Election Result Prediction using Random Forest

Naveen Suresh
Dept. of Computer Science
PES University
Bangalore, India

Akash Yadav
Dept. of Computer Science
PES University
Bangalore, India

Saiprakash L Shetty
Dept. of Computer Science
PES University
Bangalore, India

***Abstract* — This Project involves analysing and predicting the winner of an election using the 2019 Indian General Election Dataset. The method used to predict the winner is the Random Forest Classifier method.**

Index – election prediction Random Forest

I. INTRODUCTION

Elections are an integral part of any democracy. They are vital for the functioning of a democracy, especially given the weakness that a non- electoral society offers. But, with India and its population it is truly colossal. The total number of voters is bigger than the population of Europe. With over 900 Million Registered voters, the scale of the election is huge. Out of these registered voters, 15 million are aged between 18 and 19. There were nearly 1 million polling stations set up to make this massive exercise a success. Due to the dynamic nature of both globalisation and Indian Polity, the management of elections in India is continuously evolving, from separate ballot boxes for each candidate, to the marking system, to EVM's. With the change in the way elections are conducted, the way we analyse the elections are also changing in a dramatic way with data being the soul factor. This Project is about analysing the 2019 Indian General Elections data to infer certain important trends and patterns as well as to predict the winner of the election using the data.

Using the dataset, we intend to analyse important statistics of the 2019 election such as number of candidates, minimum age and maximum age of the candidate, average age of the candidates etc. We would also do visualizations that would produce the results for trends and patterns that play a major role in influencing the result of the election. Some of the important factors include reservations, assets of candidates, total number of voters by state, candidates with criminal cases against them, qualification of the candidates, vote share etc. We also intend to provide certain key visualizations such as Seats won and lost based on the age of a candidate. This would provide a clear picture of how the election shaped up.

The method used for predicting the winner in this project is Random Forest Classifier. Random forest consists of a large number of decision trees that operate as an ensemble. An ensemble of classifiers is a set of classifiers whose individual decisions are combined in some way to classify new examples. Each individual tree in the forest will result in a class prediction. The class that has the maximum number of votes will be our model's prediction. These are a large number of uncorrelated trees. The key for the success of the model that we will build will be the low correlation factor. In this project, we will be performing Data scaling before predicting the winner using the random forest classifier.

Elections are an important exercise in any part of the world, especially in democratic countries. Moreover, India being the largest democracy, fair and even elections are a must. We will drive into the 2019 Indian Election and predict the winner with maximum accuracy.

II. REVIEW ON PREVIOUS WORK

The results of an election can be predicted using various classifiers and models. A review of multiple research papers has given us a fair understanding as to why a Random forest model is considered to be the most accurate. The paper [1] presents a machine learning methodology (Random forest modelling) for identifying polling places at risk of election fraud and estimating the extent of potential electoral manipulation, using synthetic training data. They apply this methodology to mesa-level data from Argentina's 2015 national elections.

Multiple assumptions are made for the model, some of them are; In the generation of the synthetic data, an assumption has been made that 1/3 of at-risk mesas are affected by Ballot Box Stuffing(BBS) and Vote stealing(VS), respectively, and that these two forms of fraud are exclusive events. An assumption is also made that on an average, 3/4 of voter abstention in mesas at risk of Ballot Box Stuffing is counted as votes for the incumbent, and that 1/2 of votes cast for opposition parties in mesas at risk of Vote Stealing are counted as votes for the incumbent.

The method used here is; the labelled synthetic data is used to train a random forest model that can be used on actual election data to see whether there is evidence suggesting that a voting precinct is at risk of VS or BBS. The outcome variable of the model has three classes: clean, risk of VS, or risk of BBS.

The model does a highly accurate job predicting whether a synthetic mesa is clean or at risk of manipulation, correctly predicting 97% of the clean synthetic examples.

The limitation to the prediction is that the predictors lack a priori labelling of voter precincts for whether they may have been affected by manipulation. Other than this limitation, the model built is highly accurate and gives us a good understanding of the Random forest model. We do not encounter this problem because we have an accurate dataset for our prediction.

