1.Introduction

SkillFusionX, the ultimate football management solution revolutionizing the game through the fusion of machine learning and data analytics. With a keen focus on player attributes such as defensive abilities, passing accuracy, and dribbling skills, SkillFusionX employs advanced classification algorithms to accurately predict player positions and group players with similar skill sets, facilitating seamless team composition and tactical planning. Its user-friendly interface provides coaches with easy navigation through real-time player data, ensuring up-to-date insights for dynamic team management. Experience the power of SkillFusionX to maximize team synergy, elevate performance, and revolutionize your approach to player recruitment and strategic planning in football management.

1.1PROJECT DESCRIPTION

1.1.1 PROBLEM DEFINITION

In traditional football management, coaches and team managers face significant challenges in accurately determining the most suitable positions for players based on their individual skill sets. This process is often subjective, relying heavily on the experience and intuition of the coaching staff, leading to potential misplacements and suboptimal team compositions. Additionally, the lack of systematic analysis and data-driven insights makes it difficult to identify patterns and trends in player performance, hindering effective strategic planning and player recruitment efforts. Without a comprehensive solution to streamline player position prediction and team composition, sports clubs struggle to maximize team synergy and achieve peak performance on the field..

1.1.2 PROPOSED SOLUTION

To address the challenges faced in traditional football management, we propose the implementation of SkillFusionX, a cutting-edge machine learning application. SkillFusionX utilizes advanced algorithms to analyze player attributes such as defensive abilities, passing accuracy, and dribbling skills. By leveraging this data, the application accurately predicts the most suitable positions for players and groups them based on similar skill sets. This provides coaches and team managers with invaluable insights for optimizing team composition and strategic planning. With a user-friendly interface, SkillFusionX simplifies data interpretation and empowers coaches to make informed decisions. By revolutionizing player management and team strategy, SkillFusionX enhances team synergy and elevates overall performance on the field.

1.1.3 PURPOSE

The purpose of SkillFusionX is to revolutionize football management by leveraging machine learning and data analytics. It aims to provide coaches and team managers with a powerful tool to accurately predict player positions and group players based on their skill sets. By doing so, SkillFusionX facilitates informed decision-making, optimizes team composition, and enhances overall team performance on the field. The application seeks to streamline player management processes, improve strategic planning, and elevate the standard of football management by harnessing the capabilities of modern technology. Ultimately, SkillFusionX aims to empower coaches, maximize team synergy, and contribute to the success of football teams worldwide

1.1.4 SCOPE

This initiative aims to develop a user-friendly interface for coaches and team managers to navigate player data efficiently. It will use machine learning algorithms to predict player positions and group players by skill sets. The application will support real-time integration of new player data for ongoing relevance and provide insights for dynamic team management. Additionally, it will aid in player recruitment processes by offering data-driven recommendations. Overall, SkillFusionX aims to optimize football management practices and enhance team performance through data analytics and machine learning.

2.LITERATURE SURVEY

2.1 DOMAIN SURVEY

Football matches feature two teams of 11 players competing to score goals by kicking the ball into the opposing team's net. Each match consists of two halves lasting 45 minutes each, separated by a 15-minute halftime break. Players, except goalkeepers, cannot use their hands, and fouls lead to free kicks or penalties. The offside rule regulates forward positioning, and extra time or penalties may occur in knockout stages. Crucial player records such as foot preference, skill moves, stamina, GKDiving (goalkeeper diving ability), and ShotPower (shot strength) significantly influence team strategy and performance optimization in prestigious tournaments like the FIFA World Cup. Understanding these records is fundamental for coaches and managers to devise effective tactics and achieve success on the pitch.

2.2 RELATED WORK

Victor Chazan-Pantzalis,Christos Tjortjis[1] explains a research study on the use of the Random Forest algorithm, a computer-based technique, to anticipate loan defaults. The main goal is to comprehend the possibility that borrowers using online lending services won't pay back their loans. The study presents a model that predicts the probability of a borrower defaulting using the Random Forest algorithm.The results show that when it comes to anticipating situations in which borrowers could be unable to repay their loans, the Random Forest algorithm outperforms the other techniques. The long-term viability and well-being of online lending platforms depend on the conclusions drawn from this study.

Babu, S. B., Vivek, V., Kumar, D. M. T., Prathyusha, K., & Teja, G. P.[2] this novel approach using Classification Algorithms and Random Forest to predict football players' suitable positions based on their skills, achieving approximately 90% accuracy.The research compares different computer techniques and finds that Random Forest is the best at predicting positions for 11 different playing roles on the soccer field.

Cheng, G., Zhang, Z., Kyebambe, M. N., & Kimbugwe, N[3]:The study focuses on predicting NBA game outcomes using the NBAME model based on the maximum entropy principle.The NBAME model demonstrates superior performance compared to other machine learning algorithms.

Verstraete, K., Coussement, B., & Vannieuwenhoven[4]: It explains about the tensor-based methods to analyze soccer players' skill ratings over time.Utilizing a multidimensional approach, the study explores the temporal dynamics of skill ratings for enhanced insights into player performance.

Hucaljuk, J., & Rakipović, A.[5]: It explains about the software solution for predicting Champions League football match outcomes using machine learning, achieving prediction accuracy of up to 68%.Feature selection, including f team rankings, and player injuries, was combined with classifiers like Bayesian networks and artificial neural networks for superior predictions compared to a reference method.

2.3 EXSISTING SYSTEMS

- **PlayerMaker:** Wearable device and software for analyzing player performance data to optimize training and development.
- **SoccerLAB:** Comprehensive platform for football clubs to manage player data, analyze performance, and enhance tactical planning through video analysis.
- **Catapult Sports:** Wearable technology tracking athlete performance metrics with software providing detailed analytics for coaches to optimize performance and reduce injury risks.
- **Hudl:** Video analysis software for sports teams, including football, to review game footage, analyze player performance, and create actionable insights for coaching staff.

2.4 TECHNOLOGY SURVEY

Extended Technology Survey Summary: Python-Based Tools for AI and Data Science

1. Flask:

Flask is a lightweight and flexible web framework that simplifies web developmentand API creation in Python. Its modular design and ease of use make it suitable forintegrating AI services into web applications.

2. Machine Learning Libraries:

Scikit-learn: Scikit-Learn, also known as sklearn is a python library to implement machine learning models and statistical modelling. Through scikit-learn, we can implement various machine learning models for regression, classification, clustering, and statistical tools for analyzing these models.

Pandas: Pandas is a Python library used for working with data sets. It has functionsfor analyzing, cleaning, exploring, and manipulating data**.**

Matplotlib & Seaborn: Matplotlib and Seaborn are two popular Python libraries for data visualization. Matplotlib is a lower-level library that provides more controlover the appearance of plots, while Seaborn is a higher-level library that provides amore user-friendly interface and a set of default styles that are designed to make plots more visually appealing**.**

3. HTML & CSS

HTML and CSS are scripting languages used to create a web page and web applications. HTML provides web page structure, whereas CSS is mainly used to control web page styling.

3.HARDWARE AND SOFTWARE REQUIREMENTS

3.1HARDWARE REQUIRMENTS

Processor(CPU)	**Dual-core processor(Intel Core i3 and higher)**
Memory	**4GB RAM + 8GB**
Networking	**Stable internet connection with adequate bandwidth**

3.2SOFTWARE REQUIREMENTS

- **Frontend:** HTML5, CSS3 and JavaScript ES6
- **Backend:** Python 3.5
- **Deployment:** Flask 3.0.0 or AWS
- **IDE:** VS Code17.8.3 and Jupyter-Notebook6.2

Packages :

- pandas: Data manipulation and analysis.
- numpy: Numerical operations on arrays.
- matplotlib: Basic data visualization.
- seaborn: Statistical data visualization.
- scikit-learn,xgboost,adaboost: Includes various machine learning algorithms for classification and clustering
- Flask or FastAPI: Web framework for deploying machine learning models.
- Docker: Containerization for easy deployment.

