Multi-Topic Sentiment Analysis

Pedro Samuel Amaro Coelho
Dissertação de Mestrado apresentada à
Faculdade de Ciências da Universidade do Porto em
Ciência de Computadores
2013
Multi-Topic Sentiment Analysis

Pedro Samuel Amaro Coelho
Mestrado em Ciência de Computadores
Departamento de Ciência de Computadores
2013

Orientador
Professor Doutor Luís Fernando Rainho Alves Torgo
Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, _____/_____/_______
Dedicated to the love of my life Diana, to my Family and to God
I would like to thank Professor Luis Torgo the opportunity to work in a large European project. The knowledge I have acquired is invaluable and the challenges that we have faced proved to be a very enriching experience to my career.

I would also like to thank the support of the EU Seventh Framework e-Policy project (grant agreement 288147) and its members for integrating me and providing me with all the resources I needed to carry out this thesis.
Abstract

The work carried out in this thesis belongs to the field of Opinion Mining or Sentiment Analysis. The main goal of Opinion Mining is to infer the opinion or sentiment expressed in textual documents. The problems addressed in this thesis were motivated by the participation in an European research project (e-Policy), where one of the tasks was to develop tools that allow to infer the sentiment of a population concerning a series of alternative energy policies. In this context, it was necessary to gather the relevant data, store these data, provide a way for labelling the data, analyse it and infer the expressed sentiment concerning a pre-defined set of topics. The main goal of the work in this thesis is to study models that are able to infer the sentiment concerning a set of pre-defined topics expressed on textual documents. Our data are texts that may express possibly different opinions regards some topics. Our goal is to infer the sentiment score concerning each of these topics on these texts. To reach our goal we have studied several alternative approaches to obtain these sentiment scores. Moreover, we have compared some of these alternatives on real world data. The main conclusions of our work are that the approaches studied are a good starting basis and provide some interesting results even with a low amount of data pre-labelled by human experts. Given these results we expect that with further human-labelled texts, even better labelling of new text is achievable.
Resumo

O trabalho desenvolvido nesta tese pertence à área de Opinion Mining ou Sentiment Analysis. O principal objetivo do Opinion Mining é inferir a opinião ou sentimento expresso em documentos textuais. Os problemas tratados nesta tese foram motivados pela participação num projeto Europeu de investigação (e-Policy), onde uma das tarefas foi o desenvolvimento de ferramentas que permitem inferir o sentimento de uma população no que diz respeito a uma série de políticas energéticas alternativas. Neste contexto, foi necessário recolher os dados relevantes, guardar os dados, fornecer uma maneira de etiquetar os dados, analisar os dados e inferir o sentimento expresso no que diz respeito a um conjunto de tópicos pré-definidos. O principal objetivo do trabalho nesta tese é o estudo de modelos que são capazes de inferir o sentimento, no que diz respeito a um conjunto de tópicos pré-definidos, expresso em documentos textuais. Os nossos dados são textos que podem exprimir, possivelmente, diferentes opiniões no que diz respeito a alguns tópicos. O nosso objetivo é inferir a pontuação do sentimento no que diz respeito a cada um destes tópicos nestes textos. Para alcançar o nosso objetivo estudamos algumas abordagens alternativas para obter a pontuação do sentimento. Além disso, comparamos algumas destas alternativas em dados reais. As principais conclusões do nosso trabalho são que as abordagens estudadas são um bom ponto de partida e fornecem alguns resultados interessantes, mesmo com uma baixa quantidade de dados etiquetados por um especialista. Dados estes resultados, esperamos que com mais textos etiquetados por um especialista, uma melhor etiquetagem dos dados seja alcançável.
Contents

Abstract ................................................. 5

Resumo .................................................. 6

List of Tables .......................................... 10

List of Figures ........................................ 12

1 Introduction ......................................... 13
   1.1 Motivation ........................................... 13
   1.2 Problems and Objectives ......................... 15
   1.3 Thesis structure .................................... 16

2 State of the Art on Opinion Mining ................. 17
   2.1 Motivation for Opinion Mining ................... 17
   2.2 Formalization of the Task of Opinion Mining .... 19
   2.3 Main Challenges ................................... 21
   2.4 Main Approaches .................................. 25

3 Sentiment Analysis using Text Mining ............. 28
   3.1 Problem Formalization ............................. 28
List of Tables

2.1 Types of features used to describe texts [1]. ........................................... 22
2.2 Approaches and results with various techniques [40]. ............................. 24
2.3 Negation phrase results [40]. ................................................................. 25
2.4 MAP scores of 5 methods on all TREC queries [28]. .............................. 25

3.1 Total Cost matrix. ................................................................. 36

4.1 ePolicy Dataset composition. ............................................................... 42
4.2 Best performing models. ................................................................. 51
4.3 Worst performing models. ................................................................. 51
4.4 Statistical significance of the observed differences. ............................. 53

A.1 Random Forests parameters. ............................................................. 59
A.2 Support Vector Machines parameters. ............................................... 60
A.3 Neural Networks parameters. ............................................................ 61
List of Figures

2.1 Emoto [8]. ................................................................. 18
2.2 Emoto medals topic [8]. ............................................... 19
2.3 Tasks for opinion mining and its relationship with related areas [31]. .... 19
2.4 Opinion summarization system [27]. .................................. 27

4.1 e-Policy project system [15]. ......................................... 38
4.2 Energetic Ambient front page [16]. ..................................... 40
4.3 Energetic Ambient forum. ................................................ 41
4.4 Newclear blog [17]. ....................................................... 42
4.5 Number of tagged posts per day. ....................................... 43
4.6 Number of untagged posts per day. .................................... 43
4.7 Number of tagged posts per week. .................................... 44
4.8 Number of untagged posts per week. .................................. 44
4.9 Number of tagged posts per month. .................................. 44
4.10 Number of untagged posts per month. ............................... 44
4.11 Number of tagged posts per year. .................................... 45
4.12 Number of untagged posts per year. ................................ 45
4.13 Number of tagged posts per topic. ................................... 45
4.14 Economic aspects score distribution through time. ................. 45
4.15 Environmental aspects score distribution through time. ............ 46
4.16 Technological aspects score distribution through time. .................. 46
4.17 Positive documents wordcloud. ........................................... 47
4.18 Negative documents wordcloud. ......................................... 47
4.19 The performance of the best RF; SVM, NN and the baseline model. ....... 54

B.1 R Project. ................................................................. 62
B.2 Python Scrapy. ............................................................ 62
B.3 Python Django. ............................................................ 62
B.4 Tagging a post. ............................................................. 64
B.5 Search page. ............................................................... 64
B.6 Database schema. .......................................................... 65
Chapter 1

Introduction

Data Mining has become a popular research and application field because it allows the automatic extraction of useful information from data collected in a certain domain. Given that nowadays the amount of data that is collected from most human activities is increasing at a very high rate, the need for automatic analysis of this data is also very high. The work described in this thesis belongs to the area of Opinion Mining, a particular task of Text Mining a sub-field of Data Mining. This thesis is motivated by the specific goals of a task in a large European research project. This task involves inferring the sentiment of a population concerning a series of energy-related topics from posts on a set of e-participation sites.

In this chapter we start by describing the motivation of this thesis. Next we describe the problem addressed in this work as well as the main objectives of the thesis. Finally, we describe the thesis structure.

1.1 Motivation

Data Mining is a research field whose main goal is to uncover useful and unknown patterns on data. This field includes many techniques from different research fields such as machine learning and statistics and has applications in many areas such as medicine, engineering and politics. Data mining tasks are usually described by the CRISP-DM [48] model that divides a data mining project into the following major steps:

- Business Understanding
- Data Understanding
CHAPTER 1. INTRODUCTION

- Data Preparation
- Modelling
- Evaluation
- Deployment

The first steps of a data mining project usually involve actions such as data collection, preparation and visualization. Still, a key central issue on data mining is the automatic analysis of raw data in order to extract unknown patterns such as groups (cluster analysis), unusual records (outlier detection), or dependencies (association rule mining or prediction models). For example, a supermarket can use data mining techniques in order to find patterns on the consumers purchases and then take actions according to this uncovered information (such as placing the product $X$ closer to product $Y$ to reinforce some discovered buying association).

Opinion Mining encapsulates a series of techniques that are part of a broader area known as Text Mining, which again can be regarded as a sub-field of Data Mining. The main distinguishing feature of text mining with respect to general data mining lies on the type of data that is used - text documents. Opinion mining further focus the type of data analysis that is carried out by having as main objective to discover the opinions expressed within these text documents concerning a series of topics.

As we have mentioned text mining is a sub-field of data mining and thus shares many of its steps and processes. In text mining the goal is to extract useful information from text documents written in some language. For example, we can have as goal to assign documents to categories (document categorization), find groups in a large set of documents (document clustering) or find the sentiment expressed on the document (sentiment analysis or opinion mining).

With the amount of data available nowadays, especially with the advent of social networks, it became interesting to be able to categorize and extract information of people’s opinion and thoughts about a large amount of topics. This analysis would be impractical to do manually because of the unthinkable amount of human resources that would be needed to do it.

This thesis was motivated by a concrete practical problem in the context of one of the tasks of an European research project. This problem has to do with inferring the sentiment of a population concerning a series of alternative energy policies. Given this goal, the project aims to infer this opinion by allowing the population to express its sentiment concerning a set of topics on a series of e-participation sites. Our goal is thus to crawl these sites for new posts and from this textual data to infer the sentiment of the population concerning a series of energy-related topics.
1.2 Problems and Objectives

The concrete application problem that motivates the work in this thesis can be regarded as an instance of a more general opinion mining task. Namely, the overall problem being addressed in this thesis consists on the study of opinion mining techniques/models that are able to infer the sentiment score regards a set of topics on textual documents.

From a data mining perspective this is a prediction task. Prediction tasks are one of the most frequent tasks that are addressed by data mining techniques. The general idea is that we try to infer the value of a certain variable (known as the target variable) from the values of other variables (known as the predictors). This prediction is done by a model that is obtained based on historical examples where we know both the values of the predictors and of the target variable. Based on this historical data set a prediction model is induced that can then be used to forecast the value of the target variable for new instances of the same problem. In our concrete problem the predictor variables are obtained from the textual documents that are the data available for the problem at hands. We will see that several methods exist that allow to represent the information on a text by a series of variables. In our target problems the target variables represent the sentiment scores. The general idea is that we assume that the sentiment score is a function of the information in the text. The goal of the models is to induce the "shape" of this function.

In spite of the similarity of the general problem of sentiment analysis and standard data mining prediction tasks that we have just described, there are some particularities in the problem we address in this thesis. Namely, in our target problems each text may express sentiment on more than one topic. Again, from a data mining perspective, this is known as a multi-objective prediction task, i.e. a task where we want to predict the value of more than one target variable at the same time from the values of the predictor variables.

