TECHNISCHE UNIVERSITÄT WIEN

Master Thesis

Comparison of Ensemble-based Feature Selection Methods for Binary Classification of Imbalanced Data Sets

Author:

Erjola ZERALIU

Matr. Nr.: 1646629

Supervisors:

Univ. Prof. Dr.-Ing. Tanja ZSEBY

Dr.techn. Félix IGLESIAS

A thesis submitted in fulfilment of the requirements for the degree of Diplom-Ingenieurin

in the

Communication Networks Group
Institute of Telecommunications

Vienna, August 2021

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Acknowledgements

I would like to express my sincerest gratitude to my supervisor Dr.techn. Félix Iglesias, for his constant guidance, invaluable advices and patience. I could learn so much from his knowledge and expertise. I would also like to thank Univ. Prof. Dr.-Ing. Tanja Zseby, as my professor and my supervisor, for her very helpful feedback and recommendations. Furthermore, her fascinating lectures made this master program much more interesting. Special thanks go to Univ.Ass. Dipl.-Ing. Fares Meghdouri for his valuable suggestions.

A big thank you goes to my friends and colleagues who I studied with in Vienna, especially Ardit Ymeri and Navid Arshadi. It has been a pleasure to get to know and to work with such amazing and supportive people.

Last but not least, I would like to thank my family for the encouragement and continuously reminding me what is really important in life.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Kurzfassung

Die fortschreitende Digitalisierung in der heutigen Welt führt zu immer höheren Datenmengen. Dies macht eine schnelle Übertragung, robuste Verarbeitung und eine Reduzierung des Speicherbedarfs erforderlich. Dafür ist es wünschenswert, die Anzahl der Merkmale zu reduzieren, die bei der Datenverarbeitung als Eingabe verwendet werden. Dieser sogenannte Merkmalsselektionsprozess ist einer der wertvollsten Schritte der Datenvorverarbeitung. In der wissenschaftlichen Literatur wird der Merkmalsselektionsprozess jedoch oft nur unzureichend beschrieben und es besteht keine allgemeine Einigkeit über die optimale Technik zur Sicherstellung einer guten Klassifizierungsleistung und Stabilität.
Ziel dieser Arbeit ist es gut geeignete und breit anwendbaren Merkmalsauswahlmethoden für unausgewogene Datensätze zu finden. Wir untersuchen dazu drei verschiedene Ensemble-Modelle in Kombination mit der Stabilitäts-Selektion, einer Methode, von der wir erwarten, dass sie den Selektionsansatz deutlich verbessert. Später vergleichen wir die Ergebnisse mit der Merkmalsselektion durch neuronale Netze. Die Experimente werden auf elf verschiedenen Datensätzen aus unterschiedlichen Domänen durchgeführt, um etwas allgemeinere Schlussfolgerungen ziehen zu können. Zur Beurteilung der Klassifizierungsleistung mit den ausgewählten Feature-Sets, verwenden wir drei verschiedene Klassifikationsverfahren (Decision Tree, k-nearest neighbor und ein neuronales Netz) und werten deren Ergebnisse anhand von drei Metriken aus (F1-Score, average precision und ROC-AUC-Score).
Unsere Ergebnisse bestätigen, dass die Auswahl von Merkmalen nicht nur den Speicherbedarf senkt, sondern oft auch die Klassifizierung verbessert. Die Methode, die in unseren

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Experimenten die besten Feature-Subsets liefert, ist die Stabilitäts-Selektion mit Gradient Boosting. Der Ansatz des neuronalen Netzes hingegen zeigt die beste Fähigkeit, zwischen relevanten und irrelevanten Merkmalen zu unterscheiden.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Abstract

In the modern world more and more information is being digitized and shifted to the Internet sphere, resulting in a vast amount of data. Hence, quicker transmission, robust processing, and the reduction of storage requirements are needed, making it desirable to reduce the number of features that are used as input when processing data. This, so-called feature selection process, is one of the most valuable steps of data preprocessing. Nevertheless, many papers in the scientific literature portray the feature selection process only in an insufficient manner, and there is no general agreement on the optimum technique for assuring high predictive performance and stability.

In this work, we aim to find good and generally applicable feature selection methods for imbalanced data. We use ensemble models in combination with stability selection, a method that is expected to significantly improve the selection approach. Later, we compare the performance results with feature selection by neural networks. The experiments are performed on eleven different datasets from various domains. To assess the classification performance of the selected feature sets, we use three different classifiers (Decision Tree, k-nearest neighbor, and a neural network.) and evaluate three metrics (F1 score, average precision, and ROC-AUC score). We conclude our findings in terms of classification scores and discriminative power.

Our results confirm that selecting features does not only decrease storage requirements but often also improves classification scores. In our experiments, stability selection with gradient boosting delivers the set of features that achieves the highest classification scores. The neural network approach, on the other hand, shows the best ability to distinguish between relevant and irrelevant features.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Contents

Kurzfassung **v**

Abstract **vii**

Contents **ix**

1 Introduction **1**

1.1 Background 1

1.2 Motivation 3

1.3 Goals 3

1.4 Structure of the thesis 4

2 Background **7**

2.1 Introduction to feature selection and supervised machine learning . . . 7

2.1.1 Overview 7

2.2 Feature selection 8

2.3 Machine learning model 10

2.3.1 Baseline models 10

Logistic regression (LR) 10

Support vector machines (Support Vector Machine (SVM)) . . . 11

k-nearest neighbours (k-Nearest Neighbors (k-NN)) 14

Decision trees (Decision Tree (DT)) 16

2.3.2 Ensemble learning 19

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Bagging 19
Boosting 21
Stacking (STK) 23
2.3.3 Neural Networks Neural Network (NN) 24
2.3.4 Generalization and overfitting 29
2.4 Summary and contribution 30

3 Methodology and Experiments **33**
3.1 Data pre-processing 35
3.1.1 Datasets description 35
3.1.2 Encoding 37
3.1.3 Scaling 38
3.1.4 Train-Test splitting 39
3.2 Feature scoring 39
3.2.1 Feature scoring with ensemble methods 40
3.2.2 Feature scoring with Neural Networks 44
3.3 Feature subset selector 46
3.4 Validation stage 47
3.4.1 Evaluation Classifiers 47
3.4.2 Parameter tuning 50
3.4.3 Scoring metrics 51

4 Results and discussion **55**
4.1 Feature Scoring 57
4.2 Feature subset selecting 61
4.3 Evaluation scores 62
4.3.1 Half feature-subset evaluation 63
4.3.2 80% importance feature-subset evaluation 66
4.4 Performance degradation 69
4.5 Summary of results 70

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

5 Conclusion and further work **73**

5.1 Conclusions 73

5.2 Further work 74

6 Chapter Appendix **77**

6.1 Best parameters 77

6.2 Scoring metric values evaluated by k-NN 83

6.2.1 Evaluation in subsets of gradient boosting 83

6.2.2 Evaluation in subsets of random forest 84

6.2.3 Evaluation in subsets of stacking 85

6.2.4 Evaluation in subsets of neural networks 86

6.3 Scoring metrics values evaluated by NN 87

6.3.1 Evaluation in subsets of gradient boosting 87

6.3.2 Evaluation in subsets of random forest 88

6.3.3 Evaluation in subsets of stacking 89

6.3.4 Evaluation in subsets of neural networks 90

6.4 Scoring metrics values evaluated by DT 91

6.4.1 Evaluation in subsets of gradient boosting 91

6.4.2 Evaluation in subsets of random forest 92

6.4.3 Evaluation in subsets of stacking 93

6.4.4 Evaluation in subsets of neural networks 94

List of Figures **95**

List of Tables **97**

Bibliography **101**

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 1

Introduction

In this chapter, we discuss some general information behind this work, the motivation, targeted goals, and also the need for such research in the field of outlier/anomaly [1] detection. The last section of this chapter gives a brief overview of the structure of this thesis.

1.1 Background

When dealing with data analysis tasks, one of the first steps is the detection and examination of outliers, which is an imbalanced binary classification problem. Hawkins has defined an outlier as "*an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism*" [DSC20].

There are two main approaches for detecting outliers or anomalous points in static feature spaces:

[1] Note that the terms *outlier* and *anomaly* are used interchangeably in this work; however, they are not synonyms. For details on the differences between the two terms we recommend the reader to read in: https://medium.com/@alhadpofali_5697/anomaly-and-outlier-detection-concepts-1f82498851a2.

1. *Supervised methods*: here, we have historical, labeled data that defines, at least, *normality*. To evaluate the ability of the algorithm to detect anomalies, we compare the predicted labels to the original ones.

2. *Unsupervised methods*: here, we do not have labels for our data, and anomalies are purely established based on the intrinsic properties of the feature space under test.

Anomaly detection is a vital data mining problem in many real-life application domains such as fraud detection, network attack detection and disease condition detection [BG05, HBV+13, PM18].

In [IHZZ20] authors raise the question: *"Are network attacks in fact Outliers?"*. For their experiments, they used intrusion detection datasets and concluded that: *"network attacks have higher global distance-based outlierness averages than normal traffic"*, making outlier detection a promising tool for detecting unknown network attacks. Furthermore, they emphasize that the *"outlier nature"* of network attacks strongly depends on the chosen feature set. The identification of *anomalies* as *outliers* given a proper feature set is actually valid for many application domains since one of the main acceptations of the *anomaly* term is *being a minority class*. The process of selecting meaningful attributes is called *feature selection*. During this process, features that have a strong contribution to the detection (relevant features) are kept for subsequent analysis, whereas redundant or irrelevant features are removed. Authors in [GE03] list the benefits of feature selection:

- *"Improving the prediction performance of the predictors"*.
  This occurs because, if we do not use feature selection, the learning algorithms may be deceived by features that do not contribute with valid information, resulting in a noisy and complicated input dataset. This fact must be added to the intrinsic complications and increased sparsity of higher dimensional spaces (i.e., *curse of dimensionality*).

- *"Providing faster and more cost-effective predictors"*.
  Since subsequent analysis will have fewer data to process, the computational effort

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

will be reduced i.e. faster and less costly predictors.

- *"Providing a better understanding of the underlying process".*
  As the number of features in the model decreases, the complexity of the model decreases, resulting in a simplified model which is easier to understand and explain.

1.2 Motivation

The field of anomaly detection in network traffic is a paradigm of imbalanced binary classification. Therefore, insights into the data and features may lead to better detection systems for new threats, in addition to further benefits in other applications. This topic has been formerly discussed in [IZ14] where the authors studied and determined the influence of each feature on classification results.

With the enormous growth of the Internet, it is necessary to have robust and quick detection systems. For instance, in communication networks, network attacks evolve rapidly and network administrators are faced with undisclosed threats, which may be visible as anomalies, continuously. Feature selection is a very important pre-processing step that would assist in building such systems. Many researchers have studied selection techniques aiming to find the best approach, [KR18, VE15, TAL14]. However, there is still not a commonly accepted technique as a best practice for feature selection method, hence authors do not always coincide when selecting the best subsets of features. As a result, the main contribution of this thesis is disclosing the best feature selection method that is generally applicable over different imbalanced datasets.

1.3 Goals

In classification problems, the main element is the group of features that are used.
The target of our work is ranking and selecting the attributes with regard to their contribution to the detection of outliers/anomalies. We employ an adapted approach where we utilize a wrapper model, the permutation importance model [Bre01], that

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

takes as a core learner an example of each of the three ensemble algorithms, or a neural network. Aiming to achieve both: good predictive performance and also steadiness, i.e. *"producing similar features under the training data variation"* [KD19], we have employed the ensemble paradigm, *stability selection*: a very promising method for improving the robustness of the selection process. Later, we compare the performances of the nested models to assess features.

We specify four goals that we try to achieve:

- To study if ensembles in combination with stability selection are proper approaches for feature selection in imbalanced binary classification.
- To study the capabilities of neural networks as feature selectors.
- To explore permutation importance as a feature selection wrapper in imbalanced binary classification problems.
- To discover if there is an outstanding method or method-combination, or if it depends on the application/data.

1.4 Structure of the thesis

This thesis is divided into the following chapters:

- Chapter 1 describes briefly the motivation, goals and need for such work in the research community.
- Chapter 2 introduces the background knowledge needed to understand and perform the experiments. This chapter is focused especially on explaining basic machine learning algorithms, ensemble methods, and neural networks.
- Chapter 3 introduces the datasets and describes in detail the methodology and steps that are followed to conduct the experiments.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- In chapter 4 we present our results and compare the classification performance of several methods.
- Finally, chapter 5 summarizes the conclusions reached from our findings. Based on performance results on different datasets we generalize our outcomes and disclose the best method for feature selection.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

CHAPTER 2

Background

2.1 Introduction to feature selection and supervised machine learning

In this section, we introduce the essentials of feature selection, machine learning, and present popular classification algorithms.

2.1.1 Overview

The enormous growth of available data and the rise of computational power have promoted greatly the importance of Machine Learning (ML). Fundamentals about machine learning can be further consulted in [Alp10, Mit97, MRT12].

Any process starts with having a dataset that needs to be pre-processed before feeding it to the algorithm. Labeling the data, feature extraction, and feature selection are some of the pre-processing procedures. It is important to note that feature extraction is not the same as feature selection. During feature extraction process, new features are formed from the existing ones. Feature selection, on the other hand, consists of selecting the best features, but not creating new ones.

Focusing specifically on supervised machine learning, the next step is to train the

model. This stage entails selecting an algorithm and giving it the pre-processed data to determine the relationship between various features and a certain label.

After the model has been trained, we apply new sets of data that are never seen before by the algorithm. The model will then make predictions based on the data it has learned during the training process. The prediction may be a discrete label (classification) or a continuous quantity (regression).

Figure 2.1 depicts a broad framework that describes the supervised learning procedure.

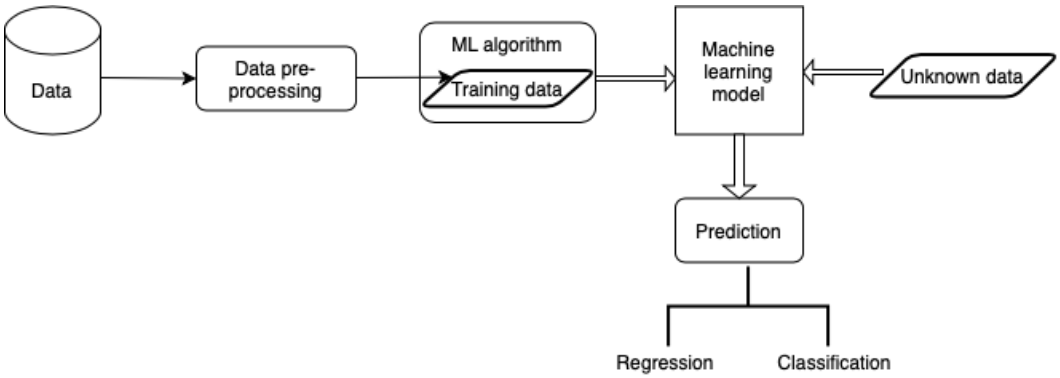


Figure 2.1: The basic scheme of the supervised machine learning approach.

2.2 Feature selection

A dataset may contain many redundant or irrelevant features. Note that redundant and irrelevant features differ from one another. Redundant features do not provide additional information, whereas irrelevant/noisy features do not provide any information at all. The process of eliminating features that are not relevant while maintaining only the ones that contribute the most to the prediction is called feature selection. This procedure not only cuts down on computational and time needs, but also improves performance. This occurs because the irrelevant features not only do not contribute to classification but also cause the results to be skewed. The following is how the authors in [GE03] define this concept: "The objective of variable selection is three-fold: improving the prediction performance of the predictors, providing faster and more cost-effective predictors, and providing a

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

better understanding of the underlying process that generated the data".

There are three commonly used feature selection methods:

1. *Filter method (learner independent).*

   This approach consists of selecting the “best subset” from the set of all features by using statistical methods (for instance, checking for the high correlation between features and the target output) [LS96, KJ97]. Later, the model is trained based on these features. The selection algorithm only relies on the characteristics of data to calculate feature importance. The lowest-ranked features are filtered out.

2. *Wrapper methods (learner dependent).*
   In this method, a subset of features is chosen from the entire set of features and utilized to train the model and assess its performance [KJ97, PLA05]. Then, a new subset of features is chosen and utilized to retrain the model. Finally, the subset of features that provides the best performance is chosen. This method has a high complexity for a large number of features $2^{(nr._of_features)}$.

3. *Embedded methods.*
   Embedded methods combine the qualities of filters and wrappers [LCWE06, SAVdP08]. These approaches consist of integrating the feature selection process as part of the learning algorithm. An example of embedded methods is *stability selection* [MB10]. The general idea behind stability selection is to generate bootstrap samples of the data and apply a core structure learning algorithm on each of the newly created subsets. The core algorithm will learn the data and choose the most important features on each sample. Later, the algorithm will tally how many times each feature has been chosen in these bootstrap samples, and based on the number of the selection times, it will be calculated a *stability score* which will define the final subset of features.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

In this work, we have used the embedded method for the feature selection process. Since some algorithms do not provide the necessary functionality to calculate feature scores, we utilize a wrapper model, the permutation model [Bre01], which wraps the core learner: random forest (RF), gradient boosting (GB), stacking (STK), or the neural network (NN). This technique allows determining the importance of a feature by calculating how much the absence of that feature would decrease the model score. The scores calculated by this model are called **permutation scores**. Later, we have incorporated also stability selection.

2.3 Machine learning model

After data has been processed, the next step is to build the model. This necessitates a learning algorithm and a set of training data. Some examples of works deepening into supervised models can be found in [Rie94], [STS16]. Some of the main families of supervised algorithms are described in the following. These algorithms can be used for both classification and regression, however, we will focus mostly on their use in classification problems.

2.3.1 Baseline models

Logistic regression (LR)

Logistic regression (LR) is a machine learning algorithm, firstly developed by David Cox in 1958 [Cox58]. This algorithm linearly establishes the probability of a sample belonging to a class. Later, by using a threshold, the probabilities are turned into a binary output, i.e. every number higher than the specified threshold will be assigned as *1* and every number lower than the threshold will be assigned as *0*.

Logistic regression is based on the logistic regression function. This function is an S-shaped curve, illustrated in 2.2, and converts input values to probabilities. The logistic

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

function is called also the *Sigmoid function* and is defined by the formula:

$$S(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

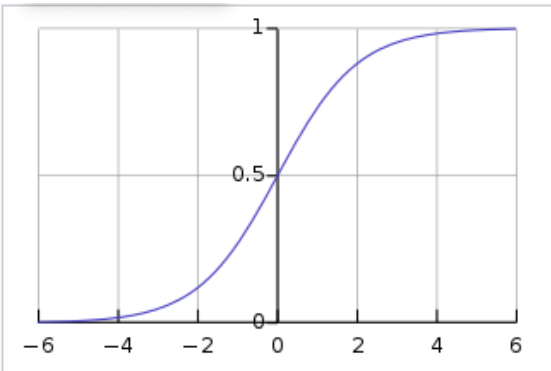


Figure 2.2: The logistic function (Source: *wikipedia.org/wiki/Logistic_regression*).

The LR classifier has several parameters which can be adjusted automatically using parameter tuning. A very important parameter is the *regularization parameter* which specifies the norm used in the penalization for fitting the curve to the data. It is called also the *penalty parameter* because it adds bias to the estimate in order to improve the generalization performance. In this way, the prediction accuracy in the already known data will not grow, but the one in the test set will; as a result, it will prevent overfitting. By default, this algorithm considers an L2 regularization, i.e. Ridge Regression. The choice for the value of the penalization parameter should be a trade-off between the overfitting possibility and how much bias is one willing to add to the estimation. Examples of LR for supervised classification and feature selection can be seen in [DHG20], [CVA06], [CO06].

Support vector machines (SVM)

Another machine learning algorithm is the Support vector machine (SVM) invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963 [CV95]. This model utilizes a technique called *"the widest street approach"* [CV95], where *"the street"* will be between

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

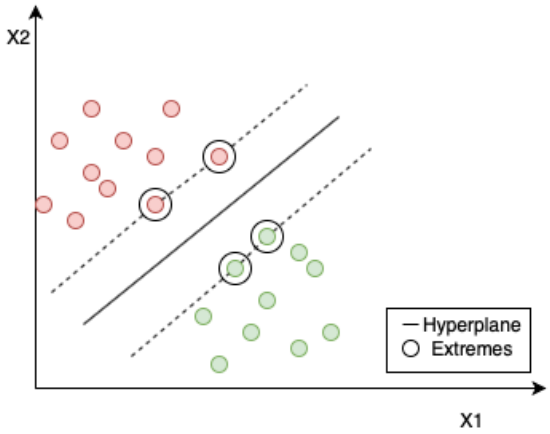


Figure 2.3: An example of SVM data separation in a 2D plane.

two different classes. SVM looks at the extreme points, i.e. the points that are closest to the other class, and draws the decision boundary, the separation between classes. Data samples are referred to as vectors because they can have many dimensions and the separation is referred to as a hyperplane.

In order to decide how to make the decision rule that will use this decision boundary, we have to consider two different scenarios:

1. Linear separable cases:

   Let's suppose that we have two different classes in a 2D plane as it is shown in figure 2.3. There are infinite lines that can be used to separate them, but SVM should choose the most optimal one: the one that optimizes the margin, the widest street.

   To accomplish so, firstly, we find the extremes of each class and create two parallel hyperplanes where each of them meets at least one of the extremes. In figure 2.3, the parallel hyperplanes are shown with dashed lines while the extremes are are surrounded by circles. Then, we adjust the rotation of the hyperplanes in order to maximize the distance between them. The last step is to create the main hyperplane that is parallel to the two other hyperplanes, and will lay in the middle of them.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

2. Non-linear cases:

   Let's consider the scenario shown in figure 2.4.

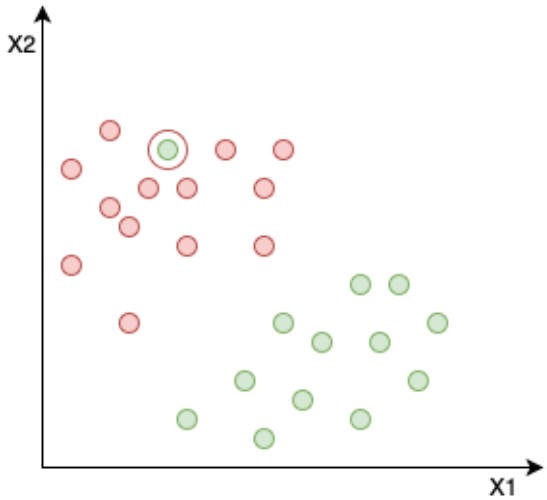


Figure 2.4: An example of SVM data separation for a non-linear case.

In this example, one of the red dots is in the green dots cluster, causing the data to become non-separable linearly. This problem can be solved by introducing two concepts:

a) Soft margin

The term *"soft"* refers to how it tolerates a few misclassified samples. Also it seeks to find a balance between maximizing the margin while minimizing the number of incorrectly classified samples.

b) Kernel Trick

"Kernel trick" refers to the method of using a linear classifier to solve a non-linear problem. In order to be able to linearly separate different classes, we do not work in the original space, but we use a transformation function to go to a higher-dimensional space. An example of this method is shown in figure 2.5 After the transformation, we are able to just draw a line to separate red dots from green dots.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

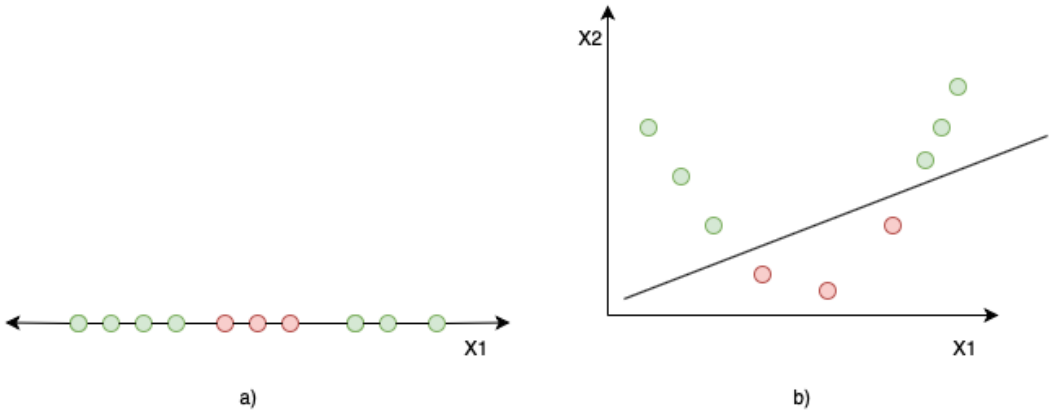


Figure 2.5: Transformation by polynomial kernel.

