Sentiment Analysis in Social Media

Master's Thesis in Computer Science

Natalia Vyrva

May 24, 2016
Halden, Norway

Østfold University College www.hiof.no

Abstract

This thesis presents a comparison of different machine learning techniques applied to the case of sentiment analysis in social media. Several machine learning methods were used during experimentation session: Naive Bayes, Multinomial Naive Bayes, Support Vector Machines, Random Forest, Multilayer Perceptron Network. Besides, we tried to compare different techniques for preprocessing natural language and find those ones which impact on the building accurate classifiers. To this purpose we applied Bag-of-Words model (vector of unigrams), Bag-of-N-grams model (vector of bigrams and vector of trigrams) to represent text data in suitable numeric format. Bag-of-Words model of the data representation showed the best results for all methods and influenced in a positive way improving the accuracy.

The best performance was achieved by the Multinomial Naive Bayes method and Support Vector Machines (SVM) method with linear kernel. The best accuracy received by these methods equals 80.6%. The values of accuracy by Multinomial Naive Bayes method with using the Bag-of-Words model is in the range from 70.97% to 80.60% subject to sizes of data. The values of accuracy by SVM with using the Bag-of-Words model is in the range from 69.6% to 80.60% subject to sizes of data. These values of performance are higher than values obtained by the rest methods.

Keywords: sentiment analysis, machine learning, classification, Naive Bayes, SVM, neural networks, Weka, scikit-learn.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Lars Vidar Magnusson, for the useful comments, advice and support. This thesis work would be impossible without his proper guidance. And also I want to thank my friend Ksenia Dmitrieva for helping me to express myself in a correct way in English, who has supported me throughout the process.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Social media sites like Facebook and Twitter have become extremely popular since their appearance. Today millions of people share their impressions about the world with their friends and acquaintances in social media.

1.1 Motivation

In today's connected world, users can send messages in any time. However, social media is not only used as a casual tool for messaging and sharing private things and thoughts; it is also used by journalists, politicians and public figures, series of companies and universities who want to be more open to the public, share their thoughts and take an interest in opinion of persons. The active growth of the audience of social media on the Internet led to the formation of these resources as a new source of the people's mood and opinion. The tracking of citizens' reactions in social media during crises has attracted an increasing level of interest in the research community [73].

Researchers note that the billions of publications left by people monthly, can not be processed manually by holding public opinion polls. This fact highlights the need for automated methods of intellectual analysis of text information, what allows in a short time to process large amounts of data and to understand the meaning of user messages. This understanding of the meaning of messages is the most important and complex element of the automated processing. Use of modern technologies and methods of big data, using artificial intelligence, has already been helping researchers to automate the process of content analysis, in particular to collect data, to prepare, to manage and to visualize data. These innovations give the opportunity to conduct large-scale research and to monitor social media in real-time.

Existing sentiment analysis techniques occur from the fields of natural language processing, computational linguistics, text mining, and a range from machine learning methods to rule-based methods. Machine learning methods involve training of models on specific collections of documents. Recently, many researchers deal with the determination of sentiment of people in various data collected from social media. They have used well-known machine learning techniques for classification and clustering data. However, the comparison of machine learning techniques for sentiment classification has not been done before and so in this thesis we need compare existing techniques applied to sentiment analysis in social media.

We are to compare existing methods applied to sentiment analysis of data collected

from the social network Twitter, as according to related work the same method implementations can perform differently for different datasets. This work presents the discussion of the challenges which arises during sentiment analysis and classification. Also this thesis includes the review of experiments and researches which were done to solve the similar problems.

Going through the related work of what methods have been used and which of them are successful, we have discovered one main tendency: the performance of the specific method mainly depends on the dataset. It depends on the complexity of the data, if there is any positive, negative, neutral data in the datasets.

1.2 Research Question

Based on the observation discussed above we have arrived at the following primary research question.

RQ 1. *How do standard machine learning techniques applied to sentiment classification compare in social media data?*

Some techniques which are discussed in Chapter 2 are not generally implemented to work with text data. To apply those models, data should be modified, the text preprocessing should be performed. This leads to the second research question:

RQ 2. *Which preprocessing techniques are available for converting natural language text into suitable format?*

To answer these questions we need to perform the sets of experimentation using different machine learning techniques and natural language processing techniques.

1.3 Report Outline

The rest of the thesis is organised as follows.

Chapter 2 provides background information on social media and sentiment analysis. First, we define what is a sentiment analysis, provide some background required for further analysis with machine learning methods. The second part briefly describes the following machine learning models: Naive Bayes, Multinomial Naive Bayes, Decision Trees, in particularly Random Forest method, also Support Vector Machines (SVM) and Neural Networks (NN), in particularly Multilayer Perceptron Network.

The third chapter describes the related literature and case studies overview. What was done before the same problem. What was the challenges.

Experimental setup is described in Chapter 4. Also the forth chapter gives data description, analysis and preprocessing steps. Chapter 5 describes several experimentation sessions and their results. Chapter 6 provides a summary and discussion on the obtained results and presents suggestions for further work. Finally last chapter provides conclusion on the obtained results.

Chapter 2

Background

Since sentiment analysis in social media requires good knowledge of sentiment analysis, and methods, in this chapter a proper overview of all of these related concepts is provided. The first part of this chapter deals with some theory of social media and sentiment analysis. In the second part we discuss different methods for sentiment analysis: machine learning approaches, provide some mathematics, which becomes the basis of the experiments in Chapters 4, 5. After that, the evaluation measures are represented. And also we discuss methods of text representation.

2.1 Social media

Social media can be referred to as the "group of internet-based applications that build on the ideological and technological foundations of Web 2.0, and that allow the creation and exchange of user-generated content", as defined by Kaplan and Haenlein [40].

In recent years in addition to the leaders of the World Wide Web such as Facebook, Google+, LinkedIn and Twitter, there are new services for different groups of users: social network for students, the network for specific groups of professionals, communities of ethnic minorities, and even a special network for all the world's drinkers. This extends the scope to very different kinds of research from consumer preferences to psychological characteristics.

As follows from the Figure 2.1, in early 2015 Facebook retained the first place among social platforms, and also Twitter was in the top ten. According to the same study by Simon Kemp [41], more than 2 billion people worldwide are active users of social networks and blogs.

Facebook dominates the global social media landscape, claiming 1.366 billion active users in January 2015. Meanwhile, instant messenger services and chat apps continue to grow, with WhatsApp, WeChat, Facebook Messenger and Viber all reporting more than 100 million new monthly active users over the past 2014. Instant messenger services and chat apps now account for 3 of the top 5 global social platforms, and 8 instant messenger brands now claim more than 100 million monthly active users.

In Twitter, the number of monthly active users is 284 million in 2015. In 2016 the number of monthly active users exceeded 320 million [9].

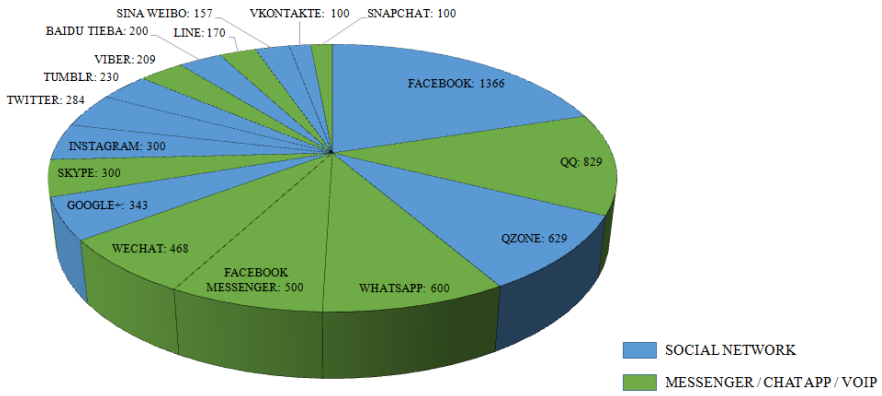


Figure 2.1: The number of monthly active user accounts of social media in 2015, in millions [41]

2.1.1 The specifics of research of the content in Twitter

Twitter is a realtime, highly social microblogging service that allows you to post short messages of 140 characters or less; these messages are called tweets. Unlike social networks like Facebook and LinkedIn, where a connection is bidirectional, Twitter has an asymmetric network infrastructure of "friends" and "followers". [59] Twitter is an important phenomenon from the standpoint of its incredibly high number of users, as well as its use as a marketing device and emerging use as a transport layer for third-party messaging services. It offers an extensive collection of APIs [10].

Analysis of information in Twitter is one of more interesting domains. There are several reasons. First, microblogging is a rich source of public information. Second, information in Twitter is open, has clear character, well-documented. Third, appearance of messages occurs, figuratively speaking, with the speed of thought, in real time. The most different sectors of society, reflecting the position of citizens of different countries, pass their opinions in Twitter. Fourthly, it is possible to trace the communication between individuals or communities through a number of mechanisms of Twitter.

Microblogging content analysis can help to evaluate changes in moods of many users, to reveal their political preferences, likes and dislikes, their choice in favor of one or another candidate during election campaigns. That is why the development of methodology for the analysis of Twitter messages got development in recent years.

Most often, the researchers used sentiment analysis. It can be used for political or sociological researches, for analysis of consumer preferences microblog users, and in other cases.

2.2 Sentiment Analysis

Sentiment analysis is an area of natural language processing and aims at determination of opinions, attitudes of a writer in the text or their attitude towards specific topics. *Sentiment* describes an opinion or attitude expressed by an individual, the opinion holder, about an entity, the target. *Attitudes* – "relatively enduring, affectively colored beliefs, preferences, and predispositions towards objects or persons (liking, loving, hating)"[62] –

are different from *emotions* – "brief episodes of synchronized responses (angry, sad, joyful, fearful, ashamed, proud)"[62] as a reaction to external influences. This distinguishes sentiment analysis from other problems such as emotion analysis where the general emotional state (influenced by various external factors) is of interest, and not the attitude towards a specific target. The degree and direction of sentiment (i. e., how positive or negative it is) is called its *polarity*.

The simplest and most common polarity scheme assumes two categories, positive and negative. These two categories constitute the extreme ends of a discrete or continuous scale. This definition covers most voting schemes used in practice, such as

- thumbs up/down (e.g., Facebook, YouTube),
- positive, neutral, negative (e.g., eBay), or
- star ratings (e.g., Amazon, IMDb).

Often, polarity is mapped to the $[-1, 1]$ interval, assuming that -1 is the most negative polarity possible, and 1 is the most positive one. There is some ambiguity regarding the center of the scale (0), which is commonly described as neutral. Note however that this can also mean a more or less balanced mix of positive and negative content [44]. It has been recognized that this data is difficult to assess even for humans [43], which is why data from this category is sometimes omitted from experiments to simplify the problem (e.g., studies by Speriosu et al. [66], Go et al. [31], da Silva et al. [27], Saif et al. [60], Blitzer et al. [20], Bakliwal et al. [17]).

2.2.1 Types of sentiment analysis

In today's research, many different views on automatic sentiment analysis exist, that leads to different tasks. The most prominent difference between them is the granularity of analysis. Sentiment analysis is performed on multiple linguistic levels.

At the *document level*, the task is to classify whether a whole opinionated document has a positive, negative or neutral sentiment.

At the *sentence level*, the task is to classify whether an individual sentence has a positive, negative or neutral sentiment.

At the *aspect level* (the entity level), the task is to classify the sentiment of individual sentences or phrases intended towards certain entities or aspects.

The goal of document-level sentiment analysis is to predict the overall polarity expressed in a document. Typically, the documents on which this type of analysis is performed are ones in which the author evaluates only a single entity, such as reviews of products, hotels, or movies. The task of predicting document-level polarity can be produced as a standard text classification problem. The problem can then be addressed using machine learning techniques, such as maximum entropy classification, Naive Bayes classifier (the research by Pang et al. [56]). There are several assumptions involved in the text classification approach. First, it is assumed that the whole text is concerned with a single target, namely the product that is the subject of the review. Second, the author is assumed to be the opinion holder.

Formally, the document-level sentiment classification task are defined the follows [49]. There is a set of documents D with opinions of writers, it determines whether each document $d \in D$ expresses a positive or negative opinion (or sentiment) on an object. Given a

document d which comments on an object o, determine the orientation oo of the opinion expressed on o, i. e., discover the opinion orientation oo on feature f in the quintuple (o, f, oo, h, t), where $f = o$ and h, t, o are assumed to be known or irrelevant.

Existing research on sentiment classification makes the following assumption. The document d (e.g., a product review) expresses opinions on a single object o and the opinions are from a single opinion holder h. This assumption holds for customer reviews of products and services. However, it may not hold for a forum and blog post because in such a post the author may express opinions on multiple products and compare them using comparative and superlative sentences.

The task of predicting the polarity of a sentence is also a problem of classification. Formally, this task is defined as follows. Given a sentence s, and two sub-task are performed: (1) subjectivity classification: determine whether s is a subjective sentence or an objective sentence, (2) sentence-level sentiment classification: if s is subjective, determine whether it expresses a positive or negative opinion. The quintuple (o, f, oo, h, t) is not used in defining the task of sentence-level classification. The first sub-task filters out those sentences which contain no opinion, and after we know what objects and features of the objects are talked about in a sentence, the second sub-task helps to determine whether the opinions on the objects and their features are positive or negative.

The task of predicting sentence-level polarity can be addressed using machine learning methods such as the Naive Bayes, decision trees, support vector machine and their combinations.

The aspect-level sentiment analysis task can be defined as follows. Identify aspects that have been commented on. For example, in the sentence[1], *"The picture quality of this camera is amazing,"* the aspect is "picture quality". Determine whether the opinions on the aspects are positive, negative or neutral.

An aspect can be defined as any object about which sentiment is expressed. Sentiment analysis of aspects does not focus on a single linguistic unit for analysis. Instead, all information about the aspect is collected and used for making a prediction. This task has been formalized as *fine-grained sentiment analysis* (e.g., the research by Yang and Cardie [74]) where the relation between opinions, targets, and sometimes opinion holders (writers) have to be recognized. In contrast to the problems discussed above, this task involves structured prediction comparable to other NLP problems such as semantic role labeling.

Liu [49] argues that analysis on the aspect level is superior to considering individual units (such as single phrases or sentences) as knowledge about the target is necessary for resolving ambiguities. Conversely, aspect-level sentiment analysis requires holder (writer) and target detection which leads to a significantly more complicated machine learning task.

Next, in this thesis we consider the sentence-level sentiment classification. The sentiment classification on document-level and on aspect-level are beyond the scope of this thesis.

2.2.2 Approaches to sentiment classification

The existing approaches to sentiment classification fall into two large categories:

1. Approaches based on lexicon and rules.

[1] Example taken from the Liu article Sentiment Analysis and Subjectivity[49]

2. Machine learning approach.

The rule-based approach uses a set of rules based on analysis of object domain that could explain and predict the polarity of the text (or the single sentence).

Rules tend to get increasingly complicated that lets to increase accuracy of results. Disadvantages are associated by the large amount of time and expertise needed to design such rules. Using of this approach for analysis of microblogging may be difficult due to noise data. The performance of rule-based approaches tends to be comparably robust across domains and texts, but it is typically inferior to the performance of machine learning methods of polarity classification [67].

Rule-based methods mostly rely on lexicons that list words and their associated sentiment scores. The sentiment scores of words in a text are typically combined (e.g., summed or averaged) in accordance with predefined rules and assumptions in order to obtain a text's overall sentiment score, which can be used as an indicator for the text's polarity.

A lexicon based approach uses an affective lexicon to derive the polarity of the examined text. Affective lexicons contain lists of words either divided by certain sentiment classes (e.g. positive, negative, neutral) or providing a single list of words each associated with a numerical value representing its polarity. The folows dictionaries (affective lexicons) can be used for the English language:

1. ANEW. Affective Norms of English Words (ANEW) is a set of normative emotional ratings for 1034 English words developed by Bradley and Lang [21] from the NIMH Center for Emotion and Attention (CSEA) at the University of Florida [1]. For each word in the dataset, there are scores for three dimensions of emotional assessment: valence (ranging from pleasant to unpleasant), arousal (ranging from calm to excited) and dominance (ranging from in-control to dominated). This dataset is a useful tool for emotion studies as well as for sentiment analysis.

2. WordNet. WordNet [52] is one of the largest lexical resource for English language which is extensively used in scientific research. Multiple words can form a synset, a set of words that may be used synonymously within a word sense. Additionally, synonymous relations are defined between synsets, leading to a taxonomy structure. These relations may be used for generalization over objects (e.g., cheese and bread are a type of food).

   WordNet 3.0 lists 117,798 nouns in 82,115 synsets. WordNet also contains words of other part-of-speech, such as adjectives and verbs, however they have much lower coverage and their taxonomies are relatively flat.

   According to the description from the project homepage [13],

   "WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual semantic and lexical relations. The resulting network of meaningfully related words and concepts can be navigated with the browser. WordNet is also freely and publicly available for download. WordNet's structure makes it a useful tool for computational linguistics and natural language processing."

   Figure 2.2 shows a graph representation of WordNet synsets and relations between them.

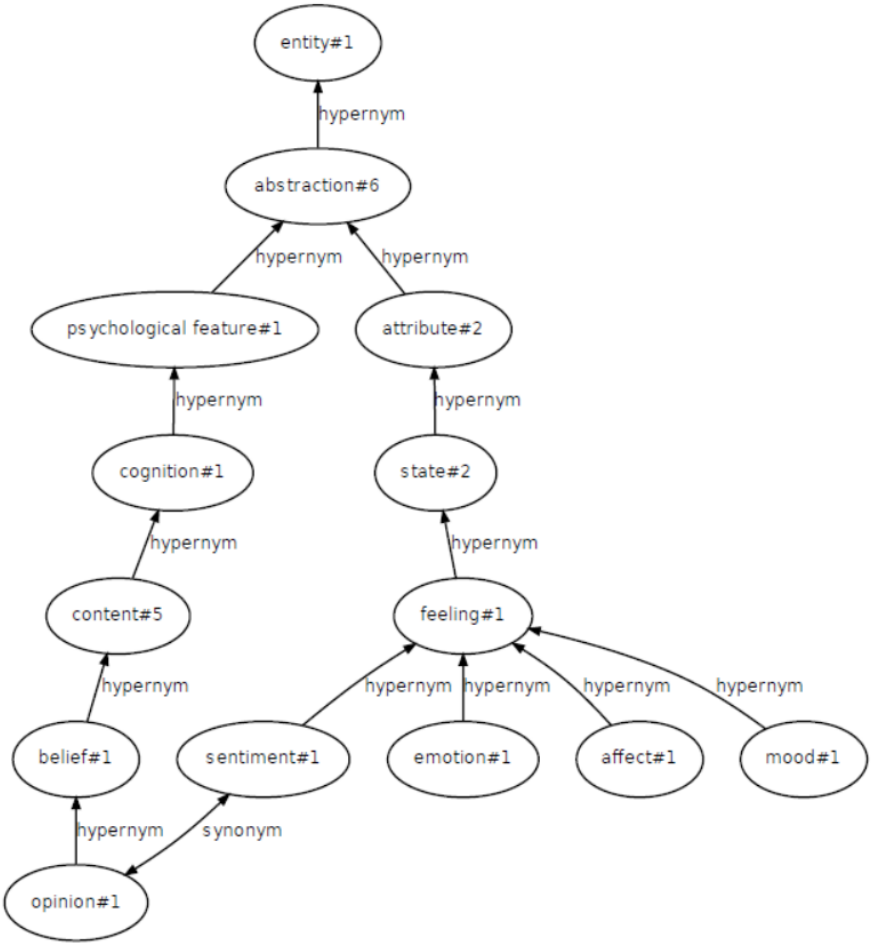


Figure 2.2: An example of a graph visualization of WordNet[54].
Nodes represent synsets, edges represent relations between synsets.

