Machine Learning and Deep Learning Techniques for Football Market Value Prediction

Arda Açıkgöz [1]   Mehmet Furkan Çalışkan [1]

Abstract

In this study, we develop a machine learning approach to estimate football player transfer fees based on data extracted from the FC2024 dataset, using the market value calculated by Transfermarkt as the ground truth. The objective is to build an accurate regression model that leverages player attributes, performance metrics, and other relevant features to predict market values effectively. Utilizing neural network and xgboost models, we achieve a good coefficient of determination ($R^2 = 0.7$), demonstrating strong predictive accuracy. Furthermore, our model calculates market values with a mean error rate of €1,000,000, which is notably low given the inherent variability and complexity of the data. These results underscore the potential of machine learning in sports economics, offering valuable insights for clubs, agents, and analysts in evaluating player valuations. Future work may focus on exploring alternative models and advanced feature engineering to further refine predictive accuracy and interpretability.

1. Introduction

Soccer, undeniably one of the most popular and widely celebrated sports globally, continues to captivate billions of fans. As of 2022, it is estimated that the sport has drawn the attention of approximately 3.5 billion people worldwide [1](Kenny, 2023). This immense popularity has transformed soccer into a lucrative industry, attracting numerous companies and funneling vast amounts of money into its ecosystem. Within this dynamic and competitive landscape, there is a growing need for effective financial strategies to manage resources. To address this, our project aims to develop a model capable of accurately estimating market valuations for soccer players. This model will not only assist clubs and companies in optimizing their financial decisions but also empower players by providing them with a fair assessment of their worth, helping them avoid undervaluation and lowball offers. To achieve this goal, we are employing a diverse range of regression models, including K-Nearest Neighbors (KNN) regression, linear regression, support vector machine regression, and random forest regression. Each of these methodologies aims unique strengths to the table, enabling us to capture different patterns and relationships within the data. Additionally, we are incorporating advanced techniques such as Principal Component Analysis (PCA) and feature selection processes to handle the high-dimensional nature of our dataset effectively. Our dataset contains numerous categorical variables, such as player nationality and position, which cannot be directly ordered and require one-hot encoding for analysis. These preprocessing steps are crucial for ensuring that the model operates efficiently and delivers accurate, actionable insights.

2. Related Work

Estimating football transfer fees has become a critical area of research in sports analytics, leveraging advances in machine learning, statistical modeling, and data analysis. Transfer fees are influenced by multiple factors, including a player's performance metrics, age, position, contract duration, and market demand, making the modeling process complex. Studies have highlighted the effectiveness of linear regression in capturing key performance indicators (KPIs) such as goals scored, assists, and defensive contributions in relation to transfer fees. However, such models often struggle with non-linear relationships, prompting the use of advanced machine learning methods such as Random Forests and Gradient Boosting Machines, which have shown superior accuracy in handling high-dimensional and non-linear datasets. Those studies also explored the integration of player-specific features, including the impact a player did on the match derived from the match events (McHale)(McHale & Holmes, 2023). Neural networks, especially deep learning models, have been applied to gather new data that regular machine learning models might not derive. For instance, deep reinforcement learning models have been used to understand a player's effects on the overall team performance. (Liu, G)(Liu et al., 2020) In addition to objective ratings, crowd-sourced ratings from the sofifa.com website have also been widely used in transfer fee modeling. These ratings, which are based on player attributes such as heading, shooting, passing, and more, have been utilized by several studies in the field. For example, Kirschstein Liebscher (2019)(Kirschstein & Liebscher, 2019), Behravan Razavi (2021)(Behravan & Razavi, 2021), Coates Parshakov (2021have all incorporated sofifa ratings in their models of transfer fees or player valuations. Specifically, they have used two key ratings from sofifa: the overall rating, which reflects a player's current playing ability, and the potential rating, which estimates a player's future ability. These ratings have consistently shown strong statistical significance when used as explanatory variables in transfer fee models, suggesting that they effectively capture attributes that decision-makers value when determining player prices. Sofifa ratings have therefore proven to be valuable tools in understanding the factors influencing transfer fees. Open-source datasets have been pivotal in advancing this field, providing rich historical data for research. However, the availability and quality of such datasets often vary, requiring researchers to address data imbalances and missing value. Overall, the intersection of sports analytics and machine learning continues to evolve, providing actionable insights for clubs and analysts to navigate the highly competitive transfer market.

3. The Approach

3.1. Dataset

We utilized the FC Ratings dataset, which contains data collected from FC games spanning the years 2015 to 2024. This dataset consists of 180,021 rows and 109 columns, including both numerical and textual attributes. It provides comprehensive player information, such as height, weight, nationality, position, speed rating, shot rating, and various other metrics. Additionally, we gathered data from Transfermarkt, a trusted and reliable source for football market prices and transfer fees. The Transfermarkt dataset contains 156,758 rows, offering key details such as player names, their former and new clubs, transfer market values, and transfer fees.

