Deriving Ideal-Point Estimates for US Representatives Using Non-Policy Speeches

Alec MacMillen
University of Chicago
amacmillen@uchicago.edu

December 11, 2019

1 Introduction

At a foundational level, citizens of democratic countries run for representative office because they have some set of normative ideas about how society should operate, and they thus seek to win election and enshrine these beliefs through enacting legislation. Ideological differences on any number of policy dimensions distinguish the two major political parties in the United States, the Democrats and Republicans. But disagreements may not end there - to the extent that some exogenous political development may threaten the realization of one side's policy agenda, the two parties are likely to disagree about even the basic facts concerning that development.

For the American presidency, the most serious political development is the discovery of such "high crimes and misdemeanors" cited in the Constitution as grounds for impeachment. Only three presidents in the last century have either experienced impeachment or been in immediate danger of it: Richard Nixon in 1974 (in the wake of the Watergate scandal), Bill Clinton in 1998 (for perjury and obstruction of justice in connection to sexual harassment allegations), and Donald Trump in 2017 (for alleged bribery of the Ukrainian president).

These proceedings provide fertile ground through which to explore the question of whether ideology, pertaining nominally to policy preferences, can be derived from a representative's response to impeachment inquiries. When a president is impeached, it deals a serious blow to his party's standing and its ability to achieve its policy goals, and therefore it might be reasonable to expect that a legislator's attitude toward impeachment proceedings are a window into her own policy preferences, despite the fact that impeachment is fundamentally a political, not policy-related, process. This linkage between policy preferences and attitudes toward impeachment, if it exists, could prove significant in cases where a president has engaged in impeachable conduct but is protected by a plurality of legislators from his own party seeking to preserve their ability to enact policy. Fundamentally, this paper seeks to answer the question of whether a legislator's ideological standing can be derived from a body of text that is tied not to policy preferences but to non-political speech.

2 Literature Review

There is no shortage of existing literature that demonstrates the derivation of ideal point estimates from textual data. Gerrish and Blei (2001) develop predictive models of legislators' voting behavior using topic models, ideal point estimates and the contents of bills. They find that their ideal point topic models are more accurate in predicting roll call votes than a naive baseline. Lauderdale and Clark (2014) combine more traditional roll call unfolding with latent Dirichlet allocation to model judicial preferences by issue area in SCOTUS decisions. Kim and Ratkovic (2018) also combine a standard vote choice model with text from US Senate floor speeches and show that such a mixed model performs strongly in predicting future votes.

To my knowledge, however, there is no existing literature that attempts to model ideal points by actively *excluding* policy-related topics and focusing on inherently political, but non-policy, speech. In spirit, the methodology of this paper hews most closely to Klemmensen, Hobolt and Hansen (2007), who use a supervised wordscore approach on Danish manifestos and government speeches to derive ideal point estimates for Danish political actors in the postwar period. The authors then validate these estimates using party expert surveys, public opinion surveys, roll call data and party manifestos.

The key contribution this paper makes is following this method but actively discarding speech that does not fall under the umbrella of the exclusively political (i.e., non-policy). Congressional floor speeches on impeachment are excellent candidate documents for such an approach, because they by nature have very little tangible connection to policy positions and yet are likely to unveil significant differences between legislators of disparate ideological persuasions. That is, they satisfy a condition set forth by Lauderdale and Herzog (2017) that "existing approaches to scaling political disagreement from texts performs poorly except when applied to narrowly selected texts discussing the same issue."

3 Methodology

This paper will expand on the methodology of Klemmensen, Hobolt and Hansen (2007) by focusing solely on impeachment-related speeches, deriving ideal point estimates through a variety of methods, and validating these estimates against established measures of ideology.

3.1 Data Sources

The data for these analyses come from three primary sources: first, Gentzkow, Shapiro and Taddy (2018) have assembled a cleaned dataset of parsed congressional speeches for the 43rd-114th Congresses. These tables contain the text of speeches made from the House floor, as well as a set of speaker-level identifying characteristics that facilitate linkage to other data sources. Gentzkow et al's dataset does not cover the 116th Congress, so I developed a web scraper to manually gather data from the congressional record available online at https://congress.gov/congressional-record/.[1]

To validate the estimated ideology scores output by the deployed models, the congressional record data were linked to two validated measures of ideology: first, Poole and Rosenthal's DW-NOMINATE (Lewis et al. (2019)), which was originally developed in the 1970s to unfold legislative roll call votes into an observable two-dimensional scale. The final data source is Adam Bonica's DIME dataset (Bonica (2016)), an alternative to DW-NOMINATE that uses campaign finance contributions to plot the ideological leanings of political actors. Both DW-NOMINATE and DIME have been extensively validated in the political science literature, making them reliable benchmarks for the ideology estimates that will be produced by the impeachment speech data.

3.2 Preprocessing

In preparation for statistical analyses, a series of standard pre-processing steps were taken on the corpus. The overall body of floor speeches was filtered to include only those containing the string "impeach," which is a simple but effective heuristic to identify speeches of interest. It is a reasonably

[1] The data coverage from the Congressional Record for the 116th Congress ends on 11/21/2019. Future analysis should replicate and extend this work to include speeches from the full Congress, when it is available.

safe assumption that floor speeches mentioning impeachment do not contain many policy specifics, given that members are usually given only a minute or two to deliver their statements. It is possible that a simple binary filter for the presence of "impeach" may not fully exclude all policy specifics, so we may not be able to derive estimates for ideological standing from *purely* political, non-policy speech, but the amount of mixture is likely to be small.

After limiting the corpus to only impeachment-related text, speeches were combined at the legislator level so that there is one large document containing all impeachment-related speeches for each individual member. This grouping enables ideology estimates made at the legislator, rather than document, level. The text was then converted to lower case and stemmed, and numbers, whitespace and punctuation were removed. The corpus was also stripped of an extensive list of stop words, including not just basic English stop words but also a list specific to congressional speech.[2] Once the data were in clean form, they were transformed into document frequency matrices for analysis.[3]

3.3 Naive Bayes for Party Prediction

The first statistical analysis performed on the cleaned corpus of text will be a binary naive Bayes classifier as described in Jurafsky and Martin (2019). In essence, this classifier treats each document as a "bag of words" where a word's order does not matter, only its relative frequency does. The classifier uses Bayes' rule and a conditional independence assumption that word probabilities given document class are independent to probabilistically predict that document's class. In this case, I will train a naive Bayes classifier to predict the binary outcome of whether a given representative is a Democrat or not, given her speech. The results of this simple prediction task will provide a first clue at whether members of Congress are separable on some ideological dimension using the low-dimensional feature space of impeachment speeches.

[2]Gentzkow et al elucidate a detailed list of congressional-related stop words in their dataset, including titular words like "speaker" and procedural words like "yield".

[3]Terms occurring fewer than three times were also dropped from the document frequency matrices to ensure convergence of the wordscore and wordfish algorithms, and the `sparse = TRUE` option was selected in the `R` implementation of these functions.

3.4 Dictionary Methods for Sentiment Analysis

As a simple heuristic for positive vs. negative sentiment, I use two widely-applied sentiment dictionaries to judge the "feeling" of a particular representative's impeachment speech. The first, BING (Hu and Liu (2004)), applies a simple positive-negative binary indicator to words that have been judged to represent feelings of those types. The second, AFINN (Nielsen (2011)), takes a similar approach but adds a multiplier taking on the range of values -5 to 5 that allows different words to contribute differentially to the overall sentiment. This approach matches words in a representative's speech to a value in the sentiment dictionary, then averages together all word frequencies and values to produce a single sentiment score. Summarizing scores by party and regressing validated ideology on them will provide insight into whether sentiment appropriately sorts legislators' ideal points.

3.5 Wordscore and Wordfish for Ideal Point Estimation

The culminating analysis will focus on ideal point estimates derived from wordscore, a supervised dictionary learning method first introduced by Laver, Benoit and Garry (2003), and wordfish, an unsupervised scaling method developed by Slapin and Proksch (2008). A helpful overview of each method is provided by Grimmer and Stewart (2013). In essence, the wordscore approach calculates the relative frequency of words in *reference texts* used to anchor the liberal-conservative ends of an ideal point spectrum, and then measures the frequency of words in *virgin texts* to produce ideal point estimates. Wordscores are therefore sensitive to the exact documents chosen as references. In contrast, the unsupervised wordfish algorithm assumes that every word is drawn from a Poisson distribution modeled as a function of the individual's "loquaciousness", word frequency, extent to which a given word discriminates the underlying ideological space, and the politician's actual underlying position. The underlying position (ideal point) therefore "falls out" of the wordfish algorithm without defined reference texts.

The estimated wordscore and wordfish ideal points are untested, so to validate their predictive power, I regress well-established measures of ideology (DW-NOMINATE and DIME) on them to determine whether they have any actual predictive power for ideological preference. This finding addresses the paper's central question.

4 Analysis

Table 1 shows the distribution of impeachment-related speeches and the speakers that delivered these speeches for the three Congresses of interest. Figure 1 plots the frequency of impeachment-related speeches for members of the 105th Congress.[4] The distribution for the 105th Congress is unimodal with a peak at one speech, indicating that most members gave only one floor speech on the topic of impeachment. Filtering text on the string "impeach" results in 728 speeches by 359 distinct speakers, which is a drastic reduction from the high-dimensional space of all congressional speech (well north of 200,000 speeches in the 105th Congress). The implication is that analyses for the majority of members are based on a single document, so we should be heartened by even passable accuracy in classification and prediction, given that we are working with such a dramatically reduced feature space.

Table 1: Count of Speeches and Distinct Speakers Mentioning "Impeach", by Congress

Unit	93rd Congress	105th Congress	116th Congress
Distinct speakers	224	359	103
Speeches	730	728	210

As Figure 1 shows, members of both parties made comparable numbers of speeches about impeachment during the 105th Congress.

4.1 Predicting party affiliation with Naive Bayes

As a precursor to ideal point estimation, I first train a Naive Bayes classifier on 80% of available members in a given Congress (with 20% held out to test) in order to predict political party membership based on the contents of impeachment-related floor speeches. Table 2 plots the confusion matrix for this simple binary classifier. As is clear from the count of correct and incorrect predictions, the classifier for the 105th Congress performs strongly, with 95.8% accuracy.

Table 3 displays performance statistics for the same type of classifier trained on all three Con-

[4]For space and clarity, the body of the paper presents figures from analysis of the 105th Congress (Clinton impeachment) only. Figures corresponding to the 93rd Congress (Nixon impeachment) and 116th Congress (Trump impeachment) can be found in the appendices.

Figure 1: Impeachment speech counts by member party, 105th Congress

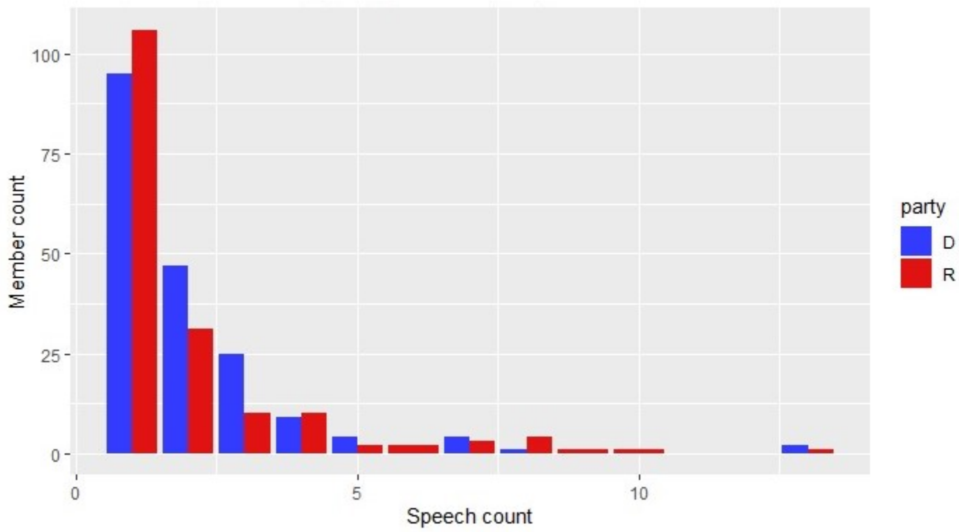


Table 2: Confusion Matrix: Party Identification, Naive Bayes Classifier, 105th Congress

Predicted outcomes		Observed outcomes: **D**	Observed outcomes: **R**
	D′	30 (41.7%)	2 (2.8%)
	R′	1 (1.4%)	39 (54.2%)

gresses of interest. The classifier for the 116th Congress also does well, with 83.3% accuracy, but the classifier for the 93rd Congress only attains a 64.4% accuracy rate. These results suggest that a binary prediction of members' party identification based on how they talk about impeachment is indeed possible. Better classification performance in the 105th and 116th Congresses suggest that there were starker differences between how the two parties spoke about impeachment, lending credence to the idea that the impeachment inquiry against Nixon in the 93rd Congress was a largely bipartisan effort, whereas the Clinton and Trump impeachment processes were much more polarized on a partisan basis.

Table 3: Model Performance Statistics for Party ID Naive Bayes Classifier

Statistic	93rd Congress	105th Congress	116th Congress
Accuracy	0.644	0.958	0.833
Precision	0.526	0.938	0.947
Recall	0.588	0.968	0.857
F1	0.556	0.952	0.900

4.2 Sentiment Analysis

To visually inspect whether there is any significant distinction in the sentiment of impeachment speeches by party, Figures 2 and 3 present, respectively, a boxplot of sentiment score and a line plot using the BING dictionary to show dispersion of members of the 105th Congress on a negative-positive sentiment scale. Figure 2 shows that Republican members of the 105th Congress spoke in markedly more favorable terms about the impeachment inquiry than Democrats. This observation is corroborated by Figure 3, in which clear left-right separation of the two parties is observable. The size and opacity of the dots on the scale line show the number of members whose impeachment speeches fell at that level of sentiment, with color corresponding to party. Although there is some overlap, there is a clear pattern with Democrats tending to fall on the lower (left, negative) side of the spectrum as opposed to Republicans on the higher (right, positive) end.

Figure 2: BING Sentiment Score by Party, 105th Congress

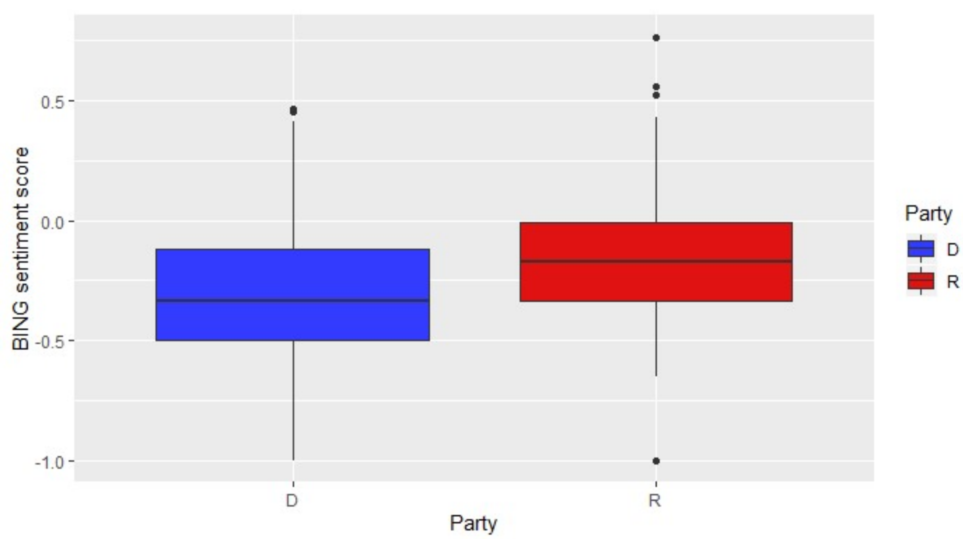
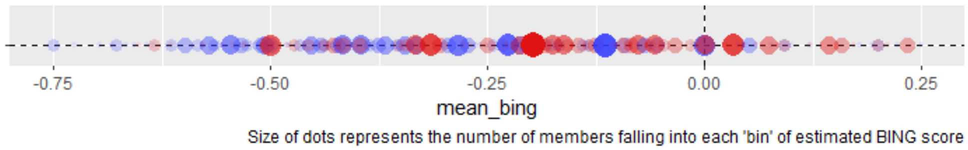


Figure 3: One-dimensional ideological dispersion, BING Sentiment Score by Party, 105th Congress



The same results are observable for sentiment scores developed using the AFINN dictionary for sentiment in Figure 4 and Table 5. Republican members of Congress used speech scored as "more positive" than their Democratic counterparts. Because BING and AFINN are dictionary-based sentiment models that rely on the relative frequency of word use, this finding suggests that we may be able to expand our modeling one step further from mere party identification to actual ideological estimation.

Figure 4: AFINN Sentiment Score by Party, 105th Congress

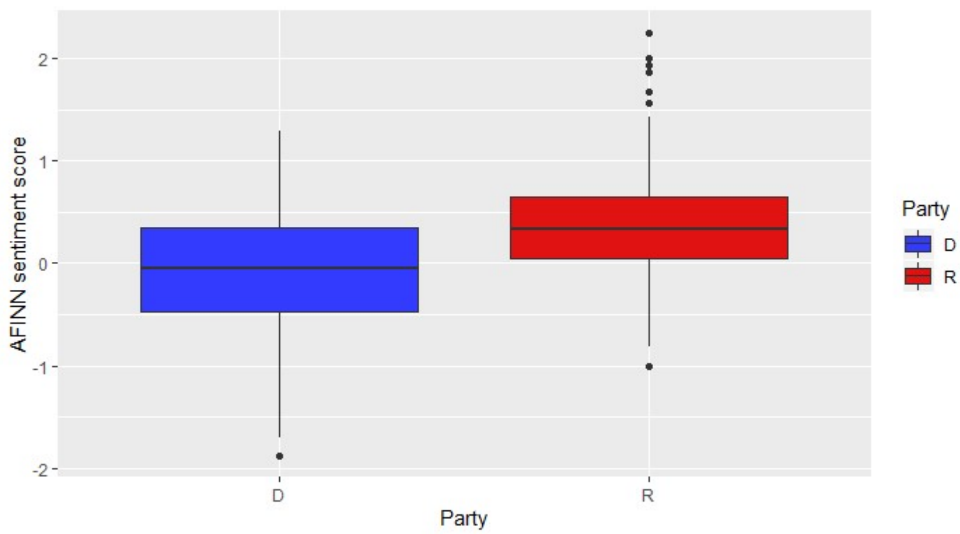
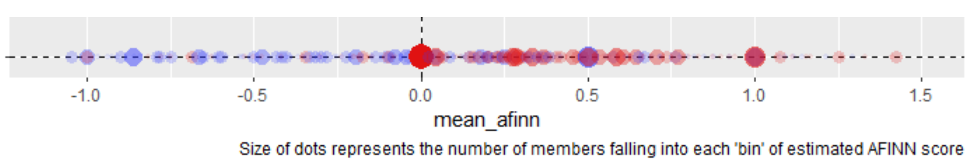


Figure 5: One-dimensional ideological dispersion, AFINN Sentiment Score by Party, 105th Congress

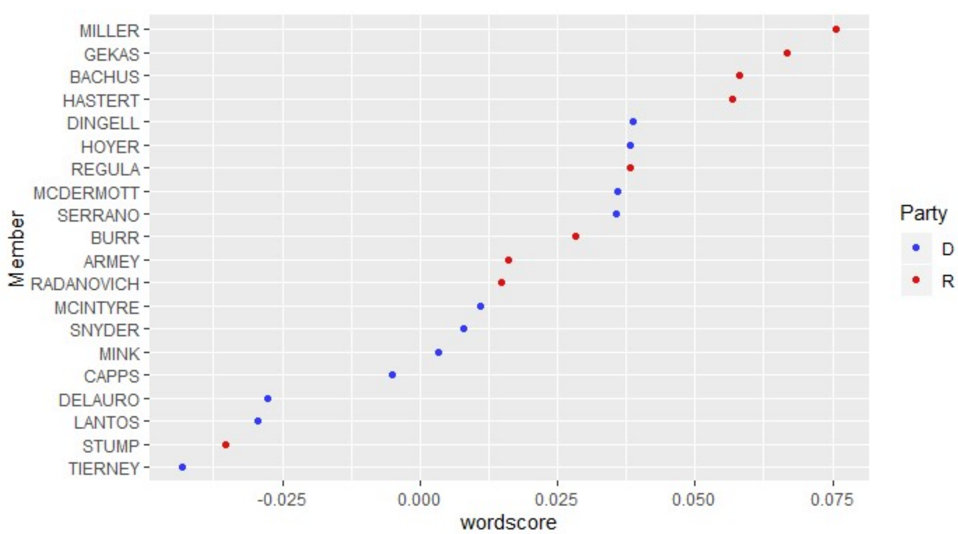


4.3 Wordscore and Wordfish

As explored in the methodology section, the chief difference between the wordscore and wordfish approaches to ideal point estimation using textual data is that the wordscore algorithm is supervised and the wordfish approach is unsupervised. That is, it is necessary to use "reference texts" in the wordscore approach to anchor the scale of possible ideal point values, while wordfish estimates emerge naturally from the corpus of training text. In the case of the 105th Congress, speeches attributed to Maxine Waters and Ron Paul were used to epitomize the left/Democratic and right/Republican sides of the scale, respectively.[5]

Figures 6, 7 and 8 show the ideological dispersion of members using wordscore estimates of ideal points. In particular, Figure 6 takes a random sample of estimates and plots members by name. The boxplot and line plot have the same interpretation as the analogous plots from the sentiment analysis models. As in that previous case, there is some overlap or mixture between members of the two parties, but even a cursory examination of the distributions shows that there is a significant difference in the mean estimated wordscore ideal point for Republicans as opposed to Democrats. It appears that there is some validity to the wordscore estimates in this case.

Figure 6: Sample of wordscore estimated ideology, 105th Congress



[5]John Conyers (D) and Harold Gross (R) were used for the 93rd Congress, and Al Green (D) and Steve Scalise (R) were used for the 116th Congress.

Figure 7: Boxplot of wordscore estimated ideology by party, 105th Congress

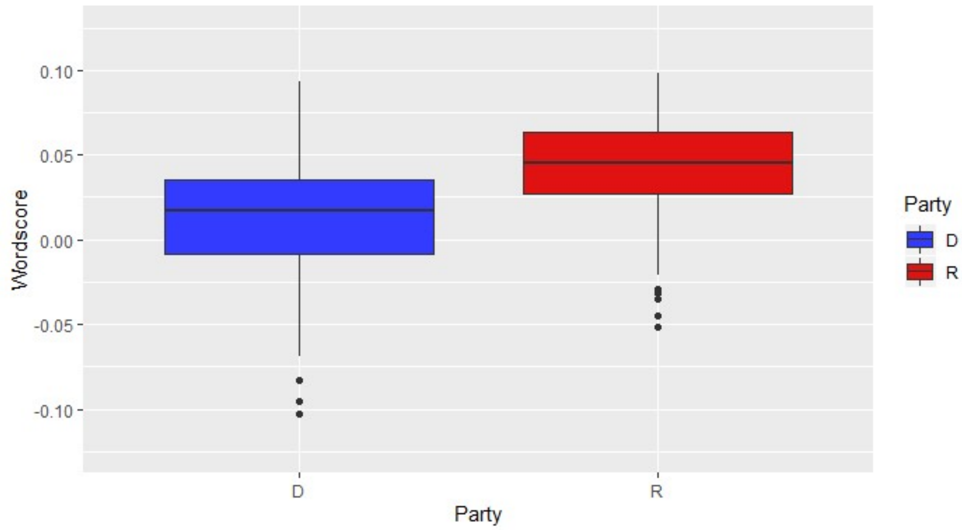
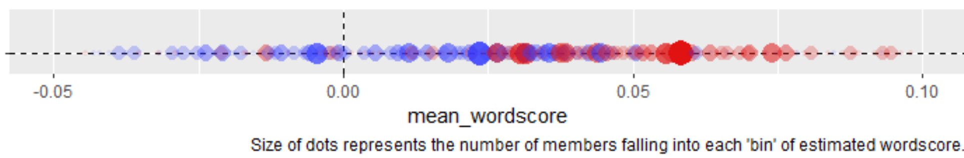


Figure 8: One-dimensional ideological dispersion by wordscore and party, 105th Congress



By contrast, the wordfish estimates are much more ambiguous. There is no easily discernible distinction between members of the two parties on the wordfish ideology scores, as depicted in Figures 9, 10 and 11. The sample plot of members by name reveals that representatives from both parties are scattered throughout the spectrum, and this finding is reinforced by the line plot that shows Democrats and Republicans overlapping at all levels of estimated ideology. It seems that the unsupervised approach is not an adequate predictor of ideal points. In essence, while there may be underlying similarities between impeachment speeches given by members of the same party that are identifiable by the supervised wordscore approach, these documents are not particularly separable without some human intervention to set the extreme points and anchor the scale of possible values.

4.4 Validating predicted ideology with established metrics

The observations about model performance in predicting ideal points have been to this point focused on visual inspection of estimate dispersion. Tables 4 and 5 report the results of a simple OLS regression of log normalized DW-NOMINATE and DIME on log-normalized ideal point es-

Figure 9: Sample of wordfish estimated ideology, 105th Congress

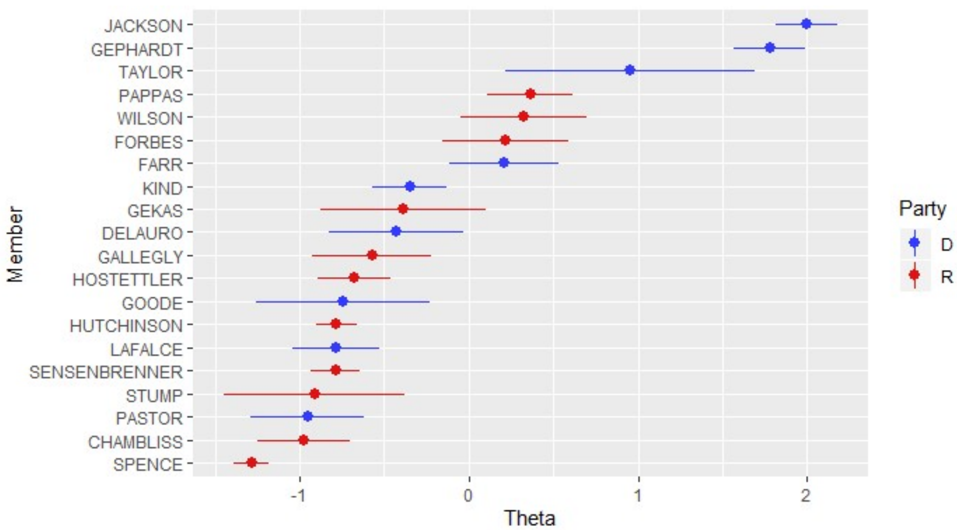


Figure 10: Boxplot of wordfish estimated ideology by party, 105th Congress

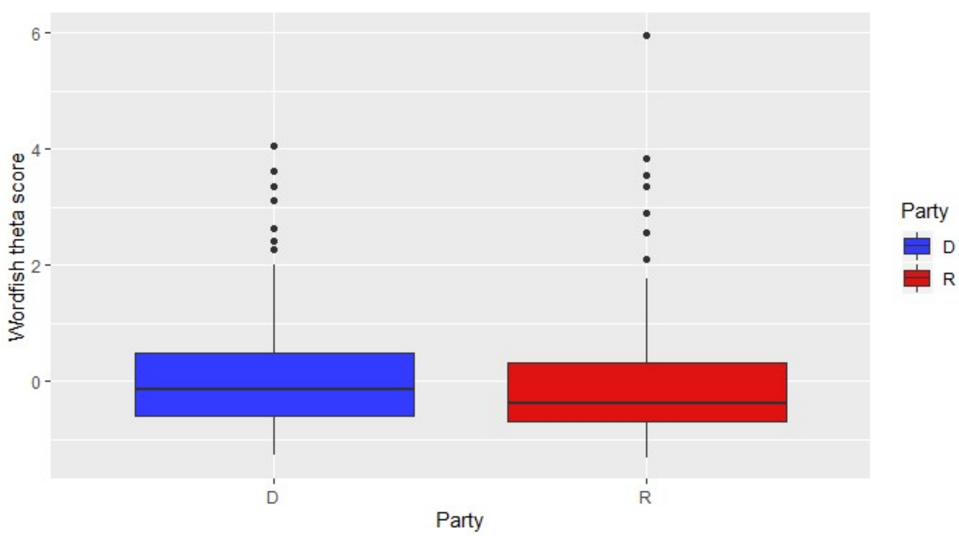
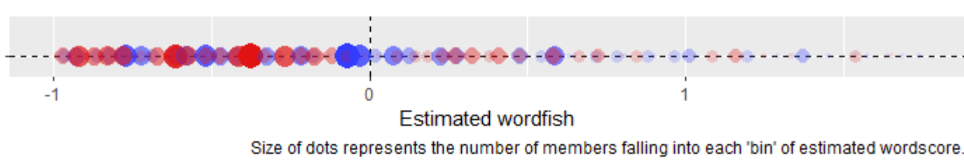


Figure 11: One-dimensional ideological dispersion by wordfish and party, 105th Congress



timate, respectively. Because DW-NOMINATE and DIME have been extensively validated, they are dependable benchmarks on which to evaluate the performance of the new ideal point estimates.

These ideal point estimates are inherently unit-less, so for ease of interpretation the dependent (DW-NOMINATE, DIME) and independent (sentiment, wordscore and wordfish estimates) variables have been normalized and logged to enhance interpretation.

Specifically, Table 4 shows that there is a strong positive and statistically significant relationship between DW-NOMINATE and the BING, AFINN, and wordscore measures in the 105th Congress. A 1% increase in log normalized estimated BING sentiment score is associated with a 0.48% increase in log normalized DW-NOMINATE rating. For AFINN sentiment, the same 1% increase results in a 0.58% increase in DW-NOMINATE score, and for wordscore rating, the 1% increase is associated with a 2.97% increase in DW-NOMINATE. This quantitatively described relationship corroborates the visually identified relationship discussed previously. Importantly, wordscore is the only new estimated measure that has a statistically significant relationship with DW-NOMINATE across all three Congresses investigated, suggesting that there may have been a higher level of partisan discord during the Clinton impeachment than the Nixon impeachment. The lack of a strong relationship between sentiment scores and DW-NOMINATE in the 116th Congress may have more to do with the quantity and quality of the data manually gathered for that period than with any actual difference in the partisan atmosphere, given reasonable prior beliefs about the rancor that characterizes the Trump impeachment inquiry. Notably, the wordfish model does not significantly predict DW-NOMINATE scores in any observed context.

The same relationship between sentiment scores, ideal point estimates and validated ideology scores holds true in Table 5, which performs the same analysis as the previous table but using Bonica's DIME scores in place of DW-NOMINATE.[6] We observe the same pattern of relationships: the estimated sentiment and wordscore measures strongly predict DIME ratings in the 105th Congress but not the 116th, and the estimated wordfish measure does not predict ideal points in any case. Given that the DIME results corroborate those observed using DW-NOMINATE, it seems clear that there was some fundamental underlying difference about the Clinton impeachment that allows for more accurate parsing of member ideology based on speech that is inherently non-policy related.

[6]Bonica's DIME data begins in 1980 and therefore does not cover the 93rd Congress.

Table 4: DW-NOMINATE Regression Coefficient Estimates by Congress

	Congress		
	93rd	105th	116th
Metric			
Log normal BING	0.042	0.482***	−0.184
	(0.193)	(0.115)	(0.148)
Log normal AFINN	−0.101	0.584***	0.032
	(0.189)	(0.126)	(0.201)
Log normal wordscore	0.809**	2.973***	0.757***
	(0.373)	(0.413)	(0.186)
Log normal wordfish	−0.357	−0.037	0.019
	(0.291)	(0.023)	(0.093)

*p<0.1; **p<0.05; ***p<0.01
Dependent variable is log normalized DW-NOMINATE score.

5 Discussion & Conclusion

Generally speaking, wordscore estimates of ideal points were predictive of validated measures of ideology across various contexts, while sentiment analysis scores had mixed results and wordfish estimates were not predictive in any scenario. This suggests that there is some merit to the claim that legislators' ideal points can be derived from inherently non-policy related speeches, and this finding is relevant to our understanding of political actors' calculation of how to act in high-stakes scenarios like impeachment inquiries. The null results (or diminished, in the case of the wordscore estimate) pertaining to the 93rd Congress are actually encouraging to this finding, when coupled with the qualitative prior that the Nixon impeachment inquiry enjoyed wide bipartisan support and Nixon resigned rather than face all-but-certain impeachment, conviction and removal from office. In a more homogeneous body of text, the proposed new estimates of ideology were less well able to parse between members with different ideal points, which makes intuitive sense.

The Clinton impeachment, on the other hand, was a much more divisive affair with Republi-

Table 5: DIME Regression Coefficient Estimates by Congress

	Congress		
	93rd	105th	116th
Metric			
Log normal BING	.	0.292***	−0.034
		(0.099)	(0.183)
Log normal AFINN	.	0.399***	0.173
		(0.108)	(0.237)
Log normal wordscore	.	2.222***	1.126***
		(0.364	(0.225)
Log normal wordfish	.	0.005	−0.004
		(0.019)	(0.100)

*p<0.1; **p<0.05; ***p<0.01
Dependent variable is log normalized DIME score.
DIME scores not available for the 93rd Congress.

cans strongly in favor and Democrats strongly opposed on a ground philosophical level, and this understanding is evinced by the strength of the relationships between ideal points derived from impeachment speeches and those presented by DW-NOMINATE and DIME. Furthermore, I suspect the surprising lack of correlation between the proposed new estimates of ideal points and DW-NOMINATE/DIME in the current 116th Congress has more to do with the dearth of data and the quality thereof: the web scraper used to parse the current Congressional record is by no means comprehensive and the null result produced should be treated with caution and revisited when Gentzkow et al update their parsed speech database.

One of the reasons there may have been difficulty in accurately ranking members using the unsupervised wordfish approach was that there was significant overlap in the terms used by members of all ideological leanings. Context and phrasing is especially crucial in this context, so reducing analysis to mere term frequency may not have captured the nuance that would distinguish liberal from conservative members. Future analysis could further explore whether and how “trimming” the document frequency matrix constructed from the impeachment speech corpus affects predictive accuracy. Perhaps words used occasionally, rather than very frequently or rarely, would be more accurate in predicting ideology. Additionally, it is important to note that the results for the sentiment analysis and wordscores method are sensitive to the specific dictionaries used for learning and to documents selected as reference text. Future work could also test a wider range of potential values for these inputs to see how sensitive the results are to various model specifications.

The implications of these findings are wide-ranging. If impeachment sentiment correlates with ideal points and by extension policy positions, it would not be unreasonable to assume that legislators might speak and act to protect a president of their own party or impugn a president of the opposite party regardless of the merits of an impeachment inquiry simply in order to bolster their own legislative agenda. Such actions could come at the expense of measured, thoughtful consideration of the evidence of impeachable acts and therefore undermine the integrity of an impeachment inquiry. Understanding the intricacies of this relationship is especially important given today's political climate and well-deserving of future inquiry.

6 Appendix

Supporting figures for 93rd Congress

Figure 12: Impeachment speech counts by member party, 93rd Congress

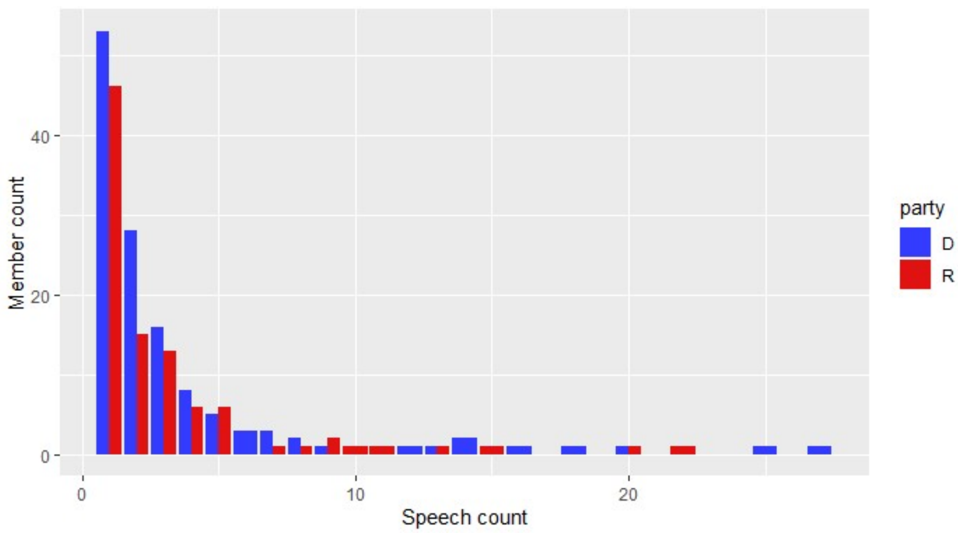


Table 6: Confusion Matrix: Party Identification, Naive Bayes Classifier, 93rd Congress

		Observed outcomes	
		D	**R**
Predicted outcomes	**D′**	10 (22.2%)	9 (20.0%)
	R′	7 (15.6%)	19 (42.2%)

Figure 13: BING Sentiment Score by Party, 93rd Congress

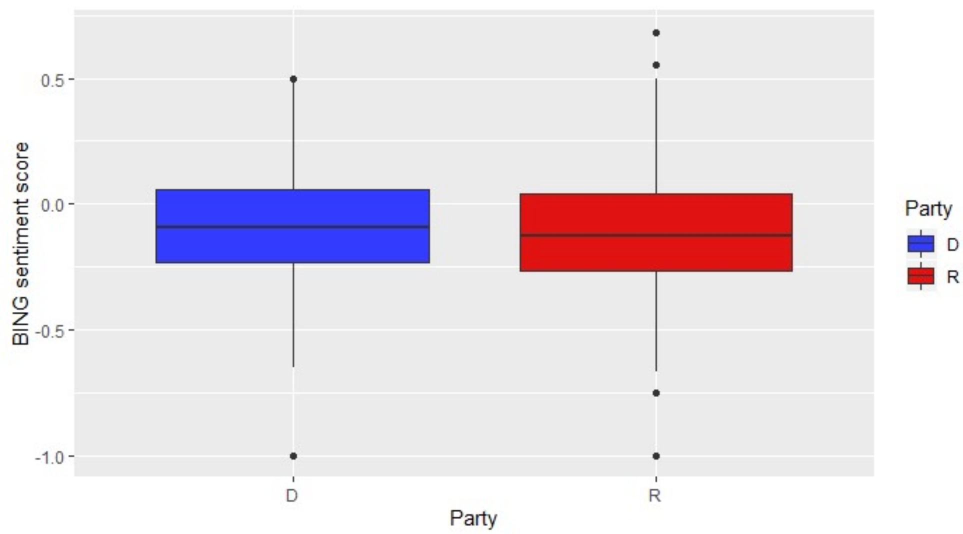


Figure 14: One-dimensional ideological dispersion, BING Sentiment Score by Party, 93rd Congress

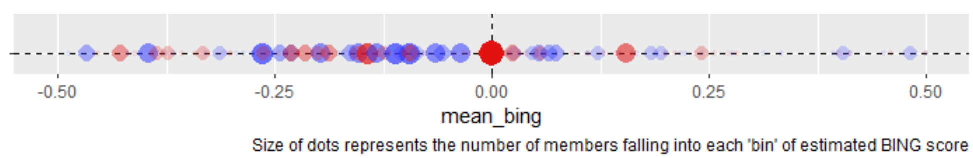


Figure 15: AFINN Sentiment Score by Party, 93rd Congress

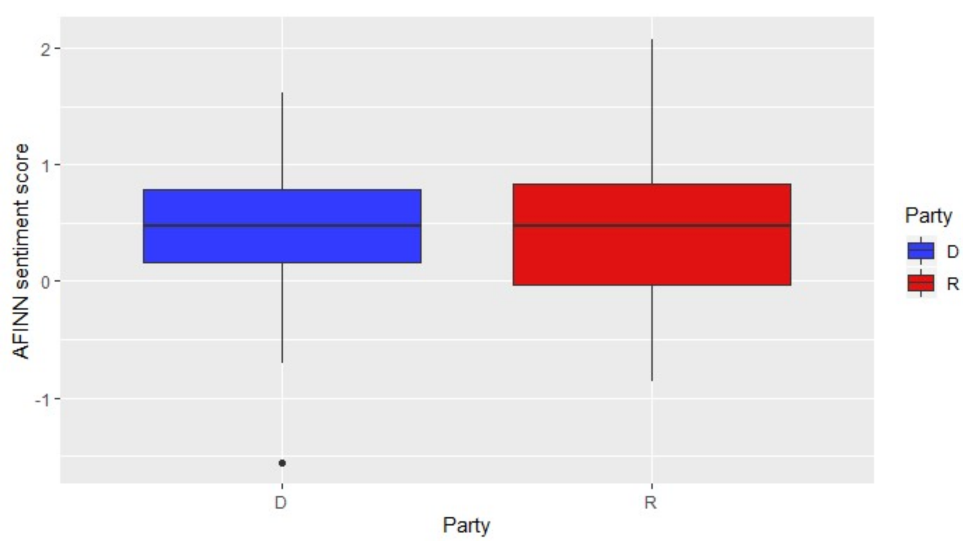


Figure 16: One-dimensional ideological dispersion, AFINN Sentiment Score by Party, 93rd Congress

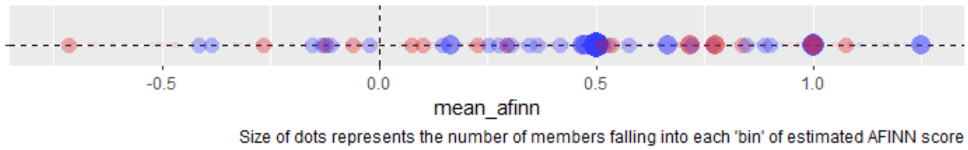


Figure 17: Sample of wordscore estimated ideology, 93rd Congress

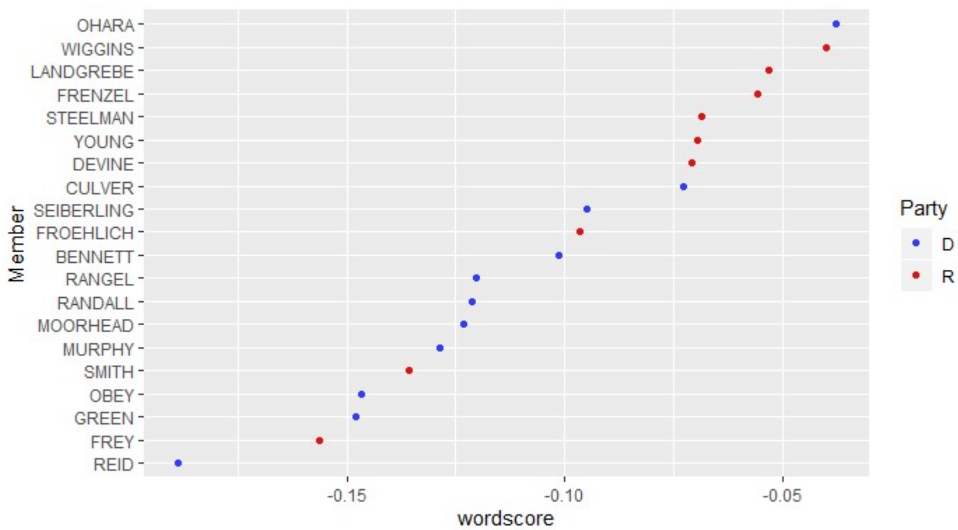


Figure 18: Boxplot of wordscore estimated ideology by party, 93rd Congress

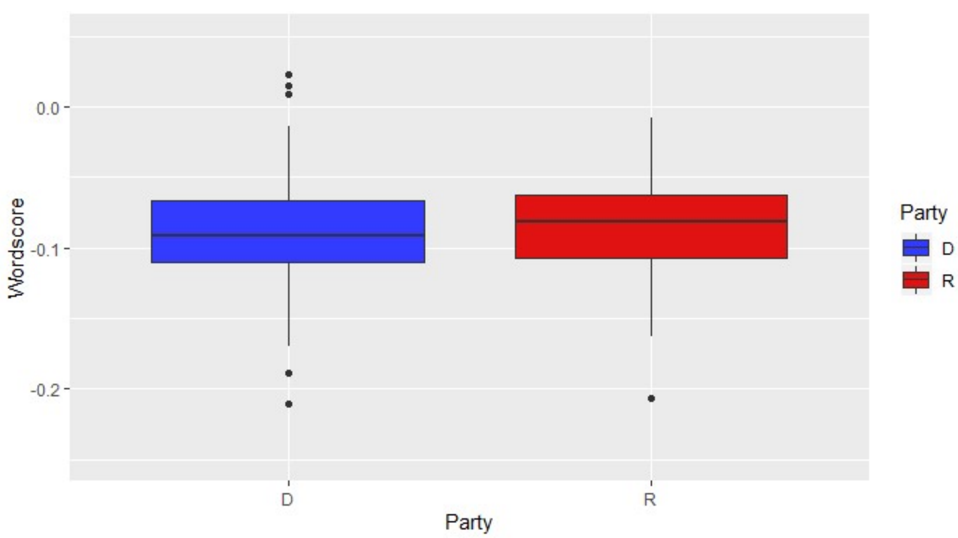


Figure 19: One-dimensional ideological dispersion by wordscore and party, 93rd Congress

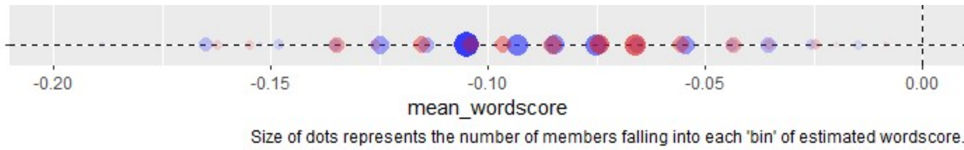


Figure 20: Sample of wordfish estimated ideology, 93rd Congress

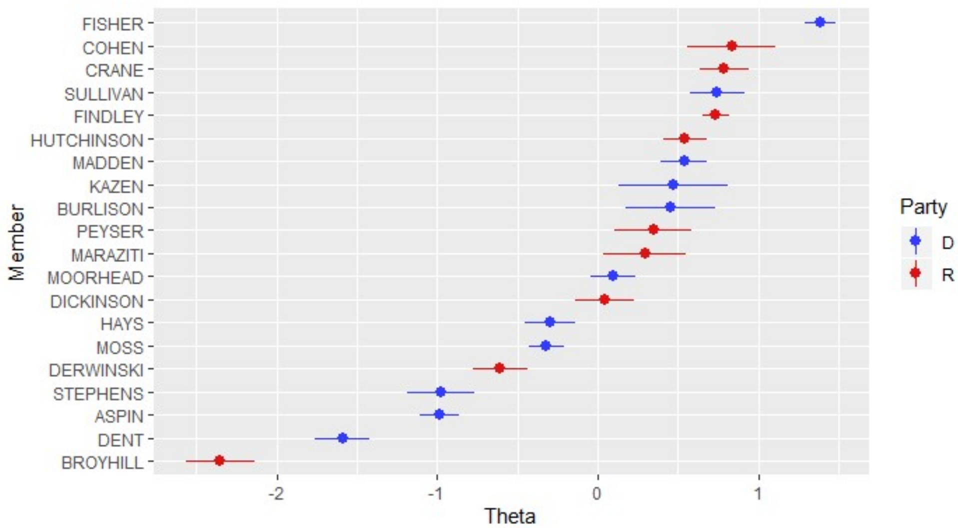


Figure 21: Boxplot of wordfish estimated ideology by party, 93rd Congress

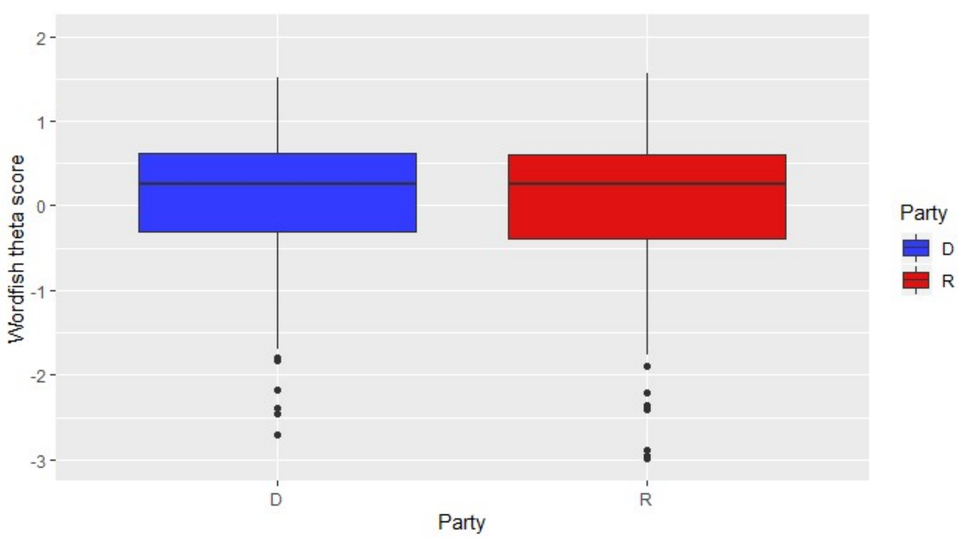
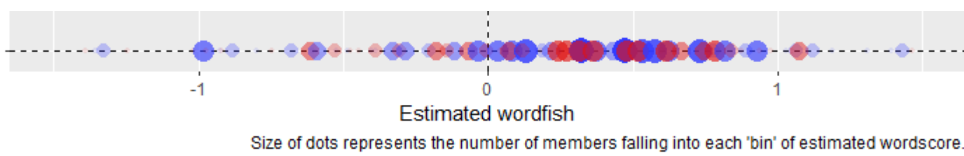


Figure 22: One-dimensional ideological dispersion by wordfish and party, 93rd Congress



Supporting figures for 116th Congress

Figure 23: Impeachment speech counts by member party, 116th Congress

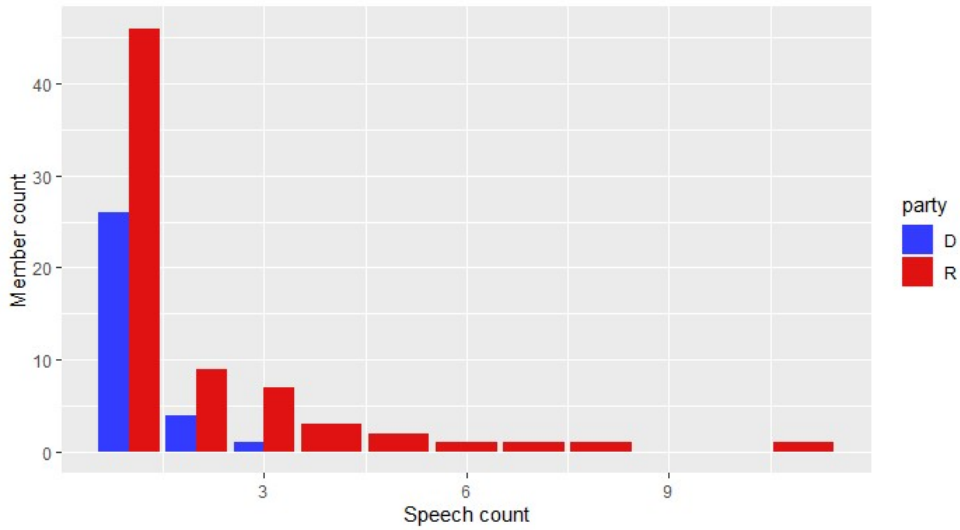


Table 7: Confusion Matrix: Party Identification, Naive Bayes Classifier 116th Congress

