INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Department of Information Science and Technology

Multilabel classification of unstructured data using Crunchbase

Marco Filipe Madeira Felgueiras

A dissertation submitted in partial fulfillment of the requirements for the degree of
Master in Computer Science

Supervisor:
Doctor Fernando Batista, Associated Professor,
Iscte – Instituto Universitário de Lisboa

July, 2020

Resumo

Este trabalho compara diferentes métodos e modelos para classificação de texto utilizando informação proveniente do Crunchbase, uma grande base de dados que contém dados sobre mais de 600000 empresas. Cada empresa está associada a uma ou mais categorias, de 46 possiveis, e os modelos propostos utilizam apenas a descrição de cada empresa para prever a sua categoria. Foram aplicadas várias técnicas de processamento de linguagem natural para extração de informação incluindo *stemming*, lematização e *Part-of-Speech Tagging*. Este *dataset* é altamente desiquilibrado, a frequência de cada categoria vai desde 0.7% a 28%. A primeira experiência, é um problema multiclasse que tenta encontrar qual a categoria mais provável para uma empresa utilizando apenas um modelo para todas as categorias, obtendo um resultado global de 67% de *accuracy* utilizando *SVM*, *Naive Bayes* e *Fuzzy Fingerprints*. A segunda experiência utiliza vários classificadores, um por cada categoria, para atribuir todas as categorias de uma determinada empresa obtendo resultados de 73% de precisão e 47% de *recall*. Os modelos resultantes do nosso trabalho podem ser um ativo importante para a classificação automática de texto, não só para descrições de empresas mas também para outros textos, como páginas de Internet, blogs, notícias, entre outros.

Palavras chave

Classificação Multilabel, Mineração de Texto, Classificação de Texto, Aprendizagem Automática, Crunchbase, Processamento de Linguagem Natural

Abstract

Our work compares different methods and models for multilabel text classification using information collected from Crunchbase, a large database that holds information of more than 600000 companies. Each company is labeled with one more categories, from a subset of 46 possible, and the proposed models predict the categories based solely on the company textual description. A number of natural language processing strategies have been tested for feature extraction, including stemming, lemmatization, and Part-of-Speech Tagging. This is a highly unbalanced dataset, where the frequency of each category ranges from 0.7% to 28%. The first experiment, is a Multiclass classification problem that tries to find the most probable category using only one model for all categories, with an overall score of 67% using SVM, Naive Bayes and Fuzzy Fingerprints. The second experiment uses makes use of multiple classifiers, one for each category, and tries to predict the complete set of categories for each company, with an overal score of 73% precision and 47% recall. The resulting models may constitute an important asset for automatic classification of texts, not only consisting of company descriptions, but also other texts, such as web pages, text blogs, news pages, etc.

Keywords

Multilabel Classification, Text Mining, Text Classification, Machine Learning, Crunchbase, Natural Language Processing

Agradecimentos
Acknowledgements

First and foremost, I would like to show my appreciation and gratitude to Professor Fernando Batista, without his endless support and guidance this work would not be possible.

I would like to thank the Department of Information Science and Technology of ISCTE for providing me all the necessary tools for my journey to be successfull and enjoyable.

A word of appreciation for all my colleagues, either professional or academic, that helped me in one way or another during this long road, thanks for everything, I will bring you with me for my entire life.

A special thanks for my Family and Friends that walked alongside with me through my academic path and without them I am sure that I could not make it untill the end.

Last, but not least, I would like to dedicate all my work to my beloved mother Carla Felgueiras, I'm sad that she cannot see the outcome of what she fought for her entire life as it was her desire, but I'm sure that she was/ is with me the entire way, all of this is for you, hope to make you proud, love you.

Lisboa, 29 de Julho de 2020
Marco Felgueiras

Contents

List of Figures

List of Tables

1 Introduction

This chapter focus on the overall goals and motivation for our work. Initially we state a small contextualization of our work followed by the motivation and research questions. After framing our work, it is presented the background for the main techniques used throughout our development. The last section presents the structure for this document.

1.1 Context

We live in a digital society where data grows day by day, most of it being textual data. This creates the need of processing all this data and collect useful information from it. Text Classification plays a fundamental role in a variety of systems that process text data. One of the early implementations of Text Classification algorithms was in the e-mail spam detection software, where the main goal is to automatically assign one of the two predefined labels (spam and not spam) to each received message in an inbox. Other well-known Text Classification tasks, nowadays receiving increasingly importance, is sentiment analysis, which consists in attributing a sentiment to a given text content (happiness, anger, sadness, ...). Sentiment analysis can be used in several fields, for instance, extract opinions over a product by analyzing its comments and reviews, analyzing tweets in order to check for cyber bullying among users, detecting general opinion from social networks over a subject (politics, sports, trending world wide topics).

Crunchbase is the largest companies database in the world, containing a large variety of up-to-date information about each company. Originally it was the data storage from its mother company TechCrunch and it was founded by Michael Arrington in 2007. Until 2015, TechCrunch was the owner of the Crunchbase data. Afterwards, Crunchbase decoupled itself from the TechCrunch to focus on its own products. Crunchbase database contains up-to-date details about over 600000 companies, including a small description, a detailed description, number of employees, headquarters regions, contacts, market share, current areas of activity and it is stored into different categories.

Having all this information available, it is possible to combine it with the latest Text Classification methods and Machine Learning algorithms and produce a classification model that based on a company description can automatically assign a category to it. Since all the information from each company is labeled into multiple categories and each of them has a description, it is possible to assume that each category is described into a set of textual data. The outcome of our work can have inumerous applications, the main goal being to interpret text data from a wide range of sources, it can be applied to news or tweets, reddit threads, documents, etc.

1.2 Motivation

With the countless information sources available and the recent technology advances the amount of text data that systems produce in a daily basis is countless. Useful information can be extracted from raw data, data is factual and has no structure. Data can be very useful, but only when organized, the outcome of this organization process is information. Information is a very useful asset for data owners. For instance, in the retail area, the opinions and comments for the users play a major role in the product selection area. Another good example is social networks, social networks are a pure raw data source, but when collecting the information that comes hidden inside, it is possible to identify, for example, relation between trending topics, natural disasters that can be happening, block violent information, among others. Journal and news are also a big area that makes use of text data to produce high quality information from it.

The individual user, when it comes to his role, he mainly sees this type of algorithm influence him when it comes to news/ trends suggestions between the different applications that he uses (twitter, spotify, reddit, google news, etc). Here, us, as users, only want to receive the information as quick and as accurate as possible, so that we can be informed of what is happening in the world as fast as we can. This can have a big impact when it comes to our society. Twitter is one of the fastest information spreading social networks, thus, as an example, we have been seeing an increase in the police and firefighters usage of it to broadcast important information to the citizens from all different places around the globe.

At the business level, however, the information is the most valuable asset of each company nowadays. Information about its clients and end users can play a big role to the approach that each company makes to the market. Google has invested a lot of time and money into Machine Learning and Natural Language Processing researching areas because it is crucial for them to make use of the data that users provide to them to offer a better experience among the different applications that they have.

However, despite the fact that this area increases day by day, it is not perfect. Text Classification performs well when approaching binary problems, where there are only two options, however, when it comes to multiple selection of multiple labels based on a text input there is still room for improvement.

1.3 Research Questions

The proposed work is a complex task that is dependent on several factors. The quality of the dataset, pre-processing techniques and the applied algorithms all have a huge influence on the outcome of our work. With this in mind, several questions arise:

- Is it possible to classify companies based only on its textual description?

  By solving this question it is possible to determine if the developed model is able to attribute categories to a company based only on its description. For example, when processing the description for the Dropbox “Dropbox provides secure file sharing, collaboration, and storage solutions.” the outcome should be the “Private Cloud” and “File Sharing” categories. Thus, this raises questions regarding the specificity and focus of the problem.

- What is the best model to classify a company based on its textual description?

  The outcome for this research question is a tuned model and respective pre-processing techniques that can have the best performance for the proposed work when comparing it to the latest known studies.

- Can the developed model be applied to a different data source?

  The developed work makes use of the Crunchbase data, but it is intended to be used with any type of data. From this point on, it should be checked if the model still has a good behavior when considering other type of information that doesn’t belong to the Crunchbase. It could be also an interesting task to extend this work for other types of subject, for example, twitter and news data.

1.4 Goals

The main goal is to develop a model that can be applied to different information sources and that is able to channel the different data to the different categories in the right way. The very first goal is to be able to structure the extracted data automatically, applying different Natural Language Processing techniques (Part-of-speech, N-grams, Named Entities) in the most efficient way in order to prepare the data for the next steps.

After having a data source in which is possible to apply classification models efficiently the main objective is to implement multiple multi-class classification algorithms and compare its performance with the latest known studies in similar problems and try to outperform them.

When the implementation stage is completed and we already have had the intended results, is intended to use the outcome of our research in other areas of knowledge and apply these models to web pages, news, tweets and assess its usage in other information sources.

1.5 Research Methods

The development of our work follows the Design Science Research Methodology (DSRM). This methodology is based on the result of specific evaluation and iteration guidelines in research projects, see Saunders, Lewis, and Thornhill (2009) and Peffers et al. (2007). The DSRM is an iterative process that starts with the identification and motivation of the problem, presented in Section 1.1 and 1.2 followed by a presentation of the objectives of the solution, that is presented in Section 1.4.

After the initial stages, the process is followed by the initiation of the design and development stages, demonstration and evaluation stages that are presented in Chapter 3 and 4. The last stage of this iterative process is presented in Chapter 5 that inlcudes the presentation of the outcome of our work.

1.6 Background

Artificial Intelligence (AI) is the base for the most recent areas of knowledge such as Machine Learning (ML), with the emergence of Machine Learning, Mitchell (2006) raises several questions that had to be addressed. For example, "How can we build machines that solve problems, and which problems are inherently tractable/intractable?". In the inductive learning area, a sub-area in ML, the learning methods are categorized based on the feedback that is given to the learner itself. When it comes to supervised learning this method is based on the input and output pairs. The expected result is fed to it as part of the training set. Some examples of supervised learning algorithms are Linear Regression, Logistic Regression, Neural Networks, Support Vector Machines, among others. On the other hand unsupervised learning is method that does not get the expected result as an input to it. Instead, it tries to label them (usually with numbers). It is also important to notice that typically this methods make use of another technique, Clustering, which groups the data samples into clusters based on a feature that they share among them. K Means Clustering is one example of an unsupervised learning algorithm. The existing literature is vast in Text Classification and Text Mining areas, however, when it comes to categorization, the literature has a bigger focus in less categories / classes experiments. The most common Text Classification approaches make use of Supervised Learning algorithms.

1.6.1 Natural language processing

The Text Classification task is the way to make an algorithm understand the content that is inside a human readable text and produce a result out of it, usually, assigning a category to it. Several techniques can be applied to compute text. The most common is the Bag-Of-Words. As said in Webster and Kit (1992) this is one of the initial steps of Natural Language Processing. It can also be called as tokenization, the step of splitting text into tokens. Each word is considered to be a token, and from this point on it can be fed to an algorithm. Often is intended to shrink the number of features to the maximum, this is meant to remove the occurrence of less valuable features. A clear example is the stop word removal. Stop words (a, for, the, if, an, but, etc) do not add any value other than completing the semantic of a sentence, see Wilbur and Sirotkin (1992). Besides stop word removal, it is often common to reduce the words to the most basic form. This is called Stemming, and as said in Willett (2006) the standard nowadays for the English language is to apply the Porter Stemming algorithm. Another way to process the features in a text is to attribute weight to it, as in TF-IDF model (Salton and Buckley 1988) where TF refers to the Term Frequency inside a document, and IDF, as the Inverse Document Feature. This can give us the weight of a word based in the number of its occurrences and the importance that it has inside a document. For instance, a stop word, will probably appear several times inside a text document, that is why it would have a low TF-IDF score, and that is why it is a good step to remove them. Besides this, also a more semantic approach is often take in place, for instance Part-of-Speech tagging, a morphosyntactic disambiguation task. In Màrquez and Rodríguez (2005) an experiment was made using POS tagging and Decision trees, and the results are very interesting with an accuracy rate of 90.6\% on unknown words when training with 2 million words of the corpus.

1.6.2 Support vector machines

When trying to solve Text Classification problems using Machine Learning techniques there are several algorithms to consider, one of them being Support Vector Machines (SVM). Support Vector Machines where first introduced by Sain and Vapnik (2006) as a solution for a binary problem with two categories associated with pattern recognition.

Support Vector Machines consist in an algorithm that can determine the best decision limit between different vectors, each belonging to a group, in this study, a category. Based on risk / limit minimization principle Cortes (1995) for a given vector space where the goal is to find the "surface" of decision that split the different classes / categories.

SVM based models are often used in Text Classification problems since they behave quite well when used in supervised learning problems. The good results are due to the high generalization capacity of the method, which can be particularly interesting when trying to solve problems in big dimensions, has shown in Figure 1.1.

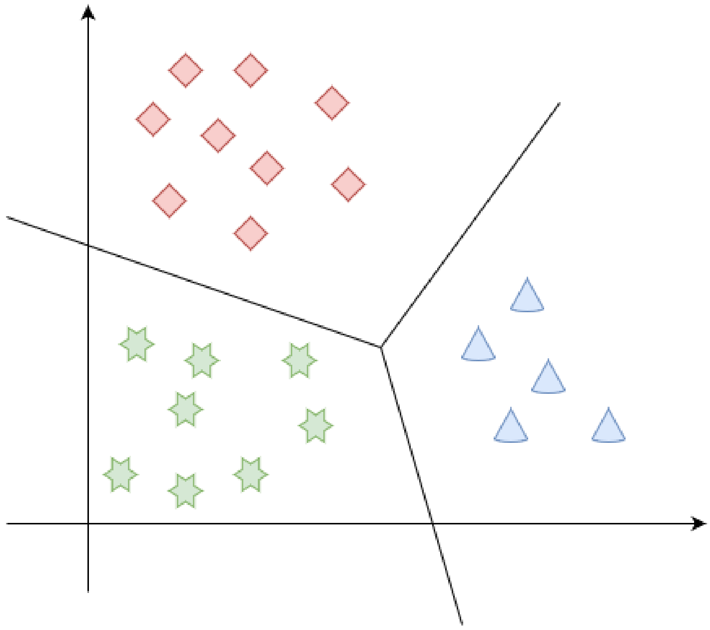

Figure 1.1: Support vector machine clustering diagram.

In Sain and Vapnik (2006) it is demonstrated that SVM outperform a lot of other algorithms when applied to Text Classification problems. In Rennie and Rifkin (2001) a comparation between Naive Bayes and SVM took place. Here, the comparation was done using two well-known datasets, different sizes of samples in multiple experiments and then the evaluation. It was found that SVM outperforms Naive Bayes by a large number, giving a much lower error rate, at that time, the lowest for the given sets of data. Also in (Basu, Walters, and Shepherd 2003) a Text Classification problem with a large number of categories is used to compare SVM and Artificial Neural Network (ANN). The results are very clear for both recall and precision, both indicating the differences in performance of the SVM and ANN. The SVM once again outperforms ANN, it is concluded that the SVM is much more suitable for this type of problems, since the performance is better and it is a less complex algorithm (computationally). Additionally, it is also tested the results of a reduced feature set against a large feature set, in here, the small feature set using the SVM has a much better performance, improving its results.

1.6.3 Fuzzy fingerprints

Fuzzy classification is any process that makes use of either a fuzzy set or fuzzy logic. Fuzzy classification can be defined as a grouping process where every item with the same features is included into a fuzzy set. A Fuzzy classifier is an algorithm that can assign a label to a given item using only its features. A Fuzzy classifier works in a same way as a general classifier, meaning that uses a set of training data combined with a training algorithm in order to learn how to predict class labels. The wide definition of a Fuzzy classifier allows a variety of models that can be defined using this type of algorithm. An example is prototype-based classifiers, a good example for this being K-nearest neighbor (KNN) classifier. Typically, in KNN an item is labeled with the majority of neighbors in a range using a reference set of data. In a Fuzzy KNN approach, not only the distance to its neighbors is considered but also its soft-labels. Usually, a set of prototypes with soft-labels is constructed and a class is obtained by combining the similarities among the given sample and the prototypes. There can also be other implementations of prototype-based classifiers (Parzen classifiers, Neural Networks, etc). Another good example of Fuzzy classifiers is rule-based classifiers. This is the most common approach to a Fuzzy Fingerprints classifier due to its simplicity. In the most basic form, it can be defined with *if* statements (*if a and b then class X*). The X label is the outcome of a met condition to the sample and it can be a linguistic label (the name of a concrete class for instance) or a function. However, these classifiers have a big difference regarding the training mechanism, in order to train a Fuzzy Classifier, it is required to partitioning of the data space by its features, see Babuska (1998).