After collecting data from these sources, we merged them to create a unified dataset, combining player attributes from the FC Ratings dataset with their corresponding transfer market values from Transfermarkt. However, due to matching challenges during the merging process, the final dataset includes 44,864 players with complete FC attributes and transfer market data. This refined dataset forms the basis for training our models to predict transfer market values effectively.

Since no established method exists to address this problem, our initial approach involves applying basic machine learning algorithms such as K-NN, SVM, Linear Regression, and Random Forest. By evaluating the performance and behavior of these algorithms, we aim to assess whether machine learning techniques can effectively solve this problem. Preliminary results from these algorithms are presented in the experimental results section. To achieve more robust and accurate predictions, our primary approach involves training a deep neural network model.

Deep learning reduces the need for extensive feature engineering by extracting meaningful representations from complex and high-dimensional data. Its ability to model nonlinear relationships and feature interactions makes it highly effective in capturing intricate factors that influence transfer fees, such as identifying which attributes are more significant for players based on their positions. Moreover, it can leverage large datasets to detect evolving transfer trends over time, as seen in the consistent rise in average market prices for players due to the expanding football economy. In areas where traditional machine learning algorithms may struggle, deep learning's capability to handle such complexities makes it a promising solution. Therefore, we plan to prioritize deep learning in our future efforts to address this task.

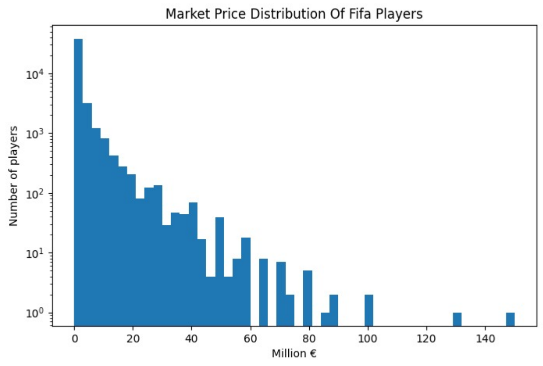


Figure 1. Market Price Distribution of Fifa Players

3.2. Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction technique used to simplify complex datasets while preserving as much variability as possible. It works by identifying the directions, or "principal components," in which the data varies the most, and these components are orthogonal (uncorrelated). The first principal component accounts for the largest variance, the second for the next largest, and so on. By projecting the data onto these components, PCA reduces the number of dimensions, making it easier to analyze and visualize, while minimizing information loss. It is commonly used in preprocessing for machine learning, noise reduction, and exploratory data analysis.

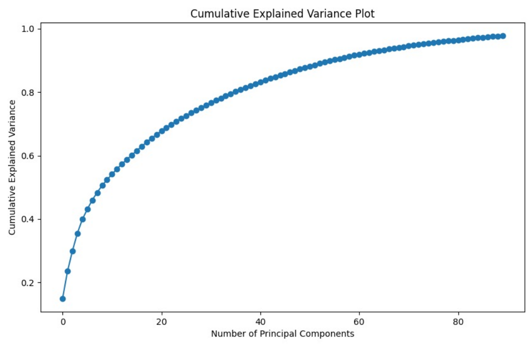


Figure 2. Cumulative Explained Variance by Number of Principal Components

4. Experimental Results

Dataset Description: The dataset contains 44,864 player and 109 features. Most of the features have integer values, however there are some features which are categorical like positions, nation etc. After future selection, one hot encoding and ordinal classification we got ... feautre. After our experiments we got these results.

Table 1. Performance of different Machine Learning Algorithms

Model	R^2	MSE(1e13)	MEA(1e6)
KNN	0.47	1.30	1.49
Random Forest	0.69	0.75	1.02
Linear Regression	0.30	1.72	2.16
SVM	0.66	0.82	1.19
Best NN	0.70	0.72	1.01
XGBoost	0.70	0.70	0.96
AutoML	0.69	0.77	1.02

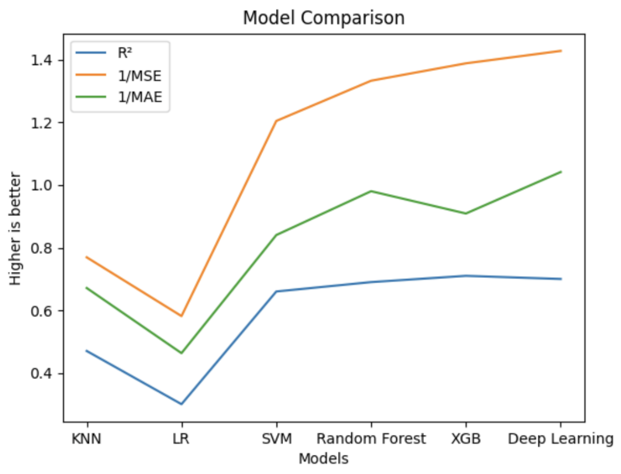


Figure 3. Complex model structure

4.1. SVR

Support Vector Regression (SVR) is a machine learning algorithm that aims to predict continuous values by finding a hyperplane in a high-dimensional space that best fits the data. Unlike traditional regression methods, SVR seeks to minimize prediction errors within a specified margin, effectively balancing model complexity and prediction accuracy. This approach is particularly useful for handling datasets with non-linear relationships, as it employs kernel functions to transform the data into a higher-dimensional space where linear regression can be applied.

