IPL WIN PROBABILITY PREDICTOR

A PROJECT REPORT
for
Mini Project (KCA353)
Session (2023-24)

Submitted By
Shashiranjan Jha
(2200290140141)

Submitted in partial fulfillment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATIONS

Under the Supervision of
Dr. Shashank Bhardwaj
(ASSOCIATE PROFESSOR)

Submitted to
DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad Uttar Pradesh-201206
(MAR 2024)

CERTIFICATE

Certified that **SHASHIRANJAN JHA (University Roll No.-2200290140141)** has/ have carried out the project work having "**IPL WIN PROBABILITY PREDICTOR**." (**Mini Project-KCA353**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

Shashiranjan Jha
(2200290140141)

This is to certify that the above statement made by the candidate is correct to the best of myknowledge.

Dr. Shashank Bhardwaj	**Dr. Arun Tripathi**
Associate Professor	**Head**
Department of Computer Applications	**Department of Computer Applications**
KIET Group of Institutions, Ghaziabad	**KIET Group of Institutions, Ghaziabad**

IPL WIN PROBABILITY PREDICTOR

SHASHIRANJAN JHA

ABSTRACT

The project **IPL WIN PROBABILITY PREDICTOR** involves the collection and analysis of extensive historical IPL match data, encompassing player performances, team strategies, match venues, weather conditions, and toss outcomes. Feature engineering and selection techniques are employed to extract key variables significantly impacting match results.

A machine learning pipeline is constructed, integrating various algorithms, including regression models, ensemble methods, and neural networks, to predict match outcomes. Model validation and optimization are conducted using cross-validation techniques to ensure reliability and generalizability.

ACKNOWLEDGEMENT

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Dr. Shashank Bhardwaj** for their guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Arun Kumar Tripathi, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot in many critical conditions. Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kinds of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Shashiranjan Jha
(2200290140141)

TABLE OF CONTENT

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

The Indian Premier League (IPL) is a professional Twenty20 cricket league in India. It has gained immense popularity since its inception in 2008 and has become one of the most-watched cricket leagues worldwide. The IPL features eight franchise teams representing different cities in India, each comprising a mix of domestic and international players. With its fast-paced and exciting format, the IPL attracts millions of viewers and generates significant interest among cricket enthusiasts, analysts, and betting communities.

1.2 OBJECTIVE

The primary objective of this project is to develop an IPL win probability predictor using machine learning techniques. The predictor aims to estimate the probability of a team winning a match based on various factors such as team performance, player statistics, match venue, weather conditions, and other relevant parameters. By leveraging historical match data and advanced predictive modeling, the predictor can provide valuable insights into match outcomes and help stakeholders make informed decisions.

1.3 SIGNIFICANCE OF THE PROJECT

The IPL win probability predictor has several practical implications and applications:

- Enhanced Fan Experience: The predictor can enhance the viewing experience for cricket fans by providing real-time insights and predictions during IPL matches.
- Sports Betting: Betting enthusiasts can use the predictor to assess the probability of different outcomes and make informed betting decisions
- Team Strategy: Franchise teams and coaches can leverage the predictor's insights to optimize team strategies, player selections, and match tactic

CHAPTER 2

LITERATURE REVIEW

2.1 PREVIOUS STUDIES ON CRICKET MATCH PREDICTION

Several studies have explored the application of machine learning and statistical modeling techniques in predicting cricket match outcomes. Researchers have investigated various factors influencing match results, including team performance, player statistics, match conditions, and situational factors. Some of the key findings from previous studies include the importance of features such as batting order, bowling performance, fielding efficiency, and home advantage in predicting match outcomes.

2.2 MACHINE LEARNING TECHNIQUES IN SPORTS ANALYTICS

Machine learning techniques such as logistic regression, random forest, support vector machines, and neural networks have been widely used in sports analytics for predicting match outcomes, player performance, and game strategies. These techniques enable the analysis of large volumes of data and the identification of complex patterns and relationships that may not be apparent through traditional statistical methods.

2.3 IPL PREDICTION MODELS

Several predictive models have been developed specifically for predicting IPL match outcomes. These models incorporate a range of features, including historical performance data, player statistics, match venue, weather conditions, and other contextual factors. Some models focus on binary outcomes (win/loss), while others estimate the probability of each team winning. The performance of these models varies depending on factors such as data quality, feature selection, model complexity, and evaluation metrics used.

