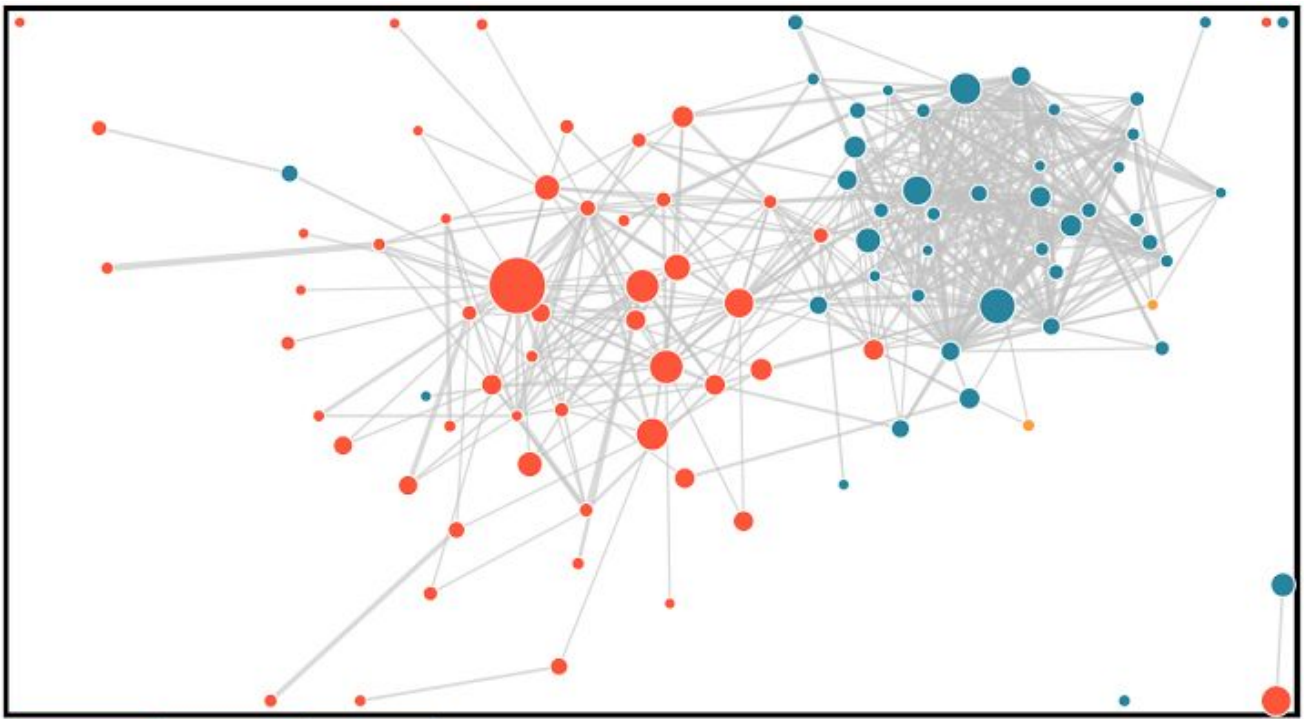
Predicting Bill Passage in the 109th Senate: An Integrated Network- and Text- Analysis Approach to Classification

Gregory Adams
CAPP 30255: Advanced Machine Learning for Public Policy
Final Project

Codebase: github.com/gregorytadams/AdvancedMachineLearning/project
Data: https://uchicago.box.com/s/j9ncizfbtn53ipg6to2inwlcjiukzmb0



Summary

This project attempted to use the text and sponsors of bill introduced in the 109th Senate to predict whether or not a future bill would pass. My initial goal was to "use Natural Language Processing methods to predict the likelihood of different bills (or different types of bills) passing; Network Analysis, including the PageRank algorithm, to measure influence of different members of Congress; and more traditional statistical and machine-learning techniques to combine those approaches in an integrated classifier" (from initial project proposal). I successfully reached that goal, building a machine-learning classifier can very reliably predict the passage or failure of a bill based on the text of the bill and congressional sponsorship networks.

