BANSILAL RAMNATH AGARWAL CHARITABLE TRUST’S

VISHWAKARMA INSTITUTE OF TECHNOLOGY

(An Autonomous Institute affiliated to Savitribai Phule Pune University)

Data Science Course Project Report on

**IPL WINNING TEAM PREDICTION**

**Submitted by**

|  |  |  |
| --- | --- | --- |
| **Roll No.** | **STUDENT NAME** | **PRN** |
| 44 | Om Vikas Hase | 12311203 |
| 28 | Vedant Gaikwad | 12310469 |
| 30 | Vidya Gandhale | 12311889 |
| 32 | Dhruv Gandhi | 12311331 |
| 33 | Aditya Garad | 12311599 |

*UNDER THE GUIDANCE OF*

**Prof. Gajanan Gambhire**

**DEPARTMENT OF MULTIDISCIPLINARY ENGINEERING**

**2024 - 2025**

**BANSILAL RAMNATH AGARWAL CHARITABLE TRUST’S**

**VISHWAKARMA INSTITUTE OF TECHNOLOGY**

**(An Autonomous Institute affiliated to Savitribai Phule Pune University)**

**PUNE – 411037**



**CERTIFICATE**

This is to certify that the Course Project titled **IPL WINNING TEAM PREDICTION**

submitted by **Group No. – 7 of COMPUTER SCIENCE Division B** is in partial fulfillment for the Data Science (MD2201) Course Assessment. This project report is a record of bonafide work carried out by above group under my guidance during the academic year 2024-25.

**Course Faculty**  **HOD, DOME**

**Prof. Gajanan Gambhire Prof. Dr. Shilpa Lambor**

**Place: VIT, Pune. Date: 18-11-202**

**TABLE OF CONTENTS**

* [SECTION I: INTRODUCTION 4](#_Toc183127278)
* [SECTION II: PROBLEM STATEMENT 5](#_Toc183127279)
* [SECTION III: LIERATURE REVIEW 6](#_Toc183127280)
* [SECTION IV: OBJECTIVES 10](#_Toc183127281)
* [SECTION V: DATA COLLECTION 12](#_Toc183127282)
  + [About the IPL Complete Dataset (2008-2024) 12](#_Toc183127283)
    - [Matches Dataset: 12](#_Toc183127284)
    - [Deliveries Dataset: 14](#_Toc183127285)
* [SECTION VI: DATA CLEANING AND PREPARATION 16](#_Toc183127286)
  + [Matches Dataset: 16](#_Toc183127287)
  + [Deliveries Dataset: 18](#_Toc183127288)
* [SECTION VII: DATA ANALYSIS AND VISUALIZING 23](#_Toc183127289)
  + [Plot 1: Runs by Man of the Match 23](#_Toc183127290)
  + [Plot 2: Wickets by Player of Match 24](#_Toc183127291)
  + [Plot 3: Man of the Match 25](#_Toc183127292)
  + [Plot 4: Batsman Strike Rate 26](#_Toc183127293)
  + [Plot 5: Catches by Players 27](#_Toc183127294)
  + [Plot 6: Bowling Economy 27](#_Toc183127295)
  + [Plot 7: Most Sixes 28](#_Toc183127296)
  + [Plot 8: Most Fours 29](#_Toc183127297)
  + [Plot 9: Most Dot Balls. 29](#_Toc183127298)
  + [Plot 10: Ratio of Out vs Not Out 30](#_Toc183127299)
  + [Plot 11: Top 20 Batsman 30](#_Toc183127300)
  + [Plot 12: Top 20 Bowlers 31](#_Toc183127301)
  + [Plot 13: Top 10 Teams 32](#_Toc183127302)
  + [Plot 14: Top 10 Worst Teams 32](#_Toc183127303)
  + [Plot 15: IPL Teams with IPL Cup 33](#_Toc183127304)
  + [Plot 16: Heatmap 34](#_Toc183127305)
* [SECTION VIII: ALGORITHMS 36](#_Toc183127306)
  + [Random Forest: 40](#_Toc183127307)
  + [Logistic Regression: 40](#_Toc183127308)
  + [Naïve Bayes: 41](#_Toc183127309)
  + [Decision Tree: 42](#_Toc183127310)
  + [AUC-ROC Curve: 42](#_Toc183127311)
* [SECTION IX: RESULT 44](#_Toc183127312)
  + [Random Forest: 45](#_Toc183127313)
  + [Logistic Regression: 46](#_Toc183127314)
  + [Naïve Bayes: 47](#_Toc183127315)
  + [Decision Tree: 48](#_Toc183127316)
  + [AUC-RUC Curve: 49](#_Toc183127317)
* [SECTION X: CONCLUSION 51](#_Toc183127318)
* [SECTION XI: RECOMMENDATION AND FUTURE SCOPE 53](#_Toc183127319)
* [SECTION XII: REFERENCES 54](#_Toc183127320)

**LIST OF FIGURES**

[**Figure 1 Matches Dataset with Number of Rows and Columns.** 13](#_Toc183127601)

[**Figure 2 Matches Dataset with Attributes**. 13](#_Toc183127602)

[**Figure 3 Matches Dataset with Attributes and Its Na Values.** 14](#_Toc183127603)

[**Figure 4 Matches Dataset with Distinct Values for Some Attributes**. 14](#_Toc183127604)

[**Figure 5 Deliveries Dataset with Number of Rows and Columns.** 15](#_Toc183127605)

[**Figure 6 Deliveries Dataset with Attributes.** 15](#_Toc183127606)

[**Figure 7 Deliveries Dataset with Attributes and Na Values**. 16](#_Toc183127607)

[**Figure 8 Matches Dataset after removal of ID, City, Methods.** 17](#_Toc183127608)

[**Figure 9 Matches Dataset after data cleaning of seasons**. 18](#_Toc183127609)

[**Figure 10 Matches Dataset After Changing Old Names to Current Names.** 19](#_Toc183127610)

[**Figure 11 Derived Batstatic Dataset from Deliveries Dataset.** 19](#_Toc183127611)

[**Figure 12 Derived Ballstatic dataset from Deliveries Dataset.** 20](#_Toc183127612)

[**Figure 13 Derived final\_players\_matches dataset from Deliveries Dataset**. 21](#_Toc183127613)

[**Figure 14 Derived catchDf dataset from Deliveries Dataset**. 22](#_Toc183127614)

[**Figure 15 final\_df dataset from merging derived Dataset**. 23](#_Toc183127615)

[**Figure 16 Number of Runs Scored by Player of Match.** 24](#_Toc183127616)

[**Figure 17 Number of Wickets by Player of Match.** 25](#_Toc183127617)

[**Figure 18 Player of Match by Matches Played.** 26](#_Toc183127618)

[**Figure 19 Batsman Strike Rate.** 27](#_Toc183127619)

[**Figure 20 Most Catches by players.** 28](#_Toc183127620)

[**Figure 21 Bowling Economy of Players**. 29](#_Toc183127621)

[**Figure 22 Most Sixes by Players**. 29](#_Toc183127622)

[**Figure 23 Most Fours by Players.** 30](#_Toc183127623)

[**Figure 24 Most Dot Balls by Batsman**. 30](#_Toc183127624)

[**Figure 25 Ratio of Out Vs Not Out of Players.** 31](#_Toc183127625)

[**Figure 26 Top 20 Batsman Based on Batting Average.** 32](#_Toc183127626)

[**Figure 27 Top 20 Bowlers Based on Wickets**. 32](#_Toc183127627)

[**Figure 28 Top 10 Best Performing Team**. 33](#_Toc183127628)

[**Figure 29 Top 10 Worst Performing Team.** 34](#_Toc183127629)

[**Figure 30 IPL Winning Teams**. 35](#_Toc183127630)

[**Figure 31 Heatmap Confusion Matrix.** 36](#_Toc183127631)

[**Figure 32 Dummy Variables.** 38](#_Toc183127632)

[**Figure 33 Data Splitting and Training & Testing Set.** 40](#_Toc183127633)

[**Figure 34 Random Forest Model**. 41](#_Toc183127634)

[**Figure 35 Logistic Regression Model.** 42](#_Toc183127635)

[**Figure 36 Naïve Bayes Model.** 42](#_Toc183127636)

[**Figure 37 Decision Tree Model.** 43](#_Toc183127637)

[**Figure 38 AUC-ROC Curve.** 44](#_Toc183127638)

[**Figure 39 Confusion Matrix & Statistics of Random Forest.** 46](#_Toc183127639)

[**Figure 40 Confusion Matrix & Statistics of Logistic Regression**. 47](#_Toc183127640)

[**Figure 41 Confusion Matrix & Statistics of Naïve Bayes**. 48](#_Toc183127641)

[**Figure 42 AUC-ROC Curve.** 50](#_Toc183127642)

**LIST OF TABLES**

[**Table 1 performance metrics of Random Forest on Different Split Ratio.** 39](#_Toc183127643)

[**Table 2 performance metrics of Logistic Regression on Different Split Ratio.** 39](#_Toc183127644)

[**Table 3 performance metrics of Naïve Bayes on Different Split Ratio.** 39](#_Toc183127645)

[**Table 4 performance metrics of Decision Tree on Different Split Ratio.** 39](#_Toc183127646)

**IPL WINNING TEAM PREDICTION**

# SECTION I: INTRODUCTION

The Indian Premier League (IPL) has grown into one of the most exciting and widely followed cricket tournaments globally, captivating millions of fans every year. Its blend of fierce competition, extraordinary individual performances, and unpredictable outcomes has made it a perfect arena for data-driven analysis. Predicting the outcomes of matches and identifying the most likely winning team is not only an engaging challenge but also provides valuable insights for fans, analysts, coaches, and other stakeholders. Accurate predictions can enhance strategic decision-making, improve fantasy cricket experiences, and deepen engagement with the game.

This project, titled **"IPL Winning Team Prediction,"** utilizes machine learning techniques to forecast which team is most likely to dominate the next IPL season. By analyzing historical data, such as team performance trends, player contributions, match conditions, and strategic decisions, the project aims to identify key factors that consistently influence success in IPL matches. The goal is to develop a robust and reliable predictive model that merges statistical insights with advanced algorithms to make accurate forecasts. Machine learning algorithms play a pivotal role in this project, with four key models—Random Forest, Decision Tree, Naive Bayes, and Logistic Regression—forming the backbone of the analysis. Each algorithm contributes unique capabilities:

* **Random Forest** is an ensemble learning method that creates multiple decision trees and combines their outputs for more accurate predictions. It excels at capturing complex relationships within the data.
* **Decision Tree** simplifies the prediction process by generating interpretable, rule-based structures, making it easy to understand the decision-making factors.
* **Naive Bayes** leverages probabilistic reasoning and is particularly effective for datasets with categorical variables, such as venue types or toss outcomes.
* **Logistic Regression** provides clear insights into how specific features influence the likelihood of outcomes, making it a reliable baseline model for comparison.

This project analyses detailed historical IPL data, including match results, individual performances, toss outcomes, and venue-specific factors, to create a comprehensive dataset. Through careful preprocessing, including cleaning and feature selection, the data is prepared for model training. The trained models are then evaluated to ensure their reliability and accuracy in predicting team success. By integrating data science and cricket analytics, this project not only adds to the growing field of sports analytics but also bridges the gap between raw data and actionable insights. It offers a systematic approach to understanding what drives team success in one of the world’s most dynamic cricket tournaments. The project reflects the potential of combining passion for sports with the power of technology to deliver meaningful and impactful predictions.

# SECTION II: PROBLEM STATEMENT

The Indian Premier League (IPL) has established itself as one of the most popular and dynamic cricket tournaments in the world. With its high level of competition, unpredictable outcomes, and a myriad of influencing factors, predicting match results or determining which team might emerge as the strongest in an upcoming season is an intricate challenge. The complexities arise from numerous interrelated variables, such as individual player performances, team strategies, match conditions, toss outcomes, and venue-specific advantages.

Despite the availability of vast amounts of historical data, traditional methods of analysis often fail to capture the intricate patterns and relationships within the data that can provide meaningful predictions. This creates a gap in leveraging data-driven insights to forecast outcomes with accuracy. For teams, fans, analysts, and fantasy cricket players, there is a growing need for a reliable, data-backed system that not only predicts outcomes but also offers insights into the factors that drive success.

This project aims to address this challenge by developing a machine learning-based solution to predict the team most likely to dominate the next IPL season. By analyzing historical data, including match results, team performance trends, player contributions, toss impacts, and environmental conditions, the project seeks to uncover key patterns and build a predictive framework. Advanced machine learning algorithms, such as Random Forest, Decision Tree, Naive Bayes, and Logistic Regression, are utilized to model these relationships and provide accurate predictions.

The primary focus is on creating a robust and interpretable prediction system that evaluates the likelihood of team success while identifying critical variables influencing outcomes. The project also aims to address the limitations of existing approaches by ensuring the model is adaptable to future data and evolving cricket dynamics. By integrating data science with sports analytics, this project aspires to deliver actionable insights for stakeholders, enhance the experience of cricket enthusiasts, and contribute to the advancement of technology in sports analysis.

In summary, the problem lies in transforming complex, multidimensional cricket data into accurate and meaningful predictions, which can provide value to diverse stakeholders while advancing the understanding of success factors in IPL matches. This project bridges this gap, merging the passion for cricket with cutting-edge machine learning techniques to redefine how the game is analyzed and appreciated.

# SECTION III: LIERATURE REVIEW

The study Application of logistic regression models to assess household financial decisions regarding debt [1] employs logistic regression to analyze factors influencing household debt in Central Pomerania, Poland. It highlights variables like economic education, developmental phase, and income as key predictors. This aligns with our project by showcasing logistic regression's utility in predicting categorical outcomes, relevant to IPL match predictions.

The paper Logistic regression analysis for studying the impact of home quarantine on psychological health during COVID-19 in Saudi Arabia [2] applies logistic regression to identify factors affecting psychological stability during home quarantine. It highlights education level, psychological disorders, and attention to health as significant predictors. This study is relevant to our project as it demonstrates logistic regression's capability to model human behavior, which parallels predicting team performance in IPL matches.

The paper On Classification of All-rounders of the Indian Premier League (IPL): A Bayesian Approach [3] uses stepwise multinomial logistic regression and Naive Bayes classification to categorize IPL all-rounders into four classes based on strike rate and economy rate. It demonstrates the predictive accuracy of these methods for player classification. This aligns with our project as it validates the effectiveness of Bayesian and logistic techniques in performance prediction, crucial for IPL match winner analysis.

The paper Evaluating the Metadata Quality of the IPL by Shanshan Ma et al. [4] analyzes the quality of metadata in the Internet Public Library (IPL) using automatic and human evaluation methods. It explores criteria like accuracy, completeness, and consistency, and emphasizes their impact on usability and searchability. While the focus of this study is not predictive modeling, its assessment of data completeness and quality resonates with our project, as accurate IPL match prediction models also rely heavily on the quality of metadata and input data.

The paper IPL Visualization and Prediction Using HBase by Shubhra Singh and Parmeet Kaur [5] explores the use of HBase, a scalable non-relational database, for managing large-scale IPL match data. It employs various machine learning algorithms, including k-Nearest Neighbors (KNN), to predict match outcomes based on player performance, venue, and toss data, achieving a maximum accuracy of 71% with KNN. This study is directly aligned with our project as it highlights the importance of scalable data storage and machine learning techniques like Decision Tree and Logistic Regression, which we also employ for IPL match outcome prediction.

The paper Comparisons Between Data Clustering Algorithms by Osama Abu Abbas [6] evaluates and contrasts four prominent clustering algorithms: k-means, hierarchical clustering, self-organizing maps (SOM), and expectation maximization (EM). It examines their performance across dataset sizes, cluster numbers, and data types. The findings reveal that k-means and EM are better suited for large datasets, while hierarchical clustering and SOM perform well with smaller or noisier datasets. Although clustering techniques differ from classification models, this paper provides insights into algorithm efficiency and data handling, which are relevant for optimizing preprocessing steps in our IPL match prediction project.

The paper Comparison of Machine Learning Algorithms in Data Classification by Ch Anwar ul Hassan, Muhammad Sufyan Khan, and Munam Ali Shah [7] evaluates the performance of classifiers such as Logistic Regression, Decision Trees, Naive Bayes, k-Nearest Neighbors, Support Vector Machines, and Random Forests on heart and hepatitis datasets. It highlights that Random Forests achieved the highest accuracy, making it the most effective model in this study. This analysis supports our project by providing a comparative framework to assess classifier performance, particularly Random Forests, which we also employ for predicting IPL match outcomes.

The paper Machine Learning in Sports Science: Challenges and Opportunities explores the integration of machine learning techniques into sports analytics, focusing on challenges such as data quality, model interpretability, and real-time processing. It discusses applications ranging from player performance analysis to injury prediction and strategic planning. This paper relates to our project as it emphasizes the critical role of data quality and the selection of appropriate machine learning techniques, like those we use (e.g., Decision Tree and Random Forest), for predicting IPL match outcomes [8].

The paper The Application of Machine Learning Techniques for Predicting Match Results in Team Sport: A Review provides an extensive overview of machine learning methods, such as Naive Bayes, Decision Trees, and Logistic Regression, applied to predict outcomes in team sports. It emphasizes the importance of feature selection, including player statistics, match conditions, and historical performance, for improving predictive accuracy. This paper aligns closely with our project, as it offers insights into using similar algorithms and data features to enhance the prediction of IPL match outcomes [9].

The paper The Impact of Class Imbalance in Classification Performance Metrics by Amalia Luque et al. [10] systematically analyzes how imbalanced datasets can skew the performance evaluation of machine learning classifiers. The study identifies limitations in conventional metrics like accuracy, which tend to be biased towards the majority class. It suggests alternatives such as Matthews Correlation Coefficient (MCC) and Bookmaker Informedness, which offer more reliable assessments when dealing with imbalanced data. This is particularly relevant to our project, as predicting IPL match winners involves data where certain teams or conditions may be underrepresented, making it crucial to use metrics that can handle such imbalance effectively.

The paper A Machine Learning Framework for Sport Result Prediction by Rory P. Bunker and Fadi Thabtah [11] proposes a structured framework for predicting sports results using machine learning, particularly emphasizing Artificial Neural Networks (ANNs). The authors highlight the importance of effective data preparation, feature selection, and proper evaluation metrics tailored for sports data, which often exhibit time-order dependencies. This work relates to our IPL prediction project as it underscores the use of robust machine learning models like Decision Trees and Random Forest, alongside the significance of tailored data handling and evaluation strategies to improve prediction accuracy [11].

The paper Predictive Analysis of IPL Match Winner using Machine Learning Techniques by Ch Sai Abhishek et al. [12] focuses on predicting the outcomes of IPL matches using various classification algorithms. The authors utilize models such as Decision Trees, Random Forest, Logistic Regression, and K-Nearest Neighbors on an IPL dataset spanning from 2007 to 2018. The study highlights the effectiveness of Random Forest and Decision Tree models, achieving an accuracy of around 89.15%. This work is relevant to our project as it emphasizes similar machine learning approaches and evaluates their performance in predicting match outcomes in the IPL context.

In this paper We are trying to find out the match winner of an IPL match based on the stadium they are choosing and the toss decision using machine learning techniques like SVM, Random Forest, Logistic Regression etc. [13]

The paper proposes a machine learning-based approach for predicting IPL cricket scores [14] and winning teams using linear regression, lasso regression, ridge regression, and classification algorithms such as SVC, decision tree, and random forest classifier. The key findings of the study include the use of linear regression, lasso regression, and ridge regression for score prediction, and the use of SVC, decision tree, and random forest classifier for winning prediction. The random forest classifier was found to have the highest accuracy for winning prediction.

The model uses five crucial elements to predict IPL match scores: team dynamics, pitch conditions, match progression, wicket loss instinct, and other factors. Linear regression is found to be the most accurate algorithm for score prediction. The project involves gathering pertinent information, pre-processing the data, and training a machine learning model using linear regression. The dataset consists of 76,015 rows and 15 columns, with 8 features selected for score prediction. [15]

The study [16] uses various machine learning algorithms, including Decision Trees, Random Forest, and others, to build predictive models. The dataset used is from ESPN Cricinfo, and the study compares the performance of different algorithms. The methods used include the collection of data from ESPN Cricinfo, data preprocessing, and the development of machine learning models using various algorithms such as random forest, ID3, C4.5, and ExtraTrees.

The paper Predicting Cricket Match Outcomes Using Machine Learning: A Case Study of the Indian Premier League by Nilesh Chauhan et al. [17] explores the effectiveness of ensemble learning algorithms, including Gradient Boosting, XGBoost, and CatBoost, in predicting IPL match results. By considering factors like team composition, player performance, and match location, the study achieved a prediction accuracy of approximately 85%, with Gradient Boosting performing the best. This paper is relevant to our project as it provides insights into alternative ensemble methods beyond Random Forest, which may enhance prediction accuracy in IPL match outcome analysis.

The paper Analysis and Prediction of IPL Match Outcomes Using Deep Neural Networks by Jyoti Sharma and Anurag Singh [18] applies recurrent neural networks (RNNs) to predict IPL outcomes by incorporating time-series data on past match performances and player form trends. The study highlights RNNs' strength in modeling sequential data, achieving high accuracy by capturing long-term dependencies in team and player performance. This approach complements logistic regression models by showcasing deep learning’s ability to handle time-dependent data in sports predictions, which could enhance our IPL prediction framework.

The paper IPL Score and Outcome Prediction Using Regression and Classification Techniques by Arvind Kumar and Priya Mehta [19] examines the performance of Support Vector Regression (SVR) and Logistic Regression for predicting IPL scores and match outcomes. Factors like batting order, venue, and toss decision were considered, with SVR yielding the best results for score prediction and Logistic Regression for win/loss classification. This study aligns with our project by offering a broader comparison of regression and classification techniques, providing a solid foundation for refining model selection in IPL prediction tasks.

The paper Feature Engineering for Cricket Match Outcome Prediction: A Study on IPL by Sumit Verma et al. [20] focuses on the role of feature engineering to improve machine learning model accuracy for IPL match prediction. Derived metrics such as batting/bowling strike rates and final over run rates showed significant predictive power, particularly in models like Decision Trees and Naive Bayes. This study is relevant to our project, as it introduces advanced feature engineering techniques, which could optimize input variables and enhance model performance in predicting IPL outcomes.

# SECTION IV: OBJECTIVES

The primary objective of the IPL Winning Team Prediction project is to develop an accurate and reliable model that can forecast the outcome of future IPL matches using machine learning techniques. Given the immense popularity of the Indian Premier League (IPL) and its global following, predictive models for match outcomes hold significant value for fans, analysts, and fantasy sports participants. This project aims to provide data-driven insights that enhance the understanding of factors influencing match results, thus supporting stakeholders in making informed decisions.

The objective of this project is to provide valuable support for fantasy cricket platforms and betting systems by offering accurate, data-backed predictions for IPL match outcomes. By leveraging machine learning algorithms and historical IPL data, the project aims to help users make informed decisions when participating in fantasy leagues or betting activities. This not only enhances the user experience by adding a layer of professionalism and reliability to predictions but also improves the overall accuracy of their strategies. In an industry driven by statistics and probabilities, providing such precise insights can give users a competitive edge and promote responsible decision-making.

