PROJECT REPORT

TransferIQ: Dynamic Player Transfer Value Prediction using AI and Multi-source Data

Milestone 1: Data Collection, Merging, and Cleaning

In modern football, clubs, scouts, and analysts want to answer one key question:

Which players deliver the best performance and value, while minimizing risk from injuries and negative public sentiment?

Raw data is scattered across many sources:

- Player profiles
- Match performances
- Market value history
- Injuries
- National team stats
- Fan sentiment from tweets

On their own, each dataset tells only part of the story.

Our goal in Milestone 1 was to bring all these pieces together into one trusted dataset that reflects a player's performance, value, fitness, and perception — the foundation for building a reliable ML model later.

1. Objective

The objective of this milestone is to create a single, reliable, and analysis-ready dataset by integrating multiple football-related data sources. In modern football, decision-makers need more than just performance numbers — they also consider a player's **market value, injury history, and public sentiment**.

By combining these aspects, we aim to support:

- Player performance evaluation
- Market value prediction
- Injury risk assessment
- Sentiment impact analysis

This unified dataset acts as a **"single source of truth"**, ensuring that all future machine learning models are trained on consistent and meaningful data. A strong data foundation is critical because poor-quality input data would directly lead to unreliable predictions.

Visualization used:

Bar charts showing number of players per dataset and missing value plots to understand data completeness.

2. Data Collection

Description

Multiple datasets were collected, each capturing a different dimension of a footballer's profile:

- **Player profiles** → demographics like name, nationality, date of birth
- **Match performances** → goals, assists, minutes played, matches
- **Market values** → financial worth (value_x)
- **Injury history** → injury start and end dates (from_date, end_date)
- **National stats** → international exposure
- **Tweet sentiment** → fan and media perception

Each dataset alone provides partial information, but together they give a **360-degree view** of a player.

Why this was done

Relying only on performance ignores business risk (injuries) and public perception (sentiment), which are crucial in real-world transfer decisions. Collecting diverse data ensures the model reflects how clubs actually evaluate players.

Outcome

- All datasets successfully loaded
- Coverage across technical, medical, financial, and social dimensions

Visualization used:

Dataset-wise record count comparison and sample tables to verify data loading.

---

3. Data Merging

Description

All datasets were merged using **player_id** as the primary key, with the player profile table as the base. Tweet data, originally based on names, was cleaned and mapped to player_id and then aggregated at player level before merging.

An **inner join** strategy was applied while combining tables.

Why this method

- Using player_id ensures accurate mapping across sources.
- Inner joins keep only those players who have **complete information** across performance, value, injuries, and sentiment.
- This avoids training models on incomplete or misleading profiles.

What change it brings

After merging, scattered data becomes a **unified dataset** where each row represents a complete player profile.

Outcome

- Unified dataset with ~75,000 records and 73 features
- Saved as **merged_data_initial.csv**
- Created a single trusted dataset for analysis

Visualization used:

Before–after row count comparison and schema overview of merged features.

---

4. Data Cleaning

Description

Cleaning focused on:

- Converting date fields like from_date, end_date, and date_of_birth into proper datetime format
- Removing identifiers such as player_id and player_name from model inputs

- Standardizing data types and handling inconsistencies

Derived variables like **days_out** (injury duration) and **age** were later created from dates.

Why this was done

- IDs and names do not add predictive value and may cause overfitting.
- Date values are difficult for models to interpret directly, but derived metrics like age are meaningful.
- Cleaning ensures consistency and prevents misleading patterns.

What change it brings

Transforms messy raw data into structured, model-friendly data.

Outcome

- Consistent formats across all columns
- Reduced noise and redundancy
- Improved data quality for feature engineering

Visualization used:

Missing value heatmap and boxplots before and after cleaning to show noise reduction.

---

5. Feature Engineering

Description

New business-relevant features were created from raw variables:

- **age** → career stage
- **days_out** → injury severity
- **goals_per_90_min**, **assists_per_90_min** → normalized performance
- **G_A_per_match** → attacking contribution
- **Normalized_sentiment** → public perception
- **Injury_Impact_Index** → combines injury duration with market value (value_x)
- **Value_Efficiency_Ratio** → performance relative to cost

Why this was done

Raw totals (like goals) are biased by playtime. Normalized features make players comparable. Combining injury and value quantifies **financial risk**, which is important for transfer decisions.

