PREDICTING THE VOTE USING LEGISLATIVE SPEECH

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

Aditya Budhwar

March 2018

COMMITTEE MEMBERSHIP

TITLE:	Predicting the Vote Using Legislative Speech
AUTHOR:	Aditya Budhwar
DATE SUBMITTED:	March 2018
COMMITTEE CHAIR:	Foaad Khosmood, Ph.D. Assistant Professor of Computer Science
COMMITTEE MEMBER:	Lubomir Stanchev, Ph.D. Assistant Professor of Computer Science
COMMITTEE MEMBER:	Franz Kurfess, Ph.D. Professor of Computer Science

ABSTRACT

Predicting the Vote Using Legislative Speech

Aditya Budhwar

As most dedicated observers of voting bodies like the U.S. Supreme Court can attest, it is possible to guess vote outcomes based on statements made during deliberations or questioning by the voting members. In most forms of representative democracy, citizens can actively petition or lobby their representatives, and that often means understanding their intentions to vote for or against an issue of interest. In some U.S. state legislators, professional lobby groups and dedicated press members are highly informed and engaged, but the process is basically closed to ordinary citizens because they do not have enough background and familiarity with the issue, the legislator or the entire process. Our working hypothesis is that verbal utterances made during the legislative process by elected representatives can indicate their intent on a future vote, and therefore can be used to automatically predict said vote to a significant degree.

In this research, we examine thousands of hours of legislative deliberations from the California state legislature's 2015-2016 session to form models of voting behavior for each legislator and use them to train classifiers and predict the votes that occur subsequently. We can achieve legislator vote prediction accuracies as high as 83%. For bill vote prediction, our model can achieve 76% accuracy with an F1 score of 0.83 for balanced bill training data.

ACKNOWLEDGMENTS

Thanks to:

- My advisor, Foaad Khosmood for his invaluable guidance and support.
- My committee members, Franz Kurfess and Lubomir Stanchev, for their indispensable advice and feedback.
- Institute for Advanced Technology and Public Policy for their generous support and for providing the data necessary to conduct this work.
- My family and friends, for their constant love and support.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Understanding the process by which a legislator comes to make a decision can be complex, mysterious and inaccessible to ordinary citizens. There is a clear and unambiguous output to the process: The vote. But the nature of the input and the decision making function itself are difficult to understand fully. Still many expert observers of voting bodies, working for news media, think tanks or lobbyists already engage in fairly accurate vote prediction based on behavioral analysis of the voting members. Such individuals base their predictions in part on what is being said during the deliberations, but they also rely on knowledge from previous votes by the same voter and subject, as well as other outside knowledge that will be difficult to quantify.

1.1 Vote prediction?

In order to predict bill outcomes, we use the utterances made by each legislative member in the hearing. We try to explore the correlation between the utterance and the vote given by the member. We use various features extracted from the speech text and various machine learning algorithms to test our accuracy. Thus this prediction algorithm uses legislative speech data to predict the overall bill outcome which is a novel approach.

1.2 Why do vote prediction?

The main reason we are doing vote prediction is to get an insight into the legislature by proving that there is a correlation between what legislators speak and what they vote. We also want to improve citizen engagement in the legislature activities.

1.3 Contribution

In the majority of the cases where vote prediction is done, fully transcribed speeches are not available. We are thankful to Digital Democracy for giving us access to speech data. By experimenting on this dataset, we are able to predict vote outcome with an accuracy of more than 83%.

1.4 Approach

Our working hypothesis is that statements made and questions asked by lawmakers during legislative proceedings can be indicative of their intent for a future vote on the issue at hand. The statements are only one dimension of the input, but the question is can they alone be predictive to a significant degree?

We test this hypothesis by using predictive analytics on records of legislative proceedings. Specifically, we run supervised machine learning experiments using models trained with lawmaker statements and voting outcomes. For this, we use a data set obtained from the Digital Democracy project containing full transcriptions of legislative proceedings in the California state legislature 2015-2016 session.

Digital Democracy is a publicly accessible platform created and maintained by the Institute for Advanced Technology and Public Policy in order to provide government transparency in US state legislatures. This organization creates the only available searchable archive of all statements made in California state legislative hearings. This platform enables users to search, watch, and share statements made by state lawmakers, lobbyists and advocates as they debate, craft, and vote on policy proposals. As shown in Figure 1.1, when a user queries a desired committee hearing and selects the author and bill, the website displays its video recording, transcript as well as additional data regarding the hearing such as the legislative bills on its agenda.

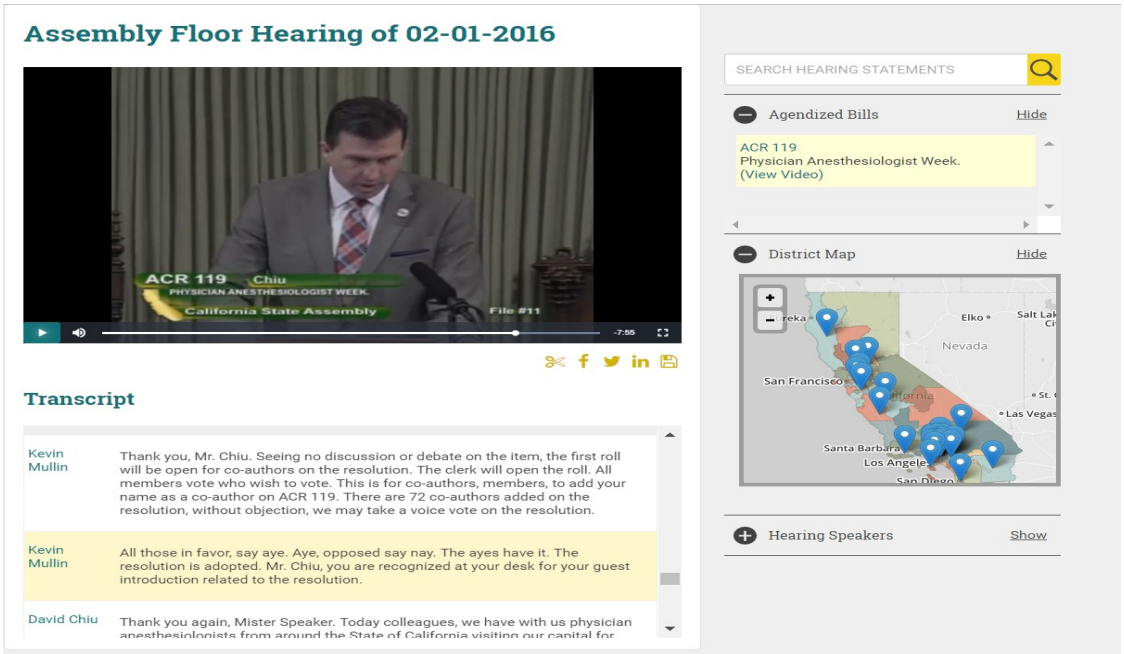


Figure 1.1: Screenshot of a committee hearing page on the Digital Democracy website. It shows the bill discussed on the right and utterance said by the member in video at the bottom, which is highlighted in yellow.

Predictive analytics includes statistical models and other empirical methods that are aimed at creating empirical predictions, as well as methods for assessing the quality of those predictions in practice. Hence, they are a necessary component of scientific research [31]. The target users of this paper are the people who are interested in finding the chances of a bill getting passed by providing the transcripts of the speeches made by committee members present during the hearing. In this experiment, we train models based on the utterances spoken by members of the legislative assembly during previous hearings of the bills tabled. Therefore, the scope of this paper is limited to transcripts from the digital democracy database.

1.5 Thesis Structure

The related work section which follows next will cover the work which has been done in this field previously. This is followed by the background section which describes

the theory behind the technologies used in this paper. In the experimental design section, we cover the detailed system design of the process. This section is followed by results and conclusion section.

Chapter 2

RELATED WORK

There have been similar types of prediction analyses done earlier for vote outcomes, but our literature review indicated that no other research project had quite the same approach as this.

Many researchers who work on legislative data for prediction modeling use roll call data and historical records of legislator's votes on a set of issues. Roll call data can reveal valuable information about the members of a government; for example, we can analyze roll call data from the United States Congress or the British Parliament to uncover the political leanings of their members [12]. Roll call data are essential for understanding legislature because it represents documented proof of the actions by its members. To analyze the bill outcome, scientists in the past have explored various fields apart from roll call such as bill texts, press releases, public plans, and speeches [22]. Much of this information is readily available on the Internet from sources like GovTrack [25]. In previous work, this data has been mined to find underlying structure like partisan affiliation, evidence of polarization, and even predict future voting outcomes based on bill text [12, 24]. These approaches typically involve complex models, such as ideal point modeling that explicitly map legislators to a point along a political line. These models have the benefit that they make analyzing polarization and party affiliation very easy, but they can be difficult to implement and suffer from theoretical deficiencies [9, 7].

2.1 Politically-oriented text

Legislators vote on more than ideology. Issue specific attributes are an important determinant of legislative voting patterns [13]. The work done in analyzing the political behavior while voting revolved around finding the correct features that were responsible for vote pattern. The features that describe legislative vote can be intractable. Due to this, such solutions were not very successful [13].

There have been two works which stand out as closely related to feature selection work where a definite list of features has been given. One of these is fLDA, which models binary or continuous ratings with user affinity to topics [6]. Another is [28], which describes a similar application that combines topic models and matrix completion. A topic model is a type of statistical model for discovering abstract "topics" that occur in a collection of documents. Researchers work also draws on "ideal point models", which are models that transition over time. An application of this can be applied to the votes of legislators on a particular type of legislation. A majority of vote prediction models previously used bag-of-words approach on the bill text, which has problems with generalization.

These papers have provided us with valuable information regarding the political behavior of legislators and feature selection strategy. However, since our aim is different, we will not be using the information provided in these papers.

2.2 Vote Prediction

Research done in the field of vote prediction previously involved roll call data. Roll call data are essential for understanding legislators because it represents atomic and concrete actions of its members. But this data is only one part of a richer record which includes bill texts, speeches, press releases, public plans, and other items [22]. To

understand the political leanings of legislators, one needs to also understand historical records of legislators votes on a set of issues. These are important for vote outcome prediction.

Topic models have been applied to Senate speeches to discern "the substantive structure of the rhetorical legislative agenda" [19]. They have also been used with legislative speeches to gauge legislator sentiments toward legislation using roll-calls [14]. Modeling sentiment in text is generally discussed in the field of sentiment analysis [3]. The ideal point topic model relates closely to user recommendation models based on matrix factorization [21]. Matrix factorization methods for recommendation are akin to large scale spatial behavior models.

The research paper [22] used 12 years of legislative data in their experiments. Their dataset covered 4447 bills, 1269 unique legislators, and 1837033 yea/noe votes. They achieved an accuracy of 82% on limited topic legislative documents.

2.3 Prediction based on speech transcripts

In some applications, speech text was used to determine support or opposition from legislative floor debates. The focus of this research is to use sources of information regarding relationships between discourse segments such as the opinion expressed by two legislators [14]. These models were a substantial improvement over classifying speeches in isolation but were very limited in their scope. They achieved an accuracy of 70% over a test set containing 58 debates. Majority of research we came across had problems with finding enough examples of noe votes/failed bills. This is the case even in prediction based on speech transcripts. If insufficient training data is available, the model will not be as accurate as desired.