A possible way of looking at the problem being addressed in this thesis is the following. Given a text document and a set of target topics for which we want to infer the sentiment we need to: i) decide whether the document talks about each of the topics; and ii) if it mentions them, then we need to infer what is the sentiment score on a pre-defined scale. More formally, given a set of documents \( D = \{d_1, d_2, \ldots, d_n\} \) and a pre-defined set of topics \( T = \{t_1, t_2, \ldots, t_q\} \), we want to infer for each \( d_i \) whether it talks about each of the topics (i.e. a binary decision) and if it talks, with which sentiment in a pre-defined scale.
1.3 Thesis structure

In Chapter 2 we start with the study of the state of the art in opinion mining. In this chapter we study the problems and solutions that currently exist to handle the main challenges in opinion mining.

In Chapter 3 we describe the specific problem addressed in this thesis and study some of the possible solutions for it.

Chapter 4 describes the concrete real world problem driving the work of this thesis as well as the results that were obtained with the proposed solutions.

Chapter 5 provides a technical description of the tools and work carried out to solve the problems addressed in the thesis.

Finally, we present the conclusions of the thesis and outline some possible future work.
Chapter 2

State of the Art on Opinion Mining

The general area of Text Mining as well as the Opinion Mining sub-field have been the object of an increasing interest of both academia and industry players. This is motivated by the wide range of potential applications of the techniques that are studied in these research fields.

In this chapter we provide a general overview of the state of the art on Opinion Mining. We start by providing some motivations for this field using concrete examples of its applicability. We then talk about text data representation approaches and finish with a brief overview of the main approaches to opinion mining or sentiment analysis.

2.1 Motivation for Opinion Mining

With the widespread use of the Web different tools of e-participation have significantly increased the possibilities for users to express their opinions concerning any topic. With the Web users have access to a large set of tools that allow them to post text messages that may contain opinions on products, people, policies, and many other topics. This user-generated content may contain relevant information on the general sentiment of the population concerning a certain issue. Knowing what people think about some issue is of key importance to decision-making. The main goal of opinion mining is the computational study of opinions, sentiments and emotions expressed through text [34].

Extracting information from comments that are written by people can be valuable in many application contexts. There are many blogs and forums across the internet talking about every topic one can think of and if this information is gathered and analysed we can start to
understand how a certain community reacts to certain events and even try to predict their reactions to future events based on their behaviour history. All these inferences are of key importance to decision and policy makers.

Moreover, with the advent of social networks and their proliferation, several of these tools have turned into invaluable sources of information on users’ opinions and trends of these opinions. Twitter, for example, is a great source of information where opinions about many topics can be extracted and analysed (e.g. [41, 49]). More and more people use these tools to express their opinion/sentiment regards a large variety of topics. As an illustrative example, during the London Olympics 2012, there were spikes of 37000 tweets per minute [3] when Michael Phelps achieved a record number of Olympic medals. Emoto [8] is an example of a system that uses information extraction and opinion mining techniques applied to messages in Twitter: "The emoto project captures and visualises the excitement around the Olympic Games in London 2012. We track twitter for themes related to the Games, analyse the messages for content and emotional expressions, and visualize topics and tone of the conversation.". Figure 2.1 gives a visual overview of the system, while in Figure 2.2 we can see the overall sentiment towards the medal topic in each competition day, with the hot colours representing positiveness and the cold colours representing negativeness.

![Figure 2.1: Emoto [8].](image-url)
2.2 Formalization of the Task of Opinion Mining

Analysing a text document with the goal of inferring users’ opinions involves several tasks that are related with several scientific domains, from standard statistical analysis to computational linguistics. Figure 2.3 [31] shows some of these main tasks and their relationship with related areas.
Independently of the approaches that are followed the general goal of opinion mining can be seen as the search for models of the unknown function that relates opinions with text content, i.e.,

\[ \text{opinion} = f(\text{text}) \]

Solving this problem involves starting by defining what is an opinion and how are we going to describe a text.

There are different ways of characterizing the opinion of a user. The most common is to classify it as either positive or negative. However, there are other possibilities that involve some kind of degree or score of positivity or negativity, like for instance the rather common star rating systems (e.g. 1-5 stars). These different approaches to opinion scoring have impact on the modelling techniques that are used to infer the users' opinions. While the first approach can be regarded as a binary classification task, the second approach involves more metric approaches (e.g. regression) as we have a scoring system where we can define a distance between the different possible values (e.g. 5 stars is "nearer" to 4 stars than to 1 star).

Regarding the way one decides to represent the information on a text document this can be seen as the central issue in opinion mining and text mining in general. In effect, the most common approaches to these problems involve finding some representation which ensures that sufficient information for inferring the opinion expressed in the text is included, but also that the resulting representation allows the use of standard "of-the-shelf" data mining tools. Because of this latter point one of the most basic, but most used approaches to text representation, is the Bag of Words (BOW) representation that simply represents a text by a (large) set of presence/no-presence binary features, one for each possible word in the used language. This type of approach, though apparently naive in the sense that all text structure is lost, often achieves surprisingly good results particularly in the presence of a very large text corpus. Obviously, much more sophisticated text representations were proposed with the goal of providing the models with other useful information on the text structure.

The modelling problem that we have just outlined can become slightly more complex if one allows for a document to address more than one topic, each with potentially different opinion/sentiment scores. Moreover, extra difficulties often arise due to the lack of a large set of documents previously labelled by a domain expert that can serve as a training set for obtaining the models. Both these situations are rather frequent in real world applications of opinion mining and may require specific techniques like multi-label classification (e.g. [45]) and semi-supervised learning (e.g. [58]).

In summary, opinion mining and sentiment analysis are areas with a strong application
relevance, that require the use of different techniques and methodologies with the goal of inferring the opinion of users concerning a topic or set of topics.

2.3 Main Challenges

Pang and Lee [42] present an excellent survey on opinion mining and sentiment analysis. According to these authors we can categorize the work in these areas as either performing classification or summarization of documents. In classification the general goal is to try to attach some form of label or score to a document that is a function of the sentiment that it expresses. In document summarization, the main goal is to try to somehow aggregate and summarize the main arguments that are present in a document and lead to a certain opinion/sentiment.

Inferring the sentiment or opinion on a text document is difficult for several different reasons. While one may think that the presence of several linguistic queues (like for instance words like "great", "happy" and "sad") may facilitate this task, the fact is that even these queues may be misleading if context is not taken into account. For instance, in the following sentence [30]

"not bad, well crafted stationery and with the country going through a recession very wise and economical. very good, president obama."

the word bad can induce a model to classify the sentence as negative when in reality it is positive because of the surrounding word "not". Approaching this contextualization problem by looking at 1-3 preceding words and inverting the sentiment has not shown any substantial improvements [30].

The task is even harder when we are looking for the strength of the sentiment present in a document [50] and not only its direction (positive or negative).

Irony detection [9] is another fundamental tool/difficulty if we want to achieve good results since it is frequently used and hard to detect.

Other difficulties arise from the impact the structure of the phrases has on the opinion score they contain [47].

As we have mentioned in the previous section two fundamental issues for addressing opinion mining are the way one labels a document and the form of representing the information on that document. This last issue is particularly relevant as it affects both the performance of the models and the time required to create and evaluate them. We must also take
into account that most of the research done is based on English only documents. The problem of text document representation has to do with the selection of the “features” we use to describe the document, i.e. from a data analysis perspective the variables we use to describe each observation (an observation being a text document in this context). Table 2.1, used by Ahmed Abbasi [1], shows examples of features that we can use in order to create a feature vector representing a text document.

Most of the used text representation strategies will lead to a large set of features. This may cause problems to most modelling algorithms, particularly if the number of features is not significantly less than the number of texts (observations). In this context, selecting a subset of these features is a frequent step in text mining projects. Several strategies of feature selection exist from simple algorithms to more sophisticated approaches such as EWGA (Entropy Weighted Genetic Algorithm) [1].

Table 2.1 Types of features used to describe texts [1].

<table>
<thead>
<tr>
<th>Category</th>
<th>Feature Group</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntactic</td>
<td>POS N-grams</td>
<td>frequency of part-of-speech tags (e.g., NP VB)</td>
</tr>
<tr>
<td></td>
<td>Word Roots</td>
<td>frequency of roots (e.g., slm, ktb)</td>
</tr>
<tr>
<td></td>
<td>Word N-grams</td>
<td>word n-grams (e.g., senior editor, editor in chief)</td>
</tr>
<tr>
<td></td>
<td>Punctuation</td>
<td>occurrence of punctuation marks (e.g., !;:,.?)</td>
</tr>
<tr>
<td>Stylistic</td>
<td>Letter N-Grams</td>
<td>frequency of letters (e.g., a, b, c)</td>
</tr>
<tr>
<td></td>
<td>Char. N-grams</td>
<td>character n-grams (e.g., abo, out, ut, ab)</td>
</tr>
<tr>
<td></td>
<td>Word Lexical</td>
<td>total words, % char. per word</td>
</tr>
<tr>
<td></td>
<td>Char. Lexical</td>
<td>total char., % char. per message</td>
</tr>
<tr>
<td></td>
<td>Word Length</td>
<td>frequency distribution of 120-letter words</td>
</tr>
<tr>
<td></td>
<td>Vocab. Richness</td>
<td>richness (e.g., hapax legomena, Yule’s K)</td>
</tr>
<tr>
<td></td>
<td>Special Char.</td>
<td>occurrence of special characters.</td>
</tr>
<tr>
<td></td>
<td>Digit N-Grams</td>
<td>frequency of digits (e.g., 100, 17, 5)</td>
</tr>
<tr>
<td></td>
<td>Structural</td>
<td>has greeting, has url, requoted content, etc.</td>
</tr>
<tr>
<td></td>
<td>Function Words</td>
<td>frequency of function words (e.g., of, for, to)</td>
</tr>
</tbody>
</table>

With respect to the features used for text representation, term (word) frequency is a frequently used strategy, namely in information retrieval. Still, for opinion mining, better results have been reported with the BOW representation [43]. This is because for sentiment classification the number of times a word appears is not as important as it is in topic categorization. Another frequently used representation is term frequency-inverse document frequency.
frequency (TFIDF), which is a numerical statistic that reflects how important a word is to a
document in a collection of texts. This measure increases proportionally with the number of
times a word appears in the text document, but is offset by the frequency of the word in the
corpus, which helps to control for the fact that some words are generally more common than
others. Variants of TFIDF have been developed, like the Delta TFIDF [36] which claims to be
an intuitive general purpose technique to efficiently weight word scores before classification.

Term positioning is another factor that can be important. Whether a term appears at the
beginning, middle or end of a document can affect the overall sentiment of the person
towards the topic. This information has been used in feature vectors [43].

Association rules can also be used for mining frequent occurring phrases [27]. By using an
association rule miner, CBA [35], which is based on the Apriori algorithm [2], all the frequent
itemsets are found. Then different techniques and parameters are used to prune and take
only the most interesting itemsets.

