes. Additionally, it is also clearly non-overlapping among the different classes. All the same, the feature selection method by Luukka (2011) suggests the removal of this feature rather than the removal of the entirely overlapping first feature. Going one step beyond this, it is obvious from this knowledge that a feature with higher overlap could also be 'preferred' for retention by the algorithm than one with a smaller overlap.

These examples demonstrate for a binary classification the problem that the feature selection algorithm by Luukka (2011) faces and how it ends up making unintuitive and incorrect feature removal decisions. Clearly, the effect of scaling and the entropy measure do not necessarily lead to unintuitive feature removal decisions. However, it is clearly a weakness of the algorithm by Luukka (2011) that for simple examples it can come up with highly unintuitive and incorrect suggestions for the feature removal.

3.2 Introduction and Reasoning

As demonstrated in the previous section, the feature selection method developed by Luukka (2011) can encounter problems with simple intuitive feature selection tasks. Even when the assumptions of a single ideal vector per class and the univariate evaluation are acceptable for a feature selection problem, the combination of fuzzy similarity and entropy (FSAE) by itself might not capture the relevance of features correctly or even rank an irrelevant feature as most relevant. Overall, this weakness can be linked to an inadequate consideration of the difference among ideal vectors, as was highlighted in Publication **I**. The similarity values of observations with their class's ideal vector element capture variance in a class since the distribution will have a longer tail in the direction of smaller similarity values. This means that if higher variance is present, more observations will have lower similarity values with their class's ideal vector element, and fewer observations will be close to it (similarity close to 1). As can be seen in Figure 3.4 in the prior section, variance can impact the ideal vectors as well by moving them towards the centre. However, even in example 2, where the feature removal decision by the method of Luukka (2011) was a close call, and in example 3, where it was clearly incorrect, the distance between the ideal vectors was large with respect to the regarded compact interval $[0,1]$. Even though the ideal vectors for the first feature were essentially at the same position and for the second feature about 0.8 (80% of the entire range from $[0,1]$) from one another, the algorithm decided that both features were comparably relevant. In these examples, each class is embodied by a single grouping, where the single ideal vectors per class represent the centre of the data points. Therefore, it is apparent that there is a link between the features' relevance and the relation of the distance between the classes' ideal vectors and their variance. In simple terms, if the ideal vector elements for a feature are distant from one another, and the variance in the class is small, this indicates that the feature is relevant. In contrast, if the ideal vectors are close, and the in-class variances are high, then the feature is only marginally relevant or even irrelevant. As mentioned previously, the in-class variance is accounted for by the similarity in the method by Luukka (2011). In contrast, the distance between ideal vector elements is not sufficiently considered. Hence, it is possible that a feature that is overlapping for different classes is considered more relevant than a non-overlapping feature with small or moderate variance.

From this line of reasoning it is apparent that taking into consideration the distance between the ideal vector elements of each feature in addition to similarity and entropy could compensate for the vulnerability of the feature selection method of Luukka (2011). Since the distance between classes for a specific feature is at the centre of this vulnerability, a modification of the class- and feature-specific entropy values appears intuitive. In the suggested improvement of the algorithm by Luukka (2011) presented below, these entropy values can be scaled, for each feature, by a measure that accounts for the distance between the classes' ideal vector elements. Following this approach, the summed scaled entropy values for each feature will incorporate the distance between the classes' ideal vector elements for that feature. The corresponding step-by-step algorithm will be depicted and illustrated in the subsequent section.

3.3 Step-by-Step Algorithm of FSAE

The algorithm underlying the novel fuzzy similarity and entropy (FSAE) -based feature selection method using a class- and feature-specific scaling factor is visualized in Figure 3.7. FSAE is conceptualized as an information-based filter method to rank features according to their relevance with respect to the classes in a dataset.

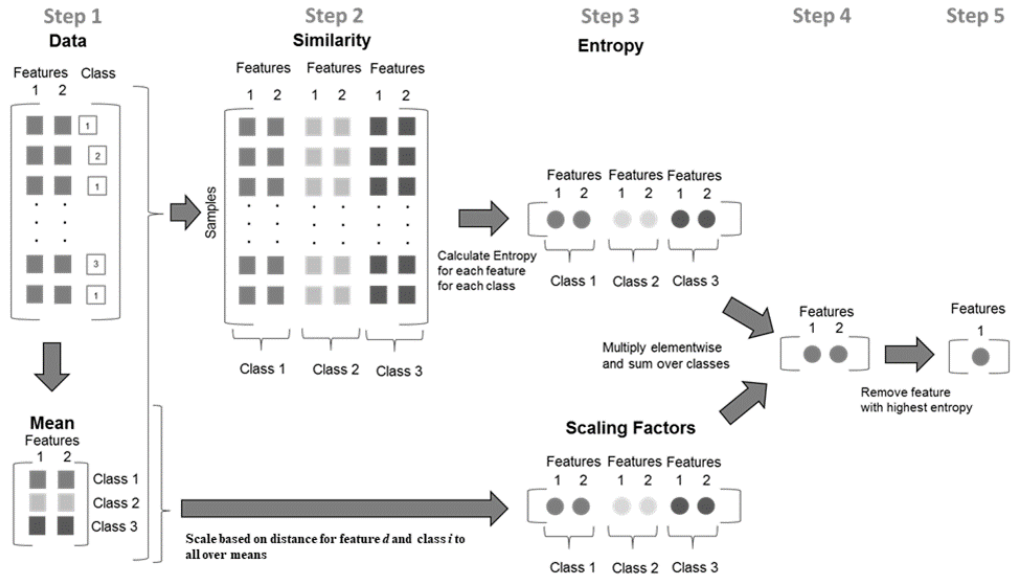


Figure 3.7: Step-by-step process underlying the FSAE feature selection for a two-dimensional example with three classes [reproduced from Publication **I** with the permission of the publisher]

The first step is the calculation of the ideal vectors for each class. As in the feature selection method of Luukka (2011), the ideal vector can be computed as a generalized mean (Luukka, 2011; Publication **I**). The ideal vector element for the *d*-th feature can be calculated as

$$v_{i,d} = \left(\frac{1}{n_i} \sum_{x \epsilon X_i} {x_d}^m \right)^{\frac{1}{m}}, \tag{3.1}$$

where n_i is the number of observations in class *i*, and the feature is denoted by *d*. The calculation of the generalized mean is the same as in Luukka (2011).

The second step is the same as that of the feature selection method by Luukka (2011). For each observation *j* and each feature *d*, this step consists of the calculation of its similarity with the ideal vector values of that feature for each class *i*.

$$S(x_{j,d}, v_{i,d}) = \sqrt[p]{\left(1 - \left|x_{j,d}^{p} - v_{i,d}^{p}\right|\right)}, \tag{3.2}$$

where j denotes the observation (from $j = 1,2, \dots n$), d the feature (from $d = 1,2, \dots D$) and i the class (from $i = 1,2, \dots N$). The resulting similarity matrix is of dimension $n*(DN)$ (Publication **I**).

The third step encompasses the calculation of the feature- and class-specific entropy and the computation of the feature- and class-specific scaling factors. The feature- and class-specific entropy values for class i and feature d denoted by $H_{i,d}$ are the sum over the observations' entropy values for a certain feature and class.

$$H_{i,d} = \sum_{j=1}^{n} H\left(S(x_{j,d}, v_{i,d})\right) \tag{3.3}$$

In contrast to the feature selection method by Luukka (2011), the summation of entropy values is conducted over all observations but not over all classes (see Equation (2.16)).

The second computation implemented in this step is to determine the class- and feature-specific scaling factors $SF_{i,d}$. The objective of the scaling factor is to emphasize the distance between the classes' ideal vectors for a certain feature (Publication **I)**. To do this, the distance of the ideal vector element of class i for feature d from all other classes' ideal vector elements for this feature is measured and the resulting distances averaged.

$$SF_{i,d} = 1 - \frac{\left(\sum_{i \neq j} \left|v_{i,d} - v_{j,d}\right|^{l}\right)^{\frac{1}{l}}}{N - 1} \tag{3.4}$$

The parameter l controls the weight of the distance between two ideal vector elements $v_{i,d}$ and $v_{j,d}$ in the quotient. The numerator in the quotient in the above equation is essentially the Minkowski distance. With $l = 1$, the result is simply the absolute distance between two ideal vectors averaged. With $l = 2$ and any higher value for l, the impact of larger distances is amplified in the formula, and the impact of lower distances between two ideal vectors is diminished. The measure of distance between the ideal vectors presented in the quotient in Equation (3.4) is subtracted from one. Hence, high distances between ideal vectors lead to low scaling factors and small distances or a distance of zero in a scaling factor close or equal to 1, respectively (Publication **I**).

The fourth step uses the feature- and class-specific entropy values obtained from Equation (3.4). These values are multiplied with the feature- and class-specific scaling factors and, subsequently, summed over all classes to obtain the scaled entropy values for each feature (Publication **I**).

$$SE_d = \sum_{i=1}^{N} \left(H_{i,d} * SF_{i,d} \right) \tag{3.5}$$

Using Equation (3.5), for each feature d a corresponding scaled entropy value SE_d is calculated. The underlying idea behind the multiplication of the entropy values and the scaling factors is based on the meaning of entropy. As mentioned in a previous section, entropy embodies the level of informativity (Yao, Wong and Butz, 1999), with low entropy values indicating a high degree and high entropy values indicating a low degree of informativity. The objective of the scaling factor is to keep the entropy values at a similar level if the ideal vector element of a class is close to the ideal vector elements of other classes and to lower the entropy values if the ideal vector element of the feature is far from all others. The first case represents no or a minor change in the informativity of the features for the class label assignment. The second case can be interpreted as a correction of the level of informativity towards more informativity if the ideal vector element for a feature is far from those of competing ideal vectors. This assumes that a distinct ideal vector element for a feature – one that is clearly separated from the feature representatives of the other classes – is desirable for classification and increases the level of informativity of a feature (Publication **I**). The examples for the feature selection by Luukka (2011) indicated that not accounting in some form for the distance between class ideal vector elements for a feature can result in unintuitive and incorrect feature removal decisions.

The fifth step uses the scaled entropy value of each feature for the feature removal. The feature ranking is implemented by sorting the scaled entropy values in ascending order, from the most informative feature to the least informative one (Publication **I**). Since the magnitude of the scaled entropy itself may be difficult to interpret, a simple normalization to the compact interval $[0,1]$ can be conducted and the meaning reversed by subtracting the normalized value from 1. This ensures that these values represent a kind of informativity score of features in relation to each other. The feature that is most informative in the dataset obtains an informativity score of 1 and the least informative a score of 0. Hence, any informativity score between 0 and 1 indicates how informative a feature is in relation to the most informative feature in the dataset (Publication **I**). It is apparent that this also means that more informative features in a dataset obtain higher informativity scores than less informative ones. For the filter method FSAE, simply the user-specified k features with the highest informativity scores (or the lowest scaled entropy values) are selected for the feature subset (Publication **I**). The less informative features with lower informativity scores (and higher scaled entropy values) are discarded.

As with the feature selection method of Luukka (2011), the FSAE algorithm can also be applied as a wrapper method using a classifier. In the first step, the data need to be divided into training and test data. Steps one to four are now conducted exclusively on the training data. After the feature ranking based on the training data is obtained, only the least informative feature (with the highest scaled entropy value) is suggested for removal. For the wrapper version, the fifth step also encompasses the calculation of the classification

accuracy on the test set before and after the feature removal. If the stopping criterion, such as a performance degradation above a user-specified threshold, is met, the wrapper approach stops and suggests the feature subset with the highest test set performance from one of the previous iterations. If the stopping criterion is not satisfied, the wrapper will remove the feature and start another iteration of the algorithm from step 1 to step 5 (Publication **I**).

3.4 Application to Artificial Data

The fuzzy similarity and entropy (FSAE) feature selection algorithm can be applied to the same examples that were previously presented in Figure 3.2. These examples demonstrated the impact of higher variance on the entropy values in the algorithm by Luukka (2011) and on its removal decision. The FSAE will be deployed on these examples to illustrate the effect on the feature removal when the distance between ideal vector elements is accounted for. It has to be stressed that the similarity values for both feature selection approaches as well as the entropy measure deployed are the same. Hence, the difference in the entropy values can exclusively be attributed to the difference in the ideal vector elements incorporated into the FSAE in the form of scaling factors. The comparison of both approaches for the second feature in all three examples is displayed in Figure 3.8.

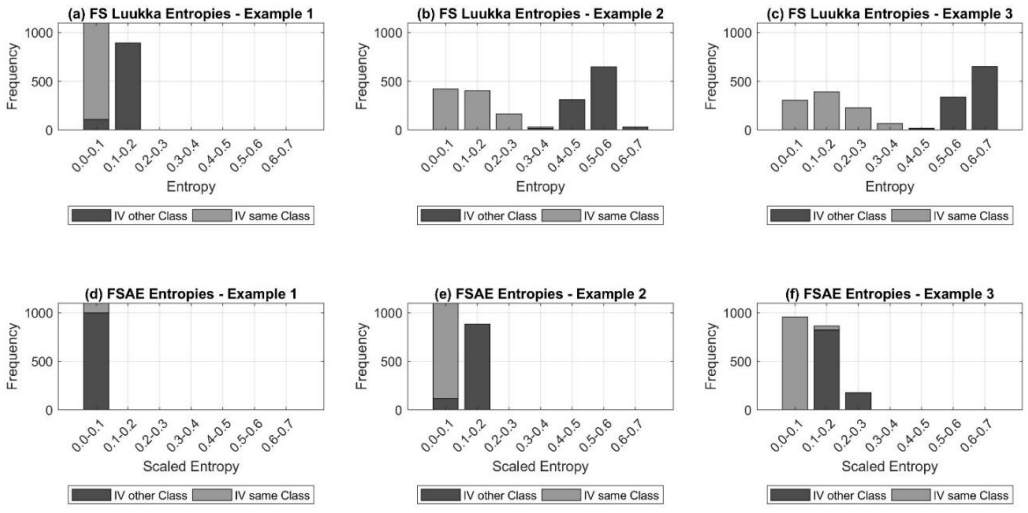


Figure 3.8: Comparison of entropy histograms for FS Luukka (2011) and FSAE (for the second feature in all binary class examples)

The first row of subplots from (a) to (c) shows the entropy values of the approach by Luukka (2011) for each of the three artificial examples. In particular, these plots demonstrate how the increase in variance from slight to moderate gradually increases the entropy values (from left to right). Both the entropy values of the similarity values to the ideal vector element of the same class as well as those to the competing class's ideal vector element experience this development. However, as discussed previously, the inability to account for the difference in the ideal vectors of the classes results in entropy values that do not represent the actual relevance of a feature.

The overall results for all three examples are presented in terms of the feature entropy values in Table 3.1.

Algorithm	Example 1		Example 2		Example 2	
	Feature 1	Feature 2	Feature 1	Feature 2	Feature 1	Feature 2
FS Luukka 2011	627.5	144.7	679.6	645.6	688.8	766.9
FSAE	626.4	3.8	668.8	140.2	665.1	236.4

Table 3.1: Comparison of the feature selection algorithms on artificial two-class data examples

As with the approach by Luukka (2011), the FSAE is characterized by an increase in the variance of the scaled entropy values. Notwithstanding, the range of these values is considerably smaller and the distribution more compact than in the counterpart by Luukka (2011). Even in the third example, the majority of scaled entropy values remains between 0 and 0.1, and the largest scaled entropy value does not exceed 0.3. Overall, the summed scaled entropy value of the FSAE for the second feature is clearly lower than for the first feature. Hence, for all three examples, FSAE suggests the completely overlapping first feature for removal. In addition, it is apparent that the summed scaled entropy in the given examples accounts for the variance that makes the feature less discriminating by resulting in a higher summed scaled entropy value for the second feature in the first to the third example. At the same time, it keeps the scaled entropy representative of the actual data structure and avoids the unintuitive removal of the second feature. Both algorithms are applied with the same standard parameters $m = 1$ and $p = 1$ to the three artificial examples. For any combination of $p \in [1,6]$ and $m \in [1,8]$, the removal decision does not alter.

The same conclusion as in the previous three examples was drawn in the three artificial examples presented in Publication **I**. These examples contained three classes and two features, with a different level of variance and overlap in each example. These examples are visualized in Figure 3.9.

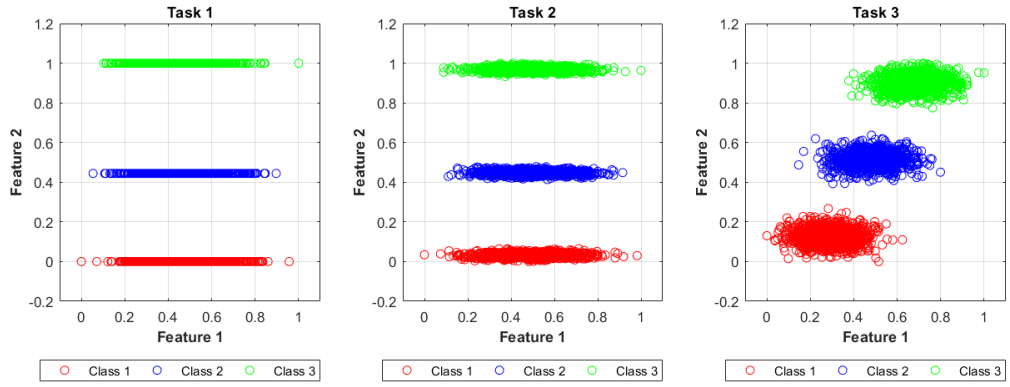


Figure 3.9: Examples of three-class classification problems [reproduced from Publication **I** with the permission of the publisher]

In all three examples, it is apparent that the first feature is less discriminative than the second one. In the first two examples, the first feature overlaps entirely for all three classes, and the second feature is non-overlapping with no variance and small variance, respectively. This can simply be regarded as the three-class extension of the previous binary examples. The third example is characterized by only a partial overlap of the first feature and higher variance for the non-overlapping second feature. Hence, the first two examples differ in terms of variance, making the second feature less discriminating, whereas the third example additionally decreases the overlap of the first feature to make it more discriminating than before.

The focus of all three examples remains on the second feature, which differs in terms of variance in all of these examples. The corresponding entropy values for the approach of Luukka (2011) and the FSAE are presented in Figure 3.10.

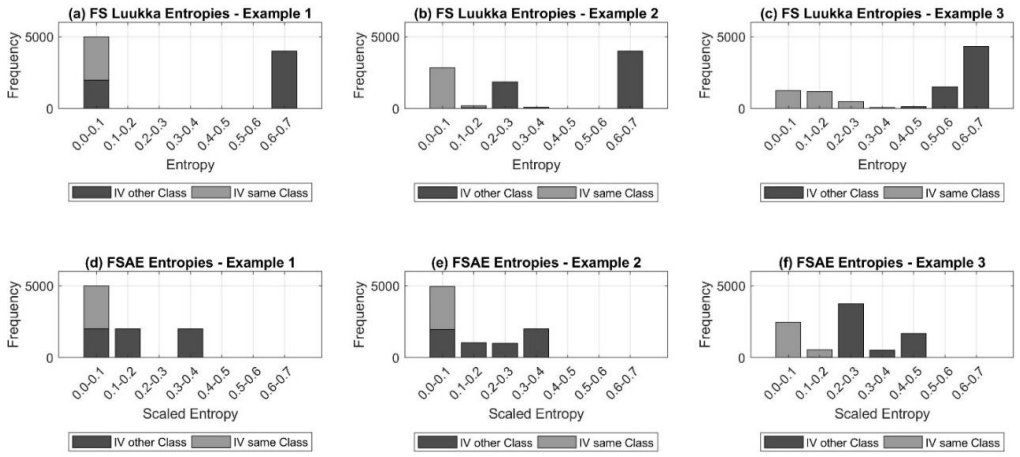


Figure 3.10: Comparison of entropy histograms for FS Luukka (2011) and FSAE (all three-class examples)

Overall, the first example for the feature selection algorithm by Luukka (2011) leads to very low and very high entropy values, which summed up equate to a still rather high entropy value. The reason for this outcome is intuitive considering the similarity values associated with this example. All similarity values for the second feature with their class's ideal vector elements is close to 1, leading to low entropy values. In contrast, the entropies of observations to competing ideal vector elements are either also negligible or at their maximum of close to 0.7. Since the ideal vector elements for the second feature are at 0 (red), 0.5 (blue) and 1 (green) (see Figure 3.9), the similarity of observations to competing ideal vector elements is either 1 or 0.5. This results in negligible and high entropy values, respectively. For the second and third examples, the summed entropy rises further, mainly due to increased similarity of feature values to the competing classes' ideal vector elements.

The results for the FSAE approach differ in terms of the distribution of scaled entropy values. This difference originates in the use of the scaling factors. For the peripheral classes (red and green), the scaling factor of 0.75 is larger than for the central cluster (blue) at 0.5. It is apparent that the scaling factor accounts for the fact that the peripheral classes are on average further from the remaining classes than the central class is. FSAE reduces the high entropy values observed by Luukka (2011) for all observations – but depending on the observation's class to an unequal magnitude. The same logic applies for the remaining two examples with an increasing degree of variance.

The entropy values for the feature selection by Luukka (2011) and the FSAE are presented in Table 3.2.

Algorithm	Example 1		Example 2		Example 2	
	Feature 1	Feature 2	Feature 1	Feature 2	Feature 1	Feature 2
FS Luukka 2011	1284.3	1474.2	1180.8	1686.9	1847.1	2058.6
FSAE	1024.5	430.3	904.3	532.4	1057.2	813.0

Table 3.2: Comparison of the feature selection algorithms on artificial three-class data examples [modified from Publication **I**][2]

In all examples, the average scaled entropy is clearly lower for the FSAE due to the scaling factor that accounts for the distance between the ideal vectors for that feature. While the approach by Luukka (2011) suggests in all three-class examples the removal of the relevant second feature, FSAE correctly suggests the first feature for removal.

3.5 Application to Medical Data

In Publication **I**, the FSAE feature selection algorithm was also tested on five real-world medical datasets. The medical datasets are freely available at the UCI Repository of Machine Learning Database (Lichman, 2013). They are summarized in a condensed way in Table 3.3.

[2] In Publication **I**, the summed entropy values for the approach by Luukka (2011) were divided by the number of classes. To make the entropy values for the approach by Luukka (2011) and the FSAE more easily comparable, the results for the approach of Luukka (2011) for this three-class problem are multiplied by three compared to the ones found in Publication **I**. This obviously has no impact on the feature ranking and removal decision since the feature ranking is conducted for each method independently.

Dataset	Short Description	Number of Features	Number of Classes	Complete Observations	Contributor
Dermatology	Differential diagnosis of erythemato-squamous diseases	34	6	358	Ilter and Guvenir (1998)
Chronic Kidney Disease	Characteristics of patients with and without chronic kidney disease	24	2	156	Soundarapandian and Rubini (2015)
Breast Cancer Wisconsin (Original)	Characteristics of patients with benign and malignant breast cancer	10	2	683	Wolberg (1992)
Diabetic Retinopathy Debrecen	Features extracted from the Messidor image set for the prediction of diabetic retinopathy	19	2	1151	Antal and Hajdu (2014)
Horse Colic	Medical features of horses including that had surgical and non-surgical lesions	27	2	379	McLeish and Cecile (1989)

Table 3.3: Description of medical datasets for the FSAE

Before comparing the approach by Luukka (2011) and the FSAE filter with other information- and distance-based filter methods, the wrapper versions of these two algorithms were compared. The classifier deployed within the wrapper method and for the comparison of feature subsets for the filter method is the standard similarity classifier with optimal value search for the $p \in [1,6]$ and $m \in [1,8]$ parameters. The comparison of the best setup for the wrapper version of the approach by Luukka (2011) with the wrapper version of the FSAE is illustrated in Table 3.4.

Dataset	Approach	Entropy	Removed Features	Removed (in %)	Avg. Performance	Variance
Dermatology	No FS	-	-	-	98.11 %	0.0110 %
	Sim + FS Luukka (2011)	Parkash et al	1	3.0 %	98.20 %	0.0119 %
	Sim + FSAE (l = 1)	De Luca and Termini	5	15.2 %	98.36 %	0.0112 %
Chronic Kidney Disease	No FS	-	-	-	99.90 %	0 %
	Sim + FS Luukka (2011)	De Luca and Termini	6	25.0 %	100.00 %	0 %
	Sim + FSAE (l = 2)	De Luca and Termini	20	83.3 %	100.00 %	0 %
Breast Cancer Wisconsin	No FS	-	-	-	97.61 %	0.0001 %
	Sim + FS Luukka (2011)	Either entropy	0	0.0 %	97.64 %	0.0001 %
	Sim + FSAE (l = 1 or 2)	Either entropy	4	44.4 %	97.30 %	0.0001 %
Diabetic Retinopathy Debrecen	No FS	-	-	-	59.57 %	0.0647 %
	Sim + FS Luukka (2011)	De Luca and Termini	7	36.8 %	61.05 %	0.0398 %
	Sim + FSAE (l = 1 or 2)	Either entropy	5	26.3 %	61.09 %	0.0419 %
Horse Colic	No FS	-	-	-	86.72 %	0.0576 %
	Sim + FS Luukka (2011)	Parkash et al	7	30.4 %	87.70 %	0.0578 %
	Sim + FSAE (l = 1 or 2)	Either entropy	11	47.8 %	87.70 %	0.0667 %

Table 3.4: Comparison of wrapper feature selection on multiple medical datasets [modified from Publication **I**]

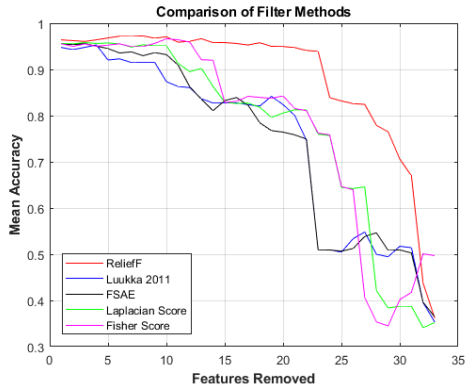
The results highlight two essential aspects of these approaches with respect to one another and to using no feature selection at all. The first aspect is that both approaches successfully conduct feature selection, meaning that for basically all datasets, they suggest a feature subset that leads to a classification accuracy at least comparable to the complete feature set. For the first three medical datasets, the average classification accuracy did not change notably when features were removed from the dataset, which suggests that these features were not relevant for the classification. For the last two medical datasets, the Diabetic Retinopathy and Horse Colic datasets, removing several features even increased the classification accuracy as opposed to using the entire feature set. This indicates that some features were not only irrelevant for classification but even rendered the classification more challenging and acted as noise in the data.

The second aspect is that, with exception of the Diabetic Retinopathy dataset, the feature subset suggested by FSAE achieved the same classification accuracy as the approach by Luukka (2011) while using a smaller subset of features. From the five considered medical datasets, the most distinct difference can be observed for the Chronic Kidney Disease dataset, where the approach by Luukka (2011) suggests the removal of six features, whereas FSAE suggests removing 20 out of the 24 features to obtain the perfect classification accuracy of 100%. The Diabetic Retinopathy dataset shows only a slight difference in favour of the approach by Luukka (2011). With the approach by Luukka (2011), seven features are removed instead of the five features discarded according to the

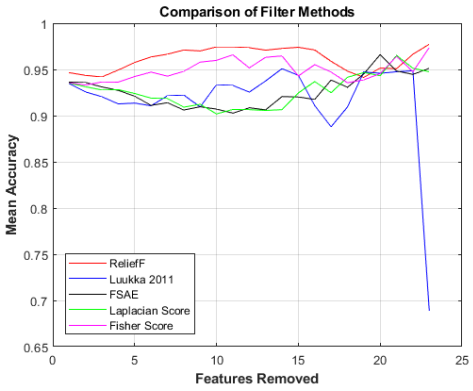
FSAE. Nevertheless, both approaches essentially obtain the same increase in classification accuracy for the dataset.

The results for the comparison of the filter version of the FSAE with the filter version of the approach by Luukka (2011), the ReliefF algorithm, the Laplacian score and the Fisher score are illustrated for all datasets in Figure 3.11. All filter methods were deployed to obtain a feature ranking. Each graph represents the classification accuracy on the feature set for that step-by-step the lowest-ranked features were removed in accordance with the feature ranking of each algorithm. A similarity classifier is deployed for the evaluation of the classification accuracy. For the Dermatology dataset, the ReliefF algorithm, which accounts for the local neighbourhood, clearly outperforms all other filter methods. The remaining filter methods perform comparably against one another. In particular, the mean accuracies of the approach by Luukka (2011) and the FSAE are very similar. This indicates that for this particular dataset, the consideration of the difference between ideal vectors does not impact the feature removal decision considerably. For the second dataset, the Chronic Kidney Disease data, once more the ReliefF algorithm performs particularly well in ranking the features. All feature selection algorithms, with the sole exception of the approach by Luukka (2011), end up within a small range of mean accuracy values around 95%. The approach by Luukka (2011) apparently removes the more relevant feature since the mean classification accuracy with only one feature is, with less than 70%, more than 20 percentage points lower than the accuracy of the remaining algorithms. Since the consideration of the difference among ideal vectors is the main difference between this approach and the FSAE, this clearly shows that, for this dataset, it is essential for the feature removal to consider this difference.

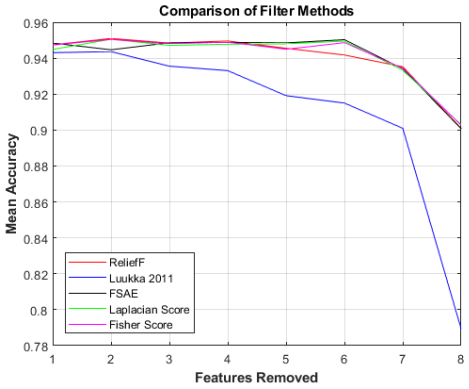
(a) Dermatology



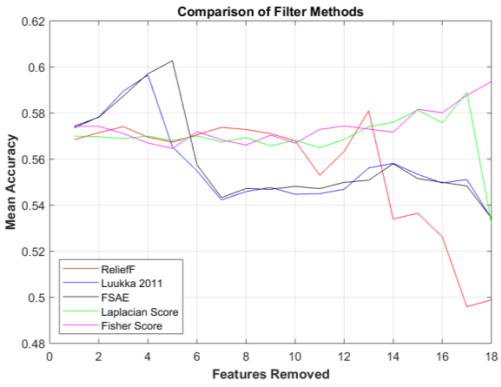
(b) Chronic Kidney Disease



(c) Breast Cancer Wisconsin (Original)



(d) Diabetic Retinopathy Debrecen



(e) Horse Colic

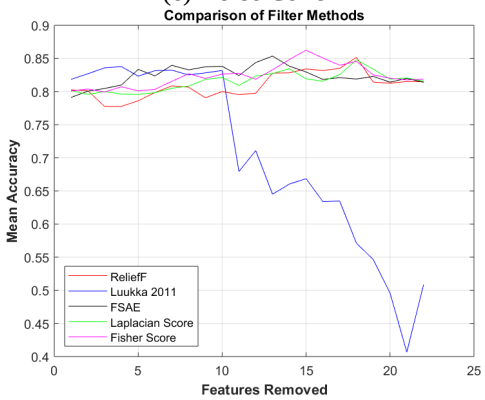


Figure 3.11: Comparison of filter feature selection on multiple medical datasets [reproduced from Publication **I** with the permission of the publisher]

For the Breast Cancer Wisconsin dataset, most approaches, including ReliefF and FSAE, result in similarly high mean accuracies. Only the approach by Luukka (2011) seems to gradually deteriorate in its performance right from the start. The remaining approaches' classification results are more or less stable until the removal of the sixth feature. At that point, the approach by Luukka (2011) had already deteriorated about 4 percentage points in performance. The choice of a single feature, for all approaches, leads to a considerable decline in mean accuracy. However, it is apparent that the feature choice of the approach by Luukka (2011) is inferior to those of the remaining algorithms. The approach's performance declines by an additional 10 percentage points to below 80%, whereas the other filter methods face only a decline of 3 to 4 percentage points to a performance of around 90%.

For the Diabetic Retinopathy Debrecen dataset, FSAE and the approach by Luukka (2011) initially experience an increase in the classification accuracy on account of the first four to five feature removals. While this increase for the approach by Luukka (2011) withstands only the removal of four features, for the FSAE an additional feature can be removed to further increase the mean accuracy. Overall, the FSAE results with five removed features in the highest classification accuracy of all approaches for any number of feature removals. That said, for more than five feature removals, the remaining filter methods outperform the FSAE and the approach by Luukka (2011) by a margin of on average 3 to 4 percentage points.

For the Horse Colic dataset, FSAE and the approach by Luukka (2011) belong initially to the top-performing approaches, also outperforming the ReliefF algorithm by several percentage points. While the FSAE remains for the majority of feature removals within the top filter methods, the approach by Luukka (2011), after the tenth feature, evidently removes several features that are relevant for classification. Thus, it experiences more than once a decline of 5 percentage points for a feature removal. Eventually, the approach by Luukka (2011) ends up at a performance of close to 50%, whereas the remaining algorithms, including the FSAE, are more than 30 percentage points more accurate. Once more, this demonstrates that incorporating the difference in the ideal vectors can improve the feature removal decisions and result in considerably better feature subsets in terms of their ability to classify observations.

Finally, it has to be noted that the FSAE and the approach by Luukka (2011) were applied only with standard parameters and that potentially better results can be achieved with the optimal parameters for each of these supervised feature selection tasks.

3.6 Application to the Prediction of S&P500 Intraday Returns

3.6.1 Introduction and Objectives

In the previous section it was demonstrated for several artificial examples in detail how the FSAE feature selection method can improve the feature ranking of the approach by

Luukka (2011). In this section, the application to a real-world dataset with a focus on the American financial stock market index the S&P500 is presented.

The prediction of price movements in financial markets, such as the S&P500, can be regarded as a pattern recognition problem (Felsen, 1975; Guo *et al.*, 2014). Hence, a machine learning approach based on a combination of feature selection and, subsequently, classification can be pursued to forecast future stock index returns. Previous research using machine learning approaches includes the prediction of stock markets with a variety of classification methods, including neural networks (Altay and Satman, 2005; Fadlalla and Amani, 2014), support vector machines (Guo-qiang, 2011; Guo, Wang, Liu, & Yang, 2014), genetic algorithms (Kim, Han, & Lee, 2004; Leigh, Purvis, & Ragusa, 2002), case-based reasoning (Chun and Park, 2005) and a random subspace classifier (Zhora, 2005). The findings in the academic literature, including momentum anomalies, (Leigh, Purvis, & Ragusa, 2002) indicate that it is possible to successfully derive trading strategies from a suitable set of financial features.

In Publication **II**, the objective is the prediction and classification of the intraday returns of the S&P500 stock market index. The intraday return is the percentage change of the stock market index between the daily opening of the stock exchange (opening price) to its closing, when the trading for the day is finished (closing price). For this purpose, the authors of that research paper used numerous features that they assumed to have an impact on the changes in this market index. The FSAE algorithm is applied to this feature set to obtain a subset of relevant features for this dataset. This subset is subsequently deployed for classification. The aim is to derive trading strategies from the classification model that can outperform a simple buy-and-hold trading strategy in terms of returns after transaction costs.

3.6.2 Data and Feature Selection

The initial data for the analysis is obtained free of charge from the webpage Yahoo Finance (2017) and covers daily data from 11/10/2010 until 28/3/2018. The features downloaded from Yahoo Finance encompass seven large stock market indices, two market exchange traded funds (ETF), six indices/ETFs representing economic sectors and commodities, three currency exchange rates of the US dollar (USD) to the currencies of major trading partners, four time series related to interest rates and yields and the VIX index.

The seven stock market indices represent the stock markets in seven of the largest equity markets around the globe. In addition, the two market ETFs replicate the medium and large capitalized companies in the emerging markets (BlackRock, 2017a) and small, medium and large capitalized ones in the global equity market (Morningstar, 2017). In terms of sector exposure, one ETF reflects the large American companies in the materials sector (State Street Global Advisors (SPDR), 2017b), one ETF represents large companies in the financial sector in the United States (SPDR, 2017a) and one embodies the financial sector in Europe (BlackRock, 2017b). The time series on commodities

includes a gold ETF, a crude oil ETF (S&P Dow Jones Indices, 2017) and an ETF for energy, precious and industrial metals overall as well as grains and livestock (USCF, 2017). The volatility index VIX is included to represent market sentiment and can be regarded as a barometer for investor sentiment in the S&P500 equity market (Rossilo, Giner and De la Fuente, 2014). From all these time series, the return from one trading day's opening price to the same day's closing price (open-close return or intraday return) and from the closing price of one day to the next day's opening price are calculated. Moreover, the change in the daily highest and lowest quoted price and the change in volume and the range (high price – low price) between two consecutive trading days were computed. In addition to the time series data that are directly available from Yahoo Finance, several features have to be derived from them. On one hand, yield spreads have to be calculated between short-, medium- and long-term yields since previous research has indicated that they are connected to the contraction and expansion of an economy (Rudebusch and Williams, 2009). On the other hand, technical indicators such as momentum, moving averages (MA), the relative strength index (RSI), Bollinger bands and the moving average convergence divergence (MACD) are derived from the S&P500 time series (Hurwitz and Marwala, 2011; Di Lorenzo, 2013). Each of these indicates short-term trends or are indicators of whether a market is overbought or oversold, indicating a sell or buy signal. The complete list of features is presented in Table 3.5.

Dependent Variable						
S&P500	Open-Close Return					
Features						
Time Series						
S&P500	-	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
DAX	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
Nikkei225	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
iShares MSCI Emerging Markets	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
Vanguard Total World Stock ETF	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
Hang Seng	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
FTSE 100	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
STOXX 50	Open-Close Return	Close-Open Return	-	-	-	-
Russell 2000	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
VIX S&P500	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
SPDR Gold Shares ETF	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
United States Commodity Index	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
Materials Select Sector SPDR ETF	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
iPath S&P GSCI Crude Oil	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
Financial Select Sector	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
iShares MSCI Europe Financials	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
CBOE Interest Rate 10 Year	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
Treasury Yield 30 Years	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
Treasury Yield 5 Years	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
13 Week Treasury Bill	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
JPY/USD Exchange Rate	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
EUR/USD Exchange Rate	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
CNY/USD Exchange Rate	Open-Close Return	Close-Open Return	Δ (%) High	Δ (%) Low	-	Δ (%) Range
Technical Indicators and Yield Spreads						
Δ (%) Spread Treasury 30y - 5y	Δ (%) Momentum (1d)	MACD (26d, 12d, Signal 9d)	Mov.Avg. (5d)			
Δ (%) Spread Treasury 30y - 13w	Δ (%) Momentum (3d)	Bollinger (2 Std)	Mov.Avg. (10d)			
Δ (%) Spread Treasury 5y - 13w	Δ (%) Momentum (5d)	RSI (14d)				
	Δ (%) Momentum (10d)					

Table 3.5: Initial list of features for the S&P500 feature selection and classification model [modified from Publication **II**]

The selection of features is similar to that in previous research on stock market prediction, where commonly at least financial time series, technical indicators, commodity prices, exchange rates, interest rates and yields/yield spreads are included (Krollner, Vanstone and Finnie, 2010).

The data are divided into two distinct and non-overlapping time periods (1) for training and testing and (2) for forecasting. This division and the transformation of the intraday returns into four classes is illustrated in Figure 3.12.

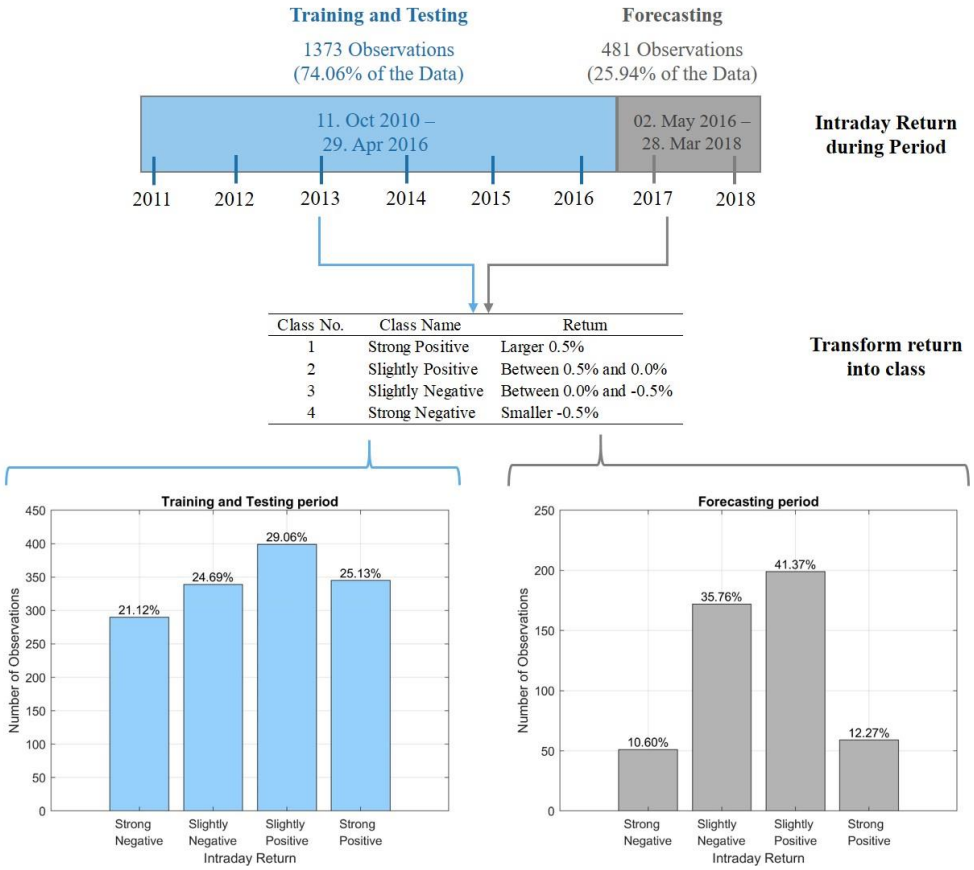


Figure 3.12: Data split and class assignment

The training and testing data span over a period of about five and a half years, whereas the out-of-sample forecast period is located after the training and testing period and is closer to two years of trading days. According to the magnitude of the S&P500 intraday returns (dependent variable), four classes were defined to represent these returns. A ‘strong positive’ class for intraday returns over 0.5%, ‘slightly positive’ for returns between 0% and up to 0.5%, ‘slightly negative’ for the range from -0.5% up to 0% and the ‘strong negative’ class for intraday returns smaller than -0.5%. These four classes represent the dependent variable for the classification task instead of the actual intraday return values. This step is conducted since the aim is not to predict the exact return value, as in a regression task, but to predict the return class, where each class represents how positive or negative the market development is during a certain trading day. The step size of 0.5% for the class assignment was selected because it divides the dependent variable in the training and testing dataset into classes that are close to being balanced in terms of observations in each of the classes. The classes in the forecast period are less balanced and show more observations in the ‘slightly positive’ and ‘slightly negative’ class.

However, this does not pose a considerable problem since the model itself was set up with the training and testing dataset and does not favour any class due to a disproportionally large share of observations in any of the classes. It is worthwhile to note that using a four-class setting differentiates this research from the majority of the existing literature. The existing scientific literature limits itself to the use of two classes to cover exclusively upward and downward movements in stock markets. For instance, in the research work of Patel et al (2015) on stock market predictions, the authors used a two-class setup for their model but suggested that using more categories than two is worth exploring.

The FSAE feature selection algorithm is applied to the initial dataset containing 136 features in order to obtain a feature subset that only contains features that are relevant for the classification of the S&P500 stock index returns. The FSAE setup with Parkash, Sharma and Mahajan (2008) entropy and $l = 1$ is selected to conduct the feature selection. Deploying the FSAE together with a random forest classifier suggests the removal of 38 out of 136 features (27.9% of the features).

3.6.3 Results and Conclusion

The comparison of several classifiers on this feature subset of 98 features demonstrates that the random forest achieved the highest average classification result (Publication **II**) (see Table 3.6).

Approach	Setup and Parameters	Avg. Performance	Variance (in %)
Similarity Classifier	$p = 3$, $m = 1$	44.04 %	0.03 %
Random Forest	Min Leaf Size = 1	43.63 %	0.04 %
Random Forest	Min Leaf Size = 10	44.72 %	0.03 %
KNN	$k = 1$	32.36 %	0.04 %
KNN	$k = 10$	36.80 %	0.05 %
Naive Bayes	Normal Kernel	38.85 %	0.07 %
Decision Tree	Min Leaf Size = 1	34.89 %	0.04 %
Decision Tree	Min Leaf Size = 10	37.47 %	0.06 %

Table 3.6: Comparison of classification algorithms on the feature subset [reproduced from Publication **II** with the permission of the publisher]

Setting up the final random forest model on a single partition into training and test set and recording the test set and out-of-sample forecast highlights the classification accuracies that this stock market model can obtain (Publication **II**).

Test	Class 1	Class 2	Class 3	Class 4
	62.1 %	54.6 %	18.8 %	48.3 %
46.3 %	Positive		Negative	
	82.40 %		50.00 %	
Forecast	Class 1	Class 2	Class 3	Class 4
	40.7 %	55.8 %	23.8 %	41.2 %
41.0 %	Positive		Negative	
	75.2 %		35.9 %	

Table 3.7: Average classification accuracy on the test and forecast data [modified from Publication **II**]

The results in Table 3.7 point out that the classification is distinct for each of the four classes. On an aggregated basis, classifying observations into positive return classes appears to be more accurate than into the negative classes. Overall, the results appear for most classes to be consistent between the test and forecast data. An exception is the 'strong positive' class, which only shows an accuracy of 40.7% on the out-of-sample forecast set, which is more than 20 percentage points less than in the test set. Evidently, the classification into the third class of 'slightly negative' is in both datasets clearly the least accurate assignment. Altogether, for the forecast dataset the classification accuracy into the 'slightly positive' class is the highest, followed by the two classes with high absolute returns, the 'strong negative' and 'strong positive' classes.

Based on these classification results, four trading strategies are suggested that aim to deploy the classification model's predictions to anticipate movements in the S&P500 index and benefit from them. To benefit directly from these market movements, a trader needs financial instruments or investments to replicate the behaviour of the S&P500 stock market as precisely as possible. There exist two simple possibilities to implement such an equivalent to the S&P500. The first is using an exchange traded fund (ETF), which is an inexpensive way to track markets such as the S&P500 index (Bodie, Kane, & Marcus, 2009). Another, more capital-intensive approach is to purchase/sell the (majority of the) stocks constituting the S&P500 in the proportions they are represented in this market index. Since these approaches directly aim to replicate the actual S&P500 index, transactions conducted to purchase or sell such investments will simply be referred to as 'buying the index' or 'selling the index'.

The four trading strategies can be implemented with investments replicating the S&P500 and following all or only a subset of the class predictions from the random forest model.[3] These strategies are detailed in Table 3.8.

[3] It should be noted that these strategies embody different levels of risk for an investor. An investor has to decide which strategy to pursue based on the risk & return profile of the strategy. For this research, the focus is exclusively on the returns achievable with the strategies accounting for only transaction costs.

No	Strategy
1	Strongly positive or positive returns predicted (Classes 1 & 2) - Long (buy) the index Strongly negative or negative returns predicted (Classes 3 & 4) - Short (sell) the index *After decision: Remain long or short until next decision requires a change*
2	Strongly positive returns predicted (Class 1) - Long (buy) the index Strongly negative returns predicted (Class 4) - Short (sell) the index *After decision: Remain long or short until next decision requires a change*
3	Strongly positive returns predicted (Class 1) - Long (buy) the index Strongly negative or negative returns predicted (Classes 3 & 4) - Short (sell) the index *After decision: Remain long or short until next decision requires a change*
4	Strongly positive or positive returns predicted (Classes 1 & 2) - Long (buy) the index Strongly negative returns predicted (Class 4) - Short (sell) the index *After decision: Remain long or short until next decision requires a change*
5	Benchmark: Buy-and-Hold - Long (buy) the index at start of period and retain

Table 3.8: Investment strategies for S&P500 stock market index [reproduced from Publication **II** with the permission of the publisher]

These strategies follow 'strong positive' or 'slightly positive' predictions for the upcoming trading day with a buy decision for the market. If a person is already invested, the decision is to remain 'long', meaning to stay invested in order to contribute from the expected positive market development during that day. In contrast, these strategies assume for 'strong negative' or 'slightly negative' predictions that the upcoming trading day will eventuate in a negative intraday return. Hence, the traders in these strategies 'short' the market, so sell their investment, if they are already invested, and, additionally, sell 'short' an investment they do not currently hold in order to make a profit on the expected negative intraday return. The benchmark strategy for the four trading strategies is a buy-and-hold strategy. A buy-and-hold strategy is a passive management strategy that includes buying an index/investment and holding it over the entire investment period. With this type of strategy, all positive returns but also all negative returns are incurred by the investor and contribute to the return over the entire investment period.

For the comparison of the buy-and-hold strategy with the four classification model-based strategies, transaction costs for buying and selling the index will be accounted for. Two different approaches are selected: (1) using a fixed percentage of the trade value (e.g. as in Pätäri and Vilska (2014)) and (2) using a fixed USD amount (e.g. as in Teixeira and De Oliveira (2010)). Following this approach, Publication **II** finds that with small transaction costs, it is possible to outperform a buy-and hold strategy. An example of such an outperformance is illustrated in Figure 3.13 for transaction costs of 0.1% of the transaction value.

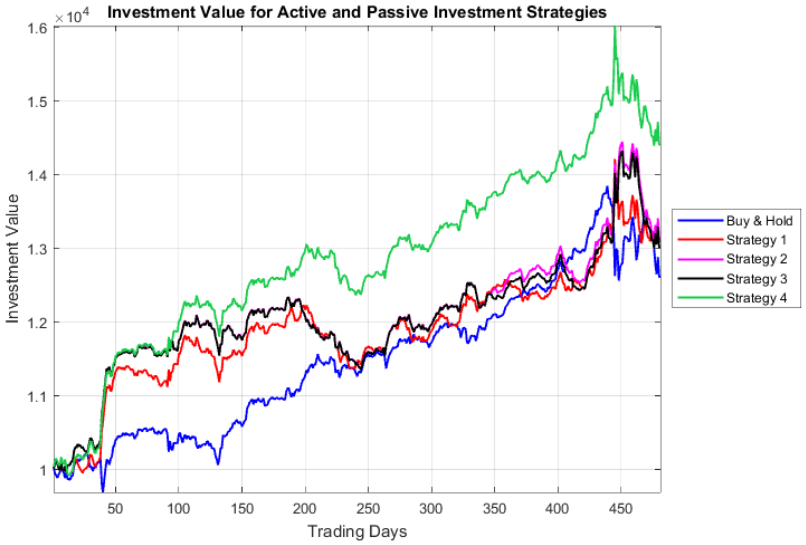


Figure 3.13: Comparison of the performance of the trading strategies for the forecasting period [reproduced from Publication **II** with the permission of the publisher]

The detailed results for the forecasting period are displayed in Table 3.9 with different levels of transaction costs.[4]

Investment Strategy		Buy & Hold	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Investment	Transactions	1	201	64	68	89
-	0	26.39 %	60.14 %	39.97 %	39.33 %	57.51 %
-	0.10 %	26.26 %	30.97 %	31.28 %	30.16 %	44.09 %
-	0.20 %	26.13 %	7.09 %	23.13 %	21.60 %	31.80 %
-	0.30 %	26.01 %	-12.46 %	15.48 %	13.58 %	20.55 %
-	0.40 %	25.88 %	-28.45 %	8.30 %	6.09 %	10.25 %
10'000	10	26.26 %	35.22 %	32.62 %	31.57 %	46.59 %
10'000	20	26.13 %	10.31 %	25.28 %	23.81 %	35.68 %
50'000	10	26.36 %	55.15 %	38.50 %	37.78 %	55.33 %
50'000	20	26.34 %	50.17 %	37.03 %	36.22 %	53.14 %

Table 3.9: Performance of trading strategies for the forecasting period [modified from Publication **II**]

[4] In Publication **II** the returns of the buy-and-hold strategy for the percentage-based transaction costs were the same for different levels of transaction costs. This was a transcription error. The actual returns for the buy-and-hold strategy differ slightly (less than 0.3% lower or higher for all transaction cost setups over the entire holding period). This did not alter either the returns for the four suggested trading strategies, the findings or conclusions presented in the original paper.

It is apparent that higher transaction costs deteriorate the return of strategies requiring frequent transactions. Thus, it is also obvious that the first strategy that acts on all class predictions and that has about three times as many transactions as Strategies 2, 3 and 4 experiences a faster return decline when the transaction costs increase. Since more transactions eventuate in higher cost, this result is intuitive. However, it is remarkable that with a small transaction cost, for instance 0.10% or 10 USD, all four investment strategies outperform the buy-and-hold strategy. Relying on all predictions, as in Strategy 1, is only the best approach when no transaction costs are present. If transaction costs are present, Strategy 4 results in the highest returns after transaction costs of all the four trading strategies. Considering the previous classification accuracies of the random forest model (see Table 3.7) offers a simple explanation for this outcome. Strategies 1 and 3 include the predictions of the 'slightly negative' class (Class 3), which are comparatively inaccurate (23.8%). In contrast to Strategy 2, which acts only on 'strong positive' and 'strong negative' return predictions, the better-performing Strategy 4 includes in addition 'slightly positive' return predictions. This comparison shows that including the 'slightly positive' return predictions is beneficial. The 'slightly positive' class shows the highest classification accuracy (55.8%) in the forecast set, making it the most reliable prediction. Hence, Strategy 4 outperforming the remaining classifier-based trading strategies is unsurprising.

The subsequent question of interest is how the class predictions contribute to the returns achieved by the random forest-based trading strategies. The contributions can be presented using the predicted classes and the actual returns in the forecast period. In Figure 3.14, the contributions are distinguished not only by the prediction of the return direction or class but also by whether the predictions were correct or incorrect (misclassification).

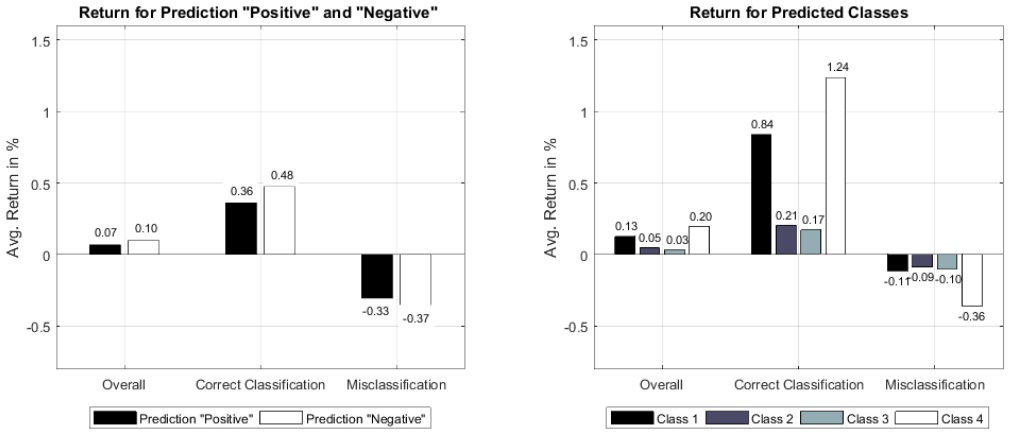


Figure 3.14: Average return contribution from direction and class predictions [reproduced from Publication **II** with the permission of the publisher]

The first graph in Figure 3.14 displays the average return achieved with positive predictions (‘strong positive’ and ‘slightly positive’ combined) and negative ones (‘strong negative’ and ‘slightly negative’ combined). It is apparent that both the positive and negative predictions lead to overall positive returns. In particular, correct classifications for positive or negative returns lead to on average high positive returns. The second graph illustrates which class predictions are associated with the largest average returns. Overall, the ‘strong positive’ and ‘strong negative’ classes show the highest average return, being 2 to 6 times larger than the slight return classes 2 and 3. This effect originates in the fact that the very large average returns achieved in the case of correct predictions for these classes is only partially offset by the negative returns experienced if these predictions are misclassifications. If they are correctly classified, the average returns achieved for classes 1 and 4 are 0.84% and 1.24%, respectively, which for both is larger than 0.5%. This is an intuitive outcome since classifying a return correctly as ‘strong positive’ means that the return is correctly classified as being in excess of 0.5%, which leads to an average return larger than 0.5%. The same logic is true for ‘strong negative’ returns, which by definition are smaller than -0.5% – meaning for a seller a profit of on average larger than 0.5%. In contrast, the consequence of misclassifying the ‘strong positive’ and ‘strong negative’ return is not necessarily a negative return. For instance, misclassifying a ‘slightly positive’ return as ‘strong positive’ leads still to a positive return – but that positive return is simply smaller in magnitude than expected. The same holds true for ‘strong negative’ returns. Hence, it is plausible that ‘strong positive’ and ‘strong negative’ predictions can contribute on average most to a trading strategy.

In conclusion, the use of a combination of feature selection with the FSAE and the classification algorithm random forest was demonstrated to yield results that can successfully be implemented into trading strategies. Moreover, it was demonstrated that, over the forecast period, all trading strategies were able to outperform a simple buy-and-hold strategy when no or only minor transaction costs were present. Thereupon, it was highlighted that the prediction of ‘strong positive’ and ‘strong negative’ returns can contribute over-proportionally to the return of a trading strategy. In addition, this suggests that considering more than two classes can be beneficial for a classification task. It also stresses that using only a subset of the predictions, such as the extreme return classes 1 and 4, can be more beneficial for a trading strategy than simply using all predictions. Finally, in future research these results can be validated and potentially be extended to other stock markets worldwide.

3.7 Conclusion and Limitations of FSAE Feature Selection

The fuzzy similarity and entropy (FSAE) feature selection algorithm was introduced as an improvement of the feature selection approach by Luukka (2011). As highlighted in detail in this chapter, the feature selection by Luukka (2011) can fall victim to the fact that it does not explicitly take into account the distance between ideal vectors (Publication

I). Deploying entropy in the approach by Luukka (2011) means that observations with high and low similarity values are dealt with equally. Thus, they are considered equally informative for a feature. This caused the approach to be vulnerable to certain data structures and levels of variance within the classes. It was demonstrated for different two- and three-class settings that even with small or moderate variance, non-overlapping classes for a feature are perceived as being less informative than a feature with completely overlapping classes and basically equal class centres. Such a feature ranking is clearly not representative of the feature's ability to discriminate between classes, and such an outcome is certainly undesirable. The objective of the FSAE algorithm was to enhance the feature selection algorithm by Luukka (2011) to add a scaling factor for the class- and feature-specific entropy values. For the artificial examples, it was detailed extensively how the scaling factor affects the features' entropy values and the feature ranking overall. In particular, it was demonstrated how this scaling factor contributed to overcoming this vulnerability by accounting for the distance between ideal vectors for each feature. In certain cases, a feature can have a high entropy value due to slight to moderate variance even though it is separating classes well due to large differences in the classes' ideal vector elements. The scaling factor for such entropy values adjusts the entropy downwards to highlight that the feature is more informative. In contrast, completely overlapping features with almost identical ideal vector values stay at the same level of informativity. In essence, the scaling factor operates as an adjustment of the informativity of features for the distance between the classes' ideal vector values for a feature.