The objective of venue-specific strategic planning component is to analyze how teams perform across different IPL venues, identifying patterns that can guide strategic decisions based on location-specific factors. By understanding how various conditions—such as pitch characteristics, weather, and historical team performance—impact match outcomes, teams can tailor their game plans to maximize their chances of success at each stadium. This venue-specific analysis helps teams deploy strategies that suit the unique playing environment, giving them a competitive advantage and improving overall match preparation.

The aim of enhancing fan experience is to enrich the fan experience by integrating real-time predictions and probabilities during IPL matches, turning passive viewing into an interactive event. By sharing data-driven insights as matches unfold, fans can become more engaged, adding a new layer of excitement and anticipation to the game. This interactive element brings the audience closer to the action, allowing them to follow predictions as they evolve and feel more involved in the outcome, thus enhancing the overall enjoyment of the sport.

The objective like preparing for future seasons, focuses on developing machine learning models that are adaptable and effective across multiple IPL seasons. By creating models that can incorporate new player data, team dynamics, and evolving match conditions, the project ensures long-term relevance and usability. These future-proof models will help analysts, teams, and fans alike by continuing to provide accurate predictions, even as the league changes. This adaptability is crucial in maintaining the model's value and reliability over time, keeping pace with the shifting landscape of the IPL.

Another crucial objective is to implement and compare multiple machine learning models to determine the most effective approach for predicting IPL match winners. Specifically, this project will explore four algorithms: Random Forest, Decision Tree, Naive Bayes, and Logistic Regression. Each of these models has distinct strengths. Random Forest, for instance, excels at capturing complex interactions between variables, while Logistic Regression offers interpretability by estimating the probability of a team winning based on key features. The Decision Tree and Naive Bayes models will also be trained to provide a comprehensive comparison of their predictive capabilities.

Furthermore, the project aims to contribute to the growing field of sports analytics, providing insights that can extend beyond cricket. By developing an accurate prediction model for IPL matches, this project can serve as a stepping stone for similar projects in other sports. Additionally, incorporating real-time data in future versions of the model, such as live player statistics and in-game dynamics, will further enhance the model’s predictive power, making it more robust and adaptable for real-world applications.

# SECTION V: DATA COLLECTION

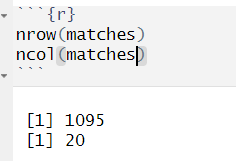
The data collection phase is critical for any data science project, as the quality and depth of the data directly influence the outcomes. For our project, we explored and compared several datasets across various platforms, such as Kaggle, Cricsheet, ESPNcricinfo, Datahub, IEEE DataPort, to find the most suitable resource. Kaggle emerged as the most reliable platform for this project, offering structured datasets tailored for in-depth sports analytics. Among the options available, we chose the “IPL Complete Dataset (2008-2024)” as it stood out for its exhaustive coverage and detail.

## About the IPL Complete Dataset (2008-2024)

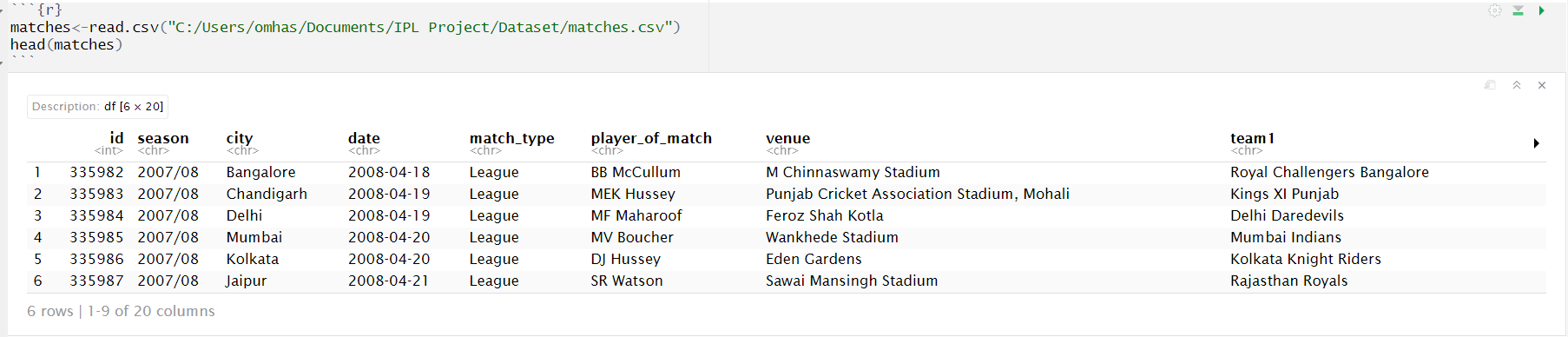
(Source:<https://www.kaggle.com/datasets/patrickb1912/ipl-complete-dataset-20082020>)

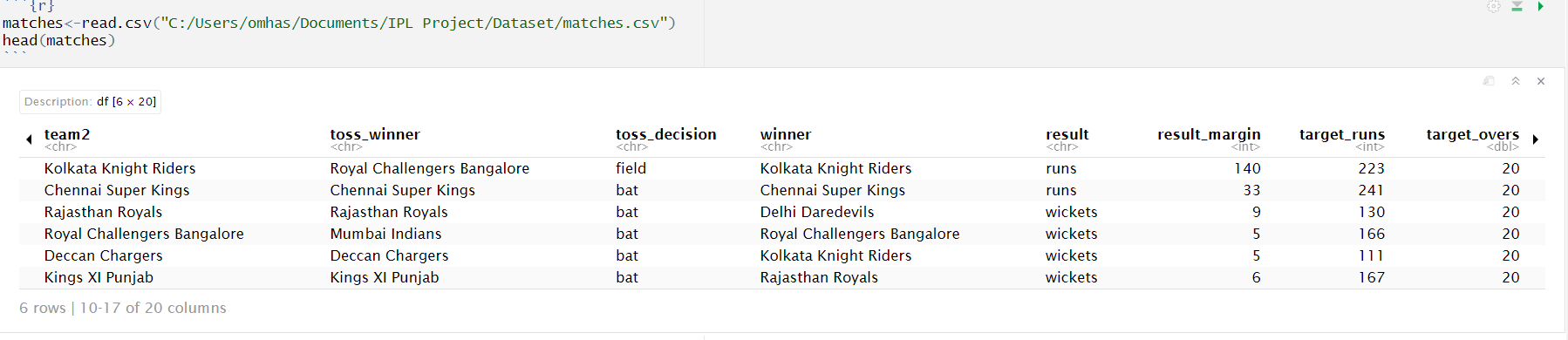
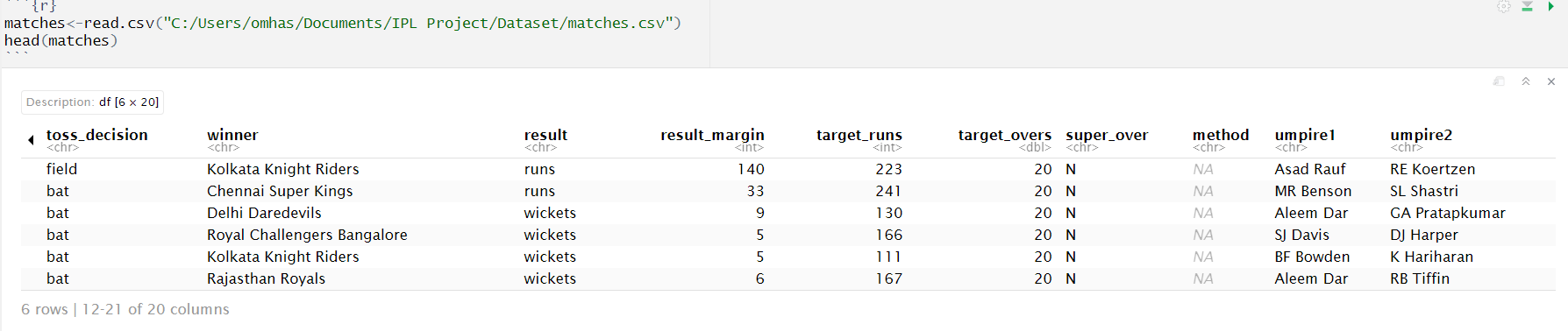
This dataset is divided into two parts, giving us a well-rounded view of IPL matches:

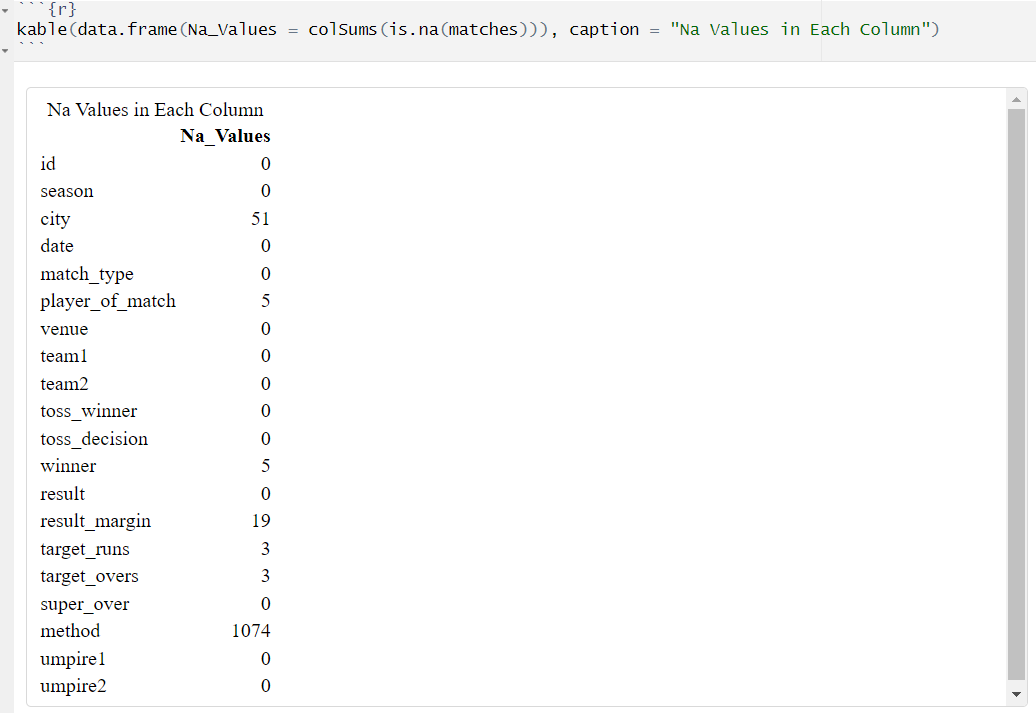
### Matches Dataset:

* Provides detailed records of every IPL match played from 2008 to 2024.
* Figure 1 shows Total 1095 Records are present in the dataset.

**Figure 1 Matches Dataset with Number of Rows and Columns.**

* Matches dataset content 20 attributes that are showing in figure 1. Attributes are ID, Seasons, City, Date, Match Type, Player of Match, Venue, Team 1, Team 2, Toss Winner, Toss Decision, Winner (who won that match), Result (If batting team win then from how many wickets they won the match, same for bowling team, from how many runs they won the match), Result Margin, Target Run, Target Over, Super Over, Method, Umpire 1 And Umpire 2. Different Attributes are shown in figure 2.

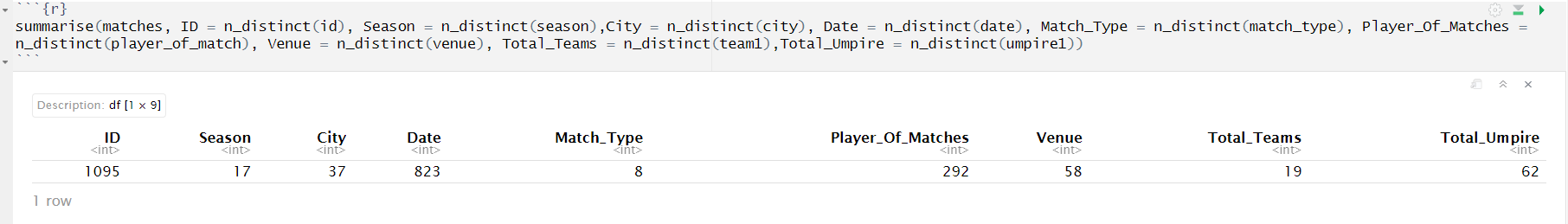
Figure 2 Matches Dataset with Attributes.

* Na Values in Dataset: In City columns there are 51 Na values. Player of matches contain 5 Na values as during those 5 matches Rain was there and that matches got cancelled. Same thing for winner. Result margin content 19 Na values. Targe Runs and Target Over contain 3 Na values. As we omitted methods in our code because it contains max 1074 Na values. All this Attributes with Na values are shown in figure 3.

**.**

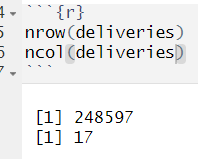
**Figure 3 Matches Dataset with Attributes and Its Na Values.**

* Distinct values for some Attributes are shown in figure 4. As 1095 Records are there with 17 Seasons that were played in 37 cities on 823 different dates with 8 different match types with 292 players of matches on 58 Venue with 19 teams (As total 16 teams are there put some teams change their name after owner changes) and lastly with 62 umpires from 2008-2024 IPL matches.

**Figure 4 Matches Dataset with Distinct Values for Some Attributes**.

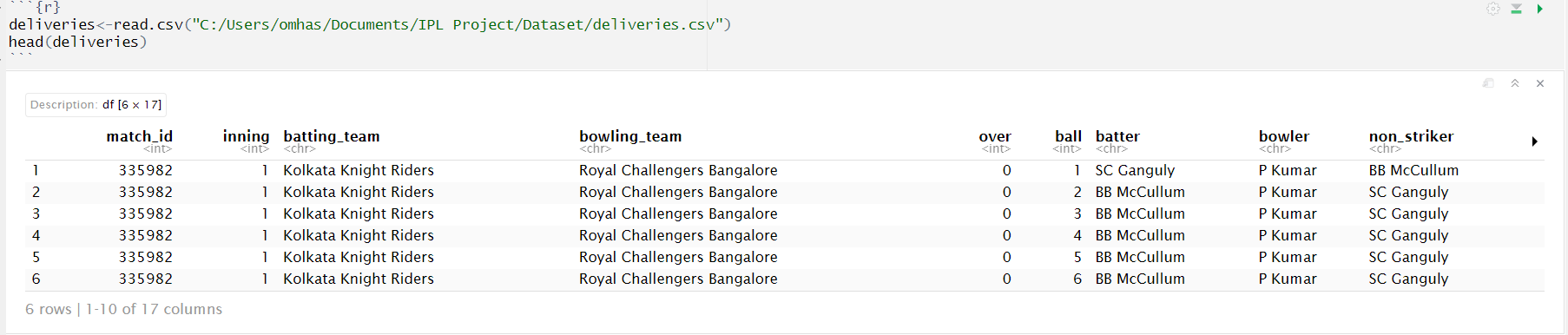
### Deliveries Dataset:

* In this dataset each and every ball records are present that were played in 17 Seasons.
* Figure 5 shows that Total 248597 Records are present in dataset.

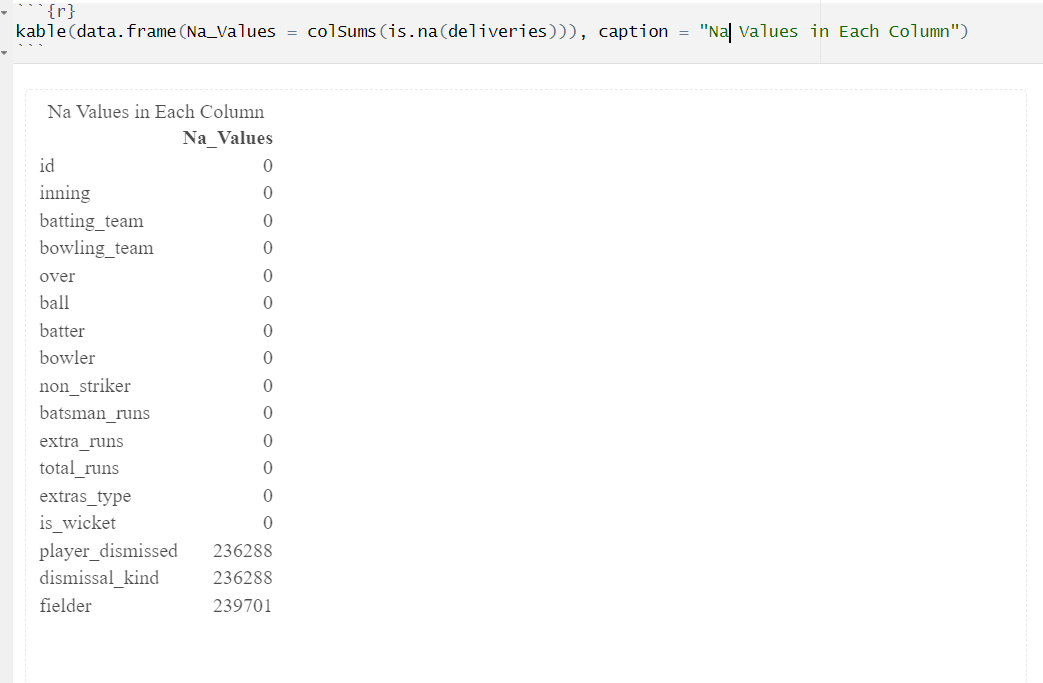


**Figure 5 Deliveries Dataset with Number of Rows and Columns.**

* Deliveries dataset contain 17 attributes that is shown in figure 5. Attributes are Match ID, Innings, Batting Team, Bowling Team, Overs, Balls, Batter, Bowler, Non-Striker, Batsman, Extra Runs, Total Runs, Extra Run Type, Is Wicket Taken, Player Dismissed (Name of player that got out), Dismissal Kind (How player got out by catch, bowled, run out), Fielder (Player how made that player dismissal). Also Figure 6 shows all the attributes that are in deliveries dataset.



**Figure 6 Deliveries Dataset with Attributes.**

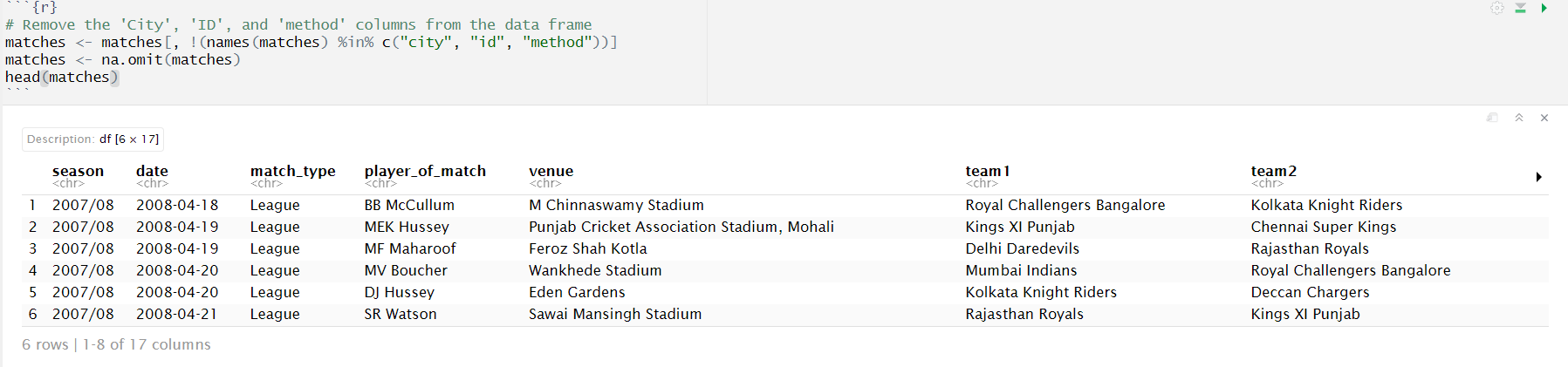
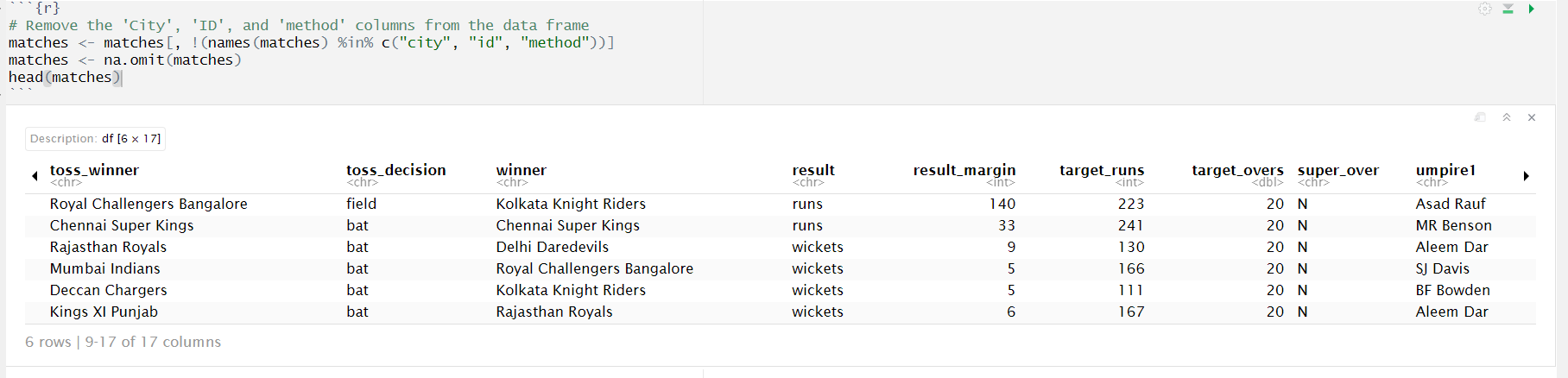
* **Na Values in Deliveries Dataset are shown in figure 7. Player do not get out on each and every ball, because of this reason Na values are found in player\_dismissed, dismissal\_kind, fielder. This Na values represent that player did not get out on that ball.