1.7 Document Structure

This document is decoupled into different sections. In the next chapter it is possible to check the literature review, in here there is an introduction to Text Classification state of the art as well as a description of the most used technologies to solve similar research questions. It is also in here that the most recent Natural Language Processing techniques are presented and a brief introduction to similar art regarding the Crunchbase dataset.

Chapter 3 is where the Dataset is described , here it is presented the extraction process and the details of the data transformation since it was collected. There is also an analysis of the Dataset, this analysis was the first approach to the data and it was crucial to this dissertation development.

Chapter 4 includes the description of every experiment made in this dissertation. It is in this section that the pre-processing techniques and Machine Learning algorithms applied to the extracted data are presented. Not only the method and approaches but also the results are present in this section.

Chapter 5 is the last chapter of the document and it presents the final conclusions that can be retrieved from the presented work and also the “Future Work” suggestions.

State of the Art 2

This chapter presents the latest research studies that are connected and relevant for our dissertation. Our dissertation aims to make use of Text Classification, therefore the first section demonstrates the most relevant studies applied to several areas of knowledge. Text Classification problems often make use of Machine Learning algorithms. Thus, the next section 2.2 presents the latest literature regarding the most relevant Machine Learning algorithms. One of the most relevant areas of knowledge for Text Classification challenges is Natural Language Processing, therefore, Section 2.3 focus on the most recent work regarding feature selection and text processing techniques. The last section aims to present the latest know studies that use the Crunchbase dataset, here, it is very important to do a deep search for similar work and use it as a comparison source.

2.1 Text Classification

Text based classification has become a major researching area in the last few years, specially because it can be used in a large number of applications. Many different areas can make use of the outcome of Text Classification research. In Pang, L. Lee, and Vaithyanathan (2002) the authors applied Machine Learning algorithms to classify documents by sentiment, more precisely movie reviews from Internet Movie Database (IMDb). Also in Jindal, B. Liu, and Street (2007) an experiment to spam detection in customer reviews took place to check if false opinions were given to a product. It can also be applied to social media, as in K. Lee et al. (2011) that the authors applied several algorithms to tweets trying to find “trending topics”, or in Arts (2015) that the authors used Twitter information to develop an automated detection model to find rumors and misinformation in social media, having an accuracy of 91%. These are examples of binary classification problems, when it comes to multiple categories, also known as multi-class, the problem is harder to solve. The authors in Homem and Carvalho (2011) developed a model based on a Fuzzy Fingerprints technique to be able to find an author of a text document using a large dataset of newspaper articles from more than 80 distinct authors having almost 60% of accuracy results. Also Rosa, Batista, and Carvalho (2014) and Czarnowski and Jedrzejowicz (2015) make use of the same technique to solve a multi-class classification problem when trying to find events and twitter topics using textual data.

2.2 Methods and Approaches

When trying to build a capable model there is a large number of approaches than can be used. Lately, one of the most common approaches is using Machine Learning algorithms as said in Ikonomakis, Kotsiantis, and Tampakas (2005) that explains the Text Classification process using Machine Learning algorithms. One of the widely used algorithms to solve this problem is Support Vector Machines. In Sun, Lim, and Ying Liu (2009) it is applied to multiple datasets and compared to a set of SVM variants that use weights. It has been highly investigated and compared with other algorithms when approaching binary classification problems. Rogati and Yang (2002) puts several algorithms to the test applying them to the Reuters-21578 and small portion of Reuters Corpus Version 1 (RCV1) datasets. Here, it is possible to check that SVM outperforms most of the other algorithms by a large margin. Rennie and Rifkin (2001) also make a very interesting comparison between SVM and Naive Bayes in a multi-class classification problem applied to two well-known datasets, 20 Newsgroups and Industry Sector, here it is demonstrated that the error generated by SVM is much lower in comparison with Naive Bayes.

However, there are a large number of algorithms that can be used to address this type of problems, Colas and Brazdil (2006) have deeply studied the SVM algorithm and compared it with Naive Bayes and K Nearest Neighbors, and got to the conclusion that even though the SVM can behave slightly better for some use cases, it is a much more time consuming task. For a large number of documents, the required time to train the algorithm increases drastically and the gain in performance can be short. Also, the SVM algorithm is very complex in comparison to the ones previously referred. When analyzing Naive Bayes algorithms we can take into consideration that it is a probabilistic algorithm, with this as said in Murphy et al. (2006) the results are a probability distribution, therefore it is possible to tell about result uncertain. With this, Howedi and Mohd (2014) used a Naive Bayes Classifier for author attribution to a dataset called AAAT dataset (i.e Authorship attribution of Ancient Arabic Texts) obtaining results up to 96% classification accuracy. This shows that Naive Bayes should also be considered when trying to address multi-class classification. More recently, Xu (2018) also used a Naive Bayes Classifier approach on 20 newsgroups and WebKB, here, additionally, a comparison between different Naive Bayes approach take place, comparing Multinomial, Bernoulli and Gaussian variants of the algorithm achieving results of 95%. The performance and overall simplicity of Naive Bayes makes it a very attractive alternative for several classification tasks. However its results are mainly obtained from an unreal assumption of independence. For this, there has been a major focus on investigating the algorithm itself. In Domingos and Pazzani (1997) it is demonstrated that the Naive Bayes algorithm can have a surprising behavior on classification tasks where the result itself appears not to be as relevant as expected.

Decision Trees are also one of the most used algorithms in Text Classification tasks. A Decision Tree is a simple structure where non-terminal nodes represent tests to one or many attributes, and the terminal nodes reflect the result of the decision itself. The robustness for very noisy data and the ability to learn disjunctive expressions seems very appropriate to document classification. One of the well known algorithms for Decision Trees its the ID3 (Quinlan 1986) having as its successors the C4.5 (Murthy, Kasif, and Salzberg 1994) and also C5.1. It is a top down method that builds a Decision Tree classifier recursively. For each Tree level, ID3 selects the attribute that has the biggest information gain.

In Homem and Carvalho (2011) it is described the usage of a Fuzzy Fingerprints technique to authorn classification when using a large set of newspaper articles, having more than 80 different authors (labels) where it is achieved an accuracy score of around 60%. Another Fuzzy Fingerprints implementation on a multi-class classification task is performed in Rosa, Batista, and Carvalho (2014) and Czarnowski and Jedrzejowicz (2015) when trying to attribute events and twitter topics using only text.

2.3 Features

Natural Language Processing tasks have a huge impact in Text Classification. The Machine Learning algorithms play a fundamental role in Text Classification and therefore its input is one of the major success factors. Most of the algorithms play with vectors and those vectors usually hold text features within. One of the most common ways to represent textual data is the bag-of-words approach Harris (1954), since it is a very simple and efficient way to quickly feed an algorithm and check what can be its potential behavior. This method consists in a simple breakdown of a sentence into a set of words that are part of it together with its frequency count. Usually it has a decent performance, and in some cases, if the dataset is already very rich in terms of features it can be a good implementation. This type of approach loses the semantic form of a sentence, and for that can lose some context. However, when the data is sparse and has a high dimension this technique might not be enough. For that, several times it is possible to use a similar technique that preserves the semantic of a word, yet splitting it into words, this technique is called tokenization. In Webster and Kit (1992) this technique is presented and it is demonstrated a clear notion of word and token. Tokenization is also used as an initial technique when approaching text mining problems, it is also one of the root origins for other techniques. Still suffers from the same escalation problem of the bag-of-words, even thought the semantic doesn't get lost over the sentence deconstruction. It is found over time that a word/ token itself might not contain a significant information. Joulin et al. (2016) described an experiment using ngrams (bag-of-ngrams), this consists in moving a N window (usually 1,2 or 3) along each sentence and collect the unique combination of words along with its count. In this work it is compared the bag-of-words with the ngrams approach it is possible to check that it has a big improvement along the entire set of experiments, this is due to the fact that each feature now has at least the double of the information than before, therefore it adds a lot more context to the algorithm. When it comes to analyzing the features for each token there is a set of techniques that are commonly used, one of them being Lemmatization Toman, Tesar, and Jezek (2006) and D. Zhang, Chen, and W. S. Lee (2005). In Plisson, Lavrac, and Mladenic (2004) it is referred that Lemmatization is the way to normalize a word. Lemmatization is a way to prepare text data for further usage and it is widely used when working with text classifiers. Lemmatization it is not just the process of removing the suffix of a word, it also analyses the morphological structure. Usually, this can mean removing the plural of a word or just finding its radical form. However, there are many cases that this is not enough, for that, Lemmatizers produce another output. Take verb "to be" for instance, it can take a lot of forms in a sentence (are, is, been) but when found it always produces "be". There is a similar approach that is much lighter of word normalize, Stemming. Stemming Sharma and Cse (2012) is close to Lemmatization, however, it does not look into the morphosyntactic form of a word . Stemming, in opposite to Lemmatization, is the process to find the radical form of every word in a sentence and it is a standard for Text Classification problems Dalal and Zaveri (2011).

Not all words that compose a sentence add valuable information to it, these are commonly called stopwords. Why stopwords? Because they do not add any value to the information in a sentence but they are very used. In Saif et al. (2014) a comparison between different stopword lists applied to Twitter Sentiment Classifiers took place. Here, it is possible to check that the stopword removal drastically improves the performance of the algorithms. Also, not only the removal but also the quality of the stopword list generates a big difference between themselves. Considering Dolamic and Savoy (2010), another comparison between different stopword lists is made, here, a list of 571 words against another with only 9. The stopword removal has once again an improvement in the algorithm performance, but here, it also proves that the gain of having a more robust stopword list when applied to the English language it is not very significant. When retrieving information from a sentence it is important to understand it from a semantic point of view. A way to do this is to use Part-of-Speech Tagging, this is a very common word category disambiguation technique. It breaks down each word in a sentence into a token with the respective tag, this tag is the Part-of-Speech (Name, Noun, Adjective, Verb, Adverb, Preposition, Conjunction, Pronoun, Interjection). Pranckevicius and Marcinkevicius (2017) approached a multi-class classification problem for Amazon product reviews. Here, the authors used a Logistic Regression approach together with Part-of-Speech Tagging. Every experiment performed significantly better being the only exception the unigram experiment. Also, Hrala and Král (2013) made a comparison between Lemmatization and Part-of-Speech tagging to represent and classify Czech documents, considering that the POS-Tagging has a big impact when it comes to document classification tasks. However, even with this amount of information, once fed to an algorithm all of the features have the same impact to it. If we think about a sentence, there are parts of it, that describe it better than others, that differentiate themselves and can quickly suggest a topic just by reading them. For instance, if we think about the combination of words “an application”, it is possible too see that does not really scream any meaning about a sentence, therefore, inside a document is not very relevant. On the other hand when considering words like “social network” it can resemble immediately a topic related to “Software” or “Internet”, however, it this set of words appear several times in one or more documents it may not be so relevant to the scope of the work. To address all this questions it is very common to attribute weights to parts of sentence, where the “heaviest” part is the part that can best differentiate a sentence or a document and the “lightest” is the one that doesn’t add much more detail. A common application of this technique in Text Classification is the Term Frequency- Inverse Document Frequency (TF-IDF) Lilleberg, Zhu, and Y. Zhang (2015) used a combined approach between TF-IDF and word2vec to a news dataset having a result of more than 90% accuracy.

2.4 Crunchbase Classification

An attempt to automatically extract information from an older version of Crunchbase has been made in Batista and Carvalho (2015). At that time, Crunchbase contained around 120K companies, each classified to one out of 42 possible categories. The dataset also contained category "Other", that grouped a vast number of other categories. The paper performs experiments using SVM, Naive Bayes, TF-IDF, and Fuzzy Fingerprints. To our knowledge, no other works have reported Text Classification tasks over a Crunchbase dataset.

2.5 Summary

This chapter presents the latest known studies for the different areas of knowledge that are used along our dissertation. Text Classification is a big research area nowadays, therefore it is the first section 2.1 that is presented. Since Text Classification problems make use of Machine Learning algorithms and Natural Language Processing techniques, in sections 2.2 and 2.3 report the most relevant work for this dissertation. Unfortunately, the literature regarding our set of data is not vast, therefore, section 2.4 can only present one work to use at a start.

Dataset 3

All the experiments reported in the scope of this work use the Crunchbase dataset as its main source of data. In this chapter we analyze and describe the dataset in detail. Crunchbase database is a large source of information to use having a large amount of data for more than 600000 companies. This is a lot of information, however some of it is not relevant for our work and therefore it will not be considered. The steps for the database trimming as well as data analysis are explained in the following sections.

3.1 Extraction

Crunchbase is a world wide company database with over 600000 companies in its records and it is a very good source of information to use for our work. To have access to the data the Crunchbase Team kindly provided us an academic research key at 18th September of 2018 available for six months. Crunchbase exposes a REST API that offers access to their data containing all the information that is present in the official Crunchbase Website in order to be used by other applications. Crunchbase has a complete Data Model that can be accessed from the API, for that, they offer a "Daily CSV Export" that contains separate files for companies, people, funding rounds, acquisitions, Initial Public Offerings,... in order to retrieve data without any coding against the REST API.

However, this CSV export is not complete and therefore it cannot be used for our work. Instead, we need to retrieve all the information for each company individually, for that, Crunchbase offers a Node List that holds the respecting references to access the API for the full information.

The file that contains the references for the companies (organizations.csv) has three columns, "name" holding the Company Name (e.g "Formel D GmbH"), "permalink" that holds the endpoint for a specific company (e.g "/organizations/formel-d-gmbh") and "updated_at" that holds the last updated timestamp for that company (e.g "2018-04-17 08:14:17"). This file as a total of 695167 companies.

When making an HTTP GET Request to the Crunchbase API for a specific company, we get a JSON response that has the full company information including its relations and metadata, as illustrated in Figure 3.1. To be able to extract the full information about

```
{
  "metadata": {...},Number of Companies by category labeling number.
  "data": {
    "type": "Organization",
    "relationships": {...}
    "uuid": "000014da0c46b9cb09413a93c027b119",
    "properties": {
      "rank": 152571,
      "name": "Resilio", (...) Number of Companies by category labeling number.
      "founded_on": "2016-11-01",
      "role_group": false,
      "api_url": "https://api.crunchbase.com/v3.1/organizations/resiliohq", (...)
      "profile_image_url": "http://public.crunchbase.com/t_api_images/ffatvhppkjvue0
         g2h7xp",
      "description": "By combining state of the art (...) system.",
      "phone_number": "+004561672261",
      "num_employees_max": 10,
      "stock_exchange": null,
      "short_description": "Resilio is developing smartphone-based resilience
         training.",
      "homepage_url": "http://www.resiliohq.com", (...)
    }
  }
}
```

Figure 3.1: Crunchbase API response example.

each company present in the Crunchbase database an iteration through the companies file took place using the permalink reference to perform an HTTP Request and retrieve the JSON object. After retrieving each JSON the data was saved into a SQLITE Database. The database table has an auto incremental ID, the organization_name column (that holds the organization name) and a JSON column that contains RAW JSON data. An extraction took place between the 17th and 18th December 2018, the first company extracted was at 2018-12-17 14:26 and the last at 2018-12-18 09:22:47, this took roughly 19:20 hours and produced a total of 695167 database entries producing an SQLITE file with a total size of 12,3 Gigabytes.

3.2 Creating a Minimal Database

With the initial extraction it was collected a set of RAW JSON data with very complete information about each company. Each JSON has a metadata object related with the API itself, a relationship field that relates this company with several other entities from the platform (investors, founders, investments, board members, categories, news, products, office locations, among others), when it comes to the information regarding the organization it contains the creation date, a description, a short description, founded date, phone number, contacts, among others.

