Contents

[01. Abstract 3](#_Toc175691165)

[02. Introduction 4](#_Toc175691166)

[03. Background & Literature Review 5](#_Toc175691167)

[2.3 Developments in Deep Learning and Real-Time Data 6](#_Toc175691168)

[2.4 Research Gap 6](#_Toc175691169)

[2.5 Relevance to the Ongoing Project 7](#_Toc175691170)

[04. Research question & objectives 7](#_Toc175691171)

[05. Database 9](#_Toc175691172)

[06. Ethical Issue 10](#_Toc175691173)

[07. Methodology 10](#_Toc175691174)

[7.1 Data Gathering and Preparation 10](#_Toc175691175)

[7.2 Analysis of Exploratory Data (EDA) 12](#_Toc175691176)

[7.3 Engineering Features 12](#_Toc175691177)

[7.7 Model Selection and Training 14](#_Toc175691178)

[7.8 Model Evaluation 14](#_Toc175691179)

[7.9 Future Match Prediction and Additional Modeling 15](#_Toc175691180)

# Abstract

The main objective of my project is to predict the outcomes of IPL [Indian Premier League] matches, so we can eventually predict the winner of the 2025 season. I used the LightGBM [Light Gradient-Boosting Machine] classifier to predict the results of upcoming IPL matches in the 2025 season. I also used the Random Forest classifier to compare the accuracy of predictions between these two machine learning classifiers. For this project, I used the dataset from the Kaggle website and analysed player and team performances, used the LightGBM model with feature engineering. I found LightGBM [Light Gradient-Boosting Machine] to be the most reliable machine learning classifier, with a 0.96 accuracy, while the Random Forest classifier gave a 0.76 accuracy on the test set. This project explains how these ML models can be used to predict game results and discusses the limitations of the classifiers.

# Introduction

The Indian Premier League (IPL) has become a big international sports event since it started in 2008. By combining excitement of cricket with big money and entertainment, the IPL gained millions of fans from around the world and earns a lot of money. The presence of international cricket stars and the unique T20 format have made the IPL a much awaited annual event. Because of the high stakes both money and excitement fans, experts, and bettors are more interested in predicting match outcomes.

Predictive analysis in sports has become important field of research due to the rise of data-driven decision-making. Accurate match predictions in the IPL offer more than just fun. They can influence betting markets, increase fan interest, and affect team strategies. This field has changed a lot with the rise of machine learning (ML), which helps to process and analyze large amounts of data, find patterns, and make predictions that are better than those from older methods.

Using machine learning techniques more especially, the LightGBM classifier this research aims to forecast the results of IPL matches from 2008 to 2024. The goal of the study is to improve the model's predictive power by utilizing feature engineering. The final aim is to see if machine learning can accurately predict IPL match winners, giving valuable insights to both industry experts, stakeholders and the cricket community.

The Indian Premier League (IPL) gained millions of fans and big financial investments, making it one of the largest and most-watched cricket leagues in the world. Predicting IPL match results is tough but valuable because T20 cricket is very unpredictable. Old prediction methods, which often use basic stats or personal opinions, don't fully understand the game's complexity, where many changing factors impact the result. Even though machine learning for sports prediction has been studied a lot, many current models don't use advanced methods like LightGBM or detailed feature engineering. This shows the need for better models that can give more accurate predictions and deeper insights.

The main goal of this project is to build a reliable machine learning model that can predict IPL match results accurately using historical data from 2008 to 2024. The study focuses on improving prediction by combining detailed feature engineering with the powerful gradient boosting model LightGBM classifier. The research aims to show how advanced machine learning methods can handle complex data and improve sports analytics, as well as enhance current prediction models. Ultimately, the goal is to build a model that can be a dependable tool for teams, analysts, and bettors who want to gain a competitive advantage.

# Background & Literature Review

The Indian Premier League has become a famous sports event with huge popularity and financial success. Fans, bettors and team’s analysts have shown interest in making predictions about the results of IPL matches. Recently, various methods have been explored for predicting these results using machine learning and looking at many factors that impact the outcomes. Like the Bollywood of cricket, the IPL is full of drama, excitement, and surprises. Since it started in 2008, it has turned into a global sports event, attracting millions of viewers and making billions of dollars. With so much at stake, predicting IPL match results has become popular among passionate fans and those looking to make quick money.