In our implementation, we utilized the Radial Basis Function (RBF) kernel, which is well-suited for capturing complex, non-linear relationships in the data. SVR is also known for its robustness against outliers and its ability to generalize well, especially when the dataset is not excessively large. Additionally, we chose SVR because of its ability to maintain high performance on high-dimensional data, making it a suitable candidate for our problem.

To fine-tune the model, we experimented with different values of the regularization parameter C (1, 10, and 100). The best performance, in terms of both training time and R^2 score, was achieved with C = 10 . When C was increased from 10 to 100, the improvement in performance was negligible relative to the additional computational cost, which led us to suspect potential overfitting at higher values of C . With C = 10 , the model achieved and R^2 score of 0.66, a Mean Absolute Error (MAE) of 1.19×10 , and a Mean Squared Error (MSE) of 0.82×10^{13} , demonstrating a good balance between accuracy and computational efficiency.

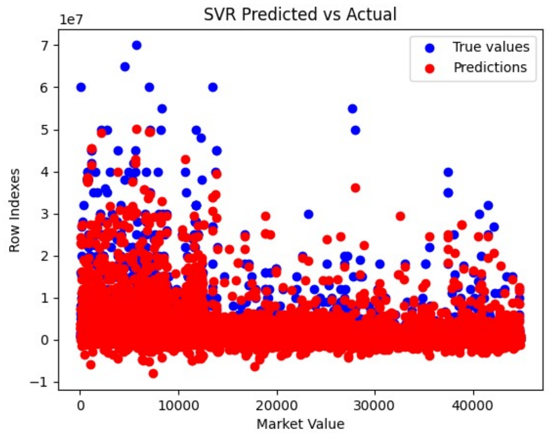


Figure 4. SVR Predicted vs. Actual Market Values

4.2. KNN

We decided to try the K-Nearest Neighbors (KNN) algorithm due to its straightforward logic and foundational role in machine learning. Initially, when applied to high-dimensional data, the algorithm produced very poor results, which was expected given KNN's limitations in handling high-dimensional spaces effectively. To address this, we implemented Principal Component Analysis (PCA) to reduce the dimensionality while retaining the most important features of the data. After applying PCA, the performance of KNN improved, achieving an R^2 score of 0.47, an MSE of 1.3×10^{13} , and an MEA of 1.49×10 . While these results were not the best in our experiments, they represent the third-best performance after Random Forest and SVR, which significantly outperformed KNN.

4.3. Random Forest

Random Forest is a machine learning algorithm that operates by constructing multiple decision trees and making final predictions based on the majority vote or the average output of these trees. This ensemble method offers several key advantages: it reduces overfitting, delivers higher accuracy compared to a single decision tree by lowering variance, and processes large datasets efficiently. In our experiments, Random Forest demonstrated the best performance among all tested algorithms.It achieved an R^2 score of 0.69, a Mean Squared Error (MSE) of 0.75×10^{13} , and a Mean Absolute Error (MEA) of 1.02×10. This indicates that the algorithm's predictions deviated, on average, by approximately 1 million euros—an acceptable margin of error in the context of the football industry, where transfer fees are often in the ten to one hundred millions.

