ML MINI PROJECT - TASK 4 - REPORT

NAME: FIONA HARIA **SAP-ID:** 600009220048

ROLL-NO: D040 **BATCH:** D1-1

TITLE OF THE PROJECT

Football Player Price Prediction using Machine Learning

AIM OF THE PROJECT

The transfer market in football is a multi-million dollar industry driven by complex player valuations. This project aims to develop a machine learning model capable of predicting the transfer market-value of football players with a reasonable degree of accuracy. By analyzing historical transfer data, player performance metrics, and other relevant factors, the model will provide valuable insights for clubs in making informed transfer decisions.

DATA DESCRIPTION

The dataset used here is the fifaindex_21.csv which contains the data of over 18000 players in the FIFA league. It has a total of 42 features which we will be using our model to train on. Here are all the columns of the dataset by category: For each feature they have a rate over 100:

1. **Player Description**: This category includes basic information about the player such as age, market value, and salary. Age indicates experience and potential, while market value and salary reflect the player's worth in the transfer market and their earnings, respectively.
2. **Technical Skills**: Technical skills encompass a player's ability to control and manipulate the ball effectively, including ball control and dribbling. These skills are fundamental for maintaining possession and beating opponents.

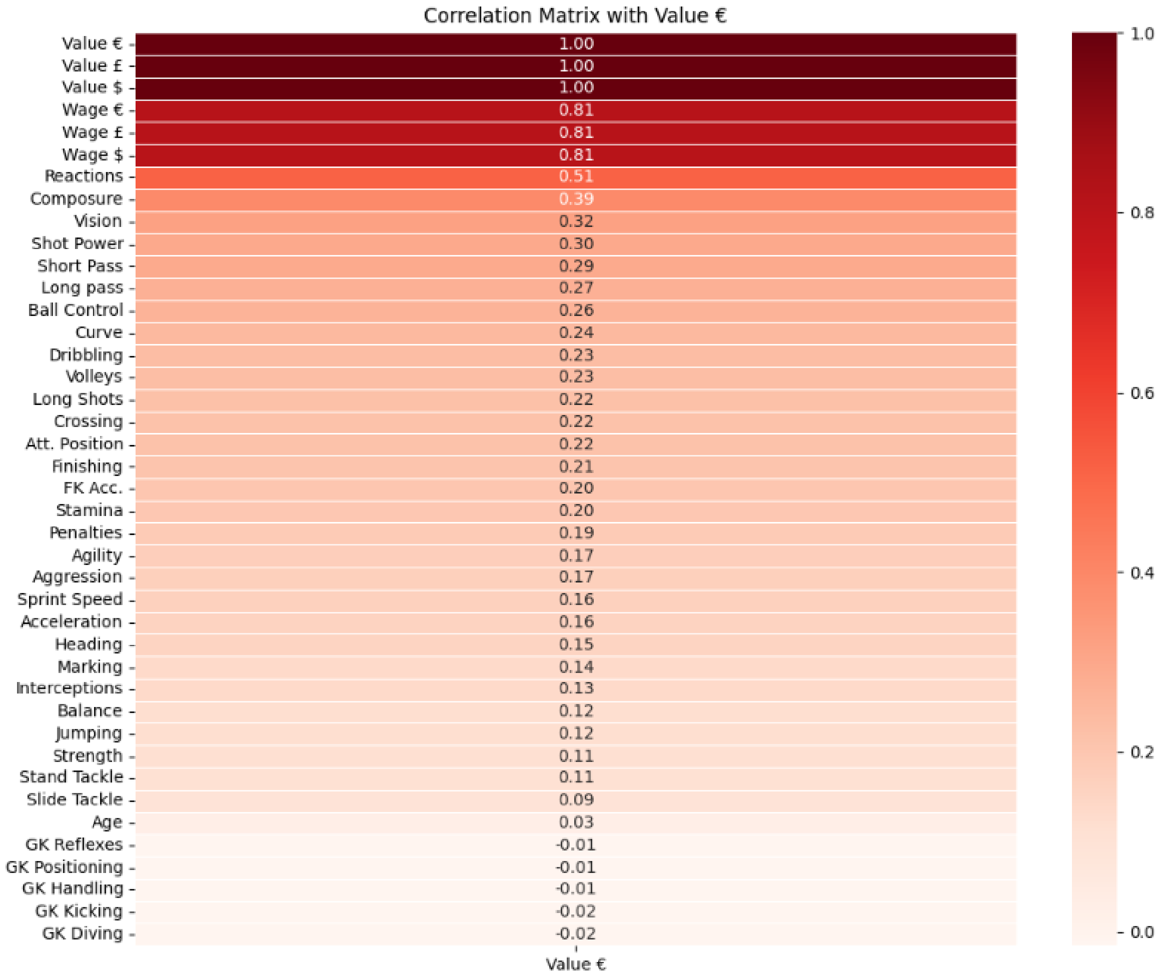
3. **Defense:** Defensive skills focus on a player's ability to thwart opponents' attacks. This includes man-to-man marking, sliding tackle, and standing tackle, essential for dispossessing opponents and regaining possession.
4. **Aptitude:** Aptitude encompasses various mental and tactical attributes crucial for success on the field. It includes commitment, responsiveness, positioning offensively and defensively, interception, vision, and discipline.
5. **Passes:** Passing skills involve delivering the ball accurately to teammates. This includes cross passes, short passes, and long passes, which are essential for maintaining possession, creating scoring opportunities, and switching play.
6. **Physical Attributes:** Physical attributes describe a player's physical capabilities. These include pace, endurance, strength, balance, speed, agility, and vertical jump, which collectively determine a player's athleticism and ability to compete at a high level.
7. **Shots:** Shooting skills pertain to a player's ability to score goals. This includes headers, shot strength, finishing, long shots, effect (adding spin or curve), precision free kicks, penalties, and volleys.
8. **Goalkeeper Abilities:** Goalkeeper abilities focus on the specific skills required for goalkeeping. This includes positioning, diving, handling (using hands to stop or catch the ball), kicking (distribution), and reflexes, crucial for making saves and preventing goals.

