A novel similarity classifier with multiple ideal vectors based on k-means clustering

Lohrmann Christoph, Luukka Pasi

DOI: 10.1016/j.dss.2018.04.003

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

christoph.lohrmann@student.lut.fi, pasi.luukka@lut.fi

* corresponding author: christoph.lohrmann@student.lut.fi, +49 1522 6266 372

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 (Bishop, 2006). Classification algorithms support the decision-making for enterprises and individuals in numerous applications, including medical diagnostics (Luukka, 2008), product positioning (Lei & Moon, 2015), recommendation systems (Jiang, Shang, & Liu, 2010) and sentiment analysis in social media (Figini, Bonelli, & Giovannini, 2017). 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 (Figini et al., 2017; Huang, Chen, Hsu, Chen, & Wu, 2004; Tsaih, Liu, Liu, & Lien, 2004).

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 (Luukka, Saastamoinen, & Könönen, 2001). The first results for the similarity classifier were published in Luukka et al. (2001). Since then, the classifier has been applied to several medical data sets (Luukka, 2008; Luukka & Leppälampi, 2006) and to two bankruptcy data sets (Luukka, 2010), showing high classification accuracies. Moreover, Luukka & Leppälampi (2006) demonstrated that the similarity classifier outperforms classifiers such as linear discriminant analysis, the C4.5 algorithm (J. R. Quinlan, 1992) and multi-layer perceptron neural networks on the medical data sets in their study. Luukka (2009b) 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 (Luukka, 2008).

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 (Formato, Gerla, & Scarpati, 1999) and the similarity classifier can be regarded as a distance-based technique. Luukka & Lampinen (2015) pointed out that distance-based techniques may face difficulties to classify complex data structures. Hence, Luukka & Lampinen (2015) 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 (Luukka & Lampinen, 2015). However, Luukka & Lampinen (2015) highlighted that the choice of the number of vectors per each class is pivotal for the 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 (2015) 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. (2001). As a second contribution, the authors in this paper clearly address the research need mentioned by Luukka & Lampinen (2015) 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 (Dougherty, 2013). In other words, observations in a cluster have small inter-point distances in relation to the distance to observations in other clusters (Bishop, 2006). The k-means clustering algorithm is one of the first and widely applied hard clustering algorithms (Dougherty, 2013; Koutroumbas & Theodoridis, 2003). 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 (Dougherty, 2013). 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 (Bishop, 2006). The assignment of an observation *i* to the cluster with the closest cluster centre can be expressed as (Bishop, 2006; Duda, Hart, & Stork, 2000):

$$u_{ij} = \begin{cases} 1 \; if \; \|x_i - \mu_j\|^2 < \|x_i - \mu_{j'}\|^2 \; for \; all \; j' \neq j \\ 0 \quad Otherwise \end{cases} \tag{1}$$

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 (Bishop, 2006):

$$\mu_j = \frac{\sum_i u_{ij} x_i}{\sum_i u_{ij}} \tag{2}$$

The objective function that will be minimized with respect to the membership coefficients and cluster centres is (Bishop, 2006; Koutroumbas & Theodoridis, 2003):

$$J = \sum_{i=1}^{N} \sum_{j=1}^{k} u_{ij} \|x_i - \mu_j\|^2 \quad (3)$$

This function represents the sum of squared distances of each observation to its cluster centre (Bishop, 2006).

