ASTON UNIVERSITY

BDM200 - MSc Business Analytics Research Project

"EVALUATING FOOTBALL PLAYER PERFORMANCE: INTEGRATING TEAM STRENGTH AND OPPOSITION QUALITY"

Student Name – Yash Pawar

Student Number – 230272215

Supervisor – Muhidin Mohamed

Date of Submission – 30th September 2024

Degree - MSc Business Analytics

Declaration

I declare that I have personally prepared this report and that it has not in whole or in part been submitted for any other degree or qualification. Nor has it appeared in whole or in part in any textbook, journal or any other document previously published or produced for any purpose. The work described here is my/our own, carried out personally unless otherwise stated. All sources of information, including quotations, are acknowledged by means of reference, both in the final reference section and at the point where they occur in the text.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, ***Dr Muhidin Mohamed****, for his encouragement, invaluable guidance, and support throughout the course of this research project. His insights and constructive feedback have been crucial in shaping the direction of this work.*

I would also like to extend my sincere thanks to our program director Dr Viktor Pekar, all the tutors and staff at Aston University for providing the necessary resources and academic support that contributed to the completion of this project. Their expertise and assistance were invaluable.

Special thanks go to my colleagues and friends for their moral support and constructive discussions. Finally, I am forever grateful to my family for their unwavering support, patience, and encouragement throughout this journey.

Table of Contents

ABSTRACT **7**

CHAPTER 1: INTRODUCTION **8**

1.1 Background 8

1.2 Problem Statement 8

1.3 Aims & Objectives 9

1.4 Research Questions 10

1.5 Structure of the Report 10

CHAPTER 2: LITERATURE REVIEW **11**

2.1 Introduction 11

2.2 Traditional Metrics and their Limitations in football 11

2.3 Advanced Metrics and KPIs in Football 12

2.4 Machine Learning in football performance evaluation 13

2.5 Benchmarking and Evaluation Techniques 13

2.6 Contextual Factors: Team Strength and Opposition Quality 14

2.7 Summary of Literature Review 14

2.8 Gaps in the Literature and Future Directions 17

CHAPTER 3: METHODOLOGY **18**

3.1 Research design & Ethics 18

3.1.1 Workflow 19

PART A. Individual Player Performance Model 20

3.2 Dataset Description 20

3.2.1 Data Analysis 22

3.2.2 Data Pre-processing 23

3.2.3 Data Cleaning 24

3.2.4 Descriptive Stats 26

3.2.5 Visualizations 27

3.2.5.1 Univariate Visualizations 27

3.2.5.2 Bivariate Visualizations 32
3.2.5.3 Multivariate Visualizations 36
3.2.6 Train Test Split 38
3.2.7 One Hot Encoding 38
3.2.8 Feature Scaling 38
3.2.9 Feature Selection 39
3.3 Machine Learning Algorithms 41
3.3.1 Baseline Model (Linear Regression) 42
3.3.2 Random Forest 42
3.3.3 Decision Tree 44
3.3.4 Support Vector Machine 45
3.3.5 K Nearest Neighbors 46
PART B - Team Strength & Opposition Quality 48
3.4 Data for evaluating team strength and opposition quality 48
3.4.1 Team strength 49
3.4.1.1 Home Teams Actual & Predicted strength **(Based on all players)** 49
3.4.1.2 Home Teams Actual & Predicted strength **(Based on key players)** 50
3.4.2 Opposition Quality 51
3.4.2.1 Opposition Teams Actual & Predicted strength **(Based on all players)** 51
3.4.2.2 Opposition Teams Actual & Predicted strength **(Based on key players)** 53
3.4.3 Players Actual vs Predicted Overall Strength 55
CHAPTER 4 – RESULTS & COMPARISON 56
4.1 PART A – Player Performance Evaluation 56
4.1.1 Evaluation Metrics Comparison 56
4.1.2 Actual & Predicted Values comparison 56
4.1.3 Feature Importance 58
4.2 PART B – Team Strength & Opposition Quality 59
4.2.1 Comparison of Home Team vs Opposition Strength **(based on all players)** 59

4.2.2 Comparison of Home Team vs Opposition Strength **(based on key players)**..... 61

4.2.3 Player level Comparison 63

CHAPTER 5 – CONCLUSION & RECOMMNEDATIONS 64

5.1 Conclusion..... 64

5.2 Limitations 64

5.3 Recommendations..... 65

REFERENCES 66

APPENDIX 71

List of Figures:

Figure 1. History of Football 8
Figure 2. Variables with missing values showing outliers 25
Figure 3. Count of players by club (Top 10) 27
Figure 4. Count of players by nationality (Top 10) 28
Figure 5. Pie chart of Preferred Foot 28
Figure 6. Height distribution of players 29
Figure 7. Weight distribution of players 29
Figure 8. Age distribution of players 29
Figure 9. Top 10 tall players 30
Figure 10. Top 10 fastest players 30
Figure 11. Top 10 heaviest players 31
Figure 12. Graph of Overall vs preferred foot 32
Figure 13. Graph of Overall vs position 33
Figure 14. Graph of Age vs Potential 33
Figure 15. Graph of Weight vs Sprint speed 33
Figure 16. Graph of Strength vs Shot Power 34
Figure 17. Graph of Average Overall Rating by skill Moves 34
Figure 18. Graph of Overall vs Player count 34
Figure 19. Graph of Overall vs Age 35
Figure 20. Graph of Preferred Foot vs Position 35
Figure 21. Distribution of skill related variables 36
Figure 22. Distribution of skill related variables 36
Figure 23. Correlation Matrix 37
Figure 24. Highly Correlated variables 39
Figure 25. Real Madrid CF Actual Overall strength vs Predicted Overall Strength (Based on all players) 49
Figure 26. Actual Overall strength of Oppositions (Based on All players) 51
Figure 27. Predicted Overall strength of Oppositions (Based on All players) 52
Figure 28. Players Actual and Predicted Overall Comparison 55
Figure 29. Actual vs Predicted Overall Graph of SVM Model 57
Figure 30. Feature Importance graph for SVR Model 58
Figure 31. Strength Difference of Opposition with Real Madrid CF (Based on all players) 60
Figure 32. Strength Difference of Opposition with Real Madrid CF (Based on key players) 62
Figure 33. K. Benzema's Strength difference vs Opposition players 63

List of Tables:

Table 1. List of Reviewed Literatures 15
Table 2. List of Variables in the Dataset 20
Table 3. List of Variables removed before data cleaning 23
Table 4. List of variables with null values 24
Table 5. Descriptive stats of player attributes 26
Table 6. List of Top 10 tall players 30
Table 7. List of top 10 fastest players 30
Table 8. List of top 10 heaviest players 31
Table 9. List of top 10 players with highest overall 31
Table 10. List of top 10 players with highest potential 32
Table 11. Final list of removed variables for machine learning models 40
Table 12. Model Performance of Linear Regression (Baseline Model) 42
Table 13. Random Forest RMSE based on hyperparameter combinations 43
Table 14. Model Performance of Random Forest Model 43
Table 15. Decision tree RMSE based on hyperparameter combinations 44
Table 16. Model performance of Decision Tree Model 45
Table 17. SVM Models RMSE based on Hyperparameter combinations 45
Table 18. Model Performance of SVM Model 46
Table 19. KNN Models RMSE based on Hyperparameter combinations 47
Table 20. Model Performance of KNN Model 47
Table 21. Variables used for Analysis in Part B 48
Table 22. Real Madrid CF Actual Overall strength vs Predicted Overall Strength (Based on key players) 50
Table 23. Oppositions Actual and Predicted Overall Strength (based on key players) 53
Table 24. Machine Learning Models Results 56
Table 25. Actual vs Predicted Overall Graph of Other Models 57
Table 26. Actual vs Predicted Overall Strength of Teams (Based on All players) 59
Table 27. Calculated Strength Difference (Based on key players) 59
Table 28. Actual vs Predicted Overall Strength of Teams (Based on Key players) 61
Table 29. Strength Difference (Based on Key players) 61

ABSTRACT

This research develops a machine learning-based framework for evaluating football players overall performance based on comprehensive individual attributes. Traditional metrics like goals and assists often overlook important contextual factors, limiting their effectiveness. The research addresses this gap by using key performance indicators (KPIs), including physical attributes, skills, and performance data, to predict player performance. The research applies 5 supervised machine learning models. The Support Vector Machine (SVM), model demonstrated the highest accuracy, with lowest RMSE and highest R-square. Using these predictions, team strength was assessed by analysing Real Madrid CF's performance against teams from top 5 European leagues. Results show a strong correlation between predicted and actual team strengths, validating the models' effectiveness. The findings offer actionable insights for improving player scouting, match strategies, and investment decisions in football. In conclusion, integrating individual and contextual factors provides a more comprehensive evaluation of player performance. Future research could expand this approach by incorporating more dynamic in-game data and additional contextual factors to further refine the predictive models.

Key words: Football, player performance, Machine learning, Team Strength, Opposition Quality.

CHAPTER 1: INTRODUCTION

1.1 Background

Figure 1. History of Football

Football, or association football, a sport in which two teams of eleven players compete to shoot the ball into the goal of the other team without using their hands or arms. The only player allowed to use their hands and arms is the Goalkeeper when they are in the penalty area in front of the goal. The Football Association of England was founded in 1863, that's when the game's first set of official rules was established. In the late 1880s, professional leagues started to emerge, initially in England and later in other nations. Since 1930, the World Cup has been held in France, which was founded in 1904 and has been hosted by FIFA every four years. Football was also introduced in the Olympic games since 1908 (Joy and Weil, 2019).

1.2 Problem Statement

Traditional metrics like goals, assists, etc are frequently used in the evaluation of football players performances, but these methods fall short of fully capturing the context in which a player operates. These metrics do not consider a player's abilities, skills, physical attributes etc. As a result, Coaches, Analysts and football teams may lack effective decision making when it comes to financial investments, player scouting or strategy formulation. In order to close this gap and provide a more precise and comprehensive evaluation framework for football analytics, this research aims to introduce machine learning models to predict players

overall performance based on comprehensive individual attributes of players and followed by evaluating team strength and opposition quality based on the predictions.

1.3 Aims & Objectives

The Main aim of this research is to develop a machine learning-based approach for evaluating football players overall performance using key performance indicators, while also assessing Real Madrid's team strength and oppositions quality from top 5 European leagues based on **all/key** players overall predictions. This research seeks to provide an accurate and holistic understanding of key players, teams & oppositions overall performance to utilize it in football analytics.

The specific objectives are:

Define Key Performance Indicators (KPIs): Identify and utilize football players performance metrics, such as their physical attributes, skills, abilities, and other performance indicators, to predict their overall performance using machine learning models.

Build Predictive Models: Develop and implement machine learning models to predict players overall performance based on the KPIs, providing a data-driven evaluation of individual player contributions in team strength and opposition quality.

Evaluate Model Effectiveness: Measure the effectiveness of the models using evaluation metrics like RMSE and R2 scores to ensure the predictive accuracy of players performance.

Evaluating Team & opposition Strength: Evaluate and compare team & oppositions overall strengths based on the predictions of players overall performances, providing insights into the relative strength of teams in comparison to their opposition. We determine how team & opposition strengths can be assessed based on **all players** and **key players** in the team.

Benchmark against Actual KPIs: Compare the predictions of the best machine learning model to actual KPIs (such as actual overall performance) to assess the added value of the predictive model.

Provide Contextual Insights: Analyse the predicted team strengths relative to opposition teams and discuss how these insights can inform strategic decisions for Coaches, Analysts and football teams.

1.4 Research Questions

- How accurate can machine learning models be to predict football players overall performance based on their performance metrics?
- How can Team & Opponent strengths be determined on the basis of **all players** and **key players** in the team?

1.5 Structure of the Report

The project is divided into 5 chapters in order to fulfil the objectives of the research:

- The topic is introduced in Chapter 1, which also gives a summary of the background, research problem and questions as well as aims and objectives of the study.
- A critical analysis is provided in Chapter 2, literature review - based on past studies related to traditional metrics, advanced metrics, machine learning and contextual factors in football.
- The research methodology is described in Chapter 3, along with the methods for data gathering, data sources, data pre-processing, and evaluating the machine learning models for Part A, while Part B assesses teams & oppositions strengths based on all/key players overall performance predictions.
- In Chapter 4 Part A, the results of machine learning models are analysed and compared while highlighting the importance of features that contributed the most in players predicted overall performance. Part B compares Real Madrid CF's team strength with opposition's strength based on **all players** and **key players** in the team.
- And finally, Chapter 5 concludes the project by addressing its limitations and offering recommendations.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This research aims to develop a machine learning-based approach to evaluate football player performance using key performance indicators like skill sets, physical attributes, and performance data. It also uses player performance predictions to compare a team's strength with its opposition, providing a more precise and comprehensive understanding of football performance prediction.

The literature review emphasised the increasing application of machine learning methods in sports analytics, showcasing the effectiveness of models like Random Forest and Support Vector Machines in player performance prediction. The importance of contextual factors such as team strength and opposition quality based on key players has becoming more widely acknowledged, which emphasises the need for more thorough evaluation techniques in football analytics. The traditional and advanced approaches relevant to the integration of multiple variables in performance evaluation have also been covered in this review. We hoped to clarify the useful applications and results of these integrated models by reviewing relevant case studies, research papers and articles from scholarly sources. Lastly, the literature review has pointed out any gaps in the knowledge base and offer possible lines of inquiry for further research. By doing this, we hope to aid in the creation of frameworks for evaluating football player performance that are more resilient and contextually significant.

2.2 Traditional Metrics and their Limitations in football

According to the study by (Qader et al., 2017) a multi-criteria decision-making approach to evaluate football players' performance using a mix of anthropometric, fitness, and skill-based tests. By ranking players according to these standards using the TOPSIS algorithm, the methodology provided an unbiased model for player selection. The system was found to be useful in balancing various performance factors by the study; however, it was limited by the difficulty of precisely measuring subjective skills like creativity and teamwork. Understanding the drawbacks of evaluating players exclusively using conventional metrics was made easier by this study.

(Cintia et al., 2015) in their research introduced a network-based model for evaluating of football teams' performances by analysing passing sequences and interactions on the pitch. The authors were able to quantify game diversity and volume by considering players and

zones as network nodes, and they were able to demonstrate a relationship between these metrics and team performance. Even though this method goes beyond conventional measurements, it shows how basic metrics like passes can overlook important tactical information, indicating the need for a more thorough investigation that takes other players attributes etc into account.

In order to evaluate football players technical and physical abilities, (Rosch et al., 2000) in their study, created the extensive F-MARC test battery which placed a strong emphasis on endurance, speed, and flexibility. Their methodology centred on the evaluation of players via organised tests tailored to the game of football, thereby yielding objective data for the analysis of individual player performance. The study demonstrated usefulness of standardised testing in establishing age and skill-level benchmarks. Although this study emphasised the value of physical profiling, it is constrained by its emphasis on physical metrics, which ignored the significance of tactical choices made during gameplay.

2.3 Advanced Metrics and KPIs in Football

(Herold et al., 2021) conducted a study to investigate the use of attacking key performance indicators (KPIs) across different levels of football, from elite to youth academy levels. Their results implied that practitioners use advanced positional data, like Expected Possession Value (EPV) and Space Control, less frequently and mostly rely on basic metrics like shots and crosses. Many coaches prefer outcome-based metrics, highlighting the disconnect between advanced KPI methodologies and their practical application (Herold et al., 2021).

By placing individual contributions in the perspective of the larger competitive environment, the incorporation of cutting-edge statistical techniques into sports analytics frameworks has been successful in delivering deeper insights into player performance (Baumer et al., 2023). These approaches are a major improvement over conventional measurements, allowing for a more thorough and precise evaluation of football players, teams & opposition strengths.

According to (Mountifield, 2023) the application of big data and sophisticated statistical techniques had allowed sports analytics to progress beyond alchemy and become a crucial component in assessing sports performance. The advanced algorithms and predictive models that examined enormous datasets and offered useful insights into player performance and team dynamics are clear examples of this growth. This study proved relevant in order to consider big data to evaluate player performance.

2.4 Machine Learning in football performance evaluation

Predicting the performance of football players has been significantly impacted by recent developments in machine learning. Based on biometric parameters like VO2 and acceleration, (Morciano et al., 2024) in their study, used supervised machine learning models, such as Random Forest and a novel whale-optimization algorithm, to predict players' performances. Their study influenced our research focused on individual players overall performance prediction using KPIs by providing a dependable method of predicting above-average player performances across various roles.

(Rajeswari et al., 2024) investigated predictive modelling in sports, highlighting the significance of past data and fitness records in predictive models and proving the usefulness of algorithms like Naïve Bayes and linear regression for performance forecasting in multiple sports.

Moreover, (Merzah et al., 2024) used a variety of machine learning models, including Random Forest, SVM, and Decision Trees, to categorise football players' performances during practice and games. This verified the model's accuracy in evaluating player activity based on metrics like speed and activity counts. These studies validate the use of such models in our research by offering crucial insights into the efficacy of ML in players performance evaluation.

In order to predict football player performance based on conventional metrics like goals scored, minutes played, and positional data, (Manish et al., 2021) compared machine learning and deep learning models. Their results highlighted the drawbacks of depending only on conventional performance metrics, emphasising how they ignore more complex elements of gameplay like player interactions and tactical decisions. The study backs up the idea that in order to increase prediction accuracy and player evaluation, more sophisticated techniques like machine learning algorithms should be added to traditional metrics.

2.5 Benchmarking and Evaluation Techniques

In their study, (Nikolaos Giannakoulas et al., 2023) compared different machine learning techniques and assessed predictive models used to predict football players' goal-scoring performance. R-squared and RMSE were the two main metrics used to evaluate the accuracy of the models. These measures offered insights into the trade-offs between guaranteeing overall accuracy and minimising large errors, as well as assisting in determining the predictive power of the models in their research.

(Tatachar, A.V.,2021) employed evaluation metrics like Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared to present a comparative analysis of different regression models, such as Linear Regression, Ridge Regression, Lasso Regression, and Support Vector Regression. Based on these metrics, the study found that Polynomial Regression performed the best; in particular, its high R-squared value, which indicated the model's superior accuracy, supported this conclusion. This study emphasised how crucial it is to assess machine learning models with all-inclusive metrics in order to guarantee precise performance forecasts.

2.6 Contextual Factors: Team Strength and Opposition Quality

Football players decisions are impacted by contextual factors that are both static and dynamic (Levi and Jackson, 2018). Personal performance, score status, momentum, and external instructions from coaches are included in dynamic factors. Static factors included preparation levels, personal pressures, and the importance of the match.

(Goncalves et al., 2019) in their study, focused on how opposition quality impacts game dynamics as they investigated the spatiotemporal aspects of football team behaviour. They discovered that teams that faced more formidable opponents adjusted by shortening the game and widening it, while decreasing possession time. This study demonstrated how the strength of the opposition greatly affects football spatial patterns.

The differences in technical performance between strong and weak teams, in comparison to their weaker counterparts, strong teams showed more consistency in important technical metrics like passes and shots on target were examined according to the study by (Liu et al., 2016). In their comparative study of successful and unsuccessful La Liga teams, (Brito de Souza et al., 2019) found that shooting accuracy was the primary factor separating the top teams from the bottom. Their research reinforced how crucial opposition quality is to a team's success, especially when it comes to offensive performance. Comprehending these contextual factors was essential for a comparison of Team strength and opposition quality based on player performance.

2.7 Summary of Literature Review

The table below provides a thorough summary of all the reviewed literatures that has been painstakingly analysed and reviewed throughout the research. From a variety of academic

sources, it provides a concise but insightful summary of the papers reviewed for literature that has shaped the foundations of this research by summarising the main purpose, methodologies, findings and limitations and how it was relevant to our research.

Table 1. List of Reviewed Literatures

Authors	Purpose	Methodology	Findings	Limitations	Relevance
Hannah R Levi, Robin C Jackson (2018)	Examine how decision-making is influenced by contextual factors	Case studies and scenario analysis	Players' decisions affected by opposition quality	Focused on elite athletes	Supports use of opposition quality in performance evaluation
Benjamin S. Baumer, Gregory J. Matthews, Quang, Nguyen (2023)	Discuss fundamentals of sports analytics and statistical tools	Review of existing literature	Emphasizes need for sophisticated statistical tools	Limited direct application to football	Highlights use of statistical tools in performance analysis
Charles Mountifield, (2023)	Evolution of sports analytics to a scientific approach	Literature review and historical analysis	Shift towards data-driven decision-making	Lack of focus on practical applications	Relevant to data-driven performance analysis
Rosch, D., Hodgson, R., Peterson, L., Graf-Baumann, T., Junge, A., Chomiak, J., & Dvorak, J. (2000)	Evaluate footballers' physical/technical performance	F-MARC test battery	Established baseline for physical/technical attributes	Focused only on physical metrics	Useful for evaluating players' physical attributes
Qader, M.A., Zaidan, B.B., Ali, S.K., Kamaluddin, M.A., & Radzi, W.B. (2017)	Propose a decision-making model for player selection	TOPSIS algorithm	Effective multi-criteria selection	Complex to measure subjective skills	Highlights traditional metric limitations
Cintia, P., Rinzivillo, S., & Pappalardo, L. (2015)	Evaluate team performance using network analysis	Network analysis of passing sequences	Network metrics correlated with success	Ignores defensive actions	Introduces new methods beyond traditional metrics
Gianluca Morciano, Andrea Zingoni, Giuseppe Calabrò (2024)	Predict player performance using machine learning	Random Forest, Neural Networks, ADA-boost	Achieved 90% accuracy for above-average players	Limited to biometric data and not match dynamics	Demonstrates machine learning's value for performance prediction
Rajeswari M, Renukadevi B, Lokesh R, Sabarish K (2024)	Develop predictive models for player performance	Linear Regression, Naïve Bayes	Demonstrated effectiveness of basic ML models	Focused on multiple sports, not tailored to football	Highlights ML models' applicability to sports performance
Baydaa M. Merzah, Muayad S. Croock, Ahmed N. Rashid (2024)	Classify football player performance	Random Forest, SVM, Decision Tree	Decision Tree performed best for match sessions	Limited to physical performance and activity features	Relevant for using Random Forest for football player performance
Herold, M., Kempe, M., Bauer, P., Meyer, T. (2021)	Investigate the use and perceptions of attacking Key Performance Indicators (KPIs)	Surveyed 145 football practitioners from 42 countries	Practitioners prefer simple metrics like shots	Low use of advanced metrics due to lack of education	Highlights the need for better KPI usage in football

Manish S., Vandana Bhagat, Pramila RM (2021)	Compare ML and deep learning models for performance prediction	Regression models	Multiple Regression performed best	Limited focus on specific metrics and positional data	Demonstrates effectiveness of ML models in performance prediction
Tatachar, A.V (2021)	Compare regression models for accuracy	Evaluated Linear, Ridge, Lasso, Support Vector	Polynomial Regression showed best performance	Models tested on non-sports data	Highlights importance of benchmarking models with RMSE and R-squared
Nikolaos Giannakoulas, George Papageorgiou & Christos Tjortjis (2023)	Parallelize outlier detection models	Ensemble models and score combination strategies	Ensemble models improve accuracy	Focused only on outlier detection	Relevant for reducing errors in football analytics models
Goncalves, B., Marcelino, R., Santos, S., Lago-Penas, C., & Sampaio, J. (2019)	Analyse how opposition quality influences team behaviour	Spatiotemporal analysis	Stronger opponents shorten game length and increase width	Focused only on spatial features	Useful for understanding how opposition quality impacts team strength
Liu, H., Gomez, M.-A., Goncalves, B., & Sampaio, J. (2016)	Examine performance variation between strong and weak teams	Analysed 380 matches using technical indicators	Strong teams were more consistent in passes and shots	Focused on technical aspects, excluded psychological factors	Highlights importance of team strength in performance consistency
Brito de Souza, D., Lopez-Del Campo, R., Blanco-Pita, H., Resta, R., & Del Coso, J. (2019)	Compare successful and unsuccessful teams in La Liga	Video analysis of top 3 and bottom 3 teams over 8 seasons	Shooting accuracy and shots conceded were key factors	Limited to match statistics, no tactical or psychological analysis	Reinforces opposition quality's role in team success

Despite the progress made, the review pointed out that there are still gaps in current research. It suggested that future studies should focus on creating more holistic approaches to better evaluate football player performance.

2.8 Gaps in the Literature and Future Directions

Although traditional metrics were frequently used to evaluate player performance, the literature review showed that these metrics fall short of fully capturing the impact of a player. These metrics provided less information about physical characteristics, individual skills, team strength, and opposition quality for coaches, analysts, and football teams. As a result, they are insufficient for player scouting, strategy development, and financial decision-making. This research gap highlighted the need for more sophisticated methods that consider a larger variety of player data and contextual factors to calculate the players overall performance.

Hence, this research aims to close these gaps by using machine learning models to evaluate player performance based on extensive Key Performance Indicators (KPIs), such as physical characteristics, skills, and abilities. This method, in contrast to earlier research, incorporated several aspects of individual performance to offer a more comprehensive and data-driven analysis. By providing a predictive framework that can enhance decision-making processes in player acquisition, financial investments, and match strategy, the research advances football analytics.

