Capstone Project 2 – Predicting Judicial Authorship of Court Opinions

Final Report

by Greg Jacobs (jacobs.greg@gmail.com)

I. Problem Statement

One of the core skills of lawyers is their ability to persuade courts to accept their arguments and positions. Legal proceedings often times are highly contentious and favorable resolutions for clients can depend significantly upon legal counsel's effective communication and established credibility with the presiding judge. Lawyers therefore stand to benefit from any techniques or strategic approaches that provide insights into a particular judge's ideology, thinking process, and style.

The project seeks to build a model that can accurately identify the particular judge who authored a court opinion based on the text of the opinion. A model that can successfully identify judge authorship could provide users with a wealth of information about a particular judge that otherwise would go undetected. An accurate model also could provide nuanced insights into a particular judge's mindset and style.

The target clients for this model would be private law firms and other litigators that could use the model to gain a competitive advantage over opposing parties in a pending court proceeding. An accurate judge authorship identification model could be used in a variety of fashions, including to identify other persuasive authority (out-of-state case opinions, secondary authorities, etc.) that are most similar to the opinions of a particular judge (under the rather logical presumption that a judge is more likely to find authorities written in a style similar to that particular judge to be persuasive). Clients could use the model to better inform their persuasive-writing techniques and overall advocacy style to increase the likelihood that the presiding judge will accept their positions and arguments.

II. Dataset

The dataset I used for this project has been made publicly available by Harvard Law School's Caselaw Access Project. Harvard has made a few states' caselaw decisions publicly available for bulk download at (https://case.law/bulk/download/). I have chosen to use the North Carolina caselaw database for this project.

The dataset has 97,601 separate case opinions and the following 14 columns worth of data for each decision:

```
Index(['casebody', 'citations', 'court', 'decision_date', 'docket_number',
       'first_page', 'frontend_url', 'id', 'jurisdiction', 'last_page', 'name',
       'name_abbreviation', 'reporter', 'volume'],
      dtype='object')
```

III. Data Wrangling and Preprocessing

The dataset is downloadable as a .json file, in which these columns contain varying degrees of nested dictionaries and/or lists of dictionaries. For a detailed explanation of the steps I took to convert this file into a pandas dataframe, please visit the Data_Wranging notebook in the GitHub repository for this project.[1]

Once converted into dataframe, the dataset required a fair amount of cleaning and preprocessing, including removal of cases without a written decision, addressing cases with multiple decisions, and remedying OCR mistakes in the judicial authorship column (which was substantial). To review the methods I employed to remedy these issues and otherwise clean and preprocess the data, please visit the Data_Preprocessing notebook in the GitHub repository for this project.[2]

IV. Exploratory Data Analysis

[1] The Data_Wranging notebook can be accessed at https://github.com/gmj110680/Springboard/blob/master/Capstone_Project_2/Data_Wrangling.ipynb.

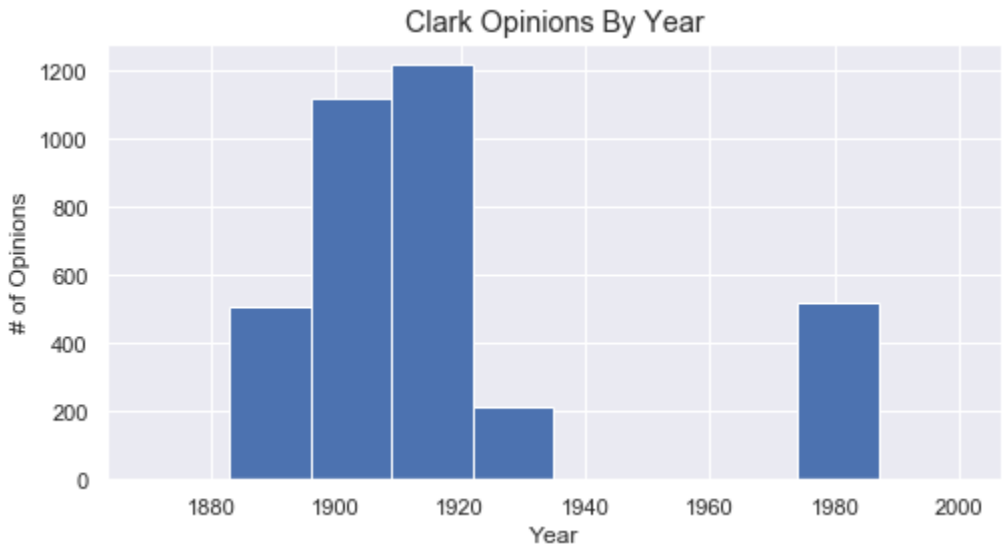
[2] The Data_Preprocessing notebook can be accessed at https://github.com/gmj110680/Springboard/blob/master/Capstone_Project_2/Data_Preprocessing.ipynb.

Once the data was cleaned and organized, I performed some initial analysis to better understand the corpus of opinions.

A. Analyzing the Judge Authorship Totals

The first thing that I analyzed is the judge authorship totals to confirm their accuracy and to get a sense of the authorship distribution for the corpus. The primary concern became the presence of multiple judges with the same last name. To the extent possible, I had to identify and parse out that data so that each judge's opinions were properly labeled separately.

To do this, I created time series histograms for each of the names that had the most opinions and/or were fairly common last names (e.g., martin, johnson). Here is an example of a graph for opinions authored by judges with the last name 'Clark':



As evidenced by the histogram, there were two distinct time periods when a Judge Clark was issuing opinions. With this information in hand and through some simple Google searches, I was able to identify that Walter Clark was a judge in the early 19th century while Edward Clark served on the bench in the 1980s time frame.

I was able to use this process to identify and segregate the opinions according to their correct author. After separating out the common-named judges, I ended up with a pool of 163 different judges (including per curiam opinions as a category), which will serve as the number of classes for my models.

B. Building Additional Features for Analysis

In addition to the actual text of the written decisions, I decided to build some additional features to gain further insight into the corpus of opinions. To do this, I used TextBlob, a powerful text-processing package that provides access to a number of common text-processing operations. Through this process, I added the word count, sentence count, and average sentence length for each opinion in the dataset. TextBlob also has two sentiment features that I included: (1) the 'polarity' feature, which seeks to measure the degree that text is positive or negative (on a -1-to-1 scale); and (2) the 'subjectivity' feature, which seeks to measure the degree to which the text is objective/fact-based versus subjective/opinion-based (on a 0-to-1 scale).