In the research done by [19], speech transcripts are used by researchers to infer the topic of the speech through word choices. This analysis can find frequently discussed

topics. We can use this research to find the correlation between the speeches given by various members of legislature. A typical year of any legislative record can include tens of thousands of speeches, and tens of millions of words. Due to the sheer size of this data set, we are currently unable to work with it. However, we will consider pursuing it in the future.

Chapter 3

BACKGROUND

In this research, we examined thousands of hours of legislative deliberations from the California state legislature's 2015-2016 session to form models of voting behavior for each legislator and use them to train classifiers as well as predict the votes that occur subsequently. This vote prediction is used to calculate the overall outcome of the bills. All the technologies explored in this thesis are described below.

3.1 Digital Democracy

When the California state legislature is in session, bill discussions that take place in public hearings are recorded and made available through services like The California Channel. While freely distributing these videos provides access to citizens and organizations about the positions and votes of their state representatives, finding information specific to a bill or legislative topic is usually an untenable task. The reason why these records cannot be searched efficiently is because the legislature does not provide transcripts of these discussions, requiring constituents to scan potentially hours of video to find topics of interest. In 2012, former State Senator Sam Blakeslee founded the Institute for Advanced Technology and Public Policy (IATPP), a non-profit organization housed at California Polytechnic State University (Cal Poly) in San Luis Obispo. Three years later, through private donations and student development, the IATPP launched Digital Democracy, a web service for increasing government transparency and accountability. In addition to providing searchable transcripts of bill discussions, this project also focuses on how this new data set can be meaningfully interpreted and acted upon. As mentioned in Section 1.2, this thesis is motivated,

in part, by the data provided by Digital Democracy and the role that automated language processing methods can serve in promoting government transparency.

3.2 Sentiment Analysis

We use sentiment analysis as a quantifiable feature of the legislative speech data. Sentiment analysis is 'the task of identifying positive and negative opinions, emotions, and evaluations' [27]. Since its outset, sentiment analysis has been subject of an intensive research effort and has been successfully applied to various areas. Some ecamples include assisting users in their development by providing them with interesting and supportive content [26], predicting the outcome of an election [1], movie sales [8] and product review sentiments. The range of sentiment analysis techniques varies from identifying polarity (positive or negative) to a complex computational treatment of subjectivity, opinion and sentiment [3]. In particular, the research on sentiment polarity analysis has resulted in a number of mature and publicly available tools (paid as well as free) such as SentiStrength [17], Alchemy, LingPipe, ElasticSearch sentiment analyzer, Lexalytics, Recursive Neural Tensor Network [20], DatumBox, text-processing, GATE and NLTK [30, 23].

The experiment to find the best tool for sentiment analysis is divided into various steps. The unstructured textual data of transcripts provided by legislature plays a vital role. Due to the sheer amount of text utterances, it is quite cumbersome to process each utterance for manual verification.

Figure 3.1 displays the overall architecture of this experiment. Before tabulating the accuracy of each sentiment analysis tool, we first extract utterances for each hearing. We used SQL language to query the database as well as Pandas library from python to do chunking and process the data in tabular manner,as shown in Figure 3.2. Pandas library provides high-performance, easy-to-use data structures and data

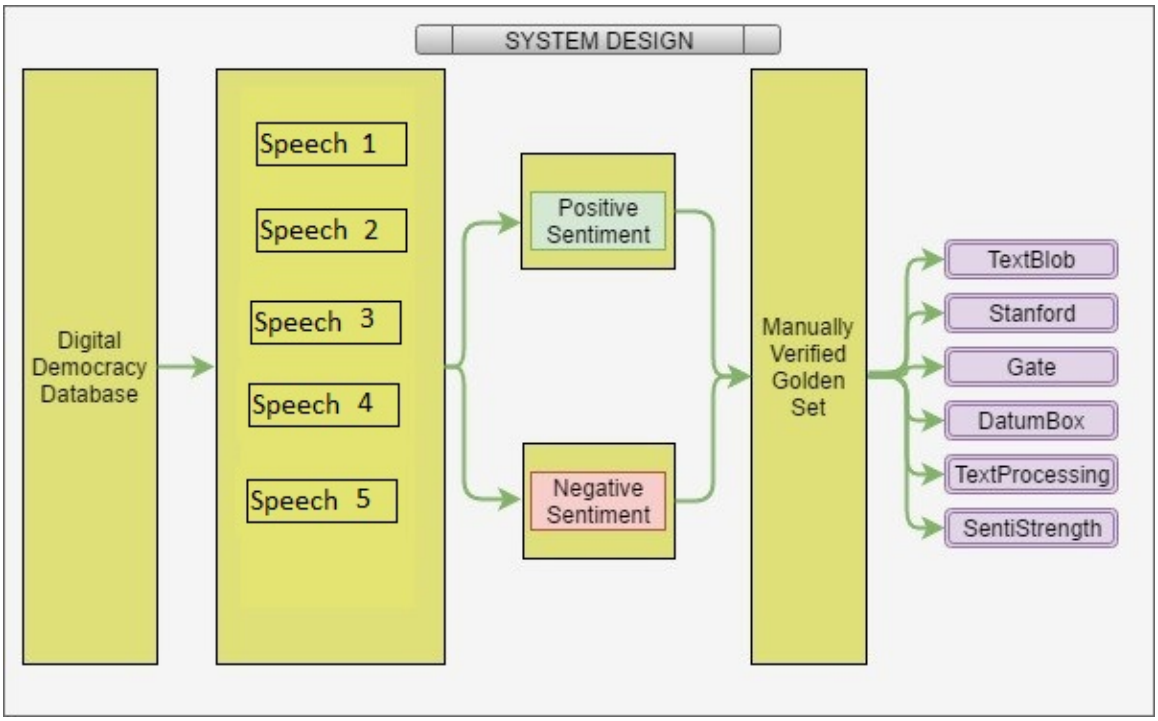


Figure 3.1: System architecture for sentiment analysis tool evaluation

analysis tools for the Python programming language. Since the number of utterances for a single database runs into more than 100 million, Pandas provide an appropriate data structure to work with. Post the data extraction, we join the utterances spoken by a member until any interrupt by another member occurs. We will refer to this collection of utterances as speech in the rest of the paper. The sentiment of each speech is tabulated chunk by chunk and classified based on their polarity and subjectivity. Speech with a high negative score and a NOE vote are considered negative and speech with high positive score and an AYE vote are considered positive. In the third stage, we created a golden benchmark Figure 3.3 by reviewing roughly 500 speeches and then running all the competing tools to find the best approach.

The SQL query used for querying the database for sentiment analysis is 'select * from Utterance JOIN (select BillDiscussion.bid, Video.vid from BillDiscussion, Video where Video.vid >= BillDiscussion.startVideo and Video.vid <= BillDiscus-

UID	VID	PID	TEXT	BID	LAST	MIDDLE	FIRST	VOTEID	CID	VOTEDATE	AYES	NEYS	ABSTAIN	OVERALLBILL	INDIVIDUALBILL	SENTIPOL	SENTISUB	KEYWORDS
1603644	1523	25	And strengthening this relationship through the establishment of a sister state designation will foster greater opportunities in trade, education, and cultural exchange. The legislature has developed a strong relationship with lawmakers and other leaders in Santa Fe through six different reciprocal legislative visits over the past decade, with a seventh visit planned for this fall.	CA_20152016OSCR6	Monning	nan	Bill	34689	452	2/9/2015 15:01	33	0	4	(PASS)	AYE	0.108333333	0.408333333	['sister state designation', 'cultural exchange', 'strong relationship', 'santa fe', 'different reciprocal', 'legislative visits', 'past decade']

Figure 3.2: Data extracted from the Digital Democracy database

sion.endVideo) as temp ON temp.vid = Utterance.vid JOIN Video ON Utterance.vid = Video.vid JOIN Person ON Utterance.pid = Person.pid JOIN BillVoteSummary ON BillVoteSummary.bid = temp.bid JOIN BillVoteDetail ON BillVoteDetail.pid = Utterance.pid WHERE BillVoteDetail.voteId = BillVoteSummary.voteId'

Utterance	Golden Set
And, there is nothing perfect. But when you're watching somebody die and, you know, they're dying they're dying. This isn't a mis-diagnosis.	NEGATIVE
Very good. I agree with your statement and the problem; I just don't agree with the solution, which is why I'm recommending a no vote. Again, as stated by the ACLU, we already have laws on the books, which allow up to four years if someone is charged under elder abuse statute.	NEGATIVE
And we should not have a lot of very good people opting out of running because they don't want to spend all their time on the telephone raising money.	NEGATIVE

Figure 3.3: Golden set

In the final stage to choose what sentiment analysis tool to use, a quality and a performance check is performed on all the available tools Figure 3.4. Though we explored both open-source and licensed tools, for our research we just focused on open-source tools. Since all the tools available in market are done specifically for social media data and our data is speech transcript, data which is more grammatical and without emoticons. The tools which are shortlisted for comparison are SentiStrength, TextBlob, Stanfords recursive neural tensor network, text-processing, GATE and DatumBox.

Utterance	Golden Set	TextBlob	Stanford	SentiStrength	DatumBox	text-processing	GATE
And, there is nothing perfect. But when you're watching somebody die and, you know, they're dying they're dying. This isn't a mis-diagnosis.	NEGATIVE	POSITIVE	NEGATIVE	POSITIVE	NEGATIVE	NEGATIVE	POSITIVE
Very good. I agree with your statement and the problem; I just don't agree with the solution, which is why I'm recommending a no vote. Again, as stated by the ACLU, we already have laws on the books, which allow up to four years if someone is charged under elder abuse statute.	NEGATIVE	POSITIVE	POSITIVE	POSITIVE	NEGATIVE	NEGATIVE	POSITIVE
And we should not have a lot of very good people opting out of running because they don't want to spend all their time on the telephone raising money.	NEGATIVE	POSITIVE	NEGATIVE	POSITIVE	NEGATIVE	NEGATIVE	NEGATIVE
I will say this, when there's this level of confusion around a bill, maybe it is best to work on it another day. Thank you, Mr. Speaker.	NEGATIVE	POSITIVE	POSITIVE	NEGATIVE	NEGATIVE	POSITIVE	POSITIVE
But-- So it seemed to me that the admissions part of the bill was in very good shape. Nobody commented on it. And it is one of the areas where we have pause because there are some charter schools that do want either money or volunteer time from parents. And that could be discouraging for some parents that don't have either of those things because of their income level.	NEGATIVE	POSITIVE	NEGATIVE	POSITIVE	NEGATIVE	NEGATIVE	POSITIVE

Figure 3.4: Sentiment tool comparison table

3.2.1 SentiStrength

This is a sentiment analysis (opinion mining) program, which employs several novel methods to simultaneously extract positive and negative sentiment strength from short informal electronic text. SentiStrength uses a dictionary of sentiment words with associated strength measures for expressing sentiment of the text. The SentiStrength sentiment analysis tool uses a bag of words approach, it finds the polarity at word level and then tabulates the overall sentiment of the sentence. The problem with sentiment analysis tools are dependent on word level polarity evaluation is when non-literal phrases or sarcastic comments appear, such tools are not accurate. For example, for oxymoron terms like “pretty ugly”, “living dead”, “amazingly awful”, etc. where two words of opposite meaning are attached, SentiStrength fails to decipher the sentiment.

