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

Infosys SpringBoard Virtual Internship

Submitted By:

Sai pujitha

pavuluri

Batch- 1

Project Mentor-

Pranaya mam

(August-October) 2025

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to **Infosys SpringBoard** and the **Infosys SpringBoard Virtual Internship Program** for providing me with the opportunity to work on the project titled ''**TransferIQ: Dynamic Player Transfer Value Prediction using AI and Multi-source Data** ''. This project has been a valuable learning experience that enhanced my understanding of web application security and the practical implementation of enumeration tools.

I am especially thankful to my project mentor(Pranayaa) for their constant support, guidance, and encouragement throughout the internship. Their feedback and insights greatly contributed to the successful completion of this project.

I also extend my appreciation to the Infosys SpringBoard community for providing resources, technical assistance, and a collaborative environment for learning and development.

Lastly, I thank my peers, family, and friends for their motivation and moral support during the course of this internship.

Sai Pujitha pavuluri
Project Trainee (August–October 2025)
Infosys SpringBoard

Introduction

Player transfer markets play a pivotal role in shaping the landscape of professional sports, particularly football, where the acquisition and sale of players directly influence a team's performance, financial stability, and long-term competitiveness. Transfers are not only about strengthening squads but also about maximizing financial returns, managing player contracts, and strategically positioning clubs in domestic and international leagues. Consequently, accurately estimating a player's transfer value has become a crucial yet highly complex task for clubs, agents, and investors.

A player's market value is determined by a combination of multiple interdependent factors. Traditional elements such as age, position, current club, and performance statistics (goals, assists, minutes played, etc.) are fundamental. However, modern transfer markets are increasingly influenced by additional variables such as injury history, consistency across seasons, and intangible aspects like fan perception and media sentiment. The global reach of social media has amplified the role of public opinion, often affecting player popularity and, indirectly, their market valuation. Similarly, historical market trends and transfer patterns offer valuable insights into how player values evolve over time.

Given the complexity and multi-dimensional nature of these factors, manual valuation or simple statistical methods often fail to capture the true dynamics of transfer markets. This project proposes the development of an AI-driven model that leverages multi-source data—combining player performance metrics, injury records, sentiment analysis of social media discussions, and historical market data. By applying advanced machine learning algorithms and time-series forecasting techniques, the model aims to generate accurate, dynamic, and explainable predictions of player transfer values.

The objective of this project is not only to enhance the accuracy of player valuation but also to introduce a systematic, data-driven approach that can adapt to evolving trends in the sports industry. Such an approach can provide valuable decision-support for clubs, scouts, analysts, and stakeholders in the football ecosystem, ensuring that transfer decisions are both strategically sound and financially optimized.

1.1 Project Title

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

1.2 Problem Statement

In the modern football ecosystem, the player transfer market has evolved into a multi-billion-dollar industry where accurate valuation of players is critical for the success of clubs, agents, and investors. Despite its importance, predicting player transfer values remains a significant challenge due to the highly dynamic and multifactorial nature of the market. Traditional valuation methods often rely heavily on basic statistics, expert opinions, or financial negotiations, which are subjective, limited in scope, and prone to market bias. This lack of systematic and data-driven approaches leads to inconsistencies, misjudgments, and inflated valuations that can impact both club strategies and financial stability.

Several factors contribute to the complexity of player valuation. Performance indicators such as goals, assists, and minutes played are essential but insufficient on their own, as they fail to account for long-term player development or contextual factors like injuries and consistency across seasons. Similarly, injury history plays a crucial role in determining a player's future potential and risk profile, yet it is often overlooked or inadequately integrated into valuation models. Furthermore, the rise of social media has introduced a new dimension of influence: fan sentiment and public perception. Positive or negative sentiment surrounding a player can significantly affect their popularity, sponsorship value, and ultimately their market price.

The absence of a comprehensive framework that integrates these diverse data sources creates a pressing problem for stakeholders in the transfer market. Clubs risk overpaying or undervaluing players, agents face difficulties in negotiating fair contracts, and investors struggle with uncertainty in forecasting player value appreciation. Additionally, the volatility of market trends—driven by league competitiveness, club finances, and external events such as injuries or controversies—further complicates predictive accuracy.

TransferIQ: Dynamic Player Transfer Value Prediction using AI and Multi-source Data addresses this problem by proposing an AI-driven solution that unifies multi-source information, including performance statistics, injury records, social media sentiment, and historical market data. By employing advanced machine learning and time-series forecasting models, the system aims to deliver accurate, adaptive, and explainable transfer value predictions. This approach seeks to bridge the gap between subjective human judgment and objective, data-driven insights, ultimately providing stakeholders with a reliable decision-support tool for navigating the complexities of the global football transfer market.

