COMPARATIVE STUDY OF MACHINE LEARNING MODELS

FOR PREDICTING THE MARKET VALUE OF

PROFESSIONAL FOOTBALL PLAYERS

A Thesis Presented to the Graduate School

of Fort Hays State University in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Science

2025

BY

ÁLVARO SALVADOR LÓPEZ

B.S., Fort Hays State University, 2022

Accepted in partial fulfillment of the requirements for

the degree of Master of Science in Computer Science

in the Graduate School of Fort Hays State University

JUNE 29, 2025

Approved by:

Dr. Hong Biao Zeng, Chair of the Defense Committee

Signature: Hongbiao Zeng

Date: ______07/29/2025_________

Dr. Angela Pool-Funai, Dean of the Graduate School

Signature: ____________________________

Date: ___07/30/2025____________

ABSTRACT

The market value of professional football players is a critical factor in decision-making for clubs, agents, and analysts. Accurate player valuation impacts transfers, contract negotiations, and financial planning. In recent years, data-driven approaches have emerged to support traditional scouting with predictive analytics. This thesis presents a comparative study of machine learning models to estimate the market value of football players based on historical performance and personal attributes.

This thesis presents a comparative study of two independently developed machine learning systems designed to predict the market value of football players for the 2020–2021 season. Both systems were trained using real data collected from Kaggle: one dataset covering the seasons 2017–2020, and another containing actual market values for the 2020–2021 season. These datasets combine player statistics and performance indicators sourced from Transfermarkt and FBref.

In Project 1, ensemble algorithms such as Random Forest, XGBoost, and Support Vector Regression (SVR) were trained and compared. In Project 2, alternative models including CatBoost, K-Nearest Neighbors (KNN), and Gradient Boosting were explored. Each system selects its best-performing model based on Root Mean Square Error (RMSE) and is deployed through a custom-built Streamlit interface that supports interactive prediction and analysis.

Predicted values for 2020–2021 were then compared with actual market values from that season, allowing for quantitative evaluation of each project's accuracy. Finally, the results of both projects were compared and analyzed to determine which approach is more suitable for real-world application in football analytics.

This work demonstrates the feasibility of using machine learning for player valuation and provides a practical framework for evaluating multiple modeling strategies in sports data science.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my thesis advisor, Dr. Hongbiao Zeng, for his continuous support, guidance, and insightful feedback throughout the development of this project. His expertise and encouragement have been instrumental in shaping the direction and quality of my research.

I am also sincerely thankful to Dr. Anas Hourani and Dr. Hussam Ghunaim for serving on my thesis committee and for their valuable comments, which helped me strengthen both the technical and academic aspects of this work.

Special thanks to my family for their unconditional love, motivation, and support throughout my academic journey. I also extend my appreciation to my teammates, friends, and my partner for their patience and encouragement during long hours of coding, debugging, and writing.

Finally, I would like to thank the open-source community and the contributors of Kaggle datasets, particularly those who made available the football performance and market value data that served as the foundation for this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviation	**Definition**
AI	Artificial Intelligence
API	Application Programming Interface
CSV	Comma-Separated Values
EDA	Exploratory Data Analysis
FBRef	Football Reference (statistical database for football)
FHSU	Fort Hays State University
Kaggle	Online platform for data science datasets and competitions
KNN	K-Nearest Neighbors
MAE	Mean Absolute Error
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
SHAP	SHapley Additive exPlanations

Abbreviation	**Definition**
Streamlit	Python framework for building interactive web applications
SVR	Support Vector Regression
TMkt	Transfermarkt (online platform with football player market values)
XGBoost	Extreme Gradient Boosting
LGBM	Light Gradient Boosting Machine (only used during development phase)

Chapter 1 INTRODUCTION

1.1 Background

The global football industry has witnessed exponential financial growth over the past decades, with player transfers and market valuations becoming increasingly influential in both sports and business domains. Estimating a football player's market value is no longer solely based on subjective judgment or negotiation tactics; rather, it is now supported by data-driven approaches that can analyze a wide range of performance indicators, biographical data, and contextual factors.

Machine learning (ML) has emerged as a powerful tool for predictive modeling in sports analytics. With the availability of large-scale datasets detailing player statistics, playing styles, and transfer histories, ML offers a systematic way to evaluate and predict the market value of players with improved accuracy and objectivity. These techniques have the potential to assist scouts, analysts, and clubs in making informed decisions, minimizing financial risks, and discovering undervalued talent.

Several online platforms such as Transfermarkt and FBRef collect and publish extensive statistics on professional football players. These databases include not only current and historical market values but also detailed metrics such as minutes played, goals, assists, xG (expected goals), and many other performance-related features. Publicly available datasets derived from these platforms — such as those hosted on Kaggle — make it possible to apply machine learning models in an academic and research setting.

This thesis leverages such data to develop predictive systems that estimate players' market values for the 2020–21 season, based on performance metrics from the preceding years (2017 to 2020). Two independent machine learning projects are constructed, each using a distinct group of algorithms. In both cases, multiple models are trained and evaluated, and the one with the best performance is selected for deployment. Finally, the results from both projects are compared to derive insights into which modeling strategy proves more effective for this task.

1.2 Problem Statement

In the current landscape of professional football, clubs invest millions of euros in player transfers, often relying on internal assessments, market trends, or subjective valuations to determine a player's worth. However, these methods are not always consistent or transparent, and they can result in overvalued or undervalued transfers, financial losses, or missed opportunities.

This thesis addresses the need for a data-driven solution by formulating the problem of player valuation as a regression task, where the goal is to predict a player’s market value based on objective historical performance data. The main challenge lies in the high variability of football-related data: player performance is influenced by numerous factors such as team quality, league competitiveness, age, injuries, and tactical roles — making prediction a non-trivial task.

To tackle this problem, two separate machine learning systems are developed. Each project trains and evaluates different sets of algorithms using player data from the 2017–2020 seasons. The objective is not only to build accurate predictors, but also to analyze and compare the performance of different ML approaches, providing insight into their relative effectiveness for football player valuation.

The ultimate aim is to determine which modeling strategy — based on algorithm selection, data processing, and prediction performance — is better suited for predicting realistic market values, as measured against real data from the 2020–21 season.

1.3 Objectives

The main objective of this thesis is to build and evaluate two independent machine learning systems capable of predicting the market value of professional football players. Each system will be based on a different set of algorithms, and both will be tested against real-world data to assess accuracy and applicability.

The specific objectives of the research are as follows:

1. To collect and preprocess real-world football player data from trusted public sources (Kaggle datasets), specifically:
    - 2017–2020 player performance and value data for training.
    - 2020–21 season data for validation and comparison of predictions.
2. To design and implement two standalone ML-based prediction systems, each comprising:
    - Data preparation and feature engineering pipelines.
    - Training of multiple machine learning regression models.

- Selection and saving of the best-performing model based on RMSE (Root Mean Square Error).

3. To evaluate model performance within each project by comparing predicted values against real 2020–21 market values.
4. To compare the best model from each project in terms of:
    1. Accuracy (RMSE).
    2. Generalization capability.
    3. Practical viability and computational cost.
5. To draw insights from the comparison, identifying which machine learning strategy proves more effective for player market value estimation.
6. To present findings through an interactive Streamlit application in each project, allowing users to explore model predictions and compare them to real data.

These objectives aim not only to improve the accuracy of player valuation but also to demonstrate the usefulness of comparative ML experimentation for practical decision-making in the sports industry.

1.4 Scope of the Study

This thesis focuses on the development and evaluation of two separate machine learning systems designed to estimate the market value of professional football players. The scope of the study is defined by the following key boundaries and considerations:

- Temporal Scope: The training data used in both projects covers three football seasons — 2017/18, 2018/19, and 2019/20 — while the prediction target is the 2020/21 season. Real market values from the 2020/21 season are used to evaluate model performance.

- Geographical Scope: The dataset includes players from multiple top European leagues, covering a broad range of nationalities and clubs. However, it is limited to the leagues and players present in the source datasets, and it may not be globally comprehensive.

- Data Sources: The study uses two publicly available datasets from Kaggle:

    - Soccer Players Values and Their Statistics (2017–2020) [1]

    - Football Players Market Value Prediction (2020–21) [2]

- Machine Learning Scope:

    - Project 1 explores traditional and ensemble methods such as Random Forest, XGBoost, and Support Vector Regression (SVR).

    - Project 2 evaluates alternative and modern techniques including CatBoost, K-Nearest Neighbors (KNN), and Gradient Boosting Regressor.

    - Each project selects the best-performing model based on RMSE and uses it for predictions.

- Evaluation and Comparison: The thesis includes a final analysis comparing the best model from each project to determine which approach is more effective and robust for real-world applications.

- Limitations:

    - The models predict only a single market value (for the 2020–21 season), without accounting for future market trends.

    - External factors such as injuries, transfers, or economic crises (e.g., COVID-19 impact) are not explicitly modeled.

    - The real value data used for validation may contain estimations, as it reflects market valuations rather than actual transaction prices.

In summary, this study is limited to historical data analysis and comparative model evaluation within the football domain. It does not aim to forecast future values beyond the 2020–21 season but rather to establish a reliable methodological foundation for player valuation using machine learning.

1.5 Structure of the Thesis

This thesis is structured into six main chapters, each contributing to a comprehensive understanding of the research objective, methodology, and findings:

- Chapter 1: Introduction

  Introduces the background and motivation for the study, defines the research problem and objectives, and outlines the scope and structure of the thesis.

- Chapter 2: Literature Review

  Surveys relevant literature on football player valuation, machine learning in sports analytics, and commonly used regression models for prediction tasks. This chapter establishes the theoretical and methodological foundations for the projects developed.

- Chapter 3: Methodology

Describes the datasets, preprocessing techniques, model selection process, evaluation metrics, and system architecture for both projects. It also explains how the models were trained, tested, and compared using RMSE as the primary performance metric.

- Chapter 4: System Implementation

Provides a detailed overview of the technical implementation of both Project 1 and Project 2. It covers the modular architecture, key components, interfaces, and development tools used, including Python, Streamlit, and relevant machine learning libraries.

- Chapter 5: Results and Discussion

Presents and analyzes the results obtained from both systems, compares the best-performing models from each project, and interprets the findings. It discusses the strengths, limitations, and practical implications of the results.

- Chapter 6: Conclusion and Future Work

Summarizes the main contributions of the thesis, draws final conclusions, and proposes directions for future improvements, such as integrating transfer history, contract details, or real-time updates into player valuation systems.

Each chapter builds upon the previous one to form a complete and logical progression from problem definition to solution and final reflection. The thesis concludes with a list of references and appendices containing supplemental materials such as screenshots, sample outputs, and source code descriptions.

Chapter 2 LITERATURE REVIEW

In recent years, the integration of machine learning (ML) into sports analytics has significantly transformed the way data is used to understand player performance, predict match outcomes, and assess the economic value of athletes. Football, in particular, has emerged as a fertile domain for predictive modeling, due to the wide availability of detailed player statistics, match data, and market valuations provided by platforms such as Transfermarkt and FBRef.

This literature review aims to provide a structured overview of previous research relevant to the development of systems that estimate the market value of professional football players using machine learning techniques. The primary objective is to identify key trends, commonly used methodologies, and significant findings that have shaped this field in recent years. Additionally, this section highlights existing limitations and research gaps that this thesis seeks to address.

To guide the discussion, the review is organized into five thematic sections:

1. Machine Learning in Sports – An overview of how ML has been applied across sports disciplines, particularly in performance analysis and tactical decision-making.

2. Regression Models in Market Valuation – A focus on regression-based approaches for predicting financial indicators, including both traditional econometric and modern ML models.

3. Data-Driven Valuation in Football – Studies specifically targeting the estimation of football players' market value using data-driven approaches.

4. Comparative Studies & Methodological Approaches – Research comparing multiple ML models and strategies within sports analytics, including feature selection, evaluation metrics, and interpretability.

5. Gaps Identified for This Thesis – A synthesis of the main limitations observed in prior work, establishing the motivation and contribution of this thesis.

Together, these areas form the theoretical foundation for the two independent systems developed and compared in this thesis, each based on different machine learning algorithms and methodologies.

2.1 Machine Learning for Player Market Value Prediction

The application of machine learning (ML) in sports has experienced a remarkable expansion in recent years, fueled by the increasing availability of structured performance data, advances in computational resources, and the growing demand for data-driven decision-making across professional sports organizations. In this context, ML has become a fundamental tool for analyzing performance indicators, optimizing training, and developing predictive models that can inform coaching strategies, scouting decisions, and player health management.

In football, the most globally followed and data-rich sport, ML techniques have been widely applied to various problems including match outcome prediction, tactical analysis, player role classification, and more recently, injury prevention. One notable study is Predicting injury risk using machine learning in male youth soccer players [3], which demonstrates how ML models can be trained on physiological and training data to identify patterns that precede injury. While injury prediction is not directly related to economic valuation, the use of supervised learning to analyze player characteristics is conceptually aligned with market value estimation.

