Word-based Classification of Political Text based on Speaker and Topic labels

Jeroen Verhoeven
ANR: 314467

HAIT Master Thesis

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF ARTS IN COMMUNICATION AND INFORMATION SCIENCES,
MASTER TRACK HUMAN ASPECTS OF INFORMATION TECHNOLOGY
AT THE SCHOOL OF HUMANITIES
OF TILBURG UNIVERSITY

Thesis committee:
Dr. M. M. van Zaanen
Dr. N. Kamoen

Tilburg University
School of Humanities
Department of Communication and Information Sciences
Tilburg Center for Cognition and Communication (TiCC)
Tilburg, Netherlands
December, 2013

“The problem with quotes on the internet, is that it is hard to verify their authenticity”

- Abraham Lincoln

Abstract

As large amounts of data become more easily available, the need for automated content analysis methods which can detect useful information becomes increasingly important. Based on an aspiration to extract useful information from company conference call transcripts, this thesis presents an investigation into the extent to which political texts can be classified according to speaker and topic labels. To this purpose, three distinct word-based classification approaches are employed; Naïve Bayes, Committee and the *tf*idf* Sequence Classifier. Furthermore, alternative versions of the dataset are created which explore the potential use of synonymous word meanings, after which the three classifiers are again applied to the new datasets.

The results show that the approach of merging synonymous terms has no positive effect on classification accuracy. Furthermore, regarding classification of speaker labels, the best prediction results for Naïve Bayes, Committee and the *tf*idf* Sequence Classifier are very similar. The best performances of each classifier do not differ significantly from each other, whereas all scores significantly outperform the baseline scores. When classifying topic labels, the best results for the Naïve Bayes approach are significantly less accurate, though still outperforming the baseline scores, whereas both Committee and the *tf*idf* Sequence Classifier achieve the best results, with no significant difference between the two.

Contents

Abstract iii

Contents iv

Chapter 1: Introduction 1
1.1 Research Questions. 3

Chapter 2: Theoretical Background 5
2.1 Company Conference Calls. 5
2.2 Text Classification in Political Applications. 6
2.3 Classification Approaches. 8
2.3.1 Naïve Bayes. 9
2.3.2 Committee. 9
2.3.3 Similarities between Naïve Bayes and Committee. 10
2.3.4 *tf*idf* Sequence Classifier. 11
2.4 Use of Synonyms. 13

Chapter 3: Experimental Setup 14
3.1 Data Collection and Data Preparation. 14
3.2 Naïve Bayes and Committee. 16
3.3 *tf*idf* Sequence Classifier. 17
3.4 Synonyms Mergers. 18
3.4.1 Creating Alternative Datasets. 20

Chapter 4: Results 21
4.1 Experiment 1, 2 & 3: Naïve Bayes, Committee and *tf*idf* Results. . 21
4.1.1 Speaker Classification Results. 22
4.1.2 Topic Classification Results. 22
4.1.3 Comparing Speaker and Topic Classification Tasks. 23
4.2 Experiment 4: Effects of Merging Synonyms. 24
4.2.1 Speaker Classification Results. 24
4.2.2 Topic Classification Results. 25
4.2.3 Comparing Speaker and Topic Classification Tasks. 26

Chapter 5: Conclusion & Discussion 27
5.1 Conclusion. 27
5.2 Discussion. 28

References 31

Appendix 1: Text Fragment Examples 34

Chapter 1

Introduction

When information is cheap, attention becomes expensive – James Gleick

With the enormous rise in data storage capacities over the last decades, it has become increasingly important to be able to store, retrieve and analyze data meaningfully, and thereby allowing specific focus of attention to that information which is relevant for a given purpose. As the amount of available data becomes too large for human observers to process, machine aided techniques offer a great opportunity for effective allocation of attention. In this regard, valuable contributions have come from the field of Natural Language Processing (NLP), which studies the extent to which computers can aid in or are capable of understanding natural human language, and includes topics such as speech recognition, machine translation, question-answering and conversational agents (Jurafsky & Martin, 2000).

A particularly useful topic in NLP when dealing with large amounts of data is automated content analysis. Although current text analysis methods are not yet capable of fully understanding the complexity of human language, new techniques for content analysis offer great opportunities for superficial but nevertheless effective investigations of large datasets, which is a specifically useful approach for enhancing or augmenting human analysis efforts (Grimmer & Stewart, 2013). The value of this emerging approach is evident from the wide range of subjects to which it is applied, such as automatic categorization of text-based documents regarding medical diagnosis (De Bruijn & Martin, 2002), political ideology (Gentzkow & Shapiro, 2010) or even sentiment expressions (Boiy & Moens, 2009).

In this thesis, automated content analysis is applied to the categorization of textual documents based on political speaker and topic. This focus originated from a collaboration with a fellow master student at the Accounting department of Tilburg University. His work examines indications of political ideologies in company-initiated conference calls between company managers and investors. Companies regularly organize these conference calls as a method for supporting open communication between company management and financial analysts, thereby facilitating a sharing of information which is valuable for company forecasts and investment planning (e.g. Bowen, Davis & Matsumoto, 2002). Both the company and its investors generally benefit from honest and full disclosures,

since a lack of information sharing is interpreted negatively and often leads to more negative assessments by the investment community (Hollander, Pronk & Roelofsen, 2010). Therefore, opportunities to extract as much information as possible from company conference calls are in the interest of both the company and its investors.

Our assumption is that one topic of information that might be particularly useful to extract from these conference calls is indication of political ideology. We expect that a company's awareness of political happenings is likely to influence policy making by that company, which therefore constitutes valuable information for investment purposes. The analysis of company awareness of the political climate is especially useful on a very large scale, where an investigation into the disclosures of many different companies could offer insights that would not be noticeable by investigating disclosures from a single company. For example, the knowledge that one single company expresses a favorable attitude towards a certain political figure or a political movement would not be of great interest. However, if this preference is adopted by a significant number of other companies, thereby indicating a wider shift in attitude towards a certain political ideology, this information would become very valuable to the investment community.

Because information extraction on such a large scale is typically very difficult to achieve by human observers, the application of automated content analysis on this specific topic would be of great value. Not only are content analysis methods capable of dealing with very large amounts of data, but these methods also have already been successfully applied on the topic of identifying political ideology (e.g. Gentzkow & Shapiro, 2010). Therefore, the goal of this thesis is to explore to what extent automated content analysis can be employed to detect political ideology, by identifying political speaker (such as, for example, Barack Obama, John McCain and Mitt Romney) and political topic (such as, for example, Crime, Drugs and Education) in a large textual dataset.

In order to achieve this goal, the content analysis of a textual dataset concerning political ideology is approached as a machine-learning based classification problem. In classification problems, the challenge is to discover an approximation of the function that describes the division of the existing classes (a class can be a political speaker or topic, e.g. a text document concerning Barack Obama is labeled as belonging to the class 'Obama'). In other words, when given a new observation (i.e. a new text document), the classifier should be able to predict to which of the classes this observation belongs (Joachims, 1997).

This prediction is based on what the classifier has learned from a set of example observations, called the training data, which is often manually classified by a human observer (Manning, Raghavan & Schütze, 2008). Because this approach relies on training examples which are already classified, it is called a supervised learning method (as opposed to unsupervised learning, where the computer learns from examples that are not yet classified). The success rate of any classifier can be evaluated by attempting to predict the classes of a new dataset, called the test data, of which the correct classifications are known to a human observer but not to the classifier (Witten, Frank & Hall, 2011).

This will not be the first time that machine learning techniques for classification have been applied to politically oriented texts, although most of the existing research only focuses on political ideology (i.e. Republican versus Democrats) and not on political speaker or topic. Furthermore, most applications

available in the relevant literature use algorithms trained on formal sources of texts, such as official transcripts from the Congressional Record (Gentzkow & Shapiro, 2010) or manifestos of Liberal Democrat and Conservative parties (Laver, Benoit & Garry, 2003). The tone used in these type of documents is likely to be very different from the tone used in conference call meetings, which is typically considered to be more informal and spontaneous (Larcker & Zakolyukina, 2012).

Therefore, this research contributes to the existing literature in two ways. First, the document collection used in this thesis has many more distinct class labels compared to prior research approaches, which makes the classification much more challenging. Whereas it is known that certain words or phrases exist that can typically be associated with a certain political party (such as the Democratic use of the term "pro-choice" versus the Republican use of the term "pro-life) (Gentzkow & Shapiro, 2010), it remains a question whether a similar machine learning approach can be used to not only distinguish between parties, but also between individual politicians. Second, the sources used to create the dataset for this thesis are very diverse and much more informal compared to the data typically used in other research papers. It is interesting to investigate to what extent such informal text documents can still be used for successful classification, because such an approach would likely be applicable to a wide range of subjects, including conference call meetings.

