Text Analysis and Voting Rights: Analyzing News Coverage of the 2020 Election

Researchers -

Daniel Chung, Bon Hee Koo, Tiffany Liang, Sean Wei, Shiny Weng

Advisors -

Laura Nixon, Lana Elauria, ReThink Media, Alex Bruefach, UC Berkeley

Introduction -

Our Fall 2021 Discovery Project aims to understand how voting rights issues were portrayed in news coverage of the 2020 election. This goal motivated the use of text analysis and machine learning methods on a small set of labeled article data to fit models that can classify larger article datasets. The technical goals of this project were to develop classifiers that predict author gender and whether an article is news or opinion. We used the accuracy of our models' predictions to evaluate our success in these goals.

Data -

Our data comes from a dataset of over 12,000 articles about voting rights from October 2020 through inauguration, which were collected from web scraping and from the Factiva news database. A random sample of 4,000 articles had been labeled by human coders from ReThink's partner, Novetta. Data attributes include publication date, author name, publisher, and the text content of the news story itself. Some obstacles this dataset presented were irregularity within the formatting of the .csv file, variations in author attribution across articles, and inclusion of irrelevant articles from the web scraper. The first step of the project was to fix these errors through Python scripts and manual removal and ensure that the data was clean and relevant to our project.

Gender Classification -

One goal of this project was to build a pipeline which classifies the gender of a journalist from their name. Our approach for this problem was to combine predictions from existing gender classification tools: Gender Guesser, Gender API, Genderize, and Namsor. To create our labeled data, we retrieved all unique author names from the article dataset, and classified the authors' gender by searching for the authors' biographies on the Cision PR platform, and determining which gender pronouns were used. For authors who could not be classified in this way, we then hand labeled authors who had names very strongly associated with one gender (e.g. Jose, Mary). For the remaining authors, who had uncommon or typically unisex names (our 'manual' dataset), we manually searched for public biorgaphies or other information to identify which pronouns they used (male, female, or non-binary).

We tested each tool's accuracy on author names in both our 'automatic' and 'manual' data sets (Table 1). Most of the tools also outputted a confidence metric that indicated how certain it is about its prediction. We calculated the accuracy of each tool at a given confidence level (Figure 1), as well as the proportion of data they retained at that confidence level (Figure 2).

Table 1. The accuracy of each classification package with automatically and manually annotated data.

Classification Package	Automatic Data Accuracy	Manual Data Accuracy
Gender Guesser	0.903	0.573

Gender API	0.964	**0.858**
Genderize	0.923	0.744
NamSor	**0.966**	0.853

Figure 1. Accuracy of each tool per confidence score.