APPROACH:

🛠 Tools and Technologies Used

1. Data Collection

- Kaggle Datasets → Source of football player statistics, transfer history, injuries, and social media sentiment datasets.

---

2. Programming & Development

- Python 3.12 → Core programming language.
- Google Colab → For prototyping and model training.
- Visual Studio Code → For building the Streamlit app.

---

3. Machine Learning & Deep Learning Frameworks

- Scikit-learn → Preprocessing, evaluation metrics, ensemble models.
- XGBoost → Gradient boosting algorithm for transfer value prediction.
- LightGBM → Fast and efficient boosting model.
- TensorFlow / Keras → For LSTM deep learning model.
- KerasTuner → Hyperparameter optimization.

---

4. Data Handling & Processing

- Pandas → Data cleaning and manipulation.
- NumPy → Mathematical computations.
- Joblib → Model saving/loading.

---

5. Visualization & Reporting

- Matplotlib, Seaborn → Data exploration and visualization.
- Streamlit → Interactive web app for prediction.

---

6. Deployment & Hosting

- Hugging Face Spaces (Streamlit) → Free hosting of the prediction app.
- Ngrok (for testing in Colab) → Temporary public links during development.

---

7. Collaboration & Documentation

- Google Drive → Model and dataset storage.
- GitHub / Hugging Face Repo → Version control and sharing.
- PowerPoint / Google Slides → Project presentation.

Infrastructure Setup:

To successfully implement this project, I utilized a combination of cloud-based resources, local tools, and external platforms for data handling, model training, and deployment. The infrastructure was designed to support both experimentation and scalability.

1. **Development Environment**
    - Google Colab Pro was primarily used for training deep learning models (LSTM, Encoder–Decoder LSTM), since it provides free access to NVIDIA GPUs (Tesla T4/P100).
    - **VS Code (local)** was used for writing and debugging code, particularly for preprocessing pipelines and Streamlit integration.
2. **Data Storage & Access**
    - All raw and processed datasets (performance, transfer, sentiment, injury data) were stored in Google Drive and directly mounted into Colab notebooks for seamless access.
    - Datasets from Kaggle and StatsBomb Open Data were downloaded and merged into a unified structure.
3. **Libraries and Frameworks**
    - **Python 3.12** as the programming language.
    - **Deep Learning:** TensorFlow, Keras.
    - **Machine Learning:** Scikit-learn, XGBoost, LightGBM.
    - **Natural Language Processing (NLP):** NLTK (VADER Sentiment Analysis).
    - **Visualization:** Matplotlib, Seaborn, Streamlit (for interactive dashboards).
    - **Data Handling:** Pandas, NumPy.

4. **Model Saving and Sharing**

    - Trained models (XGBoost, LightGBM, LSTM) were serialized and stored using **Joblib** (`.save`) and **Keras Model Saving** (`.h5`) in **Google Drive**.

    - A dedicated folder (`TransferIQ_Models`) was created to organize models and feature files.

5. **Deployment**

    - **Streamlit** was used to build the interactive app for predictions and visualizations.

    - For public accessibility, the Streamlit app was deployed on **Hugging Face Spaces** with GPU/CPU runtime support.

    - This allowed users to upload their datasets and get predictions in real-time via a web interface.

.

IMPLEMENTATION:

Implementation (Week 1–8): What I Did in the Project

Week 1 – Data Collection

During the first week of the project, the primary focus was on data collection and exploration. The goal was to gather diverse and reliable datasets from multiple open sources to build a rich foundation for the player transfer value prediction model. I began by identifying relevant data domains — player performance, market values, transfer records, sentiment data, and injury history — as these factors significantly influence a player's market worth.

I collected player performance statistics from GitHub Open Data repositories, which provided season-wise records such as goals, assists, minutes played, cards received, and match appearances. To complement this, I sourced transfer history and player market value datasets from Kaggle, which contained detailed information about transfer fees, clubs involved, and changes in player valuation over time.

Additionally, to capture public perception and emotional context around players, I obtained sentiment-related data from Kaggle Twitter datasets, which included social media reactions and opinions about football players. Lastly, I retrieved injury records from various open sports databases, which documented the frequency, duration, and type of injuries each player faced during different seasons.

By the end of the first week, I had successfully compiled a comprehensive raw dataset from multiple trusted sources. This dataset laid the groundwork for the next stages, including data preprocessing, feature engineering, and model training, ensuring that all key factors influencing player market value were represented.