Literature Review

There is a large body of academic research that considers network analysis as a way of understanding the inter - and intra- house dynamics in the United States Congress. Although not all seek to predict bill passage--some simply try to understand what the network looks like for the sake of greater understanding, it seems--the methods are nonetheless relevant to my efforts. Additionally, others have used Natural Language Processing methods to understand legislation, much as I have.

Mason A. Porter et al. from the Georgia Institute of Technology was one such team that sought to use netowrk analysis to understand the U.S. Congress.[1] By analyzing data from roll call votes (votes in which every member's vote is counted individually) in the House of Representatives, they constructed a network to analyze the structure of the body. They found varying levels of organization that largely mirrored the formal structures of organization, i.e. subcommittees, committees, parties, etc. Overall, they were able to demonstrate how some of the somewhat organic methods of organization mirrored the inorganic ones.

[1] Porter, Mason A. et al. "A Network Analysis of Committees in the U.S. House of Representatives." *Proceedings of the National Academy of Sciences of the United States of America* 102.20 (2005): 7057–7062. *PMC*. Web. 18 Mar. 2017.

In a paper much more closely related to my work, Stanford Researchers Janice Lan, Mengki Li and Suril Shah used network analysis to predict the votes of indicidual members of Congress by integrating vote data with data about the congressional networks.[2] By leveraging the nbetwork, these researchers were able to ue data about how other congressmen had voted to predict the vote of a particular congressman of interest. Though it doesn't use text data, it nonetheless demonstrates one method of integrating disparate data types and using that integration as a basis for prediction.

With regard to Natural Language Processing, while there is relatively little about prediction, there is much research that seeks to utilize NLP methods to better understand legislation, especially in the context of information extraction. One such paper, *Modeling Legislation Using Natural Language Processing,* used NLP methods to try to transform abstract regulations into formal language, allowing individuals to more easily extract actionable information from it.[3] Though I do not deal with information extraction, it nonetheless serves as a useful example of how others have tried to leverage similar methods to my own.

Analysis

There were four steps to this project:

1. Data Gathering and Cleaning -- Scraping from Congress.gov
2. Text-Based Classification -- Using the text of the bills to predict passage
3. Network-Based Classification -- Using the congressional sponsorship network to predict passage
4. Integrated Classification -- Combining the outputs of steps 2 and 3 to predict passage

[2] Lan, Janice, Mengke Li, and Suril Shah. *Utilizing Network Analysis to Model Congressional Voting Behavior*. N.p.: n.p., n.d. Print.

[3] Rouzbahan Rashidi-Tabrizi, Gunter Mussbacher, Daniel Amyot, "Transforming regulations into performance models in the context of reasoning for outcome-based compliance", *Requirements Engineering and Law (RELAW) 2013 Sixth International Workshop on*, pp. 34-43, 2013.

Data Gathering and Cleaning

I needed two types of data for my classifier and got them both by scraping the Congress.gov website. For each bill, I got the name, text, and sponsorship information (i.e. who sponsored the bill and who cosponsored the bill). I used the number of each bill as the key with which I organized the information throughout my analyses.

To gather this data, I used python's Selenium library because the urllib library seemed to be blocked (it returned a 403 error). The scraper took in a search--in this case, all bills introduced in the 109th Senate--and progressed through the website, visiting each page and gathering/downloading the information that I needed.

In terms of cleaning, I tried to maintain the integrity of the data as much as possible: I only removed clear nonsense, digits and punctuation. One issue I ran into was the structure of the bills; they are written almost as an outline, so it does not have the linear structure that most text does. Rather than attempt to reconstruct the text, I decided to treat it as any other piece of text data in order to preserve the natural order of the text. One area of extension, however, may be to reconstruct the bills as if they were written in normal sentences for text classification.

