Serienummer MAT-VET-F-23040

Examensarbete 15 hp

Juni 2023

Predicting the transfer market

Using machine learning to predict football transfer fees

Samuel Kristensen

Civilingenjörsprogrammet i teknisk fysik

UPPSALA UNIVERSITET

Predicting the transfer market

Samuel Kristensen

Abstract

Football, like the rest of the world, is constantly changing. As new techniques for data analysis and model construction continue to emerge the sport is increasingly embracing a data-driven approach, adapting to the ever-changing landscape of information and analytics. One part of the data-driven approach football have been adapting is to use machine learning to assist in player valuation. In a world where money is being pumped in from rich oligarchs and sheiks and player prices are steadily rising, clubs need to be able to locate the right players for the right prices and data analytics might help with that.

The purpose of this project is to collect data and statistics related to transfers and players over the past eight years. Through the utilization of three different machine learning techniques and analyzing the gathered data, the aim is to identify the most effective method for player valuation, thereby enhancing the accuracy of the assessment process.

During the project 14216 transfers from all over the world were gathered, featured engineered and used in the different machine learning techniques to predict player transfer costs. Evaluation metrics for the models were calculated and importance of the features are discussed. The machine learning algorithms for the project were able to predict the transfer fees of football player with a high R-squared value which indicates the proportion of the variance in the transfer fees of football players can be explained by the model's predictions

Teknisk-naturvetenskapliga fakulteten
Uppsala universitet, Utgivningsort Uppsala
Supervisor: Evengii Tikhomirov
Subject reader: Martin Sjödin
Examiner: Ken Welch

Contents

1 Introduction

Soccer, or football as it is known in many parts of the world, is the most popular sport globally, with millions of people playing and watching it every day.[1] The 2022 world cup final between Argentina and France achieved a global reach of close to 1.5 billion viewers.[2] Football clubs, especially those in top-tier leagues, invest a lot of resources and time in scouting and signing players to bolster their teams' strength and achieve success on the field. The transfer market refers to the buying and selling of players between clubs, where transfer fees are paid to compensate for the transfer of a player's rights.[3]

The spending on players is increasing rapidly. During the current 22/23 seasons clubs spent in total 9.12 billion euros on players, ten years prior the total spend were 3.86 billion euros increasing the total spend with 236%. Not only are teams spending more, the players are getting more expensive. During the same time period the average annual inflation for football players were 9.0%. That period included two years of covid where football clubs were hit hard with economical problems, from the 13/14 season to the season prior to covid the average inflation for players were 13.8%. Comparing that to the worlds average inflation of 2.84% for the last ten years the world of football have been hit hard with inflation of players.
For a similar player clubs today have to pay on average 116% more than ten years ago.[4]

For a team to master and understand the transfer market is crucial. The likes of Brentford FC, making its way from the third tier of English football all the way to the Premier League in 7 years can be explained by a large part thanks to their brilliant transfer strategy.[5]. Of course the success of the club is not only based on the transfers but it is still a crucial part of it.

The transfer of a player can not only affect the teams performance but in fact the whole club. A football player can in some cases even impact the stock market, for example the stocks of Juventus after the transfer of Cristiano Ronaldo from Real Madrid in 2018 increased with 31.8%. And Sporting CP stocks after selling Gelson Martins to Atletico Madrid went down 14.2%.[6]

The process of valuing a player for transfer is an intricate task that requires a deep understanding of the game, the financial situation of the sport, the players, and their various attributes. Many factors contribute to a player's valuation, including their age, position, skillset, performance, and transfer history.[7] Clubs and agents may use various methods to arrive at a player's value, including expert opinions, subjective analysis, and market trends. However, these methods can be flawed and lead to overpaying for a player or undervaluing a player's worth.

In recent years, data analytics and machine learning have emerged as powerful tools to assist in player valuation. These techniques provide a more objective and data-driven approach to evaluating players, reducing the risk of making costly mistakes in transfer deals. Data analytics involves collecting, processing, and analyzing large volumes of data to uncover patterns, trends, and insights. Machine learning is a subset of artificial intelligence that uses algorithms and statistical models to learn from the data and make predictions or decisions without being explicitly programmed.[8]

1.1 Problem description

The project can be divided into two steps, the first is to obtain the data set using python to create programs for web scraping, the second step is to use the gathered data in different machine learning algorithms to train them and then make predictions.

The purpose of this project is by applying machine learning algorithms to a data set of football players transfer costs and other essential stats gathered through web scraping. It is possible to accurately predict the transfer cost of football players, resulting in a more objective and data-driven approach to player valuation in the soccer transfer market.

To assess the project and evaluate its performance, the project should aim to achieve high accuracy in predicting the market value of football players using machine learning algorithms. This can be measured by evaluating the performance of the different algorithms tested in the project, using common and appropriate metrics for a regression task such as the coefficient of determination (R^2) and the root mean squared error (RMSE).

Additionally, the project can assess the significance of the relationship between the input features and the predicted market value by performing feature importance analysis. This analysis can help identify which features have the most impact on player valuation and further validate the hypothesis that machine learning algorithms can provide a more objective and data-driven approach to player valuation.

If the results of the project show that the machine learning algorithms tested can accurately predict the transfer cost of soccer players, and that the input features have significant impact on player valuation, then the hypothesis can be supported. Conversely, if the results show low accuracy and/or insignificant feature importance, the hypothesis may be rejected.

2 Theory

2.1 Web scraping and data

Web scraping is the process of extracting data from websites using automated software tools. It is a technique used to extract large amounts of data from websites that would otherwise be difficult or time-consuming to obtain manually[9]. Web scraping involves the use of software to collect data from websites automatically and then save that data into a structured format, such as a CSV file, dataframes or a database.

The theory behind web scraping is that websites are built using HTML, which is a markup language that defines the structure and content of a web page. HTML is made up of tags, which define elements such as headings, paragraphs, links, and images. Web scraping software can be used to navigate and parse the HTML of a web page and extract the desired data, using a combination of tags, attributes, and other selectors.[10]

One of the most common tools used for web scraping in Python is BeautifulSoup (version 4.12.0)which is used in this project. BeautifulSoup is a Python library that allows developers to extract structured data from HTML and XML documents. It provides a simple and intuitive interface for parsing HTML, allowing developers to easily navigate the HTML tree and extract data using a wide range of selectors.[11]

Another important concept in web scraping is HTTP requests and responses. When a user visits a website, their web browser sends an HTTP request to the website's server, which responds with an HTTP response containing the HTML content of the page. Web scraping software can mimic this process by sending HTTP requests to a website's server and receiving the HTML content in response. This allows the software to access and extract data from websites just like a web browser.[10]

Once data is scraped it can be stored in a csv or a pickle file. The csv file is human readable however the pickle requires less disk space and are faster to parse.

2.2 Feature Engineering

In machine learning, a feature is an attribute that describes an aspect of a single data object. Example features are age, height, nationality, etcetera. Meaningful features are the basis of data analysis. They can be used to describe underlying objects, and to distinguish and characterize different (explicit or latent) groups of objects.[12]

Feature engineering is a broad term that is used talking about getting relevant information from the data, it include topics such as: Feature transformation that is about constructing new features from existing ones, Feature selection which is selecting the features that are relevant for the project and feature generation which is generating new futures among others.

Python have a module called featurewiz that can automate these procedures.
The package provides the automatic feature selection algorithm Minimum Redundancy Maximum Relevance (MRMR) and selects the best number of un-correlated features that have maximum mutual information about the target without having to specify the number of features.
It also have built in categorical-to-numeric encoders and can perform feature engineering tasks such as interactions that can create new features for example multiplying two features with each other, groupby that will generate group by features by grouping categorical features and target which can encode and transform all categorical features using certain target encoders.
The feature selection of featurewiz works by setting a correlation threshold and then using the SULOV method and recursive xgboost (more on xgboost in section 2.3.1). SULOV stands for Searching for Uncorrelated List of Variables and the algorithm is based on the previously mentioned MRMR algorithm. The featurewiz runs xgboost 5 times and selects the top features for every run. The final product is then all the features with the best MRMR and a duplicate of the initial data set with only the best features. [13]

Feature selection algorithms can be broadly classified in two categories, minimal-optimal and all-relevant. The Maximum Relevance and Minimum Redundancy are in the feature selection class minimal-optimal. The goal for this class is to identify a small set of features that put together have the maximum possible predictive power. That is in contrast to all-relevant algorithms that are designed to select features that individually have any predictive power at all. The advantages of a minimal-optimal is to reduce overfitting where the model trains the model to well and the noise and random fluctuations of the training data is picked up and learned as concepts by the model.[14]

Another aspect to feature engineering is feature scaling or data normalization. An issue for many data sets can be when combining features such as for instance minutes played that can range from 0 minutes to 4000 minutes and height that can range from 140 cm to 210 cm. The problem with this is that the data is measured in different units and the distribution of the data can be very different, both within and between variables. The purpose of normalization is to transform the data so that the feature become either dimensionless and/or have similar distributions.
There are several methods that can be used for feature scaling, one of them is Z-score normalization. Z-score or standard score normalization is widely used in machine learning and transforms the data number into a score using the formula:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

Where z is the standard score, μ is mean and σ is the standard deviation. [15]
Normalization of the target variable can also be important, if the data set contains outliers there can often

be advantageous to transform the target so that it is normally distributed. One simple and often use way of doing that is by taking the logarithm of the target data in a regression model.[16]

2.3 Machine learning algorithms

2.3.1 xgboost

XGBoost (Extreme Gradient Boosting) is a popular machine learning algorithm used for supervised learning problems, such as classification and regression. It is an optimized version of the Gradient Boosting algorithm that uses a combination of gradient boosting and decision tree algorithms.

