Final Report: The Pulse of Policy - Advanced ML

GitHub Repository: https://github.com/rmccormick96/Pulse-of-Policy

Abstract

Media significantly influences public opinion and policymaking, yet there are few studies that systematically analyze this relationship. In our project, "The Pulse of Policy", we evaluate the influence of media representation on the passage of legislative bills in the House of Representatives. Utilizing a comprehensive dataset compiled from various media outlets such as The New York Times and The Washington Post, spanning from October 2016 to 2017, we: i) Apply a BERT algorithm for topic modeling and sentiment analysis to create predictive word vectors at the bill level; and ii) deploy article-level embeddings to build indicators that measure the similarity between media articles and legislative bills. Using machine learning models, we find that adding these text features significantly enhances prediction performance, suggesting that the contents of media does indeed affect the likelihood of a bill to pass or not.

Authors: John Christenson, Robert McCormick, Pablo Montenegro-Helfer, Santiago Satizábal, Robert Surridge

1. Introduction

In an era where media significantly influences public opinion and policy-making, understanding the relationship between legislative success and media representation is paramount. Our project, "The Pulse of Policy", embarks on an explorative journey to gauge legislative success through the lens of various media outlets. Concretely, we aim to employ advanced machine learning techniques to dissect and analyze the intricate dynamics between media coverage and legislative outcomes.

Our approach centers on a comprehensive dataset meticulously compiled from a range of media outlets, including The New York Times and The Washington Post, and others, spanning from October 2016 to 2017. This timeframe, strategically chosen to cover a significant political cycle, provides a rich canvas to explore the thematic evolution in media discourse and its potential impact on legislative processes. We build a machine learning model to predict the success of legislative bills from the House of Representatives using media information, and other variables related to the finances of the sponsors of bills, cosponsors and party affiliation to each bill.

After the extraction of legislative, financial and text data (from media outlets), we process it to create a table with predictive features and the target variable, which is binary. The processing of the text data follows two main methodologies. The first one is based on identifying keywords from topic modeling techniques that use the BERT algorithm and then applying sentiment analysis to create word vectors on bill level. The second methodology uses the BERT algorithm to create article-level embeddings and compare it to bill-level embeddings, thus making similarity indices. The features created from these methodologies are added to the legislative and financial features to check if predictions improve. Results show that using the text vectors indeed increases the predictive power of our machine learning models.

The rest of the document is organized as follows. Section 2, describes the literature review. Section 3 presents the data extraction procedure. Section 4 describes the data processing. Section 5

presents the results. Section 6 contains the conclusions, Section 7 has the lessons learned from the project and individual work. Finally, section 8 has the bibliography.

2. Literature Review.

Paper 1: "Analyzing Narratives of Patient Experiences: A BERT Topic Modeling Approach":
http://acta.uni-obuda.hu/Osvath_Yang_Kosa_136.pdf

This paper uses BERT topic modeling techniques to identify the most relevant topics in patient opinion in healthcare experiences. Data was extracted from an online forum of patient experience in Hungarian hospitals. A web crawler was used to extract the text of 267.631 opinions. The authors claim that traditional word embedding techniques such as Word2Vec can be problematic in very detailed data, so they used a BERT representation to learn contextual features between words, and fine-tuned it to show 15 words per topic cluster. The final goal of this paper was to use sentiment analysis in each user experience, which we also use in our project. The topic modeling section is relevant in our case to extract the important topics from media sources. It is important to mention that the authors did not use quantitative metrics to evaluate both the topic modeling and sentiment analysis models, which are unsupervised. They instead used manual interpretation. With this analysis, they are able to quickly identify the quality of health services according to user experience, which policy makers can use to identify problems in hospitals more efficiently. The most important part of this paper for our project is the topic modeling methodology using BERT, and its fine-tuning methodologies, as well as the sentiment analysis. We use this procedure to find topics of media sources.

Paper 2: "Online Twitter Bot Detection: A Comparison Study of Vectorization and Classification Methods on Balanced and Imbalanced Data":
https://engrxiv.org/preprint/view/3139

This paper aims to classify if tweets are either human or a bot and compare results with different word embedding methodologies. They develop a complete pipeline of, preprocessing (tokenization, stop-word and punctuation removal, stemming), vectorization (Bag-of-words (BoW), TF-IDF, Doc2Vec, and BERT), feature extraction (fastText), prediction (Support Vector Machine, Logistic Regression, and Naive Bayes), and evaluation (F1-Score). They gathered the data from a research group that had already gathered the data, stored it and labeled it. Their results show that the neural network embedding methods have higher F1-Scores. Although the final question of this paper differs from our project, the pipeline created by this paper and, most importantly, the vectorization techniques can be both implemented in our case for the legislative bills and media sources.

Paper 3: "Testing machine learning algorithms on a binary classification phenological model":
https://onlinelibrary.wiley.com/doi/full/10.1111/geb.13612

This paper utilizes 18 machine learning classification models to predict the first flowering date of temperate trees within a specific time window. The observational data was obtained over a 56 year period, with some missing years in between. The authors created samples for phenology prediction by assigning binary labels based on whether a phenological event occurred before or after a given date. They use a

dynamic time window to select relevant meteorological data, specifically daily mean temperature, which is crucial for temperate plant phenology. The authors set the random seeds to ensure comparability and repeatability and then split the data into training (50% of the original data), validation (20% of the original data), and test (the remaining 30%). They then used five fold stratified cross-validation and evaluated the fivefold average scores across accuracy, area under the curve (AUC), recall, precision, F1 score (balanced F Score), kappa (kappa coefficient) and the Matthews correlation coefficient (MCC). With the validation dataset the authors chose the best models to hyperparameter tune and chose accuracy as the main evaluation metric to evaluate the model. With model ensembling (bagging, boosting, blending and stacking), the authors combined different classifiers into a meta-classifier, to provide better generalization. While the subject matter differs from our topic, the process to evaluate the data matches our binary classification models.