CHAPTER 3: METHODOLOGY

3.1 Research design & Ethics

This research used secondary data and a quantitative approach to evaluate football player performance. The methodology was divided into two parts:

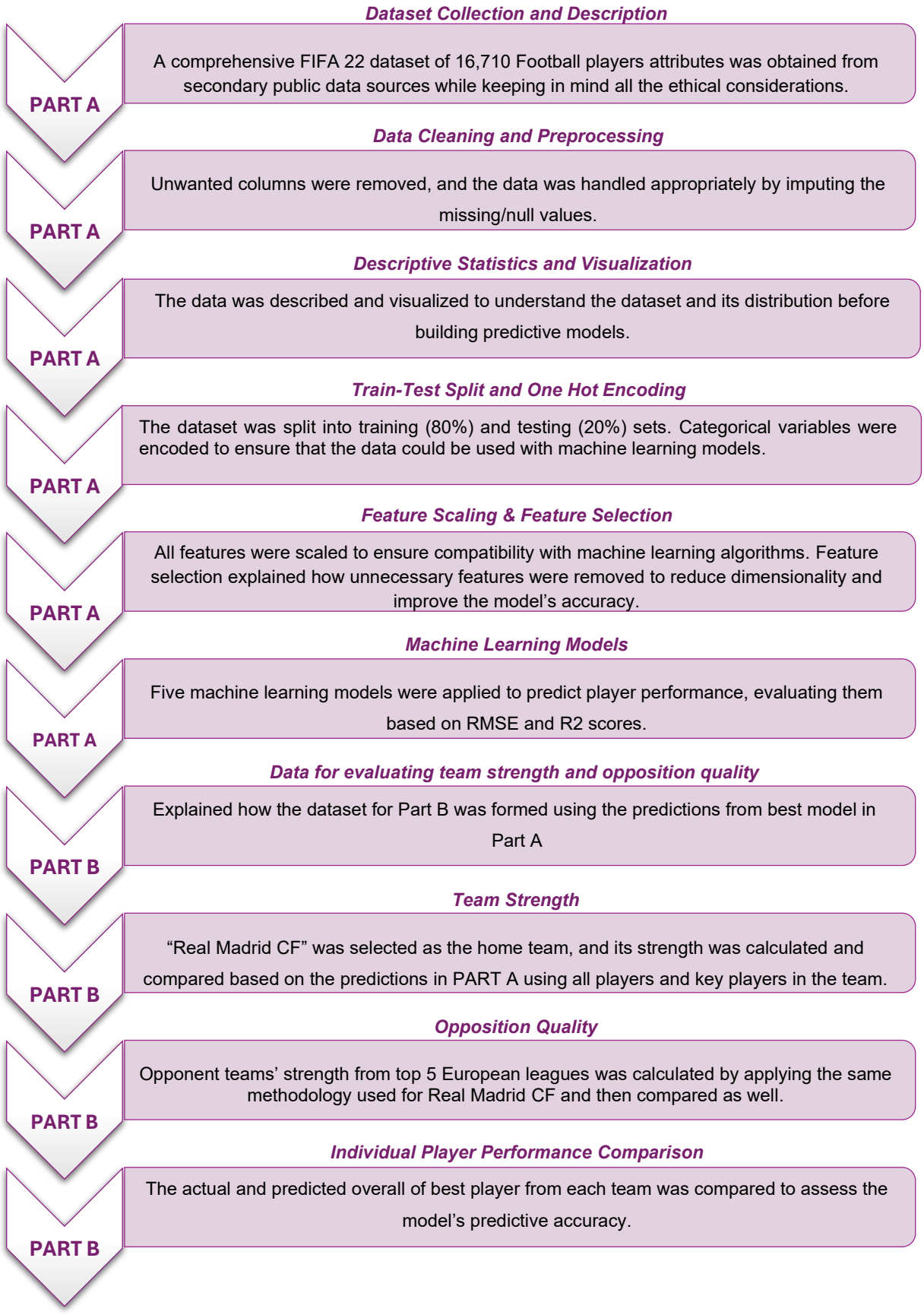
PART A - INDIVIDUAL PLAYER PERFOMANCE MODEL (Using machine learning models to predict Individual player performance)

PART B - TEAM STRENGTH & OPPOSITION QUALITY (Comparing team strength of Real Madrid CF with opposition teams from top 5 European leagues based on overall performance predictions of **all/key** players in the team)

The data has been gathered from reputable public data sources such as Kaggle and Sofifa.com. The analysis for data manipulation, modelling, visualization and comparative analysis has been done using **Python** (Jupyter Notebook).

The research complies with ethical guidelines as it uses publicly available secondary data instead of collecting primary data. There is no need for participant consent as the data sources are reliable and widely available, meaning no sensitive or personal information was used. In order to ensure that the research can be replicated and complies with ethical standards in academic research, the study maintains transparency and integrity by thoroughly documenting the data sources and methodologies.

3.1.1 Workflow



PART A. Individual Player Performance Model

3.2 Dataset Description

The FIFA 22 dataset has been taken from https://www.kaggle.com/datasets/bryanb/fifa-player-stats-database as it includes extensive data on football players, which is necessary for predicting player performance. The dataset contained data of 16,710 football players, which contained players key performance indicators (KPIs) such as players abilities, physical attributes, and overall performance. The dataset had 16711 rows and 65 columns.

About the Dataset: *The dataset was initially collected from sofifa.com using a web crawler that gathered player data for the 2021-22 season. Initially, the author compiled this information to recreate player skills in video games and later adapted it for scouting purposes (BryanB, 2022). The dataset includes players KPIs rated on a scale from 0 to 99, and these statistics have proven to be as reliable, if not better, than other football data sources. Moreover, they have been effectively used to predict football match outcomes and predict player performance.*

Table 2. List of Variables in the Dataset

VARIABLE	DEFINITION
ID	Unique identifier assigned to each player.
Name	Player's name.
Age	Player's age.
Photo	Link to the player's photo.
Nationality	Country the player represents.
Flag	Flag representing the player's nationality.
Overall	An Overall performance rating of player based on his attributes
Potential	A rough estimate of how much a player can improve.
Club	The team to which the player belongs.
Club Logo	Logo of the club the player is associated with.
Value	(€) Estimated market value of the player.
Wage	(€) salary paid by the club to the player.
Special	Total number of special traits or unique abilities possessed by the player.
Preferred Foot	Indicates if the player prefers to use their right or left foot.
International Reputation	International reputation is rated from 1 to 5, affecting the player's global recognition.

Weak Foot	Rating [1-5] indicating how well the player uses their weaker foot.
Skill Moves	Rating [1-5] that evaluates the player's technical skills and tricks.
Work Rate	Describes the player's work rate both in attacking and defensive situations.
Body Type	Describes the player's physique or body shape.
Real Face	Indicates whether the player's in-game face is an authentic recreation.
Position	The primary position the player plays on the field.
Jersey Number	The number worn by the player on their jersey.
Joined	The date when the player joined their current club.
Loaned From	The name of the club the player is loaned from (if applicable).
Contract Valid Until	The year until which the player's contract with the club is valid.
Height	The player's height in centimetres (cm).
Weight	The player's weight in kilograms (kg).
Crossing	Accuracy of the player's crosses.
Finishing	The player's ability to score a goal when finishing an attacking move.
HeadingAccuracy	Accuracy of the player's headers, especially during set pieces.
ShortPassing	Accuracy of the player's short-distance passes.
Volleys	The player's ability to perform volleys.
Dribbling	The player's ability to control the ball while moving.
Curve	The player's ability to curve the ball during shots or passes.
FKAccuracy	The accuracy of the player's free kicks.
LongPassing	Accuracy of long-distance passes made by the player.
BallControl	The player's skill in controlling the ball while in possession.
Acceleration	The player's ability to increase speed quickly.
SprintSpeed	The player's top speed when sprinting.
Agility	The player's ability to change direction quickly and smoothly.
Reactions	The player's speed in reacting to situations on the field.
Balance	The player's overall balance and ability to stay upright under pressure.
ShotPower	The power of the player's shots.
Jumping	The player's ability to leap off the ground.
Stamina	The player's endurance and ability to sustain energy throughout the match.

Strength	The player's physical power and ability to hold off opponents.
LongShots	The player's ability to take accurate long-range shots.
Aggression	The player's determination and intensity when competing for the ball.
Interceptions	The player's ability to anticipate and intercept passes from opponents.
Positioning	The player's awareness and movement to find effective spaces on the field.
Vision	The player's ability to make accurate passes by predicting movement and opportunity.
Penalties	The player's ability to take penalty kicks accurately.
Composure	The player's calmness and control under pressure.
Marking	The player's ability to track and mark opponents defensively.
StandingTackle	The player's proficiency in performing standing tackles.
SlidingTackle	The player's proficiency in executing sliding tackles.
GKDiving	The goalkeeper's ability to dive and save shots aimed at the goal.
GKHandling	The goalkeeper's skill in catching or deflecting the ball.
GKKicking	The goalkeeper's ability to accurately kick the ball.
GKPositioning	The goalkeeper's awareness and positioning when defending the goal.
GKReflexes	The goalkeeper's ability to react quickly to shots.
Best Position	The player's best or most effective playing position on the field.
Best Overall Rating	The player's best overall rating based on their optimal position.
Release Clause	The value that must be paid to release the player from their contract.
DefensiveAwareness	The player's awareness and intelligence in defensive situations.

3.2.1 Data Analysis

Players "**Overall"** Rating (0-99) has been considered as a measure of their **overall performance**. This rating was derived from various physical attributes, skills, and tactical abilities. While primarily a numerical score, it effectively represents a player's comprehensive contribution and serves as a reliable target variable for their performance on the field. 5 machine learning models were used to predict the performance of football players. **Root Mean Square Error (RMSE)** and **R-Squared score** were used as the evaluation metrics to quantify model performance and determine which predictive model performed the best.

3.2.2 Data Pre-processing

Any machine learning pipeline must include data pre-processing since it has a direct bearing on the efficacy and accuracy of the model. The procedure usually consists of multiple steps, including handling missing/null values, data cleaning, data transformation, normalising or scaling the data to make sure it is compatible with the machine learning algorithms. Feature selection is another common pre-processing step that lowers dimensionality and gets rid of extraneous or unnecessary variables to enhance model performance and training effectiveness. The performance and accuracy of machine learning models can be greatly improved by making sure the data is appropriately pre-processed, which will make it simpler for the models to find patterns and relationships in the data (Simplilearn, 2023).

We dropped the following variables from the dataset as they were not contributing to the objectives of this research.

Table 3. List of Variables removed before data cleaning

VARIABLES	DEFINITION
ID	Unique identifier assigned to each player.
Photo	Link to the player's photo.
Flag	Flag representing the player's nationality.
Club Logo	Logo of the club the player is associated with.
Value	(€) Estimated market value of the player.
Wage	(€) salary paid by the club to the player.
Special	Total number of special traits or unique abilities possessed by the player.
Work Rate	Describes the player's work rate both in attacking and defensive situations.
Body Type	Describes the player's physique or body shape.
Real Face	Indicates whether the player’s in-game face is an authentic recreation.
Jersey Number	The number worn by the player on their jersey.
Joined	The date when the player joined their current club.
Loaned From	The name of the club the player is loaned from (if applicable).
Contract Valid Until	The year until which the player's contract with the club is valid.
Marking	The player's ability to track and mark opponents defensively.

GKDiving	The goalkeeper's ability to dive and save shots aimed at the goal.
GKHandling	The goalkeeper's skill in catching or deflecting the ball.
GKKicking	The goalkeeper's ability to accurately kick the ball.
GKPositioning	The goalkeeper's awareness and positioning when defending the goal.
GKReflexes	The goalkeeper's ability to react quickly to shots.
Release Clause	The value that must be paid to release the player from their contract.

3.2.3 Data Cleaning

Handling Null/Missing Values

In football analytics, handling null or missing values in player performance data is essential to guaranteeing the accuracy of predictive models. Errors during data collection or incomplete match/player data are two common causes of missing values. There are various methods for filling in these gaps, including imputation techniques that use the mean, median, or most frequent values. Pandas regard the indicators None and NaN for missing or null values as essentially interchangeable (GeeksforGeeks, 2019). We have used imputation techniques to ensures that data gaps do not skew the analysis. The following variables had null values:

Table 4. List of variables with null values

Variables	Null Values	Variable Type
Club	264	Categorical
Position	26	Categorical
Volleys	37	Continuous
Curve	37	Continuous
Agility	37	Continuous
Balance	37	Continuous
Jumping	37	Continuous
Interceptions	37	Continuous
Positioning	8	Continuous
Vision	37	Continuous
Composure	251	Continuous
SlidingTackle	37	Continuous
DefensiveAwareness	892	Continuous

Categorical Variables: For categorical variables (Club & Position), missing values were filled using the **mode** (the most common values). This method ensured that the most frequent values were used to fill in missing values, maintaining the balance and distribution of categorical features. This imputation strategy ensured data completeness and integrity, preventing the loss of potentially useful information that might have occurred if rows or columns were discarded due to missing values.

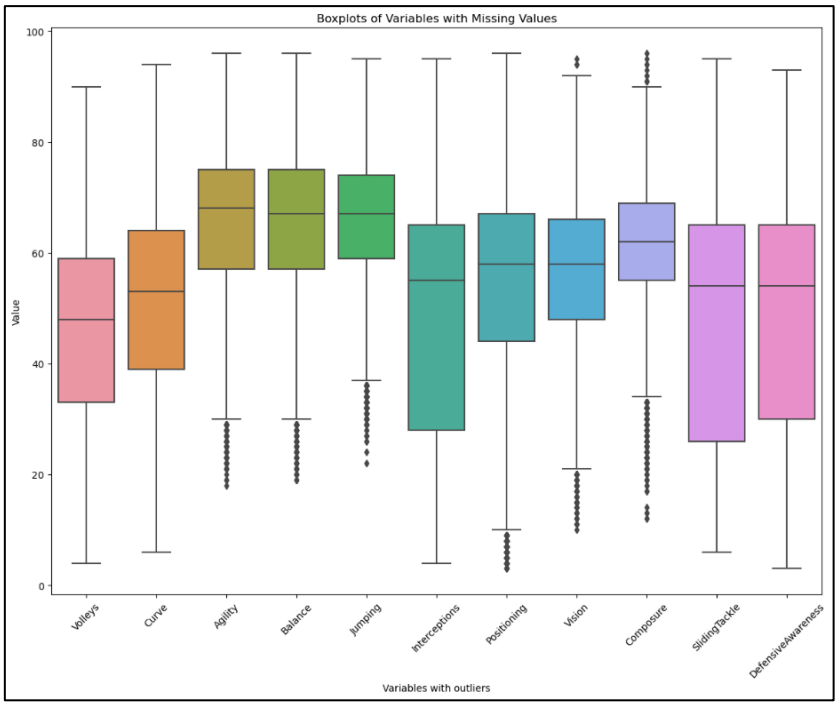


Figure 2. Variables with missing values showing outliers

Continuous Variables: Based on the graph above, Missing values in continuous variables were handled through **mean** and **median** imputation. **Mean** imputation was the default choice for filling in missing values for variables with **no outliers**. For variables **containing outliers**, missing values were filled using the **median.** This approach helped preserve the overall distribution of the continuous variables while maintaining the quality of the dataset for subsequent analysis. 0 Null/Missing values were left after handling them appropriately.

Cleaning the Position, Height & Weight columns

The embedded HTML codes present in the **'Position'** column were eliminated during the data cleaning process to retain only the relevant positional information for each player. To maintain consistency throughout the dataset, substitute ('SUB') and reserve ('RES') players were given their corresponding values from 'Best Position' column. In an effort to further streamline the

data, we combined positions like "ST" and "LS" into "CF." The **“Position”** column was made simpler and less redundant by this cleaning process. Using string manipulation function in Python, we eliminated the 'cm' and 'kg' units from each row to clean the **'Height'** and **'Weight'** columns in the dataset. To ensure consistency and ease of use for additional analysis and model training, the remaining numerical values were then converted into integer format for both columns.

3.2.4 Descriptive Stats

This section presents an analysis and summary of the dataset using different descriptive statistics, giving information about the distribution, dispersion, and central tendency of the numerical data.

Table 5. Descriptive stats of player attributes

	Age	Overall	International Reputation	Weak Foot	Skill Moves	Crossing	Finishing	HeadingAccuracy	ShortPassing	Volleys	...
count	16710.000000	16710.000000	16710.000000	16710.000000	16710.000000	16710.000000	16710.000000	16710.000000	16710.000000	16673.000000	...
mean	25.727409	67.646320	1.169958	3.008199	2.475464	52.212448	48.725075	54.123339	61.314423	45.652972	...
std	5.048910	6.457695	0.485305	0.681742	0.791414	17.772348	19.401715	17.007831	13.665353	17.828225	...
min	16.000000	28.000000	1.000000	1.000000	1.000000	7.000000	3.000000	5.000000	8.000000	4.000000	...
25%	22.000000	63.000000	1.000000	3.000000	2.000000	42.000000	34.000000	46.000000	57.000000	33.000000	...
50%	25.000000	68.000000	1.000000	3.000000	2.000000	57.000000	53.000000	57.000000	64.000000	48.000000	...
75%	29.000000	72.000000	1.000000	3.000000	3.000000	65.000000	64.000000	66.000000	70.000000	59.000000	...
max	54.000000	93.000000	5.000000	5.000000	5.000000	94.000000	95.000000	93.000000	94.000000	90.000000	...

8 rows × 36 columns

Central Tendency:

Mean: Indicates the average value for every variable, giving information about the average value for every player.

The dataset's average value is indicated by the median (50th percentile), which is significant when there are outliers present because they can cause the mean to become skewed.

Standard deviation: The standard deviation (std) shows the degree of variation from the mean. Greater variability in player attributes is indicated by a larger standard deviation.

Distribution: Min and Max numbers indicate the range of the data, ranging from each variable's lowest to highest recorded value.

The dataset is divided into four sections by the 25th and 75th percentiles, which aid in our understanding of how values are distributed in respect to the median. The value below which 25% of the data falls is represented, for instance, by the 25th percentile. For example, players

range in age from 16 to 54 years old, with an average age of about 25 years. Most players fall within the interquartile range of 22 to 29 years. Majority players are in their prime playing years, as evidenced by their physical characteristics and age distributions, and they display moderate skills in areas like crossing, finishing, and passing.

3.2.5 Visualizations

Python data visualization is essential for analysing complex datasets. Libraries like **Matplotlib** and **Seaborn** are used for creating various plots, with Matplotlib offering flexibility and Seaborn providing a high-level interface for quick and informative statistical visualizations (Kharwal, 2023).

3.2.5.1 Univariate Visualizations

1. Count of Players by Club (Top 10)

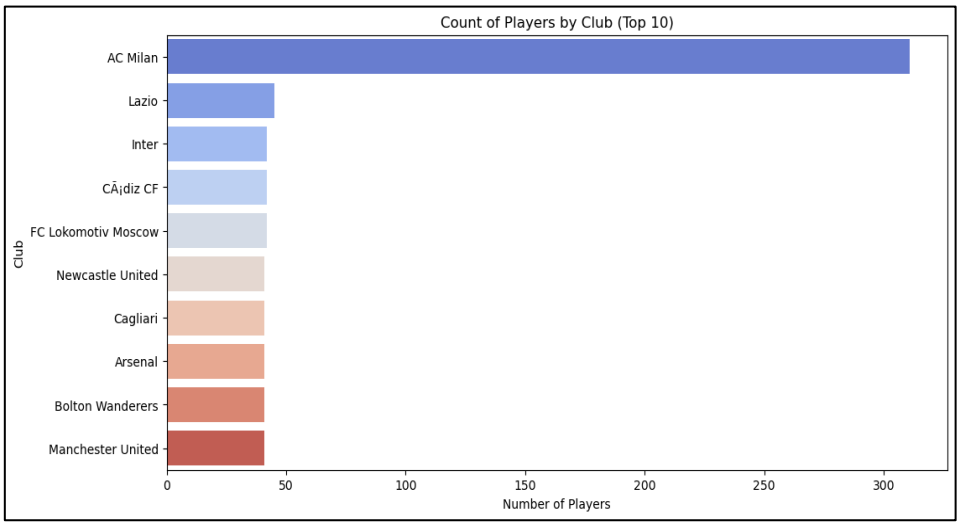


Figure 3. Count of players by club (Top 10)

The top 10 football teams with the most players can be seen in this bar graph. With around 300 players, AC Milan has the greatest number of players over Lazio and Inter, who each have about 40-45 players. Other Clubs like Manchester United, Arsenal, and Bolton Wanderers are also featured with a less player as compared to AC Milan.

2. Number of Players by Nationality (Top 10)

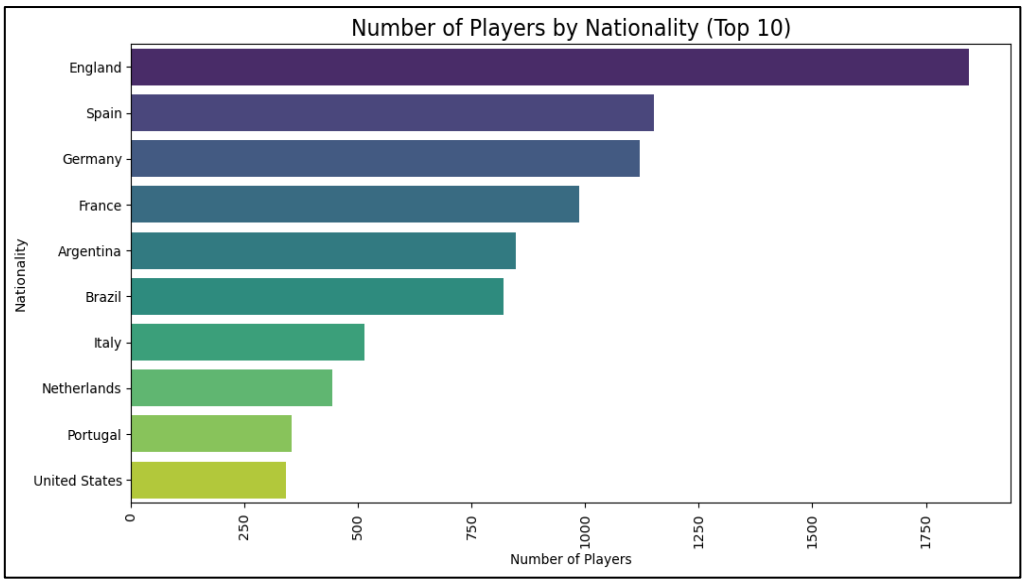


Figure 4. Count of players by nationality (Top 10)

The top 10 player nationalities are shown in this bar chart, with over 1,750 players from England, by far the largest number indicate that players start playing football from a young age and the sport is very popular in the country. Spain, Germany and France are also well-known footballing nations with each nation having players over thousand.

3. Pie Chart of Preferred Foot

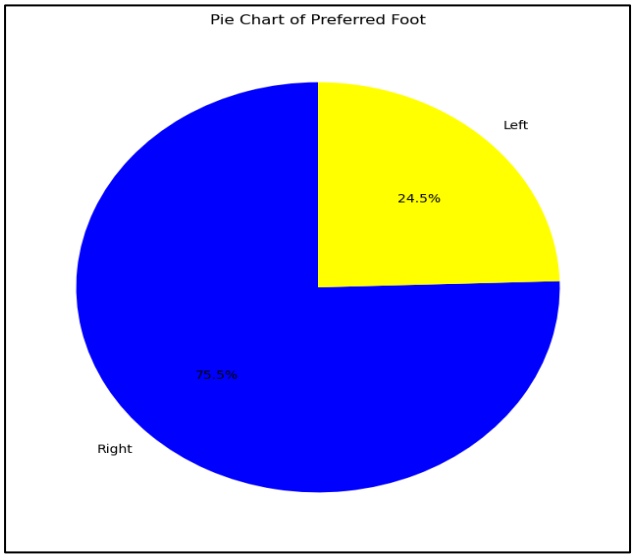


Figure 5. Pie chart of Preferred Foot

Players preferred foot distribution is displayed in this pie chart. Out of all the players, 75.5% players prefer to use their right foot and 24.5% prefer to use their left foot. This suggests that a greater percentage of football players are right-footed than left-footed. Although players who play with both feet have similar skill level but there are a few players who perform exceptionally well with left foot for e.g., Lionel Messi and Gareth Bale.

4. Height Distribution of players

The height distribution of the players is shown in this histogram. Most players are between 175 and 190 cm height, with 180 cm being the most typical height of any player. Less players are found at the extremes very few are shorter than 160 cm or taller than 200 cm and the distribution is more normal around the average height.

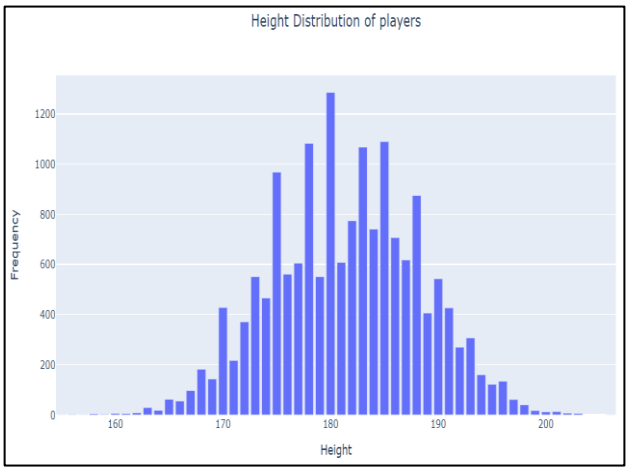


Figure 6. Height distribution of players

5. Weight Distribution of players

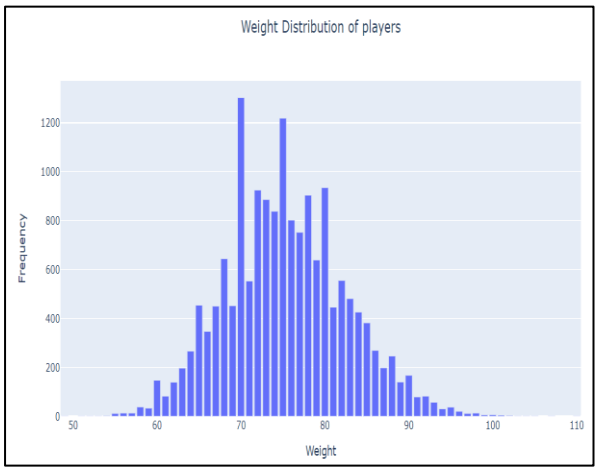


Figure 7. Weight distribution of players

The players weight distribution is displayed in this histogram. Most players weigh between 65 kg and 85 kg, with 70 kg and 75 kg being their peak weights. Less players weigh less than 60 kg or more than 90 kg, suggesting a normal distribution with extreme values being uncommon and centred around average player weights.

6. Age Distribution of players

The age distribution of the players can be seen in this histogram, with most players being in the 20-35 age range. Fewer players are over 35, and the peak age group is about 20-25. There are relatively few players over 40 in the distribution, which indicates a steady decline in player numbers as age rises. Many clubs tend to recruit young players (20-25).