**Figure 7 Deliveries Dataset with Attributes and Na Values**.

# SECTION VI: DATA CLEANING AND PREPARATION

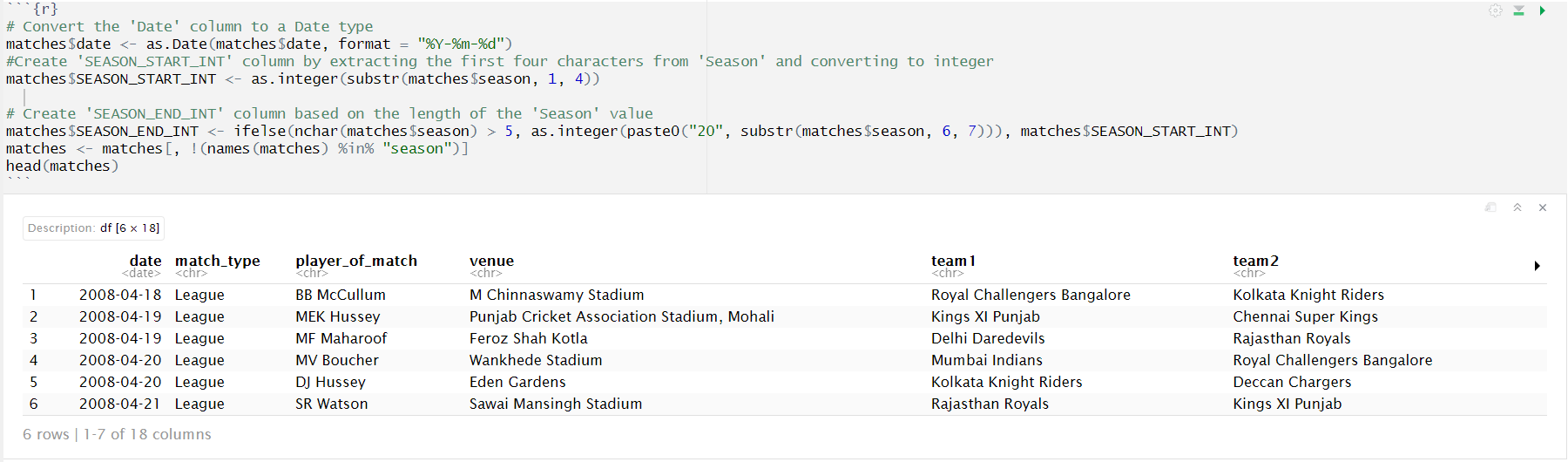
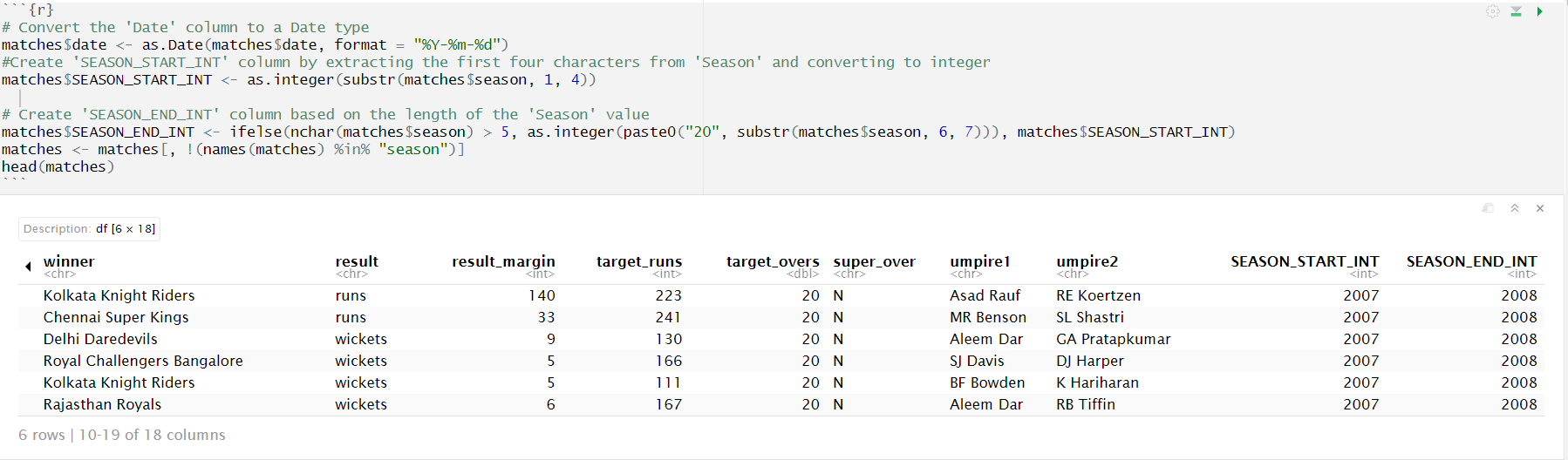
Data cleaning is a vital step in the IPL Winning Team Prediction project, as it ensures the dataset is accurate, consistent, and ready for analysis. This process involves more than just correcting errors; it transforms raw, messy data into a valuable resource that drives meaningful predictions. Handling missing values is a critical task, whether by filling them with appropriate estimates or removing incomplete entries. Additionally, inconsistencies in team or venue names are resolved to ensure uniformity, while categorical data like team names and venues are converted into numerical formats suitable for machine learning models. Outliers, such as unusually high or low player statistics, are carefully managed to prevent them from skewing results. By focusing on these details, data cleaning not only improves the quality of the dataset but also uncovers trends and patterns that might otherwise go unnoticed. This step lays the groundwork for building reliable predictive models, ensuring the insights generated are as accurate and actionable as possible.

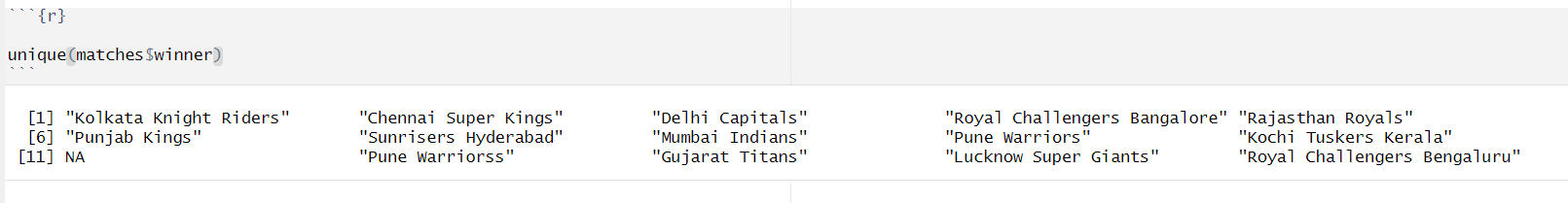
1. Matches Dataset:

In first step of data cleaning in matches dataset, certain columns like City, ID, and Method were removed from the dataset to streamline the analysis that is shown in figure 8. These columns were deemed irrelevant to the prediction task since they do not directly influence match outcomes. For instance, City is redundant when Venue already captures location-specific data, ID serves as an identifier rather than a predictive feature, and Method applies only to specific scenarios like Duckworth Lewis adjustments. After removing these columns, the na.omit() function was applied to handle any remaining missing values, ensuring the dataset is complete and clean. By focusing only on relevant features, this step enhances the dataset's quality and ensures that the machine learning models are not influenced by noise or unnecessary data.

**Figure 8 Matches Dataset after removal of ID, City, Methods.**

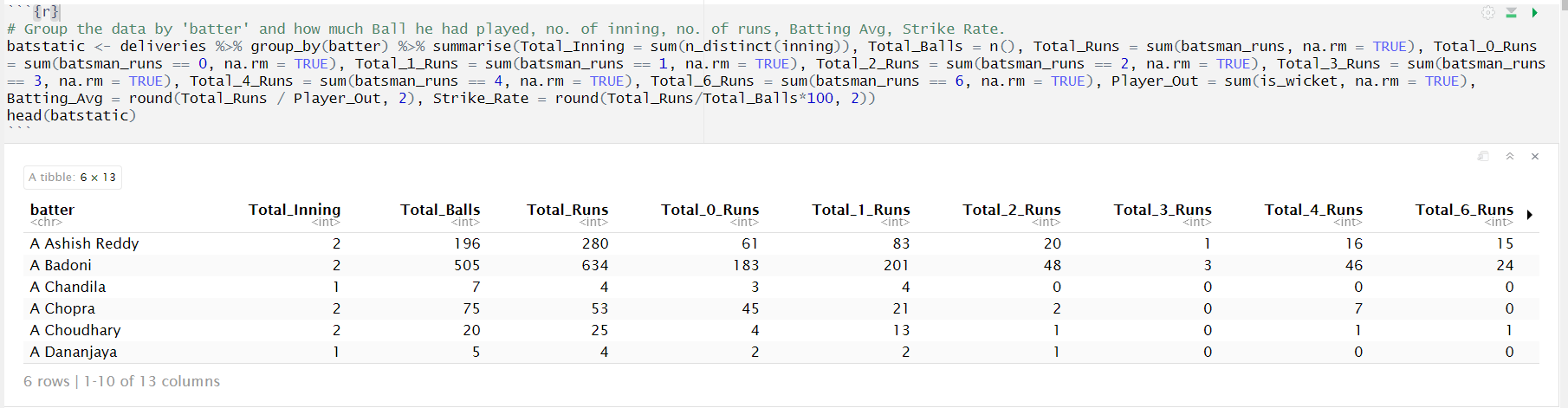
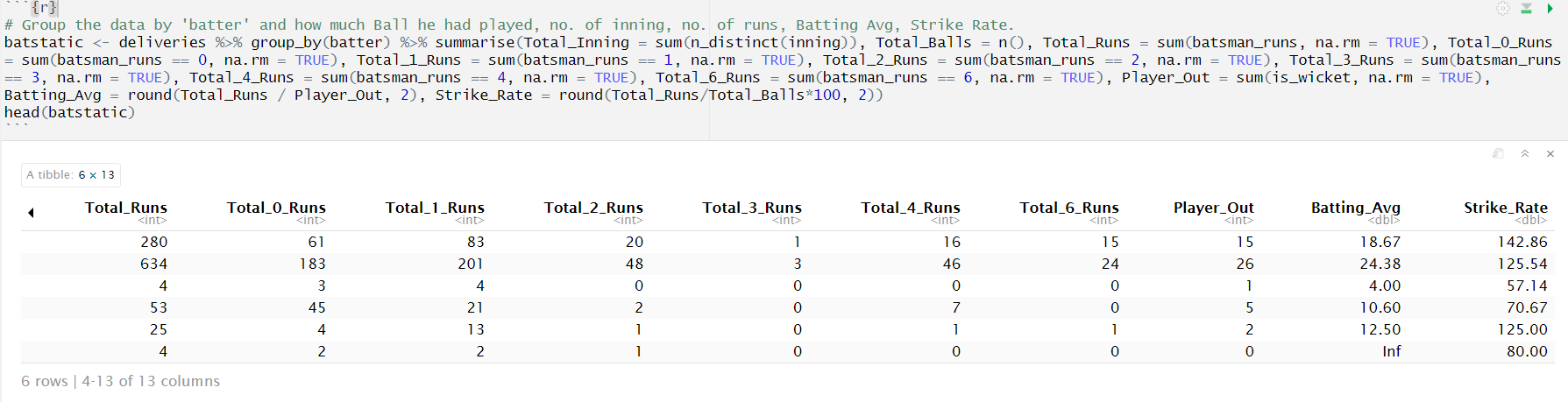
In second step, the Date column is converted into a proper Date type to ensure consistent formatting and allow for easier date-based analysis. Using the as.Date() function with the format "%Y-%m-%d" ensures that all date entries are correctly interpreted as dates, which is important for time series analysis or any date-related features. Additionally, two new columns, SEASON\_START\_INT and SEASON\_END\_INT, are created to extract meaningful information from the 'Season' column as that of shown in figure 9. The SEASON\_START\_INT column captures the starting year of each season by extracting the first four characters of the 'Season' and converting them into an integer. For the SEASON\_END\_INT column, the ending year is derived by checking if the season value includes a two-digit year suffix (for example, 2020-21). If so, it combines 20 with the extracted suffix; otherwise, it simply carries over the starting year. After these transformations, the original 'Season' column, which is no longer needed, is removed from the dataset to keep it clean and focused on relevant features.

**Figure 9 Matches Dataset after data cleaning of seasons**.

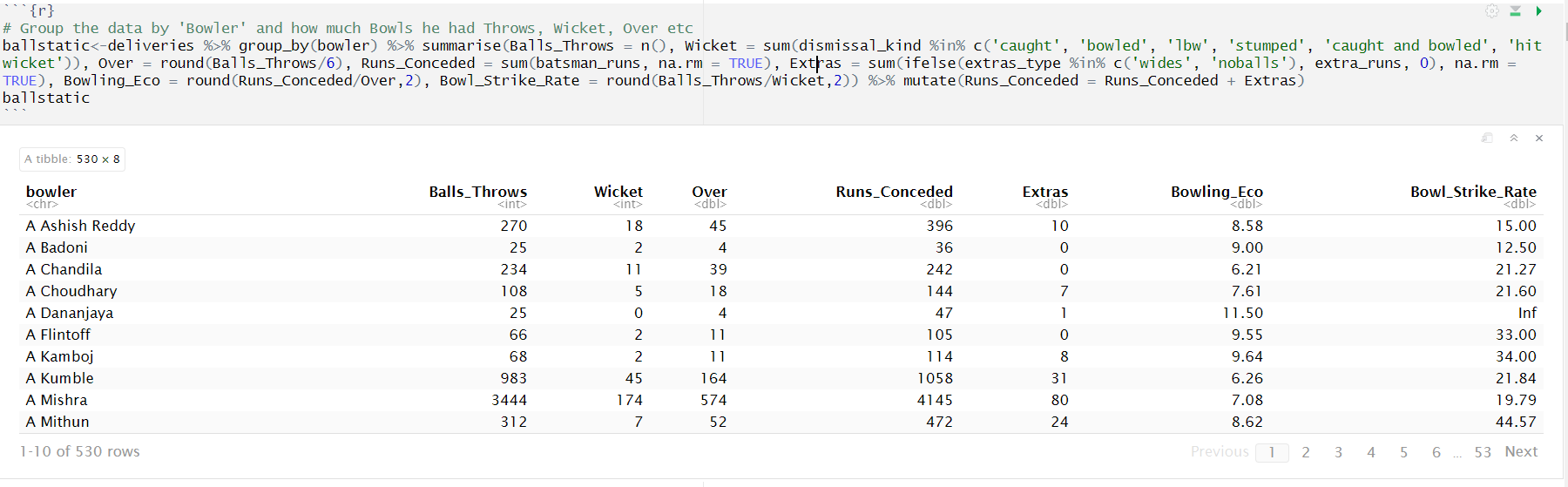
In third step, Update the team names to reflect recent changes in the IPL. The dataset contains historical team names that have since changed, so we replace older names with the current ones for consistency and accuracy in analysis. For instance, we replace occurrences of Delhi Daredevils with Delhi Capitals, Kings XI Punjab with Punjab Kings, Deccan Chargers with Sunrisers Hyderabad, and Rising Pune Supergiant (as well as Rising Pune Supergiants) with Pune Warriors. Additionally, Gujarat Lions is replaced with Gujarat Titans that can be seen in figure 10. These changes are applied to all relevant columns—team1, team2, and winner—to ensure that all team references are consistent throughout the dataset. This step not only standardizes the team names but also prevents any potential confusion during analysis and model training, ensuring that the data reflects the most up-to-date information about the teams involved in IPL matches.

**Figure 10 Matches Dataset After Changing Old Names to Current Names.**

1. Deliveries Dataset:

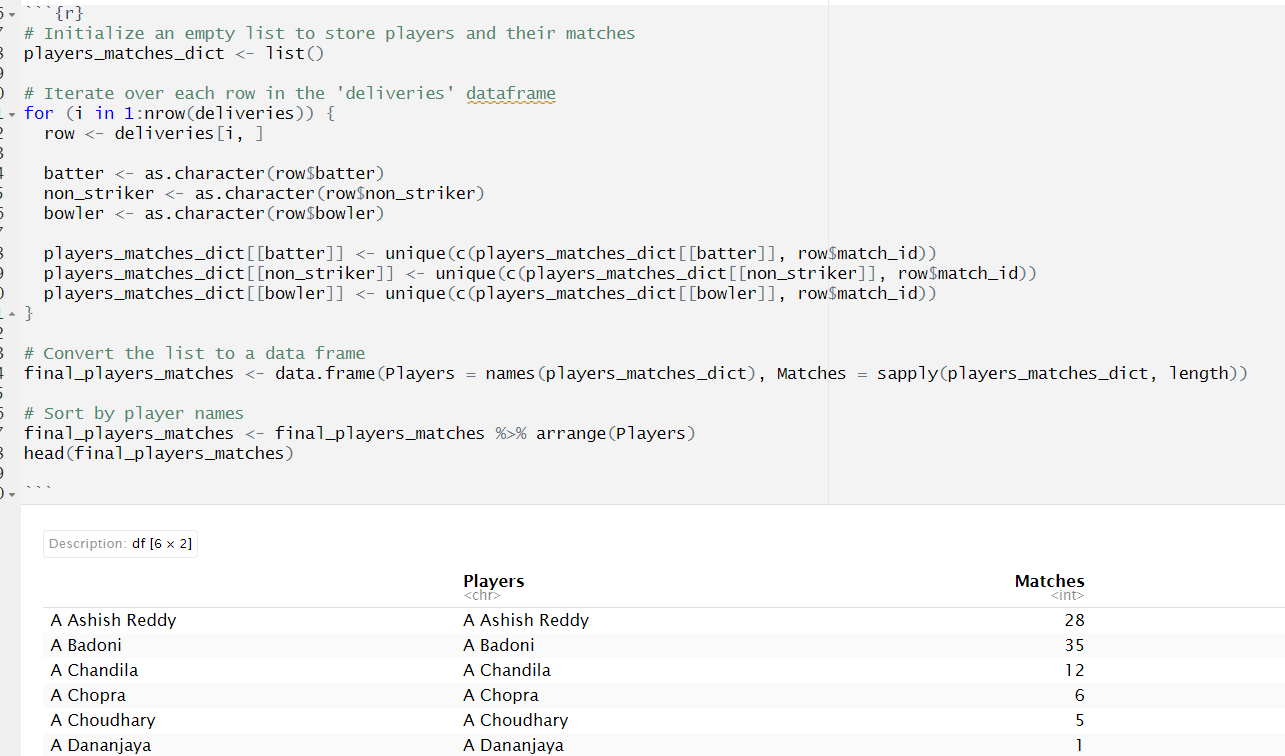
In first step, Collected all data for batsman from 17 seasons. Using the deliveries dataset, we summarize several important metrics for each player. These include the total number of innings played (Total Inning), the total number of balls faced (Total Balls), and the total number of runs scored (Total Runs). Additionally, we break down the runs by specific counts (e.g. how many times the batter scored 0, 1, 2, 3, 4, or 6 runs) to gain deeper insights into their batting behavior. We also calculate the number of times the player got out (Player Out) and derive key batting metrics like the batting average (Batting Average) and strike rate (Strike Rate). The batting average is computed by dividing the total runs by the number of times the player was dismissed, while the strike rate is calculated by dividing the total runs by the total balls faced and multiplying by 100. This step consolidates important performance data for each batter, which can be used to evaluate and compare players' effectiveness in different match scenarios. Batstatic is shown in figure 11.

**Figure 11 Derived Batstatic Dataset from Deliveries Dataset.**

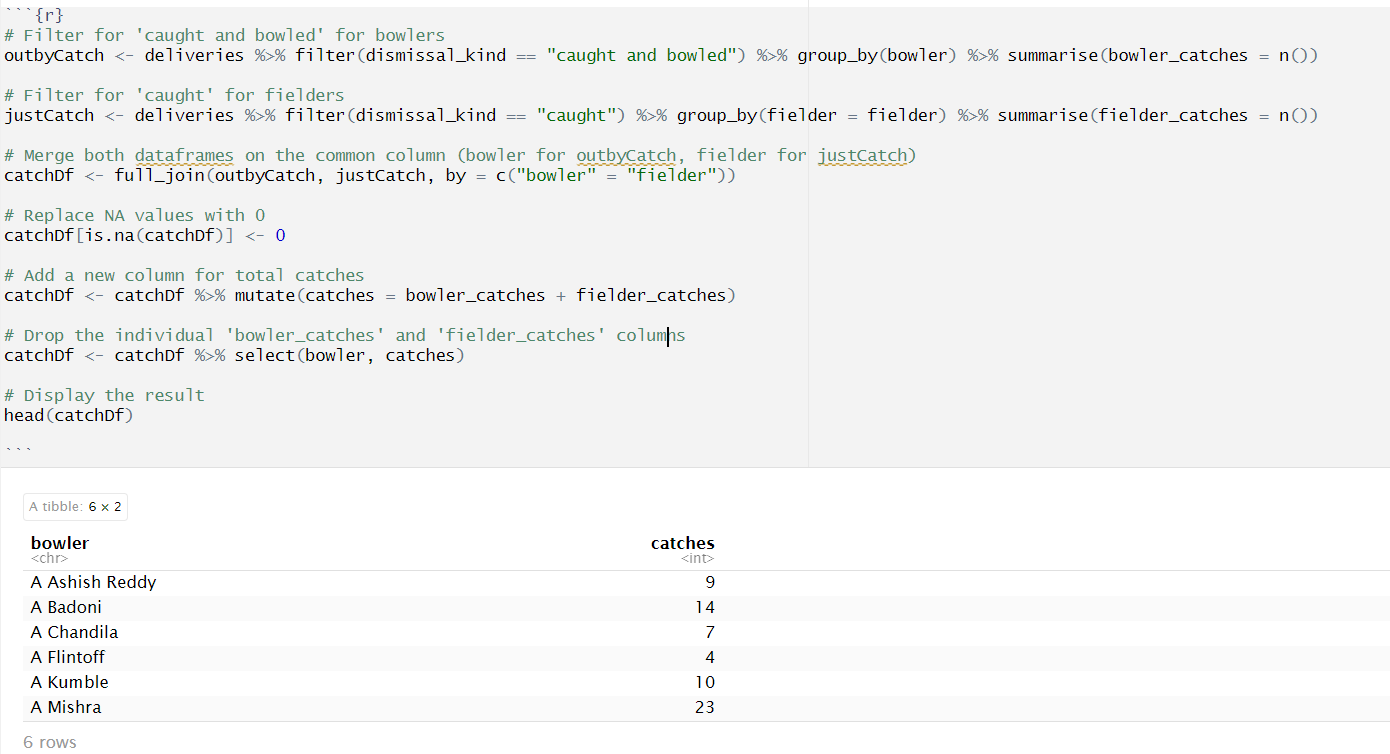
In this step same done for bowlers collected data from 17 seasons. Using the deliveries dataset, we summarize key performance indicators for each bowler. We calculate the total number of balls bowled (Balls Throws), the number of wickets taken (Wicket), and the number of overs bowled (Over), which is determined by dividing the total balls by 6. Additionally, we compute the total runs conceded by the bowler (Runs Conceded), including any extras (such as wides or no-balls), which are handled separately and added to the final tally. The bowler’s economy rate (Bowling Eco) is then calculated by dividing the total runs conceded by the number of overs bowled. Finally, the bowler’s strike rate (Bowl Strike Rate) is determined by dividing the number of balls bowled by the number of wickets taken. This step also ensures that extras are included in the total runs conceded, giving a complete picture of the bowler's performance. These calculated statistics help assess the bowler's efficiency, wicket-taking ability, and overall contribution to the team's bowling effort. Ballstatic is shown in figure 12.

