ALGORITHMS FOR AUTOMATIC LABELING OF TEXT DATA

Ioannis Matzakos Chorianopoulos

A Dissertation submitted to
the School of Computing Sciences of The University of East Anglia
in partial fulfilment of the requirements for the degree of
MASTER OF SCIENCE.
AUGUST, 2017

SUPERVISOR(S), MARKERS/CHECKER AND ORGANISER

The undersigned hereby certify that the markers have independently marked the dissertation entitled "**Algorithms For Automatic Labeling of Text Data**" by **Ioannis Matzakos Chorianopoulos**, and the external examiner has checked the marking, in accordance with the marking criteria and the requirements for the degree of **Master of Science**.

Supervisor:	Dr. Beatriz De La Iglesia
Markers:	Marker 1: Dr. Beatriz De La Iglesia
	Marker 2: Dr. Aristidis Nikoloulopoulos
External Examiner:	Checker/Moderator
Moderator:	Dr. Wenjia Wang

DISSERTATION INFORMATION AND STATEMENT

Dissertation Submission Date: **August, 2017**

Student:	**Ioannis Matzakos Chorianopoulos**
Title:	**Algorithms For Automatic Labeling of Text Data**
School:	**Computing Sciences**
Course:	**Knowledge Discovery and Data Mining**
Degree:	**M.Sc.**
Duration:	**2016-2017**
Organiser:	**Dr. Wenjia Wang**

Signature of Student

Abstract

Social media can be considered as a substantial source of data for different purposes and applications. Such data can be derived from Twitter and exploited for the benefit of public health by syndromic surveillance systems. Twitter data though can be tedious and time consuming to label manually, so it is important to investigate a classification approach for automatic labeling. Moreover, it is necessary to identify which instances of data can provide useful information for the purpose of syndromic surveillance. Furthermore, these instances can be classified as relevant or not. The problem that this study is focusing on, is to determine which tweets are relevant or not. One category of algorithms that can provide a solution to that problem is partially supervised classification. Therefore, the aim is to test a semi-supervised algorithm that can be used in the context of syndromic surveillance.

To achieve this goal, a data mining process was conducted. The data used is a collection of tweets about air pollution and breathing problems. A part of this dataset is labeled but the majority of it is not. So, the concept of the methodology that this study is following, is to compare a set of algorithms trained on the labeled data and chose one based on its performance to base a semi-supervised approach in order to automatically label the unlabeled data. Initially data cleansing and preprocessing was performed in order to improve data quality and make data understandable to the algorithms. Moreover, various classifiers were tested and evaluated to point out the most efficient regarding relevant tweets. Finally, the chosen algorithm is used as base for performing semi-supervised learning self training.

Different experiments were undertaken that pointed out the impact of "noise", in both approaches. In the partially supervised method, the produced models even though were improving at some point demonstrated a downfall in performance. Furthermore, this suggested that the more data, not only the more misclassified not relevant tweets but also the more correctly classified relevant tweets. Eventually with this approach overfitting occurred despite showing brief improvements initially.

Acknowledgements

I would like to thank my supervisor, Dr. Beatriz De La Iglesia, for her suggestions and constant support during this research.
I would also like to express my gratitude to my family for sponsoring my course in full, their patience and *love.*

Ioannis Matzakos Chorianopoulos
at Norwich, UK.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ATAM Ailment Topic Aspect Model. 11

DT Decision Trees. 26, 37, 38, 42, 51

DTM Document Term Matrix. 9, 19, 20, 23, 50

EM Expectation Maximization. 14, 15

FN False Negatives. 31

FP False Positives. 31

GMM Gaussian Mixture Models. 14

HMM Hidden Markov Models. 14

IG Information Gain. 10, 22, 43

IR Information Retrieval. 5

KDD Knowledge Discovery from Database. 3, 6, 8–10, 12, 16, 18, 20, 23, 28, 30, 32, 33, 35, 36, 39, 40, 43, 49–53

KNN K Nearest Neighbor. 12, 15

LDA Latent Dirichlet Allocation. 11, 23

MaxEnt Maximum Entropy. 27, 37, 38, 42, 51

MI Mutual Information. 10

ML Machine Learning. 5

MMH Maximum Marginal Hyperplane. 25

NB Naive Bayes. 12, 13, 15, 24, 29, 37–40, 42–44, 51

NLP	Natural Language Processing. 5
NNET	Neural Networks. 12
OR	Odds Ratio. 10, 23, 43
OS	Operating System. 32, 34
PSC	Partially Supervised Classification. 4, 5
ReSST	Real-time Syndromic Surveillance Team. 3
RF	Random Forests. 12, 37, 38, 40, 42, 51
SLDA	Scaled Linear Discriminant Analysis. 26, 27, 37, 38, 40, 42, 51
SVM	Support Vector Machines. 12, 14, 15, 25, 37, 38, 40, 42, 51
TF	Term Frequency. 9
TF-IDF	Term Frequency-Inverse Document Frequency. 9, 20, 21, 39, 41, 50, 51
TM	Text Mining. 5, 7, 18
TN	True Negatives. 31
TP	True Positives. 31
TSVM	Transductive Support Vector Machines. 13, 14
UEA	University of East Anglia. 32, 47
WkNN	Weighted K Nearest Neighbor. 12

Chapter 1

Introduction

Because of the constantly increasing use of **social media** applications like Twitter it is believed that they can be a massive source of information for the surveillance of health events and issues (Dredze, 2012; Chen et al., 2014). Social media is widely used by millions of people everyday who share their thoughts and opinions on topics such as pop culture, politics and arts, along with moments of their personal lives (Paul and Dredze, 2011). Many studies used data derived from social media in order to investigate or predict potential **trends** in numerous scientific areas (Paul and Dredze, 2012). But this study will focus on extracting and analyzing information regarding **public health monitoring** by using **machine learning** algorithms.

Nowadays, the internet dominates a great portion of our social lives. More and more people use social media in their everyday lives, such as Facebook, Twitter, Instagram. Their desire to communicate with close, distant friends or strangers leads a lot of them to make public posts about their ideas, concerns and their lives in general. Many tweets might contain information about health events and issues that are occurring in different regions on the planet at specific periods of time. Additionally, the contained information might refer to the author of a tweet or even another person, might have also details about specific illnesses or diseases (Lamb et al., 2013). Therefore, these kind of tweets can be considered a source for unraveling knowledge about specific issues.

The knowledge that will be obtained can be used to identify health events that might occur using a **syndromic surveillance** process. "Syndromic surveillance is the process of collecting, analysing and interpreting health-related data to provide an early warning of human or veterinary public health threats, which require public health action." (Public Health England, 2017). In other words, syndromic surveillance using social media data, such as tweets, can help identify health issues that people might be suffering from and may give insight into how many are suffering from those issues in a particular region (Paul and Dredze, 2011). That kind of information can enable health organizations and institutions to take action and prevent a disease from expanding. Considering that tweets are written by specific users and occur in specific times and areas, millions of tweets can provide valuable information about health events (Paul and Dredze, 2011). Thus, mining Twitter data for syndromic surveillance purposes can be beneficial to the world.

Due to the ability of some words or phrases to have both literal and metaphorical meaning, it is crucial to distinguish the tweets that are **relevant** and **irrelevant** to the purpose of syndromic surveillance. Additionally, further classification of the relevant tweets can be undertaken, such as discriminating those which are about an infection incident and those which declare awareness, identifying whether the tweets refer to their author or another person and whether there are tweets posted by people or news sites (Lamb et al., 2013). So, many types of classification tasks that can be useful for syndromic surveillance are sentiment analysis and opinion mining, but this project will focus on performing a relevance analysis of tweets for syndromic surveillance purposes regarding public health.

1.1 Motivation

Various applications using social media data have been already investigated in numerous research projects solved successfully different problems in the context of monitoring certain events (O'Connor et al., 2010; Sakaki et al., 2010; Culotta, 2013; Ginsberg et al., 2009; Chunara et al., 2012; Bian et al., 2012). The success of related works (Chen et al., 2014; Lamb et al., 2013; Heaivilin et al., 2011; Ginsberg et al., 2009) increased the interest of people working in public health, who's purpose is to improve its policies and actions (Dredze, 2012) by analyzing information derived from social media and in particular, Twitter. Additionally, the outcomes of research work in this field can be exploited on government applications as well (Chen et al., 2014). For example, health authorities can be, if they are not already, able to monitor public health in order to be able to take immediate action in the case of an outbreak. Furthermore, the early detection of events can lead to prevention of unwanted situations and benefit not only people and public health but also the health organizations, government authorities and companies (Bian et al., 2012). Therefore, the benefits of using and researching mining social media data for the purpose of syndromic surveillance can be found in numerous applications and especially in the context of public health.

This study can become a “brick in the wall” of a bigger project that can enable organizations such as Public Health England to be notified for a health event in a specific region and take immediate action. This particular organization has already active syndromic systems. These syndromic systems are looking for patterns over time in order to keep public health professionals up to date (Public Health England, 2017). These systems are being run by Real-time Syndromic Surveillance Team (ReSST) and they are looking to expand their reach. Thus, there are already active and running syndromic surveillance systems but it may be possible to expand and improve their functionality by considering additional data sources.

1.2 Problem Definition

Twitter data can be considered as semi-structured type of data, as it contains attributes such as username, date, time, the actual text of the tweet and optionally the location from where the tweet was published. That means that despite the unstructured character of the main element, which is text, tweets can be stored in a database. This leads to being able to apply a **Knowledge Discovery from Database (KDD)** process. So, in order to distinguish tweets with literal and metaphorical meaning, hence the relevant from the irrelevant, we need to perform a classification task for syndromic surveillance.

The labels, or in other words the classes, that the data are going to have are the following: “Relevant” and “Not Relevant”. For example, a tweet marked as “Relevant” would be “I’ve got a chest infection and my asthma has returned” and a tweet marked as “Not Relevant” would be “Smoke exposure increases asthma admission risk”. The first one, is relevant because it provides information about a person that has a certain infection and asthma. The second one, is irrelevant because it does not declare any incidents and does not say anything that can be used to detect health incidents. Additionally, another big difference that these two tweets have is the way that they were posted. The first was posted by a real person using the Twitter website and the other was tweeted automatically using a bot. Therefore, we need to discriminate these two kinds of tweets in order to be able to identify those that can actually be a part of a syndromic surveillance system for the benefit of public health.

To analayse tweets for syndromic surveillance, there are some issues that need to be considered:

1. Determining which tweets are relevant and not for the purpose of syndromic surveillance
2. Determining which tweets are posted by individuals and which are posted by news websites or organizations
3. Establish the location of tweets so that analysis can be restricted to specific geographical areas

In this study we are going to tackle the first issue, by evaluating which classification algorithms and methodologies are able to label each tweet appropriately as relevant or irrelevant.

The problem that this project is focusing on, is automatic labeling of text data and in particular twitter data. This can be defined as a classification task. That is because the collected tweets, about air pollution and breathing problems, need to be categorized as relevant or not, due to the fact that there are words and phrases that can be used both literally and metaphorically. Therefore, to perform such a task and solve this problem, there are two types of classification,

1. Supervised Classification
2. Partially Supervised Classification

The term “supervised” means that all the data in the training set have to be labeled in comparison with the term “partially supervised” which means that in a dataset there are both labeled and unlabeled data. Partially Supervised Classification (PSC)

(Scudder, 1965; Vapnik et al., 1974; Ratsaby and Venkatesh, 1995) is a category of algorithms and methods that tackle the problem of automatic labeling of data using unlabeled data in the process of classification. It is tedious to label data manually and often the datasets contain massive amounts of data (Liu et al., 2002). Therefore, partially supervised classification can provide the right set of algorithms to successfully build accurate and efficient classifiers.

The fact that social media data and in this case **Twitter data** are primarily text data, sets this task in the realm of **Text Mining (TM)**. Because of that, apart from **Data Mining** and **Machine Learning (ML)**, the incorporation of different techniques from several fields is required, such as **Natural Language Processing (NLP)**, and **Information Retrieval (IR)**. However, this kind of data can not only contain text but also numerical features. So, **Statistics** is also a field that can provide some useful techniques.

1.3 Aim And Objectives

The aim of the project is to test a semi-supervised algorithm that can be used in the context of syndromic surveillance. In this case the task is going to be achieved by identifying tweets where a person is saying that they have suffered a specific health issue. The objectives of this project will be:

1. To perform a **literature review** of partially supervised classification (PSC) as this type of algorithm will be applicable to text mining data where labels are scarce.

2. To **implement or use** an implementation of a PSC algorithm for tweet relevance classification.

3. To **fine tune** the algorithm for large/complex datasets.

4. To **analyse** a dataset and report on results.

5. To **compare** against other existing classification approaches for text mining.

1.4 Summary of Thesis

Twitter can be considered as a valuable resource for various tasks and applications in various sciences but most importantly in computer science, in the field of data science. (Lamb et al., 2013). Furthermore, this particular social network has been used in applications concerning public health and can be even more useful for syndromic surveillance (Paul and Dredze, 2011). Therefore, in this study we will be investigating the ability to classify tweets as relevant or not relevant in the context of syndromic surveillance for a specific syndrome by performing supervised or partially supervised classification.

The data that are used in this project are tweets posted in the United Kingdom from various sources and the overall topic is air pollution and breathing problems. The dataset has a labeled part and an unlabeled part. The labeled tweets will be used in supervised classification approaches but the both labeled and unlabeled will be used in the semi-supervised approaches where the unlabeled will be automatically labeled after a successful training of different classifiers.

Before proceeding with data mining process and the experiments, a number of research papers provide a global view of the related work. This literature review of supervised, partially supervised classification approaches and syndromic surveillance, including the use of Twitter for public health applications provide the ideas needed to tackle this particular task. So, after that several experiments in the previously mentioned approaches will be conducted and the results will be interpreted.

In the implementation and experimental phase, it is necessary to conduct a KDD process. The main task is to study how unlabeled data can be used to improve classification performance. Firstly, a supervised approach will be implemented in order to study the performance of different well known classification algorithms on labeled data. Secondly, a partially supervised approach will take place. Finally, the methodology and all the results along with the issues that occurred will be not only described and explained but also discussed and evaluated.