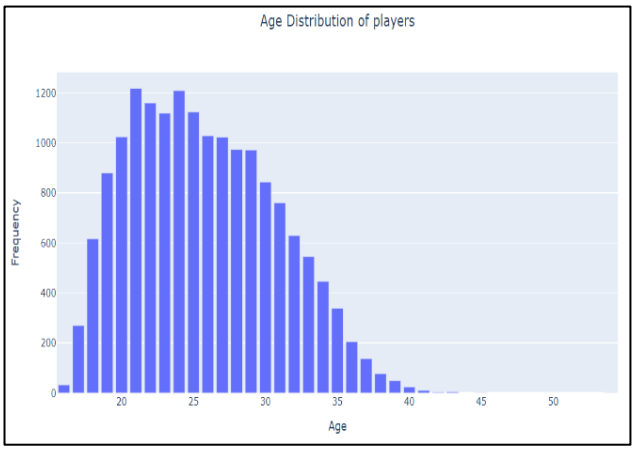


Figure 8. Age distribution of players

7. Top 10 Tall Players

	Name	Height	Position
16176	T. HolÃ½	206	GK
15358	20Â M. Casey	203	CB
12448	20Â L. TraorÃ©	203	CF
14577	P. Ndiaye	203	CB
13126	F. MuliÄ□	203	CF
15678	A. Noppert	203	GK
15751	21Â C. Pantilimon	203	GK
16591	S. Brolin	202	GK
15204	C. Ezekwem	202	CB
16197	K. Scherpen	202	GK

Table 6. List of Top 10 tall players

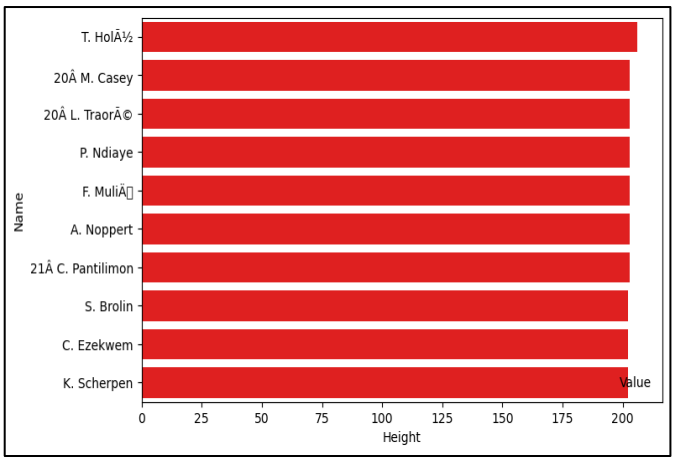


Figure 9. Top 10 tall players

A player's height is displayed in the chart. At 206 cm, T. Holý is the tallest player, followed by M. Casey and L. Traoré, who are both 203 cm tall. Most of the tall players are central backs (CB) or goalkeepers (GK), emphasising the value of height in defensive positions.

8. Top 10 Fastest players

	Name	SprintSpeed	Position
82	K. MbappÃ©	97.0	CF
2202	Adama TraorÃ©	96.0	RW
210	A. Davies	96.0	LB
4852	G. Holtmann	95.0	LW
58	A. Hakimi	95.0	RB
1967	D. James	95.0	RW
2101	VinÃcius Jr.	95.0	RW
587	IÃ±aki Williams	94.0	CF
7958	Igor Madinha	94.0	RW
511	F. Acheampong	94.0	LW

Table 7. List of top 10 fastest players

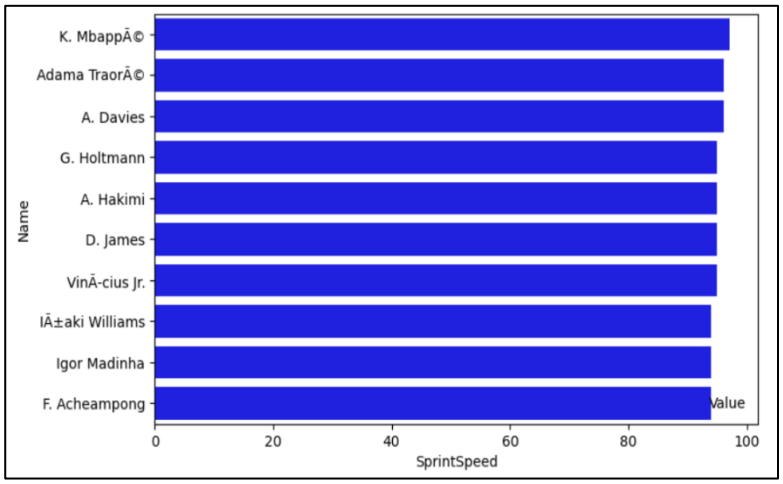


Figure 10. Top 10 fastest players

Based on their sprint speed, the top 10 fastest football players are showcased in the above graphs. With a sprint speed rating of 97, K. Mbappe was considered to be the fastest player in FIFA 22, followed by Adama Traore and A. Davies, who both have a sprint speed rating of 96.

9. Top 10 heaviest Players

	Name	Weight	Position
11849	A. Akinfenwa	110	CF
16034	C. Seitz	107	GK
16592	L. Watkowiak	105	GK
11502	O. OularÃ©	104	CF
15981	L. Unnerstall	103	GK
16128	E. Johansen	102	GK
15529	B. Hamid	102	GK
16176	T. HolÃ½	102	GK
16123	21Â R. Gilmartin	101	GK
14755	21Â W. Morgan	101	CB

Table 8. List of top 10 heaviest players

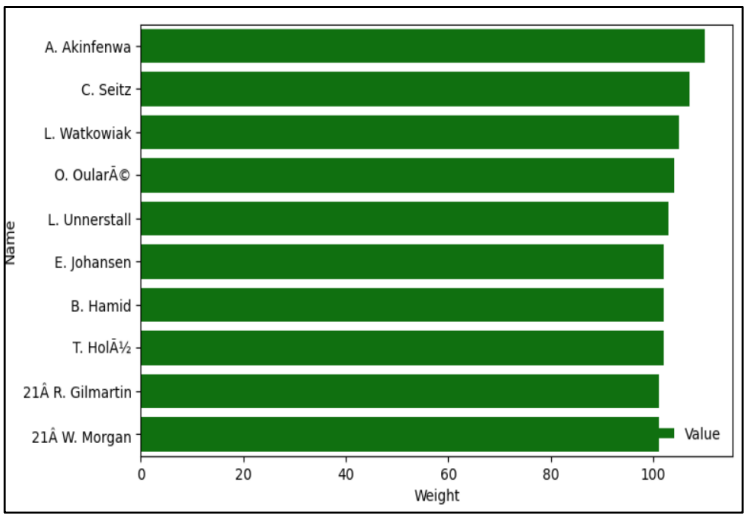


Figure 11. Top 10 heaviest players

The top 10 heaviest football players are shown in the above charts, with A. Akinfenwa at 110 kg being the heaviest player. We can see that goalkeepers (GK) are the heaviest players in a team as they tend to be tall and have a muscular physic than other players. Weight is crucial, especially for GK roles that need physical strength and height.

10. Top 10 players based on their Overall.

	Name	Overall	Potential	Club	Nationality	Position
29	L. Messi	93	93	Paris Saint-Germain	Argentina	RW
33	R. Lewandowski	92	92	FC Bayern MÃ¼nchen	Poland	CF
14244	J. Oblak	91	93	AtlÃ©tico de Madrid	Slovenia	GK
3	K. De Bruyne	91	91	Manchester City	Belgium	CM
64	Neymar Jr	91	91	Paris Saint-Germain	Brazil	LW
82	K. MbappÃ©	91	95	Paris Saint-Germain	France	CF
36	Cristiano Ronaldo	91	91	Manchester United	Portugal	CF
39	H. Kane	90	90	Tottenham Hotspur	England	CF
13890	M. ter Stegen	90	92	FC Barcelona	Germany	GK
71	N. KantÃ©	90	90	Chelsea	France	CDM

Table 9. List of top 10 players with highest overall

The top 10 football players are shown in this table according to their overall ratings, with L. Messi leading the way having a rating of 93, followed by R. Lewandowski (92) and J. Oblak (91). While there are players from a variety of positions on the list, CF dominate the list.

11. Top 10 players based on their Potential

	Name	Potential	Overall	Club	Nationality	Position
82	K. MbappÃ©	95	91	Paris Saint-Germain	France	CF
251	E. Haaland	93	88	Borussia Dortmund	Norway	CF
14608	G. Donnarumma	93	89	Paris Saint-Germain	Italy	GK
14244	J. Oblak	93	91	AtlÃ©tico de Madrid	Slovenia	GK
29	L. Messi	93	93	Paris Saint-Germain	Argentina	RW
444	P. Foden	92	84	Manchester City	England	CAM
33	R. Lewandowski	92	92	FC Bayern MÃ¼nchen	Poland	CF
588	K. Havertz	92	84	Chelsea	Germany	CAM
24	T. Alexander-Arnold	92	87	Liverpool	England	RB
13890	M. ter Stegen	92	90	FC Barcelona	Germany	GK

Table 10. List of top 10 players with highest potential

The top 10 football players by potential ratings are displayed in this table, K. Mbappé has the highest potential rating of 95 indicating that he has a great potential to be a great player in future followed by E. Haaland, G. Donnarumma and J. Oblak with 93 each. We can see that players from PSG have high potential rating as compared to other clubs.

3.2.5.2 Bivariate Visualizations

1. Boxplot of Overall vs preferred foot

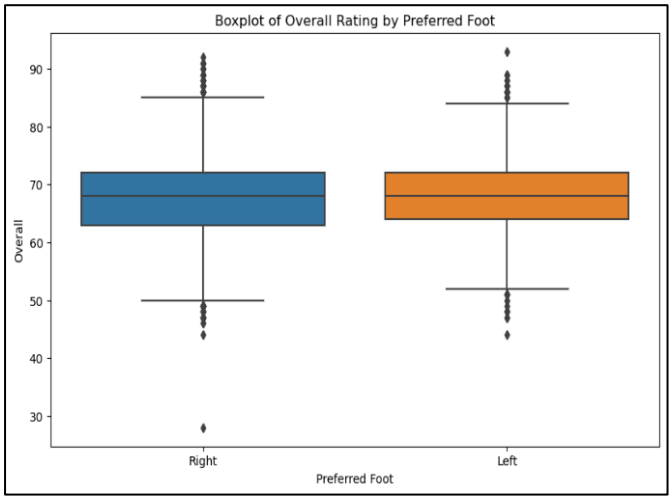


Figure 12. Graph of Overall vs preferred foot

Based on their preferred foot, a player's overall ratings are compared in this boxplot. For both left- and right-footed players, the median rating is approximately 70. For both groups, the distribution is similar, with a few outliers in each category. Compared to left-footed players, right-footed players have a marginally wider range of overall ratings.

2. Bar plot of Overall vs position

The average overall ratings of players at various positions are compared in this bar plot. players with the highest average rating are right-wingers (RW), goalkeepers (GK), and centre forwards (CF). Right-backs (RB) also have relatively high ratings but attacking midfielders (CAM) have slightly lower average ratings. There is not much difference in ratings between these positions.

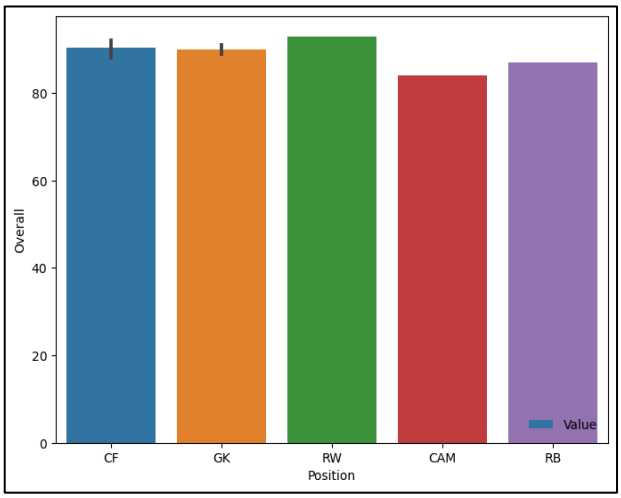


Figure 13. Graph of Overall vs position

3. Scatter plot of Age vs Potential

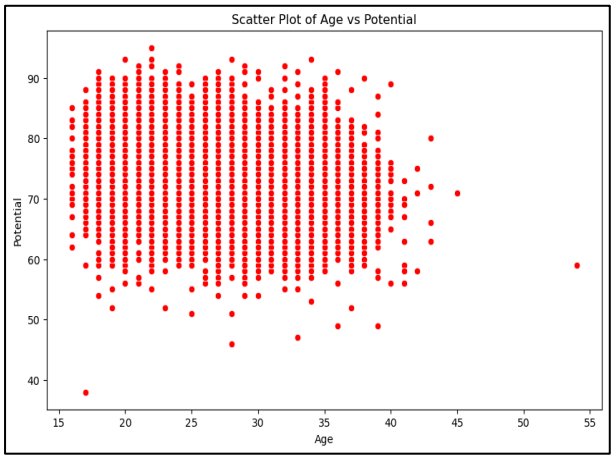


Figure 14. Graph of Age vs Potential

The correlation between age and prospective ratings is displayed in this scatter plot. There is a wide range of potential among players between the ages of 15 and 30, many of whom have high potential ratings (80-90). Players' potential ratings tend to decline as they get older, with fewer players displaying high potential after age of 35.

4. Scatter plot of Weight vs Sprint Speed

The weight and sprint speed of players are indirectly related to each other. Players with low weight (60-80 kg) typically have faster sprint speed rating of up to 80. Sprint speed typically decline as weight increases above 80 kg. Hence, very few players over 100 kg can achieve faster sprint speeds.

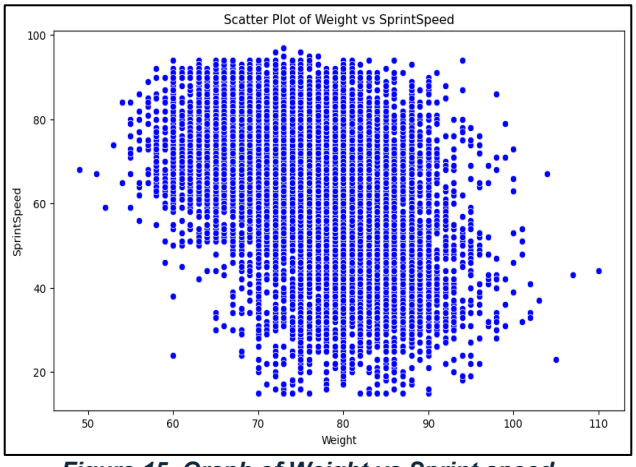


Figure 15. Graph of Weight vs Sprint speed

5. Scatter plot of Strength vs Shot Power

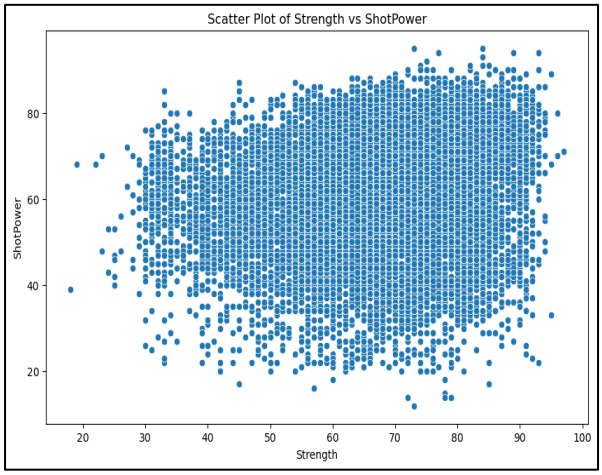


Figure 16. Graph of Strength vs Shot Power

The scatter plot shows the corelation between strength and Shot power. Shot power rating is typically higher in players with greater strength (above 70), with ratings frequently surpassing 70. Players with strength levels below 50 typically possess a broad range of shot power, suggesting that shot power is also impacted by attributes other than strength.

6. Pie Chart of Average Overall Rating by Skill Moves

The average overall rating by skill moves is shown in this pie chart. Most of the players have the highest skill move rating of 5, at 22.2%, followed by players with a skill move rating of 4, at 20.8%. There is a more even distribution of players with lower skill move ratings (1 to 3), with roughly 18-19.5% of the total belonging to each group

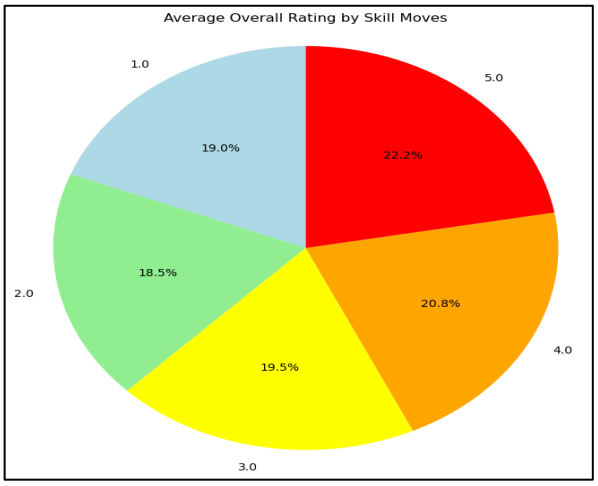


Figure 17. Graph of Average Overall Rating by skill Moves

7. Bar plot of Overall vs Player Count

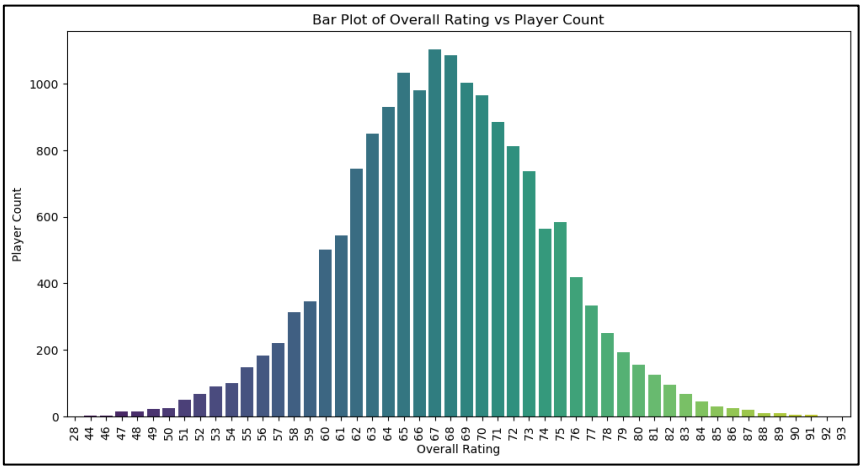


Figure 18. Graph of Overall vs Player count

We can see the distribution of overall player ratings in this histogram, with most players having ratings between 60 and 80. With a peak at 67, the distribution is roughly normal, meaning that most players are rated in the mid-range. Very few players have ratings of lower than 50 or higher than 80.

8. Joint Plot of Overall vs Age

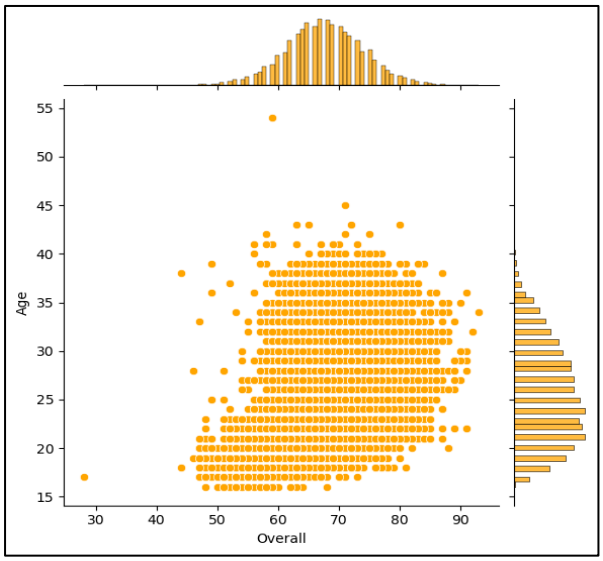


Figure 19. Graph of Overall vs Age

With marginal histograms illustrating the distributions of each variable, this graph illustrates the correlation between age and overall rating. players in the 20-35 age range typically have a broad range of high overall ratings, with a focus between 60 and 70. Players tend to have slightly lower overall ratings as they get older than the age of 35.

9. Count plot of Preferred Foot vs Position

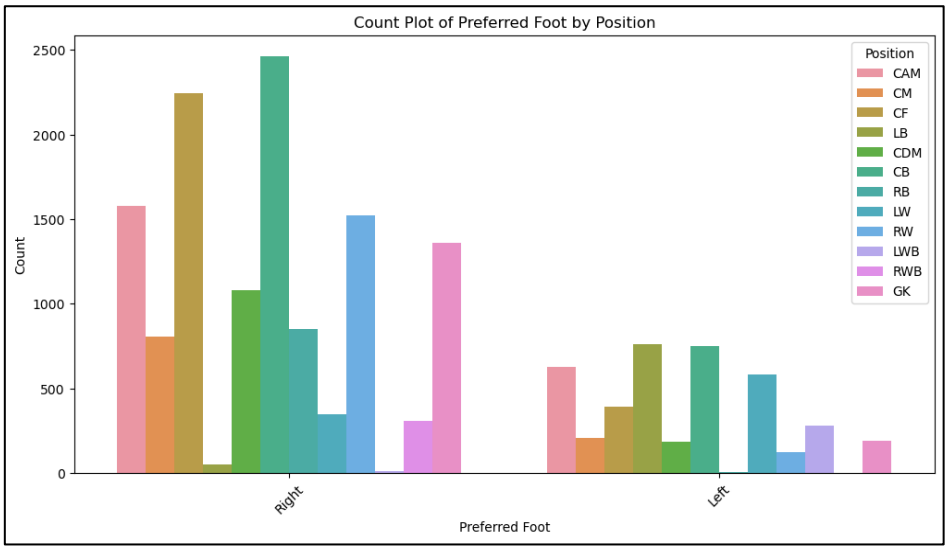


Figure 20. Graph of Preferred Foot vs Position

The distribution of players' preferred foot among various positions is displayed in this count plot. Most players are right-footed, particularly those who play central defensive midfielder (CDM), centre forward (CF) and goalkeepers (GK). left-footed players are keener towards playing in defensive positions like LB and CB.

3.2.5.3 Multivariate Visualizations

1. Distribution of players skill related variables

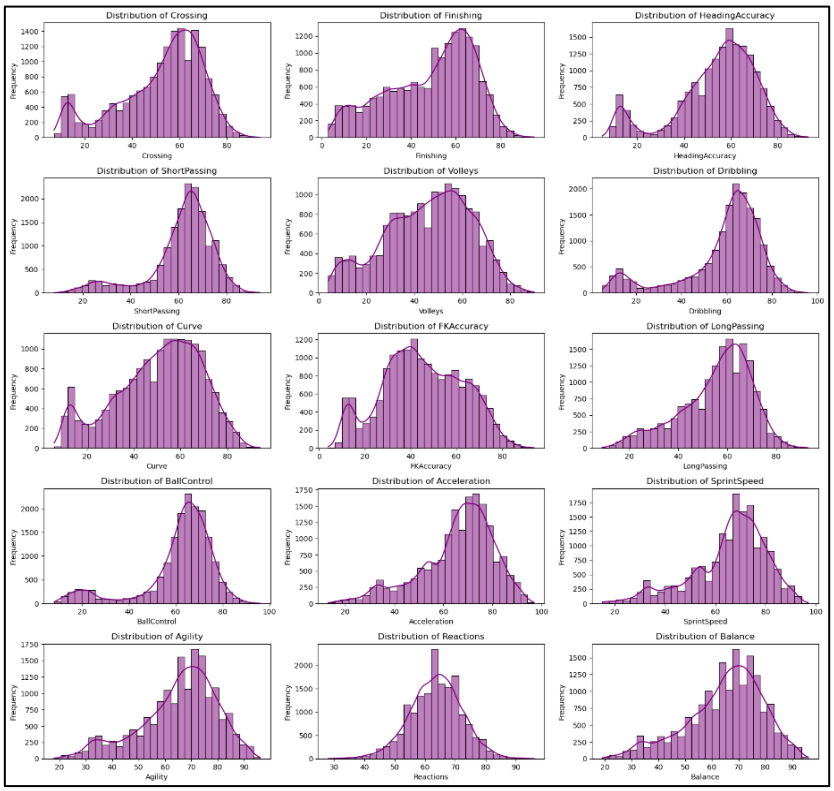


Figure 21. Distribution of skill related variables

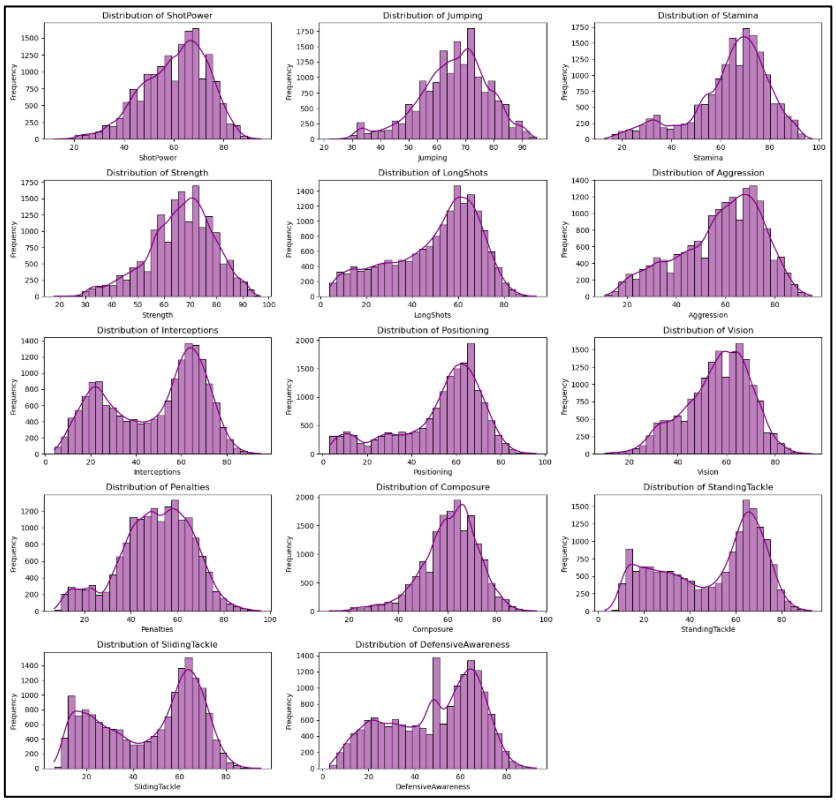


Figure 22. Distribution of skill related variables

The distribution of different football attributes, such as crossing, finishing, dribbling, shot power, strength etc among players are displayed in these histograms. Most players score between 50 and 70 in these skills, with most attributes having a normal distribution. Certain metrics, such as Defensive Awareness and Interceptions, exhibit greater variability than others, suggesting a greater variation in player performance in these domains.

