Personalising Market Values of Football Players Using Machine Learning

CS310 Computer Science Project

Paddy Connolly

2122558

Supervisor: Graham Cormode

Department of Computer Science

University of Warwick

Contents

Abstract

In football today there seems to be very high transfer fees with no guarantee of performance. In this project we will explore how the same player could have different values depending on the team they would play in. This project focuses on developing a solution use machine learning techniques, which generates a “true value” for a player. This is their intrinsic value, independent of their teammates. Using this, we can see how each player would perform in another team, and therefore see how valuable they would be to that specific team. The aim of this project is to generate a personalised market value for a player to a given team, instead of a general market value figure.

Keywords: Sports, Football, Machine Learning, Python, Gradient Boosting, Transfers, XGBoost

1 Introduction

The football transfer market has been increasing in size since the sport became professional, but in recent years it has reached new heights. In 2023, football clubs globally spent a combined \$9.63bn (€7.62bn) on transfer fees.[1] The ten most expensive transfers in football history have all occurred in the last 7 years, and 5 players were involved in deals worth over €100 million in 2023 alone.[2–11]

Due to rising transfer fees, club owners are increasingly concerned with using new techniques to maximise their sporting and financial success. However, it's not just club owners who want their transfers to be successful, there are a large number of stakeholders in every football transfer, including but not limited to players, agents, scouts, sponsors, coaches and supporters, with the latter accounting for over 5 billion people worldwide.[12]

This has led to many clubs trying to use data and statistical methods to predict and quantify performance. With recent improvements in both player and ball tracking devices, as well as video analysis techniques, there is more data than ever available in modern-day football. Since data is now available throughout entire football league systems across the world, clubs can spot players with the potential to be very successful earlier than they could have done before. This allows clubs to recruit players at a younger age before their market value increases, which in turn helps them to maximise their transfer profits. The massive sums of money involved and the wide availability of data make it both important and possible to use machine learning techniques as a predictor for performance.

However, traditional methods of scouting would not just look at how well the player is performing but would take factors into account such as whether they would fit well into a scout's team or how they interact with their teammates. For example, they may look at whether the player is a selfless player, who sets the team up for success, or a player who can make an impact on their own. Many existing market value predictions view players as a discrete entity, and take a simplified view of their stats as being independent from each other, instead of there being countless intricate relationships between the stats of players in the same team.

It is important to remember that the goal of the sport is to create the most successful team possible and not the team with the highest combined market value. Not only do we have to look at how a player would fit into a new team, but we also have to look at how their current team affects their on-pitch output. Data alone will not provide the entire picture, as we can't be sure that their stats are not affected by the strengths and weaknesses of their teammates. To accurately assess a player's true ability and potential, a method is required to isolate the individual contributions of a player, in order to get a clear idea of how they would perform in a different team.

The goal of this project is to develop a comprehensive, data-driven approach which is adapted to finding players who are undervalued in the top European leagues. This will involve analysing historical performance and transfer data to find a player's true value, independent of their teammates. Through understanding a player's true worth, we can provide tailored recommendations to specific clubs, based on which clubs the player would be the most valuable to. The ultimate goal is to allow sports organisations to make more informed and cost-effective player decisions, and to provide a structured and transparent analytic approach which can be adapted to their needs.

2 Background

2.1 Existing Value Predictions

Transfer market value predictions for football players are already common in football and many are publicly available. One such source is Transfermarkt.com, described as “a cornerstone of soccer’s digital age.”[13] It is important to note these predictions do not come from an algorithm instead a strict process is followed for each individual market value:

- The community (50,000 active members) vote on whether a value should increase, decrease or remain unchanged.
- Changes that have supporting statistical evidence are selected by the appropriate region manager.
- These changes are then moderated by the site owner to ensure consistency.

Before considering specifically which metrics are included in these predictions, it is important to clarify the exact definition of the market value prediction. Two factors not included in transfer market values are important to note.

Market values are not a prediction of the price that a player would be bought or sold for right now, instead, they attempt to predict the value of a player in a hypothetical free market scenario, for example, the price each player would sell for if every single player was released from their contract.

Although higher or lower wages would affect the predicted value slightly, the number does not include wages, and values are not correlated with wages. Many high-performing players approaching the end of their careers would be on high wages but would have a low transfer value since they would be expected to retire soon or see their performance decline. Since it is impossible to offer anything more than a prediction of transfer market value, the terms ‘transfer market value’ and ‘transfer value prediction’ are used interchangeably throughout the report.

2.2 Existing Machine Learning Methods

Two papers which this project took inspiration from, *Players' Value Prediction Based on Machine Learning Method*[14] and *Learning football player features using graph embeddings for player recommendation system*,[15] both successfully employ the XGBoost algorithm.

The first paper achieves relatively high accuracy in predicting player market values using player data from the FIFA video games, compared to both neural networks and other machine learning methods. They describe XGBoost as a suitable method due to the fact that it can "process high-dimensional sparse features in a distributed manner" and is "more efficient than similar algorithms". They also mention how they found success using a grid search to optimise XGBoost's hyperparameters, indicating high flexibility. They found success using the `reg:gamma` objective function, which they found through a simple grid search, which should also be considered for our model. The features used were the player stats used in the FIFA18 game, with 64 features used in total (54 for goalkeepers) and the ground truth was the market value attached to each player in the game. The top 10 features by importance are given below:

1. `potential`
2. `birth_date`
3. `reactions`
4. `ball_control`
5. `positioning`
6. `marking`
7. `heading_accuracy`
8. `interceptions`
9. `crossing`
10. `vision`

`potential` is a score based on the potential ability improvement over time, `vision` refers to being able to see potential passes before others

The second paper also shows the success of XGBoost in player recommendation, but this time in the context of selecting substitutes to replace players on the pitch, which in effect is similar to a transfer in this project, in the sense that a transfer is essentially the permanent replacement of a player in a team. They trained an

XGBoost model using graph embeddings and found it outperformed simple k-NN models. They concluded XGBoost models “perform better overall” than other methods, demonstrating its use in player replacement. The features used were a combination of FIFA19 stats, almost identical to the stats above, and match events. Match events refer to events such as passes, shots, yellow cards or substitutions. They only use passes to create the pass graphs, and they use the substitution data as ground truth.

Overall, these papers show that XGBoost is a promising candidate for our problem of predicting player performance when transitioning to a new team. The authors also emphasise the importance of data preprocessing and parameter tuning for maximum efficiency. The FIFA datasets provided a more simplified approach, as the data is not based on direct measures of performance such as goals scored, but they instead use stats such as shot power or finishing. It is also important to note the significance of the ‘potential’ stat as this indicates that a model predicting values cannot simply rely on historical stats but must have some way to take in probability to improve.

2.3 Project Statement

This project will aim to develop an XGBoost regression model to predict a ‘true market value’ for a football player. This refers to the value of a player independent of their teammates. By finding a ‘true value’ we can identify players who perform well irrespective of their environment meaning teams get a better idea of the expected performance of a player that they want to acquire. Then, we can develop a second XGBoost model which assesses past transfers to generate these player-club recommendations based on the needs of the club. This dual-model system is relatively similar in structure to the second paper reviewed above, as the first model assesses the players’ performance metrics and the second handles player replacement.

This project will face a significant challenge due to its reliance on existing predictions which inherently include teammate contribution bias. Since the project will attempt to mitigate the effect of teammate contribution, we will not be able to evaluate the performance of the model by directly comparing our predictions to existing ones.

There are three ways in which we can evaluate the model:

We can perform k-fold validation, where we check the model's performance on multiple folds of the training data, by averaging multiple performance metrics, such as Mean Squared Error and Mean Absolute Error.

Although the above tests that the model is performing well within the confines of the training data, we are more interested in how it extrapolates to current data so we can provide up-to-date predictions. Therefore, we can also use our domain knowledge by inputting transfers which have occurred recently and evaluating our model's prediction in comparison to what is currently happening in football.

Finally, we can evaluate the interpretability of the model. We can use the feature importance scores which are included as part of the XGBoost package to ensure that our model is making rational decisions. This can also help us identify redundant features.

3 Data

This section will attempt to outline the datasets used, as well as the cleaning methods applied, identify and remedy problems with the data and provide a methodology for feature construction and selection. It will also validate the assumptions we made about teammate contribution affecting player valuations.

3.1 Overview of Datasets Used

This project will require three separate datasets which are as follows, and will be described in more detail below. All three datasets were acquired from the data science website Kaggle,[16] but the original sources are indicated where available.

- FBRef.com on-pitch statistics (goals, assists, tackles, passes etc.)[17]
- Transfermarkt.com market value prediction.[18]
- Transfer fee history.[19]

Note that as mentioned previously, it is currently easy to access reliable, detailed data, it is much more difficult however to access high-quality data further in the past, therefore although all of these datasets have data from earlier, this project will only use data from 2016 and after. In addition, since the intended use of the project is to predict values for current players, the data used as validation data will be the most recent data available, which is the set of samples from the 2021/2022 season.

3.2 FBRef Statstics

3.2.1 Summary

This dataset holds 213 different measures relating to on-pitch actions, as well as some player-specific data such as a player's position and whether they are left/right-footed. Each row describes one season (August-May) for one player, and there are 17199 rows in the dataset (not including pre-2016 data).

3.2.2 Exploratory Data Analysis

We have made several assumptions about the data, which have become the foundation of the problem, therefore it is important to perform some exploratory data analysis to see if these assumptions are fair.

1. Fees are increasing
2. Teams want to recruit players at a younger age to try and make profits
3. Teammates are a large deciding factor in the values of a player's performance metrics.

As shown in Figure 3.1, transfer fees have increased significantly over the last few years, and the peak has also shifted from age 24 to age 22, as teams as willing to spend on younger players for the chance to make big profits.

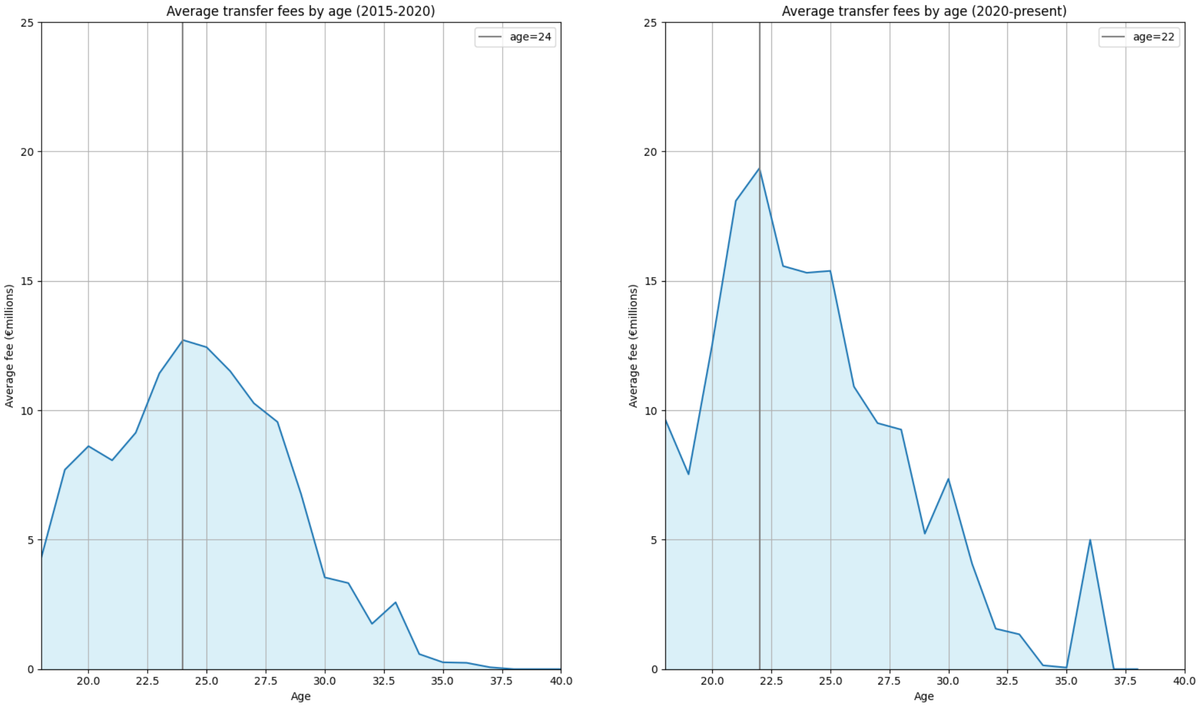


Figure 3.1: Transfer fee vs Age at time of transfer

One possible method we could use to show how teammates affect a player's metrics is to see how teammates affect the goalscoring ability of a striker. We can select the top goalscorers for each team, and calculate the Spearman's rank correlation co-effecient between both the number and the quality of chances they received, and how many goals they scored, which is calculated as shown below.

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)},$$

where d_i is the difference in rank between each observation and n is the number of observations (number of clubs in this case).

In Figure 3.2, goals are represented by G. The two metrics used to calculate chance quality and number of chances are called Expected Assists (xA) and Shot-Creating Actions (SCA). Expected Assists measures the chance any given pass results in an assist, so a pass which results in a certain goal would have 1 xA, and a pass which would never lead to a goal would have an xA of 0.[20] Shot-Creating Actions on the other hand measure the number of chances a player creates, and these are not limited to passes. Tackles, dribbles, shots or even fouls which lead to a shot are all counted under SCA.[21] We can take the sum of the xA and SCA of every player on the team, except for the main striker, to measure the quantity and quality of the chances received. The results shown below indicate a strong correlation between chance quality and goals, and a slightly less strong correlation between number of chances and goals. There is also a metric included called 'Goals minus Expected Goals' (G-xG). This the the number of goals scored subtracted by the number of goals an average player would have scored based on the shooting opportunities presented. A higher number means a striker consistently scores more goals than expected, which gives us a good indication of the striker's shot quality.