What change it brings

Converts raw statistics into **actionable signals** aligned with how clubs judge players.

Outcome

- Business-meaningful predictors created
- Improved explanatory power of dataset
- Raw data transformed into decision-ready features

Visualization used:

Distributions of age, goals_per_90_min, and correlation heatmap to show feature relevance.

---

6. Data Preprocessing Pipeline

Description

A preprocessing pipeline was applied to:

- Handle missing numerical values using **median imputation**
- Scale numeric features using **standardization**
- Encode categorical variables using **one-hot encoding**

Numerical features and categorical features were processed separately and then combined.

Why this method

- Median is robust to outliers in football stats.
- Scaling ensures features like age and value_x contribute fairly.
- One-hot encoding converts categories into numeric form usable by ML models.
- A pipeline ensures **reproducibility** and avoids data leakage.

What change it brings

Produces a clean, uniformly scaled, and encoded feature matrix ready for model training.

Outcome

- Final processed dataset saved as **cleaned_processed_data.csv**
- Fully model-ready input created
- Production-style workflow established

Visualization used:

Feature scaling comparison plots and category distribution after encoding.

---

7. Milestone 1 Summary

In this milestone, multiple football-related datasets were collected and integrated into a unified dataset using player_id as the common key. The data was cleaned, standardized, and enhanced with engineered features such as age, goals_per_90_min, days_out, and Injury_Impact_Index to reflect real-world evaluation criteria.

A robust preprocessing pipeline handled missing values, scaling, and encoding, resulting in a high-quality dataset: **cleaned_processed_data.csv**. This dataset forms a strong and trustworthy foundation for exploratory analysis and machine learning model development in the next milestones.

---

8. Key Deliverable

A unified, business-aligned, and model-ready dataset that ensures future predictions for:

- Market value estimation
- Performance analysis
- Injury risk assessment

  are built on reliable and meaningful data.

Milestone 2: Exploratory Data Analysis (EDA) and Advanced Feature Engineering

1. Objective

The objective of Milestone 2 is to deeply analyze the merged dataset to understand patterns, relationships, and data quality, and then apply advanced feature engineering and outlier handling techniques to create meaningful predictors for model training. This milestone transforms raw football, injury, and sentiment data into a refined, analysis-ready dataset.

---

2. Exploratory Data Analysis (EDA)

Description

EDA was performed on the merged dataset (analysis_df) to study the distribution of key variables and their relationship with the target variable **value_x** (player market value). The analysis focused on performance metrics, age, injury impact, and positional differences.

Key numerical features such as age, matches, minutes_played, G_A_per_match, days_out, and value_x were explored.

Why this was done

- To understand which factors influence market value.

- To detect skewness, anomalies, and relationships.
- To guide feature engineering and model choice.

Visualizations Used
- **Correlation heatmap** of major numerical features to identify strong positive or negative relationships with value_x.
- **Bar plot** of average market value by position to study positional demand.
- **Scatter plot** of G_A_per_match vs value_x segmented by position to analyze performance impact.
- **Age vs value trend plots** to observe career curves across positions.

These visualizations revealed that performance metrics such as G_A_per_match have strong positive correlation with market value, while age shows position-dependent trends.

Key Insights
- **Performance dominance:** G_A_per_match is highly correlated with value_x, showing that on-field output is the primary driver.
- **Positional value:** Forwards and midfielders command higher average market values.
- **Injury complexity:** The engineered injury metric later showed a strong relationship with value.
- **Sentiment weakness:** Raw polarity scores alone had weak correlation with market value.

Outcome

Clear understanding of important drivers of player valuation and direction for feature engineering.

---

3. Initial Outlier Detection and Handling

Description

Outliers were analyzed in key columns:
- value_x
- days_out
- G_A_per_match
- Injury_Impact_Index

Box plots were generated before treatment to visualize extreme values.

An **IQR (Interquartile Range) method** was used to cap outliers instead of removing rows. For each column, values beyond:
- lower bound = Q1 − 1.5 × IQR
- upper bound = Q3 + 1.5 × IQR
  were clipped.

This was applied first on base features like value_x and days_out, and later again on derived features.

Why this was done
- Football market values and injury days can have extreme but meaningful values.
- Removing rows could lose important elite players.
- Capping controls extreme influence while preserving data size.

Visualizations Used
- Box plots **before and after capping** for value_x and days_out.
- Box plots for derived features after base capping to ensure stability.

