FIFA VS REALITY

Soyeon Lim
Akinkunmi Adesina
Shubham Dutt

Table of Contents

1st Approach

Questions

- Q1: Does the EA FIFA player profile match to the performance in real life?
- Q2: Can FIFA be used as a digital twin of real-life football?

1st Approach

- Linear Regression
- Support Vector Regression
- Divide the players into 3 groups based on their positions
  - Defensive players
  - Midfielders
  - Offensive players

1st Approach

EDA

Name	Name	Rating	Fifa Ability Overall	Position	Apps	Minutes played
game"	Dribbles per game	Fouled per game	Offsides per game	Dispossessed per game	Bad control per game	Key passes per game
manipulation		string	string	string	0	0
manipulation		string	string	string	0	0
Lionel Messi	Lionel Messi Barcelona, 32, AM(CR),FW	8.48	94	AM(CR), FW	29	2710
Cristiano Ronaldo	Cristiano Ronaldo Juventus, 34, M(L),FW	7.68	94	M(L), FW	30	2689
Neymar	Neymar Paris Saint-Germain, 27, AM(CLR),FW	8.26	92	AM(CLR), FW	16	1444
Luis Suárez	Luis Suárez Barcelona, 32, AM(CLR),FW	7.57	91	AM(CLR), FW	31	2830
Luka Modric	Luka Modric Real Madrid, 33, M(C)	7.03	91	M(C)	31	2618
Sergio Ramos	Sergio Ramos Real Madrid, 33, D(CR)	6.91	91	D(CR)	28	2476
Leo Suárez	Leo Suárez Real Valladolid, 23, FW	6.25	91	FW	7	520
Eden Hazard	Eden Hazard Chelsea, 28, M(CLR),FW	7.81	91	M(CLR), FW	32	2926
Kevin De Bruyne	Kevin De Bruyne Manchester City, 28, M(CLR),FW	7.05	91	M(CLR), FW	11	978
Robert Lewandowski	Robert Lewandowski Bayern Munich, 31, FW	7.65	90	FW	33	2959
Toni Kroos	Toni Kroos Real Madrid, 29, M(C)	7.09	90	M(C)	26	2227
Diego Godín	Diego Godín Atletico Madrid, 33, D(C)	6.98	90	D(C)	28	2508
David Silva	David Silva Manchester City, 33, M(CLR)	7.26	90	M(CLR)	28	2412
Antoine Griezmann	Antoine Griezmann Atletico Madrid, 28, AM(CLR),FW	7.29	89	AM(CLR), FW	37	3204
Sergio Busquets	Sergio Busquets Barcelona, 31, DMC	7	89	DMC	30	2720
Edinson Cavani	Edinson Cavani Paris Saint-Germain, 32, AM(LR),FW	7.44	89	AM(LR), FW	20	1676
Sergio Agüero	Sergio Agüero Manchester City, 31, AM(CL),FW	7.53	89	AM(CL), FW	31	2480
Harry Kane	Harry Kane Tottenham, 26, AM(C),FW	7.38	89	AM(C), FW	27	2427
NGolo Kanté	N'Golo Kanté Chelsea, 28, DMC	6.93	89	DMC	36	3096
Paulo Dybala	Paulo Dybala Juventus, 25, AM(CR),FW	7.08	89	AM(CR), FW	24	2137
Giorgio Chiellini	Giorgio Chiellini Juventus, 35, D(C)	6.98	89	D(C)	22	1990
James Rodríguez	James Rodríguez Bayern Munich, 28, AM(CLR)	7.24	88	AM(CLR)	13	1143
Mats Hummels	Mats Hummels Bayern Munich, 30, D(C)	7.17	88	D(C)	20	1776
Casemiro	Casemiro Real Madrid, 27, DMC	7.15	88	DMC	27	2316
Philippe Coutinho	Philippe Coutinho Barcelona, 27, M(CLR)	6.93	88	M(CLR)	22	2023
Gareth Bale	Gareth Bale Real Madrid, 30, M(CLR),FW	6.88	88	M(CLR), FW	21	1794

1^{st} Approach

EDA

Rating	Fifa Ability Overall	Position	Apps
Minutes played	Assists	Yel	Red
Aerials Won per game	Man of the match	Tackles	Interceptions per game
Fouls	Offside won per game	Clearances per game	Dribbled past per game
Outfielder Block Per Game	OwnG	Goals	Shots per game
Dribbles per game	Fouled per game	Offsides per game	Dispossessed per game
Bad control per game	Passes per game	Key passes per game	Pass success percentage
Crosses	Long balls per game	Through balls per game	

Data Preparation

- Remove duplicates (2137 ➡ 1956)
- Clean ‘Position’ column
- Specify relevant attributes for each position
- Separate df into 3 different data frames (DEF, MID, OFF) with relevant attributes for each position

```python
pos = df['Position'].unique()

pos
```

✓ 0.1s

```
array(['AM(CR), FW', 'M(L), FW', 'AM(CLR), FW', 'M(C)', 'D(CR)', 'FW',
       'M(CLR), FW', 'D(C)', 'M(CLR)', 'DMC', 'AM(LR), FW', 'AM(CL), FW',
       'AM(C), FW', 'AM(CLR)', 'Forward, Forward', 'D(C), D(C)',
       'M(LR), FW', 'D(L)', 'D(CL)', 'AM(R), FW', 'M(C), FW', 'DMC, DMC',
       'D(R), DMC', 'D(CLR), M(R)', 'D(L), M(L)', 'M(R)', 'D(CR), DMC',
       'D(CL), M(C)', 'D(CL), M(CLR)', 'AM(C), AM(C)', 'M(CL)', 'D(R)',
       'D(CR), M(R)', 'M(CL), FW', 'M(LR)', 'D(R), M(CLR), FW',
       'AM(L), FW', 'M(CR)', 'D(R), M(CR)', 'D(C), M(C)', 'AM(L)',
       'D(R), M(C)', 'AM(CLR), AM(CLR)', 'D(CL), M(L)', 'D(C), DMC',
       'D(L), M(CLR)', 'D(L), M(CL)', 'D(R), M(R)', 'D(LR), M(R)',
       'D(R), M(CLR)', 'M(CR), FW', 'D(CR), M(C)', 'D(LR), M(CLR)',
       'D(L), DMC, M(L)', 'AM(CL)', 'D(LR)', 'AM(C)', 'M(C), M(C)',
       'M(R), FW', 'AM(R)', 'D(CLR), DMC', 'DMC, M(L)', 'D(CLR)',
       'D(LR), M(CR)', 'D(LR), M(LR)', 'D(CL), DMC', 'D(CLR), M(L)',
       'AM(LR)', 'Forward', 'D(CR), D(CR)', 'D(CLR), DMC, M(LR)',
       'D(CR), M(CR)', 'D(R), M(L)', 'D(R), M(LR)', 'D(LR), M(CL)',
       'M(CR), M(CR)', 'M(CLR), M(CLR)', 'AM(CR)', 'Midfielder',
       'D(LR), DMC, M(R)', 'D(R), DMC, M(R)', 'D(CLR), M(LR)', 'M(L)',
       'D(CL), D(CL)', 'Defender', 'AM(R), AM(R)', 'D(L), M(LR)',
       'AM(CL), AM(CL)', 'D(LR), M(L)', 'FW, FW', 'D(R), D(R)',
       'D(CLR), DMC, M(R)', 'DMC, M(R)', 'D(LR), D(LR)', 'AM(CR), AM(CR)',
       'D(L), M(CR)', 'D(R), M(CR), FW', 'D(R), M(L), FW', 'M(L), M(L)',
       'AM(LR), AM(LR)', 'D(L), D(L)', 'Midfielder, Midfielder',
       'D(C), M(L)'], dtype=object)
```

```python
def positions(pos: str):
    if 'M' in pos and 'F' in pos:
        return 'OFF'

    elif 'M' in pos and 'D' in pos:
        return 'DEF'

    elif 'M' in pos:
        return 'MID'

    elif 'F' in pos:
        return 'OFF'

    elif 'D' in pos:
        return 'DEF'


df['pos'] = df['Position'].apply(positions)
```

