Advanced Programming 2025

Football Transfer Performance Prediction: Model Comparison

Final Project Report

Philippe Borghini Villiger
Philippe.BorghiniVilliger@unil.ch
Student ID: 21420229

December 3, 2025

Abstract

Football has grown massively over the past few decades and is now the most followed sport in the world. As an example, around 1.5 billion people watched the 2022 FIFA World Cup final, which is roughly 20% of the world's population. With football becoming more global and commercial, transfer fees have reached extreme levels in past decades. It is now common for clubs to spend exorbitant amounts of money on players, e.g., PSG paying €222 million for Neymar in 2017. This raises an interesting question: can we actually predict how well a player will perform based on transfer fee and historical data? If so, which Machine Learning model best predicts a player's Goals + Assists in the season following a transfer: Linear Regression, Random Forest Regressor or Gradient Boosting Regressor?

In this project, I compare these three Machine Learning models in Python to see which one best predicts a player's Goals + Assists in the season following a transfer. I collected raw data on performance and transfer fees for midfielders and attackers from the top five European leagues from 2017 to 2024, focusing solely on transfer sums above 5 million. After cleaning and processing the data in Python, I built a training/test setup to evaluate the models.

The results show that the models do not perform equally; some are much more effective at predicting post-transfer performance than others.

Keywords: Machine Learning, football data, regression models, transfer sum, Python

Contents

1 Introduction **2**

2 Literature Review / Related Work **2**

3 Methodology **4**

3.1 Data Description 4

3.2 Approach 5

3.3 Implementation 7

4 Results **8**

4.1 Experimental Setup 8

4.2 Performance Evaluation 8

4.3 Visualizations 9

5 Discussion **10**

6 Conclusion **11**

6.1 Summary 11

6.2 Future Extensions 12

Appendices **12**

A Additional Figures **13**

B Code Repository **13**

C Additional Information **13**

1 Introduction

It has become clear in recent decades that these absurd transfer sums no longer reflect a player's real market value. Some researchers are even talking about an unprecedented speculative bubble [1]. To this extent, it may be interesting to reconsider capital allocation, as more reasonable spending may lead to more productive investments elsewhere [2]. Furthermore, a transfer sum may impose more pressure on the player, as well as on the club from the fans, which may end up having adverse effects on his performance [3].

Football has also become far more data-driven in recent years thanks to ever-evolving technological progress, increasingly complex machine learning techniques and more broadly, the advance in AI [4]. Football has shifted from a pen-to-paper analysis to sophisticated algorithmic software evaluating and analyzing your every move. It's therefore of particular importance to effectively assess historical data and determine a reasonable transfer sum that can maximize the future performance of an incoming player. Consequently, a model allowing a club to forecast a player's performance based on historical data and transfer fees is crucial in this extremely competitive environment.

It goes without saying that estimating something that hasn't occurred yet is very complicated and no matter how accurate we think we are, the future remains unknown and unpredictable. Many indirect factors can affect a player's performance anything ranging from a new boot partnership [5] to personal life struggles. Unaccounted variables such as these in the regression is where the complexity of the matter lies. We therefore can't establish a 1 to 1 relationship between pre-transfer stats and post-transfer stats

Despite the financial implications of transfers, there is little accessible academic work examining the effectiveness of machine learning models at predicting post-transfer performance. Furthermore, we will see that it isn't as straightforward as it may appear to identify which features are most predictive of future performance.

This therefore leads us to the question at hand:

Which Machine Learning model best predicts a player's Goals + Assists in the season following a transfer: Linear Regression, Random Forest Regressor, or Gradient Boosting Regressor?

What motivated me to undertake this project was first and foremost, my love for football but beyond that, my interest in how modern technology and sports can intertwine to improve a specific performance metric. With the advent of AI, an array of new possibilities has become available to us. How can software analyze current data, learn patterns, and thereby produce accurate relationships that can be applied to future data?

The goal of this project is to assess whether post-transfer performance can actually be predicted using Machine Learning.

The objectives

- Build a suitable combined dataset of players that have pre-transfer performance statistics and post-transfer performance statistics that can be used for Machine Learning.
- Train our three ML models and then successfully test them out.
- Assess which ML is more efficient in predicting a future player's performance.
- Understand which features are the most important in future performance prediction

In regard to the report structure, it's organized as follows. We begin with the introduction, which clearly defines the research question and details the context around it, including the objectives and motivations. We then present the related work section where we discuss studies and reports similar to this one on the use of machine learning in football, as well as the datasets employed in those works. Next, we move to the methodology section. Here, we first describe the data, its origin, how it was collected, and why it was merged into a final and complete database. This section also explains the machine learning models used and the overall approach adopted for the analysis. Finally, it covers the implementation details, such as the programming languages and libraries used.The following section presents the results where we report the model outputs and include complementary plots for illustration. We then proceed to the discussion section in which we analyze and interpret these results. Lastly, the conclusion summarizes the main findings and key takeaways of the project.

