Universität Hamburg
Department Informatik

# Textklassifikation als Unterdisziplin des überwachten maschinellen Lernens

Seminararbeit

Seminar Text Mining

michal.zlotnik@studium.uni-hamburg.de
bav1488@studium.uni-hamburg.de

Michal Zlotnik, Leonhard Rattay

Matr.Nr. 7054053, 7031053

michal.zlotnik@studium.uni-hamburg.de
bav1488@studium.uni-hamburg.de

31.07.2020

# Abstract (MZ)

Text classification as a sub-discipline of a supervised machine learning is a subject of ongoing large-scale research in the field of text mining. This seminar paper's focus is placed on introductory concepts related to a general approach model in the context of text classification, e.g.: feature extraction and selection, three classification algorithms: Naïve Bayes, Decision Tree, and Support Vector Machine.

# Contents

# 1 Introduction (MZ)

## 1.1 Motivation

The growth of internet use and the resulting expansion of online social platforms, communication as well as e-commerce services provide a wide range of rich information sources. As a result, unstructured data in the form of text is present in e-mails, chat messages, microblog posts, or online customer reviews. However, due to its structural nature and great volume, extracting insights or adjusting the most relevant data to users' preferences imposes a series of challenges. Hence, the application of text classification could improve the effectiveness of enhancing of business-related aspects like decision-making or process automation. Furthermore, it might be applied to spam filtering, tweet personalization, and aspect-based sentiment analysis of short texts.

## 1.2 Scope

This work aims to present text classification approaches as a sub-discipline of supervised machine learning concerning long and short texts. Its main focus is placed on theoretical foundations and review of their applications.

Due to the fact, that in the literature and academic research papers, the models and corresponding algorithms are presented based on the black-box approach, Section 2 provides a theoretical introduction mathematical foundations related to Naïve Bayes, Support Vector Machine, and Decision Tree.

A brief introduction to definitions underlying text classification itself as well as supervised machine learning is covered in Section 3.

To aid understanding of model's presentation and evaluation, the general approach model as well as further theoretical aspects are discussed in a more detailed manner in Section 4, e.g. feature extraction and selection, micro and macro averaging, aspect-based sentiment analysis.

Common problems related to text classification as well as aspect-based sentiment analysis are covered in Section 5. Finally, Section 6 contains conclusions that summarize the seminar paper.

# 2 Theoretical background (MZ)

## 2.1 Mathematical foundations

This section provides a subjective selection of theoretical aspects concerning three classifiers that are subject of interest in this seminar paper. It adds the mathematical foundations, that might deepen the understanding of their inner workings.

### 2.1.1 Bayes Theorem

The notion of **conditional probability** aids obtaining full knowledge about the outcome of an experiment based on a partial information[1]. **Prior probability** provides the probability of an event before the additional knowledge is given. **Posterior probability** of the event results from the utilization of additional knowledge.
Provided that $P(B) > 0$ the probability of an event A given that the event B has taken place is expressed as [2]:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{1}$$

Due to the multiplication rule [1] in case of $P(B) = 0$, the equation (1) can be unformed to:

$$P(A \cap B) = P(B)P(A|B) = P(A)P(B|A) \tag{2}$$

Events $A$ and $B$ are called **independent** if:

$$P(A \cap B) = P(A)P(B) \tag{3}$$

In case of $P(B) \neq 0$ equation (3) could be expressed as $P(A) = P(A|B)$ which means that the probability of $B$ does not affect the probability of $A$. The above implications is obtained from the **chain rule** that states:

$$P(A_1 \cap A_n) = P(A_1)P(A_2|A_1)P(A_3|A_1 \cap A_2)...P(A_n| \cap_{i=1}^{n-1} A_i) \tag{4}$$

The order of dependence of events can be altered as a result of **Bayes theorem**. It allows calculations of $P(B|A)$ in terms of $P(A|B)$:

$$P(B|A) = \frac{P(B \cap A)}{P(A)} = \frac{P(A|B)P(B)}{P(A)} \tag{5}$$

If group of sets $B_i$ partition set $A : A \subseteq \cap_i B_i$ ($B_i$ disjoint) then

$$P(A) = \sum P(A|B_i)P(B_i) \tag{6}$$

allows the equation (5) to be unformed provided that $A \subseteq \cup_{i=1}^{n} B_i, P(A) > 0$, $B_i \cap B_j, i \neq j$ [1]:

$$P(B_j|A) = \frac{P(A|B_j)P(B_j)}{P(A)} = \frac{P(A|B_j)P(B_j)}{\sum_{i=1}^{n} P(A|B_i)P(B_i)} \tag{7}$$

### 2.1.2 Support Vector Machine (SVM)

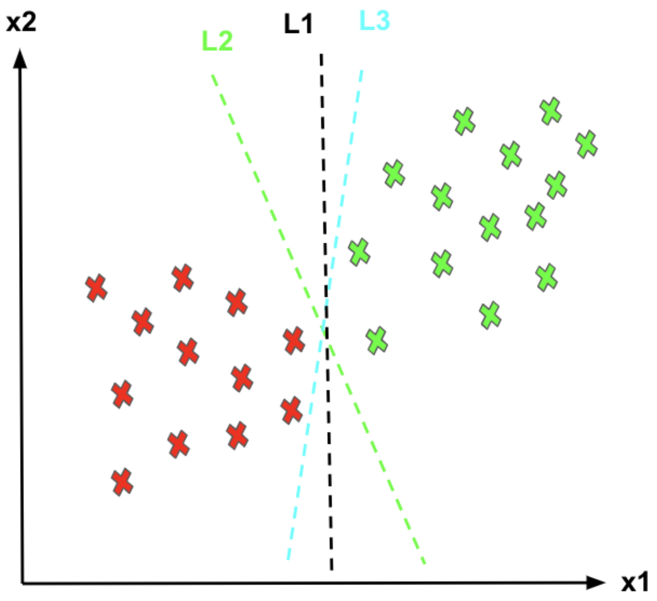


Figure 1: Possible hyperplanes $L_1, L_2, L_3$ separating two classes of data points in 2-dimensional vector space. [3]