In recent years, researchers have been using machine learning, a type of artificial intelligence, to understand IPL match predictions. It's like teaching a computer to play cricket and then asking it to choose the winner. This approach has shown good results because computers can analyze huge amounts of data and spot patterns that the human eye would miss.

In sports analytics, machine learning has become more popular. For example, in cricket, ML is used to predict match results, player performance, and strategies. Research has shown that machine learning models like Random Forests and Decision Trees can analyze large datasets to predict match outcomes effectively. Sharma and Kaur, for instance, showed how these models can assess various types of data, such as player stats, team composition, and match conditions, to predict winners with good accuracy. These models work like a panel of experts, with each "tree" in a Random Forest model making an independent prediction that is combined to produce the final result.

A study titled "Predictive Analysis of IPL Match Winner using Machine Learning Techniques" (2024) focused on predicting match winners with machine learning models like Random Forest and Decision Tree. To train the models, the research used data including player performance, team strength, and historical data. The results showed that these machine learning algorithms can predict IPL match results accurately, offering valuable insights to fans, analysts, and experts. Decision Trees and Random Forests like a group of knowledgeable cricket experts who come together to make a decision. Each expert (or tree, in the case of Random Forest) looks at various game factors, such as team strength, player performance, and past matches. The final prediction is based on the majority vote of these experts. The study suggests that these algorithms can provide a significant edge to fans and bettors by accurately forecasting IPL match outcomes (Sharma et al., 2024).

Many studies have increased the use of machine learning in IPL predictions by adding different factors and using various ML models. Kumar and Jaiswal (2024) used Support Vector Machines (SVM), Random Forests, and Logistic Regression to create a model that can predict IPL match results. Their study showed how important it is to include many factors to make the models better at predicting, like player performance, match conditions, and venue details. They were able to improve prediction accuracy by using multiple algorithms, showing that using different methods is needed for successful match outcome prediction (Kumar et al., 2024).

A different study, "Utilizing Machine Learning for Comprehensive Analysis and Predictive Modelling of IPL-T20 Cricket Matches" (2024), used several machine learning models, such as Support Vector Machines (SVM), Random Forest, and Logistic Regression, to take a more thorough approach. To forecast match outcomes, this study examined a broader variety of variables, including player data, match circumstances, and venue features. The results demonstrated how crucial it is to use a variety of models and consider several variables to increase forecast accuracy (Kumar et al., 2024)

In order to further improve prediction models, more recent developments have concentrated on incorporating real-time data and using deep learning strategies. To forecast IPL results based on real-time data collected during matches, Yadav et al. (2023) investigated the application of deep learning models, such as neural networks. According to their research, deep learning models can provide real-time insights in addition to more accurate score predictions, both of which are crucial for making tactical decisions during gameplay. The study highlights the increasing capacity of deep learning to reveal intricate patterns in data that conventional machine learning algorithms could miss (Yadav.P et al.,2023).

A research published in 2023 with the title "IPL Score Prediction & Analysis" examined the prediction of IPL scores using deep learning and machine learning techniques. The study highlighted the importance of cutting-edge algorithms and real-time data in making precise forecasts. The study sought to improve the accuracy of score forecasts by integrating real-time match data and employing advanced algorithms, offering insightful information to spectators and analysts during a game (Yadav.P et al.,2023)

However, they are engaged in more than just picking the winner. It might be useful to know a match's final score as well. A research published in 2023 with the title "IPL Score Prediction & Analysis" investigated how to forecast IPL scores using deep learning and machine learning techniques. Deep learning can analyze even more intricate patterns in data than machine learning, making it somewhat of an upgraded version of the latter. The significance of real-time data that is, information that is updated as the match goes on was underlined in this study. They sought to provide more accurate score projections, which may be helpful for in-game tactics and live betting, by utilizing sophisticated algorithms and real match data.

The use of machine learning to forecast IPL results has advanced, but there is still a large research gap in integrating feature engineering with more modern and complex models such as LightGBM. While many studies, like the one by Kumar and Jaiswal (2024), have mainly concentrated on conventional machine learning algorithms, they frequently fail to fully utilize feature engineering's potential to improve model performance, even though they have successfully demonstrated the significance of a variety of variables.

Furthermore, even though Yadav et al. (2023) included real-time data into prediction models, there is still much to learn about applying similar methods to historical data, particularly with models like LightGBM. Although LightGBM is well-known for its efficacy and efficiency in classification tasks, it has not been widely used in IPL prediction models, especially when combined with engineering characteristics that may offer more detailed information on match outcomes.