This data offers a holistic view of a player's capabilities, allowing for a more accurate assessment of their potential impact on the team. Moreover, understanding a player's strengths and weaknesses across different categories enables clubs to identify suitable transfer targets based on their specific needs and playing style. Overall, by leveraging comprehensive player data across multiple categories, clubs and analysts can enhance their player price prediction models,

DATA PRE-PREPROCESSING

Data preprocessing is a critical step in any machine learning project, as it lays the foundation for accurate and meaningful insights to be derived from the dataset. inconsistencies, errors, and missing values have been addressed ensuring that the data is suitable for analysis. Additionally, graphical representations were utilized to visualize the distribution of key variables and identify patterns and outliers within the dataset.

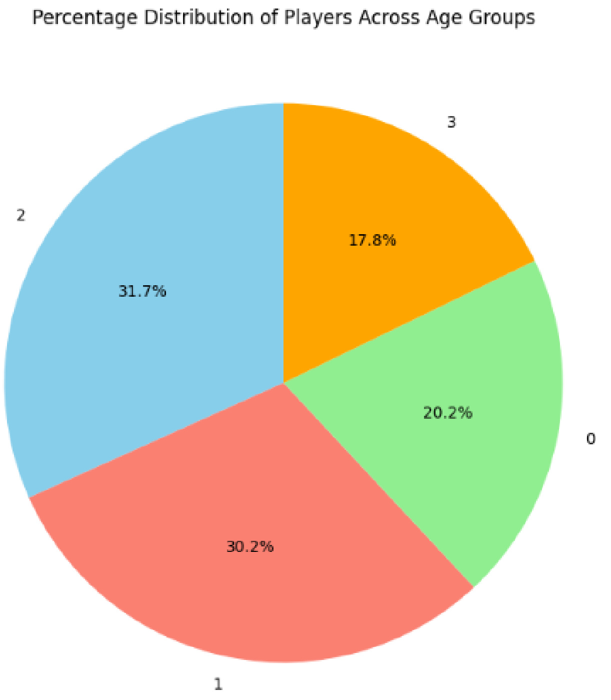
1. A total of 1.4% of the data had **missing values** in the Value column which is the target column. We have dropped the missing values and used this data later for testing.
2. To undestand which in-game attributes contribute most to a player's overall value, a **correlation matrix** against the Value column has been created

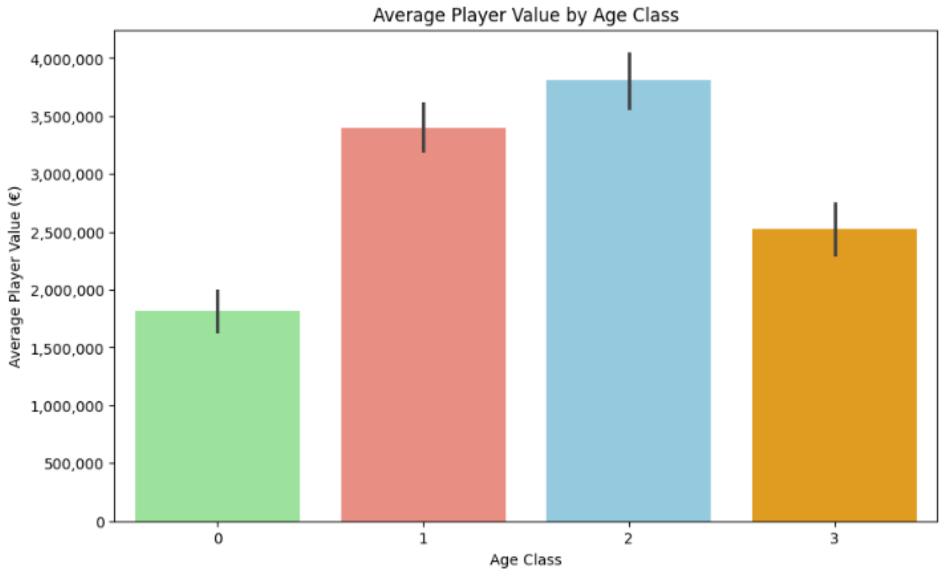


Players with higher values tend to have higher wage. Players with higher values tend to have better stats in the areas of Reactions, Composure, Vision, Short Pass, Shot Power, Long Pass Players with higher values tend to have lower ratings for goalkeeping stats

3. **Age classification encoding** facilitates the identification of age-related trends or patterns in the dataset, which may have implications for various analyses or modeling tasks

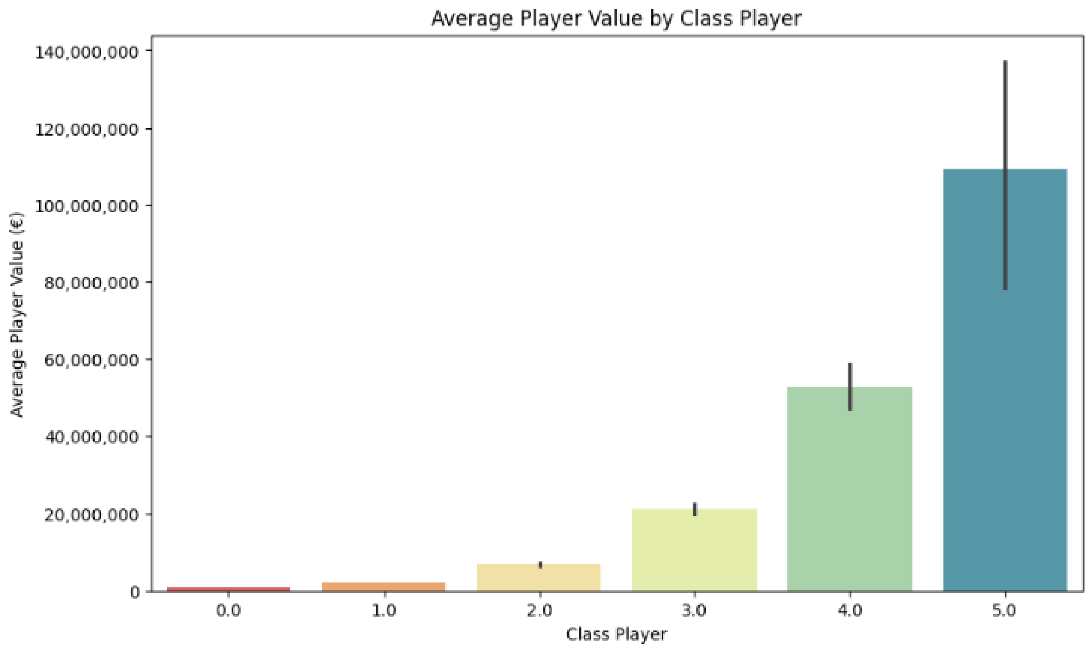
- If the age of the player is less than 21, Encoder = 0
- If the age of the player is between 21 and 25, Encoder = 1
- If the age of the player is between 25 and 30, Encoder = 2
- If the age of the player is more than 30, Encoder = 3

