**COMP 590-158 Final Project Report**

Jahnavi Alapati - 730393699 - jalapati@unc.edu

Maya Krishnamoorthy - 730391571 - [mkrishn@email.unc.edu](mailto:mkrishn@email.unc.edu)

**Project Title: Partisanship on US News Sites**

***Abstract***

Our study aims to investigate the political biases that are in U.S. news organizations through machine learning models and understand the impact of these biases on the public. Our original dataset contained 21,005 articles from 13 U.S. news organizations alongside their political leanings, rated by judges through Amazon's Mechanical Turk. Our models use feature extraction techniques like TF-IDF and Bag of Words on logistic regression, Naive Bayes, SVM, and an LSTM neural network. The results show similar performance in their evaluation metrics, with LSTM performing the best. These results highlight the limitations of working with textual data and hidden political nuances. In the future, we will continue to explore advanced models to better capture and recognize political language.

***Motivation and Problem Definition***

Partisanship is a dedication or loyalty to a political party, usually accompanied by a negative view of the opposing party. Extreme partisanship is generally regarded as detrimental to the functioning of the government because of its ability to misrepresent political reality. News organizations have extensive power over the public nowadays with their mass production of information. They have the potential to introduce polarizing conversations that affect how the public feels about political issues. The goal of this project is to investigate and identify the pre-existing bias in new organizations concerning political events worldwide. We aim to provide transparency about the publications' political alignment and address the influence news organizations have over public perception. Our goal is to empower the public to analyze their political ideologies more independently.

***Methods***

We intend to train a model to take in a news article and output a label describing the political leaning of the article (Democratic, Republican, or Neutral). We used articles from the [Quantifying News Media Bias through Crowdsourcing and Machine Learning Dataset](https://deepblue.lib.umich.edu/data/concern/data_sets/8w32r569d) published by the University of Michigan to train our model. The variables of interest are the article’s URL and the judge’s vote on how the articles lean politically toward the Democratic, Neutral, or Republican parties. The original dataset consists of 21,005 articles from 13 news organizations. Judges, sourced from Amazon’s Mechanical Turk, qualified the political attitude of each article toward the Democratic and Republican parties as Positive, Somewhat Positive, Neutral, Somewhat Negative, or Negative after reading its content.

Our first step was to map a supervised label for our models using the judge’s scores for the political attitudes of the articles. We began by mapping the qualitative scores of political attitudes to their quantitative counterpart by setting up a mapping equation that respectively assigned the values to [1, 0.5, 0, -0.5, -1]. We took the minimum of both values to create a “negatively\_affected” column that described which party was more negatively affected by the article’s tone. Whichever value was lower would be mapped to either “Democrat”, “Republican”, or “Neutral.” We then took the opposite of those values, where the opposite of “Democrat” was “Republican,” to create the “leaning\_toward” variable, which referenced which party the article leaned towards.

Since the dataset only gave us the URL, we scraped the content of the article to use as our models’ input. We created a “text\_content” column that scraped the article's text from the URL and then normalized the text to remove redundancies, improve the integrity, and allow for the models to predict better. Our normalization techniques included stopword removal, special character removal, number removal, tokenization, lemmatization, and NER tagging. We specifically used NER tagging because of the sensitivity of our articles to organizations, people, and acronyms. Lastly, we also created an organization column that will tell us what publisher released the article.

Post URL-scraping, we found that many articles were blocked by a paywall, specifically from the Wall Street Journal, New York Times, and Chicago Tribune. All articles blocked by a paywall or pop-up were discarded from the dataset for model training and testing. Ultimately, we had 10,776 usable articles for evaluation. Our final dataset consisted of the original 7 variables (URL, news type, perceived, primary topic, secondary topic, Democratic vote, Republican vote), along with the normalized text content, party approval (leaning\_toward), party disapproval (negatively\_affected), and the publishing organization.

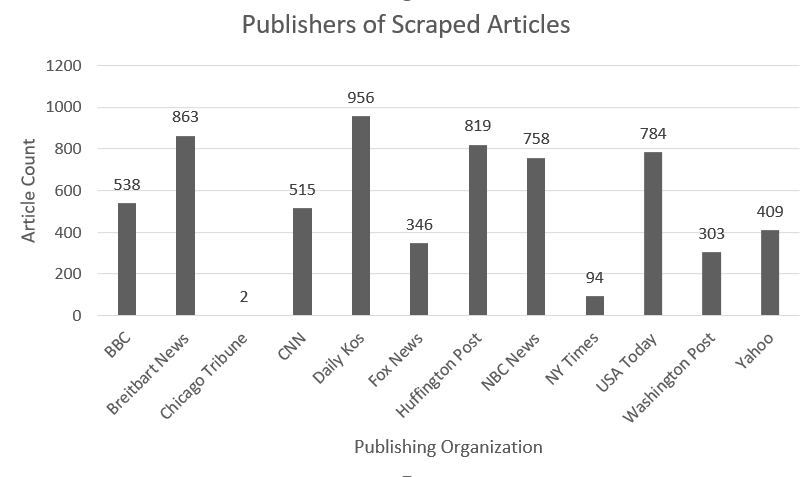
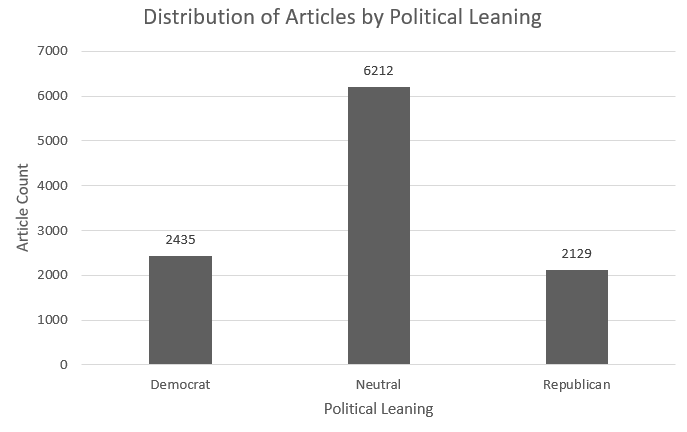
****