1st Approach

Data Preparation

Rating	Fifa Ability Overall	Position	Apps
Minutes played	Assists	Yel	Red
Aerials Won per game	Man of the match	Tackles	Interceptions per game
Fouls	Offside won per game	Clearances per game	Dribbled past per game
Outfielder Block Per Game	OwnG	Goals	Shots per game
Dribbles per game	Fouled per game	Offsides per game	Dispossessed per game
Bad control per game	Passes per game	Key passes per game	Pass success percentage
Crosses	Long balls per game	Through balls per game	

Data Preparation

Defensive Players

Rating	Fifa Ability Overall	Position	Apps
Minutes played	Assists	Yel	Red
Aerials Won per game	Man of the match	Tackles	Interceptions per game
Fouls	Offside won per game	Clearances per game	Dribbled past per game
Outfielder Block Per Game	OwnG	Goals	Shots per game
Dribbles per game	Fouled per game	Offsides per game	Dispossessed per game
Bad control per game	Passes per game	Key passes per game	Pass success percentage
Crosses	Long balls per game	Through balls per game	

1st Approach

Data Preparation

Midfielders

Rating	Fifa Ability Overall	Position	Apps
Minutes played	Assists	Yel	Red
Aerials Won per game	Man of the match	Tackles	Interceptions per game
Fouls	Offside won per game	Clearances per game	Dribbled past per game
Outfielder Block Per Game	OwnG	Goals	Shots per game
Dribbles per game	Fouled per game	Offsides per game	Dispossessed per game
Bad control per game	Passes per game	Key passes per game	Pass success percentage
Crosses	Long balls per game	Through balls per game	

1st Approach

Data Preparation

Offensive Players

Rating	Fifa Ability Overall	Position	Apps
Minutes played	Assists	Yel	Red
Aerials Won per game	Man of the match	Tackles	Interceptions per game
Fouls	Offside won per game	Clearances per game	Dribbled past per game
Outfielder Block Per Game	OwnG	Goals	Shots per game
Dribbles per game	Fouled per game	Offsides per game	Dispossessed per game
Bad control per game	Passes per game	Key passes per game	Pass success percentage
Crosses	Long balls per game	Through balls per game	

```python
df_def
```

✓ 0.1s Python

	Name	Rating	Apps	Minutes played	Assists	Aerials Won per game	Man of the match	Tackles	Interceptions per game	Fouls	Offside won per game	Clearances per game
7	Sergio Ramos	6.91	28	2476	1	2.2	2	1.5	1.3	1.0	1.0	3.0
13	Diego Godín	6.98	28	2508	1	3.0	1	1.7	1.5	1.1	0.2	4.5
16	Sergio Busquets	7.00	30	2720	1	1.5	0	2.6	1.5	1.1	0.0	0.6
20	NGolo Kanté	6.93	36	3096	4	0.8	1	2.1	1.2	1.0	0.0	0.7
22	Giorgio Chiellini	6.98	22	1990	1	2.1	0	1.0	1.2	0.6	0.2	3.7
...	...	...	...	...	...	...	...	...	...	...	...	...
2132	Yoel Armougom	6.36	20	1850	1	0.9	0	1.1	0.7	0.7	0.1	2.0
2134	Ben Wilmot	6.07	2	198	0	0.2	0	0.0	0.4	0.4	0.2	2.2
2135	Keven Schlotterbeck	6.58	8	772	0	1.4	0	2.4	1.4	0.6	0.4	2.8
2136	Nico Schlotterbeck	6.47	2	226	0	1.3	0	1.5	1.3	1.0	0.0	2.5
2137	Chima Okoroji	6.29	1	104	0	0.5	0	1.5	0.5	1.0	0.0	0.5

974 rows × 15 columns

```python
df_mid
```

✓ 0.1s Python

	Name	Rating	Apps	Minutes played	Assists	Aerials Won per game	Man of the match	Tackles	Interceptions per game	Fouls	Dribbled past per game	Dribbles per game	Fou ga
6	Luka Modric	7.03	31	2618	6	0.4	1	1.4	0.9	0.8	1.0	1.6	
12	Toni Kroos	7.09	26	2227	4	0.8	2	1.8	0.5	0.8	1.5	0.5	
14	David Silva	7.26	28	2412	8	0.6	1	0.9	0.7	0.8	0.6	0.9	
23	James Rodríguez	7.24	13	1143	3	0.2	3	0.6	0.2	0.4	0.7	0.8	
26	Philippe Coutinho	6.93	22	2023	2	0.2	0	0.5	0.4	0.6	0.4	1.5	
...	...	...	...	...	...	...	...	...	...	...	...	...	
2125	Christoph Baumgartner	5.97	1	86	0	0.5	0	0.5	0.0	1.0	1.0	2.5	
2128	Callum Slattery	6.26	1	110	0	1.0	0	1.3	0.7	1.3	1.3	0.3	
2130	Stephane Omeonga	5.95	0	54	0	0.7	0	0.3	0.0	1.3	0.3	0.0	
2133	Matty Daly	5.96	0	73	0	0.0	0	0.0	0.0	0.0	0.0	0.0	
2138	Valery Fernández	6.37	10	885	1	0.5	0	1.2	0.5	0.5	0.4	1.1	

521 rows × 20 columns

```
df_off
```

✓ 0.1s Python

	Name	Rating	Apps	Minutes played	Assists	Aerials Won per game	Man of the match	Goals	Shots per game	Dribbles per game	Offsides per game	Dispossessed per game	Bad control per game
2	Lionel Messi	8.48	29	2710	13	0.2	17	36	5.0	3.9	0.5	2.3	1.9
3	Cristiano Ronaldo	7.68	30	2689	8	1.1	10	21	5.7	1.5	0.9	1.3	1.7
4	Neymar	8.26	16	1444	7	0.5	7	15	3.2	4.4	0.4	4.2	4.1
5	Luis Suárez	7.57	31	2830	6	0.7	5	21	3.4	1.1	0.8	1.6	2.3
8	Leo Suárez	6.25	7	520	0	0.3	0	2	0.8	0.3	0.3	0.6	1.4
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2106	Jonathan Burkardt	6.46	4	262	0	1.0	0	0	1.5	0.5	0.0	2.8	1.8
2110	Robert Beric	6.74	12	1172	1	1.3	1	9	1.6	0.2	0.2	0.7	1.4
2113	Boulaye Dia	6.56	8	746	0	1.1	0	3	1.1	0.9	0.1	0.8	1.6
2121	Dusan Vlahovic	5.96	1	152	0	0.2	0	0	1.0	0.1	0.0	0.5	1.1
2131	Victor Mollejo	6.02	0	63	0	0.5	0	0	0.5	0.0	0.0	1.0	0.3

461 rows × 17 columns

Machine Learning Models

Classification?

