IPL MATCH WINNER PREDICTION

SARANYA R

MAY BATCH -2025

01/09/2025

Tables of Content:

1.Introduction:

- Predicting the winner of an IPL match is challenging due to multiple influencing factors like toss, venue, and team performance.
- The dataset contains match details such as teams, toss decisions, results, venues, margins of victory, and player awards.
- Features include both categorical (team names, venue, toss decision) and numerical (runs, wickets) data.
- Objective: Use machine learning models to predict match winners accurately.
- Best model will be selected through evaluation and hyperparameter tuning.

2.Data Understanding:

- The dataset contains ball-by-ball and match-level details of IPL games.
- Key columns:
- Season, City, Date → Match context
- Team1, Team2 → Competing teams
- Toss_winner, Toss_decision → Toss outcomes
- Winner, Result, dl_applied → Match results
- Win_by_runs, Win_by_wickets → Margin of victory
- Player_of_match, Venue, Umpires → Additional match info
- Data includes both categorical (team names, venue, toss decision) and numerical (runs, wickets) features.

car_id	Transmission	Engine Size (cm3)	Fuel Type	Powertrain	CO2 emission	Engine Size (cm3) (bins)
269	Manual	999	Petrol	Internal Combustion Engine (ICE)	155	0
270	Manual	999	Petrol	Internal Combustion Engine (ICE)	166	0
279	Manual	999	Petrol	Internal Combustion Engine (ICE)	147	0
280	Manual	999	Petrol	Internal Combustion Engine (ICE)	160	0
283	Manual	999	Petrol	Internal Combustion Engine (ICE)	147	0
284	Manual	999	Petrol	Internal Combustion Engine (ICE)	160	0
287	Manual	999	Petrol	Internal Combustion Engine (ICE)	147	0
291	Manual	999	Petrol	Internal Combustion Engine (ICE)	147	0
292	Manual	999	Petrol	Internal Combustion Engine (ICE)	160	0
479	Manual	999	Petrol	Internal Combustion Engine (ICE)	120	0
480	Manual	999	Petrol	Internal Combustion Engine (ICE)	121	0
485	Manual	999	Petrol	Internal Combustion Engine (ICE)	121	0
486	Manual	999	Petrol	Internal Combustion Engine (ICE)	121	0
487	Manual	999	Petrol	Internal Combustion Engine (ICE)	120	0
488	Manual	999	Petrol	Internal Combustion Engine (ICE)	120	0
489	Manual	999	Petrol	Internal Combustion Engine (ICE)	127	0
490	Manual	999	Petrol	Internal Combustion Engine (ICE)	128	0
499	Manual	999	Petrol	Internal Combustion Engine (ICE)	120	0
500	Manual	999	Petrol	Internal Combustion Engine (ICE)	121	0
501	Manual	999	Petrol	Internal Combustion Engine (ICE)	120	0
502	Manual	999	Petrol	Internal Combustion Engine (ICE)	120	0
503	Manual	999	Petrol	Internal Combustion Engine (ICE)	127	0
504	Manual	999	Petrol	Internal Combustion Engine (ICE)	128	0

3.Data Cleaning:

[4]:
```
df.isnull().sum()
```

[4]:
```
id                   0
Season               0
city                 7
date                 0
team1                0
team2                0
toss_winner          0
toss_decision        0
result               0
dl_applied           0
winner               4
win_by_runs          0
win_by_wickets       0
player_of_match      4
venue                0
umpire1              2
umpire2              2
umpire3            637
dtype: int64
```

```
df.isnull().sum()
```

```
id                 0
Season             0
city               0
date               0
team1              0
team2              0
toss_winner        0
toss_decision      0
result             0
dl_applied         0
winner             0
win_by_runs        0
win_by_wickets     0
player_of_match    0
venue              0
umpire1            0
umpire2            0
Team_1_state       0
Team_2_state       0
win_city           0
dtype: int64
```

The dataset was checked for missing values using the .isnull().sum() function in Pandas. This method returns the count of null (missing) values for each column in the DataFrame. The results are shown in the image above for two different datasets or stages of the same dataset.