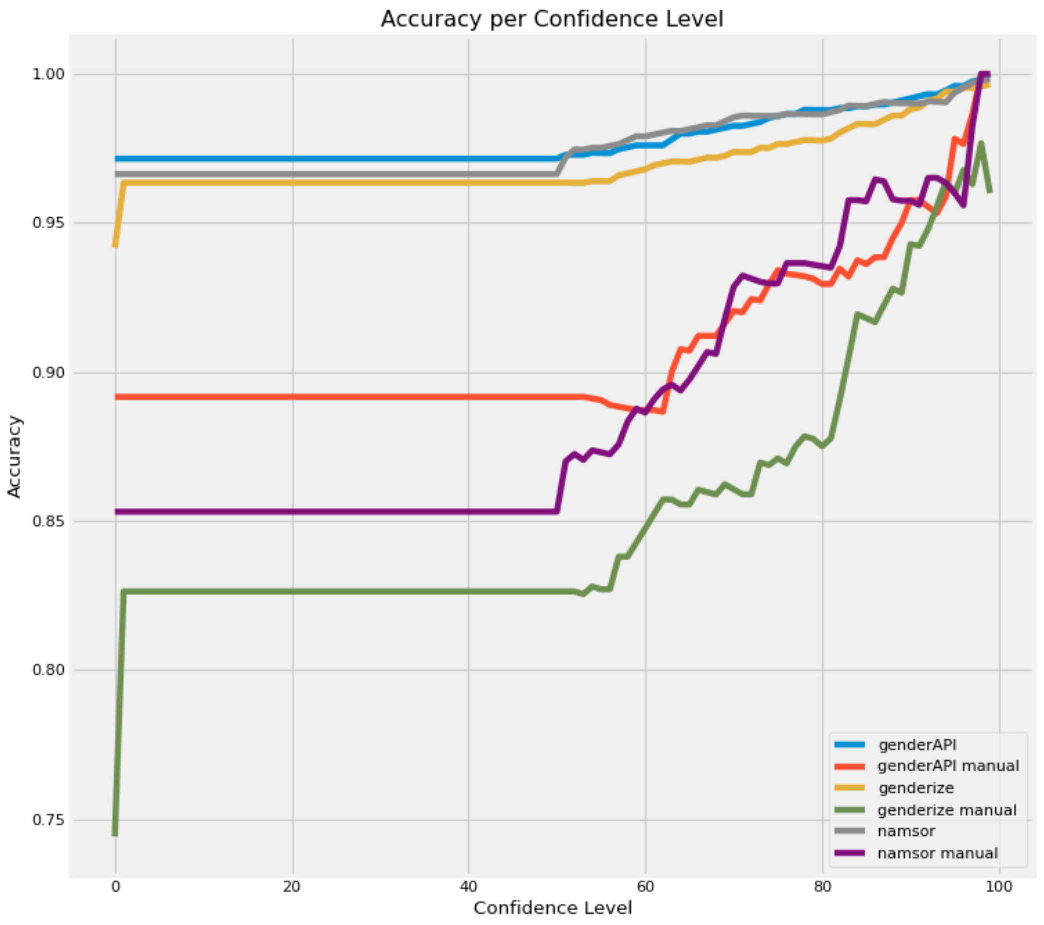
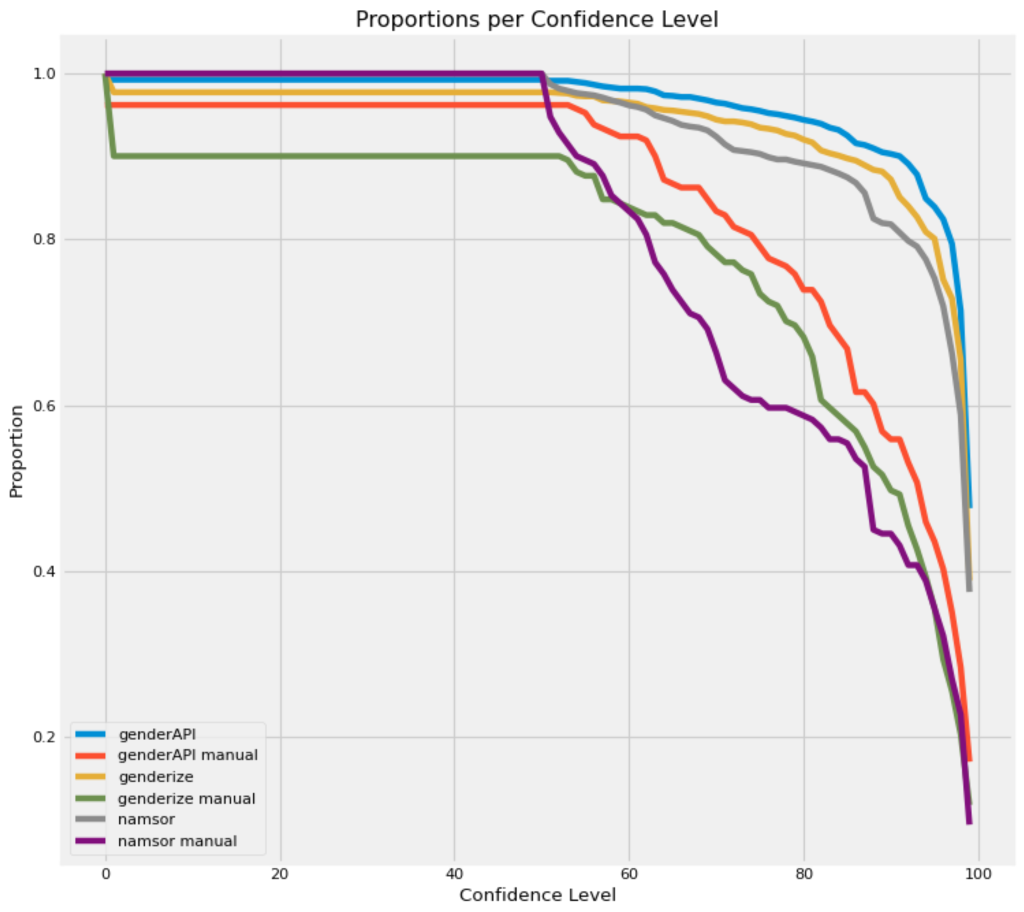