SVM can be applied in a wide range of domains, for instance: weather prediction, stock prediction, defect classification, handwriting identification, image and audio processing and medical diagnosis. Some works that studied the application of SVM in specific areas can be seen in [FM04, STA10, ZZL+17].

k-nearest neighbours (k-NN)

This is also a machine learning algorithm, first introduced by Evelyn Fix and Joseph Hodges in 1951 [FH89]. Compared to other popular machine learning algorithms that are considered as *eager*, k-NN is described as a *lazy* learner, meaning that it does not build a model, but performs all the computations during classification times. Therefore, the training phase consists only of storing all the known sample vectors together with their labels. During the classification process, for every unclassified vector, the algorithm will calculate the distances of this new sample from the already known ones. Later, these distances and the corresponding sample label will be sorted in ascending order. From this collection, only k smallest values will be chosen (k-nearest neighbors). Each of these values correspond to a class label and will be used as *votes*. Later, the label that has more votes, will be assigned as the label of the new sample.

An example illustrating this case is given in figure 2.6.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

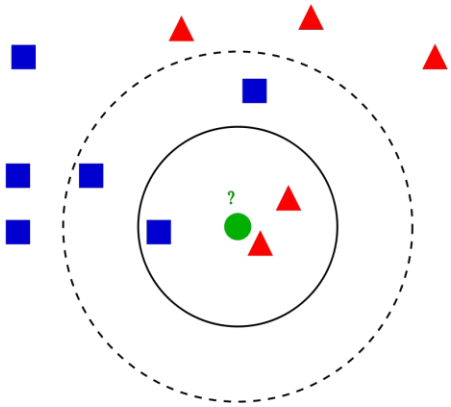


Figure 2.6: An example of k-NN classification process (Source: *wikipedia.org/wiki/K − nearest_neighbors_algorithm*).

In this case, the green circle is the test sample that we need to classify. If we choose to consider 3 neighbors, 2 train samples will vote *red triangle* while only one sample will vote *blue square* Hence, the test sample will be classified as a red triangle. However, the decision will differ if we use five neighbors (k=5). Now, 3 samples will vote *blue square* and only 2 samples will vote *red triangle*. In this case, the unknown instance will be classified as a blue square. In k-NN, the number of neighbors is a very important parameter and before using this algorithm one should be careful about the following choices:

1. k=1: means that we will predict the label based only on the one closest neighbor. As a result, we will not have stable output.

2. The number of neighbors (k) is very high: this is also not a good choice because it may happen that one of the clusters has fewer data points than the others. This means that when doing the majority voting process, the cluster with the highest number of data points will have more weight, even if it is not the correct one.

3. In problems where we use the majority voting (e.g. classification cases) we choose k to be an odd number. In this way, we avoid the problem of having equal votes from different classes.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

In our work, the number of neighbors is chosen by a random search to find the best possible value for the specific dataset.

The most important properties of k-NN are listed as follows:

- Easy to understand and interpret.
- Easy to implement.
- No training time needed.

On the other hand, k-NN also has the following disadvantages:

- Memory requirements depend on the training data.
- It becomes computationally expensive when there are many items to be classified.

Examples of k-NN for supervised classification and feature selection can be seen in [BCS15, MD03, TBK07].

Decision trees (DT)

Decision Tree (DT), invented by Charles J. Stone in 1984 [BFOS84], is one of the most popular machine learning algorithm, especially for classification problems. As the name implies, it is a tree-like graph structure: it models decisions and then outputs the outcomes of these decisions. The tree is built iteratively using the labelled samples in the training set, starting from the root to the inner nodes and then down to the leaves. The root node contains the feature that gives the lowest impurity, i.e., the feature that distinguishes samples of various classes more precisely. Next, based on certain features and their values, the inner nodes make the decision of whether to proceed left or right. Then, the leaf nodes in the tree define the outcome, i.e., the predicted class. There are two stages to using a DT:

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

1. Training phase.

   During this step, we find rules that recursively split data into subspaces. For the classification case, we first check for all the samples that we have, whether or not all the instances are from the same class. For every leaf node we have:

   1.1. If not all data is from the same class:

      1.1.1. For every feature, identify potential splits that will separate the data into subspaces.

      1.1.2. Compute the best split over all features. Note that *best* depends on a split goodness measure/criterion.

   1.2. Continue the process until data in all leaf nodes is from the same class.

2. Classification phase.

   Here we traverse the tree from the root until we arrive at the leaf node. At the leaf node, we are going to decide the outcome of the prediction.

The example in figure 2.7 shows 2-dimensional data (x, y) belonging to two different classes: star and circle. Our purpose is to find the rules that split the data into subsets such that each of the subspaces contain data from a single class. To build the tree we firstly start with a decision on the x-axis, the most suitable split is the line x=5. Now we have created two subspaces: the set of samples that have the coordinate x>5 and the other set that corresponds to x<5. Then we traverse down both of the branches and look for the potential splits in the y-axis. We decide y=2 as the next split and we notice that 3 of the subspaces contain samples from one single class, but there is still one subset that contains both circles and stars. We decide the next split be y=1, and now we achieve the purpose to have only instances with the same label in every subspace. In the right-hand figure, we have drawn the corresponding tree that illustrates the iterations.

There are many measures to compute the best split, for instance:

- Error rate - the ratio of incorrectly identified samples to the total number of samples. In other words, error rate simply counts the number of classification errors when

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Figure 2.7: An example of building a DT for classification of two shapes, circle or star.

performing a split. It is made up of the amount of false negative and false positive. This is the simplest but probably not the most effective measure to choose.

- Information gain - a better measurement than the error rate.

  To calculate the information gain for a split we subtract the weighted entropies of each subset from the original entropy. Mathematically, this can be expressed by the formula:

$$IG(X_A, X_B) = H(X) - p(x_A)H(X_A) - p(x_B)H(X_B) \tag{2.2}$$

  Where: IG(...) - information gain, p(...) - probability function, H(...) - Entropy

  Since information entropy is a measure of uncertainty, the higher the probability of an event to happen is, the lower is the entropy. A high entropy needs to be reduced, so we need to find a split that reduces entropy the most.

  The value range of this parameter is between 0 and 1.

- Gini impurity (gini index) - also a more effective measurement than error rate. This measure is the most used criteria in the implementation, which is also the default setting in the *sklearn* library. The value range is also between 0 (only one class) and 1 (total inequality). The computing process is the same process as above: the candidate split is computed, and then each split is evaluated with gini impurity instead of information gain. Finally, the split with the best value is selected.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Examples of DT for supervised classification and feature selection can be seen in [SR08, POA15, GIW09].

2.3.2 Ensemble learning

Up until this point, we have covered the fundamental machine learning algorithms, often known as *weak learners*. These algorithms can be combined to build a stronger learner that is more accurate and resilient than any of the weak learners individually. This technique is known as *ensemble learning* [Die00] and the three main kinds of these algorithms are: bagging, boosting, and stacking. When building a model, we opt for an algorithm with low bias and variance. With ensemble models, we can choose the combining method such that it reduces the problematic parameter of the base learners. For instance, if we select base models with low bias but large variance, we should combine them using an approach that reduces variance (e.g. bagging). Similarly, we should employ a combining strategy that tends to reduce bias if we choose base models with low variance but high bias (e.g. boosting or stacking). Hence, ensembles can overcome the performance of individual algorithms by lowering variation and bias.

Some overviews and comprehensive works extending on ensembles and ensemble learning can be consulted in [BVS03].

Bagging

This method employs bootstrapping, which uses random sampling with replacement, to build multiple subsets of training data that will be utilized as independent datasets [Bre94, Zho12, SMT09] . On each of these subsets, we fit homogeneous (same type) weak learners which will make their own predictions. The final decision is then calculated as the prediction that has the majority of votes.

A schematic representation of this method is shown in figure 2.8.

Advantages of bagging:

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- Reduces variance by averaging the outputs (preventing so overfitting).
- The base learners are generated independently of each other - parallel learning.

Disadvantages of bagging:

- relies on instability - we should have diversity in the training subsets in order to be provided with different models.

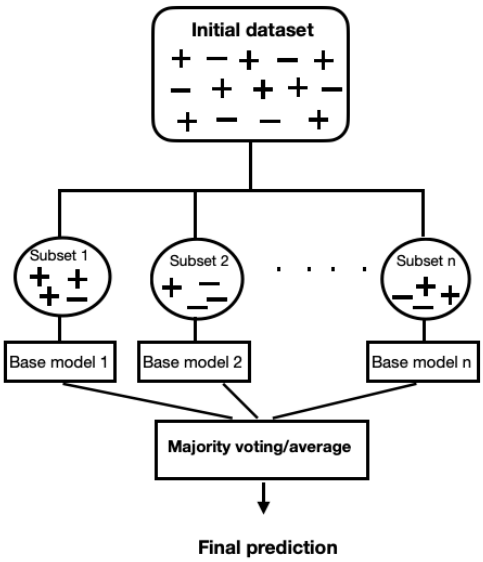


Figure 2.8: Illustration of the bagging technique.

One example of bagging is Random forest (RF). This is one of the most popular and most powerful supervised algorithms in machine learning. As the name implies, this algorithm creates a forest with many DT. In general, the higher the number of trees in the forest, the more reliable the prediction, and thus, the forest will provide greater accuracy. This classifier selects random subsets of features and does not provide each tree its full set of features. This behaviour improves the variance-bias trade-off, resulting in increased accuracy.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Boosting

Boosting is another ensemble learning technique that similarly to bagging, combines multiple models to generate a more powerful learner [Sch90, Bre98, Zho12]. Differently from bagging in this approach the underlying algorithms will learn in an adaptive manner, and if any samples are miss-classified, these samples will be assigned a larger weight.

We start with a dataset with weighted samples; initially, each sample has the same weight. The first weak learner will be trained and predict on this dataset.
Next, the miss-classified will be assigned a larger weight and the weights will be updated.
The successive model will be trained and make the predictions on the updated dataset.
This process will continue until we get the predictions of the very last weak learner.
At the end of the process, we obtain a strong learner with reduced bias.

This algorithm is illustrated in figure 2.9.

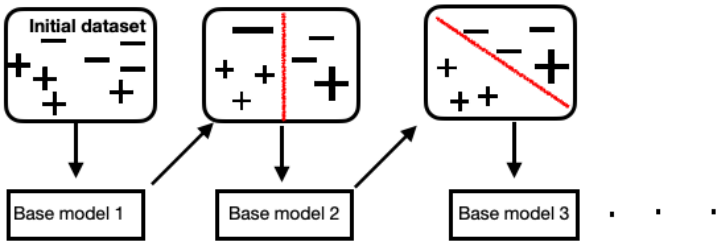


Figure 2.9: Illustration of the boosting technique.

As seen in the figure, all of the observations in the initial dataset have the same weight. The first algorithm will be trained on this dataset and the classification performance will be checked. In the next iteration, the wrongly classified samples will be given a higher weight in order to concentrate the focus on the most difficult examples. After the weights of the samples in the training set have been modified, the next weak model will be trained on the updated dataset. This sequential learning will continue in the same way for every iteration until we achieve our final model.

Advantages of boosting:

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- Decreased bias.
- Produces more diverse classifiers than bagging because of the added weights.
- More hyperparameters to control model building than in RF: learning rate - controls how you add subsequent trees together, and regularization in form of lasso and ridge parameters.

Disadvantages of boosting:

- Computationally more expensive (sequential learning).
- For added classification noise, boosting (adaboost specifically) will give worse results because it will put a lot of weight on the noisy examples by not "giving up" on them even if they keep repeating to be misclassified. A better method would be one that "gives up" on these repeatedly misclassified examples after some iterations.
- Less interpretability (each successive tree uses the residual of the previous tree, each subsequent tree slowly reduces the overall error of the previous tree => high predicting power but low interpretability).

One example of boosting is *Gradient Boosting*. This algorithm follows several sequential steps to fit a model in a given set of data. Let's suppose that we have a classification scenario and we are provided with a dataset with different features. The steps that GB follows are summarized in the list:

1. We start with a single leaf that contains the initial prediction calculated as the log of the odds:
$$log(odds) = log(positive_label/negative_label) \tag{2.3}$$
2. Later, this log of the odds is converted into probability by using a logistic function. The resulting probability is going to be put at the leaf that represents the initial prediction for every sample.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

3. Once this initial prediction is known, it can be measured how bad the initial prediction was by calculating the residuals between each observation and the predictions:

$$Residual = Observed - Predicted \tag{2.4}$$

4. Next step is building a tree using some of the features of the dataset and then calculating the output values of the leafs.

5. Since the predictions are in terms of log(odds), it is needed to use some transformations first to calculate the leaf's outputs. The most common one is:

$$\frac{\sum Residuals_i}{\sum[Previous_probability_i * (1 - Previous_probability_i)]} \tag{2.5}$$

6. Next, the initial leaf is combined with the just built tree.

7. At this point, the log of odds that we had in the first step is summed with the scaled predicted values of the leaves in the new tree and converted into the corresponding probability value. Now one can calculate the new residuals by subtracting these probabilities from the observed values.

The steps above are repeated to build the next tree. This process will continue until the maximum number of trees specified is reached, or until the additional trees fail to improve the fit (the residuals get very small).

Stacking (STK)

Unlike bagging and boosting, STK considers different learning algorithms (heterogeneous weak learners) [Wol92, Bre96, Zho12]. After selecting the weak learners, we should also choose another algorithm, the meta-model, that will make the final prediction based on the outputs from the base models. The dataset will be separated into two folds - the first fold will be used to fit the weak learners, and in the second fold we will make the predictions from the created base models. The outputs of these base models will be used as inputs for the meta-model to make the final prediction. Diagram 2.10 illustrates the construction of the STK classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

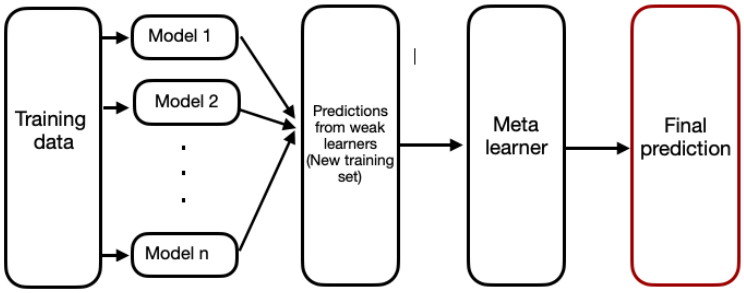


Figure 2.10: Illustration of the stacking technique.

Advantages of STK:

- The purpose of using this classifier is to improve the prediction performance. STK is expected to achieve performance at least as good as its best weak learner.

Disadvantages of STK:

- It uses fewer data to train the weak learners and the meta-learner (only half of the training data for the weak learners and half for the meta-learner). To overcome this limitation we can use k-fold cross-validation.
- There are no criteria that decides which model will contribute the most to the ensemble prediction, all the models contribute the same amount. This may cause the performance of some highly accurate algorithms to be drawn from some algorithms with low accuracy.

2.3.3 Neural Networks NN

Initial motivation

As the name implies, artificial neural networks are inspired by the biological neural system.

A biological neuron receives the information, processes it, and then transfers the result to the other neuron. This functioning way of a neuron can also be translated into an

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

artificial model. In neural networks, we have dendrites to receive the signal - some antenna-like structures which are strengthened or weakened depending on how often they are used. The strength of each connection will determine how much is the contribution of the input to the output. Figure 2.11 shows how a neuron in a human brain works.

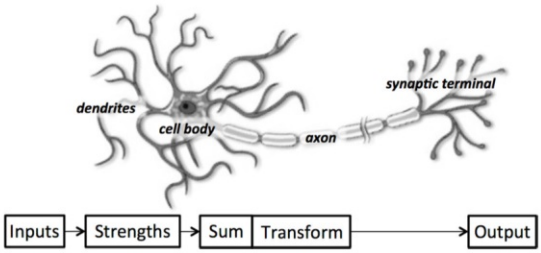


Figure 2.11: "A functional description of a biological neuron's structure" - Buduma and Locascio 2017 [BL17].

In an artificial neural network, this strength is determined by the weights that decide how much influence the input will have on the output.

Single-Layer Perceptrons

The first artificial neural network, called perceptron, was invented in the early 1950s by Frank Rosenblatt [Kan03]. It consisted of a single neuron and it is the basic building block of today's neural networks. A schematic representation of a one-neuron network is given in figure 2.12.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

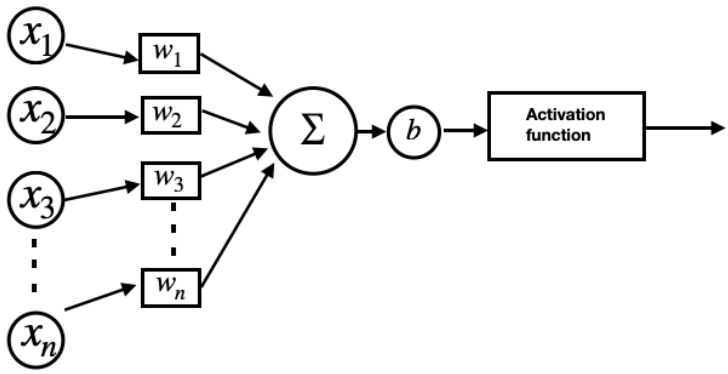


Figure 2.12: The schematic representation of an artificial one-neuron network.

In this structure, each input x_i has an associated weight w_i which will scale the input value. All weighted inputs will be summed up and after adding also the bias b, this sum is passed through a nonlinear activation function with a specific threshold to achieve the output y.

The training process of these structures consists of learning the weights from labeled samples that are available. The steps of the learning process are listed below:

- initialise weights,
- predict sample labels: y',
- compare if $y' = y$,
- if not equal, compute and assign new weights to the connections: $w' = w + a(y - y')x$, where a is the learning rate parameter and determines the magnitude of weight updates.
- continue the iteration until $y' = y$ for every sample.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

In this structure, the step function is used as an activation function, hence, the output will be defined from the system:

$$Y = \begin{cases} 1, & \text{if } \sum_{i=0}^{n} x_i w_i + b > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.6}$$

If the data is not linearly separable, the training process will never stop. In such a case, we should change stopping criteria - for instance, we can set as a stopping criterion the maximum number of iterations.

Neural Network

The perceptron that we explained above is the building block of neural network (NN) architectures. These networks are simply the aggregation of artificial neurons (nodes) working together towards the same goal and achieving the smallest possible error. These structures are also called *a black box* because it is hard to understand what is happening inside. A graph that illustrates a NN structure is shown in figure 2.13. The network in this graph is called a *fully connected network* because each node is connected to every other node of adjacent layers.

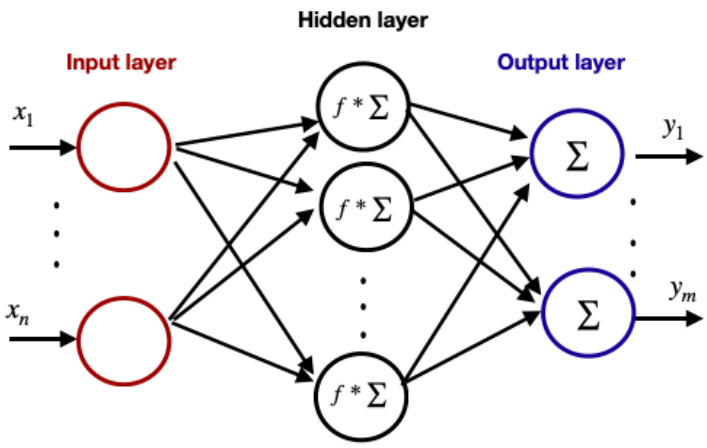


Figure 2.13: Basic neural network structure.

Despite the application purpose, a Neural network is constructed out of 3 different types

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

of layers:

1. Input layers.

   The nodes in this layer are needed to get the feature values for every sample. Usually, the number of nodes in this layer equals the number of features of the dataset. There is no computation performed in the input layer - it is needed just to duplicate feature values and to transfer them to the nodes in hidden layers unchanged. The shape of the input layer is defined by the given inputs $x_i \in \mathbb{R}^n$.

2. Hidden layers.

   In this layer, the transformation of the values and the processing takes place. With the increase of the number of hidden layers, the complexity of the network will also increase. The connections between nodes are called *edges*. These edges provide the output of a neuron as an input of another neuron. Each of these connections has its own weight, $w_i j \in \mathbb{R}$; where i and j specify the end and starting neuron respectively. The value of the weight influences the incoming signal by either decreasing it (w < 0) or amplifying it (w > 0). The weighted sum of the inputs, sometimes also adding a bias) will make the input of the neuron in the next layer.

3. Output layers.

   Neurons of the output layer will take as inputs the outputs from the hidden layer and will compute the final prediction. The shape of this layer is defined by the corresponding outputs $x_i \in \mathbb{R}^m$ The number of nodes in this layer depends on the kind of problem. In binary classification tasks, we use only 1 neuron in the output layer.

The training process in neural networks is carried out as an optimization task using the stochastic gradient descent optimization algorithm. The steps are the same as in a single neuron structure. Firstly, random weights are assigned to each connection, then we repeat the same process until it satisfies the condition:

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

1. pick a training example,
2. compute the outcome,
3. compute the gradient of the loss with respect to the outcome
4. backpropagate loss, compute gradients with respect to weights
5. update weights in direction of the negative gradient (to reduce loss).

A crucial component of neural networks is the **activation function** which is present in the hidden layers and output layers. This function is very important because it defines the output of a node given a set of inputs. Note that due to the fact that values in the output layer have to be as similar as possible to the expected output data, the activation functions in the output layer are usually different from the ones in the hidden layers. There are several parameters to be chosen when designing and implementing a neural network: number of neurons in each layer, optimization algorithm, loss function, back-propagation method, etc. The parameters that we have decided to use in this work are explained in detail in section 3.

2.3.4 Generalization and overfitting

In machine learning, there are two important and closely related terms: *generalization* and *overfitting*. These topics are a potential problem in the case of supervised learning, where the algorithm is given a set of samples and their outcomes. The model analyzes the relationships between the samples and labels and makes predictions based on the function concluded from this training set. These predictions are compared to the known labels, and the parameters of the model are adapted until these outcomes align. The bigger the training set, the better the algorithm can learn the data. The purpose of the training dataset is to develop the algorithm ability to generalize the concluded function for the given data and to accurately predict the new data. It is possible, however, that the model learns the training data in too much detail and is unable to generalize; this is referred to as *overfitting*. The inverse is known as *underfitting* and it happens when

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

the algorithm is not sufficiently trained and does not learn the training data well. The overfitted models usually have high variance and low bias, while the underfitted models have high bias and low variance. Both of these phenomenons make the model useless because the algorithm is incapable of making accurate predictions based on unseen data. To avoid them, a proper balance between variance and bias must be found. When it comes to obtaining low variance and low bias, ensemble methods are a good choice. This is due to the combining procedure, which tends to minimize the high term of the weak learner.

2.4 Summary and contribution

In most machine learning problems, we endeavor to select the best features that are the most relevant features to the prediction. In the realm of anomaly detection, several studies have examined the feature selection process and proposed baseline feature subsets. In the domain of network traffic analysis, the authors in [IZ14] have investigated and assessed the impact of each feature on classification results in the domain of network traffic analysis, demonstrating that they may cut the number of features from 41 to 16 without affecting performance significantly. Similarly to our work, in this paper it is embedded also stability selection following the technique presented in [MB10]. Later, in [FVV+17] a meta-study was conducted to identify and rank the most used features related to anomaly detection and traffic classification.

Despite the several previous publications on feature selection [KR18, VE15, TAL14], there is currently no widely acknowledged technique for feature selection that is the optimum approach. As a result, this thesis major contribution is to investigate several feature selection approaches in order to find good strategies for a given task. Authors in [AKD+12] and [Pes17] empathise that one of the limitations of present feature selection approaches is their lack of stability. This means that the predictions are too sensitive to changes in the set of training data and variations of the input data would produce a different subset of features, resulting in limited reproducibility of outcomes. Hence,

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

ensemble paradigms are increasingly being studied as a possible framework for enhancing the stability and robustness of the feature selection process with the goal of achieving a better trade-off between stability and predictive performance [SAVdP08, YM11]. Not only have these approaches been demonstrated to offer stability, but they have also been found to increase prediction performance. Authors in [DLR+18] explored some of the ensemble methods, specifically bagging and boosting, as feature selectors and demonstrated that these methods showed improved accuracy compared to a single feature selection technique. This accuracy was measured with respect to classification error metrics. [Pes19, SAVdP08] also explored homogeneous ensembles in terms of stability and classification performance. The idea of these approaches is generating bootstrap sample sets using sampling with replacement and performing a separate feature ranking on each of these sample sets. Later, the rankings from the algorithm in each bootstrap sample were aggregated using voting.