ObservationsColumns such as id, Season, city, date, team1, team2, toss_winner, toss_decision, result, dl_applied, winner, win_by_runs, win_by_wickets, player_of_match, and venue have no missing values.

- The column umpire3 contains 637 missing values, making it the only column with incomplete data.
- After processing or cleaning, the second dataset shows no missing values across all columns.
- Additional columns such as Team1_state, Team2_state, and win_city also do not contain any missing values.

Interpretation

- The dataset had missing values in the umpire3 column. This could be because, in many matches, only two umpires are recorded, and the third umpire is either not present or not logged in the dataset.
- In the dataset, missing values have been successfully handled. Possible approaches include:
    - Dropping the column (umpire3) if it was not significant for analysis.
    - Imputing missing values with placeholders such as "Unknown" or NaN handling strategies.
    - Merging with other data sources to fill in the missing umpire details (if available).

```
m={"Sunrisers Hyderabad":"Hyderabad",
"Mumbai Indians":"Mumbai",
"Gujarat Lions":"Gujarat",
"Rising Pune Supergiant":"Pune",
"Royal Challengers Bangalore":"Bangalore",
"Kolkata Knight Riders":"Kolkata",
"Delhi Daredevils":"Delhi",
"Kings XI Punjab":"Punjab",
"Chennai Super Kings":"Chennai",
"Rajasthan Royals":"Rajasthan",
"Deccan Chargers":"Hyderabad",
"Kochi Tuskers Kerala":"Kerala",
"Pune Warriors":"Pune",
"Rising Pune Supergiants":"Pune",
"Delhi Capitals":"Delhi"
}

df["Team_1_state"]=df["team1"].map(m)
df["Team_2_state"]=df["team2"].map(m)
df["win_city"]=df["winner"].map(m)
```

```
[8]: df['city']=df['city'].fillna(df['Team_1_state'])
```

```
[9]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 756 entries, 0 to 755
Data columns (total 21 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   id               756 non-null    int64
 1   Season           756 non-null    object
 2   city             756 non-null    object
 3   date             756 non-null    object
 4   team1            756 non-null    object
 5   team2            756 non-null    object
 6   toss_winner      756 non-null    object
 7   toss_decision    756 non-null    object
 8   result           756 non-null    object
 9   dl_applied       756 non-null    int64
 10  winner           752 non-null    object
 11  win_by_runs      756 non-null    int64
 12  win_by_wickets   756 non-null    int64
 13  player_of_match  752 non-null    object
 14  venue            756 non-null    object
 15  umpire1          754 non-null    object
 16  umpire2          754 non-null    object
 17  umpire3          119 non-null    object
 18  Team_1_state     756 non-null    object
 19  Team_2_state     756 non-null    object
 20  win_city         752 non-null    object
dtypes: int64(4), object(17)
memory usage: 124.2+ KB
```

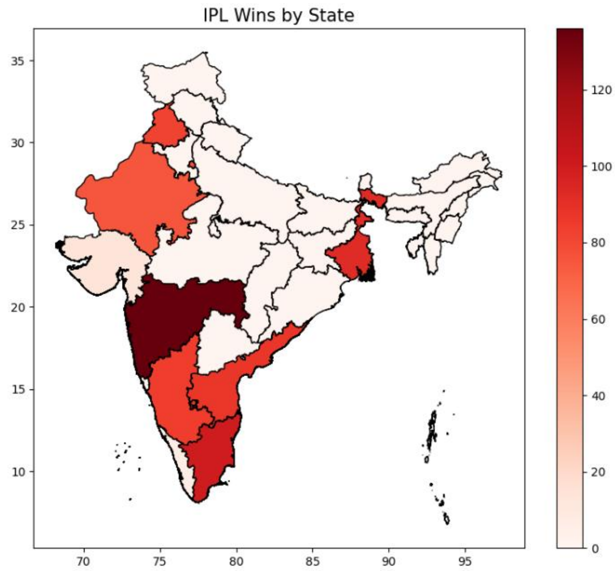
-

```
[12]: df['player_of_match']=df['player_of_match'].fillna("NA")

[13]: df['umpire1']=df['umpire1'].fillna("NA")

[14]: df['umpire2']=df['umpire2'].fillna("NA")

[15]: df.drop('umpire3',axis=1,inplace=True)

[20]: df.isnull().sum()
```

