

**IST 736**

**Text Mining Project Report**

**By Emma Gillen, Prasad Duvvuri, & Tiffannie Mac Donald**

# **Introduction**

Bias is often unavoidable, a fact increasingly obvious in news reporting and journalism. According to a survey conducted by the Pew Research Center in Spring 2019, 50% of Americans say made-up news or information is a very big problem in the United States. This ranks higher than issues like violent crime and climate change.

However, it is unclear how to mitigate this problem. According to the same survey, 63% of Americans have stopped getting their news from a particular outlet and 43% have lessened their overall news intake. While this may lower the amount of false information consumed, total avoidance of world issues is undesirable.

A possible solution is to consume media from multiple news sources, with various political affiliations. By reading reporting from different sources, the mutually reported information will likely be closer to fact. However, the task of consuming multiple times the standard amount of media is infeasible for the average person. Instead, the hope is to produce a programmatic procedure to gather news topics, devoid of bias.

This task can be completed in two parts. First, news articles across various sources need to be categorized by their topics. Next, from these articles of the same topic, the main idea will be extracted, giving a high-level overview of what is going on in the world in regards to that subject. By taking the high-level topic, across multiple sources, bias will ideally be largely removed. With bias mitigated, a concise resource for current events and news can be produced.

# **Analysis**

**About the Data**

**Data Collection**

A labeled corpus of news articles was collected as a training set. The articles were gathered from news sources with varying affiliations and perspectives. In addition to coming from various sources, these articles were labeled as pertaining to one of five topics: business, sports, entertainment, politics, and medical.

When collecting news articles from various sources, it was apparent that the data was heavily skewed toward politically-based topics as the United States presidential election was taking place at the time of data collection. In addition, discussion of COVID-19 (medical in nature) occurred throughout all topics. To account for these issues, additional articles were collected from specialized media sources.

The sources used are listed in the table below, along with the topics gathered from each source.

|  |  |  |
| --- | --- | --- |
| **News Source** | **URL** | **Topic** |
| CNN | https://www.cnn.com/us | ALL |
| Yahoo | https://news.yahoo.com/us/ | ALL |
| Fox | https://foxnews.com | ALL |
| ABC | https://abcnews.go.com/ | ALL |
| NBC | https://www.nbcnews.com/ | ALL |
| The BMJ | https://www.bmj.com/ | Medical |
| MedSci | https://www.medsci.org/ | Medical |
| JCI | https://www.jci.org/ | Medical |
| Sports Illustrated | https://www.si.com/ | Sports |
| Bleacher Report | https://bleacherreport.com/ | Sports |
| E! | https://www.eonline.com/ | Entertainment |

**Data Cleaning and Processing**

With the articles collected, their text contents need to be cleaned. URLs are filtered, all words are made to be lowercase, and a series of symbols are removed. The symbols filtered are summarized in the following table.

|  |  |
| --- | --- |
| **Symbols / Regex** | **Meaning** |
| \n | Line feed |
| \r | Carriage return |
| &amp | Ampersand |
| br | Line break |
| . | Periods |
| !#$%^&\*()~!@<>/;:{}[]"? | Various Punctuation |

**Tokenization**

A tokenizer is a tool that assembles a set of rules about how to group characters into individual tokens. For this experiment, each word in each sentence will be treated as a token. Regex is used to tokenize the articles. The regex pattern takes any number of alphanumeric symbols surrounded by white space as a word.

In addition, tokenization can be taken a step further than single words with multi-word token phrases known as n-grams. In this experiment, bigrams (two words) and trigrams (three words) are considered. While these n-grams may not appear as frequently in documents, they may add additional context compared to a bag of words approach.

**Stemming & Lemmatization**

Stemming or lemmatization can be utilized to reduce the dimensionality of a corpus by combining related tokens. Both techniques are explored in this experiment. Stemming is a process where words are reduced to their root forms. This is performed by dropping unnecessary characters from each token, usually a suffix. For this experiment, the Porter stemmer is utilized.