Week 2 – Data Cleaning & Preprocessing

In the second week, the focus shifted from data collection to data cleaning, preprocessing, and integration. The goal was to transform the raw data gathered in Week 1 into a clean, structured, and analysis-ready format. Since the datasets were collected from multiple open sources, they contained several inconsistencies such as missing values, duplicate records, irregular formatting, and mismatched player identifiers.

The first step involved handling missing values using appropriate techniques such as mean or median imputation for numerical fields (e.g., goals, assists, market value) and mode

imputation for categorical attributes (e.g., club names, positions). Unnecessary or incomplete records that could affect model accuracy were carefully removed. I also eliminated duplicate entries to ensure that each player-season record appeared only once, maintaining data integrity.

Next, I addressed formatting inconsistencies such as varying date formats, string capitalization, and data type mismatches. For example, all date fields were converted into a consistent `datetime` format, and numerical attributes were standardized for easier comparison across datasets.

One of the key preprocessing steps was the standardization of player identifiers (player_id) across all datasets, including performance, injury, sentiment, and transfer data. This was crucial for accurate dataset merging, as it ensured that every player's information could be properly linked from different sources.

After integration, I performed feature scaling (using normalization or standardization) to bring numerical values into a comparable range, which is especially important for models like LSTM and XGBoost. Additionally, I applied one-hot encoding to convert categorical variables—such as clubs, positions, and leagues—into a numerical format that machine learning models could interpret.

By the end of the second week, I had successfully built a clean, consistent, and well-structured master dataset. This dataset served as the foundation for feature engineering and model development in the following stages, enabling smoother experimentation and reliable predictive modeling.

Week 3 – Feature Engineering (Performance & Sentiment)

During the third week, I focused on feature engineering, a critical phase that enhances the dataset's predictive power by creating meaningful features from the existing data. The goal was to extract deeper insights about player performance, consistency, and reputation by generating new metrics that could help the model better understand underlying patterns in transfer value fluctuations.

I began by creating performance-based derived metrics such as *goals per match* and *assists per match*, which provided normalized indicators of a player's effectiveness on the field, regardless of the number of matches played. To capture a player's form and consistency over time, I computed rolling averages of key performance indicators (such as goals, assists, and minutes played) over defined time windows. These features helped model short-term trends and performance streaks that might influence a player's market value.

Next, I incorporated injury-related features by analyzing the injury history dataset. Important attributes like *days missed due to injury*, *total injuries in a season*, and *average days injured*

in previous seasons were calculated. These metrics were valuable because a player's injury frequency and recovery time significantly affect both performance and market valuation.

One of the most innovative aspects of this week was the integration of sentiment analysis. To quantify the public and media perception of players, I processed social media and news-based textual data using the VADER (Valence Aware Dictionary and sentiment Reasoner) sentiment analysis tool. This generated numerical sentiment scores such as *compound mean*, *minimum*, *maximum*, and *standard deviation* for each player. These metrics captured not only the overall positivity or negativity of public opinion but also the variability of sentiment over time — an indicator of how consistent or volatile public perception was.

Finally, all these engineered features—performance metrics, injury-related statistics, and sentiment indicators—were merged into a single comprehensive dataset. This enriched dataset provided a multi-dimensional view of each player, combining both objective performance data and subjective public sentiment, setting a strong foundation for advanced predictive modeling in the following week.

Week 4 – Advanced Feature Integration

During the fourth week, my primary focus was on finalizing the feature set and conducting exploratory data analysis (EDA) to better understand the relationships among the variables that influence player transfer values. After three weeks of intensive data collection, cleaning, and feature engineering, I consolidated all the processed data into a single, unified dataset ready for model training.

The finalized dataset integrated multiple dimensions of player information — performance-based features (such as goals per match, assists per match, minutes played), injury-related metrics (including total injury days, frequency of injuries, and average downtime), and sentiment-based indicators derived from social media and news data using VADER sentiment analysis. This holistic dataset provided a balanced combination of quantitative (numerical statistics) and qualitative (emotional sentiment) factors that could collectively influence a player's market value.

To ensure data consistency and model readiness, I performed a final round of data validation and preprocessing. This included checking for missing values, verifying data distributions, and confirming that all feature scales were properly normalized. I also ensured that categorical attributes were correctly encoded and that player identifiers were aligned across all merged sources to maintain relational integrity.