SentiStrength was developed through an initial set of 2,600 human-classified MySpace comments, and evaluated on a further random sample of 1,041 MySpace comments. SentiStrength can predict positive emotion with 60.6 percent accuracy and negative emotion with 72.8 percent accuracy, both based upon strength scales of 1-5 [17]. SentiStrength is free for academic research.

Figure 3.5, displays the accuracy graph for the SentiStrength sentiment analysis tool outcome and the golden set. In the Figure 3.5, the y-axis has the values as Negative, Neutral and Positive and the x-axis has the numeric values. The red color depicts the neutral, green the positive and blue the negative outcome. This color scheme is the same for all the sentiment tool comparison figures. The overall accuracy for the SentiStrength sentiment analysis tool is 72% based on utterance data.

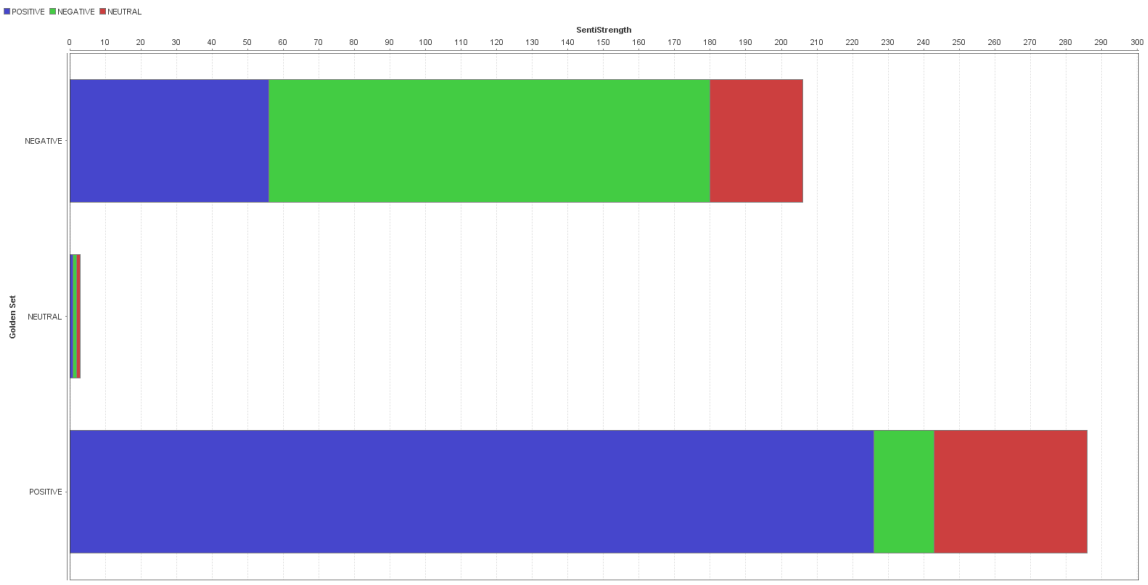


Figure 3.5: Accuracy graph SentiStrength

3.2.2 AlchemyAPI

This is a paid service from IBM which provides several text processing APIs such as sentiment analysis, emotion analysis, document categorization, keywords etc. The sentiment analysis API promises results on very short texts (e.g., tweets) as well as relatively long texts (e.g., news articles). The AlchemyAPI for a text fragment returns a status, a language, a score and a type. The score is in the range (-1, 1), the type is the sentiment of the text and is based on the score. For negative scores, the type is negative, conversely for positive scores, the type is positive. For a score of 0, the

type is neutral.

Since our requirement is a tool is free and open source, we did not use AlchemyAPI. Though it provides roughly 1000 free API requests per day, it will not work for the problem where the number of utterances runs into millions.

3.2.3 Stanford's Recursive Neural Tensor Network

Stanford's recursive neural tensor network is an open source sentiment analysis solution that uses Stanford's treebank corpus to find the sentiment at sentence level rather than at word level. The Stanford Sentiment Treebank is the first corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language. The corpus is based on the dataset introduced by [3] and consists of 11,855 single sentences extracted from movie reviews.

Sentiment Treebank Semantic word spaces have been very useful but cannot express the sentiment of longer phrases in a principled way. A Recursive Neural Tensor Network, when trained on the new treebank, outperforms all previous methods on several metrics. Recursive neural tensor network pushes the state of the art in single sentence positive/negative classification from 80% up to 85.4% when tested on social media data [20]. The accuracy of predicting fine-grained sentiment labels for all phrases reaches 80.7%, an improvement of 9.7% over bag of features baselines [20]. The accuracy numbers for sentiment analysis solution provided by Stanford are based on tests on social media data.

Figure 3.6 displays the accuracy graph for the Stanford sentiment tool outcome and the golden set.The overall accuracy for the Stanford sentiment analysis tool is 75%.

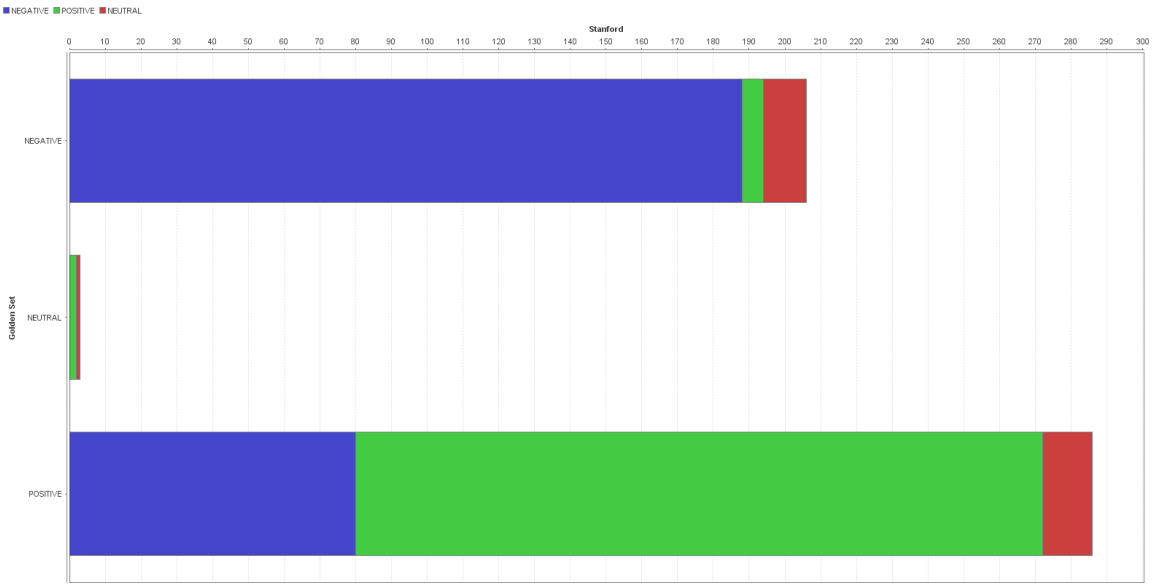


Figure 3.6: Accuracy graph Stanford

3.2.4 LingPipe

This is an open source sentiment analysis solution, it performs the sentiment analysis in three steps. Firstly, it uses a language classification framework to do two classification tasks: separating subjective from objective sentences, secondly to separate positive from negative statements and lastly, they show how to build a hierarchical classifier by composing these models.

The problem with this approach is that we don't have large tagged database or golden set for training this tool. Due to this issue, we avoided using LingPipe functionality. Moreover as has been the case for previous approaches, this approach is only tested for movie reviews.

3.2.5 ElasticSearch

Like LingPipe, Elasticsearch sentient analysis uses supervised learning training data to train its internal models and then predict the sentiment of the statement. Since

this method can be used to train any type of data whether it be social media or statements made by people, the tagged databases available on internet is only for social media. So, to use this technology one needs to develop trained models for the type of data they have. This approach would be really helpful if we have supervised data available, since for Digital Democracy database we dont have such type of data we will not be considering this tool for evaluation.

3.2.6 Lexalytics

Lexalytics is a company which processes billions of documents daily commercially and provides services such as text categorization, sentiment analysis, entity insights etc to its customers. Lexalytics acquired Semantria which provides a paid service to extract the sentiment of a document in real time, though this approach comes closes to the problem we have but it has a heavy license fee for using its services.

The internal functionality of the solution provided by Lexalytics in the first step determines the tone of a document by breaking the document into its basic parts of speech (POS tagging). POS tagging is a mature technology that identifies all the structural elements of a document or sentence, including verbs, nouns, adjectives, adverbs, etc. To identify the sentiment, adjective and noun combinations like "horrible pitching" and "devastating loss" are extracted and then compared with a vast dictionary created over time with tagged data.

3.2.7 NLTK's TextBlob

NLTK is a Python library for processing textual data. It provides APIs for solving common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, and more. TextBlob's sentiment analysis tool provides two metrics: polarity and subjectivity. The input to its sentiment API

is a sentence which can be a string, text, sentence, chunk, word or a synset. The value of polarity and subjectivity ranges between -1 and 1.

In the internal function of TextBlob's sentiment engine, written text is broadly categorized into two types: facts and opinions. Opinions carry people's sentiments, appraisals, and feelings toward the world. The pattern module bundles a lexicon of adjectives (e.g., good, bad, amazing, irritating, etc) that occur frequently in product reviews, annotated with scores for sentiment polarity and subjectivity [23]. This is an open source solution with an easy implementation which uses the best features of both the NLTK and pattern module.

Figure 3.7 displays the accuracy graph for the TextBlob sentiment analysis tool outcome and the golden set. The overall accuracy of the TextBlob when compared with its golden set is 94%.

