Machine learning model to predict the US 2024 election

by Joshua Au-Yeung

January 6, 2025

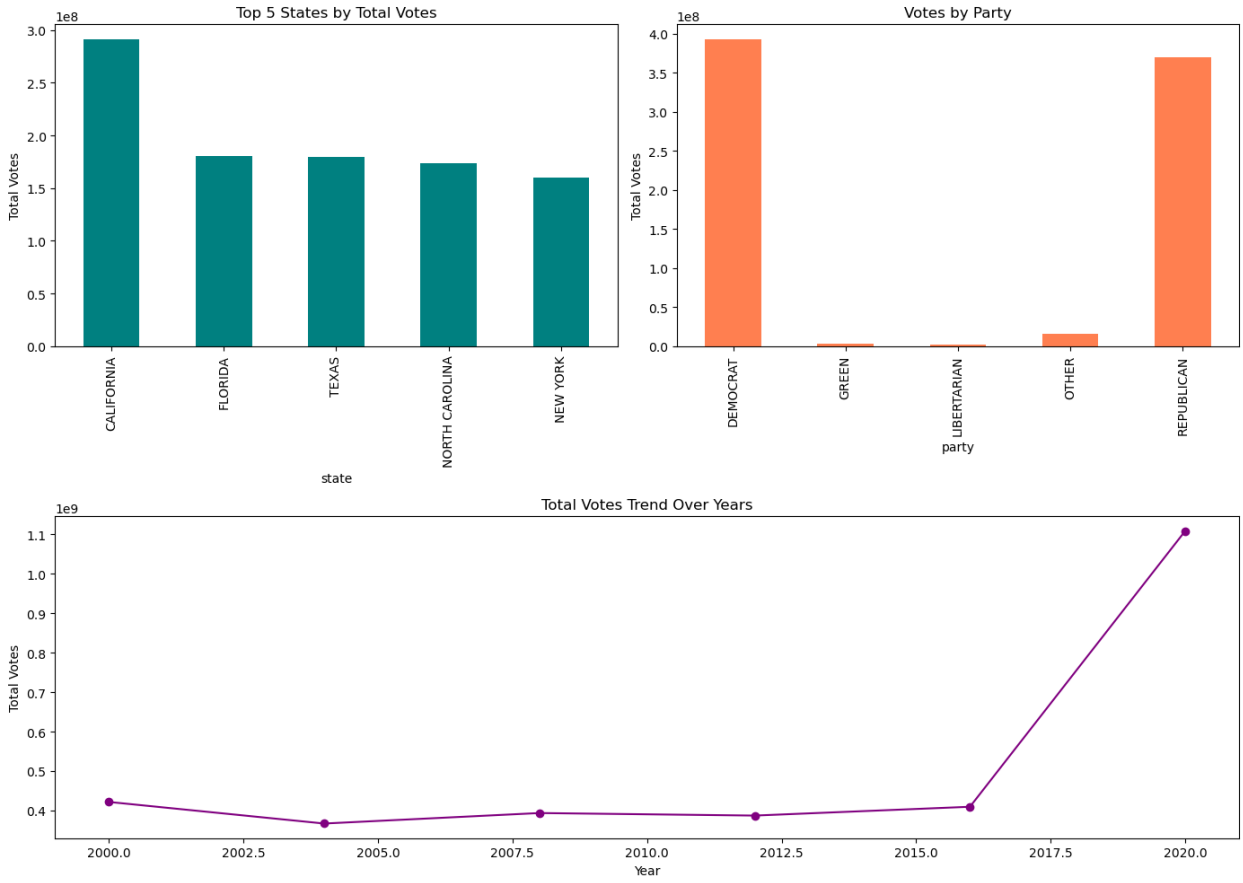
Project github code:

https://github.com/JackSparks/US-Election-Prediction-Model

This project predicts which political party will win in each U.S. county during the 2024 election. It uses three datasets: historical voting patterns from 2000-2020, 2024 betting odds for candidates, and 2024 favourability polls.

The three datasets provided offer a diverse set of information for predicting county-level outcomes in the 2024 U.S. presidential election. Each dataset contributes uniquely to the analysis by capturing different dimensions of voter behavior and election dynamics: historical trends, real-time sentiment, and voter favourability.