The FSAE algorithm is conceptualized as a filter method for feature ranking but can also be deployed as a wrapper method. It was demonstrated on five medical datasets that the FSAE wrapper can achieve at least comparable mean classification accuracies to those of the wrapper version of the approach by Luukka (2011) but often with considerably fewer features. When comparing the filter version of both approaches with ReliefF, the Laplacian score and the Fisher score, it was apparent that FSAE can achieve competitive results in terms of mean accuracy. Opposed to that, the approach by Luukka (2011) was only competitive in two of the five datasets throughout all feature removal decisions. Additionally, it experienced a severe decline in mean performance of more than 10 percentage points for three of the five datasets.

Furthermore, a stock market prediction model for the S&P500 market index using the FSAE together with a random forest was implemented. The trading strategies derived from this model demonstrated the ability to outperform a buy-and-hold investment strategy with a small to moderate amount of transaction costs. Furthermore, it was highlighted that the high absolute return class predictions contributed more to the average return of the trading strategies than the two classes with smaller absolute returns.

Overall, for the artificial and real-world datasets, the FSAE feature selection algorithm exhibited the ability to detect and remove irrelevant features and to retain those that contribute to the discrimination of the classes. Notwithstanding, there remain two limitations of the FSAE feature selection method that can be problematic for certain complex data structures. The first is the fact that it relies on a single ideal vector to

represent each class in the data. If classes consist of groups of data or are characterized by large variances, then in the majority of cases a single point to represent the class will not be sufficient. For this purpose, an extension of the concept of a single ideal vector is required. The second limitation is the evaluation of a feature on a univariate basis. In the FSAE, a feature is evaluated on its own without considering how it is related to other features. In a setup where features are relevant if they are jointly present in the data and otherwise not or less relevant, this can result in a suggestion to remove such features. As mentioned previously, using the wrapper version of the FSAE may avoid such removals but will, at a minimum, result in a larger feature subset than necessary. In the worst case, the algorithm will stop prematurely and keep irrelevant features that act as noise in the data.

4 Similarity Classifier with Multiple Ideal Vectors

4.1 Introduction and Reasoning

The previous section on the FSAE feature selection algorithm highlighted that one limitation of this algorithm is the use of a single ideal vector to represent a class. Since the FSAE is conceptually related to the similarity classifier, it is unsurprising that the similarity classifier involves the same limitation. In the context of classification algorithms, Luukka and Lampinen (2015) highlighted that distance-based techniques can encounter difficulties when facing complex data structures. To address this limitation, they introduced the differential evolution-based multiple vector prototype classifier (MVDE). Essentially, the solution they proposed is to use multiple ideal vectors to represent each class in the feature space. They highlight that this approach can handle more complex data structures, such as data clusters that a simple distance-based classifier has difficulties representing and classifying accurately (Luukka & Lampinen, 2015). The main limitation of their novel algorithm is that the number of ideal vectors to represent a class has to be known to run the algorithm. In addition, Luukka and Lampinen (2015) state that their results started to deteriorate rapidly with a smaller or larger number of ideal vectors per class than the number that is actually suitable to represent the decision regions of the classes. Hence, for future research they suggested investigating ways to optimize the number of ideal vectors. In Publication **III**, this problem was addressed for the similarity classifier – but the results can clearly also be deployed or extended for the MVDE presented by Luukka and Lampinen (2015). As Luukka and Lampinen (2015) demonstrated, the number and position of ideal vectors in the feature space is pivotal to represent classes in more complex data structures well since it allows the classification of observations from such data structures accurately. We deploy a combination of the two following approaches: a clustering algorithm and the similarity classifier. The underlying idea is intuitive: The aim of multiple ideal vectors is to represent each distinct group of data within a class with one separate ideal vector. This turns out to be a twofold problem since, first, the optimal number of distinct groups in the data needs to be known, and, second, each of these groups has to be specified so that an ideal vector for each of them can be defined that represents the group well. The second problem is a typical clustering problem, where groups of observations need to be found that are similar to each other but dissimilar to observations forming another group (Dougherty, 2013). The first problem concerning the lack of knowledge on how many ideal vectors to use is equivalent to not knowing how many distinct groups or clusters are present in a class. This is essentially the research need Luukka & Lampinen (2015) formulated for the MVDE. However, this is also a clustering-specific problem since many clustering algorithms, such as K-means clustering, can suggest a partition of the data but require the number of clusters to be specified in advance (Bishop, 2006). Hence, a possible way to address the problem of defining the number and position of multiple ideal vectors per class can be addressed using a clustering algorithm together with an algorithm that determines the optimal number of clusters for a certain dataset. In Publication **III**, the authors suggest using the popular K-means clustering algorithm together with the Jump method (Sugar and James,

2003) to determine the number of ideal vectors per class as well as their position. In the suggested classifier, which is a similarity classifier with multiple ideal vectors, the position of the ideal vectors per class is set to the cluster centres. The cluster centres constitute the cluster means (for K-means clustering) and, thus, embody cluster representatives. The ideal vectors function in the same manner as in the original similarity classifier by assigning new observations to the closest ideal vector's class. Hence, the ideal vectors operate as in the original classifier, albeit there are now multiple ideal vectors per class that allow coping with more complex data structures, such as groupings in the data.

4.2 Step-by-Step Algorithm of the Novel Similarity Classifier

The flowchart of the algorithm used for the similarity classifier with multiple ideal vectors using clustering and the Jump method is illustrated in Figure 4.1. The algorithm is divided into five steps, starting from the pre-processing until the computation of the test set performance of the classifier.[5] Before detailing the algorithm as suggested in Publication **III**, it is noteworthy that this algorithm can also be seen as a framework to setup a (similarity) classifier with multiple ideal vectors. This framework is independent of which pre-processing, clustering algorithm or method to determine the number of clusters is selected.

[5] This is less than the eight steps mentioned in Publication **III**. The algorithm was simplified to five steps since the author realized that three of the steps could be removed without likely having a noticeable impact on the obtained solution and corresponding classification accuracy.

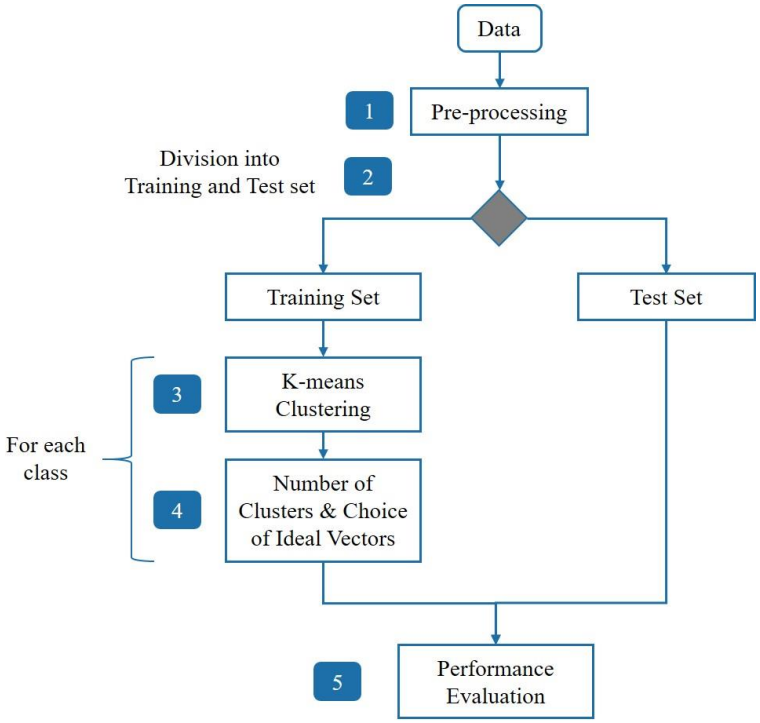


Figure 4.1: Step-by-step process underlying the similarity classifier with multiple ideal vectors [modified from Publication **III**]

The first step is the data pre-processing. In Publication **III**, two forms of pre-processing were examined – first, the standardization of the original data to the compact interval $[0,1]$ and, second, the normalization using the z-score in combination with principal component analysis (PCA) to extract new features and a subsequent standardization to $[0,1]$. It shall be noted that both approaches include a standardization to the compact interval $[0,1]$ since the data need to be scaled into this interval so that the similarity classifier can be applied to them. For the second variant of pre-processing, a suitable number of principal components has to be selected since the first several principal components are commonly sufficient to explain the majority of the variance in the data (Cangelosi and Goriely, 2007). In the literature there exist several methods for this task, including the modified broken stick model (Cangelosi and Goriely, 2007), the Guttman-Kaiser criterion (Guttman, 1954; Kaiser, 1961), the SCREE test (Cattell, 1966), the minimum average partial (MAP) test (Velicer, 1976), Bartlett's test (Bartlett, 1950) and parallel analysis (PA) (Horn, 1965). For different data structures, MAP and PA showed the highest performance of the tested methods (Zwick and Velicer, 1982, 1986; O'Connor, 2000). Hence, the PCA-based pre-processing follows the recommendation by O'Connor (2000b) to use a combination of the MAP test and PA to determine a suitable number of principal components.

The second step represents the data division into a training set to build the classification model and a test set to determine the out-of-sample classification accuracy of that model. The data can, for instance, be divided using the holdout method, with a split of 70% of observations for the training data and 30% of observations for the test set.

The third step consists of the actual clustering of the observations in each of the classes. This clustering is performed for 1 to K_{max} clusters, where K_{max} is the user-specified maximum number of clusters used within a class. Each clustering is conducted for a certain number of clusters K (with $1 \leq K \leq K_{max}$) so that the outcome of the clustering is K cluster centres denoted as $c_1, \ldots, c_K$. This step is the prerequisite for the next step (the fourth step), where the results of the clustering for the different number of clusters are needed to determine the optimal number of clusters for each class. Therefore, in this step, for each number of clusters K, the so-called estimated average distortions between the observations in a cluster to their corresponding cluster centre are calculated (Sugar and James, 2003).

$$\hat{d}_K = \frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{K}\frac{u_{jk} * \left(x_j - c_k\right)^T \Gamma^{-1}\left(x_j - c_k\right)}{p} \tag{4.1}$$

In the above equation, x_j is the j-th observation, p is the number of features[6] and c_k is the k-th cluster centre (from $k = 1$ to K) for a class. The notation Γ^{-1} refers to the inverse of the covariance matrix of the observations belonging to the k-th cluster (Sugar and James, 2003). For simplicity, instead of using the covariance matrix, the identity matrix can be used so that the calculation determines the mean squared error (Sugar and James, 2003). The notation u_{jk} denotes the crisp membership of the observation x_j to the cluster centre c_k. This membership can be formalized as:

$$u_{jk} = \begin{cases} 1 \; if \; \|x_j - c_k\|^2 < \|x_j - c_{k'}\|^2 \; for \; all \; k' \neq k \\ 0 \quad Otherwise \end{cases}, \tag{4.2}$$

where the membership of x_j to c_k is one if x_j is closer to the cluster centre c_k than to any other cluster centre $c_{k'}$ for $\forall\, k' \neq k$ and zero otherwise. The estimated average distortion $\hat{d}_K$ is the sum of the distortions over all K clusters and averaged over all n observations. After the third step, for each class the estimated average distortion for 1 to K_{max} clusters is known. Also, all clustering results are saved, including the cluster centres and cluster memberships for each number of clusters for each class.

The fourth step is the identification of the optimal number of clusters for each class. For this step, the estimated average 'distortions' for the different number of clusters from the

[6] Denoting the number of features with p follows the notation in Sugar and Gareth (2003) for the Jump method. Since the dimensionality p has an important meaning in their paper, the author refrained from changing the notation for this particular equation. However, in general, the number of features is denoted by D throughout the remaining dissertation to be consistent with the corresponding publications.

previous step are required. Based on these distortions the optimal number of clusters is determined via the Jump method (Sugar and James, 2003). To calculate the 'jump' between the distortions of two consecutive numbers of clusters, the following Equation is applied:

$$J_K = \hat{d}_K^{-Y} - \hat{d}_{K-1}^{-Y}, \tag{4.3}$$

where J_K denotes the jump in the distortion from *K*-1 to *K* clusters and *Y* the transformation power in the exponent of the distortions. The optimal number of clusters for a class is the number of clusters *K* where the largest jump J_K occurred. Since the average distortion decreases as more cluster centres are deployed, the largest jump represents the number of clusters that had the largest incremental decrease in average distortion. Hence, according to the Jump method, the number of clusters for which the largest jump occurs is the optimal number of clusters. After having applied the Jump method the optimal number of clusters, the associated cluster centres and the cluster memberships are saved. The cluster centres for each class constitute the ideal vectors that represent this class. Thus, they embody the class representatives that observations are compared to in order to assign observations to classes. The third and fourth steps are conducted for each class.

The fifth step is the performance evaluation of the similarity classifier using multiple ideal vectors. After having repeated the third and fourth steps for each class, *O* ideal vectors were determined, and each is one of the representatives of a class. Since this extension of the similarity classifier allows multiple ideal vectors per class, the number of ideal vectors $O \geq N$, where *N* is the number of classes. To assign observations to classes, they are initially compared with all *O* ideal vectors. For a feature *d* and an observation denoted by x_j, the similarity of this observation's *d*-th feature value with the *d*-th feature value of the ideal vector *o* is computed as (see also Equation (2.15)):

$$S(x_{j,d}, v_{o,d}) = \sqrt[p]{\left(1 - \left|x_{j,d}^p - v_{o,d}^p\right|\right)}, \tag{4.4}$$

where $v_{o,d}$ denotes the *d*-th element of the *o*-th ideal vector. The similarity of the entire observation x_j with the *o*-th ideal vector v_o is calculated as the average value over all *D* features.

$$S(x_j, v_o) = \left(\frac{1}{D}\sum_{d=1}^{D} S(x_{j,d}, v_{o,d})^m\right)^{\frac{1}{m}}, \tag{4.5}$$

where *m* is the parameter for the generalized mean function. In the same way, the observation x_j is compared to all *O* ideal vectors and then assigned to the cluster that it is most similar to.

$$Cl(x_j) = \arg\max_{o=1,\ldots O} S(x_j, v_o) \tag{4.6}$$

Finally, the observation is assigned to the class that corresponds to the cluster this observation is most similar to. This is executed in terms of a simple mapping from the cluster to the corresponding class.

$$C(x_j) = f\left(Cl(x_j)\right) \tag{4.7}$$

The performance evaluation is then implemented by comparing the predicted class label of the observations in the test set with the actual class labels and calculating the corresponding classification accuracy.

4.3 Application to Artificial Data

To demonstrate the ability of the novel similarity classifier with multiple ideal vectors to detect and represent multiple decision regions for each class, three artificial examples were created. Each example is characterized by multiple decision regions for each class in a two- or three-dimensional setting. All three examples are visualized in Figure 4.2.

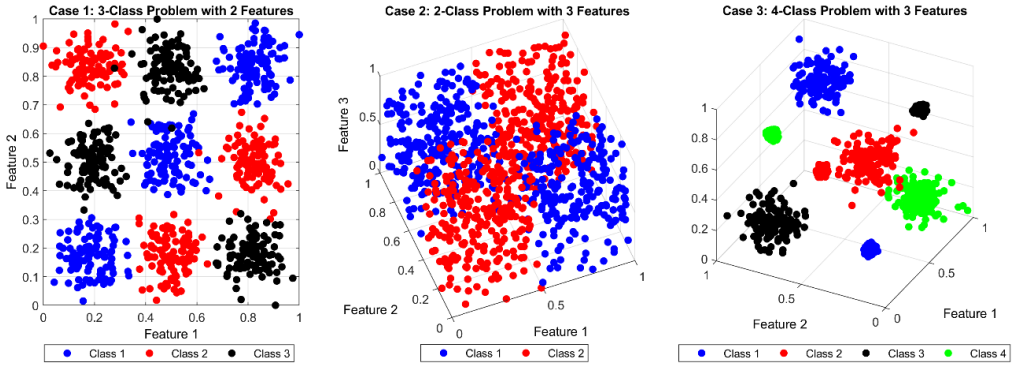


Figure 4.2: Artificial datasets for the similarity classifier with multiple ideal vectors [reproduced from Publication **III** with the permission of the publisher]

The observations for each grouping in the first example are drawn from a multivariate normal distribution without covariance. Each of the three classes are characterized by three almost entirely non-overlapping groups of observations in two-dimensional space. It is apparent that a single ideal vector for each class would neither be representative of the class nor would it support the classification of new observations. The second example is generated from a uniform distribution and illustrates a three-dimensional numerical

XOR problem with two classes. This example is composed of two distinct decision regions for each of the two classes and, hence, cannot be represented well by one ideal vector per class. The last example is another three-dimensional classification task. It differs from the previous two examples in two aspects. First, its observations are drawn from a multivariate normal distribution with different variances for different clusters. Second, it is a four-class problem, and the groups of observations are rather well separated and non-overlapping. Once more, none of the four classes can be represented well by a single ideal vector. On top of that, this example was selected to highlight that the difference in variance does not pose a problem for the novel similarity classifier with multiple ideal vectors when the classes are separated well enough.

As mentioned in above Section 4.2, the novel similarity classifier with multiple ideal vectors can be used with two different pre-processing versions. The first is a simple standardization of the features into the unit interval. The second approach is using the z-score and conducting a PCA, selecting a suitable number of principal components and subsequently standardizing them. Both approaches are implemented for the three artificial datasets for the novel similarity classifier, and the outcome is also contrasted with the original similarity classifier with a single ideal vector per class. The results are summarized in Table 4.1 to Table 4.3.

Classifier	Number of PCs (PCA)	Mean Accuracy	Variance	Transformation Power Y
Standard	-	0.3247	0.0037	-
Standard - PCA	2	0.3116	0.0039	-
Novel	-	0.9697	0.0001	p/2
Novel - PCA	2	0.9912	0	p/2
Novel	-	0.9687	0.0001	1
Novel - PCA	2	0.9913	0	1

Table 4.1: Classification accuracy of the similarity classifier with multiple ideal vectors on the first artificial example. ‘Standard’ refers to the standard similarity classifier, ‘Novel’ to the novel similarity classifier with multiple ideal vectors and ‘Standard-PCA’ and ‘Novel-PCA’ to the combination of each of these classifiers and the principal components from PCA [modified from Publication **III**]

The mean classification accuracies for the first example show that the standard similarity classifier with a single ideal vector and any of the two pre-processing approaches fails to classify the observations well. It is obvious that, for this three-class problem, the mean accuracy is not significantly different from a random assignment. In contrast, all setups of the novel similarity classifier with multiple ideal vectors, with and without PCA and with a transformation power of $p/2$ (dimensionality of the data divided by 2) or 1, achieve a significantly higher mean accuracy close to 1.

Classifier	Number of PCs (PCA)	Mean Accuracy	Variance	Transformation Power Y
Standard	-	0.5142	0.0006	-
Standard - PCA	3	0.4824	0.0005	-
Novel	-	0.9049	0.0009	p/2
Novel - PCA	3	0.9028	0.0011	p/2
Novel	-	0.9653	0.0004	1
Novel - PCA	3	0.9613	0.0001	1

Table 4.2: Classification accuracy of the similarity classifier with multiple ideal vectors on the second artificial example. 'Standard' refers to the standard similarity classifier, 'Novel' to the novel similarity classifier with multiple ideal vectors and 'Standard-PCA' and 'Novel-PCA' to the combination of each of these classifiers and the principal components from PCA [modified from Publication **III**]

In the second artificial example, a similar result as in the first example can be observed. The similarity classifier with a single ideal vector in both setups achieves an accuracy of around 50%. In contrast, all novel similarity classifiers with multiple ideal vectors achieve mean accuracies of more than 90%. In addition, using a transformation power of 1 instead of $p/2$, meaning in this example 3/2, results in an approximately 6 percentage points higher mean accuracy.

Classifier	Number of PCs (PCA)	Mean Accuracy	Variance	Transformation Power Y
Standard	-	0.3493	0.0081	-
Standard - PCA	3	0.3135	0.0085	-
Novel	-	1	0	p/2
Novel - PCA	3	1	0	p/2
Novel	-	1	0	1
Novel - PCA	3	1	0	1

Table 4.3: Classification accuracy of the similarity classifier with multiple ideal vectors on the third artificial example. 'Standard' refers to the standard similarity classifier, 'Novel' to the novel similarity classifier with multiple ideal vectors and 'Standard-PCA' and 'Novel-PCA' to the combination of each of these classifiers and the principal components from PCA [modified from Publication **III**]

The third example once more highlights the inability of a single ideal vector to classify observations for a class with multiple decision regions well. While the novel similarity classifier uses clustering to find the decision regions of each class and assigns observations to classes without any misclassification, the standard classifier achieves an accuracy of only 31 to 35%. The weak performance of the similarity classifier is premised

on the fact that the ideal vector of all four classes is around the centre of the graph $(0.5,0.5,0.5)$ so that even a small variation in the data can completely change the class assignment. Certainly, this is not desired and stresses that the standard algorithm cannot account for data structures more complex than a single decision region per class. The standard similarity classifier can only perform well in an environment where the classes are well separated from each other, and each class forms a single decision region.

The three presented artificial examples clearly emphasize the ability of the similarity classifier with multiple ideal vectors to account for multiple decision regions. With both pre-processing setups, it demonstrated that it can classify the observations in a multi-decision region problem better than the standard similarity classifier, which relies solely on a single ideal vector.

4.4 Application to Credit Risk Data

The similarity classifier with multiple ideal vectors is also applied to three credit risk datasets to demonstrate its ability to perform well in a real-world data setting. All three datasets were obtained from the UCI Machine Learning Repository (Lichman, 2013) and address the credit approval decision or the quality of borrowers. Such problems are relevant for financial institutions since these institutions need methods to support their decision making for lending to and monitoring their borrowers (Tsaih *et al.*, 2004; West, Dellana and Qian, 2005). A related problem is credit card fraud, another topic that is of pivotal interest for financial institutions (Maes, Tuyls and Vanschoenwinkel, 2002; Pun and Lawryshyn, 2012).

The first dataset is the 'Credit Approval Data Set', which is a binary classification task related to the acceptance and rejection of credit applications. The dataset contains 690 observations of 15 features that characterize the loan applicant and the loan application itself. Incomplete observations with missing values were removed so that 653 complete observations remained for the analysis.

The second dataset is the 'Statlog (German Credit Data) Data Set' in its adjusted version, where categorical variables were transformed into integer-valued ones. The binary classes embody whether a borrower is considered a good or bad debtor. The dataset contains 1000 complete observations of 24 numeric features in which the classes are unbalanced – with 70% of borrowers belonging to the non-defaulting borrowers and the remaining 30% to the defaulted creditors.

The third dataset is an adjusted version of the first dataset and is referred to as the 'Statlog (Australian Credit Approval) Data Set'. It represents a binary classification problem concerning the acceptance or rejection of credit card applicants. It encompasses 690 complete observations of 14 features that characterize the credit card applicant and his/her application details.

For these three datasets, the standard similarity classifier with a single ideal vector and the novel similarity classifier with multiple ideal vectors are contrasted against several well-known classification methods. These include the K-nearest neighbour classifier

(Cover & Hart, 1967), the naive Bayes classifier (Russell and Norvig, 2009), a decision tree (Quinlan, 1986) and the ensemble classifier random forest, which consists of multiple decision trees (Breiman, 2001).

The results for the first dataset, the 'Credit Approval' data, are illustrated in Table 4.4.

Classification Algorithm	Mean Accuracy	Variance	Mean FNR	Mean FPR	p	m	Y
Standard Similarity Classifier	0.8599	0.0004	0.196	0.072	6	4	-
Standard Similarity Classifier (PCA, 4 PCs)	0.8057	0.0005	0.01	0.284	1	1	-
Novel Similarity Classifier	0.8525	0.0005	0.133	0.160	1	1	p/2
Novel Similarity Classifier	0.8706	0.0005	0.056	0.190	6	5	1
Novel Similarity Classifier (PCA, 4 PCs)	0.7716	0.0008	0.243	0.216	1	2	p/2
Novel Similarity Classifier (PCA, 4 PCs)	0.8076	0.0005	0.324	0.085	4	4	1
K-Nearest Neighbour, k = 1	0.8184	0.0005	0.207	0.161	-	-	-
K-Nearest Neighbour, k = 10	0.8608	0.0004	0.142	0.137	-	-	-
K-Nearest Neighbour, best k = 1	0.8184	0.0005	0.207	0.161	-	-	-
Naive Bayes *(Normal Gaussian distribution)*	0.8039	0.0006	0.321	0.093	-	-	-
Naive Bayes *(Kernel with normal smoothing)*	0.6823	0.0012	0.425	0.230	-	-	-
Random Decision Forest (Min leaf size = 1)	0.8733	0.0004	0.129	0.125	-	-	-
Random Decision Forest (Min leaf size = 10)	0.8708	0.0004	0.126	0.132	-	-	-
Decision Tree (Min leaf size = 1)	0.8322	0.0007	0.194	0.147	-	-	-
Decision Tree (Min leaf size = 10)	0.8561	0.0005	0.157	0.133	-	-	-

Table 4.4: Classification results for the 'Credit Approval' dataset [modified from Publication **III**]

The results indicate that on the first dataset the standard and the novel classifier perform considerably better in terms of mean accuracy when the pre-processing is implemented via standardization instead of using the z-score combined with a PCA. The classification accuracy of the novel similarity classifier (with $Y = p/2$) is at 87.06% – higher than the 85.99% accuracy achieved with the standard classifier. Using the one-sided Welch's test, it can be demonstrated that the mean performance of the novel classifier is significantly larger (p-value < 0.01) than that of the standard similarity classifier.

In comparison with the other well-known classification algorithms, the performance of the similarity classifier with multiple ideal vectors is competitive. Comparing the novel algorithm using transformation power $Y = 1$ with the K-nearest neighbour classifiers, naive Bayes and decision trees highlights that it has the highest mean accuracy for all these single classifiers. In particular, the positive difference in the mean accuracy of the novel similarity classifier with multiple ideal vectors to these algorithms is highly significant. In addition, the mean performance of the novel algorithm with transformation power $Y = 1$ has an accuracy of 87.06% – competitive with the result achieved by the ensemble classifier random forest, which achieved with 87.33% and 87.08% the best

mean accuracies for this example. This is remarkable since the random decision forest is a classifier ensemble containing multiple classifiers (here: 50 decision trees). Opposed to that, the similarity classifier with multiple ideal vectors is only a single classifier but still achieves competitive results.

Contrasting the mean accuracies for the standard and the novel similarity classifier for both pre-processing procedures, it is apparent that using PCA and selecting four principal components underperforms simple standardization by 5.42 percentage points to 8.09 percentage points.

The results obtained for the second real-world dataset, the 'German Credit' data, are summarized in Table 4.5.

Classification Algorithm	Mean Accuracy	Variance	Mean FNR	Mean FPR	p	m	Y
Standard Similarity Classifier	0.7263	0.0003	0.099	0.683	4	1	-
Standard Similarity Classifier (PCA, 8 PCs)	0.7299	0.0004	0.142	0.570	3	5	-
Novel Similarity Classifier	0.6822	0.0005	0.158	0.691	8	1	p/2
Novel Similarity Classifier	0.7314	0.0003	0.095	0.674	4	1	1
Novel Similarity Classifier (PCA, 8 PCs)	0.6966	0.0008	0.281	0.355	3	1	p/2
Novel Similarity Classifier (PCA, 8 PCs)	0.6998	0.0006	0.298	0.304	2	1	1
K-Nearest Neighbour, k = 1	0.6715	0.0005	0.237	0.543	-	-	-
K-Nearest Neighbour, k = 10	0.7164	0.0005	0.162	0.568	-	-	-
K-Nearest Neighbour, best k = 1	0.6715	0.0005	0.237	0.543	-	-	-
Naive Bayes *(Normal Gaussian distribution)*	0.7233	0.0006	0.229	0.388	-	-	-
Naive Bayes *(Kernel with normal smoothing)*	0.7068	0.0001	0.013	0.947	-	-	-
Random Decision Forest (Min leaf size = 1)	0.7584	0.0003	0.096	0.581	-	-	-
Random Decision Forest (Min leaf size = 10)	0.7516	0.0003	0.069	0.668	-	-	-
Decision Tree (Min leaf size = 1)	0.6946	0.0007	0.218	0.510	-	-	-
Decision Tree (Min leaf size = 10)	0.7197	0.0006	0.167	0.545	-	-	-

Table 4.5: Classification results for the 'German Credit' dataset [modified from Publication **III**]

Once more, the highest mean accuracy is achieved with the random forest classifier with 75.84% and 75.16% for the two different minimum leaf sizes. For the remaining algorithms that rely on a single classifier, the similarity classifier with multiple ideal vectors and $Y = 1$ reaches the highest mean accuracy of 73.14%. Using the Welch's test, the null hypothesis that the population mean accuracy for the novel classifier is equal to or smaller than the one for the standard similarity classifier can be rejected with a significance level of 5%. It is noteworthy that the performance of the standard similarity classifier improves when the pre-processing is switched to PCA, whereas for the novel similarity classifier it deteriorates by about 3 percentage points compared to the best setup

with $Y = 1$. Another point of interest is that the standard similarity classifier with a single ideal vector performs well for this classification task and is characterized by a higher mean accuracy than the K-nearest neighbour classifier, naive Bayes and the decision tree classifier.

The classification accuracies for the third and last credit dataset, the 'Australian Credit' dataset, are presented in Table 4.6.

Classification Algorithm	Mean Accuracy	Variance	Mean FNR	Mean FPR	p	m	Y
Standard Similarity Classifier	0.8727	0.0004	0.144	0.114	3	3	-
Standard Similarity Classifier (PCA, 3 PCs)	0.8283	0.0005	0.268	0.094	1	2	-
Novel Similarity Classifier	0.8469	0.0005	0.151	0.155	1	1	p/2
Novel Similarity Classifier	0.8737	0.0004	0.118	0.133	2	3	1
Novel Similarity Classifier (PCA, 3 PCs)	0.7940	0.0006	0.273	0.152	1	3	p/2
Novel Similarity Classifier (PCA, 3 PCs)	0.8273	0.0004	0.228	0.128	1	1	1
K-Nearest Neighbour, k = 1	0.7997	0.0005	0.223	0.182	-	-	-
K-Nearest Neighbour, k = 10	0.8513	0.0004	0.177	0.126	-	-	-
K-Nearest Neighbour, best k = 1	0.7997	0.0005	0.223	0.182	-	-	-
Naive Bayes *(Normal Gaussian distribution)*	0.8016	0.0005	0.329	0.093	-	-	-
Naive Bayes *(Kernel with normal smoothing)*	0.6877	0.0015	0.417	0.228	-	-	-
Random Decision Forest (Min leaf size = 1)	0.8676	0.0004	0.143	0.124	-	-	-
Random Decision Forest (Min leaf size = 10)	0.8653	0.0004	0.153	0.12	-	-	-
Decision Tree (Min leaf size = 1)	0.8307	0.0006	0.194	0.149	-	-	-
Decision Tree (Min leaf size = 10)	0.8483	0.0005	0.164	0.142	-	-	-

Table 4.6: Classification results for the 'Australian Credit' dataset [modified from Publication **III**]

For this dataset, the novel similarity classifier with $Y = 1$ demonstrates with 87.37% the highest mean accuracy of all classification models – including both random forest setups. It is remarkable that the difference in mean accuracy of the novel similarity classifier with $Y = 1$ to the random forest is even highly significant. In general, this algorithm outperforms all classifiers by a highly significant margin, with the sole exception of the runner-up, which is the standard similarity classifier with a mean accuracy of 87.27%. Once more, the pre-processing deploying a subset of the principal components results in lower mean accuracies than a simple standardization. Moreover, for all three examples, selecting a transformation power of $Y = 1$ outperforms the parameter value $Y = p/2$.

Looking at the results computed for all three real-world financial datasets, the similarity classifier with multiple ideal vectors (and $Y = 1$) performed at least as good as the similarity classifier with a single ideal vector. For one dataset, the 'Australian Credit' dataset, the novel similarity classifiers accomplished 'only' comparable results. For the

remaining two datasets, the 'Credit Approval' and 'German Credit' datasets, the similarity classifier with multiple ideal vectors (and $Y = 1$) outperformed the standard similarity classifier with a highly significant difference for the former and a significant difference for the latter. Contrasting the results of the novel algorithm (and $Y = 1$) with the well-known classification algorithms included in this study, it demonstrated, in the majority of cases, competitive results. In particular, it outperformed the K-nearest neighbour classifiers, naive Bayes and the decision trees for all presented examples.

4.5 Conclusion and Limitations of the Novel Similarity Classifier

As pointed out by Luukka and Lampinen (2015), simple distance-based techniques may experience difficulties with more complex data structures composed of multiple decision regions for each class. The novel similarity classifier with multiple ideal vectors was introduced to address such complex data structures. The suggested algorithm for the similarity classifier with multiple ideal vectors incorporates intra-class K-means clustering in combination with the Jump method to determine the optimal number and position of the cluster centres. These cluster centres function as ideal vectors that are deployed as class representatives in the novel classifier. This is related to the research need formulated by Luukka and Lampinen (2015) for the MVDE algorithm for which the number of ideal vectors needs to be user-specified. In addition, these authors remarked that, for their datasets, a deviation from the optimal number of clusters eventuated in rapid declines in their classification results.

In response to this problem, the methodology for the similarity classifier with multiple ideal vectors incorporates a clear framework how clustering and algorithms to determine the optimal number of clusters can be deployed to select a suitable number of class representatives and their coordinates. For a distance- or similarity-based classifier, the cluster centres can function as the ideal vectors that represent the groups discovered within each of the classes. Certainly, if desired, the class representatives can also be selected in any other way based on the partitioning of a cluster algorithm.

The novel similarity classifier enables the use of multiple ideal vectors instead of just a single one for the case where two or more distinct decision regions are present within a class. Aside from that, the novel algorithm with multiple ideal vectors demonstrated on three artificial example datasets with multiple decision regions per class the consistent ability to outperform the standard similarity classifier. The two forms of pre-processing demonstrated no clear difference for these three examples.

For the real-world datasets, using the novel similarity classifier with simple standardization and a transformation power $Y = 1$ led in all cases to the best mean classification accuracy of all similarity-based classifiers. In particular, in five out of six setups, the novel similarity classifier performed better with simple standardization than with a selected number of principal components. For this reason, it is suggested to apply only the simple standardization for the similarity classifier and the transformation power $Y = 1$ for the Jump method. The deployment of other transformation powers for the Jump

method can be examined in future research and is dependent on the distribution of the clusters in the data.

It is also noteworthy that, with exception of the ensemble learning algorithm random forest, the similarity classifier with multiple ideal vectors and $Y = 1$ outperformed all benchmark algorithms, often by a highly significant margin. In comparison with the random forest setups, the novel similarity classifier was in one case comparably accurate, in the second it underperformed by a margin of 2 percentage points and another time even outperformed it.

The similarity classifier with multiple ideal vectors addresses one of the main limitations of the similarity classifier, which is the inherent assumption that a single data point can represent a class well. However, using a Euclidean distance-based clustering approach, such as the standard K-means algorithm, as well as measuring similarity without incorporating the covariance structure of the data are the main limitations of the novel classifier. These two limitations are directly related. Both illustrate that the novel similarity classifier can in future work be extended to incorporate the level of variation and correlation in the data for a more effective clustering and classification of complex data structures. Moreover, different clustering algorithms can be applied that are more effective in the context of non-convex clusters. Other methods to determine the optimal number of clusters can be applied as well. As a last point, the idea to combine clustering with a method to select the optimal number of clusters to find one or more suitable class representatives can also be extended to the context of supervised feature selection.

5 Clustering One Less Dimension (COLD) Feature Selection

5.1 Introduction and Reasoning

The differentiation between univariate and multivariate feature selection methods highlights that feature selection can be conducted either with a focus on each feature separately or by considering feature interactions. This is pivotal when considering that the subset of the *k* most relevant features (univariate) does not need to be the same as the best subset of *k* features (Cover, 1974). For redundant features, this appears obvious. The best subset of two features is evidently not a subset containing two redundant features, which are each by themselves the best standalone feature (Cover, 1974). Instead, for instance, the best single feature together with any other feature that adds at least marginal additional information and supports the discrimination between the classes would be preferable. However, this is not the only case where the best *k* features are not necessarily the best subset of *k* features. Elashoff, Elashoff and Goldman (1967) demonstrate this circumstance also for (conditionally) independent features. In addition, Toussaint (1971) extends this finding by illustrating that the subset of the *k* best features does not necessarily contain the univariate best feature and can even consist of the two worst univariate features. Guyon and Elisseeff (2003) present simple two-dimensional examples and show how independent as well as highly correlated features can together separate the two-dimensional space linearly even though each feature by itself is completely useless. This illustrates a somewhat extreme example – but it exemplifies a valid point. For their example, adding a feature that only partially overlaps for the two classes and that could reach high classification accuracies close to 100% should not be contained in the best subset of two features. Rather, two features that are essentially useless by themselves due to high correlation should be selected for the best subset of two features since they can jointly separate the classes linearly. Notwithstanding, each of these features alone will result in an accuracy of about 50%, which is the same as a random class assignment for a balanced two-class problem. Thus, Guyon and Elisseeff (2003) reached the conclusion that feature selection methods evaluating features with a univariate approach are not suitable for determining the best feature subset for such data structures. For certain applications, simply selecting the feature subset with a univariate or multivariate method can result in the same best feature subset. However, the depicted findings and examples indicate that multivariate feature selection methods considering the dependencies between two or more features can in other applications suggest better feature subsets. As a consequence, a feature selection algorithm should not evaluate features without factoring in the dependencies among features for the selection of the best feature subset.

Hence, the objective of the heuristic feature ranking method proposed in this section is to be able to account for the dependencies among features to find the best feature subset. Before going into detail about the way the so-called COLD algorithm functions, it should

be shown how the novel algorithm is linked to the research highlighted in previous sections of this dissertation. As explained in detail in Section 3, using the distance between class representatives, as implemented in the FSAE, can render the feature selection more effective and lead to more intuitive feature removal decisions. In the COLD algorithm, this logic is coupled with the concept of multiple ideal vectors per class from the novel similarity classifier (see Section 4). This means that instead of using a single class representative, as in the FSAE, COLD will deploy multiple class representatives to determine which features are relevant. As in the similarity classifier with multiple ideal vectors, these class representatives are identified using class-wise clustering. Since Guyon and Elisseeff (2003) highlight several challenges that relate to the dependency among features caused by correlation, the covariance structure of the clusters is also considered. This provides additional information on the shape of each cluster beyond the knowledge of one representative centre point.

The acronym COLD stands for 'clustering one less dimension' and hints at the logic underlying this algorithm (Publication **III**). As the name indicates, one defining characteristic of this supervised feature selection algorithm is that it deploys clustering. In the algorithm, clustering is conducted for each class separately. The clustering is implemented using all features to determine groupings within each class. Each of these groupings can be represented by an ideal vector (cluster centre) and the covariance matrix that defines the distribution of observations in that cluster. The second part of the name, 'one less dimension', refers to how the feature relevance is determined accounting for dependencies among features. First, the separation in terms of the Mahalanobis distance between each of the clusters of a class to all other classes' clusters is calculated, which is based on the complete set of features. For this purpose, the Mahalanobis distance accounts for the ideal vector values of each cluster and the corresponding covariance matrices. Subsequently, the separation among the same clusters is measured excluding a certain feature from the full set of features. This step is repeated for each feature, always looking at the entire feature set excluding only the feature currently under consideration. Then, the separation between the clusters of the classes with the complete set of features is compared to the separation achieved with the feature subset without a certain feature. The interpretation of the comparison is straightforward. If the clusters from different classes without a certain feature are overall closer, meaning the classes are less well separated, then the feature is by itself or together with one or more of the features in the dataset relevant and contributes to the separation of the classes. It is noteworthy that pursuing this approach is not dependent on whether there is a dependency between two features or among several ones. If a feature is dependent on one or multiple features that contribute to the separation among classes, then removing such a feature will deteriorate the separation of the classes in terms of distance. In contrast, if the separation between the classes remains unchanged or improves after the feature removal, the feature is irrelevant. On top of that, this feature might even act as noise in the data, which renders the discrimination between classes more challenging.

Since the general idea of using clustering for feature selection is not entirely new, the approach of the COLD algorithm has to be contrasted with other algorithms in the

scientific literature to understand how the COLD algorithm differs from them and why it was conceptualized in that way. The existing research on feature selection includes clustering for unsupervised (Mitra, Murthy and Pal, 2002) as well as supervised feature selection (Martínez Sotoca and Pla, 2010; Sahu, Dehuri and Jagadev, 2017; Chormunge and Jena, 2018). For supervised feature selection, meaning the domain in which also the COLD algorithm is applied, the existing research centres on using clustering of features. This refers to the grouping of features to find representatives for similar features and to keep only a subset of representative features (Martínez Sotoca and Pla, 2010; Sahu, Dehuri and Jagadev, 2017; Chormunge and Jena, 2018). In this way, the identified subset of features contains those features that may represent several features that are similar, meaning that they are considered (potentially) redundant. In addition, in Chormunge and Jena (2018), features that do not fit any cluster are considered irrelevant and removed as well. A different approach is taken by Martínez Sotoca and Pla (2010), who interpret one cluster as the cluster of the 'residual features' since it shares the least amount of information with the class label and discard all features in that cluster entirely.

The COLD algorithm differs from all these approaches for supervised feature selection using clustering in three distinct aspects.

1. Clustering of observations: The COLD algorithm clusters the observations in the data, where the features are the characteristics that lead to the formation of clusters. In the approaches found in the existing research, the features themselves are clustered and not the observations. This difference is premised on the fact that the aim is not to find groups of features to simply take a representative of each feature but to find groups of observations and the representatives of these groups to later determine which features contribute to the separation of these groups from each other.

2. Clustering of each class separately: To determine representatives for each class, 'pure' clusters are needed that only contain observations from one class. It is obvious that not for every data structure clustering of all observations will end up with such 'pure' clusters. Hence, it seems natural to cluster each class separately. Using class-wise clustering, a representative of each cluster specifically embodies a group of observations of only a single class. Also, the covariance matrix for that cluster is based only on observations of the same class. Finally, clustering for each class can capture more complex data structures/groupings for each class and can represent these with several class representatives (and the corresponding covariance matrices) instead of a single representative. The idea behind this is that for complex data structures, allowing multiple class representatives will be more suitable than a single representative to capture the structure of each of the classes in the data.

3. Integration of the clusters into the feature selection algorithm: In the approaches presented in the literature, clustering is directly deployed to discard features. This is accomplished either by keeping only a representative feature for each cluster, by discarding features that do not fit any of the clusters, by removing features that are highly correlated or by discarding all features that belong to a cluster of 'residual features'. In

contrast, COLD takes the clustering result as a starting point to determine each feature's contribution to the separation of the clusters of the classes. Since COLD aims to determine a feature's multivariate relevance, the algorithm compares the cluster separation with all features to the separation without each of the features. In this way, the impact of a certain feature on the set of features can be measured.

Based on these points, it is apparent that the COLD algorithm takes an approach to using clustering for feature selection entirely different from the algorithms presented in the literature. In the next section, the detailed step-by-step algorithm that implements the logic behind COLD will be depicted.

5.2 Step-by-Step Algorithm of COLD Feature Selection

The COLD algorithm can be divided into five distinct steps, which start with the pre-processing and end with the COLD scores as well as the feature ranking. The steps are outlined in Figure 5.1 and explained in detail below.

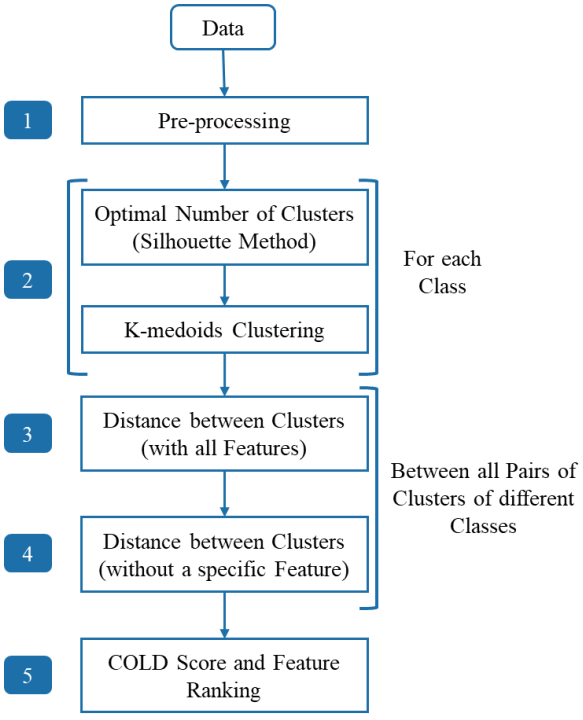


Figure 5.1: Step-by-step process underlying the COLD feature selection

The **first step** is the pre-processing of the data. Initially, the data are scaled into the compact interval [0,1]. Thereafter, redundant features are removed. The term 'redundant'

for a feature is interpreted as a feature that can be perfectly represented by another feature or a linear combination of multiple other features in the data. Fortunately, such linearly dependent features can be determined with a rank-revealing QR factorization (Chan, 1987) and then discarded. Hence, only linearly independent features are kept in the data for the subsequent steps. For the final feature ranking, these discarded features will be ranked last.

The **second step** concerns the clustering of the data and the choice of the number of clusters for each class. The clustering is conducted for each class using the K-medoids clustering algorithm (Bishop, 2006). K-medoids is a generalization of the K-means clustering algorithm, which is one of the most widely applied and popular hard clustering algorithms (Koutroumbas and Theodoridis, 2003; Dougherty, 2013). There are two main differences between K-medoids and K-means, which are the reason for the choice of K-medoids clustering for the COLD algorithm. The first is that K-means requires numerical features, whereas K-medoids can also handle categorical features (Hastie, Tibshirani and Friedman, 2009). The second reason is that K-means is sensitive to outliers, which impact the centroids used in K-means. In contrast, K-medoids is robust to outliers and noise since it utilizes medoids (Hastie, Tibshirani and Friedman, 2009; Sammut and Webb, 2017). It is well known that for clustering algorithms such as K-medoids or K-means, the number of clusters has to be user-specified ahead of clustering. In other words, the number of groups in the data to be assumed for the clustering has to be defined in advance. It is apparent, that this is rarely known in advance. Without an investigation of the data, it may not be possible for a user to specify what a suitable number of clusters is. In particular, high-dimensional data are extremely difficult, if not impossible, to investigate visually to determine a suitable number of clusters. Fortunately, there are several methods discussed in the scientific literature that aim to determine the optimal number of clusters for a given dataset. Some numerical and commonly used methods include the GAP-statistic (Tibshirani, Walther and Hastie, 2001), the Silhouette method (Rousseeuw, 1987), the Calinski-Harabasz index (Calinski and Harabasz, 1974) or the Jump method (Sugar and James, 2003). For the COLD algorithm, the Silhouette method was selected. The Silhouette method is, in the author's view, a highly intuitive method, and it also determines the number of clusters independently of the clustering algorithm but based on the partitioning of the data. In other words, it can be used with any clustering algorithm, and the optimal number of clusters depends only on the clusters suggested by the deployed clustering algorithm. In addition, in a comparison of several methods to determine the optimal number of clusters, the Silhouette method has demonstrated on average a very good performance, which was even significantly higher than that of several other methods (Arbelaitz *et al.*, 2013).[7]

[7] The Jump method that was deployed in the similarity classifier with multiple ideal vectors to determine the optimal number of clusters was not considered. The reason for this choice is that the Jump method requires the specification of the transformation power Y. The results presented in the previous section indicated that the transformation power of Y = p/2 proposed by Sugar and James (2003) did not perform well for the real-world datasets. Thus, the author decided to use the Silhouette method, which does not require the specification of a parameter and demonstrated good performance in the existing literature.

For the Silhouette method, the candidates for the optimal number of clusters need to be user-specified. For this purpose, 1 to *K* clusters are determined as candidates, where *K* can, for instance, be set to 10 (or based on the sample size). For the COLD algorithm, both the K-medoids clustering and the Silhouette method are implemented using Mahalanobis distance as the distance measure. In contrast to certain other distance measures, such as Euclidean distance, the covariance information is included in the calculation of Mahalanobis distance. In this way, the covariance among features and differences in variance can be accounted for. To calculate the Mahalanobis distance between two cluster centres, the covariance matrix has to be invertible. For the entire dataset, the QR factorization ensures that all features are linearly independent, which guarantees that the covariance matrix is invertible. However, this is not necessarily true for the clusters, which obviously contain only a subset of the observations. Thus, the clustering can suggest one or more clusters that have a covariance matrix that is not invertible. For instance, a feature can be constant in one or several of the clusters. However, such a feature can still be desirable for the classification. A simple example would be that for one class, all clusters contain one feature that is constantly '0', whereas for the other class it is constantly '1'. Obviously, this feature can by itself perfectly discriminate between the classes. Nonetheless, such a feature eventuates in non-invertible (singular) covariance matrices of the clusters. A remedy for this problem is ridge regularization (Warton, 2008) of the singular covariance matrices.

$$\Sigma_k = (1 - \alpha)\Sigma_k + \alpha I \tag{5.1}$$

In Equation (5.1), I stands for the identity matrix (a diagonal matrix with ones on the diagonal and zeros as all off-diagonal elements). The parameter α controls how much of the identity matrix is added to the singular covariance matrix to ensure that it is invertible. The regularization is applied only to singular covariance matrices. All other covariance matrices can remain unaltered for the calculation of the Mahalanobis distance. The covariance matrix is $D \times D$ dimensional, where D is the number of features that remain after step one and the removal of redundant features.

The result of the second step and using K-medoids with the Silhouette method is the knowledge of the cluster centres (ideal vectors) for the optimal number of clusters for each class, the cluster membership of observations to these clusters and the (regularized) covariance matrices of these clusters. The second step is repeated for each class so that the ideal vectors, covariance matrices and cluster memberships are known for each class's clusters.

In the **third step** the distances between clusters of different classes are calculated. For this purpose, let us denote the cluster indices by $o = 1$ to O, where O is the overall number of clusters for all classes combined. In addition, the set of cluster indices that belong to the same class as cluster o are denoted by C_o. The complement, meaning all clusters that belong to any class other than cluster o is denoted by $\bar{C}_o$. A simple example would be if the first cluster belongs to class 1, where class 1 contains the clusters $\{1,2\}$, class 2 encompasses cluster $\{3\}$ and class 3 comprises clusters $\{4,5\}$. In this example, C_1 refers

to the set containing {1,2}, and $\bar{C}_1$ encompasses the remaining clusters {3,4,5} belonging to the remaining classes 2 and 3. The notation for features are selected in a similar manner. The number of features is *D*, and a single feature can be denoted by *d*. The complement of *d* is denoted by $\bar{d}$ and contains all features with the exception of *d*, which are obviously the remaining *D*-1 features. The cluster centre of the *o*-th cluster is denoted by v_o, which is consistent with the notation used for the similarity classifier with multiple ideal vectors. The Mahalanobis distance between a cluster centre v_o and the cluster centre v_m of another class measured from the perspective of v_o and its covariance matrix is:

$$\mathrm{S}(v_o, v_m) = (v_o - v_m)^T \Sigma_o^{-1} (v_o - v_m) \tag{5.2}$$

This calculation is repeated for each combination of ideal vectors *o* and *m*, for which *o* and *m* belong to different classes. It is essential to highlight that all computations in this step include the entire set of *D* features and that both v_o and v_m are *D*-dimensional vectors.

The **fourth step** consists of the calculation of the distance between clusters without one of the features. The calculation is similar to the one conducted in the third step but does not incorporate all *D* features. In particular, for a feature *d* the separation between clusters of different classes without this feature is computed as:

$$\mathrm{S}\left(v_{o,\bar{d}}, v_{m,\bar{d}}\right) = \left(v_{o,\bar{d}} - v_{m,\bar{d}}\right)^T \Sigma_{o,\bar{d}}^{-1} \left(v_{o,\bar{d}} - v_{m,\bar{d}}\right), \tag{5.3}$$

where $v_{o,\bar{d}}$ and $v_{m,\bar{d}}$ denote the *o*-th and *m*-th cluster centres, respectively, with all features except for feature *d* – which simply means the cluster centres v_o and v_m without their *d*-th element. Hence, v_o and v_m are both (*D-1*)-dimensional vectors. The covariance matrix $\Sigma_{o,\bar{d}}$ does not include the *d*-feature, either. This means that it is the covariance matrix Σ_o without the *d*-th row and column and, hence, is of dimension (*D-1*)×(*D-1*). As in the third step, v_o and v_m belong to two different classes. This calculation is repeated for each feature *d* and each combination of ideal vectors *o* and *m*, for which *o* and *m* belong to different classes.

The **fifth step** is the calculation of the COLD score and the assignment of the corresponding feature rank. This step centres on the change in the distances between clusters of different classes with the entire feature set and the set of features without the *d*-th feature. The first sub-step is the calculation of the ratio of the Mahalanobis distance between two clusters without the *d*-th feature to the distance with the complete feature set. For cluster centres v_o and v_m, this ratio is computed as:

$$r_{o,m,d} = \frac{\mathrm{S}\left(v_{o,\bar{d}}, v_{m,\bar{d}}\right)}{\mathrm{S}(v_o, v_m)}, \tag{5.4}$$

where $r_{o,m,d}$ refers to the ratio of the Mahalanobis distances, with $r_{o,m,d} \geq 0$ since the Mahalanobis distance cannot be negative. There are three possible cases for values that $r_{o,m,d}$ can take, as follows:

$\boldsymbol{r_{o,m,d} > 1}$ **:** This case means that the distance between the cluster centres v_o and v_m without the *d*-th feature as denoted by $\mathrm{S}(v_{o,\bar{d}}, v_{m,\bar{d}})$ is larger than $\mathrm{S}(v_o, v_m)$, the distance with all features. This indicates that the *d*-th feature in the context of v_o and v_m is irrelevant and even noisy since it is easier to separate the clusters without this feature than including it.

$\boldsymbol{r_{o,m,d} = 1}$ **:** This case embodies no change in the distance between the two clusters with and without the *d*-th feature. Hence, the feature is irrelevant for the separation of these two clusters since the feature does not contribute to it (neither in a positive nor a negative way).

$\boldsymbol{r_{o,m,d} < 1}$ **:** This result represents the case when the distance $\mathrm{S}(v_{o,\bar{d}}, v_{m,\bar{d}})$ between v_o and v_m decreases compared to $\mathrm{S}(v_o, v_m)$ when the *d*-th feature is removed from the complete set of features. Therefore, the separation between the clusters deteriorates when the *d*-th feature is not present in the data. This suggests feature *d* is relevant for the separation of v_o and v_m since it supports the discrimination between the two clusters.

The value for $r_{o,m,d}$ indicates if the removal of the *d*-th feature contributes to the separation of two cluster centres v_o and v_m. But for each cluster centre v_o and a certain feature *d*, it is important to determine how the separation changes in regard to all clusters of other classes $\bar{C}_o$ since in the majority of cases there are multiple classes and/or multiple clusters of different classes. It is pivotal to measure the change in the distance between v_o with any cluster of another class. In addition, it is undoubtedly more important for the separation of the classes if a cluster that is closer to v_o changes its distance to v_o than the change of clusters comparably distant to v_o. Hence, when aggregating the ratio values $r_{o,m,d}$ for a given cluster centre v_o to all other clusters of different classes $\bar{C}_o$, these ratios are weighted to give closer clusters a higher weight than more distant ones.

$$w_{o,m} = \begin{cases} 1 & if\ card(\bar{C}_o) = 1 \\ 1 - \dfrac{\mathrm{S}(v_o, v_m)}{\sum_{m \epsilon \bar{C}_o} \mathrm{S}(v_o, v_m)} & otherwise\ , \end{cases} \tag{5.5}$$

where $w_{o,m} \epsilon\ [0,1]$ and equals one if there is solely a single cluster from another class, meaning that the cardinality of the set of clusters of other classes is 1. In a case where there are multiple other clusters from all other classes, then the weight corresponds to one minus the ratio of the distance of v_o to a cluster v_m divided by the sum of the distances of v_o to all other clusters from other classes. Therefore, the weight of closer clusters is higher than that of distant clusters. This means that changes in closer clusters have a higher impact on the evaluation of how much a feature contributes to the separation of the classes. Using the weights $w_{o,m}$ together with the corresponding ratios of the distances

$r_{o,m,d}$, the weighted change in the distances between v_o and all clusters of the other classes for the removal of the d-th feature can be computed.

$$r_{o,d} = \frac{\sum_{m \epsilon \bar{C}_o} w_{o,m} * r_{o,m,d}}{\sum_{m \epsilon \bar{C}_o} w_{o,m}} \tag{5.6}$$

In a case where there are more than two clusters from other classes, the sum of the weights is not equal to one. Thus, the denominator in Equation (5.6) is the sum of the weights. The meaning of $r_{o,d}$ is essentially the same as for $r_{o,m,d}$, with the sole exception that it accounts for a given cluster for all clusters of the remaining classes and not just a single one. Hence, $r_{o,d}$ values larger than 1 indicate an irrelevant and even noisy feature, values of exactly one indicate irrelevant features and $r_{o,d}$ values smaller than 1 embody features that are relevant for the separation of cluster v_o from the clusters of all other classes. Going one step further, to aggregate the information over all clusters for a certain feature, the COLD score can be computed as one minus the average $r_{o,d}$ value.

$$COLD_d = 1 - \frac{\sum_o r_{o,d}}{O} \tag{5.7}$$

The mean of the $r_{o,d}$ values represents the average (weighted) change in the cluster separation for a specific feature d. Changes smaller than 1 are more desirable than changes larger than 1 since they indicate an improvement in the average separation of the clusters. Threfore, the mean of $r_{o,d}$ is subtracted from 1. This converts the average change in the separation into an intuitive result. As a consequence, higher COLD scores represent more relevant features than low scores. The COLD score for any feature d is ≤ 1, with the following meaning for the corresponding feature:

$\boldsymbol{COLD_d > 0}$ **:** A relevant feature. Such a COLD score indicates a feature that contributes to the separation of the clusters of different classes.

$\boldsymbol{COLD_d = 0}$ **:** An irrelevant feature. The feature neither contributes to the separation of the clusters nor does it deteriorate the discrimination, which a noisy feature would do.

$\boldsymbol{COLD_d < 0}$ **:** An irrelevant and noisy feature. The feature does not contribute to the separation of the clusters and even on average deteriorates the distance between clusters of different classes.

Based on the COLD scores, the features can be ranked from the highest to the lowest score. Based on these scores, the user can decide how many of the highest-ranked features to keep. All features with a COLD score of less than or equal to zero can be discarded since they deteriorate the cluster separation for the classes or do not impact them. Concerning removing features in addition to those having negative or zero COLD scores, it can be noted that the impact of discarding features with comparably small positive COLD scores on the overall separation of classes will likely also be negligible.

5.3 Application to Artificial Data

To illustrate and demonstrate the ability of the COLD algorithm to rank features according to their contribution to the separation of the classes, four artificial examples were constructed that represent different data structures.

The **first artificial example** (Figure 5.2) is a binary classification task with three normally distributed features. The first two features embody a numerical XOR problem, where the two data groups of each class are not separable by one linear function but form distinct decision regions for each class. The third feature, as well as the first two features, is not correlated with any of the remaining features. Moreover, it overlaps moderately for both classes. In contrast, the first and second feature by itself overlaps entirely for both classes and is essentially irrelevant from the univariate perspective. Hence, the univariate evaluation will rank the third feature highest. Nonetheless, the best subset of two features should contain the first and second feature, which can together linearly separate the feature space.