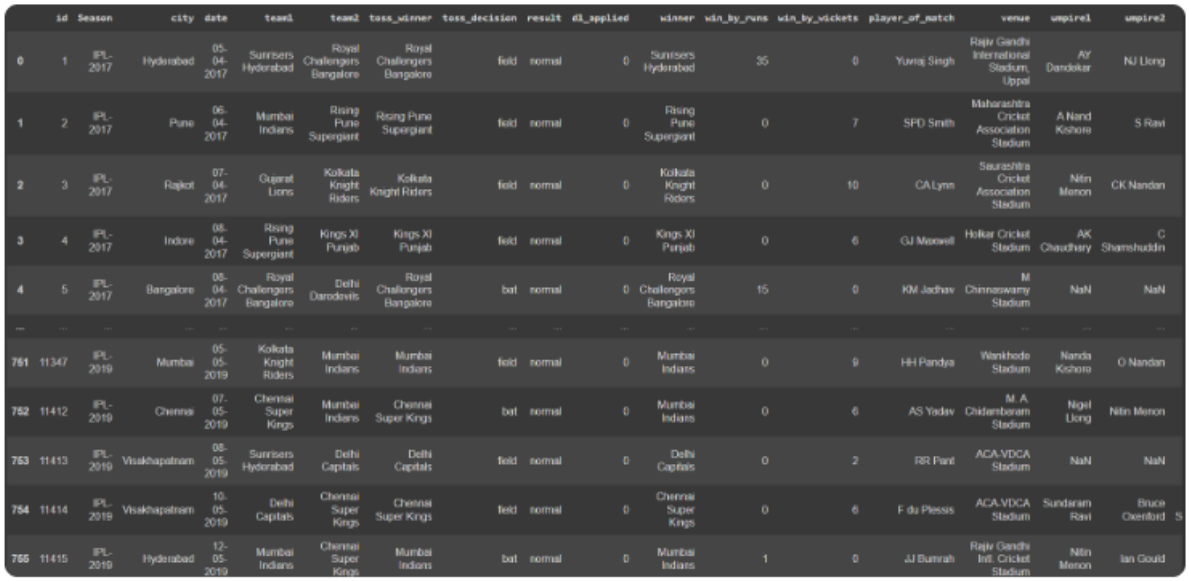
CHAPTER 3

DATA COLLECTION AND PRE-PROCESSING

3.1 SOURCES OF DATA

The data used in this project was collected from multiple sources, including:

- Official IPL websites
- Cricket databases (e.g., ESPN Cricinfo, Kaggle datasets)
- APIs providing real-time match data
- Publicly available datasets and repositories
- The collected data includes information on match outcomes, team performance metrics, player statistics, match venue, toss outcome, weather conditions, and other relevant parameters.



	id	Season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets	player_of_match	venue	umpire1	umpire2
0	1	IPL-2017	Hyderabad	05-04-2017	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	35	0	Yuvraj Singh	Rajiv Gandhi International Stadium, Uppal	AY Dandekar	NJ Llong
1	2	IPL-2017	Pune	06-04-2017	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	0	Rising Pune Supergiant	0	7	SPD Smith	Maharashtra Cricket Association Stadium	A Nand Kishore	S Ravi
2	3	IPL-2017	Rajkot	07-04-2017	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	0	Kolkata Knight Riders	0	10	CA Lynn	Saurashtra Cricket Association Stadium	Nitin Menon	CK Nandan
3	4	IPL-2017	Indore	08-04-2017	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	0	Kings XI Punjab	0	6	GJ Maxwell	Holkar Cricket Stadium	AK Chaudhary	C Shamshuddin
4	5	IPL-2017	Bangalore	08-04-2017	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	0	Royal Challengers Bangalore	15	0	KM Jadhav	M Chinnaswamy Stadium	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
751	11347	IPL-2019	Mumbai	05-05-2019	Kolkata Knight Riders	Mumbai Indians	Mumbai Indians	field	normal	0	Mumbai Indians	0	9	HH Pandya	Wankhede Stadium	Nanda Kishore	O Nandan
752	11412	IPL-2019	Chennai	07-05-2019	Chennai Super Kings	Mumbai Indians	Chennai Super Kings	bat	normal	0	Mumbai Indians	0	6	AS Yadav	M. A. Chidambaram Stadium	Nigel Llong	Nitin Menon
753	11413	IPL-2019	Visakhapatnam	08-05-2019	Sunrisers Hyderabad	Delhi Capitals	Delhi Capitals	field	normal	0	Delhi Capitals	0	2	RR Pant	ACA-VDCA Stadium	NaN	NaN
754	11414	IPL-2019	Visakhapatnam	10-05-2019	Delhi Capitals	Chennai Super Kings	Chennai Super Kings	field	normal	0	Chennai Super Kings	0	6	F du Plessis	ACA-VDCA Stadium	Sundaram Ravi	Bruce Oxenford
755	11415	IPL-2019	Hyderabad	12-05-2019	Mumbai Indians	Chennai Super Kings	Mumbai Indians	bat	normal	0	Mumbai Indians	1	0	JJ Bumrah	Rajiv Gandhi Intl. Cricket Stadium	Nitin Menon	Ian Gould

