PREDICTING FOOTBALL PLAYER MARKET VALUE

Introduction & Problem

- Predicting football player transfer values is complex and dynamic
- Influenced by many factors — performance, injuries, market trends, public sentiment
- Traditional methods are subjective and inconsistent
- Need for a data-driven, AI-based approach to improve reliability

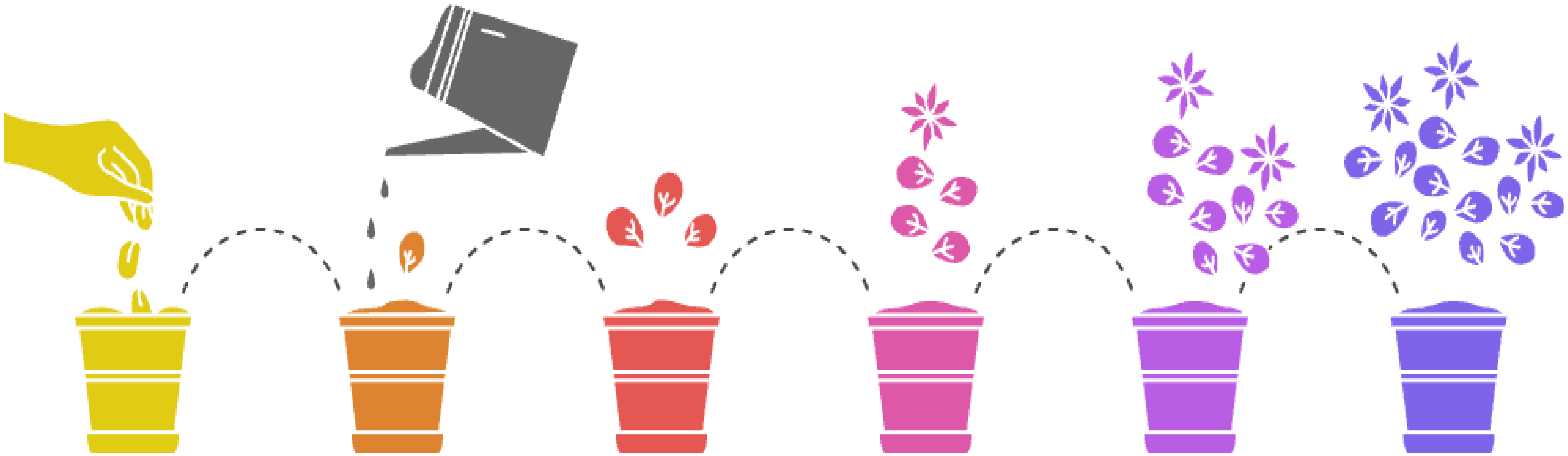
Project Objective

- Build an **AI-driven system** that integrates multiple data sources
- Apply **machine learning** and **time-series forecasting (LSTM)**
- Generate **accurate, adaptive, and real-time** transfer value predictions

Demo of this Project

https://drive.google.com/file/d/1bVqezHOFYbP53khLyly3_2tdeACpusU0/view?usp=drive_link

Step-by-Step Approach



Raw Data
Unprocessed player statistics

Data Cleaning
Handling missing values and duplicates

Feature Engineering
Creating new performance metrics

Model Building
Developing LSTM and ensemble models

Hyperparameter Tuning
Optimizing model parameters

Deployed Model
Interactive transfer value predictions

Data Collection & Preprocessing

- **Sources Used:**
- Kaggle— Player performance stats
- Kaggle— Transfer history & market values
- Twitter (X) — Sentiment data via API
- Open Sports Databases — Injury data

Preprocessing Steps:

- Removed duplicates & handled missing values
- Standardized player IDs
- Feature scaling & one-hot encoding

Feature Engineering:

- **Performance Metrics:** Goals per match, assists per match, minutes per game
- **Injury Metrics:** Days missed due to injury, number of injuries, significant injury flags
- **Sentiment Metrics:** Compound score & polarity (mean, min, max, std) from social media
- **Cumulative Stats:** Total goals, total assists, games played per season
- **Player Attributes:** Age, BMI, market value trends
- **Transfer History:** Total past transfers, previous clubs

LSTM Model Development

Univariate LSTM (Target-only Model)
- Uses **only player's past transfer values** as input
- Built with **1 LSTM layer (64 units)** + **Dense output layer**
- Predicts next-step transfer value
- Early stopping & model checkpoint used for optimal training
- Trained for 20 **epochs** with mean squared error (MSE) loss

Multivariate LSTM (All Features Model)
- Includes **performance, injury, and sentiment features**
- LSTM layer with **128 units** captures multivariate time dependencies
- Output layer: predicts single-step or multi-step values
- Same call back (early stopping, checkpoints) used
- Trained for 20 **epochs** with MSE loss and MAE metric

Result

Multivariate LSTM — Val MAE: 0.2835, RMSE: 0.9856, MAPE: 899474963.39%

MAE (Mean Absolute Error) :- 0.2835

RMSE (Root Mean Squared Error) :- 0.9856

Ensemble Model Integration

Tree-Based Models

- **XG Boost:** 500 estimators, learning rate 0.05, max depth 6
- **Light GBM:** 500 estimators, learning rate 0.05, 31 leaves
- Trained on **flattened feature sequences** for tree-based regression

Stacking Ensemble

- Combines **XG Boost + Light GBM** predictions
- Final estimator: **Linear Regression**
- Pass-through enabled → uses original features along with predictions
- Trained to improve overall prediction accuracy

Validation Performance of All Models

Model	MAE	RMSE	MAPE
Univariate LSTM	0.2835	0.9856	~8.99×10^{8} %
Multivariate LSTM	0.2835	0.9856	~8.99×10^{8} %
XG Boost	0.0367	0.4247	~1.28×10^{10} %
Light GBM	0.0396	0.5231	~1.28×10^{10} %
Stacking Ensemble	0.0368	0.5247	~1.28×10^{10} %

Hyperparameter Tuning

Objective:

- Optimize **LSTM architecture** for better transfer value predictions
- **Approach (Keras Tuner Random Search):**
- **Layer Units:** Tested 32 → 64 → 96 → 128 LSTM units
- **Learning Rate:** Tried 0.01, 0.001, 0.0001
- **Validation:** 20% split of training data
- **Epochs:** 5 per trial, 4 trials total

Outcome:

- Best model selected automatically by **lowest validation loss**
- Tuned LSTM ready for **final training and evaluation**

Technology Used

Programming & Libraries:

- **Python** – main programming language
- **Pandas, NumPy** – data processing & manipulation
- **Scikit-learn** – preprocessing, tree-based models, evaluation
- **TensorFlow & Keras** – LSTM modeling
- **Keras Tuner** – hyperparameter tuning
- **XG Boost & Light GBM** – ensemble models
- **VADER / NLTK** – sentiment analysis

Visualization & Deployment:

- **Matplotlib, Seaborn, Plotly** – charts & trend visualization
- **Streamlit** – interactive web app

Thank you