Once I had created these features, I did some basic statistical analysis to get a better sense of the corpus:

	Word Count	Sent. Count	Avg. Sent. Length	Polarity	Subjectivity
Mean	1429.29	63.35	24.05	0.04498	0.41324
STD	1635.16	77.18	9.48	0.07987	0.11141
Minimum	3	1	1.5	-0.80000	0.00000
25%	447	18	18.08	0.00000	0.36756
50%	953	38	22.14	0.04167	0.41655
75%	1849	81	28.56	0.08177	0.46629
Maximum	4911	2169	203.5	1.00000	1.00000

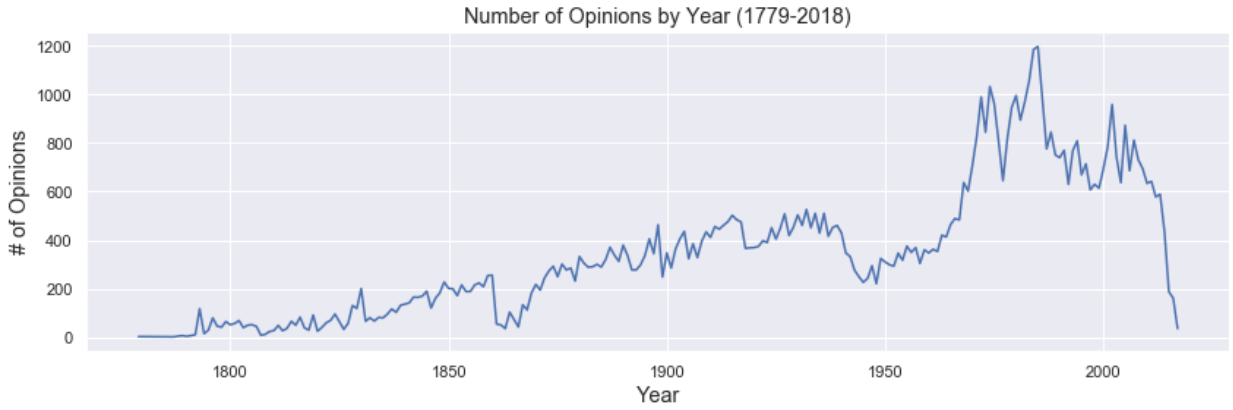
Interestingly, the word count, sentence count, and average sentence length columns all have mean values significantly higher than their median value (the 50% result). All three features also have abnormally large standard deviations. It is clear that they are being skewed by a few unusually long opinions.

The polarity and subjectivity features appear to be normally distributed, with the opinions on average trending slightly positive and slightly more fact-based than opinion-based.

C. Court Opinion Trends Over Time

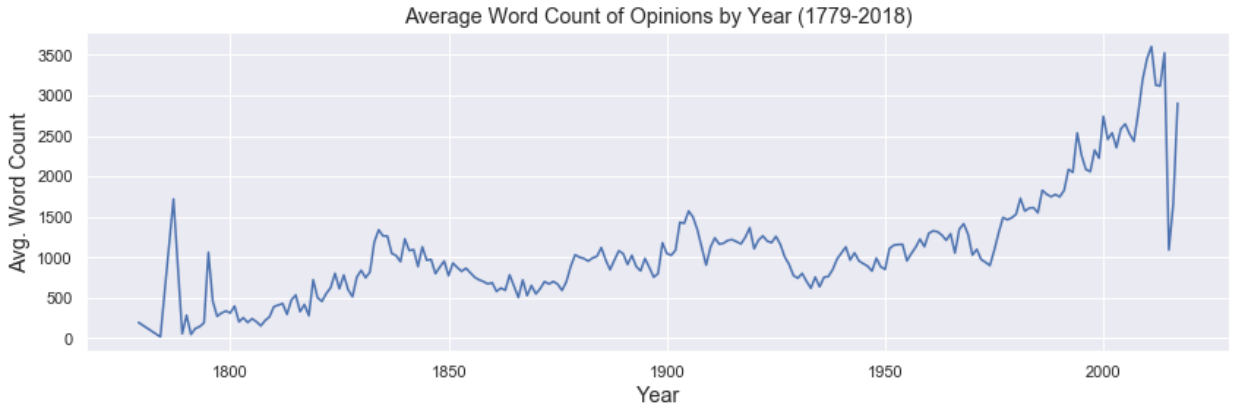
Next, I analyzed these features over time to see if I could identify any overall trends with North Carolina court opinions.

1. Number of Opinions



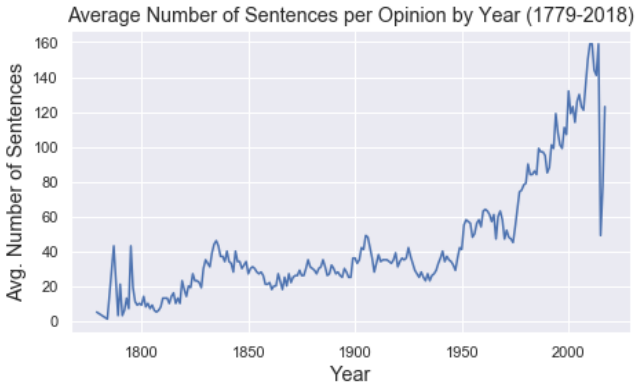
The graph demonstrates an overall gradual increase of opinions over time until recently, with a significant increase in the number of opinions in the 1960-1980 time period.

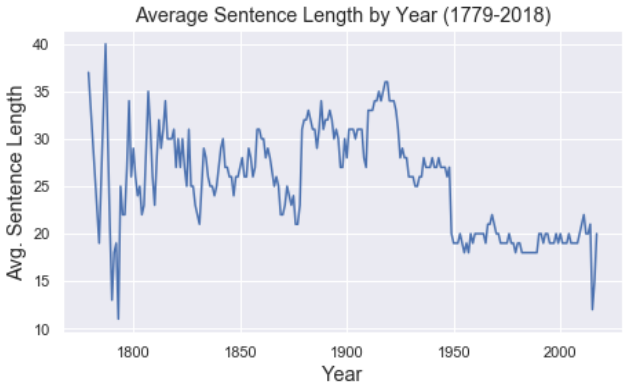
2. Word Count Over Time



Opinion lengths from North Carolina courts have held steady at a range of 600-1500 words on average until the latter part of 20th century, when courts began to issue progressively longer opinions. That trend appears to have held steady (with the most recent fluctuation likely due to a smaller sample size because not all opinions for the most recent years have yet become accessible and added to the database).

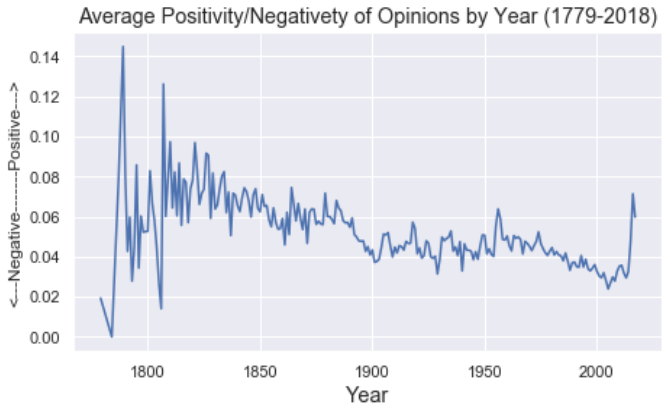
3. Sentence Count/Length Over Time

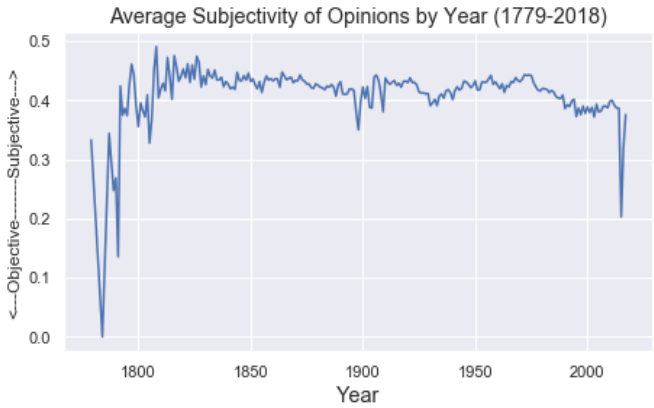




Interestingly, the data demonstrates that while the courts have tended to use more sentences per opinion (consistent with the word count increase discussed above), the courts have actually reduced the length of their sentences over time. This could be indicative of a change towards a more declarative and concise writing style.

