Predicting News Bias

This report was written as part of the requirements for SENG 474 / CSC 578D: Data Mining, offered at the University of Victoria, taught by Dr. Alona Fyshe.

Ali Dehghan*, Lloyd Montgomery†, Maryi Arciniegas-Mendez‡ and Maria Ferman-Guerra§
Department of Computer Science, University of Victoria
Victoria, BC, Canada
Email: *ali.dehghan.uvic@gmail.com, †lloydrmontgomery@gmail.com, ‡maryia@uvic.ca, §tania.ferman@gmail.com

***Abstract*—A variety of different media sources are used everyday to disseminate information about what is happening around the world. These information sources are expected to offer an objective perspective about the topic of interest, however, it is well recognized that every news broadcasting agency has its own interests in mind. In particular, differences in the information presentation around so called controversial topics (e.g., gay marriage, gun control) tend to expose a distinctive type of bias in several news agencies: a political bias. In cases like this, the perspective offered by the news agency is aligned with the agenda of the political party of preference, either liberal or conservative. Although this issue is well recognized by a knowledgeable audience, there is no automatic way to detect the bias. In order to solve such a problem, this project aims to use data mining techniques to create a system able to detect political bias in news articles by predicting the ADA score of the source agency. Initially, the project focuses on well known controversial topics for political parties. This work is divided in three phases. First, we create a dataset, with the use of Twitter as a topic indexer, from ten agencies around five topics, we select ten articles for each for a total of five hundred news stories. Second, we train our algorithm with the use of only eight agencies. For the third phase, we use the two agencies left out in the training phase for validation. Finally, we describe our results and provide some recommendations for future work.**

I. Related work

This section describes the most representative research about data mining in media sources, and the techniques and tools commonly used to detect it.

A. Data mining in media sources

In the psychology domain, *bias* is viewed often as *"adaptive adjustments to overload of information, where people need various shortcuts and heuristics to be able to cope effectively with their social environment"* [1]. When everyday we are presented with social information, we have two systems to process it. In the first one, we can have a more accurate and analytic idea about a subject, however, to achieve such results it is required a bigger investment of time and mental effort. This system requires us to analyze all relevant information about a topic to get unbiased information. In the second system, we could use stereotypes or preconceived ideas. Not surprisingly, people tend to use the second system because is faster and require less mental effort. Thus, we are highly influenceable by biased information unless we put more attention and time to analyze our sources. Specifically, bias can be more problematic and particularly harmful when present in social media. Media bias allows an influential person or an organization to reach a large amount of the population and manipulate information in order to persuade a certain sector to choose a specific idea or to acquire a certain behavior[1].

The analysis of bias in social media is not new to the research community. In the work of D'Alessio and Allen [5] researchers analyzed political bias from media sources in a presidential election campaign. According to the authors, the campaign is short enough for getting all relevant data, such as activities and events, and all data can be gathered from news, which ensures a complete set. This study classifies media bias in three types. The first type of media bias is related with a *gatekeeping bias*; usually performed by journalist and editors who make a selection of articles that will not be published. The second type is *coverage bias*, where media is codified by the amount of coverage to a particular event. Finally, the third type of bias is *statement bias*, in which case the content produced by media is favorable or unfavorable towards certain events. Surprisingly, this study did not find evidence of media bias and concluded that the information produced by the media was homogeneous. Despite the results of D'Alessio and Allen we believe that media bias exist and we refine our approach with respect to theirs by selecting data about specific topics from news agencies in an open time frame and analyzing only statement bias in the media sources.

In a more recent study, we found more hope in using data mining to detect new bias. The work of Segev and Miesch proved the existence of news bias towards Israel and identified Britain as the source of the most negative bias towards the foreign country [15]. In fact, our methodology is similar to the one followed in this study which makes us confident in our process and in the results.

Another way to define bias in media sources is called framing. According to Card et. al. [3] an example of this is how the media shapes political debates by taking some parts of the speech and making them more or less salient. In their work, researchers aimed to show how the language of a certain article relates to framing. They divided the articles into 15 framing dimensions, including economics, morality and politics, which were later used to label the article headline and the content.

On a first version of the process three policy issues were chosen: immigration, smoking and same-sex marriage. On the last stage of the study, researchers showed that disagreements are present even in well-trained annotators with high amount of feedback about their inter-reliability. These results showed that the same article could be interpreted differently by different users. In sum, the researchers underline the importance of framing to understand not only how an article can be written, but also what effect it may cause.

The goal of this project is to be able to classify news articles by the degree to which their context is aligned with a particular political party. In this regard, data mining techniques have also been used in news archives for classification purposes. In one example of such work, researchers developed a system to classify stories based on 350 different codes [12]. This coding work is usually done by the editors, however, the rapid development of news and the size of the coding scheme makes the task challenging for a human. In this research, the dataset with 50,000 stories was taken from Dow Jones Press Release News Wire and the method used in the classification system was a variation of Memory Based Reasoning (MBR). This algorithm assigned codes to new unseen stories by finding new matches from the training database and then choosing the best few codes based on a confidence threshold. The final system performed with good recall and precision for news classification which represented an excellent alleviation for editorial work.