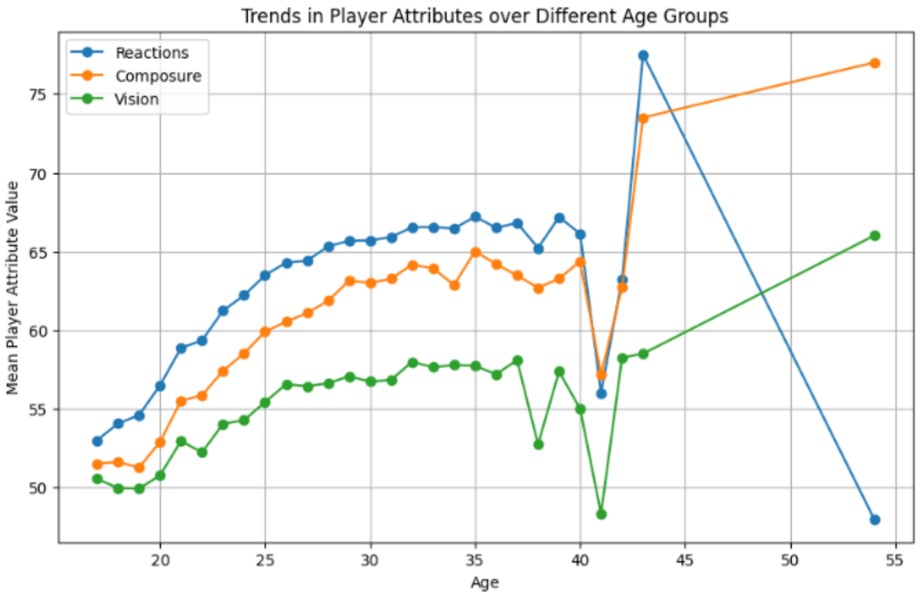




There is a significant increase in value from the Class 0 to Class 1. This suggests that younger players are steadily increasing in value as their skills develop and they gain experience. Between the age of 25-30, often considered as a prime of a player's career, is when they have reached their peak athletic ability and skill level.

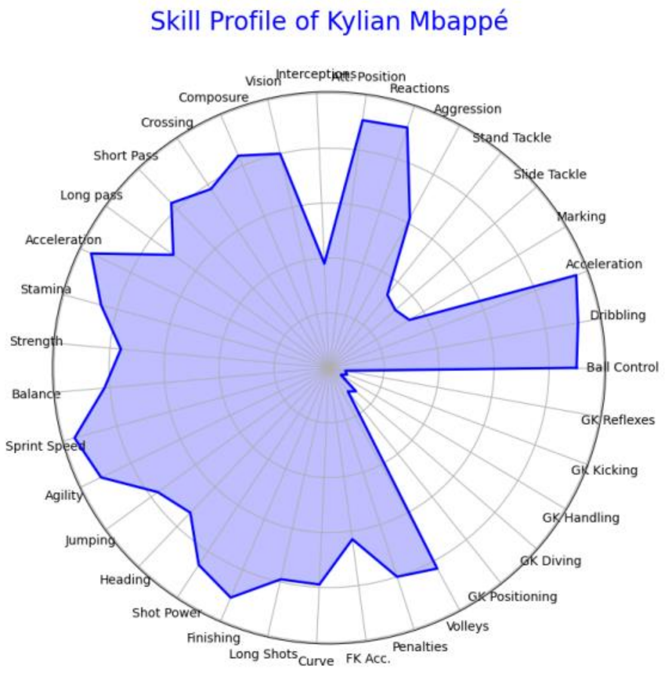
4. **The player rating classification** encoding strategy is devised to categorize players based on their perceived skill levels, as reflected by their ratings. This approach serves to simplify the analysis by grouping players into distinct tiers of performance, allowing for easier interpretation and comparison across skill levels.
If the player has 5 times more than 90 rating : he is a world class player. Encoder = 5
If he has 5 times more than 85 rating : he is a very good player. Encoder = 4
If he has 5 times more than 80 rating : he is a good player. Encoder = 3
If he has 5 times more than 75 rating : he is a normal player. Encoder = 2
If he has 5 times more than 70 rating : he is a basic player. Encoder = 1
If he has 5 times more than 60 rating : he is a district player. Encoder = 0





The difference in reaction score descreases significantly the age of 40. Composure appears to increase with age. This could be because older players have more experience dealing with pressure situations

5. **The skill profiles** for the top 4 players are determined based on a comprehensive evaluation of their performance across various attributes and metrics. The skill profiles assigned to them reflect their proficiency in key areas of the game, which align with their respective positions and playing style. For eg: The skill profile of Kylian Mbappe shows that he has exceptional proficiency in accerelation and sprint speed.