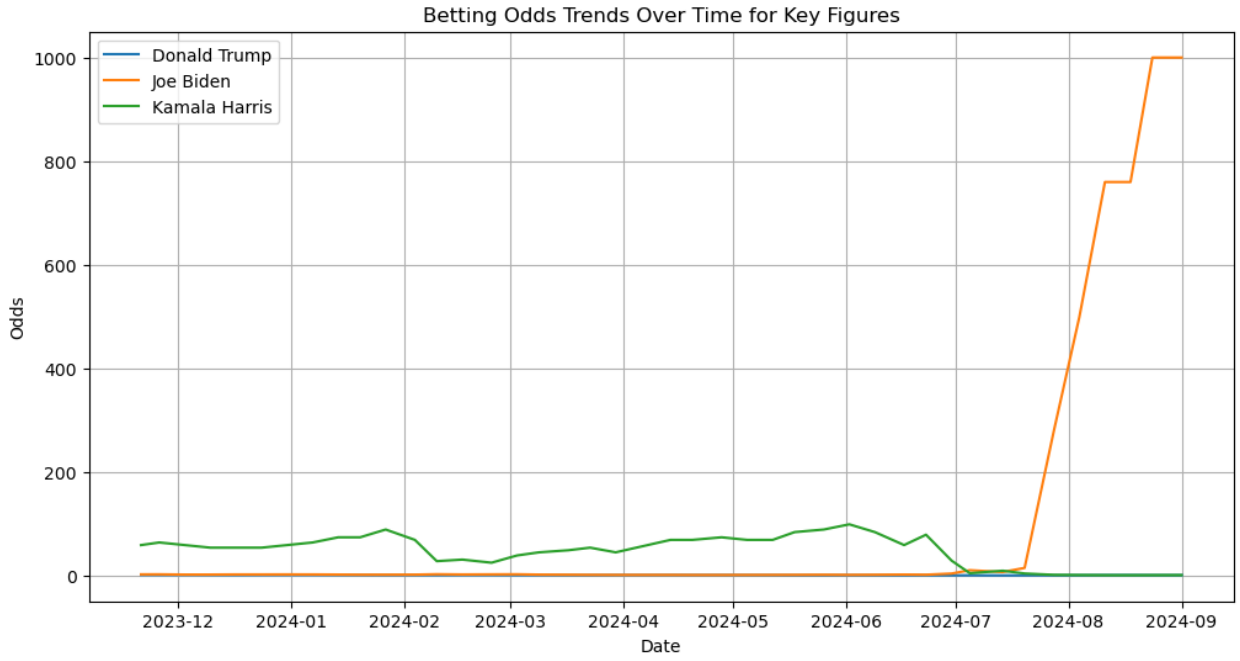
The first dataset, the county-level historical votes data, provides a detailed record of past election results across U.S. counties. This dataset includes key variables such as the year, state, county name, county FIPS code, candidate names, party affiliations, votes per candidate, and total votes cast in each county. The dataset allows for a detailed analysis of long-term voting trends and patterns at the county level. For example, identifying shifts in party dominance or understanding historical turnout rates can inform predictions about how counties may vote in 2024. Additionally, the inclusion of FIPS codes makes it possible to integrate this data with other geographic or demographic datasets. The following are some visualizations derived from the data.