In another example of classification system, researchers explored the feasibility of divided segments of news according to its bias [16]. The work of Vavpetic et al introduced a new system named Hedwig, which divides news article text into subgroups and then label them with descriptions based on an ontology vocabulary. This project used the background knowledge in the form of Resource Description Framework triples (RDF). As in our case, this study also had the purpose to find vocabulary patterns to enable classification, however, this work was carried out on a financial domain to discover credit trends.

The first phase of our project involves the creation of a database with news articles from different sources. It is not the first attempt to create a news aggregation platform and many of them have been created before. Columbia's Newsblaster is an excellent example of a tracking and a summarizing news system [13]. However, this web-crawling application always crawls information from the same list of sources, regardless of their political bias. And the final product, a computer-generated news story, is an aggregated article that has the potential of a political bias depending on the sources it was built from. In another important work, researchers from Google reported the mechanism used in their news recommendation system [6]. In this case, the purpose for the final users defined the main difference between this news aggregation system and ours. In particular, Google news aggregates news articles from more than 4,500 news sources worldwide and has more than several million unique users for which, using collaborative filtering, they generate personalized recommendations; in other words, Google selects news content based on users preference. In our project, we will classify the sources by the degree to which they are likely to be aligned to a political party and the user will decide which articles to pay attention to.

B. Data mining techniques and tools

In this subsection we will provide background on the data mining techniques and tools that will be used in the project.

Research using Twitter API has been mainly focused on the analysis of individual Tweets or in the aggregated properties of a collection of those. In particular, political preference of Twitter users has been studied with different approaches. One work introduces a method for computing political preferences of an organization's Twitter followers using congressional liberal/conservative American for Democratic Action (ADA) scores as a starting point for scoring [7]. In another case, researchers predicted political alignment using models based on a variety of contextual and behavioral features of users on the website [10]. While other study used Supported Vector Machines (SVM) trained with manually coded data for the same purpose [4]. Interestingly, we did not find studies on Twitter that analyze the bias of published content.

Nowadays, there are several ways of gathering information and getting the sentiment analysis of a writer's sentiment about a certain article, blog or just written text. There is an increased use of sentiment analysis due to its importance in providing information about a specific topic.The examples are very varied, such as market and politics research, financial investments and government's national security. Therefore, it is significantly important to have efficient techniques to gather the data. These techniques need to deal with issues such as information noise in the data set, and improve the accuracy of the gathered data.

Term Frequency Inverse Document Frequency (TF-IDF) is a common technique to compute how important a word is to a document. To assign an importance weight to each word in a document, TF-IDF takes into account the inverse proportion of the frequency of the word. In fact, Ramos work [14] provides evidence of the superiority of TF-IDF among Naive Bayes query retrieval. In particular, this work highlights key characteristics in this technique. Two main strengths: TF-IDF efficiency in matching relevant words of a document and its high performance with complicated algorithms. And one main limitation; TF-IDF is not able to make any relationship between synonyms (e.g., the word priest and reverend).

In a more recent study, Martineau and Finin introduced a new technique that improves sentiment classification [11]. In this work, researchers implemented a feature that uses a novel techniques derived from TF-IDF to weight words in different datasets. The technique is called Delta TF-IDF, and uses the difference between the words' TF-IDF score in the positive and negative datasets in order to get a better weight word score accuracy. Validation of this approach showed a significantly better classification accuracy with a 99.9% confidence interval in the classification.

According to Yu et al. [18] classifying opinions in a political context can have a number of difficulties. The first issue, it is that in the political domain people use less sentiment words than in others sources (e.g., movie reviews dataset). The second problem is the nouns used in the political context. Indeed, politicians deliberately use medical or technical terms in order to avoid more emotive words. Finally, the third difficulty is in the classification itself. In this work, researchers used the bag of words approach to get a word frequency vector. Then, Supporting Vector Machines (SVM) and Naive Bayes were used to train the classifiers. As the main contribution, researchers showed that, despite the particularities on political discourse, algorithms can be effectively trained to classify information in this context.

Sentiment analysis is used to determine whether a certain document has a positive or negative sentiment about an issue. There are several research that demonstrate that the sentiment analysis classifier can be improved by combining the classification of certain documents with its level of agreement. One excellent example is the work of Burfoot [2]. In this work, researchers describes previous approaches in which the sentiment classification in the United Stated congressional debates includes information about the agreement between speakers. Certainly, this work points out at an interesting direction for future work; can the classification of news bias be improved by adding the level of agreement of news agencies to the algorithm?

