# MACHINE LEARNING ALGORITHMS FOR THE ANALYSIS OF SOCIAL MEDIA AND DETECTION OF MALICIOUS USER GENERATED CONTENT

by

Brian Heredia

A Dissertation Submitted to the Faculty of

The College of Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

Florida Atlantic University

Boca Raton, FL

August 2018

**MACHINE LEARNING FOR THE ANALYSIS OF SOCIAL MEDIA AND DETECTION OF MALICIOUS USER GENERATED CONTENT**

by

Brian Heredia

This dissertation was prepared under the direction of the candidate's dissertation advisor, Dr. Taghi M. Khoshgoftaar, Department of Computer and Electrical Engineering and Computer Science, and has been approved by the members of his supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

SUPERVISORY COMMITTEE:

Taghi M. Khoshgoftaar, Ph.D.
Dissertation Advisor

Mehrdad Nojoumian, Ph.D.

Dingding Wang, Ph.D.

Xingquan Zhu, Ph.D.

Nurgun Erdol, Ph.D.
Chair, Department of Computer and Electrical Engineering and Computer Science

Stella N. Batalama, Ph.D.
Dean, The College of Engineering and Computer Science

Khaled Sobhan, Ph.D.
Interim Dean, Graduate College

July 16, 2018
Date

## ACKNOWLEDGEMENTS

First and foremost, I would like to thank my my graduate advisor, Dr. Taghi M. Khoshgoftaar. His tutelage, encouragement, and experience have been invaluable throughout my tenure in the doctorate program. Additionally, I would like to thank Dr. Mehrdad Nojoumian, Dr. Dingding Wang, and Dr. Xingquan Zhu, for serving on my PhD supervisory committee.

I would also like to thank FAU Data Mining and Machine Learning Laboratory members, Dr. Joseph D. Prusa and Dr. Karl Weiss for their help with the preparation of this dissertation. Moreover, I extend my thanks to all the members of the Florida Atlantic University Data Mining and Machine Learning laboratory. Finally, I would like to thank my parents for their love and support, without them none of this would be possible.

I also gratefully acknowledge partial support by the National Science Foundation, under grant number CNS-1427536. Any opinions, findings, and conclusions or recommendations expressed in this dissertation are those of the author and do not necessarily reflect the views of the National Science Foundation.

## ABSTRACT

| | |
|---|---|
| Author: | Brian Heredia |
| Title: | Machine Learning Algorithms for the Analysis of Social Media and Detection of Malicious User Generated Content |
| Institution: | Florida Atlantic University |
| Dissertation Advisor: | Dr. Taghi M. Khoshgoftaar |
| Degree: | Doctor of Philosophy |
| Year: | 2018 |

One of the defining characteristics of the modern Internet is its massive connectedness, with information and human connection simply a few clicks away. Social media and online retailers have revolutionized how we communicate and purchase goods or services. User generated content on the web, through social media, plays a large role in modern society; Twitter has been in the forefront of political discourse, with politicians choosing it as their platform for disseminating information, while websites like Amazon and Yelp allow users to share their opinions on products via online reviews. The information available through these platforms can provide insight into a host of relevant topics through the process of machine learning. Specifically, this process involves text mining for sentiment analysis, which is an application domain of machine learning involving the extraction of emotion from text.

Unfortunately, there are still those with malicious intent and with the changes to how we communicate and conduct business, comes changes to their malicious practices. Social bots and fake reviews plague the web, providing incorrect information and swaying the opinion of unaware readers. The detection of these false users or posts from reading the text is difficult, if not impossible, for humans. Fortunately,

text mining provides us with methods for the detection of harmful user generated content.

This dissertation expands the current research in sentiment analysis, fake online review detection and election prediction. We examine cross-domain sentiment analysis using tweets and reviews. Novel techniques combining ensemble and feature selection methods are proposed for the domain of online spam review detection. We investigate the ability for the Twitter platform to predict the United States 2016 presidential election. In addition, we determine how social bots influence this prediction.

**MACHINE LEARNING ALGORITHMS FOR THE ANALYSIS OF SOCIAL MEDIA AND DETECTION OF MALICIOUS USER GENERATED CONTENT**

## LIST OF TABLES

## LIST OF FIGURES

## LIST OF EQUATIONS

# CHAPTER 1
# INTRODUCTION

## 1.1 MOTIVATION

Data mining is the process of converting data into knowledge. Machine learning is one such process by which this is accomplished. Machine learning techniques extract patterns within structured or unstructured data and use these patterns to train a model. Text mining is a subfield of machine learning, where the data used is strictly text or derived from text. Text mining is applicable to many current domains, such as Natural Language Processing (NLP) [79], election prediction [57, 124], sentiment analysis [51, 52], spam detection [28, 53], and more. Text mining provides methods for solving many real-world problems and the availability of large volumes of unstructured data due to the Internet only increases the necessity for these approaches. The advent of social media and online services have given rise to many new applications for text mining, such as spam review detection [62, 61], social bot detection [29, 83], and public opinion mining [17, 124].

While text mining includes a large variety of application domains, this dissertation is primarily concerned with text extracted from social media and online reviews. Datasets employed within this dissertation are categorized as either traditional or Big Data, based on volume, velocity, and variety.

### 1.1.1 Text Classification

The advent of the Internet brought with it tremendous boons for our society. With novel methods of communication and information sharing, the Internet transformed

the way we work, study, conduct businesses, build relationships, and more. The modern Internet's defining feature is its massive connectedness; information and human connection are available with a few clicks. Many services are available Online now, such as online shopping, online banking, and online dating. In addition, social media sites, such as Facebook and Twitter, have become extremely popular over the past decade. Unfortunately, there are always those looking to take advantage of others and the Internet serves as a platform for these users as well. Spam and social media bots are some ways these users can influence the actions and beliefs of others. Fortunately, these can be combated through machine learning and text mining. Both Online services and social media provide us with a plethora of new unstructured text data through user product/service reviews and user posts, respectively.

There are three high-level approaches to learning from text, which are supervised, semi-supervised, and unsupervised. Supervised methods require labeled data, while unsupervised techniques do not. Semi-supervised techniques use a combination of supervised and unsupervised methods. This dissertation will focus on supervised learning methods, as our domains of focus all have publicly available labeled data. Typically, text is converted into a matrix-like representation using either a TF-IDF or bag-of-words approach to match the fixed-length inputs and outputs necessary for machine learning techniques. Unfortunately, this causes the number of features to increase based on the number unique words, leading to a high dimensional feature space. Therefore, other features, such as Part Of Speech (POS), Linguistic Inquiry and Word Count (LIWC), and stylometric features, can be useful for text mining [87, 84, 117].

### 1.1.2 Machine Learning for Spam Detection

As previously mentioned, text-based data is involved in many important application domains. One of these application domains is online fake review (spam) detection.

Websites, such as Amazon, have revolutionized the way we do business. From groceries to power tools, Amazon has many products and services available for purchase with a couple of clicks. However, determining whether the product or service is right for you may be difficult. People rely heavily on consumer reviews and recommendations for insight when purchasing products, trying new restaurants, finding a primary care physician, and other goods and services. Google, Yelp, and Amazon are a few of the websites loaded with user reviews on a multitude of topics. Reviews can be created by anyone who has access to the web, leading to one of the main issues with Online reviews – spam. Spam reviews are typically created to harm, or bolster, the reputation of a company or product. Determining if a review is spam, from a human perspective, is often difficult, if not impossible [53, 54]. Human readers are not able to differentiate between words used in spam reviews and words used in truthful reviews. Due to this difficulty, methods for detecting spam reviews have become a point of interest in the last decade.

Many current methods to identify review spam use a supervised machine learning approach [62, 61, 28]. Reviews in our datasets are large documents, with an average of 559 words per review and a standard deviation of 781.5 words. The conversion of unique words into a matrix-like representation creates a high-dimensional feature space. This high dimensionality creates challenges in the form of higher computational costs, useless features, and degraded model performance [47, 27]. To overcome the issue of high dimensionality, feature selection techniques can be applied to limit the feature space [27, 84]. However, these techniques do not show an increase in performance, while some even show a decrease in model performance [27]. Spam review datasets suffer from class imbalance, where instances of one class outnumber instances of the other [115]. This issue skews model performance in favor of the majority class [93]. The research into class imbalance and spam reviews is limited.

### 1.1.3 Machine Learning with Twitter

Social media sites have allowed people to communicate and share their lives with whomever they choose across the world. Twitter allows users to connect to one another by "following" accounts and users to generate short 280 character micro-blog posts. Twitters easy-to-use platform has helped it grow a large user base consisting of the news media, celebrities, politicians, and your average person. Twitter also provides an easily accessible API for data collection. The content and network generated by user connections provide an invaluable data source.

*Sentiment Analysis*

Sentiment is defined as the emotional polarity of a document; in its simplest form, sentiment defines whether a document is positive or negative. Sentiment analysis is considered a binary classification problem when working with positive and negative sentiments. It can be further divided into a multi-classification problem by adding different emotion classes. In this dissertation, we focus on sentiment as a binary classification task using tweets and reviews. As tweets generally carry the authors opinion, sentiment analysis can be applied to determine public interest in any subject. Sentiment analysis is the process of analyzing a document and labeling the document with a certain emotion conveyed by the text [89, 88]. Reviews usually contain a rating along with text describing the product or service. The rating can be directly converted into a sentiment label by setting a threshold.

Sentiment analysis in text is not a new field within machine learning. Twitter has been used to predict important events such as general elections [17, 124, 58, 55, 57] and movie box office performance [82]. Sentiment analysis suffers from the same shortcomings as spam review detection. The feature space associated with sentiment analysis with tweets is still considered high-dimensional. Although this feature space may be smaller than reviews, due to the length of the instances. The approach to

deal with high dimensionality is the same, feature selection techniques have improved sentiment analysis with tweets [103]. Sentiment analysis with tweets does not typically suffer from class imbalance, as there are a large number of tweets available at any time. However, research has been conducted in improving classification performance in the case of imbalanced tweet sentiment data [104].

*Social Bot Detection*

Although most social media accounts are created for harmless use, there are automated or bot accounts that are created for malicious intent. These social bots tend to behave differently from human controlled accounts on Twitter; having a larger number of retweets and follower-networks consisting of mostly bot connections [29]. Social bots have been active since the onset of social media; however, research into social bot detection has only recently become popular due to the 2016 United States presidential election. Their detection is paramount as they can alter public opinion and sway neural party opinions with false information.

Social bot detection methods can be trained using a varying number of features types. To detect bot accounts, studies have used warped correlations between accounts [22], time between tweets, text patterns, and account properties [24], and, a combination of network, user, friends, temporal, content, and sentiment features [29, 127]. The Botometer (formerly BotorNot) system is the most popular social bot detection platform for Twitter [29]. This model uses a Random Forest classifier trained on a variety of features. The authors claim the system generates more than 1,000 features across six categories from data available through the Twitter API.

## 1.2 CONTRIBUTIONS

The contributions of this work focus on the application of machine learning techniques for the analysis of social media and detection of malicious user generated content. Ap-

plying machine learning techniques to any domain require a number of pre-processing steps, training the machine learning models and testing the model on new data [134]. This dissertation explores all steps in this process for both the analysis of social media content and detection of malicious user generated content in full detail. This dissertation makes the following contributions:

- Explore the effects of ensemble methods and novel methods combining ensemble and feature selection techniques on the detection of online review spam.
- Determine the performance of cross-domain sentiment classification tweets and reviews.
- Determine the effectiveness of Twitter at polling and predicting the United States 2016 presidential election.
- Explore the influence social bots have on expressed public opinion of candidates on Twitter during the United States 2016 presidential election and their effects on Twitter as a polling source.

## 1.3 DISSERTATION STRUCTURE

This dissertation is organized as follows. Chapter 2 introduces the general methodology and theory for data prepossessing and performing text classification using machine learning techniques and neural networks. Chapter 4 demonstrates the effectiveness of ensemble methods and the combination of ensemble techniques and feature selection at detecting online review spam. Chapter 3 shows the ability of tweets and reviews to be used for cross-domain sentiment analysis. In Chapter 5, Twitter is explored as a data source for polling and predicting the 2016 United States presidential election. In addition, Chapter 5 also determines the influence social bots had on Twitter public opinion and on the effectiveness using Twitter as a polling source. Finally, we present our conclusions and future work in Chapter 6.

# CHAPTER 2
# THEORY AND METHODOLOGY

This chapter describes the general methodology for data collection, data processing, machine learning techniques, and text classification using neural networks. As neural networks are highly variable in architecture, details describing individual neural network layers are presented instead of full architectural descriptions. Methods to enhance machine learning technique performance such as feature selection and ensemble techniques are also described.

## 2.1 TEXT DATA RETRIEVAL AND PROCESSING

For the purpose of this dissertation, text content was extracted from five sources across two domains. Online reviews, collected from Amazon, Hotels, Restaurants, and Doctors, and tweets collected from Twitter make up our sentiment datasets. All datasets are converted into single instance per line comma separated values (CSV) files. These CSV files consist of extracted text and the sentiment label classified as 0 (negative) or 1 (positive). In the case of the election dataset, username and location are also included in the CSV file. Files were encoded into UTF-8 [133] to include most character representations.

### 2.1.1 Tweet Sentiment Data

The first tweet dataset employed is the sentiment140 corpus [41]. The sentiment140 corpus is used to construct training and testing sets for sentiment analysis in tweets. This corpus consists of tweets extracted from Twitter based on the presence of emoticon(s) in the text. Emoticons are some combination of symbols or characters used to

express emotion or represent an image, i.e. ":-)" and ":-(". Tweets were extracted using Twitter's REST API and labeled based on the emoticons present. Tweets were labeled positive or negative based on the presence of any of the eight emoticons chosen by the authors. Once instances were labeled either positive or negative, emoticons were removed from the text. The removal of emoticons is necessary as they provide a direct relation to the class label and would skew machine learning technique performance.The final dataset consists of 1.6 million tweets, with 800,000 positive and 800,000 negative instances. This dataset has no class imbalance, as it consists of a perfect 50:50 split between classes. This method of collecting and labeling tweets allows for the construction of large text sentiment datasets that would otherwise not be possible through manual labeling. Unfortunately there may be some noise due to the labeling process, as the inclusion of multiple emoticons in a single tweet may lead to incorrect labels.

### 2.1.2 Tweet Election Data

The second tweet based dataset is the collection of tweets related to the 2016 United States presidential election. This dataset consists of tweets collected from September 22nd to November 8th, 2016 using the Twitter REST API. No restrictions on location or account was implemented; however, query terms were limited to topics related to both the Republican and Democratic candidates. The following terms were used in our search queries: "Trump", "Clinton", "President Trump", "President Clinton", "#trump2016", "#clinton2016".Tweet text, username, query, and location were extracted into a CSV file. The dataset consisted of 2,942,808 tweets from 705,381 unique user accounts. No pre-processing was performed on the dataset, meaning retweets and duplicate tweets were included in the dataset. It should be noted that this dataset has no sentiment labels at collection, sentiment is labeled and added during post-processing.

### 2.1.3 SemEval

The final tweet-based dataset is the 2014 semEval dataset [110]. The semEval dataset is a significantly smaller collection of tweets when compared to the Sentiment140 corpus or the Election data. Unlike the either tweet datasets, the semEval dataset was labeled manually; thus, it is expected to have a smaller percentage of misclassified instances. The original dataset contains positive, negative, and neutral class labels. To match the binary classification format the sentiment140 corpus has, the neutral instances were removed from the semEval dataset. The altered dataset used in this dissertation consists of 3,420 positive instances and 1,399 negative instances. The class distribution is imbalanced in this dataset, as there are approximately two times as many positive instances as negative.

### 2.1.4 Amazon

The first of the online review datasets consists of Amazon Product reviews collected by Mcauley et al. in 2015 [80, 48]. This dataset consists of 148.2 million Amazon reviews, which is further divided into several datasets by filtering specific subsets of the data and binning based on product category. Of these datasets, this dissertation employs reviews from the aggressively deduplicated data, which includes only unique reviews. This subset of the dataset removed over one-third of the reviews, leaving 82.83 million reviews. This subset of the data is chosen as the presence of a duplicate review is a strong indication of spam [28]. In addition, the removal of one-third of the instances matches claims made in 2014 that over 30% of online reviews are fake [42]. This dataset is mainly used as a source of sentiment, thus the identification of fake reviews still present within the dataset is left for future work as it should not greatly impact sentiment classification [30].

In the case studies presented in this dissertation, the Amazon Product Review dataset is converted into an Amazon Sentiment dataset. The Amazon Sentiment

dataset consists of one million Amazon product reviews randomly sampled from the original deduplicated data. The Amazon Sentiment dataset contains an equal distribution of positive and negative reviews, labeled automatically based on review score. The product reviews contain a review score metric indicating whether the author liked or disliked the product. This metric is measured from one to five stars. A score of one star indicates that author did not enjoy the product while conversely a score of five indicates the author very much enjoyed the product. To automatically label the reviews with sentiment, a threshold of 2.5 stars was used. If a review held a review score of 2.5 or lower, the review was classified as negative sentiment, while if the review score was 2.5 or higher, it was classified as positive sentiment. A binary classification system was used to match the system used in the sentiment140 dataset. This same process could easily be applied to create a more variable class system by varying the threshold.

### 2.1.5 ReviewSent

The final sentiment dataset consists of online reviews created by Li et al. [73]. This dataset, referred to as ReviewSent throughout the remainder of this dissertation, contains spam reviews from three different domains: doctors, hotels, and restaurants. Reviews are generated from two sources: Amazon Mechanical Turk (AMT) and expert reviews. This dataset is significantly smaller than the previously described datasets, with only 2,836 instances. The data set contains the text found in the review, the rating given, the domain the review belongs to, and the class label.

AMT [21] is a service provided by Amazon which gives customers access to an on-demand, scalable workforce. This workforce was employed to create fake reviews using real reviews as a baseline. As AMT uses regular workers, whom may have no knowledge of domain keywords, the word distributions found in the reviews are significantly different from word distributions found in real reviews [84]. These syn-

thetic reviews are not always representative of real-world spam data; to counteract this, domain experts were hired to create false reviews for their specific domains. For example, an employee in a doctor's office was asked to write false reviews for their practice. Li et al. [73] posited employees would have a better grasp on daily processes which would lead to a better synthetic spam reviews.

## 2.2 TEXT MINING AND NLP

Text Mining generally relies on traditional NLP and machine learning techniques to generate an acceptable feature space. Bag-of-words and TF-IDF are the most popular for converting raw text into acceptable machine learning features. Once transformed into the appropriate feature space, machine learning classifiers can be trained. In this section, a brief overview of the traditional text-based feature extraction techniques and machine learning classifiers used in this dissertation are given.

### 2.2.1 Feature Engineering and Extraction

The initial task for performing any form of text mining with machine learning classifiers is feature engineering and extraction. Feature engineering is the process of defining a useful feature space based on domain knowledge. Many methods may be employed in the feature engineering process. Features may be extracted directly from the text using advanced NLP methods such as Part-of-Speech (POS) tags, lexical mapping, or syntactic features. POS tagging refers to marking words in a document based on their corresponding part-of-speech, such as verbs, nouns, adjectives, etc. Lexical features focus on word or character use, while syntactic features represent the style of the writer, such as their use of the words "the", "of", "a", etc. Although these complex features increase the feature space and may allow for a better representation of the text, the integration of these more complicated features has shown inconsistent benefits [8, 41, 70, 112]. Regardless of whether more complex feature engineering pro-

cesses are performed, many machine learning techniques require numeric input; thus, a transformation of raw text into an appropriate feature vector for machine learning classifiers must be performed. There are two popular methods for completing this task: Bag-of-words (BoW) and TF-IDF.

The most common of the two aforementioned techniques is the BoW approach. In its simplest form, BoW converts raw text into a feature vector where words are represented as features. BoW uses an n-gram based feature engineering method, where n determines the number of words used to create a new feature. BoW can be performed using either unigram (n=1), bigram (n=2), or trigrams (n=3). When using unigrams, every unique word is converted into a feature and these features consist of either '0' or '1' based on the presence of the word in a document. Bigrams and trigrams use the same process; however, instead of representing every individual word as a feature, bigrams use pairs of consecutive words, while trigrams use sequences of three words. Although simple, the BoW approach is considered highly effective. In addition, the feature space generated by BoW can be easily combined with more advanced feature engineering methods [64].

Similar to the BoW approach, Term Frequency-Inverse Document Frequency (TF-IDF) creates a feature vector using unique words [113]. The difference between these techniques lies in the vector values. Instead of populating the vectors with '0' or '1' based on the presence of a word in the document, TF-IDF populates the vector with a word weight. Weights are calculated based on a terms frequency within a document, and inversely weighted by a term's frequency throughout the dataset. For every word in a document, $tf$ can be calculated by $tf = 1 + log_{10}(f)$, where $f$ is the number of times the term appears in a document. For a collection of documents, $idf$ is calculated as $idf = log_{10}(N/df)$, where $N$ is the number of documents in the collection and $df$ is the number of documents the word occurs in. TF-IDF is the product of $tf$ and $idf$ and is commonly used in information retrieval.

### 2.2.2 Machine Learning Techniques

We employ a total of five basic machine learning algorithms to train classifiers across our case studies. Many of these algorithms are explored with the addition of feature selection and within an ensemble framework.

*Naïve Bayes*

Naïve Bayes methods are a family of supervised learners that build classifiers using Bayes' theorem on conditional probabilities [109]. Naïve Bayes methods make the "naïve" assumption that individual features are not influenced by other features. Although, in general, this feature independence is not the case, Naïve Bayes still offers good performance and we consider it a good baseline learner for comparisons [132, 136, 108]. Gaussian Naïve Bayes, also referred to as Naïve Bayes, uses Bayes' theorem to approximate the posterior probability of an instance belonging to a class based on its feature values [132, 136]. Naïve Bayes makes the assumption of a gaussian distribution in the feature space. The formula for Naïve Bayes is found below:

$$\hat{y} = p(C_k) \prod_{i=1}^{n} p(x_i|C_k) \tag{2.1}$$

where $\hat{y}$ is the predicted class, $p(C_k)$ is the probability of the instance belonging to class k, and $\prod_{i=1}^{n} p(x_i|C_k)$ is the product of all conditional probabilities for the features associated with the instance.

Multinomial Naïve Bayes (MNB) is a variant of the Naïve Bayes learner [81, 69]. The main difference between NB and MNB is the way in which the probabilities are calculated. Naïve Bayes uses conditional probabilities of each feature to help determine classification status. In Multinomial Naïve Bayes for text classification, the instance (a document in this example) is assigned to the class which has the

highest conditional probability of $P(C|X)$. To calculate this probability, a count is done of the words which overlap between the document and the class, and then the count is divided by the total number of words. If $P(C_1|X) > P(C_2|X)$ then the document is classified as C1, otherwise it is classified as C2. Simply put, NB assumes a gaussian distribution of features, while MNB assumes a discrete feature space.

*Support Vector Machines*

Support Vector Machine [60, 16] constructs a hyper-plane that divides the instances into groups based on their feature space. The hyper-plane that creates the maximum separation between instances belonging to different classes is considered the best class discriminator. New instances are assigned a class based on which side of the hyper-plane they fall into. The hyper-plane is defined by the support vectors, which are instances closest to the line.

The data may be transformed via a kernel function into linearly separable spaces [16]. The transformations allow for non-linear boundaries to be formed around the data by transforming the non-linear data into a higher level representation. Figure 2.1 illustrates hyper plane construction for a 2-dimensional linear SVM classifier.

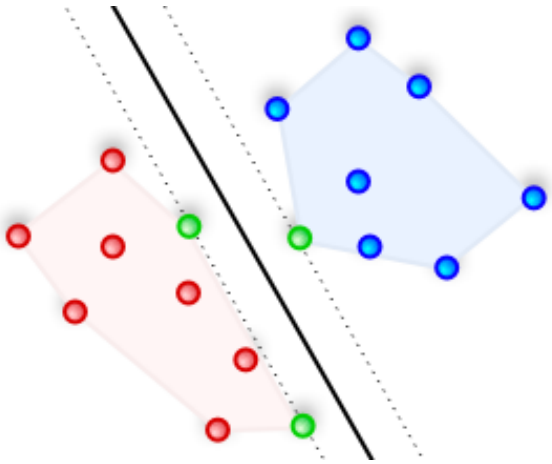


Figure 2.1: Visualization of Linear SVM, margins and support vectors.

*Logistic Regression*

Logistic regression [65] seeks to find a linear relationship between the features and the class label. This process is different from linear regression as it is used for the task of classification. Logistic regression aims to create a probability function that uses features as inputs and returns the probability of that instance belonging to a class as an output. When used for classification, class membership is determined by assigning a class that minimizes expected cost of classification [66]. The formula for logistic regression can be found below:

$$\log \frac{P}{1-P} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n \tag{2.2}$$

where $\frac{P}{1-P}$ is the odds ratio, $X_i$ is the value for that feature, and $\beta_i$ is the coefficient associated with feature $X_i$.

*C4.5 Decision Tree*

The final learner, C4.5 decision tree, creates a tree based on the features that discriminate the most between classes. The decision tree process uses information entropy to determine a decrease in entropy when examining a certain feature. The tree splits at the feature that minimizes entropy and maximizes information for a class, meaning the more discriminant features will be found towards the top of the tree [106, 105]. The final classification is found in the leaves of the tree. We use two versions of C4.5 created with j48, an open source Java implementation [45]. The first used default parameters, while the second includes Laplace smoothing and no pruning which have been shown to improve performance [130].

## 2.3 TECHNIQUES TO IMPROVE CLASSIFIER PERFORMANCE

Text mining suffers from high dimensionality due to the intrinsic nature of converting text into an acceptable feature space. In addition, class imbalance provides a separate challenge for classifier performance. This section will provide a brief overview of methods used to improve classifier performance.

### 2.3.1 Feature selection

Feature selection can be performed using wrapper-based, filter-based, embedded, or hybrid techniques. Wrapper-based techniques use a classifier to rank the performance of a subset of features. Wrapper-based techniques do not scale well with a high feature space, due to their computationally expensive approach. Filter-based techniques can be categorized into two types: filter-based subset evaluation and filter-based ranking. Filter-based subset evaluation techniques attempt to find the subset of features which maximizes classification performance. Usually, this process uses a greedy approach, where a feature set is created by adding the initial best feature that discriminates between the classes (or starting with all features and removing the feature which discriminates the least between the classes). Features are then added to the subset individually and tested for performance, the feature with the highest increase in performance is added to the subset. This process is repeated until the best subset of features is found. Similar to wrapper-based techniques, filter-based subset evaluation does not scale well with very high dimensional data as it is computationally expensive. Unlike the previous two approaches, feature rankers provide the type of fast and scalable feature selection suitable for very large feature spaces [49, 50]. They use various measures to rank features based on performance. Below we describe the 10 distinct feature rankers employed in our case studies.

*Chi-Squared*

Chi-Squared utilizes the Chi-Squared statistic to measure the relationship between a feature and the class. This feature ranker measures the independence of two events, in this case the events are the occurrence of the feature and the occurrence of a specific class. If we consider the occurrence of a feature as A and the occurrence of a class as B, then these occurrences are independent if $P(AB) = P(A) * P(B)$. The formula below shows the common form of the Chi-Squared ranker:

$$\chi^2 = \sum_{k=1}^{n} \frac{(O_k - E_k)^2}{E_k} \tag{2.3}$$

where O is the observed value and E is the expected value. Thus, a higher Chi-Squared value indicates some form of dependence between the feature and the class, whether positive or negative.

*Mutual Information*

Mutual Information determines the importance of a feature by measuring the contribution of the presence, or absence, of a feature towards a correct classification [91]. The formula below describes mutual information:

$$I(W, C) = \sum_{e_W \in [0,1]} \sum_{e_C \in [0,1]} p(e_W, e_C) log \frac{p(e_W, e_C)}{p(e_W)p(e_C)} \tag{2.4}$$

where W is the word and C is the class. This formula can be simplified to $I(X, Y) = H(Y) - H(Y|X)$ or $I(X, Y) = H(X) - H(X|Y)$, where $H(X)$ and $H(Y)$ are marginal entropy values and $H(X|Y)$ and $H(Y|X)$ are conditional entropy values. $H(X|Y)$ and $H(Y|X)$ are values that indicate the amount of uncertainty left after $H(X)$ or $H(Y)$ are learned.

*PRC*

The PRC metric measures the area under the curve of Recall on the x-axis and Precision on the y-axis. Recall is defined as the True Positive Rate (TPR) and Precision is defined as the fraction of instances classified as positive that are actually positive.

*ROC*

The ROC metric measures the area under the curve of False Positive Rate (FPR) on the x-axis and TPR on the y-axis. This directly indicates model performance, where the higher the area under the ROC curve the better the model performance across all thresholds.

*Probability Ratio*

Probability Ratio was used for text classification in [38] and was defined as the probability of the feature given the positive class divided by the sample estimate probability of the feature given the negative class. The formula ($TPR/FPR$), describes the probability ratio between the positive and negative classes.

*Gini-Index*

The Gini-Index was originally utilized for measuring income over a population. The Gini-Index has a range from 0 to 1, which indicates the following: GI=0 implies the income is split evenly across the population, and GI=1 means that a single person receives all the income. The feature ranker version of the Gini-Index analyzes the distribution of a feature across the classes.

*Kolmogorov-Smirnov Statistic*

The KS statistic is a non-parametric test that measures the difference in the cumulative distribution of two data sets.

*Significance Analysis of Microarrays (SAM)*

SAM was originally created for detecting significance in microarrays within the bioinformatics domain. SAM compares the observed and expected values of the parameter to identify features whose associated values differ by an amount of statistical significance among the sets [125].

*Signal-to-Noise*

The Signal-to-Noise ratio represents the ratio of signal information to noisy background information. This process can be used to determine how well a feature discriminates between two classes. The formula for S2N is:

$$S2N = \frac{(\mu_P - \mu_N)}{(\sigma_P + \sigma_N)} \tag{2.5}$$

where $\mu_P$ is the mean of the positive class, $\mu_N$ is the mean of the negative class, $\sigma_P$ is the standard deviation of the positive class, and $\sigma_N$ is the standard deviation of the negative class.

*Wilcoxon Rank Sum*

The Wilcoxon Rank Sum is a variation of the standard t-statistic. In a standard t-statistic, the distribution is assumed to be normal, while in the WRS no assumption on the distribution is made. There are two steps to be performed before defining the WRS: 1. all instances are ranked based on the value of the feature, and 2. all the

rankings in the positive class are summed as $W_p$. The formula for WRS is:

$$WRS = \frac{(W_p - \frac{n_p(n_p+1)}{2}) - \frac{n_p n_n}{2}}{\sqrt{\frac{n_p n_n (n_p+n_n+1)}{12}}} \tag{2.6}$$

### 2.3.2 Ensemble Learners

Ensemble techniques take advantage of multiple diverse classifiers to create a high performing classifier that is more robust [100]. In this dissertation, we employ five distinct ensemble classifiers.

*Boosting*

Boosting takes an iterative approach; training and evaluating a model using a classifier, then training and evaluating subsequent classifiers based on the results of the previous classification [39]. Instances are assigned a weight at the start of the boosting process and the weight is updated based on classification performance of a model. Instances that are misclassified are weighted higher than correctly classified instances and the model is re-trained. This process is repeated a set number of times and a set number of models are created. The final model is chosen based on majority voting across all models created. Due to the re-weighting step, Boosting is not compatible with certain base learners; however, a separate approach, where the instances are re-sampled from the original data set according to the weights, allows these base learners to be compatible with Boosting. This data sampling approach is applied to re-select instances in accordance with their assigned weights.

*Bagging*

Bagging, also known as bootstrap aggregating, uses data sampling with replacement (re-sampling from the original data set where the same instances can be re-selected)

to create a predefined number of bootstrap data sets. These new bootstrap data sets are the same size as the original and contain instances from the original data set. The creation of multiple datasets increases the variance between models as instances are sampled with replacement. The bootstrap data sets are then used to train a single base classifier each. Once all the classifiers have been trained, an aggregation technique is used to determine the final classification; typically majority voting is used. Bagging has been shown to increase performance of weak classifiers but adversely affect the performance of stable learners [18].

*Random Forest*

Random forest constructs an ensemble of decision trees in an effort to avoid overfitting and improve classification performance by averaging many deep decision trees trained on different subsets and feature subspaces of the training data [74]. Random Forest divides the data in two ways, bootstrap aggregation (bagging) and random subspace sampling. Bagging randomly samples instances from the training data with replacement (duplicates are allows). Trees are than trained on different data bootstraps to generalize a diverse forest who's decisions can be averaged together. Random subspace sampling refers to the bagging of features instances. Each tree in Random Forest is trained with a random subset of features, further enhancing diversity between trees. Random Forest has been shown to perform well on imbalanced data [68, 126].

*Select-Boosting*

Select-Boost is a modified version of the AdaBoost.M1 algorithm developed by our research team. The Select-Boost [103] framework embeds feature selection within the Boosting ensemble framework. Initially, all samples in the training data are assigned equal weights. The data is then sampled from the original data set. Feature selection

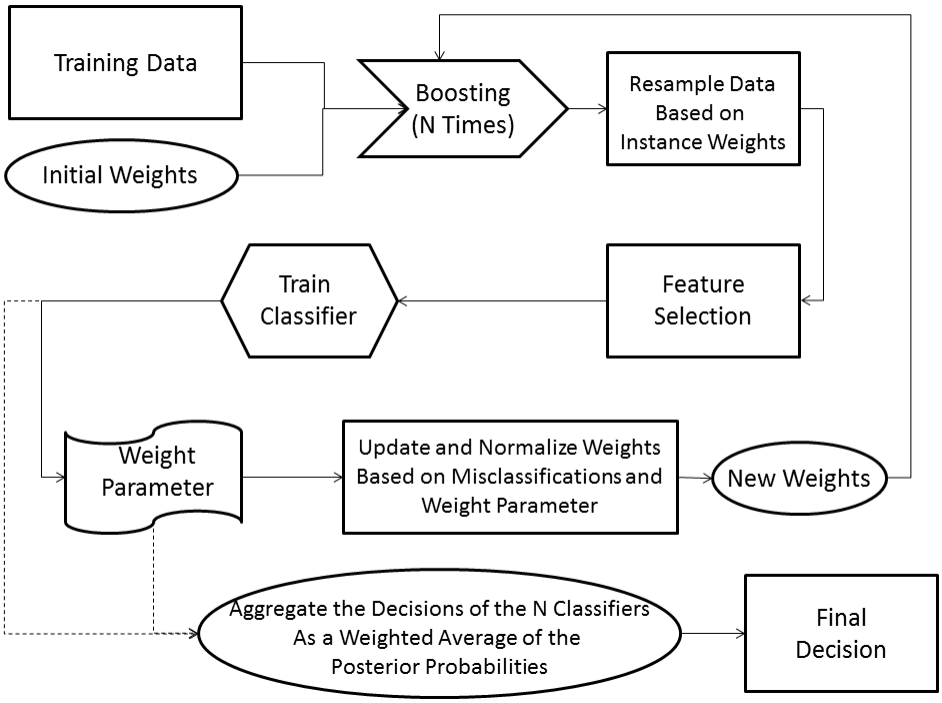



Figure 2.2: Select-Boosting Framework

is applied and classifiers are trained on the reduced feature space. The weights are then updated based on misclassified instances and the Select-Boost algorithm begins anew. As with Boosting, Select-Boost is also incompatible with certain base learners due to the re-weighting step. To mitigate this, instances can be re-sample from the original data set according to the weights assigned and create a new training data set where the probability of selecting an instance is equal to its weight. Figure 2.2 depicts the Select-Boosting framework. The dotted line represents the path the framework takes once all iterations have finished.

