**Project Documentation:**

**ML Reinforcement Project**

**IPL Dataset**

Name: Pavithra Muthumayan

Course: DA&DS

**Table of Contents**

1. Introduction
2. Objective
3. Dataset Description
4. Data Preprocessing
5. Exploratory Data Analysis (EDA)
6. Model Building
7. Hyperparameter Tuning
8. Model Evaluation and Comparison
9. Conclusion

**Introduction**

This project focuses on predicting the winner of an IPL (Indian Premier League) match using machine learning techniques. The IPL is a professional Twenty20 cricket league in India, and predicting match outcomes can provide significant insights for fans, teams, and analysts.

**Objective**

The aim of this project is to predict the winner of an IPL match based on various features such as the teams playing, toss winner, toss decision, and other match-related factors using different machine learning models like Logistic Regression, SVM, KNN, Decision Trees, Random Forest, and XGBoost. The best model must be selected after evaluating the performance and hyperparameters must be tuned for further improvements.

**Dataset Description**

The IPL dataset contains detailed information about cricket matches played in the Indian Premier League (IPL). Key columns include:

• Season: The year of the IPL season.

• City: The city where the match was played.

• Date: The date of the match.

• team1 and team2: The two teams playing in the match.

• toss\_winner and toss\_decision: The team that won the toss and the decision they made (whether to bat or bowl).

• result: The outcome of the match (win or loss).

• dl\_applied: Indicates if the Duckworth-Lewis method was applied due to interruptions.

• winner: The team that won the match.

• win\_by\_runs and win\_by\_wickets: The margin of victory by runs or wickets.

• player\_of\_match: The player awarded for exceptional performance.

• venue: The stadium where the match was held.

• umpires: The officials overseeing the match.

**Data Preprocessing**

Steps taken:

* Dropped unnecessary columns (e.g., umpire details, player of the match).
* Handled missing values.
* Encoded categorical columns using LabelEncoder and get\_dummies.
* Ensured the dataset only included completed matches (no null values in 'winner').

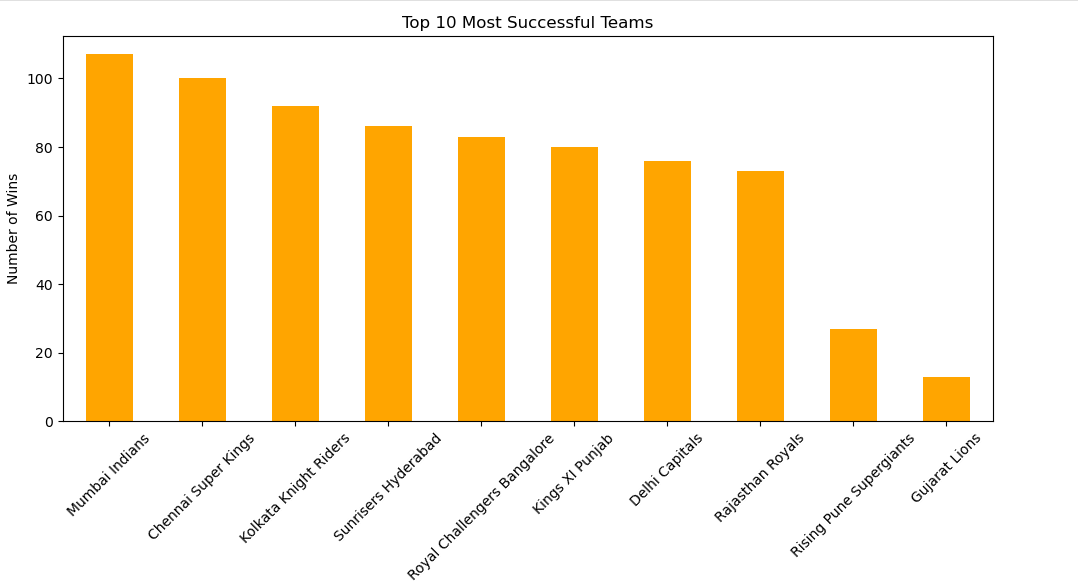
Split:

* Features: One-hot encoded columns for team and city.
* Target: Match winner (multiclass classification).
* Train-Test Split: 80-20 using train\_test\_split.

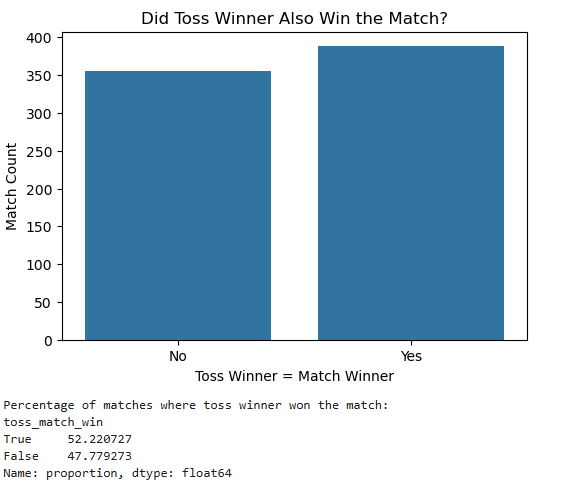
**Exploratory Data Analysis (EDA)**

Visualizations included:

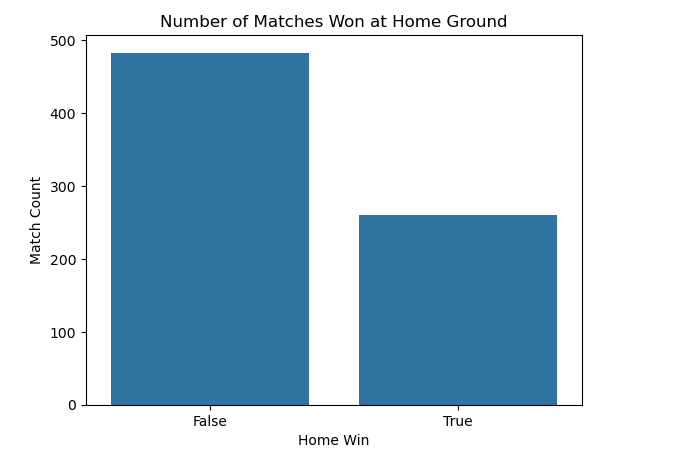
* Distribution of match winners



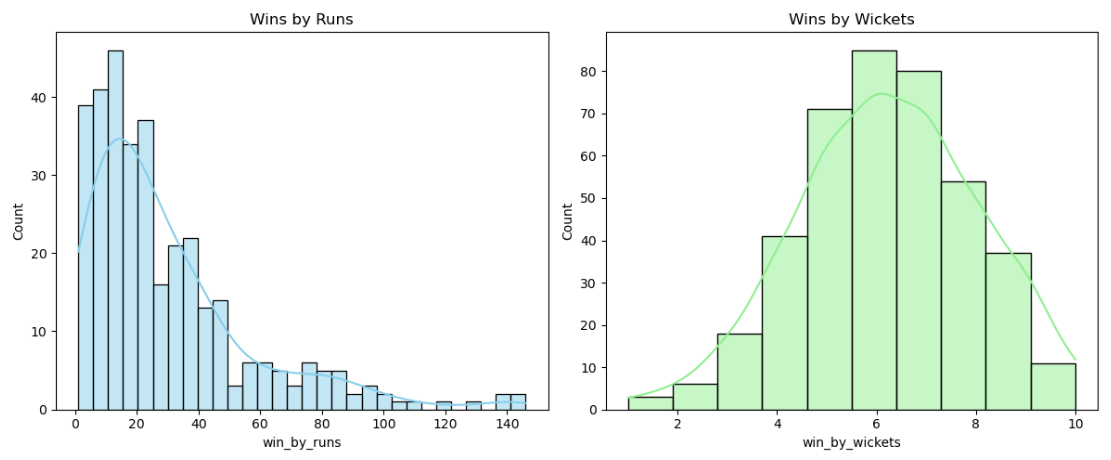
* Toss impact on winning



* Home advantage effects



* Win by runs and Win by Wickets



Insights:

* Toss winning has a slight correlation with match winning.
* Certain teams have consistently high win rates.
* Features like win\_by\_runs and win\_by\_wickets are highly correlated with the match outcome but are outcome-based.

**Model Building**

Models Implemented:

* Logistic Regression
* Support Vector Machine (SVM)
* K-Nearest Neighbors (KNN)
* Decision Tree
* Random Forest (with OOB score)
* XGBoost (with hyperparameter tuning)

**Hyperparameter Tuning**

* Used GridSearchCV for KNN, Decision Tree, Random Forest, and XGBoost.
* Evaluated combinations of:
* n\_neighbors, max\_depth, min\_samples\_split, learning\_rate, n\_estimators, gamma, colsample\_bytree, etc.
* Used Elbow method for KNN optimization.

**Evaluation Metrics:**

* Accuracy
* Precision, Recall, F1-score
* Confusion Matrix

**Model Evaluation and Comparison**

|  |  |
| --- | --- |
| Model | Accuracy |
| Logistic Regression | 61% |
| SVC | 63% |
| KNN | 66% |
| Decision Tree | 92% |
| Random Forest | 96% |
| XGBoost | 98% |

* Traditional models like Logistic Regression, SVC, and KNN underperform, with accuracies below 70%.
* Decision Tree shows dramatic improvement due to its ability to model complex patterns, but suffers from potential overfitting.
* Random Forest enhances performance by averaging over many trees, achieving 96% accuracy.
* XGBoost, with its boosting technique and fine-tuned hyperparameters, delivers the highest accuracy of 98%, making it the most robust and reliable model for this classification task.

Use XGBoost for deployment or future prediction tasks, while considering the inclusion of more match-level or player-specific features to maintain performance when new data is introduced.

**Conclusion**

This project aimed to predict IPL match winners using machine learning techniques, and through systematic data preprocessing, feature engineering, and model evaluation, it achieved remarkable results. After testing various models, XGBoost emerged as the most accurate, delivering a 98% accuracy rate by effectively capturing complex patterns in the data. The analysis revealed that factors like toss results, home ground advantage, and city played a crucial role in match outcomes. While the model performed exceptionally well, its accuracy could be further enhanced by incorporating player-level statistics and real-time data. Overall, this project demonstrates the powerful role machine learning can play in sports analytics, offering valuable insights into outcome prediction.