In terms of Sentiment analysis it is still hard to classify human opinions. One of the issues of sentiment analysis, is the problem of ambiguity, in which researchers need to determine whether a certain concept belongs to a sentiment term in a specific context. The article of Weichselbraun et al. [17] describes a new method to address the problem of sentiment analysis ambiguity by using contextualized sentiment lexicons. The contextualized sentiment lexicon is generic enough to be used across different domains. What's more, the sentiment lexicon addresses the ambiguity problem by describing three different types of context. First, is the "helpful terms" by which the Naive Bayes classifier has an aid in classifying the reviews when the baseline fails. The second is the "neutral terms" which are terms that do not fall into any classification (e.g., in a movie review of 5 stars, one star represents a bad review, five stars represent a good review and three represents a neutral review). Finally, we have the "harmful terms", which negatively affects the performance of the naive Bayes classifier creating a misclassification of the reviews. This study showed that this method can be applied to diverse decision support applications and opinion mining. In a similar study, Kaya and Conley analyzed the accuracy of sentiment analysis by using a tailored sentiment lexicon [9]. This variation showed had a cleared predicting performance.

As described in this section, research in News archives using data mining techniques have been largely explored. However, as far as we know, there is no approach so far that explores the political bias presented in news agencies (e.g. CNN, CBC).

II. Proposed Project

We have a basic need to know what is going on around us. To stay informed about the latest events in our world we use a large variety of media sources. Ideally, these sources of information should be objective, but it is widely recognized that they are not. Specifically, news agencies are recognized for having political parties of preference which influences the way certain information is reported to the public. So, depending on the political agenda news stories are adapted for presentation in what represents a clear political bias in the information the audience receives.

What's more, there are certain agencies whose political preference is openly known (e.g. Fox News is very right-wing). Consequently, there is a high risk of misinformation for the youth and for the not-so-experienced audience. In today's world with sources of information everywhere we believe it is important to be able to identify if events are reported from an objective perspective or are influenced by a political secondary interest. However, as far as we know, there is no automatic way to say whether or not a news article presents a political bias. Motivated by this issue, we use data mining techniques to create a system able to detect political bias. In particular, we explored algorithms to predict the ADA score of the news agency and the political alignment: conservative, liberal or neutral. In the following subsections we describe the project phases and details of implementation.

A. Phase I: Dataset Creation

Political differences between liberal and conservatives parties can be found in many topics; furthermore, there are many topics considered controversial where their opinions on the matter are very different (e.g., gay marriage, abortion). We expect articles about these topics to provide a significant diversity required for training our model and a better baseline for future classification. So, we purposefully selected a list of the 5 most controversial topics [18] to be used in our dataset creation. Selected topics are presented in table I

Topics
Abortion
Gun control
Gay marriage
Climate change
Immigration

TABLE I
CONTROVERSIAL TOPICS IN THE POLITICAL SCENARIO

In addition, we selected a list of ten news agencies with known political preference based on Congress peoples' American for Democratic Action (ADA) scores. These scores vary from 0 to 100. A score of 0 means complete disagreement with ADA policies which indicates the news agency to be most conservative. While a score of 100 means complete agreement with ADA policies thus the news agency is the most liberal [8].

Also, we considered only agencies with active Twitter accounts, where we could find links to their news stories about

our topics. Moreover, we made sure that the selected sources had ADA scores varying from low values to high values to avoid bias in our model. Selected news sources are presented in table II. We collected 10 stories from each source for each selected topic.

Name	Twitter ID	Website	Adjusted ADA score	Total follow-ers
Washington Times	washtimes	washington times.com	35.4	258k
Fox News	FoxNews	foxnews.com	39.7	8.3m
PBS NewsHour	NewsHour	newshour.pbs .org	55.8	763k
CNN	cnn	cnn.com	56.0	23.5m
ABC Good Morning America	gma	abcnews.go .com	56.1	3m
USA Today	usatoday	usatoday.com	63.4	2m
U.S. News and World Report	usnews	usnews.com	65.8	112k
Los Angeles Times	latimes	latimes.com	70.0	1.8m
CBS News	CBSNews	cbsnews.com	73.3	4.3m
Wall Street Journal	wsj	wsj.com	85.1	10m

TABLE II
NEWS AGENCIES INFORMATION

Twitter [1] is a very popular social networking tool to share short posts called 'Tweets', which we used as a topic/hyper-link indexer for web-crawling purposes. News sites do offer high-level classifications of their own stories, but we cannot guarantee that these labels always exist, and these categories are likely to vary from site to site. Additionally, obtaining stories based on their classifications per news site requires web-crawlers to traverse hundreds of pages per site looking through their categories. This is a task that can be avoided through the use of Twitter. Using Twitter, we can return all the Tweets under a given topic, which allows our algorithm to automatically classify the Tweets it receives. By only choosing Tweets that contain a link to a news article, the algorithm can then crawl the news story directly from that link, without the need to traverse extra pages of the news site. In addition, we only query for Tweets from selected Twitter accounts that we already know belong to actual news agencies, such as CNN and Fox News. As a final step in our dataset creation we clean up the data and leave it ready to be used in the next phase. Figure 1 illustrates the logical schema of our dataset.