The XGBoost algorithm works by constructing a sequence of decision trees, where each subsequent tree attempts to correct the errors of the previous tree. It begins by creating a single decision tree, then it calculates the error or loss of the predictions and uses this information to create a new decision tree. The new tree attempts to correct the errors of the previous tree, and the process is repeated until a stopping criterion is reached.[17]. Figure 1 illustrates a flow chart of how the xgboost algorithm works.

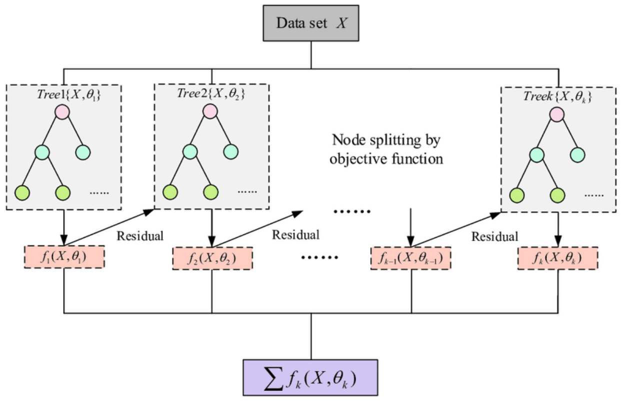


Figure 1: The flow chart of xgboost algorithm[18]

One key feature of XGBoost is that it uses a gradient-based optimization method to find the optimal weights for the decision trees. The algorithm calculates the gradient and the second derivative of the loss function with respect to the predicted value, and uses this information to update the weights of the decision trees.

XGBoost also includes several other features that make it a powerful algorithm for machine learning tasks. These features include regularization techniques to prevent overfitting, efficient handling of missing data, and a flexible objective function that allows for custom loss functions.

Another important feature of XGBoost is its ability to handle large datasets and high-dimensional data. The algorithm is designed to be scalable and can handle millions of samples and features. It also includes several parallelization techniques to speed up the training process and optimize memory usage.

One limitation of XGBoost is that it may be sensitive to the choice of hyperparameters, such as the learning rate and the maximum depth of the decision trees. It is important to carefully tune these hyperparameters to ensure the best performance of the algorithm.[19]

2.3.2 Neural Network

A Neural Network is a subset of machine learning and are the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another. A neural network is composed of node layers, containing an input layer, one or more hidden layers and an output layer. Every node from the first input layer connects to the next hidden layer with an associated weight and a threshold and if the input for that node is above the threshold value the node is activated sending data to the next set of layer. Otherwise no data is passed along to the next layer from that node. This continues from the first hidden layer to the second hidden layer and so on until it reaches the output layer.
The process of determining the weights is through training the network, that process can be very computationally intensive. Once the weights are determined though the new data can be processed very quickly.[20]

There are a lot of different things that have to be determined when building a neural network. The size of the hidden layers, the number of epochs, the batch sizes, the activation function is some of them.
The batch size is defines the number of samples (data points) to work through before updating the internal model parameters. An epoch is when the whole data set is passed forward and backward through the neural network once, when a epoch is finished the model updates its weights.[21] The activation function is what determines if the threshold is fulfilled, and thus determining if the node should be activated or not. The purpose of the activation function is to introduce non-linearity for the output of the nodes. There are a lot of different activation functions with different behaviours, but the common thing for all of them is that they just are normal functions, for example tanh, sigmoind and relu functions.[20]

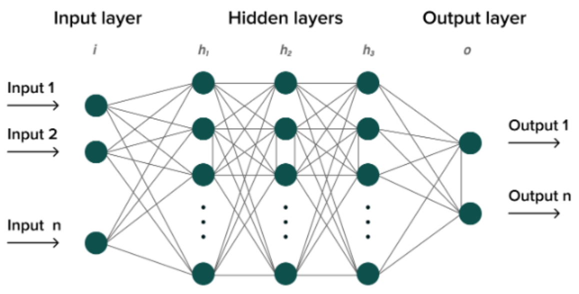


Figure 2: The tensor of a neural network[22]

Neural networks can also be equipped with regularization and early stopping, the purpose of these are to stop training the network once the improvement for the neural network is small enough. This is to avoid the overfitting of the network and decreases the amount of epoch to train.
To determine how the model is performing to be able to stop early we need to have some kind of measurement of the network. This is the loss function, it is a function that for each epoch measures the error between the training prediction and value. The most common loss function for regressions is the negative mean squared error which is calculated by the following formula.

$$-MSE = -\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2}$$

The reason for why it is negative is because standard optimization program have the function to minimize the loss function, and since the object is to maximize the MSE the result will be the same if minimizing the negative MSE.

Python have several powerful libraries and packages for building and working with neural networks, TensorFlow, Keras and PyTorch are some of them.

2.3.3 Stacking Generalization

Stacked Generalization or stacking for short, is an ensemble machine learning algorithm. It involves using a machine learning model to learn how to best combine the predictions from contributing ensemble members. The point of stacking is to explore a space of different models for the same problem. The idea is to attack a learning problem with different type of models which are capable off learning different parts of the problem and use each model to create an intermediate prediction, one prediction for each model. Then a meta model is added which learn from the intermediate predictions the same target. The final model is therefore stacked on top of each other, hence the name. By leveraging cross-validation, stacking maximizes the utilization of the available data and enhances the overall performance of the final stacked model.

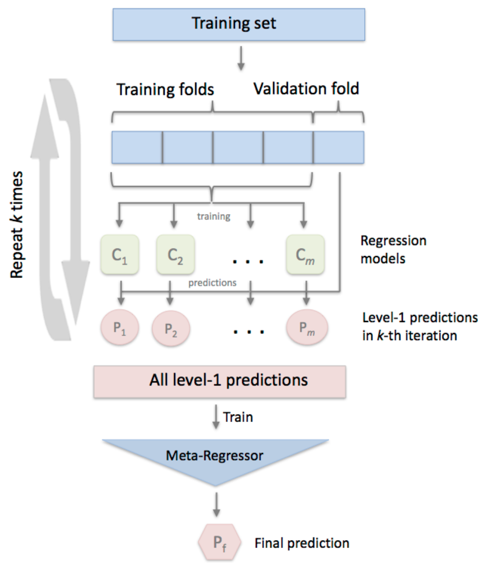


Figure 3: How Stacking Generalization works [22]

The stacking works by splitting the training data into K-folds, a base model is then fitted on the K-1 parts of the training data and predictions are made for the Kth part. This is repeated for each part of the training data. The base model is then fitted on the whole training data set to calculate its performance on the test set. This procedure is repeated for all the other base models and predictions from the train set are used as features for the second level model. The second level model finally makes the predictions on the test set. [23]

When doing a stacking ensemble a variety of models can be used and are often chosen based on the problem and the data. A good way of choosing the right regressors is to have a diverse set of base models with different strength and weaknesses, this way when integrated with a meta-model in the stacking framework, allows for better coverage of the complexity of the data and can increase the performance.
The chosen models for this framework consists of GradientBoostingRegressor, RandomForestRegressor, ExtraTreesregRessor, SVR, Lasso and xgboost.