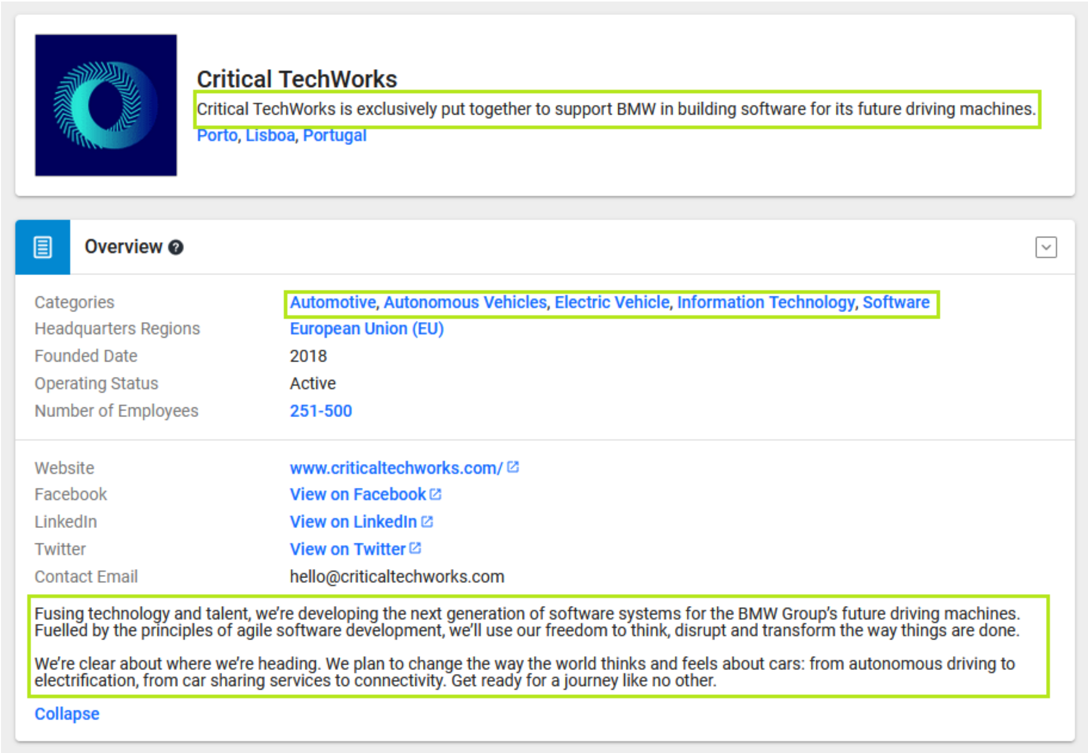

Figure 3.2: Crunchbase page data extraction example.

In Figure 3.2 it is possible to have a clear vision about the information present in Crunchbase. Also, it is possible to check what is extracted from the web page and used to feed the algorithms highlighted in green. The initial extraction from Crunchbase produced a new SQLITE Database. However, after an initial analysis to the RAW data, there were some problems and some of the original RAW JSON responses were not retrieved/ stored properly, to solve this, we created a new database (DB1) without them. The extracted JSON entries had a lot of information, however not all of this information is relevant for our task, the information required for our work is the URL for identification of the company, the company name and a JSON that is an extrapolation of the original one that contains the "description", "categories", "short_description" and "groups" fields.

We took the opportunity of filtering unparseable data to make this data transformation processing and remove unwanted information from each Database entry, producing database entries having the form presented in Figure 3.3. At the end, the new database (DB1) had a total of 685442 entries (losing 9725 entries) and the file holding the information 704 Megabytes holding only parseable JSON entries containing relevant fields for our task.

The last step when transforming the dataset removed all entries that did not belong to any group as well as all the entries that did not contain any description. All the remaining data was then exported into a new database (DB2) with a total of 405602 records, stored into two different Tables: *train* containing 380602 records that will be used from training

```
url: https://api.crunchbase.com/v3.1/organizations/formel-d-gmbh
name: Formel D GmbH
info:
{
    "description": "Formel D GmbH is a automotive manufacturer and supplier for the
        world.",
    "categories": ["Automotive", "Manufacturing"],
    "shortdescription": "Foritmel D GmbH is a automotive manufacturer and supplier
        for the world.",
    "groups": ["Manufacturing", "Transportation"]
}
```

Figure 3.3: Database entry example.

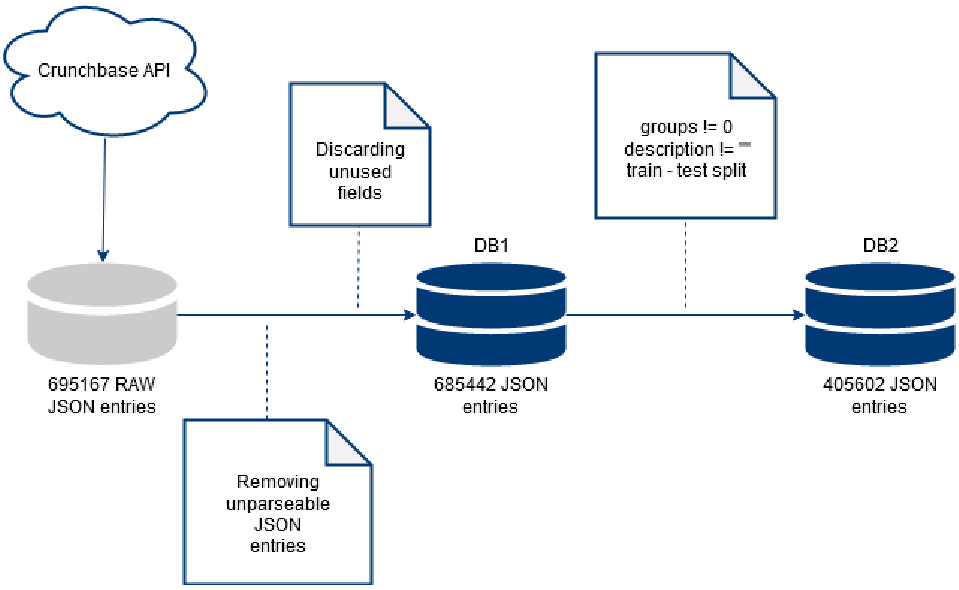

Figure 3.4: Filtering and data transformation diagram.

our models, and *test*, containing 25000 records, that will be used for evaluating our models. The complete extraction process as well as the database transformation is reflected in Figure 3.4. Having a database with a lower amount of records but already free of unparseable data and data that does not contain any value for our study is a big advantage for the next stages.

3.3 Data Description

Each company has a *description* field, that describes it for whoever wants to have a brief notion of what it does and the areas that it belongs. Adding to this, it also has a *short_description* that is a summary of the *description* itself. Each company belongs to one or more category group and each group has a number or categories. The Group is wider (e.g Software) and the Categories are more specific (e.g Augmented Reality, Internet, Software, Video

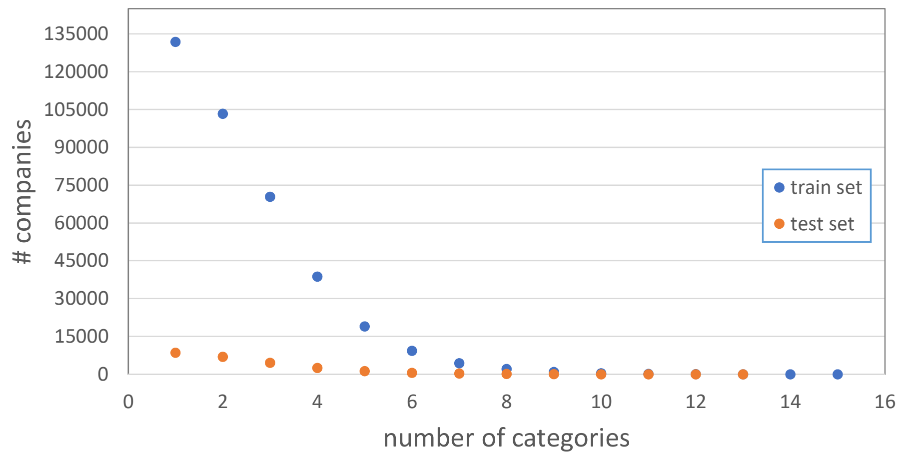

Figure 3.5: Number of companies labeled with a set of categories.

Games, Virtual Reality). Each category can be present in more than one Group, for instance "Alumni" appears as a category for "Internet Services", "Community and lifestyle", "Software", etc. Also, "Consumer" appears in "Administrative Services", "Hardware" and "Real Estate", among others. Our dataset has a total of **46** groups and in total 405602 entries. Each company can have multiple groups, the histogram in Figure 3.5 shows how each of the companies are labeled with a set of categories, from the graph, it is possible to assess that most of our companies have between 1 and 3 categories and a very low amount of them are over 7 categories.

Crunchbase dataset also contains companies that are not labeled with any group, these companies, should not be considered as a valid database entry. On the other hand, the number of maximum labels for a given company is **15**, these entries, even though they are considered as valid, they are not very relevant since if a company is labeled with this amount of groups it means that inserts itself into several different areas and therefore it will introduce a low value description to our model. The average groups assigned to each company is **2.41**, between **2** and **3** groups, however, over **125000** companies only contain one group labeled.

There was no information about the distribution of companies by the **46** groups, this information is relevant because it allows us to understand the balance of the data itself. Thus, using the extracted data is possible to obtain a distribution, as shown in Figure 3.6. Here, it is possible to understand that the dataset has an unbalanced distribution of data, for example, “Software” has over **100000** records. On the other hand, for “Platforms”, “Music and Audio” and “Gaming” has less than **20000**. The difference from “Software”

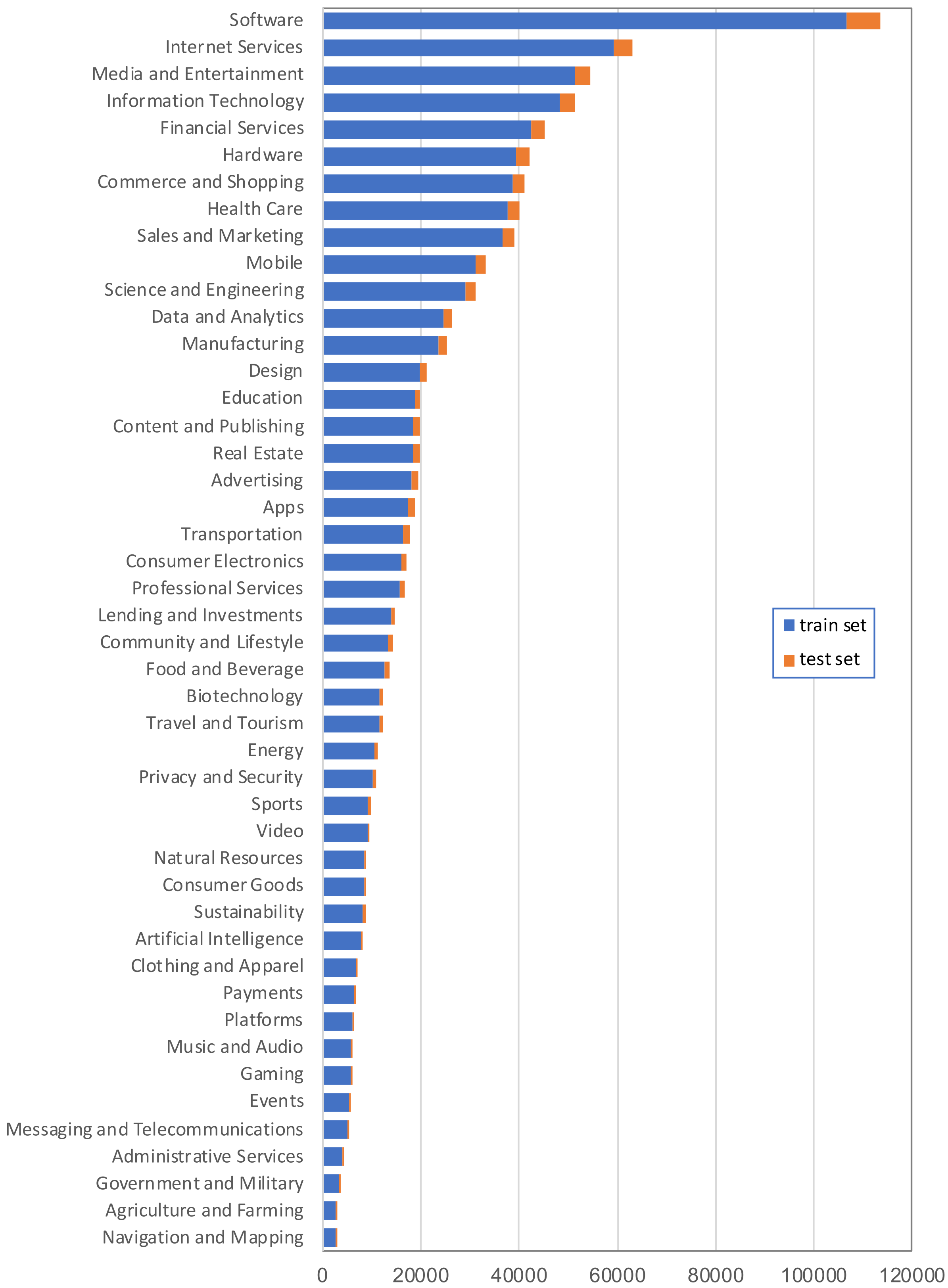

Figure 3.6: Companies distribution by groups.

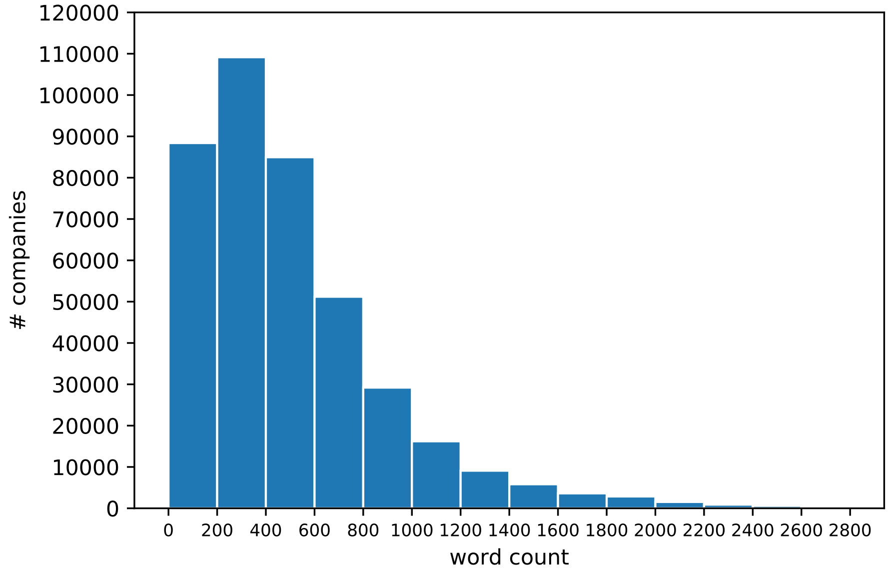

Figure 3.7: Companies distribution by word count.

to the second most common group “Internet Services” is also very significant, “Software” almost doubling the database entries.

When analyzing each description it is also possible to extract additional information. Considering Figure 3.7 it is possible to conclude that the average word count for a description is around **518** words and the maximum and minimum word count is **8184** and **1** respectively (including stopwords). Figure 3.7 shows that most of the descriptions are included between the **200-400** words range, followed by **0-200** and **400-600**, thus, we are not dealing with large texts and we can use that as an advantage for the pre-processing stage performance. Despite the fact that short texts can be a good point for performance, it might mean that each description is very generic and might not be rich enough to use as input for a Machine Learning algorithm.

3.4 Summary

The dataset is the main focus for the proposed work. In this chapter details for the Crunchbase dataset are presented along side with an initial description of the data. Initially, the extraction process was explained in detail including samples for the collected data. Afterwards, an analysis for the retrieved data took place and the need to filter unwanted data arised. Thus, a minimal database creation process took place, explained in section 3.2. The final work for the dataset is the data description in the section 3.3, where it is described hidden information that can be obtained from the raw data source.

4 Experiments and Results

This chapter describes the experiments of our work. The differences among all the approaches are explained as well as the used evaluation metrics along the development and research of this dissertation. The outcome of each experiment is included in each section and each experiment represent a research increment and make use of the acquired knowledge along the development process, therefore, they relate between them along the document.