B. Phase II: Algorithm Training

For initial training, we select data from only one topic and leave out two agencies for a later testing process. With the working data we tokenized the text of the articles to remove stopwords and then run a keyword frequency analysis, TF-IDF, to find a list of keywords that are more important for the topic given each source. Then, we train the algorithms to predict either the political party to which the news agency is aligned or the ADA score. We repeat this process for all agencies except two; testing is done later with the agencies left out.

[1]https://twitter.com/

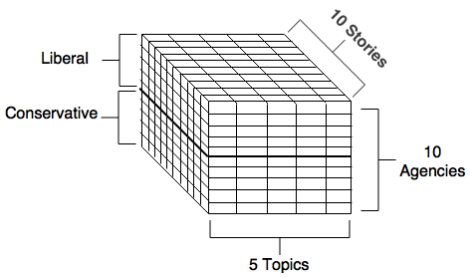

Fig. 1. Dataset Diagram

C. Phase III: Validation and Analysis

Once the algorithm has been trained, we select articles on the same topic from the two agencies we left out, and we use them for testing. The algorithms take in the articles about the related topic and predict the political bias of the news agencies. We use linear regression to predict ADA scores and logistic regression to predict political alignment.

III. PROJECT IMPLEMENTATION

In this section we present our implementation of the project in three phases. Dataset creation, Algorithm training, and Validation and analysis.

A. Phase I: Dataset creation

The dataset creation algorithm consist of three main steps: (1) gather the Tweets from Twitter, (2) follow the links in each Tweet to the underlying news source to grab the textual story, and (3) write all of this information to a database.

The database was created using MongoDB. In creating a dataset, we were given the freedom to define how it was going to be created. This meant a lot of decisions early on such as *"what news sources will we use?"* and *"how many stories from each source?"*. After discussions amongst ourselves, with our professor, and after many iterations of the project, we decided on the following attributes:

- 5 Topics
  - 1 topic trained at a time with the classifier. We will replicate the process until we complete the 5 topics of our dataset.
- 8 News Sources
  - 4 weighted towards conservative, and 4 weighted towards democrat.
  - 6 sources used for training, and the remaining 2 for testing. Equal political split for both training and testing
- 10 Stories from each News Source per topic

In other words, **for each topic**, we search for Tweets from particular news sources, and select 10 stories from each news

source. Thus we get 80 stories per topic, and 400 stories in total if we run the algorithm across all topics.

The core of the functionality is in the collection of Tweets. We opted to not use the official Twitter API after discovering a few limitations imposed on us: max 100 Tweets per call to the API and a max 450 calls per 15 minutes. In addition, the Tweets returned by the Twitter API are limited to the 7 days prior to when the query is executed. The last limitation meant that if we wanted to build our dataset of 500 news stories around 5 topics, we would have have to rely on Tweets only from last week, which is not desirable. Therefore, we chose to use a Twitter web-scraper written by a fellow developer and published through GitHub [2] instead. Due to the nature of web-scraping – and a lack of policing by Twitter – this new code does not hold limitations about query reply size, number of queries, or time-frame of the Tweets. The code we forked allowed us to select any time frame and retrieve the data we needed.

The following process was applied for each topic. **For each news source**, we ran a while loop to continuously scrape Twitter until we got the 10 stories we needed. We defined a *'valid Tweet'* as a Tweet with a link to our target news site. If a single attempt did not return 10 valid Tweets then we scraped Twitter again, changing the time-frame to be a month earlier. Once the 10 stories on the given topic had been found, we moved on to the next news source.

Through this process, we stored all information in JSON format, in the following structure:

```
{
'topic',
'source',
'link',
'title',
'content',
'twitter' : {
  'body',
  'id',
  'link',
  'date',
  'favs',
  'retweets',
  'hashtags',
  'mentions'
  }
}
```

Through the above process, we were able to fill in each of the fields nested under 'twitter', as well as the 'topic', 'source', and 'link' fields, leaving just the 'title' and 'content' to be populated with the web-crawling process listed below. This process begins by following the link provided to us in the Tweet.

