MATERIAL CLASSIFICATION IN THE INDUSTRY

Lappeenranta-Lahti University of Technology LUT

Master's Program in Computational Engineering, Master's Thesis

2023

Joonas Maljanen

Examiners:	Professor Heikki Kälviäinen M.Sc. (Tech.) Ari Suutari

ABSTRACT

Lappeenranta-Lahti University of Technology LUT
School of Engineering Science
Computational Engineering

Joonas Maljanen

Material classification in the industry

Master's thesis
2023
40 pages, 12 figures, 7 tables, 0 appendices
Examiners: Professor Heikki Kälviäinen and M.Sc. (Tech.) Ari Suutari

Keywords: machine learning, pattern recognition, classification, spectroscopy, industry

In the industry material classification and quality control are key challenges that require speed and accuracy. Automation of material classification using machine learning classifiers would provide a unified method that enables faster and more accurate classifications while reducing human errors. The focus of this study is to research industrial classification of material based on the material's features. One of the most common classification approaches is to use spectrometer analysis where aborption, emission or scattering properties of the material are analyzed at different wavelengths using a spectrometer. The goal of this study is to develop a machine learning model for the case industry that can classify materials based on spectrometer analysis. For this study the case industry has provided a dataset of 30 000 material classications with 200 unique classes. To develop the model several pre-processing methods, data balancing methods and classification methods that have achieved good results were experimented. In the conducted experiments the best 95% validation accuracy was achieved with a model that used unscaled data, was balanced with Synthetic Minority Oversampling Technique (SMOTE) and was classified with the Random Forest (RF) classifier.

TIIVISTELMÄ

Lappeenrannan-Lahden teknillinen yliopisto LUT
School of Engineering Science
Laskennallinen tekniikka

Joonas Maljanen

Materiaalien luokittelu teollisuudessa

Diplomityö
2023
40 sivua, 12 kuvaa, 7 taulukkoa, 0 liitettä
Tarkastajat: Professor Heikki Kälviäinen ja M.Sc. (Tech.) Ari Suutari

Hakusanat: koneoppiminen, hahmontunnistus, luokittelu, spektroskopia, teollisuus
Keywords: machine learning, pattern recognition, classification, spectroscopy, industry

Teollisuudessa materiaalien luokittelu ja laadunvalvonta ovat keskeisiä haasteita, joissa vaaditaan nopeutta ja tarkkuutta. Materiaalien luokittelun automatisointi koneoppivien luokittelijoiden avulla tarjoaisi yhtenäisen menetelmän, joka mahdollistaisi nopeamman ja tarkemman luokittelun samalla vähentäen inhimmillisiä virheitä. Tässä tutkimuksessa perehdytään teollisuuden materiaalin luokitteluongelmiin, jossa pyritään luokittelemaan materiaaleja niiden ominaisuuksien perusteella. Luokittelutavoista yksi yleisimpiä tapoja on luokitella teollisia materiaaleja spektrometrianalyysilla, jossa spektroskoopilla mitataan materiaalin absorptio-, emissio- tai hajontaominaisuuksia eri aallonpituuksilla. Tutkimuksen tavoitteena on kehittää koneoppiva malli kohdeteollisuudelle, joka pystyy luokittelemaan materiaaleja spektrometrianalyysin perusteella. Kohdeteollisuus antoi tutkimuksen käyttöön 30 000 materiaalin luokittelun tietokannan, joka sisälsi 200 eri luokkaa. Mallin kehittämiseksi testattiin useita esikäsittelymenetelmiä, datan tasapainottamismenetelmiä ja luokittelumenetelmiä, jotka ovat saavuttaneet hyviä tuloksia aiemmissa tutkimuksissa. Suoritetuissa kokeissa parhaan 95 %:n validointitarkkuuden saavuttivat malli, joka käytti skaalaamatonta dataa, joka oli tasapainotettu SMOTE-menetelmällä ja luokiteltu RF-luokittelijalla.

ACKNOWLEDGEMENTS

I would like to thank my supervisors Professor Heikki Kälviäinen and M.Sc Ari Suutari for their support in writing this thesis and giving guidance when needed. A big thank you to Vilma Mäkitalo who helped me get started and provided invaluable support during the planning and experimentation phases of this thesis. I would also like to thank the case industry for giving me this topic and providing expert knowledge in understanding industrial material classification and its challenges more profoundly.

Studying would not have been half as efficient or enjoyable if it weren't for the constant support from my friends and family. A special thank you to guild activities, student gatherings and kurssi-1 gang meetups for offering invaluable peer support and providing an opportunity to unwind and relax after long days of studying.

Lappeenranta, May 29, 2023

Joonas Maljanen

LIST OF ABBREVIATIONS

ADASYN	Adaptive Synthetic Sampling Approach
ATR–FTIR	Attenuated Total Reflectance–Fourier Transform Infrared Spectroscopy
CNN	Convolutional Neural Network
k-NN	k-Nearest Neighbors
LDA	Linear Discriminant Analysis
LIBS	Laser Induced Breakdown Spectroscopy
MaxAbs scaling	Maximum Absolute Scaling
MLP	Multi-layer Perceptron classifier
NIR	Near-Infrared Spectroscopy
PCA	Principal Component Analysis
RBF	Radial Basis Function
RF	Random Forest
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machine

CONTENTS

1 INTRODUCTION 8

1.1 Background 8

1.2 Objectives and delimitations 9

1.3 Structure of the thesis 9

2 MATERIAL CLASSIFICATION IN INDUSTRIAL APPLICATIONS 10

2.1 Classification for recycling 10

2.1.1 Plastic classification 10

2.1.2 Metal classification 11

2.2 Quality control for products 12

2.3 Summary 13

3 MACHINE LEARNING METHODS FOR PIPELINE 15

3.1 Data scaling 15

3.1.1 Min-Max normalization 15

3.1.2 Maximum absolute scaling 15

3.1.3 Standard score standardization 16

3.1.4 Robust scaler 16

3.2 Data balancing 17

3.2.1 Class weights 17

3.2.2 Synthetic Majority Oversampling Technique (SMOTE) 18

3.2.3 Adaptive Synthetic Sampling Approach (ADASYN) 19

3.3 Classification 19

3.3.1 k-Nearest Neighbors (k-NN) 19

3.3.2 Linear Support Vector Machine (SVM) 21

3.3.3 Radial Basis Function (RBF) kernel SVM 22

3.3.4 Decision Tree Classifier 23

3.3.5 Random Forest Classifier 24

4 EXPERIMENTS 26

4.1 Data 26

4.2 Evaluation criteria 27

4.3 Description of experiments 27

4.3.1 Chosen methods for experimentation 27

4.3.2 Linear separability 28

4.3.3 Grid search 29

4.4 k-fold cross-validation 30

4.5 Results 31

5 DISCUSSION 34

5.1 Current study 34

5.2 Future work 35

6 CONCLUSION 36

REFERENCES 37

1 INTRODUCTION

1.1 Background

Material classification is a problem where material is classified into classes based on the material's features. These features can be dimensions, mass, spectral properties among others, which from prior knowledge are known to differ between classes. Material classification has its applications in quality control such as in food safety where material is classified to determining ripeness or possible contamination [1]. Material classification is also utilized in multi-class classification such as classifying marine plastic debris [2], and in recycling various scrap metals [3].

This study focuses on automating industrial material classification with machine learning methods. More specifically this study focuses on determining the best pre-processing–data balancing–classification pipeline for classifying specific industry material. Currently the classification is done manually from concentration measurements by using past knowledge, notes and industry experience to determine the correct material code, which is slow, consumes working hours and requires extensive knowledge which can all lead to misclassification in a rapid paced industrial environment. For these reasons an automated solution is required which sets the goal of this study to implement a machine learning model, that is able to rapidly and accurately classify material to save working hours and minimize human error in the process. The industry has a non-public record of 30 000 material classifications with 200 different classes which are used in the training, validation and testing of the model. Singular records consist of chemical composition information processed from raw spectrometer data paired with correct material code.

Multiclass classification is needed to solve the problem, for which supervised learning is implemented with known classes. The architecture in Figure 1 shows the three main components of the classification architecture: pre-processing where data is normalized and filtered, data balancing where unbalanced data is addressed and classification that is a trained classifier which is responsible for the final decision.