Figure 1: Distribution of Scraped Articles by News Organization

Finally, we used TF-IDF and Bag of Words as our main methods of feature extraction for our models. TF-IDF was used because of the ability to weigh the frequency of tokens across articles. BoW was used because of its ability to measure the frequency of word occurrence across articles as a whole.

***Experiments and Results***

To avoid overfitting our models and ensure they are trained properly, we first split our scraped and normalized dataset into an 80%/20% train/test split. We began by training a baseline random classifier with all of the usable articles in our dataset. The baseline classifier gave us an accuracy of approximately 40%, where the number of neutral articles was thrice as much as the articles for either party. As the random classifier was stratified, we saw that the precision of the Neutral articles was higher than the precision of the Democratic and Republican articles, which skewed the accuracy of the model. To prevent our models from training too heavily on neutral articles, we decided to address this by balancing our data and randomly sampling from the Democratic (2,435) and Neutral (6,212) leaning articles so that they were equal to the number of Republican-leaning articles (2,129). After running a random classifier on the resampled dataset, we found an overall accuracy of 32%, which aligned with a random distribution of three labels.

****

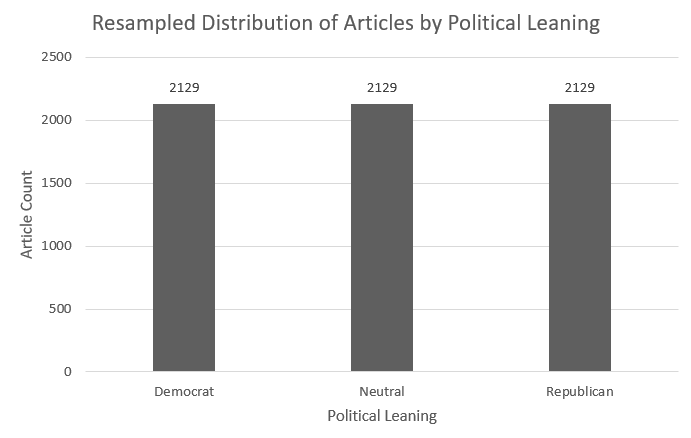


Figure 2: Distribution of Articles by Political Leaning (pre and post-resampling)

| **Model** | **Dataset Split (Train/Test)** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| --- | --- | --- | --- | --- | --- |
| **Random Classifier with all usable articles** | D: 1,926/509  N: 4,973/1,926  R: 1,721/408 | 0.40 | D: 0.21  N: 0.56  R: 0.16 | D: 0.20  N: 0.56  R: 0.17 | D: 0.20  N: 0.56  R: 0.16 |
| **Random Classifier post-resampling** | D: 1,721/408  N: 1,721/408  R: 1,721/408 | 0.32 | D: 0.35  N: 0.30  R: 0.32 | D: 0.35  N: 0.31  R: 0.31 | D: 0.35  N: 0.31  R: 0.32 |

Figure 3: Random Classifier Performance Table

After running the baseline models, we began to train other basic models, such as logistic regression, Naive Bayes, SVMs, and LSTM, on our balanced dataset. We found that each model has its strengths and weaknesses, each working differently to classify articles according to their political leanings. We used both Term Frequency-Inverse Document Frequency (TF-IDF) and Bag of Words (BoW) to extract our features from each of the articles.

Overall, we found that our models performed slightly better with TF-IDF, but our accuracy generally ranged between 55% and 59% for all models. Overall, we found that our models performed slightly better with TF-IDF, but our accuracy generally ranged between 55% and 59% for all models. Models with TF-IDF generally performed better over BoW because it is evaluated not just by how often it appears in a certain document, but also in the entire dataset, allowing it to indicate political leanings better. Logistic regression and SVM performed better than Naive Bayes in every sense. Naive Bayes assumes that features are independent of each other given the class label, whereas logistic regression and SVM do not make this independence assumption and are capable of capturing interactions between features. Moreover, Naive Bayes directly estimates probabilities from the data, rather than the optimization techniques used in Logistic Regression and SVM, which may have led to better performance from the latter.