The goal of these studies is to compare ensembles capabilities to single algorithms in the feature selection process. Our research, on the other hand, will compare the performance of different ensemble methods to each other. Furthermore, we will extend the comparison to heterogeneous approaches where we employ diverse classifiers. The examples of ensembles that we have used are random forest (RF), gradient boosting (GB) to represent the homogeneous ensembles and stacking (STK) which represents the heterogeneous methods. Bagging and boosting are built from decision trees. Stacking (STK), on the other hand, uses decision trees (DT), support vector machines (SVM) and logistic regression (LR) as a weak learner, and logistic regression (LR) as a combining algorithm. Later, we utilize also a neural network to analyse the capabilities of this algorithm as a feature selector. In contrast to the previous techniques, we analyse these algorithms not only in terms of stability and classification performance, but also in terms of discrimination power. In order to get improved results, we additionally embed stability selection.

To evaluate the performances of each approach, we have used three different algorithms: k-NN, NN, and DT. These algorithms classify the samples by building

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

a model capable of predicting the class of instances of a dataset using its features. As explained above, each of these algorithms has specific parameters which play an important role in the prediction that these classifiers make. In order to get the best possible performance, we have chosen the optimal parameters using hyperparameter tuning explained in 3.4.2.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 3

Methodology and Experiments

In this chapter, we introduce the analysis framework structure and explain how the experiments were conducted. The purpose of these experiments is to compare alternative feature selection algorithms and determine which strategy performs best in imbalanced datasets.

In Figure 3.1 we show the experiments scheme.

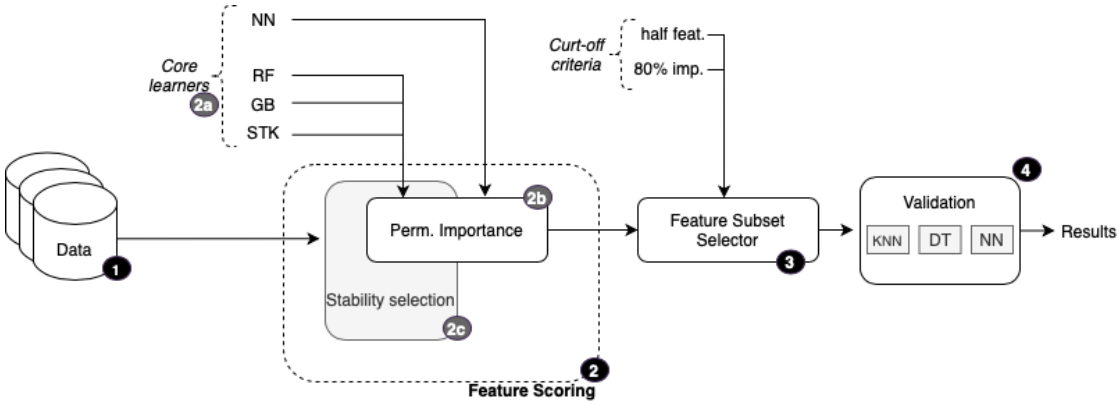


Figure 3.1: Framework of the conducted experiments.

In what follows, we provide a quick overview of each block and then give further details in the next subsections.

1. Preparing the datasets.

   Before feeding the data into the estimator, we should first process it while taking into account the algorithms that will be used later. This step consists of encoding, scaling, and train/test splitting.

2. Feature scoring.

   This stage is considered the core of this work and includes the following steps:

   2a) Select a base estimator.

   The estimator can be any of the ensemble methods: RF, GB, STK, or a NN.

   2b) Wrap the base estimator into a permutation model.

   This model will be used to calculate the permutation scores for every feature. In the case of the NN, these scores will be used directly to determine the number of features that will be selected. In the case of ensemble models, we proceed to the next step.

   2c) Apply stability selection. [MB10].

   This technique belongs to the embedded feature selection family and is based on sub-sampling in combination with a core algorithm. The idea is to generate subsets with half of the total samples and then, on each of these subsets we apply the scoring algorithm, which in our case is the permutation model. This model will assign a permutation importance score for every feature and features with a score greater than a given threshold, will be selected. Later, the selected features are compared across all subsets and based on the number of times that they are chosen, a *stability score* is calculated. The calculation of these scores is explained in 3.2.1.5. Stability selection step is not suitable to be followed for the NN approach because this algorithm does not perform very good on datasets with a small number of samples. In our work, we use also such small datasets that after bootstrapping would result in even smaller subsets with an insufficient number of samples for NN to be properly trained.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

The reason for using a permutation importance wrapper is because some algorithms do not provide the feature importance out of the box. Hence, we wrap our base model into a *PermutationImportance* model that instead offers this attribute.

3. Feature selector subset.

   Next, we choose the number of features to be selected based on the scores that are assessed after the stability selection process. We delete the undesirable features, yielding new datasets that are subsets of the original data.

4. Validation.

   The last step is to evaluate the performance in each newly formed subset. We use three common classifiers for this purpose: k-NN, DT, and NN. These algorithms are fine-tuned to find the parameters that optimize a given metric, which in our case is F1. Furthermore, we provide some statistical analysis to compare the effect of different nested models carrying out the feature selection process while embedding stability selection; this step is described in chapter 4.

3.1 Data pre-processing

Machine learning consists of building models that learn and extract meaningful insights from data. This means that in order to create a model with high accuracy, the data we feed to a machine learning algorithm should be clean and organized. To create such a dataset, we must first go through a series of steps before feeding the data to the algorithm.

3.1.1 Datasets description

In this work, we use outlier/anomaly detection datasets and intrusion detection datasets (Intrusion Detection System (IDS)). The anomaly detection datasets that we use are:

- Outlier Detection Datasets (Outlier Detection Dataset (ODDS)) [Ray16]

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- Datasets published at the Data Mining (Data Mining (DaMi)) journal [CZS+16]. In the next sections, these datasets will be referred to as *DAMI* datasets.

These databases contain pre-processed data from a variety of domains and are available in different formats depending on the pre-processing method used. We used the normalized and duplicate-free alternative. The datasets and the corresponding number of outliers and features are listed in the table below:

Dataset	**Samples**	**Outliers**	**Features**
Annthyroid	7200	534	21
Cardio	1831	176	21
Mnist	7603	700	100
Pendigits	6870	156	16
PageBlocks	5473	560	10
Satellite	6435	2036	36
Satimage	5803	71	36
SpamBase	5473	560	57
Vowels	1456	50	12
Letter Recognition	1600	100	32

Table 3.1: List of outlier detection datasets that are used in the experiments and their corresponding details [Ray16, CZS+16].

In addition to ODDS and DAMI datasets, we use also three intrusion detection datasets intended for network security and intrusion detection purposes. These datasets include:

- The CICIDS2017 dataset by Sharafaldin et al. [SLG18]. This set includes data collected over a five-day period. To generate normal traffic, there are utilized realistic user profiles. Later, attacks are triggered in a controlled environment and consist of seven separate attack families.

- The UNSW-NB15 dataset by Mustafa et al. [MS15]. The raw network packets were generated using the IXIA PerfectStorm tool. This collection contains 100GB of traffic captures (pcaps). There are nine families that represent the attack traffic.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- The ISCX-Bot-2014 dataset by Beigi et al. [BBHJSG14]. Different from the previous collections, this dataset is focused on botnets. It includes 16 types of botnets combined with normal traffic generated with real devices and mimics user's behavior.

For all three datasets, we have labelled the attacks as anomalies. Some statistics of these datasets are given in the following table:

Dataset	Samples	Anomalies	Features
CIC2017	2853820	612394	41
UNSW-NB15	3550574	60972	41
Botnet14	556806	244702	41

Table 3.2: List of IDS datasets used in the experiments and their corresponding details[SLG18, MS15, BBHJSG14].

3.1.2 Encoding

There are several types of data, such as numerical data, binary data, categorical data, and so on. To represent the features at the input, most machine learning algorithms require numerical values. As a consequence, categorical features would either cause errors or be ignored. This is not always the best solution since such features can contain valuable information that would eventually contribute to a more accurate prediction. Some algorithms, such as DT and its ensembles: RF, GB, and others, can learn categorical data. However, some models, such as SVM (an algorithm we used in our experiments), are unable to work with nominal features.

The anomaly detection datasets (ODDS and DAMI datasets) contain only numerical values, hence, they do not need encoding. On the other hand, IDS datasets contain also categorical data, as a result, we use *one-hot encoding* to substitute categorical features with a set of binary features. The categorical feature that we have encoded is the *protocolIdentifier*; in order to not have s huge dimensionality increase, we encoded only the most important protocols: Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Internet Control Message Protocol (ICMP). The traffic corresponding to

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

other protocols was discarded; however, the number of packets corresponding to these protocols was not substantial. With the type of encoding that we use, for every distinct protocol, we create a new feature. If an instance belongs to a specific protocol, it will be indicated with 1, otherwise with 0.

The source and destination Internet Protocol (IP) addresses are both nominal features, but instead of being encoded, they are discarded. The explanation for this is that they are only used to identify flows and, for generalization purposes based on the domain knowledge, it is *deemed* that they do not contain information for traffic classification.

Other features that are omitted are source and destination ports because their distribution is extremely sparse, hence encoding these feature would greatly increase the dimensionality of the data. We would have to transform features with a sparse distribution into hundreds of features, increasing so the complexity and make this encoding unsuitable.

3.1.3 Scaling

Feature scaling equalizes the relevance of features before the analysis, assuming that each feature should weigh the same. Otherwise, features with a higher value range would be predominant. Many algorithms, especially the ones based on point distances, require feature scaling as an important step in pre-processing. k-NN uses the Euclidean distance measure and hence, it is sensitive to different ranges. NN also needs similar feature ranges because it uses Euclidean point distance. Since we used k-NN and NN algorithms in our experiments, we found it essential to scale the datasets.

The technique that we have used for the scaling process is the Min-Max scaling. This method will scale every feature value individually. The minimum number will be scaled into 0 and the maximum value into 1. Every other number will be linearly scaled into a number between 0 and 1.

The formula used to map the original values to the new values is:

$$x_{\text{scaled}} = \frac{x_{\text{original}} - min(x_{\text{original}})}{max(x_{\text{original}}) - min(x_{\text{original}})} \tag{3.1}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Min-Max scaling does not decrease the importance of outliers and does not change the shape of original distributions.

3.1.4 Train-Test splitting

In order to estimate the classification performance of an estimator, we need to split the dataset into two subsets:

- The training set.

  The training set is used to train the algorithm and to build our model. This algorithm learns the data and adapts the function such as it fits the mapping: *feature values for each sample => corresponding label.*

- The test set.

  The test set is used to assess the model efficiency. Since the algorithm has no knowledge of the samples in the test set, it must make predictions based on what it has learned from the training set. The predicted labels are then compared to the actual labels to see how accurate the algorithm is.

In our experiments, the train set contains 70% of the data and the test set 30%. This partition was conducted trying to keep the same ratio of normal-to-abnormal traffic distribution in each of the newly generated subsets. Hence, we will produce a training and testing set that preserve the same percentage for each class as in the complete set. The feature selection can be done using the entire dataset, but this can lead to over-fitting. To avoid this problem, we select the features by examining only the training set and then transferring the adjustments to the test set.

3.2 Feature scoring

In this step, we assess the importance of each feature in the prediction. We investigate the ability of ensemble methods and NN as feature selectors to determine the most

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

important set of features. For each of these methods, we took a different approach, i.e., we embed stability selection only for ensemble models and not for the NN approach. This is because, after stability selection, we train the learning algorithm on a dataset with half of the instances of the initial dataset. Since ODDS and DAMI datasets are already small, reducing the dataset would result in an insufficient number of samples for the NN to be trained.

3.2.1 Feature scoring with ensemble methods

For ensemble models, we use an example of an embedded feature selection method called: *stability selection*. This technique was first introduced by Meinshausen and Bühlmann in [MB10]. Later, a variant of it was used in [SS12]. Note that this is not a method on its own, but a technique designed to improve the performance of the employed feature selection algorithm.

Stability selection is similar to bagging in that we construct separate subsets of data using random sampling with replacement, named bootstrap samples, that include half of the samples from the original dataset. Note that we sample over the rows (instances) and not over the columns (features), hence, the instances in the bootstrap samples are represented by all features. The next step is to train a core algorithm to learn the data. Rather than applying the core algorithm to the entire set, we apply it to each bootstrap sample. In order to have a general approach, we wrap the core learner into a permutation importance wrapper, an agnostic algorithm that provides the calculation of the permutation scores for each feature, regardless of the core algorithm employed. Later, using a threshold, we pick the most important features in each bootstrap sample and create one subsets of features for each bootstrap sample. Then, based on the occurrence of the features across the created feature subsets, a stability score is computed.

This process is displayed schematically in figure 3.2.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

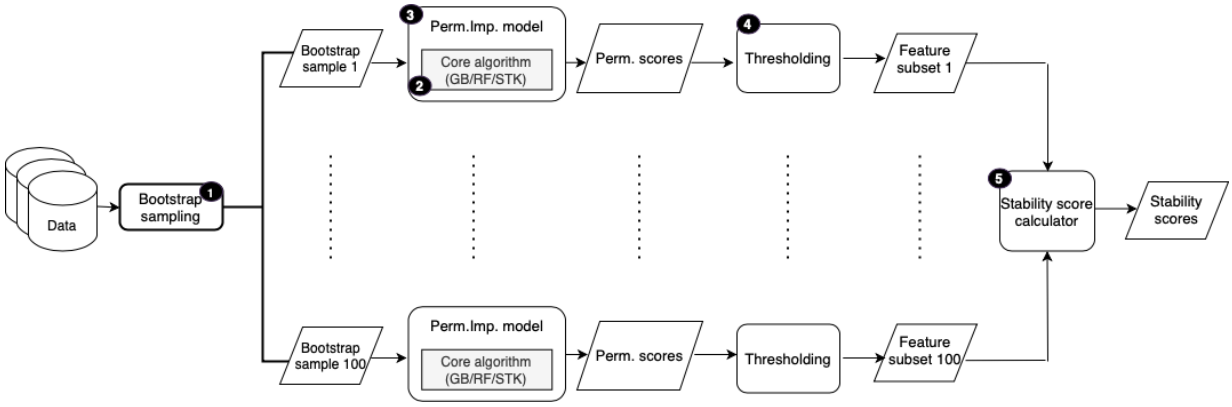


Figure 3.2: Stability selection process.

The steps that are followed to calculate the stability scores are explained in the following:

1. Bootstrap sampling.

   This step consists of generating bootstrap samples of the data by using random sampling with replacement. We have evaluated the selection probabilities over 100 bootstrap samples, a number recommended in [SS12], and each of these sets contains $n/2$ instances, where n is the total number of instances in the original data set. So we get 100 subsets (called bootstrap samples) where each of these subsets contains n/2 instances, randomly selected from the n samples in the whole data set.

   Then, the algorithm used to learn the data, the core algorithm, will be trained with each of the bootstrap samples separately.

2. Core algorithms.

   The algorithms that are employed as core learners are the ensemble methods introduced previously in 2.3.2. Three examples of the ensembles that we employ in this work are listed as follows.

   - Bagging, represented with *Random Forest* *Random Forest (RF)* *classifier.*

     RF builds a forest of decision trees in parallel and allows them to vote for the most popular class. The final decision is reached by averaging the votes by

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

each tree. These algorithms are appealing because they have strong predictive performance, little over-fitting, and are easy to interpret.

- Boosting, represented with *Gradient boosting* *Gradient Boosting (GB)* *classifier.*

  GB constructs the trees sequentially. Following that, the underlying algorithms will learn in an adaptive manner. The first model will make some predictions on the initial dataset, where each sample has the same weight, and each misclassified sample will be assigned a larger weight. Later, the weights will be updated and the next weak learner will be trained on this updated dataset. This process will be repeated for all the weak learners. In this way, this algorithm gives higher weights to wrongly classified samples, and since anomalies are most likely to be misclassified, it gives higher importance to the minority class. As a result, this method can be thought of as an upsampling process, but one that uses weighting to select samples rather than the number of samples. This method makes boosting an excellent technique for finding anomalies in imbalanced datasets.

  The authors in [XHWZ14] have described this behaviour in a similar work.

- *Stacking of the baseline classifiers (Stacking (STK)).*

  Stacking is another ensemble approach, and unlike bagging and boosting, it is constructed out of heterogeneous learners. As weak learners, we use the three most common baseline classifiers: DT, LR, SVM. As a meta-learner to combine the results, we use LR.

3. Permutation importance model.

   A permutation importance model is wrapped around the core algorithm and later, it is used as an estimator to measure the permutation scores on each bootstrap sample.

   The reason for using a permutation importance model is because they do not rely on the core algorithm, hence, they do not require specific attributes from the

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

core algorithm to measure how important a feature is. Since the *stacking classifier* does not supply the necessary attributes required from learner-dependent methods to calculate the importance of a feature, we wrapped our core algorithm into a *permutation importance model*. This model enables us to measure permutation scores for every core algorithm and they indicate how much the prediction accuracy will decrease if a particular feature does not have any contribution to the prediction.

To achieve such an effect, we choose the feature for which we should calculate the importance, and we shuffle the values that this specific feature has for different instances, while letting the other feature values and labels unchanged. In this way we break the relation between the feature and the label. In this work, we have shuffled each feature 5 times. Repeating the shuffling procedure more than 5 times, did not significantly affect the results. Furthermore, because the permutation wrapper will be trained with each bootstrap sample, meaning that it will be trained 100 times with subsets of size $n/2$ (with n being the total number of instances in the initial dataset), increasing the shuffling times would be inefficient in terms of running time.

The next step is to evaluate the prediction performance before and after shuffling. The drop in the prediction performance after shuffling, will indicate the permutation importance of the shuffled feature. Finally, we return the feature values to the original order and repeat the process for the next features.

It is worth noting that the permutation scores do not add up to one, and we are not interested in the absolute worth of features, but rather their relative predictive power. Permutation scores can often have negative values. This shows that the prediction using the shuffled data is more accurate than the one in the original dataset, implying that the feature is not important but it happened by chance that the permutation of values resulted in higher accuracy. In our tests, the procedure is replicated five times and we average over the produced scores. In this way, we reduce the effect of random permutations.

4. Thresholding. To obtain a set of *most relevant features*, we need to establish a

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

threshold for the permutation importance scores, which is automatically set to the mean of the obtained permutation scores. Features with a lower score are removed; therefore, only features with high scores are passed to the Selection probability block.

5. Stability scores.

   Based on the number of times that a specific feature appears in a boostrap sample after *thresholding*, we calculate the probability of a feature to be selected in general. For instance, let's suppose that we have derived 100 bootstrap samples from the initial dataset and hence, we have 100 subsets of features (one subset for each bootstrap sample). Then, we count how many times each of the features is revealed in these 100 substes. If a feature has been chosen in 80 out of 100 subsets, then the stability score of this feature is 0.8.

3.2.2 Feature scoring with Neural Networks

Next to ensemble methods, we also explored the ability of NN as a feature selector. As previously mentioned, we do not include stability selection in neural networks. This is due to the fact that NN requires more data to be trained than the other considered algorithms. When we use stability selection, the estimator is trained in bootstrap samples that contain half of the total dataset instances. Since the ODDS and DAMI datasets are small, the bootstrap samples are insufficient for the NN to learn and evaluate its parameters. As a result, unlike ensemble approaches, permutation scores are directly used to choose the most important features. The implementation of this method is straightforward; we wrap our NN in a permutation model and calculate the permutation scores. Later, we use these scores directly to select the most important features.
Our NN is implemented layer by layer creating a sequential model.

The three components of the NN and the specified parameters for conducting the experiments are:

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- Input layer.

  The number of nodes in the input layer corresponds to the number of features in the specific dataset used. As for the activation function, we have assigned it to be the *ReLU* function.

- Hidden layers.

  In our experiments, we have created two hidden layers where the first layer has 150 neurons and the second one has 100 neurons. The activation function is chosen to be again the *ReLU* function.

- Output layer.

  This layer contains only 1 neuron because we have a classification problem i.e. the output layer will predict one of the two options: 0 or 1. The activation function for the output layer is the *Sigmoid* function.

The activation functions are chosen to be *ReLU* and *Sigmoid* in order to make sure that predictions are probabilities in the range between 0 and 1.

Other components that we need to specify when we build the NN are:

- **Loss function.**

This function estimates the error for the actual state so that the weights can be updated intending to minimize the error in the next evaluation. We applied a loss function suitable for binary classification problems, the *binary-cross-entropy*. This function calculates the average difference between the actual probability distribution and the predicted prbability, trying to minimize it when anticipating class *1*. For using this function, the prerequisites are having a single node in the output layer and the activation function to be *sigmoid*; these conditions are satisfied in this work.

- **Optimization algorithm.**

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

An optimizer is a method that the NN uses for updating the weights or learning rate aiming to minimize the losses. There are several options that one can choose when building the NN. In this work, we have utilized *Adam* [KB17] as an optimization algorithm. This method computes individual learning rates for different parameters.

NN is very likely to suffer from over-fitting. An efficient solution to prevent this event from happening while not increasing the complexity is by randomly dropping some nodes during the training phase. This is called **Dropout**. During this process, the specified amount of nodes are removed temporarily from the network, and also all their incoming and outgoing connections are ignored. In this way, the training process becomes noisy and randomized resulting so in an increased ability of the NN to generalize. The dropout rate is specified in the input and hidden layers, but not the output layer.

In our problem, we have used a dropout rate equal to 0.25 for the input layer, and for the hidden layers, the rate is assigned to be 0.2 and 0.1 respectively.

3.3 Feature subset selector

The stability and permutation scores calculated in the previous stage differ from the feature importance and they do not sum up to 1. Therefore, we normalize them such that the sum of the stability or permutation scores equals 1. Then, we rank these normalized scores in descending order and calculate the cumulative sum. For each dataset, we later consider two subsets of features:

1. half of the most important features,
2. most important features that make up 80% of the total importance, i.e. top-ranked features whose normalized scores sum up to 0.8.

Hence, with this setup we evaluate the distinct feature selection methods twofold: on one hand, we see (a) their capability to properly rank the most relevant features for classification, and, on the other hand, we test (b) their power to discriminate relevant from irrelevant features.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

3.4 Validation stage

In the validation phase, we use different algorithms to test the suitability of the feature subsets emphasized by the feature selection methods under study. In this section, we cover the algorithms used for evaluation, their parameterization for ensuring that they work in optimal conditions, and the metrics used.

3.4.1 Evaluation Classifiers

To evaluate the performance of the chosen feature subsets we use three different classifiers, namely DT, k-NN, and NN. The rationale behind using these three different algorithms as evaluators is to, by means of a consensus among them, aim to decouple feature selection methods and their underlying algorithms from the algorithms used for the final evaluation.

We describe evaluation classifiers as follows:

1. **Decision Tree (DT).** The most widely used classification algorithms are DT. This is due to the fact that they are simple to understand and fast algorithms that do not necessitate a lot of computation. The way of working of this algorithm was previously outlined in section 2.1.6. When creating a DT, one can specify the number of parameters. The following are the most important ones:

   - *max_depth*: this parameter defines how much will be the depth of the tree and it has been tuned in our experiments. In our experiments, the range of values that this parameter can take for the hyperparameter optimization process is any number from 1 to 30.
   - *min_samples_leaf*: it shows how many instances are required to be at a leaf node and it has been also tuned in our experiments. The range of values for this parameter is assigned to be a number in {1, 2, 3 .....49, 50}
   - *class_weight*: it determines the weights of classes. By default, this parameter assigns the same weight to all classes, which is 1. In our problem, this is

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

an important parameter to specify because we are dealing with imbalanced datasets. The ideal approach in this scenario is to give more weight to the minority classes in order to achieve some balance. For this reason, we use the "balanced" mode which assigns higher weights to less frequent classes. The formula for this is:

$$w_{classes} = n_{samples}/(n_{classes} * np.bincount(class)) \tag{3.2}$$

where:

$w_{classes}$ shows the weight for the corresponding class

$n_{samples}$ is the number of total samples (rows)

$n_{classes}$ is the number of distinct classes

$np.bincount(class)$ is the number of samples for a specific class.

2. **k-nearest neighbors (k-NN).**

k-NN is another algorithm that we employ for evaluation. This classifier is not used as a weak or meta learner for stacking, nor as a base estimator for feature selection. We use it only as a classifier for calculating the prediction performance in different portions of the datasets. Despite its high predictive performance, this algorithm is slow during testing and evaluation because it checks all data points before evaluating each new sample. Subsection 2.1.5 explains how this classifier works.

The following are the most important parameters to specify when implementing k-NN:

- *The number of nearest neighbors k*: by default, this parameter takes the value *5*, but we have tuned it in our experiments. The range of values for this parameter is specified to be: {5, 10, 15, 20, 25}.
- *Weights:* also included in the parameter tuning process and can take 2 values:

  - uniform: all points have the same weight.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

- distance: closer samples have a higher weight, hence, a greater influence than the samples being further away.