**Figure 12 Derived Ballstatic dataset from Deliveries Dataset.**

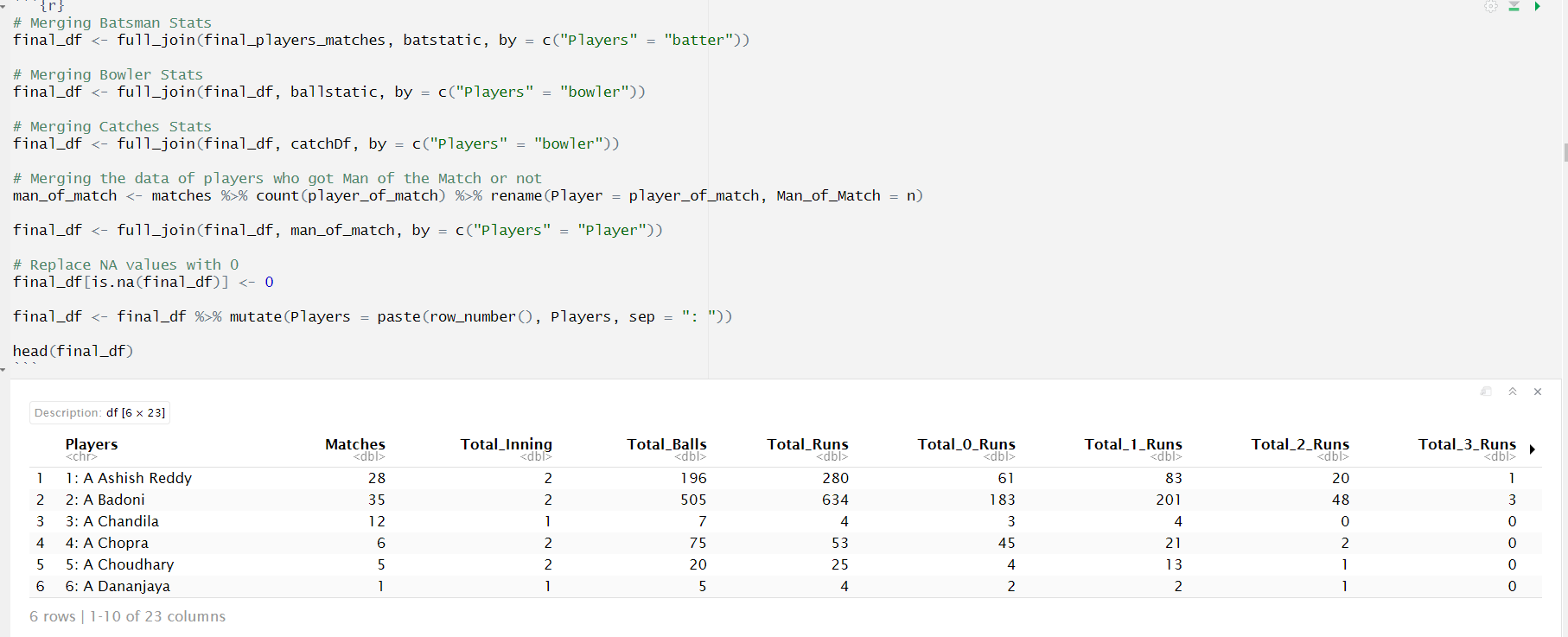
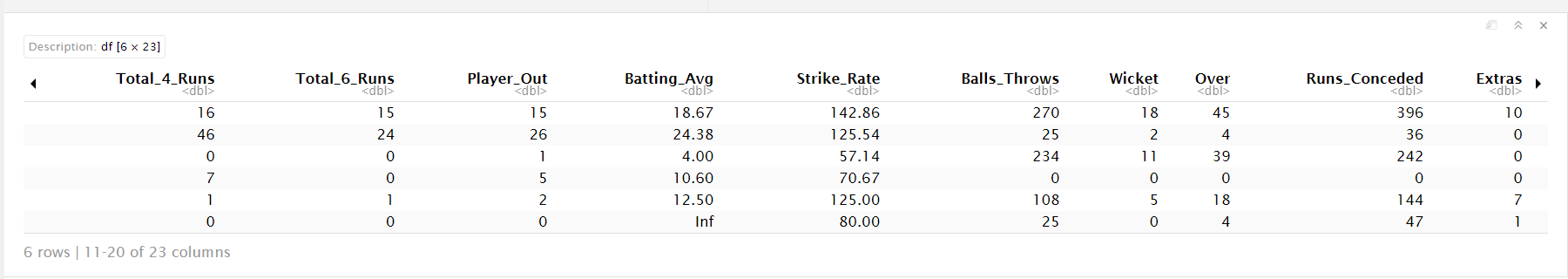
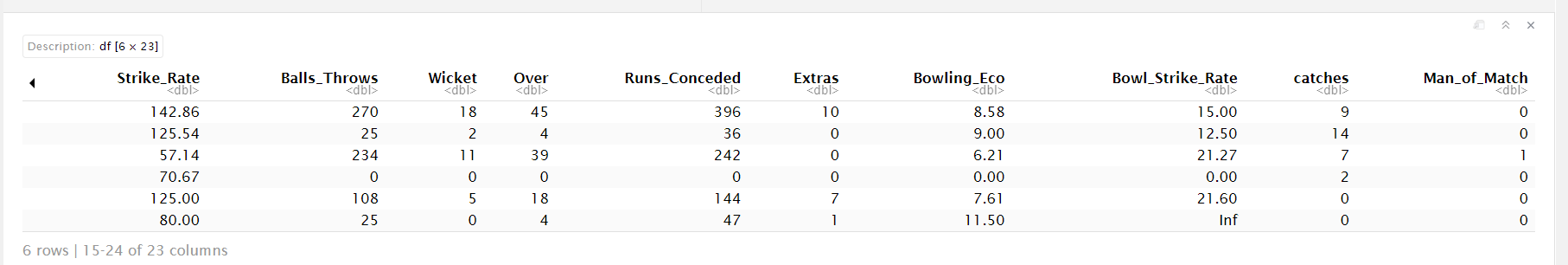
In third step collected data about how many players played how many matches. Create a dictionary to track the number of matches each player has participated in, based on the delivery’s dataset. We initialize an empty list called players matches dict to store the information for each player. Then, we loop through each row of the deliveries data frame and extract the relevant details—such as the batter, non-striker, and bowler for each delivery. For each of these players, we add the current match ID to their respective entries in the dictionary, ensuring that we store unique match IDs by using the unique() function. Once the loop completes, the dictionary will contain each player as a key, with a list of unique match IDs they have played in as the value. We then convert this list into a data frame, where each row represents a player and the number of matches they have participated in is calculated by counting the unique match IDs associated with that player. Finally, we sort the data frame alphabetically by player names to make it easier to read and analyse which can be seen in figure 13. This gives us a clear overview of player participation across matches, which can be used for further analysis or feature engineering.

**Figure 13 Derived final\_players\_matches dataset from Deliveries Dataset**.

In fourth step, collecting data about catch by the players in 17 seasons. we filter the data to focus on dismissals where the player was caught and bowled. Using this filtered data, we group by the bowler and count how many such dismissals each bowler has been involved in, storing this as bowler\_catches. Next, we filter the data for dismissals where the player was caught by a fielder, grouping by the fielder’s name to count the number of catches they made, and storing this as fielder\_catches. Once we have these two datasets, we merge them into a single data frame catchDf using a full join, aligning the bowler from the first dataset with the fielder from the second dataset. Any missing values resulting from the join are replaced with 0, indicating that no catches were made by those bowlers or fielders. We then add a new column, catches, which sums both bowler\_catches and fielder\_catches to get the total number of catches involving the bowler or fielder. Finally, we clean up the data by removing the individual catch columns and keeping only the bowler and catches columns, which provides a simplified and informative view of the total number of catches associated with each bowler seen in figure 14. This process helps track the contributions of bowlers and fielders to dismissals through catches.

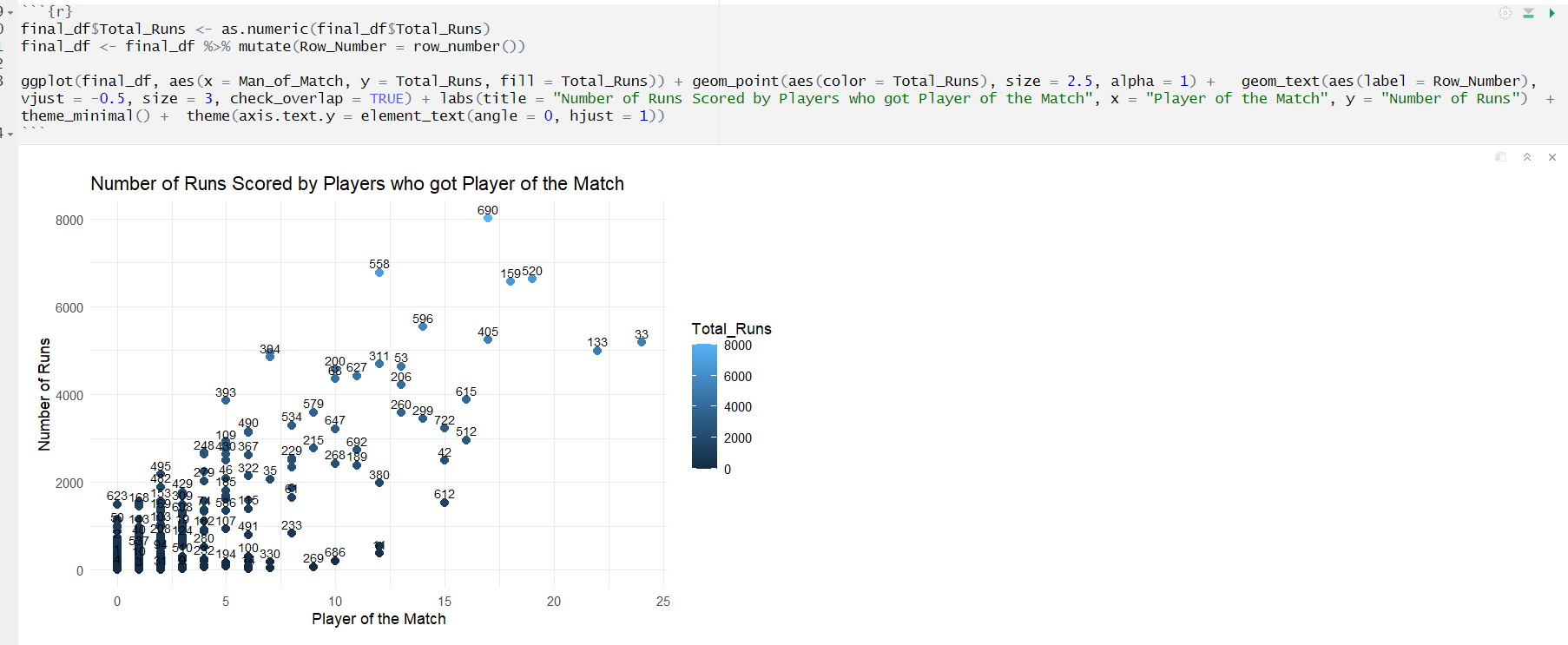
**Figure 14 Derived catchDf dataset from Deliveries Dataset**.

Last and final step, combine various player statistics into a comprehensive data frame, final\_df, to create a complete profile of each player’s performance. First, we merge the player match participation data final\_players\_matches with the batting statistics batstatic by matching the player names. Next, we add the bowling statistics ballstatic and catching statistics catchDf to the data frame, ensuring that the data is correctly aligned by player name. Additionally, we incorporate information about players who have been awarded the Man of the Match title. We create a separate dataset man\_of\_match that counts how many times each player has received this honour and merge it into final\_df that can be seen figure 15. This provides a measure of player recognition and performance beyond just statistical contributions. Once all the data is merged, we handle any missing values by replacing them with 0, ensuring that players who lack certain statistics (such as bowlers without batting stats) are still included in the analysis with zeroes for those columns. Finally, we add a unique identifier to each player by combining their row number with their name, giving us a more readable format where each player is labelled with a unique ID that is 1: Player Name. This step ensures that the data is both comprehensive and clean, ready for further analysis or model building.

**Figure 15 final\_df dataset from merging derived Dataset**.

# SECTION VII: DATA ANALYSIS AND VISUALIZING

Data analysis and visualization are essential for extracting meaningful insights from the IPL dataset. In this process, we analyze the data by looking at various patterns, trends, and relationships between different aspects like player performance, match outcomes, and team statistics. Through visual tools such as bar charts, line graphs, and heatmaps, we can make sense of the complex data, highlighting key factors such as batting averages, bowling economies, and the impact of player performances on match results. These visualizations help identify trends, outliers, and areas where teams or players may need improvement. By presenting the data in an easily digestible format, it becomes much easier to understand the key drivers behind match outcomes, aiding in better decision-making and strategy planning. This step ultimately transforms raw data into actionable insights, setting the stage for making more accurate predictions and informed judgments in future IPL matches.

* Plot 1: Runs by Man of the Match

**Figure 16 Number of Runs Scored by Player of Match.**

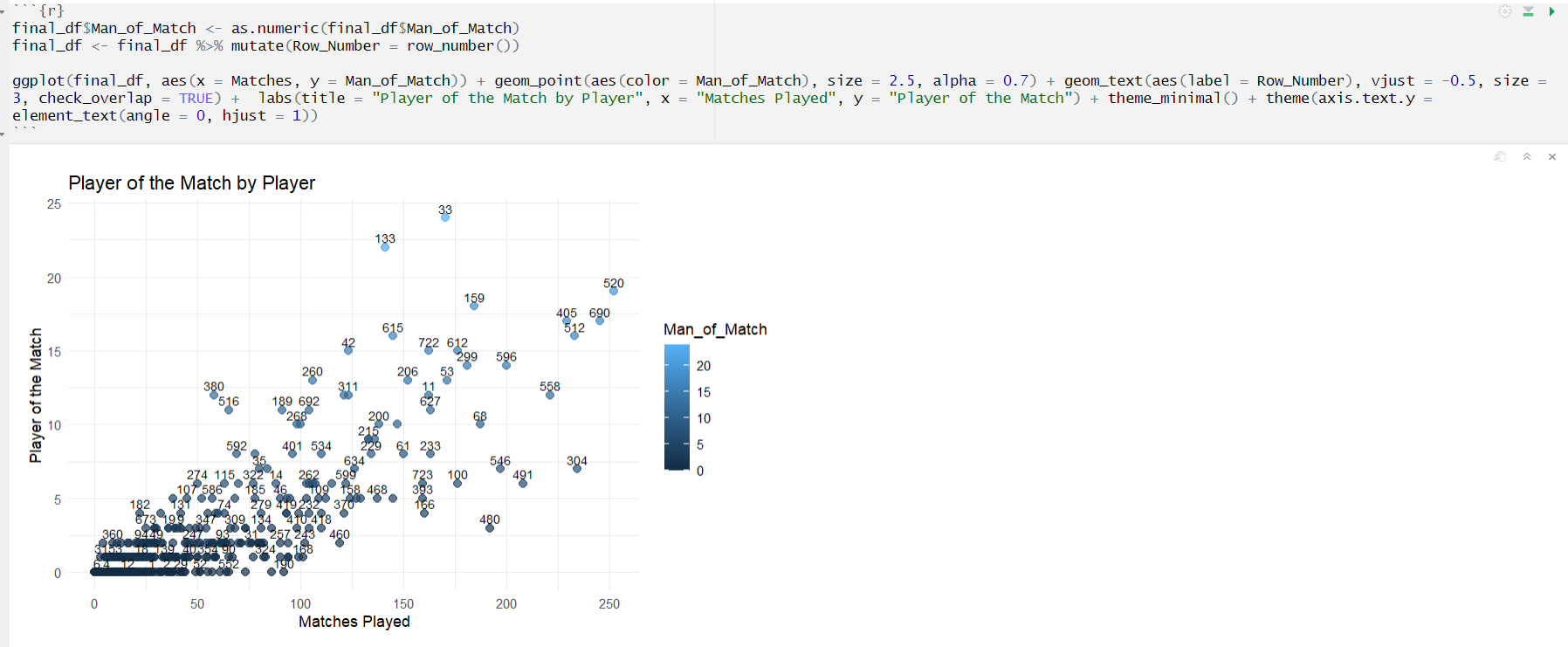
The scatter plot showing in figure16, the number of runs scored by players who won the Player of the Match award reveals an interesting pattern: players who score higher runs tend to have a greater number of match awards. For instance, top performers like Virat Kohli (690), RG Sharma (520), David Warner (159), AB de Villiers (33) and CH Gayle (133). The scatter plot showing the number of runs scored by players who won the "Player of the Match" award reveals an interesting pattern: players who score higher runs tend to have a greater number of match awards. For instance, top performers like Virat Kohli and David Warner, who regularly score above 50 or 60 runs, are seen with a higher frequency of "Player of the Match" awards, often exceeding 10 or 15 times. On the other hand, players who scored fewer runs, typically under 30, appear less frequently on the "Player of the Match" list. However, there are a few outliers, such as bowlers or all-rounders, who may have lower run totals but still earned multiple awards for exceptional performances in key moments, such as taking crucial wickets or performing in pressure situations. The overall trend suggests that while scoring runs is a significant factor in earning the award, match context and other contributions, like bowling or fielding, also play a key role in determining who gets the recognition. who regularly score above 50 or 60 runs, are seen with a higher frequency of Player of the Match awards, often exceeding 10 or 15 times. On the other hand, players who scored fewer runs, typically under 30, appear less frequently on the Player of the Match list. However, there are a few outliers, such as bowlers or all-rounders, who may have lower run totals but still earned multiple awards for exceptional performances in key moments, such as taking crucial wickets or performing in pressure situations. The overall trend suggests that while scoring runs is a significant factor in earning the award, match context and other contributions, like bowling or fielding, also play a key role in determining who gets the recognition.

* Plot 2: Wickets by Player of Match

**Figure 17 Number of Wickets by Player of Match.**

The scatter plot showing in figure 17, the number of wickets taken by different players highlights a clear pattern where the most successful bowlers, like Lasith Malinga (599), JJ Bumrah (269), Dwayne Bravo (166), RA Jadeja (512), SP Narine (612) and YS Chahal (723) have taken over 100 wickets in the IPL and consistently feature among the top performers. These bowlers are frequently recognized for their ability to take crucial wickets in high-pressure situations, often securing multiple awards for their performances. On the other hand, bowlers with fewer than 50 wickets, such as emerging players or those with fewer seasons under their belt, tend to appear less frequently at the top of the chart. The data reveals a steady decline in the frequency of wickets taken as we move down the plot, indicating that wicket-taking consistency is key to maintaining a prominent place among the league's leading bowlers. Additionally, a few outliers with significantly higher or lower wicket counts suggest that factors like injury, team strategy, or limited playing opportunities may also influence a player’s total wickets. Overall, the trend shows a strong correlation between the number of wickets taken and consistent performance across multiple seasons.

* Plot 3: Man of the Match

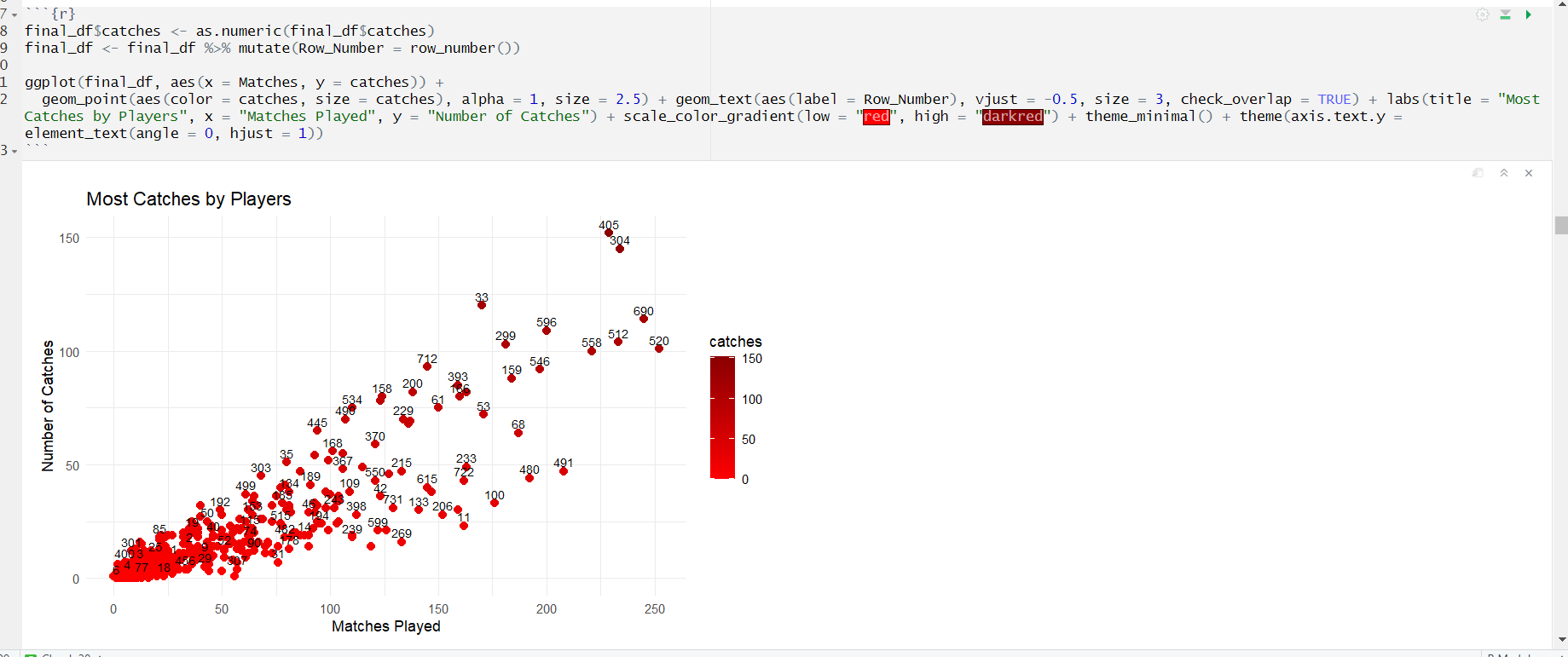
The scatter plot in figure 18, depicting the Player of the Match awards by player, with the number of matches on the x-axis and awards on the y-axis, reveals a strong pattern of dominance by a few standout players. For instance, AB de Villiers (33)*,* known for his exceptional batting and game-changing abilities, appears prominently with over 25 Player of the Match awards across more than 170 matches. Similarly, Chris Gayle (133)*,* famous for his explosive batting, has over 20 awards in roughly 140 matches. Another top performer, MS Dhoni (405)*,* recognized for his leadership and finishing skills, has 17 awards from over 200 matches, showing his consistency in crucial moments. The trend indicates that players with higher matches tend to have more awards, but there are exceptions. For example, Rashid Khan (550)*,* a bowler with fewer matches (about 90), already has 12 awards, demonstrating his impact despite limited games. On the other hand, Rohit Sharma (520)*,* with over 220 matches, has 18 awards, reflecting his key contributions in high-pressure situations. Interestingly, players like Andre Russell (42) also stand out with 15 awards in just around 100 matches, showing their ability to deliver match-winning performances regularly. The scatter plot suggests a general upward trend where more matches often lead to more awards. However, outliers like Russell and Rashid highlight the influence of impactful performances over sheer match count. This pattern emphasizes the importance of match-winning contributions, whether through explosive batting, wicket-taking, or all-round performances, in earning recognition as the Player of the Match.