We can see that the models generally achieved higher precision for Republican-leaning articles, suggesting that when the model predicts an article is Republican-leaning, it is very likely to be correct. This can indicate that the defining features or keywords for Republican-leaning articles are more distinctive or consistently used within these articles. The models also achieved higher recall for Democratic-leaning articles, but lower precision. This combination implies that the model results in a significant number of false positives. We also noticed the general trends for precision, recall, and F1 score are related to how few of the total articles of each category were sampled – there are lower scores for the labels that were less sampled.

| **Model** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| --- | --- | --- | --- | --- |
| **Logistic Regression (w/ TF-IDF)** | 0.59 | D: 0.58  N: 0.56 R: 0.63 | D: 0.59  N: 0.61  R: 0.56 | D: 0.59  N: 0.59  R: 0.59 |
| **Logistic Regression (w/ BoW)** | 0.56 | D: 0.55  N: 0.53  R: 0.60 | D: 0.55  N: 0.56  R: 0.58 | D: 0.55  N: 0.55  R: 0.59 |
| **Naive Bayes (w/ TF-IDF)** | 0.56 | D: 0.50  N: 0.59  R: 0.62 | D: 0.70  N: 0.49  R: 0.48 | D: 0.59  N: 0.53  R: 0.54 |
| **Naive Bayes (w/ BoW)** | 0.55 | D: 0.52  N: 0.56  R: 0.57 | D: 0.62  N: 0.49  R: 0.54 | D: 0.56  N: 0.52  R: 0.56 |
| **SVM (w/ TF-IDF)** | 0.59 | D: 0.58  N: 0.56  R: 0.65 | D: 0.63  N: 0.58  R: 0.57 | D: 0.61  N: 0.57  R: 0.61 |
| **SVM (w/ BoW)** | 0.56 | D: 0.49  N: 0.63  R: 0.63 | D: 0.71  N: 0.41  R: 0.57 | D: 0.58  N: 0.49  R: 0.60 |

Table 2: Logistic Regression, Naive Bayes, and SVM Model Performance

We also compared these basic models to a Long-Short Term Memory (LSTM) neural network. This model performed better than all the other models, with an overall accuracy of 61%. Across epochs, there's a steady decrease in loss and an increase in accuracy for both training (loss from 1.03 to 0.82 and accuracy from 0.52 to 0.63) and validation sets (validation loss from 1.07 to 1.03 and validation accuracy from 0.48 to 0.61), suggesting that the model is learning and improving its performance on the task with each epoch. The relatively close loss values and slight accuracy differences between training and validation suggest that the model is not overfitting and is generalizing reasonably well to unseen data. As LSTMs are recurrent, they are well-suited for text data. They can process documents of varying lengths and retain the phrases or terms that significantly influence an article's political leaning. By capturing these nuances, the LSTM can provide a more accurate political orientation classification than the "basic" models.

| **Model** | **Epoch 1** | **Epoch 2** | **Epoch 3** |
| --- | --- | --- | --- |
| **LSTM RNN** | Training Loss: 1.03  Training Accuracy: 0.52  Test Loss: 1.07  Test Accuracy: 0.48 | Training Loss: 0.94  Training Accuracy: 0.60  Test Loss: 1.04  Test Accuracy: 0.57 | Training Loss: 0.82  Training Accuracy: 0.63  Test Loss: 1.03  Test Accuracy: 0.61 |

Table 3: LSTM RNN

After examining model performance, we also chose to look at how words were classified for each label and how strongly they affected both the Logistic Regression model and the Naive Bayes model that used TF-IDF features. In Logistic Regression, higher positive coefficients suggest words strongly indicative of a class, with "republican," "obama," and "said" being the most significant for Democratic, Republican, and Neutral categories, respectively. In contrast, Naive Bayes assigns log probabilities, with lower values (more negative) indicating a stronger association. We see that words like "nbc" and "huffpost" appear influential for all three classes, perhaps suggesting these terms are common in the dataset but may not be as effective for differentiation. Similarly, Since Logistic Regression assigns coefficients for features in the context of each separate class, the results are more likely to be sensitive to the political tone of each token.

We also explored which words are least likely to appear in each class, and the results showed us that Logistic Regression can better restrict words for different classes. For example, “republican” has a high coefficient for the Democratic class versus a low coefficient for the Republican class, meaning it is far less likely for the word “republican” to appear in the Republican versus Democratic class. This is reflective of the criticism of the Republican party within Democratic-leaning articles, hence the negative coefficient.

| **Model** | **Democratic** | **Neutral** | **Republican** |
| --- | --- | --- | --- |
| **Logistic Regression** | | **Feature** | **Coefficient** | | --- | --- | | republican | 3.65 | | gop | 1.21 | | rape | 1.15 | | administration | -1.16 | | said | -1.18 | | | **Feature** | **Coefficient** | | --- | --- | | said | 1.41 | | wage | 0.97 | | marijuana | 0.95 | | republican | -2.12 | | obama | -2.19 | | | **Feature** | **Coefficient** | | --- | --- | | obama | 2.47 | | administration | 2.43 | | obamas | 1.85 | | court | -0.96 | | republican | -1.54 | |
| **Naive Bayes** | | **Feature** | **Log Prob** | | --- | --- | | nbc | -6.49 | | huffpost | -6.57 | | republican | -6.68 | | outflank | -10.97 | | dispelling | -10.97 | | | **Feature** | **Log Prob** | | --- | --- | | nbc | -6.19 | | huffpost | -6.60 | | said | -6.71 | | roared | -10.97 | | rivet | -10.97 | | | **Feature** | **Log Prob** | | --- | --- | | nbc | -6.39 | | obama | -6.65 | | news | -6.81 | | puzzles | -10.97 | | quadruple | -10.97 | |

Table 3: Strongest/Weakest Words of Interest for Each Class

***Conclusion***

When evaluating the political leaning of news articles using machine learning techniques, it is important to understand the limitations of the data input. Scraping news articles may not be the best way to gather text content, especially since news organizations implement subscription pop-ups and site blocks to limit site scraping. Moreover, human judging of news articles may have also introduced subjectivity and bias into our dataset and models, resulting in some rows of mislabeled data that conflict with the remaining data.