Chapter 2

Related Work

2.1 Background Knowledge

Since the problem, that this study is focusing on is dealing with text data derived from Twitter, it can be defined as a **Text Mining (TM)** task. TM is about discovering knowledge patterns in text data (Witten et al., 2016). It is the process of extracting potentially valuable information from collections of documents by recognizing patterns and producing knowledge about a particular subject (Feldman and Sanger, 2007). Moreover, as text being the main form of recording information an issue arises regarding its accessibility by computers. So, TM includes also the process of making text written in natural language understandable by computers, in order to process it and extract the desired information (Witten et al., 2016). Therefore, TM can be defined as *"the process of analyzing text to extract information that is useful for particular purposes"* (Witten et al., 2004, 1999).

In the data mining world there are three different types of data, which are the following:

1. Structured data
2. Semi-structured data
3. Unstructured data

The first category of structured data is the most common one, which includes either numerical or categorical data which are stored in relational database. The category of unstructured data usually defined as information recorded in natural language, which cannot be stored in a database. Nevertheless, semi-structured data contain elements from both previous categories. For example, Twitter data even though the main feature is the text of each tweet there are features such as ID, date, time and more. Therefore semi-structured data can be stored in a database despite the fact text is the main attribute because other numerical or categorical variables are accompanying it.

Regarding the input of the KDD process, text is unstructured but is the main form of recording information formally (Witten et al., 2016). However, in this case, the data are defined as **semi-structured** because despite having text in natural language, there are some features available such as user id, data, time and source, which allow the data to be stored in a database. Therefore, each instance in the database refers to a document, a tweet.

As in conventional data mining tasks, data stored in a database can be classified, so can text data. Each document has a certain topic or can be labeled in a specific way, according to the problem that needs to be solved. The process of assigning labels is called **document classification**, where each document is an instance of a dataset and the words that each document contains are the features to be analyzed (Witten et al., 2016). There are several techniques involved in this kind of classification for the different stages of the KDD process, which contribute in extracting the desired information.

2.2 Data Preprocessing

Before proceeding with classification, it is necessary for the computer to understand textual data. One of the most popular methods to transform a text document in a form understandable by computers is the **bag of words** approach. This particular method is widely used for the purpose of text mining. The logic of the bag of words approach is to create a set from all the words contained in a document without keeping duplicates, in the case of a word being mentioned multiple times in the document. For that reason, the frequency of each word is being recorded, in terms of how many time a word exists in the document.

Bag of words approach requires ways to represent text in a structured manner in order to analyze text data. The main way to achieve that is to create a document

corpus. A corpus contains all the words for every document in the dataset and it is the main form of the data during the KDD process. Afterwards, the corpus gets the form of a Document Term Matrix (DTM) in which each row represents a document and each column has two elements the word and its weight. There are some **text representation** measures in order to compute these weights. Firstly, there is **Term Frequency (TF)**, which for each document the frequency of each word in it is being recorded. Secondly, **Term Frequency-Inverse Document Frequency (TF-IDF)** computes the relative frequency of each term in every document in comparison with the ratio of that word over the corpus (Luhn, 1957; Sparck Jones, 1972).

Since a corpus is constructed, data need to be cleaned in order to improve **data quality**. Part of the preprocessing stage are the techniques of **stop word removal** and **stemming**. By implementing a bag of words approach the resulting set of features is massive. A lot of words are extremely common and might not be helpful and instead of adding value to the data, the quality is decreasing. For example, words such as articles like "the" or conjunctions like "and" and more, are not contributing in data quality. So these kind of words should be removed, but the set of stop words for removal differs according to the problem. Furthermore, another technique is stemming, which finds all the words that have the same root and keeps in the corpus only the radical of those words. This can be considered as another way to reduce features but its aim is to improve data quality by keeping in the corpus all the parts that can provide some kind of a knowledge pattern or information.

After having transformed the text data in a computer understandable form, it is necessary to find ways to identify the most important and meaningful words. In other words it is needed to find the best features. This process is called **feature selection** or **feature extraction** (Tan et al., 2005). Moreover, there is sometimes the issue of high dimensionality in document classification (Zhu and Wong, 2014). Considering how many words might exist in a collection of documents, even in this case with a collection of tweets, the features that an algorithm is called to process are numerous. This makes it more difficult to achieve an accurate classification. Therefore, the best features need to be selected, in order to improve data quality (Aggarwal, 2015).

There are several feature selection techniques. Four of the most popular feature selection algorithms are the following:

1. Chi-square

2. Information Gain (IG)

3. Odds Ratio (OR)

4. Mutual Information (MI)

Each one of those algorithms are computing the importance of each feature in a dataset in a different way. Hence, for each attribute a weight is calculated in order to determine the subset of the most important features for the purpose of optimizing the results of the KDD process. Although, these algorithms do not have the same performance quality (Zhu and Wong, 2014). This means that according to different factors, such as the available data, the nature of the problem and more, the results that each one provides are not beneficial to the overall result. Therefore, it might be necessary to use additional techniques or replace them with more optimal ones.

Another widely known feature selection method for text mining is n-grams, which tokenize text according to the preferred number of words (Damerau, 1971). In other words, the words that are featured in a document are being grouped in two or more words in order to form phrases. This method aims to exploit the association that some words might have together, because otherwise might be redundant and add noise instead of driving towards the optimal set of features. Furthermore, another method is speech tagging, which enables the identification of important words or phrases by recognizing what part of speech is each word. Additionally, speech templates is technique that identifies sentence structures in a document in the effort of exploiting the meaning of sentences or phrases, because otherwise each word by itself might not have to offer any value to the feature set (DeRose, 1988; Church, 1988; DeRose, 1990). Those techniques are aiming to select the optimal set of features in order to improve classification accuracy, precision and recall.

Despite all the different techniques that can be applied in order to reduce the feature set and work only with the most important attributes, sometimes the available features are not enough either in terms of numbers or in terms of importance. In that case it is necessary to construct new features from the existing ones by taking into consideration different aspects of the available data. For example, some features might associate in a certain way or multiple attributes can create a new one. In this way new features are being created in the effort of improving data quality and therefore classification accuracy. This process is called **feature engineering** or **feature construction**, which Tan et al. (2005) explain further in their book.

Regarding feature engineering, there are several examples in different studies. The most interesting as well as effective was word classes, which is a method that categorizes each word in a corpus (Lamb et al., 2013). Moreover, a similar feature construction method is topic modeling using algorithms, such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003) or Gibbs sampling algorithms (Geman and Geman, 1984). Additionally, topic modeling using LDA allocates topics to each word of the corpus and then for each document in the dataset computes the probabilities of each topic belonging to each document (Blei et al., 2003; Paul and Dredze, 2011). Furthermore, Paul and Dredze (2012) proposed another algorithm for topic modeling called Ailment Topic Aspect Model (ATAM), based on LDA and Gibbs sampling, which models how a part of text is expressed by its author. Finally, another interesting idea was the use of XML tags to identify the weight or style of a word in order to determine its importance (Fourli-Kartsouni et al., 2007).

2.3 Supervised Classification

Given that the text data from the documents have been converted into data structure understandable by a machine and the optimal features are selected or created, it is time to proceed in training classification models. A classification problem can be defined as "learning the structure of a data set of examples, already partitioned into groups, that are referred to as categories or classes" (Aggarwal, 2015). Therefore, it is the process of distinguishing the different instances of the data into the predefined categories or classes.

A classification task is always approached in two phases, as the KDD process requires. Firstly, there is the training phase. In this stage a model is being created from the majority of the data, which is called training set. Secondly, there is the testing phase. In this phase the rest of the data, which are not included in the training set, are testing the model (Aggarwal, 2015).

There are many different algorithms that can be used to perform supervised classification. The most popular ones are the following:

1. Naive Bayes (Naive Bayes (NB)) (Maron, 1961)
2. K Nearest Neighbor (K Nearest Neighbor (KNN)) (Fix and Hodges Jr, 1951)
3. Support Vector Machines (Support Vector Machines (SVM)) (Vapnik and Chervonenkis, 1964)
4. Random Forests (Random Forests (RF)) (Ho, 1995)
5. Decision Trees (Quinlan, 1986)
6. Neural Networks (NNET) (McCulloch and Pitts, 1943)
7. AdaBoost (Freund and Schapire, 1995)

Each one has totally different way of training a model but the aim of all is to train an efficient model.

There is a wide selection of algorithms used in different studies. For example, NB, KNN and AdaBoost were used to train models for text classification (Zhu and Wong, 2014). Moreover, in a multi-label classification task, another study used Weighted K Nearest Neighbor (WkNN) along with SVM and RF, in order to tackle incomplete labeling or incorrectly assigned labels (Kolesov et al., 2014).

2.4 Partially Supervised Classification

There are several different approaches over the years for partially supervised classification in a text mining context where labeling issues are present. Automatic labeling of text is also gaining research interest (Iwamura et al., 2013). Moreover, Liu et al. (2002) state that training classifiers for text classification tasks using both labeled and unlabeled data improves the accuracy of classifiers and presents more efficient results. Therefore, partially supervised classification is a category of algorithms and methods in the context of using unlabeled data. This particular type of classification has different types of algorithms. These "families" of algorithms are the following:

1. Generic Meta-algorithms
2. Transductive Support Vector Machines (TSVM) (Joachims, 1999)
3. Generative models (Shahshahani and Landgrebe, 1994)
4. Graph-based algorithms

Generic Meta-algorithms perform model training using well known supervised classification algorithms, such as NB and improve classifier performance by using unlabeled data as well (Aggarwal, 2015). Two different approaches exist in this kind of semi-supervised learning, which are the following:

1. Self-Training algorithms (Nagy and Shelton, 1966)
2. Co-Training algorithms (Blum and Mitchell, 1998)

Self-training algorithms can be applied using any supervised learning algorithm in order to pre-train the models. The next step is to rank the unlabeled instances in a descending order using a **confidence measure**. The ranking process is being carried out with the purpose to include the most accurate observations in the training set. So, an overall model is produced which can be used to predict new labels for the test data, just like in a supervised approach. Moreover, **Co-Training** algorithms the attributes that consist the feature set are divided into two groups. For each group a separate classifier is trained without the one having anything to do with the other at all during the prediction stage. However they do interact when their respective training sets need to be assembled (Aggarwal, 2015).

TSVM is the semi-supervised version of the classic SVM, which tries to exploit the unlabeled data to improve training efficiency. This semi-supervised version of SVMs is based on an assumption. TSVMs assume that the unlabeled instances have low variability in the more dense areas of the dataset. In this method, the unlabeled data are being labeled by assigning the different classes to instances that their predictions are potentially the most accurate ones. This happens in an iterative manner. For each iteration the assigned label of an observation changes to the opposite label in order for the soft margin of the algorithm to be optimized (Aggarwal, 2015).

Generative models assume that labeled and unlabeled data come from the same model and different algorithms are used to compute the missing labels of the unlabeled observations. Algorithms that belong in this algorithmic "family", which also deal with the unlabeled instances are the following:

1. Expectation Maximization (Expectation Maximization (EM)) (Dempster et al., 1977)
2. Spy-EM (Liu et al., 2002)
3. Hidden Markov Models (HMM) (Baum and Petrie, 1966)
4. Gaussian Mixture Models (GMM) (Shahshahani and Landgrebe, 1994)

Graph based algorithms are learning the data by learning tree-like representation by looking at different nodes (Schwenker and Trentin, 2014). These tree-like graphs are called similarity graphs. Initially a similarity graph is constructed with each node being assigned to an instance of the dataset. Afterwards, the edges of this tree structure have weights which represent the similarity between two instances. Therefore the largest the weights on the graph the greater the similarity (Aggarwal, 2015).

The general concept of performing a partially supervised classification task is to have both labeled and unlabeled data, train a model using the labeled ones and then classify the unlabeled data using that model, but in an iterative manner. In this way, the newly unlabeled data would be each time a part of a new model. Several notable approaches are presented in this section. For example, EM along with classic classifiers, such as NB is used (Liu et al., 2002). Furthermore, Baker et al. (2016) suggested a robust method for text classification of partially labeled datasets. They developed a methodology that learns distributed semantic representations at word, sentence and document level, which can be used as a feature selection technique so a model can be trained using SVM.

Liu et al. (2002) proposed a solution to a partially supervised classification task which they framed as a constraint optimization problem. They identified classic classification algorithms that can handle text mining tasks such as NB, KNN, SVM, Rocchio, but these algorithms require labeled data in order to produce the appropriate results and perform classification tasks successfully. However, NB was used because is more efficient along with EM (Liu et al., 2002). In this approach the input was two datasets of documents one containing documents labeled as positive and the other with unlabeled documents with potentially both positive and negative data. The goal was to identify the positive ones in the unlabeled set. Moreover, in order to improve the accuracy, they used EM by including also spy documents, which are a portion of positive documents in the unlabeled dataset. Classification was performed using NB and EM.

Iwamura et al. (2013) in the effort of classifying characters, proposed a method that is based on semi-supervised learning approach of self-training or self-corrective recognition. They performed an evaluation of segmentation, automatic labeling and a combination of the two previous methods as a unified system. This particular method is the first that tackles successfully an automatic labeling task (Iwamura et al., 2013).

Chapter 3

Methodology & Implementation

3.1 The Data

The data that will be used in this KDD process are Twitter data regarding air pollution and breathing problems. These were collected over the course of a specific period of time as the tweets were posted in real time. Then a number of these tweets were labeled manually. So, the dataset consists of two parts, the labeled set and the unlabeled set. The labeled set will be used to train classifiers based on different algorithms along with the unlabeled set. Then the models of those classifiers that will be produced, will be evaluated on the test set using several performance metrics.

The data processed, in the first phase (supervised), is the labeled air pollution dataset derived from twitter. This particular dataset consists of 3500 tweets, 480 of which are relevant and 2120 are irrelevant. The features that consist this dataset are tweet ID, source, date created (created_at), relevance (Relevant), text, coordinates, user id, time zone and place. The dataset has been split into a training set and a test set. In both sets the sampling is random in order to select different set of tweets. However, random sampling was not enough for the training set because the original dataset is unbalanced and the majority of the tweets are labeled as not relevant which is the opposite class from the one we are interested in. It was necessary to reduce the irrelevant tweets, in order to force the algorithm to learn the relevant instances better. Balancing the data and having the same amount of relevant instances as irrelevant may benefit the classifier. Moreover, several balancing approaches were tested. Nevertheless, balancing has its drawbacks because reduces the size of the training set. Additionally, another approach that was used to reduce "noise", in other words the incorrectly classified irrelevant instances, is to check from which platform the tweets were posted and which amount of irrelevant instances the model needs to be trained on to reduce misclassification. Despite, these techniques concerning the irrelevant tweets and noise reduction, the algorithm still does not stray from the direction of "paying attention" to relevant tweets. Thus, after the data splitting, more balancing approaches are investigated, so the classifiers can be trained to achieve the desired performance.