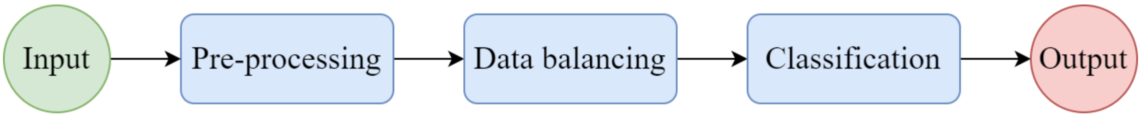

Figure 1. Architecture for the model.

1.2 Objectives and delimitations

The main objective of this thesis is to experiment on different pre-processing, data balancing and classifier combinations to find the optimal combination which classifies the material most accurately. The classifier is then compared with the latest data to determine the final test accuracy. The specific objectives of the thesis are:

1. Perform preliminary feasability test whether the data is suitable for machine learning classification.
2. Compare possible methods for pre-processing, data balancing and classification by experimentation with hyperparameters.
3. Select the best combination.

One delimitation is that the minority classes which have too few samples to be accurately learned, are ignored.

1.3 Structure of the thesis

The thesis is structured as follows: Chapter 2 provides an overview of prior research on material classification in industrial settings using spectrometer measurements. Chapter 3 describes the proposed evaluation methods, which are based on the best practices derived from the previous work. This chapter is further subdivided into pre-processing, data balancing, and classification subchapters, which provide a detailed view of the proposed methods. Chapter 4 describes the experiments and the results obtained. Chapter 5 analyzes the results and describes possible future research. Finally, Chapter 6 concludes the thesis by summarizing the work conducted and the achieved results.

2 MATERIAL CLASSIFICATION IN INDUSTRIAL APPLICATIONS

2.1 Classification for recycling

2.1.1 Plastic classification

With the ever increasing plastic waste the need for plastic classification for recycling is becoming more concerning issue where rapid and accurate machine learning solutions are needed [4]. Classification of plastics has been studied in the classification of marine plastic debris [2], in the classification of recycled plastic waste [5] and in the classification of black plastics [6]. Example of plastic classification using spectrometer features is visualized in Figure 2 [7].

The study by A. Michel, et al. [2] focused on classification of marine plastic debris using different spectroscopy techniques such as Laser Induced Breakdown Spectroscopy (LIBS), Attenuated Total Reflectance–Fourier Transform Infrared Spectroscopy (ATR–FTIR) and Near-Infrared Spectroscopy (NIR). The goal of the study was to determine by experimentation the spectroscopy - classifier pair with the highest accuracy across all customer grade plastic types. For experiments the plastic samples were selected with differing color, opaqueness and thickness, which were then rinsed and cleaned for the spectrometer measurements to eliminate surface contamination. Classification was done using k-Nearest Neighbors (k-NN), Linear Discriminant Analysis (LDA) and Support Vector Machine (SVM) classfiers.

The study by Y. Yang, et al. [5] focused on classification of plastic waste using portable NIR spectrometer to achieve effective recycling. The goal of the study was to test different pre-processing methods with Principal Component Analysis (PCA) to cluster different plastic types in three principal component space for dimension reduction and for visual inspection. After the pre-processing and dimension reduction different classifiers were tested to find by experimentation the best classifier for the given problem.

The study by F. Gruber, et al. [6] focuses on the problem of black plastic classification using spectroscopy features. The goal of the study is to experiment with different spectroscope types to determine the ones that are not absorbed by the black surface of the plastics and have the strongest discrimination between different plastic types. For experiments the plastic samples were cryogenicically grinded and measured with spectrometer, pre-processed and classified using k-NN, LDA and SVM classifiers.

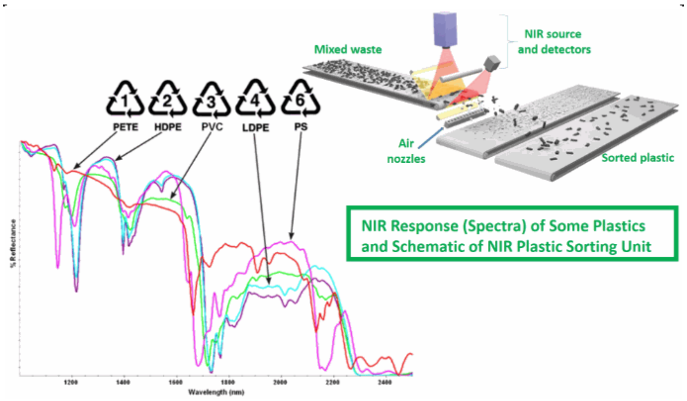

Figure 2. Near-infrared spectroscopy set-up for plastic sorting [7].

2.1.2 Metal classification

According to World Steel Association from the year 2000 to 2020 the production of crude steel has doubled which reflects the increase in consumption of steel products [8]. This increase in consumption has led to increased steel recycling which in part is possible due to the biggest steel based products being transportation machinery, industrial machinery and electrical equipment which can be efficiently recycled [9]. Metal classification for recycling has been studied with the classification of scrap metal [3] and the classification of aluminum grades [10].

The study by J. Gurell, et al. [3] studied the classification of scrap metal by utilizing LIBS for rapid metal scrap classification. The goal of the study was to determine if it is possible to discriminate between different metals and also different steel alloys. To conduct the experiments the outer layers of the scarp metal sample were evaporated from the measurement region using pulse laser. This was done to remove any possible surface contaminations such as corrosion or paint. The measurements were then measured using LIBS and classified using LabVIEW-software.

The study by D. Jossue, et al. [10] studied the classification of aluminum scrap to three commercially interesting classes by utilizing LIBS and machine learning. The goal of the study was to determine if LIBS measurements and trained classifiers can be used to discriminate between different metals and different aluminum sub-classes. The measurements were conducted using setup visualized in Figure 3 and pre-processed using simple binning to discretize the spectra. For the classification different traditional classifiers were utilized such as LDA, SVM and RF. More advanced CNNs were also experienced with different pretrained networks.

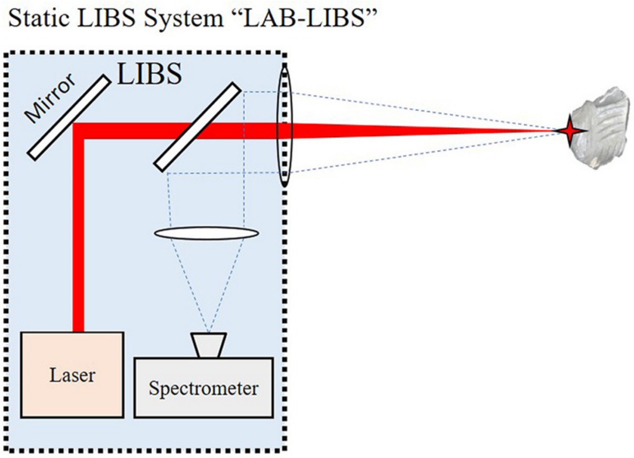

Figure 3. Laser-induced breakdown spectroscopy for aluminum classification [10].

2.2 Quality control for products

Quality control and product safety is an important issue to society as a whole, as they form the basis of human health which impacts directly upon social development, environment and stability [11] [12]. The complexity of quality control comes from hard to detect faults such as counterfeit medical products and seafood contaminated by heavy metals. Quality control for products has been studied in the detection of heavy-metal contaminated seafood [1] and quality control in pharmaceuticals [11].

The study by G. Ji, et al. [1] focused on detecting heavy-metal contamination in *Tegillarca granosa* clam species which is one of the most important commercial seafood product in East Asia [1]. The goal of the study was to experiment if it is possible to discriminate between clean and heavy-metal contaminated samples of *Tegillarca granosa* by using spectroscopy and machine learning. To conduct the experiment the samples were cleaned and prepared which were then analyzed with spectrometer displayed in Figure 4, after which the spectrometer measurements were pre-processed and finally classified as clean or contaminated using SVM and RF classfiers.

The study by J. C. Martinez, et al. [11] focuses on similarity classification of the pharmaceutic called acetaminophen based on spectrometer measurements obtained from the samples. The goal of the study was to compare acetaminophen products from thirteen different pharmaceutical laboratories against a control supplied by governmental health department. The measurements were conducted using Raman spectroscopy and pre-processed with PCA, after which Naive Bayes classifier was used to classify similarity to control sample.