*Select-Bagging*

Select-Bagging [103] adds a feature selection step to the Bagging algorithm. The Select-Bagging framework first creates a predefined number of bootstrap data sets from the original through sampling with replacement. Feature Selection is applied to every bootstrap data set and then each data set is used to train a classifier. The

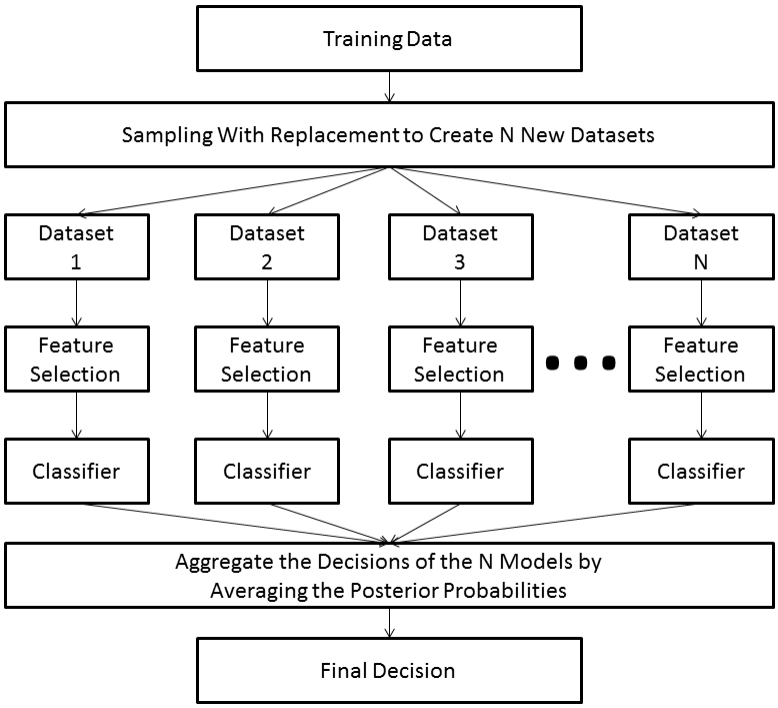



Figure 2.3: Select-Bagging Framework

feature space across datasets will vary as instances will not be the same in each dataset, increasing the variance explained through the final model. The results of those trained classifiers are aggregated and a final classification is made through majority voting. Figure 2.3 shows the Select-Bagging framework.

## 2.4 ARTIFICIAL NEURAL NETWORKS

In addition to traditional machine learning techniques, this dissertation employs deep learning methods. Deep learning refers to multi-layered neural networks made up of artificial neurons [35, 72, 85]. Artificial neurons are based on the biological neuron, with functions that correspond to biological processes. An input vector of values mimics dendrites, the summation function mimics the neuron's soma, and finally the activation function mimics the neuron firing neurotransmitters.

The input vector is the size of the feature space associated with the data. This feature space is multiplied by a weight vector of the same size. The resulting vector

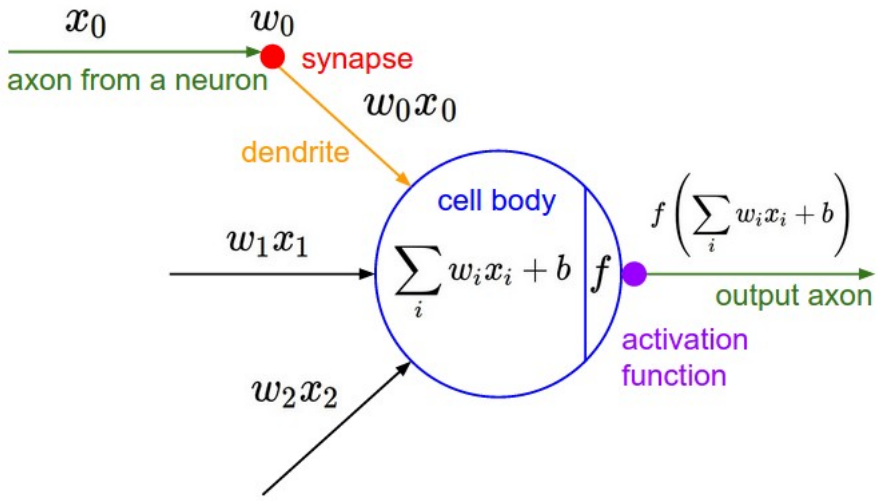



Figure 2.4: Neuron Model

is summed and passed to an activation function to determine the neuronal output. Equation 2.7 formally describes the full process within a neuron.

$$output_k = f(x)_{activation}\left(\sum_{i=1}^{n} w_{ki} x_i\right) \tag{2.7}$$

Neural networks excel in the detection of non-linear patterns [114]. This is due to the arrangement of neurons in layers and the application of non-linear activation functions. Many different neural network layers exist and neural network architectures vary largely between their application domains. In this dissertation, we utilize a Deep Convolutional Neural Network (ConvNet). These ConvNets are feed forward artificial neural networks based on the human visual cortex. In this section, layers used to create a ConvNet are explained.

### 2.4.1 Convolutional Layers

In a convolutional layer, neurons connect to a small region of the current input volume and apply a linear filter across the input space. The linear filter output is passed to a non-linear activation function. This results in the formation of a set of $k$ feature maps,

$\{\mathbf{h}^k, k = 0..K\}$, based on the number of randomly initialized filters applied. For an input vector $\mathbf{x}_{ij}$, the output of a filter $\mathbf{h}^k$, with weight $\mathbf{W}^k$, bias $\mathbf{b}_k$ and activation function $\phi$ is defined as:

$$\mathbf{h}_{ij}^k = \phi(\mathbf{W}^k \mathbf{x}_{ij} + \mathbf{b}_k) \tag{2.8}$$

The output of a convolutional layer is a learned feature map for a highly localized section of the input vector. Stacking convolutional layers allows the ConvNet to learn higher-level features that may better represent the full feature space. For this reason, ConvNets are unparalleled in application domains such as voice recognition and computer vision.

### 2.4.2 Pooling Layers

In addition to convolutional layers, convolutional networks usually employ pooling layers. These layers are a type of downsampling used to reduce the size of a convolutional layer's output, thus reducing the amount of parameters and computation required. The most popular of these methods is called max pooling. Max pooling takes a rectangular box with predefined dimensions and moves it across the output feature space taking the maximal value within each section. Figure 2.5 depicts this process.

It should be noted that if the pools do not overlap, valuable information regarding position may be lost. Instead pools should overlap to preserve the position of features through a process called "coarse coding" [59].

### 2.4.3 Dropout Layers

Dropout layers are employed to avoid overfitting [120]. A matrix matching the number of neurons is created with values of either 0 or 1. The number of 0's and 1's is

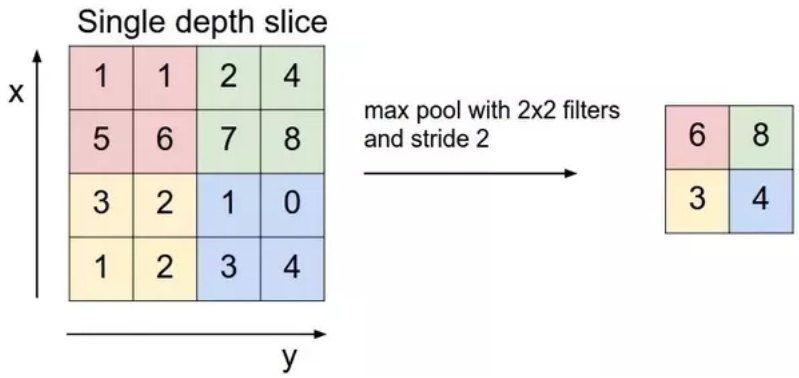



Figure 2.5: Max Pooling Process

controlled via the dropout fraction parameter. The output of the previous layer is multiplied against this matrix and the result is passed to the next layer of the neural network, essentially shutting off a fraction of neurons.

### 2.4.4 Fully Connected Layers

Fully connected layers, as their name implies, consist of neurons that are fully connected to every neuron within the layer and every neuron in the neighboring layers. These layers consider every potential relationship between the inputs. For every neuron, values from all connected neurons are aggregated and passed through a activation function. Fully connected layers are typically found towards the end of a ConvNet. When working with convolutional layers, the signal needs to be flattened into a 1-dimensional vector before being fed into a fully connected layer.

### 2.4.5 Activation Functions

Activation functions vary widely between layers and networks. However, all activation functions are non-linear differentiable functions. Below we present three common activation functions: the sigmoid/logistic function (Equation 2.9), hyperbolic tangent (Equation 2.10), and Rectified Linear Unit (Equation 2.11). Of these, ReLU is considered the best activation function and the most commonly used, while softmax is

used for classification as it outputs a probability.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.9}$$

$$f(x) = tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.10}$$

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases} \tag{2.11}$$

# CHAPTER 3
# INCORPORATING MULTIPLE DATA SOURCES FOR SENTIMENT DETECTION

## 3.1 BACKGROUND AND MOTIVATION

Sentiment analysis occurs in every human interaction. We use it when gauging a reaction to a response or determining the emotional status of one another. However, sentiment is not limited to human interaction as text can also relay the sentiment of the author. Thus, mining text for sentiment can provide additional insight into the opinions of the author and provide important information regarding public views. Sentiment analysis in text is not a new domain, countless tweets and reviews have been analyzed for sentiment and used to predict important events such as the general election [129] and movie box office performance [82]. Tweets are micro-blog posts generated by users on Twitter [2]. Users express their views and ideas through tweets, making them ripe for sentiment analysis. Reviews are an assessment of a certain entity, whether it be a product, person, or location, and can be found on websites such as Yelp and Amazon. Reviews are, by nature, opinionated, making them a good candidate for sentiment analysis. Both tweets and reviews have been extensively used in sentiment analysis [89, 97, 94]; however, their participation in cross-domain sentiment classification is relatively very limited.

From a machine learning perspective, determining sentiment in text requires previously labeled data, from which patterns are learned and used to identify sentiment in future documents. The process of using previously labeled data to train a classifier is known as supervised learning. The majority of supervised learning approaches for

sentiment analysis use words found in the text as features describing the review and sentiment as the class label. Other features such as syntactic features, lexical features, and frequency of words can also be included at the expense of computational resources. Sentiment labels vary from basic emotional polarity to complex emotions such as surprise. A classifier probes the data to learn patterns in the text, the classifier then predicts sentiment of new instances based on their features. The performance of the classifier is directly affected by the training data used. Thus, data used to train a classifier is usually from the same domain as the data used to test it (i.e. tweets are used to train a classifier which predicts sentiment of tweets). However, it is possible to use data from a related domain to train a classifier to classify instances from the domain of interest (i.e. use tweets to train the classifier which predicts sentiment of reviews). This process is known as cross-domain sentiment analysis.

Cross-domain classification is appealing due to the large amount of data that exist in certain domains and/or the ease of obtaining certain types of data. Moreover, using training data from related domains can also be used to train a classifier that can be applied to multiple related domains. Both tweets and a wide range of review types are readily available for online collection and large datasets exist for these domains. On average, 6,000 tweets are generated per second on Twitter [2] and millions of reviews can be found on Amazon alone, with many other websites existing that also provide online reviews. With labeling methods, such as distant supervised learning [41] (sentiment labels are automatically generated from emoticons), there is no lack of available labeled tweet data. Additionally, reviews can readily be labeled using the rating given by a user. Thus, the current motivation for cross-domain or multi-domain classification is the creation of an efficient general classifier that can be applied to multiple domains, including those with limited labeled data.

This chapter makes the following contributions:

- We determine the performance of cross-domain sentiment classification when

training a model with reviews or tweets.

- Determine the effectiveness of training a classifier on a combination of both tweets and reviews.
- We determine the appropriate dataset size necessary for combined sentiment detection.
- Statistical analyses, in the form of ANOVA and Tukeys HSD, are used to verify our results.

The remainder of this chapter is organized as follows. Section 3.2 presents previous works related to sentiment analysis using tweets and reviews. Section 3.3 presents our methodology, including dataset information, classifiers, experimental setup, and performance metric. Section 3.4 presents our results, discussion, and statistical analysis for the cross domain experiments. In Section 3.5 the results and discussion for integrating multiple data sources is presented. Finally, Section 3.6 presents our chapter summary.

## 3.2 RELATED WORK

The sentiment140 dataset used in our experiments was created using an automated sentiment labeling method [41]. Go et al. [41] created an automated labeling method which took advantage of emoticons found in tweets. Emoticons are a combination of symbols that express an emotion, usually forming a facial representation, such as ":)" which depicts a positive emotion. Tweets were collected and labeled based on the emotion assigned to each emoticon, resulting in a dataset of 1.6 million positive and negative tweets. They performed sentiment prediction using three machine learning algorithms: MNB, Support Vector Machines (SVM), and Maximum Entropy. They used unigrams, bigrams, unigrams and bigrams, and unigrams with Part-Of-Speech

(POS) tags as features. Their results show the use of unigrams and unigrams with bigrams have the highest performance. SVM had the highest performance when using unigrams, while Maximum Entropy had the highest performance when using unigrams with bigrams. The difference in performance between the classifiers when using unigrams and unigrams with bigrams is smaller than 2%, and they conducted no tests to determine if this difference was significant. The authors mentioned bigram features alone did not perform well due to the length of tweets. As they are shorter posts, 140 characters or less, a bigram feature space becomes very sparse.

Other studies have looked into more advanced machine learning techniques to improve sentiment analysis using tweets. Saif et al. [111] looked at methods to reduce feature space in tweet sentiment analysis. Kouloumpis et al. [70] explored the effects of the boosting ensemble technique in tweet sentiment analysis, while Hannek et al. [46] determined the effects of bagging on tweet sentiment analysis. More recently, Prusa et al. [101] used a combination of ensemble techniques and feature selection to improve performance of tweet sentiment analysis.

Although tweets are currently a very popular data source, online reviews have been used prior to the existence of Twitter. Pang et al. [90] used movie reviews to predict public opinion in 2002. They obtained their data from IMDB, selecting reviews where the author's rating was expressed using a numerical system. This allowed for a direct conversion to positive and negative sentiment. Their final dataset consisted of 752 negative reviews and 1,301 positive reviews. A subset of this full dataset, 700 positive and negative instances, was chosen and randomly split into three folds for cross-validation, maintaining a balanced class ratio. Unigrams and bigrams were used as features. Three machine learning classifiers were applied to the data: Naïve Bayes, SVM, and Maximum Entropy. They found SVM to be the best performing learner and unigrams outperforming bigrams as a feature.

In the literature, most cross-domain research involving tweets and reviews have

attempted bridging the gap between the domains using features. This approach finds features that are independent of the domain and, through different techniques, bridges the gap between them [75]. Variations of this approach have been used to bridge domains, notably using domain independent topics, POS tags, and related semantic spaces. However, only one previous study [10] has used an approach similar to ours, which is combining instances from multiple sources into one dataset.

In their survey work, Aue et al. [10] bridged the gap between four different review domains. Sentiment analysis was performed for movie reviews, book reviews, Product Support Services (PSS) web survey data, and Knowledge Base (KB) web survey data. The datasets were composed of binary representation of unigrams, bigrams, and trigrams, where any word that occurred at least three times in any of the reviews were included for that specific domain. Thus, each feature set was unique to each domain. Four approaches were used: (1) training on a mixture of labeled data from other domains, (2) training a classifier using the same approach as that of step (1) but limiting the feature set to those observed in the target domain, (3) creating an ensemble of classifiers from other domains, and (4) combining labeled and unlabeled data in the target domain. For all four approaches the authors employed Support Vector Machines. Additionally, for approach 4 they trained Naïve Bayes within the Expectation Maximization algorithm. They validate their in-domain models using cross-validation and cross-domain models on one dataset. Their results show cross-domain under performing when compared to the in-domain approach, when a model is trained using data directly related from the same domain. Their study claims to have tested the results for significance using the McNemar test, however, the output of the test was not shown. The paper by Aue et al. is a survey work and as such, their study contains a less comprehensive analysis compared to more focused studies. Their study employed four review domains, but did not experiment with a different data source.

## 3.3 METHODOLOGY

This section will describe our methodology for both legs of the experiment. Cross-domain experiments are performed first and the results influence choices in the integration experimentation. Datasets and dataset sizes vary based on the experiments and are better detailed within their respective subsection. However, the process of feature extraction is the same for all experiments. The feature space across all datasets consists of a bag-of-words model. We elect to use a bag-of-words representation over TF-IDF as preliminary studies shown it to be more effective for our data. Features are extracted from the training data and applied to the test data to ensure both datasets have the same feature space. For both legs of the chapter, Weka's StringToWordVector filter [45] is used to extract the top 25,000 most frequent words from the training data to be used as features. This value was selected as no significant difference was observed when choosing between the 10,000 to 100,000 most frequent words. For the larger datasets 25,000 features will not encompass the total number of unique words, thus the 25,000 words which most frequently appear will be selected.

### 3.3.1 Cross-Domain Design

*Datasets*

| Dataset | Positive | Negative | Total |
|---|---|---|---|
| Sentiment140 | 5,000 | 5,000 | 10,000 |
| semEval | 3,420 | 1,399 | 4,819 |
| reviewSent | 1,896 | 940 | 2,836 |
| Total | 10,316 | 7,339 | 17,655 |

Table 3.1: Cross-Domain Dataset Characteristics

For our cross-domain experiments, three distinct datasets are used: a subsection of the Sentiment140 corpus [41], the semEval dataset [110], and a review spam dataset containing sentiment labels [73]. Table 3.1 shows the class distribution of all the

datasets.

The Sentiment140 corpus is a collection of tweets pulled directly from Twitter. These tweets were then automatically labeled using emoticons. Eight emoticons were used to label tweets as either positive or negative. This labeling process trades speed for accuracy, as some tweets may be mislabeled due to the context of the emoticon. Once these tweets were labeled, the data was pre-processed by removing the emoticons from the tweets, removing retweets, removing duplicated tweets, and removing instances that contained both positive and negative sentiment in the same tweet. The final dataset consists of 1.6 million tweets, with 800,000 positive and 800,000 negative instances. For our purposes, we use a subset of this total dataset; we use 10,000 instances sampled randomly, without replacement, from the full dataset.

The semEval dataset is also a collection of tweets, however, it is significantly smaller than the Sentiment140 corpus. Unlike the Sentiment140 corpus, the semEval dataset was labeled manually; thus, it is expected to have a smaller percentage of misclassified instances. The dataset contains positive, negative, and neutral class labels. The neutral instances were removed from the semEval datasets, therefore only positive and negative instances were used in our dataset. This pre-processing step was performed to match the number of classes found in the sentiment140 dataset. The class distribution is imbalanced in this dataset, as there are approximately two times as many positive instances as negative, which can be seen in Table 3.1.

The final dataset comes from a paper by Li et al. [73], whose study was in the domain of untruthful review detection [28]. However, their data also contains sentiment of the review. The data contains reviews from three domains: doctors, hotels, and restaurants. The dataset was created by requesting reviews to be written using a mix of Amazon Mechanical Turk [21] and expert reviewers. Expert reviewers consist of people who work in the domains. The dataset contains full reviews, text of the review, rating given, the domain of the review, and the class label (positive or

negative). This dataset is referred to as reviewSent in the remainder of the chapter.

| | Train | Test | Learner |
|---|---|---|---|
| 1 | sentiment140 | semEval | SVM |
| 2 | sentiment140 | reviewSent | SVM |
| 3 | reviewSent | sentiment140 | SVM |
| 4 | reviewSent | semEval | SVM |
| 5 | semEval | sentiment140 | SVM |
| 6 | semEval | reviewSent | SVM |
| 7 | sentiment140 | reviewSent | NB |
| 8 | sentiment140 | semEval | NB |
| 9 | reviewSent | sentiment140 | NB |
| 10 | reviewSent | semEval | NB |
| 11 | semEval | sentiment140 | NB |
| 12 | semEval | reviewSent | NB |
| 13 | sentiment140 | reviewSent | MNB |
| 14 | sentiment140 | semEval | MNB |
| 15 | reviewSent | sentiment140 | MNB |
| 16 | reviewSent | semEval | MNB |
| 17 | semEval | sentiment140 | MNB |
| 18 | semEval | reviewSent | MNB |

Table 3.2: Experimental setup

To determine the effectiveness of cross-domain training with reviews and tweets, each dataset is used to train a model twice and used to test a model twice for each classifier. For example, we use the sentiment140 subset to train a SVM classifier and the semEval dataset to evaluate the classifier. We then use the same sentiment140 subset to train a SVM classifier, but we now use the reviewSent dataset to evaluate it. This process is repeated, alternating the datasets used to train and test the model, and the classifier used until all combinations of datasets and classifiers have been exhausted. Our experiments consist of three learners and six combinations of training and test data. We conduct a total of 18 experiments (3 learners x 6 dataset combinations) to determine the performance of cross-domain sentiment analysis using tweets and reviews. Table 3.2 provides the complete combinations and learners used for all the experiments conducted.

### 3.3.2 Integration Design

*Datasets*

In the multi-source integration experiments the same three datasets described in Section 3.3.1 are utilized, in addition to an Amazon Product Review dataset[80]. However, instead of sampling only 10,000 instances from the sentiment140 corpus, we randomly sample, without replacement, 50,000 positive and 50,000 negative instances to generate a balanced dataset of 100,000 instances. Of the four datasets included, the sentiment140 and the Amazon review datasets will be used as training data, while the semEval and the reviewSent will strictly be used as testing data. Dataset characteristics can be found in Table 3.3.

For the second dataset, we constructed a large sentiment dataset from the Amazon Product dataset created by McAuley et al. [80]. The authors crawled Amazon to collect 142.8 million reviews spanning from May 1996 to July 2014. As reviews were not limited to any product type, the final dataset consisted of a diverse set of reviews. After removal of duplicate reviews, the dataset consists of 82.68 million reviews. From this total, 100,000 reviews were randomly sampled, without replacement, such that 50,000 reviews had a score greater than 3.5 stars and 50,000 reviews had a score less than 2.5 stars. The first group is labeled as positive sentiment, while the second group is labeled as negative sentiment, creating a positive/negative sentiment dataset matching the size and class distribution of the tweet sentiment dataset created from the sentiment140 corpus.

| Dataset | Positive | Negative | Total |
|---|---|---|---|
| Amazon Review | 50,000 | 50,000 | 100,000 |
| sentiment140 | 50,000 | 50,000 | 100,000 |
| semEval | 3,420 | 1,399 | 4,819 |
| reviewSent | 1,896 | 940 | 2,836 |
| Total | 105,316 | 107,339 | 207,655 |

Table 3.3: Integration Data Characteristics

We use two sources of training data, the Amazon review dataset and the sentiment140 corpus, for both sets of experiments. First, each of these two datasets is used to train a classifier with the full 100,000 instances available, which is then evaluated using the two test datasets. For our second experiment, we employ combinations of these two datasets; sampling 3,000, 5,000, 10,000, 25,000, 50,000, and 100,000 from each to create combination datasets with 6,000, 10,000, 20,000, 50,000, 100,000, and 200,000 instances. Dataset size has been shown to affect model performance in previous studies [95].

As mentioned before, the reviewSent and semEval datasets were used to evaluate classifiers trained with each training set. Thus, when evaluating each training set individually, we have the following four combinations of training and evaluation: sentiment140:reviewSent, sentiment140:semEval, Amazon:reviewSent, and Amazon:semEval. When evaluating our combined datasets, we use both the semEval and the reviewSent datasets. All combination dataset sizes are tested and the best performer is kept for comparison against the single source datasets.

For the single source models, classifiers are trained using bootstrapping to create ten training datasets from the initial training data via sampling with replacement and 10 runs of Multinomial Naïve Bayes (MNB). Each training bootstrap is the same size as the original training data (100,000 instances). When building the classifiers for the combined data experiment, we also train using 10 runs and bootstrapping to create datasets from the training data; however, the bootstrap datasets are sampled without replacement from the original training data, with the exception of 200,000 instances which was sampled with replacement. The bootstrap sizes vary depending on the size of the dataset, ranging from 6,000 to 200,000 instances.

### 3.3.3 Classifiers

In the experiments, we employ three different classifiers: Naïve Bayes, Multinomial Naïve Bayes, and Support Vector Machines. These classifiers were chosen as they commonly perform well in sentiment analysis research [101, 41, 90]. All classifiers were implemented using the WEKA toolkit [45] with default parameters, unless noted otherwise.

Naïve Bayes [109] falls under the category of Bayesian learners. Naïve Bayes aggregates Bayesian probabilities to approximate the posterior probability of an instance belonging to a class based on its feature values [132]. It makes the naïve assumption that all features are independent. Although, in general, this is not the case with the majority of features, Naïve Bayes still offers good performance. This is due to how the dependencies interact on a local and global level. Locally there may be strong dependencies between two features describing a class. However, when looked at globally the dependencies between features cancel each other out, allowing Naïve Bayes to perform optimally [135].

Similar to Naïve Bayes, Multinomial Naïve Bayes [81] is a second Bayesian learner and assumes a multinomial distribution to the data, rather than a Gaussian distribution. MNB still makes the naïve assumption of feature independence. However, the difference between NB and MNB is in the calculation of the likelihood. For example, in MNB for text classification, a document is assigned to the class which has the highest conditional probability. The probability is calculated as a count of the words in the document that overlap with the class, divided by the total number of words. If the product of the prior probability and the words found in class 1 is larger than the number of words overlapped with class 2, then the instance is classified as class 1, otherwise it is classified as class 2.

Support Vector Machine has been previously used for various machine learning problems involving reviews and tweets and found to have good performance in these

domains [27, 103]. SVM is based on the assumption that there is a linear discriminant between the feature space of two classes. SVM attempts to find a hyperplane that divides instances into two groups. The best such hyperplane would be the one that maximizes the distance between the hyperplane and members of each class. For our models, the complexity constant c was set to 5.0 and the buildLogisticModels parameter set to true.

### 3.3.4 Performance Metric

Performance of the classifiers were measured using the Area Under the receiver operator characteristic Curve [132]. We elect to use AUC as the ROC curve plots the performance of the model across all decision thresholds, thus, the area under this curve would provide the overall performance of the model [116]. The ROC is a graph of the False Positive Rate versus the True Positive Rate, and the area under the curve depicts performance of the model across all decision thresholds. Thus, a larger area under the curve means a better performing classifier. AUC values range from 0 to 1.0, with 0.5 being no better than a random guess and 1.0 being the perfect fit.

## 3.4 CROSS-DOMAIN

### 3.4.1 Naïve Bayes

The results for the Naïve Bayes classifier are presented in Figure 3.1, the y axis provides information in terms of which dataset was used to train and which was used to test the classifier (train / test format). From Figure 3.1, we can see the best performing combination is using the sentiment140 dataset to train the classifier and using the review dataset to evaluate. However, the difference between using sentiment140 and semEval to train and reviews to test is approximately 0.01. It is interesting to note that using tweets to predict review sentiment performs better than using tweets to predict tweet sentiment in a different dataset. We find that

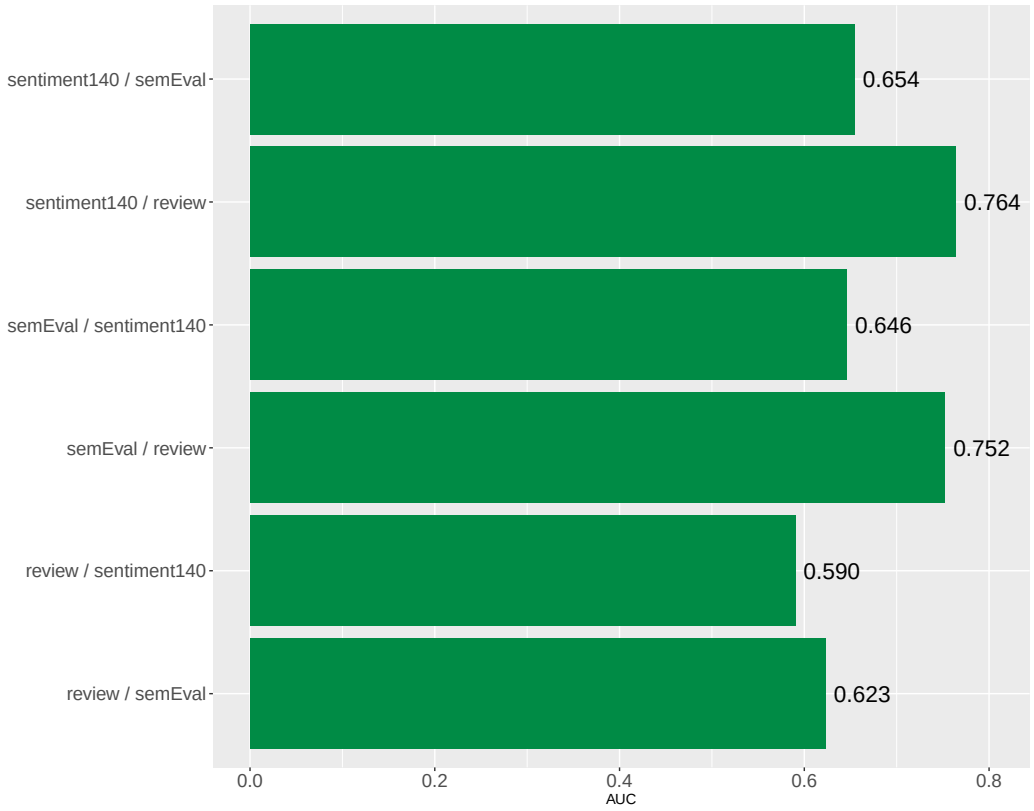



Figure 3.1: AUC values using NB for Cross-Domain Sentiment Analysis

using semEval to predict sentiment in the sentiment140 data and vice versa perform significantly worse than reviews. The lowest performance is observed when using reviews to train the model and tweets to evaluate.

### 3.4.2 Support Vector Machine

Results are again presented in a bar graph. Figure 4.2e shows performance of the support vector machine classifier on cross-domain sentiment analysis. We observe similar trends as with NB, however, in general, the AUC values when using the SVM classifier are higher. We see that using tweets to train and reviews to evaluate leads to the best performance, with sentiment140 producing a higher AUC than semEval. Similar to the results of NB, using tweets to predict review sentiment results in a higher AUC than using tweets to predict tweet sentiment in a different tweet dataset. The worst performance is observed when using reviews to train a the sentiment140

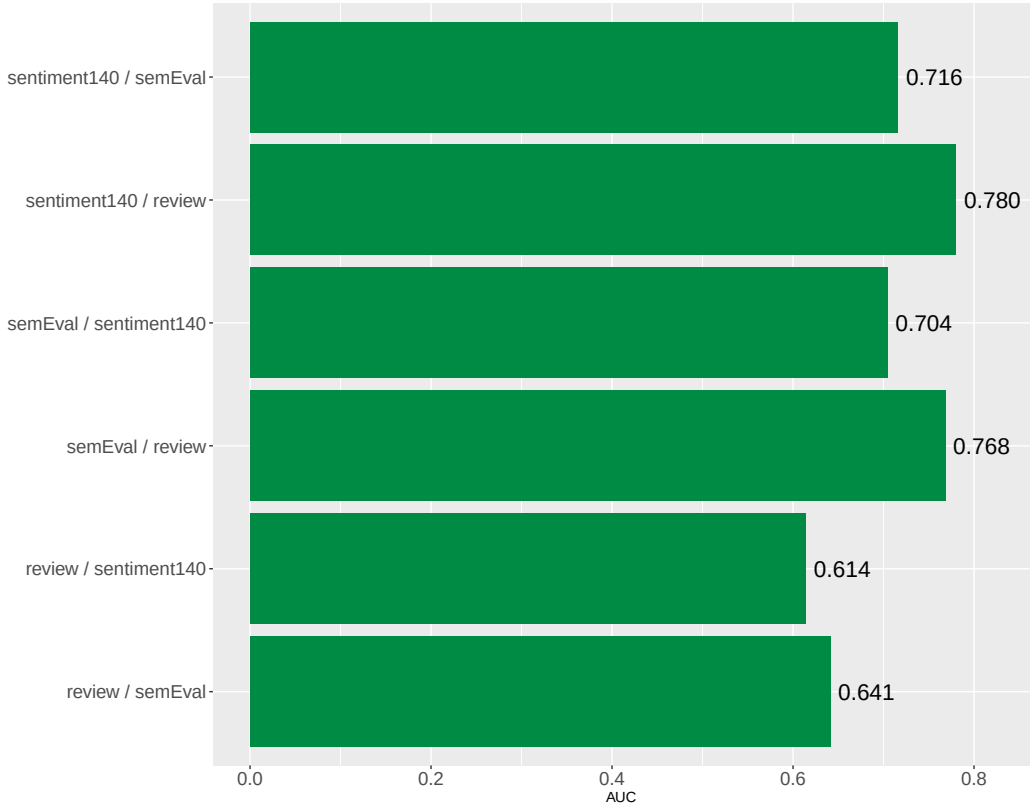



Figure 3.2: AUC values using SVM for Cross-Domain Sentiment Analysis

data to evaluate.

### 3.4.3 Multinomial Naïve Bayes

Finally, the results for the Multinomial Naïve Bayes classifier are presented in Figure 3.3. From Figure 3.3, we observe the same trends found when using NB and SVM. Moreover, the trends are more exaggerated than with the previous classifiers. The best performer is still using sentiment140 to predict sentiment in reviews; however, the difference between using sentiment140 and semEval to predict review sentiment is 0.07. This is significantly larger than the 0.01 difference in NB and the 0.015 difference in SVM. When comparing the tweet datasets we observe better performance when using sentiment140 to train a model that classifies sentiment in semEval. The worst performer is again using reviews to train a model and sentiment140 to evaluate it.