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 (Bishop, 2006). 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 (Witten & Frank, 2005). In the literature, many approaches to determine a suitable number of clusters can be found. These include the 'Elbow method', the 'Gap statistic' (Tibshirani, Walther, & Hastie, 2001), the 'Jump method' (Sugar & Gareth, 2003) and the 'Calinski-Harabasz index' (Calinski & Harabasz, 1974). 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 (Sugar & Gareth, 2003). The 'jump method' developed by Sugar & Gareth (2003) is related to rate distortion theory. Distortion is a measure for the dispersion within clusters (Dougherty, 2013). The minimum distortion d_k obtainable with *K* cluster centres is (Sugar & Gareth, 2003):

$$d_K = \frac{1}{p} \min_{c_1,\dots,c_K} E[(X - c_X)^T \Gamma^{-1} (X - c_X)] \quad (4)$$

Where X is a p-dimensional random variable with a mixture distribution with *G* components and covariance matrix Γ for *X*. In addition, $c_1, \dots, c_K$ 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 (Sugar & Gareth, 2003). The covariance matrix Γ might in practice not be known. However, Sugar & Gareth (2003) 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 (Sugar & Gareth, 2003). Consequently, the minimum distortions can easily be obtained given the observations and *K* clusters. The 'jump method' can be stated as follows (Sugar & Gareth, 2003):

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 (2003) 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 (2003) 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 (2003) 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 (Abdi & Williams, 2010; Jackson, 1991). 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 (Dougherty, 2013). 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 (O'Connor, 2000b; Velicer, 1976). Extracting too few principal components will result in a loss of information while extracting too numerous principal components might include irrelevant information or noise (Cangelosi & Goriely, 2007; Zwick & Velicer, 1986).

In the literature, various methods to determine a suitable number of principal components can be found (Cangelosi & Goriely, 2007). These methods include, but are not limited to, the modified broken stick model (Cangelosi & Goriely, 2007), the Guttman-Kaiser criterion (Guttman, 1954; Kaiser, 1961), the SCREE test (Cattell, 1966), the Minimum Partial Average (MAP) test (Velicer, 1976), Bartlett's test (Bartlett, 1950) and Parallel Analysis (Horn, 1965). Of these methods, the MAP and Parallel analysis demonstrated the highest performance across different data complexities (O'Connor, 2000b; Zwick & Velicer, 1982, 1986). The minimum average partial (MAP) test is based on conducting a PCA and subsequently analyse the matrix of partial correlations (Cangelosi & Goriely, 2007; O'Connor, 2000b; Velicer, 1976; Zwick & Velicer, 1982). The idea behind this procedure is that the average squared partial correlation will decline until a 'unique' component would be removed (Velicer, 1976; Zwick & Velicer, 1982). Therefore, the stopping point is reached at the minimum average squared partial correlation (Velicer, Eaton, & Fava, 2000; Zwick & Velicer, 1982). According to Velicer (1976) the method results in an exact stopping point for the selection of principal components. Velicer et al. (2000) 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 (Zwick & Velicer, 1982).

The second highly recommended approach is Parallel Analysis developed by Horn (1965) (Velicer et al., 2000; Zwick & Velicer, 1982). 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 (Garrido, Abad, & Ponsoda, 2013; Horn, 1965). 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 (O'Connor, 2000b). An alternative approach for Parallel Analysis is to deploy an upper percentile (commonly the 95^{th}) for the distribution of the eigenvalues as explained by Glorfeld (1995). Using this approach decreases the tendency of Parallel Analysis to extract too numerous components (Glorfeld, 1995).

MAP and Parallel Analysis usually lead to the selection of the same principal components to retain (O'Connor, 2000b). 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 (O'Connor, 2000b; Zwick & Velicer, 1986).

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 (Klawoon & Castro, 1995). In contrast to that, fuzzy sets reflect a membership degree to a class or property (Zadeh, 1965). Using membership degrees allows to model partial memberships. This is of interest for classification since it allows partial membership of an observation to classes (Luukka et al., 2001). 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 (Bishop, 2006; Webb, 2002).