Overall, we found that our models were not incredibly accurate, as a result of the limitations discussed, however, we have a deeper understanding of the subtle political nuances that publishers use in their articles and the complexity of text data. We hope to test more models, such as BERT or RoBERTa, or transformers, which may be able to provide a more nuanced understanding of word relationships and sentence structures than traditional models like Naive Bayes or Logistic Regression.

***Diversity***

Both members of this project, Jahnavi and Maya, started in Computer Science, but have since diverged in their interests. Jahnavi is beginning to transition into the field of finance and looks toward popular U.S. news publications to support her interest and understanding of the sector. Moving forward, Jahnavi hopes to use the knowledge she gained in this course to engage in machine learning projects in her new job and is curious to explore the sentiment of financial reporting by the media.

Maya, a double major in Computer Science and Biostatistics, has been interested in the political effects of healthcare as written in the media. In the fall, Maya is set to start her master’s degree in Health Data Science, where she hopes to explore NLP and machine learning techniques in the healthcare setting. She is especially interested in exploring the political misinterpretations in healthcare reporting. This project represents a culmination of our interests and how news agencies are reporting on the fields of our interest. We are in a unique position to understand different parts of the U.S. news and recognize any political nuances that are in published articles and speak on the implications of public understanding.

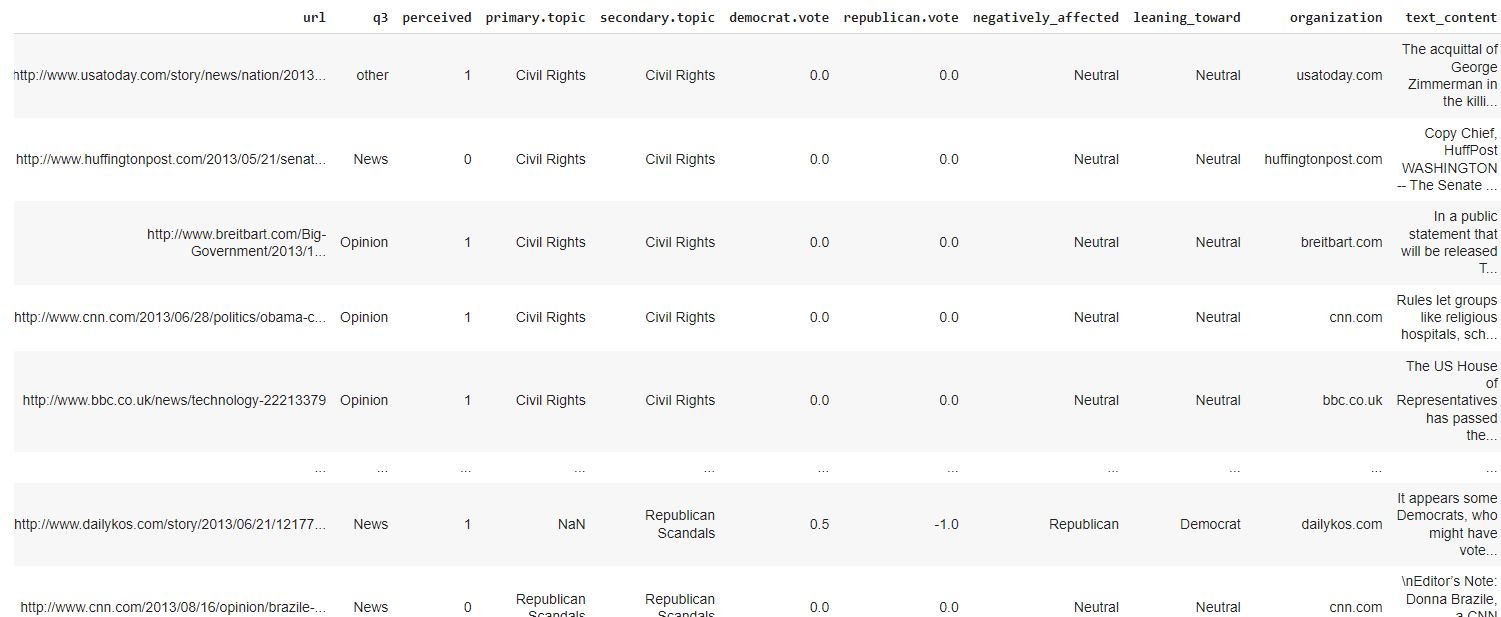
***Individual Member Contributions***

For this research project, Jahnavi and Maya both worked together a few times a week to train, test, and evaluate models. They collaborated on designing the overall methodology and strategy and then decided which models to employ. They worked together to source the dataset, scrape the data, and apply feature extraction techniques to the models. Jahnavi and Maya also each ran the models on their own time, checking in with each other whenever they ran into a problem so they could schedule a time to correct it. They regularly reviewed each other's work, provided feedback, and made adjustments as necessary.