4.SOFTWARE REQUIREMENTS SPECIFICATION

4.1 USERS

General Users: Interact with the platform, provide the input and get the recommended output by application.

4.2 FUNCTIONAL REQUIREMENTS

- **Data Collection:** Gather FIFA player data from url : https://fbref.com/en/comps/1/history/World-Cup-Seasons
- **Data Cleaning and Preprocessing:** Performed data cleaning by handling missing values and outliers, normalized numerical features, and encoded categorical variables using scikit-learn tools in Python.
- **Exploratory Data Analysis (EDA):** Analyzed player attributes using pandas and seaborn packages in Python, identifying correlations between attributes and player positions, visualizing insights for better understanding.
- **Feature Engineering:** Conducted feature engineering by modifying player features for improved model prediction through scaling and transformation using algorithms like Min-Max Scaling or Standard Scaling.
- **Model Training for Position Prediction and Clustering :** Split data into training and testing sets.Use classification algorithms (Logistic Regression, Random Forest, XGB, Adaboost) for position prediction and Use clustering algorithms (Kmeans, Hierarchical Clustering) for grouping players.
- **Integration of Prediction and Clustering:** Combined position predictions and player clustering results.Develop a recommendation system for suitable positions.
- **User Interface Development:** Design an intuitive UI for SkillFusionX.Allow users to input data and receive predictions.

4.3 NONFUNCTIONAL REQUIREMENTS

- **Usability:** User interface is easy to navigate for coaches, recruiters, or team managers.
- **Performance:** SkillFusionX will provide quick responses for data analytics and predictions.
- **Compatibility:** SkillFusionX is compatible with different devices and browsers to enhance accessibility.

5.SYSTEM DESIGN

5.1 Architecture Daigram

Presentation Layer: Represents the front-end layer that users interact with.

Business Layer : Contains the business logic and processing logic of the system by the user.

Service Layer: Manages the communication between the Business layer and the database by managing upload, extracting player position and information, search path using query

Data Layer : Stores and manages the data used by the service layer using JSON and Hashmap

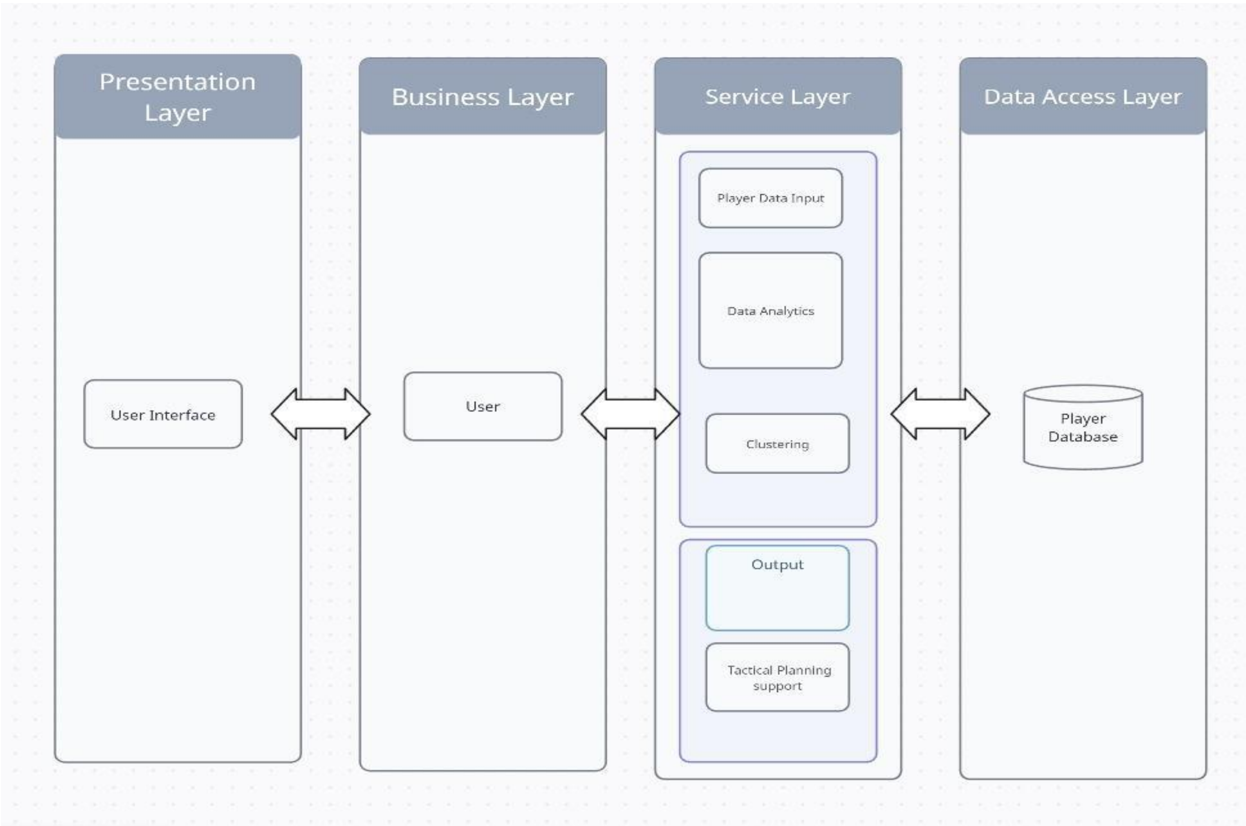


Fig 5.1: Architecture Diagram

5.2 Block Diagram/Context Daigram

A context diagram illustrates the system at its core, surrounded by external entities like users, data sources, and other applications. It visually represents the interactions between SkillFusionX and these external elements through arrows or lines, indicating the flow of information or interactions. This diagram provides a concise overview of the system's relationships with its external environment, aiding in understanding its scope and boundaries.