2.4 Parameters and Hyperparameters

Parameters and hyperparameters in machine learning are a very important part, no problem is the same and no data is the same, then it should not be strange that the parameters are not the same for all problems. There are several different parameters and hyperparameters that can be tuned for the different machine learning algorithms.
A parameter is a configuration variable that is internal to the model and are required by the model when making predictions. Their values define the skill of the model for the problem and are often set manually, some examples are the weights in a neural network or the coefficients in a linear or logistic regression.
A hyperparameter is a configuration that is external to the model and whose value most often cannot be estimated from data. The hyperparameter are often tuned for a given predictive modeling problem and used to help estimate model parameters. Most of the hyperparameters cannot be calculated from the data and can be extremely hard and computationally costly to determine for optimization. Some hyperparameters are the learningrate for xgboost and neural networks or number of estimators for tree regression models.

Most models have a default set of hyperparameters and when used the model often works good, however in order for the model to work the best for the problem in hand the hyperparameters need to be tuned.
The most common way to do this is by using a grid search or a random search of parameters. Python have modules from scikit-learn[24] with both a GridSearchCV and a RandomizedSearchCV function that can do this.
The GridSearchCV uses a grid of different hyperparameters. Using the model handed to it for example xgboost it performes an exhaustive search over all the parameters and by scoring for example with negative mean absolute error it returns the best set of parameters and hyperparameters for the problem. The downside with using GridSearchCV is that if you want to tune 5 hyperparameters with 5 different values and cross validate them 5 times it needs to run $5^5 \cdot 5 = 15625$ times and depending on the computer that is being used can mean incredible amounts of time.
The RandomizedSearchCV does instead of searching through all the combination of parameters it searches through a fixed number of iterations where the parameters are most often sampled from a specified distribution. This method is not as thorough as the GridSearchCV but the results can still be good and is most often a lot less computationaly costly.

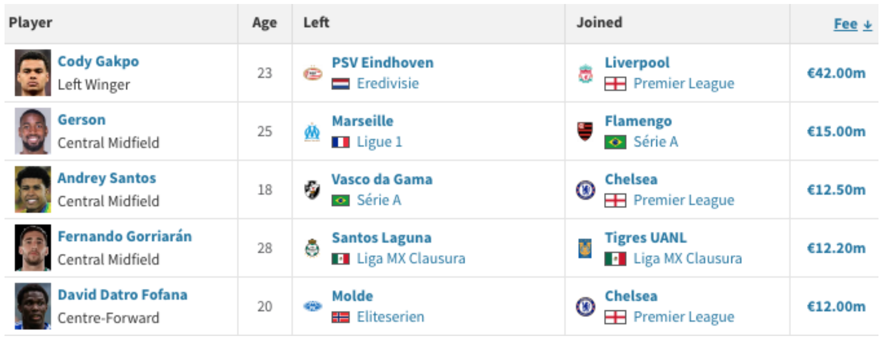
The neural network does not have any packages that can automate this procedure so they have to be tuned manually. The parameters for a neural network are important.The epochs for will update its weight every time an epoch is finished, if done to few times the model can be underfitted but when done to many times the model might get overfitted.[21]. One hyperparameter to tune in order to find the right amount of epochs is the early stopping, in this we can change the minimum delta and the patience. If an epoch does not improve the performance more than the minimum delta the number of patience times in a row the early stopping is applied and the training stops. This is to choose the right epoch parameter and avoid overfitting.

3 Method

As mentioned in section 1.1 the project consists of two larger parts, web scraping to gather the data, and using the data to train machine learning algorithms and then predict untrained data. The data used in this project is taken from Transfermarkt[25] and include statistics and other data for paid transfers from 2016 to 2023. After all data is gathered it is then stored using pythons pickle module.

3.1 Data gathering

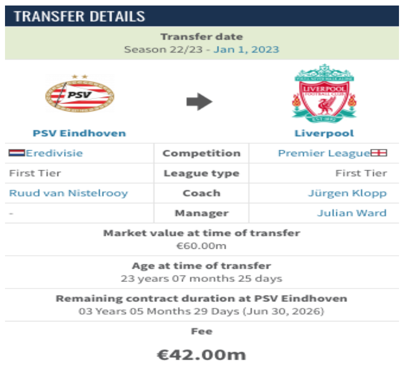
The first part of the data gathering was to get the players that have made a transfer, the figure bellow (Figure 4) shows the first five transfers on the 1 Jan 2022. With beautifulsoup we get the Player, Age, Club Left, Club Joined, Fee and Transfer Id. The transfer id can not bee seen in the figure but is crucial since it links the player with the specific transfer (A player usually have more than one transfer in his career). The scraping is looped through all dates from 1st of January 2016 till the 1st of January 2023, for every date it loops through all the pages that contain transfers and for every page loops through all the players in that page. After we have looped through all the players in this time period we then drop all rows were the transfer is a free transfer or is unknown. This results in a dataframe containing 14216 transfers.



Player	Age	Left	Joined	Fee ↓
Cody Gakpo Left Winger	23	PSV Eindhoven Eredivisie	Liverpool Premier League	€42.00m
Gerson Central Midfield	25	Marseille Ligue 1	Flamengo Série A	€15.00m
Andrey Santos Central Midfield	18	Vasco da Gama Série A	Chelsea Premier League	€12.50m
Fernando Gorriarán Central Midfield	28	Santos Laguna Liga MX Clausura	Tigres UANL Liga MX Clausura	€12.20m
David Datro Fofana Centre-Forward	20	Molde Eliteserien	Chelsea Premier League	€12.00m

Figure 4: First 5 transfers 1 Jan 2022 [25]

From this dataframe we can create other programs that uses the transfer id from the players to scrapes several other pages with statistics for each player.
We first scrape the transfer details and transfer history for each player. From this we get stats such as the league left and joined as well as what tier they are, we also get the previous clubs that the player have played for and the previous fees that have been payed for the player.



TRANSFER DETAILS

Transfer date
Season 22/23 - Jan 1, 2023

PSV Eindhoven → Liverpool

PSV Eindhoven		Liverpool
Eredivisie	Competition	Premier League
First Tier	League type	First Tier
Ruud van Nistelrooy	Coach	Jürgen Klopp
-	Manager	Julian Ward

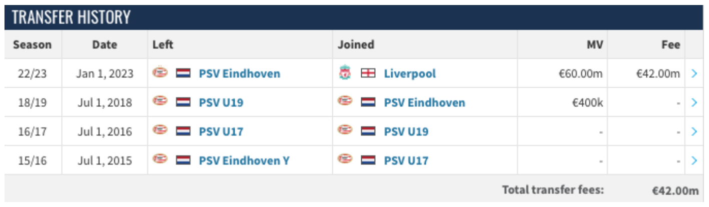
Market value at time of transfer
€60.00m

Age at time of transfer
23 years 07 months 25 days

Remaining contract duration at PSV Eindhoven
03 Years 05 Months 29 Days (Jun 30, 2026)

Fee
€42.00m

(a) Transfer details for specific transfer id [25]



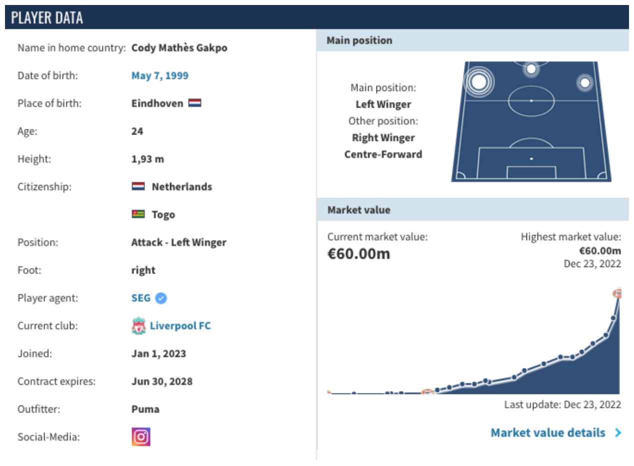
TRANSFER HISTORY

Season	Date	Left	Joined	MV	Fee
22/23	Jan 1, 2023	PSV Eindhoven	Liverpool	€60.00m	€42.00m
18/19	Jul 1, 2018	PSV U19	PSV Eindhoven	€400k	-
16/17	Jul 1, 2016	PSV U17	PSV U19	-	-
15/16	Jul 1, 2015	PSV Eindhoven Y	PSV U17	-	-
				Total transfer fees:	€42.00m

(b) Transfer history for player [25]

Figure 5: The transfer details and history for one transfer

The next program we have utilizes the same transfer id to get the players personal data. The personal data can be seen in figure 6, from that we get information such as the age, height, position etcetera. The program loops through all the players and returns a dataframe.