Lemmatization is a similar technique to stemming. Whereas stemming only looks at the form of the word, lemmatization looks at the word meaning. An additional benefit is that lemmatization results in a valid word whereas stemming often leaves an incomplete root of a word. The lemmatizer used in this analysis is the Gensim lemmatizer. By default, the Gensim lemmatizer only takes into consideration nouns, verbs, adjectives, and adverbs and discards the rest.

In the following table, a summary of the tokens making up the corpora are displayed. The counts are displayed both before and after the filtering, tokenization, stemming, and lemmatization steps.

|  |  |  |  |
| --- | --- | --- | --- |
| **Corpus Description** | | | |
| Total Articles | | | 429 |
|  | Medical | | 87 |
|  | Politics | | 85 |
|  | Business | | 88 |
|  | Sports | | 82 |
|  | Entertainment | | 87 |
| Tokens - Raw | | | 621,092 |
| Tokens - Initial Filtering | | | 376,028 |
| Tokens - Porter Stemmer | | | 156,184 |
| Tokens - Gensim Lemmatizer | | | 146,487 |

**Vectorization**

The unique tokens in the corpora are gathered in a list after tokenization to be used as the vocabulary. After the vocabulary is defined, each text article is converted into a vector, where each entry represents an item in the vocabulary. For this experiment, two vectorizers are utilized: binary and Term Frequency-Inverse Document Frequency (tf-idf). The binary representation checks for word presence only, meaning each token in the vocabulary has either occurred (1) or not occurred (0) in each document. The tf-idf representation relates the term frequency (frequency of a term in each document) to the document frequency (number of documents a term occurs in). This gives a measure of the importance of each term to each article in the corpora.

**Summary**

Four versions of each model type will be produced with the four combinations of data preparation discussed above. These include the two vectorizers, binary and tf-idf, and the stemmed and lemmatized corpora. All model types will be trained and tested with these four sets of data and those with the best performance (most accurate classification results) will be discussed in the results section.

**Models**

**Unsupervised Learning**

To get a sense for the natural grouping of the data, unsupervised learning is utilized. Ideally, this will support the labeling performed on the dataset. In this research, k-means clustering is used to split the data into clusters. K-means does this with a defined number (k) of clusters around a series of centroids.

**Supervised Learning**

To predict the topic of each article, train test splits of the labeled corpora are created. The training sets are used to produce a series of models. For this research, Multinomial Naive Bayes (MNB), Bernoulli Naive Bayes (BNB), k-Nearest Neighbors (kNN), Support Vector Machines (SVM) with various kernels, decision trees, and random forests are produced. Each model is run with all four versions of the corpora discussed above.

**Multinomial and Bernoulli Naive Bayes**

Two versions of Naive Bayes models are utilized: Multinomial and Bernoulli. Naive Bayes models are used to classify data based on Bayesian statistics. They are called “naive” because they rely on the assumption that the features are independent. Multinomial Naive Bayes models are produced based on the token frequency. In contrast, Bernoulli uses binary variables as its input.

**K-Nearest Neighbors**

KNN is a lazy learning algorithm, meaning it has no training phase. Instead, it only performs a testing phase. During each query, kNN calculates the distance between the known and unknown values. The majority rule of the k closest labeled points to each unknown value are used as labels.

**Support Vector Machine**

SVMs map input data over a multidimensional space and find a hyperplane which divides the documents into classes with maximum separation. SVM algorithms use a set of functions known as kernels to transform the data from its original form into another which can hopefully be better classified. In this analysis, linear, polynomial and radial basis function (RBF) kernels are utilized.

**Decision Tree and Random Forest**

Decision tree classification builds a model in the form of a tree structure with decision nodes and leaf nodes. At each decision node, the data is split into increasingly smaller subsets and each leaf node represents a classification decision.The random forest classifier creates many decision trees and combines them together to get a more accurate prediction.