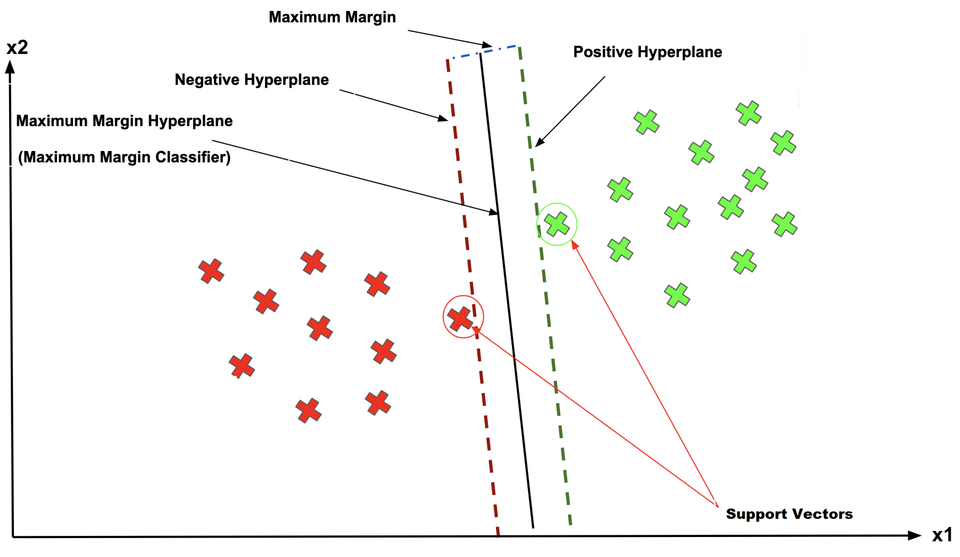


Figure 2: Negative and positive hyperplanes with maximum margins and support vector points [3]

Finding an optimal hyperplane separating two classes of data points (Figure 1), i.e. one that would enable correct separation when new points are added, is a crucial stage during the classification process to obtain high accuracy. Figure 2 presents the optimal hyperplane with negative and positive hyperplanes as well as support vectors. The distance between both hyperplanes and the optimal one is called margin, i.e. an area which does not contain any data points (which might not be the case in practical cases, however, the no-points approach is used for simplicity of explanation). The closest data points to negative and positive hyperplanes are called support vectors. As a result, the choice of the optimal one is determined by the maximal distance between itself and the closest data points.

Underlying mathematical concepts related to SVM classifier presented by Jakkula [4] contribute to a better understanding of its inner workings.
**N-dimensional vector** can be calculated by the following equation:

$$||x|| = \sqrt{x_1^2 + x_2^2 +,,, + x_n^2} \tag{8}$$

The above equation implies that the **unit vector $\hat{\mathbf{x}}$** is obtained from:

$$\hat{\mathbf{x}} = \frac{x}{||x||} \tag{9}$$

To find the possibly the most optimal maximum margin that would separate two data points classes, further calculations are to be considered. The equation of a hyperplane is derived as follows:

$$y = ax + by - ax - b = 0 \tag{10}$$

Given two vectors $w = \begin{pmatrix} -b \\ -a \\ 1 \end{pmatrix}$ and $x = \begin{pmatrix} 1 \\ x \\ y \end{pmatrix}$:

$$w^T x = -b \times 1 + (-a) \times x + 1 \times y w^T x = y - ax - b \tag{11}$$

In SVM $w$ represents a weight vector and $b$ a bias and the following equation is used:

$$w^T x + b = 0 \tag{12}$$

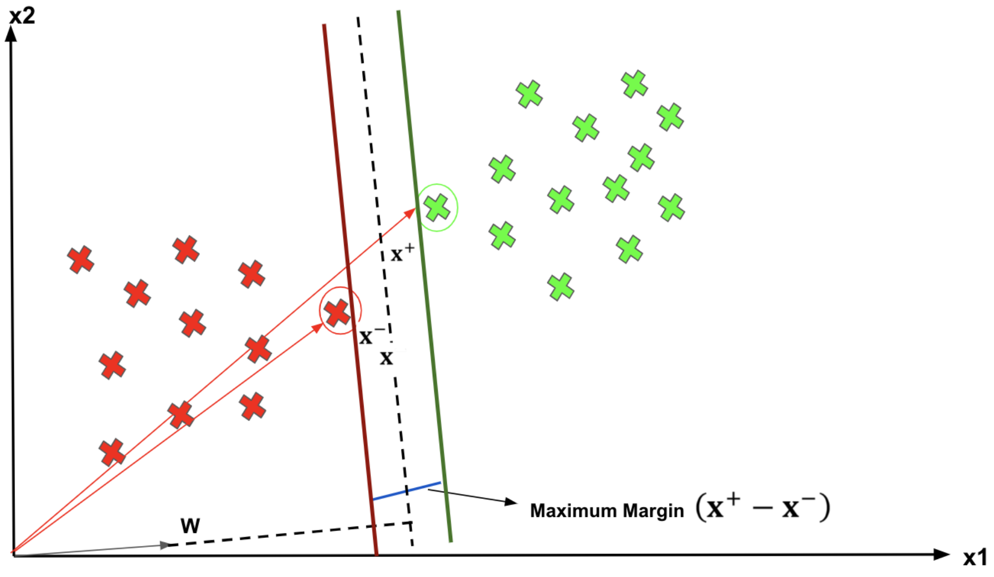


Figure 3: Maximum margin and orthogonal vector $w$ [3]