4. Court Sentiment Over Time



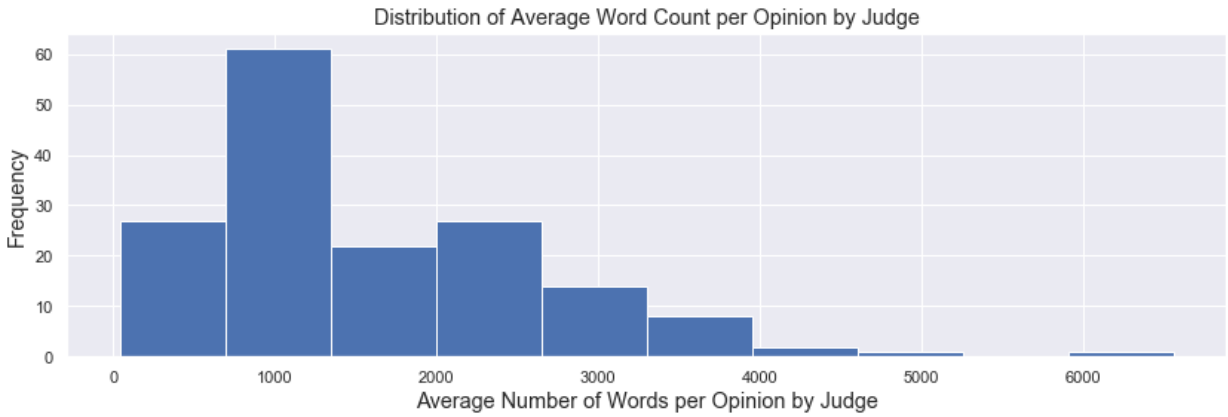


Overall, court opinions in North Carolina have skewed slightly positive, though they have steadily decreased in positivity over time. Opinions also have skewed slightly more objective than subjective and appear to be trending slightly more objective in the modern era.

D. Individual Judge Trends

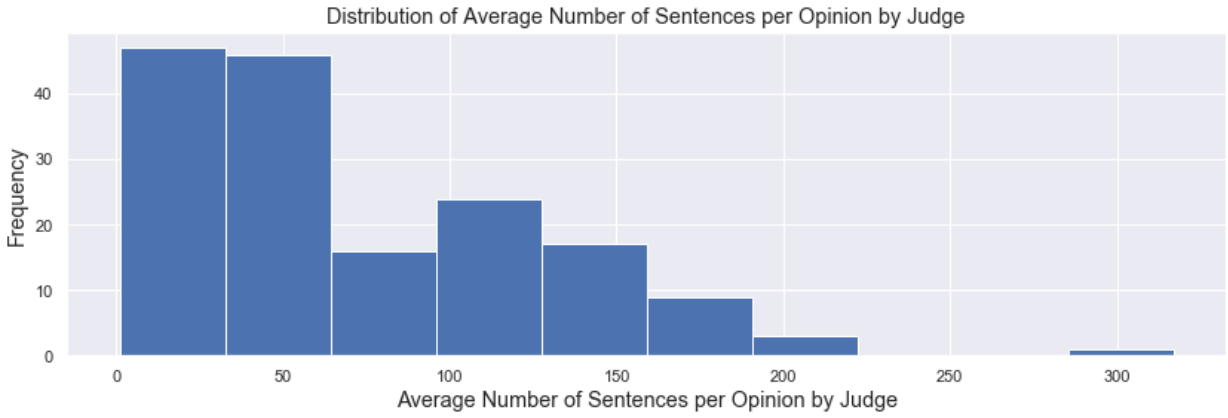
Next, I grouped the features by judge to examine the resulting distributions and any remarkable trends.

1. Average Opinion Word Counts by Judge



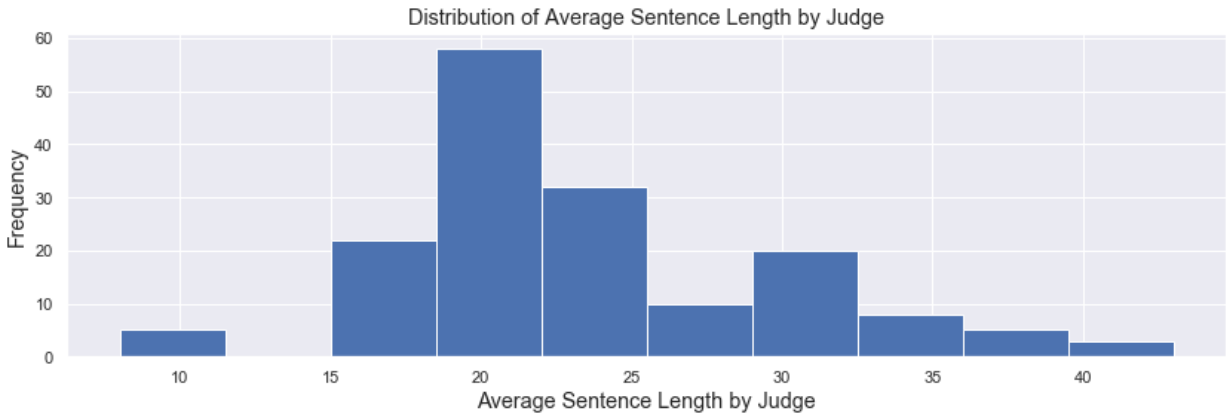
The distribution demonstrates the average length of opinions per judge varies quite significantly. While just over 60 judges average in the 1000-word opinion range, approximately 50 judges average more than double that amount per opinion, with the most verbose judges averaging over 4000 words per opinion (those poor law clerks!).

2. Average Number of Sentences per Opinion by Judge



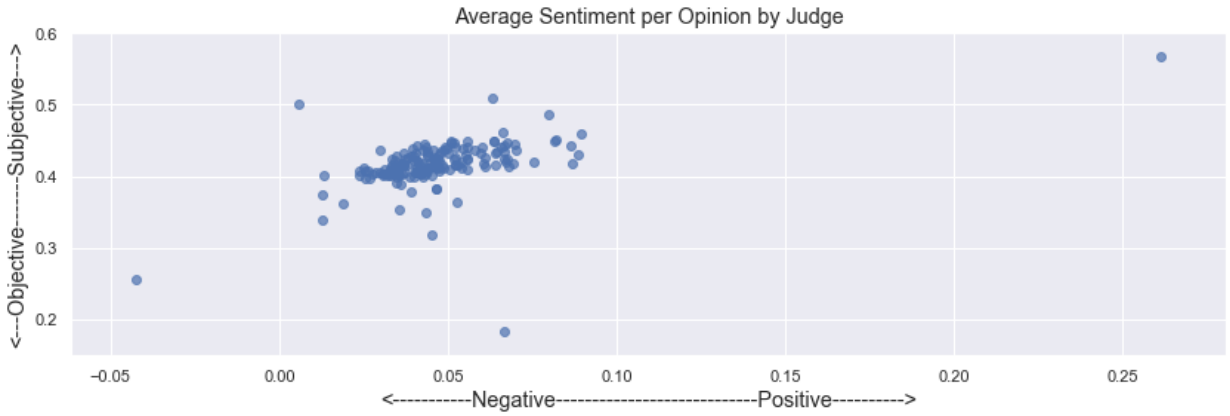
The average number of sentences used by judge skews left, with approximately half of the judges averaging 60 or less sentences per opinion. There are a group of approximately 50 judges that use well over 100 sentences per opinion, though, with a few outliers averaging in excess of 200.