What change it brings
- Reduces skewness and extreme leverage.
- Stabilizes learning for ML models.

Outcome

A cleaner dataset with controlled distributions for critical features.

---

4. Advanced Feature Engineering
Using the capped dataset (df_outlier_analysis), several new domain-driven features were created.
a) Age Calculation (age)
Computed from date_of_birth using the current year.
Why: Player value strongly depends on age and career stage.
Outcome: A continuous age feature for modeling.

---

b) Goals + Assists per Match (G_A_per_match)
Calculated as:
(goals_x + assists) / matches
Zeros in matches were replaced to avoid division errors.
Why: Normalizes performance across players with different match counts.
Benefit: Fair comparison of attacking contribution.
Outcome: A key performance intensity metric.

---

c) Per 90 Minutes Metrics
- goals_per_90_min
- assists_per_90_min

Derived using minutes played.
Why: Per-90 metrics reflect true efficiency rather than total counts.
Outcome: More realistic performance indicators.

---

d) Log Injury Days (log_days_out)
Applied log1p transform on days_out.
Why: Injury days are right-skewed; log scaling reduces skewness.
Outcome: Stabilized injury duration feature.

---

e) Injury Impact Index (Injury_Impact_Index)
Calculated as:
(log_days_out × value_x) / matches
Why: Combines injury duration with player value and activity level.
Benefit: Measures how impactful injuries are for valuable players.
Outcome: A powerful hybrid feature capturing risk vs value.

---

f) Normalized Sentiment (normalized_sentiment)
Derived from vader_polarity using min-max normalization.
Why: Brings sentiment into a 0–1 scale for compatibility with ML models.
Outcome: Comparable sentiment feature across players.

---

g) Value Efficiency Ratio (Value_Efficiency_Ratio)
Calculated as:
value_x / G_A_per_match
Why: Captures how much market value corresponds to on-field output.
Benefit: Highlights over- or under-valued players.
Outcome: Efficiency-based valuation feature.

---

5. Derived Feature Distribution Analysis

Histograms and box plots were generated for:

- G_A_per_match
- Injury_Impact_Index
- Value_Efficiency_Ratio

These showed heavy skewness and extreme values, leading to **IQR-based capping** again on derived features.

Why this was done

- Engineered features can amplify noise.
- Second-level outlier control ensures stability.

Outcome

Final engineered features with controlled ranges.

---

6. Refining the Feature Set

A refined dataset **df_refined_features** was created with 20 core columns:

Numerical:

- age, value_x, matches, goals_x, assists, minutes_played,
  days_out, vader_polarity, tb_polarity,
  goals_per_90_min, assists_per_90_min, G_A_per_match,
  normalized_sentiment, Injury_Impact_Index,
  Value_Efficiency_Ratio, log_days_out

Categorical:

- citizenship, position, current_club_name, injury_reason

Why this was done:

To keep only meaningful, engineered, and interpretable variables before encoding.

Outcome:

A compact yet information-rich feature table of shape **(75,731, 20)**.

---

7. Sentiment Text Vectorization

Tweet text related to players was processed using **CountVectorizer** with:

- maximum 1000 features
- English stop words removed

This produced word-count features aggregated by player_name_upper, forming a sentiment feature matrix of shape **(521, 1000)**.

Why this was done

- Raw polarity scores were weak.
- Text patterns may capture richer public perception.
- Converts unstructured text into numeric predictors.

Outcome

High-dimensional sentiment features ready to be merged into the main dataset.

---

8. Reapplying the Data Preprocessing Pipeline

With refined numerical, categorical, and sentiment features, the preprocessing pipeline was reapplied:

- Numerical features → scaling and imputation.
- Categorical features → one-hot encoding.
- Sentiment vector features → treated as numerical inputs.

The final processed matrix **X_processed_final** was created using a ColumnTransformer and saved as: **cleaned_processed_data_final.csv**

Final shape reported: **(75,731, 1,043)** features.

Why this was done

- To ensure all new features are consistently transformed.
- To make the dataset fully ML-ready.

Outcome

A single, high-dimensional processed dataset ready for model training.