4.1 Data Normalization and Pre-Processing

From the examples in Figure 3.2 and 3.3 it is possible to see that the descriptions may be ambiguous. To solve the ambiguity among more than 400000 companies descriptions a normalization stage was executed before the initial experiments with classification models. Pre-processing processes are a common approach among Text Classification steps and usually they include text normalization tasks applied to the complete dataset as an initial step of the development process.

For each of the upcoming experiments a normalization pre-processing took place for the input to be consistent among them. The pre-processing process includes lower casing every word of the corpus, removing punctuation (keeping only alphanumeric characters), splitting each sentence into tokens and keeping only the words that are not included in the NLTK list of stopwords for the English language, see Figure 4.1.

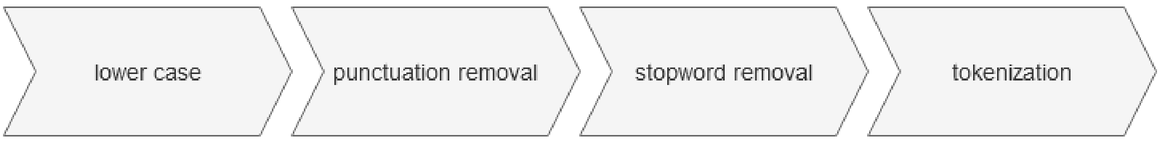

Figure 4.1: Normalization steps.

4.2 Evaluation Metrics

In order to evaluate the performance for each of the exeperiments it is necessary to calculate the respective metrics from each classifier. The metrics are calculated based on:

- **true positives (TP)** - when a company belongs to a given category and the classifier correctly outputs that category
- **true negatives (TN)** - when a company doesn't belong to a category and the classifier correctly outputs that it doesn't
- **false positives (FP)** - when a company does not belong to a given category but the classifier says that it does
- **false negatives (FN)** - when a company belongs to a given category but the classifier says that it doesn't
- **total predictions** - the amount of predictions made

After collecting these values, all of them are summed into global metrics (micro-average) so that it is possible to calculate the accuracy, precision, recall and F-measure using the following formulas:

$$Accuracy = \frac{true\,positives + true\,negatives}{total\,preditctions} \tag{4.1}$$

$$Precision = \frac{true\,positives}{true\,positives + false\,positives}$$

$$Recall = \frac{true\,positives}{true\,positives + false\,negatives}$$

$$F - measure = \frac{precision * recall}{precision + recall}$$

4.3 Initial Experiments

With the first experiment the main goal is to quickly address what is the behavior of the algorithms and to check what can be developed from this dataset in order to create a unique classifier that for a given company can return which is the most likely group. This is possible using only the "description" field as a text input and the "groups" field as the labels for the given description. After a normalization pre-processing task an experiment took place using scikit-learn (sklearn), a library that integrates multiple state-of-the-art algorithms Pedregosa et al. (2011). For this initial experiment we used the complete dataset in DB2 and a test set with 25000 companies considering only one group for each description.

4.3.1 Linear Support Vector Classification

In order to quickly assess the possible outcome of an SVM implementation on our dataset we used an already tested and known classifier from sklearn called LinearSVC, the algorithm was used with no additional parameter tuning. At this moment, the data that was fed to the algorithm was normalized as described in section 4.1.

4.3.2 Multinomial Naive Bayes

One of the most common approaches in multi-class classification is to use Naive Bayes classifiers. We implemented an initial Naive Bayes approach using sklearn Multinomial-NaiveBayes with no additional tuning and fed with the same data source as in section 4.3.1.

4.3.3 Results

From the presented results in tables 4.1 and 4.2 it is possible to see that both methods can have a good performance when applied to our dataset. Considering the work at Batista and Carvalho (2015) our results are very encouraging, immediately reaching the same accuracy values of around **40%.** In our dataset we do not have the “Other” category therefore it is expected that is possible to improve these results right from the start.

These results only represent an initial assessment of a possible outcome for our work. It is possible to see initial hints for the possible challenges, for instance, several classifiers generating **0** values for categories under **400** entries. This might mean that, in future, the test data might have to be increased in order to produce results to be analyzed.

4.4 Multi-class Classification

For the first experiment the approach is to make a data transformation. Every company can have multiple groups associated to it, the first experiment is to represent only one to one relations, exploding each description to the amount of labels that are attributed to it. Although SVM’s are mainly designed to work with binary classifiers only, there are some approaches that can deal with multi-class classification. For this, it is possible to follow

Groups	**Precision**	**Recall**	**F-measure**	**Samples**
Financial Services	0.603	0.684	0.641	2109
Information Technology	0.277	0.290	0.283	1982
Media and Entertainment	0.381	0.403	0.391	1821
Health Care	0.547	0.564	0.556	1574
Software	0.232	0.231	0.231	1475
Manufacturing	0.485	0.506	0.496	1418
Science and Engineering	0.437	0.413	0.424	1201
Mobile	0.296	0.304	0.300	1198
Advertising	0.409	0.445	0.427	997
Education	0.550	0.558	0.554	899
Data and Analytics	0.227	0.202	0.214	741
Sales and Marketing	0.188	0.155	0.170	686
Real Estate	0.459	0.502	0.480	671
Design	0.342	0.349	0.345	653
Consumer Electronics	0.144	0.113	0.127	627
Privacy and Security	0.483	0.442	0.462	615
Food and Beverage	0.550	0.582	0.565	591
Internet Services	0.156	0.128	0.140	579
Commerce and Shopping	0.233	0.216	0.224	555
Natural Resources	0.535	0.539	0.537	532
Travel and Tourism	0.506	0.508	0.507	447
Transportation	0.379	0.402	0.390	430
Professional Services	0.309	0.279	0.293	402
Music and Audio	0.566	0.527	0.546	357
Sustainability	0.329	0.312	0.320	353
Gaming	0.478	0.502	0.489	301
Sports	0.367	0.364	0.366	291
Community and Lifestyle	0.160	0.139	0.149	288
Apps	0.071	0.059	0.065	170
Hardware	0.194	0.211	0.202	166
Consumer Goods	0.215	0.204	0.209	152
Administrative Services	0.246	0.223	0.234	130
Artificial Intelligence	0.079	0.060	0.068	117
Energy	0.105	0.081	0.091	99
Agriculture and Farming	0.352	0.333	0.343	93
Platforms	0.075	0.043	0.055	92
Payments	0.033	0.024	0.028	84
Government and Military	0.148	0.143	0.145	63
Events	0.000	0.000	0.000	11
Navigation and Mapping	0.000	0.000	0.000	11
Biotechnology	0.000	0.000	0.000	4
Messaging and Telecommunications	0.125	0.250	0.167	4
Video	0.000	0.000	0.000	4
Clothing and Apparel	0.200	0.333	0.250	3
Content and Publishing	0.000	0.000	0.000	3
Lending and Investments	0.200	1.000	0.333	1
macro avg	0.275	0.296	0.279	25000

Accuracy: **0.390**

Table 4.1: SVM baseline results

Groups	**Precision**	**Recall**	**F-measure**	**Samples**
Financial Services	0.533	0.810	0.643	2109
Information Technology	0.242	0.579	0.341	1982
Media and Entertainment	0.301	0.645	0.411	1821
Health Care	0.616	0.597	0.606	1574
Software	0.272	0.215	0.240	1475
Manufacturing	0.397	0.707	0.508	1418
Science and Engineering	0.505	0.438	0.469	1201
Mobile	0.307	0.342	0.324	1198
Advertising	0.451	0.521	0.483	997
Education	0.629	0.623	0.626	899
Data and Analytics	0.418	0.104	0.166	741
Sales and Marketing	0.406	0.019	0.036	686
Real Estate	0.587	0.387	0.467	671
Design	0.501	0.285	0.363	653
Consumer Electronics	0.451	0.037	0.068	627
Privacy and Security	0.766	0.293	0.424	615
Food and Beverage	0.657	0.597	0.626	591
Internet Services	0.308	0.021	0.039	579
Commerce and Shopping	0.309	0.218	0.256	555
Natural Resources	0.689	0.500	0.580	532
Travel and Tourism	0.702	0.432	0.535	447
Transportation	0.621	0.209	0.313	430
Professional Services	0.586	0.169	0.263	402
Music and Audio	0.797	0.132	0.226	357
Sustainability	0.466	0.116	0.186	353
Gaming	0.632	0.246	0.354	301
Sports	0.658	0.086	0.152	291
Community and Lifestyle	0.400	0.007	0.014	288
Apps	0.000	0.000	0.000	170
Hardware	0.667	0.012	0.024	166
Consumer Goods	0.333	0.007	0.013	152
Administrative Services	0.000	0.000	0.000	130
Artificial Intelligence	0.000	0.000	0.000	117
Energy	0.000	0.000	0.000	99
Agriculture and Farming	0.500	0.011	0.021	93
Platforms	0.000	0.000	0.000	92
Payments	0.000	0.000	0.000	84
Government and Military	0.000	0.000	0.000	63
Events	0.000	0.000	0.000	11
Navigation and Mapping	0.000	0.000	0.000	11
Biotechnology	0.000	0.000	0.000	4
Messaging and Telecommunications	0.000	0.000	0.000	4
Video	0.000	0.000	0.000	4
Clothing and Apparel	0.000	0.000	0.000	3
Content and Publishing	0.000	0.000	0.000	3
Lending and Investments	0.000	0.000	0.000	1
macro avg	0.342	0.204	0.213	25000

Accuracy: **0.413**

Table 4.2: Multinomial Naive Bayes results.

the "one-vs-all" approach Yi Liu and Zheng (2005), using the sklearn model LinearSVC this is already implemented and it generates the required classifiers based on the amount of classes present in the data. In addition to the normalization process the first experiment will also implement TF-IDF as a pre-processing step.

4.4.1 Data Transformation

To feed the model it is required to label each description with one group only. For that, every company that had more than one group associated originated another entry to the data but containing only one group associated. For better understanding:

```
{
       "description": "Faraday Venture Partners is a private investors club that
           offers an exclusive investment service to its Partners. We analyse
           innovative start-up projects in need of private financing and offer the
           best and most promising projects for investment to our Partners, co-
           investors and business angels. ",
       "groups": ["Financial Services", "Lending and Investments"]
}
```

Transformation example:

- "Faraday Venture Partners is a private investors club that offers an exclusive investment service to its Partners. We analyze innovative start-up projects in need of private financing and offer the best and most promising projects for investment to our Partners, co-investors and business angels" - **Financial Services**

- "Faraday Venture Partners is a private investors club that offers an exclusive investment service to its Partners. We analyze innovative start-up projects in need of private financing and offer the best and most promising projects for investment to our Partners, co-investors and business angels" - **Lending and Investments**

Our approach multiplies the number of entries in the dataset for the existing labels in each description. The results for the baseline multi-class experiment are presented in Table 4.3 for SVM, Naive Bayes and Fuzzy Fingerprints with the complete train dataset (380602 entries), the descriptions explosion originated a total of **917156** entries.

The results considered as a positive guess if the classifier produced a “yes” to a given label, and it was in fact a description labeled with that concrete group. On the opposite, it considered as a “no” if the classifier marked the description as not belonging to a group and it wasn’t, in fact, originally labeled with that group.

	Positive guess	Negative guess	Accuracy	Execution time
SVM	16920	8080	0.676	21m41s
NB	10374	14626	0.414	1m18s
Fuzzy Fingerprints	14475	10525	0.672	51s

Table 4.3: Multi-class results.

4.4.2 Metrics Calculation

In this particular case, since there are multiple entries for the same description, for each entry the only possible result is only one. This also means that each description is considered to be a different company, therefore, there is no way to correlate each other and find the multiple cases, therefore, the only results considered for each entry is **correct** or **incorrect**. Having only this two types of results the only metric possible to calculate is the accuracy.

Accuracy is one of the most common metrics to use when evaluating performance for Machine Learning models. It can be defined as in the Equation in 4.1 and it represents the fraction of the number of accurate predictions over the total predictions that were made by the model itself. Even though it is widely used as an evaluation metric, it might not be the most relevant one, we can have an high accuracy score but with a low precision. This might mean, that our model might be close to be precise, but not quite yet. For example, if a model always opts for a "negative guess" it will probably produce an high accuracy measure, even though is not really predicting anything.

From the result table 4.3 it is possible to assess an improvement from the baseline experiments in both scenarios. The initial experiment for SVM presented an accuracy score of 39%, the first multi-class experiment presented an improvement of 20%, with 67%. Multinomial Naive Bayes, also outperformed the baseline results as expected, with 41% against a previous score of 31%. Right from the start, it is also possible to notice that the execution times for both experiments are much different, with the SVM being much slower than Naive Bayes, as expected.

4.5 Binary Classification Models

Another way to feed the algorithm is to split the data into different binary classifiers (Dilrukshi, De Zoysa, and Caldera 2013), this is similar to the previous approach, but instead of letting the algorithm control each classifier in a black box way, it is possible to tweak and tune every classifier to its own needs. This section presents the experiments using a multi classifier approach. Each experiment will be compared to each other further on this document, however, it is in this section that is found what they have in common. As a basis, all of the experiments are developed with the Natural Language Toolkit (NLTK) Loper and Bird (2002) which is currently one of the leading platforms to work with human language data, sklearn algorithms and will apply the same normalization task from section 4.1. From the initial experiments in section 4.4 we conclude that both of SVM and Naive Bayes are valid approaches, therefore all the experiments will be implemented using both algorithms as well as a final experiment using Fuzzy Fingerprints classifiers.

4.5.1 Preparing data

To apply one classifier by group every classifier needs to have its own set of data. In order to do so, a group was considered to be a classifier. For every group, it exists two classes, if a company description belongs to that group gets into the true class - therefore 1 - if not it will be in the false class - therefore 0. For each group a dataset was created with a specific label matching its own class.

Example:

Group: **Financial Services**

"Faraday Venture Partners is a private investors club that offers an exclusive investment service to its Partners. We analyze innovative start-up projects in need of private financing and offer the best and most promising projects for investment to our Partners, co-investors and business angels" - **1**

"Faraday Venture Partners is a private investors club that offers an exclusive investment service to its Partners. We analyze innovative start-up projects in need of private financing and offer the best and most promising projects for investment to our Partners, co-investors and business angels" - **0**

4.5.2 Features: Word weights

In order to assess the word weighting technique that performs the best to use with our set of data we applied two initial experiments using word frequency and TF-IDF. For the word frequency approach we used a basic CountVetorizer from sklearn and for TF-IDF we used the TfidfVectorizer, both of them combined with the normalization steps in section 4.1. Table 4.4 presents the for both word weighting techniques using an SVM and Naive Bayes algorithms. From the results, it is possible to see a very big step forward regarding the possible performance outcome that these techniques can have when compared to the experiment in section 4.4. We can also conclude that for the initial assessment the best overall configuration is SVM using a TF-IDF approach. On the other hand, it is possible to see a poor behavior from Naive Bayes when using TF-IDF weighting.

		Accuracy	**Precision**	**Recall**	**F-measure**	**Execution Time**
Word Frequency	SVM	0.950	0.538	0.413	0.467	43m41s
	NB	0.951	0.548	0.440	0.488	27s
TF-IDF	SVM	**0.959**	0.696	0.420	0.524	4m9s
	NB	0.948	0.705	**0.020**	**0.039**	29s

Table 4.4: Word weighting results.

4.5.3 Stemming

One of the major focus of improvements for Text Classification algorithms is in the data pre-processing stage. That said, one of the possible approaches is Stemming. Stemming is a way of finding the stem of all the words in a sentence. What is a Stem? A Stem can be the radical form of a word. For example, if we consider all forms of the word “drive” (“driving”, “driver”, “drive”, “driven”), once stemmed, all will originate the same word, “drive”.

Example:

source: “faraday **venture partners private investors** club **offers exclusive investment service partners analyse innovative** startup **projects** need **private financing** offer best **promising** projects **investment partners coinvestors business** angels”

target: “faraday **ventur partner privat investor** club **offer exclus invest servic partner analys innov** startup **project** need **privat financ** offer best **promis** project **invest partner coinvestor busi** angel”

With this pre-processing step there is a big amount of detail that gets lost and can impact the way a human can interpret a sentence. However, when interpreted by an algorithm it can turn the comparison between the different sentences easier.

Usually Stemming is based on heuristics, therefore it can also introduce some errors namely over-stemming or under-stemming. Over-stemming appears when a given word gets so cutted off that it loses meaning. Under-Stemming, in opposite, happens when there are words that are forms of another ones and are not resolved to the same stem.