Text-Based Classification

Text classification primarily consisted of trying different feature generation methods, models, and model parameters to try to find the most predictive classifier. The models I tried included linear classifiers (SGD/SVM, Logistic Regression, etc.) as well as Naive Bayes. Features included count vs. TFIDF vectorization, n-gram ranges, and

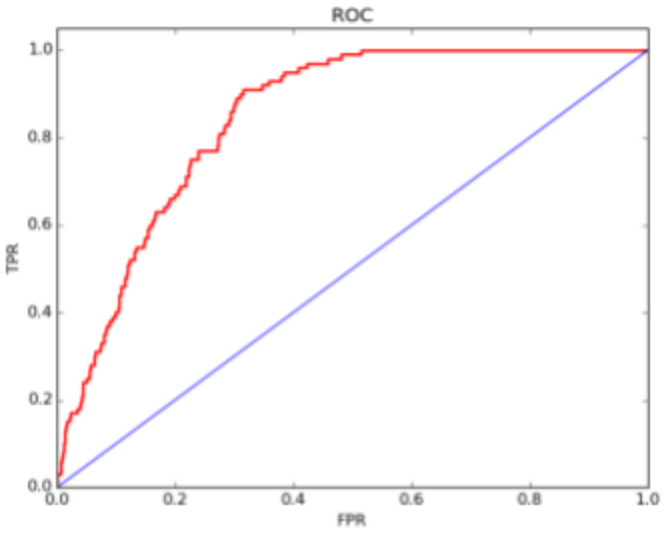


Figure 1: ROC plot for model 536

normalization, while model parameters varied for each model.

For model selection, I built a pipeline that would perform a grid search on parameters for feature generation and model/parameter combinations. I evaluated using standard k-fold cross validation where k=3 (I chose a low K for the sake of runtime; I use a higher K at later validation steps). The result of this pipeline was a results report from the gridsearch module that ranked the classifiers by test scores.

Ultimately, the Naive Bayes models seemed to do the best, though the linear classifiers also did very well. Test scores for top models reached 97%, and the ROC curve for one of those models (number 536 in the output file) is shown in Figure 1 (AUROC = 0.828). Such accurate results are almost suspiciously good; sources of error (e.g. information leakage) are discussed in the Discussion section below.

Network-Based Classification

Network-based classification was more complex than the standard text models. For this, I constructed an on-line network of cosponsorships in the 109th Senate. Each node was a senator; each edge represented one time that a senator had cosponsored another senator's bill, weighted by number of occurrences.[4] I built and used two different network types: directed and undirected (in the directed model, the direction of the edge pointed to the Senator who was the primary sponsor). I used information from both of these models to predict bill passage. A visualization of the network is shown in Figure 2.

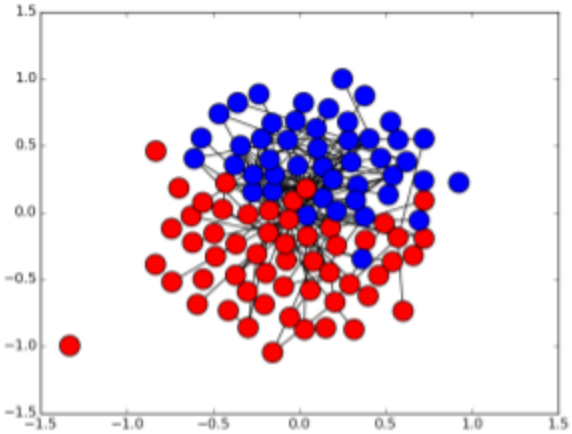


Figure 2: An undirected sponsorship network of the 109th Senate. Nodes are Senators, edges are cosponsorships

[4] Note that cosponsoring a bill together (i.e. Senators A and C cosponsoring a bill) does not create an edge between Senators A and B. I made this decision to reflect my understanding of how the dynamic of sponsorship works; one Senator writes (or, realistically, their staff writes) a bill, and that Senator is trying to gather the support of other Senators as cosponsors. Senators A and B cosponsoring the same bill implies little about their relationship; rather, it only gives information about aspects of their relationship mediated by Senator C, information which the network reflects.