A winner prediction problem can be done using several methods like Regression, clustering, Forecasting and decision trees/random forest techniques. For our project, we intend to use the Random forest technique for the classification of our target 'winner' variable. The reason to use the above method is because Random forest is a very accurate classifier and is robust in making accurate decisions, the above statement is complemented by the exceptional accuracy predicted in the paper that was surveyed.[1]

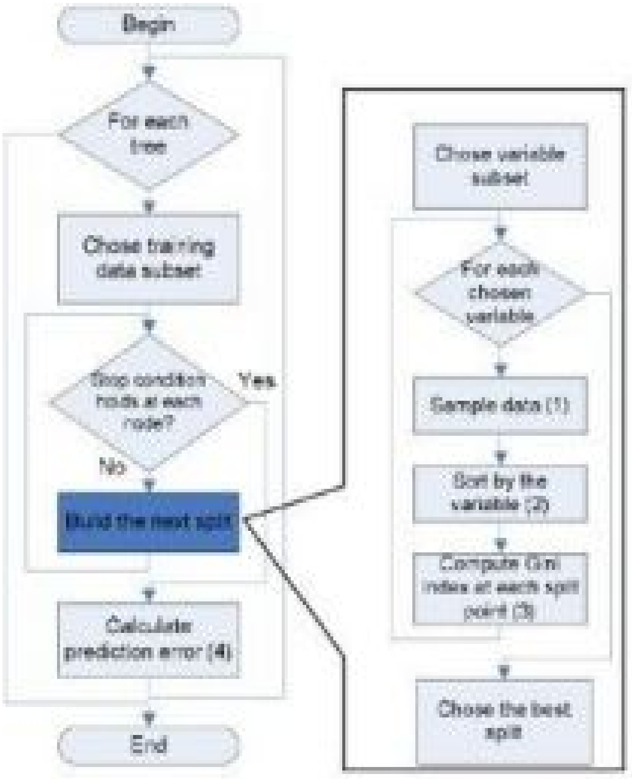
For our project, we intend to use the Random forest technique for the classification of our target 'winner' variable. The reason to use the above method is because Random forest is a very accurate classifier and is robust in making accurate decisions; the above statement is complemented by the exceptional accuracy predicted in the paper that was surveyed.

Here, a supervised machine learning ensemble approach (Random Forest) is used, there is a lack of prior labelling of voting precincts for whether they may have been affected by manipulation—and the form and extent of that electoral manipulation. Thus, they have developed a relatively naïve hierarchical model that is used to estimate what a "clean" election might look like across all of the voting precincts in the election that are studies, which forms our "clean" synthetic training data.

The only limitation to this project [1] is that more data could have been used to fine tune the model to a better extent and we intend to do away this limitation in our project as we use enough sufficient data to make predictions from our model.

III. Proposed problem statement

The proposed problem statement here is to build a model that could predict the results of an election with utmost accuracy. Our main goal here is to increase the efficiency and accuracy of the model used. We try to maximize our accuracy by using an ensemble classifier; the Random forest model.



IV. Proposed solution

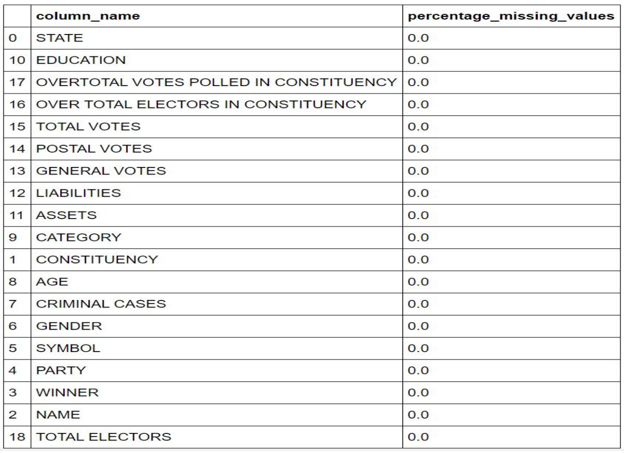
With a basic glance on the proposed problem, we find out that the solution to the problem should belong to the classification category since the candidate contesting the election can wether win the election or lose it, it's a case of binary classification. The classifier we'll be using is the Random forest classifier. By using an ensemble classifier we can aim for a higher accuracy and better prediction.

