Student: Emily Velzeboer
Supervisor: Prof. Hannu Salmela
Master's Thesis title: Twitter Sentiment Analysis on Product Reviews

ABSTRACT

We live in an era where large amounts of information, the so-called Big Data, are exploited to find patterns, trends, and make predictions.
Sentiment analysis deals with analyzing large amounts of text for the purpose of catching the mood and feeling about something, e.g., a product or an event. In particular, this thesis applies sentiment analysis techniques to classify product reviews on the Twitter social media platform, with the goal of understanding what the general feeling about a newly-launched product is. This work focuses on a recently launched line of running shoes by American sport equipment manufacturer Nike Inc.
The methodology used in this work is rooted in the field of machine learning: data is preprocessed to make it easily exploitable, and then standard techniques and algorithms like Naïve Bayes classifier and Support Vector Machine are employed to train classifiers and make predictions.
Beyond the comparison that is drawn between several techniques, the originality of this work lies in the fact that a generic training set is shown to give good results when used to train classifiers that will be applied to the specific problem of classifying product reviews. In other words, this work shows that the text elements that define and identify sentiment are constant across different use cases, and that generic texts can be used to extract knowledge which can be successfully exploited for a specific problem.
Moreover, this work presents an innovative treatment of neutral tweets, i.e., those that do not contain a positive or negative feeling. No specific training is done in this case, but the classifier is shown to be able to identify them in many instances, despite the fact that the training set only contains positive and negative samples. To this end, a different approach is used, based on the subjectivity of the text itself, under the assumption that a piece of text that conveys a feeling will be subjective, whereas a neutral piece of text will tend to be more objective.

TWITTER SENTIMENT ANALYSIS ON PRODUCT REVIEWS

A comparison between two machine learning techniques

Master's Thesis
In International Master of Management
of Information Technology – IMMIT

Author: Emily Velzeboer

Supervisor: Prof. Hannu Salmela

01.06.2019
Nice

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI - Artificial Intelligence
ME - Maximum Entropy
NB - Naive Bayes
NLP - Natural Language Processing
SA - Sentiment analysis
SO - Sentiment orientation
SVM - Support Vector Machine
VOC - Voice of the consumer

1 INTRODUCTION

"Your most unhappy customers are your greatest source of learning."
(Bill Gates)

Recent researches suggest we generate 2.5 quintillion (that is 25 followed by seventeen zeros) bytes of data every day (IBM, 2012). This means that the traffic of data is so intense on the web nowadays, that the amount of available information is hard even to imagine. This information can be data under any form: text documents, images, audio clips, software programs, or other types of data. From this large amount of data available that characterizes the era we live in, a new phenomenon was born: the phenomenon of "Big data". Big data consists of an enormous amount of data analyzed computationally in a way to gain insights, reveal patterns, trends and associations (Moreno and Redondo, 2016). Now more than ever, knowledge and information are some new kind of currency, and being able to exploit them is a key towards success in many fields. Some examples about this are easy to come by: it is sufficient to think about a company about to launch a new product in the market, or a man about to enter politics, or even a politician about to propose a law, or again an investor about to put money in some stocks. What is going to happen? As mysterious as the future is, a deeper understanding of what is going on around us right now can help us to predict with a higher degree of confidence what will happen tomorrow, thus steering us towards the most profitable direction.

Gathering information today is not difficult at all: people are now more than happy to share their thoughts through the Internet, especially on social media. In the last few years, this new form of communication has become a central part in our lives, with billions of unique users connecting every day to platforms such as Facebook, Twitter, Instagram, and so on. They have a worldwide coverage, they are able to reach every corner of the planet and, as a consequence, collect the voice of the people from all over the world. Concisely, social networks provide a large amount of real-time information that is available to everyone. During political elections, sporting events or movie film programming, a vast knowledge can be acquired about these events and in real time. To give some numbers, in April 2018 the social networks Facebook and Twitter counted over 2234 million and 330 million active users (We are social, 2018). This suggests a massive data flow: for example, users on Twitter can generate 200 million comments in a single day and 9,000 comments per second (Mashable.com, 2011). On a more daily basis, these new forms of communication influence people's normal lives, habits, and the way they form an opinion on a matter. In fact, many users acquire their knowledge

on a topic according to what they found on the web, for example they search for reviews before buying a book, eating in a restaurant, or watching a film. For example, the number of users searching online to acquire information is increasing and it influences customers' shopping behavior: according to a study made by Bazaarvoice, 45% of customers read reviews online before making a purchase in a store, and 87% consult their smartphones on a product while they are in a store (Ellett, 2019).

Social media have also had an undeniable impact on the language, fostering new ways of communication. The massive use that every day we make of emoticons and emojis is just an example of this revolution in act. Thanks to these little pictorial representations, we convey a wide range on feelings, or even fully-fledged opinions. For example, how many times have we seen a thumb pointed upward as a reply to something? Although very simple, this is already an example of a little image taking the place of a sentence like "I agree with what you have said".

Moreover, the words themselves have changed considerably since the introduction of social media. It must be noted that, many times, contents are entered from portable devices, with tiny keyboards that make typing long and complicated texts quite uncomfortable and time-consuming. For this reason, abbreviations are more and more widely used, together with slang words (Kouloumpiset al., 2011). Some examples are "thx" for "thanks", "c u" instead of "see you", "ty" for "thank you", "4" instead of "for", and the list could continue. Also, we have become increasingly accustomed to typos, whose presence is greatly favored by typing on small keyboards. Beside plain typos, it is not rare to see some grammatical rules infringed on purpose: typing an apostrophe might require changing the keyboard layout to special symbols and, thus, slow down the user who chooses to type "dont" and "doesnt" instead of "don't" and "doesn't", or "its" for both "its" and "it's", or "Ill" for both "Ill" the synonym of unhealthy and "I'll". This is all done trusting the fact that the context will help the reader understand what exactly the writer wanted to say. Precisely, in interpreting this context lies one of the biggest challenges of this research, and it will become clear in the following of this work.

Anyhow, this data is publicly available to anyone on the internet, and it reflects the true feelings of the people. In this regard, it has been noted that people tend to express their opinions more freely when sharing contents online, because they feel protected by the virtual layer between them and the reader. This led to the phenomenon known as "keyboard warriors", where people express online opinions and feelings that they would never feel comfortable (or even dare) exposing directly to a listener. Again, this has pros

and cons, because it helps people feel free to speak their mind as much as it exposes the worst and most violent side of many users' personality.

As previously stated, big amount of information is a true goldmine when trying to analyse the current situation and to predict its evolution. However, accessing this data and taking advantage of it are two completely different things. Thus, such a vastness also poses practical problems: how to collect the data in an automated way? This can be done in several ways, but usually this task is simplified directly by the platforms, which expose APIs (Application Programming Interfaces) that essentially give researchers and developers a way to programmatically gain access to the content shared on that platform, so that it can become the basis for the product they are working on (Shirdastian, 2017). Then, how to store the data? Clearly, normal hard drives are not even remotely enough for holding the amount of available information. How to categorize the data so that searching through them is as easy and quick as possible? How to aggregate the data so that pieces of information coming from different sources but related to the same topic can work together to form a bigger picture? Evidently, analysis through human operators is out of the picture as totally infeasible. Then, automated frameworks for dealing with this new challenge have to be put in place so that questions can be answered thanks to the knowledge spread around this ocean of contents.

For this reason, it should not be surprising that a new area called business intelligence has surfaced and is gaining more and more importance. The goal of business intelligence is organizing information and taking advantage of it, in order to guide businesses in taking the right strategic decisions. It also aims at finding patterns and novels behind the historical data and to build recommendations based on them. There are many companies like IBM, SAS, Microsoft, Tableau, that provide tools for analysing big data, by using techniques of data mining and text mining.

Data mining is a discipline that comes from several other disciplines and was born thank to the growth of large databases. It can be translated as "Knowledge mining from data" and it provides a set of methodologies and automatic or semi-automatic techniques for extracting useful information from a large amount of data (Han et al., 2012). Data mining is generic and it usually refers to information given in a numeric form, while text mining is more specific utilizing words or sentences as data. Text mining is needed for information overload, in social media but also scientific articles. It is important to extract key events, relations and detect sentiment, but developing a text mining tool is very complex as it takes a considerable amount of time, an important financial effort, and it is not enough accurate (Moreno and Redondo, 2016).

However, text mining is also an evolving field of study, with researchers actively looking for more efficient and effective methodologies and techniques. Google, Microsoft and IBM are among the largest companies that have done more research in this field and are still in the research phase to better improve their service. What these three big companies mainly offer is an artificial intelligence (AI) service called Natural Language Processing (NLP).

NLP is a subfield of computer science, with ramifications in artificial intelligence, that deals with how computer can understand human language (also called "natural language") (Moreno and Redondo, 2016). In fact, knowing the meaning of any possible word is of no help if one does not know what they mean when combined together. Moreover, human language is full of subtleties, like irony, sarcasm, puns, double negations, especially on the internet and on the social media. In some cases, the meaning of a message could depend on a conversation that has already taken place between the parties, thus making its comprehension even harder for a third party. As if this was not enough, even a perfectly stated sentence can have its meaning completely reversed by just adding one of those emoticons mentioned earlier. A simple example is the following: consider the messages "I really hate you" and "I really hate you :P". Both of them contain the words "I really hate you", which give the message a very negative tone. However, the latter one also has the pictogram ":P", which is commonly accepted as a representation of a tongue stuck out of one's mouth. This completely changes the meaning of the message and suggests it should be interpreted more as a tongue-in-cheek statement, rather than one of pure hatred. It is then easy to understand how hard it can be to program a machine to do all these reasoning and deduce the true meaning of a message.

Applications of NLP are not hard to imagine: automatic parsing of texts, virtual assistants capable of answering questions, and, more in general, a better integration between man and machine, towards a future where the former can talk seamlessly together with the latter.

In the context of this work, we are mostly interested in the potential of NLP for processing large amounts of data and understanding what is the general feeling that is being expressed about something. Going back to an example already cited, let us assume a company has launched a new product and wants to know what its customers' opinion is. Instead of hiring someone to write well-crafted questions, find volunteers for focus groups, submit the questionnaires, collect the answers, and elaborate conclusions, the firm could simply listen to what people are saying about the product on the social media. Filtering the information, in this case, is not a very hard task: it should be sufficient to

take into account all the messages that mention the name of the product. But then these messages have to be understood, in order to give an answer to the question "What do people think about it?".

The importance of the answer's accuracy is immediate: if the result is that people are enthusiast about a product, this could push a company to invest more in it, to study a new version, etc. On the contrary, if the feeling is that customers do not like the product, then the company might consider abandoning the project or investing less in it. These are important inputs based on facts and on carefully designed techniques that drive decision making. It consequently gives the company the possibility of "staying in touch" with its customer, to develop the so-called Voice of the consumer (VOC) and then to prevent costly mistakes from happening (Lee et al., 2014).

The scenario just outlined is something more specific than generic NLP. It is more about categorizing opinions expressed in texts, rather than understanding exactly the content of the message. For example, is the message conveying a positive or a negative feeling? Is it enthusiastic, pessimistic, full of hope, discouraged, …? The area that studies techniques to answer these questions is called Sentiment Analysis (SA), that is "*the field of study that analyzes people's opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes*." (Liu, B., 2012). In short, the goal of sentiment analysis is to try to make sense of big data in order to understand the general opinion, i.e., the personal, subjective sentiment about something.

For business-related purposes, this is a sort of online monitoring market, where opinions and comments of consumers are analysed to understand their needs and adapt the offer. This online monitoring market is also called opinion mining; however, there is a difference between "sentiment analysis" and "opinion mining": sentiment analysis is based on a judgment prompted by a feeling, usually classified as negative, positive, or neutral, whereas opinion mining is based on understanding a broader view, a judgment, or an opinion about a particular matter.

Several works have already been proposed in the field of SA, and they will be described more thoroughly in the following of this manuscript. However, despite the research efforts, this remains a challenging problem, due to the issues mentioned above. Also, sometimes it is not even easy to assess the accuracy of a tool that performs sentiment analysis, and this is because even two distinct humans might disagree on the meaning of a particular sentence. An example could be "Oh my God, look at that!": without more information, it is impossible to say if this message expresses a positive or a negative feeling. We can tell for sure that it expresses surprise, but we cannot say if

the surprise was a pleasant one or not. This makes the task even more difficult, and explains why even the best systems do not achieve impressive figures when it comes to accuracy.

In this work, we will tackle the problem of sentiment analysis on product reviews taken from the platform of Twitter. Twitter is very used in terms of product reviews, especially for newly launched products, as it clearly shows real-time opinions and comments. The consumers' feelings that are conveyed in its short messages will be extracted and analyzed. In particular, we will try to build a classification system on the basis of generic tweets (this is how posts on Twitter are called), whose content might be anything, about any kind of topic, and then apply this system to tweets whose content is very specific, like reviews of a particular product.

1.1 Research questions and contribution

Sentiment analysis and NLP are not a new research area, but they remain fields of interest in the scientific academic research and in the business community. One of the main reasons is that they can be exploited as strong and useful tools for marketing and customer service, since they allow businesses to stay in touch with customers and collect their opinions and feelings (Voice of Consumer). For these reasons, these fields are a thriving area of research and expected to be an important topic for the years to come.

However, the largest part of the research was usually on the analysis of large documents (e.g., reviews), while microblogs came into focus just over the last few years. In this thesis, Twitter microblogging platform will be utilized as a source of opinions and reviews for sentiment classification.

Classification of tweets is a demanding task because tweets can contain only a few words, lacking grammatical structure, correctness, and even context.

This research is a quantitative research that will use an experimental methodology. The research contains analysis and answers the following questions: (1) "How to measure the sentiment of a new product in the market through Twitter sentiment analysis?", (2) "How accurate is the Twitter sentiment analysis on product reviews when considering also neutral tweets (i.e., that do not contain any sentiment)?" and (3) "How accurate is the Twitter sentiment analysis on product reviews starting from a corpus of generic tweets (i.e., not related to product reviews, marketing, etc.)?".

The experimental methodology was based on applying to the set of problems at hand some well-known and recognized machine learning techniques, which have already proved effective in the literature, when applied to similar research problems. In

particular, the originality of this thesis lies in experimenting with different datasets to measure how the performance changes when keeping the initial conditions constant and experimenting on different test settings. Regarding the technical part, the support for machine learning techniques and algorithms was taken from Python and the numerous libraries and modules which are freely available, and widely used by data scientists. As it is typical in the field of machine learning, the results were measured by running the experiments in a controlled environment and comparing the answers obtained during the experiment with a set of previously known and correct answers.