2. Correlation Matrix

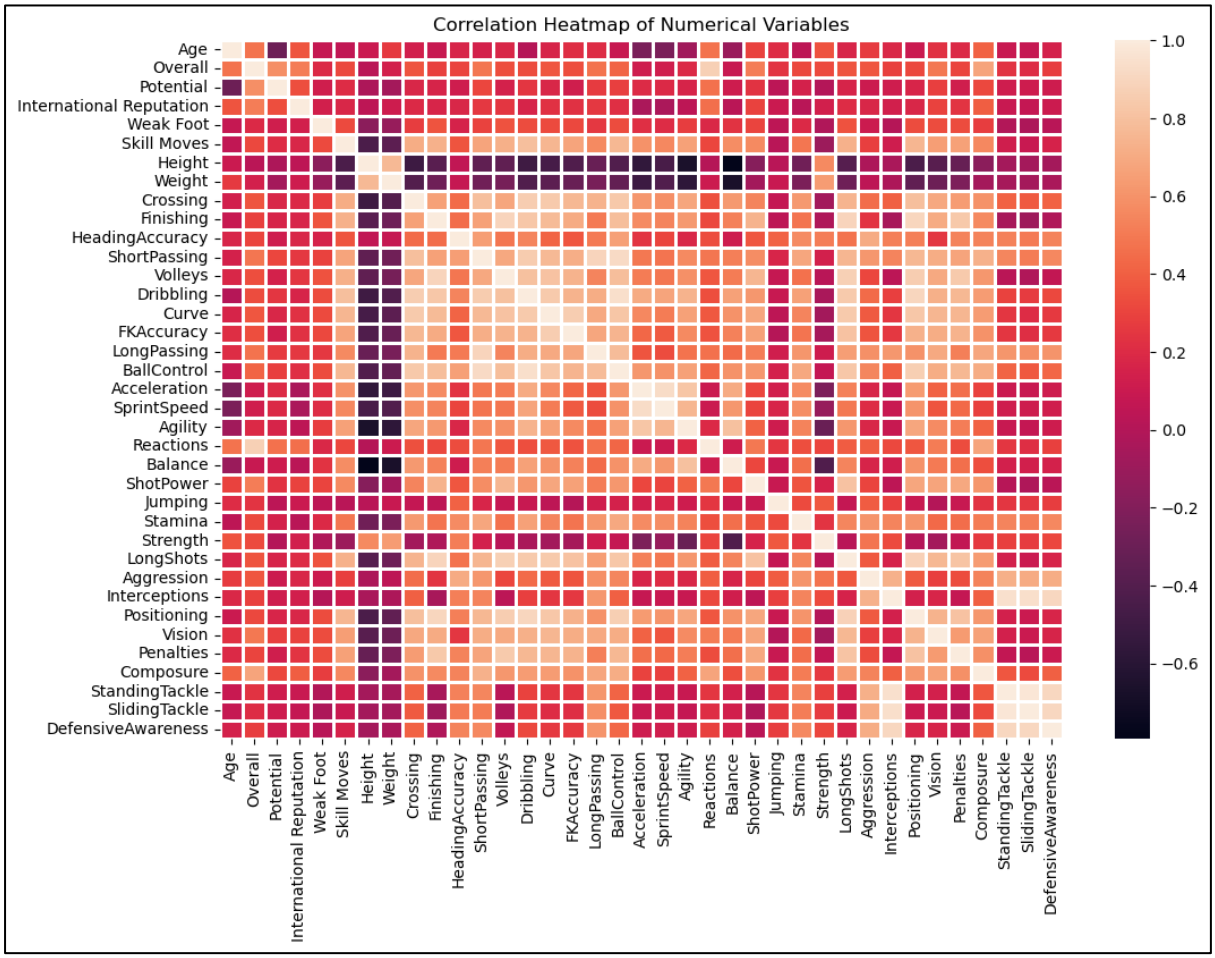


Figure 23. Correlation Matrix

The correlation matrix of continuous variables is shown in this heatmap. Strong Positive & negative correlations are indicated by lighter and darker shades respectively. There is strong positive correlation between pairs like Potential-Overall, Dribbling-BallControl, Interceptions-StandingTackle, StandingTackle-SlidingTackle etc. Most variables have weaker correlations with attributes like age. Performance analysis and model building can benefit from the identification of relationships between player attributes, which is made easier by this Heatmap.

3.2.6 Train Test Split

The dataset was divided using an 80-20 ratio into a test set and a train set to assess the predictive model's performance. 13,368 instances (80% of the data) were used in the training set, which was used to train the model. The remaining 3,342 instances (20% of the data) were designated for testing the model's performance on unseen data. To ensure reproducibility of the results, a random seed (random_state=7) was used for the split.

Since the data was going to be split randomly between train and test sets, we added a new column named **“Original Index"** where each row got its original index value from the dataset. This column maintained the original indexing prior to the train test split process, since we will be using this column when we combine the Predicted Overall values obtained from our top performing model on the train & test sets.

3.2.7 One Hot Encoding

One-Hot Encoding was used to convert the categorical variable **"Preferred Foot"** into numerical values to prepare the dataset for machine learning models. With this encoding method, new binary columns were produced for the categorical variable. For instance, a new column called **"Right"** was created with the "Preferred Foot" column, a value of 1 designates a right-footed player, and a value of 0 indicates the opposite.

To guarantee consistent encoding between the two sets, **OneHotEncoder** was first fitted on the training set and test set. After encoding, the initial categorical column **"Preferred Foot"** was removed since it was no longer required. (*The list of removed variables is mentioned in the “Feature selection” section)*

3.2.8 Feature Scaling

To normalise the data for the further analysis, Feature Scaling was utilized with the StandardScaler from the sklearn.preprocessing module. To avoid the model learning this target value during the scaling process, the target variable **"Overall"** was first eliminated before scaling the dataset. The features were then scaled with a mean of 0 and a standard deviation of 1 using StandardScaler. The scaled data was transformed back into a Data Frame and the target variable **("Overall")** was reattached after the scaler was applied to the predictors.

To guarantee uniform scaling across the dataset, this procedure was carried out again for the training and test datasets. For additional model training and assessment, the scaled training and test datasets were split into predictor variables (X) and the target variable (y). This

technique makes sure that every feature contributes equally to the predictive model, which enhances model performance. *(For Trainset & Testset Head after Feature scaling refer to Appendix A)*

3.2.9 Feature Selection

Feature selection is a crucial step in the machine learning pipeline, as it helps improve model accuracy by reducing the dimensionality of the data and eliminating irrelevant or redundant features (Kharwal, 2023).

The List of Variables removed from the dataset for Model prediction are mentioned in Table 4

The following variables were dropped to make the analysis easier:

In the correlation matrix, these pairs were highly correlated with each other. We had to drop the variables 'StandingTackle' & 'SlidingTackle' as they were common in the correlated pair with interception. Hence, we kept Interceptions and dropped them. Not dropping these columns could increase the errors and decrease the efficiency of the analysis.

```
[('Dribbling', 'BallControl'),
 ('Interceptions', 'StandingTackle'),
 ('Interceptions', 'SlidingTackle'),
 ('StandingTackle', 'SlidingTackle')]
```

Figure 24. Highly Correlated variables

Categorical Columns:

- **Name**: A player's name is a unique identifier with no predictive value. Hence, we dropped it.
- **Nationality**: While it may have some cultural significance, it doesn't directly influence a player's performance and has too many unique values, which would increase model complexity.
- **Club**: The player's club was removed for similar reasons, as it contained 869 unique values that would unnecessarily increase dimensionality without improving the model's accuracy.
- **Position**: We dropped Position as it was a categorical variable containing unique values which was leading to complexity in Model prediction.

The original column **"Preferred Foot"** was also dropped as it was encoded and converted into numeric variable **"Right"** to prepare the dataset for model prediction.

Table 11. Final list of removed variables for machine learning models

VARIABLES	DEFINITION
ID	Unique identifier assigned to each player.
Photo	Link to the player's photo.
Flag	Flag representing the player's nationality.
Club Logo	Logo of the club the player is associated with.
Value	(€) Estimated market value of the player.
Wage	(€) salary paid by the club to the player.
Special	Total number of special traits or unique abilities possessed by the player.
Work Rate	Describes the player's work rate both in attacking and defensive situations.
Body Type	Describes the player's physique or body shape.
Real Face	Indicates whether the player's in-game face is an authentic recreation.
Jersey Number	The number worn by the player on their jersey.
Joined	The date when the player joined their current club.
Loaned From	The name of the club the player is loaned from (if applicable).
Contract Valid Until	The year until which the player's contract with the club is valid.
Marking	The player's ability to track and mark opponents defensively.
GKDiving	The goalkeeper's ability to dive and save shots aimed at the goal.
GKHandling	The goalkeeper's skill in catching or deflecting the ball.
GKKicking	The goalkeeper's ability to accurately kick the ball.
GKPositioning	The goalkeeper's awareness and positioning when defending the goal.
GKReflexes	The goalkeeper's ability to react quickly to shots.
Release Clause	The value that must be paid to release the player from their contract.
Preferred Foot	Indicates if the player prefers to use their right or left foot.
Best Position	The player's best or most effective playing position on the field.
Best Overall Rating	The player's best overall rating based on their optimal position.
StandingTackle	The player's proficiency in performing standing tackles.
SlidingTackle	The player's proficiency in executing sliding tackles.

Name	Player's name.
Nationality	Country the player represents.
Club	The team to which the player belongs.
Position	The primary position the player plays on the field.

3.3 Machine Learning Algorithms

The accuracy of regression models is commonly assessed using the following evaluation metrics: **R-squared (R^2)** and **Root Mean Squared Error (RMSE).** To help measure the error in units of the target variable, RMSE offers an estimate of the average deviation between predicted and actual values. When handling normally distributed errors in particular, a lower root mean square error (RMSE) denotes superior model performance (Aporia, 2023).

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

$$R^2 = 1 - (RSS/TSS)$$

The R-squared (R^2) evaluates how well the independent variables account for the fluctuations in the dependent variable. It has a range of 0 to 1, with values closer to 1 denoting a well-fitting model to the data. The two metrics are complementary in that they provide the absolute error magnitude (RMSE) and the proportion of explained variance (R^2), which is useful for model comparison (Zach, 2021).

We will be running 5 Machine learning models using grid search as this hyperparameter tuning algorithm was proved to be the simplest one. Essentially, we established a discrete grid within the hyperparameter range, tested every possible combination of grid values, and then cross-validated a few performance metrics. The grid point that maximised the average value during cross-validation was the optimal place to set the hyperparameters.

Since we are using RMSE as the evaluation metric in GridSearchCV with the greater_is_better=False parameter, our models will have negative mean train and mean test scores. Since, lower RMSE indicates better model performance, this setting is deemed appropriate for error-based metrics.

3.3.1 Baseline Model (Linear Regression)

Linear Regression model was used as the baseline Model for prediction. A variable's value can be predicted using linear regression analysis based on the value of another variable. The dependent variable is the one that we want to predict. The independent variable is the one we use to forecast the value of the other variable (IBM, 2023).

The results for Training RMSE were **1.9128** and the Test RMSE were **1.9223** indicating a minimal difference of **0.0095**. This small amount of difference indicates that our baseline model generalized well without overfitting to the training data.

The R2 score of **0.9084** shows that 90% of the variance in player performance can be explained by the model, demonstrating a strong prediction accuracy.

Table 12. Model Performance of Linear Regression (Baseline Model)

Model	RMSE	R2 Score
Linear Regression (Baseline Model)	1.9223	0.9084

These results suggest that the Linear Regression model provided a decent baseline for the analysis, offering a reliable starting point for comparison of more intricate models. *(Refer to Appendix A to see the output after running this model in Python)*

3.3.2 Random Forest

In machine learning, regression is an ensemble technique that uses multiple decision trees along with a technique called Bootstrap and Aggregation, or bagging, to perform both regression and classification tasks. The general idea here is to use a combination of decision trees instead of depending only on one to determine the result (Dutta, 2019).

GridSearchCV was used to train the Random Forest Regressor to adjust the hyperparameters and determine the ideal values:

'n_estimators': [10, 30, 50]

'max_depth': [5, 10, 15]

'min_samples_split': [10, 25, 30]

After training the model with Fitting 3 folds of 27 distinct combinations of hyperparameters, totalling 81 fits, the optimal combination was identified as follows:

Table 13. Random Forest RMSE based on hyperparameter combinations

	params	mean_train_score	mean_test_score	diff, %
20	{'max_depth': 15, 'min_samples_split': 10, 'n_estimators': 50}	-0.666002	-1.231837	-84.959855
19	{'max_depth': 15, 'min_samples_split': 10, 'n_estimators': 30}	-0.679892	-1.245013	-83.119160
23	{'max_depth': 15, 'min_samples_split': 25, 'n_estimators': 50}	-0.933441	-1.295810	-38.820771
18	{'max_depth': 15, 'min_samples_split': 10, 'n_estimators': 10}	-0.746050	-1.305800	-75.028665
22	{'max_depth': 15, 'min_samples_split': 25, 'n_estimators': 30}	-0.944079	-1.305803	-38.314908
26	{'max_depth': 15, 'min_samples_split': 30, 'n_estimators': 50}	-1.000386	-1.320741	-32.023121
11	{'max_depth': 10, 'min_samples_split': 10, 'n_estimators': 50}	-0.945212	-1.325024	-40.182689
25	{'max_depth': 15, 'min_samples_split': 30, 'n_estimators': 30}	-1.011405	-1.330724	-31.571823
10	{'max_depth': 10, 'min_samples_split': 10, 'n_estimators': 30}	-0.956857	-1.336545	-39.680715
21	{'max_depth': 15, 'min_samples_split': 25, 'n_estimators': 10}	-0.990665	-1.353645	-36.640041
14	{'max_depth': 10, 'min_samples_split': 25, 'n_estimators': 50}	-1.065549	-1.360761	-27.705216
13	{'max_depth': 10, 'min_samples_split': 25, 'n_estimators': 30}	-1.074544	-1.369687	-27.466813
24	{'max_depth': 15, 'min_samples_split': 30, 'n_estimators': 10}	-1.053503	-1.373657	-30.389525
17	{'max_depth': 10, 'min_samples_split': 30, 'n_estimators': 50}	-1.105791	-1.376548	-24.485393
16	{'max_depth': 10, 'min_samples_split': 30, 'n_estimators': 30}	-1.115830	-1.385602	-24.176863
9	{'max_depth': 10, 'min_samples_split': 10, 'n_estimators': 10}	-1.008261	-1.389946	-37.855759
12	{'max_depth': 10, 'min_samples_split': 25, 'n_estimators': 10}	-1.117372	-1.419253	-27.017019
15	{'max_depth': 10, 'min_samples_split': 30, 'n_estimators': 10}	-1.155972	-1.430832	-23.777363
8	{'max_depth': 5, 'min_samples_split': 30, 'n_estimators': 50}	-2.134581	-2.185716	-2.395549
5	{'max_depth': 5, 'min_samples_split': 25, 'n_estimators': 50}	-2.134581	-2.185716	-2.395549
2	{'max_depth': 5, 'min_samples_split': 10, 'n_estimators': 50}	-2.134581	-2.185716	-2.395549
1	{'max_depth': 5, 'min_samples_split': 10, 'n_estimators': 30}	-2.139463	-2.190645	-2.392315
7	{'max_depth': 5, 'min_samples_split': 30, 'n_estimators': 30}	-2.139463	-2.190645	-2.392315
4	{'max_depth': 5, 'min_samples_split': 25, 'n_estimators': 30}	-2.139463	-2.190645	-2.392315
6	{'max_depth': 5, 'min_samples_split': 30, 'n_estimators': 10}	-2.163043	-2.217218	-2.504591
3	{'max_depth': 5, 'min_samples_split': 25, 'n_estimators': 10}	-2.163043	-2.217218	-2.504591
0	{'max_depth': 5, 'min_samples_split': 10, 'n_estimators': 10}	-2.163043	-2.217218	-2.504591

With max_depth: 15, min_samples_split: 10, and n_estimators: 50, the best-performing model produced the highest test score with a slight overfitting risk. Though less accurate, models with smaller max_depth and higher min_samples_split were more stable. Test performance was enhanced by deeper trees, but overfitting risk was raised. low difference between training and test scores, indicated the model generalized well and doesn't overfit heavily.

Model Performance:

Table 14. Model Performance of Random Forest Model

Model	RMSE	R2 Score
Random Forest	1.2120	0.9636

The model's RMSE of **1.2120** indicates average prediction deviation from actual values, while its R2 Score of **0.9636** indicates it explains 96.3% of player performance variance.

3.3.3 Decision Tree

A machine learning method called decision tree regression builds a model resembling a tree to forecast continuous numerical values. Decision tree regression is more concerned with estimating numerical results than classification tasks, which yield a categorical output (Viswa, 2023)

Again, in this Model we used GridSearchCV to optimise the model by utilising 5-folds each for 16 different combinations and totalling 80 fits of the following hyperparameters:

'max_depth': [5, 10, 20, 25]

'min_samples_split': [10, 15, 20, 30]

Table 15. Decision tree RMSE based on hyperparameter combinations

	params	mean_train_score	mean_test_score	diff, %
11	{'max_depth': 20, 'min_samples_split': 30}	-1.100355	-1.705762	-55.019225
15	{'max_depth': 25, 'min_samples_split': 30}	-1.100355	-1.705762	-55.019225
7	{'max_depth': 10, 'min_samples_split': 30}	-1.305205	-1.710974	-31.088497
6	{'max_depth': 10, 'min_samples_split': 20}	-1.248047	-1.720043	-37.818794
5	{'max_depth': 10, 'min_samples_split': 15}	-1.216041	-1.731096	-42.355005
14	{'max_depth': 25, 'min_samples_split': 20}	-0.932487	-1.734111	-85.966309
10	{'max_depth': 20, 'min_samples_split': 20}	-0.932671	-1.735729	-86.103130
4	{'max_depth': 10, 'min_samples_split': 10}	-1.182602	-1.741187	-47.233643
13	{'max_depth': 25, 'min_samples_split': 15}	-0.805677	-1.762027	-118.701349
9	{'max_depth': 20, 'min_samples_split': 15}	-0.805814	-1.769324	-119.569935
8	{'max_depth': 20, 'min_samples_split': 10}	-0.615639	-1.806599	-193.451201
12	{'max_depth': 25, 'min_samples_split': 10}	-0.615328	-1.812848	-194.614874
0	{'max_depth': 5, 'min_samples_split': 10}	-2.390882	-2.442399	-2.154762
1	{'max_depth': 5, 'min_samples_split': 15}	-2.390882	-2.442399	-2.154762
2	{'max_depth': 5, 'min_samples_split': 20}	-2.390882	-2.442399	-2.154762
3	{'max_depth': 5, 'min_samples_split': 30}	-2.390882	-2.442399	-2.154762

'max_depth': 20,'min_samples_split': 30} were the parameters of the best-performing model in the table. Its mean test score of **-1.7057** which indicates that it has the lowest error on the validation set. The test and train scores differ by about **-55.019%,** indicating a slight but not significant overfitting of the model. It provided good test performance and strikes a reasonable balance between generalisations.

Model Performance:

Table 16. Model performance of Decision Tree Model

Model	RMSE	R2 Score
Decision Tree	1.7137	0.9272

The Test RMSE showed an average deviation of **1.7137**, with a R2 of **0.9272**, indicating a model that accounts for 92.7% of player performance variation.

3.3.4 Support Vector Machine

According to (IBM, 2023) Support Vector Regression (SVR) is an extension of support vector machines (SVMs) used for regression problems in which the outcome is continuous. SVR, which is commonly used for time series prediction, locates a hyperplane with the maximum margin between data points, much like linear SVMs.

GridSearchCV was used to adjust the SVR model's hyperparameters using 5-fold cross-validation. This Model took slightly longer duration to train as compared to other models.

'C': [0.01, 0.1, 1, 10]

'gamma': ["auto", 0.1],

Fitting 5 folds for each of 8 candidates, totalling 40 fits, the best hyperparameters found were:

Table 17. SVM Models RMSE based on Hyperparameter combinations

	params	mean_train_score	mean_test_score	diff, %
6	{'C': 10, 'gamma': 'auto'}	-0.650327	-0.941925	-44.838701
4	{'C': 1, 'gamma': 'auto'}	-1.120182	-1.215310	-8.492200
7	{'C': 10, 'gamma': 0.1}	-0.327875	-1.564320	-377.108758
2	{'C': 0.1, 'gamma': 'auto'}	-2.078577	-2.101426	-1.099281
5	{'C': 1, 'gamma': 0.1}	-1.830890	-2.156381	-17.777712
3	{'C': 0.1, 'gamma': 0.1}	-3.625847	-3.677759	-1.431707
0	{'C': 0.01, 'gamma': 'auto'}	-4.067693	-4.071437	-0.092039
1	{'C': 0.01, 'gamma': 0.1}	-5.661352	-5.668115	-0.119464

With parameters {'C': 10, 'gamma': 'auto'}, the model that performed best had the lowest error on the test set, with a mean test score of **-0.9419**. Despite some overfitting as indicated by the comparatively small train-test difference of **-44.83%**, this model proved to be our best overall when compared to other models since it achieved good performance metrics (low RMSE and high R^2).

Model Performance:

Table 18. Model Performance of SVM Model

Model	RMSE	R2 Score
Support Vector Machine	1.0262	0.9739

The model with the lowest test RMSE of **1.0262** and R2 score of **0.9739** is the best ML model for player performance prediction, explaining 97.3% of variation.

3.3.5 K Nearest Neighbors

A non-parametric, supervised learning technique, the k-nearest neighbours (KNN) algorithm employs proximity to classify or predict how a single data point will be grouped. It is among the most widely used and straightforward regression and classification classifiers in machine learning today (IBM, 2024).

The model was optimised with 5-folds for each of 10 candidates, totalling 50 fits, using GridSearchCV to determine the ideal number of neighbours for prediction.

'n_neighbors': [8, 9, 10, 11, 12, 13, 14, 15, 16, 17]

The optimal value discovered for n_neighbors was 9:

Table 19. KNN Models RMSE based on Hyperparameter combinations

	params	mean_train_score	mean_test_score	diff, %
1	{'n_neighbors': 9}	-1.734176	-1.949574	-12.420784
3	{'n_neighbors': 11}	-1.772024	-1.950531	-10.073655
2	{'n_neighbors': 10}	-1.755204	-1.951779	-11.199523
0	{'n_neighbors': 8}	-1.714037	-1.952950	-13.938614
4	{'n_neighbors': 12}	-1.785752	-1.953175	-9.375514
5	{'n_neighbors': 13}	-1.798946	-1.956233	-8.743298
6	{'n_neighbors': 14}	-1.810749	-1.957287	-8.092686
8	{'n_neighbors': 16}	-1.833591	-1.959580	-6.871165
7	{'n_neighbors': 15}	-1.823178	-1.959920	-7.500235
9	{'n_neighbors': 17}	-1.843906	-1.962674	-6.441085

When tested with different numbers of neighbours (n_neighbors), the KNN model performed decent. Having a balanced train-test score of **-1.734** & **-1.949** with difference of **-12.42%,** n_neighbors: 9 seemed to be the best parameters, suggesting a reasonable model generalisation with little overfitting.

Model Performance:

Table 20. Model Performance of KNN Model

Model	RMSE	R2 Score
KNN	1.9261	0.9080

The average deviation between the actual values and model's predictions is approximately **1.9261** units, as indicated by the Testset RMSE of **1.9261.** With an R2 of **0.9080**, the model appears to account for about 90.8% of the variation in player performance.

PART B - Team Strength & Opposition Quality

3.4 Data for evaluating team strength and opposition quality

The best Support Vector Machine model found by GridSearchCV was used to predict the Overall on both the training and testing datasets. To create a Data Frame for evaluation, the final model was first applied to the test set and then to the train set, producing "**Predicted Overall"**. The Predicted Overall was compared with the Actual Overall values. *(Refer to Appendix B.1)*

The train and test sets were then combined for a thorough analysis and prediction of every player's **Overall** in the dataset. Since the rows were shuffled and split randomly between Train & Test Sets, **"Original Index"** column was used to return the rows to their original order after making the predictions. The predicted results were then joined with the previous dataset which had the dropped columns such as "Name," "Club," "Position," and "Overall." *(Refer to Appendix B.2)*

To conduct the analysis and visualisation for Part B, the final dataset which included players **"Name", "Club", "Position",** Actual **"Overall"** and **"Predicted Overall"** was saved as a CSV file. The analysis & visualisations for PART B were done using Python.

Table 21. Variables used for Analysis in Part B

VARIABLE	DEFINITION
Name	Player's name.
Club	The team to which the player belongs.
Position	The primary position the player plays on the field.
Overall	An overall rating of player based on rest of the attributes
Predicted Overall	Predicted overall rating of the player (predicted using SVM model)

3.4.1 Team strength

For Team Strength we have chosen **"Real Madrid CF"** as our home team to compare its strength with opponents from **Top 5 European leagues**.

3.4.1.1 Home Teams Actual & Predicted strength **(Based on all players)**

"Team strength" for Real Madrid CF was calculated by averaging the **"Overall"** and **"Predicted Overall"** performance ratings of each player on the team.

Actual Overall Strength: Calculated using the average of the actual **overall** of all players in Real Madrid CF

Predicted Overall Strength: Calculated using the average of **predicted overall** of all players which was predicted using our machine learning model (SVM) in Part A.

We were able to evaluate how well the **Actual Overall Strength,** and the **Predicted Overall Strength** outcomes aligned with this comparison. To further demonstrate the distinctions between these two metrics, visualisations were made using Python.