The first step is to *pre-process* and *visualize* the data:

The data is firstly uploaded and the required libraries are imported, the dataset has a total of 2263 rows and 19 columns. We then try to understand the data using the *describe()* function which gives us the following insights:

- There were 2018 candidates who contested the 2019 Lok Sabha Election.
- Minimum age of the candidates was 25 whereas maximum age was 86.
- Average age of all the candidates who contested election was 52.
- 19367 postal votes were casted in the election.

We then move on to find whether there are any missing values in the dataset. We find that the missing values(NaN) nentioned in the dataset are only for the NOTA votes, because these votes do not have any characteristics; heir missing values do not make a difference to our dataset. This confirms that our dataset has no missing values.



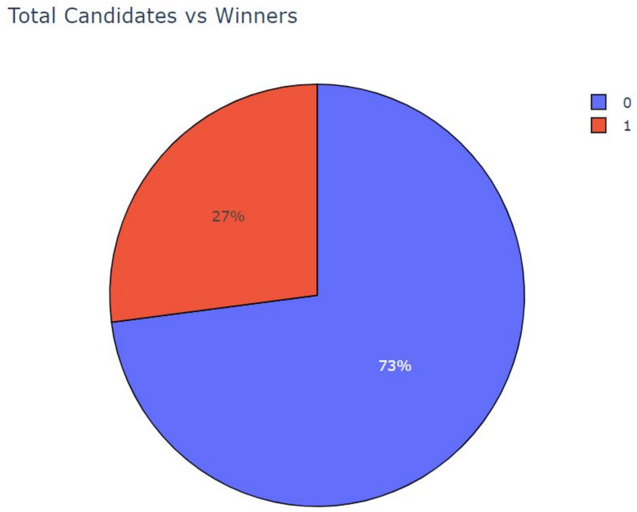
	column_name	percentage_missing_values
0	STATE	0.0
10	EDUCATION	0.0
17	OVERTOTAL VOTES POLLED IN CONSTITUENCY	0.0
16	OVER TOTAL ELECTORS IN CONSTITUENCY	0.0
15	TOTAL VOTES	0.0
14	POSTAL VOTES	0.0
13	GENERAL VOTES	0.0
12	LIABILITIES	0.0
11	ASSETS	0.0
9	CATEGORY	0.0
1	CONSTITUENCY	0.0
8	AGE	0.0
7	CRIMINAL CASES	0.0
6	GENDER	0.0
5	SYMBOL	0.0
4	PARTY	0.0
3	WINNER	0.0
2	NAME	0.0
18	TOTAL ELECTORS	0.0

In order to predict the target variable "WINNER", first we need to select the columns necessary for prediction and exclude all the redundant columns. Also we need to categorize the columns into categorical & numerical columns.The following columns are deemed necessary for prediction;*'STATE','CONSTITUENCY','WINNER','Party New', 'SYMBOL','GENDER','CRIMINAL CASES','AGE','CATEGORY','EDUCATION','TOTAL VOTES','TOTAL ELECTORS', 'ASSETS_RANGE', 'LIABILITY_RANGE'.*

The categorical columns are STATE',*'CONSTITUENCY','Party New','SYMBOL','GENDER','CATEGORY','EDUCATION','ASSETS_RANGE','LIABILITY_RANGE'.*