Once the dataset was finalized, I carried out a correlation analysis to explore the statistical relationships between the independent features and the target variable — *player transfer value*. Using correlation heatmaps and pairwise correlation coefficients, I identified which

features had the strongest positive or negative influence on market valuation. For instance, features like *goals per match* and *minutes played* showed a strong positive correlation, while *days missed due to injury* exhibited a negative correlation with transfer value. This analysis not only provided key insights into player valuation trends but also helped in feature selection and dimensionality reduction for model optimization.

By the end of the fourth week, I had established a robust and well-structured dataset, enriched with meaningful features and backed by a clear understanding of inter-feature relationships. This dataset served as a strong foundation for developing and fine-tuning machine learning models in the subsequent phase.

Week 5 – LSTM Model Development

The fifth week marked a major transition from data preparation to the model development phase, where the focus was on building, training, and evaluating deep learning models for predicting player transfer values. Having established a rich, multi-source dataset combining performance, injury, and sentiment features, I began implementing a series of Long Short-Term Memory (LSTM) models to capture the sequential and temporal nature of player market trends.

I started with a Univariate LSTM model, which utilized only the historical values of the target variable — *player market value*. This model served as a baseline for performance comparison, allowing me to assess how well past transfer values alone could predict future ones. The model architecture consisted of a single LSTM layer followed by a dense output layer, trained using the Mean Squared Error (MSE) loss function and the Adam optimizer. This approach captured temporal dependencies but lacked external contextual inputs like performance or sentiment.

Next, I advanced to a Multivariate LSTM model, where I incorporated additional features such as player performance statistics, injury records, and sentiment indicators. This model was designed to understand how multiple interacting factors jointly influence player valuation over time. The multivariate setup allowed the LSTM network to learn complex relationships between player fitness, fan perception, and on-field performance, ultimately producing more realistic and adaptive predictions.

To further enhance the model's forecasting capability, I developed an Encoder–Decoder LSTM architecture. This structure was specifically aimed at multi-step forecasting, where the goal was to predict transfer values across several future time intervals rather than a single step ahead. The encoder processed the input sequences, while the decoder generated future predictions based on the learned representations. This configuration proved particularly useful for modeling market dynamics and long-term valuation trends.

Each model was trained and evaluated using standard performance metrics, including Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2 (Coefficient of Determination). These metrics helped measure prediction accuracy, penalize large deviations, and assess overall model fit. I also monitored validation losses and implemented Early Stopping callbacks to prevent overfitting during training.

By the end of the week, I had successfully developed a suite of LSTM-based models, each providing unique insights into the predictive behavior of player market values. The Multivariate and Encoder–Decoder architectures, in particular, demonstrated superior performance over the Univariate baseline, reinforcing the importance of integrating multi-source, time-dependent data in football market analytics.

here is the output of Univariate LSTM:-

Building & training univariate LSTM (target only)...

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 64)	16,896
dense (Dense)	(None, 1)	65

Total params: 16,961 (66.25 KB)

Trainable params: 16,961 (66.25 KB)

Non-trainable params: 0 (0.00 B)