**The Dataset**

The new dataset is made up of:

* the article's URL
* Q3: article type (News, Opinion, None)
* Percieved: blinded/unblinded
* Primary Topic
* Secondary Topic
* Democratic Vote
* Republican Vote
* Negatively Affected Party (Min of Democratic/Republican Vote)
* Party Leaning Toward (Max of Democratic/Republican Vote
* Organization (Published of Article)
* Text Content



In the original dataset, the Democratic and Republican vote had the Mechanical Turk judges rank how each article negatively affected the respective party on a scale of “Negative”, “Somewhat Negative”, “Neutral”, “Somewhat Positive”, and “Positive”.

We first mapped these values to their quantitative counterpart by setting up a mapping equation that respectively assigned the values to [-1, -0.5, 0, 0.5, 1].

Then, we created a “negatively\_affected” column that chose the min of the democrat.vote and republican.vote columns to see which party was more negatively affected by the article. Next, we created a “leaning\_toward” column that was the opposite of the “negatively\_affected” column to show that the article was in favor of the said party.

Next, we created an organization colu,n that will tell us what publisher released the article. Typically, we see that some organizations are leaning towards one party more than the other and we want to take a look into how this plays out.

Finally, we created a “text\_content” column that scraped the article's text from the URL and then normalized the text to remove redundancies, improve the integrity, and allow for the models to better predict.

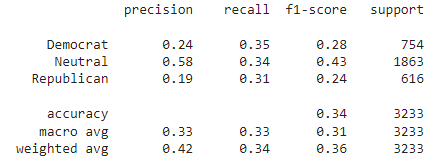
**Random Classifier (NON RESAMPLED DATA)**

The baseline model was generated using scikit-learn's **DummyClassifier** with the strategy set to uniform. A uniform strategy means the classifier will predict the classes uniformly at random. It doesn't learn anything from the training data, instead, it generates predictions by randomly selecting a class - giving each class the same probability.

The random classifier is trained with X\_train and Y\_train, but it doesn't learn anything because it is just a random predictor. It then makes predictions on the X\_test.

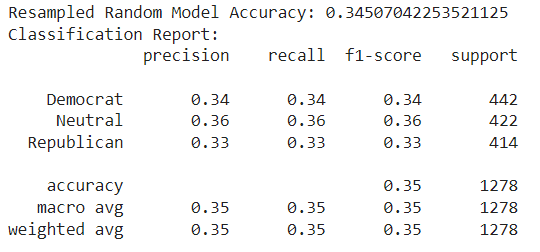
We conducted a Random Classifier before we normalized the text data because the random classifier assigns random labels to the data and does not need the textual information.

From the data, we see that there are 10,776 articles in total - 2435 (22%) Democratically Leaning articles, 6212 (58%) Neutral articles, and 2129 (19.7%) Republican Leaning articles. As the precision tells us the probability that a model predicts a label, it will be right *x* amount of times. The previsions listed in the table are aligned with the distribution of our data.

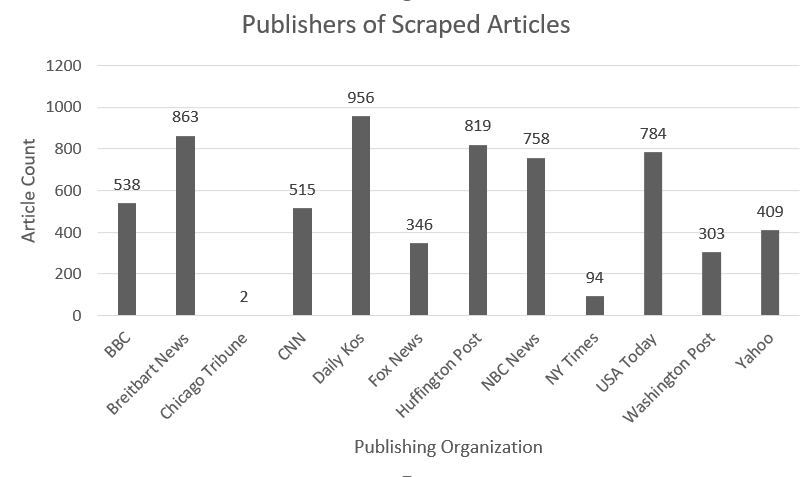


**Random Classifier (RESAMPLED DATA)**

As there is a clear unbalance in the original dataset, we decided to resample the data to get an equal number of every “Leaning Party”. This means that every party had 2129 “leaning” articles. Random Modelling on this new resampled dataset gives us a more even precision (34%) and accuracy 35%.

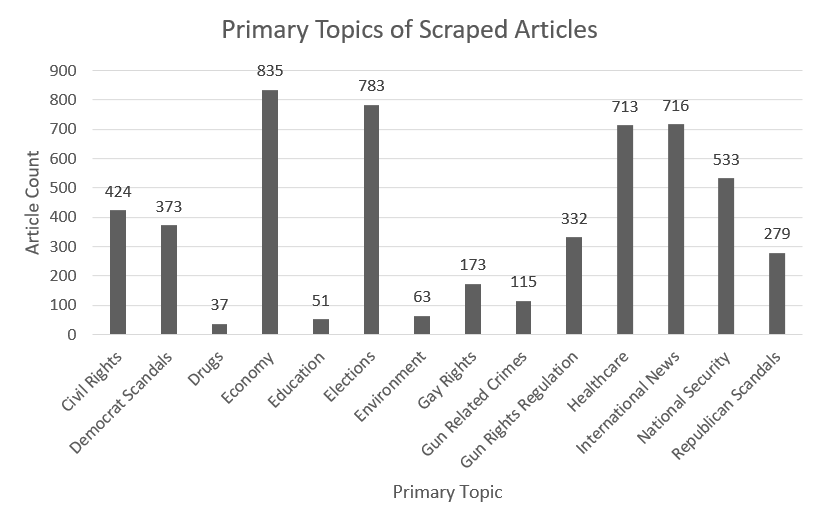
****

**Understanding the Data**

****

This bar graph shows the number of articles scraped from various publishers, offering us some context on the political leaning of different articles within the dataset.