Assuming that there are two support vectors depicted in Figure 3 where one is a positive example $x^+$ and the other negative $x^-$, that defines the margin, the maximum margin is calculated by $x^+ - x^-$. Then, performing dot product operation on the resulting vector and the orthogonal unit vector $w$:

$$(x^+ - x^-) \cdot \hat{\mathbf{w}} = (x^+ - x^-) \cdot \frac{w}{||w||} = x^+ \cdot \frac{w}{||w||} - x^- \cdot \frac{w}{||w||} \tag{13}$$

**Hinge loss** function helps to maximize the margin, by measuring the error due to misclassification and by regularisation. The latter determines the trade-off between increasing the size of the margin and ensuring the $x_i$ is located on the correct side of the margin.

$$L(w) = \sum_{i=1} max(0, 1 - y_i[w^T x_i + b]) + \lambda ||w||_2^2 \tag{14}$$

From **hinge loss** the equation for the margin can be derived under assumptions that $y_i \in \{-1, 1\}$ and the if the data point is located precisely on the classifier's margin the **hinge loss** is equal to 0.

$$max(0, 1 - y_i[w^T x_i + b]) = 0 \Rightarrow y_i[w^T x_i + b] = 1 \tag{15}$$

Applying to Equation 13 gives:

$$x^+ \cdot \frac{w}{||w||} - x^- \cdot \frac{w}{||w||} = \frac{1-b}{||w||} - \frac{-b-1}{||w||} = \frac{2}{||w||} \tag{16}$$

While $\frac{2}{||w||}$ is considered twice the margin's size it can be calculated as $\frac{1}{||w||}$. Finally, as the optimal hyperplane optimizes the margin then the SVM's objective is to maximize the term $\frac{1}{||w||}$:

$$max \frac{1}{||w||}$$

which can be rewritten as

$$min||w||$$

and as

$$min\frac{||w||^2}{2} \tag{17}$$

**Kernel trick** helps to classify data that is not linearly separable. The essential principle can be summarised by transforming low-dimensional input space into a higher-dimensional input space.

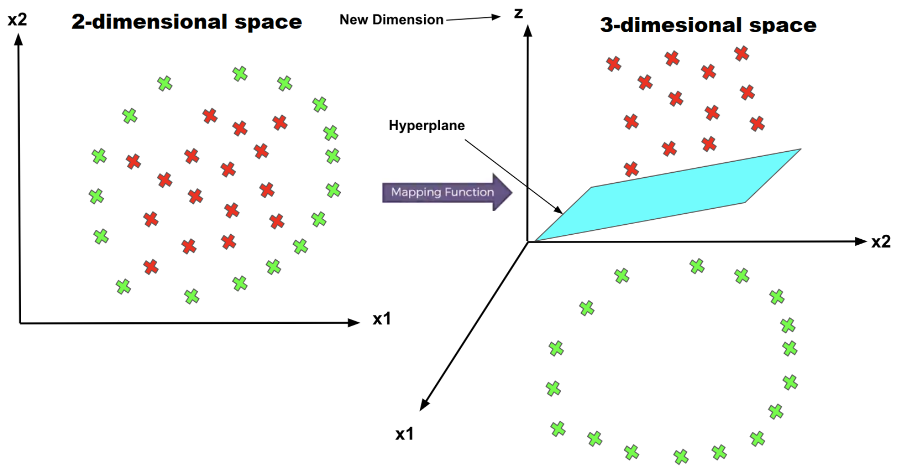
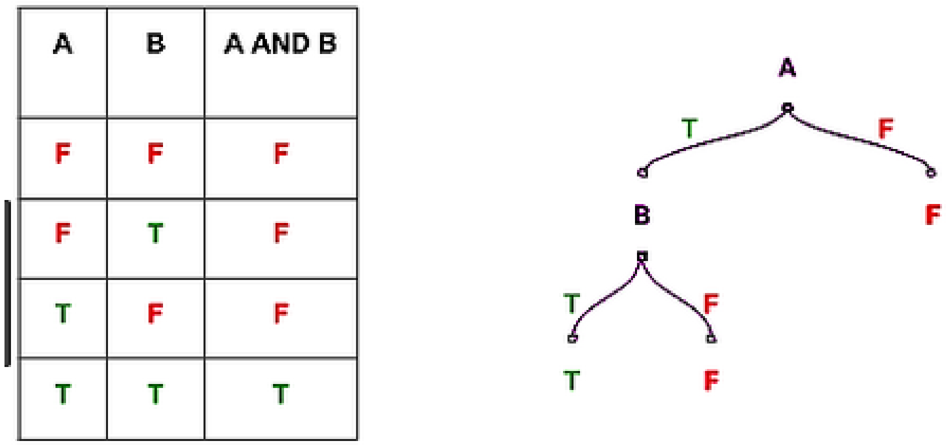


Figure 4: Kernel Trick [3]

### 2.1.3 Decision Tree



| A | B | A AND B |
|---|---|---|
| F | F | F |
| F | T | F |
| T | F | F |
| T | T | T |

Figure 5: Simple decision tree [5]

Figure 5 presents a simple example of the AND binary operation case. If $A$ and $B$ are considered as features, $A \wedge B$ corresponding label, then:

- If $A = T$ and $B = T$ then the result is $T$;
- If $A = T$ and $B = F$ then the result is $F$;
- If $A = F$ then the result is $F$.