2 Literature Review / Related Work

With the advent of AI, machine learning has become prevalent in football analytics in recent years [4]. Today ML is applied in many football related contexts, including odds estimation in betting [6], tactical assistance for set-piece strategies [7], score outcome prediction [8] and evaluation of individual player performance [9] such as goals, assists, etc. To this extent, it's also used to assess overall team performance [10], analyze player movement [11], predict incoming injuries based on biomechanical patterns [12] ("study of the structure, function and motion of the mechanical aspects of biological systems"), assist scouters and recruiters in forecasting career progression [13] and estimate transfer value [14].

Major data providers now collect huge amounts of detailed match and player data using advanced cameras and sensors [15]. This data is then processed and sold to clubs, analysts and betting companies, allowing them to build sophisticated predictive ML-based models ranging from expected goals (xG) estimation to injury prediction. Consequently, many wealthy clubs now have dedicated football analytics departments [16] equipped with state-of-the-art software using advanced machine learning techniques.

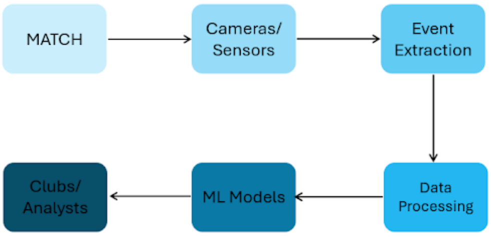

Figure 1: Football data processing pipeline

Nonetheless, while ML has been widely used to evaluate players and forecast their performance within a given team, little work has specifically focused on predicting post-transfer performance [17]. This is where I hope my project can be of particular significance. For instance, there are many projects that focus on predicting a player's transfer value using regression models and basic performance statistics but shy away from actually predicting how well this player will adjust to the new club and perform [18]. E.g., Gianluca Mazza's LSTMFootballPlayerMarketValue project developed a deep neural network (DNN) to analyze historical data and forecast player market value. While this is relevant and can indeed "aid clubs, agents, and analysts in making informed market decisions", it's a relatively well-known field within football analytics. Essentially, it falls short of helping clubs predict a player's future performance following the transfer, which is of immense importance in transfer market decision-making. This is what I seek to address in this project.

There are many studies that have attempted to predict football performance using Machine Learning or even classical statistical methods [8]. One can find many studies focusing on forecasting in season performance metrics such as goals and assists. Studies going as far back as 1982 have attempted to predict future performance using statistical modeling techniques. [19] E.g., Maher used Poisson-based models applied to historical match data to predict the number of goals that teams would score in future fixtures. While this isn't ML, it demonstrates there has always been a strong interest in predicting future performance in football. Other studies have focused on evaluating player performance across multiple leagues .E.g., PlayeRank: Data-driven Performance Evaluation and Player Ranking in Soccer via a Machine Learning Approach [20] by Pappalardo et al. develops a multi-dimensional ML system that can evaluate and compare players across leagues and roles. While this study allows performance to be measured and compared across different leagues, it does not address the impact of in-league or cross-league transfers.

There have even been studies examining how a player's performance evolves with age. E.g., Branquinho et al. (2025) How Age Affects Physical Performance in Elite Football [21] demonstrates that physical performance (speed, endurance..) reaches its peak around 25 years and tends to gradually decline thereafter (see Figure 2). While this may give insights on how a player will perform based on his age, it doesn't account for the multitude of direct and indirect factors that can affect a player's performance following a transfer.

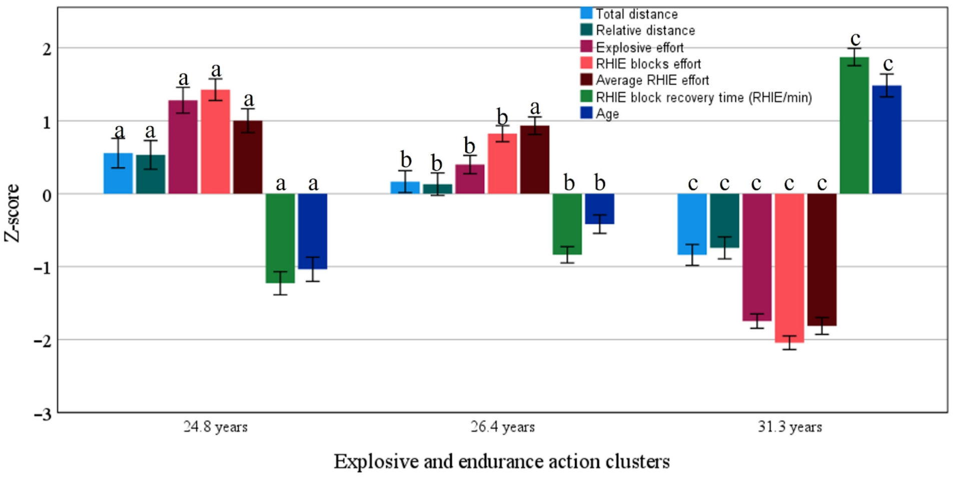

Figure 2: Endurance and Explosion Capabilities According to Age

Predicting post-transfer performance is vastly different than predicting regular upcoming-season performance. A transfer introduces a significant shift in a player's environment: new teammates, new stadium, new infrastructure, new coach, new coaching staff, etc. These on-pitch changes can vastly affect an incoming player's performance. Additionally, many off-pitch factors can also lead to a drastic change in performance e.g., new language, different weather, local media pressure, pressure surrounding a high transfer fee, etc. As we have seen, most existing studies are aimed at predicting performance in a same-club environment, meaning they do not account for all these extra factors that come with a transfer.