In order to garner useful information from these networks, I used notions of network centrality to measure influence. Intuitively, it seems likely that Senators that are able to gather large numbers of cosponsors, influential cosponsors, etc. are more likely to have their bills passed. As the exact nature of the relationship between the ability to gather cosponsors and influence is unclear, I used five different measures of centrality and let my predictive model decide what was important. Those five notions of centrality were degree centrality, pagerank centrality, closeness centrality, eigenvector centrality, and load centrality.

To generate features (centrality scores) for the classifier, I measured each Senator's centrality at the time of each bill's introduction, having initialized the network with 100 bills. For each bill, I then gathered the centrality scores of its sponsor and cosponsors in each graph. I averaged the centrality scores of a bill's cosponsors and added in the number of each bill's cosponsors to generated a set of 21 features (if there were no cosponsors, the value defaulted to 0).

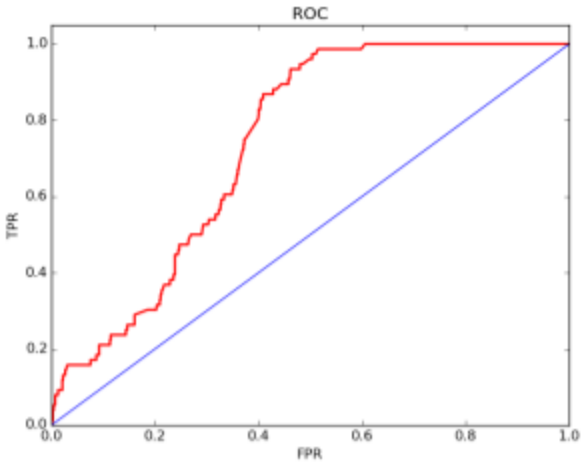


Figure 3: ROC curve for the best model from the network analysis classification pipeline.

Once I had my features, I fed those features into a classifier pipeline that performed a grid search to find the best model/parameter combinations across 10 different models. Random Forest classifiers performed particularly well, and the ROC curve from one of the top models (number 1 in the output file) is shown in Figure 3 (AUROC = 0.73). The only identifiable source of information leakage in this model is during model selection--not training or evaluation--so the results from this classifier are considerably less suspicious.

Integrated Classification

While a variety of methods for integrating the predictions of various classifiers exist, I found the simplest and most effective way of doing so was simple voting. For my final classifier, I rebuilt the top classifiers from each pipeline, evaluated them with 7-fold temporal cross-validation to double-check their predictive usefulness, and aggregated their predictions. Whichever option--passed or not passed--that got predicted most often by the top models was chosen as the final prediction[5] (changing the threshold for predicting passage did not increase overall predictive usefulness).

The resulting classifier was incredibly accurate. With an F1 score of 0.979, the model showed a very strong ability to differentiate between the characteristics of bills that passed relative to those that did not pass in my dataset.

General interpretation of this robustness must take into account the possible areas of information leakage discussed below. However, it is encouraging that the integration of the top classifiers drastically increased the ability of the classifier to discriminate between classes. Despite possible information leakage, the clear benefits of integrating these disparate types of classifiers justifies the initial goals of this project.

Discussion

Sources of Error

Data Gathering and Cleaning

When I downloaded the data, I got the first version that was presented on Congress.gov. Unfortunately, that was the most recent version, not the initial version.

[5] The tiebreaker was defaulting to the most common class (i.e. not-passed). I also considered defaulting to the consensus prediction of the more powerful classifiers, but ultimately decided that, for such a method to be defensible, it would require factoring in the certainty of those predictions (i.e. a weak classifier giving a certain prediction may be better than a strong classifier giving an uncertain one), thresholds, etc., which added unnecessary complexity.

As a result, there is likely significant information leakage in the text analysis model; phrases such as “Public Law” literally denote passage, and a model that just looks for those phrases would do very well as mine does.