		Observed outcomes	
		D	**R**
Predicted outcomes	**D′**	18 (75.0%)	1 (4.2%)
	R′	3 (12.5%)	2 (8.3%)

Figure 24: BING Sentiment Score by Party, 116th Congress

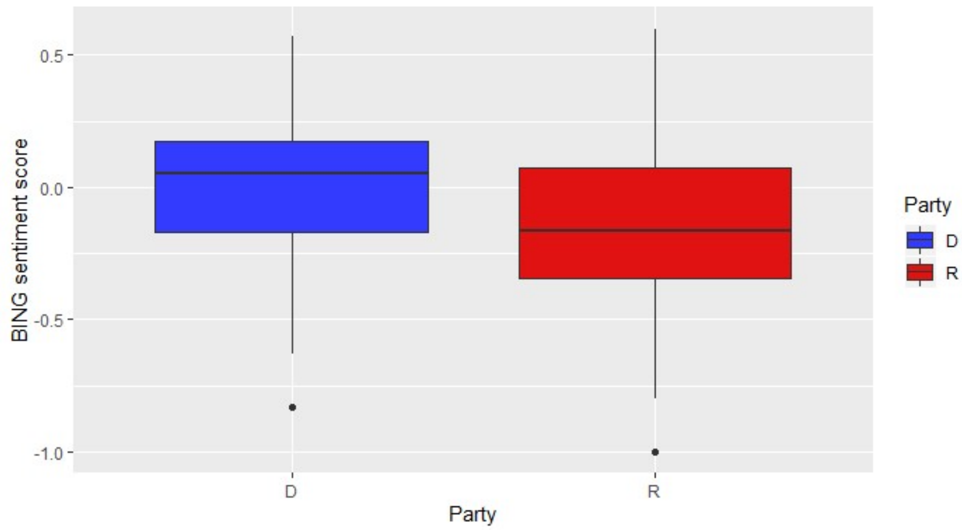


Figure 25: One-dimensional ideological dispersion, BING Sentiment Score by Party, 116th Congress

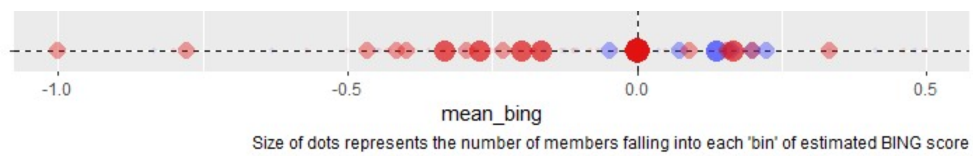


Figure 26: AFINN Sentiment Score by Party, 116th Congress

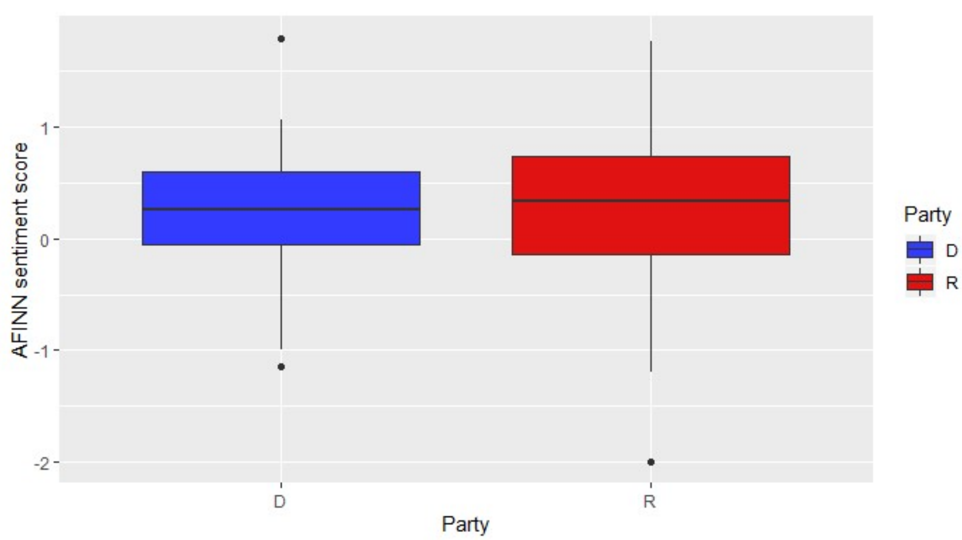


Figure 27: One-dimensional ideological dispersion, AFINN Sentiment Score by Party, 116th Congress

Figure 28: Sample of wordscore estimated ideology, 116th Congress

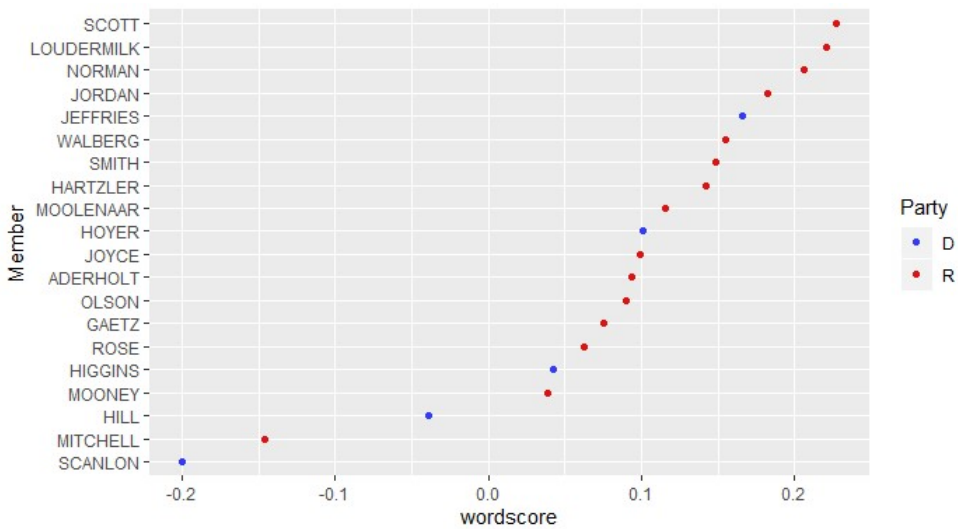


Figure 29: Boxplot of wordscore estimated ideology by party, 116th Congress

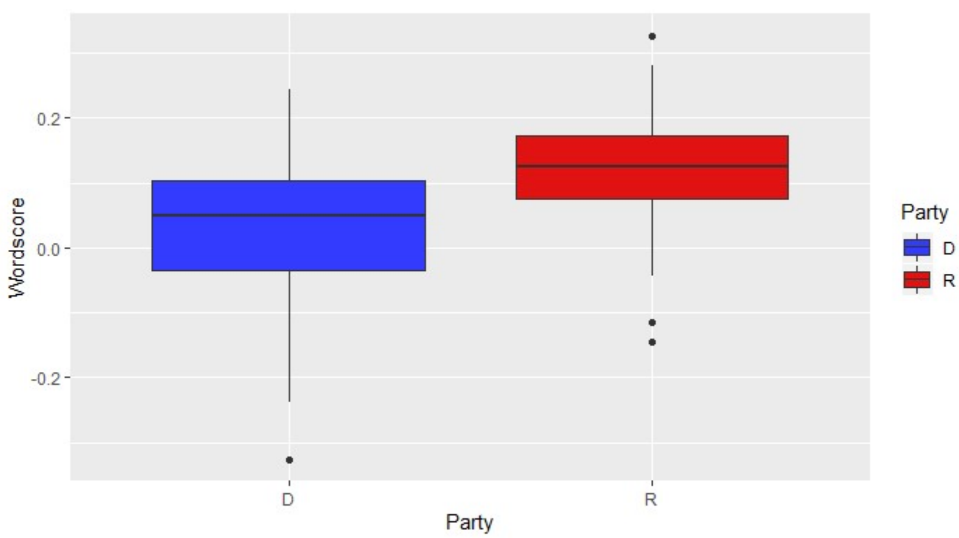


Figure 30: One-dimensional ideological dispersion by wordscore and party, 116th Congress

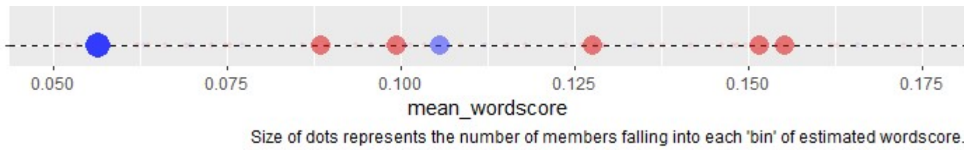


Figure 31: Sample of wordfish estimated ideology, 116th Congress

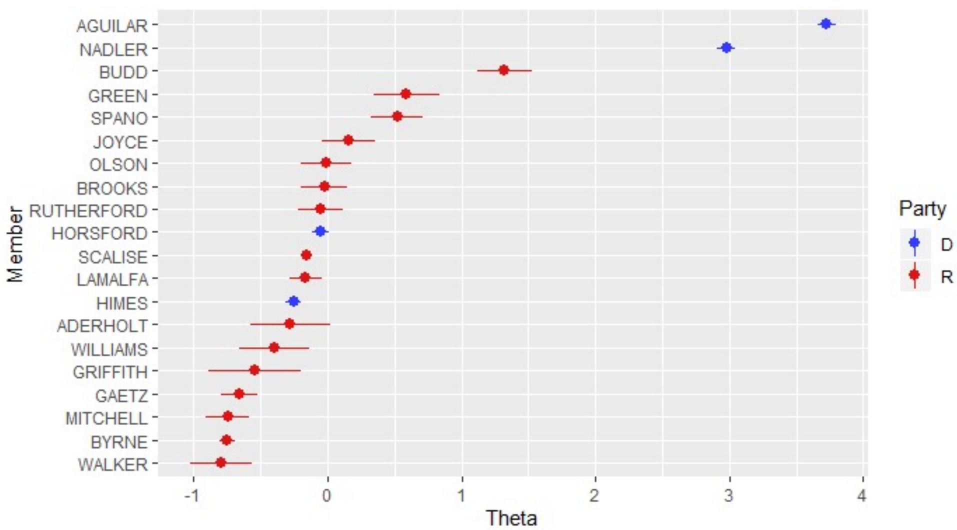


Figure 32: Boxplot of wordfish estimated ideology by party, 116th Congress

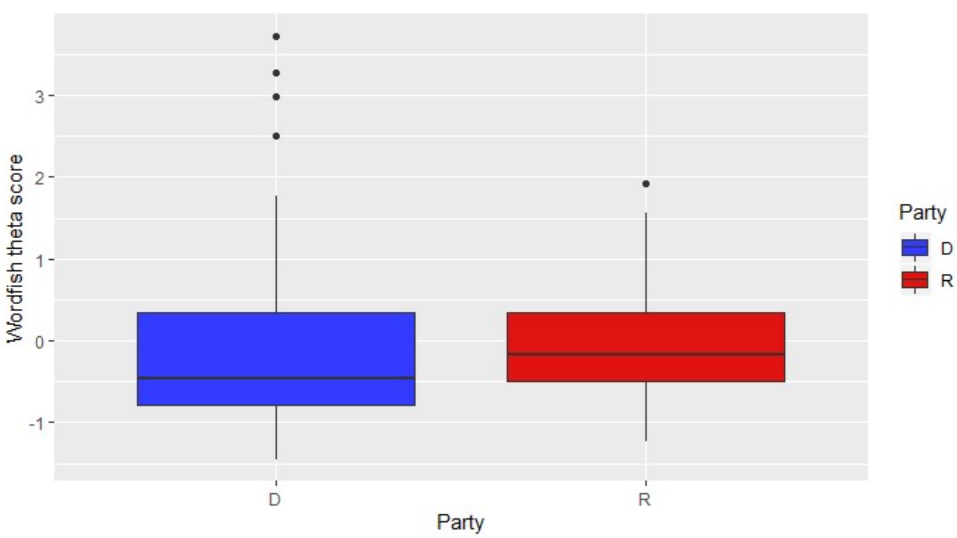
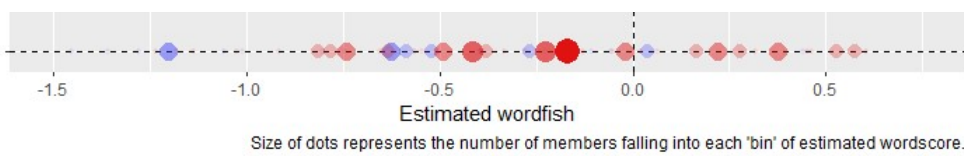


Figure 33: One-dimensional ideological dispersion by wordfish and party, 116th Congress



References

Bonica, Adam. 2016. "Database on Ideology, Money in Politics, and Elections: Public version 2.0." Stanford, CA: Stanford University Libraries. Available online at https://data.stanford.edu/dime.

Gentzkow, Matthew, Jesse M. Shapiro and Matt Taddy. 2018. "Congressional Record for the 43rd-114th Congresses: Parsed Speeches and Phrase Counts." Palo Alto, CA: Stanford University Libraries. Available online at https://data.stanford.edu/congress_text.

Gerrish, Sean M. and David M. Blei. 2001. Predicting Legislative Roll Calls from Text. In *Proceedings of the 28th International Conference on Machine Learning*. ICML pp. 489–496.

Grimmer, Justin and Brandon M. Stewart. 2013. "Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts." *Political Analysis* 21(3):267–297.

Hu, Minqing and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery Data Mining*. KDD.

Jurafsky, Daniel and James H. Martin. 2019. *Speech and Language Processing*. chapter 8, pp. 1–21. Available online at *https://web.stanford.edu/ jurafsky/slp3/4.pdf*.

Kim, In Song, James Londregon and Mark Ratkovic. 2018. "Estimating Spatial Preferences from Votes and Text." *Political Analysis* 26(7):210–229.

Klemmensen, Robert, Sara Binzer Hobolt and Martin Ejnar Hansen. 2007. "Estimating policy positions using political texts: an evaluation of the Wordscores approach." *Electoral Studies* 26(4):746–755.

Lauderdale, Benjamin E. and Alexander Herzog. 2017. "Measuring Political Positions from Legislative Speech." *Political Analysis* 24(3):374–394.

Lauderdale, Benjamin E. and Tom S. Clark. 2014. "Scaling Politically Meaningful Dimensions Using Texts and Votes." *American Journal of Political Science* 58(3):754–771.

Laver, Michael, Kenneth Benoit and John Garry. 2003. "Estimating policy positions from political texts." *American Political Science Review* 97(2):311–331.

Lewis, Jeffrey B., Keith Poole, Howard Rosenthal, Adam Boche, Aaron Rudkin and Luke Sonnet. 2019. “Voteview: Congressional Roll-Call Votes Database.”. Available online at https://voteview.com.

Nielsen, Finn Arup. 2011. Evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*. ESWC pp. 93–98.

Slapin, Jonathan B. and Sven-Oliver Proksch. 2008. “A Scaling Model for Estimating Time-Series Party Positions from Texts.” *American Journal of Political Science* 52(3):705–722.

In [1]:
```
import requests
import urllib.request
import time
from bs4 import BeautifulSoup
import re
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import matplotlib.pyplot as plt
from collections import Counter
import pandas as pd
import logging
import requests

from requests.adapters import HTTPAdapter
from requests.packages.urllib3.util.retry import Retry
```

In [2]:
```
# From congress.gov: month-day pairs for which there is an online congressional
# record to scrape for the 116th Congress, 1st session (2019)
session_dates = {1:[4,8,9,10,11,14,15,16,17,18,19,22,23,24,25,28,29,30,31],
                 2:[4,5,6,7,8,11,12,13,14,15,19,21,22,25,26,27,28],
                 3:[4,5,6,7,8,11,12,13,14,15,18,19,21,25,26,27,28,29],
                 4:[1,2,3,4,8,9,10,11,12,15,18,22,25,29,30],
                 5:[1,2,3,6,7,8,9,10,13,14,15,16,17,20,21,22,23,24,28,30,31],
                 6:[3,4,5,6,10,11,12,13,14,17,18,19,20,21,24,25,26,27,28],
                 7:[2,5,8,9,10,11,12,15,16,17,18,19,22,23,24,25,26,28,30,31],
                 8:[1,2,6,9,13,16,20,23,27,30],
                 9:[3,6,9,10,11,12,13,16,17,18,19,20,23,24,25,26,27],
                 10:[1,4,8,11,15,16,17,18,21,22,23,24,28,29,30,31],
                 11:[1,4,5,6,7,8,12,13,14,15,18,19,20,21,22,26]}
```

In [3]:
```
# Do not scrape Congressional Record pages with these words in the title
# this will eliminate most but not all non-member speech entries
STOPTITLES = ["prayer", "designation of the speaker pro tempore",
              "the journal", "pledge of allegiance",
              "announcement by the speaker pro tempore",
              "additional sponsors", "constitutional authority",
              "house of representatives", "public bills and resolutions",
              "expenditure reports", "report of expenditures",
              "adjournment"]
```

In [ ]:
```
# Iterate through the congressional record pages for 2019 session dates
# Scrape all pages that appear to us to be a member speech
speech = []
date = []
for month in range(1,12):
    print("Processing month ", str(month))
    for day in session_dates[month]:
        print("    Processing day ", str(day))
        # Build list of speech URLs to scrape
        url = 'https://www.congress.gov/congressional-record/2019/' + \
            str(month) + '/' + str(day) + '/house-section'
        response = requests.get(url)
        soup = BeautifulSoup(response.text, "html.parser")
        resultSet = soup.findAll('table', {'class':'item_table'})

        if not resultSet:
            continue

        # Make list of links to speeches to scrape from the given day
        intlist = str(resultSet[0]).split("<td>")
        linklist = [r for r in intlist if "CREC" in r and
                    not any(st in r.lower() for st in STOPTITLES)]

        for link in linklist:
            # Build the link to the specific speech page to scrape
            int_stem, _ = link.split("|")
            final_stem = re.findall('"([^"]*)"', int_stem)[0]
            url = "https://congress.gov" + final_stem
            print("        ", url)
            i = 1
            ss_response = None
            # Handle exceptions that come up for rate limit
            while ss_response is None:
                try:
                    ss_response = requests.get(url)
                    print("          Attempt ", i, " - success!")
                except:
                    print("          Attempt ", i, ": Some error, wait ",
                          str(i*5), " seconds")
                    time.sleep(i*5)
                    i += 1
                else:
                    continue

            ss_soup = BeautifulSoup(ss_response.text, "html.parser")
            # Isolate just the text and append to list
            ss_results = ss_soup.findAll("pre", {"class":"styled"})
            try:
                speech.append(str(ss_results[0]).split("\n\n\n\n")[1])
                date.append(str(month) + "-" + str(day) + "-19")
            except:
                continue
```

In [5]:
```
# List all state names
statenames = ["Alabama", "Alaska", "Arizona", "Arkansas", "California",
              "Colorado", "Connecticut", "Delaware", "Florida", "Georgia",
              "Hawaii", "Idaho", "Illinois", "Indiana", "Iowa", "Kansas",
              "Kentucky", "Louisiana", "Maine", "Maryland", "Massachusetts",
              "Michigan", "Minnesota", "Mississippi", "Missouri", "Montana",
              "Nebraska", "Nevada", "New", "Hampshire", "New Hampshire",
              "New Jersey", "New Mexico", "New York", "North Carolian",
              "North Dakota", "Jersey", "Mexico", "York", "North", "Carolina",
              "Dakota", "Ohio", "Oklahoma", "Oregon", "Pennsylvania", "Rhode",
              "Island", "Rhode Island", "South Carolina", "South Dakota",
              "South", "Carolina", "Dakota", "Tennessee", "Texas", "Utah",
              "Vermont", "Virginia", "Washington", "West", "Virginia",
              "West Virginia", "Wisconsin", "Wyoming"]
```

In [7]:
```
# Intermediate dataframe
speech_df = pd.DataFrame(
    list(zip(date, speech)), columns = ['date', 'speech'])
```

In [8]:
```
# Gather member last name and state to discern btw same-named members
# from the scraped text of the speech
lastnames = []
states = []
for i in range(len(speech_df)):
    try:
        parens = speech_df[
            'speech'][i][speech_df['speech'][i]
                         .find("(")+1:speech_df['speech'][i]
                         .find(")")].split(" ")
        ln = parens[1]
        state = ' '.join(w for w in parens if w in statenames)
    except:
        lastnames.append("NA")
        states.append("NA")
    else:
        lastnames.append(ln)
        if state in statenames:
            states.append(state)
        else:
            states.append("NA")
```

In [11]:
```
# Make the dataframe
speech_df_final = pd.DataFrame(
    list(zip(list(
        speech_df['date']), lastnames, states, list(speech_df['speech']))),
    columns = ['date', 'speaker', 'state', 'speech'])
```

In [12]:
```
# Do some cleaning of the speech field
speech_df_final['speech'] = speech_df_final['speech'].str.replace('\n', ' ')
speech_df_final['speech'] = speech_df_final['speech'].str.replace('_', '')
speech_df_final['speech'] = speech_df_final['speech'].str.replace('-', ' ')
speech_df_final['speech'] = speech_df_final['speech'].str.strip()
speech_df_final.head()
```

Out[12]:

	date	speaker	state	speech
0	1-4-19	SCHNEIDER	NA	GUN VIOLENCE (Mr. SCHNEIDER asked and was g...
1	1-4-19	WILSON	South Carolina	U.S. POLAND DIPLOMATIC RELATIONS (Mr. WILSO...
2	1-4-19	THOMPSON	Pennsylvania	HONORED TO SERVE IN THE 15TH DISTRICT IN THE 1...
3	1-4-19	HURD	Texas	RECOGNIZING THE CONTRIBUTIONS AND LIFE OF THE ...
4	1-4-19	BROOKS	Indiana	HONORING THE LIFE OF TYLER TRENT (Mrs. BROO...

In [13]:
```
# Export to CSV
speech_df_final.to_csv(".../Data/crec-116th/speeches_116.txt",
                       sep="|", index=False)
```

Analysis: 93rd Congress

Alec MacMillen

12/11/2019

Part 1: Load data

```
# Speech text
speech93 <- read_delim("Data/crec-93rd/speeches_093.txt",
                       delim = "|",
                       col_types = cols(speech_id = col_character(),
                                        speech = col_character()))

# Speaker map: identifying information about speaker
map93 <- read_delim("Data/crec-93rd/093_SpeakerMap.txt",
                    delim = "|",
                    col_types = cols(speakerid = col_character(),
                                     speech_id = col_character()))

dropspeech <- problems(speech93) %>%
  distinct(row) %>%
  mutate(speech_id = paste0("93", str_pad(row, 7, "left", "0"))) %>%
  .[[2]]

speech93 <- speech93 %>% filter(!(speech_id %in% dropspeech))
```

Part 2: Filter data to impeachment-related; basic exploratory analysis

```
impeach93h <- speech93 %>%
  filter(grepl("impeach", tolower(speech)) |
         grepl("impeachment", tolower(speech))) %>%
  left_join(map93, by = "speech_id") %>%
  filter(!is.na(speakerid) & chamber == "H")
```

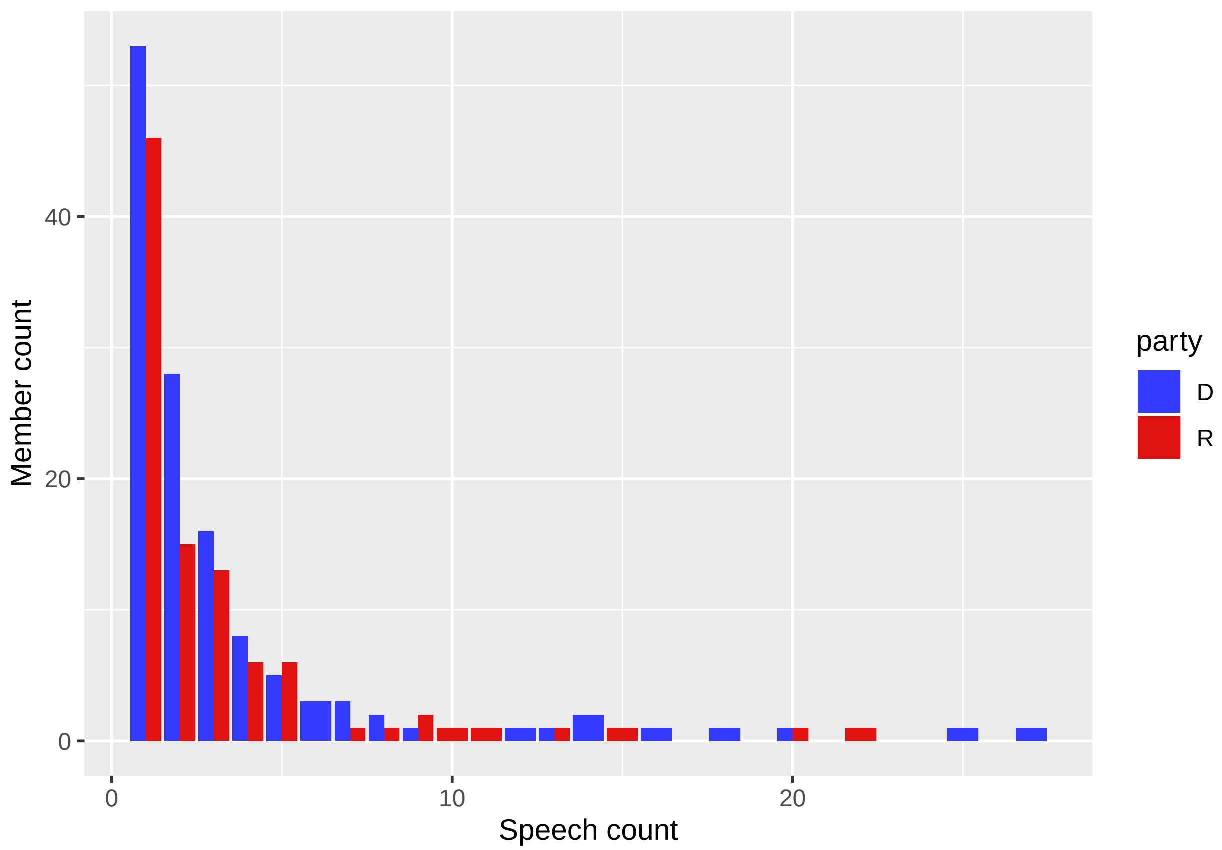
```
# For faster analysis, drop large "speech" field
impeach93h_eda <- impeach93h %>% select(-speech)

impeach93h_eda %>%
  group_by(speakerid) %>%
  summarize(count = n()) %>%
  skim(count)
```

```
## Skim summary statistics
##  n obs: 224
##  n variables: 2
##
## -- Variable type:integer -----------------------------------------------------------------
##  variable missing complete   n mean  sd p0 p25 p50 p75 p100     hist
##     count       0      224 224 3.26 4.2  1   1   2   3   27 <U+2587><U+2581><U+2581><U+2581><U+2581>
```

```r
impeach93h_eda %>%
  group_by(party) %>%
  summarize(count = n()) %>%
  print()
```

```
## # A tibble: 2 x 2
##   party count
##   <chr> <int>
## 1 D       437
## 2 R       293
```

```r
scount_by_party_93 <- impeach93h_eda %>%
  filter(party != "I") %>%
  group_by(speakerid, party) %>%
  summarize(count = n()) %>%
  ungroup() %>%
  arrange(count) %>%
  ggplot(., aes(count, fill=party)) +
  geom_bar(position = "dodge") +
  scale_fill_manual(values=group.colors) +
  labs(#title = "Dem and GOP members give similar number speeches",
       #subtitle = "On topic of impeachment, by party (93rd Congress)",
       y = "Member count",
       x = "Speech count")
scount_by_party_93
```

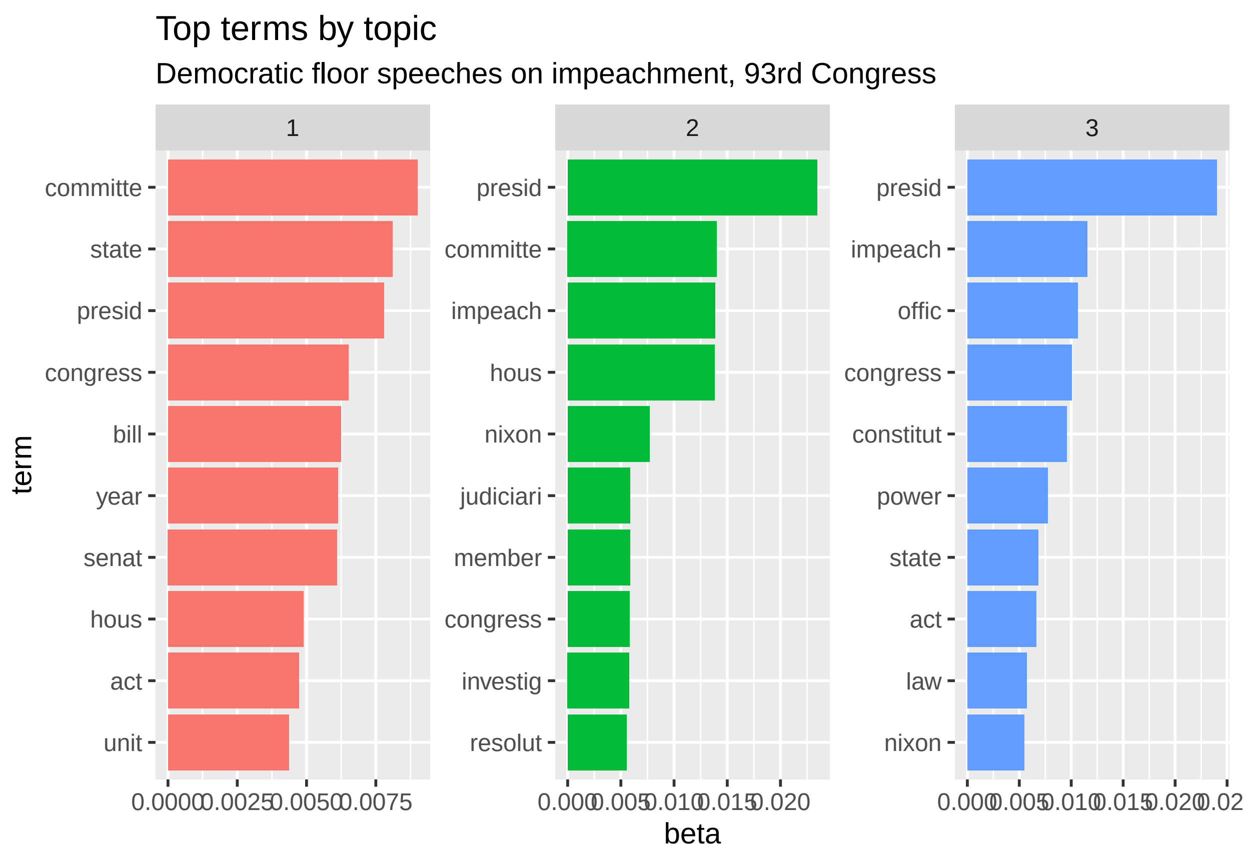


```r
# We're interested in modeling at the speaker/legislator level, so combine speeches by multiple speakers
impeach93hgrp <- impeach93h %>%
  group_by(speakerid, lastname, firstname, chamber, state, gender, party, district) %>%
  summarize(speech = paste0(speech, collapse = " ")) %>%
  ungroup() %>%
  rename(text = speech)

# Define stopwords: use basic English stopwords and Congress-related stopwords
c_stopwords <- c("absent", "adjourn", "ask", "can", "chairman", "committee",
                 "con", "democrat", "etc", "gentleladies", "gentlelady",
                 "gentleman", "gentlemen", "gentlewoman", "gentlewomen",
                 "hereabout", "hereafter", "hereat", "hereby", "herein",
                 "hereinafter", "hereinbefore", "hereinto", "hereof",
                 "hereon", "hereto", "heretofore", "hereunder", "hereunto",
                 "hereupon", "herewith", "month", "mr", "mrs", "nai", "nay",
                 "none", "now", "part", "per", "pro", "republican", "say", "senator",
                 "shall", "sir", "speak", "speaker", "tell", "tempore", "thank", "thereabout",
                 "thereafter", "thereagainst", "thereat", "therebefore", "therebeforn",
                 "thereby", "therefore", "therefor", "therefrom", "therein",
                 "thereinafter", "thereof", "thereon", "thereto", "theretofore",
                 "thereunder", "thereunto", "thereupon", "therewith", "therewithal",
                 "today", "whereabouts", "whereafter", "whereas", "whereat",
                 "whereby", "wherefore", "wherefrom", "wherein", "whereinto",
                 "whereo", "whereon", "whereto", "whereunder", "whereupon", "wherever",
                 "wherewith", "wherewithal", "will", "yea", "yes", "yield")

# Full stopwords list is congressional stopwords + base English stopwords
allstop <- c(stopwords("english"), c_stopwords)

# Split overall corpus into one for each party
dem <- impeach93hgrp %>% filter(party == "D")
demC <- VCorpus(VectorSource(dem$text))

rep <- impeach93hgrp %>% filter(party == "R")
repC <- VCorpus(VectorSource(rep$text))

# Function for corpus cleaning in preparation for topic modeling
cleanCorpus <- function(incorpus) {
  ccorp <- tm_map(incorpus, removePunctuation)
  for (j in seq(ccorp)) {
    ccorp[[j]] <- gsub("/", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("â ", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("@", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("/u2028", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("Ã¡", "a", ccorp[[j]])
    ccorp[[j]] <- gsub("â€", " ", ccorp[[j]])
  }
  ccorp <- tm_map(ccorp, removeNumbers)
  ccorp <- tm_map(ccorp, tolower)
  ccorp <- tm_map(ccorp, stemDocument)
  ccorp <- tm_map(ccorp, removeWords, allstop)
  ccorp <- tm_map(ccorp, stripWhitespace)
  ccorp <- tm_map(ccorp, PlainTextDocument)
```

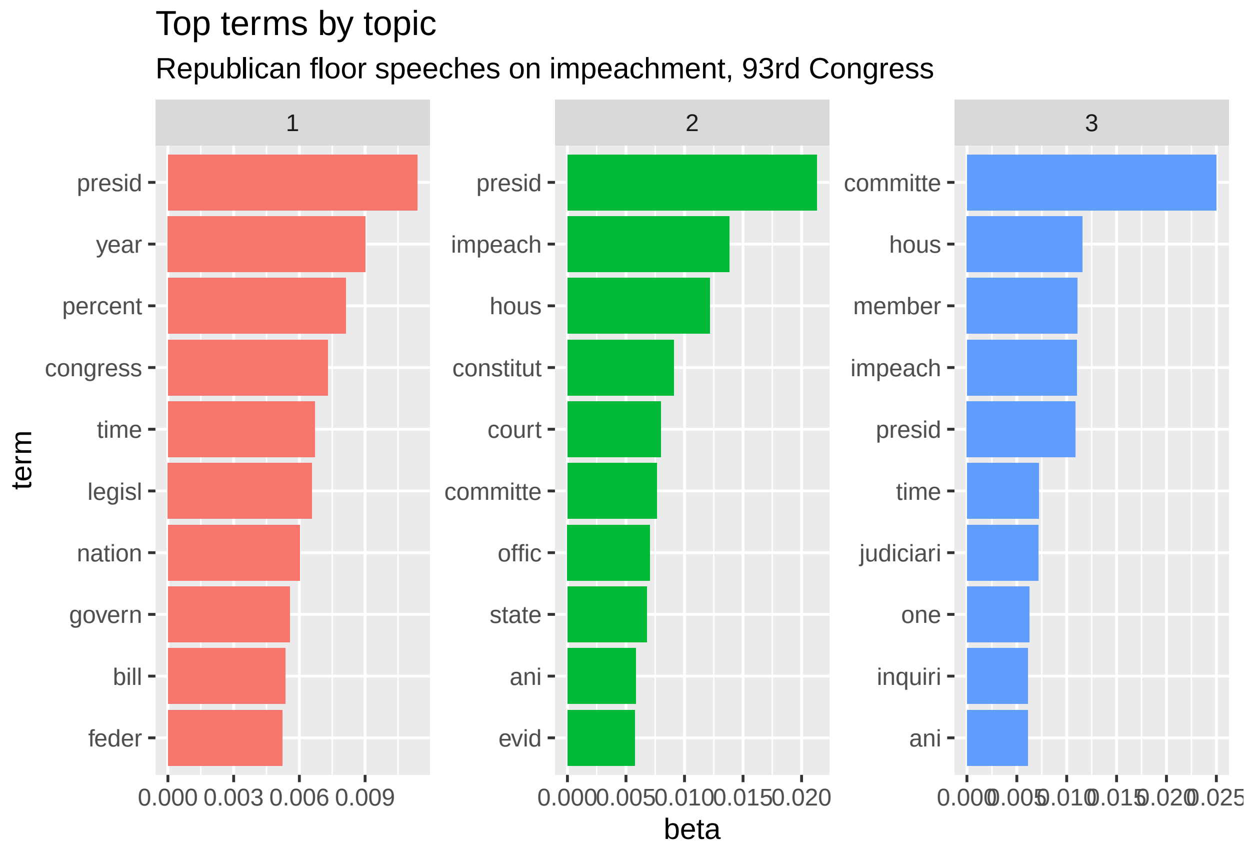
```r
  return(ccorp)
}

# Create document term matrix for each party's corpus
dem_dtm <- DocumentTermMatrix(cleanCorpus(demC))
rep_dtm <- DocumentTermMatrix(cleanCorpus(repC))
```

Topic models

```r
# These take about 80 seconds to train
dem_t3 <- topicmodels::LDA(dem_dtm, k = 3, control = list(seed = 101))
dem_topics <- tidy(dem_t3, matrix = "beta")

dem_top_terms <- dem_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, desc(beta))

dem_top_terms_plot <- dem_top_terms %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  coord_flip() +
  scale_x_reordered() +
  labs(title = "Top terms by topic",
       subtitle = "Democratic floor speeches on impeachment, 93rd Congress")
dem_top_terms_plot
```




```r
# These take about 80 seconds to run
rep_t3 <- topicmodels::LDA(rep_dtm, k = 3, control = list(seed = 101))
rep_topics <- tidy(rep_t3, matrix = "beta")

rep_top_terms <- rep_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, desc(beta))

rep_top_terms_plot <- rep_top_terms %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  coord_flip() +
  scale_x_reordered() +
  labs(title = "Top terms by topic",
       subtitle = "Republican floor speeches on impeachment, 93rd Congress")
rep_top_terms_plot
```



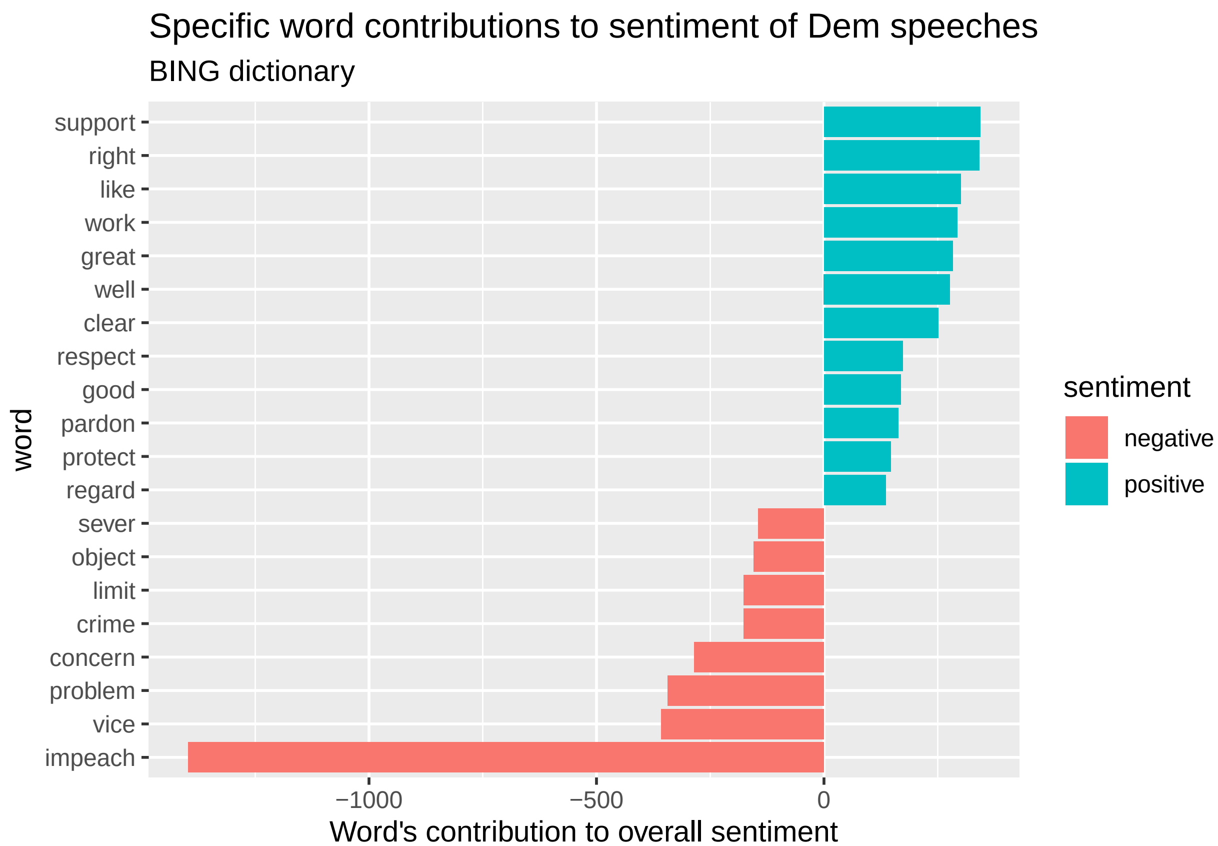
Sentiment analysis

```
freq_dem <- sort(colSums(as.matrix(dem_dtm)), decreasing=TRUE)
freq_dem_t <- tibble("word" = names(freq_dem), "n" = freq_dem)

bing <- get_sentiments("bing")
afinn <- get_sentiments("afinn")


# BING sentiment analysis
dem_bing <- freq_dem_t %>%
  inner_join(bing, by = "word") %>%
  mutate(ntone = ifelse(sentiment == "positive", n, -n)) %>%
  summarize(total_tone = sum(ntone),
            total_words = sum(n))

dem_bing$total_tone / dem_bing$total_words
```

```
## [1] -0.06471462
```

```
freq_dem_t %>%
  inner_join(bing, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
```

```r
  mutate(n = ifelse(sentiment == "positive", n, -n),
         word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Dem speeches",
       subtitle = "BING dictionary")
```

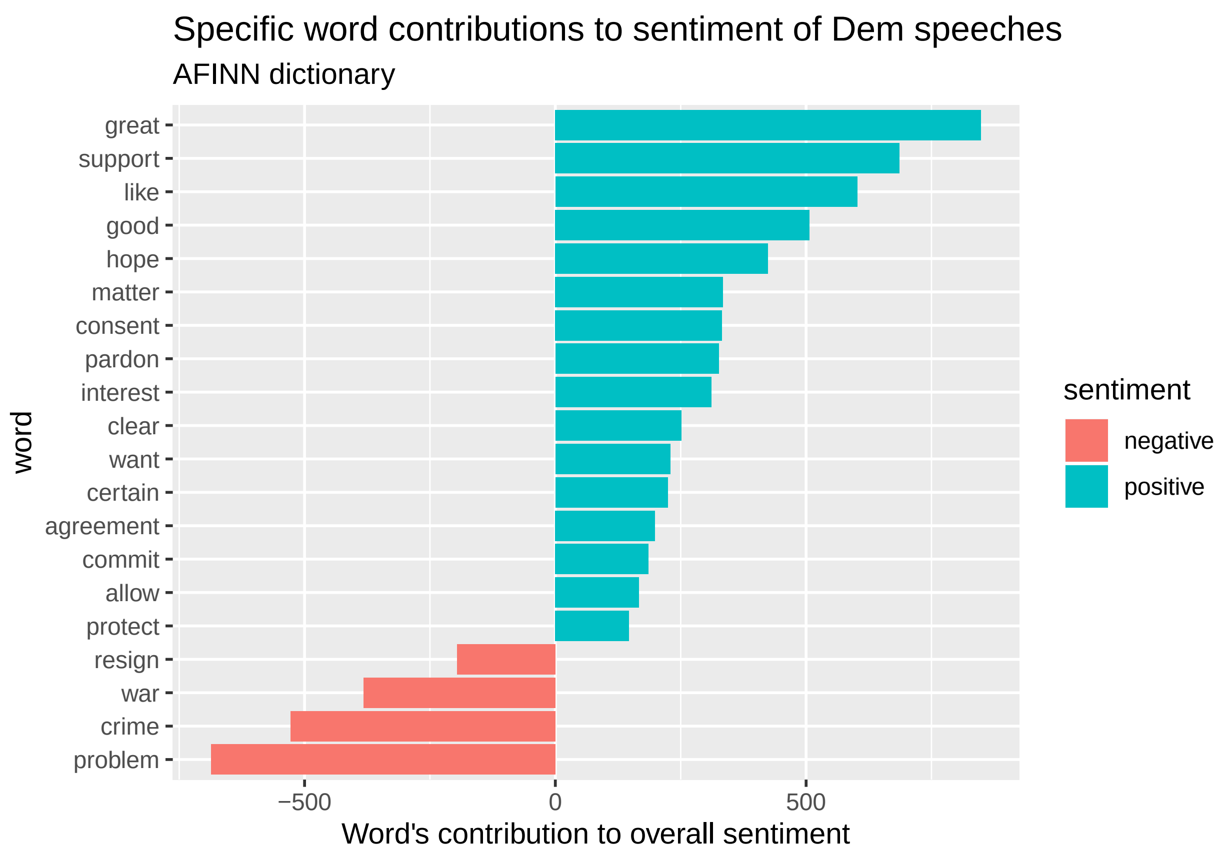



```r
# AFINN sentiment analysis
dem_afinn <- freq_dem_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(score = n*value) %>%
  summarize(totscore = sum(score),
            totwords = sum(n))

dem_afinn$totscore / dem_afinn$totwords
```

```
## [1] 0.4010586
```

```r
freq_dem_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
```

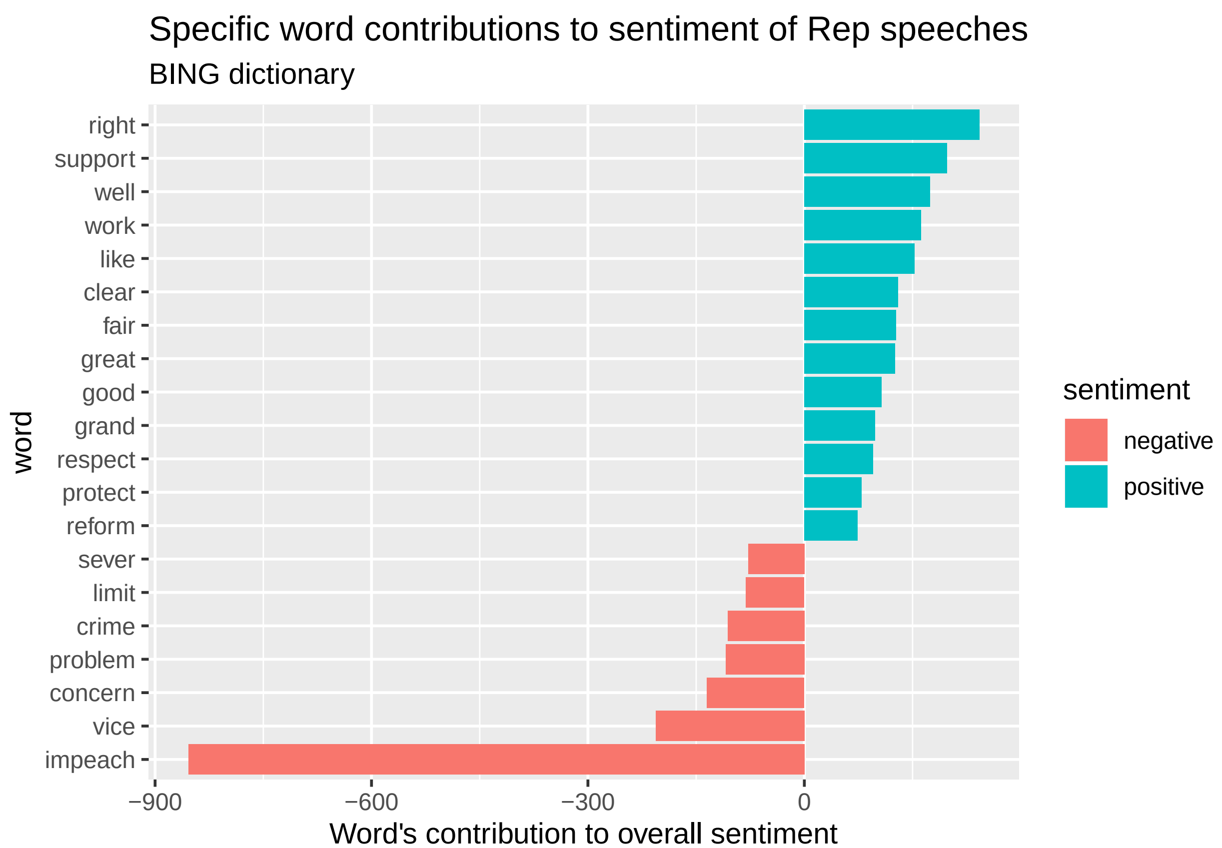
```r
  mutate(score = n*value,
         sentiment = ifelse(score >= 0, "positive", "negative"),
         word = reorder(word, score)) %>%
  ggplot(aes(word, score, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Dem speeches",
       subtitle = "AFINN dictionary")
```




```r
# Repeat sentiment analysis process for Rep
freq_rep <- sort(colSums(as.matrix(rep_dtm)), decreasing=TRUE)
freq_rep_t <- tibble("word" = names(freq_rep), "n" = freq_rep)


# BING sentiment analysis
rep_bing <- freq_rep_t %>%
  inner_join(bing, by = "word") %>%
  mutate(ntone = ifelse(sentiment == "positive", n, -n)) %>%
  summarize(total_tone = sum(ntone),
            total_words = sum(n))

rep_bing$total_tone / rep_bing$total_words
```

```
## [1] -0.06382393
```