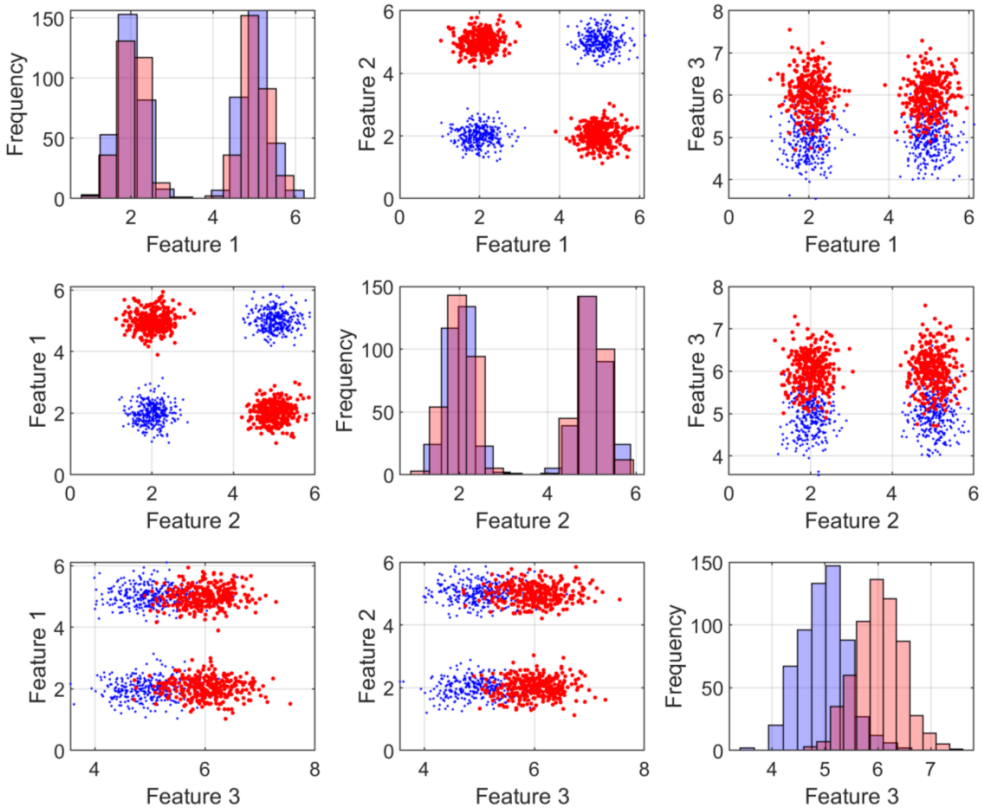


Figure 5.2: First artificial example of COLD [modified from Publication **IV**]

The **second artificial example** is a simple extension of the first example with an additional uncorrelated normally distributed feature that has identical mean values and variance for both classes. Thus, this additional feature is irrelevant by itself and also from a multivariate perspective since it does not contribute to the separation of the classes for any set of features.

The **third artificial example** consists of another three-dimensional classification problem with binary classes. All three features are normally distributed. In contrast to the previous two examples, each class consists of a single cluster. The cluster formed in the first and second dimension is elongated due to the high variance of the second feature and rotated due to the high negative covariance between these two features. The third feature is uncorrelated with the first two features and overlaps only to a small extent for both classes – making it the best univariate feature. In marked contrast to the third feature, the first and second feature overlap from a moderate to high extent and are each by themselves clearly less discriminant than the third feature. Nonetheless, the set of the first two features is capable of linearly separating the two classes. This is another example of features where the best single feature is not contained in the best subset of two features. Also, it illustrates that features that appear not or only slightly useful by themselves can be highly relevant for the separation of the classes if used together with another feature or a set of features. This example, as displayed in Figure 5.3, is related to the fourth example, presented by Guyon & Elisseeff (2003).

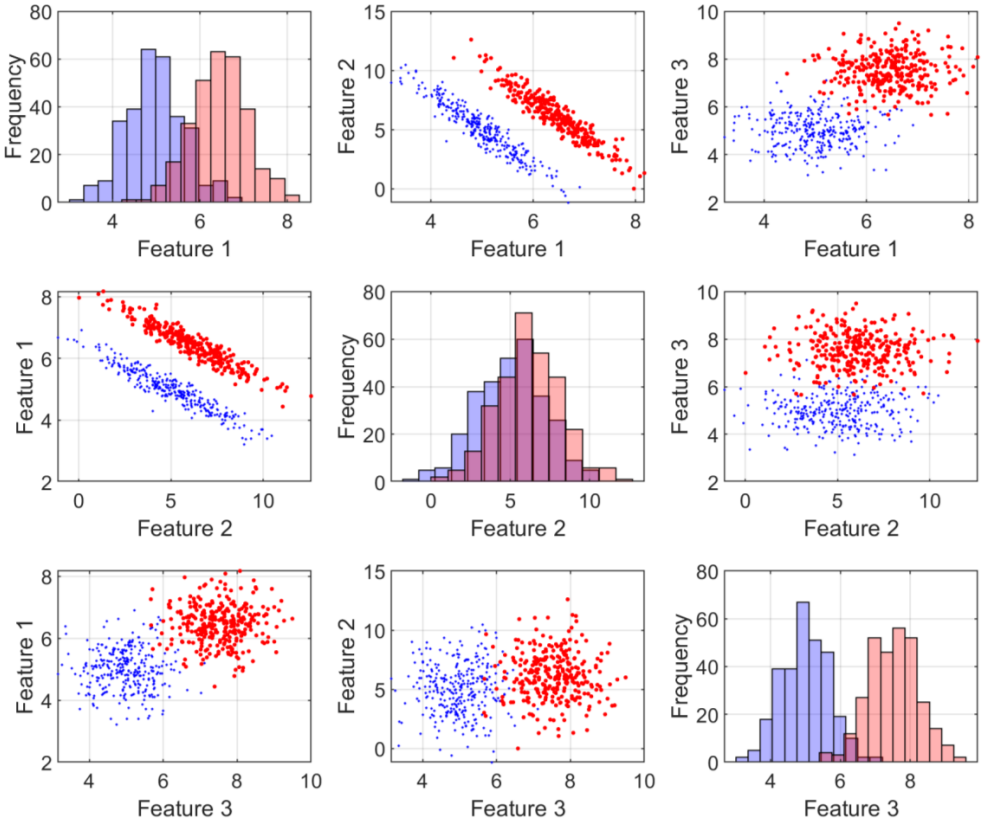
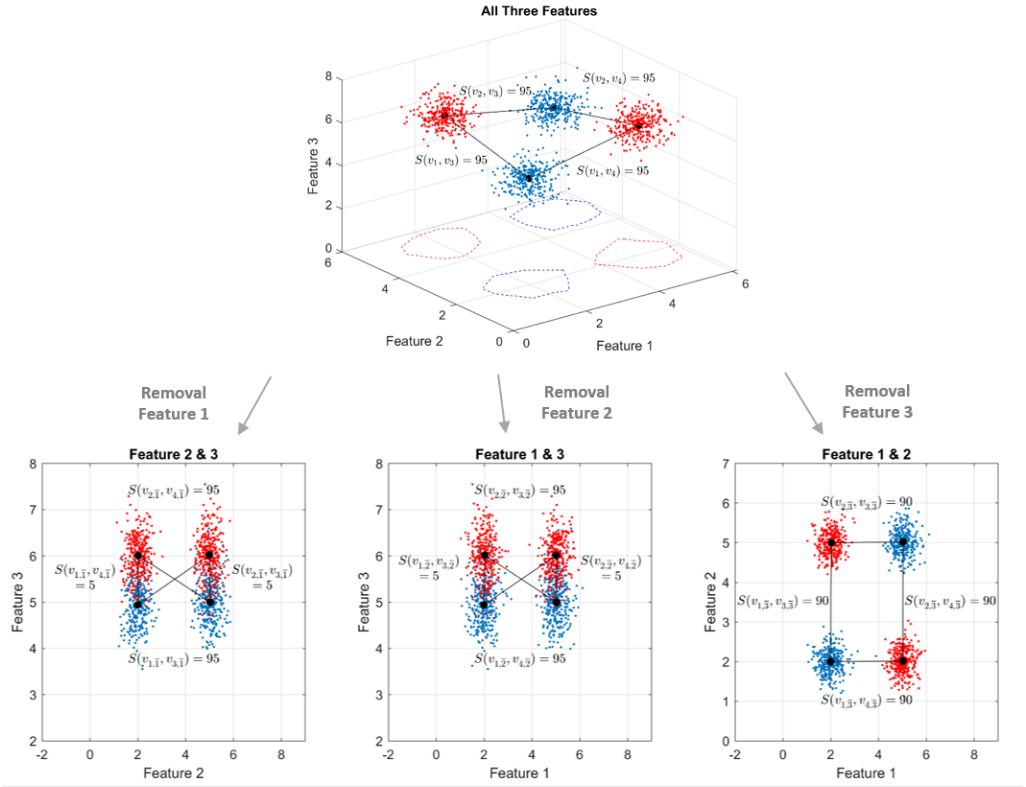


Figure 5.3: Third artificial example of COLD [modified from Publication **IV**]

The **fourth artificial example** is the same type of extension for the third example as example two was for the first example – a normally distributed feature with equal mean and variance was added that is uncorrelated with any of the remaining features. It is apparent that this feature is irrelevant for the separation of the classes.

The first and third artificial examples are both only three-dimensional, which simplifies the visualization of the classification problem they pose and the solution that COLD reaches. The logic behind COLD for the first artificial example is displayed in Figure 5.4, which shows the theoretical solution using COLD when the generated data points exactly resemble the specified mean values and covariance matrices (and the observations are not scaled to [0,1]). This representation was selected because of the simplicity of the calculation and values obtained. It is noteworthy that the covariance information is incorporated into COLD so that the scaling will not alter the results. The actual COLD scores for this example are very similar to the ones presented in this slightly simplified solution with COLD.



$r_{1,3,1} = r_{3,1,1} = \frac{S(v_{1,1}, v_{3,\bar{1}})}{S(v_1, v_3)} = 1$	$r_{1,3,2} = r_{3,1,2} = \frac{S(v_{1,2}, v_{3,\bar{2}})}{S(v_1, v_3)} \approx 0.05$	$r_{1,3,3} = r_{3,1,3} = \frac{S(v_{1,\bar{3}}, v_{3,\bar{3}})}{S(v_1, v_3)} \approx 0.95$
$r_{1,4,1} = r_{4,1,1} = \frac{S(v_{1,1}, v_{4,\bar{1}})}{S(v_1, v_4)} \approx 0.05$	$r_{1,4,2} = r_{4,1,2} = \frac{S(v_{1,2}, v_{4,\bar{2}})}{S(v_1, v_4)} = 1$	$r_{1,4,3} = r_{4,1,3} = \frac{S(v_{1,\bar{3}}, v_{4,\bar{3}})}{S(v_1, v_4)} \approx 0.95$
$r_{2,3,1} = r_{3,2,1} = \frac{S(v_{2,1}, v_{3,\bar{1}})}{S(v_2, v_3)} \approx 0.05$	$r_{2,3,2} = r_{3,2,2} = \frac{S(v_{2,2}, v_{3,\bar{2}})}{S(v_2, v_3)} = 1$	$r_{2,3,3} = r_{3,2,3} = \frac{S(v_{2,\bar{3}}, v_{3,\bar{3}})}{S(v_2, v_3)} \approx 0.95$
$r_{2,4,1} = r_{4,2,1} = \frac{S(v_{2,1}, v_{4,\bar{1}})}{S(v_2, v_4)} = 1$	$r_{2,4,2} = r_{4,2,2} = \frac{S(v_{2,2}, v_{4,\bar{2}})}{S(v_2, v_4)} \approx 0.05$	$r_{2,4,3} = r_{4,2,3} = \frac{S(v_{2,\bar{3}}, v_{4,\bar{3}})}{S(v_2, v_4)} \approx 0.95$
$r_{1,1} = \frac{1 * 0.5 + 0.05 * 0.5}{0.5 + 0.5} = 0.525$	$r_{1,2} = \frac{0.05 * 0.5 + 1 * 0.5}{0.5 + 0.5} = 0.525$	$r_{1,3} = \frac{0.95 * 0.5 + 0.95 * 0.5}{0.5 + 0.5} = 0.95$
$r_{2,1} = \frac{0.05 * 0.5 + 1 * 0.5}{0.5 + 0.5} = 0.525$	$r_{2,2} = \frac{1 * 0.5 + 0.05 * 0.5}{0.5 + 0.5} = 0.525$	$r_{2,3} = \frac{0.95 * 0.5 + 0.95 * 0.5}{0.5 + 0.5} = 0.95$
$r_{3,1} = \frac{1 * 0.5 + 0.05 * 0.5}{0.5 + 0.5} = 0.525$	$r_{3,2} = \frac{0.05 * 0.5 + 1 * 0.5}{0.5 + 0.5} = 0.525$	$r_{3,3} = \frac{0.95 * 0.5 + 0.95 * 0.5}{0.5 + 0.5} = 0.95$
$r_{4,1} = \frac{0.05 * 0.5 + 1 * 0.5}{0.5 + 0.5} = 0.525$	$r_{4,2} = \frac{1 * 0.5 + 0.05 * 0.5}{0.5 + 0.5} = 0.525$	$r_{4,3} = \frac{0.95 * 0.5 + 0.95 * 0.5}{0.5 + 0.5} = 0.95$
$COLD_1 = 1 - 0.525 = 0.475$	$COLD_2 = 1 - 0.525 = 0.475$	$COLD_3 = 1 - 0.95 = 0.05$

Figure 5.4: Theoretical COLD scores for the first artificial example

The figure illustrates that in the three-dimensional space, the clusters of different classes (in blue and red) are all 95 units away from each other. It is apparent that the first and second feature contribute most to the separation of the classes. However, it is also obvious that the third feature contributes slightly to the separation since the blue clusters are situated slightly lower than the red ones. The removal of the first or second feature always moves two of the clusters of each class closer together in the two-dimensional representation. Consequently, these clusters start to overlap and reduce the Mahalanobis distance between them to 5. The second pair of clusters (for instance v_1 and v_3 for the first example) remain distant to each other solely on account of the one remaining feature of the first two features. Hence, for the removal of the first or second feature, the average distance between clusters of different classes is reduced to 52.5% of the original distance with all three features. In contrast, the removal of the third dimension impacts all cluster distances by an equally small magnitude. The weighted average ratio remains with 0.95 (meaning 95%) largely unchanged. This shows that the third feature did contribute to the separation of the clusters of different classes but only to a small extent compared to the first two features. Overall, the COLD score of 0.475 for each of the first two features and only 0.05 for the third feature represents this difference. The corresponding feature ranking places the first and second feature on the first two ranks (with equal COLD scores the order can be arbitrary) and the third feature last.

The feature ranking for this example of COLD was compared with the ReliefF algorithm, the feature selection by Luukka (2011), the FSAE, the Laplacian score and the Fisher score. In addition to the scaling or normalization conducted within each filter method, there is no initial scaling or normalization applied to the artificial examples.[8] The results of the comparison are summarized in Table 5.1.

[8] The ReliefF setups, the feature selection of Luukka (2011), FSAE and COLD all explicitly incorporate some form of scaling or normalization within their algorithms. This is not the case for the Laplacian score and the Fisher score, which in their algorithms do not explicitly include a scaling or normalization (e.g. into the unit interval). However, both algorithms incorporate some form of measuring distances/similarities and divide these values by some form of variance (which is to a certain extent similar to a z-score). Notwithstanding, the ranking for the Laplacian score for these four examples will change (with a tendency to emulate the univariate results of, e.g., the Fisher score) if a scaling into the unit interval is initially conducted.

Filter Method	Feature 1	Feature 2	Feature 3
Actual Structure	Ranks 1 & 2		Rank 3
ReliefF (10 nearest hits/misses)	Rank 1	Rank 2	Rank 3
ReliefF (70 nearest hits/misses, sigma = 20)	Rank 1	Rank 2	Rank 3
FS Luukka (2011)	Rank 3	Rank 1	Rank 2
FSAE (De Luca & Termini entropy, $l = 1$)	Rank 3	Rank 1	Rank 2
COLD	Rank 1	Rank 2	Rank 3
Laplacian Score	Rank 2	Rank 1	Rank 3
Fisher Score	Rank 3	Rank 1	Rank 2

Table 5.1: First artificial example feature rankings [modified from Publication **IV**][9]

It is apparent that the univariate filter methods that evaluate each feature separately rank the third univariate best feature first and fail to detect that the first and second feature can jointly separate the feature space without misclassification. Opposed to that, both versions of ReliefF and the COLD algorithm as well as the Laplacian score correctly rank these two features first. The difference that these rankings lead to is apparent when the feature subsets of the two highest-ranked features are selected, and the mean accuracy for the classification task is compared. The mean classification accuracies for this purpose are the mean values of the accuracies achieved by a K-nearest neighbour classifier ($k = 10$), a decision tree (minimum leaf size = 10) and (one-versus-all) support vector machines (with radial basis function). All multivariate feature selection methods, including COLD, were able to maintain or improve their mean accuracies to about 100% due to the removal of the third-ranked feature. At the same time, the feature subsets of the univariate methods lead to a decrease in the classification accuracy to on average about 87.9%, which is significantly ($p < 0.01$) smaller. It is unsurprising that this situation would reverse in the case where two features are discarded, and only a single feature is retained. In this case, the feature subset of the univariate methods outperforms the subset of the multivariate methods by a significant margin. The reason is that a univariate algorithm aims to find the single best feature(s), whereas the multivariate algorithm determines the feature that is jointly with one or more features most relevant. This obviously does not have to be the same in the case of dependencies between features. For this artificial example, it also highlights that the set of the best two features does not contain the single best feature.

[9] The rankings for the feature selection method by Luukka (2011) and the FSAE differ from those presented in Publication **IV** since in the original source no scaling to the unit interval was conducted, which is necessary for both algorithms to calculate the similarities correctly. This is true for all four artificial examples so that the results for these two benchmark algorithms differ in most cases to those reported in the original publication. It should be noted that this alters neither the results of the remaining feature selection methods, including COLD, nor the conclusions made in Publication **IV** concerning the COLD algorithm. In particular, it does not alter the conclusion that COLD is the only algorithm that consistently ranks features according to the best feature subset. Moreover, this error only occurred in the artificial datasets. For the real-world datasets, the scaling into the unit interval was incorporated in the ranking conducted by the feature selection of Luukka (2011) and the FSAE.

The rankings for all filter methods for the second artificial example are displayed in Table 5.2.

Filter Method	Feature 1	Feature 2	Feature 3	Feature 4
Actual Structure	Rank 1 & 2		Rank 3	Rank 4
ReliefF (10 nearest hits/misses)	Rank 2	Rank 1	Rank 3	Rank 4
ReliefF (70 nearest hits/misses, sigma = 20)	Rank 2	Rank 1	Rank 3	Rank 4
FS Luukka (2011)	Rank 4	Rank 3	Rank 2	Rank 1
FSAE (De Luca & Termini entropy, $l = 1$)	Rank 3	Rank 4	Rank 2	Rank 1
COLD	Rank 1	Rank 2	Rank 3	Rank 4
Laplacian Score	Rank 1	Rank 2	Rank 3	Rank 4
Fisher Score	Rank 3	Rank 2	Rank 4	Rank 1

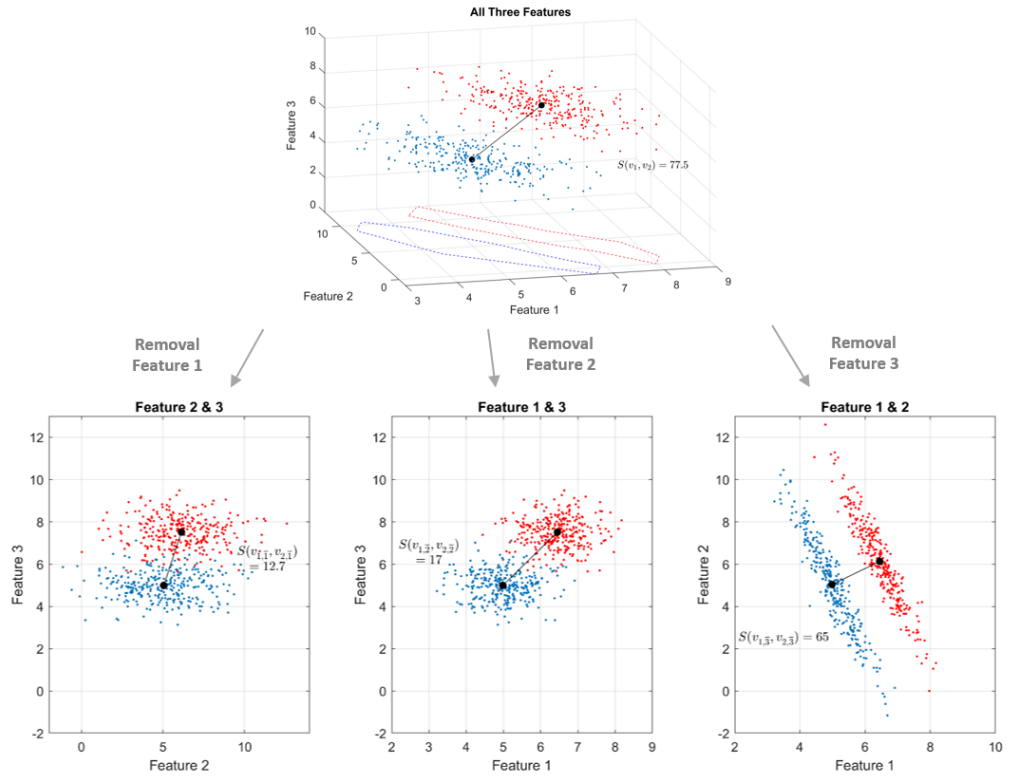
Table 5.2: Second artificial example feature rankings [modified from Publication **IV**]

For the second artificial example, the same approaches as those used in the first example are able to attain the desired correct ranking. Also, those approaches that incorrectly ranked features previously do not improve their ordering. It is noteworthy that the approach by Luukka (2011) ranks the new irrelevant normally distributed feature first, whereas the FSAE and the Fisher score do not rank it last.[10] One could argue that from the univariate view, ranking the third feature first and the remaining features in any order is plausible given that the third feature is univariately the single best feature and that, from a one-dimensional perspective, all remaining features are similarly irrelevant. However, it is apparent that the fourth feature should even according to a univariate approach not be ranked first. Without going into detail, it can be stated that for the filter method by Luukka (2011), this result is obtained due to the fact that the feature is normally distributed and completely overlapping. Because of this, many observations will show a similarity of close to one to their own class's representative as well as the representative of the other class. Unfortunately, this corresponds to a low entropy value, which indicates a relevant feature in this feature selection method. Finally, for an irrelevant uniformly distributed feature, such a behaviour would not be observed.

Before discussing the actual results for the third example, the conceptual solution with COLD for this example is illustrated in Figure 5.5 and highlights how COLD determines the multivariate relevance of the first and second feature. This solution assumes that the generated samples perfectly resemble the specified means and covariance matrices, that

[10] As mentioned previously, for this task the lack of scaling for the feature selection of Luukka (2011) and the FSAE also changed the ranking. From a univariate standpoint, the fourth, irrelevant feature does not necessarily have to be ranked last since the first and second feature are from that perspective also irrelevant. In general, this makes the ranking (of all features with the exception of the third one) dependent on minor variations in the observations. The statement in Publication **IV** that all filter methods ranked the additional irrelevant feature last was inaccurate.

no scaling is conducted and that the clustering algorithm in COLD finds exactly these two cluster centres.



$r_{1,2,1} = r_{2,1,1} = \frac{S(v_{1,1}, v_{2,\bar{1}})}{S(v_1, v_2)} \approx 0.164$	$r_{1,2,2} = r_{2,1,2} = \frac{S(v_{1,2}, v_{2,\bar{2}})}{S(v_1, v_2)} \approx 0.219$	$r_{1,2,3} = r_{2,1,3} = \frac{S(v_{1,\bar{3}}, v_{2,\bar{3}})}{S(v_1, v_2)} \approx 0.839$
$r_{1,1} = 0.164$	$r_{1,2} = 0.219$	$r_{1,3} = 0.839$
$COLD_1 = 1 - 0.164 = 0.836$	$COLD_2 = 1 - 0.219 = 0.781$	$COLD_3 = 1 - 0.839 = 0.161$

Figure 5.5: Theoretical COLD scores for the third artificial example

The contour lines for the first and second feature indicate that these two features are capable of linearly separating the two elongated clusters. If either the first or second feature are removed, the classes start to overlap to a small extent. In addition, the Mahalanobis distance between the clusters decreases from 77.5 with all features to 12.7 and 17, respectively. This demonstrates that the first and second feature are both strongly contributing to the class separation. Their high COLD scores of 0.836 and 0.781, respectively, reflect this. The third feature, which is univariately the best feature,

contributes less to the separation of the clusters. Hence, the distance between the clusters is only reduced from 77.5 to 65. This deterioration is considerably less severe than for the first two features. Moreover, the two classes remain linearly separable even after the removal of the third feature. The COLD score reflects the small but existing contribution of the third feature with a comparably low positive COLD score of 0.161. The corresponding feature ranking emphasizes that the first and second feature contribute more to the cluster separation, and, hence, they are ranked first and second. The best univariate feature only ranks third. Once more, the best subset of two features does not contain the univariately best feature.

The rankings for the actual third artificial example with one correlated elongated cluster for each class are presented in Table 5.3.

Filter Method	**Feature 1**	**Feature 2**	**Feature 3**
Actual Structure	Rank 1 & 2		Rank 3
ReliefF (10 nearest hits/misses)	Rank 1	Rank 3	Rank 2
ReliefF (70 nearest hits/misses, sigma = 20)	Rank 1	Rank 3	Rank 2
FS Luukka (2011)	Rank 2	Rank 1	Rank 3
FSAE (De Luca & Termini entropy, $l = 1$)	Rank 3	Rank 1	Rank 2
COLD	Rank 1	Rank 2	Rank 3
Laplacian Score	Rank 1	Rank 3	Rank 2
Fisher Score	Rank 3	Rank 1	Rank 2

Table 5.3: Third artificial example feature rankings [modified from Publication **IV**]

It is apparent that for this example, the ranking according to the best subset of two features is only accomplished by COLD and the approach by Luukka (2011). However, since the approach by Luukka (2011) is a univariate approach, it is obvious that the correct ranking is not due to its ability to account for the complementarity of the first and second feature. Rather, the algorithm has determined that the first and second feature are univariately the best two features. This is clearly not representative of their univariate ability to discriminate the data. Both features have almost identical means and overlap essentially completely. Once again, the overlapping features are univariately ranked high by this approach since most similarities of observations with any ideal vector are close to 1. This leads to low entropy values, falsely indicating a relevant feature. Hence, COLD is effectively the only algorithm that correctly detects and ranks the complementarity[11] of the first and second feature. However, it does not score the features exactly as theoretically expected. The K-medoids algorithm divides these strongly elongated clusters into two less strongly elongated clusters. Still, the corresponding ranking remains

[11] The term ‘complementarity’ is used in Guyon and Elisseeff (2003) to indicate a ‘perfect separation’ that can be accomplished by a set of (two) variables.

unchanged compared to its theoretical counterpart, and also the third feature is clearly ranked last, with a COLD score that is considerably lower than for either of the first two features. Regardless, due to the subdivision of clusters, the first feature received a considerably higher score than the second. This stresses the reliance of COLD on the clustering partition even though this example illustrated that the ranking can be unaltered, especially for such clear differences in the contribution of features to the separation of classes.

The multivariate filter methods, such as ReliefF and the Laplacian score, which do not rely on a distance or similarity measure that incorporates a mixture of locality and covariance information, fail to end up with the correct feature ranking. As a consequence, only COLD and the filter by Luukka (2011) and their two-feature subsets obtain a mean accuracy of close to 100%. In contrast, the feature subsets suggested by the remaining filter methods deteriorate the performance to about 96%, which is significantly smaller than with COLD (p-value < 0.01). This decrease in performance is due to the small overlap that occurs if either the first or second feature is removed.

Finally, the rankings of all filter methods for the fourth and last artificial example are displayed in Table 5.4.

Filter Method	Feature 1	Feature 2	Feature 3	Feature 4
Actual Structure	Rank 1 & 2		Rank 3	Rank 4
ReliefF (10 nearest hits/misses)	Rank 1	Rank 3	Rank 2	Rank 4
ReliefF (70 nearest hits/misses, sigma = 20)	Rank 1	Rank 3	Rank 2	Rank 4
FS Luukka (2011)	Rank 4	Rank 2	Rank 1	Rank 3
FSAE (De Luca & Termini entropy, l = 1)	Rank 3	Rank 4	Rank 2	Rank 1
COLD	Rank 1	Rank 2	Rank 3	Rank 4
Laplacian Score	Rank 1	Rank 2	Rank 3	Rank 4
Fisher Score	Rank 3	Rank 1	Rank 2	Rank 4

Table 5.4: Fourth artificial example feature rankings [modified from Publication **IV**]

Once again, COLD successfully ranks the features according to their contribution to the separation of the classes. For this example, the Laplacian score is also able to rank the features correctly. The multivariate ReliefF and the univariate Fisher score do not rank all features correctly but recognize the fourth irrelevant feature as the least relevant one. FSAE ranks the third univariately best feature first but ranks the fourth irrelevant normally distributed feature second. The least desirable result is obtained for the univariate feature selection method by Luukka (2011), where the irrelevant feature is ranked first and the univariately best feature last. This problem once more originates in the fact that the samples of the irrelevant fourth feature are normally distributed, so most

observations of either class will have high similarities with their own and the competing class's ideal vector, which is an indication of a relevant feature.

Overall, only COLD demonstrated the consistent ability to rank features according to their contribution to the separation of the classes. As a last point, it should be noted that the rankings obtained by all filter methods remain unchanged for 500 iterations for each of the artificial examples.

5.4 Application to Medical Data

The COLD algorithm demonstrated for all four artificial examples the ability to rank each feature's relevance according to its contribution to the separation of the classes. To additionally show its ability to conduct feature selection successfully in a real-world context, it will here be applied to two medical real-world datasets from the UCI Machine Learning Repository (Lichman, 2013). The first dataset is the Dermatology Data Set (Ilter and Guvenir, 1998), which was mentioned and used in Section 3.5 as well. It has 358 observations of 34 features and constitutes a 6-class problem. The second medical dataset is the Arrhythmia dataset, which differentiates the presence and absence of cardiac arrhythmia (Guvenir, Acar and Muderrisoglu, 1998). The dataset consists of 420 observations of 258 non-constant features.

The performance of COLD and that of the competing filter methods was evaluated by the following three classifiers: a K-nearest neighbour classifier (with 10 neighbours), a decision tree (with minimum leaf size of 10) and (one-versus-all) support vector machines (with radial basis function). For the Dermatology dataset, the mean accuracies (over 500 runs) for all filter methods and feature subset sizes are displayed in Figure 5.6.

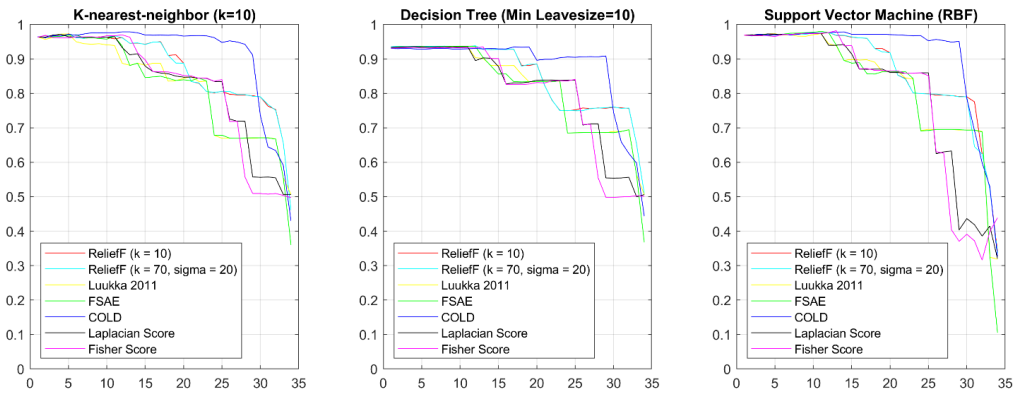


Figure 5.6: Comparison of filter feature selection on the Dermatology dataset

At first glance, it is apparent that COLD outperforms the remaining feature selection algorithms for a wide range of feature subsets. Besides that, COLD achieved for the K-nearest neighbour (KNN) classifier the highest mean accuracy of 97.9%. Using the one-sided Welch's test with unequal variances, it can be highlighted that for the KNN classifier, the outperformance of the feature subset of COLD compared to that of all other filter methods in this study is highly significant (p-value < 0.01) for 7 to 29 feature removals. For the decision tree, the highest performance of 93.83% is accomplished with ReliefF (k = 70, *sigma* = 20) for 12 removed features. COLD achieved with 17 removed features a competitive result of 93.48%. Moreover, COLD eventuates in a mean accuracy of over 90% until the removal of 29 out of 34 features. At this point, the strongest competitors, the two ReliefF setups, are already at a mean classification accuracy of close to 75%. Unsurprisingly, the outperformance of COLD's mean accuracy between 17 to 29 feature removals is highly significant (p-value < 0.01) compared to the remaining filter methods. A similar picture to that from the KNN classifier is obtained for the support vector machines. The highest performance of 98.19% is accomplished with the Fisher score and 13 removed features. However, the highest classification accuracy with COLD of 97.86% is not far off, and the subsequent mean accuracies with COLD remain at a level of over 90% accuracy for a much longer time. The mean accuracy with COLD for 14 to 29 features is highly significantly larger than that of the Fisher score and all other filter approaches. A remarkable result for COLD's ability to select relevant feature subsets can be seen for the removal of 29 out of 34 features. The classification accuracy related to this subset of five features still exceeds 95%, whereas the runner-up ReliefF is already below 80% and the Fisher score even below an accuracy of 40%.

For the Arrhythmia dataset, the mean accuracies for the three classifiers are illustrated in Figure 5.7.

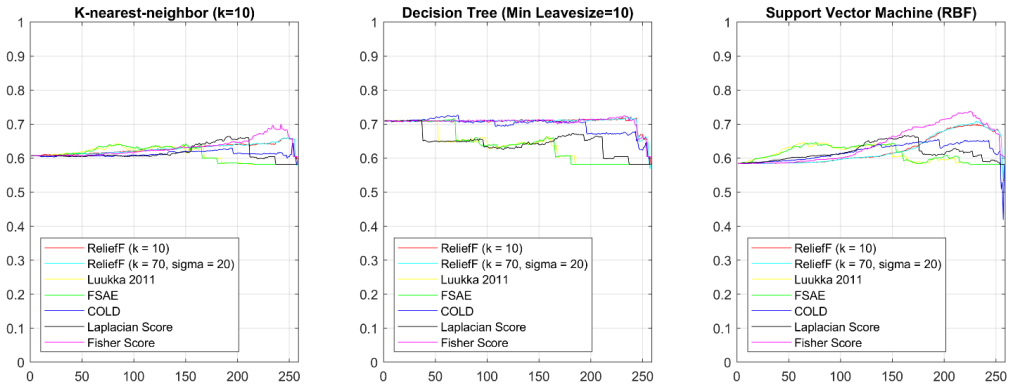


Figure 5.7: Comparison of filter feature selection on the Arrhythmia dataset

For this dataset, the KNN classifier demonstrates for all filter methods and a wide range of feature removals a performance between 61% and 65%. For up to 200 feature removals, the mean classification accuracies of all filter methods appear very similar in magnitude. Subsequently, the Fisher score clearly performs best and reaches an accuracy of 70.1% with 242 features removed. This is the highest mean accuracy observed for the KNN classifier on this dataset. All feature selection algorithms successfully conduct feature selection since they are all capable of discarding (a large share of) features and reach a comparable or even improved classification accuracy compared to using the complete set of features. It is noteworthy that for KNN, the single best feature according to all algorithms still reaches an accuracy between 58.2% and 60.4%. This is only slightly less than the mean accuracy of about 60.7% for all 258 features. Moreover, the highest performance, of 60.4%, for a single feature is reached with the Fisher score. This univariate approach outperforms the remaining approaches, even COLD and ReliefF, on this dataset. This indicates that it may be unnecessary to use a multivariate approach for this classification task. Moreover, it shows that a univariate evaluation of the features leads to even better results on this dataset. The comparison of the filter methods for the decision tree is similar to that of KNN with the main exception that the best mean accuracy of 72.66% is accomplished with COLD and 71 removed features. Even though COLD remains competitive compared to the other filter methods for the remaining feature removals, it is also for the decision tree apparent that the univariate Fisher score overall performs at least as good as the multivariate approaches. It reaches a performance of about 72.5% for 232 discarded features and, eventually, also selects the single best feature. The same outcome for the Fisher score is observed using support vector machines. The Fisher score selects the best feature subset with 225 discarded features, resulting in a mean accuracy of 73.83%. In addition, once more this algorithm selects the single best performing feature. For the second medical dataset, Arrhythmia, the univariate Fisher score performs overall very well and appears to be the most suitable algorithm for this dataset. This indicates that a multivariate method is not necessary on this dataset. There might not be any dependencies among the features that are adding any contribution to the separation of the classes. Besides that, it is noteworthy that COLD reaches competitive results for this dataset and that for the decision tree classifier it selects the best-performing feature subset. COLD is capable of improving the mean accuracy on the dataset by a few percentage points for about 200 discarded features and achieves a performance similar to that with the entire feature subset with less than 8 out of 258 features.

5.5 Conclusion and Limitations of the COLD Algorithm

This section discussed a novel multivariate supervised filter method named COLD. The algorithm ranks features according to their contribution to the separation of the groups/clusters of different classes. It deploys class-wise clustering of the observations to initially find the groups that each class is composed of and determine their characteristics represented in terms of their cluster centres and covariance matrices. Only linearly independent features are considered for the clustering since linearly dependent features can be expressed in terms of one or more other features, which renders them redundant.

After the clustering, the relevance of each feature is determined by measuring their contribution to the separation of the clusters belonging to different classes. In this context, contribution is interpreted as the change in the Mahalanobis distance between the clusters of different classes when a feature is removed from the complete set of features. Hence, the contribution of a feature is not restricted to its univariate impact on the cluster separation but on how it contributes to the separation together with all other features. Thus COLD accounts for the dependencies and complementarity of features (multivariate).

COLD and several other filter methods were tested on four artificial examples, which were constructed specifically in a way such that the set of the two best features does not contain the univariate best feature. This was accomplished by either designing a numerical XOR-like problem with two distinct decision regions for each class or by generating elongated, correlated clusters that overlap along the first two dimensions. For all four examples, only COLD consistently ranked the features according to the highest-performing feature subset by accounting for the joint relevance of the two univariate less relevant or even irrelevant features. It is apparent that the difference in the mean accuracy for those approaches that did not detect this complementarity of the first two features is highly significantly smaller than using COLD. Additionally, COLD also correctly acknowledged the additional overlapping feature as irrelevant and ranked it last.

For the two medical real-world datasets, the COLD algorithm demonstrated at least competitive results compared to the remaining six filter methods in this study. For both datasets, COLD demonstrated the ability to remove irrelevant variables and improve the classification accuracy for the three classifiers deployed for the comparison – the KNN classifier, a decision tree and support vector machines. For the Dermatology dataset, the feature ranking suggested by COLD outperformed the ranking of the remaining filter methods for a large range of features by a highly significant margin.

One limitation of the COLD algorithm is that it is computationally expensive in comparison to the other filter methods in this study. It is apparent that this limitation originates from the fact that it evaluates features in a multivariate way. Moreover, it incorporates the K-medoids algorithm for clustering, which functions well for categorical and noisy data but is more complex than several other clustering algorithms, such as its numerical counterpart K-means. Another limitation is that the algorithm's ranking depends on the clustering result and the covariance matrices determined during the clustering. If the clusters are close to each other and have a difficult data structure/shape (e.g. non-convex), or very few observations make up a cluster so that the covariance matrix is based on only a few observations, COLD may face difficulties in ranking all features according to their contribution to the class separation. This behaviour as well as COLD's ability to cope with a large number of irrelevant features in artificial and additional real-world cases has to be investigated in future research. Finally, a univariate but less complex version of COLD as well as an unsupervised setup for COLD could be developed in prospective publications and research studies.

6 Conclusion, Limitations and Future Work

The focus of this dissertation is on heuristic filter methods, in particular those using distance and information evaluation criteria. For this purpose, initially the univariate information-based filter method by Luukka (2011) was discussed. On one hand, its ability to improve the classification accuracy on selected datasets while using a subset of the original features was highlighted. On the other hand, an emphasis was placed on its vulnerabilities with respect to the use of a single ideal vector per class and the combination of similarity and entropy for certain data structures. The representation of a class by a single ideal vector is not suitable for more complex data structures in a class. That said, it is an intentional simplification, which is part of a trade-off with respect to the computational time and complexity of the algorithm. However, the vulnerability of this algorithm to highly overlapping features does not reflect any trade-off. It was clearly demonstrated and explained how measuring the similarity of observations to the ideal vector of the same and competing class and using the similarity values for the entropy calculation can result in a completely overlapping feature being preferred to a feature that separates classes well and has small to moderate variance. On account of this vulnerability, the 'Fuzzy Similarity and Entropy' (FSAE) feature selection method was introduced. It is a univariate, information-based filter premised on the approach by Luukka (2011). The FSAE incorporates a class- and feature-specific scaling factor that accounts for the distance between the ideal vectors of the classes within each feature. For the three artificial examples on which the vulnerability of the feature selection by Luukka (2011) was highlighted, the FSAE showed intuitive feature scores and feature removal decisions. The FSAE was compared to the feature selection by Luukka (2011) in regard to its filter and wrapper form as well as to several distance- and information-based heuristic filter methods (univariate and multivariate) on five real-world medical datasets. In both feature selection types, FSAE accomplished at least comparable classification results, often with fewer features than the approach by Luukka (2011). Moreover, it achieved competitive results compared to the remaining filter methods.

Subsequently, the FSAE was implemented together with different classification algorithms on a custom financial dataset constructed by the author of this dissertation for the prediction of the S&P500 intraday return. Using FSAE and a random forest, a four-class classification model was successfully developed. On top of that, several simple trading strategies based on different predictions of this classification model were tested. One important finding showed that the best feature selection- and classification-based trading strategy outperformed a passive buy-and-hold strategy after small to moderate transaction costs. Another contribution in this context was highlighted in the analysis of the actual returns associated with the predicted return classes. For this dataset, the average return of the 'strong positive' and 'strong negative' classes was higher than that of the two classes for 'slightly positive' and 'slightly negative' returns even when misclassifications were included.

All these results indicate that the FSAE is a clear improvement over the approach by Luukka (2011), that it leads to intuitive results, that it demonstrated the ability to successfully conduct feature selection and that it is a fast filter method due to its univariate evaluation. The univariate evaluation of features and the single ideal vector per class are two trade-offs made to keep the computational complexity low. Notwithstanding, these two aspects as well as leaving out covariance information can be regarded as limitations of this filter method with respect to determining the optimal feature subset.

Before addressing the limitation of using a single ideal vector, leaving out covariance information and the univariate evaluation of features in the context of feature selection, a framework to determine multiple ideal vectors per class was discussed in the context of the similarity classifier. Using a single ideal vector to represent a class is not suitable for classes with multiple distinct decision regions, neither with respect to classification nor for feature selection. Hence, in the context of classification, the novel similarity classifier with multiple ideal vectors was proposed. Using K-means clustering together with the Jump method ensured that a suitable number of clusters for each class as well as each corresponding centroid per cluster can be determined. These centroids can function as the ideal vectors, meaning the cluster representatives, for each class. The similarity classifier using these multiple ideal vectors demonstrated on artificial examples with multiple distinct decision regions per class the consistent ability to acknowledge and represent these regions. Moreover, it made use of the corresponding centre points to classify observations with a highly significant outperformance compared to the original classifier. On the real-world financial datasets, the similarity classifier with multiple ideal vectors ($Y = 1$) was the similarity-based classifier with the highest mean accuracy in all examples. Nonetheless, it is noteworthy that the similarity classifier with a single ideal vector also performed well, especially on the Australian Credit dataset. This indicates that allowing multiple decision regions per class can improve the classification results in some cases, but it is not always necessary or beneficial for classification accuracy.

After demonstrating that deploying clustering to find and characterize multiple distinct decision regions can benefit the classification of observations, this concept was also deployed in the context of feature selection. In addition to allowing multiple ideal vectors per class, covariance information for each cluster was included to incorporate differences in the variance of different features and to account for correlation. The suggested distance-based COLD filter algorithm emphasized the contribution of a feature to the set of features with respect to the separation of the clusters of different classes. Following this idea, the evaluation is multivariate since the dependencies among multiple features can be accounted for. In a comparison of COLD with several univariate and multivariate distance- and information-based filter methods for several artificial examples, only COLD demonstrated the consistent ability to rank the features according to their joint relevance. The results indicate that even in a setting where the subset of the best two features does not contain the single best feature, the remaining multivariate algorithms were not capable of ranking the features accordingly. On two real-world medical datasets, COLD demonstrated at least competitive results compared to the benchmark filter methods. Moreover, in one case it clearly outperformed all competing filter methods for

a large range of feature removals. Overall, this suggests that COLD's premise of measuring the contribution of a feature to the set of features can account for conditional dependencies among features that other algorithms cannot capture.

The results with COLD and the similarity classifier with multiple ideal vectors indicate that it can be beneficial to apply clustering and, subsequently, combining the information on the discovered decision regions for feature selection and classification. It is apparent that the drawback of this approach is its computational complexity. Hence, future work will focus on finding more efficient ways to determine (approximations) of the distinct decision regions and their representatives. Currently, approaches such as FSAE and COLD are very different from each other given that the former is univariate, more simplistic and fast, whereas the latter is multivariate and more computationally expensive. Hence, it seems intuitive to aim at finding either extensions of the FSAE to allow a competitive multivariate evaluation of features or improve COLD to achieve stable, clear clustering partitions in a more efficient way. Both enhancements can certainly contribute to a more efficient and effective feature selection overall.

References

Almuallim, H. and Dietterich, T. (1994) 'Learning Boolean concepts in the presence of many irrelevant features', *Artificial Intelligence*, 69(1–2), pp. 279–305. doi: 10.1016/0004-3702(94)90084-1.

Almuallim, H. and Dietterich, T. G. (1991) 'Learning With Many Irrelevant Features', *In Proceedings of the Ninth National Conference on Artificial Intelligence*, 91, pp. 547–552. doi: 10.1.1.48.2488.

Altay, E. and Satman, M. H. (2005) 'Stock market forecasting: artificial neural network linear regression comparison in an emerging market', *Journal of Financial Management & Analysis*, 18(2), pp. 18–33.

Ang, J. C. *et al.* (2016) 'Supervised, unsupervised, and semi-supervised feature selection: A review on gene selection', *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 13(5). doi: 10.1109/TCBB.2015.2478454.

Antal, B. and Hajdu, A. (2014) *Diabetic retinopathy debrecen data set, UCI Machine Learning Repository*. Available at: https://archive.ics.uci.edu/ml/datasets/Diabetic+Retinopathy+Debrecen+Data+Set.

Arauzo-Azofra, A., Benitez, J. M. and Castro, J. L. (2008) 'Consistency measures for feature selection', *Journal of Intelligent Information Systems*, 30(3). doi: 10.1007/s10844-007-0037-0.

Arbelaitz, O. *et al.* (2013) 'An extensive comparative study of cluster validity indices', *Pattern Recognition*, 46(1), pp. 243–256. doi: 10.1016/j.patcog.2012.07.021.

Bartlett, M. S. (1950) 'Tests of significance in factor analysis', *British Journal of Statistical Psychology*, 3(2), pp. 77–85. doi: 10.1111/j.2044-8317.1950.tb00285.x.

Battiti, R. (1994) 'Using Mutual Information for Selecting Features in Supervised Neural Net Learning', *IEEE Transactions on Neural Networks*, 5(4), pp. 537–550. doi: 10.1109/72.298224.

Belis, M. and Guiasu, S. (1968) 'A quantitative-qualitative measure of information in cybernetic systems', *IEEE Transactions on Information Theory*, 14(4), pp. 593–594.

Bell, D. A. and Wang, H. (2000) 'Formalism for relevance and its application in feature subset selection', *Machine Learning*, 41(2), pp. 175–195. doi: 10.1023/A:1007612503587.

Bins, J. and Draper, B. A. (2001) 'Feature selection from huge feature sets', *Proceedings of the IEEE International Conference on Computer Vision*. doi: 10.1109/ICCV.2001.937619.

Bishop, C. M. (2006) *Pattern Recognition and Machine Learning*. New York: Springer ScienceBusiness Media.

BlackRock (2017a) *iShares MSCI Emerging Markets ETF*. Available at: https://www.ishares.com/us/products/239637/EEM (Accessed: 20 May 2017).

BlackRock (2017b) *iShares MSCI Europe Financials ETF*. Available at: https://www.ishares.com/us/products/239645/ishares-msci-europe-financials-etf (Accessed: 30 April 2017).

Blum, A. L. and Langley, P. (1997) 'Selection of relevant features and examples in machine learning', *Artificial Intelligence*, 97, pp. 245–271.

Bodie, Z., Kane, A. and Marcus, A. J. (2009) *Investments*. 8th edn. Irwin: Mc Graw-Hill.

Breiman, L. *et al.* (1984) *Classification and regression trees, Wadsworth International Group*.

Breiman, L. (2001) 'Random forests', *Machine Learning*, 45(1), pp. 5–32. doi: 10.1023/A:1010933404324.

Calinski, T. and Harabasz, J. (1974) 'A dendrite method for cluster analysis', *Communications in Statistics - Theory and Methods*, 3(1), pp. 1–27. doi: 10.1080/03610927408827101.

Cangelosi, R. and Goriely, A. (2007) 'Component retention in principal component analysis with application to cDNA microarray data.', *Biology direct*, 2, p. 2. doi: 10.1186/1745-6150-2-2.

Caruana, R. and Freitag, D. (1994a) 'Greedy Attribute Selection', *International Conference on Machine Learning*, 48, pp. 28–36. doi: 10.1.1.41.3576.

Caruana, R. and Freitag, D. (1994b) 'How useful is relevance?', *AAAI Fall Symposium Technical Report FS-94-02*.

Cattell, R. B. (1966) 'The Scree Test For The Number Of Factors', *Multivariate Behavioral Research*, 1(2), pp. 245–276. doi: 10.1207/s15327906mbr0102_10.

Chan, T. F. (1987) 'Rank revealing QR factorizations', *Linear Algebra and Its Applications*, 88–89, pp. 67–82. doi: 10.1016/0024-3795(87)90103-0.

Chandrashekar, G. and Sahin, F. (2014) 'A survey on feature selection methods', *Computers and Electrical Engineering*, 40, pp. 16–28.

Chen, L. H. and Hsiao, H. Der (2008) 'Feature selection to diagnose a business crisis by using a real GA-based support vector machine: An empirical study', *Expert Systems with*

Applications, 35(3), pp. 1145–1155. doi: 10.1016/j.eswa.2007.08.010.

Chormunge, S. and Jena, S. (2018) 'Correlation based feature selection with clustering for highdimensional data', *Journal of Electrical Systems and Information Technology*, 5, pp. 542–549.

Chun, S.-H. and Park, Y.-J. (2005) 'Dynamic adaptive ensemble case-based reasoning: application to stock market prediction', *Expert Systems with Applications*, 28, pp. 435–443.

Cover, T. and Hart, P. (1967) 'Nearest neighbor pattern classification', *IEEE Transactions on Information Theory*, 13(1), pp. 21–27. doi: 10.1109/TIT.1967.1053964.

Cover, T. M. (1974) 'The Best Two Independent Measurements Are Not the Two Best', *IEEE Transactions on Systems, Man and Cybernetics*, SMC-4(1), pp. 116–117. doi: 10.1109/TSMC.1974.5408535.

Das, S. (2001) 'Filters, wrappers and a boosting-based hybrid for feature selection', *Proceedings of the 18th International Conference on Machine Learning*, pp. 74–81.

Dash, M. (1997) 'Feature selection via set cover', in *Proceedings of the IEEE Knowledge and Data Engineering Exchange Workshop*, pp. 165–171.

Dash, M. and Liu, H. (1997) 'Feature selection for classification', *Intelligent Data Analysis*, 1(1–4), pp. 131–156. doi: 10.3233/IDA-1997-1302.

Dash, M. and Liu, H. (2003) 'Consistency-based search in feature selection', *Artificial Intelligence*, 151(1–2), pp. 155–176. doi: 10.1016/S0004-3702(03)00079-1.

Dessì, N. and Pes, B. (2015) 'Similarity of feature selection methods: An empirical study across data intensive classification tasks', *Expert Systems with Applications*, 42(10), pp. 4632–4642. doi: 10.1016/j.eswa.2015.01.069.

Dietterich, T. (1997) 'Machine-Learning Research Four Current Directions', *AI Magazine*, 18(4), pp. 97–136.

Doak, J. (1992) *An Evaluation of Feature Selection Methodsand Their Application to Computer Security, UC Davis Dept of Computer Science tech reports*.

Dougherty, G. (2013) *Pattern Recognition and Classification: An Introduction*. New York: Springer ScienceBusiness Media.

Duda, R. O., Hart, P. E. and Stork, D. G. (2012) 'Pattern classification', *John Wiley & Sons, Inc.*

Elashoff, J. E., Elashoff, R. M. and Goldman, G. E. (1967) 'On the choice of variables in

classification problems with dichotomous variables.', *Biometrika*, 54(3), pp. 668–670. doi: 10.1093/biomet/54.3-4.668.

Fadlalla, A. and Amani, F. (2014) 'Predicting next day closing price of Qatar Exchange Index using technical indicators and artificial neural network', *Intelligent Systems in Accounting, Finance and Management*, 21, pp. 209–223.

Felsen, J. (1975) 'Learning pattern recognition techniques applied to stock market forecasting', *IEEE Transactions on Systems, Man, and Cybernetics*, 5(6), pp. 583–594.

Gu, Q., Li, Z. and Han, J. (2011) 'Generalized Fisher score for feature selection', in *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 266–273.

Guo-qiang, X. (2011) 'The optimization of share price prediction model based on support vector machine', *International Conference on Control, Automation and Systems Engineering (CASE)*, pp. 1–4.

Guo, Z. *et al.* (2014) 'Fusion based forecasting model for financial time series', *PLoS ONE*, 9(6), pp. 1–13.

Gupta, P., Doermann, D. and DeMenthon, D. (2002) 'Beam search for feature selection in automatic SVM defect classification', *Proceedings - International Conference on Pattern Recognition*. doi: 10.1109/ICPR.2002.1048275.

Guttman, L. (1954) 'Some necessary conditions for common-factor analysis', *Psychometrika*, 19(2), pp. 149–161. doi: 10.1007/BF02289162.

Guvenir, H. A., Acar, B. and Muderrisoglu, H. (1998) *Arrhythmia Data Set, UCI Machine Learning Repository*. Available at: https://archive.ics.uci.edu/ml/datasets/arrhythmia (Accessed: 10 June 2019).

Guyon, I. and Elisseeff, A. (2003) 'An introduction to variable and feature selection', *Journal of Machine Learning Research*, 3, pp. 1157–1182.

Hall, M. A. (2000) 'Correlation-based feature selection for discrete and numeric class machine learning', *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 359–366. doi: 10.1.1.34.4393.

Hall, M. A. and Holmes, G. (2003) 'Benchmarking Attribute Selection Techniques for Discrete Class Data Mining', *IEEE Transactions on Knowledge and Data Engineering*, 15(6), pp. 1437–1447. doi: 10.1109/TKDE.2003.1245283.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, *Springer Series in Statistics*. doi: 10.1007/b94608.

He, X., Cai, D. and Niyogi, P. (2005) ‘Laplacian Score for Feature Selection’, in *Proceedings NIPS*, pp. 507–514. doi: http://books.nips.cc/papers/files/nips18/NIPS2005_0149.pdf.

Horn, J. L. (1965) ‘A rationale and test for the number of factors in factor analysis’, *Psychometrika*, 30(2), pp. 179–185. doi: 10.1007/BF02289447.

Hurwitz, E. and Marwala, T. (2011) ‘Suitability of using technical indicator-based Strategies as potential strategies within intelligent trading systems’, *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 80–84.

Ilter, N. and Guvenir, H. A. (1998) *Dermatology Data Set, UCI Machine Learning Repository*. Available at: https://archive.ics.uci.edu/ml/datasets/Dermatology (Accessed: 5 March 2017).

John, G. H., Kohavi, R. and Pfleger, K. (1994) ‘Irrelevant Features and the Subset Selection Problem’, in *Machine Learning Proceedings 1994*, pp. 121–129. doi: 10.1016/B978-1-55860-335-6.50023-4.

Kaiser, H. F. (1961) ‘A note on Guttman’s lower bound for the number of common factors’, *British Journial of Psychology*, 14, pp. 1–2.

Karegowda, A. G., Manjunath, A. S. and Jayaram, M. A. (2010) ‘Comparative Study of Attribute Selection Using Gain Ratio and Correlation Based Feature Selection’, *International Journal of Information Technology and Knowledge Management*, 2(2), pp. 271–277.

Kim, M. J., Han, I. and Lee, K. C. (2004) ‘Hybrid knowledge integration using the fuzzy genetic algorithm: Prediction of the Korea Stock Index’, *Intelligent Systems in Accounting, Finance and Management*, 12, pp. 43–60.

Kira, K. and Rendell, L. (1992a) *A practical approach to feature selection, Proceedings of the ninth international workshop on Machine learning*. doi: 10.1016/S0031-3203(01)00046-2.

Kira, K. and Rendell, L. (1992b) ‘The feature selection problem: traditional methods and a new algorithm’, *AAAI-92 Proceedings*, pp. 129–134.

Klawonn, F. and Castro, J. L. (1995) ‘Similarity in fuzzy reasoning’, *Mathware and Soft Computing*, 2, pp. 197–228.

Kohavi, R. and John, G. H. (1997) ‘Wrappers for feature subset selection’, *Artificial Intelligence*, 97, pp. 273–324.

Kononenko, I. (1994) ‘Estimating attributes: analysis and extensions of Relief’, in De Raedt, L. and Bergadano, F. (eds) *Machine Learning: ECML-94*, pp. 171–182.

Kononenko, I., Simec, E. and Robnik-Sikonja, M. (1997) ‘Overcoming the myopia of inductive learning Algorithms with RELIEFF’, *Applied Intelligence*, 7, pp. 39–55.

Koutroumbas, S. and Theodoridis, K. (2003) *Pattern Recognition*. 2nd Editio. San Diego: Academic Press. doi: 10.1016/B978-012369531-4/50016-0.

Krollner, B., Vanstone, B. and Finnie, G. (2010) ‘Financial time series forecasting with machine learning techniques: A survey’, *Proceedings of the 18th European Symposium on Artificial Neural Networks (ESANN 2010)*, pp. 25–30.

Kurama, O., Luukka, P. and Collan, M. (2016) ‘A Similarity Classifier with Bonferroni Mean Operators’, *Advances in Fuzzy Systems*, 2016. doi: 10.1155/2016/7173054.

Leigh, W., Purvis, R. and Ragusa, J. M. (2002) ‘Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support’, *Decision Support Systems*, 32, pp. 361–377.

Li, J. *et al.* (2017) ‘Feature Selection: A Data Perspective’, *ACM Computing Surveys (CSUR)*, 50(6), p. 94. doi: 10.1145/3136625.

Lichman, M. (2013) *UCI Machine Learning Repository*, *UCI Machine Learning Repository*. Available at: http://archive.ics.uci.edu/ml (Accessed: 5 March 2017).