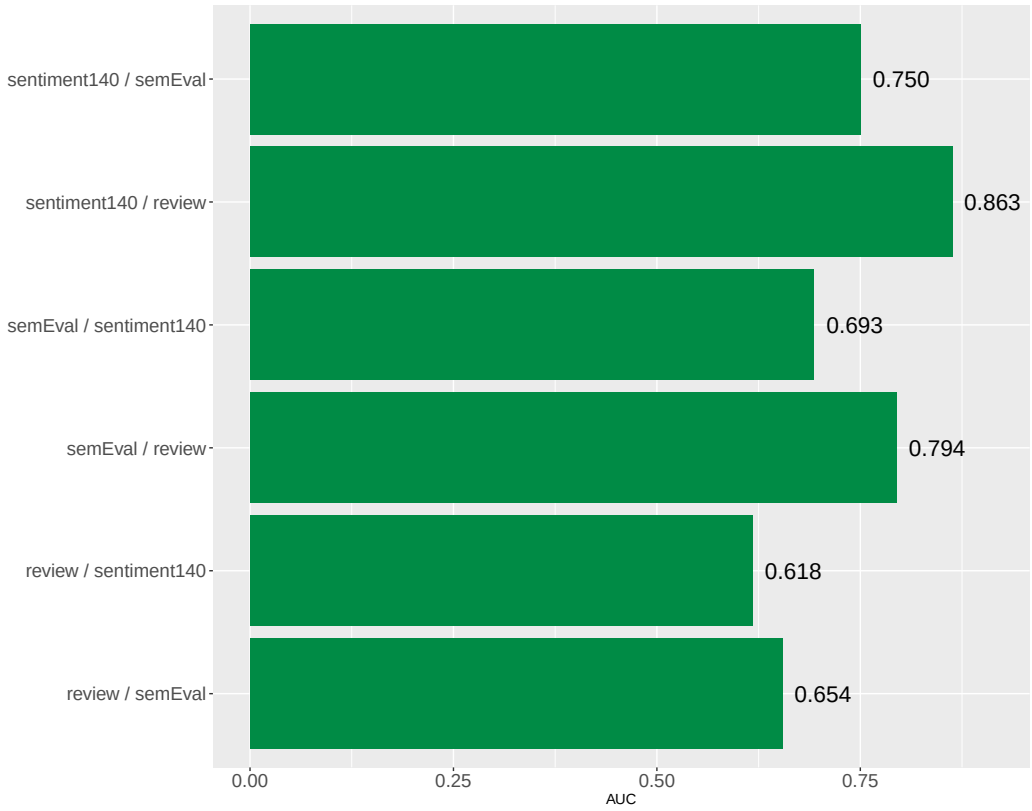



Figure 3.3: AUC values using MNB for Cross-Domain Sentiment Analysis

### 3.4.4 Discussion

The results presented in the previous subsections show similar trends across all three learners. Table 3.4 presents the AUC values grouped by learner and train/test combination, the best performer for the dataset combination is shown in **boldface**. In terms of classifiers, we see Multinomial Naïve Bayes resulting in the highest AUC for five of the six combinations, whereas SVM produces the highest AUC for the final combination. From this table and the results in previous sections, we can say that when performing cross-domain sentiment analysis between tweets and reviews, MNB serves as the best classifier.

When examining Table 3.4 and the previous results, some interesting trends appear that warrant further examination. The first of these is that using sentiment140 to predict sentiment in reviews is the top performing combination across all clas-

| Dataset | SVM | NB | MNB |
|---|---|---|---|
| sentiment140 / semEval | 0.72 | 0.65 | **0.75** |
| sentiment140 / review | 0.78 | 0.76 | **0.86** |
| review / sentiment140 | 0.61 | 0.59 | **0.62** |
| review / semEval | 0.64 | 0.62 | **0.65** |
| semEval / sentiment140 | **0.70** | 0.65 | 0.69 |
| semEval / review | 0.77 | 0.75 | **0.79** |

Table 3.4: AUC values by learner and dataset combination

sifiers. We see that the highest AUC for this combination was achieved using the Multinomial Naïve Bayes classifier. Keeping with this trend, the second top performing combination for all classifiers is using the semEval data to predict sentiment in reviews. These results confirm that tweets are a viable option for training a classifier that predicts sentiment in reviews. However, we see that the worst performing combination is using reviews to predict sentiment in tweets. The worst performer is using reviews to predict sentiment in the sentiment140 data, followed by using reviews to predict sentiment in the semEval data.

It is interesting to note that the best performer and the worst performer, both use the same datasets, however, tweets serve as better data for predicting results in reviews than the converse. This may be due to the difference in length between tweets and reviews. As tweets are composed of no more than 140 characters, patterns found in tweets are smaller than patterns found in reviews. Smaller patterns are applicable to both long and short text instances, however, longer patterns are not applicable to shorter bodies of text. Thus, the patterns learned from tweets are directly applicable to reviews, but longer patterns learned from reviews cannot be applied to tweets. Additionally, reviews are easier to classify as the reviews are written by users with the intent of conveying an opinion and users give a rating to their review that can be directly converted into a sentiment label.

Another trend worth noting is that using tweet sentiment datasets for training and testing results in lower AUC scores than training on tweets and testing on reviews.

This likely indicates that determining tweet sentiment is a more difficult task than determining review sentiment. It is possible that predicting sentiment in tweets is more difficult, as tweets are not necessarily written to convey opinion and they are shorter, therefore less data is available for determining sentiment. Additionally, tweet sentiment data is labeled with no knowledge of the users actual sentiment, while reviews sentiment labels are generated directly from user created scores. This likely leads to reviews having more accurate labels than tweets.

## 3.5 INTEGRATING TWEETS AND REVIEWS

For organizational purposes, we have split the results into subsections: Single Source datasets, Combined datasets, and Comparison. We will first delve into the results for training using only the sentiment140 corpus and the Amazon review data. Then, we explore the results of combining these two training sets using different subset sizes of this combined dataset. Finally, we compare the model performance of combining tweets and reviews against using only tweets or only reviews.

### 3.5.1 Single Source datasets

The mean AUC for the 10 bootstrap dataset runs are presented in Table 3.5 below. Using the Amazon product reviews dataset, we observe classifier results of 0.97 AUC when tested on doctor, hotel, and restaurant reviews (reviewSent); however, this classifier does not work well when tested on the tweet sentiment dataset (semEval). Training from tweets provided decent performance across both datasets used for evaluation, with the highest performance achieved when evaluating with the reviewSent dataset. These results provide us with a baseline, with which to compare the results of the combined datasets in the following subsection. To determine whether the difference in performance between datasets is significant, our results are tested for statistical significance, at the 95% confidence level, using a two-factor ANalysis Of

VAriance (ANOVA) [12]. The two factors in our ANOVA are the choice of test and training datasets.

The two-factor ANOVA test results are presented in Table 3.6 and show the choice of training and test datasets, and that the interaction between training and test datasets contribute significantly to the variance observed in our data. Our Pr(>F) values are rounded to four digits, causing some values to appear as if they are zero, when actually they are very close to zero. This ANOVA indicates a significant difference in performance will occur depending on the dataset used to train and the one used to evaluate. To determine what datasets perform significantly better, a Tukey's Honestly Significance Difference (HSD) test is performed [12]. A Tukey's HSD test is used to conduct pairwise similarity tests to determine if the differences between two given methods are statistically significant. Results for the Tukey's HSD test are presented in Table 3.7.

From Table 3.7, we see the average AUC and Tukey's groups across all experiments for both test and training data. If the datasets share the same letter, then there is no significant difference between the two. Table 3.7-A shows the ranking for the training datasets. We see that sentiment140 falls into group 'a', while the amazon review data falls under group 'b', meaning the sentiment140 dataset is significantly better at training a model than the Amazon Reviews, in the context of our test data. When looking at the performance of the test datasets, shown in Table 3.7-B, we see that there are two distinct groups. ReviewSent falls into group 'a', while semEval is group 'b'. These groupings indicate reviewSent is the easier of the two test datasets

| Training Data | Test Data | AUC |
|---|---|---|
| sentiment140 | reviewSent | 0.8681 |
| sentiment140 | semEval | 0.7705 |
| Amazon | reviewSent | 0.9677 |
| Amazon | semEval | 0.6945 |

Table 3.5: Results for Single Source dataset Experiments

| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| Test Data | 1 | 0.34 | 0.34 | 6779.51 | 0.0000 |
| Training Data | 1 | 0.00 | 0.00 | 27.37 | 0.0000 |
| Test Data:Training Data | 1 | 0.08 | 0.08 | 1519.54 | 0.0000 |
| Residuals | 36 | 0.00 | 0.00 | | |

Table 3.6: Two-Factor ANOVA for Single Source datasets

| | Training Data | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | sentiment140 | a | 0.792 | 0.057 |
| 2 | Amazon | b | 0.769 | 0.144 |

(A) Tukey's HSD Results for Training datasets

| | Test Data | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | reviewSent | a | 0.918 | 0.052 |
| 2 | semEval | b | 0.732 | 0.040 |

(B) Tukey's HSD Results for Test datasets

Table 3.7: Tukey's HSD for Training and Test Data for Single Source datasets

for predicting sentiment. Figure 3.4 plots the interactions between each training:test pair, where the dot is a representation of their average AUC across ten bootstraps. If there is no overlap between the lines extending from the dot (the confidence interval), then there exists a significant difference in performance between models.

### 3.5.2 Combined datasets

This subsection presents results for classifiers trained from a combination of both tweets and reviews. The results for using the combined approach can be found in Table 3.8. dataset size refers to the number of instances found in the combined dataset, half the instances were sampled from each data source. We observe the difference between using a sample size of 6,000 vs. 200,000 only changes AUC by 0.03 to 0.06. As seen in the previous subsection, using reviews to evaluate the performance of the models produces higher AUCs when compared to using tweets.

The purpose of Figure 3.5 is to compare the changes in AUC across different subset sizes for each test dataset, not to compare the performance of the models across test sets. From Table 3.8 and Figure 3.5, we observe there is less variation in AUC when

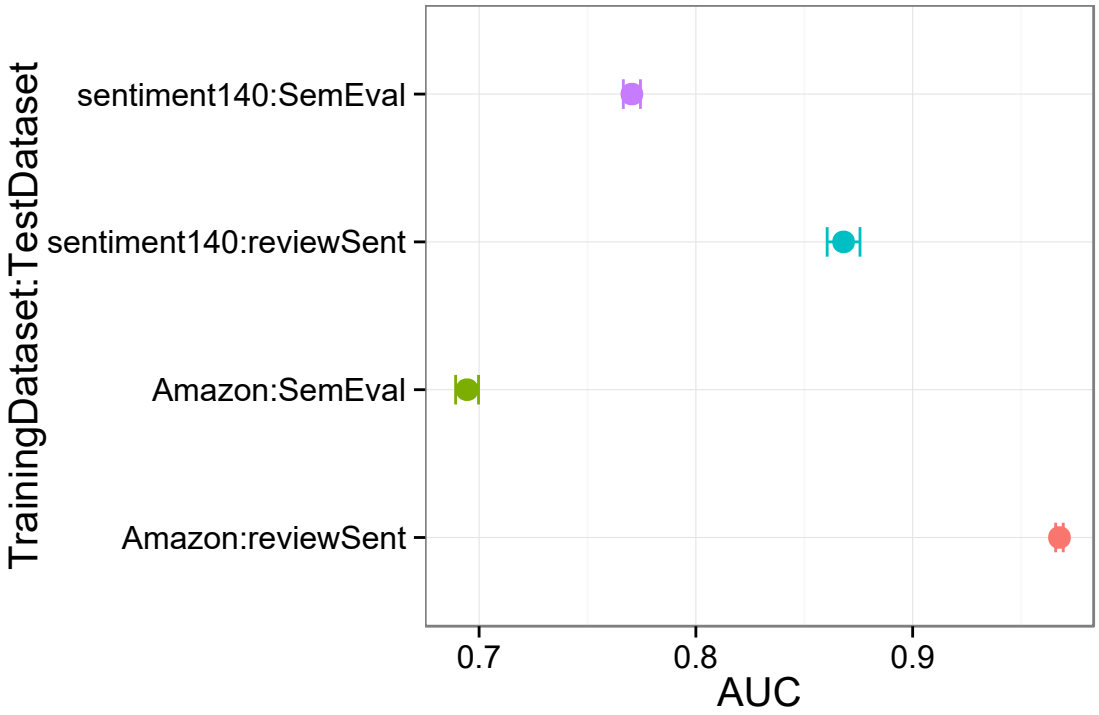



Figure 3.4: Multicomparison Figure for Tukey's HSD

| dataset Size | Test Data | AUC |
|---|---|---|
| 6,000 | reviewSent | 0.9449 |
| | semEval | 0.7163 |
| 10,000 | reviewSent | 0.9504 |
| | semEval | 0.7329 |
| 20,000 | reviewSent | 0.9601 |
| | semEval | 0.7449 |
| 50,000 | reviewSent | 0.9669 |
| | semEval | 0.7623 |
| 100,000 | reviewSent | 0.9694 |
| | semEval | 0.7725 |
| 200,000 | reviewSent | 0.968 |
| | semEval | 0.7674 |

Table 3.8: Results for Combined datasets

evaluated on reviews, compared to when evaluated on tweets, while increasing the number of training instances. This may be due to one of two reasons. As the AUC is already very high when evaluated on reviews, training on more instances results in little change in performance. Alternatively, less review instances are needed to train a

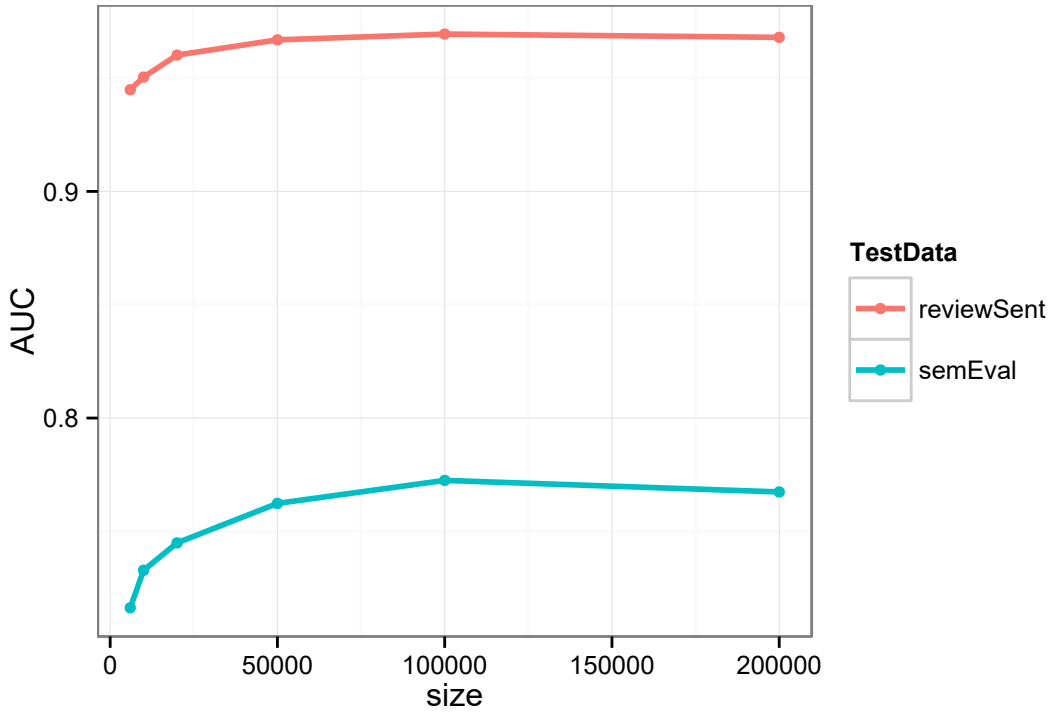



Figure 3.5: AUC vs Number of Instances for Both Evaluation datasets

classifier as reviews are longer than tweets. It is most likely that the first explanation is correct, as both exhibit similar asymptotic behavior when adding additional instances.

We also conduct a two-factor ANOVA test to determine if there is any significant difference between sampling sizes for the training dataset. The factors for this ANOVA test are dataset size and test dataset. Table 3.9-A presents the results of the ANOVA test. We observe p-values near zero, indicating there is a significant difference between sample sizes, test datasets, and their interactions. To determine what sample sizes are best, we conduct a Tukey's HSD test with results presented in Table 3.9-B. We observe no significant difference between a sampling size of 200,000 and 100,000 as both are in group 'a', however, a sampling size of 100,000 instances performs significantly better than the remaining dataset sizes. This indicates there is no significant benefit to using more than 100,000 combined instances.

| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| dataset Size | 5 | 0.03 | 0.01 | 137.05 | 0.0000 |
| Test Data | 1 | 1.33 | 1.33 | 35372.76 | 0.0000 |
| dataset Size:Test Data | 5 | 0.00 | 0.00 | 19.10 | 0.0000 |
| Residuals | 108 | 0.00 | 0.00 | | |

(A) ANOVA Results for Combined Training dataset Sizes

| | dataset Size | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | 100,000 | a | 0.871 | 0.101 |
| 2 | 200,000 | ab | 0.868 | 0.103 |
| 3 | 50,000 | b | 0.865 | 0.105 |
| 4 | 20,000 | c | 0.853 | 0.111 |
| 5 | 10,000 | d | 0.842 | 0.112 |
| 6 | 6,000 | e | 0.831 | 0.118 |

(B) Tukey's HSD Results for Comined Training dataset Sizes

Table 3.9: ANOVA and Tukey's HSD Tests for Combined dataset Sizes

### 3.5.3 Comparison

In this subsection, we compare the performance of the best combined model against the two single source models. We select the best combination training dataset, which uses 100,000 instances sampled from both Amazon reviews and the sentiment140 corpus. We compare this combined dataset, referred to as Combo in the following tables, against the classifiers trained from 100,000 tweets and 100,000 reviews. All three are evaluated on two test datasets: reviewSent and semEval. Results are presented in Table 3.10. We see from the results that the combined dataset yields the highest AUC for both test datasets. From the two-factor ANOVA presented in Table 3.11, we see there is significant difference in both training and test datasets as well as their interaction. To further examine these results and determine which datasets are statistically significant, we perform two Tukey's HSD tests, found in Table 3.12.

We first conduct a Tukey's HSD test to see if there is significant difference between training data sources for each test set, this HSD test is shown in Table 3.12-A. There are four distinct groups 'a' through 'd'. When using the reviewSent test set, models generated with the Combo and Amazon training sets are group 'a', while sentiment140

| | Training Data | Test Data | AUC |
|---|---|---|---|
| 1 | Combo | reviewSent | 0.9694 |
| 2 | Amazon | reviewSent | 0.9677 |
| 3 | sentiment140 | reviewSent | 0.8681 |
| 4 | Combo | semEval | 0.7725 |
| 5 | Amazon | semEval | 0.6945 |
| 6 | sentiment140 | semEval | 0.7705 |

Table 3.10: Average AUC of Combined and Single Source datasets

| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| Training Data | 2 | 0.03 | 0.01 | 384.47 | 0.0000 |
| Test Data | 1 | 0.54 | 0.54 | 14100.41 | 0.0000 |
| Training Data:Test Data | 2 | 0.08 | 0.04 | 1017.16 | 0.0000 |
| Residuals | 54 | 0.00 | 0.00 | | |

Table 3.11: ANOVA for Comparison of Combined dataset against Single Source datasets

is in group 'b'. We also observe that when utilizing the semEval evaluation set, we see Combo and sentiment140 sharing group 'c', while Amazon is in group 'd'. A commonality we observe is that Combo is in the top group for evaluating both tweets and reviews, unlike classifiers trained from a single source, which are only in the top group for classifying within their respective domains. We can also see if there is any significant difference between choice of training dataset when considering classifier performance across multiple domains (performance across both evaluation datasets). The HSD test found in Table 3.12-B provides these results. Here, we see the combination of reviews and tweets produces a classifier that is significantly better than using a single training domain if we are concerned with performance across both test sets. This is expected as previously we saw using a combination of reviews and tweets yielded a top performing classifier for both reviews and tweets, while using a single training source only performed well for that specific domain.

| | Training Data:Test Data | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | Combo:reviewSent | a | 0.969 | 0.001 |
| 2 | Amazon:reviewSent | a | 0.968 | 0.002 |
| 3 | sentiment140:reviewSent | b | 0.868 | 0.011 |
| 4 | Combo:semEval | c | 0.773 | 0.005 |
| 5 | sentiment140:semEval | c | 0.770 | 0.006 |
| 6 | Amazon:semEval | d | 0.694 | 0.007 |

(A) Tukey's HSD Test Results for Training datasets on Both Test Sets

| | Training Data | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | Combo | a | 0.871 | 0.101 |
| 2 | Amazon | b | 0.831 | 0.140 |
| 3 | sentiment140 | c | 0.819 | 0.051 |

(B) Tukey's HSD Test Results Averaged Across Both Test Sets

Table 3.12: Both Tukey's HSD Tests for Comparison between Combined and Single Source Data

## 3.6 CHAPTER SUMMARY

In this chapter, we set out to determine the performance of cross-domain sentiment analysis using either tweets or reviews to train the model and to determine the performance of multi-domain sentiment analysis using training data that includes both tweets and reviews.

For the first part of this chapter, we conducted an empirical study consisting of three datasets and three different classifiers. A total of 18 experiments were conducted across 6 combinations of training/testing datasets and three classifiers. For the second part of this chapter, we conducted two experiments using two training datasets and two test datasets with the Multinomial Naïve Bayes learner. The MNB learner was chosen as it was found to be the best learner in the first leg of the experiments. Next, we created multiple combined datasets, ranging from 6,000 to 200,000 instances, from the two training data sources. All combination dataset sizes are tested and the best performer was used in a comparison against the two single source models.

We found that using the sentiment140 dataset produced a top performing classifier for prediction of sentiment in tweets, while using the Amazon dataset produced a top

performing classifier for prediction of sentiment in reviews. Performing cross-domain analysis with classifiers trained using either single data source dataset performed significantly worse than in-domain classification. Thus, single source sentiment classification is not always ideal if we wish to work in multiple domains; however, by creating a dataset consisting of both tweets and reviews, a model can be trained that performs well at classifying both. The combined dataset had higher AUCs than single source datasets within their domains (though not significantly so), but significantly better performance than the single source sets outside of their domain.

In our evaluation of training set size, we found that dataset size plays a large role in the performance of classifiers trained from the combined datasets. As dataset size increases, performance of the model significantly increases, up to 100,000 instances. There was no statistical difference between using 100,000 instances and 200,000 instances.

# CHAPTER 4
# TEXT MINING FOR THE DETECTION OF FAKE ONLINE REVIEWS

## 4.1 BACKGROUND AND MOTIVATION

Spam reviews can be classified into three categories: untruthful reviews, reviews on brands, and non-reviews [34]. Untruthful reviews are fake reviews created with malicious intent. Reviews on brands are reviews not specific to a product, but the entity creating the product. Non-reviews contain all reviews that do not actually review a product. This research focuses on the detection of untruthful reviews, as these have been shown to be the most difficult to detect [63].

Spam reviews are created by "spammers" or suspicious persons to either harm or bolster the reputation of an establishment or product. As these reviews are all public, usually posted by an unknown user, and not passed via word of mouth, it becomes increasingly difficult to determine whether the review is coming from a reputable source. Up to one-third of all online reviews are considered untruthful; this increase in false reviews has led to a drop in credibility of online reviews [42]. To combat this issue, many methods using machine learning tools, specifically supervised learning, have been proposed.

Supervised machine learning algorithms are trained on labeled datasets, in our case spam review datasets. These algorithms learn to classify reviews as truthful or untruthful based on features describing the reviews. These labeled datasets contain instances (the reviews), features describing the review, and their corresponding class label. The features describing a review typically include review text features, the meta

data found in the review (rating, date, time, etc.), and reviewer-oriented features, which describe the user that created the review. Since we are interested in predicting spam reviews based only on the text, we only explore the effects of text based features and choose to leave other features for future work.

Unfortunately, due to the text-based nature of reviews, the feature space associated with a dataset in the domain of spam detection is large. Having a large feature space is commonly known as high dimensionality and creates challenges, such as higher computational costs and redundant or useless features [47]. Classification algorithms are not adept at handling a high-dimensional feature space and will see degraded performance due to overfitting [27]. A family of machine learning techniques, known as feature selection, can be implemented to combat the problem of high dimensionality. Feature selection has only recently been employed in spam review detection studies [27, 84].

In addition to feature selection, a family of techniques called ensemble learning techniques have been shown to increase classifier performance and reduce overfitting [32]. In previous work, we have found ensemble techniques, such as Bagging and Boosting, increase classification performance in other text classification domains [103]. However, ensemble techniques alone do not combat the issue of high dimensionality [54]. To combat high dimensionality, while still taking advantage of the ensemble framework, feature selection can be embedded within ensemble techniques. We employed three of these techniques: Select-Boost, Select-Bagging, and Random Forest. These three techniques combine feature selection and ensemble learning to form a classifier which is resilient to the adverse effects brought upon by high dimensionality. We build upon the results found in [27] by examining ensemble learners, which are known to improve performance in a variety of domains, and determine how the addition of feature selection to ensemble algorithms impacts classification performance.

This chapter makes the following contributions:

- We explore the effectiveness of using feature selection, ensemble methods, and the combination of the two when training models to detect spam reviews.
- Two novel ensemble techniques are proposed to handle the challenges associated with the detection of spam reviews.
- For these novel ensemble methods, we evaluate the performance of five base learners and three feature selection techniques.
- We conduct statistical analysis using ANOVA and Tukey's HSD tests to determine the best combination of ensemble, feature selection technique, and base learner.

The remainder of this chapter is organized as follows. Section 4.2 contains works related to spam detection, ensemble learning, and feature selection. Section 4.3 provides insight into feature selection techniques. Section 4.4 describes ensemble techniques, such as Boosting and Bagging. In Section 4.5, we examine techniques that combine feature selection and ensemble learning: Select-Boost, Select-Bagging, and RF. Section 4.6 summarizes our methodology including dataset information, learners, cross-validation, and the performance metric. Section 4.7 contains the first case study, which presents baseline results for each base learner and the effectiveness of feature selection. In Section 4.8, we present our second case study, discussing the results for ensemble techniques and the combination of ensemble and feature selection. Section 4.9 provides a discussion of the results. Finally, in Section 4.10, we summarize the contents of this chapter.

## 4.2 RELATED WORKS

The area of spam review detection includes various studies dedicated to detecting untruthful reviews using supervised learning. There exist multiple types of feature

sets which may be used for detection of untruthful reviews [28]. However, one of the main problems with spam review detection is labeled dataset availability. A study by Jindal et al. [63] used review-oriented features to predict whether a review was classified as spam or not spam. A dataset was generated by collecting 5.8 million reviews of products offered by Amazon. A subset of 222,000 instances were sampled from the original dataset and a method known as w-shingling [20] was utilized to detect near-duplicate reviews. The near-duplicate reviews were classified as untruthful reviews. From these reviews, 36 additional features were derived and used in a logistic regression model to predict spam reviews. These additional features consisted of Part Of Speech (POS) and reviewer-centric features. Using the full feature set, their model resulted in an AUC of 0.78 when detecting duplicate reviews. The dataset was tested against two other review categories: brand only and non-reviews. For brand only and non-reviews, the resulting AUC values were over 0.98, indicating the detection of untruthful reviews to be a more difficult task.

A study by Ott et al. [87] proposed a dataset of considerably smaller size with an equal number of positive (spam) and negative (non-spam) instances. The authors put forth a dataset created using Amazon Mechanical Turk (AMT) [21]. AMT is a service provided by Amazon that allows access to a scalable work force. By leveraging AMT, the authors were able to generate fake positive reviews for hotels. The dataset consists of 800 positive reviews with half the positive reviews being false, while the remaining half was obtained from TripAdvisor. The best model created had an accuracy measure of 89.8% using a Support Vector Machine (SVM) classifier with a combination of Linguistic Inquiry and Word Count (LIWC) and bigram features. Although this dataset has been used in many studies, it was shown to have a major flaw in that it is not representative of real-world review scenarios. The word distributions found in the AMT reviews are significantly different from distributions found in truthful reviews, allowing for easier detection of false reviews [84]. A more realistic dataset

was proposed by Li et al. [73] which extends the AMT dataset created by Ott et al. [87] through the addition of expert reviews and by combining reviews from three different domains: hotels, restaurants, and doctors. To create a more comprehensive and accurate real world dataset, Li et al. [73] obtained reviews from employees of the locations being reviewed in addition to reviews from AMT. The authors elected to use the hotel reviews to train the model and the doctor reviews to test the model. Using a classification algorithm modeled off the Sparse Additive Generic Model (SAGE), they were able to obtain an accuracy of 64.7% and 63.4% when using LIWC and POS features, respectively. We elected to use this dataset in our case studies, as it most closely represents real-world review scenarios.

In Shojaee et al. [117], stylometric attributes were used for detection of untruthful reviews. Stylometric features consist of lexical and syntactic features. These include features which give an indication of the grammar and vocabulary used by the writer. Lexical features focus on word or character use, while syntactic features represent the style of the writer, such as their use of the words "the", "of", "a", etc. Three sets of experiments were performed using the dataset from [87]. The first set used solely lexical features, the second used syntactic features, and the final set used a combination of both. Two classifiers were trained with each feature set: SVM and Naïve Bayes. The classifiers were trained using ten-fold cross-validation. In all three feature sets, SVM outperformed Naïve Bayes in terms of F-measure. The best F-measure, 0.84, was obtained using the feature set containing the combination of lexical and syntactic features. The increase in feature space, due to the addition of stylometric features, requires higher computational costs and no feature selection was done to optimize the feature space.

A study by Mukherjee et al. [84] used reviewer-centric features to detect spam reviews. The dataset used was a combination of the dataset found in [87] and reviews collected from Yelp. In total, there were three feature sets, one consisting of LIWC

features, one consisting of POS features, and the final set contained bigram features. The study applied feature selection using Information Gain (IG) to select the top 1% and 2% of features from each set. The models created using the reduced feature space did not generalize well to real-world data. Feature selection was found to offer no improvement on classification performance. However, only a single feature selection technique was tested, thus, no conclusive results can be drawn from the effect of feature selection on the classification of spam reviews. A study by Crawford et al. [27] determined the effects of ten feature rankers on classification of spam review. Ten filter-based feature selection techniques were applied across five classifiers and a range of subset sizes. Out of the ten feature selection techniques, Chi-Squared was chosen for comparison against a word frequency feature selection method. Results show Multinomial Naïve Bayes has the highest performance and Chi-Squared performs better than word frequency at low subset sizes across all classifiers. However, at large subset sizes, word frequency performs just as well as Chi-Squared, if not better.

Ensemble techniques have largely gone unused in spam review detection studies in exchange for a more traditional supervised learning approaches. Our case studies are the first to apply a wide range of ensemble techniques in the area of spam review detection. Ensemble techniques may be useful for detecting untruthful reviews as they have been shown to increase performance in noisy data [32]. There is evidence of the effects of ensemble techniques on classifier performance in related domains such as sentiment detection [103]. A previous study on tweet sentiment classification showed the improved effects of employing ensemble classifiers with feature selection [103]. The study compares the performance of Select-Boost and Select-Bagging against various feature selection techniques using five base learners: K-Nearest Neighbor (KNN), C4.5 decision tree, Multilayer Perceptron (MLP), and Logistic Regression (LR). Two tweet sentiment datasets were used: SemEval and Sentiment140. A Threshold-Based Feature Selection (TBFS) technique, and the area under the Receiver Operator Char-

acteristic (ROC) curve metric was employed within Select-Boost and Select-Bagging. Results showed Select-Boost performing significantly better than Select-Bagging and plain classification on both tweet sentiment datasets.

This chapter presents two case studies that tackle feature selection and ensemble methods for improving the detection of spam reviews. Moreover, to the best of our knowledge, no previous research has examined the effect of Boosting and Bagging in this domain. In addition, the use of novel ensemble techniques that take advantage of feature selection methods, such as Select-Boost and Select-Bagging, are explored in our experiments. To the best of our knowledge, this is the first comprehensive work to present a thorough comparison of these techniques in the realm of spam review detection.

## 4.3 FEATURE SELECTION

This section will provide an overview of the feature selection techniques included in the case studies. Feature rankers provide the type of fast and scalable feature selection suitable for very large feature spaces [102]. They use various measures to rank features based on performance. Features are given a score relative to the class label based on the performance metric used. In this case study, filter-based feature rankers were employed, as they scale well with high dimensional datasets and have been previously used in detection of untruthful reviews [27]. We applied ten different feature rankers to the dataset from three different feature selection families: commonly-used [33], Threshold-Based Feature Selection (TBFS) [33], and First Order Statistics (FOS) [67]. These families use distinctly different methods to rank features. The following three subsections describe the feature ranker techniques used and the families they belong to.

### 4.3.1 Commonly Used Techniques

This family of commonly used techniques consists of techniques that are widely used and easily accessible. These techniques are found in most open source machine learning tool-kits, such as the Weka toolkit [45]. From this family of techniques, we employ the Chi-Squared (CS) ranker [132].

Chi-Squared utilizes the Chi-Squared statistic to measure the relationship between a feature and the class. This feature ranker measures the independence of two events, in this case the events are the occurrence of the feature and the occurrence of a specific class. If we consider the occurrence of a feature as A and the occurrence of a class as B, then these occurrences are independent if $P(AB) = P(A) * P(B)$.

### 4.3.2 Threshold-Based Feature Selection Techniques

TBFS techniques are bi-variate procedures, where each feature is evaluated against the class, independent of all other features [33]. All attributes are first normalized in a range [0,1], then a classifier is built for all thresholds $t \in [0, 1]$. Two classification rules are used for building the simple classifiers. Rule 1 classifies instances with a normalized value greater than t as positive, and examples with a normalized value less than t as negative. Rule 2 is the converse of rule 1 and classifies instances with a normalized value greater than t as negative, while examples with a normalized value less than t are positive. The metric $\omega$ is used to create the attribute ranking. These $\omega$ metrics are primarily used to measure classification performance [33]. We use Mutual Information (MI), the area under the Precision-Recall Curve (PRC), the area under the Receiver Operator Characteristics curve (ROC), Gini-Index, Kolmogorov-Smirnov (KS), Significance Analysis of Microarrays (SAM), and Probability Ratio as our $\omega$ metrics used to rank the features.

Mutual Information determines the importance of a feature by measuring the contribution of the presence, or absence, of a feature towards a correct classification

[91].

The PRC metric measures the area under the curve of Recall on the x-axis and Precision on the y-axis. Recall is defined as the True Positive Rate (TPR) and Precision is defined as the fraction of instances classified as positive that are actually positive.

The ROC metric measures the area under the curve of False Positive Rate (FPR) on the x-axis and TPR on the y-axis. This directly indicates model performance, where the higher the area under the ROC curve the better the model performance across all thresholds.

Probability Ratio was used for text classification in [38] and was defined as the probability of the feature given the positive class divided by the sample estimate probability of the feature given the negative class. The formula ($TPR/FPR$), describes the probability ratio between the positive and negative classes.

The Gini-Index was originally utilized for measuring income over a population. The Gini-Index has a range from 0 to 1, which indicates the following: GI=0 implies the income is split evenly across the population, and GI=1 means that a single person receives all the income. The feature ranker version of the Gini-Index analyzes the distribution of a feature across the classes.

The KS statistic is a non-parametric test that measures the difference in the cumulative distribution of two datasets.

SAM was originally created for detecting significance in microarrays within the bioinformatics domain. SAM compares the observed and expected values of the parameter to identify features whose associated values differ by an amount of statistical significance among the sets [125].

### 4.3.3 First Order Statistics Techniques

The First Order Statistics (FOS) family of techniques are univariate feature rankers that use first order statistic measures, such as mean, mode, and standard deviation, to rank features in order of importance. In our experiments, two feature rankers from this family are used: Signal-to-Noise (S2N) [23] and Wilcoxon Rank Sum (WRS) [19].