1.1 Research Questions

The machine-learning methods that are applied in this thesis attempt to predict classification of new documents based on the occurrence of the words in those documents. Therefore, the goal of this thesis can be formulated as follows:

> *To investigate to what extent word-based classification methods can successfully predict political speaker and political topic labels of a new, unseen text document.*

In order to achieve this goal, four different experiments are conducted. In the first two experiments two different probability measures for predicting a document's classification are employed, based on the word occurrences in that document. Both these methods are relatively simple but are known to yield surprisingly good results (e.g. Zheng, 1998). In the first experiment, the well-known Naïve Bayes algorithm is applied to predict classification based on political speaker and on political topic. Based on this approach, the following research questions are formulated:

> *1.a To what extent can the Naïve Bayes algorithm successfully predict classification based on political speaker?*
>
> *1.b To what extent can the Naïve Bayes algorithm successfully predict classification based on political topic?*

In the second experiment the Committee algorithm is applied, which is based on the Naïve Bayes approach, although it employs a different technique for

measuring probabilities. Based on this approach, the following research questions are formulated:

> *2.a To what extent can the Committee algorithm successfully predict classification based on political speaker?*
>
> *2.b To what extent can the Committee algorithm successfully predict classification based on political topic?*

In the third experiment a classification approach based on the *tf*idf* metric is applied, which is a technique that originated in the field of Information Retrieval. When applied to text classification, the *tf*idf* Sequence Classifier can be used to identify similarities between documents based on weighted occurrences of the words in that document (e.g. Joachims, 1997). Based on this approach, the following research questions are formulated:

> *3.a To what extent can the tf*idf Sequence Classifier successfully predict classification based on political speaker?*
>
> *3.b To what extent can the tf*idf Sequence Classifier successfully predict classification based on political topic?*

In the fourth and final experiment, investigations focus on whether the results obtained in the previous three experiments can be improved by performing modifications on the original dataset. These modifications are based on the occurrences of synonymy between terms in the data, which is an approach that has yielded positive results in previous research (e.g. Scott & Matwin, 1999). Several modified versions of the original dataset are created, each differing in how often synonyms are merged into a new instance. Then all three machine-learning algorithms are applied to these altered datasets, to see whether the modifications have any merit for classification purposes. Based on this approach, the following research questions are formulated:

> *4.a To what extent can prediction of political speaker classification be enhanced by modifying the dataset based on term synonymy?*
>
> *4.b To what extent can prediction of political topic classification be enhanced by modifying the dataset based on term synonymy?*

The next chapter gives an overview of the existing research relevant to this thesis, including a detailed description of the classification approaches used in this thesis. In the third chapter, the setup of the experiments will be described, followed by a chapter presenting the results. The final chapter of this thesis provides a general conclusion, and finishes with a discussion of the results.

Chapter 2

Theoretical Background

This chapter provides the theoretical background that is relevant for the setup and execution of the four experiments. First a detailed description is given on the topic of company conference calls (where the motivation for this research originated), and how automated text analysis approaches are already being applied in this field. Secondly, the existing approaches to text classification focused on political applications are reviewed. In the third and final part of this chapter, the techniques that were used for classification are described: the Naïve Bayes and Committee classifiers, the *tf***idf* Sequence Classifier, and the potential value of modifying the dataset based on the occurrence of synonymous terms.

2.1 Company Conference Calls

Company-organized conference calls between managers and investors are an increasingly popular method for companies to communicate with their investors, because investment professionals consider oral communications with corporate managers as very important sources of information for evaluating company value (Balakrishnan, Qiu & Srinivasan, 2010). The implementation of conference calls differs among companies, as some companies organize multiple conference calls each year and invite a large number of investors to participate, whereas others only host conference calls once a year to a select group of participants. However, despite these variations in implementation, research has shown that conference calls have become an important mean for investors to forecast company earnings, due to the fact that these calls typically provide new and insightful information about a company (Bowen, Davis & Matsumoto, 2002).

In accounting research, it is generally accepted that both companies and investors are best served when the information disclosed in conference calls is complete and truthful; withheld information or deceptive narratives are interpreted as unfavorable news causing investors to make negative analyses, which in turn negatively affect a company's market value (Hollander, Pronk & Roelofsen, 2010). This need for information is so strong that increasingly often investors employ the special services of business analysts to participate in the conference call and extract as much information from the conversation as possible. Some of these analyst agencies are even known to employ former CIA operatives who are trained in detecting untruthful disclosures.

Because this demand for information is so high, and because both investors and companies are likely to benefit from more information sharing, it would be in the interest of both parties when any additional analysis methods can help in extracting information that would otherwise be difficult to detect. For this very purpose, an increasing amount of attention is paid to new techniques coming from the field of NLP. In fact, some NLP techniques that extract linguistic information from large text corpora have been successfully employed in this particular field already.

For example, Larcker and Zakolyukina (2012) analyzed linguistic features in managerial statements made during conference calls and classified these statements as truthful or deceptive. They accomplished this by taking classical linguistic features associated with deceptive narratives (e.g. more plural pronouns like "we" or "anybody", and references to general knowledge like "everyone knows that") and examining their occurrences in the available statements. The results showed that searching for and counting these features allowed the researchers to automatically discriminate between deceptive and truthful statements in conference calls, a result which would otherwise be very difficult for human analysts to obtain.

Similar research by Balakrishnan, Qiu and Srinivasan (2010) investigated whether managerial narratives contain information that can be used to predict market performance. The researchers gathered a large dataset of narrative transcripts which were classified based on performance measures (i.e. company market returns), and then identified words that occurred more often in any of the classes. After eliminating words that were of little discriminative power because they either occurred very little or very often, the most useful words were collected in a vocabulary list (also called a bag-of-words collection) which was capable of distinguishing between successful or unsuccessful companies. The researchers were then able to use these words to predict performances of unknown companies based on the words occurring in their annual reports.

2.2 Text Classification in Political Applications

Apart from applications in conference call related research, NLP based approaches have also been successfully applied to identify political ideology, in several different capacities. In general, the research on classifying political texts can be divided into either dictionary-based methods or machine learning-based methods (Grimmer & Stewart, 2013). Dictionary-based methods rely on a pre-determined list of keywords considered to be important to a certain class (e.g. a list of words that are commonly associated with the Democratic Party), which are simply used to count the occurrence of these words in a given text to determine classification. Machine learning-based methods rely on a set of documents classified by a human observer, which are used to train a learning algorithm and allow this algorithm to classify new texts based on probability models (e.g. determine the probability that a new text belongs to a class dependent on the occurrence of other texts belonging to that class).

A good example of a dictionary-based classification approach comes from Maynard and Funk (2012), who attempted to detect expressions of political opinions posted on the online social media platform Twitter, and classify them as being positive, negative or neutral. After gathering a large collection of Tweets, the researchers were able to find expressions of opinion in the form of *Person*,

Opinion and *Political Party* (e.g. "*Bob* is *terrified* of the *Republican* view on the war in Iraq"), in which case the *Opinion* and *Political Party* features were found using a list of pre-determined keywords. Though these classifications of opinion were not as accurate as would be when classified by a human observer (the algorithm correctly classified opinions in approximately 62% of the cases), the researchers argued that the potential capacity for this approach to quickly analyze large amounts of data is certainly useful for commercial purposes.

However, these classification results might be even lower when looking at more formal forms of text documents, as argued by Yu, Kaufmann and Diermeier (2008). These researchers draw attention to the fact that opinions uttered through social media typically contain strong emotional words to express sentiment (often adjectives or adverbs such as 'stupid' or 'terrible'), which makes these types of text fragments easier to classify compared to more formal texts with much fewer expressions of emotional words, such as political debate transcripts or news coverage items. As an alternative, Yu, Kaufmann and Diermeier suggest that instead of searching for expressions of opinions, it might be more useful to search for indication of political ideology. The idea behind this suggestion is that opinions are often correlated with political ideologies, as a person with favorable views towards topics such as abortion, healthcare and green energy (i.e. typical examples of a Democratic ideology) is likely to also have a favorable view towards stricter gun control.

Therefore, Yu, Kaufmann and Diermeier applied two different machine learning-methods, namely the Naïve Bayes and Support Vector Machines algorithms, to classify documents containing transcripts of U.S. Congressional speeches. The results showed that this approach was suitable for identifying Democratic and Republican ideologies, and that predictions were most accurate when using training documents from more 'extreme' politicians (i.e. the most conservative or most liberal Senators). Furthermore, predictions were found to be less accurate when a large time difference existed between the training and test data, for example by training on transcripts dated several years earlier than the ones used for testing. This finding was explained by the fact that documents from the same period are much more likely to address the same political issues, as the importance of such issues tends to change over time.

A similar approach to analyzing political texts, but applied to a different domain, comes from the research by Gentzkow and Shapiro (2010). These researchers investigated expressions of political ideology in American newspapers by searching for typical language associated with either Republican or Democratic parties. For this purpose, the authors applied a machine learning-based method to the 2005 Congressional Record to identify short phrases (i.e. word combinations existing of either two or three consecutive words) that were strongly associated with one of the two parties, and evaluated their predictive strength based on how much more often they occur in either Democratic or Republican statements. This yielded word combinations such as 'tax relief', typically associated with Republican politicians, or 'tax break', which is a typical Democratic phrase. These sets of words were then compared to the words occurring in several popular American newspapers, to see if they could be used to classify the ideologies associated with each newspaper. The results revealed that the predicted ideologies showed great correlation with the known ideologies of the region in which the newspapers were distributed, and were therefore a strong measure for recognizing political preferences in written texts.

These investigations show that automated content analysis can successfully be applied to identify political ideology, although it must be noted that most of the existing literature focuses on ideology in the form of Democrats versus Republicans, and not specifically on political speakers. A final research article that is worth mentioning comes from Purpura and Hillard (2006), who applied a Support Vector Machine classifier to categorize text documents based on political topic (topics included, amongst others, health, education, social welfare and foreign trade). The data used for this task consisted of a large collection of Congressional Bills, divided over 21 topic categories. For classification purposes, the data was represented as a bag-of-words (i.e. no attention was paid to the order and context of the words), which was used to compare and label new text documents. Results showed that the classifier achieved an average prediction accuracy of 71%, with upper-bound performances as high as 88%. The researchers concluded that especially for classes with many training documents, the prediction accuracies were as high as could be expected from human observers. To our knowledge, this is the only example of classification of political texts based on topic.

2.3 Classification Approaches

Although the examples described in the previous sections show that text categorization can be achieved in different ways, the goal is always the same – to discover the function that describes how to assign a new document to a fixed number of classes (Joachims, 1997). This function can be found or at least approximated by providing a supervised learning algorithm with an example collection of correctly classified documents.

Formally, we define a document $d \in \mathbb{X}$ (where $\mathbb{X}$ is the document space, i.e. all documents in the collection), and a fixed set of classes $\mathbb{C} = \{c_1, c_2, \ldots, c_j\}$, which are defined by a human observer. For classification, we need a training set $\mathbb{D}$ of labeled documents $[d, c]$, where $[d, c] \in \mathbb{X} \times \mathbb{C}$ (Manning, Raghavan & Schütze, 2008). The information contained in the training set can be used to find or approximate the function that maps the documents to the different classes. The objective is to find that function which maximizes accuracy when predicting the classes of new documents.