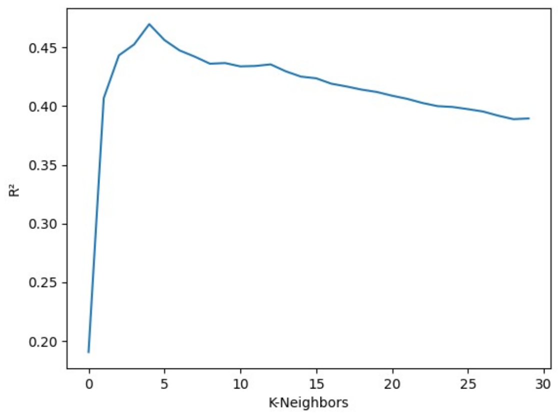


Figure 5. Knn R^2 scores with 10 PCA

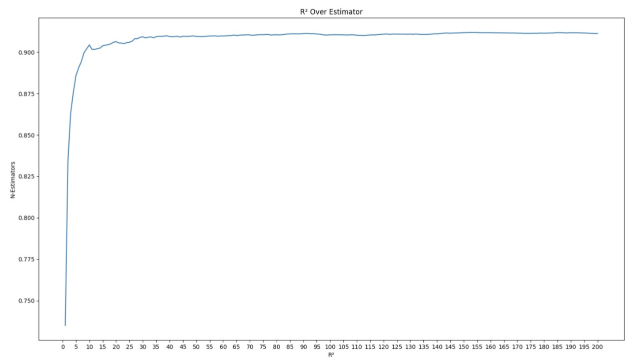


Figure 7. R^2 Score vs. Number of Estimators in the Random Forest Model

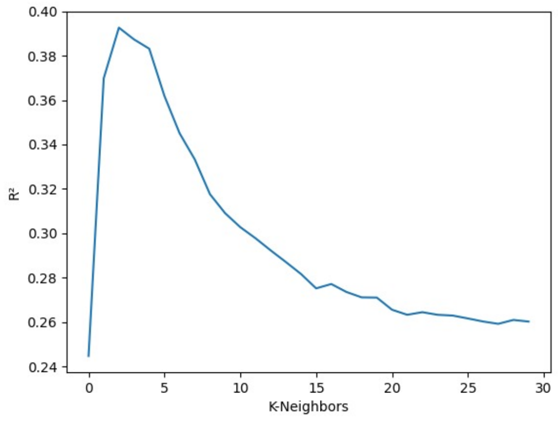


Figure 6. Knn R^2 scores with 60 PCA

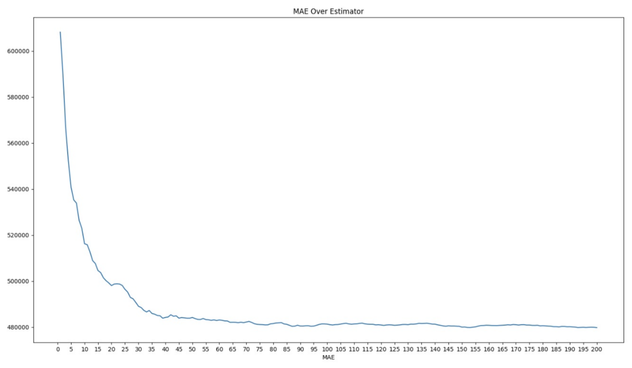


Figure 8. MAE (Mean Absolute Error) vs. Number of Estimators in the Random Forest Model

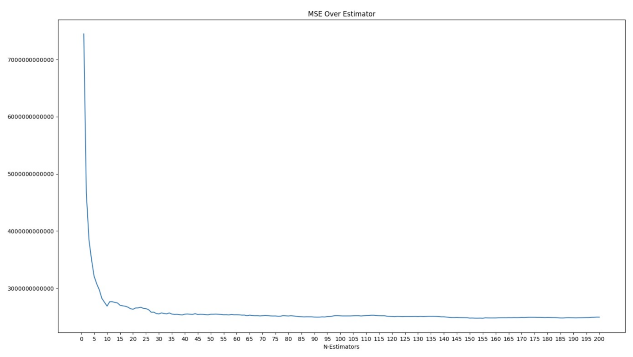


Figure 9. MSE (Mean Squared Error) vs. Number of Estimators in the Random Forest Model

4.4. Linear Regression

We decided to try linear regression, hoping it would capture the linear relationships in our dataset. This method is not the best for high-dimensional data, but it performs better than the K-NN algorithm, since the closeness in space is not the main principle of linear regression, unlike in K-NN. The initial results were poor, just like those of K-NN, so we attempted to use the PCA method with it. Our linear regression model achieved its best results with 70 principal components, which explain nearly 96% of the variance in our dataset. The R² value was 0.30. The result is worse than that of K-NN, and we believe the main reason for this is linear regression's inability to explain non-linear relationships, unlike K-NN models. The MSE was 1.72×10^{13}, and the MAE was 2.16×10.

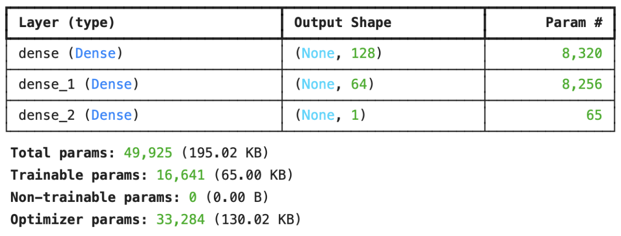
4.5. Neural Network

Neural Networks are machine learning algorithms inspired by the human brain, processing data through layers: an input layer, hidden layers, and an output layer. In our experiments, we gained valuable insights into their capabilities. However, while the model surpassed SVR by 0.02 R² points and secured second place, it did not fully meet our expectations.

We tested various architectures, both simple and complex, but observed no significant differences. This led us to question whether our data was too simple, the hyperparameters poorly tuned, or the problem inherently straightforward.