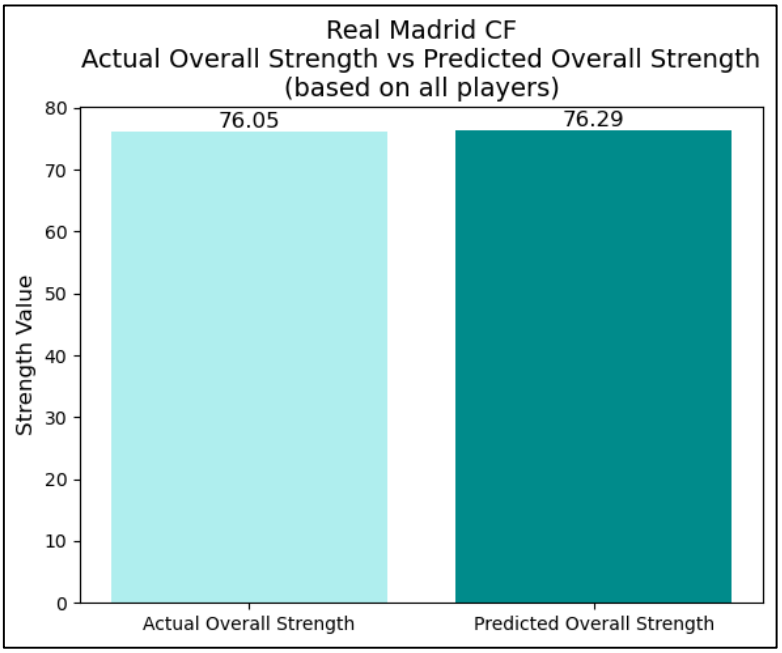


Figure 25. Real Madrid CF Actual Overall strength vs Predicted Overall Strength (Based on all players)

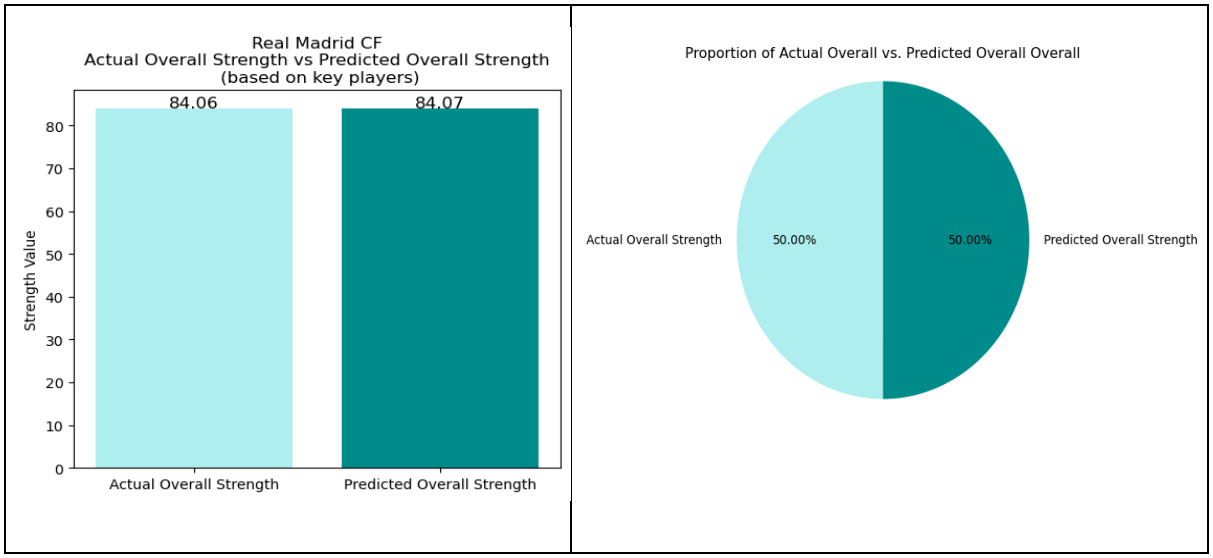
The graph shows the difference between Real Madrid CF's actual overall strength and predicted overall strength which is based on the performance of every player in the team. With only a 0.24-point difference between the actual and predicted strengths, the values indicate

that the predictive model was accurate in estimating the team's overall strength. This suggests a well-performing model with little deviation, enhancing the prediction's dependability.

3.4.1.2 Home Teams Actual & Predicted strength **(Based on key players)**

The Actual Overall and Predicted overall strength based on **key** players was calculated using the same method used in the previous section. We specifically selected **key players** from our dataset by cross validating them with FIFA 22 database, available on SoFifa.com as of September 2021, who had an 'Overall' rating of 80 or higher *(Refer to Appendix C to see the list of key players chosen for each club).* The dataset included former players and younger players **(all players)** who did not significantly contribute to the team's overall performance due to limited appearances or injuries etc. Calculating team strength based on **all players** in the dataset proved to be biased, as it didn't accurately reflect the team's actual performance. Therefore, we focused only on **key players** with substantial contributions based on appearances and performance ratings.

Table 22. Real Madrid CF Actual Overall strength vs Predicted Overall Strength (Based on key players)



The Actual Overall vs. Predicted Overall Strength comparison is almost similar, with the actual being 84.06 and the predicted being 84.07. This suggests that the model's predictions and the actual player strengths of **key players** are very similar. This is further highlighted by the pie chart, which shows an even distribution between the predicted and actual strengths for Real Madrid's overall team performance based on **key players**.

3.4.2 Opposition Quality

To compare our home team's strength with opposition quality we chose opposition teams from Europe's Top 5 Leagues:

- **FC Barcelona** – *Spanish La Liga*
- **Manchester City** – *English Premier League*
- **Paris Saint Germain** – *France Ligue 1*
- **Borussia Dortmund** – *German Bundesliga*
- **Juventus** – *Italian Serie A*

3.4.2.1 Opposition Teams Actual & Predicted strength **(Based on all players)**

Comparison of the Actual Overall Strength and Predicted Overall Strength for opposing teams in our dataset is shown in the graphs below.

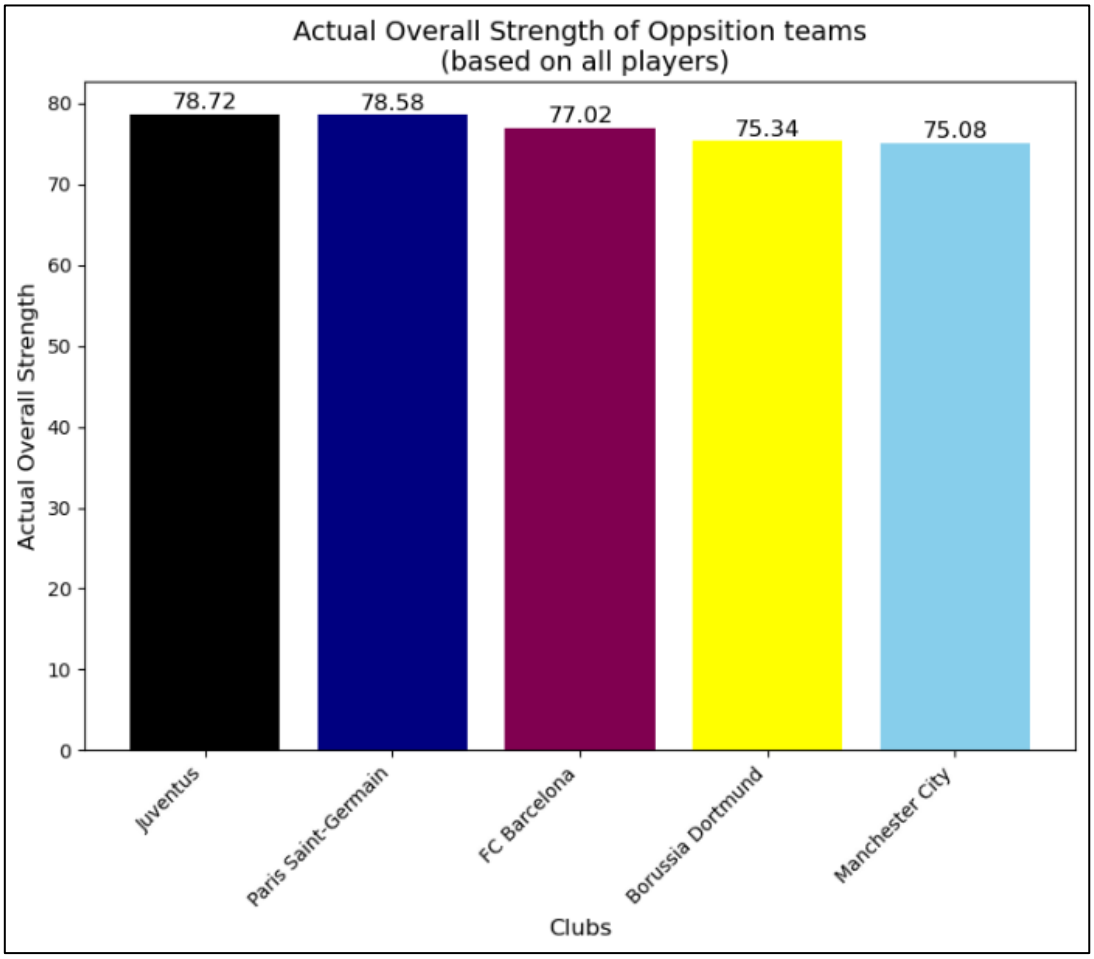


Figure 26. Actual Overall strength of Oppositions (Based on All players)

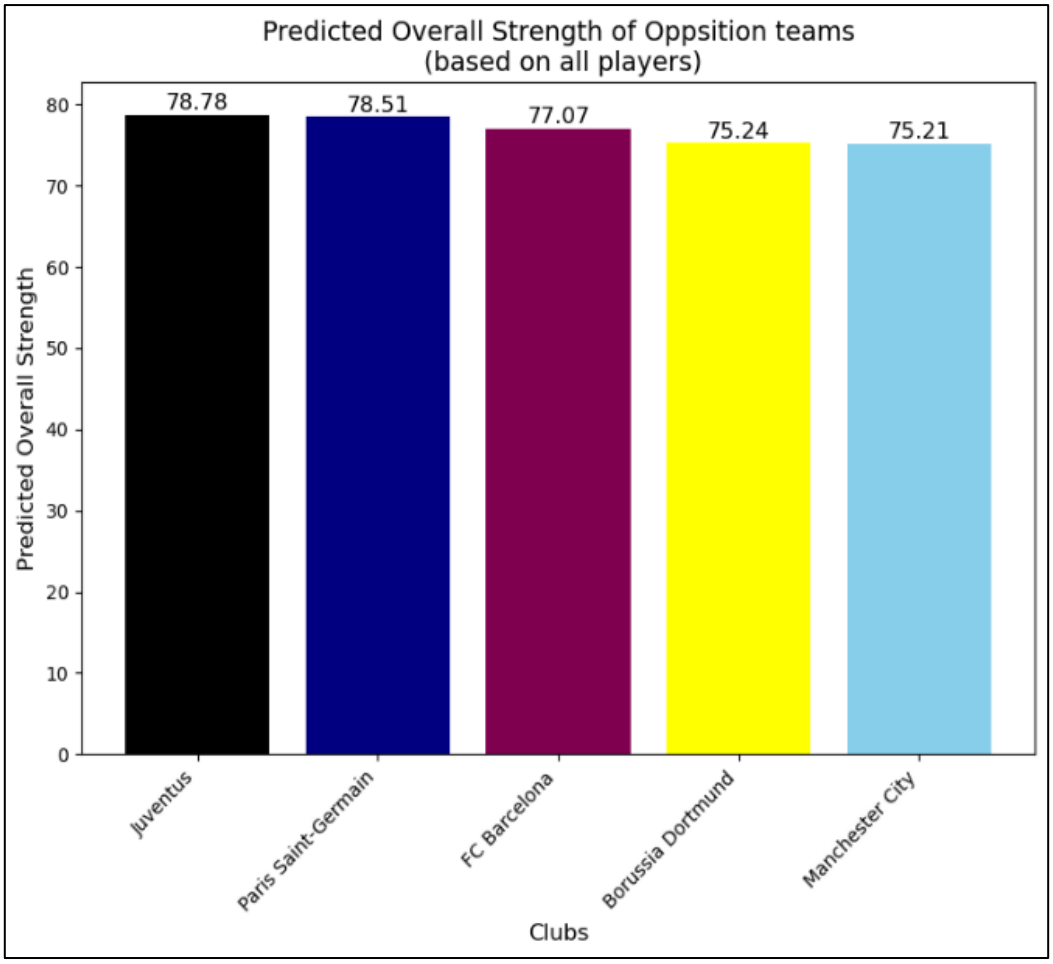
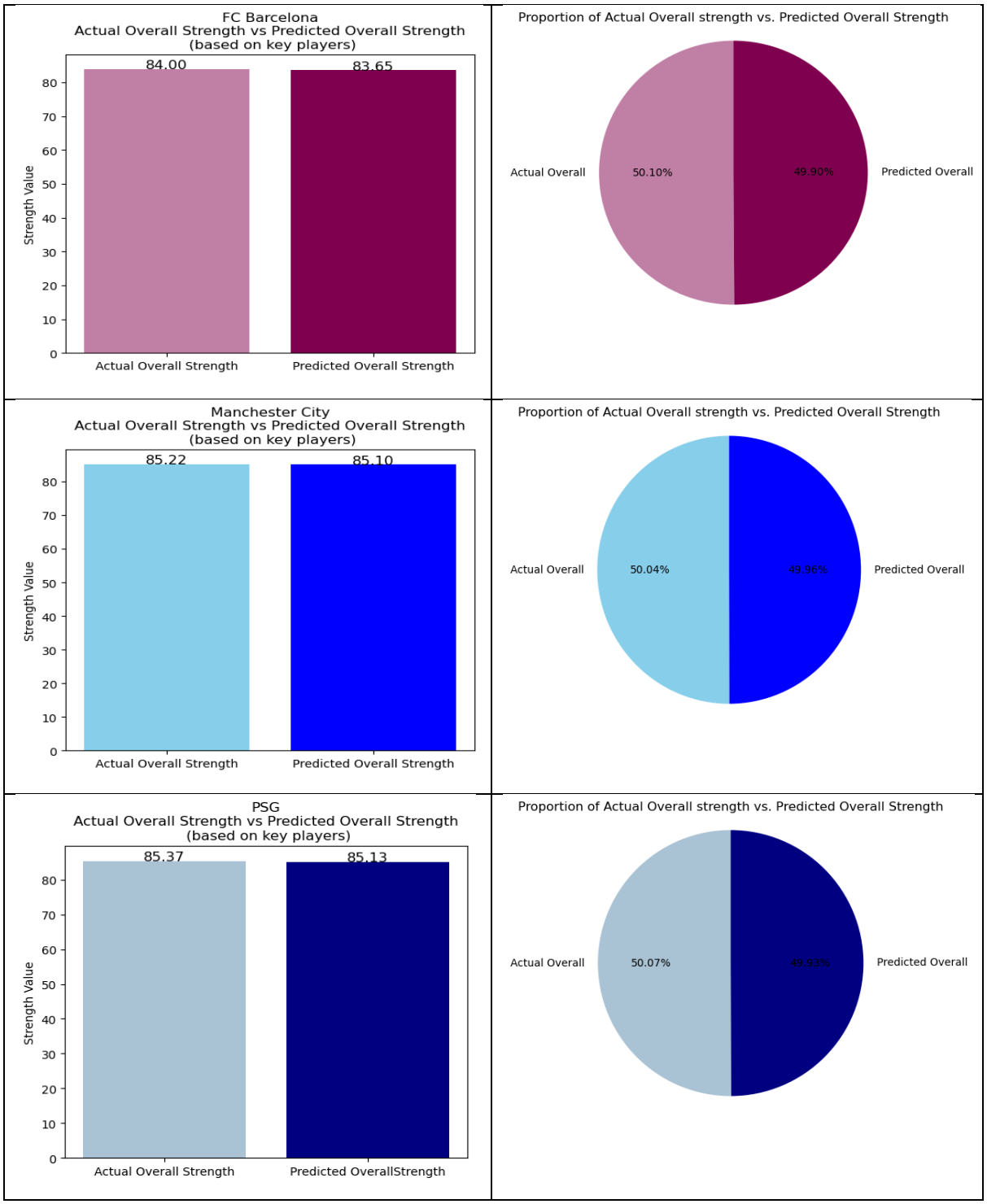


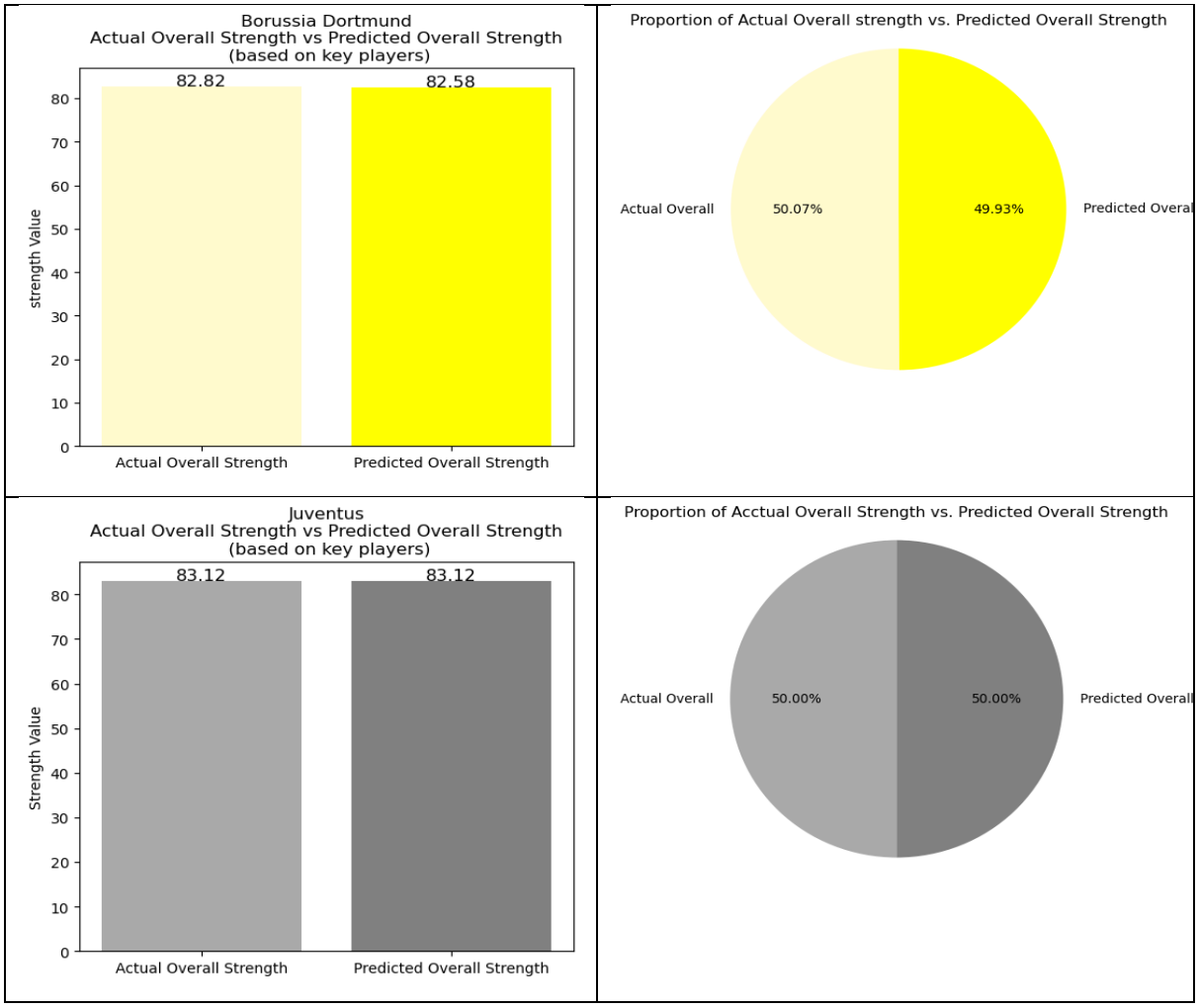
Figure 27. Predicted Overall strength of Oppositions (Based on All players)

The Actual Overall Strength and Predicted Overall Strength of each opposition team were calculated by using the same method used for Real Madrid CF. However, as mentioned in *Section 3.4.1.2*, using **all players** to determine team strength proved biased due to the inclusion of former/youth players with minimal contribution. Therefore, a refined approach focusing on **key players** with substantial influence was adopted for a more accurate representation of opposition strength.

3.4.2.2 Opposition Teams Actual & Predicted strength **(Based on key players)**

Table 23. Oppositions Actual and Predicted Overall Strength (based on key players)





The Actual Overall Strength vs Predicted Overall Strength for all opposition teams based on **key players** are shown in the above visualisations. The bar and pie charts show that, for every team, the actual overall strength and the predicted overall strength are closely aligned, with small variations depending on the team. Since the difference between actual overall and predicted overall strengths for each of the opposition teams are relatively low, these graphs collectively imply that the prediction model used has a high level of accuracy. Hence, we can determine that the model can accurately estimate team strength based on the given data.

3.4.3 Players Actual vs Predicted Overall Strength

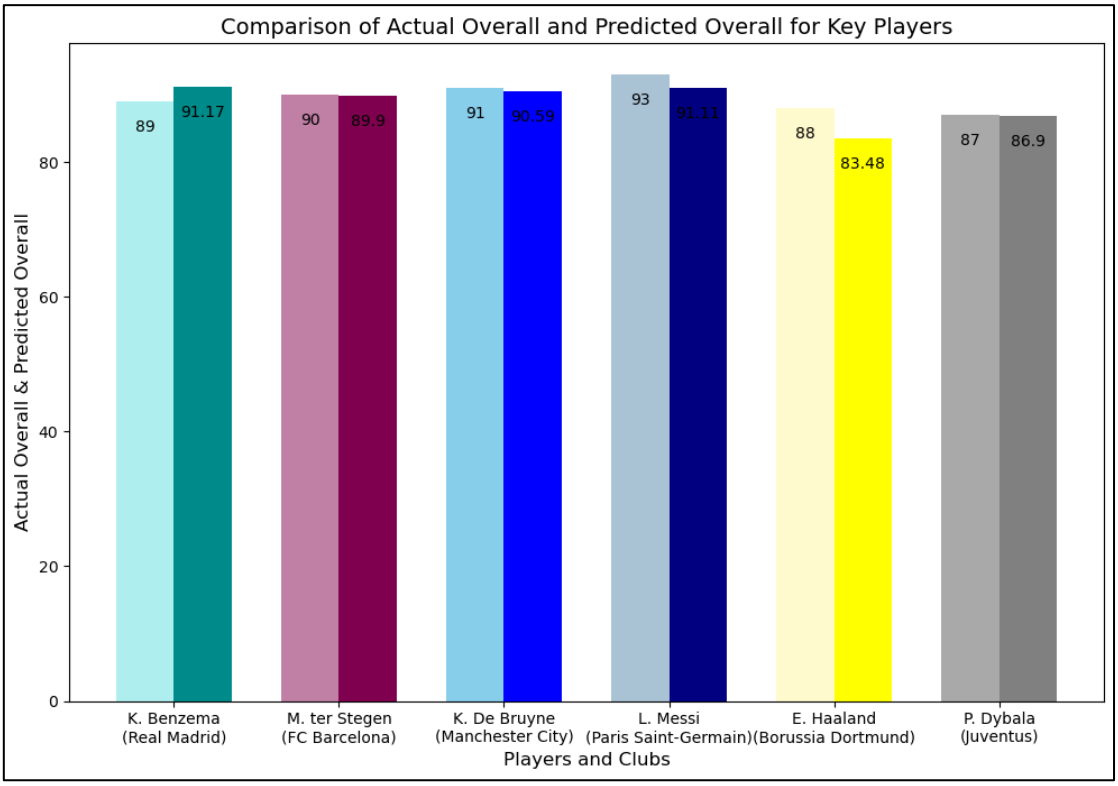


Figure 28. Players Actual and Predicted Overall Comparison

The graph compared the Actual and Predicted overall ratings of best players from the home and opposition teams. The chart highlighted variations in performance expectations and current form for these key players. We can see that **Benzema** had a room for improvement in the near future based on his predicted Overall. Rest of the opposition players from Top 5 European leagues were already performing well according to their actual overall ratings. However, their performance could slightly decrease based on their predicted overall ratings according to our model.

CHAPTER 4 – RESULTS & COMPARISON

4.1 PART A – Player Performance Evaluation

4.1.1 Evaluation Metrics Comparison

Table 24. Machine Learning Models Results

Models	RMSE	R^2 Score
Linear Regression	1.9224	0.9084
Random Forest	1.2120	0.9636
Decision Tree	1.7138	0.9272
Support Vector Machine	**1.0263**	**0.9739**
k-nearest neighbors	1.9262	0.9081

The evaluation metrics for 5 machine learning models based on RMSE (Root Mean Squared Error) and R2 Score are shown in the table. Lighter shade indicates low value whereas, darker shade indicates high values. With the lowest **RMSE (1.0263)** and highest **R-squared (0.9739)**, **Support Vector Machine (SVM)** proved to be the top performing model, suggesting improved predictive accuracy and model fit. With a decent R-square (0.9636) and a slightly higher RMSE (1.212), Random Forest is also a strong model after SVM. With an RMSE of 1.7138 and an R2 Score of 0.9272, the Decision Tree model performs decently, showing a balance between prediction accuracy and complexity. On the other hand, RMSEs of 1.92 and similar R-square of 0.90 indicate that Linear Regression and KNN perform relatively worse.