An HTTP get request is sent to the link included in each Tweet. It is common practice among Twitter users to use a URL shortener to post shorter links in Tweets, so the HTTP get request follows all redirects to reach the final news web page. When sending HTTP requests to news sites, there are some other considerations that should be taken into account in order to get the correct web page. For instance, some websites set a cookie for the client and make sure the cookie is sent back by the client before they send the actual web page. Or some other websites have specific configurations to respond differently to bots. To overcome these issues, cookies were accepted and sent back to the server and also some fabricated HTTP headers were sent alongside the original request to each news website. Here is an example header that we used to mimic a chrome browser fingerprint:

```
{'User-Agent': 'Mozilla/5.0 (X11; Linux
    x86_64) AppleWebKit/537.11 (KHTML, like
    Gecko) Chrome/23.0.1271.64 Safari/537.11',
'Accept': 'text/html,application/xhtml+xml,
application/xml;q=0.9,*/*;q=0.8',
'Accept-Charset':
    'ISO-8859-1,utf-8;q=0.7,*;q=0.3',
'Accept-Encoding': 'none',
'Accept-Language': 'en-US,en;q=0.8',
'Connection': 'keep-alive'}
```

As the final step, the web page from the news agency is parsed and the page title is stored in the database; however, extracting the story text is challenging because news sites do not use a consistent markup structure for their web pages. There are general ways to extract large bodies of text from webpages [13], but that requires a considerable amount of initial work and so to simplify the process we customized our script to our small number of news sources. Using this approach we gathered HTML tags containing news story paragraphs, converted the results to pure text, and saved the contents of news stories into the database. Finally, we went through a process to verify that the information in our dataset was in compliance with our defined criteria. The final database is in a server in the SEGAL research lab [3] and the code we used for its creation can be explored on GitHub [4]. However, this database served to be too small for mining, and lead to critical problems in our algorithm training phase such as predicting majority sets. For more concrete results, a larger dataset had to be created.

The next phase of dataset creation began with an exploration of options for increasing our dataset size. First, we attempted to increase our dataset by gathering more than 10 stories per agency per topic, but data was just not available on Twitter. Agencies only tweeted at most around 10 stories per topic, restricting this option for increasing our dataset size. Next, we considered adding more agencies, but that option would require additional news site crawlers, since each script is personalized to the individual sites. At the risk of running out of time, we did not pursue this option. Finally, at the expense of a core feature of our dataset, we decided to eliminate the filter of "topics" from our dataset and collect information only from agencies with high traffic on Twitter, which allowed us to

[2]https://github.com/Jefferson-Henrique/GetOldTweets-python

[3]http://thesegalgroup.org/

[4]https://github.com/aliwebir/cagaben/

download as many stories per agency as necessary. The only limit at this point is the creation of Twitter in 2006.

The new dataset has the same properties as the old one, and spans 8 agencies. We scraped and downloaded roughly 3,000 stories, with a varying number of stories per agency. Refer to table III for details about the number of stories collected per agency.

Name	Twitter ID	Number of Stories
USA Today	usatoday	294
The Washington Post	washingtonpost	676
The New York Times	nytimes	400
Los Angeles Times	latimes	416
Fox News	FoxNews	336
The Washington Times	washtimes	390
CNN	cnn	265
CBS News	CBSNews	182

TABLE III
NUMBER OF STORIES COLLECTED

These numbers are based off of requesting all stories from the last two weeks. This new dataset provided the algorithms much more to work with. In addition to creating the MongoDB collection with the above stories, we also wrote a script to create subsets of that data to feed into the testing models. Each of these subsets was written to CSV files to be fed into RapidMiner [5]. The subsets were created using the cross product of the following feature values:

- Type of X data
  - Boolean existence
  - Real value frequencies
- Y value being predicted
  - ADA scores
  - Alignment (6 agencies for balance across all parties)
  - Alignment (8 agencies)
  - Agency
- Number of stories per agency (up to their own maximums)
  - 100
  - 200
  - 1000 (catch-all, maximum was 676)
- Number of features (up to maximum available given the stories)
  - 2,000
  - 10,000
  - 20,000
  - 100,000 (catch-all, maximum features appears to be 53,000)

[5]https://rapidminer.com/

Given the above feature possibilities, the cross product is 96 subsets. One such example of these subsets is: x_bool__y_ada__s_800__f_2000.csv, which has the values (Boolean, ADA, 100, and 2000). The resulting set has boolean values in the X matrix, ADA scores in the y matrix, 800 stories total (from 100 stories per agency across 8 agencies), and 2000 features (chosen by the tfidfVectorizer from Scikit Learn, most likely ranked by most important based on frequencies). Although we did not test all algorithms (see Section III-B) against all 96 subsets, we did utilize the ability to quickly switch in new datasets using these CSVs which allowed for quick tweaking of our results. In the next section we talk about how this new dataset is utilized after first discussing the analysis of the initial dataset.

B. Phase II: Algorithm Training

Data was split into train and test; 6 and 2 agencies respectively. Then, we proceed to detect the list of words that were important in the news stories with the use of TF-IDF. We defined four ways to collect such list based on the scope of the information input to TF-IDF. Figure 2 illustrates our dataset and the scope of each approach – indicated with capital letters – to collect the important vocabulary.