Paper 4: "Using Artificial Intelligence to Predict Legislative Votes in the United States Congress": https://ieeexplore.ieee.org/document/9403106

This article uses machine learning models to predict the likelihood of bills passing in the House of Representatives and in the Senate of the United States. The authors extracted data mainly from ProPublica, related to information from the sponsor of the bill and the text of the bill, from the 113th Congress to the 115th Congress. Given the unbalanced nature of the target variable, they used artificial balancing techniques to balance the data and improve training and prediction results. After processing the data and feature engineering, they used binary classification machine learning models such as L2-regularized Logistic Regression, Support Vector Machine (SVM), Decision Tree, Neural Networks, and K-Nearest-Neighbors Algorithm. This paper was useful to use the ProPublica API to extract relevant information and define which variables we should use on our project. It also has a binary classification problem such as ours and we use many of the machine learning models they used, as well as the final score metrics.

3. Data Extraction

3.1. Data Extraction - Media Outlets

One of our tasks, within the process of data collection, was to scrape articles from several media outlets. The idea behind this is to use this data as a representation of topics prevalent in the public's eye and use this information as a predictor of legislation approval. We first approached this by narrowing down which media outlet would be representative of the public as a whole. We referenced a resource (https://adfontesmedia.com/interactive-media-bias-chart/) that analyzes media political bias, overall reporting accuracy, and the popularity of the media site. We believed that collecting outlets from far left to far right would capture the most prevalent subjects in society across the political spectrum. We chose outlets ranging from The Washington Post, CNN, New York Times, Fox News, and Breitbart.

To start the data collection/scraping process, we used an API website called Apify to conduct the scraping. Although we tried to scrape the front page of each news source, this strategy did not work because the front page did not have a way to crawl to the previous day's front page. Initially, we attempted to scrape trending pages, but noticed a bias towards pop culture articles. To address this, we opted for the opinion subdomain, mitigating the need to crawl through past articles and ensuring a more diverse range of topics. For websites lacking sitemaps such as Fox and Breitbart, we had to scrape the opinion page.

This led to an aggregation of articles across domains, causing data skewness towards recent years. Specifically, for Fox, we collected over 48,000 articles, with 19,000 in 2023 and 15,000 in 2022, while the desired data range for 2017-2018 had only 791 and 267 articles, respectively.

The project ultimately involved scraping all articles from The Washington Post, New York Times, and CNN between October 1, 2016, and December 31, 2017. This timeframe was chosen as it represented a non-election year, during which media coverage predominantly centered on political candidates rather than issues or topics relevant to bills being voted on. Additionally, we were provided with a database that includes a right-leaning spectrum of news, which we were lacking from 2016 to the present.

3.2. Data Extraction - Bill information: ProPublica API and congress.gov selenium crawler.

We collected bill metadata information on introduced legislation from the following Congresses using the ProPublica API: 115th (7394 bills, 1187 house resolutions, 149 house concurrent resolutions, and 146 house joint resolutions); 116th (8200 bills, 1189 house resolutions, 117 house concurrent resolutions, and 107 house joint resolutions); 117th (9704 bills, 1532 house resolutions, 125 house concurrent resolutions, and 106 house joint resolutions); and 118th (7352 bills, 1013 house resolutions, 90 house concurrent resolutions, and 114 house joint resolutions). We retrieved about 5749 raw texts from introduced legislation by web crawling congress.gov with Selenium. Table 1 shows an example of the information including in this dataset.