With prior knowledge and research, we can say that there are more conservative publishers (Fox News, Breitbart News), more liberal publishers (Huffington Post, NBC News), but also neutral publishers (BBC, USA Today). If the articles tend to come more from one publisher over the other, there may be an inherent bias in the dataset towards one party's articles. One thing to note is that since WSJ has a paywall, none of their articles were able to be scraped, leading us to miss a lot of the liberally-leaning articles for our training data. While it is relevant to know the potential bias that different media outlets have, it is also essential to analyze the content of the articles themselves.

****

This second bar graph shows the distribution of primary topics in the scraped articles. There is a diverse range of subjects, but most articles are about the Economy, Elections, International News, and Healthcare. Articles within these categories can tell us a lot about party leanings as they tend to be highly politicized and carry distinct partisan narratives. However, there are only a few articles about the environment, Education, and Drugs. These topics might not contribute as strongly to predicting party leanings due to their smaller representation in the dataset.

This information can give us a look into the media's focus or the public's interests. The presence of Democrat-Scandals and Republican-Scandals are tracked separately, showing us that some articles might be inherently biased. Again, the primary topics of the articles don't predict the political leaning themselves, we need to focus on the content and linguistic patterns that are used.

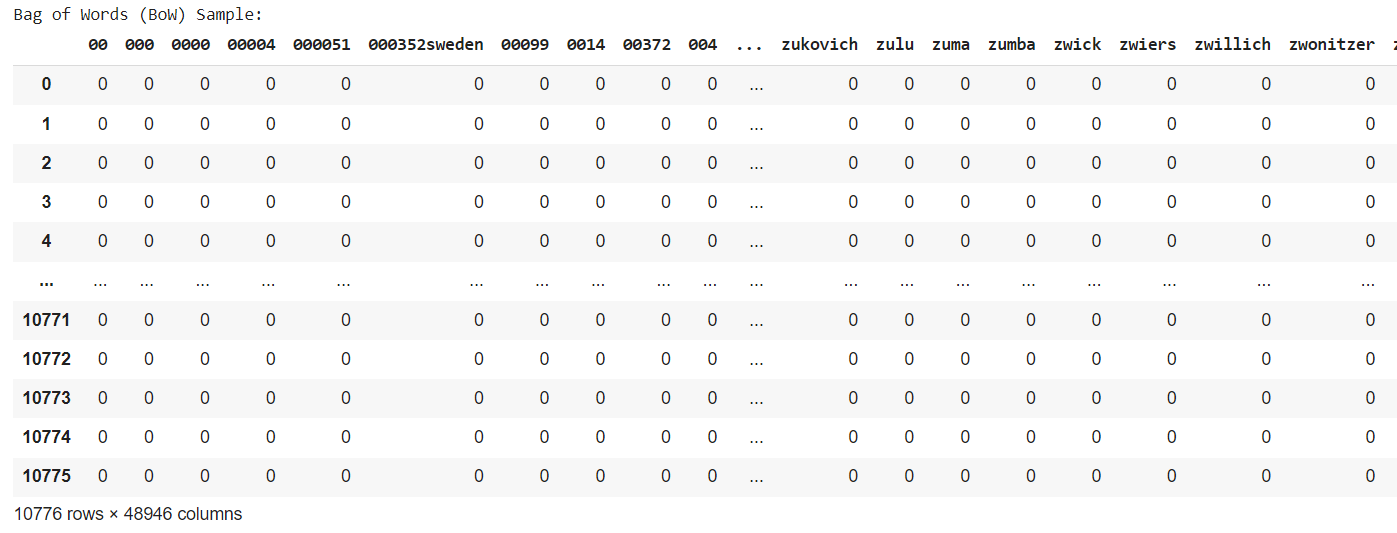
**Text Normalization**

To normalize the text content, we lowercase the text, removed special characters, tokenized, and lemmatized.