Although there are many supervised learning algorithms that can achieve this task, the focus in this thesis lies on three distinct approaches; the Naïve Bayes, Committee and *tf*idf* Sequence Classifier algorithms. The reason for using Naïve Bayes is that it is a very common approach in this field, and any investigation into text classification would almost be incomplete without looking at the performances of a Naïve Bayes classifier. The reason for using the Committee classifier is that it offers an alternative approach for classifying documents, though it still is based on the same probability measures that make Naïve Bayes so effective. Differences in the results between these two classifiers could yield interesting insights into the differences between the classifiers themselves. Finally, the motivation for using the *tf*idf* Sequence Classifier is that it has the capacity to process a wide variety of patterns. In this thesis the focus is limited to patterns in the form of up to three consecutive words (see section 3.3 for a detailed description), but the capacity to process much longer sequences makes this an interesting approach for initial investigations, with ample opportunities for extended future research.

2.3.1 Naïve Bayes

The first method used in this research, Naïve Bayes, is a well-known and often applied learning method (see, for example, Jurafsky & Martin, 2000, or Grimmer & Stewart, 2013), which employs a powerful but relatively simple approach for learning the relationship between categories and words. The Naïve Bayes method is based on the premise that choosing the best class *c* out of a set of possible classes ℂ for a document *d* simply depends on choosing the most probable class for that document (Jurafsky & Martin, 2000). This choice is based on the words that appear in each document, and in order to accomplish this the method 'naively' assumes that the probability of a word's appearance is independent of the word's context and position related to other words (e.g. it would not be possible to distinguish between the sentences "John likes Mary" and "Mary likes John"). This assumption yields the bag-of-words representation of a document, and although this representation is not viable in most real-world situations, the Naïve Bayes method often performs surprisingly well in text classification tasks (McCullum & Nigam, 1998). The probability that the label of class *c* should be assigned to an observed document *d* is computed using the following rule:

$$P(c|d) \propto P(c) \prod_{i=1}^{N} P(t_i|c),$$

where $P(c)$ is the proportion of documents belonging to class *c* (i.e. the prior probability of a class). This proportion can be seen as a weight that indicates the relative frequency of the class; more frequent classes are more likely to be the correct choice than infrequent classes. The probability $P(t_i|c)$ is the number of times term t_i appears in documents labeled with class *c*, divided by the total number of terms in documents labeled with *c* (i.e. the likelihood that a word belongs to a class). In other words, $P(t_i|c)$ is a measure for how much evidence t_i contributes to the probability of the document belonging to class *c* (Manning, Raghavan & Schütze, 2008). The class with the highest score can then be assigned to a document, as this is the best (i.e. most probable) class based on the terms in that document.

2.3.2 Committee

A simple variation to the well-known Naïve Bayes method is the Committee algorithm, which is based on the idea of combining multiple classifiers and having these classifiers vote for a particular class (similar to a voting committee). Committee classifiers are based on the general notion that, given a certain classification task, multiple classifiers are likely to perform better than one, as long as their predictions are effectively combined (Sebastiani, 2002). In the algorithm employed in this thesis, each individual term *t* in a document *d* is considered as a committee member voting for a particular label. The classification which receives the most votes is chosen to be the most probable solution (Bishop & Nasrabadi, 2006). Suppose we employ *N* classifiers (i.e. *N* number of terms *t*, each voting for a label *c*) and then use each classifier to train on a separate subset of the original dataset and learn the class distribution (c,t) where $t = 1, 2, \dots, N$. Out of all classes, we are searching for the class that is most probable given the observations of all the Committee members

$$\hat{c} = \text{argmax}_c \frac{1}{N} \sum_{i=1}^{N} \log(P(c|t_i)),$$

where we assign class *c* to a document when the sum of the votes of individual committee members (i.e. the sum of the scores of each individual term *t*) is maximized for that class. In this formula, we can ignore the *1/N* factor, as the number of features (i.e. the number of words) does not change for the same observation as long as we are examining the same single document. Furthermore, we can rewrite the formula by exponentiating the estimate of the most probable class

$$\hat{c} = \text{argmax}_c \, e^{\sum_{i=1}^{N} \log(P(c|t_i))}$$

$$= \text{argmax}_c \, e^{\log(\prod_{i=1}^{N} P(c|t_i))}$$

$$= \text{argmax}_c (\prod_{i=1}^{N} P(c|t_i),$$

where $P(c|t_i)$ can be calculated by taking the number of times term t_i occurs in documents labeled with class *c*, divided by the total number of occurrences of term t_i. The label with the highest score is assigned to the document.

2.3.3 Similarities between Naïve Bayes and Committee

Although the Naïve Bayes and Committee methods use distinctly different approaches for calculating the most probable classification of a given document, both applications also share some strong similarities. Both assume that the occurrences and positions of words are independent, thereby representing the document as a bag-of-words. This allows both methods to predict classification based not only on the words that occur in a document, but also on the number of times a word occurs in that document (Witten, Frank & Hall, 2011).

Since both the Naïve Bayes and Committee algorithms require the data to be represented as a bag-of-words, it is impossible to derive any contextual information from the words themselves. For example, any representations of the renowned opposing political positions "pro-life" and "pro-choice" would be lost, as the system would only count the occurrences of the individual terms "pro", "life" and "choice". A possible solution for this problem comes from Pang, Lee, and Vaithyanathan (2002), who suggested the use of different *n*-grams for classification. An *n*-gram is a subsequence of *n* characters (Jurafsky & Martin, 2000), such as a bigram (e.g. "I_am") or a trigram (e.g. "this_is_the"). Using *n*-grams provides a rough but simple way to gain contextual information, and is used for example by Pang, Lee, and Vaithyanathan (2002) to account for negations; the contextual difference between the expressions "support" and "not_support" can be captured using simple bigrams.

The application of multiple n-grams might be particularly useful in the domain of political text classification, as can be seen in the work by Gentzkow and Shapiro (2010). As mentioned before in section 2.2, their research attempted to classify political ideology in American newspapers, and their method yielded many bi- and trigrams which the authors identified as prototypical democratic or republican expressions, such as "budget_deficit" or "war_in_Iraq". These short

sequences represented a variety of political viewpoints that were characteristic of a certain political affiliation, and therefore very useful for accurate classification.

Another similarity between these two classifiers which is worth mentioning is that both approaches share a general weakness for zero-occurrences. Whenever either one of the two methods encounters a word that did not occur in the training data for a particular class, the score for that class is multiplied with a value of zero, leading to a total score of zero as well. Since sparse words (i.e. words that occur only once) are very common in large document collections, the effects of words occurring in only one class and thereby causing zero-occurrences in other classes is a serious problem. One possible solution to this problem is to apply Laplace smoothing (also called add-one smoothing, see: Jurafsky & Martin, 2000), which is a technique for simply adding a value of 1 to each word's count. This will ensure that words occurring zero times will receive a probability score which is non-zero, though still very small.

Both methods depend greatly on a word's capacity to provide useful information to effectively distinguish between different classes. Words that are very common in the training set (like "the" or "a"), or words that are very rare, typically have no discriminative value and therefore are of little use when predicting a document's label. Many applications of the Naïve Bayes and Committee classifiers therefore employ a technique called feature selection (Manning, Raghavan & Schütze, 2008), which is a process of selecting only a subset of the words from the training data, based on their capacity to effectively discriminate between different classes. Words that fall outside of this subset are eliminated, which not only improves the speed of the classifier since there are fewer words in the vocabulary, but generally also improves classification accuracy.

Overall, the characteristics of the Naïve Bayes and Committee methods as described above make these approaches a suitable and commonly applied solution for many different classification tasks. In general, these methods are considered to be very easy to apply while yielding relatively accurate classification results (Manning, Raghavan & Schütze, 2008). Consequently, these methods offer great potential for automated content analysis of company conference calls.

2.3.4 *tf*idf* Sequence Classifier

The Sequence Classifier is a supervised learning algorithm which tries to find similarities between documents based on the idea that the importance of a pattern in a document can be weighted using the statistics of occurrence of that pattern (Manning, Raghavan & Schütze, 2008). These individual weights can then be used to score the entire document, and to determine to which class the document belongs. This idea originated in the field of Information Retrieval, where the relevance of a document to a given search query is measured by applying the commonly used *tf*idf* metric (e.g. Joachims, 1997). Although originally used to quantify similarities between a document and a search query, this technique has successfully been applied in many other fields, including for example music classification (Van Zaanen & Gaustad, 2010) and text classification (Joachims, 1997).

The *tf*idf* metric consists of two factors, the first of which is the term frequency, or *tf*, which simply measures the number of times a term *t* occurs in a document *d* (or, when applied to text classification, the number of times the term

occurs in a class *c*). The *tf* weight in itself is not a very useful approach for determining relevance between a term and a class, as very common terms will automatically have very high *tf* values, even though such terms are not useful for discriminating between classes. Therefore, the second factor in the *tf*idf* metric is the inverse document frequency, or *idf*, which is a measure for how rare a term is in the entire document collection. Given a document collection *N*, the *idf* of a term is defined as:

$$idf_t = \log \frac{N}{df_t},$$

where df_t is the number of documents in which term *t* appears, which will be high for rare terms (i.e. words that occur in only a few documents), but low for common terms (i.e. words that appear in almost every document). Combining these two factors allows for the identification of terms that are most useful for discriminating between classes, because these terms occur frequently in the given class (term frequency) but not frequent in other classes (inverse document frequency) (Balakrishnan, Qiu & Srinivasan, 2010). This metric can now be used to compute a weight for each word in the training data with respect to each class. These weights can then be used to compute a score *s* for a new document *d*, by adding the weight of each word in that document

$$s\,(d,c) = \sum_{t \in d} tf_t * idf_{t,c},$$

where the document *d* remains the same, but the classes *c* change, so that a similarity score can be computed between the one document and each existing class. After computing the scores for each class *c*, the class most similar to the document will be assigned to that document. In order to compare similarities between documents and classes, the *tf*idf* scores for the different classes and for the documents to be classified are represented as vectors (Manning, Raghavan & Schütze, 2008). By representing the documents and classes as vectors, similarities can be identified regardless of the size of a document or a class. To find the similarity, we define a vector $\vec{V}(d)$ representing document *d* (or a class *c*), with one component in the vector for each unique term in the document or class (these components are equal to the *tf*idf* weight). The quantified similarity between a document *d* and a class *c* can be found by computing the cosine similarity of the vector representations $\vec{V}(d)$ and $\vec{V}(c)$:

$$\text{sim}(d,c) = \frac{\vec{V}(d) \cdot \vec{V}(c)}{|\vec{V}(d)||\vec{V}(c)|},$$

where the denominator is the product of the Euclidian lengths of the vectors, which is a common technique for length-normalizing the vectors. Although this method is very useful for comparing document similarities, it discards all information about class sizes (Manning, Raghavan & Schütze, 2008). As larger classes will have more terms in them, the *tf* will be higher as well, which inappropriately raises the score for that class. To compensate for this effect, an often applied technique is to length-normalize the *tf*idf* scores, so that when comparing the score between a document vector and a normalized class, the score is adjusted to account for the differing sizes of the available classes.