1.2 Structure of the thesis

This manuscript is organized as follows:

- Section 2 contains a survey on the State of the Art about sentiment analysis, in particular in the context of social media reviews, and the main technologies available and already used in the literature;
- Section 3 contains a detailed description of the methodology used in the sentiment analysis, from the experimental research set up, the data collection, the usage of tools and modules, to the processing and the machine learning algorithms used;
- Section 4 reports the experimental results on Twitter sentiment analysis applied to a newly launched product by an important multinational company using the procedure described in section 3. It reports a comparison between the techniques used, the different processes used, and an evaluation based on the grade of accuracy;
- Section 5 includes discussions, limitations, and conclusions drawn on the basis of the experiments performed. Finally, it suggests possible directions for future research.

2 THEORETICAL BACKGROUND

This chapter is meant to provide the theoretical background to properly understand the following of the manuscripts. It is divided into two parts: in the first part we introduce what Twitter and sentiment analysis are, explaining how to identify opinions and how to select them to conduct sentiment analysis, while the second part contains a description of the methodology techniques for sentiment analysis used in the literature.

2.1 Sentiment analysis

Sentiment analysis is a set of techniques and procedures for the study and the analysis of textual information, in order to detect assessments, attitudes, and emotions related to a certain entity (product, person, topic, etc.) (Liu and Zhang, 2013). This type of analysis has important applications to the political, social, and economic fields.

Research in the field of sentiment analysis boomed over the course of a few years, with a large number of works published in the last ten years. Most of the articles focus on the techniques used in this field of analysis: some of them report an overview of the possible methods to use, whereas others apply a particular method on a case, testing it and reporting the results.

For example, Liu and Zhang (2013) proposed an extended overview of all the techniques for opinions and feelings in textual documents. In particular, the work focuses on the various processes of natural language processing (NLP), that deals with the intrinsically subjective nature of natural language. The results of these techniques applied by the various authors usually report the polarity of the sentiment conveyed in a message (e.g., positive, negative, neutral). Afterwards, in some works the validity of the proposed technique is evaluated in order to see if it can be successfully applied to the problem at hand and has a sufficiently narrow margin of error to consider the results as meaningful.

When studying sentiment analysis, the first step to perform is to understand what an opinion is, how it is formed (based also on the context), what its degree of subjectivity (or objectivity) is, and the features (parts of the text) to take into account.

In fact, not all the texts have a sentiment. Indeed, not all the texts are even genuine, as many of them are just spam. As Jindal and Liu (2008) highlighted, “there is no quality control on the Web, anyone can write anything.”. Following this observation, the authors did a case study using a set of Amazon reviews to detect the spam, trying to determine which reviews are fake and which ones are not.

Regarding the fact that not all texts contain a sentiment, it will be shown later in the dissertation that this distinction is important in order to classify neutral opinions and distinguish them from those with a specific polarity.

In the next chapter we will going to deepen the study of opinions, describing its main characteristics.

2.2 Opinions and sentiment analysis

In this section we will illustrate and generalize, through definitions and examples, what an opinion is and how it is structured.
Oxford Dictionary defines opinion as "*a view or judgement formed about something, not necessarily based on fact or knowledge*". This kind of definition is perfect from a linguistic point of view but it is too abstract and inaccurate for a computational analysis of the sentiment. Therefore, opinion will be defined in a more detailed and formal way.

Firstly, all opinions have an object. The object is the entity on which the comment is expressed, and it can be a product, a person, an organization, an event, or a topic (Liu, 2010). The object, for example a mobile phone, has components like battery, screen, antenna, and attributes like color, price, design, and so on. The entity can be even more complex: components can have sub-components, attributes can have different properties, etc. These components and attributes of the object can be simply called features.

However, the object should not be represented as an entity constituted by a hierarchy of components because:

- For the sentiment analysis, it is extremely difficult to establish a hierarchy for the analysis of a text;
- For the user, it can be difficult to think about an object in its hierarchical representation when he expresses his opinion.

A better approach is to use a non-hierarchical list of features, that includes components, attributes, and the object. Applying this approach to the previous example on the mobile phone, the features that will be considered could be touch screen, voice quality, battery life, price, and the mobile phone in general.

A further distinction given by Liu is between explicit and implicit features:

- The feature is explicit when it appears in the text;
- The feature is implicit when it does not appear in the text, but it is implied.

For example, in the comment “this phone has an exorbitant price!” the feature “price” appears explicitly in the text, while in the comment “this phone is too expensive!” the feature “price” is indicated by the word expensive but it doesn’t appear in the text, thus it is implicit.

An opinion on a feature can be of two types: comparative or direct.

- A comparative opinion expresses a similarity or a difference between two or more objects or some of their features.
- A direct opinion is when the entities are clearly expressed such as the object, the feature of the object, the polarity of the opinion (positive, negative or neutral), and the opinion-holder.

The comparative opinion is quite complex when it is about subjective opinions. These types of comments are usually studied separately from the direct opinions, as they require proper tools for the analysis of natural language.

Once the opinions are understood, it is important to identify what are the classifications done by the sentiment analysis. For example, in sentiment analysis, subjectivity and sentiment are studied in the literature as two different classification problems, although at first glance they look similar (Liu, 2010; Pang and Lee, 2008):

- The subjectivity classification is meant to determine if a text contains an opinion or not;
- The sentiment classification determines if a subjective text contains a positive or a negative feeling.

Applying the example of the mobile phone, it is possible to classify the sentence in this way: the sentence (1) “I bought an iPhone a few days ago” is a non-subjective (or objective) text, whereas the sentence (2) “It was such a nice phone” can be considered as a subjective text.

Regarding sentence (2), it is also possible to make a classification of the sentiment: in this case the text conveys a positive feeling about the thing which is being discussed. A further distinction is made between the classification of text or document and the classification of sentences.

In this dissertation, we are going to classify small pieces of text from Twitter. As a side note, it will be important to acknowledge the characteristics of the comments left on this platform in order to perform a proper pre processing of the data collected, prior to the real classification.

2.3 Twitter

Twitter is a social network where users can share short textual comments (about personal facts, events, or any kind of topic) with the chance to add multimedia content related to the comment, like images, videos, links to external pages, etc. The comments on these social networks are called “tweets”, and can have a maximum length that used to be of 140 characters and was only recently raised to 280 characters.

A tweet has some characteristics, for example it can “mention” another user by pre-pending a “@” to the username. Also, more importantly for this work, tweets can be categorized directly by the user, thanks to hashtags. These markers are words preceded by ‘#’ (pound) and serve to identify the topic of the message, or to somehow identify what is being discussed. The useful feature of hashtags is that they also serve the purpose of regrouping connected tweets (for example, by selecting a hashtag, one can see all the tweets that contain it). This makes it even easier to define what the “trending topics” are at a certain moment, as it suffices to list the most used hashtags.

From an operational point of view, they also guarantee that some words will be there, and they can already hint at the feeling that is being conveyed through the message. As an example, let us consider football related tweets and the hashtags #ChelseaOneLove and #ChelseaGoHome. Both of them contain the word “Chelsea”, so we can safely assume their content is about the London-based football team, with the first hashtag carrying a positive message, and the second carrying a negative one.

However, classification of tweets is particularly hard because tweets can be very short and contain sarcasm, grammatical mistakes, emojis, slangs and acronyms. For example, the tweet “Oh! I saw dat amaaaaazing pair of shoes, but a woman took’em away from me :-(#screwup” contains a lengthened word (“amaaaaazing”), several misspelled words (“dat” instead of that, “’em” instead of them), a sad emoticons (“:-(”), a slang word (#screwup) and, from the point of view of the context, it contains both positive and negative feelings. These elements add complexity to the task of classifying the sentiment of a tweet (Clark et al., 2011).

2.4 Methodology techniques

This section provides an explanation about the most common techniques for sentiment analysis.

Sentiment analysis classification techniques used in the literature are divided into lexicon-based approaches and machine-learning-based approaches (Angulakshmi and ManickaChezian, 2014); several of these techniques are shown in Figure 1, and more detailed explanations regarding these techniques will be given in the following subsection.

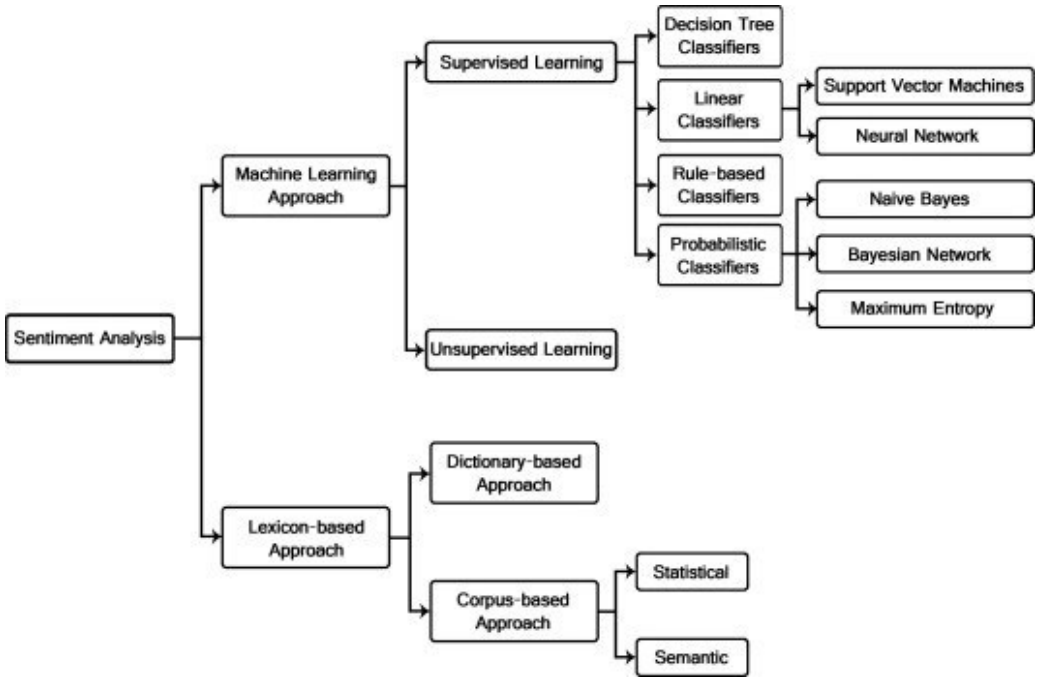

Figure 1 Sentiment classification techniques (Mrinali and Sharma, 2018)

2.4.1 Lexicon-based approach

The lexicon-based approach assumes that the contextual sentiment orientation (SO) of the sentence is the sum of the sentiment orientation of each word, phrase, or idiom (Palanisamy et al., 2013). The lexicon-based approach includes dictionary-based approach and corpus-based approach:

The dictionary-based approach. The dictionary-based approach is based on the construction of a dictionary that indicates a positive or negative polarity of a word, also called opinion word.

To start building this dictionary, a small set of opinion words is created manually. This set of seed words built manually represents the main keywords of the dictionary, then it can be integrated with synonyms and antonyms using databases of opinion words made for sentiment analysis. There are many of them on the internet, the most common

ones are Sentiwordnet (Esuli et al., 2006; Baccianella et al., 2010), Q-wordnet (Agerri et al., 2010), WordNet-Affect (Strapparava et al., 2004; Hu and Liu, 2004; Kim et al., 2004; Strapparava et al., 2006; Mohammad et al., 2009). Kamps et al. (2004) used seed words and linked them with synonyms from WordNet dictionary giving them a polarity. The originality of the authors is that they determined the polarity as the distance from seed words like "good" and "bad". For example, the adjective "nice" is close to "good", but it is not as strong as the seed word. The accuracy of this research was good, as it was about 67%.

Hu and Liu (2004) used the lexicon-based approach for sentiment analysis on product reviews. They tried to do a little more opinion mining: in practice, they extracted all the features of the product and tried to understand the feeling towards the product according to the features. The result of this research was very accurate.

The issue of dictionary-based approach is that it does not consider context. For example, the sentence "it is not that good" can be identified with a positive polarity instead of with a negative one.

The corpus-based approach. The corpus-based approach is a little more complex. The lexicon is gathered in a similar way as in the dictionary-based approach, and it can be done either manually (Tong, 2001; Taboada et al., 2011) or automatically. The automatic way to gather lexicon is to first build an initial lexicon of opinion words and then to expand it automatically, considering that the opinion words with the same previous polarity appear later in the text, unless there is a particular expression that changes the context (Kaji and Kitsuregawa, 2007). Corpus based approach has the characteristics of taking in consideration the context of the sentence. Corpus-based considers the different orientations of a word and its dependency on the context. If a positive opinion word is in a sentence, it doesn't mean that the whole sentence has a positive sentiment. Moreover, it considers the double meaning that some words may have, for example the adjective "long" based on the context "a long queue" has a negative sentiment, while based on the context of "the battery lasts long" has a positive sentiment. To achieve that, many techniques can be used. The most common one is to identify patterns or "bag of words", also called "unigrams", "bigrams" or "n-grams", depending on the number of words that contains. In this way, also combinations of words are considered, and to identify them, a large dataset is needed. Furthermore, it is also known the use of the intra-sentence sentiment consistency that identifies an opinion word through the use of conventions on connectives. For example, the word on conjunction "and" can indicate that the previous and next adjectives are of the same

polarity. The same would be for “either-and” and “neither-nor” (Hatzivassiloglou and McKeown, 1997). Kanayama and Nasukawa (2006) used the same logic but among sentences. The lexicon is then pre-processed to be reduced to a form that makes it easy to extract sentiment. The most common methods are POS (part-of-speech) tagging which is a tag that identifies the role of the word in the sentence (for example a noun, a verb, an adverb and so on). Stemming is a pre-processing technique that uses stem words as lexicon. For example, the words like believe, belief, believing use as stem word "belie", that is the root of all of them. In the case of twitter sentiment analysis, preprocessing techniques can be used such as shortening exaggerated words with repeated letters like “NOOO” (Kouloumpis et al., 2011), doing an emoticon detection to get the sentiment associated to it (Ritter et al., 2011 and 2012) and doing a hashtag detection to identify the topic or the keyword of the tweet (Palanisamy, 2013). There are some libraries that do the preprocessing automatically, like NLTK (Bird, 2006).

2.4.2 Machine learning approach

The second technique that can be used for SA is machine learning that includes unsupervised, supervised, and semi-supervised machine learning methods. First of all, we present a high-level overview of a classification problem in the machine learning domain, including the description of technical concepts such as training set, test set, and the way in which they are used.

2.4.2.1 A problem of classification

Sentiment analysis can be treated as a problem of classification, i.e., assigning a class to input samples, thus categorizing them as belonging to one of the possible classes defined for the problem at hand.