Figure 2. The proportion of the dataset over confidence thresholds given by each tool.



We tested a number of ensemble methods, including combining weighted averages (see Table 3) for the classifier results, but the method that was both most accurate (as well as cost effective), was a cascading method, where names not confidently classified by one API would be passed into another for verification.

Table 3. Accuracy of ensemble methods using weighted averages.

Ensemble Method	**Automatic Data Accuracy**	**Manual Data Accuracy**
Accuracy Aggregate	0.970	0.877
Confidence Aggregate	0.965	0.858

First, we used the Gender Guesser package, a free tool that does not return a prediction for all names, but provides very accurate results for the predictions it does return. We then classified the remaining names using Gender API and NamSor. We flagged any names where NamSor and Gender API disagree about the gender label or cannot make a prediction for manual checking.

News vs. Opinion Classification -

Our second goal was to implement models that could classify articles as news or opinion. This first necessitated a unified feature set, which we constructed primarily from linguistic features identified in this paper (Table 5). We utilized Python, NumPy, pandas, and SpaCy in our feature extraction function, and challenges included handling null values, linguistic edge cases, and software issues. We also identified specific phrases that were much more common in one class of article over the other, but these phrases did not improve prediction accuracy.

Table 5. News and opinion features used in our classifier, with descriptions

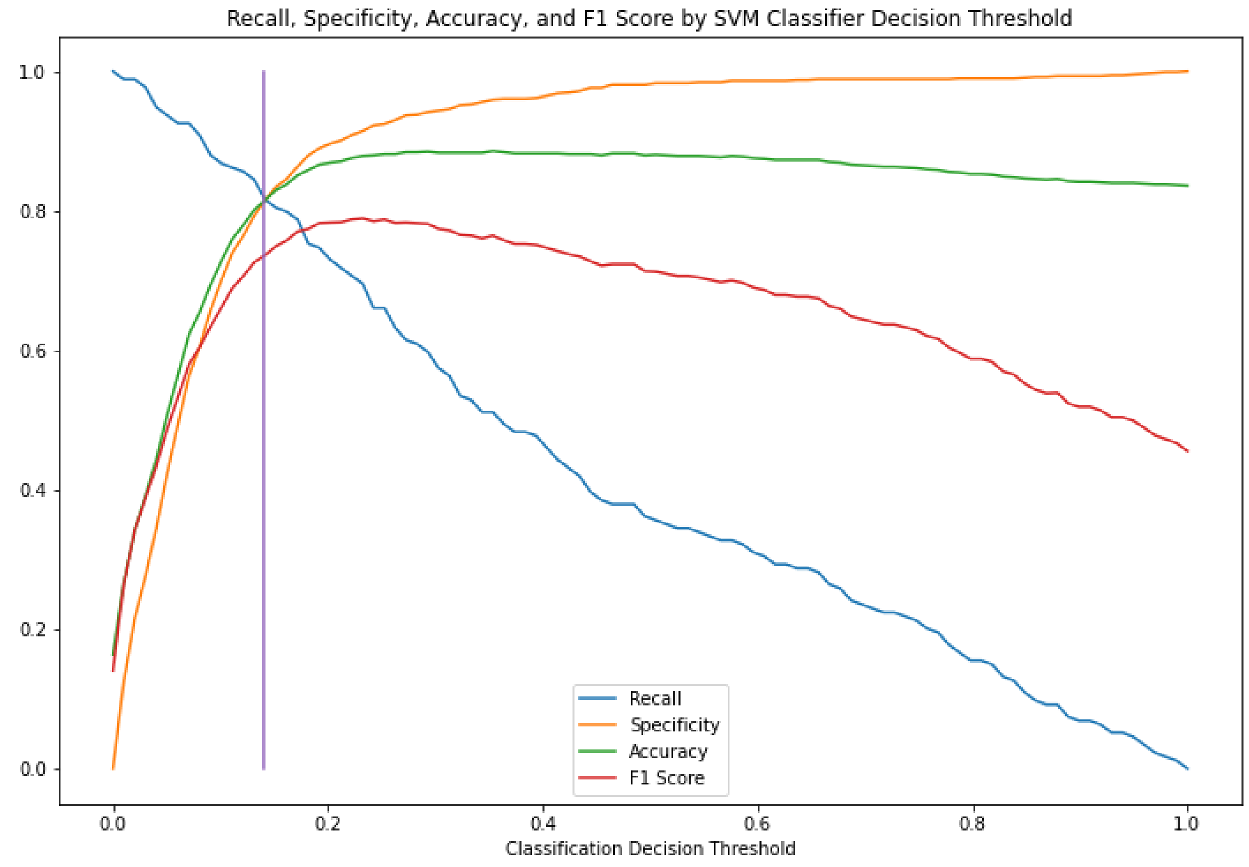
Features	**Implementation**
sent_length	Avg. sentence length in tokens (inverted)
token_length	Avg. token length in characters (inverted)
first_p	Count of first person pronouns
second_p	Count of second person pronouns
third_p	Count of third person pronouns
first/second_p	Count of first and second person pronouns
