Project Final Report

Nat Mammo
Santiago Larrain

Contents

Intro

Over time, the US Supreme Court has become more and more polarized in the makeup of its members. It has reached the point where justices often vote in line with their respective party. This partisan climate has led us to try and see how predictable the vote of a Supreme Court justice is. Using structured case data and text case summaries, we try to predict case votes for each of the last 9 justices.

The Scope

With the charged political climate currently surrounding the Supreme Court, many interesting questions could be posed in this project. To maintain a well-defined focus that matched our time constraints, we focused on accurately predicting each justice's vote and measuring whether the inclusion of text data will be useful in the predictions.

The Data

The data involved in this analysis came from two sources:

1. *The Supreme Court Database*: This database contained how every justice's vote since 1946. As well as how the justice voted, each record included facts about the case, such as the relevant policy issue, the type of plaintiff and defendant, the lower court's decision, and many other variables.
2. *The Oyez Project at IIT*: This project contained text summaries of cases since the early 1800s. These summaries are usually a couple paragraphs and also contain the question posed to the court.

We joined these two sources on the variable "docket number". Since missing data became too frequent prior to the 1990s, we decided to focus on the most recent justices of the court.

The Models

We tried different machine learning algorithms in this step over different sets of features.

Only Features by Judge

The first approach taken in our analysis was to generate 9 different models, one for each justice. To accomplish this, we created a dataset for each justice that contained all the cases they had previously voted on. The features used in these predictions were issue, issue area, lower court disposition direction, petitioner, and respondent. The models initially considered were random forest, logistic, and nearest neighbor. The baseline that we were competing against was either always predicting liberal or conservative, depending on the justice's known political belief, e.g. predicting that Scalia would vote conservatively on every case. Our main evaluation metric was accuracy as we were not particularly concerned with precision, recall, or other metrics that focused on accurately one outcome or the other. The following table contains the accuracies for each justice using 5-fold cross-validation. The bolded values were the model that performed the best for that justice. As evident below, random forest and logistic are the best initial candidates.

	Random Forest	**Logistic**	**Nearest Neighbor**	**Baseline**
Scalia	**.718**	.712	.636	.668
Kennedy	**.691**	.667	.600	.589
Thomas	.713	**.714**	.645	.682
Ginsburg	**.693**	.666	.596	.630
Breyer	**.676**	.652	.590	.590
Roberts	.631	**.651**	.596	.589
Alito	.680	**.692**	.620	.635
Sotomayor	.656	**.660**	.600	.630
Kagan	**.626**	.623	.525	.618

After observing that certain cases were consistently misclassified for all justices, we explored boosting as a potential fix for such cases. The table below displays the results for two different implementations of boosting. The bolded values are the ones that performed best in both this table and the one above.

	Gradient Boosting	**AdaBoost**	**Baseline**
Scalia	**.727**	.695	.668
Kennedy	.677	.647	.589
Thomas	**.721**	.707	.682
Ginsburg	.657	.637	.630
Breyer	**.689**	.632	.590
Roberts	.636	.639	.589
Alito	.654	.665	.635
Sotomayor	.629	.637	.630
Kagan	.614	.595	.618

Only features all dataset

Since Random Forest was performing better on the by judges models, we used random forests again, but this time we used all of the votes. So it is a model that was using the same set of features as before, but it was but for all judges this time. Obviously, we had to provide dummified version of the judges as well, so the model could predict, for the same set of features, different results depending on the judge (no additional information on the judge was provided).

Only text

We then tried different algorithms with only the facts of the case. We wanted to try the clean hypothesis of “does text work” or “can the facts of the case alone provide a reliable model to predict a judge's behavior?”.