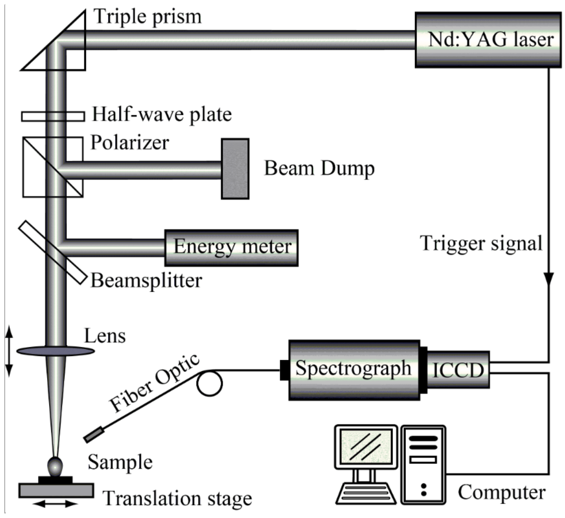

Figure 4. Diagram of the laser-induced breakdown spectroscopy (LIBS) set-up utilized in seafood quality control [1]

2.3 Summary

Although the studies differ by topic and by the methods which the observation data is gathered, they all have in common the problem of finding the optimal classifier by experimentation. The optimal classifier is the classifier with the highest validation accuracy that is found by cross-comparing different classifiers with hyperparameter tuning. The summary of the optimal classifiers with different studies is presented in Table 1. The accuracies were selected from experiments that were conducted on pre-processed spectrometer data to reflect more accurately the industry problem of this thesis.

Furthermore Convolutional Neural Network (CNN) classifiers were ignored because the industry problem of this thesis uses pre-processed chemical composition data, that does not contain any structural information which is essential for CNN models [13]. Other deep learning methods were also ignored because from prior knowledge it is known the problem is linear and simplistic although labor-intensive.

In a summary, although the studies are from different fields with different objectives and datasets, they all share a commonality in using a spectrometer for measuring spectral properties of a specific material and certain pre-processing to turn the spectra into discretized chemical composition measurements. With this information the classifiers mentioned in Table 1 are studied more closely and experimented with in this thesis. In Table 1 accuracy is the overall ratio of correct guesses (Equation 13), while precision is the ratio of true positive predictions over every true prediction [13]. The information of the data used in the studies is displayed in Table 2. Compared to the other studies the study by D. Jossue et al. [10] doesn't have equal class distribution, but instead contains the three classes in proportions of 44%, 27% and 29% of total data.

Table 1. Comparison of classifiers used in the studies.

Classifiers					
Study	k-NN	LDA	SVM	RF / ENSEMBLE	CNN
D. Jossue et al. [10]	-	0.73^p	0.66^p	0.80^p	0.80^p
A. Michel et al. [2]	0.97	0.99	0.92	-	-
Y. Yang et al. [5]	0.99	-	0.99	-	-
F. Gruber et al. [6]	0.90	0.86	0.87	0.90	0.94
G. Ji et al. [1]	-	-	0.87	0.93	-

Table 2. Comparison of the data used in the studies.

Data			
Study	Number of classes	Number of total samples	Data type
D. Jossue et al. [10]	3	983	LAB-LIBS
A. Michel et al. [2]	6	180	ATR-FTIR
Y. Yang et al. [5]	10	2000	Pynect NIR-S-G1 NIR
F. Gruber et al. [6]	12	400	Fluorescence spectrometer
G. Ji et al. [1]	5	150	LIBS

3 MACHINE LEARNING METHODS FOR PIPELINE

3.1 Data scaling

Data scaling is the first step in the three-step pipeline before data balancing and classification. The aim of data scaling is to scale features to similar magnitude, so that no feature would have disproportionate influence on the classification [14] [13]. Some classifiers are very sensitive to feature scales which if not supplied with scaled data can lead to erroneous learning and evaluation [14].

Data scaling to a certain range is used to scale features into same range of magnitude. The simplest form of scaling is Min-Max normalization [13] where all features are scaled to the range of [0,1]. Normalization is a different form of scaling, where features are scaled to have standard deviation of 1. Different methods of scaling are tested including option of not scaling the data at all.

3.1.1 Min-Max normalization

Min-Max normalization [13], which transforms the data to the range of [0,1] where minimum value is 0 and maximum value is 1. Min-Max normalization is represented by

$$Z = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{1}$$

where x is the data, $\min(x)$ is the smallest x-value, $\max(x)$ is the largest x-value and Z is the transformed data. For replication purposes the minimum and maximum values are stored in a file.

3.1.2 Maximum absolute scaling

Maximum Absolute Scaling (MaxAbs scaling) [15] only scales the observations without centering or shifting them in anyway. MaxAbs scaling is represented by the function

$$Z = \frac{x}{\max(|x|)} \tag{2}$$

where x is the data, $\max(|x|)$ is the maximum absolute value of x and Z is the transformed data. For replication the maximum absolute value is stored in a file.

3.1.3 Standard score standardization

Standard score [13] also known as z-score is a method used to transform data to zero mean and unit standard deviation. For standard score standardization the mean and standard deviation of the data are calculated. Using the mean and standard deviation calculated from the data the same data is then transformed using functions

$$\mu = \frac{\sum x_i}{N}, \qquad \sigma = \sqrt{\frac{\sum x_i - \mu}{N}}$$

$$Z = \frac{x - \mu}{\sigma}$$

where x is the observations, N is the number of observations, μ is mean, σ is standard deviation and Z is the transformed data. For the purpose of replication for scaling the test data, the mean and standard deviation calculated from the training data are stored in a file.

3.1.4 Robust scaler

Robust scaler [15] scales and centers the value by using interquartile range where Q_1 is the median of the lower half of the data and Q_3 is the upper half of the data. To calculate the quartiles the data is to be sorted from lowest to highest and split from the median into the lower and upper halves. Then the medians from the lower and upper halves are calculated and assigned as Q_1 and Q_3 respectively. With these quartiles the equation of the scaling can be expressed as

$$Z = \frac{x - Q_1}{Q_3 - Q_1}$$

where x is the input value and Z is the scaled value.

3.2 Data balancing

The rare class problem in classification is a situation where the classes are imbalanced by containing minority and majority classes [16]. More formally imbalanced dataset is a dataset that contains significant or extreme disproportionate of class samples for different classes [17]. In real life classification problems the rare classes and rare cases are usually the most interesting ones in the form of anomaly detection and otherwise rare occurrences [17]. Real life examples include credit card fraud detection [18] anomaly detection in videos [19] and object recognition [20].

3.2.1 Class weights

For class weighting, very simplistic implementation is applied, which is calculated for each class. The weighting is based on simplified version of the method introduced by Gary King and Langche Zeng [21] by calculating weight for each of the classes with the inverse of its sample frequency as

$$w_i = \frac{N}{K \cdot n_i}$$

where N is the total number of samples in the whole dataset, K is the number of classes in total and n_i is the number of samples in the class i.

With this method, the minority classes are assigned greater weights that increases the importance of individual samples. On the other hand, the majority classes are assigned lower weights, making the individuals less important. However, the weights are calculated so that the weight of the class multiplied by the number of samples in the class always equals same value for all classes, with this each class has equal importance as a whole in the learning process. Visualization of example weights are displayed in Figure 5.

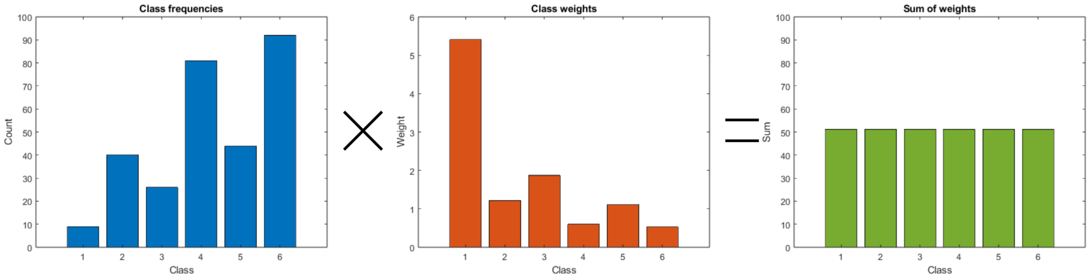

Figure 5. Example of inverse frequency weights

3.2.2 Synthetic Majority Oversampling Technique (SMOTE)

SMOTE is an oversampling technique which aims to over sample minority classes by generating new data points by interpolating the original data points. SMOTE aims at generating large and less specific decision regions instead of small and specific regions by introducing new synthetic data points joining existing data points with k nearest neighbors [22]. SMOTE is presented in Algorithm 1.