questions	Count of question marks
exclamations	Count of exclamation marks
semicolons	Count of semicolons
commas	Count of commas
periods	Count of periods
num_casual	Count of causal verbs
num_temporal	Count of temporal verbs
num_contrastive	Count of contrastive verbs
num_expansive	Count of expansive verbs
digits	Normative frequency of digits
num_modals	Count of modal verbs
num_vos	Count of verbs of saying (VOS)
num_future	Count of future tense verbs
num_negation	Count of negations
negation_suffix	Count of instances where a negation suffix is used
num_citations	Count of citations
avg_citation_len	Avg. length of citations
num_words	Count of words
subjectivity	Sum of word subjectivities (see mpqa dict)
sentiment	Sum of word subjectivities (see mpqa dict) / count of words
adjectives	Count of adjectives
adj_ratio	Count of adjectives / count of words
minmax_length	Minmax-scaled count of characters in headline
minmax_author	Minmax-scaled count of authors
normalized_length	Count of characters in headline, normalized over all articles
normalized_author_count	Count of authors, normalized over all articles
complexity	Count of finite verbs / count of tokens
present	Normative frequency of present tense verbs
past	Normative frequency of past tense verbs
interjections	Count of interjections

Our team next compared the performance of SVM, Gaussian naive Bayes, decision tree, extra trees, and random forest classifier models (Table 6), quantifying performance through accuracy, recall, specificity, and f1 score. For the SVM and naive Bayes models we also plotted ROC curves to calculate AUC, and we plotted the effect of decision thresholds on each performance metric to find thresholds that maximized recall while preserving accuracy, helping us improve SVM performance (Figure 3).

Table 6. Accuracy, recall, specificity, and f1 scores of different classification models for news/opinion classification. AUC score is included for non discrete classifiers.