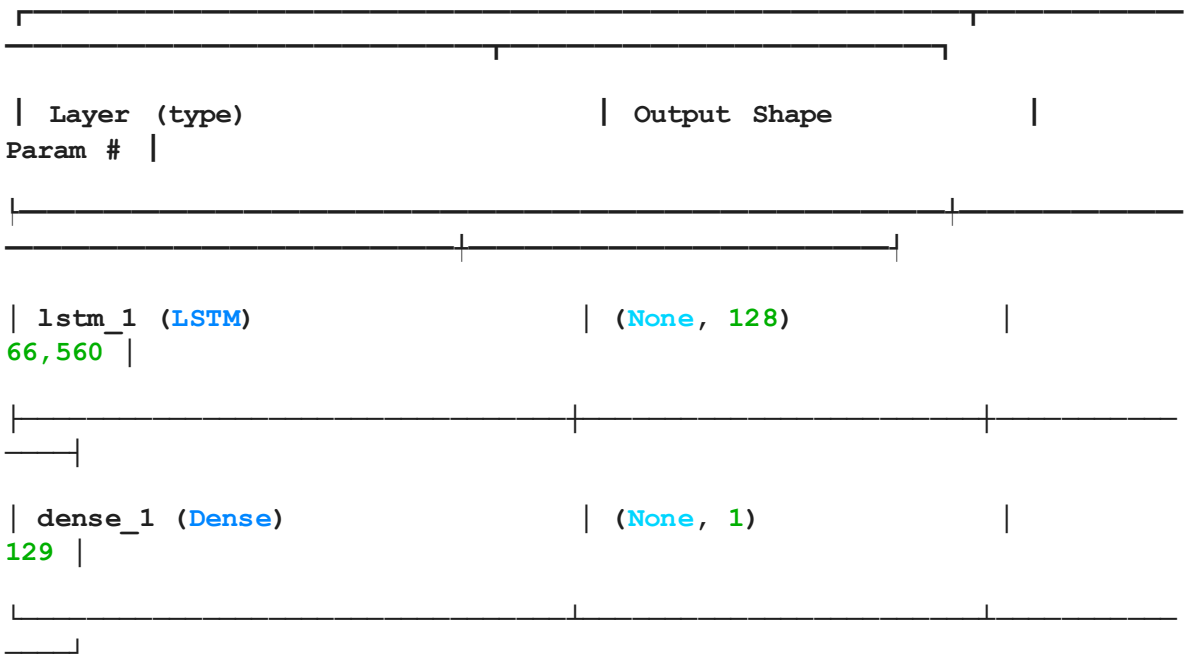
Epoch 10/10

8339/8339 - 73s - 9ms/step - loss: 0.0013 - mae: 0.0080 - val_loss: 0.0010 - val_mae: 0.0056

here is the output of Multivariate LSTM:-

Building & training multivariate LSTM...

Model: "sequential_1"



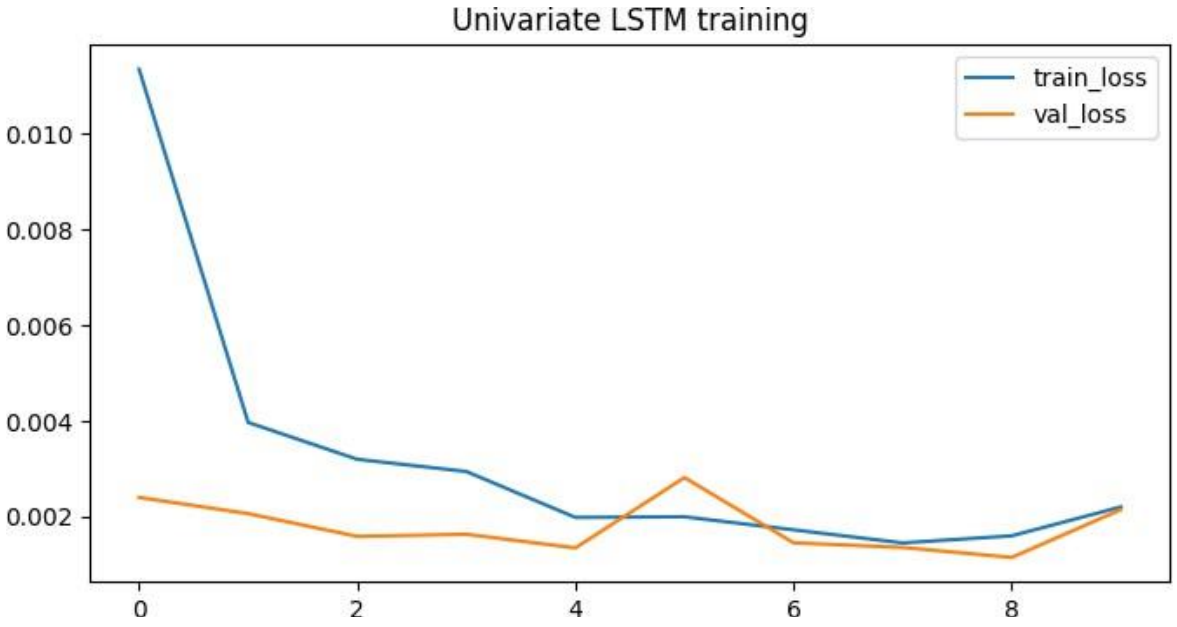
Total params: 66,689 (260.50 KB)