3. SentiWordNet. SentiWordNet is a lexical resource for sentiment analysis developed by Baccianella et al. [16]. It was constructed by automatic annotation of WordNet synsets. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity. [8]

Machine learning methods involve training of models on specific collections of documents (i.e., corpora) by means of mostly supervised methods that exploit patterns in vector representations of natural language text. For collections of documents or datasets the class attribute values for the dataset are known. This data is called *training* data. The training data consists of a set of training examples. To evaluate the performance of the learned model after training is finished, one applies it to a different set of data, known as test data. Sometimes, people also use part of the whole dataset as validation dataset for model selection, i.e., select from all the models obtained from the training process the one model with the best performance on the validation dataset.

Machine learning methods can be divided into classification, regression, clustering. When the class attribute is discrete, it is called classification; when the class attribute is

continuous, it is regression. In clustering, the data is often unlabeled. Thus, the label for each instance is not known to the clustering algorithm.

Considering the task of sentiment analysis in social media in this thesis, classification is used. Classification methods such as decision trees, naive Bayes classifier and other are considered in the Section 2.3.

Many researchers ([57], [67]) have used a combination of the two approaches: machine learning and approache based on affective lexicon and rules. The reason is that the hybrid approach in practice shows the best results.

In this thesis, machine learning classification methods are considered. Since these methods show good results in the sentiment analysis of social media (blog-sites and reviews). The approaches described above such as approaches based on rules and affective lexicon are beyond the scope of this thesis.

2.3 Machine Learning Techniques

We begin this section with a formal definition to the text classification problem (Section 2.3.1). Then, we consider different techniques which are used for sentiment analysis in social media.

2.3.1 The Classification Problem

In general, the problem of text classification is defined as follows [50]. Given a description $d \in X$ of a document, where X is the document space, a fixed set of classes $C = \{c_1, c_2, \ldots, c_m\}$ and a training set D of labeled documents $\langle d, c\rangle$, where $\langle d, c\rangle \in X \times C$ using a learning algorithm Γ, we need learn a classifier or classifier function $\Gamma(D) = \gamma$ that maps documents to classes: $\gamma : X \to C$.

For the sentiment analysis of social media, the set C consists of three classes
$C = \{positive, negative, neutral\}$.

2.3.2 Naive Bayes

Among many methods that use the Bayes theorem, the naive Bayes classifier is the simplest one [76]. Given two random variables X and Y, Bayes theorem states that

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{2.1}$$

In Naive Bayes classifier, Y represents the class variable and X represents the instance feature. Let X be $(x_1, x_2, \ldots, x_m)$, where x_i represents the value of feature i. Let $(y_1, y_2, \ldots, y_n)$ represents the value the class attribute Y can take. Then, the class attribute value of instance X can be calculated by measuring

$$\arg max_{y_i} P(y_i|X). \tag{2.2}$$

Based on the Bayes theorem,

$$P(y_i|X) = \frac{P(X|y_i)P(y_i)}{P(X)} \tag{2.3}$$

Note that $P(X)$ is constant and independent of y_i, so we can ignore the denominator of Equation 2.3 when maximizing Equation 2.2. The Naive Bayes Classifier also assumes conditional independence to make the calculations easier; that is, given the class attribute value, other feature attributes become conditionally independent. This condition, though unrealistic, performs well in practice and greatly simplifies calculation.

$$P(X|y_i) = \prod_{j=1}^{m} P(x_j|y_i). \tag{2.4}$$

Substituting $P(X|y_i)$ from Equation 2.4 in Equation 2.3, we get

$$P(y_i|X) = \frac{\prod_{j=1}^{m} P(x_j|y_i)P(y_i)}{P(X)}. \tag{2.5}$$

where m is the total number of words in Y.

Naive Bayes algorithm is also called the probabilistic method. In [25] Naive Bayes algorithm and binary keyword were simultaneously used to produce a single dimensional degree of sentiment entrenched in tweets from twitter network.

There are two different ways we can set up the Naive Bayes classifier: the multinomial model and the multivariate Bernoulli model. In this thesis the multinomial Naive Bayes model is considered.

Multinomial Naive Bayes

The multinomial Naive Bayes is a probabilistic method [50]. The probability of a document d being in class c is computed as

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c) \tag{2.6}$$

where $P(t_k|c)$ is the conditional probability of term t_k occurring in a document of class c. $P(t_k|c)$ can be interpreted as a measure of how much evidence t_k contributes that c is the correct class. $P(c)$ is a prior probability of a document occurring in class c. If a document's terms do not provide clear evidence for one class versus another, we choose the one that has a higher prior probability. $\langle t_1, t_2, \ldots, t_{n_d} \rangle$ are the tokens in d that are part of the vocabulary we use for classification and n_d is the number of such tokens in d.

The main goal of classification is to find the best class for the document. The best class in Naive Bayes classification in the most likely or maximum a posteriori class c_{map} (Equation 2.7).

$$c_{map} = \arg max_{c \in C} \hat{P}(c|d) = \arg max_{c \in C} \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c) \tag{2.7}$$

In Equation 2.7, $\hat{P}$ are used because the true values of the parameters $P(c)$ and $P(t_k|c)$ are not known, but they can be estimated from the training set.

In Equation 2.7, many conditional probabilities are multiplied, one for each position $1 \leq k \leq n_d$. This can result in a floating point underflow. It therefore requires to perform the computation by adding logarithms of probabilities instead of multiplying probabilities. The class with the highest log probability score is still the most probable; $\log(xy) =$

$\log(x) + \log(y)$ and the logarithm function is monotonic. Hence, the maximization that is actually done in most implementations of Naive Bayes is:

$$c_{map} = \arg max_{c \in C} \left[\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c) \right]. \tag{2.8}$$

Equation 2.8 can be interpreted as follows. Each conditional parameter $\log \hat{P}(t_k|c)$ is a weight that indicates how good an indicator t_k is for c. Similarly, the $\log \hat{P}(c)$ is a weight that indicates the relative frequency of c. More frequent classes are more likely to be the correct class than infrequent classes. The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class, and Equation 2.8 selects the class for which we have the most evidence.

The multinomial Naive Bayes classifier is used in the studies [18], [32].

2.3.3 Support Vector Machine

Support vector machines (SVM) is a blend of a linear modeling and instance based learning in a high-dimensional space. SVM can be applied for those problems when data cannot be separated by line. Support vector machines use nonlinear mapping – it transforms the instance space into another space which has higher dimension than the original one. In this case line in the new space can be represented as a linear boundary in the instance space. Support vector machines were originally developed for classification problems.

Kernel concept gave rise to support vector machines. Kernel is a function which fulfil mapping of a nonlinear data to a new space.

Kernel function K is an inner product $\Phi(x) \bullet \Phi(y)$ between the images of two data points x and y:

$$K(x,\ y) = \Phi(x) \bullet \Phi(y) \tag{2.9}$$

where $\Phi(x)$ and $\Phi(y)$ are mapping operators.

The feature, that kernel function is formulated as an inner product, gives an opportunity to replace scalar product with some choice of kernel [24].

The problem of finding parameters of SVM corresponds to a convex optimization problem, which means that local solution is global optimum as well.

A classification task usually involves separating data into training and testing sets. Each instance in the training set contains one "target value" (i. e. the class labels) and several "attributes" (i. e. the features or observed variables). The goal of SVM is to produce a model (based on the training data) which predicts the target values of the test data given only the test data attributes.

SVM for classification is used to find a linear model of the following form:

$$y(x) = w^T x + b \tag{2.10}$$

where x is input vector, w and b are parameters which can be adjusted for a certain model and estimated in an empirical way. In simple linear classification the task is to minimize a regularized error function given by Equation 2.11.

$$C \sum_{n=1}^{N} \xi_n + \frac{1}{2} \|w\|^2 \tag{2.11}$$

The constraints are $\xi_n \geq 0,\ \forall\ n = 1,\ \ldots\ ,\ N,$ and

$$y(w^T x + b) \geq 1 - \xi_n \tag{2.12}$$

Figure 2.3 illustrates an example of a linear SVM that has been trained on examples from two classes. Here the SVM constructs a separating hyperplane and then tries to maximise the "margin" between the two classes. To calculate the margin, the SVM constructs two parallel hyperplanes, one on each side of the initial one. These hyperplanes are then "pushed" perpendicularly away from each other until they come in contact with the closest examples from either class. These examples are known as the support vectors and are illustrated in bold in Figure 2.3.

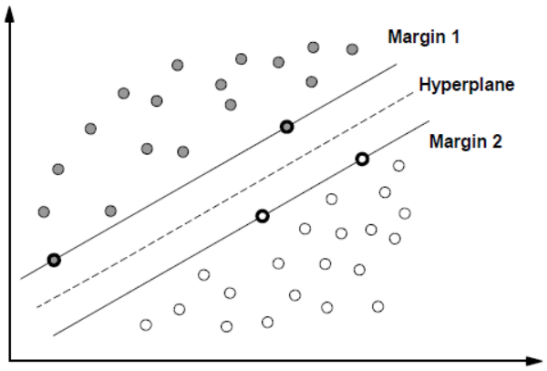


Figure 2.3: Support Vector Machine: Classification

Kernel Functions

There are many forms of kernel functions. In research [35] the four following basic kernels are described:

- linear kernel,
- polynomial kernel,
- radial basis kernel,
- sigmoid kernel.

Linear kernel is represented as

$$K(x, y) = x^T y + c \tag{2.13}$$

where x and y are vectors in input space, and c is a free parameter [19].

Polynomial kernel is given by

$$K(x, y) = (x^T y + c)^d \tag{2.14}$$

where x and y are vectors in input space, d is a dimension of a new space and c is a free parameter [19].

Radial basis kernel is represented as

$$K(x, y) = exp\left(\frac{-\|x - y\|^2}{2\sigma^2}\right) \tag{2.15}$$

where σ is a free parameter [19].

Sigmoid kernel is given by

$$K(x_i, y) = \tanh(\gamma x^T y + c) \tag{2.16}$$

where γ and c are kernel parameters [19].

2.3.4 Decision Trees

Decision trees can be adapted to almost any type of data, therefore it is one of the most widely used in machine learning algorithms. They are a supervised machine learning algorithm that divides its training data into smaller and smaller parts in order to identify patterns that can be used for classification. The data is then presented in the form of logical structure similar to as Figure 2.4 that can be easily understood without any statistical knowledge. The algorithm is particularly well suited to cases where many hierarchical categorical distinctions can be made.

They are built using a heuristic called recursive partitioning. This is generally known as the divide and conquer approach because it uses feature values to split the data into smaller and smaller subsets of similar classes. The structure of a decision tree consists of a root node which represents the entire dataset, decision nodes which perform the computation and leaf nodes which produce the classification. In the training phase the algorithm learns what decisions have to be made in order to split the labelled training data into its classes.

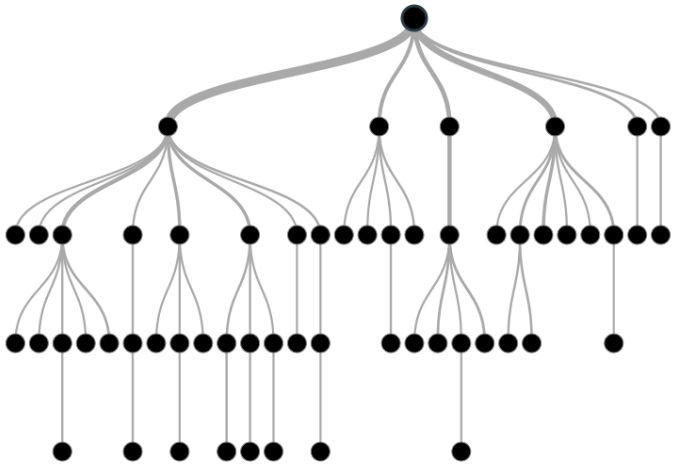


Figure 2.4: Decision tree structure [30]

In order to classify an unknown instance, the data is passed through the tree. At each decision node a specific feature from the input data is compared with a constant that was

identified in the training phase. The computation which takes place in each decision node usually compares the selected feature with this predetermined constant, the decision will be based on whether the feature is greater than or less than the constant, creating a two way split in the tree. The data will eventually pass through these decision nodes until it reaches a leaf node which represents its assigned class.

There are many different implementations and variations of the decision tree algorithm, such as Random Forest, J48 method which is a Java implementation of the C4.5 algorithm.

Random Forests

Ensemble learning focuses on techniques to combine the results of different trained models in order to produce a more accurate classifier. Ensemble models generally have considerably improved performance than that of a singular model. The random forest algorithm is an example of an ensemble method which was introduced by Breimanb [22], it is quite a simple algorithm but despite its simplicity it can produce high performance in terms of classification. The basic structure of the random forest can be seen in Figure 2.5 below.

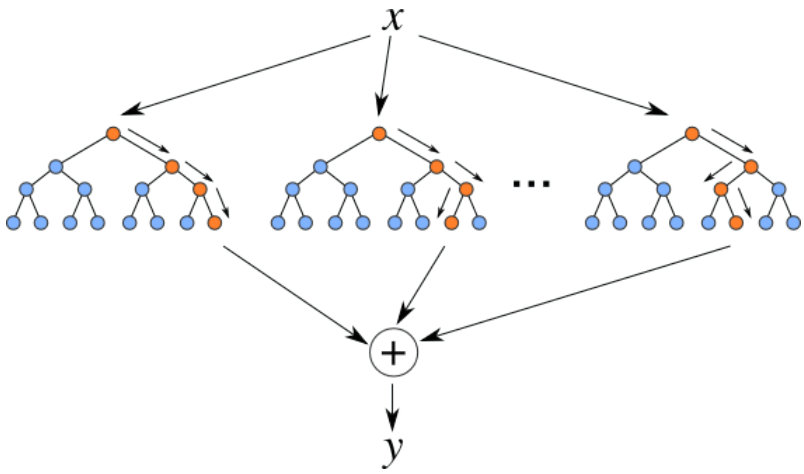


Figure 2.5: Random forest structure [2]

Random forests are constructed by combining a number of decision tree classifiers, each tree is trained using a bootstrapped subset of the training data. At each decision node a random subset of the features is chosen and the algorithm will only consider splits on those features. The main problem with using an individual tree is that it has high variance that is to say that the arrangement of the training data and features may affect its performance. Each individual tree has high variance but if we average over an ensemble of trees we can reduce the variance of the overall classification. Provided that each tree has better accuracy then pure chance, and that they are not highly correlated with one another the central limit theory states that when they are averaged they will produce a Gaussian distribution. The more decisions that are averaged the lower the variance becomes. Reducing the variance will generally increase the overall performance of the model by lowering the overall error.

The random forest algorithm finds a use for the implementation for scikit-learn tools [6], in Weka platform [12].

2.3.5 Artificial Neural Networks

Artificial Neural Networks is a machine learning technique taking its origin from the biological neural networks – a network of a nerve cells of a human brain. ANN is a system of interconnected and interacting neurons. There is an input, activation and an output function for each neuron however the process of training of a neural network is a task of finding the weights for links between neurons.

Neural networks (NNs) is a state-of-the-art technique and one of the most effective machine learning methods. An ongoing research shows that it is a powerful tool for pattern recognition and classification and proposed to be an universal approximator which can fit any function however they works well not only as a fitting tool but also has a good ability to generalize.

NNs can be grouped into two major categories:

- feedforward networks,
- feedback (recurrent) networks.

Feedforward networks (multilayer perceptron and radial basis funstion) are mostly being used for classification and function approximation problems. In the feedforward nets there are no loops in the network connections – they go only in one direction – and neurons are organized into layers. In feedback nets one or more loops may exist.

There are also fuzzy neural networks which use fuzzy logic [46], dynamic neural networks and so on.

Multilayer Perceptron Network

A multilayer perceptron is a feedforward artificial neural network model that maps sets of input data onto a set of appropriate outputs.

According to MathWork Documentation [3], feedforward neural networks can be used for any kind of input-output mapping and a feedforward network with one hidden layer and enough neurons in the hidden layers can fit any finite input-output mapping problem.

A general architecture of feedforward neural networks is presented on Figure 2.6. Here, hidden layer(s) of a neural network consists of primitive units which is perceptrons. Perceptron units takes a vector of input variables x_t and calculates it's linear combination. If the result is greater that some threshold than the output $y(x_1, x_2, \ \dots \ , \ x_t)$ of a perceptron is 1, if less than -1:

$$y(x_1, x_2, \ \dots \ , \ x_t) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_t x_t > 0, \\ -1 & \text{otherwise} \end{cases} \tag{2.17}$$

It also can be representing as a decision making in the multidimensional space of instances: the output for instances lying by one side of a hyperplane is 1 and for instances lying by the other side -1.

Learning a perceptron is a process of finding value of the weights $w_0, w_1, w_2, \ \dots \ , w_t$. One of the possible ways is starting with random weights and then each time when perceptron misclassifies the instance update the weights according to some update rule. This process is repeated iteratively until the model classify all the instances correctly.

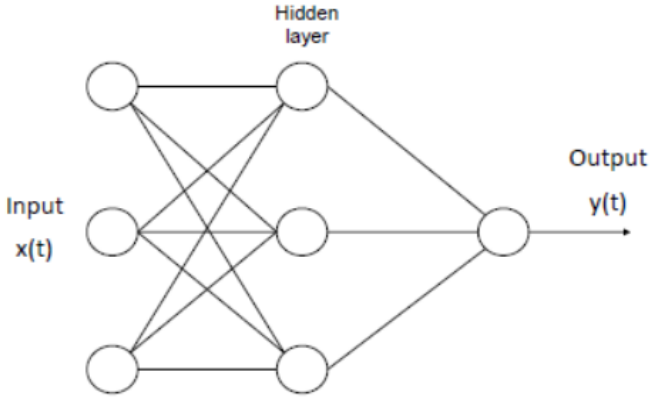


Figure 2.6: Feedforward neural network structure

2.4 Evaluation Measures

For the evaluation of classification results, we address to well-known measures from information retrieval [50]. All evaluation measures presented in the current section rely on some basic counts on a test collection of data D.

The basic measurements are the counts of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) with respect to each class c of each instance. These depend on whether the class predicted by the classifier matches the expected prediction, i.e. the true class, as shown in Table 2.1.

		actual	
		c	$\neg c$
predicted	c	$TP^{(c)}$	$FP^{(c)}$
	$\neg c$	$FN^{(c)}$	$TN^{(c)}$

Table 2.1: Confusion matrix of actual and predicted class

We define $TP^{(c)}$, $TN^{(c)}$, $FP^{(c)}$, and $FN^{(c)}$ to denote the number the respective events occurred in the collection for a class c. Based on these count statistics, we define our evaluation measures.

The most basic measure is accuracy (Acc). Here, we simply measure the ratio of correctly classified instances on the collection D (Equation 2.18).

$$Acc = \frac{\sum_{c \in C} TP^{(c)}}{|D|} \tag{2.18}$$

In the case, when we have one instance in the collection, the accuracy can be calculated by a simplified equation (Equation 2.19).

$$Acc = \frac{TP^{(c)} + TN^{(c)}}{TP^{(c)} + FP^{(c)} + FN^{(c)} + TN^{(c)}} \tag{2.19}$$

Accuracy is a good measure when classes are distributed uniformly in the collection. However, as class imbalances grow more pronounced, high accuracy might be attained by a classifier that has a bias towards the majority class.

Precision and recall are often used as an alternative, providing a more detailed analysis of the classifier's behavior with respect to each class c. *Precision* $\left(P^{(c)}\right)$ measures the relative frequency of correctly classified examples that were predicted to belong to c (Equation 2.20):

$$P^{(c)} = \frac{TP^{(c)}}{TP^{(c)} + FP^{(c)}} \tag{2.20}$$

Recall $\left(R^{(c)}\right)$ measures the relative frequency of correctly classified examples among the set of examples whose correct class is c (Equation 2.21):

$$R^{(c)} = \frac{TP^{(c)}}{TP^{(c)} + FN^{(c)}} \tag{2.21}$$

The harmonic mean of precision and recall is called the *F-measure*. In this thesis, we use the balanced F-measure, or F_1 measure, i. e. precision and recall are weighted equally (Equation 2.22):

$$F_1^{(c)} = \frac{2 \cdot P^{(c)} \cdot R^{(c)}}{P^{(c)} + R^{(c)}} \tag{2.22}$$

In contrast to the arithmetic mean in the case when $Precision = 0$ and $Recall = 1$ (or vice versa, $Precision = 1$ and $Recall = 0$), the harmonic mean would be equal zero, and the arithmetic mean would be equal 0.5. The harmonic mean is always less than or equal to the arithmetic mean and the geometric mean. When the values of two numbers (precision and recall) differ greatly, the harmonic mean is closer to their minimum than to their arithmetic mean, see Figure 2.7.