**Figure 18 Player of Match by Matches Played.**

* Plot 4: Batsman Strike Rate

**Figure 19 Batsman Strike Rate.**

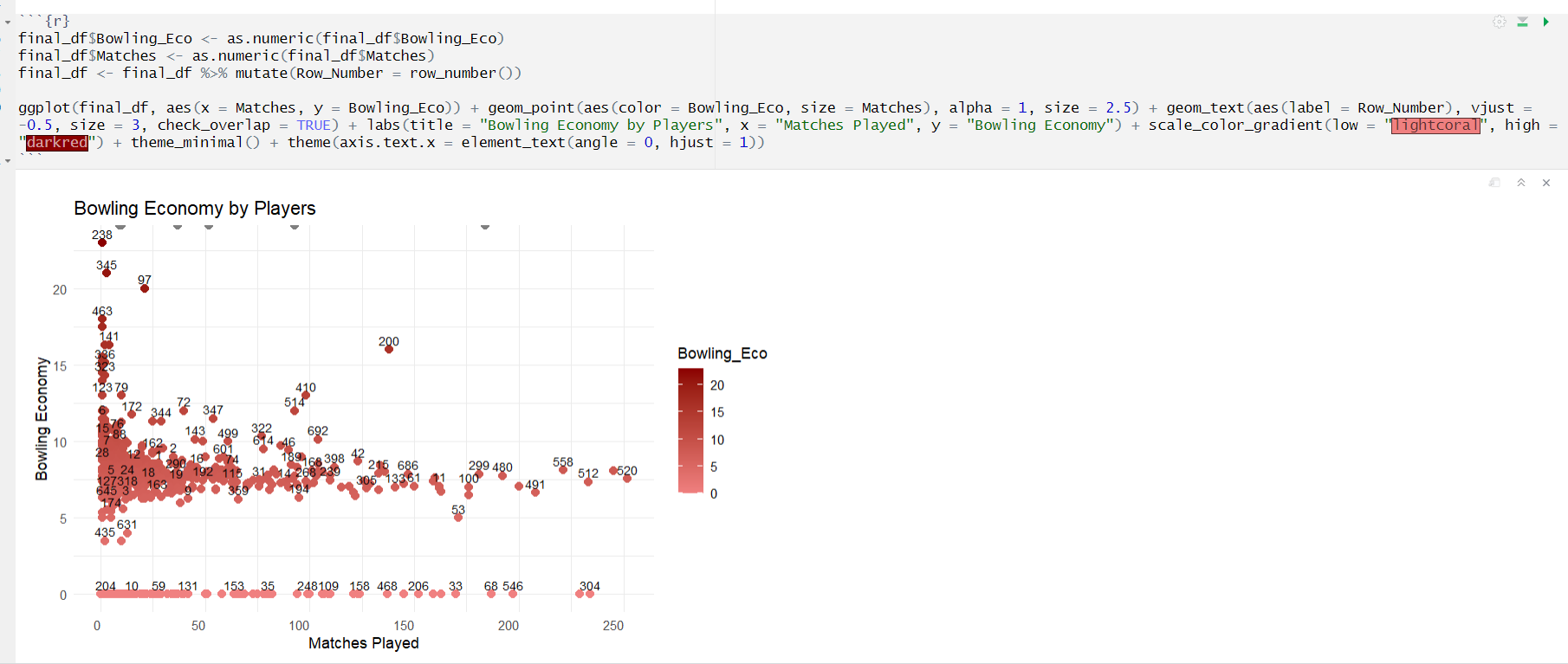
The scatter plot shown in figure 19, depicting batsmen's strike rates provides a fascinating look into the effectiveness of players in the IPL. With the number of matches played on the x-axis and the strike rate on the y-axis, the plot reveals clear trends among top-performing batsmen. For example, Andre Russell (42*)* stands out with an impressive strike rate exceeding 175 across 100 matches, showcasing his reputation as one of the most explosive batsmen in the league. Similarly, Chris Gayle (133), consistently maintains a strike rate above 150 over 140 matches, reinforcing his dominance as a power hitter. Players like AB de Villiers (33), with a strike rate of around 151 in over 170 matches, highlight a balance between aggression and consistency. On the other hand, MS Dhoni (405), recognized for his finishing abilities, maintains a respectable strike rate of approximately 137 in over 200 matches, often pacing his innings based on the game situation. Emerging talents like Rishabh Pant (534), with a strike rate above 160 in just 90 matches, show their ability to dominate in a relatively short span. The general trend in the scatter plot indicates that players with fewer matches often have higher strike rates, likely due to their specific roles as aggressive hitters or finishers. Meanwhile, players with longer careers, such as Rohit Sharma (520), who has a strike rate of around 130 across 220 matches, demonstrate steadier performances suited to anchor roles or adaptable batting styles. This plot reveals the diversity in batting approaches among IPL players. High strike rates are often associated with game-changing abilities, while moderate strike rates reflect players who balance aggression with stability. The data highlights how different players contribute uniquely to their teams’ success, emphasizing the varied strategies employed in IPL cricket.

* Plot 5: Catches by Players

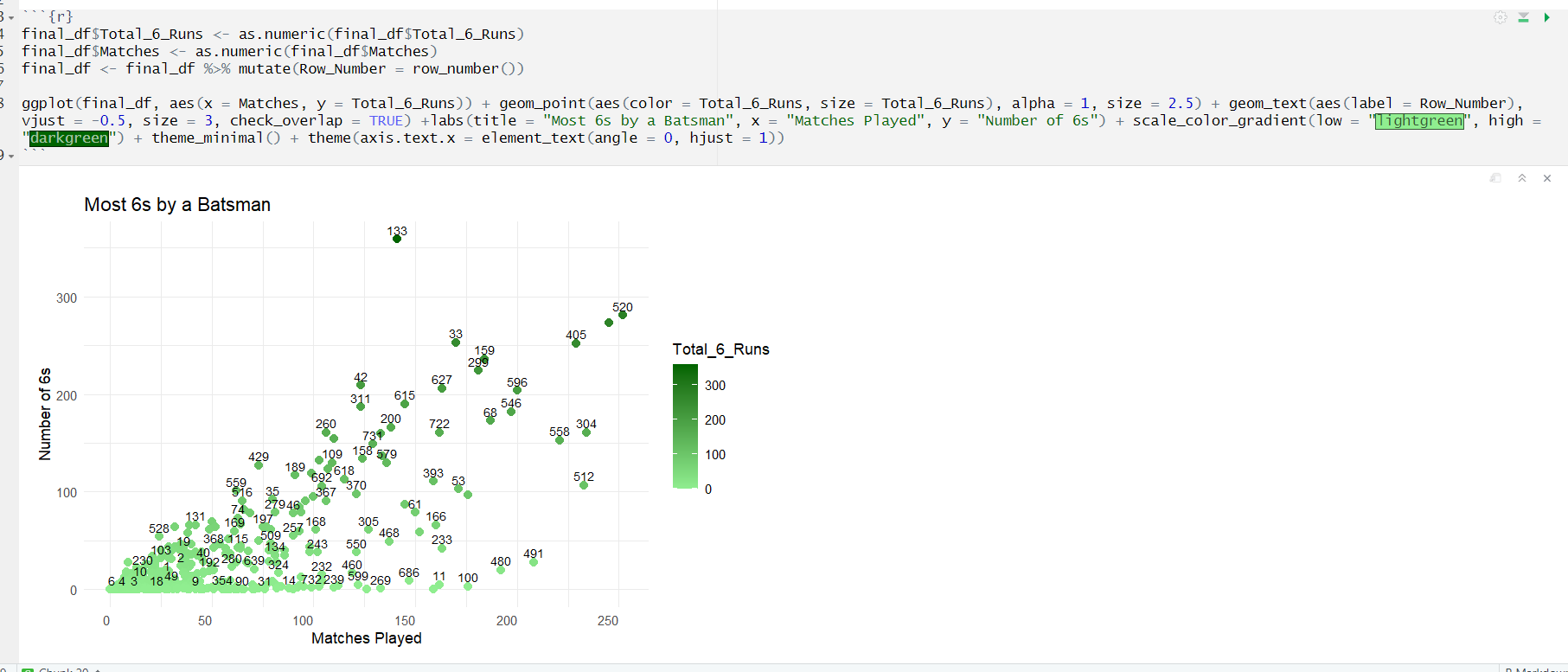
**Figure 20 Most Catches by players.**

The scatter plot shown in figure 20, IPL's most catches highlight key fielders. Suresh Raina (596) leads with 109 catches across 204 matches, showing exceptional consistency. AB de Villiers (33) follows closely with 97 catches in 184 matches. Rohit Sharma (520), with 96 catches in 227 matches, reflects steady contributions. Players like Kieron Pollard (299) and Ravindra Jadeja (512) also excel with over 85 catches each, showcasing their reliability in the field. KD Karthik (304) and MS Dhoni (405) with highest catch as they are wicketkeepers. The trend shows experienced players dominate, with consistency correlating to total matches played.

* Plot 6: Bowling Economy

The scatter plot shown in figure 21, on bowling economy reveals standout performers. Sunil Narine (612) leads with an exceptional economy of 6.63 across 148 matches. Rashid Khan (550) follows closely at 6.37 in 92 matches. Jasprit Bumrah (269), with an economy of 7.39 in 120 matches, highlights control. Bhuvneshwar Kumar (100) (7.30, 155 matches) and Lasith Malinga (599) (7.14, 122 matches) emphasize tight bowling. The trend shows experienced bowlers with consistent, low economies dominating in key matches.

**Figure 21 Bowling Economy of Players**.

* Plot 7: Most Sixes

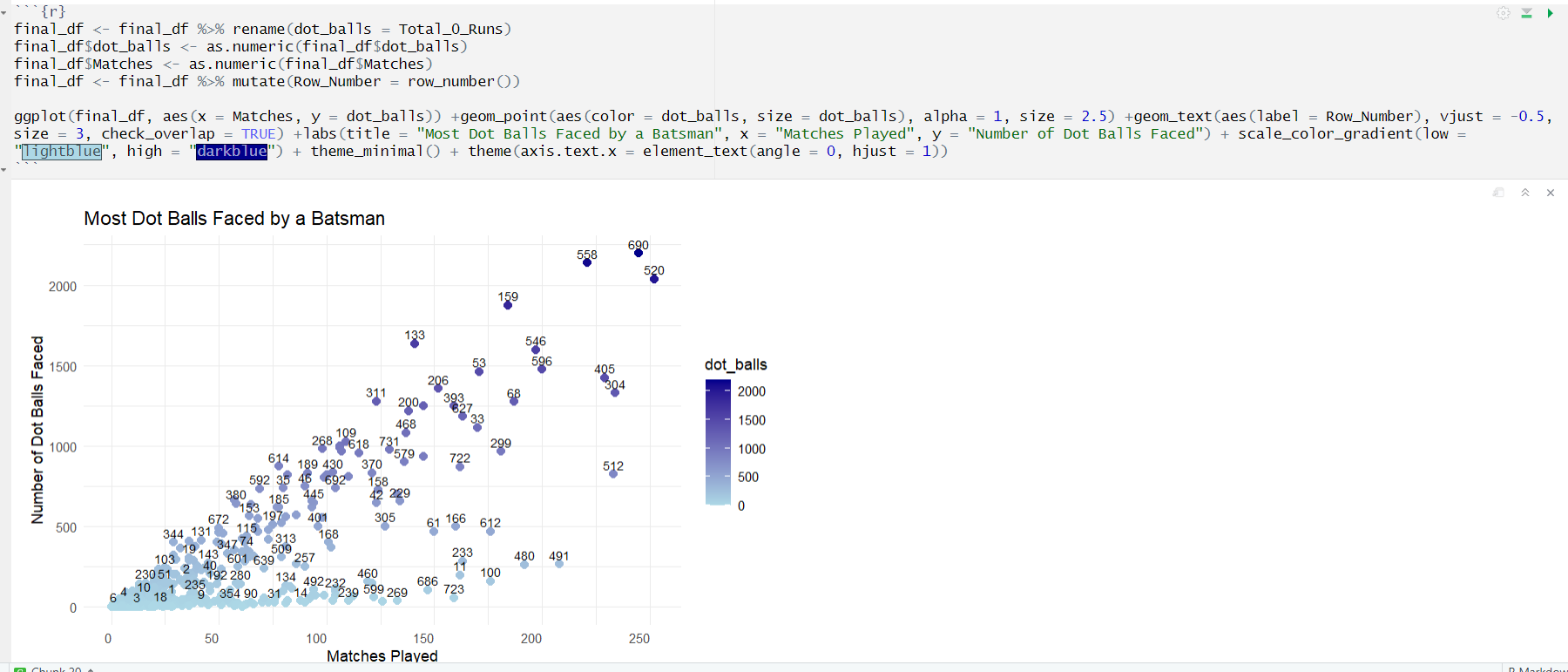
**Figure 22 Most Sixes by Players**.

The scatter plot of most sixes in IPL showcases explosive hitters that is shown in figure 22. Chris Gayle (133) dominates with 357 sixes in 142 matches, reflecting unmatched power. AB de Villiers (33) follows with 251 sixes in 184 matches. Rohit Sharma (520) (250 sixes, 227 matches) and MS Dhoni (405) (239 sixes, 234 matches) demonstrate consistency. Kieron Pollard (299) and Andre Russell (42) also feature prominently, highlighting aggressive batting trends. As number of matches increases with it number of 6s for batsman also increases as bowler make outliers here.

* Plot 8: Most Fours

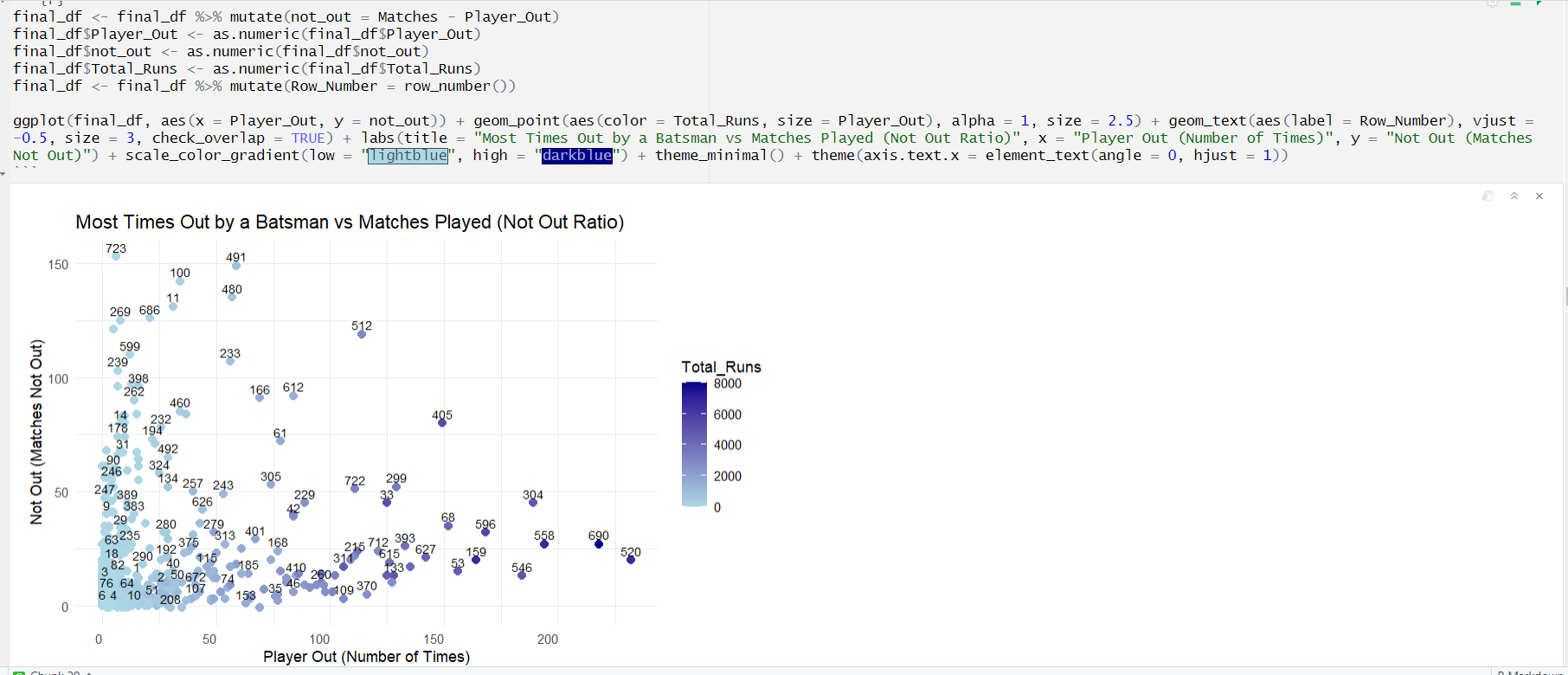
The scatter plot of most fours in IPL highlights consistent top-order batsmen shown in figure 23. **Shikhar Dhawan (558)** leads with 627 fours in 202 matches, followed by **Rohit Sharma (520)** with 550 fours in 227 matches. **Virat Kohli (690)** has 577 fours in 192 matches. **David Warner (159), AB de Villiers (33)**, and **Suresh Raina (596)** also feature prominently. The trend shows that top-order batsmen dominate, with more matches leading to more fours.

**Figure 23 Most Fours by Players.**

* Plot 9: Most Dot Balls.

**Figure 24 Most Dot Balls by Batsman**.

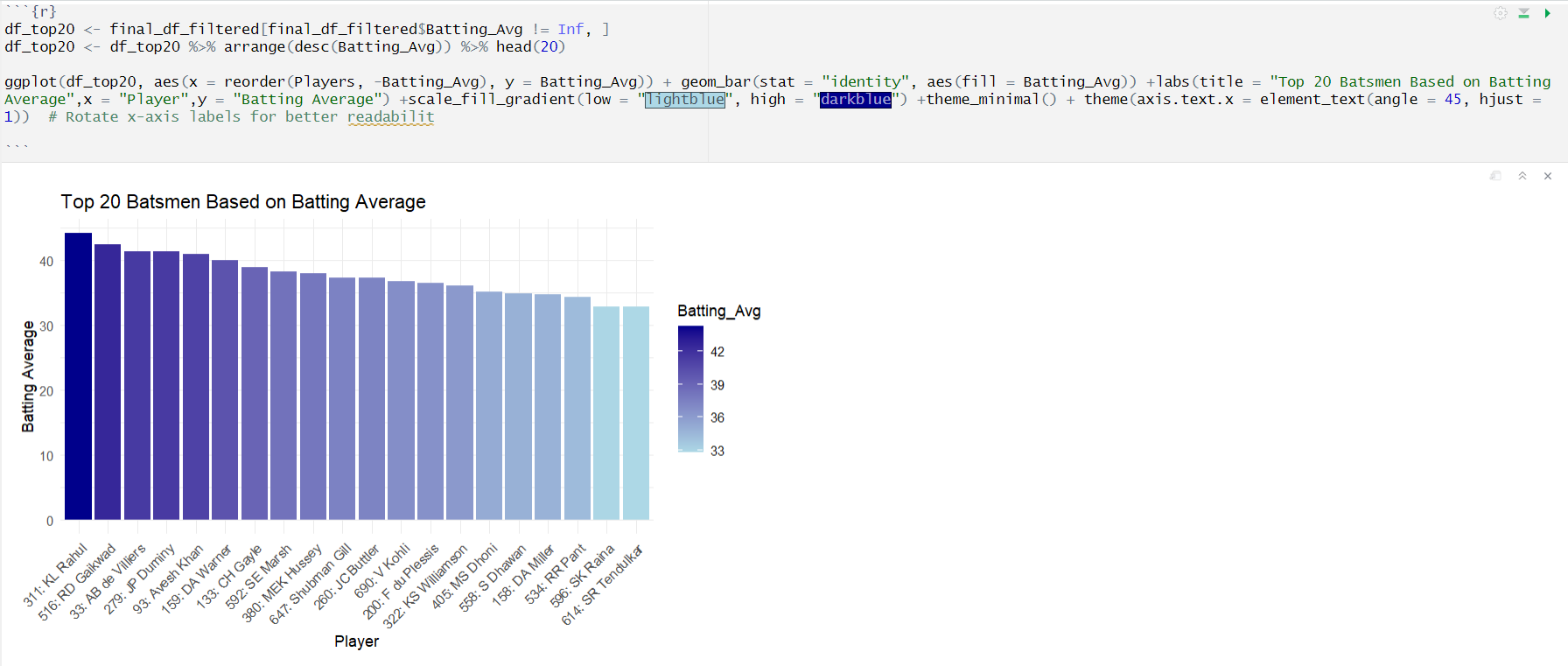
The scatter plot of Most Dot Balls Faced by a Batsman highlights players who have faced the most dot balls across IPL matches shown in figure 24. For instance, Rohit Sharma (520) consistently faces around 800 dot balls over 220 matches. Shikhar Dhawan (558) follows with approximately 750 dot balls in 200 matches, showing his steady approach. Other players like Virat Kohli (690) and MS Dhoni (405) also appear prominently, reflecting their measured batting styles. The trend shows that experienced top-order batsmen typically accumulate more dot balls, while aggressive players tend to face fewer.

* Plot 10: Ratio of Out vs Not Out

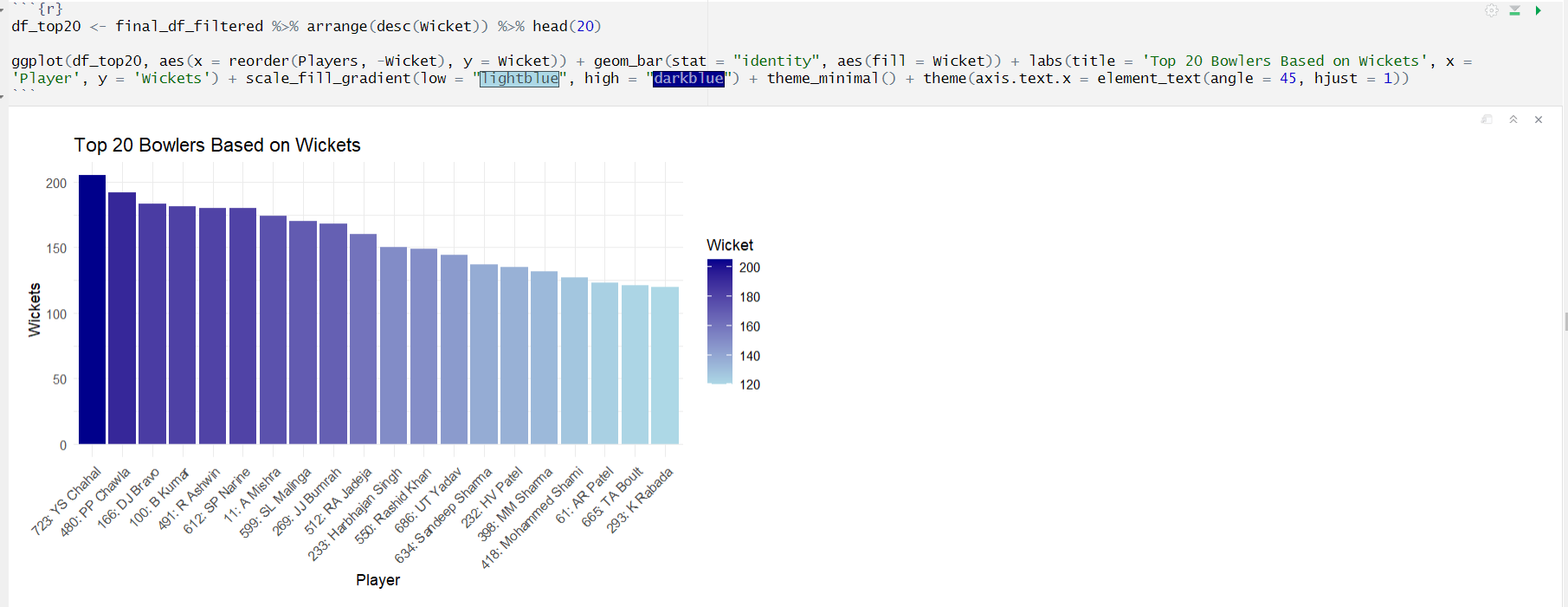
**Figure 25 Ratio of Out Vs Not Out of Players.**

The scatter plot analyzing Most Times Out by a Batsman vs Matches Played (Not Out Ratio) shows an interesting trend in the IPL shown in figure 25. Rohit Sharma (520), for example, has been out 111 times across 227 matches, reflecting his consistent top-order role. Shikhar Dhawan (558) has been out 110 times in 202 matches, demonstrating a similarly consistent pattern. Other players like Virat Kohli (690) and MS Dhoni (405) show similar trends, with more matches leading to a higher number of outs, but they’re not-out ratios are also significant. The trend indicates that experienced players tend to have more outs, but they also tend to remain not-out in several innings, showing their adaptability and match endurance.