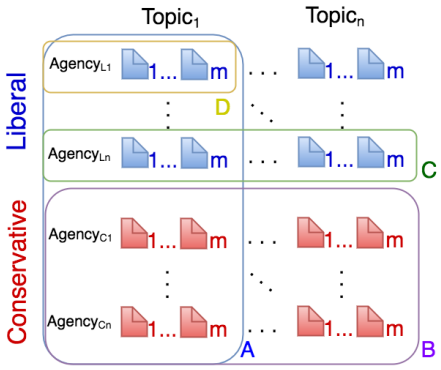

Fig. 2. Dataset Representation

Approach A, considers only one topic and has an extended scope to include all agencies. This approach collects the list of words that are relevant for a particular topic regardless of the agency or political alignment. For example, in news articles about *"gun control"* we expect the word *"gun"* to be highly likely to appear, which makes the word not so important and we expect TF-IDF to ignore it. In the approach B, we integrate news stories from all topics from news sources with similar political alignments. In this case, we expect that the word *"disaster"* so commonly used by conservative parties is not part of the relevant vocabulary. Approach C provides the vocabulary used by a single agency regardless of the topic. Finally, Approach D considers news articles from one agency and one topic.

Our tests were run using approach A with a frequency threshold of 0.2. All algorithms described here use the Python library offered by Scikit Learn. The first algorithm we tried was linear regression to predict ADA scores. Then, we tried

with logistic regression, which required us to change the nature of the predictions from ADA scores to the possible political alignment of the news source, which could be: conservative, liberal or neutral. We did not get any reliable results from either of these approaches. Linear regression was simply predicting the expected value of y (ADA score in our case), disregarding the input features. Logistic regression simply predicted the majority set of the training data, which differed depending on what fold of cross-validation we were on. We suspect these results were due to our small dataset. After trying various parameters for these two algorithms including different logistic regression solvers such as *"liblinear"* (which is supposed to work better for small datasets) we came to the conclusion that our dataset was just too small. So, as mentioned in the "Dataset Creation" section, we created a larger dataset to work with at the expense of dropping the filter of *"topics"*. Next we describe our approach with the new dataset

For the new dataset, we decided to try a data mining tool called RapidMiner. This tool allowed us to see a lot more metrics from each model we built, and also did some of the heavy lifting we had manually coded before, hopefully filtering out any possible errors we had in previous scripts we had written. The first algorithm we tried in RapidMiner was linear regression on ADA scores, with 400 input stories (split equally between agencies), with 500 features (words) which resulted in a correlation score of 0.00. We increased the number of stories and features in different amounts, however, even once the stories increased to 800 and the features were at 10,000 (out of a possible 50,000), the algorithm did not complete the building process after 20 hours, so we cancelled the process. In hopes of finding a faster algorithm, we tried SVM instead. SVM in RapidMiner predicting the agency was our first success, producing around 33% accuracy. This was done using the following parameters: Boolean, Agency, 200, 2000 (See section III-A). Following that success we had our greatest success predicting "alignment", producing around 63% accuracy. This was done with the following parameters: Boolean, Alignment(6), 1000, 2000. In the next section we will decompose these results.

IV. Results and Analysis

Initially, after several attempts at using logistic and linear regression we discovered that our problem was not the algorithms themselves but the size of the subset on which we were training on. When we limit the dataset by topic, by agency and then by political alignment, the subsets are too small such that the algorithms are unable to make any useful predictions. A deep analysis of the results revealed that the resulting small dataset is imposed by its initial design and related constraints. We collect Tweets, which implies a dependency on the information Twitter can provide. Also, we limited our dataset to 5 topics which we assumed were popular enough to build a robust dataset. However, our experience demonstrated that such was a false assumption. Because of our small dataset issue, we created the new dataset(s) discussed in Dataset Creation, which lead to better results and actual predictions.

With the new dataset the two results that stood out the most were both SVM algorithms which we used alongside a polynomial to binomial class converter, run on different datasets. The first result was executed with the RapidMiner configuration: Boolean, Agency, 200, 2000). Details of the approach are illustrated in figure 3. figure

Accuracy: 33.12%

	true usatoday	true washing...	true nytimes	true latimes	true FoxNews	true washtimes	true cnn	true CBSNews	class precision
pred. usatoday	3	2	3	1	0	0	0	0	33.33%
pred. washin...	0	2	1	0	0	0	0	1	50.00%
pred. nytimes	0	0	7	0	0	0	0	1	87.50%
pred. latimes	1	2	0	5	0	0	0	0	62.50%
pred. FoxNews	1	1	0	0	5	0	0	0	71.43%
pred. washti...	0	0	1	1	0	3	0	0	60.00%
pred. cnn	12	11	6	5	20	16	26	18	22.81%
pred. CBSNews	0	0	2	0	1	0	0	2	40.00%
class recall	17.65%	11.11%	35.00%	41.67%	19.23%	15.79%	100%	9.09%	

Fig. 3. Results from SVM polynomial classifier

The overall accuracy is rated as 33.12%, however we can see more useful information in the cross sections and labels of the confusion matrix. For instance, the SVM model was able to predict all the CNN stories as CNN (although over-guessing the class CNN overall). Additionally, the model is making precise predictions for agencies like New York Times, LA Times, Fox News, and Washington times. The over-guessing of CNN class might be due to the fact that CNN has longer stories than other agencies in general. This leads to a higher number of words/features for CNN entries and therefore, the model would become biased towards predicting items as CNN.