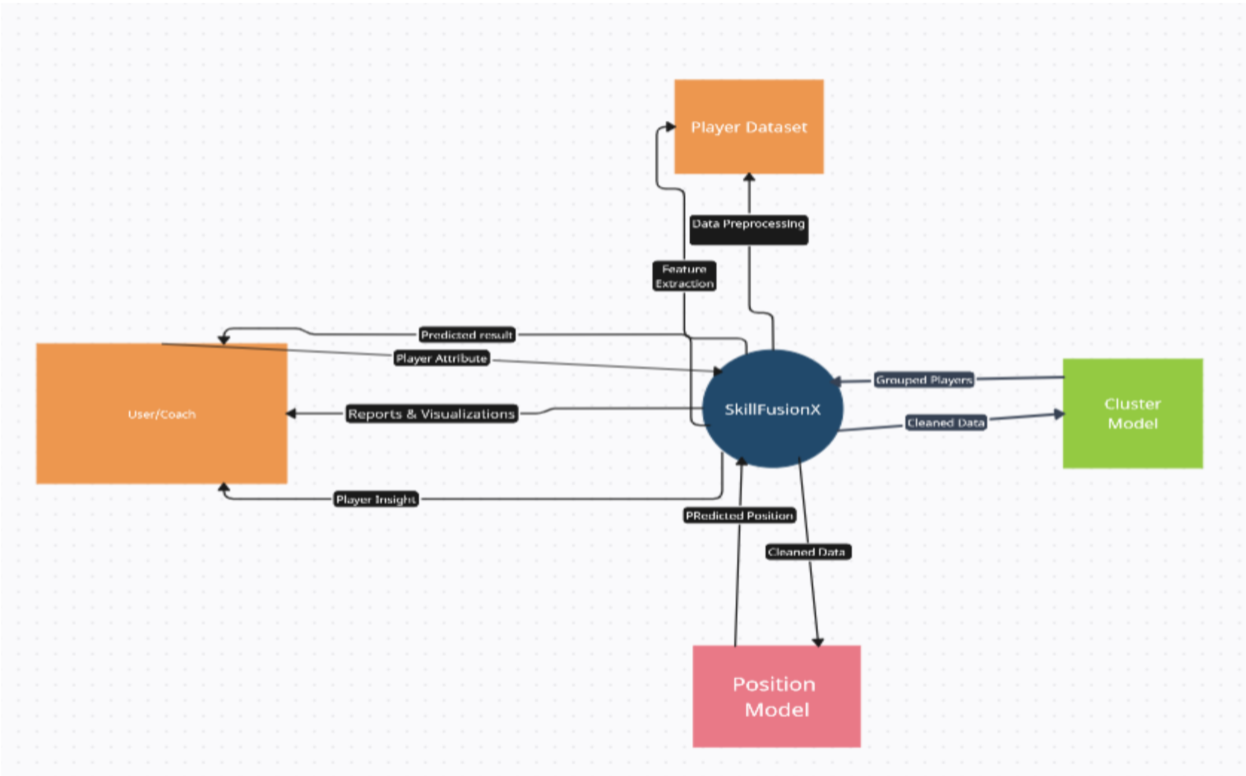


Fig 5.2: Context Diagram

6.DETAILED DESIGN

6.1 Process flow Diagram

The process flow starts with steps such as input collection of player attributes, data processing through machine learning algorithms for position prediction and skill-based clustering, output generation for visualization, user interface update to display insights, real-time integration of new data, and an iterative process allowing for feedback and adjustments. This succinctly outlines the workflow from data input to visualization and continuous improvement in managing player positions and team strategy.

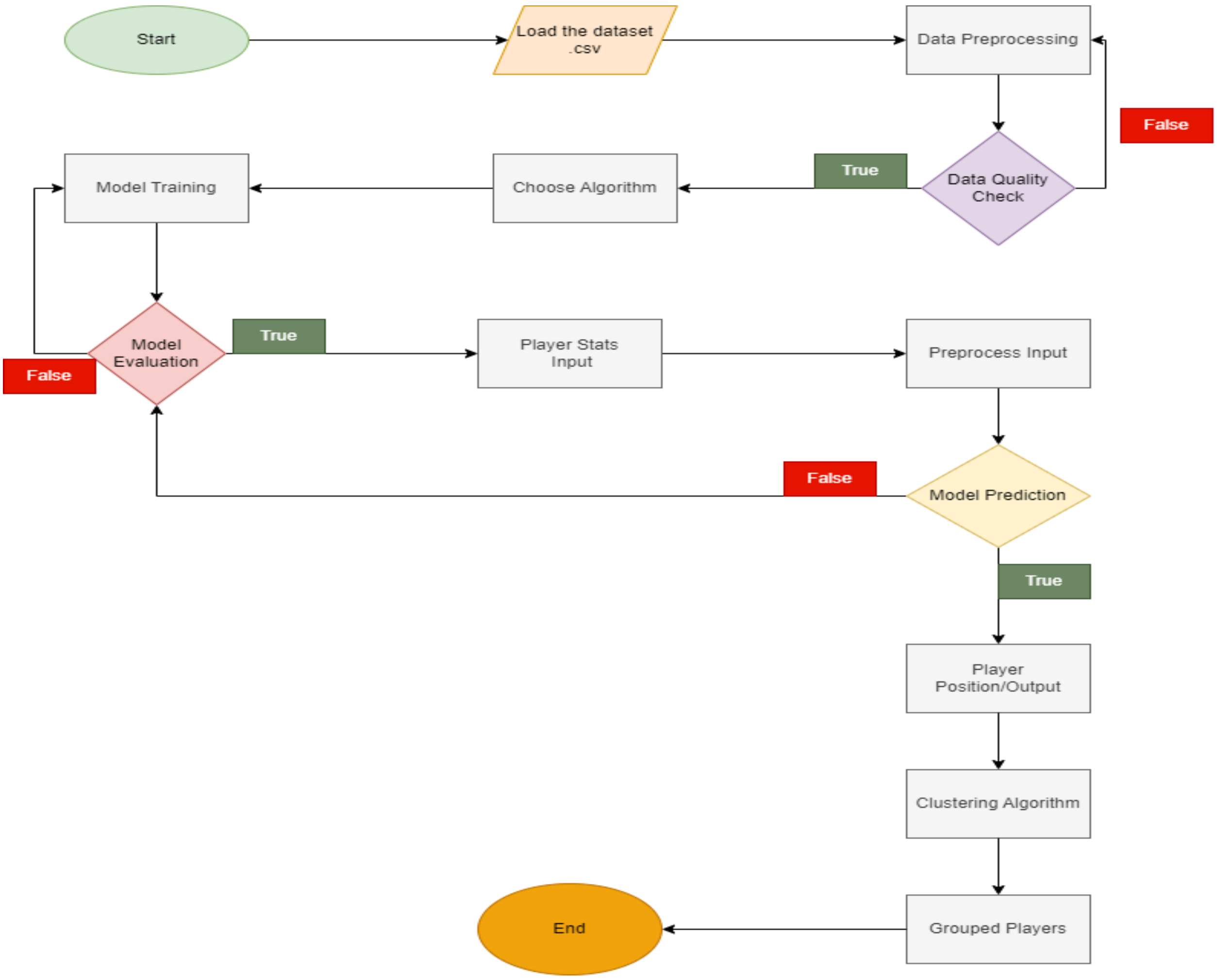


Fig 6.1: Process Flow Diagram

6.2 Class Diagram

Class Diagram shows the association and relationship between the entities in the System

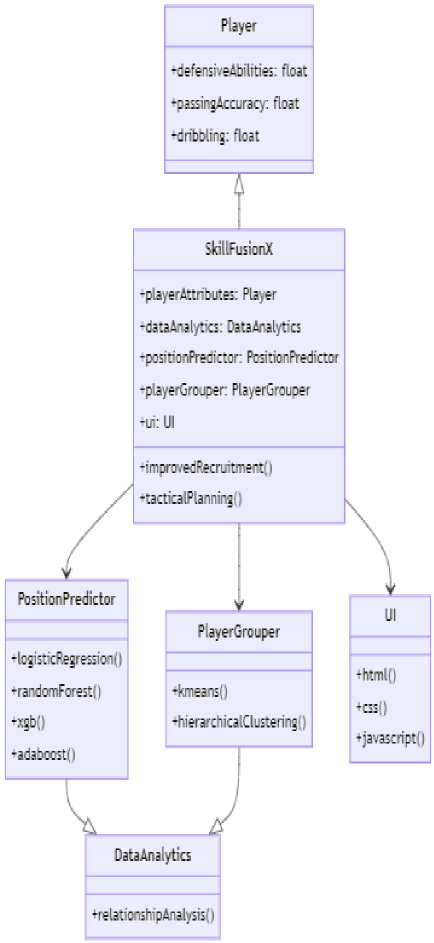


Fig 6.2 : Class Diagram

6.3 Use case Diagram

This use case diagram illustrates the primary interactions between actors and the SkillFusionX system.

Actor: User

Use Cases : user input player data, view position predictions and cluster players by skills. user position predictions and Do data Analysis.