Regression?

1st Approach

Machine Learning Models

1. Linear Regression
2. Support Vector Regression

Linear Regression

Dependent variable: 'Rating'

```python
from sklearn.linear_model import LinearRegression

lr_def = LinearRegression()

lr_def.fit(X_def_train, Y_def_train)

Y_def_pred_lr = lr_def.predict(X_def_test)
```

✓ 0.1s Python

```python
lr_mid = LinearRegression()

lr_mid.fit(X_mid_train, Y_mid_train)

Y_mid_pred_lr = lr_mid.predict(X_mid_test)
```

✓ 0.0s Python

```python
lr_off = LinearRegression()

lr_off.fit(X_off_train, Y_off_train)

Y_off_pred_lr = lr_off.predict(X_off_test)
```

✓ 0.0s Python

Linear Regression

Dependent variable: 'FIFA Ability Score'

```
lr_def1 = LinearRegression()
lr_def1.fit(X_def1_train, Y_def1_train)
Y_def1_pred_lr = lr_def1.predict(X_def1_test)

lr_mid1 = LinearRegression()
lr_mid1.fit(X_mid1_train, Y_mid1_train)
Y_mid1_pred_lr = lr_mid1.predict(X_mid1_test)

lr_off1 = LinearRegression()
lr_off1.fit(X_off1_train, Y_off1_train)
Y_off1_pred_lr = lr_off1.predict(X_off1_test)
```

✓ 0.1s

Support Vector Regression

Dependent variable: 'Rating'

```python
from sklearn import svm

X_def_train, X_def_test, Y_def_train, Y_def_test = train_test_split(X_def, Y_def, test_size=0.2,
random_state=42)

X_mid_train, X_mid_test, Y_mid_train, Y_mid_test = train_test_split(X_mid, Y_mid, test_size=0.2,
random_state=42)

X_off_train, X_off_test, Y_off_train, Y_off_test = train_test_split(X_off, Y_off, test_size=0.2,
random_state=42)


svm_def = svm.SVR(kernel='rbf')

svm_def.fit(X_def_train, Y_def_train)

Y_def_pred_svm = svm_def.predict(X_def_test)


svm_mid = svm.SVR(kernel='rbf')

svm_mid.fit(X_mid_train, Y_mid_train)

Y_mid_pred_svm = svm_mid.predict(X_mid_test)


svm_off = svm.SVR(kernel='rbf')

svm_off.fit(X_off_train, Y_off_train)

Y_off_pred_svm = svm_off.predict(X_off_test)
```

✓ 0.1s

Python

Support Vector Regression

Dependent variable: 'FIFA Ability Score'

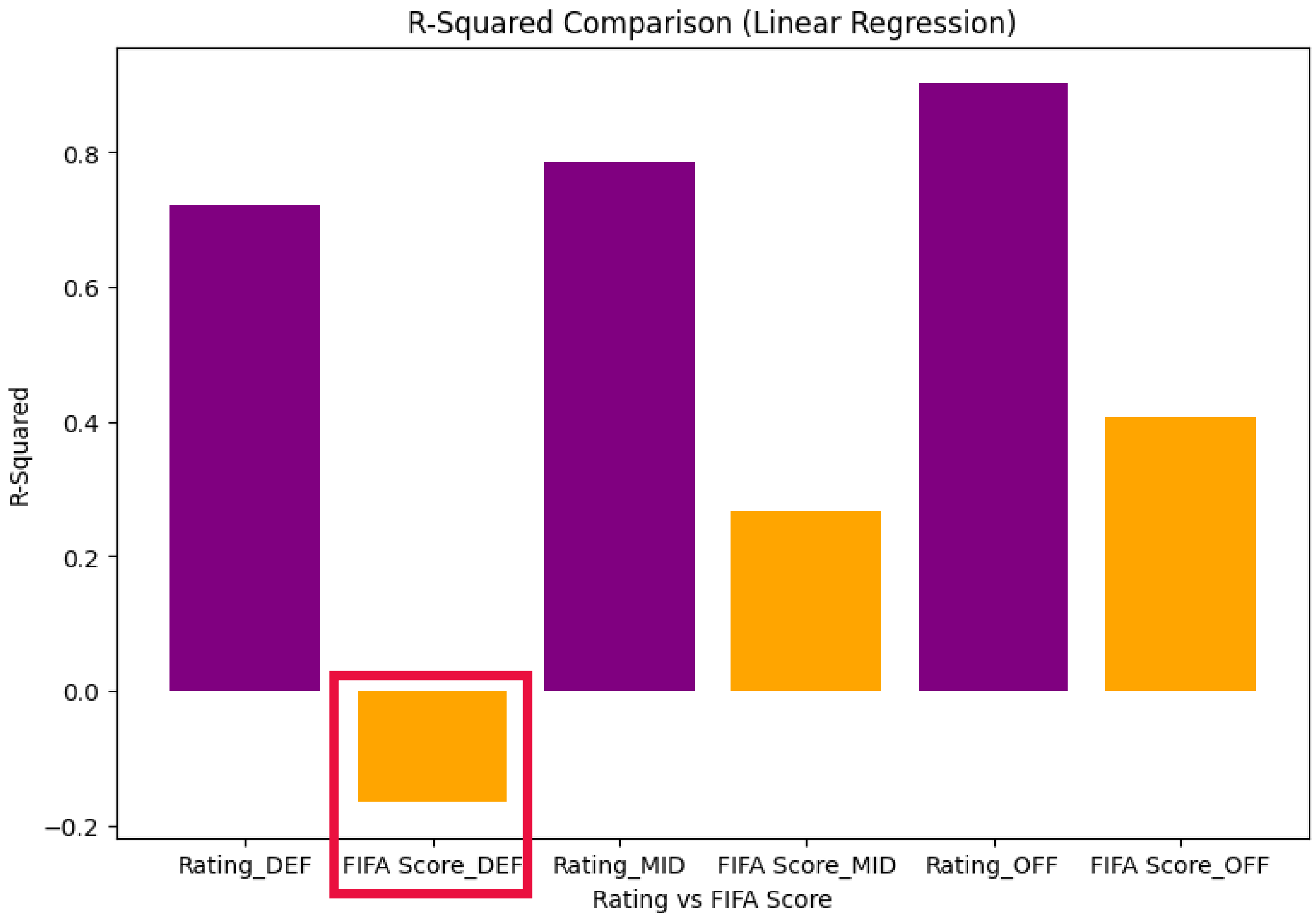
```
X_def1_train, X_def1_test, Y_def1_train, Y_def1_test = train_test_split(X_def1, Y_def1, test_size=0.2, random_state=42)

X_mid1_train, X_mid1_test, Y_mid1_train, Y_mid1_test = train_test_split(X_mid1, Y_mid1, test_size=0.2, random_state=42)

X_off1_train, X_off1_test, Y_off1_train, Y_off1_test = train_test_split(X_off1, Y_off1, test_size=0.2, random_state=42)


svm_def1 = svm.SVR(kernel='rbf')

svm_def1.fit(X_def1_train, Y_def1_train)

Y_def1_pred_svm = svm_def1.predict(X_def1_test)


svm_mid1 = svm.SVR(kernel='rbf')

svm_mid1.fit(X_mid1_train, Y_mid1_train)

Y_mid1_pred_svm = svm_mid1.predict(X_mid1_test)


svm_off1 = svm.SVR(kernel='rbf')

svm_off1.fit(X_off1_train, Y_off1_train)

Y_off1_pred_svm = svm_off1.predict(X_off1_test)
```