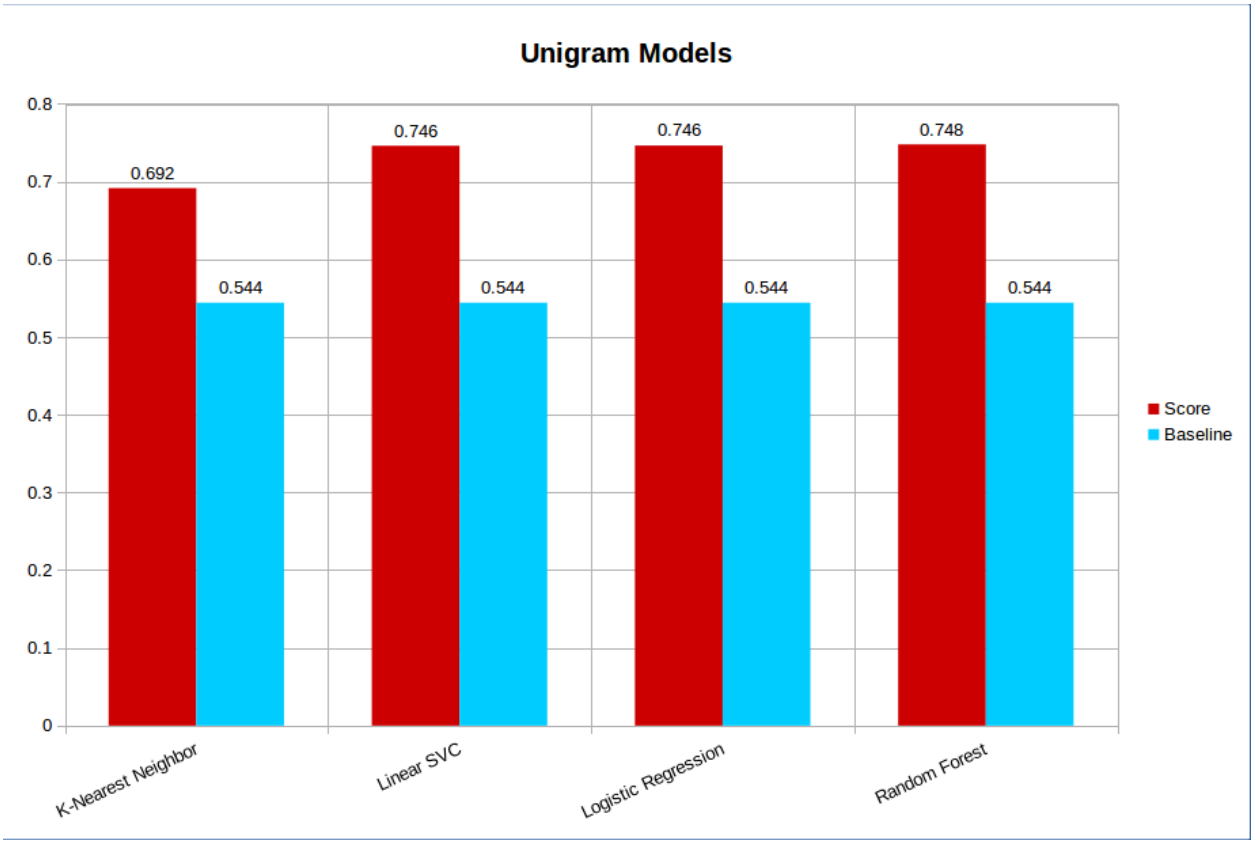
For this, we used Logistic Regression, Linear SVC, K-Nearest Neighbor and Random Forests. Since we needed to make sure that the Random Forests used the dummified version of the judges as a feature, we had to use the max_features=None so we forced to model to used all the words.

We used the TfidfVectorizer package from scikit-learn to process the text. We provided a pandas Series of the facts of the case of all votes (so several of them are repeated, one for each judge who took a vote) and then this package Vectorizes the text and applies the tf-idf -

term frequency, inverse document frequency- all in one step. After this, we have a sparse Matrix of all the facts of all cases one for each justice that voted in.

Then we dummified the justices name with the get_dunmmy function of pandas and the sparse argument. However, this creates a sparse matrix of a different type than the facts one (scipy vs. pandas). So we have to transform it into a scipy matrix and only then we can join both matrices.

When we have one sparse matrix with all the data, we can implement each of the models. These are the results:



So, besides Nearest Neighbors, they all seem to perform very similar, Random Forest being the best one.