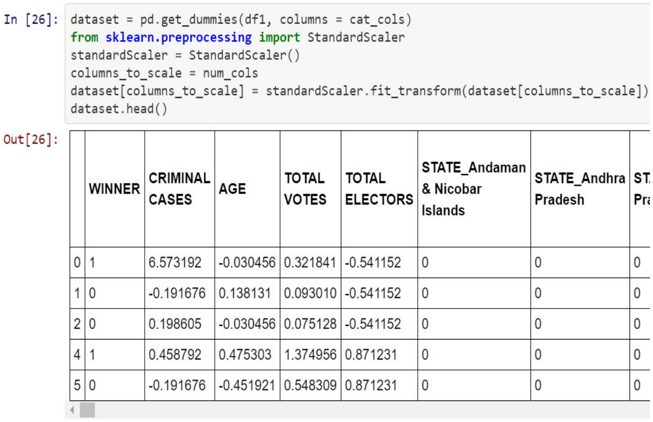
The numerical columns are *'CRIMINAL CASES','AGE','TOTAL VOTES','TOTAL ELECTORS'.*

Now, when we visualize the total candidates vs winners:


Total Candidates vs Winners

By looking at the above graph, we can clearly see that this dataset is imbalanced. So, in order to balance the dataset, we need to either upsample or down sample the dataset. Down sampling of the majority class might result in loss of some important information. So, we will be up-sampling the minority class using resample. Before doing that we need to scale the categorical columns using get_dummies and the numerical columns using StandardScaler library.

Data Scaling output:

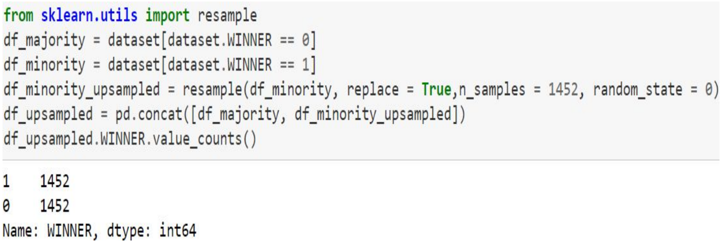


```
In [26]: dataset = pd.get_dummies(df1, columns = cat_cols)
         from sklearn.preprocessing import StandardScaler
         standardScaler = StandardScaler()
         columns_to_scale = num_cols
         dataset[columns_to_scale] = standardScaler.fit_transform(dataset[columns_to_scale])
         dataset.head()
```

Out[26]:

	WINNER	CRIMINAL CASES	AGE	TOTAL VOTES	TOTAL ELECTORS	STATE_Andaman & Nicobar Islands	STATE_Andhra Pradesh	ST Pra
0	1	6.573192	-0.030456	0.321841	-0.541152	0	0	0
1	0	-0.191676	0.138131	0.093010	-0.541152	0	0	0
2	0	0.198605	-0.030456	0.075128	-0.541152	0	0	0
4	1	0.458792	0.475303	1.374956	0.871231	0	0	0
5	0	-0.191676	-0.451921	0.548309	0.871231	0	0	0

Up-Sampling output:

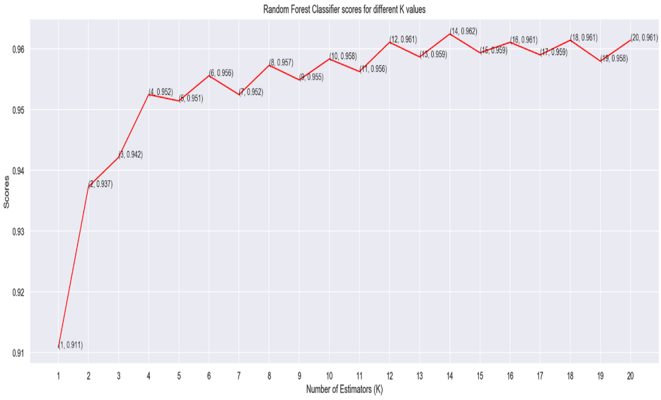


```
from sklearn.utils import resample
df_majority = dataset[dataset.WINNER == 0]
df_minority = dataset[dataset.WINNER == 1]
df_minority_upsampled = resample(df_minority, replace = True,n_samples = 1452, random_state = 0)
df_upsampled = pd.concat([df_majority, df_minority_upsampled])
df_upsampled.WINNER.value_counts()
```

```
1    1452
0    1452
Name: WINNER, dtype: int64
```

The next step is to build the model:

We will be using Random Forest Classifier to predict the winners of the election. In order to know the optimum number of trees required to predict the result with highest accuracy, we will be plotting the accuracy score for various values of k and will be selecting k values that give highest accuracy.