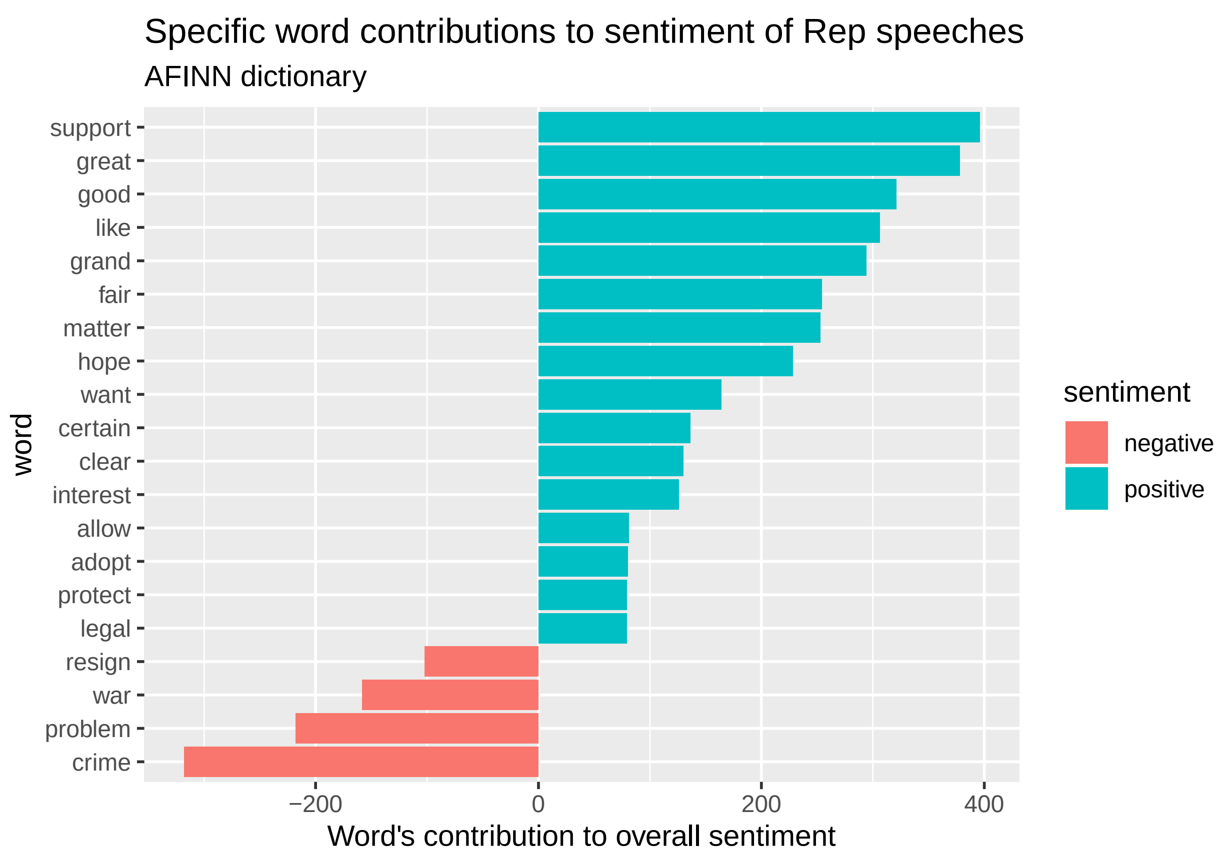
```r
freq_rep_t %>%
  inner_join(bing, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
  mutate(n = ifelse(sentiment == "positive", n, -n),
         word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Rep speeches",
       subtitle = "BING dictionary")
```




```r
# AFINN sentiment analysis
rep_afinn <- freq_rep_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(score = n*value) %>%
  summarize(totscore = sum(score),
            totwords = sum(n))

rep_afinn$totscore / rep_afinn$totwords
```

```
## [1] 0.4091514
```

```r
freq_rep_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
  mutate(score = n*value,
         sentiment = ifelse(score >= 0, "positive", "negative"),
         word = reorder(word, score)) %>%
  ggplot(aes(word, score, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Rep speeches",
       subtitle = "AFINN dictionary")
```



Part 3: Statistical analysis and prediction

```r
# Load and clean DW-NOMINATE
dwn <- read_csv("Data/dw-nominate/Hall_members.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   chamber = col_character(),
```

```
##   state_abbrev = col_character(),
##   bioname = col_character(),
##   bioguide_id = col_character(),
##   conditional = col_logical()
## )

## See spec(...) for full column specifications.
```

```
dwn93 <- dwn %>%
  filter(congress == 93 & chamber == "House") %>%
  select(state_abbrev, district_code, bioname, nominate_dim1, nominate_dim2) %>%
  rename(state = state_abbrev, district = district_code) %>%
  separate(bioname, into = c("lastname", "rest"), by = ",")
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 402 rows [1, 2,
## 3, 4, 6, 8, 9, 10, 12, 13, 14, 15, 16, 17, 18, 20, 21, 22, 23, 24, ...].
```

```
# DIME only starts in 1980, so we can't use it for the 93rd Congress

# Join DW-NOMINATE scores to speeches
impeach93analysis <- impeach93hgrp %>%
  mutate(dem = ifelse(party == "D", 1, 0)) %>%
  left_join(dwn93, by = c("lastname", "state", "district"))

# Check observations where first names don't match: robustness check for merge
problems <- impeach93analysis %>%
  filter(toupper(rest) != firstname) %>%
  select(-text, -speakerid)
# These all seem OK, so we can leave them in the analysis dataset (initials match first names, etc.)

# Create quanteda corpus object
analysis.corp <- quanteda::corpus(impeach93analysis)
summary(analysis.corp)
```

```
## Corpus consisting of 224 documents, showing 100 documents:
##
##     Text Types Tokens Sentences speakerid      lastname firstname chamber
##    text1   112    211        10  93101930        ARENDS    LESLIE       H
##    text2   159    323        13  93101960     BLACKBURN  BENJAMIN       H
##    text3   198    482        23  93101980        BRASCO     FRANK       H
##    text4   460   1287        55  93102020      BROYHILL      JOEL       H
##    text5   819   2532       122  93102060   CHAMBERLAIN   CHARLES       H
##    text6   962   4865       217  93102170        DENNIS     DAVID       H
##    text7   129    230        12  93102180       DONOHUE    HAROLD       H
##    text8   675   1958        90  93102220        FISHER      OVIE       H
##    text9   457   1028        57  93102230 FRELINGHUYSEN     PETER       H
##   text10   638   1852        83  93102240      FROEHLICH    HAROLD       H
##   text11   605   1750        78  93102280         GREEN     EDITH       H
##   text12   316    809        43  93102310         GROSS    HAROLD       H
##   text13   623   2086        63  93102340        GUNTER   WILLIAM       H
##   text14  1251   4473       182  93102350         HANNA   RICHARD       H
##   text15  3030  20002       858  93102430         HOGAN  LAWRENCE       H
```

```
## text16   780  2544    143 93102440   HOLIFIELD  CHESTER       H
## text17   131   227      9 93102460       HUBER   ROBERT       H
## text18   347   841     38 93102480        HUNT     JOHN       H
## text19   234   492     68 93102500        KING CARLETON       H
## text20   354   795     33 93102510  KUYKENDALL      DAN       H
## text21  1333  4721    202 93102530   LANDGREBE     EARL       H
## text22   224   586     29 93102560    MARAZITI   JOSEPH       H
## text23  1891 10563    451 93102590       MAYNE    WILEY       H
## text24   286   670     23 93102620   MCSPADDEN     CLEM       H
## text25  1599  6009    260 93102700      PODELL  BERTRAM       H
## text26   173   350     11 93102740        REID    OGDEN       H
## text27  3951 21081    702 93102750     ROBISON   HOWARD       H
## text28    29    36      2 93102790        RUTH     EARL       H
## text29   666  2611    129 93102800     SANDMAN  CHARLES       H
## text30   149   343     19 93102850       SMITH    HENRY       H
## text31   718  1916     86 93102930      VEYSEY   VICTOR       H
## text32  9117 89072   4171 93102940      WALDIE   JEROME       H
## text33   299   881     47 93102970    WILLIAMS LAWRENCE       H
## text34   366   979     47 93103000       YOUNG   SAMUEL       H
## text35   301   647     23 93103010        ZION    ROGER       H
## text36   370   866     31 93103030       WYMAN    LOUIS       H
## text37  4130 26489   1063 93103050       ABZUG    BELLA       H
## text38   741  2443    112 93103060      ALBERT     CARL       H
## text39   822  2578     96 93103130     BIESTER   EDWARD       H
## text40   900  3376    198 93103190      CULVER     JOHN       H
## text41   544  1523     54 93103200     DANIELS DOMINICK       H
## text42   203   369     19 93103260      FULTON  RICHARD       H
## text43   349   922     43 93103280       GREEN  WILLIAM       H
## text44   140   250     11 93103290        GUDE  GILBERT       H
## text45   549  1975    105 93103340        HAYS    WAYNE       H
## text46   312   700     32 93103350     HECHLER  KENNETH       H
## text47   490  1259     64 93103380       HICKS    FLOYD       H
## text48   989  5097    274 93103410     HUNGATE  WILLIAM       H
## text49   551  1773     68 93103420  HUTCHINSON       J.       H
## text50   433  1267     53 93103530      MADDEN      RAY       H
## text51  2080  8255    448 93103550 MCCOLLISTER     JOHN       H
## text52   322   749     32 93103570   MEZVINSKY   EDWARD       H
## text53  7604 61783   2747 93103630       OHARA    JAMES       H
## text54   455  1320     67 93103640     PASSMAN     OTTO       H
## text55   580  1573     61 93103650      PATMAN     JOHN       H
## text56   853  2661    118 93103670     RANDALL  WILLIAM       H
## text57    94   166      9 93103700       ROUSH     JOHN       H
## text58   272   758     21 93103750    STEELMAN     ALAN       H
## text59   115   219     12 93103760     STEIGER      SAM       H
## text60   335  1184     50 93103770    STEPHENS   ROBERT       H
## text61   329   955     36 93103810    SULLIVAN   LEONOR       H
## text62   705  2431    119 93103830     TALCOTT     BURT       H
## text63   163   332     19 93103850      TAYLOR      ROY       H
## text64   172   356     17 93103870    VIGORITO   JOSEPH       H
## text65   228   501     25 93103890     DU PONT   PIERRE       H
## text66   667  1808     67 93103930     BADILLO   HERMAN       H
## text67   113   229      6 93103970       BROWN    GARRY       H
## text68   766  2397     98 93104000       BURKE   YVONNE       H
## text69    92   157      5 93104070   CEDERBERG   ELFORD       H
```

```
##   text70   587   1883         88  93104130          DENT      JOHN        H
##   text71    33     47          2  93104140       EILBERG    JOSHUA        H
##   text72   281    706         31  93104160       FLOWERS    WALTER        H
##   text73   208    360         17  93104180        FRASER    DONALD        H
##   text74   165    382         19  93104190          FREY     LOUIS        H
##   text75   288    678         17  93104220    HARRINGTON   MICHAEL        H
##   text76   164    376         16  93104260        JORDAN   BARBARA        H
##   text77    36     49          4  93104270       KETCHUM   WILLIAM        H
##   text78  2114   9424        429  93104290          KOCH    EDWARD        H
##   text79  1237   4526        199  93104320       LEGGETT    ROBERT        H
##   text80   413   1245         57  93104330         MAHON    GEORGE        H
##   text81   301    915         96  93104350        MCFALL      JOHN        H
##   text82   413   1031         47  93104360         MEEDS     LLOYD        H
##   text83   567   1799         96  93104370      METCALFE     RALPH        H
##   text84   710   2203        125  93104390       MILFORD      DALE        H
##   text85  1082   3093        148  93104400          MOSS      JOHN        H
##   text86   157    298         13  93104510      RONCALIO      TENO        H
##   text87   118    234          6  93104520        ROONEY FREDERICK        H
##   text88   211    398         16  93104540          RYAN       LEO        H
##   text89   194    358         13  93104550       SARASIN    RONALD        H
##   text90  1413   5180        198  93104580         SIKES    ROBERT        H
##   text91   130    251          8  93104590          SISK   BERNICE        H
##   text92   245    595         25  93104600       SKUBITZ       JOE        H
##   text93  1061   3275        165  93104670     WAGGONNER    JOSEPH        H
##   text94   236    461         21  93104680         WALSH   WILLIAM        H
##   text95   658   1721         98  93104690        WHALEN   CHARLES        H
##   text96   866   2882        133  93104700       WIGGINS   CHARLES        H
##   text97   266    579         26  93104720         YOUNG    ANDREW        H
##   text98   852   3294        101  93104790      ANDERSON      JOHN        H
##   text99   204    462         20  93104800        ASHLEY    THOMAS        H
##  text100   336    890         31  93104830        BAUMAN    ROBERT        H
##  state gender party district dem      rest nominate_dim1 nominate_dim2
##     IL      M     R       15   0    Leslie         0.301        -0.187
##     GA      M     R        4   0  Benjamin         0.386         0.214
##     NY      M     D       11   1     Frank        -0.423        -0.151
##     VA      M     R       10   0      Joel         0.159         0.223
##     MI      M     R        6   0   Charles         0.247        -0.191
##     IN      M     R       10   0     David         0.510        -0.241
##     MA      M     D        3   1    Harold        -0.296        -0.098
##     TX      M     D       21   1      Ovie         0.038         0.987
##     NJ      M     R        5   0     Peter         0.172        -0.506
##     WI      M     R        8   0    Harold         0.296        -0.052
##     OR      F     D        3   1     Edith        -0.243         0.236
##     IA      M     R        3   0    Harold         0.955         0.298
##     FL      M     D        5   1   William        -0.121         0.309
##     CA      M     D       34   1   Richard        -0.347         0.043
##     MD      M     R        5   0  Lawrence         0.158        -0.043
##     CA      M     D       19   1   Chester        -0.467         0.218
##     MI      M     R       18   0    Robert         0.459         0.110
##     NJ      M     R        1   0      John         0.272         0.046
##     NY      M     R       29   0  Carleton         0.322         0.017
##     TN      M     R        8   0       Dan         0.246         0.193
##     IN      M     R        2   0      Earl         0.793         0.128
##     NJ      M     R       13   0    Joseph         0.148        -0.200
```

```
## IA M R 6 0 Wiley 0.307 -0.303
## OK M D 2 1 <NA> NA NA
## NY M D 13 1 Bertram -0.469 -0.215
## NY M D 24 1 Ogden -0.394 -0.448
## NY M R 27 0 Howard 0.263 -0.755
## NC M R 8 0 Earl 0.297 0.370
## NJ M R 2 0 Charles 0.139 -0.041
## NY M R 36 0 Henry 0.230 -0.503
## CA M R 43 0 Victor 0.239 -0.020
## CA M D 14 1 Jerome -0.505 -0.211
## PA M R 7 0 Lawrence 0.197 -0.052
## IL M R 10 0 Samuel 0.247 -0.480
## IN M R 8 0 Roger 0.285 0.138
## NH M R 1 0 Louis 0.236 -0.058
## NY F D 20 1 Bella -0.597 -0.802
## OK M D 3 1 Carl -0.392 0.591
## PA M R 8 0 Edward 0.035 -0.746
## IA M D 2 1 John -0.410 -0.259
## NJ M D 14 1 Dominick -0.386 0.100
## TN M D 5 1 Richard -0.332 0.312
## PA M D 3 1 William -0.415 -0.415
## MD M R 8 0 Gilbert -0.028 -0.880
## OH M D 18 1 Wayne -0.304 0.355
## WV M D 4 1 Kenneth -0.310 -0.292
## WA M D 6 1 Floyd -0.414 0.377
## MO M D 9 1 William -0.332 0.433
## MI M R 4 0 J 0.444 -0.036
## IN M D 1 1 Ray -0.422 0.030
## NE M R 2 0 <NA> NA NA
## IA M D 1 1 Edward -0.414 -0.378
## MI M D 12 1 <NA> NA NA
## LA M D 5 1 Otto -0.056 0.952
## TX M D 1 1 John -0.396 0.855
## MO M D 4 1 William -0.184 0.568
## IN M D 4 1 John -0.233 0.158
## TX M R 5 0 Alan 0.255 -0.385
## AZ M R 3 0 Sam 0.447 0.246
## GA M D 10 1 Robert -0.194 0.944
## MO F D 3 1 Leonor -0.386 0.299
## CA M R 12 0 Burt 0.250 -0.115
## NC M D 11 1 Roy -0.069 0.635
## PA M D 24 1 Joseph -0.311 0.216
## DE M R 0 0 <NA> NA NA
## NY M D 21 1 Herman -0.599 -0.617
## MI M R 3 0 Garry 0.302 -0.477
## CA F D 37 1 Yvonne -0.576 -0.177
## MI M R 10 0 Elford 0.270 -0.144
## PA M D 21 1 John -0.384 0.445
## PA M D 4 1 Joshua -0.443 0.084
## AL M D 7 1 Walter -0.147 0.718
## MN M D 5 1 Donald -0.469 -0.466
## FL M R 9 0 Louis 0.244 -0.027
## MA M D 6 1 Michael -0.477 -0.826
## TX F D 18 1 Barbara -0.522 0.253
```

```
##      CA      M     R       36   0   William          0.348          0.246
##      NY      M     D       18   1    Edward         -0.464         -0.657
##      CA      M     D        4   1    Robert         -0.468          0.158
##      TX      M     D       19   1    George         -0.168          0.867
##      CA      M     D       15   1      <NA>             NA             NA
##      WA      M     D        2   1     Edwin         -0.453          0.149
##      IL      M     D        1   1     Ralph         -0.564         -0.144
##      TX      M     D       24   1      Dale          0.012          0.639
##      CA      M     D        3   1      John         -0.582          0.191
##      WY      M     D        0   1      <NA>             NA             NA
##      PA      M     D       15   1 Frederick         -0.350          0.181
##      CA      M     D       11   1       Leo         -0.364          0.001
##      CT      M     R        5   0    Ronald          0.133         -0.591
##      FL      M     D        1   1    Robert         -0.128          0.926
##      CA      M     D       16   1   Bernice         -0.388          0.520
##      KS      M     R        5   0       Joe          0.189          0.080
##      LA      M     D        4   1    Joseph          0.015          1.000
##      NY      M     R       33   0   William          0.068         -0.092
##      OH      M     R        3   0   Charles         -0.139         -0.714
##      CA      M     R       25   0   Charles          0.359         -0.284
##      GA      M     D        5   1    Andrew         -0.589         -0.162
##      IL      M     R       16   0      John          0.183         -0.554
##      OH      M     D        9   1    Thomas         -0.350         -0.049
##      MD      M     R        1   0    Robert          0.533         -0.116
##
## Source: C:/Users/Alec/Documents/Academics/Second Year/Fall Quarter/MACS 40500 - Computational Method
## Created: Tue Dec 10 01:16:16 2019
## Notes:
```

```r
# Create document frequency matrix
dfmat_93 <- dfm(analysis.corp, tolower = TRUE, stem = TRUE, remove_punct = TRUE,
                remove = allstop)

# Trim the matrix to include only terms that occur at least 3 times to ensure convergence
dfmat_93 <- dfm_trim(dfmat_93, min_termfreq = 3, termfreq_type = "count")
```

```r
set.seed(100)
id_train <- sample(1:224, 179, replace=FALSE)

# Create ID variable to subset train/test
docvars(analysis.corp, "id_numeric") <- 1:ndoc(analysis.corp)

# Train set
dfmat_training <- corpus_subset(analysis.corp, id_numeric %in% id_train) %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
  dfm_trim(min_termfreq=3, termfreq_type="count")

# Test set
dfmat_testing <- corpus_subset(analysis.corp, !(id_numeric %in% id_train)) %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
  dfm_trim(min_termfreq=3 , termfreq_type="count")

# Train Naive Bayes classifier
tmod_nb <- textmodel_nb(dfmat_training, docvars(dfmat_training, "dem"))
```

```r
summary(tmod_nb)
```

```
##
## Call:
## textmodel_nb.dfm(x = dfmat_training, y = docvars(dfmat_training,
##     "dem"))
##
## Class Priors:
## (showing first 2 elements)
##   0   1
## 0.5 0.5
##
## Estimated Feature Scores:
##   vote  grant unlimit author   hous judiciari subpena person regard
## 0 0.59 0.4299  0.5929 0.3988 0.5349    0.5708  0.8247 0.5001 0.4417
## 1 0.41 0.5701  0.4071 0.6012 0.4651    0.4292  0.1753 0.4999 0.5583
##   inquiri impeach presid  nixon earlier consider matter motion  made
## 0  0.6348  0.5536 0.4439 0.3904   0.636   0.5638 0.6448 0.4453 0.488
## 1  0.3652  0.4464 0.5561 0.6096   0.364   0.4362 0.3552 0.5547 0.512
##   previous question resolut  order allow  amend  offer straight  parti
## 0    0.513   0.5584  0.4744 0.4049 0.522 0.4504 0.5291    0.636 0.5273
## 1    0.487   0.4416  0.5256 0.5951 0.478 0.5496 0.4709    0.364 0.4727
##     line defeat effect
## 0 0.6077  0.636 0.4087
## 1 0.3923  0.364 0.5913
```

```r
# Make features identical across train and test sets
dfmat_matched <- dfm_match(dfmat_testing, features = featnames(dfmat_training))

# Now to inspect classification
actuals <- docvars(dfmat_matched, "dem")
predictions <- predict(tmod_nb, newdata = dfmat_matched)
cMat <- table(actuals, predictions)
cMat
```

```
##        predictions
## actuals  0  1
##       0 10  9
##       1  7 19
```

```r
caret::confusionMatrix(cMat, mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##        predictions
## actuals  0  1
##       0 10  9
##       1  7 19
##
##                Accuracy : 0.6444
##                  95% CI : (0.4878, 0.7813)
##     No Information Rate : 0.6222
```

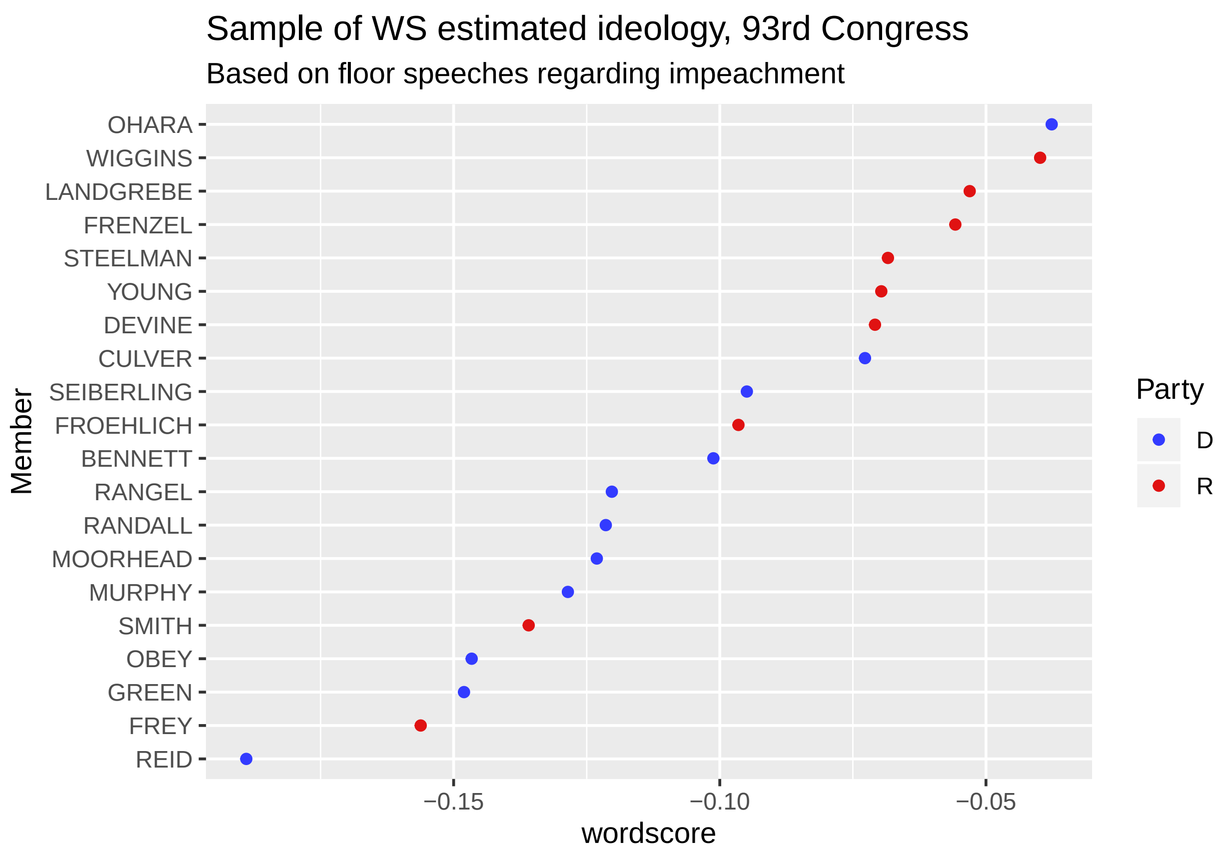
```
##     P-Value [Acc > NIR] : 0.4439
##
##                   Kappa : 0.2608
##
##  Mcnemar's Test P-Value : 0.8026
##
##             Sensitivity : 0.5882
##             Specificity : 0.6786
##          Pos Pred Value : 0.5263
##          Neg Pred Value : 0.7308
##               Precision : 0.5263
##                  Recall : 0.5882
##                      F1 : 0.5556
##              Prevalence : 0.3778
##          Detection Rate : 0.2222
##    Detection Prevalence : 0.4222
##       Balanced Accuracy : 0.6334
##
##        'Positive' Class : 0
##
```

```r
# Define sentiment calculation function
sentScore <- function(text, dictname) {
  corp <- cleanCorpus(VCorpus(VectorSource(text)))
  temp_dtm <- DocumentTermMatrix(corp)
  freq <- sort(colSums(as.matrix(temp_dtm)), decreasing=TRUE)

  tib <- tibble("word" = names(freq), "n" = freq)
  dict_ <- get_sentiments(dictname)

  if (dictname=="bing") {
    sent_calc <- tib %>%
      inner_join(dict_, by="word") %>%
      mutate(ntone = ifelse(sentiment=="positive", n, -n)) %>%
      summarize(total_tone=sum(ntone),
                total_words=sum(n))
  } else if (dictname=="afinn") {
    sent_calc <- tib %>%
      inner_join(dict_, by="word") %>%
      mutate(score=n*value) %>%
      summarize(total_tone=sum(score),
                total_words=sum(n))
  }

  return(sent_calc$total_tone/sent_calc$total_words)
}
```

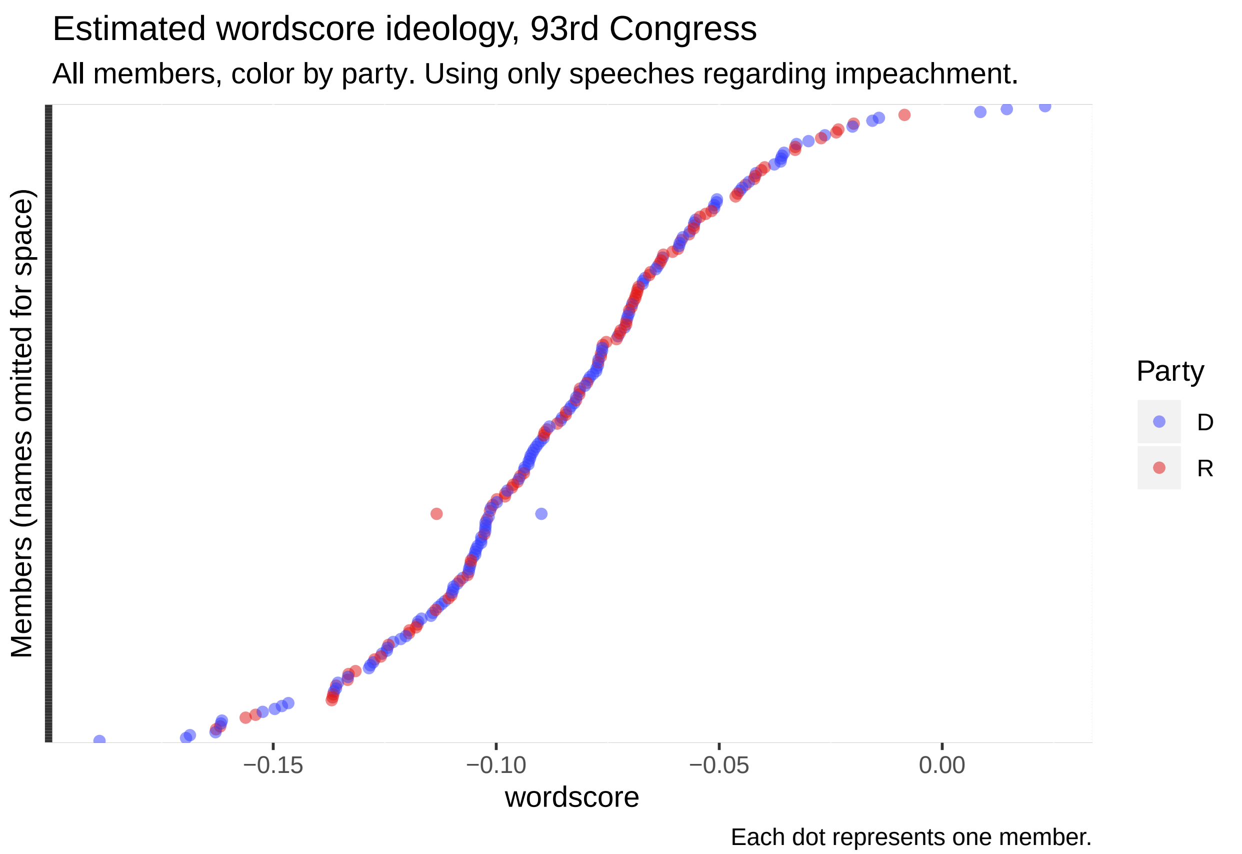
```r
# Attach sentiment scores to members' speeches
bing <- rep(NA, 224)
afinn <- rep(NA, 224)
for (i in 1:224) {
  bing[[i]] <- sentScore(impeach93analysis$text[[i]], "bing")
  afinn[[i]] <- sentScore(impeach93analysis$text[[i]], "afinn")
}
```

```r
# Train wordscores
dfmat_all <- analysis.corp %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
  dfm_trim(min_termfreq=3, termfreq_type="count")

# We'll use John Conyers (-1) and Harold Gross (+1) as anchors for wordscore training
reference.scores <- c(rep(NA, 11), 1, rep(NA, 210), -1, NA)

# Train wordscore model and attach predicted scores to names
ws.model <- textmodel_wordscores(dfmat_all, reference.scores, smooth=1)
ws.full.model <- predict(ws.model, level = 0.95)

# Train wordfish model
wf.full.model <- textmodel_wordfish(dfmat_all, sparse=TRUE)

# Train 2D correspondence analysis
ca <- textmodel_ca(dfmat_all)
ca_dim1 <- coef(ca, doc_dim=1)$coef_document
ca_dim2 <- coef(ca, doc_dim=2)$coef_document

# Create df of wordscores with info from the dfm
wswf.df <- tibble(
  firstname = docvars(dfmat_all, "firstname"),
  lastname = docvars(dfmat_all, "lastname"),
  state = docvars(dfmat_all, "state"),
  district = docvars(dfmat_all, "district"),
  party = docvars(dfmat_all, "party"),
  dem = docvars(dfmat_all, "dem"),
  bing = bing,
  afinn = afinn,
  wordscore = ws.full.model,
  wftheta = wf.full.model$theta,
  wfse = wf.full.model$se,
  ca_dim1 = ca_dim1,
  ca_dim2 = ca_dim2,
  nominate_dim1 = docvars(dfmat_all, "nominate_dim1"),
  nominate_dim2 = docvars(dfmat_all, "nominate_dim2")
)
```

```r
set.seed(800)
#update_geom_defaults("point", list(size=1.5))
ws.plot.1 <- wswf.df[sample(nrow(wswf.df), 20),] %>%
  ggplot(., aes(fct_reorder(as.factor(lastname), wordscore),
                wordscore,
                color=party)) +
  geom_point() +
  coord_flip() +
  scale_color_manual(values=group.colors) +
  labs(x = "Member",
       title = "Sample of WS estimated ideology, 93rd Congress",
       subtitle = "Based on floor speeches regarding impeachment",
       color = "Party")
ws.plot.1
```

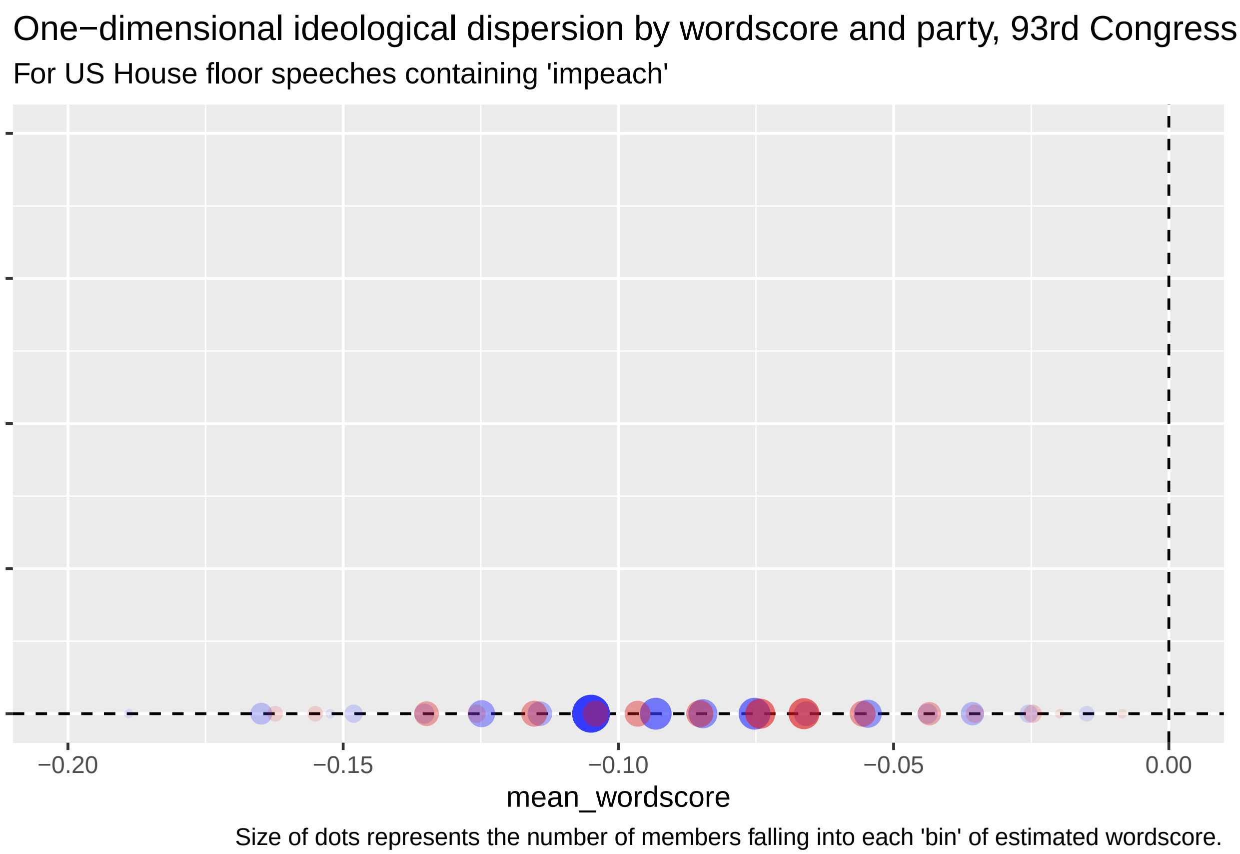



```
ws.plot.2 <- wswf.df %>%
  filter(party != "I" & wordscore < .2 & wordscore > -.2) %>%
  mutate(fullname = paste0(lastname, firstname)) %>%
  ggplot(., aes(fct_reorder(as.factor(fullname), wordscore),
              wordscore, color=party)) +
  geom_point(alpha=0.5) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  theme(axis.text.y=element_blank(),
        panel.background=element_rect(fill="white",
                                      color="lightgray", size=0.5,
                                      linetype="solid"),
        panel.grid.major=element_line(size=0.5, linetype="solid",
                                      color="white"),
        panel.grid.minor=element_line(size=0.25, linetype="solid",
                                      color="white")) +
  labs(title = "Estimated wordscore ideology, 93rd Congress",
       subtitle = "All members, color by party. Using only speeches regarding impeachment.",
       x = "Members (names omitted for space)",
       color = "Party",
       caption = "Each dot represents one member.")
ws.plot.2
```



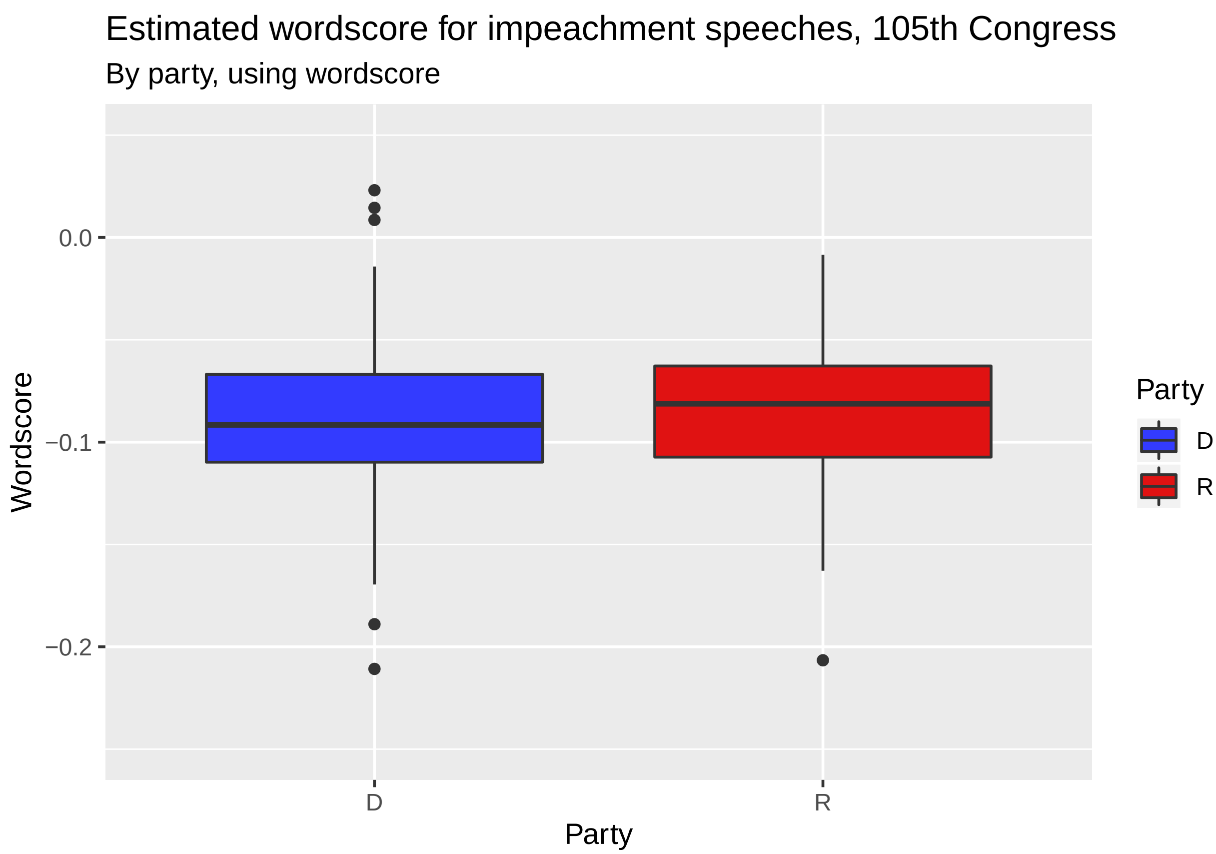

```
ws.plot.3 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = wordscore - (wordscore %% .01)) %>%
  arrange(bin, wordscore) %>%
  group_by(bin, party) %>%
  summarize(mean_wordscore = mean(wordscore, na.rm=TRUE),
            count = n()) %>%
  mutate(x = 0) %>%
  ggplot(., aes(x=c(0), mean_wordscore, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(-.2, 0) +
  xlim(0, .001) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by wordscore and party, 93rd Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated words
       color="Party")
ws.plot.3
```

```
## Warning: Removed 7 rows containing missing values (geom_point).
```

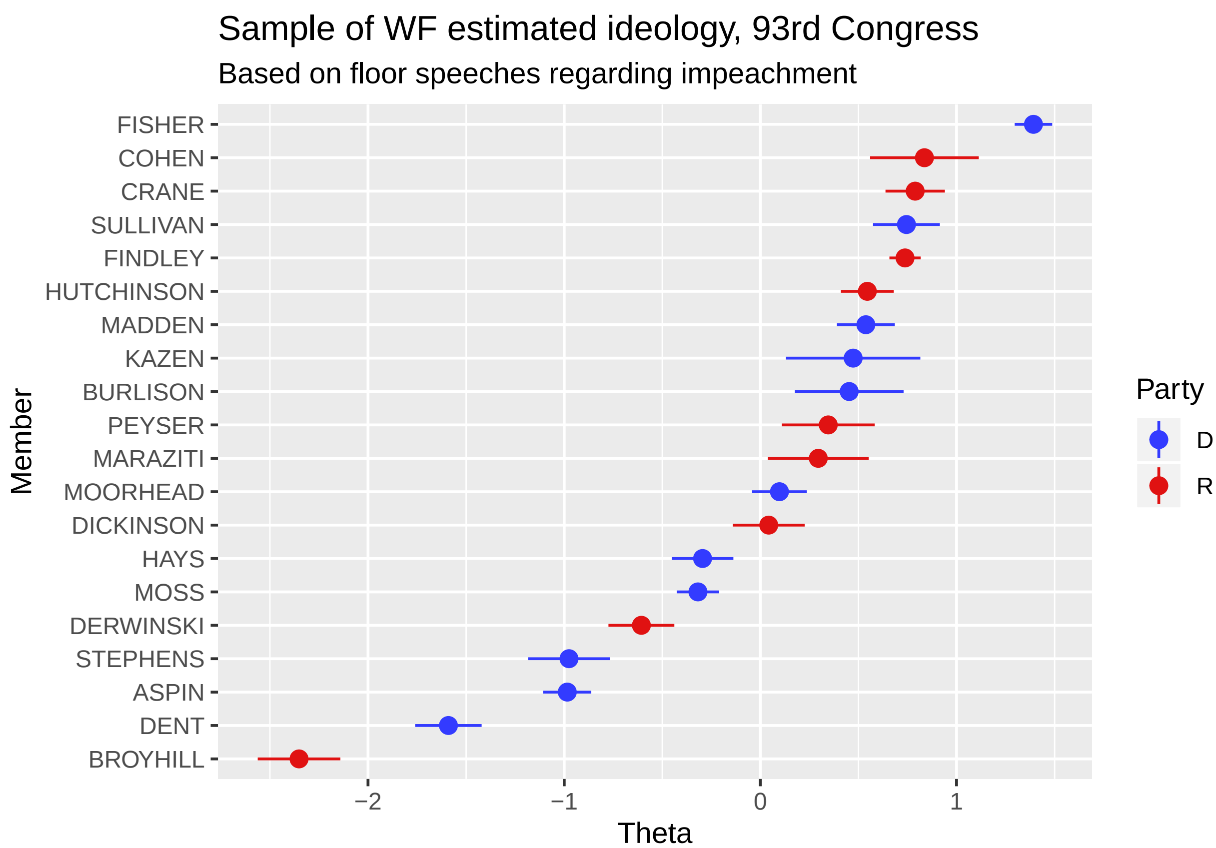



```r
# Produce boxplots of wordscore by party
ws.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=wordscore, fill=party)) +
  scale_fill_manual(values=group.colors) +
  ylim(-.25, .05) +
  geom_boxplot() +
  labs(title="Estimated wordscore for impeachment speeches, 105th Congress",
       subtitle="By party, using wordscore",
       fill="Party",
       x = "Party",
       y = "Wordscore")
ws.plot.4
```

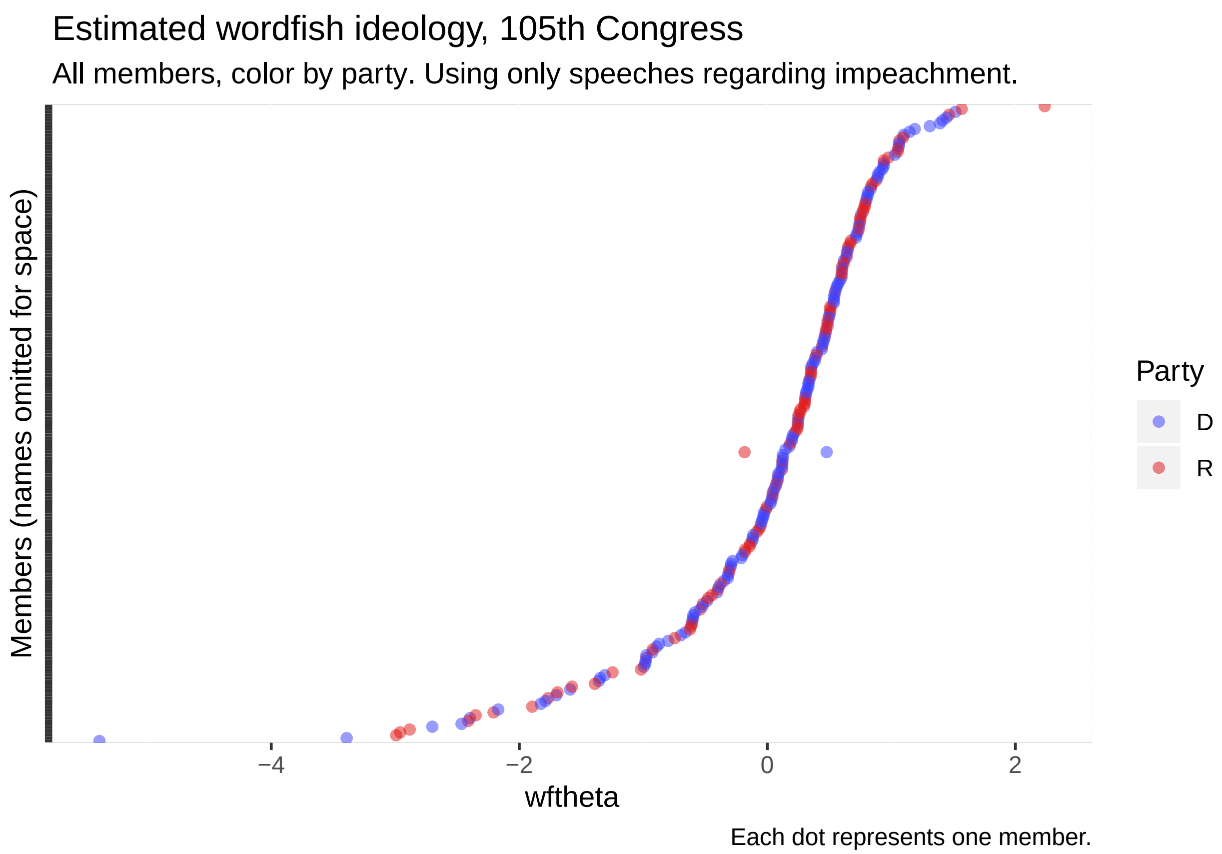
```
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
```




```
set.seed(303)
# Random sample of wordfish scores
wf.plot.1 <- wswf.df[sample(nrow(wswf.df), 20),] %>%
  arrange(wftheta) %>%
  ggplot() +
  geom_pointrange(aes(x=fct_reorder(as.factor(lastname), wftheta),
                      y=wftheta,
                      color=party,
                      ymin=wftheta-2*wfse,
                      ymax=wftheta+2*wfse)) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  labs(x = "Member",
       y = "Theta",
       title = "Sample of WF estimated ideology, 93rd Congress",
       subtitle = "Based on floor speeches regarding impeachment",
       color = "Party")
wf.plot.1
```

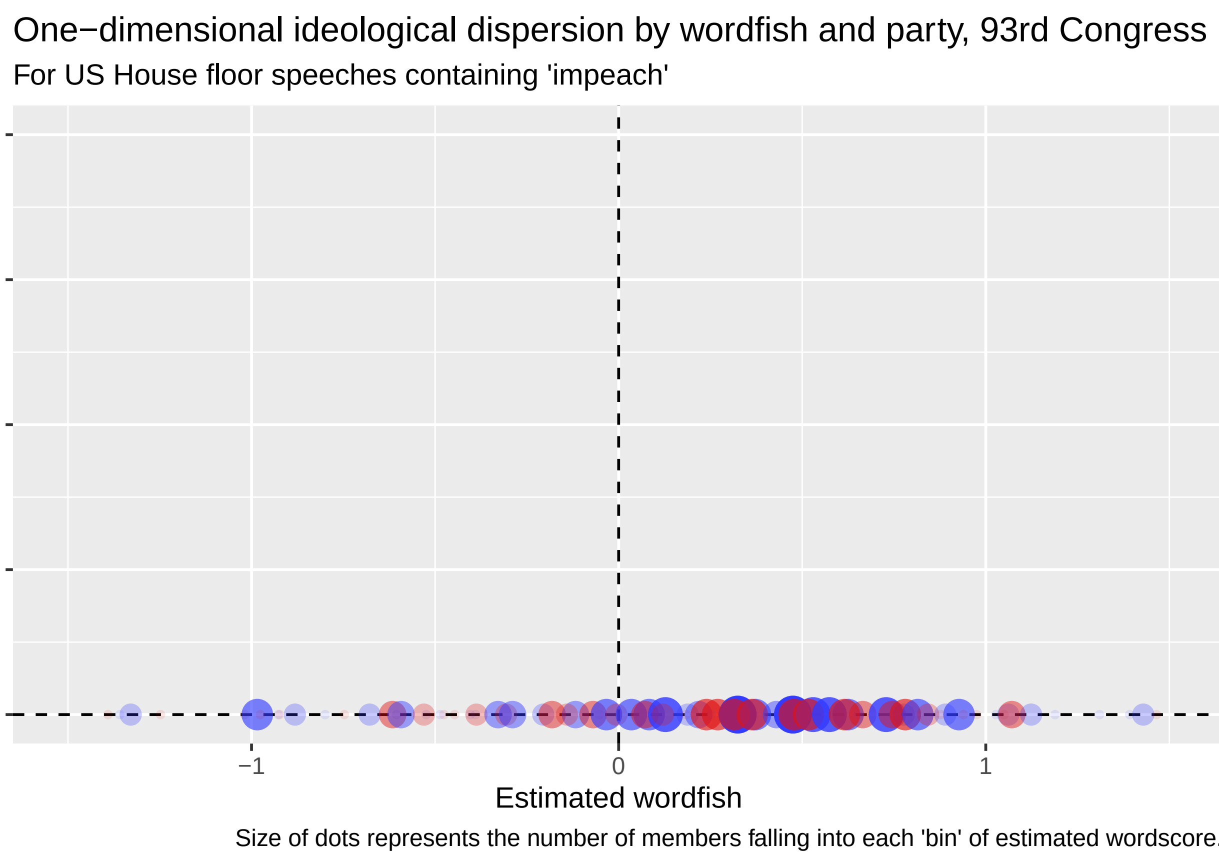



```
# All members' wordfish scores
wf.plot.2 <- wswf.df %>%
  filter(party != "I" & wftheta < 4) %>%
  mutate(fullname = paste0(lastname, firstname)) %>%
  ggplot(., aes(fct_reorder(as.factor(fullname), wftheta),
                wftheta, color=party)) +
  geom_point(alpha=0.5) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  theme(axis.text.y=element_blank(),
        panel.background=element_rect(fill="white",
                                      color="lightgray", size=0.5,
                                      linetype="solid"),
        panel.grid.major=element_line(size=0.5, linetype="solid",
                                      color="white"),
        panel.grid.minor=element_line(size=0.25, linetype="solid",
                                      color="white")) +
  labs(title = "Estimated wordfish ideology, 105th Congress",
       subtitle = "All members, color by party. Using only speeches regarding impeachment.",
       x = "Members (names omitted for space)",
       color = "Party",
       caption = "Each dot represents one member.")
wf.plot.2
```