The training R² was around 0.85, while the test R² dropped to 0.68, suggesting potential overfitting. We attempted regularization techniques like dropout and early stopping, but they did not yield the improvements we anticipated. In addition to this we have tried l1 l2 regularization with our mid complex model but it makes our results much worse training R² was around 0.55 and test R² was around 0.45.

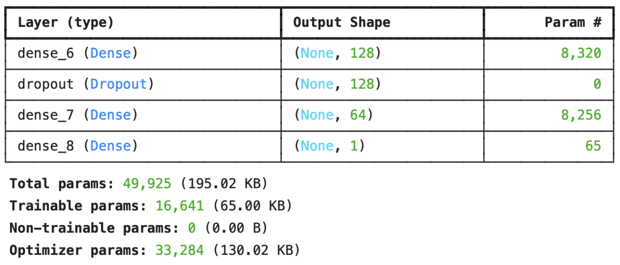
We employed the Adam optimizer with ReLU as the activation function for the layers. The initial learning rate was set to 0.001, and additional experiments were conducted with alternative learning rates such as 0.0001 and 0.01. The default learning rate of 0.001 yielded the best results in terms of performance. The mean squared error (MSE) was utilized as the loss function, while model performance was evaluated using metrics such as R², RMSE, MAE, and MSE. The epoch count was set to 300 to ensure sufficient training, though the use of early stopping in most experiments effectively mitigated the potential drawbacks of a high epoch count, allowing for adaptive termination based on performance improvement.



Layer (type)	Output Shape	Param #
dense (Dense)	(None, 128)	8,320
dense_1 (Dense)	(None, 64)	8,256
dense_2 (Dense)	(None, 1)	65

```
Total params: 49,925 (195.02 KB)
Trainable params: 16,641 (65.00 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 33,284 (130.02 KB)
```

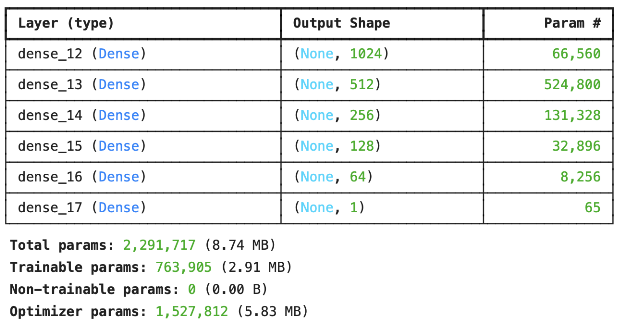
Figure 10. Basic model structure



Layer (type)	Output Shape	Param #
dense_6 (Dense)	(None, 128)	8,320
dropout (Dropout)	(None, 128)	0
dense_7 (Dense)	(None, 64)	8,256
dense_8 (Dense)	(None, 1)	65

```
Total params: 49,925 (195.02 KB)
Trainable params: 16,641 (65.00 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 33,284 (130.02 KB)
```

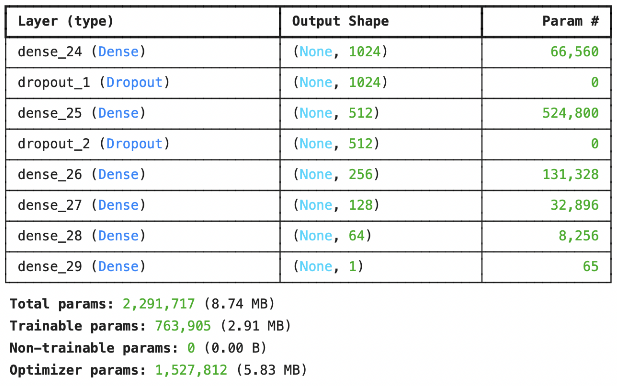
Figure 11. Basic model and dropout structure



Layer (type)	Output Shape	Param #
dense_12 (Dense)	(None, 1024)	66,560
dense_13 (Dense)	(None, 512)	524,800
dense_14 (Dense)	(None, 256)	131,328
dense_15 (Dense)	(None, 128)	32,896
dense_16 (Dense)	(None, 64)	8,256
dense_17 (Dense)	(None, 1)	65

```
Total params: 2,291,717 (8.74 MB)
Trainable params: 763,905 (2.91 MB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 1,527,812 (5.83 MB)
```