Different domains may use terms with different meaning. For instance, the word "bull" in
the context of financial markets has a very specific meaning. Knowing the domain being
addressed in a text document and being able to find and incorporate information that is
specific to the domain may be important to understand the opinion of a person. Taking into
account this information can greatly improve the results [24, 39, 44].

Grammar classifies words based on parts of speech (POS) tags such as verbs, nouns,
adjuncts, adverbs, etc. Each part of speech explains how the word is used. The same
word can be a noun in one sentence and a verb on another. POS tags can be used for
word sense disambiguation [55], sentence subjectivity finding through adjectives [26] and
more [39, 53, 25, 43, 6, 54]. Using this information can also bring additional accuracy
to opinion mining approaches.

Including syntax in the feature vector seems to be useful especially on short documents.
This can be seen as a deeper linguistic analysis and it is used by some researchers with
success [32, 29, 1].

Negation is an issue with an important role on sentiment analysis. For example, the
sentences

"There is a new vehicle currently being developed that is going to be great."

and

"There is a new vehicle currently being developed that is not going to be
great."
have a very similar bag of words representation but the negation word "not" completely changes the sentiment. Na et al. [40] have proposed an approach to solve this problem. They look for specific POS tag patterns and mark the whole phrase as a negation phrase improving the results (Tables 2.2 and 2.3). The data used on this study was taken from Review Centre [21]. 1800 documents were used, 900 positive reviews and 900 negative reviews. The sample was divided into a training set of 1200 reviews (half positive and half negative) and the test set was composed by the other documents. Review Centre uses a 10-star rating system but the authors classified as positive all documents with 7-stars or more and as negative all documents with 4-stars or less.

Table 2.2 Approaches and results with various techniques [40].

<table>
<thead>
<tr>
<th>ID</th>
<th>Approach</th>
<th>Selected Terms</th>
<th>Term Weighting</th>
<th>DF</th>
<th>Terms labeled with POS tags</th>
<th>Negation</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unigram with TF</td>
<td>All TF</td>
<td>3</td>
<td>No</td>
<td>No</td>
<td>74.17%</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Unigram with Presence</td>
<td>All Presence</td>
<td>3</td>
<td>No</td>
<td>No</td>
<td>75.50%</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Unigram with TFIDF</td>
<td>All TFIDF</td>
<td>3</td>
<td>No</td>
<td>No</td>
<td>76.50%</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Unigram with TFIDF and DF = 1</td>
<td>All TFIDF</td>
<td>1</td>
<td>No</td>
<td>No</td>
<td>74.17%</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Unigram labeled with POS</td>
<td>All TFIDF</td>
<td>3</td>
<td>Yes</td>
<td>No</td>
<td>75.83%</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Unigram with selected words (V, A, Adverb)</td>
<td>Verb, Adjective, Adverb</td>
<td>TFIDF 3</td>
<td>No</td>
<td>No</td>
<td>77.33%</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Unigram with selected words (N, V, A, Adverb)</td>
<td>Noun, Verb, Adjective, Adverb</td>
<td>TFIDF 3</td>
<td>No</td>
<td>No</td>
<td>75.50%</td>
<td></td>
</tr>
</tbody>
</table>
Table 2.3 Negation phrase results [40].

<table>
<thead>
<tr>
<th>ID</th>
<th>Approach</th>
<th>Selected Terms</th>
<th>Term Weighting</th>
<th>DF</th>
<th>Terms labeled with POS tags</th>
<th>Negation</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unigram with negation phrase and DF = 3</td>
<td>TFIDF</td>
<td>3</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>78.33%</td>
</tr>
<tr>
<td>2</td>
<td>Unigram with negation phrase and DF = 1</td>
<td>TFIDF</td>
<td>1</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>79.33%</td>
</tr>
</tbody>
</table>

Lifeng Jia [28] introduced the concept of scope of the negation term \( t \), which is the sequence of words after \( t \) and is affected by \( t \). The results obtained with this approach can be seen in Table 2.4.

Table 2.4 MAP scores of 5 methods on all TREC queries [28].

<table>
<thead>
<tr>
<th>Method</th>
<th>Positive</th>
<th>Improvement By SCT</th>
<th>Negative</th>
<th>Improvement By SCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCBL</td>
<td>0.1596</td>
<td>2.9%</td>
<td>0.0779</td>
<td>11.3%</td>
</tr>
<tr>
<td>SC4</td>
<td>0.1634</td>
<td>0.5%</td>
<td>0.0805</td>
<td>9.8%</td>
</tr>
<tr>
<td>SC5</td>
<td>0.1630</td>
<td>0.5%</td>
<td>0.0812</td>
<td>8.9%</td>
</tr>
<tr>
<td>SCNegEx</td>
<td>0.1487</td>
<td>10.4%</td>
<td>0.0823</td>
<td>7.4%</td>
</tr>
<tr>
<td>SCT</td>
<td>0.1642</td>
<td>-</td>
<td>0.0884</td>
<td>-</td>
</tr>
</tbody>
</table>

2.4 Main Approaches

The task of extracting the opinion in a text document based on a feature vector describing properties of this document can be cast as a modelling task in a standard data analysis framework. As mentioned before we aim at uncovering the function that maps descriptions into opinions. Depending on the type of data we have available for this task, different techniques may be applicable to achieve this goal.

Supervised learning is a group of techniques that take labelled/training data where each sample is described by a set of variables and it has an associated label, and produce
a model that can be seen as an approximation of the unknown function that maps the values of the variables into labels. This type of models can be used to assign labels to new unlabelled samples. These techniques can be applied to the opinion mining task provided we have a sufficiently representative sample of labelled text documents, i.e. texts which were analysed by a human expert that has assigned an opinion score to each of them. Depending on the way we represent opinions (e.g. positive vs. negative or an ordered score), different learning algorithms may be applied. Frequently used learning algorithms include support vector machines, naive Bayes and decision trees, but many other techniques can be applied provided the texts are pre-processed to conform to the standard assumed by these techniques, i.e., a data table where each line represents a text, and each column a variable (property) of this text, where one of the columns will be the opinion score assigned by the domain expert.

Unsupervised learning is another set of approaches that have as goal to find hidden structure in unlabelled data. In the context of text mining this would correspond to not having opinion scores assigned to each text, and having simply the text described by some features (e.g. Table 2.1). The task of these techniques is then to form groups of texts that share similar feature values. Theoretically, these groups of texts should correspond to similar opinions given that they are similar. Obviously, these approaches would then require an extra step of deciding to which type of opinion each of the found groups belongs. Typical approaches to unsupervised learning include clustering (e.g. K-means) and blind signal separation using feature extraction techniques for dimensionality reduction (e.g. Principal Component Analysis). On sentiment analysis the typical approach starts by creating a sentiment lexicon and then determining the positivity of the document based on the lexicon that was created [26].

Semi-supervised learning is a mixture of both previous techniques and it is particularly used approach when the labeled data is scarce. In the model training process, the unlabeled data is taken into account and used to train the model. Andrew B. Goldberg and Xiaojin Zhu [23] use a graph-based semi-supervised learning algorithm to address the sentiment analysis task of rating assignment, showing that this method achieves better predictive accuracy over methods that ignore the unlabeled data completely during training. Xiaojin Zhu [57] published a survey about generative models, self-training, co-training, multiview learning and graph based methods.

In document summarization, we can create either single-document sentiment summaries or multi-document sentiment summaries, giving us the general sentiment of the documents towards a topic. As an example, Philip Beineke and Trevor Hastie [5] introduce the idea of a sentiment summary, It is obtained with a single passage on a document in which they try to capture a key aspect of the author’s opinion. By using supervised data they search for features that appear to be helpful to locate a good summary sentence. In figure 2.4 we
have one example of a summarization system given by Minqing Hu and Bing Liu [27].

![Figure 2.4: Opinion summarization system [27].](image)
Chapter 3

Sentiment Analysis using Text Mining

The main goal of the work in this thesis is to be able to infer the sentiment of a population concerning a set of topics using textual data available in e-participation web sites. This chapter formalizes this problem and describes the main approaches that can be followed to achieve this goal, as well as the approaches that we will use in the remaining chapters.

3.1 Problem Formalization

Text mining allows us to analyse text documents and extract the information contained in the text. In our case the goal is to infer the sentiment expressed in each document concerning a set of topics. Nowadays there is a massive amount of the data available on the internet and this is an invaluable source of information on the opinion of people concerning almost every possible topic. Different e-participation tools facilitate the task of expressing our opinion. Having a system capable of classifying documents automatically will allow us to analyse massive amounts of data and extract useful information from them by looking at the sentiment expressed by the public.

The sentiment on a certain topic or set of topics, can be expressed in many ways. Usual formats include positive vs negative sentiment, or some rating scale. In this thesis we follow the latter approach by trying to infer the sentiment in a document in terms of a $-2, -1, 0, 1, 2$ scale, where negative numbers represent negative sentiment, while positive numbers the opposite. Obviously, other granularity would be possible, but the approaches
we will describe are generalisable to these other solutions as long as they can be regarded as values of an ordinal variable.

Assuming we settle on some form of representing the information in a text document as a feature vector, we can look at the task of inferring the sentiment on this text document as an instance of a standard predictive task. Predictive tasks can be described as data analysis problems where one assumes that there is a functional dependency between a target variable $Y$ and a set of descriptor variables (or predictors) $X_1, X_2, \cdots, X_p$. The goal of predictive modeling is to infer this function from a sample of mappings between values of the predictors and the target variable, i.e. a (training) data set $\{(x_i, Y_i)\}_{i=1}^N$, where $x$ is a feature vector formed by values of the $p$ predictor variables $X_1, X_2, \cdots, X_p$.

In data mining the two most common instances of predictive tasks are known as regression and classification. In regression we use the provided training data set to induce a model of the unknown function,

$$Y = f(x) \quad (3.1)$$

where $Y$ is a numeric target variable and $x$ is a vector of predictor variables $X_1, X_2, \cdots, X_p$.

In classification we have a similar inference problem but the domain of the target variable is a set of labels, i.e. $Y$ is a nominal variable.

Given that our target variable is the value of the sentiment on an ordered fixed scale, i.e. an ordinal scale, we have a particular type of prediction task that differs from the more standard regression and classification tasks. Few modelling techniques exist to handle predictive tasks with ordinal target variables. Using these approaches would strongly limit our range of applicable models. In this context, we have followed a different path, where we have tried to address the problem using the more frequent regression and classification algorithms.

Another distinguishing feature of our particular sentiment analysis task is that we want to infer the sentiment on a (pre-defined) set of topics and not a single topic. Moreover, we assume that each document may express sentiment concerning more than one topic. In this context, we have what is usually known as a multivariate prediction task, i.e. we are trying to predict the value (in this case the sentiment score) of more than one target variable (one for each topic) from the values of a set of predictors describing the text. Models being able to tackle multivariate tasks are again very uncommon within data mining and related fields. In this context, we once again will resort to approaches that allow the use of standard predictive modelling tools for these specific problems.