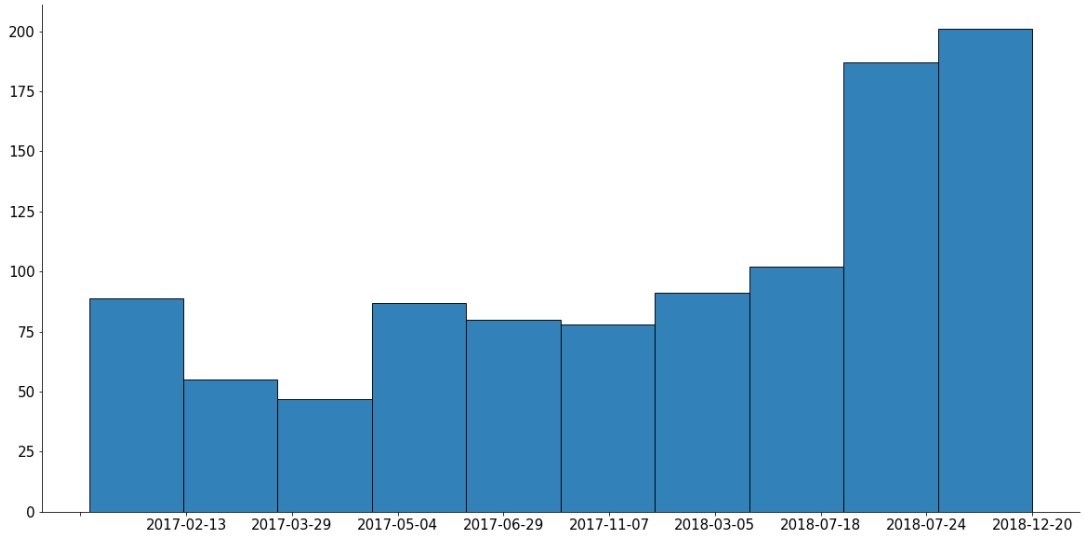
Table 1. ProPublica Sample metadata

Column	Description
bill_id	‘hr7401-115’
number	‘H.R.7401’
bill_type	‘hr’
title	‘To modify provisions of law relating to refugee resettlement, and for other purposes.’
short_title	‘Strengthening Refugee Resettlement Act’
sponsor_title	‘Rep.’
sponsor_id	‘E000288’
sponsor_name	‘Keith Ellison’
sponsor_state	‘MN’
sponsor_party	‘D’
introduced_date	‘1/2/2019’
active	FALSE
latest_major_action_date	‘1/2/2019’
latest_major_action	‘Referred to the Subcommittee on Trade.’
congressdotgov_url	‘https://www.congress.gov/bill/115th-congress/house-bill/7401’
house_passage	None
senate_passage	None

enacted	None
vetoed	None
cosponsors	0
cosponsors_by_party	0
primary_subject	‘Immigration’
summary	‘’

We used this metadata to identify if a bill passed or not, provided information for future feature creation, and merged the dataset to donation information from the Federal Election Commission. Additionally, we chose to utilize both bills and house joint resolutions as both provide actionable changes to law while house resolutions and house concurrent resolutions are more symbolic in nature. The number of passed actionable legislation in the House of the 115 Congress of the year 2017, that we use on the prediction model, represent 14.49% of the total of introduced bills and house joint resolutions, which are 642 of a total of 4.430. From the whole of the 115the legislation, 13.49% bills got approved in the House of Representatives, which are 1,017 of a total of 7,540. The histogram below shows the distribution of passed bills from 2017 to end 2018. It shows that there is sufficient data across the timeline to generate a prediction model.

Figure 1. Histogram of passed bills in the House, 115th Congress.



Notably, both the number of passed actionable legislation and percentage of proposed legislation has decreased with each Congress as Table 2 below shows.

Table 2. Congress Actionable Legislation Statistics

	115th Congress	116th Congress	117th Congress	118th Congress
Passed Bills & House	1017	754	549	245

Joint Resolutions (Count)				
Proposed Bills & House Joint Resolutions Passed (Percent)	13.49%	9.08%	5.60%	3.28%

It is worth noting that we finally only used legislative bills and financial information from 2017 because of computational constraints, and media outlet data from November 2016 to December 2017.

3.3. Data Extraction - Federal Election Commission.

The Federal Election Commission has financial information of candidates to Congress, which includes total money received, disbursements, loans, debts, individual contributions, among others. The table below shows some examples of this information.

Table 3. Federal Election Commission - Candidates

Name	SHEIN, DIMITRI	YOUNG, DONALD E
affiliation	DEM	REP
end coverage date	12/31/2018	12/31/2018
total receipts	209,916.04	1,234,680.31
total disbursements	209,574.16	1,387,687.05
candidate loans	101,440.14	-
debts owed	367.52	-
individual contribution	58,188.98	670,374.33

This data was concatenated to the bill dataframe by the name of the candidate. This way we are able to know the financial information of the sponsors of each bill and use it as predictors of the approval of bills. The concatenation was done by comparing each candidate's name on the financial dataframe to each candidate name on the ProPublica dataframe, by using Python's Fuzzywuzzy text similarity score and only keeping scores above 0.8.

4. Data Processing.

4.1. Structured Data From ProPublica and Federal Election Commission.

The processing and feature creation of this section followed a traditional approach of creating numerical columns, when necessary, relevant for a machine learning model. The variables created from the ProPublica data are:

- Party affiliation of the sponsor
- Number of cosponsors (Republican and Democrat)
- Cosponsor ratio index

On the other hand, the variables used in the model from the Federal Election Commission are related to the financial information of the sponsor. Many were dropped because they had too little variance (too many zero values), had no variance, or had too many missing values. The ones finally used are:

- Total receipts
- Total disbursements
- Total individual contributions
- Contributions from other political committees
- Refunds to individuals

4.2. Data Processing of Media Outlets

The main challenge of this part of the project was to go from the 175,797 news articles to similarity indicators in the form of scalars that could measure the correlation between each bill and the corpus of news that were published around the date the bill was introduced. Based on different NLP methods we learned in class; the processing of the news articles dataset was broadly partaken in two steps. First, a topic modeling that allowed us to reduce the dimensionality of the original data and, at the same time classify or cluster the articles into the different topics. Second, the computation of the indicators through three approaches: i) TF-IDF bag-of-words, ii) TF-IDF cosine similarity, and iii) text embeddings with cosine similarity. While the first two approaches use the topics (from the topic modeling) as input, the latter uses a smaller version of the five months datasets disaggregated by media outlets. In this section, we will thoroughly explain each of these steps and methods in detail

4.2.1. Topic Modeling

As a first preprocessing step for the topic modeling, we thought that it made no sense to use the whole dataset of news for each Bill to predict its likelihood to pass or not. Instead, we thought of a date range or window in which the content of the news could exert more pressure over the House of representatives to pass or not a bill. Considering that, for bills introduced in a specific month, we sliced the dataset to include only news published from two months prior to two months after the month of introduction (e.g. for bills introduced in January, 2017, the news dataset would include only articles published from October, 2016, to March 2017). This "sliding-window" slicing resulted in ten datasets of around 60,000 articles we preprocessed removing stopwords (*nltk*) and punctuation.