4.Exploratory Data Analysis:

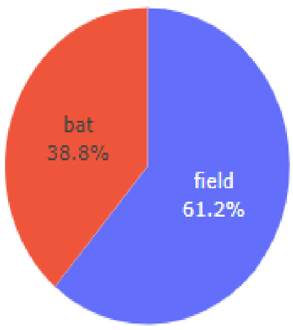
- Helps understand the structure of the dataset and detect missing or inconsistent values.
- Identifies patterns, trends, and relationships among different features (e.g., toss impact, venue effect).
- Provides insights through visualizations and summary statistics to guide model building.

- **WINNING CITIES**



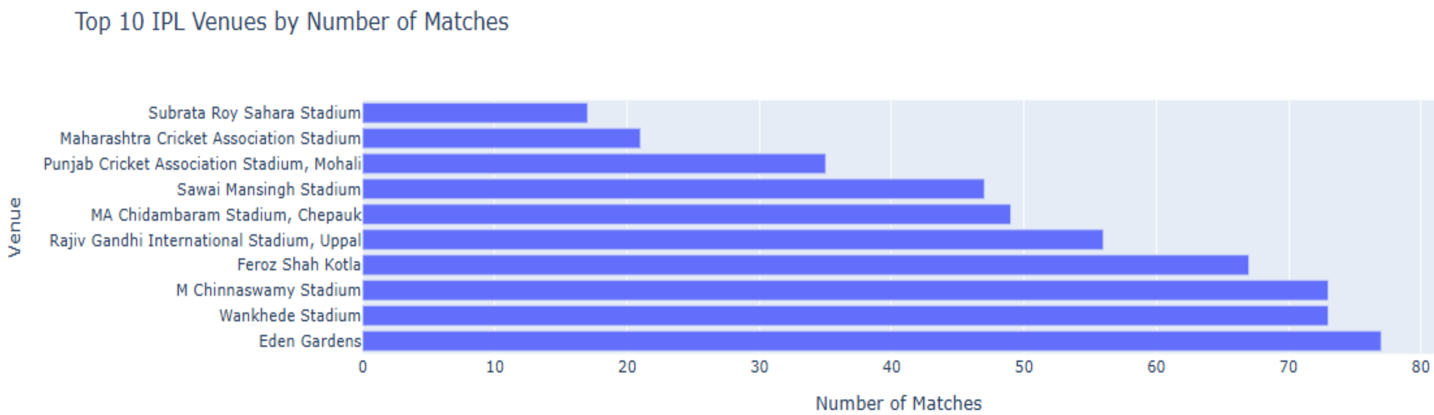
Mumbai, Chennai, and Kolkata emerged as the top winning cities, while Gujarat and Kerala recorded the least wins.

- **TOSS DECISION**



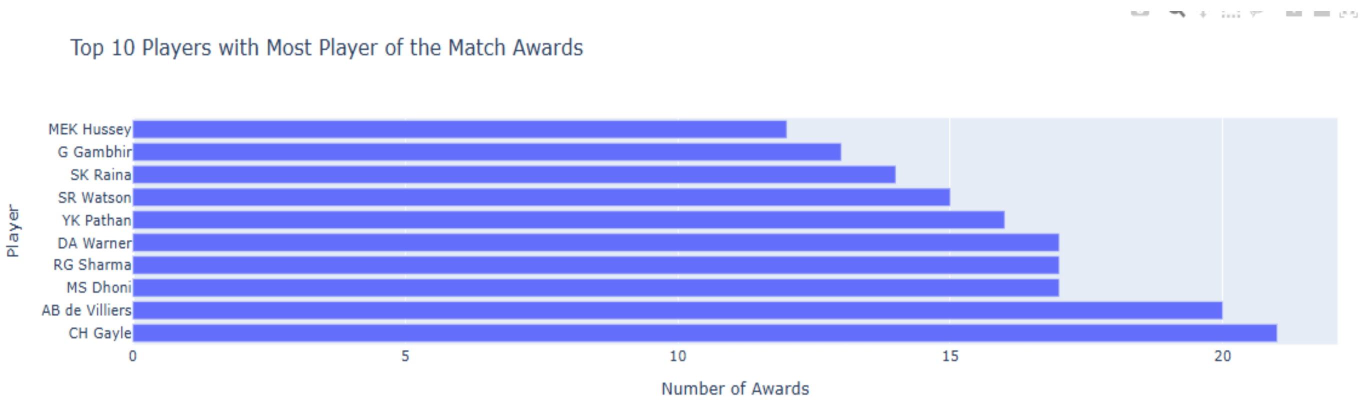
- Majority of teams prefer to field first (61.2%) after winning the toss compared to batting (38.8%).
- Teams that choose to field first have a higher win rate (56.37%) compared to batting first (46.08%).