The data of the second phase (semi-supervised) is the unlabeled air pollution dataset, also mined from twitter. This part of the data consists of 1852017 tweets. The features of this dataset are source, user id, text, date created (created_at), coordinates, time zone and place. The unlabeled data are primarily part of the semi-supervised approach of text classification. So, the purpose of this part of the dataset is to investigate if the classification is improving or getting worse by increasing gradually the training set with automatically labeled instances.

3.2 Data Labeling

In order to perform the required experiments and evaluate the performance of different algorithms in both supervised and partially supervised approaches, there must be a significant amount of labeled data. By having labeled data we are able to test the predictions of the different classifiers in any approach, compute the accuracy, precision, recall and various other performance metrics and evaluate, interpret the results in order to make conclusions. So, it is necessary to have labeled data in both classification approaches.

A first problem is how a dataset should be labeled. Firstly, the problem that needs to be solved must be clearly defined. Secondly, a set of rules must be also set in order to accurately label the data manually. Regarding this project, the available data, need to distinguish tweets as relevant or not on reporting air pollution or breathing issues. In terms of labeling rules this means:

1. distinguish literal and metaphorical meaning of air pollution and breathing problems related phrases

2. identify tweets posted from actual people and those from news sites or bots

3. discriminate tweets that report an incident or infection from those that are not giving any specific information

For example, a tweet with literal meaning would be "All I can hear is myself wheezing", which apparently is about a breathing problem and reports that this person has some health issue. Although, a tweet with metaphorical meaning would be "I'm laughing so hard, so I can't breath", which obviously does not report any breathing issues. Moreover, these were examples of tweets posted by actual people. For instance, a tweet by a news site posted automatically using a bot would be "Kids with asthma fare worse when they live with smokers - Reuters UK", which is apparent that it was posted by this particular news site. Additionally, this tweet also demonstrates a case that does not report specific information and just makes a generic statement. Therefore, all tweets about air pollution or breathing problems that have literal meaning, are posted by individuals and give specific information about a health incident are marked as "Relevant".

All of the above rules are taken into consideration while the manual labeling process takes place. A certain amount of collected tweets are labeled by a group of people, who divide the tweets equally. They label each tweet based on these rules and their judgment if there are some controversial cases. After the labeling, all the subdivisions of tweets are being redistributed among the group in order to check each others labels and see if their labeling matches and also decide on any controversial cases. Since the confirmation of all labels is completed the data are ready to be used in the KDD process.

3.3 Data Cleansing & Preprocessing

Since there is a certain amount of labeled data and unlabeled data as well the KDD can proceed with the stages of cleansing and preprocessing. In a Text Mining (TM) context there is a number of things that need to be taken care of. These are the following:

1. create initial corpus
2. convert all words in lower case
3. remove URLs
4. remove numbers

5. remove punctuation (optional)
6. define a set of stop words
7. remove stop words
8. remove white space
9. perform stemming
10. create a document term matrix from the created corpus

Text data must be represented in the form of a **corpus**, in order to be processed. The corpus is the body of text that will be subject to classification. Additionally, in terms of cleansing the data, we remove all the elements that we know will not be useful. Since the data come from twitter, it is likely to have URLs and because we deal with natural language text there are also numbers and punctuation. Thus URLs, numbers and punctuation can be removed, because they may not add any value to the data. However, punctuation removal is optional because different emojis or emoticons can be found in various tweets and those are basically combinations of punctuation marks.

A rather important step in that phase is **stop word removal**. We defined a list of words that we are sure that do not add any value to data quality. For example, articles such as “the” or conjunctions such as “and”, are extremely common and are not helpful in mining the desired knowledge. Furthermore, white space removal takes place in order to prepare the corpus for stemming.

Another important step in that phase is **stemming** (Porter, 2001). All the processes in the corpus are performed in the initial form of the text. This part of the process takes all the words that have the same root and keeps only the radical in corpus. Many words have the same radical and their ending differs depending on which part of speech are used. So, the different versions of a term, that have similar if not the same meaning, are redundant. Hence, with this method we increase data quality by reducing the number of different features that represent text data.

The last step of that phase is to construct a **Document Term Matrix (DTM)**. Every row represents each document in the dataset, hence each tweet. Every column represents a term that has been used in a document. Moreover, the text representation method used was **term frequency**. For every document (rows), under each term (columns) that exist in the document, it is represented by how many times this term is mentioned in that particular document (tweet).

3.4 Text Representation Methods

Bag of words (Harris, 1954) is one of the most well known text representation methods. This technique divides a document to its words while every word is represented only once in this bag of words. In this case each tweet is a document. For example, the tweet "I wish I lived in an area with less air and atmosphere pollution" would be represented with the following words: "I", "wish", "lived", "in", "an", "area", "with", "less", "air", "and", "atmosphere", "pollution". So, "I" because it already exists in this bag of words doesn't need to be added again.

N-grams combine terms together and each feature in DTM is a phrase where N is the number of terms we want to combine. The most common types of n-grams are bigrams, threegrams and fourgrams, which create features of two, three and four words combined respectively. The purpose of this technique is to find words that on their own might not be so important but in combination with some other word might be much more significant and increase data quality. Finally, in this project threegrams are used and tested.

3.5 Term Weighting Methods

Term Frequency is a widely used and one of the most common text representation methods. This method counts how many times each word is present in a document and assigns it to under the column of the DTM that represents this particular word. It is the default text representation method used in this KDD process. The default term frequency formula, which at the same time is the simplest one is $tf(t, d) = f_{\mathrm{t,d}}$, where t is the term, d is the document and f the frequency of that term in that document.

Terms Frequency-Inverse Document Frequency (TF-IDF) (Luhn, 1957; Sparck Jones, 1972) is another well known text representation method. Essentially, TF-IDF measures the importance of a word in a document and more specifically in a corpus. It computes the relative frequency of every word in each tweet compared with the frequency of that word over the corpus of the whole tweet collection. The formulas of how the weights of TF-IDF are calculated are the following:

$$tf(t,d) = 0.5 + 0.5 * \frac{f_{\mathrm{t,d}}}{max(f_{\mathrm{t',d}} : t'\epsilon d)}$$

(term frequency)

$$idf(t,D) = \log \frac{N}{|d\epsilon D : t\epsilon d|}$$

(inverse document frequency)

$$tfidf(t,d,D) = tf(t,d) * idf(t,D)$$

(term frequency-inverse document frequency)

where t stands for term, t' represents the most occurring word in a document, d stands for document, f stands for frequency, N is the total number of documents in the corpus and D is the collection of documents in the dataset.

3.6 Splitting and Balancing The Data

At this point the dataset needs to be split into a training and a test set. The proportion used to split the labeled part of the dataset is 60%-40%. The 60% of the dataset serves as the training set and the 40% as test set. The training set is used to train the different classification algorithms. The test set has the purpose of evaluating the different classifiers. By using the trained model for all the instances in the test set new labels are predicted. Therefore, since the actual ones are known, because they were assigned manually in an earlier phase, we can evaluate the performance of those algorithms that have been used.

As it was mentioned earlier, the instances labeled as “Relevant” are outnumbered by the ones labeled as “Not Relevant”, which is the majority class. For this reason, the training set needs to be balanced. This means that there is a certain portion of “Not Relevant” tweets that the classifiers need to learn as efficient as possible. On the one hand, if there are less instances from the majority class in the training set, the algorithms might classify incorrectly many “Not Relevant” instances in the test set as “Relevant”. On the other hand, if there are more instances from the majority class in the training set, the classifiers might label incorrectly many “Relevant” instances as “Not Relevant”. In other words, there is the danger of coming across overfitting. Therefore, a balancing approach has to be defined in order to achieve the desired performance.

Before settling in a balancing approach, several approaches were explored. The experiments for investigating a balancing approach started with both classes having the same amount of instances. In every trial, we explored the results by multiplying the size of the majority class and increasing the multiplier by 5% increments each time until we reach the preferred amount of irrelevant tweets in the training set. From that point further investigation was undertaken by increasing the multiplier 1% each time, to see if the number incorrectly classified instances of the majority class will drop any more. Finally, the balancing approach of the training set was settled in having all the available relevant instances in the 60% of the data plus 2.6 times more irrelevant tweets. Consequently, after investigating how the data should be balanced, the balancing approach that was used to perform the classification task is the following:

1. Keep all the “Relevant” instances of the initial 60% of the data in the training set
2. Keep 2.6 times more “Not Relevant” instances than the number of “Relevant” instances in that 60% of the data.

3.7 Feature Selection

The last step before classification, but after the data have been split, is feature selection. Several techniques were tested to see if there is any that can improve classification accuracy by selecting the most important features (terms). The methods that were implemented are the following:

1. Chi-Square (χ^2)
2. Information Gain (IG) (Kullback and Leibler, 1951)
3. Odds Ratio OR (Cornfield, 1951)

These techniques are well known feature selection methods, which were applied after DTM was built. The methods are developed in statistics but they found application in data mining and machine learning also. These techniques are used to calculate the importance of features in a dataset. All three methods trying to select the best features or in other words the most relevant ones in different ways.

3.8 Feature Engineering

Topic Modeling is one of the feature engineering methods that were incorporated in our KDD process. The algorithm used for that purpose is **Latent Dirichlet Allocation (LDA)** and its purpose is automatic topic discovery. Firstly, a number of topics is manually defined, depending on the possible topics that might exist in the collection of documents. Secondly, every word will be assigned to a temporary topic according to Dirichlet Distribution. Finally, the topic assignments will be checked and updated regarding the word distribution in all identified topics long with the topic distribution in every document.

Source Filtering is the technique used to reduce the number of irrelevant tweets by identifying which ones are not posted by individuals. The idea of this feature engineering technique is to remove tweets created by bots or news sites that do not give the kind of information that we are looking for and at the same time reducing the data quality. Firstly, all the sources of tweets, that belong to the "Relevant" class, were extracted and identified, in order to produce a list of "reliable" sources. Secondly, all the instances of the "Not Relevant" class that were not posted by one of these "reliable" sources were removed. So, all the sources from the relevant tweets were identified and then every source that were not in the set of relevant sources will be removed.

3.9 Supervised Learning

In the data mining phase of the KDD process were used some well known classification algorithms, which are the following:

1. Naive Bayes (Maron, 1961)
2. Support Vector Machines (Vapnik and Chervonenkis, 1964)
3. Decision Trees (Quinlan, 1986)
4. Random Forests (Ho, 1995)
5. Scaled Linear Discriminant Analysis (Fisher, 1936)
6. Maximum Entropy (Jaynes, 1957a,b)

As an input the algorithms used the training set with the balancing approach defined in a previous section. The classifiers produced a model, which were tested using the test set. Moreover, the trained models were used to predict new labels for the instances that belong in the test set. Furthermore, since the actual labels are available along with the predicted ones, a confusion matrix is constructed in order to compare and contrast the results given by each classifier. Finally, the performance of these algorithms are evaluated and discussed in a later stage.

Naive Bayes

Naive Bayes(NB) (Maron, 1961) is based on the assumption of **class-conditional independence**, which suggests that the impact of a variable on a class is independent from the other variables in the set (Han et al., 2011). Moreover, NB is also based in **Bayes' Theorem**, which has the following formula:

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

where A and B are two different observations, P(A) and P(B) are the probabilities of those events being true independently and $P(A \mid B)$ is the conditional probability of event A given that the event B is true.

Since NB's concept is based on mathematics and probabilities, a question that might be posed is how this particular algorithm will work in a text mining context. NB classifier takes as an input a document in the form of a vector $d_j = (t_1, t_2, ..., t_v)$, where v is the number of features (terms) in the dataset $D = \{d_j | d_j \epsilon D, j = 1, 2, ..., N\}$, where N is the number of documents consisting that dataset. Each document is labeled with a category, in other words class. The probability of a document to be labeled with a category is based on **Bayes rule**. Finally the classifier in order to label the data accurately assigns each time the **maximum class** (Zhu and Wong, 2014) and that probability can be estimated by the following equation:

$$P(c_i \mid d_j) = \frac{P(d_j \mid c_i)\, P(c_i)}{P(d_j)}$$

where c_i is a label for a document d_j, $P(c_i)$ is the probability of that label being true, $P(d_j)$ is the probability of that given document, $P(d_j \mid c_i)$ is the prior probability and $P(c_i \mid d_j)$ is the posterior probability, which is the conditional probability of the label c_i given a document d_j.

Support Vector Machines

Support Vector Machines (SVM) (Vapnik and Chervonenkis, 1964; Boser et al., 1992) are algorithms suitable not only for **linear data** but also **non-linear data**. The algorithm "uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating **hyperplane**." (Han et al., 2011). This hyperplane can divide the data into two classes, in this case relevant or irrelevant, with the sufficient non-linear mapping to a higher dimension. SVM calculates the optimal position of hyperplane using **support vectors** and **margins** (Han et al., 2011; Liu et al., 2011).

In a classification task that SVMs are used for learning multiple hyperplanes might exist. The algorithm uses the Maximum Marginal Hyperplane (MMH), in order to find the best hyperplane that can be used to classify the data more efficiently. A margin is the minimum distance from either side of the hyperplane. Furthermore, support vectors are essentially the important data points, which are located close to the hyperplane. Therefore by calculating the margins of all possible hyperplanes, the support vectors and the MMH, SVM find the hyperplane that separates the data and performs the classification task (Han et al., 2011).

On the other hand, the non-linearly separable data SVMs use different types of hyperplanes. These types of hyperplanes divide different non-linearly separable data by using a kernel function. There are four types of kernels used in SVMs, which are the following:

1. linear
2. polynomial
3. radial
4. sigmoid

However, the logic of how the algorithm works does not change, only how the data are treated. All the different kernels are essentially expansions of the linear SVMs. In every case, the algorithm tries to compute the best position of the hyperplane in order to perform the classification task.