The measures have been proposed and recommendations made by different authors [38], [53], [60]. According to some researchers [26], [27], the harmonic mean is more important measure since F_1 has a value of 1.0 when precision and recall are both perfect, and approaches zero when precision or recall are poor.

The accuracy, precision, recall and F-measure metrics are also used for the sentiment classification task.

2.5 Models of the vector representation of text data

A vector representation of text data (word representation) lies at the core of machine learning methods , assigning to each word of the text collection a mathematical object, is often a vector of real numbers [72]. Approaches to represent the text as vectors are tested and compared by researchers to identify the capabilities of different models to solve specific problems related to the text processing.

All instances from the text collection (the training set and the test set) are n-dimensional feature vectors. The choice of features directly affects the quality of the trained model and thus the classifier performance.

Next, let's consider several models of text representation.

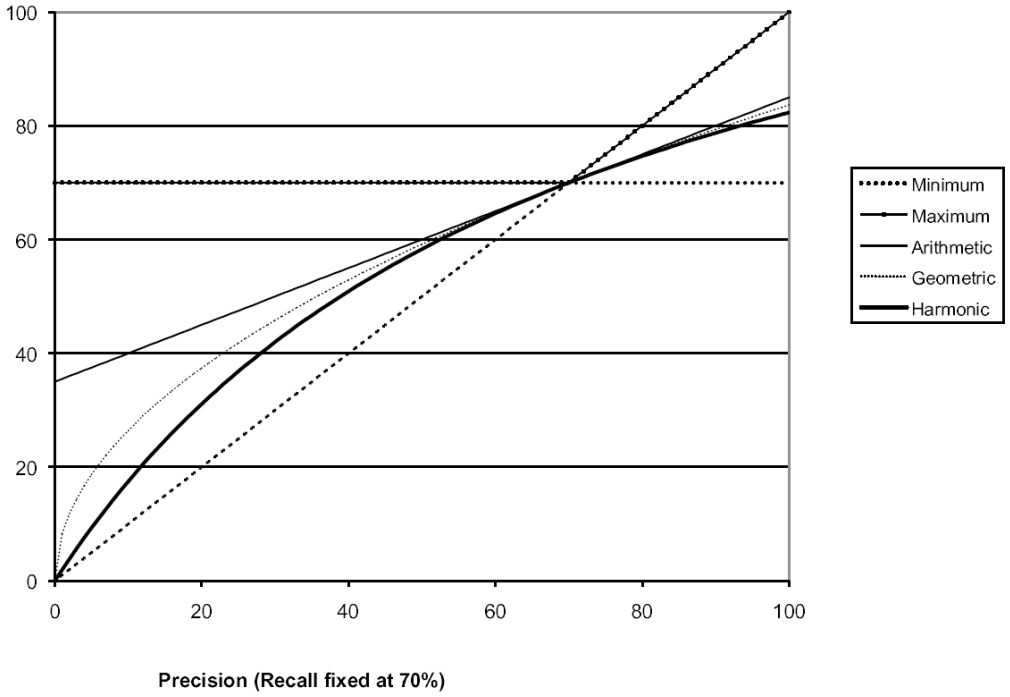


Figure 2.7: Graph comparing the harmonic mean to other means.
The graph shows a slice through the calculation of various means of precision and recall for the fixed recall value of 70%. The harmonic mean is always less than either the arithmetic or geometric mean, and often quite close to the minimum of the two numbers. When the precision is also 70%, all the measures coincide [50]

2.5.1 Bag-of-Words

A simple and popular approach for representing texts is to assume that word order does not matter. We interpret a document d_i as a set of its words $w \in d_i$ and ignore the order in which they occurred. This approach is called as the *bag-of-words* model, since we can consider the process as taking all words from the text and throwing them in a bag, losing sequence information in the process. We obtain the binary bag-of-words model through the following feature function (Equation 2.23):

$$f_i(X) = \begin{cases} 1 & \text{if } d_i \text{ contains word } w_i, \\ 0 & \text{else.} \end{cases} \tag{2.23}$$

The bag-of-words representation assumes that it is enough to use individual words as indicators. Thus, the sentence is represented as vector

$$d_i = (w_{i1}, w_{i2}, \ldots, w_{ij}, \ldots, w_{in})$$

where w_{ij} is the weight of token w_i in the sentence d_i, n is number of all tokens in the collection $|D|$ (the corpus).

The methods of defining weight of terms (tokens) are follows [50].

1. One common method of defining weight of token is to use binary attributes corresponding to word occurrence (Equation 2.23)

2. Term Frequency (TF) – the value of w_{ij} (weight of token) corresponds to the frequency of occurrence of w_i in the sentence d_i (Equation 2.24).

$$w_{ij} = tf(w_i,\ d_i) = \frac{n_i}{\sum n_k} \tag{2.24}$$

3. TF-IDF (Inverse Document Frequency)

$$idf(w, D) = \log \frac{|D|}{|(d_i \supset w_i)|} \tag{2.25}$$

$$w_{ij} = tfid(w_i,\ d_i,\ D) = tf(w_i,\ d_i) \times idf(w, D) \tag{2.26}$$

where $|D|$ – total number of documents in the corpus, $|(d_i \supset w_i)|$ – number of documents where the token w_i appears.

2.5.2 Bag-of-N-grams

In natural language processing (NLP), the n contiguous sequence of items in the text are together called a *n-gram*. The items can be phonemes, syllables, letters, words or base pairs according to the application. The n-grams typically are collected from a text corpus. For $n = 1$, the n-gram is called "unigram"; for $n = 2$, the n-gram is called "bigram", for $n = 3$, the n-gram is called "trigram", for $n > 3$, we simply replace the letter n by its numerical value, such as 4-gram, 5-gram, etc. A vector of unigrams is often called the Bag-of-Words model.

Consider the sentence "Jane likes coffee and tea". That can be represented as a vector of unigram [$Jane$; $likes$; $coffee$; and; tea]. Besides, this sentence can be represented as a vector of bigram [[$Jane\ likes$]; [$likes\ coffee$]; [$coffee\ and$]; [$and\ tea$]].

The character n-gram (or bag of character n-grams) is n consecutive characters of text. For example, consider a word "word", the character bigrams will be as follows: [$\_w$, wo, or, rd, $d\_$]. The character trigrams of the same words will be: [$\_wo$, wor, ord, $rd\_$]. Such a vector model used in the studies [15], [39] and shows good results.

Methods of defining the weight of a term in Bag-of-N-grams feature vector are similar to methods of defining the weight of a term the Bag-of-Words feature vector: binary frequency, Term Frequency (TF), and Inverse Document Frequency (TF-IDF), we described in Chapter 2.5.1. The elements of a feature vector are binary values which indicate the presence of the corresponding n-grams; alternatively, they can be integers which indicate the frequencies of the corresponding n-grams.

2.5.3 Part-of-Speech tagging

Part-of-Speech tagging is a basic form of syntactic analysis which has many applications in NLP. Part-of-Speech (POS) tagging is the process of assignment each token a tag that corresponds to its part-of-speech tag, such as a verb, a noun, adjective, etc. The list of POS tags may vary for different languages; there are different lists of tags even for the same language. In English, the most commonly used POS tags list is the one that used

in the Penn Treebank Project [5]. List of part-of-speech tags used in the Penn Treebank Project consists of 36 tags [61].

A document d_i is interpreted as a set of its words $w \in d_i$ where for each word matches POS tag from list of POS tags. The value of each POS tag represents their frequencies (0 or 1).

The Part-of-Speech tagging features can be denoted by $w_{POS} \in D^P$, where P is the number of all POS tags. A POS feature vector is a P-dimensional binary vector.

2.6 Summary

Sentiment analysis of social media is a vast domain requiring some necessary background which we have tried to cover in this chapter. We have started with the introduction in social media and sentiment analysis. In the second part of this chapter several machine learning approaches have been described. After, the evaluation measures are represented. And also we have described methods of text representation. As for machine learning methods from all the variety of them only those methods are described which were used in the experimentation session.

Chapter 3

Related Work

In this chapter we introduce our findings on research and related work regarding sentiment analysis. The study of related works basically presents how we prepared ourselves for carrying out experiments, explains why we have chosen particular methods in this work and gives some remarks regarding methods in order to better understand their work and results.

3.1 Related Work about Sentiment Analysis

Sentiment analysis is a vast domain which requires the study of related work done as well as a good knowledge of theoretical background (Chapter 2). In this section we describe relevant works in sentiment analysis as a problem domain.

The measurement of public sentiment in real-time has always been a non-simple task. If to use the traditional approaches, the researchers have to perform a large amount of survey of a significant number of people about their attitudes to a particular subject. The correct selection, the attraction of a significant number of people for carrying out surveys must be done; the researchers have to make a questionnaire and spend a considerable amount of time and money to carry out all procedures.

But since the number of users of social media grows, as we have seen in Chapter 2 in the Figure 2.1, social media and blogs can become a valuable source of information, if there is an effective method and an equipment for their research. Most often, Twitter or Facebook is selected as the content source.

Some research works have been carried out on sentiment in the microblog domain. Shamma et al. [63] examined a variety of aspects of debate modelling using Twitter, and annotated corpus of 3,269 tweets posted during the presidential debate on 2008 between Barack Obama and John McCain. Later, Diakapolous and Shamma [29] used manual annotations to characterise the sentiment reactions to various issues in a debate between John McCain and Barack Obama in the lead up to the US Presidential election in 2008, finding that sentiment is useful as a measure to identify controversial moments in the debate. In these studies, Twitter proved to be an effective source of data for identifying important topics and associated public reaction.

Thelwall et al. [70] assume that the emotions expressed by commenters reflect their feelings or invoke any surface emotions in readers, hence may be selected for their social role, for example, as part of a performance, informal ritual, or exchange. The specific questions of their study address the role of gender and age in emotion within social network

public comments, using MySpace. In particular, the following questions have been raised: "How common are positive and negative emotions in social network comments? Are there gender and age differences in the extent to which emotions are expressed in public MySpace comments?". While the measurement of emotion with any instrument is problematic [51] and human perception is inherently variable, the differences suggest that the classification of emotion from short comments is intrinsically difficult and often without a clear correct answer. Hence, the results for the overall occurrence of emotion and gender differences are subjective and cannot give definitive answers to the questions, particularly the first question. However, the answer on the second question is that two thirds of the comments expressed positive emotion, but a minority (20%) contained negative emotion, confirming that MySpace is an extraordinarily emotion-rich environment. Females are likely to give and receive more positive comments than are males, but there is no difference for negative comments. It is thus possible that females are more successful social network site users partly because of their greater ability express a positive affect.

Stahl et al. [14] defined that there are a number of research issues and challenges facing the realisation of utilising data mining approaches in social network analysis, and it could be identified as follows:

1. Linkage-based and Structural Analysis. – This is an analysis of the linkage behaviour of the social network so as to determine relevant nodes, links, communities and imminent areas of the network.

2. Static Analysis and Dynamic Analysis. – In static analysis, it is presumed that social network changes gradually over time and analysis on the entire network can be done in batch mode. Conversely, dynamic analysis of streaming networks like Facebook and YouTube are very difficult to carry out. Data on these networks are generated at high speed and capacity.

Khanaferov et al. [42] have selected healthcare informatics to demonstrate the significance of data for a complex domain. They focused on mining of public Twitter data for information relevant to obesity and health. Their main goal was to demonstrate a practical approach for solving an alarming healthcare issue through a systematic, computational approach concentrated on mining useful patterns out of public data. Due to the random nature of raw data and the exclusion of a training set of data, it was decided to use clustering since the learning process was unsupervised. Clustering has been considered to be the only possible way to find out patterns because it arranged similar sets of elements near one another for grouping. As a consequence of the uncertainty presented in the data collected, density based clustering algorithm was selected. Density-based spatial clustering of applications with noise (DBSCAN) algorithm is a type of density based clustering algorithm and was chosen to implement clustering for this study. The output of the algorithm was a set of clusters which were plotted on a 4 dimensional space where a vector of four elements was used to determine the location of any point in that space. In future these data can be used for visualization. In conclusion for this case study it was defined that tweets coming out of Europe and United States are associated with negative sentiment. In contrast, South Asia, Central Africa and Canada seem to have large clusters associated with positive sentiment.

Lipizzi et al. [48] presented a methodology combining social network and semantic analysis to extract in an automated way the data from Twitter streams created in a short time window surrounding the launch of a new product. They applied the proposed

method to two cases of new product launches that were executed in the Fall of 2013 by Apple and Samsung. The proposed method can support human analysts in the collection and interpretation of social media generated data through a variety of analytical tools including semantic and topological metrics as well as visualization of concept and social maps. The methodology is consisted in the following steps:

1. Selection of a triggering event.
2. Data collection.
3. Pre-processing.
4. Concept map extraction from the conversation.
5. Analysis of the map.

The method has been tested using two case studies in order to carry out a comparison of conversations generated by two products independently launched roughly at the same time by two well-known, competing brands. The results showed that there are significant differences in the structure of these conversations as they develop in time and authors supposed that these differences can be informative about the likelihood of early adoption and subsequent market success. In their study they also proposed a theoretical perspective to the analysis of social media based on conversational analysis studies and proposed that conversational analysis theories and methodological tools can offer an interesting base to advance our understanding of the processes of creation of content through social media as well as to empower our analytical capabilities to extract meaningful information from the chaotic and abundant flow of these data.

In 2014 the staff of the research department of the company Facebook held a large-scale experiment on manage emotions of users. For 690 thousand people the positive or negative publications were deliberately hidden. This is led to the one that the news feed of particular user consisted entirely of either positive messages either completely negative. As a result, the users themselves began to leave only positive or negative posts, respectively, thereby adopting the mood of other users via their publication [45].

3.2 Naive Bayes in Sentiment Analysis

Some of the major work in the field of sentiment analysis using the Naive Bayes was carried out by Pak and Paroubek [55]. The training data was collected using the assumption the emoticons contained in text represented the overall sentiment in that text. Using this assumption a large quantity of training data was automatically collected. This study used an ensemble of two different Naive Bayes classifiers; one trained using the presence of unigrams while the second used part of speech tagging. When the two classifiers were combined they produced an accuracy of 74%.

Pang et al. [56] used a single Naive Bayes classifier on a movie review corpus to achieve similar results as the previous study. Multiple Naive Bayes models were trained using different features such as part of speech tagging, unigrams, and bigrams. They achieved a classification accuracy of 77.3% which was considered a high performance of the Naive Bayes classifier on that domain.

3.3 Support Vector Machines in Sentiment Analysis

Researchers Ritterman et al. [58] used Twitter data to ascertain public sentiment and to inform prediction model markets. Their approach also implements a SVM-based algorithm used to analyze microblog messages about a particular topic in order to forecast public sentiment. The method was applied to microblog messages about an influenza pandemic and the results were compared with prediction market data from an independent source. Their work suggests that social media data can be used as a "proxy" for public opinion.

Java [37] have developed an application called BlogVox, to retrieve opinions from the blogosphere about a given topic. After pre-processing to remove spam and superfluous information, BlogVox uses a SVM to determine whether or not a blog post expresses an opinion. This differs from topic detection in that the data miner is interested in how people feel about a particular topic versus the topic itself.

3.4 Decision Trees in Sentiment Analysis

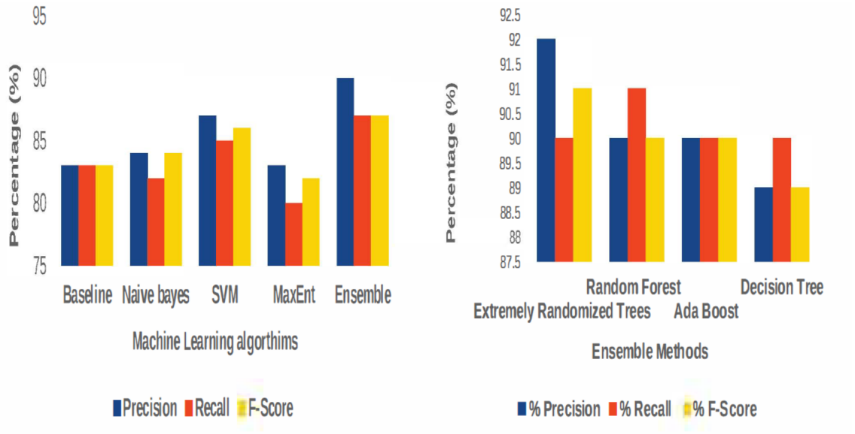
Study in the field of sentiment analysis using the Decision tree algorithm was carried out by Castillo et al. [23]. The studies main focus was on accessing the creditability of tweets posted on Twitter but there was also secondary focus on sentiment analysis. A decision tree was implemented using the J48 algorithm to classify sentiment in the twitter dataset. By training the algorithm with hand annotated examples the algorithm produced an accuracy of 70%.

Tsutsumi et al. [71] study implemented a weighted voting random forest on a movie review database. A scoring criterion was used to appoint a weighted vote to each random tree in the forest. Using this method the algorithm produced an accuracy of 83.4% on a dataset of 1400 reviews.

Kanakaraj and Guddeti [38] proposed extracting data for the purpose of analyzing the mood of the society on a particular news from Twitter posts. To increase the accuracy of classification they decided to include natural language processing techniques (NLP) especially semantics and word sense disambiguation. Various Machine Learning algorithms are widely used to solve the classification problems. "Ensemble methods" in machine learning, combines the effect of multiple machine learning algorithms on the given problem set to obtain a better predictive power than its constituent algorithms by separately. Kanakaraj and Guddeti have analyzed the performance of Decision Tree, Random Forest, Extremely Randomized Trees and Decision Tree regression with Ada Boost Classifiers on Twitter sentiment analysis. Experiments were conducted to compare the performance of Ensemble method against other machine learning algorithms like SVM, Baseline, MaxEntropy and Naive Bayes. Common results of their study represented on Figure 3.1. The work by Zhang et al. [77] focuses on emotive concepts, in this case “hope” and “fear”, and correlate with a number of market indicators.

3.5 Neural Networks in Sentiment Analysis

Sharma and Dey [64] proposed a sentiment classification model using back-propagation artificial neural network (BPANN) based approach that combines the advantages of the machine learning techniques and the lexicon based techniques. The results on two corpuses (the movie and hotel reviews) have shown that the current approach has succeeded in



(a) Comparison of ML algorithms (b) Comparison of Ensemble methods

Figure 3.1: Results of different methods for classification of data from Twitter [38]

reducing the dimensionality while performing well in terms of accuracy for sentiment based classification. The reduction of input features is an important task for machine learning technique based sentiment classification. Therefore, the proposed approach could be a possible solution with better classification performance and scalability. For large data set applications like product reviews classification, the proposed approach would be especially suitable.

Tarasov [69] in your study used Deep Recurrent Neural Networks (RNN) including Long Short-Term Memory (LSTM) model for sentiment analysis of user reviews of restaurants. He compared results obtained with using methods: logistic regression, simple Recurrent Neural Networks, Bidirectional Recurrent Neural Networks, Bidirectional Long Short-Term Memory. In this study from all RNN models, best results were obtained with deep bidirectional LSTM with 2 hidden layers.

To simplify the analysis of the general model text data, a vector representation of single words can be seen as a parameter for training of the system. That is, if the values of vector representations of the single words absent, we can initialize them with random values and train how other parameters recurrent network.