Finally, another particularity of our target problems is that any document may not address at all any of the topics from the pre-defined set. Notice that this is different from referring the topic with a neutral sentiment (a score of 0 in our scale). This means that we have to
decide what to do with these situations, i.e. what is the correct prediction of a model for the sentiment of a topic when a document does not refer this topic? We will consider two alternatives to answer this problem: i) including this as a special value of target variable; or ii) handle this as two prediction tasks: first decide on whether the topic is mentioned or not, and then decide on the sentiment.

In summary, our main predictive task of inferring the sentiment expressed in a text concerning a set of topics can be cast as a predictive task of the form,

\[ y = f(x) \]  

where \( y \) is a vector of ordinal variables \( Y_1, Y_2, \ldots, Y_q \) with domain \( D_{Y_i} = \{-2, -1, 0, 1, 2\} \); and \( x \) is a vector of predictor variables \( X_1, X_2, \ldots, X_p \).

Notice that this definition is only applicable if we assume that any document will mention/address all \( q \) topics. If that is not the case then this formalization is not applicable. As mentioned above we will also address this particular case of situations where the documents do not refer some of the topics by following two different paths: (i) maintaining the above formalization but extending the domain of the target variables to include a special value representing the absence of reference to the topic; or (ii) decomposing this into two different prediction tasks.

3.2 Proposed Solutions

The task we have defined in Section 3.1 poses 4 main challenges: i) the form of representing the information in the text documents; ii) the way to handle ordinal target variables; iii) the method used to solve a multivariate predictive task; and iv) how to address the fact that some topics may not be referred at all in some documents. There are several possible solutions to these problems. In this section we describe the approaches that were followed in this thesis.

3.2.1 Document Representation Strategy

The way we represent a document can have an impact on the final results and on the performance. As we have discussed in Section 2.3 and by looking at Table 2.1 (page 22), we have several ways of representing a document. The most popular alternatives are the Bag of Words and the N-gram representations.
The N-gram representation involves the creation of sequences of N-words. For example, on a 2-gram representation, the sentence ‘I went to the garden today’ could generate 3 groups of 2-grams, ‘I went’, ‘to the’, ‘garden today’. Then, after discovering all the groups in our corpus, we count how many times they appear in the document and assign this value to the group. This type of representation tries to keep some information about the sequence of the words or the context in which each word appears.

The Bag of Words (BOW) representation, the one we adopted, is the most frequent approach. We represent the document by separating the sentences into single words. For example, on the previous referred sentence, we can identify the words ‘I’, ‘went’, ‘to’, ‘the’, ‘garden’ and ‘today’. This strategy usually proceeds by identifying all words in a given corpus (eventually after some pre-processing steps like stop word removal, or word stemming) and then by counting the occurrences of each identified word on each document. This means that the features or predictor variables used to represent the texts in a data set will be this (often large) set of identified words. As values of these predictors an usual choice is to assign the frequency (the number of times the word appears on the document, or term frequency (tf)). Another option is the tf-idf (term-frequency inverse-document-frequency) score which attempts to normalise the term frequency with a factor related with the importance of each word (term) of a document within a collection of documents. If the word appears more frequently in the collection of documents then its tf-idf value will be high. This allows us to know which words separate documents better (if they only appear in few documents then they distinguish these from the others). Our bag of words implementation code is described in Annex B.3.

On both representations, we need to decide about what to do with all the words found in a corpora. Do all of them interest us? Should we, for example, keep numbers and punctuation? Although some of these decisions may be domain-dependent, frequent pre-processing stages include: (i) removal of stop words; (ii) removal of punctuation and numbers; and (iii) word stemming.

In summary, although many alternatives exist for representing the information in a text document we have selected the frequently used bag of words representation using term frequency as values. We have also opted to remove stop words, punctuation and numbers and apply word stemming. In order to reduce the number of words, we have removed sparse terms with a factor of less than 0.95. This resulted in using a total of 172 words that represent our documents.
3.2.2 Handling Ordinal Target Variables

Our target variables are sentiment scores in a set of pre-defined topics. The sentiment can be expressed as $-2, -1, 0, 1, 2$ and from here we can infer that there is an implicit order. A score of -1 expresses a better sentiment than a document with a score of -2. Although these are values of an ordinal variable, for the reasons already outlined, we can also address this as values of nominal or numeric variables (i.e. as classification or regression tasks). In order to be able to make this "transformation" some steps need to be taken.

Classification tasks do not assume any order among the values of the target variable, which we have seen is not true in our sentiment scale. An order among the values means that it is more serious misclassifying a document with sentiment $-2$ as having sentiment 2, than classifying it as having sentiment $-1$. Classification algorithms consider all errors equally serious and thus can not cope with the above distinction. To achieve this distinction we can resort to cost matrices. A cost matrix is a $N \times N$ matrix where $N$ is the number of possible labels of the target variable. The rows and columns of this matrix represent the possible values for the predictions and true values of any test case. The entries in the matrix specify a value (a cost) for each possible combination of predicted and true target variable value. Using these matrices we can specify the costs such that it is more costly for the model to predict a value of $2$ for a document with true sentiment of $-2$, than the cost of predicting $-1$. This means that through cost matrices we can convey the order information to the classification models by means of different costs of the errors.

Regression tasks assume that the target variable is numeric, which means that there is an implicit ordering among its values. This allows us to handle the different types of sentiment errors naturally without having to resort to cost matrices as in classification. Still, regression methods allow interpolation among values, which means that some model could come up with a predicted sentiment score of $1.234$. In order to force the predictions into our selected sentiment scale, when using regression tools, we will re-scale the predicted values to the original scale by applying a rounding operation to the predictions.

3.2.3 Methods for Addressing Our Multivariate Predictive Task

Our predictive task is multivariate because we have multiple target variables - the sentiment of each of the selected topics. A document can refer multiple topics and we can have different sentiment scores for each of them. There are different approaches to solve this problem. One of them is using multivariate prediction tools (e.g. Multivariate Regression Trees [13]) which attempt to predict all variables at the same time, trying to take advantage of eventual relationships between the target variables. This type of methods is not very
frequent and few techniques/tools exist. In this context, we have decided not to limit our options in terms of tools and handle this multivariate task in a simpler way by making the assumption that the target variables are independent. With this assumption we can transform a multivariate task with \( q \) target variables into \( q \) different univariate predictive tasks that share the same predictors but have a different target variable. We thus learn \( q \) standard predictive models, one for each of the topics in our study.

Another problem related with the selection of multiple topics of interest is the fact that each document is likely to only have an opinion expressed about a very small sub-set of these topics. This raises the problem of how to evaluate the model predictions in those situations, i.e. what is the correct prediction of a model trained to forecast the sentiment score for topic \( X \), on a document that does not talk about \( X \)? We have considered two approaches to this problem. A first solution solves this problem by decomposing it into two separate predictive tasks: a first that has the goal of deciding whether a topic is mentioned or not in a document (i.e a binary classification problem); and a second task that has the goal of forecasting the sentiment, but which is only applied if the first model says the document mentions that topic, otherwise no sentiment is predicted for that topic. The second solution we have tried is to incorporate this state of a document not mentioning a certain topic as an extra value of sentiment score, that we will name "DS" from "does not speak". Note that adding this new value into the sentiment scale allows us to still have as many predictive models as there are topics, whilst the first solution of the sequential prediction will lead to having 2 models for each topic, i.e. twice as many models. Still, the approach of adding a new sentiment "score" will raise other problems in terms of evaluating the predictions of the models. These problems and the solutions we have adopted will be described in Section 3.2.5.

### 3.2.4 Tested Solutions

In the previous sections we have described several alternative ways of handling some of the problems raised by our target application, namely the issue of having ordinal target variables and also the issues related with multivariate tasks and the absence of mention of some of the topics. We have selected four combinations for our experimental comparison that we describe in this section.

The first approach consists of doing a two-stages (sequential) prediction by having first a binary classification model that decides if each topic is mentioned or not, and then, if the answer is yes predict the sentiment again using a classification algorithm using cost matrices to handle the order among the scale. We will name this approach \( bc + c \) (binary classification plus classification).

The second solution we evaluate is again using a two-steps prediction approach by having
the same binary classification task, but then forecasting the sentiment using a regression tool instead. We will name this approach $bc + r$ (binary classification plus regression).

The third solution attempts to do the prediction in a single step. A classification model predicts the sentiment about each topic, where the predictions may include a special value with the meaning that the model predicts that the topic is not mentioned in the document. We will name this approach $c$ (single classification model).

The fourth solution is similar to the third (i.e. everything with a single model) but this time we use regression tools to solve the prediction tasks. We will name this approach $r$ (single regression model).

Please remark that for all these four solutions we will have to replicate them for each of the $q$ selected topics, given that we have decided that we would address the problem of having to forecast the sentiment for $q$ topics as $q$ independent prediction tasks.

On the $bc + c$ and $bc + r$ solutions, we need to train $2q$ models where $q$ is the number of selected topics. With the other two solutions we will only need to train $q$ models. This can have a significant impact on computation time depending on the number of topics and documents.

As we will see later, even after defining these approaches we still have many options concerning the modelling tools (and respective parameter variants) that we will use to solve each modelling task. The selected alternatives will be described in Section 4.5 (page 48) of Chapter 4.

3.2.5 How the Solutions will be Compared/Evaluated?

In order to compare and evaluate our solutions, we must consider the fact that each of them produces different outputs. Still, independently of the approaches taken to solve the original task, we must not bias the evaluation of the results by these solutions - it is exactly the opposite. In a data mining project we should first decide on what is the task and how will solutions be evaluated, and only then we can think of methods to solve the task that somehow try to optimize the selected evaluation criteria.

Our task can be summarized as follows: given a text document we want to know if the document mentions a set of pre-defined topics and if yes, with what sentiment in a pre-defined scale of sentiment. In this context, for each of the $q$ selected topics there are 6 possible answers: does not speak about the topic, or any of the 5 possible sentiment scores we have selected. Whatever the answers provided by our different approaches, they need to be compared against this ground truth (i.e. these 6 possible true values for each
In this context, we have two problems: i) first we need to cast the results of our four approaches into these 6 possible values; and then ii) we need to decide how to penalize the eventual errors the approaches make.

The first problem only arises in approaches involving regression tools, because for classifications tools (independently of using the two-steps, \( bc + c \) or single step approach, \( c \)) the predictions are already in this 6 values scale. For regression tools we resort to a truncation mechanism that transforms any real value into the 5 possible integer scores \(-2, -1, 0, 1\) and \(2\). Namely, this truncation problem arises for solutions \( bc + r \) and \( r \).