The Signal-to-Noise ratio represents the ratio of signal information to noisy background information. This process can be used to determine how well a feature discriminates between two classes.

The Wilcoxon Rank Sum is a variation of the standard t-statistic. In a standard t-statistic, the distribution is assumed to be normal, while in the WRS no assumption on the distribution is made. There are two steps to be performed before defining the WRS: 1. all instances are ranked based on the value of the feature, and 2. all the rankings in the positive class are summed as $W_p$.

## 4.4 ENSEMBLE TECHNIQUES

Two different ensemble algorithms are used in our experiments: Boosting and Bagging. These techniques are similar in that they combine multiple instances of a base learner to generate a more robust and generalized classifier; however, the process by which this is achieved is different. Boosting takes an iterative approach; training and evaluating a model using a classifier, then training and evaluating subsequent classifiers based on the results of the previous classification [39]. In our case studies, we use the AdaBoost family of techniques, specifically AdaBoost.M1. All initial weights are set to one. In our case study, ten boosting iterations are performed and model results are aggregated using majority voting. Due to the re-weighting step, Boosting is not compatible with certain base learners; however, a separate approach, where the instances are re-sampled from the original dataset according to the weights, allows these base learners to be compatible with Boosting. This data sampling approach is

applied to re-select instances in accordance with the assigned weights.

Bagging, also known as bootstrap aggregating, uses data sampling with replacement (re-sampling from the original dataset where the same instances can be re-selected) to create a predefined number of bootstrap datasets. In our case study, ten bootstrap datasets are created. These new bootstrap datasets are the same size as the original and contain instances from the original dataset. The bootstrap datasets are then used to train a single base classifier each. Once all the classifiers have been trained an aggregation technique, in our case majority voting, is used to determine the final classification. Bagging has been shown to increase performance of weak classifiers but adversely affect the performance of stable learners [18].

## 4.5 ENSEMBLE TECHNIQUES WITH FEATURE SELECTION

In addition to Boosting and Bagging, we want to examine the effects of ensemble techniques that take advantage of feature selection. Three of these techniques are used in our experiments and explained in this section: Select-Boost, Select-Bagging, and Random Forest.

Select-Boost is a modified version of the AdaBoost.M1 algorithm developed by our research team. The Select-Boost [103] framework embeds feature selection within the Boosting ensemble framework. As with Boosting, Select-Boost is also incompatible with certain base learners due to the re-weighting step. To mitigate this, we re-sample from the original dataset according to the weights assigned and create a new training dataset where the probability of selecting an instance is equal to its weight. Initially, all samples in the training data are assigned equal weights.

Select-Bagging [103], like Select-Boosting, adds a feature selection step to the Bagging algorithm.As with Bagging, ten bootstrap datasets are created for Select-Bagging. Feature Selection is applied to every bootstrap dataset and then each dataset is used to train a classifier. The results of those trained classifiers are ag-

gregated and a final classification is made through majority voting.

Random forest constructs an ensemble of decision trees in an effort to avoid overfitting and improve classification performance by averaging many deep decision trees trained on different subsets and feature subspaces of the training data [74]. Random forest implements bagging and random subspace sampling. For every tree created, a new dataset is used for training. In addition, at every node random features are used to create the split.

## 4.6 EXPERIMENTAL METHODOLOGY

This section describes the general methodology for our experiments with ensemble and feature selection techniques. We describe the dataset, the base learners, cross validation, and the performance metric.

### 4.6.1 Dataset

Our dataset originates from the study by Li et al. [73]. The dataset contains spam reviews from three distinct domains: doctors, hotels, and restaurants. The original dataset was obtained through as similar process as used in Ott et al. [87], using AMT. AMT [21] is a service provided by Amazon which gives customers access to an on-demand, scalable workforce. This workforce was employed to create fake reviews using real reviews as a baseline. As AMT uses regular workers, whom may have no knowledge of domain keywords, the word distributions found in the reviews are significantly different from word distributions found in real reviews [84]. These synthetic reviews are not always representative of real-world spam data; to counteract this, domain experts were hired to create false reviews for their specific domains. For example, an employee in a doctor's office was asked to write false reviews for their practice. Li et al. [73] posited employees would have a better grasp on daily processes which would lead to a better synthetic spam reviews. Two examples of reviews from

the dataset can be found below. From a human standpoint, determining which is spam between the two from reading the text alone is near impossible. The following review is an untruthful spam review:

> *The rates at The Talbott Hotel were cheaper than I had expected, and that was my reason for booking a room. I had been prepared for service and a room similar to what I had experienced in the past, and I was quite pleased when I did stay. The room was neat and clean, and the halls were quiet at night. The traffic noise was muffled to the point where it was no problem sleeping either. I did ask one question at the service desk and they answered it nicely, which is good because normally hotel workers can be a bit snippy, especially at night. Overall I had no problems with The Talbott Hotel and I would stay at this here again if I were in the area a second time.*

The following review is a truthful review, created by a guest who stayed in the hotel:

> *My husband & I stayed at the Fitzpatrick in early June 2004 for my birthday-great hotel! Location provides easy walking to Navy Pier, Marshall Field, Michigan Avenue & John Hancock. Room seemed spacious even though its only about 300 sq ft. Lots of room in bathroom; comfortable bed; very quiet, upscale hotel. We would definitely stay here again!*

The dataset characteristics can be found in Table 4.1. The domain of review spam detection suffers from a lack of datasets representative of real world scenarios. To the best of our knowledge, this dataset is the closest publicly available text based spam review dataset representative of real world data. The dataset contains the text found in the review, the rating given, the domain the review belongs to, and the class label. For our purposes, the rating and domain information are not required as we are interested in using only text, thus, they are removed from the dataset. To create the feature set, the StringToWordVector filter in Weka is used. While StringToWordVector can output a variety of word vector representations, we elect to use a bag-of-words representation over TF-IDF as preliminary studies have found it to be more effective. The StringToWordVector filter in Weka returns the number of words specified in the WordsToKeep parameter based on word frequency. However,

if there is a tie in the word frequencies, it returns all the words with that specific frequency, causing more words to be returned than the specified number. To remedy this, we created the ExactStringToWordVector filter that returns the exact number of words specified in WordsToKeep parameter. ExactStringToWordVector samples randomly from the words that have equal frequency. If a value larger than the number of unique words in a document is specified for the WordsToKeep parameter, all unique words are extracted as features.

| Domain | Truthful | Spam | Total |
|---|---|---|---|
| Doctors | 200 | 356 | 556 |
| Hotels | 800 | 1080 | 1880 |
| Restaurants | 200 | 200 | 400 |
| Total | 1200 | 1636 | 2836 |

Table 4.1: dataset Characteristics

### 4.6.2 Base Learners

In our case studies, we use five base learners: Naïve Bayes (NB), Multinomial Naïve Bayes (MNB), Support Vector Machine (SVM), Logistic Regression (LR), and C4.5 decision tree (C4.5) in combination with the Boosting and Bagging ensemble techniques. We provide only a brief discussion of these learners here, since they are all described in Chapter 2. All models in this paper are built using the Weka data mining open-source software package [45] with default parameter values, unless otherwise specified. Note that any changes to default parameter values were applied when experimentation showed an overall improvement of the classification performance based on preliminary analysis.

Naïve Bayes [109] falls under the category of Bayesian learners. Naïve Bayes uses Bayes' theorem to approximate the posterior probability of an instance belonging to a class based on its feature values [132].

Multinomial Naïve Bayes [81] is a variant of the Naïve Bayes learner. The main

difference between NB and MNB is the way in which the probabilities are calculated. Naïve Bayes uses conditional probabilities of each feature to help determine classification status. In Multinomial Naïve Bayes for text classification, the instance (a document in this example) is assigned to the class which has the highest conditional probability of $P(C|X)$.

Support Vector Machine [60] constructs a hyper-plane that divides the instances into two groups. The data may be transformed via a kernel function into linearly separable spaces. The transformations allow for nonlinear boundaries to be formed around the data. For our models, the complexity constant c was set to 5.0 and the buildLogisticModels parameter set to true.

Logistic regression [65] seeks to find a linear relationship between the features and the class label. Logistic regression was chosen due to its simplicity and effectiveness.

The final learner, C4.5 decision tree [106], creates a tree based on the features that discriminate the most between classes. The decision tree process uses IG to determine a decrease in entropy when examining a certain feature. When constructing the trees, we implement "Laplace smoothing" and "no pruning" as this increased performance in previous studies [132].

### 4.6.3 Cross-Validation and Performance Metric

All models are trained using four runs of five-fold cross-validation. For each run of cross-validation the data is split into five separate folds. At any time, four of the five folds are used to train the data, while the last fold is used to evaluate the model. This process is repeated five times, varying the fold that is used for testing so every fold is used for testing once. We chose to use cross-validation over random sampling as all the data is used both to test and train the model in cross-validation, while only part of the data is used with a random sampling approach. Four runs of five-fold cross-validation are performed to reduce the bias due to an unlucky split in the data

when creating the folds. The results of the four runs of five-fold cross-validation are averaged for final results.

We elect to use the Area Under the receiver operator characteristic Curve (AUC) as a performance metric [132]. This metric is chosen as the ROC plots the performance of the model across all decision thresholds. Thus the area under this curve dictates the overall model performance across all thresholds [116]. The AUC is a graph of the False Positive Rate versus the True Positive Rate. The area under the curve shows performance of the model across all decision thresholds, thus the larger the area under the curve the better the performance of the model. This is not to be confused with the TBFS technique ROC, which uses AUC to rank features.

## 4.7 CLASSIFICATION AND FEATURE SELECTION

### 4.7.1 Non-Ensemble

We consider results for classification using the base learners and word frequency in the interest of having a baseline for comparison with ensemble methods and feature selection techniques. For this purpose, we examine how our five learners perform for a variety of features, as we vary the number of words extracted by changing the WordToKeep parameter which selects words based on frequency. In Figure 4.1, we see the resulting AUC values of the models generated using the base learners and varying levels of features based on frequency. We use a modified version of the StringToWordVector in the WEKA toolkit [45]. This modified version returns the number of words specified in the frequency selection option WordsToKeep. For example, if you specify 100 WordsToKeep, you will receive 100 distinct words based on the frequency of the word in the dataset. This allows for comparison between word frequency performance and performance of features chosen by a feature ranker. Word frequency selection is used with 46 feature subset sizes ranging from 100 to 20,000 to determine performance of word frequency as a feature ranker (20,000 is chosen as

the upper limit and encompasses the full feature set as the dataset contains less than 20,000 unique words).

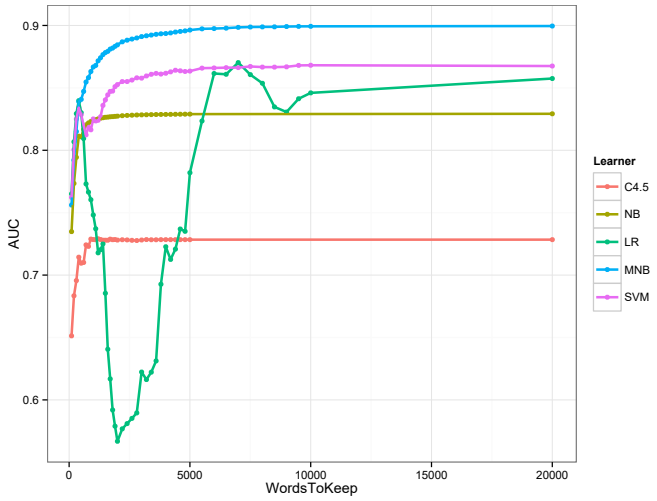



Figure 4.1: Classification with Word Frequency

In Figure 4.1, it can be seen that the best performing learner is Multinomial Naïve Bayes and its performance improves as WordsToKeep (the number of features used to train the classifier) increases; however, other learners do not follow this trend. C4.5 and NB level off at approximately 2,000 features. LR has the highest performance when using approximately 6,500 features. SVM performance levels off at approximately 10,000 features. To aid in the choice of the best learner and number of features, we conduct a two-factor ANalysis Of VAriance (ANOVA) to confirm both the choice of learner and the number of features that are significantly impacting classifier performance. The results of the ANOVA test are presented in Table 4.2 and show both factors and their interactions are significant.

| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| Subset Size | 45 | 4.34 | 0.10 | 127.73 | 0.0000 |
| Learner | 4 | 15.01 | 3.75 | 4976.04 | 0.0000 |
| Subset Size:Learner | 160 | 7.25 | 0.05 | 60.09 | 0.0000 |
| Residuals | 3990 | 3.01 | 0.00 | | |

Table 4.2: ANOVA for Plain Classification

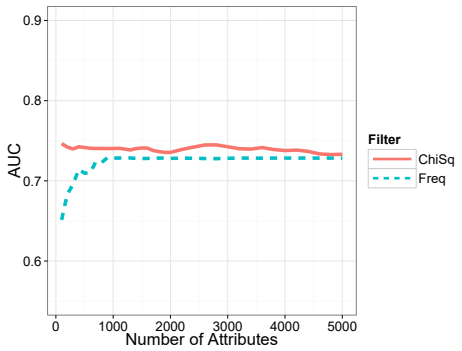
| | Learner | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | MNB | a | 0.878 | 0.034 |
| 2 | SVM | b | 0.847 | 0.027 |
| 3 | NB | c | 0.821 | 0.032 |
| 4 | LR | d | 0.734 | 0.107 |
| 5 | C4.5 | e | 0.722 | 0.025 |

Table 4.3: Tukey's HSD Test for the base learners using word frequency

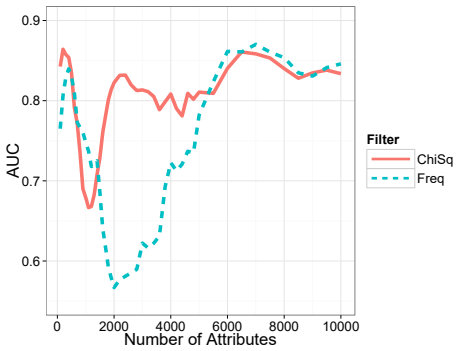
We conduct further tests using Tukey's Honestly Significant Difference (HSD) test to group factors by conducting pairwise similarity tests and determining if the choice between two levels of a factor is significant. First, we wish to determine which learner, without ensemble technique, is best. If the classifiers share the same letter, then the difference in performance is not statistically significant. Table 4.3 shows that Multinomial Naïve Bayes is the best learner across all levels. From Figure 4.1, we see that performance for Multinomial Naïve Bayes increases with number of features; however, from the figure alone we cannot determine when this increase is no longer significant. To determine the top performing subset size, an additional HSD test, found in Table 4.5, is conducted with feature subset sizes ranging from 100 to 20,000. We found there is no significant difference when using more than 900 features; however, the highest AUC (0.8995) is observed using the full feature set. Thus, Multinomial Naïve Bayes with the full-feature set will be used as the plain classifier in comparison with feature selection in the following section.

### 4.7.2 Feature Selection

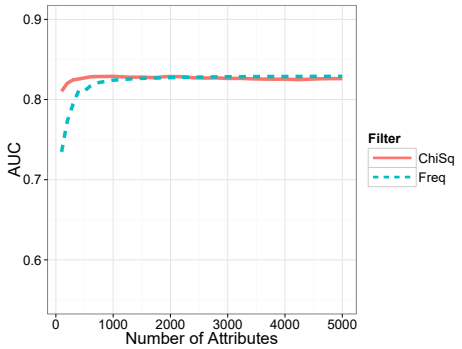
We would like to confirm the effects of feature selection on spam review detection to determine whether to use word frequency or feature selection as a baseline with which to compare ensemble techniques in Case Study II. To test performance of each feature selection technique, we use all base learners and subset sizes ranging from 100 to 1,000, as this was shown to be a good range of feature subset size [27]. As it is difficult to distinguish significant differences between the rankers, a Tukey's HSD



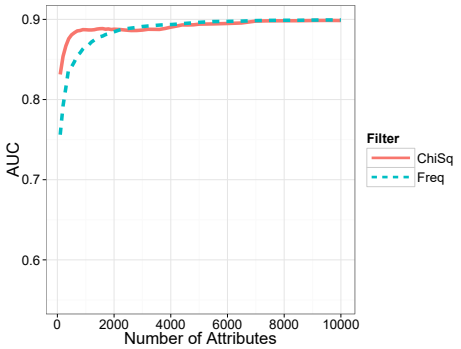

(a) C4.5, subset sizes from 100 to 5,000



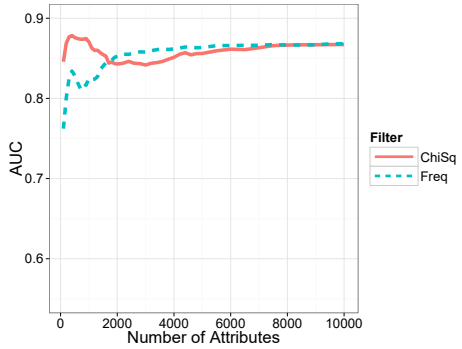

(b) Logistic Regression, subset sizes from 100 to 10,000




(c) Naïve Bayes, subset sizes from 100 to 5,000




(d) Multinomial Naïve Bayes, subset sizes from 100 to 10,000




(e) Support Vector Machines, subset sizes from 100 to 10,000

Figure 4.2: Average AUC score using feature selection

test, at a 95% confidence level, is used to conduct pairwise similarity tests to determine if the differences between two given rankers is statistically significant. Table 4.4 presents the results of this test, summarized by placing rankers into groups. Both Chi-Squared and Signal-to-Noise belong to the top group, while ROC, KS, MI, and PRC are all in the second group. The remaining rankers perform considerably worse than these 6 rankers.

| Feature Selection | Group | AUC | stdev |
|---|---|---|---|
| S2N | a | 0.822 | 0.061 |
| CS | a | 0.821 | 0.062 |
| ROC | b | 0.818 | 0.054 |
| KS | b | 0.818 | 0.054 |
| MI | b | 0.817 | 0.054 |
| PRC | b | 0.816 | 0.054 |
| SAM | c | 0.661 | 0.060 |
| WRS | d | 0.614 | 0.050 |
| GI | e | 0.605 | 0.072 |
| PR | e | 0.604 | 0.070 |

Table 4.4: Tukey's HSD Test of feature selection techniques across all classifiers with feature subset sizes from 100 to 1,000

In Figure 4.2, classification results for Chi-Squared are compared against those using word frequency. Chi-Squared is chosen over Signal-to-Noise as it is more commonly used and the results for both Signal-to-Noise and Chi-Squared are similar across all classifiers as shown in Table 4.4. We do not provide the results for every feature selection technique against every learner, as the emphasis of these experiments are on ensemble methods with feature selection and not feature selection alone. We see that as subset size increases, the difference between Chi-Squared and word frequency decreases. Subfigures a and c show performance for subset sizes up to 5,000, while subfigures b, d, and e show performance up to 10,000. This subset size difference is due to performance; for C4.5 and NB there is no difference in performance between word frequency and Chi-Squared past 5,000 features, while for LR, MNB, and SVM there is a difference up to 10,000 features. Figure 4.2 shows word frequency performs

| | WordsToKeep | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | 20000 | a | 0.900 | 0.013 |
| 2 | 10000 | a | 0.899 | 0.013 |
| 3 | 9500 | a | 0.899 | 0.014 |
| 4 | 9000 | a | 0.899 | 0.014 |
| 5 | 8500 | a | 0.899 | 0.013 |
| 6 | 8000 | a | 0.899 | 0.013 |
| 7 | 7500 | a | 0.899 | 0.013 |
| 8 | 7000 | a | 0.898 | 0.013 |
| 9 | 6500 | a | 0.898 | 0.013 |
| 10 | 6000 | a | 0.898 | 0.014 |
| 11 | 5500 | a | 0.897 | 0.014 |
| 12 | 5000 | a | 0.896 | 0.014 |
| 13 | 4800 | a | 0.896 | 0.014 |
| 14 | 4600 | a | 0.895 | 0.014 |
| 15 | 4400 | ab | 0.895 | 0.014 |
| 16 | 4200 | ab | 0.894 | 0.014 |
| 17 | 4000 | ab | 0.894 | 0.015 |
| 18 | 3800 | ab | 0.893 | 0.015 |
| 19 | 3600 | ab | 0.893 | 0.015 |
| 20 | 3400 | ab | 0.892 | 0.015 |
| 21 | 3200 | ab | 0.892 | 0.016 |
| 22 | 3000 | abc | 0.891 | 0.016 |
| 23 | 2800 | abc | 0.890 | 0.016 |
| 24 | 2600 | abc | 0.889 | 0.016 |
| 25 | 2400 | abc | 0.888 | 0.016 |
| 26 | 2200 | abc | 0.887 | 0.016 |
| 27 | 2000 | abc | 0.884 | 0.017 |
| 28 | 1900 | abcd | 0.883 | 0.016 |
| 29 | 1800 | abcd | 0.882 | 0.017 |
| 30 | 1700 | abcd | 0.881 | 0.016 |
| 31 | 1600 | abcd | 0.879 | 0.017 |
| 32 | 1500 | abcd | 0.878 | 0.017 |
| 33 | 1400 | abcde | 0.877 | 0.019 |
| 34 | 1300 | abcdef | 0.874 | 0.019 |
| 35 | 1200 | abcdef | 0.872 | 0.018 |
| 36 | 1100 | abcdef | 0.868 | 0.018 |
| 37 | 1000 | abcdef | 0.867 | 0.018 |
| 38 | 900 | abcdef | 0.863 | 0.018 |
| 39 | 800 | bcdef | 0.858 | 0.018 |
| 40 | 700 | cdef | 0.855 | 0.021 |
| 41 | 600 | defg | 0.847 | 0.021 |
| 42 | 500 | efg | 0.841 | 0.022 |
| 43 | 400 | fg | 0.839 | 0.022 |
| 44 | 300 | gh | 0.815 | 0.023 |
| 45 | 200 | hi | 0.792 | 0.025 |
| 46 | 100 | i | 0.756 | 0.025 |

Table 4.5: Tukey's HSD Test for WordsToKeep with MNB

as well as a feature ranker when working with larger subset sizes. As more features (words) are used, the overlap in feature sets between feature ranker and word frequency increases, leading to similar performance. It is important to note that, with either Chi-Squared or word frequency, Multinomial Naïve Bayes has the highest AUC scores for most subset sizes.

Additionally, Multinomial Naïve Bayes' performance increases with the number of features when using Chi-Squared, though these changes are not significant. The difference between Chi-Squared and word frequency is minimal when using larger subset sizes. In the case study on algorithms combining ensemble and feature selection techniques presented in the following section, Signal-to-Noise, Chi-Squared, and Mutual Information are chosen as the feature selection techniques to be embedded within ensemble learners. We elect to use S2N and CS due to their performances, as both techniques fall under group 'a'. MI is selected because of it's performance in other social media/text domains, such as sentiment analysis [101]. We consider this to be a similar domain as both are considered social media, both are online and publicly viewable, both are user-generated, and both contain short texts. This allows us to explore the effects of Select-Boost and Select-Bagging on top scoring rankers and a group 'b' ranker, resulting in a more robust experiment.

## 4.8 ENSEMBLE TECHNIQUES

This section presents results related to the ensemble techniques and the combination of ensemble techniques and feature selection. We first present results for Boosting and Bagging, then we present results for RF, Select-Boost, and Select-Bagging.

### 4.8.1 Boosting and Bagging

Results for our experiments utilizing Boosting and Bagging can be found in Table 4.6. From Table 4.6, we observe the top performing combination is MNB with Boosting.

| | Learner | Boosting | Bagging |
|---|---|---|---|
| 1 | MNB | 0.902 | 0.900 |
| 2 | SVM | 0.879 | 0.889 |
| 3 | LR | 0.876 | 0.888 |
| 4 | NB | 0.874 | 0.833 |
| 5 | C4.5 | 0.827 | 0.823 |

Table 4.6: AUC results for Boosting and Bagging across five base learners and all feature subset sizes

Bagging shows higher AUCs for three of the five learners (C4.5, LR, SVM), while Boosting produces higher AUCs for the remaining two (MNB, NB).

To determine whether Boosting or Bagging statistically improves over the base learners, we perform a two-factor ANOVA and a Tukey's HSD test. Table 4.7 presents both the ANOVA and Tukey's HSD results for Boosting, Bagging and the base learners. The two-factor ANOVA test is presented in Table 4.7-A. The factors for this ANOVA are ensemble techniques and base learners. Ensemble techniques include Boosting and Bagging, while the base learners are NB, MNB, SVM, C4.5, and LR. The ANOVA results show there is a significant difference between base learners, ensemble methods, and the interaction between them.

From Table 4.7-B, we see the average AUC and Tukey's groups across all experiments for Boosting and Bagging. There are six distinct groupings: 'a' through 'f'. These results indicate that ensemble techniques significantly increase performance of SVM, LR, NB, and C4.5. Bagging significantly increases performance over the base SVM and LR learners, while Boosting significantly increases performance of LR, NB, and C4.5. We see that the best base learner is MNB. Moreover, the differences between MNB with and without ensemble techniques are not significant; however, not using an ensemble technique is faster and less computationally costly. Boosting with MNB does produce the highest AUC and the lowest standard deviation.

| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| Learner | 4 | 0.41 | 0.10 | 345.73 | 0.0000 |
| Ensemble | 2 | 0.07 | 0.04 | 123.41 | 0.0000 |
| Learner:Ensemble | 8 | 0.09 | 0.01 | 37.31 | 0.0000 |
| Residuals | 275 | 0.08 | 0.00 | | |

(A) ANOVA Results for Boosting and Bagging

| | Model | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | MNB:Boosting | a | 0.902 | 0.009 |
| 2 | MNB:Bagging | a | 0.900 | 0.014 |
| 3 | MNB:None | a | 0.900 | 0.013 |
| 4 | SVM:Bagging | ab | 0.889 | 0.011 |
| 5 | LR:Bagging | ab | 0.888 | 0.011 |
| 6 | SVM:Boosting | bc | 0.879 | 0.012 |
| 7 | LR:Boosting | bcd | 0.876 | 0.015 |
| 8 | NB:Boosting | bcd | 0.874 | 0.011 |
| 9 | SVM:None | cd | 0.867 | 0.014 |
| 10 | LR:None | d | 0.857 | 0.027 |
| 11 | NB:Bagging | e | 0.833 | 0.028 |
| 12 | NB:None | e | 0.829 | 0.026 |
| 13 | C4.5:Boosting | e | 0.827 | 0.013 |
| 14 | C4.5:Bagging | e | 0.823 | 0.019 |
| 15 | C4.5:None | f | 0.728 | 0.018 |

(B) Tukey's HSD test AUC values for Boosting and Bagging

Table 4.7: ANOVA and Tukey's HSD for Boosting and Bagging

### 4.8.2 Random Forest

Random Forest provides a different ensemble approach when compared to Bagging and Boosting and performs its own internal random feature selection; thus, the classifier is trained using the full feature set. This ensemble technique does not directly use feature rankers, but does perform random feature subspace selection at every node within a tree. Presented in Table 4.8, a Tukey's HSD test includes AUC scores for each of the three RF tree sizes and for the C4.5 learner. The result for the C4.5 learner is included as a baseline for comparison, as it is a single decision tree. All RF tree sizes significantly outperform the C4.5 baseline. Moreover, the results show that RF250 and RF500 perform significantly better than RF100, but there are no significant differences between RF250 and RF500. If we compare results in Table 4.8,

we see RF500 generates the highest AUC score.

| | Model | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | RF500 | a | 0.907 | 0.016 |
| 2 | RF250 | a | 0.899 | 0.017 |
| 3 | RF100 | b | 0.876 | 0.015 |
| 4 | C4.5 | c | 0.729 | 0.020 |

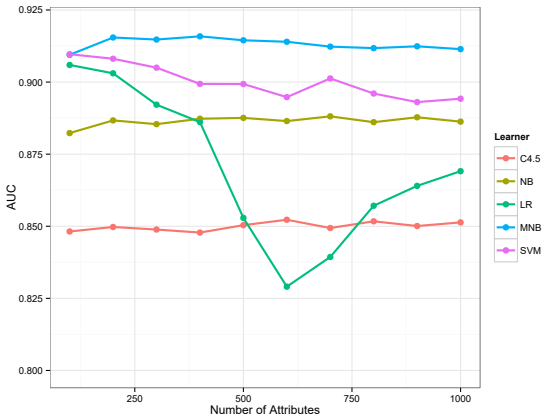
Table 4.8: Tukey's HSD Test for Random Forest and C4.5

### 4.8.3 Select-Boost and Select-Bagging

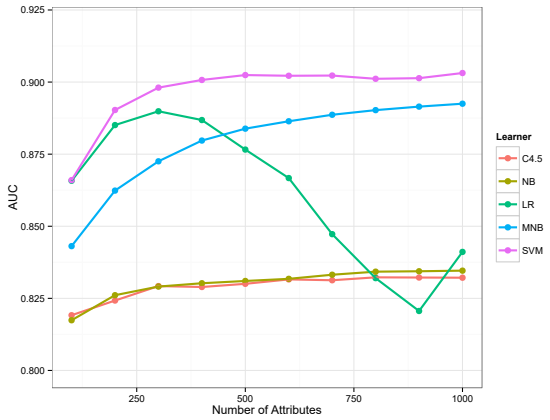
Figure 4.3 presents results for ensemble classifiers trained using Select-Boost and Select-Bagging grouped by feature ranker (Signal-to-Noise, Chi-Squared, and Mutual Information) and ensemble method (Select-Bagging and Select-Boosting). In each subfigure, AUC for each learner is plotted against the number of attributes selected.

From the subfigures, there are several important trends that warrant further discussion. First, Select-Boost yields higher performance than Select-Bagging for all rankers. We have seen this trend in previous work, where we found Select-Boost outperforms Select-Bagging in the text classification domain [103]. Second, as seen in Figure 4.3, Multinomial Naïve Bayes produces the highest AUC for any base learner using Select-Boost.

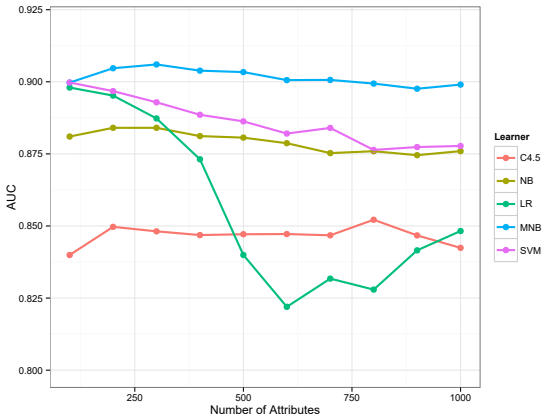
We are interested in which factors (base learner, feature selection technique, ensemble approach, and subset size) are significant, thus we conduct a four-factor ANOVA. Results are shown in Table 4.10 and indicate that all four factors and many of their interactions are significant. To investigate each factor: base learner, feature selection (FS in the table), ensemble method (ensemble in the table), and subset size, we conduct Tukey's HSD tests on each factor. Our HSD test for base learners, presented in Table 4.9-A, shows that Multinomial Naïve Bayes and SVM are the top performing learners and the difference between them and other learners is significant; however, the difference between MNB and SVM is not significant. Table 4.9-B shows



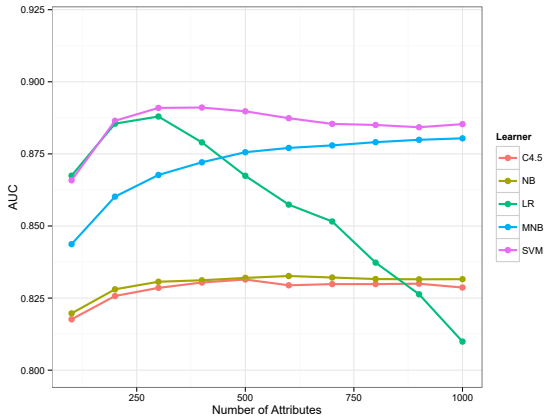

(a) Chi-Squared with Select-Boosting



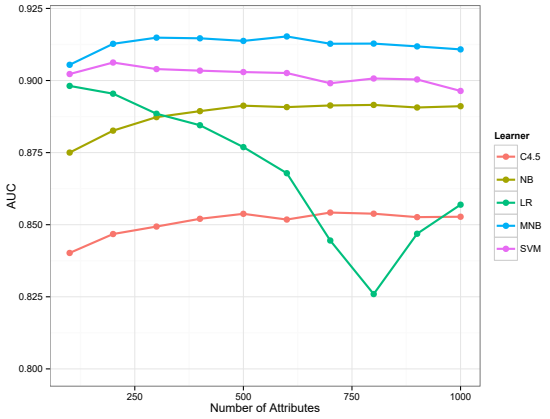

(b) Chi-Squared with Select-Bagging



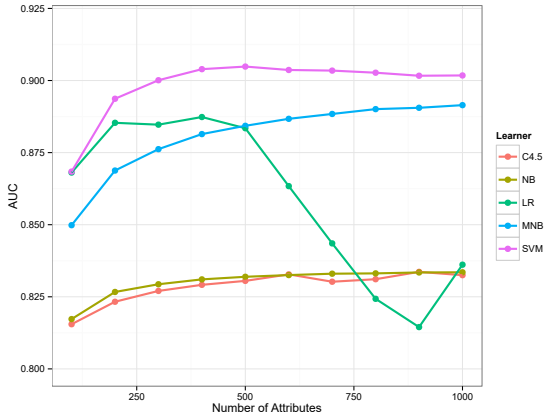

(c) Mutual Information with Select-Boosting




(d) Mutual Information with Select-Bagging




(e) Signal-to-Noise with Select-Boosting




(f) Signal-to-Noise with Select-Bagging

Figure 4.3: Results for Select-Boost and Select-Bagging

| | Learner | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | SVM | a | 0.895 | 0.015 |
| 2 | MNB | a | 0.893 | 0.024 |
| 3 | LR | b | 0.862 | 0.029 |
| 4 | NB | c | 0.857 | 0.034 |
| 5 | C4.5 | d | 0.839 | 0.020 |

(A) Tukey's HSD test for Base Learners

| | Feature Selection | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | CS | a | 0.872 | 0.034 |
| 2 | S2N | a | 0.872 | 0.034 |
| 3 | MI | b | 0.864 | 0.031 |

(B) Tukey's HSD test for Feature Rankers

| | Ensemble | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | Select-Boost | a | 0.881 | 0.028 |
| 2 | Select-Bagging | b | 0.858 | 0.034 |

(C) Tukey's HSD test for Ensemble Approaches

Table 4.9: Tukey's HSD for Base Learners, Feature Rankers, and Ensemble Techniques

that Chi-Squared or Signal-to-Noise should be chosen over Mutual Information. As the difference between using Signal-to-Noise and Chi-Squared is not significant and Chi-Square is more readily available, we will be using Chi-Squared for future experiments. Table 4.9-C shows Select-Boost performs significantly better than Select-Bagging. These tests indicate the best performing classifiers are trained with Select-Boost using Chi-Squared, or Signal-to-Noise, as a ranker and MNB, or SVM, as a base learner.