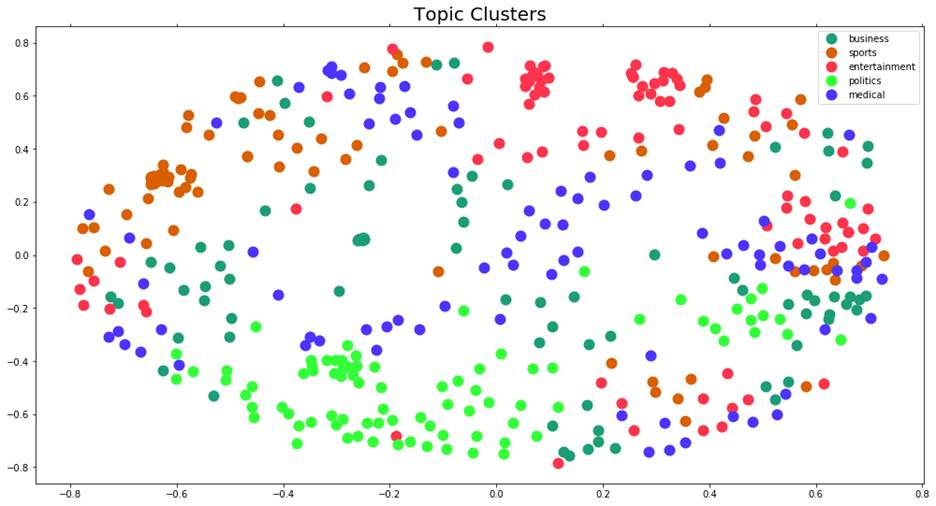
# 

# **Results**

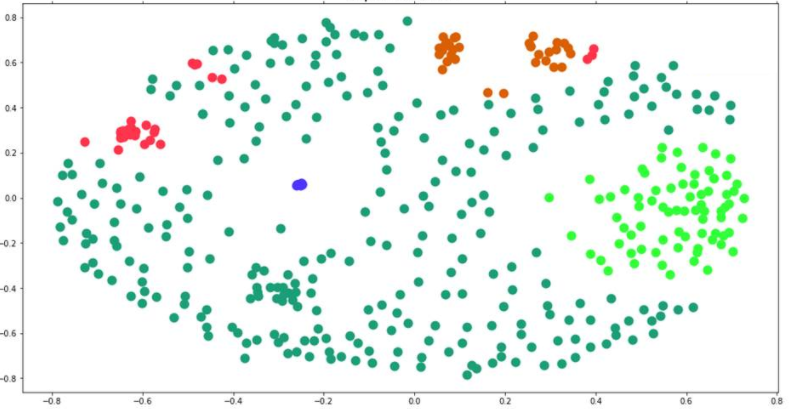
# **Unsupervised Models**

Blind to the known labels, k-means clustering was used to confirm the natural grouping of the articles. In figure 1 below, the manually labeled corpus is visualized. In figure 2, the clusters generated by k-means are visualized. The clustering using k-means appears to be heavily dominated by one cluster. However, the model appears to have correctly identified two closely clustered sections: a group of entertainment articles (at the top of figure 1, in red) and a group of sports articles (to the left of figure 1, in orange). The natural separation between political, business, and medical articles appeared to be less easily identifiable.

**Figure 1:** **Cluster Visualization - Labeled Data**



**Figure 2: Cluster Visualization - K-means Clustering**



To get a better idea of the contents of each manually labeled cluster, a group of word clouds were generated (figures 3-7). In addition to describing the contents of the training corpus, these word clouds give an idea of the type of results that can be generated for future applications. When an accurate model is identified, new articles may be gathered and labeled programmatically. Then, word clouds such as these may be produced using the new, machine-labeled corpora.

|  |  |  |
| --- | --- | --- |
| **Figure 3: Medical Word Cloud** |  | **Figure 4: Entertainment Word Cloud** |
|  |  |  |
| **Figure 5: Politics Word Cloud** |  | **Figure 6: Business Word Cloud** |
|  |  |  |
| **Figure 7: Sports Word Cloud** | | |
|  | | |

Based on the labels predicted using k-means clustering, as well as the known labels, a selection of topics were produced using Gensim topic modeling. Below, those topics are displayed. In the predicted clusters, cluster 0 appears to contain articles about medical data and cluster 2 appears to contain articles about sports. However, the topics of clusters 1, 3, and 4 are less obvious. In contrast, even without knowing the label used to produce each group, the topics of the known label groups are easily identified. In order, these are identified as: business, entertainment, medical, politics, and sports. Given the superior performance of the manually labelled data, the goal going forward is to create a supervised model which can replicate the manual labeling in an automated way.