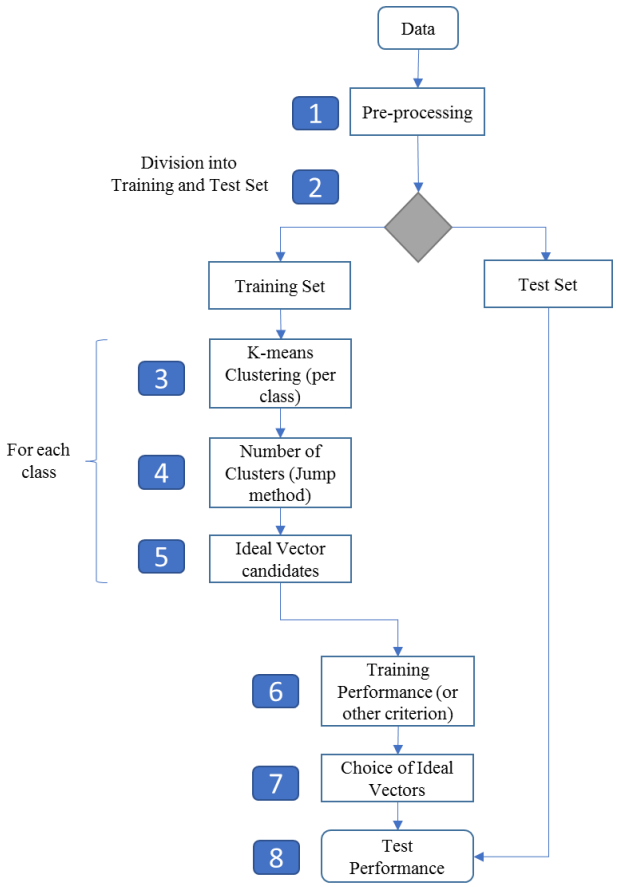
The similarity classifier presented by Luukka et al. (2001) was premised on the idea to compute for each class one so-called 'ideal vector' that is supposed the represent that class well. There are several ways of computing ideal vectors, of which arithmetic mean is one of the earliest methods used. 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. (2001) and Luukka (2008).

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 (Sugar & Gareth, 2003). 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 Figure 1 and depicted in detail subsequently.

Figure 1: Flowchart of the similarity classifier with multiple ideal vectors



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 (2000b), 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:

$$\hat{d}_K = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\frac{u_{ik} * (x_i - c_k)^T \Gamma^{-1}(x_i - c_k)}{p} \tag{5}$$

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 (2003). 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:

$$S(x_d, v_{o,d}) = \sqrt[p]{1 - \left|x_d^p - v_{o,d}^p\right|} \tag{8}$$

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:

$$Cl(x) = \underset{o=1,\ldots O}{\arg\max}\, S(x, v_o) \tag{10}$$

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:

$$C(x) = f\big(Cl(x)\big) \tag{11}$$

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 (Lichman, 2013) 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.

Table 1: Characteristics of the three artificial data sets

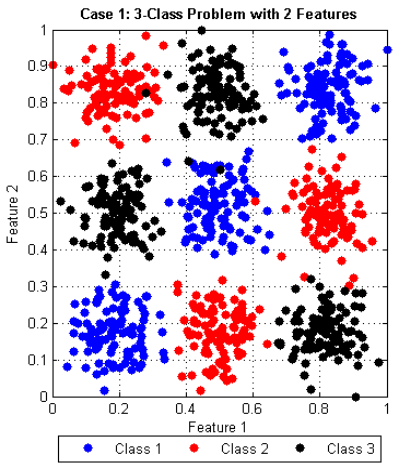
Cases	Observations	Class	Feature 1	Feature 2	Feature 3
Case A	900	1	N(1,0.2)	N(1,0.2)	-
		1	N(2,0.2)	N(2,0.2)	-
		1	N(3,0.2)	N(3,0.2)	-
		2	N(3,0.2)	N(2,0.2)	-
		2	N(2,0.2)	N(1,0.2)	-
		2	N(1,0.2)	N(3,0.2)	-
		3	N(3,0.2)	N(1,0.2)	-
		3	N(1,0.2)	N(2,0.2)	-
		3	N(2,0.2)	N(3,0.2)	-
Case B	1000	1	[0, 0.5)	[0, 0.5)	[0, 1]