4.1.2 Actual & Predicted Values comparison

Top Performing Model: Support Vector Machine

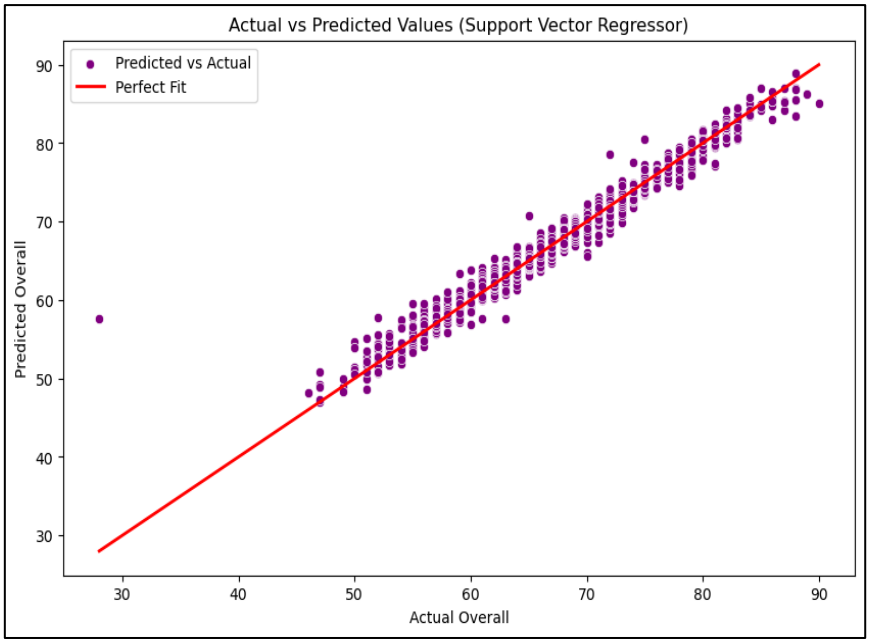
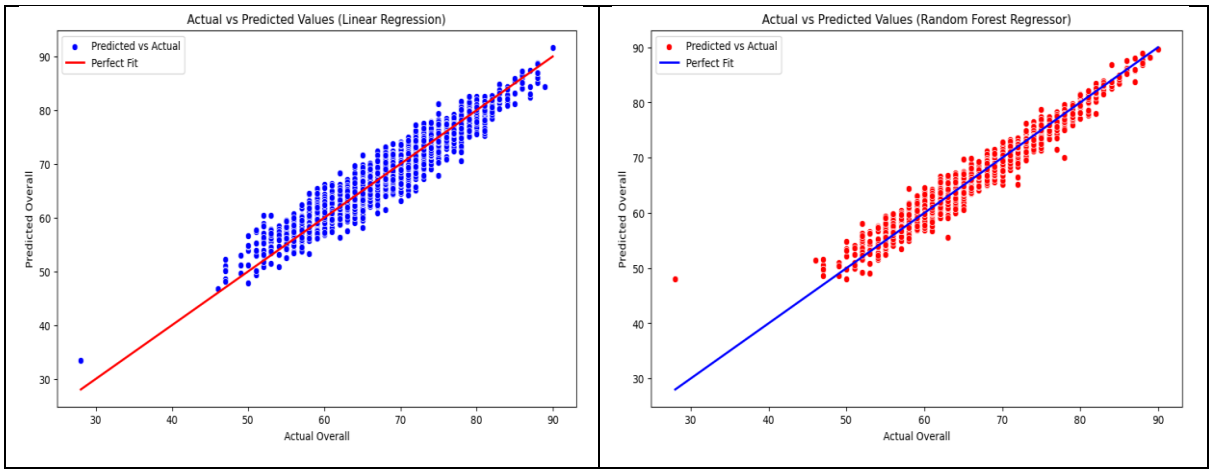


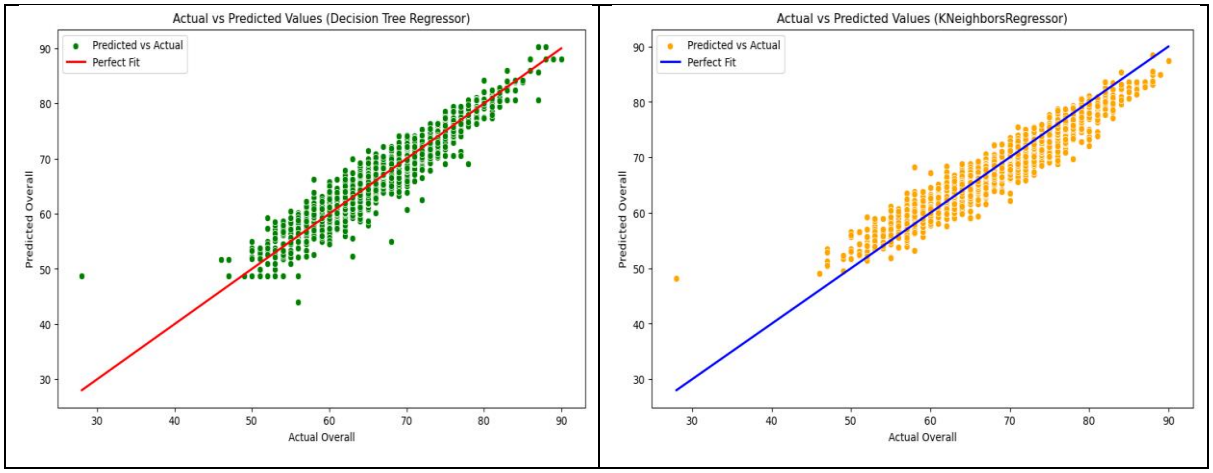
Figure 29. Actual vs Predicted Overall Graph of SVM Model

The scatterplot illustrates an excellent alignment between the predicted and actual overall values of players, indicating that Support Vector Machine (SVM) was the highest performing model. SVM was selected because of its strong ability to handle nonlinear relationships in the data and its robustness to generalize on unseen data. Since, the SVM model is able handle complex patterns that other linear models might miss, this model proved effective in predicting player performance resulting in more accurate prediction of the players' "Overall" ratings.

Other Models:

Table 25. Actual vs Predicted Overall Graph of Other Models





The scatterplots depict how actual and predicted overall for rest of the models are compared. Random Forest model exhibits the highest predictive accuracy, as evidenced by its closest alignment to the perfect fit line. Although they are more dispersed, Linear Regression and Decision Trees also follow with respectable accuracy. The largest spread is displayed by k-Nearest Neighbours, which suggests a lower prediction accuracy.

4.1.3 Feature Importance

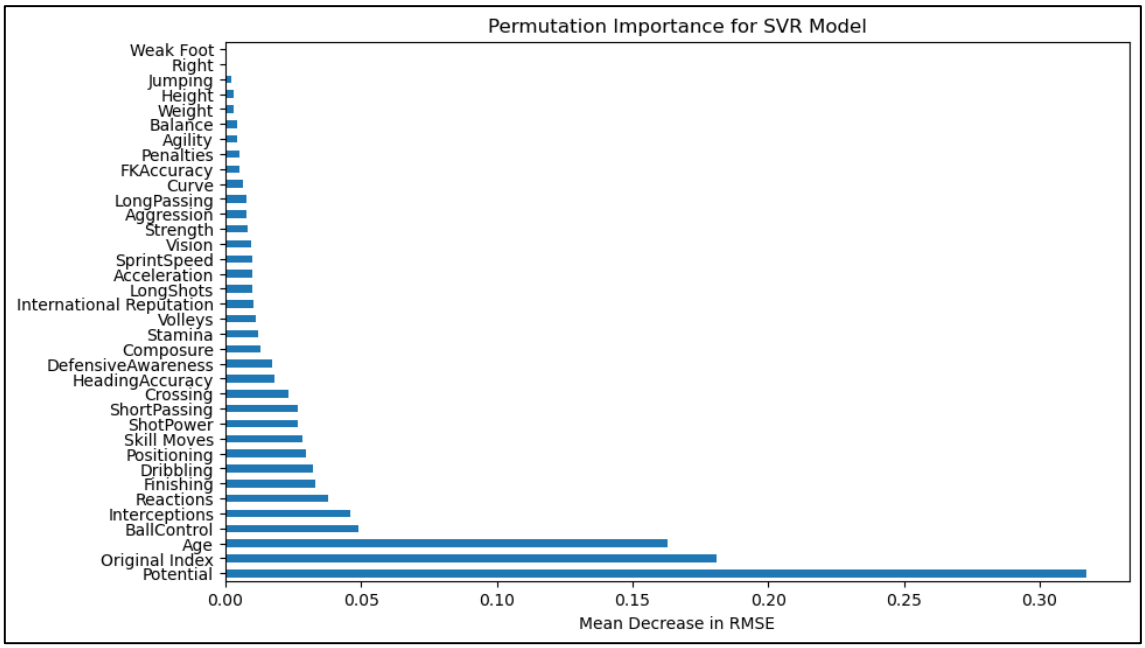


Figure 30. Feature Importance graph for SVR Model

The feature Importance graph shows the most important features in predicting a player's performance using the SVR model. In this case, Potential and Age have the highest impact on the prediction, followed by, Ball Control, Interceptions and reactions. Features like Weak Foot, Right, and Jumping have minimal influence on the model's prediction.

4.2 PART B – Team Strength & Opposition Quality

4.2.1 Comparison of Home Team vs Opposition Strength **(based on all players)**

Real Madrid CF's actual & predicted overall strength was compared against opposition teams from Europe's top 5 leagues. The average actual overall strength and predicted overall strength of **all players** in the team was used to compute the difference in predicted strength in comparison to Real Madrid.*(Refer to Appendix C.2)*

Table 26. Actual vs Predicted Overall Strength of Teams (Based on All players)

Club	Actual Overall Strength	Predicted Overall Strength
Juventus	78.73	78.79
Paris Saint-Germain	78.59	78.51
FC Barcelona	77.03	77.07
Real Madrid CF	76.05	76.29
Borussia Dortmund	75.34	75.24
Manchester City	75.08	75.21

Table 27. Calculated Strength Difference (Based on key players)

Clubs	Strength Difference
FC Barcelona	0.782
Manchester City	-1.074
Paris Saint-Germain	2.223
Borussia Dortmund	-1.046
Juventus	2.498

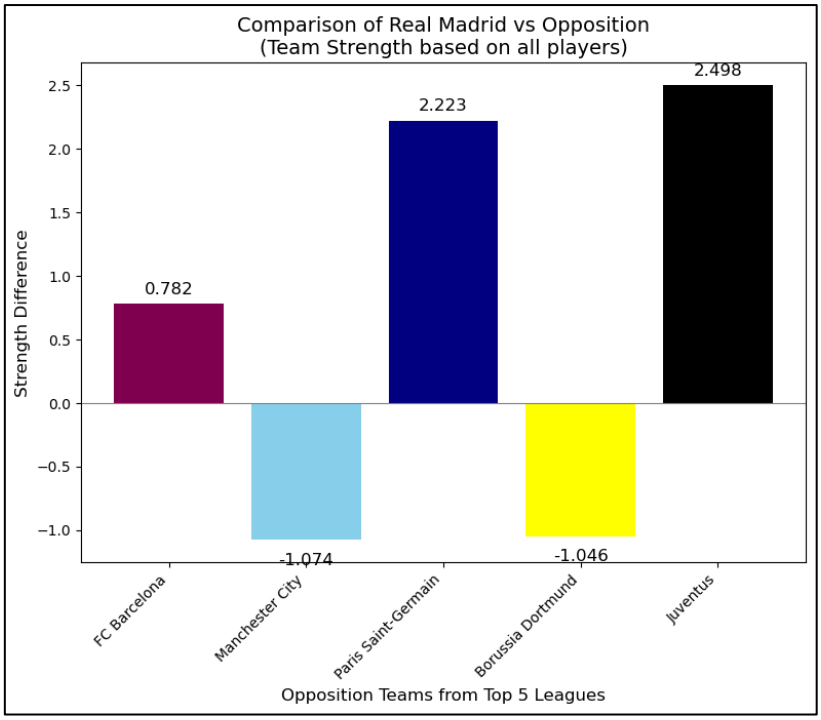


Figure 31. Strength Difference of Opposition with Real Madrid CF (Based on all players)

Based on **all players** in each team, the graph compares Real Madrid's team strength with opposition. FC Barcelona, Paris Saint-Germain and Juventus have a strength difference of 0.782, 2.223 and 2.498(respectively) with Real Madrid. The teams with higher predicted strength differences are considered to be stronger opposition when we compare the strength difference based on **all players** in each team. Real Madrid is predicted to be stronger than Manchester City and Borussia Dortmund as there is a negative strength difference of -1.074 and -1.046 respectively. However, this comparison of strengths could be proved biased as mentioned in section *3.4.1.2.* Hence, we need to select the key players for each team to ensure that we get accurate team strengths based on key players in the teams.

4.2.2 Comparison of Home Team vs Opposition Strength **(based on key players)**

The table below shows actual and predicted overall strength of the home and opposition teams based on **key players**. We can see that the actual overall & predicted overall strength of all the teams was slightly increased as compared to the overall & predicted strengths based on **all players**. The reason behind this is because we have included **key players** who had an overall of **>=80** with most appearances and contributions for their clubs.

*(Refer to appendix C for list of **key players** chosen for each club)*

Table 28. Actual vs Predicted Overall Strength of Teams (Based on Key players)

Club	Actual Overall Strength	Predicted Overall Strength
Real Madrid CF	84.06	84.07
FC Barcelona	84.00	83.65
Manchester City	85.22	85.10
Paris Saint-Germain	85.37	85.13
Borussia Dortmund	82.82	82.58
Juventus	83.12	83.12

Table 29. Strength Difference (Based on Key players)

Club	Predicted Strength Difference
FC Barcelona	-0.42
Manchester City	1.03
Paris Saint-Germain	1.06
Borussia Dortmund	-1.49
Juventus	-0.95

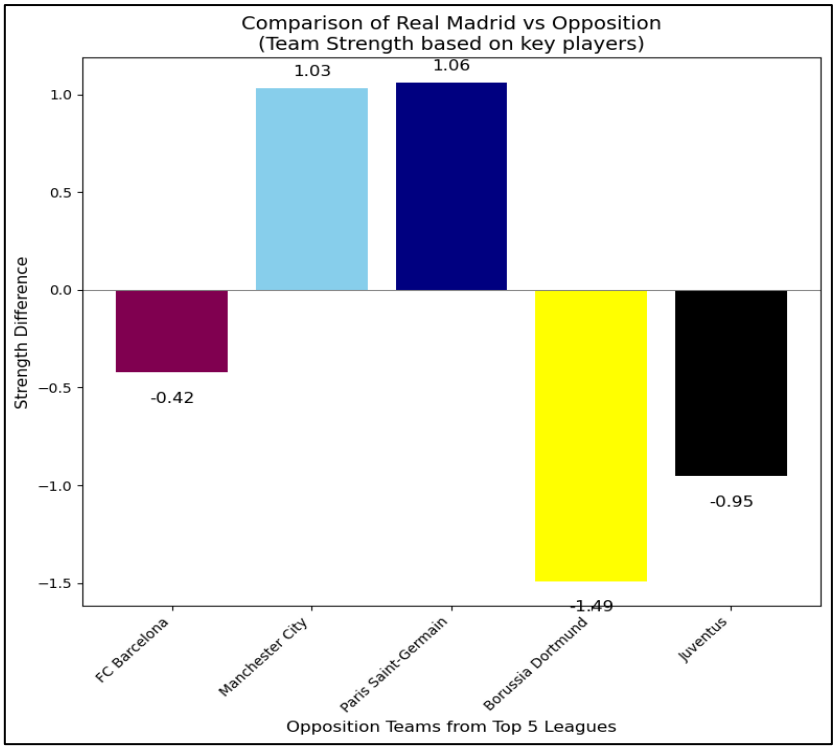


Figure 32. Strength Difference of Opposition with Real Madrid CF (Based on key players)

The predicted strength differences of Real Madrid CF against the opposition teams from top 5 European leagues are shown in the bar plot. Based on the strength difference, Paris Saint-Germain and Manchester City are predicted to be stronger than Real Madrid, with the highest positive predicted strength differences (+1.06 and +1.03, respectively). The negative strength differences of FC Barcelona (-0.42), Borussia Dortmund (-1.49), and Juventus (-0.95) suggest that they are predicted to be weaker than Real Madrid when we compare the strength difference based on **key players**.

Hence, we can infer that Real Madrid's key players play a crucial role in impacting the team's performance based on opposition strength/quality. This comparison helped us determine how Real Madrid might perform against top tier teams from top 5 European leagues when playing with their **key players.**

4.2.3 Player level Comparison

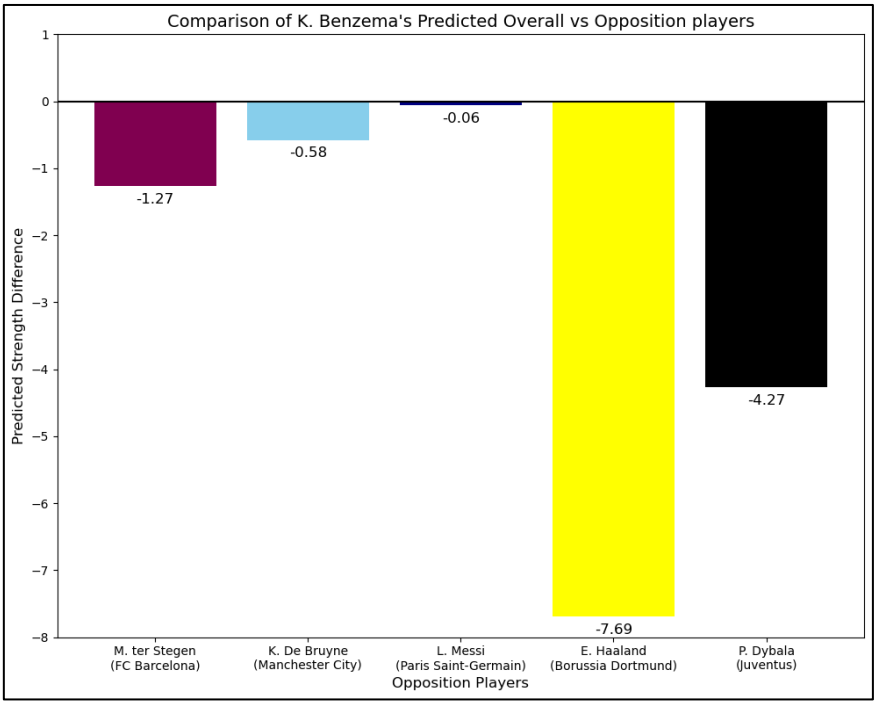


Figure 33. K. Benzema's Strength difference vs Opposition players

Our Home team player, Benzema consistently has a stronger predicted overall than all these key opposition players, except for Messi, where the difference is negligible. The largest gap appears with E. Haaland from Dortmund, where Benzema outperforms him significantly. This comparison highlights Benzema's dominance in predicted strength compared to the opposition team players. Since Manchester City & PSG are predicted to be stronger than Real Madrid the strength difference of players from these teams (K. Debruyne & L. Messi) is also less when compared to Benzema, indicating that Integrating Team strength and opposition quality plays a vital role in impacting individual player performance. It also demonstrates how effectively the machine learning models capture the true influence of a player's contributions under varied opposition strengths.

CHAPTER 5 – CONCLUSION & RECOMMNEDATIONS

5.1 Conclusion

The research effectively illustrates how contextual elements, like team strength and opposition quality, can be calculated to provide a more accurate and nuanced evaluation of football players & teams performance. The supervised machine learning models offered better predictive accuracy and allowed easier assessment of team strength and opposition quality. The research was able to predict player performance with high accuracy by using the top performing machine learning model, Support Vector Machine (SVM), as evidenced by its lowest RMSE and highest R-square among other models. The model's ability to offer reliable insights into team and opposition strength was further validated by comparing the actual and predicted team strengths for key players, thereby validating the project's objectives. These results imply that there is a great deal of room for improvement in the decision-making processes involved in scouting, team selection, and strategy formulation when machine learning is incorporated into player and team performance evaluations.

5.2 Limitations

The research had a number of limitations. The analysis was not able to be expanded any further due to time, resource, and word constraints. We could have included a wider range of analysis, such as match to match comparison or player to player comparison, which would have further influenced individual player performance, team strength & opposition quality. Furthermore, the dataset's scope was limited by the reliance on publicly available secondary data, which limited the model's applicability to different leagues or seasons. Additionally, there was less time to test more supervised learning models and adjust hyperparameters because of the time constraints. By adding more intricate contextual components and examining longitudinal data for deeper insights into player performance trends, future research could build on this foundation. Team strength and opposition quality can be integrated in machine learning models as additional factors to predict players performance based on it.

5.3 Recommendations

Based on the conclusion & findings, Coaches, Analysts and football teams would be advised to incorporate advanced analytics into player performance evaluation and decision-making procedures, with a particular emphasis on contextual factors such as team strength and opponent quality. Teams can improve their scouting, player acquisitions, and match strategies by implementing data-driven strategies. To stay competitive while cutting expenses, internal analytics skills and ongoing model updates should be established. Predictive models can also help clubs make better financial decisions about player contracts and transfers, allowing them to better manage risk. Evaluating player performance for female football players is also equally important in order to ensure equality and diversity in the sport.

Furthermore, comprehensive models that simultaneously integrate many contextual elements are lacking. Few research has attempted to develop a comprehensive model that considers a wide range of variables, even though several have examined the effects of certain aspects like match location or game status. Furthermore, additional longitudinal research that looks at player performance over longer time frames is required.

REFERENCES

Reference list

Aman Kharwal (2023a). *Data Visualization Guide using Python | Aman Kharwal*. [online] thecleverprogrammer. Available at: https://thecleverprogrammer.com/2023/11/01/data-visualization-guide-using-python/ [Accessed 13 Sep. 2024].

Aporia (2023). *Root Mean Square Error (RMSE): The cornerstone for evaluating regression models*. [online] Aporia. Available at: https://www.aporia.com/learn/root-mean-square-error-rmse-the-cornerstone-for-evaluating-regression-models/.

Baumer, B.S., Matthews, G.J. and Nguyen, Q. (2023). *Big Ideas in Sports Analytics and Statistical Tools for their Investigation*. [online] arXiv.org. doi:https://doi.org/10.48550/arXiv.2301.04001.

Bilek, G. and Ulas, E. (2019). Predicting match outcome according to the quality of opponent in the English premier league using situational variables and team performance indicators. *International Journal of Performance Analysis in Sport*, 19(6), pp.930–941. doi:https://doi.org/10.1080/24748668.2019.1684773.

Brito de Souza, D., López-Del Campo, R., Blanco-Pita, H., Resta, R. and Del Coso, J. (2019). An Extensive Comparative Analysis of Successful and Unsuccessful Football Teams in LaLiga. *Frontiers in Psychology*, 10. doi:https://doi.org/10.3389/fpsyg.2019.02566.

BryanB (2022). *FIFA23 OFFICIAL DATASET*. [online] Kaggle.com. Available at: https://www.kaggle.com/datasets/bryanb/fifa-player-stats-database/data [Accessed 12 Sep. 2024].

Dutta, A. (2019). *Random Forest Regression in Python - GeeksforGeeks*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/random-forest-regression-in-python/.

Fbref (2024). *Football Statistics and History*. [online] FBref.com. Available at: https://fbref.com/en/.

GeeksforGeeks (2019). *Working with Missing Data in Pandas*. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/working-with-missing-data-in-pandas/.

Gianluca Morciano, Zingoni, A. and Calabrò, G. (2024). Optimization and comparison of machine learning algorithms for the prediction of the performance of football players. *Neural Computing and Applications*. [online] doi:https://doi.org/10.1007/s00521-024-10260-9.

Gonçalves, B., Coutinho, D., Exel, J., Travassos, B., Lago, C. and Sampaio, J. (2019). Extracting spatial-temporal features that describe a team match demands when considering the effects of the quality of opposition in elite football. *PLOS ONE*, 14(8), p.e0221368. doi:https://doi.org/10.1371/journal.pone.0221368.

Herold, M., Kempe, M., Bauer, P. and Meyer, T. (2021). Attacking Key Performance Indicators in Soccer: Current Practice and Perceptions from the Elite to Youth Academy Level. *Journal of Sports Science and Medicine*, [online] 20(1), pp.158–169. doi:https://doi.org/10.52082/jssm.2021.158.

https://plus.google.com/u/0/+Datacamp (2011). *Learn R, Python & Data Science Online*. [online] Datacamp.com. Available at: https://www.datacamp.com/.

IBM (2023a). *About Linear Regression | IBM*. [online] www.ibm.com. Available at: https://www.ibm.com/topics/linear-regression.

IBM (2023b). *What is support vector machine? | IBM*. [online] www.ibm.com. Available at: https://www.ibm.com/topics/support-vector-machine.

IBM (2024). *What is the k-nearest neighbors algorithm? | IBM*. [online] www.ibm.com. Available at: https://www.ibm.com/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20(KNN.

Ibrahimi, E., Lopes, M.B., Xhilda Dhamo, Simeon, A., Rajesh Shigdel, Hron, K., Blaž Stres, D'Elia, D., Berland, M. and Laura Judith Marcos-Zambrano (2023). Overview of data preprocessing for machine learning applications in human microbiome research. *Frontiers in Microbiology*, 14. doi:https://doi.org/10.3389/fmicb.2023.1250909.

Joy, B. and Weil, E. (2019). Football | History, Rules, & Significant Players. In: *Britannica*. [online] Available at: https://www.britannica.com/sports/football-soccer.

Kharwal, A. (2023b). *Feature Selection using Python | Aman Kharwal*. [online] thecleverprogrammer. Available at: https://thecleverprogrammer.com/2023/09/27/feature-selection-using-python/.

Levi, H.R. and Jackson, R.C. (2018). Contextual factors influencing decision making: Perceptions of professional soccer players. *Psychology of Sport and Exercise*, [online] 37, pp.19–25. doi:https://doi.org/10.1016/j.psychsport.2018.04.001.

Liu, H., Gómez, M.-A., Gonçalves, B. and Sampaio, J. (2015). Technical performance and match-to-match variation in elite football teams. *Journal of Sports Sciences*, 34(6), pp.509–518.

Manish., S., Bhagat, V. and Pramila, R. (2021). *Prediction of Football Players Performance using Machine Learning and Deep Learning Algorithms*. [online] IEEE Xplore. doi:https://doi.org/10.1109/INCET51464.2021.9456424.

Merzah, B.M., Croock, M.S. and Rashid, A.N. (2024). Intelligent Classifiers for Football Player Performance Based on Machine Learning Models. *International journal of electrical and computer engineering systems*, 15(2), pp.173–183. doi:https://doi.org/10.32985/ijeces.15.2.6.

Mountifield, C. (2023). *Sport Analytics: Graduating from Alchemy*. [online] www.intechopen.com. Available at: https://www.intechopen.com/chapters/1162484# [Accessed 26 Jul. 2024].

Nikolaos Giannakoulas, Papageorgiou, G.K. and Christos Tjortjis (2023). Forecasting Goal Performance for Top League Football Players: A Comparative Study. *IFIP advances in information and communication technology*, pp.304–315. doi:https://doi.org/10.1007/978-3-031-34107-6_24.

Ninja, N. (2023). *Addressing Absence: Handling Missing Values*. [online] Let's Data Science. Available at: https://letsdatascience.com/handling-missing-values/.