An object that performs such action is called “classifier”: it takes as input a sample x and outputs a predicted class y, based on some prior knowledge it acquired during a procedure called training.

Given a set of pairs (x, y), where x is a sample and y is a class (or label), training a classifier means trying to abstract the underlying mapping between the two, so that it is possible to assign the correct class to a fresh sample x'. This operation is often referred to as “learning”, since the classifier is acquiring knowledge about the structure of the dataset in question. In a more formal way, it means approximating a function $f(x, y)$ that

maps inputs to labels. The resulting function *f'* which is learnt is then going to be applied to fresh samples.

There are two kinds of learning: supervised and unsupervised (Pang et al., 2002). In supervised training, the classifier is shown a set of pairs containing an input and its correct labels. Based on these examples, it will try to learn the correct relationship between inputs and labels, so that it can later be applied to make a prediction on fresh samples, i.e., samples that are given in input without a known label. This procedure amounts to learning by examples and, in some way, mimics the way a child learns how to behave in the world, i.e., by receiving examples from the surrounding environment and abstracting them to some general rules that can later be applied to new situations.

On the other hand, in unsupervised learning, inputs are presented to the classifier without any label associated to them. The goal is then for the classifier to identify common and/or recurring elements and divide the input data based on the presence or absence of these identifying features. Unsupervised learning is particularly useful in the case of cluster analysis, which has exactly the goal of grouping pieces of data together, based on their similarities (or, at least, based on the fact that they are more similar among themselves when compared to other inputs).

For this work, a supervised learning method has been used, given the fact that sentiments constitute labels associated to the inputs, which we want to be able to predict. We will now describe the concept of training set, test set, and how the training procedure happens more in details.

A training dataset *Tr* is a set of pairs composed by input samples and "true" labels, meaning that these labels are to be considered correct. Sometimes, these labels are referred to as "teaching output", because they are the examples on which the training will be based. A test dataset *Te* is also a set of pairs composed by input samples and true labels (like the training set), but it will be used in a completely different way. In fact, the true labels will not be shown to the classifier, but will be kept aside as a reference against which the output of the classifier will be compared to assess its accuracy. In other words, the classifier will be asked to output predicted labels for the samples contained in the test set; these predicted labels will later be compared with the true labels to identify how many times the classifier predicts the correct label and how many times it makes a mistake.

The training procedure then tries to minimize the difference between the predicted output and the teaching output, thus ensuring the mapping between inputs and labels is well approximated, and the function *f'* is as similar as possible to the true function *f*.

Once the training is completed, it is important that the classifier is evaluated on a set of "fresh" samples it has never seen during the training. This allows one to verify whether the classifier is able to correctly categorize new samples based on what it learnt, rather than just "remembering" a particular association between an input sample and a label. In other words, this allows one to assess the abstraction capability (also referred to as the ability to generalize) of the classifier.

A well-known problem with many classifiers is that of *overfitting* (Shirani-Mehr, 2014) which happens when the classifier tends to learn the training set "by heart", thus achieving a perfect score on it, but then behaving rather poorly on fresh inputs. In turns, this usually happens when the training is protracted for too long time, with the result that the general picture of the problem is lost and the classifier focuses on the singular, individual samples rather than gaining abstract and general knowledge. In order to prevent this from happening, usually a small portion of the training set is set apart and used to stop the learning procedure before overfitting. This portion is called *validation set* and learning is arrested once the accuracy stops improving when measured on these samples.

Another very common problem when dealing with machine learning is that of storage capacity. In order to accurately solve complex problems, classifiers need to be trained on huge datasets, that contain a very large number of samples from which they can extract the necessary knowledge. This, in turns, poses other problems: the samples have to be stored somewhere (but this issue is usually not a blocking point, given how capacious and cheap storage devices have become), and then they have to be used as the basis for somewhat complex calculations. It means that they usually have to be stored in the internal memory of a computer (RAM) and processed by its CPU. Contrary to storage on hard drives, RAM capacity is often considerably more limited, and so is the CPU's computational power, with the result that it is sometimes impossible to process the large volume of available data. An immediate solution would be that of processing as much data as it fits in the available hardware and throw away the rest, with the result of discarding a potentially precious source of knowledge and not being able to take advantage of large datasets. To overcome this limitation, a technique called *batch processing* has been proposed (Banerjee and Basu, 2007), which essentially goes as follows: instead of processing the data all at once, it is split into several batches with limited size, which are then processed one after the other. This apparently obvious procedure, which makes so that the classifier goes through the training set little by little, is based on the assumption that each single batch is a good representative of the entire

training set, in order not to skew the learning towards one particular direction, but to help the classifier gain the general knowledge it needs.

2.4.2.2 Supervised training

In the supervised method, there are some input data that are labeled with their output, and the algorithms will learn to predict the output from the input data. A few examples of supervised algorithms used in sentiment analysis are Decision tree, Rule-based, linear classifier such as Support Vector Machine (SVM) and Neural Network and probability classifiers such as Naive Bayes (NB), Bayesian Network and Maximum Entropy (ME).

The two most important supervised machine learning techniques for sentiment analysis that will be treated in this dissertation are SVM and NB, and we now move to a more detailed description of these two methods. For more details about the other approaches, we refer the reader to specialized literature.

Support Vector Machines. The Support Vector Machine (SVM) or kernel functions, are a set of supervised learning methods for regression and classification of patterns, developed by Vladimir Vapnik and his team at Bell AT Laboratories in the 90s. The basic idea of SVM is to identify the optimal hyperplane that distinguishes several linearly separable patterns and to extend this method to non-linearly separable patterns through the transformation of the original data to map it to a new space (Steinwart, 2008). In other words, if in a certain space the data are not linearly separable, then a transformation that brings these data into a new space (with a larger dimension) in which they are separable from a plane (or hyperplane) will be done. Basically, it is the equivalent of a line in a space with more than two dimensions. In the Figure 2, a simple example of this concept is shown: the data belonging to the input space 2D are not linearly separable, and those belonging to the feature space 3D, obtained by applying a function phi, they become separable from a plane. The goal that SVMs achieve is to minimize the empirical classification error and maximize the margin between the separation plan and the points of the various classes. In fact, consider the example in Figure 3 (left): in general, it is possible to find several (or even infinite) acceptable separation plans. What SVMs do, unlike other methods, is to identify the optimal surface that maximizes the margin of separation between classes. This concept of “margin” is sometimes referred as “street”, since the hyperplane and its margins actually resemble the two halves of a road with the respective central line. In general, the decision function is completely specified by a small subset of the data, called support vectors: these are

the points that lie closest to the separation surface, and therefore are the most difficult to classify and have a direct impact on the position of the optimal surface. In other words, the support vectors are those points that if they are removed from the training set, they would cause a displacement of the separation surface, unlike the most distant points that have no effect on it. Once these critical points have been identified, it is a question of solving an optimization problem to find the surface that maximizes the distance, neglecting all the other points of the dataset and thus making the problem much simpler from the computational point of view. In Figure 3 (right), an example is shown related to the identification of the optimal separation surface; it is easy to notice that any other line that is able to discriminate the samples indicated with the squares from those indicated with the circles obtains a smaller width of the "street", therefore a lower margin of separation and hence generally a worse performance (Bennett, 2000).

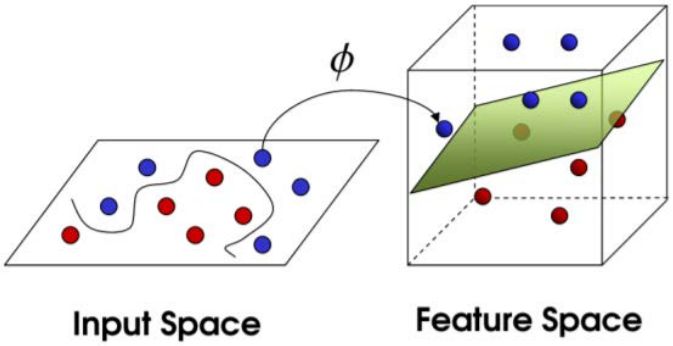

Figure 2 Example of SVM

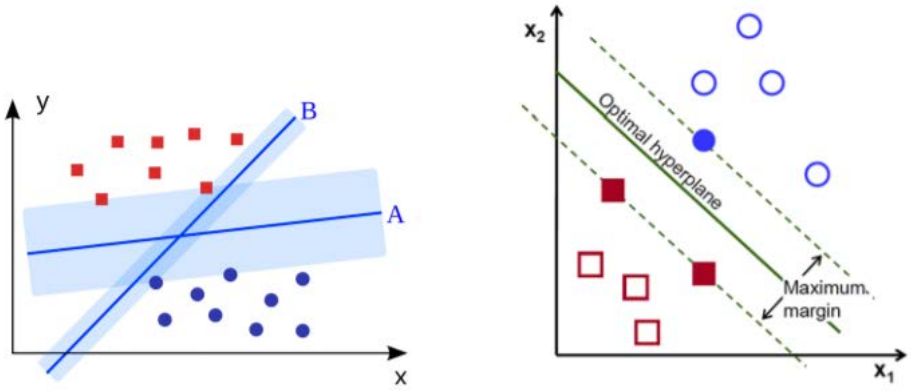

Figure 3 Possible ambiguity in identifying the separation surface (left), and the optimal separation surface (right)

The fact that the principle behind SVM is quite simple does not make implementing them easy; for this reason, we often use libraries already developed, optimized, and ready to be used.

The most popular SVM library is probably libSVM (Steinwart, 2008), developed by National Taiwan University. This tool allows to take advantage of different types of SVM, perform an efficient multi-class classification, use the cross-validation method for model selection, obtain probabilistic estimates, use different types of functions and so on.

Furthermore, it is essential to have a specific methodology available to perform the evaluation of the results obtained; otherwise, the risk is to overestimate the validity of the approach or even not realize that the approach works despite apparently poor percentages. Clearly, to better appreciate the differences between the several SVM models and all its options, it would be necessary to know the topic in depth, thus avoiding running into a typical but ineffective and inefficient approach like random walk.

In the field of twitter sentiment analysis, SVM classifier was used to classify tweets into positive and negative classes by Go (2009) and Gautam (2014). Both works use a similar approach: the input data is made up of many tweets that are encoded in features, which are the words with a negative or positive polarity of which the tweet is composed. Every feature has a value: the value 1 is given if the feature is present, the value 0 is given when the feature is absent. Afterwards, every tweet is represented as a vector (whose entries are 0 or 1), and is classified in a sentiment class that can be positive or negative.

Lastly, it is convenient to define the margin between the two classes, positive and negative, taking into account that the more distant it is from any tweet, the higher the confidence in the classification.

In another research (Chikersal, 2005), the SVM classifier is applied together with a rule-based classifier in order to improve accuracy in the final result.

Naive Bayes. A Bayesian classifier is a classifier based on the application of Bayes' theorem, that has been studied since 1960s. Naive Bayes is a popular supervised machine learning advanced method for text classification. Unlike Support Vector Machines, it performs a statistical classification based on the calculation of the probability: it allows one to calculate the probability of an event A, once it is known

that another event B has happened (Russell and Norvig, 2003). The formula of Naive Bayes is:

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$

The formula can be read in this way: given two events A and B, what is the probability that event A will occur, given that B has occurred? This is equal to the probability that B occurs since A has occurred, multiplied by the probability that A occurs, divided by the probability that B occurs.

Naive Bayes also has another characteristic: it considers the values of each feature independent from each other (hence the name "naive") (Rennie et al., 2003). For example, a banana is a fruit that has some features like the length of around 20 cm, the yellow color, and the roundish shape. Naive Bayes considers all these features independently, so to classify the fruit as a banana it does not take into account that the color, the length, and the shape may have some correlations among themselves.

In the case of sentiment analysis, the Naive Bayes classifier is one of the most common probabilistic classifiers used in text mining. In its simplest version, the classifier calculates the probability that an observation belongs to a class based on the distribution of words in the document: the order of words is ignored and therefore the position of the words in the document is not relevant. The Bayes theorem calculates the probability that a given set of features (words) is associated to a particular class (or label), based on the formula below:

$$P(C \mid x) = \frac{P(x \mid C)\, P(C)}{P(x)}$$

where x is the vector of all the features, C is the class (or label), $P(C \mid x)$ is the probability that the observation belongs to class C, given that it has features x, and $P(x \mid C)$ is the probability that the observation has feature x, given that it belongs to class C.

Now, the denominator can be discarded, since when an observation is given, the probability of it having a certain set of features x is a fixed constant. Then, we apply the

"naive" assumption which is at the basis of this classifier, i.e., that the features are mutually independent. In formulae, this means

$$P(x_1, x_2, \ldots, x_n) = P(x_1) \cdot P(x_2) \cdot \ldots \cdot P(x_n)$$

Then, we have

$$P(x \mid C) \cdot P(C) = P(x_1, x_2, \ldots, x_n \mid C) \cdot P(C)$$

and, by the naive assumption, we can finally write

$$P(C \mid x) \propto P(x_1 \mid C) \cdot P(x_2 \mid C) \cdot \ldots \cdot P(x_n \mid C) \cdot P(C)$$

An example of the application of this classifier to text analysis can be found in (Heimann and Danneman, 2014).

To summarize, the classifier learns to assign to every set of words (features) of the corpus a probability of positive or negative polarity, on the basis of the classifications already made, and it will apply the same rule to future documents that are not already classified.

According to Heimann and Danneman (2014), Naive Bayes is not very sustainable for text classification because it considers each word independently, while the coexistence of some words together can have different meanings and can reflect a different sentiment. On the other hand, ignoring the correlations between features allows you to do two things that would otherwise be problematic. Firstly, it includes a large number of features (that are the "unique" words). The unique words are often the majority in a document and can be predictive too. Secondly, it is efficient as shows results quickly regardless the size of the training set. In fact, Naive Bayes is able to identify the parameters to do the classification even with a small amount of training data.

There are more supervised machine learning techniques that give good results for SA, for example Maximum entropy (Hemalatha et al., 2014), random forest (Fang and Zhan, 2015) and convolutional neural networks (Severyn and Moschitti, 2015). For more details about the other approaches, we refer the reader to specialized literature.

Besides, in the literature there are also combinations of techniques belonging to different approaches such as the Naive Bayes lexicon-based approach used by Mali et al. (2016) and Naïve Bayes combined with Modified K means used by Bharti and Malhotra (2016).

2.4.2.3 Unsupervised and semi-supervised training

In the unsupervised method, the data are only in input and they are not labeled with an output, so in this case the algorithms learn to inherent structure from the input data. An example of unsupervised method is the search engine of the web, that searches for the most appropriate and relevant websites giving one or more keywords as an input.