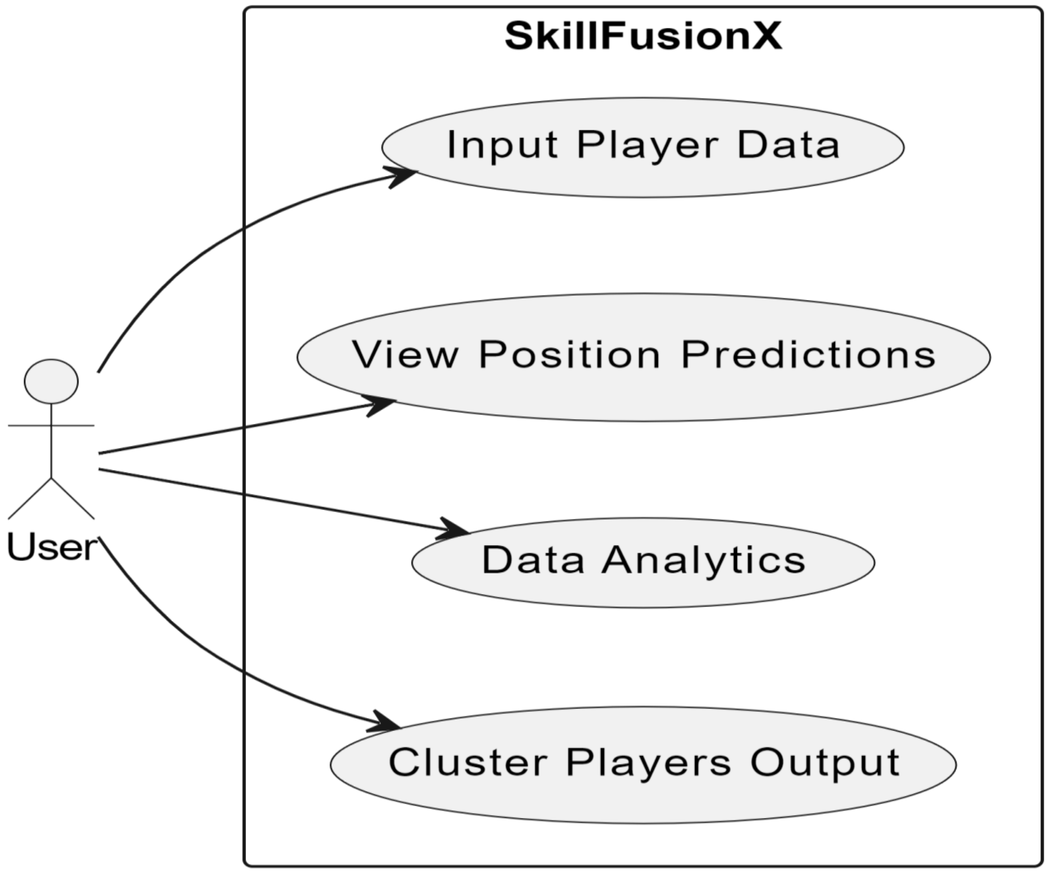


Fig 6.3 : Use Case Diagram

6.4 Sequence Diagram

The sequence diagram for SkillFusionX involves user input of player attributes, data processing using algorithms for position prediction and player clustering, output generation with updated insights, and real-time integration of new data, enabling iterative adjustments based on user feedback.

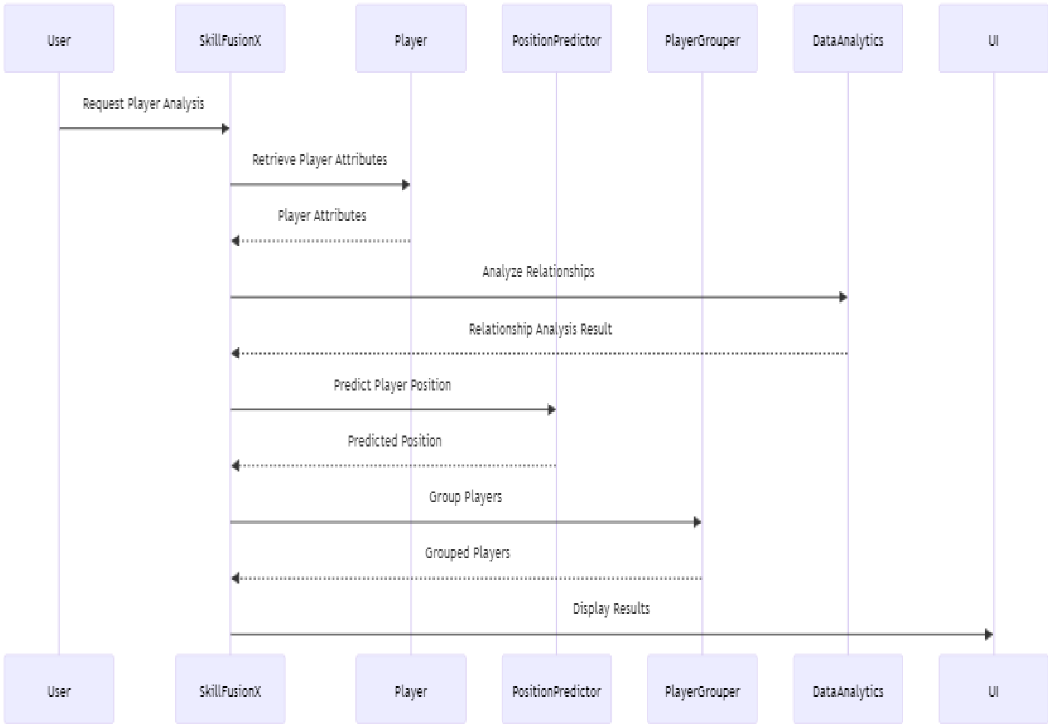


Fig 6.4 : Sequence Diagram

6.5 Methodology

Data Collection – Gather FIFA player data from url : https://fbref.com/en/comps/1/history/World-Cup-Seasons.

Data Cleaning and Preprocessing – Performed data cleaning by handling missing values and outliers, normalized numerical features, and encoded categorical variables using scikit-learn tools in Python.

Position Prediction – Split data into training and testing sets.Use classification algorithms (Logistic Regression, Random Forest, XGB, Adaboost) for position prediction

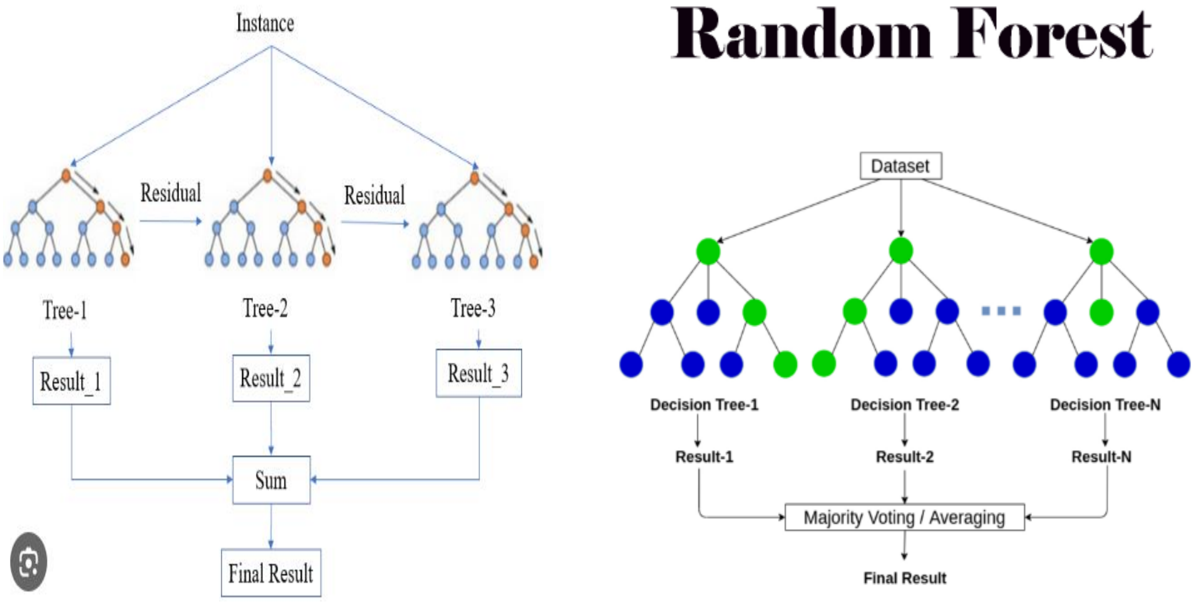


Fig 6.5 : XGB and Random Forest For Position Prediction

Clustering: Use clustering algorithms (Kmeans, Hierarchical Clustering) for grouping players with the same skillset.

