Spencer Christensen

COSC 4931 - 12/08/24

Analyzing Political Donations With Classification Models

Abstract

The purpose of this project is to determine the feasibility of using machine learning models to predict political donations, and therefore affiliation. I have explored this using various pre-processing methods and two distinct models. The models, k-Nearest Neighbor and Random Forest, performed well on train and test data, despite that data's shortcomings. Using models for these predictions was certainly effective, however higher quality data and application specific tuning should be explored before any implementations are used.

Introduction

Political allegiance is often assumed based on stereotypes or generalizations about where an individual lives or what they do for a living. In this project I want to explore the feasibility of using machine learning models to accurately classify political beliefs based on these simple attributes, as well as a measure of political activity.

An effective model capable of making these predictions would be incredibly beneficial to campaigning and advertising fields. Where the predictions could be used to most efficiently solicit votes, donations, signatures, or product purchases. Success these areas could lead to drastic improvements in profitability or fundraising, while reducing the cost of such gains.

Due to the variety and availability of political data, I hypothesize that classification models based on it will be very effective at interpreting and predicting upon new data.

Methods

Data

I pulled the data used in this paper from the Federal Elections Commission's publicly available individual contribution database. This data was filtered to only include contributions to presidential campaigns or their supporting organizations. I also took extra care to only use data from after President Joe Biden's withdrawal, as this especially unusual event may have affected the demographic make-up of donors. The date ranges for contributions in the dataset range from July 21st 2024 to October 15th 2024. Before pre-processing this dataset comprised of nearly five million contributions.

Preprocessing

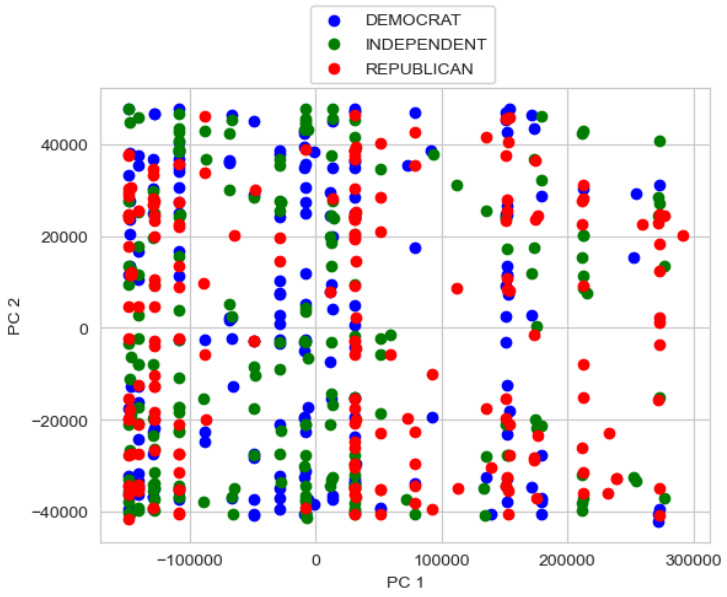
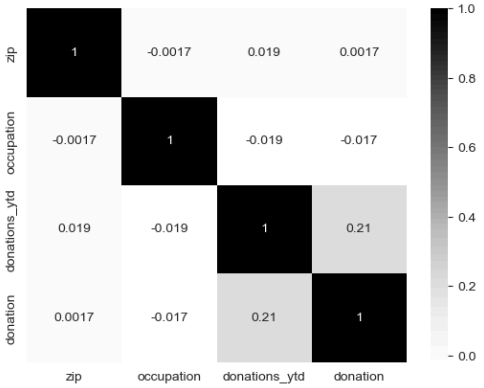
The dataset came with 74 features, almost all of these were for internal FEC tracking or records and not useful for the scope of this project. After filtering out the aforementioned administrative features, the dataset contained nine potentially useful features. Out of these features I selected four for use as input features and one as a target.

The target feature used in this project was 'committee_name' which contained 18 unique donation recipients. I mapped these to the three main political designations of democrat, republican, and independent. The four input features I selected were zip code, occupation, donation amount, and donation total year to date. I standardized both donation amount and donation total year to date using scikit-learn's StandardScalar tool. American zip codes are five digits but also have an optional 'plus four' that was present in the data, to prevent any confusion I truncated all zip codes to five digits. I embedded all occupations using the Bureau of Labor Statistics' Standard Occupational Classification or SOC system from 2018. This system classifies job titles using into eight hundred and sixty seven groups which are denoted by integers, where similar job titles link to integers that are similar in value. To embed the dataset I used a python module called sockit which uses natural language processing to map job titles to their most likely SOC classifications. When sockit returned multiple possible codes, I chose the option with the highest probability.

Before pre-processing the dataset contained nearly 5 million contributions and 74 features. This data had a class split of 77% Democrat, 22% Republican, and 1% independent. After removing the irrelevant features, and after every pre-processing step, I removed contributions with empty or invalid content. After pre-processing the dataset contained one and a half million contributions with 5 features, and a class distribution of 89% Democrat, 9% republican, and 2% independent.