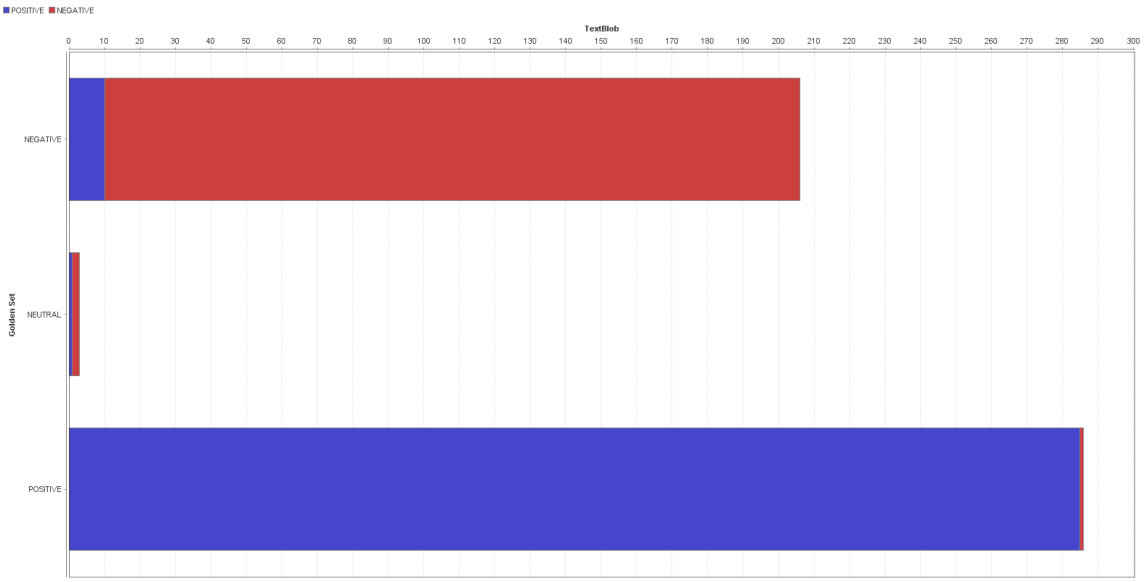


Figure 3.7: Accuracy graph TextBlob

3.2.8 Datumbox

Datumbox is an open-source Machine Learning framework written in Java which allows the rapid development of machine learning and statistical applications. The main focus of the framework is to include a large number of machine learning algorithms and statistical methods which are able to handle large sized datasets [20]. Datumbox integrates number of pre-trained models which allow users to perform sentiment analysis (Document and Twitter), subjectivity analysis, topic classification, etc.

Figure 3.8 , displays the accuracy graph for the DatumBox sentiment analysis tool outcome and the golden set. The overall accuracy for the DatumBox sentiment analysis tool is 70%.

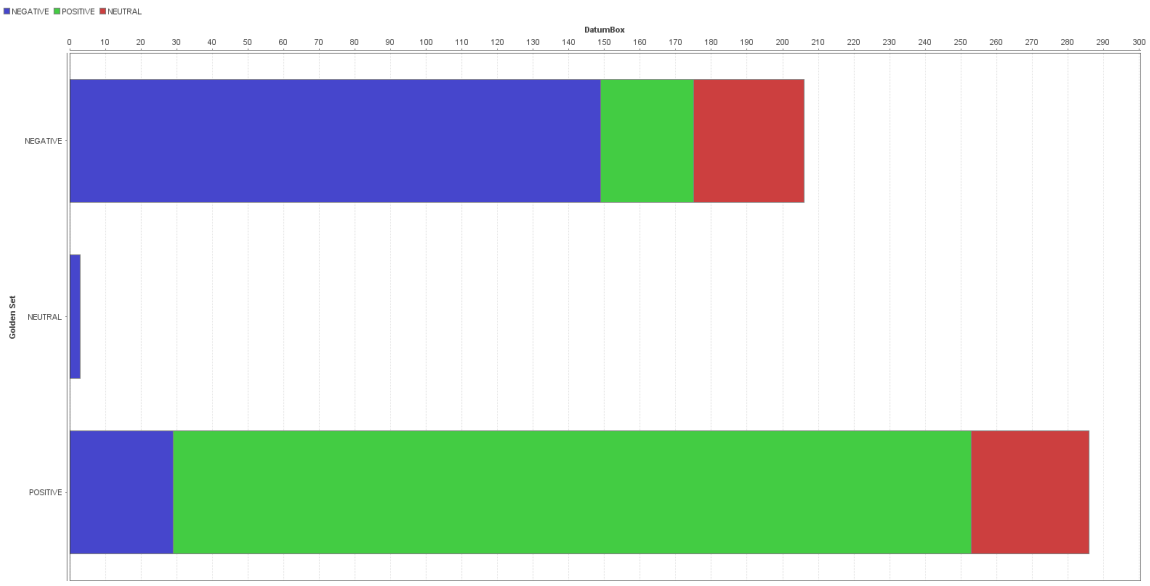


Figure 3.8: Accuracy graph DatumBox

3.2.9 Text Processing

The Text Processing API is a HTTP web Service for text mining and natural language processing and provides output in the form of JSON. It is currently free and open for

public use without authentication, but one needs to buy a commercial license. The Text Processing API uses the NLTK at back-end to analyze the sentiment of text and performs a HTTP POST to the url 'http://text-processing.com/api/sentiment/' with form encoded data containing the text sent to analyze. A JSON object response with two attributes is returned a label and a probability. Label will be either 'pos' if the text is determined to be positive, 'neg' if the text is negative, 'neutral' if the text is neither positive nor negative. The probability object contains the probability for each label, 'neg' and 'pos' labels will add up to 1, while neutral is standalone. If the value of the neutral probability tag is greater than 0.5 then the label will be marked as neutral, else the label is marked as positive or negative whichever has higher probability.

Figure 3.9 , displays the accuracy graph for the Text Processing sentiment analysis tool outcome and the golden set. The overall accuracy for the Text Processing sentiment analysis tool is 82%.

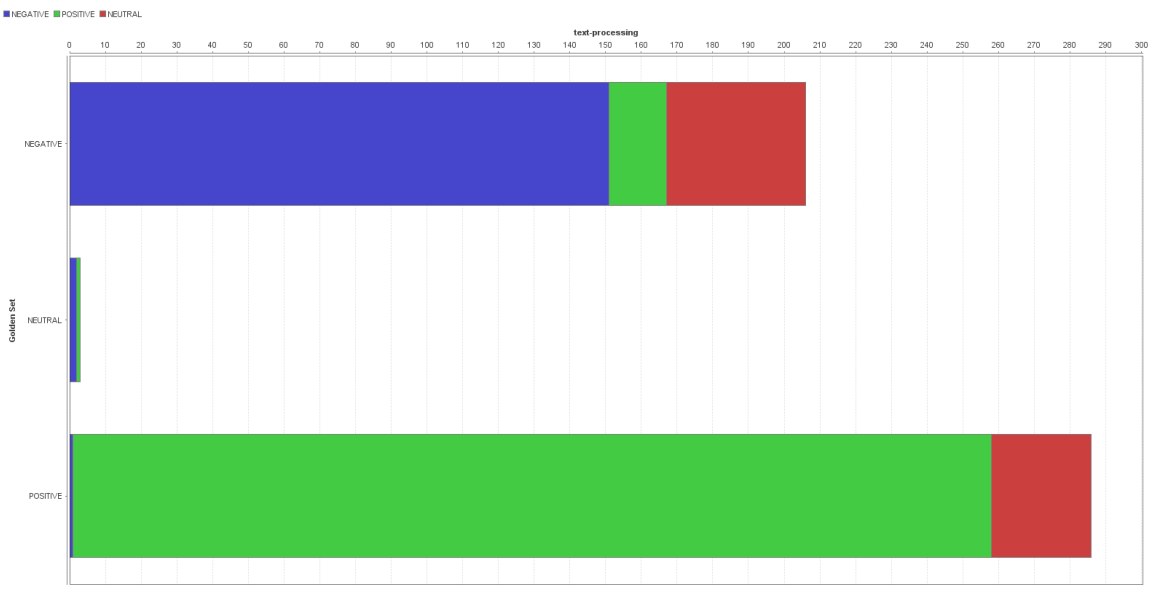


Figure 3.9: Accuracy graph textProcessing

3.2.10 GATE

GATE stands for General architecture for text engineering, this is an open-source software for text processing. The GATE provides sentiment analysis through DecarboNet, this is a research project funded by the European Commission to investigate the potential of social platforms in mitigating climate change. The web service takes as input a document or set of documents, and outputs those documents as JSON documents with opinion, term and URI information. Sentiment is classified into positive, negative and neutral polarity, as well as more fine-grained emotions such as fear, anger, joy etc.

GATE has been in existence since 1995 and claims to be fairly accurate when predicting sentiment of social media on topics of climate change. We tested this service with respect to the Digital Democracy utterances to access the sentiment quality. Figure 3.10 , displays the accuracy graph for the GATE sentiment analysis tool outcome and the golden set. The overall accuracy for the GATE sentiment analysis tool is 64%.

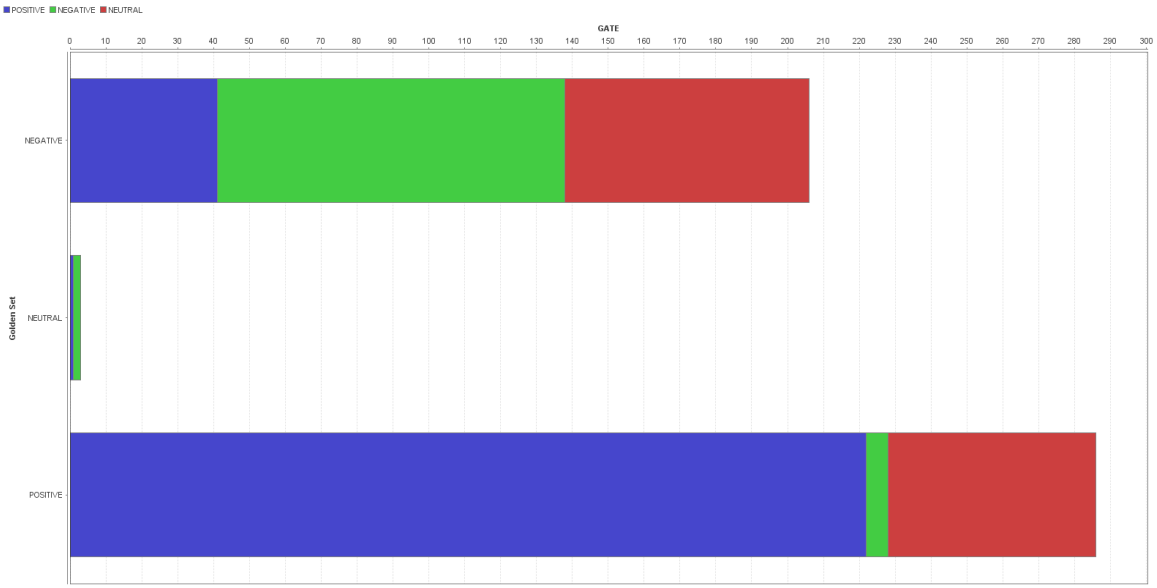


Figure 3.10: Accuracy graph Gate

3.2.11 Overall sentiment tool comparison

Figure 3.11 shows the overall sentiment tool comparison. Since TextBlob by NLTK has the best accuracy among the all sentiment tools we compared, we chose TextBlob for our sentiment analysis.