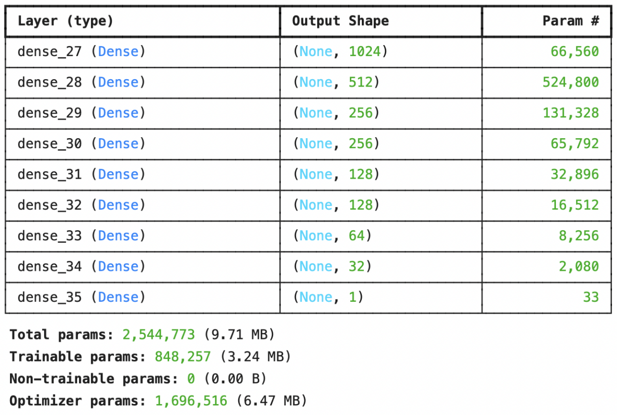
Figure 12. Mid complex model structure



Layer (type)	Output Shape	Param #
dense_24 (Dense)	(None, 1024)	66,560
dropout_1 (Dropout)	(None, 1024)	0
dense_25 (Dense)	(None, 512)	524,800
dropout_2 (Dropout)	(None, 512)	0
dense_26 (Dense)	(None, 256)	131,328
dense_27 (Dense)	(None, 128)	32,896
dense_28 (Dense)	(None, 64)	8,256
dense_29 (Dense)	(None, 1)	65

```
Total params: 2,291,717 (8.74 MB)
Trainable params: 763,905 (2.91 MB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 1,527,812 (5.83 MB)
```

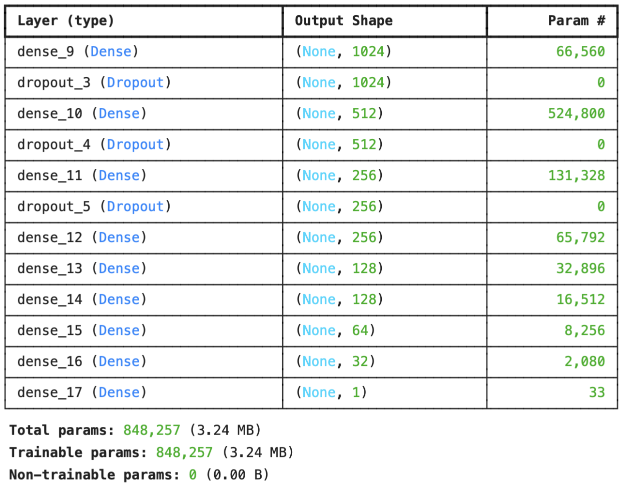
Figure 13. Mid complex model with dropout structure



Layer (type)	Output Shape	Param #
dense_27 (Dense)	(None, 1024)	66,560
dense_28 (Dense)	(None, 512)	524,800
dense_29 (Dense)	(None, 256)	131,328
dense_30 (Dense)	(None, 256)	65,792
dense_31 (Dense)	(None, 128)	32,896
dense_32 (Dense)	(None, 128)	16,512
dense_33 (Dense)	(None, 64)	8,256
dense_34 (Dense)	(None, 32)	2,080
dense_35 (Dense)	(None, 1)	33

```
Total params: 2,544,773 (9.71 MB)
Trainable params: 848,257 (3.24 MB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 1,696,516 (6.47 MB)
```

Figure 14. Complex model structure



Layer (type)	Output Shape	Param #
dense_9 (Dense)	(None, 1024)	66,560
dropout_3 (Dropout)	(None, 1024)	0
dense_10 (Dense)	(None, 512)	524,800
dropout_4 (Dropout)	(None, 512)	0
dense_11 (Dense)	(None, 256)	131,328
dropout_5 (Dropout)	(None, 256)	0
dense_12 (Dense)	(None, 256)	65,792
dense_13 (Dense)	(None, 128)	32,896
dense_14 (Dense)	(None, 128)	16,512
dense_15 (Dense)	(None, 64)	8,256
dense_16 (Dense)	(None, 32)	2,080
dense_17 (Dense)	(None, 1)	33

```
Total params: 848,257 (3.24 MB)
Trainable params: 848,257 (3.24 MB)
Non-trainable params: 0 (0.00 B)
```

Figure 15. Complex model with dropout structure

The performance of three neural network models—Basic Model, Complexer Model, and Most Complex Model—was evaluated using R^2, MAE, and MSE metrics under various configurations that incorporated E (early stopping) and D (dropout). This evaluation aimed to uncover how increasing model complexity and introducing regularization techniques affect predictive accuracy and overall performance. As the complexity of the model increases, transitioning from the Basic Model (128-64-1) to the Most Complex Model (1024-512-256-256-128-128-64-1), a consistent improvement in performance is observed. The Most Complex Model stands out with the best results due to its superior ability to capture intricate relationships and patterns within the data, making it highly effective for complex tasks. On the other hand, the Basic Model, with its simpler architecture, struggles to achieve competitive performance, reflected in its lower R^2 scores and higher MAE/MSE values, indicating its limited capacity to generalize to more complex datasets.

The introduction of dropout regularization techniques, specifically E (early stopping) and D (dropout), further enhances performance across all model architectures. Dropout works by randomly deactivating neurons during training, which reduces overfitting and forces the network to generalize better, particularly in deeper architectures like the Most Complex Model. This model benefits significantly from the inclusion of dropout, achieving the lowest MAE and MSE values while also recording the highest R^2 scores. The Basic Model, despite its computational simplicity, underperforms in all metrics, highlighting its unsuitability for tasks that demand high accuracy and robust predictive capabilities.