3. **Neural networks (NN).**

The third algorithm that we use to assess the prediction performance is a NN with the following specifications:

Layers	**Nr. of neurons**	**Dropout**	**Activation function**
Input layer	250	0.25	ReLU
Hidden layer 1	150	0.2	ReLU
Hidden layer 2	100	0.2	ReLU
Hidden layer 3	50	0.1	ReLU
Output layer	1	—	Sigmoid

Table 3.3: NN implementation for classification task.

The parameters of a NN are:

- *batch size*: this argument assigns the number of samples for gradient update. For instance, if we set the $batch_size = 100$, the algorithm will take the first 100 samples and train the network. Next, it takes the second 100 samples and trains again the network. The process goes on in this way until we have gone through all the samples of the dataset.

Before choosing a batch size, we should consider the following cases:

- Small batch size requires less memory and allows the network to be trained faster.
- On the other side, if the batch size is too small, the estimate will not be very accurate.

Because we are working with imbalanced data, if we consider only a few samples every round, it is possible that all of the samples in this batch belong to the same class, and as a result, the algorithm would not be able to learn well. In our most

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

imbalanced dataset, only 1.2% of all samples are outliers. This means that in 100 samples, on average only 1 or 2 of them will be outliers. Hence, we created the range of values for the batch size such as it takes also values higher than 100 in order to be sure that it contains instances belonging to both classes. Specifically, the batch can have any size included in: [32, 50, 100, 150, 200, 250, 300].

- *epochs*: this argument specifies the number of rounds/passes over the dataset. After 1 round, the algorithm starts again with the first sample having as a goal to achieve an output that fulfills the defined approximation requirements. In this work, this parameter is assigned to be 100.

3.4.2 Parameter tuning

Algorithms used for evaluation have a collection of hyperparameters that have a significant impact on their results. We use *parameter tuning*, also known as *hyperparameter optimization*, to improve estimation and tailor the model to the particular dataset. During this process, we search in a defined set of values for the specified parameters and check the performance for every model. The values that provide us with the greatest performance are selected and used for the prediction.

A search consists of:

- *An estimator*: in our case: DT, k-NN or NN.
- *A parameter space*: specified in the previous subsection 3.4.
- *A method for searching*: random search.
- *A cross-validation scheme*: stratified cross-validator.
- *A score function*: F1 score.

In this work, we use *Random Search* to choose the values for the hyperparameters with the stratified cross-validator where we specify how the parameters should be sampled

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

using a dictionary. Another common method is *Grid Search*[BB12], however, we prefer random search because of two reasons:

1. We can decide on the computation budget which does not depend on the number of parameters and their values. This is done using *"n_iter"* parameter.

2. If we add parameters that have no influence on the performance, they will not decrease the efficiency.

3.4.3 Scoring metrics

To evaluate the performance of the classifiers we compare the predicted labels to the original ones. For this, there are several metrics that can be used. The most used metric is *Accuracy* which is calculated as a fraction of correct predictions over total predictions, but this metric is not suitable for imbalanced datasets. Let's take the example of having a dataset in which only 2% of all samples are outliers. If an algorithm will just predict every single sample as "not an outlier", we will achieve the accuracy = 0.98. This is the reason that we have not used accuracy to evaluate the final performance.

More suitable metrics for imabalanced data are: F1 score, average precision, and ROC-AUC score; these are also the metrics that we have used in this work. All these metrics are derived from the confusion matrix, an example of which is shown in figure 3.3.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

		Actual labels	
		0	1
Predicted labels	0	True Negative	False Negative
	1	False Positive	True Positive

Figure 3.3: The structure of a confusion matrix for binary classification.

For binary classification, this matrix is constructed out of four fields:

- True negatives: predicted label and original (actual) label are both *0*.
- True positives: predicted label and original label are both *1*.
- False negatives: original label is *1* but the predicted label is *0*.
- False negatives: original label is *0* but the predicted label is *1*.

Based on the fields of confusion matrix, we can calculate precision and recall which then help in the definition of F1, average precision and ROC-AUC metrics.

1. Precision [GG05] shows how many of all the predicted positives are true positives. The formula for it is:

$$Precision = \frac{TP}{TP + FP} \tag{3.3}$$

2. Recall [GG05] shows how efficiently is the classifier to predict all the positive labels.

$$Recall = \frac{TP}{TP + FN} \tag{3.4}$$

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

3. F1 score [Sas07, GG05] is a combination of precision and recall and measures the harmonic mean of both of them. The formula to calculate F1 score is:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.5}$$

From the formula, we can notice that both Precision and Recall contribute the same to the F1 score. This is the reason that we chose to maximize this metric in the random search because by maximizing F1, we maximize both Precision and Recall simultaneously.

4. Average precision measures precision for every sample and multiplies the change in recall from the step before. Then, we get the average of all samples. [KR18]

$$AP = \sum_{k=1}^{n} P_k * (R_k - R_{k-1}) \tag{3.6}$$

Where P_k is the precision at sample k, R_k is the recall at sample k, and R_{k-1} the recall at the previous sample.

5. ROC-AUC score, or differently: Area Under the Receiver Operating Characteristic Curve, is the scoring metric that measures the probability of the algorithm to rank a randomly chosen positive instance as higher than a randomly chosen negative [Mel13, McC89, PVG+11]. ROC-AUC averages over all thresholds and is considered to be too optimistic in the case of imbalanced data. This indicates that if we achieve high scores of roc-auc, this may not always show a good performance of the model, but, it may be just the side-effect of too many negative examples. As a result, this metric does not drop drastically even when we have a considerable number of false negatives. In imbalanced data, we are more interested in checking how well our model can detect the rare cases, therefore average-precision or F1 is a better metric to use because it varies substantially with the amount of false negatives and false positives.

The reason that we use ROC-AUC is that it is the most widely used metric in research. However, it is not well suited to imbalanced datasets. F1 and average precision, on

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

the other hand, provide less biased evaluations in datasets with a strongly different distribution of classes, average precision being preferable over F1, whose definition is more arbitrary. However, F1 is more popular and commonly available by default within the internal learning processes of the software implementations of the algorithms used.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

CHAPTER 4

Results and discussion

In this chapter, we present and discuss the results of the conducted experiments which are summarized in table 4.1.

First, the feature scoring is discussed (experiment 1 in table 4.1). To asses the feature scores, we have employed stability selection Stability Selection (SS). This technique is based on bootstrapping in combination with a core algorithm. The bootstrapping uses sampling with replacement to create 100 sub-sets of the initial dataset, each of them containing half of the total number of samples. The feature scoring algorithm is applied on each of the bootstrap samples.

The next step is reducing the number of features based on their scores (experiment 2 in 4.1). We chose two subsets of features: a) half of the total number of features, b) features that make up 80% of the total importance.

The last step is evaluation of feature selection approaches (experiment 3 in 4.1). We use three popular algorithms for the evaluation process: DT, k-NN, NN and we analyze individually the behaviour of the four approaches: stability selection with random forest (Stability Selection with Random Forest (SS-RF)), stability selection with gradient boosting (Stability Selection with Gradient Boosting (SS-GB)), stability selection with stacking (Stability Selection with Stacking (SS-STK)), feature selection with the neural

network (Feature Selection with Neural Network (FS-NN)) on each specific split of the dataset: the split containing half of the features, the split containing features that account for 80% of total importance and the initial dataset.

To conclude this chapter, we review our findings and provide a final assessment of the results.

Experiments	**Parameters**
1. Feature scoring	
1.1 Stability selection	
1.1.1 Bootstrapping	Method: Sampling with replacement
	Subsets: 100 subsets
	Samples per subset: (total samples)/2
1.1.2 Permutation model	Nested algorithms: RF, GB, STK
2. Feature subset selecting	
2.1 Selecting 2 different subsets of features	50% of all features
	Features of 80% of total importance
3. Validation using DT, k-NN, NN	
3.1 DT	Max depth: {1, 2, 3 .....49, 50}
	Min samples leaf: {1, 2, 3 .....29, 30}
	Class weight: balanced
3.2 k-NN	Nr. of neighbors: {5, 10, 15, 20, 25}
	Weights: uniform, distance
3.3 NN	Input layer: Nr. neurons = 250, Droupout=0.25, Activation function = ReLU
	Hidden layer 1: Nr. neurons = 150, Droupout=0.22, Activation function = ReLU
	Hidden layer 2: Nr. neurons = 100, Droupout=0.22, Activation function = ReLU
	Hidden layer 3: Nr. neurons = 50, Droupout=0.1, Activation function = ReLU
	Output layer : Nr. neurons = 1, Activation function = Sigmoid

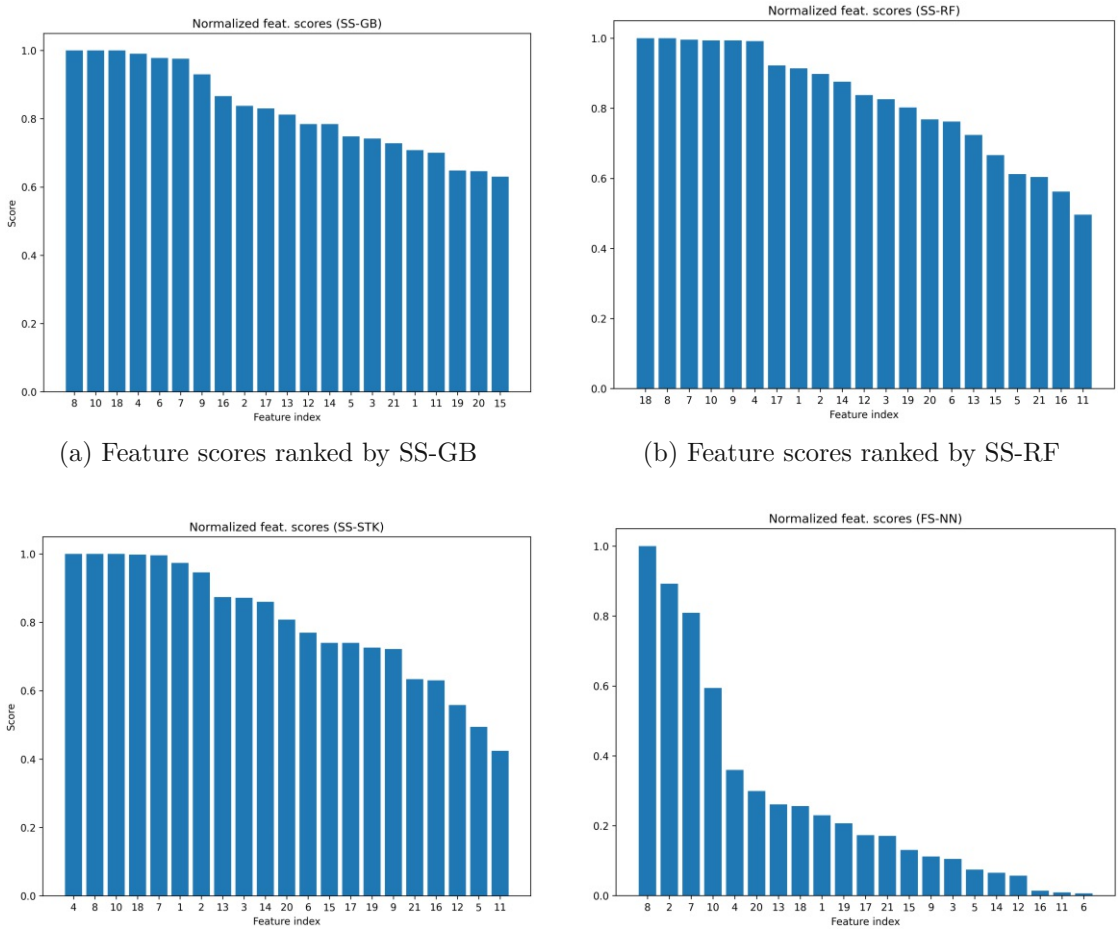
Table 4.1: Summary of the conducted experiments.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

4.1 Feature Scoring

Each selection technique assigns different ratings to distinct features as described in 3.2. Figure 4.1 shows the ranked attributes of the *Cardio* dataset produced from the four techniques, where the index of each feature is shown on the X-axis.

Note that in this example, the feature scores are normalized by dividing them by the maximum value. As a result, these scores do not sum up to 1.



(a) Feature scores ranked by SS-GB

(b) Feature scores ranked by SS-RF

(c) Feature scores ranked by SS-STK

(d) Feature scores ranked by FS-NN

Figure 4.1: Feature scores for Cardio dataset ranked by different algorithms. Note that the scores are normalized by dividing them by the maximum feature score value and that the features on the x-axis are sorted according to their importance.

The orders of features have some similarities over the four methods; for example,

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

all four techniques agree that features with indexes 8, 10, and 4 are among the most essential; nevertheless, the exact rank varies depending on the technique. The shift between very significant and less critical features in SS-RF and SS-GB is usually smooth. As a result, they have a low discriminative capability, i.e. the ability to differentiate between important and less significant features is low. SS-STK, on the other hand, distinguishes better between features than SS-RF and SS-GB. The method with the highest discriminative power is FS-NN. This behaviour comes as a result of relying solely on permutation importances. Permutation importance is defined as the reduction in a model score when a single feature is shuffled randomly. This approach replaces each feature with random noise by shuffling it and later, the model score calculated without the effect of a specific feature indicates how dependent the model is on that feature. Hence, it assigns drastically low values for correlated features because dropping such a feature will not affect the outcome as the algorithm will learn the information from the other attributes. As a result, the disparity between feature scores will widen.

In figure 4.2 the heatmap depicts the Spearman correlation of each combination of features (feature pairs) for the *Cardio* dataset. The Spearman correlation is used as a distance metric to derive also the hiearchical clustering. This is an algorithm used to group similar instances into clusters based on a distance metric, which in our case is the Spearman correlation. The output of the hiearchical clustering is the dendogram shown in the left side of figure 4.2. The x-axis in this figure shows the feature indexes, while the y-axis show the distance or dissimilarity. The higher the level in the y-axis that two features join together, the smaller the correlation between them.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

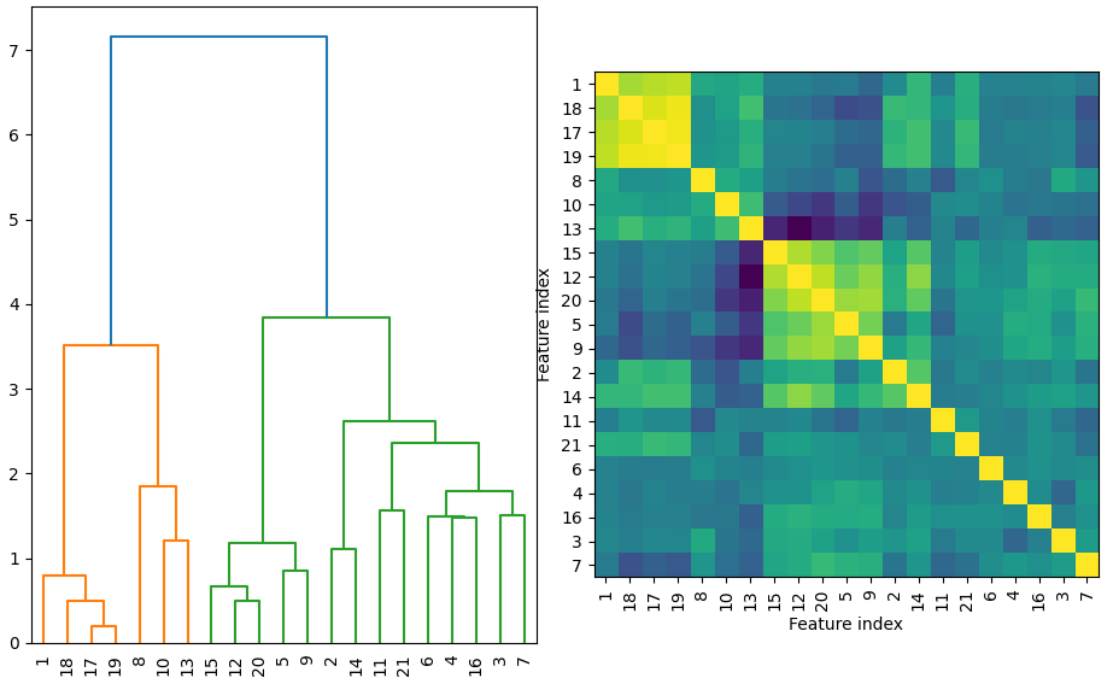
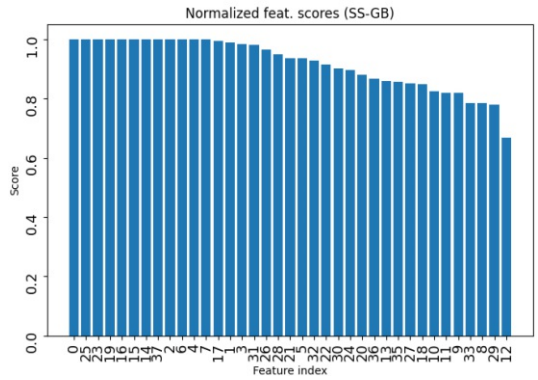


Figure 4.2: The hiearchical clustering and the derived spearman correlation between features for the dataset *Cardio*. The more yellow the color, the stronger the correlation.

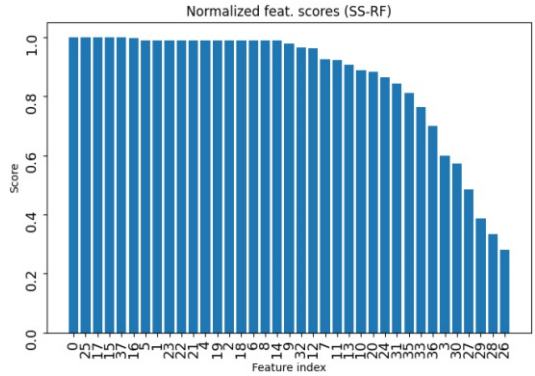
We can notice that features of indexes 1, 18, 17, and 19 show a strong correlation; in the figure in the left side (the dendogram) they have joined the same cluster in a low level on y-axis, and in the figure in the right side (heatmap) this is shown by the strong yellow color. Hence, the FS-NN approach assigns them low and very similar values, as shown in the corresponding histogram in figure 4.1. We can notice also other features with even lower scores than the correlated features. This is due to their insignificant role in the prediction caused by other reasons than correlation.

This type of behavior can be observed across all datasets. For instance, let's take into consideration the dataset *CIC2017*. Likewise in the *Cardio* dataset, we can notice that many of the features chosen as the most important ones from different approaches, are the same regardless of the used algorithm. For instance, in figure 4.3 we can depict that attributes with index 25, 14, 17, 37 are assigned high values from every approach, but again, we notice that their ranking order differs for each method.

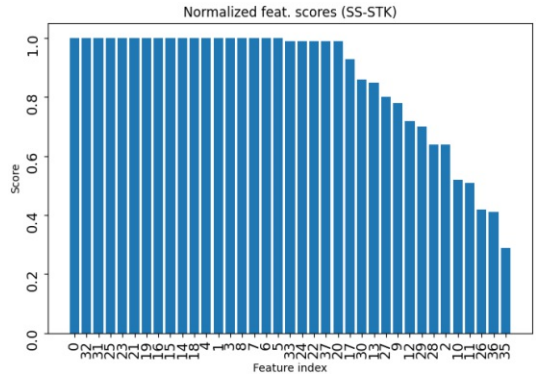
Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN



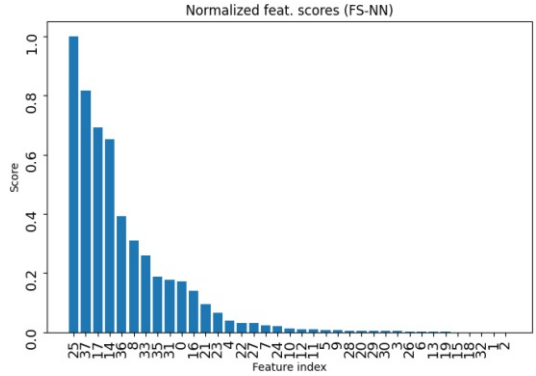
(a) Feature scores ranked by SS-GB



(b) Feature scores ranked by SS-RF



(c) Feature scores ranked by SS-STK



(d) Feature scores ranked by FS-NN

Figure 4.3: Feature scores for CIC2017 dataset ranked by different algorithms. Note that the scores are normalized by dividing them by the maximum feature score value and that the features on the x-axis are sorted according to their importance.

The feature correlation map for this dataset is illustrated in figure 4.4. We can notice that features of index: 1, 2, 18, 4, 6 show a high correlation, hence, they are assigned very low values from FS-NN; this is shown in corresponding subplot in 4.3. However, correlation is not the only cause for the low rankings given to features. The low scores could possibly be also due to their insignificance, as they do not provide any useful information on the label prediction.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

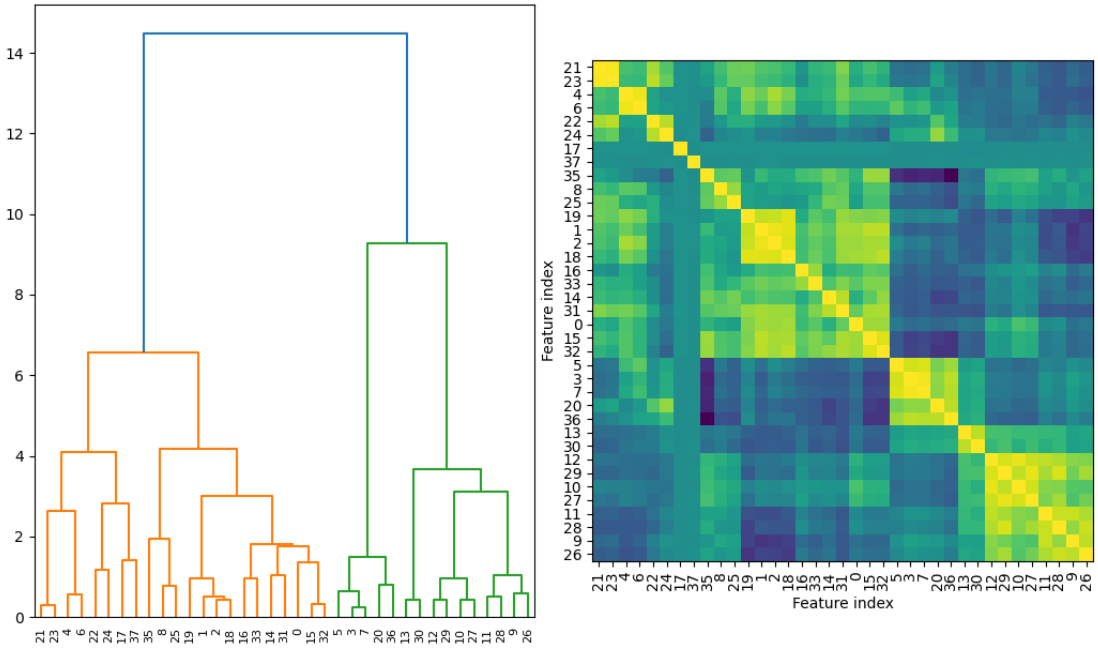


Figure 4.4: The hiearchical clustering and the derived spearman correlation between features for *CIC2017* dataset.

4.2 Feature subset selecting

The classification results are generated using the whole set of features as well as two further partitions of the dataset:

a) A subset of data containing only half of the features that are ranked highest with the score from a certain approach. This split yields the same amount of features regardless of the selection method because the selection criteria here is the number of features and not the minimum score.

b) A subset of data containing the highest ranked attributes that account for 80% of total importance.

Note that differently from the examples in 4.1 and 4.3, here feature scores are scaled such as their sum is 1, to simplify the selection of the features. Later, they are ranked in an descending order and we choose the desired amount of features based on the conditions

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

a) and b).

Table 4.2 lists the number of features resulting after different approaches of feature selection [1].

Dataset	ALL	Half	RF-80	GB-80	STK-80	NN-80
Pendigits	16	8	12	12	11	7
Cardio	21	11	15	15	15	9
PageBlocks	10	5	7	7	7	4
Vowels	12	6	9	9	9	5
Satimage	36	18	28	28	27	14
SpamBase	57	29	40	40	41	29
Mnist	100	50	74	69	74	17
Satellite	36	18	28	28	28	22
CIC2017	38	19	24	28	24	12
Botnet	38	19	26	27	27	10
Unsw	38	19	21	25	26	12

Table 4.2: Number of features selected by every approach.