✓ 0.2s Python

1^{st} Approach

Linear Regression

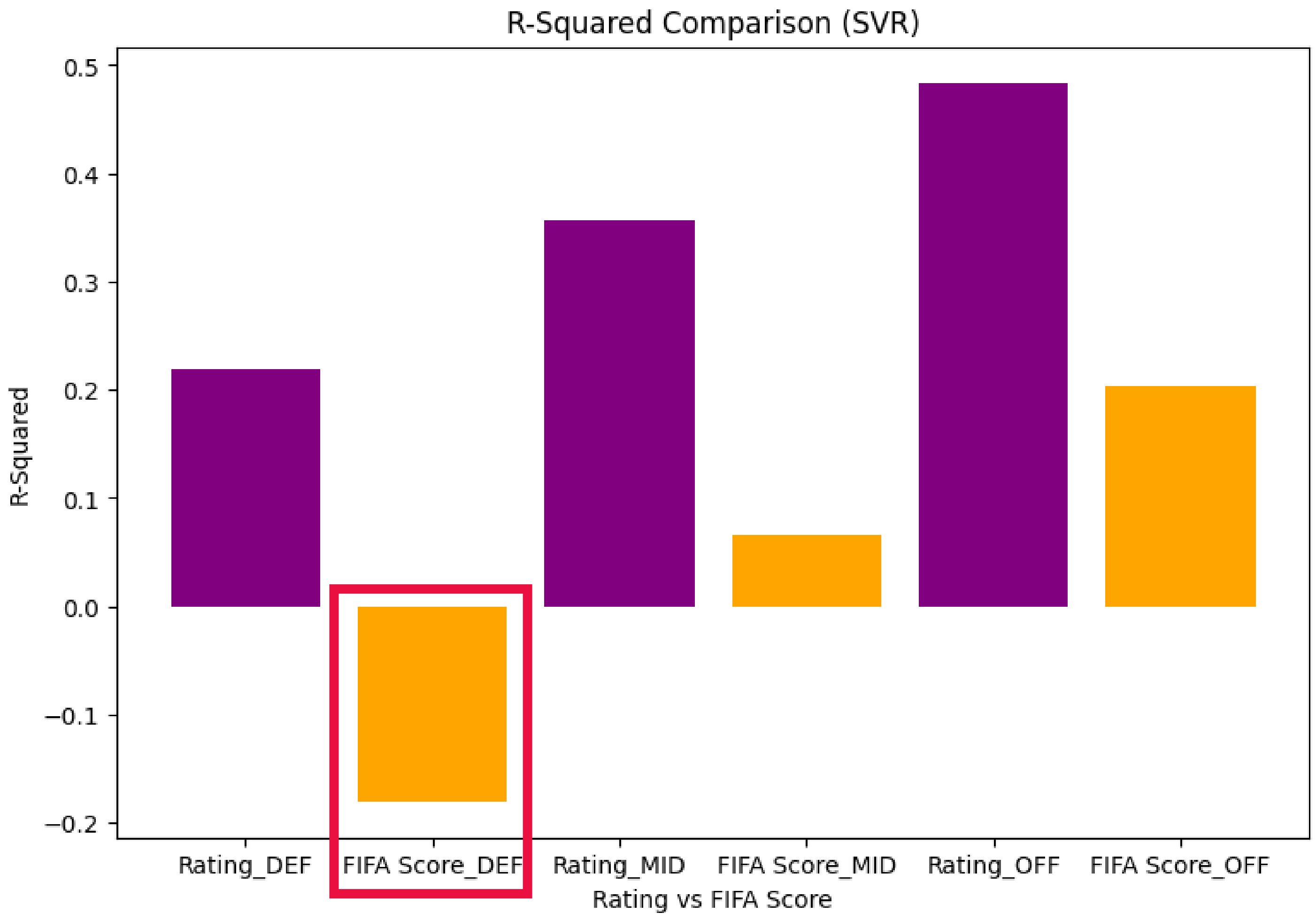
Evaluation



1st Approach

Support Vector Regression

Evaluation



1st Approach

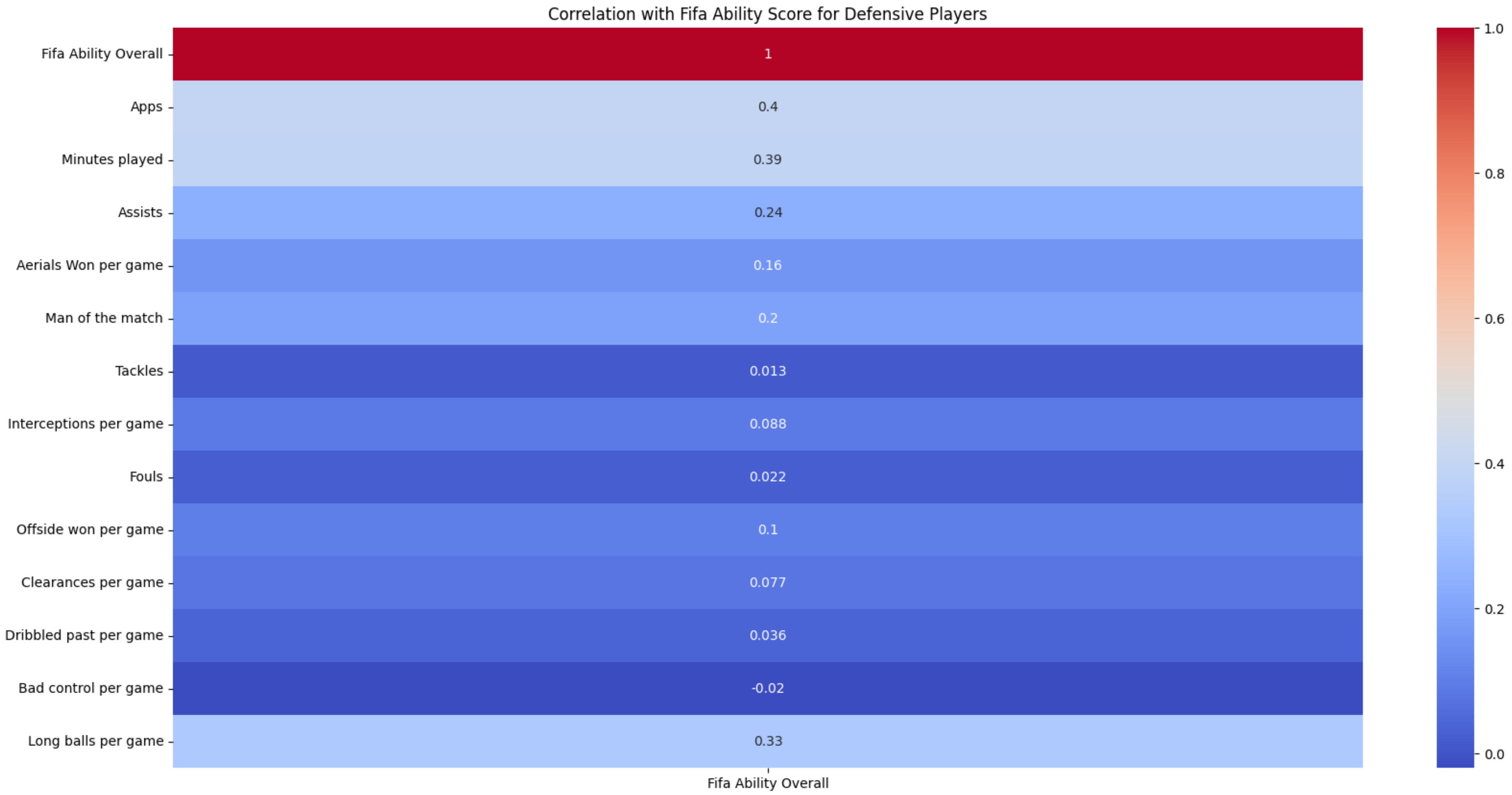
Visualization & Analysis

'Rating' - DEF

1st Approach

Visualization & Analysis

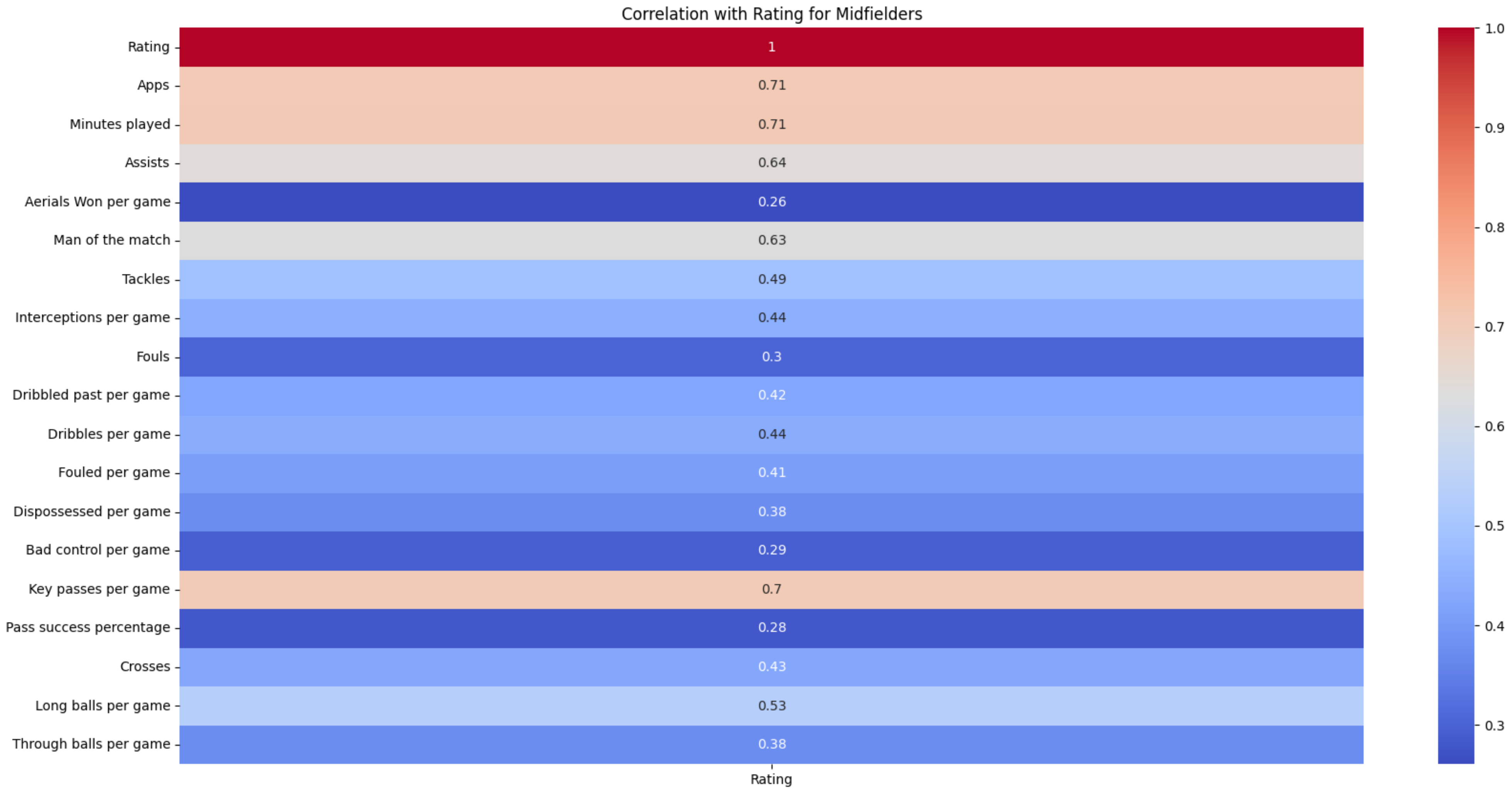
'FIFA Score' - DEF



1st Approach

'Rating' - MID

Visualization & Analysis



1st Approach

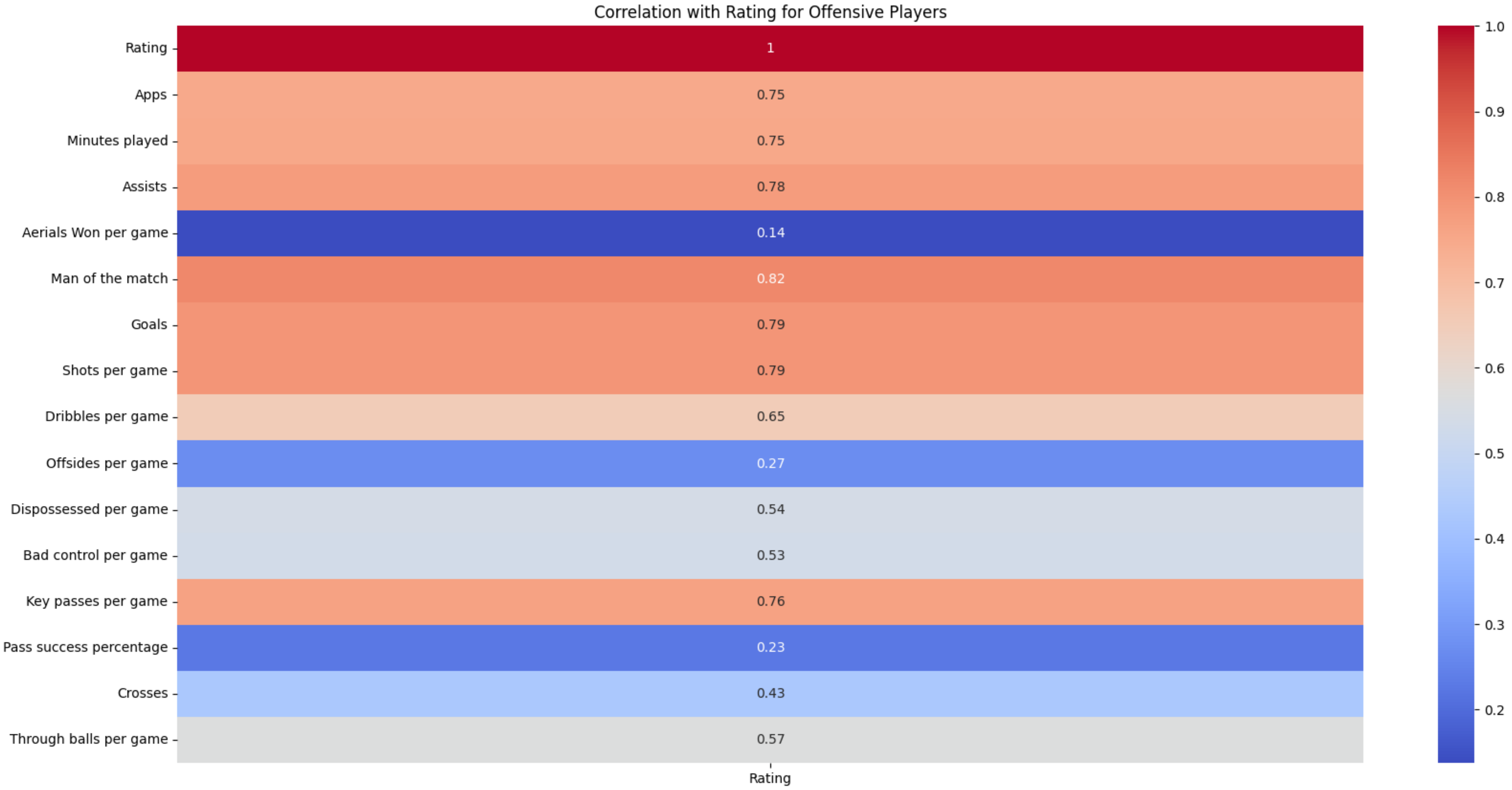
'FIFA Score' - MID

Visualization & Analysis

1st Approach

'Rating' - OFF

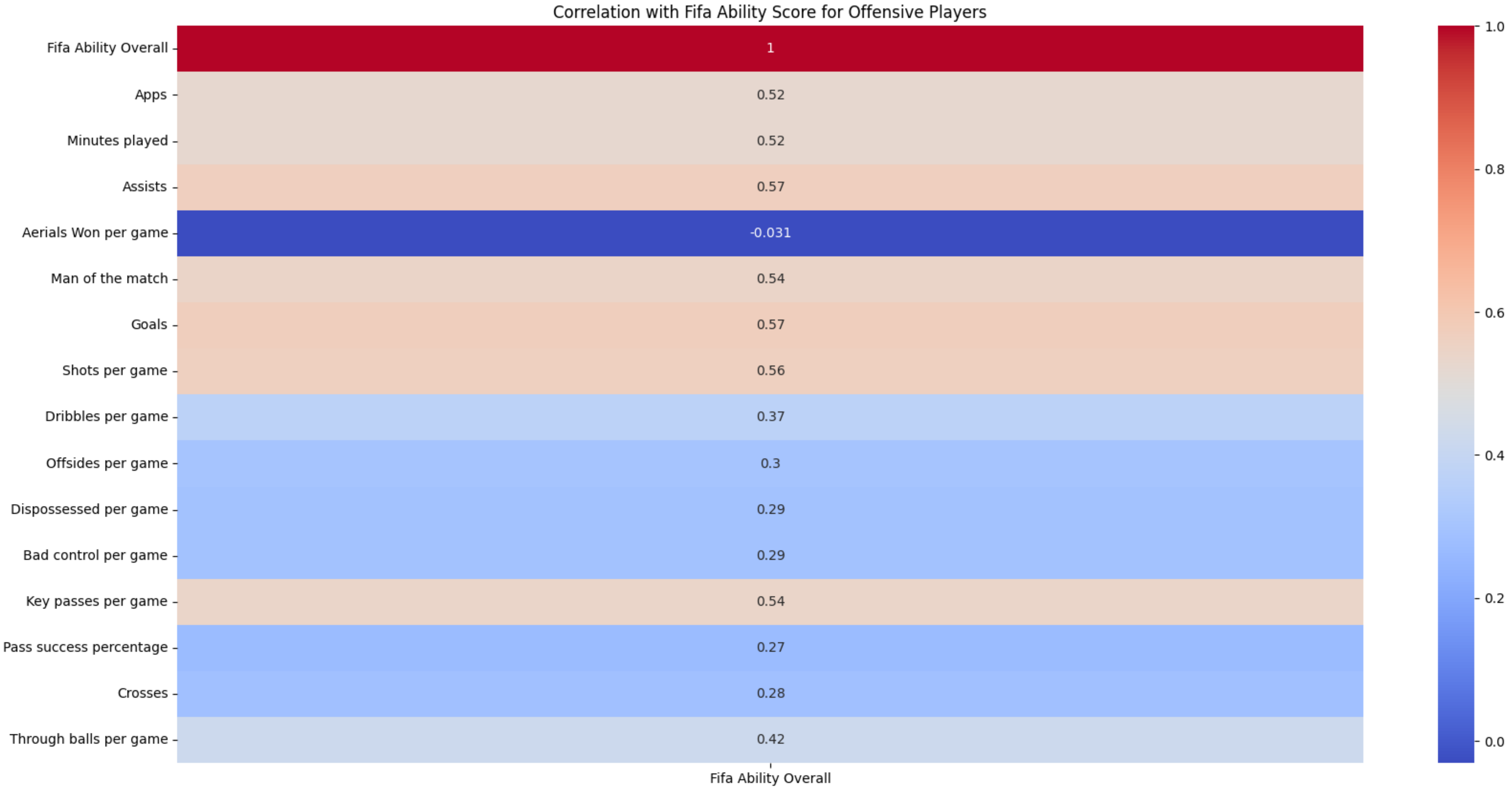
Visualization & Analysis



1st Approach

'FIFA Score' - OFF

Visualization & Analysis



1^{st} Approach

Findings

- There is not a strong correlation between the FIFA Scores and real-life stats of the player.
- The most important attributes

  for defensive players:

  Appearances, Minutes played, Man of the match

  for midfielders:

  Appearances, Minutes played, Key passes per game

  for offensive players:

  Man of the match, Goals, Shots per game

1st Approach

Findings

- There is not a strong correlation between the FIFA Scores and real-life stats of the player.
- The most important attributes

  for defensive players:

  Appearances, Minutes played, Man of the match

  for midfielders:

  Appearances, Minutes played, Key passes per game

  for offensive players:

  Man of the match, Goals, Shots per game

Findings

- There is not a strong correlation between the FIFA Scores and real-life stats of the player.
- The most important attributes

  for defensive players:

  Appearances, Minutes played, Man of the match

  for midfielders:

  Appearances, Minutes played, Key passes per game

  for offensive players:

  Man of the match, Goals, Shots per game

Conclusion

- Q1: Does the EA FIFA player profile match to the performance
  in real life?