3. Average Sentence Length by Judge



This distribution is quite interesting. There are a handful of judges that prefer quite concise sentences of approximately 10 words, while the few judges on the other end of the spectrum average nearly 40 words per sentence. The most popular sentence length is in the 17-25 word range, where over half of the judges fall.

4. Average Sentiment of Opinions by Judge



The data demonstrates a quite localized cluster of opinions leaning slightly objective and positive where most judges fall. There do appear to be about a quarter of the judges that deviate significantly from the cluster, however, with a few extreme outlier judges with very distinct average sentiments.

E. Conclusions

Exploring the opinion structure and sentiment data has provided some valuable insights into the text that should assist in interpreting and analyzing model performance. I have learned the following important characteristics about the corpus:

- There are a few outlier judges that issued abnormally long opinions, which has skewed the mean and standard deviation for the word count and sentence count features. It may be necessary to remove those outliers if they interfere with the model's ability to accurately identify those judges;
- The courts issued progressively more opinions over time until the late 1900s, when the volume peaked. Volume of opinions has steadily decreased in modern times;
- Court opinions have gotten progressively longer over time, though interestingly the average sentence length has decreased over time. This could be indicative of a stylistic writing change by the more modern judges;
- There appears to be a fairly wide distribution of average words, sentences, and sentence length per opinion by judge. We can use these features to gain insight into a particular judge's distinct writing style relative to his or her peers; and
- Most North Carolina judges tend to write slightly objective and positive opinions, though about a quarter of them deviate significantly from this trend.

V. Machine Learning

Having completed exploratory data analysis, the next step I took was to start building some “out-of-the-box” benchmark machine learning classifiers to see how they would perform.

A. Initial Benchmark Model Performance

I elected to start simple by feeding the models only the handful of characteristic features about the opinions that I built using TextBlob. I prepared the data by converting the categorical features (year, opinion type) into dummy variables and normalizing the continuous variables (word count, sentence count, average sentence length, and polarity). The subjectivity feature values were already on a 0-to-1 scale.

Here are the results:

Model	Training Set Accuracy	Test Set Accuracy
Linear SVM	0.3022	0.2856
Logistic Regression	0.2898	0.2699

Random Forest	0.9885	0.2415
SGD Classifier	0.2313	0.2234
Multinomial Naive Bayes	0.2104	0.2000

The best performing models were able to correctly predict the judge who wrote the opinion just over 25% of the time on the test set. The top three performing models also appear to have overfit on the training set (the Random Forest Classifier to a much more significant degree). While this is far from our ultimate goal, it does demonstrate that these features contain some signal.

B. Vectorized Text Features

Next, I used TfidfVectorizer to vectorize the text of the opinions into features to see how the models performed based on the particular words used in each opinion. I set the min_df feature of the vectorizer to 0.02, meaning that a word had to appear in at least 2% of the opinions in the corpus in order to be included in the relevant vocabulary. This produced a total vocabulary of 3,677 words.

Here are the results using a vocabulary of 3,677 words:

Model	Training Set Accuracy	Test Set Accuracy
Linear SVM	0.9044	0.6814
Logistic Regression	0.5816	0.4922
SGD Classifier	0.7290	0.5581
Random Forest	0.2464	0.2277
Multinomial Naive Bayes	0.2324	0.2111

Three of the classifiers performed substantially better with the vectorized text. I elected to drop the Random Forest and Naive Bayes models given their inferior performance.

Next, I decided to decrease the min_df value to 0.001 and re-vectorize the opinions, which resulted in a substantially larger vocabulary of 16,905 words. I trained the

models on this expanded vocabulary to see how it impacted model performance. Here are the results:

Model	Training Set Accuracy	Test Set Accuracy
Linear SVM	0.9885	0.6753
Logistic Regression	0.6064	0.4778
SGD Classifier	0.8424	0.5781

Expanding the vocabulary from 3,677 to 16,905 words did not meaningfully improve model performance. The Linear SVM and Logistic Regression models actually performed worse on the test set and the SGD Classifier improved by only 2%. This isn't necessarily surprising and demonstrates that expanding the vocabulary even further is not likely to improve model performance, as it is adding less-utilized, and likely less-significant, words to the vocabulary. Expanding the vocabulary further would also open the door for more OCR errors to be included, which I know is an issue with this dataset.

For these reasons, I will proceed with the 3,677 word vocabulary, as it will make model building more efficient without sacrificing much, if any, performance.

C. Combined Feature Performance

Next, I decided to provide the models with both the initial characteristic features as well as the vectorized vocabulary features to see how much, if at all, it improved the model performance.

Here are the results:

Model	Training Set Accuracy	Test Set Accuracy
Linear SVM	0.9655	0.7419
Logistic Regression	0.7364	0.6236
SGD Classifier	0.7165	0.5903

Combining the features improved the accuracy of the Linear SVM and Logistic Regression models significantly. Both models are producing favorable accuracy percentages considering that the model has to identify the correct author from a pool of over 150 judges. The SGD Classifier did not improve as significantly as the other two models and is now producing the lowest accuracy, so I elected to move forward without it.

D. Model Hyperparameter Tuning

The key hyperparameter for both the Linear SVM and Logistic Regression models is the C parameter, which determines the regularization strength applied by the model. Here are the results of tuning the C parameter for both models:

Model	Solver	C Value	Training Set Accuracy	Test Set Accuracy
Linear SVC		1	0.9655	0.7419
Logistic Reg.	lbgfs	100	0.9998	0.7295
Logistic Reg.	lbfgs	1000	0.9999	0.7244
Linear SVC		10	0.9976	0.7232
Logistic Reg.	lbfgs	10	0.9650	0.7232
Logistic Reg.	lbfgs	10000	0.9999	0.7202
Linear SVC		100	0.9996	0.6945
Linear SVC		1000	0.9996	0.6843
Linear SVC		0.1	0.8105	0.6642
Logistic Reg.	lbfgs	1	0.7517	0.6322
Logistic Reg.	newton-cg	1	0.7517	0.6322
Linear SVC		0.01	0.5206	0.4569
Logistic Reg.	lbfgs	0.1	0.4171	0.3786
Logistic Reg.	lbfgs	0.001	0.1384	0.1355

The chart demonstrates that both models perform better with higher C parameter values, though it also results in overfitting on the training data. The best Linear SVC model is generating an accuracy percentage of 74.19% on the test set and the best Logistic Regression Classifier is generating an accuracy percentage of 72.95%. Both models are performing quite well at this point.

E. SpaCy Vectors

As a final step, I will add some SpaCy word embeddings vectors for each opinion to see if they improve model performance. The SpaCy package converts documents into 300 dimension vectors that attempt to capture the substance of the text with numerical values.