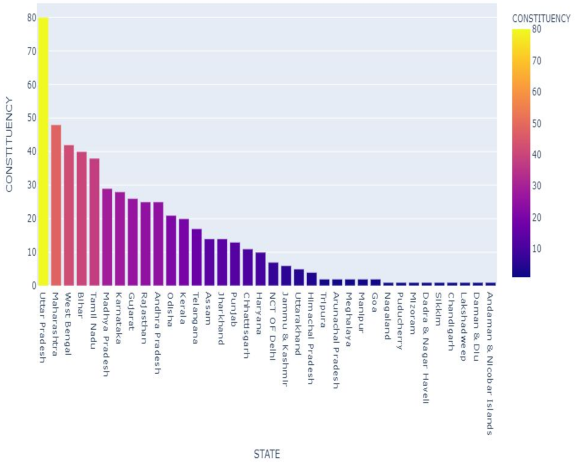

Random Forest Classifier scores for different K values

As we can see from the graph accuracy is maximum at k =14. Hence we will be selecting n_estimators=14.

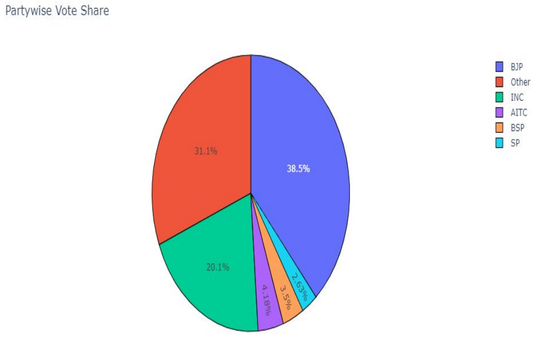
Using this value of estimator we calculate the cross validation score of our model and it turns out to be 96.2% accurate. This means that our model can correctly classify the winners 96% of the time.

V. EXPERIMENTAL RESULTS

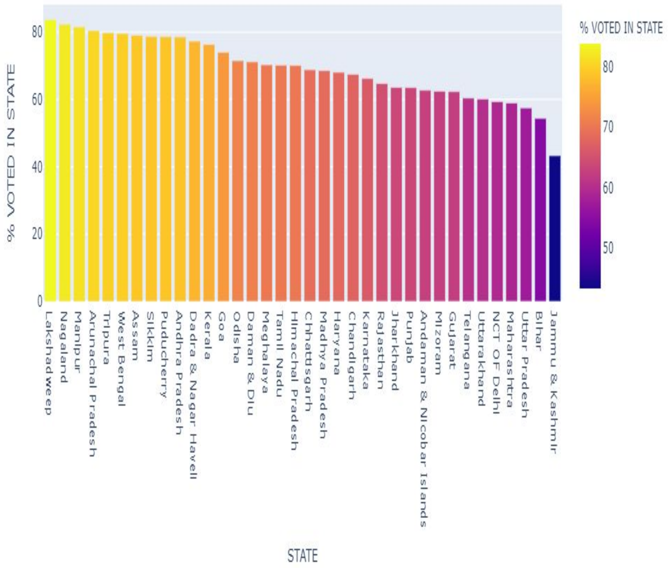
The dataset used here has a total of 2263 rows and 19 columns. The number of candidates who contested the election is 2018. The age of the candidates ranged from 25 to 86. The average age of the candidates was 52. The missing values are only in NOTA rows. Excluding these rows, there are no missing values in the dataset. The state with the maximum number of constituencies in Uttar Pradesh.