In the area of match prediction, the work A machine learning framework for sport result prediction [4] presents a modular approach that uses player and team statistics to forecast outcomes in competitive settings. The framework includes stages of data preprocessing, feature selection, and model evaluation, which are also key elements in valuation systems. Likewise, Predictive analysis and modelling football results using machine learning for the English Premier League [5] highlights the importance of model interpretability and the choice of performance metrics in football analytics. Although the main focus is on match outcomes, the methodology—especially the use of ensemble learning and regression—is transferable to player-centric prediction tasks.

Beyond football, ML has shown promise in sports like tennis. For instance, A study of forecasting tennis matches via the Glicko model [6] applies ranking-based systems and probabilistic models to improve prediction accuracy. The approach is conceptually similar to the modeling of latent value in footballers, where true player quality must be inferred from noisy and heterogeneous performance data.

All these studies demonstrate that machine learning is no longer used just for descriptive statistics or historical analysis—it is now a powerful tool for forecasting, classification, and strategic decision-making. In the context of this thesis, these insights support the viability of building predictive systems that estimate player market value based on performance data, age, position, and contextual attributes. They also underline the importance of a solid model validation process, including the use of historical data, cross-validation, and a robust evaluation metric like RMSE to assess predictive accuracy.

2.2 Regression Models in Market Valuation

Regression models are a fundamental component of predictive analytics, particularly when estimating continuous variables such as market value, salary, or performance metrics. In sports, and football specifically, regression has been widely applied to quantify the economic worth of players based on observable characteristics and historical data. The use of these models provides an interpretable and statistically grounded method to understand how individual variables—such as age, position, playing time, or team performance—influence player valuation.

Early studies focused on traditional linear regression techniques, which are easy to interpret and implement but often insufficient when handling complex, non-linear relationships. A clear example is found in Predicting Market Value of Football Players Using Machine Learning [5], where the authors compare linear regression, decision trees, and ensemble methods to estimate player value. Their findings highlight that although linear regression offers a useful baseline, it is often outperformed by more flexible algorithms capable of capturing non-linear interactions between features.

The importance of variable selection and data preprocessing is another key aspect in regression-based valuation models. In [7], the researchers emphasize that poorly selected or redundant features can negatively affect model accuracy. Therefore, the use of feature engineering, dimensionality reduction, and encoding strategies becomes essential when working with multi-source football datasets.

The parallel between real estate pricing and player valuation has also been explored in the literature. The article Proyecto Machine Learning: Predicción de Precios de Viviendas en Boston con Regresión [8] presents a regression-based approach to house price prediction that relies on variables such as location, size, and age—analogous to features like nationality, minutes played, and age in football. Although the domain is different, the methodological similarity is striking and illustrates the versatility of regression models across disciplines.

A more specialized contribution is found in Modelling Football Players Values and Their Determinants on a Transfer Market using Robust Regression Models [9], a thesis that uses Iteratively Reweighted Least Squares (IRLS) to model the value of players segmented by position. The author analyzes the influence of various factors—such as minutes played, nationality, goals, and age—on market value, building interpretable models for each positional category. While the approach is analytically sound, it relies on robust statistics rather than modern machine learning techniques and does not incorporate automated validation or model comparison frameworks.

More recently, research has shifted towards integrating advanced machine learning regressors to improve prediction accuracy. In Predicting the Value of Football Players: Machine Learning Techniques and Sensitivity Analysis [10], the author combines multiple data sources and applies algorithms such as Random Forest, Support Vector Machines, and XGBoost. The study introduces sensitivity analysis to evaluate the contribution of each feature, enhancing both model interpretability and transparency, an increasingly important aspect when applying ML in high-stakes environments such as player transfers.

In this thesis, regression models constitute the core predictive mechanism in both Project 1 and Project 2. Project 1 implements Random Forest, XGBoost, and SVR (Support Vector Regression), while Project 2 explores CatBoost, K-Nearest Neighbors (KNN), and Gradient Boosting. By evaluating all models using RMSE and holding consistent datasets and preprocessing steps, this research offers a fair and reproducible comparison of regression strategies applied to market value prediction in football.

2.3 Data-Driven Valuation in Football

The growing availability of player-level data in football—covering technical, physical, and contextual metrics—has enabled the emergence of data-driven approaches to player valuation. Moving beyond subjective assessments by scouts or purely economic models based on demand and media presence, recent studies have focused on quantifying player value through machine learning models trained on historical performance data and market trends.

One key contribution in this area is the thesis Modelling Football Players Values and Their Determinants on a Transfer Market using Robust Regression Models [11], which uses structured data from Transfermarkt and FBRef to build position-specific valuation models. The study emphasizes the role of domain knowledge in selecting appropriate features such as goals, minutes played, nationality, and age. The segmentation by player role (goalkeepers, defenders, midfielders, forwards) allows for more granular analysis, improving the interpretability and reliability of the regression outputs.

Building on that, Shen (2025) presents an advanced machine learning pipeline in Predicting the Value of Football Players: Machine Learning Techniques and Sensitivity Analysis [12], where the author constructs a large, customized dataset combining data from scouting reports, league statistics, and public valuation platforms. Algorithms such as Random Forest, SVM, and XGBoost are applied, and the inclusion of sensitivity analysis provides insight into which features most significantly impact predicted market value. The study demonstrates how data aggregation and model explainability can be balanced to develop accurate yet interpretable prediction systems.

Hill et al. (2025) contribute a complementary perspective in A Review of Football Player Metrics and Valuation Methods [13], offering a typological classification of valuation approaches. These range from statistical and econometric models to data-driven machine learning systems. The authors argue that while traditional models are often easier to interpret, they may fall short in capturing the multidimensional nature of player value, which includes not only technical output but also potential, tactical fit, and injury history. Their work supports the thesis that combining quantitative performance indicators with advanced modeling techniques is key to building robust valuation frameworks.

In all three studies, a common theme emerges: the importance of integrating multiple sources of information—performance stats, biographical data, contextual factors—and processing them through adaptable ML pipelines. These findings directly support the methodology adopted in this thesis, where player valuation is approached through data fusion, feature engineering, and comparative evaluation of multiple machine learning models. By using real-world football data from 2017 to 2020, the two systems developed in this work aim to replicate the type of predictive performance and interpretability demonstrated in recent academic contributions.

2.4. Comparative Studies & Methodological Approaches

One of the most critical aspects in applying machine learning to player valuation lies not only in choosing a suitable model but also in understanding how different algorithms behave under comparable conditions. Comparative studies in the sports analytics literature have played an essential role in identifying the strengths and limitations of various techniques, often using shared datasets, standardized evaluation metrics, and common feature sets.

The work Predicting Market Value of Football Players Using Machine Learning [7] exemplifies this approach by benchmarking multiple models—including linear regression, decision trees, and ensemble methods—on the same dataset. The study highlights the superior performance of ensemble learning techniques such as Random Forest, particularly in handling complex feature interactions and reducing overfitting. It also stresses the importance of selecting evaluation metrics aligned with the prediction goal—in this case, mean squared error and RMSE—to assess model accuracy objectively.

A similar methodology is adopted in Shen's 2025 study [12], where the author not only compares Random Forest, SVM, and XGBoost, but also applies sensitivity analysis to assess how model outputs vary with respect to different inputs. This provides an additional layer of interpretability and helps identify which features (e.g., age, goals, minutes) have the most influence on market value predictions. This multi-model, multi-metric evaluation framework is essential for robust research in player valuation, where ground truth is often subjective or imprecise.

In a broader methodological review, Hill et al. (2025) [13] analyze the different paradigms used in football valuation—from econometric modeling to black-box machine learning. They argue that while no single model offers a perfect solution, comparative analysis allows practitioners to balance accuracy, interpretability, and computational efficiency depending on the application. For example, simpler models may be preferred in scouting contexts where transparency is key, whereas more complex models may be better suited for internal club analytics with richer data access.

This thesis adopts a similar comparative framework. Two independent systems were developed: Project 1, which evaluates Random Forest, XGBoost, and SVR, and Project 2, which applies CatBoost, K-Nearest Neighbors (KNN), and Gradient Boosting. Both systems use the same underlying dataset (players from 2017–2020) and preprocessing steps, allowing for a controlled comparison. RMSE is employed as the primary evaluation metric across all models to ensure consistency.

Additionally, both projects integrate explainability elements—either through feature importance or SHAP values—to support interpretation and practical usability. This methodological alignment with current academic standards ensures that the comparative insights gained from this thesis are not only reliable but also relevant to real-world applications in football scouting, sports economics, and performance analysis.

2.5. Gaps Identified for This Thesis

Although the body of research on football player valuation using machine learning has expanded in recent years, several limitations and gaps persist in the current literature. These gaps relate not only to the models and features used, but also to evaluation practices, dataset consistency, and practical applicability of the developed systems.

First, many studies rely on limited datasets or fail to include multiple seasons, which weakens the generalizability of their findings. For instance, while [5] and [11] present robust valuation models, their experiments are based on static or position-specific datasets, which may not fully capture temporal trends or market fluctuations. This thesis addresses this issue by training and testing all models on a unified, multi-season dataset (2017–2020), allowing for a broader and more realistic evaluation of market dynamics.

Second, there is a lack of comprehensive comparative studies using consistent pipelines and validation strategies. Most existing works either focus on a single algorithm or use different preprocessing techniques and feature sets for each model. As noted in [12] and [13], this hinders fair comparison and limits reproducibility. In contrast, this thesis adopts a controlled design in which both projects use the same dataset, preprocessing logic, and evaluation metric (RMSE), enabling a transparent and valid comparison of model families.

Another recurring limitation is the insufficient attention given to model explainability. While some studies explore feature importance or conduct sensitivity analysis ([12]), many rely on black-box models without integrating interpretability tools. This restricts the adoption of ML-based valuation systems in real-world decision-making, where stakeholders require justification for predictions. In response, this thesis includes interpretable elements such as feature importance visualizations (when available) and a consistent evaluation metric (RMSE) to enhance trust and usability.

Finally, few academic works aim to produce fully functional, end-to-end systems for market value estimation. Most are prototypes or analyses performed in isolated environments, lacking an interface or real user interaction. By contrast, this thesis delivers two complete, deployable systems using Streamlit, allowing user input, retraining, and direct comparison with real market values. This bridges the gap between theoretical development and practical implementation.

These identified gaps shape the main contribution of this thesis: the development and comparison of two independent ML systems for football player valuation, designed with methodological rigor, transparency, and real-world applicability in mind.

Chapter 3 METHODOLOGY

3.1 Overview of the Approach

The main objective of this thesis is to develop machine learning systems capable of estimating the market value of professional football players based on their performance, physical characteristics, and contextual variables. Rather than relying on a single predictive model, the study proposes and compares two independent systems, Project 1 and Project 2, each built using different sets of machine learning algorithms and processing strategies.

This dual-system approach was chosen to allow for a fair and detailed comparison between model families and preprocessing methodologies. While both systems use the same underlying dataset, they differ in several technical aspects, including the selection of algorithms, missing value treatment, and data imputation strategies. These differences are intentional and serve to explore how modeling choices affect predictive performance in the context of football player valuation.

Project 1 focuses on ensemble and kernel-based models, namely Random Forest, XGBoost, and Support Vector Regression (SVR). It uses a straightforward preprocessing pipeline and evaluates the models using root mean squared error (RMSE). Project 2, on the other hand, includes CatBoost, K-Nearest Neighbors (KNN), and Gradient Boosting. It incorporates mean-based imputation and follows a similar training-evaluation-deployment structure, enabling direct comparison with Project 1.

Both systems were developed with usability in mind. Each includes a graphical user interface built with Streamlit that allows users to explore the data (EDA), retrain the models, and predict the current value of specific players. This hands-on interaction makes the systems accessible to non-technical stakeholders and increases the practical value of the research.

By implementing two independent but comparable solutions, the thesis aims to provide insight into which machine learning strategies are best suited for player market value prediction, and under what conditions.

3.2 Dataset Description and Preprocessing

The dataset used in this thesis was compiled from two publicly accessible sources: Transfermarkt and FBRef. These platforms provide structured and semi-structured information on professional football players, including market value estimates, personal attributes, and detailed performance statistics. The data spans three full seasons—2017/18, 2018/19, and 2019/20—and covers players from the top five European leagues: Premier League, LaLiga, Bundesliga, Serie A, and Ligue 1.

The raw data for each season was stored in separate CSV files and subsequently loaded using custom functions in the system's utility modules. A new column named year was added to each file to facilitate temporal tracking. Afterward, the seasonal datasets were concatenated into a single DataFrame, resulting in a unified dataset of several thousand player-season observations.

The initial dataset contained multiple features, including:

- Personal attributes (e.g., age, nationality, height, dominant foot)
- Contextual information (e.g., league, team, position)

- Match statistics (e.g., minutes played, goals, assists, expected goals, passes, defensive actions)
- The target variable: estimated market value in euros (converted to numeric format)

To ensure data quality and consistency, several preprocessing steps were performed:

- Observations with missing market value were removed.
- Redundant columns such as Attendance and Season were dropped if present.
- Categorical features were transformed using one-hot encoding (pd.get_dummies).
- All missing numerical values were handled differently across the two projects:
  - In Project 1, missing values were filled with zero (fillna(0)).
  - In Project 2, missing values were imputed using the mean strategy via SimpleImputer.