PLAYER DATA

Name in home country:	Cody Mathès Gakpo
Date of birth:	May 7, 1999
Place of birth:	Eindhoven
Age:	24
Height:	1,93 m
Citizenship:	Netherlands
	Togo
Position:	Attack - Left Winger
Foot:	right
Player agent:	SEG
Current club:	Liverpool FC
Joined:	Jan 1, 2023
Contract expires:	Jun 30, 2028
Outfitter:	Puma
Social-Media:	

Main position

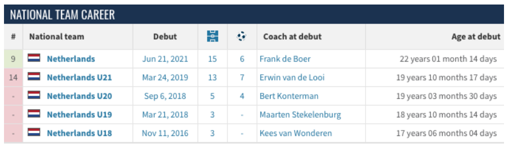
Main position: Left Winger
Other position: Right Winger, Centre-Forward

Market value

Current market value: €60.00m
Highest market value: €60.00m Dec 23, 2022
Last update: Dec 23, 2022
Market value details

Figure 6: Player personal data [25]

From then we scrape for the player statistics from both club and country. For the club the stats for the three seasons prior to the transfer were scraped for each player resulting in a dataframe with roughly 2 million matches scraped. For the National team all the matches for the players national career were scraped.

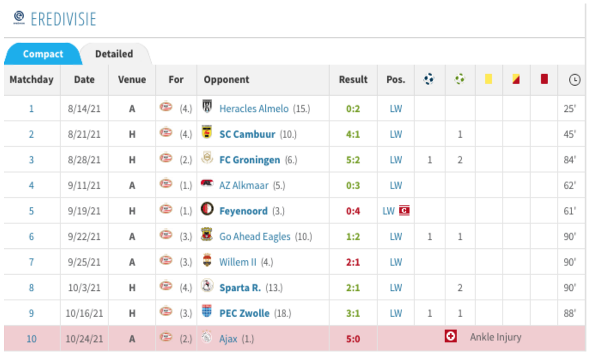


NATIONAL TEAM CAREER

#	National team	Debut		⚽	Coach at debut	Age at debut
9	Netherlands	Jun 21, 2021	15	6	Frank de Boer	22 years 01 month 14 days
14	Netherlands U21	Mar 24, 2019	13	7	Erwin van de Looi	19 years 10 months 17 days
-	Netherlands U20	Sep 6, 2018	5	4	Bert Konterman	19 years 03 months 30 days
-	Netherlands U19	Mar 21, 2018	3	-	Maarten Stekelenburg	18 years 10 months 14 days
-	Netherlands U18	Nov 11, 2016	3	-	Kees van Wonderen	17 years 06 months 04 days

Figure 7: Player National team career [25]

The player statistics for club is given by season so if a player changes the club during a season, for example during the winter transfer window or between leagues with different periods for the season the player will get the stats for the whole season. To counter that the program checked and matched every game with the club the player represented based on the dates of the transfers.

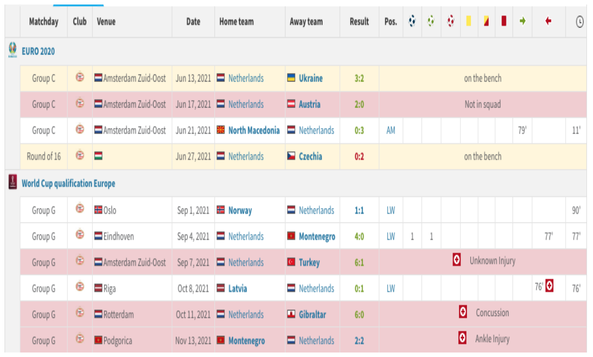


EREDIVISIE

Compact | Detailed

Matchday	Date	Venue	For	Opponent	Result	Pos.	⚽	⚽	🟨	🟨🟥	🟥	🕒
1	8/14/21	A	(4.)	Heracles Almelo (15.)	0:2	LW						25'
2	8/21/21	H	(4.)	SC Cambuur (10.)	4:1	LW		1				45'
3	8/28/21	H	(2.)	FC Groningen (6.)	5:2	LW	1	2				84'
4	9/11/21	A	(1.)	AZ Alkmaar (5.)	0:3	LW						62'
5	9/19/21	H	(1.)	Feyenoord (3.)	0:4	LW						61'
6	9/22/21	A	(3.)	Go Ahead Eagles (10.)	1:2	LW	1	1				90'
7	9/25/21	A	(3.)	Willem II (4.)	2:1	LW						90'
8	10/3/21	H	(4.)	Sparta R. (13.)	2:1	LW		2				90'
9	10/16/21	H	(3.)	PEC Zwolle (18.)	3:1	LW	1	1				88'
10	10/24/21	A	(2.)	Ajax (1.)	5:0			Ankle Injury				

(a) Player statistics for Club [25]

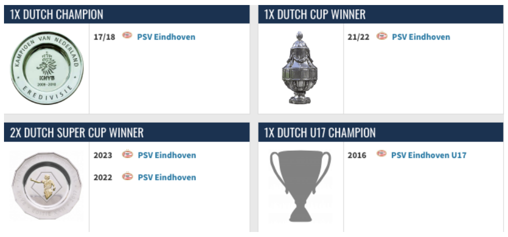


Matchday	Club	Venue	Date	Home team	Away team	Result	Pos.	⚽	⚽	⚽	🟨	🟨🟥	🟥	→	←	🕒
EURO 2020																
Group C		Amsterdam Zuid-Oost	Jun 13, 2021	Netherlands	Ukraine	3:2	on the bench									
Group C		Amsterdam Zuid-Oost	Jun 17, 2021	Netherlands	Austria	2:0	Not in squad									
Group C		Amsterdam Zuid-Oost	Jun 21, 2021	North Macedonia	Netherlands	0:3	AM							79'		11'
Round of 16			Jun 27, 2021	Netherlands	Czechia	0:2	on the bench									
World Cup qualification Europe																
Group G		Oslo	Sep 1, 2021	Norway	Netherlands	1:1	LW									90'
Group G		Eindhoven	Sep 4, 2021	Netherlands	Montenegro	4:0	LW	1	1						77'	77'
Group G		Amsterdam Zuid-Oost	Sep 7, 2021	Netherlands	Turkey	6:1	Unknown Injury									
Group G		Riga	Oct 8, 2021	Latvia	Netherlands	0:1	LW								76'	76'
Group G		Rotterdam	Oct 11, 2021	Netherlands	Gibraltar	6:0	Concussion									
Group G		Podgorica	Nov 13, 2021	Montenegro	Netherlands	2:2	Ankle Injury									

(b) Player statistics for country [25]

Figure 8: Player statistics for club and country

The final part of the data gathering was to get the players achievements.
Achievements might not tell how good a player is since it depends on how integral the player was to achieve the trophy and how hard the competition were to win, but it might boost the players reputation and the hype around the player.


1X DUTCH CHAMPION
17/18 PSV Eindhoven

1X DUTCH CUP WINNER
21/22 PSV Eindhoven

2X DUTCH SUPER CUP WINNER
2023 PSV Eindhoven
2022 PSV Eindhoven

1X DUTCH U17 CHAMPION
2016 PSV Eindhoven U17

Figure 9: Player Achievements [25]

In the end 8 datafiles were created for each web scraping that could be worked on in the Feature Engineering step.

3.2 Feature Engineering

The featured engineering was done in two parts, the first part is to translate some of the data that were on the wrong format into something useful and creating some new features that might be relevant from the club and national matches played.
This step included changing the height (1,93m), age (24 years 5 months 11 days), Market value (€60.00m) etcetera, to floating numbers. It also included creating features from the large data set of club and national matches played into something useful. For each and every player features were created based on the last 30, 20, 10 and 5 matches played for club and national teams.

The second part consisted of using the automated feature engineering algorithm featurewiz (section 2.2) to generate and keep the best features. This was done by simply inputting the data file together with a correlation threshold that finds all the pairs of highly correlated variables exceeding the threshold. Then some other feature engineering task could be added for featurewiz to perform and after the program were done the output is a data file with the selected features and the new created features together with all its data. After the five xgboost rounds were completed we were left with the final dataset to use for training and predicting.