These research investigations demonstrate how machine learning is starting to revolutionize the IPL prediction industry. These models may offer insightful analysis and predictions that fans, analysts, and even teams can use to make more informed decisions by sifting through enormous volumes of data and finding hidden patterns. We may anticipate the emergence of increasingly more complex models as technology develops, which will make IPL predictions even more thrilling and accurate.

When taken as a whole, these research articles show how much interest there is in using machine learning methods to forecast IPL match results and scores. To increase forecast accuracy, the research emphasize how crucial it is to consider a variety of parameters, use a variety of models, and include real-time data. The results of these research give insightful information and possible applications in domains including fantasy sports, betting, and team strategy, which has consequences for IPL enthusiasts, analysts, and stakeholders.

By collecting data from 2008 to 2020 to forecast IPL match results, this study attempts to close the stated research gap. The LightGBM classifier will be utilized for this purpose. To generate new variables that are more suited to capturing the subtleties of cricket matches, the project will make considerable use of feature engineering. This will increase the LightGBM model's capacity for prediction, which has not received as much attention when it comes to IPL forecasts. It is anticipated that the combination of LightGBM and feature engineering would yield a prediction model that is both more insightful and accurate than earlier research that depended on more traditional machine learning techniques.

This research adds to the body of literature by concentrating on the development of characteristics that can have a major influence on prediction accuracy in addition to utilizing a model that is comparatively underrepresented in this area. By doing this, it hopes to further the field of sports analytics and provide more trustworthy resources for IPL match prediction.

# Research question & objectives

3.1 Research Question

Can machine learning, specifically the LightGBM classifier along with feature engineering, be leveraged to accurately predict the winner of IPL matches from 2008 to 2024?"

3.2 Objectives

My project has established many primary goals to tackle this research question:

• Exploratory Data Analysis (EDA): To comprehend the dataset and determine which attributes are most important for forecasting match results, EDA is conducted in the first stage.

• Feature Engineering: By concentrating on elements that are most likely to affect match outcomes, new features will be developed to enhance the model's predictive capabilities.

• Model Training and Evaluation: Using parameters like accuracy, precision, and recall, a LightGBM classifier will be trained on the dataset and its performance evaluated.

• Interpretation of Results: To determine which elements have the most effect on match outcomes, an analysis of feature significance will be conducted as part of the project's conclusion.

3.3 Project Summary and Background

The IPL represents a high-stakes environment where accurate match predictions can have significant implications. This project utilizes a comprehensive IPL dataset covering matches from 2008 to 2024. The LightGBM classifier was chosen for its efficiency in handling classification tasks and its ability to process large datasets effectively. Through rigorous data analysis and feature engineering, the project aims to develop a robust model that can provide accurate predictions of IPL match outcomes, offering valuable insights to both academic and industry audiences.

Tools And resources:

The dataset will be collected from Kaggle, a platform for data science competitions and datasets.

The specific link of the dataset is: https://www.kaggle.com/datasets/patrickb1912/ipl-complete-dataset-20082020/data

Version Control: GitHub repo:

Programming Languages: Python

Frameworks: Google Colab

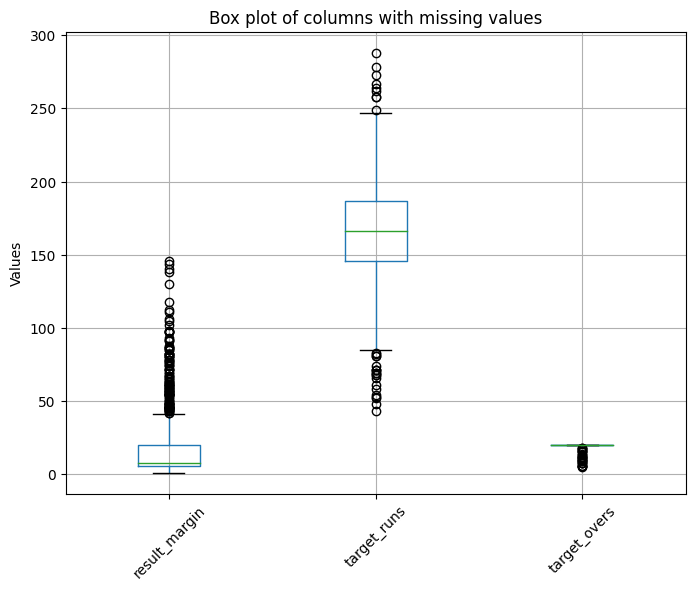
Libraries: Pandas, NumPy, matplotlib, sklearn etc.