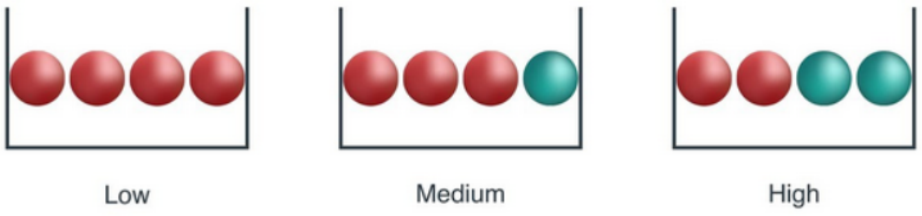


Figure 6: Impurity [5]

**Impurity measures**. Taking into consideration Figure 6, if a ball is drawn from the right bowl then the maximum information is needed about the balls as the number of the red and green ball is the same. The analogous principle applies to the remaining bowls. The medium needs less information as one is blue and the left-most requires minimum information as all the balls are red. To sum up, if the information is regarded as a measure of purity, then starting from right to left, the right bowl is the impurest and the most left bowl is the purest. **Entropy** is one of the impurity measures and can be understood as the amount required to describe a sample. Its values range between zero when the sample is homogeneous and one when the sample is distributed uniformly. In other words, "it is the sum of probability of label time the log probability of the same label"[6]:

$$Entropy = -\sum_{i=1}^{n} p_i \times log_2(p_i) \tag{18}$$

Gini index measures inequality in a sample [7]. Similarly to entropy, its values range from zero (all elements are similar) to one (all elements are unequal):

$$GI = 1 - \sum_{i=1}^{n} p_i^2 \tag{19}$$

**Information gain** is a metric that defines the number of information bits the presence of a given word guesses the class ($c_i$ is the i-th class, $\bar{w}$ when the document does include a word)[7]:

$$\begin{aligned} IG(w) = &-\sum_{i=1}^{C} P(c_i)log_2P(c_i) \\ &+ P(w)\sum_{i=1}^{C} P(c_i|w)log_2P(c_i|w) \\ &+ P(\bar{w})\sum_{i_1}^{C} P(c_i|\bar{w})log_2P(c_i|\bar{w}) \end{aligned} \tag{20}$$

# 3 Text classification (LR)

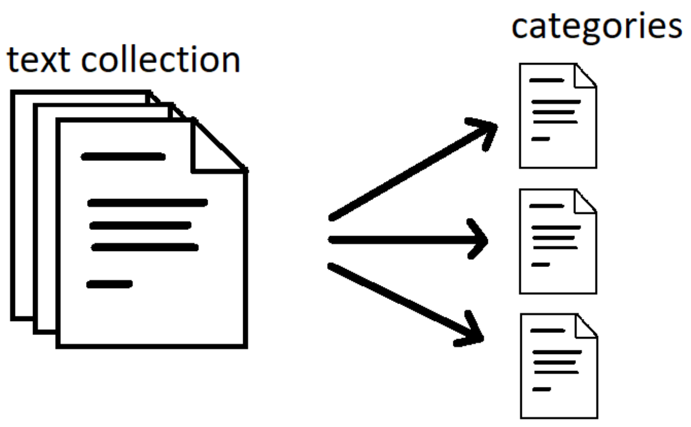


Figure 7: Text classification tasks

The task in classification problems is to sort a collection of objects into different categories (Figure 7)[1]. Examples of text classification problems are: a spam-filter with the task to sort e-mails into relevant and irrelevant e-mails, a library-sorting engine with the task to categorize books into different topics, or a restaurant-review evaluation engine with the task to categorize reviews into different aspects of what the costumers enjoyed or didn't like.[8]

## 3.1 Machine learning (LR)

Given a specific task to be solved by a program, there are two main approaches. The first approach is the knowledge engineer approach in which an expert puts his knowledge directly into the program. Often in form of a set of rules which the program follows or bases its decision on. The main drawback of this approach is that it often takes a huge amount of work to create and maintain these set of rules.[8]

The second approach is through machine learning. It is an inductive training process in which a program can create its own set of rules, based on its past experience. In general Machine learning programs are easier to create and to maintain [8]. Usually already existing machine learning algorithms can be modified to solve a specific task. Machine learning algorithms can be split into two groups: supervised and unsupervised machine learning.[8]

Supervised machine learning means that throughout the training process the program gets feedback. Usually in form of a labeled data set, in which the right solution is imprinted on each training sample. But to create the labeled data sets takes usually a lot of work and sometimes their creation is not possible, due to not enough information on the subject[8]. So there is also unsupervised machine learning in which the machine learning algorithms have no feedback.[8]

## 4 General approach model (LR)

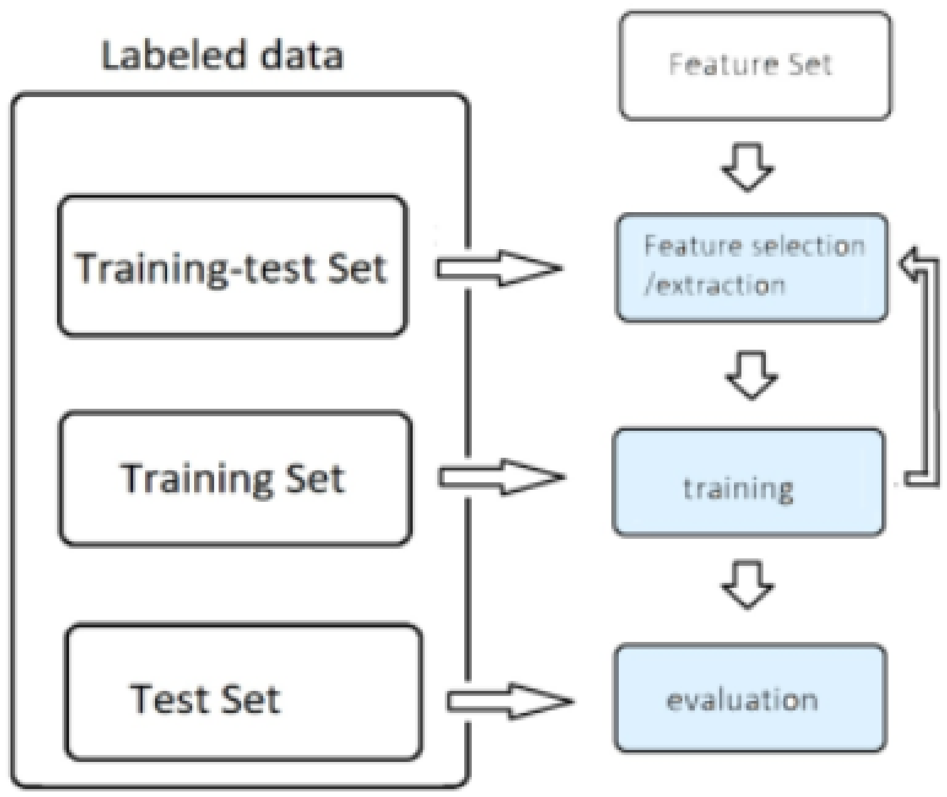


Figure 8: Text classification tasks

Figure 8 presents a general approach on how to solve text classification problems through machine learning (based on [9]).