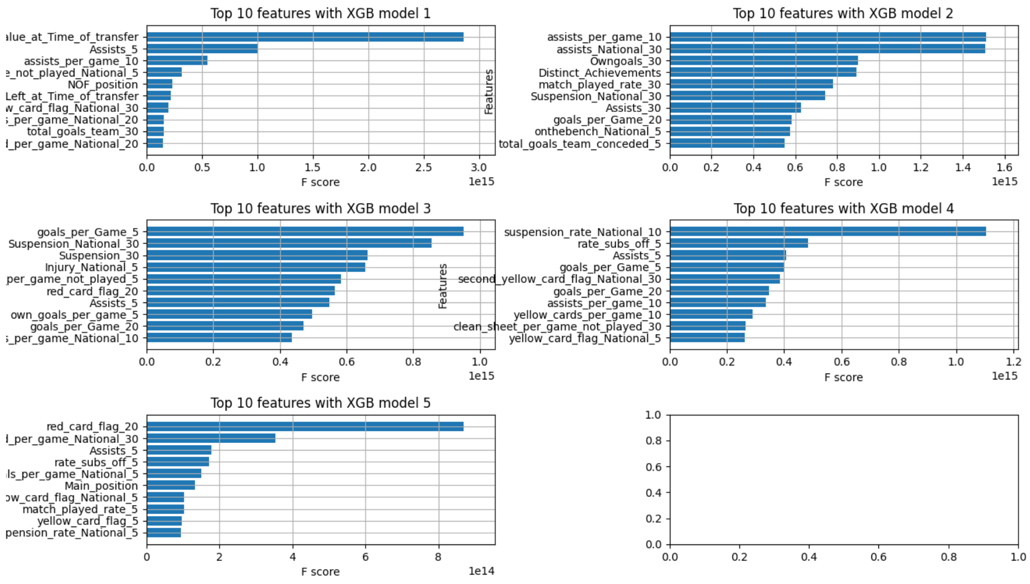


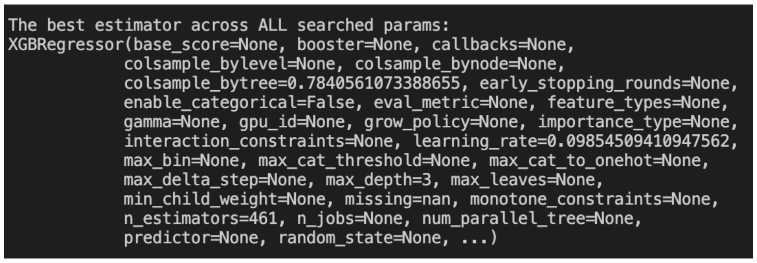
Figure 10: Featurewiz top 10 features every iteration

Finally the target is transformed logarithmic so that it becomes more normally distributed.

3.3 Training and predicting

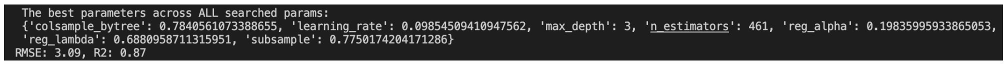
The training of the different models are quite similar to each other. The first thing we do is to split the data in to two sets, one training set that consisted of 80% of the data and one test set which consisted of the remaining 20%. The training set is to train the models and is also divided into smaller subsets for cross-validation, and the test set is used by the models to make predictions so we then can compare the predicted values with the actual target values from the test set. To most accurate evaluate the different models the train and test sets are split the same way each and every time.

For the xgboost algorithm and the stacking algorithm the RandomizedSearchCV is applied and then searches 200 different combinations for seven hyperparameters and crossvalidates each search five times. The search takes about three hours and the results were given.



```
The best estimator across ALL searched params:
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=0.7840561073388655, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=0.09854509410947562,
             max_bin=None, max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=3, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             n_estimators=461, n_jobs=None, num_parallel_tree=None,
             predictor=None, random_state=None, ...)
```

Figure 11: Best estimator xgboost



```
 The best parameters across ALL searched params:
 {'colsample_bytree': 0.7840561073388655, 'learning_rate': 0.09854509410947562, 'max_depth': 3, 'n_estimators': 461, 'reg_alpha': 0.19835995933865053,
 'reg_lambda': 0.6880958711315951, 'subsample': 0.7750174204171286}
RMSE: 3.09, R2: 0.87
```

Figure 12: Best params for xgboost

The xgboost then uses the best estimator with the best parameters to train the model and then predicts the test data, From the predicted target values we can analyse them and compare it to the true values of the test data set and thus get the R^2 and RMSE score for the model.

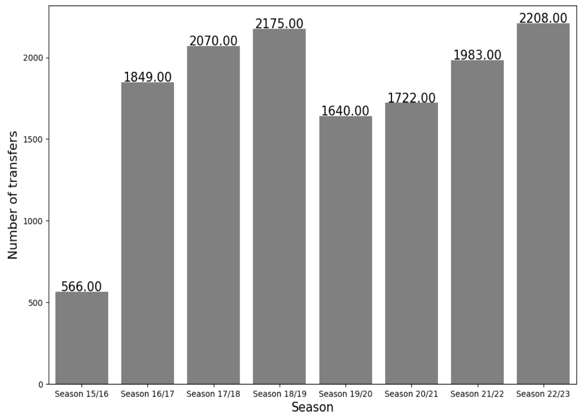
For the stacking grid search the number of hyperparameters to search is a lot more since it consists of seven different models that have up to 20 different hyperparameters each. Therefore the search is limited to one hyperparameter for each model, totaling a search of 50 different combinations.
After the best parameters are selected the model gets trained with the training set and when finished training the model predicts the target from the test data set and evaluates the performance.

The neural network trains a bit different from the other models. The model iterates over the training data in batches, performing forward and backward passes, updating the model parameters, and evaluating the model on the cross validation data. This is done for each batch up until 1000 batches or the early stopping criteria is fulfilled. The input layer of the neural network consists of 112 nodes, one for each input feature. The input layer then connects to the next two layers wit a node ratio of roughly 2/3 of the previous one. The final layer is the output layer and is only one single node that gives a floating number which is the output.
Once the training is done the model can similarly to the other two models predict the unseen test data and the evaluation metrics can be calculated.

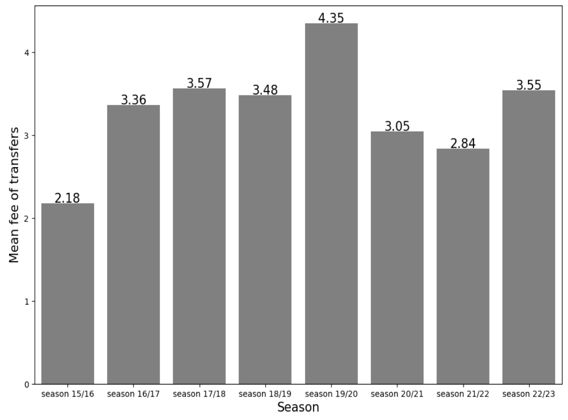
4 Results and discussion

4.1 Data exploration

The data that was gathered from the web scraping consist of 14 213 different transfers spanning from the 15/16 season till the 22/23 season. The 15/16 season only consists of the winter transfers because the scraping begin on January 1st 2016, hence the low amount for season 15/16 in the left plot of figure 13. The right plot of the same figure shows us the mean fee of transfers for every season, and after downwards trend after the 19/20 season which can largely be explained by covid and the financial effect it had on football clubs the mean fees for players are starting to reach the save levels as prior to covid. The low mean fee of the 15/16 season can also be explained by the winter transfer window and clubs tend to spend less money in it.



(a) Number of transfer for every season



(b) Mean fee of transfer every season

Figure 13: Transfers and fees every season

The distribution of the players main position can be seen in figure 14. The biggest category is the centre forward which represents about 20 percents of the data. There are some positions such as left midfield, second striker and right midfield that are significantly underrepresented in the data set with only 0.9 percent for each category, this might introduce some bias in the model and to solve it one idea would be to instead only have the four broader categories of goalkeeper, defender, midfielder and attacker.

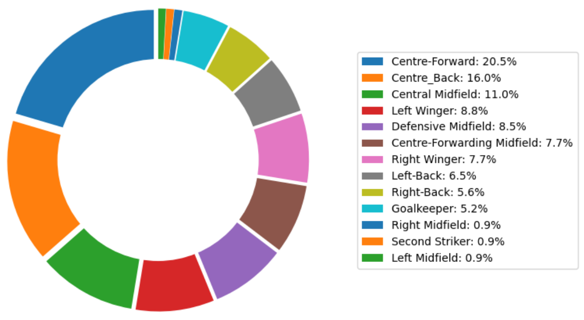
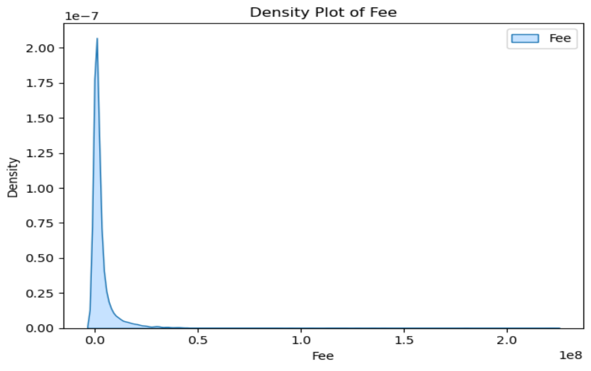
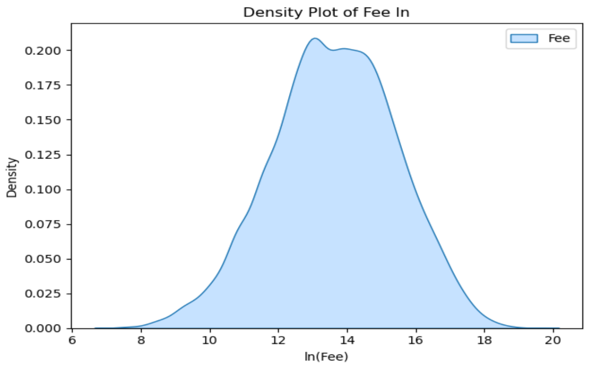


Figure 14: The distribution of the transfer fees

The density of all transfer fees (left) and the density of the transfer fees transformed with the natural logarithm (right) can bee sen in the figure 15. We can see that after the transformation to logarithmic the distribution of the transfer fees are more normalized.