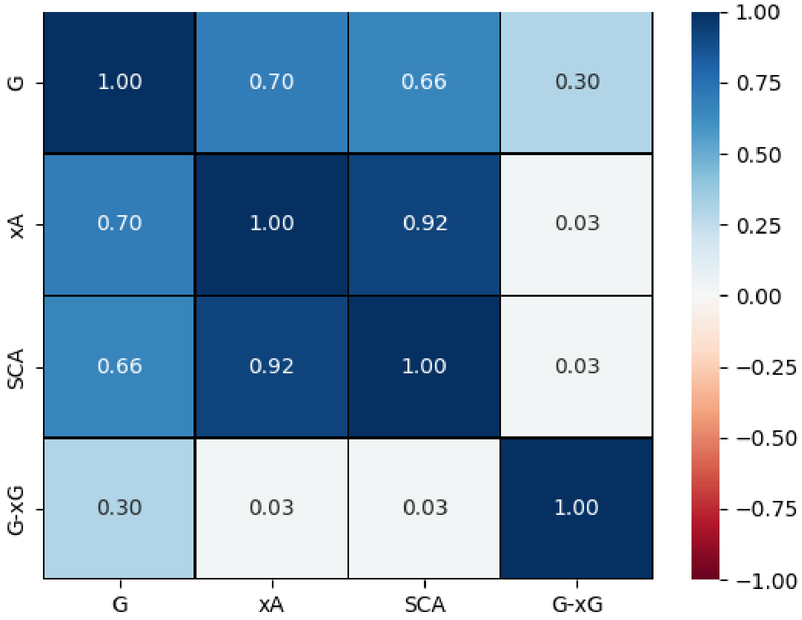


Figure 3.2: Correlation between goals and teammate contribution

Now it is clear that the assumptions made above were valid, as both metrics relating to teammate contribution towards the striker have a much stronger correlation to goals scorer than the goalscorer's own shooting ability. This means that when the predictive model is built, it should find a way to account for the fact that any player valuation which takes player statistics into account will be significantly affected by teammate contribution.

3.2.3 Position vs Role

There is an issue with this dataset with the way it describes a player's position. A player's *position* defines the area of the pitch they occupy and their main responsibilities. However, a player's *role* is more to do with the specific tasks that they may be required to do individually, meaning that multiple players can have the same position, but may have different roles. When comparing players and attempting to decide which players would fit well into a team, it would be more beneficial to be able to compare players in the same role, instead of the same position.

The issue is that the dataset holds every position that a player is capable of playing, but doesn't distinguish between a position they may have played once, and a position they typically play. For example, in the initial dataset, there are 148 unique position combinations present. These positions are formatted using four broad positions (GK, DF, MF, FW), which represent goalkeepers, defenders, midfielders and forwards. They also sometimes have more specific positions available for example 'MF (WM, left)', which specifies Left (Wide) Midfield as the position, but these are not always available, or sometimes multiple specific positions are used.

There are two possible ways to solve this problem, either attempt to determine a specific role for each player from the data available or use the four broad position categories provided.

Clustering

Using clustering, we can find complex relationships between players and get more detailed role information than what their on-paper position may dictate. Also, it may allow us to find players who could be utilised better in a different position. However, some similar projects mentioned later on have attempted clustering with a similar goal and have had some success, but these projects had access to spatiotemporal data for each match. Although player roles are dictated by more than just their location on the pitch, it is clear that any clustering attempt which has access to spatiotemporal data will likely be more successful than using just player stats alone.

There are two commonly used clustering algorithms that we could use, k-means clustering and hierarchical clustering. k-means clustering allows us to split our set of players without being constrained by pre-defined position labels. k-means is also flexible enough to handle our high-dimensional dataset and could allow us to adapt player positions towards modern football which often requires players to have a varied skill-set instead of relying on traditional rigid position labels.

Hierarchical clustering on the other hand allows us to understand the underlying structure which relates positions. Also, by building a dendrogram, we can visualise the similarities and differences between clusters, something which is often difficult when clustering with high dimensions. The dendrogram can also assist us in selecting effective cluster amounts through careful evaluation of the structure of the plot.

Results

Not all football teams play in the same *formation*, which means that each team may select a different number of defenders, midfielders, and attackers. This also means that a *role* which is present in one team may not be present in another team. Since a team consists of 11 players, this would mean we may expect to see at least 11 roles. However, our first attempt at clustering below indicated an optimal cluster amount of 6.

We determine the optimum cluster count by using the elbow method. This is where we find the point for which the rate of decrease of the Within Cluster Sum of Squares (WCSS) slows down the most. One drawback of the elbow method is that the result of some clustering attempts renders a graph shape as below with no clear elbow point. In this case, we can also look at the Silhouette Score which defines how well separated our clusters are. The Silhouette Score is calculated for each point and the Silhouette Score of a cluster is the mean of each point's score. In the clustering attempt in Figure 3.3, we have relatively low Silhouette Scores for all values of k. However, we would expect to find this sort of result as many player roles do not have a lot of difference when it comes to player stats and would correlate more with player location. In addition, players who can fit into multiple roles will have a negative effect on each cluster's score, and most professional players usually are able to play more than one position.

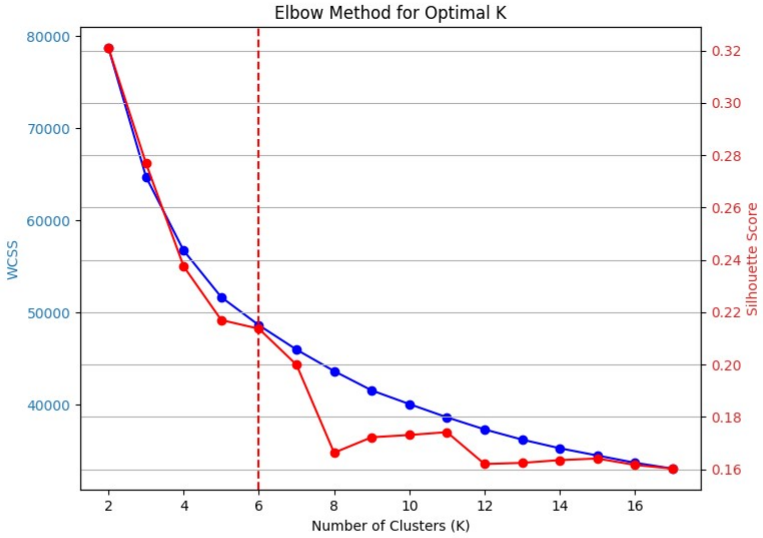


Figure 3.3: Results of k-means clustering

In the search for the optimal k value, we can turn to hierarchical clustering to see how clusters interact with each other. Goalkeepers have been removed from the dataset for the dendrogram below as they are easily classified and are so separated from the rest of the samples that they make the distinction between the other classes hard to interpret. Figure 3.4 shows that the samples can be split into 4-5 clusters before outliers begin to be separate clusters. If we add back in the goalkeepers we get the same 5-6 clusters as before.

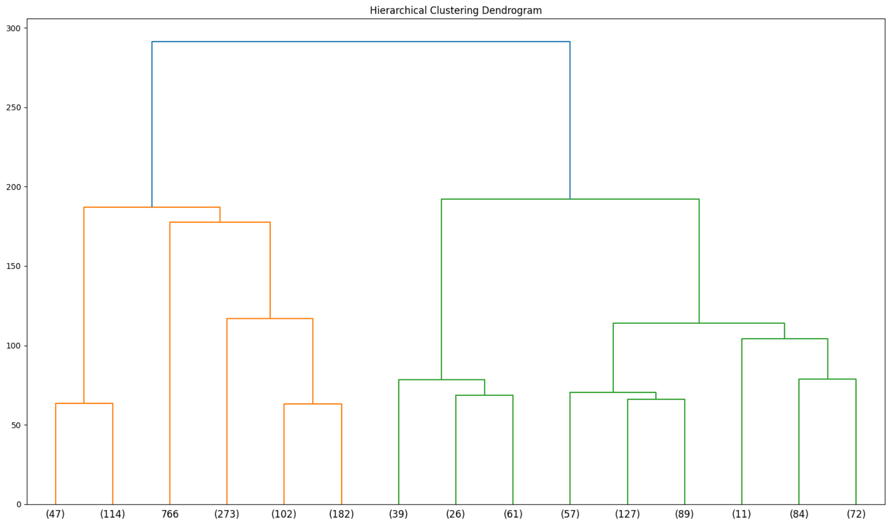
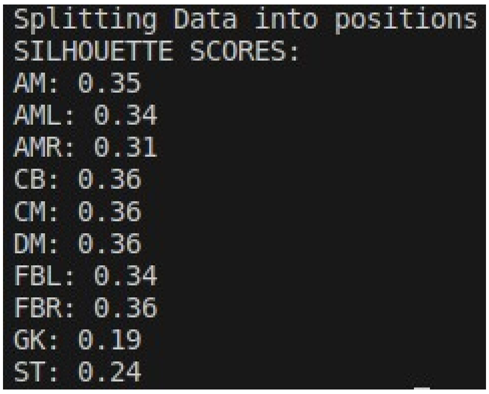


Figure 3.4: Results of hierarchical clustering

One additional method we can try is to split the dataset using the broad position values we have and then apply a clustering algorithm with a small k-value of between 2 and 4 to try and detect roles within positions. Splitting the dataset first would mean each subset should be more homogeneous, meaning each decision boundary for each cluster assignment can be more simple and interpretable, which would lead to much more useful role assignments. [1]

[1] Note the low score for goalkeepers is due to it trying to cluster a position into multiple roles where only one role would be expected



```
Splitting Data into positions
SILHOUETTE SCORES:
AM: 0.35
AML: 0.34
AMR: 0.31
CB: 0.36
CM: 0.36
DM: 0.36
FBL: 0.34
FBR: 0.36
GK: 0.19
ST: 0.24
```

Figure 3.5: Results of position-based k-means clustering

As seen in Figure 3.5, excluding goalkeepers we found clusters with silhouette scores in the interval $[0.24, 0.36]$. The low score for strikers would likely be for the same reasons as mentioned previously for the goalkeepers. Further investigation into the contents of each cluster found a wide variety of positions, and since the silhouette scores did not differ much from previous attempts, it seems that clustering for role detection will not lead to any further insights into our data.

Moving forward, the best indication towards role that we possess is the on-paper positions, but we can take some steps to make them more specific. For example, traditionally fullbacks (left and right backs), denoted by `FB` in the dataset play on the same side as their stronger foot. Using this, we can assign a more specific position `FBL/FBR` to players whose on-paper position may just state `FB`. While some classification error may result from this approach, the number of players without a specific position is relatively low, and the number of players misclassified is even lower. This was also the technique used in the above clustering attempt used to split the dataset.

3.3 Transfermarkt Valuations

3.3.1 Summary

This dataset holds data related to the transfer values of football players. It contains 31 columns, but with significant overlap with the previous FBRef dataset, so we only use the `id`, `name`, `date` and `market_value_in_eur` columns. There are 241032 market value entries, but only the ones corresponding to players in the FBRef dataset are used. While the FBRef data is structured as one row per player per season, this data has one row per player per valuation, where a player can have multiple valuations in a season.

3.3.2 Exploratory Data Analysis

We have shown there is a significant correlation between teammate contribution and traditional success metrics, but we need to determine if this is reflected in actual valuations. Using the same technique displayed in Figure 3.2, we can replace a striker's goals with their market value, and see similar results replicated in Figure 3.6. [2]

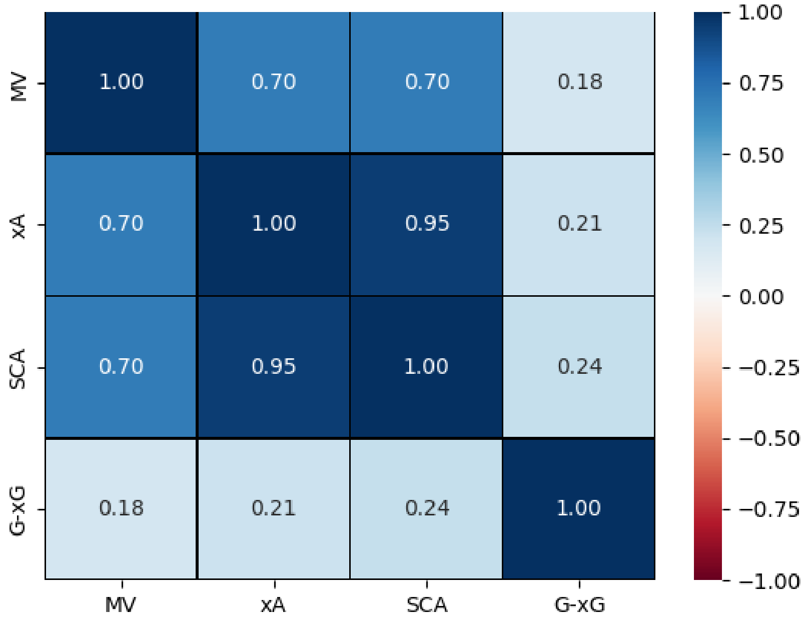


Figure 3.6: Correlation between Transfermarkt prediction and teammate contribution

The heatmap above shows very promising signs that our assumption of teammate involvement correlating strongly with player value is valid. We can use this in the

[2] Note the change in correlation between the unchanged metrics is due to the removal of rows with incorrect/missing market values

future stages to test which features we want to remove from our model due to their bias.