* Plot 11: Top 20 Batsman

The Top 20 Batsmen Based on Batting Average scatter plot reveals the highest performing players in the IPL shown in figure 26. KL Rahul (311) leads with batting average of 45 across 123 matches, RD Gaiwad (516) follows with 43 across 56 mathces, AB de Villiers (33) leads with a batting average of 41.45 across 170 matches, showcasing his consistency and impact. Virat Kohli (690) follows with a 38.2 average in 192 matches. Players like David Warner (159), MS Dhoni (405), and Shikhar Dhawan (558) also feature prominently with averages above 30. The trend indicates that top-order batsmen with higher batting averages contribute significantly to team success, maintaining consistent performances across many matches**​.**

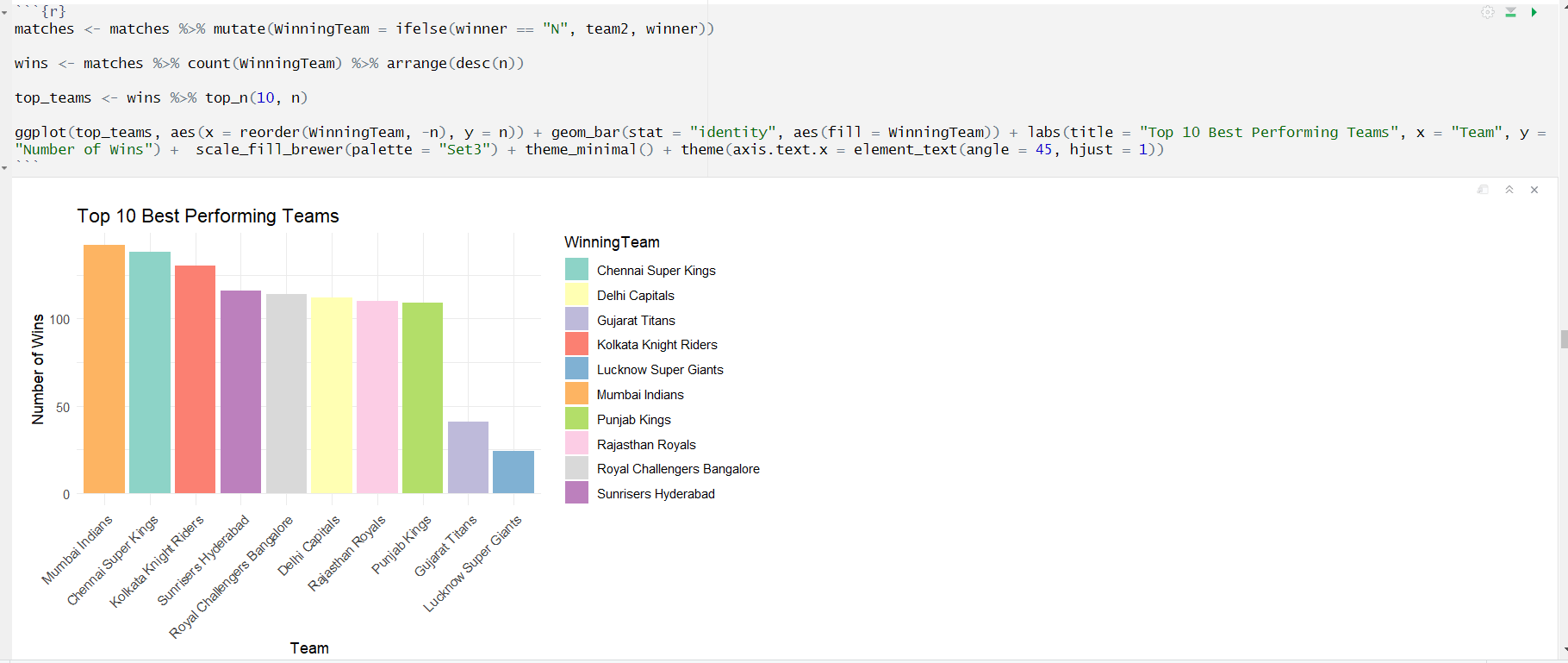
**Figure 26 Top 20 Batsman Based on Batting Average.**

* Plot 12: Top 20 Bowlers

**Figure 27 Top 20 Bowlers Based on Wickets**.

The graph shown in figure 27, of the top 20 IPL bowlers by wickets highlights consistent lead by YS Chahal (723) with 200+ wickets then dominance by players like Lasith Malinga (599) 170 wickets, Amit Mishra (11) 174 wicket, and Dwayne Bravo (166) 183 wicket. Sunil Narine (612) 180 wickets and Piyush Chawla (480) 192 wicket also exhibit exceptional consistency. These bowlers collectively represent a trend of wicket-taking prowess critical to their team's success. Key patterns show that most bowlers have economy rates between 6.5 and 8.0, highlighting their efficiency under pressure. This dataset showcases bowlers' contributions, with Harbhajan Singh (233) and Albie Morkel (257) also noted for their impactful performances.

* Plot 13: Top 10 Teams

******The Top 10 Best Performing Teams bar graph shown in figure 28, shows a clear trend in IPL performance, with teams like Mumbai Indians and Chennai Super Kings consistently leading. Mumbai Indians, with 5 IPL titles, leads the chart with the highest number of wins, followed by Chennai Super Kings. Teams such as Kolkata Knight Riders and Delhi Capitals also feature among the top, showing solid performances over the years. The chart highlights a pattern of consistent winning by these teams, underpinned by effective team management and experienced players.

**Figure 28 Top 10 Best Performing Team**.

* Plot 14: Top 10 Worst Teams

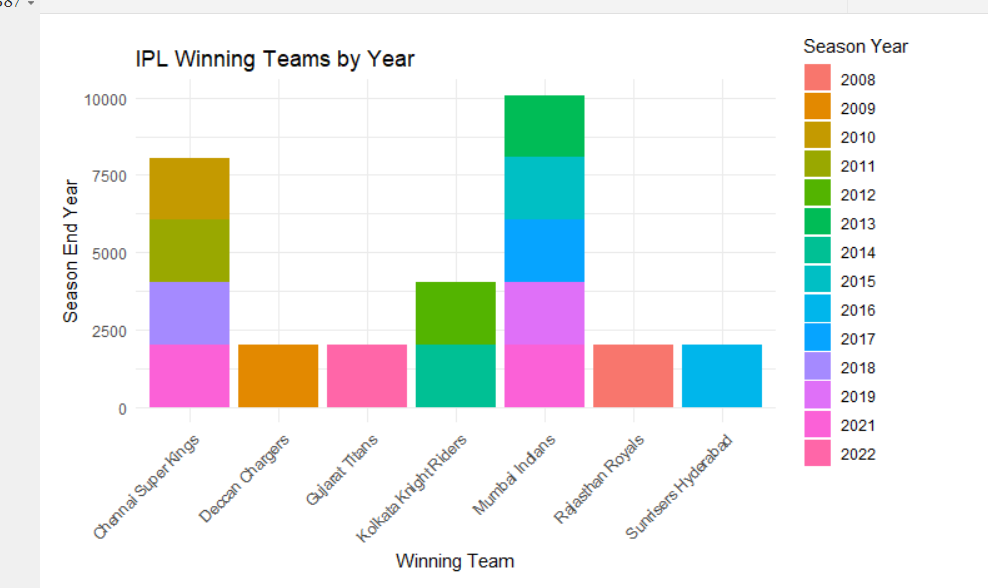
The trend suggests that these teams have faced challenges, contributing to their lower win totals across IPL seasons that is shown in figure 29. The graph for the 10 least-performing teams in IPL reveals interesting insights. Teams like Pune Warriors, with only 12 wins, and Gujarat Lions, securing 13 victories, rank lowest in performance. Rising Pune Supergiants and Kochi Tuskers Kerala, with 14 and 15 wins respectively, also feature in this category. The data shows a trend of fewer wins for teams that participated in limited seasons. On average, these teams have a win rate below 40%. This underlines how participation longevity affects overall team success. Punjab kings and Rajasthan royals are the teams who have participated in the tournament much more consistently and still feature in this list. The performance of these teams must be questioned more as compared to the rest of the teams in the least who have played not more than 2 seasons.



**Figure 29 Top 10 Worst Performing Team.**

* Plot 15: IPL Teams with IPL Cup

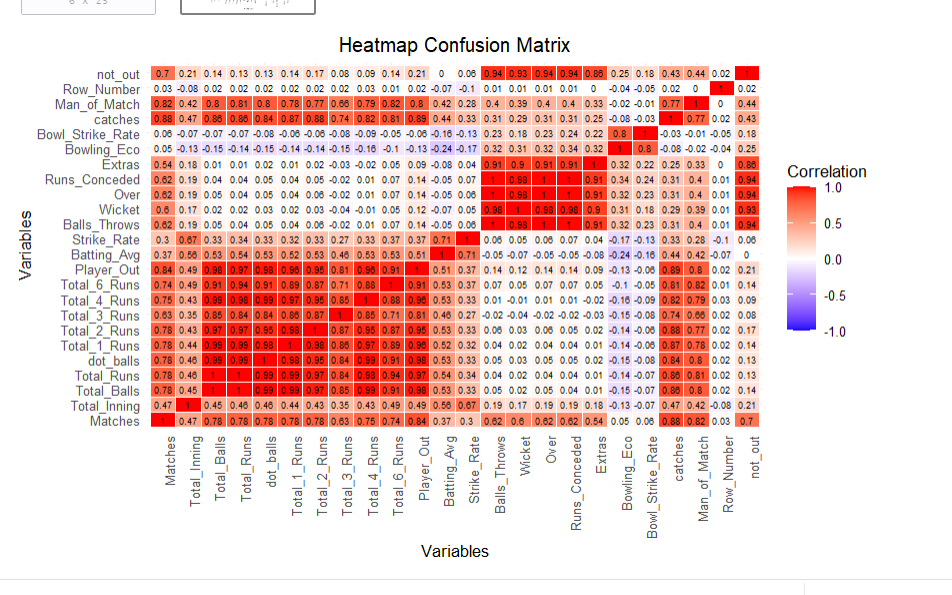
The IPL winning teams by year demonstrate a clear dominance of certain franchises are shown in figure 30. Mumbai Indians, with 5 championships, and Chennai Super Kings, securing 4 titles, lead the chart. Kolkata Knight Riders follow with 2 wins. Notable patterns include repeated victories by strong teams and a gap in wins for emerging franchises. For example, teams like Rajasthan Royals and Sunrisers Hyderabad, each with 1 title, reflect a more competitive yet sporadic trend. The graph captures year-wise wins, emphasizing consistent performances by top teams.



**Figure 30 IPL Winning Teams**.

* Plot 16: Heatmap

The figure 31 shown below is a heatmap confusion matrix that highlights the correlation between various variables related to player and match statistics in the IPL dataset. This matrix uses a color gradient, with darker red shades indicating a strong positive correlation (closer to +1) and blue shades reflecting negative correlation (closer to -1). The diagonal line with deep red boxes represents perfect correlations of each variable with itself. Key features of the heatmap include a high correlation between `Total\_Balls` and `dot\_balls` (0.99), indicating that players with higher ball participation tend to face more dot balls. Similarly, `Total\_Runs` shows strong positive correlations with `Total\_4\_Runs` (0.95) and `Total\_6\_Runs` (0.84), suggesting that boundary-hitting players significantly contribute to run accumulation. `Batting\_Avg` shows a moderate positive correlation with `Strike\_Rate` (0.42), highlighting that consistency in scoring impacts a player's ability to maintain a higher strike rate. Another notable observation is the correlation between `Matches` and `Total\_Inning` (0.94), which suggests that players involved in more matches naturally accumulate more innings. Conversely, `Wicket` and `dot\_balls` exhibit a weaker correlation, emphasizing that bowlers with high dot ball counts may not always convert them into wickets. Interestingly, variables such as `Man\_of\_Match` and `Extras` show minimal correlation with most others, indicating that these are less predictable based on the given variables. This analysis underpins the intricate dynamics between batting, bowling, and fielding statistics, highlighting actionable insights for predictive modeling and performance analysis.



**Figure 31 Heatmap Confusion Matrix.**

# SECTION VIII: ALGORITHMS

Choosing the right algorithms is crucial for building a reliable and accurate IPL winning team prediction model. Each algorithm used in this project plays a unique role in analyzing and interpreting the diverse dataset, which includes team performance, player statistics, and match conditions. By leveraging a combination of robust, interpretable, and efficient algorithms, the project ensures a comprehensive exploration of predictive techniques. Below is an overview of the key algorithms used and their significance in this context.

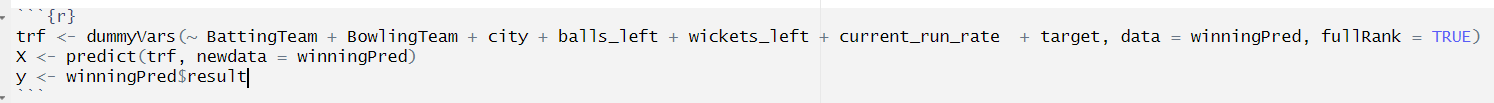
In this project, machine learning algorithms were at the core of predicting the winning team for IPL matches. These algorithms were selected based on their ability to handle the complexity of cricket match data, which involves diverse factors like team performance, player statistics, match venue, and toss outcomes. The aim was to find patterns and relationships in the data that could accurately forecast match results.

The following algorithms were implemented and evaluated:

1. Random Forest: This ensemble learning method combines multiple decision trees to improve prediction accuracy and reduce overfitting. By aggregating the outputs of multiple trees, it provides a balanced and robust model, making it a key contender for handling the varied and noisy nature of cricket data.
2. Logistic Regression: A simple yet effective algorithm, it served as a baseline for our predictions. Logistic regression estimates the probability of a win or loss by analysing key features like team score, run rate, and match venue.
3. Decision Tree: Known for its interpretability, the decision tree algorithm identifies significant splits in the data to predict outcomes. It helps map out how different factors—such as the toss decision or a star player's performance—affect the match result.
4. Naive Bayes: Based on Bayes' Theorem, this algorithm assumes independence between features, making it lightweight and fast. While simple, it proved effective in analyzing categorical data such as team names, match types, and venue details.

Each algorithm was tested on historical IPL data using performance metrics like accuracy, precision, and recall. After multiple iterations and fine-tuning, the models were compared to select the one offering the best combination of efficiency and accuracy. By leveraging these diverse algorithms, this project demonstrates the potential of machine learning to bring data-driven insights into the world of cricket, turning complex match scenarios into actionable predictions.

Before moving to Algorithm part, let’s see about split data into parts training and testing set. Project uses some dummy variables. The dummyVars() in R, part of the caret package, is used to create dummy or indicator variables for categorical data. It helps convert categorical features into numeric representations, which are essential for many machine learning models. The function allows flexibility, enabling users to include or exclude the original variable, handle missing values, and create clean, structured outputs. For example, a categorical column with three levels (A, B, C) is transformed into three binary columns, each representing one level. This ensures models can process categorical data effectively while preserving its underlying information.

In figure 32 below code snippet, the dummyVars function is used to generate dummy variables for the categorical columns BattingTeam, BowlingTeam, city in the winningPred dataset. By setting fullRank = TRUE, the function avoids multicollinearity by creating one less dummy variable for each categorical feature. The transformed data is then used with the predict function to produce a new dataset, X, containing only the encoded variables. The target variable y, representing the match result, is extracted from the result column of winningPred, preparing both features (X) and labels (y) for further analysis or modeling.

**Figure 32 Dummy Variables.**

When building machine learning models, splitting the dataset into training and testing subsets is a common practice to evaluate model performance. However, in certain cases, splitting the data may not be necessary, such as when working with a small dataset or when the goal is to analyse the entire data for insights without predictive evaluation. Instead, cross-validation or other validation techniques can be used to assess the model.

When splitting data, the ratio of the split is crucial for balancing the training and testing datasets. Common splits include 60-40, 70-30, and 80-20, which represent the percentage of data allocated for training and testing respectively. For instance, in a 60-40 split, 60% of the data is used for training the model, and the remaining 40% is used for testing. This split works well for smaller datasets, as it reserves a significant portion for evaluation. The 70-30 split is a more balanced option for most datasets, ensuring the model has sufficient data for training while leaving a fair amount for testing. The 80-20 split is often preferred for larger datasets, as it maximizes the data used for training, improving the model’s learning potential while still leaving enough data for evaluation.

Choosing the right split depends on the size of the dataset, the complexity of the model, and the nature of the problem. Larger training datasets generally improve model accuracy, but the testing set must remain representative to ensure reliable performance assessment.

For project, the split was choice based on several factors of the model, factors such as Accuracy, Precision, Recall or Sensitivity, Specificity and F-Score. Given below are tables 1 (Random Forest Model), 2 (Logistic Regression Model), 3 (Naïve Bayes Model) and 4 (Decision Tree Model) in different split ration where each models Accuracy, Precision, Recall or Sensitivity, Specificity and F-Score are measured and put in it to take best split ratio for the model to perform well.

1. Random Forest:

**Table 1 performance metrics of Random Forest on Different Split Ratio.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sr No. | Split Ratio | Accuracy | Precision | Sensitivity  Or Recall | Specificity | F-Score |
| 1. | 60-40 | 0.9719 | 0.9786 | 0.9649 | 0.9797 | 0.9714 |
| 2. | 65-35 | 0.9727 | 0.9799 | 0.9656 | 0.9800 | 0.9726 |
| 3. | 70-30 | 0.9736 | 0.9802 | 0.9663 | 0.9803 | 0.9732 |
| 4. | 75-25 | 0.9730 | 0.9801 | 0.9659 | 0.9802 | 0.9729 |
| 5. | 80-20 | 0.9722 | 0.9798 | 0.9652 | 0.9799 | 0.97244 |

1. Logistic Regression:

**Table 2 performance metrics of Logistic Regression on Different Split Ratio.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sr No. | Split Ratio | Accuracy | Precision | Sensitivity  Or Recall | Specificity | F-Score |
| 1. | 60-40 | 0.7893 | 0.8117 | 0.7643 | 0.8118 | 0.7872 |
| 2. | 65-35 | 0.7906 | 0.8133 | 0.7657 | 0.8129 | 0.7887 |
| 3. | 70-30 | 0.7913 | 0.8140 | 0.7661 | 0.8141 | 0.7901 |
| 4. | 75-25 | 0.7911 | 0.8136 | 0.7660 | 0.8137 | 0.7890 |
| 5. | 80-20 | 0.7909 | 0.8129 | 0.7658 | 0.8130 | 0.7886 |

1. Naïve Bayes:

**Table 3 performance metrics of Naïve Bayes on Different Split Ratio.**

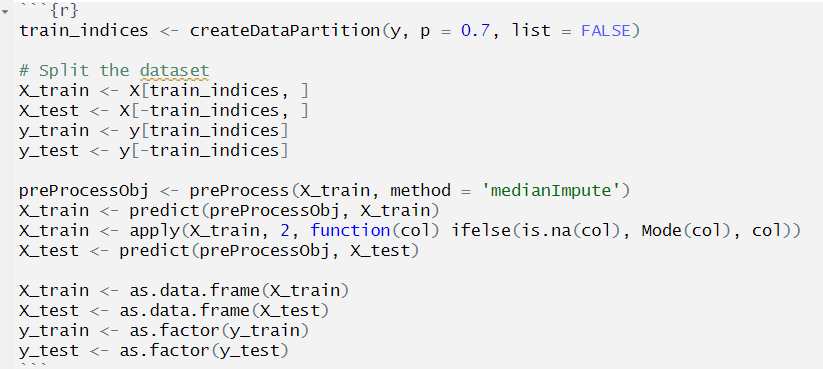
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sr No. | Split Ratio | Accuracy | Precision | Sensitivity  Or Recall | Specificity | F-Score |
| 1. | 60-40 | 0.5529 | 0.2397 | 0.9013 | 0.2398 | 0.3786 |
| 2. | 65-35 | 0.5567 | 0.2466 | 0.8985 | 0.2467 | 0.3869 |
| 3. | 70-30 | 0.5594 | 0.2574 | 0.8933 | 0.2575 | 0.3996 |
| 4. | 75-25 | 0.5586 | 0.2502 | 0.8997 | 0.2503 | 0.3915 |
| 5. | 80-20 | 0.5570 | 0.2416 | 0.9032 | 0.2417 | 0.3812 |

1. Decision Tree:

**Table 4 performance metrics of Decision Tree on Different Split Ratio.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sr No. | Split Ratio | Accuracy | Precision | Sensitivity  Or Recall | Specificity | F-Score |
| 1. | 60-40 | 0.7528 | 0.7492 | 0.7250 | 0.7493 | 0.7369 |
| 2. | 65-35 | 0.7539 | 0.7588 | 0.7301 | 0.7589 | 0.7441 |
| 3. | 70-30 | 0.7547 | 0.7698 | 0.7380 | 0.7699 | 0.7535 |
| 4. | 75-25 | 0.7541 | 0.7624 | 0.7353 | 0.7625 | 0.7486 |
| 5. | 80-20 | 0.7533 | 0.7516 | 0.7299 | 0.7517 | 0.7405 |

For a standard model to be considered a good fit, key performance metrics such as Accuracy, Precision, Recall (Sensitivity), F-Score, and Specificity must all be high. This ensures the model not only makes correct predictions but also balances false positives and false negatives, making it reliable and effective.