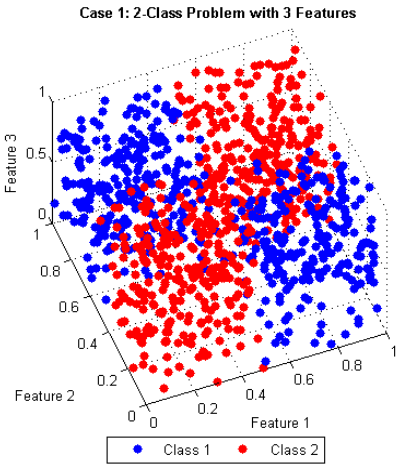
		1	[0.5, 1]	[0.5, 1]	[0, 1]
		2	[0, 0.5)	[0.5, 1]	[0, 1]
		2	[0.5, 1]	[0, 0.5)	[0, 1]
Case C	1000	1	N(2,0.1)	N(2,0.1)	N(2,0.1)
		1	N(6,0.5)	N(6,0.5)	N(6,0.5)
		2	N(6,0.1)	N(6,0.1)	N(2,0.1)
		2	N(2,0.5)	N(2,0.5)	N(6,0.5)
		3	N(2,0.5)	N(6,0.5)	N(2,0.5)
		3	N(6,0.1)	N(2,0.1)	N(6,0.1)
		4	N(6,0.5)	N(2,0.5)	N(2,0.5)
		4	N(2,0.1)	N(6,0.1)	N(6,0.1)

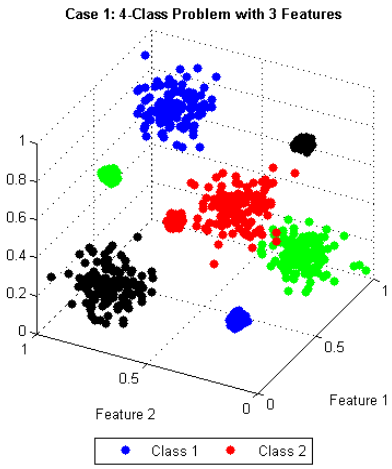
The first data set, Case A, normally distributed features with small variations are generated that form two-dimensional clusters for each class. In this data set small overlap of classes is present, but the feature space can almost distinctly be divided into the multiple decision regions for each class. 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 Figure 2

Figure 2.

Figure 2: Artificial data sets







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 (Tsaih et al., 2004; West, Dellana, & Qian, 2005). The credit granting decision copes with the risk of granting credits to not suitable applicants and the non-acceptance of credits for solvent clients (Lee, Chiu, Chou, & Lu, 2006). The classification of clients is particularly important since a credit scoring that is conducted effectively will most likely lead to savings in the future (Yu, Wang, Lai, & Zhou, 2008).

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 are 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 represents 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 are 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 (Velicer et al., 2000) 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' (O'Connor, 2000b), it is defined as an increase of the average partial correlations per step of less than 70 percent 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 percent points appeared small compared to larger changes that were characterized by increases of at least 100% for an additional component. Therefore, the 70 percent 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 (Cover & Hart, 1967), the Naive Bayes classifier (Russell & Norvig, 2009), decision trees (Quinlan, 1986) and the ensemble learning algorithm called random forest (Breiman, 2001). All calculations are implemented with the MATLAB™- software. The code for the MAP and PA are based on Matlab-files provided by O'Connor (2000a).

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 are 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 more than 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 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 more than 99.99% confidence).

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

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 more than 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.

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

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. It is noteworthy, that the mean performance with 4 PCs for the novel similarity classifier with Y = 1 is the largest among the results with the selected PCs.