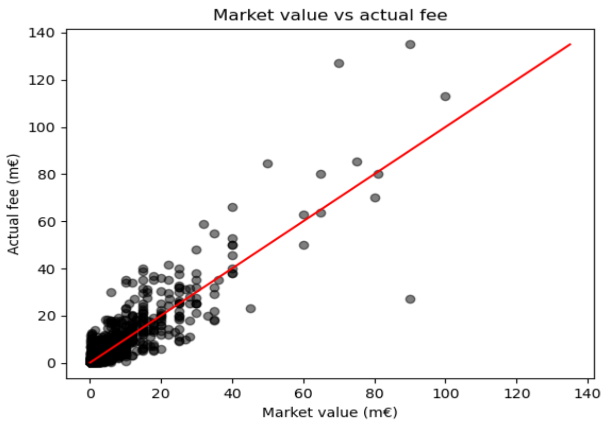
(a) The density of the transfer fees



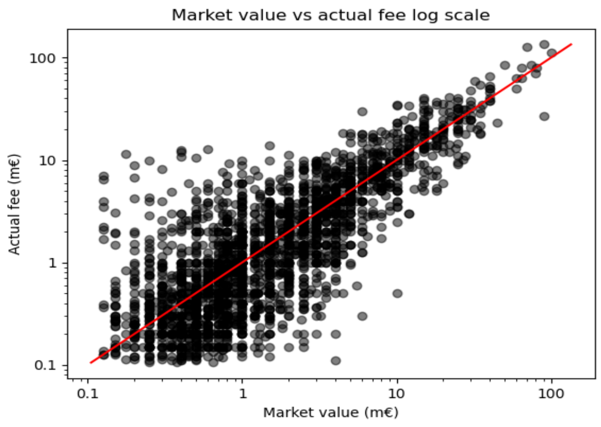
(b) The density of the transfer fees ln

Figure 15: Density graphs for fees untransformed and transformed

To somewhat benchmark the performances of our models we can look at how transfermarkt market valuation (in millions euro) compares to the fees that the clubs had to pay for the players (in millions euro). Figure 16 and table 1 shows the scatterplots and the evaluation metrics that we are using to evaluate our models. The left plot visualizes the relationship between transfermarkts valuation of the player and the fees in a linear scale while the right plot visualizes the relationship on a logarithmic scale. The players are split in the same way as the data for the models and is only showing the test data set in the scatterplot so that the comparison is as fair as possible.



(a) Scatter plot for transfermarkt market value



(b) Logarithmic scatter plot for transfermarkt market value

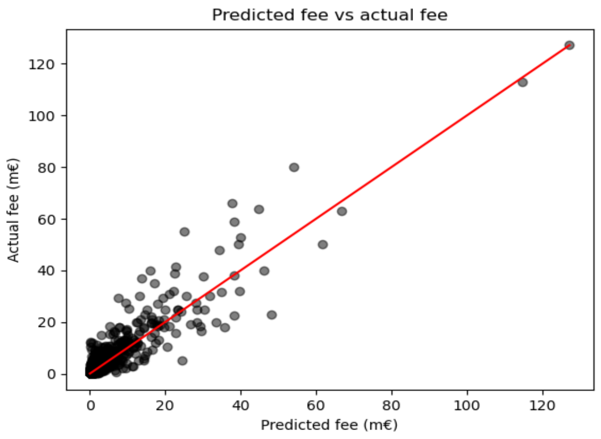
Figure 16: Scatter plot for market value vs actual fee

Table 1: Performance metrics of transfermarkt market valuation

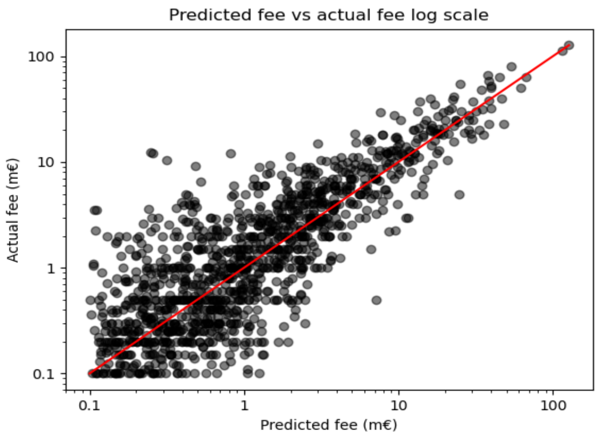
Transfermarkt	RMSE	R^2
	3.68	0.72

4.2 xgboost

The model predicted 1422 players from the unseen test set. The models prediction plotted against the actual values is shown in the scatter plots bellow (figure 17). The left figure is plotted on a linear scale and shows how far of the model is from the actual fee whereas the right figure is plotted on a logarithmic scale and shows a more representative picture of the relative error for the predictions.



(a) Scatter plot for xgboost



(b) Logarithmic scatter plot for xgboost

Figure 17: Scatter plot for xgboost with actual vs predicted fee

For the xgboost model a plot of the feature importance can be made, figure 18 shows the 20 most important features.
There is no surprise that the transfer value at the time of transfer and the average transfer value for the player before the transfer are the two most important features since they are basically in some sense the fair price of the player according to transfermarkt.