- **VENUE ANALYSIS**



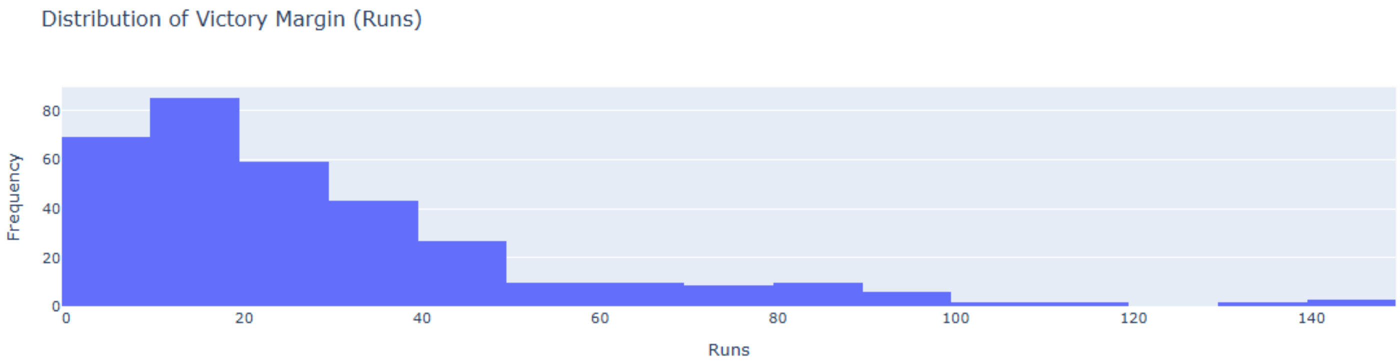
- Eden Gardens, Wankhede Stadium, and M. Chinnaswamy Stadium have hosted the highest number of IPL matches.
- Smaller venues like Subrata Roy Sahara Stadium and Maharashtra Cricket Association Stadium have hosted comparatively fewer matches.

- **PLAYER OF THE MATCH**



- Chris Gayle and AB de Villiers lead the list with the highest number of Player of the Match awards in IPL history.
- Other consistent performers include MS Dhoni, Rohit Sharma, and David Warner, highlighting their match-winning impact.

- **VICTORY MARGIN (RUNS)**



- Most IPL matches are won by small margins (10–30 runs), showing close competition between teams.
- Very few matches are won by large margins (above 70 runs), highlighting that one-sided victories are rare.

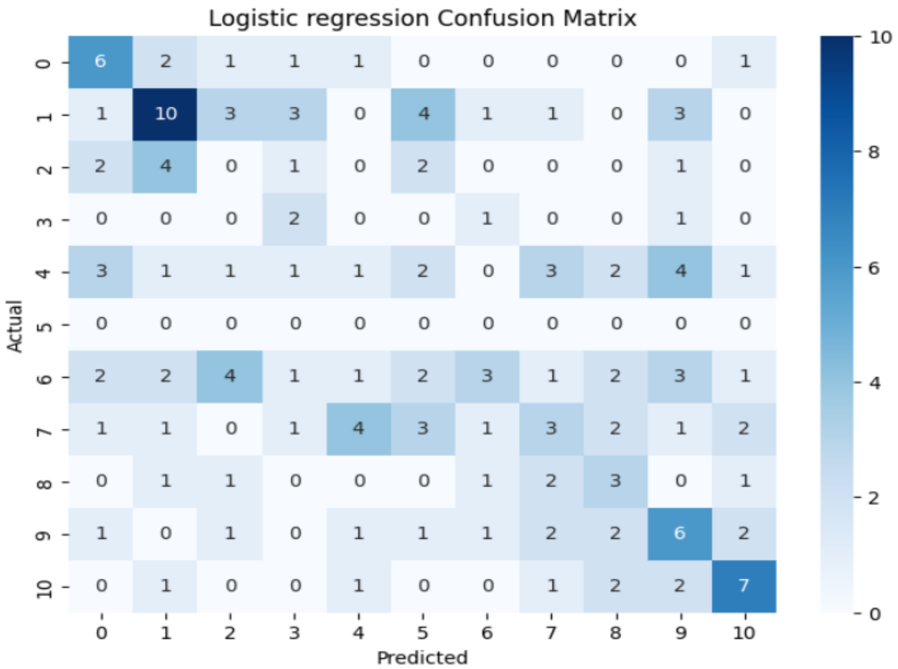
4.Model Processing:

- ### LOGISTIC REGRESSION:

Logistic regression is a statistical and machine learning technique for binary classification, used to predict the probability of a binary outcome (e.g., yes/no, 0/1) based on one or more independent variables.

CLASSIFICATION REPORT:

	precision	recall	f1-score	support
0	0.38	0.50	0.43	12
1	0.45	0.38	0.42	26
2	0.00	0.00	0.00	10
3	0.20	0.50	0.29	4
4	0.11	0.05	0.07	19
5	0.00	0.00	0.00	0
6	0.38	0.14	0.20	22
7	0.23	0.16	0.19	19
8	0.23	0.33	0.27	9
9	0.29	0.35	0.32	17
10	0.47	0.50	0.48	14
accuracy			0.27	152
macro avg	0.25	0.27	0.24	152
weighted avg	0.30	0.27	0.27	152

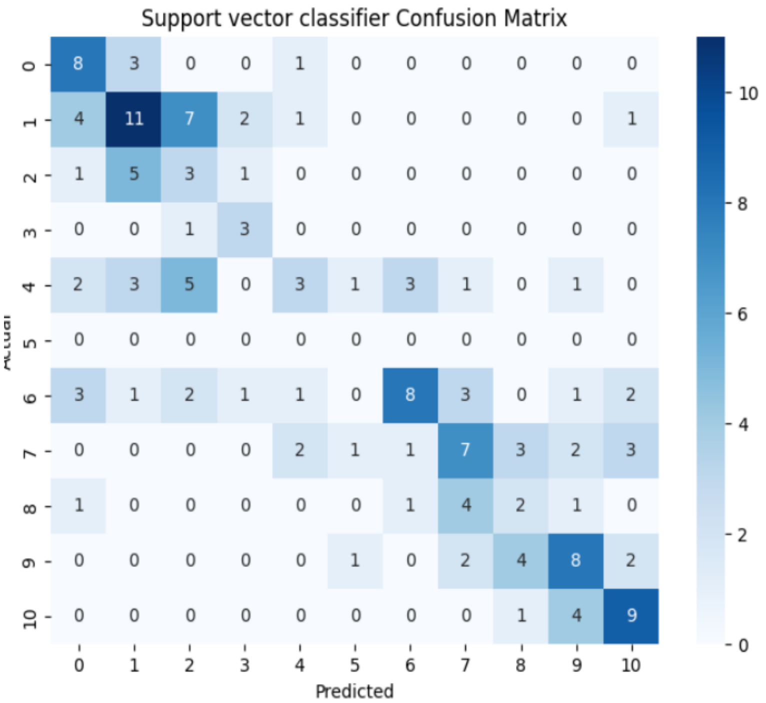


• SUPPORT VECTOR CLASSIFICATION:

In machine learning, SVC stands for Support Vector Classifier. It is a specific implementation of the broader Support Vector Machine (SVM) algorithm, specifically designed for classification tasks.