Liu, H. and Motoda, H. (2001) *Feature extraction, construction and selection: A data mining perspective*, *Springer Science & Business Media*.

Liu, H., Motoda, H. and Dash, M. (1998) ‘A monotonic measure for optimal feature selection’, in *Proceedings of the 10th European Conference on Machine Learning*, pp. 101–106.

Liu, H. and Setiono, R. (1996) ‘A probabilistic approach to feature selection - a filter solution’, *Proceedings of the 13th International Conference on Machine Learning*. doi: citeulike-article-id:7791632.

Liu, H. and Yu, L. (2005) ‘Toward integrating feature selection algorithms for classification and clustering’, *IEEE Transactions on Knowledge and Data Engineering*, 17(4), pp. 491–502. doi: 10.1109/TKDE.2005.66.

Di Lorenzo, R. (2013) *Basic Technical Analysis of Financial Markets. A Modern Approach*. Milan: Springer Italia.

De Luca, A. and Termini, S. (1972) ‘A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory’, *Information and Control*, 20, pp. 301–312.

Luukka, P. (2008) ‘Similarity classifier in diagnosis of bladder cancer’, *Computer Methods and Programs in Biomedicine*, 89, pp. 43–49.

Luukka, P. (2010) ‘Nonlinear fuzzy robust PCA algorithms and similarity classifier in bankruptcy analysis’, *Expert Systems with Applications*, 37(12), pp. 8296–8302. doi: 10.1016/j.eswa.2010.05.055.

Luukka, P. (2011) ‘Feature selection using fuzzy entropy measures with similarity classifier’, *Expert Systems with Applications*, 38, pp. 4600–4607.

Luukka, P. and Kurama, O. (2013) ‘Similarity classifier with ordered weighted averaging operators’, *Expert Systems with Applications*, 40, pp. 995–1002.

Luukka, P. and Lampinen, J. (2011) ‘Differential evolution classifier in noisy settings and with interacting variables’, *Applied Soft Computing Journal*. doi: 10.1016/j.asoc.2010.01.009.

Luukka, P. and Lampinen, J. (2015) ‘Differential evolution based multiple vector prototype classifier’, *Computing and Informatics*, 34(5), pp. 1151–1167.

Luukka, P. and Leppälampi, T. (2006) ‘Similarity classifier with generalized mean applied to medical data’, *Computers in Biology and Medicine*, 36, pp. 1026–1040.

Luukka, P. and Lohrmann, C. (2019) ‘Information Transmission and Nonspecificity in Feature Selection’, in Kearfott R. et al. (eds) *Fuzzy Techniques: Theory and Applications. IFSA/NAFIPS 2019 2019. Advances in Intelligent Systems and Computing, vol 1000*. Springer, pp. 340–350.

Luukka, P., Saastamoinen, K. and Könönen, V. (2001) ‘A classifier based on the maximal fuzzy similarity in the generalized Lukasiewicz-structure’, *10th IEEE International Conference on Fuzzy Systems*.

Maes, S., Tuyls, K. and Vanschoenwinkel, B. (2002) ‘Credit card fraud detection using bayesian and neural networks’, in *Proceedings of the 1st International NAISO Conference on Neuro-Fuzzy Technologies*.

Maldonado, S., Pérez, J. and Bravo, C. (2017) ‘Cost-based feature selection for Support Vector Machines: An application in credit scoring’, *European Journal of Operational Research*, 261(2), pp. 656–665. doi: 10.1016/j.ejor.2017.02.037.

Martínez Sotoca, J. and Pla, F. (2010) ‘Supervised feature selection by clustering using conditional mutual information-based distances’, *Pattern Recognition*, 43, pp. 2068–2081. doi: 10.1016/j.patcog.2009.12.013.

McLeish, M. and Cecile, M. (1989) *Horse colic data set, UCI Machine Learning Repository*. Available at: https://archive.ics.uci.edu/ml/datasets/Horse+Colic.

Mitra, P., Murthy, C. A. and Pal, S. K. (2002) ‘Unsupervised feature selection using feature similarity’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*,

24(3), pp. 301–312. doi: 10.1109/34.990133.

Molina, L. C., Belanche, L. and Nebot, À. (2002) 'Feature selection algorithms: a survey and experimental evaluation', *In Proceedings of the IEEE International Conference on Data Mining 2002*, pp. 306–313.

Morningstar (2017) *Vanguard Total World Stock Index Fund ETF*. Available at: http://www.morningstar.co.uk/uk/etf/snapshot/snapshot.aspx?id=0P0000G5T2.

Motoda, H. and Liu, H. (2002) 'Feature selection, extraction and construction', *Communication of IICM (Institute of Information and Computing Machinery, Taiwan)*, 5(2), pp. 67–72.

Narendra, P. M. and Fukunaga, K. (1977) 'A Branch and Bound Algorithm for Feature Subset Selection', *IEEE Transactions on Computers*, C-26(9), pp. 917–922. doi: 10.1109/TC.1977.1674939.

O'Connor, B. P. (2000) 'SPSS and SAS programs for determining the number of components using parallel analysis and Velicer's MAP test', *Behavior Research Methods, Instruments, & Computers*, 32(3), pp. 396–402.

Parkash, O., Sharma, P. and Mahajan, R. (2008) 'New measures of weighted fuzzy entropy and their applications for the study of maximum weighted fuzzy entropy principle', *Information Sciences*, 178, pp. 2389–2395.

Pätäri, E. and Vilska, M. (2014) 'Performance of moving average trading strategies over varying stock market conditions: The Finnish evidence', *Applied Economics*, 46(24), pp. 2851–2872. doi: 10.1080/00036846.2014.914145.

Patel, J. *et al.* (2015) 'Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques', *Expert Systems with Applications*, 42(1), pp. 259–268. doi: 10.1016/j.eswa.2014.07.040.

Piramuthu, S. (2004) 'Evaluating feature selection methods for learning in data mining applications', *European Journal of Operational Research*, 156(2), pp. 483–494. doi: 10.1016/S0377-2217(02)00911-6.

Piramuthu, S. and Sikora, R. T. (2009) 'Iterative feature construction for improving inductive learning algorithms', *Expert Systems with Applications*, 2(2), pp. 3401–3406. doi: 10.1016/j.eswa.2008.02.010.

Pun, J. and Lawryshyn, Y. (2012) 'Improving Credit Card Fraud Detection using a Meta-Classification Strategy', *International Journal of Computer Applications*, 56(10), pp. 41–46. doi: 10.5120/8930-3007.

Quinlan, J. R. (1986) 'Induction of Decision Trees', *Machine Learning*, 1(1), pp. 81–106.

doi: 10.1023/A:1022643204877.

Rhee, F. and Lee, Y. (1999) ‘Unsupervised Feature Selection using a Fuzzy-Genetic Algorithm’, in *Proceedings of the International Conference on Fuzzy Systems*, pp. 1266–1269.

Robnik-Šikonja, M. and Kononenko, I. (2003) ‘Theoretical and Empirical Analysis of ReliefF and RReliefF’, *Machine Learning*, 53(1–2), pp. 23–69. doi: 10.1023/A:1025667309714.

Rodriguez-Galiano, V. F. *et al.* (2018) ‘Feature selection approaches for predictive modelling of groundwater nitrate pollution: An evaluation of filters, embedded and wrapper methods’, *Science of the Total Environment*, 624, pp. 661–672. doi: 10.1016/j.scitotenv.2017.12.152.

Rossilo, R., Giner, J. and De la Fuente, D. (2014) ‘The effectiveness of the combined use of VIX and support vector machines on the prediction of S&P 500’, *Neural Computing and Applications*, 25, pp. 321–332.

Rousseeuw, P. J. (1987) ‘Silhouettes: A graphical aid to the interpretation and validation of cluster analysis’, *Journal of Computational and Applied Mathematics*, 20(C), pp. 53–65. doi: 10.1016/0377-0427(87)90125-7.

Rudebusch, G. D. and Williams, J. C. (2009) ‘Forecasting recessions: The puzzle of the enduring power of the yield curve’, *Journal of Business and Economic Statistics*, 27(4), pp. 492–503. doi: 10.1198/jbes.2009.07213.

Russell, S. and Norvig, P. (2009) ‘Artificial Intelligence: A Modern Approach, 3rd edition’, *Prentice Hall*, pp. 1–1132. doi: 10.1017/S0269888900007724.

S&P Dow Jones Indices (2017) *S&P GSCI CRUDE OIL*. Available at: https://us.spindices.com/indices/commodities/sp-gsci-crude-oil (Accessed: 10 April 2017).

Saeys, Y., Inza, I. and Larranaga, P. (2007) ‘A review of feature selection techniques in bioinformatics’, *Bioinformatics*, 23(19), pp. 2507–2517.

Sahu, B., Dehuri, S. and Jagadev, A. K. (2017) ‘Feature selection model based on clustering and ranking in pipeline for microarray data’, *Informatics in Medicine Unlocked*, 9, pp. 107–122. doi: 10.1016/j.imu.2017.07.004.

Sammut, C. and Webb, G. I. (2017) *Encyclopedia of Machine Learning and Data Mining 2017 Edition*. New York: Springer Science+Business Media.

Soundarapandian, P. and Rubini, L. (2015) *Chronic Kidney Disease Data Set, UCI Machine Learning Repository*. Available at:

https://archive.ics.uci.edu/ml/datasets/Chronic_Kidney_Disease (Accessed: 5 March 2017).

Souza, J. T., Matwin, S. and Japkowicz, N. (2006) 'Parallelizing feature selection', *Algorithmica*, 45, pp. 433–456.

State Street Global Advisors (SPDR) (2017a) *SPDR S&P U.S. Financials Select Sector UCITS ETF*. Available at: https://fi.spdrs.com/en/professional/etf/spdr-sp-us-financials-select-sector-ucits-etf-ZPDF-GY?cid=1365706 (Accessed: 5 May 2017).

State Street Global Advisors (SPDR) (2017b) *SPDR S&P U.S. Materials Select Sector UCITS ETF*. Available at: https://fi.spdrs.com/en/professional/etf/SPDR-SP-US-Materials-Select-Sector-UCITS-ETF-ZPDM-GY (Accessed: 5 May 2017).

Sugar, C. and James, G. (2003) 'Finding the number of clusters in a data set : An information theoretic approach', *Journal of the American Statistical Association*, 98, pp. 750–763. doi: 10.1198/016214503000000666.

Teixeira, L. A. and De Oliveira, A. L. I. (2010) 'A method for automatic stock trading combining technical analysis and nearest neighbor classification', *Expert Systems with Applications*, 37(10), pp. 6885–6890. doi: 10.1016/j.eswa.2010.03.033.

Tibshirani, R., Walther, G. and Hastie, T. (2001) 'Estimating the number of clusters in a data set via the gap statistic', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2), pp. 411–423. doi: 10.1111/1467-9868.00293.

Toussaint, G. T. (1971) 'Note on Optimal Selection of Independent Binary-Valued Features for Pattern Recognition', *IEEE Transactions on Information Theory*, 17(5), p. 618. doi: 10.1109/TIT.1971.1054685.

Trambaiolli, L. R. *et al.* (2017) 'Feature selection before EEG classification supports the diagnosis of Alzheimer's disease', *Clinical Neurophysiology*, 128(10), pp. 2058–2067. doi: 10.1016/j.clinph.2017.06.251.

Tsaih, R. *et al.* (2004) 'Credit scoring system for small business loans', *Decision Support Systems*, 38(1), pp. 91–99. doi: 10.1016/S0167-9236(03)00079-4.

USCF (2017) *United States Commodity Index Fund*. Available at: http://www.uscfinvestments.com/usci (Accessed: 5 May 2017).

Vafaie, H. and Imam, I. F. (1994) 'Feature Selection Methods : Genetic Algorithms vs . Greedy-like Search', in *Proceedings of the International Conference on Fuzzy and Intelligent Control Systems*.

Velicer, W. F. (1976) 'Determining the number of components from the matrix of partial correlations', *Psychometrika*, 41(3), pp. 321–327. doi: 10.1007/BF02293557.

Warton, D. I. (2008) ‘Penalized normal likelihood and ridge regularization of correlation and covariance matrices’, *Journal of the American Statistical Association*, 103(481), pp. 340–349. doi: 10.1198/016214508000000021.

Webb, A. R. (2002) *Statistical Pattern Recognition*. Malvern: John Wiley Sons.

West, D., Dellana, S. and Qian, J. (2005) ‘Neural network ensemble strategies for financial decision applications’, *Computers and Operations Research*, 32(10), pp. 2543–2559. doi: 10.1016/j.cor.2004.03.017.

Witten, I. and Frank, E. (2005) *Data Mining: Practical machine learning tools and techniques*. 2nd editio, *Machine Learning*. 2nd editio. San Francisco: Morgan Kaufman.

Wnek, J. and Michalski, R. S. (1994) ‘Hypothesis-Driven Constructive Induction in AQ17-HCI: A Method and Experiments’, *Machine Learning*, 14(2), pp. 139–168. doi: 10.1023/A:1022622132310.

Wolberg, W. H. (1992) *Breast cancer Wisconsin (original) data set, UCI Machine Learning Repository*. Available at: https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(original).

Xu, L., Yan, P. and Chang, T. (1998) ‘Best first strategy for feature selection’, in *Proceedings of the 9th International Conference on Pattern Recognition*, pp. 706–708.

Yahoo Finance (2017) *Selected time series*. Available at: https://finance.yahoo.com/.

Yao, Y. Y., Wong, S. K. and Butz, C. J. (1999) ‘On information-theoretic measures of attribute importance’, *PacificAsia Conference on Knowledge Discovery and Data Mining*, pp. 133–137.

Zadeh, L. A. (1965) ‘Fuzzy sets’, *Information and Control*, 8, pp. 338–353.

Zadeh, L. A. (1971) ‘Similarity relations and fuzzy orderings’, *Information Sciences*, 3, pp. 177–200.

Zheng, H. and Zhang, Y. (2008) ‘Feature selection for high-dimensional data in astronomy’, *Advances in Space Research*, 41(12), pp. 1960–1964. doi: 10.1016/j.asr.2007.08.033.

Zhora, D. V. (2005) ‘Data preprocessing for stock market forecasting using random subspace classifier network’, *Proceedings of International Joint Conference on Neural Networks, Montreal, Canada*, pp. 2549–2554.

Zhou, X. *et al.* (2017) ‘Eye tracking data guided feature selection for image classification’, *Pattern Recognition*, 63, pp. 56–70. doi: 10.1016/j.patcog.2016.09.007.

Zwick, W. R. and Velicer, W. F. (1982) ‘Factors influencing four rules for determining the number of components to retain’, *Multivariate Behavioral Research*, 17(2), p. 253. doi: 10.1207/s15327906mbr1702_5.

Zwick, W. R. and Velicer, W. F. (1986) ‘Comparison of five rules for determining the number of components to retain’, *Psychological Bulletin*, 99(3), p. 432. doi: 10.1037/0033-2909.99.3.432.

Publication I

Lohrmann, C., Luukka, P., Jabłońska-Sabuka, M., and Kauranne, T.
A combination of fuzzy similarity measures and fuzzy entropy measures for supervised feature selection

Reprinted with permission from
Expert Systems with Applications
Vol. 110, pp. 216-236, 2018

Expert Systems With Applications 110 (2018) 216–236

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A combination of fuzzy similarity measures and fuzzy entropy measures for supervised feature selection

Christoph Lohrmann[a,b,*], Pasi Luukka[b], Matylda Jablonska-Sabuka[a], Tuomo Kauranne[a]

[a] School of Engineering Science, Lappeenranta University of Technology, Skinnarilankatu 34, 53850 Lappeenranta, Finland
[b] School of Business and Management, Lappeenranta University of Technology, Skinnarilankatu 34, 53850 Lappeenranta, Finland

ARTICLE INFO

Article history:
Received 13 October 2017
Revised 31 May 2018
Accepted 1 June 2018
Available online 3 June 2018

Keywords:
Feature ranking
Filter method
Wrapper method
Machine learning
ReliefF

ABSTRACT

Large amounts of information and various features are in many machine learning applications available, or easily obtainable. However, their quality is potentially low and greater volumes of information are not always beneficial for machine learning, for instance, when not all available features in a data set are relevant for the classification task and for understanding the studied phenomenon. Feature selection aims at determining a subset of features that represents the data well, gives accurate classification results and reduces the impact of noise on the classification performance. In this paper, we propose a filter feature ranking method for feature selection based on fuzzy similarity and entropy measures (FSAE), which is an adaptation of the idea used for the wrapper function by Luukka (2011) and has an additional scaling factor. The scaling factor to the feature and class-specific entropy values that is implemented, accounts for the distance between the ideal vectors for each class. Moreover, a wrapper version of the FSAE with a similarity classifier is presented as well. The feature selection method is tested on five medical data sets: dermatology, chronic kidney disease, breast cancer, diabetic retinopathy and horse colic. The wrapper version of FSAE is compared to the wrapper introduced by Luukka (2011) and shows at least as accurate results with often considerably fewer features. In the comparison with ReliefF, Laplacian score, Fisher score and the filter version of Luukka (2011), the FSAE filter in general achieves competitive mean accuracies and results for one medical data set, the breast cancer Wisconsin data set, together with the Laplacian score in the best results over all possible feature removals.

1. Background

In the field of machine learning, researchers and practitioners are commonly faced with classification problems, which is assigning observations to discrete classes premised on the characteristics of the observations (Bishop, 2006). The classification is conducted based on the features, where the term 'feature' can in general refer to a variable, attribute or aspect in the data set that was observed and recorded (e.g. height of a person) or constructed from other variables (such as a principal component score in principal component analysis). The objective of the classification is to obtain decision regions of the feature space for each class (Luukka, 2008). The class refers to the group that an observation belongs to, which can be intuitively defined (e.g. positive or negative diagnosis) or defined in another way by the data analyst. In many machine learning applications, information and features are available in large volumes, or easily obtainable, but are potentially of low quality (Blum & Langley, 1997). Moreover, increased amounts of information are not always beneficial (Chandrashekar & Sahin, 2014), for example, when using data sets in which not all the available features are relevant for understanding the studied phenomenon and carrying out the classification task (Luukka, 2007). Classification refers to the assignment of observations into discrete categories, called classes (Bishop, 2006). If the information required for effective classification can be obtained with fewer features, the classification process becomes less computationally expensive and irrelevant features, which may be a source of undesirable noise, can be eliminated (Chandrashekar & Sahin, 2014). In other words, the inclusion and use of irrelevant features can lead to bias and reduce classification accuracy (Chandrashekar & Sahin, 2014; Dougherty, 2013). Consequently, only those features should be selected for a model that are relevant for the classification and lead to qualitatively good classification results (Luukka & Leppälampi, 2006). Such feature selection aims to determine a subset of features that represents the data well, reduces the effect of noise and gives accurate classification results (Chandrashekar & Sahin, 2014). Theo-

* Corresponding author at: School of Business and Management, Lappeenranta University of Technology, Skinnarilankatu 34, 53850 Lappeenranta, Finland.
E-mail addresses: christoph.lohrmann@student.lut.fi (C. Lohrmann), pasi.luukka@lut.fi (P. Luukka), matylda.jablonska-sabuka@lut.fi (M. Jablonska-Sabuka), tuomo.kauranne@lut.fi (T. Kauranne).

https://doi.org/10.1016/j.eswa.2018.06.002
0957-4174/

retical study has demonstrated that by concentrating on a small subset of the features of a data set, the generalization ability of the classifier, i.e. the performance of the classifier on previously unseen observations, can be improved (Blumer, Ehrenfeucht, Haussler, & Warmuth, 1987). The objective of feature selection thus becomes to choose a feature subset that can augment the performance of the classifier or, at a minimum, remove features that lead only to a minimal performance degradation (Liang, Yang, & Winstanley, 2008).

When discussing such data treatment, it is imperative to discriminate feature selection from techniques that conduct feature extraction. Feature extraction also relates to dimensional reduction of the feature space, but it does so by creating new features. Feature selection, on the other hand, is a dimensional reduction technique in which a subset of existing features is selected (Kittler & Mardia, 1994). Feature selection exists in both, supervised and unsupervised forms, i.e. where the associated class label for the observations is known (supervised feature selection) or unknown (unsupervised feature selection) (Liang et al., 2008). In this paper, we focus on supervised feature selection.

Feature selection methods are commonly divided into three types: filter techniques, wrapper functions and embedded functions (Blum & Langley, 1997). Filter techniques do not use the classifier during feature selection. They are applied in the feature preprocessing step (Luukka, 2011). The aim of such filter techniques is to filter out features based on their characteristics and consistency (Blum & Langley, 1997; Junttila, Maltamo, & Kauranne, 2008; Seo & Oh, 2012). The wrapper approach is also applied during preprocessing but includes the classifier in the feature selection (Luukka, 2011). The classification accuracy of the classifier takes the role of the feature selection criterion (Chandrashekar & Sahin, 2014; Seo & Oh, 2012). The feature subset is chosen that leads to the best test performance with the classifier. Using the classifier to find the optimal feature subset provides superior performance compared to a simple filter technique but is also computationally more expensive (Blum & Langley, 1997; Seo & Oh, 2012). It should be noted that the classifier deployed after feature selection should be the same as during feature selection, because the feature selection is optimized with respect to the particular classifier (Liang et al., 2008). The last type of feature selection techniques are the embedded methods. Embedded methods are incorporated into the training process and search for the optimal feature subset during the training (Chandrashekar & Sahin, 2014; Seo & Oh, 2012). The reasoning behind this approach is to decrease the computational time required by deploying the classifier only during the training process (Chandrashekar & Sahin, 2014). It must be kept in mind that the chosen feature subset is also optimized for a given classifier (Seo & Oh, 2012).

The supervised feature selection method discussed in this paper is an adaptation of the wrapper method introduced by Luukka (2011) in which a combination of entropy measures and a similarity classifier is used to determine the least relevant features in the dataset. This feature selection method has demonstrated to be capable of reducing the number of features to a subsets that gives an improvement in classification performance or only a small degradation in classification accuracy (Luukka, 2011). The method that will be discussed in this paper is conceptually a feature ranking method (filter). However, it can also be used as a wrapper method, for instance with a similarity classifier as the method by Luukka (2011). A similarity classifier is a suitable choice for the wrapper version of the adapted method presented in this paper since it has demonstrated high classification accuracy in several applications with medical data sets (Luukka, 2008; Luukka & Leppälampi, 2006).

On these data sets it was shown that a similarity classifier can outperform other machine learning techniques such as linear discriminant analysis, multi-layer perceptrons and the C4.5 algorithm (Quinlan, 1992). The main advantages of this classifier type are that it requires a comparably small computational time and can accomplish high classification accuracy with a small number of samples (Luukka, 2008). Recent research on a similarity classifier with multiple ideal vectors also indicated the competitive performance of a similarity classifier with other machine learning techniques (Lohrmann & Luukka, 2018).

2. Objectives

The objective of this paper is to point out a clear deficiency of the feature selection algorithm by Luukka (2011) that occurs for certain data structures, demonstrate a measure that helps to detect such data structures and address this deficiency with an adapted feature selection algorithm. Moreover, the feature selection method will be introduced as a supervised filter method. However, we will also present the application as a wrapper method together with a similarity classifier that uses backward elimination of features, which is the process behind the initial algorithm proposed by Luukka (2011). In the work, we first show that the feature selection technique developed by Luukka (2011) may remove essential features of a data set and, thus, lead to performance degradation. This deficiency derives from the inadequate consideration of the difference between ideal vectors in the feature selection algorithm. The idea is that good class separability is, beside other factors, also dependent on the length of the interclass distances, which is a notion also found in filter techniques such as CScore (Seo & Oh, 2012). A simple measure based on the standard deviation of class means of a feature from the feature mean is presented that supports the detection of the cases where the feature selection by Luukka (2011) will have this deficiency. To overcome this weakness, an adjusted version of the feature selection algorithm is introduced, where a scaling factor to the feature and class-specific entropy values is used to account for the distance between the ideal vectors for each class. The goal is to avoid the removal of essential features and achieve a better classification performance or as small performance degradation as possible. The proposed feature selection method is tested on three simple artificial tasks and five real-world medical data sets.

3. Methods

3.1. Entropy measures

Entropy can be regarded as a "measure of the degree of fuzziness" (De Luca & Termini, 1972). Additionally, De Luca and Termini (1972) describe it as the average information contained in data that is available for making a decision, e.g. to classify objects. An entropy measure for input $\in [0,1]$ has to satisfy at least the following properties (De Luca & Termini, 1972):

(1) Entropy $= 0$ if the input value is 0 or 1
(2) The maximum entropy value is obtained for an input of 0.5
(3) The entropy of input f has to be greater or equal to the entropy of f^* where f^* is any "sharpened" version of f, which is any fuzzy set such that $f^*(x) \geq f(x)$ if $f(x) \geq 0.5$ and $f^*(x) \leq f(x)$ if $f(x) \leq 0.5$

In this paper, the entropy measures developed by De Luca and Termini (1972) and Parkash, Sharma, and Mahajan (2008) are applied to feature selection. The entropy introduced by De Luca and Termini (1972) can be described as follows:

$$H(A) = -\sum_{i=1}^{n} [\mu_A(x_i)\log\mu_A(x_i) + (1-\mu_A(x_i))\log(1-\mu_A(x_i))] \tag{1}$$

where $\mu_A(x_i) \in [0, 1]$ is the membership degree of x_i to the fuzzy set A.

The entropy measures developed by Parkash et al. (2008) are related to the concept of weighted entropy (Belis & Guiasu, 1968) and are defined in the following way:

$$H^1(A) = \sum_{i=1}^{n} w_i \left[\sin\frac{\pi\,\mu_A(x_i)}{2} + \sin\frac{\pi\,(1-\mu_A(x_i))}{2} - 1\right] \tag{2}$$

$$H^2(A) = \sum_{i=1}^{n} w_i \left[\cos\frac{\pi\,\mu_A(x_i)}{2} + \cos\frac{\pi\,(1-\mu_A(x_i))}{2} - 1\right] \tag{3}$$

The entropy measures are applied to classification tasks since small entropy values signal regularities and structure in the data, whereas high entropy values indicate randomness (Yao, Wong, & Butz, 1999). Thus, entropy can show whether the data is informative or whether it is characterized by uncertainty (Luukka, 2007). More specifically, fuzzy entropy measures can be used to determine the relevance of features in a data set (Luukka, 2011). Since both entropy measures of Parkash et al. (2008) lead to the same entropy values, only the first entropy measure will in the following be deployed.

3.2. Feature selection algorithm using a similarity classifier and entropy measures

Feature selection using a combination of a similarity classifier and entropy measures was introduced by Luukka (2011). This wrapper feature selection algorithm uses fuzzy entropy measures to determine the importance of features. Similarity values are deployed as input to the entropy measure. As mentioned in the previous section, similarity $S \in [0, 1]$, where 0 indicates that an observation is completely dissimilar from the ideal vector and 1 indicates the highest possible degree of similarity. The corresponding entropy values for similarities 0 or 1 are low and represent high informativeness. On the other hand, a similarity that is close to 0.5 is characterized by the highest entropy value and signals uncertainty.

Luukka (2011) utilized this idea to calculate entropy values for the similarities of features with the ideal vector of each class. The entropy values are summed over all observations and classes to obtain a single entropy value for each feature d. Given that high entropy values represent uncertainty, during each step of feature selection, the feature with the largest entropy value is removed. The underlying assumption is that the feature removed does not contribute to the difference between the classes (Luukka, 2011). The accuracy of the classification with the similarity classifier is examined after each feature removal, and the procedure is stopped when performance degrades. Essentially, this procedure operates as a wrapper function with stepwise backward elimination. The procedure for the removal of a feature is illustrated in Fig. 1 for a three-class case with two features.

The corresponding pseudocode for the feature removal procedure can be found in Luukka (2011). The process as depicted by Luukka (2011) can be separated in four distinct steps.

The first step is the division of the data into training and test data, and calculation of the ideal vector for the training data. The ideal vectors can, for instance, be calculated by using generalized means over the samples for particular classes. With these ideal vectors, the performance of the similarity classifier on the test set is computed (before the removal step).

The second step is to determine the similarity S of feature d of an observation x_j of the training set with that of the ideal vector of class i, which can be stated as:

$$S\left(x_{j,d}, v_{i,d}\right) = \sqrt[p]{\left(1 - \left|x_{j,d}^p - v_{i,d}^p\right|\right)} \tag{4}$$

where the parameter p during the feature selection is set to 1. This formula is used for each sample x_j of the training data X for each feature d and class i. The result is a $n^*(DN)$ matrix, where n is the number of samples, D the number of features and N the number of classes as illustrated in Fig. 1.

The third step is concerned with the calculation of the entropy value for each feature. The equation to determine the entropy H for a feature d can subsequently be defined as:

$$H_d = \sum_{i=1}^{N}\sum_{j=1}^{n} H\left(S\left(x_{j,d}, v_{i,d}\right)\right) \tag{5}$$

where the similarity for feature d of an observation x_j with the ideal vector of class i is summed over all observations ($j = 1,\ldots, n$) and classes ($i = 1,\ldots, N$).

The fourth step is the feature removal. The feature with the highest entropy value is removed from the data set. Subsequently, the performance of the similarity classifier with the test data set before and after the removal of this feature is compared. The classifiers performance on the test set before removal was already obtained in the *first step* and solely the classfication accuracy on the test set without the removed feature has to be computed. Note, that the ideal vectors for the classes do not change, only the element the element in the ideal vector related to the removed feature is excluded. If the performance of the classifier is improved or the stopping criterion is not met, the subsequent steps are repeated. After the stopping criterion is met, the feature subset that led to the highest test performance will be selected.

Luukka (2011) applied this feature selection method to four different medical datasets from the UCI Repository of Machine Learning Database and compared its performance with the performance of the similarity classifier without feature removal. The entropy measures by De Luca and Termini (1972) and Parkash et al. (2008) were chosen as entropy measures. Removal of features using this feature selection algorithm led to inproved classifier performance for some data sets, and resulted in a comparable classification result, while using fewer features, for the other data sets (Luukka, 2011).

3.3. Fuzzy similarity and entropy measure (FSAE) feature selection

In this work, an adaptation to the wrapper feature selection algorithm of Luukka (2011) is proposed that is referred to as 'fuzzy similarity and entropy' (FSAE) feature selection. In particular, two changes to the original method are presented: First, a scaling factor for the distance between ideal vectors of classes, and, second, the conceptualization of this adapted method as a feature ranking filter method.

The first proposed adaptation is the scaling factor that emphasizes the distance between the ideal vectors of the classes by scaling the entropy measure for each class and feature. It is suggested that a measure of distance for each feature and class should be used that in its logic is related to similarity. The form of the scaling factor SF can be presented in generalized form as:

$$\mathrm{SF}_{i,d} = 1 - \frac{\left(\sum_{i\neq j}\left|v_{i,d} - v_{j,d}\right|^l\right)^{\frac{1}{l}}}{N-1} \tag{6}$$

The numerator determines the sum of the absolute distances of the ideal vector value for feature d for class i to all other classes (in the most simple case with $l = 1$). In simple terms, the nominator measures how far the mean value of the feature of a class is from the means of all the other classes. To obtain a value within

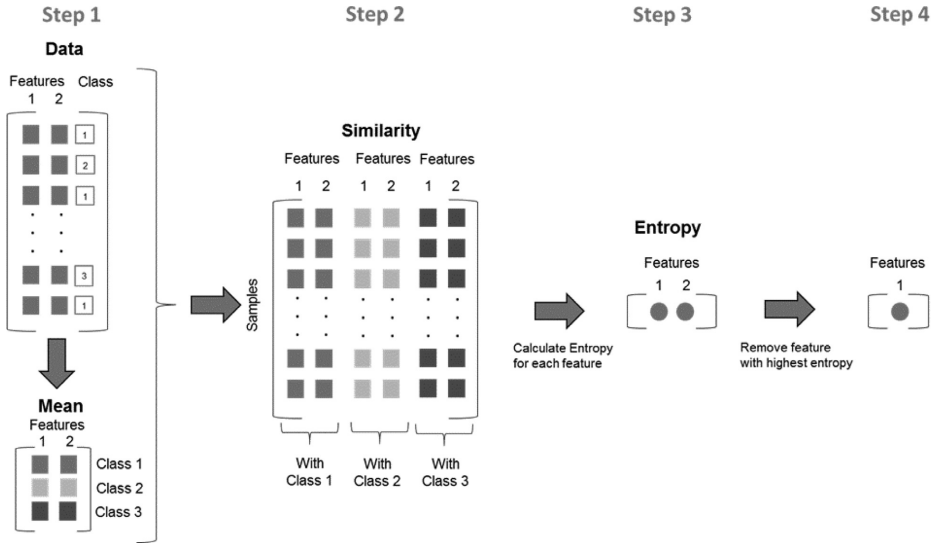


Fig. 1. Feature selection with a similarity classifier and entropy measures.

[0,1], the numerator is divided by *N-1*, where *N* is the number of classes. The denominator is *N-1* since a class is compared to all other classes (*N-1* classes). The distance between two means cannot exceed 1 since the ideal vectors can only take values ∈ [0,1]. Therefore, the quotient is also within [0, 1]. Since a large distance to other ideal vectors is desirable, multiplying the entropy with this quotient will have the opposite of the desired effect. It would lead to higher scaled entropy values for ideal vectors that are far from and, therefore, distinct from other ideal vectors. Thus, the quotient is subtracted from 1, leading to a scale factor within [0,1] but also with the desirable property that distinct ideal vectors for a feature decrease the entropy value for that feature, while entropy values of features where the ideal vectors are close remain at their initial level or are only slightly decreased. In simple terms: if the feature in a class takes on average largely different values than in other classes, then this leads to smaller entropy values. The scaled entropy *SE* for a feature *d* for all classes is:

$$SE_d = \sum_{i=1}^{N} \left(H_{i,d} * \mathrm{SF}_{i,d} \right) \tag{7}$$

The second change from the initial approach of Luukka (2011) is the introduction of this adapted feature selection method as a filter, in particular, a feature ranking method. The use of the scaled entropy values for the feature ranking is straight forward. Small scaled entropy values characterize good features that are informative from information theory point of view and that have ideal vectors that are on average far from another. In contrast to that, large scaled entropy values are found for features that have low informativeness in terms of entropy, and ideal vectors that are on average close to each other. Therefore, a ranking could be implemented directly by sorting the features by their scaled entropy in ascending order, starting from the most important feature with the smallest scaled entropy down to the least relevant feature with the highest scaled entropy. A disadvantage of this approach is that the magnitude of the scaled entropy values per feature can be very high and also difficult to interpret relative to each other in a precise manner. For that reason, scaling the scaled entropy values into the compact interval [0,1] fixes the range of the variable importance values for the variable ranking. In order to obtain an intuitive result, the authors suggest to subtract the standardized scaled entropy values [0,1] from 1. With this scaling, the most relevant feature is set to 1 and the least relevant to 0. Every other feature possesses a value between 0 and 1, which can be regarded as a ratio of how relevant the feature is in comparison to the most important feature. In the authors view, this makes this additional scaling more useful for the interpretation of the results than a simple ranking based on the *SE* values directly.

Obviously, the logic underlying this feature ranking method can also be used in a wrapper function. The implementation of the adapted method as a wrapper is presented since the initial approach was conceptualized as a wrapper function and a comparison is simpler this way. Moreover, it presents a clear framework for those that attempt to improve their classification performance since, as mentioned above, wrapper functions, due to their iterative process, often result in a better performance than the filter counterpart. Also, the question of how many of the features to maintain is directly answered in the wrapper function itself. The overall wrapper feature selection procedure is illustrated in Fig. 2 for a three-class case with two features.

The step by step process for the FSAE wrapper is as follows:

The first step is the division of the data into training and test data, the calculation of the ideal vectors for each class and the test set performance before removal. This is equal to the first step of the original method of Luukka (2011).

The second step of the new feature selection algorithm is exactly the same as in the original one of Luukka (2011). At the end of this step, the similarity of every sample for each feature of all ideal vectors for the classes was determined.

The third step consists of the calculation of the scaling factor for each feature and class, denoted by $\mathrm{SF}_{i,d}$, as defined in Eq. (6),

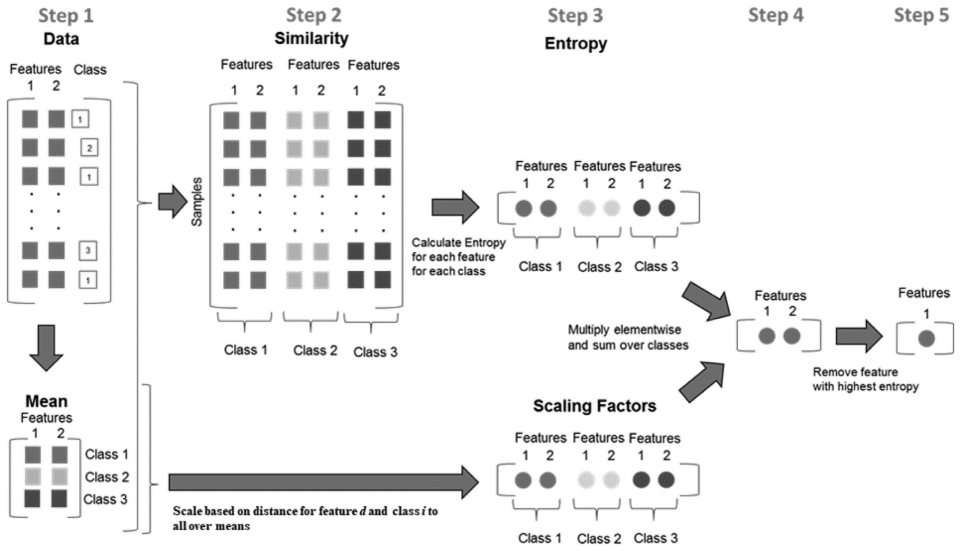


Fig. 2. Adapted feature selection with a similarity classifier and entropy measures.

and the computation of entropy for each feature and class as:

$$H_{i,d} = \sum_{j=1}^{n} H\left(S\left(x_{j,d}, v_{i,d}\right)\right) \tag{8}$$

The difference of Eq. (8) to Eq. (5) is that there is no summation over the classes.

The fourth step is characterized by the calculation of the scaled entropy. The class- and feature-specific entropy and scaling factors are multiplied with each other and summed over the classes as stated in Eq. (7). The result is a scaled entropy value for each of the D features.

The fifth step is equivalent to the fourth step of the original algorithm of Luukka (2011). The feature with the highest (scaled) entropy is removed, the test set performance calculated and compared to the one without the removal in this iteration of the algorithm. All steps are repeated until a stopping criterion such as a performance degradation is met.

3.4. Measure for detecting variation of class means

The objective of the new algorithm for feature selection is to overcome the deficiency of the original method to incorporate the difference of class means into the classification. In order to highlight features with class means that have large (Euclidean) distance between them, a suitable measure for the variation of class means is required. The idea behind such a measure is that class means for features that are far from each other on average also deviate from the overall feature mean. Therefore, a simple measure based on the well-known formula for the standard deviation of a population can be used. The difference in the calculation is that not the deviation of a feature value from its mean is calculated but the deviation from the class mean of a feature from the overall mean of a feature. In simple terms, it measures the standard deviation of the average value that a feature takes for the classes from the overall average of the feature.

$$M_d = \left(\frac{\sum_{i=1}^{N} \left(\bar{x}_{i,d} - \bar{x}_d\right)^2}{N} \right)^{\frac{1}{2}} \tag{9}$$

Where d is a feature, $\bar{x}_{i,d}$ is the mean of feature d for class i, and $\bar{x}_d$ is the mean of feature d. This measure M_d is specific to feature d and takes large values, when the class means for the feature deviate much from the feature mean. Vice versa, low values for M_d point out that the class means do not deviate much from the feature mean, which means that the class means are on average close to one another.

In order to obtain an indication whether for all features on average the class means vary strongly, the mean of the standard deviations can be calculated. The overall formula for this measure, which results in a single number for a dataset, is presented in Eq. (10):

$$M = \frac{\sum_{d=1}^{D} \left(\frac{\sum_{i=1}^{N} \left(\bar{x}_{i,d} - \bar{x}_d\right)^2}{N} \right)^{\frac{1}{2}}}{D} \tag{10}$$

This is essentially the mean of M_d. The idea is that a high value for this measure M indicates that the class means of features in the dataset are on average far from each other, whereas a small value signals that they are close.

Assuming a simplified case where two features have the same or comparable variation of the samples in each class. The feature with the larger standard deviation between the class means as measured by M_d will likely be more discriminative and result on its own in a superior classification accuracy than the other feature alone. When this less discriminative feature, which possesses a low value of M_d is removed from the data set, the average standard deviation of class means for all features, as measured by M, should decrease. This case can be generalized to several features. If one or several features have low values for M_d and the variation of samples in the classes do not differ strongly from other features, fea-

Table 1
Features of the dermatology data set (Ilter & Guvenir, 1998).

Clinical Attributes (values: 0, 1, 2, 3, unless otherwise indicated)	Hispathological Attributes (take values 0, 1, 2, 3)	Class	Instances
1. Erythema	12. Melanin incontinence	Psoriasis	112
2. Scaling	13. Eosinophils in the infiltrate	Seboreic dermatitis	61
3. Definite borders	14. PNL infiltrate	Lichen planus	72
4. Itching	15. Fibrosis of the papillary dermis	Pityriasis rosea	49
5. Koebner phenomenon	16. Exocytosis	Chronic dermatitis	52
6. Polygonal papules	17. Acanthosis	Pityriasis rubra pilaris	20
7. Follicular papules	18. Hyperkeratosis		
8. Oral mucosal involvement	19. Parakeratosis		
9. Knee and elbow involvement	20. Clubbing of the rete ridges		
10. Scalp involvement	21. Elongation of the rete ridges		
11. Family history (0 or 1)	22. Thinning of the suprapapillary epidermis		
34: Age (linear)	23. Spongiform pustule		
	24. Munro microabscess		
	25. Focal hypergranulosis		
	26. Disappearance of the granular layer		
	27. Vacuolation and damage of basal layer		
	28. Spongiosis		
	29. Saw-tooth appearance of retes		
	30. Follicular horn plug		
	31. Perifollicular parakeratosis		
	32. Inflammatory mononuclear infiltrate		
	33. Band-like infiltrate		

ture selection should discard these features, which would improve the classification accuracy. At the same time, this would increase the average standard deviation of class means of a feature to the feature mean, as expressed by measure M. Therefore, effective feature selection will likely remove a feature d or multiple features with a low value of the feature-specific M_d and by doing this, elevate M.

It has to be remarked, that the removal of features depends also to a large extent on variation of data as measured by e.g. standard deviation or with support of an entropy measure. Thus, it is not necessarily true that the features with low M_d will be removed and M improved.

3.5. Data

The data sets that are utilized in this paper are taken from the UCI Repository of Machine Learning Database (Lichman, 2013). Classification is performed for five medical datasets: the dermatology data set (Ilter & Guvenir, 1998), the chronic kidney disease data set (Soundarapandian & Rubini, 2015), the breast cancer Wisconsin (original) data set (Wolberg, 1992), the diabetic retinopathy Debrecen data set (Antal & Hajdu, 2014) and the horse colic data set (McLeish & Cecile, 1989).

3.5.1. Dermatology data set

The dermatology data set (Ilter & Guvenir, 1998) was donated in 1998 and relates to differential diagnosis of erythemato-squamous diseases, which is considered a complicated problem in dermatology. The diseases in this group are: (1) psoriasis, (2) seboreic dermatitis, (3) lichen planus, (4) pityriasis rosea, (5) cronic dermatitis, and (6) pityriasis rubra pilaris. The data set contains 34 explanatory variables, of which 12 are clinical features acquired on the basis of a clinical evaluation and 22 are histopathological features obtained from skin samples. The data set contains six class labels. The features and classes are given in Table 1. The features in the table are referred to as 'Attributes' since this term is used in the original description of this data set. The dataset initially contained 366 observations. Observations for which any of the feature values were missing were removed manually, which narrowed down the number of complete observations for the classification task to 358.

Table 2
Features and class of the chronic kidney disease data set (Soundarapandian & Rubini, 2015).

Attribute and Class	Scale
1. Age (numerical)	age in years
2. Blood pressure (numerical)	in mm/Hg
3. Specific gravity (nominal)	(1.005,1.010,1.015,1.020,1.025)
4. Albumin (nominal)	(0,1,2,3,4,5)
5. Sugar(nominal)	(0,1,2,3,4,5)
6. Red blood cells (nominal)	(normal, abnormal)
7. Pus cell (nominal)	(normal, abnormal)
8. pus cell clumps (nominal)	(present, notpresent)
9. Bacteria (nominal)	(present, notpresent)
10. Blood glucose random (numerical)	in mgs/dl
11. Blood urea (numerical)	in mgs/dl
12. Serum creatinine (numerical)	in mgs/dl
13. Sodium (numerical)	in mEq/L
14. potassium (numerical)	in mEq/L
15. Hemoglobin (numerical)	in gms
16. Packed cell volume (numerical)	–
17. White blood cell count (numerical)	in cells/cumm
18. Red blood cell count (numerical)	in millions/cmm
19. Hypertension (nominal)	(yes, no)
20. Diabetes mellitus (nominal)	(yes, no)
21. Coronary artery disease (nominal)	(yes, no)
22. Appetite (nominal)	(good, poor)
23. Pedal edema (nominal)	(yes, no)
24. Anemia (nominal)	(yes, no)
25. Class (nominal)	(ckd, no ckd)

3.5.2. Chronic kidney disease data set

The chronic kidney data set (Soundarapandian & Rubini, 2015) was donated in 2015. According to UCI Machine Learning Repository, at the time of conducting this research, no papers directly relevant to the subject of our study have been published that use this data set. The dataset contains 24 variables, of which 11 are numeric and 13 are nominal. The classification task for this dataset is binary, with the distinction being between patients with and without chronic kidney disease. The features and the scale of the data are presented in Table 2. The features in the table are also referred to as 'Attributes' since this term is used in the original description of this data set. The dataset initially contained 400 observations. Observations for which any of the feature values were miss-

Table 3
Features and class of the breast cancer Wisconsin (Original) data set (Wolberg, 1992).

Features	Class
1. Sample code number / ID number	2 - benign
2. Clump thickness (from 1 to 10)	4 - malignant
3. Uniformity of cell size (from 1 to 10)	
4. Uniformity of cell shape (from 1 to 10)	
5. Marginal adhesion (from 1 to 10)	
6. Single epithelial cell size (from 1 to 10)	
7. Bare nuclei (from 1 to 10)	
8. Bland chromatin (from 1 to 10)	
9. Normal nucleoli (from 1 to 10)	
10. Mitoses (from 1 to 10)	

Table 4
Features and class of the Diabetic retinopathy Debrecen data set.

Features	Class
1. Quality assessment (0 = bad quality, 1 = sufficient quality)	0 = No sign of DR
2. Pre-screening result (0 = no abnormality, 1 = severe retinal abnormality)	1 = Signs of DR
3.–8. MA detection result (confidence level 0.5 to 1) (numeric)	
9.–16. MA detection result for exudates (confidence level 0.5 to 1) (numeric)	
17. Euclidean distance center of macula to center of optic disc	
18. Diameter of optic disc	
19. AM/FM-based classification {0,1}	

ing were removed manually, which reduced the number of complete observations for the classification task to only 156.

3.5.3. Breast cancer Wisconsin (Original) data set

The breast cancer Wisconsin (Original) data set (Wolberg, 1992) was donated in 1992. It consists of samples on breast cancer patients that were reported periodically. The data set contains 10 variables and a binary class label indicating whether a breast cancer is benign or malignant. The first feature is the ID number while the remaining features represent medical information related to the patient and the cancer. All features and the class labels are enumerated in Table 3. The dataset initially contained 699 observations. Observations for which any of the feature values were missing (indicated by a '?') were removed manually, so that the number of complete observations for the classification task was reduced to 683. The first feature, the ID number, was removed from the data set for the classification since it is not directly related to the cancer and has no meaning for the classification task.

3.5.4. Diabetic retinopathy Debrecen data set

The diabetic retinopathy Debrecen data set (Antal & Hajdu, 2014) encompasses features that were extracted from the Messidor image set for the prediction of diabetic retinopathy from images. Each of the 19 features relates to a detected lesion, a feature depicting the anatomy or is a descriptor of the image level (Antal & Hajdu, 2014). The class label is binary for samples that contain signs of diabetic retinopathy and those that do not contain such signs. All features and the class labels are enumerated in Table 4. The data set contains 1151 observations without any missing values. Moreover, with 611 instances of signs for diabetic retinopathy and 540 without such signs, the classes are quite well balanced.

3.5.5. Horse colic data set

The horse colic data set contains medical features of horses including those that suffered from certain types of lesions (McLeish & Cecile, 1989). The data contained initially 27 features and a binary class label indicating whether a lesion was surgical (retrospectively). All features and the class labels are depicted in Table 5.

Table 5
Features and class of the Horse colic data set.

Features	Class
1. Surgery (1 = yes, 2 = no)	1 = Surgical lesion
2. Age (1 = Adult, 2 = Young (< 6 months))	2 = Not surgical lesion
3. Hospital ID (numeric)	
4. Rectal temperature (numeric)	
5. Pulse (numeric)	
6. Respiratory rate (numeric)	
7. Temperature of extremities (categorical from 1 to 4)	
8. Peripheral pulse (categorical from 1 to 4)	
9. Mucous membranes (categorical from 1 to 6)	
10. Capillary refill time (1 = ' < 3 seconds', 2 = ' ≥ 3 seconds'	
11. Pain level (categorical from 1 to 5)	
12. Peristalsis (categorical from 1 to 4)	
13. Abdominal distension (categorical from 1 to 4)	
14. Nasogastric tube (1 = none, 2 = slight, 3 = significant)	
15. Nasogastric reflux (1 = none, 2 = ' > 1 liter', 3 = ' < 1 liter')	
16. Nasogastric reflux pH (numeric)	
17. Rectal examination (categorical from 1 to 4)	
18. Abdomen (categorical from 1 to 5)	
19. Packed cell volume (numeric)	
20. Total protein (numeric)	
21. Abdominocentesis appearance (1 = clear, 2 = cloudy, 3 = serosanguinous)	
22. Abdomcentesis total protein (numeric)	
23. Outcome (1 = lived, 2 = died, 3 = was euthanized)	
24. Type of Lesion (First Lesion)	
25. Type of Lesion (Second Lesion)	
26. Type of Lesion (Third Lesion)	
27. CP Data (1 = Yes, 2 = No)	

The data set (including the 'horse-colic.test' data set) contains 27 features of 368 observations. The 'Hospital ID' (3) and the 'CP data' (27) were removed since they are not relevant. The features 'Nasogastric reflux PH' (12), 'Abdominocentesis appearance' (21) and 'Abdomcentesis total protein' (22) were removed since the majority of their values was missing. Besides that, horses can have 0 to 3 lesions (as indicated by feature 25 to 27, showing whether a lesion is present). The details of the lesion are presented in form of a digit code indicating its site, type, subtype, and specific code. If a horse had more than 1 lesion (feature 26 and/or 27 were not zeros), the sample was duplicated so that each sample only encompasses one lesion. One sample has the first lesion and the second sample the second lesion, and so on. Furthermore, the digit code of the lesion was split into four categorical features indicating the site (12 categories), the type (4 categories), the subtype (3 categories) and a specification (11 categories). We interpolated missing values by using basic mean /mode interpolation. As a consequence, the final data set contains 23 features and 379 complete observations.

3.6. Feature selection and training procedure

The training and testing during both, the filter and wrapper feature selection is conducted with the holdout method. The first calculations are conducted to compare the wrapper method of Luukka (2011) with the wrapper version of the proposed adapted algorithm termed FSAE. The complete training and testing procedure for the FSAE wrapper is illustrated in Fig. 3. The main idea for the wrapper feature selection is to iteratively remove the feature with the highest entropy value. The procedure is stopped when the test set performance deteriorated by more than 3% compared

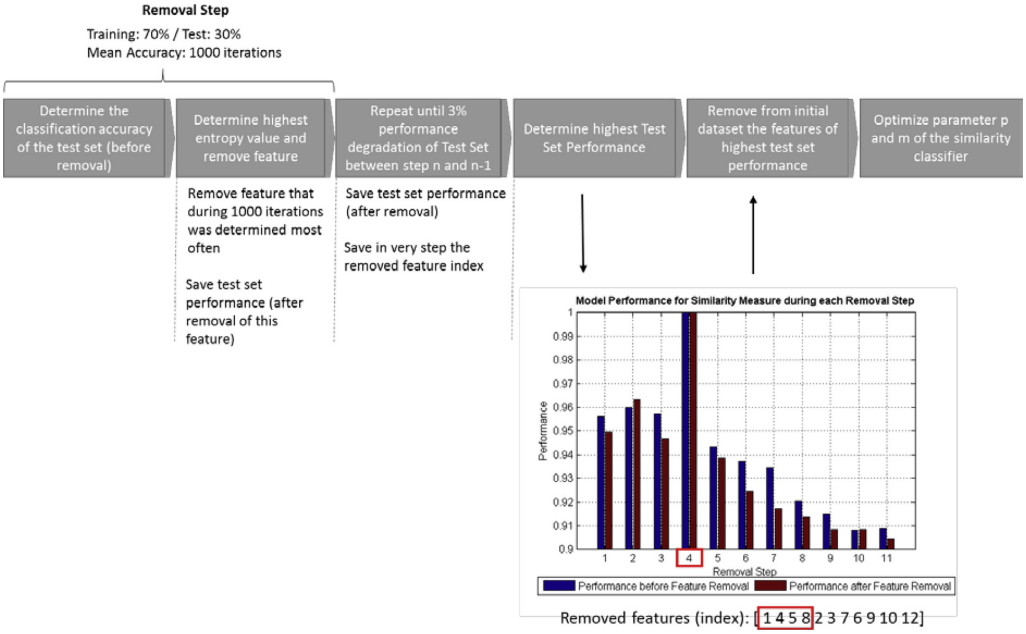


Fig. 3. Training procedure for FSAE Wrapper.

to the previous step. Afterwards, the decision, which features remain, is conducted based on the highest test set performance after all removal steps that occurred before the procedure was stopped and the features removed to reach this performance are noted.

For the supervised filter methods, the FSAE filter is compared to the well-known ReliefF algorithm, the Laplacian Score, the Fisher Score and the feature selection by Luukka (2011) implemented with its logic as a feature ranking method. The ReliefF algorithm is an extension of the Relief algorithm of Kira and Rendell (1992a) that is filter feature selection method suitable for multi-class problems that can even handle noise and incomplete data (Kononenko, Simec, & Robnik-Sikonja, 1997). This algorithm adjusts iteratively its weights, which are initially set to zero, to determine how relevant a feature is (Kononenko et al., 1997; Souza, Matwin, & Japkowicz, 2006). The relevance of a feature is determined by the k Near-hit and Near-miss instances, so the closest observations (based on Euclidean distance) from the same and other classes (Kira & Rendell, 1992b; Kononenko et al., 1997). The contribution of each k instances is averaged and the averaged weights for each feature $\in [-1, 1]$ indicate the relevance of the features (Kononenko et al., 1997). A value that is larger than zero is expected for relevant features and irrelevant features should end up with negative values or values close to zero (Kira & Rendell, 1992b). Now, either a user-specified number of features with the largest weights or all those features larger than a threshold τ can be chosen (Kira & Rendell, 1992b; Souza et al., 2006).

The Laplacian score introduced by He et al. (2005) is a similarity-based unsupervised feature selection method that can also be used as a supervised method when the class labels are available. In the unsupervised version, only the k-nearest neighbours - independent of the class label - are considered whereas in the supervised version only the observations with the same class label are considered for the calculation for the weight matrix S (also called affinity matrix) (He et al., 2005; Li et al., 2017). The value for an element S_{fj} in the weight matrix S for two observations x_f and x_j is:

$$S_{f,j} = e^{\frac{\|x_f - x_j\|^2}{t}} \tag{11}$$

if f and j belong to the same class, and where t is a 'suitable' constant. For all combinations of elements in the weight matrix S that do not belong to the same class, the value for $S_{f,j}$ is set to zero. Subsequently, a diagonal matrix D is calculated where an element $D(f,f)$ is defined as $\sum_{j=1}^{n} S_{f,j}$ and the so-called 'graph Laplacian' is calculated as $L = D - S$ (Li et al., 2017). In order to calculate the Laplacian score, $\tilde{x}_d$ is determined as (He et al., 2005; Li et al., 2017):

$$\tilde{x}_d = x_d - \frac{x_d^t D1}{1^t D1}1 \tag{12}$$

Where x_d is the column vector of feature d (in the reference literature denoted by f_r for the r-th feature), 1 is a column vector of ones, and t denotes the transpose (He et al., 2005). The Laplacian score for a feature d can then be determined as:

$$Laplacian\ Score_d = \frac{\tilde{x}_d^t L \tilde{x}_d}{\tilde{x}_d^t D \tilde{x}_d} \tag{13}$$

Based on this weight matrix, the Laplacian matrix can be computed and the k features with the smallest Laplacian scores shall be retained (Li et al., 2017). For further details on the method, please see He at al. (2005).

The Fisher Score is a similarity-based supervised feature selection method (Duda et al., 2012). It uses the squared distances between the mean of a feature and the mean of the feature in a class, the standard deviations for the feature in the class and the number

Table 6
Artificial example tasks.

Features	Task 1	Task 2	Task 3
Feature 1	All classes mean 50 **with normal noise (0, 0.01)**	All classes mean 50 **with normal noise (0, 0.01)**	Mean 80, 100, 120 with normal noise (0, 10) – **Overlapping**
Feature 2	Mean 10, 50, 100 **with normal noise (0, 0.01)**	Mean 10, 50, 100 **with normal noise (0, 1)**	Mean 10, 20, 30 with normal noise (0, 1) – **Non-Overlapping**

of observations in a class as a weight to determine the importance of a feature. The Fisher score for a feature d can be defined as:

$$Fisher\ Score_d = \frac{\sum_{i=1}^{N} n_i\left(\bar{x}_{i,d} - \bar{x}_d\right)^2}{\sum_{i=1}^{N} n_i \sigma_{i,d}} \tag{14}$$

Where d is a feature, $\bar{x}_{i,d}$ is the mean of feature d for class i, $\bar{x}_d$ is the mean of feature d, $\sigma_{i,d}$ is the standard deviation of feature d in class i, N is the number of classes and n_i is the number of observations in class i. For the Fisher score, the k variables with the largest value of the Fisher score are chosen as feature subset (Li et al., 2017).

The choice of the number of features to retain for the feature ranking methods is based on the threshold for the ReliefF algorithm and the result obtained from the FSAE wrapper method previously. In general, the number of features to keep can be user-specified and the suggested approach is used for simplicity since for ReliefF there are suggested procedures for the choice of a threshold to determine which features to keep. First, all variables with a positive ReliefF weight are chosen for the first feature subset. Second, the features are selected according to the threshold determined by $\tau = \frac{1}{\sqrt{\alpha m}}$ (Robnik-Šikonja & Kononenko, 2003). In particular, the threshold is used with $\alpha = 1$ (not strict) and m equal to the training set size of 70% of the sample points.

The third way is premised on the same number of features that lead to the best mean accuracy of the FSAE wrapper, which will be calculated at that time. The authors want to stress, that these approaches to select the number of features to retain are simply suggestions and another number for k could be chosen. After the removal of features for each of the wrapper and filter techniques, the parameters of the similarity classifier are determined with an optimal value search for the remaining features to accomplish the highest mean accuracy of the given dataset. It is important to note that the performance achieved before the optimal value search is independent of the tuning of the parameters p and m (Luukka, 2008). All calculations were implemented with MATLAB™- software.