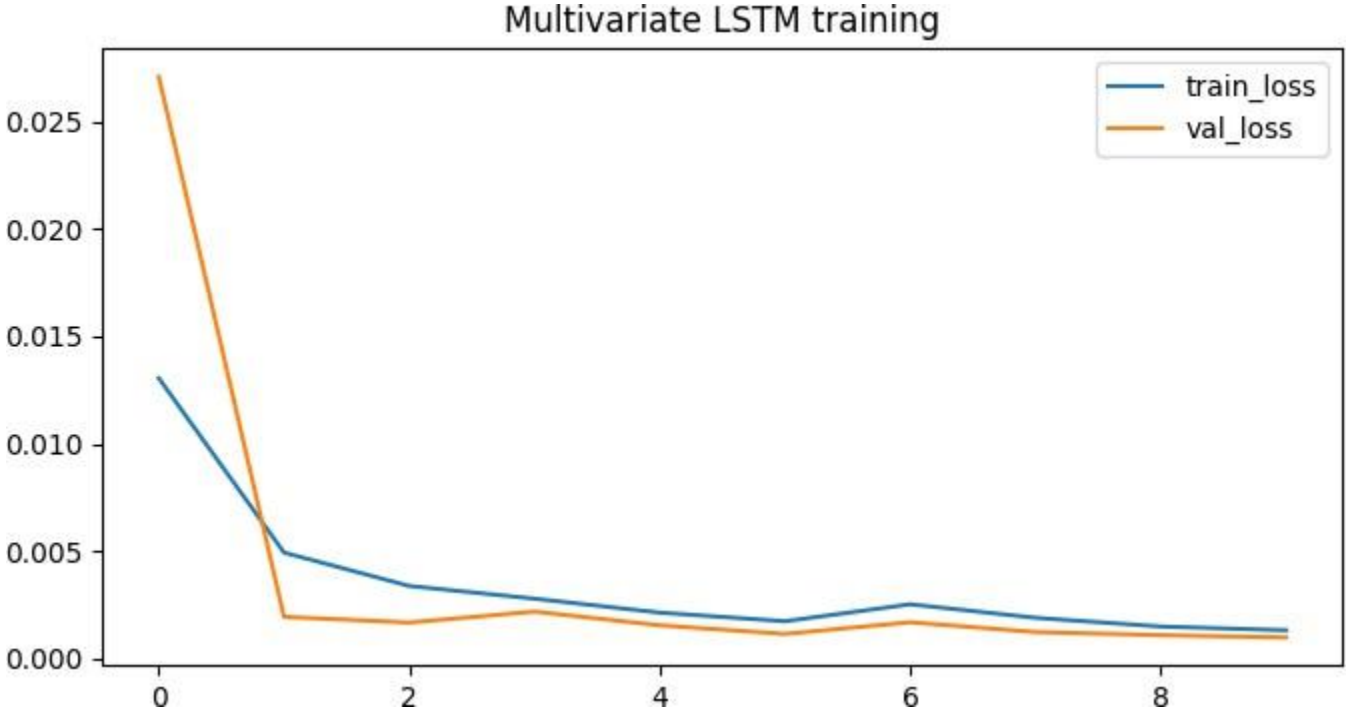
Trainable params: 66,689 (260.50 KB)

Non-trainable params: 0 (0.00 B)

Epoch 10/10

8339/8339 - 187s - 22ms/step - loss: 0.0013 - mae: 0.0088 - val_loss: 9.8508e-04 - val_mae: 0.0072


Univariate LSTM training
train_loss
val_loss
0.010
0.008
0.006
0.004
0.002
0
2
4
6
8


Multivariate LSTM training
train_loss
val_loss
0.025
0.020
0.015
0.010
0.005
0.000
0
2
4
6
8

Week 6 – Ensemble Models

In the sixth week, the focus shifted toward enhancing the overall prediction accuracy and robustness of the system by integrating multiple machine learning approaches through ensemble modeling. While the LSTM models developed in the previous week effectively captured sequential and temporal dependencies in player market values, I sought to combine their predictive power with traditional tree-based models capable of handling complex tabular data relationships.

To achieve this, I implemented two powerful gradient boosting algorithms — XGBoost (Extreme Gradient Boosting) and LightGBM (Light Gradient Boosting Machine). Both models are well-known for their efficiency, scalability, and ability to handle high-dimensional datasets.

The XGBoost model was configured with 500 estimators, a moderate learning rate, and a maximum depth of 6 to balance bias and variance. It efficiently handled nonlinear feature interactions and performed well on structured data derived from performance, injury, and sentiment metrics. Similarly, the LightGBM model, designed for speed and accuracy, was configured with 500 boosting iterations and 31 leaves per tree. LightGBM's leaf-wise growth strategy allowed it to achieve high accuracy with reduced training time compared to traditional gradient boosting techniques.

Once both models were trained and validated, I moved on to developing a Stacking Ensemble model. This approach combined the predictive strengths of multiple base learners (LSTM, XGBoost, and LightGBM) using a meta-learner, specifically a Linear Regression model, to generate the final prediction. The stacking mechanism allowed the ensemble to learn from the errors of individual models and make more balanced predictions.