We therefore observe that, on one hand, machine learning has made considerable progress in evaluating and forecasting football performance. However, the existing studies overwhelmingly assume that the player remains in the same club. On the other hand, very few studies examine how a player's performance evolves following a transfer. While this may initially seem like a subtle distinction, it represents a huge potential shift in performance, as a transfer introduces a multitude of direct and indirect factors capable of influencing future performance. This is precisely the information gap that the present project aims to fill by evaluating whether machine learning models can predict a player's Goals + Assists in the season following a transfer.

3 Methodology

3.1 Data Description

The data was initially separated into two parts: the historical performance data and the transfer fee data. Given the fact that there weren't any functional Python libraries allowing us to directly extract all necessary data for the project, I had to extract and make the data myself.

The aim was obviously to end up with a larger subset of players having historical data before and after their transfer to the extent that the following conditions were met:

- Only the top 5 major leagues (the most important leagues with the bigger dispersion of transfer sums [22]).
- Only attackers and midfielders (their success criteria are similar, i.e., scoring goals or assisting them, in contrast with a defender whose purpose is to concede as few goals as possible).
- Only transfers with a 5 million euro fee and greater (in order to filter out the worst signings with less playtime) from the 2016-2017 season to 2023-2024.

Transfermarkt

I went on Transfermarkt and manually made CSV files (no option to extract them directly) for every league and every year falling in the criteria range, e.g., `Bundesliga_2021`, `Serie_A_2018`, . . . Every single one of these raw CSV files contained the player's name, age, nationality, position, market value, previous club, and transfer value. I then made a Python file (`data_collection_main.py`) that read and loaded each CSV file, extracted the necessary data (i.e., player, age, nationality, position, market value, previous club, transfer fee, source file, and league), and made a combined dataset for all 5 leagues in each season. This was the preliminary dataset that would be used for merging later on, called `all_transfers_combined.csv`. Although preliminary filtering according to the criteria was done initially when making the CSV files, in order to ensure that no outliers were left out, I conducted an additional filtering phase in the `data_collection_main.py`. This returned a new, now fully filtered dataset of 709 players respecting the criteria, named `transfers_filtered.csv`.

FBref

I went on FBref and manually made CSV files (no option to extract them directly) for every league and every season, e.g., `premier_league_2022-2023.csv` and `laliga_2019-2020.csv`. Each file had a plethora of features, anything ranging from yellow cards to penalty kicks. There was unfortunately no option to preliminary filter out what we didn't want, such as defenders and goalkeepers, useless features, so this was to be done later. We first combined all of these CSV files into one big merged dataset, `fbref_stats_raw.csv`, via `combine_fbref_files.py`. In addition, this script normalized multi-index headers and ensured consistent column naming. We subsequently made a script to remove all unnecessary columns (e.g., all the unnamed columns), standardize player names, and convert statistical columns into numeric ones (e.g., "90 mins" → 90). In addition, based on the available data, it computed lacking crucial input features for the project (i.e., goals per 90 min, assists per 90 min, and G + A per 90 min).

Once both of these merged subsets have been acquired, we can now focus on combining them together.

Thanks to functions from the `fuzzywuzzy` library that are able to assess similarity between strings and find the best match between them, we were able to make several merged datasets of interest. The `match_transfers.py` file uses these functions to match, respectively, pre-transfer FBref statistics and post-transfer FBref statistics with the transfer fee. For each transfermarkt transfer, the script finds the before-transfer season, filters the dataset accordingly, keeping only the seasons we are interested in, and via fuzzy matching, matches on player names within that league-season to find the closest player entry. Same for post-transfer season data. Fuzzy is essential, as different data sources may write names differently (e.g., Cristiano Ronaldo and C. Ronaldo).

This file returned 3 different CSV files:

1. `transfers_matched_complete.csv`: this is the one that will actually be used for machine learning; it contains all players whose transfer fee was successfully matched with pre- and post-transfer historical data. Features include everything we need for our ML training/testing, such as G+A per 90 min pre-transfer and G+A per 90 min post-transfer. We ended up with 233 rows of data.
2. `transfers_matched_all.csv`: this contains transfer records matched in at least one season (pre, post, or both). We ended up with 654 rows of data. This is not used for ML.
3. `transfers_unmatched.csv`: this includes all those players who weren't successfully matched in either season and are therefore useless to our ML. 87 rows.

3.2 Approach

Three regression models were selected to predict post-transfer performance:

- Linear Regression,
- Random Forest Regressor,
- Gradient Boosting Regressor.

The target variable y is the player's total Goals + Assists in the season following the transfer. The input features x include:

- Pre-transfer performance metrics (goals, assists, xG, xAG, Goals+Assists per 90, etc.),
- Transfer-related variables (transfer fee, market value, age, position, league, etc.).