2.4 Use of Synonyms

A commonly addressed issue associated with text classification is the use of synonyms. Two words are synonymous to each other when they share the same meaning, and therefore can be used to substitute each other in a given sentence without changing its meaning (Jurafsky & Martin, 2000). It should be noted, though, that synonyms only share partial meaning, and are not completely identical but rather can only substitute each other in a sentence given a certain appropriate context. For example, when attempting to classify documents that contain information about the topic *aircrafts*, it could be useful to base class predictions on the synonymous term *plane*. However, the word *plane* would only be useful for proper classification in the right context, as it would not be beneficial to classify documents based on the occurrence of the word *plane*, when these documents actually refer to the term in the sense of something that is even or smooth.

A suitable example of incorporating synonyms in text classification tasks comes from Rodriguez, Gomez-Hidalgo and Diaz-Agudo (1997), who attempted to increase classification accuracy by expanding the training set with additional words in the form of synonyms. To find appropriate synonyms they relied on WordNet, which is a large online lexical database, commonly used to access semantic relationships between words such as synonyms or antonyms (Miller *et al*., 1990). They succeeded in improving their results by finding synonyms to the names of the labels to be classified, and by increasing the weights of these synonyms in the learning phase. However, the researchers found that these improvements were only possible due to the fact that the synonyms were chosen manually (in order to understand the context and choose the appropriate synonyms), which was only feasible because they were working with a relatively small dataset.

Similar to this research, another attempt to expand the training data with synonyms comes from Scott and Matwin (1999), who created an algorithm which automatically selected synonyms so that processing of larger data sets would become possible. The difficulty with automated synonym selection is choosing the correct sense for a word given its context, which in the field of NLP is called word sense disambiguation. Scott and Matwin tackled this problem by examining hypernyms, which are words that share a 'type-of' relation (e.g. a *knife* is a type of its hypernym *weapon*). They apply this method by replacing words in the training data with their hypernym, thereby effectively reducing the size of the training data. So instead of replacing the word *knife* with all its synonyms (possibly including unwanted synonyms such as *sword* or *bayonet*), they replace words like *knife* or *gun* with their hypernym *weapon*, creating a smaller database grounded in the synonymous relations between the original words. Their results showed that this approach is effective in a Committee-like approach, where one classifier is trained on the original data, and one classifier is trained on the data containing hypernyms, after which the average scores are taken to label the documents.

Overall, there are potential gains to be made by adding synonyms to the training data, but the challenge is to only use the synonyms that are appropriate in a given context. The next chapter describes how this difficulty was dealt with, along with a description of the implementation of the three classifiers: the Naïve Bayes, Committee and Sequence Classifier algorithms.

Chapter 3

Experimental Setup

This chapter provides a description of the data collection and preparation, including the division over classes based on political speaker and topic. Furthermore, a description is given of the implementation of the Naïve Bayes and Committee algorithms, and of the implementation of the Sequence Classifier. Finally, the fourth and last section of the chapter elaborates on the creation of a new dataset to investigate the potential use of synonyms.

3.1 Data Collection and Data Preparation

The dataset that was used in this thesis contained 5450 documents, which were manually downloaded from an American politics oriented website[1]. This website is designed to present the general public with information about political candidates, news topics and public opinions, by providing a wide variety of relatively short text fragments categorized on political speaker and topic. The sources for these text fragments are very diverse, ranging from excerpts from political speeches and debates, to news articles, interviews and even passages from personal memoires. Some example fragments are given in Appendix A.

The variety in sources is matched in the variety of text sizes; the shortest document in the collection is a mere 7 words long, whereas the longest document counts 4109 words. Each document is classified as belonging to one of six

Table 3.1

Number of documents per category, classification based on political speaker

Classification	Class	# of documents	# of tokens
Political Speaker	Al Gore	706	87,402
	Barack Obama	1,338	219,133
	George W. Bush	1,148	149,115
	John Kerry	584	124,809
	John McCain	957	158,357
	Mitt Romney	717	109,013
Total		5,440	847,829

[1] www.ontheissue.org

possible political speakers, and as one of twenty-four possible political topics. For the purpose of classification, the classifiers discussed in this thesis are applied to the dataset twice, once for classifying speaker (see table 3.1) and once for classifying topic (see table 3.2). The sizes of the different classes vary considerably; for the speaker labels, the largest class comprises 24.55% of all documents, whereas the smallest class comprises 10.72% (a uniform division would equal 16.67% of the documents in each class). Regarding the topic labels, the largest class comprises 12.22% of all documents, whereas the smallest class comprises 1.63% (a uniform division would equal 4.17% of the documents in each class).

After all the documents were collected and categorized, the data was prepared as input for the three algorithms by removing all punctuation and special characters, and by changing all letters into lowercase form. Each class now contained all terms from documents categorized as belonging to that class, represented as a bag-of-words. No stemming was used (i.e. removing the ends of words in order to reduce vocabulary size) (Grimmer & Stewart, 2013). The final and full dataset contained a total of 847,829 tokens (i.e. single words) and 20,564 types (i.e. unique words).

Table 3.2

Number of documents per category, classification based on political topic

Classification	**Class**	**# of documents**	**# of tokens**
Political Topic	Abortion	167	30,215
	Budget & Economy	242	38,710
	Civil Rights	245	35,765
	Corporations	109	17,186
	Crime	109	13,980
	Drugs	89	11,749
	Education	301	38,694
	Energy & Oil	265	47,513
	Environment	152	23,488
	Families & Children	95	11,968
	Foreign Policy	295	44,815
	Free Trade	118	18,109
	Government Reform	272	47,189
	Gun Control	93	13,986
	Health Care	391	59,673
	Homeland Security	406	63,536
	Immigration	165	30,746
	Jobs	148	24,319
	Principles & Values	666	99,210
	Social Security	127	17,210
	Tax Reform	222	39,634
	Technology	156	23,255
	War & Piece	510	81,459
	Welfare & Poverty	107	15,420
Total		5,450	847,829

For the purpose of measuring classification accuracy, the dataset was split into ten equal parts (called folds), which is necessary in order to apply the common evaluation approach known as tenfold cross-validation. In general, an effective classifier benefits from as much data as possible for training, but in order to obtain an accurate error estimate, as much data as possible is required for testing (Witten, Frank & Hall, 2011). The solution to this problem is to repeat the training and testing processes several times with different samples. This can be achieved by dividing the available data into *N* equal folds, and repeating the process of training and testing an *N* number of times (Jurafsky & Martin, 2000). In the case of tenfold cross-validation, the data is divided into ten equal folds, of which nine folds are used for training and one fold is used for testing the classifiers, after which an estimate on the error rate is computed. This process is repeated ten times, using a different fold for testing each time, until finally an average error rate can be computed.

3.2 Naïve Bayes and Committee

The Naïve Bayes and Committee algorithms both share strong similarities in their estimation of the most probable class label. As mentioned before, the Naïve Bayes calculations require the proportion of documents in each class, the total number of words in each class, and for every word the number of occurrences of that word in each class (see section 2.3.1). The Committee calculations require the number of occurrences of a word in each class, and the total number of occurrences of that word (see section 2.3.2). During the experiments, for each word found in the training data, these values were simply counted and stored. Since the Naïve Bayes and Committee classifiers are not resistant to zero-occurrences (see section 2.3.2), Laplace smoothing was applied to increment all counts with a value of 1. Furthermore, for both classifiers, logarithmic scales were used to represent probability scores, in order to prevent probabilities to become too small to use for calculations.

As described in section 2.3.3, previous research has successfully applied the use of *n*-grams to mitigate the loss of context (such as negations), and to include short phrases that are in general strongly associated with certain political views. Therefore, this approach is applied here as well by expanding the vocabulary with 2- and 3-gram word sequences. However, as early trial runs quickly showed, both the Naïve Bayes and Committee classifiers tend to perform better when the vocabulary is of a modest size. Adding 2- and 3-grams greatly increased the total number of words in the vocabulary (increasing the number of types from 20,564 to 818,651), and therefore yielded worse results than would be expected based on prior research. In order to maintain an appropriately sized vocabulary while simultaneously incorporating useful *n*-grams, the choice was made to only include bigrams and trigrams with high discriminative values between the classes. In order to find these words, a strict process of feature selection was applied.

As mentioned in section 2.3.2, feature selection is the process of selecting a subset of words from the training data that are most useful for classification (Manning, Raghavan & Schütze, 2008). Three commonly used feature selection methods are removing infrequent words, removing high frequency words, and removing words that do not share high mutual information with the target classes (Joachims, 1997). In this thesis two of these methods were applied, the first of

which is the removal of infrequent words. This means that for all unigrams, bigrams and trigrams, all words or word combinations that occur less than three times were removed from the vocabulary.