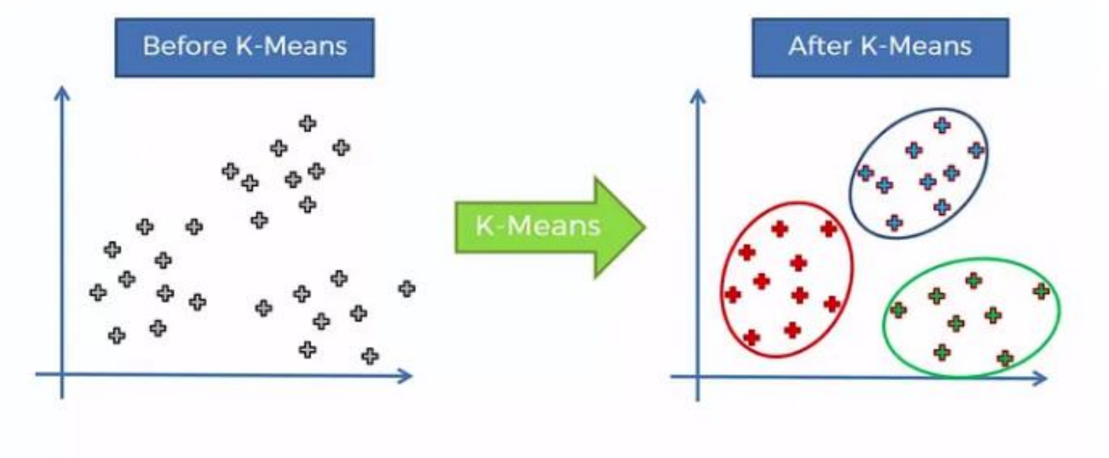


Fig 6.6: K-Mean Clustering Process to group Players

7.IMPLEMENTATION

7.1 Source Code

In [3]:
```
data = pd.read_csv("players_fifa23.csv")
data.head()
```

Out[3]:

	ID	Name	FullName	Age	Height	Weight	PhotoUrl	Nationality	Overall	Potential	...	LMRating	CMRating
0	158023	L. Messi	Lionel Messi	35	169	67	https://cdn.sofifa.net/players/158/023/23_60.png	Argentina	91	91	...	91	88
1	165153	K. Benzema	Karim Benzema	34	185	81	https://cdn.sofifa.net/players/165/153/23_60.png	France	91	91	...	89	84
2	188545	R. Lewandowski	Robert Lewandowski	33	185	81	https://cdn.sofifa.net/players/188/545/23_60.png	Poland	91	91	...	86	83
3	192985	K. De Bruyne	Kevin De Bruyne	31	181	70	https://cdn.sofifa.net/players/192/985/23_60.png	Belgium	91	91	...	91	91
4	231747	K. Mbappé	Kylian Mbappé	23	182	73	https://cdn.sofifa.net/players/231/747/23_60.png	France	91	95	...	92	84

5 rows × 90 columns

In [4]:
```
data.shape
```

Out[4]: (18539, 90)

In [5]:
```
needed_columns = ['ID', 'Name', 'Age', 'Height', 'Weight',
        'Overall', 'Potential', 'Growth', 'TotalStats',
        'BaseStats', 'BestPosition', 'Club', 'ValueEUR', 'WageEUR',
        'ReleaseClause', 'ContractUntil', 'ClubJoined', 'OnLoad',
        'PreferredFoot', 'IntReputation', 'WeakFoot','Nationality',
        'SkillMoves', 'AttackingWorkRate', 'DefensiveWorkRate', 'PaceTotal',
        'ShootingTotal', 'PassingTotal', 'DribblingTotal', 'DefendingTotal',
        'PhysicalityTotal', 'Crossing', 'Finishing', 'HeadingAccuracy',
        'ShortPassing', 'Volleys', 'Dribbling', 'Curve', 'FKAccuracy',
        'LongPassing', 'BallControl', 'Acceleration', 'SprintSpeed', 'Agility',
        'Reactions', 'Balance', 'ShotPower', 'Jumping', 'Stamina', 'Strength',
        'LongShots', 'Aggression', 'Interceptions', 'Positioning', 'Vision',
        'Penalties', 'Composure', 'Marking', 'StandingTackle', 'SlidingTackle',
        'GKDiving', 'GKHandling', 'GKKicking', 'GKPositioning', 'GKReflexes']


data = data[needed_columns]
```

Fig 7.1 Dataset Overview

Data Preprocessing:

1. Handle the missing values:

In [6]:
```
for i in data.columns:
    if data[i].isnull().sum() > 0:
        print(i, " ", data[i].isnull().sum())
```

```
ContractUntil   92
```

In [ ]:
```
data.na()
```

In [9]:
```
print("The percentage of the null values is: ", (data["ContractUntil"].isnull().sum()/data.shape[0])*100, "%")
```

```
The percentage of the null values is:  0.49625114623226707 %
```

As the percentage of nulls is small so we can drop the values of nulls in that column.

In [10]:
```
data.drop(data[data["ContractUntil"].isnull()].index, axis = 0, inplace =True)
```

In [11]:
```
data["ContractUntil"].isnull().sum()
```

Out[11]: 0

2. Handle The Categorical Columns:

In [12]:
```
for i in data.columns:
    if data[i].dtype == 'object':
        print(i)
```

```
Name
BestPosition
Club
PreferredFoot
Nationality
AttackingWorkRate
DefensiveWorkRate
```

In [13]:
```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

data["PreferredFoot"] = le.fit_transform(data["PreferredFoot"])
data["AttackingWorkRate"] = le.fit_transform(data["AttackingWorkRate"])
data["DefensiveWorkRate"] = le.fit_transform(data["DefensiveWorkRate"])
data["Club"] = le.fit_transform(data["Club"])
```

Merge Some Players Positions to reduce the number of classes:

```
In [ ]: merge_pos = {'LWB' : 'LW', 'RWB': 'RW', 'ST': 'CF', 'CAM': 'CM', 'CDM': 'CM'}

        data = data.replace({'BestPosition': merge_pos})
```

```
In [ ]: mapping = {'CF': 0, 'CM': 1, 'RW': 2, 'GK': 3, 'CB': 4, 'LW': 5, 'LM': 6, 'LB': 7,'RM': 8, 'RB': 9}

        data = data.replace({'BestPosition': mapping})
```

```
In [ ]: data = pd.DataFrame(data)

        data.to_csv("players_fifa23_cleaned.csv", index=False)
```