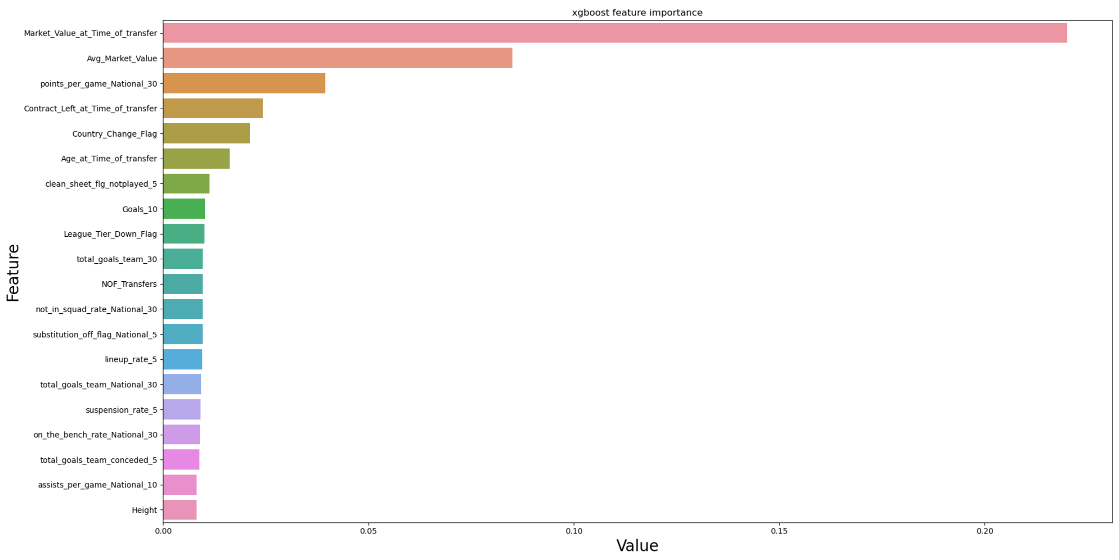


Figure 18: Feature importance for xgboost

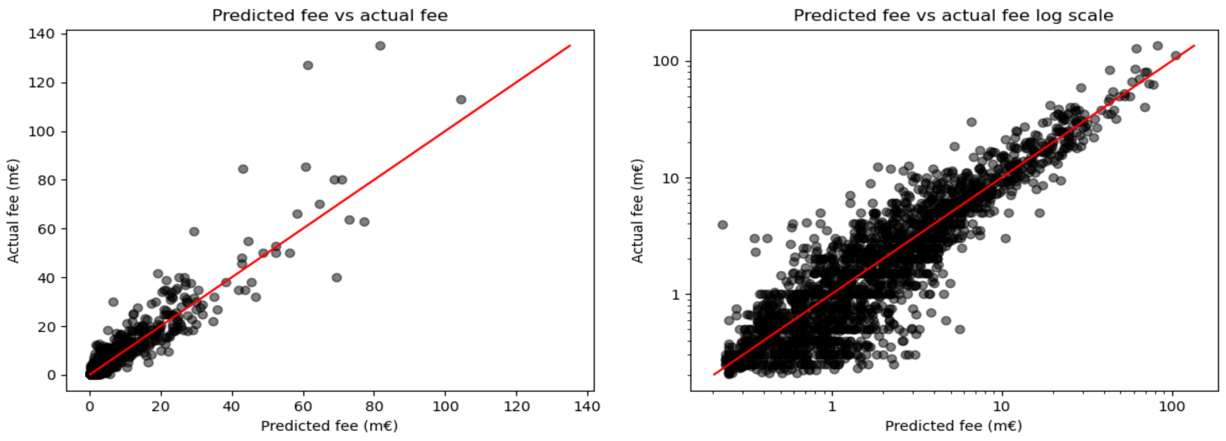
Finally the evaluation metrics for the xgboost model is seen in the table. The model have an root mean squared error of 3.35 million euros, meaning that on average the model deviates from the actual transfer fees with approximately 3.35 million euros. The R-squared value indicates the proportion of the variance in the transfer fees of football players that can be explained by the model's predictions. The R^2 score for the model is 0.85, or in other words approximately 85% of the variability in the transfer fees can be accounted for by the model.

Table 2: Performance metrics of xgboost model

xgboost	RMSE	R^2
	3.35	0.85

4.3 Stacking Generalization

The stacking generalizations predictions is plotted in figure 19 the same way as for the xgboost, the left figure on a linear scale and the right figure on a logarithmic scale. For an ensemble method plotting feature importance is a lot harder since it differs for the different models and is therefore not included for this method.



(a) Scatter plot for stacking generalization (b) Logarithmic scatter plot for stacking generalization

Figure 19: Scatter plot for stacking generalization with actual vs predicted fee

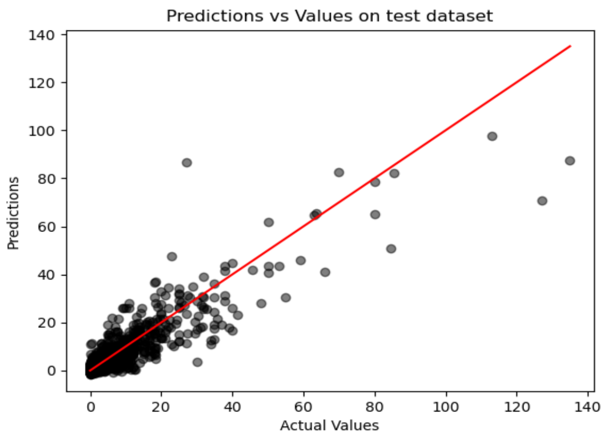
The evaluation metrics of the stacking generalization is found in Table 3. The stacking generalization achieved a root mean squared error of 2.89 million euros and a R-squared value of 0.88. The RMSE for the stacking method is lower and the R^2 for the model is higher than for the xgboost, both suggesting that the model gives better predictions. If we compare the scatter plots for both figures, the scatter plots for the stacking generalization looks a lot more compact than the ones for the xgboost.

Table 3: Performance metrics of stacking model

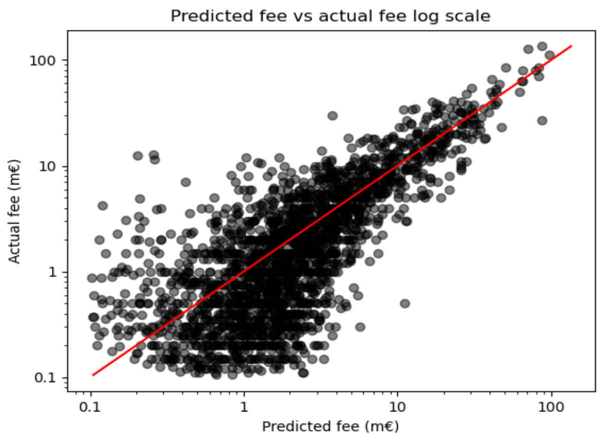
stacking	RMSE	R^2
	2.89	0.88

4.4 Neural Network

The results for the predictions using the neural network can bee seen in the scatter plots of figure 20 and the evaluation metrics in table 4. With the right figure showing the actual fee against the predicted fee on a linear scale and the right figure on a logarithmic scale.



(a) Scatter plot for the neural network



(b) Logarithmic scatter plot for the neural network

Figure 20: Scatter plot for the neural network with actual vs predicted fee

The Neural network underperformed a lot in comparison to the other two methods. With an RMSE of 3.62 million euros and an R^2 score of 0.75 it can predict transfer fees slightly better than just the market value but not a lot better.

Table 4: Performance metrics of the neural network

Neural network	RMSE	R^2
	3.62	0.75

We can compare the models in table 5.

Table 5: Performance metrics of all the models

	RMSE	R^2
Market value	3.68	0.72
xgboost	3.35	0.85
stacking	2.89	0.88
NN	3.62	0.75

5 Conclusions

The aim for this project was to gather data for football transfers and stats for the players, and using the data build machine learning models to be able to predict football transfers and give a more objective and data-driven approach to the transfer market. Three models were programmed, trained and used to predict transfer fees for an unseen test data set. Out of the three models the stacking generalization performed the best, with a root mean squared error of 2.89 million euros and a R-squared value of 0.88.

All of the models performs better both in terms of the RMSE and the R^2 score than just by looking at transfermarkts valuation of the players. All the models had a lower root mean squared error and a higher R-squared value suggesting that the models have a higher explanation of the transfer fees than only the market value.
It is difficult to say if the models works as a predictor for football transfers since for a transfer around 120 million euro a deviation of three million euro seems fairly good, however for a low transfer of one million a three million deviation is an error of 300% which is not a good prediction.

When looking at the features from the xgboost algorithm some features seems very reasonable being high up such as the market value at the time of the transfer and the average market value upp to the time of the transfer since they are an objective valuation of what the price should be as well as the age of the player and the remaining contract of the player, these are all features that applies to all players. Some other features seems more random to rank highly in the feature importance, why should points per game the last 30 games for the national team be more important then in the last 20, 10 or 5 games or points per game the last 30 games for club. And some other features were removed where the correlation were to high between them and therefore can not be measured on how important they might be, this step was automatic to avoid overfitting but a manual selection of the features could be made instead gathering more information about the transfermarket an carefully choosing what features would be best.

There are numerous factors that make it incredibly challenging to accurately predict the transfer market. When significant sums of money are involved, multiple parties such as lawyers, sponsors, clubs, and other stakeholders strive to secure the most advantageous deals for their own interests. Often, these negotiations occur discreetly, behind closed doors, in pursuit of highly lucrative arrangements.

Some factors that can be accounted for that most likely would have been relevant for this project is to include release clauses. Some transfers are hard for a model to explain, for example how Erling Haaland could be bought for 60 million euros according to transfermarkt even though he according to them were worth 150 million euros, had two years left on his contract and were under 22 years at the time of the transfers. In fact according to the stacking generalization model the predicted transfer fee came out to 130 million euros. Adding a flag for release clauses could have helped explain that.

Other things that can explain some transfers are the hype around a player and the requirement of the purchasing team. A team in desperate need of a striker will probably spend a lot more than other teams, especially of the selling club knows that. A lot of talks around a player can also bring the price up for that player, even though the stats behind the players performance might look quite regular if a lot of people see a bright future for the player in question and he starts getting hyped, the fee might rocket. Those could have been some of the reasons why João Felix price went as high as 127.2 million euros with a valuation of 70 million euros at the time or why Neymar were bought for 222 million euros while valuated at 100 million euros. It is a lot harder to find concrete stats about a players hype however the amount of articles or youtube videos about that player can be measured.

In conclusion, even though the models can not perfectly predict every transfer in the world of football it can to a certain extent explain it better than by just looking at the market valuation of the player and give some insight using a data-driven approach on what controls the transfer fee of a football player.

