University of New Haven
TAGLIATELA COLLEGE OF ENGINEERING
Electrical & Computer Engineering and Computer Science

Computer Engineering

Election Forecasting

CONTENTS

Executive Summary

This project aims to predict election outcomes using machine learning. It involves data collection, cleaning, preprocessing, and feature engineering. Various algorithms such as logistic regression and decision trees are used for predictions.

The project is useful for analyzing voter behavior, identifying swing states, and understanding election factors. Accuracy depends on data quality and model assumptions. This tool is valuable for political parties, election commissions, and media organizations.

Team Members:
Anil Kumar Muttineni
Shashidhar Reddy Katikam
Varshitha Jonnalagadda

Questions?
Contact: amutt2@unh.newhaven.edu

Problem Statement

This project aims to develop a predictive model for forecasting the results of political elections. The model will be based on analysis of historical voting data, demographic information, and other relevant factors. Machine learning algorithms, statistical analysis, and data visualization tools will be employed to analyse large datasets. The successful completion of this project will have significant implications for political science, political parties, candidates, and media organizations.

Our approach for this project is as per the following steps:

- Identify and collect data with necessary attributes
- EDA & perform necessary data cleansing
- Segregate data into training set and testing set
- Select appropriate algorithm
- Executing best training algorithm with testing data
- Deployment

Information About Data Set

The Data Set represent various demographic and socioeconomic factors of individuals in certain counties, as well as election data for those counties. The columns include information such as year, county fips code, income, mortgage amount, average age, sex, marital status, race, citizenship status, language spoken, education attainment, employment status, and political party affiliation. Some columns have null values, such as state_po, county_name, democrat, green, liberitarian, other, republican, and winner, which represent the election data for certain counties.

Identify and collect data with necessary attributes

In [2]:
```
# import dataset
df = pd.read_csv('/kaggle/input/us-census-for-election-predictions-20002020/county_census_and_election_result.csv')
df.head()
```

Out[2]:

	year	county_fips	inctot	mortamt1	avrg_age	ftotinc	foodstmp_1_freq	foods
0	2000	1	24566.4	634.6	46.0	46912.7	93.6	6.4
1	2000	2	33842.9	1080.1	42.4	65021.9	95.3	4.7
2	2000	4	28331.7	814.7	45.3	52826.7	95.8	4.2
3	2000	5	22782.6	557.5	46.2	43941.3	92.5	7.5
4	2000	6	32245.0	1216.3	43.8	61455.3	95.7	4.3

5 rows × 45 columns

The data contains demographic, economic, and social information related to various counties in the year 2020. The data includes variables such as income, mortgage amount, age, sex, employment status, and political affiliation. Dataset contains 45 columns and 5 rows. The analysis of this data may be useful in identifying trends and relationships between different factors and political outcomes in the given counties.

Exploratory Data Analysis & Data Cleaning

As we consider each row as a data point to our ML model, we will drop the categorical state names and abbreviation. We will also drop the all the number of votes labels except for "winner" columns (0 for democrats and 1 for republican). We don't concern other parties as they do not have a history of winning anyway.

county_fips: as there is no correlation between fips and vote numbers. state_po: as there is no correlation between state appreviation and vote numbers. county_name: as there is no correlation between county names and vote numbers. democrat: as we do classification task. green: as we do classification task. liberitarian: as we do classification task. other: as we do classification task. republican: as we do classification task.

Machine Learning Models

machine learning models provide a powerful tool for analyzing and making sense of large datasets, which can lead to improved decision-making, increased efficiency, and ultimately, better outcomes.

We have used two machine learning models to have to improved decision-making.