4. Results

4.1. Weaknesses of the feature selection algorithm

Three simple classification tasks were constructed to illustrate the weakness in the benchmark feature selection technique as regards accounting for the inter-class distance. Two features were generated for each task as illustrated in Table 6. The corresponding visual illustration of the tasks can be regarded in Fig. 4.

The feature selection and classification results are presented in Table 7. For the first task, one feature was generated that is approximately the same for all classes and contains a small amount of noise (or variation), and one feature with means considerably different in magnitude and the same small level of noise. Obviously, the first feature is not suitable for discrimination between classes, whereas the second feature does not overlap, is precise, and discriminates well between the classes.

When applied to the first task, the feature selection algorithm presented by Luukka (2011) resulted in removal of the second feature, which leads to a significant performance decline. The FSAE algorithm and the other feature ranking methods chose the first, clearly less discriminant feature as the feature to be removed.

When applied to the second task, where the first feature is constructed as in the first task and the second feature is degraded by more noise (or simply possesses more variation), the choice of the feature to be removed remained the same for all algorithms compared to the first task. The result acknowledges that the second feature is still highly discriminatory since feature values of the classes do not overlap.

The third task had one feature with large absolute mean differences and large absolute normal noise and one feature with smaller absolute mean difference and less absolute noise. However, it should be remarked, that for the first feature the difference between class means is only two standard deviations, so the classes overlap to a certain extent. On the other hand, for the second feature the mean values are at least 10 standard deviations from one another so that the features do not overlap for the classes. This makes this second feature more discriminative than the first feature, where the feature values between classes overlap. Therefore, removal of the second feature will still lead to a classification rate of 100%, whereas false removal of the first feature, as found when using the original feature selection technique, will reduce the accuracy to less than 80%.

The results indicate that the FSAE feature selection algorithms, but also the other tested feature ranking methods, more successfully distinguish between highly discriminant and less/non-discriminant features than the approach by Luukka (2011). Despite the inability of the benchmark feature selection algorithm to overcome the scaling issue and account for the effect of the inter-class distance, the algorithm resulted in high classification accuracies in other applications (see Luukka, 2011).

As part of this study, we additionally investigated the effect of different l values up to 10 and found that the removal decision is not altered but the difference between the original algorithm and the proposed scaled version decreases gradually with increase in l. Therefore, only parameter values of $l = 1$ and $l = 2$ are applied in subsequent calculations.

4.2. Dermatology data sets

For the application on the real-world data sets, we initially compare the original wrapper feature selection method of Luukka (2011) with our adapted approach, the FSAE, in its wrapper version. Feature selection and classification results for the dermatology data set are presented in Table 8. Results for the classification appear very good for almost all classifiers and feature selection techniques, ranging from mean accuracy of 96.72% to 98.36%. The most accurate result of 98.36% was obtained with the FSAE wrapper method with $l = 1$, DeLuca & Termini entropy and 29 out of 34 features. The second highest performance was found with the benchmark feature selection method by Luukka (2011) with the entropy measure of Parkash et al. (2008) and 33 features. These two results are the only cases that exceed the performance accuracy of the entire set of features of 98.11%.

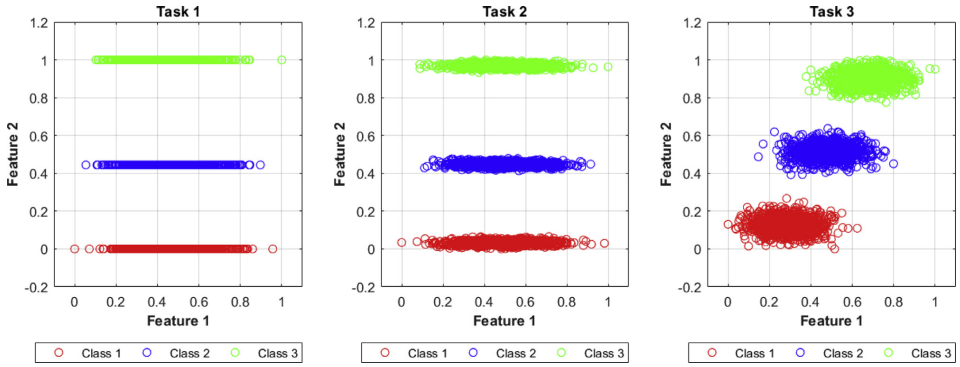


Fig. 4. Artificial example cases.

Table 7
Example cases for weakness of original feature selection.

Task	Classifier	Perf. Before Removal (%)	Entropy / Score Feature 1	Entropy / Score Feature 2	Removal	Perf. After Removal (%)
1	FS (Entropy Parkash et al.)	100.00	428.1	**491.4**	Feature 2	33.92
	FSAE (Entropy Parkash et al., l = 1)	100.00	**1024.5**	430.3	Feature 1	100.00
	FSAE (Entropy Parkash et al., l = 2)	100.00	**1022.8**	622.1	Feature 1	100.00
	Laplacian Score	100.00	**0.0351**	0.999	Feature 1	100.00
	ReliefF	100.00	**0.0001**	0.666	Feature 1	100.00
	Fisher Score	100.00	**0**	1555.4	Feature 1	100.00
2	FS (Entropy Parkash et al.)	100.00	393.6	**562.3**	Feature 2	33.48
	FSAE (Entropy Parkash et al., l = 1)	100.00	**904.3**	532.4	Feature 1	100.00
	FSAE (Entropy Parkash et al., l = 2)	100.00	**907**	740.8	Feature 1	100.00
	Laplacian Score	100.00	**0.0351**	0.999	Feature 1	100.00
	ReliefF	100.00	**0.0024**	0.601	Feature 1	100.00
	Fisher Score	100.00	**0**	14.0716	Feature 1	100.00
3	FS (Entropy Parkash et al.)	100.00	615.7	**686.2**	Feature 2	78.90
	FSAE (Entropy Parkash et al., l = 1)	100.00	**1057.2**	813	Feature 1	100.00
	FSAE (Entropy Parkash et al., l = 2)	100.00	**1152.9**	1022.6	Feature 1	100.00
	Laplacian Score	100.00	**0.7409**	0.9848	Feature 1	100.00
	ReliefF	100.00	**0.0738**	0.4374	Feature 1	100.00
	Fisher Score	100.00	**0.2628**	2.4402	Feature 1	100.00

Table 8
Performance on dermatology data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
Sim		–	34	None	98.11	–	0.0110	0.2	0.1
Sim + FS Luukka (2011)		De Luca and Termini	31	4, 19, 16	96.72	95.02	0.0195	0.7	0.3
Sim + FS Luukka (2011)		Parkash et al.	33	16	98.20	97.17	0.0119	0.2	0.1
Sim + FSAE	l = 1	De Luca and Termini	29	32, 18, 19, 1, 17	98.36	95.73	0.0112	0.3	2
Sim + FSAE	l = 1	Parkash et al.	27	19, 32, 18, 2, 3, 17, 1	98.01	95.28	0.0118	0.5	3
Sim + FSAE	l = 2	De Luca and Termini	30	19, 3, 4, 17	97.03	96.23	0.0198	0.6	0.3
Sim + FSAE	l = 2	Parkash et al.	30	19, 4, 3, 32	97.36	95.28	0.0215	0.2	0.1

The impact of the scaling factor for the improvement of the result from the original classifier compared to the best performing FSAE wrapper method can be regarded in Fig. 5. The figure shows that the features that were removed, as indicated in blue, show in general comparably small values for the measure of M_d. Since their removal resulted in a slight improvement of the classifier, it is probable that the variation of the class means contributed to this effect on the mean accuracy. The value for M, which is the mean over the feature-specific M_d, increased from 0.2079 before the removal to 0.2266 after the removal, showing that on average the difference between class means was elevated. The result of the optimal parameter value search for FSAE wrapper with $l = 1$ and DeLuca & Termini entropy is illustrated in Fig. 6.

Overall, the adapted FSAE feature selection method (with $l = 1$) is equally accurate to the algorithm presented by Luukka (2011) but leads to the removal of more features. In

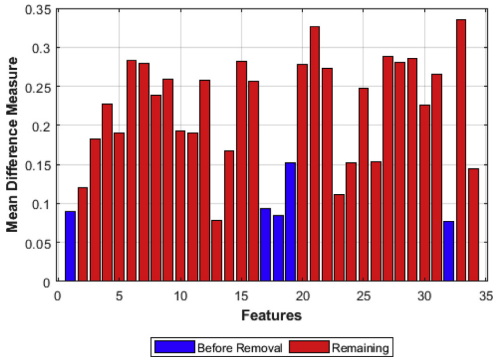


Fig. 5. Feature specific measure M_d for the Dermatology data set.

the next step, the FSAE filter will be deployed and compared with ReliefF, the Laplacian Score, Fisher Score and the method of Luukka (2011) implemented as a feature ranking filter method. First, the ReliefF algorithm will be used with all features that have positive weights as well as with a threshold based on the formula presented above. With the same number of features, 12 and 3 respectively, all feature ranking methods are used. The classification accuracy was determined with a similarity classifier, as used by Luukka (2011). The results are presented in Table 9. The ReliefF algorithm shows with 12 and 3 features the best performance among all filter methods for this data set. Other filter methods only show performances between 85.17% and 83.45% for 12 features and 66.38% and 50.91% for 3 features respectively. It is noticable, that for 12 features, the performances of the Laplacian Score, the Fisher Score, the approach by Luukka (2011) as a filter and the FSAE filter are very close in their mean accuracies. Especially the fact that the approach by Luukka (2011) performes well given its obvious deficiencies that were highlighted above, is remarkable. For only 3 features, the Laplacian Score and the Fisher Score end up with mean accuracies of 55.64% and 50.91%, which are well below the performance of the FSAE filter as well as the filter version of the feature selection by Luukka (2011) of 66.38% and 66.34%.

In order to gain a better understanding of the dependence of the mean accuracies on the number of features removed by each algorithm, the performance was recorded for all possible numbers of features to keep based on the variable importance values determined on the entire data set. The mean performances for the filter methods and standard parameters $p = 1$ and $m = 1$ are displayed in Fig. 7. It is apparent, that on this data set the ReliefF algorithm performs best for most numbers of features. Only initially and for the choice of only a single feature, the performance of ReliefF is lower than that of at least one other filter method. The performance of FSAE, the method by Luukka (2011), the Laplacian score and the Fisher score appears initially very similar, with the Fisher score being for the removal the best approach from these 4 methods. While Luukka (2011) and FSAE drop in performance few features earlier, their drop is not as severe as the one by the Laplacian score and Fisher score. In other words, for the initial removals, Fisher and Laplacian Score perform comparable to FSAE and the filter version of Luukka (2011). In the medium range of removed variables, Fisher score and Laplacian score seem to result in better accuracies than FSAE and the approach by Luukka (2011) but for the choice of only few variables, FSAE and Luukka (2011) are more capable to retain suitable features, with accuracies of above 50% for 11 to 2 features, while the other two approaches in that interval drop well below 40%. The performance for a single feature,

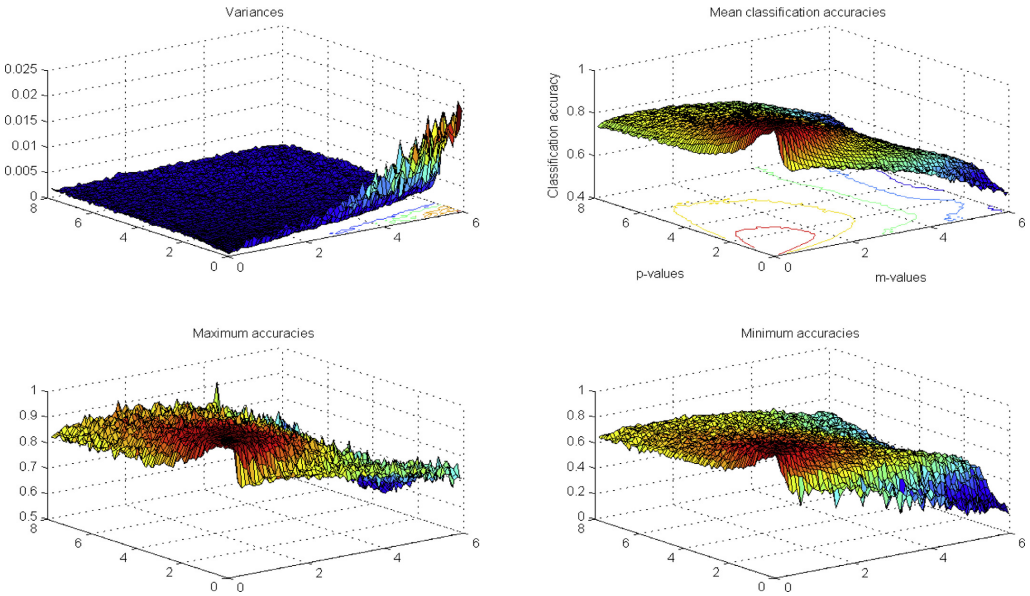


Fig. 6. Accuracies and variance w.r.t. to parameter p and m (dermatology data set).

Table 9
Performance with filter feature selection on dermatology data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
ReliefF	t = 0, k = 10		12	10, 3, 18, 23, 11, 24, 25, 8, 27, 12, 26, 6, 1, 29, 17, 32, 34, 9, 2, 19, 20, 4	95.52	93.91	0.0284	0.2	0.1
ReliefF	t = 0.0632, k = 10		3	10, 3, 18, 23, 11, 24, 25, 8, 27, 12, 26, 6, 1, 29, 17, 32, 34, 9, 2, 19, 20, 4, 30, 33, 21, 16, 31, 7, 13, 28, 5	74.40	66.50	0.0343	0.1	0.1
Laplacian Score	–		12	30, 9, 10, 28, 16, 24, 7, 26, 4, 11, 14, 5, 3, 23, 19, 2, 18, 13, 17, 1, 32, 34	85.15	81.29	0.0107	0.5	0.2
Laplacian Score	–		3	6, 12, 29, 20, 25, 22, 15, 8, 21, 30, 9, 10 28, 16, 24, 7, 26, 4, 11, 14, 5, 3, 23, 19, 2, 18, 13, 17, 1, 32, 34	55.64	38.46	0.0077	1	0.1
Fisher Score	–		12	30, 28, 9, 7, 16, 10, 24, 14, 26, 5, 4, 3, 23, 11, 19, 2, 34, 1, 13, 18, 17, 32	85.17	81.07	0.0098	0.2	0.1
Fisher Score	–		3	12, 33, 25, 31, 20, 8, 15, 22, 21, 30, 28, 9, 7, 16, 10, 24, 14, 26, 5, 4, 3, 23, 11, 19, 2, 34, 1, 13, 18, 17, 32	50.91	41.89	0.0000	0.8	0.1
FS Luukka	–	De Luca and Termini	12	24, 20, 23, 11, 13, 26, 10, 9, 21, 14, 1, 5, 17, 32, 34, 2, 18, 28, 3, 19, 16, 4	83.49	75.05	0.0247	0.8	0.1
FS Luukka	–	Parkash et al.	12	13, 23, 20, 24, 11, 26, 10, 9, 21, 14, 1, 17, 5, 32, 34, 2, 18, 28, 3, 19, 16, 4	83.77	74.86	0.0258	0.8	0.1
FS Luukka		De Luca & Termini, Parkash et al.	3	6, 27, 30, 29, 12, 7, 8, 25, 22, 13, 23, 20, 24, 11, 26, 10, 9, 21, 14, 1, 17, 5, 32, 34, 2, 18, 28, 3, 19, 16, 4	66.34	50.85	0.0183	1	0.1
FSAE	l = 1 & l = 2	De Luca and Termini	12	22, 20, 11, 24, 23, 21, 9, 13, 10, 26, 14, 5, 28, 16, 1, 17, 34, 4, 2, 3, 32, 19, 18	83.45	74.88	0.0241	0.8	0.1
FSAE	l = 1 & l = 2	Parkash et al.	12	20, 11, 24, 23, 13, 21, 9, 10, 26, 14, 5, 28, 1, 16, 17, 34, 2, 32, 4, 3, 18, 19	83.89	75.06	0.0244	0.8	0.1
FSAE	l = 1 & l = 2	De Luca & Termini, Parkash et al.	3	6, 27, 29, 30, 12, 8, 7, 25, 22, 20, 11, 24, 23, 21, 9, 13, 10, 26, 14, 5, 28, 16, 1, 17, 34, 4, 2, 3, 32, 19, 18	66.38	50.94	0.0245	1	0.1

however, is best for the Fisher score, the second best by FSAE and ReliefF, and third by Luukka (2011) and Laplacian Score.

Comparing the filter results with the wrapper performances, it is apparent that the FSAE wrapper performs better than the filter methods, even better than the ReliefF algorithm with 12 and 3 suggested removals respectively. However, FSAE wrapper requires more features than ReliefF for the higher mean accuracy.

4.3. Chronic kidney disease data set

Feature selection and classification results for the chronic kidney data set are presented in Table 10. The highest mean accuracy was obtained with the FSAE feature selection method with $l = 2$ and DeLuca & Termini entropy and only 4 out of 24 features. In most cases, the FSAE methods led to only a small degradation in classification performance for the removal of 12 to 21 features. Moreover, the algorithms achieved accuracies between 99.42–100.00%. In contrast, feature selection using the benchmark feature selection approach by Luukka (2011) achieved accuracies between 96.92–100.00%. Besides that, the highest performance of 100.00% accuracy for the benchmark feature selection method of Luukka (2011) was achieved with 18 features, while the FSAE (with $l = 2$ and DeLuca & Termini entropy) needed only 4 features and, with the same entropy measure, produced an equivalent result. Us-

Table 10
Performance on chronic kidney disease data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
Sim		–	24	None	99.90	–	0	0.5	0.1
Sim + FS Luukka (2011)		De Luca and Termini	18	3, 18, 13, 15, 16, 10	100.00	97.83	0	0.4	0.1
Sim + FS Luukka (2011)		Parkash et al.	5	3, 18, 13, 16, 15, 1, 4, 8, 20, 24, 12, 7, 11, 6, 21, 2, 10, 22, 9	96.92	97.83	0.0355	5	1.4
Sim + FSAE	$l=1$	De Luca and Termini	12	1, 2, 13, 16, 15, 10, 3, 18, 17, 11, 12, 23	99.84	95.65	0	0.5	0.1
Sim + FSAE	$l=1$	Parkash et al.	3	2, 1, 3, 13, 16, 18, 15, 12, 17, 10, 5, 9, 11, 24, 8, 23, 21, 22, 6, 20, 19	99.42	97.83	0.0123	0.4	0.1
Sim + FSAE	$l=2$	De Luca and Termini	4	1, 13, 2, 16, 3, 11, 18, 15, 17, 10, 12, 6, 23, 8, 22, 9, 21, 24, 5, 14	100.00	95.72	0	0.9	0.1
Sim + FSAE	$l=2$	Parkash et al.	12	13, 2, 1, 16, 3, 18, 15, 12, 11, 17, 8, 6	99.98	97.83	0.0004	5.9	0.1

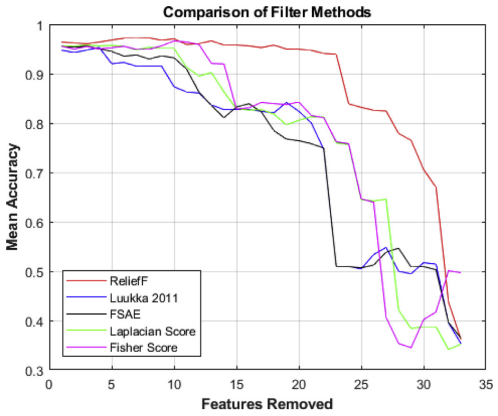


Fig. 7. Comparison of filter methods (dermatology data set).

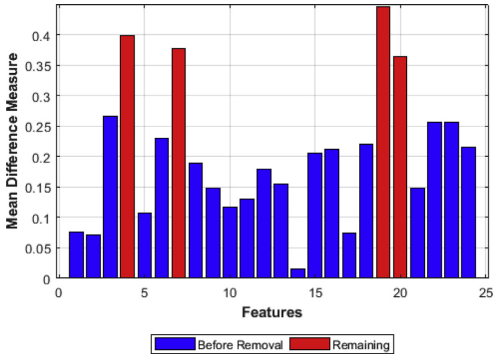


Fig. 8. Feature specific measure M_d for the Kidney data set.

ing exhaustive search to determine if it is possible to reach 100.00% accuracy with any combination of fewer features, showed, that 3 features can be enough to reach the best mean accuracy in the test set. Therefore, the result of the FSAE in the setup mentioned above was fairly close to the optimal result of 100.00% accuracy with 3 features.

The values of the feature specific measure M_d for the best performing FSAE wrapper are illustrated in Fig. 8. The figure highlights that all features despite the four features with the clearly highest M_d values were removed. The performance after the removal of 20 out of 24 features reached 100.00%, which is an indication that all removed features were irrelevant for the classification task. Since all features with small and medium values for M_d were removed, it is unsurprising that the value for M strongly increased from 0.2024 to 0.3965, highlighting a larger average difference of class means. The fact that comparably many features of the Kidney data set could be removed, indicates that the difference in means is highly relevant for the discrimination of the classes for this data set. The result of the optimal parameter value search for this algorithm and DeLuca & Termini entropy is illustrated in Fig. 9. It is noteworthy that a high mean accuracy can be achieved with several variations of the p and m parameters.

The results for all discussed filter methods on the chronic kidney disease data are presented in Table 11. The ReliefF algorithm suggests 12 and 17 features respectively, which results in both cases in an accuracy of 100.00%. All other filter methods also result in 100.00% accuracies with 12 features. Since with the FSAE wrapper, a mean accuracy of 100.00% could be achieved with only 4 features, this number of ranked features is also tested with all filter methods. With 4 features, ReliefF, Laplacian score and Fisher score result in 100.00% accuracy and the FSAE filter in 99.50% (for both entropies and $l=1$ and $l=2$). Only the performance of Luukka (2011) deteriorates considerably with 4 features compared to 12. The mean accuracy for this approach decreases to 97.25% and 95.69% respectively.

The mean performances for the filter methods and standard parameters $p=1$ and $m=1$ are displayed in Fig. 10. The comparison of the filter methods shows that ReliefF performs again very well from the beginning up to the removal of 19 features. Between the removal of 1 to 19 features, the performance of the Laplacian score and FSAE is largely comparable, whereas the approach of Luukka (2011) first underperforms, then outperforms and then clearly underperforms the performance of these two filter

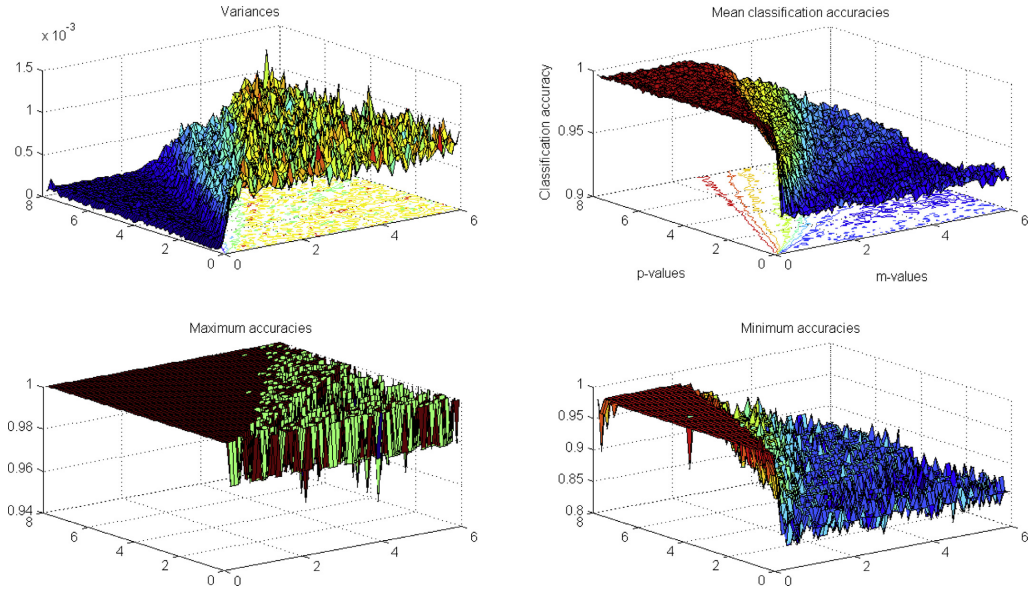


Fig. 9. Accuracies and variance w.r.t. to parameter p and m (chronic kidney disease data set).

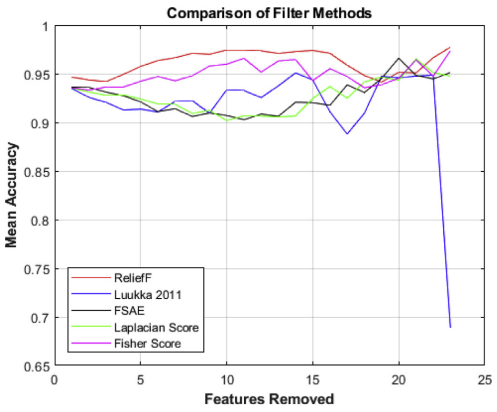


Fig. 10. Comparison of filter methods (kidney data set).

methods. For the standard parameters, FSAE shows the highest mean performance for 4 out of 24 features. In general, when 5 to 2 features remain, all filter methods perform comparably. Towards the end, ReliefF and the Fisher score perform slightly better than the other filter methods. At the same time, the approach by Luukka (2011) leads to a considerable decrease in the classification performance to less than 70% with a single feature whereas the remaining filter approaches still result in a mean accuracy of about and more than 95%.

In comparison with the FSAE wrapper, all filter methods can also lead to a feature subset selection of 4 features that are capable to reach a mean accuracy of 100%. However, it is for the feature ranking methods not obvious that 4 features are sufficient to reach this performance. For ReliefF, 12 and 17 features are suggested. Clearly, this number of features also leads to an accuracy of 100.00% but it is three or even more than four times the number of features that are sufficient to end up with the same result. For this data set, the FSAE wrapper as well as the FSAE filter show competitive results to their benchmark algorithms.

4.4. Breast cancer Wisconsin (Original) data set

The feature selection and classification results for the breast cancer Wisconsin data set are presented in Table 12. Without any feature selection, a performance of 97.61% can be accomplished. The wrapper approach from Luukka (2011) with both entropies indicates that no feature removal should be conducted. Opposed to that, the FSAE wrapper for both entropies and $l = 1$ as well as $l = 2$ leads to the removal of four features. With 5 out of 9 features, still a performance of over 97% can be achieved, which indicates that these features were not important for the classification.

The values of the feature specific measure M_d for the best performing FSAE wrapper feature selection method are illustrated in Fig. 11. The figure shows that all features despite the four features with the highest M_d values and one feature with the lowest value were removed. The performance after the removal of 4 out of 9 features reached with 97.30% a comparable performance than the 97.61% without feature selection. Since all removed features have lower values of M_d than 4 out of 5 of the remaining features, it is unsurprising that the value for M increased from 0.2475 to 0.2649. The fact that four out of 5 of the remaining features are the ones with the largest difference in class means indicates that the difference in means appears to be relevant to discriminate between the classes in the Breast cancer Wisconsin data set. The result of

Table 11
Performance with filter feature selection on chronic kidney disease data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
ReliefF	t = 0, k = 10		17	23, 1, 14, 24, 21, 5, 11	100.00	96.85	0	0.1	0.1
ReliefF	t = 0.0957, k = 10		12	23, 1, 14, 24, 21, 5, 11, 2, 12, 9, 10, 13	100.00	97.42	0	0.1	0.1
ReliefF	k = 10		4	7, 3, 16, 15, 17, 22, 18, 8, 13, 10, 9, 12, 2, 11, 5 21, 24, 14, 1, 23	100.00	95.04	0	1	0.1
Laplacian Score	–		12	12, 14, 11, 10, 3, 16, 15, 18, 1, 13, 17, 2	100.00	90.69	0	1	0.1
Laplacian Score	–		4	20, 24, 22, 23, 6, 9, 8, 21, 5, 12, 14, 11, 10, 3, 16, 15, 18, 1, 13, 17, 2	100.00	94.17	0	0.9	0.1
Fisher Score	–		12	18, 8, 11, 9, 21, 13, 5, 10, 17, 1, 2, 14	100.00	95.50	0	1	0.1
Fisher Score	–		4	22, 23, 3, 16, 6, 15, 24, 12, 18, 8, 11, 9, 21, 13, 5, 10, 17, 1, 2, 14	100.00	94.67	0	1	0.1
FS Luukka (2011)		De Luca and Termini	12	22, 23, 10, 11, 2, 12, 1, 13, 15, 16, 18, 3	100.00	90.46	0	1	0.1
FS Luukka (2011)		De Luca and Termini	4	21, 7, 17, 8, 20, 4, 24, 6, 22, 23, 10, 11, 2, 12, 1, 13, 15, 16, 18, 3	97.25	94.60	0.0315	6	1.2
FS Luukka (2011)		Parkash et al.	12	2, 24, 6, 12, 1, 22, 23, 13, 15, 16, 18, 3	100.00	92.40	0	0.8	0.1
FS Luukka (2011)		Parkash et al.	4	9, 21, 10, 7, 4, 11, 8, 20, 2, 24, 6, 12, 1, 22, 23, 13, 15, 16, 18, 3	95.69	94.69	0.0572	7.7	4.9
FSAE	l = 1 & l = 2	De Luca and Termini	12	5, 12, 3, 11, 17, 10, 16, 15, 18, 13, 2, 1	100.00	91.38	0	1	0.1
FSAE	l = 1 & l = 2	Parkash et al.	12	8, 17, 12, 10, 11, 3, 15, 16, 18, 13, 2, 1	100.00	90.25	0	1	0.1
FSAE	l = 1 & l = 2	De Luca & Termini, Parkash et al.	4	20, 22, 23, 5, 6, 9, 21, 24, 8, 17, 12, 10, 11, 3, 15, 16, 18, 13, 2, 1	99.50	96.56	0.0080	1.3	0.1

Table 12
Performance on breast cancer Wisconsin (original) data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
Sim		–	9	None	97.61	94.55	0.0070	0.3	0.8
Sim + FS Luukka (2011)		De Luca & Termini Parkash et al.	9	None	97.64	94.62	0.0082	0.6	0.3
Sim + FSAE	l = 1 & l = 2	De Luca & Termini Parkast et al.	5	7, 1, 5, 4	97.30	94.93	0.0072	0.6	0.2

Table 13
Performance with filter feature selection on breast cancer data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
ReliefF	t = 0.0457, k = 10		7	7, 9	97.70	95.40	0.0076	1	0.2
ReliefF	k = 10		5	5, 2, 7, 9	97.59	94.73	0.0071	1.3	0.2
Laplacian Score	–		7	1, 9	97.39	95.18	0.0063	0.7	0.2
Laplacian Score	–		5	7, 5, 1, 9	97.05	94.68	0.0097	0.5	0.4
Fisher Score	–		7	5, 9	97.61	95.12	0.0097	0.8	0.2
Fisher Score	–		5	7, 1, 5, 9	96.95	94.77	0.0074	0.6	0.3
FS Luukka (2011)	–	De Luca & Termini, Parkash et al.	7	7, 1	97.14	94.33	0.0133	0.8	0.2
FS Luukka (2011)		De Luca & Termini, Parkash et al.	5	4, 3, 7, 1	97.33	93.36	0.0101	0.8	0.2
FSAE	l = 1 & l = 2	De Luca & Termini, Parkash et al.	7	1, 7	97.25	94.34	0.0077	0.7	0.2
FSAE	l = 1 & l = 2	De Luca & Termini, Parkash et al.	5	4, 5, 1, 7	97.40	94.94	0.0085	0.5	0.2

Table 14
Performance on diabetic retinopathy Debrecen data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
Sim		–	19	–	59.57	–	0.0647	0.2	0.1
Sim + FS Luukka (2011)		De Luca and Termini	12	19, 7, 6, 5, 8, 4, 2	61.05	58.22	0.0398	2.0	1.5
Sim + FS Luukka (2011)		Parkash et al.	15	19, 7, 6, 5	60.53	59.49	0.0677	0.3	0.1
Sim + FSAE	l = 1 & l = 2	De Luca & Termini, Parkast et al.	14	19, 7, 6, 5, 8	61.09	60.22	0.0419	0.8	0.2

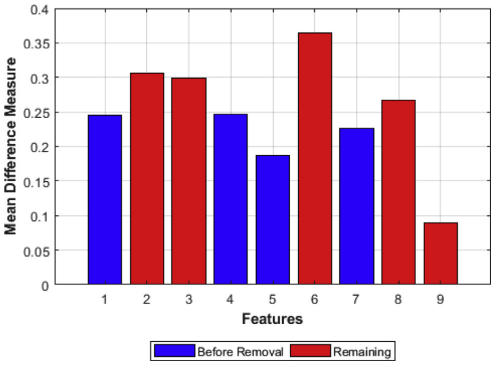


Fig. 11. Feature specific measure M_d for the Breast Cancer data set.

the optimal parameter value search for FSAE wrapper and Parkash et al. entropy is illustrated in Fig. 12.

Subsequently, the results for the filter methods on the breast cancer Wisconsin data are presented in Table 13. For ReliefF, all features possess positive weights, which makes the approach of using only the features with weights larger zero equivalent to conducting no feature selection. In contrast to that, using the formula for the threshold for the ReliefF algorithm suggests using 7 features. In addition, the knowledge from the previous comparison of wrappers indicates that 5 features can result in a performance comparable to using all features. The performances for all filter methods with 7 and 5 features are very close to each other, only ranging from 96.95% to 97.70%. There appears to be no clear difference in the performance for these two numbers of features among the filter methods.

The mean accuracies for the filter methods with standard parameters $p = 1$ and $m = 1$ are displayed in Fig. 13. Three aspects of these results are remarkable. First, the ReliefF, Laplacian score, Fisher score and FSAE perform comparable over the entire range of removals. Second, FSAE but also the Laplacian score and Fisher score perform at least as good as ReliefF. For only 3 remaining features (6 removals) they even perform about 1% better than ReliefF. It is apparent that FSAE and the Laplacian score peform best for this data set. The third point is that the approach by Luukka (2011) starts right from the beginning to continuously increase the difference in performance to the remaining 4 filter approaches. Until the removal of 7 out of 9 features, there is already a difference of more than 3% in the mean performance to the other filters but with only a single feature the approach by Luukka (2011) shows a performance of around 79% whereas the remaining approaches achieve a mean accuracy of more than 90%.

Finally, the results from the FSAE wrapper and the filter methods are comparable in terms of mean accuracy and variance. Once more, the main consideration is that the FSAE wrapper incorporates the choice of the number of features whereas the filter ranking methods do not – and ReliefF suggest to retain more features than the FSAE wrapper.

4.4.1. Diabetic retinopathy Debrecen data set

The results for classification with and without feature selection for the diabetic retinopathy Debrecen data set are presented in Table 14. The mean accuracy with all features is the lowest, indicating that all feature selection techniques were capable to remove irrelevant features and improve the performance. The two best results were accomplished with the wrapper of Luukka (2011) and the FSAE wrapper (for both entropies and l parameter values). The feature selection of Luukka (2011) uses between 12 and 15 features for the classification whereas the result of FSAE is more stable with 14 features for all setups. Besides that, the wrapper of Luukka (2011) requires the smallest amount of features to achieve the second highest result. However, at the same time, it has the lowest result before the optimal value search for the p and m parameter.

The values of the feature specific measure M_d for the best performing FSAE wrapper are displayed in Fig. 14. The figure shows that the features that were removed had medium to large values for M_d. Besides that, the two largest M_d values of the third and fourth feature were not removed, indicating that at least in these cases the variation in the means were relevant for the classification. Given that the performance after the removal of 5 features improved by about 1.5% points, the variation in means for these features was irrelevant for the classification. Their irrelevance might be on account of larger variation in these features. This data set embodies the only case in this research, where the value for M decreased after feature selection from 0.0182 to 0.0133. It should be noted that the variation in the means M_d is in this data set considerably lower than in the previous data sets, by a factor of more than 10. Therefore, other factors such as variance could have impacted the discriminative ability of the features more easily. The result of the optimal parameter value search for FSAE and DeLuca & Termini entropy is illustrated in Fig. 15.

The results for the filter methods on the diabetic retinopathy Debrecen data are presented in Table 15. For ReliefF, all features possess positive weights. Therefore, using the rule to select for the feature subset only the features with weights larger zero is equivalent to conducting no feature selection at all. Moreover, using the threshold formula depicted before, suggests removing all features, which is also not suitable for feature selection. As a consequence, only the performance on 14 features, as suggested by the FSAE wrapper, will be compared. The FSAE filter method for both entropies and l parameters results in the same feature removal and the highest mean performance. Moreover, using the similarity classifier only with standard parameters on the feature subset

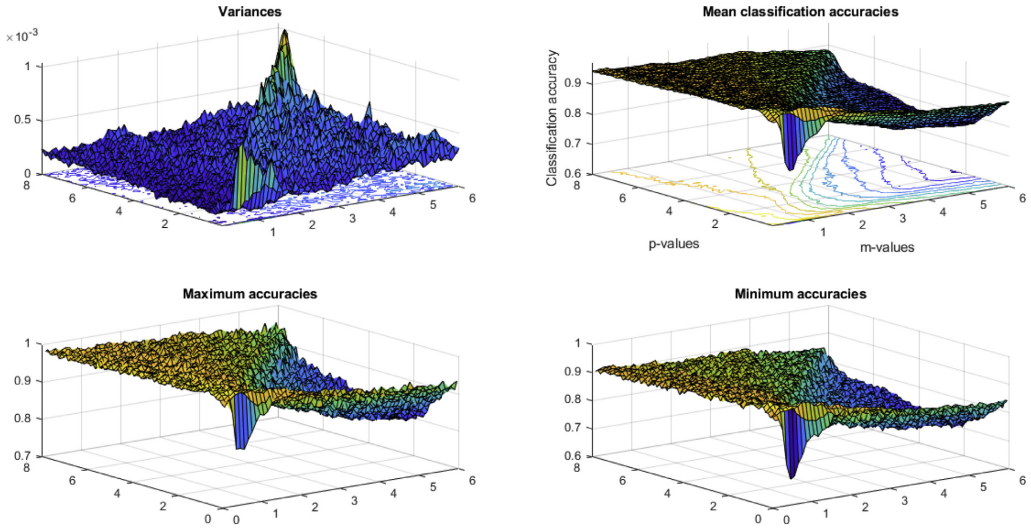


Fig. 12. Accuracies and variance w.r.t. parameter p and m (breast cancer data set).

Table 15
Performance with filter feature selection on diabetic retinopathy Debrecen data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
ReliefF	k = 10		14	13, 18, 2, 1, 17	59.71	57.07	0.0466	0.1	0.7
Laplacian Score	–		14	16, 13, 17, 18, 1	59.39	56.92	0.0563	0.2	0.1
Fisher Score	–		14	1, 11, 18, 17, 10	59.64	56.90	0.0438	0.2	0.1
FS Luukka (2011)		De Luca & Termini, Parkash et al.	14	4, 5, 6, 7, 19	59.51	56.46	0.0418	0.2	0.5
FSAE	l = 1 & l = 2	De Luca & Termini, Parkash et al.	14	8, 5, 6, 7, 19	61.06	60.11	0.0470	0.5	0.2

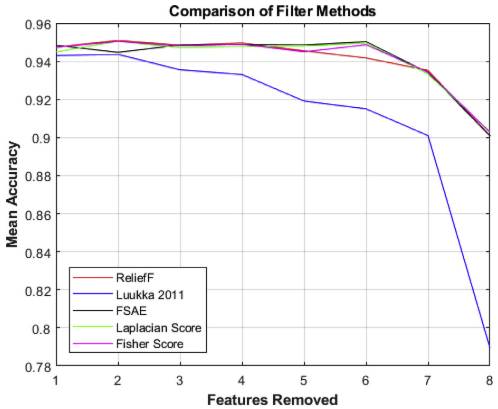


Fig. 13. Comparison of filter methods (breast cancer data set).

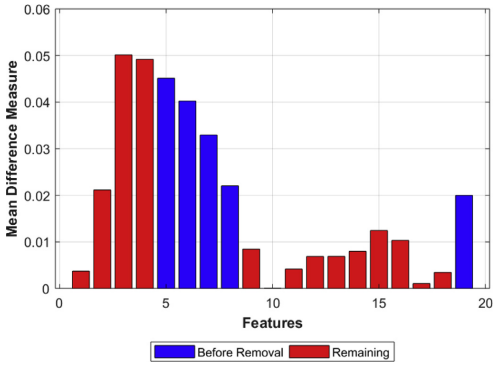


Fig. 14. Feature specific measure M_d for the Diabetic retinopathy Debrecen data set.

obtained by the FSAE still outperforms the remaining approaches after the optimal value search for the p and m parameter.

The mean accuracies for the filter methods with standard parameters $p = 1$ and $m = 1$ are illustrated in Fig. 16. The comparison highlights, that the FSAE filter performs best for the removal of up to 5 features. It shows with over 60% mean accuracy for the

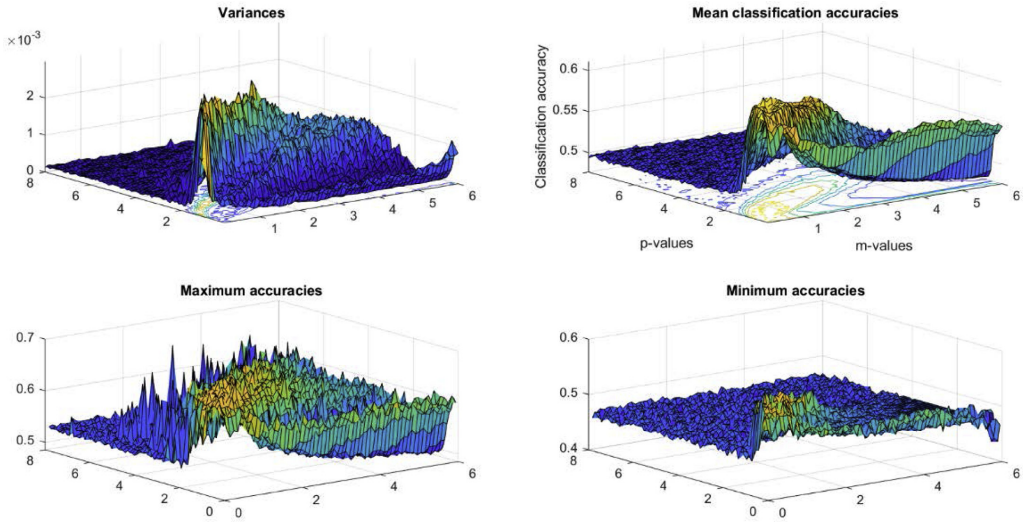


Fig. 15. Accuracies and variance w.r.t. parameter p and m (diabetic retinopathy Debrecen data set).

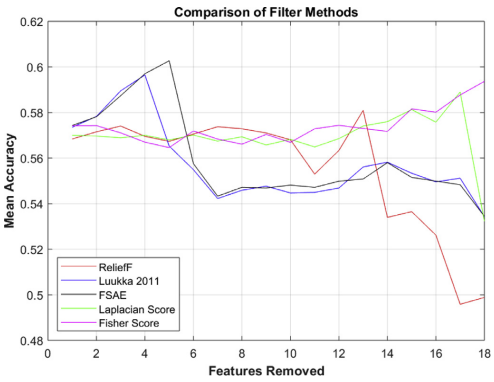


Fig. 16. Comparison of filter methods (Diabetic retinopathy Debrecen data set).

standard parameters the highest performance for all approaches and removal decisions overall. Between 6 to 13 features, ReliefF, Laplacian score and Fisher score show about 2% higher mean accuracies. However, after 13 removed features, ReliefF experiences a strong decline in performance, which makes it the worst approach for the removal of the last features. Towards, the end, the Fisher score performs best while FSAE, Laplacian score and the approach of Luukka (2011) demonstrate comparable performance.

In comparison with the FSAE wrapper, the FSAE filter ends up with the same feature subset and performance. For the diabetic retinopathy Debrecen data set overall, the FSAE wrapper demonstrates good and stable results, whereas the FSAE filter obtains the best results compared to ReliefF, Laplacian score, Fisher score and Luukka (2011).

4.5. Horse colic data set

The feature selection and classification results for the Horse colic data set are presented in Table 16. Once more, both feature selection methods demonstrate an improvement of the mean classification accuracy after feature removal. The FSAE wrapper together with the approach by Luukka (2011) eventuate in the highest mean accuracies of 87.70%, which is an improvement of about 1% compared to no feature selection. The feature subset selected by the FSAE wrapper is preferable to that of Luukka (2011) since it is with 12 features smaller and also leads already to a higher mean accuracy before optimal value search. Once more, the result of the FSAE is more stable since all setups of entropy and l parameter suggest the same feature subset. Opposed to that, the approach by Luukka (2011) selects a different number of features – both higher than those of the FSAE wrapper.

The feature specific measures M_d for the best performing FSAE wrapper are presented in Fig. 17. The results show that the eleven removed features show only small to medium values for the variation of the means. The features with the three largest, as well as numerous of the medium to large values for M_d, are not removed from the feature set. This indicates that the difference in mean values is relevant for the classification. Furthermore, the removal led to an increase in the value for M from 0.0753 to 0.0940. The performances of the optimal parameter value search for the FSAE wrapper and DeLuca & Termini entropy is illustrated in Fig. 18.

The performances of the filter methods on the horse colic data set are depicted in Table 17. The performances for all approaches are tested for 21 features, as suggested by all positive ReliefF weights, with 14 features, as indicated with the FSAE wrapper, and with 6 features based on the threshold formula for ReliefF. The mean accuracies with 21 features are highly comparable among all approaches, ranging from 86.77% to 87.37%. With 12 features, the FSAE filter with Parkash et al. entropy is the best performing approach with a mean accuracy of 89.05%. It even demonstrates the highest mean accuracy of all options and number of features. On top of that, this performance for 12 features is higher than that of

Table 16
Performance Horse colic data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance	Before Optimization	Variance (in %)	p	m
Sim		–	23	–	86.72	–	0.0576	0.7	0.4
Sim + FS Luukka (2011)		De Luca and Termini	18	12, 7, 19, 8, 6	87.41	84.38	0.0821	0.6	0.5
Sim + FS Luukka (2011)		Parkash et al.	16	15, 19, 7, 18, 6, 12, 8	87.70	83.40	0.0578	1	0.5
Sim + FSAE	l = 1 & l = 2	De Luca & Termini, Parkash et al.	12	18, 6, 19, 8, 15, 7, 22, 9, 12, 13, 14	87.70	85.16	0.0667	0.7	0.4

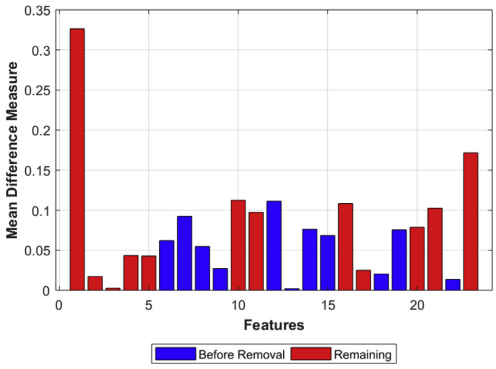


Fig. 17. Feature specific measure M_d for the Horse colic data set.

the best performing wrapper approach. In contrast to that, the approach of Luukka (2011) is characterized by the worst performance for 12 features and also overall. Finally, with 6 features, the FSAE is not performing as strong as ReliefF, Laplacian score and Fisher score but is still clearly outperforming Luukka (2011).

The mean accuracies for the filter methods with standard parameters $p = 1$ and $m = 1$ are diagrammed in Fig. 19. All filter methods perform for the removal of up to 10 features without any considerable decline in performance. Initially, the approach by Luukka (2011) performs best but subsequently is ranked only second after FSAE, which is up to the removal of 14 features performing better than the Fisher score, Laplacian score and ReliefF. The difference to ReliefF are in this range even several % points more accurate. However, towards the removal of the last features, these approaches again perform better. The method of Luukka (2011) experiences a clear performance drop after the removal of 11 features and continues to decline in mean accuracy for most subsequent removals. With 10 out of 23 features it is already about 15% points less accurate than all other feature selection methods, whereas at 1 remaining feature its performance is already approximately 30% points worse.

Overall, the FSAE approaches achieve on the horse colic data set very good results. The FSAE wrapper accomplishes the same performance than the wrapper of Luukka (2011) with 4 features less. At the same time, the FSAE filter achieves the highest mean accuracy of all approaches, including the wrapper approaches, with 12 out of 23 features.

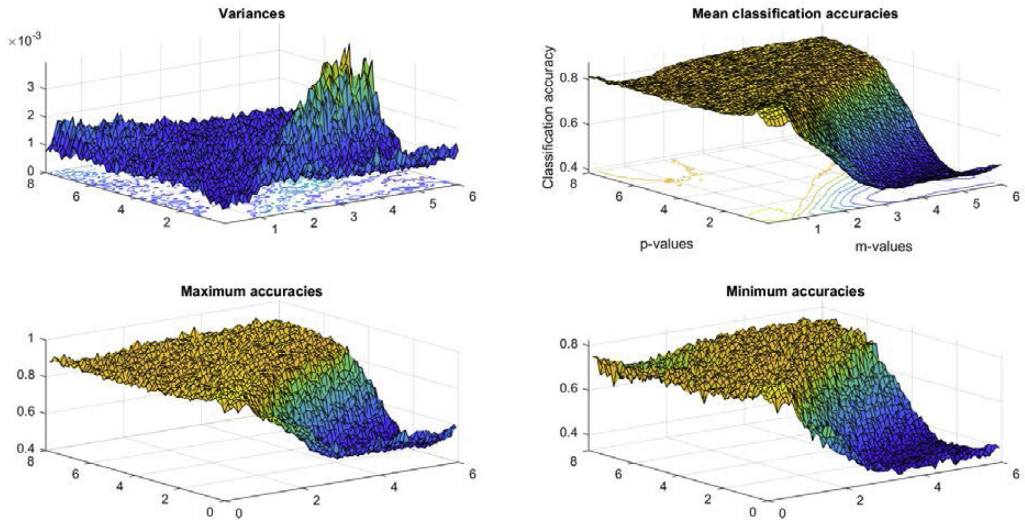


Fig. 18. Accuracies and variance w.r.t. parameter p and m (Horse colic data set).

Table 17
Performance filter feature selection Horse colic data set.

Approach	Parameters	Entropy	No. Features	Features Removed (not ordered)	Avg. Performance (%)	Before Optimization (%)	Variance (in %)	p	m
ReliefF	t = 0, k = 10		21	6, 9	86.79	81.02	0.0643	0.8	0.2
ReliefF	k = 10		12	15, 11, 5, 13, 8, 4, 2, 17, 3, 6, 9	88.25	81.44	0.0700	0.9	0.5
ReliefF	t = 0.0614, k = 10		6	18, 12, 7, 16, 14, 10, 15, 11, 5, 13, 8, 4, 2, 17, 3, 6, 9	88.24	85.10	0.0760	0.5	0.2
Laplacian Score	–		21	13, 3	86.77	79.95	0.0741	0.9	0.5
Laplacian Score	–		12	20, 4, 6, 2, 9, 18, 17, 22, 5, 13, 3	86.57	81.20	0.0838	0.8	0.4
Laplacian Score	–		6	19, 15, 11, 10, 14, 8, 20, 4, 6, 2, 9, 18, 17, 22, 5, 13, 3	87.97	82.44	0.0610	0.5	0.2
Fisher Score	–		21	3, 13	86.79	80.10	0.0649	1.1	0.4
Fisher Score	–		12	6, 4, 5, 8, 17, 9, 18, 2, 22, 3, 13	87.52	82.47	0.0561	0.6	0.5
Fisher Score	–		6	20, 11, 7, 14, 19, 15, 6, 4, 5, 8, 17, 9, 18, 2, 22, 3, 13	87.43	84.19	0.0689	0.7	0.3
FS Luukka (2011)		De Luca & Termini, Parkash et al.	21	12, 7	87.37	82.88	0.0538	0.3	0.5
FS Luukka (2011)		De Luca & Termini, Parkash et al.	12	23, 1, 10, 16, 8, 18, 15, 6, 19, 12, 7	81.67	68.21	0.0598	0.1	0.1
FS Luukka (2011)		De Luca & Termini, Parkash et al.	6	13, 21, 11, 22, 9, 14, 23, 1, 10, 16, 8, 18, 15, 6, 19, 12, 7	82.15	63.86	0.0525	0.4	0.2
FSAE	l = 1 & l = 2	De Luca & Termini, Parkash et al.	21	6, 18	86.86	80.35	0.0603	0.8	0.4
FSAE	l = 1 & l = 2	De Luca and Termini,	12	14, 13, 12, 9, 22, 7, 15, 8, 19, 6, 18	87.40	85.15	0.0691	0.6	0.1
FSAE	l = 1 & l = 2	Parkash et al.	12	16, 13, 9, 22, 12, 15, 7, 8, 19, 6, 18	89.05	82.31	0.0746	0.6	0.2
FSAE	l = 1 & l = 2	De Luca & Termini, Parkash et al.	6	23, 4, 21, 11, 10, 16, 14, 13, 12, 9, 22, 7, 15, 8, 19, 6, 18	84.20	81.75	0.0828	0.3	0.1

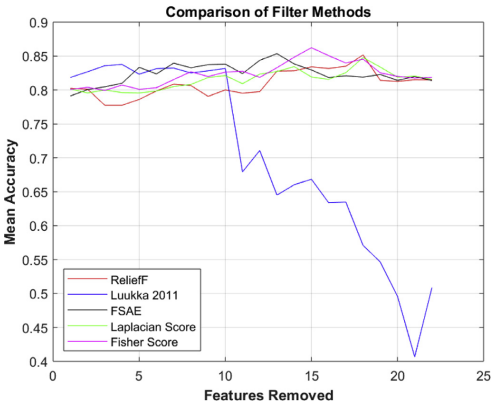


Fig. 19. Comparison of filter methods (Horse colic data set).

5. Conclusions

In this paper, an adapted version of the wrapper by Luukka (2011) was introduced that is termed fuzzy similarity and entropy (FSAE) feature selection. This approach is intended as a filter method, in particular, a feature ranking method, but is also presented as a wrapper method that is used together with a similarity classifier. The results for the FSAE feature selection method are three-fold. First, the FSAE wrapper approach can achieve at least comparable results to the benchmark feature selection method presented by Luukka (2011) but with on average fewer features. In several cases, they can even outperform the original feature selection wrapper method with a significant reduction in the number of features. Second, the FSAE filter achieves results that are at least comparable but often better than that of the method of Luukka (2011) implemented as a feature ranking method. For the first real-world data set, the dermatology data set, and the diabetic retinopathy Debrecen data no considerable difference can be observed. However, for the chronic kidney disease, the breast cancer Wisconsin data set, and the Horse colic data, the mean performance of the approach by Luukka (2011) declines considerable after a certain amount of removals. For the chronic kidney data set, this deterioration in the accuracy is present especially in the choice of the last feature to retain. For the breast cancer Wisconsin data set and the Horse colic data, the performances consistently deteriorated for each removal, ending up with mean performances that are more than 10% and 30% lower for a single feature than of all other filter methods in this study. Third, the results of the FSAE filter is often competitive to those by the other four feature selection techniques. For the first data set, the ReliefF clearly outperforms the remaining filter methods for a large range of feature removals. However, of the remaining four approaches non is clearly better throughout the entire range of removals. For the chronic kidney disease, the performance of FSAE for the initial removals and the last removals is comparable with the remaining approaches whereas for the removal of 3 to 18 features the mean accuracy is worse. For the third data, the breast cancer Wisconsin data set, the filter FSAE leads together with the Laplacian score to the highest mean accuracies throughout all the removals. For

the diabetic retinopathy Debrecen data, the FSAE wrapper leads to good and stable results, whereas the FSAE filter achieved the highest mean accuracy of all filter methods. For the last data set, the horse colic data, the FSAE wrapper reaches the same high performance as the method by Luukka (2011) but accomplishes this with 4 features less. For the filter methods, FSAE shows competitive results, being for many removals few percentage points better than ReliefF and clearly better than the filter version of Luukka (2011).

For the five medical data sets in almost all cases the results for $l=1$ and $l=2$ are the same, suggesting the same features for removal. Overall, the results clearly demonstrate that the FSAE algorithms can find irrelevant features in the data well, while retaining the discriminating features. At a minimum, performance degradation is low compared to the use of no feature selection. However, in many cases the performance accuracy can be improved compared to the wrapper feature selection by Luukka (2011) or no feature selection, and, moreover, the results to the regarded filter methods are comparable.

Acknowledgements

Peter Jones (Saimaa University of Applied Sciences) is acknowledged for his help with the English language. A preliminary version of the work was presented at the “Real Option Workshop ROW17” at Lappeenranta University of Technology, Finland. This research would like to acknowledge the funding received from the Finnish Strategic Research Council, grant number 313396 / MFG40 Manufacturing 4.0.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.eswa.2018.06.002.

References

Antal, B., Hajdu, A. *Diabetic retinopathy debrecen data set*. 2014. Retrieved from https://archive.ics.uci.edu/ml/datasets/Diabetic+Retinopathy+Debrecen+Data+Set.
Belis, M., & Guiasu, S. (1968). A quantitative-qualitative measure of information in cybernetic systems. *IEEE Transactions on Information Theory*, 593–594.
Bishop, C. M. (2006). *Pattern recognition and machine learning*. New York: Springer Science Business Media.
Blum, A. L., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence, 97*, 245–271.
Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. K. (1987). Occam's razor. *Information Processing Letters, 24*, 377–380.
Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers and Electrical Engineering, 40*, 16–28.
De Luca, A., & Termini, S. (1972). A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory. *Information and Control, 20*, 301–312.
Dougherty, G. (2013). *Pattern recognition and Classification: an introduction*. New York: Springer Science Business Media.
Ilter, N., & Guvenir, H. A. *Dermatology Data Set* Retrieved March 5, 2017, from https://archive.ics.uci.edu/ml/datasets/Dermatology.
Junttila, V., Maltamo, M., & Kauranne, T. (2008). Sparse bayesian estimation of forest stand characteristics from airborne laser scanning. *Forest Science, 54*(5), 543–552.
Kira, K., & Rendell, L. A. *A practical approach to feature selection* https://doi.org/10.1016/S0031-3203(01)00046-2.
Kira, K., & Rendell, L. A. (1992b). The feature selection problem: Traditional methods and a new algorithm. In Proceedings of the AAAI-92 (pp. 129–134).
Kittler, J., & Mardia, K. V. (1994). Statistical pattern recognition in image analysis. *Journal of Applied Statistics, 21*(1,2), 61–75.
Kononenko, I., Simec, E., & Robnik-Sikonja, M. (1997). Overcoming the myopia of inductive learning Algorithms with RELIEFF. *Applied Intelligence, 7*, 39–55.
Liang, J., Yang, S., & Winstanley, A. (2008). Invariant optimal feature selection: A distance discriminant and feature ranking based solution. *Pattern Recognition, 41*, 1429–1439.
Lichman, M. *UCI machine learning repository* Retrieved March 5, 2017, from http://archive.ics.uci.edu/ml.
Lohrmann, C., & Luukka, P. (2018). A novel similarity classifier with multiple ideal vectors based on k-means clustering. *Decision Support Systems In press*.
Luukka, P. (2007). Similarity classifier using similarity measure derived fromYu's norms in classification of medical data sets. *Computers in Biology and Medicine, 37*, 1133–1140.
Luukka, P. (2008). Similarity classifier in diagnosis of bladder cancer. *Computer Methods and Programs in Biomedicine, 89*, 43–49.
Luukka, P. (2011). Feature selection using fuzzy entropy measures with similarity classifier. *Expert Systems with Applications, 38*, 4600–4607.
Luukka, P., & Leppälampi, T. (2006). Similarity classifier with generalized mean applied to medical data. *Computers in Biology and Medicine, 36*, 1026–1040.
McLeish, M., & Cecile, M. *Horse colic data set* Retrieved from https://archive.ics.uci.edu/ml/datasets/Horse+Colic.
Parkash, O., Sharma, P., & Mahajan, R. (2008). New measures of weighted fuzzy entropy and their applications for the study of maximum weighted fuzzy entropy principle. *Information Sciences, 178*, 2389–2395.
Quinlan, J. R. (1992). *C4.5: programs for machine learning*. San Mateo: Morgan Kaufmann Publishers Ed.
Robnik-Šikonja, M., & Kononenko, I. (2003). Theoretical and empirical analysis of relieff and rrelieff. *Machine Learning, 53*(1–2), 23–69. https://doi.org/10.1023/A:1025667309714.
Seo, M., & Oh, S. (2012). CBFS: High performance feature selection algorithm based on feature clearness. *PLoS ONE, 7*(7), 1–10.
Soundarapandian, P., & Rubini, L. (2015). Chronic kidney disease data set. https://archive.ics.uci.edu/ml/datasets/chronic_kidney_disease.
Souza, J. T., Matwin, S., & Japkowicz, N. (2006). Parallelizing feature selection. *Algorithmica, 45*, 433–456.
Yao, Y. Y., Wong, S. K., & Butz, C. J. (1999). On information-theoretic measures of attribute importance. In Proceedings of the pacificasia conference on knowledge discovery and data mining (pp. 133–137).