It can be seen that the labeled data set is split into three sets: a training set, a training-test set and a test set.[9] To get unbiased test results it is important that each test set is separate from the other sets, otherwise the test results only represents the learning process on known data samples and not the performance on a completely new input.[9]
Usually a machine learning program does not work with the raw data samples. The data samples are often transformed into a simpler and more manageable form through the help of a feature set [1]. This will be covered in section 4.1
The general process of this approach is as follows: First the program selects or extracts from a given feature set a couple of features and uses these to represent the labeled data samples. Then it trains on the training set. There are a lot of different training Algorithms. We will introduce a couple of them in section 4.3.

Selecting the best feature set is often a difficult task. Using a lot of features is not always useful because some features don't have valuable information or can even be counter productive if their insights are based on a peculiarity of the training

set. So after the training process the program can reevaluate its chosen feature set and optimise its choices. This process between training and feature reevaluation might go on, till the program thinks its got the optimal feature set or the gain of optimising the feature set compared to the old feature set is getting too small.
At last it often useful to evaluate how well the program performs on a new test set. Note that it is important that the test set is separate of the training-test set, because the feature sets optimisation is based on the training-test set, so the program usually performs better on the training-test set than on completely new test samples[9]. We will introduce a couple of commonly used evaluation methods in section 4.4.

## 4.1 Feature representation (LR)

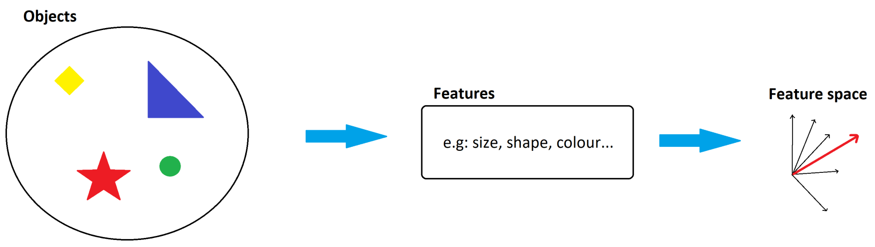
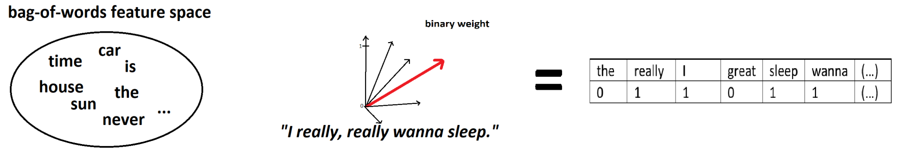


Figure 9: Feature representation

The goal of a feature representation is to represent for the task useful characteristics of a set of objects through a set of features and by doing that reducing the amount of data, which makes the objects easier to handle[8].
For example in figure 9 we want to represent a set of objects through features. Useful features in this case might be: size, shape or colour. Now each feature can be represented as a vector through that feature space.
For example: blue triangle = (big,triangle,blue), green circle = (small,circle,green).

For text classification problems there are a few commonly used feature spaces. One common feature space is the bag-of-words feature space. In the bag-of-words feature space each different word used in the documents represents a feature[8]. There are different weighting methods. A simple weighting method is a binary weighting scheme, which is either one (if the word is used in a certain document) or zero (if the word is not used)[8]. One example is shown in Figure 10.



| the | really | I | great | sleep | wanna | (...) |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 1 | (...) |

Figure 10: Bag-of-words feature space

One of the most commonly used weighting scheme is the TF-IDF scheme (Term Frequency - Inverse Document Frequency)[8]:

**common weighting scheme:**

$$\text{TF-IDF-Weight}(\overset{\text{word}}{w}, \overset{\text{document}}{d}) = \underbrace{\text{TermFreq}(w, d)}_{\text{number of occurences of w in d}} \bullet \log\left(\overset{\text{number of all documents}}{N} / \overset{\text{number of documents containing w}}{\text{DocFreq}(w)}\right)$$

Figure 11: TF-IDF scheme

The TF-IDF scheme also takes into account how often a word is used in a document and how rare the word is in the whole text collection.[8]

## 4.2 Feature selection/extraction (LR)

Selecting the best set of features and optimising the feature selection is a difficult task with a lot of research into it. Here we just want to cover two main ideas on which many feature optimisation algorithms are built upon: feature selection and feature extraction:

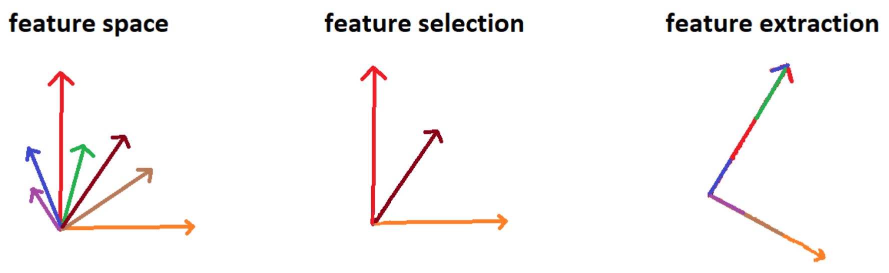


Figure 12: Feature selection / Feature extraction

One of the main goals of feature selection and feature extraction is to reduce the dimensionality of the feature space by removing irrelevant or redundant data[8]. This can improve the accuracy, performance, or the comprehensibility of the program. Feature selection evaluates the usefulness of each feature and reduces the feature space by simply selecting the useful features and removing the hollow features[8]. This method has good results if there are is a lot of information in a small set of features, but has worse performances if a lot of the information is very distributed throughout all the features, because in that case selecting just a couple of features could result in a piece of information lost[8].