Tai et al. [68] used the LSTM network to solve two tasks: sentiment classification of sentences sampled from movie reviews and predicting the semantic relatedness of sentence pairs.

Socher et al. [65] introduced Recursive Neural Tensor Networks and the Stanford Sentiment Treebank to solve the task of sentiment detection. Using of Recursive Neural Tensor Networks increased the performance in single sentence positive/negative classification from 80% up to 85.4%. The accuracy of predicting fine-grained sentiment labels for all phrases reached 80.7% by Recursive Neural Tensor Network method, an improvement of 9.7% over bag of features for methods such as SVM, Naive Bayes.

Recurrent Neural Networks, as well as their variations in the form of LSTM (Long Short-Term Memory) model, are quite effective in dealing with text data analysis tasks. The application of these models is a promising approach to define the sentiment of users on the social networks.

3.6 Combined methods in Sentiment Analysis

Coletta et al. [26] demonstrated the performance of Support Vector Machines (SVM) combined with cluster ensemble classification on Twitter data. The focus of their paper is on the sentiment analysis of tweets. Taking into account sentiment analysis of tweets, the concerns are different, in particular:

- the frequency of misspellings and slang in tweets is much higher than in other domains, since users frequently post messages (that can contain a specific cultural vocabulary) using many different electronic devices (e.g., cell phones and tablets);
- Twitter users post messages on a variety of topics, unlike other sites, which are tailored to specific topics.

Coletta's study takes into account an algorithm C^3E (from Consensus between Classification and Clustering Ensembles). This algorithm assumes that clustering can provide supplementary constraints that help to classify new data – in particular, that similar instances are more likely to share the same class label. To classify tweets, a C^3E version based on Squared Loss (SL) function, named C^3E-SL, was used. Such an algorithm needs the information (a priori) of two parameters, namely: the relative importance of classifier and cluster ensembles, and the number of iterations of the algorithm. To estimate these parameters, they have used a Dynamic Differential Evolution (D^2E) algorithm. Experimental results showed that the combination of a SVM classifier with a ensemble cluster, as done by C^3E-SL algorithm, can improve the tweet classification. In most of the cases, C3E-SL provided better results than the stand-alone SVM. Furthermore, C^3E-SL showed competitive results. In summary, researchers obtained that the refinements provided by C^3E-SL (optimized by means of D^2E) can improve the classification accuracy of a stand-alone SVM classifier for tweet sentiment analysis.

In [36] Iglesias et al. proposed an approach in the field of data mining (in particular, web news mining). In this sense, their proposal is an evolving approach for classifying different web news articles into various topic areas based on the text content of the articles. Since news websites are daily overwhelmed with plenty of news articles, one of the main advantages of the proposed approach is that it can cope with huge amounts of news in real-time. Since the web news articles change every day, they have proposed an evolving classifier which also changes constantly. This classifier is based on Evolving Fuzzy Systems (EFS) and the model that describes a specific topic area changes according to the change in the text content of their articles. The approach has been successfully tested using real on-line news. This approach can be used in other different areas in which the process of huge amounts of data in real time is needed. For example, since a web news is represented by a set of terms, this approach can be used to categorize social networks messages (tweets) in real-time.

In work [47] by Lima and Castro, a personality prediction system for social media data is introduced. This system named PERSOMA. Its objective is to identify the personality trait of groups of Tweets based only on the information contained in the Tweets themselves, thus, not relying on profile data. To achieve this, the Tweets are initially separated into clusters, which can be, for instance, groups of messages discussing the same subject or the result of the application of a clustering algorithm to the whole dataset. It works with groups of texts, instead of single texts, and does not take users' profiles into account. Also,

the proposed approach extracts meta-attributes from texts and does not work directly with the content of the messages. The set of possible personality traits is taken from the Big Five model and allows the problem to be characterized as a multi-label classification task. The problem is then transformed into a set of five binary classification problems and solved by means of a semi-supervised learning approach, due to the difficulty in annotating the massive amounts of data generated in social media. The system was applied to predict the personality of Tweets taken from three datasets available in the literature (Obama–McCain Debate (OMD): the main subjects that appear in these Tweets are the candidates themselves, plus some hashtags; Sanders: this dataset contains four main subjects: Apple, Google, Microsoft, and Twitter. Clustering was then performed by a combination of these brands with their products or each other; SemEval2013: this dataset does not contain well-defined subjects; thus, a keyword extraction algorithm was applied, and the clusters were defined based on the determined words, and resulted in an approximately 83% accurate prediction (Table 3.1). In the present paper, a Naive Bayes (NB), a Support Vector Machine (SVM), and a Multilayer Perceptron (MLP) Neural Network were used as classifiers to evaluate the proposed system.

Measure	NB	SVM	MLP
Accuracy	0.839 ± 0.090	0.831 ± 0.122	0.834 ± 0.116
Preciision	0.839 ± 0.091	0.831 ± 0.122	0.834 ± 0.116
Recall	0.855 ± 0.095	0.851 ± 0.095	0.857 ± 0.105

Table 3.1: Multi-label semi-supervised classification results of system PERSOMA for the prediction of specific personality traits presented in tweets [47]

The article [33] by Gross and Murthy explored a variety of methods for applying the Latent Dirichlet Allocation (LDA) automated topic modeling algorithm to modeling of the structure and behavior of virtual organizations found within modern social media and social networking environments. Latent Dirichlet Allocation (LDA) is an NLP technique, which involves the use of machine learning techniques to perform semantic analysis of a corpus by building structures that approximate concepts from a large set of documents without relying on external knowledge bases. The authors introduced variants of LDA and made the argument that natural language processing is a critical interdisciplinary methodology to make better sense of social "Big Data" and they were able to successfully model nested discussion topics from forums and blog posts using LDA. Also, they found that LDA is key to go beyond the state-of-the-art in conventional Social Network Analysis techniques.

Renuga Devi and Hemalatha [28] presented a novel algorithm for automatically detecting number of clusters for mining communities in heterogeneous Social Networks. This algorithm named Convergence aware Dirichlet Process Mixture Model (CADPM). The existing Dirichlet Process Mixture model was used for the community mining problem. But it has few drawbacks. The number of clusters for clustering process is unknown in prior. Most of the existing methods are not suitable for a large scale mining application. To overcome these problems the CADPM model was proposed. The proposed CADPM algorithm was evaluated using the Stanford data set, which consists of the web page details of the Stanford University. In this Stanford network, the nodes represent the web pages,

and the edges represent the hyperlinks between the nodes. The proposed algorithm's clustering efficiency was compared with the existing Dirichlet Process Mixture model. For the evaluation three parameters (accuracy, precision, recall) were used. The CADPM model gave 4.29% higher clustering accuracy than the existing method (Table 3.2). From the experimental analysis, the proposed method works best in large scale networks.

Measure	Amazon dataset		Stanford dataset	
	Dirichlet Model	CADPM	Dirichlet Model	CADPM
Accuracy	87.6	94.5	89.5	94.6
Precision	0.88	0.95	0.9	0.95
Recall	0.88	0.95	0.9	0.95

Table 3.2: Results of proposed CADPM algorithm in terms of Accuracy, Precision, and Recall values for Amazon and Stanford datasets [28]

The many researchers in their works rely on the machine learning approach, and this has a reasonable explanation. Algorithms based on rules are given more accurate results, since the work of these methods is closely related to the semantics of words, in contrast to the methods of machine learning, operating statistics and the probability theory. But, the machine learning approach has some disadvantages. According to Yang et al. [75]: "Most existing techniques rely on natural language processing tools to parse and analyze sentences in a review, yet they offer poor accuracy, because the writing in online reviews tends to be less formal than writing in news or journal articles. Many opinion sentences contain grammatical errors and unknown terms that do not exist in dictionaries."

Through the study of relative work done in recent years in sentiment analysis domain we can say that this is a vast domain where a range of techniques available for classification sentiment for data from social media and still there are a lot of ongoing research. The accuracy and complexity of methods depends on the dataset and the number of dataset features.

Chapter 4

Experimental Setup

In previous chapters we have described a necessary background regarding this work on sentiment analysis of data from social network Twitter. In this section we describe our datasets, and all preprocessing of text for each dataset required for using machine learning methods. After that, experimental setup is presented.

4.1 Data Analysis

4.1.1 Description of the datasets

In our work we use a three distinct annotated datasets obtained from tweets on different subjects:

1. Health Care Reform (HCR) dataset. This dataset consist of 4618 tweets from Twitter. Speriosu et al. [66] created this annotated dataset based on tweets about health care reform in the USA. Researchers extracted tweets containing the hashtag "#hcr" (health care reform) from March 2010. A subset of this dataset was manually annotated for polarity (positive, negative, neutral, irrelevant) and polarity targets (health care reform, Obama, Democrats, Republicans, Tea Party, conservatives, liberals, and Stupak). These was separated into training, development and test sets. The training set contains 1 499 tweets, the development set contains 1 618 tweets, and test set contains 1 501 tweets. We refer to this dataset as "Dataset I".

2. Stanford Twitter Sentiment Corpus (STS). This dataset has 1 600 000 training tweets from various areas. It was collected based on specific emoticons by searching the Twitter API by Go et al. [31]. It consists of 800 000 tweets with positive emoticons (such as :), :-), =), :D), and 800 000 tweets with negative emoticons (such as :(, :-(.). We refer to this dataset as "Dataset II".

3. Sanders – Twitter Sentiment Corpus. This dataset consists of 5 513 hand-classified tweets collected from four Twitter search terms: @apple, #google, #microsoft, #twitter. [11] Each tweet has a sentiment label (polarity): positive, neutral, negative, and irrelevant. We refer to this dataset as "Dataset III".

To compare different techniques of classification, we need at least 2 datasets. We can see that selected datasets, described above, have a different sample size. We need conduct experiments on several datasets and analyze results, to be able to answer more exactly

on research questions. The appreciable difference in sample size of the three considered datasets can be affected on results of our experiments. That is why we have stopped on the choice of these datasets.

4.1.2 Preprocessing

Each of datasets was contained in `.csv` file.

Since Stanford Twitter Sentiment Corpus reffered as Dataset II, has 1,6 million tweets, for our experiments we selected two random datasets – 5 000 tweets and 15 000 tweets – from the original dataset. We use only part of the large dataset for experiments due to limitations in technical resources. All experiments are performed on computer with the following characteristics: 8 GB of RAM and 2-core processor. The Datasets include random tweets, i. e. the number of positive and negative tweets for each dataset are random. For this we used MySQL database [4]. We have imported the original dataset and selected 5 000 and 15 000 random entries (tweets) that were placed in tables and were exported in a new `.csv` files. We refer to these datasets as "Dataset II.1" and "Dataset II.2".

Thus, for experiments we use four datasets.

For Dataset I, Dataset II.1, and Dataset II.2 we conducted experiments considering only the positive and negative tweets. Other tweets was removed. For Dataset III we kept positive, negative, neutral, and irrelevant tweets.

To preprocess the tweet datasets, we removed non-English tweets, replaced the links and URLs on token "http".

The statistics of the datasets are shown in Table 4.1.

	No. of tweets	Positive	Negative	Neutral	Irrelevant
Dataset I	1 286	369	917	-	-
Dataset II.1	5 000	3 042	1 958	-	-
Dataset II.2	15 000	7 121	7 879	-	-
Dataset III	5 114	519	572	2 333	1 690

Table 4.1: Statistics of the Twitter datasets used in the experiments

4.1.3 Data representation

We use the vector models of representations of text, such as Bag-of-Words features (unigrams) and Bag-of-N-grams features (bigrams and trigrams), which are used to train classifiers. In these models tweets are represented by a table in which the columns represent the existing words in the tweets and the values represent their frequencies. Therefore, a collection of tweets – after the preprocessing step – can be represented as illustrated in Table 4.2, in which there are n tweets and m words. Each tweet is represented as

$$tweet_i = (w_{i1},\ w_{i2},\ \dots\ ,\ w_{im}),$$

where w_{ij} is the frequency of word (or n-words, in case when we use n-grams) w_j in the $tweet_i$. This value are calculated in binary frequency (0 or 1).

Weka platform [12] is used in order to build vector models of unigram (Bag-of-Words), bigram and trigram features for all datasets. We use unigram and bigram features to

	w_1	w_2		w_m
$tweet_1$	a_{11}	a_{12}	...	a_{1m}
$tweet_2$	a_{21}	a_{22}	...	a_{2m}
...	...	...	...	...
$tweet_n$	a_{n1}	a_{n2}	...	a_{nm}

Table 4.2: Representation of tweets

represent tweets of all datasets. Trigram features are used to represent tweets of Dataset I, due to a little sample size of this dataset. Table 4.3 lists, for each dataset, the total number of the extracted unigram, bigram and trigram features that are used for the classification training.

	Unigram	Bigram	Trigram
Dataset I	2 897	6 404	6 571
Dataset II.1	1 044	1 262	-
Dataset II.2	1 006	1 040	-
Dataset III	1 080	1 212	-

Table 4.3: Total number of unigram, bigram and trigram features extracted from each dataset

4.2 Implementation

In our experiments, each of the classifiers is trained and tested for each of the datasets. For the learning process, we need to separate training and testing sets. For this purpose, we can either randomly split the datasets into training and testing subsets (e.g., 60% and 40%) or apply the n-fold cross-validation procedure. In a n-fold cross-validation the classifier is trained on $n-1$ folds of the data and tested on the remaining fold, then this is repeated n times for different splits, and the results are averaged over the n experiments. We report results for 2, 4, 8, 16-fold cross-validation for Dataset I and Dataset III, because these two datasets have a little sample size.

In addition, we use existing separated training and testing subsets (48% and 52%) for Dataset I. We split Dataset II.1 on training and testing subsets in ratio 60% and 40%, respectively. We split Dataset II.2 and Dataset III on training and testing subsets in following ratios: 50% and 50%, 60% and 40%, 70% and 30%, and 80% and 20%. We performed a random permutation before splitting the datasets for training and testing (see Table 4.4).

We would like to find out the effect of each set of features (Bag-of-words and Bag-of-Ngrams) to the performance of the classifiers. To achieve an overall comparison, we tested Bag-of-words features (or unigrams) and bigrams, and plus the trigrams for Dataset I, to answer on research question 2 of our work.

We use five machine learning classification methods that are commonly used by sentiment analysis, such as, a Naive Bayes classifier, Multinomial Naive Bayes, Support Vector Machines, Multilayer Perceptron Network, Random Forest.

Dataset	Split,%	Type	Positive	Negative	Neutral	Irrelevant	Total
Dataset I	48/52	train	215	406	-	-	621
		test	154	511	-	-	665
Dataset II.1	60/40	train	1 609	1 391	-	-	3 000
		test	1 433	567	-	-	2 000
Dataset II.2	50/50	train	3 582	3 918	-	-	7 500
		test	3 539	3 961	-	-	7 500
	60/40	train	4 298	4 702	-	-	9 000
		test	2 823	3 177	-	-	6 000
	70/30	train	5 023	5 477	-	-	10 500
		test	2 098	2 402	-	-	4 500
	80/20	train	5 712	6 288	-	-	12 000
		test	1 409	1 591	-	-	3 000
Dataset III	50/50	train	247	274	1 156	880	2 557
		test	272	298	1 177	810	2 557
	60/40	train	301	345	1 419	1 003	3 068
		test	218	227	914	687	2 046
	70/30	train	348	410	1 648	1 774	3 580
		test	171	162	685	516	1 534
	80/20	train	430	455	1 861	1 345	4 091
		test	89	117	472	345	1 023

Table 4.4: Statistics of different split of the Twitter datasets used in the experiments

We use existing WEKA [34] implementations of Naive Bayes, Multinomial Naive Bayes, Support Vector Machines, and Random Forest, with their default settings. We used the Library for Support Vector Machines (LibSVM) for training SVM classifiers with linear kernel and the cost parameter, C, set to 1. Optimising classifier parameters and using alternative kernels most likely would improve performance, however such an exercise is outside the scope of this work. But still we carried out an experiment and compared results obtained for all kernel type and made sure that the using of the linear kernel in SVM gives a better result. For these experiments the Dataset III (consists of 5 114 tweets) is used.

Besides, we decided to use scikit-learn [6] for supposed robustness in handling big data. Scikit-learn is a free software machine learning library for the Python programming language. This software is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn has an expansive list of available algorithms [7]. We used the following methods: Multinomial Naive Bayes, Multilayer Perceptron Network.

In our experimentation we use simply accuracy, precision, recall and F-measure to evaluate the performance of the classifiers (all evaluation measure are described in Chapter 2). The precision, recall and F-measure metrics evaluate the quality of algorithms separately for each class (e. g. positive or negative). Accuracy metric is convenient for multiclass classification tasks to account for imbalanced test data. Since our datasets include tweets of various polarities, for comparing the performance of different methods the *accuracy*

metric is chosen.

Details about each experimentation session are described in Chapter 5.

Chapter 5

Experiments

This chapter describes several experimentation sessions with the datasets described in Chapter 4. We have used various machine learning techniques described in Chapter 2. All main experimental setups are described in the Chapter 4. This chapter has the following structure: Chapters 5.1 and 5.2 show how the Naive Bayes and Multinomial Naive Bayes methods have been applied for sentiment classification datasets from social network Twitter. After that the following machine learning models have been applied to these datasets: support vector machines (5.3), Random Forest (5.4) and Multilayer Perceptron Network (5.5). The obtained results and progress during each experiment session are described in this chapter.

5.1 Naive Bayes Classifier

In the first experiment probabilistic method namely Naive Bayes Classifier are used. We have used this method for all datasets. For experiments Weka implementation was chosen since Weka is an open source software, has no limits for the number of instances and contains tools for data preprocessing, visualisation, classification, regression, rules and others [12]. Separate training and test sets are used for all datasets. For this experiment, we focused on features based on unigrams and bigrams. Also trigrams features are used for vector representation of Dataset I.

We compare the performance of Naive Bayes method for Dataset I with three features: unigram, bigram, and trigram based classifiers. Accuracy, precision, recall and F-measure were chosen as evaluation measures. Last three measures are measures evaluate the quality of algorithms only in relation to one specific class (positive or negative). The *accuracy* metric is convenient for multiclass classification tasks to account for imbalanced test data.

Dataset I includes tweets of two classes: positive and negative. Therefore, for comparing the performance of method, accuracy metric is used. The results of the classification evaluation are presented in Table 5.1. We observe that the accuracy by a trigrams based classifier is 77.44% which is better than the accuracy obtained by a unigram or bigram based classifier. Results by bigram and unigram are comparable, and equal to 74.89% and 73.83%, respectively.

Dataset II.1 also includes tweets of two classes. We compare the performance of Naive Bayes classifier for Dataset II.1 with two features: unigrams and bigrams based classifiers. Accuracy, precision, recall and F-measure were chosen as evaluation measures. However, accuracy metric is main evaluation measure. The results of the classification evaluation

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	2 897	73.83%	0.835	0.822	0.828	0.438	0.461	0.449
bigrams	6 404	74.89%	0.794	0.910	0.848	0.418	0.214	0.283
trigrams	6 571	77.44%	0.773	1	0.872	1	0.026	0.051

Table 5.1: Performance of Naive Bayes (Dataset I)

are presented in Table 5.2. We observe that the accuracy by a bigrams features is higher than the accuracy obtained by a unigrams features.

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	1 044	62.85%	0.405	0.660	0.502	0.821	0.616	0.704
bigrams	1 262	69.15%	0.446	0.363	0.400	0.765	0.821	0.792

Table 5.2: Performance of Naive Bayes (Dataset II.1)

Dataset II.2 consists of tweets of two classes: positive and negative. In order to be able to compare experimental results Dataset II.2 was split into two parts in four ratios: 50% for training and 50% for testing, 60% for training and 40% for testing, 70% for training and 30% for testing, and 80% for training and 20% for testing. The results of the classification evaluation are presented in Table 5.3.