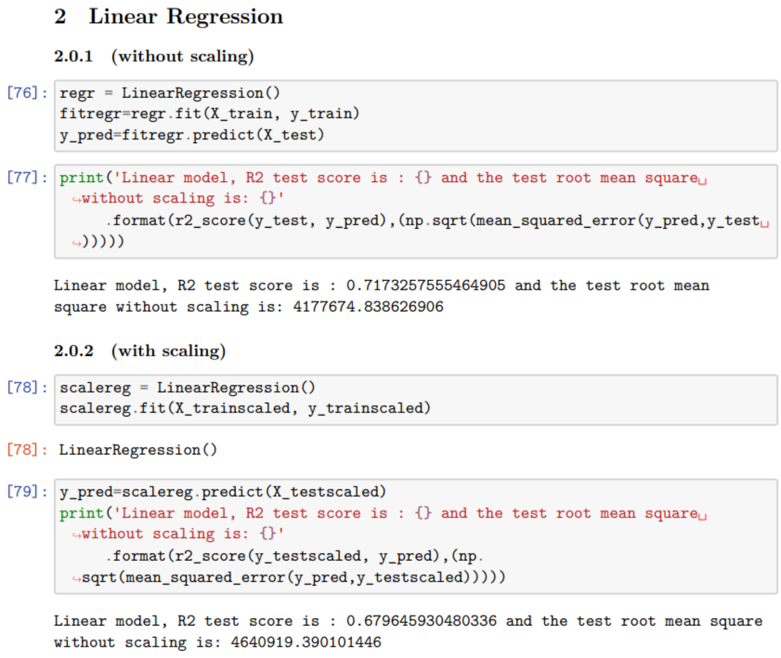
DATA MODELLING

In the pursuit of accurately predicting football player prices, a variety of regression models were employed and evaluated within the framework of the machine learning project. These models encompassed a diverse range of methodologies, each offering unique advantages and considerations in the context of player price prediction.

Additionally we have standardizing as it was a crucial step in preparing it for the machine learning modeling phase of the project. By transforming the features to have a mean of 0 and a standard deviation of 1, we ensured that all variables contributed equally to the modeling process. Let’s look at the models we have used and their accuracies.

1. Linear Regression

Linear Regression was employed as a baseline model to establish a simple linear relationship between player attributes and their respective prices. Despite its simplicity, Linear Regression can provide valuable insights into the linear relationships between predictor variables and the target variable. However, it may not capture non-linear relationships present in the data.



2 Linear Regression

2.0.1 (without scaling)

```
[76]: regr = LinearRegression()
      fitregr=regr.fit(X_train, y_train)
      y_pred=fitregr.predict(X_test)
```

```
[77]: print('Linear model, R2 test score is : {} and the test root mean square␣
       ↪without scaling is: {}'
          .format(r2_score(y_test, y_pred),(np.sqrt(mean_squared_error(y_pred,y_test␣
       ↪)))))
```

```
Linear model, R2 test score is : 0.7173257555464905 and the test root mean
square without scaling is: 4177674.838626906
```

2.0.2 (with scaling)

```
[78]: scalereg = LinearRegression()
      scalereg.fit(X_trainscaled, y_trainscaled)
```

```
[78]: LinearRegression()
```

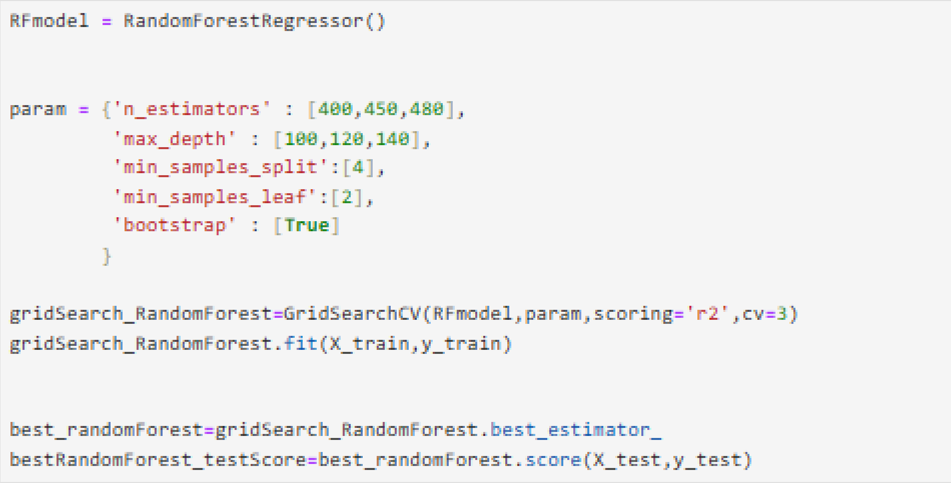
```
[79]: y_pred=scalereg.predict(X_testscaled)
      print('Linear model, R2 test score is : {} and the test root mean square␣
       ↪without scaling is: {}'
          .format(r2_score(y_testscaled, y_pred),(np.
       ↪sqrt(mean_squared_error(y_pred,y_testscaled))))
```

```
Linear model, R2 test score is : 0.679645930480336 and the test root mean square
without scaling is: 4640919.390101446
```

The results on data without scaling is better than scaled data Linear Regression with an accuracy of 71%

2. **Random Forest Regressor**

Random Forest Regression was chosen for its ability to handle non-linear relationships and interactions within the data. By leveraging an ensemble of decision trees, Random Forest Regression offers robustness against overfitting and noise, making it well-suited for complex prediction tasks such as football player price estimation.

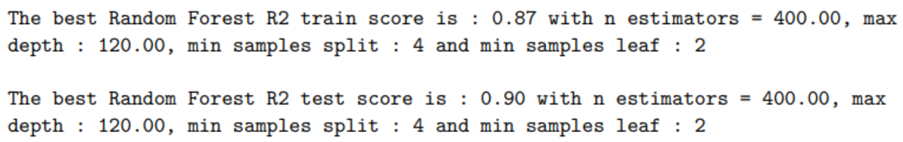


```
RFmodel = RandomForestRegressor()


param = {'n_estimators' : [400,450,480],
         'max_depth' : [100,120,140],
         'min_samples_split':[4],
         'min_samples_leaf':[2],
         'bootstrap' : [True]
        }

gridSearch_RandomForest=GridSearchCV(RFmodel,param,scoring='r2',cv=3)
gridSearch_RandomForest.fit(X_train,y_train)


best_randomForest=gridSearch_RandomForest.best_estimator_
bestRandomForest_testScore=best_randomForest.score(X_test,y_test)
```