The processed dataset was then split into features (X) and target variable (y), where y represents the market value in euros. This preprocessing pipeline ensures that both models are trained and evaluated on consistent data structures, allowing for a fair and replicable comparison.

The final version of the dataset used for model training contains hundreds of engineered features after encoding, representing a comprehensive and high-dimensional description of each player's performance and characteristics.

3.2.1 Exploratory Data Analysis

Before training any predictive models, an exploratory data analysis (EDA) was performed to understand the structure, distribution, and relationships within the dataset. This step is essential for identifying potential data quality issues, guiding feature selection, and detecting patterns that may influence player valuation. The following visualizations were generated using seaborn and matplotlib and are accessible from the Streamlit application under the EDA tab.

Figure 1 – Pairplot of Numerical Variables

A pairplot matrix was generated to visualize the pairwise relationships between key numerical features, such as matches played, minutes, goals, assists, expected goals (xG), expected assists (xA), and total shots. The plots reveal several positive linear correlations, especially between minutes played and cumulative stats like goals, assists, or xG. Some features show strong co-linearity, suggesting that dimensionality reduction or regularization might help avoid redundancy during model training.

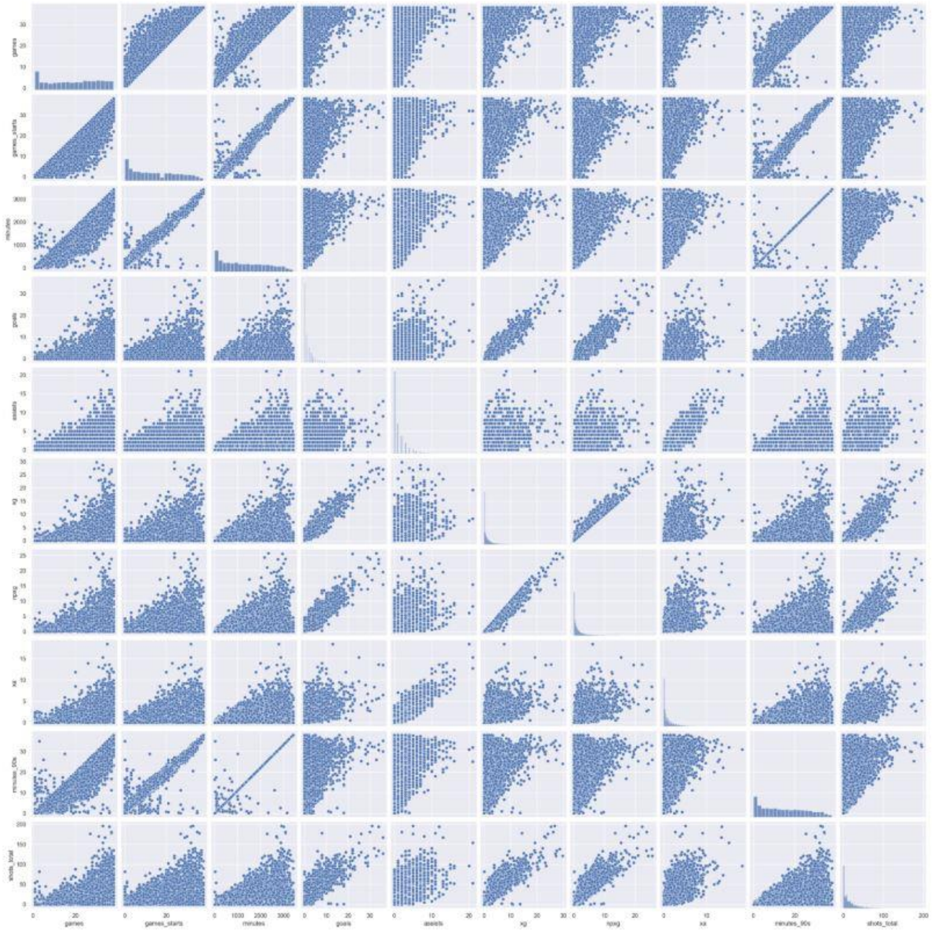

Figure 1.Pairplot of Key Numerical Variables: Visual correlation matrix between features such as games, minutes, goals, assists, xG, and xA.

Figure 2 – Age Distribution

The age histogram shows that most players in the dataset are between 20 and 30 years old, with a peak around age 25. The distribution approximates a normal shape with a slight right skew, indicating a smaller proportion of older players. This pattern reflects the natural age profile of active professionals in top European leagues and highlights age as a potentially influential variable in player valuation.

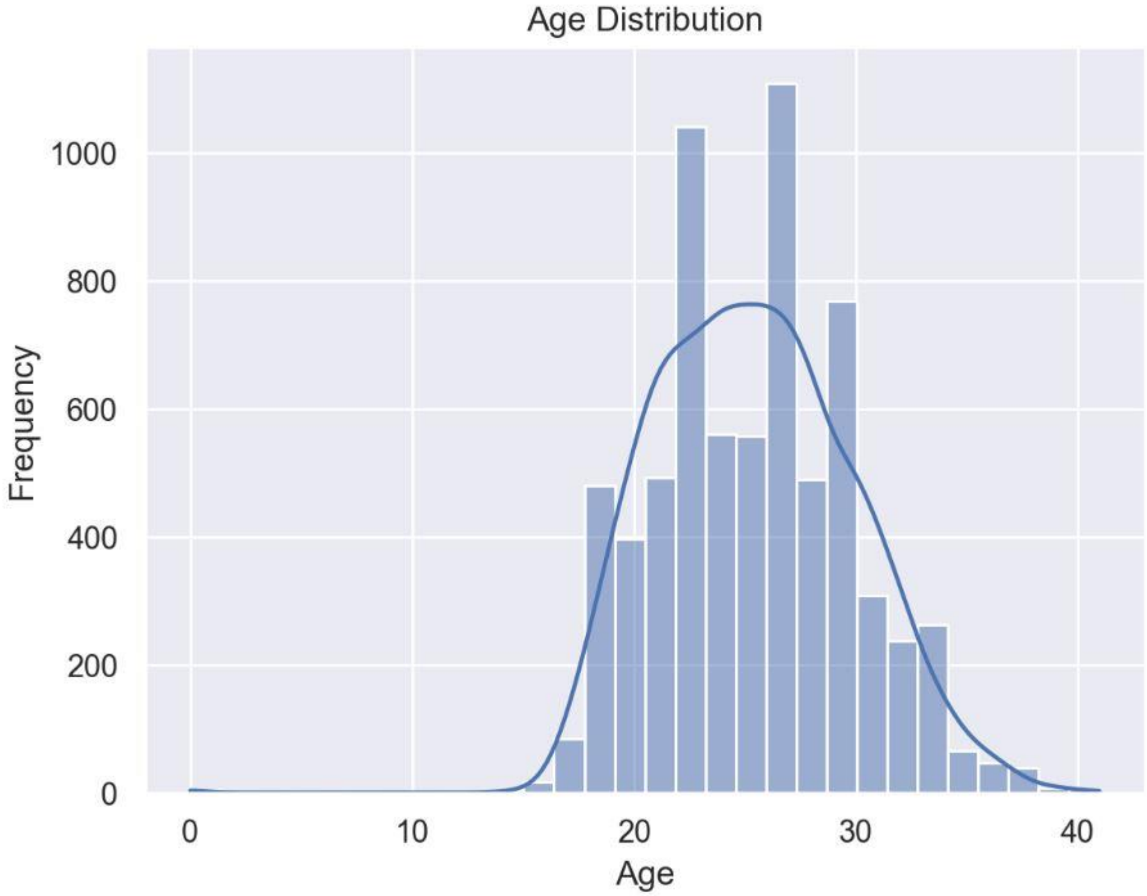

Figure 2. Age Distribution of Players: Histogram showing the frequency of players by age across all seasons.

Figure 3 – Average Value by Position

This bar chart shows the average market value grouped by playing position. Forwards (FW) and attacking midfielders (MF,F) tend to have the highest average values, often above €10M, followed by wingers and central midfielders. Defenders and goalkeepers consistently show lower average valuations. This confirms the economic tendency of the football market to prioritize offensive roles, reinforcing the importance of position as a categorical input variable.

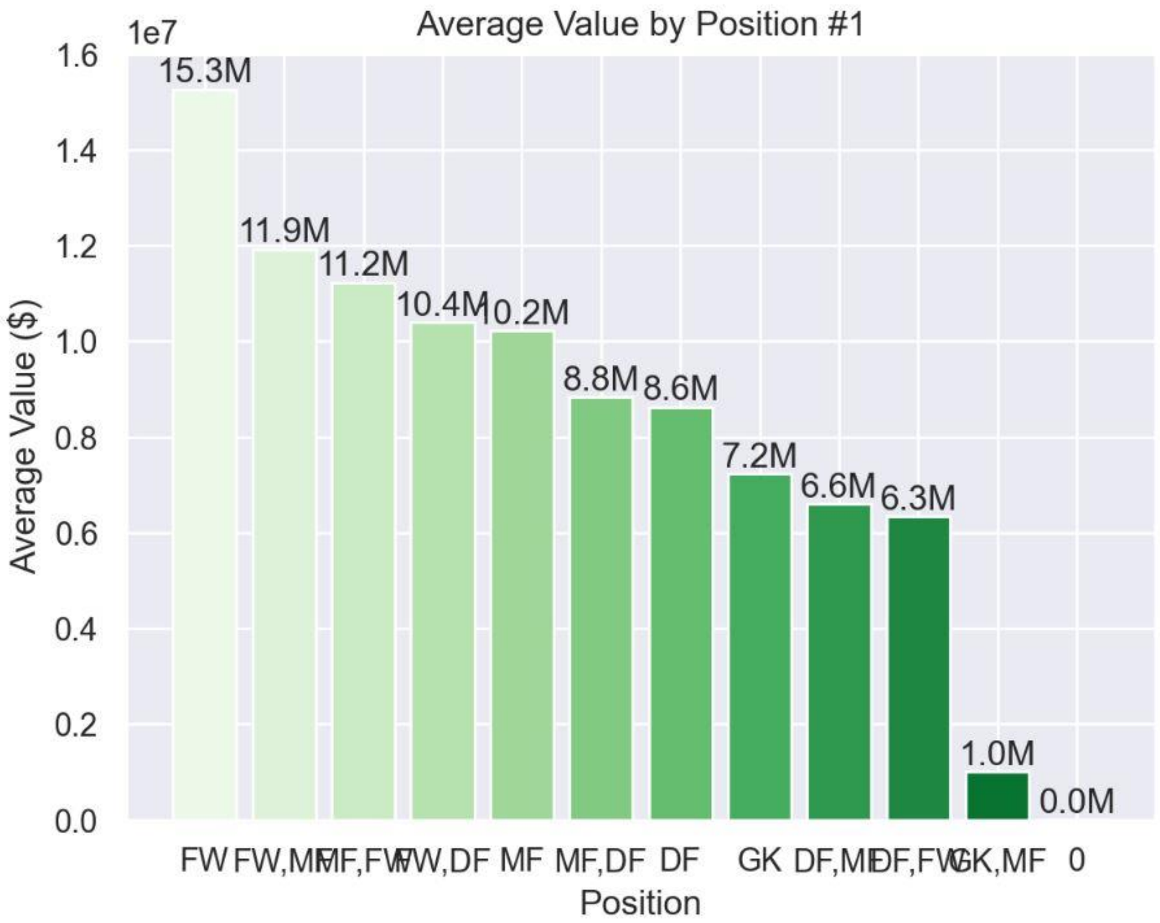

Figure 3. Average Market Value by Position: Bar chart comparing the average estimated value for each playing position.

Figure 4 – Value Distribution by League

A combination of stripplots and boxplots illustrates the distribution of player market values across the five major European leagues. While all leagues include both high- and low-value players, the Premier League exhibits the widest range and the highest median values. This may reflect greater financial power in the English market, which influences how similar player profiles are valued differently by league.