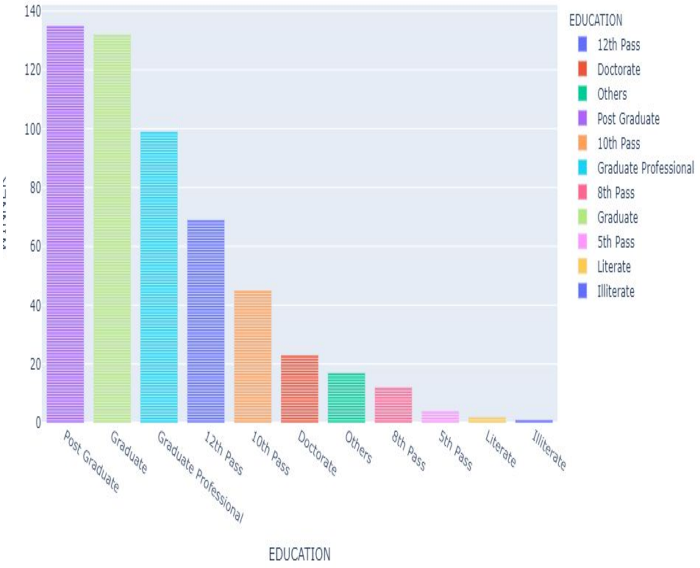
The party that had the maximum number of candidates was Bharatiya Janata Party with 420 candidates. This was followed by Indian National Congress with 413 candidates. The Bharatiya Janata Party also had the maximum vote share with 38.5%. Thus, the Bharatiya Janata Party also won the maximum number of seats with 300 seats.

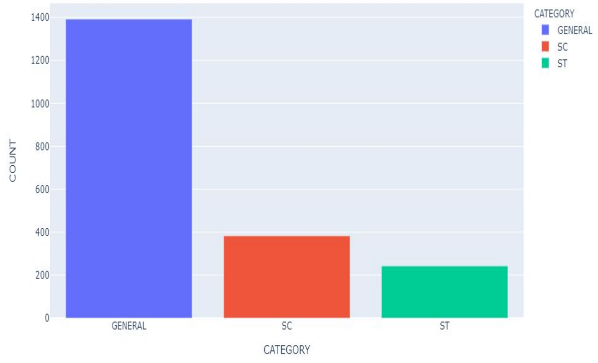


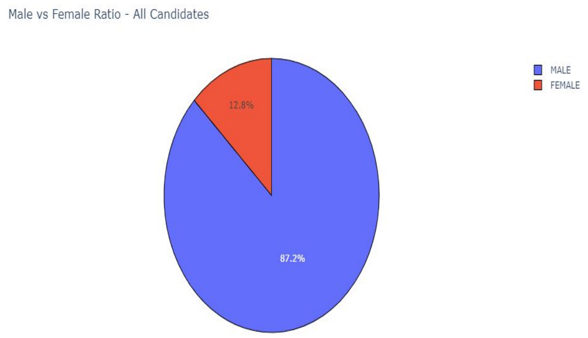
The female candidates had an average age that was slightly lesser compared to that of the male candidates. The youngest Member of Parliament was from the YSRCP party. The oldest member was from the SP party. The party with the maximum number of criminal cases on their candidates was the Bharatiya Janata Party.

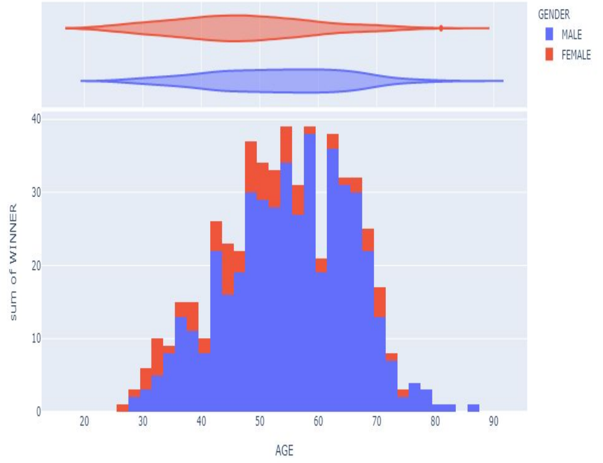


Most of the candidates in the election were graduates or post graduates. The state with highest voter turnout was Lakshadweep islands. The state with the maximum number of eligible voters was UP. The general category saw the maximum number of candidates fielded followed by SC and ST. The percentage of male candidates massively outnumbered the female candidates.