Decision Trees

Decision Trees (DT) (Quinlan, 1986) are structures that consist of different types of nodes. At the top of the tree we have the root node, the internal nodes and the leaf nodes. The learning process of decision trees is called **decision tree induction**. Every decision that has to be made has only two possible outcomes. In the context of classification every instance of the data, every tweet is tested on the decision tree constructed during the training process. The testing phase begins from the root node and the path that takes leads to a specific leaf node, which represents the prediction that the algorithm makes. Every document has already a label assigned and the predicted labels can be compared to the actual ones to evaluate the performance of this algorithm (Han et al., 2011).

Random Forests

Random Forests (Ho, 1995) is an ensemble method, based on Bagging (Breiman, 1996) and Decision Trees (Quinlan, 1986). "Random forests are defined as an ensemble of decision trees, in which randomness has explicitly been inserted into the model building process of each decision tree." (Aggarwal, 2015). Thus, the concept of this particular method is to incorporate randomized decision tree models, in order to reduce variability. Furthermore, this algorithm it is known to work better in datasets with high-dimensionality (Aggarwal, 2015). This means that in the case of tweet classification with a training and a test set having numerous columns, in other words dimensions, random forests will potentially perform well.

Scaled Linear Discriminant Analysis

Scaled Linear Discriminant Analysis (SLDA) (Fisher, 1936) is a well known statistics method that found applications in machine learning either for classification or as a feature selection method. This algorithm " searches for the project axes on which data points of different classes are far from each other while data points of the same class are close to each other." (Nassara et al., 2016). SLDA is based on the assumption that the probability density functions of both classes are normally distributed. The formulas that describe those functions are the following:

$$p(\vec{\chi}|y=0), \quad (\vec{\mu}_0, \Sigma_0) \qquad\qquad p(\vec{\chi}|y=1), \quad (\vec{\mu}_1, \Sigma_1)$$

where $\vec{\chi}$ is the training set, $\vec{\mu}$ is the mean and Σ is the covariance for each class respectively. Moreover, this algorithm uses a bayesian approach to predict labels. Furthermore, if the logarithm of the confidence probability of an instance is more than the predefined threshold then this instance is assigned to the class with the maximum confidence probability. Otherwise, if the log of the confidence probability of an instance is less than the predefined threshold then it is assigned to the opposite class. The formula that describes the concept of SLDA is the following:

$$((\vec{\chi}-\vec{\mu}_0)^T\Sigma_0^-1(\vec{\chi}-\vec{\mu}_0)+\ln|\Sigma_0|)-((\vec{\chi}-\vec{\mu}_1)^T\Sigma_1^-1(\vec{\chi}-\vec{\mu}_1)-\ln|\Sigma_1|)>T$$

where $\vec{\chi}$ is the training set, $\vec{\mu}$ is the mean, Σ is the covariance for each class and T is the threshold involved in assigning labels.

Maximum Entropy

Maximum Entropy (MaxEnt) (Jaynes, 1957a,b) is another classification method that is based on multinomial logistic regression and the principle of maximum entropy. The concept of this classifier is to predict the probabilities of an dependent variable to belong to different classes regarding a set of independent variables. In this case, the classes are "Relevant" and "Not Relevant" and the independent variables are the words that are featured in each tweet. This algorithm is based on uniform distribution to produce a model that performs classification. As a probabilistic classifier does not assume that the data are conditionally independent, in comparison with Naive Bayes. The idea of how MaxEnt does classification is to fit a model that has the highest entropy. One of the first applications of this classifier in document classification was proposed by Nigam et al. (1999).

3.10 Semi-Supervised Learning

Conventional classification algorithms, such as the ones previously mentioned, use only labeled data in both training and testing. Researching an approach of partially supervised classification of documents takes place because labeling data manually can be tedious and time consuming. Additionally, manual labeling is a difficult and complicated process, because there are a lot of aspects to consider and it can also be subjective. However, it is necessary to have labeled data in a semi-supervised learning approach on document classification, in order to be able not only to train a classifier but to test also the results. The aim is to investigate how unlabeled data affect the results of a learning process. Therefore, we need to know how many unlabeled instances can be beneficial to the overall accuracy, precision and recall, by taking into consideration the number of the labeled instances.

To investigate the impact of unlabeled data in classification accuracy a separate KDD process was conducted. The process of the semi-supervised approach slightly differs from the one that was conducted for conventional, supervised classification. It started with performing a preliminary analysis and preparing the dataset for cleansing and preprocessing. In this phase, IDs are assigned to each unlabeled observation and the order of features in the unlabeled dataset is modified according to the labeled set in order to be compatible during the partially supervised classification process and avoid unexpected errors. Furthermore data cleansing and preprocessing is performed.

Moreover, the data are split into training, testing and unlabeled set with the training set being totally balanced. Finally, the training set is the input of the semi-supervised classification algorithm and the test set is used to evaluate its performance.

The approach that this project uses to perform partially supervised classification is **Self Training** or **Self Learning** (Yarowsky, 1995). This method's training process consists of two parts. Firstly, a well known classifier is trained and tested using the labeled observations. Secondly, a fixed amount of unlabeled instances are labeled using the model that was just built. Thirdly, the unlabeled observations that acquired labels and have confidence probability above a specific threshold are being part of the training set which the model was trained on. Furthermore, the new training set becomes the new input of the classifier which will produce another model. The new model will be tested again and will be used to predict another set of unlabeled data with the same amount as the previous one. This process is repeated until the all instances of the unlabeled set have predicted labels and there are no more data left to predict. Finally, after several iterations the self training approach on performing partially supervised classification produces the final model which is tested using the test set consisting manually labeled instances, in order to be able to evaluate the final results (Torgo, 2016). This method is also called Yarowsky's algorithm (Yarowsky, 1995).

Algorithm: *Self Training Using Naive Bayes*	
Input	Training Set, Test Set, Unlabeled Set, N Size of Unlabeled Data for Prediction
Output	Model trained using both labeled and unlabeled data
1	Train model using Naive Bayes classifier
2	Test model
3	Produce a confusion matrix and performance metrics
4	Predict labels for N unlabeled instances
5	Add every instance with confidence probability above 0.95 in the training set
6	Repeat

Table 3.1: Self Training Algorithm Based On Naive Bayes

A question that might be posed is why Naive Bayes. NB was chosen as the base of self training approach to perform partially supervised classification after conducting experiments for the comparison of the different classifiers described previously. The performance of each algorithm was evaluated using various metrics. Based on those metrics this decision was made. The results of these experiments along with discussion are presented in th next chapter, while the measures used are described below.

3.11 Analytics & Performance Metrics

In the last phase of the KDD process, it is necessary to have a number of metrics in order to evaluate the performance of the different classifiers and analyze their results. The implemented performance metrics are the following:

1. Accuracy
2. Precision
3. Recall
4. F1 Score
5. F2 Score

After each model is trained and the new labels are predicted, it is necessary to compare the actual labels of the test set with the predicted ones. A confusion matrix is constructed in order to do that comparison and be able to evaluate classifiers' performance. Each row of the matrix represents the actual labels of the data and each column the predicted labels. On the diagonal of the matrix are located the number of correctly labeled instances and on the anti-diagonal the number of incorrectly classified observations. Thus, a confusion matrix looks like the following table:

	Predicted	
Actual	***Not Relevant***	***Relevant***
Not Relevant	True Negatives	False Negatives
Relevant	False Positives	True Positives

Table 3.2: Example of a Confusion Matrix

Since a confusion matrix is constructed for each model, at this point we can calculate all these performance metrics that were mentioned above. Firstly, we have the overall accuracy of the model, which is the sum of correctly classified labels over the total number of instances, where:

$$TotalInstances = TruePositives(TP) + FalsePositives(FP) +$$
$$+ TrueNegatives(TN) + FalseNegatives(FN)$$

$$Accuracy = \frac{TP + TN}{TotalInstances}$$

Secondly, we have precision, which is calculated for each label, hence there are two values that represent how precise was the classifier in handling each class. Precision is equal to the true positives or true negatives over the sum of the predicted values of each class. In other words, precision is equal to the number of the correctly classified instances, which are located at the diagonal of the confusion matrix, over the sum of each column.

$$Precision = \frac{TP}{TP + FP} \qquad Precision = \frac{TN}{TN + FN}$$

(True Positive Rate) (True Negative Rate)

Thirdly, there is recall or sensitivity, which is also computed for both classes, where each class has its respective value. Recall is equal to the true positives or true negatives over the sum of the actual labels. In other words is the the number of the correctly classified instances, which are located at the diagonal of the confusion matrix, over the sum of each row.

$$Recall = \frac{TP}{TP + FN} \qquad Recall = \frac{TN}{TN + FP}$$

(Relevant Instances) (Not Relevant Instances)

Additionally, there is Fβ Score, where $\beta = 1, 2$. Hence Fβ Score becomes F1 and F2 Score respectively. On one hand, F1 Score is considered the harmonic mean between precision and recall, while the F2 Score gives more weight on recall. F1 and F2 Scores are calculated as follows:

$$F\beta = (1 + \beta^2) * \frac{Precision * Recall}{\beta^2 * Precision + Recall}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}, \qquad \beta = 1$$

$$F2 = 5 * \frac{Precision * Recall}{4 * Precision + Recall}, \qquad \beta = 2$$

3.12 Tools

Programming in R

The presented methodology and the respective experiments are implemented in R. R is a programming language used not only in mathematics, statistics and actuarial sciences but also in computing sciences for data mining and data analysis. It is part of the GNU project and its development began at 1992 by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand. R provides a wide range of packages of statistical, graphical and data mining techniques. R is available as Free Software under the terms of the Free Software Foundations GNU General Public License in source code form and compatible with Windows, MacOS, Linux and other UNIX-like systems.

High Performance Computing

Due to the large quantity of data, personal computers are not always capable of processing them given the limited resources in memory. For that reason all the experiments had to be conducted using the high performance computing cluster of the University of East Anglia (UEA). It runs Linux as an Operating System (OS) with Linux terminal as the interface for accessing its functions and capabilities. It was a vital part of the KDD process that was conducted.

3.13 Problems That Occurred

Overfitting

A common problem that might occur in a data mining process is overfitting. This problem is occurring when a model has been trained on a training set that includes too many unnecessary features, in other words “noise”. This “noise” is influencing the training process of the classifier. As a result it affects its predicting abilities in the testing phase. There are two types of overfitting identified when “Using a model that is more flexible than it needs to be.” (Hawkins, 2004) and when “Using a model that includes irrelevant components” (Hawkins, 2004). This issue occurred in the semi-supervised learning part of this KDD process, as it is revealed in the next chapter.

Programming Errors

While implementing the above methodology and conducting the necessary experiments, it is common to come across several programming issues. Some types of errors that occurred are the following:

1. Syntax Errors
2. Logical Errors

The first category refers to issues caused by mistyping commands. This is a common type of errors even though most compilers and development environments provide warnings to the users. So, it is the easiest type to solve. Moreover, logical errors have to do with the way the program works. In this case the commands that implement a certain task are not defined clearly. Therefore, it is necessary that the commands, that implement the different tasks, to be explicitly defined.

Issues With Memory

Due to the size of the dataset, along with the high-dimensional corpus and document term matrix, the conducted KDD process requires a lot of memory. In real time several gigabytes of data must be processed in order to perform the previously mentioned methods and techniques and get the experimental results that will be discussed in the next chapter. In several occasions during experiments that were undertaken on a personal computer the process stopped because the computer was running out of memory. Therefore, the use of High Performance Computing was absolutely necessary.

Software Compatibility

Another problem that we came across is the incompatibility of certain packages between R versions and OSs. In this case, the two OSs used are Windows and Linux. Because of the differences on the way these OSs function and on how they are built, the way different pieces of software are implemented depends on which OS it is developed for. Therefore, running R scripts in a Windows machine and on a Linux machine might have different behavior in certain occasions.

Chapter 4

Results & Analysis

One of the most important parts of a KDD process is the evaluation and the interpretations of results. “Evaluation is the key to making real progress in data mining.” (Witten et al., 2016). There are several means to evaluate the performance of a classifier. These performance metrics are accuracy, precision, recall, specificity, F1 Score and F2 Score.

The experiments that were undertaken are divided into two approaches, the supervised and the semi-supervised. For each approach the equivalent performance metrics are presented for every classification algorithm involved. This allows the comparison of the performance of the different classifiers and at the same time multiple views on the same problem. Moreover, by analyzing these metrics we are also able to interpret the results, in the effort of deriving knowledge. Additionally, some graphs are added to the equation in order to examine the relationships between metrics, such as precision and recall.

Before proceeding with the evaluation and interpretation, it is necessary to point out again some characteristics and limitations of the data. The dataset used for these experiments is a collection of tweets about air pollution and breathing problems. The labeled set consists of 3500 tweets and the unlabeled set 1852017. The first limitation, which is potentially the cause of issues such as overfitting, is that the dataset is unbalanced, with the majority class being the “Not Relevant” and the minority class being the “Relevant”. Regarding that tweets labeled as relevant or the ones that can be marked as such are the ones that we are interested in, makes the training more complicated and testing challenging to all the different classifiers.

4.1 Supervised Classification using Term Frequency

The experiments of the supervised approach followed the KDD process as it was described in the previous chapter. The dataset used is a collection of tweets about air pollution and breathing problems and consists of 3500 instances. For text representation was used the bag of words approach and term frequency for weighting. By using bag of words the constructed document term matrix consisted of 6867 terms. Afterwards, the labeled dataset was split into training and test using random sampling with proportions 60% - 40% respectively, with the balancing approach described earlier. In order to perform supervised classification several well known classifiers were used. Below the results that these classifiers produced are presented along with their interpretation regarding only the class of interest which is the “Relevant” class.

Accuracy	**Precision**	**Recall**	**F1**	**F2**
Naive Bayes				
72.2%	37%	93.2%	53%	71.5%
Support Vector Machines				
88.8%	72.7%	54.5%	62.3%	57.4%
Random Forests				
65.6%	29.5%	74.9%	42.3%	57.3%
Decision Trees				
55.5%	21.9%	63.4%	32.5%	46%
Scaled Linear Discriminant Analysis				
74.5%	38.3%	83.3%	52.4%	67.4%
Maximum Entropy				
80.4%	45.7%	84.8%	59.4%	72.4%

Table 4.1: Performance metrics of Classifiers for the “Relevant” class using Term Frequency

By looking how these classifiers performed the results vary in all metrics regarding relevant tweets. Some classifiers are more appropriate for tackling tweet classification than others. But we have to consider all metrics in order to judge which one actually outperformed the others according to the defined problem and the aim of the project.