Linear Regression

Linear Regression is a statistical model that predicts a target variable y by assigning coefficients to the input features x such that a linear relationship exists between them (see Figure 3). In other words, it gives the straight line (or hyperplane) that best explains how changes in the features influence the output. In this project, the model was used to test whether expensive players perform proportionally to their transfer fee and their historical data. Here, y corresponds to the player's after-transfer season performance (total G+A), and x consists of the pre-transfer performance metrics (e.g., goals, assists, xG, xAG, G+A per 90) as well as transfer-related variables (transfer fee, market value, age, position, etc.).

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon.$$

This model stands out for its straightforward interpretability, making it useful as a baseline to understand how each variable influences post-transfer output.

Random Forest Regressor

Random Forest is a machine learning model where each decision tree works by choosing thresholds on individual features. For example, a split like xAG > 0.4 will divide players into two groups: those who satisfy the condition and those who don't. For every possible split, the model calculates the mean squared error (MSE) in each subgroup and compares the combined MSE to the MSE before splitting. If this new split reduces the overall error, it means the prediction becomes more accurate, so the split is kept. The tree then tries many different threshold values for that feature, and then does the same with the other features. By keeping the splits that reduce error the most, each tree ends up with a set of thresholds that best predict post-transfer performance.

Once all trees are trained independently, the Random Forest gives its final prediction by simply averaging the outputs of every tree (see Figure 3). When the relationship between inputs and outputs is more complex or non-linear, having many trees helps the model capture these patterns much better than a single tree could. It also reduces overfitting because no single outlier ends up corrupting the prediction, which makes the model more robust overall.

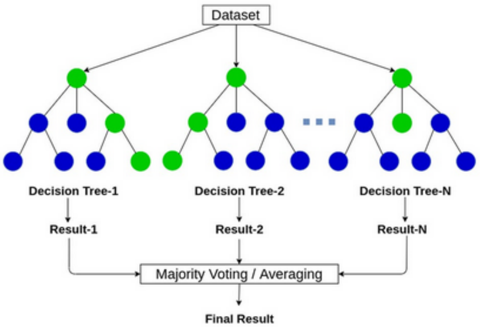

Figure 3: Random Forest Tree Overview

Gradient Boosting Regressor

The Gradient Boosting Regressor is a method that builds decision trees sequentially, where each new tree focuses on correcting the errors made by the previous one (see Figure 4). It still uses thresholds like a normal decision tree, but instead of trying to explain the whole prediction, each new tree learns a small correction of the mistakes made earlier. E.g., the first tree might split on xG and if that split results in a relatively high MSE for certain players, the next tree will try to reduce this by splitting on another feature such as xA. By stacking these small corrections, the model gradually becomes very accurate over time.

This approach captures non-linear relationships between x and y and picks up subtle patterns and interactions, e.g., how league quality might interact with transfer fee. It fits our project well because it can model these nuanced relationships without overfitting.

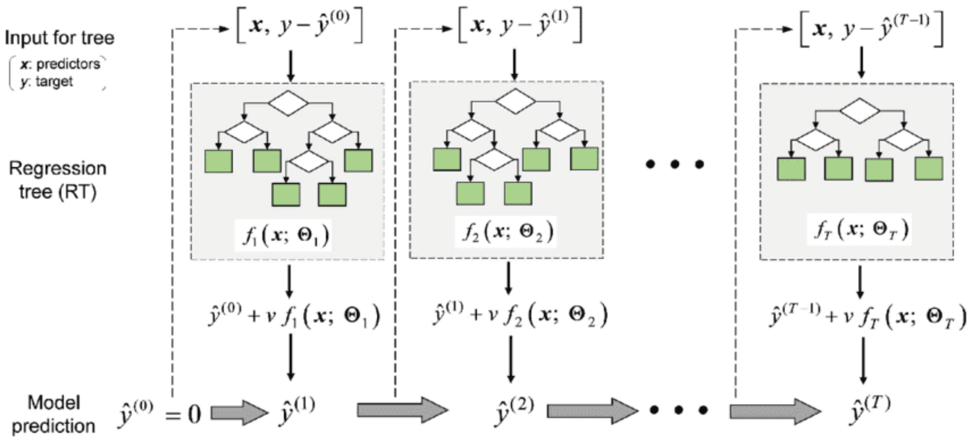

Figure 4: Gradient Boosting Regressor Tree Overview

Evaluation Metric

To evaluate the models, we use the coefficient of determination R^2:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}.$$

This measures the proportion of variation in y (post-transfer G+A) that is explained by the model's predictions $\hat{y}$. It is well suited here because:

- It can be interpreted relative to a baseline that predicts the mean: $R^2 > 0$ means better than predicting the average, $R^2 = 0$ equal, $R^2 < 0$ worse.
- It is commonly used in sports analytics.
- It is easily comparable across different models.

3.3 Implementation

The implementation was done thanks to different python files that each served different purposes. Python is the standard language for data analysis and machine learning.