Table 4: Results for the 'Credit Approval' data set

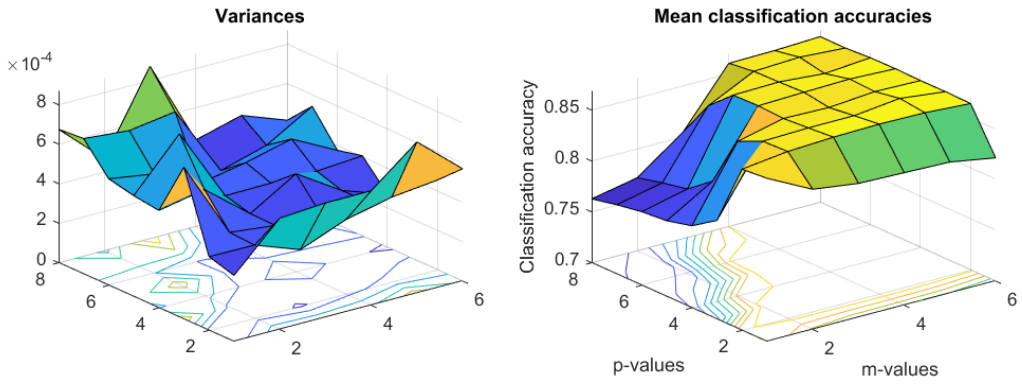
Classification Algorithm	Mean Accuracy	Variance	Mean FNR	Mean FPR	p	m	Y
Standard Similarity Classifier	0.8599	0.0004	0.196	**0.072**	6	4	-
Novel Similarity Classifier	0.8525	0.0005	0.133	0.160	1	1	p/2
Novel Similarity Classifier	0.8706	0.0005	0.056	0.190	6	5	1
Novel Similarity Classifier (Minimize FPR)	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	-	-	-

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) (Berardi & Zhang, 1997; Chuang & Huang, 2011; Tsai & Wu, 2008).

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 Figure 3.

Figure 3: Optimal value search for the novel similarity classifier with Y=1 ('Credit Approval' data set)



The surface for the mean accuracy for the novel similarity classifier appears smooth and high classification 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' 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).

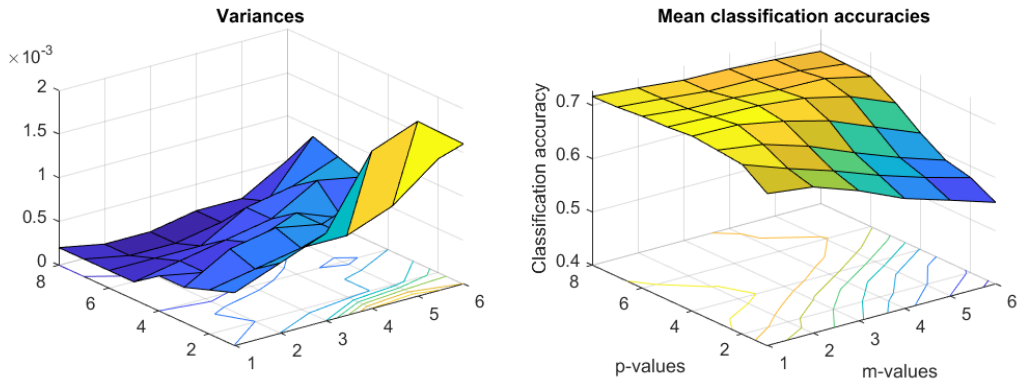
Table 5: Results for the 'German Credit' data set

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	-	-	-

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. This, however, is achieved at the expense of the classification accuracy. 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 Figure 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. .

Figure 4: Optimal value search for the novel similarity classifier with Y=1 ('German Credit' data set)



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.