3.4 Transfer Fee History

3.4.1 Summary

This dataset holds historical data on past transfer fees, note the distinction between transfer *fees*, and transfer *values* is that transfer fees refer to the price a certain club paid in a transfer deal which happened in the past, and a transfer value is a prediction of how much a given player is worth in a free market. The dataset is a combination of files from 9 major leagues in Europe (the top 2 divisions in England, and the top divisions of Germany, France, Spain, Italy, Netherlands, Portugal, and Russia). From this dataset, we use the following columns:

- `fee_cleaned` - Numeric conversion of the `fee` column, converted to euros
- `transfer_movement` - Either 'in' or 'out', each transfer appears twice in the dataset, with one of each value.
- `name` - Player name, used to create a common key between datasets
- `season` - Season in which transfer occurred, used to create a common key

3.5 Data Cleaning

Before we create a model, we need to perform extensive data cleaning in order to have a reliable and high-performing model. We need to ensure the input data is consistently formatted and accurate. In order to create a high-quality feature set, the below operations we performed prior to developing a model:

Firstly, we had to merge the stats CSV files together, as the dataset was split between multiple files. As a result of the merge, we had to remove duplicate columns, fix column names as suffixes were added post-merge, and we also had to remove rows with low data to guard against outliers. Now we had to remove all rows which occurred prior to 2016, as every row before has missing entries. After attempting to remove NULLs as much as possible over the previous steps, we can now fill the remaining NULLs with zeros.

The next steps are more aimed towards converting our data into a form which can be entered into our model. First, we had to convert position values into readable form, which meant converting from a list of possible positions to one position value. We also had to split `comp_level` to distinguish between 1st and 2nd division, by introducing a new `comp_tier` feature. This meant that first in the second division would not be on equal footing with first in the top division. To continue accounting for the difference in divisions, we converted `lg_finish` to an Integer e.g. 1st → 1 and added around 20 places onto league finishes in 2nd divisions for example, 1st in the second division is equivalent to 21st in the country.

The final steps involve incorporating the other transfer related datasets into our stats dataset. We matched the date of a transfer to a season e.g. 1/1/2016 → 2015/16 season, which aided us in creating a common ID between stats and transfer value datasets, which we then used in order to merge transfer values and player stats tables. We then repeated some of the cleaning steps used after the previous merges, including removing duplicated columns and removing NULLs. We then had to convert the `season` feature to an Integer e.g. 2015/16 → 1, 2016/17 → 2. Finally, we can merge the transfer fee data with the rest of the data after removing player loans from the dataset.

3.6 Feature Engineering

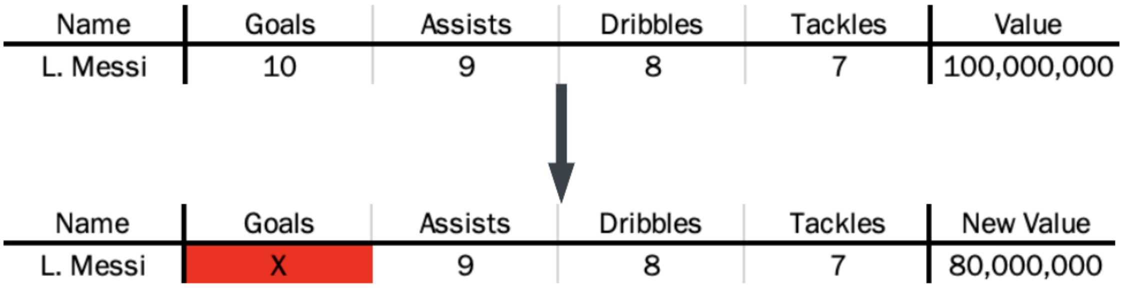
The purpose of the model is to remove the bias that we have found associated with features that depend heavily on teammates. Therefore, we need to craft a set of features which only consist of metrics which measure intrinsic player attributes while minimising the effect of teammate ability. There are three categories which we can sort the metrics into:

- Metrics which require teammate input: e.g. Goals require chances to be created by teammates

- Metrics which require teammate output: e.g. Assists require chances to be finished by a teammate

- Metrics which require no teammate contribution: e.g. Tackles require no input or output from teammates

Now it's important to note that although the third category of metrics is ideally what we want our feature set to include, most metrics will be in one or both of the first two categories, in which case we need to find the ones which minimise teammate contribution. Earlier in Figure 3.2, we showed that goals are more strongly correlated with metrics we chose to highlight teammate contribution (xA, SCA), than with a metric we chose to capture the shooting ability of the player (G-xG). Now consider Figure 3.7 below, if, for the sake of example we were to find that goals were too strongly correlated with teammate ability to be used in the model, we could remove the feature and train a new model, and get a new prediction with reduced 'teammate bias'. [3]



Name	Goals	Assists	Dribbles	Tackles	Value
L. Messi	10	9	8	7	100,000,000

Name	Goals	Assists	Dribbles	Tackles	New Value
L. Messi	X	9	8	7	80,000,000

Figure 3.7: Removing biased feature example

This is an effective way to remove teammate bias, however, we can lose important performance metrics in this way, after all, goals are a vital part of football. A better way to mitigate any bias is to normalise our features to account for the bias. One way we can do this is to split up our feature set and categorise each feature depending on what part of the game it relates to. Conveniently, this is already done for us, as the original form of the stats dataset is divided into different categories, just as the data is arranged on the FBRef website.

[3] Note the data in the example is for illustrative purposes only

Metric Type	Adjust for
All	Minutes played
Defensive	Opposition possession
Goalkeeping	Post-Shot xG faced
Goalscoring	Sum of teammate xA
Chance Creation	Sum of teammate Goals - xG
Shots	Sum of teammate SCA

Table 3.1: Metric Normalisation

For each category, we can define a metric which represents teammate contribution, as we did in the Exploratory Analysis section. Table 3.1 indicates the metric which requires normalisation and the metric used to account for teammate contribution. Data on goalkeepers and defenders is normalised by chances faced and opposition possession respectively, which indirectly accounts for teammate contribution as it accounts for the negative contribution of teammates. The metric 'Post-Shot xG faced' refers to the sum of the probabilities that the goalkeeper saves each shot. We also have to normalise all metrics by minutes played, and although some metrics have this built-in, for example goals per 90 minutes, it is easier to just remove these metrics and normalise the others as a group. This method will work well for the majority of features, however, there are some features which just inherently measure teammate contribution and must be removed. For example, consider the metric `plus_minus`, which is simply the goals scored by the team minus the goals scored by the opposition, irrespective if the player had any involvement at all. This process has allowed us to reduce the feature count from 213 to 166, or 144 not including stats which are only relevant to goalkeepers. This number will decrease further as redundant features are identified as part of model validation.

4 Model Selection and Implementation

We have identified some issues with the Transfermarkt.com predictions, yet we don't have access the the wide variety of data and expert opinions that are included in these predictions. Therefore, it would be easier to leverage the power of machine learning algorithms to adapt the existing data to become more objective and robust, than to create our own predictions. This would allow us to only select the features we believe to be intrinsic to a player's performance, or to add features which we don't think are accounted for in the existing predictions.

The result of the bias towards higher-performing teams will mean that certain players will be overvalued and certain players will be undervalued compared to the mean. However, we can also realise that over our entire diverse set of samples, the individual overvaluations and undervaluations may balance each other out. Using this, we can create a model where we only introduce features which we believe to be unaffected by teammate contribution, and in the training process, we can make our predictions fit the original distribution of the Transfermarkt predictions. In this way, we can still utilise elements of the Transfermarkt data which we do not have access to, we can use the distribution to inform our training process, but the output we receive should provide a more objective valuation.

4.1 eXtreme Gradient Boosting (XGBoost)

4.1.1 Generalised Algorithm

The core purpose of gradient boosting is to build an ensemble of weak learners and then combine them to create a strong predictive model. The algorithm starts with a simple prediction and then each additional model improves the prediction. At each iteration, the next model is trained to predict the difference between the ground truth and the current ensemble's prediction (often referred to as residuals). In particular, we will be focusing on XGBoost Regression, which uses a form of decision tree called regression trees as the weak learners.

XGBoost improves on traditional gradient boosting techniques by introducing a regularisation term, as well as several enhancements to its efficiency including the ability to leverage multiple CPU cores.

At its core, XGBoost attempts to optimise the objective function below, consisting of a loss function and a regularisation term.

$$\text{obj}(\theta) = \sum_{i}^{n} l(y_i, \hat{y_i}) + \sum_{k=1}^{K} \omega(f_k)$$

,

where n is the number of training examples, $l(y_i, \hat{y_i})$ is the loss function, K is the number of trees and $\omega(f_k)$ is the complexity score of each tree. We combine the trees to create our final prediction $\hat{y_i}$ as shown below

$$\hat{y_i} = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F}$$

where $\mathcal{F}$ is the set of all possible trees. Then at each iteration t we can calculate the first and second partial derivatives of the loss functions with respect to the current prediction $\hat{y}_i^{(t-1)}$.

$$g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$
$$h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

These gradients are used to construct the next tree, which is then added to the ensemble aiming to minimise the overall objective function.

The regularisation term $\omega(f)$ can be implemented in different ways, the implementation shown below includes L1 (Lasso) and L2 (Ridge) regularisation.

$$\omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 + \alpha \sum_{j=1}^{T} |w_j|$$

where T is the number of leaves in the tree, γ, λ and α are hyper-parameters that control the degree of regularisation and w_j is the score of the j-th leaf. The regularisation term helps to control the complexity of the trees and prevent overfitting.

4.1.2 Python Implementation

We can use the `scikit-learn` implementation of XGBoost in Python to simplify the implementation. There is an independent XGBoost library but the `scikit-learn` implementation offers several benefits not found in the independent version. The main benefit is that using the API allows us to also seamlessly integrate `scikit-learn`'s model selection and parameter tuning tools, like the built-in `GridSearchCV` function for example.

There are many tunable parameters we can alter to affect the model's predictive ability, the main ones are summarised below.

- `max_depth` - Set the maximum depth of each tree
- `n_estimators` - Set the number of trees
- `learning_rate` - Set the contribution of each tree
- `reg_alpha` - L1 regularisation parameter
- `reg_lambda` - L2 regularisation parameter
- `gamma` - Set the minimum reduction in loss to make a split
- `subsample` - Set a fraction of the training data to be used by each tree
- `colsample_bytree` - Set a fraction of the features to be used by each tree

4.2 Value Prediction Model

4.2.1 Total Variation Distance

Our model needs a way to incorporate the wide variety of factors which Transfermarkt.com predictions use such as community knowledge, statistical metrics and contract details which we don't have access to. We need to use this information as a guideline, without letting it bias our predictions. One way we could do this is mentioned previously, which is to use the distribution of valuations instead of the valuations themselves. We can accomplish this using Total Variation Distance (TVD).

TVD measures the distance between two distributions. However, in our case. we don't have distributions as inputs, we have lists of predictions. We can convert these into distributions using histograms. Suppose we take a list, and split it into 50 bins. Assume Bin 1 holds all of the predictions which are less than €1 million, Bin 2 holds €1-5 million and Bin n holds all predictions above €100 million. Then we can calculate the probability that a prediction would land in each bin, resulting in a distribution. Therefore, the formula for TVD for our use case is

$$\text{TVD} = \frac{1}{2}\sum_{i=1}^{n} |p_i - q_i|$$

where p_i, q_i are the probabilities of the i-th bin, and n is the number of bins. The number of bins, and where we place the bin edges will be important in deciding how accurate our results will be. If we select too few bins, we may struggle to fit the underlying patterns in our data, However, if we use too many bins, it may lead to overfitting, and could also introduce noise as a result of some bins being empty. The bins we select should also be distributed according to our goals. For example, the most important bins will be on the right of the distribution, as this will hold the more expensive players, so it is important to have enough bins to achieve the necessary granularity in this part of the distribution.

4.2.2 Objective Function

One of the most important parameters for our model is the objective function, this is the function which gets minimised throughout the training process, and it also defines the goal the the model attempts to achieve. Since TVD measures the distance between our distributions and not individual data points, we can't use it as an objective function, so we have to select a different function and since our problem is a regression problem, we can select a function from the built-in regression objective functions. Many objective functions make assumptions about how our data is distributed, so it is important for us to know what sort of distribution we are aiming for, the `reg:squarederror` function for example assumes that the errors are normally distributed, so would be suitable for problems where the target variable is normally distributed.