Libraries. The main libraries used are:

- `pandas` for dataset manipulation
- `numpy` for numerical operations
- `scikit-learn` for ML models, train-test split and $\mathbf{R^2}$ metric
- `fuzzywuzzy` to match a player's transfer fee and his historical performance
- `matplotlib` and `seaborn` for plotting.

Project structure. The project is set up in such a way that all the data is already ready for use i.e., processed, merged and combined. If you create and activate the conda environment, the project should run smoothly upon running `main.py`.

As mentioned previously, the preliminary merging and combining of the various FBref and Transfermarkt csv files was done in data-`collection_main.py`, `combine_fbref_files.py`, etc. I ultimately ended up with a combined merged dataset called `transfers_matched_complete.csv` which was to be used for the ML testing/training.

We first split the dataset into a training set and a testing set using an 80/20 split, this is done directly in `main.py` via the train_test_split function from scikit-learn. This ensures that 80% of the data is used for training while the remaining 20

All preprocessing was handled directly inside `main.py`. When executed, `main.py` begins by loading the merged dataset and applying all preprocessing steps needed for ML use i.e., selecting numeric features, converting transfer-related variables (like age and market value) into numeric format, and encoding categorical variables such as position and league.

Once the data is prepared, the file imports the three ML models from `modeling.py` and trains each model on the training subset. For every model, the script receives the trained model object (e.g., RandomForestRegressor()), the R^2 score on the training set, the R^2 score on the testing set, and the predicted post-transfer G+A values for the players in the test set. These results are then displayed using functions from `evaluation.py` that compute the R^2 score and print them.

In essence, `main.py` acts as the central hub for the entire project: it handles data loading, preprocessing, model training, model evaluation, and the generation of all outputs

4 Results

4.1 Experimental Setup

Hardware.

- CPU: Intel i7
- RAM: 16 GB

Software / Libraries.

- Python 3.10
- `pandas`
- `numpy`
- `scikit-learn`
- `matplotlib`, `seaborn`
- `fuzzywuzzy`
- `glob`,
- `os`

Hyperparameters.

- Linear Regression: default hyperparameters
- Random Forest Regressor:
  - `n_estimators = 300`
  - `max_depth = None`
  - `random_state = 50`
  - `n_jobs = -1`
- Gradient Boosting Regressor:
  - `n_estimators = 300`
  - `learning_rate = 0.05`
  - `max_depth = 3`
  - `random_state = 70`

4.2 Performance Evaluation

Table 1: Model Performance (R^2 score on test set)

Model	**R^2 Score**
Linear Regression	**0.1900**
Random Forest Regressor	0.1725
Gradient Boosting	0.0807

Among the three ML models, the Linear Regression model is the most accurate. This model explains about 19% of the variation in next-season goals and assists (i.e., after_G+A). Falling closely behind, we find the Random Forest Regressor with an R^2 of roughly 17%. This means that the model explains around 17% of the variation in next-season performance. Gradient Boosting comes in last place, explaining only about 8% of the variation in next-season goals and assists.