Publication II

Lohrmann, C., Luukka, P.
Classification of intraday S&P500 returns with a random forest

Reprinted with permission from
International Journal of Forecasting
Vol. 35, pp. 390-407, 2019

International Journal of Forecasting 35 (2019) 390–407

Contents lists available at ScienceDirect

International Journal of Forecasting

journal homepage: www.elsevier.com/locate/ijforecast

Classification of intraday S&P500 returns with a Random Forest

Christoph Lohrmann [a,b,*], Pasi Luukka [b]

[a] LUT University - School of Engineering Science, Skinnarilankatu 34, Lappeenranta, South Karelia 53850, Finland
[b] LUT University - School of Business, Skinnarilankatu 34, Lappeenranta, South Karelia 53850, Finland

ARTICLE INFO

Keywords:
Financial markets
Machine learning
Forecasting
Trading strategy
Feature selection

ABSTRACT

Stock markets can be interpreted to a certain extent as prediction markets, since they can incorporate and represent the different opinions of investors who disagree on the implications of the available information on past and expected events and trade on their beliefs in order to achieve profits. Many forecast models have been developed for predicting the future state of stock markets, with the aim of using this knowledge in a trading strategy. This paper interprets the classification of the S&P500 open-to-close returns as a four-class problem. We compare four trading strategies based on a random forest classifier to a buy-and-hold strategy. The results show that predicting the classes with higher absolute returns, 'strong positive' and 'strong negative', contributed the most to the trading strategies on average. This finding can help shed light on the way in which using additional event outcomes for the classification beyond a simple upward or downward movement can potentially improve a trading strategy.

1. Introduction

1.1. Background

The ability to predict the stock market accurately is of pivotal interest to investors, stakeholders, researchers and even governments (Fadlalla & Amani, 2014). For instance, investors use forecasts as a tool for making investment decisions (Lu & Wu, 2011), to identify opportunities and challenges in a market (Enke, Grauer, & Mehdiyev, 2011) and to generate trading strategies (Krauss, Do, & Huck, 2017; Leigh, Purvis, & Ragusa, 2002; Leung, Daouk, & Chen, 2000). Many forecast models assume that the past can be analyzed in order to approximate future stock market movements (Guerard Jr., 2013). The two main forms of analysis that are deployed for generating investment strategies are technical and fundamental analysis. Technical analysis focuses on the market price dynamics and trading volume behavior in order to predict the future behavior of a stock or financial market (Leigh et al., 2002). The technical approach follows the assumption that the price patterns that have occurred in the past repeat and will continue to occur in the future and, therefore, can be used to predict future price movements (Bodie, Kane, & Marcus, 2009; Fama, 1965b). It can be regarded as a pattern recognition problem (Felsen, 1975; Guo, Wang, Liu, & Yang, 2014). In contrast, fundamental analysis assumes that stock prices are predicated on fundamental data. Fundamental analysis uses information such as company-specific earnings and prospects to predict future cash flows and the company's value to forecast future stock price movements (Bodie et al., 2009; Leigh et al., 2002).

However, the efficient market hypothesis (EMH) states that market prices follow a random walk and therefore cannot be forecasted based on past market movements and behaviors (Leigh et al., 2002). The efficient market hypothesis (EMH), introduced by Eugene Fama in 1970,

* Correspondence to: Lappeenranta University of Technology, School of Engineering Science, Skinnarilankatu 34, 53850 Lappeenranta, Finland.
E-mail addresses: christoph.lohrmann@student.lut.fi (C. Lohrmann), pasi.luukka@lut.fi (P. Luukka).

https://doi.org/10.1016/j.ijforecast.2018.08.004
0169-2070/

defines a financial market as 'efficient' if it reflects the available information fully (Fama, 1970). One form of financial market that is related to stock markets via the concept of information efficiency is the prediction market, which can be regarded as a new form of betting market (Page, 2012), as traders on prediction markets are effectively betting on the outcome of a certain event. The payoff from this bet depends on the outcome of that future event (Wolfers & Zitzewitz, 2004). The price setting occurs between traders, as on a financial market, and in the case of well-calibrated traders may be considered as an estimate of the probability of the event (Page, 2012). A central aspect for financial markets, including prediction markets, is information efficiency, which refers to the fact that information is incorporated in and reflected by a price, and that no market participant is able to influence the market price directly (Vaughan Williams & Reade, 2016). However, prediction markets also show irrationality and anomalies that are common for other financial markets, such as price misalignments (Rothschild & Pennock, 2014) and the tendency to overweight low-probability events and underweight events that are almost certain to occur (Wolfers & Zitzewitz, 2004). Consequently, prediction markets show many parallels to financial stock markets. To a certain extent, stock markets can also be interpreted as prediction markets, because, according to Fama (1965a), they reflect the effects of information on past events as well as events that are expected to happen. Moreover, Fama (1970) stated that market efficiency does not have to be violated by the fact that investors disagree on the implications of the available information, as long as no investor can consistently beat the market. As a consequence, a financial market such as a stock market can incorporate and represent the different opinions of investors who disagree on the implications of the available information and trade on their beliefs in order to achieve a profit.

Various different strategies have been developed over the years for forecasting stock prices and returns. These strategies include support vector machines (Guo et al., 2014; Guo-qiang, 2011), neural network models (Altay & Satman, 2005; Fadlalla & Amani, 2014), the random subspace classifier (Zhora, 2005), systems incorporating genetic algorithms (Kim, Han, & Lee, 2004; Leigh et al., 2002) and case-based reasoning (Chun & Park, 2005). Cao and Tay (2001) use a support vector machine to forecast the S&P500 daily index between 1993 and 1995 by transforming the data into five-day relative difference in percentage (RDP) values, and use lagged RDP values and technical indicators for the prediction. Their model obtains better forecasting results in terms of normalized mean squared errors (NMSE) than a backpropagation neural network. Kim (2003) also used support vector machines on daily data from the Korean stock exchange (KOSPI) from 1989 to 1998. He deployed 12 common technical indicators and found the SVM to outperform the benchmark neural network and the CBR model, obtaining results that were compatible with those of Cao and Tay (2001). Kim et al. (2004) used a hybrid integration mechanism with a fuzzy genetic algorithm that encompasses nine technical indicators such as the moving average, the relative strength index (RSI) and the stochastic %D. Their approach can generate accurate results for the prediction of the Korean stock index KOSPI. Subsequent research by Kim, Min, and Han (2006) used five technical indicators from weekly KOSPI index data from 1990 to 2001 to address this as a four-class classification problem. They combined knowledge obtained from a neural network and human experts for a genetic algorithm and were able to outperform the benchmark methods. However, the data set used for this research was rather small, with only 312 weekly observations. Teixeira and De Oliveira (2010) built a method for automatic stock trading based on a k-nearest-neighbor classifier, with the inputs to the model including closing prices, trading volumes and technical indicators such as moving averages, the RSI, stochastics and Bollinger bands. For daily data from 1998 to 2009 for 15 stocks from the Sao Paulo stock exchange, they managed to achieve profits after transaction costs for 12 of the 15 stocks. For the two-class problem, they accomplished these profits even though the accuracy of the KNN classifier was well below 50%. Nyberg (2013) used monthly data from 1957 to 2010, encompassing both technical and fundamental data (e.g. industrial production and unemployment), to predict bear and bull markets for the S&P500. Using a dynamic probit model, they were able to produce predictions for these two types of market sentiments that were superior to those from a static model. Bhaduri and Saraogi (2010) investigated stock and bond markets with a probit model with the aim of identifying bull and bear markets and finding a relationship between yield spreads and these market states. They use a proxy for the Indian stock market from 1996 to 2008 (monthly) to find entry and exit points to the market, and achieve returns in excess of those of a conventional buy-and-hold strategy. Guo et al. (2014) used 39 features, including the open price, high price, low price, moving averages, momentum terms, RSI, stochastic %K and %D, MACD, momentum and other technical indicators, to forecast the Shanghai stock market and the Dow Jones index. Their model outperformed the two other models that they compared their classifier with. Fadlalla and Amani (2014) used 10 features to predict the Qatar Stock Exchange closing price, including simple and weighted moving averages, RSI, MACD, stochastic %K and %D momentum, and the commodity channel index. Their neural network outperformed an ARIMA model on the given dataset. The study by Karymshakov and Abdykaparov (2012) on forecasting price movements of the Istanbul Stock Exchange (ISE) included a currency exchange rate, the gold price, common technical indicators such as moving averages and price information such as the high and low prices of the ISE during a trading day. O'Connor and Madden (2006) constructed a neural network for forecasting the Dow Jones Industrial Average Index and incorporated fundamental factors including currency exchange rates and commodity prices (crude oil). They report an accurate model performance. Research by Lendasse, De Bodt, Wertz, and Verleysen (2000) deployed external variables such as other stock market indices, exchange rates and interest rates, combined with technical indicators of the daily Belgian Bel 20 stock index, to predict the sign of the change up to five days in the future. Niaki and Hoseinzade (2013) included 27 financial and economic factors in their analysis for forecasting the direction of the daily S&P500 and were

able to outperform a buy-and-hold strategy. Moreover, Zhong and Enke (2017) included the factors of Niaki and Hoseinzade (2013) among the variables in their study for forecasting the direction of the closing price of the SPDR S&P 500 ETF. Their variables encompassed 60 financial and economic factors, including the trading volume of the SPDR S&P 500 ETF, interest and exchange rates, commodity prices, other stock market indices and common technical indicators. Their results show that they are able to outperform the benchmarks, including a buy-and-hold strategy, significantly in terms of risk-adjusted returns.

Some authors attempted to identify a pattern for the classification of transformed technical indicators into features that represent a trading signal/ strategy in order to better classify returns. Leigh et al. (2002) deployed a combination of a genetic algorithm and a neural network that attempted to use the "bull flag" pattern to predict the NYSE Composite Index. They forecast stock prices successfully and showed a violation of the weak-form EMH. Chang and Wu (2015) used daily data from 15 US stocks between 2008 and 2012 to compute 32 technical indicators. Using kernel-based feature extraction to identify trading signals, and their stock trading model with SVR, they reached a higher profitability than the other dimensionality-reduction methods with this classifier. Patel, Shah, Thakkar, and Kotecha (2015) extracted trend deterministic data from 10 technical indicators for two stocks and two stock indices (CNX Nifty and S&P Bombay Stock Exchange Sensex) from daily data from 2003 to 2012. They showed that the performances of all of the prediction models in their study improved when the technical indicators were converted into trend deterministic data.

Overall, previous research and momentum anomalies (Leigh et al., 2002) have indicated that both fundamental factors and technical indicators can be integrated successfully into a trading strategy.

1.2. Objectives

The objective of this paper is to use feature selection together with the ensemble classifier random forest to build a classification model for predicting the open-to-close returns of one of the main equity indices, the S&P500, in a four-class setting. Subsequently, a more detailed analysis of the feature importance will be conducted to gain a better understanding of which features are relevant for the prediction task. The classifier and its result on the feature subset from the feature selection will then be used as the basis of several trading strategies. These trading strategies will be derived from the four classes related to the magnitude of the S&P500 open-to-close returns, and their performances will be benchmarked against a buy-and-hold strategy. Finally, the contributions of the four classes in the prediction to the trading strategies will be investigated.

The remaining paper is structured as follows: Section 2 discusses the methods, including the feature selection algorithm, and Section 3 depicts the data set and the application of the methodology to it. Section 4 presents the results for the random forest classifier and the analysis of the feature importance and the investment strategies, which are evaluated critically with respect to the buy-and-hold strategy and the contributions of the predicted classes to the returns in Section 5.

Table 1
Example observations.

Observation	Feature 1	Feature 2	Class
X_1	5	10	1
X_2	5.2	30	1
X_3	5.1	50	2
X_4	4.9	70	2

2. Methodology

2.1. Entropy measures

Entropy can be regarded as a "measure of the degree of fuzziness" (De Luca & Termini, 1972). Furthermore, De Luca and Termini (1972) described it as the average information contained in a dataset that is available when making a decision.

This paper will apply the entropy measures defined by De Luca and Termini (1972) and Parkash, Sharma, and Mahajan (2008) for feature selection. This entropy measure can be defined as (De Luca & Termini, 1972):

$$H(A) = -\sum_{i=1}^{n} [\mu_A(x_i) \log \mu_A(x_i) + (1 - \mu_A(x_i)) \log (1 - \mu_A(x_i))], \quad (1)$$

where $\mu_A(x_i) \in [0, 1]$ is the membership degree of x_i to the fuzzy set A. The entropy measures introduced by Parkash et al. (2008) are related to the concept of weighted entropy (Belis & Guiasu, 1968), and are defined as follows:

$$H^1(A) = \sum_{i=1}^{n} w_i \left[\sin \frac{\pi \mu_A(x_i)}{2} + \sin \frac{\pi (1 - \mu_A(x_i))}{2} - 1 \right] \quad (2)$$

$$H^2(A) = \sum_{i=1}^{n} w_i \left[\cos \frac{\pi \mu_A(x_i)}{2} + \cos \frac{\pi (1 - \mu_A(x_i))}{2} - 1 \right]. \quad (3)$$

The shape of the entropy function values is illustrated in Fig. 1. The characteristic that all three entropy measures share is that they reach their maximums at an input of 0.5, while their minimums are reached at inputs of 0 and 1. The idea of an entropy measure is that a small entropy value, which is reached with an input close to 0 or 1, represents certainty and structure, while high entropy values, which occur for inputs close to 0.5, suggest uncertainty and a low level of informativity (Yao, Wong, & Butz, 1999). This aspect can be used for classification tasks in combination with similarity. Since the outputs obtained with both entropy measures of Parkash et al. (2008) are the same for the same input values, it is sufficient to consider only the first measure (see Eq. (6)) in what follows.

Imagine a simple classification problem where four observations are available. The observations belong to one of two classes and contain two features that are independent of each other, as is presented in Table 1.

After scaling the observation values of each feature to the unit interval with max–min-scaling, the observations and ideal vectors can be illustrated as in Fig. 2. It is apparent

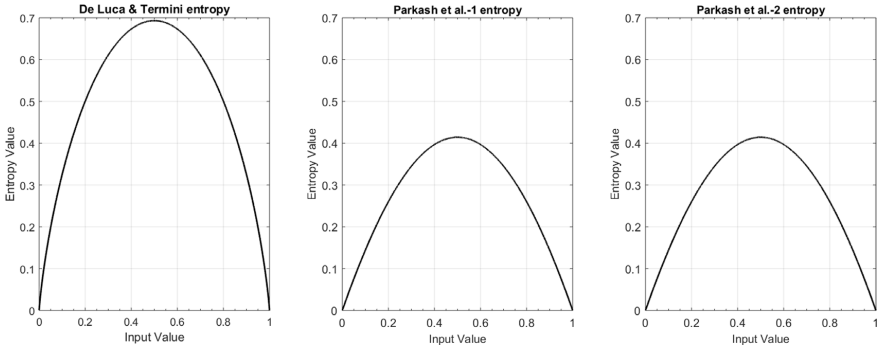


Fig. 1. Comparison of different entropy measures.

from Fig. 2 that the scaled feature values vary strongly within each class for the first feature and considerably less for the second feature. Moreover, the classes for the first scaled feature overlap, whereas those for the second feature take values in a different range of values.

Following the logic of the feature selection algorithm proposed by Luukka (2011), the first step is to calculate an ideal vector for each class that aims to characterize each class well (Luukka, Saastamoinen, & Könönen, 2001). An ideal vector should differ between classes so that it can discriminate between these classes well.

There exist several ways of computing ideal vectors, with the arithmetic mean being one of the earliest methods. An ideal vector can also be calculated with more generalized mean operators such as the generalized mean, the Bonferroni mean or ordered weighted averaging (OWA).

Using the generalized mean, the ideal vector v_i for a class C_i is:

$$v_{i,d} = \left(\frac{1}{\#X_i} \sum_{x \in X_i} {x_d}^m \right)^{\frac{1}{m}}, \quad \forall d = 1, \ldots, D, \tag{4}$$

where $v_{i,d}$ is the value of the ideal vector for class i for feature d and $\#X_i$ refers to the number of observations that belong to class i. For this simple example, $m = 1$, so that the ideal vector is simply the class mean for each feature.

The second step is the calculation of the similarity between each ideal vector v_i and each observation x_j, with $j = 1$ to n. This is carried out by computing the similarities between the observation and the ideal vector:

$$S\left(x_{j,d}, v_{i,d}\right) = \sqrt[p]{1 - \left| {x_{j,d}}^p - {v_{i,d}}^p \right|}, \tag{5}$$

where, for simplicity, p is set to 1. Using Eq. (1) for De Luca and Termini entropy, entropy values for each feature d can be obtained as

$$H_d = \sum_{i=1}^{N} \sum_{j=1}^{n} H\left(S\left(x_{j,d}, v_{i,d}\right)\right). \tag{6}$$

The entropy value H_d for each feature, which is the sum of the two entropy values for that feature in the two classes,

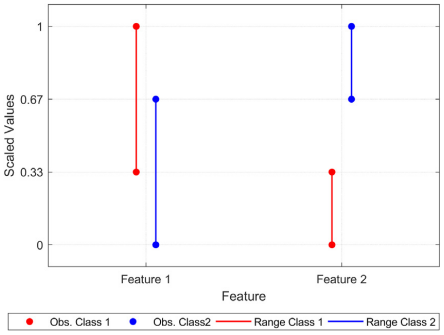


Fig. 2. Scaled example observations.

indicates which feature should be removed. The removal decision is based on the idea that small entropy values refer to regularities and structure in a dataset, while high entropy values indicate randomness (Yao et al., 1999). This means that entropy can show whether data are characterized by uncertainty or are informative (Luukka, 2007).

More specifically, fuzzy entropy measures can be used to determine the relevance of features (Luukka, 2011). For feature removal, the feature that has the highest entropy value and, therefore, is assumed to be least informative, will be removed; thus, for the given example, feature 2 would be removed (see Table 2).

This example illustrates how entropy can be included in the feature selection so as to remove less informative features. However, a closer look suggests that the proposed feature removal for the given example is not a good choice with respect to the classification accuracy. Assuming that the observations in this example are a representative sample of the underlying population, it is obvious that the feature space for feature 2 can be split into distinct regions for the decision making, whereas feature 1 shows a clear overlap in the values that this feature takes in the two

Table 2
Example similarity and entropy values.

Similarity	Feature 1	Feature 2	Class
$S(x_1, v_1)$	0.67	0.83	1
$S(x_2, v_1)$	0.67	0.83	1
$S(x_3, v_1)$	1.00	0.50	1
$S(x_4, v_1)$	0.33	0.17	1
$S(x_1, v_2)$	1.00	0.17	2
$S(x_2, v_2)$	0.33	0.50	2
$S(x_3, v_2)$	0.67	0.83	2
$S(x_4, v_2)$	0.67	0.83	2
$H(d$, Class1)	1.91	2.04	1
$H(d$, Class 2)	1.91	2.04	2
$H(d)$	3.82	4.09	

Table 3
Example of scaled entropy.

Entropy	Feature 1	Feature 2
$H(d)$	3.82	4.09
$SE(d)$	2.55	1.36

different classes. In simple terms, the second feature allows an observation to be assigned to one of the classes without uncertainty, but the first feature does not. Thus, the removal of the second feature will lead to a deterioration in the classification with these observations. However, as this simple example has illustrated, this shortcoming can be overcome by using a scaling factor for the entropies that takes into account the distance between a feature value of one ideal vector and the corresponding feature values of all other ideal vectors (Lohrmann, Luukka, Jablonska-Sabuka, & Kauranne, 2018). Feature selection based on fuzzy similarity and entropy measures (FSAE), which uses a scaling factor for the entropy values, is discussed in more detail in the subsequent section.

2.2. Fuzzy similarity and entropy measure (FSAE) based feature selection

Feature selection using fuzzy similarity and entropy measures (FSAE) using scaled entropy was introduced by Lohrmann et al. (2018), and has its origin in the algorithm developed by Luukka (2011). This feature selection algorithm is designed as a filter method, and in particular a feature ranking method, but can also be used as a wrapper; for instance, together with a similarity classifier. It is based on the idea of using scaled fuzzy entropy measures on similarity values to determine the importance of features. First, the similarity $S \in [0, 1]$ is calculated, where 0 implies complete dissimilarity of an observation to the ideal vector while 1 emphasizes the highest degree of similarity. Second, the entropy values for similarities are computed. Similarities of 0 or 1 will lead to the lowest entropies, which emphasizes high informativeness. On the other hand, a similarity close to 0.5 results in the highest entropy value and signals uncertainty. This idea is applied to a classification problem in order to calculate the similarity of features from observations with the ideal vector of each class and determine their entropy values. This entropy value will be low if the feature is highly informative and high if the uncertainty of the feature is high (Luukka, 2011). In addition, a scale factor for the entropy values is used to emphasize the distances among the ideal vectors of the classes. Using the scale factor on the entropy has the desirable property that distinct features of ideal vectors decrease the entropy value, while the entropy values of features where the ideal vectors are close remain at their initial level or decrease only slightly. In other words, if a feature on average has largely different values in one class from those in all other classes, this results in a smaller entropy value. In this case, the scaled entropy will indicate that the feature is more informative.

In generalized form, the scaling factor SF can be denoted as

$$SF_{i,d} = 1 - \frac{\left(\sum_{o \neq j} \left| v_{i,d} - v_{o,d} \right|^l\right)^{\frac{1}{l}}}{N - 1}. \tag{7}$$

The numerator determines the sum of the absolute distances of the ideal vector value for feature d for class i to all other classes (in the most simple case with $l = 1$).

The scaled entropy SE for a feature d for all classes is calculated based on Eq. (6) for the entropy and Eq. (7) for the scaling factor:

$$SE_d = \sum_{i=1}^{N} \left(\sum_{j=1}^{n} H\left(S\left(x_{j,d}, v_{i,d}\right)\right) \right) * \left(1 - \frac{\left(\sum_{o \neq j} \left| v_{i,d} - v_{o,d} \right|^l\right)^{\frac{1}{l}}}{N - 1} \right). \tag{8}$$

The result of the FSAE filter is a scaled entropy value for each feature. Since high scaled entropy values indicate uncertainty, the features with the lowest scaled entropy values are most important for distinguishing between classes. For the feature selection, the user specifies which number of features (denoted k) should be kept, and subsequently only uses the k features with the lowest scaled entropy values. The underlying assumption is that this removes features that do not contribute to the deviation among classes (Luukka, 2011).

For the simple example presented in Section 2.1, the use of the scaled entropy (SE) changes the feature removal from feature 2 to feature 1 (see Table 3).

The second feature has a higher degree of informativity than the first, since the distance between the values of the ideal vectors of the two classes is larger for this feature than for the first feature. The scaling factor accounts for this interclass distance, which led to the decision to remove feature 1 instead.

The feature importance based on the scaled entropy values determined by the FSAE filter method can be presented in an intuitive form between [0,1] by dividing the scaled entropy by the sum of scaled entropy values, subtracting this value from 1, and standardizing the resulting feature importance vector to [0,1].

$$FI_d = 1 - \frac{SE_d}{\sum_{d=1}^{D} SE_d}. \tag{9}$$

The feature importance will be close to one for informative features, while uninformative and irrelevant features will show feature importance values of close or equal to zero.

This feature selection method was chosen because it showed results that were competitive with those of the most common feature selection methods, is intuitive in its use and is computationally inexpensive (Lohrmann et al., 2018). Moreover, using a feature ranking method allows the feature importance values to be analyzed for a single feature as well as for a group of features.

3. Application of our methodology to the data

3.1. Data

The dataset that is analysed in this paper consists of time series obtained from Yahoo Finance (2017). The time horizon for the training and testing of the classifier is from 10/11/2010 to 04/29/2016, and the time period selected for the out-of-sample forecast is from 05/2/2016 to 3/28/2018 . The dependent variable is the daily open-to-close return of the S&P500 index from the opening price to the closing price of a single trading day. The feature dataset includes seven financial market indices, two market ETFs, six indices and ETFs related to sectors and commodities, three currency time series, seven time series related to interest rates, yields and yield spreads, nine technical indicators, and the VIX index. The seven financial market indices encompass large indices such as the S&P500 (US), the STOXX50 (EU), the Hang Seng (C), the Nikkei225 (J), the FTSE100 (UK), and the DAX (GER). Moreover, it contains the Russel 2000, which is an index premised on firms with small market capitalization in the US. The two market exchange traded funds (ETF) are intended to represent large- and medium-capitalized emerging market companies (iShares MSCI Emerging Markets; see BlackRock, 2017a), and to track the performances of large-, medium- and small-capitalized firms worldwide (Vanguard Total World Stock ETF; see Morningstar, 2017).

Commodities and materials are represented by the United States Commodity Index and the SPDR S&P U.S. Materials Select Sector UCITS ETF, respectively. The former is supposed to reflect the performance of a portfolio of commodity futures, which represents the energy, precious metals, industrial metals, grains, livestock and softs sectors (USCF, 2017). The latter aims to track the performance of American large-capitalized material firms within the S&P500 (State Street Global Advisors (SPDR), 2017b). These features are intended to capture the influence of the commodities and materials sector on the American economy, and thus, on the American stock market. The SPDR Gold Shares ETF is the largest physically backed gold ETF, and tracks mainly long exposure to gold. This ETF is included because commodities, especially gold, can indicate future inflation, and their price volatility is believed to have negative consequences for financial markets (Baur, 2012), which may have a severe impact on the US economy (Gokmenoglu & Fazlollahi, 2015). The iPath S&P GSCI Crude Oil Total Return Index concentrates its exposure on the S&P GSCI Crude Oil Total Return Index, which is a benchmark for the total return accomplished in the crude oil market (S & P Dow Jones Indices, 2017). This index is used to incorporate the impact of changes in the crude oil market on financial markets, as the oil price can have an extensive impact on both the economy and financial markets (Gokmenoglu & Fazlollahi, 2015). Three additional factors are the exchange rates of the USD to the Yen (Japan), the Euro (EU) and the Yuan (China), which reflect the attractiveness of US exports and its purchasing power with respect to imports for the US economy relative to its largest trading partners. We also integrate the relevance of the financial sector for the S&P500 index by including the Financial Select Sector ETF and the iShares MSCI Europe Financials. The former is supposed to track the investment results of large financial companies in the US that are listed in the S&P 500 (State Street Global Advisors (SPDR), 2017a), while the latter attempts to track the performance of an index of European equities in the financial sector (BlackRock, 2017b).

The category encompassing interest rates and yields contains the CBOE 10-year interest rate, the 30-year Treasury yield (US), the 5-year Treasury yield (US) and the 13-week Treasury bill (US). The CBOE 10-year interest rate is a time series of the Chicago Board Options Exchange that represents interest rate options for the 10-year Treasury note (Chicago Board Options Exchange, 2017). The short-term 13-week Treasury bill (US) will also be used as a proxy for the 13-week yield for the calculation of two of the yield spreads. From the three yield curves, the 30-year-to-5-year yield spread, the 30-year-to-13-weeks yield spread, and the 5-year-to-13-weeks yield spread are computed as additional time series. No 10-year yield curve is available from Yahoo Finance, but a 30-year yield curve and a 5-year yield curve are used instead, in addition to the 13-week one. This follows the convention of using a short-term government yield curve and a long-term one of at least several years for the calculation of yield spreads (Bhaduri & Saraogi, 2010; Nyberg, 2013; Rudebusch & Williams, 2009). Research has indicated that yield spreads contain useful information in relation to the contraction and expansion of the economy, and therefore they might also be of relevance for predicting a stock market index (Rudebusch & Williams, 2009). Finally, the authors use the VIX as an additional financial time series to represent the market sentiment. The volatility index VIX is included in the features because it can be regarded as a barometer for investor sentiment in the market (Rossilo, Giner, & de la Fuente, 2014).

In general, our choice of features is in line with previous research that has used at least a subset of these features, such as lagged index data, technical indicators, the oil price, exchange rates, the gold price, or short- and long-term interest rates/yields (Krollner, Vanstone, & Finnie, 2010).

For each of these time series, the closing and opening prices, daily high and low values and volumes are downloaded if available. These data are included in the feature dataset because price and volume information are the major components in technical analysis (Achelis, 1995). Moreover, the daily range values of the indices are derived from their daily high and low values. The range indicates the maximum daily variation in the price series. For each time series, we also calculate the returns between the opening

Table 4
List of features.

Dependent variable						
S&P500	Open-close return					
Features						
Time series						
S&P500	–	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
DAX	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
Nikkei225	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
iShares MSCI Emerging Markets	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
Vanguard Total World Stock ETF	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
Hang Seng	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
FTSE 100	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
STOXX 50	Open-close return	Close-open return	–	-	–	–
Russell 2000	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
VIX S&P500	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
SPDR Gold Shares ETF	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
United States Commodity Index	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
Materials Select Sector SPDR ETF	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
iPath S&P GSCI Crude Oil	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
Financial Select Sector	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
iShares MSCI Europe Financials	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	Δ (%) Volume	Δ (%) Range
CBOE Interest Rate 10 Year	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
Treasury Yield 30 Years	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
Treasury Yield 5 Years	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
13 Week Treasury Bill	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
JPY/USD Exchange Rate	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
EUR/USD Exchange Rate	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
CNY/USD Exchange Rate	Open-close return	Close-open return	Δ (%) High	Δ (%) Low	–	Δ (%) Range
Technical indicators and yield spreads						
Δ (%) Spread Treasury 30y – 5y	Δ (%) momentum (1d)	MACD (26d, 12d, Signal 9d)	Mov.Avg. (5d)			
Δ (%) Spread Treasury 30y – 13w	Δ (%) momentum (3d)	Bollinger (2 Std)	Mov.Avg. (10d)			
Δ (%) Spread Treasury 5y – 13w	Δ (%) momentum (5d)	RSI (14d)				
	Δ (%) momentum (10d)					

and the respective closing prices, which are the changes that occur during the trading day, as well as the returns between the closing and opening prices, reflecting the price changes during the times when no trading is occurring. Table 4 lists all features for the classification problem.

The remaining features are technical indicators, including the changes in the 1-, 3-, 5- and 10-day momentums, the relative strength index (RSI), the moving average convergence–divergence (MACD), moving averages (MA) and Bollinger bands (Di Lorenzo, 2013; Hurwitz & Marwala, 2011). The technical indicators MACD, RSI and Bollinger bands are transformed into trading signals instead of just using their values directly. A comparable approach was pursued by Patel et al. (2015), who referred to this as trend deterministic data and found that this transformation led to a significantly higher classification accuracy than using the technical indicators in their original form. The transformation into signals follows common trading rules. An RSI of less than 30 takes a value of 1 (oversold/buy signal), an RSI of over 70 takes a value of 0 (overbought/sell signal), and an RSI between 30 and 70 takes a value of 0.5. For the MACD, the feature is assigned a value of 1 if the MACD (26 days EMA/12 days MA) is smaller than the signal line (exponential moving average (EMA), 9 days), which indicates to buy, and a value of zero if the MACD exceeds the signal line. If the MACD equals the signal line, the feature takes on a value of 0.5 (Achelis, 1995).

Lastly, the Bollinger bands feature is assigned a value of 1 if the signal line crosses the lower Bollinger band from below (buy signal), a value of 0 if the signal line crosses the upper Bollinger band from above (sell signal), and 0.5 if the signal line is within the lower and upper Bollinger bands (Di Lorenzo, 2013).

When the initial time series are downloaded from Yahoo Finance, volume time series for which no information can be downloaded and where the entire feature download consists of zeros are removed. For the remaining features, missing values are replaced by the last previously known value to avoid biasing the time series with future values that are yet unknown (e.g. through interpolation). Moreover, the rank of the data set is calculated in order to check the features for multicollinearity, with the objective of removing features that are highly correlated with other features in the dataset, which essentially means that this feature does not provide any additional information for the classification. However, no features have to be removed here since no multicollinearity is present in the dataset.

After this procedure, the training and testing dataset contains 1373 observations and the dataset for the forecasting period consists of 481 observations for all 136 features. Finally, the features are normalized to the unit interval [0, 1]. The time series to be classified is the open-to-close return of the S&P 500. The open-to-close returns are split into four classes according to their daily magnitudes. Table 5 lists the classes for both the training/testing and forecast periods.

The idea is to create distinct groups for ‘strong positive’, ‘slightly positive’, ‘slightly negative’ and ‘strong negative’

Table 5
Classes for the S&P 500 closing returns.

S&P 500 closing return	Class	Training and testing		Forecast period	
		Observations	in %	Observations	in %
Larger than 0.5%	1	345	25.13%	59	12.27%
Between 0.5% and 0.0%	2	399	29.06%	199	41.37%
Between 0.0% and −0.5%	3	339	24.69%	172	35.76%
Smaller than −0.5%	4	290	21.12%	51	10.60%

returns. The proportions of the four classes are similar in the training and testing data, but more unbalanced in the forecast period, with slightly positive and slightly negative returns being the majority classes and higher-magnitude returns having smaller numbers of observations. The classification of the daily S&P data into four classes is one of the aspects that differentiates the approach pursued in this paper from most of the existing literature. Kim et al. (2006) used a four-class approach, but on weekly data for the Korean KOSPI index. Patel et al. (2015) worked with their trend deterministic features on a binary class case, but mentioned that a setting with more categories is also worth exploring.

Another aspect that is worth mentioning is that all of the information presented can be downloaded without cost, and is available easily to potential investors and researchers.

3.2. Training procedure

The first step is feature selection for the initial 136 features in the data set, and for that purpose we use the FSAE filter method. Each feature is ranked according to FSAE based on its scaled entropy value (in ascending order). Initially, we calculate the performance using a simple similarity classifier with all features, and one feature will be removed in each step, that with the next lowest rank. This procedure is conducted using the FSAE with different combinations of entropy measures and l-parameters. Each setup is tested for different values of the p (from 1 to 8) and m (from 1 to 6) parameters, and we report only the accuracies for the p and m values that lead to the highest mean accuracy for each combination of entropy and l-parameter. Finally, we choose the setup of the entropy and l-parameter that appears most suitable in terms of performance and number of features removed.

With this setup, the main classifier in this study, the random forest (Breiman, 2001), will be used together with the FSAE to determine which features should be removed. The random forest is an ensemble classifier that trains multiple decision trees and combines their results in order to assign observations to a class (Adele, Cutler, & Stevens, 2012). This procedure is supposed to avoid the common problem of single decision trees, which tend to overfit data easily if their parameters are not set suitably. Other advantages of this model include its ability to model interactions between features and its robustness to outlier values for features (Hastie, Tibshirani, & Friedman, 2009). Here, the random forest will consist of 50 decision trees. As was demonstrated by Breiman (2001), the choice of the number of decision trees can be as desired, since the generalization error is converging to a limit. Random forests have been applied successfully in a variety of different applications (Adele et al., 2012), including the classification of financial time series. Khaidem, Saha, and Dey (2016) used a random forest in a context comparable to that in this paper, using technical indicators but considering the prediction of stock returns, and reported high classification results. Recently, Zhang, Cui, Xu, Li, and Li (2018) used a random forest in their stock price trend prediction system and demonstrated its ability to outperform a KNN classifier, support vector machines and an artificial neural network.

In order to find a model with limited complexity and a good generalization ability for previously unseen observations, noise can be added during the training procedure (Özesmi, Tan, & Özesmi, 2006). Since noise is supposed to prevent a model from being overfitted to the given observations, adding noise should make the learning algorithm less sensitive to the variation in the features for reasonable amounts of noise, thus preventing overfitting (Matsuoka, 1992; Özesmi et al., 2006). In this paper, the authors add independently and uniformly distributed noise to the features before using the random forest algorithm with FSAE for the feature selection. The idea behind this proceeding is that using a certain amount of noise should make the choice of features more robust and reliable.

Once more, the classifier, in this case the random forest, will be used first with all features, then features will be removed iteratively based on their ranking from the FSAE feature selection. For this purpose, the entire testing and training time series (with 1373 observations) is split using the hold-out method into 70% of observations for training and the remaining 30% for testing. The use of stratified sampling ensures that the training and test data consist of observations from all four return classes and in proportions that represent the classes. Noise is added to the training data solely to ensure that the result generalizes to the actually observed test data, not to fit the perturbed data. The magnitude of the noise added at each iteration is varied from +/− 0 standard deviations (Std) up to +/− 4 Std, by steps of 1 Std. The standard deviation is determined based on the feature values of all observations. The level of noise that is added is random, but is limited in each step to +/− the number of standard deviations for that level of noise. The only exception is when adding noise to the technical indicators and the trading strategies premised on these indicators. Since these indicators all depend on the price series, it would be inconsistent and implausible to perturb them separately rather than perturbing the underlying stock price series. Thus, we inject the trading strategies with noise by perturbing the underlying S&P500 closing price series according to the standard deviation of the S&P500 index (from +/− 0 up to +/− 4 Std), then determine the technical indicators and trading strategy values

Table 6
Classification results with feature selection (before noise injection).

Approach	Parameter	Entropy	No. of features	Features removed	Avg. performance	Variance (in %)	p	m
Sim		–	136	None	44.73%	0.05%	2	2
Sim + FSAE	$l = 1$	De Luca & Termini	121	15	44.78%	0.03%	5	3
Sim + FSAE	$l = 1$	Parkash et al.	64	72	44.38%	0.05%	3	1
Sim + FSAE	$l = 2$	De Luca & Termini	119	17	43.06%	0.06%	2	2
Sim + FSAE	$l = 2$	Parkash et al.	120	16	42.91%	0.05%	3	2

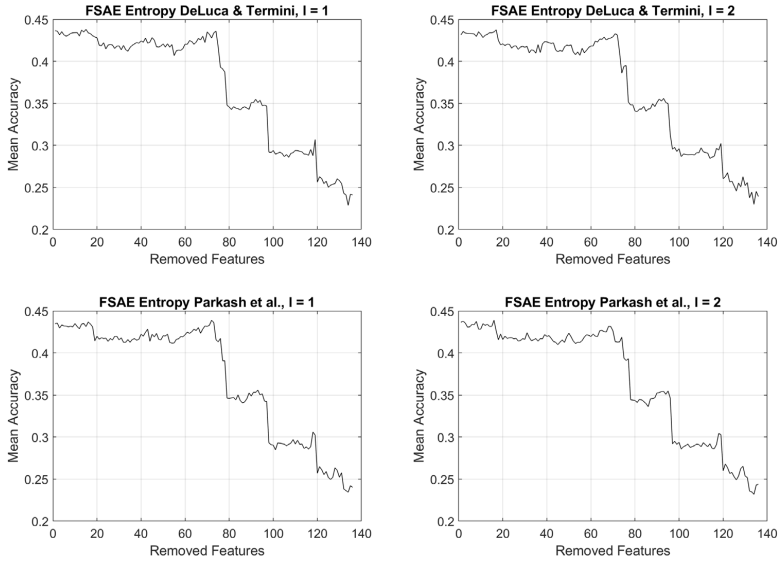


Fig. 3. Results for FSAE setups.

afterwards. In each iteration, and therefore for each feature subset, the data set is split 20 times and for each split, the perturbed training data is used for the FSAE feature ranking and the noise-free testing set with the random forest is used for the evaluation of the performance for the feature subset. The test accuracy is averaged over the 20 runs for a feature subset. After removing the features iteratively according to their ranks and computing the mean accuracy for each of these subsets, we select the feature subset that results in the highest mean accuracy on the test data.

The second step of the training procedure is classification. The feature subset determined previously will be deployed for the classification with the random forest, while four other classification algorithms, the k-nearest neighbour algorithm (KNN; see Cover & Hart, 1967), the naive Bayes classifier (Russell & Norvig, 2009), decision trees (Breiman, Friedman, Stone, & Olshen, 1984) and a similarity classifier (Luukka et al., 2001), will be applied as benchmarks for the classification accuracy. The classifier with the highest classification accuracy will be used as the basis for the evaluation of four different strategies that are conceptualized according to the four classes that were created for the classification problem. These strategies are depicted in detail in the next section. The returns generated with these strategies (after transaction costs) are then determined for the test set and validated with the separate data set for the forecast period. The out-of-sample forecast data set is used with the trading strategies that are premised on the best classification model's predictions in order to validate the performance against a buy-and-hold strategy. All of the calculations are implemented using MATLAB™ software.

4. Results

The results for the different setups of the FSAE filter for feature selection show that most algorithms are capable of identifying features that are redundant or of small relevance for the classification. Table 6 lists the results with the similarity classifier for the choice of the FSAE setup. The best performance accuracy, 44.78%, is accomplished with the entropy measure of De Luca and Termini (1972) and an l-parameter of 1. However, the combination of Parkash

Table 7
FSAE results with random forest and noise.

Perturbation	0 Std.	1 Std.	2 Std.	3 Std.	4 Std.
Remaining features	129	117	98	105	127
Mean accuracy	47.16%	47.51%	47.78%	47.60%	47.76%
Variance (in %)	0.07%	0.07%	0.07%	0.08%	0.08%

Table 8
Comparison of classifiers on the feature subset.

Approach	Setup and parameters	Avg. performance	Variance (in %)
Similarity classifier	$p = 3, m = 1$	44.04%	0.03%
Random forest	Min leaf size = 1	43.63%	0.04%
Random forest	Min leaf size = 10	44.72%	0.03%
KNN	$k = 1$	32.36%	0.04%
KNN	$k = 10$	36.80%	0.05%
Naive Bayes	Normal kernel	38.85%	0.07%
Decision tree	Min leaf size = 1	34.89%	0.04%
Decision tree	Min leaf size = 10	37.47%	0.06%

et al. (2008) entropy and $l = 1$ leads to a comparable accuracy of 44.38%, but with only 64 features instead of 121. Since almost the same performance as the most accurate approach can be achieved with only slightly over half the number of features, the approach with Parkash et al. (2008) entropy and $l = 1$ is more suitable for further analysis.

Fig. 3 shows that, for all FSAE setups, the performance initially decreases, but then has a peak or secondary peak at around 70 removed features. This stresses that all setups work well, but using Parkash et al. (2008) entropy and $l = 1$, which found the best performance at this peak of around 70 removed features, is the most suitable choice of these setups.

In the next step, we conduct the actual feature selection with the classifier random forest (based on 50 decision trees) as the evaluation criterion and the FSAE with Parkash et al. (2008) entropy and $l = 1$, which was the setup selected in the previous step. The feature selection is conducted with and without noise, and the results are displayed in Table 7.

This procedure shows that the most accurate mean accuracy on the test set is achieved with noisy features with two standard deviations. Moreover, this accuracy is achieved on the smallest subset with 98 remaining features. Unlike the initial step with the choice of the FSAE setup, it can be seen that the number of features removed can vary depending on the classification algorithm used for the evaluation, but that their order does not. With the random forest, only 38 features are removed, but the performance of 47.78% is not only higher than the 44.38% after FSAE and the similarity classifier, but also considerably higher than the 44.73% with the similarity classifier and the entire data set.

In what follows, the feature subset that results from the feature selection with the random forest is deployed with different classifiers in order to determine whether the random forest is the most accurate classifier on this feature subset. The results of the comparison are presented in Table 8.

The results demonstrate that the random forest with a minimum leaf size of 10 is the most accurate classifier of the eight classifier setups presented. The minimum leaf size is a parameter that sets the minimum number of observations that must be in a "branch" of a decision tree. A minimum leaf size that is too low may lead to overfitting, since it allows overly complex models, whereas a value for the minimum leaf size that is too high can oversimplify the model and make it unable to capture certain patterns in the data. Both the minimum leaf size parameter (random forest and decision tree) and the k-parameter (KNN) are varied from 10 to 100, by steps of 10. For each of these algorithms, the value of the parameter was chosen based on the highest mean training performance, and the corresponding mean accuracy on the test set is reported. This procedure avoids overfitting the parameters to the test set, since they are chosen based on the training set.

Since the random forest with a minimum leaf size of 10 has been shown to be the most accurate classifier with this feature subset, the performance of this classifier will be examined in more detail, including with respect to the out-of-sample forecast data set. Table 9 presents the classification of this setup on one random split of the training and test data, and on the forecast data. The classification rates of 46.3% and 41.0% for the random test set and the given forecast data set do not seem very high; however, low classification results do not have to mean a low ability to generate excess returns with a strategy based on this classifier (Teixeira & De Oliveira, 2010).

The correct classification rates between the classes range from 28.8% to 62.1% within the test data. Class 1 shows the highest correct classification rate, and is the only class to exceed 60.0%. Clearly, the worst classification result is obtained for Class 3, which addresses 'slightly negative' returns. The returns for the positive classes, Class 1 and Class 2, are the highest in the test set. For the validation dataset, the accuracies range between 23.8% for Class 3 and 55.8% for Class 2. Only for the 'strong positive' Class 1 is the classification accuracy considerably lower than on the test set, at 40.7% compared to 62.1%. Simplifying things by considering the results from the perspective of an investor, who will probably focus mainly on whether the predicted returns are positive or negative, the correct classification rates can differ considerably from those of the single classes. The correct classification rates for positive and negative returns for the data are 75.2% and 35.9%. This is worse than the result for the test data, and indicates a

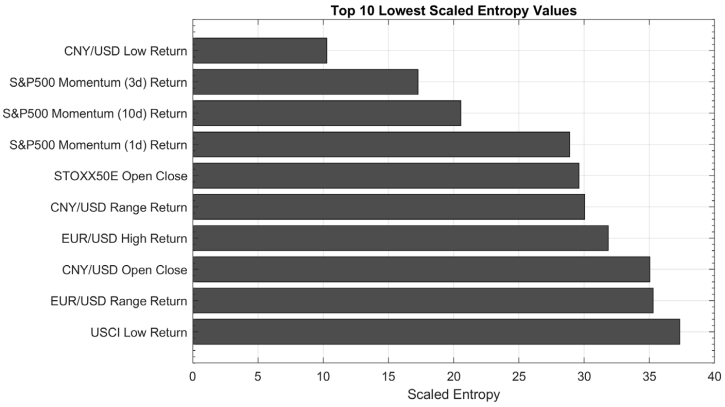


Fig. 4. Top 10 lowest scaled entropy values.

Table 9
Classification results for test and forecast data.

Test	Class 1 62.1%	Class 2 54.6%	Class 3 18.8%	Class 4 48.3%
46.3%	Positive 82.40%		Negative 50.00%	
Forecast	Class 1 40.7%	Class 2 55.8%	Class 3 23.8%	Class 4 41.2%
41.0%	Positive 75.2%		Negative 35.9%	

clearly higher ability of the classifier to predict positive returns correctly. The low negative prediction is burdened by the low classification rate in Class 3, which accounts for more than three times as many observations as Class 4. In conclusion, the classifier for the validation dataset seems to be more accurate at determining positive returns, but performs more poorly at predicting the negative return classes.

We will attempt to obtain a better understanding of the decision-making abilities of the classifier by discussing the features with the lowest scaled entropy values, since low entropy corresponds to high informativity. As Fig. 4 illustrates, the S&P500 Momentum terms, currency exchange rates, the European stock market and the United States Commodity Index (USCI) possess the highest information content. Three of the top four features for predicting the S&P500 open-to-close return are related to the change in momentum. The remaining features consist of the exchange rates USD/Yuan and USD/EUR, the European STOXX 50 open-to-close return and the change in the low value of the USCI.

We examine patterns for a certain type of information that may potentially be relevant for the prediction of the S&P500 open-to-close return by investigating in more detail the results of the variable importance obtained from FSAE with the random forest as the evaluation criterion. The results illustrated in Fig. 5 show that the momentum terms of the S&P500 are characterized by the largest variable importance, and that the currencies on average show a large importance for the classification task related to the S&P500. On the othter hand, the relevance of spreads appears to be low, since the variable importance has a low average value and comparatively low values for the whiskers. The range features show the longest whisker, indicating that some of the range values are relevant for this classification. It is noteworthy that the minimum for the indicators (excluding outliers) is the largest of all technical feature groups except the momentum group. This, together with one of the higher mean values, indicates that technical indicators do possess an elevated relevance for the classification overall.

The analysis of the feature importance grouped by financial market indices as displayed in Fig. 6 reveals that, according to these results, the S&P500 related information (which also includes the momentum terms) is the most relevant group of information for the classification. The groups with the second and third largest mean importance are the Nikkei225 and the STOXX50 group. This appears to be in line with the fact that the STOXX50 open-close return is the only non-S&P500 financial market feature in the top 10 lowest scaled entropies presented earlier.

Since grouping according to the remaining features such as gold, oil or sectors only resulted in lower to medium importance values that could not be distinguished clearly, these results will not be presented or discussed here in detail.

After analyzing the features and the classification results for the four-class problem, we consider various different trading strategies based on this FSAE and the random forest results. We derive four distinct trading strategies, which represent different levels of risk tolerance of the investor (e.g. willingness to "short" (sell) the index) and varying levels of confidence in the model (e.g. willingness to invest solely for strongly positive or negative predictions). The benchmark strategy in this paper is a classic buy-and-hold strategy (passive management), where the

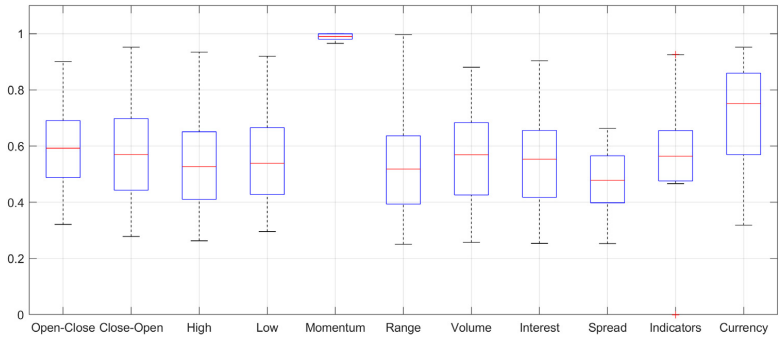


Fig. 5. Feature importance for technical features.

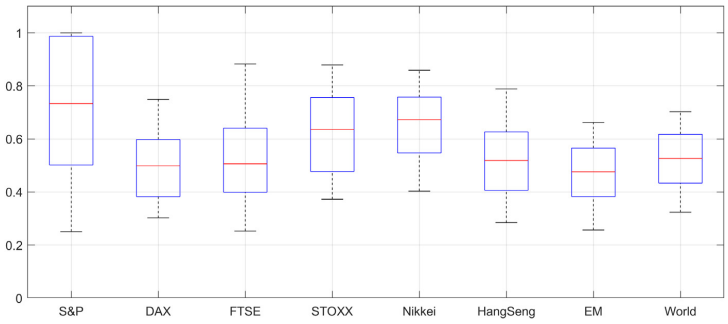


Fig. 6. Feature importance for financial market indices.

Table 10
Investment strategies.

No	Strategy
1	Strongly positive or positive returns predicted (Classes 1 & 2) - Long (buy) the index Strongly negative or negative returns predicted (Classes 3 & 4) - Short (sell) the index *After decision: Remain long or short until next decision requires a change*
2	Strongly positive returns predicted (Class 1) - Long (buy) the index Strongly negative returns predicted (Class 4) - Short (sell) the index *After decision: Remain long or short until next decision requires a change*
3	Strongly positive returns predicted (Class 1) - Long (buy) the index Strongly negative or negative returns predicted (Classes 3 & 4) - Short (sell) the index *After decision: Remain long or short until next decision requires a change*
4	Strongly positive or positive returns predicted (Classes 1 & 2) - Long (buy) the index Strongly negative returns predicted (Class 4) - Short (sell) the index *After decision: Remain long or short until next decision requires a change*
5	Benchmark: buy-and-hold - Long (buy) the index at start of period and retain

index is bought and then held over the respective investment period. All investment strategies are enumerated in Table 10.

It is important to mention that only the index returns are regarded in the subsequent analysis of the returns; no dividends of the underlying stocks are included.

For the comparison to a buy-and-hold strategy, various different levels of transaction costs are considered. Transaction costs can be incorporated either as a percentage of the underlying trade (e.g. Pätäri & Vilska, 2014) or as a fixed amount in a certain currency (e.g. Teixeira & De Oliveira, 2010), and can vary considerably from country to country (Domowitz, Glen, & Madhavan, 2001). In this study, both approaches will be used – a fixed amount in US dollars that can be expected to be paid to a broker in the US, as well as a low percentage of the underlying trade value. Using both approaches ensures that our results can be compared to other existing and future findings more easily. For the

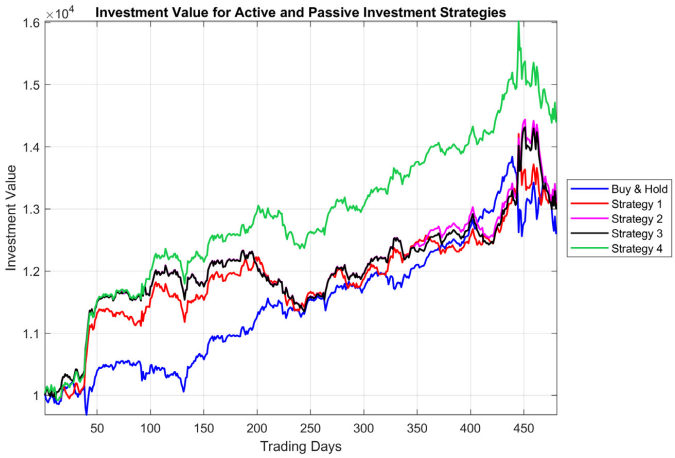


Fig. 7. Performances of trading strategies out-of-sample (with transaction costs).

first approach, the transaction cost varied between $10 and $20, which is within the range of transaction costs that an investor can expect for an order at a broker to purchase a stock or financial instrument (Nasdaq, 2017). The percentage-based transaction costs are set at 0.1%, 0.2%, 03% and 0.4%, to incorporate low to medium levels of transaction costs.

In addition, it is assumed that no capital or gains are withdrawn during the investment period considered. Since the S&P500 open-to-close return is the dependent variable, the results of the strategies assume that the financial instruments used track the S&P500 as closely as possible. This may be by buying all, or at least the majority, of the shares in the proportions that represent the firms in the S&P500, by investing in an exchange traded fund (ETF) that attempts to replicate the behavior of the S&P500, or potentially by using a suitable financial derivative. An ETF is an inexpensive alternative for tracking a certain market or sector, and can also be bought at a broker (Bodie et al., 2009). It is crucial for some of the strategies presented here that ETFs can be sold short (Bodie et al., 2009). If an investor deploys an ETF to follow one of the strategies presented here, it must be taken into account that, in addition to broker costs for the purchase of the ETF, costs are also incurred for the ETF itself. Thus, investors should consider that the performance numbers presented here need to be reduced by any additional cost such as the total expense ratio (TER) of the ETF, which is commonly rather low (Bodie et al., 2009).

The performances of the buy-and-hold strategy and the four strategies considered here are illustrated in Fig. 7 for an investment of $10,000 and transaction costs of 0.1% for the forecast data set.

The graph highlights the fact that all strategies based on the classifier results can outperform the buy-and-hold strategy after transaction costs out-of-sample in the forecast period. The results show that, for transaction costs of 0.1%, investment strategy 4 is characterized by the highest performance of 44.09% for the forecast period (21.09% p.a.), while the buy-and-hold strategy returns 26.13% (12.93% p.a.). With respect to the classification results, this is not surprising. Strategy 4 does not initiate a transaction if the prediction is Class 3, which is the class with by far the lowest classification accuracy. Thus, it leaves out the class that the model predicts worst and that is associated with low absolute returns even when it is classified correctly, since it is the low negative return class. Investment strategy 2, which also does not use the Class 3 predictions, performs well too, but it does worse than strategy 4 since it does not conduct transactions based on Class 2 predictions, though this was the class with the highest prediction accuracy. The performances of the other investment strategies range between 30.16% (14.81% p.a.) and 31.28% (15.33% p.a.). The results show that higher percentage-based transaction costs affect the performances of the trading strategies adversely, since they make using these strategies more expensive. The first strategy, which shows with the largest return without transaction costs, at 60.14% (27.98% p.a.), is affected most by higher transaction costs, since it uses all four class predictions as signals to conduct transactions if possible. Since it therefore conducts two to three times as many transactions as any of the other trading strategies, it underperforms the buy-and-hold strategy even for transaction costs of only 0.2%. The remaining strategies, which initiate trades premised only on a subset of the classes, show fewer transactions and are therefore less sensitive to changes in the transaction costs. For investments of $10.000 and $50.000, the performances again depend on the transaction costs, with the returns logically being higher the lower the value of the transaction costs, as the costs are in proportion to the investment amount.

Table 11
Performances of the investment strategies for all datasets.

		Transactions	1	150	58	60	88
Test (410 Obs.)	–	0	16.88%	311.73%	543.19%	520.14%	322.68%
	–	0.1%	16.83%	254.35%	506.93%	484.01%	287.06%
	–	0.2%	16.83%	204.93%	472.68%	449.95%	254.41%
	–	0.3%	16.83%	162.35%	440.33%	417.85%	224.48%
	–	0.4%	16.83%	125.69%	409.78%	387.59%	197.05%
	10,000	10	16.76%	287.61%	530.36%	506.92%	310.65%
	10,000	20	16.65%	263.50%	517.52%	493.70%	298.62%
	50,000	10	16.86%	306.91%	540.63%	517.50%	320.28%
	50,000	20	16.83%	302.09%	538.06%	514.86%	317.87%
		Transactions	1	201	64	68	89
Forecast (481 Obs.)	–	0	26.39%	60.14%	39.97%	39.33%	57.51%
	–	0.1%	26.13%	30.97%	31.28%	30.16%	44.09%
	–	0.2%	26.13%	7.09%	23.13%	21.60%	31.80%
	–	0.3%	26.13%	−12.46%	15.48%	13.58%	20.55%
	–	0.4%	26.13%	−28.45%	8.30%	6.09%	10.25%
	10,000	10	26.26%	35.22%	32.62%	31.57%	46.59%
	10,000	20	26.13%	10.31%	25.28%	23.81%	35.68%
	50,000	10	26.36%	55.15%	38.50%	37.78%	55.33%
	50,000	20	26.34%	50.17%	37.03%	36.22%	53.14%

Table 11 displays the detailed results for all strategies for the test and forecast periods. It is apparent that the differences between the performances on the test and forecast data sets are considerable.