|  |  |
| --- | --- |
| **Topics Based on Predicted Labels** | |
| 0 | marijuana, vaccine, drug, health, university |
| 1 | stroke, marketing, appreciate, normal, winner |
| 2 | winner, loser, carry, ball, touchdown, role |
| 3 | background, socialism, solid, tony, mask, drug |
| 4 | apple, initial, community, travel, approach |
|  |  |
| **Topics Based on Known Labels** | |
| 0 | trade, recall, account, chapter, thanksgiving |
| 1 | perry, actor, miller, election, film, walker |
| 2 | miller, vaccine, quarantine, priority, wave |
| 3 | congress, lawsuit, constitution, scathing |
| 4 | daily, ballot, golf, match, food, fantasy |

**Supervised Models**

In total, 32 combinations of models and corpora were produced. These were made up of combinations of the eight model types and the four vectorized text combinations. The best results were found when using the Gensim lemmatizer over the Porter stemmer. The average accuracies for the eight models produced for each vectorization method using the Gensim lemmatizer are summarized below (figures 8-9).

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | **Figure 8: Model Accuracies (Gensim Lemmatizer & TFIDF Vectorization)** | |  | | **Figure 9: Model Accuracies (Gensim Lemmatizer & Binary Vectorization)** | |

When using both the tf-idf and binary vectorized versions, the best results were found when using the linear kernel SVM, radial kernel SVM, or kNN. The Multinomial Naive Bayes model followed closely behind in both cases. While the SVM and kNN models had great results, the Multinomial Naive Bayes model is better suited to this task for two reasons: runtime complexity and feature ranking.

The goal of this experiment is to produce a model to generalize future, unlabeled data. Due to the large number of samples, an SVM model takes a long time to train and apply. As a “lazy learning” algorithm, kNN requires no training. However, it has the same runtime complexity in its prediction phase as the Naive Bayes model does in its training phase. In contrast, the runtime complexity of predicting using a Naive Bayes model is substantially faster. In other words, the time taken to train a Naive Bayes model once would be the same as the time it takes to get predictions for a kNN model. However, for each time going forward, the kNN model would need to be completely rerun while the Naive Bayes model could be completed much more quickly. For this reason, the time taken to run the SVM and kNN models many times would be prohibitive.

In addition, the feature ranking provided by Multinomial Naive Bayes is better suited for this task. This means that the model will highlight which tokens are most indicative of each news topic. As the goal of this task is to understand the contents of each topic, feature ranks will be very useful. The best run of the MNB highlighted the following features as the most salient for each topic:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Most Salient Features of the Lemmatized Text** | | | | |
| **Medical** | **Politics** | **Entertainment** | **Business** | **Sports** |
| advantage spread | advance maintain | accord  hope  obtain | administration transition  team | admit  injury  away |
| accounting deserve  attorney | accord  familiar  think | able accommodate year | access  closely  health | adjust  part  problem |
| accidentally accelerate investigation | actually  think | activity  exposure  green | absolutely suggestion  seek | accept resignation  immediately |
| administrator administration | administration republican widespread | activation contraction | advisor  black | accord  plan |
| abuse  judicial | additional determine  here | accord  public | active | accord  time  story |

# 

# 

# **Feature Selection**

# After selecting MNB as the best algorithm for this task, feature selection was explored. A series of MNB models were trained on different subsections of the original data. The top 1000, 2000, 3000, 4000, and 5000 features for each label were used. With this process, the highest accuracy was 93% when using a total of 20,000 features (figure 10). For a second attempt at optimization, chi square feature selection was utilized. With this method, the highest accuracy was 99% when using a p value of 0.4 (figure 11).

|  |  |  |
| --- | --- | --- |
| **Figure 10: Multinomial Naive Bayes: Top Feature Selection** | | |
|  |  |  |
| **Figure 11: Multinomial Naive Bayes: P-value Manipulation** | | |
|  |  |  |