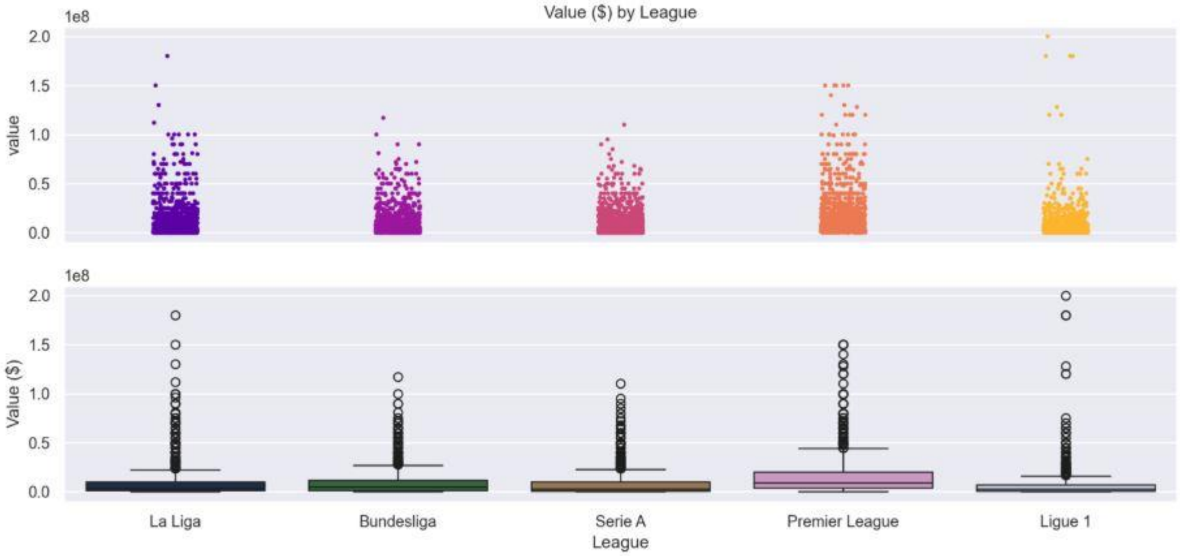

Figure 4. Distribution of Player Values by League: Stripplot and boxplot visualizing value dispersion across Europe's top five leagues.

Figure 5 – Number of Players by Dominant Foot

The majority of players in the dataset are right-footed (~5,000), followed by left-footed players (~1,700), and a small number who use both (~200). This distribution aligns with general population trends. Including footedness as a feature may add predictive power for certain positions (e.g., wingers, full-backs), where foot dominance affects marketability and tactical fit.

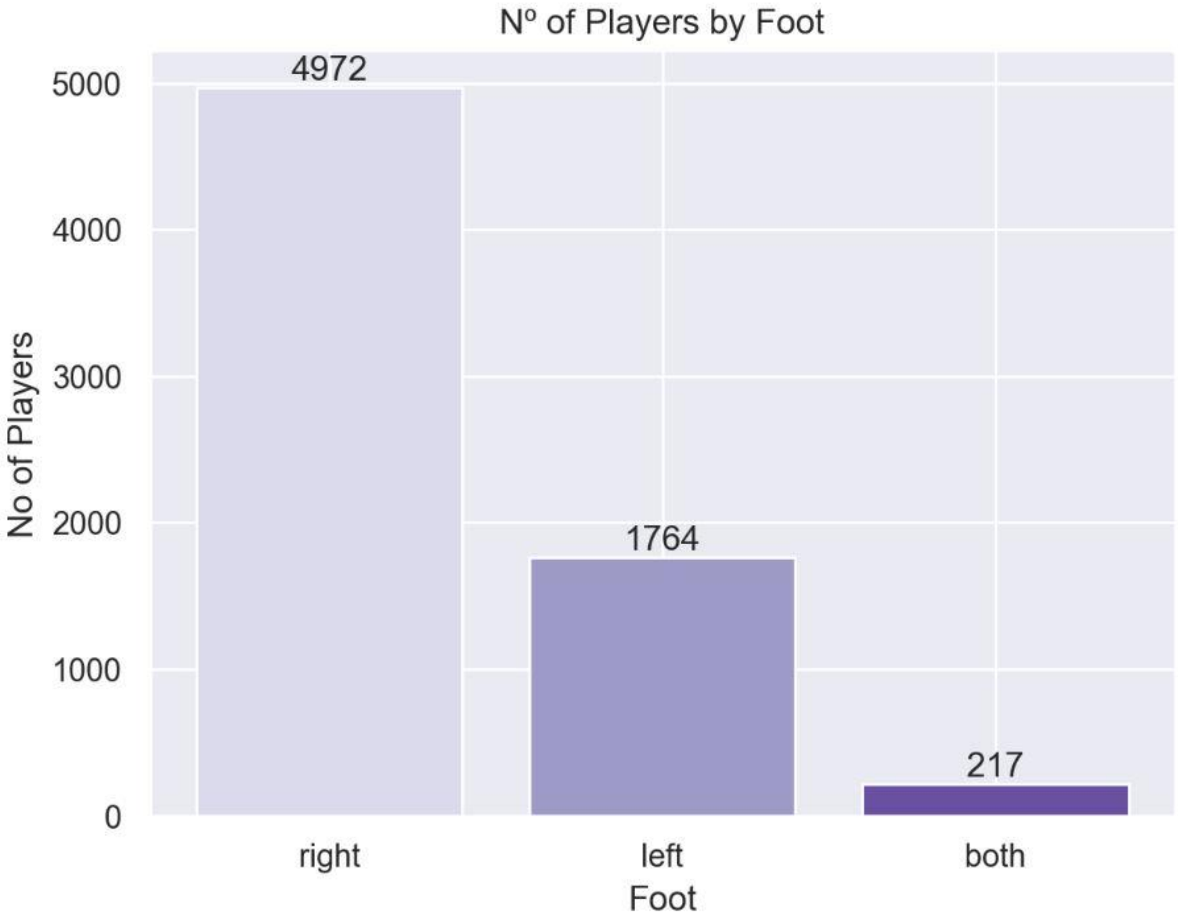

Figure 5. Number of Players by Dominant Foot: Count of players grouped by right-footed, left-footed, and both-footed profiles.

Figure 6 – Market Value Distribution by Foot

This violin plot compares the value distribution for each foot category. While the median values are similar across groups, the "both" category shows a slightly wider spread, suggesting that ambidextrous players may be more variable in quality or market perception. This insight justifies keeping the "foot" variable in the feature set.

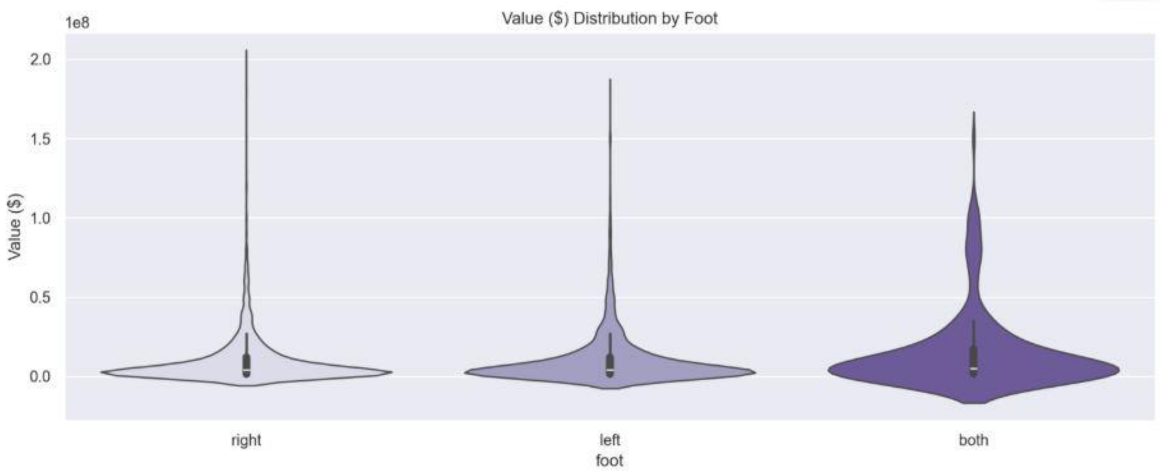

Figure 6. Market Value Distribution by Foot: Violin plots illustrating value distribution according to footedness.

Figure 7 – Number of Athletes by Nationality

This bar chart shows the top twenty nationalities represented in the dataset. Spain, France, and Italy lead the list, followed by Germany, England, Brazil, and Argentina. The distribution reflects the dominance of European football in the dataset and suggests that nationality may indirectly encode other important traits like training background, league exposure, or tactical style.

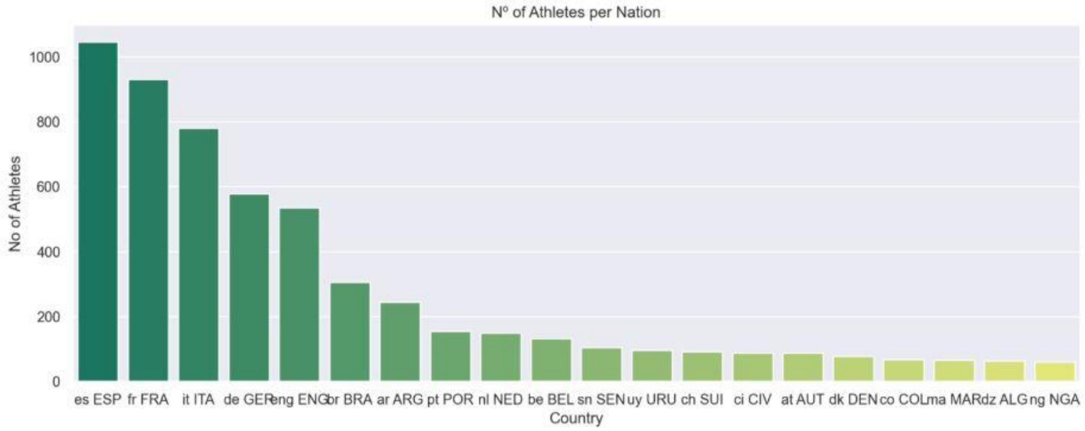

Figure 7. Number of Athletes by Nationality: Bar chart of the top twenty most represented countries in the dataset.

3.3 Project 1: Model Design and Implementation

3.3.1 General Architecture of Project 1

Project 1 was implemented as a modular and maintainable machine learning system designed to estimate the market value of football players using historical data and classical regression models. The system is organized into multiple Python scripts, each responsible for a specific stage of the pipeline. This structure improves code readability, reusability, and ease of debugging.

The main components of the system are as follows:

- **train.py**: Contains the train_model_project1() function, which loads the data, preprocesses it, trains the three candidate models (Random Forest, XGBoost, SVR), evaluates them using RMSE, and saves the best-performing model as a .joblib file.

- **predict.py**: Implements the predict_player_value() function, which receives a player's feature vector (as a Python dictionary), aligns it with the trained model's expected input structure, and returns the predicted market value.
- **utils.py**: Provides utility functions for loading and preparing the dataset:
  - load_and_prepare_data(): Loads the three seasonal datasets (2017/18, 2018/19, 2019/20) from CSV files, adds a year column to each, concatenates them, and drops any entries with missing target (value) data.
  - preprocess_data(): Splits the DataFrame into features (X) and target (y), applies one-hot encoding to categorical variables, drops unused columns (e.g., Attendance), fills missing values with zero, and returns the cleaned X and y.
- **eda.py**: Contains all visual functions used in the Streamlit interface for exploratory data analysis (e.g., age distribution, value by position, value by league, pairplots, etc.), using seaborn and matplotlib.
- **app_project1.py**: The main Streamlit web application. This script creates the GUI with four tabs: "About", "EDA", "Train Model", and "Predict Player Value". It orchestrates the logic from all other modules and allows the user to interact with the system from a browser.
- **main_project1.py**: A command-line version of the application that allows training and prediction through a text-based menu. It is mostly used for testing purposes or as a lightweight interface without GUI dependencies.

- **config.py**: Contains static configuration elements such as file paths for models and data files, improving code organization and portability.

The project also includes a models/ directory to store trained model files (.joblib), a data/ directory for the CSV datasets and real value Excel file, and a requirements.txt (if needed) to manage dependencies.

The system was developed in Python and follows a modular design pattern, where each component can be tested independently. This separation of concerns allows clear data flow from loading and preprocessing to training, prediction, and user interaction.

.

3.4.1 Step 1: Dataset Overview and Preparation

The preprocessing pipeline in Project 1 is designed to transform the raw data into a structured and model-ready format. This step is critical for ensuring that the machine learning algorithms can process the data efficiently and learn meaningful patterns from it. The pipeline is implemented in the utils.py module through two core functions: load_and_prepare_data() and preprocess_data().

Loading and Merging Datasets

The function load_and_prepare_data() is responsible for importing the three seasonal datasets—transfermarkt_fbref_201718.csv, transfermarkt_fbref_201819.csv, and transfermarkt_fbref_201920.csv—from the data/ directory. These datasets contain a mix of structured features such as:

- Personal information: player name, age, nationality, dominant foot.
- Contextual variables: league, club, playing position.
- Performance statistics: matches played, goals, assists, xG, xA, minutes, and others.
- Target variable: value, representing the player's estimated market value in euros.

To track the temporal origin of each entry, a year column is added manually to each dataset before concatenation. The three DataFrames are then merged into one using pd.concat() with ignore_index=True to reset row indices. Any player record without a value in the value column is discarded using dropna(subset=['value']), as these entries cannot be used for supervised learning.

Feature Selection and Cleaning

The merged dataset is passed to the preprocess_data() function. The following preprocessing operations are performed:

1. Target separation:

   The column value is extracted and stored as y. The remaining columns are stored as the feature matrix X.