Algorithm 1 SMOTE [22]

```
Input: Sample matrix(k, M, N)
Output: New sample(1, N)
  1. Set random sample from the class as pivot,
  2. Select the k-nearest neighbors from the pivot sample,
  3. Calculate the difference vector between one randomly selected neighbor and the pivot,
  4. Multiply the difference vector with a random number from [0,1] constant distribution,
  5. Generate the new data point as $point = pivot + vector$,
  6. Repeat the whole process as many times as required,
```

This operation is repeated until the desired number of synthetic samples is reached. The algorithm is visualized in Figure 6.

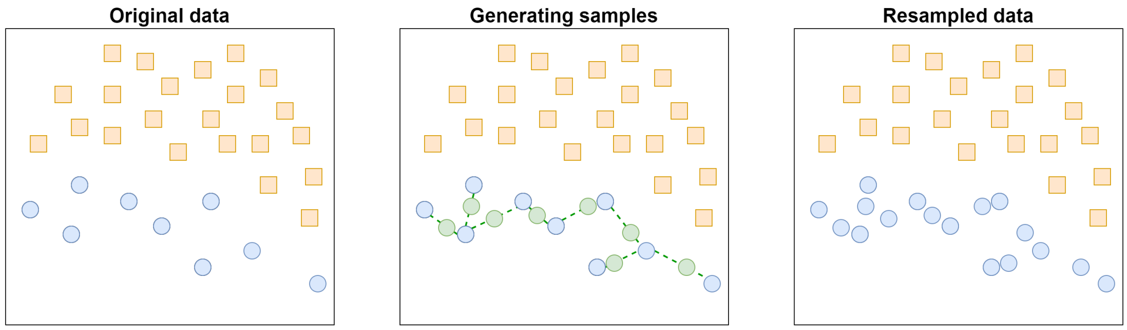

Figure 6. SMOTE visualized

3.2.3 Adaptive Synthetic Sampling Approach (ADASYN)

Adaptive Synthetic Sampling Approach (ADASYN) is synthetic oversampling technique used in imbalanced learning problems to synthetically sample minority classes. Compared to SMOTE, ADASYN focuses on generating data points for minority classes that are harder to learn than those that are easier to learn [23]. ADASYN is described in Algorithm 2.

3.3 Classification

Classification in machine learning is a supervised learning task, where the goal is to assign labels or classes to data samples based on their features [13] [14] [24]. The primary objective of classification is to determine decision boundaries or functions that can distinguish different classes from each other. These decision boundaries or functions, receive a set of features as input and output a class label that corresponds to the class of the given instance. The correct labeling is learned from a training dataset, which pairs different samples with their known class labels. By doing so, the training algorithm can penalize the classifier for any incorrect classifications and gradually improve its accuracy in classifying new instances.

3.3.1 k-Nearest Neighbors (k-NN)

k-NN is a supervised classifier that uses the k number of nearest train samples to vote for the class of the pivot point [25] [26]. For this thesis k-NN was used as the benchmark

Algorithm 2 ADASYN [23]

```
Input: Sample matrix(k, M, N)
Output: New sample(1,N)

1. Calculate the degree of class imbalance $d = m_s/m_l$,
2. If $d < d_{th}$, where $d_{th} \in (0, 1]$ is the maximum tolerated degree of imbalance,
    (a) Calculate the number of synthetic data samples that are to be generated for the minority class
            $$G = (m_l - m_s) \times \beta \qquad (3)$$
        where $\beta \in [0, 1]$ is the desired level of balance. $\beta = 1$ stands for fully balanced data set,
    (b) For each sample $x_i \in minorityclass$, find k nearest neighbors by using Euclidean distance, and calculate ratio $r_i$
            $$r_i = \Delta_i/k, \qquad i = 1, \ldots, m_s \qquad (4)$$
        where $\Delta_i$ is the number of samples in the k nearest neighbors of $x_i$,
    (c) Normalize $r_i$ with
            $$\hat{r_i} = r_i / \sum_{i=1}^{m_a} r_i \qquad (5)$$
        so that $\sum_i \hat{r_i} = 1$,
    (d) Calculate the number of synthetic samples to generate for each minority sample $x_i$
            $$g_i = \hat{r_i} \times G \qquad (6)$$
        where $G$ is the number of samples to be generated for the minority class as defined in Equation 3,
    (e) For each $x_i$ generate $g_i$ synthetic data samples according to the steps by looping from 1 to $g_i$,
        i. Choose random data sample $x_{zi}$ from the k nearest neighbors for data $x_i$,
        ii. Generate the synthetic data sample with
                $$s_i = x_i + (x_{zi} - x_i) \times \lambda \qquad (7)$$
            where $(x_{zi} - x_i)$ is the difference vector in $n$ dimensional space and $\lambda \in [0, 1]$ is a random number,
```

classifier due to its' simplistic and naive nature, and because k-NN methods have reached good results in plastic classification [2], mould classification [27] and other material classification problems [28]. The fundamental concept of the k-NN algorithm involves the classification of samples, which is based on their closest train samples through a majority voting scheme. In this scheme, k nearest neighbors of the classified sample are selected to vote with their own respective classes. Ultimately, the class that gains the highest number of votes is the decision for the predicted sample. The algorithm for deciding the class of an arbitrarily chosen sample is described in Algorithm 3.

```
Algorithm 3 K-NN [22]
Input: k, Observation space(M, N), Sample(M, 1)
Output: Class label(1)
  1. Calculate the Euclidean distance $d_i$ from the sample point $x$ to every observation $m_i$,
         $d(m_i, x) = \sqrt{(m_{i1} - x_1)^2 + (m_{i2} - x_2)^2 + \cdots + (m_{in} - x_n)^2}$    (8)
  2. Select the $k$ nearest observations by taking the observations with smallest distances $\min_k(d_i)$,
  3. Get the corresponding labels $\omega_i$ for the $k$ nearest observations $\min_k(d_i)$,
  4. Calculate the most frequent class label $\hat{\omega}$ by taking mode from the labels $\omega_i$
         $\hat{\omega} = \text{mode}(\omega_1, \omega_2, \ldots, \omega_i)$    (9)
  5. Return the most frequent class label $\hat{\omega}$ as the predicted label for sample $x$,
```

3.3.2 Linear Support Vector Machine (SVM)

The support vector machine is a type of supervised learning classifier that constructs a set of hyperplanes to divide the high dimensional input space into class specific regions [29]. The basic idea behind SMV is to find hyperplanes that can best separate the input data into distinct classes. SVM constructs these hyperplanes by using support vectors which are data points that best separate the different classes, which also maximize the margin between the different classes. The hyperplanes are fitted using optimization that maximizes the margin and minimizes the classification error. Figure 7 displays a simple 2-dimensional SVM where two classes blue and green are separated by a hyperplane $w * x - b = 0$ where w is the normal vector of the hyperplane, x is the position vector in the 2-dimensional space, and b is the margin value. The hyperplane is spanned by using the support vectors, which are the bold dots in the figure. These support vectors lie closest to the hyperplane and define the margin of the SVM. The margin is bounded by the dotted lines in the figure, which intersect the support vectors.

For the first SVM classifier a linear kernel is utilized which retains the data in its original space without any transformations. Linear kernel SVM has reached good results with plastic classification [2] and heavy-metal contamination classification [1], making it viable classifier to compare and study. Linear SVMs works well with linearly separable data, but cannot perfectly fit problems with non-separable data [30]. Using this knowledge it is possible to estimate the linear separability of the data by fitting linear SVM with zero error tolerance. If the fit is possible, it indicates the data might be linearly separable, but if not the data contains non linearity [14].