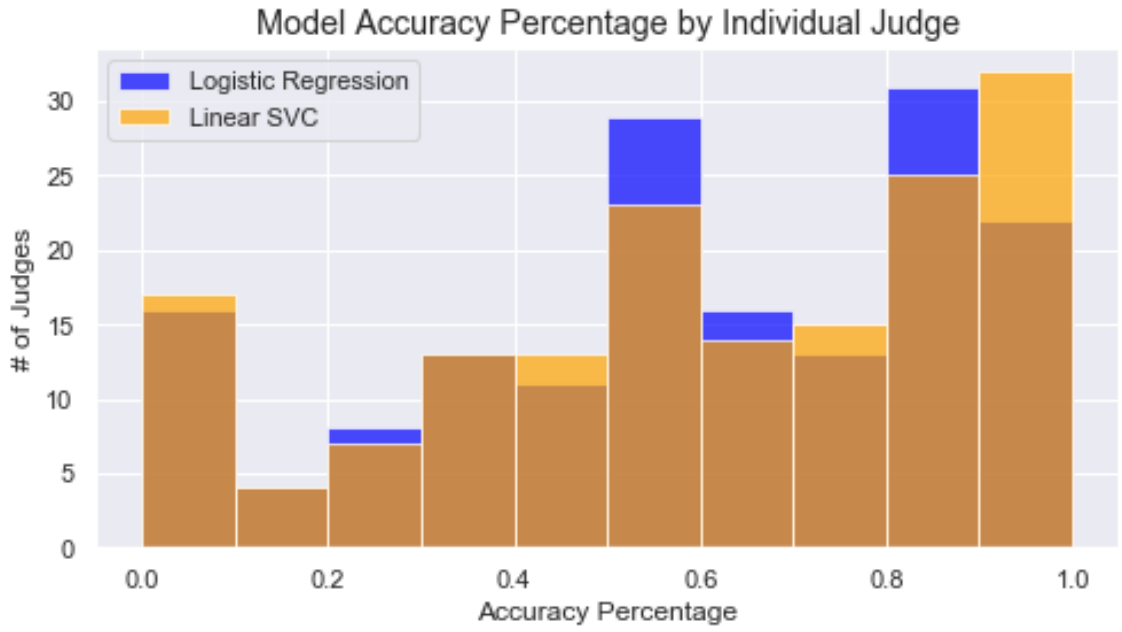
I added these additional 300 features to the existing dataset and re-trained the models. Here are the results:

Model	Training Set Accuracy	Test Set Accuracy
Linear SVM	0.9747	0.7676
Logistic Regression	0.9999	0.7542

The SpaCy vectors improved the accuracy of both models by approximately 2%, demonstrating that the substance of the opinions provides signal for the models. With both models generating an accuracy percentage above 75%, I have succeeded in building two models that will provide value to my clients.

VI. Analysis of Results

While I know that in the aggregate both models were able to identify the correct judicial author for just over 75% and 76% of the opinions within the test set, respectively, it would be helpful to get a more detailed understanding of the models' performance. I will start by graphing a distribution of how the models performed for each judge given that we have approximately 160 judges in our dataset:



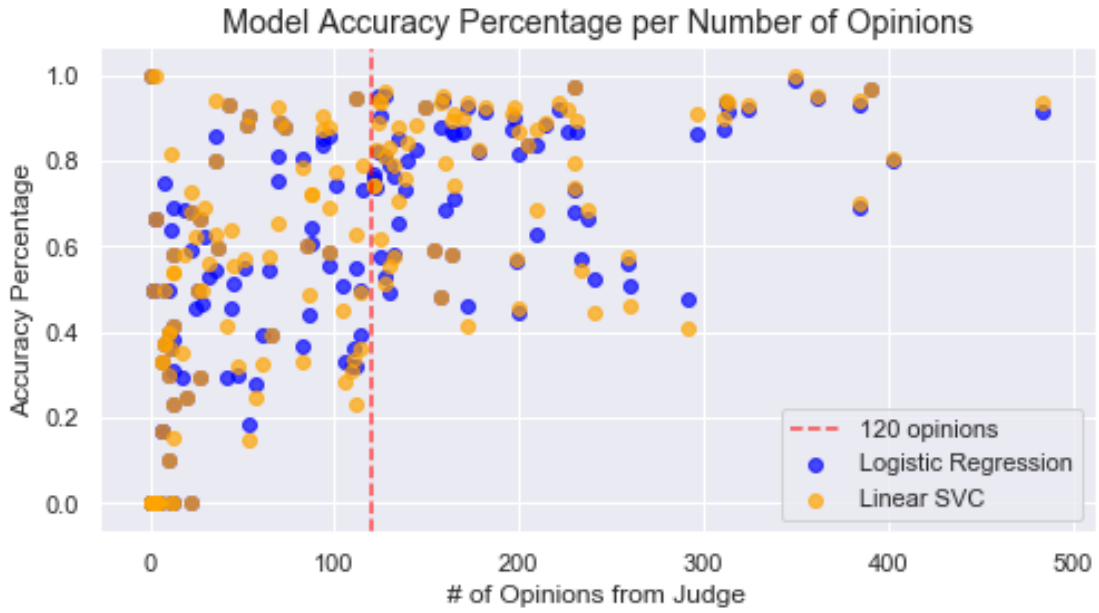
The distribution is quite informative. As a threshold matter, it appears that both of our models are producing similar distributions overall, with the Linear SVC model producing approximately ten more judges with an accuracy rate 90% or greater.

Both models produce distributions with three local maximums: a group of approximately 60 judges in which the models produce highly accurate results, a group of approximately 25 judges in which the models produce an accuracy of 50-60%, and a group of approximately 15 judges in which the models produce an accuracy of 10% or less.

I will now investigate to see what I can learn about this wide discrepancy in performance.

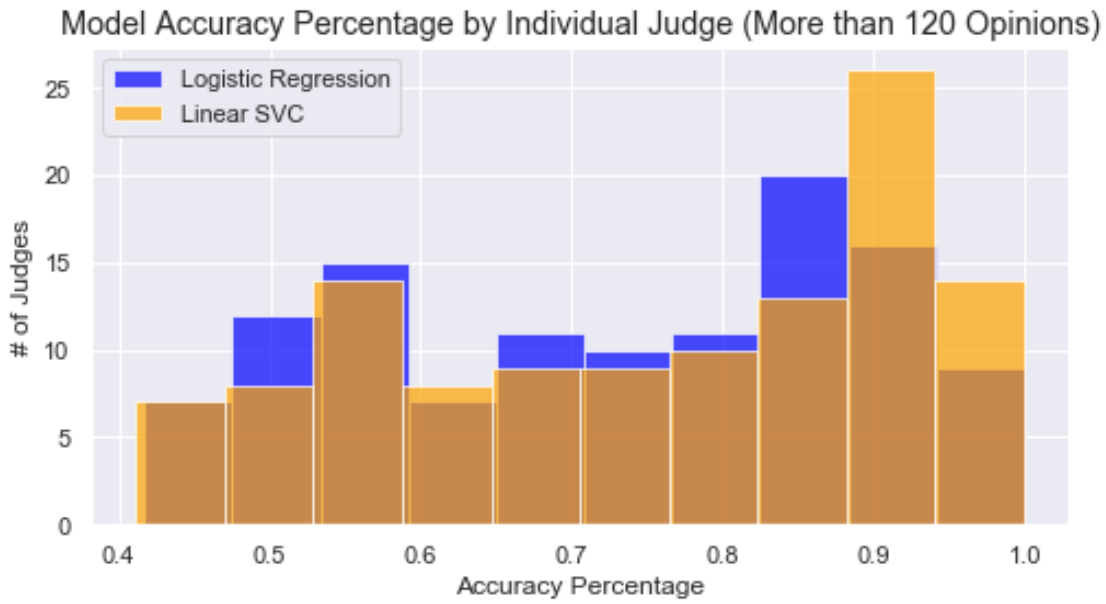
A. Number of Opinions Per Judge

The first area I will investigate is whether the number of opinions in the test set for each judge is influencing the results. Below, I graph the number of opinions of each judge in the test set along with the accuracy percentage for each judge produced by both models. Because there were two labels that had a considerably greater number in the test set ('per_curiam' and 'clark_walter'), I removed these from the below graph so that I could scale the graph to get a more meaningful result. (NOTE: Both models produced very high accuracy numbers for both of those categories).