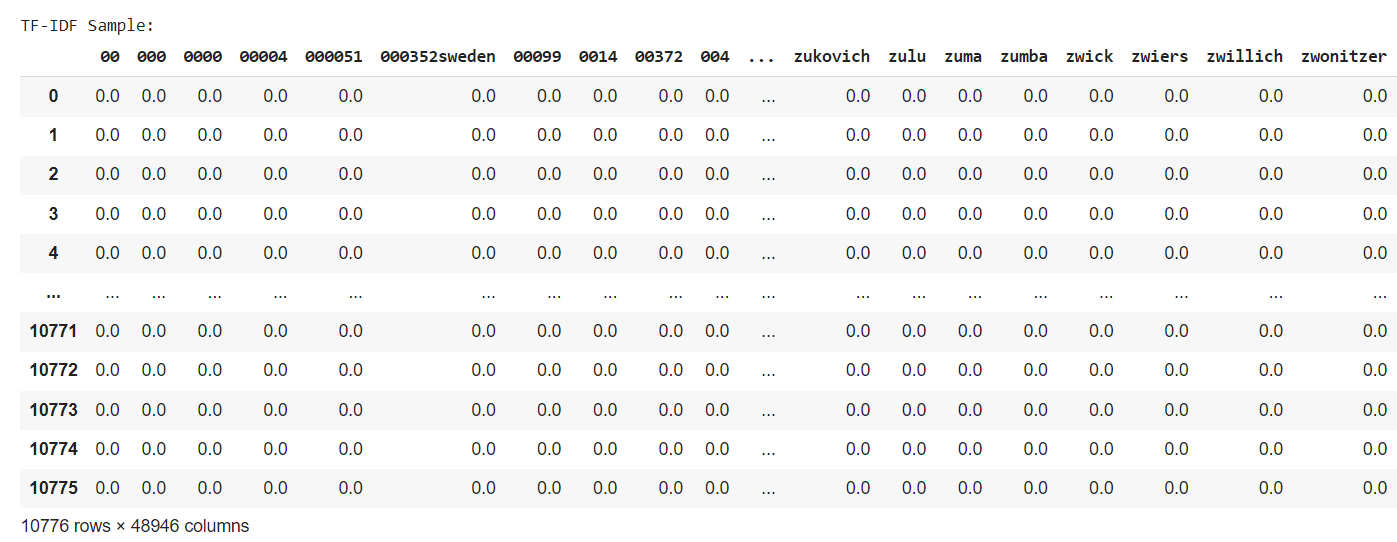
**Feature Extraction**

To extract features, we used Bag of Words, TF-IDF, and Word2Vec.

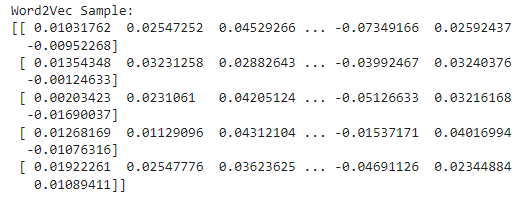
**Bag of Word**

****

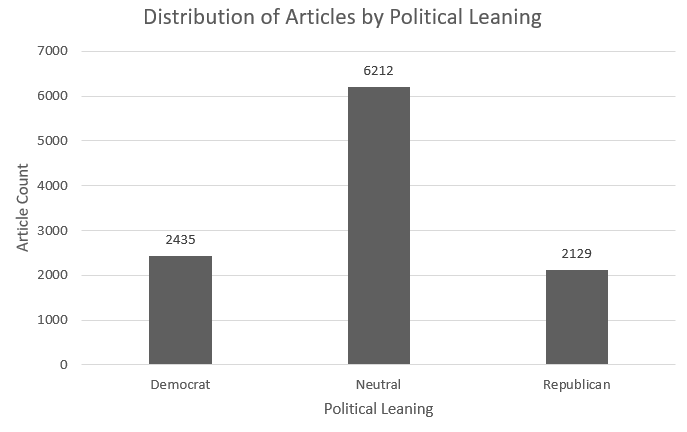
**TF-IDF**

****

**Word2Vec**

****

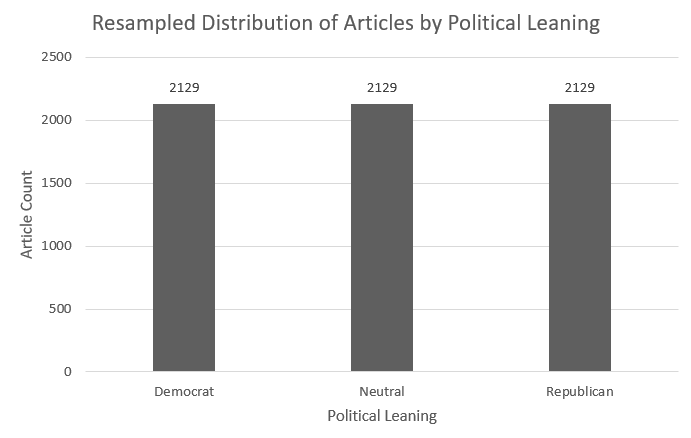
**Rebalancing**

****

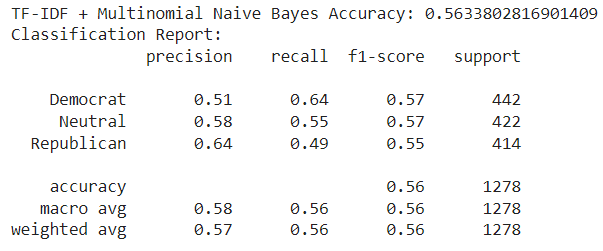
After running our Multinomial Naive Bayes classifier, we noticed that there was a significant unbalance among the precisions on Democratic, Neutral, and Republican Leaning parties. As a result, we first decided to implement Laplace smoothing in our classifier. After that didn't work, we decided to rebalance the dataset.

Rebalancing the dataset helped us address the issue of class imbalance, which is a significant hindrance in classification tasks. When one class outnumbers the others (6212 Neutral articles, 2435 Democratic articles, and 2129 Republican articles), it can lead to poor evaluation performance and model bias due to overfitting of the majority class.