4.2.1.1. Choosing the Topic model.

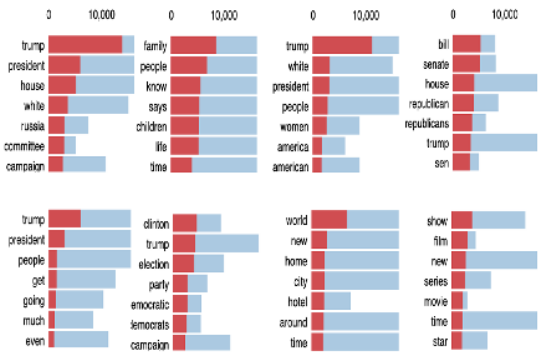
One of the strongest limitations of our approach was the size of the data and the limited time and (unpaid?) computational power we had available. This limitation didn't allow us to test different topic models let alone fine tune them with different hyperparameters. Under those circumstances, we chose to use BERTopic as one of the state-of-the-art models that is widely utilized and versatile as well. Even if we were not able to test its results exhaustively against other models or to hypertune it, we did make a small test, using the circa 16 thousand articles of CNN as a subsample of the news data. More specifically, we compared the results of BERTopic with the results of a widely used traditional probabilistic topic model: Latent Dirichlet Allocation (LDA). The coherence of the resulting topics of LDA (0.5194) was significantly lower than that of the resulting topics using BERTopic, which suggests that – even if the "c_v" coherence might be rudimentary - the words that composed the topics of BERTopic were not only more related into them but also more semantic meaningful as a whole and more interpretable.

Building upon this, Figure 2, shows the top eight topics of each model and, through visual inspection, it becomes apparent why there is such a gap between the coherence of the two models. For instance, for LDA, topic two (family, people, know, says, children, life, time) and topic seven (world, new, home, city, hotel, around, time) seem generic (the words are not as related between them) and do not convey a very concrete idea of what could be the content of the articles. On the contrary, for BERTopic, for instance, topic five (Isis, Mosul, Iraqi, forces, Aleppo, Iraq, al) immediately conveys the conflict in the Middle East and more specifically in Syria or Iraq; topic seven (JKorea, north, korean, kim, missile, pyongyang), in turn, points towards the missile crisis with North Korea. The difference in coherence is most likely because, while LDA uses the traditional BoW as the way to do the embeddings of the documents, considering the co-occurrence of words but disregarding the order and the context, BERTopic uses more advanced NLP techniques such as transformers or SBERT that consider these aspects.

Furthermore, for the representation of the topics, the BERTopic default architecture uses cTF-IDF that not only considers the importance of a specific word within the topic but also how strange or rare that word is across other topics. LDA doesn't have this feature and that is probably the reason why within its top 8 topics, almost five include the words "Trump" and "president" in their own top words.

Figure 2. BERTopic vs LDA: Top 8 topics

LDA



BERTopic

Using BERTopic with its default hyperparameters as our model, we found the topics for each of the ten sliced datasets and reduced the dimensionality of the data from around 60,000 articles per month of bill introduction, to around 760 topics (vectors of ten words). Furthermore, the process allowed us to classify or cluster the articles in each topic. This was important considering that we ran a sentiment analysis per article (more information on this the next section) to classify whether the article was positive (1) or negative (0) and so the classification, allowed us to compute an overall indicator of sentiment per topic by aggregating the proportion of positive-sentiment articles in each topic.

As a final step to get the outcome from this step of the data processing, we sliced the top fifty topics and merged it with the Bills dataset via month, meaning that the months introduced in the same month had the same representation (vector) of topics to calculate its similarity indicators.

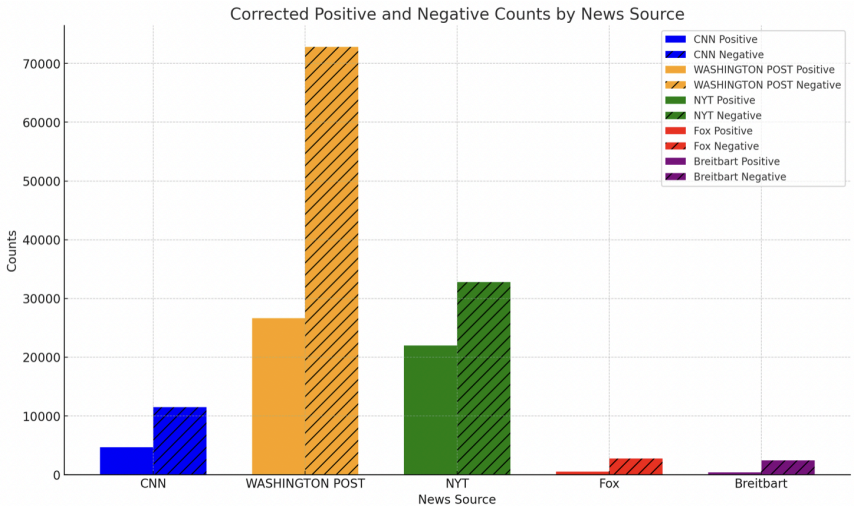
4.2.2. Sentiment Analysis with Distilbert-base-uncased-finetuned-sst-2-english

We incorporated a simple sentiment analysis into the feature creation process. We aimed to capture the sentiment (positive or negative) of related articles and use this to gauge the sentiment for each topic in Topic Modeling. We believed understanding the sentiment of prevalent topics in right- or left-leaning media outlets could introduce a new dimension to the analysis and help us discern whether the sentiment of a topic correlates with the party in power in the respective branches of government. However, due to computational constraints in the topic modeling, we decided to conduct it for the whole news corpus over a more granular level of topics in right or left-leaning media. Nonetheless, understanding public sentiment as a whole added an important dimension to our claim.