The final result we obtained was also achieved using a SVM binomial classifier, run against the (Boolean, Alignment(6), 1000, 2000) subset dataset. This time SVM was given access to all of the stories, and instead of predicting the agency, it was predicting the alignment of the agency. This alignment was a categorical label chosen based on ADA scores. The output is illustrated in figure 4.

Accuracy: 63.60%

	true Neutral	true Liberal	True Conservative	class precision
pred. Neutral	116	25	7	78.38%
pred. Liberal	14	43	0	75.44%
pred. Conservative	83	37	131	52.19%
class recall	54.46%	40.95%	94.93%	

Fig. 4. Results from SVM binomial classifier

From the final results, the overall accuracy is much better, and the attributes of the confusion matrix confirm that average. The algorithm was able to predict almost all of the Conservative agencies, while not over-predicting Conservative overall. Its considerably precise in predicting Liberal and Neutral stories and it has a good F-measure score for all of the three possible class values. Whats more, it never predicts a conservative as a liberal and only mis-predicts 7 of 138

tested conservatives as neutral. On the other hand, it is likely to mislabel liberal and neutral stories. Considering the fact that the agencies like CNN which we considered as neutral based on ADA scores are known as being more liberal than conservative in real world, and also the assumption we had that conservatives are more likely to use some special extreme words. In other words, the model distinguishes liberals from conservatives very well based on the difference in the tone of the languages they use; however, it sometimes predicts liberals as conservatives because liberals also might use an extreme language in some stories around certains topics.

In addition to the results we obtained, the algorithms themselves trained quite quickly (within a few minutes) which was a nice change compared to the unfinished linear regression algorithm we cut off at 20 hours.

V. Conclusions

The exploration of related work allowed us to construct an understanding about how previous studies have used data in media sources, and what techniques have been successful in mining this information. The literature review was very significant for our project, as it gave us a clearer understanding of the models other researchers have used in a similar context and we also acknowledge the limitations each approach represents. Certainly, with the information studied as related work, we got a broad background about techniques to gather information from media sources and they provided us with great ideas we will use to continue the development in our project.

In our preliminary data-collection phase, we have built a dataset of 500 stories. The technique we used automatically indexes news stories based off of a list of news agencies and a list of topics. The algorithm guarantees the number of requested stories as long as the agencies have talked about them enough on Twitter. The algorithm does not favour certain stories over others, it is blind to the process of selection; therefore, the only selection bias introduced is the favouring of recent stories over older ones (a bias we are comfortable with). Furthermore, the real usefulness in the creation of this dataset is not the dataset itself, but rather the ability to run this algorithm at any time with any topics listed to build a new and robust dataset. The only lack of portability is the restriction to news sites we have listed, as the web-scraping algorithm does not generalize to all sites; however, a basic understanding of the code we have written would allow anyone to add new news agencies with a minor amount of work.

After running TF-IDF on our dataset, we realized that it is a powerful tool for finding important terms in news stories. Manual analysis of the results showed a list that closely shadowed a list that we as group members would have manually created given the particular story that was analyzed. This tells us that the process by which TF-IDF selects frequencies is effective; therefore, by extension, the words that are picked as being the "most important" because they surpass some frequency threshold.

While training the algorithms, we discovered the results were biased towards the majority. This issue was due to the fact that Tweets with Tags related to our selected controversial topics and a link to a news agency were not as common as we expected. Surprisingly, even the most popular news sources had less that 15 stories on average. Another problem faced was imposed by Twitter. The platform was developed in 2006, so news created before were not accessible. The sum of limitations, made our dataset flawed because our algorithm did not have enough stories to work with. Most of the data mining algorithms do not perform well with small datasets. To overcome this, we had to re-think our dataset-creation approach.

To simplify – and therefore broaden – our dataset creation, we dropped the concept of topics. Our original hypothesis was centered on the concept of topics and how it would align the word usage of news agencies under the same political alignment, but we needed more data. With topics out of the way, we ran the same dataset creation algorithms, obtaining a few thousand stories that had all been Tweeted within the past two weeks. This new dataset – along with a slightly better partition algorithm – gave us access to better results.

The results we obtained show a trend in word usage across individual news agencies as well as news agencies under the same political alignment. These results are not without their faults, particularly in the ambiguity around predicting the Liberal alignment, but our initial assumptions of the results have direction for how we might improve the results for all political alignments. Our preliminary results of 33% accuracy for agencies and 63% accuracy for political alignment show promise for the use of text mining to predict political bias; however, there is still plenty of work to be done in understand the results of our current algorithms, as well as the modification and creation of new algorithms based on the same premise to build more concrete predictions.