Milestone 2 Summary

In Milestone 2, comprehensive exploratory data analysis was performed to understand relationships between player performance, age, injuries, sentiment, and market value. Visualizations such as correlation heatmaps, bar plots, and scatter plots revealed that G_A_per_match and positional roles strongly influence valuation, while raw sentiment has limited impact. Outliers in critical variables were controlled using IQR-based capping to stabilize distributions. Advanced feature engineering created meaningful metrics such as goals_per_90_min, G_A_per_match, Injury_Impact_Index, normalized_sentiment, and Value_Efficiency_Ratio, combining domain knowledge with statistical transformations. High-dimensional sentiment features were extracted using CountVectorizer, and all numerical and categorical features were reprocessed through a unified pipeline to produce **cleaned_processed_data_final.csv** with 1,043 features. This milestone delivered a robust, enriched, and ML-ready dataset that directly enabled accurate model building in the next stage.

Milestone 3: Model Building, Training, and Feature Selection

1. Objective

The objective of Milestone 3 is to build, train, and compare machine learning models to accurately predict a football player's market value (**value_x**) using the processed dataset. This milestone focuses on selecting suitable algorithms, resolving overfitting issues, and identifying the most important features that influence player valuation.

---

2. Defining Features and Target Variable

From the final processed dataset:

- **Feature matrix (X)** consists of player attributes such as age, matches, minutes_played, goals_per_90_min, assists_per_90_min, G_A_per_match, Injury_Impact_Index, Value_Efficiency_Ratio, and normalized_sentiment.
- **Target variable (y)** is the player's latest market value stored as **value_x**.

This step corresponds to selecting columns for X and separating value_x as y.

Why this was done

Machine learning models require a clear separation between predictors and the value to be predicted. Keeping value_x only in y prevents the model from indirectly learning the answer.

Outcome

A clean dataset ready for model training.

---

3. Train–Test Split

The dataset was divided into:

- **Training set (X_train, y_train)**

- **Testing set (X_test, y_test)**

An 80–20 split was applied using a fixed random state.

Why this was done

This allows the model to learn from X_train and be evaluated on unseen data X_test, ensuring that performance reflects real-world generalization.

What change it brings

Creates a fair and reproducible evaluation setup.

Outcome

Independent training and testing datasets saved for experiments.

---

4. Baseline Model: Linear Regression and Overfitting Detection

Description

A **Linear Regression** model was first trained using X_train and y_train. The model learns linear weights for each feature to predict value_x. Predictions (y_pred) were generated on X_test, and metrics such as **R^2, MAE, MSE, and RMSE** were computed by comparing y_pred with y_test.

This reflects the steps of initializing the model, fitting it, predicting, and evaluating.

Observed Result

The model produced:

- **R^2 close to 1.0**
- **MAE and RMSE close to 0**

Why this was a problem

Such perfect performance is unrealistic and indicated **overfitting due to data leakage**, meaning:

- Some form of value_x or closely related value information had leaked into X.
- The model was effectively seeing the answers during training.

Why Linear Regression was used

- To set a baseline.
- To quickly detect issues in preprocessing.
- To understand whether linear relationships exist.

Action Taken

To fix this issue:

- The preprocessing pipeline was revisited.
- value_x was strictly removed from all feature columns.
- Merging logic was corrected to avoid duplicate value fields.
- Feature engineering and scaling were reapplied.
- New datasets (X_train_final_processed, X_test_final_processed, y_train_final_processed, y_test_final_processed) were generated.

What change this brought

- Removed data leakage.
- Restored realistic learning conditions.
- Ensured trustworthy future results.

Outcome

A corrected dataset suitable for proper modeling.

---

5. Lasso Regression: Regularization and Feature Selection

Description

After correction, **Lasso Regression** was trained using X_train_final_processed and y_train_final_processed. Lasso adds an L1 penalty that shrinks weak feature coefficients toward zero while fitting the model to predict value_x.
This corresponds to initializing the Lasso model, fitting it, and using it to predict on X_test_final_processed.

Why this method

- The dataset contains hundreds of features after encoding.
- Lasso controls model complexity.
- Performs **automatic feature selection**.

What change it brings

Eliminates noisy or irrelevant features and keeps important ones such as:
age, goals_per_90_min, G_A_per_match, Injury_Impact_Index, and normalized_sentiment.

Outcome

- More stable linear model than plain regression.
- Reduced feature space for tree-based models.

---

6. Decision Tree Regressor (Final Model)

Description

A **Decision Tree Regressor** was trained using X_train_final_processed and y_train_final_processed. The tree recursively splits data based on feature thresholds to minimize prediction error for value_x.
Predictions were generated on X_test_final_processed and compared with y_test_final_processed.
This corresponds to model initialization, fitting, predicting, and evaluating.