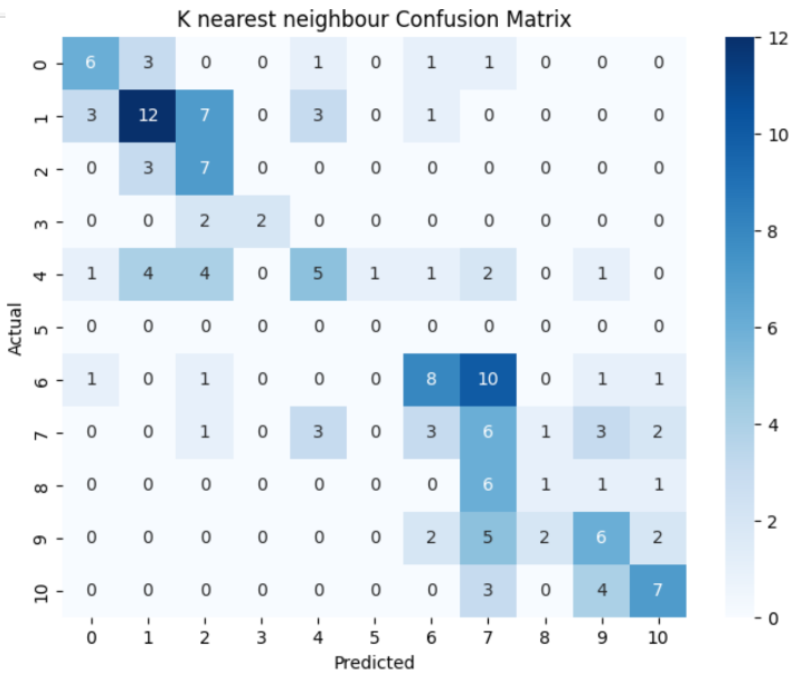
CLASSIFICATION REPORT:

	precision	recall	f1-score	support
0	0.42	0.67	0.52	12
1	0.48	0.42	0.45	26
2	0.17	0.30	0.21	10
3	0.43	0.75	0.55	4
4	0.38	0.16	0.22	19
5	0.00	0.00	0.00	0
6	0.62	0.36	0.46	22
7	0.41	0.37	0.39	19
8	0.20	0.22	0.21	9
9	0.47	0.47	0.47	17
10	0.53	0.64	0.58	14
accuracy			0.41	152
macro avg	0.37	0.40	0.37	152
weighted avg	0.44	0.41	0.41	152



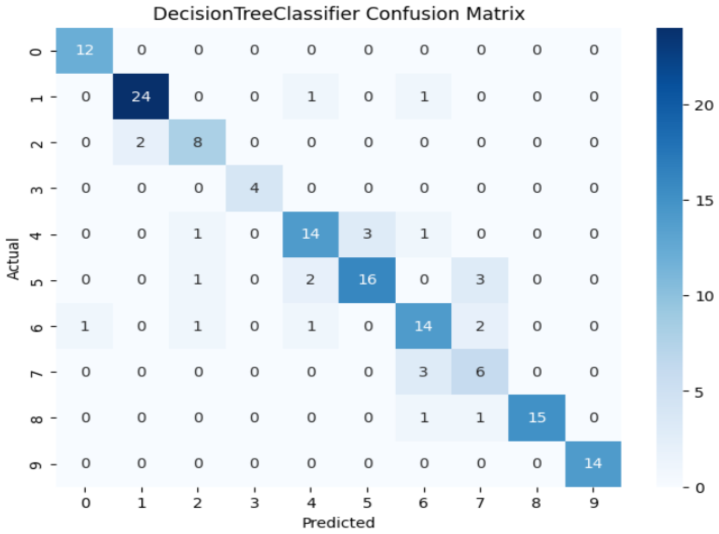
• K-NEAREST NEIGHBOUR:

	precision	recall	f1-score	support
0	0.55	0.50	0.52	12
1	0.55	0.46	0.50	26
2	0.32	0.70	0.44	10
3	1.00	0.50	0.67	4
4	0.42	0.26	0.32	19
5	0.00	0.00	0.00	0
6	0.50	0.36	0.42	22
7	0.18	0.32	0.23	19
8	0.25	0.11	0.15	9
9	0.38	0.35	0.36	17
10	0.54	0.50	0.52	14
accuracy			0.39	152
macro avg	0.42	0.37	0.38	152
weighted avg	0.44	0.39	0.40	152