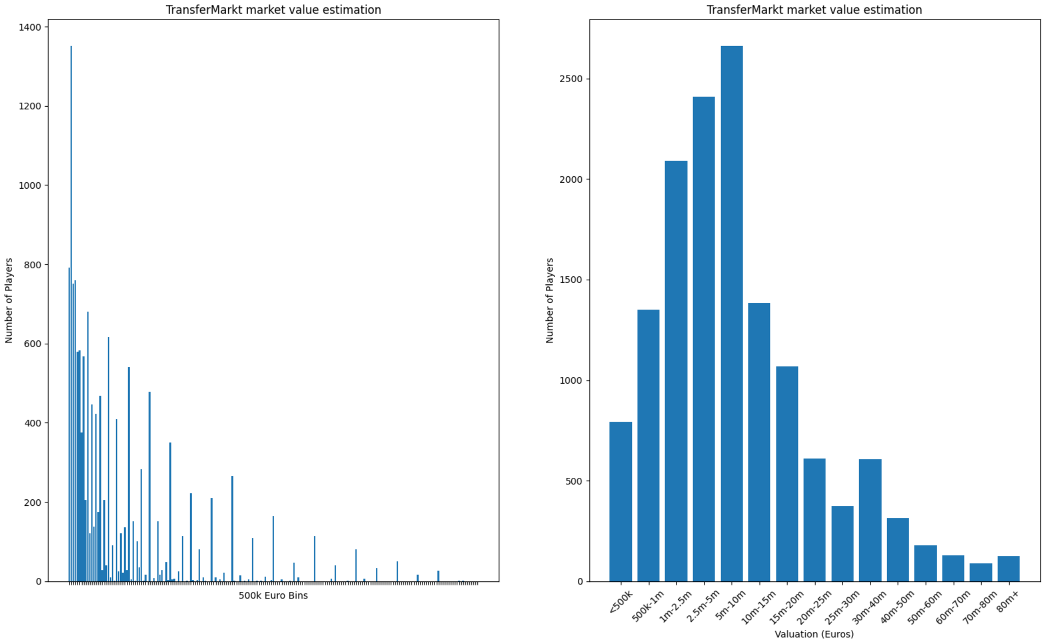


Figure 4.1: Existing Transfer Value Distribution

Figure 4.1 above shows our distribution of values. We can clearly see the values are skewed and do not follow a normal distribution. The data also has peaks around round numbers, there are clear spikes at multiples of 5-10m, especially as price increases. The two main objective functions which are best for skewed distributions are the `reg:gamma` function and the `reg:tweedie` function. As mentioned previously, the `reg:gamma` function was used in one of the reviewed papers which this project was inspired by. The `reg:gamma` function assumes the errors follow a Gamma distribution, which is commonly used for different price prediction models due to the skewed nature of the distribution. The `reg:tweedie` function on the other hand is also adept at handling skewed data. This function assumes that the errors follow the Tweedie distribution family which actually includes the Gamma distribution as a special case. Therefore, it seems that the `reg:tweedie` is the better choice due to the increased flexibility offered. Using this function allows the model to adapt to a wider range of distributions in the Tweedie family without being limited by the conditions of the Gamma distribution, while still being able to capture the features of a Gamma distribution if required. The paper which selected `reg:gamma` as an objective function had a similar purpose to this project. They selected the function using a simple grid search, while attempting to minimise `squaredlogerror`. Since we want to minimise TVD we can also use a grid search to help us, and this results in a TVD of 0.15 for `reg:tweedie` and 0.22 for `reg:gamma`, confirming the success of our chosen `reg:tweedie` function.

Tweedie and Gamma Distributions

Tweedie distributions refer to a family of distributions encompassing gamma, normal and Poisson distributions in specific cases. The PDF of a Tweedie distribution is defined as:

$$f(y; p, \mu, \phi) = \frac{y^{p-1} e^{-\frac{y\mu^{1-p}}{\phi}}}{Z(p, \mu, \phi)}$$

where:

- y is the random variable representing the outcome
- p is the power parameter which indicates the type of distribution. $p = 0$ for Normal, $p = 1$ for Poisson, $p = 2$ for Gamma. This parameter is also an additional tunable parameter for XGBoost called `tweedie_variance_power`, which is specific to the `reg:tweedie` objective function.
- μ represents the mean of the distribution
- ϕ is the dispersion parameter and controls the variance of the distribution
- $Z(p, \mu, \phi)$ is a constant to ensure the integral of the PDF sums to 1.

Careful selection of the power parameter p allows the distribution to be symmetric, right-skewed or left-skewed. It can also allow the distribution to be zero-inflated, which is useful for modelling price data, however, in our case it is not required.

In the case where $p = 2$, we get a Gamma distribution, which can simplified to have the following PDF definition:

$$f(y; 2, \mu, \phi) = \frac{ye^{-\frac{y\mu^{-1}}{\phi}}}{\phi\mu}$$

4.2.3 Parameter Tuning

Total variation distance is an important metric for our model, however eliminating what I've called 'teammate contribution bias' above using this metric is just one requirement of this model. We can use total variation distance to attempt to get our model's predictions to fit the distribution of the existing predictions we have in our dataset, and this would be relatively simple to implement. However, we also have to make sure that the resulting predictions are accurate.

Therefore, the problem is in essence finding a balance between two competing factors. Firstly, we need to fit our predictions to the distribution of market values in our dataset, since although the predictions we have access to may be flawed, they take into account data and insights which aren't widely available. Secondly, we need to predict market values, using an unbiased set of features with the goal of minimising our chosen loss function, since otherwise we would just end up with a well-arranged set of meaningless predictions.

Firstly, we ran a grid search over the set of the below parameters, with the resulting best parameters highlighted in bold. Note that `n_estimators` is capped at 1000 as any further increase results in significant increases in running time.

- `max_depth`: [1,**3**,5,7]
- `learning_rate`: [0.01, **0.05**, 0.1, 0.2]
- `n_estimators`: [**1000**]
- `min_child_weight`: [**1**]
- `gamma`: [**0**]

- `tweedie_variance_power`: [1.05, **1.5**, 1.95]

Then we can run a more specific grid search with the parameters below in order to more finely tune the parameters and to tune the parameters left as constants previously. However, this search did not yield a better result than the previous run, so we will keep our parameters unchanged from above.

- `max_depth`: [2,3,4]
- `learning_rate`: [0.025, 0.05, 0.075]
- `n_estimators`: [1000]
- `min_child_weight`: [1, 2]
- `gamma`: [0, 0.01]
- `tweedie_variance_power`: [1.25, 1.5]

Figure 4.2 displays the Mean Squared Error plotted against the Total Variation Distance for each set of parameters in our initial grid search, as this visualises our trade-off between prediction accuracy and correct distribution. The point which represents the optimal set of parameters is in red.

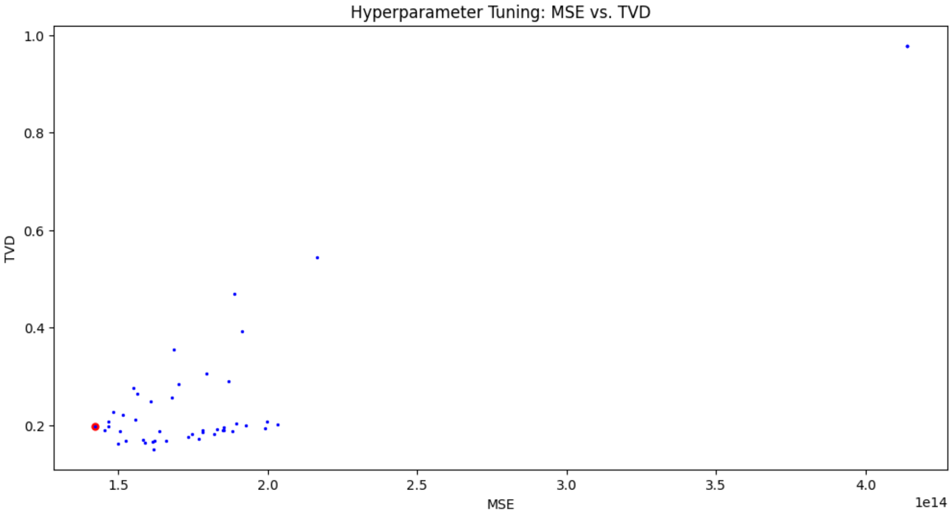


Figure 4.2: Tradeoff between TVD and MSE

Although optimising the model parameters above using a grid search seemed to lead to promising results, the Total Variation Distance results can be misleading. Figure 4.3 shows the two distributions side-by-side using seemingly optimal parameters. Visual inspection reveals that the model fails to capture the long tail shape of the distribution towards the right. Since this tail refers to the highest-value players, it is vital that our model can capture these intricacies.

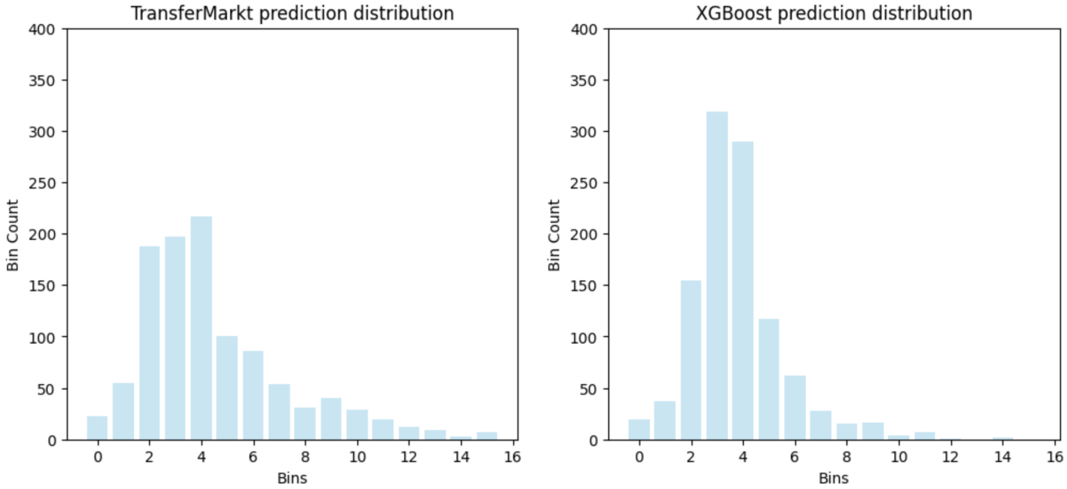


Figure 4.3: Prediction distribution using optimal parameters

We can adjust the parameters to reduce our TVD even further, however this makes the actual predictions themselves become highly unstable. The next problem this project needs to address is the personalisation aspect. We now have some more objective valuations than the original dataset, but we now have to be able to recommend players to each club.

4.2.4 Next Steps

In order to build a model which takes our generated market values and tailors them to specific clubs, we need to define what makes a player right for a club. In essence, we want to predict which players will be successful at which club. But we need a way to measure success to predict success in the future.

We can start by defining success using a simple metric such as the change in a player's market value over time at the club. By doing this we can tie market value to performance, which seems on the surface to be a sensible observation. However, player performance is not the only factor when considering a player for transfer. For some clubs, in their current state, the transfer policy may focus more on profit than purely on performance. Transfer policies adopted by individual clubs will be inherently complex, based on the club's objectives, current performance and financial standing. Therefore, we have to attempt to create some general rules, which will broadly apply to most of the clubs across the world.

Clubs towards the top of their respective league are fighting for trophies, qualification to European competitions or promotion to a higher division. All of these achievements come with significant financial rewards, which enables these clubs to focus their transfer strategy on acquiring players who will perform well in the short term to meet these objectives, which will allow them to reap financial rewards in the future.

At the opposite end, clubs towards the bottom of a league can also focus their transfer policy on high-performing players, since they need to attempt to avoid relegation, to hold onto the financial benefits which come with staying in a higher division.

However, in between these two groups, are teams occupying the middle ground. Those which are safe from relegation, but are also still a distance from the top of the table. These are the clubs which can aim their transfer policies more towards generating profit through player sales instead. Since there isn't much gap between 8th and 15th, the rewards which come with a slightly higher league finish may not be worth investing in well-established players when they could instead use their matches to develop younger players, which they can then sell on for massive profits.

We can conclude that any model which makes recommendations on players to be transferred should take a holistic approach, it should consider a transfer as not only a function of a player's current and projected on-pitch performance but as an investment opportunity as well. In this way, we can make intelligent recommendations which would also be realistic for clubs to consider. We can do this by creating an ensemble model, where one model 'fixes' the market value as above, and the other uses that value to personalise the value to a team.

4.3 Value Personalisation Model

Before we attempt to personalise transfer values, it is necessary to clearly define what we mean by the personalisation of a market value. This refers to the generation of a true market value, as completed above, which is then weighted depending on the needs of the given club. Traditionally, this would focus on how a player would fit into a team's formation, and how they would interact with the new team. However, since formation and positioning data is scarcely available, this section will instead focus more on how a player can help their team reach their financial objectives, and while this does include performing well (and therefore helping to win prize money), it will also focus on if a player purchase will end up being profitable.