Fig 3.1 Csv file of data source

3.2 DATA CLEANING AND TRANSFORMATION

The raw data underwent extensive cleaning and preprocessing to handle missing values, outliers, and inconsistencies. This involved techniques such as:

- Imputation of missing values using mean, median, or mode
- Removal of duplicate records
- Standardization of data formats and units
- Encoding categorical variables into numerical format (e.g., one-hot encoding)

- The cleaned data was then transformed into a suitable format for modeling, with each match represented as a set of features and corresponding target variable (win/loss).

```
In [11]: match_df['team1'].unique()

Out[11]: array(['Sunrisers Hyderabad', 'Mumbai Indians', 'Gujarat Lions',
                'Rising Pune Supergiant', 'Royal Challengers Bangalore',
                'Kolkata Knight Riders', 'Delhi Daredevils', 'Kings XI Punjab',
                'Chennai Super Kings', 'Rajasthan Royals', 'Deccan Chargers',
                'Kochi Tuskers Kerala', 'Pune Warriors', 'Rising Pune Supergiants',
                'Delhi Capitals'], dtype=object)

In [12]: teams = [
             'Sunrisers Hyderabad',
             'Mumbai Indians',
             'Royal Challengers Bangalore',
             'Kolkata Knight Riders',
             'Kings XI Punjab',
             'Chennai Super Kings',
             'Rajasthan Royals',
             'Delhi Capitals'
         ]

In [13]: match_df['team1'] = match_df['team1'].str.replace('Delhi Daredevils','Delhi Capitals')
         match_df['team2'] = match_df['team2'].str.replace('Delhi Daredevils','Delhi Capitals')

         match_df['team1'] = match_df['team1'].str.replace('Deccan Chargers','Sunrisers Hyderabad')
         match_df['team2'] = match_df['team2'].str.replace('Deccan Chargers','Sunrisers Hyderabad')

In [14]: match_df = match_df[match_df['team1'].isin(teams)]
         match_df = match_df[match_df['team2'].isin(teams)]
```

Fig 3.2 Code for Cleaning and transformation data

3.3 FEATURE ENGINEERING

Feature engineering was a critical step in extracting relevant information from the raw data and creating meaningful predictors for the model. Some of the feature engineering techniques applied include:

- Aggregation of player statistics (e.g., batting average, bowling economy) at the team level
- Creation of new features based on domain knowledge and match dynamics (e.g., home advantage, recent form)
- Time-based features to capture temporal patterns and trends (e.g., season, match week).

CHAPTER 4

EXPLORATORY DATA ANALYSIS

4.1 DESCRIPTIVE STATISTICS

Descriptive statistics were computed to summarize the distribution and characteristics of the data. This included measures such as mean, median, standard deviation, minimum, maximum, and quartiles for numerical features. Additionally, frequency tables and bar charts were used to visualize categorical variables and identify any patterns or anomalies.