Why this method

- Captures **non-linear relationships**.
- Learns feature interactions automatically.
- Produces interpretable if–else decision rules.

Model Performance

The Decision Tree achieved:

- **R^2 = 0.9686**

What this means

- Explains about **96.86%** of the variance in player market value.
- Average prediction error is around **€29.9k**, which is low compared to typical market values.
- Indicates excellent predictive capability.

What change it brings

Significantly improves accuracy compared to linear models by capturing complex patterns between performance, injury, and sentiment features.

Outcome

The **Decision Tree Regressor** was selected as the **final model** due to its highest accuracy.

---

7. Random Forest Regressor (Comparison Model)

Description

A **Random Forest Regressor** was also trained as an ensemble of many decision trees using the same training data. Predictions were averaged to estimate value_x.

Why this method

- Reduces overfitting of single trees.
- Generally improves stability.

Model Performance
The Random Forest achieved:
- **$R^2 = 0.9394$**

Outcome
Although robust, its accuracy was lower than the Decision Tree, so it was not chosen as the final model.

---

8. Feature Importance and Selection
Using Random Forest, feature importance scores were extracted to understand which variables contribute most to predicting the target **value_x**. The model assigns higher importance to features that reduce prediction error the most while splitting the data.
To interpret these results clearly, a **bar chart visualization of the top 20 most important features** was created.
The most influential features identified included:
- age
- matches, minutes_played
- goals_per_90_min, assists_per_90_min
- G_A_per_match
- Injury_Impact_Index
- Value_Efficiency_Ratio
- normalized_sentiment

Why this was done
- To make the model's decisions transparent and explainable.
- To verify that predictions are driven by meaningful football and business factors.
- To reduce the feature space by focusing on the most impactful predictors.
- To communicate results clearly to non-technical stakeholders using visualization.

What change it brings
Feature selection based on importance:
- Removes less useful and noisy features.
- Improves interpretability of the model.
- Helps in building a simpler and more robust final system.

Outcome
A ranked list of the **top 20 most important features**, supported by a visualization, which confirms that player age, normalized performance metrics, injury impact, and sentiment are the key drivers of market value prediction.

Milestone 3 Summary
In Milestone 3, the dataset was split into training and testing sets with value_x as the target. A baseline Linear Regression model revealed severe overfitting due to data leakage, leading to a full correction of preprocessing and retraining. Lasso Regression was then applied to regularize the model and perform feature selection. A Decision Tree Regressor captured complex non-linear relationships and achieved the best performance with **$R^2 = 0.9686$**, outperforming the Random Forest model (**$R^2 = 0.9394$**). Based on accuracy and interpretability, the Decision Tree was selected as the final model. Feature importance analysis, visualized through the top 20 features, confirmed that age, normalized performance, injury impact, and sentiment are the primary drivers of player market value.

Milestone 4: Model Evaluation and Hyperparameter Tuning

1. Objective

The objective of Milestone 4 is to rigorously evaluate the trained models on unseen data and further improve their performance using hyperparameter tuning. This milestone ensures that the selected model is not only accurate on training data but also generalizes well to new players, making it reliable for real-world deployment.

2. Final Evaluation on Untouched Test Set (Decision Tree Regressor)

Description

After selecting the Decision Tree Regressor as one of the strongest models in Milestone 3, the previously trained model (dt_model) was used to make predictions on a completely **untouched test set** (X_test_final). This dataset had not been used during training or validation.

Predictions (y_pred_dt_final) were compared against the true values (y_test_final) to compute:

- **R^2 score**
- **Mean Absolute Error (MAE)**
- **Mean Squared Error (MSE)**
- **Root Mean Squared Error (RMSE)**

This corresponds to loading the final test data, generating predictions, and calculating evaluation metrics.

Why this was done

- To simulate real-world performance on new, unseen players.
- To verify that the model is not overfitting.
- To obtain a trustworthy estimate of final accuracy.

Observed Result

For the untouched test set with shape:

- X_test_final: (23,069, 408)
- y_test_final: (23,069, 1)

The Decision Tree achieved:

- **R^2 = 0.9842**
- **MAE = 14,499.33**
- **MSE = 10,626,989,683.12**
- **RMSE = 103,087.29**

What this means

- The model explains about **98.42%** of the variance in player market value.
- On average, predictions differ from actual values by only about **€14.5k**.
- The low RMSE indicates strong accuracy even for high-value players.