Feature	Count of attributes	Split,%	Accuracy	Negative			Positive		
				P	R	F	P	R	F
unigrams	1 006	50/50	64.32%	0.658	0.675	0.667	0.626	0.607	0.616
		60/40	64.95%	0.664	0.683	0.674	0.632	0.611	0.621
		70/30	65.56%	0.671	0.695	0.683	0.636	0.610	0.627
		80/20	65.87%	0.670	0.702	0.686	0.644	0.610	0.627
bigrams	1 040	50/50	59.45%	15 sec	0.574	0.905	0.702	0.699	0.365
		60/40	59.67%	0.573	0.932	0.710	0.741	0.219	0.338
		70/30	60.04%	0.577	0.938	0.715	0.751	0.214	0.333
		80/20	58.9%	0.567	0.950	0.710	0.762	0.182	0.293

Table 5.3: Performance of Naive Bayes classifier for different split for Dataset II.2

Figure 5.1 shows the results of the accuracy for different splits. The graph shows the obtained accuracy when using unigrams and bigrams. We obtained a maximum accuracy of 65.87% for split 80/20 when using unigrams, and a maximum accuracy of 60.04% for split 70/30 when using bigrams.

Dataset III consists of tweets with four polarity classes: positive, negative, neutral, and irrelevant. For experiment on Dataset III, we split the dataset into two parts in four ratios: 50% for training and 50% for testing, 60% for training and 40% for testing, 70%

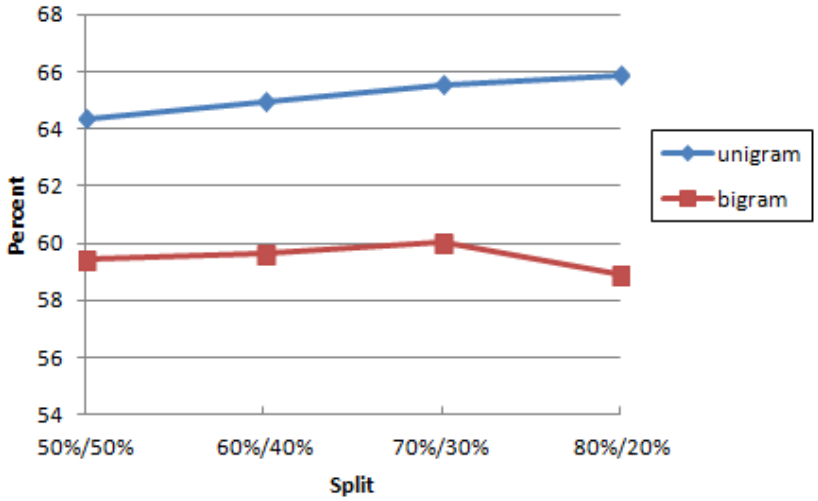


Figure 5.1: Accuracy Naive Bayes classifier on training and test sets for Dataset II.2

for training and 30% for testing, and 80% for training and 20% for testing. The results of the classification evaluation are presented in Table 5.4.

	Measure	unigrams				bigrams			
		50/50	60/40	70/30	80/20	50/50	60/40	70/30	80/20
	Acc	62.18%	62.46%	62.06%	63.44%	52.41%	50.20%	50.85%	51.42%
pos	P	0.311	0.328	0.333	0.346	0.217	0.500	0.157	0.207
	R	0.272	0.202	0.211	0.303	0.055	0.028	0.047	0.067
	F	0.290	0.250	0.258	0.323	0.088	0.052	0.072	0.102
neg	P	0.373	0.349	0.319	0.363	0.733	0.625	0.556	0.667
	R	0.651	0.656	0.654	0.658	0.037	0.022	0.031	0.034
	F	0.474	0.456	0.429	0.468	0.070	0.043	0.058	0.065
neu	P	0.720	0.702	0.702	0.726	0.497	0.473	0.484	0.492
	R	0.556	0.578	0.555	0.568	0.934	0.990	0.965	0.966
	F	0.627	0.634	0.620	0.637	0.649	0.640	0.645	0.652
irrel	P	0.750	0.760	0.778	0.761	0.817	0.974	0.981	0.968
	R	0.825	0.811	0.833	0.803	0.265	0.162	0.205	0.174
	F	0.785	0.785	0.804	0.781	0.401	0.277	0.340	0.295

Table 5.4: Performance of Naive Bayes Classifier for different split for Dataset III

Figure 5.2 shows the results of the accuracy for different splits. As seen in Figure 5.2, the obtained accuracies for each of splits are almost equal. When using unigrams a maximum accuracy is 63.44% for split 80/20; when using bigrams a maximum accuracy is 52.42% for split: 50% for training and 50% for testing.

For Dataset I we apply 2, 4, 8, 16-fold cross-validation procedure. We would like to

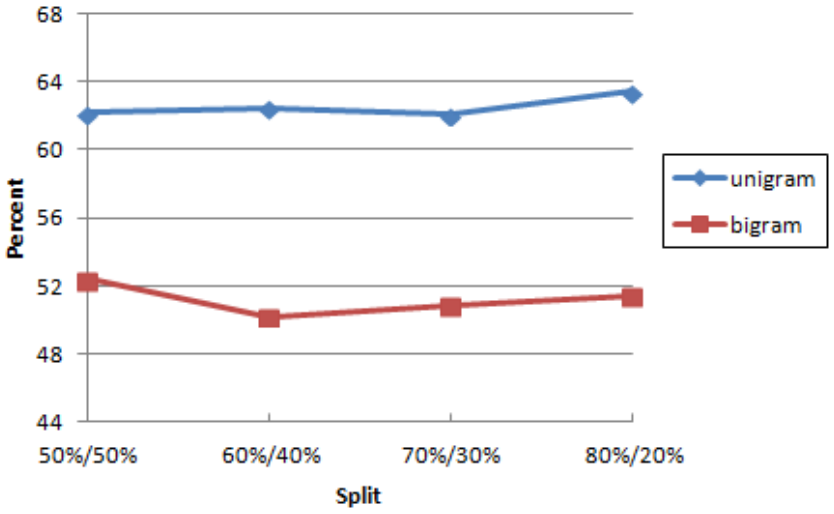


Figure 5.2: Accuracy Naive Bayes classifier on training and test sets for Dataset III

find out the effect of each of features: unigrams, bigrams and trigrams, to the performance of the Naive Bayes classifier.

The results are listed in the Table 5.5.

Feature	Cross Val	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	2	73.02%	0.81	0.812	0.811	0.53	0.526	0.528
	4	72.86%	0.803	0.821	0.812	0.529	0.499	0.513
	8	73.72%	0.81	0.824	0.817	0.544	0.52	0.532
	16	74.81%	0.818	0.832	0.825	0.564	0.539	0.551
bigrams	2	71.23%	0.758	0.877	0.813	0.498	0.304	0.377
	4	71.70%	0.763	0.876	0.815	0.511	0.322	0.395
	8	73.17%	0.766	0.899	0.827	0.557	0.317	0.404
	16	72.86%	0.769	0.884	0.823	0.543	0.341	0.419
trigrams	2	73.95%	0.733	0.999	0.845	0.972	0.095	0.173
	4	73.72%	0.732	0.997	0.844	0.919	0.092	0.167
	8	73.48%	0.73	0.998	0.843	0.938	0.081	0.15
	16	73.48%	0.73	0.998	0.843	0.938	0.081	0.15

Table 5.5: Performance of Naive Bayes for 2-, 4-, 8-, 16-fold cross validation (Dataset I)

In Figure 5.3 the results from 2, 4, 8, 16-fold cross validation; the last columns show the results from split of the dataset where 48% are the training set and 52% are the test set, are represented.

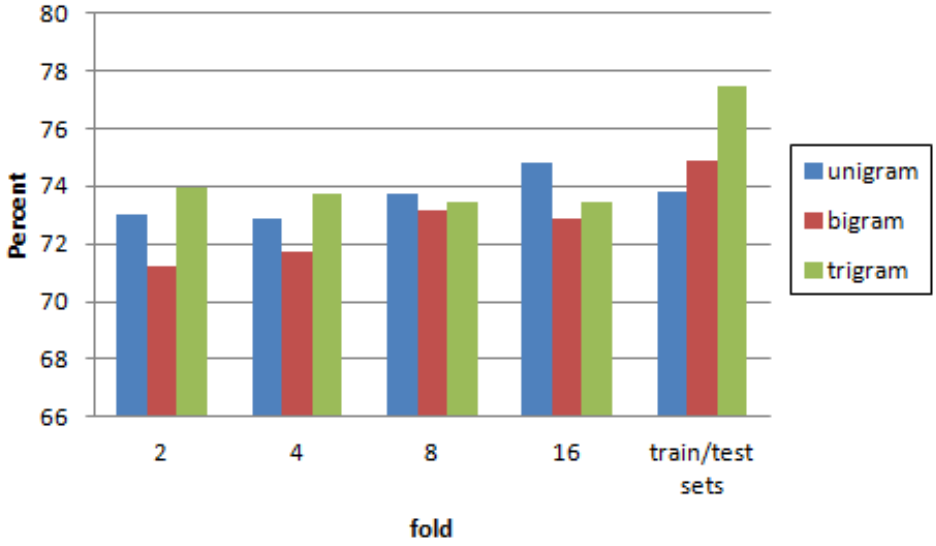


Figure 5.3: Comparison of an accuracies for Naive Bayes trained on unigrams, bigrams and trigrams features for Dataset I.

For comparison we also include the results when one part of the dataset (48%) is used as training data and the rest (52%) as test data. The results are slightly different to that of 2, 4, 8, 16-fold cross validation. The marked difference appears when classifier trains on dataset by trigrams features. As follows from the Tables 5.1 and 5.5, the values of F-measures are slightly different for cross validation and split. In case split on training and test sets, the values of F-measures is higher, likely because more data are used for training.

5.2 Multinomial Naive Bayes

As we have described in Section 2.3.2 the Multinomial Naive Bayes is a probabilistic method. For experiments Weka implementation was chosen. And also, for supposed robustness in handling big data, Scikit-learn [6] tool was chosen.

As in the previous experiment, separate training and test sets are used for all datasets. For this experiment, we focused on features based on unigrams and bigrams. Also trigrams features are used for vector representation of Dataset I.

The results for Dataset I are listed in the Table 5.6. From Table 5.7 seen, that the obtained results by Weka and Scikit-learn are identical. We observe that the accuracy by a unigrams based classifier is 80.6% which is better than the accuracy obtained by a bigram and trigram based classifier. Results by bigram and trigram are identical, and equal to 76.54% and 76.69%, respectively.

The results for Dataset II.1 are listed in the Table 5.7. For the Dataset II.1 the obtained values of performances by Weka and Scikit-learn are also identical. The accuracy by a

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	2 897	80.60%	0.819	0.959	0.884	0.687	0.299	0.416
bigrams	6 404	76.54%	0.774	0.980	0.865	0.444	0.052	0.093
trigrams	6 571	76.69%	0.771	0.992	0.867	0.429	0.019	0.037
unigrams	2 897	80.60%	0.82	0.96	0.88	0.69	0.30	0.42
bigrams	6 404	76.54%	0.77	0.98	0.87	0.44	0.05	0.09
trigrams	6 571	76.69%	0.77	0.99	0.87	0.43	0.02	0.04

Table 5.6: Performance of Multinomial Naive Bayes (Dataset I).
The first 3 rows show the results obtained using Weka; the last three rows show the results obtained using Scikit-learn.

unigrams features is 71.4% that is higher than the accuracy obtained by a bigrams features (67.2%).

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	1 044	71.4%	0.497	0.674	0.572	0.850	0.730	0.785
bigrams	1 262	67.2%	0.437	0.545	0.485	0.800	0.722	0.759
unigrams	1 044	71.4%	0.50	0.67	0.57	0.85	0.73	0.79
bigrams	1 262	67.2%	0.44	0.54	0.49	0.80	0.72	0.76

Table 5.7: Performance of Multinomial Naive Bayes (Dataset II.1).
The first 2 rows show the results obtained using Weka; the last two rows show the results obtained using Scikit-learn.

Since values of performances of first two datasets obtained from by Weka and Sciket-learn are coincide, for Dataset II.2 only Weka implementation are used. For Multinomial Naive Bayes method we are also used split of Dataset II.2 into traing set and test set. The results for Dataset II.2 are listed in the Table 5.8.

Figure 5.4 shows the results of the accuracy for four different splits of Dataset II.2. From this graph, we can see that accuracy is in the range from 62% to 64% for bigrams and in the range from 71% to 72% for unigrams. As well, from Table 5.8 we can see that the values of F-measures are almost identical for all splits based on unigrams and bigrams features.

For Dataset III Weka implementation are used. Dataset III includes tweets with four polarity classes: positive, negative, neutral, and irrelevant. For experiment on dataset III, we split the dataset into two parts in four ratios: 50% for training and 50% for testing, 60% for training and 40% for testing, 70% for training and 30% for testing, and 80% for training and 20% for testing. The results of the classification evaluation are presented in Table 5.9

Figure 5.5 shows the results of the accuracy for different splits. When using unigrams a maximum accuracy is 70.97% for split 60/40; when using bigrams a maximum accuracy is 59.82% for split: 80% for training and 20% for testing.

Feature	Count of attributes	Split,%	Accuracy	Negative			Positive		
				P	R	F	P	R	F
unigrams	1 006	50/50	71.11%	0.714	0.756	0.734	0.708	0.660	0.683
		60/40	71.68%	0.719	0.765	0.741	0.715	0.663	0.688
		70/30	71.8%	0.725	0.760	0.742	0.709	0.670	0.689
		80/20	71.87%	0.722	0.762	0.742	0.714	0.669	0.691
bigrams	1 040	50/50	62.96%	0.625	0.745	0.680	0.637	0.501	0.561
		60/40	62.42%	0.622	0.741	0.676	0.628	0.493	0.552
		70/30	63.62%	0.637	0.742	0.685	0.636	0.515	0.569
		80/20	63.33%	0.630	0.746	0.683	0.638	0.506	0.565

Table 5.8: Performance of Multinomial Naive Bayes for different split for Dataset II.2

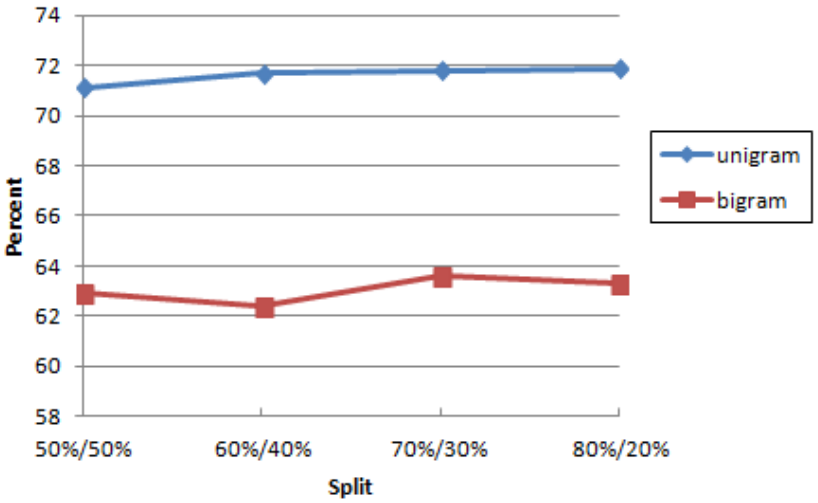


Figure 5.4: Accuracy Multinomial Naive Bayes on training and test sets for Dataset II.2

For Dataset I we apply 2, 4, 8, 16-fold cross-validation procedure. We would like to find out the effect of each of features: unigrams, bigrams and trigrams, to the performance of the Multinomial Naive Bayes classifier.

For experiment Weka implementation is used. The results are listed in the Table 5.10.

In Figure 5.6 the results from 2, 4, 8, 16-fold cross validation are represented; the last columns show the results for train/test split.

We can see that the value of accuracy increases with increasing of folds cross validation. This trend keeps for each features of Dataset I.

Using unigrams features of Dataset I showed high accuracy both on cross validation and on train/test split.

Using the Multinomial Naive Bayes method, we obtain a more accurate classification results than the results of the Naive Bayes method.

	Measure	unigrams				bigrams			
		50/50	60/40	70/30	80/20	50/50	60/40	70/30	80/20
	Acc	69.85%	70.97%	69.1%	69.89%	55.18%	57.04%	57.11%	59.82%
pos	P	0.468	0.490	0.455	0.433	0.206	0.161	0.205	0.270
	R	0.217	0.234	0.269	0.326	0.206	0.161	0.205	0.270
	F	0.296	0.317	0.388	0.372	0.251	0.233	0.285	0.331
neg	P	0.436	0.454	0.380	0.436	0.413	0.371	0.346	0.433
	R	0.654	0.626	0.574	0.607	0.295	0.330	0.346	0.385
	F	0.523	0.526	0.457	0.507	0.344	0.350	0.346	0.407
neu	P	0.733	0.712	0.719	0.732	0.539	0.541	0.546	0.574
	R	0.714	0.756	0.699	0.701	0.777	0.806	0.784	0.805
	F	0.723	0.733	0.709	0.716	0.636	0.648	0.644	0.670
irrel	P	0.826	0.863	0.847	0.833	0.745	0.800	0.790	0.811
	R	0.854	0.827	0.857	0.823	0.436	0.466	0.481	0.472
	F	0.840	0.845	0.852	0.828	0.550	0.589	0.598	0.597

Table 5.9: Performance of Multinomial Naive Bayes for different split for Dataset III

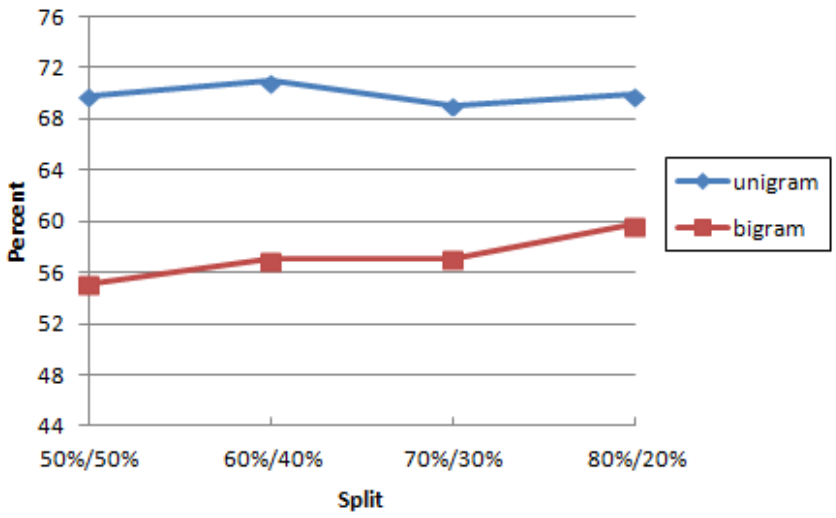


Figure 5.5: Accuracy Multinomial Naive Bayes on training and test sets for Dataset III

5.3 Support Vector Machines

This experimentation deals with Support Vector Machines and its implementation in Weka was chosen as a tool.