Considering this, the initial stemming approach was applied as a pre-processing stage to feed the SVM and Naive Bayes algorithms.

The results for the SVM implementation can be found in Table 4.5. When using Stemming in data pre-processing the results remain very similar to the ones in Table 4.4. However we can see small improvements in every metric with the exception of recall for the SVM implementation.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.805**	0.684	0.553	0.612	6929
Internet Services	0.856	0.591	0.285	0.385	3956
Media and Entertainment	0.903	0.706	0.471	0.565	3338
Information Technology	0.884	0.577	0.246	0.345	3108
Financial Services	0.947	0.825	0.666	0.737	2767
Hardware	0.913	0.671	0.346	0.457	2630
Commerce and Shopping	0.920	0.684	0.395	0.501	2527
Health Care	0.957	**0.846**	**0.699**	**0.765**	2521
Sales and Marketing	0.932	0.743	0.447	0.558	2387
Mobile	0.927	0.586	0.309	0.404	2017
Science and Engineering	0.944	0.748	0.422	0.540	1949
Data and Analytics	0.944	0.649	0.261	0.373	1595
Manufacturing	0.951	0.659	0.463	0.544	1576
Design	0.954	0.648	0.268	0.379	1305
Education	0.972	0.796	0.572	0.665	1226
Content and Publishing	0.959	0.664	0.335	0.445	1233
Real Estate	0.969	0.771	0.514	0.617	1231
Advertising	0.963	0.690	0.362	0.475	1156
Apps	0.952	0.460	0.083	0.141	1190
Transportation	0.967	0.763	0.421	0.542	1155
Consumer Electronics	0.958	0.561	0.115	0.191	1084
Professional Services	0.965	0.666	0.280	0.394	1018
Lending and Investments	0.971	0.672	0.431	0.525	933
Community and Lifestyle	0.965	0.562	0.091	0.157	888
Food and Beverage	0.981	0.780	0.609	0.684	844
Biotechnology	0.981	0.768	0.556	0.645	766
Travel and Tourism	0.982	0.806	0.505	0.621	723
Energy	0.982	0.781	0.573	0.661	754
Privacy and Security	0.979	0.746	0.348	0.475	666
Sports	0.983	0.751	0.448	0.561	607
Video	0.982	0.656	0.380	0.481	563
Natural Resources	0.984	0.729	0.522	0.608	579
Consumer Goods	0.980	0.661	0.270	0.383	571
Sustainability	0.982	0.686	0.392	0.499	574
Artificial Intelligence	0.983	0.726	0.297	0.421	509
Clothing and Apparel	0.987	0.757	0.483	0.590	470
Payments	0.986	0.641	0.362	0.462	409
Platforms	0.985	**0.333**	**0.029**	**0.054**	375
Music and Audio	0.990	0.788	0.489	0.603	403
Gaming	0.989	0.687	0.472	0.560	358
Events	0.987	0.671	0.283	0.398	367
Messaging and Telecommunications	0.988	0.525	0.198	0.288	313
Administrative Services	0.989	0.577	0.110	0.185	272
Government and Military	0.991	0.514	0.082	0.141	220
Agriculture and Farming	**0.993**	0.735	0.387	0.507	222
Navigation and Mapping	0.993	0.586	0.098	0.168	173
Total Average (micro-average)	0.960	0.705	0.411	0.519	n/a

Table 4.5: SVM binary classification - stemming results.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.769**	0.569	0.690	0.623	6929
Internet Services	0.805	0.407	0.505	0.450	3956
Media and Entertainment	0.874	0.523	0.635	0.574	3338
Information Technology	0.856	0.425	0.443	0.434	3108
Financial Services	0.939	0.730	0.705	0.718	2767
Hardware	0.891	0.479	0.454	0.466	2630
Commerce and Shopping	0.901	0.512	0.505	0.509	2527
Health Care	0.952	**0.785**	**0.719**	**0.750**	2521
Sales and Marketing	0.916	0.563	0.525	0.543	2387
Mobile	0.908	0.424	0.394	0.409	2017
Science and Engineering	0.916	0.467	0.556	0.508	1949
Data and Analytics	0.937	0.516	0.270	0.355	1595
Manufacturing	0.925	0.433	0.612	0.507	1576
Design	0.948	0.503	0.268	0.350	1305
Education	0.967	0.721	0.521	0.605	1226
Content and Publishing	0.949	0.471	0.350	0.401	1233
Real Estate	0.958	0.615	0.405	0.489	1231
Advertising	0.954	0.506	0.346	0.411	1156
Apps	0.940	0.269	0.157	0.198	1190
Transportation	0.957	0.573	0.290	0.385	1155
Consumer Electronics	0.950	0.347	0.161	0.220	1084
Professional Services	0.962	0.593	0.226	0.327	1018
Lending and Investments	0.963	0.510	0.483	0.496	933
Community and Lifestyle	0.959	0.228	0.069	0.106	888
Food and Beverage	0.975	0.657	0.518	0.579	844
Biotechnology	0.973	0.549	0.655	0.598	766
Travel and Tourism	0.974	0.573	0.376	0.454	723
Energy	0.971	0.527	0.485	0.506	754
Privacy and Security	0.975	0.599	0.218	0.319	666
Sports	0.975	0.440	0.157	0.231	607
Video	0.976	0.398	0.167	0.235	563
Natural Resources	0.977	0.510	0.463	0.485	579
Consumer Goods	0.973	0.299	0.142	0.192	571
Sustainability	0.975	0.421	0.293	0.345	574
Artificial Intelligence	0.979	0.390	0.059	0.102	509
Clothing and Apparel	0.981	0.496	0.266	0.346	470
Payments	0.982	0.303	0.088	0.136	409
Platforms	0.981	0.110	0.035	0.053	375
Music and Audio	0.983	0.435	0.141	0.213	403
Gaming	0.985	0.421	0.156	0.228	358
Events	0.982	0.087	0.025	0.038	367
Messaging and Telecommunications	0.986	0.180	0.035	0.059	313
Administrative Services	0.987	0.119	0.026	0.042	272
Government and Military	0.990	0.024	0.005	0.008	220
Agriculture and Farming	0.990	0.180	0.050	0.078	222
Navigation and Mapping	**0.991**	**0.000**	**0.000**	-	173
Total Average (micro-average)	0.949	0.517	0.456	0.485	n/a

Table 4.6: Naive Bayes binary classification - stemming results.

Using Stemming with Naive Bayes does not represent a major improvement on the overall results. In Table 4.6 it is possible to find the close results the ones presented in Table 4.4.

From Tables 4.5 and 4.6 it is possible to check that Software is the Group with the lowest accuracy score, however, it is also the one with the biggest sample amount, followed by Internet Services with nearly 4000 samples which is a little more than half of Software's sample. This might mean that this result is not so bad after all and that the sample is too big for the classifier to perform well enough. From the Tables it is also possible to check that Health Care has the maximum values for the remaining measures, setting the maximum precision at **0.846 (SVM)** and **0.785 (NB)**, the recall at **0.699 (SVM)** and **0.719 (NB)** and the F-measure at **0.765 (SVM)** and **0.750 (NB).** From Table 4.6 it is also possible to see that the amount of test samples has a big impact in the performance of the algorithm. When it comes to accuracy score, the highest is Navigation and Mapping at **0.991** being also the lowest sample count among all labels and the lowest is Software at **0.769** being the one with the highest label samples among all. In Navigation and Mapping, precision and recall came out as **0** making the F-measure calculation impossible.

- Software Sample

"styleme revolutionary virtual styling solution fashion brands provide online shoppers personalized social shopping experience powerful plugin ecommerce platform integrates 3d virtual fitting room online retail website solving biggest pain points online apparel retailers facing low conversions high return rates styleme founded 2014 aim transforming fashion ecommerce developed proprietary technology 3d scanning patented 3d geometric deform simulation layering technology worlds first true social media marketing toocost"

- Health Care Sample

"kidogo social enterprise platform improves access early childhood care education fitree healthcare fitness firm aims uphold development genuine healthy prosperous lifestyle use wearable technology mobile health applications option integrate platforms social media via mobile health model key patrons thriving interest delivering insight implementing growth towards aspects healthier lifestyle feel brand amuses eye user finds solution create healthier lifestyle fun efficiency use mobile wearable devices wearable devices " goto device " mobile health fitness technology outcome success developing ideal product greater understanding frustrations concerns user expert critic wearable technology market"

From the examples we can see that Health Care is a much more specific topic, therefore when we find words like healthcare, fitness, lifestyle, healthier, medical, care,.. is a much more immediate conclusion.

4.5.4 Lemmatization

Another approach for the pre-processing stage is to use Lemmas. The process of finding a Lemma is called Lemmatization and it is a way to find a normalized form of a given word. It is different from stemming in a way that considers the morphological structure of a word and tries to find its "normal" form, instead of its "radical".

Example:

source: "page capital spc business accelerator specifically designed ground address business **needs** early stage **startups** spc also operates investment arm pagevc spvc accelerates seedearly stage startups investorsadvisorsoperators"

target: "page capital spc business accelerator specifically designed ground address business **need** early stage **startup** spc also operates investment arm pagevc spvc accelerates seedearly stage startup investorsadvisorsoperators"

Usually, a Lemmatizer is a complex algorithm that makes use of a big dictionary in order to find the correct form of a given word. Given this, Natural Language Toolkit Loper and Bird (2002) provides a WordNetLemmatizer as an open source Lemmatizer that can be used in this experiment as a complement of the experiment in section 4.5.3.

Table 4.7 presents the results for lemmatization experiment, similar to Stemming in section 4.5.4 it does not represent a major improvement in the overall performance when compared to the base results in 4.4.

For the Naive Bayes approach the results are presented in Table 4.8. The results are once again very similar to the ones in section 4.5.4 experiment when it comes to the lowest and highest scores for precision, recall and F-measure with Health Care being the best overall Group and Navigation and Mapping, the worst. When it comes to the overall micro-average score the results are close, representing a small improvement in recall and F-measure and a slight decrease in accuracy and precision.

4.5.5 Part-of-speech tagging

One of the most interesting ways to make a machine understand the meaning of a sentence is using Part-of-Speech Tagging (POS-Tagging). With POS-Tagging it is possible to check the semantic of a sentence attributing tags to every word that compose it.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.806**	0.684	0.554	0.612	6929
Internet Services	0.856	0.590	0.293	0.392	3956
Media and Entertainment	0.904	0.708	0.476	0.569	3338
Information Technology	0.883	0.567	0.254	0.351	3108
Financial Services	0.948	0.825	0.668	0.738	2767
Hardware	0.914	0.678	0.348	0.460	2630
Commerce and Shopping	0.921	0.687	0.398	0.504	2527
Health Care	0.957	**0.846**	**0.707**	**0.770**	2521
Sales and Marketing	0.932	0.739	0.452	0.561	2387
Mobile	0.926	0.582	0.308	0.403	2017
Science and Engineering	0.944	0.747	0.428	0.544	1949
Data and Analytics	0.944	0.648	0.263	0.374	1595
Manufacturing	0.949	0.638	0.457	0.533	1576
Design	0.954	0.652	0.268	0.380	1305
Education	0.973	0.808	0.596	0.686	1226
Content and Publishing	0.960	0.673	0.351	0.462	1233
Real Estate	0.969	0.767	0.529	0.626	1231
Advertising	0.963	0.686	0.383	0.492	1156
Apps	0.951	0.447	0.097	0.159	1190
Transportation	0.968	0.759	0.440	0.557	1155
Consumer Electronics	0.958	0.561	0.123	0.201	1084
Professional Services	0.965	0.671	0.291	0.406	1018
Lending and Investments	0.970	0.657	0.424	0.516	933
Community and Lifestyle	0.965	0.528	0.106	0.176	888
Food and Beverage	0.981	0.774	0.609	0.682	844
Biotechnology	0.981	0.763	0.570	0.653	766
Travel and Tourism	0.982	0.801	0.508	0.622	723
Energy	0.983	0.786	0.581	0.668	754
Privacy and Security	0.980	0.748	0.362	0.488	666
Sports	0.983	0.749	0.438	0.553	607
Video	0.982	0.662	0.385	0.487	563
Natural Resources	0.985	0.733	0.525	0.612	579
Consumer Goods	0.980	0.648	0.268	0.379	571
Sustainability	0.982	0.683	0.387	0.494	574
Artificial Intelligence	0.984	0.752	0.305	0.434	509
Clothing and Apparel	0.988	0.766	0.494	0.600	470
Payments	0.986	0.637	0.347	0.449	409
Platforms	0.985	**0.405**	**0.045**	**0.082**	375
Music and Audio	0.99	0.786	0.501	0.612	403
Gaming	0.989	0.684	0.472	0.559	358
Events	0.987	0.651	0.294	0.405	367
Messaging and Telecommunications	0.987	0.500	0.204	0.290	313
Administrative Services	0.990	0.603	0.129	0.212	272
Government and Military	0.991	0.585	0.109	0.184	220
Agriculture and Farming	**0.993**	0.730	0.401	0.517	222
Navigation and Mapping	**0.993**	0.576	0.110	0.184	173
Total Average (micro-average)	0.960	0.703	0.416	0.523	n/a

Table 4.7: SVM binary classification - lemmatization results.

Groups	Accuracy	Precision	Recall	F-measure	Samples
Software	**0.768**	0.565	0.701	0.626	6929
Internet Services	0.800	0.399	0.524	0.453	3956
Media and Entertainment	0.871	0.512	0.653	0.574	3338
Information Technology	0.853	0.416	0.459	0.437	3108
Financial Services	0.937	0.715	0.712	0.714	2767
Hardware	0.888	0.468	0.476	0.472	2630
Commerce and Shopping	0.899	0.501	0.522	0.512	2527
Health Care	0.952	**0.783**	**0.729**	**0.755**	2521
Sales and Marketing	0.916	0.559	0.559	0.559	2387
Mobile	0.904	0.407	0.416	0.412	2017
Science and Engineering	0.914	0.459	0.572	0.509	1949
Data and Analytics	0.936	0.495	0.295	0.370	1595
Manufacturing	0.923	0.424	0.638	0.509	1576
Design	0.946	0.478	0.287	0.359	1305
Education	0.966	0.695	0.540	0.608	1226
Content and Publishing	0.948	0.461	0.376	0.414	1233
Real Estate	0.958	0.600	0.428	0.500	1231
Advertising	0.953	0.486	0.403	0.441	1156
Apps	0.938	0.279	0.193	0.229	1190
Transportation	0.957	0.565	0.318	0.407	1155
Consumer Electronics	0.949	0.332	0.176	0.230	1084
Professional Services	0.962	0.581	0.246	0.345	1018
Lending and Investments	0.962	0.497	0.510	0.504	933
Community and Lifestyle	0.958	0.237	0.083	0.123	888
Food and Beverage	0.974	0.646	0.534	0.585	844
Biotechnology	0.972	0.533	0.667	0.593	766
Travel and Tourism	0.974	0.567	0.419	0.482	723
Energy	0.970	0.495	0.511	0.503	754
Privacy and Security	0.975	0.574	0.239	0.337	666
Sports	0.974	0.433	0.186	0.260	607
Video	0.975	0.404	0.199	0.267	563
Natural Resources	0.976	0.484	0.482	0.483	579
Consumer Goods	0.972	0.302	0.163	0.212	571
Sustainability	0.974	0.405	0.321	0.358	574
Artificial Intelligence	0.979	0.410	0.081	0.135	509
Clothing and Apparel	0.981	0.473	0.300	0.367	470
Payments	0.981	0.310	0.108	0.160	409
Platforms	0.981	0.106	0.040	0.058	375
Music and Audio	0.983	0.433	0.169	0.243	403
Gaming	0.984	0.401	0.187	0.255	358
Events	0.982	0.118	0.038	0.058	367
Messaging and Telecommunications	0.985	0.173	0.045	0.071	313
Administrative Services	0.987	0.117	0.033	0.052	272
Government and Military	0.989	0.018	0.005	0.007	220
Agriculture and Farming	0.989	0.145	0.045	0.069	222
Navigation and Mapping	**0.991**	**0.000**	**0.000**	**-**	173
Total Average (micro-average)	0.948	0.505	0.476	0.490	n/a

Table 4.8: Naive Bayes binary classification - lemmatization results.