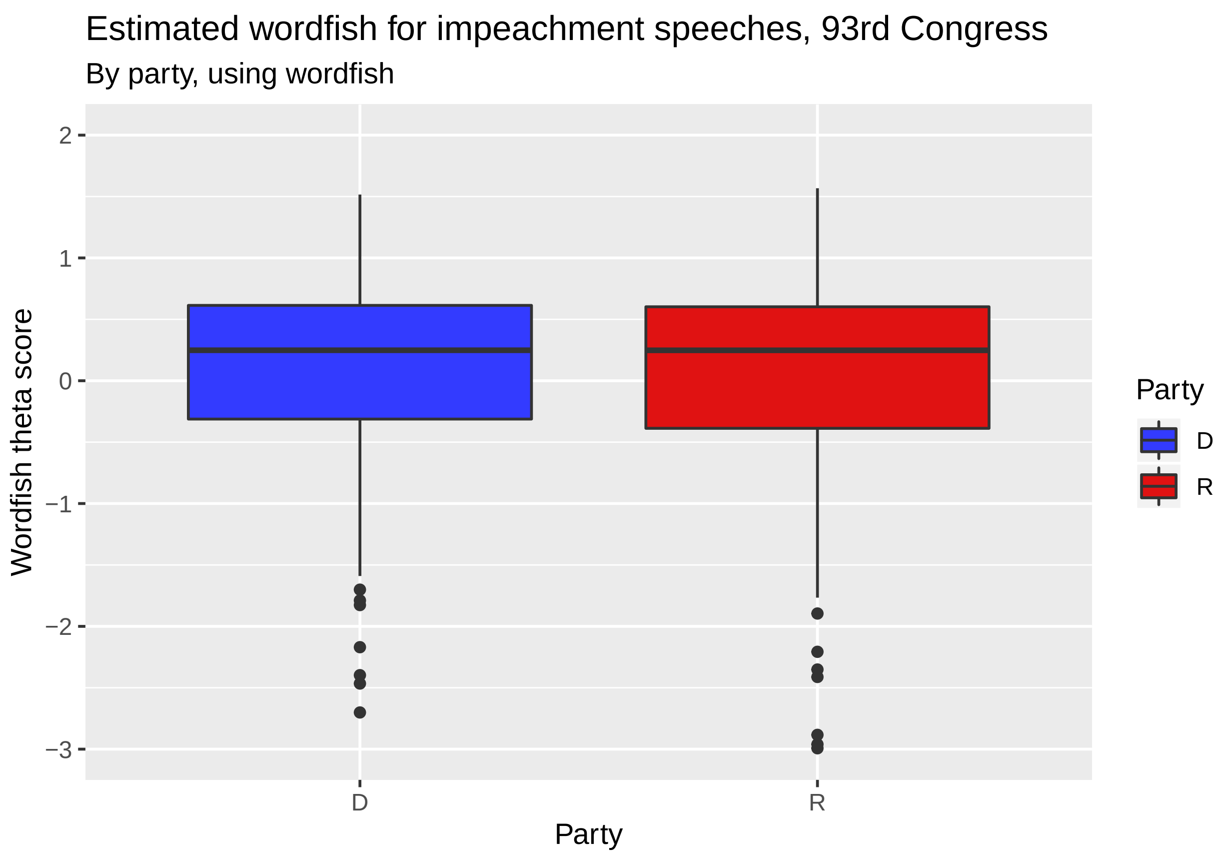

```
#update_geom_defaults("point", list(size=1.5))
wf.plot.3 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = wftheta - (wftheta %% .05)) %>%
  arrange(bin, wftheta) %>%
  group_by(bin, party) %>%
  summarize(mean_wftheta = mean(wftheta, na.rm=TRUE),
            count = n()) %>%
  mutate(x = 0) %>%
  ggplot(., aes(x=c(0), mean_wftheta, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-1.5, 1.5)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by wordfish and party, 93rd Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated words
       color="Party",
       y="Estimated wordfish")
wf.plot.3
```

```
## Warning: Removed 22 rows containing missing values (geom_point).
```

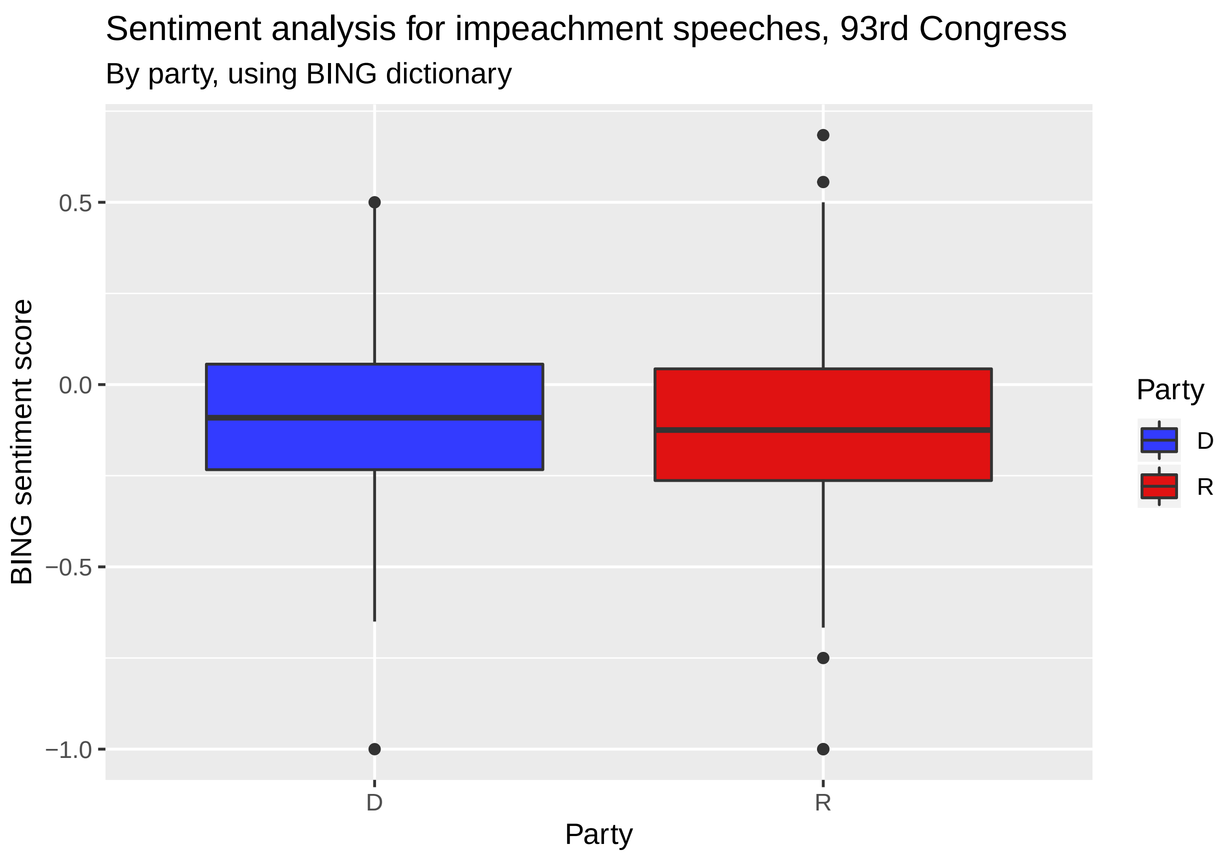



```r
# Produce boxplots of wordfish by party
wf.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=wftheta, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  ylim(-3, 2) +
  labs(title="Estimated wordfish for impeachment speeches, 93rd Congress",
       subtitle="By party, using wordfish",
       fill="Party",
       x = "Party",
       y = "Wordfish theta score")
wf.plot.4
```

```
## Warning: Removed 3 rows containing non-finite values (stat_boxplot).
```

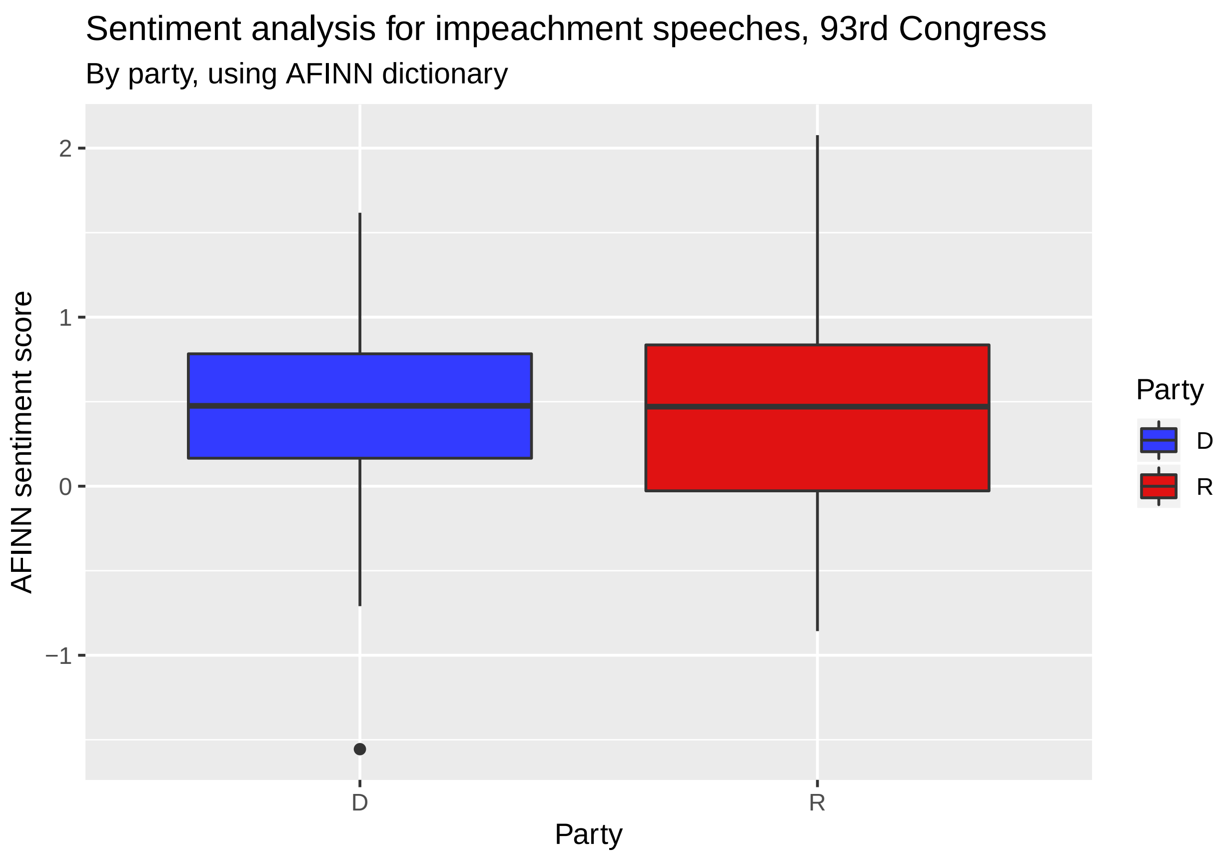



```r
# Produce boxplots of sentiment analysis
sent.plot.1 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=bing, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  labs(title="Sentiment analysis for impeachment speeches, 93rd Congress",
       subtitle="By party, using BING dictionary",
       fill="Party",
       x = "Party",
       y = "BING sentiment score")
sent.plot.1
```



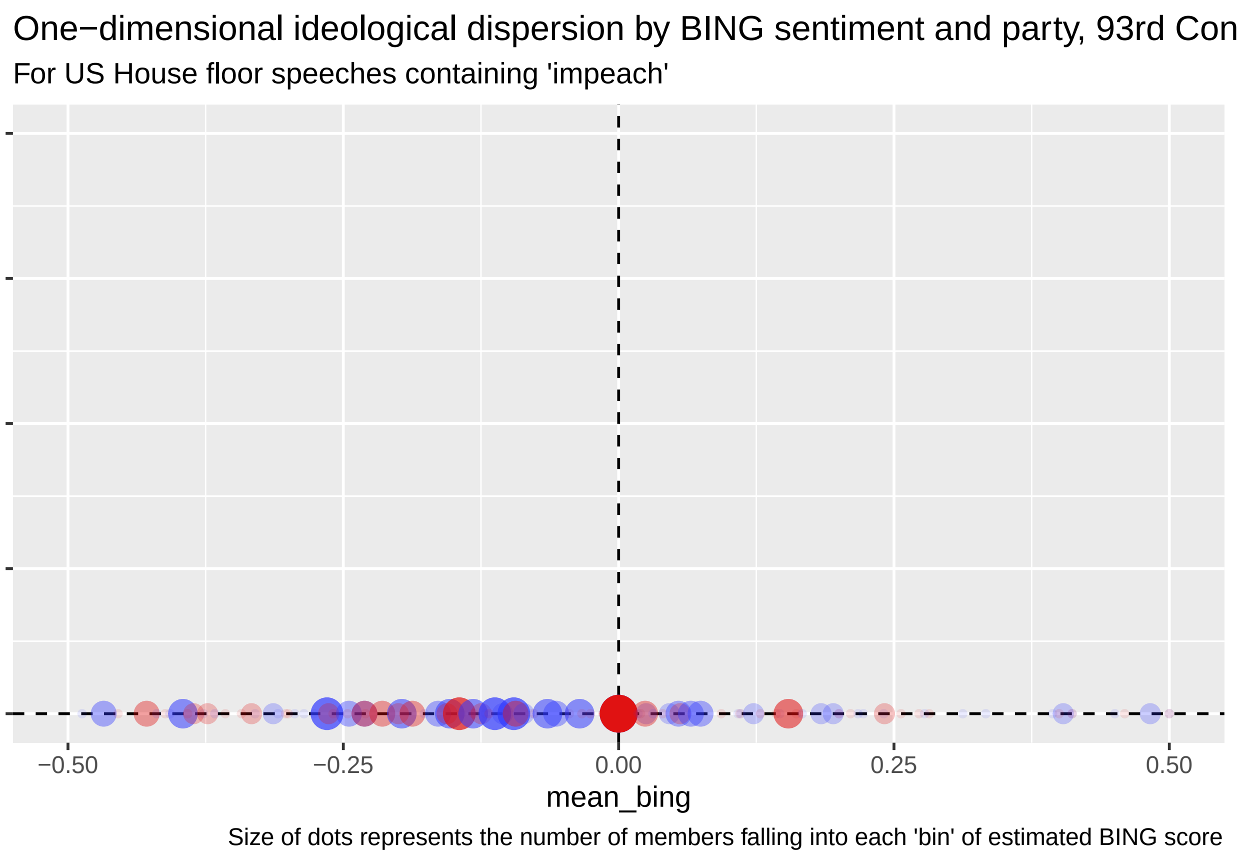

```
sent.plot.2 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=afinn, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  labs(title="Sentiment analysis for impeachment speeches, 93rd Congress",
       subtitle="By party, using AFINN dictionary",
       fill="Party",
       x = "Party",
       y = "AFINN sentiment score")
sent.plot.2
```

```
## Warning: Removed 1 rows containing non-finite values (stat_boxplot).
```



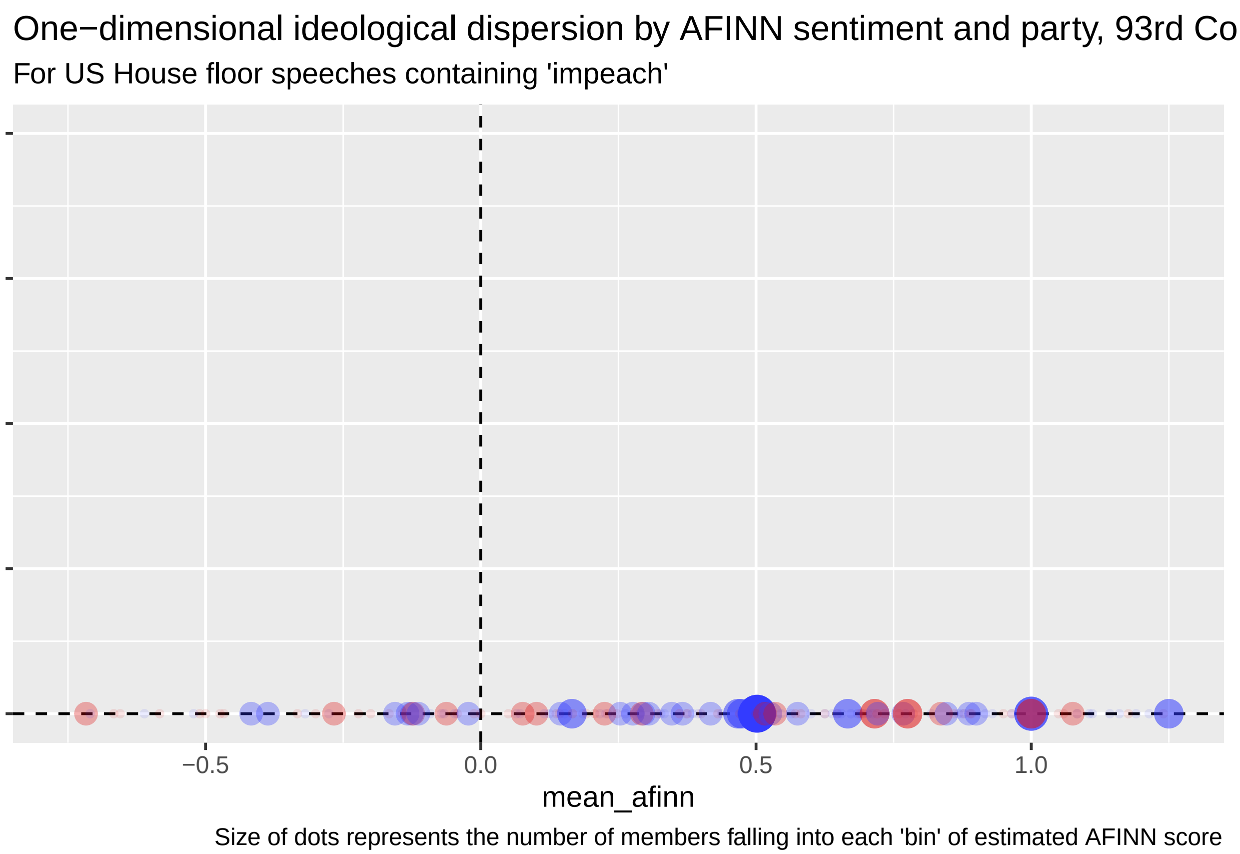

```
# Produce 1D line plots of ideology using sentiment
sent.plot.3 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = bing - (bing %% .01)) %>%
  arrange(bin, bing) %>%
  group_by(bin, party) %>%
  summarize(mean_bing = mean(bing, na.rm=TRUE),
            count=n()) %>%
  ggplot(., aes(x=c(0), mean_bing, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-.5,.5)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by BING sentiment and party, 93rd Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated BING
       color="Party")
sent.plot.3
```

```
## Warning: Removed 12 rows containing missing values (geom_point).
```



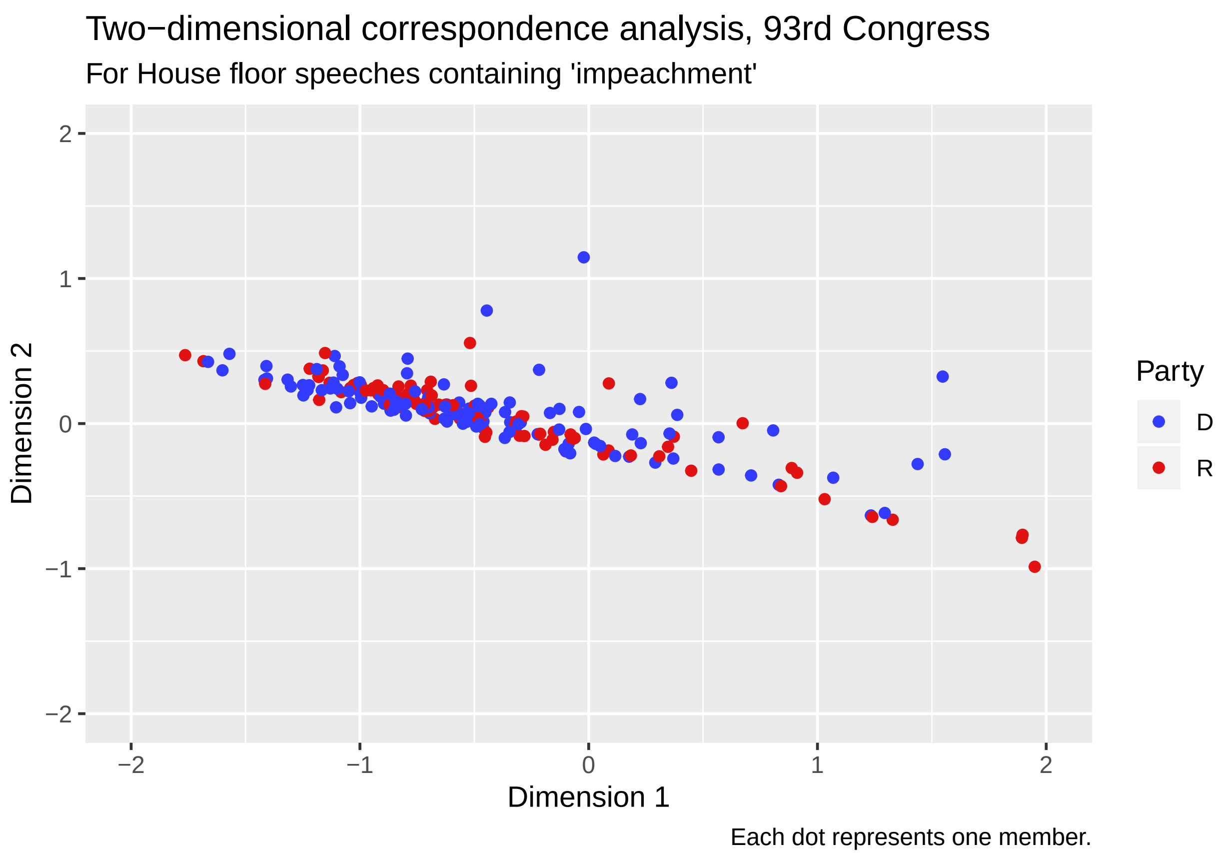

```r
sent.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = afinn - (afinn %% .01)) %>%
  arrange(bin, afinn) %>%
  group_by(bin, party) %>%
  summarize(mean_afinn = mean(afinn, na.rm=TRUE),
            count=n()) %>%
  ggplot(., aes(x=c(0), mean_afinn, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-.75,1.25)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by AFINN sentiment and party, 93rd Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated AFINN
       color="Party")
sent.plot.4
```

```
## Warning: Removed 17 rows containing missing values (geom_point).
```




```r
# Plot 2D correspondence analysis
ca2d <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(ca_dim1, ca_dim2, color=party)) +
  geom_point() +
  xlim(-2, 2) +
  ylim(-2, 2) +
  scale_color_manual(values=group.colors) +
  labs(title="Two-dimensional correspondence analysis, 93rd Congress",
       subtitle="For House floor speeches containing 'impeachment'",
       color="Party",
       x="Dimension 1",
       y="Dimension 2",
       caption="Each dot represents one member.")
ca2d
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```




```
# Use a function to min-max scale our estimated ideology variables
normalize <- function(df, col) {
  min <- min(df[col], na.rm=TRUE)
  max <- max(df[col], na.rm=TRUE)
  newcol <- rep(NA, nrow(df))
  for (i in 1:nrow(df)) {
    if (is.na(df[[col]][[i]])) {
      newcol[[i]] <- NA
    } else {
      newcol[[i]] <- (df[[col]][[i]] - min) / (max-min)
    }
  }
  return(newcol)
}

wswf.df.normalized <- wswf.df %>%
  mutate(ln.bing.n = log(normalize(., "bing")),
         ln.afinn.n = log(normalize(., "afinn")),
         ln.wordscore.n = log(normalize(., "wordscore")),
         ln.wftheta.n = log(normalize(., "wftheta")),
         ln.nd1.n = log(normalize(., "nominate_dim1")))
```

```
# Now do the log of scaled vars regression for percent interpretation
dwn.md1 <- lm(ln.nd1.n ~ ln.bing.n, data = subset(wswf.df.normalized,
                                                   !is.infinite(ln.nd1.n) & !is.infinite(ln.bing.n) &
                                                     !is.na(ln.nd1.n) & !is.na(ln.bing.n)))
```

```r
dwn.md2 <- lm(ln.nd1.n ~ ln.afinn.n, data = subset(wswf.df.normalized,
                                                    !is.infinite(ln.nd1.n) & !is.infinite(ln.afinn.n) &
                                                      !is.na(ln.nd1.n) & !is.na(ln.afinn.n)))

dwn.md3 <- lm(ln.nd1.n ~ ln.wordscore.n, data = subset(wswf.df.normalized,
                                                        !is.infinite(ln.nd1.n) & !is.infinite(ln.wordsc
                                                          !is.na(ln.nd1.n) & !is.na(ln.wordscore.n)))

dwn.md4 <- lm(ln.nd1.n ~ ln.wftheta.n, data = subset(wswf.df.normalized,
                                                      !is.infinite(ln.nd1.n) & !is.infinite(ln.wftheta.
                                                        !is.na(ln.nd1.n) & !is.na(ln.wftheta.n)))

stargazer(dwn.md1, dwn.md2, dwn.md3, dwn.md4, type = "text",
          title = "Regression of log DW-NOMINATE on log ideology estimate")
```

```
##
## Regression of log DW-NOMINATE on log ideology estimate
## ==================================================================================================
##                                                    Dependent variable:
##                       ----------------------------------------------------------------------------
##                                                         ln.nd1.n
##                               (1)                 (2)                   (3)                  (4)
## --------------------------------------------------------------------------------------------------
## ln.bing.n                    0.042
##                             (0.193)
##
## ln.afinn.n                                       -0.101
##                                                  (0.189)
##
## ln.wordscore.n                                                        0.809**
##                                                                       (0.373)
##
## ln.wftheta.n                                                                              -0.356
##                                                                                           (0.290)
##
## Constant                   -1.289***           -1.382***             -0.610*             -1.441***
##                             (0.139)             (0.135)               (0.329)              (0.118)
##
## --------------------------------------------------------------------------------------------------
## Observations                  206                 207                   209                  208
## R2                          0.0002               0.001                 0.022                0.007
## Adjusted R2                 -0.005              -0.003                 0.018                0.002
## Residual Std. Error  0.833 (df = 204)    0.833 (df = 205)     0.822 (df = 207)     0.830 (df = 206)
## F Statistic         0.048 (df = 1; 204) 0.285 (df = 1; 205) 4.709** (df = 1; 207) 1.504 (df = 1; 206)
## ==================================================================================================
## Note:                                                              *p<0.1; **p<0.05; ***p<0.0
```

MACS 40500 Project Analysis

Alec MacMillen

12/2/2019

Part 1: Load data

First, load the pipe-delimited data from the Stanford dataset \texttt{read_delim()}. There are some parsing errors where strings start with quotation marks but don't finish - not sure how to fix that right now, so the strategy is to identify these erroneously parsed rows and drop them (for now).

```
# Speech text
speech105 <- read_delim("Data/crec-105th/speeches_105.txt",
                        delim = "|",
                        col_types = cols(speech_id = col_character(),
                                         speech = col_character()))

# Speaker map: identifying information about speaker
map105 <- read_delim("Data/crec-105th/105_SpeakerMap.txt",
                     delim = "|",
                     col_types = cols(speakerid = col_character(),
                                      speech_id = col_character()))

dropspeech <- problems(speech105) %>%
  distinct(row) %>%
  mutate(speech_id = paste0("105", str_pad(row, 7, "left", "0"))) %>%
  .[[2]]

speech105 <- speech105 %>% filter(!(speech_id %in% dropspeech))
```

Part 2: Filter data to impeachment-related; basic exploratory analysis

The next step is to filter the speeches dataframe to include only speeches that contain "impeach" or "impeachment", then to join that to the speaker map so we know who made what speech. We will also filter to include only members of the House of Representatives, since that is the chamber in which impeachment took place.

```
impeach105h <- speech105 %>%
  filter(grepl("impeach", tolower(speech)) |
         grepl("impeachment", tolower(speech))) %>%
  left_join(map105, by = "speech_id") %>%
  filter(!is.na(speakerid) & chamber == "H")
```

This results in a corpus of 728 documents with 359 distinct speakers. This is a fairly low-dimensional space, considering the high-dimensional data (200,000+ speeches) that we began with. We'll see if we can accurately predict members' ideological ideal points using only this small subset of floor speeches.

Let's do a little bit of EDA on this reduced corpus, including: - Doc count by party - Distribution of docs per speaker

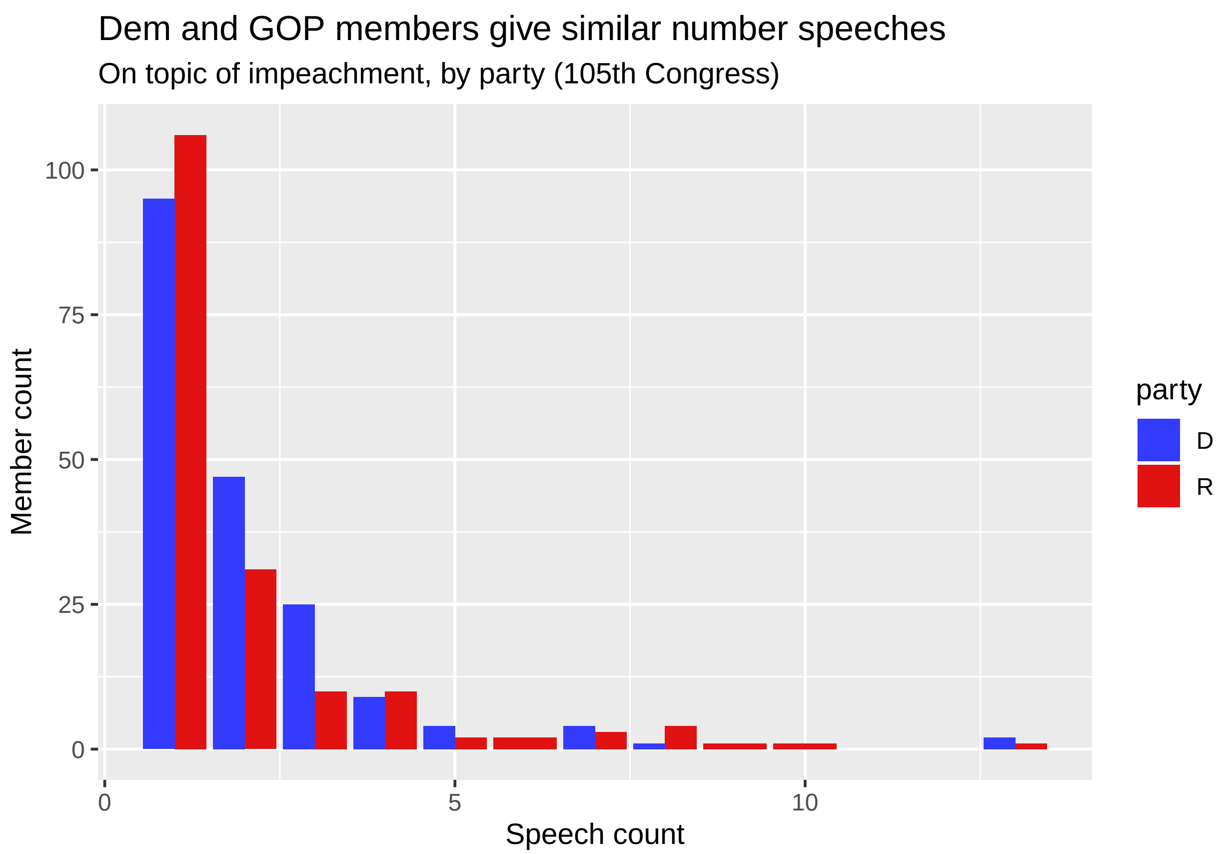
```r
# For faster analysis, drop large "speech" field
impeach105h_eda <- impeach105h %>% select(-speech)

impeach105h_eda %>%
  group_by(speakerid) %>%
  summarize(count = n()) %>%
  skim(count)
```

```
## Skim summary statistics
##  n obs: 359
##  n variables: 2
##
## -- Variable type:integer ------------------------------------------------------------------
##  variable missing complete   n mean   sd p0 p25 p50 p75 p100     hist
##     count       0      359 359 2.03 1.85  1   1   1   2   13 <U+2587><U+2582><U+2581><U+2581><U+2581
```

```r
impeach105h_eda %>%
  group_by(party) %>%
  summarize(count = n()) %>%
  print()
```

```
## # A tibble: 3 x 2
##   party count
##   <chr> <int>
## 1 D       382
## 2 I         1
## 3 R       345
```

```r
# There's only 1 speech by an independent, so we can drop that
# to better visualize relative speech frequency by party
scount_by_party_105 <- impeach105h_eda %>%
  filter(party != "I") %>%
  group_by(speakerid, party) %>%
  summarize(count = n()) %>%
  ungroup() %>%
  arrange(count) %>%
  ggplot(., aes(count, fill=party)) +
  geom_bar(position = "dodge") +
  scale_fill_manual(values=group.colors) +
  labs(title = "Dem and GOP members give similar number speeches",
       subtitle = "On topic of impeachment, by party (105th Congress)",
       y = "Member count",
       x = "Speech count")
scount_by_party_105
```



Most speakers give only 1 or 2 speeches on the topic of impeachment. We have decent variation by party, suggesting that we may be able to discriminate between members of different parties (and maybe between members with different ideal points ideologically) using this corpus.

Let's also do a bit of basic topic modeling using impeachment floor speeches split into two distinct corpuses by party. This will give us insight into whether, in a broad sense, Democrats and Republicans' speech can be parsed into separate topics on the subject of impeachment, which could be one clue that some kind of ideal point estimation is possible. Start by performing pre-processing and converting corpuses into document term matrices.

```
# We're interested in modeling at the speaker/legislator level, so combine speeches by multiple speaker
impeach105hgrp <- impeach105h %>%
  group_by(speakerid, lastname, firstname, chamber, state, gender, party, district) %>%
  summarize(speech = paste0(speech, collapse = " ")) %>%
  ungroup() %>%
  rename(text = speech)

# Define stopwords: use basic English stopwords and Congress-related stopwords
c_stopwords <- c("absent", "adjourn", "ask", "can", "chairman", "committee",
                 "con", "democrat", "etc", "gentleladies", "gentlelady",
                 "gentleman", "gentlemen", "gentlewoman", "gentlewomen",
                 "hereabout", "hereafter", "hereat", "hereby", "herein",
                 "hereinafter", "hereinbefore", "hereinto", "hereof",
                 "hereon", "hereto", "heretofore", "hereunder", "hereunto",
                 "hereupon", "herewith", "month", "mr", "mrs", "nai", "nay",
                 "none", "now", "part", "per", "pro", "republican", "say", "senator",
                 "shall", "sir", "speak", "speaker", "tell", "tempore", "thank", "thereabout",
```

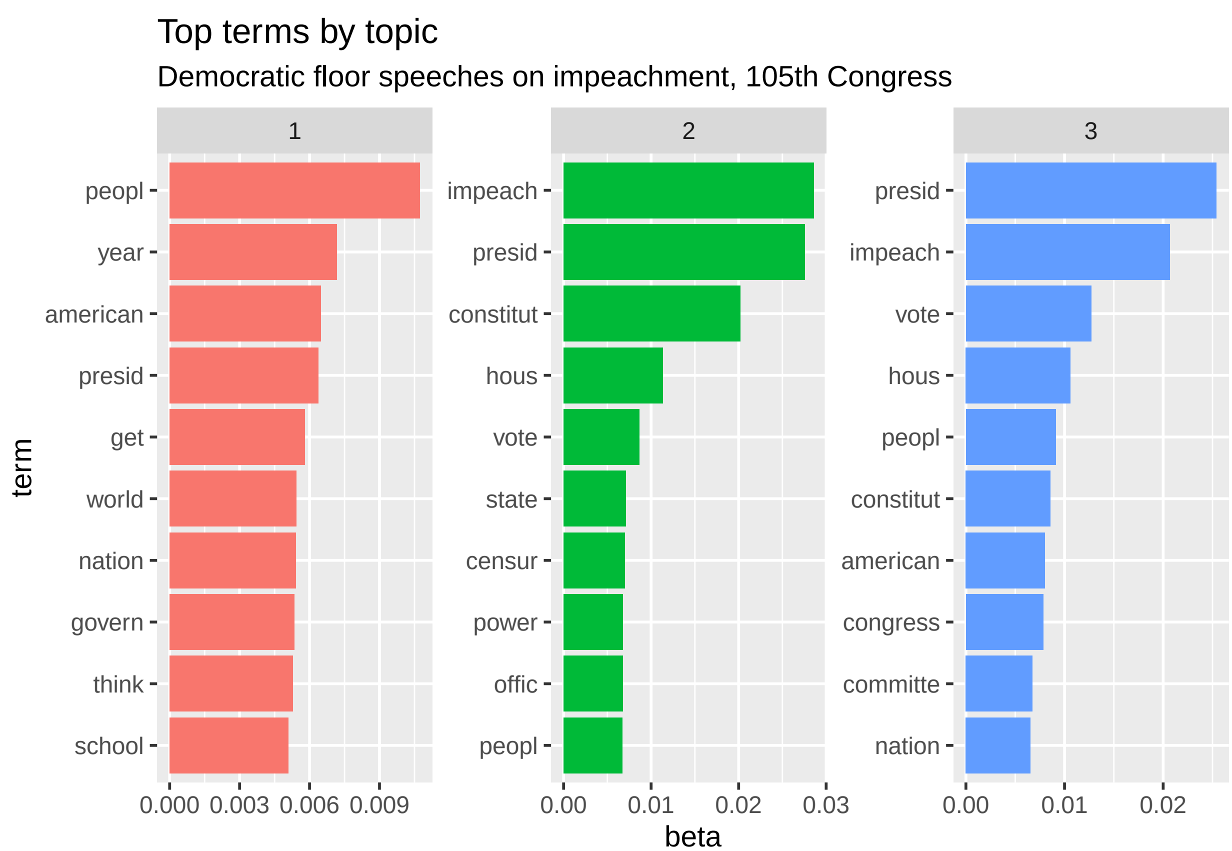
```r
                 "thereafter", "thereagainst", "thereat", "therebefore", "therebeforn",
                 "thereby", "therefore", "therefor", "therefrom", "therein",
                 "thereinafter", "thereof", "thereon", "thereto", "theretofore",
                 "thereunder", "thereunto", "thereupon", "therewith", "therewithal",
                 "today", "whereabouts", "whereafter", "whereas", "whereat",
                 "whereby", "wherefore", "wherefrom", "wherein", "whereinto",
                 "whereo", "whereon", "whereto", "whereunder", "whereupon", "wherever",
                 "wherewith", "wherewithal", "will", "yea", "yes", "yield")

# Full stopwords list is congressional stopwords + base English stopwords
allstop <- c(stopwords("english"), c_stopwords)

# Split overall corpus into one for each party
dem <- impeach105hgrp %>% filter(party == "D")
demC <- VCorpus(VectorSource(dem$text))

rep <- impeach105hgrp %>% filter(party == "R")
repC <- VCorpus(VectorSource(rep$text))

# Function for corpus cleaning in preparation for topic modeling
cleanCorpus <- function(incorpus) {
  ccorp <- tm_map(incorpus, removePunctuation)
  for (j in seq(ccorp)) {
    ccorp[[j]] <- gsub("/", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("â ", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("@", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("/u2028", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("Ã¡", "a", ccorp[[j]])
    ccorp[[j]] <- gsub("â€", " ", ccorp[[j]])
  }
  ccorp <- tm_map(ccorp, removeNumbers)
  ccorp <- tm_map(ccorp, tolower)
  ccorp <- tm_map(ccorp, stemDocument)
  ccorp <- tm_map(ccorp, removeWords, allstop)
  ccorp <- tm_map(ccorp, stripWhitespace)
  ccorp <- tm_map(ccorp, PlainTextDocument)

  return(ccorp)
}

# Create document term matrix for each party's corpus
dem_dtm <- DocumentTermMatrix(cleanCorpus(demC))
rep_dtm <- DocumentTermMatrix(cleanCorpus(repC))
```

Topic models

Now train topic models (with a naive starting k value of 3, which can be revised) and plotting.

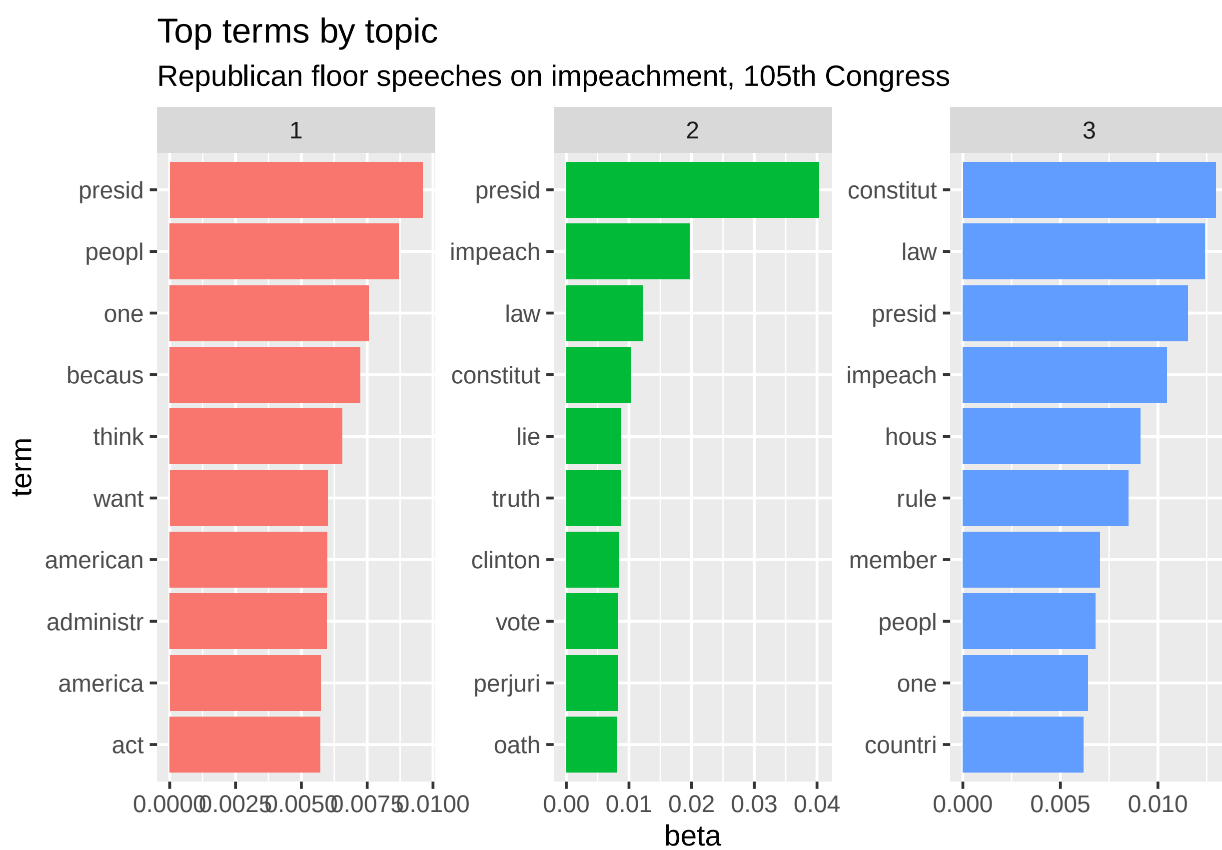
```r
# These take about 80 seconds to train
dem_t3 <- topicmodels::LDA(dem_dtm, k = 3, control = list(seed = 101))
dem_topics <- tidy(dem_t3, matrix = "beta")

dem_top_terms <- dem_topics %>%
```

```r
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, desc(beta))

dem_top_terms_plot <- dem_top_terms %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  coord_flip() +
  scale_x_reordered() +
  labs(title = "Top terms by topic",
       subtitle = "Democratic floor speeches on impeachment, 105th Congress")
dem_top_terms_plot
```




```r
# These take about 80 seconds to run
rep_t3 <- topicmodels::LDA(rep_dtm, k = 3, control = list(seed = 101))
rep_topics <- tidy(rep_t3, matrix = "beta")

rep_top_terms <- rep_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, desc(beta))
```

```r
rep_top_terms_plot <- rep_top_terms %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  coord_flip() +
  scale_x_reordered() +
  labs(title = "Top terms by topic",
       subtitle = "Republican floor speeches on impeachment, 105th Congress")
rep_top_terms_plot
```



There are some similarities between the topics: both Democrats and Republicans regularly rely on broad terms like "American" and "people," as well as "Constitution," to make what one assumes are diametrically opposed arguments on the floor of the House. Interestingly, topic 2 on the Republican side contains words specifically tied to the actions of which Clinton was accused, including "lie", "truth", "perjury", and "oath". These words are nowhere to be found in the Democratic topics, suggesting that Republicans made their case for impeachment by emphasizing the specific wrongdoings Clinton had committed, while Democrats may have avoided discussing the specifics because they felt they didn't have as strong a position on the merits.

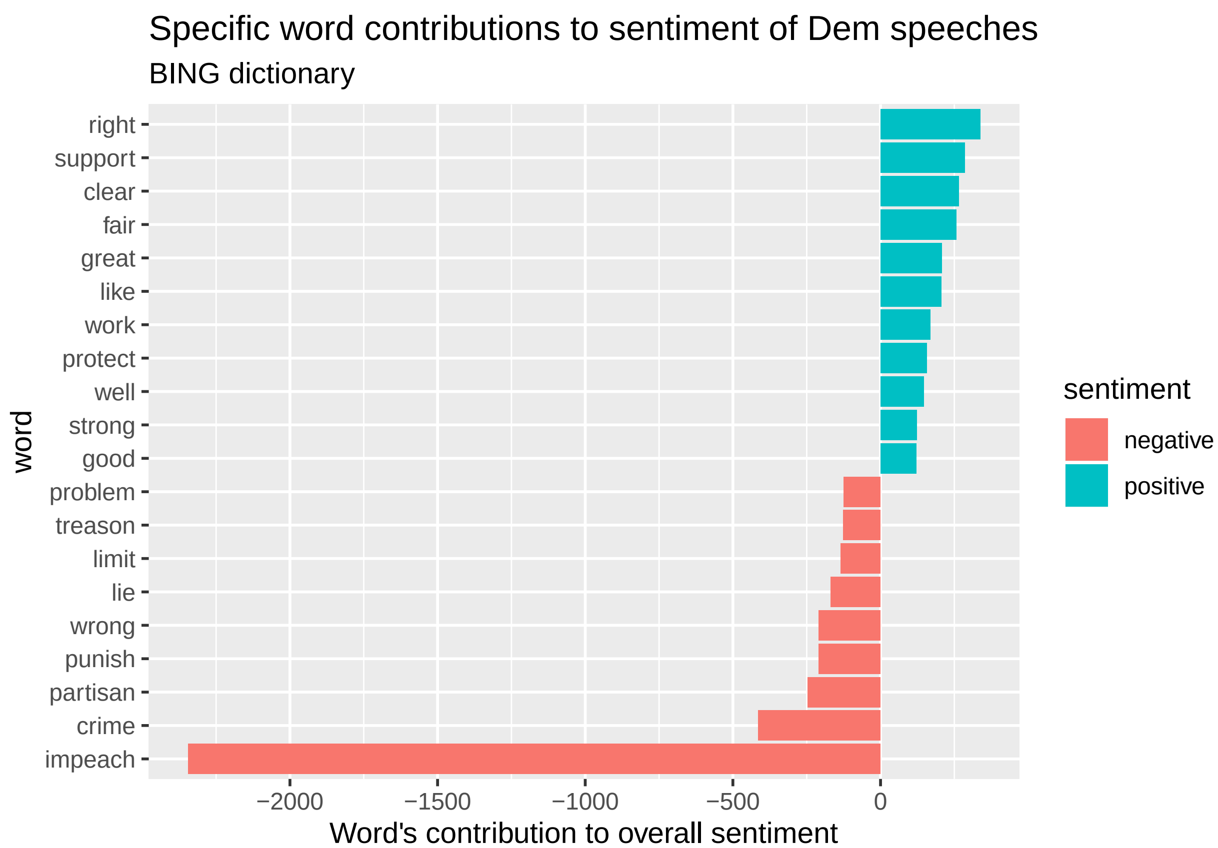
This cursory exploration suggests that in the case of the Clinton impeachment, there were meaningful differences in how Democrats and Republicans spoke about impeachment. This is an indication that we may be able to differentiate between members of the two parties using only their words on impeachment, an inherently non-policy related topic.

Sentiment analysis

```r
freq_dem <- sort(colSums(as.matrix(dem_dtm)), decreasing=TRUE)
freq_dem_t <- tibble("word" = names(freq_dem), "n" = freq_dem)

bing <- get_sentiments("bing")
afinn <- get_sentiments("afinn")