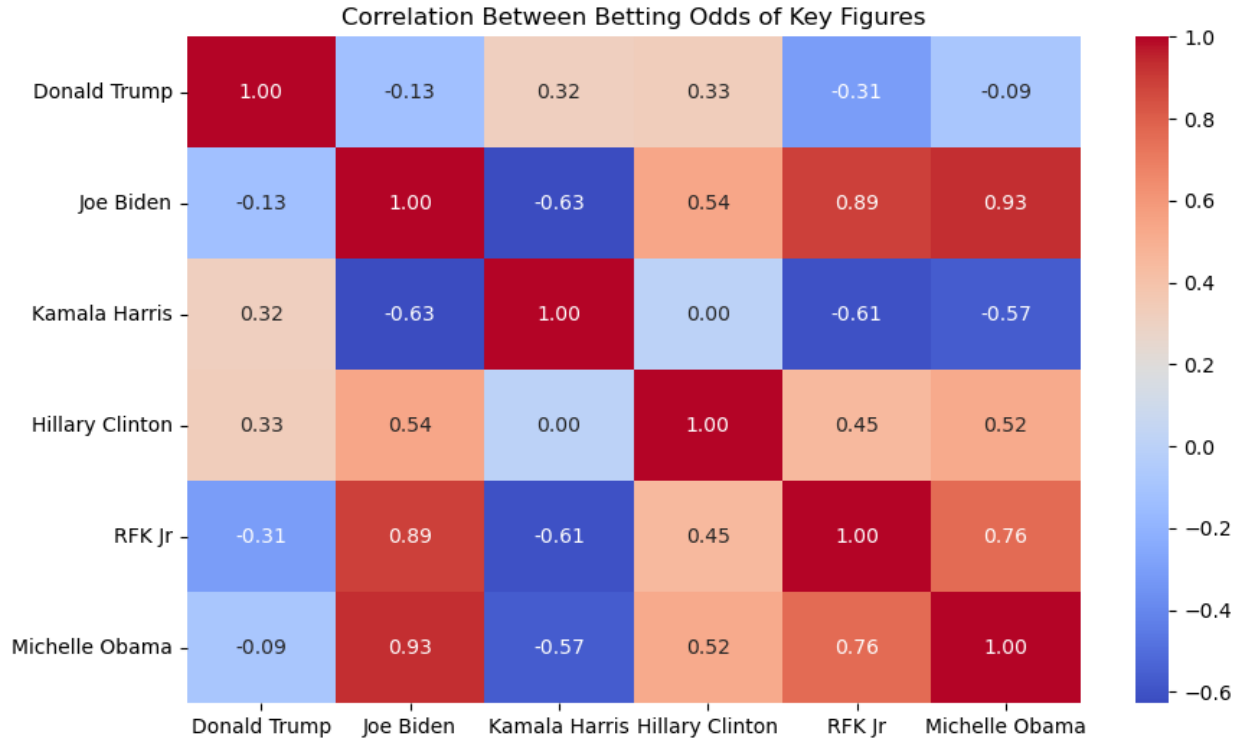


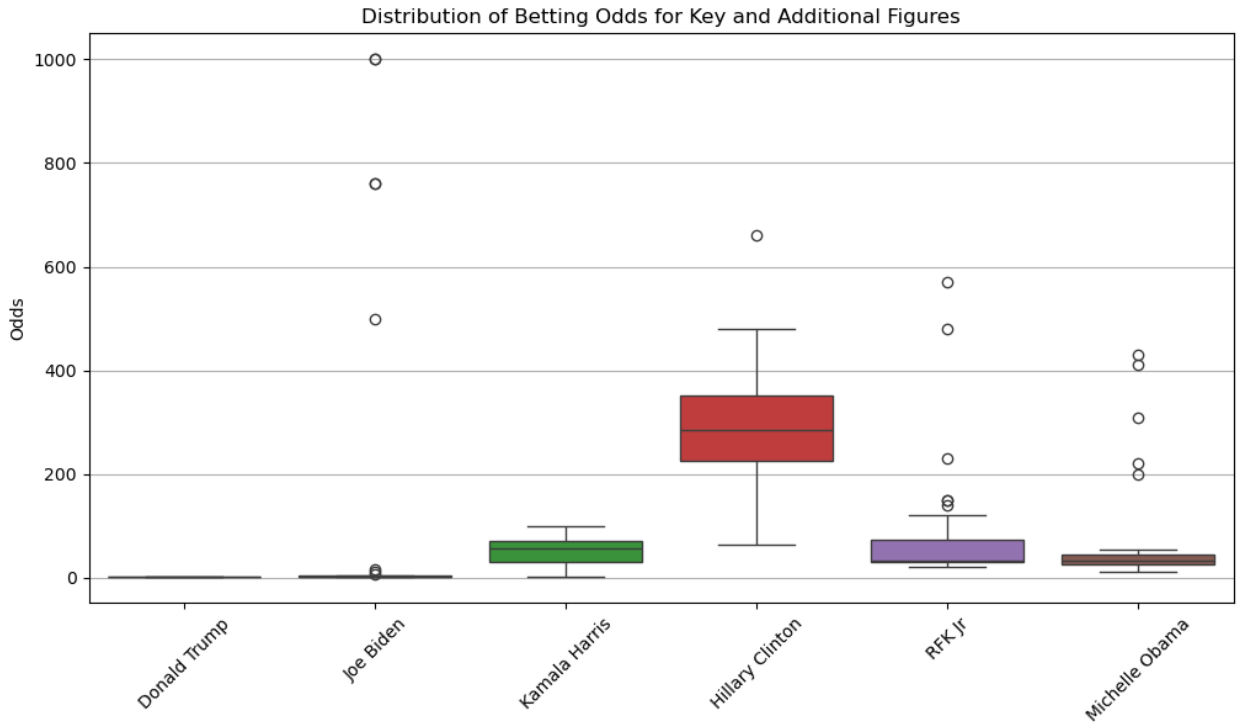
The dataset spans from 2000 to 2020. The average number of candidate votes per county is approximately 10,789, with a wide range (from 0 to over 3 million). The average total votes per county is about 42,544, with some counties recording no votes and others exceeding 4 million.