In Figure 4.3, we see that the AUC values for Select-Boost are higher than Select-Bagging and Table 4.9-C shows this difference is significant. Furthermore, we observe higher AUC values when using MNB as the base learner within Select-Boost, thus we elect to examine this combination further. There is little change in AUC for Multinomial Naïve Bayes, across feature subset sizes with Select-Boost. As this is our best performing learner, further investigation is warranted to determine the optimal subset size. A Tukey's HSD test will determine if feature subset size significantly im-

| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| Base Learner | 4 | 2.80 | 0.70 | 2851.65 | 0.0000 |
| Subset Size | 9 | 0.09 | 0.01 | 42.25 | 0.0000 |
| FS | 2 | 0.08 | 0.04 | 155.24 | 0.0000 |
| Ensemble | 1 | 0.80 | 0.80 | 3253.94 | 0.0000 |
| Base Learner:Subset Size | 36 | 0.58 | 0.02 | 65.14 | 0.0000 |
| Base Learner:FS | 8 | 0.02 | 0.00 | 11.96 | 0.0000 |
| Subset Size:FS | 18 | 0.02 | 0.00 | 4.93 | 0.0000 |
| Base Learner:Ensemble | 4 | 0.53 | 0.13 | 544.35 | 0.0000 |
| Subset Size:Ensemble | 9 | 0.10 | 0.01 | 44.91 | 0.0000 |
| FS:Ensemble | 2 | 0.01 | 0.00 | 17.89 | 0.0000 |
| Base Learner:Subset Size:FS | 72 | 0.03 | 0.00 | 1.87 | 0.0000 |
| Base Learner:Subset Size:Ensemble | 36 | 0.11 | 0.00 | 12.92 | 0.0000 |
| Base Learner:FS:Ensemble | 8 | 0.00 | 0.00 | 1.84 | 0.0656 |
| Subset Size:FS:Ensemble | 18 | 0.01 | 0.00 | 1.83 | 0.0176 |
| Base Learner:Subset Size:FS:Ensemble | 72 | 0.02 | 0.00 | 1.15 | 0.1886 |
| Residuals | 5700 | 1.40 | 0.00 | | |

Table 4.10: ANOVA for Ensemble Classifiers

pacts classifier performance with MNB as the learner and Chi-Squared as the ranker. Results, presented in Table 4.11, show that there are no significant differences as all subset sizes are in the same group; however, 400 features resulted in the highest observed AUC.

## 4.9 DISCUSSION AND COMPARISONS

The results presented in the previous sections indicate that Multinomial Naïve Bayes is the best performing learner, when using no ensemble technique, and one of the top base learners when using the Select-Boosting ensemble framework. Thus, it is of interest to compare how this learning algorithm performs, both with and without the Select-Boost framework, and also how it compares to the best performing models generated with Random Forest. Additionally, we have yet to compare ensemble algorithms with no feature selection and ensemble methods with feature selection. Thus, we compare RF250 and RF500, MNB as a classifier, and Select-Boost with Chi-Squared and 400 features, Bagging, and Boosting using MNB.

| | Feature Set Size | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | 400 | a | 0.916 | 0.011 |
| 2 | 200 | a | 0.915 | 0.010 |
| 3 | 300 | a | 0.915 | 0.010 |
| 4 | 500 | a | 0.914 | 0.010 |
| 5 | 600 | a | 0.914 | 0.011 |
| 6 | 900 | a | 0.912 | 0.008 |
| 7 | 700 | a | 0.912 | 0.009 |
| 8 | 800 | a | 0.912 | 0.009 |
| 9 | 1000 | a | 0.911 | 0.012 |
| 10 | 100 | a | 0.909 | 0.010 |

Table 4.11: Tukey's HSD Test for feature subset size with Multinomial Naïve Bayes and Select Boost with Chi-Squared

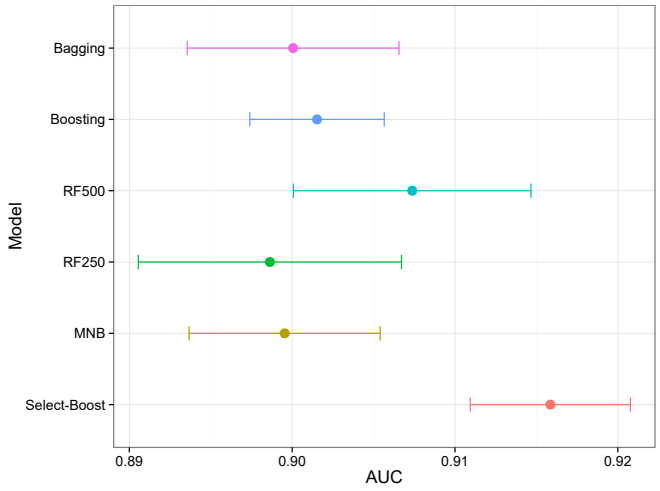



Figure 4.4: Visualization of Tukey's HSD for Models

Table 4.12-A presents a one-factor ANOVA result (choice of model) for the previously stated classifiers. In the case of this ANOVA, model encompasses the combination of base learner, feature selection technique and ensemble technique. From Table 4.12-A, we note that the choice of model is significant, thus, we use a Tukey's HSD test to determine the difference between the six rankers. Table 4.12-B presents a Tukey's HSD test showcasing the AUC values for each of these techniques grouped by pairwise similarity. Figure 4.4 depicts the results found in Table 4.12-B where each each model is shown along with their confidence interval. If there is no overlap between two models, then their averages are significantly different. We observe that

| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| Model | 5 | 0.00 | 0.00 | 4.94 | 0.0004 |
| Residuals | 114 | 0.02 | 0.00 | | |

(A) ANOVA for RF500 and RF250, and Select-Boost, Bagging, Boosting and plain classification using MNB

| | Model | Group | AUC | stdev |
|---|---|---|---|---|
| 1 | Select-Boost | a | 0.916 | 0.011 |
| 2 | RF500 | ab | 0.907 | 0.016 |
| 3 | Boosting | b | 0.902 | 0.009 |
| 4 | Bagging | b | 0.900 | 0.014 |
| 5 | MNB | b | 0.900 | 0.013 |
| 6 | RF250 | b | 0.899 | 0.017 |

(B) Tukey's HSD Test for RF500 and RF250, and Select-Boost, Bagging, Boosting and plain classification using MNB

Table 4.12: ANOVA and Tukey's Results for Models

there are two groups 'a' and 'b'. The bottom group, 'b', contains RF250, MNB as a classifier, and Boosting and Bagging using MNB. Group 'a' contains the Select-Boost algorithm with the previously listed components, implying Select-Boost performs significantly better than RF250, Bagging, Boosting, and MNB with no ensemble technique. RF500 is in group 'ab' indicating no statistically significant difference from the other classifiers.

Our results show a combination of Select-Boost, Multinomial Naïve Bayes, and, either Chi-Squared or Signal-to-Noise, significantly outperforms all methods except RF500. However, it is interesting to note that while Select-Boost improves performance significantly, the components which create the Select-Boost framework do not increase performance when applied separately.

Feature selection techniques do not improve classification over the full dataset, in fact, feature selection shows degraded performance with lower feature set sizes. This could be due to properties specific to this dataset. The dataset contains reviews from three domains, features (words) that determine spam in one domain may not be the same for a different domain. For example, the word 'small' may be positive in respect

to cell phones, but negative in respect to hotel rooms. Thus, the reduced feature set may not be able to discriminate between truthful and untruthful reviews across all three domains, leading to lower classification performance.

We also observe that, while boosting increases the AUC of our MNB model, the difference is not significant. It is likely that Boosting and Bagging show no improvement over the base MNB model because they do not address the high dimensionality of the data. When we compare Select-Boost to ensemble techniques using feature selection (RF and Select-Bagging), we see closer performance. Select-Boost still outperforms Select-Bagging, RF100, and RF250. However, the difference between RF500 and Select-Boost is not significant. Overall, we recommend Select-Boost over RF500, since Select-Boost has less variance than RF500, most likely due the independent trees, and is faster than RF500.

Select-Boost significantly outperforms Select-Bagging, implying an aspect of the Select-Boost's iterative framework significantly affects performance. This performance difference may be due to a combination of the re-weighing step found in Boosting and feature selection done in Select-Boost. The Select-Boost framework re-weights instances at every iteration, with greater weight being placed on previously misclassified instances, by resampling (with replacement) from the original dataset with probabilities of selecting instances determined by their weight. As feature selection is performed during every iteration, after the resampling step (where misclassified instances are more likely to be present), the weights assigned to the instances directly affect the features chosen by our feature rankers. Every iteration allows for more focus on misclassified instances and their features, which leads to a more optimized feature set as the Select-Boosting framework progresses. Thus, Select-Boost is more effective than Select-Bagging, which performs feature selection on randomly sampled data bootstraps, and offers a significant improvement in classification performance, whereas Boosting and feature selection, individually, do not.

## 4.10 CHAPTER SUMMARY

In this chapter, we determined the performance of ensemble classifiers, classification using feature selection, and the combination of both on spam review detection. The performance of Select-Boost and Select-Bagging was evaluated against Random Forest, feature selection, Boosting, and Bagging. Select-Boost and Select-Bagging were employed with five learners, three feature selection techniques, and subset sizes ranging from 100 to 1,000. Random Forest was done using three tree sizes: 100, 250, and 500. The effects of ensemble classifiers on spam review detection had previously not been explored.

The results show that feature selection has degradative effects on classification performance when compared to using the full feature set. The performance of the model steadily increases as feature subset size increases, although this increase is no longer significant above 900 features when using word frequency. Ensemble techniques, without feature selection, had no significant effect on classification performance for spam detection. However, using Select Boost, a combination of feature selection and boosting, significantly increased classification performance. Select-Boost significantly outperformed the base MNB classifier, Boosting, Bagging, Select-Bagging, RF100, and RF250. Select-Boost provides a higher AUC value than all other learners; however, the difference between Select-Boost and RF500 was not significant.

We recommend the use of Select-Boost over Select-Bagging. The re-weighting step in Select-Boost is crucial in generating an optimized, reduced feature set, allowing for significantly better performance, since each iteration selects a new subset of features that aid in the correct classification of previously misclassified instances. Select-Boost should be chosen over RF500 as it results in a higher AUC, although not significantly different, and is faster. We found using the combination of Select-Boost, Multinomial Naïve Bayes, Chi-Squared, and 400 features has the highest AUC when detecting spam reviews, although Chi-Squared and Signal-to-Noise may be interchanged. There

is no significant difference in performance between feature subset sizes when using feature selection within Select-Boost, thus, while the highest AUC is observed when selecting 400 features, computational resources can be preserved by using a smaller number of features without significantly degrading classification performance.

# CHAPTER 5
# MINING SOCIAL MEDIA FOR DETERMINING ELECTION OUTCOMES

## 5.1 BACKGROUND AND MOTIVATION

The President of the United States (U.S.) is considered one of the most powerful and influential people in the world; however, attaining this position is a rigorous and complicated process. Under the U.S. democratic process, every four years the U.S. holds a general election to determine who will become the new president. During this time period, media coverage and commercial ads focus mostly on presidential campaigns, generating political interest and allowing voters to make informed decisions on the candidates. It is crucial during this time for voters to keep updated with current events surrounding the candidates, both political and personal. Many voters take into consideration not only policy choices but also moral standings, as voters desire a candidate who aligns with their beliefs.

U.S. politics primarily revolves around two political parties: the Republicans and the Democrats. Both hold their own primaries to determine their candidate for the general election. On election day, every eligible U.S. citizen over the age of 18 is allotted a vote and can vote for any candidate participating in the race. While most countries use popular vote to determine the victor of an election, in the U.S popular vote does not elect the president. Instead, that decision is made through the electoral college process. The electoral college is a process established in the U.S. Constitution as to allow influence both from Congress and the popular vote [11]. Each state is allocated a number of electors equal to the sum of its number of U.S. senators

and representatives. The number of senators is always two, while the number of representatives can vary depending on the state's population at the time of the most recent official U.S. Census. Since 1913, there are 538 electors and the candidate is declared the victor when they have a majority of those electors. Electoral votes are awarded on the basis of the popular vote in each state [5].

States operate under the winner-take-all approach, where the candidate with the highest popular vote is awarded all the electors allotted to that state, with the exception of Maine and Nebraska, who split electoral votes based on district popular vote. Only the electoral votes matter towards the election of a candidate; these votes normally follow the popular vote outcome in the state; however, electors can cast what are called faithless votes. Votes are considered faithless when the elector does not follow the will of the people. This does not occur often, with only eight faithless elector votes across eight elections since 1948 [121]. Typically, states are classified based on the popularity of the candidates and previous voting records, states that have a track record of voting for Republican candidates are considered right leaning, states who previously voted for Democrats are left leaning, and states which have no clear bias are considered swing states. Candidates focus their campaigns on the swing states, since the allocation of time and resources into favored states (left or right) may not sway enough voters to turn the tide in their favor.

The 2016 general election included many participants; however, we choose to focus on the two front runners: the Republican candidate Donald Trump and the Democratic candidate Hillary Clinton. Throughout the election period, there were three presidential debates, in addition to continuous news coverage on candidate political and personal lives. A voter's decision depends on many factors, ranging from policy choices to personal beliefs. Due to this, the most effective method of predicting election outcome is through public opinion polling. Polling refers to the process of asking a set list of questions to a subset of the population in order to

gauge public opinion. As with every election, many different polls were conducted at varying levels by different organizations to determine public opinion on candidates [1]. This election is of particular interest as it was one of the most popular elections, with 139 million Americans participating and record-breaking turnout on election day [3]. While polling did correctly estimate the popular vote, many of these polls incorrectly predicted the election outcome. This is because popular vote does not indicate the victor, as the electoral college is the only method of determining the president-elect. Polling is also conducted on a state-level, to determine the candidate's standings for that state. State-based polling and the allocation of their electors based on polling results can serve as a form of prediction for the election outcome. Previous studies have found that social media is an efficient data source for polling the popular vote in the 2009 German [124] and the 2010 U.K. elections [17].

Social media platforms have risen in popularity over the last decade, with Facebook and Twitter becoming two of the most popular choices in the U.S. Although Facebook is the company that has become synonymous with the term social media, Twitter is the social media platform most used by politicians to release information and communicate with their constituents. Both candidates in the 2016 election consistently took to Twitter to disseminate information related to their campaigns and to deter voters from voting for the other candidates. This is because Twitter allows users to publish tweets, which are short 280 (formerly 140) character micro-blog posts and view other users' tweets [2]. Politicians were not the only ones who took to social media; conventional media outlets released articles and stories on Twitter and many people took to Twitter to argue their preferred candidate's position or debase their candidate's opponents. These tweets can be used for sentiment analysis, as they will convey the opinion of the author. Sentiment has been shown to be a strong predictor for many domains, such as box office results [82], product sales performance [76], stock price prediction [118], and modeling public opinion [15]. Sentiment analysis is

the process of analyzing a document and labeling the document with a certain emotion conveyed by the text [89]. Many emotions can be used for classification; however, we opt for only labeling using positive and negative emotion. For example, the tweet 'I love Trump' would be considered positive emotion, and 'I hate Trump' would be considered negative emotion. Labeling sentiment in tweets would help us to better comprehend the views of the public in regards to each candidate.

Although most accounts on Twitter are controlled by humans, automated or fake social bot accounts are still present. Social bots tend to behave differently from human controlled accounts; having a larger number of retweets and networks consisting of mostly bot connections [29]. These social bots existed during the 2016 U.S. election and spread misinformation regarding the candidates. A study claims anywhere from 64% to 79% of Donald Trump's followers are believed to be fake [86]. Their objective is to sway voter decisions or breed doubt and discourse among voters. These bots can affect perception, making it seem as if the public views regarding candidates are exacerbated or even opposite the actual views. In addition, the presence of social bots hinders sentiment analysis as it can skew emotion in different directions.

This chapter makes the following contributions:

- We determine the effectiveness of social media, specifically Twitter, at polling and predicting the 2016 presidential election.
- Use tweet sentiment and volume as metrics for polling/predicting the U.S. election. This method has not previously been applied to the U.S. election.
- We examine the data at two levels: national and state. The national level refers to vote count per candidate without the inclusion of the electoral college process, while the state level, tweets are grouped based on location and the electoral college process is factored in.
- We explore the influence social bots had on public opinion of candidates on

social media during the 2016 election and their effects on polling the election.

- We identify and remove social bot accounts from the dataset. Furthermore, we limit users to only one tweet to better represent the real-world constraints related to voting.

The remainder of this chapter is organized as follows. Section 5.2 presents previous works related to social media mining in the political domain and social bot detection. Section 5.3 presents our methodology, including dataset, neural network architecture, and approaches. Sections 5.4 and 5.5 presents our results and discussions for the first and second Case Studies, respectively. Finally, Section 5.6 summarizes this chapter.

## 5.2 RELATED WORKS

In politics, understanding public opinion plays a large role in anticipating election outcomes. Since campaigns have begun to utilize social media, more candidates are focused on connecting to their constituents via social networking. Research into the effects of social media on political discourse has become necessary due to this rising trend of social media politics. In addition, dissemination of propaganda has been exacerbated in this modern era of digital media. Companies such as Devumi sell Twitter follower plans, which will artificially increase your follower count by having these fake accounts follow you [92]. As social bots rise in use, a method for their detection becomes increasingly necessary as well.

Twitter is the platform of choice for communication between those holding political office and the general public [17]. A study by Boutet et al. proposed an algorithm for determining the political party of a Twitter user during the 2010 United Kingdom (UK) election cycle [17]. The UK political system differs from the U.S. system as it contains three main parties: Labour, Conservative and Liberal Democrat. A total of 419 trending topics were chosen for tweet collection. If a topic had more than 10,000

related tweets, posts for that topic were collected and added to the dataset. The authors manually labeled self-identified party members for each party as a ground truth dataset. A Support Vector Machine (SVM) and two Bayesian classification methods were proposed for the identification of user party. Bayesian-Retweet and Bayesian-Volume were the two Bayesian models proposed. The former detects communities of users using a label propagation method, while the latter counts references to a party, or party members, and assigns the most frequently referenced party as the label. The SVM model was trained using six user features. Their results showed the Bayesian-Volume classifier significantly outperformed the remaining classifiers used. The UK election has a three-party system, while the U.S. national election has only two main parties. The three-party system changes the problem from a binary classification to a multi-classification problem and this alteration may show different results for the U.S. election.

A study by Bermingham and Smeaton explores the effects of Twitter on the 2011 Irish election cycle [13]. During this study, the authors developed "Twitter Tracker", a software allowing them to tap into content on Twitter pertaining to the election. A total of 32,578 tweets relevant to the five main parties were collected by searching for the party names, abbreviations, and the election hash-tag. The authors compare their Twitter-based predictions against nine polls collected during the election. The sets of tweets used were separated into time-based, sample size-based, cumulative, and manual. Time-based tweets were then grouped into three subsets: the most recent 24 hours, 3 days, and 7 days. The sample size included four subsets: the most recent 1,000, 2,000, 5,000 or 10,000 tweets. Nine annotators were trained before the date of the election and used for labeling sentiment in tweets. The tweets used to train these annotators were taken from different points in time as to develop a diverse training corpus. Authors found the addition of sentiment analysis to increase the accuracy of their predictions.

Similar to the above study, Tumasjan et al. explored the power of Twitter and tweet sentiment in the 2009 German national parliament election [124]. The authors collected 104,003 tweets in the weeks leading up to the election. The Linguistic Inquiry and Word Count 2007 software was used to extract sentiment from tweets. Users were categorized based on their frequency of tweets. Authors found that political discussions are usually driven by a small number of heavy users (80+ tweets). This study also finds that the volume of tweets mirrored the results of the elections.

A study by Anuta et al. focused on exploring the accuracy of polls in the U.S. and comparing them to the accuracy of Twitter as a polling source [9]. Five polls from eight different sources were gathered between January and the election day. The study was split into two categories, popular and state votes. Polls were adjusted to represent a two-party system and sentiment was added to tweets using the Vader python package [40]. For the state votes, nine states were chosen and only two polling sources were used. Of these nine, three were Clinton favored, three were Trump favored, and the final three were swing states. Tweet collection was done using a previously created dataset of tweet IDs. The dataset is limited to users in the tweet ID dataset, which may not be representative of the total population or the state population. The authors present time series data of the bias for each poll and tweets and show that as the election date draws nearer, the bias lowers. They found twitter yielded a popular vote bias of 3.4% in favor of Trump (higher than any poll) and, across all nine states, an average bias of 2.4% towards Trump.

Many studies have explored methods of social bot identification using methods such as differences in retweets and follower counts [122], sentiment [31], network features [123], and natural language processing [25].

Ferrara et al. [37] describe new iterations of social bots and their uses in their 2016 publication. The authors note that bots are used in malicious ways to deceive and defame humans. This has direct consequences for the political domain, as bots can

be used to directly debase candidates or spread false information. The authors make a point to explain that the current level of sophistication of social bots threatens multiple social domains. They emphasize that if social bots are considered the puppets, finding their "masters" is of utmost importance or the spread of false information will only continue to increase.

A study by Davis et al. [29] proposed the BotOrNot system (now Botometer) for evaluation of social bots on Twitter. Their proposed system utilizes the Twitter screen name to retrieve the account's recent activities, then the system computes and returns a bot-likelihood score. At the time of publication, the system stored approximately 900,000 unique user account classifications and found that approximately 87% of users had a bot score under 0.5. The bot-likelihood score is computed via an ensemble classification algorithm, specifically Random Forest (RF). The system generates more than 1,000 features using available data [29]. Features can be grouped into 6 classes: Network, User, Friends, Temporal, Content, and Sentiment. These features are used to train seven different classifiers, six for the corresponding feature groups and one for overall score. The RF model was trained using a dataset of more than 5.6 million tweets and yielded an Area Under the receiver operator characteristic Curve (AUC) of 0.95.

Ratkiewicz et al. [107] propose a system to analyze the diffusion of information in social media during an astroturf political campaign. Using their Klatsch analytical framework, they analyze behavior of users and diffusion of ideas. Their system utilizes network and content features from Twitter. The system monitors the Twitter data stream, detects relevant memes, collects the tweets that match themes of interest, and builds statistical features relative to pattern diffusion. Two classes were utilized for classification: truthy and legitimate. Four models were trained and AdaBoost with support vector machines and resampling performed the best, with a 96% accuracy.

Abokhodair et al. mapped the use of political bots during the Syrian conflict [7].

The authors investigate a Syrian social botnet with over 3,000 tweets in both Arabic and English, that were active for 35 weeks. Their aim was three-fold: understanding how a botnet grows over time, how the content of tweets differ from regular users, and how they influence relevant discussions. Their findings show the botnet was not trying to mimic human behavior, but instead was trying to spread civil war related hashtags with topics not related to war.

Bessi and Ferrara explored how social bots distorted the 2016 U.S. presidential election [14]. They collected nearly 20.7 million tweets from 2.8 million distinct users using different hashtag queries. Utilizing the BotOrNot detection API, they labeled the top 50,000 accounts ranked by activity volume. The top 50,000 accounts make up 12.6 million tweets. A threshold of 0.5 was used to determine bot status and 7,183 users were classified as bots. Sentiment analysis was performed on the tweets using SentiStrength. The study draws three conclusions: influence can be redistributed across suspicious accounts with malicious purposes, conversation can become further polarized, and incorrect information can become more widespread.

More recently, Ferrara et al. explored the disruptive campaign conducted on social media known as "MacronLeaks" [36]. MacronLeaks was a coordinated dissemination of disinformation during the 2017 French election. Tweets were collected in the months leading to the French election and identified as social bots or real accounts. Their findings show the majority of users who engaged in MacronLeaks were part of the alt-right Twitter community not the French population. Most importantly, they found bot accounts linked to the 2016 presidential election joined the MacronLeaks campaign, indicating a possible market for bots.

## 5.3 EXPERIMENTAL METHODOLOGY

### 5.3.1 Datasets

Two distinct datasets are used for both case studies; the first is our tweet election dataset. This dataset was collected from September 22nd to November 8th, 2016. The dataset consists of 2,942,808 tweets from 705,381 unique user accounts. No restrictions on location or account was implemented; however, query terms were limited to topics related to both the Republican and Democratic candidates. The following terms were used in our search queries: "Trump", "Clinton", "President Trump", "President Clinton", "#trump2016", "#clinton2016". Due to the size of our election dataset, labeling sentiment in tweets by hand is not feasible. Thus, we opt for a state-of-the-art deep neural network approach.

To train our neural network, we use the sentiment140 corpus [41]. The sentiment140 corpus is a collection of tweets collected based on the presence of emoticon(s) in the text. Emoticons are combinations of symbols or characters used to express emotion or represent an image. Tweets were labeled positive or negative based on eight emoticons present in the text. Emoticons were removed from the text after the labeling process as they are essentially the class label. The final dataset consists of 1.6 million tweets, with an even split between positive and negative instances. We elect to use the sentiment140 dataset because we have used it to train high performing models for sentiment detection in previous studies [51, 104, 52]. More information on our neural network and training parameters are presented in the next section.

### 5.3.2 Deep Neural Network Architectures

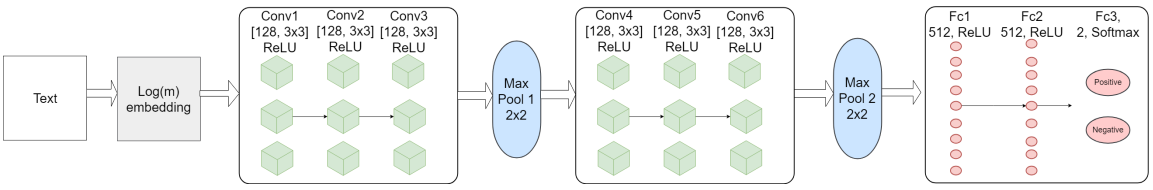
Typical ConvNet architecture consists of one or more convolutional layers, max pooling, and dense fully connected layers in some sequence. We elect to use an Alexnet-like ConvNet architecture [71], which has been demonstrated to be effective for text anal-

ysis, specifically at the character-level [99, 137, 96]. Table 5.1 presents the breakdown of our neural network by layer, with the number of trainable parameters for each layer. The convolutional layer connects neurons to small regions of the current input space and applies a number of linear filters across this input space. The output of those linear filters are passed to a non-linear (ReLU) activation function. A max pooling layer takes the maximum value from each of a cluster of neurons at the prior layer. Our architecture consists of six convolutional layers and two max pooling layers, with a max pooling layer occurring every three convolutional layers. The output is then passed through two fully connected layers and into the softmax classification layer. The ConvNet was implemented using Tensorflow 1.1.0 [6] with the Python API.

| | HyperParameters | | | | Num. of Trainable Parameters |
|---|---|---|---|---|---|
| Conv Layers | Number of Filters | Filter | Pool | Activation | Max Instance Length (374) |
| conv1 | 128 | $3 \times 3$ | – | ReLU | 1,280 |
| conv2 | 128 | $3 \times 3$ | – | ReLU | 147,584 |
| conv3 | 128 | $3 \times 3$ | $2 \times 2$ | ReLU | 147,584 |
| conv4 | 128 | $3 \times 3$ | – | ReLU | 147,584 |
| conv5 | 128 | $3 \times 3$ | – | ReLU | 147,584 |
| conv6 | 128 | $3 \times 3$ | $2 \times 2$ | ReLU | 147,584 |
| FC Layers | Number of Neurons | | | Activation | |
| fc1 | 512 | | | ReLU | 12,190,208 |
| fc2 | 512 | | | ReLU | 262,656 |
| fc3 | 2 | | | softmax | 1,026 |
| Total Trainable Parameters | | | | | 13,193,090 |

Table 5.1: Neural Network Layers, Hyperparameters and Trainable Parameters

Figure 5.1: Neural Network Process




### 5.3.3 Character Embedding

The text needs to be processed before being used with the neural network as inputs are strictly numeric. To match this constraint, we convert the text to an image-like

matrix representation using a character embedding. We elect to use log(m) embedding with a 256-character alphabet, as this encompasses the full UTF-8 character list [98, 99]. Log(m) embedding takes a character $c$ in an instance of length $n$ and creates a vector representation of $c$ by converting it to its corresponding numeric value and then calculating the binary representation of that number. For an alphabet of $m$ characters, this creates a vector of size log(m), which is then combined with the remaining character vectors in the instance to form a matrix of size $l$ x $m$. This method is chosen as it reduces training time and memory requirements, and improves classification performance over other character embedding methods [99]. Figure 5.2 shows a visualization of this representation and Figure 5.1 depicts the full process by which the neural network is trained. The data was padded to the size of the largest instance, in our case 374 characters, to fill the uniformity requirement. Training and validation were performed using a 90:10 train:test split and the model was trained for 20 epochs, with random shuffling of the training data occurring at the beginning of every epoch. We elected to use 20 epochs as we found no significant increase in performance with more epochs. The resulting sentiment analysis model has an accuracy of 84% on the test dataset. As this model is used to label our election data, a number of the tweets may be mislabeled. At the individual tweet level, accuracy is a useful and important measure; however, when working with a larger dataset, it is more important to get a proper distribution of positive/negative instances than correctly labeling every individual tweet.

Figure 5.2: Visualization of log(m) embedding which results in an $8 \times max\_len$ matrix representation.

### 5.3.4 Polling and Predicting

As previously mentioned, we want to explore the use of Twitter as both a polling resource and a predictor of the election results. In Tumasjan et al. [124], the authors found the volume of tweets about candidates directly mirrored election results for the 2009 German election and Boutete et al. [17] found volume of tweets directly linked a user to a political party. For this reason, we utilize the ratio of tweets related to Trump and Clinton as a metric for volume and the ratio of positive tweets for Trump to Clinton as our metric for sentiment. For polling purposes, we compare the volume and sentiment ratios against leading polls from different sources. Poll results are obtained from Real Clear Politics [1], which contains polls from October 26th to November 8th. To determine if Twitter is an effective polling resource, we compare our results to polls conducted by different entities across the 13 days leading up to the election. We choose three different entities to observe: IBD/TIPP, the LA Times, and ABC. These polls were chosen for the following reasons: 1) IBD/TIPP has been the most accurate poll in previous elections [4], 2) the LA Times poll leaned more towards Trump, and 3) the ABC poll leaned more towards Clinton. This is first done on a national level, then broken down into a state-level.

For state-level analysis, tweets are sectioned into their respective state based on the location data found in the user profile. This was generally in a "City, State" format found in the user profile. Twenty-one states were selected, with 11 being swing states (CO, FL, IA, MI, MN, NC, NH, NV, OH, PA, VA), five favored for Clinton (CA, NY, MD, MA, HI), and five favored for Trump (AL, OK, TN, WV, WY). The favored states were chosen because they had the largest discrepancies between the candidates in terms of vote percentage in the actual election. The remaining 11 states were considered swing states by major polls. Only 21 states were included due to the limited availability of location data in the collected tweets. Tweet volume and sentiment by state are compared against polls conducted within the state. Polling

entities vary by state, as not all institutions conduct polling in every state. State-level analysis will also be used for determining election outcome.

Both volume and sentiment ratios are explored for determining election outcome. For our purpose, the data was normalized based on the amount of electors in the state and, either volume or positive sentiment originating from said state. Two approaches are employed to determine outcome: winner-take-all and shared elector count. The winner-take-all approach reflects the real-world, where states award all their electors to the victor. The shared elector count awards each candidate a fraction of the electors for each state based on the ratio of volume, or positive sentiment, between both candidates.

We did not collect data regarding the independent candidates running for presidency, as we are only interested in the two front-runners. This should cause no difference when determining election outcome as no independent candidate won any state during the 2016 election; however, volume and sentiment ratios may be slightly higher for the popular vote as the amount of votes cast for Clinton and Trump will not equal 100% of votes cast in the actual election.

### 5.3.5 Removal of Social Bots

To determine whether users are social bots or not, the Botometer (formerly BotOrNot) model was used [29]. Botometer is a social bot detection model created specifically for Twitter in 2016. The model has been used in related studies to determine whether accounts are social bots [44, 14]. The model uses a Random Forest classifier and over 1,000 features collected directly using Twitter's API. Model performance was measured in AUC and scored 0.95 [29]. The Botometer model outputs many scores depending on various feature types, such as temporal, network, content, etc. The all-feature score is an overall score given based on all the feature types and is the score used to determine whether an account is a bot or not. The Botometer documentation

states many users fall within the range of 0.40 to 0.60, and a related study suggested using 0.50, since they found the majority of their accounts fell under that value [14]. As there is no certain threshold that explicitly identifies an account as a social bot, we elect to use six thresholds for bot detection: 0.50, 0.55, 0.60, 0.65, 0.70, and 0.75. We report and compare results for every threshold; however, we cannot confirm whether accounts are actually social bots or not.

Again, tweet volume and sentiment are used as metrics. Volume and sentiment between candidates are explored in three distinct scenarios: 1. Removing social bot accounts from the dataset, 2. Limiting Tweets to one per user, and 3. Both Removing social bot accounts and limiting tweets to one per user. These are analyzed both with all tweets and with the removal of tweets that included both a mention of Trump and Clinton. Double mentions are removed because there is no feasible method to determine which candidate the expressed sentiment is towards when both candidates are mentioned in the text. With respect to limiting users to one tweet, we chose to keep the most recent tweet as that would most likely reflect the authors opinion closest to the election date.