Regarding the accuracy of NB at 72.2% does not seem promising enough. This notion is being supported by the level of precision that this classifier achieved, which is at 37%. On the other hand SVM's performance seem much more successful with 88.8% accuracy and 72.7% precision. However, having high precision, as in SVM's performance, suggests that there were much less misclassified not relevant tweets, while low precision means there were more not relevant tweets predicted as relevant. By considering this, seems that all the other classifiers also categorized incorrectly many irrelevant tweets as relevant. SLDA and MaxEnt classifiers achieved higher accuracy and precision than NB, with the later being at 38.3% and 45.7% respectively while RF and DTs have much lower rates in both metrics. Therefore, NB, RF, DTs, SLDA and MaxEnt have a lot of "noise" in the class we are interested in while SVM was more successful regarding the classification of not relevant tweets.

The levels of accuracy and precision that these algorithms achieved suggest that SVM performed better than the other classifiers. Nevertheless, recall suggests that the roles have been reversed. NB achieved the highest recall at 93.2% and SVM 54.5%, which is the lowest. This means that NB performed very well regarding the class of interest. Having really high recall in relevant tweets, implies that there were very few misclassified relevant tweets as not relevant. On the other hand low recall in the class of interest means that there were many tweets incorrectly categorized as not relevant, which is the case for SVM. SLDA and MaxEnt demonstrated the next most promising recall. This suggests that most of the relevant instances were correctly classified and because they are outnumbered by misclassified not relevant tweets, precision is lower. Moreover, RF and DTs achieved lower rates in this metric compared to the others but still greater than their precision, which leads to the same conclusion regarding their performance. Nevertheless, high recall on the class of interest is a satisfying result. Therefore, regarding recall the most appropriate classifier to tackle this classification task is Naive Bayes.

Another way to evaluate the performance of a classifier is the $F\beta$ measure with $\beta = 1$, in other words F1. This metric balance precision and recall equally and serves as their harmonic mean. Considering the results of these classifiers SVM seems to have the best performance due to high precision in the majority class. Furthermore, it is followed by MaxEnt, NB and SLDA which were higher in recall. The rest classifiers had also higher recall than precision in which both metrics were in lower levels and this is reflected in F1 measure too. Thus, F1 score suggest that recall combined with precision can show the overall performance considering both the how accurate in the class of interest a classifier is together with the ratio of misclassified instances of the opposite class.

Having the first impression of a classifiers performance from measuring its accuracy is not enough. At the same time precision and recall give more insight regarding "noise" in the class of interest and misclassified observations. $F\beta$ measure with $\beta = 2$, alternatively F2, provides the most representative view of the performance of a classifier because gives more weight on recall than precision. Hence it is focused more on how complete a classifier is than how precise. In this case, the classifiers that have the most promising performance are MaxEnt and NB with 72.4% and 71.5% respectively, while the rest rank as follows: SLDA, SVM, RF and DT. Therefore, NB and MaxEnt seem to be the most suitable algorithms of the six to tackle a document classification problem such as this one.

To sum up, as all of the above results have demonstrated that accuracy might not always represent if a classifier is the appropriate one for tackling a classification task. In this case for example, NB with accuracy 72.2% seemed to performed worse than SVM with 88.8%. However, if it was necessary to choose one between the two the most appropriate, Naive Bayes would be a better choice because it demonstrated higher recall in class of interest which is the "Relevant" tweets. The same applies if SLDA or MaxEnt were compared with SVM. However, NB in comparison with MaxEnt or SLDA the same decision would be made due to higher recall. Regarding RF and DT, the overall performance was poor having misclassified instances of both classes. Therefore, classifiers such as NB, SLDA and MaxEnt, seem to perform better in the context of mining text data.

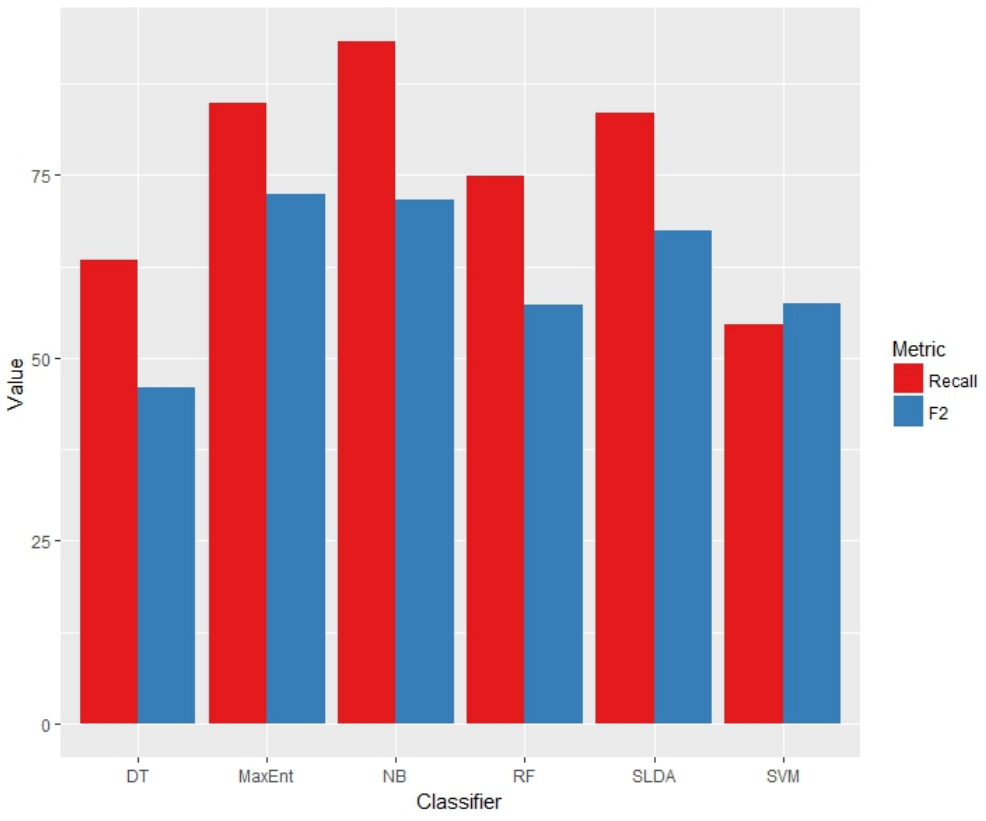

Figure 4.1: Recall and F2 Score of Supervised Classification with Term Frequency for “Relevant” tweets

In the above figure, it is shown the metrics of recall and F2 Score of these classifiers, regarding the class of interest.

4.2 Supervised Classification using TF-IDF

Another supervised experiment was conducted, this time by weighting the terms of the dataset with TF-IDF comparing the same classifiers. It was used the same dataset about air pollution and breathing problems with 3500 labeled tweets. The 60% of those instances served as training set and the 40% as test set. The balancing approach described in chapter 3 was used on the training set. The text was represented using the bag of words approach and TF-IDF weights. The following table shows the results provided by this KDD process.

Accuracy	**Precision**	**Recall**	**F1**	**F2**
Naive Bayes				
50.9%	20.6%	93%	33.7%	54.6%
Support Vector Machines				
91.2%	68.6%	63.8%	66.1%	64.7%
Random Forests				
60.3%	24.2%	91.4%	38.2%	58.7%
Decision Trees				
43.4%	14.5%	65.9%	23.8%	38.6%
Scaled Linear Discriminant Analysis				
73.8%	31.9%	83.5%	46.1%	63.1%
Maximum Entropy				
75.6%	34%	88.7%	43.1%	67.1%

Table 4.2: Performance metrics of Classifiers on “Relevant” class using Term Frequency-Inverse Document Frequency

Having compared and discussed the performance of those classifiers when using simple term frequency, the first impression from the results provided by using TF-IDF is that all classifiers performed worse. NB performance regarding accuracy was dropped at 50.9%, which is a 21.3% difference. Its precision was dropped also at 20.6%. On the other hand SVM's accuracy was increased but its precision fell at 68.2%. This means that there are more relevant tweets are incorrectly classified. In SVM's case precision dropped but the accuracy increased. This reflects that the incorrectly classified irrelevant tweets were more this time around and the true relevant ones were more as well. The other classifiers demonstrated dropped both accuracy and precision.

Regarding recall the results also indicate lower performance except in the cases of RF and SLDA that recall was increased. Especially RF performed surprisingly well compared to its performance with term frequency, achieving recall at 91.4%, while SLDA had a 0.2% increase. However, NB achieved once again the highest recall and

SVM the lowest. The drop in recall mean that more relevant tweets were classified as irrelevant this time around. As a result we lost more “signal” while “noise” increased.

Considering the F1 score it is reflected that this approach performed worse than the previous one. Similarly, the same applies to the F2 score, while it is greater than F1 because of weighting recall more. In both metrics SVM appears to have the best performance because it has similar rates of precision and recall. Although NB and SLDA demonstrated bigger difference between F1 and F2 which reflects the high performance on recall in both classifiers. Therefore these two classifiers are considered more suitable.

Overall, by looking at the accuracy, classifier’s performance does not seem satisfying, but can not efficiently represent the overall performance. Metrics such as precision and recall can provide a better description on how a classifier performed in a case such as this one and along with accuracy and the F measures we can have a better understanding on how NB performed. Therefore, Naive Bayes performed well in classifying the relevant tweets but in the case of the irrelevant ones despite many of them where correctly categorized, a significant number of them was classified incorrectly.

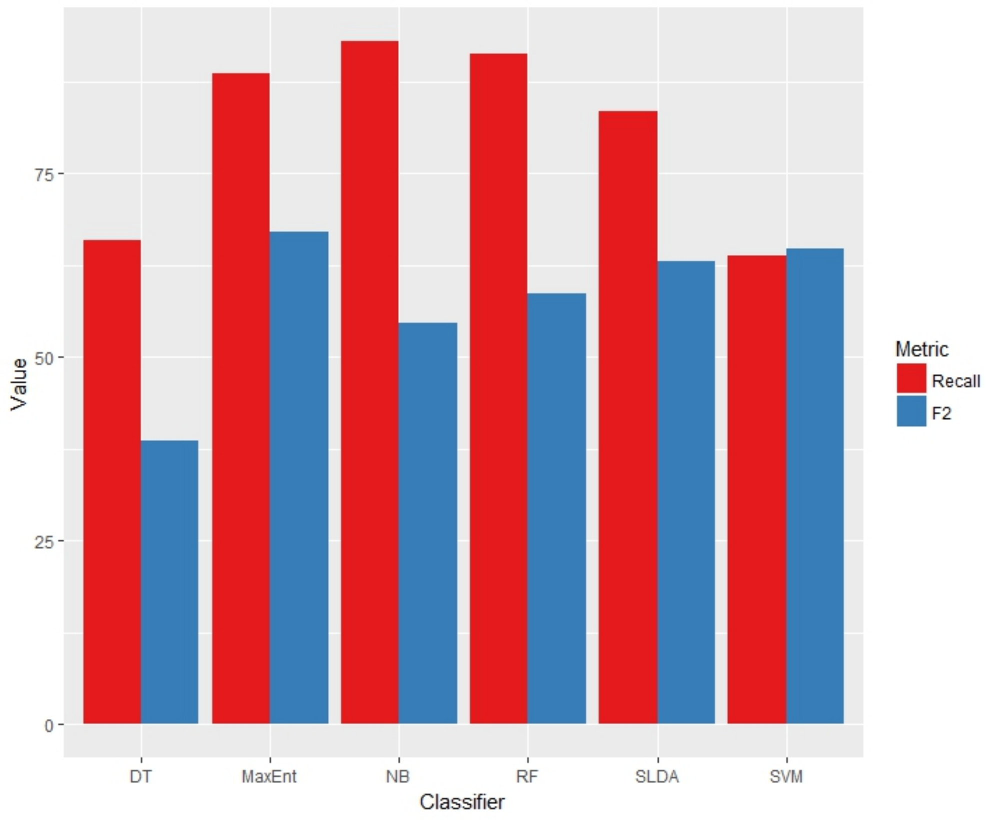

Figure 4.2: Recall and F2 Score of Supervised Classification with TF-IDF for “Relevant” tweets

In this graph above, it is demonstrated the performance of these classifiers through recall and F2 Score, regarding the class of interest.

4.3 Supervised Classification using Three-grams

This set of experiments are based on a supervised approach which follows the KDD process as it was described in the third chapter. The data used are about air pollution and breathing problems and consist of 3500 tweets labeled either as "Relevant" or "Not Relevant". This time the text representation method used is three-grams and term frequency for weighting the terms. The dataset is split into training and test with random sampling, 60% - 40% respectively, having 2289 terms. Additionally, the training set uses the balancing approach discussed earlier. In order to perform supervised classification several well known classifiers were used. Below the results are presented along with their interpretation regarding only the "Relevant" class.

Accuracy	**Precision**	**Recall**	**F1**	**F2**
Naive Bayes				
72.7%	38.5%	94.9%	54.8%	73.4%
Support Vector Machines				
89.9%	79.4%	56.8%	66.2%	60.2%
Random Forests				
62.5%	28.9%	79.1%	42.3%	58.7%
Decision Trees				
51.5%	16.1%	43.1%	23.5%	32.3%
Scaled Linear Discriminant Analysis				
77%	42.3%	88.8%	57.3%	72.8%
Maximum Entropy				
82.6%	50.1%	87.8%	63.8%	76.3%

Table 4.3: Performance metrics of Classifiers using 3-grams on the "Relevant" class

So far regarding the experimental results of supervised classification using term frequency the algorithms some demonstrated well regarding the class of interest but having "noise" as well. In the case of TF-IDF the performance was lower. This time around term frequency is the term weighting method and every term consists of three words, alternatively three-grams (n-grams, where n=3). The results that this text representation method brought in combination with term frequency and these six classifiers, are slightly improved.