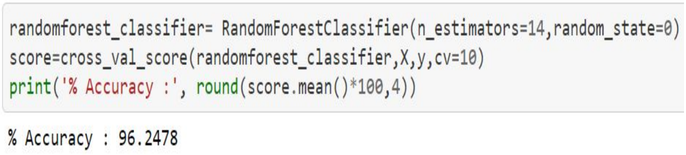


The dataset that we have used here was not balanced. In order to balance the dataset we can use two techniques which are downsampling and upweighting. In our model, we have used upweighting as downsampling results in loss of information.

Downsampling refers to training on disproportionate subsets of the majority class. Downsampling adds tremendous importance to the minor class and thus brings down precision. Low precision answers are uncommon. Upsampling reduces the weight on the minority classes. The precision is much better with upsampling. Downsampling is more preferred in cases where there is excessive data than the required amount.

In our model we have used the random forest classifier to predict the winner. The random forest classifier is highly accurate due to the number of trees it uses in classification. In order to obtain high accuracy, we need to determine the optimum number of trees. For this, we will be plotting the accuracy values for various values of k and the k value with the highest accuracy will be selected.

The accuracy from the graph plotted was k=14. Hence we have selected n_estimators = 14. This estimator value is used to cross validate the score of our model. The accuracy that we have achieved is 96.2%. This is a very good accuracy rate as this signifies the fact that the model that we have built will predict the winners accurately 96.2% of the time. Hence, our model is highly accurate.



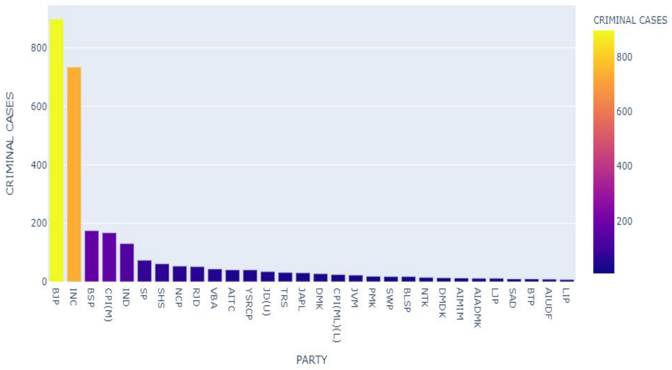
```
randomforest_classifier= RandomForestClassifier(n_estimators=14,random_state=0)
score=cross_val_score(randomforest_classifier,X,y,cv=10)
print('% Accuracy :', round(score.mean()*100,4))

% Accuracy : 96.2478
```

VI.CONCLUSION

Each member of the team contributed equally. Initially we individually found different relations between the attributes of the dataset. Later we performed EDA on these relations using various visualizations. Further, we balanced the dataset using data scaling. We tried different classifiers and regression analysis. We found that Random Forest Classifier was providing more accuracy.

For further extreme accuracy and analysis, hyper parameter tuning can be opted along with Random Forest Classifier. The below graph is an interesting find. Most of the election candidates have criminal cases, but still they are elected.



The experiences we learnt during the course of this project were vivid and interesting. We imagined how the professional data scientist would use these methods for predicting elections, weather or other things.

VII.REFERENCES

[1] Zhang M, Alvarez RM, Levin I (2019) Election forensics: Using machine learning and synthetic data for possible election anomaly detection. PLoS ONE 14(10): e0223950

[2] Fauzi, Muhammad. (2018). Random Forest Approach for Sentiment Analysis in Indonesian Language. Indonesian Journal of Electrical Engineering and Computer Science. 12. 46-50. 10.11591/ijeecs.v12.i1.pp46-50.

[3] Ali, Jehad & Khan, Rehanullah & Ahmad, Nasir & Maqsood, Imran. (2012). Random Forests and Decision Trees. International Journal of Computer Science Issues (IJCSI) Vol. 9, Issue 5, No 3, September 2012