SVM for classification problem is implemented in Weka as LibSVM. However as it was described in theory section SVM has parameters and they affect models performance. Our approach here was to run Dataset III trying different types of the kernel and find the

Feature	Cross Val	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	2	79.08%	0.788	0.967	0.868	0.813	0.352	0.491
	4	81.65%	0.813	0.964	0.882	0.834	0.45	0.585
	8	81.96%	0.817	0.962	0.884	0.831	0.466	0.597
	16	82.04%	0.821	0.957	0.884	0.819	0.48	0.605
bigrams	2	75.97%	0.755	0.983	0.854	0.826	0.206	0.33
	4	76.67%	0.761	0.981	0.857	0.835	0.233	0.364
	8	77.61%	0.768	0.984	0.862	0.865	0.26	0.4
	16	77.92%	0.771	0.983	0.864	0.863	0.274	0.416
trigrams	2	73.02%	0.731	0.984	0.839	0.712	0.1	0.176
	4	73.79%	0.736	0.986	0.843	0.776	0.122	0.211
	8	74.18%	0.737	0.991	0.846	0.849	0.122	0.213
	16	74.49%	0.74	0.991	0.847	0.86	0.133	0.23

Table 5.10: Performance of Multinomial Naive Bayes 2-, 4-, 8-, 16-fold cross validation (Dataset I)

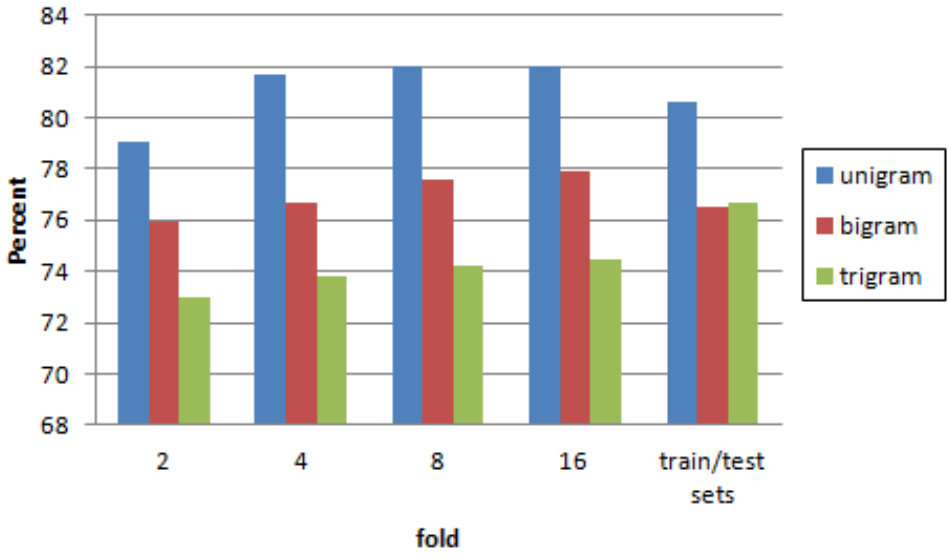


Figure 5.6: Comparison of an accuracies for Multinomial Naive Bayes trained on unigrams, bigrams and trigrams features for Dataset I.

optimal one.

The main disadvantage of SVM models is that it is very slow: time to build a model depends on the complexity of this model and with the growing parameters values can increase exponentially. Taking into consideration this fact we have decided to run all other datasets with one kernel type, with that which gives the best result on Dataset III.

We used the values of all paremeters for SVM classifier defined by default. Comparison

of different kernels is produced by the values of accuracy.

In Table 5.11 are represented results obtained for all kernel types by cross validation on Dataset III with unigrams features. As we can see from the experiments, linear kernel is much better than radial basis kernel, polynomial kernel and sigmoid kernel.

Kernel type	**Cross validation**	**Accuracy**
linear	2	71.10%
	4	72.59%
	8	73.35%
	16	73.70%
radial basis function (RBF)	2	52.44%
	4	58.72%
	8	60.36%
	16	60.97%
polynomial	2	45.62%
	4	45.62%
	8	45.62%
	16	45.62%
sigmoid	2	45.64%
	4	47.52%
	8	50.08%
	16	51.19%

Table 5.11: Comparison results of SVM with different kernel types on Dataset III

According to Hsu et al. [35] a choice of the kernel type depends on a number of instances and a number of features in dataset. When number of features much more than number of instances, the linear kernel is used. When number of features is small (number of instances much more than number of features), the linear or nonlinear kernels (RBF) are used. And finally, when both number of features and instances are large, the package LibSVM is not particularly good for this type of problems. In this case another software LIBLINEAR, which is very suitable for such data, is used. LIBLINEAR is efficient for large-scale document classification.

In our experiment Dataset III consists 5 114 instances (tweets) and 1 080 unigrams features. Thus, our results confirm the theory of Hsu.

For next experiments with other datasets we will use linear kernel.

We run the SVM method for Dataset I with three features: unigram, bigram, and trigram. Dataset I includes 1 286 instances, and 2 897 unigrams, 6 404 bigrams, 6 571 trigrams. The split on training and test sets and 2,4,8,16-cross validation procedure are used.

The results are listed in the following tables. Table 5.12 shows the results for split on training set and test set, Table 5.13 for cross validation.

From the Tables 5.12 and 5.13, and Figure 5.7 we can see that models with 2, 4, 8, 16-fold cross validation show similar values. Values of accuracy by dataset with unigrams features are higher as for cross validation as for train/test split of dataset.

Dataset II.1 includes 5 000 instances (tweets), 1 044 unigrams, 1 262 bigrams features.

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	2 897	80.60%	0.819	0.959	0.884	0.687	0.299	0.416
bigrams	6 404	76.54%	0.774	0.980	0.865	0.444	0.052	0.093
trigrams	6 571	76.69%	0.771	0.992	0.867	0.429	0.019	0.037

Table 5.12: Performance of SVM method (Dataset I)

Feature	Cross Val	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	2	75.74%	0.797	0.884	0.839	0.606	0.442	0.511
	4	76.83%	0.806	0.89	0.846	0.63	0.466	0.536
	8	77.14%	0.808	0.892	0.848	0.637	0.472	0.542
	16	76.98%	0.807	0.89	0.846	0.633	0.472	0.54
bigrams	2	74.81%	0.74	0.996	0.849	0.925	0.133	0.232
	4	75.43%	0.747	0.991	0.852	0.884	0.165	0.279
	8	75.89%	0.752	0.988	0.854	0.864	0.19	0.311
	16	76.05%	0.754	0.987	0.855	0.859	0.198	0.322
trigrams	2	73.41%	0.73	0.996	0.842	0.886	0.084	0.153
	4	74.26%	0.737	0.995	0.846	0.896	0.117	0.206
	8	74.26%	0.737	0.995	0.846	0.896	0.117	0.206
	16	74.81%	0.741	0.995	0.849	0.909	0.136	0.236

Table 5.13: Performance of SVM 2,4,8,16-fold cross validation (Dataset I)

The results for Dataset II.1 are listed in the Table 5.14. The accuracy by a unigrams features is 69.6% that is higher than the accuracy obtained by a bigrams features (67.3%).

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigram	1 044	69.6%	0.472	0.608	0.532	0.825	0.731	0.775
bigram	1 262	67.3%	0.45	0.63	0.52	0.83	0.69	0.75

Table 5.14: Performance of SVM (Dataset II.1)

Dataset II.2 consists of 15 000 instances, 1 006 unigrams features and 1 040 bigrams features. For SVM method we are also used split of Dataset II.2 into traing set and test set. The results for Dataset II.2 are listed in the Table 5.15.

From the Table 5.15 and Figure 5.8 we can see that accuracy of different splits is in the range from 62% to 65% for dataset by bigrams features and in the range from 71% to 74% for dataset by unigrams.

And also we perform the experiments to train SVM classifier on Dataset III by different

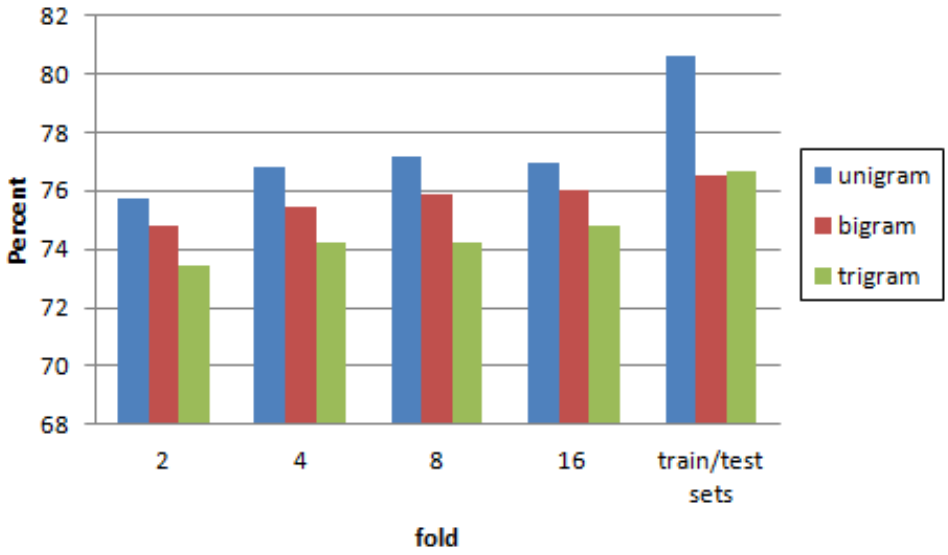


Figure 5.7: Comparison of an accuracies for SVM trained on unigrams, bigrams and trigrams features for Dataset I.

Feature	Count of attributes	Split,%	Accuracy	Negative			Positive		
				P	R	F	P	R	F
unigrams	1 006	50/50	71.8%	0.744	0.710	0.727	0.691	0.727	0.709
		60/40	72.68%	0.750	0.725	0.738	0.702	0.729	0.715
		70/30	73.58%	0.764	0.730	0.747	0.706	0.742	0.724
		80/20	73.2%	0.756	0.731	0.743	0.707	0.733	0.720
bigrams	1 040	50/50	62.91%	0.693	0.535	0.604	0.585	0.734	0.651
		60/40	63.28%	0.701	0.535	0.607	0.587	0.743	0.656
		70/30	64.36%	0.716	0.550	0.622	0.593	0.750	0.662
		80/20	64.2%	0.709	0.551	0.620	0.595	0.744	0.661

Table 5.15: Performance of Support Vector Machine method for different split for Dataset II.2

splits.

The results of the classification evaluation are presented in Table 5.16 and Figure 5.9. Figure 5.9 shows the results of the accuracy for different splits.

We can see that accuracy is increased when increasing the amount of data for training. We obtaine a maximum values of accuracy when dataset is splited in range 80% for traning and 20% for test. When using unigrams a maximum accuracy is 70.28%, when using bigrams a maximum accuracy is 63.05%.

Comparing the results obtained by SVM method with previous methods, it can be concluded that SVM method shows a higher classification accuracy than the above Naive Bayes method and similar results of classification to the Multinomial Naive Bayes method.

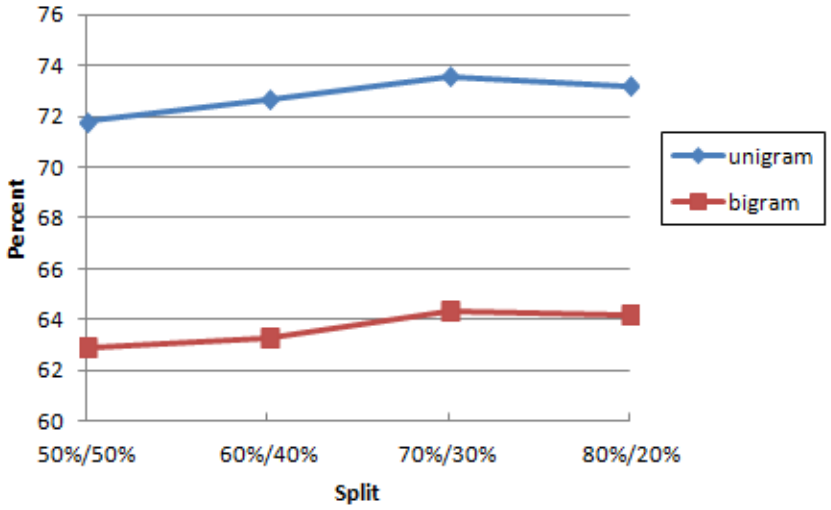


Figure 5.8: Accuracy SVM method on training and test sets for Dataset II.2

	Measure	unigrams				bigrams			
		50/50	60/40	70/30	80/20	50/50	60/40	70/30	80/20
	Acc	65.66%	65.59%	67.99%	70.28%	57.96%	57.97%	59.71%	63.05%
pos	P	0.367	0.388	0.407	0.464	0.440	0.432	0.370	0.510
	R	0.228	0.284	0.216	0.292	0.176	0.161	0.117	0.292
	F	0.281	0.328	0.282	0.359	0.252	0.234	0.178	0.371
neg	P	0.403	0.438	0.416	0.450	0.426	0.539	0.425	0.519
	R	0.537	0.617	0.488	0.650	0.201	0.242	0.210	0.231
	F	0.460	0.512	0.449	0.531	0.273	0.334	0.281	0.320
neu	P	0.692	0.675	0.694	0.729	0.612	0.537	0.621	0.651
	R	0.684	0.674	0.704	0.725	0.686	0.886	0.708	0.725
	F	0.688	0.674	0.699	0.727	0.647	0.669	0.662	0.686
irrel	P	0.788	0.802	0.797	0.836	0.574	0.808	0.609	0.633
	R	0.805	0.763	0.862	0.797	0.699	0.416	0.731	0.725
	F	0.797	0.782	0.829	0.816	0.630	0.549	0.664	0.676

Table 5.16: Performance of SVM method for different split for Dataset III

5.4 Random Forest

This experimentation deals with Random Forest method and its implementation in Weka was chosen as a tool.

Random Forest method for classification is implemented in Weka as RandomForest. We run the Random Forest classifier with 100 random trees.

As we have seen in the literature (Chapter 3) the Random Forest algorithm can produce

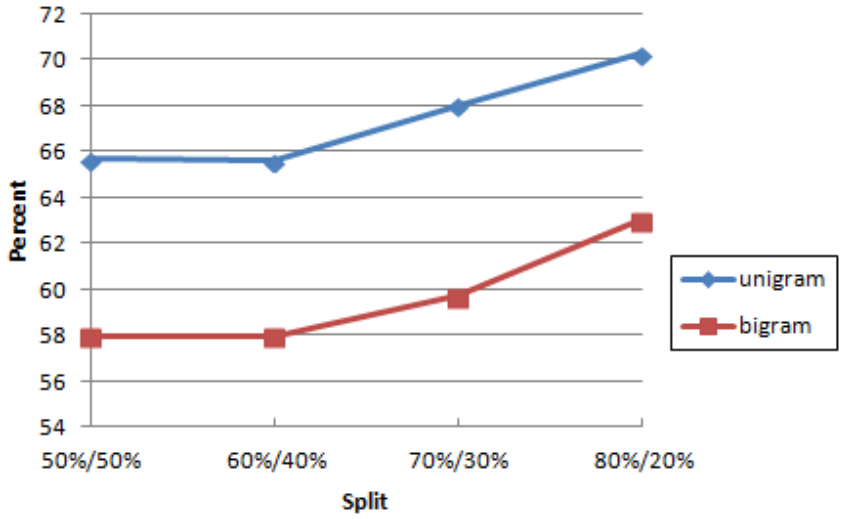


Figure 5.9: Accuracy SVM on training and test sets for Dataset III

high performance for text based classification. By combining multiple simple random trees the Random Forest algorithm can produce significantly higher performance than each tree individually. For such a simple algorithm the accuracy is really high.

We are used three datasets to carry out experiments. Dataset II.2 was not used due to its large sample size.

In Tables 5.17, 5.18 and 5.19 results for Dataset I, Dataset II.1 and Dataset III are represented.

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	2 897	80.60%	0.819	0.959	0.884	0.687	0.299	0.416
bigrams	6 404	76.54%	0.774	0.980	0.865	0.444	0.052	0.093
trigrams	6 571	76.69%	0.771	0.992	0.867	0.429	0.019	0.037

Table 5.17: Performance of Random Forest classifier (Dataset I)

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	1 044	71.4%	0.497	0.674	0.572	0.850	0.730	0.785
bigrams	1 262	67.2%	0.437	0.545	0.485	0.800	0.722	0.759

Table 5.18: Performance of Random Forest classifier (Dataset II.1)

As we can see the Random Forest method produces good results on dataset which consists of a small sample size. The accuracy reduces with sample increasing. Also, the accuracy is higher for each datasets by unigrams features.

Figure 5.10 shows the results of the accuracy for different splits of Dataset III which

	Measure	unigrams				bigrams			
		50/50	60/40	70/30	80/20	50/50	60/40	70/30	80/20
	Acc	68.91%	67.89%	69.82%	70.09%	56.75%	60.31%	58.93%	59.92%
pos	P	0.652	0.727	0.333	0.300	0.477	0.373	0.579	0.395
	R	0.055	0.037	0.035	0.067	0.195	0.174	0.193	0.191
	F	0.102	0.070	0.063	0.110	0.277	0.238	0.289	0.258
neg	P	0.687	0.730	0.688	0.658	0.335	0.413	0.361	0.443
	R	0.154	0.119	0.204	0.214	0.208	0.220	0.241	0.265
	F	0.252	0.205	0.314	0.323	0.257	0.287	0.289	0.332
neu	P	0.634	0.639	0.632	0.649	0.681	0.719	0.677	0.703
	R	0.868	0.809	0.869	0.869	0.523	0.574	0.533	0.547
	F	0.733	0.714	0.731	0.743	0.591	0.639	0.596	0.615
irrel	P	0.794	0.730	0.831	0.829	0.531	0.568	0.563	0.565
	R	0.838	0.895	0.847	0.800	0.890	0.904	0.905	0.890
	F	0.816	0.804	0.839	0.814	0.665	0.698	0.694	0.691

Table 5.19: Performance of Random Forest method for different split for Dataset III

includes tweets with four polarity classes: positive, negative, neutral, and irrelevant.

When we use unigrams features to train of classifier a maximum accuracy is 70.09% for split dataset into 80% for training and 20% for testing; when we use bigrams features a maximum accuracy is 60.31% for split: 60% for training and 40% for testing.

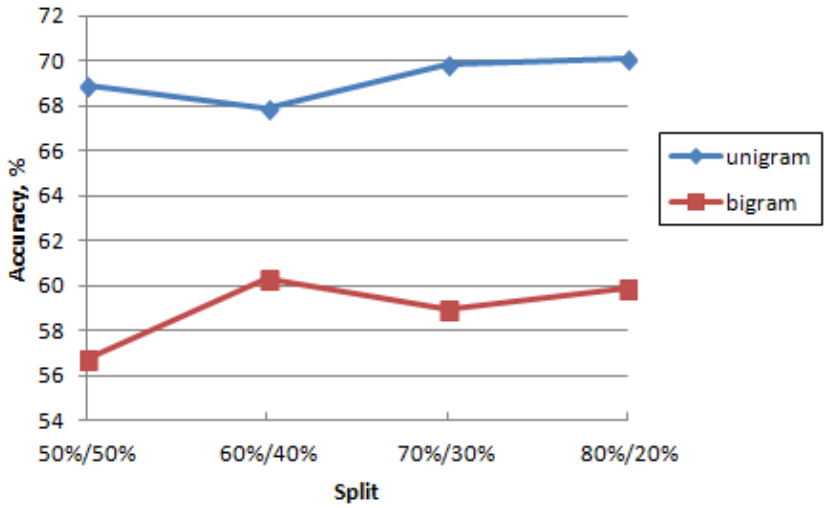


Figure 5.10: Accuracy Random Forest classifier on training and test sets for Dataset III

5.5 Multilayer Perceptron Network

This part of Chapter 5 describes experiments which deal with artificial neural networks: Multilayer Perceptron. As a tool Scikit-learn [6] for this experimentation was used.

Scikit-learn is a free software machine learning library for the Python programming language. This software is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

For experimentation we use three datasets: Dataset I, Dataset II.1 which are splited on training and test sets, and Dataset III that is splited on training and testing sets in following ratios: 50% for training and 50% for testing, 60% and 40%, 70% and 30%, and 80% and 20%.

Accuracy was chosen as the primary evaluation measure of this method.

The Python scripts are documented in CD attached to this report.

In Tables 5.20, 5.21 and 5.22 results for Dataset I, Dataset II.1 and Dataset III are represented.

The value of accuracy on Dataset I with trigrams features is higher than the values obtained on dataset with unigrams and bigrams features. However, the values of F-measure differ greatly when classifying data into positive and negative by trigrams representation of dataset.

Values of accuracy on Dataset II.1 with unigrams and bigrams features are almost identical.