Feature extraction, on the other hand, is often able to reduce the dimensionality of a feature space through combining features, without losing any information, but the resulting feature spaces are for humans hardly interpretable and it is often unclear on how much an original feature contributes to the resulting feature space.[8]
Figure 12 shows the basic concept of both approaches.

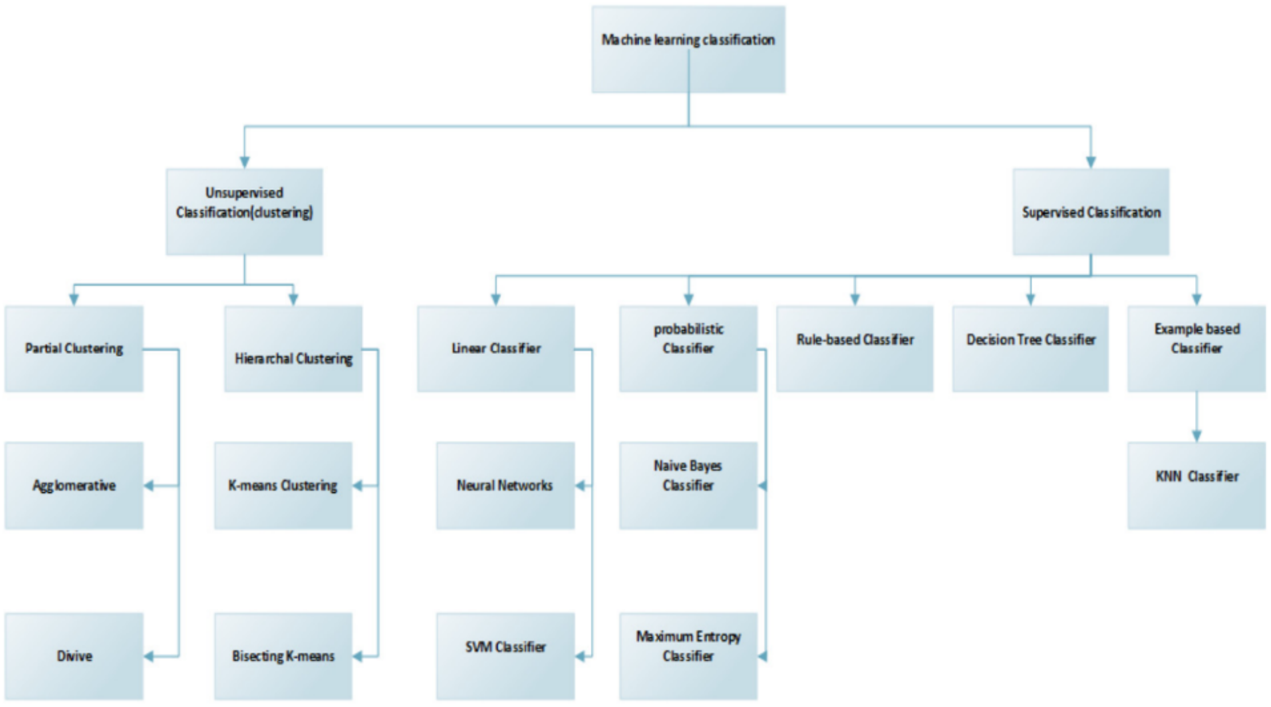


Figure 13: Overview of machine learning classifiers [10]

## 4.3 Training: classifiers (MZ)

### 4.3.1 Naïve Bayes

**Naïve Bayes (NB)** is part of the family of probabilistic classifiers (Figure 13). It finds the probability of a document belonging to a particular class by taking into account all features in that class. Figure 14 depicts a simplified yet insightful classification process by using NB. In a given corpus, the majority of documents are *automotive*, hence the initial point is located near the *automotive* label. However, NB classificator takes into account all features. *Dark* is a weak indicator for *murder mystery*, which results in a minor shift. Finally, *football* is a strong indicator for *sports* and the process ends with classifier checking which label it is closest to and attached that label to the input. More detailed as well as the realistic process of likelihood calculations is presented by Figure 15. First, the frequency of occurrence of every label in the training dataset is evaluated (prior probability). The next step consists of every feature contribution to the estimate for each label by multiplying it by the probability that input values with that label assumed to have that feature. Finally, under the assumption of independence the resulting score (label likelihood) can be interpreted as an estimation that of the probability that a randomly selected label from the training dataset value would have the set of features and the given label. The reason for NB being *naive* is directly linked to the independence assumption of features. A strict application of the independence of feature construction of the optimal feature sets would be challenging. On the other hand, ignoring the notion of independence would result

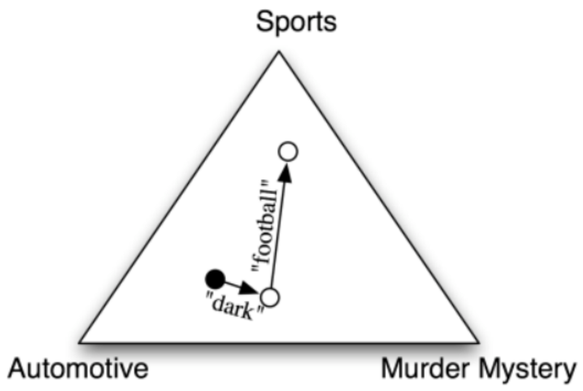


Figure 14: Simplified procedure of choosing correct topic for a document by using Naïve Bayes approach [6]

in highly correlated features that would *push* the classifier closer than necessary to a given label (**double counting**) [6].