2. Column dropping:

   Certain columns that are not useful for prediction or may introduce noise are removed. These include:

- Attendance: irrelevant for player-level valuation
- Season: redundant since year was already added during loading

3. Categorical encoding:

   The DataFrame X contains several categorical variables such as position, league, club, and foot. These are transformed into numerical form using one-hot encoding via pd.get_dummies(drop_first=True). This ensures that all input features are numeric and suitable for scikit-learn estimators.

4. Handling missing values:

   After encoding, some features still contain missing values (e.g., for players who did not register data in certain stats). In Project 1, the strategy used is simple zero imputation:

   X = X.fillna(0).

   This approach ensures compatibility with scikit-learn models and avoids the need for row elimination. While simplistic, it maintains data dimensionality and avoids bias from feature removal.

5. Target alignment:

   After preprocessing, the index alignment between X and y is preserved explicitly using:

   y = y[X.index].

At the end of this process, the function returns two objects:

- X: a fully numeric, high-dimensional feature matrix ready for model input
- y: the corresponding market values in euros

This preprocessing pipeline guarantees consistency and reproducibility. It is called once before model training in train_model_project1() and again during prediction, ensuring that new input data is processed using the exact same logic as the training data.

3.3.3 Model Training and Selection

The model training and selection process in Project 1 is implemented in the function train_model_project1(), located in the train.py module. This function automates the full pipeline from data loading and preprocessing to model training, evaluation, and final selection of the best-performing algorithm.

1. **Dataset Preparation**

The function begins by calling:

- load_and_prepare_data() to import the raw dataset.
- preprocess_data() to transform it into the model-ready format.

The processed feature matrix X and target variable y are then split into training and testing subsets using the train_test_split() function from sklearn.model_selection. The test size is fixed at 20% and the random state is set to 42 for reproducibility.

2. Model Selection

Project 1 compares three distinct regression models:

- Random Forest Regressor (sklearn.ensemble.RandomForestRegressor)

  A bagging ensemble method that constructs multiple decision trees on random subsets of the data and averages their predictions to reduce variance and prevent overfitting.
- XGBoost Regressor (xgboost.XGBRegressor)

  A gradient boosting algorithm optimized for performance and accuracy. It sequentially builds weak learners to correct the errors of previous ones, making it ideal for complex relationships.
- Support Vector Regression (SVR) (sklearn.svm.SVR)

  A kernel-based regression technique that fits a function within a margin of tolerance (ε) and seeks to minimize prediction error. SVR performs well with high-dimensional, sparse data.

Each model is trained using .fit() on the training set. Predictions are made on the test set using .predict(). The evaluation metric is the Root Mean Squared Error (RMSE)

This value is printed for each model to the console to enable transparent performance comparison. RMSE is used because it penalizes large errors more heavily than MAE and is expressed in the same units as the target variable (euros), making it easy to interpret in a business context.

3. **Best Model Selection**

During the iteration, the model with the lowest RMSE is selected and stored as the final model. Once all models are evaluated, the best-performing one is serialized using joblib.dump() and saved in the models/ directory under the filename best_model_project1.joblib. The function then prints a confirmation message with the name of the best model and its RMSE score.

4. **Output**

Finally, the function returns a tuple containing:

- The name of the best model (e.g., "RandomForest")
- Its RMSE score (float)

This return value is used later in the Streamlit app to inform the user about the result of training.

3.3.4 Saving and Loading the Final Model

Once the training and evaluation of all candidate models are completed in Project 1, the best-performing model is preserved for future use. This is a crucial step in any applied machine learning project, as it enables prediction on new data without having to retrain the model every time the application is launched.

The selected model—either Random Forest, XGBoost, or SVR, depending on performance—is serialized using Python's joblib library and saved to disk in a dedicated models/ directory. This serialized model file encapsulates both the learned parameters and the internal structure of the trained algorithm, making it fully portable and restorable across sessions and systems.

- By saving the model externally, the system achieves two goals:
- Efficiency: Avoids retraining the model on each execution, reducing computation time.

Reproducibility: Ensures that the same trained model is used consistently when making predictions, as long as the underlying data and feature structure remain unchanged.

To use the model during prediction, it is loaded back into memory using the same serialization tool. The loading process restores the exact same object with its trained state, allowing it to make predictions immediately upon receiving properly formatted input.

This approach also allows users to retrain the model on demand (e.g., when new data is available or performance degrades) via the Streamlit interface. After retraining, the best new model simply overwrites the existing file. This dynamic yet controlled mechanism enables flexibility without compromising stability or reliability.

The saved model file is later accessed by the prediction module and by the web application, ensuring a seamless workflow between training and deployment phases. This modularity ensures that each component (training, prediction, interface) interacts with the model in a consistent and independent way.

3.3.5 Prediction System: Flow and Logic

The prediction system in Project 1 is designed to estimate the current market value of a specific football player based on their most recent available data from the 2017–2020 seasons. This system is fully integrated into the Streamlit application under the "Predict Player Value" tab and interacts directly with the trained model stored on disk.

The process begins when a user inputs the name of a player into the application. The system performs a case-insensitive search across the combined dataset to identify matching entries. If multiple players with similar names are found, the application prompts the user to select the exact individual from a dropdown list. This step ensures accuracy and avoids ambiguity in player identification.

Once the player is selected, the system retrieves all historical records associated with that player. These may include data from one or more seasons. The most recent entry is then chosen as the basis for prediction. This entry includes all relevant variables previously used during training—such as age, position, league, goals, minutes played, and other performance metrics.

Before the prediction can be made, the player's data must be processed in the same way as the training data. The system applies the exact same preprocessing pipeline: unnecessary columns are removed, categorical variables are one-hot encoded, and the final input vector is aligned to match the structure expected by the trained model.

To ensure compatibility, any features present in the training model but missing in the input are added with default values (typically zero). This guarantees that the input dimensions and feature order are identical to those used during training, which is essential for the model to function properly.

Once the input vector is finalized, it is passed to the loaded model, which returns a predicted market value in euros. This value is then formatted and displayed to the user directly within the application.

In addition to the prediction, the system checks whether the player has an associated real market value for the 2020–21 season. This information is extracted from a separate Excel file containing actual valuations. If available, the real value is also displayed alongside the predicted value, allowing the user to visually compare model output with real-world data.

This interactive and dynamic prediction workflow transforms the application from a static academic tool into a usable prototype with real-world applicability. It allows users to explore historical player data, receive instant predictions, and assess the system’s performance in a transparent and intuitive manner.

3.3.6 User Interface Design (Streamlit)

The graphical user interface (GUI) for Project 1 was developed using Streamlit, an open-source Python library that allows for the rapid development of interactive web applications directly from Python scripts. The interface is designed to be simple, intuitive, and functional, enabling both technical and non-technical users to interact with the system without needing to modify the underlying code.

Upon launching the application, the user is presented with a wide-layout dashboard containing four main navigation tabs accessible from a sidebar menu:

1. About

This introductory section provides an overview of the application, its purpose, and the technologies involved. It briefly describes the machine learning models used, the source of the data, and the goal of the project. This section is particularly useful for first-time users and for contextualizing the system within the scope of the thesis.

2. EDA (Exploratory Data Analysis)

The EDA tab offers a series of interactive visualizations that allow the user to explore the dataset graphically. The visualizations are created using seaborn and matplotlib and rendered directly in the browser via Streamlit. The following plots are included:

- Age Distribution: A histogram showing the frequency of players across different age groups.
- Average Value by Position: A bar chart highlighting how player value varies by playing position.
- Value by League: A combined stripplot and boxplot showing the distribution of player values per league.
- Value by Foot: A bar chart and violin plot analyzing player value distribution by dominant foot.
- Top Nationalities: A bar chart of the countries with the highest number of players in the dataset.
- Pairplot of Selected Variables: A scatterplot matrix showing correlations among key performance metrics.

These visualizations help the user understand the distribution and relationships between variables in the dataset before engaging with the predictive tools.

3. Train Model

This tab allows the user to trigger the training process directly from the interface. With a single button click, the system loads the data, preprocesses it, trains the three regression models (Random Forest, XGBoost, SVR), evaluates them, and selects the best-performing model based on RMSE.

Once training is complete, the name of the best model and its RMSE score are displayed to the user. This feedback loop makes the process transparent and allows for retraining in case the dataset is updated or modified.

4. Predict Player Value

In this final tab, the user can enter the name of any player from the dataset. The system searches for all matching entries and offers a selection interface if needed. Once the player is confirmed, the system retrieves the most recent data entry, processes the features, and uses the trained model to predict the player's market value.

If the player also appears in the external 2020–21 value dataset, the system retrieves and displays their actual market value for comparison. This side-by-side output allows users to assess how closely the model's prediction aligns with real-world data.

Throughout the interface, results are displayed in clear numerical format and styled for readability. The use of Streamlit ensures that the application is responsive, browser-accessible, and easy to maintain or expand in the future.

3.3.7 Comparison with Real Market Values

A key feature of Project 1 is its ability to compare predicted player market values with real valuations from an external, verifiable source. This comparison enhances the transparency and credibility of the model by situating its outputs within the context of real-world football economics.

To enable this functionality, the system incorporates a separate dataset: an Excel file containing the actual market values of players for the 2020–21 season, collected from Transfermarkt. This file is loaded at runtime in both the backend prediction script and the Streamlit application. The data includes two key columns: the player's name and their real market value (in euros), as published for that season.

Before comparison, a preprocessing step is applied to standardize player names. This includes:

- Trimming white spaces
- Converting all names to lowercase
- Removing special characters if needed

This normalization process ensures that the player names from the main dataset and the 2020–21 file can be matched reliably, despite formatting differences.

When a user requests a prediction for a given player, the system first computes the estimated market value using the trained model. Immediately afterward, it checks whether that player exists in the real value dataset. If a match is found, the application retrieves and displays the actual market value from the 2020–21 season alongside the predicted figure.

The user sees both values in clear numerical form within the interface:

- Predicted value: output of the trained model.
- Real value: from the Transfermarkt-based Excel file

If the real market value is not available for a specific player, the system informs the user accordingly. This avoids confusion and highlights the boundaries of the comparison feature.

This design decision provides an important layer of model validation. It allows users to evaluate how close the prediction is to reality, spot potential over- or underestimations, and reflect on the possible reasons for discrepancies (e.g., injuries, transfers, media influence).

By integrating real-world reference data into the predictive workflow, Project 1 not only delivers estimations but also invites critical interpretation of the results—turning the system into a practical tool for football valuation analysis.

3.4 Project 2: Model Design and Implementation

3.4.1 General Architecture of Project 2

Project 2 was developed as a second, fully independent system to estimate football players' market value using an alternative set of regression algorithms and a slightly refined preprocessing strategy. The goal of this second system is to assess whether different modeling choices—such as imputation methods and algorithm families—can lead to improved or more stable prediction performance when applied to the same dataset.

The overall structure of Project 2 mirrors that of Project 1 for consistency and comparability. However, Project 2 is stored in a separate module directory and uses distinct scripts with nearly identical roles but adapted to the new models and processing logic.

The main components of Project 2 are:

- train.py: Contains the train_model_project2() function. This script handles model training, evaluation, and selection among three algorithms: CatBoost, K-Nearest Neighbors (KNN), and Gradient Boosting Regressor.
- predict.py: Defines the predict_player_value() function, which loads the trained model and uses it to make predictions on player input vectors.
- utils.py: Provides the data loading and preprocessing functions:
    - load_and_prepare_data() performs the same task as in Project 1: loading the three seasonal CSVs, tagging them with a year column, and combining them.
    - preprocess_data() performs feature transformation, encoding, and importantly, introduces a mean imputation strategy to handle missing values, replacing the simpler fill-zero approach used in Project 1.
- eda.py: This file is reused from Project 1 and contains all exploratory visualizations. The same visual tools are used for both systems to maintain consistency and allow direct visual comparison.

- app_project2.py: The Streamlit application for Project 2. It has the same layout and logic as in Project 1 but points to a different model file and retrains the CatBoost/KNN/Gradient Boosting models. The user experience remains identical in structure but is backed by different technical components.
- main_project2.py: Offers a command-line interface for training and prediction, similar to main_project1.py, providing a lightweight alternative to the GUI.
- config.py: Stores static variables such as file paths and model names specific to Project 2.

All trained models in this project are saved under the filename best_model_project2.joblib in the models/ directory, avoiding conflict with models from Project 1.

By keeping the two systems modular and structurally parallel, the thesis ensures a controlled environment for testing and comparing different learning strategies. The separation also allows each project to be independently deployed or expanded in future research or production contexts.

3.4.2 Preprocessing Pipeline

While Project 2 uses the same base dataset as Project 1, it introduces a refined preprocessing strategy, particularly regarding how missing values are handled. The preprocessing pipeline is implemented in the utils.py file and consists of two key functions: load_and_prepare_data() and preprocess_data().

Data Loading

As in Project 1, the function load_and_prepare_data() imports data from the three separate CSV files corresponding to the 2017/18, 2018/19, and 2019/20 seasons. Each file is assigned a year column to preserve temporal context. All datasets are concatenated into a single DataFrame using vertical stacking, and player entries with missing target values (value) are excluded to maintain label integrity.

The result is a unified dataset covering thousands of player-season records across multiple leagues and positions.