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	2 897	76.99%	0.83	0.89	0.86	0.50	0.38	0.44
bigrams	6 404	76.84%	0.79	0.95	0.86	0.50	0.16	0.24
trigrams	6 571	78.20%	0.78	1	0.88	0.85	0.07	0.13

Table 5.20: Performance of Multilayer Perceptron method (Dataset I)

Feature	Count of attributes	Accuracy	Negative			Positive		
			P	R	F	P	R	F
unigrams	1 044	69.6%	0.47	0.52	0.49	0.80	0.77	0.78
bigrams	1 262	68.9%	0.43	0.28	0.34	0.75	0.85	0.80

Table 5.21: Performance of Multilayer Perceptron method (Dataset II.1)

Figure 5.11 represents the results of the accuracy for different splits of Dataset III which includes tweets with four polarity classes: positive, negative, neutral, and irrelevant.

A maximum accuracy is 69.70% for dataset with unigrams features which is splited in range 80% for training and 20% for testing. When we use bigrams features to train of classifier a maximum accuracy is 60.23% for split: 70% for training and 30% for testing.

The values of accuracy of Multilayer Perceptron method decreases with increasing dataset sizes.

	Measure	unigrams				bigrams			
		50/50	60/40	70/30	80/20	50/50	60/40	70/30	80/20
	Acc	66.72%	65.35%	66.75%	69.70%	56.39%	53.91%	60.23%	55.23%
pos	P	0.44	0.27	0.40	0.40	0.37	0.35	0.34	0.31
	R	0.25	0.43	0.19	0.18	0.28	0.27	0.37	0.35
	F	0.32	0.33	0.25	0.25	0.32	0.31	0.36	0.33
neg	P	0.45	0.44	0.39	0.42	0.38	0.22	0.40	0.24
	R	0.41	0.59	0.40	0.67	0.24	0.58	0.20	0.54
	F	0.43	0.51	0.39	0.51	0.30	0.32	0.26	0.33
neu	P	0.63	0.71	0.63	0.74	0.54	0.65	0.74	0.66
	R	0.85	0.66	0.87	0.68	0.84	0.66	0.52	0.67
	F	0.72	0.68	0.73	0.71	0.66	0.65	0.61	0.67
irrel	P	0.94	0.94	0.97	0.82	0.88	0.84	0.60	0.86
	R	0.64	0.74	0.65	0.86	0.38	0.45	0.91	0.45
	F	0.76	0.83	0.78	0.84	0.53	0.59	0.72	0.59

Table 5.22: Performance of Multilayer Perceptron method for different split for Dataset III

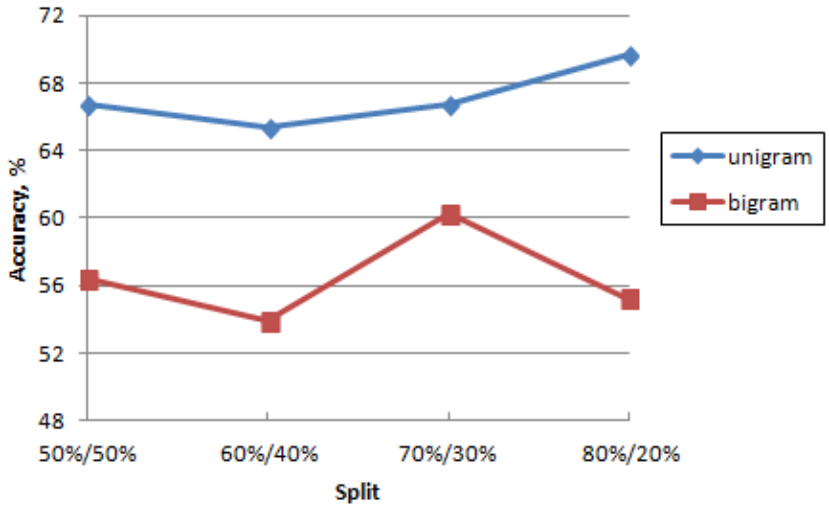


Figure 5.11: Accuracy Multilayer Perceptron method on training and test sets for Dataset III

5.6 Summary

This chapter represents the experiments and results of experiments that were conducted for the datasets from social network Twitter. Since we have four datasets of different sample size with tweets from different domains, experiments for each dataset were carried out. Still main tendencies was the same for all datasets. To answer the research question and find the most accurate method for classification these datasets, on the first place the

data was analyzed (see Chapter 4)

In Chapters 5.1 and 5.2 the Naive Bayes and Multinomial Naive Bayes methods were applied to our datasets that consists of the data from Twitter. As discoursed in Chapter 2.5 before applying machine learning techniques to text data, the data itself should be represented in the vector form. In purpose to find the best data representation, experiments were conducted and Bag-of-Words model (unigrams) and Bag-of-N-grams model (bigrams and trigrams) of representation of the text were chosen. Bag-of-Words model (unigrams) of representation of the text showed the best result for all methods. From Naive Bayes methods Multinomial Naive Bayes performed best. The accuracy by a unigrams model of Dataset I is 80.6%, 71.4% for Dataset II.1, 71.87% for Dataset II.2, and 70.97% for Dataset III. Chapter 5.3 describes how different support vector machines with polynomial, linear, sigmoid and Radial Basis kernels were applied to sentiment classification of Dataset III. As experiments showed, choosing of the kernel function influences the model performance for this dataset. Also when SVM with linear kernel was applied, we obtained results much better than results of SVM with other kernels. The most accurate model was the model with SVM with linear kernel. Therefore SVM with linear kernel was applied for the sentiment classification on all the datasets. The performance of the Random Forest method and Multilayer Perceptron Network method do not difference much. As for Random Forest method, the model with 100 random trees achieved the best performance for each dataset. By combining multiple simple random trees the Random Forest algorithm is produced significantly higher performance than each tree individually. Chapter 5.5 describes how Multilayer Perceptron Network has been applied to sentiment classification of the datasets. The accuracy by a unigrams model is 76.99% for Dataset I, 69.6% for Dataset II.1, and 69.7% for Dataset III. These values of performance are lower than values obtained by the rest methods.

Comparing performance of different methods we can say that Multinomial Naive Bayes and Support Vector Machines classifiers represented high results. The results and their discussion are described in Chapter 6.

Chapter 6

Discussion

This chapter deals with discussion over the results and findings described in the previous chapter. First, we compare results of different methods applied for the sentiment analysis of data obtained from Twitter. Second, the discussion of the impact of different features are presented. Also we discuss the best obtained results (which were given by Support Vector Machines and Multinomial Naive Bayes methods). Finally, we propose further work as there is still a lot of room for improvement.

The goal of this work was to compare standard machine learning methods for the sentiment analysis of data collected from Twitter, to find out which machine learning classifier are more accurate. Finding the best performing method is the answer of the first research question. The second research question is more oriented on the preprocessing techniques of data. It might seem that this two research questions are concentrated on different things, but they are highly correlated as analysing the performance of these methods depends on the data preprocessing (depends on models of vector representation of the text data).

The experimentation session, described in previous chapter, was carried out on three datasets with data extracted from Twitter. However, one large dataset was divided into two datasets with less sizes. So, we carried out experiments on the four datasets of different sample sizes. Dataset I consists of 1 286 tweets, Dataset II.1 and Dataset II.2 have 5 000 and 15 000 tweets, respectively, and Dataset III includes 5 114 tweets.

Our main objective was to find the best technique for sentiment classification of messages from social network Twitter. For this purpose we have carried out several experimentation sessions which involved different techniques to classification: Naive Bayes, Multinomial Naive Bayes, Support Vector Machines, Random Forest and Multilayer Perceptron. For each method classification was performed.

The experimentation we began with the probabilistic machine learning methods like Naive Bayes and Multinomial Naive Bayes and moved towards the more complex ones like support vector machines. All together five different techniques were carried out:

- Naive Bayes (NB)
- Multinomial Naive Bayes (MultNB)
- Support Vector Machines (SVM)

- Random Forest (RF)
- Multilayer Perceptron Network (MP)

The most complex technique here is support vector machines.

A summary of the best performance for Naive Bayes, multinomial Naive Bayes, support vector machines, random forest and multilayer perceptron methods is presented in Tables 6.1, 6.2, 6.3 and 6.4 for Dataset I, Dataset II.1, Dataset II.2 and Dataset III, respectively.

To answer the research question and find the method which performs best on the text data, on the first place the following issue should be clarified: What criterion should be used when comparing different methods for the specific datasets? To estimate the classifier's performance the following measures can be used: accuracy, precision, recall, and F-measure or F-score. The precision is the number of instances correctly classified as its true class out of all the instances classified as that class. The recall represents the number correctly classified instances of a class out of all the instances of that class. The F-score or F-measure can be interpreted as the harmonic mean of precision and recall. In our experimentation we used simply accuracy, precision, recall and F-measure to evaluate the performance of the classifiers (all evaluation measure are described in Chapter 2). The precision, recall and F-measure metrics evaluate the quality of algorithms separately for each class (e. g. positive or negative). Accuracy metric represents the overall percentage of correctly classified instances for multiclass classification tasks. As we have mentioned before in the thesis, our datasets include tweets of various polarities. For comparing the performance of different methods the *Accuracy* metric was chosen as a main metric, because exactly Accuracy (Acc) is convenient for multiclass classification tasks to account for imbalanced test data.

Features	Count of attributes	NB	MultNB	SVM	RF	MP
unigram	2897	73.83	**80.60**	**80.60**	**80.60**	76.99
bigram	6404	74.89	76.54	76.54	76.54	**76.84**
trigram	6571	77.44	76.69	**79.69**	76.69	78.20

Table 6.1: Comparison of the accuracies for Naive Bayes, Multinomial Naive Bayes, SVM, Random Forest and Multilayer Perceptron methods (Dataset I). Boldface: best performance for a given setting (row).

Features	Count of attributes	NB	MultNB	SVM	RF	MP
unigram	1044	62.85	**71.4**	69.6	**71.4**	69.6
bigram	1262	**69.15**	67.2	67.3	67.2	68.9

Table 6.2: Comparison of the accuracies for Naive Bayes, Multinomial Naive Bayes, SVM, Random Forest and Multilayer Perceptron methods (Dataset II.1). Boldface: best performance for a given setting (row).

Comparing Naive Bayes method and Multinomial Naive Bayes method between themselves, we obtain that Multinomial Naive Bayes method produces a more high classification accuracy. It can be explained as follows.

The Naive Bayes classifier is a simple probabilistic classifier which is based on Bayes theorem with strong and naive independence assumptions. It is one of the most basic text

	Split	Features	Count of attributes	NB	MultNB	SVM	RF	MP
(1)	50/50	unigram	1006	64.32	71.11	**71.8**	NA	NA
(2)		bigram	1040	59.45	**62.96**	62.91	NA	NA
(3)	60/40	unigram	1006	64.95	71.68	**72.68**	NA	NA
(4)		bigram	1040	59.67	62.42	**63.28**	NA	NA
(5)	70/30	unigram	1006	65.56	71.8	**73.58**	NA	NA
(6)		bigram	1040	60.04	63.62	**64.36**	NA	NA
(7)	80/20	unigram	1006	65.87	71.87	**73.2**	NA	NA
(8)		bigram	1040	58.9	63.33	**64.2**	NA	NA

Table 6.3: Comparison of the accuracies (in percent) for Naive Bayes, Multinomial Naive Bayes, SVM, Random Forest and Multilayer Perceptron methods. Boldface: best performance for a given setting (row) (Dataset II.2).

	Split	Features	Count of attributes	NB	MultNB	SVM	RF	MP
(1)	50/50	unigram	1080	62.18	**69.85**	65.66	68.91	66.72
(2)		bigram	1212	52.41	55.18	**57.97**	56.75	56.39
(3)	60/40	unigram	1080	62.46	**70.97**	65.59	67.89	65.35
(4)		bigram	1212	50.20	57.04	57.97	**60.31**	53.91
(5)	70/30	unigram	1080	62.06	69.1	67.99	**69.82**	66.75
(6)		bigram	1212	50.85	57.11	59.71	58.93	**66.75**
(7)	80/20	unigram	1080	66.44	69.89	**70.28**	70.09	69.70
(8)		bigram	1212	51.42	59.82	**63.05**	59.92	55.23

Table 6.4: Comparison of the accuracies (in percent) for Naive Bayes, Multinomial Naive Bayes, SVM, Random Forest and Multilayer Perceptron methods (Dataset III). Boldface: best performance for a given setting (row).

classification techniques. Naive Bayes is convenient since it can be trained very fast. And also it can be used when we have limited resources in terms of CPU and Memory. Naive Bayes method performs well in many complex tasks: sentiment detection and sentiment classification, document categorization, email spam detection, personal email sorting, language detection and so on. Despite the naive design and oversimplified assumptions that this technique uses.

Multinomial Naive Bayes method is a specialized version of Naive Bayes that is designed more for text documents. Whereas Naive Bayes method simulates a document as the presence and absence of particular words, Multinomial Naive Bayes explicitly models the word counts and adjusts the underlying calculations to deal with in. Generally, Multinomial Naive Bayes is used when the multiple occurrences of the words has the meaning. Such an example is when we try to perform classification of text and sentiment classification, in particularly. This method estimates the conditional probability of a particular word (token) given a class as the relative frequency of token w in tweets of dataset belonging to defined class c (for example, positive class). Thus this method takes into account the number of occurrences of token w in training documents from c class, including multiple occurrences.

As we can see from the Tables 6.1, 6.2, 6.3, 6.4 the Multinomial Naive Bayes method gives more accurate results than Naive Bayes. But we can note that results of Naive Bayes method increase when we increase size of training dataset.

Also if we look at the performance of SVM which also gave a notable result we can see that the increase of size of training dataset also improved the overall performance for Dataset II.2, which includes bigger count of tweets (15 000 tweet), in more degree than for rest datasets. In our work we carried out experiment on one dataset to compare which kernel type of SVM method produces better result of sentiment classification. And we found out that linear kernel represents better results is fit and for our datasets.

Random Forest method and Multilayer Perceptron method also show good performance for all datsets. But the results obtained with using these methods were lower than the results of SVM classifier and Multinomial Naive Bayes classifier. Random Forest method shows higher accuracy on classification of datasets with unigrams features.

Referring to the research question №1, by analyzing the results of sentiment classification shown in Tables 6.1, 6.2, 6.3 and 6.4 we can see that multinomial naive Bayes and SVM give best results for all datasets. However SVM could give higher result in case additional analysis and selecting optimal parameters for classification.

Looking in retrospect we could have done another experiment involving deep learning with neural networks or using other, much more complex, techniques involving dynamic memory modules and so on which are lately are using for classification big data.

In this work the different methods have been analysed and applied to sentiment analysis of data. The performance of an particular model very much depends on the data. The first research question is about standard machine learning techniques that will have the best performance for sentiment classification. The second research question is about data preprocessing techniques. This two research questions have a crossing point as generally the classifier performance greatly depends on the model representation of data.

The research question №2 throughout this work was defining the effects of preprocessing techniques available for representation of natural language texts, which would have a positive resulting impact on the learning of classifier in terms of accuracy. We used the following vector models (features) available for analysis:

- unigrams (Bag-of-Words vector model)
- bigrams (Bag-of-N-grams vector model)
- trigrams (Bag-of-N-grams vector model)

By comparing the performance on different features (vector models for text representation), we find out that the selection of features are most significant for the sentiment classification tweets on the different sample sizes (dataset sizes), and least significant for the classification of sentiment which associate with the number of classes for sentiment classification. We also observe that the simplest features, namely Bag-of-Words model features (or unigrams features), in most cases produces the best performance (see Figures 6.1, 6.2, 6.3, 6.4).

The size of dataset affects the sentiment classification accuracy. The more sample size the more often single words and phrases are repeated. Hence that the number of unigrams, bigrams and trigrams significantly reduced on a large dataset.

In our experimentation we use the Dataset I which consists of 1 286 tweets. After its representation in vector forms we got 2 897 unigrams, 6 404 bigrams and 6 571 trigrams.

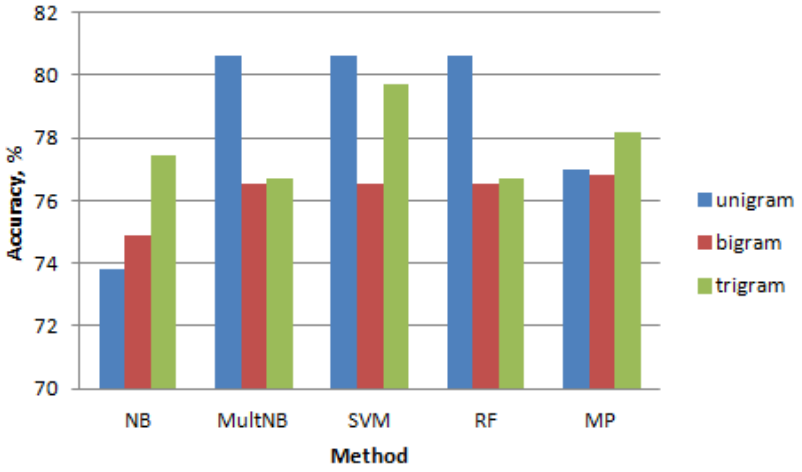


Figure 6.1: Classification accuracy for Naive Bayes, Multinomial Naive Bayes, SVM, Random Forest and Multilayer Perceptron methods on Dataset I

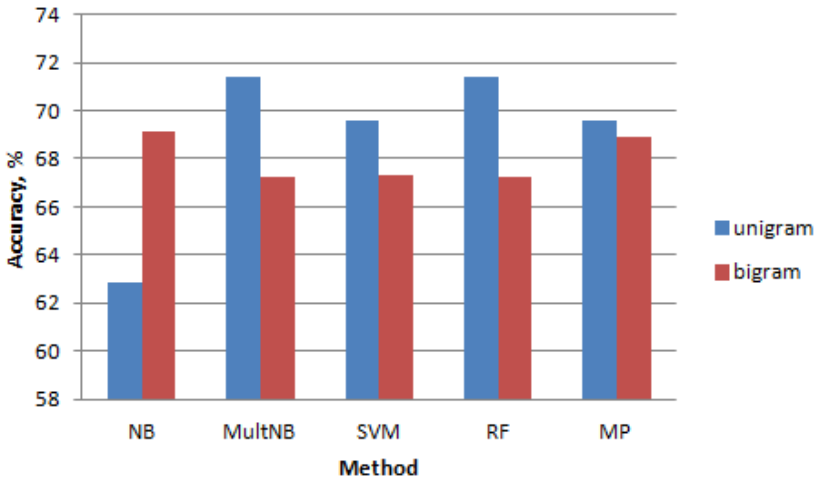


Figure 6.2: Classification accuracy for Naive Bayes, Multinomial Naive Bayes, SVM, Random Forest and Multilayer Perceptron methods on Dataset II.1

As we can see from Table 6.1 and Figure 6.1 the large number of these features had a impact to results of classification.

The best accuracy for this dataset obtained for unigrams equals 80.60% using Multinomial Naive Bayes, SVM and Random Forest methods. It significantly differ from results obtained on more large datasets which after their representation into vector models included much less number of features.

The most large dataset (Dataset II.2 consists of 15 000 tweets) in our study includes just 1 006 unigrams and 1 040 bigrams. The best accuracy for this dataset obtained for unigrams equals 73.58% using the SVM method.