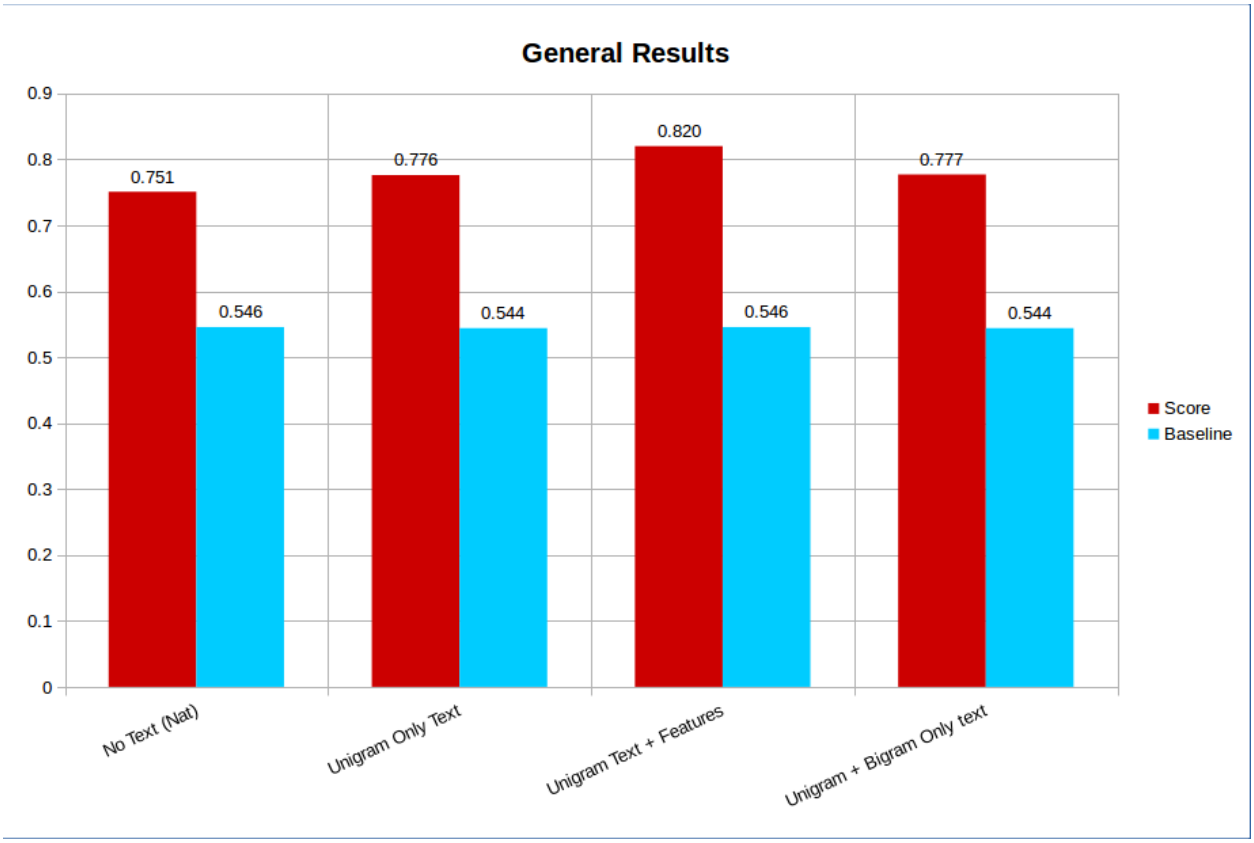
However, the most interesting thing to be noted, is that, contrary to our initial Hypothesis and fear, a model of only text actually performs better than a model of only relevant features. So this was, to us, a very nice and interesting discovery.

We then tried our best model, the Random Forest, with not only a Unigram but a Unigram and a Bigram together. This increased the features of our dataset considerably, from 14670 unique grams to over 140,000. So running this model took a while and the results were kind of disappointing of similar to the ones of the unigram (check General Results Graph).

Text + Features

In this model, we combined the features that we had used before and the unigram text data. We tested the different algorithms -the same as before- and here we did actually had a clear winner with the Random Forest so this was our final model.

Overall, this were the best results we got for Random Forest for each different set of features.