Preprocessing Enhancements in Project 2

The major differences between Project 1 and Project 2 begin in the preprocess_data() function:

1. Target Separation and Column Cleaning

   As in Project 1, the value column is extracted as the target variable y, while the remaining columns form the feature matrix X. Redundant or irrelevant columns such as Attendance and Season are dropped to reduce noise.

2. Categorical Encoding

   All categorical variables are transformed into binary indicators using one-hot encoding. The same drop_first=True setting is used to avoid multicollinearity. This transformation increases the dimensionality of the dataset but makes it compatible with the regression algorithms selected for Project 2.

3. Numerical Imputation (Mean Strategy)

   Unlike Project 1—which fills all missing values with zero—Project 2 uses a more statistically grounded method: mean imputation.

This is done using a SimpleImputer from sklearn.impute, which replaces any missing numeric value with the meaning of its column. This approach helps preserve the natural scale of the data and avoids introducing artificial zeroes, which could distort relationships between variables and target.

This change is particularly relevant when using distance-based algorithms such as KNN or boosting methods like CatBoost, where feature distribution and scaling can have a significant impact on model performance.

4. DataFrame Reconstruction

   After imputation, the imputed data is converted back into a pandas DataFrame, ensuring that column names are preserved and aligned with the original structure. This step is essential for model interpretation and downstream compatibility during prediction.

5. Index Alignment

   As in Project 1, the target variable y is aligned with the final version of the feature matrix X to ensure consistent indexing and avoid training errors.

By introducing mean imputation and maintaining the rest of the pipeline aligned with Project 1, Project 2 offers a more refined and realistic treatment of missing data. This change may improve model generalization and is particularly beneficial in high-dimensional datasets where zero imputation may introduce unintended bias.

3.4.3 Model Training and Selection

The training and selection of models in Project 2 follow the same general framework established in Project 1, but introduce a different set of machine learning algorithms. These were chosen to explore the behavior of models from diverse families: gradient boosting, instance-based learning, and categorical-aware boosting. The objective is to determine whether these alternative approaches offer better predictive accuracy or robustness compared to the models used in Project 1.

The training procedure is managed by the train_model_project2() function located in the train.py script. It begins by loading the unified dataset and applying the preprocessing pipeline described in the previous section. The resulting data is split into training and testing sets using a fixed 80/20 split, ensuring comparability across projects.

Three regression models are then trained and evaluated:

1. **CatBoost Regressor**

CatBoost is a gradient boosting framework specifically designed to handle categorical features efficiently and prevent overfitting through techniques like ordered boosting and minimal data leakage. In this project, even though the features have already been one-hot encoded, CatBoost still offers performance advantages due to its robustness, native handling of missing values (if any remained), and automatic regularization. The model is configured with minimal verbosity to focus on output performance.

2. **K-Nearest Neighbors Regressor (KNN)**

KNN is a non-parametric algorithm that makes predictions based on the values of the 'k' closest training samples in the feature space. It is sensitive to the scale and distribution of the data, which makes the mean imputation strategy especially relevant. KNN provides a simple and interpretable alternative that is often overlooked in football analytics, offering a useful benchmark from a different modeling paradigm.

3. Gradient Boosting Regressor

This model, part of the sklearn ensemble module, builds an additive model in a forward stage-wise fashion. It combines weak learners (decision trees) in a sequential manner, where each new learner attempts to correct the residual errors of the previous ones. Gradient Boosting is known for its strong performance on structured tabular data and was included to offer a comparable counterpart to XGBoost from Project 1.

Each model is trained on the same training set and evaluated on the same testing set using the Root Mean Squared Error (RMSE). This metric penalizes larger errors more heavily and provides an intuitive interpretation in terms of monetary deviation from the actual market value.

Throughout the training loop, the model with the lowest RMSE is selected as the final model for the system. Its name and performance score are printed to the console for transparency and saved for downstream use.

This comparison-based training strategy ensures that Project 2 uses the most suitable model for the given data and configuration, and that the results can be directly compared with those from Project 1 under consistent experimental conditions.

3.4.4. Saving and Loading the Final Model

Once the best-performing model has been identified in Project 2 through RMSE-based comparison, it is preserved for future use by saving it to disk as a serialized binary file. This process is essential for ensuring that the selected model can be reused efficiently without the need for retraining each time a prediction is requested.

The final model—whether it is CatBoost, K-Nearest Neighbors, or Gradient Boosting—is saved using Python's joblib library, which is optimized for storing scikit-learn and similar model objects. The model file is stored in the models/ directory under the name best_model_project2.joblib. This filename is unique to Project 2 to prevent any conflict with the output from Project 1.

The use of model serialization achieves several objectives:

- Efficiency: The system avoids redundant computation by eliminating the need to retrain the model on every execution.
- Consistency: Once trained and saved, the model always provides the same output for identical input, ensuring reproducibility.
- Integration: The saved model can be easily loaded by both the Streamlit application and the command-line version of the system, maintaining a clean separation between training and inference logic.

When the prediction module or user interface is launched, the saved model is loaded from disk into memory. This restored model includes all trained weights, decision trees, or instance memory (in the case of KNN), depending on the algorithm selected. It is immediately ready to accept preprocessed player data and generate output.

The modular design of this system allows for retraining and overwriting of the model whenever desired. For example, a user may initiate a new training session from the web interface, which will replace the existing model file with a newly trained one. This flexible but controlled setup makes the system both dynamic and stable in its deployment.

By following this approach, Project 2 ensures seamless integration between model development, deployment, and usage, and maintains parity with the design principles established in Project 1.

3.4.5 Prediction System: Flow and Logic

The prediction system in Project 2 follows the same fundamental logic as in Project 1 but is adapted to work with the alternative models and preprocessing strategy specific to this version of the application. It is fully integrated into the Streamlit interface under the "Predict Player Value" tab and shares the same user experience design to ensure continuity and usability.

Player Search and Selection

The process begins when the user enters the name of a football player in a text field. The system performs a case-insensitive search within the full dataset (2017–2020) to locate matching entries. If the name is unique, the system proceeds automatically; if multiple entries are found, a dropdown menu allows the user to disambiguate and select the correct player. This matching mechanism ensures user control and reduces the risk of errors caused by duplicate or similar names.

Retrieving the Player's Data

Once a player is selected, the system collects all available seasonal entries for that player and identifies the most recent one (typically from the 2019–20 season). This record includes all variables used during model training: physical characteristics, position, league, statistical performance, and contextual information. This entry is then prepared for model input.

Preprocessing for Prediction

Before feeding the player data into the model, it must be transformed using the same preprocessing pipeline that was applied during training. This includes:

- Dropping irrelevant or unused columns
- Applying one-hot encoding to categorical variables
- Ensuring all feature columns match the expected structure of the trained model.
- Imputing any missing values using the mean strategy, as defined in the training phase.

The resulting feature vector is then aligned with the model's original input structure. Any missing columns are added with default values (typically zero), and the column order is matched precisely to avoid input errors.

Generating the Prediction

The preprocessed player data is passed to the trained model, which has been previously loaded into memory from the serialized file. The model immediately returns a predicted market value for the selected player, expressed in euros. This prediction is formatted and displayed clearly within the interface.

Comparison with Real Market Value

If the selected player also appears in the separate 2020–21 market value dataset (loaded from an Excel file), the system retrieves and displays the real value for comparison. This side-by-side output provides users with a clear understanding of the model's accuracy and practical relevance. If no real value is available for that player, a message is shown indicating the lack of reference data.

This seamless workflow enables dynamic, data-driven valuation of players while preserving interpretability and control for the user. It also demonstrates how predictive models can be integrated into user-facing applications to support decision-making processes in real-world football economics.

3.4.6 User Interface Design (Streamlit)

The user interface for Project 2 is built using Streamlit and maintains the same structural layout and user experience as Project 1. This decision ensures consistency across both systems, making it easier to compare model behavior under equivalent conditions. While the appearance and functionality of the application remain virtually identical, the backend logic is fully adapted to accommodate the specific models and data handling techniques of Project 2.

Upon launching the Streamlit application, users are presented with a sidebar navigation menu containing four main sections: "About", "EDA", "Train Model", and "Predict Player Value". Each tab performs the same role as in Project 1 but is internally connected to Project 2’s specific training script, model file, and prediction logic.

About

The "About" tab explains that the application corresponds to Project 2 and outlines the differences in modeling techniques used. It lists the three algorithms implemented—CatBoost, K-Nearest Neighbors, and Gradient Boosting—and briefly describes the evaluation strategy based on RMSE. This section helps users understand that the system represents an alternative approach to the one used in Project 1.

EDA (Exploratory Data Analysis)

The EDA tab offers the same interactive plots available in Project 1, as both systems use the same dataset. Visualizations include age distribution, value by position, league, nationality, footedness, and pairwise relationships between performance variables. This consistent visual analysis supports the comparison of player trends, model performance, and contextual patterns across both systems.

Train Model

This tab allows users to initiate model training from within the interface. Upon clicking the training button, the application triggers the train_model_project2() function, which trains and evaluates the three candidate models. Once completed, the application informs the user of the best model selected and its RMSE score. This functionality enables the retraining of the model whenever the dataset is updated, or the user wants to test alternative results.

Predict Player Value

This tab mirrors the prediction interface from Project 1. Users input a player’s name, confirm their selection from the search results, and receive a predicted market value for that player. The system processes the player’s data using Project 2’s specific preprocessing pipeline, then passes the input vector to the best-performing trained model. If the player's real value for the 2020–21 season is available, it is displayed alongside the prediction for comparison.

Although the user interface remains familiar, the application is internally connected to different model files, logic, and evaluation mechanisms. This modularity makes it possible to switch between predictive strategies while keeping the user experience stable and intuitive.

3.4.7 Comparison with Real Market Values

As with Project 1, Project 2 includes a validation mechanism that compares the predicted market value of a football player with their real-world valuation during the 2020–21 season. This feature is designed to provide immediate feedback on the model's performance, improve user trust, and support critical interpretation of the predictions.

The real market values are loaded from an external Excel file containing verified data sourced from Transfermarkt. This dataset includes the names of professional players along with their market value expressed in euros. The file is read during application startup and stored in memory for fast access during the prediction phase.

Before attempting to match a player, the system normalizes all player names in both the primary dataset and the real value dataset. This normalization includes converting names to lowercase, trimming whitespace, and removing inconsistent characters. The purpose of this step is to ensure reliable matches despite potential inconsistencies in formatting between sources.

Once a prediction is generated by the trained model, the system automatically checks whether the same player exists in the real value dataset. If a match is found, the actual value from the 2020–21 season is retrieved and displayed next to the predicted figure. Both values are formatted clearly and presented together on the interface.

If the player is not found in the external dataset, the system displays a notice informing the user that no real value is available for comparison. This avoids confusion and sets expectations regarding data completeness.

By providing both the predicted and real market value side by side, the system encourages critical analysis of model performance. It allows users to assess whether the model overestimated, underestimated, or closely approximated the real value. This also gives insight into factors the model may not fully capture—such as player injuries, media hype, contractual details, or off-pitch considerations.

This comparison feature adds an essential layer of interpretability and bridges the gap between academic modeling and real-world football economics. It reinforces the thesis's commitment to transparency and practical validation of machine learning in player valuation.

3.5 Evaluation Strategy

The evaluation strategy in this thesis is designed to ensure that both Project 1 and Project 2 are assessed under equivalent conditions using consistent metrics, datasets, and validation logic. This methodological alignment is essential to enable a fair comparison between different machine learning models and pre-processing pipelines applied to the same predictive task: estimating the market value of professional football players.

Choice of Evaluation Metric

The primary evaluation metric used in both systems is the Root Mean Squared Error (RMSE). RMSE was selected for several reasons:

- Interpretability: RMSE is expressed in the same units as the target variable—in this case, euros—which makes the error easy to interpret and directly comparable to real-world values.
- Penalty for large errors: RMSE penalizes larger deviations more severely than other metrics such as Mean Absolute Error (MAE), making it a better choice for applications where significant misestimations are more problematic than minor fluctuations.
- Standard usage: RMSE is widely adopted in regression problems and provides a familiar benchmark for comparison across studies.

For each trained model, the RMSE is calculated on the test dataset, which is held out during training and not used for model fitting. This ensures that the evaluation reflects generalization performance rather than overfitting the training data.

Dataset Consistency

To guarantee comparability, both Project 1 and Project 2 are trained and evaluated on the exact same dataset: a combined multi-season collection of player records from 2017–2020, derived from Transfermarkt and FBRef. The same feature set is used in both projects after one-hot encoding and cleaning, with the only difference being the missing value imputation method (zero-filling in Project 1 vs. mean-imputation in Project 2).

Furthermore, both projects use the same train/test split logic, using an 80/20 split with a fixed random seed. This ensures that both models are exposed to the same training examples and are evaluated on the same testing examples, eliminating variability due to data partitioning.

Model Comparison Strategy

In both projects, three candidate models are trained on the same data and compared based on their RMSE scores. Only the best-performing model in each project is retained for deployment and prediction. This approach guarantees that each project uses its most accurate model and allows the final comparison to reflect the best-case scenario for each methodology.