References

[1] Nielsen Holdings plc. World fotball report 2018. URL https://www.nielsen.com/wp-content/uploads/sites/2/2019/04/world-football-report-2018.pdf. Accessed: 9th April 2023.

[2] FIFA. One month on: 5 billion engaged with the fifa world cup qatar 2022™. URL https://www.fifa.com/tournaments/mens/worldcup/qatar2022/news/one-month-on-5-billion-engaged-with-the-fifa-world-cup-qatar-2022-tm. Accessed: 10th May 2023.

[3] Robert Simmons. Implications of the bosman ruling for football transfer markets. *Economic Affairs*, 17(3):13–18, 1997.

[4] Loïc Ravenel Drs Raffaele Poli and Roger Besson. Inflation in the football players' transfer market (2013/14-2022/23). URL https://football-observatory.com/IMG/sites/mr/mr82/en/. Accessed: 24th May 2023.

[5] OneFootball. How moneyball has changed football transfers forever. URL https://onefootball.com/en/news/how-moneyball-has-changed-football-transfers-forever-34949231. Accessed: 12th May 2023.

[6] Atanas Nikolaev. This is how football transfers impact the stock market. URL https://www.fool.co.uk/2022/02/08/this-is-how-football-transfers-impact-the-stock-market/. Accessed: 12th May 2023.

[7] Transfermarkt.Co.In. Transfermarkt market value explained—how is it determined? URL https://www.transfermarkt.co.in/transfermarkt-market-value-explained-how-is-it-determined-/view/news/385100. Accessed: 9th April 2023.

[8] Edward Nsolo, Patrick Lambrix, and Niklas Carlsson. Player valuation in european football. In Ulf Brefeld, Jesse Davis, Jan Van Haaren, and Albrecht Zimmermann, editors, *Machine Learning and Data Mining for Sports Analytics*, pages 42–54, Cham, 2019. Springer International Publishing. ISBN 978-3-030-17274-9.

[9] NetNut. Web scraping: Definition, tools techniques). URL https://netnut.io/web-scraping/. Accessed: 24th May 2023.

[10] Mitchell. (2018). web scraping with python: collecting more data from the modern web (second edition.). o'reilly. URL https://uub.primo.exlibrisgroup.com/permalink/46LIBRIS_UUB/d23b4h/alma991018319207807596. Accessed: 9th April 2023.

[11] beautifulsoup4 4.12.2. URL https://pypi.org/project/beautifulsoup4/. Accessed: 12 May 2023.

[12] Chapman Hall/CRC. Feature engineering for machine learning and data analytics. URL https://books.google.se/books?hl=sv&lr=&id=661SDwAAQBAJ&oi=fnd&pg=PP1&dq=feature+engineering+for+machine+learning&ots=zIx5csooCl&sig=wsi0_Rg1cC6u_8KApuG0-oS1fAw&redir_esc=y#v=onepage&q=feature$%$20engineering$%$20for$%$20machine$%$20learning&f=false. Accessed: 12 May 2023.

[13] AutoViML. featurewiz. URL https://github.com/AutoViML/featurewiz. Accessed: 15th May 2023.

[14] Peng H. Ding, C. (2005). minimum redundancy feature selection from microarray gene expression data. URL https://doi.org/10.1142/s0219720005001004. Accessed: 15th May 2023.

[15] Mahbubal Alam. Data normalization in machine learning. URL https://towardsdatascience.com/data-normalization-in-machine-learning-395fdec69d02. Accessed: 19th April 2023.

[16] Carlos R. Gonz´alez. Optimal data distributions in machine learning.

[17] NVIDEA. Xgboost. URL https://www.nvidia.com/en-us/glossary/data-science/xgboost/. Accessed: 24th May 2023.

[18] Rui Guo, Zhiqian Zhao, Tao Wang, Guangheng Liu, Jingyi Zhao, and Dianrong Gao. Degradation state recognition of piston pump based on iceemdan and xgboost. *Applied Sciences*, 10:6593, 09 2020. doi: 10.3390/app10186593.

[19] Yingui Qiu, Jian Zhou, Manoj Khandelwal, Haitao Yang, Peixi Yang, and Chuanqi Li. Performance evaluation of hybrid woa-xgboost, gwo-xgboost and bo-xgboost models to predict blast-induced ground vibration. *Engineering with computers*, 38(Suppl 5):4145–4162, 2022. ISSN 0177-0667.

[20] Chris M. Bishop. Neural networks and their applications. rev sci instrum 1 june 1994; 65 (6): 1803–1832. URL https://doi.org/10.1063/1.1144830. Accessed: 15th May 2023.

[21] Epochs, batch size, iterations. URL https://machine-learning.paperspace.com/wiki/epoch. Accessed: 16th May 2023.

[22] mlxtend. Stackingregressor: a simple stacking implementation for regression. URL https://rasbt.github.io/mlxtend/user_guide/regressor/StackingRegressor/. Accessed: 16th May 2023.

[23] L. Breiman. Stacked regressions. machine learning 24, 49–64 (1996). URL https://doi.org/10.1023/A:1018046112532. Accessed: 15th May 2023.

[24] Scikit-learn. Machine learning in python2. URL https://scikit-learn.org/stable/. Accessed: 15th May 2023.

[25] Transfermakt. URL https://www.transfermarkt.com/. Accessed: 12th May 2023.

Populärvetenskaplig sammanfattning

Detta är ett projekt som fokuserar på att förutsäga fotbollspelares övergångssummor med hjälp av maskininlärning. Maskininlärning är en gren av artificiell inteligens och datavetenskap som fokuserar på att använda data och algoritmer för att träna program att lära sig mönster och konstant förbättra sig så att de bästa förutsägningarna kan göras. Man tränar alltså modeller att imitera sättet som människor lär sig, och gradvis förbättra dess noggrannhet.

Projektet delades upp i två delar, den första var att samla in data för övergångar och spelarnas statistik så som ålder, marknadsvärde, längd med mera. Den andra delan är att använda denna uppsamlade data för att träna modellerna, skapa förutsägningar och värdera deras prestanda.
Datan uppsamlades genom att programmera funktioner som automatiskt går in på en hemsida som innehåller fotbollsspelares övergångssummor samt all statistik och övrig information som behövs för de alla spelarna. Datan sammanstalls efter informationen är uppsamlad och sammanställs i en datafil.
Datafilen bearbetas genom att omvandla och välja ut statistiken som är mest fördelaktig för projektet.
Den andra delen av projektet var att programmera tre olika maskininlärnings algoritmer. Algoritmerna var olika men de har lika funktion och mål, att hitta samband mellan olika statistiska mått från datan och skapa en regressionsmodell från det för att förutsäga fotbolsspelares övergångssummor.

I den första delen hämtades övergångssummor för 14216 fotbollsspelare med personlig statistik för dessa fotbollspelare samt även matchstatistik från runt 2 miljoner matcher. Matchstatistiken omvandlades och samlades upp för varje spelare så att kategorier som mål per match de senaste 30, 20, 10 och fem matcherna skapades med mera.

De olika maskininlärningsalogritmerna presterade olika bra men den bästa hade ett medelfel på 2.89 miljoner euro för varje spelare samt kunde förklara hur värderingen av hur fotbollspelare sker med 85 %.

Hållbarhetsaspekt

Ekonomisk hållbarhet: Transfermarknaden för fotboll, om den hanteras på ett ansvarsfullt sätt, kan bidra till ekonomisk hållbarhet. Övergångar i fotbollen genererar betydande intäkter för klubbar, ligor och spelare. Dessa intäkter kan återinvesteras i infrastruktur, ungdomsutvecklingsprogram och samhällsinitiativ. Detta är i linje med SDG 8 (Decent Work and Economic Growth) genom att främja hållbar ekonomisk tillväxt och skapa arbetstillfällen inom fotbollsindustrin.

Social hållbarhet: Fotbollsöverföringar kan ha en social inverkan genom att främja inkludering och mångfald. När klubbar investerar i spelare från olika bakgrunder och regioner främjar det mångkultur och främjar SDG 10 (Reduced Inequalities). Fotboll i sig är en oerhört social och sedd sport, representation av olika bakrunder, kulturer och religioner främjar SDG 10.

Etiska överväganden: Marknaden för fotbollstransfer kan också hantera etiska problem relaterade till spelarens rättigheter och arbetsvillkor. Att säkerställa rättvisa löner, säkra arbetsmiljöer och respektera spelarnas rättigheter är i linje med SDG 16 (Peace, Justice, and Strong Institutions). Överföringar som prioriterar spelarens välfärd och följer etablerade regler främjar ett hållbart och ansvarsfullt fotbollsekosystem.