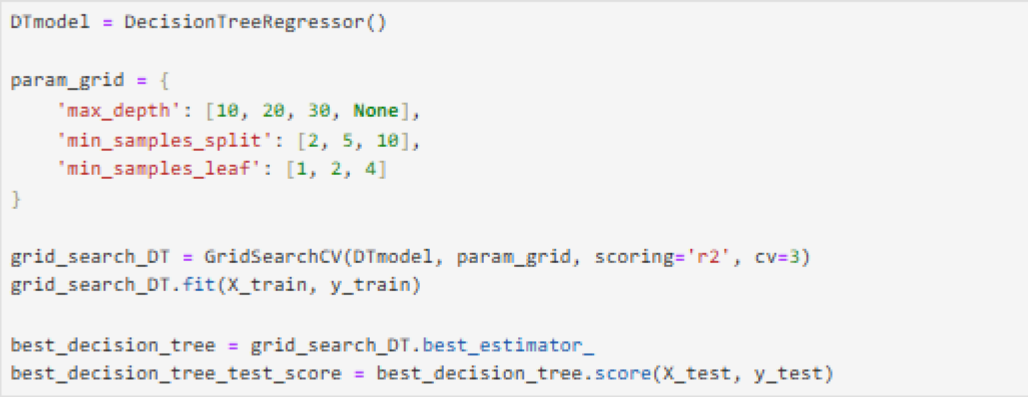
```
The best Random Forest R2 train score is : 0.87 with n estimators = 400.00, max
depth : 120.00, min samples split : 4 and min samples leaf : 2

The best Random Forest R2 test score is : 0.90 with n estimators = 400.00, max
depth : 120.00, min samples split : 4 and min samples leaf : 2
```

The accuracy of the Random Forest Regressor is best with n estimators = 400.00, max depth : 120.00, min samples split : 4 and min samples leaf : 2 at **90%**

3. **Decision Tree Regressor**

The Decision Tree Regressor was selected for its simplicity and interpretability, as well as its capability to handle both numerical and categorical data. This model partitions the feature space into hierarchical structures, allowing for intuitive visualization and understanding of the decision-making process. However, Decision Trees may be prone to overfitting, particularly with noisy data.

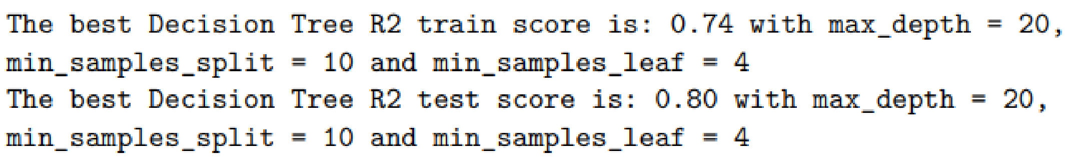


```
DTmodel = DecisionTreeRegressor()

param_grid = {
    'max_depth': [10, 20, 30, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

grid_search_DT = GridSearchCV(DTmodel, param_grid, scoring='r2', cv=3)
grid_search_DT.fit(X_train, y_train)

best_decision_tree = grid_search_DT.best_estimator_
best_decision_tree_test_score = best_decision_tree.score(X_test, y_test)
```



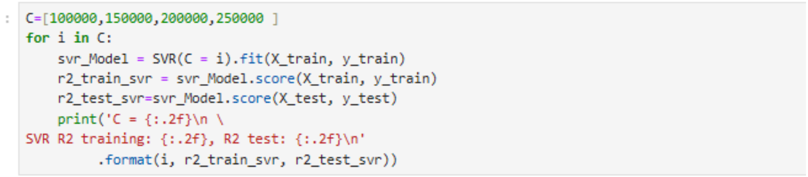
```
The best Decision Tree R2 train score is: 0.74 with max_depth = 20,
min_samples_split = 10 and min_samples_leaf = 4
The best Decision Tree R2 test score is: 0.80 with max_depth = 20,
min_samples_split = 10 and min_samples_leaf = 4
```

The accuracy of the decision tree is best with max_depth= 20, min_samples_split = 10 and min_samples_leaf = 4 at **80%**

4. **Support Vector Regression**

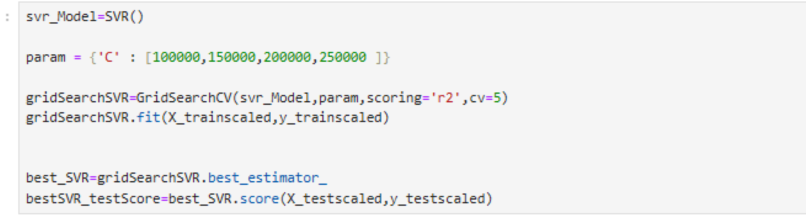
Support Vector Regression (SVR) was utilized for its ability to handle non-linear relationships while mitigating the risk of overfitting. SVR works by mapping the input space into a higher-dimensional feature space, where a linear regression model is applied. This allows SVR to capture complex patterns in the data while maintaining generalization performance.

(without scaling)



```
C=[100000,150000,200000,250000 ]
for i in C:
    svr_Model = SVR(C = i).fit(X_train, y_train)
    r2_train_svr = svr_Model.score(X_train, y_train)
    r2_test_svr=svr_Model.score(X_test, y_test)
    print('C = {:.2f}\n \
SVR R2 training: {:.2f}, R2 test: {:.2f}\n'
          .format(i, r2_train_svr, r2_test_svr))
```

(with scaling)

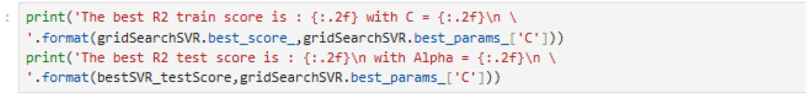


```
svr_Model=SVR()

param = {'C' : [100000,150000,200000,250000 ]}

gridSearchSVR=GridSearchCV(svr_Model,param,scoring='r2',cv=5)
gridSearchSVR.fit(X_trainscaled,y_trainscaled)


best_SVR=gridSearchSVR.best_estimator_
bestSVR_testScore=best_SVR.score(X_testscaled,y_testscaled)
```