Real-World Validation

In addition to RMSE-based evaluation, the system includes a real-world validation layer. During the prediction phase, the estimated market value of each player can be compared with the actual market value recorded for the 2020–21 season. This comparison serves as an informal, post-hoc validation of the model's accuracy on unseen players and seasons and provides users with an intuitive sense of prediction quality.

Although this validation is not used in model selection or training, it offers valuable practical insight and supports the applied nature of the thesis.

3.6 Technical Stack and Tools

The development of both Project 1 and Project 2 was carried out entirely in Python, a widely used programming language in data science and machine learning. The choice of Python was motivated by its rich ecosystem of libraries, active community, and compatibility with both research and production environments.

Programming Language

- **Python** 3.9 [14]: The core language used for data processing, model development, and interface creation. Its simplicity and expressiveness allowed for rapid prototyping and modular design.

Data Manipulation and Analysis

- **Pandas** [15]: Used extensively for loading, cleaning, transforming, and analyzing tabular data. Provided essential functionality for merging datasets, handling missing values, and preparing features.
- **NumPy** [16]: Used in combination with pandas for numerical operations and efficient data structures.

Machine Learning Libraries

- **scikit-learn** [17]: Provided implementations for Random Forest, SVR, K-Nearest Neighbors, Gradient Boosting, and utilities like train/test splitting, model evaluation (RMSE), and feature processing (e.g., one-hot encoding, imputation).
- **XGBoost** [18]: Used specifically in Project 1 as a high-performance boosting algorithm known for its accuracy and speed.
- **CatBoost** [19]: Used in Project 2 as a gradient boosting method optimized for categorical data, offering better handling of feature interactions and default regularization.

- **Joblib** [20]: Used for saving and loading trained models to and from disk in a compressed, portable format.

Web Application Framework

- **Streamlit** [21]: Chosen for building the graphical user interface of both projects. It allowed for fast deployment of interactive dashboards without requiring web development experience. Each app includes sections for exploratory data analysis, model training, and player value prediction.

Visualization Libraries

- **matplotlib** [22] and **seaborn** [23]: Used in the EDA (Exploratory Data Analysis) modules to create informative charts such as histograms, bar plots, violin plots, and pairplots. These visualizations help users explore the structure of the dataset and interpret relationships between variables.

Data Sources

- **Transfermarkt** [24] and **FBRef** [25]: Provided the base datasets containing player information and performance statistics for the 2017–2020 seasons.
- Excel dataset (2020–21 values): Used as a reference for real-world player valuations to compare with model predictions.

Environment and Tools

- **VS Code** [26]: The primary code editor used for development and debugging.

- Local execution environment: The system was designed to be executed locally without requiring cloud infrastructure, ensuring reproducibility and ease of testing.

Directory structure:

- data/: Contains raw CSV files and the Excel file with real market values.
- models/: Stores serialized model files (.joblib).
- project1/ and project2/: Contain the source code for each project, including scripts for training, prediction, preprocessing, and GUI.

This technical stack was chosen for its balance between flexibility, power, and accessibility, allowing the development of two fully functional machine learning applications suitable for both academic experimentation and real-world demonstration.

Chapter 4 RESULTS

4.1 Quantitative Results (Project 1)

The first experimental system developed in this thesis—Project 1—was evaluated through a quantitative comparison of three widely used supervised regression algorithms: Random Forest, XGBoost, and Support Vector Regression (SVR). All models were trained using the same dataset composed of players from the top five European leagues over three seasons (2017–2020) and evaluated using the Root Mean Squared Error (RMSE) as the primary performance metric.

This metric was selected because it penalizes larger errors more heavily than MAE (Mean Absolute Error), is expressed in the same unit as the target variable (euros) and provides an intuitive interpretation of prediction accuracy in a financial context.

- The experimental setup maintained strict consistency:
- The dataset split (80% training / 20% testing) was fixed using a defined random seed.
- All models were trained using identical input data structures, after one-hot encoding and filling missing values with zero.
- No hyperparameter tuning was performed, ensuring a fair baseline comparison.

The RMSE scores obtained during evaluation were:

Model	**RMSE (€)**
Random Forest Regressor	9,493,750.69
XGBoost Regressor	8,816,509.86

Model	RMSE (€)
Support Vector Regression (SVR)	18,993,197.34

Table 1. RMSE results for the three regression models evaluated in Project 1. XGBoost achieved the lowest error and was selected as the final model.

The results clearly show that XGBoost outperforms the other two models, achieving the lowest RMSE and therefore the highest overall predictive accuracy on unseen data. While Random Forest also performed well, it did not match the gradient-boosted optimization and regularization techniques offered by XGBoost. SVR, on the other hand, delivered significantly higher error, which can be attributed to its sensitivity to input scale, high-dimensional data, and lack of feature interaction modeling.

These results highlight the robustness of ensemble tree-based models—particularly boosting methods—in handling complex, noisy, and structured datasets such as football performance data.

4.2 Model Selection and Prediction Examples (Project 1)

Based on the results presented in the previous section, XGBoost Regressor was selected as the final model for Project 1, having achieved the lowest Root Mean Squared Error (RMSE) of approximately €8.82 million. This choice reflects both its empirical performance and its well-established capacity to model non-linear relationships and feature interactions in structured data.

Once the best model was saved, it was deployed within the Streamlit application to allow interactive prediction of market values for individual players. Below are several real examples that illustrate how the system performs when predicting current market values based on past seasons.

Cristiano Ronaldo

- **Historical market values**:
    - 2017/18: €100,000,000
    - 2018/19: €90,000,000
    - 2019/20: €60,000,000
- **Predicted value (2020–21)**: €66,410,000
- **Real value (2020–21)**: €60,000,000
- **Difference**: +€6.41M (~10.7% overestimation)

The model captured the general downward trend in Cristiano Ronaldo's market value as he progressed in age, but slightly overestimated his final valuation. This may be attributed to the fact that his technical performance indicators remain high, which could bias the model towards a higher predicted value. The gap is moderate and reflects the challenge of modeling intangible factors such as market perception, age-related depreciation, or strategic contract decisions.

Neymar Jr.

- **Historical market values**:

    - 2017/18: €180,000,000
    - 2018/19: €180,000,000
    - 2019/20: €128,000,000
- **Predicted value (2020–21)**: €112,600,000
- **Real value (2020–21)**: €128,000,000
- **Difference**: –€15.4M (~12% underestimation)

In Neymar's case, the model underestimates his market value despite his historically high ratings and consistent performance. This could be due to injuries or limited appearances affecting his short-term statistics, which are heavily weighted in the model. Moreover, players with exceptional commercial value may be systematically undervalued by performance-based models alone.

Antonio Rüdiger

- **Historical market values**:
    - 2017/18: €40,000,000
    - 2018/19: €50,000,000
- **Predicted value (2020–21)**: €47,075,000
- **Real value (2020–21)**: €40,000,000

- **Difference**: +€7.07M (~17.7% overestimation)

Antonio Rüdiger’s predicted value is slightly higher than his real one, although both are in the same general range. As a central defender, his performance metrics may not always reflect his tactical impact, leadership, or club negotiations—all of which influence market valuation but are not explicitly captured in the dataset.

Antonio Sivera

- **Historical market values**:
  - 2017/18: €1,000,000
  - 2018/19: €1,500,000
  - 2019/20: €1,000,000
- **Predicted value (2020–21)**: €942,100
- **Real value (2020–21)**: Not available

For lower-profile players like Antonio Sivera, the model offers predictions that align closely with historical trends. However, the absence of a real reference value prevents formal validation. Still, the system successfully avoids extreme outliers or unreasonable estimations, showing that it generalizes well even on modest-value cases.

These examples demonstrate that while the model does not achieve perfect alignment with real-world valuations, it consistently produces reasonable, data-driven estimations that can serve as a solid foundation for valuation analysis.

4.3 Project 1: Summary of Observations

Project 1 has demonstrated the effectiveness of supervised learning models—especially ensemble-based methods—for estimating the market value of professional football players based on historical performance data. Through the evaluation of three distinct regression algorithms, the following key observations emerged:

1. **XGBoost outperforms traditional models.**

Among the three models tested, XGBoost achieved the lowest RMSE, outperforming both Random Forest and Support Vector Regression. This is consistent with expectations, as XGBoost implements regularized gradient boosting, which allows it to efficiently model complex, non-linear relationships in structured datasets. Its ability to handle sparsity and feature interactions likely contributed to its superior performance.

2. **SVR is not suitable for this context.**

Support Vector Regression yielded the highest RMSE by a wide margin. This underperformance is likely due to:

- Sensitivity to feature scale and high dimensionality.
- Lack of embedded feature importance or interaction modeling

- Limited capacity to generalize in the presence of categorical variables, even after one-hot encoding.

This result supports the broader consensus in machine learning literature that tree-based methods are more robust for tabular data, particularly when interpretability and performance are both required.

3. **Preprocessing had a stabilizing effect.**

The preprocessing strategy in Project 1—one-hot encoding for categorical variables and zero imputation for missing values—proved effective in delivering consistent results across all models. While zero imputation may not be optimal in every context, it avoided introducing statistical bias or leakage from test data and simplified pipeline reproducibility.

4. **The model generalizes well on both high- and low-value players.**

Prediction examples suggest that the model performs reasonably well across a wide spectrum of market values—from superstars like Neymar and Ronaldo to low-profile goalkeepers such as Antonio Sivera. Although deviations exist, the model avoided extreme overfitting or unrealistic estimations, showing that the system is not biased toward high-value outliers.

5. **Limitations are evident in unmodeled variables.**

Despite its strengths, the model cannot capture qualitative or external factors such as injury history, player marketability, contract clauses, or agent influence. These aspects often play a decisive role in market valuation but are not reflected in raw performance data.

In conclusion, Project 1 lays a solid foundation for predictive market value modeling, offering an accurate, generalizable, and interpretable approach. However, the results also reveal potential areas for improvement in preprocessing and feature engineering—gaps that Project 2 aims to address through an alternative model pipeline.

4.4 Quantitative Results (Project 2)

In the second experimental system developed in this thesis—Project 2—a new set of regression models was tested in order to provide an alternative approach to the player market value prediction task. The three models evaluated were: CatBoost Regressor, K-Nearest Neighbors (KNN), and Gradient Boosting Regressor. These models were selected with the goal of diversifying the modeling strategies and assessing whether more efficient or different algorithms could yield superior results.

The evaluation followed the same pipeline and dataset as in Project 1, ensuring comparability between experiments. Data preprocessing was identical, and the same train-test split was used to avoid sampling bias. The Root Mean Squared Error (RMSE) was again used as the main evaluation metric.

The results obtained in Project 2 were:

Model	**RMSE (€)**
CatBoost Regressor	8,454,431.43
Gradient Boosting Regressor	9,474,673.96

Model	RMSE (€)
K-Nearest Neighbors (KNN)	15,264,672.06

Table 2. RMSE results for the three regression models evaluated in Project 2. CatBoost achieved the best performance and was selected as the final model.

Once again, a boosting-based model achieved the best result. CatBoost delivered the lowest RMSE, slightly outperforming the XGBoost model used in Project 1 and outperforming Gradient Boosting in this setup. Its superior performance can be attributed to several factors:

- Built-in support for categorical variables (although one-hot encoding was used here),
- Efficient handling of missing values,
- Strong regularization that prevents overfitting.

On the other hand, KNN performed poorly, yielding the highest RMSE in this experiment. This confirms its limitations in high-dimensional, noisy datasets where Euclidean distance is not a reliable similarity measure.

As a result, the CatBoost Regressor was selected as the final model for Project 2 and saved for deployment. Its performance demonstrates that even small changes in model architecture can lead to measurable gains in accuracy, especially when handling complex relationships within performance metrics and player attributes.

4.5 Model Predictions and Examples (Project 2)

Once the CatBoost model was selected and saved as the final regressor for Project 2, it was deployed through the Streamlit interface to allow the prediction of market values for specific players. To ensure a consistent and comparable analysis, the same players evaluated in Project 1 were tested again under this new model. This section presents the predicted values alongside the actual 2020–21 market values and discusses the accuracy and patterns observed.

Cristiano Ronaldo

- **Predicted value (CatBoost)**: €61,271,717.02
- **Real value (2020–21)**: €60,000,000
- **Difference**: +€1.27M (~+2.1%)

Compared to Project 1, CatBoost shows improved accuracy in Ronaldo's case, reducing the error from over €6M to just above €1M. The model closely followed the downward valuation trend over the years while not overestimating the player's market longevity or residual brand power. This precise estimate indicates the robustness of CatBoost in handling structured data even for high-profile players.

Neymar Jr.

- **Predicted value (CatBoost)**: €143,255,747.31
- **Real value (2020–21)**: €128,000,000
- **Difference**: +€15.25M (~+11.9%)

While still within a tolerable range, the CatBoost model slightly overestimated Neymar's market value. Interestingly, the direction of the error is opposite to that in Project 1, where XGBoost underestimated it. This suggests that CatBoost may place more emphasis on consistent historical high-value levels, leading to a more aggressive forecast. However, this behavior also introduces potential risks of overvaluation in players with limited recent playing time due to injuries or external factors.