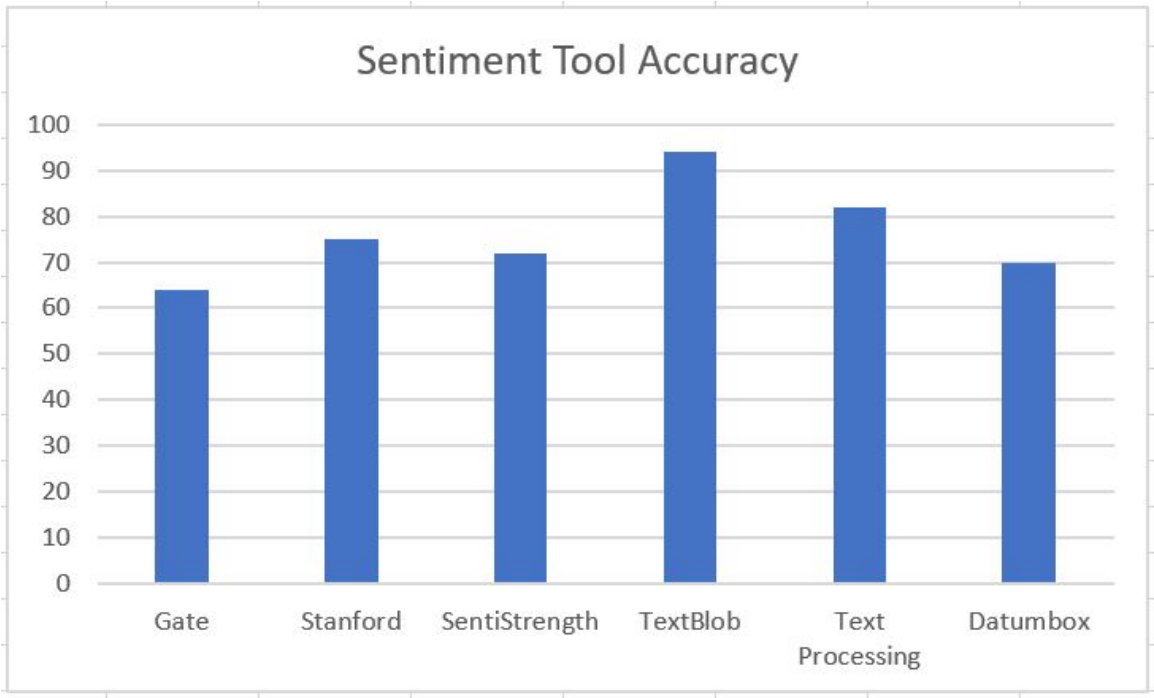


Figure 3.11: Overall sentiment tool comparison

3.3 Machine Learning Algorithm

For topic modeling we use supervised learning algorithms. This is the process where models are trained on labeled data and these models then provide label to the unlabeled data. There are multiple supervised learning algorithms available, but which one best fits the experiment data needs to be tested [4, 15, 11]. We test our feature set on various supervised learning algorithms such as Support Vector Machines, Random Forests and Keras with Tensor flow in background.

3.3.1 Support Vector Machines

This is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we take pre-labeled data and then generate the hyperplane to separate the data by label. We plot each data item along with feature set with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes. There are various kernel tricks in SVM such as linear, polynomial, sigmoid and RBF (radial basis function) as shown in Figure 3.12. SVM with RBF kernel provided us with the best in class accuracy and F1 score.

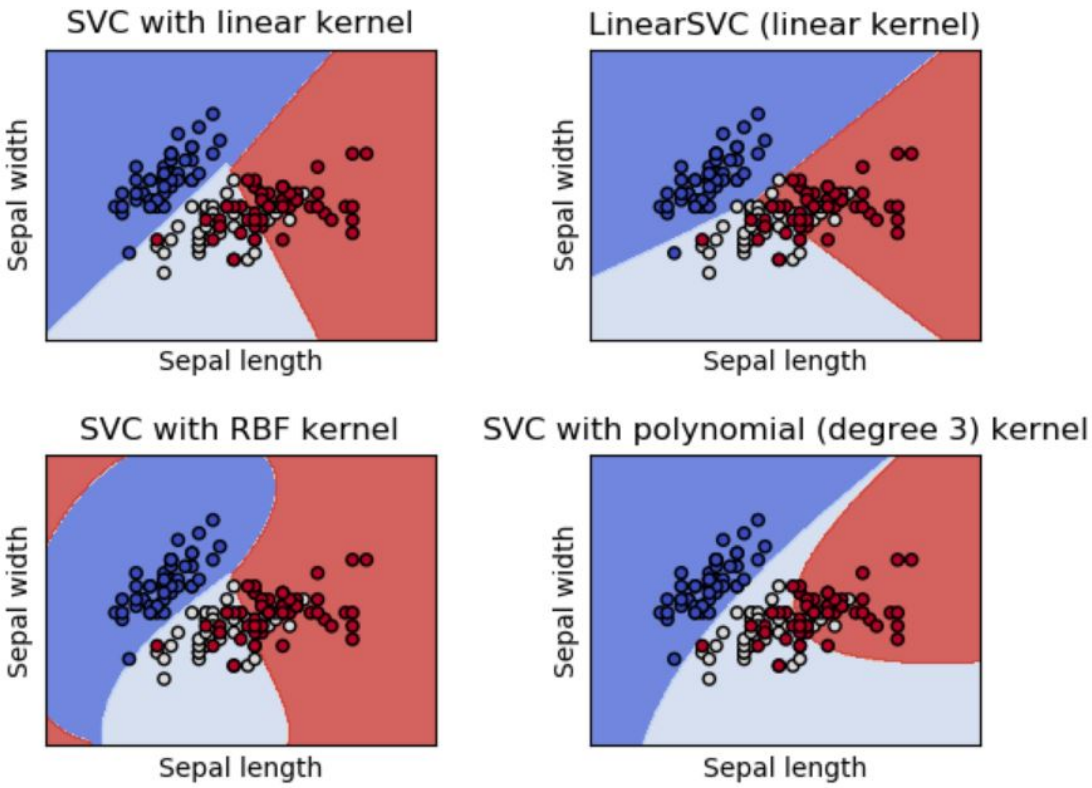


Figure 3.12: SVM example with various kernel

3.3.2 Random Forest

A Random Forest is a meta-estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.We were interested in exploring Random Forest algorithm as the prediction categories for the votes is binary and decision tress perform well with binary classifiers.

3.3.3 TensorFlow

TensorFlow is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. TensorFlow was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well.

3.4 Software Tools

The detailed architecture of the system for this work is described in Chapter 4, is implemented in the Python language and partially in Java for some processing. We make reference to the following libraries that we use for data representation, prediction and analysis.

3.4.1 Pandas

Pandas is a popular python software library providing fast, flexible, and expressive data structures designed to make working with labeled data in an intuitive manner. To facilitate efficient selection and merging of tabular data, we use DataFrames from the Pandas library to store all data that we query from the Digital Democracy database. In this format, we can easily aggregate data by personId, bill number, hearingID and discussionID.

3.4.2 scikit-learn

To perform supervised learning on our data we use the scikit-learn machine learning library. In addition to learning algorithms, this library provides methods for feature extraction and selection, a means to easily collect learners into an ensemble, as well as scoring metrics to assess predictions.

3.4.3 The Natural Language Toolkit

For majority of our text processing tasks we used Natural Language Toolkit, which provides a simple interface and a wealth of natural language processing techniques and corpora across various domains.Some of the text processing libraries provided by NLTK are for classification, tokenization, stemming, tagging, parsing, and semantic reasoning.

3.4.4 Keras

Keras is a high-level neural networks API, written in Python and capable of running on top of Tensorflow, CNTK, or Theano. We are using Keras on top of Tensorflow library. Generally keras is used for applications which require deep learning on data as

this library allows for easy and fast prototyping (through user friendliness, modularity, and extensibility) and supports both convolutional networks and recurrent networks, as well as combinations of the two.

Chapter 4

EXPERIMENTAL DESIGN

The transcripts of committee hearings are a set of unstructured textual data. The quality of transcripts is strictly based on the quality of the speech the member makes and the transcription system. If there are multiple speakers or if the pronunciation of certain words is unusual, the transcripts tend to be erroneous.

For these experiments, we assume that the transcripts which we have are accurate and we need minimal preprocessing on them syntactically. The design of the vote prediction system is shown in Figure 4.1. The step-by-step description for each phase and each feature used in prediction modeling is mentioned below.

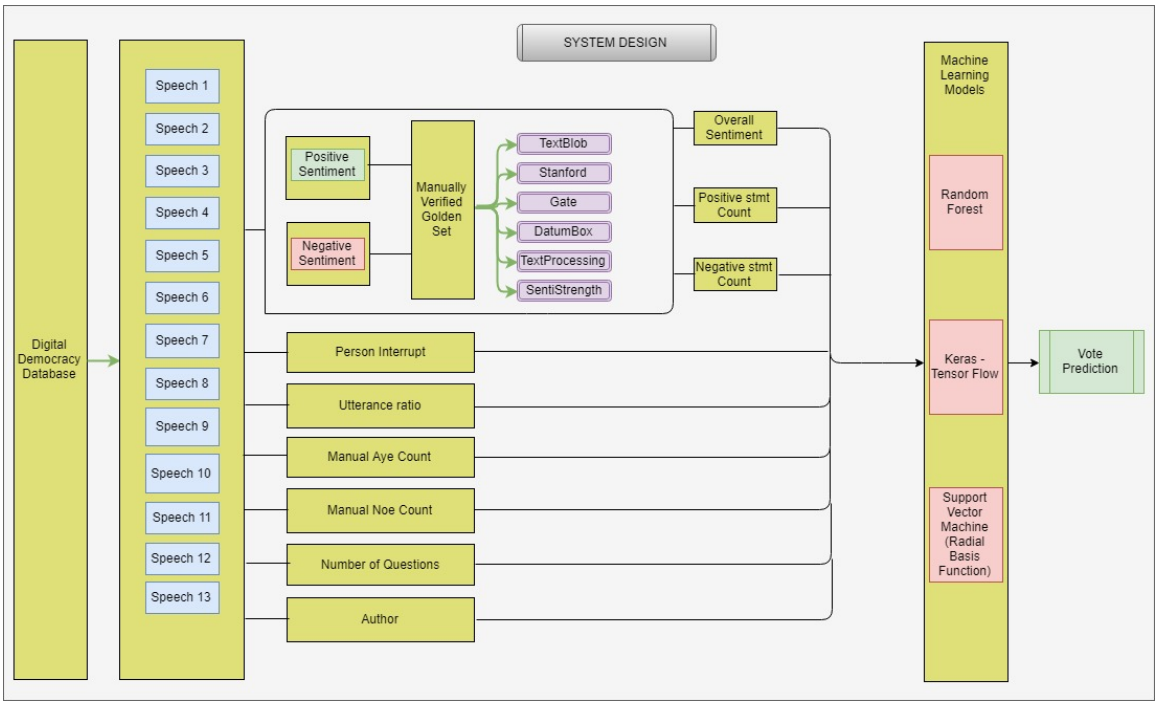


Figure 4.1: The overall system design

4.1 Data Extraction

In the first step of this process, we extract the utterances (short frames of speech), bill details, member details and voting details for each hearing and each bill. This is complex process as a bill could be discussed over multiple sessions and can easily run into multiple hearings which in-turn can run into multiple videos thus leaving transcripts scattered throughout. The Digital Democracy database is in MySQL and there are 34 tables of which 10 tables are used for data extraction. We use SQL to query the database and extract the relevant information. The fields extracted from the database can be seen in Figure 4.2. The data extracted from the database is then stored in dataframes using the 'Pandas' library from Python such that data can be processed in a tabular manner. The Pandas library provides high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Since the number of utterances runs into more than 100 million, Pandas provides an appropriate data structure to work with.