Semi-supervised learning is a technique in between of unsupervised learning and supervised learning that uses both tagged and untagged training data: typically, a small amount of tagged data along with a large amount of untagged data. Researchers sustain that in the field of machine learning, untagged data, when used in conjunction with a small amount of tagged data, can significantly improve learning accuracy (Abney, 2008).

3 METHODOLOGY

The articles and books used for the technical background in the previous chapter, are retrieved based on the content. Most of the articles were found through the Google scholar engine as there are a large number of articles that had similar topics and were very close to the scope of this thesis. The other two main databases of scientific articles that were used to retrieve sources for this dissertation are arXiv.org and ResearchGate. Keywords like “sentiment analysis”, “twitter sentiment analysis”, sentiment analysis on product reviews”, “Naive Bayes”, “Support Vector Machines” and “Sentiment analysis using machine learning” were used for the search.

Considering that the boom of sentiment analysis was in the 2000s, many articles of that time are considered by the literature as fundamental and basis of sentiment analysis, and they are still cited in many of the current articles. Thus, many articles of the 2000s are used in this dissertation too. Recent articles are retrieved mainly for experiments made using different types of training sets, test sets, approaches, algorithms or a combination of different methods. For example, most of the articles retrieved for extracting sentiment from microblogging platforms like Twitter are of the years from 2008 onwards.

3.1 Research Set Up

Given the research purpose focused on the product reviews on Twitter, an experiment is applied to an actual case of a company that offers products and uses Twitter as main social media channel. A profile that matches well these criteria is the company Nike Inc. Nike is a successful American multinational, established in 1971, that produces sports shoes, clothing and accessories and offers a wide range of products and many launch products every year. Nike launches products every month and most of their advertisement and campaign announcements are online and through the social media channel of Twitter.

The company is the largest manufacturer of sports equipment in the world and the main product it offers is footwear. In fact, in this dissertation we are going to analyze a running shoe model launched in February 2018: Nike Epic React. These running shoes were created as a response to the Boost running shoe model launched by Adidas in 2015, the main competitor of Nike.

Indeed, Nike Epic React is one of the most important launches the company made last year, as it was studied for more than three years bringing new technologies and it was soon proclaimed as one of the lightest running shoes in the market.

In this work, tweets on Nike Epic React will be used in this research to build our test set of product reviews.

Now that we have addressed our research to a specific case of product reviews, a choice on which method to apply to do the twitter sentiment analysis has to be made. This choice will be based on the conclusions already made in the literature. The table below (Table 1) reports all the methods used in the scientific articles that conducted sentiment analysis on a similar domain to the one of this manuscript. In fact, not all the articles cited in the previous chapter are reported in the table, but only those that treated similar domains such as general tweets and/or reviews.

The table contains the source of the article, the dataset used and the domain in order to know what kind of data was analysed, the method used and its performance.

Table 1 Twitter sentiment analysis techniques used in the existing literature.

Source	**Dataset type and Domain**	**Method**	**Evaluation of the performance**
Hemalatha et al., 2014	Tweets on movie reviews	SVM, NE, ME	-
Jayvant et al., 2014	Products reviews	NB	-
Bharti and Malhotra, 2016	Tweets on mobile phone reviews	Modified K means with NB Classification algorithm	Accuracy: • NB only: 79.66% • KNN only: 83.59% • Modified K-Means hybrid: 89% and 91%
Sasikumar and Joshila Grace, 2017	Tweets on product reviews from flipkart.com	NB	NB accuracy split by feature: • 47% • 76%

			• 65% • 73%
Fang and Zhan, 2015	Product reviews collected from Amazon	NB, SVM, Random Forest. According to the results, the best method based on accuracy is Random Forest.	ROC curve area: • NB: 90% • SVM: 82% • Random Forest: 92%
Liu et al., 2013	Three datasets: one of tweets on Apple, Google, Microsoft, and Twitter, one of tweets that mentioned "Taco Bell" and one of tweets on the 2008 Presidential debate.	SVM	Accuracy split by sample ratio and feature goes from 44% to 81%.
Mali et al., 2016	Product reviews	NB lexicon-based	-
Pang et al., 2002	Two datasets of movie reviews	SVM, NB, ME using unigrams	Accuracy: • NB: 78.7% and 81% • SVM: 72.8% and 82.9% • ME: 80.4%
Mudinas et al., 2012	Two datasets: one on software reviews, and one on movie reviews	SVM trained on bag-of-words	Accuracy: 82.30%
Turney (2002)	Two datasets: one on product reviews and one on movie reviews	Lexicon-based using PMI-IR to calculate semantic orientation	Accuracy: • Product reviews: 74% • Movie reviews: 66%

According to the table above, the main method used is Naïve Bayes, followed by Support Vector Machines, Maximum Entropy and hybrid methods that derive from a combination of commonly-known techniques. Naïve Bayes, although it is a very simple

and fast classifier, gives also good results in terms of performance with an accuracy from 60% to 80% in average. Moreover, it is known to be a good classifier for unstructured text documents like the ones from Twitter (Mowafy, 2018). Support Vector Machines has also a high accuracy and most of the times it gives results very similar to Naïve Bayes. In this dissertation, we are going to classify the sentiment of tweets using these two classifiers.

Once the classifiers have been chosen, there are different steps to follow for conducting a twitter sentiment analysis on product reviews. The first step is to collect the data from Twitter to build a test set. The data must be preprocessed in a way to remove the data that is irrelevant and not important for the classification. In the context of Twitter, stop words such as "a", "an", "in", "the", hashtags, mentions and URLs will be removed from the text. The next step includes feature extraction that will encode every tweet with binary vector based on the presence of a world in the text. Afterwards, the test set is ready to be trained (by a training set divided in batches) with SVM and NB classifiers. The classifiers will give a sentiment polarity as output, and in the end, a few evaluation algorithms will be applied to assess their performance. Figure 4 represents the steps of sentiment analysis, that will be elaborated in section 3.3.

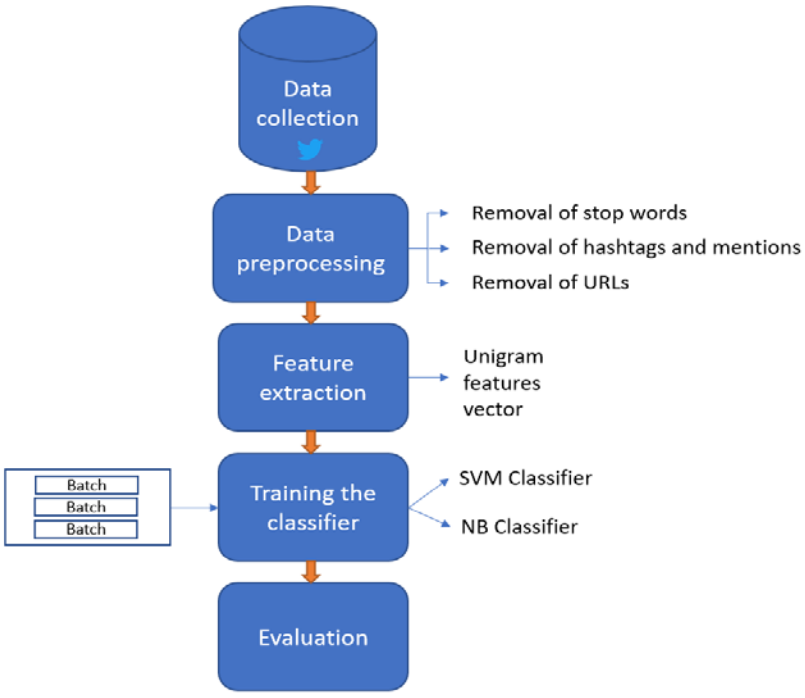

Figure 4 The steps of the process, from data collection to the evaluation of the trained classifier.

3.2 Libraries and code

Before describing each step of sentiment analysis in detail, an overview on some of the technical tools that were used for this work is provided.

3.2.1 Python

Among the many existing programming languages, one in particular have become increasingly associated to machine learning. Its simple and immediate syntax and, above all, the availability of many libraries and module have made Python the language of choice for a large number of data scientists, who are more focused on obtaining results rather than spending time on lower-level coding operations. Python offers an environment that requires essentially no setup effort, being ready to perform calculation from the very first line of a script. It is also particularly apt to perform mathematical calculations, thanks to libraries like numpy, that adds support for multi-dimensional arrays and a large variety of mathematical function. In this work, for example, many tasks will be carried out thanks to it, and we will also take advantage of its convenient data structures to store and process the data.

3.2.2 Sklearn

A very interesting module devoted to machine learning is sklearn, which stands for scikit-learn. It is a free software machine learning library that promises "simple and efficient tools for data mining and data analysis" and offers a wide array of tools for carrying out tasks like classification, regression, clustering, dimensionality reduction, data preprocessing, etc. It is largely written in Python and it is freely available for download, together with a thorough documentation and a large variety of examples, from the official website https://scikit-learn.org/stable/.

3.2.3 TextBlob

The problem of automated sentiment analysis being so interesting and far-reaching, some dedicated tools and libraries have already been developed. Among them, one of the most prominent ones is called TextBlob. On its website (https://textblob.readthedocs.io/en/dev/) it is advertised as "*[…] a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common*

natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more."
This library offers a large variety of features, including part-of-speech tagging, language translation (provided by Google Translate), tokenization of texts, word frequencies, n-grams, spelling corrections, etc. It also offers a direct interface for training and using classifiers like decision trees, maximum entropy classifiers, and naive Bayes. However, for this work we chose to re-implement the training part in order to have a more fine-grained control on the procedure and enable some optimizations. TextBlob will be used as a submodule for the project, when trying to categorize neutral samples.

3.2.4 Get Old Tweets

One of the most important things about machine learning is the availability of a good dataset, that correctly represents the problem at hand and gives the possibility to perform knowledge extraction from the data. As already mentioned, nowadays large quantities of data are available but, in order to be exploited, first of all they need to be collected. Many social networks or similar kinds of platforms offer APIs (Application Programming Interfaces) that developers can use in order to access (some of) the contents shared on the platform itself, with the aim of fostering the development of new plug-ins, new features, and also to allow for statistical analysis of the contents that the users share. However, sometimes these APIs can have heavy restrictions: for example, they might require registering officially and creating a project in order to let the platform know what one is doing, or they could have limitations regarding to the time span, thus limiting the number of contents that are accessible. In order to bypass these limitations, we are going to use a library freely available on GitHub called "Get Old Tweets": https://github.com/Jefferson-Henrique/GetOldTweets-python.

It is a library written in Python to recover old tweets from Twitter, bypassing the limitations imposed on the platform's API. In particular, one of its features allows a developer to provide a query, a time range, and a maximum number of tweets that should be recovered; the result is a list of all the tweets that match the specified criteria.

In the context of this project, this tool will be used to collect tweets related to Nike, that will constitute the basis for the test set on which our classifiers will be evaluated. In order to respect users' privacy, all the contents will be fully anonymized.

3.2.5 Datasets used

It is not always easy to find good datasets, with a dataset qualifying as good if it contains a large number of elements, that represent as many instances of the problem as possible, extensively cover a large portion of the input space, and are accurately labelled.

In the specific case of tweets, one of the datasets that are most often used by researchers in the field of sentiment analysis is that of http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip, which consists of a training set of 1.6 million labelled tweets, and a test set of roughly 500 labelled tweets.

There are many training datasets on the internet that can be used for text analysis, but we wanted to use a collection of authentic and generic tweets that are taken directly from the platform. The training dataset used in this thesis contains real tweets that have an emoticon in the text. The emoticon is used to label the sample, if it contains smileys like ":)", it is labeled as positive and if it contains something traditionally associated to a sad expression, like ":(", it is labeled as negative. This method allows one to retrieve a large number of labeled authentic tweets. On the other hand, a limitation of this approach is that it is not possible to automatically label neutral tweets, due to the absence of emojis.

The dataset comes into the form of a csv (comma separated value) file with the following format: sentiment, tweet_ID, date, query, user, text. Here, the only two fields that will be of interest for our work are "sentiment" and "text", since the other fields do not contain information about the emotion conveyed through the text. Moreover, it should be noted that this dataset comes into an already pre-processed form, stripped of all the emoticons and most of the hashtags.

3.3 Building the Nike test set: data collection, pre-processing, and formatting

The challenge of building a test set composed of tweets related to Nike products was addressed thanks to the GoT (Get Old Tweets) library presented above. A Python script was built in order to query the Twitter database for all the messages that contained "nike epic react" and that were posted on the platform between January 22, 2018 (when the product was officially released to the public) and August 31, 2018, with the aim of capturing the reactions to the introduction of this new product. A snippet of the Python script is presented in section 1 of Appendix I.

Although the library gives the possibility to collect a wide array of metadata, only the text of each tweet was deemed important, thus disregarding pieces of information like timestamp, username, etc.

A sample of the collected data can be found below:

- Check out 2018- Nike Epic React Flyknit Men's Running Shoes NEW WITH BOX #Nike #RunningShoes https://ebay.us/nL9Tw1 via @ eBay
- The "Black/Solar Red" Colorway Will Dawn The Nike Epic React Element 55 READ MORE: http://bit.ly/2BI5P1p
- Nike Epic React Shoes! Garmin GPS Watch! 1 Year Supply of SOS! Don't miss out! Enter Now! http://win.soshydration.com.au/ref/zA15632392 via @
- Dammit, @Nike, I appreciate the Epic React # BeTrue shoe, and I would get it if I had all the money in the world, but what I desperately need is a stability shoe (a-la the Zoom Structure) with the Epic foam!
- NIKE EPIC REACT FLYKNIT ID LIGHT OCEAN FOG-WHITE-GOLD MENS AJ7283-991 Sz 8.5 http://boann.flussstone.info/US/categories/twt/?item=153160047752
- MENS NIKE EPIC REACT ID FLYKNIT BLACK-RED-DARK GREY BV5968 994 SZ 9.5 http://aress.firefeuer.info/US/categories/twt/?item=362425329859
- The Epic React Flyknit Hyper Jade provides crazy comfort that lasts as long as you can run. Its Nike React foam cushioning is responsive yet lightweight, durable yet soft. @nikesportswear pic.twitter.com/Qd6IhW6Qas
- The new Rise React Flyknit silhouette debuts Saturday at Nike CA for $235 in "Sonic Yellow" featuring a sock-like laceless upper and the same soft and responsive React foam midsole found in the Epic React. Preview: https://bit.ly/2wj5Eo4 pic.twitter.com/JFG1DdoeJd
- Nike Epic React Flyknit Belgium AT0054-700 US 10-100% Authentic 1.888 pairs http://rover.ebay.com/rover/1/711-53200-19255-0/1?ff3=2&toolid=10039&campid=5338251676&item=123342620970&vectorid=229466&lgeo=1
- Nike Epic React Shoes! Garmin GPS Watch! 1 Year Supply of SOS! Don't miss out! Enter Now!

http://win.soshydration.com.au/ref/Mx15629427 via @

It is easy to see that this dataset is quite “raw”; for example, it contains hashtags, mentions (a ‘@’ followed by another user’s nickname), and links, which are not useful to classify sentiments.