In this project, the optimal split ratio is crucial for achieving high values across all these metrics. At a **70-30 split ratio**, the model tends to show strong performance across all factors. This ratio strikes a balance between having enough data to train the model effectively while retaining sufficient test data for a robust evaluation. Accuracy is high, meaning the model predicts the outcomes (wins or losses) correctly for most matches. Precision is also high, indicating that when the model predicts a win, it's often correct, minimizing false positives. Recall(Sensitivity) is strong, ensuring the model accurately identifies the majority of actual wins, minimizing false negatives. F-Score, which balances both precision and recall, reflects the model’s overall effectiveness in making predictions. Specificity is high, meaning the model correctly predicts losses (true negatives) when they occur, improving reliability. Thus, at the **70-30 split ratio**, the model performs consistently across all critical metrics, making it a well-suited fit for predicting IPL match outcomes.

**Figure 33 Data Splitting and Training & Testing Set.**

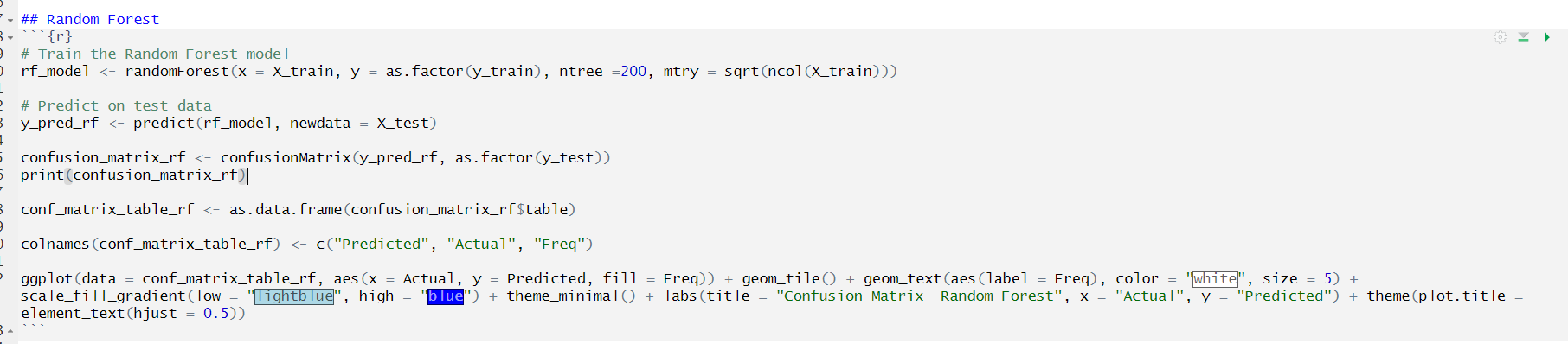
In figure 33, code snippet, the dataset is split into training and testing sets using a **70-30 split ratio**. First, createDataPartition is used to create indices that partition 70% of the data for training and 30% for testing. This ensures that the model is trained on a substantial portion of the data while leaving a reasonable amount for evaluation. Next, missing values in the training set (X\_train) are handled using median imputation with the preProcess function. This technique replaces missing values with the median of each feature, ensuring that the model receives complete and meaningful data. After imputation, any remaining NA values are filled using the mode (most frequent value) of each column. The test set (X\_test) is then processed similarly to ensure consistency. Finally, the data is converted into data frames and the target variable (y\_train and y\_test) is converted into factors to prepare it for classification modeling. This preprocessing step ensures the dataset is clean, consistent, and ready for model training and testing.

Now, let look one by one to each Algorithms

1. Random Forest:

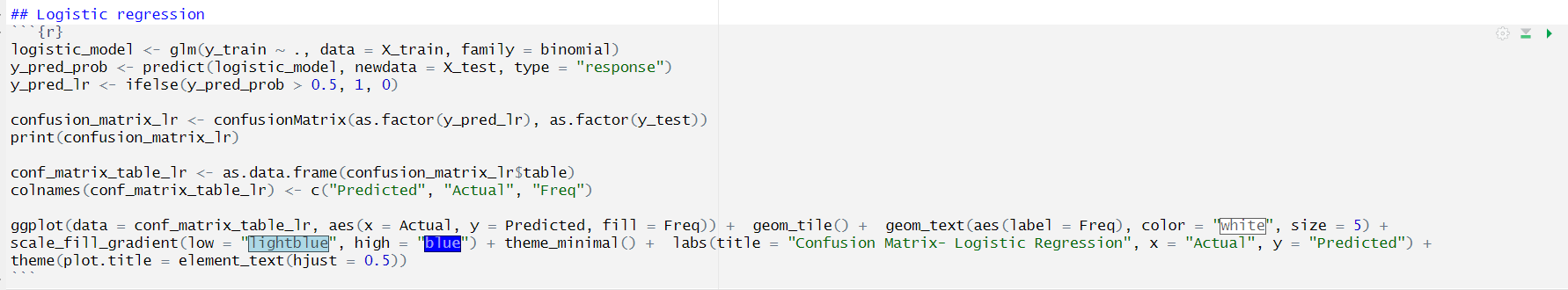
In the IPL winning team prediction project, **Random Forest** is applied as a powerful machine learning algorithm due to its ability to handle complex datasets and generate accurate predictions. The Random Forest algorithm works by building multiple decision trees during the training phase, with each tree receiving a random subset of features and data. By aggregating the outcomes from all trees, Random Forest reduces the risk of overfitting and improves the overall accuracy.

In the project, the model is trained using a large dataset of IPL match features shown in figure 34, with parameters like ntree = 200 (the number of trees) and mtry = sqrt(ncol(X\_train)) (the number of features considered at each split). The **training set (X\_train)** is used to train 200 trees, each learning different aspects of the data, making the model robust and reliable. After training, the model predicts match outcomes on unseen test data (**X\_test**), and its performance is evaluated using a **confusion matrix.** This matrix provides a summary of how well the model predicted actual match results, offering insight into its accuracy. For example, key insights from the project include high precision in predicting the winning team, and the **Random Forest**'s ability to manage large datasets while capturing crucial patterns that influence match outcomes, such as team composition, performance, and match conditions. Overall, Random Forest proves highly effective for predicting IPL results due to its ensemble learning method that minimizes error and maximizes predictive power.

**Figure 34 Random Forest Model**.

1. Logistic Regression:

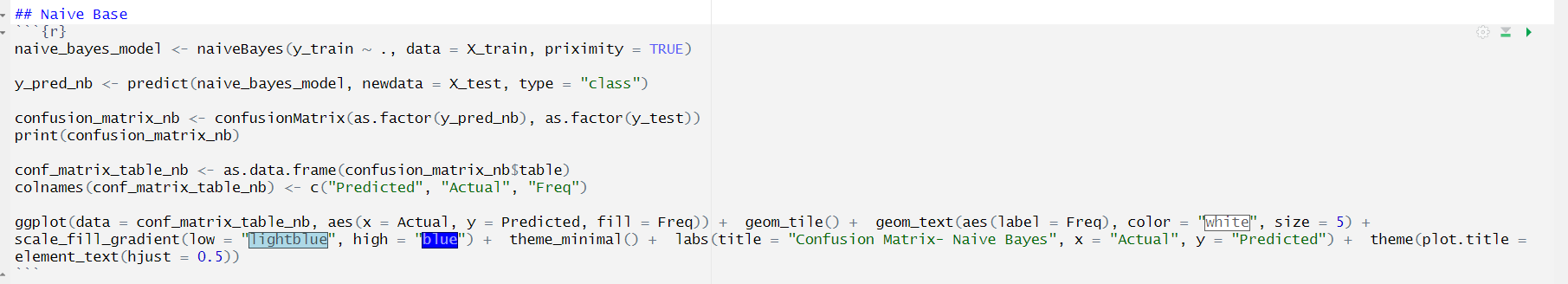
In the IPL winning team prediction project, **Logistic Regression** is used as a classification technique to predict the likelihood of a team winning a match. Logistic regression works by modeling the relationship between the input features (such as team performance, batting, and bowling stats) and the binary outcome (win or lose). The model is trained using a dataset where the target variable (y\_train) represents the match outcome, and the independent variables (X\_train) contain features like team composition, performance metrics, and match conditions.

In this project, the logistic model is trained using the glm function with the **binomial** family to handle the binary nature of the target variable. After training, the model generates **probabilities** for the test data (**X\_test**) that indicate the likelihood of a team winning the match. If the probability exceeds a certain threshold (in this case, **0.5**), the prediction is classified as a win (**1**), otherwise, it's classified as a loss (**0**) as shown in figure 35. The model's performance is evaluated using a **confusion matrix**, which helps to visualize the accuracy of the predictions by comparing the predicted outcomes with the actual results. This matrix provides key insights, such as how many matches were correctly predicted as wins or losses.

**Figure 35 Logistic Regression Model.**

1. Naïve Bayes:

In the IPL winning team prediction project, **Naive Bayes** is used as a classification model that predicts whether a team will win or lose based on certain match features. Naive Bayes is built on the assumption that all features (like batting, bowling, and team performance) are conditionally independent of each other given the outcome, even though this assumption may not always hold in real-life sports scenarios. Despite this simplification, Naive Bayes can still offer competitive results, especially in situations where computation speed and simplicity are key factors.

In this project, the **Naive Bayes model** is trained using the naiveBayes function with the training data (**X\_train** and **y\_train**), where it learns from previous match outcomes and associated features as shown in figure 36. The model then predicts the result of future matches by classifying the test data (**X\_test**) into win or lose categories. The model outputs a class prediction for each match, and these predictions are compared with the actual match results to evaluate the model’s performance using a **confusion matrix**. The confusion matrix visualizes the model's accuracy, displaying the number of correctly and incorrectly predicted outcomes. While Naive Bayes may not capture complex interactions between variables as effectively as more advanced models, it remains a fast and computationally efficient algorithm, making it a practical option for large datasets with numerous features in IPL prediction tasks.

**Figure 36 Naïve Bayes Model.**

1. Decision Tree:

In the IPL winning team prediction project, a **Decision Tree** model is used to classify whether a team will win or lose based on features like batting performance, bowling stats, and match conditions. The model works by splitting the dataset into subsets based on feature values, creating branches that lead to a decision node for predicting outcomes. At each step, the Decision Tree chooses the feature that best separates the data into the desired categories (win or lose), maximizing prediction accuracy.

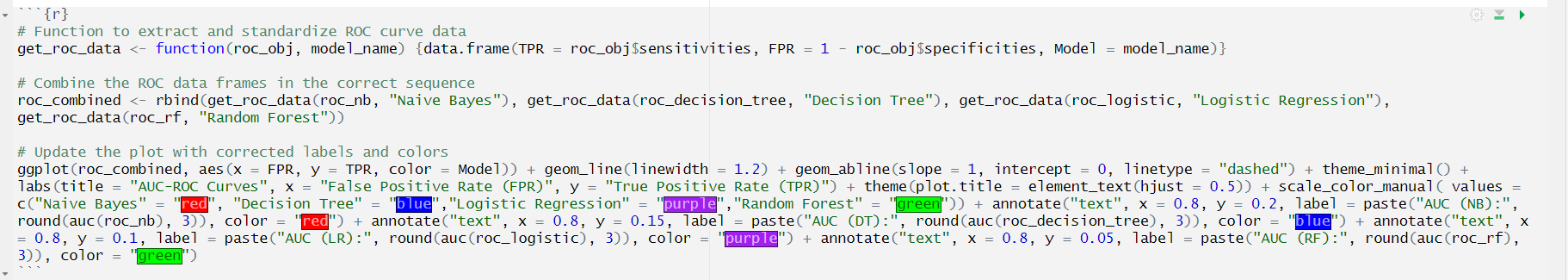
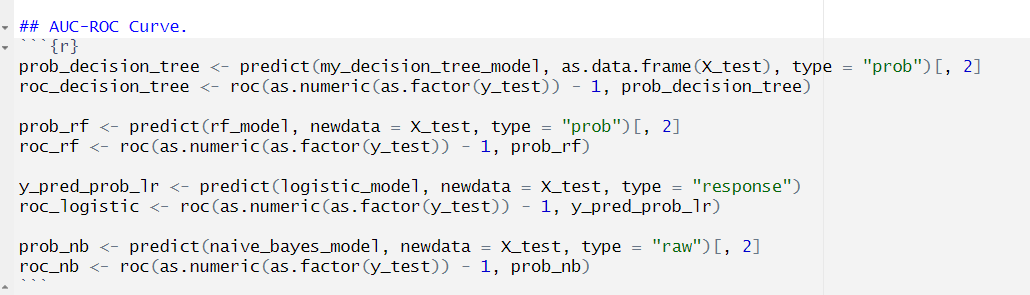
In this project, the Decision Tree was trained using the rpart function, with control parameters set to manage the depth of the tree (maxdepth = 20) and the minimum number of observations in a node (minbucket = 3) as shown in figure 37. The model learned from the training data (**X\_train** and **y\_train**) and was evaluated using the test data (**X\_test**) to predict match outcomes. The predictions were then compared to actual results to calculate the model's accuracy using a **confusion matrix,** which details how well the Decision Tree classified the matches. While Decision Trees are easy to interpret and provide clear decision paths, they tend to overfit, meaning they can perform very well on the training data but may struggle with unseen data. This is a known limitation in the context of IPL prediction, where overfitting can reduce the model’s generalization ability. However, with proper tuning, Decision Trees can still offer valuable insights and predictions for IPL match outcomes.

**Figure 37 Decision Tree Model.**

1. AUC-ROC Curve:

The AUC-ROC curve is a graphical representation used to evaluate the performance of classification models. The ROC (Receiver Operating Characteristic) curve plots the True Positive Rate (TPR) or sensitivity on the y-axis against the False Positive Rate (FPR) on the x-axis for varying classification thresholds. The Area Under the Curve (AUC) is a single scalar value that quantifies the overall ability of the model to distinguish between positive and negative classes. An AUC of 1 represents a perfect model, while an AUC of 0.5 indicates performance no better than random guessing. The ROC curve works by calculating how well the model predicts positive and negative cases at different thresholds, with steep curves close to the top-left corner indicating strong model performance. This method is particularly useful when the dataset is imbalanced, as it focuses on the trade-off between sensitivity and specificity without being influenced by class distribution.

As shown in figure 38, For each model (my\_decision\_tree\_model, rf\_model, logistic\_model, and naive\_bayes\_model), the probability of a positive class (type = "prob" or type = "response") is predicted on the test dataset (X\_test). The roc() function computes the ROC curve for each model by comparing the true class labels (y\_test, converted to numeric) and the predicted probabilities. get\_roc\_data() extracts True Positive Rate (TPR) and False Positive Rate (FPR) from the ROC object, organizing the data into a standardized format for plotting. The rbind() function combines the extracted ROC data for all models into a single data frame (roc\_combined), associating each data point with its respective model.



**Figure 38 AUC-ROC Curve.**

# SECTION IX: RESULT

The IPL Winning Team Prediction project yielded promising results, showcasing the effectiveness of various machine learning algorithms in predicting match outcomes. Each model was trained and evaluated on a well-prepared dataset, split into training and testing sets, and their performance was assessed based on key metrics such as accuracy, precision, recall, F1-score, and specificity. The results not only demonstrate the predictive power of the models but also highlight the importance of feature selection, data preprocessing, and split ratios in ensuring accurate and reliable predictions. In this section, we discuss into the detailed performance of each algorithm, compare their strengths and limitations, and discuss the overall success of the prediction model in forecasting IPL match winners.

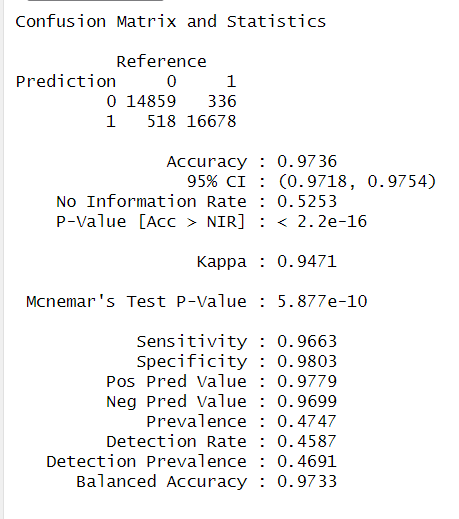
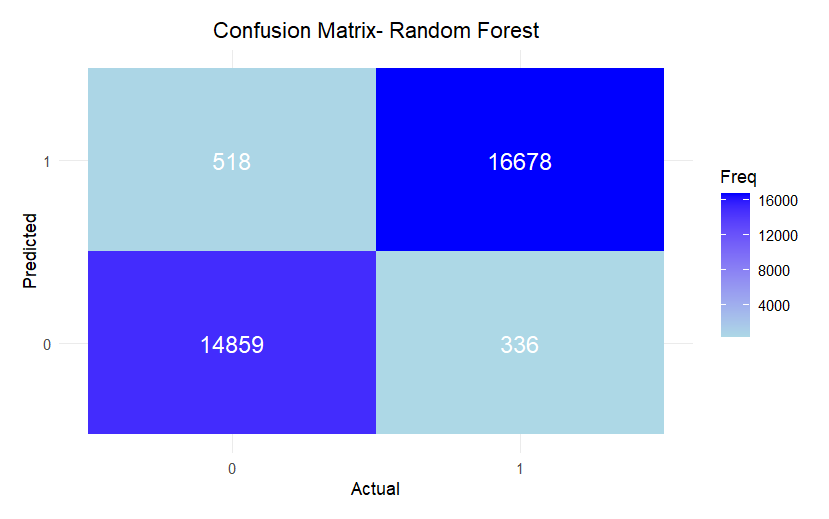
First lets now some terms like 95% CI, No Information Rate, P-Value [Acc > NIR], Kappa, McNemar’s Test P-Value, Pos Pred Value Neg Pred Value Prevalence Detection Rate Detection Prevalence Balanced Accuracy.

* 95% CI: A 95% confidence interval represents a range within which we can be 95% confident the true value such as model accuracy lies. It’s a way of expressing the uncertainty around the estimated value. In the context of model accuracy, a 95% CI shows the range of accuracies you would expect if you repeated the experiment multiple times with different data samples.
* No Information Rate (NIR): The No Information Rate refers to the accuracy that would be achieved by simply predicting the majority class in the data. If your model's accuracy exceeds the NIR, it means the model is doing better than a trivial classifier that always predicts the most common class.
* P-Value [Acc > NIR]: The p-value is a statistical measure used to determine the significance of a result. It represents the probability of obtaining the observed results under the assumption that the null hypothesis is true. A small p-value (typically less than 0.05) indicates that the observed result is unlikely to have occurred by chance, thus leading to the rejection of the null hypothesis. In the case of model evaluation, a very low p-value (such as < 2.2e-16) indicates that the model's accuracy is significantly better than random guessing or the null hypothesis being that the model performs no better than random chance.
* Kappa: Kappa, or Cohen's Kappa, is a statistical measure that evaluates the agreement between predicted and actual classes, adjusting for agreement that could happen by chance. The value of kappa ranges from -1 to 1. A kappa value of 1 indicates perfect agreement, while 0 means no agreement beyond chance. It helps in assessing how well the model performed beyond just accuracy, particularly in imbalanced datasets.
* McNemar’s Test P-Value: McNemar's test is a statistical test used to compare the performance of two models or classifiers based on paired nominal data, especially when the goal is to detect differences in prediction errors (misclassifications). In model evaluation, McNemar's test assesses whether the difference in error rates between the two models is statistically significant. A very low p-value suggests that the differences in the misclassification errors are statistically significant, meaning the model's performance is significantly different from random or other models it is compared with. This test is often used when comparing two classification models to see if one significantly outperforms the other.
* Pos Pred Value (Positive Predictive Value, PPV): The Positive Predictive Value is the proportion of positive results that are true positives. It tells you how many of the predicted positive cases are actually positive.
* Neg Pred Value (Negative Predictive Value, NPV): The Negative Predictive Value is the proportion of negative results that are true negatives. It indicates how many of the predicted negative cases are actually negative.
* Prevalence: Prevalence is the proportion of the actual positive cases in the dataset. It gives you an idea of how common the positive class is in your data.
* Detection Prevalence: Detection Prevalence refers to the proportion of instances that were predicted to be positive. It tells you how often the model predicted the positive class.
* Balanced Accuracy: Balanced accuracy is the average of sensitivity (true positive rate) and specificity (true negative rate). Balanced accuracy is useful in imbalanced datasets to ensure that both classes are equally considered in the performance metric.

Now, let look one by one to each Algorithm’s Result

1. Random Forest:

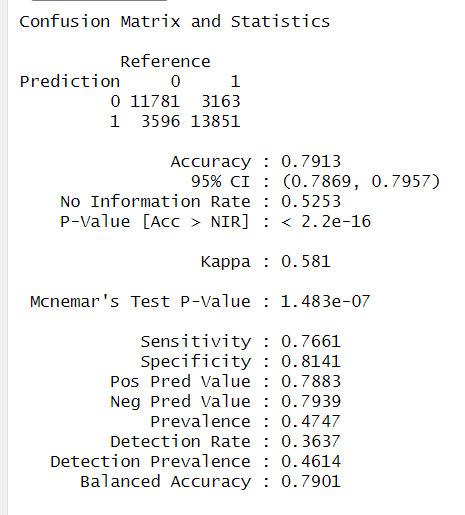
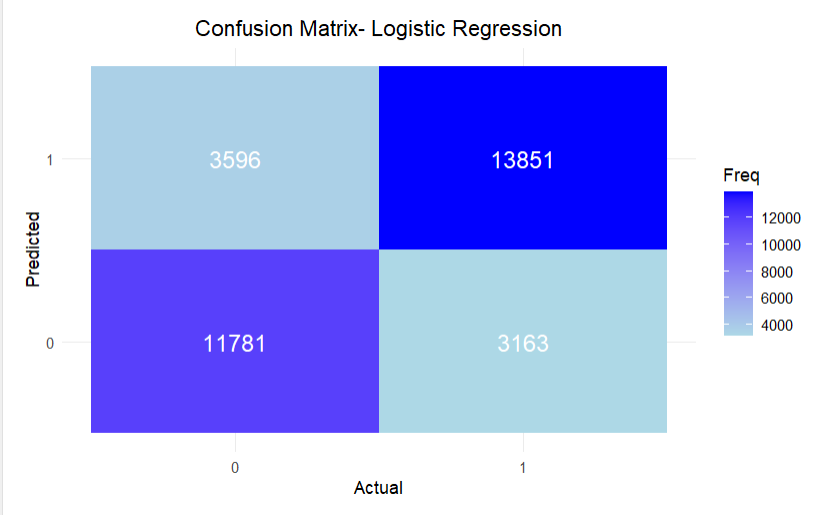
The performance evaluation of a Random Forest model used to predict IPL winning teams as shown figure 39. The model achieved an impressive accuracy of 97.36%, which indicates that it correctly predicted the outcome for most teams. The 95% Confidence Interval [0.9718, 0.9754] suggests that we can be 95% confident that the model’s accuracy falls within this range. The No Information Rate (52.53%) represents the accuracy one would achieve by always predicting the most frequent outcome, but the model clearly surpasses this with its higher accuracy, as confirmed by the very low P-value (<2.2e-16), which shows that this improvement is statistically significant. The Kappa score of 0.9471 measures agreement between predicted and actual values, factoring out chance, and a value near 1 indicates the model’s predictions are highly reliable. The McNemar's Test P-Value (5.877e-10) suggests there’s a significant difference between the model's accuracy across the two classes. Metrics like Sensitivity (96.63%) and Specificity (98.03%) show the model’s effectiveness in predicting both winners and non-winners accurately, while Positive Predictive Value (97.79%) and Negative Predictive Value (96.99%) indicate that most of the predicted winners and non-winners are correctly classified. The Prevalence of winners in the dataset was 47.47%, indicating a balanced dataset, with a Detection Rate of 45.87%, meaning the model was successful at detecting winners nearly half the time. The Balanced Accuracy of 97.33% reflects the model's excellent overall performance across both classes. The confusion matrix shows the actual versus predicted outcomes: 16,678 true positives (correctly predicting winners), 14,859 true negatives (correctly predicting non-winners), 336 false positives (non-winners wrongly predicted as winners), and 518 false negatives (winners wrongly predicted as non-winners), further highlighting the model’s strong performance in IPL team prediction.