utterance_text	video_utterance_vid	pid	billdiscussion_utterance_did	bid	author	hid	voteId	bill_result	bill_ayes	bill_naes	bill_abstain	result	person_interrupt	utterance_ratio_overall	sentiment	pos_stmts_count	neg_stmts_count	manual_aye_pred_count	manual_noe_pred_count	question_count
Good morning, Mr. Speaker and members. AB 1 is back for concurrence, Senate amendments clarified this bill to apply to all cities including charter cities and added Senator Nielsen as a co-author. AB 1 is a simple bill. It will prohibit municipalities from fining residents who choose to conserve water by not watering their lawns during a drought emergency. I respectfully ask for your aye vote.Good morning, Mr. Speaker and members. AB 1 is back for concurrence, Senate	4994	13	5004	CA_201520160AB1	1	773	40359	(PASS)	21	1	3	AYE	0.0	0.17	0.188	0.3	0.75	1	0	0
Mrs. Brown, you may open. Good morning.Mrs. Brown, you may open. Good morning. Thank you, Mrs. Brown. Seeing no discussion or debate on this item, the clerk will open the roll. All members vote who desire to vote. All members vote who desire to vote. All members vote who desire to vote. Clerk willThank you, Mrs. Brown. Seeing no discussion or debate on this item, the clerk will open the roll. All members vote who desire to vote. All members vote who desire to vote. All members vote who desire to vote. Clerk	4994	31	5004	CA_201520160AB2	0	773	40359	(PASS)	25	5	0	AYE	0.5	0.03	0.23	0.2	0.93	0	0	0
Thank you Mr. Chair. Senator Brown, local municipalities should be taking care of this and you'd think they'd have understanding fining people for not watering their lawns right now. It's sort of...I'm concerned about local control. These mayors should pony up and get a clue about what's going on. So I just wonder what kind of message are we sending and is it	1406	4657	1637	CA_201520160AB3	0	305	39065	(PASS)	7	0	0	AYE	0.0	0.02	0.22	0.5	0.83	1	0	1

Figure 4.2: Data Sample used for training prediction model

The attributes we extract from the Digital Democracy table are, 'personID' which represents a unique key value for the member of committee, 'utterance' which is the utterance the member committee makes, 'billId' is the key which identifies which

bill the utterance is from. 'videoId' is a tricky attribute as a bill can be in multiple hearings and there can be multiple videos of same bill so the extraction of videoId is really important if we have to extract the utterance. 'HearingId' is the ID which identifies the hearing. A hearing can have multiple bill discussions. 'Vote outcome' is the vote the member gives for that billID in the particular hearing.

4.2 Data Organizing

The second stage is mainly preprocessing of textual data. A speech is a collection of utterances by a member until he is interrupted by another member during his speech. To tabulate the speech, we concatenate the utterances said by a single member without interruption. This is done because the utterances in the database can be one-third of a sentence or maybe four sentences Figure 4.3. For example, if a member spoke 10 lines without being interrupted, the database would have 10 entries of the same member on the same hearing. This leads to a situation where multiple entries of utterance by the same member have different entries. To resolve this issue, we clubbed these continuous entries to one entry which we call a speech. This enabled us to develop chunks of uninterrupted speeches spoken by the member, thereby helping us understand the context of the member's speech more clearly. The speeches after this step provided us with the insight that the member stopped either after completing their point or was interrupted by another member. The interruption provides us with a big indication that the person who interrupted had some issues with the points made by the speaker before. All the speeches said by a member in the entire discussion are joined. We also count how many times each person was interrupted.

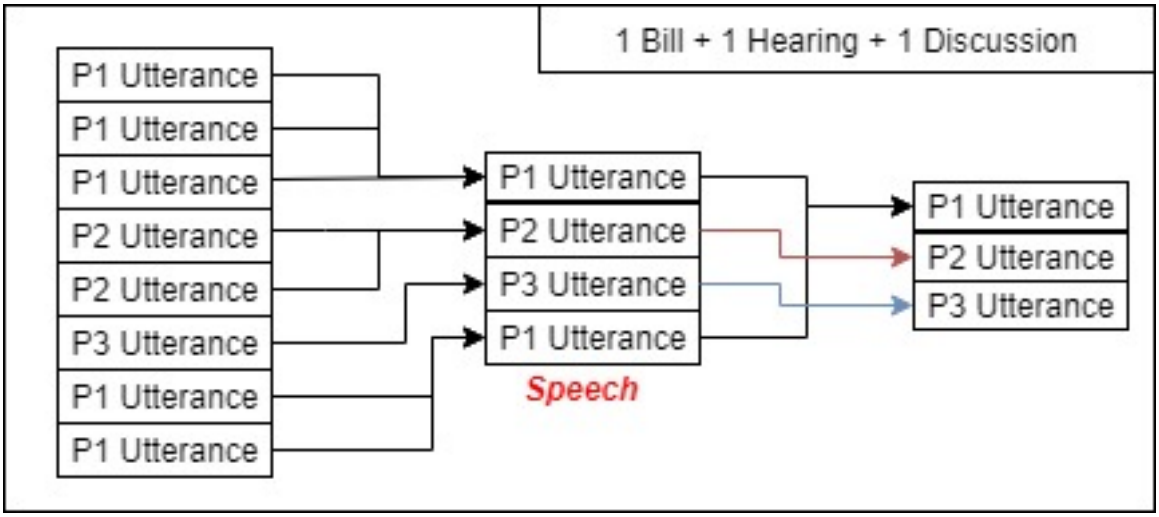


Figure 4.3: Tabulating the speech from utterances

4.3 Feature Extraction

We extracted the features based on certain hypothesis. The overview of the hand crafted features extracted can be seen in Figure 4.4. In total, we extracted nine features. Their values are normalized accordingly for better prediction accuracy.

4.3.1 Speech Interrupt

The number of times a member speech was interrupted made a feature for our prediction model. Speech, as explained in the section before, is collection of utterances by a member until he is interrupted by another member. This happens in one discussion and in one hearing. This feature was normalized by dividing the person interrupted by number of sentences he said in a speech. Speech interrupt was significantly high if the person who interrupted was against the bill and was more likely to vote NOE. This was observed experimentally when all NOE votes were filtered and the value of speech interrupt feature was noted. The value was high as compared to when people voted AYE. This was also proved correct when removing this feature from the nine

Feature Name	Description	Is Normalized ?
Speech_Interrupt	Number of times the member was interrupted	Yes, interrupt count is divided by total sentences member said
Volume_of_Speech	Ratio of number of words member said to total number of words spoken in one dicussion.	Yes, word count per member is divided by overall word count in discussion
Speech_Sentiment	Sentiment of overall speech	The range is already between 0 and 1.
Positive_utterance_ratio	Ratio of positive statements in speech	Yes, Count of positive statements in overall speech is divided by number of sentences in a speech
Negative_utterance_ratio	Ratio of negative statements in speech	Yes, Count of negative statements in overall speech is divided by number of sentences in a speech
Question_count	Number of questions asked in a speech	Yes, number of questions asked in a speech is divided by total number of statements in the speech.
Hit_Rate_Aye	Intersection count of AYE common phrases in speech and in AYE dictionary	Yes, intersection count of AYE common phrases in speech and in AYE dictionary divided by total number of AYE phrases in dictionary.
Hit_Rate_Noe	Intersection count of NOE common phrases in speech and in NOE dictionary	Yes, intersection count of NOE common phrases in speech and in NOE dictionary divided by total number of NOE phrases in dictionary.
Is_Author	Binary feature if speech is by Author/Co-author	The feature is already in binary format.

Figure 4.4: Hand crafted feature overview

feature list caused the F1 score to go down by 2%. This can be seen in Figure 5.1.

4.3.2 Volume of Speech

The volume of speech (measured as number of words) said by a member, makes a new feature for us. This feature is normalized by dividing the count of words spoken by a member in his speech to the total words spoken by all members in one discussion. The main idea behind this feature is that the higher the value of ratio the better the chances are that member is in favor of the bill. This was observed experimentally. Members who were authors/co-authors tend to present information about the bill and then justify the issues which other members had during discussion.

4.3.3 Speech Sentiment

This feature is one of the basic features which is mandatory when predicting vote from a speech. Here we tabulate the sentiment of each speech chunk by chunk and

classify each utterance based on its polarity and subjectivity. This gives us an insight into whether there is a correlation between members tone and vote they cast. Since sentiment analysis is a critical part of the prediction process, we engage in evaluating appropriate tools and services as explained in Section 3.2 . For this evaluation, we use a manually tagged set of 500 utterances with their polarity as benchmark and execute all the tools against those utterances. However, all the tools available in market are designed specifically for social media data. Since our data is more of a speech/regular data, we cannot use the advertised accuracy claimed by these tools. We run the tools on our own customized data for comparison. The tools which we shortlist for comparison are SentiStrength, Alchemy API, LingPipe, ElasticSearch sentiment analyzer, Lexalytics, Recursive Neural Tensor Network and NLTK's Text Blob. The best results which we get by using single sentiment engine for our speech data is with TextBlob with an accuracy of 97 percent.

The feature as described above is the sentiment score for the entire speech a member said in a discussion. The usefulness of this feature is also proven experimentally. When we removed this feature from the prediction modeling, the F1 score dropped by 1.5%.

4.3.4 Positive utterance ratio

On manually analyzing the speeches, we observe some disparity. The sentiment score is not very accurate as people typically start speaking positively but then say something negative. Due to this, the overall sentiment score is never as negative as we predicted. So we decide to count the positive and negative sentiment score per utterance in a speech. Performing a per utterance sentiment analysis doesn't provide the context the statement was used for. Neverthless, it gives an overview of how the entire speech is positioned. The value of this feature is also normalized so that it is

in sync with the value range of other features. We divide the count of the positive statements in overall speech by the number of statements in the speech by a member. The experimental justification of this feature is also shown in Figure 5.1.

4.3.5 Negative utterance ratio

As discussed in previous section in this feature instead of taking the positive sentiment count of statements line by line we take negative sentiment count. The experimental viability of this feature is also shown in Figure 5.1 , where the F1 score dips as soon as we remove this feature.

4.3.6 Question count

We observed the question count feature while analyzing the NOE vote patterns. One of the most common characters found in all the NOE votes speeches was a question mark. Thus, we decided to use this as a feature and check how it impacts the accuracy. Here we tabulate the number of questions asked by the member while speaking about a bill in a discussion. Normalization here was done by dividing the question count by the number of statements in the speech. This is particularly high when a person has doubts about a bill and is more likely to vote against the bill. The second experimental justification showed that when we removed this feature, the F1 score dropped by more than 5%. This was observed by calculating the average of multiple iterations.

4.3.7 Hit rate AYE

The process to tabulate the 'hit rate AYE' is a lengthy one. We create a dictionary of words and phrases which are common when people vote AYE. To tabulate the value for this feature, we first execute the n-gram NLTK filter with token count starting

from 1 to 5. This is followed by listing 25 of the most common tokens for each category. Finally, we manually review all the n-gram results for AYE speeches and create a golden set. Some examples of words/phrases in the AYE dictionary are, 'Aye', 'like to move the bill', etc. The value of this feature is normalized by taking the intersection of AYE common phrases in speech and in AYE dictionary and then dividing by total number of AYE phrases in dictionary.