The next phase was that of pre-processing this dataset in order for it to become more easily exploitable for the current work. Several operations were carried out: first of all, the so-called stop words were removed. In natural language processing, the term “stop words” usually denotes those words that do not influence the meaning of a sentence, but are present mostly for grammatical reasons. This set includes words like “the”, “a”, “an”, “in”, etc. By removing these words, that clearly appear very often across the dataset, we help the classifier focus on those words that are important for establishing the opinion expressed by the text.

Then punctuation was removed as well. This was done mainly for the following reason: when processing the text, we will split it into words according to white spaces, which means that we might find words like “good” and “good,” (notice the comma). When computing the histogram of each single word, i.e., the number of its appearances in the text, these two strings would be counted as different, because technically the second one contains one character that the first does not. However, it is clear that for the purposes of sentiment analysis, both have the same value and should be treated as the same thing. Thus, removing punctuation helps avoiding this kind of issues and focusing only on words.

Finally, links and URLs were removed too. It is very common for tweets to contain links, either to pictures (those beginning with pic.twitter.com) or to external webpages. In the case of links, the relevance for sentiment analysis is not easy to assess. Indeed, not all links have the same meaning. For instance, a link towards an auction website like eBay might suggest that the tweet is merely relaying an offer for a product to purchase, whereas a link to an informative page about consumers’ rights might suggest that there is a problem with a product and customers can request a refund. In other words, the links themselves could be analysed to try to obtain more information about the content of the surrounding text. However, this approach calls for a much lower-level approach than the one intended for this project, so the final choice was to remove links altogether, in order to let the training procedure, focus exclusively on words.

An example of the pre-processed dataset can be found below:

- `Check 2018- Nike Epic React Flyknit Mens Running Shoes NEW WITH BOX`
- `BlackSolar Red Colorway Will Dawn Nike Epic React Element 55 READ MORE`
- `Nike Epic React Shoes Garmin GPS Watch 1 Year Supply of SOS Dont miss out Enter Now via`
- `Dammit appreciate Epic React shoe would get had all the money world, what desperately need stability shoe a-la the Zoom Structure with Epic foam`
- `...`

Finally, the last part regarded the construction of the real test set. In fact, the content gathered so far is still unlabeled, which means that it cannot be as a benchmark for evaluating a classifier's accuracy.

Thus, the final part was the selection of some positive and some negative tweets, which were put in a csv file together with the corresponding label, either "0" (negative) or "1" (positive).

An extended test set was built that contained also neutral tweets (labelled with "2"), for the second set of experiments. In this case, neutral tweets are those that do not express any particular feeling, but are mostly informative and objective. For example, the first tweet of the list shown above is clearly neutral.

3.4 Training the classifiers

The training happens only on the training set containing positive and negative tweets, i.e., there are no neutral tweets whatsoever in the training set.

We now present the steps of the training algorithm, and then we move to a more thorough explanation:

1. The entire corpus composed of the training set is processed and a histogram of the words' frequency is built;
2. The N most frequent words (where N is a tunable parameter) are selected as the most representative;
3. The training set is divided into B batches, each of which contains S samples (where S is a tunable parameter and B is computed as a consequence of this choice);
4. The next batch is selected;

5. Each tweet in the batch is converted into a vector of features according to the presence or absence of the most common words identified at step (2);
6. The classifier is trained on this batch of samples;
7. Return to step (4) and repeat until all the batches have been shown to the classifier;
8. The training is completed.

We now move to the analysis of each step of the algorithm.

Step (1) is a processing that involves the entire training set (there is no notion of batches at this point). In a nutshell, for each unique word across the entire corpus, we compute the number of times it appears in the training set. This helps identifying the most common ones, which are then used as "features", i.e., elements thanks to which any sample can be compactly described. This corresponds to the following Python function:

```
def build_vocabulary(inputs):
    global vocabulary
    for tweet in inputs:
       for word in str(tweet).split():
             if word in vocabulary:
                   vocabulary[word] += 1
             else:
                   vocabulary[word] = 1
    return vocabulary
```

Through this function, a dictionary is built that maps each word to the number of times it appears in the corpus. It is then easy to sort this dictionary according to the number of appearances, thus ordering the words from the most common to the least common.

In step (2) we put a threshold on how many features we want to consider and keep only the N most frequent words, i.e., the first N entries when we sort the histogram computed at step (1). The purpose of this step is to identify the words on which the training should focus, keeping only the most common ones and discarding the rare or strange ones. It is clear that tuning this parameter has a considerable importance and influence on the final result. If the parameter N is too small, then not enough words will be selected and the set of features will not be able to represent samples in an expressive

way; for example there could be tweets where none of the most common words appear, thus carrying essentially no useful information. On the other hand, if we choose N too large (at most, the number of unique words in the corpus), then we will not operate any sort of selection, and everything will be a feature. This is not desirable as well, since it means the classifier will not be able to abstract and deduce the sentiment from the key words.

Step (3) helps in overcoming a problem described above, linked to the usage of large datasets. In fact, processing millions of tweets could be a problem if one is not equipped with a powerful machine. In particular, the memory consumption becomes an issue, and the training can quickly saturate the computer's RAM, thus making it impossible to proceed. In this scenario, the problem is that the entire dataset has to be kept in memory and processed all at once, which makes it impossible to deal with the large datasets that are instead very common in machine learning tasks. A possible solution proposed, e.g., by (Banerjee and Basu, 2007) is then to split the training set into smaller batches that can be easily handled and processed by the available machine. These batches will be shown to the classifier one by one, and it will learn progressively from each one of them. Of course, it is important that these batches satisfy some properties; for example, each one of them should be roughly a good representation of the variety which is contained in the training set. More in details, if the training set contains examples which are evenly associated to ten different labels, then each batch should contain samples from each class roughly in the same proportion. Also, the size of the batches should not be too small to prevent the classifier from focusing on only a few samples and losing the sort of general view on the training set which is instead desirable; on the other hand it should not be too big either, to avoid the memory-related issues that were described before. In general, the training is not too sensible to this value, so a good rule of thumb is choosing the largest size that can be easily processed by the machine that one is using for performing the operations.

This step corresponds to the following code snippet:

```
num_of_inputs = len(inputs_train)
num_of_batches = int(num_of_inputs / BATCH_SIZE)
for i in range(num_of_batches):
    b_inputs_train  = inputs_train[i*BATCH_SIZE:(i+1)*BATCH_SIZE]
    b_targets_train = targets_train[i*BATCH_SIZE:(i+1)*BATCH_SIZE]
    ...
```

Step (5) is called “feature extraction” or “encoding phase”. In this phase, each tweet is encoded in a way that can be easily understood by the classifier. This amounts to finding a way of representing the content of the message, so that it can communicate in a mathematically treatable way the information about the text. The encoding that we chose to use is a classical representation with binary vectors. Each tweet is represented by a vector whose length is N (the number of most frequent words selected at step (2)), and whose entries are 1 if the word is present in the text that is being analysed, or 0 otherwise. In order to clarify this concept, let us consider the following example: let the most common words selected at step (2) be [“shoes”, “car”, “awesome”, “awful”]. Then let us say that we have to encode the text “my new shoes are awesome”: its representation will be [1, 0, 1, 0], because the words “shoes” and “awesome” are present, whereas “car” and “awful” are not. On the other hand, if we see the vector [0, 1, 1, 0], then we do not know the precise content of the text, but we do know that it contained the words “car” and “awful”, so we might be able to make hypotheses on the overall statement contained in the message. Moreover, this format is particularly convenient and compact for being used during the training phase.

This feature extraction phase corresponds to the following part in the code:

```
def build_features(document):
    document_words = set(str(document).split())
    features = np.zeros(len(common_words_as_list))
    for j in range(len(common_words_as_list)):
       features[j] = (common_words_as_list[j] in document_words)
    return features

def extract_features(dataset):
    features = [(build_features(dataset[i])) for i in
range(len(dataset))]
    return np.array(features)
```

In the first function, the document (e.g., the tweet) is split into the words that compose it. Then a vector is initialized with N zeroes, i.e., as many zeroes as the number of most common words chosen at step (2). Finally, each entry in the vector is set to 1 if the corresponding word appears among the words of the current document. The second function simply calls the first one on each document that composes the dataset, thus extracting the features from all the documents.

Step (6) is where the real training happens. This step depends on the classifier that one wants to train, but it amounts to showing the batch to the classifier and trying to make it learn the function that maps samples to labels. Finally, this step is repeated for all batches, until the classifier has had the opportunity of seeing each sample (with its corresponding label).

Using sklearn, this is as simple as:

```
train_features = extract_features(b_inputs_train)
classifier.partial_fit(train_features, b_targets_train, classes=[0,1])
```

The first line calls the function to extract the features on one of the batches, whereas the second line performs a partial fit (i.e., an incremental training) on the classifier with the features extracted in the line above, together with the appropriate labels for that batch of samples.

As a side note, notice that steps (3) to (7) could be repeated multiple times, even with different orderings of the input data. For example, in the case of neural networks, it is normal to shuffle the order of the batches, and then show them to the network over and over again, to reinforce the learning.

From the point of view of the code, our Python scripts were inspired by those of Ramaneek Gill (available as a GitHub project at https://github.com/RamaneekGill/Twitter-Sentiment-Analysis). However, compared to the original code, some enhancements were made. Notably, the author recommends to use only a small portion (as low as 10%) of the training set, as it would not fit in the memory in its entirety, requiring more than 12 GB to be stored and processed. Instead, by implementing the batching technique described above, we were able to take full advantage of all the pairs (inputSample, label) contained in the training set. As for the parameters, after several experiments we chose N=2000, i.e., we consider the 2000 most common words in the corpus as features, and we set the size of each batch to be S=30000.

4 EXPERIMENTAL RESULTS

4.1 Evaluating the classifiers

In this part we analyse how the classifiers' accuracy was evaluated. Experimental results will be then presented in the next section.

The evaluation of the classifier was again carried out through automated Python scripts, that allowed to measure the classifier's accuracy against a correctly labelled test set. The evaluation will be divided into sub-parts, according to the classifier that was used (either Naive Bayes or SVM) and according to the test set (either composed of only positive and negative tweets, or composed of positive, negative, and neutral tweets).

It should be noted that the training set did not contain any neutral tweets, so one might legitimately wonder how it is possible to correctly classify samples whose class is unknown to the classifier. Our strategy here is twofold: first of all, we will make the classifier compute the prediction probabilities for the classes "positive" and "negative", i.e., the probabilities with which, according to the classifier, a certain sample from the test set belongs to one class or the other. Then, we will make the tentative assumption that for neutral tweets, these two probabilities will be rather close, thus showing some sort of indecision by the classifier. If this happens to be the case, i.e., the classifier is not able to give a clear answer, then we will run the text through an external library called TextBlob and use it to evaluate the level of subjectivity of the text. Said differently, TextBlob tries to associate a piece of text to a number that measures its level of subjectivity or objectivity. This number ranges from 0.0 (purely objective text) to 1.0 (purely subjective text). The way this is achieved is by analysing the text word by word and assigning a contribution and a weight to each word. For example, this is the part of the lexicon associated to the word "great" (from https://github.com/sloria/TextBlob):

<word form="great" cornetto_synset_id="n_a-525317" wordnet_id="a-01123879" pos="JJ" sense="very good" polarity="1.0" subjectivity="1.0" intensity="1.0" confidence="0.9" />

<word form="great" wordnet_id="a-01278818" pos="JJ" sense="of major significance or importance" polarity="1.0" subjectivity="1.0" intensity="1.0" confidence="0.9" />

<word form="great" wordnet_id="a-01386883" pos="JJ" sense="relatively large in size or number or extent" polarity="0.4" subjectivity="0.2" intensity="1.0" confidence="0.9" />

<word form="great" wordnet_id="a-01677433" pos="JJ" sense="remarkable or out of the ordinary in degree or magnitude or effect" polarity="0.8" subjectivity="0.8" intensity="1.0" confidence="0.9" />

As we can see, there are values regarding polarity, subjectivity, intensity, and confidence related to the several meanings that this word can have.

To summarize, after training a classifier (either Naive Bayes classifier or SVM classifier) on a training set composed only of positive and negative tweets, we will evaluate it on two different test sets: the first one will be composed only of positive and negative samples, whereas the second one will contain positive, negative, and neutral samples.

Another important thing to notice is the feature space: which features (i.e., which words) are going to be used as the principal ones for extracting the features from the test samples? This choice is of fundamental importance, as it will determine how test samples are described and represented to be processed by the classifier. To this end, the best thing to do seems to use the same set of words that were identified during the training as the most common ones. The reason is that this is the list of words that the classifier has already learnt to use for categorization, thus making it simpler to recognize patterns and structure if the same words are used again for extracting the test set's features.

This introduces another important assumption, which is that the set of words that are most important for identifying a sentiment are the same across several datasets. Said differently, it means that, once we have removed punctuation, stop words, etc., we can identify some words in a training set of positive and negative tweets that appear as the most common and the most valuable for classification, and subsequently use the same set of words as features for an unrelated test set. In turn, this implies that we do not have to process the test set to find the words that constitute the features: we can directly move to the encoding part, where a sample from the test set is represented as a binary vector according to the presence (or absence) of each single feature word.

For the test set composed only of positive and negative samples, the classifier is simply asked to output the prediction and, as side information, the probabilities it attributes to the two classes. Using the library sklearn, it is as simple as

```
predictions = classifier.predict(test_features)
pred_probabilities = classifier.predict_proba(test_features)
```

This allows us to assess the accuracy of the classifier on this simpler test set.

On the other hand, for the test set composed of positive, negative, and neutral samples, the previously described algorithm has been implemented:

1. The classifier is asked to output the prediction probabilities associated to the classes “positive” and “negative”: let the results be p_{pos} and p_{neg};
2. If $|p_{pos} - p_{neg}| > 0.25$, i.e., if the two probabilities are further apart than 0.25, than it means the classifier is confidently classifying the sample as either positive or negative, so we take the class predicted by the classifier and the algorithm ends;
3. If the condition in step (2) is not verified, i.e., if the probabilities differ by less than 0.25, than we run an analysis of the text with the library TextBlob and we try to determine its degree of subjectivity;
4. If the subjectivity level calculated by TextBlob is less than 0.4, it means the library classifies the text as mostly objective. This, together with the fact that the classifier output two rather close probabilities for positive and negative classes makes us classify the text as neutral, and the algorithm ends;
5. If the condition in step (4) is not verified, i.e., if TextBlob assigns a level of subjectivity equal to or larger than 0.4, then it means that the text is believed to be mostly subjective. For this reason, we take the largest probability output by the classifier and we categorise the sample as either positive or negative.