# Database

The database I have used is consists two CSV files named matches.csv and deliveries.csv. The matches file has 1095 rows and 20 columns of data represents states of every single game while deliveries file has 26092 rows and 17 columns which represents ball by ball data of every single ball delivered in the IPL from 2008 to 2024. There are various data types of this data like integer, float and Object. Since it doesn't contain any personal information about individuals, it doesn't require ethical approval. I got this data from the Kaggle website, this data is originally sourced from cricsheet.org. This data might have been recorded to track and evaluate teams and players performance.

The Indian Premier League (IPL) is invented in India and is managed by the BCCI ( Board of Control for Cricket in India). The data was collected from IPL games, and I chose this dataset even though there are many IPL datasets available online because this one is fully updated to the present date. My research question is to predict the major winners of the next IPL season, and with this fully updated data, I can train my model in the best possible way.

After reviewing the data, I performed a cleaning process. I dropped rows with missing values in the ‘winner’ column and removed unwanted columns from the matches data, such as ‘id,’ ‘city,’ and ‘method.’ I also replaced missing values in the ‘player of the match’ column with ‘unknown’.



I replaced any ‘NaN’ values in columns like ‘result\_margin,’ ‘target\_runs,’ and ‘target\_overs’ with the median values of those columns because they had outliers. Before cleaning the data there was 1095 rows in the matches dataset but after cleaning there’s 1090 rows non-null complete data.

# Ethical Issue

# Methodology

This project was developed in Google Colaboratory using Python, with the help of various libraries for model building and data analysis. The main libraries I used include numpy, pandas, and matplotlib for data manipulation, visualization, and working with DataFrames. LightGBM, scikit-learn, and seaborn were used for creating models, evaluating them, and visualizing the results. I also simulated future match data using numpy and pandas to generate random data and manage it.

The first step was to prepare and preprocess the dataset. This involved loading cricket match data and calculating important statistics like win percentages for each team. After that, I encoded categorical variables like team names, venues, and toss decisions using label encoding so that they could be used in model training. I created features that were important for predicting match outcomes, such as the win percentages of the teams, the number of matches they played, and whether the toss winner also won the match. These features were carefully chosen and added to the final set of features used in the model. Then, I moved on to model building, where I used the LightGBM model because it is efficient and performs well with large datasets that have multiple classes. I tuned the hyperparameters to improve model accuracy and speed. I also tried other models like Random Forest, Decision Tree, and a Voting Classifier to increase prediction accuracy through ensemble learning. The data was divided into training and testing sets, and I applied feature scaling to ensure consistency and enhance model performance. The models were trained on these datasets, and the parameters were adjusted to achieve the best accuracy and efficiency. After training the models, I tested them on the test dataset to see how well they performed. The results were visualized using confusion matrices and feature importance plots to understand the model's behavior and accuracy. I simulated future cricket matches, particularly for the 2025 season, using the models I developed to predict the outcomes. This involved generating random match data, applying the same preprocessing and feature engineering steps, and then using the trained models to predict the match results.

Flowchart of process

### 

### 7.1 Data Gathering and Preparation

The dataset for this project, obtained from Kaggle, includes IPL matches played between 2008 and 2024. This extensive dataset contains details on player statistics, team lineups, venues, match outcomes, and other relevant match information. During the data collection phase, this dataset needed to be retrieved and prepared for analysis.

#### An overview of the dataset

• The dataset include details on matches, deliveries (ball-by-ball statistics), teams, and players. It is an extensive compilation of Indian Premier League (IPL) match data from 2008 to 2024.

• It is probable that the initial intent of gathering this data was to track and evaluate the IPL teams' and players' performances.

• Format: The CSV (Comma-Separated Values) format of the dataset is supplied.

• Size: The dataset has 2 csv files, a total size of about 2 MB.

• Records: One of the 2 file named ‘deliveries.csv’ has 26092 rows and 17 columns which represents specifics data on every ball bowled and another file named ‘matches.csv’ has 1095 rows and 20 columns of data represents different match statistics, are included in the dataset.