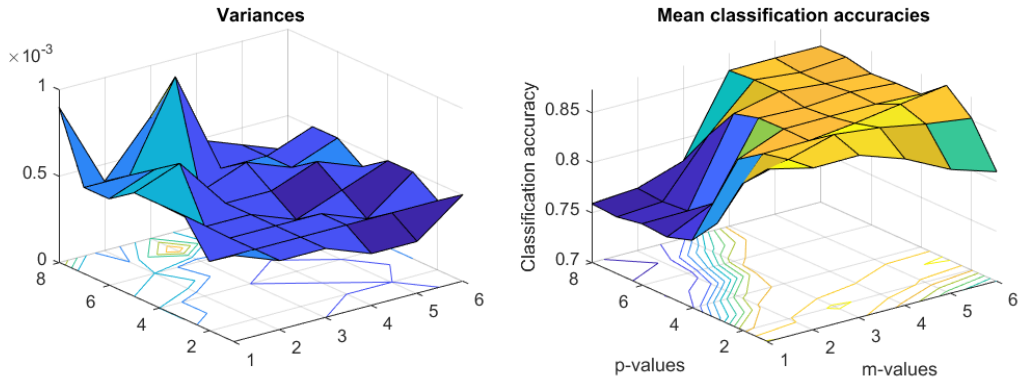
Table 6: Results for the 'Australian Credit' data set

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	-	-	-

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 Figure 5:

Figure 5: Optimal value search for the novel similarity classifier with Y=1 ('Australian Credit' data)



Overall, the novel similarity classifier achieved for all artificial data sets superior classification results to the standard similarity classifier 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 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

Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*.

Bartlett, M. S. (1950). Tests of significance in factor analysis. *British Journal of Statistical Psychology, 3*(2), 77–85.

Berardi, V. L., & Zhang, G. Q. (1997). The effect of misclassification costs on neural network classifiers. *Decision Sciences Institute, 1997 Annual Meeting, Proceedings, Vols 1-3*, *30*(3), 364–366.

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

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Calinski, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics - Theory and Methods*, *3*(1), 1–27.

Cangelosi, R., & Goriely, A. (2007). Component retention in principal component analysis with application to cDNA microarray data. *Biology Direct*, *2*, 2.

Cattell, R. B. (1966). The Scree Test For The Number Of Factors. *Multivariate Behavioral Research*, *1*(2), 245–276.

Chuang, C.-L., & Huang, S.-T. (2011). A hybrid neural network approach for credit scoring. *Expert Systems*, *28*(2), 185–196.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27.

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

Duda, R. O., Hart, P. E., & Stork, D. G. (2000). *Pattern Classification*. *New York: John Wiley*, *Section*.

Figini, S., Bonelli, F., & Giovannini, E. (2017). Solvency prediction for small and medium enterprises in banking. *Decision Support Systems*.

Formato, F., Gerla, G., & Scarpati, L. (1999). Fuzzy subgroups and similarities. *Soft Computing*, *3*, 1–6.

Garrido, L. E., Abad, F. J., & Ponsoda, V. (2013). A new look at Horn's parallel analysis with ordinal variables. *Psychological Methods*, *18*(4), 454–474.

Glorfeld, L. W. (1995). An Improvement on Horn's Parallel Analysis Methodology for Selecting the Correct Number of Factors to Retain. *Educational and Psychological Measurement*, *55*(3), 377–393.

Guttman, L. (1954). Some necessary conditions for common-factor analysis. *Psychometrika*, *19*(2), 149–161.

Horn, J. L. (1965). A rationale and test for the number of factors in factor analysis. *Psychometrika*, *30*(2), 179–185.

Huang, Z., Chen, H., Hsu, C.-J., Chen, W.-H., & Wu, S. (2004). Credit rating analysis with support vector machines and neural networks: a market comparative study. *Decision Support Systems*, *37*(4), 543–558.

Jackson, J. E. (1991). *A User's Guide to Principal Components*. *John Wiley & Sons, Inc.*

Jiang, Y., Shang, J., & Liu, Y. (2010). Maximizing customer satisfaction through an online recommendation system: A novel associative classification model. *Decision Support Systems*, *48*(3), 470–479.