Regarding NB's performance the improvement was not significant with its accuracy being at 72.7%, which is a 0.5% increase. Precision increase 1.5% at 38.5%, which means there were less misclassified not relevant instances. In SVM's case, the performance was also better. It achieved 1.1% increase in accuracy which is 89.9% but i precision the improvement was significant. The 79.4% is the result of 7.3% increase. Similarly, SLDA and MaxEnt had also improved performance. However, this is not the case with RF and DT. These two classifiers demonstrated a drop both in accuracy and precision, which leads to the conclusion that three-grams was not as efficient in decision making for predictions in comparison with the other classifiers that achieved better performance and this text representation method was more beneficial overall.

The interesting part as always is to investigate classifiers performance in recall, since it reflects how complete and efficient these algorithms were in predicting instances of the "Relevant" class. Once again NB demonstrated the highest recall in this set of experiments with this time DTs having the lowest. Like in accuracy and precision all classifiers increased their performances regarding recall except DTs. This means that less incorrect categorizations took place while the slight increase on precision indicates that "noise" was decreased as well. Nevertheless this does not apply to decision tree performance using three-grams. Therefore, most classifiers found beneficial the use of three-grams for predicting and labeling tweets as either "Relevant" or "Not Relevant".

This improvement in most classifiers' performance is reflected in both F1 and F2 scores. Both these metrics represent the relationship between precision and recall in different ways. On the one hand F1 is more the harmonic mean between them and indicates that despite the good recall in relevant tweets, there is difficulty in predicting accurately enough the not relevant tweets. On the other hand F2 is the weighted average of these two metrics and represents in a more complete way the overall performance of each classification algorithm involved in this experiment.

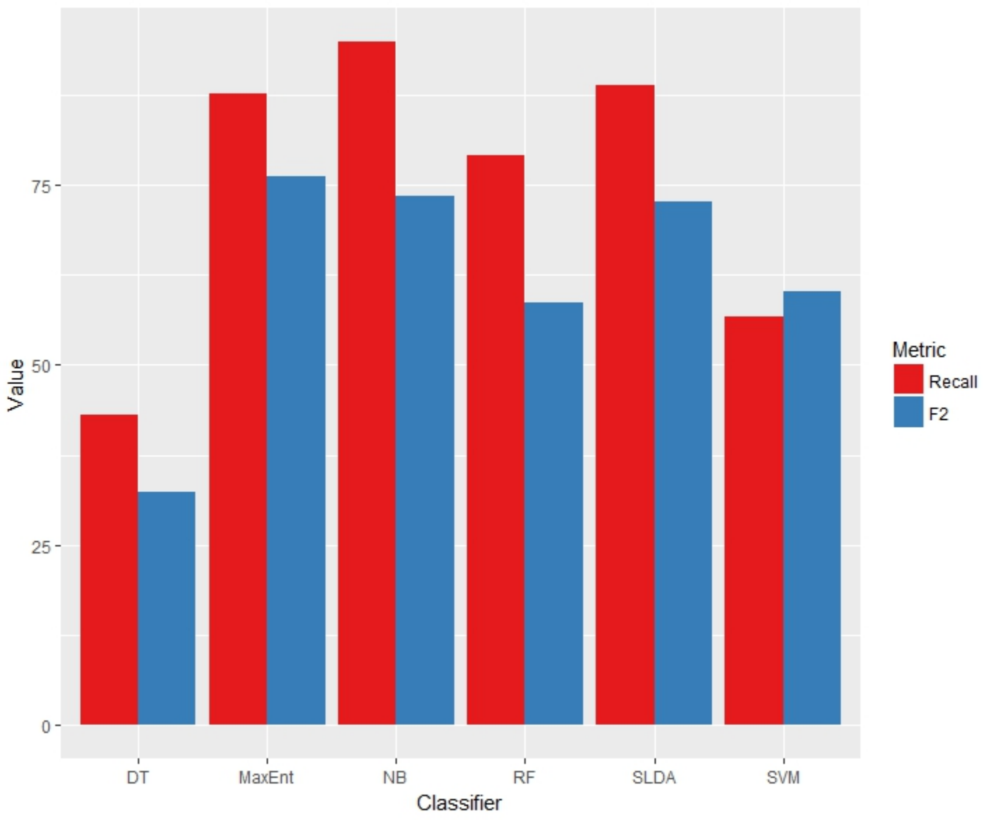

Figure 4.3: Recall and F2 Score of Supervised Classification with 3-grams for "Relevant" tweets

In the above histogram, the performance of these classifiers through recall and F2 Score is shown, regarding the class of interest.

4.4 Partially Supervised Classification

Self Training With Naive Bayes

The experiments of the partially supervised approach followed the KDD process as it was described in the third chapter. The dataset used is the collection of tweets regarding air pollution and breathing problems, which was used in the supervised experiments as well. There are 3500 labeled instances and 1852017 unlabeled. Although for memory reasons and the capabilities of R to handle very large datasets only 12964 instances were used in these experiments. Another reason for making these experiments expensive in memory is high dimensionality because of the way the text is represented. For text representation was used the bag of words approach and term frequency for weighting. The labeled dataset was split into training and test using random sampling with proportions 60% - 40%. Moreover, the training set is totally balanced, hence there is equal amount of tweets representing both "Relevant" and "Not Relevant" classes. Finally, in order to perform partially supervised classification Naive Bayes classifier was used. Two versions of this approach were undertaken. The first one tries to label automatically 500 unlabeled instances in every iteration and the second one 300. Below the results that this method produced are presented along with their interpretation.

Class	**Precision**	**Recall**	**F1**	**F2**
Not Relevant	99.2%	50.7%	67%	56.2%
Relevant	26.4%	97.8%	41.6%	63.5%
	Accuracy: 57.8%			

Table 4.4: Performance metrics of the first model produced by Self Training using Naive Bayes v1

The first model of Naive Bayes in a self training approach, in order to tackle automatic labeling for semi-supervised classification by looking at the accuracy the impression is not good enough. The classifier achieved extremely high recall in relevant tweets and the fairly low on the not relevant ones. This suggests that there were many irrelevant instances misclassified as relevant. The big amount of "noise" in the "Relevant" class brings down the precision and the accuracy of the model. However, extremely high accuracy on the "Not Relevant" class indicates that there are still many true irrelevant tweets predicted. Considering F1 score of not relevant is greater than the relevant tweets it is apparent that the model was more precise at the opposite class from the class of interest. On the other hand F2 has greater score for relevant tweets than not relevant. This represents more efficiently the overall performance of the model of self trained NB.

Class	Precision	Recall	F1	F2
Not Relevant	98.5%	55.8%	71.2%	61%
Relevant	28.1%	95.5%	43.4%	64.5%
	Accuracy: 61.9%			

Table 4.5: Performance metrics of the best model produced by Self Training using Naive Bayes v1

In the next iteration the model got better by considering the increased accuracy of 61.9%. Its precision of not relevant reached 98.5% but the recall of relevant tweets dropped at 95.5%. The increase of accuracy is caused by the decrease of misclassified tweets in irrelevant class, but that change is not so significant in order the model to perform even better. Once again the F1 score of not relevant tweets is greater than the relevant, while for F2 the roles are reversed.

Class	Precision	Recall	F1	F2
Not Relevant	0%	0%	0%	0%
Relevant	15.3%	100%	26.6%	47.5%
	Accuracy: 15.3%			

Table 4.6: Performance metrics of the last model produced by Self Training using Naive Bayes v1

Despite the promising increase of accuracy after the best model that was presented above, the model's performance dropped dramatically by classifying all instances in the "Relevant" class. This unexpected behavior of the classifier is considered overfitting.

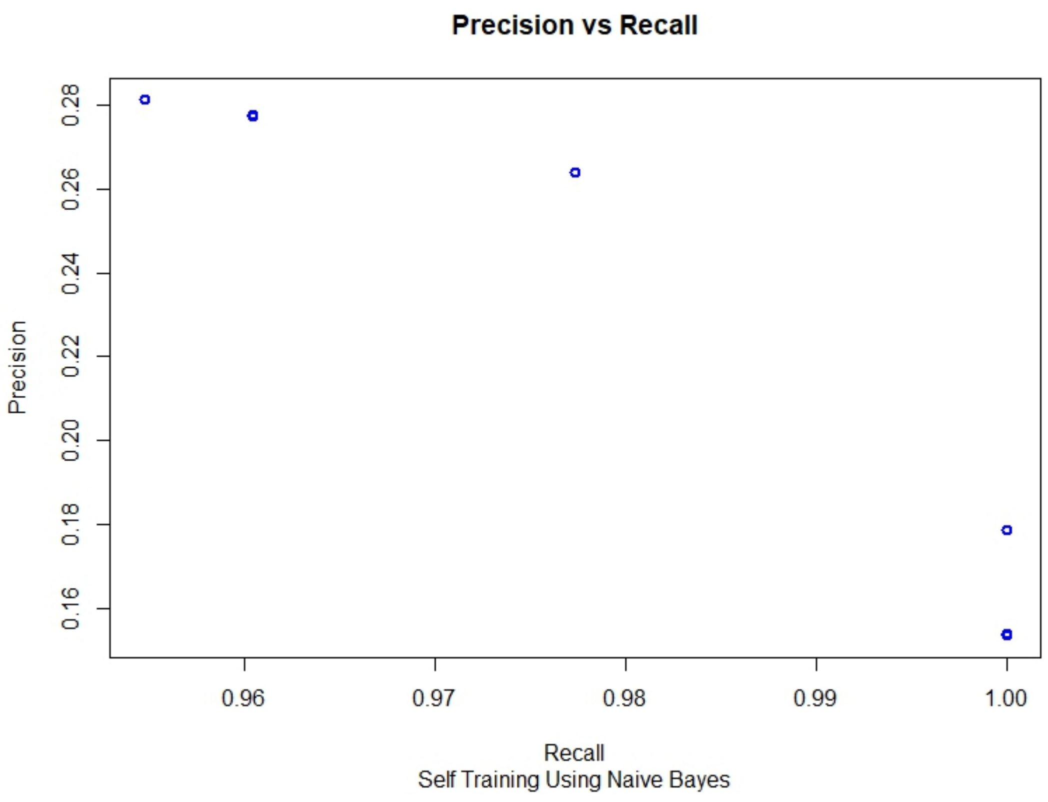

Figure 4.4: Precision vs Recall for “Relevant” v1

In this graph is apparent that as precision falls recall increases. This means that as classification of the relevant instances is getting better, there are even more misclassified not relevant tweets as relevant. This leads to the fact that after a certain iteration all instances are classified as relevant. In other words, the algorithm demonstrates overfitting.

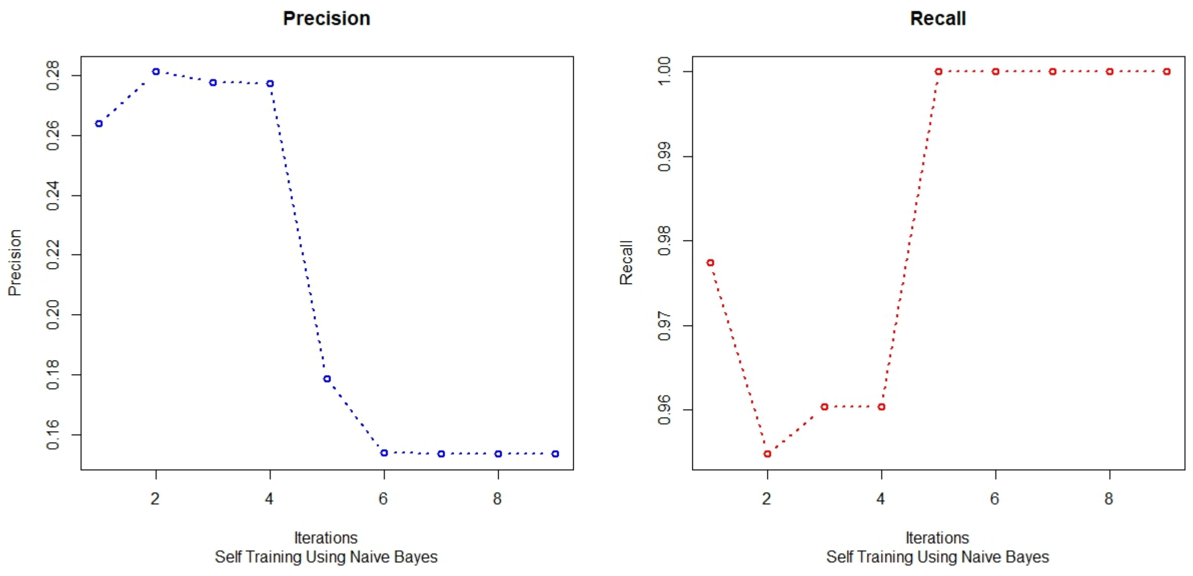

Figure 4.5: Precision and Recall at each iteration for “Relevant” v1

In these graphs are plotted the levels of precision and recall respectively at the first nine iterations, in order to give an alternative view of how self training performs. In the beginning, precision gets better before started having a gradual downward trend. Afterwards it demonstrates a steep fall before overfitting. On the other hand, recall demonstrates a steep fall after the first model and it plateaus at the third and fourth model before presenting overfitting.

Regarding the second version of this experiment, the classifier labels automatically 300 tweets in every iteration. In this experiment, the changes in this methods performance are clearer, because the training set increases with less instances in every iteration. Below the first, the best and the last models are presented and evaluated.

Class	**Precision**	**Recall**	**F1**	**F2**
Not Relevant	98.3%	54%	69.7%	59.4%
Relevant	28.5%	95.2%	43.8%	64.8%
	Accuracy: 60.7%			

Table 4.7: Performance metrics of the first model of Self Training Naive Bayes v2

In the first model, the classifier achieved high recall in relevant tweets which means most of the instances of this class were correctly classified. However, its low precision at the relevant tweets and the low recall of irrelevant suggests that there are a lot of misclassified not relevant tweets. F1 score reflects the relationship of precision and recall in both classes as the harmonic mean. Although F2 score in relevant tweets reflects the overall performance as the weighted average of these metrics. Finally the overall accuracy is low because of the amount of “noise” in the class of interest.