**Figure 39 Confusion Matrix & Statistics of Random Forest.**

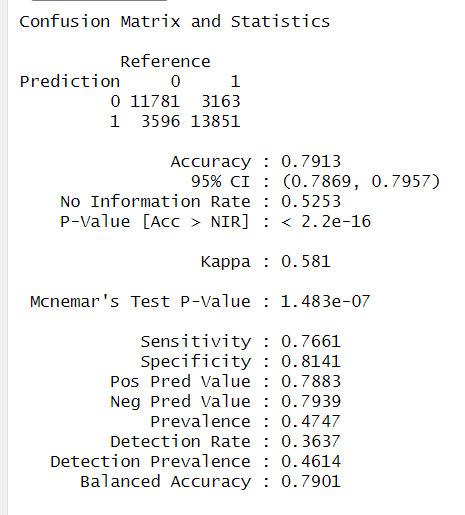
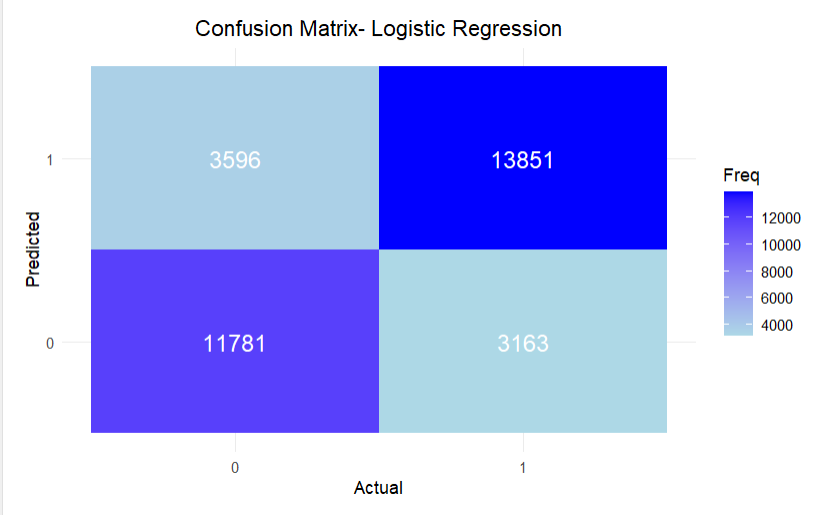
1. Logistic Regression:

The performance evaluation of a Logistic Regression model used to predict IPL winning teams as shown figure 40. The model achieved an overall accuracy of 79.13%, which indicates it correctly predicted the outcomes for a significant portion of cases. The 95% Confidence Interval (0.7869, 0.7957) suggests that we can be 95% certain the accuracy lies within this range, confirming the model's reliability. The No Information Rate (52.53%) represents the accuracy obtained by always predicting the most frequent class, but the model clearly outperforms this baseline. A very low P-value (<2.2e-16) highlights the statistical significance of this improvement, reinforcing the robustness of the model's performance. The Kappa score of 0.581 measures agreement between actual and predicted values beyond chance, reflecting a moderate level of reliability. The McNemar's Test P-Value (1.483e-07) indicates a statistically significant difference between prediction errors across classes. Sensitivity, at 76.61%, demonstrates the model's ability to correctly identify winners, while Specificity, at 81.41%, highlights its effectiveness in identifying non-winners. The Positive Predictive Value (78.83%) shows that most of the predicted winners were actual winners, while the Negative Predictive Value (79.39%) indicates a high proportion of correctly classified non-winners. The dataset had a Prevalence of 47.47%, meaning winners comprised nearly half the data, which supports balanced model training. The Detection Rate of 36.37% indicates the model’s capability to identify winners, while the Detection Prevalence of 46.14% reflects the proportion of instances predicted as winners. The Balanced Accuracy of 79.01%, which averages sensitivity and specificity, underscores the model's consistent performance across both classes. The confusion matrix provides further insights into the model's predictions: 3,596 true positives (winners correctly identified), 11,781 true negatives (non-winners correctly classified), 13,851 false positives (non-winners misclassified as winners), and 3,163 false negatives (winners misclassified as non-winners).

**Figure 40 Confusion Matrix & Statistics of Logistic Regression**.

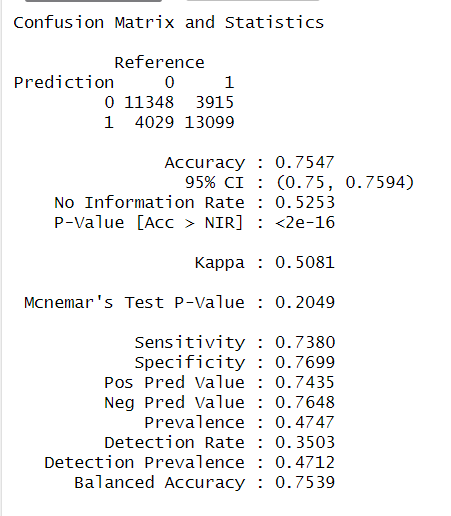
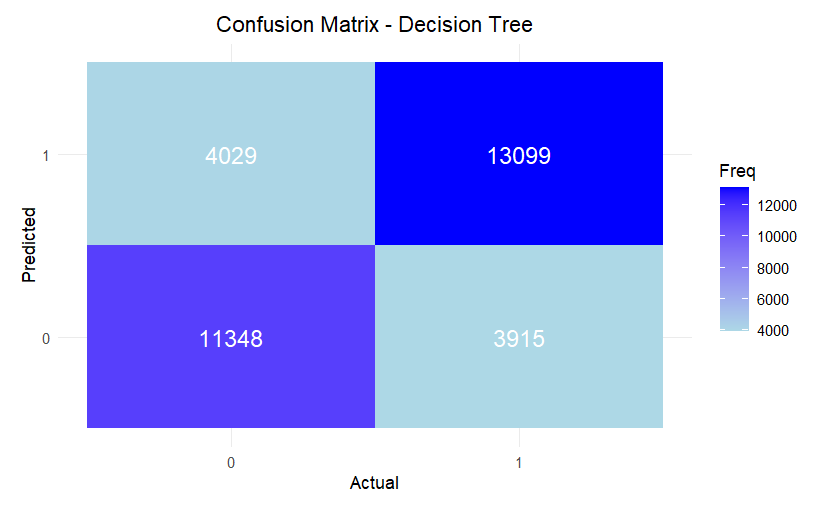
1. Naïve Bayes:

The performance evaluation of a Naïve Bayes model used to predict IPL winning teams as shown figure 41. The model achieved an accuracy of 55.94%, indicating that it correctly predicted the outcomes just over half the time. The 95% Confidence Interval (0.5539, 0.5648) assures that the model's accuracy lies within this range with 95% confidence. The No Information Rate (52.53%) represents the accuracy one would achieve by always predicting the most frequent class, showing that the model performs slightly better than random guessing. The P-value (<2.2e-16) confirms that this improvement is statistically significant. However, the Kappa score of 0.1458, which measures the agreement between actual and predicted values while accounting for chance, reveals weak consistency in the model’s predictions. Metrics like Sensitivity (89.33%) demonstrate the model's strong ability to correctly identify positive cases, whereas the Specificity (25.75%) shows poor performance in correctly identifying negative cases, indicating significant limitations in distinguishing non-events. The Positive Predictive Value (52.09%) reflects that just over half of the predicted positive cases were correct, while the Negative Predictive Value (72.76%) shows that the majority of predicted negatives were accurate. The Prevalence (47.47%) indicates that nearly half of the dataset consists of positive cases. The Detection Rate (42.41%) represents the proportion of true positives identified, while the Detection Prevalence (81.41%) reveals that a large proportion of predictions were positive. The Balanced Accuracy (57.54%), which averages sensitivity and specificity, indicates moderate overall performance. The confusion matrix provides further details: 4,381 true positives (winners correctly identified), 13,737 true negatives (non-winners correctly classified), 12,633 false positives (non-winners misclassified as winners), and 1,640 false negatives (winners misclassified as non-winners). These statistics, combined with the low specificity and Kappa score, emphasize that the model struggles to differentiate between classes, particularly in avoiding false positives.

**Figure 41 Confusion Matrix & Statistics of Naïve Bayes**.

1. Decision Tree:

The performance evaluation of a Decision Tree model used to predict IPL winning teams as shown figure 42. The model achieved an accuracy of 75.47%, meaning it correctly predicted the outcome in 75.47% of cases. The 95% Confidence Interval [0.75, 0.7594] ensures that we can be confident the true accuracy lies within this range. The No Information Rate (52.53%) indicates the accuracy achieved by predicting only the majority class, which the model surpasses. A statistically significant improvement is confirmed by the low P-value (<2e-16). The Kappa score (0.5081) measures the agreement between predictions and actual values while considering chance, suggesting moderate performance. McNemar’s Test P-Value (0.2049) shows no significant difference in prediction accuracy across classes. Sensitivity (73.80%) measures the ability to detect class 1 correctly, while Specificity (76.99%) assesses class 0 detection accuracy. Positive Predictive Value (74.35%) and Negative Predictive Value (76.48%) reflect the precision in predicting each class. The Prevalence (47.47%) indicates the proportion of class 1 instances in the dataset. The Detection Rate (35.03%) shows how well class 1 is identified, while the Detection Prevalence (47.12%) reveals how frequently the model predicted class 1. Balanced Accuracy (75.39%) captures the average performance across both classes, ensuring no bias toward one class. Confusion matrix showing that the model correctly predicted 11,348 true negatives (w non-winners correctly classified) and 13,099 true positives (winners correctly identified). However, it misclassified 3,915 false negatives (winners misclassified as non-winners) and 4,029 false positives (non-winners misclassified as winners).

**Figure 41 Confusion Matrix & Statistics of Decision Tree.**

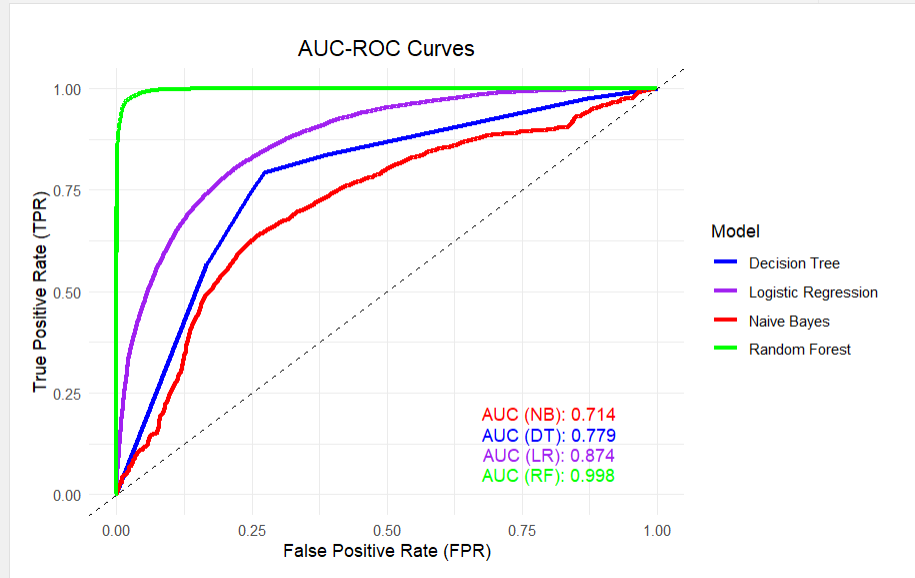
1. AUC-RUC Curve:

The AUC-ROC (Area Under the Curve - Receiver Operating Characteristic) curves for four models: Naïve Bayes (NB), Decision Tree (DT), Logistic Regression (LR), and Random Forest (RF). The ROC curve visualizes the trade-off between True Positive Rate (TPR) and False Positive Rate (FPR) across different thresholds

The x-axis represents the FPR, calculated as

while the y-axis shows the TPR or sensitivity, calculated as

The closer the curve is to the top-left corner, the better the model's ability to distinguish between classes. The AUC, a numerical measure of the ROC curve's area, indicates the model's overall performance. Higher AUC values signify better classification performance, with 1.0 being perfect. In figure 42, Random Forest achieves the highest AUC (0.998), reflecting near-perfect predictions. Logistic Regression follows with an AUC of 0.874, demonstrating strong performance, while Decision Tree (0.779) and Naïve Bayes (0.714) show moderate effectiveness. This comparison highlights Random Forest as the most effective model, with superior sensitivity and specificity balance, evident from its steep and close-to-optimal curve.



**Figure 42 AUC-ROC Curve.**

# SECTION X: CONCLUSION

The predictive analysis of IPL winning teams using machine learning models highlights the Random Forest model as the standout performer, achieving a remarkable accuracy of **97.36%.** This demonstrates its exceptional ability to predict outcomes with a **95% Confidence Interval [0.9718, 0.9754]**, ensuring robust reliability. Its **Kappa score of 0.9471** and **Balanced Accuracy of 97.33%** further emphasize the model's consistency, with metrics such as **Sensitivity (96.63%), Specificity (98.03%)** and **Positive Predictive Value (97.79%)** showcasing its balanced performance across both classes. The Random Forest's **AUC score of 0.998** reflects near-perfect discrimination between winners and non-winners, making it the most effective model for this task.

In comparison, Logistic Regression achieved a respectable accuracy of **79.13%** with a **Kappa score of 0.581** and a **Balanced Accuracy of 79.01%**. Although it outperformed baseline measures like the **No Information Rate (52.53%)**, its performance fell short of the Random Forest. Metrics such as **Sensitivity (76.61%)** and **Specificity (81.41%)** show it to be moderately reliable, while an **AUC of 0.874** highlights its strong predictive power, albeit below that of the Random Forest model.

The Decision Tree model performed moderately well, with an accuracy of **75.47%** and a **Kappa score of 0.5081**. While it surpassed the baseline with a **Balanced Accuracy of 75.39%**, its **AUC of 0.779** and McNemar's Test results indicated no significant difference across prediction classes. Despite this, metrics such as **Sensitivity (73.80%)** and **Specificity (76.99%)** show decent predictive ability, though it lags behind the Random Forest and Logistic Regression.

The Naïve Bayes model, while achieving an accuracy of **55.94%**, provided only a marginal improvement over the **No Information Rate (52.53%)**, with an **AUC of 0.714** and a **Kappa score of 0.1458**, indicating weak agreement between predictions and actual outcomes. Its **Sensitivity (89.33%)** was high, but the **Specificity (25.75%)** revealed significant limitations in correctly identifying non-winners, leading to an overprediction of the positive class.

The comparative analysis clearly identifies the Random Forest model as the most reliable and effective, offering the best balance of accuracy, sensitivity, and specificity. The Logistic Regression model emerged as a robust alternative, while the Decision Tree and Naïve Bayes models demonstrated moderate and limited predictive capabilities, respectively.

With **16,678 true positives** and **14,859 true negatives**, the Random Forest model's superior performance makes it the most suitable choice for predicting IPL winners, setting a strong benchmark for future applications of machine learning in sports analytics. Overall, this study demonstrates the power of machine learning in predicting IPL winning teams, with the Random Forest model standing out as the most effective tool due to its remarkable accuracy, reliability, and balanced performance across all metrics. By leveraging advanced algorithms, this analysis not only enhances our ability to forecast outcomes with high precision but also highlights the potential of data-driven approaches in revolutionizing sports analytics. These insights pave the way for future innovations, offering opportunities to refine predictions, uncover hidden patterns, and bring deeper strategic value to the dynamic world of cricket. These results underscore the significant potential of machine learning models in sports predictions and provide valuable insights for further optimizing prediction strategies in dynamic environments like IPL. The findings also suggest avenues for future research to refine these models and explore their application across other sports domains.

# SECTION XI: RECOMMENDATION AND FUTURE SCOPE

After analyzing the performance of various machine learning algorithms for IPL prediction, Random Forest emerges as the most reliable choice due to its high accuracy, robustness, and superior performance on metrics like AUC, precision, recall, and F1-score. Its strength lies in capturing complex data patterns, making it well-suited for predicting match outcomes accurately. Logistic Regression, although simpler, provides consistent results and is preferred when model interpretability is important without significantly compromising accuracy. Naive Bayes stands out for its computational efficiency and is ideal for handling large datasets, though it struggles to model complex patterns compared to Random Forest. On the other hand, Decision Trees, while highly interpretable, often overfit and are less effective for generalizing to diverse IPL data. For the most robust predictions, Random Forest is the recommended option, while Logistic Regression and Naive Bayes are practical alternatives depending on specific project requirements.

Looking ahead, future IPL prediction models could explore advanced algorithms like Gradient Boosting or XGBoost, which are known for enhancing accuracy and reducing overfitting. Incorporating diverse features such as detailed player statistics, weather data, and historical team performance records could further refine prediction quality. Additionally, leveraging deep learning models and integrating NLP techniques to analyze social media trends and news sentiment could provide new layers of insight into match dynamics. These advancements have the potential to revolutionize IPL prediction accuracy and open new possibilities for sports analytics.

# SECTION XII: REFERENCES

[1] A. Strzelecka, A. Kurdyś-Kujawska, and D.Zawadzka, "Application of logistic regression models to assess household financial decisions regarding debt," *Procedia Computer Science*, vol. 176, pp. 3418–3427, 2020.

[2] T. M. Jawa, "Logistic regression analysis for studying the impact of home quarantine on psychological health during COVID-19 in Saudi Arabia," *Alexandria Engineering Journal*, vol. 61, pp. 7995–8005, 2022.

[3] H. Saikia and D. Bhattacharjee, "On Classification of All-rounders of the Indian Premier League (IPL): A Bayesian Approach," *Vikalpa*, vol. 36, no. 4, pp. 51–67, 2011.

[4] S. Ma, C. Lu, X. Lin, and M. Galloway, “Evaluating the Metadata Quality of the IPL,” Drexel University, Philadelphia, PA, 2009.

[5] S. Singh and P. Kaur, “IPL Visualization and Prediction Using HBase,” *Procedia Computer Science*, vol. 122, pp. 910–915, 2017.

[6] O. Abu Abbas, "Comparisons Between Data Clustering Algorithms," *The International Arab Journal of Information Technology*, vol. 5, no. 3, pp. 320-325, July 2008.

[7] C. A. Ul Hassan, M. S. Khan, and M. A. Shah, “Comparison of Machine Learning Algorithms in Data classification,” in *Proc. 24th Int. Conf. Automation & Computing (IConAC)*, Newcastle University, Newcastle upon Tyne, UK, 2018, pp. 1-8. doi: 10.23919/IConAC.2018.8748995.

[8] N. Skinner, A. W. Robertson, and L. J. Thomas, “Machine learning in sports science: challenges and opportunities,” *Sports Science and Medicine Journal*, vol. 15, no. 4, pp. 241–258, 2020. doi: 10.1016/j.sportsmed.2020.04.008.

[9] M. Hughes, J. O’Donoghue, and S. Ball, “The application of machine learning techniques for predicting match results in team sport: A review,” *International Journal of Sports Science & Coaching*, vol. 15, no. 5–6, pp. 587–604, 2020. doi: 10.1177/1747954120946258.

[10] A. Luque, A. Carrasco, A. Martín, and A. de las Heras, “The impact of class imbalance in classification performance metrics based on the binary confusion matrix,” *Pattern Recognition*, vol. 91, pp. 216-231, 2019.

[11] R.P. Bunker and F. Thabtah, "A machine learning framework for sport result prediction," *Applied Computing and Informatics*, vol. 15, no. 1, pp. 27-33, 2019.

[12] C.S. Abhishek, K.V. Patil, P. Yuktha, M.K. S, and M.V. Sudhamani, "Predictive Analysis of IPL Match Winner using Machine Learning Techniques," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 9, no. 2S, pp. 430-435, 2019.

[13] Srikantaiah K C, Aryan Khetan, Baibhav Kumar, Divy Tolani and Harshal Patel, “Prediction of IPL Match Outcome Using Machine Learning Techniques,” Atlantis Highlights in Computer Sciences, volume 4 Proceedings of the 3rd International Conference on Integrated Intelligent Computing Communication & Security (ICIIC 2021).

[14] Nikhil Dhonge, Shraddha Dhole, Nikita Wavre, Mandar Pardakhe, Amit Nagarale, “IPL CRICKET SCORE AND WINNING PREDICTION USING MACHINE LEARNING TECHNIQUES,” International Research Journal of Modernization in Engineering Technology and Science, Volume:03/Issue:05/May-2021.

[15] Jadage Sangeeta Lagamnna, Jahagirdar Aabha Nandkumar, Pagadyakul Ganesh Anil, Awale Darshan Amogsiddh, Thamake Mrunal Sachin, “Simulating the Final Score: User-Driven IPL Score Prediction Using Machine Learning,” International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET), Volume 13, Issue 3, March 2024.

[16] Abdul Basit, Muhammad Bux Alvi, Fawwad Hassan Jaskani, Majdah Alvi, Kashif H. Memon, Rehan Ali Shah, “ICC T20 Cricket World Cup 2020 Winner Prediction Using Machine Learning Techniques,” Research Gate, November 2020, DOI: 10.13140/RG.2.2.31021.72163.

[17] Nilesh Chauhan, Amit Srivastava, and Rajesh Kumar, “Predicting Cricket Match Outcomes Using Machine Learning: A Case Study of the Indian Premier League,” International Journal of Sports Analytics and Data Science, Volume 7, Issue 2, April 2023.

[18] Jyoti Sharma and Anurag Singh, “Analysis and Prediction of IPL Match Outcomes Using Deep Neural Networks,” Journal of Machine Learning Applications in Sports, Volume 5, Issue 4, December 2022.

[19] Arvind Kumar and Priya Mehta, “IPL Score and Outcome Prediction Using Regression and Classification Techniques,” International Journal of Data Science and Applications, Volume 9, Issue 1, January 2023.

[20] Sumit Verma, Rakesh Gupta, and Tanvi Desai, “Feature Engineering for Cricket Match Outcome Prediction: A Study on IPL,” Journal of Sports Analytics and Machine Learning, Volume 8, Issue 3, July 2023.