4.3 Visualizations

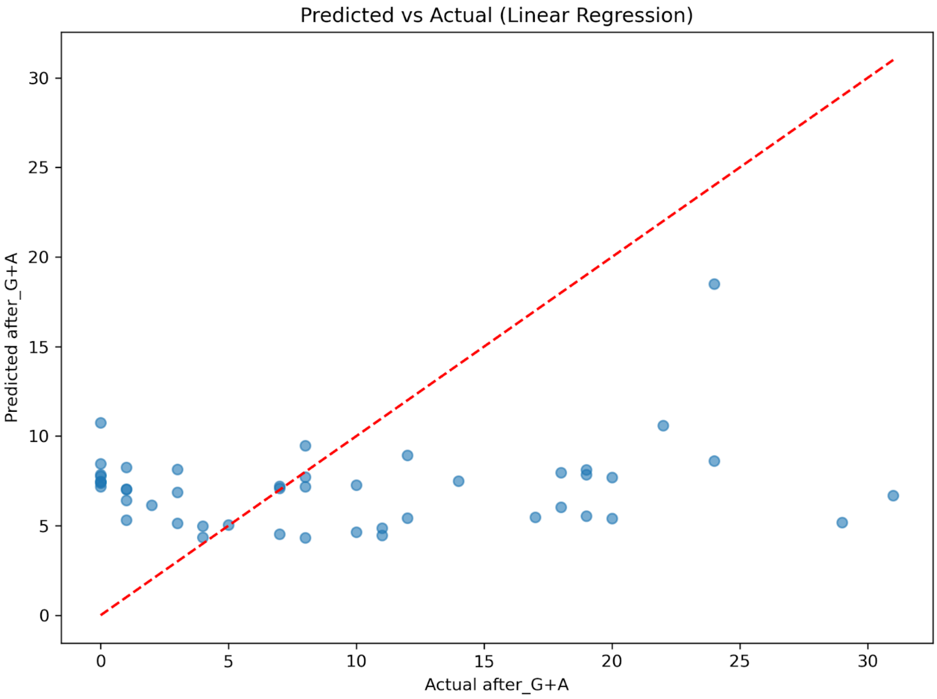

Figure 5: Linear Regression: Predicted vs. Actual Post-Transfer G+A

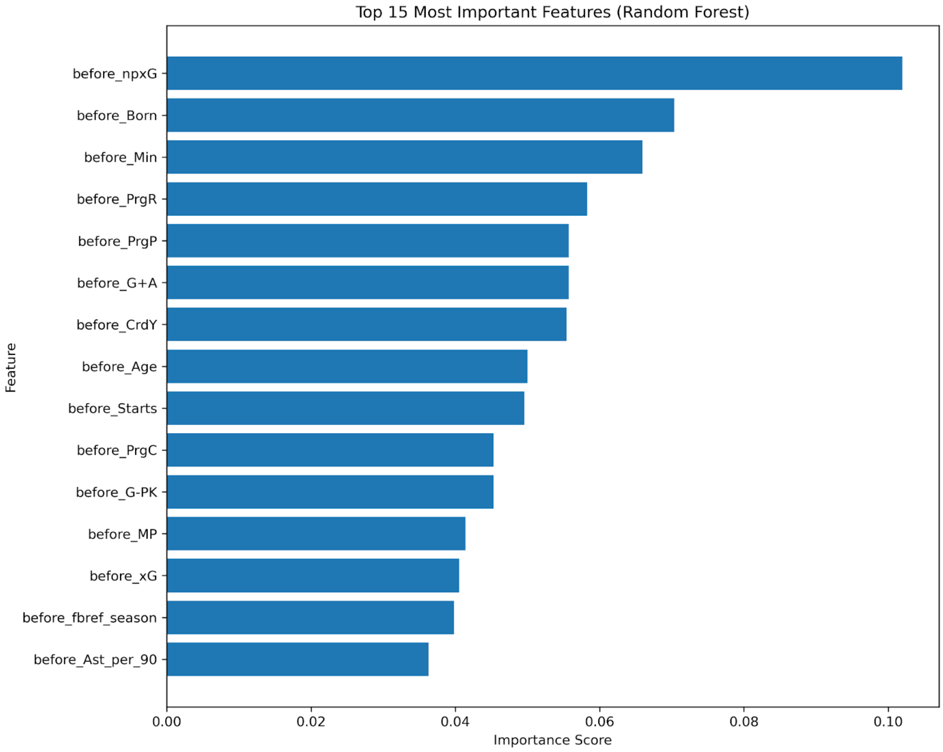

Figure 6: Random Forest Feature Importances in Predicting Post-Transfer G+A

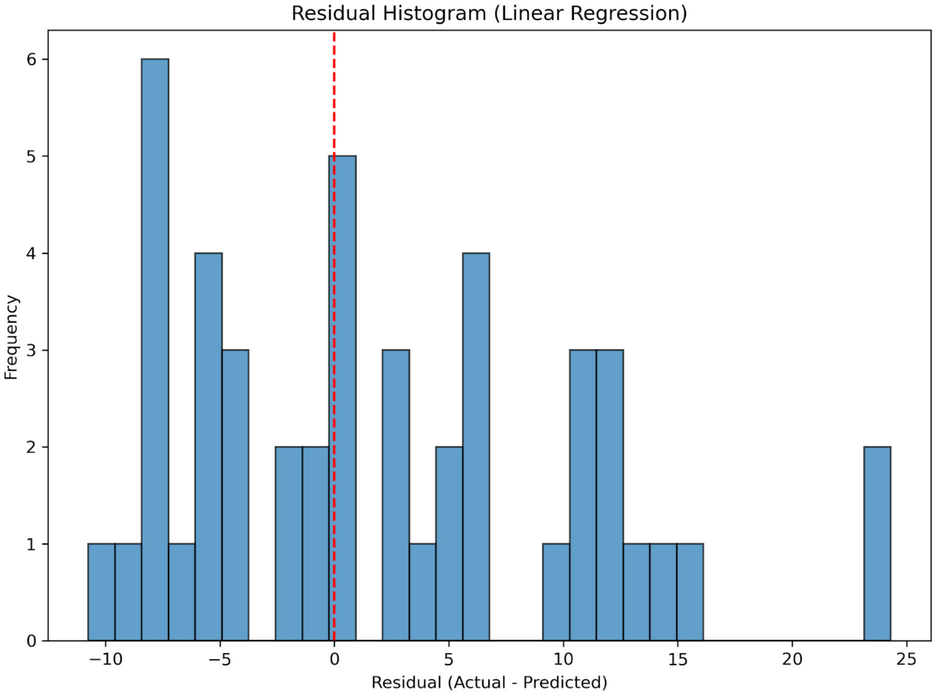

Figure 7: Linear Regression Residual Distribution

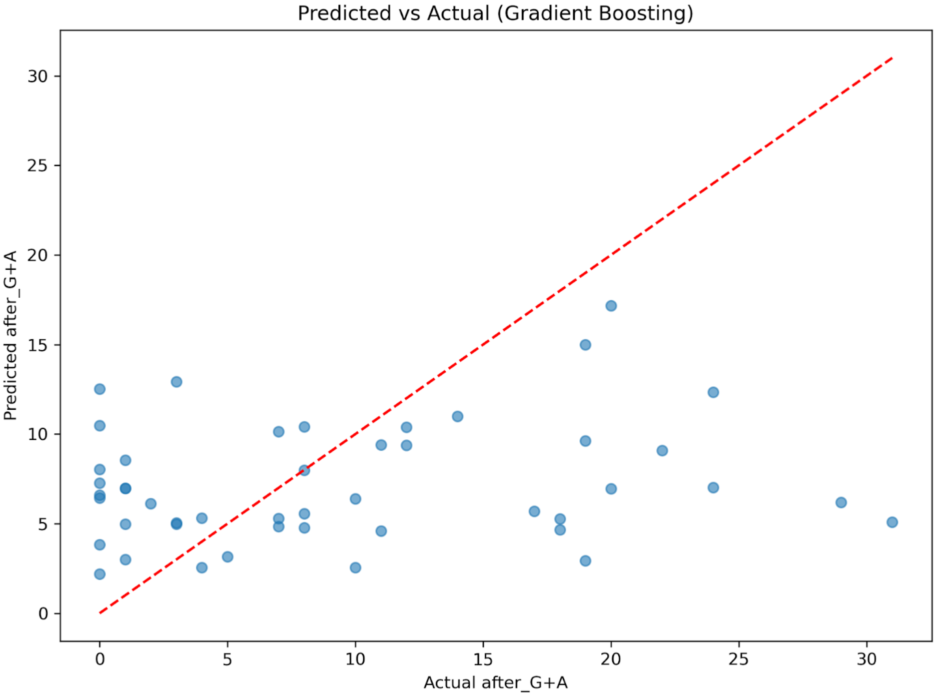

Figure 8: Gradient Boosting: Predicted vs. Actual Post-Transfer G+A

5 Discussion

Linear Regression ended up performing the best with an R^2 of 0.19, followed by Random Forest at 0.1725 and finally Gradient Boosting at 0.0807. The fact that the linear model came out on top suggests that the dominant relationships in the dataset behave in a roughly linear way. This validates football logic as past performance (goals, assists, etc.) tends to be one of the most reliable indicators of future performance. Last season's penalty kicks scored was the most important feature in predicting future output; with a coefficient near 1, the model therefore suggests that holding all other features constant, each penalty scored last season translates to roughly one additional G+A in the following post-transfer season.

Random Forest performed slightly worse than Linear Regression but still fairly well. It was able to capture some non-linear interactions between variables such as transfer fee, past performance, and future G+A. Random Forests are known to handle noisy data quite effectively. This is especially relevant here given the presence of outliers both on the transfer front and performance front, e.g., Ronaldo's 48 G+A league season in 2014–2015.

Gradient Boosting on the other hand, performed drastically worse with an R^2 of only 0.0807. This model is much more sensitive to noise and usually requires larger datasets [23] to function properly. This helps explain why it performed noticeably worse than Random Forest, despite both models being tree-based. Although Linear Regression was the most accurate of the three models, all models struggled to predict future performance. This can be explained by the fact that on one hand, football is very noisy [25] [26] (especially now with extreme outliers in the transfer market). And, on the other hand football performance post-transfer is affected by a plethora of direct and indirect factors unaccounted for in the dataset (e.g., new teammates, new coaching style, etc.).

Nevertheless, the models still performed better than simply averaging past goals and assists, indicating that the available features do contain some predictive signal.