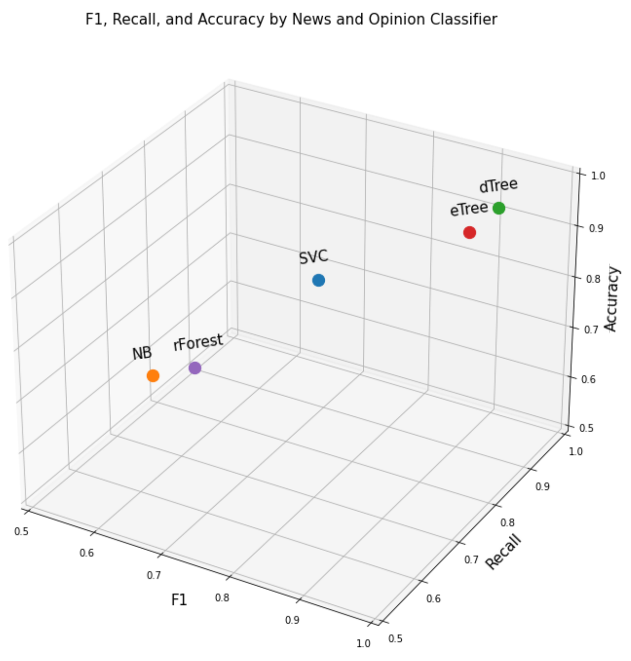
Classifier Model	**Accuracy**	**Recall**	**Specificity**	**F1**	**AUC**
SVM (Threshold=0.14)	0.814	0.822	0.813	0.736	0.881
Naive Bayes (Threshold=0.06)	0.803	0.523	0.858	0.672	0.817
Decision Tree	0.970	0.914	0.981	0.945	N/A
Extra Trees	0.970	0.839	0.995	0.942	N/A
Random Forest	0.900	0.460	0.986	0.772	N/A

Figure 3. Classifier performance metrics for our SVM model, by proposed decision threshold.



Our results showed tree models have the highest article status prediction accuracy (>=0.9) while our SVM, extra trees, and decision tree classifier best minimized false negatives (recall >= 0.8) Notably, a decision tree has both the highest accuracy and recall of the models we evaluated, showing it is indeed viable to apply ML to predict article status based on text and related metadata (Figure 4).

Figure 3. Visualization of classifier performance by accuracy, recall, and f1



We also investigated the most significant features of each model (Figure 4). For the tree models, these are verbs of speech, sentiment, and causal connectives. Verbs of speech proved most significant for the naive Bayes model as well (Figure 5). For the SVM, however, headline length features had the greatest weights, along with the normative frequency of past tense verbs.

Figure 4. Normalized feature importance for each classifier, compared in one stacked bar plot

