**Predictive Modeling of Indian Elections Using Machine Learning**

AUTHOR – SHIGIL.S

SUBMISSION – 16-08-2024

EVALUTION – 29-08-2024

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **OVERVIEW OF PROBLEM STATEMENT** | **3** |
| **OBJECTIVE** | **4** |
| **DATA COLLECTION** | **4** |
| **DATA DESCRIPTION** | **5** |
| **EXPLORATORY DATA ANALYSIS** | **6** |
| **DATA PREPROCESSING** | **7** |
| **VISUALIZATION** | **8** |
| **FEATURE ENGINEERING** | **9** |
| **FEATURE SELECTION** | **9** |
| **MODEL SELECTION** | **10** |
| **MODEL EVALUATION** | **13** |
| **HYPERPARAMETER TUNING** | **14** |

**OVERVIEW OF PROBLEM STATEMENT:**

Elections are a vital aspect of any democratic nation, including India, where diverse social, economic, and demographic factors influence electoral outcomes. Understanding these factors and predicting election results can provide valuable insights for political parties, analysts, and policymakers.

This project aims to leverage machine learning techniques to analyze the Indian election dataset, which includes data from various states and constituencies across different election years. The primary objective is to build predictive models that can identify key factors influencing election outcomes and forecast the results based on historical data.

The project will involve data preprocessing, exploratory data analysis, feature engineering, and the application of machine learning algorithms to develop models that can predict election outcomes with a reasonable degree of accuracy. The results of this analysis can help in understanding voter behavior, the impact of socio-economic factors, and the effectiveness of campaign strategies.

**KEY GOALS:**

* To analyze the historical election data of India to identify trends and patterns.
* To build and evaluate machine learning models that can predict election outcomes.
* To gain insights into the factors that most significantly influence election results.

**OBJECTIVE**

The objective of this project is to develop and implement machine learning models to analyze and predict election outcomes in India using historical election data. Specifically, the project aims to:

1. **Analyze Election Data**: Explore the Indian election dataset to identify significant patterns, trends, and correlations that influence election results across different states and constituencies.
2. **Predict Election Outcomes**: Build predictive models using machine learning algorithms to forecast election results based on historical data, including voter demographics, socio-economic factors, and previous election outcomes.
3. **Evaluate Model Performance**: Assess the accuracy and effectiveness of various machine learning models in predicting election outcomes, and compare their performance to identify the most suitable approach.
4. **Gain Insights into Electoral Factors**: Identify and interpret the key factors that drive electoral outcomes in India, providing actionable insights for political analysts, parties, and policymakers.
5. **Provide a Data-Driven Decision Support Tool**: Create a reliable model that can be used as a decision-support tool for forecasting future election results, helping stakeholders make informed decisions based on data-driven analysis.

**DATA COLLECTION:**

INDIAN ELECTION DATASET FROM KAGGLE

<https://www.kaggle.com/datasets/awadhi123/indian-election-dataset>

**DATA DESCRIPTION:**

Dataset Name - Indian Election dataset This database contains detailed candidate‐level data for elections to the lower houses of India’s national and state legislatures, i.e., the Lok Sabha and Vidhan Sabhas. The data span 1977‐2015, with each row representing a candidate that ran for office in that state‐year.

Variable name : Storage: type Variable contents

st\_name : str35 : State

Year : Int : General election year

pc\_no : Byte : Parliamentary constituency number

pc\_name : str25 : Parliamentary constituency name

pc\_type : str3 : Parliamentary constituency reservation status

cand\_name : str70 : Candidate name

cand\_sex : str1 : Candidate sex

partyname : str57 : Party name

partyabbre : str10 : Party abbreviation

totvotpoll : Long : Votes received

electors : Long : Number of registered voters

Potential Target Variables:

totvotpoll: The total votes polled for each candidate.

partyname: The party name if the analysis is focused on predicting or analyzing party performance.

For this dataset I am choosing the target variable: **Total Votes Polled (totvotpoll)**: This is a straightforward target for predicting the number of votes a candidate receives. It provides a quantitative outcome that can be modeled using regression techniques.

**EXPLORATORY DATA ANALYSIS**

**Try to Understand the data in length:**

In this dataset we have 73081 rows and 11 columns.

**Identify the Numerical and Categorical Values:**

**Numerical Values:**

1. Year
2. Pc\_no
3. Totvotpoll
4. Electors

**Categorical Values:**

1. State
2. Pc\_Name
3. Cand\_Name
4. Cand\_Sex
5. Partyname
6. Partyabbre
7. Pc\_Type

**Check for Null values:**

State:0

Year:0

pc\_no:0

pc\_name:0

pc\_type:8070

Candidate:0

cand\_sex:542

Party:0

Totvotpoll:0

Electors:0

**Perform Descriptive Statistics to understand the distribution of data:**



**Data Preprocessing**

**Handling the Null Values:**

In handling the null values, we got two columns has null values in Pc\_Type, Cand\_sex. In Pc\_type we are going to replace the values the null values as Others and In Cand\_sex we are going to replace the values as Others.

**Removing unnecessary Rows:**

In performing the descriptive statistics, we saw the totvotpoll the minimum value has 0. After finding that there are 5 rows which contains 0 totvotpoll. So that we had decide we remove these 5 rows.

**Removing unnecessary Columns:**

In the given data we are going to remove PartyName.

**Renaming the column names:**

As reading the data we are going to change some column names:

st\_name': 'State' , 'year': 'Year', 'cand\_name': 'Candidate', 'partyabbre': 'Party'.