To combat this heavy class imbalance I balanced the training data, which was 80% of the processed dataset. For this balancing I resampled the group of democrats down to one hundred and fifteen thousand contributions, which is how many republicans were present. I also used bootstrapping to resample the independent group up to one hundred and fifteen thousand. This created a training set of three hundred and fifty thousand data points equally distributed among each of the three classes.

Data Visualizations

PCA Plot	Feature Heatmap
 $PC_1 \approx 1 * occupation,$ $PC_2 \approx -1 * zipcode$	

Model Selection

Due to the low linearity and correlation of the features, I decided to avoid linear models such as Logistic Regression and Naive Bayes. Instead I opted to use a Random forest and a k-Nearest Neighbor model. These models are more suited to handle class based data and the complexities that come with it. After selecting my models I experimented with their arguments, looking to optimize f1-score. I chose to optimize f1-score because both recall and precision have importance in different use cases, and I wanted to create a generally applicable model. I found 2 neighbors and Euclidean distance was optimal for kNN, while 100 estimators was optimal for random forest. I also found that any adjustments to other parameters decreased the model's sensitivity to independent contributors, in both cases.

Metrics

Random Forest

RF on Test Data

Classification Report:

	Precision	Recall	f1-Score	Support
DEM	0.99	0.96	0.97	285591
IND	0.34	0.66	0.44	3620
REP	0.78	0.95	0.86	28711
Macro Avg	0.7	0.86	0.76	317922
Weighted Avg	0.97	0.95	0.96	317922

Accuracy: 0.95 Balanced Accuracy: 0.86

Confusion Matrix:

True / Predicted	DEM	IND	REP
DEM	273666	4593	7332
IND	1055	2381	184
REP	1196	116	27399

RF on Train Data

Classification Report:

	Precision	Recall	f1-Score	Support
DEM	1.0	1.0	1.0	114572
IND	1.0	1.0	1.0	114572
REP	1.0	1.0	1.0	114572
Macro Avg	1.0	1.0	1.0	343716
Weighted Avg	1.0	1.0	1.0	343716

Accuracy: 1.0 Balanced Accuracy: 1.0

Confusion Matrix:

True / Predicted	DEM	IND	REP
DEM	114563	9	0
IND	0	114572	0
REP	13	0	`114559`

k-Nearest Neighbor

kNN on Test Data

Classification Report:

	Precision	Recall	f1-Score	Support
DEM	0.97	0.87	0.92	285591
IND	0.17	0.68	0.27	3620
REP	0.45	0.77	0.57	28711
Macro Avg	0.53	0.77	0.59	317922
Weighted Avg	0.92	0.86	0.88	317922

Accuracy: 0.86 Balanced Accuracy: 0.77

Confusion Matrix:

True / Predicted	DEM	IND	REP
DEM	247286	11249	27056
IND	850	2477	293
REP	5796	810	22105

kNN on Train Data

Classification Report:

	Precision	Recall	f1-Score	Support
DEM	0.91	1.0	0.95	114572
IND	0.99	1.0	0.99	114572
REP	1.0	0.89	0.94	114572
Macro Avg	0.96	0.96	0.96	343716
Weighted Avg	0.96	0.96	0.96	343716

Accuracy: 0.96 Balanced Accuracy: 0.96

Confusion Matrix:

True / Predicted	DEM	IND	REP
DEM	114571	1	0
IND	57	114515	0
REP	11560	1587	101425

Comparison

On metrics Random Forest absolutely outclasses the k-Nearest Neighbor model, with a consistent lead on all weighted metrics. Random Forest still outperforms on the averaged metrics, however both models show a significant decrease in capability. This is most likely due to quality and sample size issues as the democrat class which had the best metrics across the board also had the vast majority of data points.

I also ran metrics on the same models, but they were only trained on the zip code and occupation features. This setup allows for much more flexible prediction as donation data would certainly be the hardest to acquire in an applied setting. Notably k-Nearest Neighbor lost almost no performance, most likely because it relied much less on the donation features. Random Forest however had nearly identical metrics to k-Nearest Neighbor. It seems that interpreting the donation data is what gives random forest its edge.

Another notable comparison is that of time and computation resources. Random Forest took seventy nine seconds to fit and predict, while k-Nearest Neighbor took only forty. This is not a significant difference at this sample size, however it is important to note for scaling.

Conclusion

Overall both models tested in this project performed very well. The most significant drawback was both models' underperformance on non-democrat classes. Due to the greater losses of these classes in pre-processing, and the smaller amount of them in the original dataset I suspect that data quality is the reason for this performance loss. This could be corrected by gathering more data from unique data from those classes, perhaps from previous elections, to give the models more variance to work with. After this project I would conclude that effective prediction of political affiliations is certainly achievable with the machine learning models we have available.

If this concept were to be extended to practical applications I would recommend a few changes. I would consider finding or collecting more feature rich data, demographic information such as age, race, gender, and familial status are likely to give a model better insight, but also allow for better generalizations of groups that campaigns may want to target. Additionally, investing in higher quality data than that which the FEC provides would certainly reduce error. I would also recommend tailoring the model of choice to the use case, for example a binary classification would likely provide better results for a particular party's campaign than multiclass would.