Split the Data to Train and Test sets:

```
In [ ]: X = data.drop(["BestPosition", "Nationality", "ID"], axis = 1)
        Y = pd.DataFrame(data["BestPosition"])

        top = data.sort_values(by=["Overall"], ascending=False).head(20)
```

```
In [ ]: from sklearn.model_selection import train_test_split
        X_Train, X_Test, Y_Train, Y_Test = train_test_split(X, Y, test_size=0.20, random_state=1)
```

```
In [ ]: X_Train = X_Train.drop(["Name"], axis = 1)
        test_names = X_Test["Name"]
        X_Test = X_Test.drop(["Name"], axis = 1)

        top_pos = top["BestPosition"]
        top_names = top["Name"]
        top = top.drop(["Name", "BestPosition", "Nationality", "ID"], axis = 1)
        top
```

```
In [ ]: from sklearn.preprocessing import MinMaxScaler

        mms = MinMaxScaler()

        X_Train = mms.fit_transform(X_Train)
        X_Test = mms.fit_transform(X_Test)

        top = mms.fit_transform(top)
```

```
In [ ]: print(f' X_shape: {X_Train.shape} \n y_shape: {Y_Train.shape}')
```

```
In [ ]:
```

Fig 7.2 Data Preprocessing and Training

7.2 Screenshots

EDA

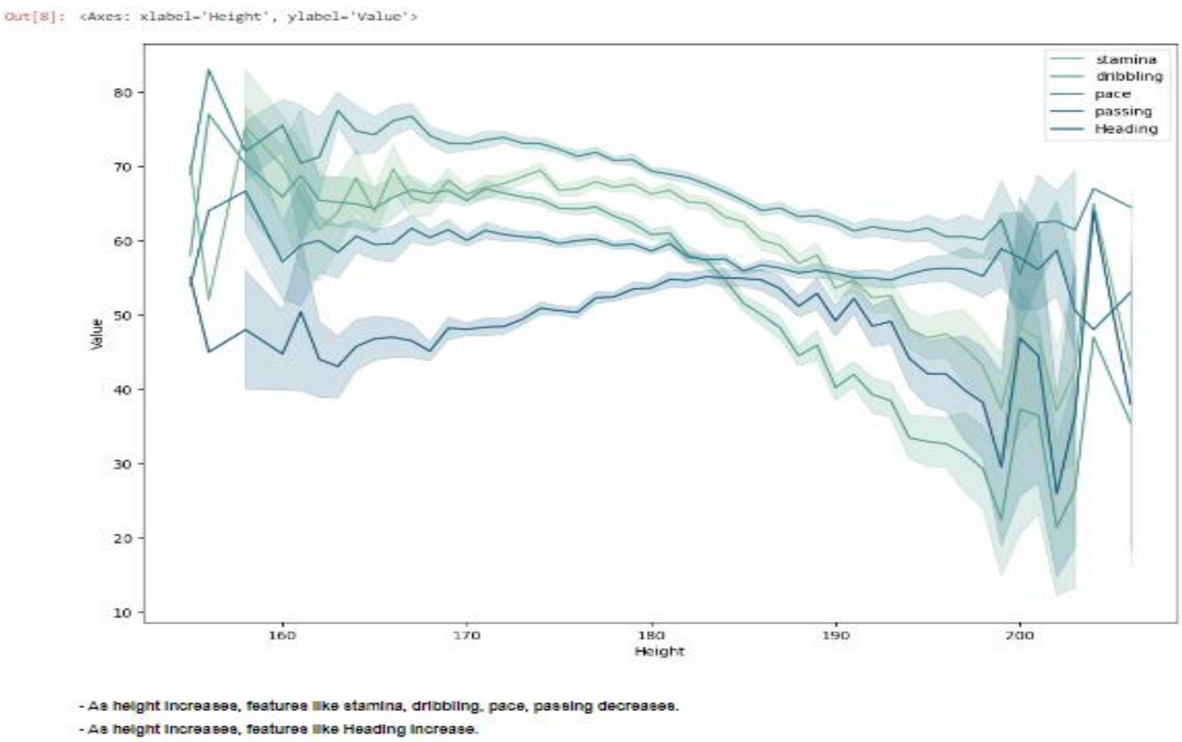


Fig 7.3 EDA of different columns

4. Show the top Quikest Players:

t[11]: <Axes: xlabel='Name', ylabel='Acceleration'>

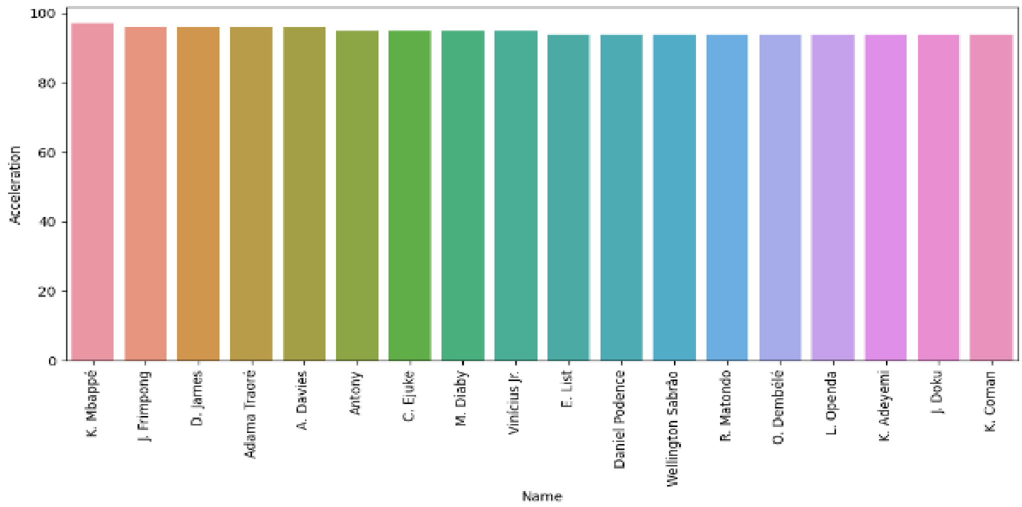


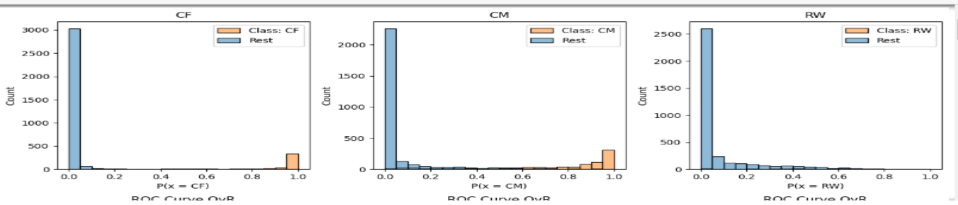
Fig 7.4 EDA of different columns

1.Logistic Regression

```
    macro avg       0.68      0.64      0.61      3690
 weighted avg       0.81      0.77      0.76      3690
```

```
In [37]: Y_Proba_lr = lr.predict_proba(X_Test)
```

```
In [38]: draw_pco(Y_Proba_lr, ['CF', 'CM', 'RW'])
         draw_pco(Y_Proba_lr, ['GK', 'CB', 'LW'])
         draw_pco(Y_Proba_lr, ['LM', 'LB','RM'])
         draw_pco(Y_Proba_lr, ['RB'])
```




```
In [39]: auc_score_lr = roc_auc_score(Y_Test, Y_Proba_lr, multi_class = 'ovr', average = 'macro')
         auc_score_lr
Out[39]: 0.9648498298163668
```

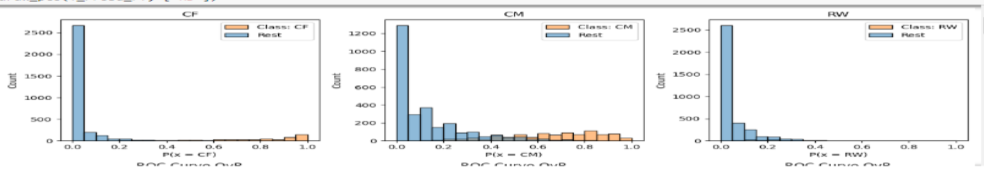
2. Random Forest:

```
The Classification Report for Random Forest Classifier:
              precision    recall  f1-score   support

           0       0.89      0.93      0.91       529
           1       0.72      0.91      0.80       921
           2       0.36      0.29      0.32       140
           3       1.00      1.00      1.00       404
           4       0.89      0.90      0.89       748
           5       0.45      0.17      0.25       116
           6       0.68      0.44      0.54       191
           7       0.65      0.78      0.71       167
           8       0.66      0.40      0.49       288
           9       0.66      0.53      0.59       186

    accuracy                           0.79      3690
   macro avg       0.70      0.64      0.65      3690
weighted avg       0.77      0.79      0.77      3690
```