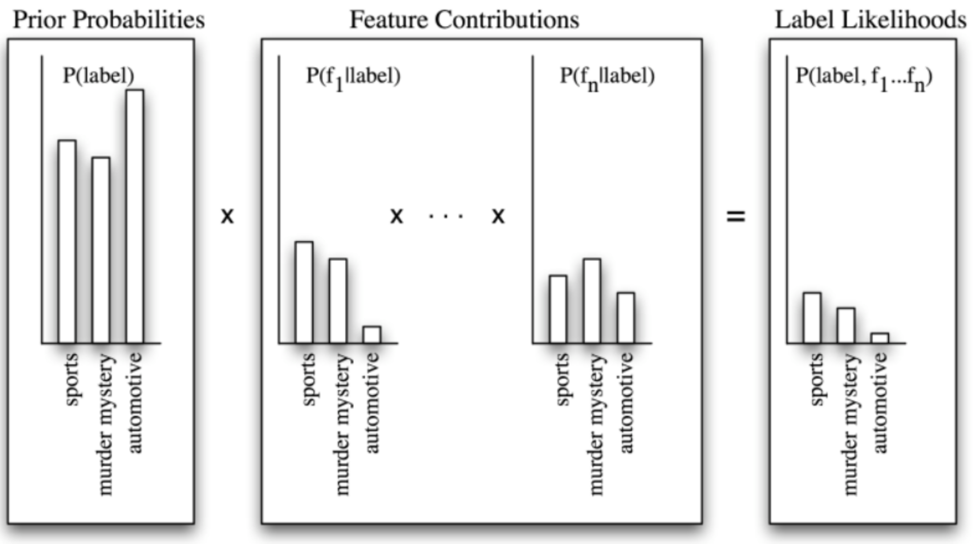


Figure 15: Label likelihood calculations with Naïve Bayes [6]

### 4.3.2 Support Vector Machine

**Support Vector Machine (SVM)** belongs to a group of linear classifiers. Its decision-making is based on the linear combination value of features [10]. The binary SVM classification method can be understood as a hyperplane that in the n-dimensional feature space that separates positive and negative instances (which are sometimes referred to as members and non-members [10]). Selection of the most optimal hyperplane separating the known positive and negative instances with the maximal margin takes place during the training stage. Hyperplanes in SVM are determined by a relatively small set of training instances - support vectors. It is worth noting, that as a result the rest of the training data has no impact on the trained classifier [8]. SVM is considered as one of the best text classifiers[11]. It is considered to be an excellent approach, for the reason that the text dataset described as sparse data in which the text has a tendency to be correlated and dispersed into linearly separated classes [10]. Subsequently, it posses an advantage when handling overfitting problems, which enables its high performance with disregard of the dimensionality of the feature space [8].

### 4.3.3 Decision Tree

**Decision Tree (DT)** in text classification is represented as an uncomplicated flowchart, where decision nodes prove feature values and leaf nodes attach labels. The process of text classification start at the root node that contains condition that proves one the of the input value's features. Then, it selects a branch based on its value. By following the branch containing input value a new decision node is reached and process continues once the leaf node is reached. Leaf node provides a label for the input value.[6] Usually a decision tree is build recursively by choosing a feature at each step and splitting the training dataset in two subsets. One containing that feature and another not containing. Recursion continues until documents of a single label are left. Furthermore, the choice of the features for each step is based on information measures described in more detail in Section 2.1.3. However, building the decision tree based on information measures may lead to over-fitting, hence some approaches might include **prunning** as a method of removing overly specific branches [8].

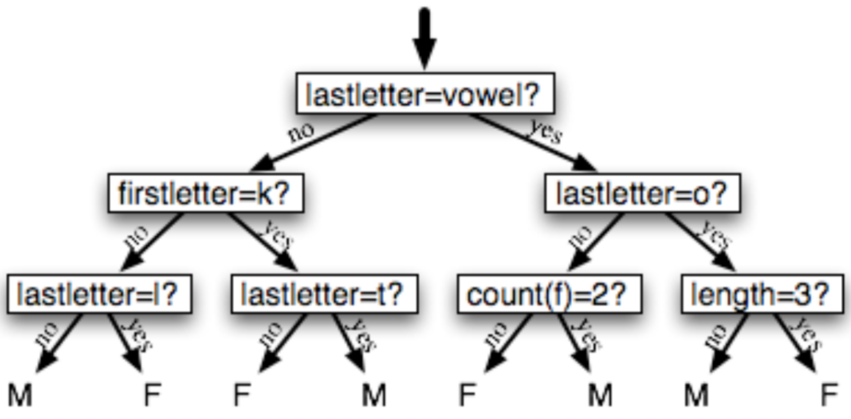


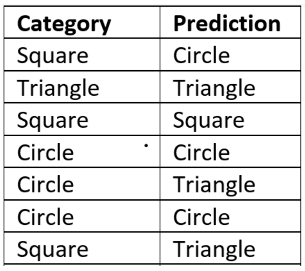
Figure 16: Decision Tree for a assigning gender to given names [6]

## 4.4 Evaluation (LR)

| | yes is correct | no is correct |
|---|---|---|
| yes was assigned | a (correct yes) | b (wrong yes) |
| no was assigned | c (wrong no) | d (correct no) |

Figure 17: Contingency table for a binary classifier (based on table 16.2 in [1])

To evaluate a program there are a couple of different measurements. The most commonly used ones are accuracy, precision, recall and fallout. For a binary classifier usually a contingency table is used (shown in Figure 17)[1]. The measurement for such a classifier are determined as follows: accuracy: $\frac{a+d}{a+b+c+d}$, precision: $\frac{a}{a+b}$, recall: $\frac{a}{a+c}$ and fallout: $\frac{b}{b+d}$ [1]. For more categories one usually create a contingency table for each category and then combine these tables either through micro averaging or through macro averaging. In micro averaging one first calculate the performance of each category and take the average of all the categories[1]. In macro averaging one create one big contingency table by taking the sum of each cell of all the categories and then calculate the overall performance[1]. Note that these two approaches usually leads to different results. While in micro averaging each category has the same weight, in macro averaging each sample has the same weight. So if the categories don't have the same amount of sample in macro averaging categories with more samples have a bigger impact than smaller categories and in micro averaging a sample of a smaller category has a bigger impact than a sample of a larger category. Figure 18 shows a example.



| Category | Prediction |
|---|---|
| Square | Circle |
| Triangle | Triangle |
| Square | Square |
| Circle | Circle |
| Circle | Triangle |
| Circle | Circle |
| Square | Triangle |

Figure 18: A (multi category) classifier test result. The recall of this test result is with micro averaging: $\frac{\frac{1}{3}+\frac{1}{1}+\frac{2}{3}}{3} = 0.66$. With macro averaging: $\frac{4}{7} = 0.57$.