Although the Most Complex Model with dropout delivers the best results, it comes with a higher computational cost due to its deep architecture. For scenarios where computational resources are limited, the Complexer Model with dropout offers a viable alternative, balancing performance and efficiency effectively. The Complexer Model, while not as powerful as the Most Complex Model, achieves reasonable predictive performance and maintains lower computational requirements, making it suitable for environments where resources are constrained.

Overall, the findings emphasize the importance of balancing model complexity with appropriate regularization techniques to achieve optimal generalization and robust performance. The Most Complex Model with dropout emerges as the ideal choice for applications requiring high accuracy and the ability to model intricate relationships. However, in situations where computational efficiency is a priority, the Complexer Model with dropout provides a practical and effective solution. This analysis highlights the trade-offs between model complexity, accuracy, and computational demands, offering valuable insights for selecting the right architecture based on specific needs and constraints.

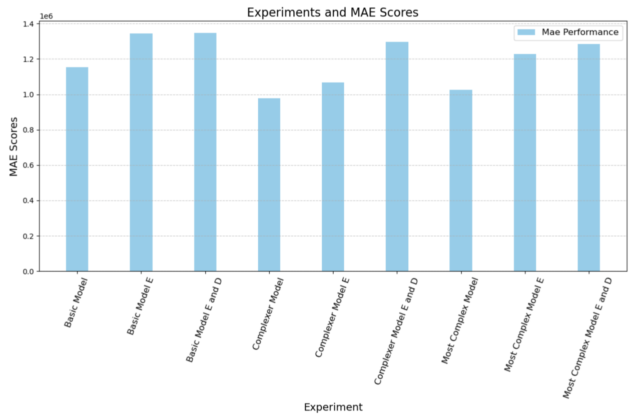


Figure 16. Different Models Mae Scores

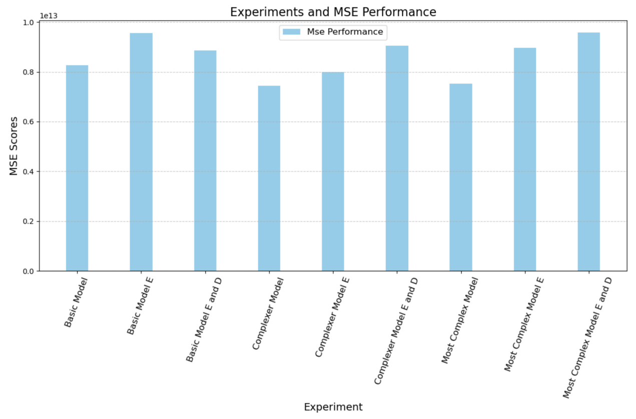


Figure 17. Different Models MSE scores

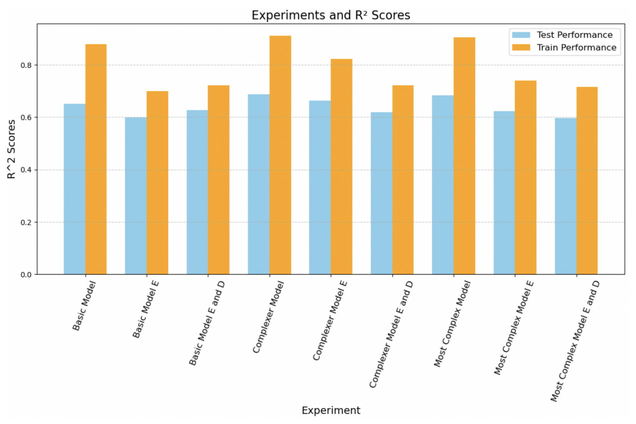


Figure 18. Different Models R^2 scores

4.6. XGBoost

XGBoost (Extreme Gradient Boosting) is a powerful machine learning algorithm based on gradient boosting, designed to enhance prediction accuracy and computational efficiency. It constructs an ensemble of decision trees sequentially, where each new tree aims to correct the errors of the previous ones. Unlike traditional gradient boosting, XGBoost incorporates advanced features such as regularization, parallel processing, and efficient handling of missing values, making it highly effective for both classification and regression tasks. Final predictions are made by aggregating the outputs of all trees, typically using a weighted sum.

In our experiments, XGBoost emerged as one of the best-performing models. The optimal configuration included parameters such as objective: squared error, max depth: 10, eta (learning rate): 0.04, subsample: 0.9, colsample bytree: 0.9, and eval metric: mean absolute error. This configuration resulted in an R^2 score of 0.7, an MSE of 0.7×10^{13}, and an MAE of 0.96×10 . Considering all evaluated parameters and metrics, XGBoost stands out as the best model in our experiments, delivering superior performance and accuracy.