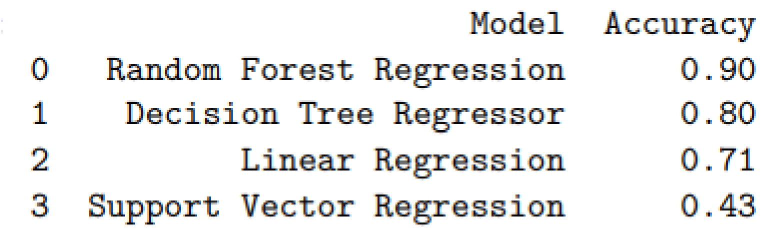


```
print('The best R2 train score is : {:.2f} with C = {:.2f}\n \
'.format(gridSearchSVR.best_score_,gridSearchSVR.best_params_['C']))
print('The best R2 test score is : {:.2f}\n with Alpha = {:.2f}\n \
'.format(bestSVR_testScore,gridSearchSVR.best_params_['C']))
```

The accuracy of the support vector regression model is better without scaling at 43%

PERFORMANCE EVALUATION

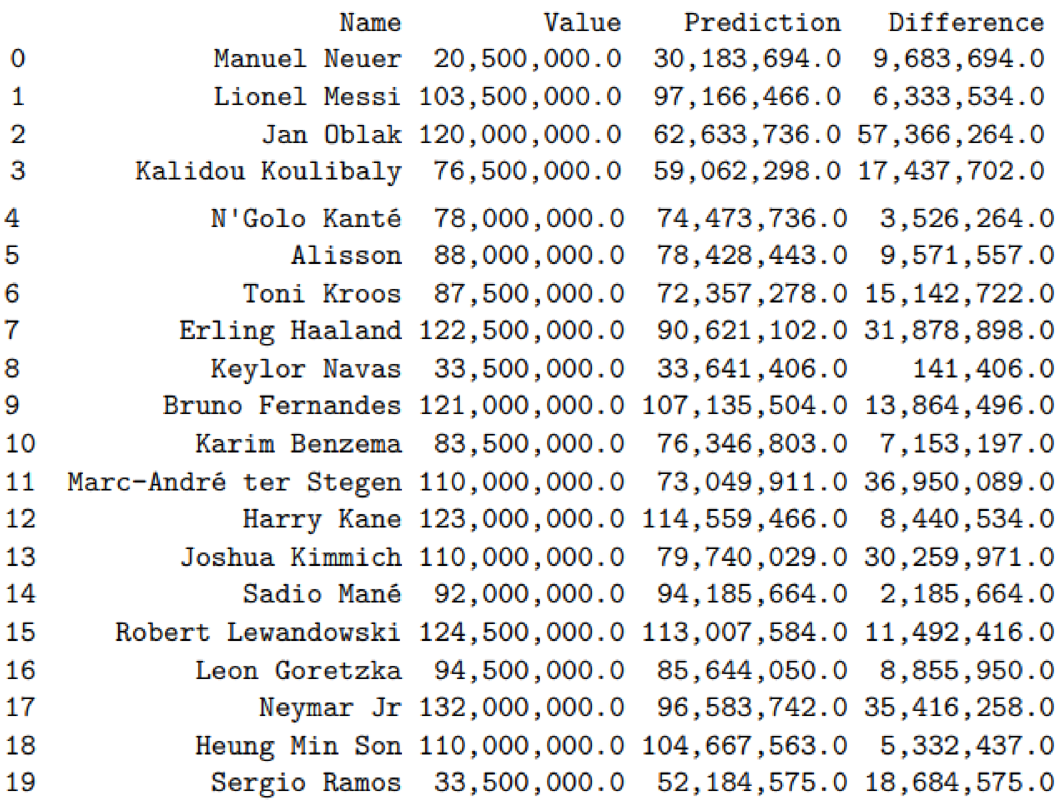
This performance evaluation examines the accuracy of four regression models on a specific dataset. We'll compare Random Forest Regression, Decision Tree Regressor, Linear Regression, and Support Vector Regression to determine the most effective model for this task.



```
                       Model  Accuracy
0   Random Forest Regression      0.90
1    Decision Tree Regressor      0.80
2          Linear Regression      0.71
3  Support Vector Regression      0.43
```

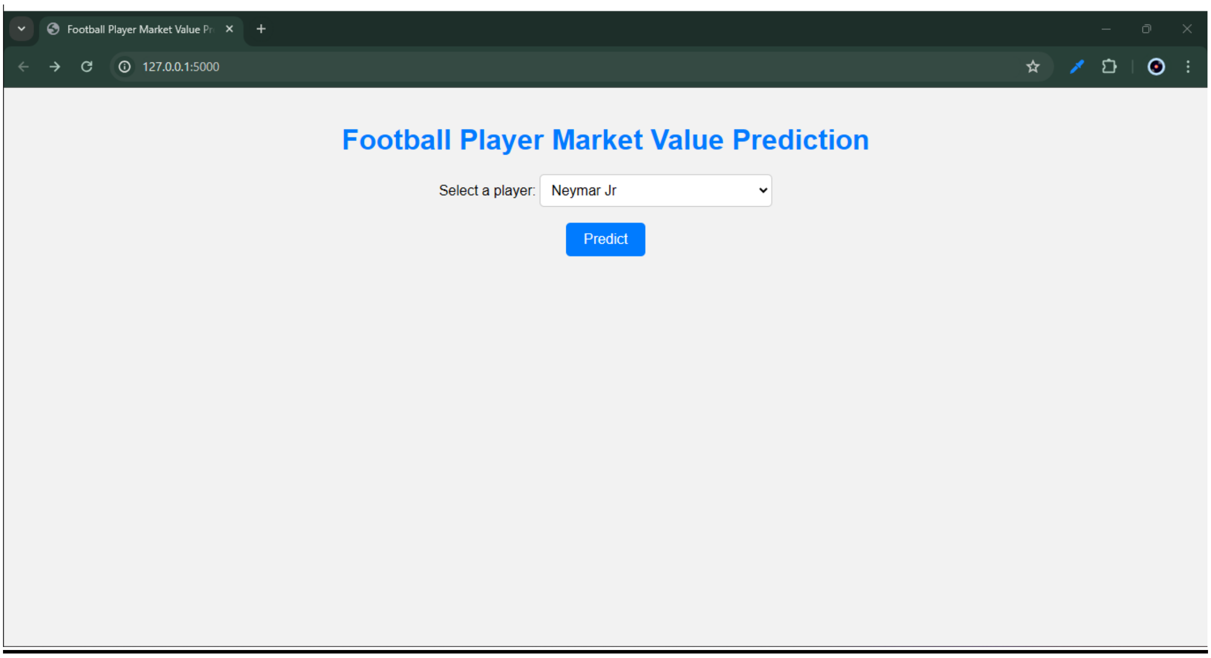
We can clearly observe that **random forest regression** gives us the highest accuracy. the combination of ensemble learning, ability to handle non-linear relationships, robustness to noise and outliers, and feature importance analysis makes Random Forest Regression a powerful and effective algorithm for predicting football player prices

We have further tested this accuracy on the first 20 players in the dataset and compared them with the actual value.



