US Federal Circuit Court: Language Features and Court Decision Prediction

Yurui Mu   Li Lin Qin   Xue Yang   Adviser: Daniel L. Chen   Elliot Ash

NYU DATA SCIENCE

Goal

In this Machine Learning project, our goal is to manipulate, generate and analyze multiple language features in federal circuit court cases ranging from 1880 – 2013 and predict court decisions (affirm-reverse) along with other useful datasets. We also aim to predict citations by judges in future cases.

Introduction

In United States, there are 13 appellate courts that sit below the U.S. Supreme Court. All 94 federal judicial districts are organized into 12 regional circuits, each of which has a court of appeals. The appellate court's task is to determine whether or not the law was applied correctly in the trial court. In Appeals courts, judge panels consist of three judges. Given the majority opinion written by judges, our goal is to predict whether Courts of Appeals would affirm or reverse the trial court decisions, along with analyzing the causal factors influencing judges' decisions. Potential factors include judges' personal sentiments recorded in the texts, whether this case could help legislation toward a certain direction and influences from the past cases.

Data Interpretation

Although with a large amount of cases over the years, there are only 273592 cases that have a clear affirm or reverse labeling. Specifically, there are 201048 affirmed cases, and 72544 reversed cases.

Due to the diversity of features we will be analyzing, the following datasets are used:

1) Main texts from 1880 – 2013 that documented the judges' majority opinions, concurring opinions and dissenting opinions.
2) Vote-level case information: This dataset contains three data entries for each case that store information of the three judges besides case info.
3) Case-level case info: contains 387898 entries with features like citations and majority word counts.
4) Facts descriptions from year 1980 to 1985, including 164 text files.
5) Policy dataset from manifesto project, in which included political campaign speeches and political statements with hand-labeled topics. Some example topics are: culture, immigration, nationalization, democracy. Valence column indicating positive or negative of each topic.
6) LIWC dictionary dataset, which included 73 dictionaries that span a variety of topics, for example certainty and tentative words.

Generated Language Features

Some important features we created include:

- N-gram features: used NLTK package and grammar sparse tree to capture meaningful expressions up to 3-grams. Words and phrases that appear in at least two cases a year in consecutively ten years are preserved. The final feature consists of 25215 words.
- LIWC word count: we used 73 dictionaries to count number of words belonged to each word categories. Some example dictionaries include: comparison, interrogation, social, sad, anger, power…
- Average word vector: Word2vec model takes a large corpus of text as input and produces a vector space. Each unique word in the corpus was assigned a corresponding vector in the space. Words that share common contexts in the corpus are located in close proximity to one another in the space. Our output is an averaged vector of all the words that appeared in each paragraph.
- Sentiment metrics: using NLTK package in Python, we assign a sentiment value to each paragraph in each case, -1 being negative, 0 being neutral and 1 positive. Later, we will take dot product of sentiment and n-gram frequency across paragraphs for each case.
- TF-IDF similarity between cases: Apply Tf-idf to transform each case into real-valued vectors. We then compare cases pairwise by calculating the cosine similarity of their vectors.
- Paragraph about facts, law, or analysis: Incorporated another dataset that contained facts descriptions in 1980 – 1985. The below graphs show some of the most common words that appeared in law cases (left) and facts (right):

Policy Topic Model

By running the policy topic model, we can predict the political topic a case is referring to, and also incorporate the results into our final language features. Since the political datasets included a variety of languages, we filtered out non-English files using the package PyEnchant. By setting the threshold to 0.8, we keep documents with more that 80% words being English.

We used two methods to generate features from texts: Tf-idf, and Doc2vec. Since this is a multiclass classification problem, machine learning models SVM and Random Forest seemed suitable. Unfortunately, both models performed poorly using Doc2vec generated features. Our final score using Tf-idf features are 0.32 and 0.507, from SVM and Random Forest respectively.

Next Steps

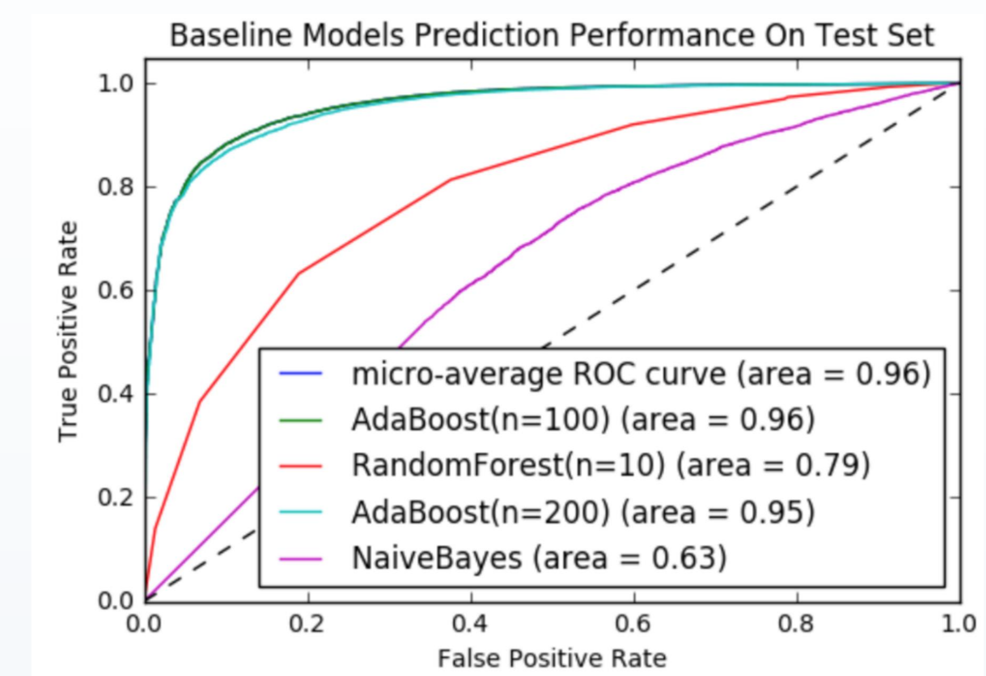
Features:

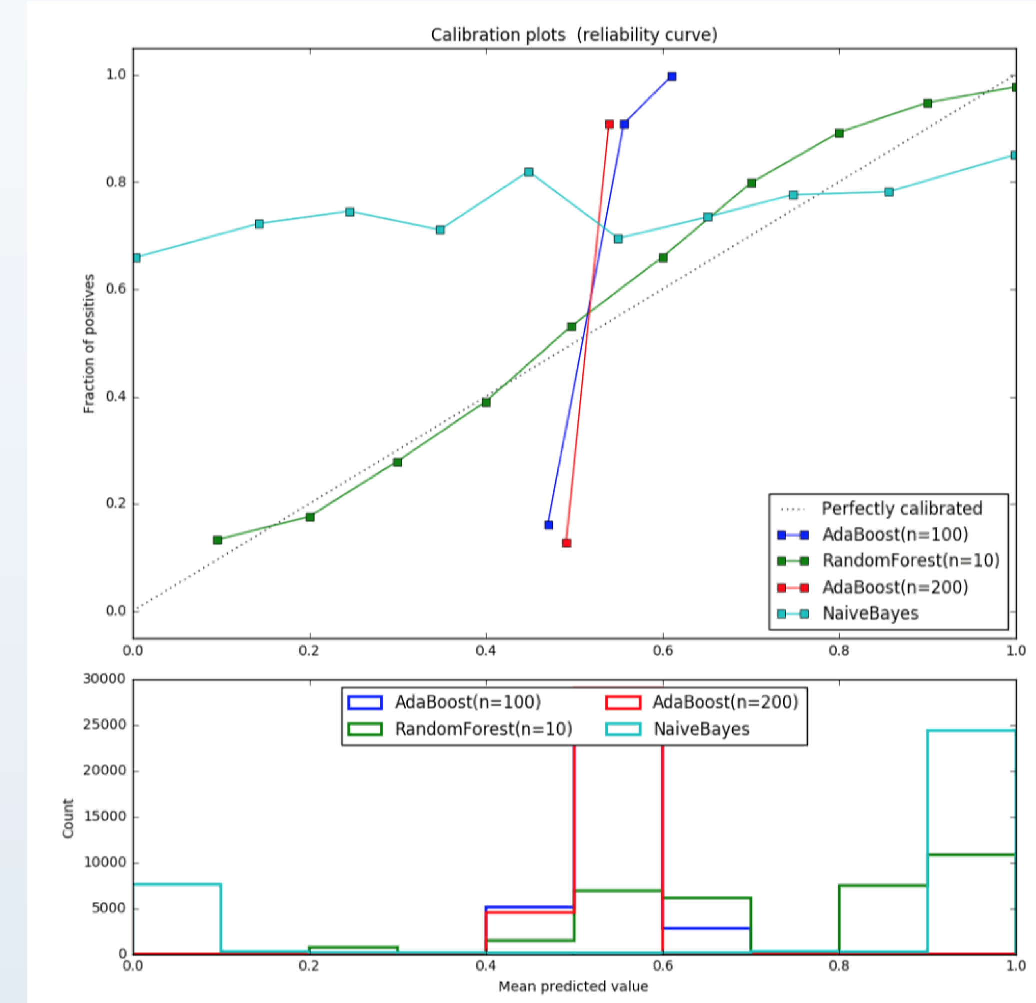
- Which party is being discussed (plaintiff, defendant, petitioner, respondent, appellant, appellee)
- Compute TF-IDF similarity between judges

Modeling:

- Continue implement different models with all the new text features and judges' backgrounds

Baseline Model





In our baseline model, we use all of the documents in majority opinions, concurring opinions and dissenting opinions from each cases as input. Taking the n-gram vocabulary generated from our NLTK n-gram generator as vocabulary input, we use sklearn CountVectorization as feature extraction method, using Scipy LIL sparse matrix to stack on each vector, and convert to CSR sparse matrix as the final feature input.

Then we use AdaBoost, Random Forest, and Naïve Bayes model as baseline binary classifiers. From our model output, we could see that AdaBoost has an extremely high score, but poorly calibrated. While Random Forest model seems better calibrated, it does poorly with an 0.79 AUC score.

References

"Court Role and Structure." United States Courts. N.p., n.d. Web. 09 May 2017. http://www.uscourts.gov/about-federal-courts/court-role-and-structure