Class	Precision	Recall	F1	F2
Not Relevant	98.2%	55.9%	71.3%	61.3%
Relevant	29.2%	94.6%	44.7%	65.4%
	Accuracy: 62.2%			

Table 4.8: Performance metrics of the best model of Self Training Naive Bayes v2

The best model in this experiment appeared in the fourth iteration. Its recall is slightly lower than the first one but the effort of predicting not relevant tweets is apparent. So, the precision of relevant and recall of not relevant tweets imply that there was a minimal reduction of "noise" in the class of interest. This minimal improvement is reflected in the F1 and F2 scores too.

Class	Precision	Recall	F1	F2
Not Relevant	0%	0%	0%	0%
Relevant	16.1%	100%	27.7%	48.9%
	Accuracy: 16.1%			

Table 4.9: Performance metrics of the last model of Self Training Naive Bayes v2

Despite the signs of improvement in the first few models, the incorrect automatic labeling as the training data are growing is unavoidable. As self training keeps predicting labels for each set of unlabeled instances more predictions are actually wrong. Thus, the models produced after the "best" model demonstrated a downward trend in overall performance until there is a point that overfitting is occurring.

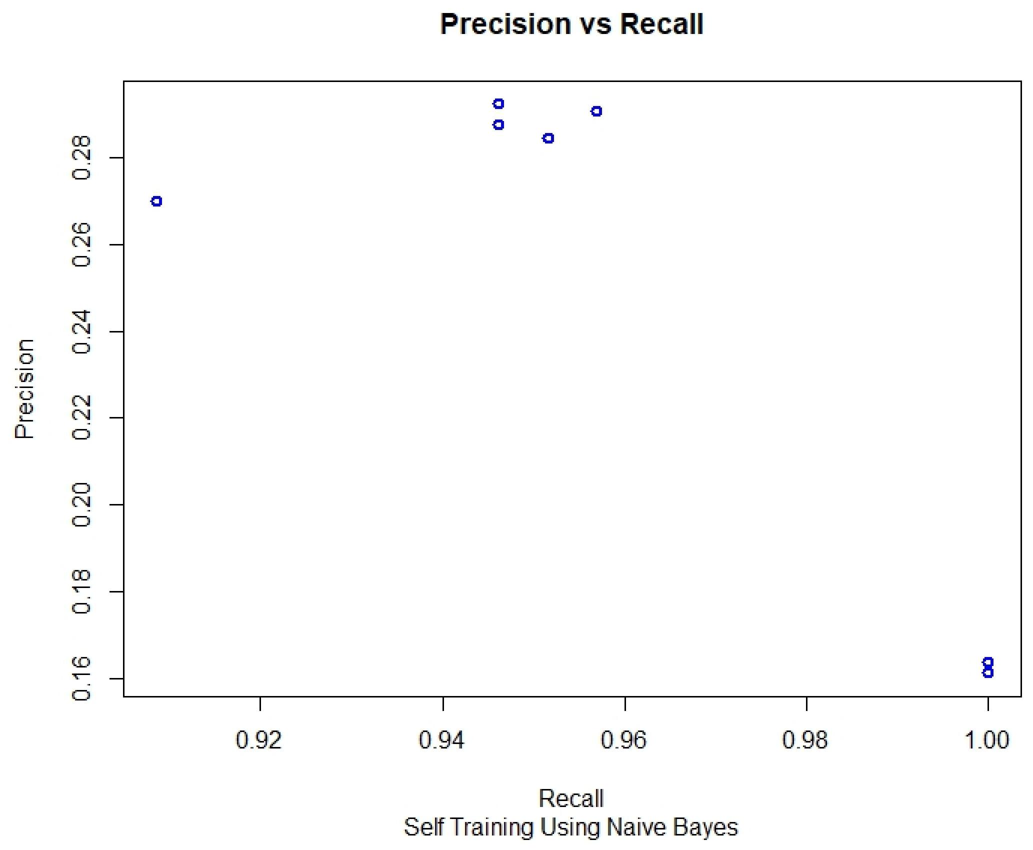

Figure 4.6: Precision vs Recall for “Relevant” v2

In this graph it is demonstrated that as precision increases, recall does not show any specific trend. Comparing tho the previous version of the experiment this time the precision-recall relationship does not follow a particular pattern.

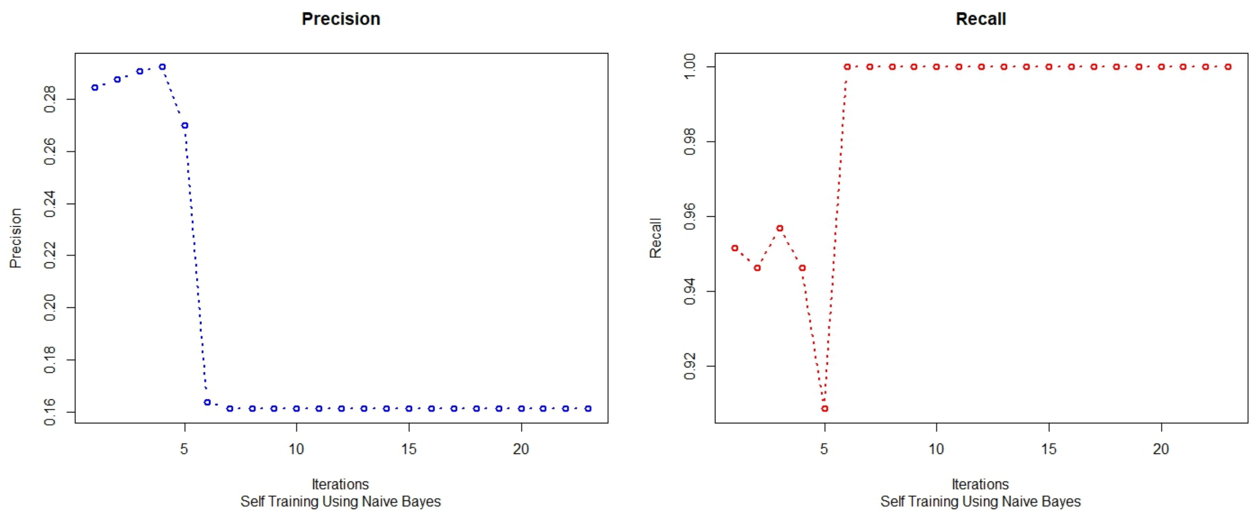

Figure 4.7: Precision and Recall at each iteration for “Relevant” v2

In these two graphs, it is shown how precision and recall progress in each iteration. Initially precision demonstrates an upward trend. After the fourth iteration shows a steep downfall until it starts overfitting. On the other hand, recall begins to show a downward trend in the second iteration but in the third increases until it demonstrates a steep fall before overfitting occurs.

This time the first model was slightly better than the previous effort. This also affected the next models. In the first few iterations this method demonstrates a gradual upward trend regarding overall performance but at some point classifies all the instances of the test set as relevant, which means the classifiers that produces in each iteration are not able to perform automatic labeling anymore. Therefore, the model initially improves as the training set is getting more data from the automatically labeled data but at some point its performance is downgraded due to potentially incorrect automatic labeling.

Chapter 5

Conclusion

Summary

Social media are a massive source of information that may be used for public benefit (Dredze, 2012; Chen et al., 2014). One of the fields that social media data, such as Twitter data, can be useful is public health. This kind of data can be exploited to perform syndromic surveillance by organizations such as Public Health England, which is already developing and operating syndromic systems. Moreover, there are some successful applications that use social media data for various purposes. Therefore, the use of such data can be beneficial to the world for public health monitoring and the evolution of fields like machine learning and data mining.

The data used in this study is drawn from Twitter and can be considered semi-structured. These data were collected for the purpose of syndromic surveillance. In order to accomplish that, it is necessary to perform a classification task to distinguish tweets with metaphorical and literal meaning, in other words the relevant from irrelevant. This study focused on exploring a solution for this issue, by evaluating which classification algorithms and methodologies are able to label each tweet appropriately as “Relevant” or “Not Relevant” and investigate the performance of a partially supervised classification technique in automatic labeling. Therefore, the aim of this project was to investigate a semi-supervised learning approach that can be used for syndromic surveillance.

After performing a literature review in order to be more familiar with text mining and document classification, a KDD process was designed to be used on twitter data and perform classification tasks for the purpose of syndromic surveillance. The dataset used is a collection of tweets about air pollution and breathing problems consisting of 3500 labeled and 1852017 unlabeled tweets. Some of the tweets went through a manual labeling process in order to be able to perform supervised classification. Since this part was completed the data were ready to go through the designed KDD process.

The process begins with loading the data in order to perform cleansing and preprocessing. Firstly the initial corpus was created in order to do text transformations. All tweets were converted to lowercase and the URLs, numbers, punctuation, white spaces and stop words were removed. Secondly, with a stemming process all the words were represented in the form of their root in order to construct a document term matrix. In DTM the text representation methods bag of words and n-grams were used, along with the term weighting methods term frequency and TF-IDF. Furthermore, after the preprocessing stage the data were divided into a training and a testing set using a 60% - 40% split while on the training set was applied balancing. Moreover, well known feature selection and feature engineering methods were tested, but not included in any experiments described. Finally, different combinations were applied and tested performance-wise in different classification tasks.

The algorithms used in the first part of experiments were Naive Bayes, Support Vector Machines, Random Forests, Decision Trees, Scaled Linear Discriminant Analysis and Maximum Entropy. These are classifiers of different concepts and algorithmic processes. Their performances were tested in three different versions of the KDD process. Firstly, it was used the bag of words representation with term frequency. Secondly, the bag of words with TF-IDF took place. Thirdly, there were three-grams with term frequency. Additionally the feature selection methods were tested using the Naive Bayes classifier. Finally, after the completion of those experiments Naive Bayes was chosen as the base classifier for the Self Training algorithm that performed partially supervised classification.

Conclusions of Experimental Results

The first set of experiments was about supervised classification with text represented using the bag of words model and the terms weighted using term frequency. The analysis was focused on the relevant tweets because it is the class of interest. According to the performance of these algorithms NB was chosen as the most appropriate one due to the highest recall. MaxEnt and SLDA were following as the ones with the most correctly classified relevant tweets while RF, DT and SVM achieved lower performance regarding this class. These results suggest that the more recall in the class of interest, the more relevant tweets were correctly classified. Moreover, all the algorithms demonstrated low precision in relevant tweets. This implies that the lower precision in the class of interest, the more misclassified not relevant tweets. Despite the success of some classifiers in predicting labels for relevant tweets, there were a lot of "noise". Therefore, it is necessary to tackle the issue of misclassified tweets.

The second set of experiments involved supervised classifiers using bag of words with TF-IDF and the same set of classification algorithms were tested. In this case, the performance of classifiers got worse compared to the previous set of experiments. This means that the TF-IDF approach on weighting the terms proved not as efficient as term frequency regarding this particular dataset and KDD process. This outcome increased the misclassified not relevant tweets and slightly decreased recall in the "Relevant" class. NB once again achieved the highest recall proving once again its efficiency predicting labels for relevant tweets but also its difficulty recognizing some not relevant ones. Thus, the results of this set of experiments suggests that "noise" is lowering massively how precise these classifiers are, despite the success of predicting relevant tweets and it is necessary to be solved.

The third set of experiments tested the performance of these algorithms using three-grams for text representation along with term frequency. The classifiers performed sightly better this time. In general, recall and precision were improved, implying that more relevant tweets were correctly classified and there were less misclassified not relevant ones. However, the "noise" factor made once again its presence apparent. So, neither a different text representation approach handled this issue and different ways in different parts of the process need to be investigated.

For the partially supervised approach Naive Bayes seem to be the most appropriate because of consistently achieving high levels of recall. The aim of predicting labels for the relevant tweets was achieved. So the chosen classifier can produce a good model regarding the class of interest in order to serve as a base for the self training process.

Although the number of misclassified tweets erodes its performance and that is a factor that affected self training as well. According to the presented results initially the models are getting improved due to the increasing size of the training data. However, at some point in the process results showed that the more data in the training set, the more labels are predicted wrong. Therefore, Self Training Naive Bayes it does improve the initial model but because of inefficient automatic labeling the models predictive abilities become worse.

To sum up, the supervised part of this KDD process apart from indicating the most appropriate algorithm for the self training method from the six that were tested, pointed out the impact of "noise" in the class of interest. Moreover, this issue also affected the performance of self training over time. The produced models even though were getting better there was a point that was demonstrated a steep downward trend in overall performance. Furthermore, the semi-supervised learning approach pointed out that the more data, not only the more misclassified not relevant tweets but also the more correctly classified relevant tweets. Eventually with this approach overfitting seemed unavoidable despite demonstrating brief improvements initially. Therefore, self training is susceptible to overfitting due to incorrect automatic labeling.

Future Work

The future work that can be undertaken in this project is to research how other semi-supervised learning techniques perform, such as co-training, transductive support vector machines, graph-based methods, in order to find a method that is less susceptible to noise and at the same time accurate in predicting relevant tweets. Another way that might be useful in improving classification of relevant and not relevant tweets is to classify whether or not a tweet is posted by a news site or an individual. Also an additional classification task that can be performed is whether a tweet reports health events about its author or another person. Moreover, the use of ensemble methods can be considered in order to exploit the learning and predictive abilities of models trained by the same or multiple algorithms. It is interesting to research way to combine a classifier that predicts precisely relevant tweets and a classifier that predicts well not relevant tweets. Furthermore, feature engineering methods could be explored regarding sentiment analysis and the potential exploitation of emojis and emoticons or any element that can be useful in determining which tweets are actually relevant and improve the accuracy of automatic labeling for syndromic surveillance. Additionally, the use of local search or heuristic algorithms, such as Hill Climbing (Langley et al., 1987) can be investigated in order to select the best instances to consist the training set. Exploring methods, such as Map Reduce, can be beneficial also in order to be able to process large amounts of data. Finally, regarding the implementation of a KDD process such as this one, it is necessary to incorporate software engineering techniques for memory usage optimization such as dynamic memory allocation, garbage collection, dynamic programming and parallel computing. Therefore, in order to overcome the need to process large amounts of data as close to real time as possible or in a insignificant amount of time, several fields of computing science have to join forces. Such fields might be Software Engineering, Information Systems Design along with the already related fields of Artificial Intelligence, Information Retrieval, Natural Language Processing and Statistics.

Bibliography

Aggarwal, C. C. (2015). *Data mining: the textbook*. Springer.