To fix this error, I have two options: first, I can try to clean the data better. I made a cursory attempt to remove the problematic phrases, but to be effective, such a task would need to be much more in-depth. The second (much better) option is to go and get the initial versions of the bills. While this is feasible to have corrected, I unfortunately ran out of time as I got my final results late in the quarter; I prioritized the network analysis model earlier.

Cosponsorship Timing

I made the assumption that the sponsorship and cosponsorships of a bill are determined by the time a bill is introduced. That is, in fact, not true. In the Senate (unlike the House of Representatives), cosponsors can add themselves to bills at any time. Intuitively, there is likely to be some relationship between a bill’s likelihood of passage (as determined by Senators after introduction) and influential sponsors being willing to sponsor a bill.[6] As a result, failing to factor in initial vs. later sponsors likely leads to an unrealistic model.

More generally, I am trying to predict passage when a bill is introduced. As such, I should use the initial sponsors in my predictions, and only add in the later sponsors as they added themselves on in real life. Instead, I include the cosponsors when I’m predicting (i.e. at my simulated introduction), which uses information from the simulated future. In order to fix this leakage problem, I would need data from Senate records about when each Senator added themselves on as a cosponsor to each bill.

Model selection

During model selection, I used standard k-fold cross validation to evaluate my models. In doing so, I trained on bills that were introduced after some of the bills I was

[6] Interestingly, this relationship is not that more cosponsors implies a higher likelihood of passage (no evidence by logistic regression).

trying to predict. As there is likely to be a relationship between past and future bills (even as a bill referencing a past bill), my model introduced mild information leakage. Instead, I should have used temporal cross validation to evaluate my models.

That said, this is probably a relatively minor source of error. Given that the textual structure of bills is very likely similar across congresses, I would still be justified in using this same model if predicting for today's Congress, just trained on different data.

Possible Extensions

Bill Network

One thing I wanted to do was to create a network with the bills as nodes and cosponsorships as edges (i.e. if person A sponsor/cosponsors bills 1 and 2, there is an edge between bills 1 and 2). With this network, I could introduce a new bill and measure its relationship to or centrality relative to other successful bills. That measurement would be a rough measurement of how effective the sponsors of the new bill's sponsors are at getting legislation passed.

While the network functioned fine, because of the large increase in the number of edges with every new bill--the network required calculating every possible pairing of every senator's sponsorships--it became unmanageable to get centrality measurements from it. If I had access to a more powerful computer or could have parallelized the calculations, then it would have been more manageable.

Causal Analysis

One use of this algorithm (beyond, for example, online betting markets) is for legislators, activists or lobbyists to optimize their legislation for passage. That would require analysis of two aspects of the legislation: what exactly is causing the model to predict passage or non-passage, and how those elements could be changed to increase the probability of passage.

One way to do this inference is through feature importance. Especially in the random forest models that do best predicting based off of centrality measurements, it is easy to pull out feature importance measurements, which would give solid indicators as to the relative influence of each congressperson (in contrast to just their centrality).

A danger of this analysis is improper causal inference; if one were to, for example, pull out all the important words from the Naive Bayes model, that may suggest that adding words correlated with passage would increase the probability of the bill's passage--and the model would agree with you. But because the model is not drawing causal inferences itself, such a strategy would be invalid. More sophisticated methods would be warranted.

Conclusion

This project serves as a proof-of-concept for the idea of integrating network analysis and natural language processing methods for predicting bill passage. The network analysis methods (despite possible sources of error) demonstrated fairly convincingly the efficacy of using network analysis methods to predict bill passage. The text analysis methods, while incredibly effective with the dataset, proved less convincing as there was a significant source of error present.

Most importantly, the integration of these two methods by simple voting proved very effective at increasing the predictive efficacy of the overall model. It seems the combination was able to effectively compensate for the idiosyncrasies of each individual model, allowing it to find the signal in the proverbial noise. Overall this analysis seems to justify the methods it explored, and perhaps even warrant further exploration.