The second method that was applied is the removal of words that share little mutual information with the target classes. For each word, the number of times this word occurred in each of the classes was counted, as well as the total number of times this word occurred in the training data. These values can be used to determine how much discriminative information a certain word holds for a particular class. For example, a very common word like "the" occurred on average (depending on the fold) a total number of 39,703 times, and was divided relatively equal over all classes (e.g. for the first fold in the speaker classification task: 4,181 – 9,557 – 7,233 – 6,162 – 7,748 – 4,822). Due to its even distribution over the classes, the word "the" offered very little discriminative power and could easily be removed. In order to select only the most useful words, all words that did not represent one particular class for a certain minimum proportion of its occurrences were removed from the vocabulary.

The value of this proportion differed, based on the classification task and on the size of the *n*-gram. Since the speaker classification task had only 6 different classes, whereas the topic classification task was divided over 24 different classes, the proportions in the latter task are often much smaller than in the former task. Furthermore, as mentioned before, only a limited number of bigrams and trigrams were included, in order to keep the vocabulary size modest. Therefore, based on various trial runs to find the optimal values, unigrams in the speaker classification task were only included when they appeared in one particular class for at least 50% of all occurrences. For the topic classification task, unigrams were only included when they appeared in one class for at least 40% of the cases. For both tasks, bigrams and trigrams were only included when occurring in one particular class for at least 75% of the cases.

3.3 *tf*idf* Sequence Classifier

The Sequence Classifier is designed to apply the *tf*idf* metric to any type of input and search for relevant sequences, whether the goal is to classify music, languages or text categories. Although several different variations of the *tf*idf* metric exist, the application presented here employs only three variations, called *ntl1*, *ntl2* and *ntl3*. All three variations use the natural term frequency *tf* (denoted by *n*), the logarithmic inverse document frequency *idf* (denoted by *t*) and are normalized for the length of the document (denoted by *l*) (see Manning, Raghavan & Schütze, 2008). The difference lies in the application of varying sized *n*-grams – *ntl1* uses only unigrams, *ntl2* uses only bigrams and *ntl3* uses only trigrams.

Although the *tf*idf* Sequence Classifier is capable of using combinations of unigrams, bigrams and trigrams (as are used in the application of the Naïve Bayes and Committee classifiers), early exploratory runs with the algorithm showed that performances were better when using single features (e.g. only bigrams) than when using combinations of features (e.g. unigrams and bigrams). Therefore, the deliberate choice was made to use different features for the *tf*idf* Sequence Classifier than for the Naïve Bayes and Committee classifiers.

Compared to the other two classifiers, the Sequence Classifier is able to work with much bigger vocabularies without suffering diminishing effects on accuracy, so there was no need to apply any feature selection methods. However,

some form of feature selection is automatically incorporated in the calculation of the inverse document frequency, as the most common terms will receive a value of zero. Recall from section 2.3.3 the formula for calculating *idf*:

$$idf_t = \log \frac{N}{df_t}$$

where N is the total number of documents, and df_t is the number of documents in which term t occurs. For terms that are so common that they appear in every document (e.g. words like "the" or "a"), the *idf* value will be the log of 1 (since N and df_t are equal), which is zero. Therefore, these words will have no influence on the total score of the *tf*idf* metric.

3.4 Synonyms Mergers

In order to find out whether the use of synonyms would have any effects on the accuracy of classification, the goal was to create an alternative version of the dataset which incorporated synonyms. This was done with the use of WordNet, which is a large lexical database that represents words using so-called synsets (e.g. Miller *et al.*, 1990). When presented with a word, WordNet first lemmatizes the word (which means taking the most basic form of the word, e.g. *cars* becomes *car*, or *sank* becomes *sink*), and then finds the stored synsets of that lemma, which represent a set of particular meanings of the word. Each synset consists of a list of synonymous words in regard to that particular meaning, and a word with multiple meanings can appear in multiple synsets. For example, the word *car* has multiple synsets, each with different meanings and each with different synonyms attached to it; one synset interprets the word as a motorized vehicle and includes synonyms such as *auto* and *automobile*, whereas another synset interprets it as part of a train and includes synonyms such as *railcar* and *railway_car* (Bird, Klein & Loper, 2009).

As mentioned before in section 2.3.4, synonyms have been applied to classification tasks in different ways, the most distinct difference being that they can be used to either increase or decrease the size of the dataset. For example, Rodriguez *et al.* (1997) expanded their dataset by adding relevant synonyms to the existing words. The advantage of this approach is that a larger dataset increases the chances of finding rare terms that can be used for classification, which in Information Retrieval is referred to as increasing recall. Contrary to this approach, Scott and Matwin (1999) decreased the size of their dataset by combining words that are synonymous of each other into a single word. The advantage of a smaller dataset is that the (relatively) few words that remain are likely to have higher discriminative value between classes, which in Information Retrieval is referred to as increasing precision.

The difficulty with the approach employed by Rodriguez *et al.* (1997) is that this requires a clear understanding of the meaning of the synonyms, because otherwise words might be added that only share similarities with the target word in a certain specific context. For example, a text referring to a politician as being *right* might mean that the politician is *correct*, or that he/she is *not_left*, depending on the context. Rodriguez *et al.* solved this problem by selecting the appropriate synonyms manually, but this would not be feasible with the large size of the dataset used in this thesis. Scott and Matwin (1999) bypassed this difficulty by

only selecting the synonyms that shared a matching hypernym, and then replacing both synonyms with that hypernym.

Therefore, in this thesis synonyms are used in a similar fashion by combining words with the same meaning, and thus effectively decreasing the size of the dataset in an effort to improve classification precision. For this purpose, a different approach is employed than suggested by Scott and Matwin (1999). Instead of searching for hypernyms, a program was created called the Synonyms Merger, which is used to modify the content of the original dataset. The idea is to take all possible meanings of a single word (in the form of synsets) and determine how many of these meanings overlap with the meanings of any other words in the document collection. When this number exceeds a certain pre-determined threshold, the Synonyms Merger merges these words and their synsets into a single new instance. This pre-determined value defines how many words will be merged, as a low threshold will result in many mergers, but a high threshold will result in only a few mergers. It is important to note that not only the words are merged, but their corresponding synsets as well, thereby making it increasingly probable that a new word has a match with the merged words, and also becomes part of the new instance.

To see how this works, table 3.3 provides an example of three distinct words, and a few of the synsets that WordNet ascribes to these words. The synsets represent different meanings associated with the original words, so the word *good* can be interpreted as *good* or as *commodity*. When the pre-determined threshold is set to one, which means that a word must have at least one synset that matches with the synset of another word before they are merged, the word *good* shares no similar synsets with the word *correct*, and therefore will not be merged with that word. However, *correct* does share a similarity with *right*, and as a result the two words, and their associated synsets, are merged into a single instance. Consequently, the word *good* now does share a synset with this new instance (i.e. the synset *good*), and therefore will be merged with the new instance as well. Especially in large datasets, where many words will share similar synsets and be merged into a single new instance with an increasingly large number of synsets, even more new words will share a synset with this instance and will be merged with it as well.

This effect can be prevented by increasing the threshold value for the required number of matching synsets. In the same example, the threshold can be set to two, which will result in a different combination of mergers. The word *good* now still shares no matching synsets with the word *correct*, and the word *correct* still shares enough matching synsets (i.e. *correct* and *right*) to be merged with the word *right*. However, *good* now only shares one synset with the newly merged

Table 3.3

Examples of WordNet Synsets

Words:	**Good**	**Correct**	**Right**
Synsets:	*Good*	*Correct*	*Right*
	Commodity	*Right*	*Proper*
	Beneficial	*Compensate*	*Correct*
	Well	*Discipline*	*Good*

instance, which is not enough for *good* to be merged with this instance as well. Of course, in a large dataset, it remains likely that another word exists which shares matching synsets with both the words *right* and *good*, causing these words to still merge eventually. But by trying out several increasing values for the required number of matches, the chance of unrelated words being inappropriately merged decreases.

3.4.1 Creating Alternative Datasets

In order to find out what thresholds for mergers are most suitable for classification purposes, the Synonyms Merger is applied to the original dataset and six new datasets are created, each differing in the number of matches required before words are merged. The first value for required number of matches is set to infinite, which means that none of the words are merged. The second threshold is 1, which means that words are merged when one or more of the synsets are similar, and which will likely cause a very large proportion of the words to be merged into a single feature. The threshold for the third dataset is set to 5, after which come 10, 15 and 20, with higher values expected to lead to a decreasing number of mergers.

The mergers are represented in the new datasets by unique numerical values. Looking at the previous example, the words *correct* and *right* might be merged into a new instance represented by the (random) number *11*. Then, when the word *good* is encountered, it will be merged into a new instance which may be represented by the number *22*. Each new word that matches with an existing instance will be merged into a new instance, denoted by a new numerical value (and, in the process, discarding the value for the old instance). Once all the new instances and their numerical values have been stored, the Synonyms Merger goes through the entire dataset again, and substitutes all the original words with the corresponding numerical values. Words that have no synsets (e.g. the words *the* or *since* have no existing synsets) are not replaced by numerical values, but remain in the dataset as they were. Once the new datasets are completed, the three classifiers are applied once more. Note that for accurate classification, the classifiers do not differentiate between numerical values or actual words, only the number of occurrences of unique tokens.

Expectations are that all six new datasets will yield different results. Even the dataset with no mergers (i.e. threshold for matches set to infinity) is still different from the original dataset, due to the fact that WordNet lemmatizes all words before finding the appropriate synsets. This means that the Synonyms Merger will find synsets for words with different forms but equal lemma's. For example, the unigrams *car* and *cars* will both be converted into a numerical value representing the lemma *car*, and the trigram *he_has_been* will be converted into the lemma *he_have_be*. For the datasets with many mergers (i.e. with low thresholds for matches, such as 1 or 5), expectations are that as the majority of words are merged, only a few words with unique meaning (e.g. *ObamaCare*) will remain useful for classification. For datasets with infrequent mergers (i.e. with high thresholds for matches, such as 15 or 20), expectations are that only the strongest synonyms are combined into a new instance. The next chapter describes which of these approaches resulted in more accurate classification predictions.