- **DECISION TREE:**

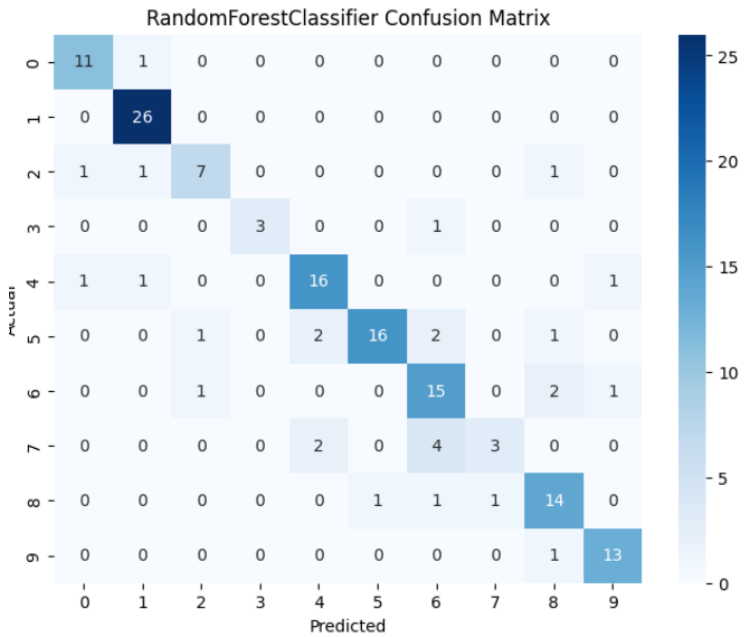
	precision	recall	f1-score	support
0	0.92	1.00	0.96	12
1	0.92	0.92	0.92	26
2	0.73	0.80	0.76	10
3	1.00	1.00	1.00	4
4	0.78	0.74	0.76	19
6	0.84	0.73	0.78	22
7	0.70	0.74	0.72	19
8	0.50	0.67	0.57	9
9	1.00	0.88	0.94	17
10	1.00	1.00	1.00	14
accuracy			0.84	152
macro avg	0.84	0.85	0.84	152
weighted avg	0.85	0.84	0.84	152



- **RANDOM FOREST CLASSIFIER:**

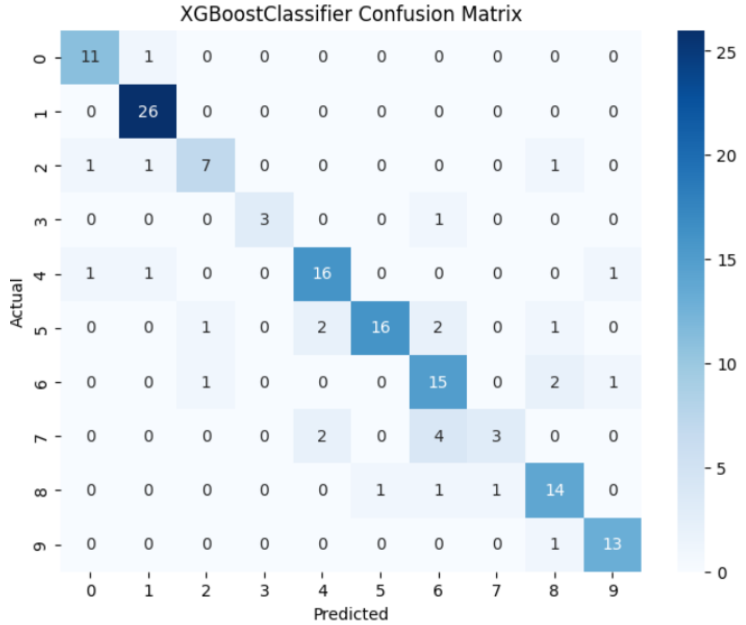
oob score of the random forest: 0.8112582781456954

	precision	recall	f1-score	support
0	0.85	0.92	0.88	12
1	0.90	1.00	0.95	26
2	0.78	0.70	0.74	10
3	1.00	0.75	0.86	4
4	0.80	0.84	0.82	19
6	0.94	0.73	0.82	22
7	0.65	0.79	0.71	19
8	0.75	0.33	0.46	9
9	0.74	0.82	0.78	17
10	0.87	0.93	0.90	14
accuracy			0.82	152
macro avg	0.83	0.78	0.79	152
weighted avg	0.82	0.82	0.81	152