Use the ROC Curve and ROC AUC to Evaluate the model:

```
In [51]: Y_Proba_rf = grid_search.predict_proba(X_Test)
```

```
In [52]: draw_pco(Y_Proba_rf, ['CF', 'CM', 'RW'])
         draw_pco(Y_Proba_rf, ['GK', 'CB', 'LW'])
         draw_pco(Y_Proba_rf, ['LM', 'LB','RM'])
         draw_pco(Y_Proba_rf, ['RB'])
```




```
In [53]: auc_score_rf = roc_auc_score(Y_Test, Y_Proba_rf, multi_class = 'ovr', average = 'macro')
         auc_score_rf
```

Fig 7.5 Position Prediction Model Building

Prediction Model Evaluation

Compare the Test Accuracy of the 7 Classification Algorithms:

Based on the Accuracy:

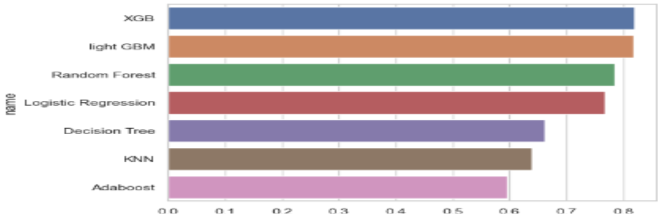
In [112]:
```
comp = pd.DataFrame()
comp['name'] = models_names
comp['score'] = models_scores
comp
```
Out[112]:

	name	score
0	Logistic Regression	0.768022
1	Random Forest	0.785366
2	Decision Tree	0.662080
3	Adaboost	0.596477
4	XGB	0.818699
5	light GBM	0.816802
6	KNN	0.638482

In [113]:
```
cm = sns.light_palette("green", as_cmap=True)
s = comp.style.background_gradient(cmap=cm)
s
```
Out[113]:

	name	score
0	Logistic Regression	0.768022
1	Random Forest	0.785366
2	Decision Tree	0.662080
3	Adaboost	0.596477
4	XGB	0.818699
5	light GBM	0.816802
6	KNN	0.638482

In [114]:
```
sns.set(style="whitegrid")
ax = sns.barplot(y="name", x="score", data=comp.sort_values(by="score", ascending=False))
```



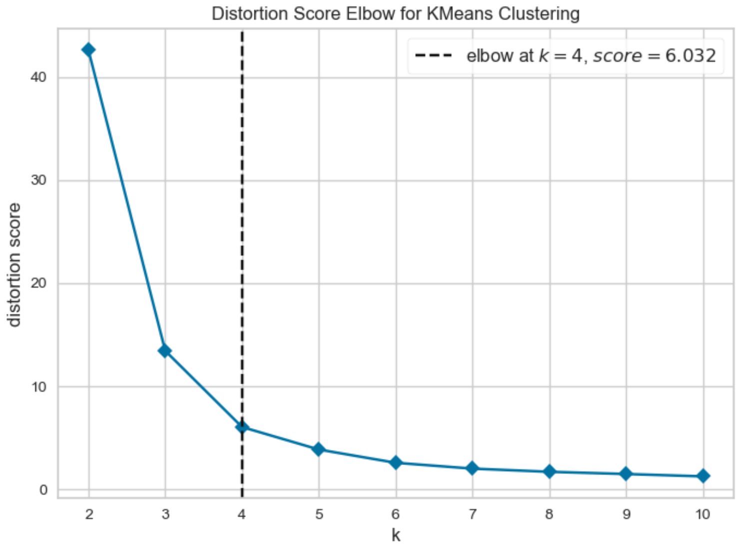
Xgb is Classification Aglorithmm giving best result for this dataset

Test the Algorithm on the top 20 Players:

```
Name: L. Messi
Prediction: CM
 True Best Pos: CM

Name: R. Lewandowski
Prediction: CF
 True Best Pos: CF

Name: K. De Bruyne
Prediction: CM
 True Best Pos: CM

Name: K. Mbappé
Prediction: CF
 True Best Pos: CF

Name: K. Benzema
Prediction: CF
 True Best Pos: CF

Name: M. Salah
Prediction: RW
 True Best Pos: RW

Name: T. Courtois
Prediction: GK
 True Best Pos: GK

Name: M. Neuer
Prediction: GK
 True Best Pos: GK

Name: Cristiano Ronaldo
Prediction: CF
 True Best Pos: CF
```