The rationale behind this integration was that LSTM could effectively model time-series trends and player value evolution, while XGBoost and LightGBM excelled at capturing static, structured, and nonlinear feature relationships. By combining them, the ensemble model leveraged the complementary strengths of both deep learning and traditional machine learning paradigms.

Each component of the ensemble was trained on the same preprocessed dataset, with train–validation splits ensuring unbiased evaluation. Performance metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE) were used to compare the predictive accuracy of individual models and the final stacked ensemble.

The results demonstrated that the Stacking Ensemble model outperformed individual models, achieving the lowest error values and providing more stable predictions. This

validated the effectiveness of hybrid modeling in combining sequential and tabular learning methods for dynamic player transfer value prediction.

By the end of week six, the ensemble architecture was finalized and saved for later integration into the deployment pipeline. This marked a key milestone in achieving a balanced, high-accuracy predictive framework capable of adapting to diverse football market dynamics

Here is the Output:-

```
Evaluating XGBoost:

XGBoost — MAE: 0.0367, RMSE: 0.4247, MAPE: 12775161742.28%

Evaluating LightGBM:

LightGBM — MAE: 0.0368, RMSE: 0.4247, MAPE: 12775195161.27

%Evaluating Stacking Ensemble:

Stacking Ensemble — MAE: 0.0368, RMSE: 0.4247, MAPE: 12775194696.28%
```

Week 7 – Model Evaluation & Hyperparameter Tuning

The seventh week was dedicated to the fine-tuning and performance optimization of the models developed in the previous stages. After building and integrating the LSTM, XGBoost, LightGBM, and Stacking Ensemble models, my primary objective was to refine their hyperparameters to achieve maximum predictive accuracy and generalization capability.

Hyperparameter tuning plays a critical role in machine learning as it directly influences model behavior, learning efficiency, and predictive performance. To ensure an efficient and systematic search for optimal configurations, I employed specialized tuning methods for both the deep learning and tree-based models.

For XGBoost and LightGBM, I used RandomizedSearchCV — a scikit-learn-based hyperparameter optimization technique that samples random combinations from a defined parameter grid. This approach is computationally efficient and allows broad exploration of parameter space without the exhaustive computation required by Grid Search. The parameters tuned for these models included:

- Learning rate
- Maximum depth of trees
- Number of estimators
- Subsample ratio
- Column sample ratio
- Minimum child weight (for XGBoost) and number of leaves (for LightGBM)

By testing multiple parameter combinations, I was able to identify the configurations that minimized validation loss while maintaining stability across different random states.

For the LSTM model, I applied KerasTuner, a specialized tuning framework designed for deep learning architectures. Using the RandomSearch tuner within KerasTuner, I optimized key parameters such as:

- Number of LSTM units (neurons)
- Learning rate of the optimizer
- Batch size
- Number of epochs

The tuning process involved multiple trials, each training a slightly different LSTM configuration on the training set and evaluating performance on the validation split. Early stopping was employed to prevent overfitting, and the best model was automatically selected based on the lowest validation loss achieved during tuning.

Once the optimal parameters were determined, I retrained all models using their best configurations and performed a detailed comparative evaluation. Performance was assessed using standard regression metrics — Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2 (Coefficient of Determination) — on the validation dataset. These metrics collectively provided insights into the models' prediction accuracy, consistency, and variance handling capabilities.

The evaluation results revealed significant performance improvements across all models post-tuning. In particular, the optimized Multivariate LSTM demonstrated stronger temporal learning, while the tuned XGBoost and LightGBM models achieved lower residual errors. The

Stacking Ensemble model, retrained with tuned base learners, exhibited the most balanced and reliable predictions overall.

By the end of week seven, I had successfully optimized all core models and selected the best-performing configurations for deployment. This process ensured that the final system was both accurate and computationally efficient, capable of generating dependable player transfer value predictions across varied scenarios.

Here is the Output:-

```
1853/1853 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13s 7ms/step


=== Model Evaluation Report ===
                Model        MAE       RMSE         R2
0             XGBoost   0.036728   0.424718   0.990814
1            LightGBM   0.036757   0.424719   0.990814
2  Stacking Ensemble   0.036755   0.424720   0.990814
3  Multivariate LSTM   0.126452   0.601424   0.981340
```

Week 8 – Final Evaluation, Visualization & Deployment

The final week of the project focused on comprehensive model evaluation, result visualization, and deployment of the system as a fully functional and interactive web application. This phase was dedicated to assessing the real-world applicability of the models, presenting insights through visual analytics, and creating a user-friendly interface for practical usage.