Democratic candidates received the highest cumulative votes (370 million). Third-party and independent candidates (e.g., Green, Libertarian) collectively account for a much smaller proportion of the votes.

The second dataset, which captures betting odds for candidates over time, reflects market sentiment and public expectations. This time-series data, sourced from a British betting bookmaker, provides insights into how candidates' chances were perceived leading up to the election. By covering a broad range of candidates and updating odds over time, this dataset allows us to monitor momentum shifts, public confidence, and the perceived viability of candidates. The following are some visualizations derived from the data.


Betting Odds Trends Over Time for Key Figures
Donald Trump
Joe Biden
Kamala Harris
Odds
1000
800
600
400
200
0
2023-12
2024-01
2024-02
2024-03
2024-04
2024-05
2024-06
2024-07
2024-08
2024-09
Date

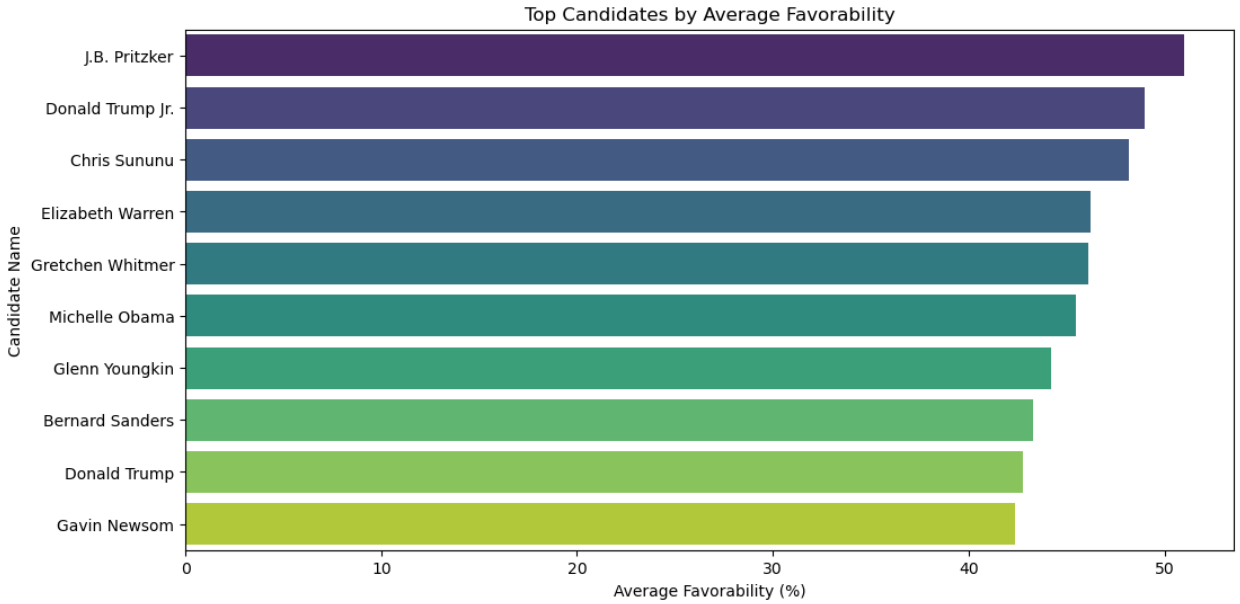

Correlation Between Betting Odds of Key Figures
Donald Trump
Joe Biden
Kamala Harris
Hillary Clinton
RFK Jr
Michelle Obama
1.00 -0.13 0.32 0.33 -0.31 -0.09
-0.13 1.00 -0.63 0.54 0.89 0.93
0.32 -0.63 1.00 0.00 -0.61 -0.57
0.33 0.54 0.00 1.00 0.45 0.52
-0.31 0.89 -0.61 0.45 1.00 0.76
-0.09 0.93 -0.57 0.52 0.76 1.00
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