4.3 Evaluation scores

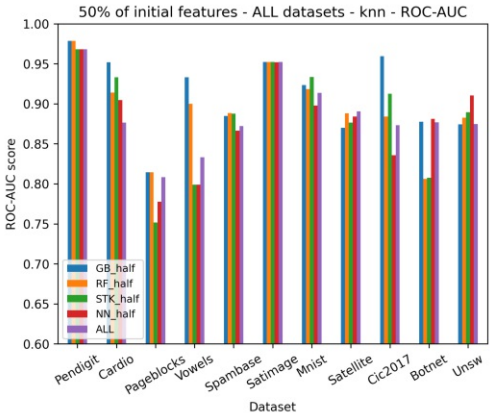
After feature selection, three models were created with the best parameters derived from the random search and used to predict the class of each sample in the test set. The best parameters extracted for each model can be found in the appendix 6.1.
Classification scores calculated in both train and test sets are listed in the appendix 6.6 - 6.17. In these tables, we notice that the training and test results of each model are almost the same, indicating that the prediction model does not over-fit the training set. In the following we analyze individually the behaviour of the approaches on each specific split of the dataset. Note that the focus of this thesis is to compare alternative feature selection approaches and reveal the optimal strategy, rather than improving the classification performance.

[1] 'ALL' represents the initial number of features for every dataset, 'Half' represents half of the initial number of features, and 'RF-80', 'GB-80', 'STK-80', 'NN-80' represent the number of features resulting after selecting only features of 80% of importance with the corresponding approach.

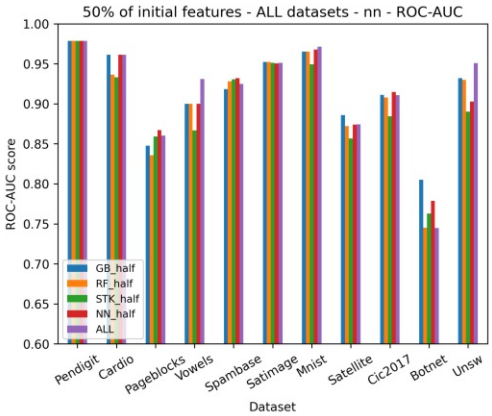
Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

4.3.1 Half feature-subset evaluation

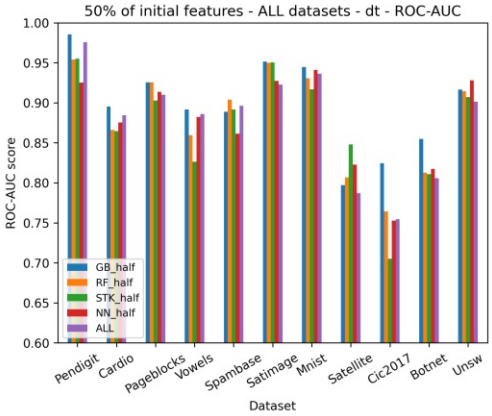
The classification scores obtained on datasets with only half of the initial number of features are shown in figures: 4.5 - 4.6 .



(a)



(b)



(c)

Figure 4.5: ROC-AUC scores evaluated by DT, k-NN and NN when using half of the most important features.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

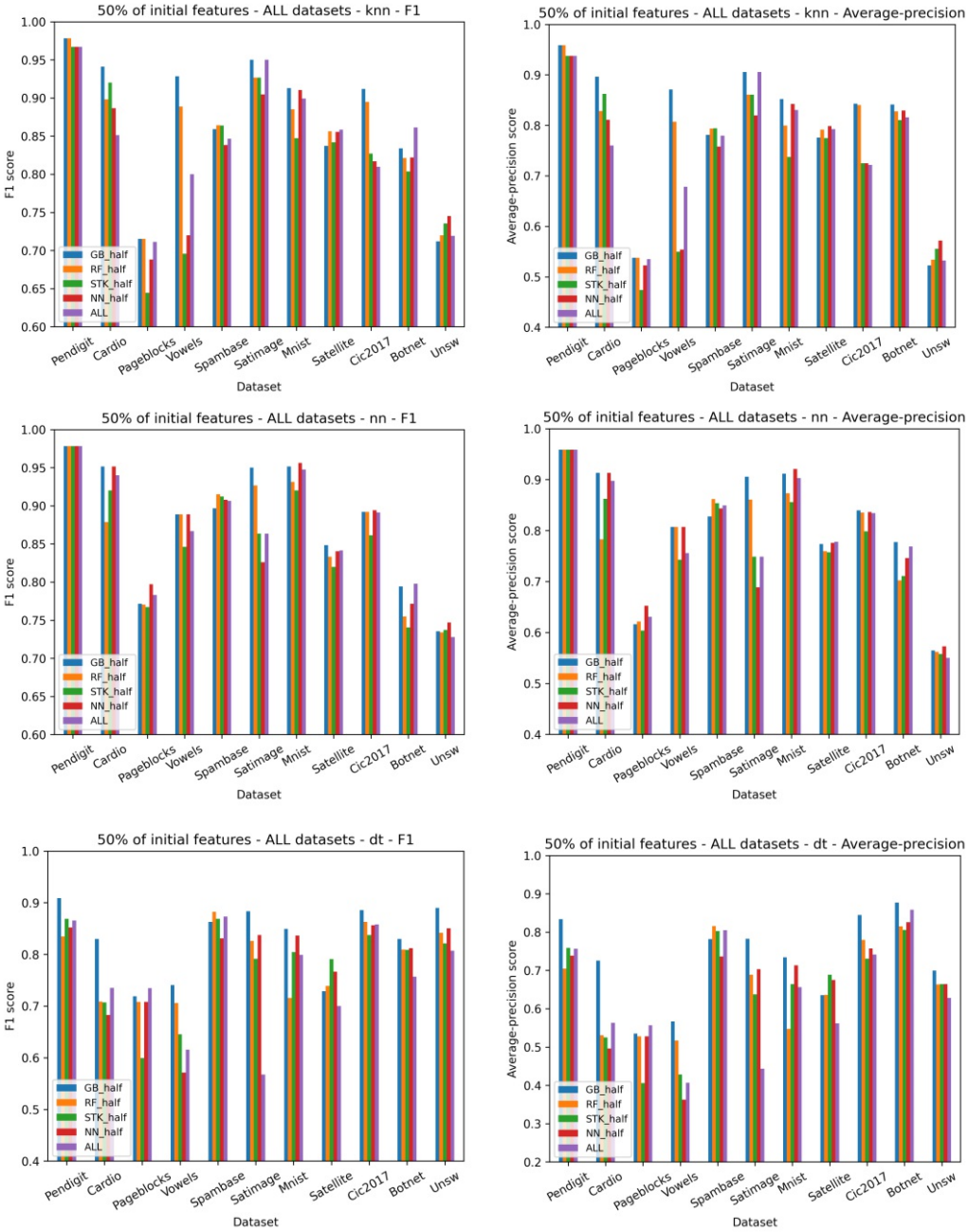


Figure 4.6: F1 and average-precision scores evaluated by DT, KNN and NN when using half of the most important features.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

These plots illustrate the three separate metrics calculated from each classification model. We compare the results of several methods to each other as well as to the results on the baseline dataset marked by *ALL*. These plots reveal that SS-GB outperforms other approaches in almost every dataset regardless of the classifier or scoring metric. The SS-GB algorithm performs particularly well with the *Satimage* dataset, which has the fewest outliers (only 1.2 percent of the samples are positive examples). The next best method is SS-RF, followed by FS-NN. This poorer performance of FS-NN can be a result of using small datasets that do not contain enough samples for the neural networks to correctly determine its parameters. Another cause for the poorer performance of FS-NN is the lack of stability selection, which has increased ensemble methods selection performance.
Regarding SS-STK, we can see that it produces inconsistent results across datasets or classifiers.

To summarize our results, we use *critical difference diagrams* shown in figure 4.7, originally proposed in [Dem06].

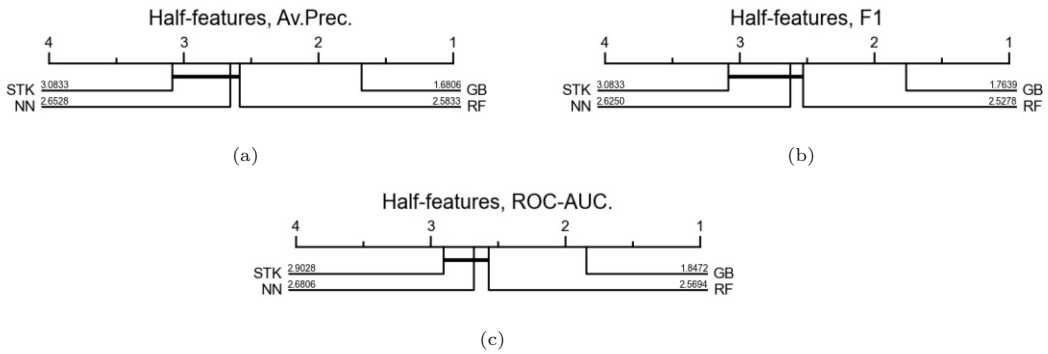


Figure 4.7: Critical Difference (CD) diagrams showing the results of a statistical comparison of the performance of every feature selection method on subsets with only half of the initial features. The bold lines connects approaches are not statistically different. The numbers show the average ranks. The lower the rank, the better the performance of the approach.

These graphs are typically used to compare algorithms across numerous datasets. They calculate the average ranks for each method, which is the average of the rankings that

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

a particular method displays across all datasets. The better the method, the lower the average rank. The bold lines in these diagrams indicate groups of approaches whose performances do not show statistically significant differences. The corresponding diagrams for our experiments expose that, regardless of the scoring metric, GB clearly performs better than the other approaches with an average rank of around 2. The rest of the methods reveal no significant differences.

Imbalanced datasets are challenging because the minority class lacks enough examples for the algorithm to learn and to generalize well. GB is an excellent approach for this type of data since it focuses on challenging samples by enhancing the impact of the minority class. Throughout the boosting process, we adjust the distribution of samples since the minority class is assigned greater weights. Thus, we may view boosting as an advanced sampling strategy that mitigates the effect of having imbalanced labels.

4.3.2 80% importance feature-subset evaluation

For each dataset and feature selection methodology, selecting only features with an 80% relevance yields different numbers of attributes. In the case of ensembles, this number is higher than half of the features, but not in the case of neural networks. Figures 4.8 and 4.9 illustrate the classification scores on the newly formed subsets.

Overall, the results for ensemble methods (SS-GB, SS-RF, SS-STK) over several datasets are less consistent than in the preceding scenario. This could be due to the addition of new features that can happen to be redundant or irrelevant features and mislead the outcomes.

We particularly notice that SS-RF shows improved relative scores in this scenario and surprisingly, it even outperforms SS-GB in the *Satimage* dataset. However, since this is a very difficult dataset with just a minimal number of outliers, this result can not be considered representative for all datasets. SS-GB on the other hand shows continuously very good performance. As for the dataset with the highest amount of outliers, the *Satellite* dataset with 36% positive samples, SS-STK is the technique that produces the best results.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

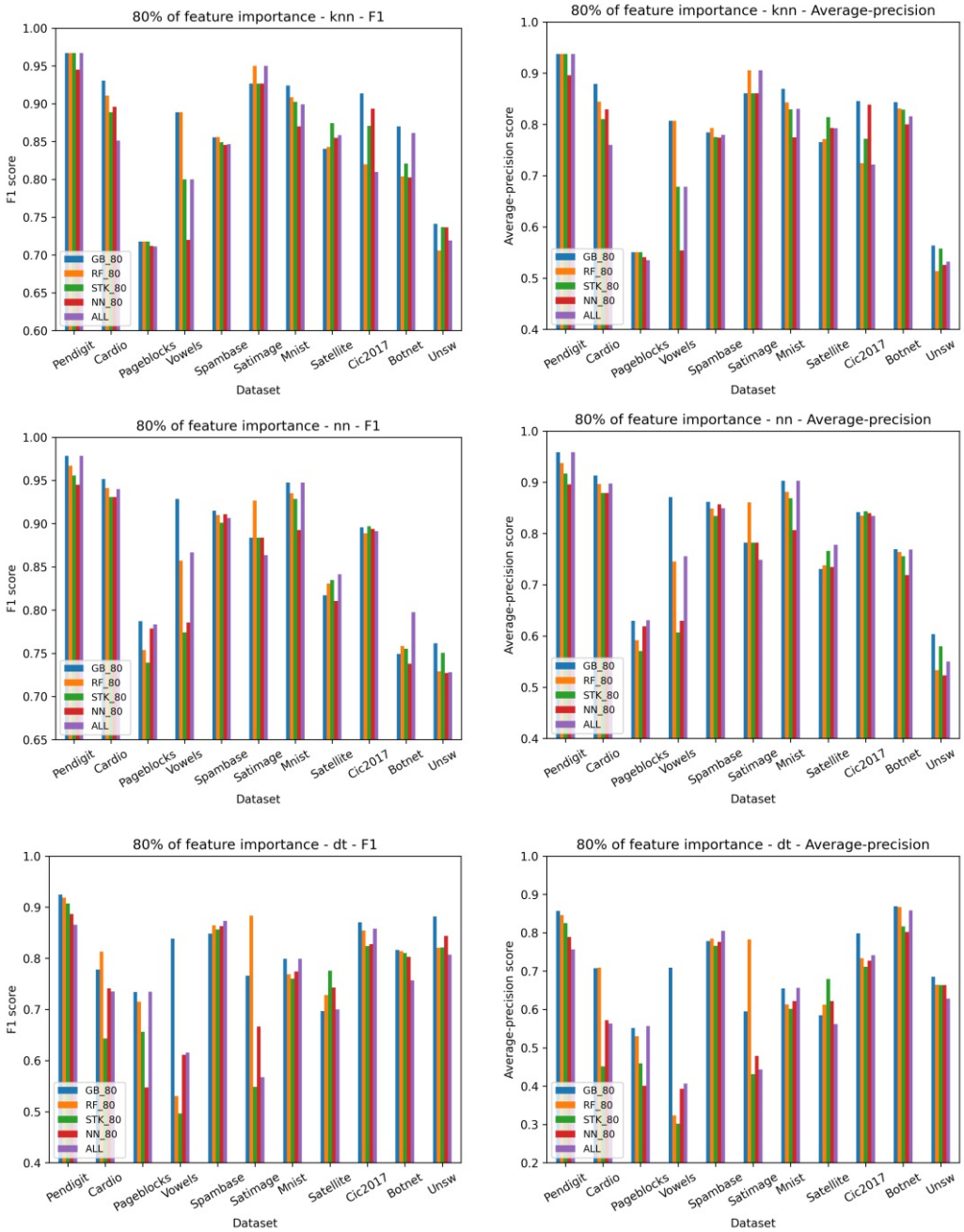
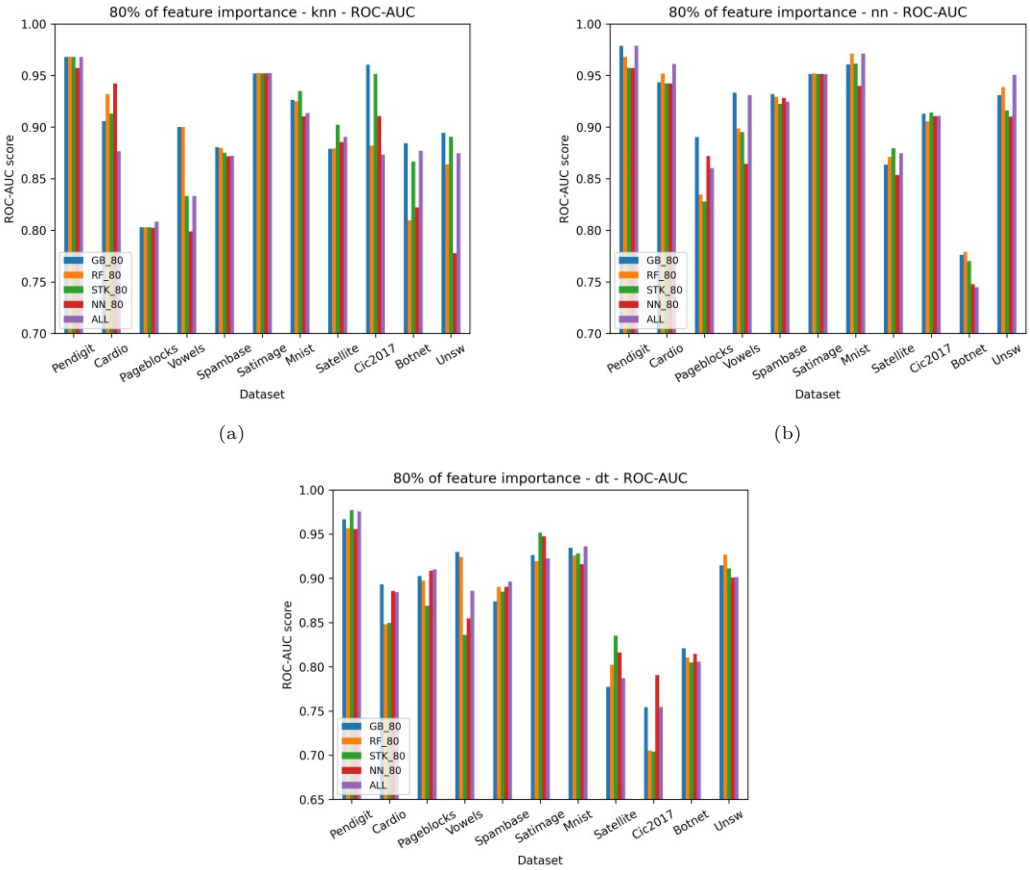


Figure 4.8: F1 and Average-precision scores evaluated by DT, k-NN and NN for features of 80% of total importance.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.



(a) (b) (c)

Figure 4.9: ROC-AUC scores evaluated by DT, k-NN and NN for features of 80% of total importance.

To summarize our finding, we use again the critical difference diagrams 4.10. FS-NN, as we noticed previously, has high discriminative power, resulting in a minimal number of features when selecting features that account for 80% of the relevance. This leads to a degradation of the performance, even though negligible. As a result, the FS-NN approach is listed as the algorithm that performs the worst in this split of the dataset. However, if we are looking for the fewest number of features possible, a performance hit is acceptable. For instance, in the *Mnist* dataset, we can reduce the number of features from 100 to 17 and still get an F1 score reduced just by 0.0292. SS-GB, on the other hand, is the method that performs best, however, the rank difference with SS-RF is less

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

substantial than in the prior split of the dataset.

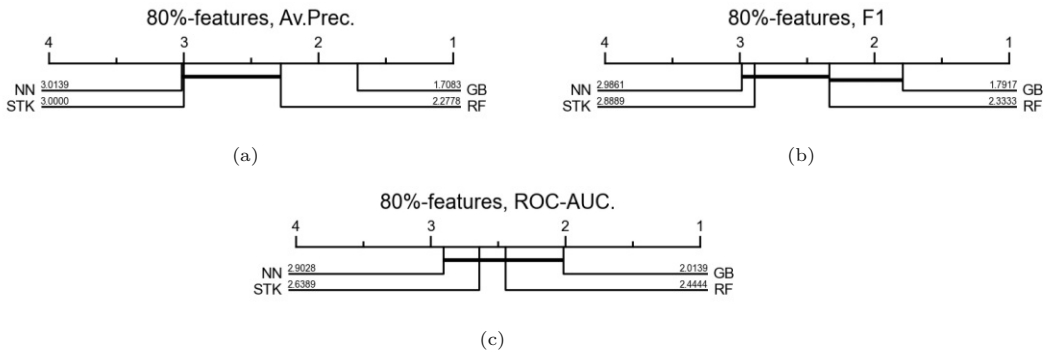


Figure 4.10: Critical Difference (CD) diagrams showing the results of a statistical comparison of the performance of every feature selection method on subsets of features with the relevance of 80% of importance.

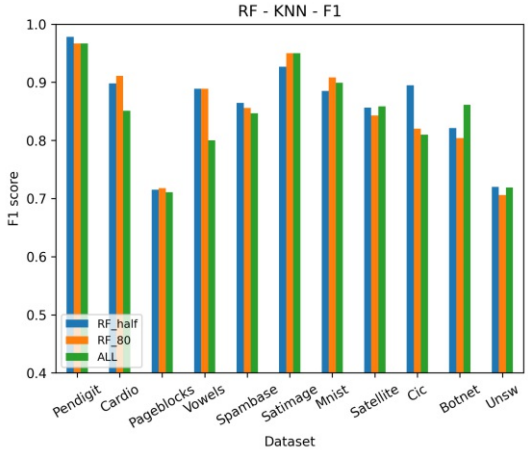
4.4 Performance degradation

The more features we reduce, the higher this degradation can be. In the following, F1 scores calculated by k-NN are compared on both subsets of the data (half and 80%) and also on the initial dataset with all the features, denoted by *ALL*.

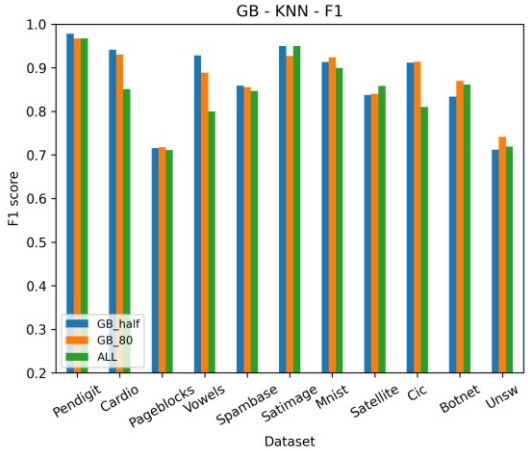
The reason for using the scores from k-NN is because this algorithm has not been previously used in the feature selection process. Hence, the results of k-NN are not biased.

Figure 4.11 shows that generally, we achieve higher performance when we reduce features. This is a result of removing correlated and irrelevant features that mislead the predictions. Nevertheless, it may happen that reducing features results in performance degradation. This is noticed mostly in the cases when we reduce the most number of features, for instance, in the case of FS-NN for 80% of importance subsets. However, this degradation is not that strong, and reducing features is commonly acceptable.

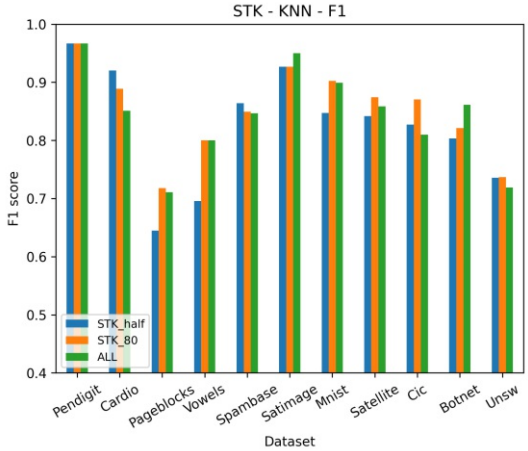
Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN



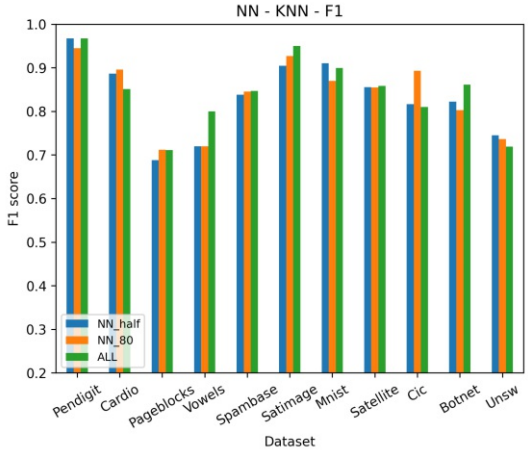
(a) SS-RF



(b) SS-GB



(c) SS-STK



(d) FS-NN

Figure 4.11: Performance scores on the two splits of the datases resulted from the specific feature selection method and on the intial dataset.

4.5 Summary of results

To summarize our findings, we reconsider the classification scores calculated by the three classifiers 4.6 - 4.9. NN and k-NN achieve consistent and high classification scores across datasets while the decision tree produces poorer results. It is worth noting that F1 scores and average precision have very comparable relative values. On the other hand, not every time they are consistent with roc-auc scores. For example, when we select features of 80%

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

relevance for the *Satimage* dataset, SS-RF shows the highest performance in terms of average precision and F1. However, this is not the case for roc-auc scores. As previously explained in 3.4.3, this metric does not show a very accurate result for imbalanced dataset because it averages over all thresholds, giving us a result which is too optimistic.