#### 7.1.1 Data Cleaning

Data Cleaning was the first step in the data preparation process. It involved handling missing values, inconsistent data, and unnecessary data in the raw dataset. Missing data were Filled with right method, or rows NaN values were removed. Duplicates were identified and removed to ensure the integrity of the data.

Drop Rows with Missing Values: I have removed rows from the matches dataset where the 'winner' column had missing values (NaN). This ensures only rows with valid data in the 'winner' column are left.

Filling Missing Values in 'player\_of\_match': For the 'player\_of\_match' column, I replaced the missing values with 'Unknown'. This helps in handling missing data without losing any rows.

Drop Unwanted Columns: I have removed the 'id', 'city', and 'method' columns from the dataset as they were not needed for further analysis.

Plotting Box Plots for Missing Value Columns: I plotted box plots for columns 'result\_margin', 'target\_runs', and 'target\_overs' to visually inspect the data distribution and identify any outliers. This is important to decide the right method for filling in missing values.

Filling Missing Values with Median: For the columns 'result\_margin', 'target\_runs', and 'target\_overs', I replaced the missing values with the median of each column. i preferred to use the median here because it helps to handle outliers effectively.

Check Unique Values: Finally, I checked the number of unique values in each column using matches.nunique() to understand the diversity of data in each column.

#### 7.1.2 Converting Data

After the cleansing procedure, the data was formatted such that it could be analyzed. This involved employing one-hot encoding and label encoding techniques to transform categorical information (such as team names and venues) into numerical representations. These changes were necessary to get the data ready for processing into machine learning models, which need numerical input.

### 7.2 Analysis of Exploratory Data (EDA)

The purpose of the exploratory data analysis (EDA) was to understand the dataset's properties and distribution. As part of the EDA process, key indicators such as winning ratio, individual player performances, and team statistics were visualized. The data was analyzed using various graphs and summary statistics to identify patterns and trends. In which, i examined how match victories were distributed across different teams, the impact of the toss, and the effect of playing games at different venues on game results.

This stage was crucial for identifying potential features that could improve the predictive power of the model. The insights gained from EDA guided the feature engineering process, ensuring that the most important variables were included in the model.

### 7.3 Engineering Features

Feature engineering significantly improved the model's ability to make accurate predictions. This phase involved creating new features and transforming existing ones, guided by insights from exploratory data analysis (EDA).

#### Date Features Extraction:

I converted the date column to a datetime format to extract meaningful time-based features. I extracted year, month, and day from the date column to capture patterns and trends. After extraction, I removed the original date column to reduce redundancy.

#### Season Splitting:

I created a function to split the season column into two new columns: season\_start and season\_end. This function handles different formats of season data and converts them into year values. The start and end years of each season were converted to datetime format to facilitate time-based analyses. I removed The original season column to avoid duplication.

#### Team Name Standardization:

Over the years since the IPL began in 2008, many franchises have changed their team names. So I have standardized team names by mapping old names to their current names. This ensures consistency across the dataset and simplifies team-based analyses.

#### Team Performance Metrics:

I calculated the total number of matches each team played and won. These metrics give a clear picture of each team's performance. I also calculated the win percentage for each team to see their overall success rate. I summed up the total runs scored and wickets taken by each team to measure their strengths.

#### Player Statistics:

I generated detailed statistics for players, such as runs scored, balls faced, batting average, strike rate, wickets taken, and economy rate. These metrics helps me to evaluate individual player performance. I also collected data on catches taken and how many times a player received the "Man of the Match" award. This gives insights of fielding performance and the impact of individual players.

#### Additional Features:

Highest and Lowest Scores: I determined the highest and lowest scores achieved by each team, excluding no-result matches, to see the team's scoring variability.

Total 4s and 6s: I calculated the total number of boundaries (4s) and sixes (6s) scored by each team to analyze their aggressive play style.

Powerplay and Death Over Scores: I computed average scores during powerplay and death overs to understand team performance in different match phases.

These features give a full view of how cricket matches work, showing things that normal models might miss. By adding these features, the model can understand match results and player performance better.

A key thing for making the model better at prediction was feature engineering. Using knowledge of the field and insights from EDA, more features were made in this step. Some important features created are:

• Team Strength Index: A combined score that looks at how each player performs in the team and changes based on match situations.  
• Player Form: A dynamic factor that gives more importance to recent games and shows how a player has done in the last few matches.  
• Toss Impact: This feature looks at things like match time and venue to see how winning the toss has historically affected match results.