Chapter 4

Results

Since the goal of this thesis is to classify texts both on political speaker and political topic, the results will be judged separately so that the scores for topic and speaker can be compared. As these two tasks differ in number of labels and class sizes, it is conceivable that different classifiers achieve varying scores on the separate tasks. The results are obtained using the commonly applied method of tenfold cross-validation (as mentioned in section 3.1), which calculates average classification accuracy, with accuracy simply defined as the number of correctly predicted labels (similar to, for example, Joachims, 1997). For evaluation purposes, accuracy scores are often compared to the majority Baseline, which is the accuracy score obtained when the label from the largest class (i.e. the label with the highest prior probability) is assigned to each document (Jurafsky & Martin, 2000).

The first section of this chapter describes the results achieved by the Naïve Bayes and Committee classifiers, followed by the results achieved by the *tf*idf* Sequence Classifier. The third section describes how results varied dependent on the datasets with different synonym representations, and the final section concludes with a comparison of all the results.

4.1 Experiment 1, 2 & 3: Naïve Bayes, Committee and *tf*idf* Results

In the first two experiments, both the Naïve Bayes and Committee classifiers were applied to the data three times; once with unigrams, once with combinations of unigrams and bigrams, and once with combinations of unigrams, bigrams and trigrams as features for classification. In the third experiment, the *tf*idf* Sequence Classifier was applied to the data three times as well; once with unigrams, once with bigrams and once with trigrams (note that this is an intentionally different approach from the first two experiments, as described in section 3.3). Table 4.1 shows the resulting accuracies on the speaker classification task (table 4.1a) and on the topic classification task (table 4.1b).

In the next two sections, the differences in accuracy scores are examined in detail, first for the speaker classification task, and then for the topic classification task. For each task, the overall performances of each classifier are compared, without looking at variations for different *n*-grams. Next, each

Table 4.1 a

Mean accuracy for Speaker Classification task (Std.Dev. in parentheses)

# *n*-grams	Baseline	Naïve Bayes	Committee	*tf*idf (ntl)*
1	24.55 (1.17)	65.28 (1.66)	63.44 (1.81)	61.83 (2.69)
2	24.55 (1.17)	**67.65 (1.50)**	**68.29 (1.81)**	**68.95 (2.21)**
3	24.55 (1.17)	**68.60 (1.49)**	**69.56 (1.62)**	60.90 (2.09)
Average	24.55 (1.17)	67.18 (2.06)	67.10 (3.11)	63.90 (4.30)

Values printed in bold type do not differ significantly from each other

classifier is examined separately to see how the scores differed based on variations in *n*-grams, followed by a comparison of the best results between all classifiers. Finally, in the third section, the scores for the speaker classification task are compared to the scores for the topic classification task.

4.1.1 Speaker Classification Results

When comparing classifiers for the speaker classification task (table 4.1a), the results show that overall (i.e. not looking at differences for *n*-grams), the accuracy scores differ significantly ($F(3, 96) = 528.18$, $p < .001$). Tukey post-hoc comparisons show that all three classifiers score significantly better than the majority baseline ($M = 24.55$, $p < .001$). Furthermore, both the Naïve Bayes and Committee classifiers significantly outperform the *ntl* classifier ($M = 63.90$, $p < .001$), although the difference between the Naïve Bayes ($M = 67.18$) and Committee ($M = 67.10$) classifiers is not significantly different ($p = 1.00$).

When comparing number of features for the speaker classification task, the Naïve Bayes classifier produces significantly different accuracy scores across the different *n*-grams ($F(2, 27) = 12.20$, $p < .001$), where a Tukey post-hoc analysis shows that the values for bigram classifications (M = 67.65) are significantly better than unigram classifications ($M = 65.28$, $p < .01$), but not significantly different from trigram classifications ($M = 68.60$, $p = .37$). The Committee classifier shows a similar significant difference across different *n*-grams ($F(2, 27) = 39.72$, $p < .001$), with a significant difference between bigrams ($M = 68.29$) and unigrams ($M = 63.44$, $p < .001$) but not between bigrams and trigrams ($M = 69.56$, $p = .21$). For the last classifier, the *ntl* metrics' scores are significantly different across number of features as well ($F(2, 27) = 35.35$, $p < .001$), although in this case the post-hoc analysis reveals that only bigrams ($M = 68.96$) significantly outperform unigrams ($M = 61.83$, $p < .001$), whereas unigram scores are not significantly different from trigram scores ($M = 60.90$, $p = .65$).

Finally, when examining each score separately (not including the majority baseline scores), the results show that the accuracy scores for the three classifiers are very close. The highest accuracy scores (ranging from $M = 67.65$ to $M = 69.56$, emphasized in bold in table 4.1a) with no significant differences ($p = .37$) come from each of the three classifiers.

Table 4.1 b

Mean accuracy for Topic Classification task (Std.Dev. in parentheses)

# *n*-grams	**Baseline**	**Naïve Bayes**	**Committee**	***tf*idf (ntl)***
1	12.22 (1.84)	64.61 (2.09)	**70.59 (1.54)**	**68.75 (1.91)**
2	12.22 (1.84)	61.12 (2.14)	**71.40 (1.88)**	**69.41 (1.64)**
3	12.22 (1.84)	56.42 (2.68)	**71.31 (1.93)**	59.58 (1.43)
Average	12.22 (1.84)	60.72 (4.08)	71.10 (1.77)	65.91 (4.84)

Values printed in bold type do not differ significantly from each other

4.1.2 Topic Classification Results

For the topic classification task (table 4.1b), the results show that overall (i.e. not looking at differences for *n*-grams), the accuracy scores again differ significantly ($F(3, 96) = 686.50$, $p < .001$). All three classifiers significantly outperform the majority baseline score. However, contrary to the results for the speaker classification task, a post-hoc analysis shows that each classifier scores significantly different from the other; the Naïve Bayes classifier ($M = 60.72$) is significantly outperformed by the *ntl* classifier ($M = 65.91$, $p < .001$), which in turn is significantly outperformed by the Committee classifier ($M = 71.10$, $p < .001$).

When looking at the differences based on number of features, the Naïve Bayes classifier scores are significantly different across different *n*-grams ($F(2, 27) = 31.47$, $p < .001$), although the effects are opposite to the results found for the speaker classification task. The lowest accuracy scores are obtained using trigrams as features ($M = 56.42$), which are significantly outperformed by the bigram scores ($M = 61.12$, $p < .001$), which in turn are significantly outperformed by the unigram scores ($M = 64.61$, $p < .01$). For the Committee classifier, number of features seems to have little impact on accuracy scores, as there is no significant difference between the three *n*-grams ($F(2, 27) = .61$, $p = .55$). Lastly, when examining the *ntl* metrics, the results show that the scores again differ significantly across the three *n*-grams ($F(2, 27) = 108.28$, $p <$,001). Post-hoc analysis reveals that trigram scores ($M = 59.58$) differ significantly from bigram scores ($M = 69.41$, $p < .001$), whereas bigram scores do not differ significantly from unigram scores ($M = 68.75$, $p = .66$).

Finally, when examining each score separately, the results again show that the classifier performances are very close. However, contrary to the speaker classification task, the highest accuracy scores (ranging from $M = 68.75$ tot $M = 71.40$) which show no significant differences ($p = .08$) are only obtained by the Committee and *tf*idf* classifiers, indicating that these classifiers score significantly better compared to the Naïve Bayes classifier on the topic classification task.

4.1.3 Comparing Speaker and Topic Classification Tasks

The last step in the analysis is to compare results for each classifier based on the task of classifying speaker and topic. When looking at the accuracy scores obtained by the Naïve Bayes classifier, the results show that the scores for predicting speaker labels (M = 67.18) are significantly better than the scores for topic labels (M = 60.72): $t(58) = 7.73$, $p < .001$. The accuracy scores obtained by the Committee classifier show the opposite results, as scores for predicting speaker labels (M = 67.10) are significantly worse than the scores for topic labels (M = 71.10): $t(58) = 6.13$, $p < .001$. Finally, the accuracy scores obtained by the *tf*idf* classifier do not seem to be affected by classification task, as the scores for speaker labels (M = 63.90) and topic labels (M = 65.91) show no significant difference: $t(58) = 1.71$, $p = .09$.

4.2 Experiment 4: Effects of Merging Synonyms

In the final experiment, alternative versions to the original dataset were created by merging synonymous terms. This process yielded six new datasets, each differing in the frequency with which synonyms are merged into a single new term. Consequently, the three classifiers were once again applied to the new datasets, in order to investigate whether merging synonyms would benefit classification accuracy. Table 4.2 shows the accuracy results on the speaker classification task (table 4.2a) and on the topic classification task (table 4.2b) for each new dataset. For each classifier, only the best scoring methods (i.e. based on number of *n*-grams) are shown.

The next two sections describe the differences in accuracy scores, first for the speaker classification task, and then for the topic classification task. For each task, the overall performances of each classifier are compared, without looking at differences in datasets. Next, the differences dependent on datasets are compared for each classifier, to analyze what effects the synonym mergers had on prediction accuracies. Finally, in the third section, the scores for the different tasks are again compared to see whether the classifiers perform better on speaker or on topic classification.

Table 4.2 a

Mean accuracy for Speaker Classification task (Std.Dev. in parentheses)

Dataset	**Method + Accuracy**		
	Naïve Bayes 3	Committee 3	*ntl* 2
Merged0	69.26 (1.66)	69.89 (2.01)	69.71 (2.83)
Merged1	46.38 (2.09)	42.89 (1.36)	41.60 (3.05)
Merged5	50.06 (1.76)	48.95 (1.55)	52.22 (1.81)
Merged10	66.12 (0.93)	65.64 (1.74)	63.82 (2.13)
Merged15	66.82 (1.54)	66.93 (1.43)	67.69 (3.11)
Merged20	67.85 (1.46)	68.35 (1.89)	69.74 (2.88)

4.2.1 Speaker Classification Results

For the speaker classification task (table 4.2a), the results show that overall (i.e. not looking at differences between datasets), the three classifiers show no significant differences between each other ($F(2, 177) = .06$, $p = .94$). However, all three classifiers still significantly outperform the majority baseline ($M = 24.55$, $p < .001$).