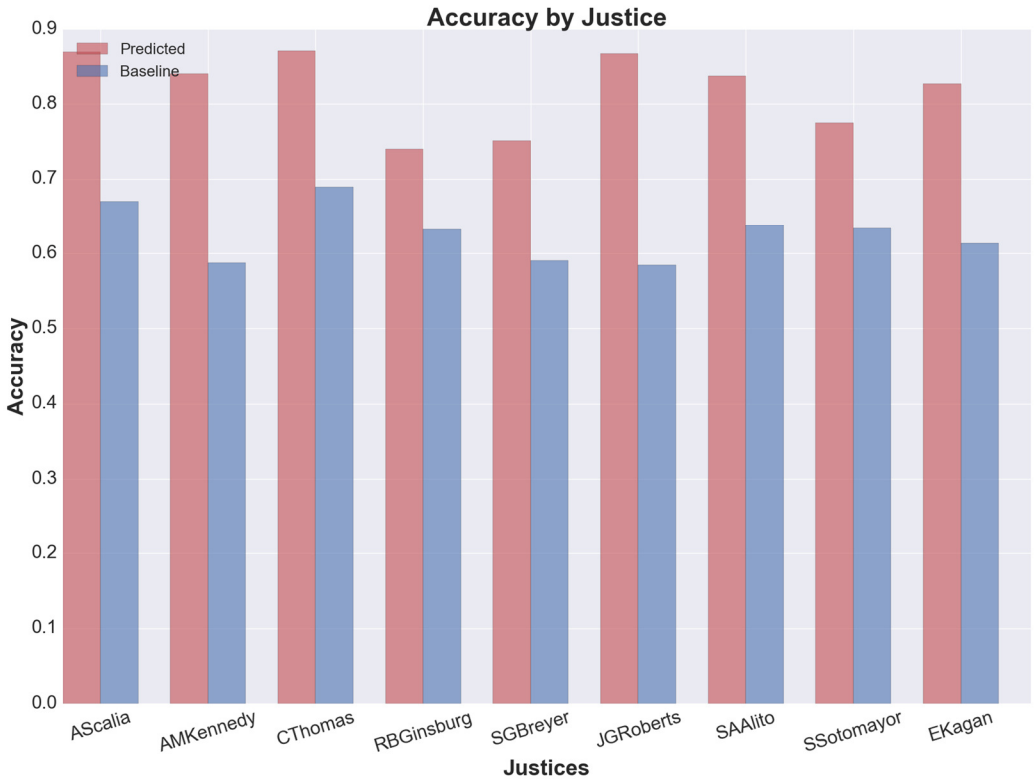


We can observe that the Unigram Text + Features got 82% accuracy. That's a 27.4% improvement from the baseline, and we think is significant.

Results

The following results and observations are all drawn from our best model.

We first checked the improvement our model had with respect to the baseline for each Justice:

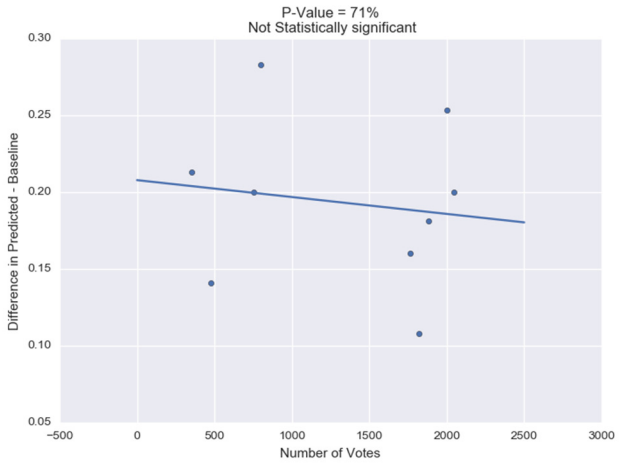


We observed that our model did fairly well with everybody. However, there were some notable differences.

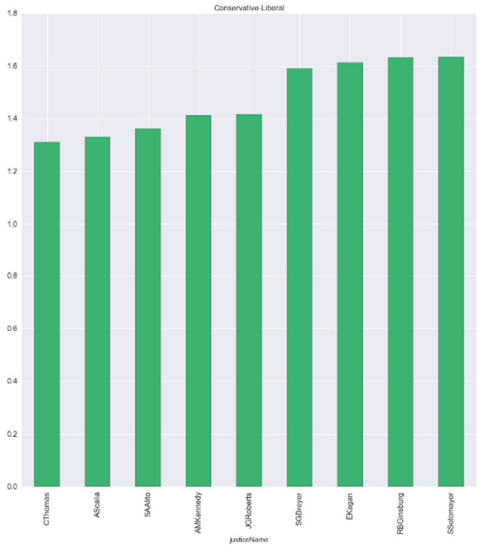
So we tried to understand why, and seek some relationship between the ratio of improvement and the amount of times a justice had vote. We understand that, given that there are only 9 judges, this could hardly be significant, but we still wanted to explore if there was something that could be interesting, even though not significant. There wasn't. The results were highly not significant with a high p-value that meant that we failed to the reject the Null hypothesis that there was no relationship. If anything, the coefficient was very close to zero and negative:

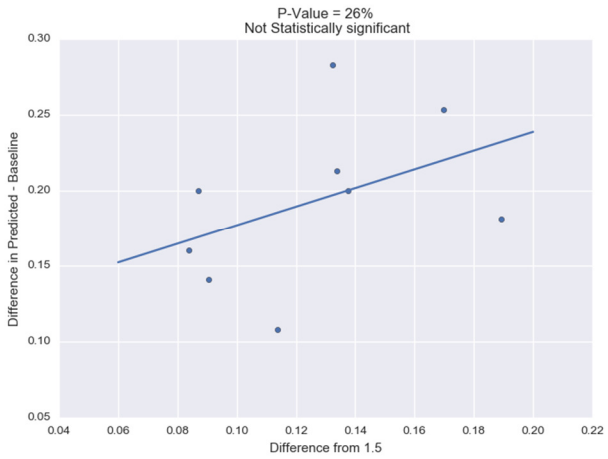
	justiceName	Predicted	Baseline	n_votes	diff
0	AScalia	0.869565	0.669761	2047	0.199805
1	AMKennedy	0.8405	0.587	2000	0.2535
2	CThomas	0.870745	0.689362	1880	0.181383
3	RBGinsburg	0.739967	0.632216	1819	0.107752
4	SGBreyer	0.75071	0.590574	1761	0.160136
5	JGRoberts	0.867168	0.58396	798	0.283208