**Outlier Detection and treatment:**

As outlier detection we used boxplot to identify the outlier as for this data we have outlier detection for the numerical values we can’t able to remove the outlier data's even after the z-score there are huge data as outlier. As for treatment we are going to do Standard Scaler.

**Skewness:**

As for skewness before Standard Scaler we checked the skewness as for electors it was distributed and year was also distributed but for totvotpoll and pc\_no was rightly skewed after standard scaler the electors and year was same but there was no change in totvotpoll and pc\_no.

**Visualization**

**Boxplot:**

Used boxplot for outlier detection before and after transformation.

**Histogram:**

Used histogram for skewness before and after transformation.

**Pair plot:**

Used pair plot to help to visualize the numeric columns.

**Bar Chart:**

For this data bar chart is very important. We have many things using bar chart like:

1. Top 10 States by Number of Parliamentary Constituencies
2. Top 10 Parties by Count
3. Top 10 candidates (Used Vertical bar chart)
4. Year wise top 8 party counts

**Pie Chart:**

Used pie chart for Parliamentary Constituency Reservation Status.

**Line Plot:**

Used line plot for total electors by year wise.

**FEATURE ENGINEERING**

**Feature Extraction:**

In feature extraction we have added a new column as voting %. So that we can identify each candidate votes percentage.

**Feature Transformation:**

For feature transformation we did label encoding for the categorical variables.

After label encoding do the Standard Scaler transformation.

**FEATURE SELECTION**

To select the relevant features that have the most impact on the target variable.

The feature selection that we used is wrapper method and used an algorithm as

RECURSIVE FEATURE ELMINATION and used the model as LINEAR

REGRESSION.

**DATA SPLITTING**

After label encoding and Standard Scaler transformation, we need to split the data for training and testing.

**MODEL SELECTION**

As you see our target variable is totvotpoll it’s not categorical variable, it is a continuous variable so that we can choose the Regression Algorithm.

**REGRESSION ALOGRITHM:**

1. Linear Regression
2. Decision Tree Regression
3. Support Vector Regression
4. Lasso Regression
5. Random Forest Regression
6. Gradient Boosting Regression

**MODEL EVALUATION**

Evaluate model performance using appropriate evaluation metrics

**1.Linear Regression**

MAE: 0.1765

R²: 0.8964

MSE: 0.1015

RMSE: 0.3186

**Analysis**: The performance of Linear Regression has improved significantly compared to previous iterations. With an R² of 0.8964, the model now explains almost 90% of the variance, and the error metrics have decreased, showing that the model is making more accurate predictions.

**2.Decision Tree**

MAE: 0.0634

R²: 0.9709

MSE: 0.0285

RMSE: 0.1687

**Analysis**: The Decision Tree model continues to perform well, with an R² of 0.9709, indicating that it explains 97.1% of the variance. The low error metrics suggest that the model is still making accurate predictions, though slightly less accurate than before scaling and transformation adjustments.

**3.Support Vector Regression (SVR)**

MAE: 0.4779

R²: -0.1219

MSE: 1.0991

RMSE: 1.0484

**Analysis**: The SVR model still underperforms, with a negative R² value indicating that it does worse than a simple mean model. Despite scaling, the SVR model seems unsuitable for this dataset.

**4.Lasso**

MAE: 0.1663  
  
R²: 0.8625  
  
MSE: 0.1347  
  
RMSE: 0.3671

**Analysis**: The Lasso model shows decent performance with an R² of 0.8625, explaining about 86.2% of the variance. It performs similarly to Linear Regression but slightly worse, which is expected as Lasso includes regularization.

**5.Random Forest**

MAE: 0.0491

R²: 0.9794

MSE: 0.0201

RMSE: 0.1419

**Analysis**: The Random Forest model continues to perform exceptionally well, with a high R² and low error metrics. This model remains a strong contender, providing accurate predictions with minimal error.

**6.Gradient Boosting**

MAE: 0.0067

R²: 0.9996

MSE: 0.0004

RMSE: 0.0200

**Analysis**: Gradient Boosting continues to outperform all other models, with an R² close to 1 and extremely low error metrics. This model is the best performer for this dataset, providing highly accurate predictions.

As to select a model we will choose Linear regression but other model has performed well except support vector regression. Why we choose linear regression another model can be overfitting. linear regression will be the best model.

**HYPERPARAMETER TUNING**

In Hyperparameter Tuning we are going to choose the random search because as you see this dataset is very huge go that we can use random search will be very useful.

**Best Parameters:**

fit\_intercept: True

Performance Metrics:

Mean Absolute Error (MAE): 0.1765

R-squared (R²): 0.8964

Mean Squared Error (MSE): 0.1015

Root Mean Squared Error (RMSE): 0.3186

**Analysis**:

**Fit Intercept**: The best parameter found was to include the intercept(fit\_intercept=True), which is standard in most linear regression problems.

**R² Value**: An R² of 0.8964 indicates that the model explains approximately 89.6% of the variance in the target variable. This is a strong performance for a linear model, especially after tuning.

**Error Metrics**: The MAE, MSE, and RMSE values are relatively low, suggesting that the model is making accurate predictions.

**Conclusion**: The Linear Regression model has performed well after Random Search tuning, achieving a good balance between simplicity and accuracy. With an R² close to 0.9, the model is explaining a substantial amount of the variance, and the error metrics indicate reasonable prediction accuracy.

If we are satisfied with these results, we can confidently use this model for prediction.

**SAVE THE MODEL**

After we selected Linear Regression save the model to test the unseen data.