4.3.1 Defining Success

In order to predict a personalised market value, we need some way of quantifying what success would look like at a club. The most obvious choice would be to look at historical transfers to a club and score them based on success. Previously, we were referring to a transfer as the acquisition of a player by a club. However, from now we will be defining a transfer as the combination of the acquisition of a player by a club, their entire career at that club, and their eventual departure. This allows us to define transfer success as a combination of three components, with each club prioritising different components.:

- How good of a deal the club gets when they buy the player (Market Value In - Fee In)
- How well the player performs during their time at the club (Market Value Out - Market Value In)
- How much profit does the club make when selling the player (Fee Out - Fee In)

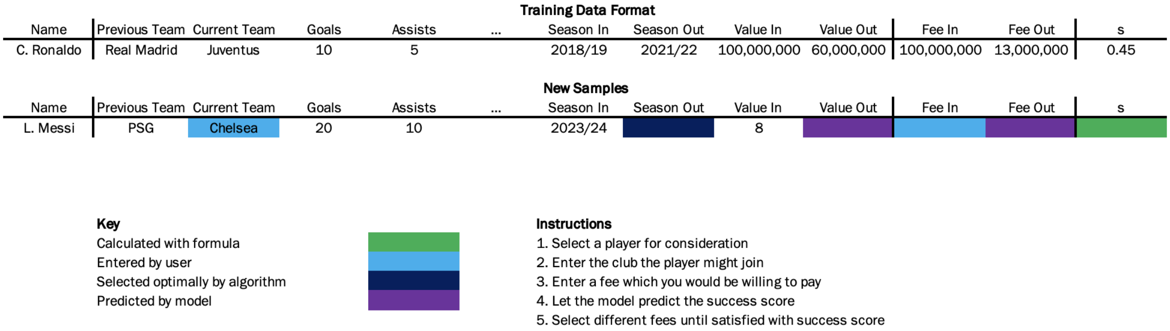
In order to be able to implement a personalisation system, we have to heavily simplify the second point above. The change in market value doesn't directly measure player performance, but it does provide a useful starting point for our model. The main issue with this approach is that for older players, market value will almost always decrease irrespective of performance. We can use a similar model as our previous one to complete our solution, except now we aren't constrained by TVD. But we do need to find a new metric to use to define transfer success. The metric has to be able to account for both the performance and financial aspects of the transfer. To start, we can use a weighted sum of the three parts of a transfer, and as mentioned previously, we can adjust the weightings for each club.

Changing the definition of a transfer means that we will also have to change the format of our data for the new model. Whereas previously our dataset was in the form [`id`, `name`, `season`, `goals`, ..., `market_value`] we now need to convert our data into the form [`id`, `name`, `season_in`, `season_out`, `goals`, ..., `fee_in`, `fee_out`]. This requires us to find a method of aggregating our data, since most transfers span multiple seasons, and our data is currently formatted on a per-season basis. We can start by using the mean, but ideally we will move towards using a method which has a higher weighting towards more recent performances. In addition, we have to add a column for our new success metric, which is called s.

$$s = s_1(v_i - f_i) + s_2(v_o - v_i) + s_3(f_o - f_i)$$

where v_i and v_o, represent the player's valuation when they enter and leave the club respectively, f_i and f_o represent the player's fee when they enter and leave the club. s_1, s_2, s_3 are the weightings set for the three parts of a transfer.

As mentioned above, we have to rearrange our data as we move from a season-based approach (One row per player per season) to a transfer-based approach (One row per player per transfer).



Training Data Format

Name	Previous Team	Current Team	Goals	Assists	...	Season In	Season Out	Value In	Value Out	Fee In	Fee Out	s
C. Ronaldo	Real Madrid	Juventus	10	5		2018/19	2021/22	100,000,000	60,000,000	100,000,000	13,000,000	0.45

New Samples

Name	Previous Team	Current Team	Goals	Assists	...	Season In	Season Out	Value In	Value Out	Fee In	Fee Out	s
L. Messi	PSG	Chelsea	20	10		2023/24	⬛	8	🟪	🟦	🟪	🟩

Key	
Calculated with formula	🟩
Entered by user	🟦
Selected optimally by algorithm	⬛
Predicted by model	🟪

Instructions

1. Select a player for consideration
2. Enter the club the player might join
3. Enter a fee which you would be willing to pay
4. Let the model predict the success score
5. Select different fees until satisfied with success score

Figure 4.4: Format of samples for success prediction

However, we run into a problem if we attempt to predict s with our model. In order to calculate s, we would have to manually set the values of our parameters $s1$, $s2$, $s3$ for each club in the training set, and since each club would have had different objectives at different times this is an almost impossible task.

Since we can't calculate s before running the model, we can simply do it afterwards. We do this by using a `MultiOutputRegressor`, which fits one regressor per target variable. We can then use this to predict the three terms of our equation individually, and then combine them after the model has finished running, which allows us to still be able to use our equation, but means we avoid the problem of having to manually assign the parameter values $[s1, s2, s3]$ for each different team in our dataset. Figure 4.4 is an example of how we can rearrange our dataset to include the transfer fee data. We could attempt to train a regressor for each of the three parts of the equation, however aside from the three weightings, the three parts of the equation are comprised of 4 terms in total, and as shown in Table 4.1, we already know two of them, so we can speed up the process by only using two regressors and directly predicting the fee and market value when the player leaves the club.

Term	Source
Fee In	User-Selected
Fee Out	New Model
Market Value In	Previous Model
Market Value Out	New Model

Table 4.1: Equation Term Sources

We can set up an XGBoost Regressor in the same way as before and pass it into the `MultiOutputRegressor` which creates multiple predictors, one for each of the labels we want to predict. Then, we can use a grid search in the same way as before to tune the parameters, which is much simpler in this case as we only have to reduce one loss function. We start with the same parameters as before, except we need to pick a new objective function since Tweedie regression is not supported by the `MultiOutputRegressor` yet. A simple search across objective functions reveals "reg:absoluteerror" as the best objective function for model performance.

The problem with the approach outlined above, is that we have predicted our variable s for each player, however, we wanted to predict their personalised valuation. And while s gives us an indication of whether the transfer would be successful or not, it is not a valuation. One way to conceptualise this is that if a player was bought for €1 billion, it would have a large negative effect on s, while intuitively, the price which a player is purchased for should not have an effect on their value.

There are a few approaches we can take to convert s into a useful metric for transfer valuations:

- Attempting to map the value s to the same distribution as our training data market values, which would convert it from an arbitrary value into a sensible valuation.
- Map s into an interval [0, 100] and model it as a “success chance” instead of a market value.
- Use a combined approach where we directly show the user the outputs (Fee Out and Market Value Out) of our model, as well as one or both of the

methods above, to allow the user more information to make decisions

Each of these approaches has its benefits and drawbacks. The first approach gets us the closest to achieving our original goal of a personalised market value, but it may be difficult to be able to accurately convert from s to a market value while keeping our commitment to transparency in scoring. The main drawback of the second method is that we have no way of knowing how transfer success probabilities are distributed, so we still wouldn't be able to generate a statistically sound metric, but since s is affected by purchase price a success chance would be a more accurate term to use. The third approach could provide more useful information, however this is at the expense of simplicity of interpretation. It also takes the requirement of the project to provide a valuation and moves the responsibility towards the user instead. After careful consideration, the first approach should allow us to achieve our goals in a way that is transparent if we are careful in how we manipulate our score s into a valuation estimate. The main reason for this approach is that it is the only way where we can continue our progress towards the goal of the project while balancing the need for simplicity and speed of implementation.

We can consider the distribution of market valuations in our training data and scale s accordingly. This will allow us to both use what we believe to be a good metric for transfer success and also generate valuations which are closely tied to market values. One issue with this approach is that the we will effectively be working with a distribution which encompasses transfers from 2016-2021, and so when we make predictions on current transfers, we will need to find a way to account for market inflation. Since this is out of scope for the project, a constant of 5% inflation is used to account for this.

5 Model Evaluation

Data preprocessing, parameter tuning and rigorous implementation are all vital steps in developing a good model and should lead to us achieving good scores in performance metrics, which in turn should lead to our model providing consistent, high-quality valuations and predictions. However, if our model remains opaque and difficult to understand it isn't enough. We also would like a way to be able to see how a model arrived at its decisions. Without interpretability, we will struggle to see the reliability or limitations of our model. One advantage of XGBoost is that it provides a built-in list of feature importance scores. We can easily see how each feature contributes to the final prediction of the model, which is not just useful for debugging and optimisation, but is also a useful feature for the end user as it could allows scouts or managers to look at different metrics they may not have considered before.

5.1 Value Prediction Model

There are three types of validation which need to be applied to our first model to ensure that we get useful and insightful results. Firstly, we need to ensure that the ordering of the players is correct, which means that the high-performing players we would expect to rank highly in our results appear as intended. Secondly, the transfer values of these players must be realistic. For example, the top ranked samples in our dataset range from €80 million to €200 million, so we would expect to this reflected in our predictions. Finally, we need to verify that the way our model is calculating our predictions make sense, asking questions such as 'Do the most important features make sense?' and 'Do we have redundant features?'

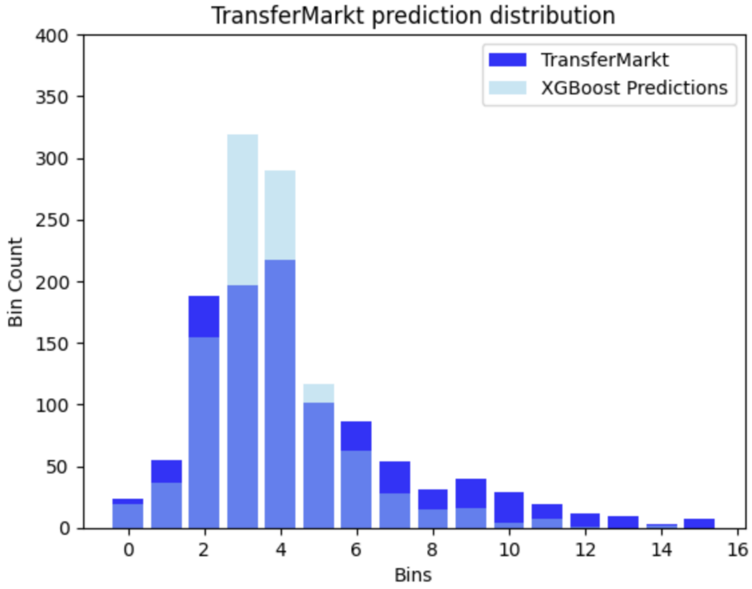


Figure 5.1: Overlay of predicted vs actual distribution

When developing this model, we tuned the parameters attempting to balance our trade-off between MSE and TVD. Shown above in Figure 5.1 are the two parts of Figure 4.3 overlayed on top of each other. We are trying to balance the trade-off between two metrics so naturally we wouldn't expect to perfectly mirror the original distribution, however, the difference between the two distributions, especially in the right tail where the high-performing players lie, is too high. In order to remedy this, we can tune the parameters again, using only the total variation distance between the bins in the tail, which should prioritise keeping the shape of the tail.

Unfortunately, this attempt was unsuccessful, as the parameters were unchanged between the two methods. The explanation for this behaviour is most likely a sample size issue. Since our validation set is around 20% of the size of the training set, when we have zero samples in the higher bins, it is actually only one or two samples away from the correct amount, and perhaps if we had access to more data, we would develop a stronger right tail shape.

Continuing to the second method of evaluation, we need to evaluate the order of the players shown in the results. To ensure that our predictions are reliable, it is necessary to tie in model evaluation with the real world. The most reputable award in football for individual performance is the Ballon D'Or.[22] It is awarded to the best-performing player in a season. The validation data which used for the project is all of the samples from the 2021-2022 season.

Team	Ballon D'Or	Age	Rank	Age	XGBoost	Team
Real Madrid	Karim Benzema	34	1	29	Mohamed Salah	Liverpool
Liverpool	Sadio Mané	30	2	23	Kylian Mbappé	Paris Saint Germain
Manchester City	Kevin De Bruyne	30	3	27	Aymeric Laporte	Manchester City
Bayern Munich	Robert Lewandowski	33	4	25	Rodri	Manchester City
Liverpool	Mohamed Salah	29	5	27	Memphis Depay	Barcelona
Paris Saint Germain	Kylian Mbappé	23	6	21	Erling Haaland	Borussia Dortmund
Real Madrid	Thibaut Courtois	30	7	30	Riyad Mahrez	Manchester City
Real Madrid	Vinícius Júnior	21	8	24	Rúben Dias	Manchester City
Real Madrid	Luka Modrić	36	9	28	Antonio Rüdiger	Chelsea
Borussia Dortmund	Erling Haaland	21	10	30	Ciro Immobile	Lazio

Table 5.1: Comparison of Ballon D'Or rankings and XGBoost's rankings

Consider Table 5.1, which shows the ranking of the 2022 Ballon D'Or beside the XGBoost results. The Ballon D'Or list on the left measures performance in the 2021/22 season alone, taking into account not just stats, but trophies won with the team whereas the right list aims to predict market values based on a player's 2021/22 stats with the impact of teammates removed. Therefore, we are not expecting to see the same list, but we can compare the two to gain valuable insights. Note that the Champions League, the most prestigious competition in Europe, was won by Real Madrid in 2022, who defeated Liverpool.[23]

Firstly, notice the difference in the ages shown in the lists. Our predictions have 8 players under the age of 30, while the Ballon D'Or rankings only have 3. This is to be expected, as the Ballon D'Or is measuring performance in 2022, and our predictions are attempting to measure market value in 2022. Therefore, even if our model thinks a player performed well, if they are above 30 they will receive a significant penalty to their value. Also, notice the Manchester City and Real Madrid players in the table. The Ballon D'Or prefers Real Madrid players, likely due to the fact they won the Champions League Final, however, based on performance, the model seems to prefer Manchester City players. Also, note the lone Manchester City player on the left is not present on the right. This would indicate that the model believes his stats were in fact boosted by the good performances of his teammates. It is also interesting to see that the XGBoost predictions have more players without teammates on the list, with one player from each of Liverpool, Paris Saint Germain, Barcelona, Borussia Dortmund, Chelsea and Lazio. This is a positive result which indicates the model is identifying players who are performing exceptionally, even when their teams are not. In particular, note how the XGBoost model moves Erling Haaland up 4 places, which would be expected as he has no other teammates within the Ballon D'Or top 30, again prioritising individual success.

Finally, using the feature importance scores, we can evaluate our highest-performing samples, and see which features they scored highly in. Note that it does not tell us if an important feature makes a positive or negative contribution.