- Q2: Can FIFA be used as a digital twin of real-life football?

A. "No."

2^{nd} Approach

Decision tree

Entropy (by Python)

Information gain (by Python)

Sort dependent and independent variables

Dividing sets into training and test sets

Feature scaling

Decision tree in regression, confusion matrix and general statistics

Visualization of the results

2nd Approach

EDA & Data Cleaning

2nd Approach

Machine Learning Models

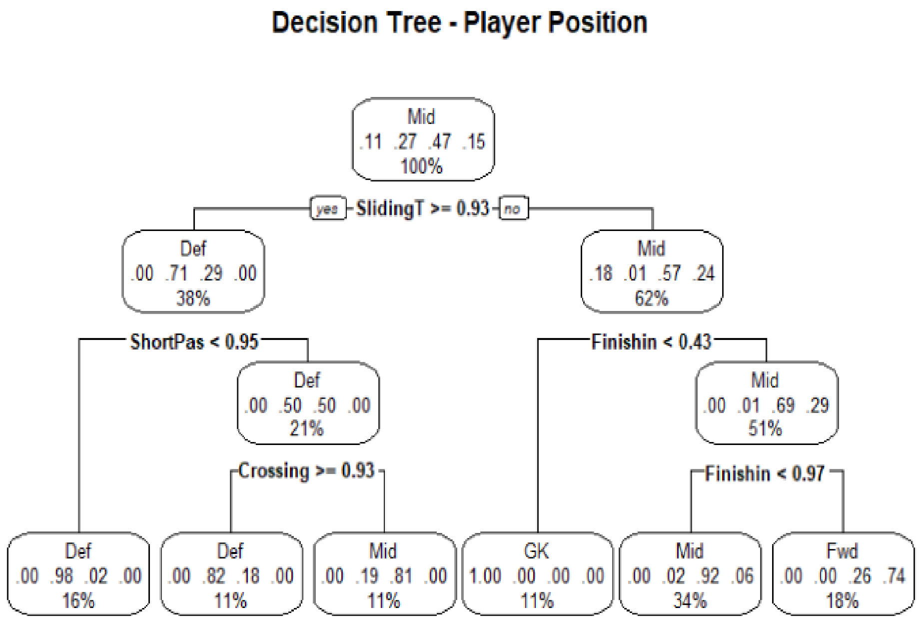
Decision Tree

Definition

Decision Tree is one of the most widely used machine learning algorithms due to its ease of interpretation.

However, we need to understand how it works and it can be used to make predictions

Sample



2^{nd} Approach

Decision Tree

Information Gain

- This is defined as a measure of how much information a feature provides about class, this also helps us to get the order of attributes in the nodes with the help of entropy

Entropy

- This is an information theory that measures the impurity or uncertainty of a group and observation
- This is necessary for our data set in order to be able to know which columns have the highest information gain

2nd Approach

Decision Tree

Entropy and Information Gain

Information gain and entropy calculation

We can calculate our information gain by looping through the columns

IG = 1 – Entropy

Entropy= -(p(0) * log(P(0)) + p(1) * log(P(1)))

Python result about IG and entropy

```
THIS FOLLOWING VALUES IS FOR THE FIFA WERTE TABLE
Variable FIFA has Entropy of 7.5663
Variable OPTA has Entropy of 7.6067
Variable Fifa Ovr has Entropy of 3.0829
Variable Crossing has Entropy of 3.9628
Variable Finishing has Entropy of 4.1847
Variable ShortPassing has Entropy of 3.4527
Variable Dribbling has Entropy of 3.7754
Variable LongPassing has Entropy of 3.7797
Variable StandingTackle has Entropy of 4.0539
Variable SlidingTackle has Entropy of 4.0989
Split on Fifa Ovr With Information Gain of -2.083
```

2nd Approach

Sort dependent and independent variable

Dependant Variable

Dependent variables are variables which can be considered as the output of any event

In our case, the FIFA Overall is our dependent variable, which also happened to be our best information gain column

Independent Variable

- Independent variables are variables which contribute to the prediction of the dependent variable
- In our case we have at least 6 columns that are independent to get our outcome

Features scaling and data extraction

```
#Extracting Independent and dependent Variable
x= df.iloc[:, [2,3]].values
y= df.iloc[:, 4].values

# Splitting the dataset into training and test set.
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.25, random_state=0)

#feature Scaling
from sklearn.preprocessing import StandardScaler
st_x= StandardScaler()
X_train= st_x.fit_transform(x_train)
X_test= st_x.transform(x_test)
```

- Feature scaling is a method used to standardize the range of independent variables, which is also known as normalization
- This is done before training a model
- Feature scaling is necessary to get a better result from a model

Other areas where decisions can reach

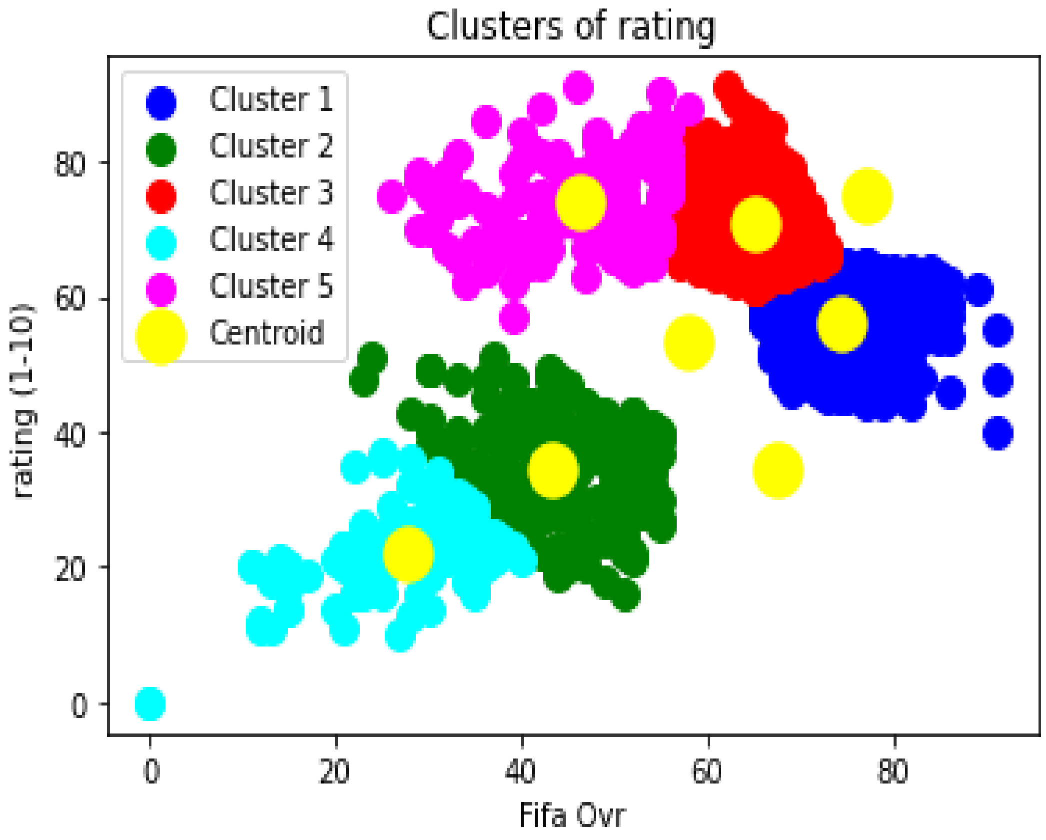
- K-means
- Clustering
- Confusion Matrix
- KNN
- Correlation Matrix
- Linear Regression (presented by Soyeon)