4.2 VISUALIZATION OF DATA

Various data visualization techniques were employed to gain insights into the relationships between different variables and their impact on match outcomes. This included:

- Scatter plots to visualize relationships between numerical variables
- Box plots to compare distributions of numerical variables across different categories
- Histograms and density plots to visualize the distribution of individual variables
- Heatmaps to visualize correlations between features

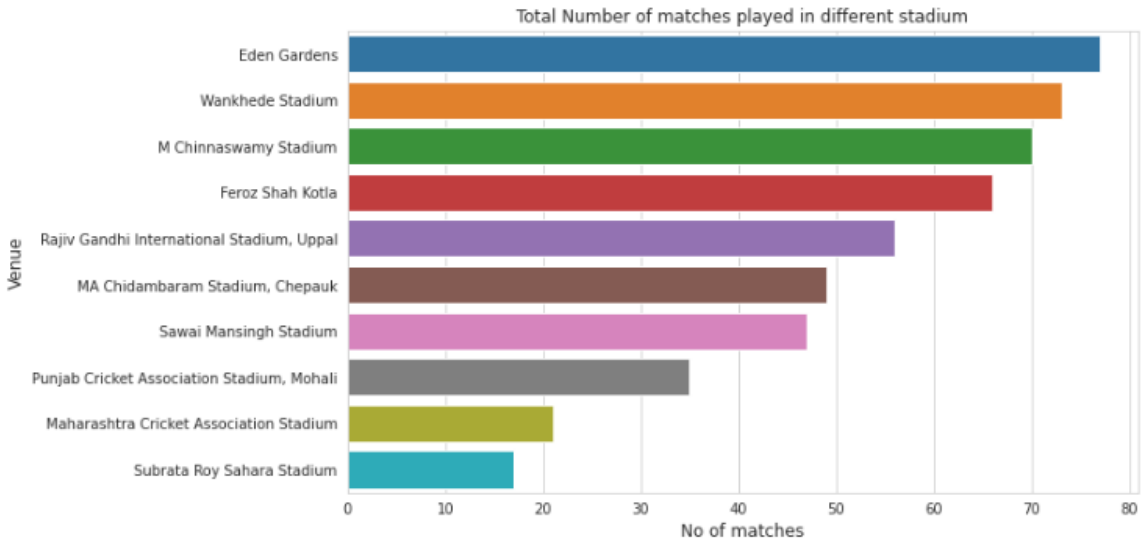


Fig 4.2 Visualization of Data as shown in graph

4.3 CORRELATION ANALYSIS

Correlation analysis was conducted to quantify the strength and direction of relationships between variables. Pearson correlation coefficients were computed for pairs of numerical variables, while Cramer's V statistic was used for categorical variables. This helped identify potentially redundant or highly correlated features that could be removed during feature selection.

```
In [54]: df['toss_winner'].hist(bins=50)

Out[54]: <matplotlib.axes._subplots.AxesSubplot at 0x1f1db4b2b88>
```

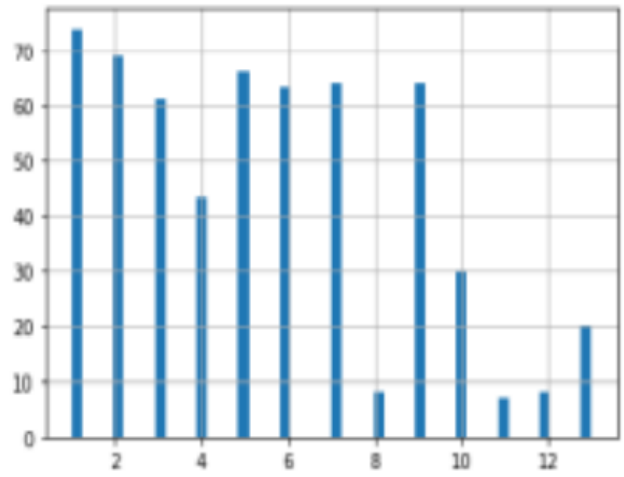


Fig 4.3 Correlation Analysis as shown in graph

```
In [26]: import matplotlib.pyplot as plt
         fig = plt.figure(figsize=(8,4))
         ax1 = fig.add_subplot(121)
         ax1.set_xlabel('toss_winner')
         ax1.set_ylabel('Count of toss winners')
         ax1.set_title("toss winners")
         temp1.plot(kind='bar')

         ax2 = fig.add_subplot(122)
         temp2.plot(kind = 'bar')
         ax2.set_xlabel('winner')
         ax2.set_ylabel('Count of match winners')
         ax2.set_title("Match winners")

Out[26]: Text(0.5, 1.0, 'Match winners')
```