Paolo Cintia, S. Rinzivillo and Pappalardo, L. (2015). *A network-based approach to evaluate the performance of football teams*. [online] Available at: https://www.semanticscholar.org/paper/A-network-based-approach-to-evaluate-the-of-teams-Cintia-Rinzivillo/14b999687a1a27eec238ae6eafdd93496132e993 [Accessed 28 Sep. 2024].

Pappalardo, L., Cintia, P., Ferragina, P., Massucco, E., Pedreschi, D. and Giannotti, F. (2019). PlayeRank: Data-driven Performance Evaluation and Player Ranking in Soccer via a Machine Learning Approach. *ACM Transactions on Intelligent Systems and Technology*,

10(5), pp.1–27. doi:https://doi.org/10.1145/3343172.

Qader, M.A., Zaidan, B.B., Zaidan, A.A., Ali, S.K., Kamaluddin, M.A. and Radzi, W.B. (2017). A methodology for football players selection problem based on multi-measurements criteria analysis. *Measurement*, 111, pp.38–50. doi:https://doi.org/10.1016/j.measurement.2017.07.024.

R. M, R. B, L. R and S. K (). Player Performance Prediction in Sports Using Machine Learning. In: *2024 International Conference on Intelligent Systems for Cybersecurity (ISCS)*. pp.1–6. doi:https://doi.org/10.1109/ISCS61804.2024.10581086.

Robberechts, P., Haaren, V. and Davis, J. (2019). *Who Will Win It? An Ingame Win Probability Model for Football*.

Rosch, D., Hodgson, R., Peterson, L., GrafBaumann, T., Junge, A., Chomiak, J. and Dvorak, J. (2000). Assessment and Evaluation of Football Performance. *Am J Sports Med*, [online] 28(5_suppl), pp.29–39. doi:https://doi.org/10.1177/28.suppl_5.s29.

Santín, D. (2014). Measuring the technical efficiency of football legends: who were Real Madrid's all-time most efficient players? *International Transactions in Operational Research*, 21(3), pp.439–452. doi:https://doi.org/10.1111/itor.12082.

Simplilearn (2023). *Data Preprocessing in Machine Learning: A Beginner's Guide*. [online] Simplilearn.com. Available at: https://www.simplilearn.com/data-preprocessing-in-machine-learning-article?utm_source=frs_article_page&utm_medium=top_share_option&utm_campaign=frs_copy_share_icon [Accessed 12 Sep. 2024].

SoFIFA (n.d.). *Players FIFA 20 May 6, 2020 SoFIFA*. [online] sofifa.com. Available at: https://sofifa.com/.

Tatachar, A.V., 2021. Comparative assessment of regression models based on model evaluation metrics. *International Journal of Innovative Technology and Exploring Engineering*, *8*(9), pp.853-860.

Viswa (2023). *Unveiling Decision Tree Regression: Exploring its Principles, Implementation*. [online] Medium. Available at: https://medium.com/@vk.viswa/unveiling-decision-tree-regression-exploring-its-principles-implementation-beb882d756c6.

Zach (2021). *RMSE vs. R-Squared: Which Metric Should You Use?* [online] Statology. Available at: https://www.statology.org/rmse-vs-r-squared/.

APPENDIX

Appendix A

1. Train Test Split

TRAIN TEST SPLIT

```
from sklearn.model_selection import train_test_split

train_set, test_set = train_test_split(df, test_size=0.2, random_state=7)
print(f"{train_set.shape[0]} train and {test_set.shape[0]} test instances")
```

```
13368 train and 3342 test instances
```

2. Feature Scaling

Trainset Head after feature scaling:

	Age	Overall	Potential	Preferred Foot	International Reputation	Weak Foot	Skill Moves	Height	Weight	Crossing	...
7655	25	77	81	Right	1.0	3.0	2.0	189	75	52.0	...
4565	31	76	76	Right	1.0	3.0	3.0	180	80	45.0	...
722	29	73	73	Left	1.0	2.0	3.0	171	69	80.0	...
5904	29	67	67	Right	1.0	3.0	2.0	178	72	62.0	...
11894	28	63	65	Right	1.0	4.0	2.0	185	70	47.0	...

5 rows × 38 columns

LongShots	Aggression	Interceptions	Positioning	Vision	Penalties	Composure	DefensiveAwareness	Original Index	Right
27.0	75.0	80.0	40.0	57.0	38.0	72.0	78.0	7655	1.0
66.0	56.0	44.0	84.0	71.0	73.0	81.0	30.0	4565	1.0
74.0	79.0	71.0	63.0	77.0	67.0	69.0	68.0	722	0.0
67.0	62.0	54.0	67.0	71.0	65.0	62.0	44.0	5904	1.0
33.0	61.0	61.0	40.0	49.0	32.0	67.0	62.0	11894	1.0

Testset Head after feature scaling:

	Age	Overall	Potential	Preferred Foot	International Reputation	Weak Foot	Skill Moves	Height	Weight	Crossing	...
609	30	76	76	Right	1.0	5.0	3.0	175	75	78.0	...
10057	21	63	75	Right	1.0	3.0	3.0	172	65	59.0	...
7098	21	70	70	Right	1.0	3.0	2.0	179	73	66.0	...
16375	24	65	69	Right	1.0	3.0	1.0	189	76	12.0	...
14372	17	52	75	Right	1.0	3.0	3.0	168	63	44.0	...

5 rows × 38 columns

LongShots	Aggression	Interceptions	Positioning	Vision	Penalties	Composure	DefensiveAwareness	Original Index	Right
53.0	90.0	73.0	71.0	70.0	51.0	72.0	75.0	609	1.0
61.0	54.0	29.0	60.0	60.0	51.0	64.0	39.0	10057	1.0
61.0	56.0	22.0	57.0	69.0	49.0	65.0	37.0	7098	1.0
5.0	28.0	8.0	8.0	50.0	13.0	34.0	14.0	16375	1.0
44.0	32.0	31.0	43.0	47.0	43.0	56.0	22.0	14372	1.0

3. Machine Learning Models

BASELINE MODEL (Linear Regression)

In [86]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import time
import math

# Initialize the Linear Regression model
lr_model = LinearRegression()

# Measure the time taken for training
start = time.time()
lr_model.fit(X_train, y_train)  # Fit the model with training data
end = time.time() - start
print(f"Took {end} seconds to train Linear Regression model")

# Make predictions on the test set
y_pred_lr = lr_model.predict(X_test)

# Make predictions on the training set
y_train_pred_lr = lr_model.predict(X_train)

# Calculate RMSE for the training set
train_mse_lr = mean_squared_error(y_train, y_train_pred_lr)
train_rmse_lr = math.sqrt(train_mse_lr)

# Calculate RMSE for the test set
test_mse_lr = mean_squared_error(y_test, y_pred_lr)
test_rmse_lr = math.sqrt(test_mse_lr)

# Calculate evaluation metrics for the test set
mae_lr = mean_absolute_error(y_test, y_pred_lr)
mse_lr = mean_squared_error(y_test, y_pred_lr)
rmse_lr = math.sqrt(mse_lr)
r2_lr = r2_score(y_test, y_pred_lr)

# Print the results
print("Linear Regression Model Evaluation:")
print(f"Training RMSE: {train_rmse_lr}")
print(f"Test RMSE: {test_rmse_lr}")
print(f"RMSE Difference (Test - Train): {test_rmse_lr - train_rmse_lr}")
print(f"Root Mean Squared Error (RMSE): {rmse_lr}")
print(f"R² Score: {r2_lr}")
```

```
Took 0.021081209182739258 seconds to train Linear Regression model
Linear Regression Model Evaluation:
Training RMSE: 1.9128881787384153
Test RMSE: 1.9223966033245667
RMSE Difference (Test - Train): 0.00950842458615142
Root Mean Squared Error (RMSE): 1.9223966033245667
R² Score: 0.9084315582022222
```

Random Forest

In [88]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, make_scorer
import time
import math

# Custom scoring function for RMSE
def rmse_scorer(y_true, y_pred):
    return math.sqrt(mean_squared_error(y_true, y_pred))

# Create the RMSE scorer for GridSearchCV
scorer = make_scorer(rmse_scorer, greater_is_better=False)

# Initialize the RandomForestRegressor
rf = RandomForestRegressor(random_state=7)

# Provide the values of the hyperparameters. There will be 3x3x3=36 combinations
param_grid = {
    'n_estimators': [10, 30, 50],
    'max_depth': [5, 10, 15],
    'min_samples_split': [10, 25, 30]
}

# 5-fold cross-validation has been used with RMSE scoring
rf_grid_search = GridSearchCV(rf, param_grid, cv=3,
                              scoring=scorer,  # Use custom RMSE scorer
                              return_train_score=True, verbose=2, n_jobs=-1) # Use all available CPU cores

start = time.time()
rf_grid_search.fit(X_train, y_train)  # Ensure X_train and y_train are defined
end = time.time() - start
print(f"Took {end} seconds")

# Print best parameters and best score (negative RMSE)
print("Best parameters found: ", rf_grid_search.best_params_)
print("Best score (negative RMSE): ", rf_grid_search.best_score_)

# Get the best estimator
best_rf = rf_grid_search.best_estimator_

# Predictions and evaluation on the test set
y_pred = best_rf.predict(X_test)

# Calculate the RMSE on the test set
rmse = math.sqrt(mean_squared_error(y_test, y_pred))

print(f"Test RMSE: {rmse}")
```

```
Fitting 3 folds for each of 27 candidates, totalling 81 fits
Took 44.20388126373291 seconds
Best parameters found:  {'max_depth': 15, 'min_samples_split': 10, 'n_estimators': 50}
Best score (negative RMSE):  -1.2318366368011233
Test RMSE: 1.2120171892075011
```

In [90]:
```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Using the best hyperparameters from the GridSearchCV to predict the test set
best_rf = rf_grid_search.best_estimator_  # Use the best regressor found by GridSearchCV
y_pred = best_rf.predict(X_test)

# Measuring regression metrics: MSE, MAE, and R²
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
rmse = math.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("Random Forest Regressor:")
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"Mean Absolute Error (MAE): {mae}")
print(f"R² Score: {r2}")
```

```
Random Forest Regressor:
Mean Squared Error (MSE): 1.4689856669344514
Root Mean Squared Error (RMSE): 1.2120171892075011
Mean Absolute Error (MAE): 0.7641478864265232
R² Score: 0.9636020099943994
```

DECISION TREE

In [92]:
```
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, make_scorer
import time
import math

# Custom scoring function for RMSE (Root Mean Squared Error)
def rmse_scorer(y_true, y_pred):
    return math.sqrt(mean_squared_error(y_true, y_pred))

# Create the RMSE scorer
scorer = make_scorer(rmse_scorer, greater_is_better=False)

# Initialize the DecisionTreeRegressor
dtr = DecisionTreeRegressor(random_state=7)

# Define the parameter grid for hyperparameter tuning
param_grid = {
    'max_depth': [5, 10, 20, 25],
    'min_samples_split': [10, 15, 20, 30]
}

# 5-fold cross-validation has been used with RMSE scoring
dt_grid_search = GridSearchCV(dtr, param_grid, cv=5,
                              scoring=scorer,  # Use custom RMSE scorer
                              return_train_score=True, verbose=2,
                              n_jobs=-1) # Use all available CPU cores

# Measure the time taken for training
start = time.time()

# Fit the model with training data
dt_grid_search.fit(X_train, y_train)  # Ensure X_train and y_train are defined

end = time.time() - start
print(f"Took {end} seconds")

# Print best parameters and best score (negative RMSE)
print("Best parameters found: ", dt_grid_search.best_params_)
print("Best score (negative RMSE): ", dt_grid_search.best_score_)

# Get the best estimator
best_dtr = dt_grid_search.best_estimator_

# Predictions and evaluation on the test set
y_pred = best_dtr.predict(X_test)

# Calculate the RMSE on the test set
rmse = math.sqrt(mean_squared_error(y_test, y_pred))

print(f"Test RMSE: {rmse}")
```

```
Fitting 5 folds for each of 16 candidates, totalling 80 fits
Took 2.3564453125 seconds
Best parameters found:  {'max_depth': 20, 'min_samples_split': 30}
Best score (negative RMSE):  -1.7057624810101306
Test RMSE: 1.7137929818938564
```

In [94]:
```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Using the best hyperparameters from the GridSearchCV to predict the test set
best_dtr = dt_grid_search.best_estimator_  # Use the best regressor found by GridSearchCV
y_pred = best_dtr.predict(X_test)

# Measuring regression metrics: MSE, MAE, and R²
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
rmse = math.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("Decision Tree Regressor:")
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"Mean Absolute Error (MAE): {mae}")
print(f"R² Score: {r2}")
```

```
Decision Tree Regressor:
Mean Squared Error (MSE): 2.9370863847886364
Root Mean Squared Error (RMSE): 1.7137929818938564
Mean Absolute Error (MAE): 1.0894239471676104
R² Score: 0.9272259469336996
```

SUPPORT VECTOR MACHINE

In [96]:
```
from sklearn.svm import SVR
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, make_scorer
import time
import math

# Custom scoring function for RMSE (Root Mean Squared Error)
def rmse_scorer(y_true, y_pred):
    return math.sqrt(mean_squared_error(y_true, y_pred))

# Create the RMSE scorer for GridSearchCV
scorer = make_scorer(rmse_scorer, greater_is_better=False)

# Initialize the SVR model
svr = SVR(kernel='rbf')

# Specify the hyperparameters and their values
param_grid = {
    'C': [0.01, 0.1, 1, 10],  # Regularization parameter
    'gamma': ["auto", 0.1],  # Kernel coefficient
}

# 5-fold cross-validation for SVR with RMSE scoring
svr_grid_search = GridSearchCV(svr, param_grid, cv=5, scoring=scorer, return_train_score=True, verbose=2,
                               n_jobs=-1) # Use all available CPU cores

start = time.time()
svr_grid_search.fit(X_train, y_train)  # Ensure X_train and y_train are defined
end = time.time() - start
print(f"Took {end} seconds")

# Print best parameters and best score (negative RMSE)
print("Best parameters found: ", svr_grid_search.best_params_)
print("Best score (negative RMSE): ", svr_grid_search.best_score_)

# Get the best estimator
best_svr = svr_grid_search.best_estimator_

# Predictions and evaluation on the test set
y_pred = best_svr.predict(X_test)

# Calculate the RMSE on the test set
rmse = math.sqrt(mean_squared_error(y_test, y_pred))

print(f"Test RMSE: {rmse}")
```

```
Fitting 5 folds for each of 8 candidates, totalling 40 fits
Took 339.9077067375183 seconds
Best parameters found:  {'C': 10, 'gamma': 'auto'}
Best score (negative RMSE):  -0.9419248464177186
Test RMSE: 1.026257583582593
```

In [98]:
```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Using the best hyperparameters from the GridSearchCV to predict the test set
best_svr = svr_grid_search.best_estimator_  # Use the best regressor found by GridSearchCV
y_pred = best_svr.predict(X_test)

# Measuring regression metrics: MSE, MAE, and R²
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
rmse = math.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("Support Vector Regressor:")
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"Mean Absolute Error (MAE): {mae}")
print(f"R² Score: {r2}")
```

```
Support Vector Regressor:
Mean Squared Error (MSE): 1.0532046278607827
Root Mean Squared Error (RMSE): 1.026257583582593
Mean Absolute Error (MAE): 0.6338804277125898
R² Score: 0.97390408063087
```

k-nearest neighbors (KNN)

In [104]:
```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, make_scorer
import time
import math

# Custom scoring function for RMSE (Root Mean Squared Error)
def rmse_scorer(y_true, y_pred):
    return math.sqrt(mean_squared_error(y_true, y_pred))

# Create the RMSE scorer for GridSearchCV
scorer = make_scorer(rmse_scorer, greater_is_better=False)

# Initialize the KNN Regressor model
knn_regressor = KNeighborsRegressor()

# Specify the hyperparameters and their values for KNN Regressor
param_grid = {
    'n_neighbors': [8, 9, 10, 11, 12, 13, 14, 15, 16, 17],  # Number of neighbors to consider
}

# 5-fold cross-validation for KNN Regressor with RMSE scoring
knn_grid_search = GridSearchCV(knn_regressor, param_grid, cv=5, scoring=scorer, return_train_score=True,
                               verbose=1, n_jobs=-1)  # Use all available CPU cores

start = time.time()
knn_grid_search.fit(X_train, y_train)  # Ensure X_train and y_train are defined
end = time.time() - start
print(f"Took {end} seconds")

# Print best parameters and best score (negative RMSE)
print("Best parameters found: ", knn_grid_search.best_params_)
print("Best score (negative RMSE): ", knn_grid_search.best_score_)

# Get the best estimator
best_knn = knn_grid_search.best_estimator_

# Predictions and evaluation on the test set
y_pred = best_knn.predict(X_test)

# Calculate the RMSE on the test set
rmse = math.sqrt(mean_squared_error(y_test, y_pred))

print(f"Test RMSE: {rmse}")
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
Took 7.180791139602661 seconds
Best parameters found:  {'n_neighbors': 9}
Best score (negative RMSE):  -1.949574434397637
Test RMSE: 1.9261567222470286
```

In [106]:
```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Using the best hyperparameters from the GridSearchCV to predict the test set
best_knn = knn_grid_search.best_estimator_  # Use the best regressor found by GridSearchCV
y_pred = best_knn.predict(X_test)

# Measuring regression metrics: MSE, MAE, and R²
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
rmse = math.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("KNeighborsRegressor:")
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"Mean Absolute Error (MAE): {mae}")
print(f"R² Score: {r2}")
```

```
KNeighborsRegressor:
Mean Squared Error (MSE): 3.7100797186574166
Root Mean Squared Error (RMSE): 1.9261567222470286
Mean Absolute Error (MAE): 1.4608684088037769
R² Score: 0.9080730006021904
```

Appendix B

1. Predictions on Test and Train sets using Top performing Model (SVM)

Predicted Values of Test Set based on the top performing Model

In [100]:
```
# Using the best SVR model found by GridSearchCV to predict values
best_svr = svr_grid_search.best_estimator_  # This is the best SVR model after tuning

# Predict on the test set (X_test)
y_pred_svr = best_svr.predict(X_test)

# Print or return the predicted values
print("Test_set Predictive Overall:")
print(y_pred_svr)

predictions_df = pd.DataFrame({
    'Actual': y_test,  # Replace with your actual test set labels
    'Predicted': y_pred_svr
})

# Display the predictions DataFrame
print(predictions_df)
```

```
Test_set Predictive Overall:
[75.56572041 63.94735055 67.81191869 ... 59.51493797 76.58458899
 71.71606403]
      Actual  Predicted
0         76  75.565720
1         63  63.947351
2         70  67.811919
3         65  64.359304
4         52  54.431317
...      ...        ...
3337      72  71.755107
3338      82  80.724438
3339      59  59.514938
3340      76  76.584589
3341      72  71.716064

[3342 rows x 2 columns]
```

Predicted Values of Train Set based on the top performing Model

In [101]:
```
# Using the best SVR model found by GridSearchCV to predict values
best_svr = svr_grid_search.best_estimator_  # This is the best SVR model after tuning

# Predict on the train set (X_train)
y_pred_train_svr = best_svr.predict(X_train)

# Print or return the predicted values for the train set
print("Train_set Predictive Overall:")
print(y_pred_train_svr)

# Create a DataFrame to compare actual vs predicted for the train set
train_predictions_df = pd.DataFrame({
    'Actual': y_train,  # Actual labels for the train set
    'Predicted': y_pred_train_svr
})

# Display the train predictions DataFrame
print(train_predictions_df)
```

```
Train_set Predictive Overall:
[77.45402379 75.89991033 72.93579016 ... 79.10032799 72.67704617
 62.90017693]
       Actual  Predicted
0          77  77.454024
1          76  75.899910
2          73  72.935790
3          67  66.905439
4          63  63.375156
...       ...        ...
13363      72  72.100133
13364      60  60.100350
13365      79  79.100328
13366      73  72.677046
13367      63  62.900177

[13368 rows x 2 columns]
```

2. **Combining the Train & Test sets after predictions and sorting the predicted values by 'Original Index' to align with the original dataset order.**

Prediction on Both sets after combining Train and Test set to Predict Overall of all the players

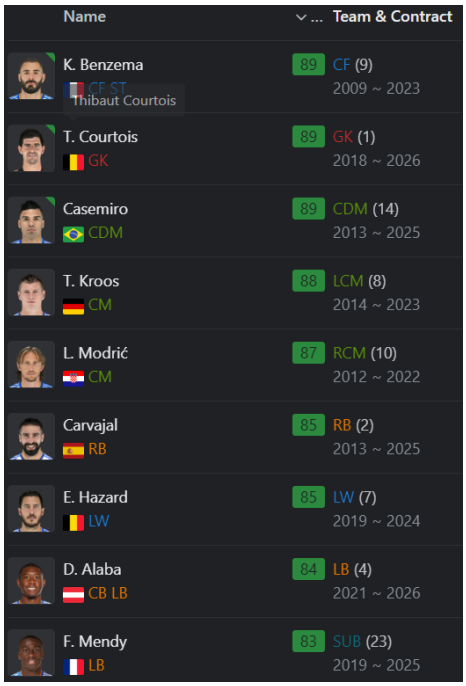
In [102]:
```
# Step 4: Use the best SVR model found by GridSearchCV to predict values on both train and test sets
best_svr = svr_grid_search.best_estimator_

# Predict on the train and test sets
y_pred_train = best_svr.predict(X_train)
y_pred_test = best_svr.predict(X_test)

# Combine predicted values for both train and test sets
X_combined = pd.concat([X_train, X_test], axis=0)
y_pred_combined = np.concatenate([y_pred_train, y_pred_test])

# Step 5: Restore the original indices and ensure correct order after prediction
combined_original_indices = pd.concat([train_set['Original Index'], test_set['Original Index']], axis=0)
predicted_df = pd.DataFrame({
    'Original Index': combined_original_indices,
    'Predicted Overall': y_pred_combined
})

# Step 6: Sort the predicted_df by 'Original Index' to align with the original dataset order
predicted_df_sorted = predicted_df.sort_values(by='Original Index').reset_index(drop=True)


# Load the original dataset that contains 'Name', 'Club', 'Position', etc.
file_path = r'C:\Users\Yash\Desktop\Research Project\FIFA 22 FINAL FILE(Cleaned data with OG Index).csv'
df_before_split = pd.read_csv(file_path)

# Step 7: Join the predicted values with the original dataset (sorted by 'Original Index')
final_df = df_before_split[['Name', 'Club', 'Position', 'Overall']].join(predicted_df_sorted['Predicted Overall'])

# Display the first few rows of the final DataFrame
print(final_df.head())
```

```
              Name                Club Position  Overall  Predicted Overall
0  Bruno Fernandes     Manchester United      CAM       88          87.666627
1      L. Goretzka   FC Bayern MÃ¼nchen       CM       87          87.099988
2       L. SuÃ¡rez  AtlÃ©tico de Madrid       CF       88          88.922819
3     K. De Bruyne       Manchester City       CM       91          90.588701
4        M. AcuÃ±a           Sevilla FC       LB       84          83.937940
```

Appendix C

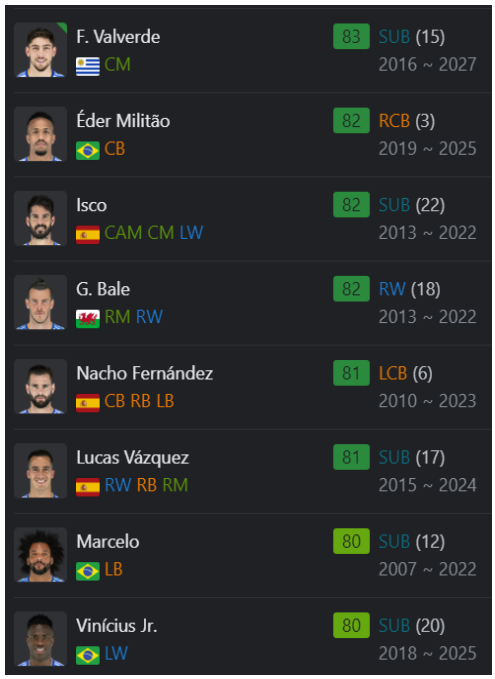
1. **List of Key players for each club compared from SoFifa.com (FIFA 22, September 2021)**

Real Madrid CF:



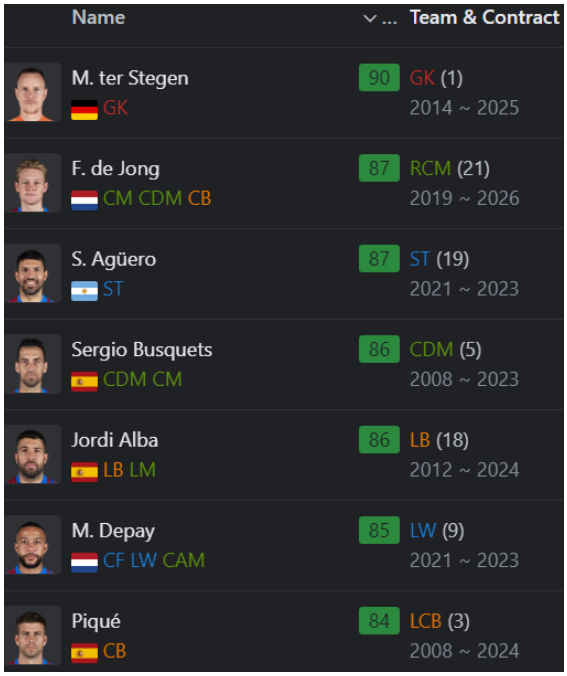
Name	...	Team & Contract
K. Benzema CF ST	89	CF (9) 2009 ~ 2023
T. Courtois GK	89	GK (1) 2018 ~ 2026
Casemiro CDM	89	CDM (14) 2013 ~ 2025
T. Kroos CM	88	LCM (8) 2014 ~ 2023
L. Modrić CM	87	RCM (10) 2012 ~ 2022
Carvajal RB	85	RB (2) 2013 ~ 2025
E. Hazard LW	85	LW (7) 2019 ~ 2024
D. Alaba CB LB	84	LB (4) 2021 ~ 2026
F. Mendy LB	83	SUB (23) 2019 ~ 2025