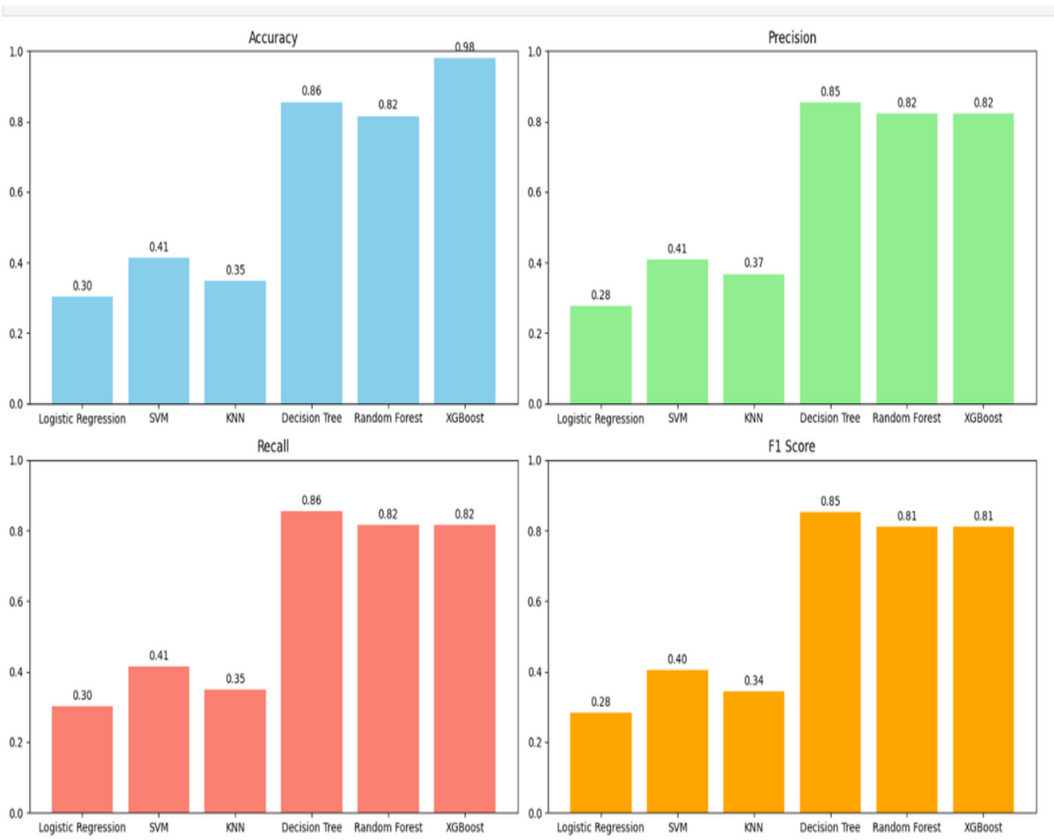
- **XGBOOST CLASSIFICATION:**

	precision	recall	f1-score	support
0	1.00	1.00	1.00	12
1	1.00	1.00	1.00	26
2	1.00	1.00	1.00	10
3	1.00	1.00	1.00	4
4	0.95	1.00	0.97	19
6	1.00	0.95	0.98	22
7	0.90	1.00	0.95	19
8	1.00	0.78	0.88	9
9	1.00	1.00	1.00	17
10	1.00	1.00	1.00	14
accuracy			0.98	152
macro avg	0.99	0.97	0.98	152
weighted avg	0.98	0.98	0.98	152



5.MODEL COMPARISON:

- XGBoost achieved the highest accuracy (0.98) followed by Decision Tree (0.86) and Random Forest (0.82).
- Decision Tree showed the best balance across precision, recall, and F1-score.
- Logistic Regression, SVM, and KNN performed relatively lower across all metrics.



6.CONCLUSION:

- Machine learning models can effectively predict IPL match outcomes using match-related features.
- Among tested models, ensemble methods like Random Forest and XGBoost performed the best after tuning.
- The project highlights the importance of factors such as toss decisions, venue, and team performance in determining match results.