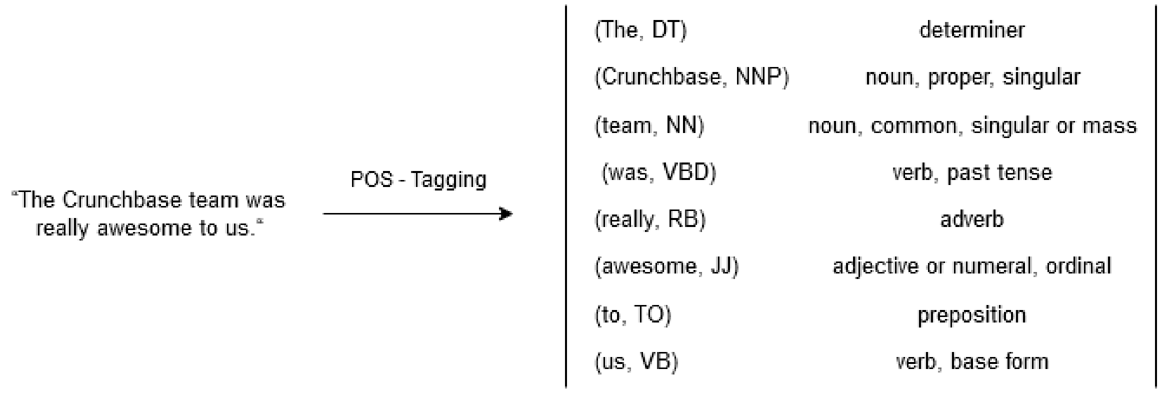

Figure 4.2: Part-of-Speech Tagging example.

POS-Tagging technique can be applied before the feeding the algorithm with input and complement the experiment in section 4.5.3. In this case, for each sentence composing each description, a Part-of-Speech pre-processing technique took place. The Part-of-Speech applied was different in a way that the output was not a tuple (word, tag) as it is the most common implementation (an example can be found in Figure 4.2), instead, a new word composed with Word + "_" + tag:

Example:

source: "streamlabs formerly known twitchalerts cuttingedge company video game industry specifically dealing video game streaming streamlabs notification crowdfunding platform streamers twitchtv strives innovate offer broadcasters best tools increase awareness brand also improve interaction viewers streamlabs offers alerts donations much tools streamers looking increase viewer engagement"

target: "streamlabs_NNS formerly_RB known_VBN twitchalerts_NNS cuttingedge_VBP company_NN video_NN game_NN industry_NN specifically_RB dealing_VBG video_JJ game_NN streaming_VBG streamlabs_JJ notification_NN crowdfunding_VBG platform_NN streamers_NNS twitchtv_VBP strives_NNS innovate_VBP offer_NN broadcasters_NNS best_VBP tools_NNS increase_VB awareness_NN brand_NN also_RB improve_VB interaction_NN viewers_NNS streamlabs_VBP offers_NNS alerts_NNS donations_NNS much_RB tools_IN streamers_NNS looking_VBG increase_NN viewer_NN engagement_NN"

Using POS-Tagging combined with the SVM and TF-IDF does not represent a major improvement when compared with the previous experiments. The results in Table 4.9 still do not overcome the baseline from the initial experiment.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.804**	0.676	0.558	0.612	6929
Internet Services	0.856	0.583	0.306	0.402	3956
Media and Entertainment	0.903	0.703	0.478	0.569	3338
Information Technology	0.883	0.560	0.274	0.368	3108
Financial Services	0.948	0.830	0.665	0.739	2767
Hardware	0.913	0.665	0.352	0.460	2630
Commerce and Shopping	0.919	0.671	0.399	0.500	2527
Health Care	0.956	**0.838**	**0.697**	**0.761**	2521
Sales and Marketing	0.931	0.724	0.448	0.553	2387
Mobile	0.927	0.583	0.327	0.419	2017
Science and Engineering	0.943	0.735	0.422	0.536	1949
Data and Analytics	0.944	0.641	0.280	0.389	1595
Manufacturing	0.951	0.650	0.469	0.545	1576
Design	0.954	0.649	0.272	0.383	1305
Education	0.972	0.796	0.582	0.672	1226
Content and Publishing	0.959	0.653	0.345	0.452	1233
Real Estate	0.968	0.760	0.504	0.606	1231
Advertising	0.963	0.666	0.382	0.486	1156
Apps	0.951	**0.441**	0.109	0.175	1190
Transportation	0.967	0.753	0.422	0.541	1155
Consumer Electronics	0.958	0.550	0.121	0.198	1084
Professional Services	0.965	0.681	0.286	0.403	1018
Lending and Investments	0.971	0.659	0.436	0.525	933
Community and Lifestyle	0.965	0.560	0.110	0.184	888
Food and Beverage	0.981	0.778	0.597	0.676	844
Biotechnology	0.981	0.758	0.557	0.643	766
Travel and Tourism	0.982	0.817	0.505	0.624	723
Energy	0.982	0.788	0.564	0.657	754
Privacy and Security	0.979	0.732	0.357	0.480	666
Sports	0.982	0.734	0.423	0.537	607
Video	0.981	0.637	0.377	0.473	563
Natural Resources	0.984	0.738	0.509	0.603	579
Consumer Goods	0.980	0.645	0.261	0.372	571
Sustainability	0.982	0.679	0.376	0.484	574
Artificial Intelligence	0.984	0.726	0.312	0.437	509
Clothing and Apparel	0.988	0.786	0.468	0.587	470
Payments	0.987	0.654	0.379	0.480	409
Platforms	0.985	0.451	**0.061**	**0.108**	375
Music and Audio	0.990	0.790	0.476	0.594	403
Gaming	0.989	0.675	0.441	0.534	358
Events	0.987	0.649	0.272	0.384	367
Messaging and Telecommunications	0.988	0.545	0.214	0.307	313
Administrative Services	0.990	0.600	0.121	0.202	272
Government and Military	0.991	0.488	0.091	0.153	220
Agriculture and Farming	**0.993**	0.755	0.374	0.500	222
Navigation and Mapping	0.993	0.533	0.139	0.220	173
Total Average (micro-average)	0.960	0.695	0.417	0.521	n/a

Table 4.9: SVM binary classification - POS-Tagging results.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.773**	0.576	0.679	0.624	6929
Internet Services	0.813	0.418	0.456	0.436	3956
Media and Entertainment	0.879	0.543	0.592	0.566	3338
Information Technology	0.867	0.460	0.398	0.427	3108
Financial Services	0.941	0.766	0.671	0.716	2767
Hardware	0.898	0.517	0.402	0.453	2630
Commerce and Shopping	0.908	0.555	0.442	0.492	2527
Health Care	0.951	**0.800**	**0.687**	**0.739**	2521
Sales and Marketing	0.922	0.616	0.474	0.536	2387
Mobile	0.914	0.450	0.309	0.367	2017
Science and Engineering	0.922	0.501	0.507	0.504	1949
Data and Analytics	0.939	0.569	0.199	0.295	1595
Manufacturing	0.932	0.464	0.551	0.504	1576
Design	0.948	0.499	0.195	0.281	1305
Education	0.964	0.723	0.434	0.542	1226
Content and Publishing	0.951	0.509	0.242	0.328	1233
Real Estate	0.957	0.633	0.322	0.426	1231
Advertising	0.954	0.514	0.263	0.348	1156
Apps	0.946	0.286	0.086	0.132	1190
Transportation	0.956	0.571	0.198	0.294	1155
Consumer Electronics	0.953	0.337	0.093	0.146	1084
Professional Services	0.961	0.560	0.161	0.250	1018
Lending and Investments	0.965	0.539	0.374	0.441	933
Community and Lifestyle	0.961	0.211	0.035	0.060	888
Food and Beverage	0.973	0.659	0.393	0.493	844
Biotechnology	0.976	0.607	0.591	0.599	766
Travel and Tourism	0.973	0.563	0.260	0.356	723
Energy	0.973	0.584	0.359	0.445	754
Privacy and Security	0.974	0.512	0.129	0.206	666
Sports	0.974	0.336	0.082	0.132	607
Video	0.975	0.292	0.071	0.114	563
Natural Resources	0.979	0.570	0.352	0.435	579
Consumer Goods	0.973	0.220	0.065	0.100	571
Sustainability	0.976	0.451	0.169	0.246	574
Artificial Intelligence	0.978	0.217	0.026	0.046	509
Clothing and Apparel	0.980	0.424	0.149	0.220	470
Payments	0.982	0.198	0.042	0.069	409
Platforms	0.982	0.056	0.013	0.022	375
Music and Audio	0.982	0.247	0.050	0.083	403
Gaming	0.984	0.257	0.053	0.088	358
Events	0.982	0.034	0.008	0.013	367
Messaging and Telecommunications	0.986	0.062	0.010	0.017	313
Administrative Services	0.987	0.061	0.011	0.019	272
Government and Military	0.990	0.023	0.005	0.008	220
Agriculture and Farming	0.990	0.067	0.014	0.022	222
Navigation and Mapping	**0.992**	**0.000**	**0.000**	**-**	173
Total Average (micro-average)	0.951	0.543	0.399	0.460	n/a

Table 4.10: Naive Bayes binary classification - POS-Tagging results.

For Naive Bayes implementation using POS-Tagging the results in Table 4.10 show similar accuracy and precision scores and a small decrease in recall and F-measure as an opposite to the Lemmatization experiment. However, once again, Health Care is the most dominant Group for precision, recall and F-measure and Navigation and Mapping in an opposite way, the worst.

4.5.6 Bigrams

In Llan (2003) it is shown that when using bigrams researchers can see an improvement in Text Classification algorithms. Here, the authors used it combined with TF-IDF and SVM as the experiment in section 4.5.3 and it shows an improvement over the unigrams approach while also showing that unigrams together with bigrams is the best approach to follow. Bigrams, consist in splitting a sentence using a window with min_lenght = 1 and max_lenght = 2.

Example:

source: “music industry progression platform empowers artists find success efficient practices models strategies”

target: [’artists’, ’artists find’, ’efficient’, ’efficient practices’, ’empowers’, ’empowers artists’, ’find’, ’find success’, ’industry’, ’industry progression’, ’models’, ’models strategies’, ’music’, ’music industry’, ’platform’, ’platform empowers’, ’practices’, ’practices models’, ’progression’, ’progression platform’, ’strategies’, ’success’, ’success efficient’]

Table 4.11 presents the results for the SVM implementation using Bigrams. The results are very similar to what was produced by lemmatization approach in section 4.5.3. Once again, does not improve the overall metrics result by a large margin.

Naive Bayes results for Bigrams can be found in Table 4.12. The results are very close of what can be found in the previous experiments, once again not improving the overall metrics and maintaining the same Groups with the highest and lowest values.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.804**	0.676	0.561	0.613	6929
Internet Services	0.853	0.567	0.310	0.401	3956
Media and Entertainment	0.903	0.703	0.474	0.566	3338
Information Technology	0.884	0.571	0.284	0.379	3108
Financial Services	0.950	0.841	0.673	0.748	2767
Hardware	0.914	0.672	0.365	0.473	2630
Commerce and Shopping	0.921	0.684	0.400	0.505	2527
Health Care	0.956	**0.842**	**0.691**	**0.759**	2521
Sales and Marketing	0.932	0.734	0.449	0.557	2387
Mobile	0.927	0.594	0.320	0.416	2017
Science and Engineering	0.944	0.738	0.430	0.543	1949
Data and Analytics	0.944	0.635	0.271	0.380	1595
Manufacturing	0.951	0.662	0.456	0.540	1576
Design	0.955	0.657	0.288	0.401	1305
Education	0.973	0.819	0.567	0.670	1226
Content and Publishing	0.959	0.665	0.341	0.450	1233
Real Estate	0.970	0.793	0.516	0.625	1231
Advertising	0.963	0.685	0.388	0.496	1156
Apps	0.952	**0.467**	0.113	0.181	1190
Transportation	0.967	0.764	0.416	0.538	1155
Consumer Electronics	0.958	0.589	0.122	0.202	1084
Professional Services	0.966	0.681	0.294	0.410	1018
Lending and Investments	0.971	0.678	0.432	0.528	933
Community and Lifestyle	0.965	0.563	0.110	0.185	888
Food and Beverage	0.982	0.801	0.623	0.701	844
Biotechnology	0.981	0.787	0.542	0.642	766
Travel and Tourism	0.982	0.824	0.494	0.618	723
Energy	0.983	0.807	0.560	0.661	754
Privacy and Security	0.980	0.760	0.347	0.476	666
Sports	0.983	0.776	0.422	0.546	607
Video	0.981	0.654	0.355	0.460	563
Natural Resources	0.984	0.746	0.501	0.599	579
Consumer Goods	0.980	0.670	0.263	0.377	571
Sustainability	0.981	0.680	0.362	0.473	574
Artificial Intelligence	0.983	0.747	0.279	0.406	509
Clothing and Apparel	0.988	0.778	0.470	0.586	470
Payments	0.987	0.673	0.362	0.471	409
Platforms	0.985	0.490	**0.064**	**0.113**	375
Music and Audio	0.990	0.795	0.471	0.592	403
Gaming	0.989	0.674	0.444	0.535	358
Events	0.988	0.754	0.292	0.420	367
Messaging and Telecommunications	0.988	0.522	0.188	0.277	313
Administrative Services	0.990	0.660	0.121	0.205	272
Government and Military	0.992	0.615	0.109	0.185	220
Agriculture and Farming	0.993	0.752	0.342	0.471	222
Navigation and Mapping	**0.994**	0.667	0.139	0.230	173
Total Average (micro-average)	0.960	0.703	0.417	0.524	n/a

Table 4.11: SVM binary classification - bigram results.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.769**	0.569	0.690	0.623	6929
Internet Services	0.805	0.407	0.505	0.450	3956
Media and Entertainment	0.874	0.523	0.635	0.574	3338
Information Technology	0.856	0.425	0.443	0.434	3108
Financial Services	0.939	0.730	0.705	0.718	2767
Hardware	0.891	0.479	0.454	0.466	2630
Commerce and Shopping	0.901	0.512	0.505	0.509	2527
Health Care	0.952	**0.785**	**0.719**	**0.750**	2521
Sales and Marketing	0.916	0.563	0.525	0.543	2387
Mobile	0.908	0.424	0.394	0.409	2017
Science and Engineering	0.916	0.467	0.556	0.508	1949
Data and Analytics	0.937	0.516	0.270	0.355	1595
Manufacturing	0.925	0.433	0.612	0.507	1576
Design	0.948	0.503	0.268	0.350	1305
Education	0.967	0.721	0.521	0.605	1226
Content and Publishing	0.949	0.471	0.350	0.401	1233
Real Estate	0.958	0.615	0.405	0.489	1231
Advertising	0.954	0.506	0.346	0.411	1156
Apps	0.940	0.269	0.157	0.198	1190
Transportation	0.957	0.573	0.290	0.385	1155
Consumer Electronics	0.950	0.347	0.161	0.220	1084
Professional Services	0.962	0.593	0.226	0.327	1018
Lending and Investments	0.963	0.510	0.483	0.496	933
Community and Lifestyle	0.959	0.228	0.069	0.106	888
Food and Beverage	0.975	0.657	0.518	0.579	844
Biotechnology	0.973	0.549	0.655	0.598	766
Travel and Tourism	0.974	0.573	0.376	0.454	723
Energy	0.971	0.527	0.485	0.506	754
Privacy and Security	0.975	0.599	0.218	0.319	666
Sports	0.975	0.440	0.157	0.231	607
Video	0.976	0.398	0.167	0.235	563
Natural Resources	0.977	0.510	0.463	0.485	579
Consumer Goods	0.973	0.299	0.142	0.192	571
Sustainability	0.975	0.421	0.293	0.345	574
Artificial Intelligence	0.979	0.390	0.059	0.102	509
Clothing and Apparel	0.981	0.496	0.266	0.346	470
Payments	0.982	0.303	0.088	0.136	409
Platforms	0.981	0.110	0.035	0.053	375
Music and Audio	0.983	0.435	0.141	0.213	403
Gaming	0.985	0.421	0.156	0.228	358
Events	0.982	0.087	0.025	0.038	367
Messaging and Telecommunications	0.986	0.180	0.035	0.059	313
Administrative Services	0.987	0.119	0.026	0.042	272
Government and Military	0.990	0.024	0.005	0.008	220
Agriculture and Farming	0.990	0.180	0.050	0.078	222
Navigation and Mapping	**0.991**	**0.000**	**0.000**	**-**	173
Total Average (micro-average)	0.949	0.517	0.456	0.485	n/a

Table 4.12: Naive Bayes binary classification - bigram results.