6	SAAlito	0.837302	0.637566	756	0.199735
7	SSotomayor	0.774737	0.633684	475	0.141053
8	EKagan	0.826705	0.613636	352	0.213068

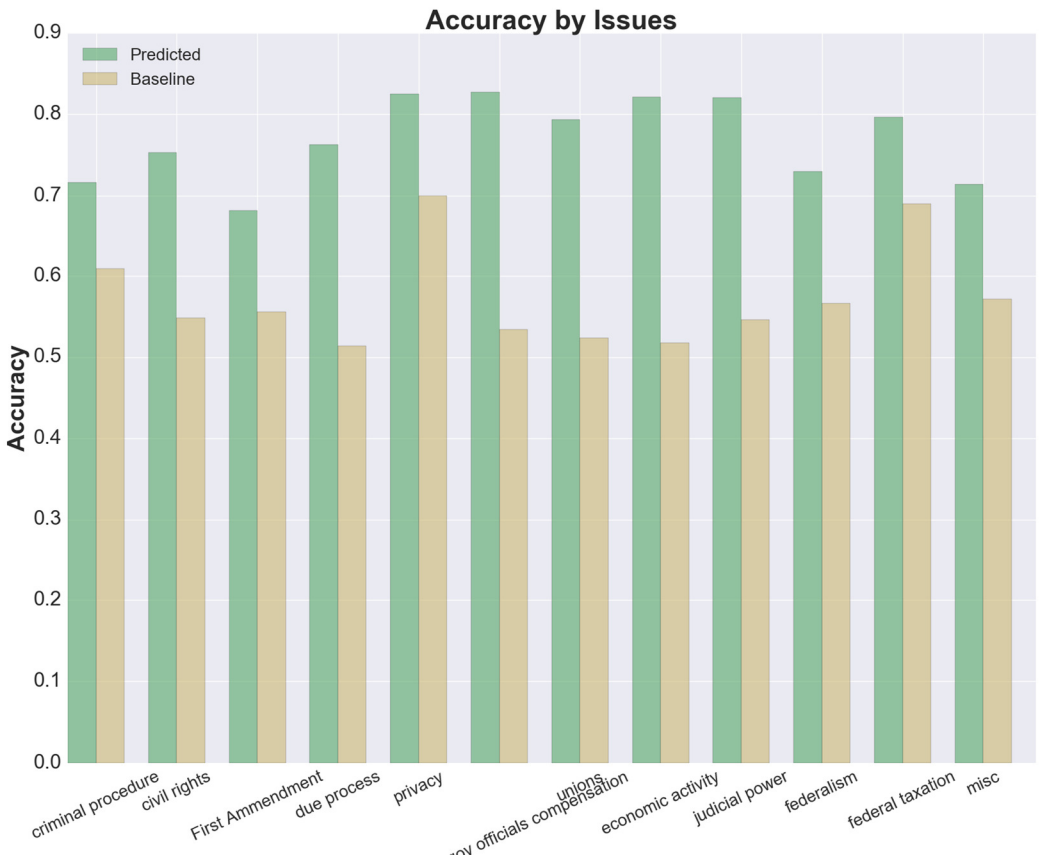


We then explore the relationship of the “extremeness” that we simple define as |1.5-x| with x = the mean of all their direction votes: so the higher, the more liberal, the lower, the more conservative.
Again, there was no significance, however the p-value was lower, and the magnitude of the coefficient did show indeed some relationship. So, we can’t affirm that we do better with more extreme justices, but we can’t certainly rule it out.