# BING sentiment analysis
dem_bing <- freq_dem_t %>%
  inner_join(bing, by = "word") %>%
  mutate(ntone = ifelse(sentiment == "positive", n, -n)) %>%
  summarize(total_tone = sum(ntone),
            total_words = sum(n))

dem_bing$total_tone / dem_bing$total_words
```

```
## [1] -0.2804309
```

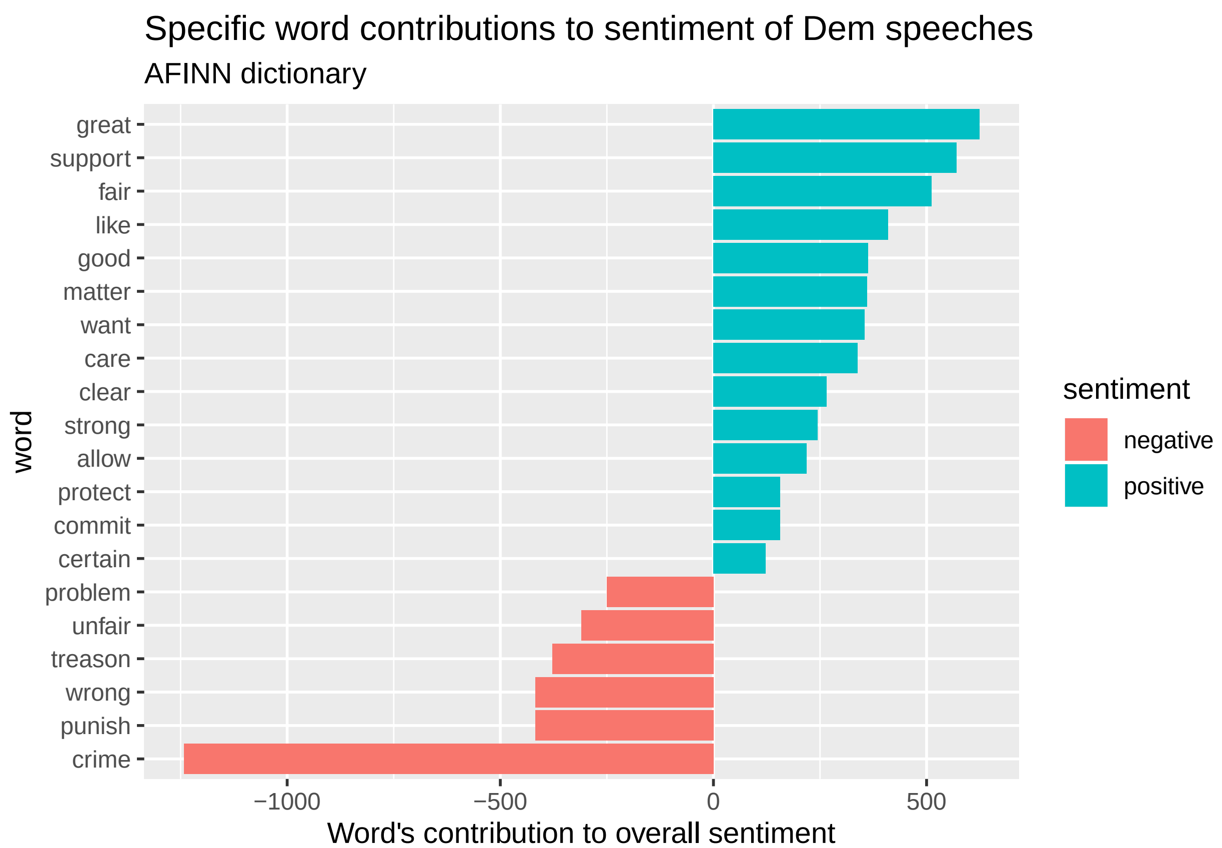
```r
freq_dem_t %>%
  inner_join(bing, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
  mutate(n = ifelse(sentiment == "positive", n, -n),
         word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Dem speeches",
       subtitle = "BING dictionary")
```




```r
# AFINN sentiment analysis
dem_afinn <- freq_dem_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(score = n*value) %>%
  summarize(totscore = sum(score),
            totwords = sum(n))

dem_afinn$totscore / dem_afinn$totwords
```

```
## [1] 0.03028391
```

```r
freq_dem_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
  mutate(score = n*value,
         sentiment = ifelse(score >= 0, "positive", "negative"),
         word = reorder(word, score)) %>%
  ggplot(aes(word, score, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Dem speeches",
       subtitle = "AFINN dictionary")
```

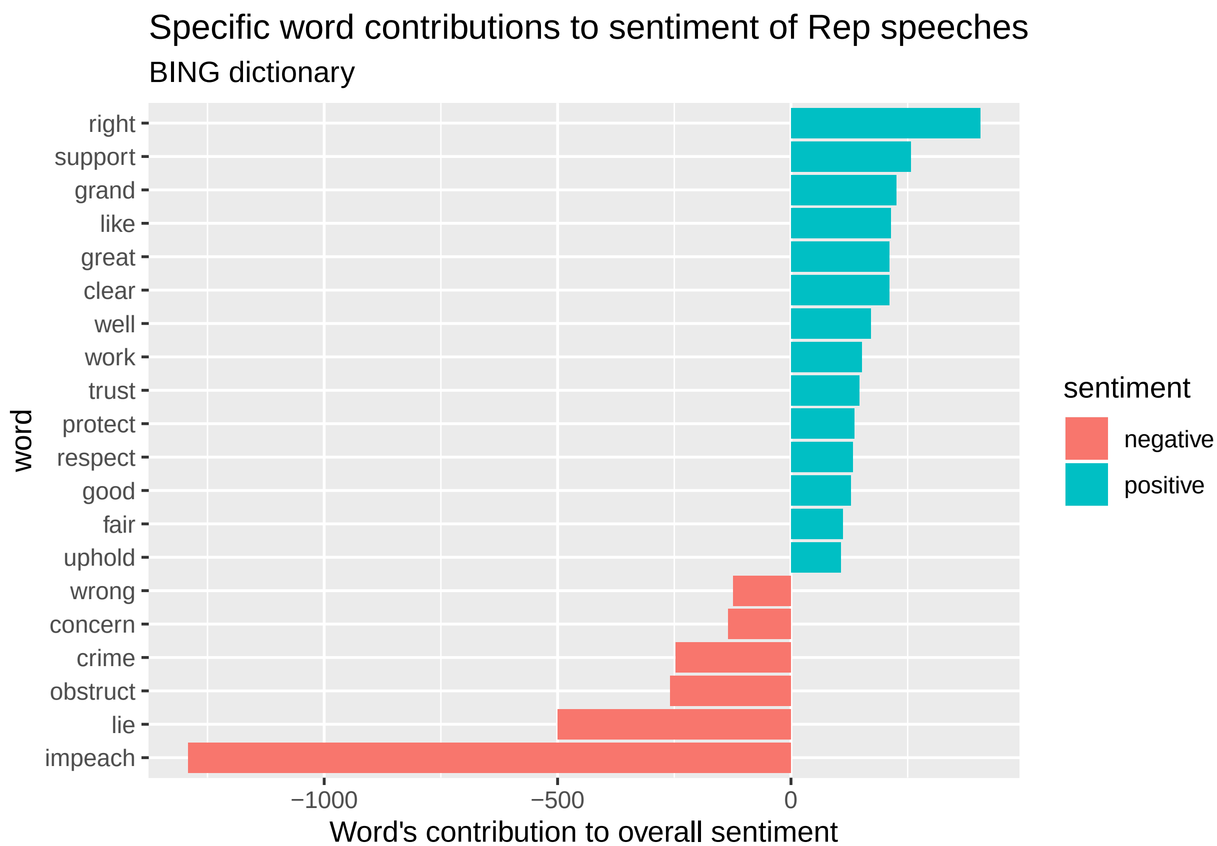



```r
# Repeat sentiment analysis process for Rep
freq_rep <- sort(colSums(as.matrix(rep_dtm)), decreasing=TRUE)
freq_rep_t <- tibble("word" = names(freq_rep), "n" = freq_rep)


# BING sentiment analysis
rep_bing <- freq_rep_t %>%
  inner_join(bing, by = "word") %>%
  mutate(ntone = ifelse(sentiment == "positive", n, -n)) %>%
  summarize(total_tone = sum(ntone),
            total_words = sum(n))

rep_bing$total_tone / rep_bing$total_words
```

```
## [1] -0.1346754
```

```r
freq_rep_t %>%
  inner_join(bing, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
  mutate(n = ifelse(sentiment == "positive", n, -n),
         word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col() +
  coord_flip() +
```

```r
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Rep speeches",
       subtitle = "BING dictionary")
```

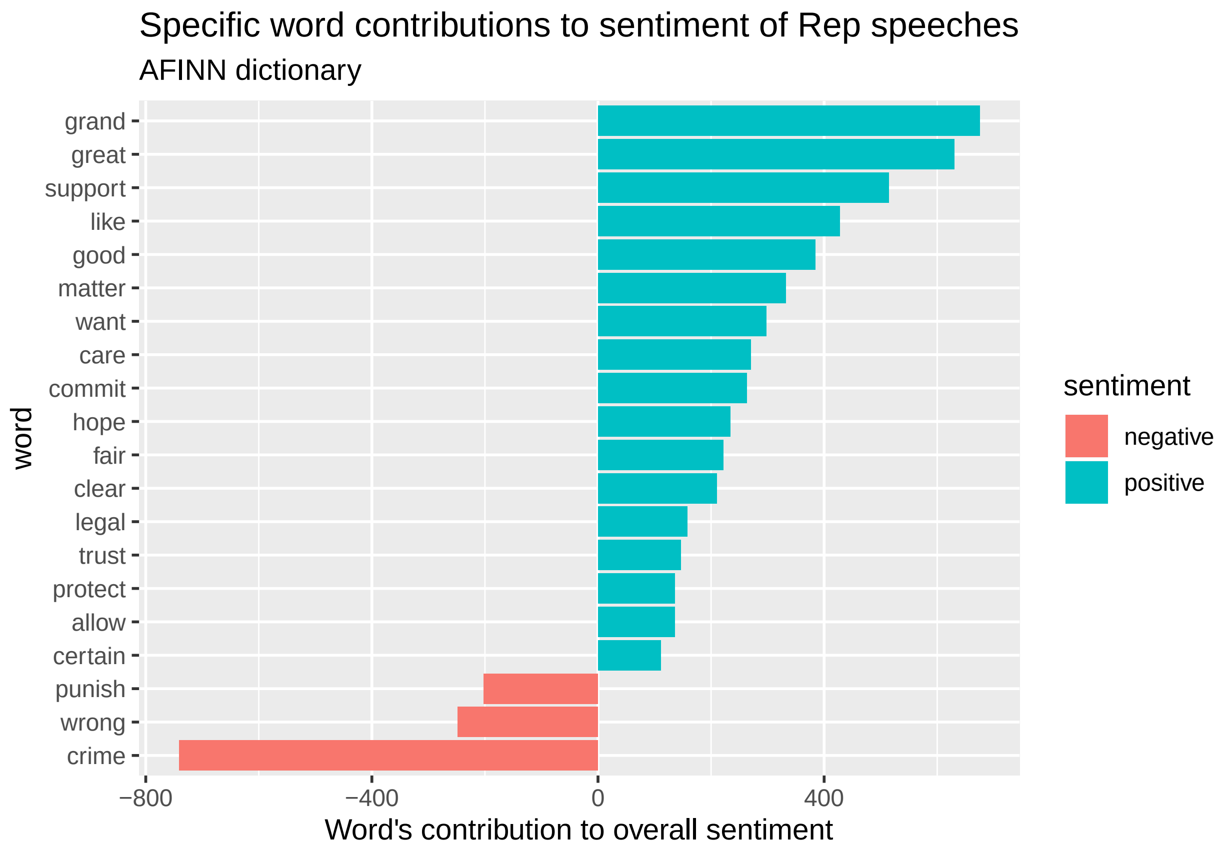



```r
# AFINN sentiment analysis
rep_afinn <- freq_rep_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(score = n*value) %>%
  summarize(totscore = sum(score),
            totwords = sum(n))

rep_afinn$totscore / rep_afinn$totwords
```

```
## [1] 0.3625787
```

```r
freq_rep_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
  mutate(score = n*value,
         sentiment = ifelse(score >= 0, "positive", "negative"),
         word = reorder(word, score)) %>%
  ggplot(aes(word, score, fill = sentiment)) +
  geom_col() +
```

```r
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Rep speeches",
       subtitle = "AFINN dictionary")
```



Part 3: Statistical analysis and prediction

We'll start by creating a document frequency matrix for use with the `quanteda` package from a cleaned corpus of all impeachment-related floor speeches.

```r
# Load and clean DW-NOMINATE
dwn <- read_csv("Data/dw-nominate/Hall_members.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   chamber = col_character(),
##   state_abbrev = col_character(),
##   bioname = col_character(),
##   bioguide_id = col_character(),
##   conditional = col_logical()
## )

## See spec(...) for full column specifications.
```

```r
dwn105 <- dwn %>%
  filter(congress == 105 & chamber == "House") %>%
  select(state_abbrev, district_code, bioname, nominate_dim1, nominate_dim2) %>%
  rename(state = state_abbrev, district = district_code) %>%
  separate(bioname, into = c("lastname", "rest"), by = ",")
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 351 rows [1, 2,
## 3, 4, 5, 8, 9, 10, 12, 13, 14, 15, 16, 17, 19, 20, 21, 23, 24, 25, ...].
```

```r
# Load and clean DIME
dime <- read_csv("Data/dime/dime_cong_elections_current.csv",
                 col_types = cols(gpct = col_double(),
                                  ppct = col_double(),
                                  gwinner = col_character()))

dime105 <- dime %>%
  filter(cycle == 1996 & seat == "federal:house") %>%
  select(Name, state, district, recipient_cfscore) %>%
  mutate(district = as.integer(substr(district, 3, 4))) %>%
  separate(Name, into = c("lastname", "rest"), by = ",") %>%
  select(-rest)
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 1620 rows [1, 5,
## 6, 7, 8, 10, 11, 12, 13, 15, 17, 18, 19, 20, 21, 22, 23, 24, 25, 28, ...].
```

```r
# Join DW-NOMINATE and DIME scores to speeches
impeach105analysis <- impeach105hgrp %>%
  mutate(dem = ifelse(party == "D", 1, 0)) %>%
  left_join(dwn105, by = c("lastname", "state", "district")) %>%
  left_join(dime105, by = c("lastname", "state", "district"))

# Check observations where first names don't match: robustness check for merge
problems <- impeach105analysis %>%
  filter(toupper(rest) != firstname) %>%
  select(-text, -speakerid)

# Note that Mary Bono succeeded her father in the House during the middle of the Congress, and so she s
impeach105analysis <- impeach105analysis %>%
  filter(!(rest == "Mary" & lastname == "BONO")) %>%
  select(-rest)

# Create quanteda corpus object
analysis.corp <- quanteda::corpus(impeach105analysis)
summary(analysis.corp)
```

```
## Corpus consisting of 362 documents, showing 100 documents:
##
##     Text Types Tokens Sentences speakerid       lastname firstname chamber
##    text1   233    512        20 105111730           BONO     SONNY       H
##    text2   138    267        12 105111780         FAWELL    HARRIS       H
##    text3   709   2666       112 105111790          FAZIO    VICTOR       H
##    text4   777   2598       170 105111830          FURSE ELIZABETH       H
```

```
##     text5  216   644          31 105111880         HEFNER    WILLIE        H
##     text6  301   937          39 105111920        KENNEDY    JOSEPH        H
##     text7  223   475          21 105111930       KENNELLY   BARBARA        H
##     text8  295   555          43 105111950           KLUG     SCOTT        H
##     text9  200   398          26 105111960         MANTON    THOMAS        H
##    text10  470  1170          62 105111970         MCDADE    JOSEPH        H
##    text11   58    77           6 105111980         MCHALE      PAUL        H
##    text12 1003  5783         264 105112000        NEUMANN      MARK        H
##    text13  536  1598          88 105112010         PAPPAS   MICHAEL        H
##    text14  127   262          15 105112040        POSHARD     GLENN        H
##    text15   12    15           2 105112050        REDMOND   WILLIAM        H
##    text16 1098  4099         182 105112070          RIGGS     FRANK        H
##    text17 1132  4026         184 105112110         SKAGGS     DAVID        H
##    text18  341   769          26 105112140          SMITH    ROBERT        H
##    text19  425  1193          58 105112150     SNOWBARGER   VINCENT        H
##    text20 1093  5223         222 105112160        SOLOMON    GERALD        H
##    text21  482  1230          68 105112170         STOKES     LOUIS        H
##    text22  112   200          12 105112210         TORRES   ESTEBAN        H
##    text23  267   580          43 105112250          YATES    SIDNEY        H
##    text24  265   701          42 105112270        BARRETT      BILL        H
##    text25  207   560          19 105112280        BATEMAN   HERBERT        H
##    text26  194   423          23 105112310          BROWN    GEORGE        H
##    text27  524  1398          71 105112330       CAMPBELL       TOM        H
##    text28  794  3934         164 105112340         CANADY   CHARLES        H
##    text29  794  3934         164 105112340         CANADY   CHARLES        H
##    text30  259   616          36 105112350 CHENOWETH-HAGE     HELEN        H
##    text31  222   415          24 105112360           CLAY   WILLIAM        H
##    text32  231   542          29 105112370           COOK   MERRILL        H
##    text33  452  1110          57 105112390         DANNER       PAT        H
##    text34  391   971          51 105112410         DICKEY       JAY        H
##    text35  219   458          29 105112420          DIXON    JULIAN        H
##    text36  443  1440          67 105112440          EWING    THOMAS        H
##    text37  282   639          41 105112460         FORBES   MICHAEL        H
##    text38  237   536          33 105112470         FOWLER    TILLIE        H
##    text39  347   969          56 105112480         FRANKS       BOB        H
##    text40 1021  3752         172 105112490      GEJDENSON       SAM        H
##    text41  240   530          29 105112500       GOODLING   WILLIAM        H
##    text42  186   421          17 105112520           HILL      RICK        H
##    text43  559  1628          95 105112550          KLINK       RON        H
##    text44  485  1254          58 105112570          LAZIO      RICK        H
##    text45  826  2829         102 105112580     LIVINGSTON    ROBERT        H
##    text46  966  4881         232 105112600       MCCOLLUM      BILL        H
##    text47  582  1543          62 105112610       MCINTOSH     DAVID        H
##    text48  189   332          22 105112620        METCALF      JACK        H
##    text49  471  1103          66 105112630          MINGE     DAVID        H
##    text50  212   581          35 105112660        PACKARD       RON        H
##    text51  177   378          14 105112670          PEASE    EDWARD        H
##    text52  244   526          19 105112680        PICKETT      OWEN        H
##    text53  227   486          21 105112690         PORTER      JOHN        H
##    text54  568  1741          88 105112700          ROGAN     JAMES        H
##    text55  891  2971         129 105112710          VENTO     BRUCE        H
##    text56  825  2758         135 105112720        WEYGAND    ROBERT        H
##    text57  447  1320          59 105112730           WISE    ROBERT        H
##    text58  726  2778         133 105112760          ARMEY   RICHARD        H
```

```
##    text59   548   2033        113 105112770       BALDACCI      JOHN         H
##    text60   423   1019         53 105112780         BARCIA     JAMES         H
##    text61   876   3180        135 105112790           BARR       BOB         H
##    text62   394   1232         72 105112800        BARRETT    THOMAS         H
##    text63   972   3299        148 105112810        BENTSEN       KEN         H
##    text64   202    539         28 105112820    BLAGOJEVICH       ROD         H
##    text65   481   1340         69 105112830         BONIOR     DAVID         H
##    text66   532   1659         93 105112840         BORSKI    ROBERT         H
##    text67  1032   3754        187 105112850         BRYANT        ED         H
##    text68   168    362         15 105112870       CALLAHAN        H.         H
##    text69   552   2128        137 105112900        CLAYTON       EVA         H
##    text70   318    872         53 105112910        CLEMENT    ROBERT         H
##    text71   294    595         36 105112920         CONDIT      GARY         H
##    text72   287    738         42 105112950          COYNE   WILLIAM         H
##    text73   486   1226         66 105113000         GANSKE      GREG         H
##    text74   126    276          9 105113010          GEKAS    GEORGE         H
##    text75   793   2553        136 105113020         GILMAN  BENJAMIN         H
##    text76   565   1526         87 105113040           HALL      TONY         H
##    text77   190    396         16 105113050         HANSEN     JAMES         H
##    text78   417   1333         71 105113060       HILLEARY       VAN         H
##    text79   252    560         28 105113070       HILLIARD      EARL         H
##    text80   163    335         23 105113080           HORN   STEPHEN         H
##    text81   864   3488        179 105113090     HUTCHINSON       ASA         H
##    text82   267    662         31 105113120        LAFALCE      JOHN         H
##    text83   209    455         25 105113140         LUTHER   WILLIAM         H
##    text84   347    832         34 105113150        MALONEY     JAMES         H
##    text85   191    555         35 105113160        MASCARA     FRANK         H
##    text86   216    557         35 105113170           MEEK    CARRIE         H
##    text87   399   1125         61 105113180         MILLER       DAN         H
##    text88   487   1501         86 105113190           MINK     PATSY         H
##    text89   784   2857        166 105113200        MOAKLEY      JOHN         H
##    text90   390   1213         53 105113210        MORELLA CONSTANCE         H
##    text91   194    476         28 105113230          RILEY       BOB         H
##    text92   287    628         24 105113240         RIVERS      LYNN         H
##    text93   185    381         19 105113250         ROEMER   TIMOTHY         H
##    text94   320    942         55 105113260        ROUKEMA     MARGE         H
##    text95   228    510         29 105113290         SAWYER    THOMAS         H
##    text96   959   4112        236 105113300    SCARBOROUGH       JOE         H
##    text97  1876   9971        517 105113310       SCHAFFER       BOB         H
##    text98   343    772         47 105113340        SISISKY    NORMAN         H
##    text99   281   1861         99 105113360         SPENCE     FLOYD         H
##   text100    72    115          7 105113370          STUMP    ROBERT         H
##   state gender party district dem nominate_dim1 nominate_dim2
##      CA      M     R       44   0         0.375         0.001
##      IL      M     R       13   0         0.332        -0.616
##      CA      M     D        3   1        -0.455         0.360
##      OR      F     D        1   1        -0.457        -0.438
##      NC      M     D        8   1        -0.264         0.609
##      MA      M     D        8   1        -0.413        -0.453
##      CT      F     D        1   1        -0.346        -0.130
##      WI      M     R        2   0         0.258        -0.713
##      NY      M     D        7   1        -0.361         0.382
##      PA      M     R       10   0            NA            NA
##      PA      M     D       15   1            NA            NA
```

```
##      WI      M      R      1   0        0.667       -0.431
##      NJ      M      R     12   0        0.361       -0.319
##      IL      M      D     19   1       -0.211        0.392
##      NM      M      R      3   0        0.300        0.317
##      CA      M      R      1   0        0.333       -0.280
##      CO      M      D      2   1       -0.355       -0.187
##      OR      M      R      2   0        0.396       -0.203
##      KS      M      R      3   0        0.522        0.165
##      NY      M      R     22   0        0.492       -0.333
##      OH      M      D     11   1       -0.578       -0.312
##      CA      M      D     34   1       -0.487        0.012
##      IL      M      D      9   1       -0.494       -0.494
##      NE      M      R      3   0        0.379       -0.073
##      VA      M      R      1   0        0.242        0.191
##      CA      M      D     42   1       -0.506       -0.028
##      CA      M      R     15   0        0.256       -0.907
##      FL      M      R     12   0        0.382        0.031
##      FL      M      R     12   0        0.382        0.031
##      ID      F      R      1   0           NA           NA
##      MO      M      D      1   1       -0.490       -0.872
##      UT      M      R      2   0        0.364       -0.059
##      MO      F      D      6   1       -0.186        0.529
##      AR      M      R      4   0        0.391        0.220
##      CA      M      D     32   1       -0.458       -0.076
##      IL      M      R     15   0        0.361       -0.094
##      NY      M      R      1   0        0.109       -0.112
##      FL      F      R      4   0        0.329       -0.209
##      NJ      M      R      7   0        0.259       -0.836
##      CT      M      D      2   1       -0.416       -0.486
##      PA      M      R     19   0        0.336       -0.494
##      MT      M      R      0   0           NA           NA
##      PA      M      D      4   1       -0.310        0.526
##      NY      M      R      2   0        0.216       -0.588
##      LA      M      R      1   0        0.326        0.109
##      FL      M      R      8   0           NA           NA
##      IN      M      R      2   0           NA           NA
##      WA      M      R      2   0        0.384       -0.169
##      MN      M      D      2   1       -0.213       -0.131
##      CA      M      R     48   0        0.419        0.152
##      IN      M      R      7   0        0.413       -0.129
##      VA      M      D      2   1       -0.149        0.555
##      IL      M      R     10   0        0.217       -0.604
##      CA      M      R     27   0        0.475       -0.088
##      MN      M      D      4   1       -0.472       -0.376
##      RI      M      D      2   1       -0.330        0.183
##      WV      M      D      2   1       -0.320        0.253
##      TX      M      R     26   0        0.635       -0.089
##      ME      M      D      2   1       -0.333       -0.001
##      MI      M      D      5   1       -0.184        0.547
##      GA      M      R      7   0        0.632        0.218
##      WI      M      D      5   1       -0.380       -0.507
##      TX      M      D     25   1       -0.292        0.116
##      IL      M      D      5   1       -0.317       -0.188
##      MI      M      D     10   1       -0.547       -0.006
```

```
##     PA      M     D        3   1        -0.416          0.406
##     TN      M     R        7   0         0.442          0.250
##     AL      M     R        1   0         0.373          0.202
##     NC      F     D        1   1        -0.459         -0.032
##     TN      M     D        5   1        -0.216          0.327
##     CA      M     D       18   1        -0.103          0.300
##     PA      M     D       14   1        -0.494         -0.119
##     IA      M     R        4   0         0.239         -0.300
##     PA      M     R       17   0         0.426         -0.295
##     NY      M     R       20   0         0.043         -0.425
##     OH      M     D        3   1        -0.280          0.155
##     UT      M     R        1   0         0.496          0.143
##     TN      M     R        4   0         0.543          0.449
##     AL      M     D        7   1        -0.555          0.569
##     CA      M     R       38   0         0.167         -0.651
##     AR      M     R        3   0         0.358          0.133
##     NY      M     D       29   1            NA             NA
##     MN      M     D        6   1        -0.306         -0.381
##     CT      M     D        5   1        -0.244         -0.053
##     PA      M     D       20   1        -0.271          0.467
##     FL      F     D       17   1        -0.483          0.120
##     FL      M     R       13   0         0.453         -0.494
##     HI      F     D        2   1        -0.513         -0.067
##     MA      M     D        9   1        -0.418         -0.054
##     MD      F     R        8   0        -0.018         -0.893
##     AL      M     R        3   0         0.433          0.692
##     MI      F     D       13   1        -0.383         -0.453
##     IN      M     D        3   1        -0.165          0.229
##     NJ      F     R        5   0         0.162         -0.701
##     OH      M     D       14   1        -0.377         -0.114
##     FL      M     R        1   0         0.672         -0.433
##     CO      M     R        4   0         0.638          0.770
##     VA      M     D        4   1        -0.122          0.453
##     SC      M     R        2   0         0.320          0.185
##     AZ      M     R        3   0         0.703          0.203
##  recipient_cfscore
##                 NA
##              0.655
##             -0.485
##             -1.028
##             -0.238
##                 NA
##             -0.685
##              1.008
##             -0.513
##              0.039
##             -0.603
##                 NA
##                 NA
##             -0.597
##              1.123
##              0.805
##             -0.917
##                 NA
```

```
##                 1.202
##                 0.219
##                -0.522
##                -0.502
##                -0.652
##                    NA
##                 0.632
##                -0.766
##                    NA
##                -0.697
##                 0.897
##                    NA
##                -0.677
##                 0.859
##                -0.359
##                 0.867
##                    NA
##                 0.770
##                -0.098
##                 0.803
##                 0.418
##                -0.719
##                 0.703
##                    NA
##                -0.627
##                 0.720
##                    NA
##                    NA
##                 1.192
##                 0.853
##                -0.711
##                    NA
##                 0.950
##                -0.277
##                 0.418
##                 1.044
##                -0.676
##                    NA
##                -0.367
##                 1.144
##                -0.942
##                -0.253
##                    NA
##                -1.020
##                -0.444
##                -0.893
##                -0.664
##                -0.490
##                    NA
##                 0.814
##                -0.828
##                -0.208
##                -0.137
##                -0.461
```

```
##              0.986
##              0.824
##                 NA
##             -0.516
##              0.844
##              1.164
##             -0.435
##                 NA
##                 NA
##                 NA
##                 NA
##             -0.811
##             -0.251
##             -0.822
##              1.004
##             -0.877
##             -0.647
##              0.264
##              0.962
##             -1.184
##                 NA
##              0.294
##             -0.555
##              0.892
##              1.238
##             -0.202
##              0.661
##                 NA
##
## Source: C:/Users/Alec/Documents/Academics/Second Year/Fall Quarter/MACS 40500 - Computational Methods
## Created: Tue Dec 10 01:07:21 2019
## Notes:
```

```r
# Create document frequency matrix
dfmat_105 <- dfm(analysis.corp, tolower = TRUE, stem = TRUE, remove_punct = TRUE,
                 remove = allstop)

# Trim the matrix to include only terms that occur at least 3 times to ensure convergence
dfmat_105 <- dfm_trim(dfmat_105, min_termfreq = 3, termfreq_type = "count")
```

Now we're ready to start our analysis! We'll first do a simple naive Bayes classifier that predicts the binary class of Democrat vs. Republican, as the first test of a prior that we can guess whether a member is of a given party based on how they speak about impeachment.

```r
set.seed(100)
id_train <- sample(1:362, 290, replace=FALSE)

# Create ID variable to subset train/test
docvars(analysis.corp, "id_numeric") <- 1:ndoc(analysis.corp)

# Train set
dfmat_training <- corpus_subset(analysis.corp, id_numeric %in% id_train) %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
```

```r
  dfm_trim(min_termfreq=3, termfreq_type="count")

# Test set
dfmat_testing <- corpus_subset(analysis.corp, !(id_numeric %in% id_train)) %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
  dfm_trim(min_termfreq=3 , termfreq_type="count")

# Train Naive Bayes classifier
tmod_nb <- textmodel_nb(dfmat_training, docvars(dfmat_training, "dem"))
summary(tmod_nb)
```

```
##
## Call:
## textmodel_nb.dfm(x = dfmat_training, y = docvars(dfmat_training,
##     "dem"))
##
## Class Priors:
## (showing first 2 elements)
##   0   1
## 0.5 0.5
##
## Estimated Feature Scores:
##    yield  time    may   well   myth  georg washington confess cherri
## 0 0.6324 0.529 0.5462 0.5813 0.4435 0.5045     0.7999  0.8416 0.8416
## 1 0.3676 0.471 0.4538 0.4187 0.5565 0.4955     0.2001  0.1584 0.1584
##      tree    lie   know abraham lincoln  young   man actual   walk sever
## 0 0.92118 0.7607 0.5656  0.7798  0.3826 0.6182 0.722 0.6859 0.5151 0.532
## 1 0.07882 0.2393 0.4344  0.2202  0.6174 0.3818 0.278 0.3141 0.4849 0.468
##     mile return   gave incorrect  chang   true  stori   truth justic
## 0 0.5151 0.3826 0.6871    0.4146 0.5499 0.5258 0.6522 0.91433 0.7251
## 1 0.4849 0.6174 0.3129    0.5854 0.4501 0.4742 0.3478 0.08567 0.2749
##     hold special
## 0 0.7283  0.5445
## 1 0.2717  0.4555
```

```r
# Make features identical across train and test sets
dfmat_matched <- dfm_match(dfmat_testing, features = featnames(dfmat_training))

# Now to inspect classification
actuals <- docvars(dfmat_matched, "dem")
predictions <- predict(tmod_nb, newdata = dfmat_matched)
cMat <- table(actuals, predictions)
cMat
```

```
##        predictions
## actuals  0  1
##       0 30  2
##       1  1 39
```

```r
caret::confusionMatrix(cMat, mode = "everything")
```

```
## Confusion Matrix and Statistics
```

```
##
##         predictions
## actuals  0  1
##       0 30  2
##       1  1 39
##
##                Accuracy : 0.9583
##                  95% CI : (0.883, 0.9913)
##     No Information Rate : 0.5694
##     P-Value [Acc > NIR] : 6.739e-14
##
##                   Kappa : 0.9154
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9677
##             Specificity : 0.9512
##          Pos Pred Value : 0.9375
##          Neg Pred Value : 0.9750
##               Precision : 0.9375
##                  Recall : 0.9677
##                      F1 : 0.9524
##              Prevalence : 0.4306
##          Detection Rate : 0.4167
##    Detection Prevalence : 0.4444
##       Balanced Accuracy : 0.9595
##
##        'Positive' Class : 0
##
```

The naive Bayes classifier performs extremely well, suggesting that members of the two parties are very easily predictable based on the content of their speeches about impeachment. This is an encouraging result because we have more evidence that we may be able to derive ideal point estimates from non-policy related speeches.

```r
# Define sentiment calculation function
sentScore <- function(text, dictname) {
  corp <- cleanCorpus(VCorpus(VectorSource(text)))
  temp_dtm <- DocumentTermMatrix(corp)
  freq <- sort(colSums(as.matrix(temp_dtm)), decreasing=TRUE)

  tib <- tibble("word" = names(freq), "n" = freq)
  dict_ <- get_sentiments(dictname)

  if (dictname=="bing") {
    sent_calc <- tib %>%
      inner_join(dict_, by="word") %>%
      mutate(ntone = ifelse(sentiment=="positive", n, -n)) %>%
      summarize(total_tone=sum(ntone),
                total_words=sum(n))
  } else if (dictname=="afinn") {
    sent_calc <- tib %>%
      inner_join(dict_, by="word") %>%
      mutate(score=n*value) %>%
```

```r
      summarize(total_tone=sum(score),
                total_words=sum(n))
  }

  return(sent_calc$total_tone/sent_calc$total_words)
}
```

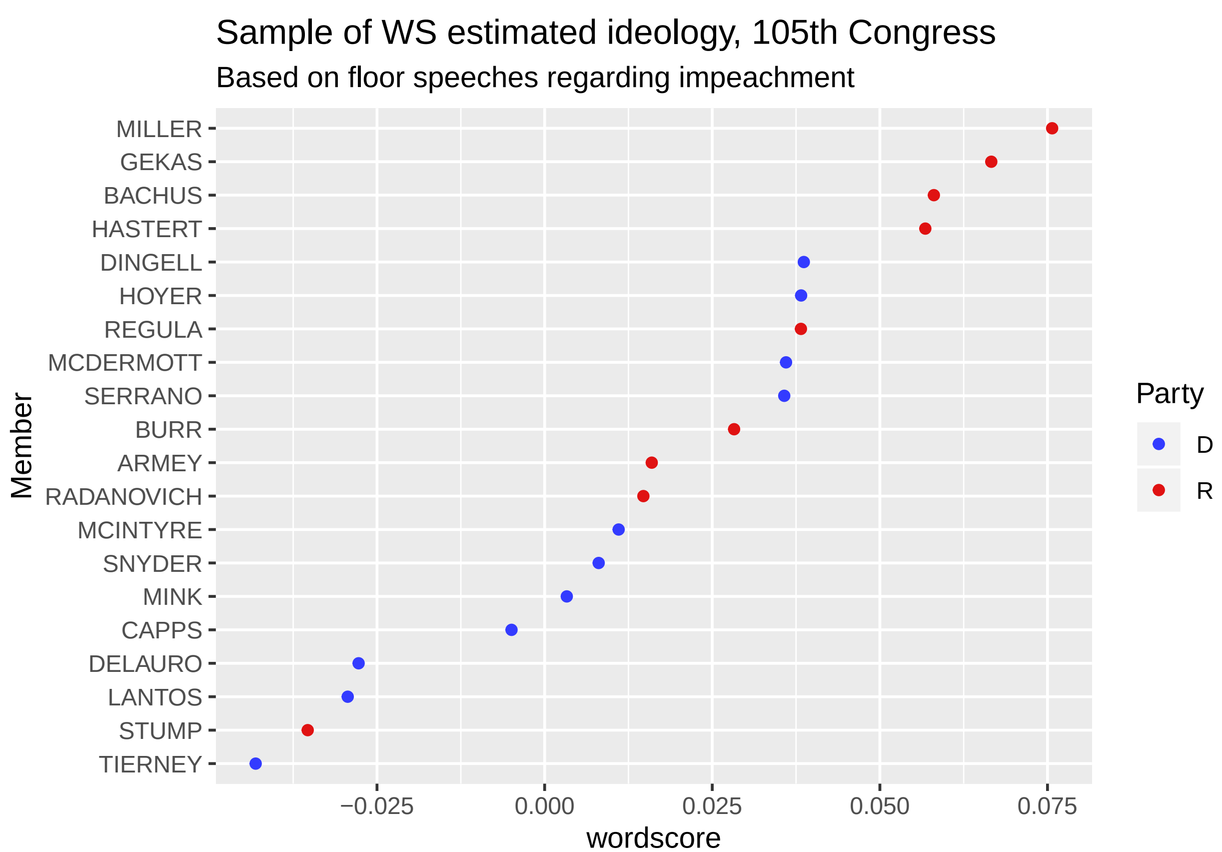
```r
# Attach sentiment scores to members' speeches
bing <- rep(NA, 362)
afinn <- rep(NA, 362)
for (i in 1:362) {
  bing[[i]] <- sentScore(impeach105analysis$text[[i]], "bing")
  afinn[[i]] <- sentScore(impeach105analysis$text[[i]], "afinn")
}
```

```r
# Train wordscores
dfmat_all <- analysis.corp %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
  dfm_trim(min_termfreq=3, termfreq_type="count")

# We'll use Maxine Waters (-1) and Ron Paul (+1) as anchors for wordscore training
reference.scores <- c(rep(NA, 240), 1, rep(NA, 119), -1, NA)

# Train wordscore model and attach predicted scores to names
ws.model <- textmodel_wordscores(dfmat_all, reference.scores, smooth=1)
ws.full.model <- predict(ws.model, level = 0.95)

# Train wordfish model
wf.full.model <- textmodel_wordfish(dfmat_all, sparse=TRUE)

# Train 2D correspondence analysis
ca <- textmodel_ca(dfmat_all)
ca_dim1 <- coef(ca, doc_dim=1)$coef_document
ca_dim2 <- coef(ca, doc_dim=2)$coef_document

# Create df of wordscores with info from the dfm
wswf.df <- tibble(
  firstname = docvars(dfmat_all, "firstname"),
  lastname = docvars(dfmat_all, "lastname"),
  state = docvars(dfmat_all, "state"),
  district = docvars(dfmat_all, "district"),
  party = docvars(dfmat_all, "party"),
  dem = docvars(dfmat_all, "dem"),
  bing = bing,
  afinn = afinn,
  wordscore = ws.full.model,
  wftheta = wf.full.model$theta,
  wfse = wf.full.model$se,
  ca_dim1 = ca_dim1,
  ca_dim2 = ca_dim2,
  recipient_cfscore = docvars(dfmat_all, "recipient_cfscore"),
  nominate_dim1 = docvars(dfmat_all, "nominate_dim1"),
  nominate_dim2 = docvars(dfmat_all, "nominate_dim2")
```

```
)
```

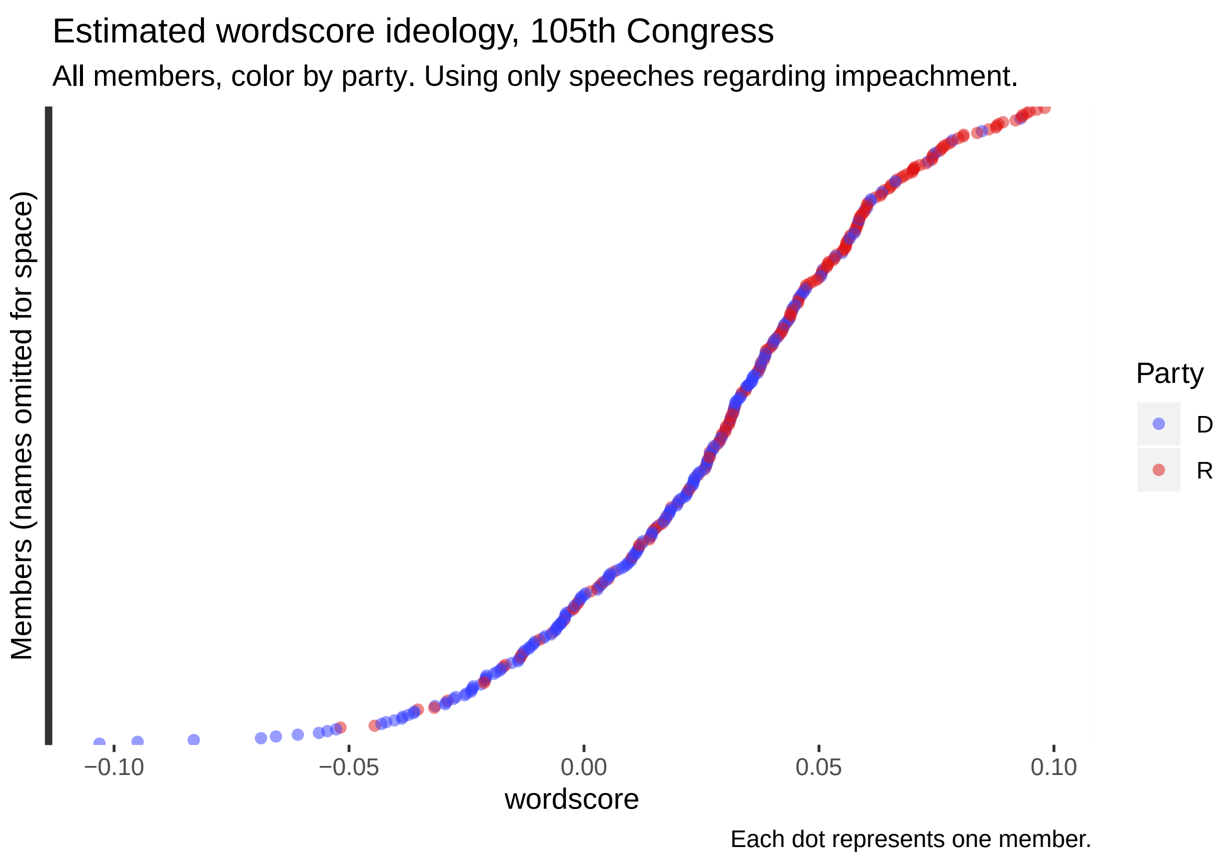
We can plot a random subsample of wordscores:

```
set.seed(500)
#update_geom_defaults("point", list(size=1.5))
ws.plot.1 <- wswf.df[sample(nrow(wswf.df), 20),] %>%
  ggplot(., aes(fct_reorder(as.factor(lastname), wordscore),
                wordscore,
                color=party)) +
  geom_point() +
  coord_flip() +
  scale_color_manual(values=group.colors) +
  labs(x = "Member",
       title = "Sample of WS estimated ideology, 105th Congress",
       subtitle = "Based on floor speeches regarding impeachment",
       color = "Party")
ws.plot.1
```




```
ws.plot.2 <- wswf.df %>%
  filter(party != "I" & wordscore < .2 & wordscore > -.2) %>%
  mutate(fullname = paste0(lastname, firstname)) %>%
  ggplot(., aes(fct_reorder(as.factor(fullname), wordscore),
                wordscore, color=party)) +
  geom_point(alpha=0.5) +
```

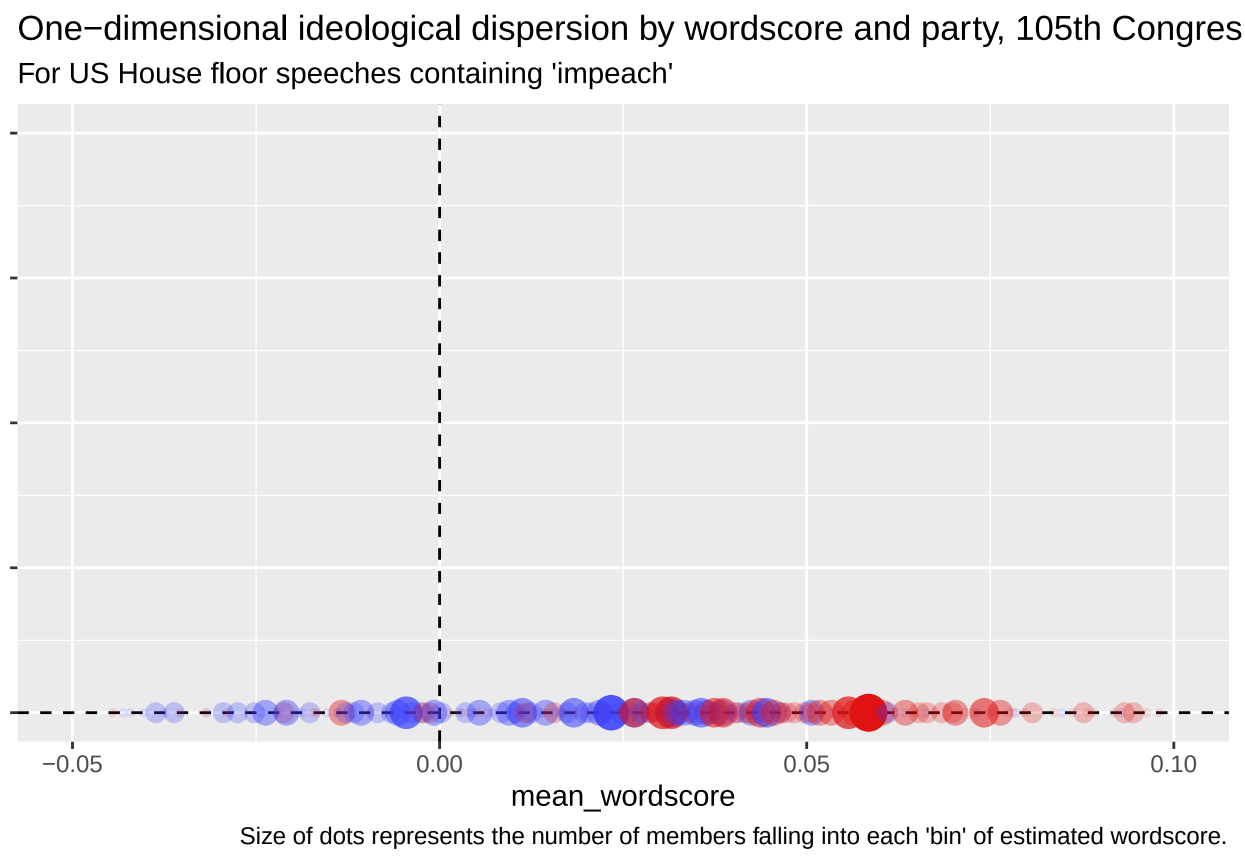
```r
  scale_color_manual(values=group.colors) +
  coord_flip() +
  theme(axis.text.y=element_blank(),
        panel.background=element_rect(fill="white",
                                      color="lightgray", size=0.5,
                                      linetype="solid"),
        panel.grid.major=element_line(size=0.5, linetype="solid",
                                      color="white"),
        panel.grid.minor=element_line(size=0.25, linetype="solid",
                                      color="white")) +
  labs(title = "Estimated wordscore ideology, 105th Congress",
       subtitle = "All members, color by party. Using only speeches regarding impeachment.",
       x = "Members (names omitted for space)",
       color = "Party",
       caption = "Each dot represents one member.")
ws.plot.2
```




```r
ws.plot.3 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = wordscore - (wordscore %% .001)) %>%
  arrange(bin, wordscore) %>%
  group_by(bin, party) %>%
  summarize(mean_wordscore = mean(wordscore, na.rm=TRUE),
            count = n()) %>%
  mutate(x = 0) %>%
```

```
  ggplot(., aes(x=c(0), mean_wordscore, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(-.05, .1) +
  xlim(0, .001) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by wordscore and party, 105th Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated word
       color="Party")
ws.plot.3
```

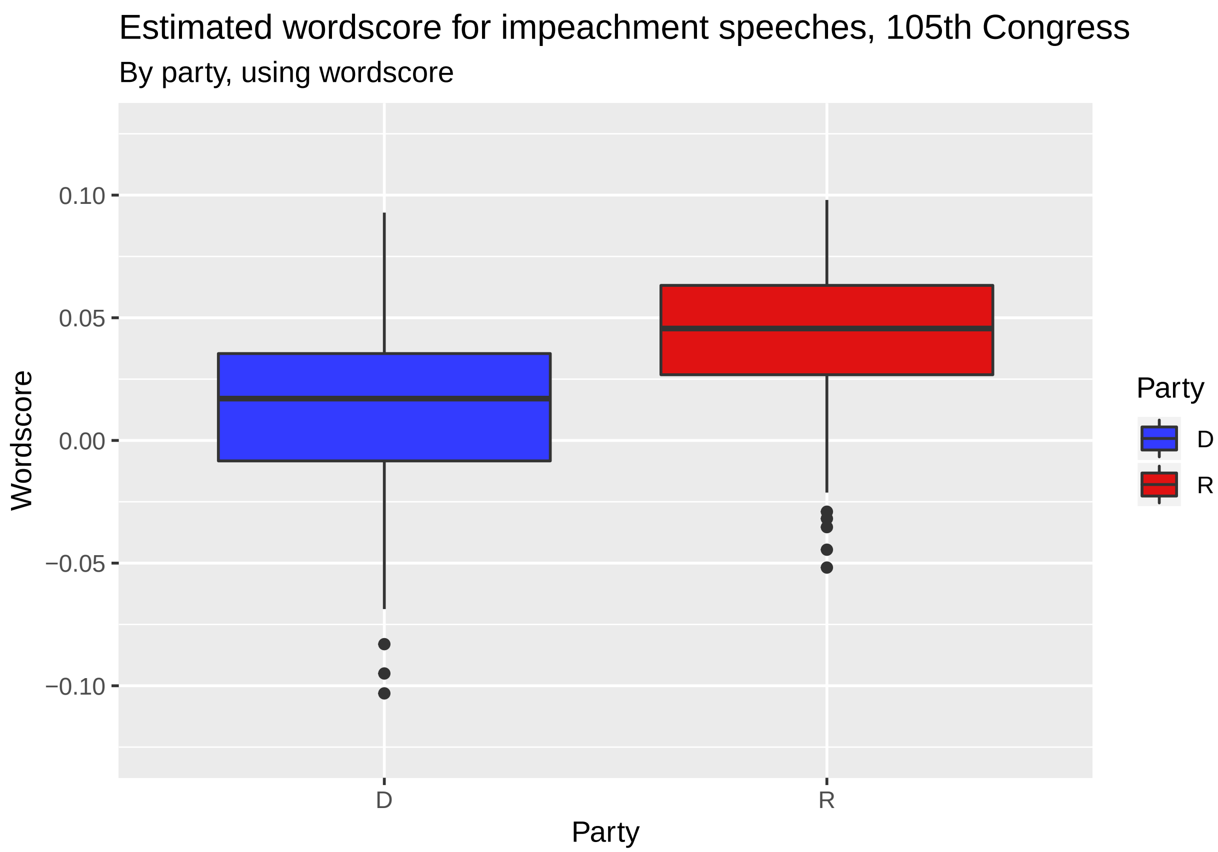
```
## Warning: Removed 12 rows containing missing values (geom_point).
```