However, this difference can be explained based on the data and the class imbalance. The test data set possesses a standard deviation of returns of 0.96%, while in the forecast set this value is about one third lower, at 0.63%. The same is true for the standard deviation as a measure of the volatility in the returns that were classified as positive (Classes 1 and 2), with values of 0.84% and 0.54% for the test and forecast data, respectively, as well as for those observations classified as negative (Classes 3 and 4), with values of 0.98% and 0.82%, respectively. Moreover, in both data sets the average returns for observations that are classified as Classes 1 and 4 are particularly distinct. In the test set, the average return for Class 1 predictions is 0.57%, whereas that in the forecast data is only 0.26%, which is not even half. We find comparable results for the average return for Class 4 predictions, with –0.73% on the test set versus –0.20% on the forecast set, which is not even a third of the magnitude. This discrepancy between the results in the test and forecast data sets is due to the fact that the predictions in Classes 1 and 4 contribute most to the returns achieved by any of the trading strategies depicted. The effect is amplified by the fact that the forecast data set is more unbalanced, with fewer observations in Classes 1 and 4, whose correct prediction could boost the returns of the strategy. As a consequence, the difference between the results in the test and forecast sets is considerable, but plausible after further investigation. Moreover, the returns achieved by the trading strategies clearly depend on the market volatility and returns.

The next step is an analysis of the performances of strategies in the forecast period solely for the forecasted open-to-close returns (without transaction costs). Strategy 4 is characterized by average daily returns of 0.076%, while the buy-and-hold strategy performs more than 40% worse, with average daily returns of 0.044%. However, an optimal strategy, namely a strategy that always foresees positive and negative returns correctly and invests accordingly, would achieve a daily return of 0.411% in the forecast period, outperforming any of the presented investment strategies considerably. All average daily returns are presented in Fig. 8. Even though the best investment strategy, strategy 4, clearly underperforms the optimal strategy, it still notably outperforms the passive buy-and-hold strategy.

Fig. 9 highlights the average returns for positive and negative predictions, as well as for returns related to class-specific predictions following the logic of the trading strategy of buying in the case of a Class 1 or Class 2 classification and selling short in the case of a predicted Class 3 or Class 4. The first graph shows that the prediction of the negative classes (Class 3 and Class 4) accounts for a higher average return than the positive classes in the forecast period. Moreover, it stresses that the correct prediction of the positive and negative classes (0.36% and 0.48%) is higher in magnitude than the average return of misclassifying the direct counterparts of these two return directions (–0.33% and –0.37%).

The second graph in Fig. 9 highlights the average class-specific returns overall, in the cases of both correct classification and misclassifications. The graph illustrates two relevant aspects of the classification result. The first is that, for all classes, the returns are higher in magnitude in the case of correct classification than when a misclassification occurs. This contributes to the fact that the average return overall is positive for all classes. In particular, all predicted classes lead to positive returns on average. In simple terms, if the classifier predicts a class for a daily return, following this prediction will lead to a positive return on average – even though the prediction may be incorrect.

The second aspect that this graph highlights is the difference between the average returns for the classes with ‘strong positive’ (Class 1) and ‘strong negative’ (Class 4)

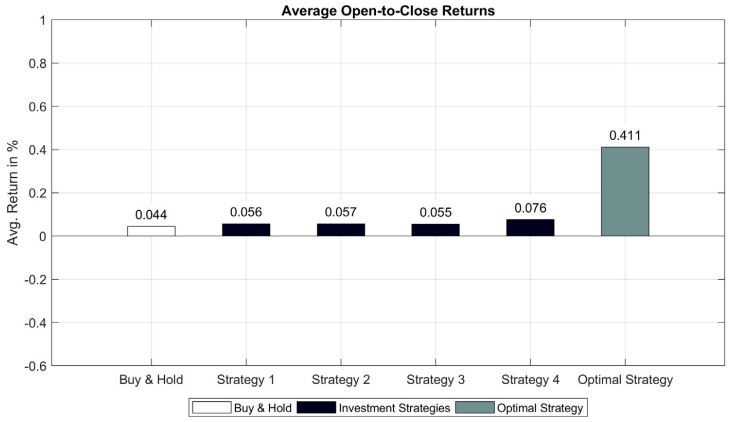


Fig. 8. Average daily open-to-close returns (without transaction costs).

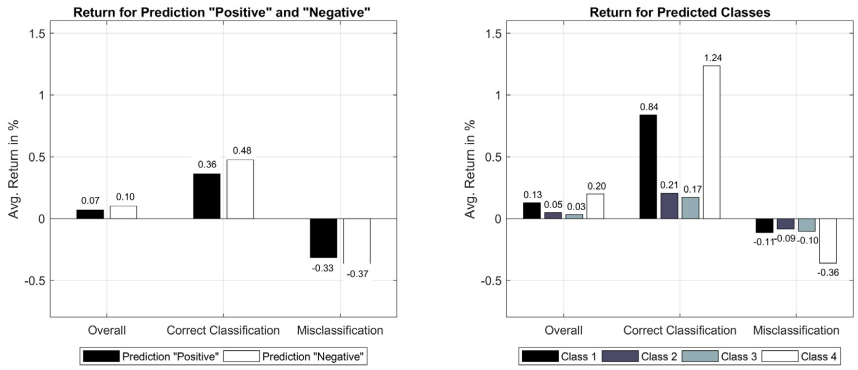


Fig. 9. Performances for all predictions for trading (Strategy 1).

predicted returns, as opposed to those with 'slightly positive' and 'slightly negative' returns (Classes 2 and 3). In the case of correct classifications, the average returns of Classes 1 and 4 (at 0.84% and 1.24%) are considerably higher than those of the two other classes. This result is intuitive, given that correct classification means that the returns for Classes 1 and 4 are respectively larger than 0.5% and smaller than –0.5% (larger than 0.5% for the trading strategy that sells short). The noticeable impact of this higher average return is that the negative return in the case of a misclassification apparently does not entirely compensate for the positive return from a correct classification. This results not only in positive average returns for Classes 1 and 4 overall (independently of whether the classification is correct or not), but also in considerably higher average returns than those for Classes 3 and 4. Predicting 'strong positive' returns (Class 1) leads to more than double the average return of 'slightly positive' returns (Class 2). The pattern is seen even more strongly for negative returns, where the prediction of 'strong negative' returns (Class 4) results in an average return that is more than 6 times that of a classification of 'slightly negative' returns (Class 3). The considerably smaller numbers of observations from Classes 1 and 4 in the forecast data set are part of the reason why the average returns for the positive and negative returns in the first graph are positive but considerably lower than the average returns for these classes.

Overall, this analysis has demonstrated that it is plausible for trading strategies without transaction costs, or even up to a certain level of transaction costs, to result in considerably higher positive returns than the buy-and-hold benchmark strategy. These results indicate that our trading model with FSAE feature selection and a random forest can result in superior returns after transaction costs.

It should be noted, though, that no taxes, slippage costs, or adjustments for risk are considered, and no statement concerning the validity of the efficient market hypothesis can be made without these aspects being included. However, this was not an objective of the current study.

5. Discussion

This paper has classified the S&P500 open-to-close returns (intraday) as a four-class problem that incorporates 'strong positive', 'slightly positive', 'slightly negative' and 'strong negative' return classes. A variety of features from stock markets, related to currencies and commodities and technical indicators, were incorporated for the prediction. With regard to three of the technical indicators, namely the RSI, the MACD and the Bollinger bands, we did not use their values, but transformed them into trading strategies, which were then used as the features. The feature subset for the classification was determined using the FSAE feature ranking method with noise injection in order to make the feature subset selection more robust. The subsequent analysis of the feature importance indicated that the changes in the momentum terms embody the highest level of information for the classification of the S&P500, while currencies rank second overall. Technical indicators such as the moving averages and the trading strategies based on RSI, MACD and Bollinger bands showed only medium to high relevance. In terms of markets, the S&P500-related information, including momentum, shows the highest importance of information from all markets regarding the prediction of the S&P500 returns. The second and third most important markets were the Japanese Nikkei 225 and the European STOXX50. In previous research studies, features (with and without feature extraction and selection) have commonly been used simply for classification, without examining in more detail their feature importance for the class labels/event prediction for the future state of the financial market. In future research, our results for the feature importance must be validated and analyzed with respect to their generalizability to other equity markets.

The classification of the feature subset was conducted using the random forest classifier, and contrasted with the performances of different setups of the KNN algorithm, decision trees, naive Bayes and the similarity classifier. The mean classification accuracy was demonstrated to be the highest for the random forest model. It is noteworthy that the prediction rates for the four classes differ considerably. The subsequent use of four different trading strategies showed that trading based on only a subset of the predicted classes can be more profitable than following all predictions. In particular, using only predictions of Classes 1, 2 and 4 led to the highest return for the forecasting period, since the classification accuracy for Class 3 was clearly the lowest and the average return from using this class prediction was also comparatively low. Another essential finding of this research is that the contributions of the classes to the returns of the trading strategies vary. Predictions on a 'buy' decision for the 'strong positive' Class 1 or a 'sell' for the 'strong negative' Class 4 have overall (correct classifications and misclassifications) multiple times higher returns than those on the 'slightly positive' and 'slightly negative' return classes (Classes 2 and 3). In other words, using the two extreme classes with returns that are far from zero in absolute terms contributes most to the trading strategies on average. Since most previous research has simply used a binary classification problem with only upward and downward movements being considered, this finding can help to shed light on the way in which using more event outcomes for the classification, rather than merely simple upward or downward movements, can improve the benefit of a trading strategy (or a bet on an event). This finding needs to be validated in future research. It will be also of interest to see whether this pattern is observed for the forecasts of other financial markets and whether the prediction of the extreme classes can result in higher average returns or payoffs in other contexts too.

Acknowledgments

This research would like to acknowledge the funding received from the Finnish Strategic Research Council, grant number 313396/MFG40 Manufacturing 4.0.

References

Achelis, S. B. (1995). *Technical analysis from A to Z* (2nd ed.). New York: McGraw-Hill.

Adele, C., Cutler, D. R., & Stevens, J. R. (2012). Random forests. In *Ensemble machine learning: methods and applications* (pp. 157–175). Springer.

Altay, E., & Satman, M. H. (2005). Stock market forecasting: artificial neural network linear regression comparison in an emerging market. *Journal of Financial Management and Analysis*, *18*(2), 18–33.

Baur, D. G. (2012). Asymmetric volatility in the gold market. *Journal of Alternative Investments*, *14*(4), 26–38.

Belis, M., & Guiasu, S. (1968). A quantitative-qualitative measure of information in cybernetic systems. *IEEE Transactions on Information Theory*, *14*(4), 593–594.

Bhaduri, S., & Saraogi, R. (2010). The predictive power of the yield spread in timing the stock market. *Emerging Markets Review*, *11*(3), 261–272.

BlackRock (2017a). iShares MSCI emerging markets ETF. Retrieved May 20, 2017, from https://www.ishares.com/us/products/239637/EEM.

BlackRock (2017b). iShares MSCI Europe financials ETF. Retrieved April 30, 2017, from https://www.ishares.com/us/products/239645/ishares-msci-europe-financials-etf.

Bodie, Z., Kane, A., & Marcus, A. J. (2009). *Investments* (8th ed.). Irwin: McGraw-Hill.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. Wadsworth International Group.

Cao, L., & Tay, F. E. H. (2001). Financial forecasting using support vector machines. *Neurocomputing*, *1*(2), 1–36.

Chang, P. C., & Wu, J. L. (2015). A critical feature extraction by kernel PCA in stock trading model. *Soft Computing*, *19*(5), 1393–1408.

Chicago Board Options Exchange (2017). CBOE interest rate 10 year. Retrieved May 20, 2017, from http://www.cboe.com/delayedquote/advanced-charts?ticker=TNX.

Chun, S. -H., & Park, Y. -J. (2005). Dynamic adaptive ensemble case-based reasoning: application to stock market prediction. *Expert Systems with Applications*, *28*, 435–443.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27.

De Luca, A., & Termini, S. (1972). A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory. *Information and Control*, *20*, 301–312.

Di Lorenzo, R. (2013). *Basic technical analysis of financial markets. A modern approach*. Milan: Springer Italia.

Domowitz, I., Glen, J., & Madhavan, A. (2001). Liquidity, volatility and equity trading costs across countries and over time. *International Finance*, *4*(2), 221–255.

Enke, D., Grauer, M., & Mehdiyev, N. (2011). Stock market prediction with multiple regression, fuzzy type-2 clustering and neural networks. *Procedia Computer Science*, *6*, 201–206.

Fadlalla, A., & Amani, F. (2014). Predicting next day closing price of Qatar Exchange Index using technical indicators and artificial neural network. *Intelligent Systems in Accounting, Finance and Management*, *21*, 209–223.

Fama, E. F. (1965a). Random walks in stock market prices. *Financial Analysts Journal*, *21*(5), 55–59.

Fama, E. F. (1965b). The behavior of stock market prices. *Journal of Business*, *38*(1), 34–105.

Fama, E. F. (1970). Efficient capital markets: a review of theory and empirical work. *Journal of Finance*, *25*(2), 383–417.

Felsen, J. (1975). Learning pattern recognition techniques applied to stock market forecasting. *IEEE Transactions on Systems, Man and Cybernetics*, *5*(6), 583–594.

Gokmenoglu, K. K., & Fazlollahi, N. (2015). The interactions among gold, oil, and stock market: evidence from S & P500. *Procedia Economics and Finance*, *25*, 478–488.

Guerard Jr., J. B. (2013). *Introduction to financial forecasting in investment analysis*. New York: Springer ScienceBusiness Media.

Guo, Z., Wang, H., Liu, Q., & Yang, J. (2014). Fusion based forecasting model for financial time series. *PLoS ONE*, *9*(6), 1–13.

Guo-qiang, X. (2011). The optimization of share price prediction model based on support vector machine. In *international conference on control, automation and systems engineering* (pp. 1–4).

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. In *Springer Series in Statistics*: *Vol. 2*.

Hurwitz, E., & Marwala, T. (2011). Suitability of using technical indicator-based strategies as potential strategies within intelligent trading systems. In *IEEE international conference on systems, man, and cybernetics* (pp. 80–84).

Karymshakov, K., & Abdykaparov, Y. (2012). Forecasting stock index movement with artificial neural networks: the case of İstanbul stock exchange. *Trakya University Journal of Social Science*, *14*(2), 231–242.

Khaidem, L., Saha, S., & Dey, S. R. (2016). Predicting the direction of stock market prices using random forest. *Applied Mathematical Finance*, *1*(5), 1–20.

Kim, K. (2003). Financial time series forecasting using support vector machines. *Neurocomputing*, *55*(1–2), 307–319.

Kim, M. J., Han, I., & Lee, K. C. (2004). Hybrid knowledge integration using the fuzzy genetic algorithm: prediction of the Korea Stock Index. *Intelligent Systems in Accounting, Finance and Management*, *12*, 43–60.

Kim, M. -J., Min, S. -H., & Han, I. (2006). An evolutionary approach to the combination of multiple classifiers to predict a stock price index. *Expert Systems with Applications*, *31*(2), 241–247.

Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: statistical arbitrage on the S & P 500. *European Journal of Operational Research*, *259*, 689–702.

Krollner, B., Vanstone, B., & Finnie, G. (2010). Financial time series forecasting with machine learning techniques: a survey. In *European symposium on artificial neural networks: Computational and machine learning* (pp. 25–30).

Leigh, W., Purvis, R., & Ragusa, J. M. (2002). Forecasting the nyse composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support. *Decision Support Systems*, *32*, 361–377.

Lendasse, A., De Bodt, E., Wertz, V., & Verleysen, M. (2000). Non-linear financial time series forecasting – application to the Bel 20 stock market index. *European Journal of Economic and Social Systems*, *14*(1), 81–91.

Leung, M. T., Daouk, H., & Chen, A. -S. (2000). Forecasting stock indices: a comparison of classification and level estimation models. *International Journal of Forecasting*, *16*(2), 173–190.

Lohrmann, C., Luukka, P., Jablonska-Sabuka, M., & Kauranne, T. (2018). Supervised feature selection with a combination of fuzzy similarity measures and fuzzy entropy measures. *Expert Systems with Applications*, *110*, 216–236.

Lu, C. -J., & Wu, J. -Y. (2011). An efficient CMAC neural network for stock index forecasting. *Expert Systems with Applications*, *38*(12), 15194–15201.

Luukka, P. (2007). Similarity classifier using similarity measure derived from Yu's norms in classification of medical data sets. *Computers in Biology and Medicine*, *37*, 1133–1140.

Luukka, P. (2011). Feature selection using fuzzy entropy measures with similarity classifier. *Expert Systems with Applications*, *38*, 4600–4607.

Luukka, P., Saastamoinen, K., & Könönen, V. (2001). A classifier based on the maximal fuzzy similarity in the generalized Lukasiewicz-structure. In *10th IEEE international conference on fuzzy systems*.

Matsuoka, K. (1992). Noise injection into inputs in back-propagation-learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *22*(3), 436–440.

Morningstar (2017). Vanguard total world stock index fund ETF. Retrieved from http://www.morningstar.co.uk/uk/etf/snapshot/snapshot.aspx?id=0P0000G5T2.

Nasdaq (2017). Start investing with only $1, 000. Retrieved April 30, 2017, from http://www.nasdaq.com/investing/start-investing-1000stm.

Niaki, S. T. A., & Hoseinzade, S. (2013). Forecasting S & P 500 index using artificial neural networks and design of experiments. *Journal of Industrial Engineering International*, *9*(1), 1–9.

Nyberg, H. (2013). Predicting bear and bull stock markets with dynamic binary time series models. *Journal of Banking and Finance*, *37*(9), 3351–3363.

O'Connor, N., & Madden, M. G. (2006). A neural network approach to predicting stock exchange movements using external factors. *Knowledge-Based Systems*, *19*(5), 371–378.

Özesmi, S. L., Tan, C. O., & Özesmi, U. (2006). Methodological issues in building, training, and testing artificial neural networks in ecological applications. *Ecological Modelling*, *195*(1–2), 83–93.

Page, L. (2012). "It ain't over till it's over." Yogi Berra bias on prediction markets. *Applied Economics*, *44*(1), 81–92.

Parkash, O., Sharma, P., & Mahajan, R. (2008). New measures of weighted fuzzy entropy and their applications for the study of maximum weighted fuzzy entropy principle. *Information Sciences*, *178*, 2389–2395.

Pätäri, E., & Vilska, M. (2014). Performance of moving average trading strategies over varying stock market conditions: the finnish evidence. *Applied Economics*, *46*(24), 2851–2872.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, *42*(1), 259–268.

Rossilo, R., Giner, J., & de la Fuente, D. (2014). The effectiveness of the combined use of VIX and support vector machines on the prediction of S & P 500. *Neural Computing and Applications*, *25*, 321–332.

Rothschild, D., & Pennock, D. M. (2014). The extent of price misalignment in prediction markets. *Algorithmic Finance*, *3*(1–2), 3–20.

Rudebusch, G. D., & Williams, J. C. (2009). Forecasting recessions: The puzzle of the enduring power of the yield curve. *Journal of Business & Economic Statistics*, *27*(4), 492–503.

Russell, S., & Norvig, P. (2009). *Artificial intelligence: A modern approach* (3rd ed.). Prentice Hall.

S & P Dow Jones Indices (2017). S & P GSCI CRUDE OIL. Retrieved April 10, 2017, from https://us.spindices.com/indices/commodities/sp-gsci-crude-oil.

State Street Global Advisors (SPDR) (2017a). SPDR S & P US financials select sector UCITS ETF. Retrieved May 5, 2017, from https://fi.spdrs.com/en/professional/etf/spdr-sp-us-financials-select-sector-ucits-etf-ZPDF-GY?cid=1365706.

State Street Global Advisors (SPDR) (2017b). SPDR S & P US materials select sector UCITS ETF. Retrieved May 5, 2017, from https://fi.spdrs.com/en/professional/etf/SPDR-SP-US-Materials-Select-Sector-UCITS-ETF-ZPDM-GY.

Teixeira, L. A., & De Oliveira, A. L. I. (2010). A method for automatic stock trading combining technical analysis and nearest neighbor classification. *Expert Systems with Applications*, *37*(10), 6885–6890.

USCF (2017). United states commodity index fund. Retrieved May 5, 2017, from http://www.uscfinvestments.com/usci.

Vaughan Williams, L., & Reade, J. J. (2016). Prediction markets, social media and information efficiency. *Kyklos*, *69*(3), 518–556.

Wolfers, J., & Zitzewitz, E. (2004). Prediction markets. *Journal of Economic Perspectives*, *18*(2), 107–126.

Yahoo Finance (2017). Selected time series. Retrieved from https://finance.yahoo.com/.

Yao, Y. Y., Wong, S. K., & Butz, C. J. (1999). On information-theoretic measures of attribute importance. In *PacificAsia conference on knowledge discovery and data mining* (pp. 133–137).

Zhang, J., Cui, S., Xu, Y., Li, Q., & Li, T. (2018). A novel data-driven stock price trend prediction system. *Expert Systems with Applications*, *97*, 60–69.

Zhong, X., & Enke, D. (2017). Forecasting daily stock market return using dimensionality reduction. *Expert Systems with Applications*, *67*, 126–139.

Zhora, D. V. (2005). Data preprocessing for stock market forecasting using random subspace classifier network. In *Proceedings of international joint conference on neural networks, Montreal, Canada* (pp. 2549–2554).

Christoph Lohrmann received the M.Sc. degree with Distinction in Banking and Financial Management in 2016 from the Institute for Financial Services, University of Liechtenstein, Liechtenstein. He is currently Ph.D. student in Computational Engineering with the School of Engineering Science, Lappeenranta University of Technology. His research interests include data analysis, classification, feature selection, decision making and financial markets.

Pasi Luukka received the M.Sc. degree in 1999 from the Department of Information Technology, Lappeenranta University, Finland, where he also received the D.Sc. degree in Applied Mathematics in 2005 from the Department of Mathematics and Physics. He is currently Full Professor with the School of Business and Management, Lappeenranta University of Technology. His research interests include fuzzy data analysis, classification, feature selection and fuzzy decision making.

Publication III

Lohrmann, C., Luukka, P.
A novel similarity classifier with multiple ideal vectors based on k-means clustering

Reprinted with permission from
Decision Support Systems
Vol. 111, pp. 27-37, 2018

Decision Support Systems 111 (2018) 27–37

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

A novel similarity classifier with multiple ideal vectors based on k-means clustering

Christoph Lohrmann*, Pasi Luukka

Lappeenranta University of Technology, School of Engineering Science, Skinnarilankatu 34, 53850 Lappeenranta, Finland

ARTICLE INFO

Keywords:
Supervised classification
Jump method
Principal component analysis
MAP test
Parallel Analysis

ABSTRACT

In the literature, researchers and practitioners can find a manifold of algorithms to perform a classification task. The similarity classifier is one of the more recently suggested classification algorithms. In this paper, we suggest a novel similarity classifier with multiple ideal vectors per class that are generated with k-means clustering in combination with the jump method. Two approaches for pre-processing, via simple standardization and via principal component analysis in combination with the MAP test and Parallel Analysis, are presented. On the artificial data sets, the novel classifier with standardization and with transformation power Y = 1 for the jump method results in significantly higher mean classification accuracies than the standard classifier. The results of the artificial data sets demonstrate that in contrast to the standard similarity classifier, the novel approach has the ability to cope with more complex data structures. For the real-world credit data sets, the novel similarity classifier with standardization and Y = 1 achieves competitive results or even outperforms the k-nearest neighbour classifier, the Naive Bayes algorithm, decision trees, random forests and the standard similarity classifier.

1. Introduction

1.1. Background

One common type of problem in machine learning is classification, which means using characteristics of observations to assign these observations to discrete classes [4]. Classification algorithms support the decision-making for enterprises and individuals in numerous applications, including medical diagnostics [28], product positioning [26], recommendation systems [21] and sentiment analysis in social media [13]. A common interest in these algorithms in finance is with respect to the evaluation of the creditworthiness of customers and for the credit granting decision [13,19,42].

In the literature, researchers and practitioners can find a manifold of algorithms to conduct a classification tasks, which include, but are not limited to, the well-known neural networks, support vector machines, decision trees, k-nearest neighbours, random forests and numerous more. One of the more recently developed and applied classifiers is the similarity classifier [33]. The first results for the similarity classifier were published in Luukka et al. [33]. Since then, the classifier has been applied to several medical data sets [28,32] and to two bankruptcy data sets [30], showing high classification accuracies. Moreover, Luukka & Leppälampi [32] demonstrated that the similarity classifier outperforms classifiers such as linear discriminant analysis, the C4.5 algorithm [36] and multi-layer perceptron neural networks on the medical data sets in their study. Luukka [29] even deployed the classifier on linguistic statements that were transformed into fuzzy numbers. Overall, the advantages of the similarity classifier are that it is comparably computationally inexpensive and requires only a small amount of observations to achieve high classification results [28].

The similarity classifier is premised on the idea to represent each class in the data by one so-called ideal vector, which can be, for instance, determined with a generalized mean. Each ideal vector is essentially a point in the feature space and the class assignment is conducted based on the highest similarity of an observation with one of these points that represent the classes. The idea of similarity is closely related to the concept of distance [14] and the similarity classifier can be regarded as a distance-based technique. Luukka & Lampinen [31] pointed out that distance-based techniques may face difficulties to classify complex data structures. Hence, Luukka & Lampinen [31] introduced the differential evolution based multiple vector prototype classifier (MVDE). Their approach included defining multiple vectors that represent each class. This approach demonstrated to be able to handle data structures for which a simple distance-based technique was not sufficient [31]. However, Luukka & Lampinen [31] highlighted that the choice of the number of vectors per each class is pivotal for the

* Corresponding author.
E-mail addresses: christoph.lohrmann@student.lut.fi (C. Lohrmann), pasi.luukka@lut.fi (P. Luukka).

https://doi.org/10.1016/j.dss.2018.04.003
Received 19 October 2017; Received in revised form 23 February 2018; Accepted 19 April 2018
Available online 24 April 2018
0167-9236/

accuracy of the classifier performance. The reason behind this is that too few ideal vectors per class may not be sufficient to appropriately capture the data complexity while too many will result in overfitting. As a final remark, these authors stated that a subject for future research is to optimize for a given data structure a suitable number of representative class vectors.

1.2. Objectives

In this paper, the idea of using multiple representatives, as presented in Luukka & Lampinen [31] in the context of their MVDE classifier, will be transferred to the context of the similarity classifier. The aim is to define a novel similarity classifier that uses multiple ideal vectors for the classification. This should enable to classify more complex data structure, including those that are characterized by multiple decision regions for each class, better than the standard similarity classifier as presented in Luukka et al. [33]. As a second contribution, the authors in this paper clearly address the research need mentioned by Luukka & Lampinen [31] to provide a framework for the choice of the number of representatives of a class, which is in case of the similarity classifier the number of ideal vectors. The number and position of these ideal vectors is pivotal for the classification since the distance-based classifier's ability to capture complex data structures but at the same time not to overfit the data depends on it. In this paper, a novel approach for the similarity classifier will be presented, where k-means clustering in combination with the jump method is conducted to determine suitable multiple deal vectors for each class. The multiple ideal vectors are then used within the similarity classifier to assign class labels to observations. The novel similarity classifier aims to overcome the problem of classifying observations with complex data structures.

In particular, we will illustrate the inability of the standard similarity classifier to cope with more complex data structures with artificial data sets and contrast its result to the novel similarity classifier.

The remaining paper is structured as follows: in Section 2 the methods deployed for the novel similarity classifier approach will be introduced and the artificial and real-world data sets will be depicted, on which the standard and the novel classifier are applied in order to compare their performances. Moreover, the training procedure for the classifiers will be described. In Section 3, the results of the comparison will be presented, which will subsequently be discussed in detail in Section 4.

2. Methods

2.1. K-means clustering

Clustering in general is concerned with finding clusters that encompass observations that are similar to one another and dissimilar to those observations in other clusters [11]. In other words, observations in a cluster have small inter-point distances in relation to the distance to observations in other clusters [4]. The k-means clustering algorithm is one of the first and widely applied hard clustering algorithms [11,24]. The process behind k-means clustering is rather simple. Initially, one observation for each cluster is chosen randomly and used as the centroid for the initial cluster [11]. In an iterative procedure, each observation is assigned first to the nearest cluster and, second, the cluster centre is adjusted to represent all observations in the cluster [4]. The assignment of an observation i to the cluster with the closest cluster centre can be expressed as [4,12]:

$$u_{ij} = \begin{cases} 1 \text{ if } \|x_i - \mu_j\|^2 < \|x_i - \mu_{j'}\|^2 \text{ for all } j' \neq j \\ 0 \quad \text{Otherwise} \end{cases} \tag{1}$$

For the second step, the centre of the closest cluster is adapted for the new additional observation. A cluster centre μ_j is updated as [4]:

$$\mu_j = \frac{\sum_i u_{ij} x_i}{\sum_i u_{ij}} \tag{2}$$

The objective function that will be minimized with respect to the membership coefficients and cluster centres is [4,24]:

$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} u_{ij} x_i - \mu_j^2 \tag{3}$$

This function represents the sum of squared distances of each observation to its cluster centre [4].

2.2. Jump method

An essential aspect of the k-means algorithm is that the data is partitioned into K clusters. However, K, the number of clusters, has to be specified in order to conduct the clustering and the choice of K is nontrivial [4]. The problem for choosing K arises from the fact that the total squared distance, which is commonly used in the evaluation of a clustering, will always prefer more clusters to less. Therefore, using this way of evaluating clusters will end up choosing as many clusters as observations are available [47]. In the literature, many approaches to determine a suitable number of clusters can be found. These include the 'Elbow method', the 'Gap statistic' [40], the 'Jump method' [39] and the 'Calinski-Harabasz index' [6]. For the novel similarity classifier, the k-means with the jump method is chosen since this approach is theoretically motivated, applicable for a wide range of problems and mixture distributions, and even performs well when clusters are overlapping to a large extent [39]. The 'jump method' developed by Sugar & Gareth [39] is related to rate distortion theory. Distortion is a measure for the dispersion within clusters [11]. The minimum distortion d_k obtainable with K cluster centres is [39]:

$$d_K = \frac{1}{p} \min_{c_1,\ldots,c_K} E[(X - c_X)^T \Gamma^{-1} (X - c_X)] \tag{4}$$

where X is a p-dimensional random variable with a mixture distribution with G components and covariance matrix Γ for X. In addition, $c_1, \ldots, c_K$ are the candidates for the K cluster centres, c_X is the cluster centre that is closest to X, and T indicates the transpose. For the use in practice, the distortion d_K can be estimated based on the minimum distortion $\hat{d}_K$ obtained in the k-means clustering [39]. The covariance matrix Γ might in practice not be known. However, Sugar & Gareth [39] stress that the identity matrix can be used as a simplification, which makes $\hat{d}_K$ the mean squared error. They deployed this approach and found it to be robust concerning the shape of the distortion curve for different covariance matrices [39]. Consequently, the minimum distortions can easily be obtained given the observations and K clusters. The 'jump method' can be stated as follows [39]:

1. Conduct k-means clustering with a different number of clusters from 1 to K and determine the values for $\hat{d}_K$, the distortions that correspond to the number of clusters
2. Choose the parameter called 'transformation power' denoted by Y, where $Y > 0$, which is required for the calculation of the 'jumps' in the next step. A common choice is $Y = p/2$
3. Transform the distortions with the transformation power Y by computing $\hat{d}_K^{-Y}$. Calculate 'jumps' as $J_K = \hat{d}_K^{-Y} - \hat{d}_{K-1}^{-Y}$, which is the difference between the transformed distortions of k-means clustering with K clusters compared to K-1 clusters
4. Determine the estimated number of clusters denoted K^* as the k corresponding to the largest 'jump', which is the maximum J_K. In order to be able to obtain as a result $K = 1$, the distortion for no clusters is defined as $\hat{d}_0^{-Y} = 0$

The choice of the Y parameter, the transformation power, is no straight forward. For uncorrelated features and Gaussian clusters, Sugar

& Gareth [39] suggest choosing $Y = p/2$, where p is the number of dimensions of the data. However, features are often correlated to a certain extent and do not need to be in Gaussian clusters. If it is impractical to analyse the cluster distribution, Sugar & Gareth [39] recommend to either use a relatively low value for the transformation power Y (e.g. 1 or even lower) or to determine Y with the help of the 'effective' dimension of the data set. In this paper, two approaches will be considered, selecting Y premised on the 'effective' dimensionality or simply setting it to 1.

2.3. 'Effective dimensionality'

In an example, Sugar & Gareth [39] explain that the effective dimensionality of a data set is lower than the dimensionality of the feature space if there are features that are highly correlated. In this paper, we will use principal component analysis to transform the data into uncorrelated principal components [1,20]. We will keep only a subset of all principal components, since the first principal components are often enough to represent the overall data set and its variance well [11]. Since the new features are uncorrelated, their effective dimension should be equal to their dimension. Yet, the choice of how many of the principal components should be retained is not trivial [35,43]. Extracting too few principal components will result in a loss of information while extracting too numerous principal components might include irrelevant information or noise [7,51].

In the literature, various methods to determine a suitable number of principal components can be found [7]. These methods include, but are not limited to, the modified broken stick model [7], the Guttman-Kaiser criterion [17,22], the SCREE test [8], the Minimum Partial Average (MAP) test [43], Bartlett's test [2] and Parallel Analysis [18]. Of these methods, the MAP and Parallel Analysis demonstrated the highest performance across different data complexities [35,50,51]. The minimum average partial (MAP) test is based on conducting a PCA and subsequently analyse the matrix of partial correlations [7,35,43,50]. The idea behind this procedure is that the average squared partial correlation will decline until a 'unique' component would be removed [43,50]. Therefore, the stopping point is reached at the minimum average squared partial correlation [44,50]. According to Velicer [43] the method results in an exact stopping point for the selection of principal components. Velicer et al. [44] find that the average of the partial correlations to the fourth power outperforms the initial approach with average squared partial correlations for continuous data. A disadvantage of the MAP is that it can in certain situations underestimate the number or principal components to select [50].

The second highly recommended approach is Parallel Analysis developed by Horn [18] [44,50]. It is based on the criticism that the proof for another well-known approach for choosing the number of principal components, the Guttman-Kaiser criterion (also referred to as K1 rule), is concerned with population statistics and, therefore, not applicable for samples [15,18]. Essentially, Parallel Analysis is concerned with finding those principal components that account for a larger amount of variance than a counterpart based on random data [35]. An alternative approach for Parallel Analysis is to deploy an upper percentile (commonly the 95th) for the distribution of the eigenvalues as explained by Glorfeld [16]. Using this approach decreases the tendency of Parallel Analysis to extract too numerous components [16].

MAP and Parallel Analysis usually lead to the selection of the same principal components to retain [35]. However, since results may differ, applying both approaches is beneficial since MAP and Parallel Analysis complement each other given that the first may extract too few components and the second too many [35,51].

2.4. Novel classification algorithm

The idea of the similarity classifier originates in fuzzy theory. Fuzzy theory is based on the idea that a number of non-mathematical properties cannot be reflected by crisp sets since they solely indicate whether a certain property is present or not [23]. In contrast to that, fuzzy sets reflect a membership degree to a class or property [49]. Using membership degrees allows to model partial memberships. This is of interest for classification since it allows partial membership of an observation to classes [33]. As a consequence, the similarity measure can be used as a classifier using the partial membership values of an observation to classes in order to assign an observation to the class it is most similar to. This type of classification is referred to as supervised classification since the class label of observations is known [4,45].

The similarity classifier presented by Luukka et al. [33] was premised on the idea to compute for each class one so-called 'ideal vector' that is supposed the represent that class well. There are several ways of computing ideal vectors, of which arithmetic mean is one of the earliest methods used. To classify an observation, it is compared to the ideal vector of each class and, eventually, a similarity value is calculated. The similarity embodies a membership degree for an observation to a class. The class assignment is then simply conducted based on the highest similarity, meaning that the observation is assigned to the class for which is shows the highest membership degree (between [0,1]). For more details, please see Luukka et al. [33] and Luukka [28].

The novel similarity classifier algorithm is premised on the idea to represent each class by multiple ideal vectors. The k-means clustering algorithm with the jump method will be deployed in order to determine the ideal vectors per class and gives a clear answer to the question how many ideal vectors per class should be constructed. An observation is then assigned to the class that corresponds to the ideal vector it is closest to. This approach seems suitable in case that classes have one or more decision regions that can be represented by one or more clusters. It should be even adequate when the clusters representing the decision regions overlap since using the jump method has demonstrated to perform well even when clusters overlap to a large extent [39]. The idea is related to the K-Nearest Neighbour (KNN) algorithm but attempts to be more robust for classification by finding the nearest cluster instead of the nearest neighbours to conduct the class assignment. Opposed to KNN, it is not necessary to define the number of nearest neighbours/clusters, since the nearest cluster aims at representing the nearest region where observations of a class are located.

The novel algorithm can be characterized by several distinct steps, which are illustrated in a flow chart in Fig. 1 and depicted in detail subsequently.

Step 1: Data pre-processing. Before the novel similarity classifier algorithm is applied, the input data require pre-processing. We examined two different setups: one based on simple standardization to the compact interval [0,1] and using the original features of the data set, and, a second one, based on normalization of the raw data, so that that they follow a standard normal distribution and using PCA to extract new features from the existing features in the data. For the second approach, a combination of MPA and PA can be used to select a suitable number of principal components, as recommended by O'Connor [35], and subsequently standardize them to the compact interval [0,1] in order for the similarity classifier to be applicable.

Step 2: Division of the data set. The available data is divided into a training set and a test set via the hold-out method (e.g. 70% training samples and 30% test samples).

Step 3: Conduct k-means clustering for each class. For the training data, the k-means clustering is performed for each class. The clustering is performed for each suggested number of clusters from 1 to K, where K is a user-specified number. For each number of clusters K, the average distortion over the observations x_i from $i = 1$ to N is estimated as:

$$\widehat{d}_K = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\frac{u_{ik}{}^{*}(x_i - c_k)^T\Gamma^{-1}(x_i - c_k)}{p} \tag{5}$$

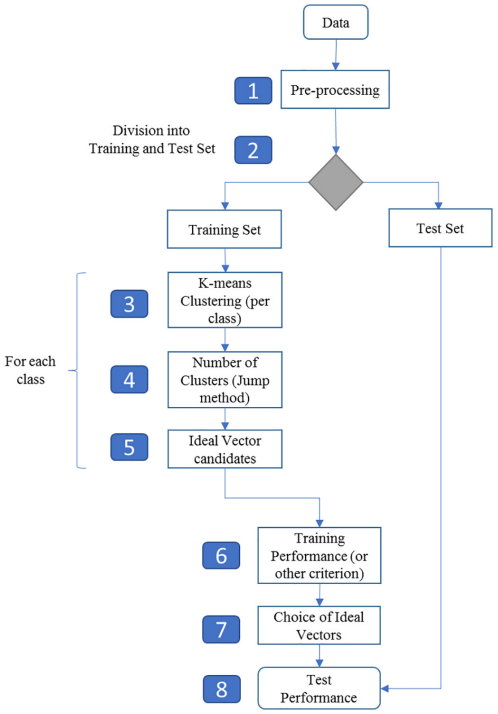


Fig. 1. Flowchart of the similarity classifier with multiple ideal vectors.

where u_{ik} shows the membership of an observation x_i to cluster c_k, which takes for the cluster with the closest cluster centre the value 1 and otherwise 0:

$$u_{ik} = \begin{cases} 1 \; if \; \|x_i - c_k\|^2 < \|x_i - c_{k'}\|^2 \; for \; all \; k' \neq k \\ 0 \quad Otherwise \end{cases} \tag{6}$$

This notation differs in certain elements from the one presented above from Sugar & Gareth [39]. First, the minimization of the distortion with respect to the cluster centres for a given K is conducted already in the k-means algorithm, so that is not present in this formula any more. Second, we use the membership to a cluster in our formula and include all clusters in it since it appeared more straight-forward for the implementation then using c_X for the notation as the closest cluster centre. The x_i denotes a p-dimensional observation and Γ is the covariance matrix for X, the data set, but can for reasons of simplicity be the identity matrix, as explained before. The cluster centre candidates are denoted $c_1, \ldots, c_K$ and T indicates the transpose. The distortion estimate $\hat{d}_K$ is obtained by summing for each observation x_i over the cluster centres from 1 to K and then summing over the observations themselves and taking the average over the observations. The outcome for the class-specific clustering is a distortion vector with each element being a value of $\hat{d}_K$ corresponding to a specific number of clusters K.

Step 4: Determine optimal number of clusters for each class. For the jump method, the transformation power Y is then used in the exponent of the distortions $\hat{d}_K$ to obtain $\hat{d}_K^{-Y}$. Afterwards, the 'jumps', meaning the differences between subsequent values of these transformed distortions $\hat{d}_K^{-Y}$, are calculated as:

$$J_K = \hat{d}_K^{-Y} - \hat{d}_{K-1}^{-Y} \tag{7}$$

where J_K is the jump between the distortions of using K and K-1 clusters on the training data. The number of clusters where the maximum jump J_K can be observed, is the candidate for the optimal number of clusters. The cluster centres that correspond to the candidate for the optimal number of clusters is recorded/saved. To choose the optimal number of clusters for each class, the k-means clustering (Step 3) and the Jump method (Step 4) are repeated n times (e.g. $n = 10$). This eventuates in n candidates for the optimal number of clusters for the class. The number of clusters for a class is then chosen as the most frequent candidate number of clusters suggested (mode-value).

Step 5: Record ideal vector candidates. For future steps, the cluster centres of all of the n repetitions of Step 3 and Step 4 that also led to the optimal number of clusters are recorded/saved. Therefore, for each class, there are one or more sets of ideal vector candidates.

Step 6: Training with ideal vector candidates. For each class, a randomly selected set of ideal vector candidates from those saved in the previous step is chosen and they are used together for the similarity classifier. The calculations in this step correspond to a large extent to those of the original similarity classifier with the difference that each set of ideal vector candidates contains multiple ideal vectors. First, for each feature d the similarity between each ideal vector candidate v_o and each sample (vector), for simplicity of the index notation denoted x instead of x_i, of the training set is calculated as:

$$S(x_d, v_{o,d}) = \sqrt[p]{1 - |x_d^p - v_{o,d}^p|} \tag{8}$$

where x_d denotes the d-th element of the vector of observation x and $v_{o,d}$ is the d-th element of the ideal vector v_o. Moreover, p is a parameter for the similarity that is in the most basic case set to 1. Afterwards, the generalized mean from this similarity vector is computed by summing over all features d and then dividing by the number of features denoted by D to obtain the similarity of the observation x with the entire ideal vector candidate v_o:

$$S(x, v_o) = \left(\frac{1}{D} \sum_{d=1}^{D} S(x_d, v_{o,d})^m \right)^{\frac{1}{m}} \tag{9}$$

where m is a parameter for the applied mean function and $S(x, v_o)$ represents the scalar similarity value of the observation x with the ideal vector v_o. This is repeated for all ideal vectors to obtain for the observation x the similarity with all clusters (for all classes). Finally, observation x is assigned to a cluster based on the highest similarity value that the observation has with the ideal vector (candidate) of a cluster:

$$Cl(x) = \arg\max_{o=1,\ldots,O} S(x, v_o) \tag{10}$$

Since the cluster to which x is assigned, belongs to one of the classes, the observation is assigned to the corresponding class. This can be formally expressed as a simple mapping from the cluster Cl of the observation x to the class C:

$$C(x) = f(Cl(x)) \tag{11}$$

Repeating these calculations of Step 6 for each observation gives all the predicted class labels. These are compared to the target class labels in the training data set and the classification accuracy (or another evaluation criterion) is calculated. The evaluation criterion can be specified by the user, for instance also the False-Positive-Rate (FPR) or the False-Negative-Rate (FNR) on the training set can be chosen as evaluation criterion. For the given combination of sets of ideal vector candidates for each class, this evaluation criterion is computed. The calculations in this step are repeated (e.g. 50 times) and for each run a different combination of sets of ideal vectors are used and the value for the evaluation criterion and the corresponding ideal vector candidates (for all classes) are recorded.

Step 7: Choice of ideal vectors. The combination of sets of ideal vector candidates that resulted in the best value for the evaluation criterion for the training set, e.g. the highest performance, are chosen as the ideal vectors for the similarity classifier. This allows to customize the choice of ideal vectors to the evaluation criterion. The authors suggest for instance to choose the ideal vectors to maximise the mean accuracy or minimize the False-Negative-Rate or False-Positive-Rate, depending on the application and objective.

Step 8: Calculation of the test set performance. The ideal vectors obtained from the previous Step 7 are deployed with the similarity classifier on the test data set from Step 2. The calculation of the similarities, the assignment of classes and of the performance are conducted with the formulas (8) to (11) from Step 6.

2.5. Data

For this paper, three artificial data sets are generated to investigate the difference between the original and novel similarity classifier approaches. In addition to that, three real-world data sets were obtained from the UCI Machine Learning Repository [27] to compare the performance of these approaches with other well-known supervised classification algorithms.

The three artificially composed data sets are all characterized by multiple decision regions for each class. This setup is supposed to demonstrate the novel similarity classifier's ability to use multiple ideal vectors to cope with more complex decision regions than the original similarity classifier using only a single ideal vector. Moreover, the performance with different pre-processing and Y parameters is investigated. The specific features for each of the three artificial datasets A, B and C is depicted in Table 1.

The first data set, Case A, normally distributed features with small variations are generated that form two-dimensional clusters for each class. In this data set small overlap of classes is present, but the feature space can almost distinctly be divided into the multiple decision regions for each class. The second artificial data set, referred to as Case B, is related to the binary XOR problem with the three-dimensional feature space being divided into two distinct decision regions for each class (overall 4 decision regions). The last case, Case C, is characterized by a three-dimensional feature space for a 4-class classification problem. For each class, there exist two clusters, one cluster with small variation in the data and the other with moderate variation. None of the clusters shows an overlap with another cluster of the same or another class. All features are scaled into the compact interval [0,1]. The three artificial data sets are plotted in Fig. 2.

Table 1
Characteristics of the three artificial data sets.

Cases	Observations	Class	Feature 1	Feature 2	Feature 3
Case A	900	1	N(1,0.2)	N(1,0.2)	–
		1	N(2,0.2)	N(2,0.2)	–
		1	N(3,0.2)	N(3,0.2)	–
		2	N(3,0.2)	N(2,0.2)	–
		2	N(2,0.2)	N(1,0.2)	–
		2	N(1,0.2)	N(3,0.2)	–
		3	N(3,0.2)	N(1,0.2)	–
		3	N(1,0.2)	N(2,0.2)	–
		3	N(2,0.2)	N(3,0.2)	–
Case B	1000	1	[0, 0.5)	[0, 0.5)	[0, 1]
		1	[0.5, 1]	[0.5, 1]	[0, 1]
		2	[0, 0.5)	[0.5, 1]	[0, 1]
		2	[0.5, 1]	[0, 0.5)	[0, 1]
Case C	1000	1	N(2,0.1)	N(2,0.1)	N(2,0.1)
		1	N(6,0.5)	N(6,0.5)	N(6,0.5)
		2	N(6,0.1)	N(6,0.1)	N(2,0.1)
		2	N(2,0.5)	N(2,0.5)	N(6,0.5)
		3	N(2,0.5)	N(6,0.5)	N(2,0.5)
		3	N(6,0.1)	N(2,0.1)	N(6,0.1)
		4	N(6,0.5)	N(2,0.5)	N(2,0.5)
		4	N(2,0.1)	N(6,0.1)	N(6,0.1)

The real-world data sets discussed in this paper are all related to the approval and quality of credit borrowers. It seemed reasonable to use these data sets since we assumed that distinct decision regions for good and bad applicants exist and that they can be characterized rather well in form of multiple clusters. Moreover, the class imbalance that is common for many credit default/approval problems, meaning that one class can be considerably larger than the other, is assumed to be more effectively addressed with a classifier based on clusters than e.g. simply based on nearest neighbours. However, we want to remark, that our selection for real-world data sets is by no means exhaustive and knowing in advance in what real-world data sets this is useful is not possible.

The subject of credit approval is essential for financial institutes since they require approaches to support the decision-making for loan applications as well as for the ongoing monitoring of the financial situation of their clients [42,46]. The credit granting decision copes with the risk of granting credits to not suitable applicants and the non-acceptance of credits for solvent clients [25]. The classification of clients is particularly important since a credit scoring that is conducted effectively will most likely lead to savings in the future [48].

The first credit data set is available at the UCI Machine Learning Repository as 'Credit Approval Data Set'. It is listed as a 'Financial' data set and neither the date of donation nor the author is known. The data set contains 690 observations of 15 features related to credit card applications. Six features are continuous. The remaining attribute values in the data set have been adjusted to meaningless symbols by the donor. We changed these symbols for the similarity classifier into discrete integer values. The class label is binary and indicates whether a credit was granted to a client or if the credit proposal was rejected. The features characterize the client and represent properties of the credit decision. The 'Credit Approval' data set contains missing values, which have been removed for this study, which leaves 653 complete observations for the classification task.

The second real-world data set is the numeric version of the 'Statlog (German Credit Data) Data Set'. The original data set was donated by Professor Dr. Hans Hofmann in 1994 and adjusted by Strathclyde University by changing categorical features into numeric integer-valued ones. This data set encompasses 1000 observations with 24 numeric features. It does not contain any missing values. The data are characterized by two classes, which represent the evaluation if a person is a good or bad credit-taker and the 24 features embody characteristics of the credit borrower. In contrast to all other data set, the imbalance in this data set was high with 70% belonging to the first group and 30% to the second.

The third and last real-world data set is the 'Statlog (Australian Credit Approval) Data Set', which is an adapted form of the 'Credit Approval Data Set'. Neither the donor nor the date of donation for this data set is known. This financial data set is also related to credit card applications. The data set contains 690 observations without missing values. The 14 features, of which 6 are continuous and 8 are discrete, represent the characteristics of a credit applicant. The binary class label indicates whether the credit decision was positive or negative.

2.6. Data pre-processing and training process

As mentioned above, one out of two approaches for the pre-processing in this paper is based on principal component analysis and choosing a suitable number of principal components as new features. For the choice of the number of principal components, Parallel Analysis (with 1000 random data sets) as the upper bound for the number of components, and MAP will be used. If the result differs between MAP [44] and PA, it will be investigated whether the MAP decision was 'close'. Since the authors did not find a specification for what constitutes a 'close call' [35], it is defined as an increase of the average

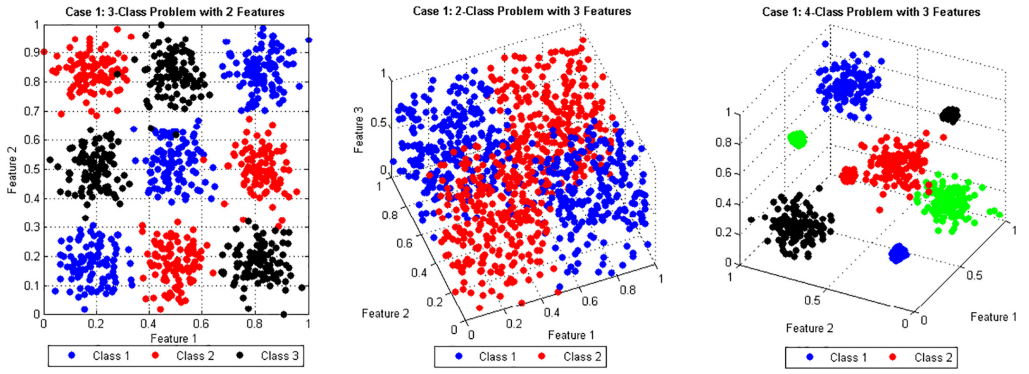


Fig. 2. Artificial data sets.

partial correlations per step of < 70% points. The reasoning behind this choice is that in the regarded cases in this paper, changes per additional component that showed a difference of up to 70% points appeared small compared to larger changes that were characterized by increases of at least 100% for an additional component. Therefore, the 70% points threshold for an additional component appears to be justified.

For the k-means clustering, we suggest *K*, the maximum number of clusters that k-means is performed with, to be set as the maximum of, first, 10 clusters and, second, of the number of observations contained in the smallest class divided by 20. This should ensure that the number of clusters *K* is only set larger than 10 if on average 20 or more observations will be contained in a cluster. If the data set is small or the minority class(es) encompass few observations, the minimum number of clusters might have to be reduced below 10 to avoid a potential overfit. On the other hand, if the data set is large, the average number of observations per cluster to allow additional clusters can be set higher to capture all pattern contained in the data.

For the classification, the data is divided with the holdout method and using stratified sampling. For all algorithms in this paper the observations were split into 70% training data and the remaining 30% for testing. For all classifiers, despite the standard and novel similarity classifiers, 1000 iterations are performed during the training of the classifiers.

For the standard and novel similarity classifiers, the entire algorithm is run for different combinations of the p- (varied from 1 to 8) and m-parameter (varied from 1 to 6) to find the values for p and m with which highest mean accuracy for the given dataset can be reached. This is referred to as 'optimal value search' and for each combination of p and m, 100 iterations of the algorithm are performed before the mean performances are computed. In general, conducting the optimal value search increases the number of required computations to improve the mean classification accuracy. In order to avoid increasing the computational complexity notably, 100 iterations are conducted with optimal value search as opposed to 1000 iterations for the remaining classification algorithms.

For the novel similarity classifier, the number of clusterings *n* (in Step 3 and 4 of the algorithm) was set to 10 and the random combinations for the ideal vector candidates was chosen to be 50 (Step 6 of the algorithm).

For the simplified artificial data sets with known structure using the standard parameters p = 1 (parameter for similarity) and m = 1 (parameter for the generalized mean) for all similarity classifiers is sufficient, since the data structures are simple enough to find very good solutions without an optimal value search. For the real-world data sets, the performance of the novel and original similarity classifiers is compared to the K-nearest neighbour algorithm [10], the Naive Bayes classifier [38], decision trees [37] and the ensemble learning algorithm called random forest [5]. All calculations are implemented with the MATLAB™- software. The code for the MAP and PA are based on Matlab-files provided by O'Connor [34].

3. Results

3.1. Results for the artificial data sets

First, the results for the artificial datasets are presented in Table 2. For the first artificial dataset, Case A, the standard classifier in the three-class problem shows a mean accuracy of only 32.47%. In contrast to that, the mean accuracy of the novel similarity classifiers is 96.97% and 96.87% respectively. It has to be stressed, that for the two-dimensional Case A transformation power Y = p/2 is also equal 1 (since p is in the context of the jump method the dimensionality). Consequently, the results in Case A are for both novel classifiers essentially equal. Using the one-sided version of the Welch's test with unequal variance to test whether one population mean is larger than another, the means for the novel similarity classifier with both transformation powers are highly significantly larger than that of the standard classifier (with > 99.99% confidence). Clearly, for the two remaining 3-dimensional cases, Y = p/2 and Y = 1 do not take the same values. For Case B the highest result is accomplished with the novel similarity classifier with Y = 1 with 96.53%. For the novel similarity classifier with transformation power of Y = p/2 the mean accuracy is 90.49%, which is highly significantly lower than that of the same classifier with Y = 1. However, only the standard classifier reaches a mean accuracy of close to 50%. On account of this, the novel similarity classifiers both perform highly significantly better on Case B than the original similarity classifier with a single ideal vector. For the last data set, Case C, the

Table 2
Performance for artificial data sets with standard parameters.

Data set	Similarity	Mean accuracy	Variance	runs	Y
Case A	Standard	0.3247	0.0037	100	–
Case A	Novel	0.9697	0.0001	100	p/2
Case A	Novel	0.9687	0.0001	100	1
Case B	Standard	0.5142	0.0006	100	–
Case B	Novel	0.9049	0.0009	100	p/2
Case B	Novel	0.9653	0.0004	100	1
Case C	Standard	0.3493	0.0081	100	–
Case C	Novel	1	0	100	p/2
Case C	Novel	1	0	100	1

Table 3
Performance for artificial data sets after optimal value search.

Data set	Similarity	PC	Mean accuracy	Variance	Y
Case A	Standard-PCA	2	0.3116	0.0039	–
Case A	Novel - PCA	2	0.9912	0.0000	p/2
Case A	Novel - PCA	2	0.9913	0.0000	1
Case B	Standard-PCA	3	0.4824	0.0005	–
Case B	Novel - PCA	3	0.9028	0.0011	p/2
Case B	Novel - PCA	3	0.9613	0.0001	1
Case C	Standard-PCA	3	0.3135	0.0085	–
Case C	Novel - PCA	3	1	0	p/2
Case C	Novel - PCA	3	1	0	1

standard similarity classifier reaches for the four-class problem a mean accuracy of 34.93% while the novel similarity classifier with Y = 1 and Y = p/2 accomplishes a 100% performance on the non-overlapping class clusters of the data set. This result is once again highly significant compared to the standard classifier (with > 99.99% confidence).

The performance of the novel similarity classifier and the standard similarity classifier are also tested with the suggested pre-processing with PCA. The mean accuracies obtained with this pre-processing are highlighted in Table 3. The magnitude of the performances for the artificial data sets is comparable with those without PCA. For Case 3 both transformation powers for the novel classifier eventuate in a 100% mean accuracy. Overall, the results for all artificial data sets show that the novel similarity classifier with and without PCA as pre-processing clearly outperforms the standard similarity classifier with the mean accuracy being in all cases highly significantly larger (with > 99.99% confidence). However, the difference in the performances with the two transformation powers Y can be significant, as was observed for Case B. Overall, these artificially created classification problems clearly show the advantage of the proposed novel method compared to the standard similarity classifier.

The results for the artificial data sets are calculated only for the default parameters for the similarity of p = 1 and m = 1, since the results of the novel similarity classifier are already high and only a marginal improvement could be expected for these data sets.

3.2. Results for the real-world data sets

In this next step, the results of the real-world credit data sets achieved with the similarity classifiers and different pre-processing methods are presented and compared with the performances of the KNN algorithm, the Naive Bayes classifier, decision trees and random forests on these credit data sets.

The performance of all classifiers on the first real-world data sets, the ‘Credit Approval’ data set, is highlighted in Table 4. The first seven classifiers presented there are the standard and novel similarity classifier with different pre-processing methods. The remaining 9 classifiers are different setups for the remaining benchmark algorithms. For KNN the result on the test set with a single nearest neighbour, the 10 nearest neighbours and for the optimal number k are displayed. To obtain the optimal number for *k*, the KNN algorithm was run for all *k* from 1 to the training sample size and the result on the test data set for the setup leading to the best mean accuracy on the training data set was chosen and is displayed in the table. For the Naive Bayes classifier two setups were used: the first assumed normal Gaussian distributions, the second used a kernel with normal smoothing. The random forest is composed of 50 decision trees and is implemented in the first setup with minimum leafsize of 1. The second setup displays the mean performance on the test data set based on the minimum leafsize (from 10 to 100 by steps of 10) that showed the highest mean training performance. The same procedure was deployed for the two decision tree setups. The different leafsizes are tried since too small leafsizes may incorporate noise and harm the generalization ability while too large leafsizes can result in a classifier that only captures the broadest patterns.