# 5 Text classification problems (LR)

The following section will introduce two recently in popularity increasing text classification problems. In section 5.1 we will talk about the characteristics and challenges of working with short texts and in section 5.2 we will have a short introduction into aspect based sentiment analysis.

## 5.1 Short Texts (MZ)

**Short texts** tend to have not more than 200 characters. Generally, they encompass text and chat messages, posts on social media platforms, and reviews on e-commerce sites. Their classification involves the same sequence of stages as in the case of long texts. However, the main challenges are related to the following attributes of short text[12]:

- *Misspelling and informal writing*;
- *Instant*: determined by its application i.e. often being processed and transmitted in real time;
- *Sparsity*: it usually has a limited length (e.g. Tweets), hence spare capacity might be used to express a variety of topics.
- *Shortness:* it contains a few words, which might result in an inadequate representation of a document.

## 5.2 Aspect based sentiment analysis (LR)

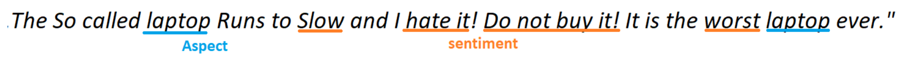

*The So called laptop Runs to Slow and I hate it! Do not buy it! It is the worst laptop ever."*

Figure 19: Example of a aspect based sentiment analysis

The goal of Aspect based sentiment analysis (ABSA) is to analyse texts (mostly short messages) and extract the sentiment expressed toward a certain object or aspect (Figure 19)[13]. Typical ABSA problems are review analyse programs, with the goal to extract the overall sentiment of the reviews, the positive aspects of them and the negative aspects of them. This task can usually be divided into 4 sub tasks[14]:

- aspect term extraction
- aspect term polarity
- aspect category detection
- aspect category polarity

A lot these Sub tasks could be seen as a classification problem. For example aspect term polarity could be seen as a classification problem into the categories: negative, neutral, positive.[14] All in all it is a text mining problems with probably high prospects and economic value.

# 6 Conclusion (LR, MZ)

The goal of this paper was to give insight into how text classification problems can be solved through supervised machine learning and to provide a general approach to how learning algorithms can be implemented. We first covered the mathematical basis, then a general approach model, and finally a short introduction into short text classifier problems and aspect-based sentiment analysis. In general, through the recent developments of social networks and the tremendous amount of new data, old methods like solving presented tasks manually is not sustainable. Consequently, the gap between knowledge engineering and machine learning is continually shrinking. In future there should be a continuous demand for text classification through supervised machine learning, because machine learning algorithms prove to be relatively simple in terms of implementation, adaptability to new tasks, and undemanding in maintenance.

## List of Figures

## References

[1] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge, MA, USA, 1999.

[2] `https://www.python-kurs.eu/text_klassifikation_einfuehrung.php`. Accessed: 2020-06-09.

[3] A top machine learning algorithm explained: Support vector machines (svm). `https://www.kdnuggets.com/2020/03/machine-learning-algorithm-svm-explained.html`. Accessed: 2020-06-25.

[4] Vikramaditya R. Jakkula. Tutorial on support vector machine ( svm ). 2011.

[5] `https://medium.com/@ankitnitjsr13`. Accessed: 2020-06-25.

[6] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit.* O'Reilly, Beijing, 2009.

[7] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition.* Pearson Prentice Hall, Upper Saddle River, N.J., 2009.

[8] Ronen Feldman and James Sanger. *Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data.* Cambridge University Press, USA, 2006.

[9] S. Bird, E. Klein, and E. Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit.* O'Reilly Media, 2009.

[10] Issa Alsmadi and Keng Hoon Gan. Review of short-text classification. *International Journal of Web Information Systems*, pages 155–182, 01 2019.

[11] Pascal Soucy and Guy W. Mineau. Feature selection strategies for text categorization. In Yang Xiang and Brahim Chaib-draa, editors, *Advances in Artificial Intelligence*, pages 505–509, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[12] Ge Song, Yunming Ye, Xiaolin Du, Xiaohui Huang, and Shifu Bie. Short text classification: A survey. *Journal of Multimedia*, 9, 05 2014.

[13] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülşen Eryiğit. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California, June 2016. Association for Computational Linguistics.

[14] Naveen Kumar Laskari and Suresh Kumar Sanampudi. Aspect based sentiment analysis survey. *IOSR Journal of Computer Engineering (IOSR-JCE)*, 18(2):24–28, 2016.

[15] Eugen Ruppert, Abhishek Kumar, and Chris Biemann. Lt-absa : An extensible open-source system for document-level and aspect-based sentiment analysis. 2017.

[16] Ian H. Witten, Eibe Frank, Mark A. Hall, and Christopher J. Pal. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann, Amsterdam, 4 edition, 2017.

[17] Alexander Clark, Chris Fox, and Shalom Lappin, editors. *The Handbook of Computational Linguistics and Natural Language Processing.* Wiley-Blackwell, 2010.

[18] J. Eisenstein. *Introduction to Natural Language Processing.* Adaptive Computation and Machine Learning series. MIT Press, 2019.