What worked well was the fact that despite the coefficient of determination being low for all three models, they still provide value as they are more effective at predicting future performance than simply predicting the mean post-transfer G+A for every player. After all, the Linear Regression's predictions explain roughly 20% of the variation in post-transfer G+A.

What was especially challenging was the data collection process as there were no updated libraries that could directly extract the data for me. I had to manually make csv files for each season, each league for both FBref and Transfermarkt. Additionally, for a variety of reasons e.g., players incoming from foreign leagues, position changes . . . We ended up with a far smaller combined dataset than anticipated at 230 rows which increased the risk of overfitting. Moreover, the multiple unaccounted factors discussed earlier, contribute to the likelihood of an omitted variable bias.

I obviously expected the ML models to be more accurate at predicting future performance. One would naturally assume that key performance metrics such as goals, assists, or G+A per 90 min would play a much stronger role in predicting post-transfer performance. That being said, this highlights the importance of all the limitations of this project i.e., small dataset, omitted factors, extreme outliers, high-variance G+A, etc. Additionally, it's worth pointing out that training models on a singular season is a bit of an unreliable approach as football performance can be highly volatile for a given player, from one season to another [27] (e.g., Mahrez went from 28 G+A in 2015/2016 to 9 G+A in 2016/2017)

6 Conclusion

6.1 Summary

This project set out to determine whether Machine Learning models could predict a player's Goals + Assists in the season following a transfer using pre-transfer performance metrics and the transfer fee itself. It also aimed to evaluate which of the three tested models i.e., Linear Regression, Random Forest, and Gradient Boosting would be the most accurate. Linear Regression performed best with an R^2 of 0.19, meaning it explains about 19% of the variation in post-transfer G+A. Random Forest followed closely at around 17%, while Gradient Boosting was the least accurate with an R^2 of 0.0807. Since all three models achieved an R^2 above zero, they outperform the baseline that predicts the average post-transfer G+A for every player.