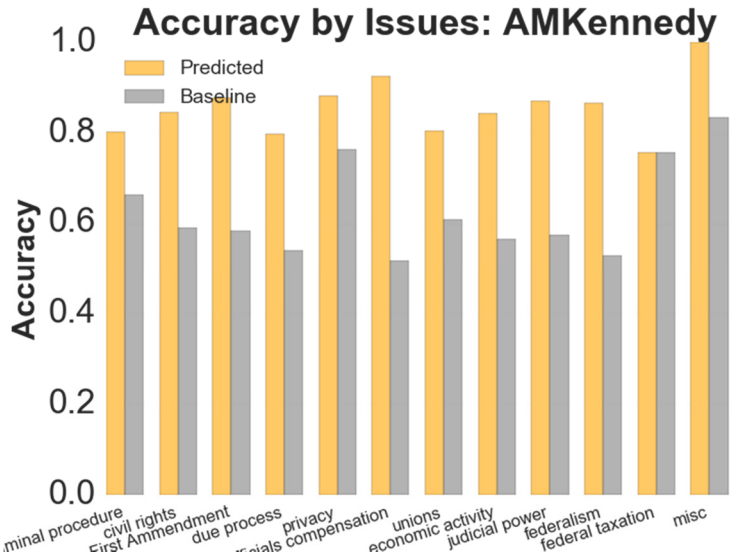


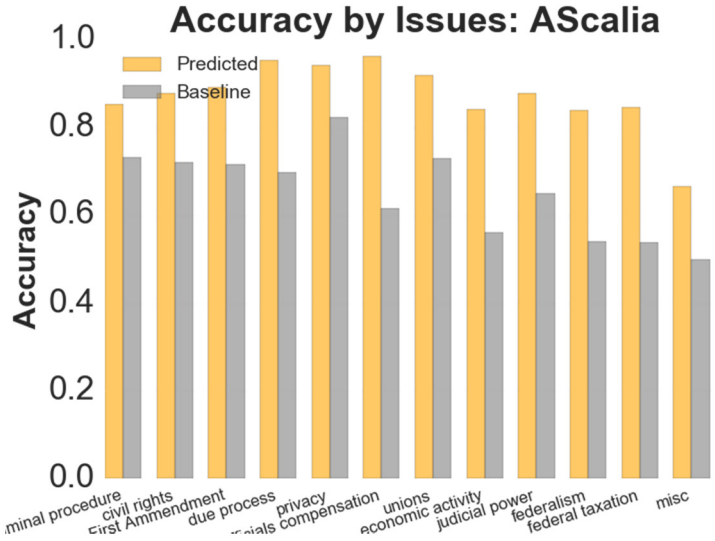


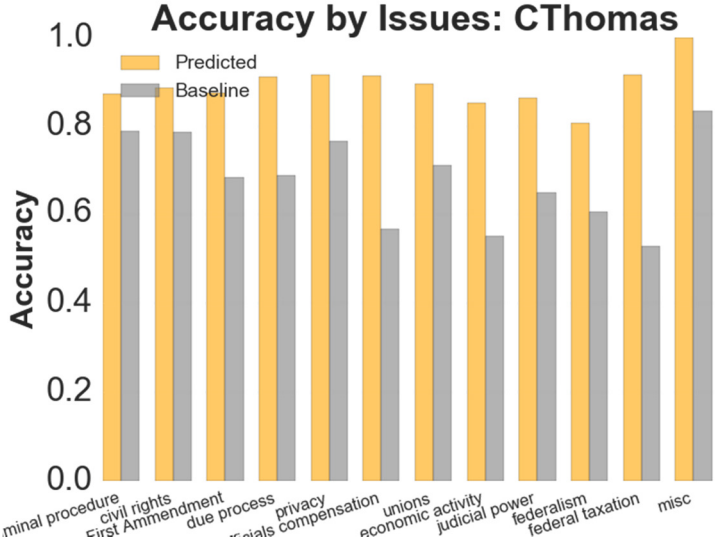
After that, we analyzed the improvement of our model by issue, and as expected, we did better in some than others:

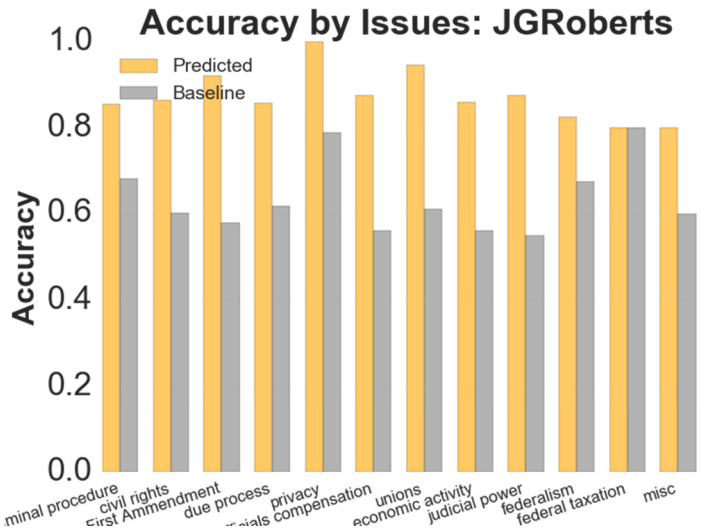


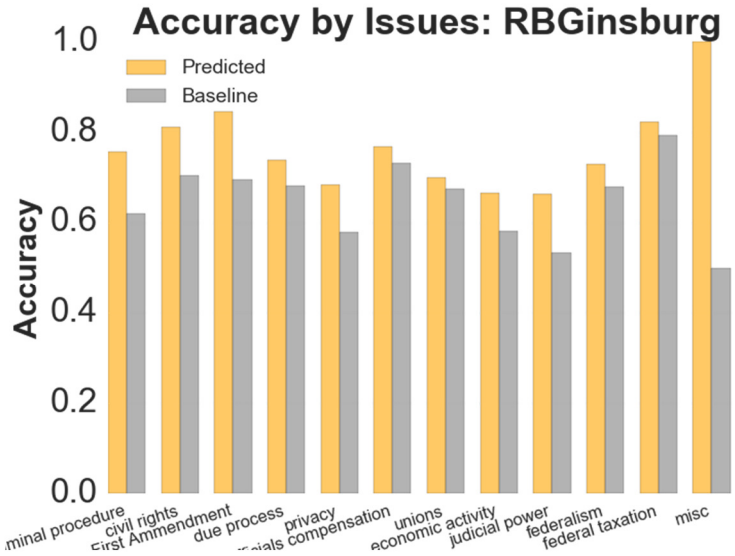
And then, we proceeded to look at how the model perform by issue and by justice. In general, we observe a very good model that did better in almost all of the issues on all of the candidates. However, there were some few cases where we did worse than the baseline.

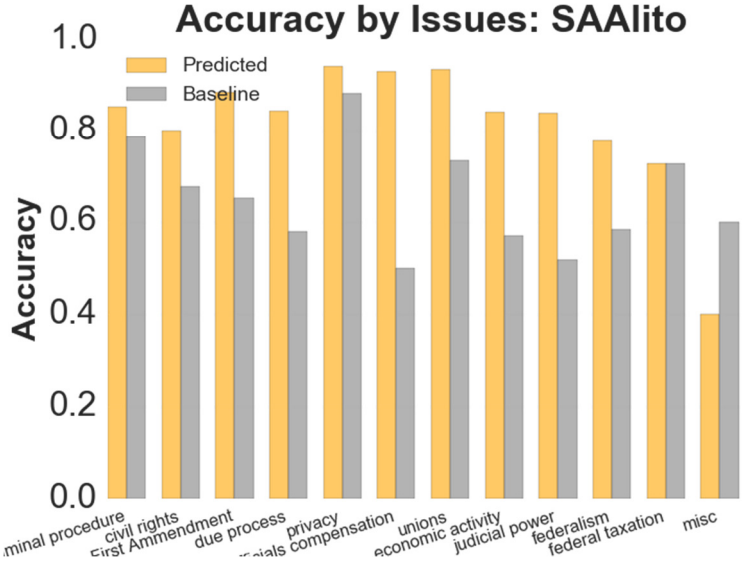

Accuracy by Issues: AMKennedy
Predicted
Baseline
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
minal procedure
civil rights
First Ammendment
due process
privacy
unions
economic activity
judicial power
federalism
federal taxation
misc

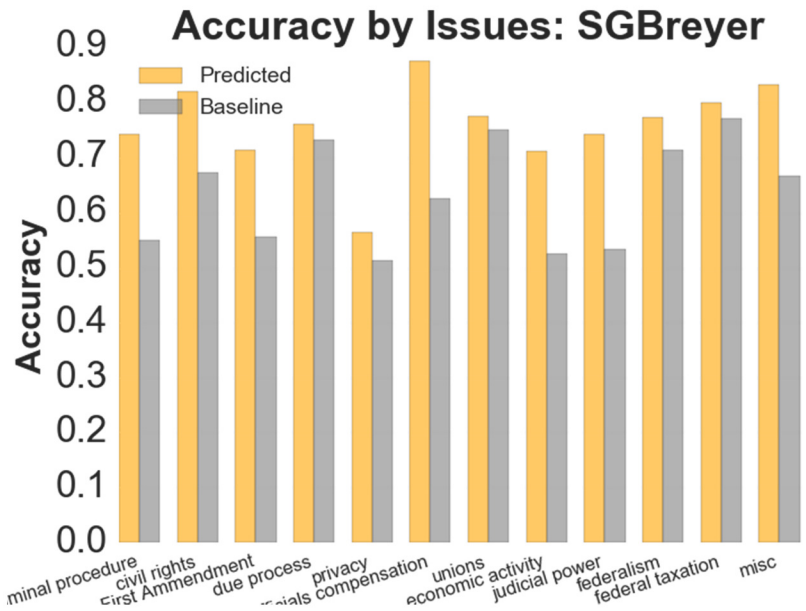

Accuracy by Issues: AScalia
Predicted
Baseline
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
minal procedure
civil rights
First Ammendment
due process
privacy
unions
economic activity
judicial power
federalism
federal taxation
misc