SVM classifier is visualized in the Fig 7 [31]. In the figure two classes blue and green are separated by a hyperplane $w * x - b = 0$. The purpose of this hyperplane is to act as the decision boundary, that can classify given input sample into certain class based on which side of the decision boundary the sample resides.

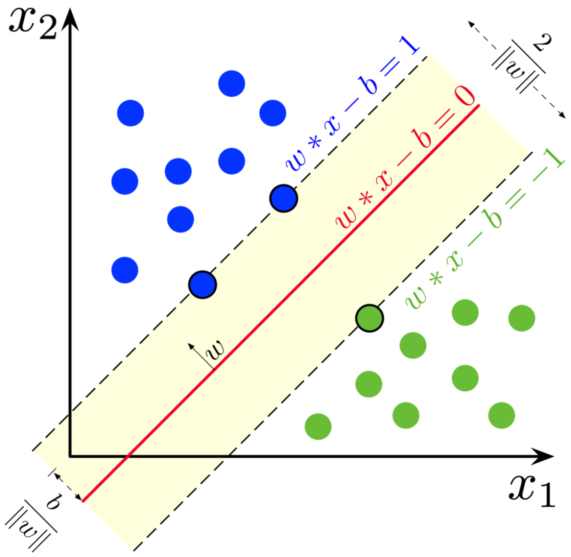

Figure 7. Example of SVM decision boundary with the b margin which separates the blue class from the green class [31].

3.3.3 Radial Basis Function (RBF) kernel SVM

Radial Basis Function (RBF) is a kernel that is utilized for the previously mentioned SVM [30]. Compared to linear kernel the RBF kernel maps the input space into non-linear, distance based space from a center point. The RBF mapping can be represented by

$$K(X_1, X_2) = e^{-\frac{||X_1 - X_2||^2}{2\sigma^2}} \tag{10}$$

where X_1 is the support vector from which the Euclidean distance is calculated from, X_2 is any observation in the observation space to which the distance is calculated to, σ is hyperparameter that determines the standard deviation used in the distance metric [32].

3.3.4 Decision Tree Classifier

The decision tree is a classifier that classifies new samples into their corresponding classes using one or several decision functions in a successive manner [33]. Furthermore decision trees can be visualized by a tree diagram with one root node, a number of interior nodes and a number of terminal nodes. Classification starts from the root node moving downwards based on the result of the node's decision function into a number of interior nodes with each their own decision functions, and finally ending in a terminal node which will be the final class prediction. Simple decision tree for arbitrary $\{x, y, z\} \in R$ inputs is visualized in Figure 8.

Decision trees are commonly trained by using the Gini impurity criterion [34] or the Shannon information gain criterion [35]. The training criterion determines how the algorithm chooses the best split at each node of the tree. Gini impurity measures the degree of impurity of a set of samples, and the algorithm chooses the split that minimizes the Gini impurity of the child nodes using the following function:

$$G(\mathbf{n}) = 1 - \sum_i n_i^2 / N^2 \tag{11}$$

where n is the number of samples in each class, and N is the total number of samples at a given node [34]. The Shannon information gain, on the other hand, measures the reduction in entropy achieved by a split, and the algorithm chooses the split that maximizes the information gain of the child nodes using the following function:

$$h(\mathbf{n}) = -\sum_i \frac{n_i}{N} - \sum_i n_i \log n_i \tag{12}$$

where similarly n is the number of samples in each class, and N is the total number of samples at a give node [34]. The training process continues recursively until all leaf nodes are pure, or until a stopping criterion is met, such as reaching a maximum depth or a minimum number of samples per leaf node. Once the tree is trained, it can be used in classification of new samples by inferring the trained model.

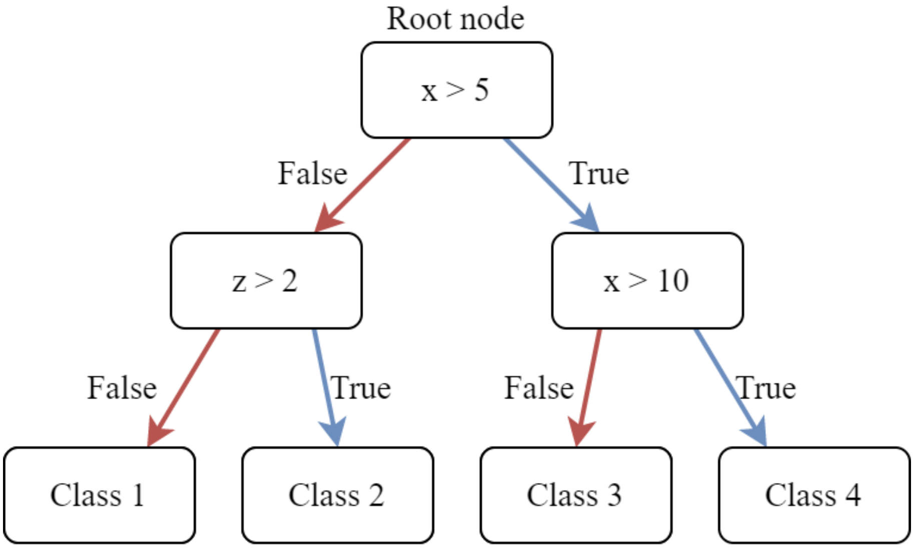

Figure 8. Decision tree visualized for $\{x, y, z\}$ inputs with four different classifications.

3.3.5 Random Forest Classifier

The random forest is a classifier that utilizes multiple decision tree classifiers by letting the decision trees vote for the most popular class. More formally random forest is a classifier consisting of a collection of tree-structured classifiers $\{h(\boldsymbol{x}, \Theta_k), k = 1, \dots\}$ where the Θ_k are independent random vectors where each tree casts a vote for the most popular class for input $\boldsymbol{x}$ [36] [37]. The basic idea of random forest with voting system is visualized in Figure 9 [38].

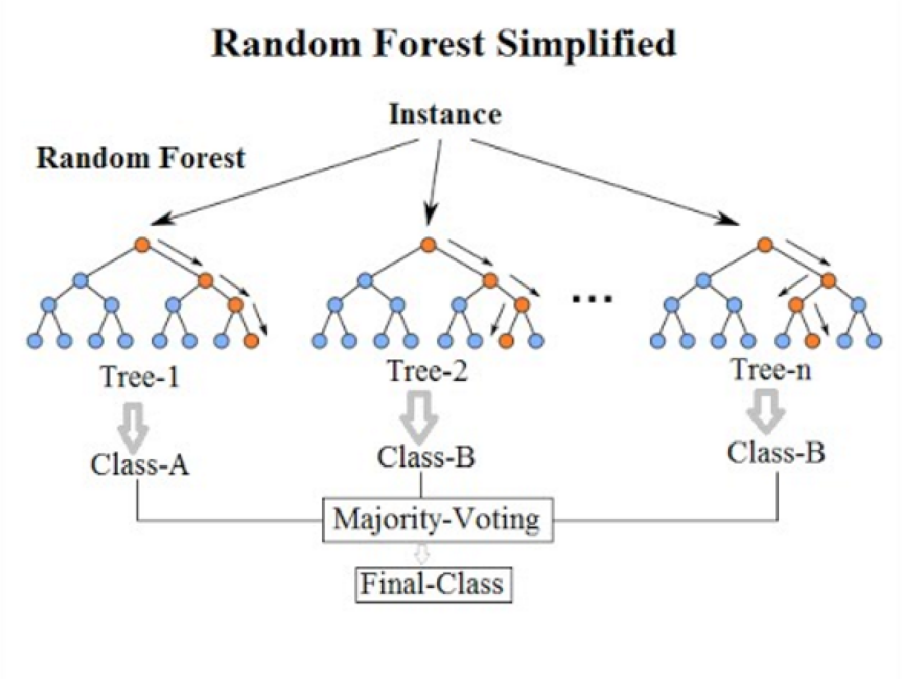

Figure 9. Simplified random forest [38].

4 EXPERIMENTS

4.1 Data

The data used in this study consists of 30,000 material classifications with 200 different classes which are used for training, validation, and testing each selected machine learning model. Each record in the dataset contains chemical composition information from preprocessed spectrometer data and its associated material code. As is with many real life problems with multiple classes, the studied dataset presents a rare class problem [39], as evidenced by the class frequency figure depicted in Figure 10. In the figure the samples have been binned into 250 bins based on their class and sorted from most common classes to rare classes. The subclass problems, where one class is subsumed within another class, were initially identified in the data. However these cases were considered insignificant by the industry and were consequently excluded from the dataset.