## 5.4 POLLING AND PREDICTING THE ELECTION

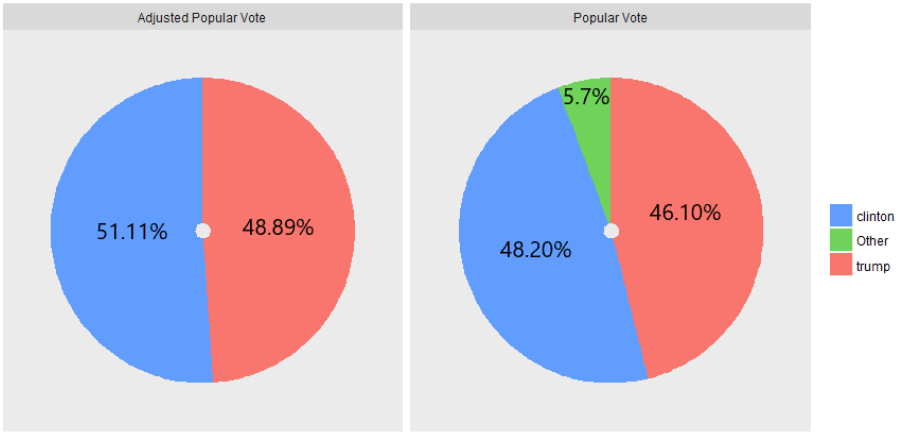



Figure 5.3: Actual election results for electoral and popular votes

### 5.4.1 National Level

*Polling*

Table 5.2 contains the polling characteristics and the results of the popular vote for the 2016 general election. We see that the majority of the polls predict a Clinton victory, with the highest spread putting her ahead by 7 points. Only the polls conducted by

| Date | Poll | Clinton | Trump | Spread |
|---|---|---|---|---|
| Oct 27 | ABC | 51 | 44 | Clinton +7 |
| | IBD/TIPP | 44 | 42 | Clinton +2 |
| | LA Times | 45 | 45 | Tie |
| Oct 28 | ABC | 50 | 45 | Clinton +5 |
| | IBD/TIPP | 45 | 42 | Clinton +3 |
| | LA Times | 44 | 46 | Trump +2 |
| Oct 29 | ABC | 49 | 46 | Clinton +3 |
| | IBD/TIPP | 46 | 41 | Clinton +5 |
| | LA Times | 44 | 46 | Trump +2 |
| Oct 30 | ABC | 49 | 46 | Clinton +3 |
| | IBD/TIPP | 45 | 41 | Clinton +4 |
| | LA Times | 44 | 46 | Trump +2 |
| Oct 31 | ABC | 49 | 47 | Clinton +2 |
| | IBD/TIPP | 45 | 43 | Clinton +2 |
| | LA Times | 43 | 47 | Trump +4 |
| Nov 01 | ABC | 48 | 47 | Clinton +1 |
| | IBD/TIPP | 45 | 44 | Clinton +1 |
| | LA Times | 43 | 47 | Trump +4 |
| Nov 02 | ABC | 48 | 47 | Clinton +1 |
| | IBD/TIPP | 44 | 44 | Tie |
| | LA Times | 42 | 48 | Trump +6 |
| Nov 03 | ABC | 49 | 47 | Clinton +2 |
| | IBD/TIPP | 44 | 44 | Tie |
| | LA Times | 43 | 48 | Trump +5 |
| Nov 04 | ABC | 49 | 45 | Clinton +4 |
| | IBD/TIPP | 45 | 44 | Clinton +1 |
| | LA Times | 43 | 47 | Trump +4 |
| Nov 05 | ABC | NA | NA | NA |
| | IBD/TIPP | 46 | 43 | Clinton +3 |
| | LA Times | 43 | 48 | Trump +5 |
| Nov 06 | ABC | 49 | 44 | Clinton +5 |
| | IBD/TIPP | 45 | 44 | Clinton +1 |
| | LA Times | 43 | 48 | Trump +5 |
| Nov 07 | ABC | 49 | 46 | Clinton +3 |
| | IBD/TIPP | 43 | 42 | Clinton +1 |
| | LA Times | 43 | 48 | Trump +5 |
| Nov 08 | ABC | NA | NA | NA |
| | IBD/TIPP | 43 | 42 | Clinton +1 |
| | LA Times | 44 | 47 | Trump +3 |
| Nov 08 | Popular Vote | 48.2 | 46.1 | Clinton +2.10 |
| Nov 08 | Adjusted Popular Vote | 51.11 | 48.89 | Clinton +2.22 |

Table 5.2: Poll results across days leading to the election

the LA Times put Trump in the lead. When we compare these polls to the popular vote result, we see that the majority were accurate in determining that Clinton would win the popular vote, with the IBD/TIPP tracking poll being the most accurate. Figure 5.3 presents the ratio of votes in the election in terms of the popular vote. In Figure 5.3, Popular Vote includes vote count ratio for Clinton, Trump, and independent candidates, while Adjusted Popular Vote shows the popular vote ratio when considering only Clinton and Trump. The adjusted popular vote ratio is calculated by taking the count of votes for one candidate and dividing it by the total votes for both candidates. We also believe timing factors into the calculation, thus, in addition to the full dataset, we split the dataset into seven different subsets for the national polling experiment. The data is split into tweets from the month of September (22nd to the end of the month), October, and November. For November, we look at 8, 5, 3, and 2 days before the election, and the election day itself. The data is taken from the date chosen to the end of the month, for example at 8 days before the election the subset consists of tweets from November 1st to November 8th, while October consists of October 1st to October 31st.

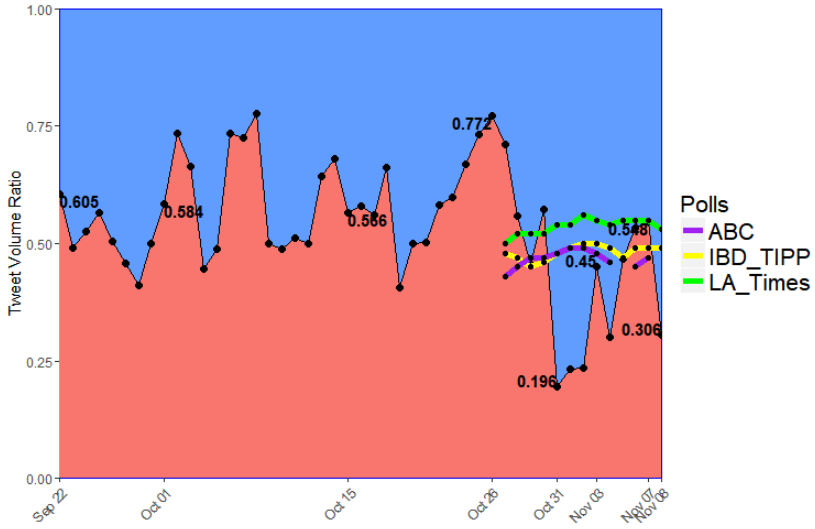



Figure 5.4: Daily Trump:Clinton volume ratios of tweets and polls for each candidate across the full dataset

Polls were only available from October 26th to November 8th on Real Clear Politics

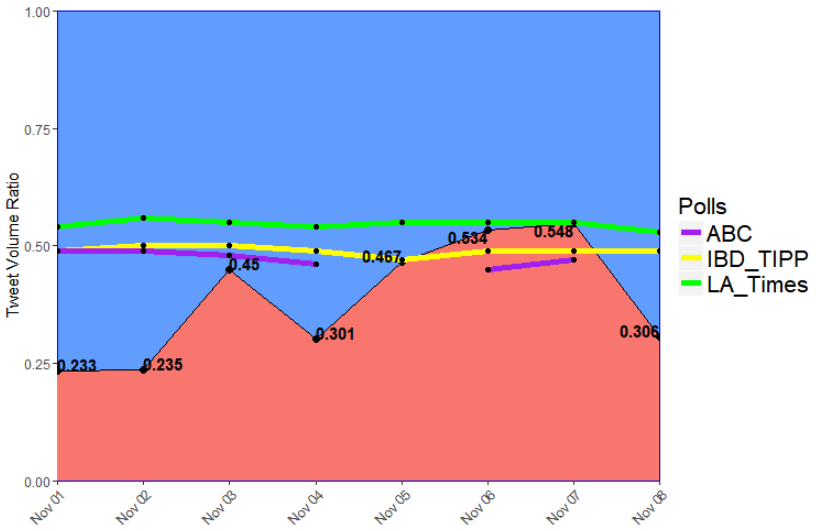



Figure 5.5: Daily Trump:Clinton volume ratios of tweets and polls for each candidate during November

[1]. Figures 5.4 and 5.5 show daily ratios of the volume tweets related to Trump (red) and Clinton (blue). In addition, these figures contain poll results across the full dataset and the month of November, respectively. The y-axis is the ratio of Trump:Clinton tweets, thus a higher value means more tweets about Trump. In the months of September and October, the number of tweets referring to Trump is greater than the number of tweets referring to Clinton; however, this trend reverses in the days leading up to the election. As the election grows closer, the volume ratios begin to reflect the poll results. To further explore the effects of volume of tweets as a polling resource, we aggregate the volume across eight time cut-offs, this is presented in Figure 5.6. Tweets from the 22nd to the end of September refer more to Clinton than Trump, with Clinton having a 5-point lead. In the month of October, the volume of Trump-related tweets overtakes the volume of Clinton tweets with Trump being 11 points in front of Clinton. Finally, in the days leading up to, and the day of the election, Clinton has a large lead when looking at aggregated volume of tweets. The aggregated volume across the full dataset gives Trump a 5-point lead over Clinton. Daily ratio of volume seems to reflect polls in the final days of the election but not

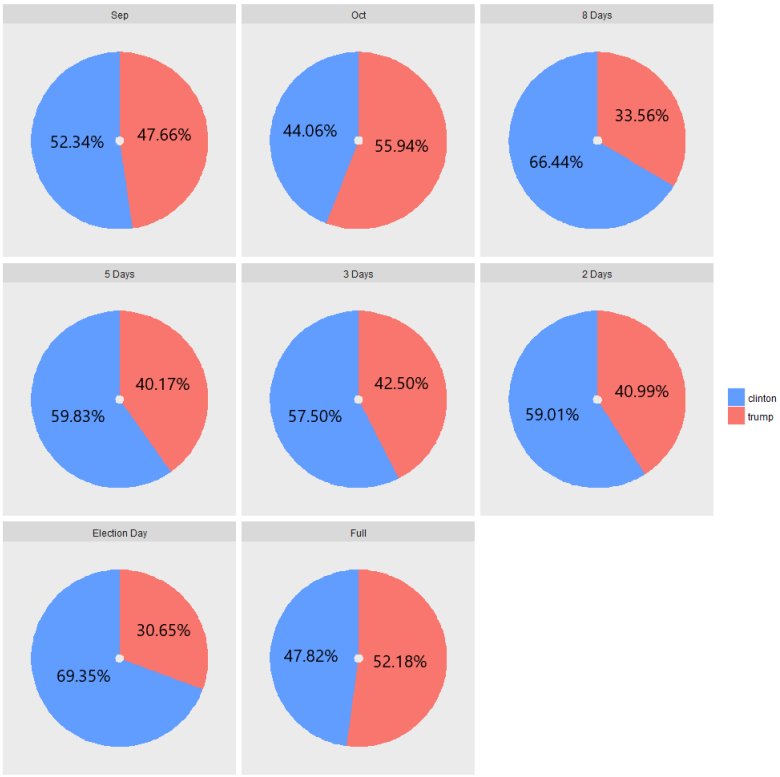



Figure 5.6: Tweet percentages for each candidate based on volume by subset

in the months before. Moreover, the the volume aggregated over the full dataset resembles the LA Times poll results. When compared to the election results, volume does not seem to be a good polling resource, as Clinton won the popular vote.

Figures 5.7 and 5.8 show daily ratio of tweets with positive sentiment related to Clinton (blue) and Trump (red), and poll results across the full dataset and the month of November, respectively. We see that these graphs follow a similar trend to the daily volume graphs, with Trump having a larger positive presence in the month of October. Again, in the month of November, this lead flips and Twitter users have a more positive opinion on Clinton than Trump. The polls generally hold steady around 0.5 while the daily ratio of Trump:Clinton fluctuates between 0.154 to 0.653 in the days leading up to the election. From November 4th to November 7th, the ratios are close to poll values. Figure 5.9 presents the ratio of positive Trump tweets and positive Clinton tweets. We consider a positive tweet about a candidate as a "twitter vote"

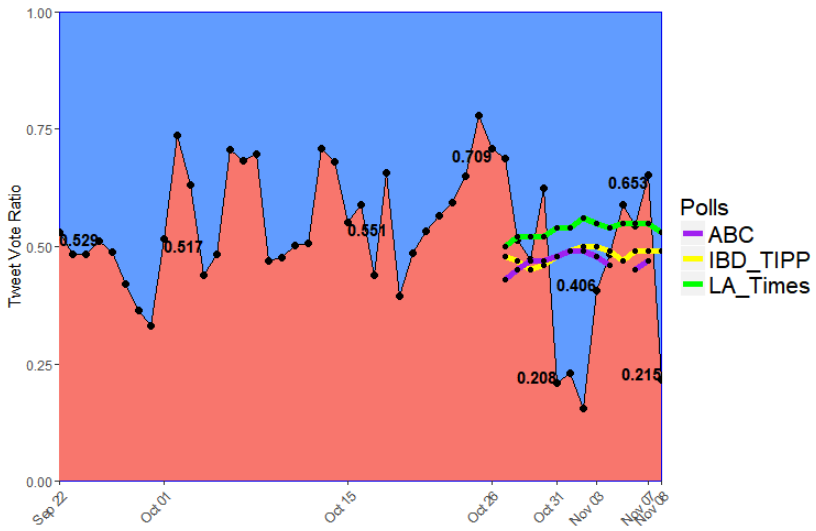



Figure 5.7: Daily Trump:Clinton positive tweet ratio and polls for each candidate across the full dataset

for that candidate. This allows us to compare the ratio of Twitter votes to Popular vote. This also allows us to compare our twitter votes with polling results and see if tweet sentiment can be an indicator on par with polling of individuals. Theoretically, polling individuals and sentiment go hand in hand since their answers to the polls should reflect their sentiment towards their candidate. The trends observed with positive sentiment are similar to the trends in volume, but they are more exaggerated in sentiment. In September, Clinton had the lead by 15 points; however, this lead was reversed and Trump was ahead by 9 points during October. November has Clinton in the lead across all time cut-offs. Finally, across the full dataset Clinton has a lead in positive sentiment of approximately 1 point, mirroring the results of the IBD/TIPP tracking polls found in Table 5.2, which has been claimed to be the most accurate poll in previous elections [4]. When compared to election results, sentiment comes closer to the correct difference in vote, indicating sentiment is a better polling metric than volume and Twitter can be successfully utilized for polling the popular vote.

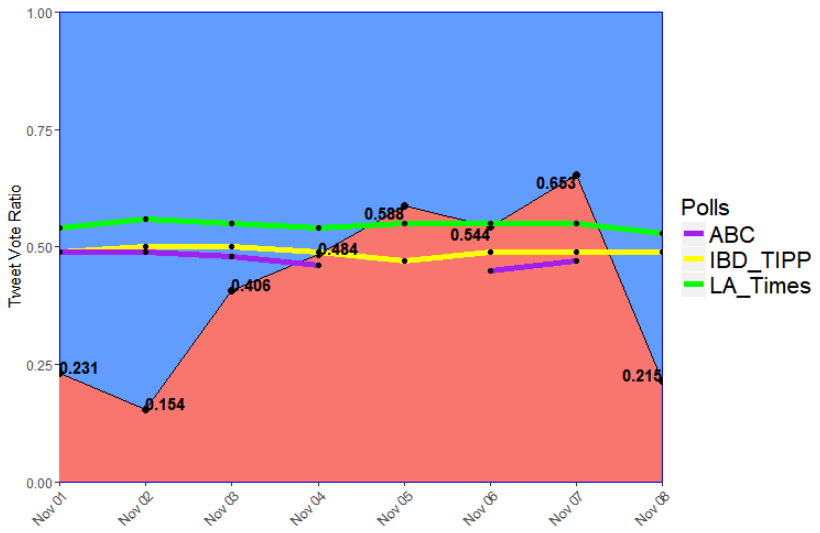



Figure 5.8: Daily Trump:Clinton positive tweet ratio and polls for each candidate during November

### 5.4.2 State-level

*Predicting Outcome*

In this section, sentiment and volume ratio will be combined with the winner-take-all and shared elector vote approaches to determine whether Twitter can effectively predict the election results. This experiment is conducted across the full dataset and for each of our 21 individual states. We explore the differences and similarities found between swing states and favored states. The results for the 2016 presidential election declared Trump the victor with 306 electors (56.88%) across 30 states, leaving Clinton with the remaining 232 electors (43.12%) across the 20 states and the District of Columbia (D.C.). When limiting electors to the 21 selected states, Trump had 139 (47.93%) electors and Clinton had 151 (52.07%) electors. The states chosen may bias Clinton, as the favored states chosen represent a larger percentage of her electors.

Figures 5.10 and 5.11 depict the volume and sentiment across all 21 states and the full dataset. Figures 5.12 and 5.13 breaking down volume and sentiment based on swing and favored states, respectively. The y-axis represents the Trump:Clinton ratio, meaning a larger value indicates a higher volume (or positive sentiment) of tweets

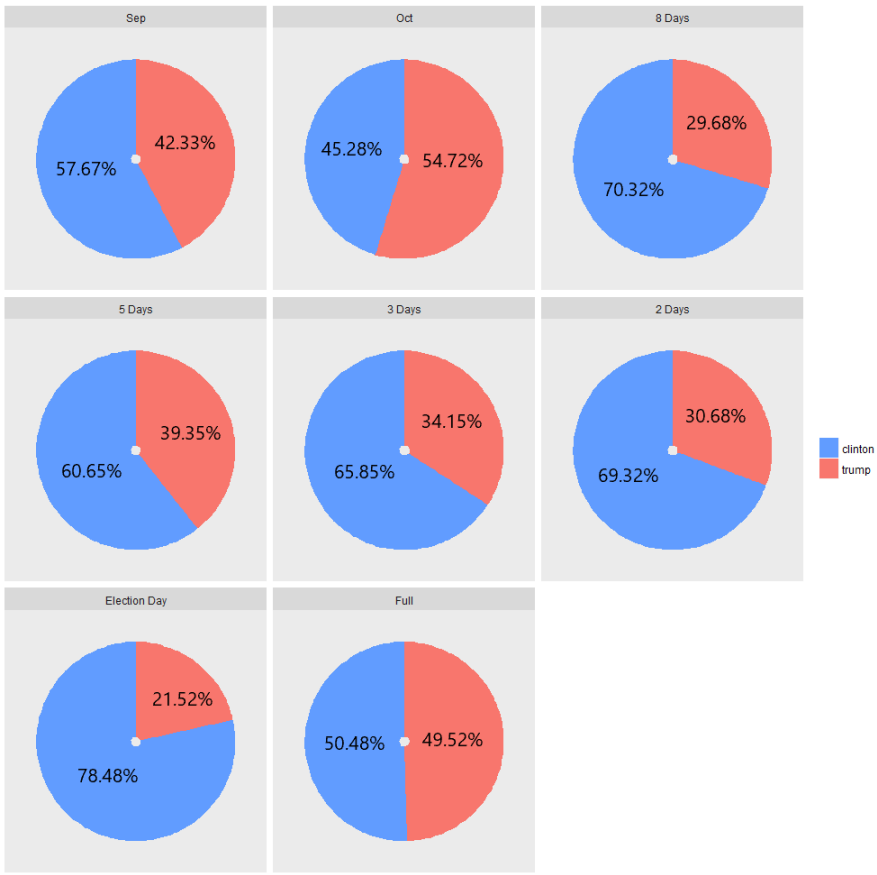
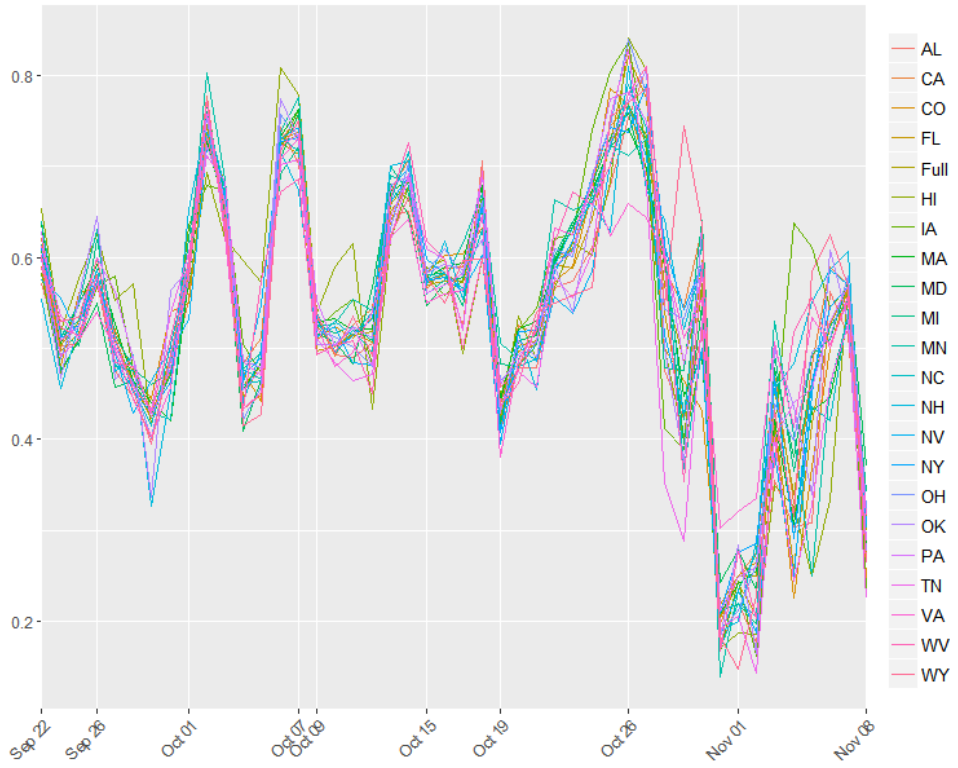



Figure 5.9: Tweet percentages for each candidate based on sentiment by subset

about Trump. The number of tweets regarding the candidates fluctuate greatly across the election period. When comparing all graphs, the ratio in tweets mentioning either candidate follows a trend across the majority of the election time frame regardless of state, volume, or sentiment. The large advantage held by candidates in their favored states is not depicted on Twitter. This may be due to the age demographic, as Twitter primarily consists of users in the 18-36 age range [119]. It is interesting to note, after the first debate (September 26th) the number of tweets related to Clinton increases and the majority of news outlets agree that Clinton won the first debate. On October 7th, a video of Trump musing about sexually assaulting women was released to the public and we see a sharp decrease in popularity over the following days. For the second debate (October 9th), polls conducted by Politico, CNN, NBC, Gallup, and Fox News all found Clinton to be the victor. However, Twitter holds a steady

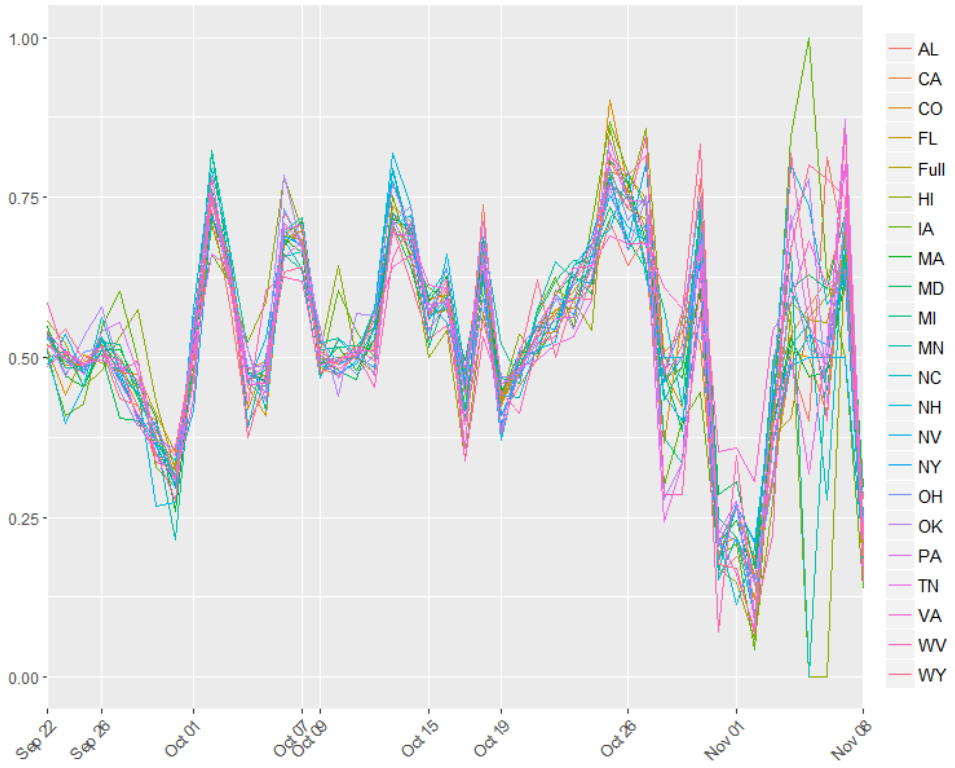
Figure 5.10: Visualization of the daily volume ratio of Trump tweets to Clinton tweets (Trump:Clinton) for all 21 states and the full dataset




ratio in the days after this debate, with a small increase for Clinton on Oct 13th. According to the polls, Clinton was again victorious in the third debate (October 19th), although this time by a smaller margin. Twitter does not depict this trend, as after the 19th there is only a large steady increase for Trump. Generally, tweets follow the same pattern regardless of state, with few deviations, until the final days leading towards the election. For these days, sentiment seems to exhibit greater variation than volume. To examine volume and sentiment as predictors of election result, we take two approaches: a shared elector count and a winner-take-all approach.

Table 5.3 displays the volume, positive sentiment, and percentage of positive sentiment per candidate before normalization. Items in boldface indicate the victor in the corresponding state. This table shows that Trump has more tweets pertaining to him across every state, thus the winner-take-all approach grants Trump a unanimous victory. Since volume of tweets is imbalanced, positive sentiment of a candidate based

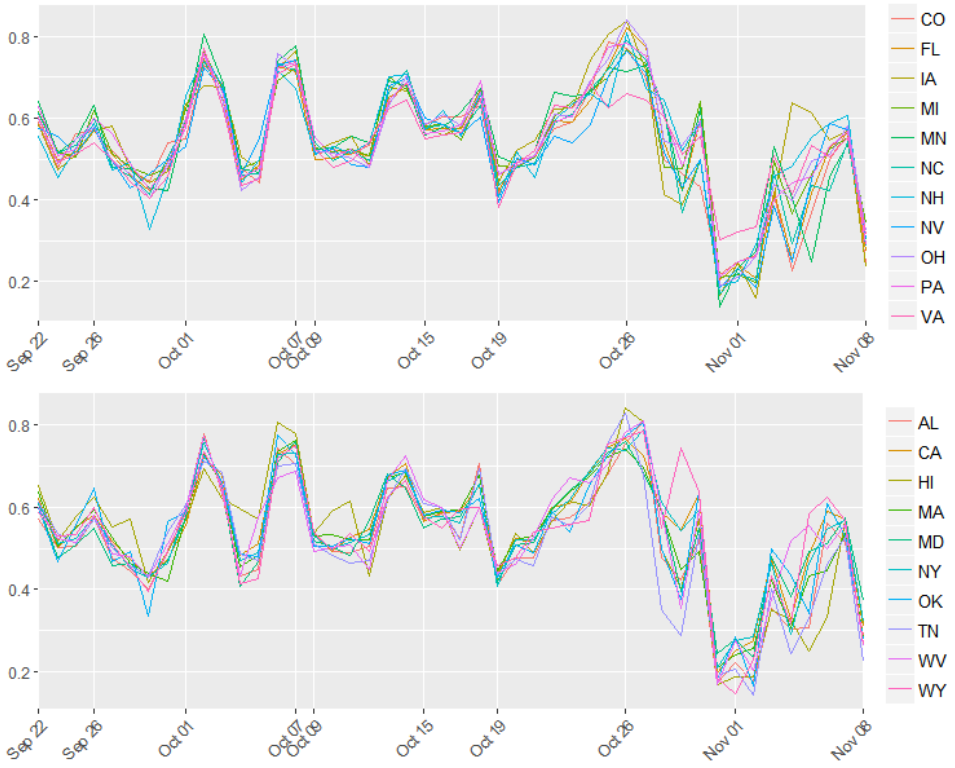
Figure 5.11: Visualization of the daily positive sentiment ratio of Trump tweets to Clinton tweets (Trump:Clinton) for all 21 states and the full dataset




on total tweet volume for that candidate needs to be taken into account. When examining the percentage of positive tweets contained within the total volume of tweets for a candidate, winner-take-all outcome is reversed and Clinton is victorious in all states. This indicates there were more positive tweets about Clinton than Trump across all states when you account for volume of tweets for each candidate. As for positive sentiment alone, Clinton has more positive tweets in AL, CO, NC, NV, TN, and WV, granting her 55 (18.97%) electors and leaving Trump with the remaining 235 (81.03%) electors. These results do not mirror election results as two of the three approaches result in a candidate winning every state, and the final approach awards Clinton only five states, of which three are Trump favored and two are swing states.

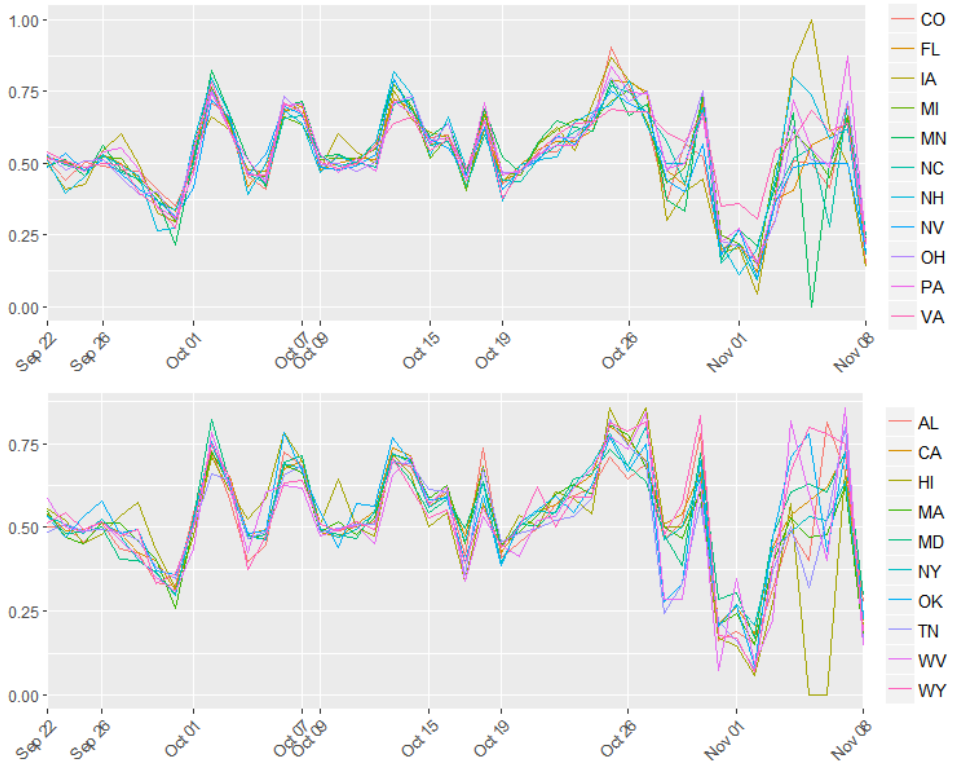
The shared elector count approach requires normalizing the data by state elector count and allowing for both candidates to take a percentage based on their total tweets within each state. Table 5.4 depicts the normalized values for the electors from each

Figure 5.12: Visualization of the daily volume ratio of Trump tweets to Clinton tweets (Trump:Clinton) of tweets for swing (top) and favored (bottom) states.




state based on volume and positive sentiment. The winner-take-all results do not differ after data is normalized, thus we do not include it in this table. These results indicate when using volume, out of the 290 electors allotted across the states, Trump receives 153.51 (52.93%) electors and Clinton receives 136.49 (47.07%) electors and while using sentiment Trump receives 145.65 (50.22%) electors and Clinton receives 144.35 (49.78%) electors. If we extrapolate this to total electors, Trump gets 284.76 and Clinton gets 253.23 electors when using volume, and 270.18 and 267.82 when using sentiment, respectively. It should be noted, sentiment predicts seven of the eleven swing states correctly, regardless of approach. The large advantages for candidates in their favored states are not represented by either positive sentiment or volume, indicating neither option to be a good model for predicting election results in favored states.

Figure 5.13: Visualization of the daily positive sentiment ratio of Trump tweets to Clinton tweets (Trump:Clinton) of tweets for swing (top) and favored (bottom) states.




*Polling*

Twitter does not represent the large advantages found in favored states, implying it will not be a good polling resource in states where one candidate is known to have a tremendous advantage. This leaves the swing states, which will be the focus of determining the effectiveness of Twitter as a polling resource at the state-level. To determine whether volume or positive sentiment reflect polling values, we compare the Trump:Clinton ratios across these states to the average of the poll results conducted within the states. We elect to use utilize the Real Clear Politics (RCP) average instead of individual polls as polling entities within states may differ. Of the 21 states chosen, 11 are swing states. Of these, ten are selected for further experimentation. The only state excluded is MN, as they do not have an RCP average score. As previously seen in the national level polling analysis, the full-time period provides the closest values

| State | Class | Trump Vol | Clinton Vol | Trump Pos | Clinton Pos | Trump pos/vol | Clinton pos/vol |
|---|---|---|---|---|---|---|---|
| AL | Trump | **7,180** | 6,918 | 2,958 | **3,122** | .4120 | **.4513** |
| CA | Clinton | **71,967** | 61,369 | **29,329** | 28,265 | .4075 | **.4606** |
| CO | Swing | **10,220** | 9,196 | 4,073 | **4,112** | .3985 | **.4472** |
| FL | Swing | **50,544** | 46,215 | **21,066** | 20,959 | .4168 | **.4535** |
| HI | Clinton | **2,174** | 1,904 | **863** | 849 | .3970 | **.4459** |
| IA | Swing | **4,366** | 3,709 | **1,749** | 1,658 | .4006 | **.4470** |
| MA | Clinton | **16,050** | 13,924 | **6,408** | 6,386 | .3993 | **.4586** |
| MD | Clinton | **12,364** | 11,104 | **5,097** | 5,071 | .4122 | **.4567** |
| MI | Swing | **13,193** | 11,539 | **5,281** | 5,178 | .4003 | **.4487** |
| MN | Swing | **7,037** | 5,931 | **2,801** | 2,648 | .3980 | **.4465** |
| NC | Swing | **16,799** | 15,361 | 6,574 | **6,937** | .3913 | **.4516** |
| NH | Swing | **4,914** | 4,363 | **1,988** | 1,982 | .4046 | **.4543** |
| NV | Swing | **7,844** | 7,227 | 3,142 | **3,187** | .4006 | **.4410** |
| NY | Clinton | **59,297** | 51,826 | **24,242** | 24,080 | .4088 | **.4646** |
| OH | Swing | **17,449** | 15,681 | **7,161** | 7,145 | .4104 | **.4556** |
| OK | Trump | **5,049** | 4,431 | **2,171** | 1,997 | .4300 | **.4507** |
| PA | Swing | **18,602** | 16,222 | **7,521** | 7,380 | .4043 | **.4549** |
| TN | Trump | **9,907** | 9,627 | 4,069 | **4,388** | .4107 | **.4558** |
| VA | Swing | **37,064** | 34,102 | **15,770** | 15,399 | .4255 | **.4516** |
| WV | Trump | **2,999** | 2,709 | 1,186 | **1,238** | .3955 | **.4570** |
| WY | Trump | **4,131** | 3,774 | **1,697** | 1,668 | .4108 | **.4420** |

Table 5.3: Non-Normalized Winner-Takes-All results

to polls; thus, we elect to look at the volume and sentiment ratios across the full time period for each individual state. In addition, if each state dataset is split into time-periods the smaller datasets will not contain enough instances for a thorough analysis.