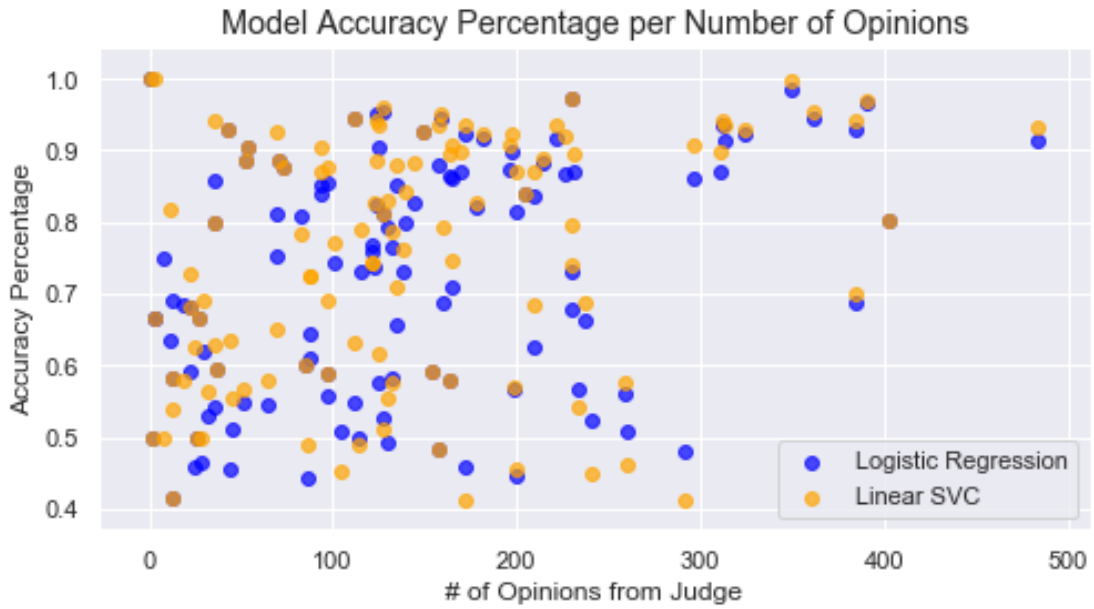
This graph demonstrates that the models achieve a minimum accuracy score of 0.40 for any judge that has at least 120 written opinions in the test set. This is encouraging, as it shows that the models are adept at identifying some sort of signal in the text of a judge's opinions so long as it has a large enough sample size. And, in fact, the graph demonstrates that often times the models are able to identify some meaningful signal even if a judge's opinion sample size is less than 120.

Let's remove the judges that produced an accuracy below 0.40 and re-evaluate the distribution:



While both models perform well on a considerable portion of the remaining pool of judges (with the Linear SVC model outperforming the Logistic Regression model on the high end), there still is over 30 judges that produce an accuracy lower than 0.60.

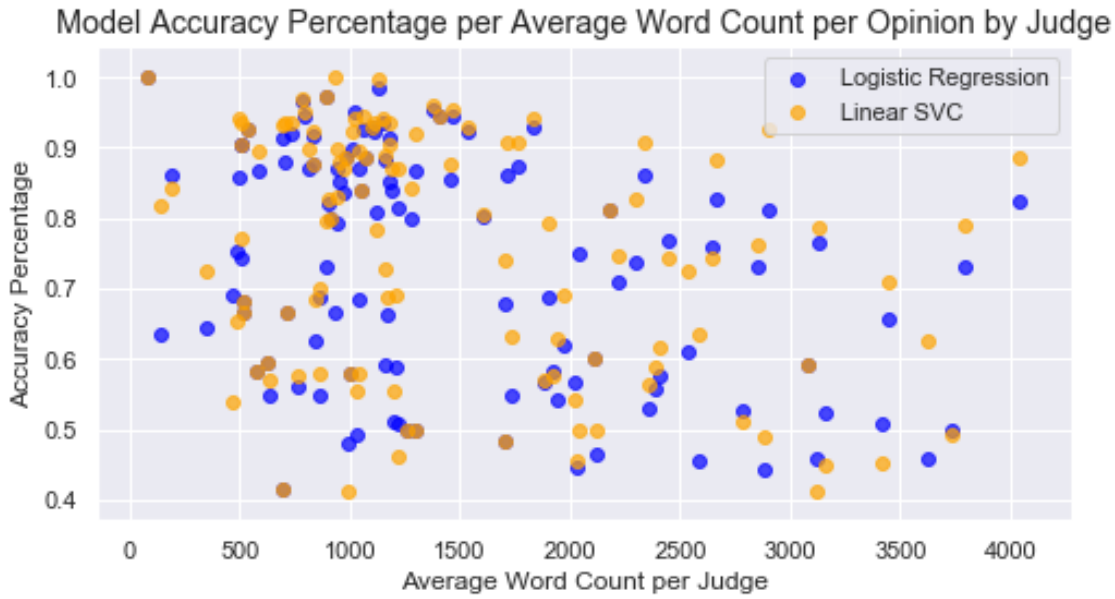
Let's see what else we can learn about the models' performance. First, I will re-create the number of opinions graph:



Pearson correlation coefficient (Logistic Regression): 0.28993114216460114
Pearson correlation coefficient (Linear SVC): 0.23835028161794186

The graph demonstrates that both models produce mixed results until the sample size of opinions eclipses 300, in which case performance improves. The Pearson correlation coefficients of 0.28 and 0.23, respectively, confirms that the number of opinions is slightly positively correlated.