```
# Produce boxplots of wordscore by party
ws.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=wordscore, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
```

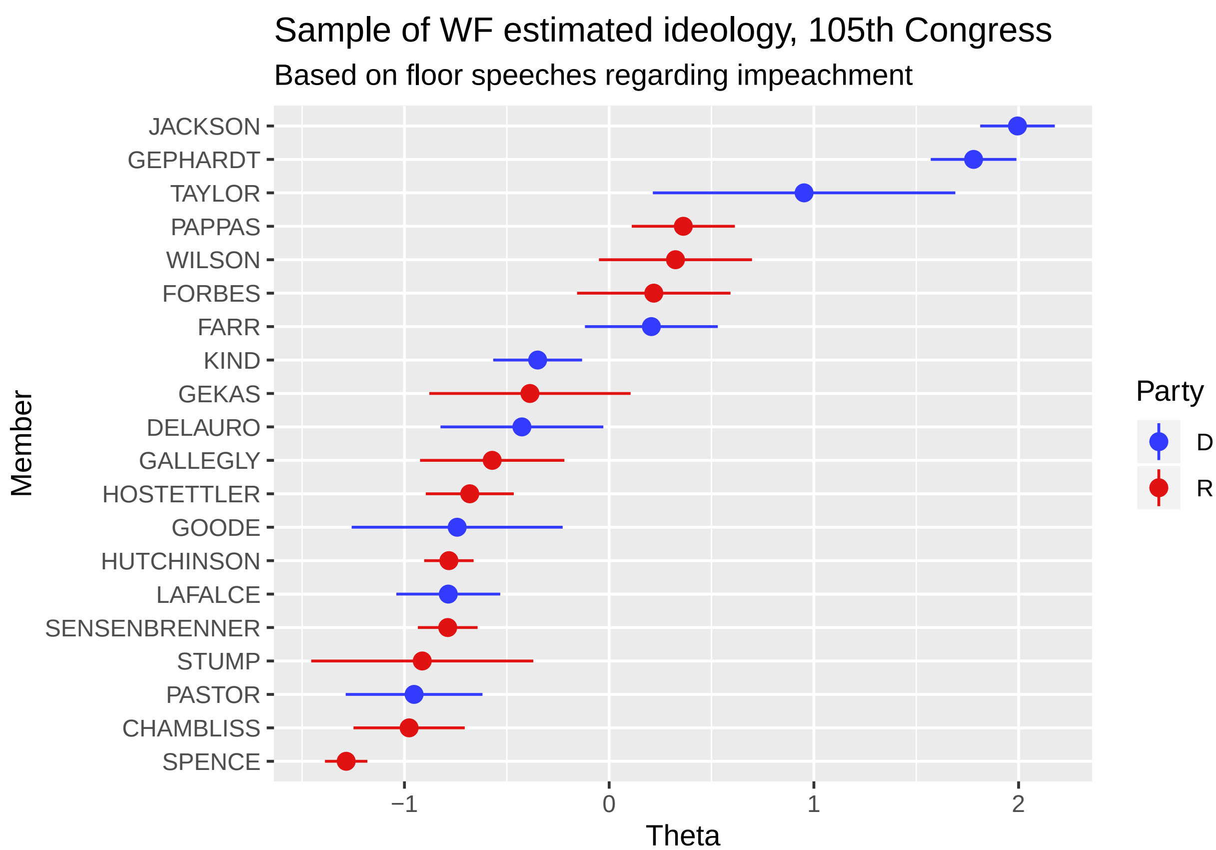
```r
  ylim(c(-.125,.125)) +
  labs(title="Estimated wordscore for impeachment speeches, 105th Congress",
       subtitle="By party, using wordscore",
       fill="Party",
       x = "Party",
       y = "Wordscore")
ws.plot.4
```

```
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
```



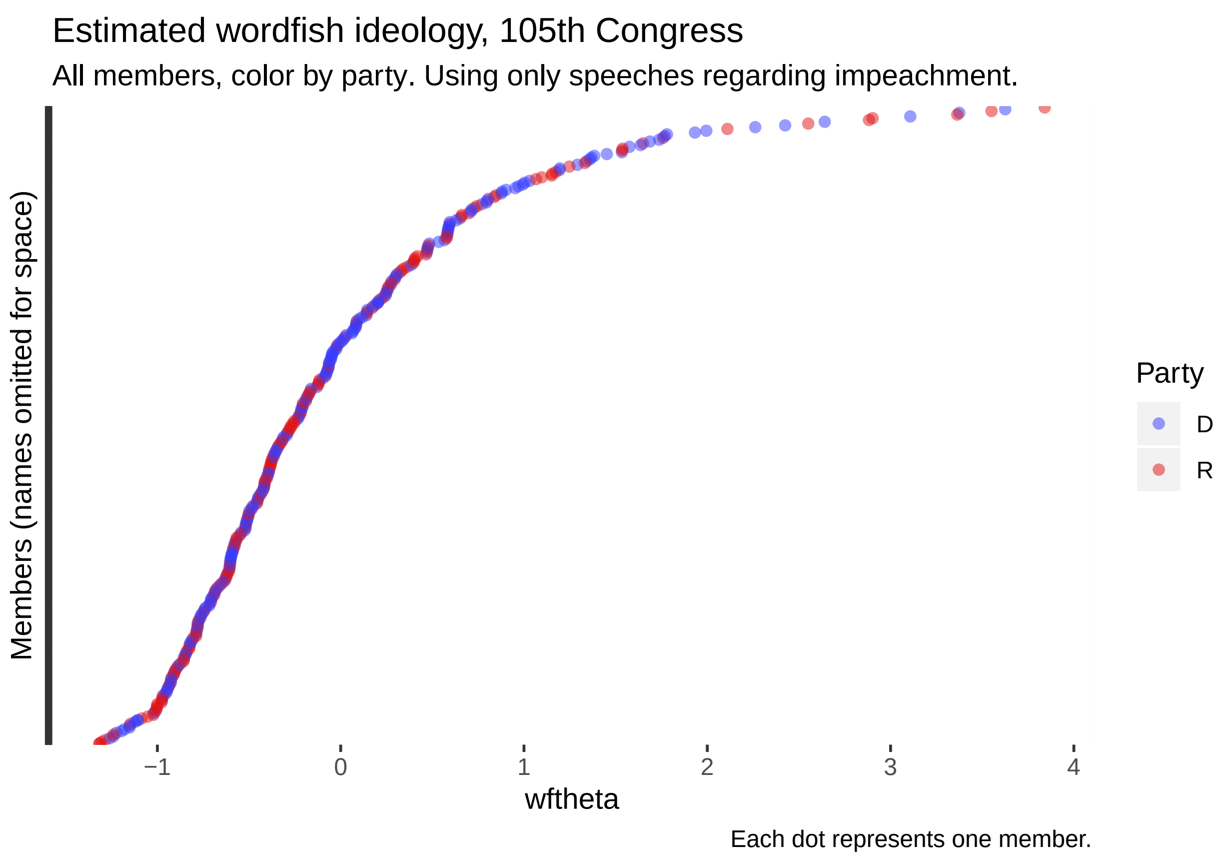

```r
set.seed(303)
# Random sample of wordfish scores
wf.plot.1 <- wswf.df[sample(nrow(wswf.df), 20),] %>%
  arrange(wftheta) %>%
  ggplot() +
  geom_pointrange(aes(x=fct_reorder(as.factor(lastname), wftheta),
                      y=wftheta,
                      color=party,
                      ymin=wftheta-2*wfse,
                      ymax=wftheta+2*wfse)) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  labs(x = "Member",
       y = "Theta",
```

```
       title = "Sample of WF estimated ideology, 105th Congress",
       subtitle = "Based on floor speeches regarding impeachment",
       color = "Party")
wf.plot.1
```




```
# All members' wordfish scores
wf.plot.2 <- wswf.df %>%
  filter(party != "I" & wftheta < 4) %>%
  mutate(fullname = paste0(lastname, firstname)) %>%
  ggplot(., aes(fct_reorder(as.factor(fullname), wftheta),
              wftheta, color=party)) +
  geom_point(alpha=0.5) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  theme(axis.text.y=element_blank(),
        panel.background=element_rect(fill="white",
                                      color="lightgray", size=0.5,
                                      linetype="solid"),
        panel.grid.major=element_line(size=0.5, linetype="solid",
                                      color="white"),
        panel.grid.minor=element_line(size=0.25, linetype="solid",
                                      color="white")) +
  labs(title = "Estimated wordfish ideology, 105th Congress",
       subtitle = "All members, color by party. Using only speeches regarding impeachment.",
       x = "Members (names omitted for space)",
```

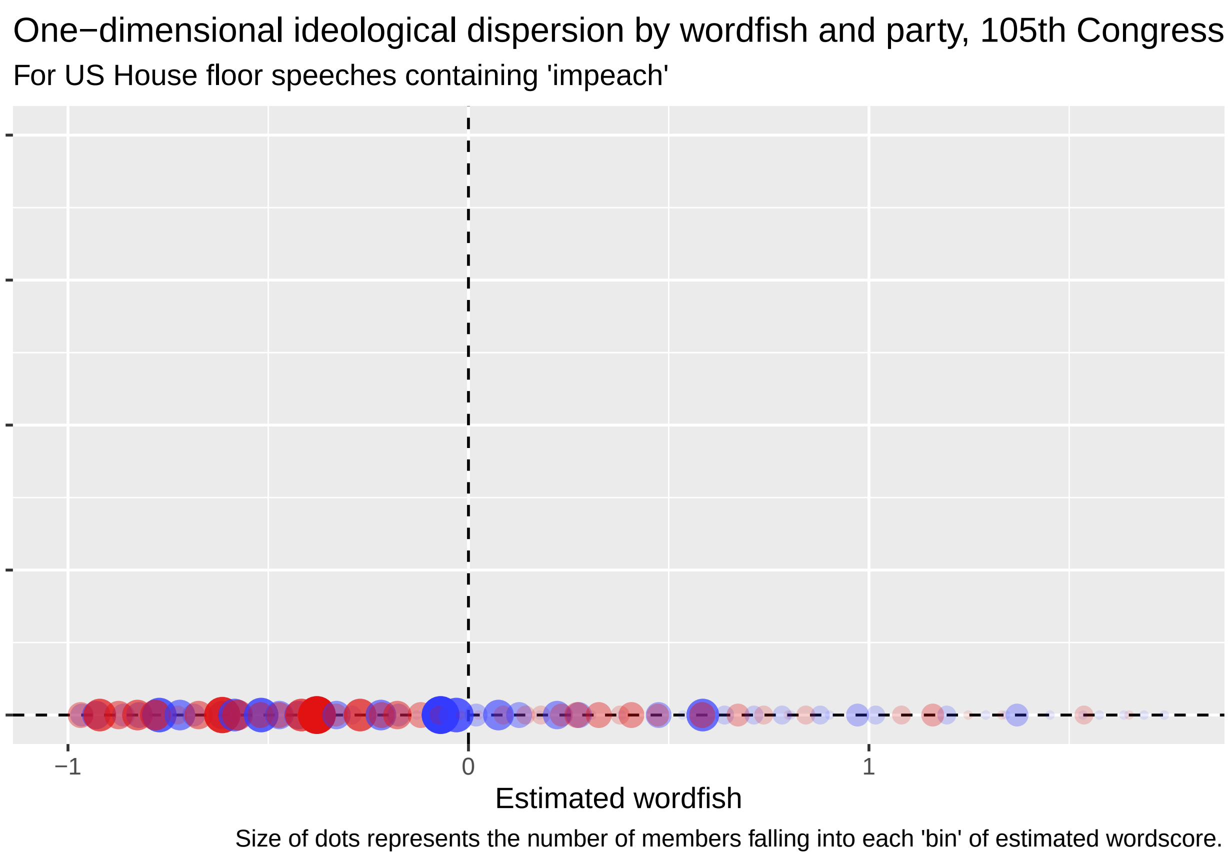
```r
       color = "Party",
       caption = "Each dot represents one member.")
wf.plot.2
```



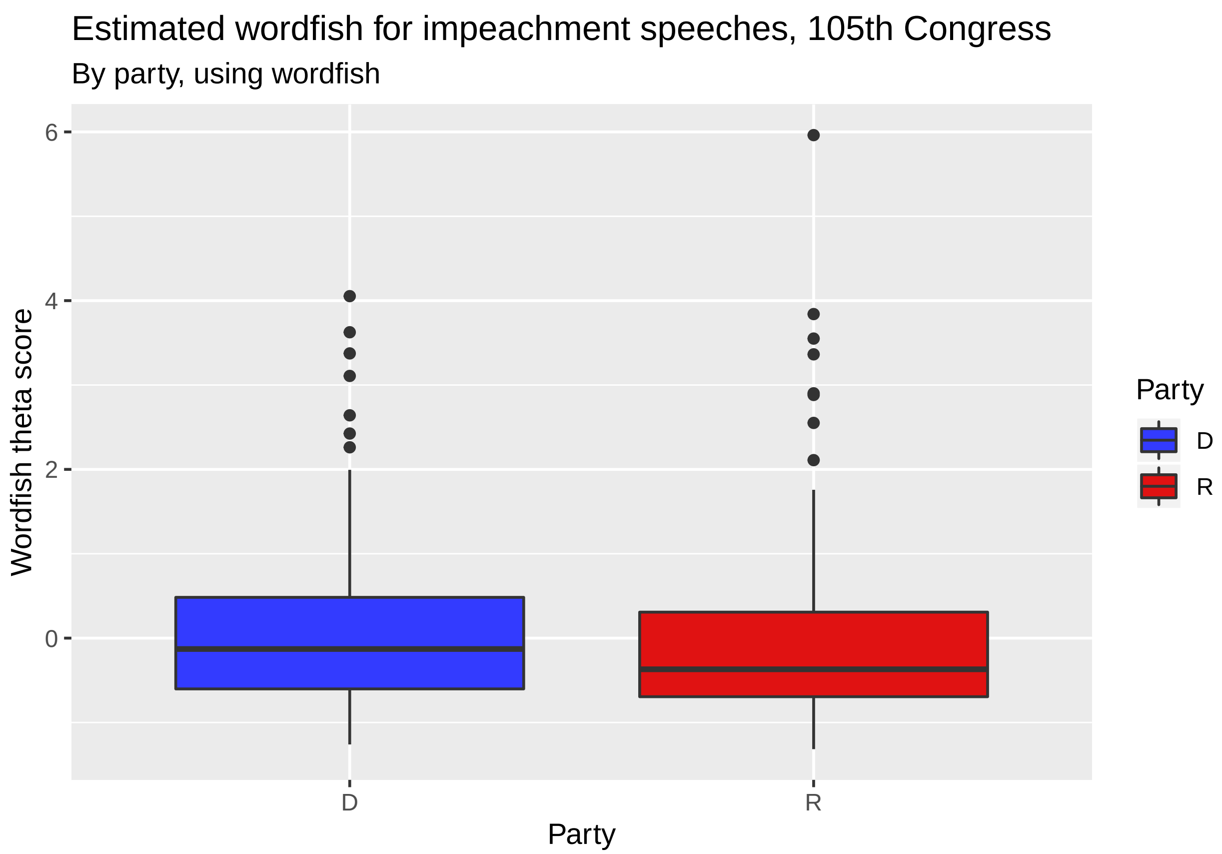

```r
#update_geom_defaults("point", list(size=1.5))
wf.plot.3 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = wftheta - (wftheta %% .05)) %>%
  arrange(bin, wftheta) %>%
  group_by(bin, party) %>%
  summarize(mean_wftheta = mean(wftheta, na.rm=TRUE),
            count = n()) %>%
  mutate(x = 0) %>%
  ggplot(., aes(x=c(0), mean_wftheta, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-1, 1.75)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by wordfish and party, 105th Congress",
       subtitle="For US House floor speeches containing 'impeach'",
```

```
       caption="Size of dots represents the number of members falling into each 'bin' of estimated words
       color="Party",
       y="Estimated wordfish")
wf.plot.3
```

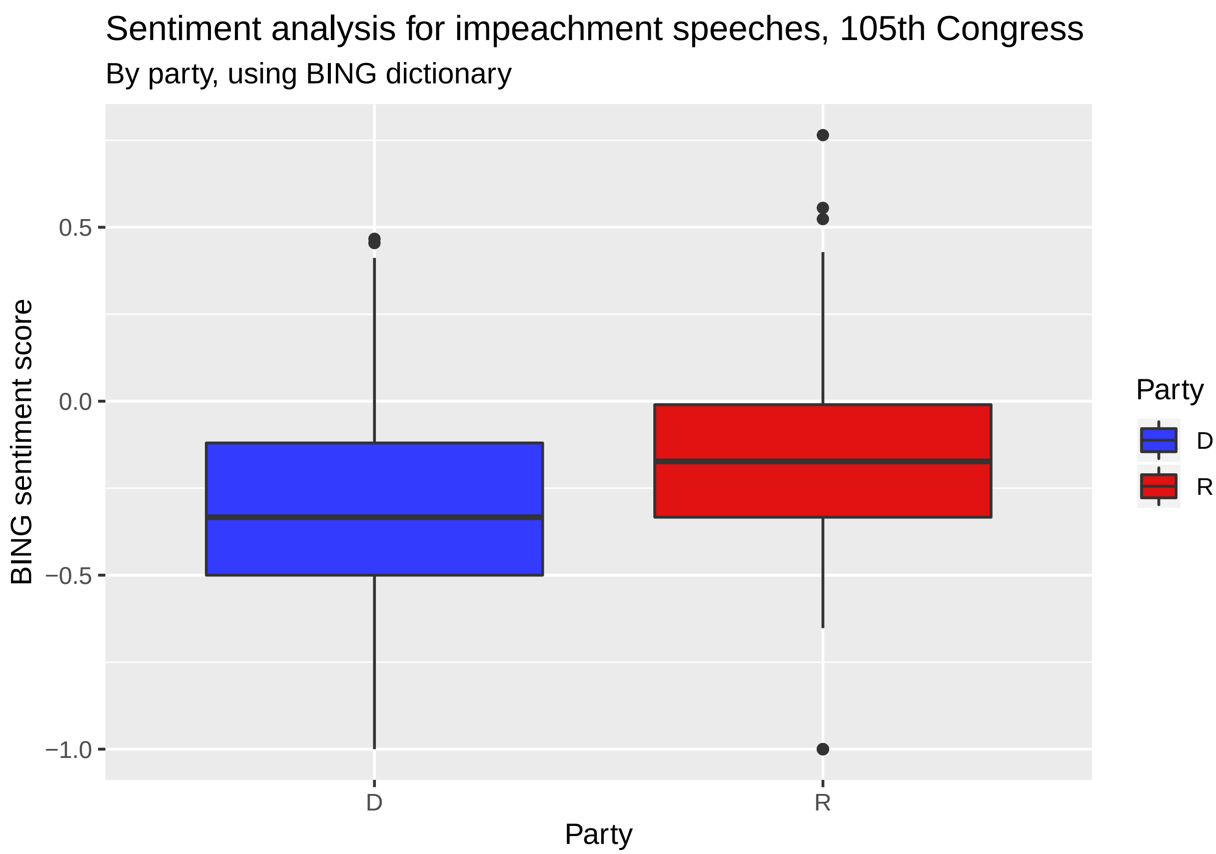
```
## Warning: Removed 30 rows containing missing values (geom_point).
```




```
# Produce boxplots of wordfish by party
wf.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=wftheta, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  labs(title="Estimated wordfish for impeachment speeches, 105th Congress",
       subtitle="By party, using wordfish",
       fill="Party",
       x = "Party",
       y = "Wordfish theta score")
wf.plot.4
```

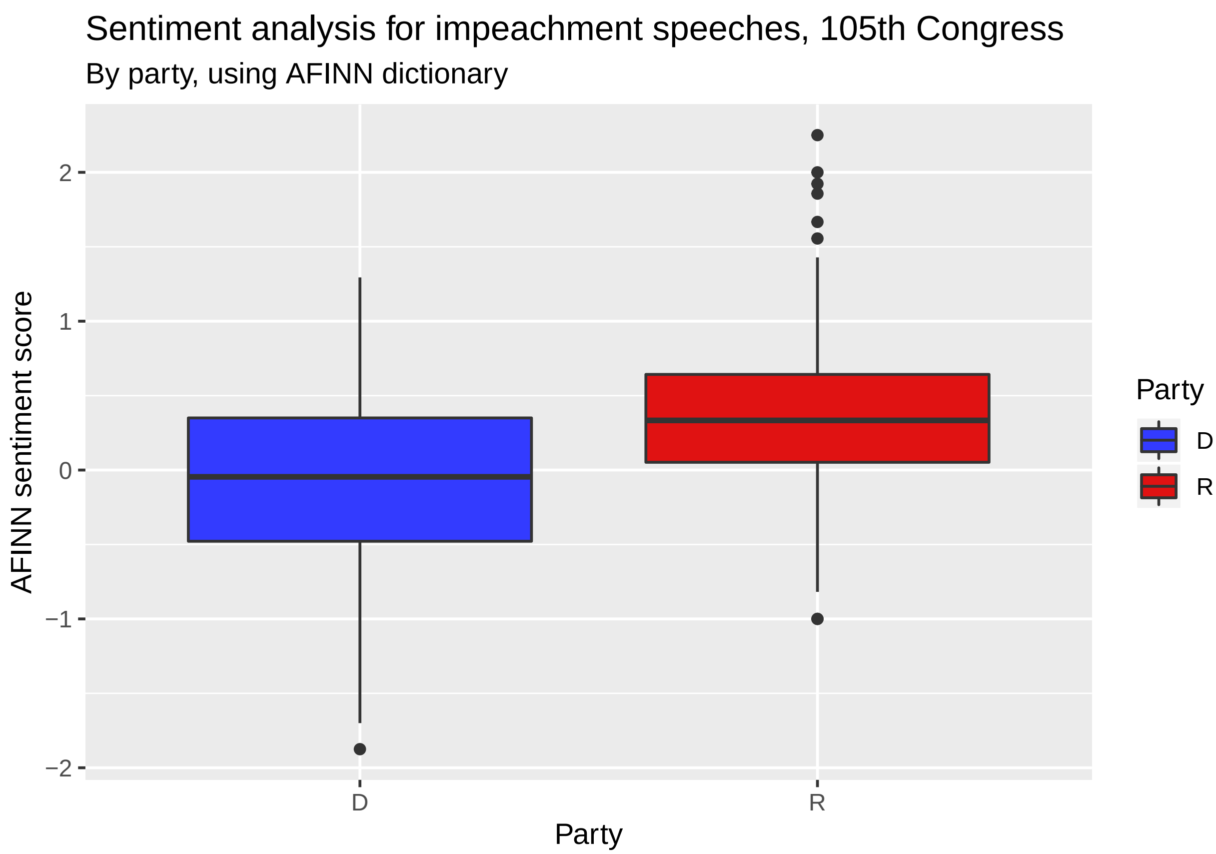



```
# Produce boxplots of sentiment analysis
sent.plot.1 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=bing, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  labs(title="Sentiment analysis for impeachment speeches, 105th Congress",
       subtitle="By party, using BING dictionary",
       fill="Party",
       x = "Party",
       y = "BING sentiment score")
sent.plot.1
```



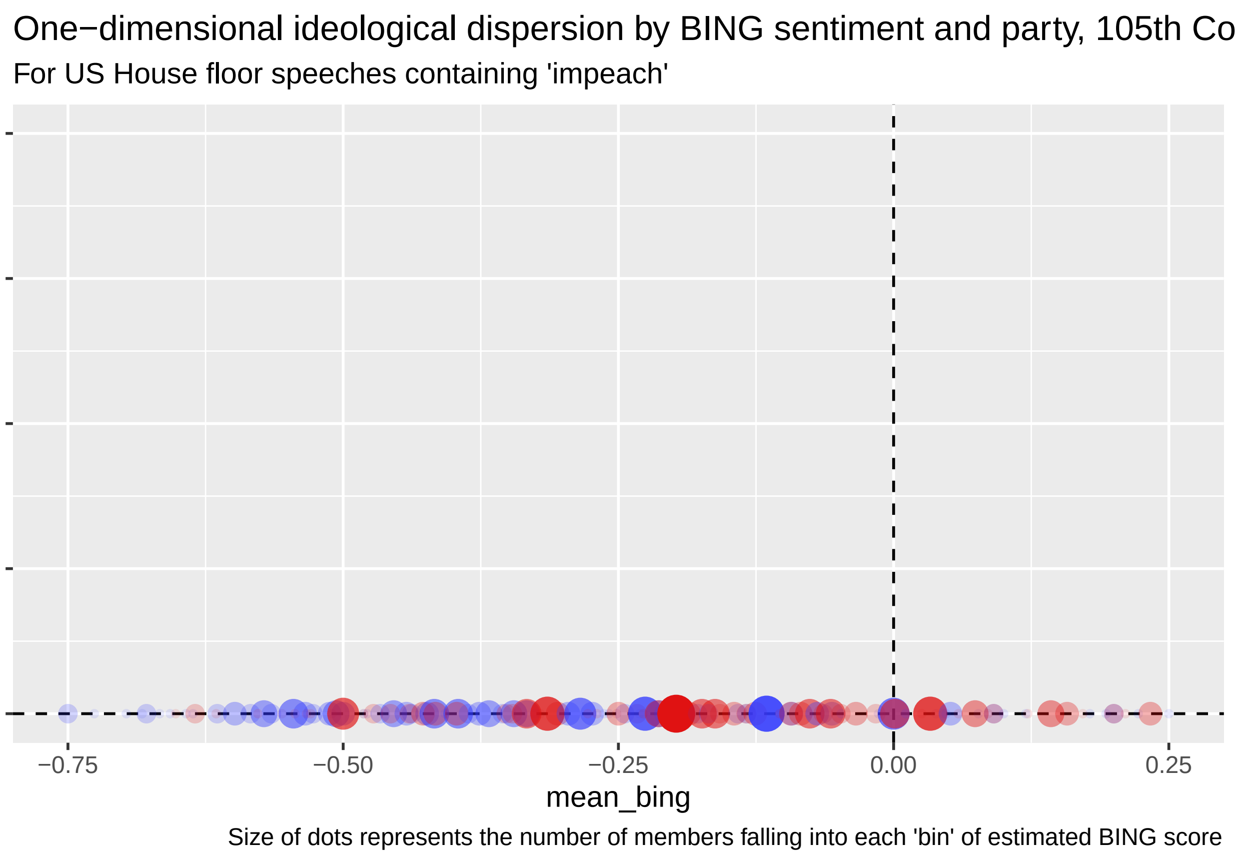

```
sent.plot.2 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=afinn, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  labs(title="Sentiment analysis for impeachment speeches, 105th Congress",
       subtitle="By party, using AFINN dictionary",
       fill="Party",
       x = "Party",
       y = "AFINN sentiment score")
sent.plot.2
```

```
## Warning: Removed 1 rows containing non-finite values (stat_boxplot).
```



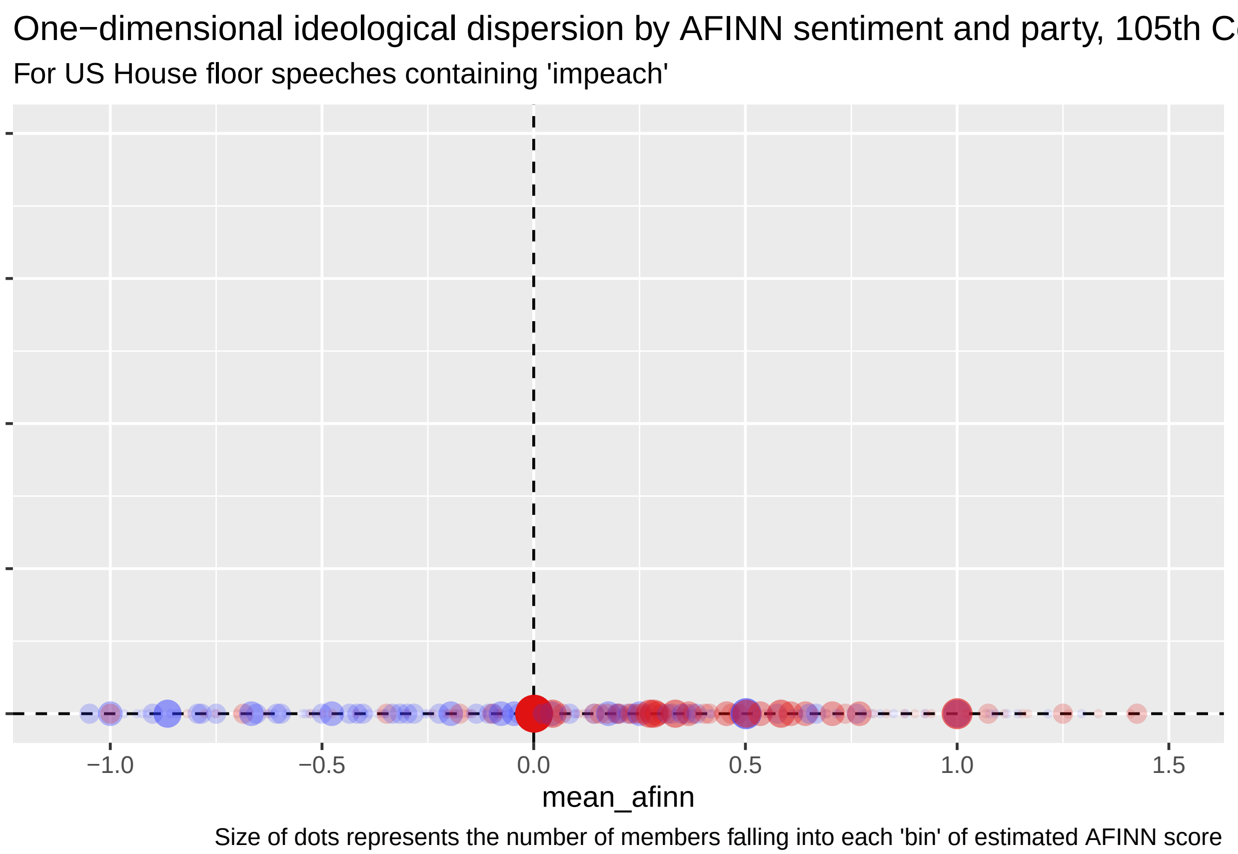

```r
# Produce 1D line plots of ideology using sentiment
sent.plot.3 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = bing - (bing %% .01)) %>%
  arrange(bin, bing) %>%
  group_by(bin, party) %>%
  summarize(mean_bing = mean(bing, na.rm=TRUE),
            count=n()) %>%
  ggplot(., aes(x=c(0), mean_bing, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-.75,.25)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by BING sentiment and party, 105th Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated BING
       color="Party")
sent.plot.3
```

```
## Warning: Removed 17 rows containing missing values (geom_point).
```



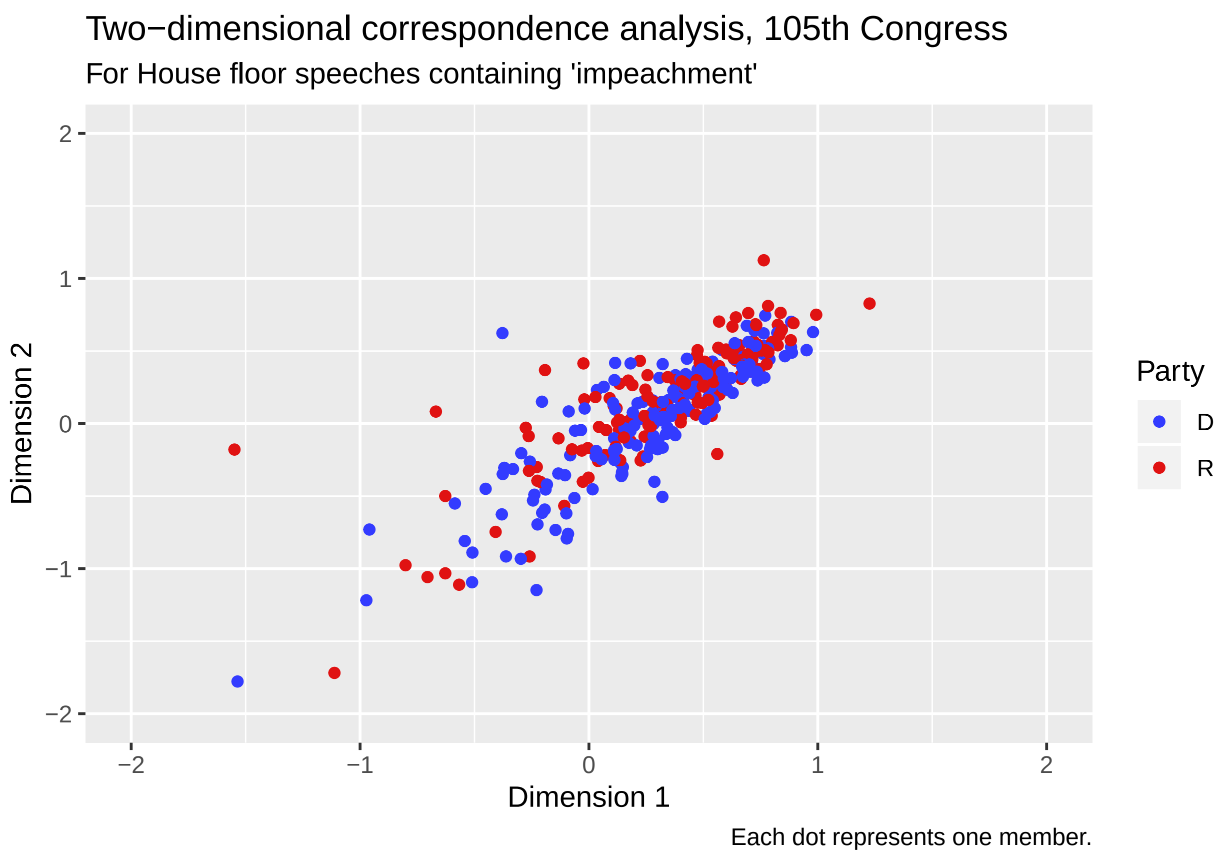

```r
sent.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = afinn - (afinn %% .01)) %>%
  arrange(bin, afinn) %>%
  group_by(bin, party) %>%
  summarize(mean_afinn = mean(afinn, na.rm=TRUE),
            count=n()) %>%
  ggplot(., aes(x=c(0), mean_afinn, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-1.1,1.5)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by AFINN sentiment and party, 105th Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated AFINN
       color="Party")
sent.plot.4
```

```
## Warning: Removed 11 rows containing missing values (geom_point).
```




```r
# Plot 2D correspondence analysis
ca2d <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(ca_dim1, ca_dim2, color=party)) +
  geom_point() +
  xlim(-2, 2) +
  ylim(-2, 2) +
  scale_color_manual(values=group.colors) +
  labs(title="Two-dimensional correspondence analysis, 105th Congress",
       subtitle="For House floor speeches containing 'impeachment'",
       color="Party",
       x="Dimension 1",
       y="Dimension 2",
       caption="Each dot represents one member.")
ca2d
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```




```
# Use a function to min-max scale our estimated ideology variables
normalize <- function(df, col) {
  min <- min(df[col], na.rm=TRUE)
  max <- max(df[col], na.rm=TRUE)
  newcol <- rep(NA, nrow(df))
  for (i in 1:nrow(df)) {
    if (is.na(df[[col]][[i]])) {
      newcol[[i]] <- NA
    } else {
      newcol[[i]] <- (df[[col]][[i]] - min) / (max-min)
    }
  }
  return(newcol)
}

wswf.df.normalized <- wswf.df %>%
  mutate(ln.bing.n = log(normalize(., "bing")),
         ln.afinn.n = log(normalize(., "afinn")),
         ln.wordscore.n = log(normalize(., "wordscore")),
         ln.wftheta.n = log(normalize(., "wftheta")),
         ln.rcf.n = log(normalize(., "recipient_cfscore")),
         ln.nd1.n = log(normalize(., "nominate_dim1")))
```

```
# Now do the log of scaled vars regression for percent interpretation
dwn.md1 <- lm(ln.nd1.n ~ ln.bing.n, data = subset(wswf.df.normalized,
                                                 !is.infinite(ln.nd1.n) & !is.infinite(ln.bing.n) &
```

```r
                                                     !is.na(ln.nd1.n) & !is.na(ln.bing.n)))

dwn.md2 <- lm(ln.nd1.n ~ ln.afinn.n, data = subset(wswf.df.normalized,
                                                    !is.infinite(ln.nd1.n) & !is.infinite(ln.afinn.n) &
                                                      !is.na(ln.nd1.n) & !is.na(ln.afinn.n)))

dwn.md3 <- lm(ln.nd1.n ~ ln.wordscore.n, data = subset(wswf.df.normalized,
                                                        !is.infinite(ln.nd1.n) & !is.infinite(ln.wordsc
                                                          !is.na(ln.nd1.n) & !is.na(ln.wordscore.n)))

dwn.md4 <- lm(ln.nd1.n ~ ln.wftheta.n, data = subset(wswf.df.normalized,
                                                      !is.infinite(ln.nd1.n) & !is.infinite(ln.wftheta.r
                                                        !is.na(ln.nd1.n) & !is.na(ln.wftheta.n)))

stargazer(dwn.md1, dwn.md2, dwn.md3, dwn.md4, type = "text",
          title = "Regression of log DW-NOMINATE on log ideology estimate")
```

```
##
## Regression of log DW-NOMINATE on log ideology estimate
## ===========================================================================================
##                                                         Dependent variable:
##                     ------------------------------------------------------------------------
##                                                               ln.nd1.n
##                               (1)                      (2)                      (3)
## -------------------------------------------------------------------------------------------
## ln.bing.n                  0.482***
##                            (0.115)
##
## ln.afinn.n                                          0.584***
##                                                     (0.126)
##
## ln.wordscore.n                                                               2.973***
##                                                                              (0.413)
##
## ln.wftheta.n                                                                                    -0
##                                                                                                 (0
##
## Constant                  -0.644***                -0.628***                 0.817***          -1.
##                            (0.113)                  (0.105)                  (0.264)            (0
##
## -------------------------------------------------------------------------------------------
## Observations                 319                      322                      323               3
## R2                          0.052                    0.063                    0.139              0
## Adjusted R2                 0.049                    0.060                    0.136              0
## Residual Std. Error    0.784 (df = 317)        0.778 (df = 320)        0.732 (df = 321)      0.801
## F Statistic         17.556*** (df = 1; 317) 21.386*** (df = 1; 320) 51.736*** (df = 1; 321) 2.896* (d
## ===========================================================================================
## Note:                                                                          *p<0.1; **p<0.0
```

```r
# Now do the log of scaled vars regression for percent interpretation (DIME)
dwn.md5 <- lm(ln.rcf.n ~ ln.bing.n, data = subset(wswf.df.normalized,
                                                   !is.infinite(ln.rcf.n) & !is.infinite(ln.bing.n) &
                                                     !is.na(ln.rcf.n) & !is.na(ln.bing.n)))
```

```r
dwn.md6 <- lm(ln.rcf.n ~ ln.afinn.n, data = subset(wswf.df.normalized,
                                                    !is.infinite(ln.rcf.n) & !is.infinite(ln.afinn.n) &
                                                      !is.na(ln.rcf.n) & !is.na(ln.afinn.n)))

dwn.md7 <- lm(ln.rcf.n ~ ln.wordscore.n, data = subset(wswf.df.normalized,
                                                        !is.infinite(ln.rcf.n) & !is.infinite(ln.wordsc
                                                          !is.na(ln.rcf.n) & !is.na(ln.wordscore.n)))

dwn.md8 <- lm(ln.rcf.n ~ ln.wftheta.n, data = subset(wswf.df.normalized,
                                                      !is.infinite(ln.rcf.n) & !is.infinite(ln.wftheta.
                                                        !is.na(ln.rcf.n) & !is.na(ln.wftheta.n)))

stargazer(dwn.md5, dwn.md6, dwn.md7, dwn.md8, type = "text",
          title = "Regression of log normalized DIME on log ideology estimate")
```

```
##
## Regression of log normalized DIME on log ideology estimate
## ==========================================================================================================
##                                                     Dependent variable:
##                     --------------------------------------------------------------------------------------
##                                                          ln.rcf.n
##                               (1)                     (2)                     (3)                     (4
## ----------------------------------------------------------------------------------------------------------
## ln.bing.n                  0.292***
##                            (0.099)
##
## ln.afinn.n                                         0.399***
##                                                     (0.108)
##
## ln.wordscore.n                                                              2.222***
##                                                                              (0.364)
##
## ln.wftheta.n                                                                                         -0.
##                                                                                                       (0.
##
## Constant                  -0.621***               -0.576***                 0.529**                 -0.9
##                            (0.097)                 (0.090)                 (0.233)                 (0.
##
## ----------------------------------------------------------------------------------------------------------
## Observations                 282                     284                     286                     28
## R2                          0.030                   0.046                   0.116                   0.
## Adjusted R2                 0.027                   0.043                   0.113                   0.
## Residual Std. Error    0.641 (df = 280)        0.636 (df = 282)        0.611 (df = 284)        0.648 (d
## F Statistic         8.690*** (df = 1; 280) 13.647*** (df = 1; 282) 37.351*** (df = 1; 284) 2.212 (df
## ==========================================================================================================
## Note:                                                                             *p<0.1; **p<0.05;
```

Analysis: 116th Congress

Alec MacMillen

12/11/2019

Part 1: Load data

Load scraped 116th text data

```r
# Read in speeches from the 116th Congress gathered using web scraping
impeach116h <- read_csv("Data/crec-116th/speeches_116_impeach.csv") %>%
  mutate(speech = removeNumbers(stripWhitespace(speech)),
         lastname = toupper(lastname),
         firstname = toupper(firstname))
```

```
## Parsed with column specification:
## cols(
##   date = col_character(),
##   lastname = col_character(),
##   firstname = col_character(),
##   state = col_character(),
##   speech = col_character(),
##   party = col_character(),
##   district = col_double()
## )
```

```r
impeach116h <- impeach116h %>% mutate(speakerid = as.factor(group_indices(., lastname, firstname)))
```

Basic exploratory analysis

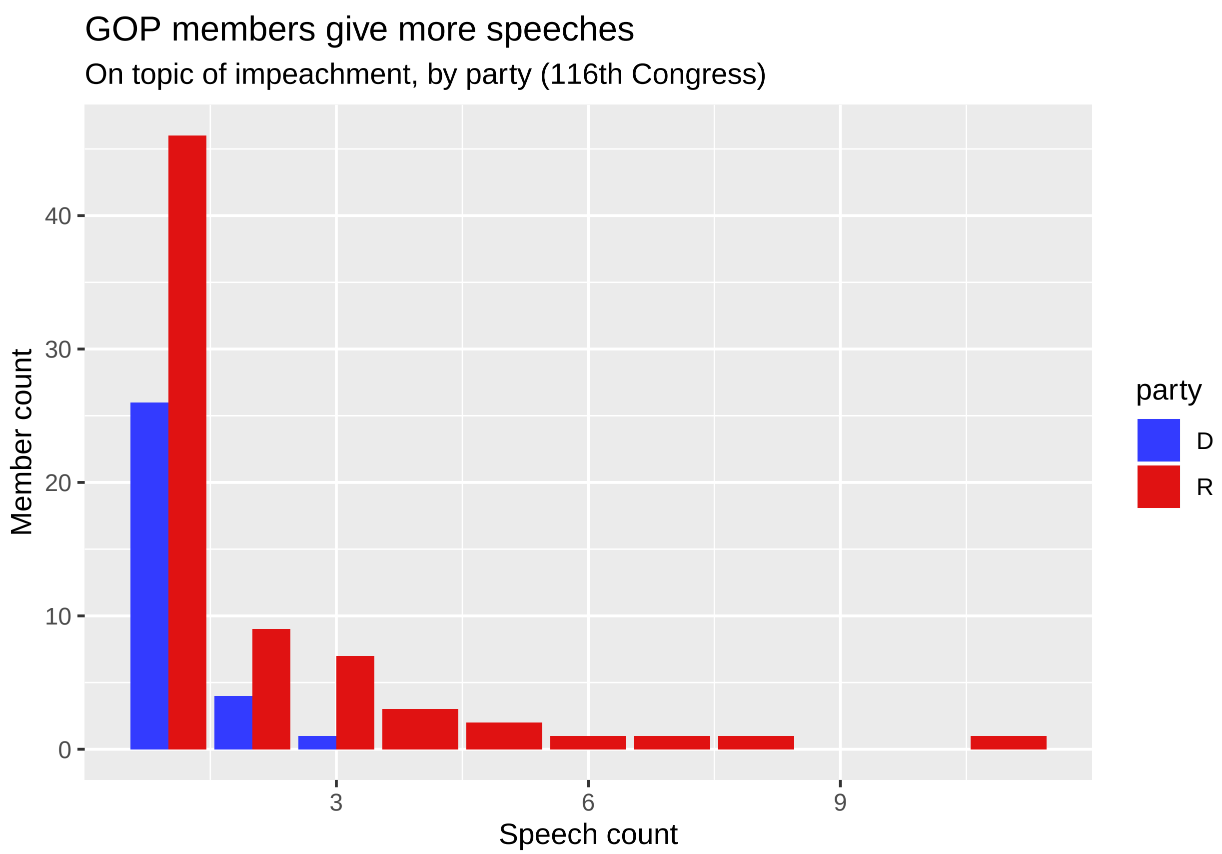
```r
# For faster analysis, drop large "speech" field
impeach116h_eda <- impeach116h %>% select(-speech)

impeach116h_eda %>%
  group_by(speakerid) %>%
  summarize(count = n()) %>%
  skim(count)
```

```
## Skim summary statistics
##  n obs: 103
##  n variables: 2
##
## -- Variable type:integer ------------------------------------------------------------------
##  variable missing complete   n mean   sd p0 p25 p50 p75 p100     hist
##     count       0      103 103 2.04 3.56  1   1   1   2   34 <U+2587><U+2581><U+2581><U+2581><U+2581>
```

```r
impeach116h_eda %>%
  group_by(party) %>%
  summarize(count = n()) %>%
  print()
```

```
## # A tibble: 2 x 2
##   party count
##   <chr> <int>
## 1 D        71
## 2 R       139
```

```
# Visualize relative speech frequency by party
scount_by_party_116 <- impeach116h_eda %>%
  group_by(speakerid, party) %>%
  summarize(count = n()) %>%
  ungroup() %>%
  arrange(count) %>%
  # Drop Al Green, who gives by far the most speeches of anyone
  filter(count < 15) %>%
  ggplot(., aes(count, fill=party)) +
  geom_bar(position = "dodge") +
  scale_fill_manual(values=group.colors) +
  labs(title = "GOP members give more speeches",
       subtitle = "On topic of impeachment, by party (116th Congress)",
       y = "Member count",
       x = "Speech count")
scount_by_party_116
```



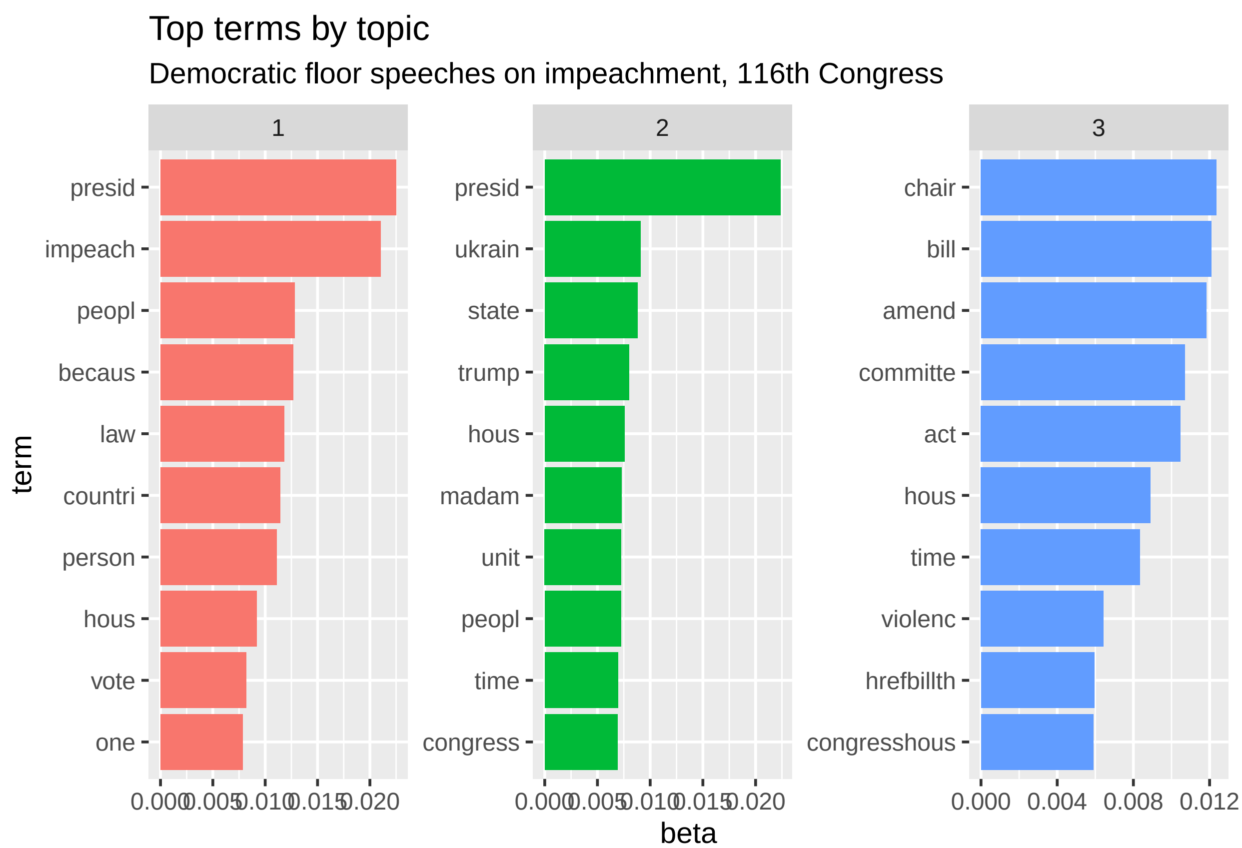

```
# We're interested in modeling at the speaker/legislator level, so combine speeches by multiple speaker
impeach116hgrp <- impeach116h %>%
```

```r
  group_by(speakerid, lastname, firstname, state, party, district) %>%
  summarize(speech = paste0(speech, collapse = " ")) %>%
  ungroup() %>%
  rename(text = speech)

# Define stopwords: use basic English stopwords and Congress-related stopwords
c_stopwords <- c("absent", "adjourn", "ask", "can", "chairman", "committee",
                 "con", "democrat", "etc", "gentleladies", "gentlelady",
                 "gentleman", "gentlemen", "gentlewoman", "gentlewomen",
                 "hereabout", "hereafter", "hereat", "hereby", "herein",
                 "hereinafter", "hereinbefore", "hereinto", "hereof",
                 "hereon", "hereto", "heretofore", "hereunder", "hereunto",
                 "hereupon", "herewith", "month", "mr", "mrs", "nai", "nay",
                 "none", "now", "part", "per", "pro", "republican", "say", "senator",
                 "shall", "sir", "speak", "speaker", "tell", "tempore", "thank", "thereabout",
                 "thereafter", "thereagainst", "thereat", "therebefore", "therebeforn",
                 "thereby", "therefore", "therefor", "therefrom", "therein",
                 "thereinafter", "thereof", "thereon", "thereto", "theretofore",
                 "thereunder", "thereunto", "thereupon", "therewith", "therewithal",
                 "today", "whereabouts", "whereafter", "whereas", "whereat",
                 "whereby", "wherefore", "wherefrom", "wherein", "whereinto",
                 "whereo", "whereon", "whereto", "whereunder", "whereupon", "wherever",
                 "wherewith", "wherewithal", "will", "yea", "yes", "yield")

# Full stopwords list is congressional stopwords + base English stopwords
allstop <- c(stopwords("english"), c_stopwords)

# Split overall corpus into one for each party
dem <- impeach116hgrp %>% filter(party == "D")
demC <- VCorpus(VectorSource(dem$text))

rep <- impeach116hgrp %>% filter(party == "R")
repC <- VCorpus(VectorSource(rep$text))

# Function for corpus cleaning in preparation for topic modeling
cleanCorpus <- function(incorpus) {
  ccorp <- tm_map(incorpus, removePunctuation)
  for (j in seq(ccorp)) {
    ccorp[[j]] <- gsub("/", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("â ", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("@", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("/u2028", " ", ccorp[[j]])
    ccorp[[j]] <- gsub("Ã¡", "a", ccorp[[j]])
    ccorp[[j]] <- gsub("â€", " ", ccorp[[j]])
  }
  ccorp <- tm_map(ccorp, removeNumbers)
  ccorp <- tm_map(ccorp, tolower)
  ccorp <- tm_map(ccorp, stemDocument)
  ccorp <- tm_map(ccorp, removeWords, allstop)
  ccorp <- tm_map(ccorp, stripWhitespace)
  ccorp <- tm_map(ccorp, PlainTextDocument)

  return(ccorp)
```