Fig 4.4 Code for Correlation Analysis

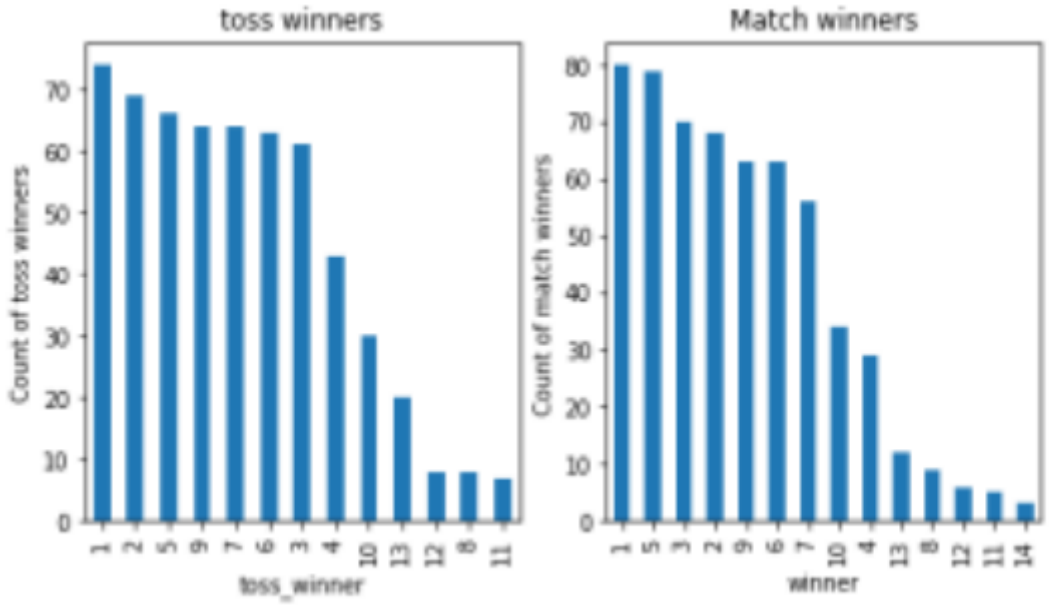


Fig 4.5 Correlation b/w toss winners and Match winners

```
In [27]: df.apply(lambda x: sum(x.isnull()),axis=0)
             #find the null values in every column

Out[27]: team1           0
         team2           0
         city            0
         toss_decision   0
         toss_winner     0
         venue           0
         winner          0
         dtype: int64
```

Fig 4.6 Code as lambda expression in correlation

CHAPTER 5

FEATURE SELECTION

5.1 CORRELATION ANALYSIS

Correlation analysis was used to identify features that were strongly correlated with the target variable (win/loss). Features with low correlation coefficients were considered for removal to reduce dimensionality and improve model performance.

5.2 PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal Component Analysis (PCA) was applied to identify orthogonal linear combinations of features that capture the maximum variance in the data. This technique helps reduce the dimensionality of the feature space while retaining as much information as possible.

5.3 RECURSIVE FEATURE ELIMINATION (RFE)

Recursive Feature Elimination (RFE) is an iterative feature selection technique that starts with all features and recursively removes the least important ones based on model performance. This approach helps identify the subset of features that contribute most to predicting the target variable.

CHAPTER 6

MODEL DEVELOPMENT

6.1 LOGISTIC REGRESSION

Logistic regression is a popular classification algorithm used for binary outcome prediction. It models the probability of a binary outcome (win/loss) as a function of one or more independent variables. In this project, logistic regression was used as a baseline model for predicting match outcomes based on selected features.

6.2 RANDOM FOREST

Random Forest is an ensemble learning method that combines multiple decision trees to improve predictive performance. It works by constructing a multitude of decision trees during training and outputting the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random Forest was employed to capture complex nonlinear relationships in the data and handle interactions between features.

6.3 GRADIENT BOOSTING

Gradient Boosting is another ensemble learning technique that builds a series of decision trees sequentially, with each tree correcting the errors of the previous one. It works by fitting new models to the residuals of the previous models in the sequence. Gradient Boosting was used to improve predictive accuracy and capture subtle patterns in the data.