For solution \( bc + r \) we have a binary classifier outputting either "DS" (does not speak of the topic) or "S" (mentions the topic). If the output is "S" then the document is passed to the second stage where a regression algorithm outputs a real value as the predicted sentiment score. This real value \( x \) is then truncated using the following rule:

\[
f(x) = \begin{cases} 
2 & \text{if } x \geq 1.5 \\
1 & x \in ]0.5, 1.5] \\
0 & x \in ]-0.5, 0.5] \\
-1 & x \in ]-1.5, -0.5] \\
-2 & \text{if } x < -1.5 
\end{cases}
\] (3.3)

For solution \( r \) we need to transform the real value \( x \) output by the models into a scale of 6 possible values (the 5 sentiment scores plus the "DS" value). This is done using the following mapping function:

\[
f(x) = \begin{cases} 
2 & \text{if } x \geq 1.25 \\
1 & x \in ]0.75, 1.25] \\
\text{DS} & x \in ]0.25, 0.75] \lor x \in ]-0.75, -0.25] \\
0 & x \in ]-0.25, 0.25] \\
-1 & x \in ]-1.25, -0.75] \\
-2 & \text{if } x < -1.25 
\end{cases}
\] (3.4)

The mapping of a real value into the "DS" score is clearly very debatable. It is not clear how to do this mapping. Negative (positive) predictions indicate that the regression model believes that the document contains a negative (positive) sentiment on some topic. This could lead us to use 0 as a prediction that should be mapped into the value "DS". However, if we proceeded this way we would not be able to distinguish situations where the document
speaks of the topic but without a particular sentiment. In this context, we have decided that values too near 0 would mean that the model believes that the document mentions the topic but with no defined sentiment. Values not too near 0, but still not with a clear sentiment score would arbitrarily be mapped into the “DS” value. We are aware that this is a highly debatable decision but no better solution was found that allows the use of the $r$ approach.

After these transformations, and independently of the approach followed, we will have predictions in the intended range of 6 values. Next we need to decide how to compare these predictions against the true values in our pre-labelled text documents. We have used as evaluation metric the total cost of the predictions. This evaluation metric assumes the existence of a cost matrix indicating the cost of each misclassification. Models should try to minimize this score. We have used the following cost matrix in our experimental comparisons:

<table>
<thead>
<tr>
<th></th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>DS</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>2.5</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1.5</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>2.5</td>
</tr>
<tr>
<td>DS</td>
<td>2.5</td>
<td>1.5</td>
<td>0.5</td>
<td>1.5</td>
<td>2.5</td>
<td>0</td>
</tr>
</tbody>
</table>

Given this cost matrix the total cost of the predictions of a model for a given test set with $n$ documents is given by,

$$\text{TC} = \sum_{i=1}^{n} C(\hat{y}_i, y_i)$$

(3.5)

where $C(\hat{y}_i, y_i)$ is the cost of the $q$ sentiment predictions for the document $i$, which is given by,

$$C(\hat{y}, y) = \sum_{t=1}^{q} M_{\hat{y}_t, y_t}$$

(3.6)

where $M_{\hat{y}_t, y_t}$ is the entry in the cost matrix $M$ corresponding to a prediction of $\hat{y}_t$ for topic $t$ of a document whose true value for that topic is $y_t$. 
Chapter 4

e-Policy Photovoltaic Problem

In this chapter we begin by describing the general goals of e-Policy European project and the opinion mining problem we have to solve in the context of this project. Following this introduction we will describe the data that was collected in the context, and also the results of our exploratory analysis of the data. The evaluation and experimental methodologies we have used in order to compare different models that were tried is then explained in detail. Finally, we present the results of this experimental analysis of different models in our opinion mining tasks.

4.1 Problem Description

As we have mentioned before the work presented in this thesis was developed in the context of an European research project - e-Policy. The main goal of this project is to develop a decision support system to help energy policy makers to take their decisions. This system integrates several components, one of which is an opinion mining system whose goal is to infer the sentiment of the population concerning different alternative energy policies. Figure 4.1 provides a general overview of the e-Policy decision support system, where we may find the role of the opinion mining components. The following is a brief project description taken from the document of work (DOW) of the project:

"The e-Policy project is a FP7 STREP project funded under the Information and Communication Technologies (ICT) theme, Objective 5.6 ICT solutions for Governance and Policy Modeling.

Its main aim is to support policy makers in their decision process across a multi-disciplinary effort aimed at engineering the policy making life-cycle. For
the first time, global and individual perspectives on the decision process are merged and integrated. The project focuses on regional planning and promotes the assessment of economic, social and environmental impacts during the policy making process (at both the global and individual levels). For the individual aspects, e-POLICY aims at deriving social impacts through opinion mining on e-participation data extracted from the web. To aid policy makers, citizens and stakeholders, e-POLICY heavily relies on visualization tools providing an easy access to data, impacts and political choices.

The e-Policy case study is the Emilia Romagna Regional Energy plan. e-Policy will provide a tool for supporting regional planners to create an energy plan that is in line with strategic EU and national objectives, consistent with financial and territorial constraints, partecipated including opinion mining results, well assessed from an environmental perspective and optimal with respect to one or more metrics. In addition to the regional plan, e-Policy will provide a portfolio of implementation instruments (namely fiscal incentives, tax exemption, investment grants) for pushing the society and the energy market to go in the direction envisaged by the plan."

Figure 4.1: e-Policy project system [15].
In the context of the e-Policy Project [15], the role of opinion mining is to provide feedback to policy makers concerning the sentiment of the population with respect to different energy policies. The idea is to provide a series of e-participation tools to the population so that they can express their opinions on several issues related to energy policies. The goal of opinion mining is then to infer the sentiment of the population concerning different topics of interest to policy makers, from data collected at these e-participation sites.

The e-Policy project is concerned with energy policies for the region of Emilia-Romagna in Italy. In this context, all activities concerning the involvement of the population with e-participation tools will naturally use the Italian language. Most of the existing research on text mining is carried out with the English language but work on other languages is growing [4]. Especially in huge global events such as the Olympics or Soccer World Championships, it is very important for the media to be able to extract and process large amounts of data as fast as possible which makes the study and development of this field very important in all languages. On the e-Policy project, having efficient models and tools tailored for the Italian language is essential.

In terms of the goals of opinion mining within the project the consortium has decided to focus on 14 main topics and 3 subcategories (economic, environmental and technological aspects) for each, totaling 42 topics. The goal of the tools to be developed within the project is to infer the sentiment of the population concerning these 42 topics and also to provide information on tendencies of this sentiment along time, so that the eventual impact of decisions taken by policy makers can be measured. The list of 14 selected main topics is the following:

- Photovoltaic
- Thermal
- Wind power
- Hydroelectric
- Biomass
- Geothermal
- Biogas
- Fusion
- Biofuels
- Eco-Mobility
CHAPTER 4. E-POLICY PHOTOVOLTAIC PROBLEM

- Combustion
- Free energy
- Energy saving
- Waste to energy

As mentioned above for each of these 14 topics, 3 different aspects were considered.

4.2 Data Collection

It was decided to extract documents from two Italian websites [16, 17] - Energetic Ambient (Figure 4.2 and 4.3) and the Newclear blog (Figure 4.4). On both websites we have structured the different posts as a hierarchy starting with a top post and then subsequent posts discussing this main post. After deciding on this representation we created two crawlers, one for each website, that on a daily basis try to find and extract new documents. We discuss the crawlers implementation in Annex B.1.

![Energetic Ambient front page](image)

Figure 4.2: Energetic Ambient front page [16].
From crawling these two websites we have collected a data set with posts and some information associated with each post. Table 4.1 presents the information that is collected for each post by our crawlers, like the date, title and post counter of each post (if it is a main post or a reply to the main post), etc. In spite of the availability of all this information, the approaches described in this thesis will only make use of the text of each post.

All approaches described in Chapter 3.1 are designed to solve predictive tasks. These tasks require a training set where the values of the target variables are known. In the context of our opinion mining tasks this means that we need a data set with posts which are tagged regards the sentiment expressed for each of the topics selected for this study. Tagging a large amount of posts for these specific topics is a task that requires huge human resources with expertise in the energy field. That is the main justification for the fact that the amount of tagged data is very low when compared with all the available data as seen in Table 4.1. Moreover, we only have sentiment scores for 3 topics from the 42: ‘Photovoltaic economic aspects’, ‘Photovoltaic environmental aspects’ and ‘Photovoltaic technology aspects’. A website, which is described in Annex B.2, was created so that a user can view the posts and tag them accordingly.

We should remark that the total number of posts mentioned on Table 4.1 is the number used in this thesis. Still, this number is growing on a daily basis as the crawlers are being executed in "real time".
Table 4.1 ePolicy Dataset composition.

<table>
<thead>
<tr>
<th>Number of Documents</th>
<th>Number of Tags</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>582382</td>
<td>1048</td>
<td>ID, Author ID, Title, Text, Date, Postcounter, URL, Blogname, Topic, Score</td>
</tr>
</tbody>
</table>

4.3 Exploratory Analysis

This section presents the results of the exploratory analysis of the data set presented in Table 4.1.

All graphs besides the bar graph on Figure 4.13 show a black line (dots in the scatterplots) which corresponds to the frequency, a blue line and a bandwidth that provide an idea of both the tendency an variability of the individual scores along time. The blue line is calculated using a local polynomial regression model [10] in which the fit at point $x$ is obtained using points in a neighborhood of $x$, weighted by their distance from $x$. We used the default size for the neighbourhood which consists of 50 points. The bandwith represents the confidence interval of the fit.
The graphs in Figures 4.5 and 4.6 show the distribution of the number of tagged posts along the period for which we have posts. We can verify that we have a peak of tagged posts in 2008 and a peak of untagged posts in 2011. Although the labelling process was not controllable by us, this distribution has the danger of being not equally distributed across the years, which means that if there is some time dependency of the sentiment of the population concerning the different topics the models may not be able to capture this effect. Moreover, it is also clear that we will not be able to do a daily analysis of the tagged posts because we do not have sufficient posts per day.
Graphs in Figures 4.7 and 4.8 reveal that if we aggregate posts by week, especially on the untagged posts case, we have data for every week but we still have a lot of weeks where no tagged data is available.
The graphs on Figures 4.9 to 4.12 show the same numbers for other aggregation levels. Figure 4.11 reveals that 2007 is the year where most of the tagged posts are and Figure 4.12 reveals that 2010 is the year where we can find most of the untagged posts. Besides this, 2007 is the year where we have most of the tagged posts and also a considerable amount of untagged posts.

The bar graph on Figure 4.13 reveals that we have a small number of labelled posts for the
topic 'Fotovoltaico Environmental Aspects’. This fact will have an impact on the performance of the models and their evaluation, as we will confirm in Section 4.6.

The scatter plots show in Figures 4.14 to 4.16 present the distribution of sentiment expressed in the labelled posts for the 3 selected topics along the time. Each dot in these figures is the sentiment assigned by the human annotator to each post.

On average, the overall sentiment for all three topics seems rather neutral. On the scatter plot of Figure 4.16 we could say that the sentiment about this topic has been increasing, though since we have less amount of tagged data on recent years there is some degree of uncertainty in such statement. The economic and environmental aspects are close to neutral sentiment on most of the analysed time frame.