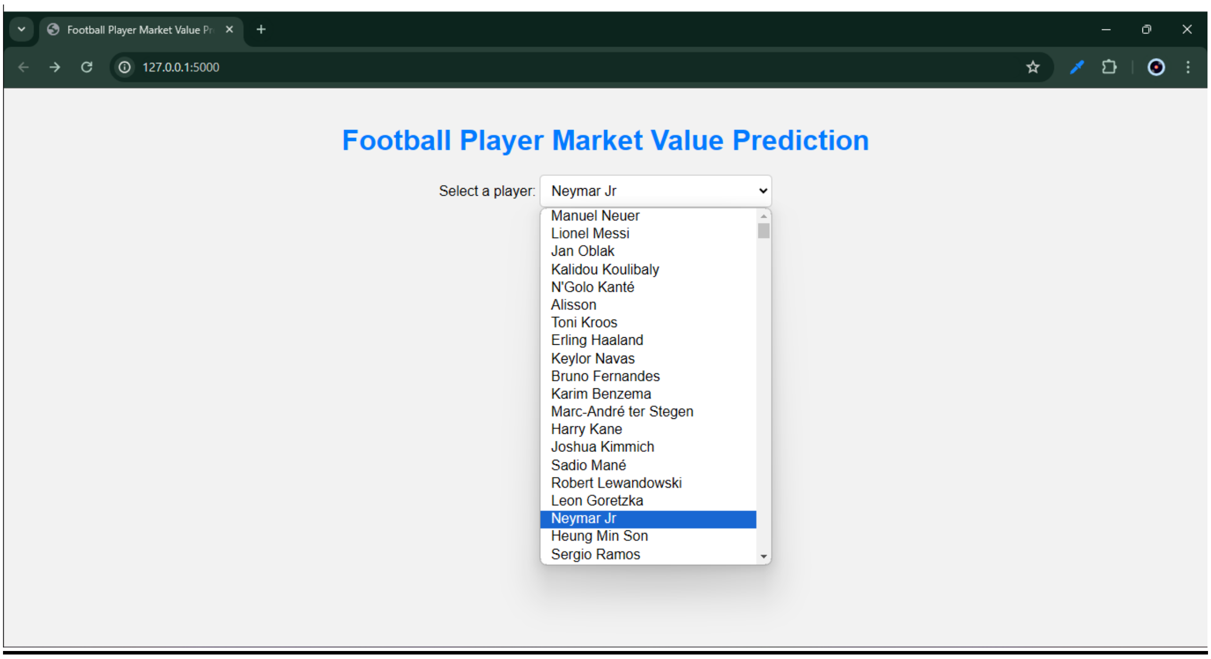
```
                 Name         Value    Prediction   Difference
0        Manuel Neuer  20,500,000.0  30,183,694.0  9,683,694.0
1        Lionel Messi 103,500,000.0  97,166,466.0  6,333,534.0
2          Jan Oblak 120,000,000.0  62,633,736.0 57,366,264.0
3    Kalidou Koulibaly  76,500,000.0  59,062,298.0 17,437,702.0
4         N'Golo Kanté  78,000,000.0  74,473,736.0  3,526,264.0
5              Alisson  88,000,000.0  78,428,443.0  9,571,557.0
6           Toni Kroos  87,500,000.0  72,357,278.0 15,142,722.0
7       Erling Haaland 122,500,000.0  90,621,102.0 31,878,898.0
8         Keylor Navas  33,500,000.0  33,641,406.0    141,406.0
9      Bruno Fernandes 121,000,000.0 107,135,504.0 13,864,496.0
10       Karim Benzema  83,500,000.0  76,346,803.0  7,153,197.0
11 Marc-André ter Stegen 110,000,000.0  73,049,911.0 36,950,089.0
12          Harry Kane 123,000,000.0 114,559,466.0  8,440,534.0
13      Joshua Kimmich 110,000,000.0  79,740,029.0 30,259,971.0
14          Sadio Mané  92,000,000.0  94,185,664.0  2,185,664.0
15  Robert Lewandowski 124,500,000.0 113,007,584.0 11,492,416.0
16        Leon Goretzka  94,500,000.0  85,644,050.0  8,855,950.0
17           Neymar Jr 132,000,000.0  96,583,742.0 35,416,258.0
18       Heung Min Son 110,000,000.0 104,667,563.0  5,332,437.0
19        Sergio Ramos  33,500,000.0  52,184,575.0 18,684,575.0
```