We then evaluated three potential pretrained models for this approach: Twitter-roberta-base-sentiment, Distilbert-base-uncased-emotion, and the default Distilbert-base-uncased-finetuned-sst-2-english. After weighing various factors, we decided to use Distilbert-base-uncased-finetuned-sst-2-english. The major factors for this decision were the efficiency and accuracy of the model and its generalizability. DistilBERT is a BERT-based model that is forty percent smaller in terms of parameters, which allows the model to be 60 percent faster while maintaining over 95 percent accuracy compared to the original. We believed this aspect would be perfect for the large amount of data we were evaluating (175,000 articles). We also chose this model for its generalizability. DistilBERT is trained on general text and fine tuned on the Stanford Sentiment Treebank, which is the premier dataset for sentiment analysis. This gives the model the ability to accurately describe sentiment across many domains, making the model suitable for understanding the nuanced language across different categories of media. We considered Twitter-roberta-base-sentiment for its accuracy but decided against it due to computational cost. In addition, we looked at Distilbert-base-uncased-emotion, but the model did not seem applicable for our approach since its purpose is to find emotion within text.

We conducted the sentiment classification across all news articles and applied a function that splits text articles over 512 tokens into chunks and finds the overall sentiment across a whole article. We classified articles as zero for negative and one for positive. The results showed a one-to-two ratio of positive to negative classification. Through evaluating the results, the model seemed to classify neutral text as negative but still gave us a meaningful representation of sentiment. We then averaged the sentiment per topic and used this average as the value representing the overall sentiment of a given topic.

Figure 3. Distilbert Sentiment for all Media



4.2.3 Data Processing - TF-IDF with Bag of Words

In our data analysis, we set the minimum similarity threshold at 0.0 and the maximum at 0.0138, resulting in 4992 bills showing some level of similarity. This meticulous approach was guided by the implementation of a computational method that stands out for its efficiency and affordability, completing similarity mapping and calculation in less than a minute compared to alternative methods like TF-IDF Cosine Similarity (20 minutes) and Text Embeddings Cosine Similarity (3-5 hours).

The generated data encapsulates the dynamic landscape of legislative bills, offering a glimpse into their thematic interconnectedness. Each bill, represented by a set of top 10 keywords, underwent a similarity score calculation based on its alignment with 50 monthly topics derived from aggregated lists over the past five months. In this analysis, it's crucial to highlight that out of the extensive dataset, a total of 4,992 bills exhibited some degree of similarity based on the established criteria, underscoring the broad spectrum of legislative content captured in our approach.

However, the strengths of our approach are accompanied by a noteworthy weakness. The calculated similarity scores tend to be relatively low, with a capped maximum of 0.014. This limitation poses a challenge when integrating this parameter into the final classification model, as its impact may be considered relatively weak. The process involved meticulous steps, including summarizing monthly topics, identifying top 10 keywords for each bill, and calculating similarity scores. This was achieved by mapping bills to topic lists, multiplying keywords by topic weights based on media frequency, and then dividing by the fixed number of keywords.

4.2.4 Data Processing - TF-IDF with GloVe Embeddings

In this approach, we decided to understand the subject of the bill with TF-IDF word frequency. We chose this method because we are working on a single bill text rather than a corpus of text. We then used TF-IDF to calculate the word frequencies and extracted the top 10 words, placing them in a word vector to match the number of words in a vector from the Media topics. Using TF-IDF for a single document (the bill text) helps to distill the text down to its most significant words which would represent the subject of the bill.

Next, we converted both the bill keywords and the ten words for one topic into GloVe embeddings using GloVe 6B. The idea behind converting the keywords from the bill and the selected

topic words into GloVe embeddings allows for a meaningful comparison of semantic content between the legislative text and various topics. From there, we took one word embedding from the bill keywords vector and calculated the cosine similarity for every word embedding in the topic. We then took the max similarity and repeated this process for all the embeddings in the bill keywords vector. With the ten top similarity numbers, we then averaged them to get the final number to represent the similarity of that one bill with that specific topic. Taking the maximum similarity score across embeddings, ensures that the strongest semantic relationships are identified, providing a focused measure of how closely the bill's content aligns with each topic. We performed this process for every bill and every topic.

From the topic modeling, we had over 750 topics; however, we decided due to time, to limit the analysis to the top 50 topics. Then, for each bill, we calculated the similarity score for each of the top 50 topics and had an associated sentiment score. From here, we wanted to convert this large dataset into two others for evaluation so we would have a total of three for evaluation. The first was the original with 100 columns of the top 50 similarity scores and 50 sentiment scores, the next had the top 20 similarity scores and 20 sentiment scores, and the last dataset we created was with PCA analysis on the data from 100 columns. We saw that there was a high correlation between the features in the large dataset, so we extracted the top 32 principal components, which accounted for 95 percent of the variance in the dataset. Lastly, we repeated this process to create three datasets but applied a weight to each word from the News Topics that was created in the topic modeling process. We did this in hopes of capturing the importance of each word from all topics.