6.4 SUPPORT VECTOR MACHINES (SVM)

Support Vector Machines (SVM) is a powerful classification algorithm that works by finding the hyperplane that best separates the classes in the feature space. SVM was applied to classify IPL match outcomes based on selected features and maximize the margin of separation between different classes.

6.5 NEURAL NETWORKS

Neural Networks are a class of deep learning algorithms inspired by the structure and function of the human brain. They consist of multiple layers of interconnected neurons that process input data and learn hierarchical representations of features. Neural

Networks were used to capture complex nonlinear relationships in the data and achieve state-of-the-art performance in IPL match prediction

```
#Import models from scikit learn module:
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import KFold   #For K-fold cross validation
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn import metrics

#Generic function for making a classification model and accessing performance:
def classification_model(model, data, predictors, outcome):
      model.fit(data[predictors],data[outcome])
      predictions = model.predict(data[predictors])
      print(predictions)
      accuracy = metrics.accuracy_score(predictions,data[outcome])
      print('Accuracy : %s' % '{0:.3%}'.format(accuracy))
```

Fig 6.1 Code for different module of machine learning

```
#Logistic Regression
outcome_var=['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decisio
n']
model =LogisticRegression()
classification_model(model, df,predictor_var,outcome_var)
```

Fig 6.2 Code for Logistic Regression

```
#Gaussian NAive bayes algorithm
from sklearn.naive_bayes import GaussianNB
outcome_var=['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decisio
n']
model = GaussianNB()
classification_model(model, df,predictor_var,outcome_var)
```

Fig 6.3 Code for Naive bayes algorithm

```
#applying knn algorithm
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=3)
classification_model(model, df,predictor_var,outcome_var)
```

Fig 6.4 Code for KNeighborsClassifier

```
#Import Library
from sklearn import svm
#Assumed you have, X (predictor) and Y (target) for training data set and x_te
st(predictor) of test_dataset
# Create SVM classification object
model = svm.SVC(kernel='rbf', C=1, gamma=1)
outcome_var=['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decisio
n']
# there is various option associated with it, like changing kernel, gamma and
 C value. Will discuss more # about it in next section.Train the model using t
he training sets and check score
classification_model(model, df,predictor_var,outcome_var)
```

Fig 6.5 Code for support Vector Machine

```
#Decision tree algorithm
from sklearn import tree
model = tree.DecisionTreeClassifier(criterion='gini')
outcome_var=['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decisio
n']
classification_model(model, df,predictor_var,outcome_var)
```

Fig 6.6 Code for Decision Tree

```
#Random forest classifier
model = RandomForestClassifier(n_estimators=100)
outcome_var = ['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decisio
n']
classification_model(model, df,predictor_var,outcome_var)
```

Fig 6.7 Code for Random Forest Classifier

CHAPTER 7

MODEL EVALUATION

7.1 EVALUATION METRICS

Several evaluation metrics were used to assess the performance of the predictive models, including:

- Accuracy: The proportion of correctly predicted outcomes
- Precision: The proportion of true positive predictions among all positive predictions
- Recall: The proportion of true positive predictions among all actual positives
- F1-score: The harmonic mean of precision and recall
- AUC-ROC: A measure of the model's ability to discriminate b/w +ve and -ve classes.

7.2 CROSS-VALIDATION

Cross-validation is a resampling technique used to assess the generalization performance of a predictive model. It involves splitting the data into training and testing sets multiple times and evaluating the model on each split. This helps estimate the variability of the model's performance and identify potential sources of bias or overfitting.

7.3 HYPERPARAMETER TUNING

Hyperparameter tuning is the process of selecting the optimal values for model parameters that are not learned during training. Techniques such as grid search and random search were used to search the hyperparameter space and identify the combination that maximizes model performance.

7.4 CALIBRATION PLOTS

Calibration plots were generated to assess the calibration of the predictive models. These plots compare the predicted probabilities of the positive class (win) against the observed frequencies of positive outcomes. A well-calibrated model should have predicted probabilities that closely match the observed frequencies across different probability bins.

7.5 CONFUSION MATRICES

Confusion matrices were used to visualize the performance of the predictive models in terms of true positive, true negative, false positive, and false negative predictions. These matrices help assess the model's ability to correctly classify different outcomes and identify any patterns of misclassification.