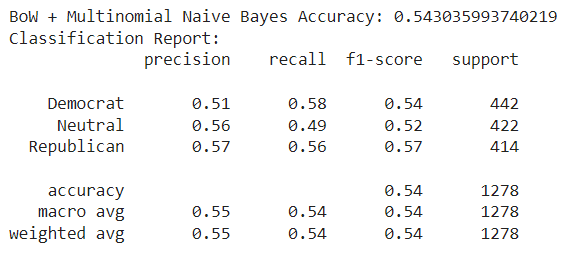
Thus, we rebalanced the dataset by calculating the number of occurrences for each class within the "Leaning Toward" column and found the minimum count of all the classes. This created a balanced dataset by having the same number of samples as the least represented class. Now, the new dataset called resampled\_content, has 2129 articles of each class.



**Multinomial Naive Bayes (TF-IDF)**

****

**Multinomial Naive Bayes (Bag of Words)**

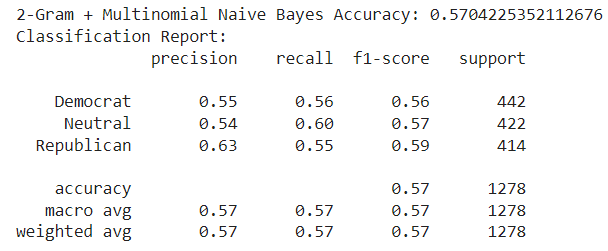
****

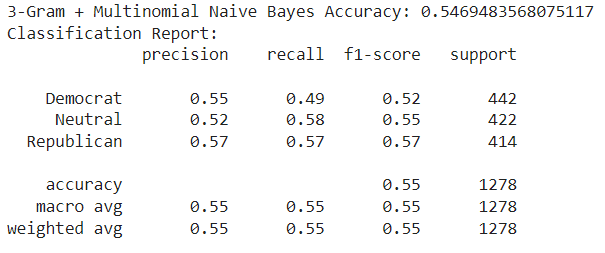
**Multinomial Naive Bayes**

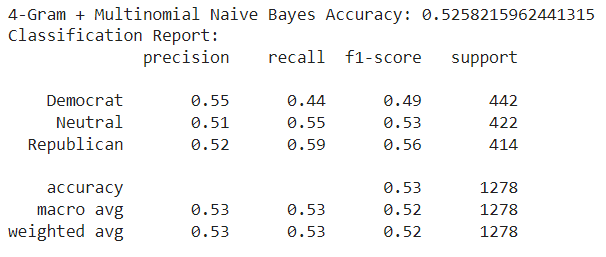
We choose to use Multinomial Naive Bayes because it works well with discrete features that can describe the occurrence counts of the leaning party. We found that it is typically well-suited for problems that involve word counts in text classification and topic classification. Other reasons include the probabilistic nature, efficiency, feature independence, scalability, and interpretability.

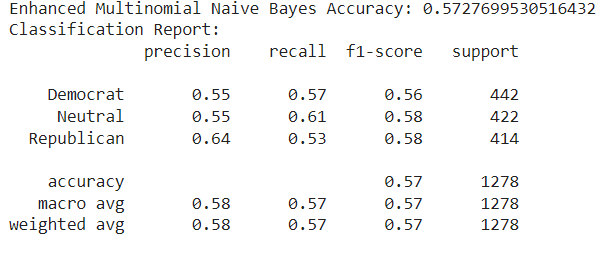
We used Multinomial Naive Bayes with TF-IDF and BoW and although both models have a similar performance rate, we found that the Naive Bayes had a better accuracy with the TF-IDF data.

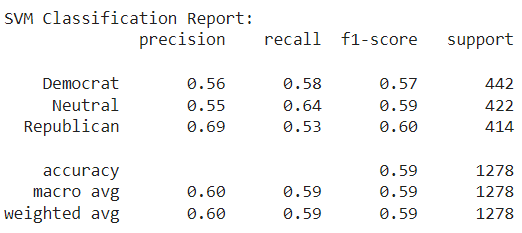
**Multinomial Naive Bayes n-gram**

****

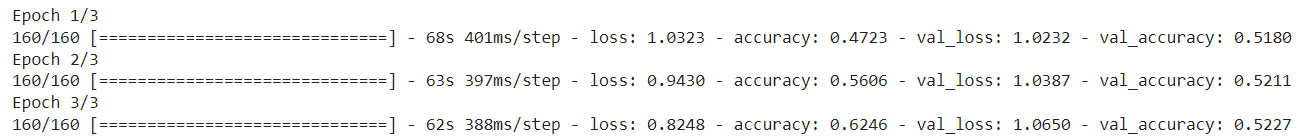
****

****

****

****

**LSTM**

****

### **Detailed Epoch-by-Epoch Analysis**

* Epoch 1:
  + Training loss starts at 1.0323 and accuracy at 47.23%.
  + Validation loss is 1.0232 with an accuracy of 51.80%.
  + This suggests initial learning is happening, but the model is still not very accurate.
* Epoch 2:
  + Training loss decreases to 0.9430, and accuracy improves significantly to 56.06%.
  + However, validation loss increases slightly to 1.0387 with a small increase in accuracy to 52.11%.
  + This could indicate the model is starting to fit the training data better but isn't improving as much on the validation data.
* Epoch 3:
  + Training loss further decreases to 0.8248, and accuracy improves to 62.46%.
  + Validation loss continues to increase to 1.0650, while validation accuracy slightly increases to 52.27%.
  + This may suggest that while the model is getting better at fitting the training data, it might be beginning to overfit, as indicated by the increasing validation loss.

OVERFITTED DATA

**Bert?**