This approach showed promise as shown in the increase in F1 score described in the results section. It has promise in its computational efficiency creating similarity scores for over 4900 bills and 50 topics within 18 min on CPU.

4.3 Data Processing - Text Embeddings

For our third approach, we created text embeddings for every bill and news article to compare later on via cosine similarity. To do this we cleaned the data, especially the legislative text, removing formatting and lingo unique to legislative texts that add little informative value or context and otherwise may create noise. We then similarly correctly formatted all texts, bills, and news articles. Stopwords are kept as they provide contextual information for BERT.

Using BERT-base-uncased, the clean text is tokenized and segmented out into segments of tokens with a max length of 510 tokens. If the tokenized text of a bill or news article is over 510 tokens that is when the text is segmented out into blocks of 510 tokens (the total may be less in the final segment). If the final or only segment is less than 510 tokens, then padding is added as required. A [CLS] token ID is added to the beginning, and a [SEP] token ID is added at the end of each segment. An attention mask is then applied to any padded token.

These segments are then input into the BERT model and the text embeddings are then pulled from the last hidden state. We chose to pool our data with mean pooling the 512 tokens together and then either mean or max pooling segments together if needed. This embedded tensor is then stacked with the rest of the created tensors representing each bill or article from the input source and returned as a large Pytorch tensor. These processes were run separately for all potential textual inputs: legislation, Fox News, Breitbart, New York Times, and The Washington Post.

We then calculate the cosine similarity between each legislative bill and every news article by news source within a 5 month window. We did not perform normalization due to it making the data unusable: all results were approximately 0.99.

Our features were created from these cosine similarity scores on both an approach, mean-mean pooling or mean-max pooling, and news source basis. These features were the mean, median, max, min, standard deviation, and percentiles (99th, 95th, 90th, 75th, and 25th) for the individual piece of legislation by the news source (Fox News, Breibart, New York Times, or The Washington Post) within the 5 month window. A small sample is shown below in Table 4.

Table 4. Example of Features Created by Bill

index	number	title	max_fox_meanmean	90_percentiles_fox_meanmean	median_fox_meanmean	25_percentiles_fox_meanmean	min_fox_meanmean
0	H.R.4198	To promote the economic security and safety of survivors of domestic violence, dating violence, sexual assault, or stalking, and for other purposes.	0.874643087387085	0.8025921285152435	0.748144656419754	0.6975264549255371	0.07491542597254558
1	H.R.4194	To direct the Mayor of the District of Columbia to establish a District of Columbia National Guard Educational Assistance Program to encourage the enlistment and retention of persons in the District of Columbia National Guard by providing financial assistance to enable members of the National Guard of the District of Columbia to attend undergraduate, vocational, or technical courses.	0.8698894381523132	0.8034332394599915	0.7563745677471161	0.7184199243783951	0.06240624349134781
2	H.J.RES.120	Proposing an amendment to the Constitution of the United States limiting the pardon power of the President.	0.8751451373100281	0.8077190816402435	0.7720199525356293	0.7451876550912857	0.052755491859421105
3	H.R.4181	To amend the Higher Education Act of 1965 regarding proprietary institutions of higher education in order to protect students and taxpayers.	0.8624138832092285	0.7884517908096313	0.74082812666893	0.7012868523597717	0.06455981300209063
4	H.R.4186	To amend title 18, United States Code, to protect more victims of domestic violence by preventing their abusers from possessing or receiving firearms, and for other purposes.	0.860110878944397	0.7966400384902954	0.7604035437107086	0.7413358688354492	0.035608050557337574

Of note, we also planned to utilize the CLS token from the first segment. However, this was dropped due to the cosine similarity scores results being 1 or -1 for every result and that calculating them continuously crashed the RAM of collab running at A100 GPU. CNN was also dropped due to unforeseen issues with calculating the cosine similarity scores with CNN articles.

5. Results.

The prediction model consists in a binary classification model where the outcome variable determines whether a bill of the House of Representatives is voted to pass or not. This target is highly unbalanced, as only 14.49% of the bills in our sample got approved by the House. Because of this, we used a random oversampling method to balance the training data. We used the following machine learning models for training: Logistic Regression, CatBoost Classifier, Support Vector Machine, Random Forest Classifier, and XGBoost. Each model was trained on the benchmark, which uses legislative and financial information only, and also on each dataset that adds the vectorized text features to the benchmark. The methodologies to create these features are the following:

- Similarity with BOW (from topic modeling and sentiment analysis)
- Similarity with Glove Embeddings Unweighted (from topic modeling and sentiment analysis)
- Similarity with Glove Embeddings Weighted (from topic modeling and sentiment analysis)
- Mean Max Text Embeddings with BERT
- Mean Squared Text Embeddings with BERT.

The table below shows the F1 Score results on the benchmark (legislative and financial information) and by adding text vectors on the benchmark, which were calculated using the

methodologies presented above. We used 5 machine learning models, which are identified on the first column.