For the ‘Credit Approval’ data set, the highest performance of 87.33% is reached with the ensemble learning algorithm random decision forest with minimum leafsize = 1. This performance is closely followed by the random decision forest with minimum leafsize = 10 with mean accuracy 87.08% and the novel similarity classifier with transformation power Y = 1 leading to mean performance of 87.06%. Three aspects of this result are noteworthy. First, the novel similarity classifier with Y = 1 achieves a performance that is competitive to the one of the ensemble learning algorithm, random forest, and possesses the highest mean accuracy for all classifiers that are based on a single learning algorithm. Second, using the Welch's test (with unequal variances), it can be demonstrated that the mean accuracy accomplished with the novel similarity classifier with Y = 1 is highly significantly larger than that of the standard similarity classifier (p-value < 0.001). Thirdly, in comparison with all other single learning algorithm-based classifiers, the mean accuracy of the novel similarity classifier with Y = 1 shows a highly significant positive difference in the mean performance. The classifier mean accuracies with the 4 selected principal components (PCs) in the pre-processing are between 5.42% to 8.09% lower than its direct counterpart without PCA and only standardized initial features.

In credit scoring and for the evaluation of credit applications, the consequences of misclassification are unequal. Consequently, it appears suitable to evaluate the classifiers' performances also with respect to the False-Negative-Rate (FNR) and the False-Positive-Rate (FPR). For all real-world data sets, the FNR represents the proportion of falsely rejected customers to the sum of falsely rejected customers and the rightfully accepted customers. In other words, it is the share of customers that is falsely classified as bad compared to all customers that are actually good. Opposed to that, the FPR is the proportion of falsely accepted customers to the sum of falsely accepted customers and the rightfully rejected ones. The FPR is with respect to credit decisions more relevant than the FNR. In particular, classifying a bad customer falsely as a good one and giving him/her a credit that may not be repaid (as focused on by FPR) outweighs the potential forgone profit of assigning a good customer to the bad customer class (as emphasized by FNR) [3,9,41].

Since the FPR is of additional relevance for credit scoring, for each real-world data set one novel similarity classifier was customized in the choice of ideal vectors with respect to the FNR rate. This classifier is referred to as ‘Novel Similarity Classifier (Minimize FPR)’. The lowest FPR rate for the ‘Credit Approval’ data set of 7.2% is achieved for the standard similarity classifier. On the other hand, the FNR for this setup belongs with 19.6% to one of the higher rates and is above the mean and median of all classifiers. The FNR of the novel similarity classifier with Y = 1 is with 5.6% one of the lowest, while the FPR with 19.0% is above the median of all algorithms. Comparing FPR and FNR stressed that the novel similarity classifier with Y = 1 performs very well with respect to avoiding allocating good customers in the ‘bad’ class and foregoing profits but worse than the average in recognizing customers that should not be assigned to the ‘good’ class and, therefore, avoiding credit default. The ‘Novel Similarity Classifier (Minimize FPR)’ with Y = 1 leads to a slight improvement of the FPR from 19.0% to 15.5% compared to the novel similarity classifier with Y = 1 that was customized with respect to the mean accuracy. This improvement in FPR was accomplished as a trade-off to the mean accuracy. However, for this data set the ensemble learner random forest still achieved a better FPR and at the same time a higher classification accuracy. The result of the optimal parameter value search for the ‘Credit Approval’ data set with the novel similarity classifier with Y = 1 is illustrated in Fig. 3.

The surface for the mean accuracy for the novel similarity classifier appears smooth and high accuracies are achieved and seem robust with respect to several different setups of the p and m parameter.

The classification performances for the ‘German Credit’ data set are presented in Table 5. The best mean accuracy for the ‘German Credit’

Table 4
Results for the 'Credit Approval' data set (the highest mean accuracy, the lowest FNR and the lowest FPR are highlighted in bold).

Classification algorithm	Mean accuracy	Variance	Mean FNR	Mean FPR	p	m	Y
Standard similarity classifier	0.8599	0.0004	0.196	**0.072**	6	4	–
Novel similarity classifier	0.8525	0.0005	0.133	0.160	1	1	p/2
Novel similarity classifier	0.8706	0.0005	0.056	0.190	6	5	1
Novel similarity classifier (minimize FPR)	0.8615	0.0003	0.119	0.155	2	6	1
Standard similarity classifier (PCA, 4 PCs)	0.8057	0.0005	**0.010**	0.284	1	1	–
Novel similarity classifier (PCA, 4 PCs)	0.7716	0.0008	0.243	0.216	1	2	p/2
Novel similarity classifier (PCA, 4 PCs)	0.8076	0.0005	0.324	0.085	4	4	1
K-nearest neighbours, k = 1	0.8184	0.0005	0.207	0.161	–	–	–
K-nearest neighbours, k = 10	0.8608	0.0004	0.142	0.137	–	–	–
K-nearest neighbours, best k = 1	0.8184	0.0005	0.207	0.161	–	–	–
Naive Bayes (normal Gaussian distribution)	0.8039	0.0006	0.321	0.093	–	–	–
Naive Bayes (kernel with normal smoothing)	0.6823	0.0012	0.425	0.230	–	–	–
Random decision forest (min leafsize = 1)	**0.8733**	0.0004	0.129	0.125	–	–	–
Random decision forest (min leafsize = 10)	0.8708	0.0004	0.126	0.132	–	–	–
Decision tree (min leafsize = 1)	0.8322	0.0007	0.194	0.147	–	–	–
Decision tree (min leafsize = 10)	0.8561	0.0005	0.157	0.133	–	–	–

data set of 75.84% is again reached with the random forest algorithm. Notwithstanding, the highest classification accuracies of single classifier algorithms is once more accomplished with the novel similarity classifier with Y = 1. Compared to the remaining single classifier algorithms, the novel similarity classifier's mean classification accuracy is highly significant with the single exception of the standard similarity classifier based on 8 PCs. Notably, the performance of the standard similarity classifier with and without PCA belongs to the best mean accuracies for all algorithms on this data set. However, the novel similarity classifier's mean accuracy is significantly larger than that of the standard similarity classifier (p-value = 0.0193).

For the 'German credit' data set, the novel similarity classifier with Y = 1, eventuates in a FPR of 67.4%, which is in absolute terms high but compared to all other algorithms not far from the mean FPR. For the FNR, this classifier ends up with a value of 9.5%, which belongs to the better results for FNR, being well below the median value. The most accurate classifiers, the random forests, show FPR values of 58.1% and 66.8%. The tendency of most algorithms to result in high FP rates and lower FN rates appears to be the consequence of the high class imbalance with the positive class being with 70% the apparent majority. However, the novel similarity classifier that was customized to result in lower FPR values shows the opposite behaviour, being with a low FPR of 17% good at avoiding to give credits to 'bad' customers while with 53.5% FNR being worse at not giving credits to 'good' customers. Given that the FPR for credit decisions is of higher relevance, this algorithm seems very suitable to reduce potential losses. The result of the optimal parameter value search for the 'German Credit' data set of the novel similarity classifier with Y = 1, is illustrated in Fig. 4. It again shows a rather stable and smooth surface for the mean classification depending on the p and m parameter showing that good classification performances can be reached with different setups of these parameters.

The classification results on the third real-world data set, the 'Australian Credit' data set, are presented in Table 6. The highest mean accuracy on the 'Australian Credit' data set is 87.37%, which is achieved with the novel similarity classifier with transformation power Y = 1. The performance of the standard similarity classifier with 87.27% embodies the second highest mean accuracy. It is remarkable, that the mean performance of the novel similarity classifier with Y = 1 not only exceeds the mean performance of the ensemble learner random forest, but this difference is even highly significant. On top of that, the mean accuracy of the novel similarity classifier with Y = 1 is highly significantly larger than that of almost all other algorithms - the KNN classifiers, decision trees, random decision forests, Naive Bayes and the novel similarity classifiers – with the sole exception of the standard similarity classifier.

The lowest FPR rate for the 'Credit Approval' data set of 9.0% is accomplished with the novel similarity classifier that was customized to result in low FPR rates. Also, this algorithm still leads to a performance that is competitive or higher than that of the KNN algorithms, the Naive Bayes and the decision trees. The novel similarity classifier with Y = 1, the best performing algorithm on this data set, is with a FPR of 13.3% still below the average FPR rate of all classifiers. The FPR of the random forests is with 12.4% and 12.0% in magnitude comparable to that of the novel similarity classifier with Y = 1. The result of the optimal parameter value search for the 'Australian Credit' data set and the novel similarity classifier with Y = 1 is illustrated in Fig. 5.

Overall, the novel similarity classifier achieved for all artificial data sets superior classification results to the standard similarity classifier

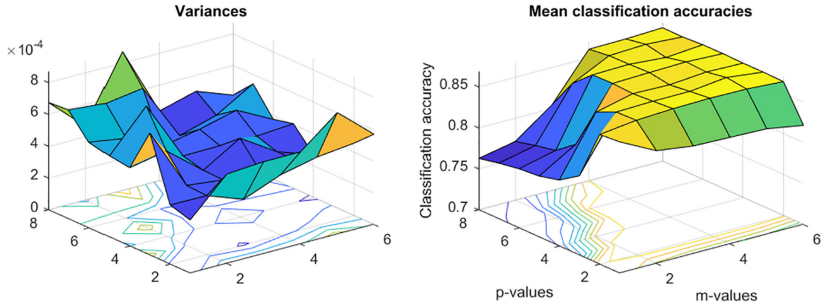


Fig. 3. Optimal value search for the novel similarity classifier with Y = 1 ('Credit Approval' data set).

Table 5
Results for the 'German Credit' data set (the highest mean accuracy, the lowest FNR and the lowest FPR are highlighted in bold).

Classification algorithm	Mean Accuracy	Variance	Mean FNR	Mean FPR	p	m	Y
Standard similarity classifier	0.7263	0.0003	0.099	0.683	4	1	–
Novel similarity classifier	0.6822	0.0005	0.158	0.691	8	1	p/2
Novel similarity classifier	0.7314	0.0003	0.095	0.674	4	1	1
Novel similarity classifier (minimize FPR)	0.5750	0.0012	0.535	**0.170**	2	6	1
Standard similarity classifier (PCA, 8 PCs)	0.7299	0.0004	0.142	0.570	3	5	–
Novel similarity classifier (PCA, 8 PCs)	0.6966	0.0008	0.281	0.355	3	1	p/2
Novel similarity classifier (PCA, 8 PCs)	0.6998	0.0006	0.298	0.304	2	1	1
K-nearest neighbours, k = 1	0.6715	0.0005	0.237	0.543	–	–	–
K-nearest neighbours, k = 10	0.7164	0.0005	0.162	0.568	–	–	–
K-nearest neighbours, best k = 1	0.6715	0.0005	0.237	0.543	–	–	–
Naive Bayes (normal Gaussian distribution)	0.7233	0.0006	0.229	0.388	–	–	–
Naive Bayes (kernel with normal smoothing)	0.7068	0.0001	**0.013**	0.947	–	–	–
Random decision forest (min leafsize = 1)	**0.7584**	0.0003	0.096	0.581	–	–	–
Random decision forest (min leafsize = 10)	0.7516	0.0003	0.069	0.668	–	–	–
Decision tree (min leafsize = 1)	0.6946	0.0007	0.218	0.510	–	–	–
Decision tree (min leafsize = 10)	0.7197	0.0006	0.167	0.545	–	–	–

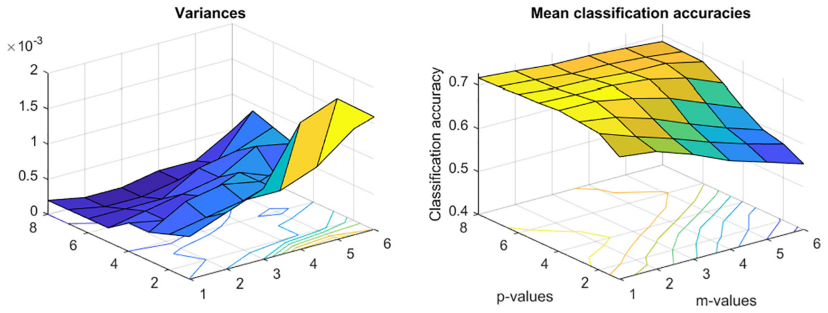


Fig. 4. Optimal value search for the novel similarity classifier with Y = 1 ('German Credit' data set).

with only a single ideal vector per class. For the real-world data sets, the novel similarity classifier with Y = 1 was performing at least as accurate as the standard classifier, in two data sets it was significantly more accurate than the standard similarity classifier, in one of them the difference was even highly significant. Compared to the remaining benchmark algorithms, the novel similarity classifier showed in most cases competitive result, often even outperforming the benchmark classifiers.

4. Discussion

In this paper, the authors designed a novel similarity classifier based on k-means clustering. The k-means clustering is deployed in combination with the jump method to determine the number of clusters and also the cluster centres themselves for each class. These clusters are then used as the multiple ideal vectors for each class in the similarity classifier. It is also possible to a certain extent to customize the classifier

Table 6
Results for the 'Australian Credit' data set (the highest mean accuracy, the lowest FNR and the lowest FPR are highlighted in bold).

Classification algorithm	Mean accuracy	Variance	Mean FNR	Mean FPR	p	m	Y
Standard similarity classifier	0.8727	0.0004	0.144	0.114	3	3	–
Novel similarity classifier	0.8469	0.0005	0.151	0.155	1	1	p/2
Novel similarity classifier	**0.8737**	0.0004	**0.118**	0.133	2	3	1
Novel similarity classifier (minimize FPR)	0.8478	0.0005	0.229	**0.090**	2	6	1
Standard similarity classifier (PCA, 3 PCs)	0.8283	0.0005	0.268	0.094	1	2	–
Novel similarity classifier (PCA, 3 PCs)	0.7940	0.0006	0.273	0.152	1	3	p/2
Novel similarity classifier (PCA, 3 PCs)	0.8273	0.0004	0.228	0.128	1	1	1
K-nearest neighbours, k = 1	0.7997	0.0005	0.223	0.182	–	–	–
K-nearest neighbours, k = 10	0.8513	0.0004	0.177	0.126	–	–	–
K-nearest neighbours, best k = 1	0.7997	0.0005	0.223	0.182	–	–	–
Naive Bayes (normal Gaussian distribution)	0.8016	0.0005	0.329	0.093	–	–	–
Naive Bayes (kernel with normal smoothing)	0.6877	0.0015	0.417	0.228	–	–	–
Random decision forest (min leafsize = 1)	0.8676	0.0004	0.143	0.124	–	–	–
Random decision forest (min leafsize = 10)	0.8653	0.0004	0.153	0.120	–	–	–
Decision tree (min leafsize = 1)	0.8307	0.0006	0.194	0.149	–	–	–
Decision tree (min leafsize = 10)	0.8483	0.0005	0.164	0.142	–	–	–

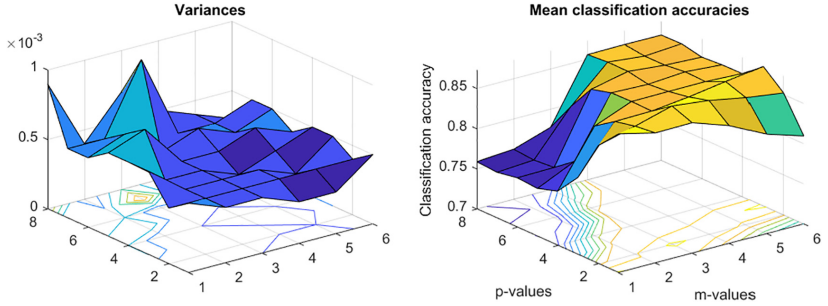


Fig. 5. Optimal value search for the novel similarity classifier with Y = 1 ('Australian Credit' data).

by the choice of the evaluation criterion during the training to focus on the mean accuracy, the False-Positive-Rate (FPR) or another metric. In this research, two methods for pre-processing and for the choice of the transformation power Y are proposed. The first one is premised on a simple standardization to [0,1] and using simple transformation power Y = 1. This method led on the artificial and real-world data sets in most cases to the highest performance accuracy. The second approach based on the 'effective dimensionality' eventuated in the majority of cases in lower mean accuracies than the first method. Therefore, the authors suggest, premised on the observed results, to use the novel similarity classifier on standardized data with transformation power Y = 1 since it showed superior results compared to the standard similarity classifier. On the real-world data sets, the novel similarity classifier with transformation power Y set to 1 achieved in most cases competitive mean accuracies and on the Australian Credit Data set even the highest mean accuracy. Except for the ensemble learning technique random forest, the novel similarity classifier with Y = 1 was often significantly or highly significantly more accurate than the benchmark algorithms in this study. Moreover, the novel similarity classifier customized to achieve small FPR reached comparably low FPR values, in two out of three cases even accomplishing the lowest FPR of all algorithms. Finally, a future research need is a systematic analysis of the transformation power for the novel similarity classifier for different data sets.

References

[1] H. Abdi, L.J. Williams, Principal component analysis, Wiley Interdisciplinary Reviews: Computational Statistics 2 (2010) 433–459.
[2] M.S. Bartlett, Tests of significance in factor analysis, British Journal of Statistical Psychology 3 (2) (1950) 77–85.
[3] V.L. Berardi, G.Q. Zhang, The effect of misclassification costs on neural network classifiers, Decision Sciences Institute, 1997 Annual Meeting, Proceedings, Vols. 1–3, 30(3) 1997, pp. 364–366.
[4] C.M. Bishop, Pattern Recognition and Machine Learning, Springer ScienceBusiness Media, New York, 2006.
[5] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32.
[6] T. Calinski, J. Harabasz, A dendrite method for cluster analysis, Communications in Statistics - Theory and Methods 3 (1) (1974) 1–27.
[7] R. Cangelosi, A. Goriely, Component retention in principal component analysis with application to cDNA microarray data, Biology Direct 2 (2007) 2.
[8] R.B. Cattell, The scree test for the number of factors, Multivariate Behavioral Research 1 (2) (1966) 245–276.
[9] C.-L. Chuang, S.-T. Huang, A hybrid neural network approach for credit scoring, Expert Systems 28 (2) (2011) 185–196.
[10] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1) (1967) 21–27.
[11] G. Dougherty, Pattern Recognition and Classification: An Introduction, Springer ScienceBusiness Media, New York, 2013.
[12] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, John Wiley, Section, New York, 2000.
[13] S. Figini, F. Bonelli, E. Giovannini, Solvency prediction for small and medium enterprises in banking, Decision Support Systems 102 (2017) 91–97.
[14] F. Formato, G. Gerla, L. Scarpati, Fuzzy subgroups and similarities, Soft Computing 3 (1999) 1–6.
[15] L.E. Garrido, F.J. Abad, V. Ponsoda, A new look at Horn's parallel analysis with ordinal variables, Psychological Methods 18 (4) (2013) 454–474.
[16] L.W. Glorfeld, An improvement on Horn's parallel analysis methodology for selecting the correct number of factors to retain, Educational and Psychological Measurement 55 (3) (1995) 377–393.
[17] L. Guttman, Some necessary conditions for common-factor analysis, Psychometrika 19 (2) (1954) 149–161.
[18] J.L. Horn, A rationale and test for the number of factors in factor analysis, Psychometrika 30 (2) (1965) 179–185.
[19] Z. Huang, H. Chen, C.-J. Hsu, W.-H. Chen, S. Wu, Credit rating analysis with support vector machines and neural networks: a market comparative study, Decision Support Systems 37 (4) (2004) 543–558.
[20] J.E. Jackson, A User's Guide to Principal Components, John Wiley & Sons, Inc., 1991.
[21] Y. Jiang, J. Shang, Y. Liu, Maximizing customer satisfaction through an online recommendation system: a novel associative classification model, Decision Support Systems 48 (3) (2010) 470–479.
[22] H.F. Kaiser, A note on Guttman's lower bound for the number of common factors, British Journal of Psychology 14 (1961) 1–2.
[23] F. Klawoon, J.L. Castro, Similarity in fuzzy reasoning, Mathware and Soft Computing, 2 1995, pp. 197–228.
[24] S. Koutroumbas, K. Theodoridis, Pattern recognition, Pattern Recognition 8 (2003).
[25] T.-S. Lee, C.-C. Chiu, Y.-C. Chou, C.-J. Lu, Mining the customer credit using classification and regression tree and multivariate adaptive regression splines, Computational Statistics and Data Analysis 50 (4) (2006) 1113–1130.
[26] N. Lei, S.K. Moon, A decision support system for market-driven product positioning and design, Decision Support Systems 69 (2015) 82–91.
[27] M. Lichman, UCI machine learning repository, Retrieved from, 2013. http://archive.ics.uci.edu/ml, .
[28] P. Luukka, Similarity classifier in diagnosis of bladder cancer, Computer Methods and Programs in Biomedicine 89 (2008) 43–49.
[29] P. Luukka, PCA for fuzzy data and similarity classifier in building recognition system for post-operative patient data, Expert Systems with Applications 36 (2 Part 1) (2009) 1222–1228.
[30] P. Luukka, Nonlinear fuzzy robust PCA algorithms and similarity classifier in bankruptcy analysis, Expert Systems with Applications 37 (12) (2010) 8296–8302.
[31] P. Luukka, J. Lampinen, Differential evolution based multiple vector prototype classifier, Computing and Informatics 34 (5) (2015) 1151–1167.
[32] P. Luukka, T. Leppälampi, Similarity classifier with generalized mean applied to medical data, Computers in Biology and Medicine 36 (2006) 1026–1040.
[33] P. Luukka, K. Saastamoinen, V. Könönen, A classifier based on the maximal fuzzy similarity in the generalized Lukasiewicz-structure, 10th IEEE International Conference on Fuzzy Systems, 2001.
[34] B.P. O'Connor, Code for minimum average partial correlation test and parallel analysis, Retrieved from, 2000. https://people.ok.ubc.ca/brioconn/nfactors/nfactors.html, .
[35] B.P. O'Connor, SPSS and SAS programs for determining the number of components using parallel analysis and Velicer's MAP test, Behavior Research Methods, Instruments, & Computers 32 (3) (2000) 396–402.
[36] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, 1992.
[37] J.R. Quinlan, Induction of decision trees, Machine Learning 1 (1) (1986) 81–106.
[38] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, 3rd edition, Prentice Hall, 2009, pp. 1–1132.
[39] C. Sugar, J. Gareth, Finding the number of clusters in a data set: an information theoretic approach, Journal of the American Statistical Association 98 (2003) 750–763.
[40] R. Tibshirani, G. Walther, T. Hastie, Estimating the number of clusters in a data set via the gap statistic, Journal of the Royal Statistical Society, Series B (Statistical Methodology) 63 (2) (2001) 411–423.
[41] C.-F. Tsai, J.-W. Wu, Using neural network ensembles for bankruptcy prediction and credit scoring, Expert Systems with Applications 34 (4) (2008) 2639–2649.
[42] R. Tsaih, Y.J. Liu, W. Liu, Y.L. Lien, Credit scoring system for small business loans, Decision Support Systems 38 (1) (2004) 91–99.

[43] W.F. Velicer, Determining the number of components from the matrix of partial correlations, Psychometrika 41 (3) (1976) 321–327.
[44] W.F. Velicer, C.A. Eaton, J.L. Fava, Construct explication through factor or component analysis: a review and evaluation of alternative procedures for determining the number of factors or components, Problems and Solutions in Human Assessment, vol. 1998, 2000, pp. 41–71.
[45] A.R. Webb, Statistical Pattern Recognition, John Wiley Sons, Malvern, 2002.
[46] D. West, S. Dellana, J. Qian, Neural network ensemble strategies for financial decision applications, Computers and Operations Research 32 (10) (2005) 2543–2559.
[47] I.H. Witten, E. Frank, Data mining: practical machine learning tools and techniques, Machine Learning, 2005.
[48] L. Yu, S. Wang, K.K. Lai, L. Zhou, Bio-inspired credit risk analysis, Bio-inspired Credit Risk Analysis: Computational Intelligence With Support Vector Machines, 2008.
[49] L.A. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353.
[50] W.R. Zwick, W.F. Velicer, Factors influencing four rules for determining the number of components to retain, Multivariate Behavioral Research 17 (2) (1982) 253.
[51] W.R. Zwick, W.F. Velicer, Comparison of five rules for determining the number of components to retain, Psychological Bulletin 99 (3) (1986) 432.

Christoph Lohrmann received the M.Sc. degree with Distinction in Banking and Financial Management in 2016 from the Institute for Financial Services, University of Liechtenstein, Liechtenstein. He is currently PhD student in Computational Engineering with the School of Engineering Science, Lappeenranta University of Technology. His research interests include data analysis, classification, feature selection, decision-making and financial markets.

Pasi Luukka received the M.Sc. degree in 1999 from the Department of Information Technology, Lappeenranta University, Finland, where he also received the D.Sc. degree in Applied Mathematics in 2005 from the Department of Mathematics and Physics. He is currently Full Professor with the School of Business and Management, Lappeenranta University of Technology. His research interests include fuzzy data analysis, classification, feature selection and fuzzy decision-making.

Publication IV

Lohrmann, C., Luukka, P.
Using clustering for supervised feature selection to detect relevant features

Reprinted with permission from
Lecture Notes in Computer Science
Vol. n.d, pp. n.d, 2019

Using clustering for supervised feature selection to detect relevant features

Christoph Lohrmann and Pasi Luukka

LUT University, Yliopistonkatu 34, 53850 Lappeenranta, Finland
christoph.lohrmann@lut.fi and pasi.luukka@lut.fi

Abstract. In many applications in machine learning, large quantities of features and information are available, but these can be of low quality. A novel filter method for feature selection for classification termed COLD is presented that uses class-wise clustering to reduce the dimensionality of the data. The idea behind this approach is that if a relevant feature would be removed from the set of features, the separation of clusters belonging to different classes will deteriorate. Four artificial examples and two real-world data sets are presented on which COLD is compared with several popular filter methods. For the artificial examples, only COLD is capable to consistently rank the features according to their contribution to the separation of the classes. For the real-world Dermatology and Arrhythmia dataset, COLD demonstrates the ability to remove a large number of features and improve the classification accuracy or, at a minimum, not degrade the performance considerably.

Keywords: Feature Selection · Dimensionality Reduction · Clustering · Classification · Filter Method · Machine Learning

1 Introduction

A large number of features, meaning a high dimensionality of a dataset, can lead to severe disadvantages for the analysis of data sets such as computational cost, performance of an algorithm deployed on the data and a lack of generalizability of the results obtained [2, 6]. Feature selection is an approach that selects a subset of the existing features in a dataset to reduce the dimension of the data [13]. The objective of feature selection is to discard irrelevant and redundant features [10]. In this paper, we focus on supervised feature selection, which refers to feature selection in the context of a classification task. Since feature selection attempts to select a suitable feature subset, it has to be emphasized that the choice of the subset of the k most relevant features is not necessarily the same as taking the k features that are each by themselves the most relevant ones [5]. This appears obvious when e.g. two features are redundant [5]. However, Elashoff, Elashoff and Goldman [8] show that also in case of independent features, the subset of the k best features is not necessarily the union of the k best single features. This finding is extended by Toussaint [25], who finds that the best subset of k features does not need to contain the single best feature. As a consequence,

a feature selection algorithm should not evaluate features without taking into account the dependence to other features for the selection of the best subset.

2 Objectives

The objective of this research is to introduce a heuristic feature ranking method for supervised feature selection that deploys an intuitive approach via clustering. There has been some research on using clustering in the context of unsupervised [19] and supervised feature selection [4, 22, 24]. The research on supervised feature selection using clustering evolves around the idea to cluster features and keep for each cluster one (or more) representative feature(s) instead of all features in that cluster [4, 22, 24].

In contrast to that, the approach suggested in this paper has three distinct differences to these approaches. First, the clustering is not focused on the features but on the observations in the data. Second, the clustering is conducted for each class separtely to identify groups of observations within each class and not within the data as a whole. The focus of the clustering is on classes since the novel algorithm aims to measure the separation among classes. In order to measure how well classes are separated, one or more representative points and information on the covariance can be used. Clustering is deployed for each class, since classes that have more complex structures or groupings can be better represented by multiple representative points (cluster centroids) and the corresponding cluster covariance matrices than by a single representative and covariance matrix. Third, the clusters are not used to discard features that are similar (belonging to the same cluster), (highly) correlated or do not fit any cluster [4, 24]. Instead, the approach presented in this paper focuses solely on each feature's ability to contribute to the separation of clusters of different classes in the data. In order to be able to capture each feature's relevance not just on a stand-alone basis (univariate) but also jointly with one or more other features (multivariate), the proposed COLD algorithm compares the cluster separation between the clusters of each class with all features with the cluster separation when a certain feature is removed. In this way, the contribution of a feature to the set of the remaining features can be measured. The following sections will discuss the methods (Section 3), the data as well as the training procedure (Section 4), the results (Section 5) and the conclusion (Section 6).

3 Methods

3.1 K-medoids Clustering and Determining the Number of Clusters

In this paper, we will use a generalization of the k-means algorithm, called the k-medoids clustering [1]. K-means requires the features to be numerical while K-medoids also works with categorical variables [11] and the K-means algorithm is sensitive to the existence of outliers while K-medoids is more robust with respect to outliers and noise present in the data [11, 23]. In the scientific literature several

methods exist that estimate the optimal number of clusters. In this paper, we will use the Silhouette method due to, in our view, its intuitive idea, and the fact that the optimal number of clusters depends on the partition of the observations and not on the clustering algorithm [21].

3.2 COLD Algorithm

In this paper, we suggest a novel supervised filter method, called COLD, which ranks the features / variables according to their relevance. The name stands for "Clustering One Less Dimension" and is intended as a reference to the way this algorithm functions. The idea is to cluster each class separately to find groupings of observations for each class and then determine how each feature contributes to the separation of the clusters of each class to the clusters of all other classes, where the separation is measured in terms of the Mahalanobis distance. This cluster separation is compared between using the entire feature set and a feature subset that does not contain a certain feature. If clusters of different classes are closer to each other, so less well separated if a certain feature is not included, then this feature is by itself or jointly with other features relevant. The novel algorithm can be illustrated in a five step process:

Step 1: Pre-processing of the Data. The first step in the pre-processing is the scaling of the data into the compact interval $[0, 1]$. Also, those features that are linearly independent of each other remain in the data, whereas all others are considered redundant and are removed. The linear independent features are determined with a rank-revealing QR factorization [3].

Step 2: Clustering and the Choice of the Optimal Number of Clusters for Each Class. To determine the number of clusters for each class, the Silhouette method is used with clustering partitions for each number of clusters k from the minimum value of k specified (e.g. 1) to the maximum value of k (e.g. 10 or based on the sample size). In this paper, the Mahalanobis distance is chosen as distance measure for the k-medoids clustering and for the Silhouette index since it accounts for the covariance of the clusters (groups). To calculate the Mahalanobis distance during the clustering, the covariance matrix of each cluster has to be invertible. However, the clustering algorithm can suggest a cluster in which the covariance matrix is not invertible. If the covariance matrix is rank deficient, it will be regularized via ridge regularization [26] to be invertible by:

$$\Sigma_k = (1 - \alpha)\Sigma_k + \alpha I \tag{1}$$

Where I is the identity matrix. The parameter α is controlling how much of the identity matrix I is added to the (rank deficient) original covariance matrix Σ_k to create the regularized version. The outcome of Step 2 is the optimal number of clusters O, and the best clustering result - including cluster memberships, cluster centres and the (regularized) covariance matrices.

Step 3: Calculate the Distance Among Clusters. Let us denote the cluster indices obtained in Step 2 by $o = 1$ to O. We denote with C_o the set of cluster indices that belong to the same class as cluster o and with $\overline{C}_o$ the complement of C_o, so all cluster indices that belong to any other class than the one o belongs to. The same logic is applied to the features. D denotes the number of features in the data set. The index d denotes a feature, whereas $\overline{d}$ denotes the complement containing all features from 1 to D except d itself. The cluster centre of the o-th cluster is denoted by v_o. Then the Mahalanobis distance between two cluster centres v_o and v_m, where $m \in \overline{C}_o$ can be denoted by:

$$S(v_o, v_m) = (v_o - v_m)^T \Sigma_o^{-1} (v_o - v_m) \tag{2}$$

Where v_o and v_m are the centres of two clusters that belong to different classes. The Mahalanobis distance between each cluster centre v_o and the cluster centres corresponding to $\overline{C}_o$ is calculated.

Step 4: Calculate the Distance Among Clusters Without One Feature. This step is similar to Step 3 but does not use all features in the calculation. In particular, the Mahalanobis distance between two cluster centres v_o and v_m without the d-th feature can be expressed as follows:

$$S(v_{o,\overline{d}}, v_{m,\overline{d}}) = (v_{o,\overline{d}} - v_{m,\overline{d}})^T \Sigma_{o,\overline{d}}^{-1} (v_{o,\overline{d}} - v_{m,\overline{d}}) \tag{3}$$

Where $v_{o,\overline{d}}$ denotes the o-th cluster centre with all the features except for the d-th one. Similar to Step 3, $v_{o,\overline{d}}$ and $v_{m,\overline{d}}$ do not belong to the same class. The covariance matrix $\Sigma_{o,\overline{d}}$ of the o-th cluster without the d-th feature is simply Σ_o without the d-th row and column. This calculation is conducted for each feature between each cluster centre v_o and the cluster centres corresponding to $\overline{C}_o$.

Step 5: Determine the Rank and Score for Each Feature. A ratio of the Mahalanobis distance between the cluster centres v_o and v_m without and with feature d is calculated. This ratio is denoted as:

$$r_{o,m,d} = \frac{S(v_{o,\overline{d}}, v_{m,\overline{d}})}{S(v_o, v_m)} \tag{4}$$

This ratio is $r_{o,m,d} \geq 0$ since the distance between the clusters v_o and v_m measured by Mahalanobis distance cannot take a negative value. The value for $r_{o,m,d}$ will be larger than 1, showing that $S(v_{o,\overline{d}}, v_{m,\overline{d}})$ is larger than $S(v_o, v_m)$, when the distance between clusters increases after the removal of feature d. This implies that the d-th feature is irrelevant or even noise given that it is easier to discriminate between the clusters without it. In contrast to that, $r_{o,m,d}$ is smaller than 1, when $S(v_{o,\overline{d}}, v_{m,\overline{d}})$ is smaller than $S(v_o, v_m)$. Hence, removing the d-th feature decreases the distance between clusters v_o and v_m, making it more difficult to discriminate between them. The values of $r_{o,m,d}$ provide an indication whether the removal of a feature d is supporting the discrimination between two clusters o and m. But, all clusters $\overline{C}_o$ that do not belong to the same class as

v_o, have to be taken into account, and they are not equally important. This is premised on the consideration that the change of the distance between cluster v_o to its closest cluster of any other class has likely more impact on the ability to separate classes than any more distant cluster. Hence, the aggregation of $r_{o,m,d}$ over all other clusters $\overline{C}_o$ is weighted according to their distance to v_o.

$$w_{o,m} = \begin{cases} 1 & \text{if } card(\overline{C}_o = 1) \\ 1 - \frac{S(v_o,v_m)}{\sum_{m \in \overline{C}_o} S(v_o,v_m)} & \text{otherwise} \end{cases} \tag{5}$$

The weight defined in this way will be $w_{o,m} \in [0, 1]$. First, the formula considers the case with cardinality $card(\overline{C}_o) = 1$, which simply means that there is only one cluster from another class and the weight of it is set to 1. If there is more than one cluster of any other class, then the weight for the clusters that are closer to v_o are higher. The corresponding weighted average is calculated as:

$$r_{o,d} = \frac{\sum_{m \in \overline{C}_o} r_{o,m,d} * w_{o,m}}{\sum_{m \in \overline{C}_o} w_{o,m}} \tag{6}$$

Since the sum over $w_{o,m}$ is often not 1, the numerator is divided by the sum of the weights. A value of $r_{o,d}$ larger than 1 indicates that the removal of feature d on average separates the cluster o better from clusters of other classes than using all features including d. In contrast to that, a value equal to or smaller than 1 implies that the clusters of other classes are on average as distant or closer to cluster o without feature d. The values of $r_{o,d}$ can simply be averaged by summation and division by O, the number of clusters. To obtain a more intuitive result, the average over $r_{o,d}$ is subtracted from 1, as is stated below:

$$COLD_d = 1 - \frac{\sum_o r_{o,d}}{O} \tag{7}$$

If the $COLD_d$ feature score (being in general ≤ 1) takes a positive value, feature d is relevant for the separation between classes. In contrast to that, if the score is close to zero or negative, this indicates that the feature leads to almost no improvement or on average even an increase in the distances among clusters of different classes, which characterizes a feature that is irrelevant or even noisy. Subsequently, the features can be ranked according to their relevance from the highest feature score to the lowest. In general, at least all features with a COLD score of zero or smaller can be discarded since they deteriorate the separation among the clusters of different classes.

4 Data and Training Procedure

4.1 Artificial Examples

Four artificial classification examples were created with different data structures in order to test the new algorithm and compare it to several popular filter algorithms for feature selection. The first artificial example (Figure 1) is a classification task with two classes. Each class is characterized by three normal distributed

features without any covariance, so the variables are uncorrelated. The first two features resemble a simple numerical XOR problem with two distinct decision regions for each of the classes. The observations of each class's clusters are normally distributed. The third feature overlaps for both classes moderately. The first two features are each on their own overlapping to a large extent, which makes them almost irrelevant by themselves, but they are together able to linearly separate the two classes. The third feature that overlaps only moderately for both classes is by itself the best feature. However, the best feature subset contains only the first two features since they together can linearly separate the two classes without misclassification.

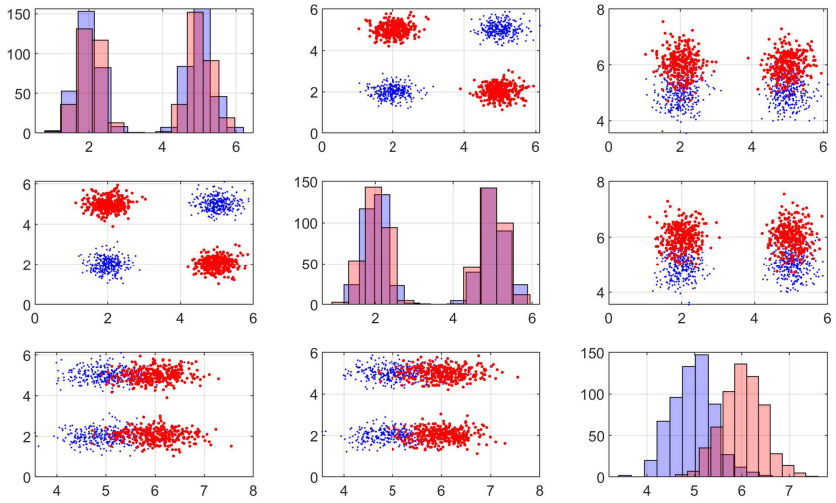


Fig. 1. First artificial example: $\mu_1 = (2, 2, 5)$, $\mu_2 = (5, 5, 5)$, $\mu_3 = (2, 5, 6)$, $\mu_4 = (5, 2, 6)$, $\Sigma = (0.1, 0, 0; 0, 0.1, 0; 0, 0, 0.2)$.

The second artificial example is based on the first example but contains a fourth normal distributed feature with the same mean and variance for both classes, which makes it clearly irrelevant for the separation of the classes.

The third artificial example is another binary class problem with three dimensions. Both classes are drawn from a normal distribution and have exactly the same covariance structure but a different mean. Feature one and two are on their own overlapping to a moderate to large extent, which makes them almost useless by themselves, but they are jointly able to linearly separate the two classes. The third feature overlaps for both classes slightly and is uncorrelated with the other two features. Only considering each dimension on its own suggests

that the third feature is the best feature. This example is highlighted in Figure 2 and is related to the fourth example in [10].

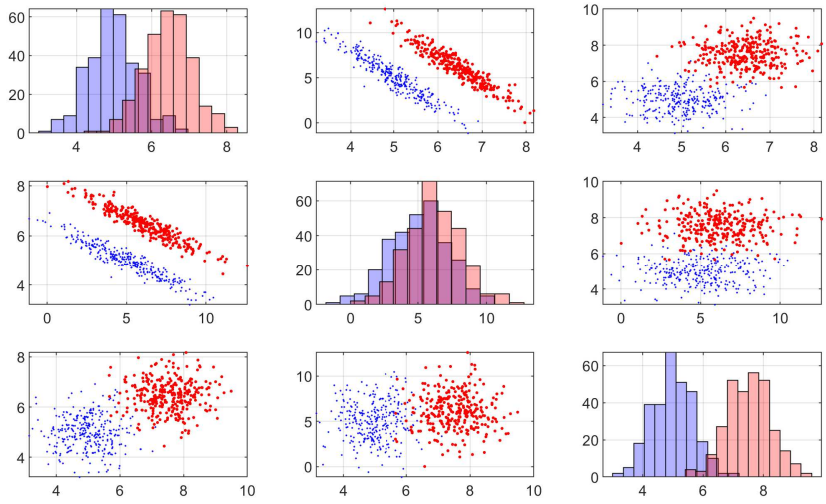


Fig. 2. Second artificial example: $\mu_1 = (5, 5, 5)$, $\mu_2 = (6.5, 6, 7.5)$, , $\Sigma = (0.5, -1.5, 0; -1.5, 5, 0; 0, 0, 0.5)$.

The fourth artificial example is based on the third example but contains a fourth normal distributed feature with the same mean and variance for both classes so it is irrelevant for the separation of the classes.

4.2 Training Procedure

For the Silhouette index as well as the k-medoids clustering, the best clustering (in terms of within-sum-of-squares) out of 100 random initializations of the initial cluster centres is selected. In the Silhouette index, the minimum number of clusters is set to 1 and the maximum number of clusters for a class is set to the rounded down quotient of the number of observations in that class divided by 100. This approach is chosen to ensure that there are on average at least 100 observations in each cluster. The parameter α is selected for each cluster separately to be the maximum of a small positive constant 0.000001 and the minimum variance on the diagonal of the covariance matrix that is larger than 0 divided by 2. The idea behind the second part is to keep the ordering of the variances in the covariance matrix the same, but, also to ensure that this value is not too small, but a suitable small positive constant. For the artificial examples, 300 observations for each cluster of a class are generated.

The feature ranking of the COLD algorithm is compared to ReliefF [15] with 10 nearest hits / misses [14] and with 70 nearest hits / misses and a decay factor $s = 20$ [20], the similarity- and entropy-based feature selection approach by Luukka [18], the 'fuzzy similarity and entropy' (FSAE) feature selection [17], the Laplacian score [12], and the Fisher score [7]. All feature selection algorithms are run 500 times to determine how volatile the feature ranking of each method is. All calculations were implemented with MATLAB^{TM}- software. The code for the Laplacian score was taken from the Matlab toolbox 'Feature Selection Library' [9]. The MATLAB^{TM} code for the COLD algorithm can be send via e-mail upon request.

5 Results

Most methods, including the new COLD algorithm, can solve the first artificial example (see Table 1) and only algorithms such as FSAE and the Fisher Score fail to identify that the subset of feature 1 and 2 can be used to better discriminate among classes than any subset of the same size containing feature 3. All feature selection algorithms such as COLD that rank the third feature last,

Table 1. Results first artificial example.

Filter Method	Feature 1	Feature 2	Feature 3
Actual Structure	Rank 1 & 2		Rank 3
ReliefF (10 h/m)	Rank 1	Rank 2	Rank 3
ReliefF (70 h/m, s = 20)	Rank 1	Rank 2	Rank 3
FS Luukka (2011)	Rank 2	Rank 1	Rank 3
FSAE (l = 1)	Rank 3	Rank 1	Rank 2
COLD	Rank 1	Rank 2	Rank 3
Laplacian Score	Rank 2	Rank 1	Rank 3
Fisher Score	Rank 3	Rank 1	Rank 2

keep or even improve their classification accuracy with the K-nearest-neighbor classifier (k=10), decision tree (Minimum leave size=10) and support vector machines (SVM, with radial basis function) of about 100%. This result represents a highly significant (p-value<0.01) outperformance to the 87.9% obtained using the Fisher Score or the FSAE, which evaluate each feature only by itself. It is obvious, that when only a single feature is considered and two features have to be discarded, the situation reverses and the Fisher Score and the FSAE outperform the remaining feature selection algorithms highly significantly. This highlights once more that algorithms such as COLD can find for this example the best (and smallest) feature subset that leads to the best class separation and highest classification accuracy, but also that this subset does not need to contain the best single feature. The results remain unaltered after a fourth irrelevant normal

distributed feature is added. Moreover, all methods correctly identify the fourth feature as the least relevant one.

For the third example, methods that rely on Euclidean distance or simple distance measures such as similarity that do not account for the cluster covariance, face difficulties. As highlighted in Table 2, only the COLD algorithm and the FSAE classify this example correctly, whereas all other methods, including ReliefF, fail to account for the dependency of the first two features.

Hence, after the first feature removal the mean classification accuracy of COLD and the FSAE remains at the same level or even improves closer to 100% for the three classification algorithms. This is once again a highly significant outperformance (p-value<0.01) of the feature subset containing 2 features suggested by COLD, but also by the FSAE, compared to the remaining feature selection algorithms. Only the Fisher Score has selected after two removals the single best feature, which leads to a performance of about 96%, which is inferior in mean performance to the best subset of two features obtained by COLD and the FSAE, but the best performing subset of a single feature.

Table 2. Results third artificial example.

Filter Method	Feature 1	Feature 2	Feature 3
Actual Structure	Rank 1 & 2		Rank 3
ReliefF (10 h/m)	Rank 1	Rank 3	Rank 2
ReliefF (70 h/m, s = 20)	Rank 1	Rank 3	Rank 2
FS Luukka (2011)	Rank 2	Rank 3	Rank 1
FSAE (l = 1)	Rank 2	Rank 1	Rank 3
COLD	Rank 1	Rank 2	Rank 3
Laplacian Score	Rank 1	Rank 3	Rank 2
Fisher Score	Rank 3	Rank 1	Rank 2

For the fourth artificial example again almost all methods leave their ranking unchanged and recognize the fourth feature as the least relevant one. Only FSAE and the Laplacian score change their ranking, the first in an adverse way, and the second now correctly ranks the four features. For each feature selection algorithm for all 500 iterations and each artificial example, the feature ranking obtained is the same and unchanged.

In addition, the feature selection algorithms were tested on two real-world datasets, the "Dermatology Data Set" (358 observations of 34 features) and the "Arrhythmia Data Set" (420 observations of 258 non-constant features) from the UCI Machine Learning Repository [16]. The computational time required to generate the feature ranking for these data sets is presented in Table 3.

It is apparent, that the COLD algorithm requires more computational time to determine the feature ranking than the other six feature ranking methods. This is the result of deploying clustering, in particular the rather computation-

Table 3. Computational time of feature selection methods (in seconds).

Data Set	ReliefF(10)	ReliefF(70,20)	Luukka	FSAE	COLD	Laplacian	Fisher
Dermatology	0.6	0.7	0.3	0.4	1.6	0.3	0.1
Arrhythmia	6.6	7.3	1.9	2.1	75.4	0.5	0.2

ally expensive but more noise-resistant k-medoids clustering algorithm, for a multivariate feature ranking that attempts to account for feature interactions.

The results for the mean performance for the three classifiers in this study on each set of features is illustrated in Figure 3.

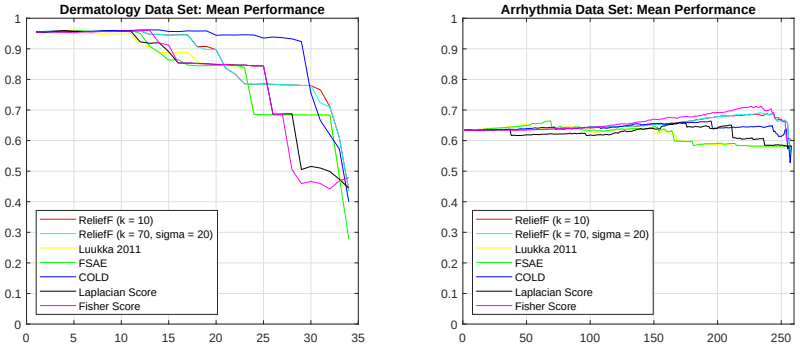


Fig. 3. Mean performance with the KNN, Decision Tree and SVM classifier

The results clearly indicate that for the Dermatology data set, the COLD algorithm determines a feature ranking that outperforms the remaining six feature selection methods between 15 to 29 feature removals with a considerable margin. Until the removal of the 30th out of 34 features, the mean performance of COLD is consistently over 92%. The highest performance with COLD is achieved with the K-nearest-neighbor classifier with 97.9% for the removal of 12 features. The one-sided Welch's test with unequal variance demonstrates that for the K-nearest-neighbor algorithm, the positive difference in the mean accuracy achieved with COLD is highly significant (p-value<0.01) for 7 to 29 removed features compared to all of the other 6 feature selection methods. The same is the case for the decision tree classifier for 17 to 29 discarded features, and with the SVM for 14 to 29 eliminated features. For the Arrhythmia data set, the Fisher Score obtains overall the highest mean accuracies. In general, COLD receives competitive results to the remaining feature selection algorithms for the Arrhythmia data set and can also improve the classification accuracy several percentage points with

about 200 features removed and can obtain a similar performance than with all features when discarding more than 250 out of 258 features.

6 Conclusion

In this research, we presented the novel supervised feature selection algorithm COLD. The algorithm is premised on clustering data class-wise and determining clusters within classes and their covariance structure. The algorithm aims to evaluate the contribution of features to the separation among clusters with and without each feature. It only considers linearly independent features, so that redundant features are automatically ranked last. Using multiple artificial examples, it was demonstrated that COLD can correctly rank features according to their joint relevance for the classification and not just by each feature's standalone characteristics. For the artificial examples, only the novel algorithm COLD consistently ranked the features according to their joint relevance. It also always correctly detected the irrelevant feature and ranked it last. On the real-world Dermatology and Arrhythmia data sets, COLD demonstrates that it can in both cases improve the mean classification accuracy for the K-nearest-neighbor, decision tree and SVM classifier and even outperform the remaining six feature ranking methods for the majority of the feature subset sizes for the Dermatology data set. In the future research, the behaviour of the COLD algorithm can be tested for higher dimensional data and additional data structures and improvements of the algorithms complexity can be investigated.

7 Acknowledgements

This research would like to acknowledge the funding received from the Finnish Strategic Research Council, grant number 313396/MFG40 Manufacturing 4.0.

References

1. Bishop, C. M.: Pattern Recognition and Machine Learning. Springer ScienceBusiness Media, New York, NY (2006)
2. Caruana, R., Freitag, D.: Greedy attribute selection. In: Cohen, W., Hirsh, H., Proceedings of the 11th International Conference on Machine Learning (ICML 1994), pp. 28–36. Morgan Kaufmann, New Brunswick, NJ (1994)
3. Chan, T. F.: Rank revealing QR factorizations. Linear Algebra and its Applications **88-89**, 67-82 (1987)
4. Chormunge, S., Jena, S.: Correlation based feature selection with clustering for highdimensional data. Journal of Electrical Systems and Information Technology **5**, 542-549 (2018)
5. Cover, T. M.: The best two independent measurements are not the two best. IEEE Transactions on Systems, Man and Cybernetics **4**(1), 116-117 (1974)
6. Dessì, N., Pes, B.: Similarity of feature selection methods. An empirical study across data intensive classification tasks. Expert Systems with Applications **42**(10), 4632-4642 (2015)

7. Duda, R., Hart, P., Stork, D.: Pattern Classification. John Wiley and Sons, New York, NY (2012)
8. Elashoff, J. E, Elashoff, R. M., Goldman, G. E.: On the choice of variables in classification problems with dichotomous variables. Biometrika **54**(3), 668-670 (1967)
9. Ruffo, G.: Matlab Toolbox: Feature selection library. https://se.mathworks.com/matlabcentral/fileexchange/56937-feature-selection-library. Last accessed 1 Dec 2018
10. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research **3**, 1157-1182 (2003)
11. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Data Mining, Inference, and Prediction. Springer Series in Statistics, New York, NY (2009)
12. He X., Cai, D., Niyogi, P.: Laplacian score for feature selection. In: Proceedings of the 18th International Conference on Neural Information Processing Systems (NIPS 2005), pp. 507–514. MIT Press Cambridge, Vancouver, British Columbia (2005)
13. Kittler, J., Mardia, K. V.: Statistical pattern recognition in image analysis. Journal of Applied Statistics **21**(1-2), 61-75 (1994)
14. Kononenko, I.: Estimating attributes. analysis and extensions of Relief. In: European Conference on Machine Learning (ECML 1994), pp. 171–182. Springer, Berlin Heidelberg (1994)
15. Kononenko, I., Simec, E., Robnik-Sikonja, M.: Overcoming the myopia of inductive learning Algorithms with RELIEFF. Applied Intelligence **7**, 39-55 (1997)
16. Lichman, M., UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/index.php. Last accessed 20 June 2019
17. Lohrmann, C., Luukka, P., Jablonska-Sabuka, M., Kauranne, T.: Supervised feature selection with a combination of fuzzy similarity measures and fuzzy entropy measures. Expert Systems with Applications **110**, 216-236 (2018)
18. Luukka, P.: Feature selection using fuzzy entropy measures with similarity classifier. Expert Systems with Applications **38**, 4600-4607 (2011)
19. Mitra, P., Murthy, C. A., Pal, S. K.: Unsupervised feature selection using feature similarity. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(3), 301-312 (2002)
20. Robnik-Šikonja, M., Kononenko, I.: Theoretical and Empirical Analysis of ReliefF and RReliefF. Applied Intelligence **53**(1-2), 23-69 (2003)
21. Rousseeuw, P. J.: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics **20**, 53-65 (1987)
22. Sahu, B., Dehuri, S., Jagadev, A. K.: Feature selection model based on clustering and ranking in pipeline for microarray data. Informatics in Medicine Unlocked **9**, 107-122 (2017)
23. Sammut, C., Webb, G. I.: Encyclopedia of Machine Learning and Data Mining 2017 Edition. Springer Science+Business Media, New York, NY (2017)
24. Sotoca, J., M., Pla, F.: Supervised feature selection by clustering using conditional mutual information-based distances. Pattern Recognition **43**, 2068–2081 (2010)
25. Toussaint, G. T.: Note on optimal selection of independent binary-valued features for pattern recognition. IEEE Transactions on Information Theory **17**(5), 618 (1971)
26. Warton, D. I.: Penalized normal likelihood and ridge regularization of correlation and covariance matrices. Journal of the American Statistical Association **103**(481), 340-349 (2008)

ACTA UNIVERSITATIS LAPPEENRANTAENSIS

852. RISSANEN, TOMMI. Perspectives on business model experimentation in internationalizing high-tech companies. 2019. Diss.

853. HASSANZADEH, AIDIN. Advanced techniques for unsupervised classification of remote sensing hyperspectral images. 2019. Diss.

854. POPOVIC, TAMARA. Quantitative indicators of social sustainability applicable in process systems engineering. 2019. Diss.

855. RAMASAMY, DEEPIKA. Selective recovery of rare earth elements from diluted aqueous streams using N- and O –coordination ligand grafted organic-inorganic hybrid composites. 2019. Diss.

856. IFTEKHAR, SIDRA. Synthesis of hybrid bio-nanocomposites and their application for the removal of rare earth elements from synthetic wastewater. 2019. Diss.

857. HUIKURI, MARKO. Modelling and disturbance compensation of a permanent magnet linear motor with a discontinuous track 2019. Diss.

858. AALTO, MIKA. Agent-based modeling as part of biomass supply system research. 2019. Diss.

859. IVANOVA, TATYANA. Atomic layer deposition of catalytic materials for environmental protection. 2019. Diss.

860. SOKOLOV, ALEXANDER. Pulsed corona discharge for wastewater treatment and modification of organic materials. 2019. Diss.

861. DOSHI, BHAIRAVI. Towards a sustainable valorisation of spilled oil by establishing a green chemistry between a surface active moiety of chitosan and oils. 2019. Diss.

862. KHADIJEH, NEKOUEIAN. Modification of carbon-based electrodes using metal nanostructures: Application to voltammetric determination of some pharmaceutical and biological compounds. 2019. Diss.

863. HANSKI, JYRI. Supporting strategic asset management in complex and uncertain decision contexts. 2019. Diss.

864. OTRA-AHO, VILLE. A project management office as a project organization's strategizing tool. 2019. Diss.

865. HILTUNEN, SALLA. Hydrothermal stability of microfibrillated cellulose. 2019. Diss.

866. GURUNG, KHUM. Membrane bioreactor for the removal of emerging contaminants from municipal wastewater and its viability of integrating advanced oxidation processes. 2019. Diss.

867. AWAN, USAMA. Inter-firm relationship leading towards social sustainability in export manufacturing firms. 2019. Diss.

868. SAVCHENKO, DMITRII. Testing microservice applications. 2019. Diss.

869. KARHU, MIIKKA. On weldability of thick section austenitic stainless steel using laser processes. 2019. Diss.

870. KUPARINEN, KATJA. Transforming the chemical pulp industry – From an emitter to a source of negative CO2 emissions. 2019. Diss.

871. HUJALA, ELINA. Quantification of large steam bubble oscillations and chugging using image analysis. 2019. Diss.

872. ZHIDCHENKO, VICTOR. Methods for lifecycle support of hydraulically actuated mobile working machines using IoT and digital twin concepts. 2019. Diss.

873. EGOROV, DMITRY. Ferrite permanent magnet hysteresis loss in rotating electrical machinery. 2019. Diss.

874. PALMER, CAROLIN. Psychological aspects of entrepreneurship – How personality and cognitive abilities influence leadership. 2019. Diss.

875. TALÁSEK, TOMÁS. The linguistic approximation of fuzzy models outputs. 2019. Diss.

876. LAHDENPERÄ, ESKO. Mass transfer modeling in slow-release dissolution and in reactive extraction using experimental verification. 2019. Diss.

877. GRÜNENWALD, STEFAN. High power fiber laser welding of thick section materials - Process performance and weld properties. 2019. Diss.

878. NARAYANAN, ARUN. Renewable-energy-based single and community microgrids integrated with electricity markets. 2019. Diss.

879. JAATINEN, PEKKO. Design and control of a permanent magnet bearingless machine. 2019. Diss.

880. HILTUNEN, JANI. Improving the DC-DC power conversion efficiency in a solid oxide fuel cell system. 2019. Diss.

881. RAHIKAINEN, JARKKO. On the dynamic simulation of coupled multibody and hydraulic systems for real-time applications. 2019. Diss.

882. ALAPERÄ, ILARI. Grid support by battery energy storage system secondary applications. 2019. Diss.

883. TYKKYLÄINEN, SAILA. Growth for the common good? Social enterprises' growth process. 2019. Diss.

884. TUOMISALO, TEEMU. Learning and entrepreneurial opportunity development within a Finnish telecommunication International Venture. 2019. Diss.

885. OYEDEJI, SHOLA. Software sustainability by design. 2019. Diss.

886. HUTTUNEN, MANU. Optimizing the specific energy consumption of vacuum filtration. 2019. Diss.

887. LIIKANEN, MIIA. Identifying the influence of an operational environment on environmental impacts of waste management. 2019. Diss.

888. RANTALA, TERO. Operational level performance measurement in university-industry collaboration. 2019. Diss.

889. LAUKKANEN, MINTTU. Sustainable business models for advancing system-level sustainability. 2019. Diss.

ISBN 978-952-335-472-2
ISBN 978-952-335-473-9 (PDF)
ISSN-L 1456-4491
ISSN 1456-4491
Lappeenranta 2019