Table 5.5 reflects the RCP average, actual, volume, and sentiment spreads per swing state. Figures 5.14 and 5.15 depict a visualization of the volume and sentiment ratios for the swing states, respectively]. When compared to the election results per state, the RCP average correctly predicts seven of the eleven states. Of these correctly predicted states the spread was close to the actual spread shown in popular vote. RCP also has a Clinton bias, with 2/3 of incorrect states claiming a Clinton victory. Volume predicts all states as Trump victories, with the smallest spread

| State | Class | Electoral votes | Trump volume | Clinton volume | Trump positive | Clinton positive |
|---|---|---|---|---|---|---|
| AL | Trump | 9 | 4.58 | 4.42 | 4.37 | 4.62 |
| CA | Clinton | 55 | 29.69 | 25.31 | 28.01 | 26.99 |
| CO | Swing | 9 | 4.74 | 4.26 | 4.48 | 4.52 |
| FL | Swing | 29 | 15.15 | 13.85 | 14.54 | 14.46 |
| HI | Clinton | 4 | 2.13 | 1.87 | 2.02 | 1.98 |
| IA | Swing | 6 | 3.24 | 2.76 | 3.08 | 2.92 |
| MA | Clinton | 11 | 5.89 | 5.11 | 5.51 | 5.49 |
| MD | Clinton | 10 | 5.27 | 4.73 | 5.01 | 4.99 |
| MI | Swing | 16 | 8.54 | 7.46 | 8.08 | 7.92 |
| MN | Swing | 10 | 5.43 | 4.57 | 5.14 | 4.86 |
| NC | Swing | 15 | 7.84 | 7.16 | 7.30 | 7.70 |
| NH | Swing | 4 | 2.12 | 1.88 | 2.01 | 1.99 |
| NV | Swing | 6 | 3.12 | 2.88 | 2.98 | 3.02 |
| NY | Clinton | 29 | 15.47 | 13.53 | 14.55 | 14.45 |
| OH | Swing | 18 | 9.48 | 8.52 | 9.01 | 8.99 |
| OK | Trump | 7 | 3.73 | 3.27 | 3.65 | 3.35 |
| PA | Swing | 20 | 10.68 | 9.32 | 10.10 | 9.90 |
| TN | Trump | 11 | 5.58 | 5.42 | 5.30 | 5.70 |
| VA | Swing | 13 | 6.77 | 6.23 | 6.58 | 6.42 |
| WV | Trump | 5 | 2.63 | 2.37 | 2.45 | 2.55 |
| WY | Trump | 3 | 1.57 | 1.43 | 1.51 | 1.49 |
| Full | | 290 | 153.51 | 136.49 | 145.65 | 144.35 |

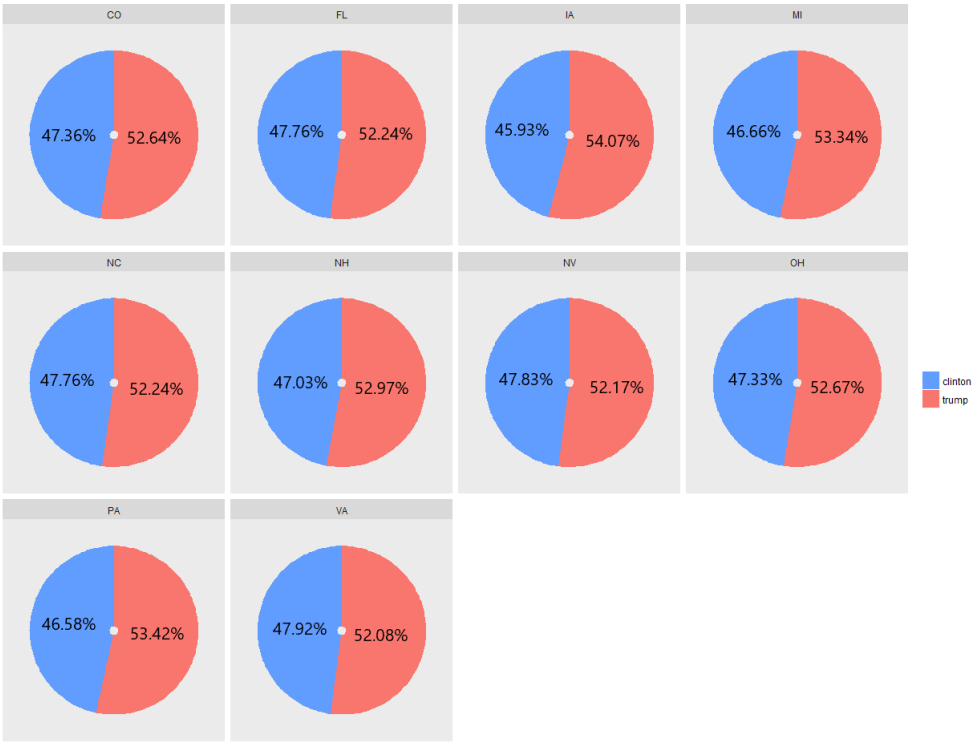
Table 5.4: Normalized Electors by Volume and Sentiment for both Shared Elector Vote

| State | RCP Average | Actual Spread | Vol Spread | Sent Spread |
|---|---|---|---|---|
| CO | Clinton 2.9 | Clinton 4.9 | Trump 2.6 | Clinton 0.2 |
| FL | Trump 0.2 | Trump 1.2 | Trump 2.2 | Trump 0.1 |
| IA | Trump 3.0 | Trump 9.5 | Trump 4.1 | Trump 1.3 |
| MI | Clinton 3.4 | Trump 0.3 | Trump 3.3 | Trump 0.4 |
| NC | Trump 1.0 | Trump 3.7 | Trump 2.2 | Clinton 1.3 |
| NH | Clinton 0.6 | Clinton 0.3 | Trump 3.0 | Trump 0.1 |
| NV | Trump 0.8 | Clinton 2.4 | Trump 2.0 | Clinton 0.3 |
| OH | Trump 3.5 | Trump 8.1 | Trump 2.7 | Trump 0.1 |
| PA | Clinton 1.9 | Trump 0.7 | Trump 3.4 | Trump 0.4 |
| VA | Clinton 5.0 | Clinton 5.4 | Trump 2.1 | Trump 0.6 |

Table 5.5: Polling per Swing State

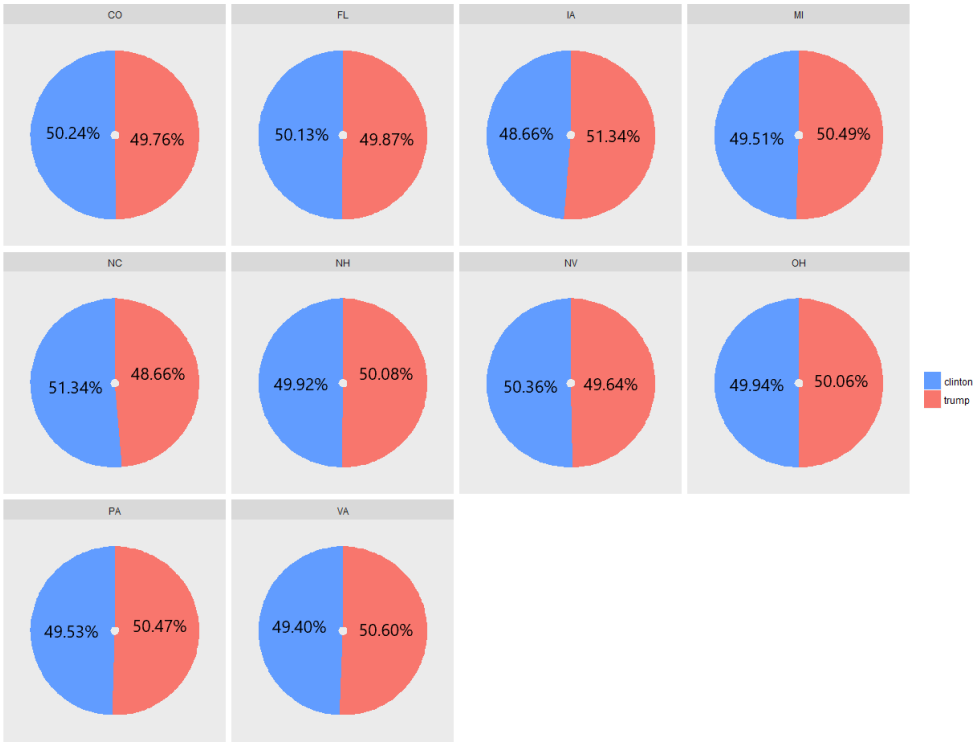
being 2.1 points. This is unrealistic as a polling resource as it assumes Trump will be victorious in every state. Sentiment more closely resembles the RCP average

Figure 5.14: Volume ratio between Trump and Clinton across 10 swing states




and actual spread. Positive sentiment correctly predicts seven of the eleven states exactly like the RCP average; however, sentiment has a Trump bias, with 2/3 of incorrect states claiming a Trump victory. Moreover, the states correctly predicted by sentiment are incorrectly predicted by polls and the inverse is true as well, indicating a combination of sentiment and polling may lead to better results. The spreads are much smaller when observing positive sentiment, with the smallest being 0.1 in OH and NH. In swing states where the election was close, such as NH, PA, NV, FL, and MI, sentiment spread corresponds to actual spreads. However, when looking at states that had a wider margin, such as OH, IA, VA, and CO, the sentiment spread does not come close to matching the actual spread or the RCP average spread. These results indicate that on a state-level, volume is not a good predictor for popular vote outcome. Sentiment produces spreads closer to actual results in states where the race is very close. However, when a state has a larger difference, sentiment does not perform well and the RCP average produces closer spreads. Using a combination

Figure 5.15: Positive sentiment ratio between Trump and Clinton across 10 swing states




of RCP average and positive sentiment could produce more accurate spreads across swing states in future elections.

### 5.4.3 Significance Testing

| Factor | Metric | W-score | p-value |
|---|---|---|---|
| Swing | volume | 71 | .5190 |
| Favored | volume | 57 | .6305 |
| Swing | sentiment | 64 | .8469 |
| Favored | sentiment | 51 | .9705 |

Table 5.6: Wilcoxon-Mann-Whitney test results

Table 5.6 displays the results of the Wilcoxon-Mann-Whitney [78] test comparing volume and sentiment across swing and favored states. This test was chosen because our data does not follow a normal distribution. We find that neither factor was statistically significant, indicating neither volume nor sentiment can be considered

an accurate predictor of election outcome per state. This could be due to small sample size as we only use 21 of the 50 states. This could also be caused by allowing multiple tweets per user, as this would skew the perception of volume and sentiment. In addition, the number of tweets within each state dataset is also a small sample size of the state population. Even so, the trend in positive sentiment values mirrors the close race that was the 2016 election in swing states. During the election, swing states were won by very small margins, for example MI was won by only 0.23%, and all swing states exhibit this small margin between both candidates using sentiment.

## 5.5 THE INFLUENCE OF SOCIAL BOTS ON TWITTER

| Threshold | Bot Count | Percentage | Bot Count Double Mention | Percentage |
|---|---|---|---|---|
| 0.50 | 146,044 | 25.60% | 109,938 | 19.27% |
| 0.55 | 96,729 | 16.95% | 71,000 | 12.45% |
| 0.60 | 67,177 | 11.77% | 48,421 | 8.49% |
| 0.65 | 48,010 | 8.41% | 34,265 | 6.01% |
| 0.70 | 34,774 | 6.10% | 24,696 | 4.33% |
| 0.75 | 24,033 | 4.21% | 17,126 | 3.00% |

Table 5.7: Accounts classified as social bots based on different thresholds and percentage of the total dataset classified as bots

The Botometer model uses the Twitter API to retrieve their features; thus, accounts that have been deleted since collection of the dataset are unable to be classified. These accounts totaled 134,849 (19.12%) and were removed from the dataset, lowering the number of unique users to 570,532. There is no method for determining why these accounts were removed from Twitter; reasons may range from users deactivating/deleting their accounts to Twitter removing them for being bots or violating Twitter's Terms of Service. Unfortunately, Botometer cannot be used to score these accounts as they can not query Twitter's API for their account information if an account does not exist. This group will be referred to as the deleted cohort from this point on. The remaining accounts were classified as either real or social bots.

A related study used a threshold of 0.50 and found 14% of the top 50,000 active accounts were considered bots [14], while another study found that anywhere from 9-15% of accounts on Twitter are bots [128]. Table 5.7 summarizes the accounts determined to be bots based on different thresholds using all instances and by removing instances with double mentions. When comparing to [14], our results show higher bot presence at a 0.50 threshold. Limiting accounts to the top 50,000 active users may have removed bot accounts that are not as active. When removing tweets that include double mentions, we significantly reduce the number of accounts classified as bots across all thresholds. This indicates there was a significant number of accounts that only tweeted text including references to both candidates.

| | Vol Trump | Vol Clinton | Vote Trump | Vote Clinton |
|---|---|---|---|---|
| Multi-Tweet All Accounts | 52.18 | 47.82 | 49.52 | 50.48 |
| Deleted Cohort Removed | 50.20 | 49.80 | 47.45 | 52.54 |
| Single-Tweet All Accounts | 58.58 | 41.42 | 55.07 | 44.93 |
| Deleted Cohort Removed | 42.28 | 57.72 | 37.17 | 62.83 |

Table 5.8: Basline Results, with no changes

| | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|
| Multi-Tweet All Accounts | 41.73 | 58.27 | 46.42 | 53.58 |
| Deleted Cohort Removed | 40.56 | 59.44 | 46.39 | 53.61 |
| Single-Tweet All Accounts | 43.85 | 56.15 | 50.58 | 49.42 |
| Deleted Cohort Removed | 38.51 | 61.49 | 47.69 | 52.31 |

Table 5.9: Basline Sentiment Distributions

Table 5.8 shows the volume, sentiment, and vote ratios for the multi-tweet dataset with no changes, the multi-tweet dataset with the removal of the deleted accounts, the single-tweet dataset with no changes, and the single-tweet datasets with removal of the deleted accounts. Table 5.9 depicts the sentiment for the aforementioned datasets. These will be the baselines with which we compare future outcomes. In our analysis, we consider a vote to be a positive sentiment tweet regarding the candidate. All references to the ratio between candidates will be in a Trump:Clinton format.

There are stark differences seen between using the multi-tweet dataset and reducing the dataset to a single tweet per user. Most noticeably, the ratio of votes flips from 49:50 to 55:44 and the ratio in terms of volume also increases in favor of Trump, from 52:47 to 58:41. Finally, the positive:negative ratio for both candidates increases with respect to positive tweets, indicating there were more negative tweets from users regarding the candidates. It is important to note, our data initially shows a Trump bias for most cases, with the exception of the single-tweet deleted dataset.

When removing the deleted cohort, there is a decrease in both volume and sentiment for Trump in both the multi- and single-tweet datasets. Moreover, the magnitude of this change is much larger in the single-tweet dataset, lowering the vote and volume of Trump tweets from 55.07% to 37.17% and 58.58% to 42.28%, respectively. The contrast in positive sentiment is particularly striking, with an almost 18% decrease in votes from positive tweets. From the discrepancies, we can conclude the deleted cohort played a large role in our dataset and many of these deleted accounts had tweets that were favorable towards Trump. Unfortunately, these accounts cannot be labeled as either bots or real; however, Twitter has been removing a large amount of bot accounts over the previous year that caused many Republican conservatives to lose a significant number of followers [77, 26]. As the deleted cohort has a large effect on positive Trump tweets, it is possible many of these accounts were included in the recent bans.

### 5.5.1 Multi-tweet dataset

| Threshold | Vol Trump | Vol Clinton | Vote Trump | Vote Clinton | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|---|---|---|---|
| 0.50 | 53.13 | 46.87 | 50.27 | 49.73 | 40.34 | 59.66 | 45.23 | 54.77 |
| 0.55 | 52.81 | 47.19 | 49.97 | 50.03 | 40.52 | 59.48 | 45.41 | 54.59 |
| 0.60 | 52.63 | 47.37 | 49.80 | 50.20 | 40.70 | 59.30 | 45.59 | 54.41 |
| 0.65 | 52.49 | 47.51 | 49.69 | 50.31 | 40.89 | 59.11 | 45.75 | 54.25 |
| 0.70 | 52.40 | 47.60 | 49.63 | 50.37 | 41.06 | 58.94 | 45.89 | 54.11 |
| 0.75 | 52.35 | 47.65 | 49.58 | 50.42 | 41.20 | 58.80 | 46.03 | 53.97 |
| None | 50.20 | 49.80 | 47.45 | 52.54 | 40.56 | 59.44 | 46.39 | 53.61 |

Table 5.10: Multi-tweet dataset removed deleted accounts and removed bots

Table 5.10 presents the ratios for volume, vote, and sentiment for the multi-tweet dataset with the removal of the deleted accounts and bots. The removal of bots and the deleted accounts cause interesting trends, while Table 5.11 shows the ratios for the multi-tweet dataset with the removal of deleted accounts, bots, and tweets where both candidates are mentioned in the text. Figures 5.16 and 5.17 both depict the volume and vote ratios for Tables 5.10 and 5.11, respectively. The lower the threshold, the more accounts are classified as bots and removed from the dataset. Overall, within the thresholds there is a slight variation in values; however, the largest difference between the lowest threshold and the highest threshold in Table 5.10, in terms of volume, is only 0.78. When examining Table 5.11, the addition of removing double mentions causes a large increase in volume and vote ratio for Trump. This difference is better visualized in the figures, where there is a larger gap between Trump and Clinton in Figure 5.17 than in Figure 5.16. In addition, the sentiment ratio for each candidate significantly increases in favor of negative tweets. This indicates that there is a larger number of Clinton tweets being removed than Trump tweets. When we compare the outcome of removing deleted accounts and bots back to our multi-tweet baselines, the removal of bots makes a small difference at the threshold of 0.75 and a larger difference at 0.50 threshold. Removal of double-mentions largely favors Trump across all thresholds.

| Threshold | Vol Trump | Vol Clinton | Vote Trump | Vote Clinton | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|---|---|---|---|
| 0.50 | 58.79 | 41.21 | 56.14 | 43.86 | 38.01 | 61.99 | 42.36 | 57.64 |
| 0.55 | 58.45 | 41.55 | 55.81 | 44.19 | 38.12 | 61.88 | 42.46 | 57.54 |
| 0.60 | 58.34 | 41.66 | 55.74 | 44.26 | 38.24 | 61.77 | 42.52 | 57.48 |
| 0.65 | 58.26 | 41.74 | 55.70 | 44.30 | 38.35 | 61.65 | 42.58 | 57.42 |
| 0.70 | 58.26 | 41.74 | 55.74 | 44.26 | 38.45 | 61.55 | 42.62 | 57.38 |
| 0.75 | 58.29 | 41.71 | 55.80 | 44.20 | 38.53 | 61.47 | 42.65 | 57.35 |
| None | 50.20 | 49.80 | 47.45 | 52.54 | 40.56 | 59.44 | 46.39 | 53.61 |

Table 5.11: Multi-tweet dataset removed deleted accounts, removed bots, and removed double mentions

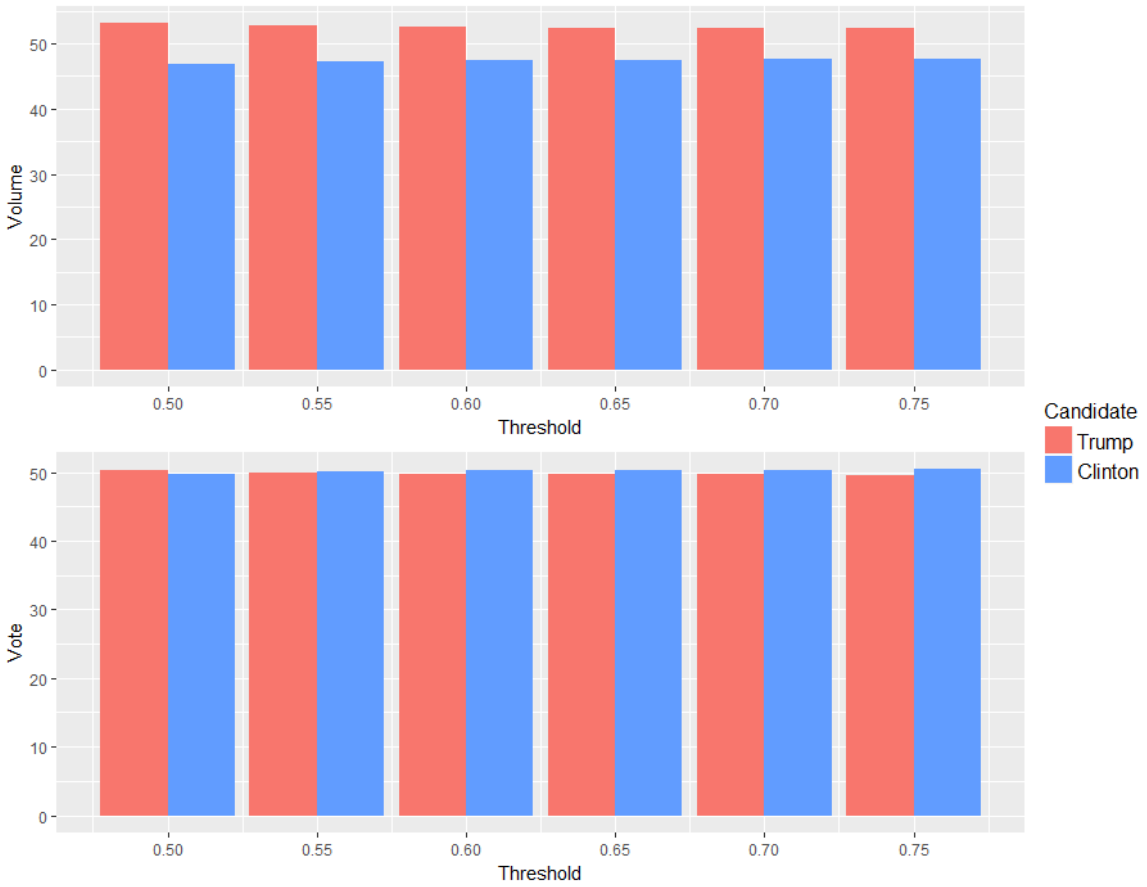



Figure 5.16: Bar graph depicting volume and vote for the multi-tweet dataset with the removal of the deleted accounts and bots

### 5.5.2 Single-tweet dataset

Table 5.12 presents the ratios for the single-tweet dataset with the removal of deleted accounts and bots, and Figure 5.18 depicts the volume and vote ratio for both candidates. The removal of deleted accounts and bots causes a large increase in volume and vote ratio for Clinton. When compared to the single-tweet deleted baseline, we see small changes in respect to volume and vote ratios towards Trump. In addition, compared to either single-tweet dataset baseline, the sentiment ratio for each candidate swings towards the negative.

As more bots are removed from the dataset, the ratio of Trump:Clinton increases towards Trump for both volume and vote; however, the ratio still largely favors Clin-

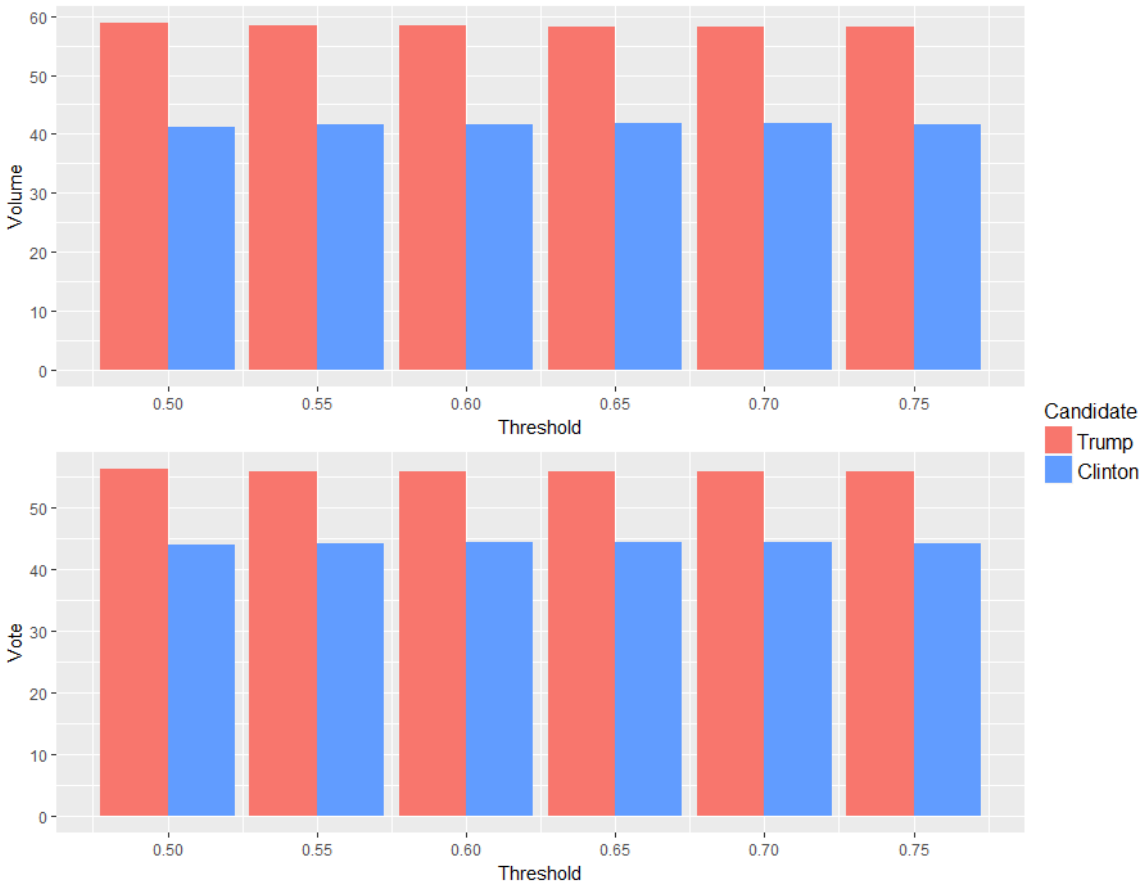



Figure 5.17: Bar graph depicting volume and vote for the multi-tweet dataset with the removal of the deleted accounts, bots, and double mentions

ton. This is interesting, as the single-tweet baseline had a more favorable ratio for Trump in both volume and vote. Table 5.13 shows the ratios for the single-tweet dataset with the removal of deleted accounts, bots, and tweets which contained mentions of both Trump and Clinton in the text, and Figure 5.19 depicts the volume and vote ratios for the candidates. The removal of all these factors causes a large decrease for Trump in both volume and vote across all thresholds. With the removal of the deleted accounts and the bots, we see a shift in bias from Trump to Clinton. It should be noted that the majority of Clinton's voter demographic ranges from age 18 to 49 [131]. Vote outcome based on exit polls reported a 39:55 split for voters between ages 18 and 29, and a 41:51 split for voters of age 30 to 49 [131]. When using the single-tweet dataset and removing bots we achieve a vote ratio (approximately 38:62

| Threshold | Vol Trump | Vol Clinton | Vote Trump | Vote Clinton | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|---|---|---|---|
| 0.50 | 44.40 | 55.60 | 39.27 | 60.73 | 37.69 | 62.31 | 46.54 | 53.46 |
| 0.55 | 43.70 | 56.30 | 38.55 | 61.45 | 37.83 | 62.16 | 46.80 | 53.20 |
| 0.60 | 43.28 | 56.72 | 38.11 | 61.89 | 37.93 | 62.07 | 46.99 | 53.01 |
| 0.65 | 43.00 | 57.00 | 37.84 | 62.16 | 38.09 | 61.91 | 47.19 | 52.81 |
| 0.70 | 42.80 | 57.20 | 37.66 | 62.34 | 38.19 | 61.81 | 47.30 | 52.70 |
| 0.75 | 42.65 | 57.35 | 37.52 | 62.48 | 38.29 | 61.71 | 47.41 | 52.59 |
| None | 42.28 | 57.72 | 37.17 | 62.83 | 38.51 | 61.49 | 47.69 | 52.31 |

Table 5.12: Single Tweet dataset removed deleted accounts, removed bots

| Threshold | Vol Trump | Vol Clinton | Vote Trump | Vote Clinton | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|---|---|---|---|
| 0.50 | 39.07 | 60.93 | 35.33 | 64.67 | 37.53 | 62.47 | 44.04 | 55.96 |
| 0.55 | 37.52 | 62.48 | 33.80 | 66.20 | 37.66 | 62.34 | 44.30 | 55.70 |
| 0.60 | 36.50 | 63.50 | 32.83 | 67.17 | 37.76 | 62.24 | 44.40 | 55.60 |
| 0.65 | 35.76 | 64.24 | 32.15 | 67.85 | 37.91 | 62.09 | 44.54 | 55.46 |
| 0.70 | 35.22 | 64.78 | 31.67 | 68.33 | 38.01 | 61.99 | 44.58 | 55.42 |
| 0.75 | 34.74 | 65.26 | 31.26 | 68.74 | 38.10 | 61.90 | 44.60 | 55.40 |
| None | 42.28 | 57.72 | 37.17 | 62.83 | 38.51 | 61.49 | 47.69 | 52.31 |

Table 5.13: Single tweet dataset removed deleted accounts, removed bots, and removed double mentions

depending on threshold) that resembles the exit polls. Together, these constraints move the popular vote towards the expected value for the Twitter demographic, as the majority of Twitter users in the U.S. fall under the age of 49 [43]. However, the changes in sentiment are especially large when comparing to the single-tweet baseline, which warrants a deeper look into the bots and the deleted cohort.

### 5.5.3 Sentiment for Candidates

It is important to determine sentiment for each candidate among the accounts considered bots. The tables below depict the positive and negative sentiment for both candidates for the tweets made by the bot accounts. Table 5.14 and Table 5.15 both show the sentiment for Trump and Clinton across the bot accounts in the multi-tweet dataset with removed deleted accounts and, removed deleted accounts and double mentions, respectively. Tables 5.16 and 5.17 show sentiment for bot accounts in the single-tweet dataset with removed deleted accounts and, removed deleted accounts and double mentions, respectively.

It is interesting to note, for the multi-tweet and single-tweet datasets, when removing tweets that mention both candidates the amount of positive tweets regarding

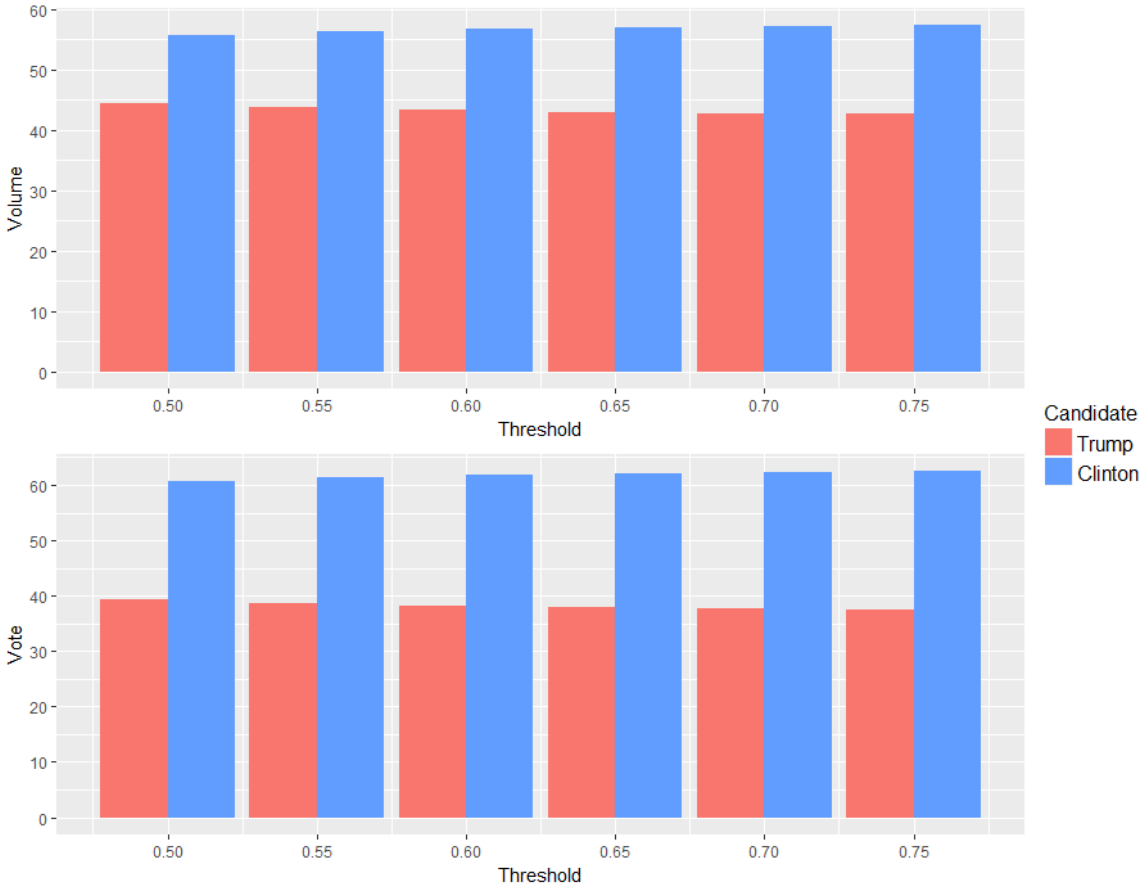



Figure 5.18: Bar graph depicting volume and vote for the single-tweet dataset with the removal of the deleted accounts and bots

| Threshold | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|
| 0.50 | 0.4441 | 0.5559 | 0.4879 | 0.5121 |
| 0.55 | 0.4603 | 0.5397 | 0.5014 | 0.4986 |
| 0.65 | 0.4809 | 0.5191 | 0.5200 | 0.4800 |
| 0.70 | 0.4826 | 0.5174 | 0.5234 | 0.4766 |
| 0.75 | 0.4832 | 0.5168 | 0.5251 | 0.4749 |

Table 5.14: Sentiment for bots in the multi-tweet dataset with removed deleted accounts

Trump and Clinton diminish significantly. This is an important result as it indicates a correlation of higher positive sentiment with tweets that included both candidates. This indicates many tweets mention both Trump and Clinton in the text. As the threshold increases, so does the positive sentiment, with the threshold of 0.75 con-

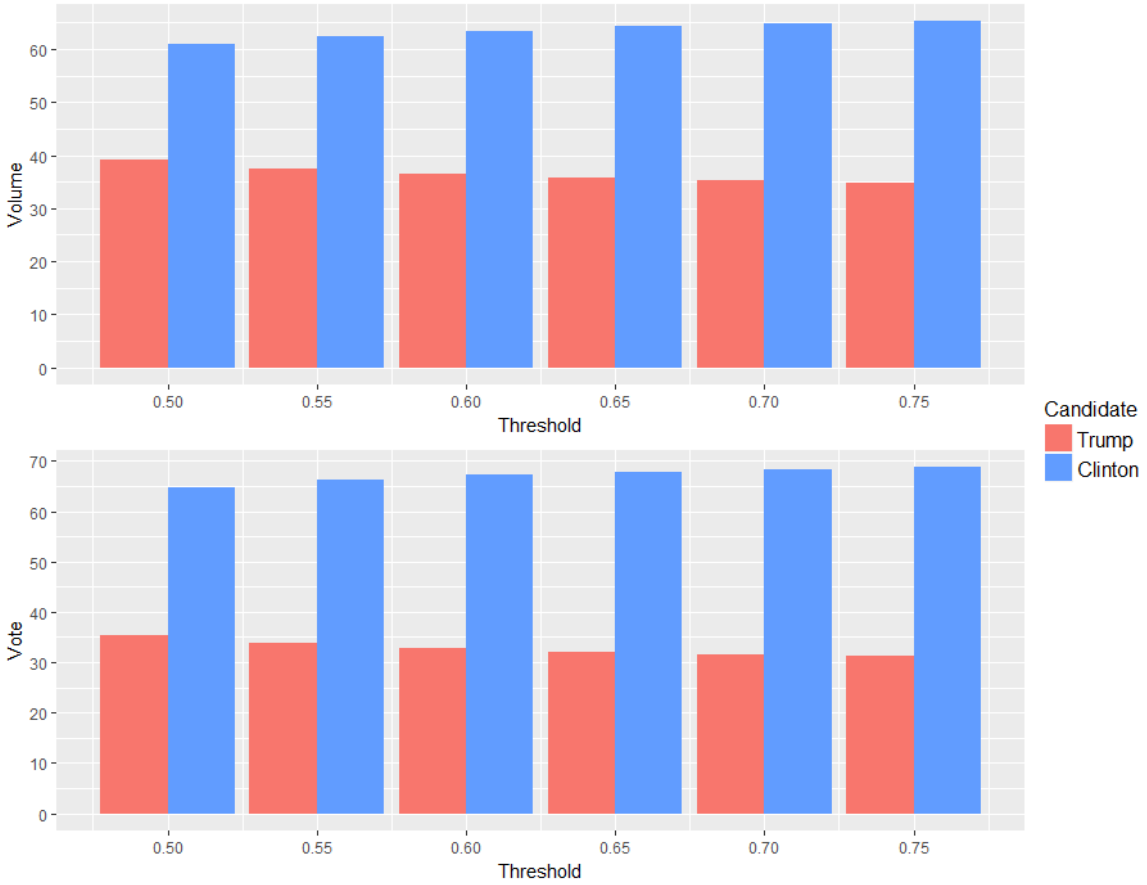



Figure 5.19: Bar graph depicting volume and vote for the single-tweet dataset with the removal of the deleted accounts, bots, and double mentions

| Threshold | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|
| 0.50 | 0.4056 | 0.5944 | 0.4379 | 0.5621 |
| 0.55 | 0.4173 | 0.5827 | 0.4435 | 0.5565 |
| 0.60 | 0.4283 | 0.5717 | 0.4494 | 0.5506 |
| 0.65 | 0.4322 | 0.5678 | 0.4536 | 0.5464 |
| 0.70 | 0.4323 | 0.5677 | 0.4563 | 0.5437 |
| 0.75 | 0.4327 | 0.5673 | 0.4588 | 0.5412 |

Table 5.15: Sentiment for bots in the multi-tweet dataset with removed deleted accounts and double mentions

taining the highest positive sentiment for both candidates. However, in most scenarios there are more negative tweets than positive tweets, especially for Trump. The higher the threshold, the higher likelihood these accounts are social bots, thus the higher

| Threshold | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|
| 0.50 | 0.4795 | 0.5205 | 0.5311 | 0.4689 |
| 0.55 | 0.4929 | 0.5071 | 0.5386 | 0.4614 |
| 0.60 | 0.5038 | 0.4962 | 0.5458 | 0.4542 |
| 0.65 | 0.5079 | 0.4525 | 0.5476 | 0.4921 |
| 0.70 | 0.5152 | 0.4848 | 0.5532 | 0.4468 |
| 0.75 | 0.5172 | 0.4828 | 0.5566 | 0.4434 |

Table 5.16: Sentiment for bots in the single-tweet dataset with removed deleted accounts

| Threshold | Pos Trump | Neg Trump | Pos Clinton | Neg Clinton |
|---|---|---|---|---|
| 0.50 | 0.4176 | 0.5824 | 0.5078 | 0.4922 |
| 0.55 | 0.4284 | 0.5716 | 0.5073 | 0.4927 |
| 0.60 | 0.4379 | 0.5621 | 0.5092 | 0.4908 |
| 0.65 | 0.4397 | 0.5603 | 0.5039 | 0.4961 |
| 0.70 | 0.4458 | 0.5542 | 0.5049 | 0.4951 |
| 0.75 | 0.4478 | 0.5522 | 0.5121 | 0.4879 |

Table 5.17: Sentiment for bots in the single-tweet dataset with removed deleted accounts and double mentions

thresholds may be a better representation of bot behavior. We extracted location data from the accounts labeled (predicted) as bots for each threshold. From Table 5.18, it seems roughly 60% of the bot accounts have location data, with slight variation based on threshold. When further inspecting bot location, we found bot locations to be sparse with no real clustering between accounts, indicating location may not be indicative of whether an account is a bot or not.