Fig 7.6 Model Evaluation and Test Algorithm for top 20 players

Clustering Model



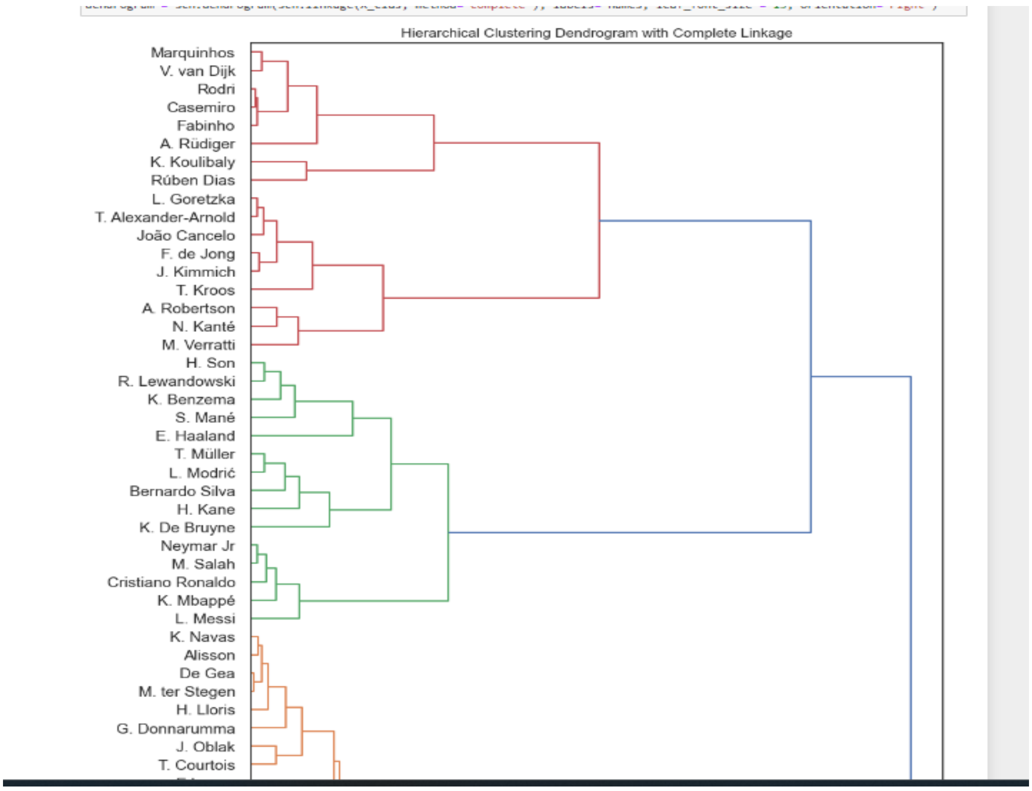


Fig 7.7 Clustering on different Algorithms

Compare the 4 clustering Algorithms based on silhouette Score

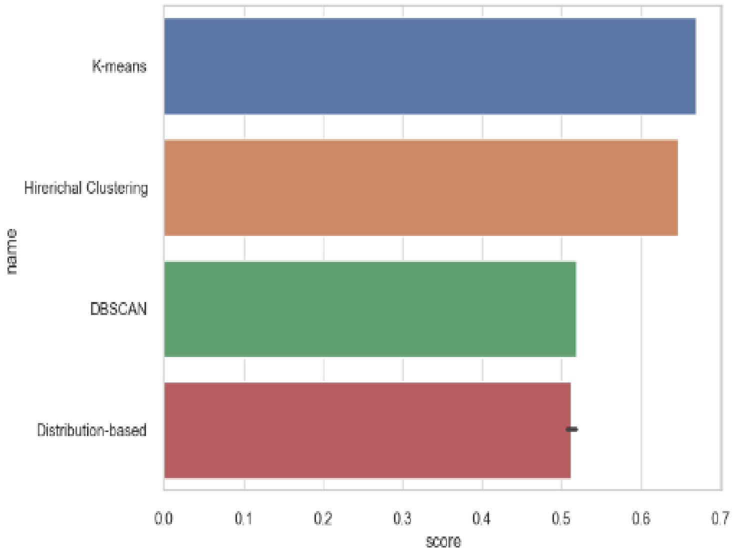
Out[175]:

	name	score
0	K-means	0.669469
1	Hirerichal Clustering	0.646751
2	DBSCAN	0.518514
3	Distribution-based	0.518514
4	Distribution-based	0.507780
5	Distribution-based	0.507780

In [176]:
```
cm = sns.light_palette("green", as_cmap=True)
s = comp.style.background_gradient(cmap=cm)
s
```

Out[176]:

	name	score
0	K-means	0.669469
1	Hirerichal Clustering	0.646751
2	DBSCAN	0.518514
3	Distribution-based	0.518514
4	Distribution-based	0.507780
5	Distribution-based	0.507780



So Based on The Silhouette Score, The Hirerichal Clustring and the K-means are the best Clustering Algorithms for this Data.

Fig 7.8 Clustering Model evaluation on silhouette Score

User Interface

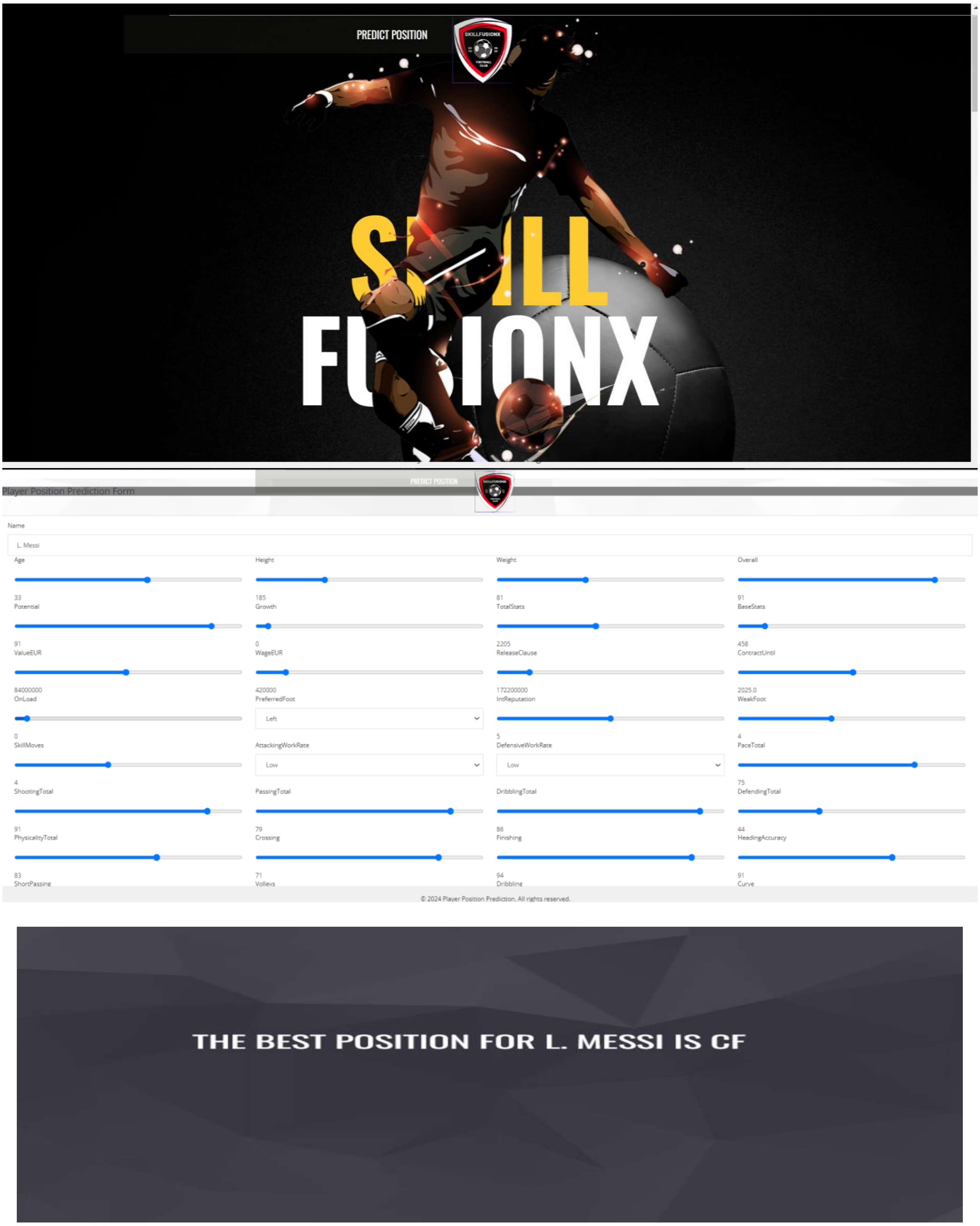


Fig 7.9 User interface

POSITIONS AND FULL FORMS

Abbreviation	Full Form
CF	Center Forward
CM	Central Midfielder
RW	Right Winger
GK	Goalkeeper
CB	Center Back
LW	Left Winger
LM	Left Midfielder
LB	Left Back
RM	Right Midfielder
RB	Right Back

Fig 7.10 User interface

8. BIBILOGRAPHY(References)

[1] Chazan-Pantzalis, V., & Tjortjis, C. (2020). Sports Analytics for Football League Table and Player Performance Prediction. Presented at the 11th International Conference on Information, Intelligence, Systems and Applications (IISA). Publisher: Victor Chazan-Pantzalis (v.chazan-pantzalis@ihu.edu.gr).
[2] Babu, S. B., Vivek, V., Kumar, D. M. T., Prathyusha, K., & Teja, G. P. (2022). "Predicting Football Player's Position." International Research Journal of Modernization in Engineering Technology and Science.
[3]Cheng, G., Zhang, Z., Kyebambe, M. N., & Kimbugwe, N. "Predicting the Outcome of NBA Playoffs Based on Maximum Entropy Principle." Department of Mathematics and Computational Science, Xiangtan University, China.
[4] Verstraete, K., Coussement, B., & Vannieuwenhoven, N. "Analyzing Soccer Players' Skill Ratings Over Time Using Tensor-Based Methods." Presented at the 2020 Joint European Conference on Machine Learning and Knowledge Discovery in Databases
[5] Hucaljuk, J., & Rakipović, A. "Predicting Football Scores Using Machine Learning Technique." Presented at MIPRO 2011, May 23-27, 2011, Opatija, Croatia.