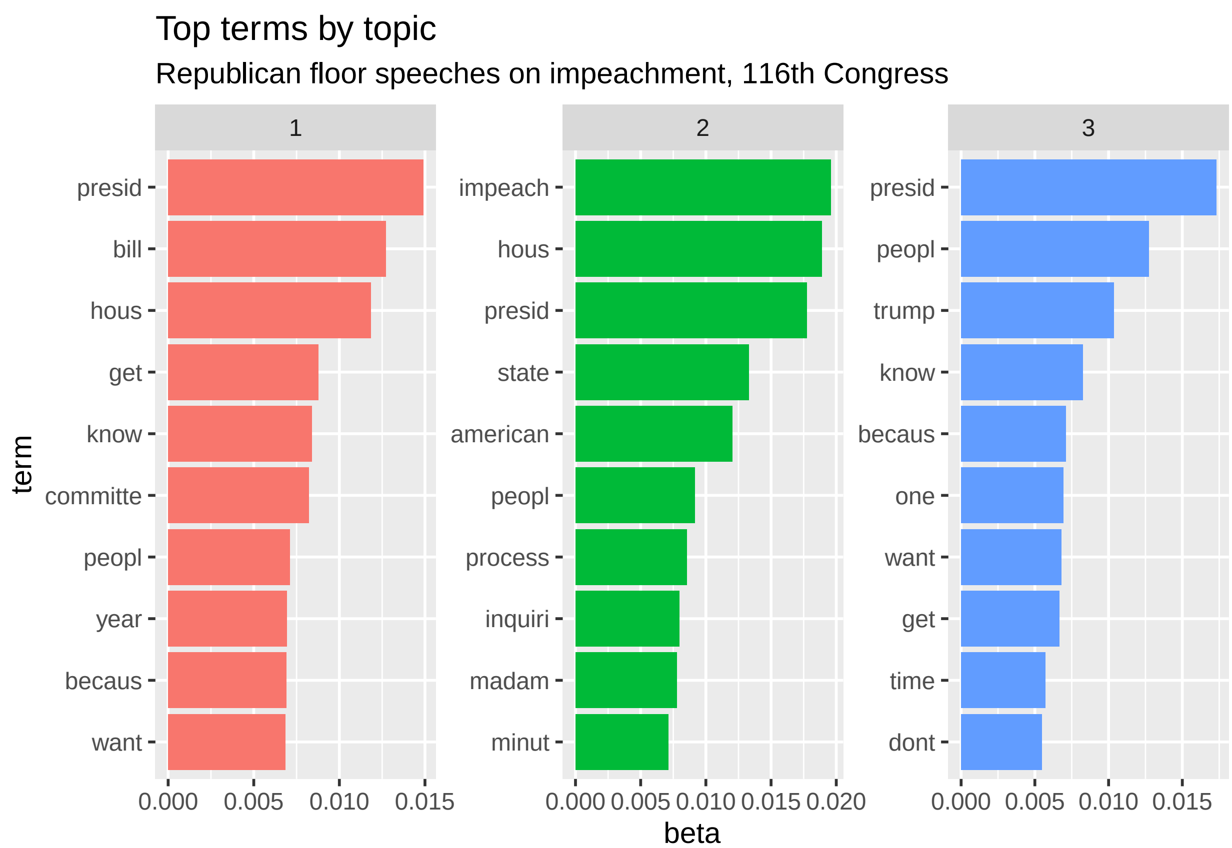
```
}

# Create document term matrix for each party's corpus
dem_dtm <- DocumentTermMatrix(cleanCorpus(demC))
rep_dtm <- DocumentTermMatrix(cleanCorpus(repC))
```

Topic models

```
# These take about 80 seconds to train
dem_t3 <- topicmodels::LDA(dem_dtm, k = 3, control = list(seed = 101))
dem_topics <- tidy(dem_t3, matrix = "beta")

dem_top_terms <- dem_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, desc(beta))

dem_top_terms_plot <- dem_top_terms %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  coord_flip() +
  scale_x_reordered() +
  labs(title = "Top terms by topic",
       subtitle = "Democratic floor speeches on impeachment, 116th Congress")
dem_top_terms_plot
```




```r
# These take about 80 seconds to run
rep_t3 <- topicmodels::LDA(rep_dtm, k = 3, control = list(seed = 101))
rep_topics <- tidy(rep_t3, matrix = "beta")

rep_top_terms <- rep_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, desc(beta))

rep_top_terms_plot <- rep_top_terms %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~topic, scales = "free") +
  coord_flip() +
  scale_x_reordered() +
  labs(title = "Top terms by topic",
       subtitle = "Republican floor speeches on impeachment, 116th Congress")
rep_top_terms_plot
```



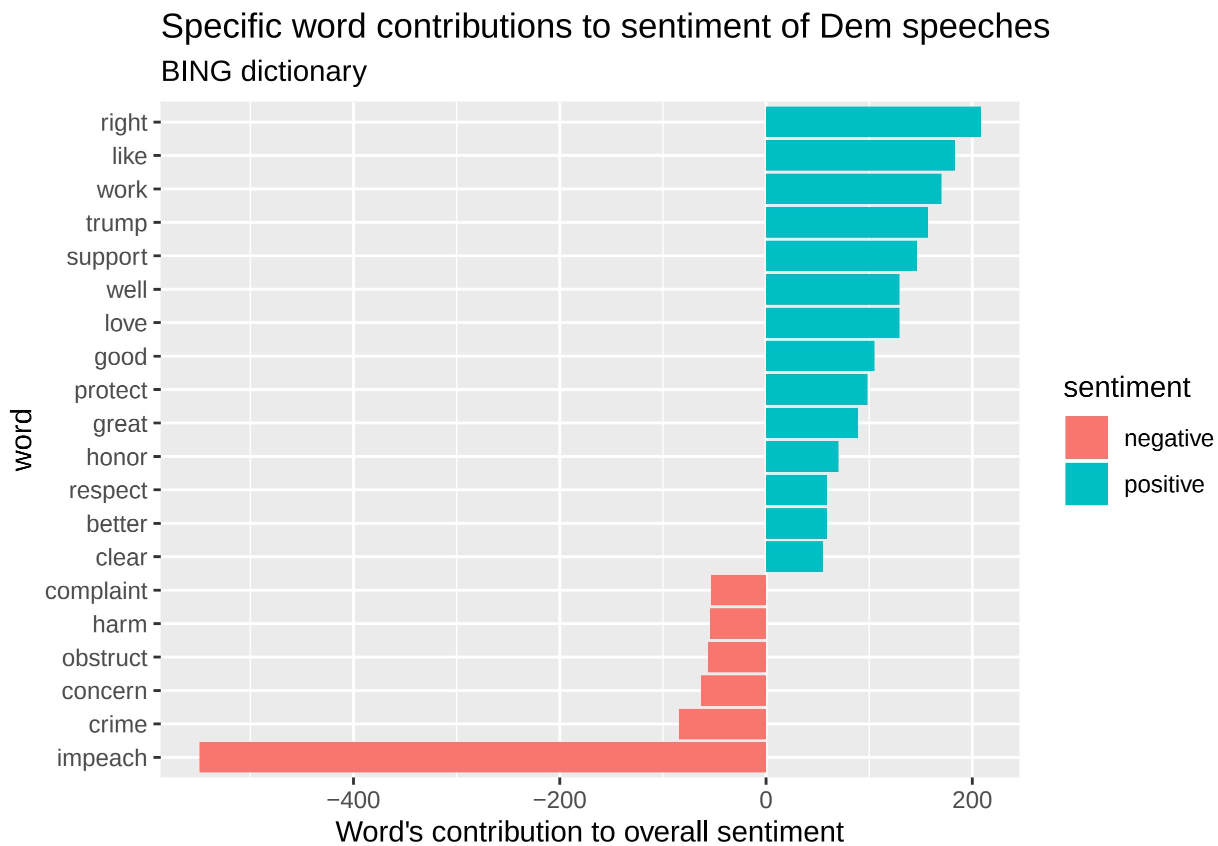
Sentiment analysis

```
freq_dem <- sort(colSums(as.matrix(dem_dtm)), decreasing=TRUE)
freq_dem_t <- tibble("word" = names(freq_dem), "n" = freq_dem)

bing <- get_sentiments("bing")
afinn <- get_sentiments("afinn")


# BING sentiment analysis
dem_bing <- freq_dem_t %>%
  inner_join(bing, by = "word") %>%
  mutate(ntone = ifelse(sentiment == "positive", n, -n)) %>%
  summarize(total_tone = sum(ntone),
            total_words = sum(n))

dem_bing$total_tone / dem_bing$total_words
```

```
## [1] 0.005092761
```

```
freq_dem_t %>%
  inner_join(bing, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
```

```r
  mutate(n = ifelse(sentiment == "positive", n, -n),
         word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Dem speeches",
       subtitle = "BING dictionary")
```

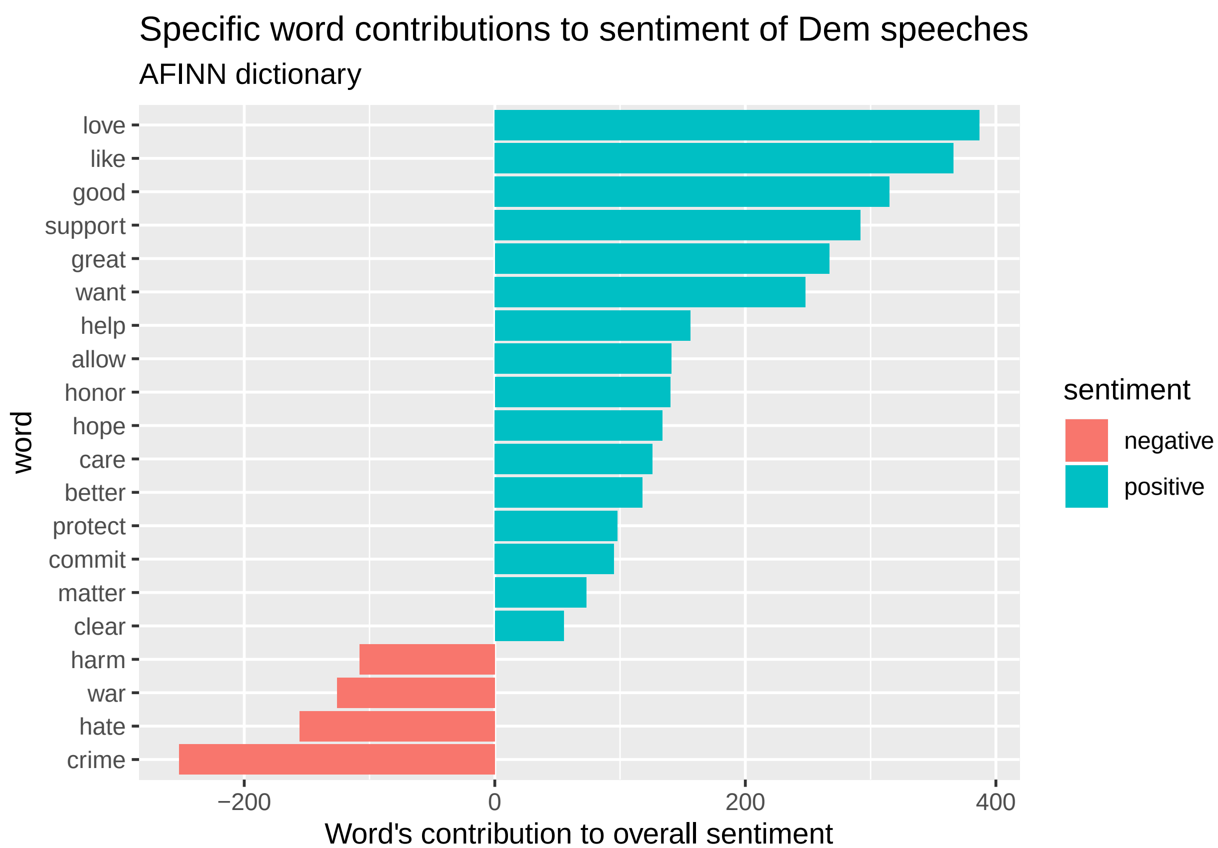



```r
# AFINN sentiment analysis
dem_afinn <- freq_dem_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(score = n*value) %>%
  summarize(totscore = sum(score),
            totwords = sum(n))

dem_afinn$totscore / dem_afinn$totwords
```

```
## [1] 0.3823079
```

```r
freq_dem_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
```

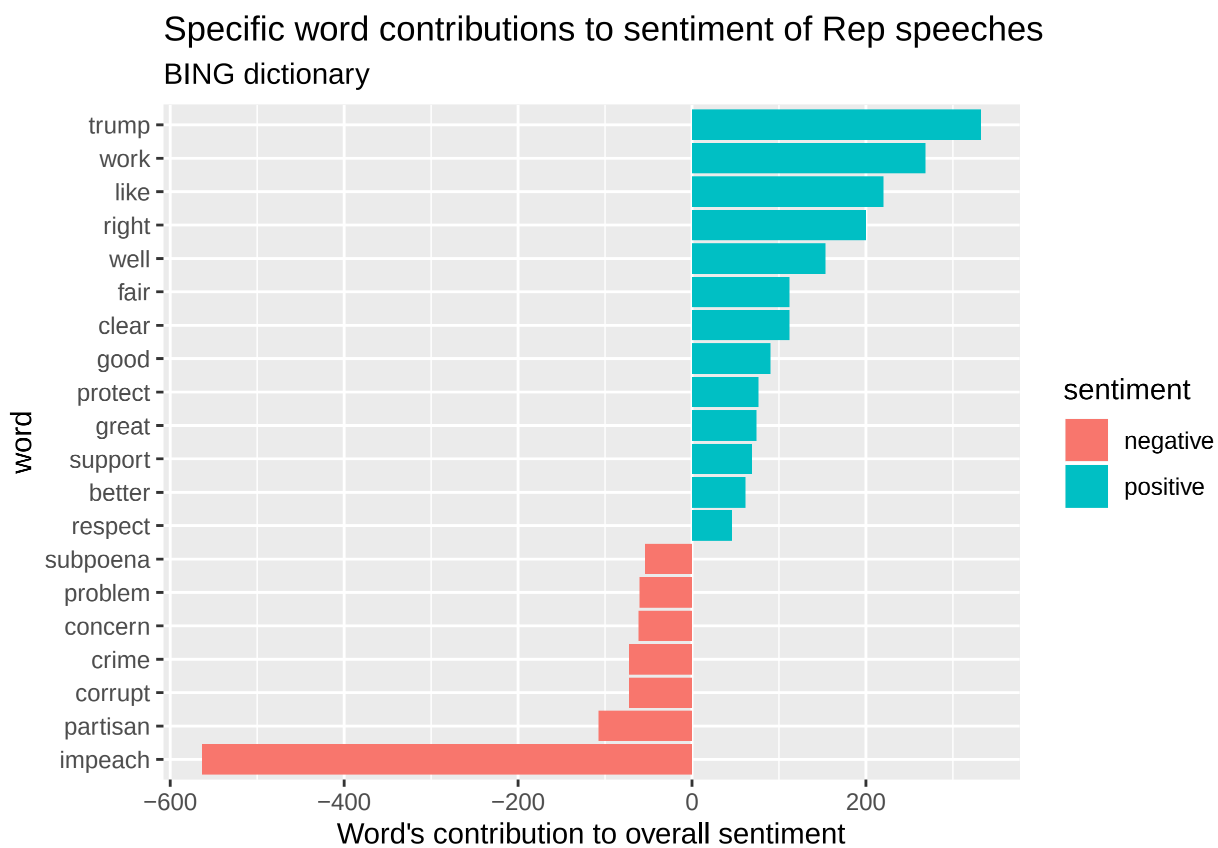
```r
  mutate(score = n*value,
         sentiment = ifelse(score >= 0, "positive", "negative"),
         word = reorder(word, score)) %>%
  ggplot(aes(word, score, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Dem speeches",
       subtitle = "AFINN dictionary")
```




```r
# Repeat sentiment analysis process for Rep
freq_rep <- sort(colSums(as.matrix(rep_dtm)), decreasing=TRUE)
freq_rep_t <- tibble("word" = names(freq_rep), "n" = freq_rep)


# BING sentiment analysis
rep_bing <- freq_rep_t %>%
  inner_join(bing, by = "word") %>%
  mutate(ntone = ifelse(sentiment == "positive", n, -n)) %>%
  summarize(total_tone = sum(ntone),
            total_words = sum(n))

rep_bing$total_tone / rep_bing$total_words
```

```
## [1] 0.01192879
```

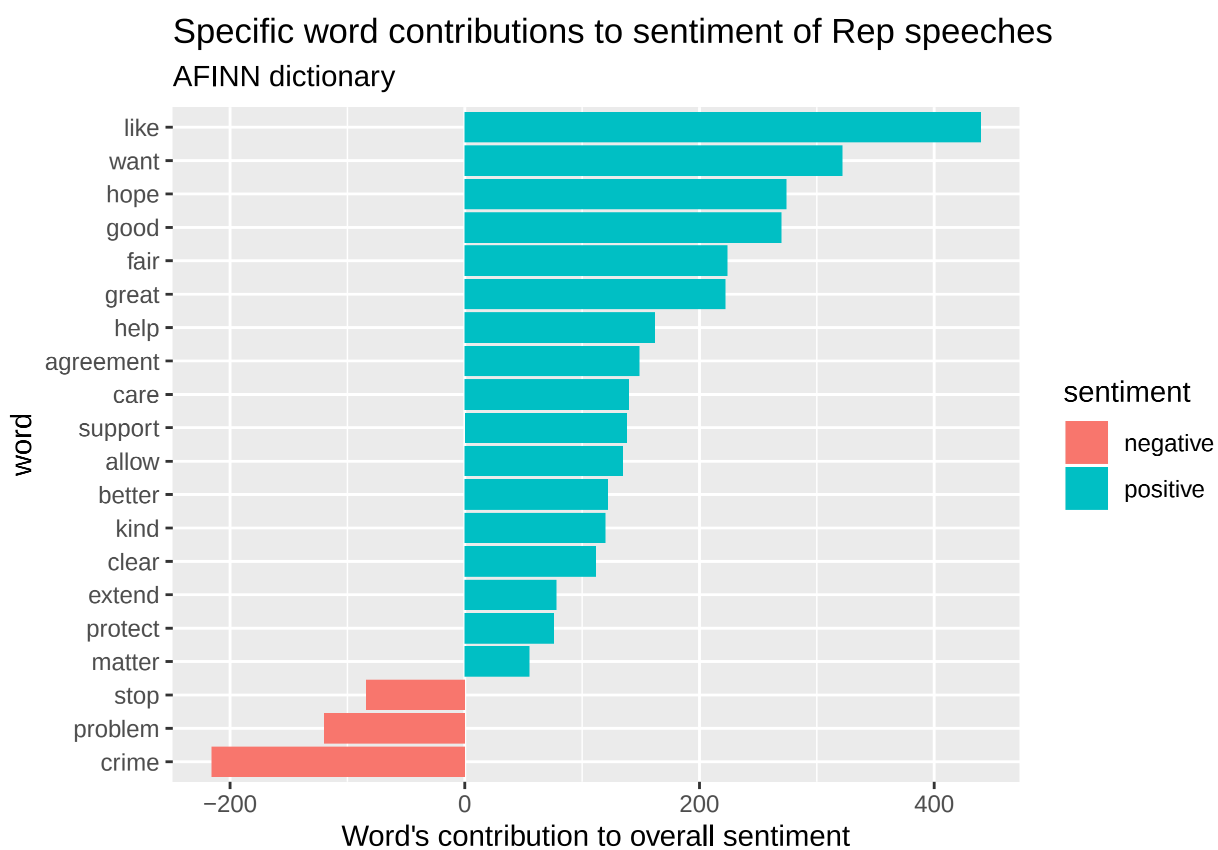
```r
freq_rep_t %>%
  inner_join(bing, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
  mutate(n = ifelse(sentiment == "positive", n, -n),
         word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Rep speeches",
       subtitle = "BING dictionary")
```




```r
# AFINN sentiment analysis
rep_afinn <- freq_rep_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(score = n*value) %>%
  summarize(totscore = sum(score),
            totwords = sum(n))

rep_afinn$totscore / rep_afinn$totwords
```

```
## [1] 0.4393333
```

```r
freq_rep_t %>%
  inner_join(afinn, by = "word") %>%
  mutate(rank = seq_along(word)) %>%
  filter(rank <= 20) %>%
  mutate(score = n*value,
         sentiment = ifelse(score >= 0, "positive", "negative"),
         word = reorder(word, score)) %>%
  ggplot(aes(word, score, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Word's contribution to overall sentiment",
       title = "Specific word contributions to sentiment of Rep speeches",
       subtitle = "AFINN dictionary")
```



Part 3: Statistical analysis and prediction

```r
# Load and clean DW-NOMINATE
dwn <- read_csv("Data/dw-nominate/Hall_members.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   chamber = col_character(),
```

```
##   state_abbrev = col_character(),
##   bioname = col_character(),
##   bioguide_id = col_character(),
##   conditional = col_logical()
## )

## See spec(...) for full column specifications.
```

```
dwn116 <- dwn %>%
  filter(congress == 116 & chamber == "House") %>%
  select(state_abbrev, district_code, bioname, nominate_dim1, nominate_dim2) %>%
  rename(state = state_abbrev, district = district_code) %>%
  separate(bioname, into = c("lastname", "rest"), by = ",")
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 160 rows [1, 6,
## 8, 9, 14, 15, 21, 24, 26, 31, 35, 49, 50, 52, 53, 55, 57, 67, 68, 74, ...].
```

```
# Load and clean DIME
dime <- read_csv("Data/dime/dime_cong_elections_current.csv",
                 col_types = cols(gpct = col_double(),
                                  ppct = col_double(),
                                  gwinner = col_character()))

# Filter DIME to 2018 cycle, corresponding to 116th house
dime116 <- dime %>%
  filter(cycle == 2018 & seat == "federal:house") %>%
  select(Name, state, district, recipient_cfscore) %>%
  mutate(district = as.integer(substr(district, 3, 4))) %>%
  separate(Name, into = c("lastname", "rest"), by = ",") %>%
  select(-rest)
```

```
## Warning: NAs introduced by coercion

## Warning: Expected 2 pieces. Additional pieces discarded in 1764 rows [3,
## 6, 8, 11, 14, 15, 16, 18, 22, 23, 25, 26, 32, 33, 37, 38, 40, 41, 42,
## 43, ...].
```

```
# Join DW-NOMINATE and DIME scores to speeches
impeach116analysis <- impeach116hgrp %>%
  mutate(dem = ifelse(party == "D", 1, 0)) %>%
  left_join(dwn116, by = c("lastname", "state", "district")) %>%
  left_join(dime116, by = c("lastname", "state", "district"))

# Check observations where first names don't match: robustness check for merge
# These all appear ok - just nicknames highlighted (e.g. "THEODORE" != "Ted")
# No need to update or change anything further here
problems <- impeach116analysis %>%
  filter(toupper(rest) != firstname) %>%
  select(-text, -speakerid)

# Create quanteda corpus object
analysis.corp <- quanteda::corpus(impeach116analysis)
summary(analysis.corp)
```

```
## Corpus consisting of 105 documents, showing 100 documents:
##
##     Text Types Tokens Sentences speakerid      lastname firstname state
##    text1   123    192        11         1       ABRAHAM     RALPH    LA
##    text2   143    233         9         2      ADERHOLT    ROBERT    AL
##    text3   834   6555       237         3       AGUILAR      PETE    CA
##    text4   233    487        21         4         ALLEN      RICK    GA
##    text5   418    996        42         5     ARRINGTON     JODEY    TX
##    text6   138    232        13         6         BABIN     BRIAN    TX
##    text7   119    187        11         7         BACON       DON    NE
##    text8   181    474        21         8          BARR      ANDY    KY
##    text9   125    215        10         9        BISHOP       ROB    UT
##   text10  1142   6624       336        10      BONAMICI   SUZANNE    OR
##   text11   343    767        36        11        BROOKS     SUSAN    IN
##   text12  1184   6477       306        12         BROWN   ANTHONY    MD
##   text13   325    747        30        13          BUDD       TED    NC
##   text14  1183   5154       255        14         BYRNE   BRADLEY    AL
##   text15   151    251        15        15        CARTER     BUDDY    GA
##   text16   178    341        13        16     CICILLINE     DAVID    RI
##   text17   139    246         8        17         CLINE       BEN    VA
##   text18   196    459        22        18         COHEN     STEVE    TN
##   text19   495   1679        70        19         COMER     JAMES    KY
##   text20   141    291        15        20       CONAWAY      MIKE    TX
##   text21   385   1136        38        21      COURTNEY       JOE    CT
##   text22  1238   6741       358        22           COX        TJ    CA
##   text23  1240   6421       322        23      CUMMINGS    ELIJAH    MD
##   text24    96    159         9        24       DEFAZIO     PETER    OR
##   text25   153    241        14        25    DESJARLAIS     SCOTT    TN
##   text26   135    222        11        26        DUNCAN      JEFF    SC
##   text27   155    251        10        27          DUNN      NEAL    FL
##   text28   138    229         8        28          FOXX  VIRGINIA    NC
##   text29   459   1114        53        29         FUDGE    MARCIA    OH
##   text30   121    210        12        30       FULCHER      RUSS    ID
##   text31   360    889        43        31         GAETZ      MATT    FL
##   text32   281    564        22        32       GALLEGO     RUBEN    AZ
##   text33  1095   4631       212        33        GARCIA    SYLVIA    TX
##   text34  1095   4631       212        33        GARCIA    SYLVIA    TX
##   text35  1095   4631       212        33        GARCIA    SYLVIA    TX
##   text36   117    210        13        34     GIANFORTE      GREG    MT
##   text37  3529  31386      1280        35       GOHMERT     LOUIE    TX
##   text38  3041  46843      2462        36         GREEN        AL    TX
##   text39   242    513        27        37         GREEN      MARK    TN
##   text40   100    171        15        38     GRIFFITH    MORGAN    VA
##   text41  1088   5120       201        39      GROTHMAN     GLENN    WI
##   text42   227   1079        47        40         GUEST      MIKE    MS
##   text43   857   4357       247        41       HAALAND       DEB    NM
##   text44   230    423        23        42      HARTZLER     VICKY    MS
##   text45   149    234        13        43          HERN     KEVIN    OK
##   text46   113    169         9        44       HIGGINS     BRIAN    NY
##   text47   124    185         9        45       HIGGINS      CLAY    LA
##   text48  1821   8909       241        46          HILL    FRENCH    AR
##   text49   536   1571        55        47          HILL     KATIE    CA
##   text50  1440   7157       341        48         HIMES       JIM    CT
##   text51   115    191        12        49 HOLMES NORTON   ELEANOR    DC
```

```
##    text52   287    786        45         50        HORN    KENDRA    OK
##    text53  1345   6668       291         51    HORSFORD    STEVEN    NV
##    text54   292    859        27         52       HOYER     STENY    MD
##    text55  1273   4922       193         53 JACKSON LEE    SHEILA    TX
##    text56   239    532        33         54    JEFFRIES    HAKEEM    NY
##    text57   153    242        12         55     JOHNSON      BILL    OH
##    text58   184    326        16         56     JOHNSON      MIKE    LA
##    text59   135    277        18         57      JORDAN       JIM    OH
##    text60   268    649        31         58       JOYCE      JOHN    PA
##    text61  2024   9448       410         59      KAPTUR     MARCY    OH
##    text62   397   1053        41         60      KELLER      FRED    PA
##    text63   346    886        48         61       KELLY     TRENT    MS
##    text64  1801  12621       497         62        KING     STEVE    IA
##    text65   139    239        12         63 KIRKPATRICK       ANN    AZ
##    text66   574   1597        77         64     LAMALFA      DOUG    CA
##    text67  1307   6749       304         65    LAWRENCE    BRENDA    MI
##    text68   225    503        29         66       LEWIS      JOHN    GA
##    text69   145    270        12         67  LOUDERMILK     BARRY    GA
##    text70  1245   4174       192         68    MARSHALL     ROGER    KS
##    text71   391    927        47         69    MCCARTHY     KEVIN    CA
##    text72   508   1744        82         70      MEUSER       DAN    PA
##    text73   128    219        13         71      MILLER     CAROL    WV
##    text74   275    549        25         72    MITCHELL      PAUL    MI
##    text75   158    255         9         73   MOOLENAAR      JOHN    MI
##    text76   149    275         9         74      MOONEY      ALEX    WV
##    text77  1217   6671       310         75      NADLER     JERRY    NY
##    text78   136    222        12         76    NEWHOUSE       DAN    WA
##    text79   167    387        26         77      NORMAN     RALPH    SC
##    text80   340    747        49         78       OLSON      PETE    TX
##    text81   150    254        13         79      PALMER      GARY    AL
##    text82    99    185         8         80       PERRY     SCOTT    PA
##    text83   135    238        12         81        ROSE      JOHN    TN
##    text84   129    204        11         82      ROUZER     DAVID    NC
##    text85   408    909        41         83  RUTHERFORD      JOHN    FL
##    text86  3154  56425      2825         84    SCALISE     STEVE    LA
##    text87   776   3845       196         85     SCANLON      MARY    PA
##    text88   743   2809       145         86  SCHAKOWSKY       JAN    IL
##    text89   159    291        14         87       SCOTT    AUSTIN    GA
##    text90   150    762        48         88       SMITH    ADRIAN    NE
##    text91   149    238        11         89     SMUCKER     LLOYD    PA
##    text92   337    767        35         90       SPANO      ROSS    FL
##    text93   157    279        11         91     STAUBER      PETE    MN
##    text94   259    557        38         92       STEIL     BRYAN    WI
##    text95   240    461        21         93    THOMPSON     GLENN    PA
##    text96  1306   6040       243         94       TLAIB   RASHIDA    MI
##    text97   148    231        12         95     WALBERG       TIM    MI
##    text98   143    235        13         96      WALKER      MARK    NC
##    text99   140    227        14         97    WALORSKI    JACKIE    IN
##   text100   125    223        11         98     WATKINS     STEVE    KS
##   party district dem    rest nominate_dim1 nominate_dim2 recipient_cfscore
##       R        5   0   Ralph         0.527         0.301             1.037
##       R        4   0  Robert         0.366         0.594             0.907
##       D       31   1   Peter        -0.289         0.130            -1.187
##       R       12   0    Rick         0.674         0.294             1.324
```

```
##       R       19   0     Jodey          0.624         -0.038            1.319
##       R       36   0     Brian          0.686          0.164            1.139
##       R        2   0    Donald          0.421         -0.127            1.158
##       R        6   0   Garland          0.495          0.278            0.921
##       R        1   0    Robert          0.536          0.099            0.786
##       D        1   1   Suzanne         -0.393         -0.400           -1.105
##       R        5   0     Susan          0.362          0.222            1.081
##       D        4   1   Anthony         -0.342         -0.159           -0.954
##       R       13   0  Theodore          0.672         -0.079            1.212
##       R        1   0   Bradley          0.606          0.253            0.992
##       R        1   0     Buddy          0.586          0.274            1.038
##       D        1   1     David         -0.390         -0.290           -1.086
##       R        6   0  Benjamin          0.723         -0.227            1.131
##       D        9   1   Stephen         -0.400         -0.354           -0.294
##       R        1   0     James          0.643         -0.050            0.829
##       R       11   0         K          0.591          0.365            1.217
##       D        2   1       Joe         -0.344          0.015           -1.109
##       D       21   1        TJ         -0.312          0.370           -1.215
##       D        7   1    Elijah         -0.438         -0.148           -0.743
##       D        4   1      <NA>             NA             NA           -0.742
##       R        4   0     Scott          0.578         -0.096            1.304
##       R        3   0      Jeff          0.730         -0.195            1.125
##       R        2   0      Neal          0.542          0.167            1.011
##       R        5   0  Virginia          0.631          0.119            0.975
##       D       11   1    Marcia         -0.580          0.140           -0.527
##       R        1   0   Russell          0.623          0.143            1.449
##       R        1   0   Matthew          0.626         -0.554            0.891
##       D        7   1     Ruben         -0.451         -0.022           -0.817
##       D       29   1    Sylvia         -0.771          0.559               NA
##       D       29   1    Sylvia         -0.771          0.559           -0.830
##       D       29   1    Sylvia         -0.771          0.559           -0.369
##       R        1   0      Greg          0.423          0.100            1.381
##       R        1   0     Louie          0.622         -0.328            1.342
##       D        9   1        Al         -0.438          0.316           -0.454
##       R        7   0      Mark          0.749          0.176            1.248
##       R        9   0         H          0.550         -0.436            0.914
##       R        6   0     Glenn          0.617         -0.288            1.309
##       R        3   0   Michael          0.470          0.251            1.232
##       D        1   1     Debra         -0.292         -0.326           -1.471
##       R        4   0      <NA>             NA             NA               NA
##       R        1   0     Kevin          0.686          0.042            1.155
##       D       26   1     Brian         -0.347         -0.033           -0.473
##       R        3   0      Clay          0.563         -0.081            1.132
##       R        2   0    French          0.452          0.225            0.887
##       D       25   1     Katie         -0.306          0.114           -1.593
##       D        4   1     James         -0.241         -0.245           -1.052
##       D       NA   1      <NA>             NA             NA               NA
##       D        5   1   Kendra          -0.185          0.524           -1.041
##       D        4   1    Steven         -0.351          0.089           -0.853
##       D        5   1     Steny         -0.380          0.116           -0.546
##       D       18   1      <NA>             NA             NA               NA
##       D        8   1    Hakeem         -0.485         -0.092           -0.943
##       R        6   0      Bill          0.432          0.275            0.947
##       R        4   0      Mike          0.556         -0.098            1.315
```

```
##      R      4  0      Jim        0.719       -0.225            1.212
##      R     13  0     John        0.526        0.176            0.954
##      D      9  1   Marcia       -0.350        0.109           -0.541
##      R     12  0     Fred        0.476        0.242               NA
##      R      1  0    Trent        0.629        0.233            1.151
##      R      4  0    Steve        0.610        0.204            1.335
##      D      2  1      Ann       -0.161       -0.008           -1.417
##      R      1  0     <NA>           NA           NA            0.826
##      D     14  1   Brenda       -0.445       -0.040           -0.567
##      D      5  1     John       -0.589       -0.223           -0.999
##      R     11  0    Barry        0.688       -0.020            1.309
##      R      1  0    Roger        0.526        0.194            1.114
##      R     23  0    Kevin        0.458        0.208            0.799
##      R      9  0      Dan        0.583        0.469            0.684
##      R      3  0    Carol        0.453        0.387            0.916
##      R     10  0     Paul        0.432        0.302            0.936
##      R      4  0     John        0.402        0.445            0.972
##      R      2  0     Alex        0.579       -0.295            1.127
##      D     10  1  Jerrold       -0.508       -0.508           -0.823
##      R      4  0   Daniel        0.350        0.222            0.665
##      R      5  0    Ralph        0.781       -0.169            1.322
##      R     22  0     Pete        0.548        0.309            1.120
##      R      6  0     Gary        0.715       -0.046            1.234
##      R     10  0    Scott        0.627       -0.350            1.014
##      R      6  0     John        0.703        0.141            0.981
##      R      7  0    David        0.575        0.205            1.001
##      R      4  0     John        0.397        0.159            0.972
##      R      1  0    Steve        0.563        0.151            0.977
##      D      5  1     Mary       -0.485       -0.042           -1.217
##      D      9  1   Janice       -0.607       -0.267           -1.313
##      R      8  0   Austin        0.572        0.167            1.177
##      R      3  0   Adrian        0.516        0.182            1.104
##      R     11  0    Lloyd        0.415        0.302            1.063
##      R     15  0     Ross        0.451       -0.010            1.200
##      R      8  0    Peter        0.293        0.210            1.318
##      R      1  0    Bryan        0.418       -0.069            1.333
##      R     15  0    Glenn        0.308        0.386            0.958
##      D     13  1  Rashida       -0.315       -0.949           -1.260
##      R      7  0      Tim        0.520        0.065            1.226
##      R      6  0  Bradley        0.642        0.020            1.149
##      R      2  0   Jackie        0.431        0.317            1.227
##      R      2  0    Steve        0.559        0.059            1.033
##
## Source: C:/Users/Alec/Documents/Academics/Second Year/Fall Quarter/MACS 40500 - Computational Methods
## Created: Tue Dec 10 02:05:09 2019
## Notes:
```

```r
# Create document frequency matrix
dfmat_116 <- dfm(analysis.corp, tolower = TRUE, stem = TRUE, remove_punct = TRUE,
                 remove = allstop)

# Trim the matrix to include only terms that occur at least 3 times to ensure convergence
dfmat_116 <- dfm_trim(dfmat_116, min_termfreq = 3, termfreq_type = "count")
```

```r
set.seed(100)
id_train <- sample(1:105, 81, replace=FALSE)

# Create ID variable to subset train/test
docvars(analysis.corp, "id_numeric") <- 1:ndoc(analysis.corp)

# Train set
dfmat_training <- corpus_subset(analysis.corp, id_numeric %in% id_train) %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
  dfm_trim(min_termfreq=3, termfreq_type="count")

# Test set
dfmat_testing <- corpus_subset(analysis.corp, !(id_numeric %in% id_train)) %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
  dfm_trim(min_termfreq=3 , termfreq_type="count")

# Train Naive Bayes classifier
tmod_nb <- textmodel_nb(dfmat_training, docvars(dfmat_training, "dem"))
summary(tmod_nb)
```

```
##
## Call:
## textmodel_nb.dfm(x = dfmat_training, y = docvars(dfmat_training,
##     "dem"))
##
## Class Priors:
## (showing first 2 elements)
##   0   1
## 0.5 0.5
##
## Estimated Feature Scores:
##   violenc  women reauthor    act general   leav  nadler unanim consent
## 0 0.06797 0.2975   0.0707 0.2505  0.4369 0.3335 0.04539  0.462  0.4669
## 1 0.93203 0.7025   0.9293 0.7495  0.5631 0.6665 0.95461  0.538  0.5331
##   member   may legisl    day  revis extend remark  insert extran materi
## 0 0.4789 0.391 0.3558 0.6033 0.7602 0.6584 0.6178 0.09613 0.1035 0.2122
## 1 0.5211 0.609 0.6442 0.3967 0.2398 0.3416 0.3822 0.90387 0.8965 0.7878
##     href   bill     th congress   hous    h.r object request   new   york
## 0 0.3207 0.3999 0.1909   0.3231 0.5106 0.1542 0.4189   0.338 0.328 0.2398
## 1 0.6793 0.6001 0.8091   0.6769 0.4894 0.8458 0.5811   0.662 0.672 0.7602
##   pursuant
## 0   0.1009
## 1   0.8991
```

```r
# Make features identical across train and test sets
dfmat_matched <- dfm_match(dfmat_testing, features = featnames(dfmat_training))

# Now to inspect classification
actuals <- docvars(dfmat_matched, "dem")
predictions <- predict(tmod_nb, newdata = dfmat_matched)
cMat <- table(actuals, predictions)
cMat
```

```
##        predictions
```

```
## actuals  0  1
##       0 18  1
##       1  3  2
```

```r
caret::confusionMatrix(cMat, mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##        predictions
## actuals  0  1
##       0 18  1
##       1  3  2
##
##                Accuracy : 0.8333
##                  95% CI : (0.6262, 0.9526)
##     No Information Rate : 0.875
##     P-Value [Acc > NIR] : 0.8271
##
##                   Kappa : 0.4074
##
##  Mcnemar's Test P-Value : 0.6171
##
##             Sensitivity : 0.8571
##             Specificity : 0.6667
##          Pos Pred Value : 0.9474
##          Neg Pred Value : 0.4000
##               Precision : 0.9474
##                  Recall : 0.8571
##                      F1 : 0.9000
##              Prevalence : 0.8750
##          Detection Rate : 0.7500
##    Detection Prevalence : 0.7917
##       Balanced Accuracy : 0.7619
##
##        'Positive' Class : 0
##
```

```r
# Define sentiment calculation function
sentScore <- function(text, dictname) {
  corp <- cleanCorpus(VCorpus(VectorSource(text)))
  temp_dtm <- DocumentTermMatrix(corp)
  freq <- sort(colSums(as.matrix(temp_dtm)), decreasing=TRUE)

  tib <- tibble("word" = names(freq), "n" = freq)
  dict_ <- get_sentiments(dictname)

  if (dictname=="bing") {
    sent_calc <- tib %>%
      inner_join(dict_, by="word") %>%
      mutate(ntone = ifelse(sentiment=="positive", n, -n)) %>%
      summarize(total_tone=sum(ntone),
                total_words=sum(n))
  } else if (dictname=="afinn") {
```

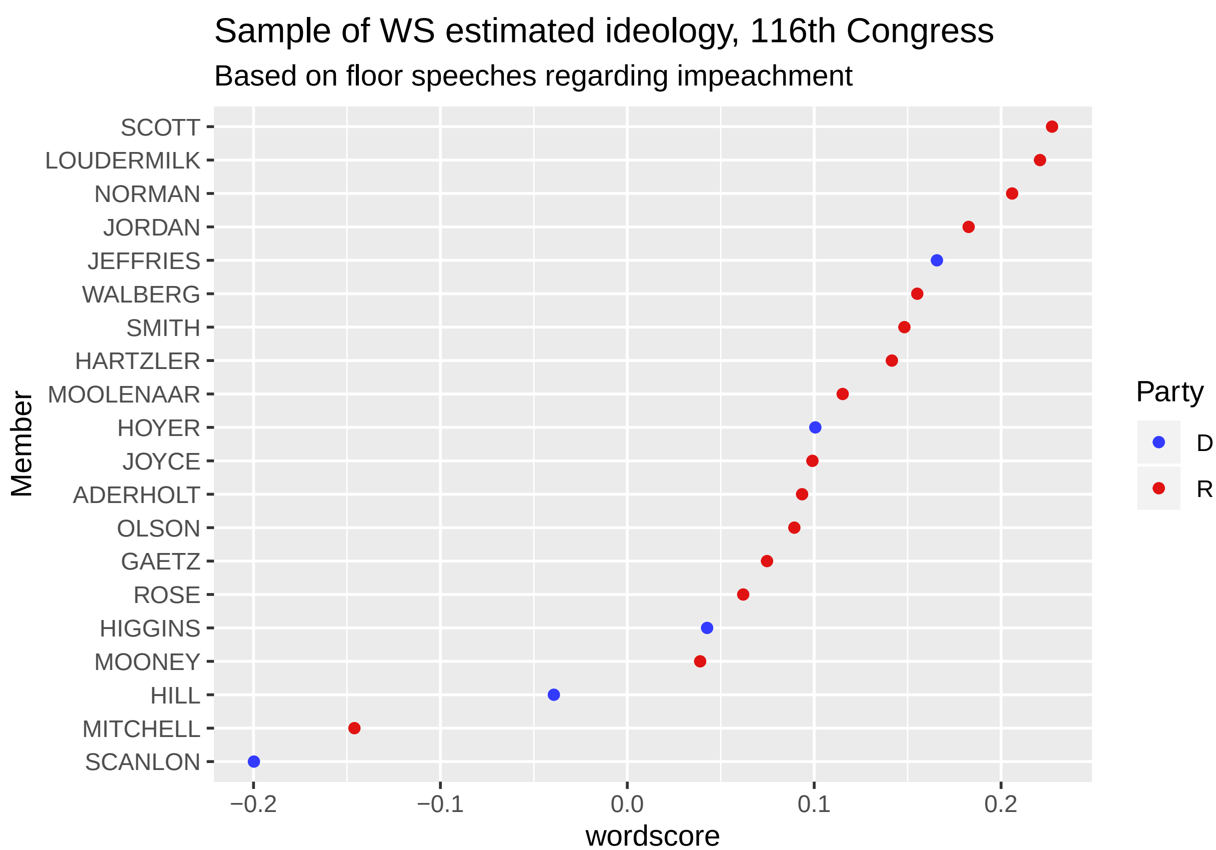
```r
    sent_calc <- tib %>%
      inner_join(dict_, by="word") %>%
      mutate(score=n*value) %>%
      summarize(total_tone=sum(score),
                total_words=sum(n))
  }

  return(sent_calc$total_tone/sent_calc$total_words)
}
```

```r
# Attach sentiment scores to members' speeches
bing <- rep(NA, 105)
afinn <- rep(NA, 105)
for (i in 1:105) {
  bing[[i]] <- sentScore(impeach116analysis$text[[i]], "bing")
  afinn[[i]] <- sentScore(impeach116analysis$text[[i]], "afinn")
}
```

```r
# Train wordscores
dfmat_all <- analysis.corp %>%
  dfm(tolower=TRUE, stem=TRUE, remove_punct=TRUE, remove=allstop) %>%
  dfm_trim(min_termfreq=3, termfreq_type="count")

# We'll use Al Green (-1) and Steve Scalise (+1) as anchors for wordscore training
reference.scores <- c(rep(NA, 37), -1, rep(NA, 47), 1, rep(NA, 19))

# Train wordscore model and attach predicted scores to names
ws.model <- textmodel_wordscores(dfmat_all, reference.scores, smooth=1)
ws.full.model <- predict(ws.model, level = 0.95)

# Train wordfish model
wf.full.model <- textmodel_wordfish(dfmat_all, sparse=TRUE)

# Train 2D correspondence analysis
ca <- textmodel_ca(dfmat_all)
ca_dim1 <- coef(ca, doc_dim=1)$coef_document
ca_dim2 <- coef(ca, doc_dim=2)$coef_document

# Create df of wordscores with info from the dfm
wswf.df <- tibble(
  firstname = docvars(dfmat_all, "firstname"),
  lastname = docvars(dfmat_all, "lastname"),
  state = docvars(dfmat_all, "state"),
  district = docvars(dfmat_all, "district"),
  party = docvars(dfmat_all, "party"),
  dem = docvars(dfmat_all, "dem"),
  bing = bing,
  afinn = afinn,
  wordscore = ws.full.model,
  wftheta = wf.full.model$theta,
  wfse = wf.full.model$se,
  ca_dim1 = ca_dim1,
  ca_dim2 = ca_dim2,
```

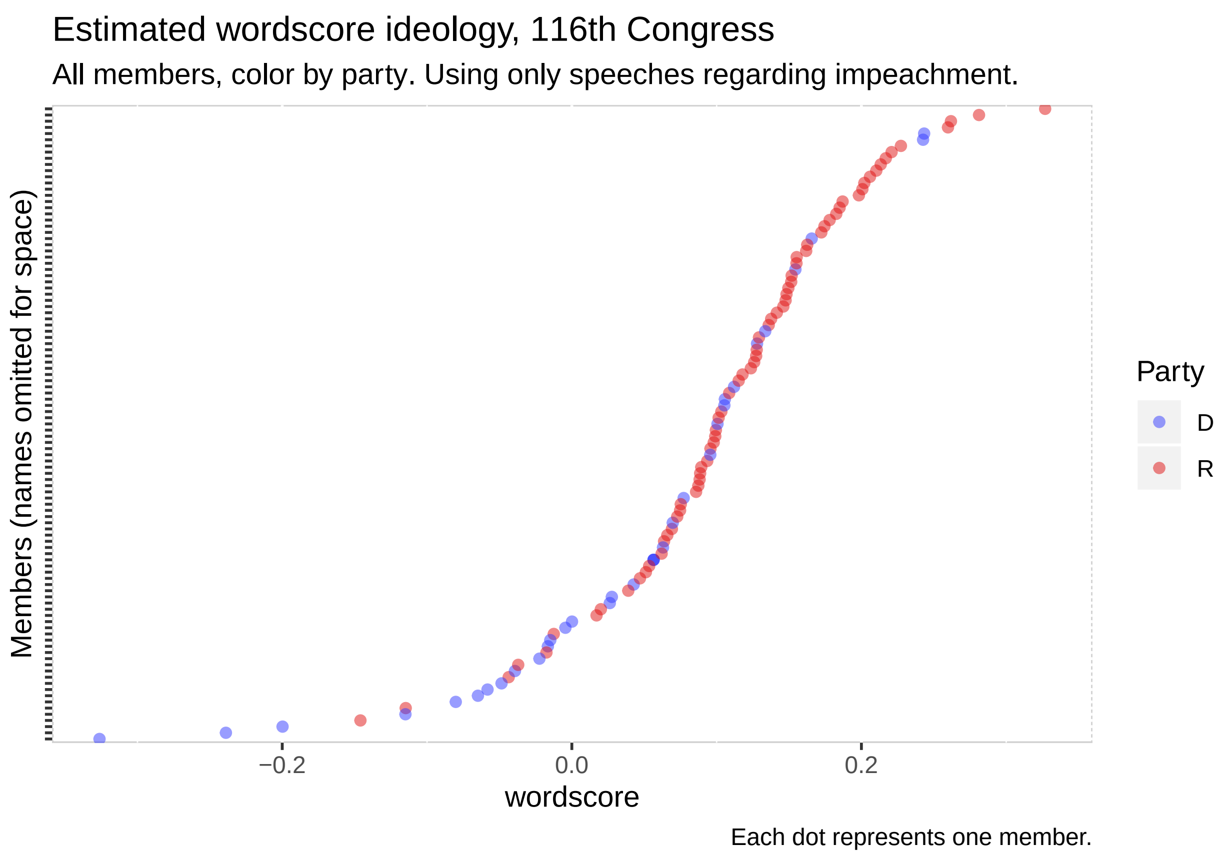
```
  recipient_cfscore = docvars(dfmat_all, "recipient_cfscore"),
  nominate_dim1 = docvars(dfmat_all, "nominate_dim1"),
  nominate_dim2 = docvars(dfmat_all, "nominate_dim2")
)
```

```
set.seed(300)
#update_geom_defaults("point", list(size=1.5))
ws.plot.1 <- wswf.df[sample(nrow(wswf.df), 20),] %>%
  ggplot(., aes(fct_reorder(as.factor(lastname), wordscore),
                wordscore,
                color=party)) +
  geom_point() +
  coord_flip() +
  scale_color_manual(values=group.colors) +
  labs(x = "Member",
       title = "Sample of WS estimated ideology, 116th Congress",
       subtitle = "Based on floor speeches regarding impeachment",
       color = "Party")
ws.plot.1
```




```
ws.plot.2 <- wswf.df %>%
  #filter(party != "I" & wordscore < .2 & wordscore > -.2) %>%
  mutate(fullname = paste0(lastname, firstname)) %>%
  ggplot(., aes(fct_reorder(as.factor(fullname), wordscore),
                wordscore, color=party)) +
```