Antonio Rüdiger

- **Predicted value (CatBoost)**: €50,274,297.21
- **Real value (2020–21)**: €40,000,000
- **Difference**: +€10.27M (~+25.7%)

As with Project 1, the model overestimated Rüdiger's value. However, the error was slightly larger here, possibly due to the model's structure favoring players with strong recent statistics regardless of contractual or tactical factors. This example shows how predictive models can struggle to capture less quantifiable elements such as transfer strategies or individual visibility in the media.

Antonio Sivera

- **Predicted value (CatBoost)**: €546,000.90
- **Real value (2020–21)**: Not available

Even though no real value is recorded for this player, the CatBoost model again demonstrates stable behavior, producing a prediction close to his historical range. Compared to Project 1, where the estimate was around €942,100, CatBoost's prediction is significantly lower, suggesting that the model gives more weight to recent declines or inactivity.

These examples illustrate how Project 2's CatBoost model delivers more precise results in some cases (e.g., Ronaldo) but may also exhibit stronger variance in others (e.g., Neymar or Rüdiger). This reinforces the idea that no model is universally superior; instead, different approaches may offer advantages depending on the player's profile and data characteristics.

4.6 Project 2: Summary of Observations

Project 2 introduced a new set of regression models to assess whether alternative algorithms could improve upon the performance and generalization capabilities of Project 1. The following key insights emerged from this second experimental system:

1. **CatBoost outperformed other models and surpassed XGBoost.**

The CatBoost Regressor not only delivered the best RMSE in Project 2 (8.45M) but also slightly improved upon the best model in Project 1 (XGBoost, 8.82M). While the improvement margin was modest, it is statistically relevant in large-scale predictive systems and suggests CatBoost's superior handling of feature interactions and regularization in this dataset.

2. **KNN confirmed its limitations for this type of data.**

The K-Nearest Neighbors (KNN) model exhibited the highest RMSE of all tested models across both projects. This was expected, as KNN struggles in high-dimensional spaces, is sensitive to irrelevant or noisy features, and lacks any internal mechanism for feature weighting or interaction modeling. Its performance reinforces the necessity of using models with embedded complexity control in real-world applications.

3. Gradient Boosting delivered solid, but not leading, performance.

While Gradient Boosting Regressor performed better than KNN and relatively close to XGBoost and CatBoost, it did not provide enough justification to replace either. Its results suggest that boosting frameworks remain highly competitive, but newer variants like CatBoost offer implementation advantages and slight performance boosts.

4. Player-level predictions showed more accurate behavior in some cases.

For example, the CatBoost model reduced the prediction error for Cristiano Ronaldo by over 80% compared to Project 1, demonstrating improved individual calibration. However, in other cases, such as Neymar and Rüdiger, the model produced slightly more aggressive predictions, leading to higher deviation. This behavior indicates that CatBoost may weigh historical trends more heavily than performance plateaus.

5. Model simplicity and performance coexisted well.

Project 2 maintained the same pipeline structure and preprocessing strategy as Project 1, demonstrating that it is possible to swap models without increasing pipeline complexity. CatBoost integrates easily with existing Python environments and requires no special tuning to outperform more complex systems like SVR or manually tuned Random Forests.

In summary, Project 2 confirmed that CatBoost is a competitive, high-performing alternative to traditional boosting models for market value prediction in football. While its advantages are not dramatic, they are consistent and justify its selection as the final model for comparative analysis.

4.7 Comparative Analysis Between Projects

Having completed the development and evaluation of both Project 1 and Project 2, this section presents a comparative analysis of their respective strengths, weaknesses, and overall performance. The aim is to determine which model is better suited for real-world implementation in the task of football player market value prediction.

1. Predictive Accuracy

The comparison of RMSE values reveals a slight but consistent advantage for Project 2:

Project	**Best Model**	**RMSE (€)**
Project 1	XGBoost	8,816,509.86
Project 2	CatBoost	8,454,431.43

Table 3. Comparison of the best-performing models in each project based on RMSE. CatBoost (Project 2) achieved the lowest error overall.

Although the margin (~€360,000) may seem modest, it is statistically meaningful in a predictive context involving high-value targets. Project 2's CatBoost model therefore demonstrates greater accuracy in estimating market values across a diverse set of players.

2. Prediction Consistency and Behavior

Both models performed well on high-profile cases like Cristiano Ronaldo, Neymar, and Antonio Rüdiger. However:

- CatBoost (Project 2) produced more accurate predictions for Ronaldo and Sivera, but tended to overestimate top-tier players (e.g., Neymar).
- XGBoost (Project 1) showed more conservative estimates overall, reducing the risk of overvaluation but underestimating peak players in some cases.

This suggests that XGBoost might be preferred in risk-averse scenarios, while CatBoost excels when aggressive precision is needed.

3. Computational Efficiency and Implementation

Both projects followed the same architecture and preprocessing steps. However:

- XGBoost is more mature and widely adopted in production environments.
- CatBoost, while newer, handled categorical data more naturally and required fewer parameter adjustments to achieve strong results.

From a developer perspective, CatBoost integrates seamlessly and simplifies deployment without sacrificing performance.

4. Generalization to Lower-Profile Players

In both projects, lower-profile players like Antonio Sivera received predictions aligned with their historical range, demonstrating good generalization. However, CatBoost produced a notably lower and more cautious prediction for this player than XGBoost, potentially signaling a better treatment of underrepresented or declining cases.

Recommendation

Based on the comparative evaluation, Project 2's CatBoost model is recommended for the following reasons:

- It achieved the lowest RMSE overall.
- It delivered more precise results for top players, especially in cases with nonlinear historical trends.
- It showed good generalization across the value spectrum, from elite players to modest profiles.

That said, XGBoost remains a reliable and explainable alternative, especially in environments where interpretability, transparency, or model explainability tools are required (e.g., SHAP support, which is better documented for XGBoost).

Chapter 5 CONCLUSION

This thesis sets out to explore and evaluate the effectiveness of machine learning models in predicting the market value of professional football players. Through the development of two independent systems—Project 1 and Project 2—the study aimed to identify which algorithms, and preprocessing strategies deliver the most accurate and generalizable results using real player performance data.

Key Findings

1. **Machine learning is a viable tool for player valuation:** The experiments confirmed that structured machine learning models can provide meaningful and consistent estimates of player market value using historical statistics, player attributes, and contextual features. Even without access to contractual or qualitative data, both systems produced predictions that aligned closely with real-world valuations.
2. **Ensemble boosting models outperform classical techniques:** Both XGBoost and CatBoost significantly outperformed Support Vector Regression and K-Nearest Neighbors in terms of RMSE. These results align with the current state of the art, which favors boosting algorithms for tabular data with complex feature interactions.
3. **CatBoost yielded the best overall performance**: With an RMSE of approximately €8.45 million, CatBoost slightly outperformed XGBoost and showed improved precision on individual high-value players such as Cristiano Ronaldo. Its regularization and stability make it a robust option for football analytics applications.

4. **Preprocessing decisions matter**: .

   The use of mean imputation in Project 2, as opposed to zero-filling in Project 1, likely contributed to improved generalization in the final model. This highlights the importance of preprocessing strategy when building real-world prediction systems.

5. **User-facing implementation adds real value**: .

   Both systems were deployed as interactive web applications using Streamlit. These apps allowed users to explore the data, train models, and predict player values, demonstrating how machine learning tools can be translated into practical solutions for sports management, scouting, or media analysis.

Limitations

- The models rely exclusively on publicly available structured data, and thus cannot incorporate more subjective or qualitative factors such as injury risk, player popularity, or contractual clauses.
- The absence of extensive hyperparameter tuning or ensemble stacking may limit performance gains, although simplicity and reproducibility were prioritized.
- The real market value dataset used for validation (2020–21 season) may not fully reflect transfer market volatility or off-field variables influencing valuations.

Future Work

This thesis opens several paths for further research and development:

- Integration of unstructured data: Incorporating text-based scouting reports, social media sentiment, or match commentary could enrich the feature set.
- Advanced model interpretability: Implementing SHAP values or LIME could enhance model transparency and support decision-making.
- Time-series forecasting: Instead of static predictions, future models could attempt to forecast the evolution of player value over time.
- Real-time applications: Connecting the system with live data sources could allow clubs or analysts to receive automatic valuation updates during a season.

In conclusion, this work demonstrates the feasibility and potential of using machine learning to model and predict the market value of football players. The two systems developed offer not only strong predictive performance but also a foundation for building accessible, data-driven tools for sports professionals.

References

[1] [En línea]. Available: https://www.kaggle.com/datasets/kriegsmaschine/soccer-players-values-and-their-statistics.

[2] [En línea]. Available: https://www.kaggle.com/datasets/akarshsinghh/football-players-market-value-prediction.

[3] [En línea]. Available: https://www.sciencedirect.com/science/article/pii/S0960077922012589.

[4] [En línea]. Available: https://www.sciencedirect.com/science/article/pii/S2210832717301485.

[5] [En línea]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0169207018300116.

[6] [En línea]. Available: https://pubmed.ncbi.nlm.nih.gov/35395047/.

[7] [En línea]. Available: https://arno.uvt.nl/show.cgi?fid=161188.

[8] [En línea]. Available: https://medium.com/datos-y-ciencia/proyecto-machine-learning-predicci%C3%B3n-de-precios-de-viviendas-en-boston-con-regresi%C3%B3n-e8655e6c3655.

[9] [En línea]. Available: https://github.com/RSKriegs/Modelling-Football-Players-Values-on-a-Transfer-Market/blob/main/RS82640%20Modelling%20Footballers%20Values%20on%20a%20Transfer%20Market%20.pdf.

[10] [En línea]. Available: https://link.springer.com/article/10.1007/s10489-024-06189-0.

[11] [En línea]. Available: https://github.com/RSKriegs/Modelling-Football-Players-Values-on-a-Transfer-Market/blob/main/RS82640%20Modelling%20Footballers%20Values%20on%20a%20Transfer%20Market%20.pdf.

[12] [En línea]. Available: https://link.springer.com/article/10.1007/s10489-024-06189-0.

[13] [En línea]. Available: https://www.tandfonline.com/doi/full/10.1080/23750472.2025.2459727.

[14] [En línea]. Available: https://www.python.org/.

[15] [En línea]. Available: https://pandas.pydata.org/.

[16] [En línea]. Available: https://numpy.org/.

[17] [En línea]. Available: https://scikit-learn.org/stable/.

[18] [En línea]. Available: https://xgboost.readthedocs.io/en/stable/.

[19] [En línea]. Available: https://catboost.ai/.

[20] [En línea]. Available: https://joblib.readthedocs.io/en/latest/.

[21] [En línea]. Available: https://streamlit.io/.

[22] [En línea]. Available: https://matplotlib.org/.

[23] [En línea]. Available: https://seaborn.pydata.org/.

[24] [En línea]. Available: https://www.transfermarkt.com/.

[25] [En línea]. Available: https://fbref.com/en/.

[26] [En línea]. Available: https://code.visualstudio.com/.

[27] [En línea]. Available: https://www.researchgate.net/publication/349838505_Predicting_Market_Value_of_Football_Players_Using_Machine_Learning.

Fort Hays State University

FHSU Scholars Repository

Non-Exclusive License Author Agreement

I hereby grant Fort Hays State University an irrevocable, non-exclusive, perpetual license to include my thesis ("the Thesis") in FHSU Scholars Repository, FHSU's institutional repository ("the Repository"). I hold the copyright to this document and agree to permit this document to be posted in the Repository, and made available to the public in any format in perpetuity. I warrant that the posting of the Thesis does not infringe any copyright, nor violate any proprietary rights, nor contains any libelous matter, nor invade the privacy of any person or third party, nor otherwise violate FHSU Scholars Repository policies. I agree that Fort Hays State University may translate the Thesis to any medium or format for the purpose of preservation and access. In addition, I agree that Fort Hays State University may keep more than one copy of the Thesis for purposes of security, back-up, and preservation. I agree that authorized readers of the Thesis have the right to use the Thesis for non commercial, academic purposes, as defined by the "fair use" doctrine of U.S. copyright law, so long as all attributions and copyright statements are retained. To the fullest extent permitted by law, both during and after the term of this Agreement, I agree to indemnify, defend, and hold harmless Fort Hays State University and its directors, officers, faculty, employees, affiliates, and agents, past or present, against all losses, claims, demands, actions, causes of action, suits, liabilities, damages, expenses, fees and costs (including but not limited to reasonable attorney's fees) arising out of or relating to any actual or alleged misrepresentation or breach of any warranty contained in this Agreement, or any infringement of the Thesis on any third party’s patent, trademark, copyright or trade secret. I understand that once deposited in the Repository, the Thesis may not be removed.

Thesis: **Comparative Study of Machine Learning Models for Predicting the Market Value of Professional Football Players**

Author: **Álvaro Salvador López**

Signature: Álvaro

Date: **07/29/2025**