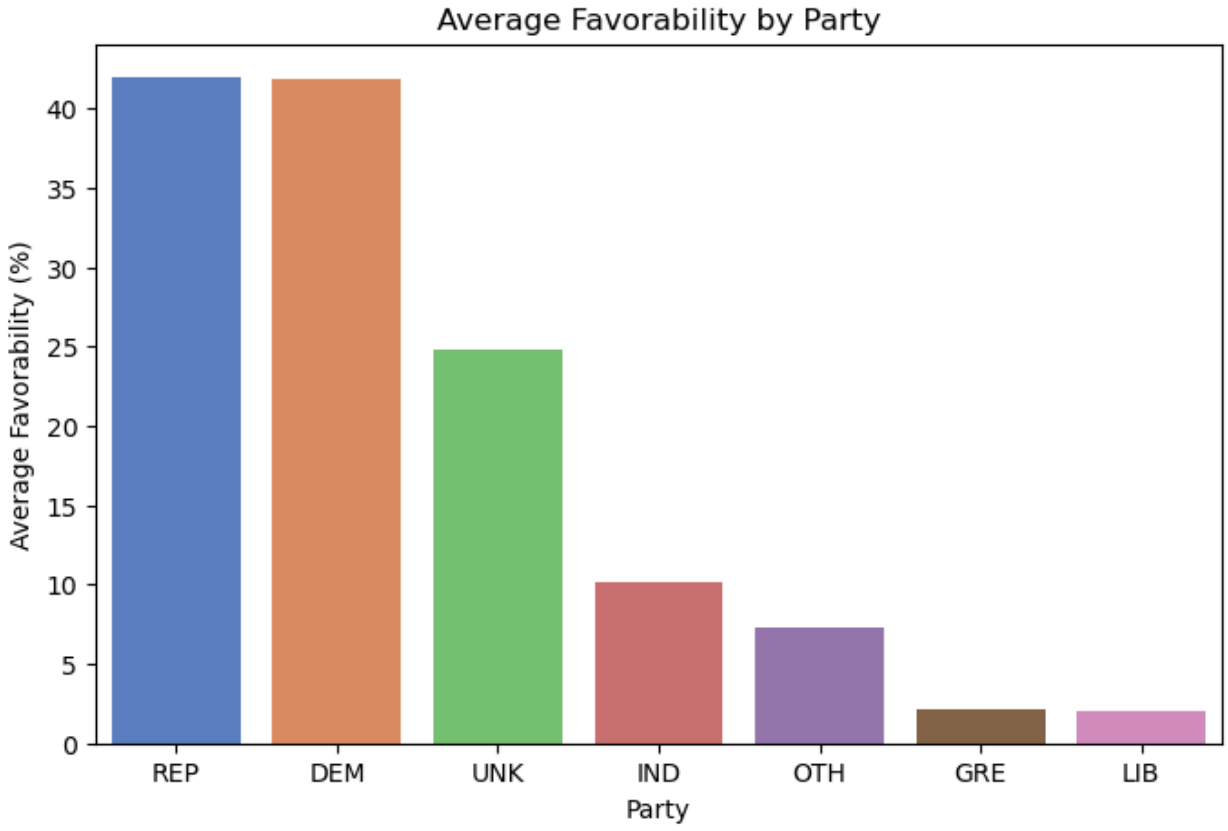


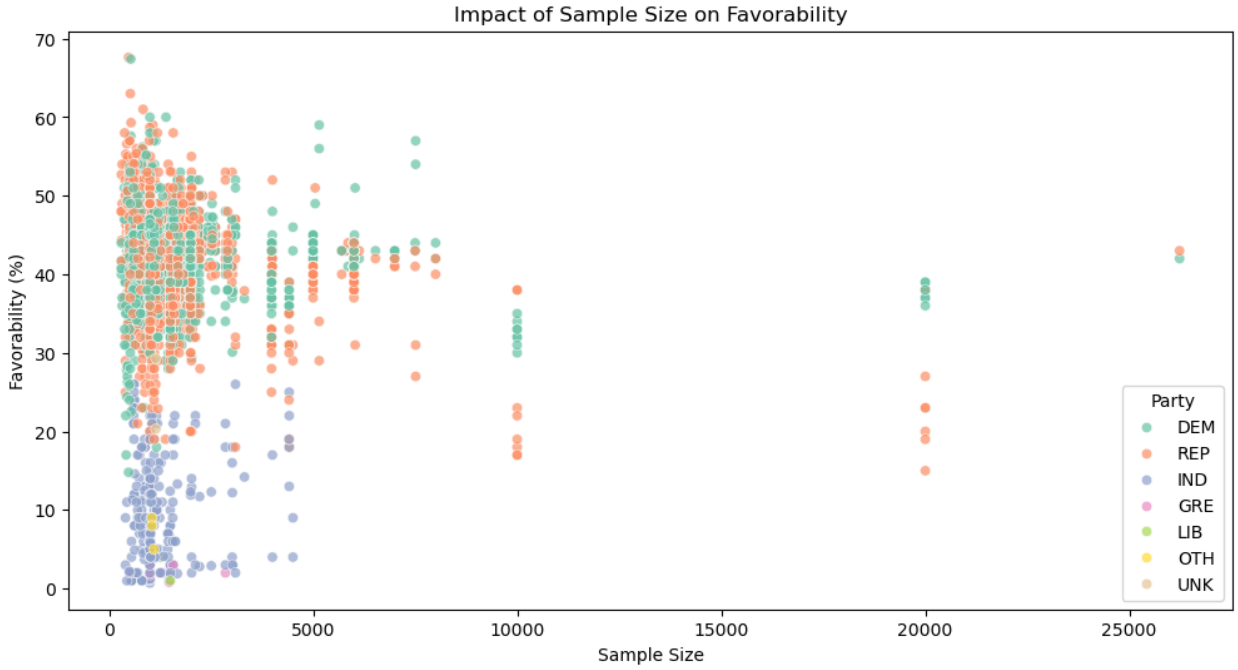
Correlation Analysis: Donald Trump: Weakly correlated with Joe Biden (-0.13) and RFK Jr (-0.31). Moderately positively correlated with Hillary Clinton (0.33). Joe Biden: Strong positive correlation with RFK Jr (0.89) and Michelle Obama (0.93). Strong negative correlation with Kamala Harris (-0.63), potentially due to competing odds as key Democratic figures. Kamala Harris: Weakly correlated with Hillary Clinton (0.002), indicating independent betting sentiments.