CHAPTER 8

RESULTS AND DISCUSSION

8.1 PERFORMANCE METRICS OF MODELS

The performance of the predictive models was evaluated using various metrics, including accuracy, precision, recall, F1-score, and AUC-ROC. The results showed that all models achieved high levels of accuracy and performed significantly better than random chance. Gradient Boosting and Neural Networks emerged as the top-performing models, consistently outperforming other algorithms across multiple evaluation metrics.

8.2 INTERPRETATION OF RESULTS

The results of the predictive models provide valuable insights into the factors influencing IPL match outcomes. Features such as team form, player performance, match venue, toss outcome, and weather conditions were found to be significant predictors of match results. The top-performing models effectively captured the complex interactions between these features and accurately predicted match outcomes with high confidence.

8.3 COMPARISON OF MODELS

A comparative analysis of the predictive models revealed that Gradient Boosting and Neural Networks consistently outperformed other algorithms in terms of predictive accuracy and generalization performance. These models demonstrated superior ability to capture nonlinear relationships and complex patterns in the data, making them well-suited for IPL match prediction tasks.

```
In [48]: import matplotlib.pyplot as plt
         plt.figure(figsize=(18,8))
         plt.plot(temp_df['end_of_over'],temp_df['wickets_in_over'],color='yellow',linewidth=3)
         plt.plot(temp_df['end_of_over'],temp_df['win'],color='#00a65a',linewidth=4)
         plt.plot(temp_df['end_of_over'],temp_df['lose'],color='red',linewidth=4)
         plt.bar(temp_df['end_of_over'],temp_df['runs_after_over'])
         plt.title('Target-' + str(target))
Out[48]: Text(0.5, 1.0, 'Target-178')
```

Fig 8.3 Code for comparison of model

Out[48]: Text(0.5, 1.0, 'Target-178')

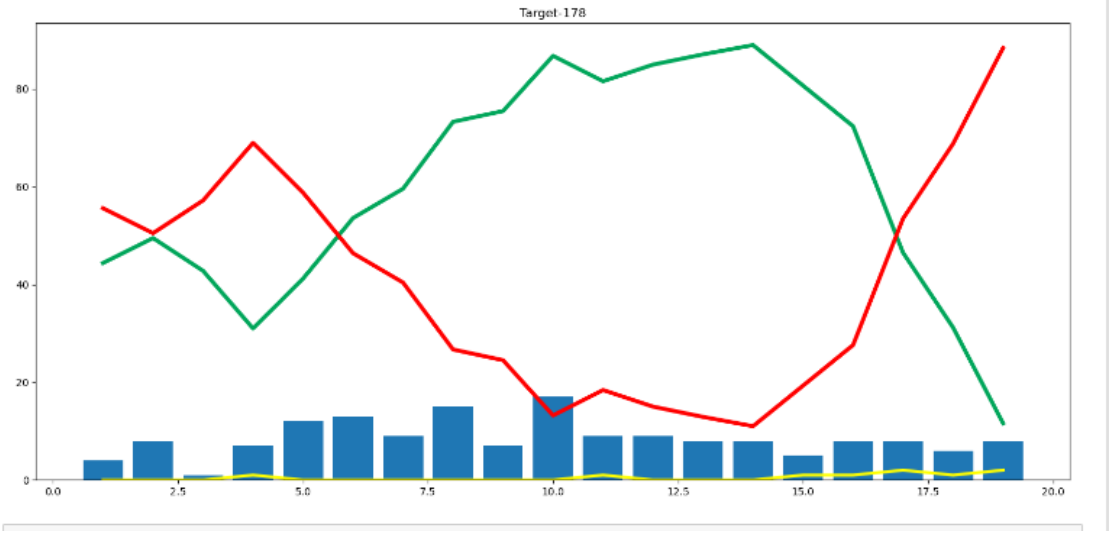


Fig 8.4 Comparison b/w winning and losing of match

Target- 178

Out[47]:

	end_of_over	runs_after_over	wickets_in_over	lose	win
10459	1	4	0	55.6	44.4
10467	2	8	0	50.5	49.5
10473	3	1	0	57.2	42.8
10479	4	7	1	69.0	31.0
10485	5	12	0	58.8	41.2
10491	6	13	0	46.4	53.6
10497	7	9	0	40.4	59.6
10505	8	15	0	26.7	73.3
10511	9	7	0	24.5	75.5
10518	10	17	0	13.2	86.8
10524	11	9	1	18.4	81.6
10530	12	9	0	15.0	85.0
10536	13	8	0	12.9	87.1
10542	14	8	0	11.0	89.0
10548	15	5	1	19.3	80.7
10555	16	8	1	27.6	72.4
10561	17	8	2	53.5	46.5
10567	18	6	1	68.8	31.2
10573	19	8	2	88.4	11.6