Thibaut Courtois

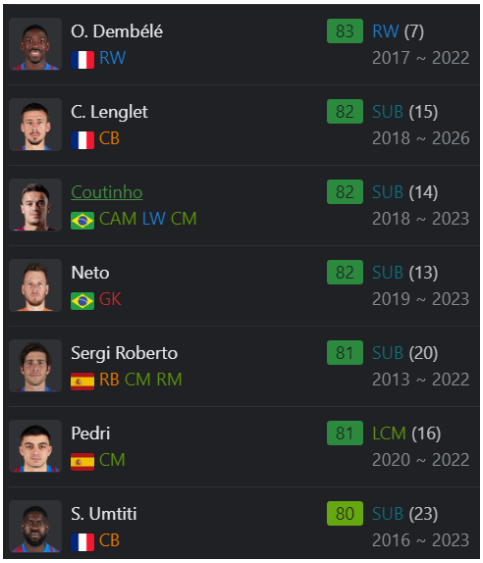


F. Valverde CM	83	SUB (15) 2016 ~ 2027
Éder Militão CB	82	RCB (3) 2019 ~ 2025
Isco CAM CM LW	82	SUB (22) 2013 ~ 2022
G. Bale RM RW	82	RW (18) 2013 ~ 2022
Nacho Fernández CB RB LB	81	LCB (6) 2010 ~ 2023
Lucas Vázquez RW RB RM	81	SUB (17) 2015 ~ 2024
Marcelo LB	80	SUB (12) 2007 ~ 2022
Vinícius Jr. LW	80	SUB (20) 2018 ~ 2025

FC Barcelona:

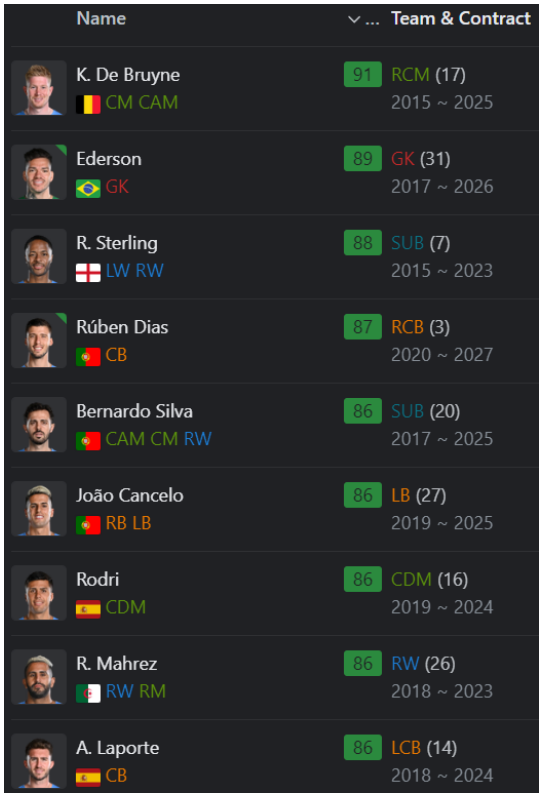


Name	...	Team & Contract
M. ter Stegen GK	90	GK (1) 2014 ~ 2025
F. de Jong CM CDM CB	87	RCM (21) 2019 ~ 2026
S. Agüero ST	87	ST (19) 2021 ~ 2023
Sergio Busquets CDM CM	86	CDM (5) 2008 ~ 2023
Jordi Alba LB LM	86	LB (18) 2012 ~ 2024
M. Depay CF LW CAM	85	LW (9) 2021 ~ 2023
Piqué CB	84	LCB (3) 2008 ~ 2024

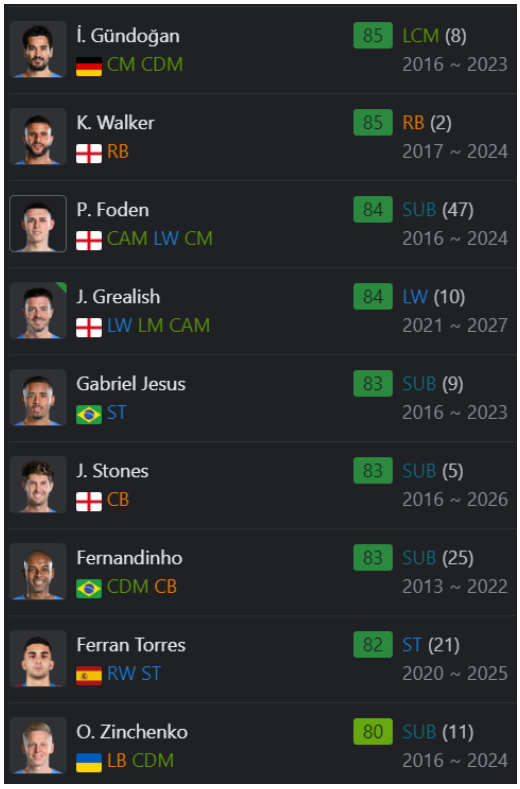


O. Dembélé RW	83	RW (7) 2017 ~ 2022
C. Lenglet CB	82	SUB (15) 2018 ~ 2026
Coutinho CAM LW CM	82	SUB (14) 2018 ~ 2023
Neto GK	82	SUB (13) 2019 ~ 2023
Sergi Roberto RB CM RM	81	SUB (20) 2013 ~ 2022
Pedri CM	81	LCM (16) 2020 ~ 2022
S. Umtiti CB	80	SUB (23) 2016 ~ 2023

Manchester City:

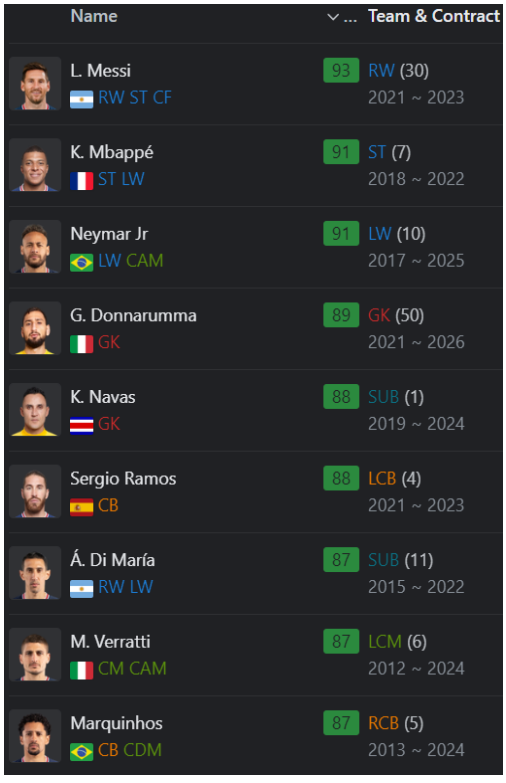


Name	...	Team & Contract
K. De Bruyne CM CAM	91	RCM (17) 2015 ~ 2025
Ederson GK	89	GK (31) 2017 ~ 2026
R. Sterling LW RW	88	SUB (7) 2015 ~ 2023
Rúben Dias CB	87	RCB (3) 2020 ~ 2027
Bernardo Silva CAM CM RW	86	SUB (20) 2017 ~ 2025
João Cancelo RB LB	86	LB (27) 2019 ~ 2025
Rodri CDM	86	CDM (16) 2019 ~ 2024
R. Mahrez RW RM	86	RW (26) 2018 ~ 2023
A. Laporte CB	86	LCB (14) 2018 ~ 2024

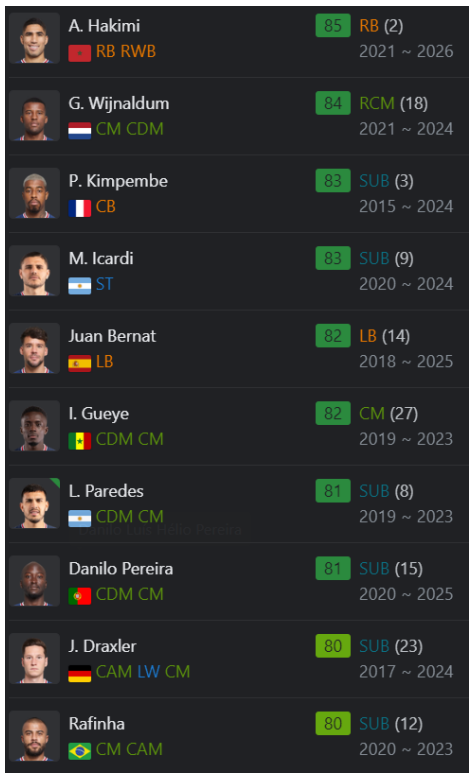


İ. Gündoğan CM CDM	85	LCM (8) 2016 ~ 2023
K. Walker RB	85	RB (2) 2017 ~ 2024
P. Foden CAM LW CM	84	SUB (47) 2016 ~ 2024
J. Grealish LW LM CAM	84	LW (10) 2021 ~ 2027
Gabriel Jesus ST	83	SUB (9) 2016 ~ 2023
J. Stones CB	83	SUB (5) 2016 ~ 2026
Fernandinho CDM CB	83	SUB (25) 2013 ~ 2022
Ferran Torres RW ST	82	ST (21) 2020 ~ 2025
O. Zinchenko LB CDM	80	SUB (11) 2016 ~ 2024

PSG:

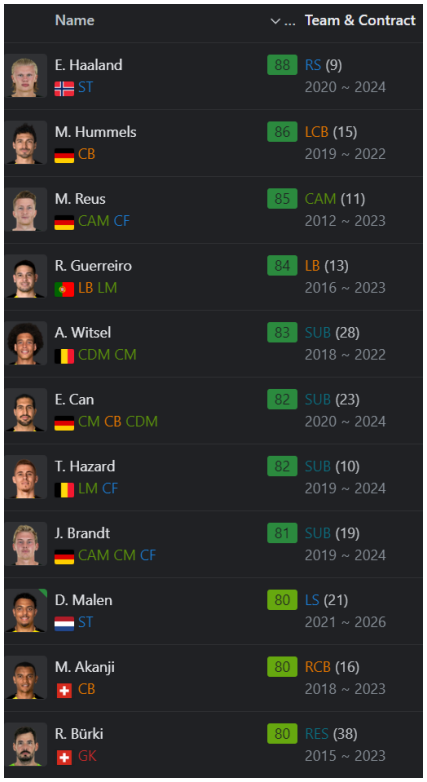


Name	...	Team & Contract
L. Messi RW ST CF	93	RW (30) 2021 ~ 2023
K. Mbappé ST LW	91	ST (7) 2018 ~ 2022
Neymar Jr LW CAM	91	LW (10) 2017 ~ 2025
G. Donnarumma GK	89	GK (50) 2021 ~ 2026
K. Navas GK	88	SUB (1) 2019 ~ 2024
Sergio Ramos CB	88	LCB (4) 2021 ~ 2023
Á. Di María RW LW	87	SUB (11) 2015 ~ 2022
M. Verratti CM CAM	87	LCM (6) 2012 ~ 2024
Marquinhos CB CDM	87	RCB (5) 2013 ~ 2024



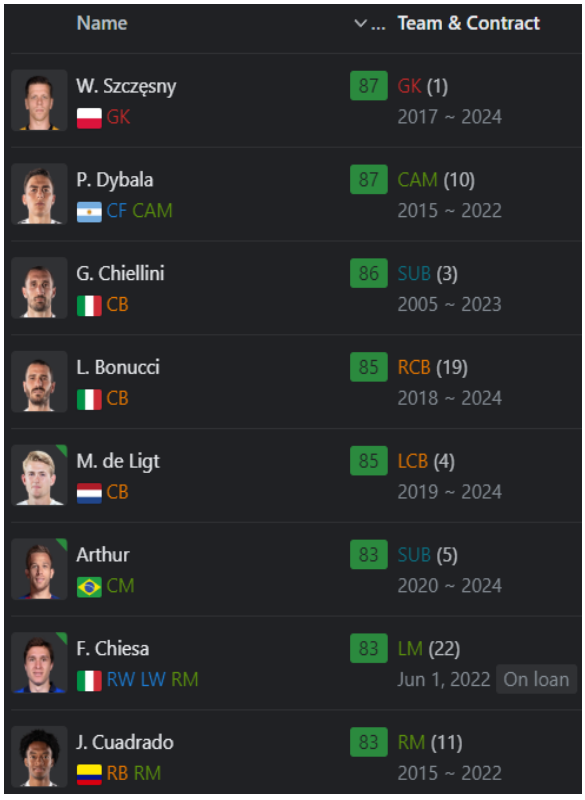
A. Hakimi RB RWB	85	RB (2) 2021 ~ 2026
G. Wijnaldum CM CDM	84	RCM (18) 2021 ~ 2024
P. Kimpembe CB	83	SUB (3) 2015 ~ 2024
M. Icardi ST	83	SUB (9) 2020 ~ 2024
Juan Bernat LB	82	LB (14) 2018 ~ 2025
I. Gueye CDM CM	82	CM (27) 2019 ~ 2023
L. Paredes CDM CM	81	SUB (8) 2019 ~ 2023
Danilo Pereira CDM CM	81	SUB (15) 2020 ~ 2025
J. Draxler CAM LW CM	80	SUB (23) 2017 ~ 2024
Rafinha CM CAM	80	SUB (12) 2020 ~ 2023

Borussia Dortmund:

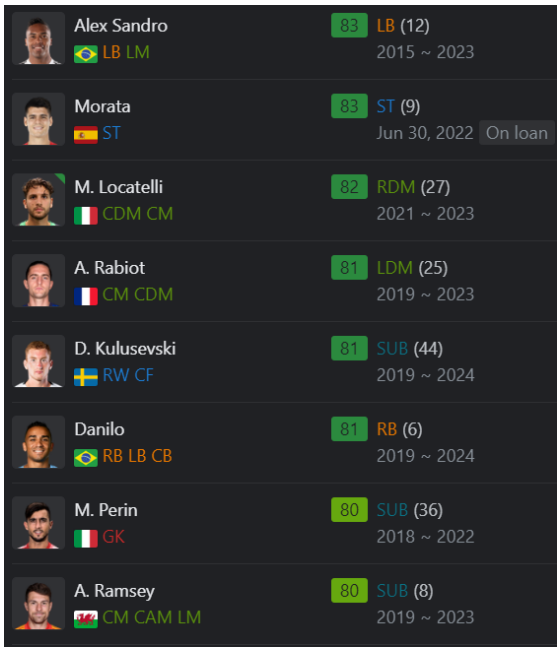


Name	...	Team & Contract
E. Haaland ST	88	RS (9) 2020 ~ 2024
M. Hummels CB	86	LCB (15) 2019 ~ 2022
M. Reus CAM CF	85	CAM (11) 2012 ~ 2023
R. Guerreiro LB LM	84	LB (13) 2016 ~ 2023
A. Witsel CDM CM	83	SUB (28) 2018 ~ 2022
E. Can CM CB CDM	82	SUB (23) 2020 ~ 2024
T. Hazard LM CF	82	SUB (10) 2019 ~ 2024
J. Brandt CAM CM CF	81	SUB (19) 2019 ~ 2024
D. Malen ST	80	LS (21) 2021 ~ 2026
M. Akanji CB	80	RCB (16) 2018 ~ 2023
R. Bürki GK	80	RES (38) 2015 ~ 2023

Juventus:



Name	...	Team & Contract
W. Szczęsny GK	87	GK (1) 2017 ~ 2024
P. Dybala CF CAM	87	CAM (10) 2015 ~ 2022
G. Chiellini CB	86	SUB (3) 2005 ~ 2023
L. Bonucci CB	85	RCB (19) 2018 ~ 2024
M. de Ligt CB	85	LCB (4) 2019 ~ 2024
Arthur CM	83	SUB (5) 2020 ~ 2024
F. Chiesa RW LW RM	83	LM (22) Jun 1, 2022 On loan
J. Cuadrado RB RM	83	RM (11) 2015 ~ 2022



Alex Sandro LB LM	83	LB (12) 2015 ~ 2023
Morata ST	83	ST (9) Jun 30, 2022 On loan
M. Locatelli CDM CM	82	RDM (27) 2021 ~ 2023
A. Rabiot CM CDM	81	LDM (25) 2019 ~ 2023
D. Kulusevski RW CF	81	SUB (44) 2019 ~ 2024
Danilo RB LB CB	81	RB (6) 2019 ~ 2024
M. Perin GK	80	SUB (36) 2018 ~ 2022
A. Ramsey CM CAM LM	80	SUB (8) 2019 ~ 2023

2. Comparison & Strength difference of Home Team vs Opposition Strength (based on all players)

REAL MADRID VS OPPOSITION STRENGTH

In [7]:
```
# Assuming 'df' is your dataset with 'Club', 'Overall', and 'Predicted Overall' columns
# Calculate the average 'Overall' and 'Predicted Overall' for each club
team_strength = df.groupby('Club')[['Overall', 'Predicted Overall']].mean().reset_index()
team_strength.columns = ['Club', 'Actual Overall Strength', 'Predicted Overall Strength']

# Filter for Clubs from Top 5 Leagues
selected_clubs = ['Real Madrid CF', 'FC Barcelona', 'Manchester City','Paris Saint-Germain','Borussia Dortmund','Juventus']
filtered_team_strength = team_strength[team_strength['Club'].isin(selected_clubs)]

# Sort the values from large to small based on 'Team Overall Strength'
filtered_team_strength_sorted = filtered_team_strength.sort_values(by='Predicted Overall Strength', ascending=False)

# Print the team strength and predicted overall for the selected clubs
print("Actual Overall Strength and Predicted Overall Strength for Home & Oppisition Teams")
print(filtered_team_strength_sorted)
```

```
Actual Overall Strength and Predicted Overall Strength for Home & Oppisition Teams
                   Club  Actual Overall Strength  Predicted Overall Strength
467            Juventus                78.727273                   78.786061
592  Paris Saint-Germain                78.585366                   78.511951
309         FC Barcelona                77.025000                   77.070000
637       Real Madrid CF                76.052632                   76.288421
116    Borussia Dortmund                75.342857                   75.242857
525      Manchester City                75.081081                   75.214324
```

In [8]:
```
# Create a DataFrame for the selected clubs and their team strength and predicted overall
team_strength = {
    'Club': ['Real Madrid CF', 'FC Barcelona', 'Manchester City','Paris Saint-Germain','Borussia Dortmund','Juventus'],
    'Actual Overall Strength': [76.052,77.025, 75.081, 78.585,75.342,78.727],  # Replace with actual values if needed
    'Predicted Overall Strength': [76.288,77.070, 75.214, 78.511,75.242,78.786]
}

# Convert it into a DataFrame
df_team_strength = pd.DataFrame(team_strength)

# Set Real Madrid CF as the Home team
home_team = df_team_strength[df_team_strength['Club'] == 'Real Madrid CF']

# Instead of converting the Series directly, use iloc[0] to access the first element
df_team_strength['Predicted Strength Difference'] = df_team_strength['Predicted Overall Strength'] - float(home_team['Predicted C

# Filter out Real Madrid CF to compare with opposition
df_opposition = df_team_strength[df_team_strength['Club'] != 'Real Madrid CF']

# Print the results
print("Comparison of Real Madrid's Predicted Team Strength vs Opposition:")
print(df_opposition[['Club', 'Predicted Strength Difference']])
```

```
Comparison of Real Madrid's Predicted Team Strength vs Opposition:
                   Club  Predicted Strength Difference
1         FC Barcelona                          0.782
2       Manchester City                         -1.074
3   Paris Saint-Germain                          2.223
4     Borussia Dortmund                         -1.046
5              Juventus                          2.498
```

3. Comparison & Strength difference of Home Team vs Opposition Strength (based on key players)

Real Madrid VS OPPOSITION STRENGTH based on key players ¶

In [34]:
```
# Create a DataFrame for the selected clubs and their team strength and predicted overall
team_strength = {
    'Club': ['Real Madrid CF', 'FC Barcelona', 'Manchester City','Paris Saint-Germain','Borussia Dortmund','Juventus'],
    'Team Overall key players': [84.06,84, 85.22, 85.37,82.82,83.12],  # Replace with actual values if needed
    'Team Predicted key players': [84.07,83.65, 85.10, 85.13,82.58,83.12]
}

# Convert it into a DataFrame
df_team_strength = pd.DataFrame(team_strength)

# Set Real Madrid CF as the Home team
home_team = df_team_strength[df_team_strength['Club'] == 'Real Madrid CF']

# Instead of converting the Series directly, use iloc[0] to access the first element
df_team_strength['Predicted Strength Difference'] = df_team_strength['Team Predicted key players'] - float(home_team['Team Predic

# Filter out Real Madrid CF to compare with opposition
df_opposition = df_team_strength[df_team_strength['Club'] != 'Real Madrid CF']

# Print the results
print("Comparison of Real Madrid's Predicted Team Strength vs Opposition based on key players:")
print(df_opposition[['Club', 'Predicted Strength Difference']])
```

```
Comparison of Real Madrid's Predicted Team Strength vs Opposition based on key players:
                   Club  Predicted Strength Difference
1         FC Barcelona                         -0.42
2       Manchester City                         1.03
3   Paris Saint-Germain                         1.06
4     Borussia Dortmund                        -1.49
5              Juventus                        -0.95
```

4. Player Level Comparison

Player level Comparison

In [74]:
```
# Define key players data (Name, Overall, Predicted Overall, and Club Names)
players = ['K. Benzema', 'M. ter Stegen', 'K. De Bruyne', 'L. Messi', 'E. Haaland', 'P. Dybala']
clubs = ['Real Madrid', 'FC Barcelona', 'Manchester City', 'Paris Saint-Germain', 'Borussia Dortmund', 'Juventus']
overall = [89, 90, 91, 93, 88, 87]
predicted_overall = [91.17, 89.90, 90.59, 91.11, 83.48, 86.90]

# Define two different shades for each player's Overall and Predicted Overall (lighter and darker)
overall_colors = ['#AFEEEE', '#C080A5', 'skyblue', '#A9C2D4', '#FFFACD', 'darkgrey']  # For "Overall"
predicted_colors = ['#008B8B', '#800050', 'blue', 'navy', 'Yellow', 'grey']  # For "Predicted Overall"

# Combine player names and clubs for x-axis labels
x_labels = [f'{player}\n({club})' for player, club in zip(players, clubs)]

# Set the figure size
plt.figure(figsize=(10, 7))

# Width of each bar
bar_width = 0.35

# Bar positions for actual overall and predicted overall
index = np.arange(len(players))

# Plot the bars with different shades
bars1 = plt.bar(index, overall, bar_width, label='Overall', color=overall_colors)
bars2 = plt.bar(index + bar_width, predicted_overall, bar_width, label='Predicted Overall', color=predicted_colors)

# Set the labels and title
plt.xlabel('Players and Clubs', fontsize=12)
plt.ylabel('Actual Overall & Predicted Overall', fontsize=12)
plt.title('Comparison of Actual Overall and Predicted Overall for Key Players', fontsize=14)

# Set the x-ticks and add player names and club names
plt.xticks(index + bar_width / 2, x_labels)


# Add values on top of each bar for Overall
for i, bar in enumerate(bars1):
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height() - 5, f'{overall[i]}', ha='center', va='bottom', color='black', fo

# Add values on top of each bar for Predicted Overall
for i, bar in enumerate(bars2):
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height() - 5, f'{predicted_overall[i]}', ha='center', va='bottom', color='

# Display the plot
plt.tight_layout()
plt.show()
```

Strength Difference:

In [75]:
```
import pandas as pd

# Data for key players, their actual and predicted overall ratings
player_strength = {
    'Player': ['K. Benzema', 'M. ter Stegen', 'K. De Bruyne', 'L. Messi', 'E. Haaland', 'P. Dybala'],
    'Actual Overall': [89, 90, 91, 93, 88, 87],  # Actual overalls
    'Predicted Overall': [91.17, 89.90, 90.59, 91.11, 83.48, 86.90]  # Predicted overalls
}

# Convert it into a DataFrame
df_player_strength = pd.DataFrame(player_strength)

# Set K. Benzema as the Home team player
home_player = df_player_strength[df_player_strength['Player'] == 'K. Benzema']

# Calculate the predicted strength difference (difference from K. Benzema's predicted overall)
df_player_strength['Predicted Strength Difference'] = df_player_strength['Predicted Overall'] - float(home_player['Predicted Ove

# Filter out K. Benzema to compare with opposition players
df_opposition_players = df_player_strength[df_player_strength['Player'] != 'K. Benzema']

# Print the results
print("Comparison of K. Benzema's Predicted Overall vs Opposition players based on key players:")
print(df_opposition_players[['Player', 'Predicted Strength Difference']])
```

```
Comparison of K. Benzema's Predicted Overall vs Opposition players based on key players:
          Player  Predicted Strength Difference
1  M. ter Stegen                          -1.27
2   K. De Bruyne                          -0.58
3       L. Messi                          -0.06
4     E. Haaland                          -7.69
5      P. Dybala                          -4.27
```

In [85]:
```
# Data: Predicted Strength Difference for K. Benzema vs Opposition players
players = ['M. ter Stegen\n(FC Barcelona)', 'K. De Bruyne\n(Manchester City)',
           'L. Messi\n(Paris Saint-Germain)', 'E. Haaland\n(Borussia Dortmund)',
           'P. Dybala\n(Juventus)']
predicted_strength_diff = [-1.27, -0.58, -0.06, -7.69, -4.27]

# Bar plot to visualize the comparison
plt.figure(figsize=(10, 8))  # Adjusting size for readability

# Create the bar chart
bars = plt.bar(players, predicted_strength_diff, color=['#800050', 'skyblue', 'navy', 'yellow', 'black'])

# Add title and labels
plt.title("Comparison of K. Benzema's Predicted Overall vs Opposition players", fontsize=14)
plt.xlabel('Opposition Players', fontsize=12)
plt.ylabel('Predicted Strength Difference', fontsize=12)

# Add value labels on top of each bar
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval + 0.05 if yval > 0 else yval - 0.1,
             f'{yval:.2f}', ha='center', va='bottom' if yval > 0 else 'top', fontsize=12)

# Add horizontal line at y=0 for easier comparison
plt.axhline(0, color='black', linewidth=1.5)

# Set y-axis limits from -8 to 1
plt.ylim(-8, 1)

# Rotate x-axis labels for better readability
plt.xticks(rotation=0, ha='center')  # Players' names and clubs will now appear directly under each other

# Show the plot
plt.tight_layout()  # Ensure everything fits into the figure
plt.show()
```