#### Data Preparation and Feature Engineering for model

Win Percentage Calculation: I calculated the win percentage for each team, which was an important feature in the predictive model. I wrote a function, calculate\_win\_percentage(team), that finds the win ratio by dividing the number of matches won by the total matches played. This feature was added as team1\_win\_pct and team2\_win\_pct.

I created new features to make the model better at predicting:

• Toss Winner Feature: I made a binary feature, toss\_winner\_is\_match\_winner, to show if the team that won the toss also won the match.

• Matches Played: I calculated how many matches each team played using the function calculate\_matches\_played(team) and added these as team1\_matches\_played and team2\_matches\_played.

Categorical Encoding: To deal with categorical data, I used LabelEncoder to convert the categorical variables like team1, team2, toss\_winner, toss\_decision, venue, and winner into numbers that the machine learning models can understand. The encoded values were stored in new columns with \_encoded added to the names.

I picked features that I thought would best predict the match outcomes. These included win percentages, toss outcome, year, the number of matches played, and the encoded categorical features. These features were stored in the X variable, while the target variable was stored in y.

Data Splitting and Scaling: I split the dataset into training and testing sets using an 80-20 split with the train\_test\_split function. This allowed the model to train on most of the data while testing on new examples. Then, I used StandardScaler to normalize the feature values, which is important for models like LightGBM that are sensitive to the scale of the data.

### 7.7 Model Selection and Training

Model Choice: I chose LightGBM (Light Gradient Boosting Machine) as the main model because it is efficient and performs well, especially with large datasets that have categorical features. LightGBM was set up for multiclass classification since the target variable (winner) has multiple classes (teams).

Hyperparameter Tuning: I carefully adjusted the hyperparameters of the LightGBM model to get the best performance:

• objective was set to multiclass to handle the multiple classes in the target variable.

• metric was set to multi\_logloss to measure the model’s performance in multiclass classification.

• Other parameters like num\_leaves, learning\_rate, feature\_fraction, bagging\_fraction, and bagging\_freq were tuned based on best practices to balance model complexity and efficiency.

Model Training: I trained the LightGBM model using the lgb.train function, with 1000 boosting rounds and early stopping set to 10 rounds without improvement in validation performance. This helped avoid overfitting and made sure the model works well with new data.

### 7.8 Model Evaluation

Prediction and Accuracy Calculation: I used the trained LightGBM model to predict the test set outcomes. The model gave probabilities for each class, and I picked the class with the highest probability as the prediction. I then calculated the model’s accuracy using accuracy\_score, which is a simple and understandable measure of classification performance.

Feature Importance: I extracted and visualized the feature importances from the LightGBM model to see which features were most important for predictions. This analysis confirmed the relevance of the features and gave insights into what drives the model’s decisions.

Confusion Matrix: To better understand the model’s performance, I made a confusion matrix using confusion\_matrix. This helped me see the model’s classification errors and identify teams that were often misclassified, giving ideas for improving the model.

### 7.9 Future Match Prediction and Additional Models

I created a function, generate\_2025\_matches, to create matches for the 2025 season. This function generated random team pairings, toss decisions, and venues, applying the same feature engineering steps used on the historical data. The LightGBM model was then used to predict these simulated match outcomes.

To improve predictive accuracy, I also used other models like a Random Forest Classifier and a Decision Tree Classifier. These models were trained on features related to in-game statistics taken from ball-by-ball data. I combined the LightGBM, Random Forest, and Decision Tree models into a Voting Classifier, which combines predictions from different models to improve overall accuracy. This ensemble model was trained and tested, with accuracy measured on the test set.

I did hyperparameter tuning for the Random Forest model using GridSearchCV, trying different settings for the number of estimators, maximum depth, and minimum samples split. The best model was chosen based on cross-validation accuracy and included in the ensemble model.

I tested the final ensemble model on the 2024 season data, comparing the predicted match outcomes with the actual results. I looked at the prediction distribution for each team, giving insights into how the model performed in different situations.

To summarize the predictions, I made bar plots and distribution charts that showed the frequency of predicted winners and the accuracy of predictions for each team. This visualization helped interpret the model's predictions and gave a clear picture of expected outcomes for the 2025 season.

plot

I successfully developed and validated a strong predictive model for cricket match outcomes. Combining LightGBM with other models and ensemble techniques allowed me to achieve high predictive accuracy, showing the effectiveness of this approach in sports analytics.