In Figure 5, we show a schematic representation of the classification procedure in this case.

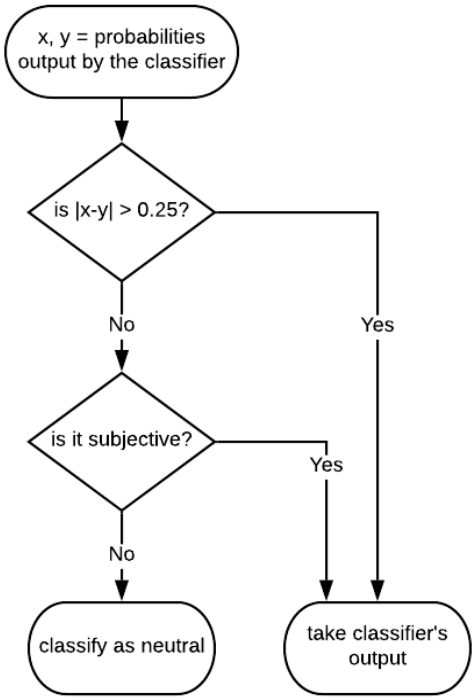

Figure 5 Flow chart for classification in the case of a test set that comprises positive, negative, and neutral tweets.

For completeness, a pertinent part of the Python code regarding the classification procedure of the test set of positive, negative, and neutral tweets is shown in section 2 of Appendix I.

4.2 Experimental results

In this section we present the experimental results we obtained. As mentioned above in the section related to training, we choose the following parameters: N = 2000 and S = 30000, where the first is the number of most common words that will be used as features, and the second is the size of each batch that will be presented to the classifier.

The outcome of the test set with only positive and negative tweets using SVM and NB classifiers is respectively 60.78% and 62.74% for the positive class and the rest is in the negative class, whereas the output of the test set with positive, neutral and negative tweets using SVM and NB classifiers is respectively 50% and 51.32% for the positive class, 21.05% and 19.73% for the neutral class and 28.95% for the negative

class. The results are also represented in a column chart in Figure 6 and Figure 7, that highlight the similarity of results given by the two methods.

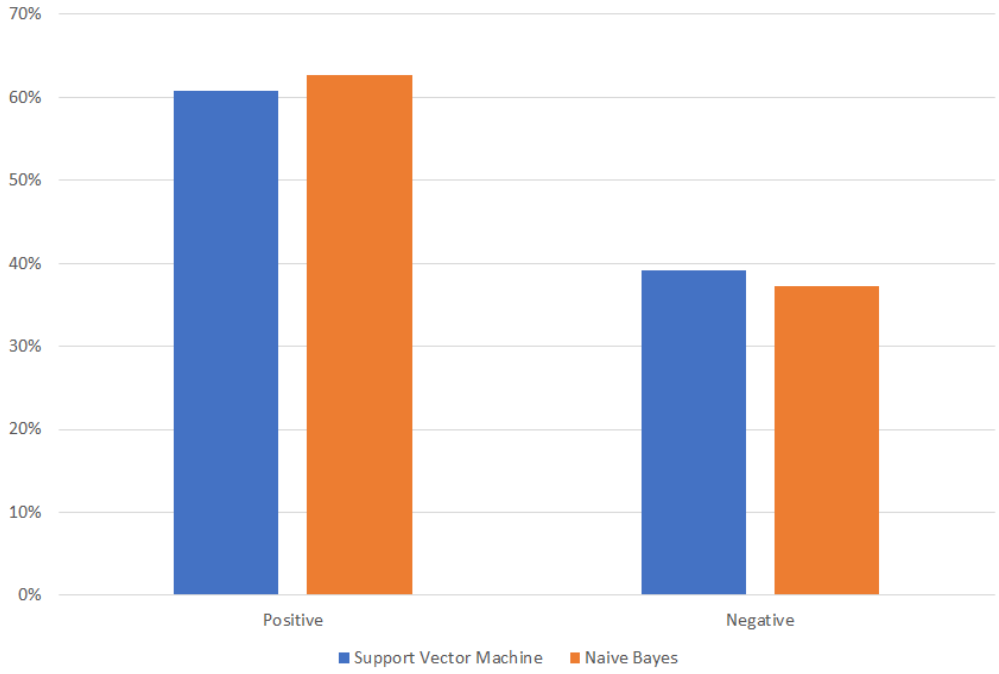

Figure 6 A comparison between SVM and NB in the case of a test set composed of positive and negative samples.

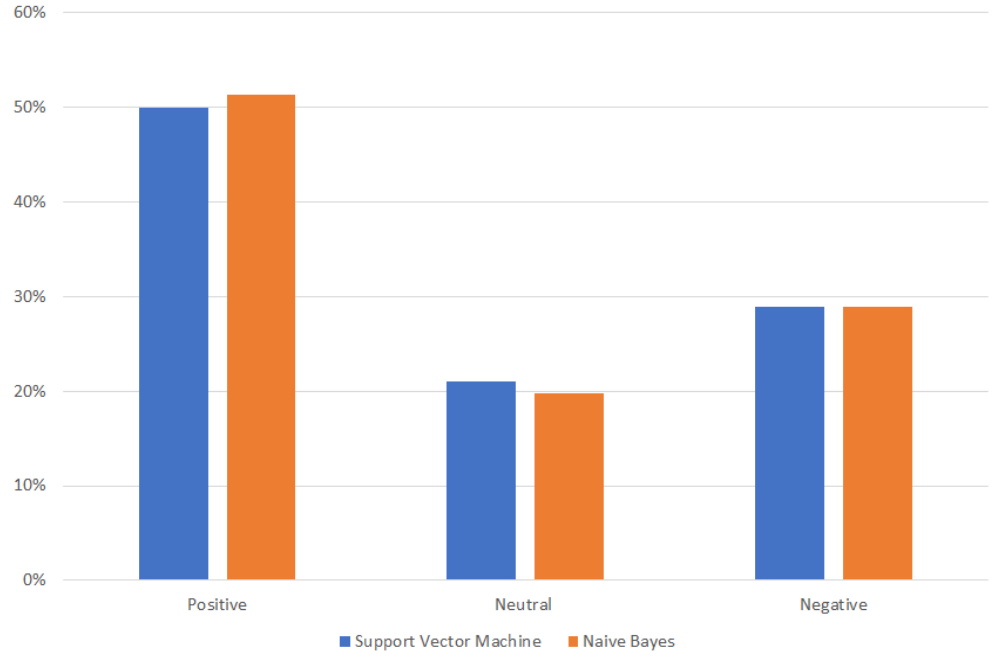

Figure 7 A comparison between SVM and NB in the case of a test composed of positive, negative, and neutral samples.

We remind the reader that the classifiers' accuracy will be evaluated on two different test sets (built manually) containing tweets about a specific product by shoemaking company Nike, called "Nike Epic React". The first test set will contain tweets that have either a positive or a negative sentiment. The second test set will contain all the tweets of the first one, plus tweets that have a neutral sentiment.

Naive Bayes. For the Naive Bayes classifier, we reported an accuracy of 76.47% on the test set with only positive and negative tweets, and 63.16% on the test set with positive, negative, and neutral tweets.
Below, we show an example of the output of the classifier in both cases:

- Only positive and negative tweets:

```
Accuracy against test set is 76.47058823529412 percent
...
Correct: 1 - 1 (0.1726231162864978, 0.8273768837135003)
Correct: 1 - 1 (0.4916039858819611, 0.5083960141180358)
Correct: 1 - 1 (0.22864702072277704, 0.7713529792772184)
Correct: 0 - 0 (0.8831954456497473, 0.1168045543502511)
Wrong: 1 - 0 (0.7770775637971958, 0.22292243620281105)
Wrong: 0 - 1 (0.28294036638888137, 0.7170596336111185)
Wrong: 1 - 0 (0.6404561235926849, 0.35954387640731456)
Correct: 1 - 1 (0.18765803817973642, 0.8123419618202624)
Wrong: 0 - 1 (0.35270883773265144, 0.6472911622673501)
Correct: 0 - 0 (0.649097310017074, 0.3509026899829258)
Correct: 0 - 0 (0.514861341687171, 0.48513865831282954)
Correct: 1 - 1 (0.2211464843283664, 0.7788535156716327)
Correct: 1 - 1 (0.49657635998609284, 0.503423640013904)
Correct: 1 - 1 (0.2022993818323552, 0.7977006181676335)
Correct: 0 - 0 (0.8459795303106634, 0.15402046968933475)
Correct: 1 - 1 (0.24619213454072403, 0.7538078654592767)
Correct: 1 - 1 (0.14489078279844522, 0.8551092172015514)
Wrong: 1 - 0 (0.686088103733277, 0.31391189626672394)
Correct: 1 - 1 (0.04236112953639608, 0.9576388704635996)
Correct: 0 - 0 (0.9618776738718978, 0.03812232612810733)
Correct: 1 - 1 (0.4138450031782385, 0.5861549968217629)
Correct: 1 - 1 (0.49392709121020417, 0.5060729087897959)
Correct: 1 - 1 (0.15821272023399915, 0.8417872797660005)
Correct: 0 - 0 (0.9821468136335878, 0.017853186366410662)
...
```

In the results shown above, the first column tells whether the classifier categorized the sample correctly or not; the second and the third columns represent the true label of the sample and the predicted label, respectively; the number in parentheses represent the prediction probabilities that the classifier assigns to class 0 (negative) and class 1 (positive), respectively. It is already possible to see that, for some samples, the classifier outputs a prediction with large confidence (i.e., the two probabilities are rather far apart), whereas for some other samples, the two probabilities are very close, so one class prevails on the other by just a small margin.

- Positive, negative, and neutral tweets:

```
Accuracy against test set is 63.1578947368421 percent
...
Correct: 0 - 0 (0.8008079462675901, 0.19919205373241075)
Correct: 1 - 1 (0.13149438384249995, 0.8685056161574989)
Correct: 0 - 0 (0.5792777464270106, 0.4207222535729872) subj
Correct: 0 - 0 (0.8438167697253467, 0.15618323027465877)
Correct: 0 - 0 (0.8165307578634473, 0.18346924213655894)
Correct: 1 - 1 (0.22107390321932788, 0.7789260967806653)
Correct: 1 - 1 (0.14440733721151694, 0.8555926627884803)
Correct: 1 - 1 (0.012562037950594006, 0.9874379620494074)
Correct: 1 - 1 (0.012476701870736004, 0.9875232981292648)
Correct: 1 - 1 (0.1331291557685615, 0.8668708442314377)
Wrong: 0 - 1 (0.3618597049249015, 0.6381402950751012)
Correct: 0 - 0 (0.6288652711724527, 0.37113472882754367)
Wrong: 0 - 2 (0.5309274817474521, 0.4690725182525438) obj
Wrong: 0 - 1 (0.06594226014574618, 0.9340577398542553)
Correct: 0 - 0 (0.6962993756066709, 0.3037006243933292)
Wrong: 0 - 1 (0.1570285487360472, 0.8429714512639586)
Wrong: 0 - 1 (0.42892150253013694, 0.5710784974698697) subj
Wrong: 2 - 0 (0.718035720856852, 0.28196427914314576)
Wrong: 2 - 1 (0.4997277777777775, 0.5002722222222225) subj
Correct: 2 - 2 (0.4997277777777775, 0.5002722222222225) obj
Wrong: 2 - 0 (0.6098897211006802, 0.3901102788993202) subj
Wrong: 2 - 0 (0.5376464160535803, 0.46235358394641973) subj
Wrong: 2 - 0 (0.6828109528931596, 0.31718904710683954)
Correct: 2 - 2 (0.4997277777777775, 0.5002722222222225) obj
Correct: 2 - 2 (0.4997277777777775, 0.5002722222222225) obj
Correct: 2 - 2 (0.4997277777777775, 0.5002722222222225) obj
...
```

The results shown above, formatted in the same way as the previous ones, demonstrate how we can correctly classify some tweets as neutral. In the cases where the probabilities output by the classifier are sufficiently close, we also request the subjectivity analysis by TextBlob library: if the result is “objective”, then we classify the sample as neutral, whereas if the result is “subjective”, we still take the class associated to the largest probability, however tiny the margin might be.

Support Vector Machines (SVM). For the SVM classifier, we reported an accuracy of 78.43% on the test set with only positive and negative tweets, and 65.79% on the test set with positive, negative, and neutral tweets. The formatting out the output is exactly the same as in the case of Naive Bayes.

- Only positive and negative tweets:

```
Accuracy against test set is 78.43137254901961 percent
...
Wrong: 1 - 0 (0.5879320405695738, 0.41206795943042623)
Correct: 1 - 1 (0.41312324098316855, 0.5868767590168315)
Correct: 1 - 1 (0.306027689210456, 0.693972310789544)
Correct: 0 - 0 (0.7123842402156986, 0.2876157597843015)
Wrong: 0 - 1 (0.4342347589381734, 0.5657652410618266)
Correct: 1 - 1 (0.3124639976128917, 0.6875360023871083)
Correct: 1 - 1 (0.23025535138699826, 0.7697446486130017)
Correct: 1 - 1 (0.49897376981285924, 0.5010262301871408)
Correct: 1 - 1 (0.3300665003551342, 0.6699334996448658)
Correct: 0 - 0 (0.8500941186180703, 0.14990588138192976)
Wrong: 1 - 0 (0.72600944834571, 0.2739905516542899)
Correct: 0 - 0 (0.538501461432753, 0.46149853856724704)
Wrong: 1 - 0 (0.5506974112632934, 0.44930258873670664)
Correct: 1 - 1 (0.14755853642665007, 0.8524414635733499)
Wrong: 0 - 1 (0.3005560191040163, 0.6994439808959837)
Correct: 0 - 0 (0.5635737118609421, 0.43642628813905787)
Correct: 0 - 0 (0.6462629872496981, 0.35373701275030184)
Correct: 1 - 1 (0.24634660922086327, 0.7536533907791367)
...
```

- Positive, negative, and neutral tweets

```
Accuracy against test set is 65.78947368421052 percent
```

```
   ...
   Correct: 1 - 1 (0.27012180049658585, 0.7298781995034141)
   Correct: 1 - 1 (0.3421666684637067, 0.6578333315362933)
   Correct: 1 - 1 (0.03127788665120734, 0.9687221133487927)
   Correct: 1 - 1 (0.05680625187722654, 0.9431937481227735)
   Correct: 1 - 1 (0.252910091027834, 0.747089908972166)
   Wrong: 0 - 2 (0.47113313820007596, 0.528866861799924) obj
   Correct: 0 - 0 (0.6448820900175869, 0.35511790998241305)
   Wrong: 0 - 2 (0.4618333152488374, 0.5381666847511626) obj
   Wrong: 0 - 1 (0.0792492531432617, 0.9207507468567383)
   Correct: 0 - 0 (0.7442477574794191, 0.2557522425205809)
   Wrong: 0 - 1 (0.2808106732751363, 0.7191893267248637)
   Correct: 0 - 0 (0.5287950531510869, 0.4712049468489131) subj
   Wrong: 2 - 0 (0.77218578465821, 0.22781421534179)
   Wrong: 2 - 1 (0.4133438211113868, 0.5866561788886132) subj
   Correct: 2 - 2 (0.4133438211113868, 0.5866561788886132) obj
   Wrong: 2 - 1 (0.4988791004047822, 0.5011208995952178) subj
   Wrong: 2 - 1 (0.41861542498189064, 0.5813845750181094) subj
   Wrong: 2 - 0 (0.5194935392972031, 0.48050646070279696) subj
   Correct: 2 - 2 (0.4133438211113868, 0.5866561788886132) obj
   Correct: 2 - 2 (0.4133438211113868, 0.5866561788886132) obj
   Correct: 2 - 2 (0.4133438211113868, 0.5866561788886132) obj
   Wrong: 2 - 1 (0.30593275930454533, 0.6940672406954547)
   Wrong: 2 - 1 (0.30593275930454533, 0.6940672406954547)
   Correct: 2 - 2 (0.4133438211113868, 0.5866561788886132) obj
...
```

Answers to research questions based on experimental results. The experiments presented above allow us to answer the three research questions that were presented in the introduction.