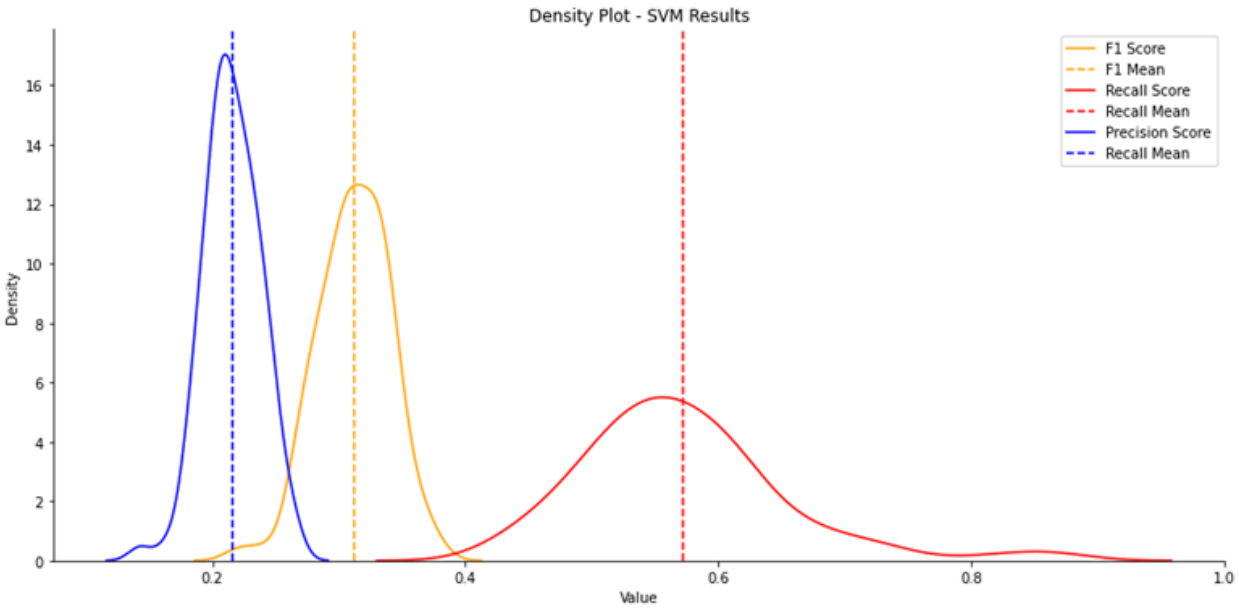
Table 5. F1 Scores of Machine Learning Models

Model	Benchmark	Similarity with BOW	Similarity with Glove Embeddings Unweighted	Similarity with Glove Embeddings Weighted	Mean Max Text Embeddings with BERT	Mean Squared Text Embeddings with BERT
Logistic Regression	0.282	0.282	0.281	0.281	0.283	0.283
CatBoost Classifier	0.336	0.341	0.349	0.330	0.358	0.364
Support Vector Machine	0.338	0.317	0.384	0.369	0.417	0.456
Random Forest Classifier	0.335	0.334	0.342	0.348	0.381	0.419
XGBoost	0.330	0.335	0.328	0.339	0.381	0.359

As seen on the table, adding the text vector columns improves the F1 score the most when using the embeddings from the Mean Squared Text Embedding with BERT methodology and training with a Support Vector Machine prediction model.

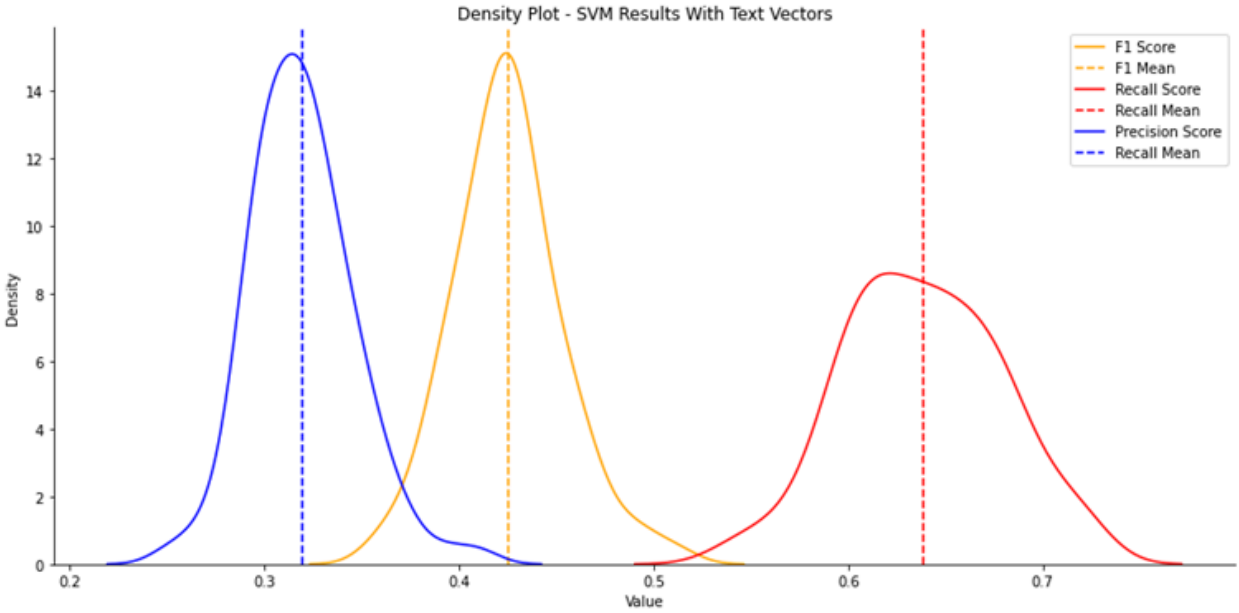
To check that we were not lucky because of the specific train-test split used above, we used 100 different random splits and subsequent oversampling of the training data to calculate scores. The graph below shows the density plots and the mean of the F1, recall and precision scores of the benchmark, using the SVM model. The means for each score are also shown on the graphs, and are: 0.311 (F1), 0.571 (Recall), and 0.215 (Precision).