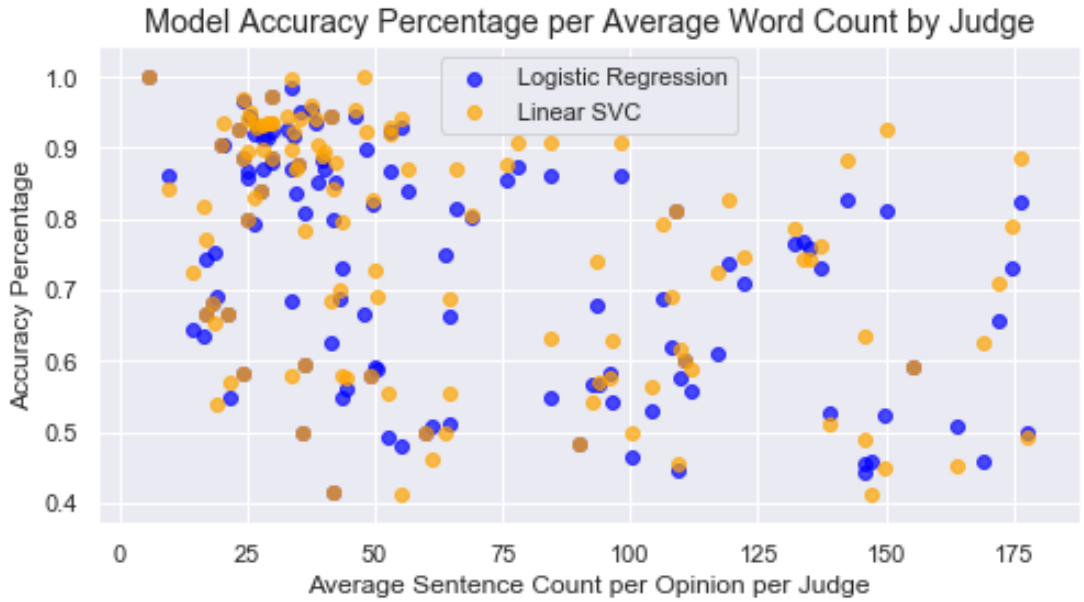
B. Average Word Count Per Opinion



Pearson correlation coefficient (Logistic Regression): -0.3575672392738484
Pearson correlation coefficient (Linear SVC): -0.3156747786731042

Interestingly, it appears that the model performance did not improve based on the average number of words a judge used in his or her opinions. In fact, the Pearson correlation coefficient suggests that the models' performance decreased as the judge's average word count per opinion increased.

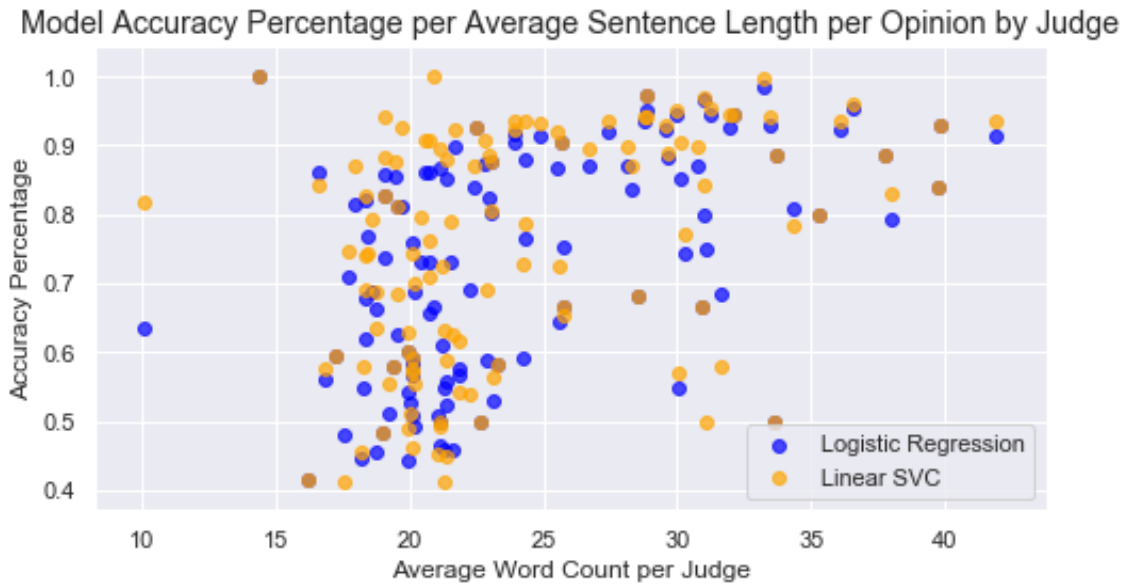
C. Average Number of Sentences Per Opinion



Pearson correlation coefficient (Logistic Regression): -0.4584591111136079
Pearson correlation coefficient (Linear SVC): -0.3879253712808552

Similar to word count, it appears that the models both performed worse as the average number of sentences per opinion increased. Let's see if we get a similar result from average sentence length.

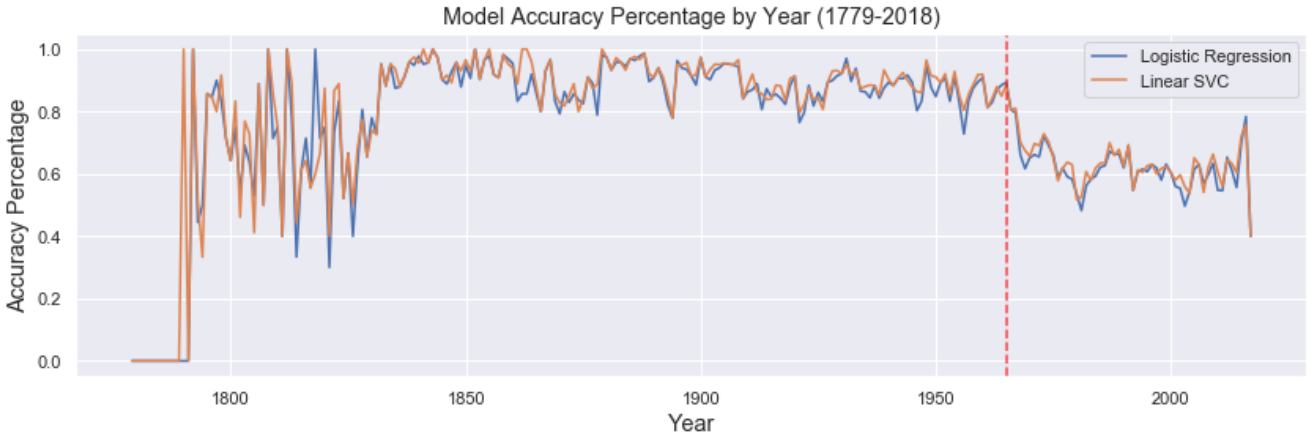
D. Average Sentence Length Per Opinion



Pearson correlation coefficient (Logistic Regression): 0.4909574447484058
Pearson correlation coefficient (Linear SVC): 0.38431717487223266

The models clearly produced a positive correlation between average sentence length of opinion and the resulting accuracy. So, while the models perform better on opinions with shorter overall word counts and number of sentences, it actually does better with longer individual sentences within an opinion. Next, I will graph the models' respective performance based on year of the opinions.

E. Model Performance Based on Year of Decision



Both models experienced a sharp decline in performance starting in approximately 1965 and then stabilizing at a 55-65% accuracy rate thereafter. I know from my exploratory data analysis that this time period correlates directly with a sharp increase in the volume

of yearly decisions issued in the dataset. There also was an increase in the average length of court opinions during this time period. This seems to indicate that the models' performance suffers as the volume of decisions within a particular year and/or the volume of words per decision increases.

I will now dig a bit deeper to see if I can glean any more information about what may be causing the sharp decline in the performance of the models.