VI. Future Work

The work reported here is part of a more ambitious project: a web-platform that lists news from different sources ranked by their degree of statement bias towards a particular political party. Carrying out this idea will require the expansion of the dataset and the refinement of the learning algorithm. In addition, more research is required to investigate the generalizability of properties of our algorithm to detect bias in sources with a different format from news articles (e.g., blog posts).

The algorithms used have not been thoroughly examined and questioned, a process that will likely lead to improvements. Furthermore, we have yet to explore topics across a wider range of media coverage which may solve the small-dataset problem. One such way in which we might address this issue is to manually code topics for the stories we already have (all 3000 of them), building a set of topics that are more general than the ones this project started with.

The future work of this project will likely be defined by a deeper analysis of our results coupled with manual inspection of the underlying predictions and the stories that were not properly classified. Now that proof-of-concept results have

been established, dedicating additional time to this project is far more tangible.

VII. Team Member Roles

Below we describe the role of each group member.

- Ali Dehgan: Leader, Script Programmer
- Lloyd Montgomery: Script Programmer, Document Reviewer
- Maria Ferman: Researcher, Document edition
- Maryi Arciniegas: Researcher, Document edition

VIII. Estimated Timeline

Table IV presents a description of our work plan and milestones.

Date	Activity
January 27th	Proposal Due
February 8th	Scripts Querying Twitter API
February 15th	Scripts Crawling Websites for Data
February 22nd	Simple Cleaning Algorithms Running on Data
February 23rd	Midterm Report
March 7th	Naive Bayes / SVM Algorithm Running on Data
March 14th	Evaluation of Dataset through Custom Algorithms Labelling Bias (Stretch Goal)
March 21st	Preparing for Presentation and Writing Final Report
March 28th	Final Report Draft
April 1st	Final Report Due

TABLE IV
Estimated Timeline

References

[1] E. Babad and E. Peer. Media bias in interviewers' nonverbal behavior: Potential remedies, attitude similarity and meta-analysis. *Journal of Nonverbal Behavior*, 34(1):57–78, 2010.

[2] C. Burfoot. Using multiple sources of agreement information for sentiment classification of political transcripts. In *Australasian Language Technology Association Workshop*, volume 6, pages 11–18, 2008.

[3] D. Card, A. E. Boydstun, J. H. Gross, P. Resnik, and N. A. Smith. The media frames corpus: Annotations of frames across issues. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, July 2015.

[4] M. D. Conover, B. Gonçalves, J. Ratkiewicz, A. Flammini, and F. Menczer. Predicting the political alignment of twitter users. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, pages 192–199. IEEE, 2011.

[5] D. D'Alessio and M. Allen. Media bias in presidential elections: A meta-analysis. *Journal of communication*, 50(4):133–156, 2000.

[6] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280. ACM, 2007.

[7] J. Golbeck and D. Hansen. Computing political preference among twitter followers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1105–1108. ACM, 2011.

[8] T. Groseclose and J. Milyo. A measure of media bias. *The Quarterly Journal of Economics*, pages 1191–1237, 2005.

[9] M. Kaya and S. Conley. Comparison of sentiment lexicon development techniques for event prediction. *Social Network Analysis and Mining*, 6(1):1–13, 2016.

[10] A. Makazhanov, D. Rafiei, and M. Waqar. Predicting political preference of twitter users. *Social Network Analysis and Mining*, 4(1):1–15, 2014.

[11] J. Martineau and T. Finin. Delta tfidf: An improved feature space for sentiment analysis. *ICWSM*, 9:106, 2009.

[12] B. Masand, G. Linoff, and D. Waltz. Classifying news stories using memory based reasoning. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 59–65. ACM, 1992.

[13] K. R. McKeown, R. Barzilay, D. Evans, V. Hatzivassiloglou, J. L. Klavans, A. Nenkova, C. Sable, B. Schiffman, and S. Sigelman. Tracking and summarizing news on a daily basis with columbia's newsblaster. In *Proceedings of the second international conference on Human Language Technology Research*, pages 280–285. Morgan Kaufmann Publishers Inc., 2002.

[14] J. Ramos. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, 2003.

[15] E. Segev and R. Miesch. A systematic procedure for detecting news biases: The case of israel in european news sites. *International Journal of Communication*, 5:20, 2011.

[16] A. Vavpetič, P. K. Novak, M. Grčar, I. Mozetič, and N. Lavrač. Semantic data mining of financial news articles. In *Discovery Science*, pages 294–307. Springer, 2013.

[17] A. Weichselbraun, S. Gindl, and A. Scharl. Extracting and grounding context-aware sentiment lexicons. *IEEE Intelligent Systems*, 28(2):39–46, 2013.

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