4.5.7 Part-of-speech tagging and bigrams

At this moment, the experiments did not manage to improve the results in experiment section 4.5.3. However, some of the experiments can be bound together to achieve better results. In Smith and M. Lee (2012) the authors combine POS-Tagging and Bigram approaches together. In this experiment we will combine the experiments in section 4.5.5 and 4.5.6 to obtain an improvement from the base experiment.

For this new experiment to work, it needs to be combined in a specific way. It is not possible to tag a sentence once it is splitted into Unigrams + Bigrams, for that, the POS-Tagging has priority in this process and only once it is finished it is applied the Bigram splitting.

Example:

source: “noogacom offers website enables users find news articles categorized according government business lifestyle entertainment opinion outdoor mainly provides news chattanooga tennessee”

target: [’according_vbg’, ’according_vbg government_nn’, ’articles_nns’, ’articles_nns categorized_vbn’, ’business_nn’, ’business_nn lifestyle_vbd’, ’categorized_vbn’, ’categorized_vbn according_vbg’, ’chattanooga_nn’, ’chattanooga_nn tennessee_nn’, ’enables_nns’, ’enables_nns users_nns’, ’entertainment_nn’, ’entertainment_nn opinion_nn’, ’find_vbp’, ’find_vbp news_nn’, ’government_nn’, ’government_nn business_nn’, ’lifestyle_vbd’, ’lifestyle_vbd entertainment_nn’, ’mainly_rb’, ’mainly_rb provides_vbz’, ’news_nn’, ’news_nn articles_nns’, ’news_nn chattanooga_nn’, ’noogacom_nn’, ’noogacom_nn offers_vbz’, ’offers_vbz’, ’offers_vbz website_jj’, ’opinion_nn’, ’opinion_nn outdoor_in’, ’outdoor_in’, ’outdoor_in mainly_rb’, ’provides_vbz’, ’provides_vbz news_nn’, ’tennessee_nn’, ’users_nns’, ’users_nns find_vbp’, ’website_jj’, ’website_jj enables_nns’]

Table 4.13 presents the results for combining the experiments in section 4.5.5 and 4.5.6. These results still do not represent an improvement over the initial experiment for the given data when considering the global performance measures.

Table 4.14 presents the results for the last experiment using a Naive Bayes classifier combined with POS-Tagging and bigrams. The results did not manage to improve any of the overall metrics and kept the same Group performance pattern as before.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.803**	0.675	0.558	0.611	6929
Internet Services	0.851	0.553	0.305	0.393	3956
Media and Entertainment	0.902	0.701	0.463	0.558	3338
Information Technology	0.883	0.561	0.280	0.373	3108
Financial Services	0.949	0.845	0.655	0.738	2767
Hardware	0.913	0.671	0.346	0.456	2630
Commerce and Shopping	0.920	0.681	0.392	0.498	2527
Health Care	0.955	**0.845**	**0.673**	**0.749**	2521
Sales and Marketing	0.931	0.729	0.442	0.550	2387
Mobile	0.927	0.596	0.313	0.410	2017
Science and Engineering	0.943	0.738	0.408	0.525	1949
Data and Analytics	0.943	0.638	0.260	0.370	1595
Manufacturing	0.952	0.675	0.447	0.538	1576
Design	0.954	0.638	0.254	0.364	1305
Education	0.972	0.816	0.561	0.665	1226
Content and Publishing	0.959	0.667	0.324	0.436	1233
Real Estate	0.968	0.780	0.489	0.601	1231
Advertising	0.963	0.687	0.376	0.486	1156
Apps	0.952	**0.465**	0.107	0.174	1190
Transportation	0.966	0.766	0.394	0.520	1155
Consumer Electronics	0.958	0.582	0.108	0.182	1084
Professional Services	0.966	0.715	0.288	0.410	1018
Lending and Investments	0.971	0.684	0.434	0.531	933
Community and Lifestyle	0.965	0.562	0.113	0.188	888
Food and Beverage	0.981	0.801	0.586	0.677	844
Biotechnology	0.981	0.770	0.529	0.627	766
Travel and Tourism	0.982	0.816	0.484	0.608	723
Energy	0.982	0.811	0.541	0.649	754
Privacy and Security	0.979	0.744	0.332	0.459	666
Sports	0.983	0.774	0.407	0.533	607
Video	0.981	0.671	0.348	0.458	563
Natural Resources	0.985	0.760	0.487	0.594	579
Consumer Goods	0.980	0.681	0.254	0.370	571
Sustainability	0.982	0.708	0.359	0.476	574
Artificial Intelligence	0.983	0.718	0.265	0.387	509
Clothing and Apparel	0.987	0.794	0.451	0.575	470
Payments	0.987	0.678	0.335	0.448	409
Platforms	0.985	0.476	**0.053**	**0.096**	375
Music and Audio	0.989	0.815	0.427	0.560	403
Gaming	0.989	0.664	0.402	0.501	358
Events	0.988	0.706	0.262	0.382	367
Messaging and Telecommunications	0.988	0.546	0.169	0.259	313
Administrative Services	0.990	0.659	0.107	0.184	272
Government and Military	0.992	0.647	0.100	0.173	220
Agriculture and Farming	**0.993**	0.773	0.338	0.470	222
Navigation and Mapping	0.993	0.531	0.098	0.166	173
Total Average (micro-average)	0.960	0.702	0.406	0.514	n/a

Table 4.13: SVM binary classification - POS-Tagging + bigram results.

Groups	**Accuracy**	**Precision**	**Recall**	**F-measure**	**Samples**
Software	**0.773**	0.576	0.679	0.624	6929
Internet Services	0.813	0.418	0.456	0.436	3956
Media and Entertainment	0.879	0.543	0.592	0.566	3338
Information Technology	0.867	0.460	0.398	0.427	3108
Financial Services	0.941	0.766	0.671	0.716	2767
Hardware	0.898	0.517	0.402	0.453	2630
Commerce and Shopping	0.908	0.555	0.442	0.492	2527
Health Care	0.951	**0.800**	**0.687**	**0.739**	2521
Sales and Marketing	0.922	0.616	0.474	0.536	2387
Mobile	0.914	0.450	0.309	0.367	2017
Science and Engineering	0.922	0.501	0.507	0.504	1949
Data and Analytics	0.939	0.569	0.199	0.295	1595
Manufacturing	0.932	0.464	0.551	0.504	1576
Design	0.948	0.499	0.195	0.281	1305
Education	0.964	0.723	0.434	0.542	1226
Content and Publishing	0.951	0.509	0.242	0.328	1233
Real Estate	0.957	0.633	0.322	0.426	1231
Advertising	0.954	0.514	0.263	0.348	1156
Apps	0.946	0.286	0.086	0.132	1190
Transportation	0.956	0.571	0.198	0.294	1155
Consumer Electronics	0.953	0.337	0.093	0.146	1084
Professional Services	0.961	0.560	0.161	0.250	1018
Lending and Investments	0.965	0.539	0.374	0.441	933
Community and Lifestyle	0.961	0.211	0.035	0.060	888
Food and Beverage	0.973	0.659	0.393	0.493	844
Biotechnology	0.976	0.607	0.591	0.599	766
Travel and Tourism	0.973	0.563	0.260	0.356	723
Energy	0.973	0.584	0.359	0.445	754
Privacy and Security	0.974	0.512	0.129	0.206	666
Sports	0.974	0.336	0.082	0.132	607
Video	0.975	0.292	0.071	0.114	563
Natural Resources	0.979	0.570	0.352	0.435	579
Consumer Goods	0.973	0.220	0.065	0.100	571
Sustainability	0.976	0.451	0.169	0.246	574
Artificial Intelligence	0.978	0.217	0.026	0.046	509
Clothing and Apparel	0.980	0.424	0.149	0.220	470
Payments	0.982	0.198	0.042	0.069	409
Platforms	0.982	0.056	0.013	0.022	375
Music and Audio	0.982	0.247	0.050	0.083	403
Gaming	0.984	0.257	0.053	0.088	358
Events	0.982	0.034	0.008	0.013	367
Messaging and Telecommunications	0.986	0.062	0.010	0.017	313
Administrative Services	0.987	0.061	0.011	0.019	272
Government and Military	0.990	0.023	0.005	0.008	220
Agriculture and Farming	0.990	0.067	0.014	0.022	222
Navigation and Mapping	**0.992**	**0.000**	**0.000**	**-**	173
Total Average (micro-average)	0.951	0.543	0.399	0.460	n/a

Table 4.14: Naive Bayes binary classification - POS-Tagging + bigram results.

4.5.8 Word embeddings

Word embeddings is one of the best performing techniques and it has been applied in several NLP tasks, namely entity recognition and parsing, see Bengio et al. (2003) and Mnih and Hinton (2009). Word embeddings consider both syntactic and semantic structures from words, they represent these words into vectors and therefore, close words appear closely in a vector space, this make them very useful and well performant when it comes to Text Classification tasks. In Subramani et al. (2019) several experiments took place using different methods combined with multiple vector representations, for instance GloVe (see ePennington, Socher, and Manning (2014)) which is a very common approach when implementing word embeddings.

For our word embeddings approach we used a custom implementation using Word2Vec with our own test dataset with two vector representations: one mean representation where an average function is applied to the document vectors and a TF-IDF representation, based on the work at http://nadbordrozd.github.io/blog/2016/05/20/text-classification-with-word2vec/. After this initial implementations we followed the same approach of Subramani et al. (2019) and used a pre-trained model from GloVe trying to improve the results. All the word embeddings experiments used the same SVM implementation previously reported, however, no other classifiers were tested.

		Accuracy	Precision	Recall	F-measure
Crunchbase	Mean	0.957	**0.738**	0.282	0.408
	TF-IDF	0.957	0.737	0.282	**0.409**
GloVe	Mean	0.956	0.732	0.273	0.398
	TF-IDF	0.955	0.727	0.255	0.378

Table 4.15: Word embeddings results.

Table 4.15 shows the outcome of the experiments applied to the Cruchbase dataset. When it comes to the overall performance using word embeddings it is possible to assess a big improvement when it comes to precision scores (+3%) while maintaining a good accuracy score, on pair of what we have reached so far. However, it does represent a performance drop when it comes to recall and F-measure.

4.5.9 Fuzzy Fingerprints

So far in our work the main focus has been in the pre-processing techniques while having the same SVM and Naive Bayes classifiers as the main algorithms. However, with the reports in 4.3 we can conclude that a Fuzzy Fingerprints Classifier can also have a good performance when applied to the Crunchbase dataset. For this experiment we used the same pre-processing steps as in section 4.1. For each of the groups we have created a unique Fuzzy Classifier with our own implementation based on a Pareto Rule with k=4000,

Experiments	**Accuracy**	**Precision**	**Recall**	**F-measure**
SVM				
Word Frequency	0.950	0.538	0.413	0.467
TF-IDF	0.960	0.696	**0.420**	**0.524**
TF-IDF + Stemming	0.960	0.705	0.411	0.519
TF-IDF + Lemmas	0.960	0.703	0.416	0.523
TF-IDF + POS-Tagging	0.960	0.695	0.417	0.521
TF-IDF + Bigram	0.960	0.703	0.417	**0.524**
TF-IDF + Bigram + POS-Tagging	0.960	0.702	0.406	0.514
Word embeddings (Crunchbase)	0.957	**0.738**	0.282	0.408
Word embeddings (Crunchbase) + TF-IDF	0.957	0.737	0.282	0.409
Word embeddings (GloVe)	0.956	0.732	0.273	0.398
Word embeddings (GloVe) + TF-IDF	0.955	0.727	0.255	0.378
NB				
Word Frequency	**0.951**	0.548	0.440	0.488
TF-IDF	0.948	0.705	**0.020**	**0.039**
Word Frequency + Stemming	0.949	0.517	0.456	0.485
Word Frequency + Lemmas	0.948	0.505	**0.476**	**0.490**
Word Frequency + POS-Tagging	0.951	0.543	0.399	0.460
Word Frequency + Bigram	0.949	0.517	0.456	0.485
Word Frequency + Bigram + POS-Tagging	0.951	0.543	0.399	0.460
FFP				
Pareto Rule, K=4000	0.926	0.351	**0.475**	0.404

Table 4.16: Summary of classification results.

see Batista and Carvalho (2015). We have performed two experiments with these classifiers. The first experiment was using our implementation with no TF-IDF weights using K=4000. For this experiment we achieved a micro average accuracy score of 0.926, a precision of 0.351, an average recall of **0.475** and an F-measure of 0.404. We have also tried to increase the K value and we saw no improvements on the previous results. These classifiers performed much better when compared to the base work in Batista and Carvalho (2015). It was noticeable that the best recall was also achieved using this Fuzzy Fingerprints approach, however, the remaining metrics did not perform this well by a large margin.

4.5.10 Results

When analyzing the Table 4.16 it is clear the evolution of the proposed work. Having an initial baseline of **39%** accuracy using SVM and **41%** accuracy with Multinomial Naive Bayes it was noticeable that both methods would be well suited for our work. Moving into the real classification task, it was implemented an initial approach of multiplying the amount of descriptions in the dataset for each labeled group, as it is explained in section 4.4. With this initial experiment, the SVM started to outperform Multinomial Naive Bayes at **67%** against **41%**, respectively. However, right away, is noticeable a clear improvement from the latest know experiments using a similar source of data. The second experiment consists on having one individual binary classifier for each group in the Crunchbase dataset (section 4.5.2).

The main goal for the initial experiment was to set a baseline and assess the best word weighting and algorithm combination to use further on to the next experiments. section 4.5.2 presents these experiments and results and from there it was clear that Naive Bayes doesn't work well with TF-IDF presenting the worst recall and F-measure scores among all the experiments, however, it presented the best precision score along side with SVM + TF-IDF + Stemming.

The results cannot be directly compared with Batista and Carvalho (2015) due to different test sets aswell as evaluation metrics and modeling approaches, however, in our work, the SVMs' are the best performing methods (at **96%** accuracy), this is interesting to note since in that work the SVM did not perform well. Thus, the main goal now was to achieve the best possible performance out of this model, for that, the focus was mainly on improving and implementing new pre-processing techniques.

Initially, we implemented Stemming approach (section 4.5.4). This, however, was not an improvement when compared with the baseline results, specially in recall and F-measure, both dropping its scores, however, there was an improvement when it comes to precision. In the same experiment, using Naive Bayes, we noticed a slight improvement in recall and a decrease in the remaining metrics. These results do not overcome the SVM approach, one possible explanation for these results in both scenarios is the loss of detail by using the radical of a word.

Another possible approach is Lemmatization (section 4.5.3). The results of the Lemmatization experiment do not represent a major improvement over the Stemming approach, however, it is possible to notice an improvement of recall and F-measure for the Naive Bayes implementation, representing the best recall and F-measure results for the Naive Bayes algorithm.

A common technique to improve this type of algorithms performance is to use Part-of-Speech Tagging (section 4.5.5). For this experiment the results were not as good as the literature refers, even though it outperforms the Baseline approach, it has a performance loss over the Lemmatization approach for SVM, this may be due to the similarity in the descriptions of each company in a syntactic way and therefore the Tagging does not add a major value for our work.

At this point, it was necessary to take into consideration the tokenization approach, implementing a Unigram + Bigram approach to the dataset (section 4.5.6). With the Bigram approach we achieved good results for our work, having a slight improvement in precision and matching the highest F-measure for SVM when compared to the Baseline. This is due to the fact of considering “word combinations” like “software company” being much more efficient than just considering each feature separately, therefore the results improved. Next, the Bigram approach was combined with POS-Tagging (section 4.5.7), however, it does not show a performance improvement for both SVM and Multinomial Naive Bayes approaches.

When it comes to feature representation a well known approach is word embeddings, following the latest experiments with Bigrams and pre-processing techniques we implemented a different feature representation (section 4.5.8). This technique immediately showed the best precision scores so far, by simply using a mean representation combined with a vocabulary build from our own sample of Crunchbase. However, it presented a major loss of performance when it comes to recall and F-measure when compared with the previous experiments.