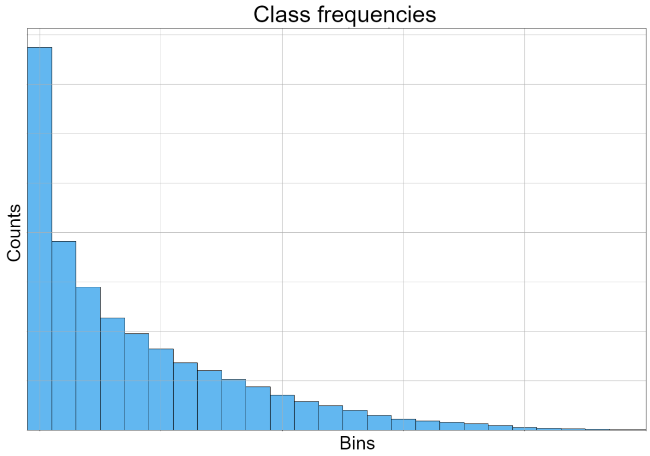

Figure 10. Histogram of the class frequency in the dataset.

The data was obtained from an industrial application that has remained relatively unaltered for several years, which allows the use of dated data in the training without introducing biases into the learning process. Furthermore, the bias is controlled by the fact that the data was collected by on-site experts with extensive experience in the industry, ensuring the reliability of the class labels.

More specifically the structure of the data is shown in Table 3, where each row starts with an unique identifier index, which is followed by several dozen chemical composition concentration measurements, and ended with the corresponding class label which is a class unique serial number. Furthermore the data were provided in the csv file format.

Table 3. Mockup of the data structure used for the model training.

Example data structure					
Index	Chemical 1	Chemical 2	Chemical 3	Chemical n	Class label
1	...	...	...	...	...
2	...	...	...	...	...
3	...	...	...	...	...
n	...	...	...	...	...

4.2 Evaluation criteria

The evaluation of the models was done using cross-validated mean accuracy, where the accuracy is multi-class accuracy which is the proportion of correct classes over every tested class as [13]

$$\text{acc(f, D)} = \frac{1}{m}\sum_{i=1}^{m}(f(x_i) = y_i) \tag{13}$$

where the accuracy function has two input parameters, the model's inference function f and the evaluated test data D, where D contains the features x and the true labels y, and finally m is the length of the evaluated test data D. In the function, for each of the i data element a comparison is made whether the inference of the function f returns the true class label y_i with input x_i. If every inference with input x_i returns the correct true label y_i the accuracy is 1.

4.3 Description of experiments

4.3.1 Chosen methods for experimentation

The chosen methods for experimentation are chosen from the methods utilized by the previous studies which achieved good results. The following methods were chosen to be tested for pre-processing the data:

1. Min-Max normalization
2. Maximum absolute scaling
3. Standard score normalization
4. Robust scaler

For data balancing the following methods were chosen to be tested:

1. Class weights
2. SMOTE
3. ADASYN

For classification the following methods were chosen to be tested:

1. k-NN
2. SVM
3. RBF
4. Decision tree classifier
5. RF

4.3.2 Linear separability

Linear separability of the data was tested with visual observations of PCA dimension reduced sets and by attempting to fit linear SVM to the data. The first method of visual observation builds on the property of PCA where the principal components are a linear combination of the original factors [40]. With this information, if two or more classes are clearly linearly separable with visual PCA inspection with reasonably high explained variance of the principle components, it can be used as indicative of linear separability. Visualization using PCA is displayed in Figure 11. The second method is to evaluate the linear separability by fitting linear SVM to the data. If the data can be classified using linear SVM with reasonably high accuracy that is indicative of linear separability, which is a direct property of linear SVM where the SVM algorithm aims at separating the classes by using a combination of linear hyperplanes [29] [41].

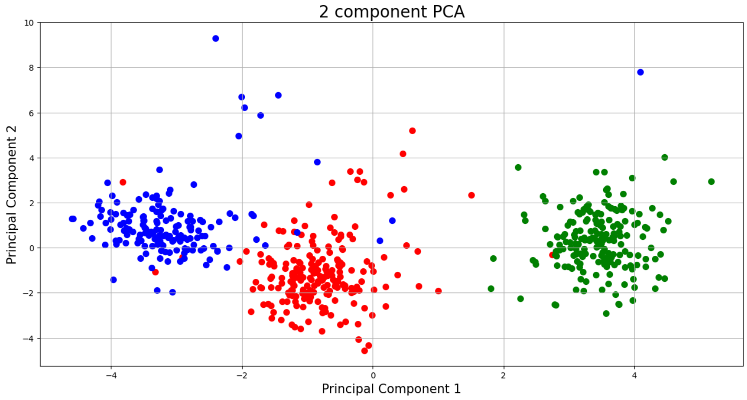

Figure 11. Three most common materials in two component PCA

4.3.3 Grid search

The grid search is an exhaustive hyperparameter tuning algorithm for given discrete set of of parameter options [15]. Grid search works by iterating through every possible combination of given parameter options and records the evaluation metrics for each combination. For example, when hyperparameter tuning RBF SVM with two parameters C and $gamma$ the parameter options are as follows:

$$\begin{aligned} \ldots &= \{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\} \\ C_{params} &= \{0.001, 0.01, 0.1, 1, 10, 100, 1000\} \\ Gamma_{params} &= \{0.001, 0.01, 0.1, 1, 10, 100, 1000\} \end{aligned}$$

which represents log-uniform distribution which is a common way of searching for hyperparameters where magnitudes are compared rather than linear scales [15]. The grid search is a useful tool for iterating over a range of parameters by comparing sets of parameters, instead of attempting to test every parameter combination which is impossible [13]. However, there is a trade-off between computational cost and quality of estimation, which leads to the challenge of selecting the optimal parameters sets.

Another common way of searching for hyperparameters is using a randomized search which samples given parameter distributions for given n times rather than systematically going through every given combination as the exhaustive grid search does [42] [15]. Although the randomized search has been studied to result in models that are as good or even better than those tuned with grid search [42], the grid search was still utilized in most cases where the models had a limited number of hyperparameter options, in order to fully search the hyperparameter space for better understanding of the models.

4.4 k-fold cross-validation

In k-fold cross-validation, the data is split into k subsets (or folds) of similar sizes, where the first fold is selected as the validation data and the rest of the folds are used to train the model [14]. For optimal results each subset should maintain the original data distribution which can be achieved by stratified sampling [13]. After the first model has been trained and validated, the second model is trained and validated by using the second fold as validation and the rest of the folds as training. The final accuracy of the k-fold cross-validated model is the average of the k training iterations. The concept of k-fold cross-validation is visualized in Figure 12 [15].

Although k-fold cross-validation is a computationally expensive process, since the model is trained k times instead of traditional one time, it offers a more robust and accurate estimation of the model than a singular training would. Because of these properties k-fold cross-validation has become a widely used standard practice in model training [13] and machine learning libraries [15].

The rationale for utilizing k-fold cross-validation as an accuracy metric, is the uncertain distribution of the training data, which poses challenges when using a traditional random split into training and validation sets. The traditional random split does not ensure that the validation data accurately represents the problem, especially in rare class problems. In rare class problems, if all samples of a rare class X are selected as validation data, the model will fail to learn the rare class X, resulting in decreased accuracy. Also, if a rare class X is absent from the validation set, the accuracy metric will increase despite the model's inability to learn the rare class X. Therefore k-fold cross-validation provides a more reliable and unbiased approach to estimate model performance by reducing the impact of data randomness and ensuring that classes are better represented in both training and validation sets. [43] [15]

For this thesis, 5-fold cross-validation was utilized. This means that the data was split into five subsets of equal size with each subset serving as validation data once, while the rest of the data was used for training. The results obtained from each of the five iterations were then averaged to provide accuracy metric for each model.