When looking at the scores in relation to the different dataset, the Naïve Bayes results show that the results differ significantly ($F(5, 54) = 391.96$, $p < .001$). A post-hoc analysis shows that the lowest scores come from the Merged1 dataset ($M = 41.60$), whereas the highest scores come from the Merged20 ($M = 67.85$) and Merged0 ($M = 69.26$) datasets, with no significant difference between these two ($p = .38$). The Committee classifier shows the same pattern: results differ significantly dependent on the dataset ($F(5, 54) = 468.17$, $p < .001$), with the lowest scores again coming from the Merged1 dataset ($M = 42.89$), and with no significant difference ($p = .33$) between the highest scores of the Merged20 ($M = 68.35$) and Merged0 ($M = 69.89$) datasets. Finally, the *ntl2* results show that these scores too differ significantly ($F(5, 54) = 207.61$, $p < .001$), with the lowest score belonging to the Merged1 dataset ($M = 41.60$). However, contrary to the other two classifiers, the highest accuracy scores for the *ntl2* metric are obtained on three datasets: the Merged0 ($M = 69.71$), Merged15 ($M = 67.69$) and Merged20 ($M = 69.74$) datasets show no significant differences between them ($p = .46$).

4.2.2 Topic Classification Results

When analyzing the topic classification task (table 4.2b), the results show that overall (i.e. not looking at differences between datasets), the three classifiers again show no significant differences between each other ($F(2, 177) = 1.07$, $p = .34$). Compared to the majority baseline though, the results of all three classifiers significantly outperform the majority scores ($M = 12.22$, $p < .001$).

Examining the scores in relation to the different datasets reveals a significant difference for each dataset for the Naïve Bayes classifier ($F(5, 54) = 607.36$, $p < .001$). A post-hoc analysis shows that each score significantly differs from the rest, with the lowest scores belonging to the Merged1 dataset ($M = 32.32$), and the highest score belonging to the Merged0 dataset ($M = 65.31$, $p <$

Table 4.2 b

Mean accuracy for Topic Classification task (Std.Dev. in parentheses)

Dataset	**Method + Accuracy**		
	Naïve Bayes 1	Committee 3	*ntl* 2
Merged0	65.31 (1.19)	70.28 (2.08)	69.94 (2.00)
Merged1	32.32 (1.43)	28.07 (1.14)	34.48 (1.46)
Merged5	50.37 (1.92)	49.71 (1.72)	50.75 (1.54)
Merged10	58.38 (1.83)	59.15 (2.22)	59.93 (1.78)
Merged15	60.40 (1.72)	63.60 (2.08)	67.10 (1.21)
Merged20	62.89 (1.14)	66.78 (1.87)	67.94 (1.14)

.05). Results for the Committee algorithm show a similar pattern: the datasets differ significantly ($F(5, 54) = 679.20$, $p < .001$), where the lowest scores again come from the Merged1 dataset ($M = 28.07$) and the highest scores come from the Merged0 dataset ($M = 70.28$, $p < .01$). Finally, results obtained by the *ntl2* metric are also significantly different ($F(5, 54) = 775.65$, $p < .001$), with the lowest scores coming from the Merged1 dataset ($M = 34.47$). However, contrary to the other two classifiers, the highest accuracy scores for the *ntl2* metric come from both the Merged0 ($M = 67.95$) and Merged20 ($M = 69.95$) datasets, with no significant difference between these two ($p < .01$).

4.2.3 Comparing Speaker and Topic Classification Tasks

For the final part of the analysis, the accuracy scores for each classifier are compared in relation to the speaker and topic classification tasks. When looking at the Naïve Bayes classifier, the overall results (i.e. not looking at differences between datasets) show that the scores for predicting speaker label ($M = 61.08$) are significantly better compared to the scores for predicting topic labels ($M = 54.94$): $t(118) = 3.22$, $p < .01$. This indicates that, similar to the results on the original dataset, the Naïve Bayes classifier is better suited for predicting speaker labels than topic labels. When looking at the Committee classifier, the results show that the differences between accuracy scores for speaker ($M = 60.44$) and topic ($M = 56.27$) classification tasks are not significant: $t(118) = 1.80$, $p = .07$. The accuracy scores obtained by the *ntl2* metric show a similar pattern, as here too the difference between speaker ($M = 60.80$) and topic ($M = 58.36$) classification tasks are not significant: $t(118) = 1.13$, $p = .26$.

Chapter 5

Conclusion & Discussion

This final chapter provides an overview of the most important results, followed by a discussion on whether these results were as expected, and when and why expectations differed from the observations.

5.1 Conclusion

The goal of this thesis was to investigate to what extent a word-based classification approach could successfully predict to what label of political speaker and political topic a text belongs. For this purpose, four different experiments were executed, each one designed to answer one of the four research questions. The first research question asked to what extent the Naïve Bayes classifier could successfully predict political speaker and political topic labels. Based on the results of the first experiment it can be concluded that Naïve Bayes is better capable of classifying speaker labels than topic labels. Furthermore, accuracy results can be improved by providing the classifier with extra features in the form of bigrams and trigrams when predicting speaker labels, but not when predicting topic labels. Compared to the other classifiers, Naïve Bayes produces less accurate results when predicting topic labels, and is the only classifier which achieves the highest accuracy scores by using unigrams. When predicting speaker labels, Naïve Bayes performs best when using a combination of unigrams, bigrams and trigrams, in which case it does not differ significantly from the best scores achieved by the other two classifiers.

The second research question asked to what extent the Committee classifier could successfully predict political speaker and political topic labels. The results from the second experiment show that the Committee algorithm performs best when predicting topic labels, in which case the results rank among the highest compared to the other classifiers, independent of the number of features used. When predicting topic labels, the Committee classifier is best served with combinations of multiple features (i.e. using only unigrams produces the least accurate scores), in which case it again ranks amongst the highest performing classifiers.

The third research question aimed to investigate to what extent the *tf*idf* Sequence Classifier could successfully predict political speaker and political topic labels. Contrary to the other two classifiers, the Sequence Classifier seemed to perform equally well on both tasks, with no significant differences in accuracy

results between speaker and topic classification tasks. When predicting speaker labels, the best results were obtained with the *ntl2* metric (i.e. using bigrams), which were equivalent to the best results of the other classifiers. When predicting topic labels, both the *ntl1* and *ntl2* metrics delivered accuracy scores that were as good as the Committee classifier, and significantly better than the Naïve Bayes classifier.

Finally, the fourth research question asked to what extent prediction accuracies could be enhanced by merging synonyms. The results show a distinct pattern over all three classifiers, as more mergers lead to lower accuracy scores. The datasets with the least mergers (i.e. Merged20, or for the *ntl2* metric even merged15) significantly outperform datasets with many mergers. This effect holds for both speaker and topic classification tasks. Although the scores for the database with the most mergers are still better than the majority baseline, the results show that the best scores come from the database with zero mergers, which are matched in only a few cases by the Merged20 dataset. Therefore it can be concluded that merging synonymous terms is not a useful approach for increasing classification accuracies.

5.2 Discussion

One of the first conclusions that can be drawn from the experiment results is that the merger of synonyms has no effect on prediction accuracies, and even has a negative effect when synonym mergers occur frequently. The intention was to find out whether many mergers could result in a dataset where only a few words would not merge because of their distinctness (e.g. words like *ObamaCare*, which has no existing synonyms in WordNet), and to see whether these distinct words could be useful for discriminating between classes. However, the results clearly showed that this was not the case, as an increasing number of mergers would consistently decrease the accuracy of class predictions. Apparently, on this particular dataset, the classifiers work better with many distinct words than with a few highly discriminative words.

As the frequency of mergers decreased, the results significantly improved, but never to a level above the original results that was worth the extra effort. Expectations were that merging only words with strong synonymy would improve results in a similar fashion as discussed by Scott and Matwin (1999), but the results showed that the dataset with the fewest number of mergers only scored marginally better for speaker classification tasks compared to the other approaches, and this difference was not significant. Perhaps the better approach would have been to add new synonyms, instead of merging existing ones, although this would require a better understanding of the context of the words, which currently remains a very difficult task to achieve using automated analysis methods.

One striking observation that can be made by inspecting the results from the Merged0 dataset (i.e. an adapted dataset but without the merger of synonyms) is that these results are rather close to the results achieved on the original dataset, both for speaker and topic classification tasks. As mentioned before, the dataset with no mergers only differs from the original dataset in that it applies a process of lemmatization to the texts, as part of the process when working with WordNet. For the application of the Naïve Bayes, Committee and *tf*idf* classifiers, the choice was made to not employ lemmatization, since expectations were that

differences in lemma's are not useful for categorizing political texts. As suggested by Yu, Kaufmann and Diermeier (2008), political texts are often distinguishable based on their content rather than on the form of the text, and lemmatization typically affects form more than content. For example, characteristic political phrases such as *pro-life* or *death taxes* are often used in standard formulations, and hence lemmatization would have little impact on these words. Therefore, the fact that the results obtained with the original dataset and the lemmatized dataset (i.e. the Merged0 set) show strong similarities seems to confirm the expectation that the use of lemmas provides no noticeable advantages when classifying political texts.

When looking at the results obtained by the different classifiers on the original dataset, one noticeable conclusion is that the lowest performance scores come from the Naïve Bayes classifier, especially when dealing with many different labels, as in the topic classification task. Where the Committee classifier benefits from adding bigrams and trigrams to the vocabulary, the reverse appears to be true for the Naïve Bayes approach. It seems to be the case that the Naïve Bayes approach is negatively affected by increasing vocabulary sizes when dealing with many labels. A similar observation was made during the process of finding an optimal threshold for feature selection (see section 3.2.1), where it was found that decreasing feature selection thresholds (i.e. allowing more features to be added to the vocabulary) also decreased the Naïve Bayes performance.

This effect might be explained by looking at the different approaches that the Naïve Bayes and Committee classifiers use for estimating the probability of a class label. When new words are added to the vocabulary, the Committee algorithm scores remain constant over the other words in the vocabulary (i.e. the 'votes' of each word are only dependent on their own number of occurrences). Conversely, in the case of Naïve Bayes, individual word scores are dependent on the size of the vocabulary, which are therefore affected when the vocabulary size increases. However, it remains a question whether this is the actual cause for the decrease in classification performance, and it seems that more attention to the differing scores is required in order to confirm what causes these effects.