Regarding research question 1 and how to measure sentiment through Twitter text analysis, we showed that, despite limitation imposed by text length, grammar mistakes, abbreviations, etc., it is still possible to successfully classify the sentiment conveyed in tweets by using standard machine learning and lexicon analysis techniques.

Regarding research question 2 and how classification accuracy changes when introducing neutral tweets, we showed that it worsens considerably. However, this significant change in the test set can be partially countered with other techniques and a deeper analysis that involves, e.g., the text's subjectivity.

Regarding research question 3 and the classification accuracy when the training is carried out on a generic dataset (as opposed to a dataset made of product reviews), we show that it achieves acceptable levels, that seem in line with previous research in the

field. This seems to suggest that the text elements that define and identify sentiment are constant across different use cases, and that generic texts can be used to extract knowledge which can be successfully exploited for a specific problem. In other words, the results displayed above demonstrate that we obtain good results even when classifying tweets that do not belong to the original corpus, i.e., on the basis of features identified by analysing the most frequent words in another corpus.

Precision-Recall and Receiver Operating Characteristic. We now present an easy and intuitive way to compare the results obtained with the two classifiers: these comparisons will be based on the concepts of precision-recall curve and ROC curve.

When dealing with a problem of binary classification (i.e., a problem that asks to classify each sample as belonging to one of only two possible classes), there are several metrics that help assess the performance of a classification algorithm. They are briefly recalled below, assuming that one of the two classes is "positive" (P) and the other is "negative" (N):

- True positives (tp): samples classified as P and whose correct label is P;
- True negatives (tn): samples classified as N and whose correct label is N;
- False positives (fp): samples classified as P and whose correct label is N;
- False negatives (fn): samples classified as N and whose correct label is P.

Then the *precision* and the *recall* of a classifier are defined as

$$Prec = \frac{tp}{tp + fp} \qquad\qquad Recall = \frac{tp}{tp + fn}$$

In words, the precision identifies the ratio between the number of samples that are correctly classified as P and the number of samples that are classified as P (both correctly and wrongly), whereas the recall is the ratio between the number of samples that are correctly classified as P and the number of samples that *should* be classified as P.

It is easy to see that, by just having a classifier output always P, one can immediately achieve 100% of recall, but would likely obtain a poor precision. This is an example of why these two metrics are usually paired in a curve where precision is plotted against recall. In figure 6 we show the precision-recall curve for the two classifiers. Note that these curves are referred to the problem of binary classification (i.e., only positive and negative tweets).

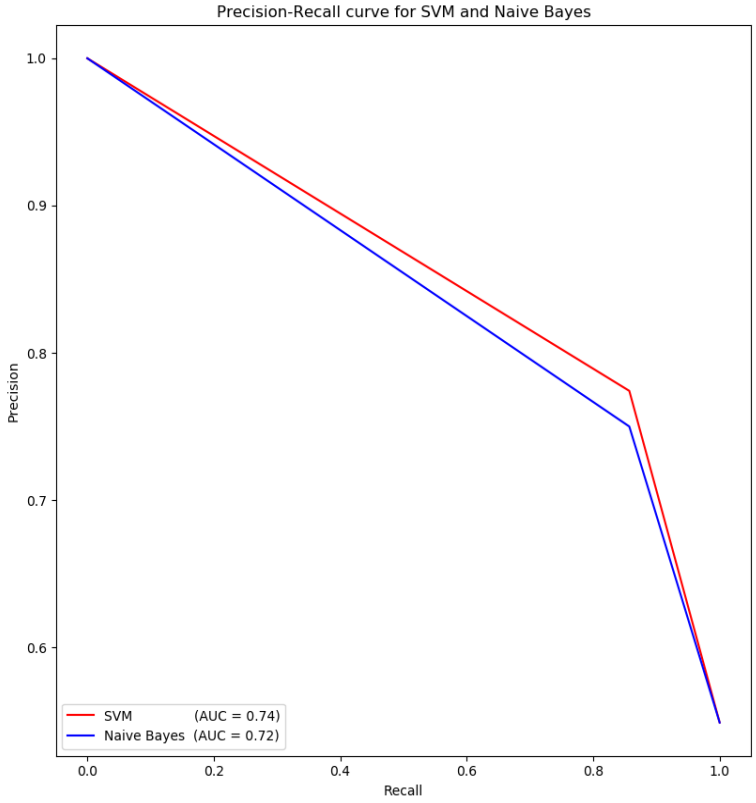

Figure 8 Precision-Recall curve for SVM and Naive Bayes classifiers.

Another way of assessing the reliability of a classifier is to plot its ROC (Receiver Operating Characteristic) curve. The pieces of information we need to build this graph are:

- True positive rate (tpr): ratio between the samples correctly identified as P and those that should be identified as P
- False positive rate (fpr): ratio between the samples wrongly identified as P and those that should be identified as N. This is also called probability of false alarm.

In formulas,

$$tpr = \frac{tp}{tp + fn} \qquad fpr = \frac{fp}{tn + fp}$$

Notice that the true positive rate is exactly the same as the recall, and sometimes these quantities are referred to as the probability of detection.

The ROC curve then plots the true positive rate (tpr) against the false positive rate (fpr) and gives an idea of how well the classifier performs as the discrimination

threshold changes. In figure 7 we show the two ROC curves for the classifiers that were used during the experiments presented above. Once again, we remind the reader that these curves are referred solely to the case of binary classification (i.e., when the test set contains only positive and negative tweets).

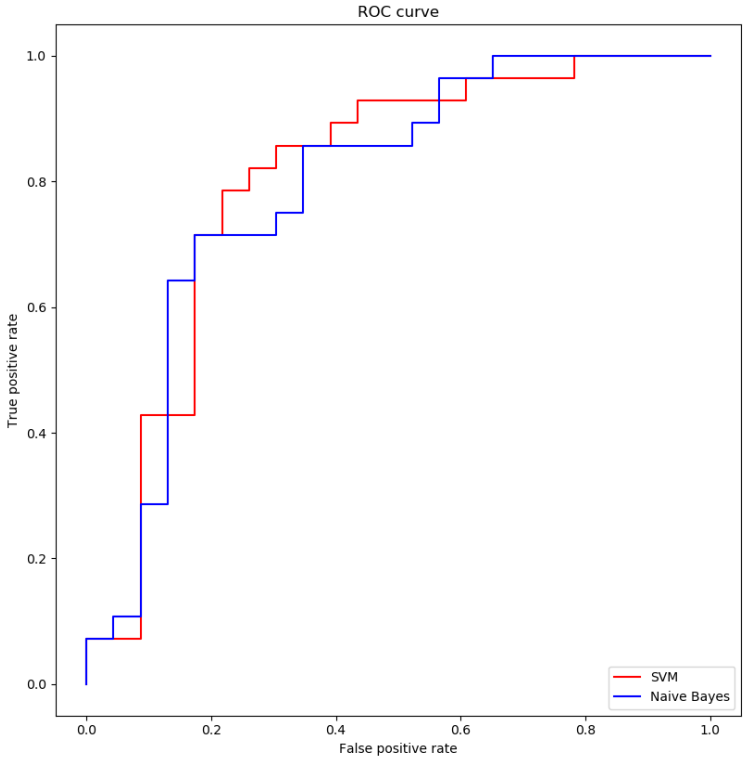

Figure 9 Receiver operating characteristic (ROC) curve for SVM and Naive Bayes classifiers.

Regarding the choice of the classifier, we notice that we obtain a slightly better accuracy with SVM than with Naive Bayes. However, drawing a comparison between them is a difficult task and is outside of the scope of this work. In fact, their performance depends on many factors, parameter optimization being one of the most important: applying a model to a specific problem, often requires a lot of effort for identifying the best set of parameters and meta-parameters that maximize the quality of the result.

Moreover, not all classifying algorithms are equally suited to any problem, depending on the intrinsic characteristics of each model. For example, Naive Bayes makes the fundamental assumption that the features are independent, which is generally not the case (hence the name “naive”), and which might be not particularly well suited to a problem where features are strongly correlated.

In general, there seems to be consensus on the fact that SVM performs generally better because of its ability of handling non-linearities, whereas Naive Bayes performs best when dealing with mutually independent features.

5 CONCLUSION

In this section, we draw some conclusions, highlight several limitations of the approach proposed in this work, and propose some possible directions for future research.

The overall result of this research was satisfactory, as it allowed us to answer the research questions set in the introduction. First of all, this work confirmed that there are different methods to measure the twitter sentiment analysis of a new product in the market (research question 1). In particular, NLP techniques are efficient when applied to the problem of sentiment analysis for short messages, thus validating that messages like tweets, with their shortness, their usual lack of grammatical rigor, distorted words, etc., can still be successfully processed and analysed through these techniques.

This work also proved that the knowledge that is extracted from a set of generic messages can then be exploited to categorize messages regarding a very specific topic, like reviews of a newly launched product (research question 3). In turn, this confirms that the set of words which express feelings and sentiments is almost constant across several datasets and that, once trained, the same classifier can be almost seamlessly applied to solve different problems of categorization.

Finally, this work allowed us to verify that the highly nuanced nature of the concept of sentiment adds to the difficulty of the classification problem. Moreover, even if it was verified that the content of the datasets is of secondary importance (i.e., a classifier trained on a generic training set can perform well even on a more specific test set), this is probably not the case for the number of classes represented in the datasets, which should be homogeneous. In other words, a classifier trained on a dataset that contains samples from two classes will not be able to achieve an equivalent accuracy when applied to a test set that contains samples from three classes, and this was verified by adding neutral messages to the test set. This suggests that some ability to distinguish between two classes does not automatically imply a matching ability to identify a sample as belonging to a third class. In some cases, with a less nuanced problem, it could be simpler to train the classifier to reason as “if it does not belong neither to the first class, nor to the second one, then classify the sample as belonging to the third class”, but this research suggests that this is apparently not the case for a complicated problem like sentiment analysis. This limitation remained true even when helping the classification procedure through another external library (TextBlob), capable of providing useful side information. Therefore, using our approach to include the neutral class in the test set without a training set that provides them, the answer to research

question 2 on whether neutral tweets influence in the accuracy of the experiment is affirmative. However, it may not necessarily be the case when using other approaches or training sets that comprise neutral tweets from the beginning. Anyhow, the accuracy achieved is significantly better than mere random guessing, thus suggesting some validity of the approach that was proposed. In fact, the accuracy resulting from both test sets is in the range of accuracy of the results from the literature (60%-80%, as per Table 1), and is considered a good index by the literature itself.

5.1 Limitations

The approach proposed in this work suffers from several limitations. First of all, the classifiers that were used are limited, in the sense that they need to make some limiting assumptions: Naive Bayes assumes all the features are mutually independent, whereas SVM tries to find a mapping to a space where the samples are linearly separable. Making these simplifying assumptions is a double-edged sword: on the one hand, they allow to make the training procedure easier, faster, and possible to run on normal machines; on the other hand, they limit the power and the accuracy of the classifier. Exploring other techniques, like using deep neural networks trained over a large number of passes on the training set, could be beneficial from the point of view of the accuracy, at the cost of making the process slower and more computationally demanding.

Another limitation to this approach is inherent to the nature of the training set. Given that it contains only positive and negative samples, a classifier trained on this dataset will be in particularly disadvantageous conditions when applied to a test set that comprises positive, negative, and neutral samples. Having at one's disposal a large training set that contains samples from all the three classes would certainly be beneficial, but it must be said that building these datasets is a long and painful operation that is often carried out manually.

Finally, a limitation regarding this work lies in the definition of features that was used. In this dissertation, a feature was defined as a single word, but there are other more powerful approaches. In fact, it could be possible to define features as combinations of multiple words (N-gram is the name of a combination of N words), to better capture the meaning that several words assume when they appear together. In a simple example, one could take as features all the words, and all the combinations of two words. This augments the representative power of a feature vector, at the cost of combinatorically increasing its size.

5.2 Future research

The goal of this dissertation was to answer the research questions, especially the first one on how to measure the sentiment on a product through twitter sentiment analysis. In the experimental results this question has been answered by comparing two machine learning methods, chosen among the most used one in the literature.
However, to answer the research question, other techniques may have been used. For example, there are several machine learning classifiers pretty accurate for sentiment analysis such as Maximum Entropy, Neural Networks, Random Forests, Rule-based classifiers, Decision Trees, a simple lexicon-based method or also a combination of them.

Moreover, the classifiers NB and SVM used unique words as features, without considering expressions, idioms and phrases in the bag of words, also called bi-grams or n-grams. N-grams or bi-grams could have been considered by combining two or more words as a unique feature. For example, in the sentence "I love it", the feature would have been the unique words "I", "love" and "you" and a combination of them "I love", "I you", "love you". To obtain that, a formula is applied:

$$N + \frac{(N - I)}{2}$$

And in the case of the example it is 3+(3*2)/2 with a result of 6 features in the sentence instead of three. In this way, it would have been more complex but more accurate.