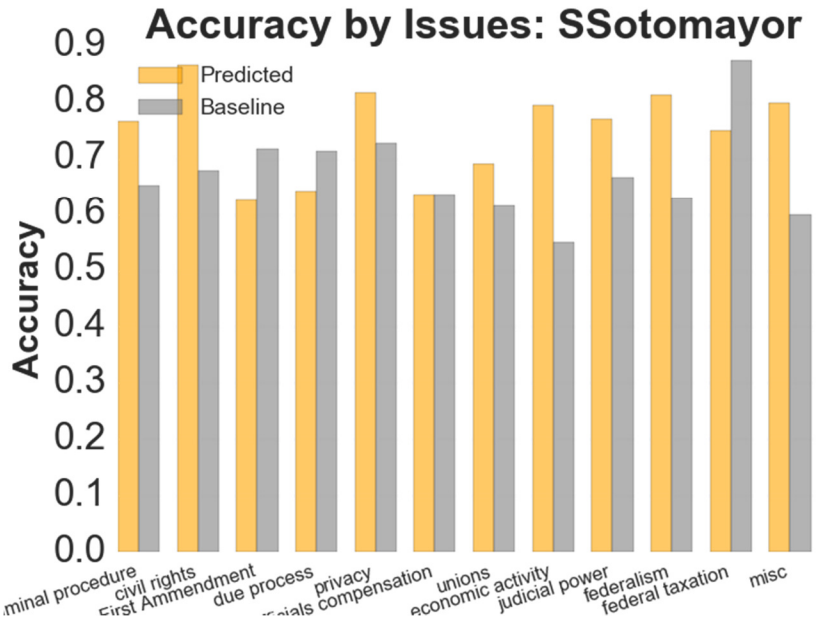

Accuracy by Issues: CThomas
Predicted
Baseline
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
minal procedure
civil rights
First Ammendment
due process
privacy
unions
economic activity
judicial power
federalism
federal taxation
misc


Accuracy by Issues: JGRoberts
Predicted
Baseline
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
minal procedure
civil rights
First Ammendment
due process
privacy
unions
economic activity
judicial power
federalism
federal taxation
misc


Accuracy by Issues: RBGinsburg
Predicted
Baseline
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
minal procedure
civil rights
First Ammendment
due process
privacy
unions
economic activity
judicial power
federalism
federal taxation
misc

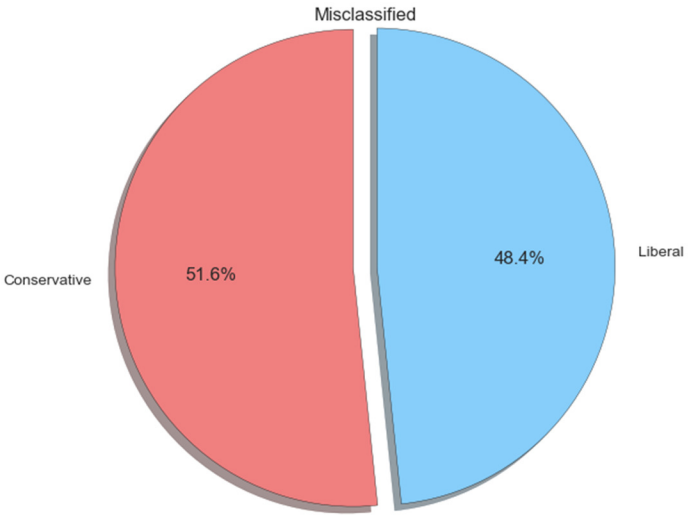

Accuracy by Issues: SAAlito
Predicted
Baseline
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
minal procedure
civil rights
First Ammendment
due process
privacy
unions
economic activity
judicial power
federalism
federal taxation
misc





Misclassified

There was less than 20% of misclassification and we analyzed if this was heavily biased towards conservative or liberals and we found out that it wasn't. The model misclassified justices in a failry even proportions. There were a total of 2674 misclassified votes:



Conclusion

Our analysis yielded interesting results at each stage of our work. Our preliminary data analysis showed us that there is unsurprisingly a partisan slant to each justice, even including Kennedy. As a baseline, this partisan slant though is not that good a predictor. Our models with justice specific models increased the accuracy by about 7%. While a definite improvement, this increase brought the accuracies to the high 60s/low 70s. When including text data, our accuracies greatly improved to 82%. This model incorporated both unigram text features as well as general information about the case. This 27% improvement from the baseline confirmed for us that our text model is a success.

Datasets:

Facts of the case:
https://github.com/slarrain/MachineLearning/blob/master/project/cases_55-15.csv?raw=true

Details of the case:
https://github.com/slarrain/MachineLearning/blob/master/project/SCDB_2015_02_justiceCentered_LegalProvision.csv?raw=true