Figure 4.15: Environmental aspects score distribution through time. Figure 4.16: Technological aspects score distribution through time.
Figures 4.17 and 4.18 are word clouds obtained using the documents with positive and negative sentiment scores, respectively. We can see some words that are more frequently found on one type of documents than in the other. The word 'alcun' is the most frequent on both type of documents while for example words such as 'bast', 'arriv' and 'buon' are more frequent on positive documents while words 'camb', 'bass' and 'banc' are more frequent on negative sentiment documents\(^1\). The details of the code used to produce the wordclouds are presented in Annex B.5.

4.4 Evaluation and Experimental Methodology

The goal of the work carried out in this thesis is to create models that are able to predict the sentiment concerning a series of topics in a document. In Section 3.2.4 we have described four different approaches to this predictive task. To obtain the necessary models we will use the labeled posts that we were able to obtain in the context of the e-Policy project. For each of the possible approaches to the problem different modeling techniques may be applied. In our work we have considered random forests, support vector machines and neural networks as the base learners. A brief introduction to these learning techniques will be given in the next section.

In order to evaluate and compare the models we have measured the total cost, which is

\(^1\)These are not complete Italian words, given that we were using word stemming before this analysis took place.
given in Equation 3.5 (page 36) and that was calculated using the cost matrix on Table 3.1 (page 36).

To obtain reliable estimates of the total cost for the selected models we have resorted to the use of 10 repetitions of a 10-fold cross validation experiment. The code used to carry out this experiment is available and explained in Annex B.4.

4.5 The Used Modelling Techniques

As the base learners for our sentiment analysis models we used some of the most popular techniques: Random Forests, Support Vector Machines and Neural Networks.

Random Forests [7] are an ensemble learning method for classification and regression tasks composed of many decision trees created with the training data. Each tree is trained on a bootstrapped sample of the original dataset and each time a split node is created, only a randomly chosen subset of the predictors are considered for splitting. In terms of using random forests for prediction, their forecasts are the mode of the classes outputted by each tree in the ensemble in the case of classification tasks, or the average of the predicted values if it is a regression problem. In our experiments we have used the implementation available in the R package ‘randomForest’, ported from the original Fortran code by Andy Liaw and Matthew Wiener [33].

Support Vector Machines [11, 12], or SVMs, are a relatively recent modelling approach that has been witnessing a large success in many application domains. This approach can be applicable to both classification and regression tasks. Nevertheless, the approach was originally developed for binary classification problems and it is easier to explain their method within this setup. SVMs try to find a hyperplane that separates the cases belonging to each class (as for instance linear discriminants also do). With the goal of finding the hyperplane that maximizes the margin between the cases of the two classes, SVMs use quadratic optimization algorithms. Unfortunately, most real world problems are not linearly separable. The solution provided by SVMs to this problem consists in mapping the original data into a higher dimension input space where the cases belonging to the two classes can already be linearly separable. Although this solves the problem of linear separability, this creates another problem - applying the quadratic optimization algorithms on these high dimension spaces is computationally very demanding. To solve this extra problem SVMs use what is known as the kernel trick, which consists in using certain kernel functions that are cheap to compute and that are proven to lead to the same result as the expensive dot products that are used in the quadratic optimization algorithms when applying them in the high dimension space. These kernel functions are cheap to compute because they are
calculated in the original, low dimension space. Still, their result is equal to the mentioned dot products which allows SVMs to obtain the hyper-planes in the high dimension space without having to carry out heavy computation steps on this space. This general approach has been generalized to both multi-class problems and regression tasks, and thus we can use this methodology in our tasks. We have used the SVM implementation available in the R package 'e1071' created by David Meyer [38].

Artificial Neural Networks [37, 56] are models with a strong biological inspiration. They are composed by a set of units (neurons) that are connected. These connections have an associated weight and the learning process consists of updating these weights. Each unit has an activation level and means to update this level. Some of these units are connected to the outside, being called input and output neurons. Each unit has one simple task, receive the input impulses and calculate its output as a function of these impulses. This calculation is divided in two parts: a linear computation of the inputs and a non-linear computation (activation function). Different activation functions provide different behaviours. Some examples of common functions are the Step function, the Sign function and the Sigmoid function. The units can also have thresholds that represent the minimum value of the weighted sum of the inputs that activates the neuron. There are two main types of Artificial Neural Networks:- the feed-forward networks and the recurrent networks. The feed-forward networks have unidirectional connections (from input to output), without cycles, while the recurrent networks have arbitrary connections. Usually the networks are structured in layers. On a feed-forward network each unit is connected only to units on the following layers while on a recurrent network this does not happen and the network can have feedback effects, possibly exhibiting chaotic behaviour. They usually take longer to converge. The learning process of Artificial Neural Networks consists of updating the weights of the connections. The most popular way to do this is by using the Backpropagation algorithm. Each example is presented to the network. Then, if the output produced is correct, nothing is done. If it is not correct then we need to re-adjust the network weights. In networks with multiple layers the adjustment is not simple as we need to divide the adjustments across the nodes and layers of the network. A detailed description of the back-propagating algorithm is given by David E. Rumelhart [46]. In our experiments we have used the implementation of feed-forward Artificial Neural Networks available in the R package 'nnet' created by Brian Ripley [52].

In our experiments we have tried different parameter variants of the above 3 modelling techniques. For Random Forests we created different variants by changing the parameter ntree which controls the number of trees to grow, and the parameter mtry that controls the number of variables randomly sampled as candidates for each split. With respect to Support Vector Machines we have varied the parameters cost, epsilon and gamma. The parameter cost sets the value associated with the cost of constraints violation, it is the
'C'-constant of the regularization term in the Lagrange formulation. The parameter $\epsilon$ controls the epsilon in the insensitive-loss function and $\gamma$ is a parameter used in the kernel. Finally, for Artificial Neural Networks we varied the parameter $size$ that controls the number of units in the hidden layer, and the parameter $decay$ which controls the weight decay.

The details on the concrete parameter values used on all variants that we considered in our experiments can be checked in Annex A.

4.6 Analysis of the Results

As mentioned before we have compared the different modelling approaches using the total cost evaluation metric. The estimates of the true total cost of each model variant were obtained using a $10 \times 10$-fold Cross Validation process. The R code used to carry out all experiments is presented in Annex B.4.

To facilitate the comparison among the models and also to better understand the advantage of using our predictive approaches, we have used a kind of baseline prediction model. This naive model will forecast the same sentiment score for each document in a test set. This score will be the mode of the sentiment scores of the documents in the training set. A reference value was created which consists on always predicting the mode. For example, if on the training set the most frequent sentiment score was 0 then the model will always predict 0 for the posts in the test set.

Table 4.2 summarizes the results obtained on the 10 best model variants. These are the models whose CV estimate of the total cost (column Total Cost on the table) is the lowest. The column Relative Cost is obtained by dividing the estimated total cost of each model by the estimate of the above mentioned baseline model. The table also includes information on the name of the model variant and the modelling approach that it follows. The parameter values corresponding to each variant are described in Annex A.
Table 4.2 Best performing models.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Model</th>
<th>Total Cost</th>
<th>Relative Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>cv.randomF.v7</td>
<td>200±10.9</td>
<td>0.39</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.randomF.v7</td>
<td>201±11.0</td>
<td>0.39</td>
</tr>
<tr>
<td>c</td>
<td>cv.randomF.v9</td>
<td>201±11.2</td>
<td>0.39</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.randomF.v8</td>
<td>202±10.8</td>
<td>0.39</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.randomF.v9</td>
<td>202±11.0</td>
<td>0.39</td>
</tr>
<tr>
<td>c</td>
<td>cv.randomF.v8</td>
<td>202±10.9</td>
<td>0.39</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.svm.v10</td>
<td>206±11.0</td>
<td>0.40</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.svm.v13</td>
<td>206±11.0</td>
<td>0.40</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.svm.v16</td>
<td>206±11.0</td>
<td>0.40</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.randomF.v5</td>
<td>206±11.3</td>
<td>0.40</td>
</tr>
</tbody>
</table>

A first thing we can remark from the results shown in Table 4.2 is the fact that a large percentage of the best 10 use random forests as modelling technique. Still, most of the tried models outperform the baseline by a considerable margin.

Another interesting observation from our results is the absence of trials using regression approaches. None of the regression variants reached our top 10, with the best being an SVM with a relative score of 0.77.

Table 4.3 Worst performing models.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Model</th>
<th>Total Cost</th>
<th>Relative Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>bc + c</td>
<td>cv.nnet.v9</td>
<td>494±7.1</td>
<td>0.97</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.nnet.v6</td>
<td>494±6.7</td>
<td>0.97</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.nnet.v4</td>
<td>495±6.9</td>
<td>0.97</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.nnet.v1</td>
<td>495±7.2</td>
<td>0.97</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.nnet.v7</td>
<td>495±7.0</td>
<td>0.97</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.nnet.v2</td>
<td>495±7.2</td>
<td>0.97</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.nnet.v3</td>
<td>495±7.0</td>
<td>0.97</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.nnet.v8</td>
<td>495±6.6</td>
<td>0.97</td>
</tr>
<tr>
<td>bc + c</td>
<td>cv.nnet.v5</td>
<td>497±7.5</td>
<td>0.97</td>
</tr>
<tr>
<td>Baseline</td>
<td>cv.modePred</td>
<td>512±0.0</td>
<td>1.00</td>
</tr>
</tbody>
</table>
Table 4.3 shows the worst performing models. All of them use neural networks as the base technique. Neural networks are known to typically require heavy tuning of their parameters in this may be a possible explanation for such poor results, frequently approaching the performance of the baseline method as seen in the table.

In order to understand if these differences have any statistical significance, we performed a Wilcoxon Signed Rank test which is a non-parametric statistical hypothesis test that compares two related repeated measurements to assess whether each set population mean ranks differ. Table 4.4 shows the results of performing this test with paired comparisons between the best Random Forest against the best Support Vector Machine, Neural Network and the baseline model. The statistics were measured topic by topic instead of using Equation 3.5 (page 36) that sums up the scores on all topics. This means that the results on Table 4.4 refer to total cost by topic, and not summed up for all topics as in the previous tables. This way we can understand if there is any difference on the performance of the models depending on the topic being tested. A ‘++’ sign means that the random forest is better (i.e. has lower estimated cost) with confidence level of 99% while a ‘--’ sign means that the random forest is worst with the same confidence. From the table we can conclude that in most cases the random forest performs better with a confidence level of 99% with the exception being in the technology topic, where the SVM performed significantly better.
Table 4.4 Statistical significance of the observed differences.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Approach</th>
<th>Learner</th>
<th>Average</th>
<th>Standard Deviation</th>
<th>Statistical Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Economic</td>
<td>c</td>
<td>cv.randomF.v7</td>
<td>99</td>
<td>11.2</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.svm.v10</td>
<td>108</td>
<td>12.8</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.nnet.v3</td>
<td>149</td>
<td>8.5</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.modePred</td>
<td>155</td>
<td>7.9</td>
<td>++</td>
</tr>
<tr>
<td>Environmental</td>
<td>c</td>
<td>cv.randomF.v7</td>
<td>12</td>
<td>5.9</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.svm.v10</td>
<td>11.6</td>
<td>5.9</td>
<td>No statistical significance</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.nnet.v3</td>
<td>199</td>
<td>3.4</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.modePred</td>
<td>199</td>
<td>3.2</td>
<td>++</td>
</tr>
<tr>
<td>Technology</td>
<td>c</td>
<td>cv.randomF.v7</td>
<td>89</td>
<td>14.0</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.svm.v10</td>
<td>87</td>
<td>12.8</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.nnet.v3</td>
<td>147</td>
<td>7.9</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>bc + c</td>
<td>cv.modePred</td>
<td>157</td>
<td>6.6</td>
<td>++</td>
</tr>
</tbody>
</table>

Figure 4.19 shows us a series of boxplots of the performance achieved by the same models across the different iterations of the $10 \times 10$-fold CV process. We can clearly see that the topic environmental is the topic in which Neural Networks and the baseline model performed very badly. The SVM and RF performance were very similar on the three topics. When we look to the performance of the baseline model, we can clearly conclude that our approaches perform much better.
From these experiments we can draw the following main conclusions:

- Random forests were the best performing models in general, and with the difference being statistically significant on most set ups.