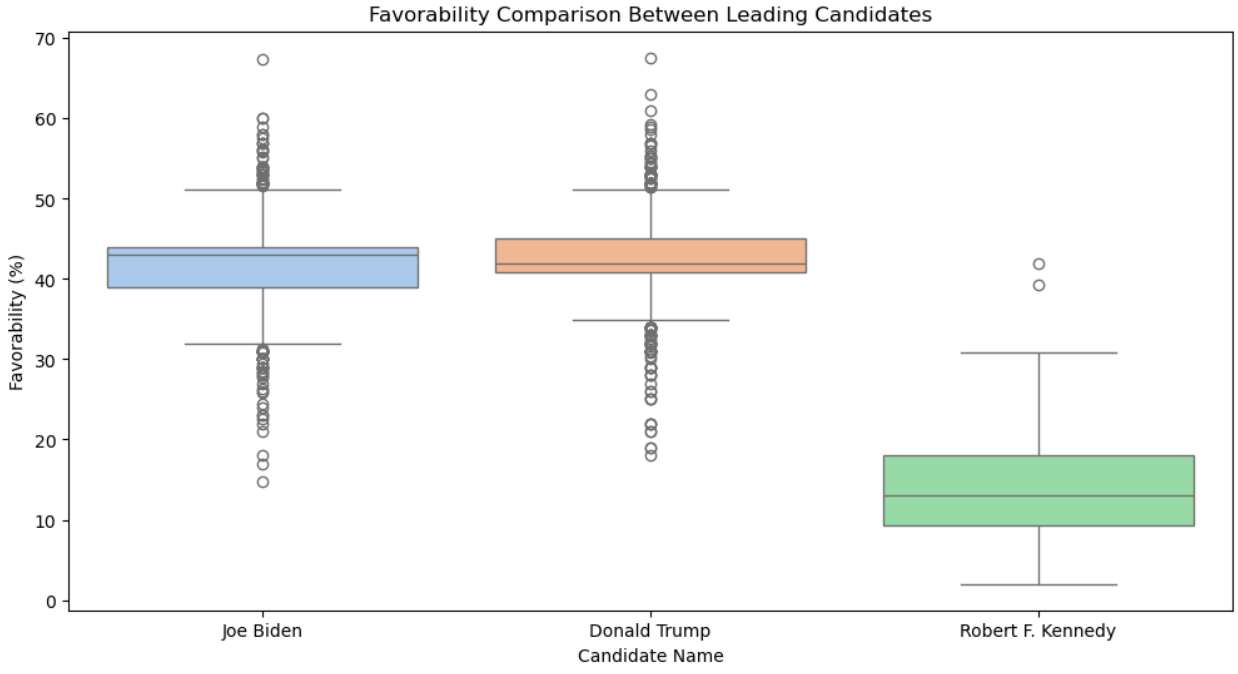
Boxplot Analysis: Donald Trump: Lowest variability in odds, showing consistent frontrunner status. Kamala Harris: High variability, reflecting uncertainty or mixed sentiment. Hillary Clinton: Wide range of odds, reinforcing speculative nature. RFK Jr and Michelle Obama: Moderate variability, showing consistent but dynamic betting interest.