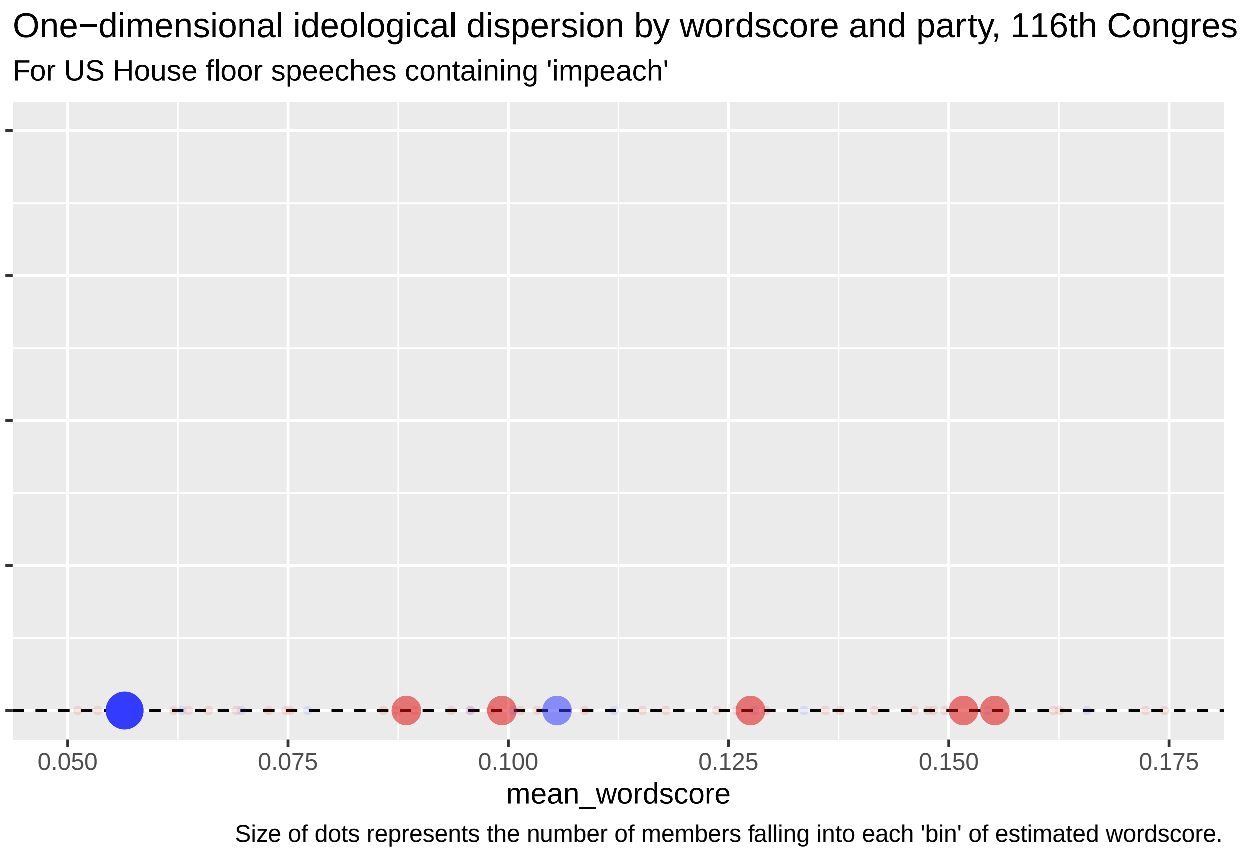
```
  geom_point(alpha=0.5) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  theme(axis.text.y=element_blank(),
        panel.background=element_rect(fill="white",
                                      color="lightgray", size=0.5,
                                      linetype="solid"),
        panel.grid.major=element_line(size=0.5, linetype="solid",
                                      color="white"),
        panel.grid.minor=element_line(size=0.25, linetype="solid",
                                      color="white")) +
  labs(title = "Estimated wordscore ideology, 116th Congress",
       subtitle = "All members, color by party. Using only speeches regarding impeachment.",
       x = "Members (names omitted for space)",
       color = "Party",
       caption = "Each dot represents one member.")
ws.plot.2
```




```
ws.plot.3 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = wordscore - (wordscore %% .001)) %>%
  arrange(bin, wordscore) %>%
  group_by(bin, party) %>%
  summarize(mean_wordscore = mean(wordscore, na.rm=TRUE),
            count = n()) %>%
```

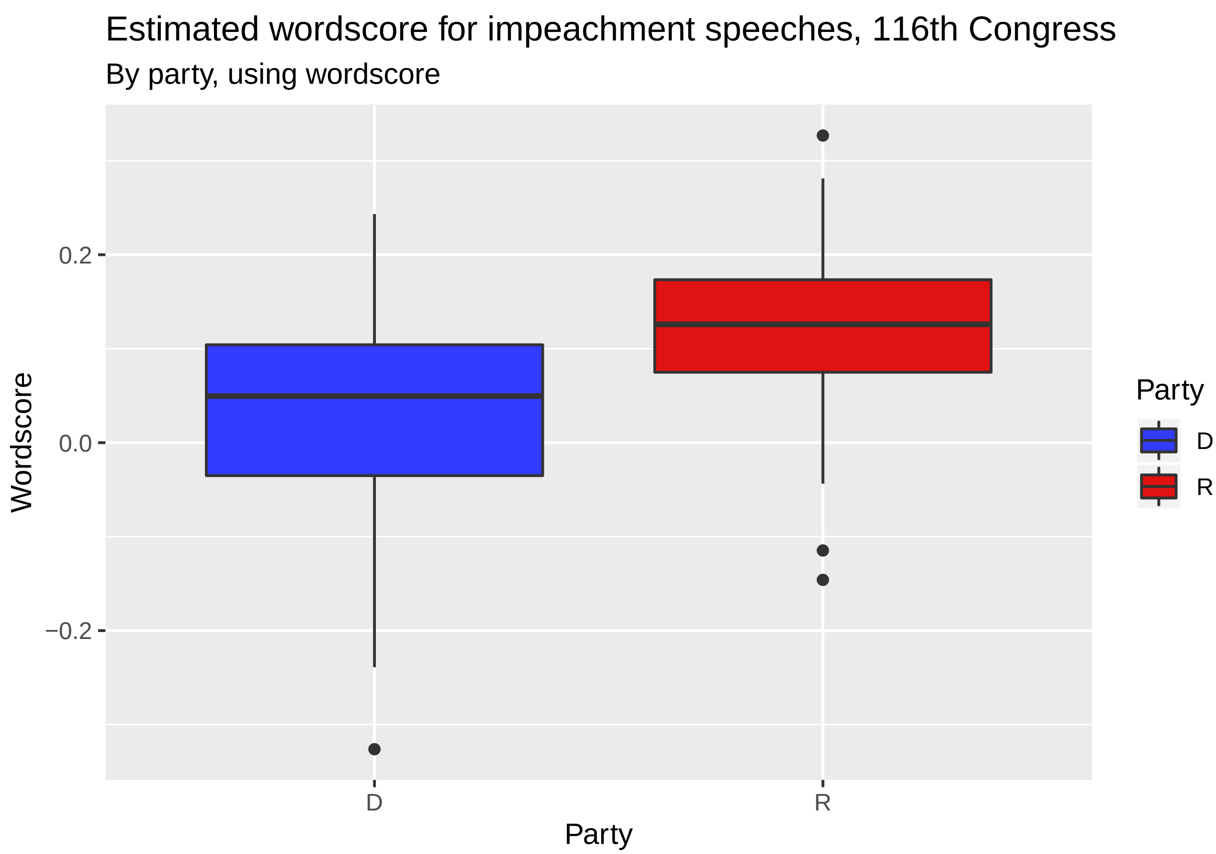
```
  mutate(x = 0) %>%
  ggplot(., aes(x=c(0), mean_wordscore, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(.05, .175) +
  xlim(0, .001) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by wordscore and party, 116th Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated words
       color="Party")
ws.plot.3
```

```
## Warning: Removed 97 rows containing missing values (geom_hline).

## Warning: Removed 46 rows containing missing values (geom_point).
```



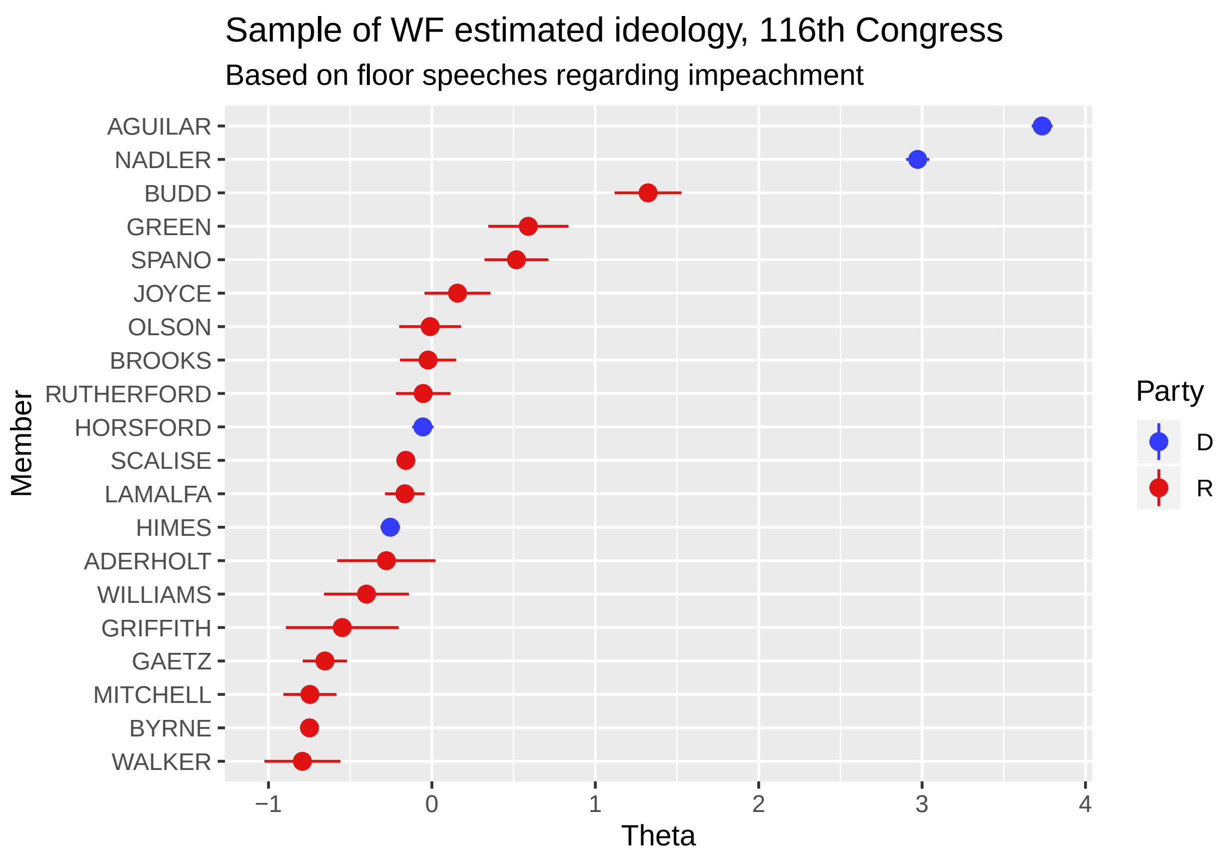

```
# Produce boxplots of wordscore by party
ws.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
```

```r
  ggplot(., aes(x=party, y=wordscore, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  #ylim(c(-.125,.125)) +
  labs(title="Estimated wordscore for impeachment speeches, 116th Congress",
       subtitle="By party, using wordscore",
       fill="Party",
       x = "Party",
       y = "Wordscore")
ws.plot.4
```



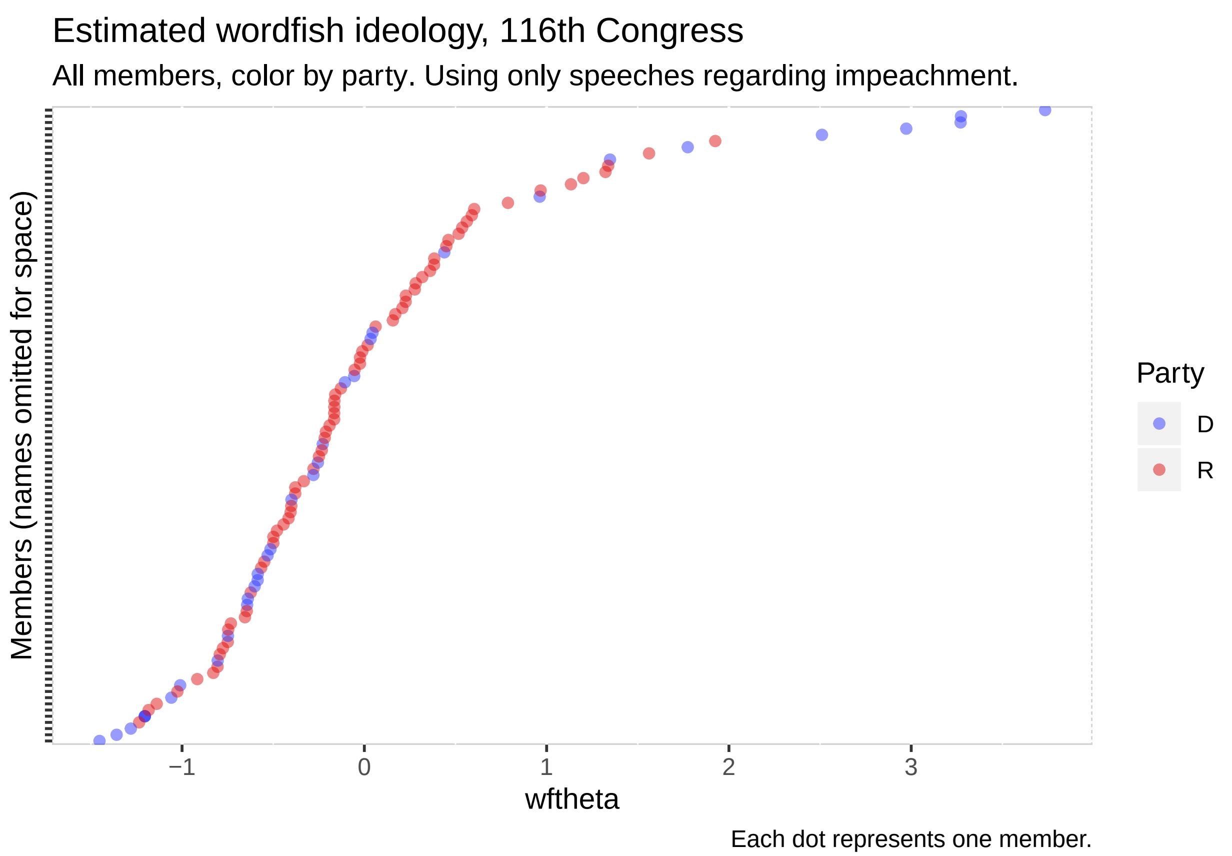

```r
set.seed(1190)
# Random sample of wordfish scores
wf.plot.1 <- wswf.df[sample(nrow(wswf.df), 20),] %>%
  arrange(wftheta) %>%
  ggplot() +
  geom_pointrange(aes(x=fct_reorder(as.factor(lastname), wftheta),
                      y=wftheta,
                      color=party,
                      ymin=wftheta-2*wfse,
                      ymax=wftheta+2*wfse)) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  labs(x = "Member",
       y = "Theta",
```

```
       title = "Sample of WF estimated ideology, 116th Congress",
       subtitle = "Based on floor speeches regarding impeachment",
       color = "Party")
wf.plot.1
```




```
# All members' wordfish scores
wf.plot.2 <- wswf.df %>%
  #filter(party != "I" & wftheta < 4) %>%
  mutate(fullname = paste0(lastname, firstname)) %>%
  ggplot(., aes(fct_reorder(as.factor(fullname), wftheta),
              wftheta, color=party)) +
  geom_point(alpha=0.5) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  theme(axis.text.y=element_blank(),
        panel.background=element_rect(fill="white",
                                      color="lightgray", size=0.5,
                                      linetype="solid"),
        panel.grid.major=element_line(size=0.5, linetype="solid",
                                      color="white"),
        panel.grid.minor=element_line(size=0.25, linetype="solid",
                                      color="white")) +
  labs(title = "Estimated wordfish ideology, 116th Congress",
       subtitle = "All members, color by party. Using only speeches regarding impeachment.",
       x = "Members (names omitted for space)",
```

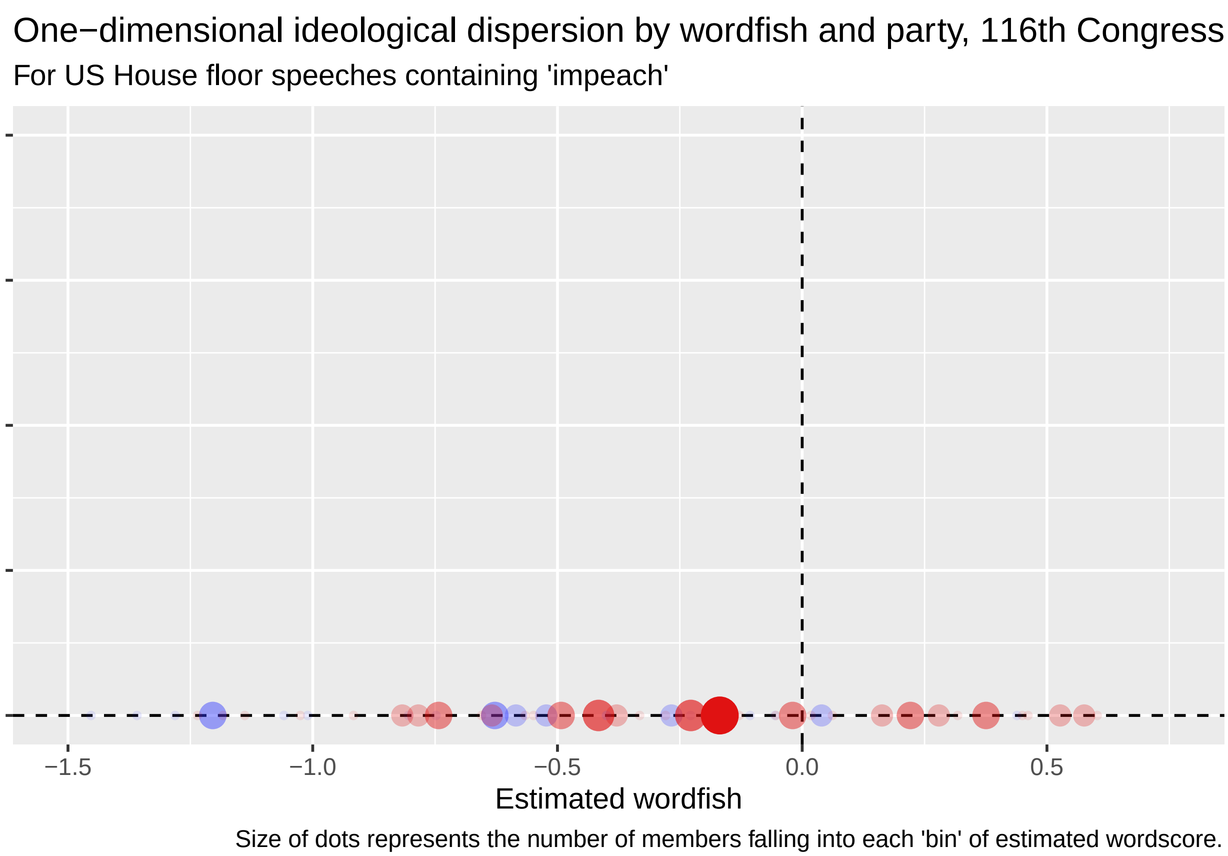
```
       color = "Party",
       caption = "Each dot represents one member.")
wf.plot.2
```



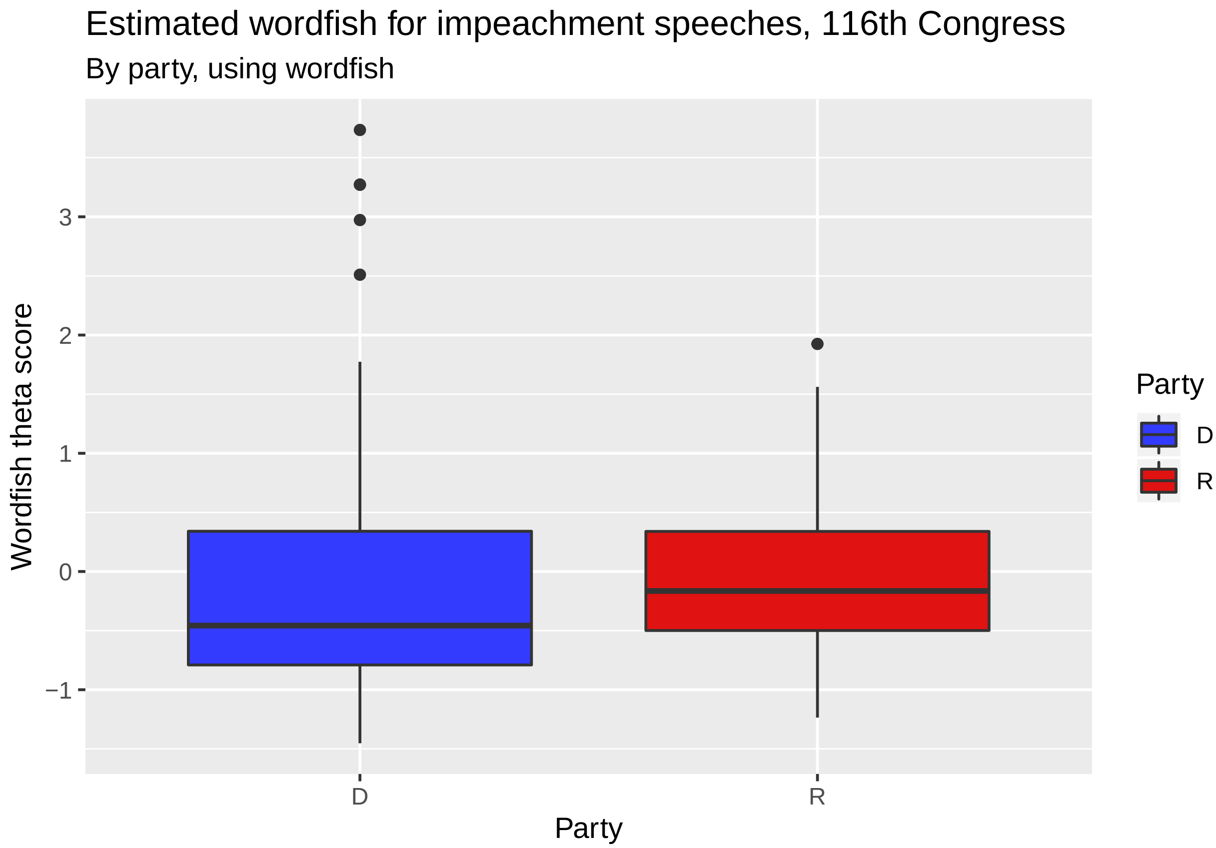

```
#update_geom_defaults("point", list(size=1.5))
wf.plot.3 <- wswf.df %>%
  #filter(party != "I") %>%
  mutate(bin = wftheta - (wftheta %% .05)) %>%
  arrange(bin, wftheta) %>%
  group_by(bin, party) %>%
  summarize(mean_wftheta = mean(wftheta, na.rm=TRUE),
            count = n()) %>%
  mutate(x = 0) %>%
  ggplot(., aes(x=c(0), mean_wftheta, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-1.5, .75)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by wordfish and party, 116th Congress",
       subtitle="For US House floor speeches containing 'impeach'",
```

```
       caption="Size of dots represents the number of members falling into each 'bin' of estimated word
       color="Party",
       y="Estimated wordfish")
wf.plot.3
```

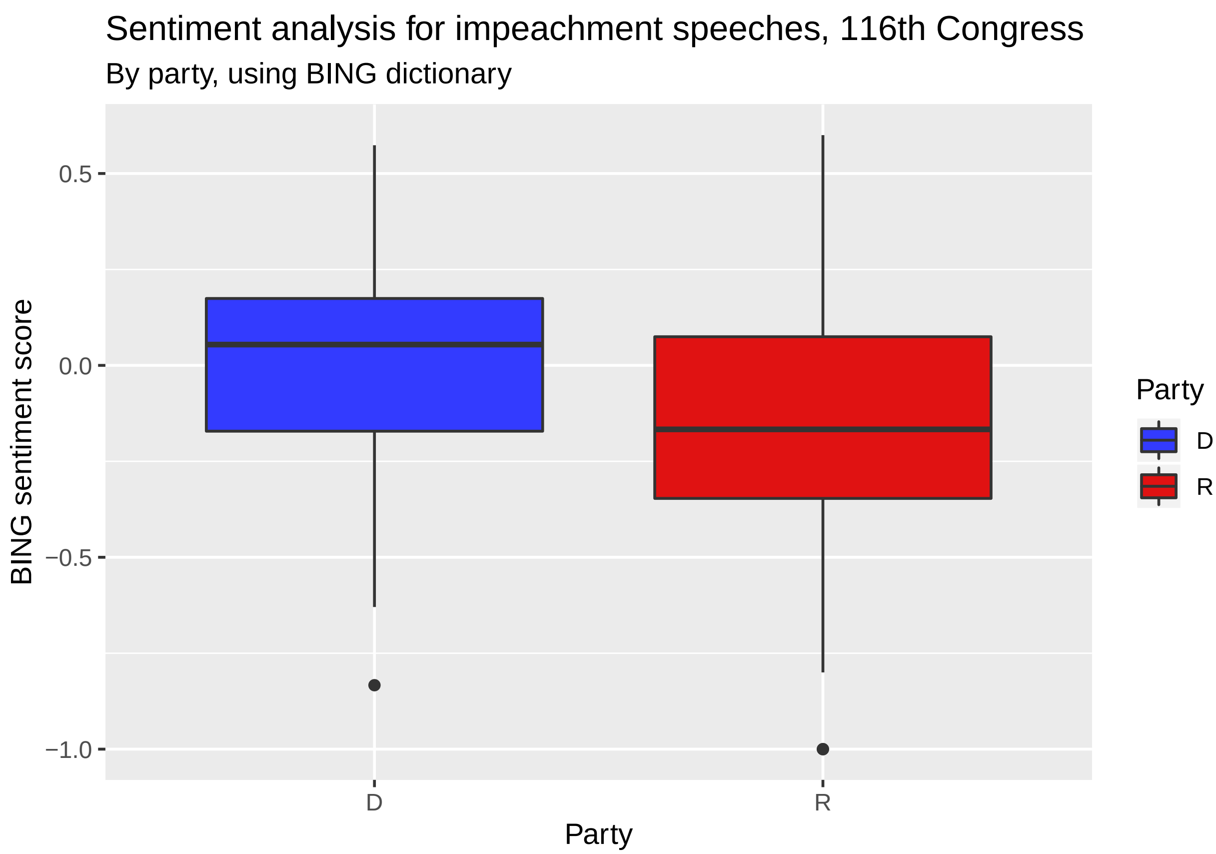
```
## Warning: Removed 14 rows containing missing values (geom_point).
```



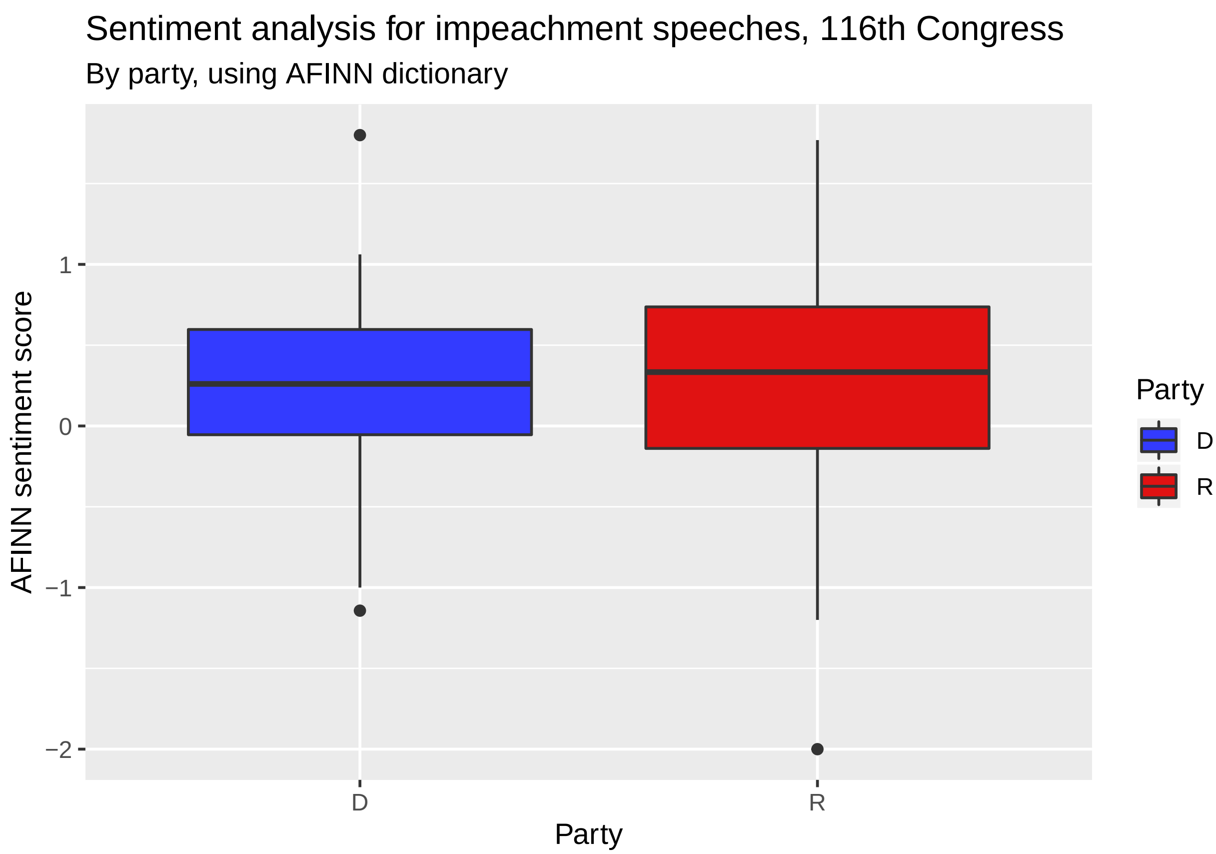

```
# Produce boxplots of wordfish by party
wf.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=wftheta, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  labs(title="Estimated wordfish for impeachment speeches, 116th Congress",
       subtitle="By party, using wordfish",
       fill="Party",
       x = "Party",
       y = "Wordfish theta score")
wf.plot.4
```




```r
# Produce boxplots of sentiment analysis
sent.plot.1 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=bing, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  labs(title="Sentiment analysis for impeachment speeches, 116th Congress",
       subtitle="By party, using BING dictionary",
       fill="Party",
       x = "Party",
       y = "BING sentiment score")
sent.plot.1
```

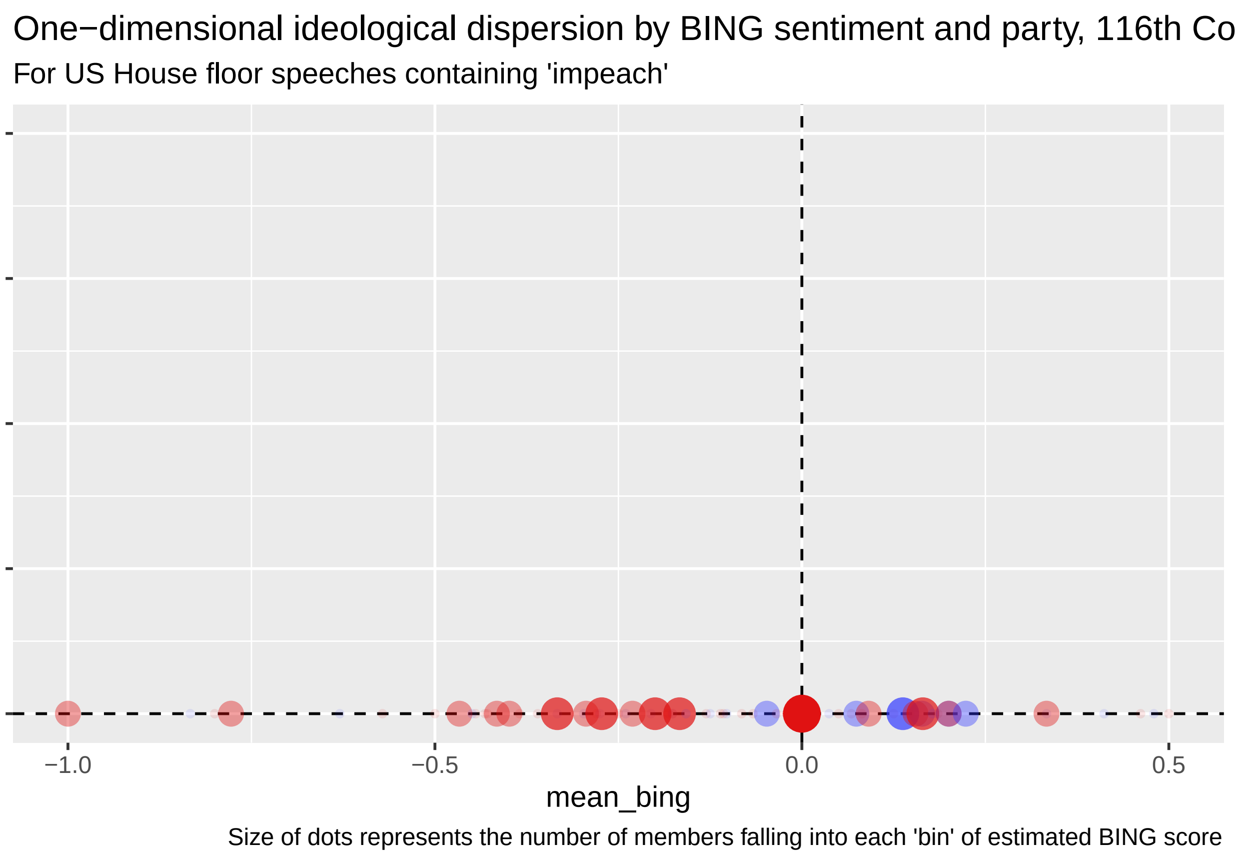



```
sent.plot.2 <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(x=party, y=afinn, fill=party)) +
  scale_fill_manual(values=group.colors) +
  geom_boxplot() +
  labs(title="Sentiment analysis for impeachment speeches, 116th Congress",
       subtitle="By party, using AFINN dictionary",
       fill="Party",
       x = "Party",
       y = "AFINN sentiment score")
sent.plot.2
```



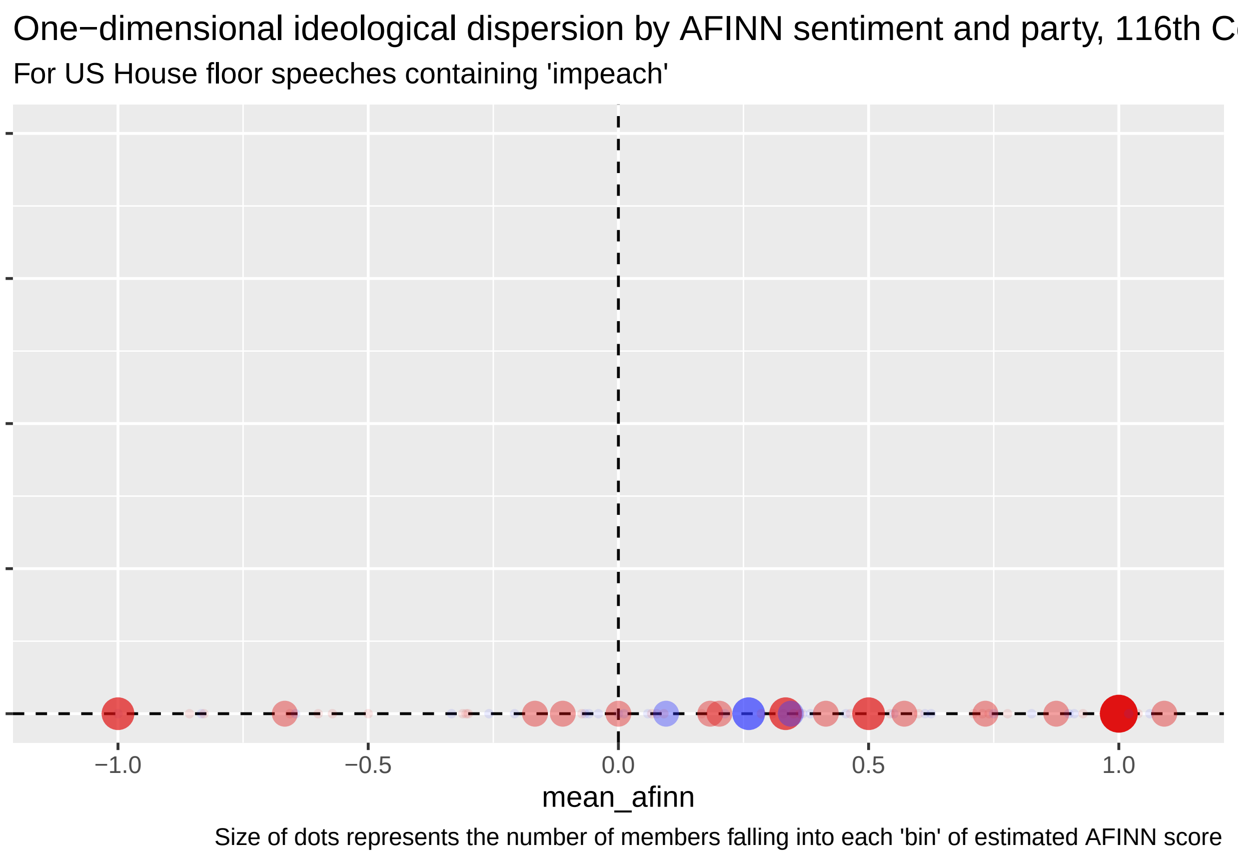

```
# Produce 1D line plots of ideology using sentiment
sent.plot.3 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = bing - (bing %% .01)) %>%
  arrange(bin, bing) %>%
  group_by(bin, party) %>%
  summarize(mean_bing = mean(bing, na.rm=TRUE),
            count=n()) %>%
  ggplot(., aes(x=c(0), mean_bing, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-1,.5)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by BING sentiment and party, 116th Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated BING
       color="Party")
sent.plot.3
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



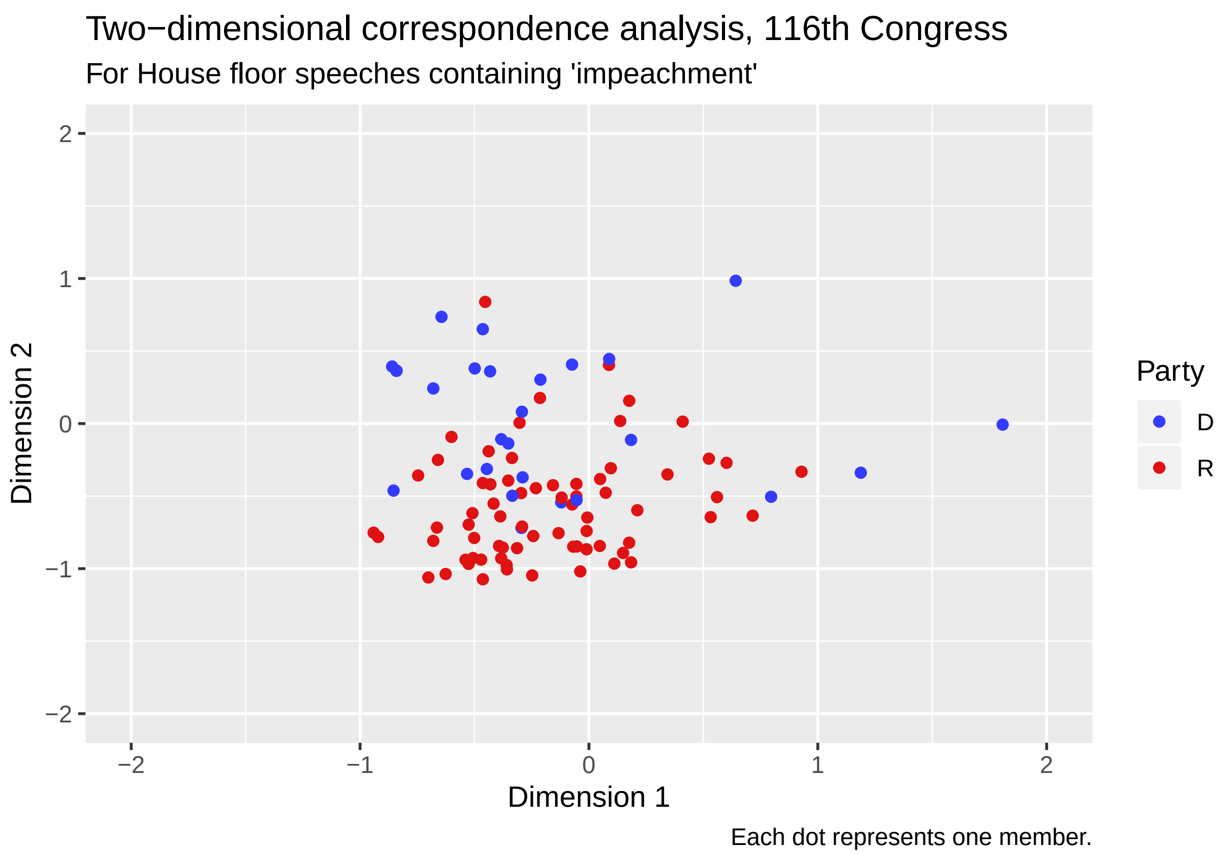

```r
sent.plot.4 <- wswf.df %>%
  filter(party != "I") %>%
  mutate(bin = afinn - (afinn %% .01)) %>%
  arrange(bin, afinn) %>%
  group_by(bin, party) %>%
  summarize(mean_afinn = mean(afinn, na.rm=TRUE),
            count=n()) %>%
  ggplot(., aes(x=c(0), mean_afinn, color=party, alpha=count, size=count)) +
  geom_vline(aes(xintercept=0), linetype="dashed") +
  geom_hline(aes(yintercept=0), linetype="dashed") +
  geom_point(show.legend=FALSE) +
  scale_color_manual(values=group.colors) +
  coord_flip() +
  ylim(c(-1.1,1.1)) +
  xlim(c(0, .001)) +
  theme(axis.text.y=element_blank(),
        axis.title.y=element_blank()) +
  labs(title="One-dimensional ideological dispersion by AFINN sentiment and party, 116th Congress",
       subtitle="For US House floor speeches containing 'impeach'",
       caption="Size of dots represents the number of members falling into each 'bin' of estimated AFINN
       color="Party")
sent.plot.4
```

```
## Warning: Removed 10 rows containing missing values (geom_point).
```




```r
# Plot 2D correspondence analysis
ca2d <- wswf.df %>%
  filter(party != "I") %>%
  ggplot(., aes(ca_dim1, ca_dim2, color=party)) +
  geom_point() +
  xlim(-2, 2) +
  ylim(-2, 2) +
  scale_color_manual(values=group.colors) +
  labs(title="Two-dimensional correspondence analysis, 116th Congress",
       subtitle="For House floor speeches containing 'impeachment'",
       color="Party",
       x="Dimension 1",
       y="Dimension 2",
       caption="Each dot represents one member.")
ca2d
```

```
## Warning: Removed 7 rows containing missing values (geom_point).
```




```
# Use a function to min-max scale our estimated ideology variables
normalize <- function(df, col) {
  min <- min(df[col], na.rm=TRUE)
  max <- max(df[col], na.rm=TRUE)
  newcol <- rep(NA, nrow(df))
  for (i in 1:nrow(df)) {
    if (is.na(df[[col]][[i]])) {
      newcol[[i]] <- NA
    } else {
      newcol[[i]] <- (df[[col]][[i]] - min) / (max-min)
    }
  }
  return(newcol)
}

wswf.df.normalized <- wswf.df %>%
  mutate(ln.bing.n = log(normalize(., "bing")),
         ln.afinn.n = log(normalize(., "afinn")),
         ln.wordscore.n = log(normalize(., "wordscore")),
         ln.wftheta.n = log(normalize(., "wftheta")),
         ln.rcf.n = log(normalize(., "recipient_cfscore")),
         ln.nd1.n = log(normalize(., "nominate_dim1")))
```

```
# Now do the log of scaled vars regression for percent interpretation
dwn.md1 <- lm(ln.nd1.n ~ ln.bing.n, data = subset(wswf.df.normalized,
                                                  !is.infinite(ln.nd1.n) & !is.infinite(ln.bing.n) &
```

```r
                                                    !is.na(ln.nd1.n) & !is.na(ln.bing.n)))

dwn.md2 <- lm(ln.nd1.n ~ ln.afinn.n, data = subset(wswf.df.normalized,
                                                   !is.infinite(ln.nd1.n) & !is.infinite(ln.afinn.n) &
                                                     !is.na(ln.nd1.n) & !is.na(ln.afinn.n)))

dwn.md3 <- lm(ln.nd1.n ~ ln.wordscore.n, data = subset(wswf.df.normalized,
                                                       !is.infinite(ln.nd1.n) & !is.infinite(ln.wordsco
                                                         !is.na(ln.nd1.n) & !is.na(ln.wordscore.n)))

dwn.md4 <- lm(ln.nd1.n ~ ln.wftheta.n, data = subset(wswf.df.normalized,
                                                     !is.infinite(ln.nd1.n) & !is.infinite(ln.wftheta.n
                                                       !is.na(ln.nd1.n) & !is.na(ln.wftheta.n)))

stargazer(dwn.md1, dwn.md2, dwn.md3, dwn.md4, type = "text",
          title = "Regression of log DW-NOMINATE on log ideology estimate")
```

```
##
## Regression of log DW-NOMINATE on log ideology estimate
## ==========================================================================================
##                                                    Dependent variable:
##                       --------------------------------------------------------------------
##                                                          ln.nd1.n
##                              (1)                (2)                  (3)                 (4)
## ------------------------------------------------------------------------------------------
## ln.bing.n                 -0.184
##                          (0.148)
##
## ln.afinn.n                                  0.032
##                                            (0.201)
##
## ln.wordscore.n                                                  0.757***
##                                                                 (0.186)
##
## ln.wftheta.n                                                                          0.019
##                                                                                      (0.093)
##
## Constant                -0.652***          -0.505***            -0.147              -0.488***
##                          (0.116)            (0.132)             (0.106)              (0.148)
##
## ------------------------------------------------------------------------------------------
## Observations               94                 95                   95                   95
## R2                        0.017             0.0003               0.151               0.0004
## Adjusted R2               0.006             -0.010               0.142               -0.010
## Residual Std. Error  0.611 (df = 92)    0.615 (df = 93)      0.559 (df = 93)      0.612 (df = 93)
## F Statistic         1.544 (df = 1; 92) 0.026 (df = 1; 93) 16.545*** (df = 1; 93) 0.042 (df = 1; 93)
## ==========================================================================================
## Note:                                                         *p<0.1; **p<0.05; ***p<0.01
```

```r
# Now do the log of scaled vars regression for percent interpretation (DIME)
dwn.md5 <- lm(ln.rcf.n ~ ln.bing.n, data = subset(wswf.df.normalized,
                                                  !is.infinite(ln.rcf.n) & !is.infinite(ln.bing.n) &
                                                    !is.na(ln.rcf.n) & !is.na(ln.bing.n)))
```

```r
dwn.md6 <- lm(ln.rcf.n ~ ln.afinn.n, data = subset(wswf.df.normalized,
                                                    !is.infinite(ln.rcf.n) & !is.infinite(ln.afinn.n) &
                                                      !is.na(ln.rcf.n) & !is.na(ln.afinn.n)))

dwn.md7 <- lm(ln.rcf.n ~ ln.wordscore.n, data = subset(wswf.df.normalized,
                                                        !is.infinite(ln.rcf.n) & !is.infinite(ln.wordsc
                                                          !is.na(ln.rcf.n) & !is.na(ln.wordscore.n)))

dwn.md8 <- lm(ln.rcf.n ~ ln.wftheta.n, data = subset(wswf.df.normalized,
                                                      !is.infinite(ln.rcf.n) & !is.infinite(ln.wftheta.
                                                        !is.na(ln.rcf.n) & !is.na(ln.wftheta.n)))

stargazer(dwn.md5, dwn.md6, dwn.md7, dwn.md8, type = "text",
          title = "Regression of log normalized DIME on log ideology estimate")
```

```
##
## Regression of log normalized DIME on log ideology estimate
## ==========================================================================================================
##                                                    Dependent variable:
##                      -------------------------------------------------------------------------------------
##                                                          ln.rcf.n
##                              (1)                (2)                  (3)                    (4)
## ----------------------------------------------------------------------------------------------------------
## ln.bing.n                  -0.034
##                           (0.183)
##
## ln.afinn.n                                     0.173
##                                              (0.237)
##
## ln.wordscore.n                                                    1.126***
##                                                                   (0.225)
##
## ln.wftheta.n                                                                             -0.004
##                                                                                         (0.100)
##
## Constant                 -0.594***          -0.467***             -0.016              -0.542***
##                           (0.141)            (0.156)              (0.128)              (0.166)
##
## ----------------------------------------------------------------------------------------------------------
## Observations                96                 97                   97                   97
## R2                        0.0004             0.006                0.209                0.00002
## Adjusted R2               -0.010             -0.005               0.201                -0.011
## Residual Std. Error  0.758 (df = 94)    0.754 (df = 95)     0.672 (df = 95)      0.706 (df = 95)
## F Statistic         0.034 (df = 1; 94) 0.534 (df = 1; 95) 25.138*** (df = 1; 95) 0.002 (df = 1; 95)
## ==========================================================================================================
## Note:                                                                   *p<0.1; **p<0.05; ***p<0.01
```