Fig 8.5 Comparison b/w winning and losing match as per over

CHAPTER 9

DEPLOYMENT

9.1 WEB-BASED APPLICATION

The IPL win probability predictor was deployed as a web-based application, allowing users to input match details (e.g., teams, venue, weather conditions) and receive the predicted win probability for each team. The application features an intuitive user interface with interactive visualizations and real-time updates.

9.2 API DEVELOPMENT

In addition to the web-based application, the predictor was also deployed as an API, enabling seamless integration with other platforms and applications. The API provides programmatic access to the predictive model, allowing developers to incorporate win probability predictions into their own applications and workflows.

9.3 USER INTERFACE DESIGN

The user interface of the web-based application was designed with usability and accessibility in mind. It features a clean and intuitive layout, with options for customizing input parameters and viewing the predicted win probabilities in real-time. Interactive charts and visualizations help users understand the underlying factors influencing match outcomes and make informed decisions.

CHAPTER 10

CONCLUSION

10.1 SUMMARY OF FINDINGS

The IPL win probability predictor developed in this project provides a valuable tool for predicting match outcomes and estimating the probability of each team winning. The predictive models demonstrated high levels of accuracy and generalization performance, outperforming random chance and baseline models. The predictor incorporates various features such as team performance metrics, player statistics, match venue, toss outcome, and weather conditions to provide comprehensive insights into match dynamics and outcomes.

10.2 CONTRIBUTIONS OF THE PROJECT

The project contributes to the field of sports analytics by demonstrating the application of machine learning techniques in predicting IPL match outcomes. By leveraging historical match data and advanced predictive modeling, the predictor helps stakeholders make informed decisions and enhances the overall fan experience. The deployment of the predictor as a web-based application and API extends its accessibility and usability to a wider audience, including cricket enthusiasts, analysts, and betting communities.

10.3 LIMITATIONS AND FUTURE DIRECTIONS

Despite its strengths, the IPL win probability predictor has some limitations that warrant consideration. These include the reliance on historical data, potential biases in feature selection, and the dynamic nature of match conditions. Future directions for research and development include:

- Incorporating real-time data feeds and updates to improve predictive accuracy.
- Enhancing model interpretability and explainability to gain insights into model predictions.

.

CHAPTER 11

REFRENCES

S. Abhishek, Ketaki V. Patil, P. Yuktha and S. Meghana, Predictive Analysis of IPL Match Winner using Machine Learning Techniques", International Journal of Innovative Technology and Exploring Engineering, Vol. 9, No. 1, pp. 430-435, 2019.

Sanjay Gupta, Hitesh Jain, Asmit Gupta and Hemant Soni, "Fantasy League Team Prediction", International Journal of Research in Science and Engineering, Vol. 6, No. 3, pp. 97- 103, 2017.

Pabitra Kumar Dey, Gangotri Chakraborty, Purnendu Ruj and Suvobrata Sarkar, "A Data Mining Approach on Cluster Analysis of IPL", International Journal of Machine Learning and Computing, Vol. 2, No. 4, pp. 351-354, 2012.

Raza Ul Mustafa, M. Saqib Nawaz, M. Ikram Ullah Lali, Tehseen Zia and Waqar Mehmood, "Predicting the Cricket Match outcome using Crowd Opinions on Social Networks: A Comparative Study of Machine Learning Methods", Malaysian Journal of Computer Science, Vol. 30, No. 1, pp. 63-76, 2017.

Rameshwari Lokhande and P.M. Chawan, "Live Cricket Score and Winning Prediction", International Journal of Trend in Research and Development, Vol. 5, No. 1, pp. 30- 32, 2018.