For rest of datasets the using of unigrams (Bag-of-Words vector model of representation of the text data) gave the higher accuracy than using bigrams features (Bag-of-N-grams

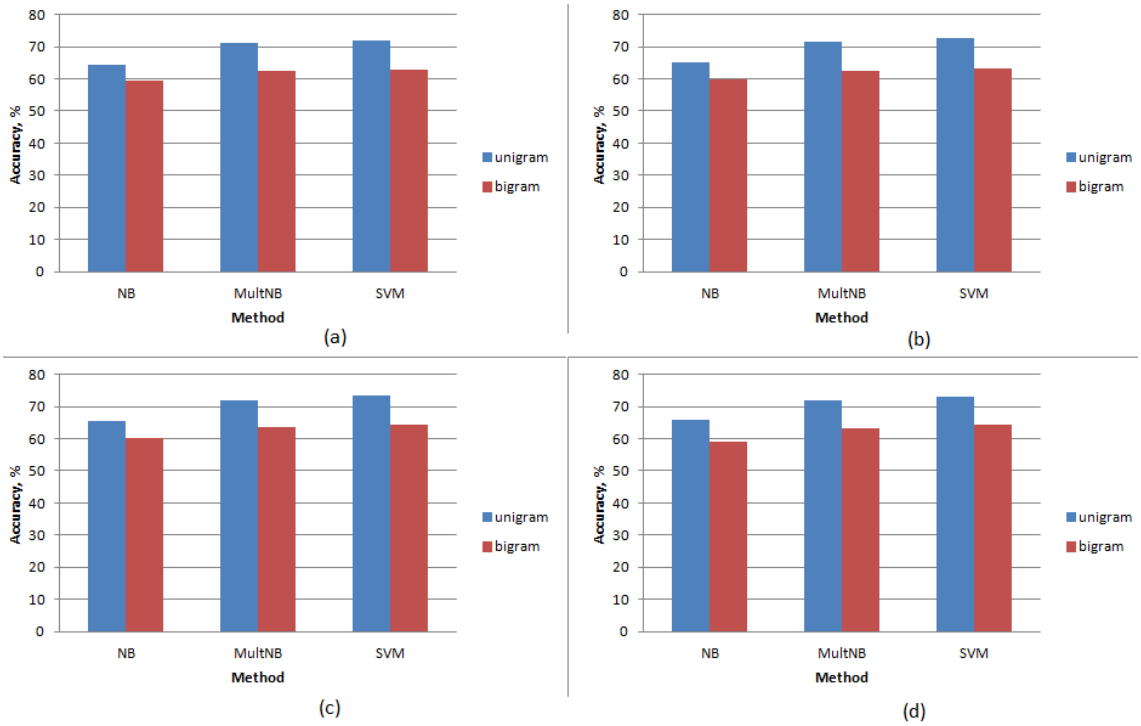


Figure 6.3: Classification accuracy for Naive Bayes, Multinomial Naive Bayes, SVM, Random Forest and Multilayer Perceptron methods on Dataset II.2.
(a) – dataset is splited into 50% for training and 50% testing sets; (b) – dataset is splited into 60% for training and 40% testing sets; (c) – dataset is splited into 70% for training and 30% testing sets; (d) – dataset is splited into 80% for training and 20% testing sets

vector model of representation of the text data).

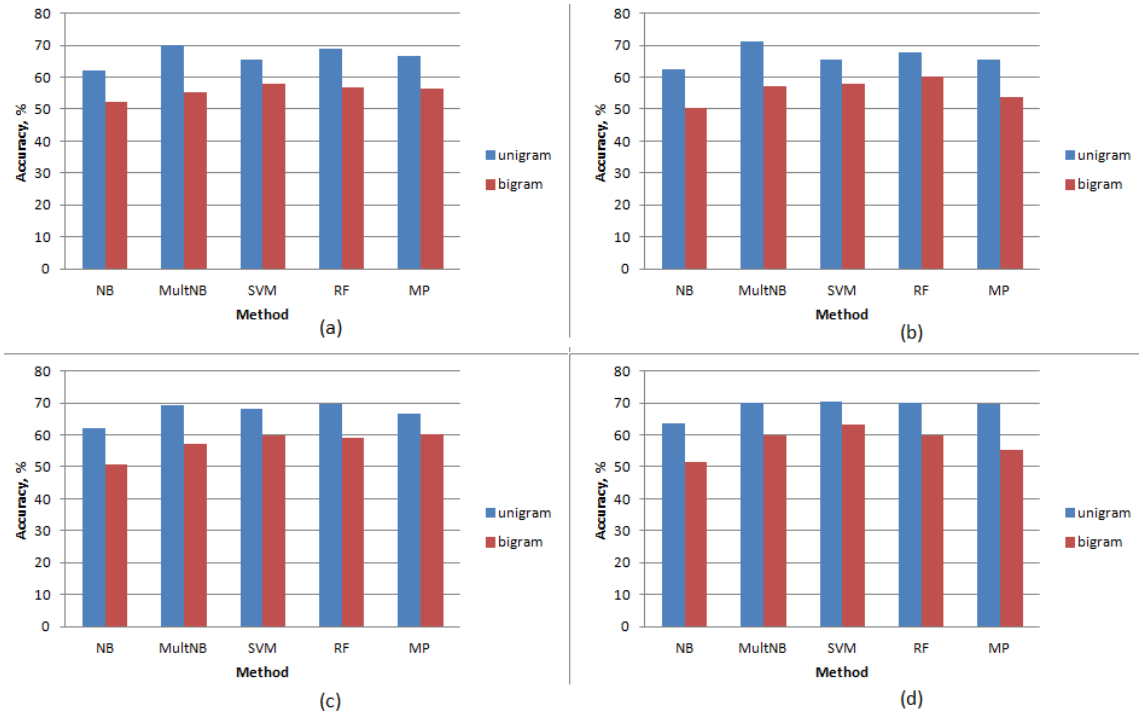


Figure 6.4: Classification accuracy for Naive Bayes, Multinomial Naive Bayes, SVM, Random Forest and Multilayer Perceptron methods on Dataset III.
(a) – dataset is splited into 50% for training and 50% testing sets; (b) – dataset is splited into 60% for training and 40% testing sets; (c) – dataset is splited into 70% for training and 30% testing sets; (d) – dataset is splited into 80% for training and 20% testing sets

Further Work

Deep learning is one of the domains that we have not considered during our work. However a lot of authors, as we have mentioned in Chapter 3 have used deep learning methods for sentiment analysis of text. Currently, deep learning algorithms show better results for many machine learning tasks: computer vision, speech recognition, text processing and natural language processing. A big number of recent studies described in Chapter 3 demonstrate the superiority of deep learning models in the field of English texts for sentiment analysis over shallow algorithms, which include linear and logistic regression, Bayesian classifier, as well as widely used SVM method. Moreover, the results of Deep Learning approach set new records classification accuracy in tasks and normal aspect of sentiment analysis for English texts. Therefore we could use these methods for sentiment analysis in microblog area. Another promising technique is combining machine learning techniques and approaches based on lexicon.

In case representation text as vector models could be also used another techniques taking into account the frequency with which a term appears in the collection of tweets. Examples are the model TF (Term Frequency – the frequency of the term) and TF-IDF (Term Frequency – Inverse Document Frequency).

Chapter 7

Conclusion

This work represents case study and aims to compare standard machine learning techniques applied to sentiment analysis of social media – specifically to the sentiment classification of the data gathered from social network Twitter – and to find the best performing method for three datasets various sizes and from different domains collected from Twitter.

To answer research questions we conducted the experiments. And since we have datasets of different sample size with tweets from different domains, experiments for each dataset were carried out. Still main tendencies was the same for all datasets. To answer the research question and find the most accurate method for classication these datasets, on the first place the data were analyzed and before applying machine learning techniques to text data, the data were represented in the vector form. In purpose to find the best data representation, experiments were conducted and Bag-of-Words model (vector of unigrams) and Bag-of-N-grams model (vector of bigrams and vector of trigrams) of representation of the text were used.

The one of research questions was about the data itself, which techniques for the preprocessing of text data can be used to provide the best foundation for classifiers. To find the best data representation, all datasets were represented in vector models: Bag-of-Words model (unigrams) and Bag-of-N-grams model (bigrams and trigrams). Bag-of-Words model (unigrams) of representation of the datasets showed the best results for all methods and influenced in a positive way improving the overall accuracy of the machine learning techniques than Bag-of-N-grams model (bigrams and trigrams).

The primary research question is about comparing machine learning models for sentiment classification of the datasets collected from Twitter and than analysing the outcome. Several machine learning methods were used during experimentation session: Naive Bayes, Multinomial Naive Bayes, Support Vector Machines, Random Forest, Multilayer Perceptron Network. All the methods were compared in terms of accuracy. As turned out all the machine learning models that have been applied to these datasets gave good performance results. However, the best performance achieved Multinomial Naive Bayes and Support Vector Machines methods on all datasets. The accuracy of Multinomial Naive Bayes performed by Bag-of-Words model is 80.6% for Dataset I, 71.4% for Dataset II.1, 71.87% for Dataset II.2, and 70.97% for Dataset III. SVM with linear kernel was applied for the sentiment classication on all the datasets. The accuracy of SVM performed by Bag-of-Words model is 80.6% for Dataset I, 69.6% for Dataset II.1, 73.2% for Dataset II.2, and 70.28% for Dataset III. These values of performance are higher than values obtained by the rest methods.

The main point for further work is to use recurrent neural networks, particularly LSTM (Long short-term memory) model and deep learning models for experimentation. The resulting vectors of text data representation could be used if necessary as part of deep learning models. Regarding representation text as vector models could be also used another techniques taking into account the frequency with which a term appears in the collection of tweets. Examples are the model TF (Term Frequency - the frequency of the term) and TF-IDF (Term Frequency - Inverse Document Frequency).

Bibliography

[1] Affective norms for english words (anew). http://csea.phhp.ufl.edu/media/anewmessage.html. December 2015.

[2] Decision trees. http://webtutplus.com/decision-trees/. April 2016.

[3] Mathwork documentation. feedforward neural network. http://se.mathworks.com/help/nnet/ref/feedforwardnet.html.

[4] Mysql, the official homepage. http://www.mysql.com/. December 2015.

[5] The penn treebank project. https://www.cis.upenn.edu/~treebank/. April 2016.

[6] scikit-learn. machine learning in python. http://scikit-learn.org/stable/. November 2015.

[7] scikit learn. scikit-learn: Supervised learning. http://scikit-learn.org/stable/supervised_learning.html#supervised-learning. November 2015.

[8] Sentiwordnet. http://sentiwordnet.isti.cnr.it/. December 2015.

[9] Twitter. https://about.twitter.com/company. March 2016.

[10] Twitter api documentation. https://dev.twitter.com/overview/documentation. February 2016.

[11] Twitter sentiment corpus. http://www.sananalytics.com/lab/twitter-sentiment/. September 2015.

[12] Weka 3: Data mining software in java. http://www.cs.waikato.ac.nz/ml/weka/. September 2015.

[13] Wordnet project homepage. http://wordnet.princeton.edu/. December 2015.

[14] Mariam Adedoyin-Olowe, Mohammed Medhat Gaber, and Frederic Stahl. A survey of data mining techniques for social network analysis. *Journal of Data Mining & Digital Humanities*, 2014.

[15] Fotis Aisopos, George Papadakis, and Theodora Varvarigou. Sentiment analysis of social media content using n-gram graphs. In *Proceedings of the 3rd ACM SIGMM international workshop on Social media*, pages 9–14. ACM, 2011.

[16] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, volume 10, pages 2200–2204, 2010.

[17] Akshat Bakliwal, Piyush Arora, Senthil Madhappan, Nikhil Kapre, Mukesh Singh, and Vasudeva Varma. Mining sentiments from tweets. *Proceedings of the WASSA*, 12, 2012.

[18] Adam Bermingham. *Sentiment analysis and real-time microblog search.* PhD thesis, Dublin City University, 2011.

[19] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[20] John Blitzer, Mark Dredze, Fernando Pereira, et al. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *ACL*, volume 7, pages 440–447, 2007.

[21] Margaret M Bradley and Peter J Lang. Affective norms for english words (anew): Instruction manual and affective ratings. Technical report, Technical report C-1, the center for research in psychophysiology, University of Florida, 1999.

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

[23] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM, 2011.

[24] Olivier Chapelle. Support vector machines et classification d'images. 1998.

[25] William B Claster, Malcolm Cooper, and Philip Sallis. Thailand–tourism and conflict: Modeling sentiment from twitter tweets using naïve bayes and unsupervised artificial neural nets. In *Computational Intelligence, Modelling and Simulation (CIMSiM), 2010 Second International Conference on*, pages 89–94. IEEE, 2010.

[26] Luiz FS Coletta, Nadia FF da Silva, and Estevam R Hruschka. Combining classification and clustering for tweet sentiment analysis. In *Intelligent Systems (BRACIS), 2014 Brazilian Conference on*, pages 210–215. IEEE, 2014.

[27] Nadia FF da Silva, Eduardo R Hruschka, and Estevam R Hruschka. Tweet sentiment analysis with classifier ensembles. *Decision Support Systems*, 66:170–179, 2014.

[28] R Renuga Devi and M Hemalatha. A novel approach for secure hidden community mining in social networks using data mining techniques. *International Journal of Computer Applications*, 87(7), 2014.

[29] Nicholas A Diakopoulos and David A Shamma. Characterizing debate performance via aggregated twitter sentiment. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1195–1198. ACM, 2010.

[30] J. Donaldson. Beautiful decisions: Inside bigml's decision trees. `http://blog.bigml.com/2012/01/23/beautiful-decisions-inside-bigmls-decision-trees/`. April 2016.

[31] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1:12, 2009.

[32] Balakrishnan Gokulakrishnan, Pavalanathan Priyanthan, Thiruchittampalam Ragavan, Nadarajah Prasath, and Amitha Perera. Opinion mining and sentiment analysis on a twitter data stream. In *Advances in ICT for Emerging Regions (ICTer), 2012 International Conference on*, pages 182–188. IEEE, 2012.

[33] Alexander Gross and Dhiraj Murthy. Modeling virtual organizations with latent dirichlet allocation: A case for natural language processing. *Neural Networks*, 58:38–49, 2014.

[34] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[35] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification. 2003.

[36] José Antonio Iglesias, Alexandra Tiemblo, Agapito Ledezma, and Araceli Sanchis. Web news mining in an evolving framework. *Information Fusion*, 28:90–98, 2016.

[37] Akshay Java. Mining social media communities and content, 2008.

[38] Monisha Kanakaraj and Ram Mohana Reddy Guddeti. Performance analysis of ensemble methods on twitter sentiment analysis using nlp techniques. In *Semantic Computing (ICSC), 2015 IEEE International Conference on*, pages 169–170. IEEE, 2015.

[39] Ioannis Kanaris, Konstantinos Kanaris, Ioannis Houvardas, and Efstathios Stamatatos. Words versus character n-grams for anti-spam filtering. *International Journal on Artificial Intelligence Tools*, 16(06):1047–1067, 2007.

[40] Andreas M Kaplan and Michael Haenlein. Users of the world, unite! the challenges and opportunities of social media. *Business horizons*, 53(1):59–68, 2010.

[41] Simon Kemp. Digital, social and mobile worldwide in 2015, special report. `http://wearesocial.com/uk/special-reports/digital-social-mobile-worldwide-2015`. March 2016.

[42] David Khanaferov, Christopher Luc, and Taehyung Wang. Social network data mining using natural language processing and density based clustering. In *Semantic Computing (ICSC), 2014 IEEE International Conference on*, pages 250–251. IEEE, 2014.

[43] Soo-Min Kim and Eduard Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1367. Association for Computational Linguistics, 2004.

[44] Moshe Koppel and Jonathan Schler. The importance of neutral examples for learning sentiment. *Computational Intelligence*, 22(2):100–109, 2006.

[45] Adam DI Kramer, Jamie E Guillory, and Jeffrey T Hancock. Experimental evidence of massive-scale emotional contagion through social networks. *Proceedings of the National Academy of Sciences*, 111(24):8788–8790, 2014.

[46] Samuel C Lee and Edward T Lee. Fuzzy neural networks. *Mathematical Biosciences*, 23(1):151–177, 1975.

[47] Ana Carolina ES Lima and Leandro Nunes De Castro. A multi-label, semi-supervised classification approach applied to personality prediction in social media. *Neural Networks*, 58:122–130, 2014.

[48] Carlo Lipizzi, Luca Iandoli, and José Emmanuel Ramirez Marquez. Extracting and evaluating conversational patterns in social media: A socio-semantic analysis of customers' reactions to the launch of new products using twitter streams. *International Journal of Information Management*, 35(4):490–503, 2015.

[49] Bing Liu. Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2:627–666, 2010.

[50] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[51] Iris B Mauss and Michael D Robinson. Measures of emotion: A review. *Cognition and emotion*, 23(2):209–237, 2009.

[52] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[53] R Muhamedyev, K Yakunin, S Iskakov, S Sainova, A Abdilmanova, and Y Kuchin. Comparative analysis of classification algorithms. In *Application of Information and Communication Technologies (AICT), 2015 9th International Conference on*, pages 96–101. IEEE, 2015.

[54] Alexander Pak. *Automatic, adaptive, and applicative sentiment analysis*. PhD thesis, Citeseer, 2012.

[55] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, pages 1320–1326, 2010.

[56] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[57] Rudy Prabowo and Mike Thelwall. Sentiment analysis: A combined approach. *Journal of Informetrics*, 3(2):143–157, 2009.

[58] Joshua Ritterman, Miles Osborne, and Ewan Klein. Using prediction markets and twitter to predict a swine flu pandemic. In *1st international workshop on mining social media*, volume 9, pages 9–17. ac. uk/miles/papers/swine09. pdf, 2009.

[59] Matthew A Russell. *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More.* " O'Reilly Media, Inc.", 2013.

[60] Hassan Saif, Yulan He, and Harith Alani. Semantic sentiment analysis of twitter. In *The Semantic Web–ISWC 2012*, pages 508–524. Springer, 2012.

[61] Beatrice Santorini. Part-of-speech tagging guidelines for the penn treebank project (3rd revision). 1990.

[62] Klaus R Scherer. Vocal communication of emotion: A review of research paradigms. *Speech communication*, 40(1):227–256, 2003.

[63] David A Shamma, Lyndon Kennedy, and Elizabeth F Churchill. Tweet the debates: understanding community annotation of uncollected sources. In *Proceedings of the first SIGMM workshop on Social media*, pages 3–10. ACM, 2009.

[64] Anuj Sharma and Shubhamoy Dey. A document-level sentiment analysis approach using artificial neural network and sentiment lexicons. *ACM SIGAPP Applied Computing Review*, 12(4):67–75, 2012.

[65] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer, 2013.

[66] Michael Speriosu, Nikita Sudan, Sid Upadhyay, and Jason Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the First Workshop on Unsupervised Learning in NLP*, EMNLP '11, pages 53–63, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[67] Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307, 2011.

[68] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*, 2015.

[69] DS Tarasov. Deep recurrent neural networks for multiple language aspect-based sentiment analysis of user reviews. In *Proceedings of the 21st International Conference on Computational Linguistics Dialog-2015*, volume 2, pages 77–88, 2015.

[70] Mike Thelwall, David Wilkinson, and Sukhvinder Uppal. Data mining emotion in social network communication: Gender differences in myspace. *Journal of the American Society for Information Science and Technology*, 61(1):190–199, 2010.

[71] Kimitaka Tsutsumi, Kazutaka Shimada, and Tsutomu Endo. Movie review classification based on a multiple classifier. In *Proceedings of the annual meetings of the Pacific Asia conference on language, information and computation (PACLIC)*, pages 481–488, 2007.

[72] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

[73] Saúl Vargas, Richard McCreadie, Craig Macdonald, and Iadh Ounis. Comparing overall and targeted sentiments in social media during crises. In *Tenth International AAAI Conference on Web and Social Media*, 2016.

[74] Bishan Yang and Claire Cardie. Joint inference for fine-grained opinion extraction. In *ACL (1)*, pages 1640–1649, 2013.

[75] Christopher C Yang, Xuning Tang, YC Wong, and Chih-Ping Wei. Understanding online consumer review opinions with sentiment analysis using machine learning. *Pacific Asia Journal of the Association for Information Systems*, 2(3), 2010.

[76] Reza Zafarani, Mohammad Ali Abbasi, and Huan Liu. *Social media mining: an introduction*. Cambridge University Press, 2014.

[77] Xue Zhang, Hauke Fuehres, and Peter A Gloor. Predicting stock market indicators through twitter "i hope it is not as bad as i fear". *Procedia-Social and Behavioral Sciences*, 26:55–62, 2011.