1. Decision Tree With Gini Criterion
2. Decision Tree With Entropy Criterion

1. Decision Tree with Gini Criterion

A Decision Tree is a machine learning algorithm used for classification and regression tasks. It works by recursively partitioning the data into subsets based on the most significant attribute that provides the maximum amount of information gain. The Gini impurity is a measure of the probability of incorrectly classifying a randomly chosen element from the set, and the Decision Tree with Gini criterion chooses the attribute that results in the lowest Gini impurity value. This algorithm is popular due to its interpretability, simplicity, and ability to handle non-linearly separable data. However, it may suffer from overfitting and instability with noisy data.

```
clf1_pred_y_2020 = clf1.predict(X_df2)

accuracy = metrics.accuracy_score(y_df2, clf1_pred_y_2020)
f1 = metrics.f1_score(y_df2, clf1_pred_y_2020)
prec = metrics.precision_score(y_df2, clf1_pred_y_2020)
recall = metrics.recall_score(y_df2, clf1_pred_y_2020)
roc_auc = metrics.roc_auc_score(y_df2, clf1_pred_y_2020)

print("Testing MSE = %f" % metrics.mean_squared_error(y_df2, clf1_pred_y_2020))
print('Accuracy = %f' % (accuracy))
print('F1 Score = %f' % (f1))
print('Precision Score = %f' % (prec))
print('Recall Score = %f' % (recall))
print('ROC-AUC Score = %f' % (roc_auc))
```

```
Testing MSE = 1.138075
Accuracy = 0.715481
F1 Score = 0.696429
Precision Score = 0.735849
Recall Score = 0.661017
ROC-AUC Score = 0.714806
```

2. Decision Tree with Entropy Criterion

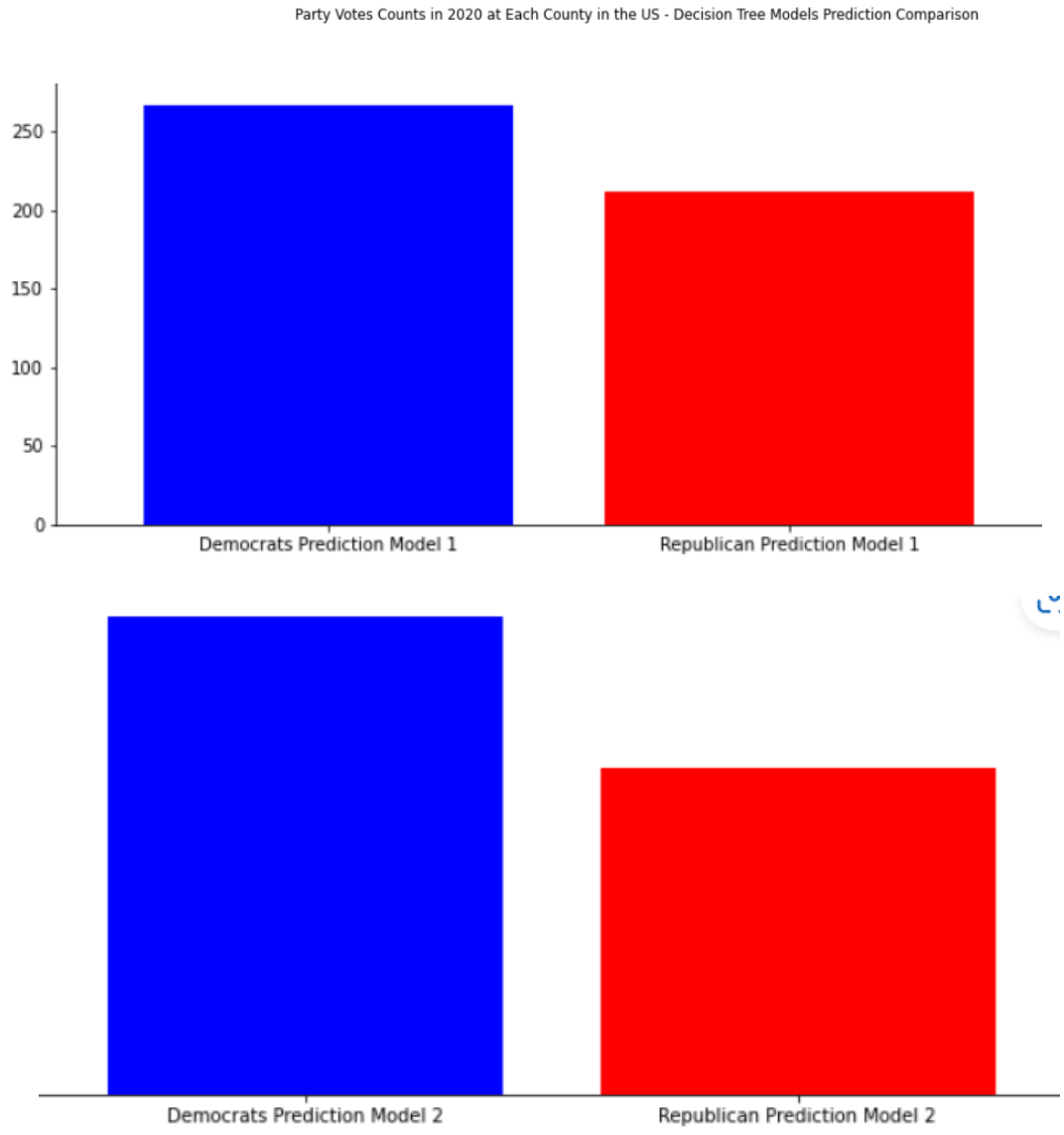
A Decision Tree with Entropy Criterion is a machine learning algorithm used for classification and regression problems. It builds a decision tree by recursively splitting the data based on the feature that maximizes the information gain, which is measured using the entropy criterion. Entropy is a measure of impurity or randomness in the data. The goal is to create a tree that accurately predicts the target variable by minimizing the entropy at each node. The resulting tree can be used for both classification and regression tasks.

```
clf2_pred_y_2020 = clf2.predict(X_df2)

accuracy = metrics.accuracy_score(y_df2, clf2_pred_y_2020)
f1 = metrics.f1_score(y_df2, clf2_pred_y_2020)
prec = metrics.precision_score(y_df2, clf2_pred_y_2020)
recall = metrics.recall_score(y_df2, clf2_pred_y_2020)
roc_auc = metrics.roc_auc_score(y_df2, clf2_pred_y_2020)

print("Testing MSE = %f" % metrics.mean_squared_error(y_df2, clf2_pred_y_2020))
print('Accuracy = %f' % (accuracy))
print('F1 Score = %f' % (f1))
print('Precision Score = %f' % (prec))
print('Recall Score = %f' % (recall))
print('ROC-AUC Score = %f' % (roc_auc))
```

```
Testing MSE = 1.004184
Accuracy = 0.748954
F1 Score = 0.720930
Precision Score = 0.798969
Recall Score = 0.656780
ROC-AUC Score = 0.747811
```