Although the predictive power of the models is relatively low, the results show that historical data and transfer-related variables do contain meaningful information. At the same time, the unexpectedly low scores highlight the importance of omitted factors such as adaptation, new teammates and league transitions, and reveal just how noisy football performance can be, especially when working with a small dataset sensitive to extreme outliers.

Overall, this project highlights both the value and the limitations of statistical modeling in football prediction. While the models used here are inherently limited, they show that ML-driven approaches can extract useful patterns from historical performance. With larger datasets, additional contextual variables, and more complex models, Machine Learning has the potential to become an extremely powerful tool in forecasting football outcomes in the years to come.

6.2 Future Extensions

Future extensions of this project could drastically improve model performance and address the limitations encountered. First, collecting a larger dataset, ideally covering multiple seasons per player instead of one would help reduce noise and allow the models to learn more stable patterns. Incorporating additional contextual variables would also be important, such as expected playing time, tactical role, team strength, coaching changes and injury history, all of which strongly influence post-transfer performance but are not captured in the current dataset. It would also be interesting to apply other ML models to this problem, such as CatBoost or Extreme Gradient Boosting which may provide stronger predictive power. Finally, besides predicting post-transfer G+A, this framework could be used to forecast other football-related outcomes, such as transfer value changes or expected playing time.

It would also be interesting to apply other ML models to this problem, such as CatBoost or Extreme Gradient Boosting, which may provide stronger predictive power and handle categorical and imbalanced data more effectively.

Finally, besides predicting post-transfer G+A, this framework could be extended to forecast other football-related outcomes, such as transfer value changes, expected playing time, or even the probability that a player will move again within a certain time horizon.

References

1. Fry, J. & Binner, J. *Quantifying speculative-bubble effects in major European soccer leagues.*
2. Öner, Ö., Özgür Karataş, & Öztürk Karataş. *Financial Sustainability in Football Clubs.*
3. Mittal. *High Value Football Transfers: A Win or Loss? Correlation of High Value Football Transfers with Their Subsequent Performances.*
4. Markel Rico-González et al. *Machine learning application in soccer: a systematic review.*
5. Henning & Sterzing. *The influence of soccer shoe design on playing performance.*
6. Stübinger, Mangold & Knoll. *Machine Learning in Football Betting: Prediction of Match Results Based on Player Characteristics.*
7. Wang & Veličković. *TacticAI: an AI assistant for football tactics.*
8. Rodrigues & Pinto. *Prediction of football match results with Machine Learning.*
9. Chandra B., Shinny D. & Adhitya M. *Prediction of Football Player Performance Using Machine Learning Algorithm.*
10. Moustakidis et al. *Predicting Football Team Performance with Explainable AI: Leveraging SHAP to Identify Key Team-Level Performance Metrics.*
11. Wang. *Analyzing Players' Tactical Positions and Movement Trajectories in Soccer Matches Using Machine Vision Algorithms.*
12. Majumdar, Bakirov, Hodges et al. *Machine Learning for Understanding and Predicting Injuries in Football.*
13. Hugo Vicente. *How AI and Data are Shaping the Future of Scouting.*
14. Tamim et al. *Machine learning-driven market value prediction for European football players.*
15. https://www.statsperform.com/?utm-source=chatgpt.com
16. Lolli et al. *Data analytics in the football industry: a survey investigating operational frameworks and practices in professional clubs and national federations.*
17. Dinsdale & Gallagher. *Transfer Portal: Accurately Forecasting the Impact of a Player Transfer in Soccer.*
18. https://github.com/gianlucamazza/LSTM-FooballPlayer- MarketValue.
19. Maher *Modelling association football scores 1989.*
20. Pappalardo et al. *PlayeRank: Data-driven Performance Evaluation and Player Ranking in Soccer via a Machine Learning Approach.*
21. Branquinho et al. *The Aging Curve: How Age Affects Physical Performance in Elite Football.*
22. Poli, Besson & Ravenel. *The economics of big-5 league transfers: past decade and post-pandemic.*
23. https://wandb.ai/mostafaibrahim17/ml-articles/reports/An-introduction-to-Gradient-Boosted-Trees-for-machine-learning–Vmlldzo2NTQ4NzYx
24. Breiman *Random Forests*
25. Poli, Besson et Ravenel *Statistical Modeling of Football Players' Transfer Fees Worldwide*
26. Rampinini, Coutts and Castagna *Variation in Top Level Soccer Match Performance.*
27. https://www.thesun.co.uk/sport/3487580/riyad-mahrez-stunning-decline-leicester-barcelona-arsenal/

A Additional Figures

Additional Figures can be found in the github repository in the results folder.

B Code Repository

https://github.com/philippebvilliger/project-repo

- Data processing and merging code can be found in src.
- Plot code can be found in src.
- Main code for training, testing and evaluating ML models can be found directly in the project root.

C Additional Information

LLMs such as ChatGPT and Claude were used in this project to structure the report, explain ML concepts and review code logic.