As a final experiment we followed the approach of using Fuzzy Fingerprints classifiers (section 4.5.9), in here, the results were very good when compared to the previous work when it comes to the recall value. For the remaining scores these classifiers did not show an improvement over the SVM and Naive Bayes approaches.

Wrapping up, the best results overall represent a very good improvement when comparing to the known literature, these results reflect several techniques of pre-processing such as lower casing, punctuation removal, removing stopwords, unigram + Bigram tokenization, TF-IDF and applying an SVM classifier individually for each Group. With this set of data, applying the previously referred techniques, it was possible to obtain an overall micro-average of **96%** accuracy, **70%** precision, **42%** recall and **52%** F-measure. The experiments presented in this Chapter result in a conference paper for IPMU: International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, see Felgueiras, Batista, and Carvalho (2020).

4.6 Summary

This section presented the experiments made throughout our work. Starting in section 4.3 a baseline experiment took place as a starting point for the proposed work. Immediately it was clear that the SVM was the best approach, however we did not discard the Naive Bayes and have had a good performance out of it aswell. It is also in this section that the description normalization process is defined and explained in detail (section 4.1). After having decided what was the best ML algorithm to use the focus was towards the multi-class classification challenge. In section 4.5 it is where the variant for the different experiments take place. Each experiment is explained in its own section and includes a table with the results for both SVM and Naive Bayes implementations. The first multi-class classification experiment in section 4.4 already presents very interesting results outperforming the latest known studies by more than 20% for SVM and 10% for Naive Bayes. It is also described the metrics calculation that were used throughout the rest of the work. In section 4.5 it is where we have the largest amount of experiments due to the jump in performance that this method represents. At the end, there is a wrap up for the obtained results in section 4.5.10

comparing all the results for the entire experiment set.

Acknowledgements

This work was supported by national funds through FCT, Fundação para a Ciência e a Tecnologia, under project UIDB/50021/2020.

5 Conclusions and Future Work

This chapter revisits the research questions and overviews the difficulties and outcomes of the proposed work. This chapter is divided into two sections one being the conclusions that can be drawn from the developed work, and the unanswered questions that have raised during the development of this dissertation, leaving some next steps suggestions for future work.

5.1 Conclusions

Multi-class and multi-label Text Classification are demanding tasks, specially when the number of classes is high, in order to achieve the best performance when trying to classify text into multiple labels it is necessary to have a large set of data. The first challenge with this dissertation was data analysis for the Crunchbase dataset. Crunchbase holds complete information about companies in which we had to analyze what was the fields that were useful even before having the implementation for the algorithms. After the initial extraction of the dataset from the REST API some of the data was unparseable or corrupted in a way that the information provided was not reliable for our work. It was necessary to perform an initial processing step to remove unparseable data, so we took the opportunity to create a new database with only parseable and relevant data, removing all unwanted fields from the JSON objects. After a quick analysis to the data it was clear that some of the entries were not yet ready to be fed into the experiments, some of them had no description, others had no groups assigned, with this, we decided to free the database from this type of entries creating a final database already splitted into two different *train* and *test* tables. From that point, our work describes multi-class Text Classification experiments over a dataset with over 400000 companies.

Our experiments include three classification models, SVM, Naive Bayes and Fuzzy Fingerprints and they are combined with several NLP techniques in order to achieve the optimal performance for our dataset. In addition to the NLP techniques and Machine Learning models we also tried to include multiple feature extraction techniques by using word frequency and word weighting approaches. Using this tools, two major experiments took place, an initial experiment trying to build a single model that was able to predict the most probable category for a given company description and a second one using binary classification models for each of the categories considered in our work that was able to predict all the categories that labeled a company. The initial experiments showed encouraging results with accuracy scores around 40% for both SVM and Naive Bayes. At this point, we already have reached the same accuracy scores of the previous known work for our dataset.

In order to improve these results, we implemented our first experiment. In our Multi-class experiment we introduced a new classification method, Fuzzy Fingerprints. These three algorithms where tested in combination with the normalization steps, TF-IDF and a data transformation technique and showed improvements over the initial experiments for SVM and Fuzzy Fingerprints, with results of 67% accuracy, each.

However, these models are only able to predict one label for each company and our main goal was to predict all the possible categories with the best possible precision for our dataset. To achieve this we implemented a second experiment where each category has its own classifier that is able to classify if a description belongs or not to a given category and achieved much better results. This is where most of our work is implemented by using all of the previously referred classifiers, a different data approach, multiple NLP and feature extraction techniques. Our dataset is highly unbalanced with each category frequency ranging between 0.7% and 28%.

Regardless, our results reveal that the text description of a company contain enough features that allow us to predict its area of activity just by itself and label it into its corresponding category with an overall performance of 69% precision and 42% recall. Our work results in a conference paper for IPMU2020: Information Processing and Management of Uncertainty in Knowledge-Based Systems, “Creating Classification Models from Textual Descriptions of Companies Using Crunchbase”.

5.2 Future Work

From this point onward we are planning to improve out work by considering additional metrics for ranking problems such as precision@k, recall@k and f1@k, that may be suitable for measuring the multi-label performance. In addition to this, we are planning to introduce features based on named entities and introduce methods based on neural networks.

Bibliography

Arts, Media (2015). “Automatic Detection and Verification of Rumors on Twitter”. In: 2008.

Babuska, Robert (1998). *Fuzzy Modeling for Control*. 1st. USA: Kluwer Academic Publishers. ISBN: 0792381548.

Basu, A., C. Walters, and M. Shepherd (2003). “Support vector machines for text categorization”. In: *Proceedings of the 36th Annual Hawaii International Conference on System Sciences, HICSS 2003*, pp. 1–7. DOI: `10.1109/HICSS.2003.1174243`.

Batista, Fernando and Joao Paulo Carvalho (2015). “Text based classification of companies in CrunchBase”. In: *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1–7. DOI: `10.1109/FUZZ-IEEE.2015.7337892`.

Bengio, Yoshua et al. (2003). “A neural probabilistic language model”. In: *Journal of machine learning research* 3.Feb, pp. 1137–1155.

Colas, Fabrice and Pavel Brazdil (2006). “On the behavior of SVM and some older algorithms in binary text classification tasks”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4188 LNCS, pp. 45–52. ISSN: 16113349.

Cortes, Corinna (1995). *Support-Vector Networks*. Tech. rep., pp. 273–297. URL: `http://image.diku.dk/imagecanon/material/cortes%7B%5C_%7Dvapnik95.pdf`.

Czarnowski, Ireneusz and Piotr Jedrzejowicz (2015). “Intelligent Systems’2014”. In: *Advances in Intelligent Systems and Computing* 322, pp. 671–682. ISSN: 21945357. DOI: `10.1007/978-3-319-11313-5`. URL: `http://www.scopus.com/inward/record.url?eid=2-s2.0-84921341471%7B%5C&%7DpartnerID=tZOtx3y1`.

Dalal, Mita K and Mukesh A Zaveri (2011). “Automatic text classification: a technical review”. In: *International Journal of Computer Applications* 28.2, pp. 37–40.

Dilrukshi, Inoshika, Kasun De Zoysa, and Amitha Caldera (2013). “Twitter news classification using SVM”. In: *Proceedings of the 8th International Conference on Computer Science and Education, ICCSE 2013* Iccse, pp. 287–291. DOI: `10.1109/ICCSE.2013.6553926`.

Dolamic, Ljiljana and Jacques Savoy (2010). “When stopword lists make the difference”. In: *Journal of the American Society for Information Science and Technology* 61.1, pp. 200–203. DOI: `10.1002/asi.21186`.

Domingos, Pedro and Michael Pazzani (1997). *On the Optimality of the Simple Bayesian Classifier under Zero-One Loss*. Tech. rep., pp. 103–130. URL: `http://engr.case.edu/ray%7B%5C_%7Dsoumya/mlrg/optimality%7B%5C_%7Dof%7B%5C_%7Dnb.pdf`.

Felgueiras, Marco, Fernando Batista, and Joao Paulo Carvalho (2020). “Creating Classification Models from Textual Descriptions of Companies Using Crunchbase”. In: *Information Processing and Management of Uncertainty in Knowledge-Based Systems*. Ed. by Marie-Jeanne Lesot et al. Cham: Springer International Publishing, pp. 695–707. ISBN: 978-3-030-50146-4.

Harris, Zellig S (1954). “Distributional structure”. In: *Word* 10.2-3, pp. 146–162.

Homem, Nuno and Joao Paulo Carvalho (2011). “Authorship identification and author fuzzy "fingerprints"”. In: *Annual Conference of the North American Fuzzy Information Processing Society - NAFIPS*, pp. 180–185. DOI: `10.1109/NAFIPS.2011.5751998`.

Howedi, Fatma and Masnizah Mohd (2014). “Text Classification for Authorship Attribution Using Naive Bayes Classifier with Limited Training Data”. In: *Computer Engineering and Intelligent Systems* 5.4, pp. 48–56. ISSN: 2222-2855. URL: `http://iiste.org/Journals/index.php/CEIS/article/view/12132`.

Hrala, Michal and Pavel Král (2013). “Evaluation of the Document Classification Approaches”. In: *Advances in Intelligent Systems and Computing* 226, pp. 877–885. ISSN: 21945357. DOI: `10.1007/978-3-319-00969-8_86`.

Ikonomakis, M., Sotos Kotsiantis, and V. Tampakas (2005). “Text classification using machine learning techniques”. In: *WSEAS Transactions on Computers* 4.8, pp. 966–974. ISSN: 11092750.

Jindal, Nitin, Bing Liu, and South Morgan Street (2007). “Review Spam Detection”. In: pp. 1189–1190.

Joulin, Armand et al. (2016). “Bag of tricks for efficient text classification”. In: *arXiv preprint arXiv:1607.01759*.

Lee, Kathy et al. (2011). “Twitter Trending Topic Classification”. In: pp. 251–258. DOI: `10.1109/ICDMW.2011.171`.

Lilleberg, Joseph, Yun Zhu, and Yanqing Zhang (2015). “Support vector machines and Word2vec for text classification with semantic features”. In: *Proceedings of 2015 IEEE 14th International Conference on Cognitive Informatics and Cognitive Computing, ICCI*CC 2015*, pp. 136–140. DOI: `10.1109/ICCI-CC.2015.7259377`.

Liu, Yi and Yuan F. Zheng (2005). “One-against-all multi-class SVM classification using reliability measures”. In: *Proceedings of the International Joint Conference on Neural Networks* 2, pp. 849–854. DOI: `10.1109/IJCNN.2005.1555963`.

Llan, J Ames A (2003). “Using Bigrams in Text Categorization Department of Computer Science”. In: *Work*, pp. 1–10.

Loper, Edward and Steven Bird (2002). “NLTK: The Natural Language Toolkit”. In: arXiv: `0205028 [cs]`. URL: `http://arxiv.org/abs/cs/0205028`.

Màrquez, Lluís and Horacio Rodríguez (2005). “Part-of-speech tagging using decision trees”. In: pp. 25–36. DOI: `10.1007/bfb0026668`.

Mitchell, Tom M (2006). “The Discipline of Machine Learning”. In: *Machine Learning* 17.July, pp. 1–7. ISSN: 0264-0414. DOI: `10.1080/026404199365326`. arXiv: `9605103 [cs]`. URL: `http://www-cgi.cs.cmu.edu/%7B~%7Dtom/pubs/MachineLearningTR.pdf`.

Mnih, Andriy and Geoffrey E Hinton (2009). “A scalable hierarchical distributed language model”. In: *Advances in neural information processing systems*, pp. 1081–1088.

Murphy, Kevin P et al. (2006). “Naive bayes classifiers”. In: *University of British Columbia* 18, p. 60.

Murthy, Sreerama K, Simon Kasif, and Steven Salzberg (1994). “A System for Induction of Oblique Decision Trees”. In: pp. 1–32.

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan (2002). *Thumbs up? Sentiment Classification using Machine Learning Techniques*. Tech. rep. URL: `http://reviews.imdb.com/Reviews/`.

Pedregosa, F. et al. (2011). “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Peffers, Ken et al. (2007). “A design science research methodology for information systems research”. In: *Journal of management information systems* 24.3, pp. 45–77.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Plisson, Joël, Nada Lavrac, and Dunja Mladenic (2004). *A Rule based Approach to Word Lemmatization*. Tech. rep.

Pranckevicius, Tomas and Virginijus Marcinkevicius (2017). “Application of Logistic Regression with part-of-the-speech tagging for multi-class text classification”. In: *2016 IEEE 4th Workshop on Advances in Information, Electronic and Electrical Engineering, AIEEE 2016 - Proceedings*, pp. 1–5. DOI: `10.1109/AIEEE.2016.7821805`.

Quinlan, J. Ross (1986). “Induction of decision trees”. In: *Machine learning* 1.1, pp. 81–106.

Rennie, Jason D. M. and Ryan Rifkin (2001). “Improving Multiclass Text Classification with the Support Vector Machine”. In: *Massachusetts Institute of Technology AI Memo 2001-026* October 2001, pp. 1–14. URL: `http://dspace.mit.edu/handle/1721.1/7241`.

Rogati, Monica and Yiming Yang (2002). “High-performing Feature Selection for Text Classification”. In: *Proceedings of the Eleventh International Conference on Information and Knowledge Management*. CIKM ’02. McLean, Virginia, USA: ACM, pp. 659–661. ISBN: 1-58113-492-4. DOI: `10.1145/584792.584911`. URL: `http://doi.acm.org/10.1145/584792.584911`.

Rosa, Hugo, Fernando Batista, and Joao Paulo Carvalho (2014). “Twitter topic fuzzy fingerprints”. In: *IEEE International Conference on Fuzzy Systems*, pp. 776–783. ISSN: 10987584. DOI: `10.1109/FUZZ-IEEE.2014.6891781`.

Saif, Hassan et al. (2014). “On stopwords, filtering and data sparsity for sentiment analysis of twitter”. In: *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014*, pp. 810–817.

Sain, Stephan R. and V. N. Vapnik (2006). *The Nature of Statistical Learning Theory*. Vol. 38. 4. Berlin, Heidelberg: Springer-Verlag, p. 409. ISBN: 0-387-94559-8. DOI: `10.2307/1271324`.

Salton, Gerard and Christopher Buckley (1988). “Term-weighting approaches in automatic text retrieval”. In: *Information processing & management* 24.5, pp. 513–523.

Saunders, Mark, Philip Lewis, and Adrian Thornhill (2009). *Research methods for business students*. Pearson education.

Sharma, Deepika and Me Cse (2012). “Stemming algorithms: a comparative study and their analysis”. In: *International Journal of Applied Information Systems* 4.3, pp. 7–12.

Smith, Phillip and Mark Lee (2012). “Cross-discourse Development of Supervised Sentiment Analysis in the Clinical Domain”. In: July, pp. 79–83.

Subramani, Sudha et al. (2019). “Deep learning for multi-class identification from domestic violence online posts”. In: *IEEE Access* 7, pp. 46210–46224.

Sun, Aixin, Ee Peng Lim, and Ying Liu (2009). “On strategies for imbalanced text classification using SVM: A comparative study”. In: *Decision Support Systems* 48.1, pp. 191–201. ISSN: 01679236. DOI: 10.1016/j.dss.2009.07.011.

Toman, Michal, Roman Tesar, and Karel Jezek (2006). “Influence of word normalization on text classification”. In: *Proceedings of InSciT*, pp. 354–358. URL: http://www.kiv.zcu.cz/research/groups/text/publications/inscit20060710.pdf.

Webster, Jonathan J. and Chunyu Kit (1992). “Tokenization as the initial phase in NLP”. In: *Adv. Mater.* 4, p. 1106. DOI: 10.3115/992424.992434.

Wilbur, W. John and Karl Sirotkin (1992). “The automatic identification of stop words”. In: *Journal of Information Science* 18.1, pp. 45–55. ISSN: 17416485. DOI: 10.1177/016555159201800106.

Willett, Peter (2006). “The Porter stemming algorithm: Then and now”. In: *Program* 40.3, pp. 219–223. ISSN: 00330337. DOI: 10.1108/00330330610681295.

Xu, Shuo (2018). “Bayesian Naive Bayes classifiers to text classification”. In: *Journal of Information Science* 44.1, pp. 48–59.

Zhang, Dell, Xi Chen, and Wee Sun Lee (2005). “Text classification with kernels on the multinomial manifold”. In: *SIGIR 2005 - Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 266–273. DOI: 10.1145/1076034.1076081.