4.3.8 Hit rate NOE

Similar to the previous section, 'hit rate NOE' is tabulated by finding words and phrases which are common when people vote NOE. We create a n-gram model and list 25 of the most common tokens using NLTK library. The example of phrases which are there in golden set for NOE are like "don't support", "no vote", "rise in opposition", "cannot support", etc. The final value of 'hit rate Not' is evaluated and normalized by taking the intersection count of NOE common phrases in speech and in NOE dictionary and then dividing by total number of NOE phrases in dictionary.

4.3.9 Is author

The last feature for our prediction model is a binary feature which indicates if the person speaking authored/co-authored the bill under discussion; chances are that if the member authored the bill then they are more likely to vote AYE. On experimental analysis, we found that 99.2% of the people who authored/co-authored the bill voted in favor of the bill.

4.4 Prediction Model

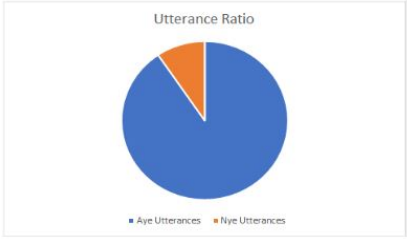
All the features as described in previous section are run against various prediction models to find the best accuracy. But before we discuss about the prediction models, we will describe the details about the prediction labels and data set used.

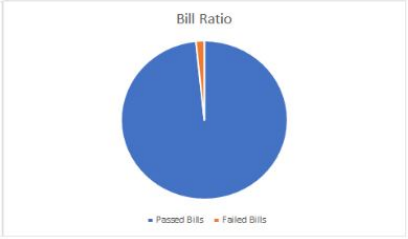
4.4.1 Label Description

Speeches by members are tagged per bill discussion (a portion of a hearing typically lasting about 20-40 minutes followed by a vote). Speeches by members who vote Aye are tagged as the positive class, and those who vote NOE or abstain as negative. As the concept of 'passage' in legislative discourse is important, we count abstain votes the same as no votes for binary classification purposes.

4.4.2 DataSet Description

As discussed in Chapter 3, the dataset is highly skewed towards the AYE votes. This is shown in Figure 4.5. One can observe from the pie chart that the data is highly skewed towards AYE votes. The number of failed bills is not even 2% of the all bills tabled in the California senate in session 2015-16.





Speech Count	40788
Aye Speeches	36925
Noe Speeches	3863
Bill Count	5198
Passed Bills	5113
Failed Bills	85

Figure 4.5: The overall dataset description

Controversial Bills

On observing the data and bill outcomes, we discovered that many bills were passed unanimously. However, we were more interested in the bills which had atleast one NOE vote. This bill set was used for doing vote prediction under various settings which will be discussed later in the section. This bifurcation helped us to normalize the dataset a bit more towards balanced approach. As earlier when we tried to train our model with all bills, the NOE utterances were so few that majority of the class predicted was AYE.

Balanced Bills

For 'balanced bill' dataset which we used for bill prediction we filtered bills based on number of speeches in that bill discussion. We placed a lower limit of 3 speeches in a discussion and an upper limit of 23 speeches in the discussion. As we don't want to be in a situation where just based on one speech we predict the bill outcome. Due to huge difference in passed and failed bills even after the first filtering, we further limited our dataset to equal number of failed and passed bills.Twenty percent bills were chosen, such as equal number of failed and passed bill occur in test set, rest eighty percent of the bills were used for training the model. The majority count of the vote prediction decides the fate of the bill.

4.4.3 Machine Learning Algorithms

Since we are doing supervised learning prediction modeling, the various algorithms we explore are Naive Bayes algorithm, Maximum Entropy algorithm, Decision Tree algorithm, Boosting(AdaBoosting and gradient boosted regression techniques), Random forest algorithm, Support vector machines and Neural Networks(Tensor flow).

In limited early experiment we choose three prediction models as they achieved high accuracy, the three models are Support vector machines, Random Forest and Tensor flow.

We use Python's scikit-learn library for SVM, as scikit-learn features various classification, regression and clustering algorithms also it is designed to inter operate with the Python numerical and scientific libraries NumPy and SciPy. Support Vector Machine with Radial bias function(RBF) kernel trick was comparatively best among the various kernel tricks tried for SVM. The feature set we used for training and prediction in this model are the hand-crafted features we tabulated in previous section. We use balanced RBF kernel to balance out the data(AYE/NOW utterances) and then train the model, we also use unbalanced data for more realistic results. The class weight when used as 'balanced' nullifies the effect of data being skewed to one category. The training data is divided into 90 to 10 split where 90 percent of randomly selected items is used for training and 10 percent for prediction.

The second prediction model we use is Random Forest, the training model generated thorough this algorithm provide the vote prediction F1 score much higher than the SVM with RBF kernel. Here the features set used for training were same as the one used in SVM, the hand-crafted feature set. The details about the results from all algorithms are explained in the section 5.2.

The third prediction model which we use for the evaluation is Tensorflow, the library Keras which we use uses tensorflow at back-end. Through this model we achieved the highest F1 score, details for the result can be found in section 5.3. The configuration we use for tensorflow are three hidden layers, loss used was binary crossentropy, optimizer used was adam and activation function used was relu and sigmoid. The reason we choose only three layer in our experiment is because we got the highest F1 score with three layers as we increased number of layers the F1

score dropped.The feature set we use for tensorflow is slight different because the results with hand-crafted feature set the results were not very accurate. The detailed comparison result can be seen in Figure 5.6

To experiment more with tensorflow and by observing its nature we decided to use a different input to check its accuracy. So we use speeches only as the feature, in preprocessing stage the speeches were stemmed such as the feature vector matrix generated in the next step was more accurate. This was achieved through the NLTK's text processing library. Post pre-processing, the text processing api of Keras library was used which on the basis of TFIDF(Term frequency Inverse document frequency) generated the matrix of feature vector per utterance. The number of features in the feature vector were approximately 26000 and the number of hidden layers used in the neural network were three. The settings used in our experiment are explained earlier.

Chapter 5

RESULTS

For tabulating the accuracy of the prediction model, the big challenge was that the data was heavily skewed towards 'AYE' votes. Out of 41000 utterances by members of legislative assembly, 37000 were associated with an AYE vote, leaving the data highly biased to one category. Hardly 10 percent of the data had utterances for 'NOE' category.

5.1 Support Vector Machine

While using the SVM prediction modeling, we use balanced the RBF kernel to balance out the data(AYE/NOW utterances) and then train the model, we also use unbalanced data for more realistic results. The class weight setting when used as 'balanced' nullifies the effect of data being skewed to one category. The training data is divided into 90 to 10 split where 90 percent of randomly selected items are used for training and 10 percent for testing.

5.1.1 Vote Prediction

For experiment 1 to predict the vote outcome we have two prediction models, one created from unbalanced training data and the other from balanced training data. The result of this experiment is shown in Figure 5.1 . This figure shows the accuracy, precision, recall and F1 scores for the vote prediction model over multiple iterations and with various features. The results show that the accuracy in case of unbalanced-SVM is much higher when compared to the balanced-SVM also the best results are when we consider the complete set of features. The reason for this difference in

accuracy is because the data is highly skewed towards AYE votes, so the model is trained mostly on data that predicts the category as AYE. So if the prediction model predicts all votes as AYE the accuracy will be almost 90 percent. The difference can be seen in the f1 scores where, the unbalanced SVM has a value 57.3 while that of balanced SVM is 61.5.

		Vote Prediction	
		Average Balanced	Average UnBalanced
All 9 Features	Accuracy	65.5	79
	Precision	63.3	82
	Recall	68	57.5
	F1	61.5	57.3
ALL Features minus neg_stmt_count	Accuracy	57.25	79.3
	Precision	59.3	76.3
	Recall	62.4	56.5
	F1	54	56.1
ALL Features minus pos_stmt_count	Accuracy	62.3	80.4
	Precision	61	78.8
	Recall	65.8	57.8
	F1	58.35	57
ALL Features minus question_count	Accuracy	59.3	79.6
	Precision	61.2	76.5
	Recall	65.6	57.5
	F1	56.8	57.5
ALL Features minus person_interrupt	Accuracy	65	79.3
	Precision	60.8	80.5
	Recall	65	56.5
	F1	59.8	55.7

Figure 5.1: SVM vote prediction results, multiple iterations

Since we are exploring the accuracy of our vote prediction modeling tool, we run the model on a subset of bills deemed controversial. There are two kinds of bills in the legislature controversial bills and non-controversial bills. Non-controversial bills are passed unanimously. The California legislature has a high percentage of uncontroversial bills that are unanimously passed. Controversial bills are defined as having at least one member in the voting body who is opposed to the bill. Figure 5.2 shows the results of the vote prediction on the controversial bill subset.

5.1.2 Bill Prediction

For Experiment 2, we list bills that have failed (voted down) in legislature and take equal number of bills that passed to create a balanced training set. On this set of 244

Total bills	2088	Vote Prediction	
Total Utterances	16400		
		Average Balanced	Average UnBalanced
All 9 Features	Accuracy	69.2	84.8
	Precision	59.4	88
	Recall	64.3	57.2
	F1	59.4	58.3

Figure 5.2: SVM vote prediction results

bills we do a random 80-20 split, where utterances from these 80 percent of the bills are taken and then trained with balanced and unbalanced RBF kernel SVM. Before selecting the bills, we filter those bills with at-least 3 and at-max 23 members who speak on that bill in legislature. This constraint is added as bills with only one or two speakers won't be interesting examples for vote prediction. Figure 5.3 shows the results for the bill vote prediction. Accuracy for the unbalanced-SVM is almost same as balanced-SVM, but there is difference in the F1 score.

	Bill Prediction	
	Balanced_TrainingData	UnBalanced_TrainingData
Total Bills	49	49
Accuracy	0.73	0.73
Precision	0.87	0.72
Recall	0.74	1
F1	0.8	0.84
True Positive	26	35
False Positive	4	13
True Negative	10	1
False Negative	9	0

Figure 5.3: SVM bill prediction results

5.2 Random Forest

The Random Forest prediction modeling, we use the controversial bill data tabulated in the previous section as training on unanimous bills won't help us predict the failed bills. The training data is divided into a 80 to 20 split where 80 percent of randomly selected items are used for training and 20 percent for prediction.

5.2.1 Vote Prediction

Figure 5.4 shows the results of the vote prediction on the controversial bill set when using Random Forest machine learning algorithm. We see a significant improvement in the F1 score when compared to the SVM vote prediction F1 score.

		Vote Prediction
All 9 features	Accuracy	75
	Precision	72
	Recall	66
	F1	67

Figure 5.4: Random Forest vote prediction results

5.2.2 Bill Prediction

Figure 5.5 shows the results of the bill prediction using Random Forest machine learning algorithm on the controversial bill set.