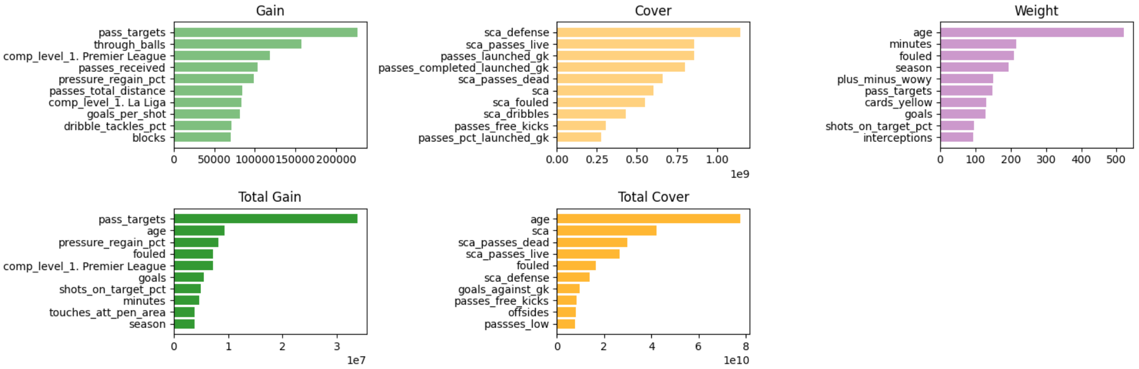
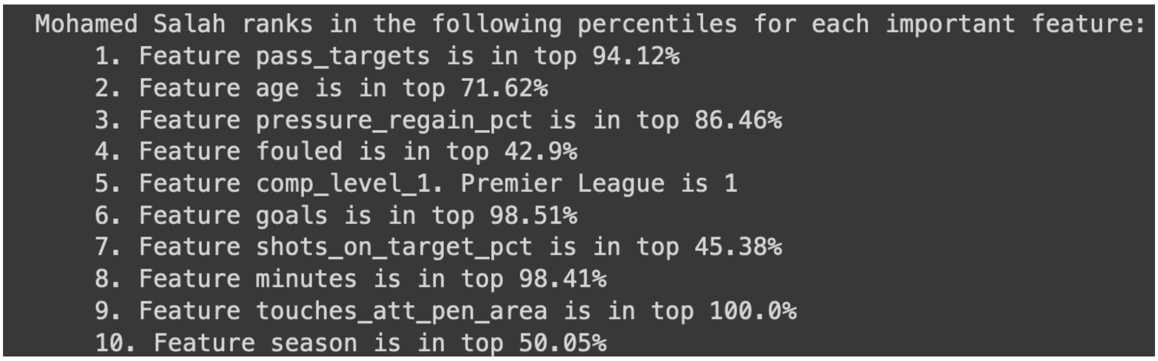


Figure 5.2: Different feature importance types

Figure 5.2 shows the different metrics we can use to measure feature importance. Gain measures the average improvement in the loss function created by splitting the tree based on a given feature. The only information in the documentation about the Cover metric is that it is a function of the second derivative of the loss function with respect to the feature, so a high score means the feature is highly important in reducing the loss. Weight measures the number of times a feature appears while training the model, with no indication of whether the split had a large effect or not.[24] Both Gain and Cover are averages over the number of appearances of the feature, so Total Gain and Total Cover refer to the sum instead.



```
Mohamed Salah ranks in the following percentiles for each important feature:
    1. Feature pass_targets is in top 94.12%
    2. Feature age is in top 71.62%
    3. Feature pressure_regain_pct is in top 86.46%
    4. Feature fouled is in top 42.9%
    5. Feature comp_level_1. Premier League is 1
    6. Feature goals is in top 98.51%
    7. Feature shots_on_target_pct is in top 45.38%
    8. Feature minutes is in top 98.41%
    9. Feature touches_att_pen_area is in top 100.0%
    10. Feature season is in top 50.05%
```

Figure 5.3: Explanation of top ranking sample (Total Gain)

In Figure 5.3, we use the `total_gain` scores. This is because it offers the most comprehensive assessment when compared to the other metrics. Weight shows us a count of appearances, therefore it doesn't tell us how important each appearance is. Gain alone doesn't show us the full picture, since it ranks a feature which is important and only present in one tree the same as a feature which is important in every tree. Total Gain is also more closely tied to our loss function when compared to Cover and Total Cover which use the Hessian, which makes it more suited to finding features which influence the performance of our model.

We can look through the features that the sample performs well in and get an indication of why this sample was rated so highly. There are 5 clear areas which the player performs well in: `pass_targets`, which refers to the number of times a player was the target of an attempted pass, `pressure_regain_pct`, which refers to the number of times possession is regained when the player applies pressure to an opposition player, `touches_att_pen_area`, which refers to the number of times a player touches the ball in the other team's penalty area, `minutes` and `goals`. We had to do some research into what the `pass_targets` feature meant as it has since been removed from the source website.

Since `pass_targets` refers to the amount of times the player was the target of a teammate's pass, we considered deleting it, as it contains the contribution of the player's teammate. However, it may also refer to the ability of the player to get into positions where they can receive passes. We opted not to delete it but this may be something to consider in the future. It most likely ranks highly as it is a metric which both applies to every player, no matter the position, and also measures how involved a player is in each match. This would make sense as we are attempting to remove players who do not contribute much to the performance of the team.

It is also good to see both `age` and `minutes` as important scores. We want the age to rank highly as it should work as we have shown that market values are strongly correlated with age, and minutes ranking highly ensures that we get players who are consistently performing to a high level, and not players who benefit from low sample size. Most of the high ranking features have a good explanation, however the appearance of the `fouled` feature at second in the list is unexpected. This refers to the amount of times a player is fouled by an opponent. The top 10 is split evenly between players who rank low and high in the `fouled` metric, so we have no indication of whether it has a large positive or large negative effect. A few possible explanations could be:

- Players are often fouled when they get past a defender, or if a defender gets frustrated, which tends to indicate that a player is performing well.
- The fouled metric acts as a combination of free kicks won and penalties won, both of which can put the team in a good position to score.
- Players who get fouled more are more liable to get injured, and frequent or long-term injuries can have a negative effect on market value.

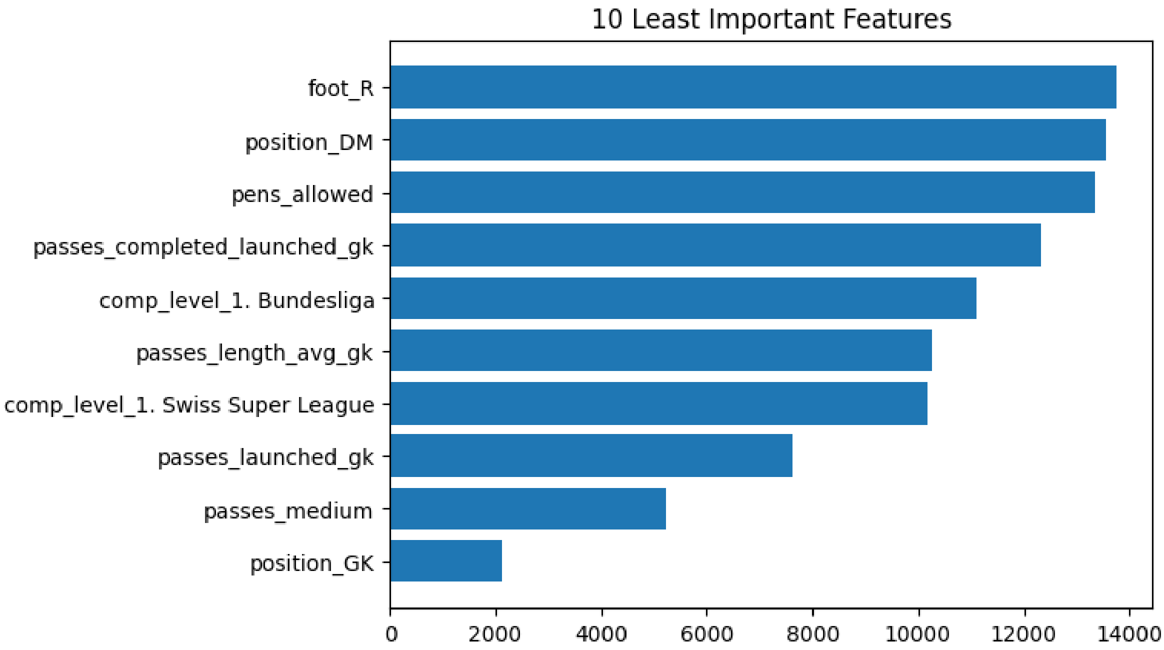


Figure 5.4: Least useful features

Figure 5.4 shows the least important features to our model's prediction. There seem to be more categorical features towards the bottom of the list which we should expect as they can only be present once per tree due to their binary values. These features must provide a large change to the prediction, but features like `foot_R`, referring to right-footed players who make up a majority, do not give the model much information. There are also many goalkeeping stats which would make sense as they are not applicable to the vast majority of the players in the dataset.

5.2 Value Personalisation Model

We need to apply the same logic as above when validating our personalisation model. However, it is much more difficult for us to tie our results into the real world, as predicting how much a player is worth to a given team is more subjective than predicting how much a player is worth generally. After testing the personalisation model with a few examples of successful transfers, it became clear that the model was highly inaccurate. It was assigned less than 1 per cent success rates to objectively successful transfers. One possible reason is that the model is too overfitted, and was simply remembering training examples, which led it to struggle when presented with a new sample. A solution to this was to remove the player stats. Since the model takes in our non-biased market values, it already is indirectly taking player stats into account, so including them in the feature set for this model just increases complexity and leaves us prone to overfitting. Therefore, the stats were removed from the feature set and the model was fitted again using the following grid search.

- `objective`: [“reg:squarederror”, “reg:squaredlogerror”, “reg:pseudohubererror”,“reg:absolutee
- `max_depth`: [1,3,5,**7**]
- `learning_rate`: [0.01, **0.05**, 0.1, 0.2]
- `n_estimators`: [**1000**]
- `min_child_weight`: [**1**]
- `gamma`: [**0**]

We can then repeatedly narrow the existing parameters down, and can also start searching using the previously unused parameters until they no longer reduce our loss function. This eventually results in the following parameters:

- `objective`: “reg:absoluteerror”
- `max_depth`: 7
- `learning_rate`: 0.025
- `n_estimators`: 1000
- `min_child_weight`: 2
- `gamma`: 0.1

This resulted in a massive improvement to our model’s accuracy on paper. However, consider Table 5.2 below, where there is a selection of real world transfers which occurred after the cutoff for our training data in order to evaluate our model.

Three of these were almost objectively successful, and two transfers were unsuccessful, as indicated in the 'Successful' column. Through testing each of these players using the fee which they sold for recently, we can ground our model's predictions in reality. We can see that the model does not achieve useful predictions, and has extremely high levels of error for each sample.

Player	Club Transferred To	Fee (€)	Successful	Model Prediction (€)
Erling Haaland	Manchester City	60,000,000	Y	525,000
Alexander Isak	Newcastle	70,000,000	Y	210,000
Antonio Rüdiger	Real Madrid	Free	Y	262,500
Romelu Lukaku	Chelsea	113,000,000	N	52,500
Casemiro	Manchester United	95,000,000	N	105,00

Table 5.2: Evaluating real-world transfers ($s_1 = 1, s_2 = 1, s_3 = 1$)

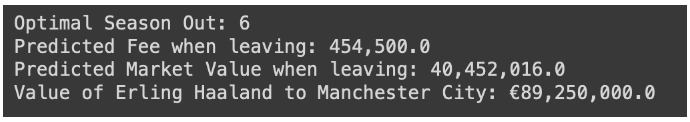
We can see that the model is not performing as expected, with extremely large discrepancies between actual fees and predicted values. However, there are three reasons for this. Firstly, using the difference in market value from when a player joins a club until they leave is a good way to measure performance for players who would expect to improve over their time at the club, however, it heavily penalises players who are already performing well when they join and remain consistent. We can fix this by using the average market value over their time at the club instead. Secondly, since we have set $s_1 = s_2 = s_3 = 1$, we are effectively asking our model to rate the chance that our player is bought for less than their market value, improves significantly, and is sold for a profit. We would expect this to be very rare, so it makes sense that our values are low. Finally, although the previous problems are affecting performance significantly, it does seem that the model's predictions for both fee and market value are relatively low as well.

Since these teams are all relatively high performing, we can change our parameters to mainly prioritise performance (s_2), with a minor focus on getting a good deal when buying (s_1). As shown in Table 5.3, this results in much more accurate predictions.

Player	Club Transferred To	Fee (€)	Successful	Model Prediction (€)
Erling Haaland	Manchester City	60,000,000	Y	89,250,000
Alexander Isak	Newcastle	70,000,000	Y	21,000,000
Antonio Rüdiger	Real Madrid	Free	Y	78,750,000
Romelu Lukaku	Chelsea	113,000,000	N	31,500,000
Casemiro	Manchester United	95,000,000	N	26,250,000

Table 5.3: Evaluating real-world transfers ($s_1 = 0.1, s_2 = 1, s_3 = 0$)

However, as shown in Figure 5.5, this doesn't fix the fact that the outputs of the model (shown in the second and third lines below) are still too low, with the fee prediction in particular being extremely inaccurate, highlighting issues with the current approach.



```
Optimal Season Out: 6
Predicted Fee when leaving: 454,500.0
Predicted Market Value when leaving: 40,452,016.0
Value of Erling Haaland to Manchester City: €89,250,000.0
```

Figure 5.5: Model Output

In order to further investigate the cause of the model's behaviour, it is necessary to investigate the distribution of some key features and outputs. In Figure 5.6, we can see that the four inputs to the equation all have a significant skew towards the right. When initially checking the model, the three terms of the 'success score' equation were used, therefore the Mean Squared Error function was used when searching for parameters. This assumes that the target is normally distributed, when in fact a gamma distribution would fit much better when attempting to predict the Fee Out and MV Out variables. In an ideal world, we could use `reg:tweedie` or `reg:gamma` objective functions, however these are not supported in the `MultiOutputRegressor` at the time of writing.