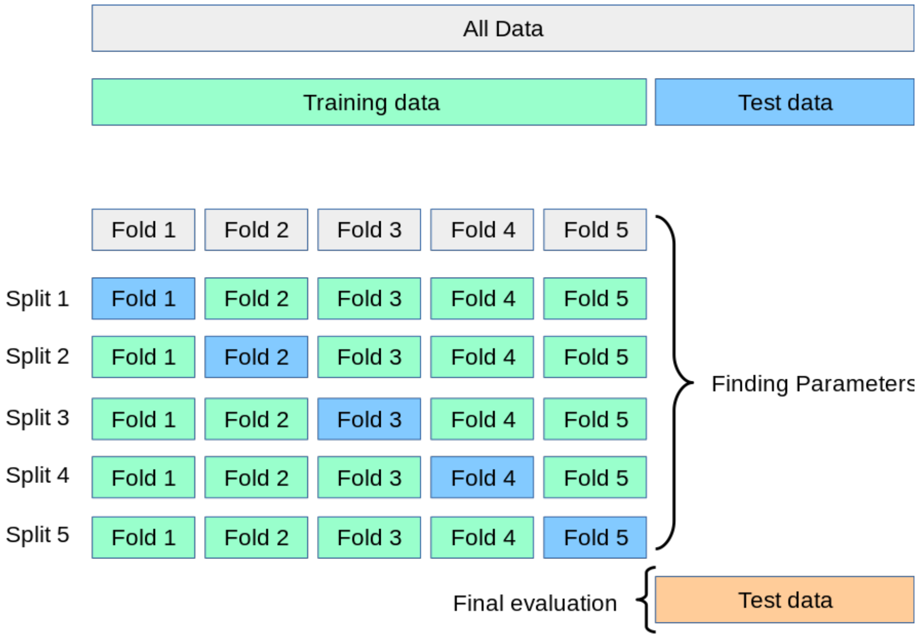

Figure 12. Cross-validation visualized [15]

4.5 Results

The findings were gathered through hyperparameter optimization of each model, which involved exhaustive testing of discrete options or evaluation of values at various intervals for continuous variables. Since the training was computationally heavy and thus was time-consuming significant time, the testing was conducted in multiple batches, each of which trained the model 100 times, utilizing given pre-processing, data balancing, and classifier options. As the training progressed, models that could not be further improved due to their limited hyperparameter options, such as k-NN, AdaBoost, and LDA, were excluded from subsequent testings. These methods were exhaustively searched, and were found inferior to the other methods with higher hyperparameter spaces as seen in Table 5. The models with higher hyperparameter spaces were SVM, RBF and RF which were tested further with more batches.

Table 4 presents the top five models sorted by their validation accuracy. The hyperparameters of the models were opitmized using validation accuracy. The test accuracy is the model's accuracy for unseen data that is not filtered which explains the drop in accuracy. Based on these results in Table 5 RF classifier has the highest accuracy for this problem, with SVM with RBF kernel following as the second best option. Conversely, k-NN, LDA and AdaBoost had the lowest accuracies and in so are the inferior alternatives. From the results, LDA and random forest are the most stable models as they have minimal standard deviation of accuracy compared to other models. On the contrary, k-NN and AdaBoost are the most unstable models having the highest standard deviations, which are over twenty times greater than those with LDA and random forest.

Table 6 and Table 7 display the accuracies for RF models with different pre-processing and data balancing techniques. The results suggest that the selection of pre-processing and data balancing methods has a minimal effect on the model's performance, especially when compared to other classifiers that have significantly higher standard deviations of accuracy, as shown in Table 1. Also, the optimal combination of pre-processing and data balancing for Random Forest is the use of unscaled data that has been balanced using either SMOTE with k=5 or SMOTE with k=10, as their accuracies are highly similar.

Table 4. Comparison of the top-5 models ranked by validation accuracy.

Top-5 models					
Rank	Validation accuracy	Test accuracy	Pre-processing	Data balancing	Classifier
1	0.9546	0.7338	Unscaled data	SMOTE k=10	Random forest
2	0.9544	0.7312	Minmax scaler	SMOTE k=5	Random forest
3	0.9542	0.7350	Standard scaler	SMOTE k=5	Random forest
4	0.9535	0.7317	Maxabs scaler	SMOTE k=5	Random forest
5	0.9535	0.7327	Maxabs scaler	SMOTE k=10	Random forest

Table 5. Comparison of different classifiers.

Validation accuracies of different classifiers				
Rank	Classifier	Mean accuracy	Max accuracy	Standard deviation of accuracy
1	Random forest	0.951	0.955	0.004
2	SVM RBF	0.926	0.940	0.014
3	SVM linear	0.924	0.940	0.012
4	LDA	0.916	0.924	0.007
5	k-NN	0.833	0.931	0.074
6	AdaBoost	0.405	0.525	0.086

Table 6. Random forest pre-processing, with highest accuracies bolded.

Validation accuracies of RF with different pre-processing options			
Pre-processing	Mean accuracy	Max accuracy	Standard deviation of accuracy
Unscaled data	**0.953**	**0.955**	0.001
Maxabs scaling	0.951	0.954	0.004
Robust scaler	0.952	0.953	0.001
Minmax scaler	0.952	0.954	0.002
Standard scaler	0.949	0.954	0.006

Table 7. Random forest data balancing, with highest accuracies bolded.

Validation accuracies of RF with different data balancing options			
Data balancing	Mean accuracy	Max accuracy	Standard deviation of accuracy
No balancing	0.947	0.951	0.004
SMOTE k=5	**0.951**	0.954	0.006
SMOTE k=10	0.953	**0.955**	0.001
ADASYN	0.952	0.953	0.001

5 DISCUSSION

5.1 Current study

The obtained results of each method with the considered new dataset are similar to the previous work presented in Table 1. Specifically, the accuracies achieved by the SVM and random forest models are found to be comparably high, consistent with the findings of prior work. Moreover, in the previous studies that utilized both SVM and random forest methods, the latter was observed to outperform the former, as is also confirmed in this study. However, the k-NN algorithm yielded significantly lower accuracy in contrast to the previous work, and the underlying cause of this discrepancy remains unclear.

When comparing machine learning models, it is worth considering how they handle different preprocessing steps, such as feature scaling or normalization. Some models, such as k-NN, PCA and SVM models, are known to be sensitive to the scale of features and may require normalization to perform well [44]. However, tree-based models such as random forest and decision tree classifiers are not affected by the scale of features, making normalization unnecessary in many cases [45]. This explains why random forest was unaffected by the pre-processing method, while it did have an effect with k-NN and SVM methods which are known to be influenced by the scale of the data [14].

The reason why tree-based models are not affected by feature scaling is the way they are constructed. In a decision tree, nodes are split based on maximizing information gain or minimizing impurity. The threshold values used to split the data into equal portions at each node are determined by the information gain or impurity of the feature, rather than the actual feature values themselves. As a result, the decision making process is not influenced by the scale of the features. [36]

In addition according to Aurélien Géron [24], normalization may have a negative impact on the performance of tree-based models. When features are normalized, the differences between them may be reduced, which can make them less informative for the model. In some cases, normalization may even introduce artificial patterns or correlations that do not exist in the original data, leading to overfitting and poor generalization performance.

The model was further evaluated in co-operation with the professionals from the case study industry, and was deemed as a working solution. This evaluation was conducted using permutation feature importances [15] to extract the importances of the input features.

These features were then analyzed by the professionals and verified as being crucial in distinguishing between classes.

The current study demonstrates that the RF classifier with inverse weights and no data scaling, results in comparable, and in some instances better, performance to previous studies on material classification utilizing spectrometer analysis. However, it is important to note that the reference studies had minimal spectrometer data pre-processing, compared to the current case study where the data was pre-processed into chemical compound concentrations from the raw spectrometer data. Although the previous studies had very different datasets with varying levels of pre-processing, those studies that used the RF classifier achieved the highest accuracy. The findings of this case study indicate that the RF classifier is the most effective classifier in industrial material classification that utilize spectrometer analysis.

5.2 Future work

As previously discussed, the RF algorithm achieved a remarkable degree of accuracy and was verified by previously mentioned professionals in having learned from the correct features. However, since the data suffer of a major rare class issue, a significant proportion of the training samples from the minority classes had to be discarded to prevent overfitting, which could have made the model unstable and inaccurate. This could be improved by gathering more diverse data from the rare classes, and by experimenting with different data balancing methods and their combinations. Several data balancing methods that combine multiple techniques have been reported to produce better outcomes in comparison to using a single method alone [17] [39]. Some of these methods combine oversampling methods with undersampling methods, in order to better balance the data and address the rare class problem.