In addition to examining the bot accounts, we analyze the deleted cohort to see what kind of tweets exist for these accounts. The characteristics of the deleted cohort in terms of percentage are shown in Table 5.19. There are a majority of Trump instances and a majority of those are positive tweets. As for the Clinton instances, there is an almost 50:50 split between positive and negative tweets. This confirms our previous conclusion, that the majority of instances in the deleted cohort are Trump

| Threshold | With location | Without location | Total |
|---|---|---|---|
| 0.50 | 89,422 | 56,621 | 146,044 |
| 0.55 | 54,639 | 35,377 | 96,729 |
| 0.60 | 37,894 | 25,108 | 67,177 |
| 0.65 | 26,971 | 18,293 | 48,010 |
| 0.70 | 19,286 | 13,417 | 34,774 |
| 0.75 | 12,964 | 9,278 | 24,033 |

Table 5.18: Bot account location information

related.

| | Percentage of Deleted Cohort |
|---|---|
| Trump Instances | 59.17 |
| Clinton Instances | 40.83 |
| Positive Trump | 33.31 |
| Negative Trump | 25.86 |
| Positive Clinton | 20.69 |
| Negative Clinton | 20.14 |

Table 5.19: Deleted Cohort characteristics in terms of percentages

## 5.6 CHAPTER SUMMARY

In this chapter, we determined the effectiveness of Twitter as a polling resource at both the national and state-levels, and we explored the role of social bots on Twitter. Approximately 3 million tweets were collected from 705,381 unique users from September 22nd to November 8th, 2016. This data was labeled using a deep convolutional neural network trained on the sentiment140 dataset. For polling at the national level, the dataset was examined both daily and with eight different time cut-offs. When analyzing at the state-level, twenty-one states were chosen, 11 swing states and 10 favored states. To predict election outcome, two approaches were enacted: winner-take-all and shared elector count. Tweet volume and positive sentiment were utilized as metrics by which to poll or predict the election. In addition, social bots

were detected using the Botometer model and five different thresholds.

We then employ the Botometer model and five different thresholds to classify 570,532 user accounts as bots or real. Based on the threshold set, and whether we remove tweets with double mentions, anywhere from 3% to 25% of all users are classified as bots. The bots are also examined as a separate dataset to determine their polarity towards candidates. In addition, 20% of users belonged to the deleted cohort, which likely contains a high percentage of bots. Excluding the deleted cohort, our results show bots have roughly similar sentiment towards both candidates; with Clinton having more positive tweets from bots and Trump having more negative tweets from bots.

Our findings show that neither sentiment nor volume are predictors of election outcome or polling at the state level. The number of positive tweets for each candidate is about equal regardless of state. This may be due to Twitter demographics, as Twitter may not be a representative sample of the population. Voters in rural areas that may not have access to Twitter and the older population are not well represented [119]. At the national level, positive sentiment aggregated across the full dataset matches the IBD/TIPP poll. However, the differences in distribution between candidates in swing states and favored states, with either volume or sentiment, is not statistically significant using our data and our labeling method.

When taking into account social bots and multiple tweets on a national-level, neither volume nor sentiment reflect the polls. However, the removal of bots, deleted accounts, and multiple tweets from users results in Twitter matching the voting results for the 18-49 demographic. When considering the Twitter platform demographic, the combination of single-tweet with removed bots and the deleted cohort most closely resembles actual election results. These results suggest Twitter is a suitable source of data for polling election results in the specific 18-49 age group.

It should be noted that the deleted cohort plays a large role in the dataset, specif-

ically towards positive sentiment for Trump, as can be seen by comparing the single-tweet to single-tweet deleted dataset. Unfortunately, these accounts cannot be analyzed with Botometer since the accounts no longer exist. However, there is a high likelihood bot accounts were present in this cohort as their removal significantly lowered only the positive tweets for Trump and the removal of these tweets moves public opinion in-line with actual voting patterns of the Twitter demographic. In addition, Twitter has actively been deleting social bot accounts and these accounts may have been deleted due to this.

# CHAPTER 6
# CONCLUSION AND FUTURE WORKS

In this current age of user generated content, text mining plays a key role in machine learning tasks. Twitter is an ever increasing text repository filled with user generated content related to a multitude of topics. Websites like Amazon and Yelp provide a plethora of online user reviews on a countless number of products. The information available from these sources provide insight into various useful topics through the process of sentiment analysis. Whether it be potential presidential candidates or company products, public opinion can be measured by leveraging sentiment analysis and user generated content.

Although the majority of user generated content is legitimate, automated or fake user generated content plays a large role in swaying public opinion. Malicious user generated content comes in various forms; however, social bots and fake reviews seem to be the most prevalent [56, 53]. Social bots aim to breed doubt and/or alter public opinion on social media, their presence may exacerbate or flip actual public opinion. In addition, the presence of social bots may mislead sentiment analysis results as it can skew emotion in different directions. Fake (spam) online reviews are created to harm or bolster the reputation of an establishment or product. As reviews can be posted by any user, it can be difficult to determine whether the review was generated by a reputable source. Determining the validity of a review based solely on text can be nearly impossible for a human.

## 6.1 CONCLUSIONS

The research in this dissertation leverages machine learning for the analysis of social media content and the detection of social bots and online reviews. This dissertation provides an overview of the machine learning techniques used to train and evaluate our models. We improve upon the ability of text mining methods to detect online review spam, explore the effectiveness of cross-domain sentiment analysis using tweets and reviews, determines effectiveness of Twitter at predicting the U.S. 2016 presidential election, and investigates the influence of social bots on Twitter and the election. Model performances are included and statistical testing verify the significance of our findings. From this research, conclusions can be drawn on the need for text mining in our current era of user generated content.

### 6.1.1 Text Mining for the Detection of Fake Reviews

Spam reviews are created to alter the perception of an establishment or product in a positive or negative manner. Due to the open nature of reviews, it can be near impossible to determine whether a review was written by an actual customer or a malicious third party. As determining the validity of a review from a human standpoint can be nearly impossible, text mining needs to be leveraged to detect these spam reviews.

In Chapter 4, text mining for the detection of online spam reviews is explored. The performance of ensemble classifiers, classification using feature selection, and the combination of both is examined for the detection of spam reviews. In this chapter, the performance of Select-Boost and Select-Bagging was evaluated against Random Forest, feature selection, Boosting, and Bagging. To the best of our knowledge, the effects of ensemble classifiers on spam review detection had not been previously explored. As spam reviews make up more than a third of all online reviews, the exploration of advanced machine learning techniques, to further current methods for

the detection of spam reviews, is increasingly necessary [42].

Experimental results show that feature selection has a degradative effect on classification performance when compared to using the full feature set. The performance of the model steadily increases as feature set size increases, although this increase is no longer significant above 900 features when using word frequency. Ensemble techniques, without feature selection, had no significant effect on classification performance for spam detection. However, using Select Boost, a combination of feature selection and boosting, significantly increased classification performance. Select-Boost significantly outperformed the base Multinomial Naïve Bayes classifier, Boosting, Bagging, Select-Bagging, Random Forest with 100 trees, and Random Forest with 250 trees. Select-Boost provides a higher AUC value than all other learners; however, the difference between Select-Boost and Random Forest with 500 trees was not significant.

The use of Select-Boost is recommended over Select-Bagging and Random Forest. The re-weighting step in Select-Boost is crucial in generating an optimized, reduced feature set, allowing for significantly better performance, since each iteration selects a new subset of features that aid in the correct classification of previously misclassified instances.

### 6.1.2 Incorporating Multiple Data Sources for Sentiment Detection

In Chapter 3, the ability of tweets and reviews to be used interchangeably is determined. Cross-domain sentiment analysis using either tweets or reviews and the integration of both for multi-domain sentiment analysis is explored in a two-leg experiment.

For the first part, an empirical study consisting of three datasets and three different classifiers is conducted. Eighteen experiments were performed using 6 combinations of training/test datasets and three classifiers. For the second part, two main experiments

were conducted with two training sets and two testing sets. Each of the training and testing sets consisted of one tweet dataset and one review dataset. This second set of experiments only utilized the Multinomial Naïve Bayes learner as it was found to be the best performing learner in the first leg. Data was sampled into varying dataset sizes ranging from 6,000 to 200,000 equally from both training sets.

Results show when performing cross-domain analysis with classifiers trained using either reviews or tweets perform significantly worse than in-domain classification. Thus, single source sentiment classification is not always ideal if we wish to work in multiple domains; however, by creating an equally sized data set consisting of both tweets and reviews, a model can be trained that performs well at classifying both. The dataset combining tweets and reviews showed significantly better performance than the single source data. In addition, dataset size plays a large role in the performance of classifiers.

### 6.1.3 Mining Social Media for Election Prediction

In Chapter 5, we investigated the effectiveness of Twitter as a polling resource at both the national and state-levels, and we explored the role of social bots on Twitter. For polling at the national level, the dataset was examined both daily and with eight different time cut-offs. For the state-level analysis, twenty-one states were chosen, 11 swing states and 10 favored states. In addition, social bots were detected using the Botometer model and five different thresholds. The botometer system was employed with five different thresholds. Tweets belonging to bot accounts and tweets with double mentions were removed.

Our results show bots have roughly similar sentiment towards both candidates; with Clinton having more positive tweets from bots and Trump having more negative tweets from bots. Our findings also show that neither sentiment nor volume are predictors of election outcome or polling at the state level. At the national level, positive

sentiment aggregated across the full dataset matches the IBD/TIPP poll. However, the differences in distribution between candidates in swing states and favored states, with either volume or sentiment, is not statistically significant using our data and our labeling method. When accounting for social bots and multiple tweets per user on a national-level, neither volume nor sentiment reflect the polls. However, the removal of bots, deleted accounts, and multiple tweets from users results in Twitter matching the voting results for the 18-49 demographic. When considering the Twitter platform demographic, this result most closely resembles actual election results.

It should be noted that the deleted cohort plays a large role in the dataset, specifically towards positive sentiment for Trump, as can be seen by comparing the single-tweet to single-tweet deleted dataset. There is a high likelihood bot accounts were present in this deleted cohort as their removal significantly lowered only the positive tweets for Trump and their removal aligns voting patters with the appropriate demographic. It is highly probable that Twitter deleted these accounts for violation of their Terms of Service.

## 6.2 FUTURE WORK

This dissertation covered the detection of spam reviews, cross-domain sentiment analysis, and election prediction using social media. Future work within these application domains may include:

- For spam review detection, future work needs to establish what types of features are most beneficial as only the text was used in our experiments.
- Future work within cross-domain sentiment analysis may involve extending our methodology for sentiment classification to other domains in order to work towards a broader multi-domain sentiment classifier.
- Future work may also involve cross-referencing known bot accounts to our iden-

tified bot accounts to reaffirm our election prediction results.

- In addition, future work may involve limiting Twitter user accounts based on activity and determining what quantity of those are social bots.

## BIBLIOGRAPHY


[1] Real clear politics - election 2016 presidential polls. https://www.realclearpolitics.com/epolls/latest_polls/. Accessed: 08-2017.

[2] Twitter usage statistics, June, 2015.

[3] Election night 2016 was the most watched in cable news history, Nov 9 2016.

[4] Ibd/tipp tracking poll: The most accurate presidential poll in america, 2016.

[5] U. s. electoral college: Frequently asked questions, 2018.

[6] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[7] N. Abokhodair, D. Yoo, and D. W. McDonald. Dissecting a social botnet: Growth, content and influence in twitter. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work and Social Computing*, pages 839–851, March 2015.

[8] A. Agrawal and A. An. Kea: Sentiment analysis of phrases within short texts. *SemEval 2014*, page 380, 2014.

[9] D. Anuta, J. Churchin, and J. Luo. Election Bias: Comparing Polls and Twitter in the 2016 US Election. *arXiv preprint arXiv:1701.06232*, 2017.

[10] A. Aue and M. Gamon. Customizing sentiment classifiers to new domains: A case study. In *Proceedings of recent advances in natural language processing (RANLP)*, 2005.

[11] J. F. Banzhaf iII. One man, 3.312 votes: a mathematical analysis of the electoral college. *Vill. L. Rev. 13*, page 304, 1968.

[12] M. L. Berenson, M. Goldstein, and D. Levine. *Intermediate Statistical Methods and Applications: A Computer Package Approach 2nd Edition*. Prentice Hall, 1983.

[13] A. Bermingham and A. F. Smeaton. On using Twitter to monitor political sentiment and predict election results. In *Sentiment Analysis where AI meets Psychology (SAAIP) Workshop at the International Joint Conference for Natural Language Processing (IJCNLP)*, 2011.

[14] A. Bessi and E. Ferrara. Social bots distort the 2016 us presidential election online discussion. *First Monday*, 21(11), 2016.

[15] J. Bollen, H. Mao, and A. Pepe. Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. *ICWSM*, page 450453, 2011.

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

[17] A. Boutet, H. Kim, and E. Yoneki. What's in your tweets? I know who you supported in the UK 2010 general election. In *The International AAAI Conference on Weblogs and Social Media (ICWSM)*, 2012.

[18] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, Sept 1996.

[19] R. Breitling and P. Herzyk. Rank-based methods as a non-parametric alternative of the t-statistic for the analysis of biological microarray data. *Journal of bioinformatics and computational biology*, 3(5):1171–1189, oct Oct 2005.

[20] A. Broder. On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings*, pages 21–29, June 1997.

[21] K. Buhrmester and Goslining. Amazon's mechanical turk a new source of inexpensive, yet high-quality data? *Perspectives on Psychological Science*, 6(1):3–5, Jan 2011.

[22] N. Chavoshi, H. Hamooni, and A. Mueen. Debot: Twitter bot detection via warped correlation. In *ICDM*, page 817822, 2016.

[23] X. Chen and M. Wasikowski. Fast: a roc-based feature selection metric for small samples and imbalanced data classification problems. In *Proceedings of the 14th ACM SIGKDD Int'l Conf. Knowldege Discovery and Data Mining (KDD '08)*, pages 124–132, New York, NY, Aug 2008. ACM.

[24] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. Detecting automation of twitter accounts: Are you a human, bot, or cyborg? *IEEE Transactions on Dependable and Secure Computing*, 9(6):811824, 2012.

[25] E. M. Clark, J. R. Williams, C. A. Jones, R. A. Galbraith, C. M. Danforth, and P. S. Dodds. Sifting robotic from organic text: a natural language approach for detecting automation on twitter. *Journal of Computational Science*, 16:1–7, 2016.

[26] N. Confessore, G. J. Dance, and R. Harris. Twitter followers vanish amid inquiries into fake accounts, 2018.

[27] M. Crawford, T. M. Khoshgoftaar, and J. D. Prusa. Reducing feature set explosion to facilitate real-world review spam detection. In *Proceedings of the 29th International FLAIRS conference*, pages 304–309, May 2016.

[28] M. Crawford, T. M. Khoshgoftaar, J. D. Prusa, A. N. Richter, and H. Al Najada. Survey of review spam detection using machine learning techniques. *Journal Of Big Data*, 2(1):1–24, Dec 2015.

[29] C. A. Davis, V. Onur, E. Ferrara, A. Flammini, and F. Menczer. Botornot: A system to evaluate social bots. In *International World Wide Web Conferences Steering Committee*, pages 273–274, April 2016.

[30] X. Deng and R. Chen. Sentiment analysis based online restaurants fake reviews hype detection. In *Asia-Pacific Web Conference*, pages 1–10. Springer, 2014.

[31] J. P. Dickerson, V. Kagan, and V. S. Subrahmanian. Using sentiment to detect bots on twitter: Are humans more opinionated than bots? In *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*, pages 620–627, 2014.

[32] T. G. Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.

[33] D. J. Dittman, T. M. Khoshgoftaar, R. Wald, and J. Van Hulse. Comparative analysis of dna microarray data through the use of feature selection techniques. In *Proceedings of the Ninth IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 147–152. ICMLA, 2010.

[34] S. Dixit and A. Agrawal. Survey on review spam detection. *International Journal of Computer & Communication Technology*, 4(2):68–72, June 2013.

[35] R. Duda and R. Singleton. Training a threshold logic unit with imperfectly classified patterns. Technical report, STANFORD RESEARCH INST MENLO PARK CALIF, 1944.

[36] E. Ferrara. Disinformation and social bot operations in the run up to the 2017 french presidential election. 2017.

[37] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini. The rise of social bots. *Communications of the ACM*, 59(7):96–104, 2016.

[38] G. Forman. An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.*, pages 3:1289–1305, 2003.

[39] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, pages 148–156, Jan 1996.

[40] C. H. E. Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14). Available at (20/04/16) http://comp. social. gatech. edu/papers/icwsm14. vader. hutto. pdf*, 2014.

[41] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12, 2009.

[42] I. C. Government of Canada. Dont buy into fake online endorsements not all reviews are from legitimate consumers, July 2014.

[43] S. Greenwood, R. Perrin, and M. Duggan. Social media update 2016, 2016.

[44] C. Grimme, M. Preuss, L. Adam, and H. Trautmann. Social bots: Human-like by means of human control? *Big Data*, 5(4):279–293, 2017.

[45] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009.

[46] A. Hannak, E. Anderson, L. F. Barrett, S. Lehmann, A. Mislove, and M. Riedewald. Tweetin'in the rain: Exploring societal-scale effects of weather on mood. In *ICWSM*, 2012.

[47] S. Haykin. *Neural Networks: A Comprehensive Foundation 2nd edition*. Prentice Hall, 1998.

[48] R. He and J. McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, pages 507–517. International World Wide Web Conferences Steering Committee, 2016.

[49] B. Heredia. Effects of gene selection and data sampling on prediction of breast cancer treatments. Master's thesis, 2014.

[50] B. Heredia, T. M. Khoshgoftaar, A. Fazelpour, and D. J. Dittman. Building an effective classification model for breast cancer patient response data. In *2015 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 229–235, 2015.

[51] B. Heredia, T. M. Khoshgoftaar, J. D. Prusa, and M. Crawford. Cross-domain sentiment analysis: An empirical investigation. In *2016 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 160–165, 2016.

[52] B. Heredia, T. M. Khoshgoftaar, J. D. Prusa, and M. Crawford. Integrating multiple data sources to enhance sentiment prediction. In *Collaboration and Internet Computing (CIC), 2016 IEEE 2nd International Conference*, pages 285–291, 2016.

[53] B. Heredia, T. M. Khoshgoftaar, J. D. Prusa, and M. Crawford. Improving detection of untrustworthy online reviews using ensemble learners combined with feature selection. *Journal of Social Netw. Analys. Mining*, 7(1):37, 2017.

[54] B. Heredia, T. M. Khoshgoftaar, J. D. Prusa, and M. Crawford. An investigation of ensemble techniques for detection of spam reviews. In *Machine Learning and Applications (ICMLA), 2016 15th International Conference on*, pages 127–133, Dec 2016.

[55] B. Heredia, J. D. Prusa, and T. M. Khoshgoftaar. Exploring the effectiveness of twitter at polling the united states 2016 presidential election. In *Collaboration and Internet Computing (CIC), 2017 IEEE 3rd International Conference*, pages 283–290, 2017.

[56] B. Heredia, J. D. Prusa, and T. M. Khoshgoftaar. The influence of social bots on twitter as a polling source for election outcomes. *Journal of Social Netw. Analys. Mining*, Under Review, 2018.

[57] B. Heredia, J. D. Prusa, and T. M. Khoshgoftaar. Location-based twitter sentiment analysis for predicting the u.s. 2016 presidential election. In *The Thirty-First International FLAIRS Conference*, pages 265–270. AAAI, 2018.

[58] B. Heredia, J. D. Prusa, and T. M. Khoshgoftaar. Social media for polling and predicting united states election outcome. *Journal of Social Netw. Analys. Mining*, 2018.

[59] G. E. Hinton, J. L. McClelland, and D. E. Rumelhart. Distributed representations. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1(3):77109, 1986.

[60] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. A practical guide to support vector classification. pages 1–16, 2003.

[61] N. Jindal and B. Lui. Analyzing and detecting review spam. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, page 547552, 2007.

[62] N. Jindal and B. Lui. Review spam detection. In *Proceedings of the 16th international conference on World Wide Web*, page 11891190, 2007.

[63] N. Jindal and B. Lui. Opinion spam and analysis. In *In: Proceedings of the 2008 International Conference on Web Search and Data Mining*, Feb 2008.

[64] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[65] D. W. H. Jr and S. Lemeshow. *Applied Logistic Regression*. John Wiley & Sons, 2004.

[66] T. M. Khoshgoftaar and E. B. Allen. Logistic regression modeling of software quality. *International Journal of Reliability, Quality and Safety Engineering*, 6(04):303–317, 1999.

[67] T. M. Khoshgoftaar, D. J. Dittman, R. Wald, and A. Fazelpour. First order statistics based feature selection: A diverse and powerful family of feature selection techniques. In *Proceedings of the Eleventh International Conference on Machine Learning and Applications (ICMLA)*, pages 151–157. ICMLA, Dec 2012.

[68] T. M. Khoshgoftaar, M. Golawala, and J. Van Hulse. An empirical study of learning from imbalanced data using random forest. In *Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE international conference on*, volume 2, pages 310–317. IEEE, 2007.

[69] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes. Multinomial naive bayes for text categorization revisited. In *Australasian Joint Conference on Artificial Intelligence*, pages 488–499. Springer, 2004.

[70] E. Kouloumpis, T. Wilson, and J. Moore. Twitter sentiment analysis: The good the bad and the omg! *ICWSM*, 11:538–541, 2011.

[71] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, page 10971105, 2012.

[72] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436, 2015.

[73] J. Li, O. Myle, C. Cardie, and E. Hovy. Towards a general rule for identifying deceptive opinion spam. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1556–1576, June 2014.

[74] A. Liaw and M. Wiener. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[75] B. Liu. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, pages 1–167, 2012.

[76] Y. Liu, X. Huang, A. An, and X. Yu. Arsa: A sentiment-aware model for predicting sales performance using blogs. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 607–614, 2007.

[77] M. Locklear. Twitter says most recent follower purge is about bots, not politics, 2018.

[78] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.

[79] C. D. Manning. *Foundations of statistical natural language processing*. MIT press, 1999.

[80] J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2015.

[81] A. McCallum and K. Ni-gam. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, Jul 1998.

[82] C. Meador and J. Gluck. Analyzing the relationship between tweets, box-office performance and stocks. *Methods*, 2009.

[83] F. Morstatter, L. Wu, T. H. Nazer, K. M. Carley, and H. Liu. A new approach to bot detection: striking the balance between precision and recall. In *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM*, page 533540, 2016.

[84] A. Mukherjee, V. Venkataraman, B. Liu, and N. Glance. What yelp fake review filter might be doing? In *Seventh International AAAI Conference on Weblogs and Social Media*, June 2013.

[85] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic. Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1):1, 2015.

[86] B. L. Ott. The rise of social bots. *Critical Studies in Media Communication*, 34(1):59–68, 2017.

[87] M. Ott, Y. Choi, C. Cardie, and J. Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 309–319, June 2011.

[88] A. Pak and P. Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, 2010.

[89] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, pages 1–135, 2008.

[90] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[91] H. Peng, L. Long, and C. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, pages 1226–1238, June 2005.

[92] J. Peterson. Trumps twitter followed by millions of inactive or fake accounts, 2016.

[93] J. Prusa, T. M. Khoshgoftaar, D. J. Dittman, and A. Napolitano. Using random undersampling to alleviate class imbalance on tweet sentiment data. In *Information Reuse and Integration (IRI), 2015 IEEE International Conference on*, pages 197–202. IEEE, 2015.

[94] J. Prusa, T. M. Khoshgoftaar, and A. Napolitano. Utilizing ensemble, data sampling and feature selection techniques for improving classification performance on tweet sentiment data. In *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*, pages 535–542. IEEE, 2015.

[95] J. Prusa, T. M. Khoshgoftaar, and N. Seliya. The effect of dataset size on training tweet sentiment classifiers. In *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*, pages 96–102. IEEE, 2015.

[96] J. D. Prusa and T. M. Khoshgoftaar. Comparing approaches for combining data sampling and feature selection to address key data quality issues in tweet sentiment analysis. In *FLAIRS Conference*, pages 608–613, 2016.

[97] J. D. Prusa and T. M. Khoshgoftaar. Designing a better data representation for deep neural networks and text classification. In *Information Reuse and Integration (IRI), 2016 IEEE 17th International Conference on*, pages 411–416. IEEE, 2016.

[98] J. D. Prusa and T. M. Khoshgoftaar. Improving deep neural network design with new text data representations. *Journal of Big Data*, 4(1):7, 3 2017.

[99] J. D. Prusa and T. M. Khoshgoftaar. Training convolutional networks on truncated text. In *Tools with Artificial Intelligence (ICTAI), 2017 IEEE 29th International Conference on*, pages 330–335. IEEE, 2017.

[100] J. D. Prusa, T. M. Khoshgoftaar, and D. J. Dittman. Using ensemble learners to improve classifier performance on tweet sentiment data. In *2015 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 252–257, 2015.

[101] J. D. Prusa, T. M. Khoshgoftaar, and D. J. Dittman. Impact of feature selection techniques for tweet sentiment classification. In *Proceedings of the 28th International FLAIRS conference*, pages 299–304, May 2015.

[102] J. D. Prusa, T. M. Khoshgoftaar, and A. Napolitano. Necessity of feature selection when augmenting tweet sentiment feature spaces with emoticons. In *FLAIRS Conference*, pages 614–620, 2016.

[103] J. D. Prusa, T. M. Khoshgoftaar, and A. Napolitano. Using feature selection in combination with ensemble learning techniques to improve tweet sentiment classification performance. In *Proceedings of the 27th International Conference on Tools with Artificial Intelligence*, pages 186–193, Nov 2015.

[104] J. D. Prusa, T. M. Khoshgoftaar, and N. Seliya. Enhancing ensemble learners with data sampling on high-dimensional imbalanced tweet sentiment data. In *The Twenty-Ninth International FLAIRS Conference*, pages 322–327, 2016.

[105] J. R. Quinlan. *C4.5: programs for machine learning*. Elsevier, 2014.

[106] R. J. Quinlan. *C4.5: Programs for Machine Learning*. Elsevier, June, 2014.

[107] J. Ratkiewicz, M. D. Conover, M. R. Meiss, B. Gonalves, A. Flammini, and F. Menczer. Detecting and tracking political abuse in social media. *International Conference on Weblogs and Social Media*, 11:297304, 2011.

[108] A. N. Richter, M. Crawford, B. Heredia, and T. M. Khoshgoftaar. Efficient modeling of user-entity preference in big social networks. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 982–988, 2015.

[109] I. Rish. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, Nov 2001.

[110] S. Rosenthal, P. Nakov, A. Ritter, and V. Stoyanov. Semeval-2014 task 9: Sentiment analysis in twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, 2014.

[111] H. Saif, Y. He, and H. Alani. Alleviating data sparsity for twitter sentiment analysis. In *2nd Workshop on Making Sense of Microposts*, 2012.

[112] H. Saif, Y. He, and H. Alani. Semantic sentiment analysis of twitter. In *The Semantic Web–ISWC 2012*, pages 508–524. Springer, 2012.

[113] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[114] W. S. Sarle. Neural networks and statistical models, 1994.

[115] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano. Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 40(1):185–197, January 2010.

[116] N. Seliya, T. M. Khoshgoftaar, and J. Van Hulse. A study on the relationships of classifier performance metrics. In *Proceedings of the 21st IEEE International Conference on Tools with Artifical Intelligence (ICTAI'09)*, pages 59–66, Newark, NJ, November 2009. IEEE Computer Society.

[117] S. Shojaee, M. Murad, N. Sharef, and S. Nadali. Detecting deceptive reviews using lexical and syntactic features. In *2013 13th International Conference on Intelligent Systems Design and Applications (ISDA)*, Dec 2013.

[118] J. Si, A. Mukherjee, B. Liu, Q. Li, H. Li, and X. Deng. Exploiting topic based twitter sentiment for stock prediction. *ACL (2)*, page 2429, 2013.

[119] L. Sloan, J. Morgan, P. Burnap, and M. Williams. Who tweets? deriving the demographic characteristics of age, occupation and social class from twitter user meta-data. *PloS one*, page e0115545, 2015.

[120] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[121] H. Staff. Electoral college, 2010.

[122] S. Stieglitz, F. Brachten, D. Berthel, M. Schlaus, C. Venetopoulou, and D. Veutgen. Do social bots (still) act different to humans?comparing metrics of social bots with those of humans. In *International Conference on Social Computing and Social Media*, page 379395, 2017.

[123] V. Subrahmanian, A. Azaria, S. Durst, V. Kagan, and A. Galstyan. The darpa twitter bot challenge. *Computer*, 49(6):3846, 2016.

[124] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. *ICWSM*, 10(1):178–185, 2010.

[125] V. G. Tusher, R. Tibshirani, and G. Chu. Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Sciences*, 98(9):5116–5121, 2001.

[126] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano. Experimental perspectives on learning from imbalanced data. In *Proceedings of the 24th international conference on Machine learning*, pages 935–942. ACM, 2007.

[127] O. Varol, E. Ferrara, C. A. Davis, F. Menczer, and A. Flammini. Online human-bot interactions: Detection, estimation, and characterization. *arXiv preprint arXiv:1703.03107*, 2017.

[128] O. Varol, E. Ferraram, C. A. Davis, F. Menczer, and A. Flammini. Online human-bot interactions: Detection, estimation, and characterization. *arXiv:1703.03107v2 [cs.SI]*, 2017.

[129] H. Wang, D. Can, A. Kazemzadeh, F. Bar, and S. Narayanan. A system for real-time twitter sentiment analysis of 2012 us presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations*, pages 115–120. Association for Computational Linguistics, 2012.

[130] G. M. Weiss and F. J. Provost. Learning when training data are costly: the effect of class distribution on tree induction. *J. Artif. Intell. Res.(JAIR)*, 19:315–354, 2003.

[131] K. Weldon. How groups voted 2016, 2016.

[132] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2nd edition, 2005.

[133] F. Yergeau. Utf-8, a transformation format of iso 10646, 2003.

[134] M. Zamani and M. Movahedi. Machine learning techniques for intrusion detection. *arXiv preprint arXiv:1312.2177*, 2013.

[135] H. Zhang. The optimality of naive bayes. *AA*, 1:3, 2004.

[136] H. Zhang. The optimality of nave bayes. In *Proceedings of the 17th International Florida Artificial Intelligence Research Society (FLAIRS) Conference*, pages 562–567. AAAI, 2004.

[137] X. Zhang and Y. LeCun. Text understanding from scratch. Technical report, Cornell University, 02 2015.