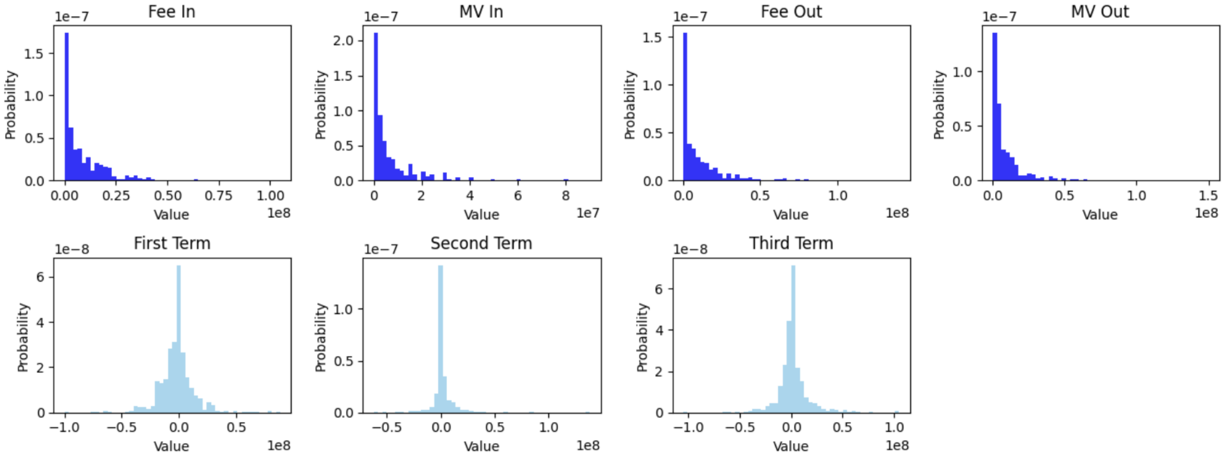


Figure 5.6: Distribution of equation terms in the training set

As we did with the previous model, we can also look at the feature importance scores to attempt to decipher the problem with the model. This time however, it doesn't yield any unexpected results. Regarding the Total Gain metric, we can see our features representing fee, market value, season and age all scoring highly, followed by various categorical features as we would expect.

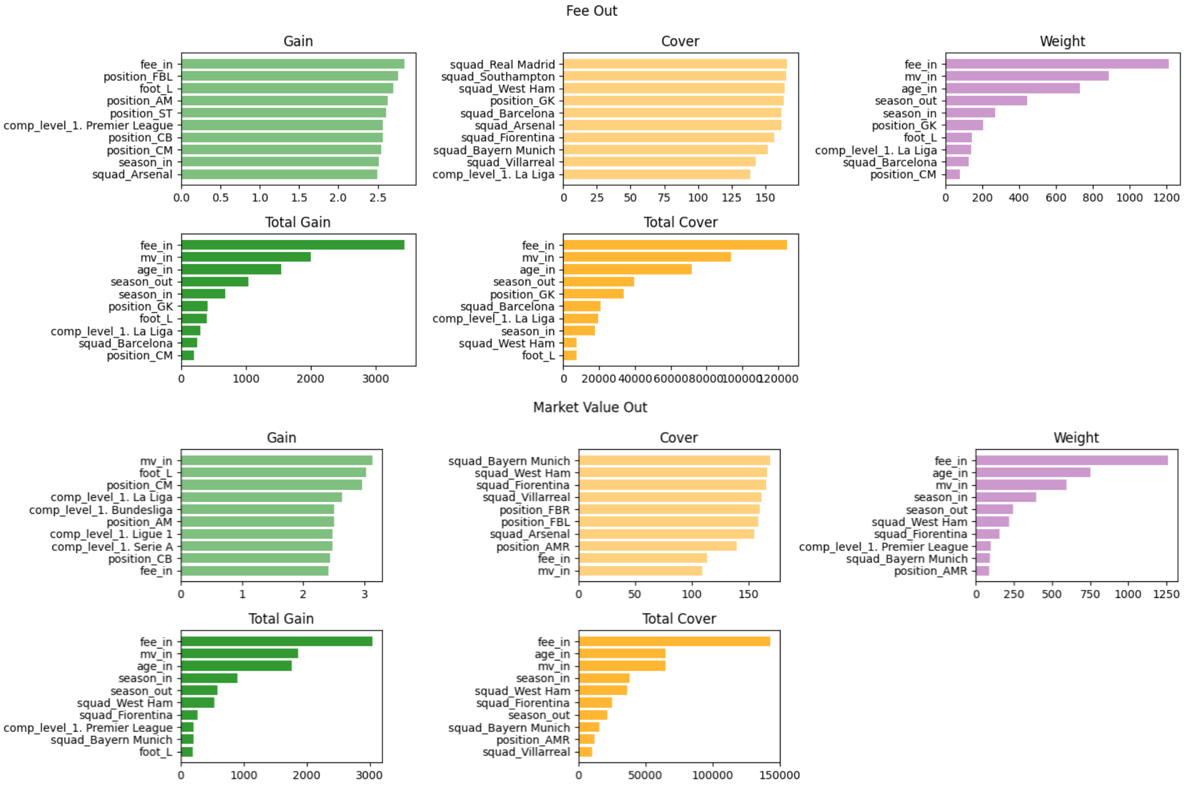


Figure 5.7: Feature importance scores for both outputs

This seems to indicate that the model simply does not have either enough samples, or enough useful features to make an accurate prediction. It seems more likely that the issue is a result of a lack of features, since as we mentioned at the start of this section, we only have the resources to predict this personalised value using financial aspects. However, to accurately measure how valuable a player is to a given club, we need to be able to measure the impact that they would have on the team, which would require detailed positioning, formation and performance data, of which we only have the latter.

6 Project Management

6.1 Resources Used

I predominantly used the `scikit-learn` package in Python for my machine learning tasks. I also used the `scipy` package for some useful ML metrics. I had to do extensive research on Machine Learning due to the fact that it was a new topic for me. The CS342 Module helped greatly in understanding the basics, but since it ran concurrently to the development of this project it was also necessary to do some of the research independently.

All of the software development done over the course of the project was split between using VS Code on the Computer Science department computers, and Google Colab, which is a hosted Jupyter Notebook service. The main reason for using the latter was access to T4 GPU, which offered higher speeds for training the model which was particularly useful when performing large searches across the parameter space. Since this meant that I had data saved on both my personal laptop and the department PCs, there was a low risk of losing any important data. Therefore, I opted not to use a source control website like GitHub as I had planned prior to development.

This project was also the first time I was responsible for handling a relatively large dataset and arranging data into a format which was useful was significantly more challenging than I had previously thought. I read many different articles and watched videos on using the `pandas` Python package for manipulating data, but I found just using the package and taking tasks step by step to be the most useful for learning.

6.2 Methodology

My chosen methodology at the start of the project was to use a 'Water-Scrum-Fall' methodology. This is a hybrid between the Waterfall and Scrum project management methodologies. It consists of the Waterfall planning phase at the start with clearly defined requirements, then it leads into Agile's iterative development phase, and ends with Waterfall's deployment phase. It was particularly suited to this sort of project as it allows us to write a clear plan in the specification, iterate through and potentially re-evaluate requirements after the Progress report (similar to a Scrum sprint review), and then deliver a finished product with this report (like the delivery phase in a Waterfall methodology).

The main issue with this methodology was that I knew that my requirements would change, which is why I ensured I used some sort of Agile methodology, however I severely underestimated how much change would be caused by developing a solution, thinking of a better way and the developing a new solution repeatedly. The amount of issues encountered along the way meant that it may have been beneficial to have used a purely Agile methodology.

However, the main strength of the Waterfall part of the methodology was having a rigorous planning phase. I think that if I had not had such a long planning phase initially, it would have taken a lot longer to make any progress. As mentioned above I knew I had to do some initial research into machine learning, but I neglected research into data handling methods, and if I had elected to use a purely Agile methodology, I believe that I would've spent several sprints making flawed models or incorrectly handling data.

6.3 Timetable

Attached in Appendix A are the timetables outlined in the Specification and Progress Report respectively. The main mistake I made with my initial timetable was that I thought creating my model would take the longest and then validation and visualisation would be quick. However, I found that it was more a cycle of tuning parameters, training a model, removing/adding features, changing objective functions, and reducing over-fitting. In my second timetable I changed my approach, adding in a vital second research phase and focusing more on the cyclical aspect. I think that this timetable was a lot more accurate, and I ended up following it all the way until the time of writing. I think the categories are correct but they all took around a week longer than I had predicted. At the time that

I created the second timetable I didn't anticipate needing two models, but since I had created both 'Preliminary Rating' and 'Team-based Valuation' segments, I had the space in my timetable to create a second model which still fell into the topics which I had outlined beforehand.

6.4 Changes

If I could do this project again, there would be a few changes which I would make with regards to planning, project management and resources used.

I believe the project chosen may have had a slightly too broad scope, as the simultaneous development of two interlinked models proved more difficult than anticipated. I believe that if I were to have only focused on developing a model for more effective valuations (the Value Prediction Model), it would have allowed me to focus on making sure that the model was rigorously fine-tuned and more robust than the solutions present.

I think that when starting a new topic as this project was for me, it is good to do a certain amount of background research and to make a detailed plan which I did, however I believe that I learnt the most and the fastest when I was writing code. Therefore, next time instead of having a detailed research and planning phase at the beginning, I would have one much shorter phase at the start, and then have subsequent short research phases every 1-2 months. This would mean that whenever I was doing research into a topic, it would be directly relevant to what I would be doing next, instead of having to predict months into the future. Secondly, I would have made use of Jupyter Notebooks earlier, as the ability to run blocks of code once and have the variables stored was invaluable towards the end of the project. One advantage of Google Colab specifically was the convenience of my code being available on any device. At some stages of the project it was difficult to tell how long certain tasks would take, and therefore sometimes it was difficult to stick to the timetable. In the future, I would be more strict on myself and only work on the tasks present on the timetable. This would have the added benefit of forcing me to make changes to the timetable before I could spend extra time on tasks, which would lead me to spend more time on my planning processes, and therefore would allow me to have a clearer picture of my next steps at each stage of the project.

7 Conclusions and Future Work

This section will outline the contributions of the two models developed, the extent to which they can be useful in providing a solution to our problem, and possible future changes and extensions to each. The first model, referred to as the 'Value Prediction Model' showed that using XGBoost with total variation distance is a viable strategy to extract an unbiased form of a market value. It seems that the first model vastly outperformed the second model due to the fact that market value is very closely tied to performance, whereas there can be a much larger discrepancy between a transfer fee (a key component of the second model) and performance. From managers, to wages, contract lengths, release clauses and agent fees, there are many variables involved in the prediction of transfer fees, which were simply not freely available in a usable format. The second model, while not as outwardly successful as the first, provides a foundation for future work. Although slightly limited by the data available, it highlighted some of the key factors behind value in the football market. With more comprehensive data in the future the model could be extended to capture the complex nature of the football transfer market.

There are many possible extensions and changes which can be applied to the current project in order to make it a more useful tool in the future. In its current state, the first model considers TVD and MSE equally when selecting parameters. Experimenting with the weighting towards each error metric could yield more reliable and accurate results. The project in its current state could also benefit from a graphical user interface. This would help fix a few usability issues with the project, for example, when inputting a team, it is necessary to type the name exactly how it is stored, so for Manchester United, it is stored as 'Manchester Utd', and alternative spellings will not work. In addition, many European and South American players have accented characters in their names, which have to be replicated in the entry box, so a player search or dropdown box would make the project a lot more usable.