The third dataset consists of favourability poll data sourced from the FiveThirtyEight website. This dataset offers a direct measure of voter preferences and includes variables such as pollster grades, transparency scores, sample sizes, party affiliations, candidate IDs, candidate names, and favourability percentages. The inclusion of pollster reliability scores (e.g., FiveThirtyEight's FTE grades) and sample sizes ensures that the quality of the polls can be assessed, enabling the identification of more trustworthy sources. The following are visualizations derived from the data.


Top Candidates by Average Favorability
J.B. Pritzker
Donald Trump Jr.
Chris Sununu
Elizabeth Warren
Gretchen Whitmer
Michelle Obama
Glenn Youngkin
Bernard Sanders
Donald Trump
Gavin Newsom
Candidate Name
0
10
20
30
40
50
Average Favorability (%)


Average Favorability by Party
Average Favorability (%)
0
5
10
15
20
25
30
35
40
REP
DEM
UNK
IND
OTH
GRE
LIB
Party





The average favourability percentage across all candidates is 40.18%, with a range from 0.7% to 67.6%. Joe Biden emerges as the most frequently polled candidate. Favourability percentages are generally skewed toward higher values, with the majority falling between 35% and 45%.

In terms of top-performing candidates, J.B. Pritzker leads with an average favourability of 51.00%, followed by Donald Trump Jr. (49.00%) and Chris Sununu (48.20%). Party-wise analysis indicates that Republicans (41.95%) and Democrats (41.81%) show comparable average favourability, while Independents lag significantly at 10.15%.

Comparing favourability among leading candidates, Donald Trump displays a wider variance, indicating more polarized public opinions compared to Joe Biden and Robert F. Kennedy.

In combination, these 3 datasets offer a balanced approach to election prediction. The historical county voting data establishes a solid baseline for understanding county-level trends, while the betting odds and favourability poll data add dynamic and real-time perspectives. The diverse origins of the data—ranging from objective historical records to subjective polling and market sentiment—reduce reliance on any single source and improve the robustness of predictions.

The datasets all had columns of irrelevant data that wouldn't assist the analysis. These columns were removed. Rows containing empty values or unusable values were also removed.

To check how well our model performs, I have a 4th dataset containing the election results of the US 2024 presidential election. This dataset shows the election results for counties throughout the United States. These results will be used to check how well my models performed.

To make predictions, I use a method called ensemble modeling. This means I test different machine learning models on each dataset, pick the best ones, and then combine their predictions into a final model.

Step 1: Model Selection and Evaluation

First, I tested several machine learning models on each dataset separately. The models include Logistic Regression, Random Forest, Support Vector Classifier (SVC), AdaBoost, Gradient Boosting, K-Nearest Neighbors (KNN), and Decision Tree. These models were chosen because they are reliable, fast, and work well for different types of data.

To check how good each model is, I split the data into training and testing sets. I trained the models on the training set and tested them on the testing set. I also used cross-validation, a method that helps us see how the models perform on different parts of the training data. I measured performance using metrics like ROC-AUC (which shows how well the model separates winners from losers), accuracy, precision, recall, and F1-score. ROC-AUC was the main metric because it is important for deciding between two classes, like winning and losing.

Step 2: Choosing the Best Model for Each Dataset

Next, I trained the models on their specific datasets ("counties," "odds," and "favourability"). Some models have settings, like the number of trees in Random Forest, which affect how they perform. These settings were kept simple and used default values to save time. I made sure the data was split the same way for all models to make the comparison fair. The best model for each dataset was the one with the highest average ROC-AUC score.

Step 3: Hyperparameter Tuning for Better Performance

After the initial model evaluation, I improved the models by tuning their hyperparameters. Hyperparameters are settings that control how the model learns from the data. For example, in

Random Forest, the number of trees or their depth can be adjusted to improve results. I used a method called Grid Search to try out different combinations of these settings and find the ones that work best.

Each model had a specific set of parameters to tune. For instance, Logistic Regression tested different regularization strengths, while Random Forest varied the number of trees and their depth. Grid Search was applied on the training data, and the best combination of parameters was selected based on cross-validation scores using the ROC-AUC metric. The tuned models were then re-evaluated on the testing data to confirm their improved performance. This step ensured that the models I used were optimized for accuracy and reliability.

Tuning the hyperparameters did not result in better (more accurate) results than the default model weightings in my tests.

Step 4: Creating Meta-Features

After finding the best model for each dataset, I used those models to make predictions. These predictions, called meta-features, are the probabilities of a party winning in each county. For example, the model might say there is a 70% chance a party will win a county. These probabilities from all three datasets were then combined to create new features for the final model.