- There is no clear distinction among the trials using a two-stages approach versus using a single model. Given that using a single model requires lower computation time, this provides some evidence that this may be the best direction for this particular opinion mining task.

- The approaches using a regression tool to estimate the sentiment score achieved a
poor performance when compared to the equivalent classification tools.

• The neural network variants that were considered performed very bad.

• The best models that were tried achieved a performance that is significantly better than the baseline of always forecasting the most frequent sentiment score.

Finally, we should remark that these experiments were strongly limited by the small number of tagged documents that were available. Nevertheless, some interesting patterns of results were already observed in this experimental comparisons. These observations increase our confidence on being able to correctly infer the sentiment of the population concerning these topics, provided more effort is invested in labelling existing posts to better train our models.
Chapter 5

Conclusions

In this chapter we summarize the conclusions of the work carried out in this thesis and outline a few possible directions of future work.

5.1 Summary

This work was motivated by the need to provide the sentiment that a population has toward a set of topics. It involved several steps such as the study of the state of the art on Opinion Mining, data collection, data exploratory analysis, study and testing of several alternative approaches and the analysis of the respective results. Each of these steps had its challenges that were described in the thesis. Our proposed solutions allowed us to give an answer to the requirements of the e-Policy project and create models that can estimate the sentiment score concerning a series of topics for new posts that appear in the selected e-participation sites.

To facilitate the tedious task of labelling posts we have developed a web site that can be used by human experts to label the posts regards the set of pre-defined topics. With the resulting data we have constructed a data base for the ePolicy opinion mining problem. To construct data sets that could be used for sentiment analysis algorithms we have adopted the Bag of Words text representation schema, which is one of the most frequently used and easy to understand representations. To address the predictive tasks of our problem we have considered two types of models: i) regression and ii) classification. Moreover, concerning the fact that our overall goal involves inferring the sentiment on posts for multiple topics, we have adopted the strategy of handling each topic as a different predictive task. Finally, to solve the issue of some of the topics possibly not being mentioned on some posts, we
have considered two alternatives: i) handling this as a special value of the target variable; or ii) handling this as a two-stages prediction task. From all these set-ups we have selected form main approaches to tackle the original problem. For each of these approaches a large set of model variants from three different algorithms were considered. These variants were compared in terms of total cost of their predictions using a $10 \times 10$-fold cross validation methodology.

The results of our experimental analysis in the context of the available labelled data allow us to conclude that the most promising alternatives use Random Forests as baseline learning algorithm, for solving the problem as a classification task. Moreover, we have not observed any advantage of using the computationally more demanding, two-stages approaches. In the tasks we have considered these conclusions are statistically significant.

### 5.2 Future Work

In the future, new approaches to the problem should be considered as well as testing new model variants, particularly for artificial neural networks. Semi-supervised learning should definitely be considered since we have a low amount of tagged data, and these approaches could help in overcoming this major drawback. Analysing other features that are available for the posts, such as the date and title, might also improve the results.

We are still working on the e-Policy project trying new approaches and obtaining new data that will improve our models and allow us to do more experiments. A fully automatic system that will do all the steps mentioned (data collection, model learning and sentiment prediction) is also being finalised.

### 5.3 Final Remarks

This thesis contributes to the opinion mining field by providing an overview of the current state of the art on opinion mining. We show that it is currently possible to automatically tag a large amount of posts with few tagged data. The approaches implemented are simple and easily replicable and they can be applied in any context that involves the tagging of posts that express a sentiment toward any topic.
Appendix A

Model Variants

In this annex we describe the variants of the models detailing the parameter values that were used in each variant. The models whose name ends in $r$ are the variants in which regression was used to obtain the results.
### Table A.1 Random Forests parameters.

<table>
<thead>
<tr>
<th>Name</th>
<th>Number of trees</th>
<th>Mtry</th>
</tr>
</thead>
<tbody>
<tr>
<td>cv.randomF.v1</td>
<td>100</td>
<td>3</td>
</tr>
<tr>
<td>cv.randomF.v2</td>
<td>500</td>
<td>3</td>
</tr>
<tr>
<td>cv.randomF.v3</td>
<td>1000</td>
<td>3</td>
</tr>
<tr>
<td>cv.randomF.v4</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>cv.randomF.v5</td>
<td>500</td>
<td>5</td>
</tr>
<tr>
<td>cv.randomF.v6</td>
<td>1000</td>
<td>5</td>
</tr>
<tr>
<td>cv.randomF.v7</td>
<td>100</td>
<td>7</td>
</tr>
<tr>
<td>cv.randomF.v8</td>
<td>500</td>
<td>7</td>
</tr>
<tr>
<td>cv.randomF.v9</td>
<td>1000</td>
<td>7</td>
</tr>
<tr>
<td>cv.randomFr.v1</td>
<td>100</td>
<td>3</td>
</tr>
<tr>
<td>cv.randomFr.v2</td>
<td>500</td>
<td>3</td>
</tr>
<tr>
<td>cv.randomFr.v3</td>
<td>1000</td>
<td>3</td>
</tr>
<tr>
<td>cv.randomFr.v4</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>cv.randomFr.v5</td>
<td>500</td>
<td>5</td>
</tr>
<tr>
<td>cv.randomFr.v6</td>
<td>1000</td>
<td>5</td>
</tr>
<tr>
<td>cv.randomFr.v7</td>
<td>100</td>
<td>7</td>
</tr>
<tr>
<td>cv.randomFr.v8</td>
<td>500</td>
<td>7</td>
</tr>
<tr>
<td>cv.randomFr.v9</td>
<td>1000</td>
<td>7</td>
</tr>
</tbody>
</table>
Table A.2  Support Vector Machines parameters.

<table>
<thead>
<tr>
<th>Name</th>
<th>Cost</th>
<th>Epsilon</th>
<th>Gamma</th>
</tr>
</thead>
<tbody>
<tr>
<td>cv.svm.v1</td>
<td>3</td>
<td>0.01</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v2</td>
<td>5</td>
<td>0.01</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v3</td>
<td>7</td>
<td>0.01</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v4</td>
<td>3</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v5</td>
<td>5</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v6</td>
<td>7</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v7</td>
<td>3</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v8</td>
<td>5</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v9</td>
<td>7</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svm.v10</td>
<td>3</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svm.v11</td>
<td>5</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svm.v12</td>
<td>7</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svm.v13</td>
<td>3</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svm.v14</td>
<td>5</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svm.v15</td>
<td>7</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svm.v16</td>
<td>3</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svm.v17</td>
<td>5</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svm.v18</td>
<td>7</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v1</td>
<td>3</td>
<td>0.01</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v2</td>
<td>5</td>
<td>0.01</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v3</td>
<td>7</td>
<td>0.01</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v4</td>
<td>3</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v5</td>
<td>5</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v6</td>
<td>7</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v7</td>
<td>3</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v8</td>
<td>5</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v9</td>
<td>7</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.svmr.v10</td>
<td>3</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v11</td>
<td>5</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v12</td>
<td>7</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v13</td>
<td>3</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v14</td>
<td>5</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v15</td>
<td>7</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v16</td>
<td>3</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v17</td>
<td>5</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.svmr.v18</td>
<td>7</td>
<td>0.1</td>
<td>0.01</td>
</tr>
</tbody>
</table>
### Table A.3 Neural Networks parameters.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Decay</th>
</tr>
</thead>
<tbody>
<tr>
<td>cv.nnet.v1</td>
<td>3</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.nnet.v2</td>
<td>5</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.nnet.v3</td>
<td>7</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.nnet.v4</td>
<td>3</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.nnet.v5</td>
<td>5</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.nnet.v6</td>
<td>7</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.nnet.v7</td>
<td>3</td>
<td>0.05</td>
</tr>
<tr>
<td>cv.nnet.v8</td>
<td>5</td>
<td>0.05</td>
</tr>
<tr>
<td>cv.nnet.v9</td>
<td>7</td>
<td>0.05</td>
</tr>
<tr>
<td>cv.nnetr.v1</td>
<td>3</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.nnetr.v2</td>
<td>5</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.nnetr.v3</td>
<td>7</td>
<td>0.1</td>
</tr>
<tr>
<td>cv.nnetr.v4</td>
<td>3</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.nnetr.v5</td>
<td>5</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.nnetr.v6</td>
<td>7</td>
<td>0.01</td>
</tr>
<tr>
<td>cv.nnetr.v7</td>
<td>3</td>
<td>0.05</td>
</tr>
<tr>
<td>cv.nnetr.v8</td>
<td>5</td>
<td>0.05</td>
</tr>
<tr>
<td>cv.nnetr.v9</td>
<td>7</td>
<td>0.05</td>
</tr>
</tbody>
</table>
Appendix B

Implementation Details

The work carried out in this thesis involved the development of a software system with the following main features:

- Extract documents from websites such as blogs and forums.
- Interface that allows an expert to easily label documents regards the sentiment.
- Train models that are able to label documents.

Currently, the system consists of individual applications performing the different tasks but the goal within the project ePolicy is to develop a single opinion mining module.

The technologies used to implement the system were chosen mainly because of the experience we have on them.

In this annex we begin by describing the technology and implementation of the crawlers used to extract the documents, then we give an overview of the website used to tag the documents. After that we show how we have implemented the bag of words representation
of the text documents, the models used to solve the problem and how we did the experimental comparisons. Finally, we present an example of a visualization technique that is applicable to this type of data.

## B.1 Post Crawlers

In order to extract documents from websites, two crawlers were created using Scrapy [19], one for each website. Two websites were chosen by the e-Policy project consortium: the EnergeticAmbient.it forum [16] and the newclear blog [17].