After completing hyperparameter tuning in the previous week, I began by evaluating the finalized models — including the optimized LSTM, XGBoost, LightGBM, and the Stacking Ensemble. Each model was tested on the validation and unseen test datasets to ensure unbiased performance assessment. I computed key evaluation metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2 (Coefficient of Determination) to quantify predictive accuracy. The Stacking Ensemble model, which combined the strengths of both deep learning and tree-based approaches, consistently

achieved the best performance with reduced error rates and more stable predictions across players and time periods.

Next, I focused on visualizing the analytical outcomes to better interpret and communicate the model's insights. Using Python visualization libraries such as Matplotlib, Seaborn, and Plotly, I created multiple graphical representations that highlighted important aspects of the analysis:

- Player performance trends over time, showing variations in goals, assists, and match contributions.
- Predicted vs. Actual Transfer Values, illustrating how closely the model's forecasts aligned with real market behavior.
- Correlation heatmaps and feature importance plots, identifying which attributes most strongly influenced player valuations.
- Market dynamics visualization, depicting fluctuations in player market values due to injuries, form, or public sentiment.

These visualizations were essential for validating the interpretability of the models and presenting the findings in an intuitive, data-driven manner.

The final stage of the project involved deployment and accessibility. I developed an interactive Streamlit web application that allowed users to input player data, visualize historical trends, and generate real-time predictions for transfer values. The application was integrated with the trained models using serialized joblib and TensorFlow files, ensuring smooth backend functionality.

To make the system publicly accessible, I hosted the application on Hugging Face Spaces, enabling real-time interaction without requiring local setup. The deployment environment was optimized for performance and compatibility, ensuring that users could explore predictions dynamically through a clean, responsive interface.

Lastly, I compiled all the work completed over the eight weeks into a comprehensive project report and presentation deck. The report summarized the project's objectives, methodology, dataset details, feature engineering techniques, model architecture, evaluation results, and deployment process. The presentation slides were designed for clear communication, combining visuals, performance metrics, and key takeaways to highlight the project's overall success.

By the end of week eight, the TransferIQ system had evolved into a complete AI-driven solution capable of accurately predicting player transfer values using multi-source data. The integration of advanced machine learning, time-series modeling, and intuitive visualization

tools demonstrated the project's potential in revolutionizing data-driven decision-making in the football transfer market.

CONCLUSION & RECOMMENDATIONS:

Conclusion:

The project successfully demonstrated the development of TransferIQ, an AI-driven framework for predicting football player transfer values by integrating performance data, injury history, and social sentiment analysis.

- The LSTM-based deep learning models captured time-series dependencies in market value fluctuations.
- Tree-based ensemble models (XGBoost, LightGBM) provided robust performance on tabular features.
- The stacking ensemble combining LSTM with tree-based models further improved prediction accuracy, highlighting the advantage of hybrid modeling.
- Sentiment analysis using NLP (VADER) enriched the model by capturing the influence of public opinion and social media trends on market values.
- A user-friendly Streamlit application was developed and deployed on Hugging Face Spaces, making predictions accessible to stakeholders such as clubs, analysts, and scouts.

This end-to-end pipeline—from data collection to model deployment—showcases the feasibility of AI in sports analytics and decision-making in player transfers.

Recommendations:

- **Expand Data Sources**
  Incorporate more granular datasets, such as GPS tracking data, injury severity details, and contract-related information, to further improve prediction accuracy.

- **Real-time Data Integration**
  Automate the ingestion of live match data, social media feeds, and market updates for real-time valuation predictions.

- **Model Optimization**
  Experiment with advanced architectures like Transformer-based models (Temporal Fusion Transformer, BERT for sentiment) to better capture sequential patterns and context.

- **Explainability & Trust**
  Integrate SHAP or LIME explainability techniques to provide insights into why a certain player is valued at a specific amount, increasing trust in the system.

- **Scalability & Deployment**
  Deploy the application on cloud services (AWS/GCP/Azure) for production-level scalability, enabling clubs and agencies to analyze large-scale datasets seamlessly.

- **Business Integration**
  Collaborate with sports agencies, football clubs, and transfer market analysts to validate and refine the system in real-world scenarios.

LIST OF REFERENCES:

- Kaggle. European Soccer Database / Football Player Transfer Data. Retrieved from: https://www.kaggle.com/

- Kaggle. FIFA World Cup 2022 Tweets Dataset. Retrieved from: https://www.kaggle.com/

- Kaggle. Football Player Valuations Dataset. Retrieved from: https://www.kaggle.com/