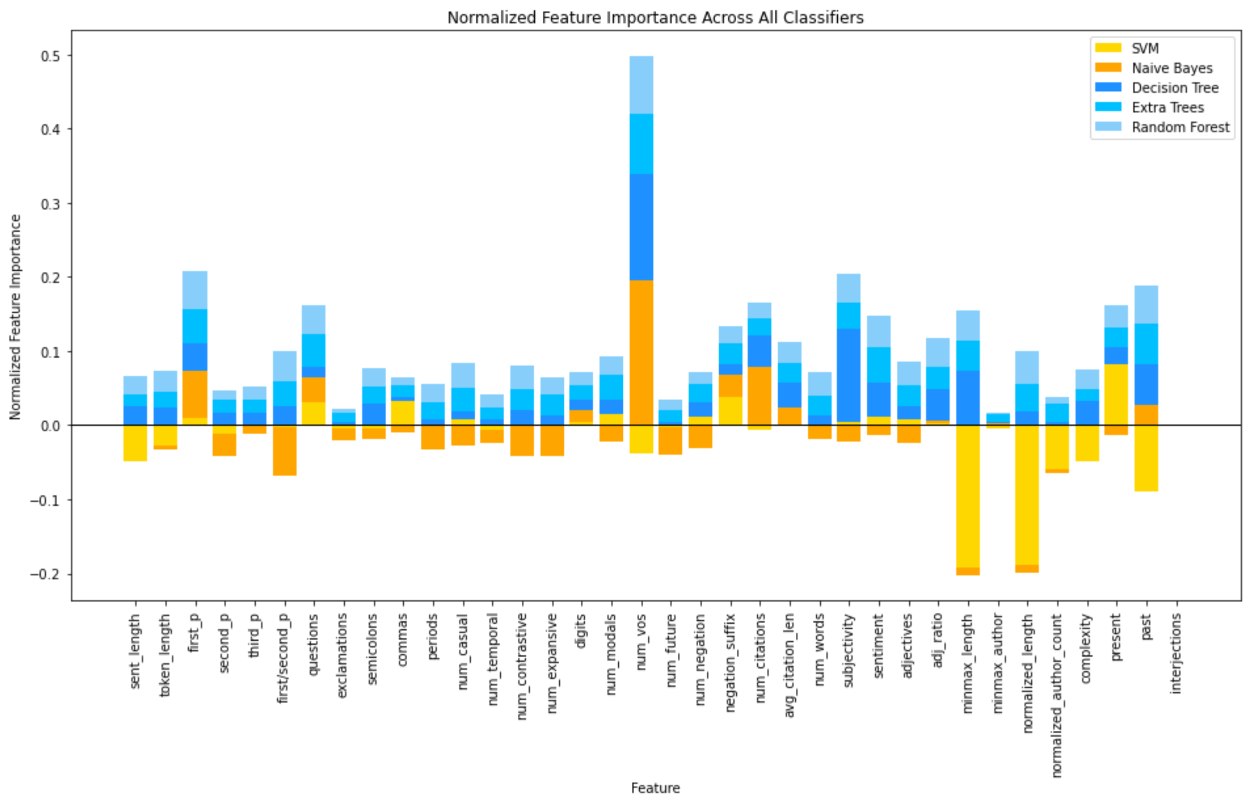
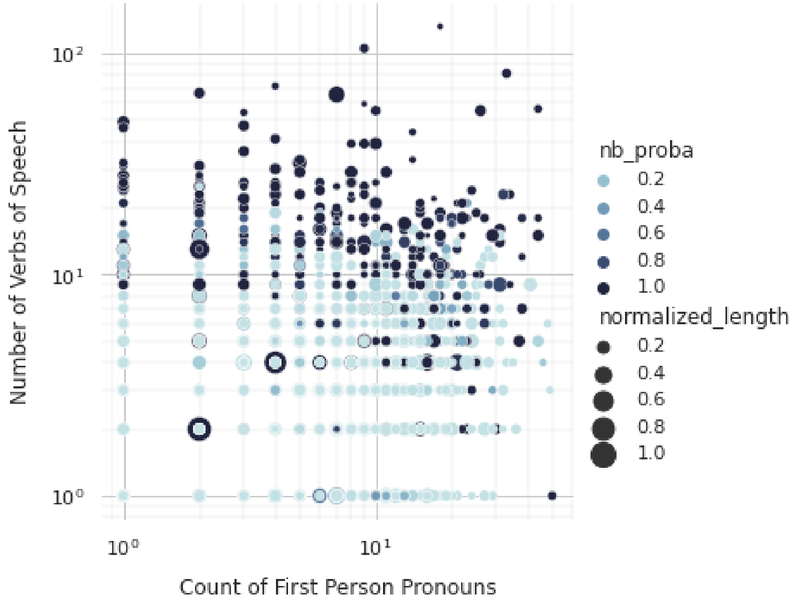


Figure 5. Visualization of how naive Bayes predicts the probability of an article being opinion, based on num_vos, first_p, and normalized_length, which had feature importance across all models.


10^2
10^1
10^0
Number of Verbs of Speech
10^0
10^1
Count of First Person Pronouns
nb_proba
normalized_length