A potential approach to further investigate the differences in prediction performances between the classifiers, and therefore a recommendation for future research, is to classify the documents according to a hierarchy in the labels, similar to the research by McCallum, Rosenfeld, Mitchell and Ng (1998). The experiments in this thesis are deliberately focused on a dataset with more distinct classes (in the form of political speaker) than is common in political classification tasks (often in the form of political ideology, e.g. Democrat versus Republican). Classifying more labels can be a more difficult task as statistical approaches to automated classification typically rely on large amounts of hand-labeled data, which in this case are made smaller by creating a division over more classes. Therefore, a possible approach for improving the results could be to classify the data in hierarchical steps. By first focusing on the relatively easier task of classification based on political ideology, and only in a second stage focusing on the classification of political speaker, the results might be improved. By comparing performances on varying numbers of class labels, differences in the results might give new insights into differences between the classifiers.

The final conclusion that can be drawn from the results which is worth mentioning, is that the Committee classifier and *tf***idf* Sequence Classifier both achieve approximately similar results on speaker and topic classification tasks,

with an average accuracy of around 70 percent. Consequently, it can be concluded that these are the two best approaches, as they are more accurate and consistent than Naïve Bayes, and less labor-intensive to apply on the original dataset compared to the synonyms-converted dataset. Therefore, the Committee and *tf***idf* classifiers are suitable solutions for dealing with the specific type of data that was used in this thesis.

This conclusion has important implications for the future perspective of applying automated text classification to the field of company conference call analysis. As mentioned before, the dataset used here is different from other datasets commonly applied when classifying political speaker and topic labels, which are often of a more formal nature. For example, Gentzkow and Shapiro (2010) use Congressional Record transcripts, and Laver, Benoit and Garry (2003) use party manifestos, which are texts filled with words and phrases used by the politicians themselves. For the purpose of classifying conference call transcripts in reference of political activities, though, it will likely be more useful to use a wider variety of text formats, such as news articles and interviews. The fact that the Committee and *tf***idf* classifiers obtain such consistent accuracy results over such a variety of source materials as applied here, implies that these approaches would be very suitable for applications in conference call analysis.

References

Balakrishnan, R., Qiu, X. Y., & Srinivasan, P. (2010). On the predictive ability of narrative disclosures in annual reports. *European Journal of Operational Research*, *202*(3), 789-801.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python*. O'reilly.

Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 1, p. 740). New York: Springer.

Boiy, E., & Moens, M. F. (2009). A machine learning approach to sentiment analysis in multilingual Web texts. *Information retrieval*, *12*(5), 526-558.

Bowen, R. M., Davis, A. K., & Matsumoto, D. A. (2002). Do conference calls affect analysts' forecasts? *The Accounting Review*, *77*(2), 285-316.

De Bruijn, B., & Martin, J. (2002). Getting to the (c) ore of knowledge: mining biomedical literature. *International journal of medical informatics*, *67*(1), 7-18.

Gentzkow, M., & Shapiro, J. M. (2010). What drives media slant? Evidence from US daily newspapers. *Econometrica*, *78*(1), 35-71.

Grimmer, J., & Stewart, B. M. (2013). Text as data: the promise and pitfalls of automatic content analysis methods for political texts. *Political Analysis*.

Hollander, S., Pronk, M., & Roelofsen, E. (2010). Does silence speak? An empirical analysis of disclosure choices during conference calls. *Journal of Accounting Research*, *48*(3), 531-563.

Joachims, T. (1997). A probabilistic analysis of the rocchio algorithm with TFIDF for text categorization. In *International Conference on Machine Learning (ICML)*.

Jurafsky, D. & Martin, J. H. (2000). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition* (Second edition). Upper Saddle River: Prentice Hall.

Larcker, D. F., & Zakolyukina, A. A. (2012). Detecting deceptive discussions in conference calls. *Journal of Accounting Research*, *50*(2), 495-540.

Laver, M., Benoit, K., & Garry, J. (2003). Extracting policy positions from political texts using words as data. *American Political Science Review*, *97*(02), 311-331.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval* (Vol. 1). Cambridge: Cambridge University Press.

Maynard, D., & Funk, A. (2012). Automatic detection of political opinions in tweets. In *The Semantic Web: ESWC 2011 Workshops* (pp. 88-99). Springer Berlin Heidelberg.

McCallum, A. & Nigam, K. (1998). A comparison of event models for naive Bayes text classification. *AAAI-98 workshop on learning for text categorization*, *Vol. 752*, pp. 41-48.

McCallum, A., Rosenfeld, R., Mitchell, T. M., & Ng, A. Y. (1998). Improving text classification by shrinkage in a hierarchy of classes. In *Proceedings of the Fifteenth International Conference on Machine Learning* (pp. 359-367). Morgan Kaufmann Publishers Inc.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (1990). Introduction to WordNet: An on-line lexical database*. *International journal of lexicography*, *3*(4), 235-244.

Pang, B., Lee, L., & Vaithyanathan, S. (2002, July). Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10* (pp. 79-86). Association for Computational Linguistics.

Purpura, S., & Hillard, D. (2006). Automated classification of congressional legislation. In *Proceedings of the 2006 international conference on Digital government research* (pp. 219-225). Digital Government Society of North America.

Rodríguez, M. D. B., Gomez-Hidalgo, J. M., & Diaz-Agudo, B. (1997). Using WordNet to complement training information in text categorization. *arXiv preprint cmp-lg/9709007*.

Scott, S., & Matwin, S. (1999). Feature engineering for text classification. In *ICML: Proceedings of the Sixteenth International Conference on Machine Learning* (pp. 379-388). Morgan Kaufmann Publishers Inc.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, *34*(1), 1-47.

Van Zaanen, M., & Gaustad, T. (2010). Grammatical inference as class discrimination. In *Grammatical Inference: Theoretical Results and Applications* (pp. 245-257). Springer Berlin Heidelberg.

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining: practical machine learning tools and techniques*. Burlington, Massachusetts: Elsevier.

Yu, B., Kaufmann, S., & Diermeier, D. (2008). Classifying party affiliation from political speech. *Journal of Information Technology & Politics, 5*(1), 33-48.

Zheng, Z. (1998). *Naive Bayesian classifier committees* (pp. 196-207). Springer Berlin Heidelberg.

Appendix 1

Text Fragment Examples

Al Gore on Abortion:

* Opposes partial birth abortion but opposes banning it Gores approach to abortion is just as pragmatic as Bushs As a Congressman for a conservative Tennessee district he cast more votes against abortion than in favor But as he became a national figure he changed his position and now claims that he will do everything in his power to prevent Roe v Wade from being overturned Gore opposes parental notification laws Opposes partial birth abortion but also opposes Republican attempts to ban it supports Medicaid funding of abortion

Barack Obama on Drugs:

* We share cross border drug problem with Mexico Q You have supported Mexico s policy against drug trafficking After 65 000 deaths is it time to consider a change the strategy A What I will be saying to the new President of Mexico when he takes office is that we want to continue cooperation and we recognize this is a threat on both sides of the border We obviously generate a lot of demand for drugs in this country and guns and cash flow south at the same time as drugs flow north Q How many more people have to die A Well what we need to do is to weaken the grip of these drug cartels and there are a couple of things we can do The US can focus on drug treatment and prevention and helping people deal with addiction making sure that young people are not getting hooked on drugs If we can reduce demand that means less cash flowing into these drug cartels The other thing that we try to do is to work much more aggressively in preventing the flow of guns and cash down into Mexico And so interdiction has to work both ways

George W. Bush on Environment:

* Proposed different measures to improve the environment Q What specifically has your administration done to improve the condition of our nations air and water supply A Off road diesel engines we a reached an agreement to reduce pollution from off road diesel engines by 90 Ive got a plan to increase the wetlands by 3 million Weve got an aggressive brownfield program to refurbish inner city sore spots to useful pieces of property I proposed to the Congress

a Clear Skies Initiative to reduce sulfur dioxide nitrogen oxide and mercury by 70 I fought for a strong title in the farm bill for the conservation reserve program to set aside millions of acres of land for to help improve wildlife and the habitat We proposed and passed a healthy forest bill to protect old stands of trees and at the same time make sure our forests arent vulnerable to the forest fires that have destroyed acres after acres in the West I proposed a hydrogen generated automobile Were spending a billion dollars to come up with the technologies to do that

John Kerry on Free Trade:

* All new trade must include labor and environmental standards Q Should the US seek more free or liberalized trade agreements A I support free trade but I dont support what the Bush administration calls free trade I will order an immediate 120 day review of all trade agreements to ensure that our trading partners are living up to their labor and environment obligations and that trade agreements are enforceable and are balanced for Americas workers I wont sign any new trade agreements unless they contain strong labor and environmental standards

John McCain on Social Security:

* Future retirees cannot get what current ones get I think its very important that we reform our entitlement programs We are not going to be able to provide the same benefit for present day workers that present day retirees have today Were going to have to sit down across the table Republican and Democrat as we did in 1983 between Ronald Reagan and Tip ONeill I know how to do that I have a clear record of reaching across the aisle whether it be Joe Lieberman or Russ Feingold or Ted Kennedy or others Thats my clear record

Mitt Romney on Technology:

* China counterfeits our electronics and steals our technology ROMNEY On day one I will label China a currency manipulator OBAMA As far as currency manipulation China s currency has actually gone up 11 percent since I ve been president because we have pushed them hard And we ve put unprecedented trade pressure on China That s why exports have significantly increased under my presidency That s going to help to create jobs here Q Apple iPhones are all manufactured in China How do you convince Apple to bring that manufacturing back here ROMNEY The answer is very straightforward We can compete with anyone in the world as long as the playing field is level China s been cheating over the years One by holding down the value of their currency Number two by stealing our intellectual property our designs our patents our technology There s even an Apple store in China that s a counterfeit Apple store selling counterfeit goods They hack into our computers We will have to have people play on a fair basis