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

Klawoon, F., & Castro, J. L. (1995). Similarity in fuzzy reasoning. *Mathware and Soft Computing*, *2*, 197–228.

Koutroumbas, S., & Theodoridis, K. (2003). *Pattern Recognition*. *Pattern recognition* (Vol. 8).

Lee, T.-S., Chiu, C.-C., Chou, Y.-C., & Lu, C.-J. (2006). Mining the customer credit using classification and regression tree and multivariate adaptive regression splines. *Computational Statistics & Data Analysis*, *50*(4), 1113–1130.

Lei, N., & Moon, S. K. (2015). A Decision Support System for market-driven product positioning and design. *Decision Support Systems*, *69*, 82–91.

Lichman, M. (2013). UCI Machine Learning Repository. Retrieved March 5, 2017, from

http://archive.ics.uci.edu/ml

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

Luukka, P. (2009). PCA for fuzzy data and similarity classifier in building recognition system for post-operative patient data. *Expert Systems with Applications*, *36*(2 PART 1), 1222–1228.

Luukka, P. (2010). Nonlinear fuzzy robust PCA algorithms and similarity classifier in bankruptcy analysis. *Expert Systems with Applications*, *37*(12), 8296–8302.

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

Luukka, P., & Leppälampi, T. (2006). Similarity classifier with generalized mean applied to medical data. *Computers in Biology and Medicine*, *36*, 1026–1040.

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

O'Connor, B. P. (2000a). Code for minimum average partial correlation test and parallel analysis. Retrieved from https://people.ok.ubc.ca/brioconn/nfactors/nfactors.html

O'Connor, B. P. (2000b). 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), 396–402.

Quinlan, J. R. (1992). *C4.5: Programs for Machine Learning*. (Morgan Kaufmann Publishers, Ed.). San Mateo.

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

Russell, S., & Norvig, P. (2009). Artificial Intelligence: A Modern Approach, 3rd edition. *Prentice Hall*, 1–1132.

Sugar, C., & Gareth, J. (2003). Finding the number of clusters in a data set : An information theoretic

approach. *Journal of the American Statistical Association*, *98*, 750–763.

Tibshirani, R., Walther, G., & 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), 411–423.

Tsai, C.-F., & Wu, J.-W. (2008). Using neural network ensembles for bankruptcy prediction and credit scoring. *Expert Systems with Applications*, *34*(4), 2639–2649.

Tsaih, R., Liu, Y. J., Liu, W., & Lien, Y. L. (2004). Credit scoring system for small business loans. *Decision Support Systems*, *38*(1), 91–99.

Velicer, W. F. (1976). Determining the number of components from the matrix of partial correlations. *Psychometrika*, *41*(3), 321–327.

Velicer, W. F., Eaton, C. A., & Fava, J. L. (2000). Construct Explication through Factor or Component Analysis: A Review and Evaluation of Alternative Procedures for Determining the Number of Factors or Components. In *Problems and Solutions in Human Assessment* (Vol. 1998, pp. 41–71).

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

West, D., Dellana, S., & Qian, J. (2005). Neural network ensemble strategies for financial decision applications. *Computers and Operations Research*, *32*(10), 2543–2559.

Witten, I. . H., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. *Machine Learning*.

Yu, L., Wang, S., Lai, K. K., & Zhou, L. (2008). *Bio-Inspired Credit Risk Analysis*. *Bio-Inspired Credit Risk Analysis: Computational Intelligence with Support Vector Machines*.

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

Zwick, W. R., & Velicer, W. F. (1982). Factors influencing four rules for determining the number of components to retain. *Multivariate Behavioral Research*, *17*(2), 253.

Zwick, W. R., & Velicer, W. F. (1986). Comparison of five rules for determining the number of

components to retain. *Psychological Bulletin*, *99*(3), 432.