Figure 4. Density Plots and Means - Benchmark



The graph below shows that all the average scores improve when adding the vector columns calculated with the Mean Squared Text Embedding with BERT methodology. The new average scores are: 0.425 (F1), 0.638 (Recall), and 0.319 (Precision).

Figure 5. Density Plots and Means - Mean Squared Text Embedding with BERT



The implications of these results are that the media outlet information is relevant to predict if legislative bills pass or not.

6. Conclusions.

The results of the binary machine learning classification models show that text from media outlets can help predict the approval of a bill from the House of Representatives. This was verified by higher F1, Recall and Precision scores when using text related features, compared to the benchmark.

The prediction model could be improved in many ways. First, more media articles could be scraped and integrated into the training model. Additional legislative information could also be searched and processed to help improve prediction. This includes including non-political topics from a wide range of sources beyond New York Times, CNN and The Washington Post, and balancing media portfolio by curating an equal mix of left and right-leaning news sources.

It is worth noting that the topic modeling and BERT vectorization processes were computationally demanding and took a long time to run. This limits the number of different timeframes that can be used to train prediction models and experimentation. In addition, it would be worth exploring how increasing the number of topics in the method of TF-IDF with GloVe Embeddings would affect the model. As mentioned before the method was computationally efficient and we can afford expanding the topics to potentially capture more similarity in bills.

7. Lessons Learned From the Project and Individual Work

Generally speaking, we learned new methodologies to process text using state-of-the-art models such as BERT. We had never done topic modeling and embedding creation using this algorithm. We also learned how to merge datasets based on a key of strings, which required a similarity metric. Additionally, we extracted information from official sources that we had not used previously, as well as media sources.

As a group, we already had experience programming in Python in areas such as structured data processing, a more basic level of text processing, plotting, and using binary classification models.

Nevertheless, we all learned from one another new skills as we all aimed to solve problems from which we had no previous experience in solving.

For individual effort, Rob McCormick worked with data collection and preprocessing of right wing media sources, creation and implementation of sentiment analysis, and creation of the TF-IDF with GloVe Embeddings approach and application.

Bobby Surridge created an approach for media selection, data collection and preprocessing of all left-leaning media outlets, applying sentiment analysis to left-leaning media outlets, creation of the TF-IDF with Bag of Words approach and application, and creation of README files. He, along with John Christenson, used the ProPublica API to extract legislative info for each bill and then scrape Congress.gov for the raw legislative text.

Santiago Satizábal, performed several preprocessing steps with the news outlets dataframe, from removing stop words and punctuation, to slicing it into 10 5-months smaller dataframes. Furthermore, he performed the BERTopic model in each of those 10 datasets and aggregated by topic, the sentiment score of the articles. Finally, he run an example comparison between LDA and BERTopic that brought compelling insights about the convenience of the latter. Together with John Christenson he performed feature creation for text embeddings with the already calculated cosine similarity scores, turning them into means, medians, standard deviation, and percentiles.

Pablo Montenegro Helfer read 3 papers of the literature review and described them in this section. He also merged the ProPublica and Federal Commission information and created the financial and legislative features, prepared the features for the binary prediction models, ran the models and their results, and ran an initial version of the topic modeling code.

John Christenson read one paper (paper 3) and wrote its literature review for. He, along with Bobby, used the ProPublica API to extract legislative info for each bill and then scrape Congress.gov for the raw legislative text. Individually, he performed the text data cleaning, text embedding creation, and cosine calculations for the text embeddings. Together with Santiago Satizábal he performed feature creation for the text embeddings with the already calculated cosine similarity scores, turning them into means, medians, standard deviation, and percentiles.

8. Bibliography

Osváth, M.; Yang, Z., G.; Kósa, K. (2023) *Analyzing Narratives of Patient Experiences: A BERT Topic Modeling Approach.* Acta Polytechnica Hungarica, Vol. 20, No. 7, 2023.

https://acta.uni-obuda.hu/Osvath_Yang_Kosa_136.pdf

Bari, A.; Brower, W.; Davidson, C. (2021). *Using Artificial Intelligence to Predict Legislative Votes in the United States Congress*. 2021 IEEE the 6th International Conference on Big Data Analytics.

https://ieeexplore.ieee.org/document/9403106

Dai, W.; Jin, H.; Zhou, L.; Liu, T.; Zhang, Y.; Zhou, Z.; Fu, Y., H.; Jin, G. (2022). *Testing machine learning algorithms on a binary classification phenological model*. Wiley Online Library - Global Ecology and Biogeography.

https://onlinelibrary.wiley.com/doi/full/10.1111/geb.13612

Chen, Y. and Ling, J. (2023). *Online Twitter Bot Detection: A Comparison Study of Vectorization and Classification Methods on Balanced and Imbalanced Data.* Engineering EngrXiv Archive.

https://engrxiv.org/preprint/view/3139