4.7. H2O Library

When researching neural networks, we came across H2O, an open-source machine learning platform designed for fast and efficient model development, particularly suited for big data and distributed systems. H2O provides a comprehensive range of algorithms, including Gradient Boosting Machines (GBM), Random Forests, Generalized Linear Models (GLM), XGBoost, and neural networks, making it highly versatile for both classification and regression tasks. One of the platform's standout features is H2O AutoML, which simplifies the process of automated model development by performing tasks such as hyperparameter optimization, model evaluation, and algorithm selection with minimal user intervention. This functionality allows even less experienced users to harness the power of advanced machine learning models effectively.

After discovering the H2O library, we were excited to test it on our project to see how well it could handle our data and whether it could outperform other algorithms. H2O AutoML's automated approach and ability to combine multiple machine learning algorithms made it an attractive option for our experiments. Despite its sophisticated optimization processes and ensemble techniques, the results indicated that our dataset posed certain challenges that prevented the R^2 score from exceeding 0.7. The H2O AutoML model achieved an R^2 score of 0.68 on the test set, which was comparable to the results obtained with other algorithms, such as neural networks, XGBoost, and Random Forest.

Key	Value
Stacking strategy	cross_validation
Number of base models (used / total)	12 / 30
# GBM base models (used / total)	6 / 13
# XGBoost base models (used / total)	4 / 9
# DRF base models (used / total)	1 / 2
# DeepLearning base models (used / total)	1 / 5
# GLM base models (used / total)	0 / 1
Metalearner algorithm	GLM
Metalearner fold assignment scheme	Random
Metalearner nfolds	5
Metalearner fold_column	None
Custom metalearner hyperparameters	None

Table 2. Model Summary for Stacked Ensemble

5. Conclusion

This study demonstrates the potential of machine learning and deep learning techniques for predicting football players' market values based on comprehensive datasets combining FC Ratings and Transfermarkt data. A variety of models, including KNN, linear regression, support vector regression, random forest, XGBoost, neural networks, and H2O AutoML, were evaluated to identify the most effective approach for this complex task. Among these, XGBoost, neural networks, and random forest models emerged as the top performers, with the highest R² scores (0.7) and lowest error rates, highlighting their ability to capture complex patterns and nonlinear relationships in the data.

Dimensionality reduction techniques such as PCA significantly improved the performance of simpler models like KNN and linear regression by handling the high-dimensional nature of the dataset. Regularization techniques such as dropout and early stopping in neural networks helped mitigate overfitting, though the improvements were modest. The H2O AutoML platform provided an efficient way to combine multiple algorithms but showed limited improvement over standalone models, suggesting that the dataset's characteristics set an upper limit for predictive accuracy.

While advanced machine learning models demonstrated strong performance, their computational costs varied significantly, emphasizing the importance of selecting models based on the available resources and desired trade-offs between accuracy and efficiency. The findings also underscore the potential for further improvements through advanced feature engineering, hyperparameter optimization, and incorporating additional data sources.

Overall, this research provides valuable insights for clubs, analysts, and stakeholders in the football industry by offering a data-driven approach to evaluate player valuations. Future work could focus on refining these models by incorporating players' health-related conditions, adding inflation and time-related features, and considering time-dependent economic factors that might influence transfer fees. Additionally, integrating real-time performance metrics and exploring advanced ensemble methods could further enhance the predictive capabilities and applicability of these models in dynamic and evolving market scenarios.

Our GitHub repository for this project is available at: GitHub Link

References

Behravan, I. and Razavi, S. M. A novel machine learning method for estimating football players' value in the transfer market. *Soft Computing*, 25(3):2499–2511, Feb 2021. ISSN 1433-7479. doi: 10.1007/s00500-020-05319-3. URL `https://doi.org/10.1007/s00500-020-05319-3`.

Kenny, G. The world's most popular sports: Ranking and estimated fan numbers. *SportsNewsIreland*, November 2 2023.

Kirschstein, T. and Liebscher, S. Assessing the market values of soccer players – a robust analysis of data from german 1. and 2. bundesliga. *Journal of Applied Statistics*, 46(7):1336–1349, 2019. doi: 10.1080/02664763.2018.1540689. URL `https://doi.org/10.1080/02664763.2018.1540689`.

Liu, G., Luo, Y., Schulte, O., and Kharrat, T. Deep soccer analytics: Learning an action-value function for evaluating soccer players. *Data Mining and Knowledge Discovery*, 34(5):1531–1559, 2020. doi: 10.1007/s10618-020-00705-9.

McHale, I. G. and Holmes, B. Estimating transfer fees of professional footballers using advanced performance metrics and machine learning. *European Journal of Operational Research*, 306(1):389–399, 2023. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2022.06.033. URL `https://www.sciencedirect.com/science/article/pii/S0377221722005082`.