F. Model Coefficient Values

For this analysis, I divided the data into two separate groups: decisions from the 1850-1965 time frame and post-1965 decisions (using the average decision date per judge provided in the df_groupby dataframe). I then calculated the average coefficient values for each group for each model and compared which features were assigned the highest average coefficient values for each group for each model. Here are the results:

	LR Classic	LR Modern	SVC Classic	SVC Modern
1	tbe	sentence_count	the	the
2	spacy_9	word_count	of	that
3	avg_sent_length	defendant	count	of
4	appeals	app	spacy_227	spacy_87
5	in	year_1980	that	spacy_298
6	was	spacy_17	spacy_248	spacy_46
7	hence	year_1979	spacy_208	spacy_248
8	spacy_263	concur	spacy_286	spacy_26
9	spacy_227	year_1983	spacy_140	spacy_59
10	spacy_277	year_1982	tbe	spacy_124
11	spacy_152	2d	was	spacy_113
12	decision	spacy_85	spacy_60	spacy_60
13	spacy_163	year_1981	spacy_163	year_1990
14	cited	spacy_179	spacy_67	spacy_3

15	year_1957	year_1985	spacy_240	year_2006
16	year_1956	spacy_140	spacy_205	spacy_67
17	year_1959	year_1984	tax	year_1997
18	spacy_201	year_1993	spacy_121	year_1979
19	year_1892	trial	spacy_45	year_2001
20	year_1939	year_1976	they	year_1983

For the Linear Regression model, it appears that it was able to identify the older decisions at a higher rate by labeling a few key words and spacy vector values with the highest coefficient values. It also placed importance on the decision year, with particular years making up roughly half of the top 30 coefficient values. For the more recent decisions, it appears that the model labeled the word count and sentence count features as most important, with the spacy vector values not as important as for the older opinions. Approximately half of the features were particular years for the more recent decisions as well, demonstrating that the logistic regression model found signal in the year a decision was issued.

As for the Linear SVC model, it interestingly did not have a single year feature in its top 30 coefficient values for the older decisions. It prioritized the spacy vector values primarily, with a few word features sprinkled in. The model also prioritized spacy vector values for the more recent decisions, but it also included a number of particular year features for those opinions (unlike the older decisions). Given that the model performed better at identifying judicial authorship for the older decisions, it appears likely that the Linear SVC model is identifying more signal in the substance of the opinions (which the spacy vectors represent) rather than the year an opinion is issued.

G. Individual Judge Word Usage

Next, I have selected at random three pre-1965 judges (Barnhill, Higgins, Varser) and three post-1965 judges (Erwin, Hudson, Stephens) to analyze to see if I can uncover any observable distinctions in their word usage. I started by generating word clouds for each judge's set of opinions.

1. Word Clouds

Interestingly, it appears that all six judges use by and large the same words most often in their respective opinions. A substantial amount of the most frequent words appear to be common english stop words. I recreated the word clouds and excluded the traditional english language stop words to see if that generated more meaningful results:

Even when excluding traditional stop words, the most common word usage for each of the six judges seemed to overlap substantially with traditional legal words (e.g., plaintiff, defendant, court, trial, state, case, etc.). This makes sense from a practical standpoint and also provides some indication as to why more individual word features didn't receive the larger coefficient values (i.e., were not prioritized by the models).

2. Highest Tfidf Word Values

Lastly, I compiled a chart demonstrating the 20 words that received the highest cumulative tfidf value for each of the six judges. Perhaps that will provide more meaningful results.

	Barnhill	Higgins	Varser	Erwin	Hudson	Stephens
1	the	the	the	the	the	the
2	of	of	tbe	of	of	of
3	tbe	to	of	to	to	to
4	to	and	to	and	and	and
5	and	in	in	in	that	that
6	in	is	and	that	in	in
7	is	that	is	defendant	defendant	defendant
8	that	was	that	is	court	court
9	it	2d	it	2d	2d	2d
10	was	for	not	was	for	on
11	or	not	this	not	on	trial
12	by	by	by	court	at	was
13	as	on	as	or	is	is
14	not	court	or	for	not	at
15	be	evidence	be	by	plaintiff	for
16	for	defendant	supra	on	was	not
17	plaintiff	or	for	plaintiff	by	or
18	this	as	co	trial	or	by
19	defendant	be	was	as	as	as
20	on	plaintiff	possession	state	trial	plaintiff

Similar to the word cloud results, the words with the highest tfidf values overlap substantially between the six judges and almost exclusively consist of either common stop words or classic legal jargon. It appears to be pretty definitive at this point that the models were able to identify more signal in the non-individual word features than the individual word tfidf features to identify judicial authorship.

VII. Summary/Conclusions

In this project, I was able to build two models that could identify the judicial author of a court opinion at an overall accuracy rate of approximately 75% and 76%, respectively. Given that the models had a pool of over 160 judges to choose from, generating accuracy rates as high as 75% indicates that the models were successfully able to learn some meaningful distinctions between each of the judges' particular written opinions.

Overall, this project demonstrates that attempting to build machine learning models to identify judicial authorship of case decisions is a worthwhile endeavor that can provide real, tangible value to the legal industry. The particular use cases for such a model are many. For example, a practitioner could use a model to analyze a brief to be filed in order to see which judge or judges the text of the brief most resembles. One could also feed opinions of a particular judge that were not used during training into the model to see which judge or judges the model finds to be most similar. Or a practicing attorney could even simply perform some deeper analysis on the results of a fully-trained model to uncover more details about a particular judge (e.g., common word usage, grammatical patterns, etc.). The potential for accurate judicial authorship classifiers really is substantial.

The following are some more specific conclusions I reached about the particular models I built in this project:

- The Linear SVC model slightly outperformed the Linear Regression Classifier by producing an accuracy rate of 90% or higher for substantially more judges and, therefore, is the model I would recommend;
- Both models were able to produce a minimum accuracy percentage of 40% for all judges in which they received a large enough sample size of opinions, and both models' performance was directly positively correlated with the number of opinions for a particular judge. This demonstrates that both models very likely were able to extract real, tangible signal from the features used to train the models;

- Both models experienced a significant dropoff in performance as the number of judges issuing opinions in a particular year increased. During this same time period, the average number of words and sentences per decision increased, and the average sentence length decreased, significantly. It is unclear which of these trends (or combination of trends) was the cause of the model performance dropoff (though if I had to venture a guess, it most likely is the increase in the number of judges issuing opinions within the same year);
- Both models on average assigned higher coefficient values to the non-individual word features. The higher-performing Linear SVC model placed a greater importance on the spacy vector values, indicating that it was likely able to glean some signal from the particular substance of the text opinions; and
- There was substantial overlap among the most frequent and/or important words identified by the models for each particular judge, which likely explains why on average the individual word features were assigned less significant coefficient values.

VIII. Future Project Work

While I was able to build two high-performing judicial authorship classifiers, I believe that I could improve the models (the Linear SVC model in particular) through future work to provide even more meaningful results. I believe the following future work could improve the models:

- **Building a More Precise Corpus Vocabulary.** Analyzing the results of the models showed that they were not identifying a distinct vocabulary for any particular judge. I believe it would be worthwhile to spend time crafting the particular vocabulary to be used in the model, such as eliminating traditional stop words, removing common legal jargon, and limiting the vocabulary to particular parts of speech (nouns, verbs, adjectives, etc.). This could lead to more meaningful insights into a particular judge's opinion style;
- **Removing the Year of the Opinion.** While the year of the opinion very likely assisted model performance, it does not provide much meaningful insight for the likely use cases for this model. It therefore may be beneficial to remove that feature from the dataset and require the models to use other features to identify judicial authorship; and
- **Exploring the Meaning of the SpaCy Vector Values.** The results indicated that both classifiers, and in particular the Linear SVC model, were able to identify

signal from the SpaCy vectors to correctly identify judicial authorship. I believe it would be worthwhile to unpack the SpaCy vector values to determine more specifically which portions of the opinion text the models were identifying as significant. Being able to identify particular substantive distinctions among judges would be extremely valuable to the likely users of the models.