The choice of the feature selection method strongly depends on the target. If the purpose of such a process is selecting the smallest number of attributes, FS-NN with only features of 80% relevance is the method that yields the smallest feature subset. If the goal is improving classification scores, SS-GB produces the best results. This approach outperforms the other techniques especially when only half of the attributes are chosen. Figure 4.12 illustrates the relation between performance degradation and the number of selected features. The performance reduction is calculated as the difference of the average precision of all features and average precision scores in the reduced datasets: "$AvPrec_{AllFeats} - AvPrec_{ReducedFeatSet}$", while the portion of selected features is calculated as "$NumberSelectedFeats/TotalNumberFeats$". The performance of both approaches is visually compared in the figure; squares represent performance degradation for half of the features, circles for features of 80% importance. Different colors, on the other hand, show the performances for different datasets. We can see that selecting half[2] of the features not only results in fewer attributes but also leads to a larger improvement in classification scores. This indicates that there are a lot of attributes that are not relevant. We see the largest improvement in performance in the *Vowels* dataset. In the *UNSW* dataset, SS-GB has the highest discriminative power, picking around 65 percent of features. On the contrary, *CIC2017* is the dataset where SS-GB selects the most features, around 80%. This behavior can be noticed also in figure 4.3 where the shift from important to less significant features is very smooth.

[2]Some datasets have an uneven number of features, hence we can not take exactly half of them. For instance, Cardio and Spambase datasets which have 21 and 57 features respectively. In these cases we find the number of features to choose by rounding up the division: $TotalNumberofFeatures/2$.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

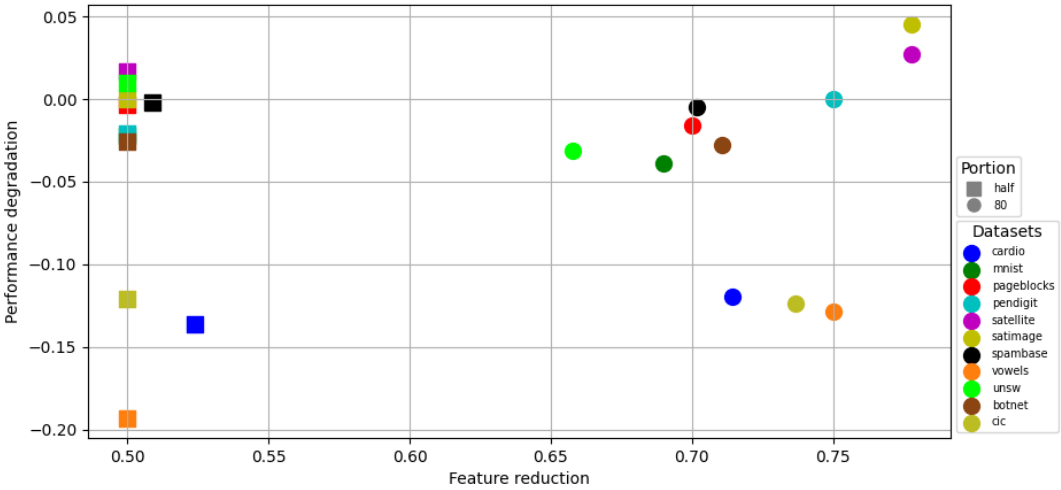


Figure 4.12: Comparison of the k-NN performance scores of SS-GB for selecting half or features of 80% of importance. Note that a *negative* performance degradation actually means a *performance improvement* caused by reducing the number of features.

A big advantage of SS-GB is that it is consistent across datasets and classifiers. In the case of the k-NN classifier, we can see that SS-GB can improve the performance by 0.2, with the most significant drop being almost 0.05. However, due to embedding stability selection, such a performance comes at the cost of higher computational power. As a result, if we are opting for the fastest method, FS-NN, which does not employ stability selection, would be a more suitable choice.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 5

Conclusion and further work

5.1 Conclusions

In this work, we investigated different approaches for feature selection in imbalanced datasets. We compared the capabilities of stability selection with ensemble models, as well as neural networks for selecting the most important attributes in a dataset. To accomplish this task, the experiments were performed on 11 labeled datasets related to outlier detection and intrusion detection. Three well-known machine learning algorithms were used as evaluators: K-nearest neighbors, neural networks, and decision trees to calculate three different scoring metrics: average-precision, F1, and roc-auc. Since we were working with imbalanced data, average precision and F1 yielded more reliable results because they heavily depend on false negatives and false positives, two necessary measures for fairly ranking methods in class-imbalance situations. Roc-auc scores, on the other hand, were inflated indices because of the high number of negative samples (and therefore less reliable for evaluation). The findings are drawn in terms of classification performance and discriminative power.

Four significant conclusions emerged from the experiments:

1. Ensemble models in combination with stability selection are demonstrated to be a

very effective technique in identifying the subset of features that produce improved classification scores. However, these approaches showed a smooth transition from a significant feature to less important ones, resulting in limited discriminative power.

2. The neural network approach, on the other hand, showed high discriminative power. As a result, while selecting features of 80% importance, FS-NN selected as important a very small number of features and removed the rest which corresponded to a large amount of features that equates to more than half of the total number of features. This leads to some performance degradation; nevertheless, in a considerable set of applications and analysis goals, this deterioration can be assumed.

3. Using permutation importance as a wrapper increased the gap between feature scores, allowing for better discrimination of relevant and irrelevant features. This could be because the neural network employed the permutation scores directly instead of using stability selection, resulting in a high discrimination power.

4. Although the input data and the classifier play a significant impact in the performance of each method, gradient boosting consistently achieved high classification scores and in most of the datasets, it surpassed the other approaches. This is especially noticeable when only half of the features were chosen.
   Neural networks gave poorer classification scores than ensembles. This could be caused by the small size of our datasets and absence of embedding stability selection. However, in terms of discriminative power, the neural network outperformed the other approaches.

5.2 Further work

Imbalanced datasets pose a challenge to learning algorithms because samples from the minority class are given insufficient weight and hence, they are considered less important. As a result, they may be overlooked or inadequately learned by the method, resulting in poor generalization and classification in the test set. *Siamese networks* are known to be

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

one-shot learners, meaning that they require very few samples of each class, therefore they can be incorporated instead of vanilla neural networks for such imbalanced data. These architectures consist of two identical neural networks and are able to learn by converting the classification problem into a similarity detection. We recommend the reader look into [Chi21] for more information. Another solution for imbalanced data is using up-sampling for the less frequent class. This is the reason why, in our work SS-GB outperformed the other approaches for the feature selection task because it re-samples the minority instances through their weights. A question that arises is if this approach will continue outperforming the other datasets also in the balanced data. As a result, this research can be expanded by analyzing the capabilities of the previous feature selection methods to datasets with a balanced distribution of classes.

Additionally, further work can consider merging the predictions of gradient boosting, random forest, and neural network into one ensemble model. This can be accomplished by combining the results using an advanced ensemble integration method, such as stacking. Later, stability selection can be incorporated to improve results.

Finally, it would be interesting to extend the comparison and include traditional filters, wrappers, and embedded options for feature selection and study performances in terms of accuracy, discriminancy, and computational requirements.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 6

Chapter Appendix

6.1 Best parameters

The values that are chosen from the random search (discussed in subsection 3.4.2) in every subset of each dataset is listed in tables 6.1 - 6.5:

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

Dataset	Portion	Best: DT	Best: k-NN	Best: NN
Cardio	half_gb	min_samples_leaf= 13 max_depth= 16	k= 20 weights='distance'	batch_size=300
	80_gb	min_samples_leaf= 14 max_depth= 23	k=5 weights='distance'	batch_size=200
	half_rf	min_samples_leaf= 2 max_depth= 16	k=10 weights='distance'	batch_size=300
	80_rf	min_samples_leaf= 1 max_depth= 26	k=5 weights='uniform'	batch_sizes=300
	half_xtf	min_samples_leaf= 5 max_depth= 10	k=5 weights='distance'	batch_size=200
	80_xtf	min_samples_leaf= 25 max_depth= 14	k=5 weights='distance'	batch_size=250
	half_stk	min_samples_leaf= 14 max_depth= 24	k=5 weights='distance'	batch_size=250
	80_stk	min_samples_leaf= 3 max_depth= 3	k=5 weights='distance'	batch_size=300
	half_nn	min_samples_leaf= 10 max_depth= 15	k=5 weights='uniform'	batch_size=250
	80_nn	min_samples_leaf= 10 max_depth= 20	k=5 weights='distance'	batch_size=200
	All features	min_samples_leaf= 26 max_depth= 26	k=5 weights='distance'	batch_size=200
Pageblocks	half_gb	min_samples_leaf= 8 max_depth=26	K= 5 weights='distance'	batch_size=200
	80_gb	min_samples_leaf=8 max_depth=30	k=5 weights='distance'	batch_size=200
	half_rf	min_samples_leaf=8 max_depth=14	k=5 weights='distance'	batch_size=300
	80_rf	min_samples_leaf=2 max_depth=6	k=5 weights='distance'	batch_sizes=200
	half_xtf	min_samples_leaf=3 max_depth=7	k=5 weights='distance'	batch_size=200
	80_xtf	min_samples_leaf=3 max_depth=13	k=5 weights='distance'	batch_size=200
	half_stk	min_samples_leaf=14 max_depth=24	k=10 weights='distance'	batch_size=200
	80_stk	min_samples_leaf=3 max_depth=18	k=5 weights='distance'	batch_size=250
	half_nn	min_samples_leaf=4 max_depth=29	k=10 weights='distance'	batch_size=200
	80_nn	min_samples_leaf=13 max_depth=17	k=5 weights='distance'	batch_size=200
	All features	min_samples_leaf=7 max_depth=18	k=5 weights='distance'	batch_size=200

Table 6.1: Best parameters for Cardio and Pageblocks datasets

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Dataset	Portion	Best: DT	Best: k-NN	Best: NN
Cardio	half_gb	min_samples_leaf= 11 max_depth= 24	K= 5 weights='uniform'	batch_size=300
	80_gb	min_samples_leaf= 4 max_depth= 9	k=5 weights='distance'	batch_size=250
	half_rf	min_samples_leaf= 9 max_depth= 30	k=5 weights='uniform'	batch_size=200
	80_rf	min_samples_leaf= 2 max_depth= 11	k=5 weights='uniform'	batch_sizes=200
	half_xtf	min_samples_leaf= 4 max_depth= 25	k=5 weights='distance'	batch_size=300
	80_xtf	min_samples_leaf= 11 max_depth= 8	k=5 weights='uniform'	batch_size=250
	half_stk	min_samples_leaf= 4 max_depth= 9	k=5 weights='distance'	batch_size=200
	80_stk	min_samples_leaf= 4 max_depth= 28	k=5 weights='uniform'	batch_size=200
	half_nn	min_samples_leaf= 1 max_depth= 26	k=5 weights='uniform'	batch_size=200
	80_nn	min_samples_leaf= 4 max_depth= 8	k=5 weights='uniform'	batch_size=200
	All features	min_samples_leaf= 7 max_depth= 28	k=5 weights='uniform'	batch_size=300
Letter	half_gb	min_samples_leaf= 3 max_depth=26	K= 5 weights='distance'	batch_size=200
	80_gb	min_samples_leaf=9 max_depth=12	k=5 weights='distance'	batch_size=250
	half_rf	min_samples_leaf=5 max_depth=25	k=5 weights='distance'	batch_size=250
	80_rf	min_samples_leaf=1 max_depth=26	k=5 weights='distance'	batch_sizes=200
	half_xtf	min_samples_leaf=1 max_depth=11	k=5 weights='distance'	batch_size=250
	80_xtf	min_samples_leaf=2 max_depth=26	k=5 weights='uniform'	batch_size=250
	half_stk	min_samples_leaf=12 max_depth=5	k=5 weights='distance'	batch_size=250
	80_stk	min_samples_leaf=3 max_depth=11	k=5 weights='uniform'	batch_size=250
	half_nn	min_samples_leaf=5 max_depth=25	k=5 weights='distance'	batch_size=200
	80_nn	min_samples_leaf=3 max_depth=5	k=5 weights='distance'	batch_size=250
	All features	min_samples_leaf=7 max_depth=9	k=5 weights='uniform'	batch_size=200

Table 6.2: Best parameters for Pendigit and Letter datasets

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Dataset	Portion	Best: DT	Best: k-NN	Best: NN
Vowels	half_gb	min_samples_leaf= 5 max_depth= 14	k= 10 weights='distance'	batch_size=300
	80_gb	min_samples_leaf= 6 max_depth= 21	k=10 weights='distance'	batch_size=300
	half_rf	min_samples_leaf= 7 max_depth= 19	k=5 weights='distance'	batch_size=300
	80_rf	min_samples_leaf= 25 max_depth= 19	k=5 weights='distance'	batch_sizes=300
	half_xtf	min_samples_leaf= 4 max_depth= 20	k=5 weights='distance'	batch_size=200
	80_xtf	min_samples_leaf= 7 max_depth= 7	k=5 weights='distance'	batch_size=250
	half_stk	min_samples_leaf= 3 max_depth= 19	k=5 weights='distance'	batch_size=200
	80_stk	min_samples_leaf= 2 max_depth= 3	k=5 weights='distance'	batch_size=250
	half_nn	min_samples_leaf= 10 max_depth= 19	k=5 weights='distance'	batch_size=250
	80_nn	min_samples_leaf= 5 max_depth= 28	k=5 weights='distance'	batch_size=250
	All features	min_samples_leaf= 5 max_depth= 14	k=5 weights='distance'	batch_size=300
Spambase	half_gb	min_samples_leaf= 2 max_depth=14	K= 15 weights='distance'	batch_size=200
	80_gb	min_samples_leaf=1 max_depth=28	k=5 weights='distance'	batch_size=250
	half_rf	min_samples_leaf=8 max_depth=30	k=15 weights='distance'	batch_size=250
	80_rf	min_samples_leaf=11 max_depth=16	k=10 weights='distance'	batch_sizes=300
	half_xtf	min_samples_leaf=7 max_depth=20	k=5 weights='distance'	batch_size=300
	80_xtf	min_samples_leaf=1 max_depth=13	k=15 weights='distance'	batch_size=250
	half_stk	min_samples_leaf=3 max_depth=26	k=5 weights='distance'	batch_size=200
	80_stk	min_samples_leaf=5 max_depth=18	k=5 weights='distance'	batch_size=300
	half_nn	min_samples_leaf=16 max_depth=16	k=5 weights='distance'	batch_size=250
	80_nn	min_samples_leaf=5 max_depth=18	k=10 weights='distance'	batch_size=300
	All features	min_samples_leaf=4 max_depth=10	k=5 weights='distance'	batch_size=200

Table 6.3: Best parameters for Vowels and Spambase datasets

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Dataset	Portion	Best: DT	Best: k-NN	Best: NN
Mnist	half_gb	min_samples_leaf= 1 max_depth= 18	k= 5 weights='distance'	batch_size=250
	80_gb	min_samples_leaf= 8 max_depth= 29	k=5 weights='distance'	batch_size=200
	half_rf	min_samples_leaf= 21 max_depth= 25	k=10 weights='distance'	batch_size=250
	80_rf	min_samples_leaf= 9 max_depth= 14	k=5 weights='uniform'	batch_sizes=250
	half_xtf	min_samples_leaf= 14 max_depth= 24	k=5 weights='uniform'	batch_size=300
	80_xtf	min_samples_leaf= 12 max_depth= 22	k=5 weights='uniform'	batch_size=300
	half_stk	min_samples_leaf= 11 max_depth= 16	k=5 weights='distance'	batch_size=300
	80_stk	min_samples_leaf= 7 max_depth= 21	k=5 weights='uniform'	batch_size=300
	half_nn	min_samples_leaf= 2 max_depth= 29	k=5 weights='uniform'	batch_size=300
	80_nn	min_samples_leaf= 4 max_depth= 21	k=5 weights='uniform'	batch_size=250
	All features	min_samples_leaf= 5 max_depth= 29	k=5 weights='uniform'	batch_size=250
Annthyroid	half_gb	min_samples_leaf= 4 max_depth=22	k= 5 weights='distance'	batch_size=200
	80_gb	min_samples_leaf=9 max_depth=23	k=5 weights='distance'	batch_size=300
	half_rf	min_samples_leaf=4 max_depth=3	k=5 weights='distance'	batch_size=250
	80_rf	min_samples_leaf=4 max_depth=19	k=5 weights='distance'	batch_sizes=200
	half_xtf	min_samples_leaf=2 max_depth=13	k=5 weights='distance'	batch_size=200
	80_xtf	min_samples_leaf=10 max_depth=23	k=5 weights='distance'	batch_size=200
	half_stk	min_samples_leaf=6 max_depth=20	k=5 weights='distance'	batch_size=200
	80_stk	min_samples_leaf=3 max_depth=19	k=5 weights='distance'	batch_size=200
	half_nn	min_samples_leaf=7 max_depth=28	k=5 weights='distance'	batch_size=250
	80_nn	min_samples_leaf=6 max_depth=1	k=20 weights='uniform'	batch_size=250
	All features	min_samples_leaf=11 max_depth=13	k=5 weights='distance'	batch_size=200

Table 6.4: Best parameters for Mnist and Annthyroid datasets

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Dataset	Portion	Best: DT	Best: k-NN	Best: NN
Satimage	half_gb	min_samples_leaf= 28 max_depth= 14	k= 5 weights='distance'	batch_size=200
	80_gb	min_samples_leaf= 2 max_depth= 12	k=5 weights='uniform'	batch_size=300
	half_rf	min_samples_leaf= 3 max_depth= 23	k=5 weights='distance'	batch_size=200
	80_rf	min_samples_leaf= 42 max_depth= 13	k=5 weights='distance'	batch_sizes=300
	half_xtf	min_samples_leaf= 33 max_depth= 24	k=5 weights='uniform'	batch_size=200
	80_xtf	min_samples_leaf= 23 max_depth= 28	k=5 weights='uniform'	batch_size=200
	half_stk	min_samples_leaf= 1 max_depth= 3	k=5 weights='distance'	batch_size=200
	80_stk	min_samples_leaf= 3 max_depth= 24	k=5 weights='uniform'	batch_size=250
	half_nn	min_samples_leaf= 2 max_depth= 16	k=5 weights='distance'	batch_size=200
	80_nn	min_samples_leaf= 12 max_depth= 18	k=5 weights='distance'	batch_size=300
	All features	min_samples_leaf= 20 max_depth= 21	k=5 weights='distance'	batch_size=250
Satellite	half_gb	min_samples_leaf= 28 max_depth=27	K= 10 weights='distance'	batch_size=300
	80_gb	min_samples_leaf=9 max_depth=15	k=5 weights='distance'	batch_size=300
	half_rf	min_samples_leaf=11 max_depth=22	k=5 weights='distance'	batch_size=250
	80_rf	min_samples_leaf=1 max_depth=10	k=5 weights='uniform'	batch_sizes=250
	half_xtf	min_samples_leaf=18 max_depth=23	k=5 weights='distance'	batch_size=250
	80_xtf	min_samples_leaf=14 max_depth=25	k=5 weights='uniform'	batch_size=250
	half_stk	min_samples_leaf=19 max_depth=26	k=5 weights='uniform'	batch_size=200
	80_stk	min_samples_leaf=18 max_depth=25	k=5 weights='distance'	batch_size=200
	half_nn	min_samples_leaf=16 max_depth=16	k=5 weights='uniform'	batch_size=200
	80_nn	min_samples_leaf=5 max_depth=27	k=5 weights='uniform'	batch_size=300
	All features	min_samples_leaf=3 max_depth=19	k=5 weights='uniform'	batch_size=250

Table 6.5: Best parameters for Satimage and Satellite datasets

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.2 Scoring metric values evaluated by k-NN

6.2.1 Evaluation in subsets of gradient boosting

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	GB	half	11	0.9412	0.9018	0.8963	0.9118	0.9518	0.9512
	GB	80	15	0.9307	0.9351	0.8792	0.8898	0.9057	0.9390
	ALL		21	0.8511	0.9091	0.7599	0.8440	0.8764	0.9255
Pageblocks	GB	half	5	0.7153	0.7914	0.5379	0.6463	0.8144	0.8577
	GB	80	7	0.7176	0.7643	0.5507	0.6123	0.8082	0.8330
	ALL		10	0.7111	0.7750	0.5347	0.6245	0.8083	0.8439
Pendigit	GB	half	8	0.9783	0.9813	0.9584	0.9641	0.9787	0.9817
	GB	80	12	0.9670	0.9860	0.9376	0.9731	0.9681	0.9862
	ALL		16	0.9670	0.9813	0.9376	0.9641	0.9681	0.9817
Vowels	GB	half	6	0.9286	0.8254	0.8712	0.6986	0.9333	0.8704
	GB	80	9	0.8889	0.8138	0.8069	0.6971	0.9000	0.8471
	ALL		12	0.8000	0.7500	0.6781	0.6137	0.8333	0.8000
Spambase	GB	half	29	0.8590	0.8818	0.7811	0.8338	0.8848	0.9000
	GB	80	40	0.8555	0.9030	0.7843	0.8556	0.8805	0.9190
	ALL		57	0.8466	0.9006	0.7793	0.8565	0.8722	0.9160
Satimage	GB	half	18	0.9500	0.9474	0.9059	0.9012	0.9524	0.9500
	GB	80	28	0.9268	0.9583	0.8607	0.9210	0.9521	0.9600
	ALL		36	0.9500	0.9583	0.9059	0.9219	0.9524	0.9600
Mnist	GB	half	50	0.9128	0.9256	0.8522	0.8709	0.9233	0.9369
	GB	80	69	0.9242	0.9242	0.8692	0.8692	0.9264	0.9350
	ALL		100	0.8992	0.9065	0.8303	0.8414	0.9136	0.9179
Satellite	GB	half	19	0.8371	0.8888	0.7757	0.8416	0.8700	0.9093
	GB	80	28	0.8405	0.9212	0.7654	0.8767	0.8789	0.9402
	ALL		37	0.8586	0.9241	0.7926	0.8821	0.8907	0.9416
UNSW	GB	half	19	0.8371	0.8888	0.7757	0.8416	0.8700	0.9093
	GB	80	27	0.8405	0.9212	0.7654	0.8767	0.8789	0.9402
	ALL		38	0.8586	0.9241	0.7926	0.8821	0.8907	0.9416
Cic2017	GB	half	19	0.9120	0.9221	0.8427	0.8530	0.9596	0.9596
	GB	80	28	0.9136	0.9223	0.8454	0.8609	0.9603	0.9623
	ALL		38	0.8099	0.8899	0.7215	0.8382	0.8732	0.9053
Botnet	GB	half	19	0.8339	0.9318	0.8412	0.9012	0.8775	0.9390
	GB	80	27	0.8699	0.9386	0.8432	0.9077	0.8844	0.9454
	ALL		38	0.8615	0.9525	0.8157	0.9293	0.8770	0.9575

Table 6.6: Performance scores for every split of the datasets determined by GB evaluated by k-NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.2.2 Evaluation in subsets of random forest

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	RF	half	11	0.8980	0.8869	0.8281	0.8383	0.9141	0.9106
	RF	80	15	0.9109	0.9442	0.8445	0.9045	0.9320	0.9472
	ALL		21	0.8511	0.9091	0.7599	0.8440	0.8764	0.9255
Page.	RF	half	5	0.7153	0.7914	0.5379	0.6463	0.8144	0.8577
	RF	80	7	0.7176	0.7643	0.5507	0.6123	0.8028	0.8330
	ALL		10	0.7111	0.7750	0.5347	0.6245	0.8083	0.8439
Pendigit	RF	half	8	0.9783	0.9954	0.9584	0.9910	0.9787	0.9954
	RF	80	12	0.9670	0.9860	0.9376	0.9731	0.9681	0.9862
	ALL		16	0.9670	0.9813	0.9376	0.9641	0.9681	0.9817
Vowels	RF	half	6	0.8889	0.8387	0.8069	0.7242	0.9000	0.8709
	RF	80	9	0.8889	0.8136	0.8069	0.6965	0.9000	0.8429
	ALL		12	0.8000	0.7500	0.6781	0.6137	0.8333	0.8000
Spambase	RF	half	29	0.8643	0.8633	0.7934	0.8093	0.8884	0.8848
	RF	80	40	0.8560	0.8745	0.7930	0.8332	0.8798	0.8926
	ALL		57	0.8466	0.9006	0.7793	0.8565	0.8722	0.9160
Satimage	RF	half	29	0.9268	0.9362	0.8607	0.8815	0.9521	0.9400
	RF	80	40	0.9500	0.9474	0.9059	0.9012	0.9524	0.9500
	ALL		57	0.9500	0.9583	0.9059	0.9210	0.9524	0.9600
Mnist	RF	half	51	0.8850	0.8960	0.7997	0.8245	0.9183	0.9123
	RF	80	74	0.9086	0.9336	0.8428	0.8839	0.9250	0.9441
	ALL		100	0.8992	0.9065	0.8303	0.8414	0.9136	0.9197
Satellite	RF	half	19	0.8560	0.9144	0.7916	0.8720	0.8882	0.9317
	RF	80	28	0.8429	0.9240	0.7714	0.8823	0.8793	0.9413
	ALL		37	0.8586	0.9241	0.7926	0.8821	0.8907	0.9416
UNSW	RF	half	19	0.7198	0.7775	0.5339	0.6177	0.8827	0.9180
	RF	80	21	0.7057	0.7577	0.5137	0.5873	0.8638	0.8938
	ALL		38	0.7192	0.7366	0.5323	0.5564	0.8746	0.8788
Cic2017	RF	half	19	0.8950	0.9221	0.8399	0.8607	0.8840	0.9620
	RF	80	24	0.8199	0.92665	0.7244	0.8675	0.8820	0.9661
	ALL		38	0.8099	0.8899	0.7215	0.8382	0.8732	0.9053
Botnet	RF	half	19	0.8212	0.9321	0.8279	0.9009	0.8061	0.9393
	RF	80	26	0.8038	0.9503	0.8308	0.9247	0.8098	0.9558
	ALL		38	0.8615	0.9525	0.8157	0.9293	0.8770	0.9575