Step 5: Combining Predictions with a Stacking Model

To bring everything together, I used a method called stacking. In stacking, the meta-features (probabilities from the best models) are used as inputs to another model, called the meta-classifier. I chose Logistic Regression for the meta-classifier because it is easy to understand and works well for combining predictions. This model learns from the combined meta-features and makes the final prediction about which party will win.

Step 6: Evaluating the Final Model

I tested the stacking model using metrics like accuracy and ROC-AUC. The stacking model performed better than any single model on its own. This shows that combining information from the three datasets creates a stronger and more accurate model.

Summary of Results from Model Tuning and Stacking

Individual Models Performance

CV ROC-AUC is a measure to show how well the model predicted on the training data. Test ROC-AUC is the same but to show how it performed on the test data.

The performance of individual models varied, with most showing limited predictive power. GradientBoosting performed the best among the tested models, with a test ROC-AUC of 0.5341 and a cross-validation (CV) ROC-AUC mean of 0.5741. However, its results were only slightly better than random guessing, which has an ROC-AUC of 0.5. K-Nearest Neighbors (KNN) and DecisionTree also had weak performances, with test ROC-AUC scores close to 0.51 and CV ROC-

AUC means around 0.54. These scores suggest that none of the models performed strongly on their own.

Best Models for Each Feature Set

When evaluating models on specific feature sets, GradientBoosting stood out for both counties and favourability features. For counties, it achieved a CV ROC-AUC mean of 0.6692 and a test ROC-AUC of 0.6460, demonstrating moderate predictive ability. For favourability features, its performance was weaker, with a CV ROC-AUC mean of 0.5741 and a test ROC-AUC of 0.5341. LogisticRegression was the best model for odds features, but its performance was no better than random guessing, with both the CV and test ROC-AUC scores at 0.5.

Stacking Meta-Classifier Performance

The stacking meta-classifier is a model that combines predictions from other models to make a final decision. It achieved an accuracy of 84%, meaning it made correct predictions 84% of the time. Its F1-score was 0.77, which measures how well the model balances precision (being right when it predicts "positive") and recall (not missing actual positives). A perfect F1-score is 1, and a score of 0.77 shows it was fairly effective but not exceptional. However, the model was heavily biased toward predicting positives (class 1) and failed to correctly identify any negatives (class 0). This means it completely ignored one side of the problem.

In other words, the model is reasonably reliable at predicting when a county would vote Gop or is likely to swing its vote from democratic to republican. But it is not helpful at predicting when a state would change from GOP to democratic. Why? Because the model is likely heavily predicting one outcome and ignoring the other.

The ROC-AUC score, which measures how well the model distinguishes between positive and negative cases, was 0.6451. While 0.6451 is better than random guessing (0.5), it shows that the model's ability to separate the two classes was only moderate. The main reason for these shortcomings was class imbalance, where there were far more positive cases than negative ones. This imbalance caused the model to focus too much on predicting positives and overlook negatives entirely. In summary, while the model performed well for positive predictions, its failure to handle negative cases made its overall performance less reliable.

Key Takeaways

1. The counties dataset provided the most predictive power, followed by the favourability dataset. The odds dataset did not contribute meaningful predictions.
2. GradientBoosting was the most effective individual model, but its results were modest, indicating room for improvement.
3. The stacking classifier improved accuracy and F1-score compared to individual models but faced challenges with class imbalance, which limited its ability to predict negative cases.

Conclusion

This project predicts the 2024 U.S. election results at the county level. By testing different models, tuning their hyperparameters, picking the best ones, and combining their predictions, I created a final model that achieved an accuracy of 84%. However, it was found that the model is only successful at predicting when the GOP party would win a county, but is not successful at predicting when the democrats would win a county.

This model result is not surprising when you consider the datasets used. The favourability polls and the odd betting data only reflect the national level sentiment as they don't break down into regional demographics. They can only really predict whether the GOP or democrat would win the whole election and they both show that the GOP was favored to win. So, they give a slight positive influence towards predicting that the GOP to be more likely to win any given county.

For next steps, what would be needed is to have a favourability polls and betting odds dataset for each election from 2000-2020. Then we could see how accurate the influence of the favourability polls and odds over time rather than relying on a single election. However, for this back testing, we'd need additional datasets.

Datasets used in analysis:

MIT historical county election results dataset

https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/VOQCHQ

538 website dataset with 2024 election poll favourability data:

https://github.com/fivethirtyeight/data/tree/master/polls

Dataset with betting odds to back individual candidates for 2024 US President

https://www.kaggle.com/datasets/jdaustralia/2024-us-presidential-election-odds

2024 election results dataset used to compare model predictions against:

https://github.com/tonmcg/US_County_Level_Election_Results_08-24