FLASK DEPLOYMENT



Football Player Market Value Prediction

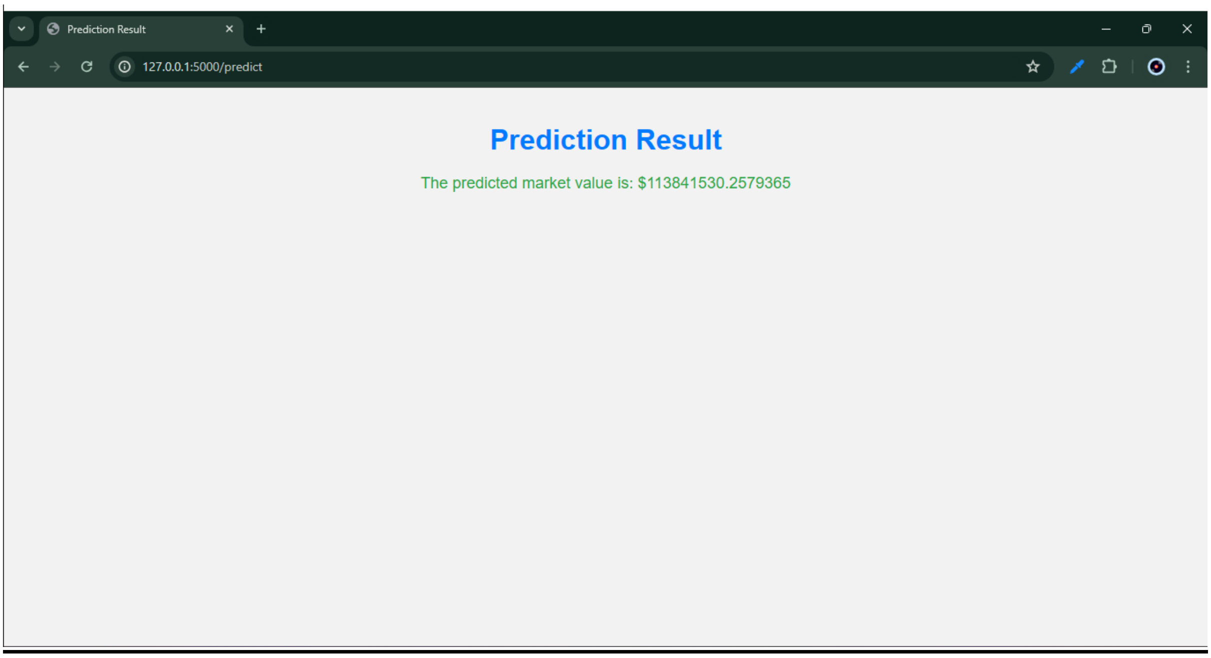
Select a player: Neymar Jr



Football Player Market Value Prediction

Select a player: Neymar Jr

Manuel Neuer
Lionel Messi
Jan Oblak
Kalidou Koulibaly
N'Golo Kanté
Alisson
Toni Kroos
Erling Haaland
Keylor Navas
Bruno Fernandes
Karim Benzema
Marc-André ter Stegen
Harry Kane
Joshua Kimmich
Sadio Mané
Robert Lewandowski
Leon Goretzka
Neymar Jr
Heung Min Son
Sergio Ramos

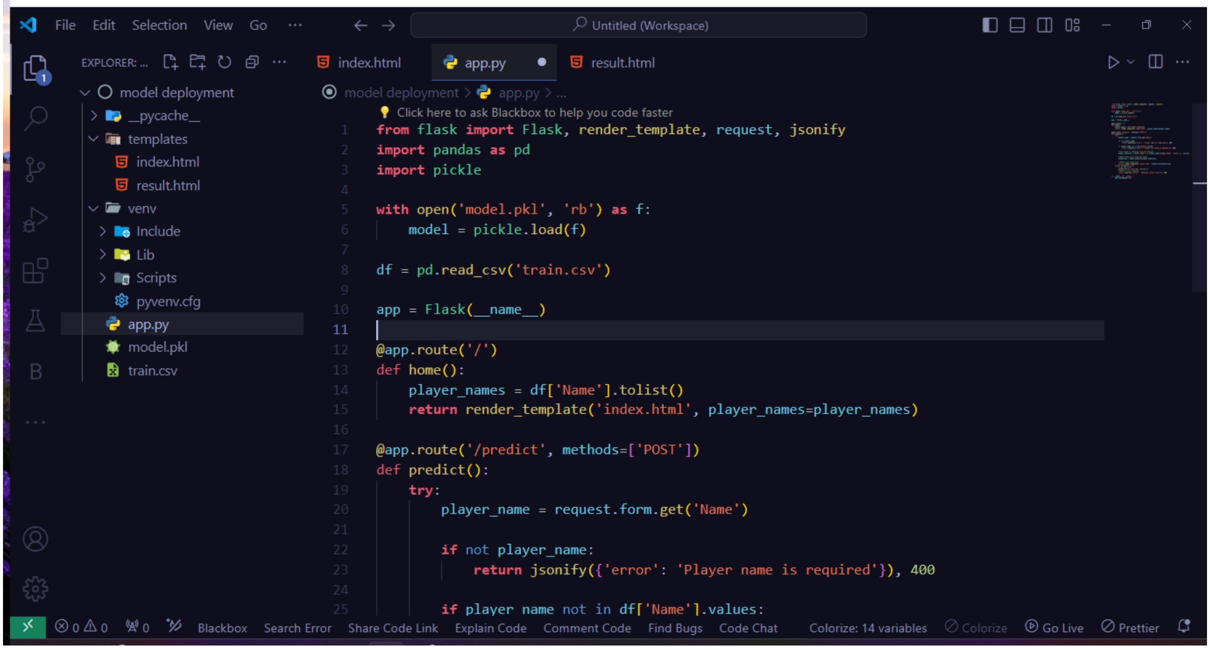


Prediction Result

The predicted market value is: $113841530.2579365

I have deployed the machine learning model using flask and created two pages: one rendering a dropdown which contains the list of all 18000 players. Once you select a player click the predict the 'Predict' Button, it will redirect to the predict route and show the predicted value.

Here is the file and folder structure of my flask deployment.


File Edit Selection View Go ...
Untitled (Workspace)
EXPLORER: ...
model deployment
__pycache__
templates
index.html
result.html
venv
Include
Lib
Scripts
pyvenv.cfg
app.py
model.pkl
train.csv
index.html
app.py
result.html
model deployment > app.py > ...
Click here to ask Blackbox to help you code faster
Blackbox Search Error Share Code Link Explain Code Comment Code Find Bugs Code Chat Colorize: 14 variables Colorize Go Live Prettier

```python
from flask import Flask, render_template, request, jsonify
import pandas as pd
import pickle

with open('model.pkl', 'rb') as f:
    model = pickle.load(f)

df = pd.read_csv('train.csv')

app = Flask(__name__)

@app.route('/')
def home():
    player_names = df['Name'].tolist()
    return render_template('index.html', player_names=player_names)

@app.route('/predict', methods=['POST'])
def predict():
    try:
        player_name = request.form.get('Name')

        if not player_name:
            return jsonify({'error': 'Player name is required'}), 400

        if player_name not in df['Name'].values:
```