Table 6.7: Performance scores for every split of the datasets determined by RF evaluated by k-NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.2.3 Evaluation in subsets of stacking

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	STK	half	11	0.9200	0.9304	0.8622	0.8824	0.9330	0.9350
	STK	80	15	0.8889	0.9217	0.8105	0.8670	0.9131	0.9305
	ALL		21	0.8511	0.9091	0.7599	0.8440	0.9131	0.9305
Page.	STK	half	5	0.6446	0.6897	0.4736	0.5253	0.7515	0.7777
	STK	80	7	0.7176	0.7643	0.5507	0.6123	0.8028	0.8330
	ALL		10	0.7111	0.7750	0.5347	0.6247	0.8083	0.8439
Pendigit	STK	half	8	0.9670	0.9954	0.9376	0.9910	0.9681	0.9954
	STK	80	11	0.9670	0.9907	0.9376	0.9821	0.9681	0.9908
	ALL		16	0.9670	0.9813	0.9376	0.9641	0.9681	0.9817
Vowels	STK	half	6	0.6957	0.5106	0.5494	0.3654	0.7988	0.6714
	STK	80	9	0.8000	0.7719	0.6781	0.6413	0.8333	0.8143
	ALL		12	0.8000	0.7500	0.6781	0.6137	0.8333	0.8000
Spambase	STK	half	29	0.8638	0.9016	0.794	0.8573	0.8878	0.9170
	STK	80	41	0.8491	0.9049	0.7750	0.8608	0.8752	0.9200
	ALL		57	0.8466	0.9006	0.7793	0.8565	0.8722	0.9160
Satimage	STK	half	18	0.9268	0.9583	0.8607	0.9210	0.9521	0.9600
	STK	80	27	0.9268	0.9583	0.8607	0.9210	0.9521	0.9600
	ALL		36	0.9500	0.9583	0.9059	0.9210	0.9524	0.9600
Mnist	STK	half	51	0.8471	0.8882	0.7376	0.8056	0.9335	0.9428
	STK	80	74	0.9023	0.9231	0.8295	0.8674	0.9350	0.9340
e	ALL		100	0.8992	0.9056	0.8303	0.8414	0.9136	0.9197
Satellite	STK	half	19	0.8420	0.9061	0.7747	0.8588	0.8765	0.9260
satellite	STK	80	28	0.8742	0.9190	0.8141	0.8736	0.9021	0.9384
	ALL		37	0.8586	0.9241	0.7926	0.8821	0.8907	0.9416
UNSW	STK	half	11	0.7356	0.7365	0.5558	0.5451	0.8895	0.8001
	STK	80	14	0.7370	0.7766	0.5578	0.6147	0.8907	0.9010
	ALL		21	0.7192	0.7366	0.5323	0.5564	0.8746	0.8788
CIC2017	STK	half	19	0.8270	0.8952	0.7250	0.8440	0.9124	0.9105
	STK	80	24	0.8706	0.9010	0.7721	0.8531	0.9514	0.9140
	ALL		381	0.8099	0.8899	0.7215	0.8382	0.8732	0.9053
Botnet	STK	half	19	0.8035	0.9109	0.8104	0.8965	0.8077	0.9287
	STK	80	27	0.8211	0.9392	0.8288	0.9011	0.8665	0.9391
	ALL		38	0.8615	0.9525	0.8157	0.9293	0.8770	0.9575

Table 6.8: Performance scores for every split of the datasets determined by STK evaluated by k-NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.2.4 Evaluation in subsets of neural networks

Dataset	alg.	split	feat.	f1_ts	f1_tr	avg_ts	avg_tr	roc_ts	roc_tr
Cardio	NN	half	11	0.8866	0.9356	0.8105	0.8891	0.9047	0.9427
	NN	80	9	0.8958	0.9367	0.8295	0.8881	0.9424	0.9499
	ALL		21	0.8511	0.9091	0.7599	0.8440	0.8764	0.9255
Page.	NN	half	5	0.6880	0.6996	0.5223	0.5351	0.7775	0.7807
	NN	80	4	0.7121	0.7938	0.5411	0.6536	0.8025	0.8527
	ALL		10	0.7111	0.7750	0.5347	0.6245	0.8083	0.8439
Pendigit	NN	half	8	0.9670	0.9954	0.9376	0.9910	0.9681	0.9954
	NN	80	4	0.9451	0.9671	0.8960	0.9371	0.9572	0.9724
	ALL		16	0.9670	0.9813	0.9376	0.9641	0.9681	0.9817
Vowels	NN	half	6	0.7200	0.7719	0.5537	0.6413	0.7988	0.8143
	NN	80	5	0.7200	0.7037	0.5537	0.5586	0.7988	0.7714
	ALL		12	0.8000	0.7500	0.6781	0.6137	0.8333	0.8000
Spambase	NN	half	29	0.8382	0.9003	0.7577	0.8555	0.8663	0.9158
	NN	80	27	0.8453	0.9619	0.7740	0.8164	0.8716	0.8825
	ALL		57	0.8466	0.9006	0.7793	0.8565	0.8722	0.9160
Satimage	NN	half	18	0.9048	0.9583	0.8197	0.9210	0.9518	0.9600
	NN	80	14	0.9268	0.9375	0.8607	0.8817	0.9521	0.9499
	ALL		36	0.9500	0.9583	0.9059	0.9210	0.9524	0.9600
Mnist	NN	half	51	0.9104	0.9305	0.8424	0.8781	0.8976	0.9186
	NN	80	17	0.8700	0.8870	0.7746	0.8038	0.9104	0.9176
	ALL		100	0.8992	0.9065	0.8303	0.8414	0.9136	0.9197
Satellite	NN	half	19	0.8556	0.9112	0.7985	0.8639	0.8840	0.91315
	NN	80	22	0.8543	0.9140	0.7929	0.8672	0.8850	0.9339
	ALL		37	0.8586	0.9241	0.7926	0.8821	0.8907	0.9416
UNSW	NN	half	19	0.7451	0.8027	0.5717	0.6638	0.9103	0.9392
	NN	80	12	0.7366	0.7717	0.5257	0.6443	0.7777	0.8002
	ALL		38	0.7192	0.7366	0.5323	0.5564	0.8746	0.8788
Cic2017	NN	half	19	0.8168	0.9044	0.7249	0.8579	0.8357	0.9166
	NN	80	12	0.8933	0.8918	0.8388	0.8401	0.9106	0.9073
	ALL		38	0.8099	0.8899	0.7215	0.8382	0.8732	0.9053
Botnet	NN	half	19	0.8220	0.9187	0.8295	0.8647	0.8811	0.9294
	NN	80	10	0.8027	0.9200	0.8001	0.8972	0.8222	0.9274
	ALL		38	0.8615	0.9525	0.8157	0.9293	0.8770	0.9575

Table 6.9: Performance scores for every split of the datasets determined by NN evaluated by k-NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.3 Scoring metrics values evaluated by NN

6.3.1 Evaluation in subsets of gradient boosting

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	GB	half	11	0.9515	0.9752	0.9133	0.9552	0.9613	0.9792
	GB	80	15	0.9515	0.9793	0.9133	0.9633	0.9434	0.9797
	ALL		21	0.9400	1	0.8977	1	0.9613	1
Pageblocks	GB	half	5	0.7714	0.8480	0.6160	0.7301	0.8476	0.9024
	GB	80	7	0.7871	0.8511	0.6299	0.7313	0.8903	0.9415
	ALL		10	0.7832	0.8530	0.6307	0.7370	0.8602	0.9105
Pendigit	GB	half	8	0.9783	1	0.9584	1	0.9787	1
	GB	80	12	0.9783	1	0.9584	1	0.9787	1
	ALL		16	0.9783	1	0.9584	1	0.9787	1
Vowels	GB	half	6	0.8889	0.9394	0.8069	0.8896	0.9000	0.9429
	GB	80	9	0.9286	0.9412	0.8712	0.8895	0.9333	0.9566
	ALL		12	0.8667	0.9589	0.7557	0.9211	0.9310	0.9985
Spambase	GB	half	29	0.8966	0.9532	0.8275	0.9275	0.9185	0.9609
	GB	80	40	0.9151	0.9756	0.8617	0.9634	0.9320	0.9790
	ALL		57	0.9065	0.9855	0.8492	0.9783	0.9248	0.9874
Satimage	GB	half	18	0.9500	1	0.9059	1	0.9524	1
	GB	80	28	0.8837	0.9901	0.7825	0.9804	0.9515	0.9999
	ALL		36	0.8636	0.9901	0.7486	0.9804	0.9512	0.9999
Mnist	GB	half	51	0.9515	0.9949	0.9117	0.9904	0.9652	0.9967
	GB	80	69	0.9476	0.9919	0.9028	0.9848	0.9607	0.9983
	ALL		100	0.9476	0.9949	0.9028	0.9904	0.9712	0.9967
Satellite	GB	half	19	0.8482	0.9014	0.7734	0.8421	0.8857	0.9290
	GB	80	28	0.8170	0.8963	0.7308	0.8336	0.8636	0.9260
	ALL		37	0.8414	0.8929	0.7780	0.8448	0.8745	0.9136
UNSW	GB	half	19	0.7355	0.7803	0.5647	0.6335	0.9320	0.9720
	GB	80	25	0.7616	0.7873	0.6039	0.6438	0.9310	0.9339
	ALL		38	0.7280	0.7692	0.5502	0.6091	0.9507	0.9691
Cic2017	GB	half	19	0.8920	0.8972	0.8395	0.8464	0.110	0.9129
	GB	80	28	0.8957	0.8955	0.8416	0.8430	0.9131	0.9119
	ALL		38	0.8912	0.8899	0.8972	0.8453	0.9108	0.9132
Botnet	GB	half	19	0.7941	0.8831	0.7776	0.8440	0.8052	0.8959
	GB	80	27	0.7942	0.8919	0.7691	0.8590	0.7761	0.9034
	ALL		38	0.7977	0.8635	0.7689	0.8289	0.7448	0.8218

Table 6.10: Performance scores for every split of the datasets determined by GB evaluated by NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.3.2 Evaluation in subsets of random forest

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	RF	half	11	0.8785	0.9707	0.7827	0.9486	0.9364	0.9715
	RF	80	15	0.9412	0.9919	0.8963	0.9846	0.9518	0.9955
	ALL		21	0.9400	1	0.8977	1	0.9613	1
Page.	RF	half	5	0.7704	0.8408	0.5280	0.7219	0.8356	0.8890
	RF	80	7	0.7536	0.8688	0.5919	0.7646	0.8346	0.9140
	ALL		10	0.7832	0.8530	0.6307	0.7370	0.8602	0.9105
Pendigit	RF	half	8	0.9783	1	0.9584	1	0.9787	1
	RF	80	12	0.9670	1	0.9376	1	0.9681	1
	ALL		16	0.9783	1	0.9584	1	0.9787	1
Vowels	RF	half	6	0.8889	0.8955	0.8069	0.8085	0.0.9000	0.9276
	RF	80	9	0.8571	1	0.7453	0.1	0.8988	1
	ALL		12	0.8667	0.9589	0.7557	0.9211	0.9310	0.9985
Spambase	RF	half	29	0.9151	0.9742	0.8617	0.9641	0.9281	0.9776
	RF	80	40	0.9100	0.9821	0.8487	0.9733	0.9294	0.9844
	ALL		57	0.9065	0.9855	0.8492	0.9783	0.9248	0.9874
Satimage	RF	half	29	0.9268	1	0.8607	1	0.9521	1
	RF	80	40	0.9268	0.9796	0.8607	0.9605	0.9521	0.9800
	ALL		57	0.9636	0.9901	0.7486	0.9804	0.9521	0.9999
Mnist	RF	half	51	0.9314	0.9969	0.8733	0.9939	0.9652	0.9997
	RF	80	74	0.9353	0.9929	0.8813	0.9862	0.9712	0.9937
	ALL		100	0.9476	0.9949	0.9028	0.9904	0.9712	0.9967
Satellite	RF	half	19	0.8333	0.8923	0.7595	0.8344	0.8720	0.9185
	RF	80	28	0.8307	0.8880	0.7379	0.8161	0.8711	0.9209
	ALL		37	0.8414	0.8929	0.7780	0.8448	0.8745	0.9136
UNSW	RF	half	11	0.7341	0.7830	0.5622	0.6383	0.9297	0.9699
	RF	80	21	0.7291	0.7830	0.5331	0.6379	0.9388	0.9687
	ALL		38	0.7280	0.7692	0.5502	0.6091	0.9507	0.9121
Cic2017	RF	half	19	0.8919	0.8971	0.8355	0.8457	0.9079	0.9163
	RF	80	24	0.8889	0.9020	0.8345	0.8528	0.9056	0.9132
	ALL		38	0.8912	0.8972	0.8340	0.8553	0.9108	0.9667
Botnet	RF	half	19	0.7549	0.8952	0.8152	0.8594	0.7453	0.9064
	RF	80	26	0.7583	0.8909	0.7638	0.8510	0.7790	0.9027
	ALL		38	0.7977	0.8635	0.7689	0.8289	0.7448	0.8218

Table 6.11: Performance scores for every split of the datasets determined by RF evaluated by NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.3.3 Evaluation in subsets of stacking

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	STK	half	11	0.9200	0.9919	0.8622	0.9846	0.9330	0.9955
	STK	80	15	0.9307	0.9877	0.8792	0.9780	0.9424	0.9878
	ALL		21	0.9400	1	0.8977	1	0.9613	1
Page.	STK	half	5	0.7671	0.8563	0.6039	0.7409	0.8592	0.9211
	STK	80	7	0.7391	0.8504	0.5706	0.7345	0.8278	0.9025
	ALL		10	0.7832	0.8530	0.6307	0.7370	0.8602	0.9105
Pendigit	STK	half	8	0.9783	1	0.9584	1	0.9787	1
	STK	80	11	0.9556	1	0.9168	1	0.9574	1
	ALL		16	0.9783	1	0.9584	1	0.9787	1
Vowels	STK	half	6	0.8462	0.9429	0.7424	0.8909	0.8667	0.9704
	STK	80	9	0.7742	0.9167	0.6069	0.8429	0.8953	0.9694
	ALL		12	0.8667	0.9589	0.7557	0.9211	0.9310	0.9985
Spambase	STK	half	29	0.9122	0.9718	0.8539	0.9573	0.9307	0.9759
	STK	80	41	0.9012	0.9668	0.8342	0.9431	0.9225	0.9742
	ALL		57	0.9056	0.9855	0.8492	0.9783	0.9248	0.9874
Satimage	STK	half	18	0.8636	0.9608	0.7586	0.9237	0.9512	0.9896
	STK	80	27	0.8837	0.9899	0.7825	0.9802	0.9515	0.9900
	ALL		36	0.8636	0.9901	0.7486	0.9804	0.9212	0.9999
Mnist	STK	half	51	0.9021	0.9918	0.8556	0.9848	0.9492	0.9937
	STK	80	74	0.9286	0.9918	0.8688	0.9841	0.9614	0.9974
e	ALL		100	0.9476	0.9949	0.9028	0.9904	0.9712	0.9967
Satellite	STK	half	19	0.8199	0.8695	0.7575	0.8166	0.8565	0.8944
	STK	80	28	0.8349	0.9004	0.7662	0.8161	0.8796	0.9237
	ALL		37	0.8414	0.8929	0.7780	0.8448	0.8745	0.9136
UNSW	STK	half	19	0.7371	0.7558	0.5579	0.5841	0.8901	0.8920
	STK	80	26	0.7505	0.7748	0.5801	0.6153	0.9161	0.9267
	ALL		38	0.7280	0.7692	0.5502	0.6091	0.9507	0.9691
CIC2017	STK	half	19	0.8615	0.8987	0.7984	0.8488	0.8845	0.9133
	STK	80	24	0.8970	0.9027	0.8435	0.8535	0.9140	0.9172
	ALL		38	0.8912	0.8972	0.8340	0.8453	0.9108	0.9132
Botnet	STK	half	19	0.7404	0.8701	0.7106	0.8410	0.7628	0.8878
	STK	80	27	0.7553	0.8899	0.7556	0.8490	0.7703	0.9003
	ALL		38	0.7977	0.8635	0.7689	0.8289	0.7448	0.8218

Table 6.12: Performance scores for every split of the datasets determined by STK evaluated by NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.3.4 Evaluation in subsets of neural networks

Dataset	alg.	split	feat.	f1_ts	f1_tr	avg_ts	avg_tr	roc_ts	roc_tr
Cardio	NN	half	11	0.9515	0.9877	0.9133	0.9780	0.9613	0.9878
	NN	80	9	0.9307	0.9580	0.8792	0.9258	9424	0.9630
	ALL		21	0.9400	1	0.8977	1	0.9613	1
Page.	NN	half	5	0.7972	0.8661	0.6523	0.7579	0.8671	0.9207
	NN	80	4	0.7785	0.8287	0.6191	0.6949	0.8718	0.9142
	ALL		10	0.7832	0.8530	0.6307	0.7370	0.8602	0.9105
Pendigit	NN	half	8	0.9783	1	0.9584	1	0.9787	1
	NN	80	4	0.9451	0.9907	0.8960	0.9821	0.9572	0.9908
	ALL		16	0.9783	1	0.9584	1	0.9787	1
Vowels	NN	half	6	0.8889	0.9552	0.8069	0.9172	0.0.9000	0.9571
	NN	80	5	0.7857	0.8831	0.6297	0.7874	0.8643	0.9816
	ALL		12	0.8667	0.9585	0.7557	0.9211	0.9310	0.9985
Spambase	NN	half	29	0.9077	0.9728	0.8435	0.9566	0.9320	0.9768
	NN	80	27	0.9109	0.9697	0.8570	0.9586	0.9281	0.9726
	ALL		57	0.9065	0.9855	0.8492	0.9783	0.9248	0.9874
Satimage	NN	half	18	0.8261	0.9800	0.6888	0.9606	0.9506	0.9899
	NN	80	14	0.8837	0.9800	0.7825	0.9606	0.9515	0.9899
	ALL		36	0.8636	0.9901	0.7486	0.9804	0.9512	0.9999
Mnist	NN	half	51	0.9563	0.9939	0.9206	0.9885	0.9678	0.9957
	NN	80	17	0.8926	0.9816	0.8068	0.9654	0.9399	0.9890
	ALL		100	0.9476	0.9949	0.9028	0.9904	0.9712	0.9967
Satellite	NN	half	19	0.8403	0.8883	0.7760	0.8267	0.8739	0.9169
	NN	80	22	0.8105	0.8974	0.7349	0.8364	0.8536	0.9259
	ALL		37	0.8414	0.8929	0.7780	0.8448	0.8745	0.9136
UNSW	NN	half	11	0.7469	0.7638	0.5729	0.5968	0.9029	0.9074
	NN	80	12	0.7270	0.7747	0.5233	0.5520	0.9100	0.9233
	ALL		21	0.7280	0.7692	0.5502	0.6091	0.9507	0.9691
Cic2017	NN	half	11	0.8943	0.8968	0.8366	0.8445	0.9148	0.9131
	NN	80	12	0.8937	0.8897	0.8397	0.8365	0.9107	0.9061
	ALL		21	0.8912	0.8972	0.8340	0.8453	0.9108	0.9132
Botnet	NN	half	19	0.7716	0.8981	0.7461	0.8598	0.7788	0.9090
	NN	80	10	0.7379	0.8868	0.7189	0.8450	0.7476	0.8992
	ALL		38	0.7977	0.8635	0.7689	0.8289	0.7448	0.8218

Table 6.13: Performance scores for every split of the datasets determined by NN evaluated by NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.4 Scoring metrics values evaluated by DT

6.4.1 Evaluation in subsets of gradient boosting

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	GB	half	11	0.8298	0.9840	0.7254	0.9685	0.8954	0.9823
	GB	80	15	0.7777	0.9954	0.7073	0.9721	0.8933	0.9806
	ALL		21	0.7350	0.7855	0.5633	0.6427	0.8845	0.9574
Pageblocks	GB	half	5	0.7191	0.8504	0.5351	0.7379	0.9255	0.9826
	GB	80	7	0.7342	0.8972	0.5517	0.8136	0.9026	0.9845
	ALL		10	0.7345	0.7975	0.5564	0.6605	0.9101	0.9865
Pendigit	GB	half	8	0.9091	1	0.8337	1	0.9856	0.9973
	GB	80	12	0.9247	0.9864	0.8572	0.9732	0.9666	0.9991
	ALL		16	0.8654	0.9083	0.7568	0.8321	0.9757	0.9977
Vowels	GB	half	6	0.7407	0.8861	0.5670	0.7955	0.8917	0.9893
	GB	80	9	0.8387	0.8333	0.7087	0.7143	0.9298	0.9929
	ALL		12	0.6154	0.7447	0.4069	0.5932	0.8858	0.9878
Spambase	GB	half	29	0.8628	0.9631	0.7821	0.9366	0.8888	0.9714
	GB	80	40	0.8481	0.9991	0.7779	0.9986	0.8739	0.9993
	ALL		57	0.8734	0.9404	0.8046	0.9055	0.8964	0.9513
Satimage	GB	half	18	0.8837	0.9804	0.7825	0.9615	0.9515	0.9998
	GB	80	28	0.7660	0.9804	0.5951	0.9615	0.9262	0.9998
	ALL		36	0.5676	0.7246	0.4432	0.5682	0.9227	0.9953
Mnist	GB	half	51	0.8491	1	0.7341	1	0.9447	1
	GB	80	69	0.7991	0.9057	0.6547	0.8277	0.9344	0.9819
	ALL		100	0.7992	0.8950	0.6565	0.8099	0.9362	0.9881
Satellite	GB	half	19	0.7289	0.8537	0.6351	0.7648	0.7971	0.8999
	GB	80	28	0.6966	0.8983	0.5849	0.8261	0.7774	0.9359
	ALL		37	0.6999	0.9508	0.5620	0.9072	0.7871	0.9750
UNSW	GB	half	19	0.8900	0.9753	0.6997	0.7465	0.9167	0.9840
	GB	80	25	0.8817	0.9855	0.6848	0.7467	0.9145	0.9838
	ALL		38	0.8071	0.9683	0.6284	0.7383	0.9015	0.9831
Cic2017	GB	half	19	0.8858	0.9227	0.8447	0.8603	0.8246	0.9660
	GB	80	28	0.8702	0.9266	0.7983	0.8671	0.7542	0.9647
	ALL		38	0.8580	0.9662	0.7413	0.8639	0.7547	0.9667
Botnet	GB	half	19	0.8297	0.9419	0.8774	0.9361	0.8550	0.9648
	GB	80	27	0.8164	0.9598	0.8690	0.9307	0.8208	0.9621
	ALL		38	0.7567	0.9466	0.8584	0.9070	0.8057	0.9465

Table 6.14: Performance scores for every split of the datasets determined by GB evaluated by NN classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.4.2 Evaluation in subsets of random forest