Scrapy is a fast high-level screen scraping and web crawling framework, used to crawl websites and extract structured data from their pages. It can be used for a wide range of purposes, from data mining to monitoring and automated testing.

Scrapy works by defining the Xpath that leads to the information we want and then assemble everything in an object that is going to be inserted on a database. As an example, the following code is used in order to obtain the text from a post on the Nuclear website.

```python
1 item['text'] = hxs.select('string('//div[@class="comment-body"])['+str(i)+']')).extract()
2 finStringText = ''.join([smart_str(x) for x in item['text']]).strip()
3 item['text'] = finStringText
```

Listing B.1: Crawler code.

On the first line we select the text by using Xpath and then extract it. On the second line we clean the text from html tags, blank spaces and other characters using the smart_str function from Python’s Django framework [18].

## B.2 Website for Tagging Documents

Since the data on the italian websites were not labeled (no sentiment score given by the users), we created a website in order to let a human tagger label some of the documents.

We used Python’s Django framework and its administrator interface, producing the results in Figures B.4 and B.5. This website features user accounts and a simple interface in which
a person can view the documents, order them by postcounter (if the post is the main post from the thread than it’s postcounter is ‘1’) or by date.

Django’s framework had most of the features already implemented and with minimal tweaks we got the desired tagging website. The database schema corresponding to the posts is described in Figure B.6.

Figure B.4: Tagging a post.

Figure B.5: Search page.
APPENDIX B. IMPLEMENTATION DETAILS

B.3 Representing Documents through Bags of Words

The function we created is based on the infra-structure of the R package tm [22] and takes the documents and generates a bag of words representation of these documents, using the parameters that we want. We can set the language of the documents, if everything should be set to lower case, remove punctuation and numbers, minimum number of characters, if a dictionary of words should be used, if stop-words should be removed and if we want to apply word stemming. The dictionary is useful because we need to use only terms that the models were trained with.

```r
generateTermMatrix <- function(data, lang='english', lower = FALSE, removePunc = FALSE, removeNumb = FALSE,
                               wordLen=c(4, Inf), dict=NULL,
                               stopw=function(x) removeWords(x, stopwords("italian")),
                               stem=function(x) stemDocument(x, language="italian")

{  
corpus <- Corpus(VectorSource(data), readerControl=list(language=lang))
DocumentTermMatrix(corpus, control=list(tokenize=scan_tokenizer, tolower=lower,
                                       removePunctuation=removePunc, removeNumbers=removeNumb,
                                       stopwords=stopw, stemming=stem,
                                       wordLengths=wordLen, dictionary=dict))
}
```

Figure B.6: Database schema.
B.4 Code of the Experimental Comparisons

We conducted a series of tests that provided us with some interesting results described in Section 4.6. The tests were carried out using the experimentalComparison() function from the infrastructure of the DMwR R package [14]. This infrastructure allows us to automatically test and compare different model variants, using some experimental methodology (in our case $10 \times 10$-fold cross validation) to estimate the values of some predictive performance metrics.

This section describes the code used to perform these tests after pre-processing the data. Since the code to train and test Random Forests, Support Vector Machines and Neural Networks is very similar, we omit that part of the code. For each approach, we start by describing the functions that need to be created in order for the experimental comparison infrastructure to work and then we show the final call to this infrastructure. We ended up with two R objects, each with the results obtained on the two approaches.

B.4.1 Two-step Approach

In the first place, we train the models to classify the topic by training a model $m1$ with the training data. Then, using the test data, we create a vector of predictions $p1$. After that we do the same to classify the sentiment by creating a model $m2$ and the predictions vector $p2$. If the first vector indicates that the post does not speak about the document then we ignore the prediction given by the model $m2$.

```r
### Classification

cv.randomF <- function(form, train, test, ...) {
  require("randomForest")
  ### Dataset with topic at 0 or DS
  temp <- train
  objective <- colnames(temp[1])
  for (i in 1:nrow(temp)) {
    if (temp[i, colnames(temp)[1]] != 'DS') {
      temp[i, colnames(temp)[1]] <- '0'
    }
  }
  temp <- droplevels(temp)
  m1 <- randomForest(form, temp,...)
```
p1 <- predict(m1, test, type='class')

train <- droplevels(train)
test <- droplevels(test)
m2 <- randomForest(form, train, ...)
p2 <- predict(m2, test, type='class')

for (i in 1:length(p2))
{
  if ((p1[i]) == 'DS') p2[i] <- 'DS'
}
eval.stats(form, test, p2)

### Regression

cv.randomFr <- function(form, train, test, ...)
{
  require("randomForest")
  ### Dataset with topic at 0 or DS
  temp <- train
  objective <- colnames(temp[1])
  for (i in 1:nrow(temp))
  {
    if (temp[i, colnames(temp)[1]] != 'DS')
    {
      temp[i, colnames(temp)[1]] <- '0'
    }
  }
  temp <- droplevels(temp)

  m1 <- randomForest(form, temp, ...)
p1 <- predict(m1, test, type='class')

  train[,1] <- as.numeric(as.character(train[,1]))
  train <- na.omit(train)

  require("randomForest")
  test <- droplevels(test)
m2 <- randomForest(form, train, ...)
p2 <- predict(m2, test)
eval.stats2(form, test, p2, p1)
}

Listing B.2: Two-step approaches code.
We then send both vectors to the `eval.stats` functions (Listing B.3). Lines 1-3 are the cost matrix used to evaluate the model. The `eval.stats` function processes the classification models predictions and gives their statistics. `eval.stats2` does the same for the regression models and it has the additional work of rounding the predictions. These functions (`eval.stats` and `eval.stats2`) are used in all the approaches.

```r
costmatrix <- as.data.frame(matrix(c(0,1,2,3,4,2.5,1,0,1,2,3,1.5,2,1,0,1,2,0.5,3,2,1,0,1.5,4,3,2,1,0,2.5,2,5,1.5,0.5,1.5,2.5,0), nrow=6))
rownames(costmatrix) <- c('-2','-1','0','1','2','DS')
colnames(costmatrix) <- c('-2','-1','0','1','2','DS')

# Matriz de custo
eval.stats <- function(form,test,preds) {
  objective <- unlist(strsplit(as.character(form),"\~"))[2]
  trues <- subset(test,,c(objective),drop=T)
  errsum <- 0
  for (i in 1:length(preds)) {
    errsum <- errsum + costmatrix[trues[i],preds[i]]
  }
  c(matriz_custo=errsum)
}

eval.stats2 <- function(form,test,preds2,preds1) {
  objective <- unlist(strsplit(as.character(form),"\~"))[2]
  trues <- subset(test,,c(objective),drop=T)
  preds <- c()
  errsum <- 0
  for (i in 1:length(preds1)) {
    if ((preds1[i]) == 'DS') preds <- c(preds,'DS')
    else if (preds2[i] >= 1.25) preds <- c(preds,'2')
    else if (preds2[i] >= 0.75) preds <- c(preds,'1')
    else if (preds2[i] >= 0.25) preds <- c(preds,'DS')
    else if (preds2[i] >= -0.25) preds <- c(preds,'0')
    else if (preds2[i] >= -0.75) preds <- c(preds,'DS')
    else if (preds2[i] >= -1.25) preds <- c(preds,'-1')
    else preds <- c(preds,'-2')
    errsum <- errsum + costmatrix[trues[i],preds[i]]
  }
  c(matriz_custo=errsum)
}
```
Listing B.3: Evaluation’s functions code.

The following code is the call to the experimental comparison infrastructure that conducts the experiences with the parameters we want.

```r
resultsTwoStepApproach <- experimentalComparison(Datasets,
                                                  c(variants('cv. randomF', ntree=c(100,200,300),
                                                   mtry=c(3,5,7)),
                                                   variants('cv. svm', cost=c(3,5,7), epsilon=c
                                                   (0.01,0.05,0.1), gamma=c(0.1,0.01)),
                                                   variants('cv. nnet', size=c(3,5,7), decay=c
                                                   (0.1,0.01,0.05)),
                                                   variants('cv. randomFr', ntree=c(100,500,1000),
                                                   mtry=c(3,5,7)),
                                                   variants('cv. svmr', cost=c(3,5,7), epsilon=c
                                                   (0.01,0.05,0.1), gamma=c(0.1,0.01)),
                                                   variants('cv. nnetr', size=c(3,5,7), decay=c
                                                   (0.1,0.01,0.05)))
                                                  ,CVsets)
```

B.4.2 Single-step Approach

In the single step approach, we do not separate the training of the topic and sentiment (Listing B.4). We just create a model $m$ trained with all the data and create a prediction vector $p$. For the regression models, we have to convert all 'DS' tags in the training set to numeric data (0.3 and -0.3). After that we send the vector to the eval.stats and eval.stats2 functions.

```r
### Classification

```r
cv.randomF <- function(form, train, test, ...) {
  require("randomForest")
  train <- droplevels(train)
  test <- droplevels(test)
  m <- randomForest(form, train,...)
  p <- predict(m, test, type='class')
  eval.stats(form,test,p)
}
```
### Regression

```r
cv.randomFr <- function(form, train, test, ...) {
  require("randomForest")
  train[,1] <- as.numeric(as.character(train[,1]))
  nas <- which(is.na(train))
  nasup <- nas[1:(floor(length(nas)/2))]
  nasdown <- nas[(ceiling(length(nas)/2)):length(nas)]
  train[nasup,1] <- -0.3
  train[nasdown,1] <- 0.3
  test <- droplevels(test)
  m <- randomForest(form, train, ...)
  p <- predict(m, test)
  eval.stats2(form, test, p)
}
```


Finally the following code makes the call to the infrastructure with the variants that we want to test.

```r
resultsSingleStepApproach <- experimentalComparison(Datasets,
  c(variants('cv.randomF', ntree=c(100,200,300),
       mtry=c(3,5,7)),
   variants('cv.svm', cost=c(3,5,7), epsilon=c(0.01,0.05,0.1), gamma=c(0.1,0.01)),
   variants('cv.nnet', size=c(3,5,7), decay=c(0.1,0.01,0.05)),
   variants('cv.randomFr', ntree=c(100,200,300),
       mtry=c(3,5,7)),
   variants('cv.svmr', cost=c(3,5,7), epsilon=c(0.01,0.05,0.1), gamma=c(0.1,0.01)),
   variants('cv.nnetr', size=c(3,5,7), decay=c(0.1,0.01,0.05))))
```

,CVsets)
B.5 Word Clouds Visualization

Using R’s package `wordcloud` [20], we created a visualization of the words contained in the posts. The following code is used to create this cloud. This visualization is very popular nowadays and provides a quick overview of the most popular words in a set of posts.

```r
generateWordClouds <- function(data, sparsity=0.99, ...)
{
  x <- removeSparseTerms(data, sparsity)
  mat <- as.matrix(x)
  vec <- sort(colSums(mat), decreasing=TRUE)
  names <- colnames(mat)
  d <- data.frame(word=names, freq=vec)
  wordcloud(d$word, d$freq, ...)
}
```
References


REFERENCES