Another future improvement would be to include the classes that exhibit subclass problem, and experimenting with more advanced non-linear classifiers such as the Multi-layer Perceptron classifier (MLP) classifier and deep neural networks that can classify them efficiently. As of the current state these subclasses that are contained inside one another are not of interest. Advanced non-linear classifiers such as the MLP and deep neural networks are known to find complex patterns and structures in the data [13], which could help when dealing with classes than can have multiple subclasses.

6 CONCLUSION

In this thesis, a classifier model was constructed and trained for classifying industrial material from a case industry. The training data consists of 30,000 pre-processed spectrometer analysis from over 200 different material classes. The model was constructed from three parts, with the first part being data scaling to normalize and center the data, the second part being data balancing to address the rare class problem, and finally the third part being the classifier. Several methods for each data scaling, data balancing and classifier were tested in different combinations to find the combination with the highest classification accuracy.

The Random Forest (RF) classifier with unscaled data and the SMOTE data balancer was found to be the highest accuracy yielding combination. The found results correlate with the previous studies where random forest methods has had great classification success in different industrial fields. The trained classifier was further verified by industry professionals from the extracted feature importances the model had learned. Finally, several improvements were recognized such as improved data collection for addressing rare class problem, and MLP and deep learning classifiers for addressing subclass problem.

REFERENCES

[1] Guoli Ji, Pengchao Ye, Yijian Shi, Leiming Yuan, Xiaojing Chen, Mingshun Yuan, Dehua Zhu, Xi Chen, Xinyu Hu, and Jing Jiang. Laser-induced breakdown spectroscopy for rapid discrimination of heavy-metal-contaminated seafood tegillarca granosa. *Sensors*, 2017.

[2] Anna Michel, Alexandra Morrison, Victoria Preston, Charles Marx, Beckett Colson, and Helen White. Rapid identification of marine plastic debris via spectroscopic techniques and machine learning classifiers. *Environmental Science Technology*, 2020.

[3] M. Falkenström B.A.M. Hansson J. Gurell, A. Bengtson. Laser induced breakdown spectroscopy for fast elemental analysis and sorting of metallic scrap pieces using certified reference materials. *Spectrochimica Acta Part B: Atomic Spectroscopy*, 74–75, 2012.

[4] Roland Geyer, Jenna R. Jambeck, and Kara Lavender Law. Production, use, and fate of all plastics ever made. *Science Advances*, 3, 2017.

[5] Yinglin Yang, Xin Zhang, Jianwei Yin, and Xiangyang Yu. Rapid and nondestructive on-site classification method for consumer-grade plastics based on portable nir spectrometer and machine learning. *Journal of spectroscopy (Hindawi)*, 2020, 2020.

[6] Florian Gruber, Wulf Grählert, Philipp Wollmann, and Stefan Kaskel. Classification of black plastics waste using fluorescence imaging and machine learning. *Recycling*, 4, 2019.

[7] Yvette Mattley. Spectroscopy for plastics recycling. https://www.oceaninsight.com/blog/spectroscopy-for-plastics-recycling, 2021. 2023-02-28.

[8] World Steel Association. World steel in figures 2022. https://worldsteel.org/steel-topics/statistics/world-steel-in-figures-2022/, 2022. 2023-02-06.

[9] Ernst. Worrell and Markus A. Reuter. *Handbook of recycling: state-of-the-art for practitioners, analysts, and scientists*. Elsevier, 2013.

[10] Dillam Jossue Díaz-Romero, Simon Van den Eynde, Wouter Sterkens, Alexander Eckert, Isiah Zaplana, Toon Goedemé, and Jef Peeters. Real-time classification of aluminum metal scrap with laser-induced breakdown spectroscopy using deep and other machine learning approaches. *Spectrochimica Acta Part B: Atomic Spectroscopy*, 196, 2022.

[11] J. C. Martinez, J. R. Guzmán-Sepúlveda, G. R. Bolañoz Evia, T. Córdova, and R. Guzmán-Cabrera. Enhanced quality control in pharmaceutical applications by combining raman spectroscopy and machine learning techniques. *International Journal of Thermophysics*, 39, 2018.

[12] Xiangyu Deng, Shuhao Cao, and Abigail L. Horn. Emerging applications of machine learning in food safety. *Annual Review of Food Science and Technology*, 12, 2021.

[13] Zhi-Hua Zhou. *Machine Learning*. Tsinghua University Press, 2016.

[14] Ravi. Sanjay Churiwala Gopinath, Rebala. Ajay. *An Introduction to Machine Learning*. Springer Nature Switzerland AG, 2019.

[15] scikit-learn developers. scikit-learn. https://scikit-learn.org/stable/index.html, 2023. 2023-03-01.

[16] Gary M. Weiss. Mining with rarity: A unifying framework. *ACM SIGKDD Explorations Newsletter*, 2004.

[17] Alberto Fernández, Salvador García, Mikel Galar, Ronaldo Cristiano Prati, B. Krawczyk, and Francisco Herrera. *Learning from Imbalanced Data Sets*. Springer, 2018.

[18] Salvatore J. Stolfo Philip K. Chan. Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *International Conference on Knowledge Discovery and Data Mining*, 2001.

[19] B. Ravi Kiran, Dilip Mathew Thomas, and Ranjith Parakkal. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. *Journal of Imaging*, 2018.

[20] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie. Learning from noisy large-scale datasets with minimal supervision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2017.

[21] Gary King and Langche Zeng. Logistic regression in rare events data. *Political Analysis*, 9, 2001.

[22] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *The Journal of Artificial Intelligence Research*, 2002.

[23] Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2008.

[24] Géron Aurélien. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc, 2019.

[25] E. Fix and J.L. Hodges. An important contribution to nonparametric discriminant analysis and density estimation: Commentary on fix and hodges (1951). *International Statistical Review*, 1989.

[26] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 1967.

[27] M. Kuske, R. Rubio, A.C. Romain, J. Nicolas, and S. Marco. Fuzzy k-nn applied to moulds detection. *Sensors and Actuators B: Chemical*, 2005.

[28] Bo Sun and Haiyan Chen. A survey of k nearest neighbor algorithms for solving the class imbalanced problem. *Wireless Communications Mobile Computing (Online)*, 2021.

[29] Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20, 1995.

[30] Olivier Devos, Cyril Ruckebusch, Alexandra Durand, Ludovic Duponchel, and Jean-Pierre Huvenne. Support vector machines (svm) in near infrared (nir) spectroscopy: Focus on parameters optimization and model interpretation. *Chemometrics and Intelligent Laboratory Systems*, 2009.

[31] Larhmam. Svm margin. https://commons.wikimedia.org/wiki/File:SVM_margin.png, 2018. 2023-05-11.

[32] D. S. Broomhead and David Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2, 1988.

[33] Philip H. Swain and Hans Hauska. The decision tree classifier: Design and potential. *IEEE Transactions on Geoscience Electronics*, 15, 1977.

[34] Se. June. Hong. Jonathan. R.M. Hosking Don. Coppersmith. Partitioning nominal attributes in decision trees. *Data Mining and Knowledge Discovery*, 3, 1999.

[35] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27, 1948.

[36] Leo Breiman. Random forests. *Machine Learning*, 45, 2001.

[37] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, 1995.

[38] Venkata Jagannath. Random forest. "https://commons.wikimedia.org/wiki/File:Random_forest_diagram_complete.png", 2017. 2023-02-16.

[39] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 2009.

[40] Hervé Abdi and Lynne J. Williams. Principal component analysis. *WIREs Computational Statistics*, 2, 2010.

[41] Elizondo David. The linear separability problem: Some testing methods. *IEEE Transactions on Neural Networks*, 17, 2006.

[42] Bengio James, Bergstra. Yoshua. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13, 2012.

[43] Witten. Trevor Hastie. Robert Tibshirani Gareth, James. Daniela. *An Introduction to Statistical Learning with Applications in R*. Springer, 2021.

[44] Field. Cady. *The Data Science Handbook*. John Wiley Sons, Inc., 2017.

[45] Landgrebe S. Rasoul, Safavian. David. A survey of decision wee classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21, 1991.