# 

# 

# The Naive Bayes model fit to a 99% accuracy is likely overfitted to our data. However, the hyperfit feature selection is very indicative of the high level summary of each topic. Below, the top features from the improved Multinomial Naive Bayes model are displayed for each topic.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Improved Naive Bayes - Top Features** | | | | |
| **Medical**  **(1082 features)** | **Politics**  **(1876 features)** | **Entertainment**  **(1162 features)** | **Business**  **(4235 features)** | **Sports**  **(1459 features)** |
| vaccine | trump | entertainment | stock | coach |
| health | election | actor | company | season |
| care | president | movie | investment | fantasy |
| study | republican | music | dividend | team |
| testing | senate | perry | oracle | football |
| disease | fraud | check fashion | business | sport |
| medicine | campaign | check fashion entertainment | analyst | fantasy football |
| test | court | division re | homology | game |
| drug | presidential | division re serve | price target | player |
| blood | trump campaign | entertainment gossip | income | fantasy football expert |

# 

# **Conclusion**

Bias is often unavoidable in news reporting and journalism. To mitigate this problem, a programmatic method was produced to gather news topics devoid of bias. Overall, this experiment produced extremely promising results. When considering articles about business, sports, entertainment, politics, and medicine, this method was able to correctly determine a topic in over 90% of cases and was able to generate high level key words surrounding the topics. The table below shows the words that were determined as important to each topic for the articles used in the experiment.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Medical** | **Politics** | **Entertainment** | **Business** | **Sports** |
| vaccine | trump | entertainment | stock | coach |
| health | election | actor | company | season |
| care | president | movie | investment | fantasy |
| study | republican | music | dividend | team |
| testing | senate | perry | oracle | football |
| disease | fraud | check fashion | business | sport |
| medicine | campaign | check fashion entertainment | analyst | fantasy football |
| test | court | division re | homology | game |
| drug | presidential | division re serve | price target | player |
| blood | trump campaign | entertainment gossip | income | fantasy football expert |

While many of these words and phrases are general descriptors, some current events are also highlighted. In medicine, vaccine and testing are highlighted, aligning with the current news about the Coronavirus vaccine. In politics, the presidential election and fraud are highlighted relating to the current accusations of election fraud in the US presidential election. The entertainment topic contains less specific words but the name Perry is highlighted suggesting news surrounding celebrities such as Katy Perry or Matthew Perry. In business, Oracle is highlighted, a company currently making headlines by moving to Texas. In sports, fantasy football is highlighted as the NFL season is currently taking place.

By looking at just these high level keywords, a general view of the current news climate can be gleaned. These results are representative of the type that would be produced in further applications using this method. The goal of this process was to obtain a very high level overview of current events devoid of bais. By looking at only key phrases, there is little room for opinion.

# **Works Cited**

Agrawal, Sanidhya. “What Is Instance-Based and Model-Based Learning? .” *Medium*, Medium, 3 June 2020, medium.com/@sanidhyaagrawal08/what-is-instance-based-and-model-based-learning-s1e10-8e68364ae084.

Amy Mitchell, Jeffrey Gottfried. “Many Americans Say Made-Up News Is a Critical Problem That Needs To Be Fixed.” *Pew Research Center's Journalism Project*, 17 Aug. 2020, www.journalism.org/2019/06/05/many-americans-say-made-up-news-is-a-critical-problem-that-needs-to-be-fixed/.

Ben.k. “Text Mining Preprocess and Naive Bayes Classifier (Python).” *Medium*, Medium, 11 June 2018, medium.com/@baemaek/text-mining-preprocess-and-naive-bayes-classifier-da0000f633b2.

“Computational Complexity of Machine Learning Algorithms.” *The Kernel Trip*, 16 Apr. 2018, www.thekerneltrip.com/machine/learning/computational-complexity-learning-algorithms/.

“Gensim.utils.lemmatize()¶.” *Gensim.utils.lemmatize()*, tedboy.github.io/nlps/generated/generated/gensim.utils.lemmatize.html.

Kanani, Bhavika. “Cosine Similarity - Text Similarity Metric.” *Machine Learning Tutorials*, 30 Apr. 2020, studymachinelearning.com/cosine-similarity-text-similarity-metric/.

Robinson, Julia Silge and David. “Text Mining with R: A Tidy Approach.” *3 Analyzing Word and Document Frequency: Tf-Idf | Text Mining with R*, www.tidytextmining.com/tfidf.html.