Baker, S., Kiela, D., and Korhonen, A. (2016). Robust text classification for sparsely labelled data using multi-level embeddings. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2333–2343.

Baum, L. E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563.

Bian, J., Topaloglu, U., and Yu, F. (2012). Towards large-scale twitter mining for drug-related adverse events. In *Proceedings of the 2012 International Workshop on Smart Health and Wellbeing*, SHB '12, pages 25–32, New York, NY, USA. ACM.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Chen, L., Hossain, K. T., Butler, P., Ramakrishnan, N., and Prakash, B. A. (2014). Flu gone viral: Syndromic surveillance of flu on twitter using temporal topic models. In *Data Mining (ICDM), 2014 IEEE International Conference on*, pages 755–760. IEEE.

Chunara, R., Andrews, J. R., and Brownstein, J. S. (2012). Social and news media enable estimation of epidemiological patterns early in the 2010 haitian cholera outbreak. *The American journal of tropical medicine and hygiene*, 86(1):39–45.

Church, K. W. (1988). A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, ANLC '88, pages 136–143, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cornfield, J. (1951). A method of estimating comparative rates from clinical data. applications to cancer of the lung, breast, and cervix. *Journal of the National Cancer Institute*, 11(6):1269–1275.

Culotta, A. (2013). Lightweight methods to estimate influenza rates and alcohol sales volume from twitter messages. *Language resources and evaluation*, 47(1):217–238.

Damerau, F. J. (1971). *Markov models and linguistic theory: an experimental study of a model for English.* Number 95. Mouton De Gruyter.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.

DeRose, S. J. (1988). Grammatical category disambiguation by statistical optimization. *Comput. Linguist.*, 14(1):31–39.

DeRose, S. J. (1990). *Stochastic Methods for Resolution of Grammatical Category Ambiguity in Inflected and Uninflected Languages.* PhD thesis, Providence, RI, USA. UMI Order No: GAX90-02217.

Dredze, M. (2012). How social media will change public health. *IEEE Intelligent Systems*, 27(4):81–84.

Feldman, R. and Sanger, J. (2007). *The text mining handbook: advanced approaches in analyzing unstructured data.* Cambridge university press.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of human genetics*, 7(2):179–188.

Fix, E. and Hodges Jr, J. L. (1951). Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, California Univ Berkeley.

Fourli-Kartsouni, F., Slavakis, K., Kouroupetroglou, G., and Theodoridis, S. (2007). A bayesian network approach to semantic labelling of text formatting in xml corpora of documents. *Universal Access in Human-Computer Interaction. Applications and Services*, pages 299–308.

Freund, Y. and Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, (6):721–741.

Ginsberg, J., Mohebbi, M. H., Patel, R. S., Brammer, L., Smolinski, M. S., and Brilliant, L. (2009). Detecting influenza epidemics using search engine query data. *Nature*, 457(7232):1012–1014.

Han, J., Pei, J., and Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

Hawkins, D. M. (2004). The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1):1–12.

Heaivilin, N., Gerbert, B., Page, J., and Gibbs, J. (2011). Public health surveillance of dental pain via twitter. *Journal of dental research*, 90(9):1047–1051.

Ho, T. K. (1995). Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278–282. IEEE.

Iwamura, M., Tsukada, M., and Kise, K. (2013). Automatic labeling for scene text database. In *2013 12th International Conference on Document Analysis and Recognition, Washington, DC, USA, August 25-28, 2013*, pages 1365–1369.

Jaynes, E. T. (1957a). Information theory and statistical mechanics. *Physical review*, 106(4):620.

Jaynes, E. T. (1957b). Information theory and statistical mechanics. ii. *Physical review*, 108(2):171.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *ICML*, volume 99, pages 200–209.

Kolesov, A., Kamyshenkov, D., Litovchenko, M., Smekalova, E., Golovizin, A., and Zhavoronkov, A. (2014). On multilabel classification methods of incompletely labeled biomedical text data. *Comp. Math. Methods in Medicine*, 2014:781807:1–781807:11.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Lamb, A., Paul, M. J., and Dredze, M. (2013). Separating fact from fear: Tracking flu infections on twitter. In *HLT-NAACL*, pages 789–795.

Langley, P., Gennari, J., and Iba, W. (1987). Hill climbing theories of learning. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 312–323.

Liu, B., Lee, W. S., Yu, P. S., and Li, X. (2002). Partially supervised classification of text documents. In *ICML*, volume 2, pages 387–394. Citeseer.

Liu, T., Du, X., Xu, Y.-D., Li, M., and Wang, X. (2011). Partially supervised text classification with multi-level examples. In *AAAI*.

Luhn, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317.

Maron, M. E. (1961). Automatic indexing: An experimental inquiry. *J. ACM*, 8(3):404–417.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.

Nagy, G. and Shelton, G. (1966). Self-corrective character recognition system. *IEEE Transactions on Information Theory*, 12(2):215–222.

Nassara, E. I. G., Grall-Mas, E., and Kharouf, M. (2016). Linear discriminant analysis for large-scale data: Application on text and image data. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 961–964.

Nigam, K., Lafferty, J., and McCallum, A. (1999). Using maximum entropy for text classification. In *IJCAI-99 workshop on machine learning for information filtering*, volume 1, pages 61–67.

O'Connor, B., Balasubramanyan, R., Routledge, B. R., and Smith, N. A. (2010). From tweets to polls: Linking text sentiment to public opinion time series. *ICWSM*, 11(122-129):1–2.

Paul, M. J. and Dredze, M. (2011). You are what you tweet: Analyzing twitter for public health. *Icwsm*, 20:265–272.

Paul, M. J. and Dredze, M. (2012). A model for mining public health topics from twitter. *Health*, 11:16–6.

Porter, M. F. (2001). Snowball: A language for stemming algorithms.

Public Health England (2017). Syndromic surveillance: systems and analyses. https://www.gov.uk/government/collections/syndromic-surveillance-systems-and-analyses.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1):81–106.

Ratsaby, J. and Venkatesh, S. S. (1995). Learning from a mixture of labeled and unlabeled examples with parametric side information. In *Proceedings of the eighth annual conference on Computational learning theory*, pages 412–417. ACM.

Sakaki, T., Okazaki, M., and Matsuo, Y. (2010). Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM.

Schwenker, F. and Trentin, E. (2014). Pattern classification and clustering: A review of partially supervised learning approaches. *Pattern Recognition Letters*, 37:4–14.

Scudder, H. (1965). Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.

Shahshahani, B. M. and Landgrebe, D. A. (1994). The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon. *IEEE Transactions on Geoscience and remote sensing*, 32(5):1087–1095.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Tan, P.-N., Steinbach, M., and Kumar, V. (2005). *Introduction to data mining*. Pearson Addison Wesley.

Torgo, L. (2016). *Data mining with R: learning with case studies*. CRC press.

Vapnik, V. and Chervonenkis, A. (1964). A note on one class of perceptrons. *Automation and remote control*, 25(1):103.

Vapnik, V., Chervonenkis, A. Y., and Moskva, N. (1974). Pattern recognition theory. *Statistical Learning Problems*.

Witten, I. H., Bray, Z., Mahoui, M., and Teahan, B. (1999). Text mining: A new frontier for lossless compression. In *Data Compression Conference, 1999. Proceedings. DCC'99*, pages 198–207. IEEE.

Witten, I. H., Don, K. J., Dewsnip, M., and Tablan, V. (2004). Text mining in a digital library. *International Journal on Digital Libraries*, 4(1):56–59.

Witten, I. H., Frank, E., Hall, M. A., and Pal, C. J. (2016). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics.

Zhu, D. and Wong, K. W. (2014). Text categorization using an automatically generated labelled dataset: An evaluation study. In *Neural Information Processing - 21st International Conference, ICONIP 2014, Kuching, Malaysia, November 3-6, 2014. Proceedings, Part I*, pages 479–486.

Appendix A

Aggregate Performance Tables

A.1 Supervised Classifiers Using Term Frequency

Class	Accuracy	Precision	Recall	F1	F2
Naive Bayes					
Not Relevant	72.2%	98%	67.9%	80%	72.4%
Relevant	72.2%	37%	93.2%	53%	71.5%
Support Vector Machines					
Not Relevant	88.8%	91.2%	95.8%	93.5%	94.9%
Relevant	88.8%	72.7%	54.5%	62.3%	57.4%
Random Forests					
Not Relevant	65.6%	92.6%	63.7%	75.5%	67.9%
Relevant	65.6%	29.5%	74.9%	42.3%	57.3%
Decision Trees					
Not Relevant	55.5%	87.9%	54%	66.9%	58.5%
Relevant	55.5%	21.9%	63.4%	32.5%	46%
Scaled Linear Discriminant Analysis					
Not Relevant	74.5%	95.6%	72.9%	82.6%	76.4%
Relevant	74.5%	38.3%	83.3%	52.4%	67.4%
Maximum Entropy					
Not Relevant	80.4%	96.3%	79.5%	87%	82.4%
Relevant	80.4%	45.7%	84.8%	59.4%	72.4%

Table A.1: Performance metrics of Classifiers using Term Frequency

A.2 Supervised Classifiers Using TF-IDF

Class	Accuracy	Precision	Recall	F1	F2
Naive Bayes					
Not Relevant	50.9%	97.6%	44.4%	61%	49.8%
Relevant	50.9%	20.6%	93%	33.7%	54.6%
Support Vector Machines					
Not Relevant	91.2%	94.4%	95.4%	94.9%	95.2%
Relevant	91.2%	68.6%	63.8%	66.1%	64.7%
Random Forests					
Not Relevant	60.3%	97.6%	55.5%	70.8%	60.7%
Relevant	60.3%	24.2%	91.4%	38.2%	58.7%
Decision Trees					
Not Relevant	43.4%	88.3%	39.9%	55%	44.8%
Relevant	43.4%	14.5%	65.9%	23.8%	38.6%
Scaled Linear Discriminant Analysis					
Not Relevant	73.8%	96.5%	72.3%	82.7%	76.2%
Relevant	73.8%	31.9%	83.5%	46.1%	63.1%
Maximum Entropy					
Not Relevant	75.6%	97.7%	73.6%	83.9%	77.4%
Relevant	75.6%	34%	88.7%	43.1%	67.1%

Table A.2: Performance metrics of Classifiers using Term Frequency-Inverse Document Frequency

A.3 Supervised Classifiers Using Threegrams

Class	Accuracy	Precision	Recall	F1	F2
Naive Bayes					
Not Relevant	72.7%	98.4%	68.1%	80.5%	72.6%
Relevant	72.7%	38.5%	94.9%	54.8%	73.4%
Support Vector Machines					
Not Relevant	89.9%	91.4%	96.8%	94%	95.7%
Relevant	89.9%	79.4%	56.8%	66.2%	60.2%
Random Forests					
Not Relevant	62.5%	93%	59%	72.2%	63.6%
Relevant	62.5%	28.9%	79.1%	42.3%	58.7%
Decision Trees					
Not Relevant	51.5%	81.5%	52.8%	64.1%	56.8%
Relevant	51.5%	16.1%	43.1%	23.5%	32.3%
Scaled Linear Discriminant Analysis					
Not Relevant	77%	96.9%	74.5%	84.2%	78.1%
Relevant	77%	42.3%	88.8%	57.3%	72.8%
Maximum Entropy					
Not Relevant	82.6%	96.9%	81.6%	88.6%	84.2%
Relevant	82.6%	50.1%	87.8%	63.8%	76.3%

Table A.3: Performance metrics of Classifiers using 3-grams

A.4 Supervised Classification & Feature Selection

For this set of experiments it was decided to use Naive Bayes due to his high performance on the class of interest. The dataset used is about air pollution and breathing problems represented using bag of words with term frequency. The 60% of the data consists the training set and the 40% the test, while the training set has the balancing approach discussed in the third chapter. In the following table are presented the results of Naive Bayes using IG, OR and Chi-Square on the training set to select the most important features with threshold 1%.

Naive Bayes					
Method	**Accuracy**	**Precision**	**Recall**	**F1**	**F2**
Chi-Square	55.9%	24.6%	67.8%	36.1%	50.1%
Information Gain	72.7%	38.5%	94.9%	54.8%	73.4%
Odds Ratio	55.9%	24.6%	67.7%	36.1%	50.1%

Table A.4: Performance of Naive Bayes on the "Relevant" class using Feature Selection methods

NB in the case of IG performed slightly better than the simple bag of words and term frequency approach. There is an increase in all metrics but not significant. On the other hand, surprisingly chi-square and OR demonstrated identical performances. In both cases, all metrics are low, indicating the model's poor predictive abilities. Their accuracy is slightly greater than a random guess. Therefore, IG's feature selection contributed in the improvement of the models predictions but OR and chi-square was unsuccessful.

Appendix B

Memory Usage of Self Training

A huge issue regarding the self training process was the memory it requires to perform classification tasks iteratively. The algorithm starts with a totally balanced training set and the first model is trained in a supervised manner. Since the first model is trained and tested, a certain amount of data from the unlabeled set will have predicted labels with confidence probability greater than 0.95 which are joining the training set. Therefore the training set in every iteration grows continuously and along with its size the memory needed and processing time also increases.

CPU Time	54135.05 sec
Max Memory	58197 MB
Average Memory	54242.61 MB
Total Requested Memory	60000.00 MB
Delta Memory	1803.00 MB
Max Swap	74553 MB
Max Processes	5
Max Threads	6

Table B.1: Resource Usage Summary of Self Training Naive Bayes 1

The table above provides information about the use of the UEA high performance computing cluster. According to these information provided by the system the processing time was more than 15 hours but due to memory limit the process stopped at 60 gigabytes of RAM, which is the maximum memory size that was requested. This means that in order to finish the process the cluster would need probably twice as much time and memory. Moreover, the average memory that this algorithm used was approximately 54 gigabytes and the difference between total requested memory and maximum memory used was 1.8 gigabytes, which is called delta memory. Finally, the experiment was repeated requesting more RAM. The information provided by the system are provided in the following table:

CPU Time	377421.16 sec
Max Memory	346136 MB
Average Memory	118924.54 MB
Total Requested Memory	500000.00 MB
Delta Memory	153864.00 MB
Max Swap	356092 MB
Max Processes	5
Max Threads	6

Table B.2: Resource Usage Summary of Self Training Naive Bayes 2