	Bill Prediction
Precision	0.59
Recall	0.62
Accuracy	0.86
F1	0.72

Figure 5.5: Random Forest bill prediction results

5.3 Tensor Flow

We use the Keras library with Tensorflow, which is widely used for speech recognition. Initially we tried training Keras with recursive neural network with the hand-crafted feature set, but the accuracy there was not high and the F1 score was a mere 0.49. This can be seen in Figure 5.6, also from this figure we see why we choose three hidden layers, for our data the F1 score dropped significantly as we increased the number of hidden layers.

FeatureSet		Precision	Recall	F1	Accuracy
Document word matrix	3 layers	82.8	82.8	82.8	83
	5 layers	82.3	82.3	82.3	
	10 layers	25.1	50	33.5	
Hand crafted feature set	3 layers	84	54	49	70

Figure 5.6: Tensorflow settings comparison

This is why we decided to train the Tensorflow with speech words. As explained in section 4.4.2, post preprocessing steps such as stemming, vector formation using TFIDF we generate training models. The training model is created on the balanced speech for vote prediction and balanced bills for bill prediction.

5.3.1 Vote Prediction

For vote prediction, we extract all the NOE voted speeches from the controversial bill dataset. These number of negative speeches were around 4000, we took same number of AYE speeches randomly from the controversial bill dataset as number of AYE voted speeches is more than 12000. On running the Tensorflow on this data the result we got is shown in Figure 5.7.

		Vote Prediction
word features	Accuracy	83
	Precision	82.8
	Recall	82.8
	F1	82.8

Figure 5.7: Tensorflow vote prediction results

The accuracy, precision, recall and F1 scores we get from Keras library are the highest and this shows that just based on word features we can achieve a vote prediction accuracy of almost 83%.

5.3.2 Bill Prediction

In bill prediction, we used the balanced bill dataset as explained in Section 4.4.2. We have in total around 244 such bills which we used across all the other machine learning bill prediction experiments. The bills were split 90-10 with 90 percent bills used for training and 10 percent used for testing. The results for the Keras library for predicting the bills can be seen in Figure 5.8. This experiment gave us the best bill prediction F1 score when compared to other prediction algorithms we tested our data on.

	Bill Prediction
Precision	0.76
Recall	0.9
Accuracy	0.76
F1	0.83

Figure 5.8: Tensorflow bill prediction results

5.4 Overall Result Comparison

Figure 5.9 shows the comparison chart of vote prediction done by all the three machine learning algorithms used. The clear winner for vote prediction is Tensorflow.

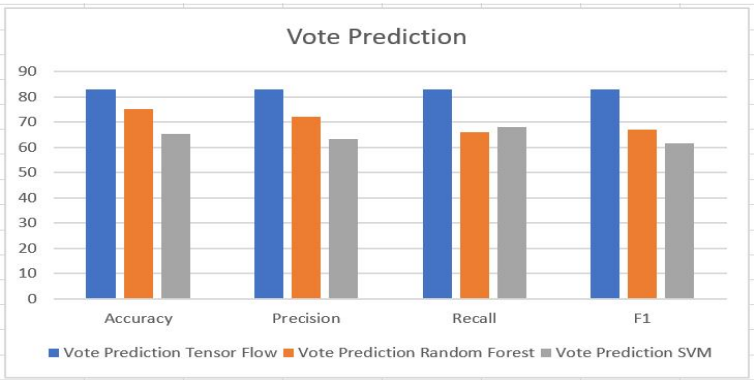


Figure 5.9: Overall vote prediction comparison chart

Figure 5.10 shows the comparison chart of bill prediction done by all the three machine learning algorithms. The bill prediction is calculated by finding the majority of the votes predicted for the utterances said during the bill discussion. If for example there were eight utterances for a bill discussion, then if 4 or more vote prediction of those speeches are 'AYE' then the bill is passed otherwise failed.

The results show that the F1 score for the Tensor flow ie. 0.83 to be maximum

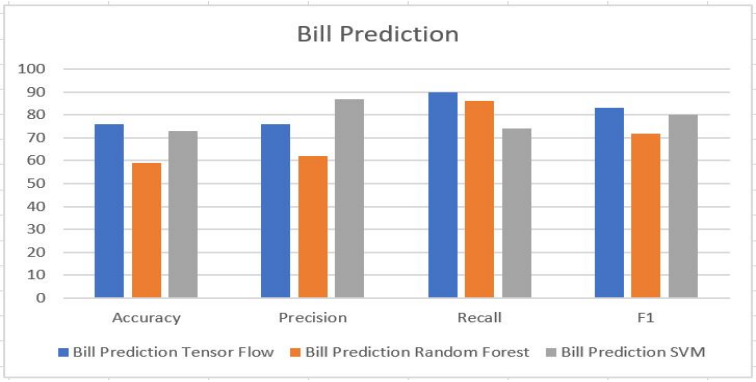


Figure 5.10: Overall bill prediction comparison chart

among the three machine learning algorithms. One noteworthy observation is that even though the vote prediction accuracy was relatively high when compared to other prediction modeling techniques, the bill prediction F1 score was not as high. The reason for this is for a bill to pass some people don't speak just vote NOE, so we have no way to find such kind of cases. Also the bill to pass/fail the majority of the speeches should be in one category so it becomes tricky if we have a smaller number of speeches. It will be interesting to observe in future research if speeches impact the decision of other committee members who went from in favor of the bill to voting against and vice-versa.

Chapter 6

CONCLUSIONS

In this thesis, we show a definite and significant correlation between spoken language and the eventual vote by members of the California legislature, though it is obvious that many more factors need to be analyzed to get better predictions.

Given the highly skewed nature of the votes in our data set, we create a balanced data of equal yes and no votes both at the legislator level and the bill outcome level. The idea is to measure the predictive power of our approach given an even apriori proposition. We are able to achieve accuracies as high as 83% with F1 scores of 0.828 on the "controversial" discussion set of data. Finally, predicting the bill outcomes themselves is achievable with an accuracy of 76% and F1 score of 0.83 given a balanced bill set. The discrepancy between individual vote and bill prediction is likely due to the fact that bill prediction relies on simple majorities which can be achieved in multiple ways, even with some inaccuracies in the underlying member votes.

6.1 Future Work

Future work will expand our model to include voting histories, relationships among legislators and committees, bills or topics, using methods from the literature discussed in related work section. This research could also be applied to other legislative assemblies apart from California where speech text is available, such as New York, Florida, etc.

6.1.1 Explore different Machine Learning algorithms

The features tabulated from the spoken language in this research can be used with different combinations of machine learning algorithms to obtain higher accuracy and prediction. In our experiments the focus was on only three machine learning algorithms: Support Vector Machines, Random Forests and Tensorflow(Neural Network).

6.1.2 External influence evaluation

The bill/vote outcome can be influenced by a number of various factors. The exploration of the effect of lobbyists on the bill result, the effect of profile of members in a committee and the previous history of failed bills will be really interesting to explore and will certainly help in improving the prediction modeling.

6.1.3 Member speech relation

The one thing which we really believe might give a significant boost to the prediction model will be, to find a correlation between the speech same member makes on different bills also speech different members make on same bill. This would really help the researcher in understanding the psyche of the committee member.

6.1.4 Host a Web service on Digital Democracy

Once this research is mature enough, we can create a Web service and host it on the Digital Democracy Web site. This would definitely help and give insights into the voting behavior of members of a committee in the legislature.

BIBLIOGRAPHY

[1] P. A.Tumasjan, T.Sprenger and I.Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. *International AAAI Conference on Weblogs and Social Media*, 2010.

[2] B. Bojduj. Extraction of causal-association networks from unstructured text data. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2009.

[3] B.Pang and L.Lee. *Opinion mining and sentiment analysis*. Foundations and Trends in Information Retrieval, 2nd edition, 2007.

[4] S. Calabrese. Nonnegative matrix factorization and document classification. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2015.

[5] C. Cushing. Detecting netflix service outages through analysis of twitter posts. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2012.

[6] D.Agarwal and B.Chen. flda: Matrix factorization through latent dirichlet allocation. *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, 2010.

[7] S. Gerrish and D. Blei. The ideal point topic model: Predicting legislative roll calls from text. Technical report, Princeton University, 2010.

[8] G.Mishne and N.Glance. Predicting movie sales from blogger sentiment. *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, 2006.

[9] D. Goldblatt and T. O'Neil. How a bill becomes a law - predicting votes from legislation text. Technical report, Stanford University, 2012.

[10] M. B. Grap. A hybrid approach to general information extraction. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2015.

[11] J. Herlocker. Evaluating collaborative filtering recommender systems. Master's thesis, Oregon State University, 2004.

[12] S. J.Clinton and D.Rivers. The statistical analysis of roll call data. *American Political Science Review*, 2004.

[13] J.Heckman and M.Snyder. Linear probability models of the demand for attributes with an emperical application to estimating the preferences of legislators. *National bureau of economic research*, 1996.

[14] B. P. Matt Thomas and L. Lee. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. *Empirical methods in natural language processing*, 2006.

[15] J. D. McElroy. Automatic document classification in small environments. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2012.

[16] D. Miller. A system for natural language unmarked clausal transformations in text-to-text applications. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2009.

[17] G. D. M.Thelwall, K.Buckley and A.Kappas. *Sentiment in short strength detection informal text*. J. Am. Soc. Inf. Sci. Technol., Sebastopol, CA USA, 61st edition, Dec 2010.

[18] J. Patterson. Parsing of natural language requirements. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2013.

[19] M. Quinn and Radev. An automated method of topic-coding legislative speech over time with application to the 105th - 108th u.s. senate. *In Midwest Political Science Association Meeting*, 2006.

[20] A. N. Richard Socher and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. *Empirical Methods in Natural Language Processing. Ass. for Comp. Linguistics*, October, 2013.

[21] Salakhutdinov and Mnih. Probabilistic matrix factorization. *Advances in Neural Information Processing Systems*, 2008.

[22] S.Gerrish and S.Blei. Predicting legislative roll calls from text. *International Conference on Machine Learning*, 2011.

[23] E. K. Steven Bird and E. Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., Sebastopol, CA USA, 1st edition, 2009.

[24] N. A. Tae Yano and J. D.Wilkerson. Textual predictors of bill survival in congressional committees. *Association for Computational Linguistics*, 2012.

[25] Tauberer and Joshua. Govtrack, Nov. 2012.

[26] Z. T.Honkela and K.Lagus. *Text mining for wellbeing: Selecting stories using semantic and pragmatic features*. Artificial Neural Networks and Machine Learning, Part II, ser. LNCS. Springer, vol. 7553 edition, 2012.

[27] J. T.Wilson and P.Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. *Human Language Technology and Empirical Methods in Natural Language Processing: Association for Computational Linguistics*, 2005.

[28] C. Wang and Lawrence. Joint analysis of time-evolving binary matrices and associated documents. *Advances in Neural Information Processing Systems*, 2010.

[29] C. Wei. Bottom-up ontology creation with a direct instance input interface. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2009.

[30] C. J. Wu. Skewer: Sentiment knowledge extraction with entity recognition. Master's thesis, California Polytechnic State University, San Luis Obispo, California, USA, 2016.

[31] P. C. Zach Cain and K. Gampong. Predicting congressional bill outcomes. Technical report, Stanford University, 2012.