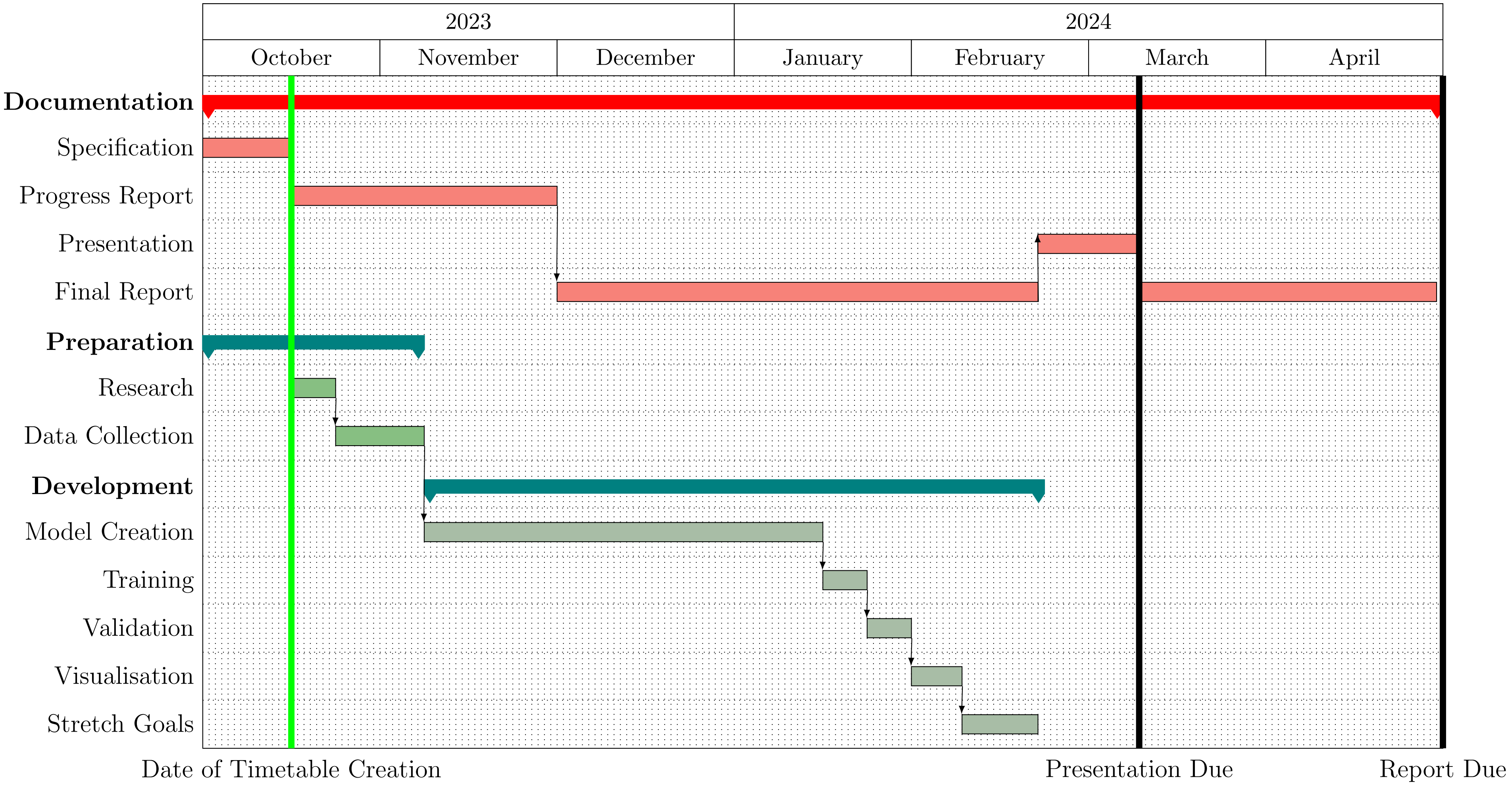
Although most parameter combinations for the models seemed to converge to a local minimum in the error function, a more thorough search of the parameter space may improve model performance, which would be easier with consistent access to more powerful computing resources. Only the most important parameters were adjusted, so tuning some of the unused parameters may have a positive effect on the model's performance. Splitting the `MultiOutputRegressor` into two separate regressors, allowing us to make use of the Tweedie objective function would have been the next step in the project if it were not subject to time constraints, as this should lead to the largest improvement in model accuracy without requiring extensive changes to the codebase.

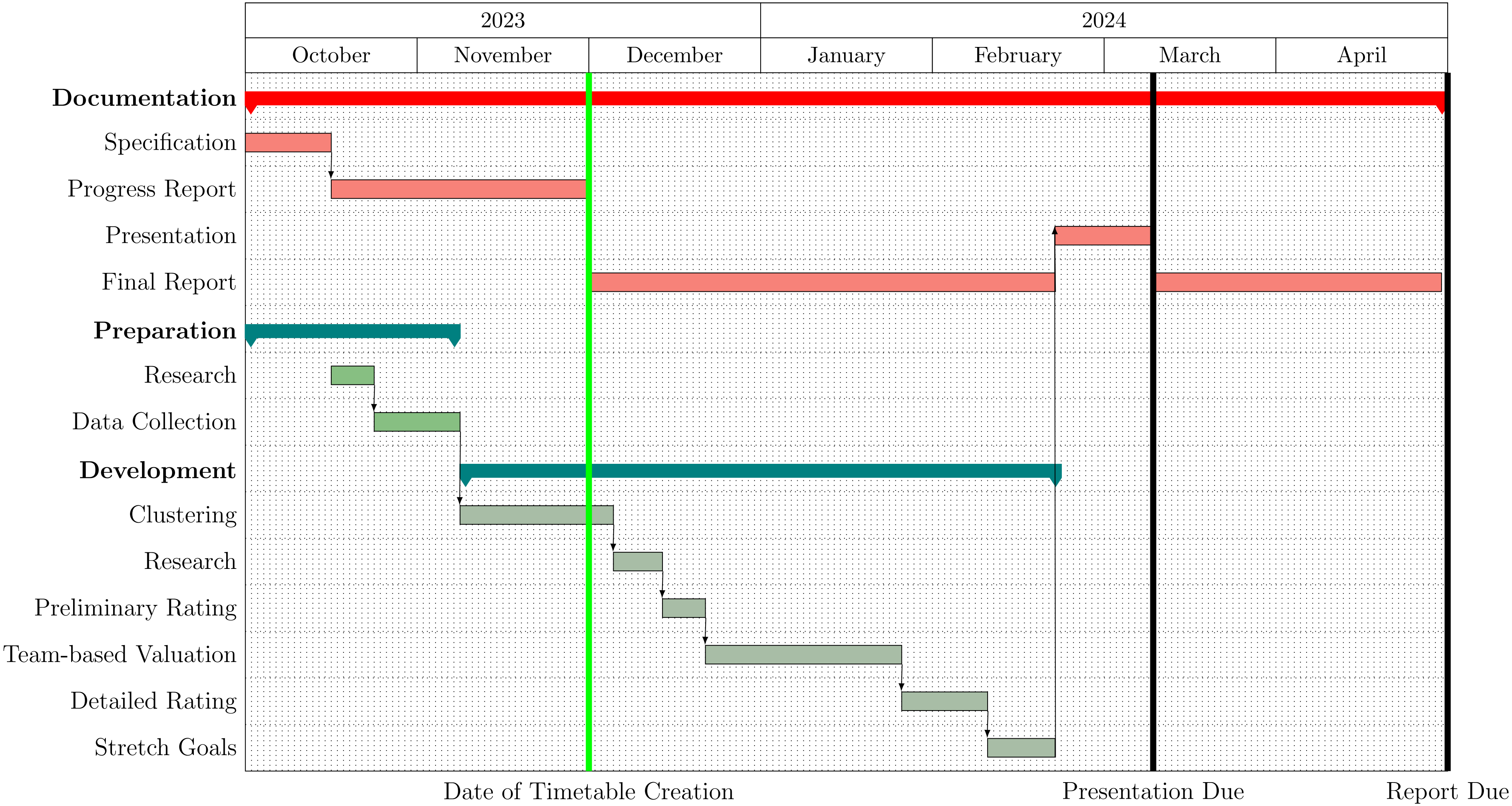
However, by far the most valuable improvement to the model would be access to a larger dataset with increased quality. Some useful features to include would be detailed player positions, match event data, wage data, player contract data, manager data or formation data. Combining all of these data types into a dataset for this project would be a monumental task in itself, but it would allow us to fix issues frequently encountered throughout the project.

8 Author's Assessment of the Project

This project has provided analysis and developed a starting point for generating more accurate individualised market values. This will allow recruitment teams to be able to make more objective decisions with regards to player ability. However, the project suffers slightly from a small dataset, which led to the samples we were most interested in (high-performing players) being treated as outliers which resulted in many predictions being too low. The project also demonstrates the utility of computer science techniques, including machine learning and software engineering applied to the field of sports analytics. Others can use this project as starting point to develop a fully-fledged market value prediction service, or alternatively, some areas of the project could also be generalised into working in any field where valuations are required. This project also explores working with biased features, and proactively mitigating the effects of the bias, something which has not been present in similar projects.

A Appendix - Past Timetables




2023
2024
October
November
December
January
February
March
April
Documentation
Specification
Progress Report
Presentation
Final Report
Preparation
Research
Data Collection
Development
Clustering
Research
Preliminary Rating
Team-based Valuation
Detailed Rating
Stretch Goals
Date of Timetable Creation
Presentation Due
Report Due

B References

[1] FIFA.com. Global Transfer Report 2023. https://digitalhub.fifa.com/m/114622e4e17cf6a8/original/FIFA-Global-Transfer-Report-2023.pdf, January 2024. Accessed on 3 April 2024.

[2] BBC Sport. Neymar: Paris St-Germain sign Barcelona forward for world record 222m euros. https://www.bbc.co.uk/sport/football/40762417, August 2017. Accessed on 3 April 2024.

[3] ESPN. PSG trigger Kylian Mbappe's permanent transfer from Monaco. https://www.espn.com/soccer/story/_/id/37546008/psg-trigger-kylian-mbappe-permanent-transfer-monaco, February 2018. Accessed on 3 April 2024.

[4] BBC Sport. Philippe Coutinho: Barcelona to sign Liverpool and Brazil midfielder in £142m deal. https://www.bbc.co.uk/sport/football/42580173, January 2018. Accessed on 3 April 2024.

[5] The Guardian. Atlético Madrid sign Benfica teenager João Felix for fee of €126m . https://web.archive.org/web/20190911122217/https://www.theguardian.com/football/2019/jul/03/atletico-madrid-sign-benfica-teenager-joao-felix-for-fee-of-126m, July 2019. Accessed on 3 April 2024.

[6] BBC Sports. Antoine Griezmann: Barcelona sign Atletico Madrid forward. https://www.bbc.co.uk/sport/football/48967047, July 2019. Accessed on 3 April 2024.

[7] BBC Sports. Jack Grealish: Man City sign England midfielder from Aston Villa for £100m. https://www.bbc.co.uk/sport/football/57818660, August 2021. Accessed on 3 April 2024.

[8] Sky Sports. Enzo Fernandez: Chelsea sign midfielder in £106.8m British-record transfer deal from Benfica. https://www.skysports.com/football/news/11668/12799395/enzo-fernandez-to-chelsea-blues-

agree-premier-league-record-105m-transfer-deal-for-benfica-midfielder, February 2023. Accessed on 3 April 2024.

[9] BBC Sports. Declan Rice: Arsenal sign England midfielder from West Ham for £105m. https://www.bbc.co.uk/sport/football/65982835, July 2023. Accessed on 3 April 2024.

[10] BBC Sports. Moises Caicedo transfer news: Chelsea sign Brighton midfielder for £100m. https://www.bbc.co.uk/sport/football/66504891, August 2023. Accessed on 3 April 2024.

[11] The Guardian. Chelsea confirm Romelu Lukaku signing from Inter in €115m deal. https://www.theguardian.com/football/2021/aug/12/chelsea-confirm-romelu-lukaku-signing-from-inter-in-115m-deal, August 2021. Accessed on 3 April 2024.

[12] FIFA. The football landscape. https://publications.fifa.com/en/vision-report-2021/the-football-landscape/, 2021. Accessed on 3 April 2024.

[13] The New York Times. The Wisdom of the Crowd. https://www.nytimes.com/2021/08/12/sports/soccer/soccer-football-transfermarkt.html, August 2021. Accessed on 4 April 2024.

[14] Daokang Zhang and Caixin Kang. Players' Value Prediction Based on Machine Learning Method. *Journal of Physics: Conference Series*, 1865(4):042016, apr 2021.

[15] Öznur Ilayda Yılmaz and Şule Gündüz Öğüdücü. Learning football player features using graph embeddings for player recommendation system. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, SAC '22, page 577–584, New York, NY, USA, 2022. Association for Computing Machinery.

[16] Kaggle. https://www.kaggle.com. Accessed on 5 April 2024.

[17] Biniyam Yohannes. Soccer Player Data from fbref.com. https://www.kaggle.com/datasets/biniyamyohannes/soccer-player-data-from-fbrefcom, January 2022. Accessed on 5 April 2024.

[18] Mexwell. Football Data from Transfermarkt. https://www.kaggle.com/datasets/mexwell/football-data-from-transfermarkt?select=player_valuations.csv, October 2023. Accessed on 5 April 2024.

[19] Bhavik Chandna. Football Transfers from 1992/93 to 2021/22 seasons. https://www.kaggle.com/datasets/cbhavik/football-transfers-from-199293-to-202122-seasons, September 2022. Accessed on 5 April 2024.

[20] Opta Analyst. What Are Expected Assists (xA)? https://theanalyst.com/eu/2021/03/what-are-expected-assists-xa/, March 2021. Accessed on 4 April 2024.

[21] FBRef. Premier League Stats. https://fbref.com/en/comps/9/gca/Premier-League-Stats, April 2024. Accessed on 4 April 2024.

[22] UEFA. History of the Ballon d'Or: All the winners. https://www.uefa.com/news-media/news/0287-195e642735da-0594342b9554-1000--history-of-the-ballon-d-or-all-the-winners/, November 2023. Accessed on 26 April 2024.

[23] UEFA. Champions League final highlights and report: Vinícius Júnior scores only goal as Real Madrid beat Liverpool to claim 14th European Cup. https://www.uefa.com/uefachampionsleague/news/0275-15415a62aaf0-36d3e17d7404-1000/, May 2022. Accessed on 26 April 2024.

[24] XGBoost Developers. XGBoost Documentation. https://xgboost.readthedocs.io/en/release_2.0.0/R-package/discoverYourData.html. Accessed on 26 April 2024.