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	RF	half	11	0.7087	0.8571	0.5309	0.7500	0.8659	0.9983
	RF	80	15	0.8132	1	0.7088	0.9721	0.8481	1
	ALL		21	0.7350	0.7855	0.5633	0.6427	0.8845	0.9574
Page.	RF	half	5	0.7083	0.7505	0.5280	0.6007	0.9255	0.9826
	RF	80	7	0.7151	0.7716	0.5299	0.6281	0.9085	0.9819
	ALL		10	0.7345	0.7956	0.5565	0.6605	0.9101	0.9865
Pendigit	RF	half	8	0.8350	0.8516	0.7044	0.7415	0.9542	0.9960
	RF	80	12	0.9184	0.9646	0.8458	0.9316	0.9567	0.9997
	ALL		16	0.8654	0.9083	0.7568	0.8321	0.9757	0.9977
Vowels	RF	half	6	0.7059	0.7609	0.5167	0.6140	0.8596	0.9888
	RF	80	9	0.5308	0.5828	0.3236	0.3481	0.9240	0.9616
	ALL		12	0.6154	0.7447	0.4069	0.5932	0.8858	0.9878
Spambase	RF	half	29	0.8822	0.9222	0.8161	0.8771	0.9040	0.9366
	RF	80	40	0.8647	0.9037	0.7846	0.8437	0.8905	0.9229
	ALL		57	0.8734	0.9404	0.8046	0.9055	0.8964	0.9513
Satimage	RF	half	29	0.8261	0.9901	0.9804	0.8815	0.9501	0.9995
	RF	80	40	0.8837	0.9524	0.7825	0.9091	0.9195	0.9894
	ALL		57	0.5676	0.7246	0.4432	0.5682	0.9227	0.9953
Mnist	RF	half	51	0.7156	0.7669	0.5471	0.6197	0.9305	0.9623
	RF	80	74	0.7689	0.8689	0.6129	0.7676	0.9256	0.9820
	ALL		100	0.7992	0.8950	0.6565	0.8099	0.9362	0.9881
Satellite	RF	half	19	0.7392	0.8832	0.6363	0.8040	0.8070	0.9244
	RF	80	28	0.7279	0.8641	0.6124	0.7745	0.8022	0.9117
	ALL		37	0.6999	0.9508	0.5620	0.9072	0.7871	0.9750
UNSW	RF	half	19	0.8417	0.9504	0.6637	0.7399	0.9142	0.9835
	RF	80	21	0.8208	0.9212	0.6642	0.7546	0.9270	0.9846
	ALL		38	0.8071	0.9683	0.6284	0.7383	0.9015	0.9831
Cic2017	RF	half	24	0.8628	0.9305	0.7796	0.8731	0.7643	0.9710
	RF	80	14	0.8543	0.9347	0.7335	0.8803	0.7054	0.9731
	ALL		38	0.8580	0.9662	0.7413	0.8639	0.7547	0.9667
Botnet	RF	half	19	0.8091	0.9567	0.8152	0.8819	0.8125	0.9413
	RF	80	26	0.8143	0.9514	0.8667	0.9235	0.8105	0.9352
	ALL		38	0.7567	0.9466	0.8584	0.9070	0.8057	0.9465

Table 6.15: Performance scores for every split of the datasets determined by RF evaluated by DT classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.4.3 Evaluation in subsets of stacking

Dataset	alg.	split	Feat.	F1_ts	F1_tr	Avg_ts	Avg_tr	roc_ts	roc_tr
Cardio	STK	half	11	0.7069	0.8328	0.5253	0.7126	0.8659	0.9752
	STK	80	15	0.6432	0.7766	0.4515	0.6239	0.8497	0.9251
	ALL		21	0.7350	0.7855	0.5633	0.6427	0.8845	0.9574
Page.	STK	half	5	0.5991	0.6925	0.5351	0.5296	0.9030	0.9767
	STK	80	7	0.6564	0.7427	0.4592	0.5908	0.8688	0.9940
	ALL		10	0.7345	0.7956	0.5565	0.6605	0.9101	0.9865
Pendigit	STK	half	8	0.8687	0.9277	0.7585	0.8651	0.9552	0.9982
	STK	80	11	0.9072	0.9646	0.8253	0.9316	0.9772	0.9991
	ALL		16	0.8654	0.9083	0.7568	0.8321	0.9757	0.9977
Vowels	STK	half	6	0.6452	0.8642	0.4281	0.7609	0.8262	0.9944
	STK	80	9	0.4963	0.5000	0.3023	0.3482	0.8360	0.9344
	ALL		12	0.6154	0.7447	0.4069	0.5932	0.8858	0.9878
Spambase	STK	half	29	0.8689	0.9671	0.8026	0.9455	0.8917	0.9737
	STK	80	41	0.8566	0.9373	0.7662	0.8905	0.8849	0.9522
	ALL		57	0.8734	0.9404	0.8046	0.9055	0.8964	0.9513
Satimage	STK	half	18	0.7917	0.9615	0.6378	0.9259	0.9506	0.9999
	STK	80	27	0.5484	0.5405	0.4314	0.3904	0.9515	0.9994
	ALL		36	0.5676	0.7246	0.4432	0.5628	0.9227	0.9953
Mnist	STK	half	51	0.8042	0.8602	0.6644	0.7534	0.9168	0.9773
	STK	80	74	0.7600	0.8589	0.6015	0.7527	0.9282	0.9833
e	ALL		100	0.7992	0.8950	0.6565	0.8099	0.9362	0.9881
Satellite	STK	half	19	0.7902	0.8514	0.6891	0.7594	0.8481	0.8998
	STK	80	28	0.7757	0.9302	0.6795	0.8753	0.8351	0.9593
	ALL		37	0.6999	0.9508	0.5620	0.9072	0.7871	0.975
UNSW	STK	half	19	0.8213	0.9331	0.6645	0.7298	0.9073	0.9818
	STK	80	26	0.8210	0.8837	0.6634	0.7441	0.9112	0.9834
	ALL		38	0.8071	0.9683	0.6284	0.7383	0.9015	0.9831
CIC2017	STK	half	19	0.8376	0.9246	0.7308	0.8634	0.7052	0.9667
	STK	80	24	0.8237	0.9287	0.7114	0.8702	0.7041	0.9692
	ALL		38	0.8580	0.9662	0.7413	0.8639	0.7547	0.9667
Botnet	STK	half	19	0.8090	0.9116	0.8057	0.8722	0.8107	0.9358
	STK	80	27	0.8102	0.9215	0.8162	0.9078	0.8047	0.9013
	ALL		38	0.7567	0.9466	0.8584	0.9070	0.8057	0.9465

Table 6.16: Performance scores for every split of the datasets determined by STK evaluated by DT classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.4.4 Evaluation in subsets of neural networks

Dataset	alg.	split	feat.	f1_ts	f1_tr	avg_ts	avg_tr	roc_ts	roc_tr
Cardio	NN	half	11	0.6825	0.8227	0.4961	0.6989	0.8755	0.9771
	NN	80	9	0.7414	0.8227	0.5719	0.6989	0.8855	0.9771
	ALL		21	0.7350	0.7855	0.5633	0.6427	0.8845	0.9574
Page.	NN	half	5	0.7083	0.7505	0.5280	0.6007	0.9136	0.9908
	NN	80	4	0.5472	0.5917	0.4010	0.6536	0.9085	0.9639
	ALL		10	0.7345	0.7956	0.5565	0.6605	0.9101	0.8965
Pendigit	NN	half	8	0.8519	0.8971	0.7385	0.8134	0.9253	1
	NN	80	4	0.8866	0.9121	0.7887	0.8385	0.9557	0.9978
	ALL		16	0.8654	0.9083	0.7568	0.8321	0.9757	0.9977
Vowels	NN	half	6	0.5714	0.5932	0.3624	0.4217	0.8822	0.9756
	NN	80	5	0.6111	0.7292	0.3933	0.5738	0.8548	0.9868
	ALL		12	0.6154	0.7447	0.4069	0.5932	0.8858	0.9878
Spambase	NN	half	29	0.8310	0.9015	0.7367	0.8468	0.8160	0.9193
	NN	80	27	0.8629	0.9350	0.7763	0.8903	0.8902	0.9492
	ALL		57	0.8734	0.9404	0.8046	0.9055	0.8964	0.9513
Satimage	NN	half	18	0.8372	0.9709	0.7030	0.9434	0.9274	0.9996
	NN	80	14	0.6667	0.7634	0.4787	0.6173	0.9474	0.9961
	ALL		36	0.5675	0.7264	0.4432	0.5682	0.9227	0.9953
Mnist	NN	half	51	0.8364	0.9780	0.7135	0.9570	0.9409	0.9977
	NN	80	17	0.7742	0.8485	0.6217	0.7368	0.9160	0.9894
	ALL		100	0.7992	0.8950	0.6565	0.8099	0.9362	0.9881
Satellite	NN	half	19	0.7664	0.8684	0.6748	0.9072	0.8229	0.9123
	NN	80	22	0.7427	0.9139	0.6215	0.8433	0.8163	0.9542
	ALL		37	0.6999	0.9508	0.5620	0.9072	0.7871	0.9740
UNSW	NN	half	11	0.8502	0.8968	0.6639	0.7485	0.9279	0.9839
	NN	80	12 0.8437	0.8951	0.6637	0.7591	0.9009	0.8006	
	ALL		38	0.8071	0.9683	0.6284	0.7383	0.9015	0.9831
Cic2017	NN	half	19	0.8564	0.8938	0.7573	0.8088	0.7527	0.9642
	NN	80	12	0.8277	0.8260	0.7269	0.8037	0.7907	0.9397
	ALL		38	0.8580	0.9662	0.7413	0.8639	0.7547	0.9667
Botnet	NN	half	19	0.8119	0.9254	0.8261	0.9104	0.8175	0.9487
	NN	80	10	0.8028	0.9315	0.8018	0.8719	0.8147	0.9573
	ALL		38	0.7567	0.9466	0.8584	0.9070	0.8057	0.9465

Table 6.17: Performance scores for every split of the datasets determined by NN evaluated by DT classifier.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

List of Figures

2.1 The basic scheme of the supervised machine learning approach. 8

2.2 The logistic function (Source: *wikipedia.org/wiki/Logistic_regression*). . 11

2.3 An example of SVM data separation in a 2D plane. 12

2.4 An example of SVM data separation for a non-linear case. 13

2.5 Transformation by polynomial kernel. 14

2.6 An example of k-NN classification process (Source: *wikipedia.org/wiki/K − nearest_neighbors_algorithm*). 15

2.7 An example of building a DT for classification of two shapes, circle or star. 18

2.8 Illustration of the bagging technique. 20

2.9 Illustration of the boosting technique. 21

2.10 Illustration of the stacking technique. 24

2.11 "A functional description of a biological neuron's structure" - Buduma and Locascio 2017 [BL17]. 25

2.12 The schematic representation of an artificial one-neuron network. 26

2.13 Basic neural network structure. 27

3.1 Framework of the conducted experiments. 33

3.2 Stability selection process. 41

3.3 The structure of a confusion matrix for binary classification. 52

4.1 Feature scores for Cardio dataset ranked by different algorithms. 57

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

4.2 The hiearchical clustering and the derived spearman correlation between features for the dataset *Cardio*. The more yellow the color, the stronger the correlation. 59

4.3 Feature scores for CIC2017 dataset ranked by different algorithms. 60

4.4 The hiearchical clustering and the derived spearman correlation between features for *CIC2017* dataset. 61

4.5 ROC-AUC scores evaluated by DT, k-NN and NN when using half of the most important features. 63

4.6 F1 and average-precision scores evaluated by DT, KNN and NN when using half of the most important features. 64

4.7 Critical Difference (CD) diagrams showing the results of a statistical comparison of the performance of every feature selection method on subsets with only half of the initial features. The bold lines connects approaches are not statistically different. The numbers show the average ranks. The lower the rank, the better the performance of the approach. 65

4.8 F1 and Average-precision scores evaluated by DT, k-NN and NN for features of 80% of total importance. 67

4.9 ROC-AUC scores evaluated by DT, k-NN and NN for features of 80% of total importance. 68

4.10 Critical Difference (CD) diagrams showing the results of a statistical comparison of the performance of every feature selection method on subsets of features with the relevance of 80% of importance. 69

4.11 Performance scores on the two splits of the datases resulted from the specific feature selection method and on the intial dataset. 70

4.12 Comparison of the k-NN performance scores of SS-GB for selecting half or features of 80% of importance. Note that a *negative* performance degradation actually means a *performance improvement* caused by reducing the number of features. 72

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

List of Tables

3.1 List of outlier detection datasets that are used in the experiments and their corresponding details [Ray16, CZS+16]. 36
3.2 List of IDS datasets used in the experiments and their corresponding details [SLG18, MS15, BBHJSG14]. 37
3.3 NN implementation for classification task. 49

4.1 Summary of the conducted experiments. 56
4.2 Number of features selected by every approach. 62

6.1 Best parameters for Cardio and Pageblocks datasets 78
6.2 Best parameters for Pendigit and Letter datasets 79
6.3 Best parameters for Vowels and Spambase datasets 80
6.4 Best parameters for Mnist and Annthyroid datasets 81
6.5 Best parameters for Satimage and Satellite datasets 82
6.6 Performance scores for every split of the datasets determined by GB evaluated by k-NN classifier. 83
6.7 Performance scores for every split of the datasets determined by RF evaluated by k-NN classifier. 84
6.8 Performance scores for every split of the datasets determined by STK evaluated by k-NN classifier. 85
6.9 Performance scores for every split of the datasets determined by NN evaluated by k-NN classifier. 86

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

6.10 Performance scores for every split of the datasets determined by GB evaluated by NN classifier. 87
6.11 Performance scores for every split of the datasets determined by RF evaluated by NN classifier. 88
6.12 Performance scores for every split of the datasets determined by STK evaluated by NN classifier. 89
6.13 Performance scores for every split of the datasets determined by NN evaluated by NN classifier. 90
6.14 Performance scores for every split of the datasets determined by GB evaluated by NN classifier. 91
6.15 Performance scores for every split of the datasets determined by RF evaluated by DT classifier. 92
6.16 Performance scores for every split of the datasets determined by STK evaluated by DT classifier. 93
6.17 Performance scores for every split of the datasets determined by NN evaluated by DT classifier. 94

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Acronyms

DaMi Data Mining. 36

DT Decision Tree. ix, 16

FS-NN Feature Selection with Neural Network. 56

GB Gradient Boosting. 42

ICMP Internet Control Message Protocol. 37

IDS Intrusion Detection System. 35

IP Internet Protocol. 38

k-NN k-Nearest Neighbors. ix, 14

ML Machine Learning. 7

NN Neural Network. x, 24

ODDS Outlier Detection Dataset. 35

RF Random Forest. 41

SS Stability Selection. 55

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU WIEN Bibliothek Your knowledge hub

SS-STK Stability Selection with Stacking. 55

SS-GB Stability Selection with Gradient Boosting. 55

SS-RF Stability Selection with Random Forest. 55

STK Stacking. 42

SVM Support Vector Machine. ix, 11

TCP Transmission Control Protocol. 37

UDP User Datagram Protocol. 37

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Bibliography

[AKD+12] Wael Awada, Taghi M. Khoshgoftaar, David Dittman, Randall Wald, and Amri Napolitano. A review of the stability of feature selection techniques for bioinformatics data. In *2012 IEEE 13th International Conference on Information Reuse Integration (IRI)*, pages 356–363, 2012.

[Alp10] Ethem Alpaydin. *Introduction to Machine Learning*. The MIT Press, 2nd edition, 2010.

[BB12] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(10):281–305, 2012.

[BBHJSG14] Elaheh Biglar Beigi, Hossein Hadian Jazi, Natalia Stakhanova, and Ali A. Ghorbani. Towards effective feature selection in machine learning-based botnet detection approaches. In *2014 IEEE Conference on Communications and Network Security*, pages 247–255, 2014.

[BCS15] Shemim Begum, Debasis Chakraborty, and Ram Sarkar. Data classification using feature selection and knn machine learning approach. In *2015 International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 811–814, 2015.

[BFOS84] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[BG05] Irad Ben-Gal. *Outlier Detection*, pages 131–146. Springer US, Boston, MA, 2005.

[BL17] Nikhil Buduma and Nicholas Locascio. *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*. O'Reilly Media, Inc., 1st edition, 2017.

[Bre94] Leo Breiman. Bagging predictors, 1994.

[Bre96] Leo Breiman. Stacked regressions. *Mach. Learn.*, 24(1):49–64, July 1996.

[Bre98] Leo Breiman. Arcing classifier (with discussion and a rejoinder by the author). *The Annals of Statistics*, 26(3):801 – 849, 1998.

[Bre01] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[BVS03] R. Barandela, R. M. Valdovinos, and J. S. Sánchez. New applications of ensembles of classifiers. *Pattern Anal. Appl.*, 6(3):245–256, December 2003.

[Chi21] Davide Chicco. *Siamese Neural Networks: An Overview*, pages 73–94. Springer US, New York, NY, 2021.

[CO06] John Cavazos and Michael F. P. O'Boyle. Method-specific dynamic compilation using logistic regression. In *Proceedings of the 21st Annual ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages, and Applications*, OOPSLA '06, page 229–240, New York, NY, USA, 2006. Association for Computing Machinery.

[Cox58] D. R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, 20(2):215–242, 1958.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[CVA06] Q. Cheng, P.K. Varshney, and M.K. Arora. Logistic regression for feature selection and soft classification of remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, 3(4):491–494, 2006.

[CZS+16] Guilherme O. Campos, Arthur Zimek, Jörg Sander, Ricardo J. G. B. Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E. Houle. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30(4):891–927, 2016.

[Dem06] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30, 2006.

[DHG20] Doreswamy, Mohammad Kazim Hooshmand, and Ibrahim Gad. Feature selection approach using ensemble learning for network anomaly detection. *CAAI Trans. Intell. Technol.*, 5:283–293, 2020.

[Die00] Thomas G. Dietterich. Ensemble methods in machine learning. In *MULTIPLE CLASSIFIER SYSTEMS, LBCS-1857*, pages 1–15. Springer, 2000.

[DLR+18] Gregory Ditzler, Joseph LaBarck, James Ritchie, Gail Rosen, and Robi Polikar. Extensions to online feature selection using bagging and boosting. *IEEE Transactions on Neural Networks and Learning Systems*, 29(9):4504–4509, 2018.

[DSC20] Agnieszka Duraj, Piotr S. Szczepaniak, and Lukasz Chomatek. Intelligent detection of information outliers using linguistic summaries with non-monotonic quantifiers. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 787–799, Cham, 2020. Springer International Publishing.

[FH89] E. Fix and J. L. Hodges. Discriminatory analysis - nonparametric discrimination: Consistency properties. *International Statistical Review*, 57:238, 1989.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[FM04] Giles M. Foody and Ajay Mathur. Toward intelligent training of supervised image classifications: directing training data acquisition for svm classification. *Remote Sensing of Environment*, 93(1):107–117, 2004.

[FVV+17] Daniel C. Ferreira, Félix Iglesias Vázquez, Gernot Vormayr, Maximilian Bachl, and Tanja Zseby. A meta-analysis approach for feature selection in network traffic research. In *Proceedings of the Reproducibility Workshop*, Reproducibility '17, page 17–20, New York, NY, USA, 2017. Association for Computing Machinery.

[GE03] Isabelle Guyon and Andre Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, 2003.

[GG05] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In David E. Losada and Juan M. Fernández-Luna, editors, *Advances in Information Retrieval*, pages 345–359, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[GIW09] P. Geurts, A. Irrthum, and L. Wehenkel. Supervised learning with decision tree-based methods in computational and systems biology. *Molecular bioSystems*, 5 12:1593–605, 2009.

[HBV+13] Milos Hauskrecht, Iyad Batal, Michal Valko, Shyam Visweswaran, Gregory F. Cooper, and Gilles Clermont. Outlier detection for patient monitoring and alerting. *Journal of Biomedical Informatics*, 46(1):47–55, 2013.

[IHZZ20] Félix Iglesias, Alexander Hartl, Tanja Zseby, and Arthur Zimek. Are network attacks outliers? a study of space representations and unsupervised algorithms. In Peggy Cellier and Kurt Driessens, editors, *Machine Learning and Knowledge Discovery in Databases - International Workshops of ECML PKDD 2019, Proceedings*, Communications in Computer and Information Science, pages 159–175, Germany, 2020. Springer. 19th Joint European Conference on Machine Learning and Principles and Practice of Knowledge

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

Discovery in Databases, ECML PKDD 2019 ; Conference date: 16-09-2019 Through 20-09-2019.

[IZ14] Félix Iglesias and T. Zseby. Analysis of network traffic features for anomaly detection. *Machine Learning*, 101:59–84, 2014.

[Kan03] Laveen N. Kanal. *Perceptron*, page 1383–1385. John Wiley and Sons Ltd., GBR, 2003.

[KB17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[KD19] Utkarsh Mahadeo Khaire and R. Dhanalakshmi. Stability of feature selection algorithm: A review. *Journal of King Saud University - Computer and Information Sciences*, 2019.

[KJ97] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1):273–324, 1997. Relevance.

[KR18] Denys Katerenchuk and Andrew Rosenberg. Rankdcg: Rank-ordering evaluation measure, 2018.

[LCWE06] Thomas Navin Lal, Olivier Chapelle, Jason Weston, and André Elisseeff. *Embedded Methods*, pages 137–165. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[LS96] Huan Liu and R. Setiono. A probabilistic approach to feature selection - a filter solution. In *ICML*, 1996.

[MB10] Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society, Series B*, 72:417–473, 2010.

[McC89] Donna Katzman McClish. Analyzing a portion of the roc curve. *Medical Decision Making*, 9(3):190–195, 1989. PMID: 2668680.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[MD03] Melanie Middlemiss and Grant Dick. *Feature Selection of Intrusion Detection Data Using a Hybrid Genetic Algorithm/KNN Approach*, page 519–527. IOS Press, NLD, 2003.

[Mel13] Francisco Melo. *Area under the ROC Curve*, pages 38–39. Springer New York, New York, NY, 2013.

[Mit97] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[MRT12] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012.

[MS15] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*, pages 1–6, 2015.

[Pes17] Barbara Pes. Feature selection for high-dimensional data: The issue of stability. In *2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, pages 170–175, 2017.

[Pes19] B. Pes. Ensemble feature selection for high-dimensional data: a stability analysis across multiple domains. *Neural Computing and Applications*, 32:5951–5973, 2019.

[PLA05] T.M. Phuong, Z. Lin, and R.B. Altman. Choosing snps using feature selection. In *2005 IEEE Computational Systems Bioinformatics Conference (CSB'05)*, pages 301–309, 2005.

[PM18] Utkarsh Porwal and Smruthi Mukund. Credit card fraud detection in e-commerce: An outlier detection approach. *CoRR*, abs/1811.02196, 2018.

[POA15] Elnaz Pashaei, Mustafa Ozen, and Nizamettin Aydin. Improving medical diagnosis reliability using boosted c5.0 decision tree empowered by particle

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

swarm optimization. In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 7230–7233, 2015.

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[Ray16] Shebuti Rayana. Odds library, 2016.

[Rie94] Martin Riedmiller. Advanced supervised learning in multi-layer perceptrons - from backpropagation to adaptive learning algorithms, 1994.

[Sas07] Yutaka Sasaki. The truth of the f-measure. *Teach Tutor Mater*, 01 2007.

[SAVdP08] Yvan Saeys, Thomas Abeel, and Yves Van de Peer. Robust feature selection using ensemble feature selection techniques. In Walter Daelemans, Bart Goethals, and Katharina Morik, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 313–325, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[Sch90] Robert E. Schapire. The strength of weak learnability. *Mach. Learn.*, 5(2):197–227, July 1990.

[SLG18] Iman Sharafaldin, Arash Habibi Lashkari, and A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSP*, 2018.

[SMT09] C. Strobl, J. Malley, and G. Tutz. An introduction to recursive partitioning: rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological methods*, 14 4:323–48, 2009.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[SR08] Shina Sheen and R. Rajesh. Network intrusion detection using feature selection and decision tree classifier. In *TENCON 2008 - 2008 IEEE Region 10 Conference*, pages 1–4, 2008.

[SS12] Rajen D. Shah and Richard J. Samworth. Variable selection with error control: another look at stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(1):55–80, Jun 2012.

[STA10] Nasser H. Sweilam, A.A. Tharwat, and N.K. Abdel Moniem. Support vector machine for diagnosis cancer disease: A comparative study. *Egyptian Informatics Journal*, 11(2):81–92, 2010.

[STS16] Amanpreet Singh, Narina Thakur, and Aakanksha Sharma. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1310–1315, 2016.

[TAL14] Jiliang Tang, Salem Alelyani, and Huan Liu. *Feature selection for classification: A review*, pages 37–64. CRC Press, January 2014.

[TBK07] Muhammad Atif Tahir, Ahmed Bouridane, and Fatih Kurugollu. Simultaneous feature selection and feature weighting using hybrid tabu search/k-nearest neighbor classifier. *Pattern Recognition Letters*, 28(4):438–446, 2007.

[VE15] Jorge R. Vergara and Pablo A. Estévez. A review of feature selection methods based on mutual information. *CoRR*, abs/1509.07577, 2015.

[Wol92] David H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

[XHWZ14] Zhixiang Xu, Gao Huang, Kilian Q. Weinberger, and Alice X. Zheng. Gradient boosted feature selection. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN

[YM11] Feng Yang and K. Z. Mao. Robust feature selection for microarray data based on multicriterion fusion. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(4):1080–1092, 2011.

[Zho12] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman and Hall/CRC, 1st edition, 2012.

[ZZL+17] Wenying Zhang, Huaguang Zhang, Jinhai Liu, Kai Li, Dongsheng Yang, and Hui Tian. Weather prediction with multiclass support vector machines in the fault detection of photovoltaic system. *IEEE/CAA Journal of Automatica Sinica*, 4(3):520–525, 2017.

Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.
TU Bibliothek Your knowledge hub WIEN