Plot representation of 2 Models

```
votes = [demo_count1, repu_count1, demo_count2, repu_count2]
names = ['Democrats Prediction Model 1', 'Republican Prediction Model 1',
         'Democrats Prediction Model 2', 'Republican Prediction Model 2']

plt.figure(figsize=(20, 5))
plt.bar(names, votes, color=['blue', 'red', 'blue', 'red'])
plt.title('Party Votes Counts in 2020 at Each County in the US - Decision Tree Models Prediction Comparison')
```



After plotting the results, it is evident that democrats are doing better in both the models.

Deployment

AWS EMR (Elastic MapReduce) is a cloud-based big data processing service that allows you to easily deploy and run big data frameworks and can also use it to deploy "IPython" Notebooks (IPYNB files) for interactive data analysis.

Here are the high-level steps used for deploying an IPYNB file on AWS EMR:

1. Launch an EMR cluster: select the appropriate configuration options and EC2 instance types based on workload and budget.

2. Install Jupyter Notebook on EMR: Once the EMR cluster is up and running, we need to install Jupyter Notebook on the cluster. This can be done using the bootstrap actions feature in EMR.

3. Upload IPYNB file to EMR: After Jupyter Notebook is installed, upload IPYNB file to the EMR cluster. We can do this using the EMR file system (EMRFS).

4. Launch Jupyter Notebook on EMR: Finally, we need to launch Jupyter Notebook on the EMR cluster and open our IPYNB file in the notebook. We can then access the Jupyter Notebook interface via your web browser, and open your IPYNB file for interactive data analysis.

Conclusion

- Decision tree Algorithm is used for both classification and regression
- It can be used to classify data into multiple classes based on their features
- In our project we need to classify data and need to predict, hence algorithm is best fit for our project
- Using Gini and Entropy methods we predicted accuracy and obtained the Election Result Prediction.
- By following this model after running for all categories we got the prediction as "Democrat's" are going to win the elections.

Source file

https://www.kaggle.com/minhbtnguyen/presidential-election-prediction-via-us-census/input

GitHub Link

https://github.com/MrKatikam/FP-Group8