Outcome

The Decision Tree Regressor demonstrated **excellent generalization** and very high predictive performance on unseen data.

3. Hyperparameter Tuning using Grid Search

Description

To further improve the Decision Tree model, **hyperparameter tuning** was performed using GridSearchCV. The training data (X_train, y_train) and a separate validation set (X_val, y_val) were used.

A grid of parameters was defined, including:

- maximum tree depth (max_depth)
- minimum samples required to split a node (min_samples_split)
- minimum samples in a leaf (min_samples_leaf)
- number of features considered at each split (max_features)

Grid search trained multiple Decision Tree models using different combinations and evaluated them using **3-fold cross-validation**, with **MAE** as the primary scoring metric.

Why this was done

- Decision Trees are sensitive to depth and split rules.
- Improper settings can cause overfitting or underfitting.
- Grid search systematically finds the **best parameter combination**.

What change it brings

Optimizes the structure of the tree to balance:

- learning capacity
- generalization ability
- prediction stability

Outcome of Grid Search

From 108 parameter combinations, the best parameters found were:

- max_depth = 20
- max_features = 1.0
- min_samples_leaf = 1
- min_samples_split = 2

The best cross-validated score achieved (negative MAE) was approximately **−31,756**, indicating low average error.

---

4. Evaluation of Tuned Decision Tree on Validation Set

Description

The best estimator (best_dt_model) obtained from grid search was then used to make predictions on the validation set (X_val). These predictions (y_pred_best_dt) were compared with y_val.

Why this was done

- To verify that tuning genuinely improves performance.
- To ensure that gains are consistent beyond training folds.

Observed Result

On the validation set, the tuned Decision Tree achieved:

- **R^2 = 0.9793**
- **MAE = 24,830.43**
- **MSE = 14,123,691,713.46**
- **RMSE = 118,843.14**

What this means

- Explains about **97.93%** of variance in player value.
- Average error reduced compared to untuned versions.
- Confirms strong and stable model performance.

Outcome

The tuned Decision Tree model showed **consistent high accuracy** across validation and test data.

---

5. Comparison with LightGBM and Final Model Decision

Description

In parallel, a **LightGBM Regressor** model was also trained and evaluated. It achieved around **97% accuracy** on both training and test sets.

However:

- On validation and untouched evaluation,
- The **Decision Tree model achieved ~98% R^2**, outperforming LightGBM in this context.

Why this comparison was important

- To ensure the chosen model is not only powerful but also stable across different data splits.
- To avoid selecting a model that performs well only on specific subsets.

Final Decision

Although LightGBM showed strong performance, the **Decision Tree Regressor** was finally selected because:

- It achieved the **highest R^2 (~98.4%)** on the untouched test set.
- It showed consistent results on validation data.
- It offers better interpretability for feature importance.

Outcome

The **tuned Decision Tree Regressor** was finalized as the deployment model.

---

6. Final Model Saving and Readiness for Deployment

Description

The optimized Decision Tree model was retrained on the full training data and prepared to be saved for use in the backend API and Streamlit application.

Why this was done

- To ensure the deployed model uses the best learned patterns.
- To integrate seamlessly into the prediction pipeline.

Outcome

A single, optimized, and validated Decision Tree model ready for production use.

Milestone 4 Summary

In Milestone 4, the Decision Tree Regressor was rigorously evaluated on an untouched test set, achieving excellent performance with **R^2 = 0.9842**, **MAE ≈ 14.5k**, and **RMSE ≈ 103k**, confirming strong generalization. Hyperparameter tuning using GridSearchCV optimized key parameters such as tree depth and split criteria, resulting in a tuned model that achieved **R^2 = 0.9793** on the validation set. Although a LightGBM model also achieved around **97% accuracy**, the tuned Decision Tree consistently outperformed it on validation and final evaluation data and was therefore selected as the final model. This milestone ensured that the system is accurate, robust, and ready for deployment in the final application.

Overall Project Summary

This project aims to develop an intelligent machine learning system to accurately predict the **market value of football players** by integrating performance statistics, injury history, and public sentiment. In modern football, player valuation is influenced not only by on-field performance but also by fitness and perception. However, relevant data is scattered across multiple sources, making holistic analysis challenging. This project addresses that gap by building a unified, data-driven valuation framework.