Another future research that could be done is to use different types of training sets, for example using one with neutral label or one with more specific content about product reviews. The training set on product reviews could be downloaded from GitHub or any other website that provides labelled datasets for sentiment analysis, or it can be retrieved through Twitter using emoticons as sentiment label and specific keywords that identifies that the tweet is about a review of a product. For example, for the Nike set, a large training set can be retrieved from Twitter by using generic product-specific keywords like "nike", "shoes", "footwear", "sport".

In this thesis we discussed classifiers and techniques applied to a piece of text to recognize a feeling. However, sentiment analysis can evolve and is already evolving nowadays. Lately, there have been several experiments on the combination of techniques to increase the accuracy of the analysis, or other more complex techniques of unsupervised machine learning to get the feeling from raw data. Nonetheless, sentiment analysis is limited to be only able to classify a polarity information. Thus, in the future, research will tend more to study the opinion mining or opinion analysis, to

understand what are the motivations behind the positive or negative sentiment. This will be useful also in marketing analysis, so the company will be more aware of the motivation of the sentiment expressed by their customers on the social media.

Opinion mining, a term first introduced by Dave et al. (2003), is a technique that processes a search on keywords identifying, for each term, attributes (positive, neutral, negative) such that it becomes possible to extract the opinion associated with each key term. In general, almost all subsequent works do the classification of sentiments linked to single keywords (Pang et al., 2002; Pang and Lee, 2004).

A strong evolution of opinion mining started in 2010, when Hopkins and King laid the foundations of a new technique called integrated Sentiment Analysis (iSA). In the words of its authors, "*instead of performing an individual classification and then aggregate the predicted values, iSA directly estimates the aggregated distribution of opinions. [...] iSA is a language-agnostic algorithm (based on human coders' abilities). iSA exploits a dimensionality reduction approach which makes it scalable, fast, memory efficient, stable and statistically accurate*." (Ceron et al., 2016).

In the case of product reviews on Twitter, this method can be applied to see which features of the products are more appreciated (i.e., linked to a positive sentiment) or less appreciated (i.e., linked to a negative sentiment) by the customers. This can also be used in marketing analysis or in brand management to identify and classify the several motivations in the respective positive, neutral and negative classes.

REFERENCES

Abney, S. P. (2008) *Semisupervised Learning for Computational Linguistics*. Chapman and Hall/CRC.

Agerri, R. - Garcia-Serrano, A. (2010). Q-WordNet: Extracting Polarity from WordNet Senses. In: *Proceedings of the International Conference on Language Resources and Evaluation*, LREC, pp. 2300-2305.

Angulakshmi, G. - ManickaChezian, R, (2014) An analysis on opinion mining: techniques and tools. *International Journal of Advanced Research in Computer Communication Engineering*, vol. 3, no. 7, pp. 7483-7487.

Baccianella, S. - Esuli, A. - Sebastiani, F. (2010) SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. *Proceedings of the International Conference on Language Resources and Evaluation*, vol. 0, pp. 2200-2204.

Banerjee, A. - Basu, S. (2007) Topic Models over Text Streams: A Study of Batch and Online Unsupervised Learning. In: *Proceedings of the 2007 SIAM International Conference on Data Mining*, pp. 431-436

Bennett, K. P. - Campbell, C. (2000) Support Vector Machines: Hype or Hallelujah?. *SIGKDD Explorations*, vol. 2, issue 2, pp. 1–13.

Bharti, O. - Malhotra, M. (2016) Sentiment analysis on twitter data. *International Journal of Computer Science and Mobile Computing*, IJCSMC, vol. 5, issue 6, pp. 601 – 609.

Bird, Steven. (2006). NLTK: The natural language toolkit. In: *Proceedings of International Conference on Computational Linguistics,* 44th Annual Meeting of the Association for Computational Linguistics.

Ceron, A. - Curini, L. - Iacus, S. M. (2016). iSA: A fast, scalable and accurate algorithm for sentiment analysis of social media content. In: *Information Sciences*, pp. 105-124.

Chikersal, P. - Poria, S. - Cambria, E. (2015). SeNTU: sentiment analysis of tweets by combining a rule-based classifier with supervised learning. In: *Proceedings of the International Workshop on Semantic Evaluation,* SemEval*,* pp. 647-651.

Clark, E., Araki, K. (2011) *Text Normalization in Social Media: Progress, Problems and Applications for a Pre-processing System of Casual English*. Graduate

School of Information Science and Technology, Hokkaido University, Sapporo, Japan.

Dave, K. - Lawrence, S., Pennock, D.M. (2003) Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In: *Proceedings of WWW-03, 12th international conference on the World Wide Web*, ACM Press, pp 519–528.

Ellett, J. (2019) New Research Shows Growing Impact Of Online Research On In-Store Purchases. *Forbes*, *Forbes Media LLC* <https://www.forbes.com/sites/forbes-finds/2019/05/10/10-chic-and-timeless-floor-lamps-that-will-make-a-statement/#493652b85cff)>, retrieved 28.5.2019

Esuli, A. - Sebastiani, F. (2006) SentiWordNet: A publicly available lexical resource for opinion mining. In: *Proceedings of the 5th Conference on Language Resources and Evaluation*, LREC, vol. 6, pp. 417-422.

Fang, X. - Zhan, J. (2015) Sentiment analysis using product review data. *Journal of Big Data.*

Gautam, G. - Yadav, D. (2014). Sentiment analysis of twitter data using machine learning approaches and semantic analysis. In: *Proceedings of Seventh International Conference on Contemporary Computing,* IC3*,* pp. 437-442.

Go, A. - Bhayani, R. - Huang, L. (2009) Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford.*

Han, Jiawei - Pei, Jian - Kamber, Micheline (2012) *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers, Elsevier.

Hatzivassiloglou, V. - McKeown K. R. (1997) Predicting the semantic orientation of adjectives. In: *Proceedings of the 8th conference on European chapter of the Association for Computational Linguistics*, pp. 174–181, Association for Computational Linguistics.

Heimann, R. - Danneman, N. (2014) *Social Media Mining with R.* Packt Publishing, Birmingham.

Hemalatha, I. - Varma, G. P. S. - Govardhan A. (2014) Case Study on Online Reviews Sentiment Analysis Using Machine Learning Algorithms. *International Journal of Innovative Research in Computer and Communication Engineering*, Vol. 2, Issue 2.

Hopkins, D. J. - King, G. (2010) A Method of Automated Nonparametric Content Analysis for Social Science. *American Journal of Political Science 54*, pp. 229-247.

Hu, M. - Liu, B. (2004) Mining Opinion Features in Customer Reviews. In: *Proceedings of the 19th national conference on Artificial intelligence*, pp. 755-760.

IBM (2012) *Big Data at the Speed of Business*. <http://www-01.ibm.com/software/data/bigdata/>, retrieved 28.5.2019.

Jayvant, D. - Punde, S. - Sahane, S. - Shendage S. - Kanase, R. (2014) Product Review By Sentiment Analysis. *International Journal Of Engineering And Computer Science*, Volume 3 Issue 5 pg. 6202-6205

Jindal, N. - Liu, B. (2008) Opinion Spam and Analysis. In: *Proceedings of the International Conference on Web Search and Data Mining*, pp. 219-230.

Kaji, N. - Kitsuregawa, M. (2007) Building lexicon for sentiment analysis from massive collection of HTML documents. In: *Proceedings of the joint conference on empirical methods in natural language processing and computational natural language learning* pp. 1075–1083, Association for Computational Linguistics.

Kamps, J. - Marx, M. - Mokken, R. J. - de Rijke M. (2004) Using wordnet to measure semantic orientations of adjectives. In: *Proceedings of International Conference on Language Resources and Evaluation*, LREC.

Kanayama, H. - T. Nasukawa (2006) Fully automatic lexicon expansion for domain-oriented sentiment analysis. In: *Proceedings of Conference on Empirical Methods in Natural Language Processing,* EMNLP.

Kim, S. - Hovy, E. (2004) Determining the sentiment of opinions. In: *Proceedings of International Conference on Computational Linguistics,* COLING, pp. 1367–1373.

Kouloumpis, E. - Wilson, T. - Moore, J. (2011) Twitter sentiment analysis: The good the bad and the OMG!. In: *Proceedings of the Fifth International Conference on Weblogs and Social Media*, ICWSM, pp. 538–541.

Lee, H. - Han, J. - Suh, Y. (2014) Gift or threat? An examination of voice of the customer: The case of MyStarbucksIdea.com. *Electronic Commerce Research and Applications*, pp. 205–219

Liu, B. (2010) Sentiment Analysis and Subjectivity. *Handbook of Natural Language Processing,* Second Edition.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, Vol. 5, No. 1, pp. 1-167

Liu, B. - Zhang, L. (2013) A survey of opinion mining and sentiment analysis. *Mining Text Data*. pp. 415-463.

Liu, S. - Li, F. - Li, F. - Cheng, X. - Shen, H. (2013) Adaptive co-training SVM for sentiment classification on tweets. In: *Proceedings of the 22nd ACM international conference on Information & Knowledge Management,* CIKM, pp. 2079–2088.

Mali, D. - Abhyankar, M. - Bhavarthi, P. - Gaidhar, K. - Bangare, M. (2016) Sentiment analysis of product reviews for e-commerce recommendation. *International Journal of Management and Applied Science.*

Mashable (2011) *Twitter Has 100 Million Monthly Active Users; 50% Log In Every Day.* <http://mashable.com/2011/10/17/twitter-costolo-stats/>, retrieved 28.5.2019.

Mohammad, S. - Dunne, C. & Dorr, B. (2009) Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing,* EMNLP.

Moreno, A. - Redondo, T. (2016) Text Analytics: the convergence of Big Data and Artificial Intelligence. *International Journal of Interactive Multimedia and Artificial Intelligence,* Vol. 3, No. 6.

Mowafy, M., Rezk, A., El-bakry, H. M. (2018) An Efficient Classification Model for Unstructured Text Document. *American Journal of Computer Science and Information Technology,* Vol.6 No.1.

Mrinali, A. - Sharma, S. K. (2018). Sentimental Analysis: A Survey. *International Journal of Computer Sciences and Engineering*, 6(7), 939-951.

Mudinas, A. – Zhang, D. – Levene, M. (2012) Combining Lexicon and Learning based Approaches for Concept-Level Sentiment Analysis. In: *Proceedings of the First International Workshop on Issues of Sentiment Discovery and Opinion Mining*, p.1-8.

Palanisamy, P. - Yadav, V. - Elchuri, H. (2013) Serendio: Simple and Practical lexicon based approach to Sentiment Analysis. *Proceedings of Second Joint Conference on Lexical and Computational Semantics*, pp. 543-548.

Pang, B. - Lee, L. - Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing,* EMNLP, pp. 79–86.

Pang, B. - Lee, L. (2004) A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of the Association for Computational Linguistics (ACL), pages 271–278.

Pang, B. - Lee, L. (2008) Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval 2*.

Rennie, J. - Shih, L. - Teevan, J. - Karger, D. (2003) Tackling the poor assumptions of Naive Bayes classifiers. *Proceedings of the Twentieth International Conference on Machine Learning,* ICML.

Ritter, A. - Etzioni, O. - Etzioni, M. - Clark, S. (2011) Named entity recognition in tweets: an experimental study. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP, pp. 1524–1534.

Ritter, A. - Etzioni, O. - Etzioni, M. - Clark, S. (2012) Open domain event extraction from twitter. In: *Proceedings of the 18th international conference on Knowledge discovery and data mining*, KDD, pp. 1104–1112.

Russell, S. - Norvig, P. (2003) Artificial Intelligence: A Modern Approach. *Prentice Hall*, New Jersey

Sasikumar, A. N. - Joshila Grace, L. K. (2017) Sentiment analysis of social network sites for categorization of product reviews. *International Journal of Pure and Applied Mathematics*, vol. 117 No. 21

Severyn, A., - Moschitti, A. (2015). Twitter sentiment analysis with deep convolutional neural networks. In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 959-962.

Shirani-Mehr H. (2014) Applications of Deep Learning to Sentiment Analysis of Movie Reviews. *Technical report*, Stanford University.

Shirdastian, D. - Laroche, M. - Richard, M. (2017) Using big data analytics to study brand authenticity sentiments: The case of Starbucks on Twitter. *International Journal of Information Management*.

Steinwart, I. - Christmann, A. (2008) *Support Vector Machines*. Springer-Verlag.

Strapparava C. - Valitutti A. (2004) WordNet-Affect: an affective extension of WordNet. In: *Proceedings of 4th International Conference on Language Resources and Evaluation*, LREC, pp. 1083-1086.

Strapparava, C. - Valitutti, A. - Stock, O. (2006) The affective weight of lexicon. In: *Proceedings of International Conference on Language Resources and Evaluation*, LREC, pp.423-426.

Taboada, M. - Brooke, J. - Tofiloski, M - Voll, K. - Stede, M. (2011) Lexicon-based methods for sentiment analysis. *Computational Linguistics*, vol. 37, issue 2, pp. 267-307.

Tong, R. M. (2001) An operational system for detecting and tracking opinions in on-line discussions. In: *Proceedings of SIGIR Workshop on Operational Text Classification*.

Turney, P.D (2001) Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. In: *Proceedings of the Twelfth European Conference on Machine Learning*, Springer-Verlag, pp. 491-502.

We Are Social (2018) *Global Digital Report 2018*. <https://wearesocial.com/blog/2018/01/global-digital-report-2018>, retrieved 28.5.2019

APPENDIX I PYTHON CODE

1. Definition of search criteria for automated tweet extraction

```
tweetCriteria = got.manager.TweetCriteria().\
setQuerySearch('nike epic react').\
setSince("2018-01-22").setUntil("2018-08-31").setMaxTweets(50000)
allTweets = got.manager.TweetManager.getTweets(tweetCriteria)
for tweet in allTweets:
      print(tweet.text)
```

2. Classification procedure in case of a test set composed of positive, negative, and neutral tweets

```
for i in range(num_of_test_samples):
      if abs(pred_probabilities[i][0]-pred_probabilities[i][1])>0.25:
            # take the classifier's prediction
            predictions[i] =
classifier.predict(np.reshape(test_features[i], (1, NUM_FEATURES)))
      else:
            text = inputs_test[i]
            analysis = TextBlob(text)
            if analysis.subjectivity < 0.4 :
                  # mark as 'neutral'
                  predictions[i] = 2
            else:
                  # take the classifier's prediction
                  predictions[i] =
classifier.predict(np.reshape(test_features[i], (1, NUM_FEATURES)))

      if predictions[i] == targets_test[i]:
            correctly_predicted += 1
```