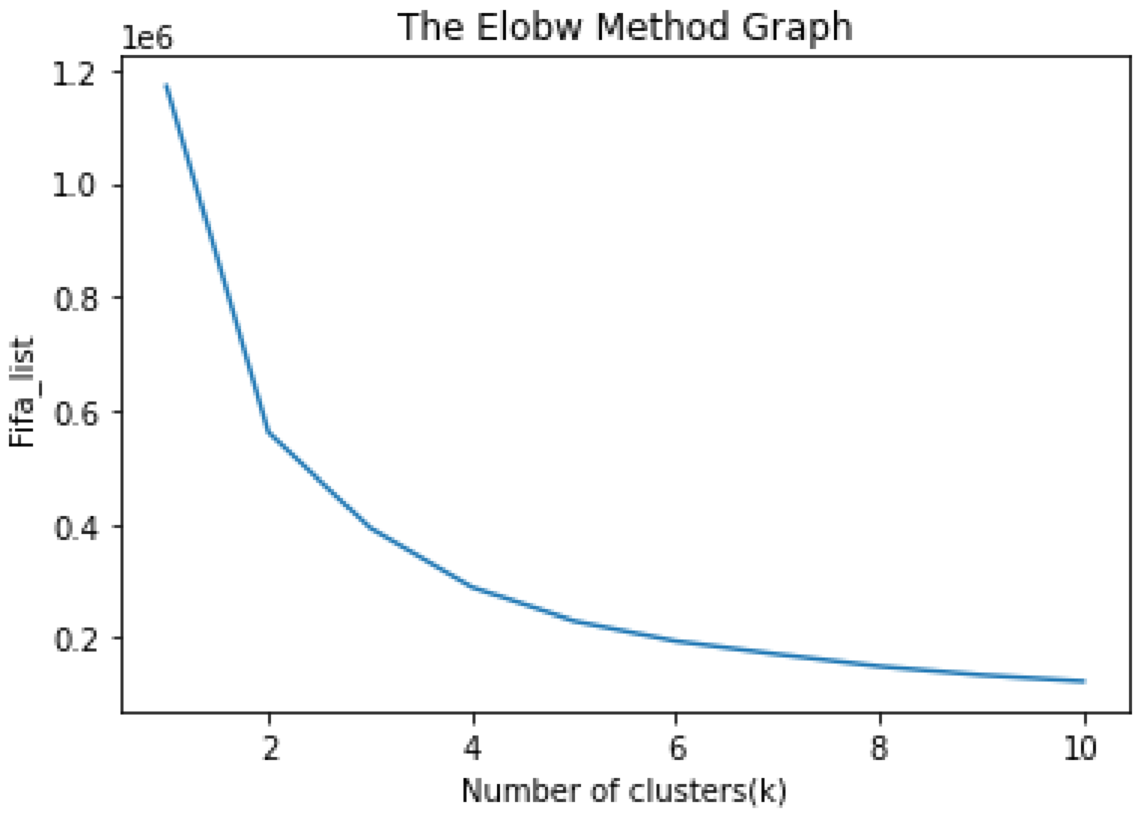
- Predicted values of our model

```
[77. 74. 76. 81. 75. 67. 75. 69. 67. 73. 82. 72. 77. 73. 72. 78. 73. 74.
 79. 76. 72. 77. 72. 79. 67. 89. 74. 76. 75. 80. 79. 78. 78. 76. 81. 60.
 73. 78. 83. 75. 63. 77. 73. 72. 69. 76. 75. 70. 80. 73. 76. 75. 64. 71.
 81. 77. 81. 78. 73. 76. 75. 72. 78. 73. 75. 68. 77. 77. 77. 81. 73. 71.
 75. 71. 74. 77. 71. 80. 76. 81. 87. 83. 75. 64. 70. 88. 73. 76. 80. 76.
 90. 76. 72. 71. 82. 64. 76. 71. 74. 64. 85. 81. 80. 75. 77. 74. 81. 70.
 69. 70. 76. 71. 75. 79. 75. 68. 73. 76. 74. 75. 72. 70. 81. 80. 73. 61.
 76. 79. 80. 70. 68. 86. 72. 77. 74. 79. 71. 61. 79. 71. 73. 73. 77. 81.
 73. 73. 61. 77. 73. 75. 76. 79. 77. 74. 75. 75. 80. 76. 75. 76. 76. 70.
 82. 80. 77. 72. 60. 63. 67. 80. 77. 89. 79. 76. 76. 72. 70. 70. 77. 77.
 84. 73. 77. 64. 77. 82. 80. 60. 76. 70. 75. 77. 74. 76. 75. 73. 77. 82.
 62. 74. 79. 76. 74. 77. 79. 77. 75. 70. 73. 79. 80. 74. 80. 73. 82. 70.
 80. 77. 81. 63. 71. 74. 72. 75. 89. 81. 79. 75. 75. 83. 80. 88. 60. 76.
 76. 76. 68. 73. 81. 75. 74. 68. 76. 65. 76. 69. 75. 64. 71. 73. 73. 81.
 75. 73. 70. 76. 76. 66. 80. 77. 78. 72. 74. 72. 73. 71. 71. 82. 70. 80.
 77. 74. 73. 70. 70. 81. 68. 80. 70. 76. 82. 89. 60. 77. 78. 74. 75. 74.
 73. 79. 72. 79. 67. 75. 78. 64. 64. 76. 75. 78. 63. 75. 68. 77. 77. 60.
 71. 75. 71. 75. 64. 76. 76. 60. 75. 76. 82. 80. 83. 79. 78. 76. 77. 79.
 76. 70. 80. 73. 73. 85. 81. 79. 74. 73. 77. 73. 80. 84. 76. 72. 84. 75.
 72. 72. 76. 74. 87. 75. 80. 71. 73. 75. 74. 77. 72. 77. 77. 73. 64. 69.
 78. 72. 78. 75. 77. 76. 76. 73. 77. 74. 79. 77. 73. 72. 75. 81. 76. 81.
 80. 73. 78. 75. 79. 71. 74. 72. 80. 83. 70. 72. 61. 74. 75. 61. 76. 74.
 75. 73. 81. 75. 70. 68. 81. 71. 80. 79. 66. 84. 76. 64. 77. 78. 74. 74.
 79. 68. 73. 78. 75. 72. 76. 85. 78. 61. 74. 89. 75. 75. 75. 79. 76. 70.
 74. 74. 74. 68. 77. 71. 89. 74. 76. 75. 86. 70. 75. 68. 76. 83. 80. 81.
 70. 82. 74. 68. 70. 77. 73. 79. 72. 75. 80. 76. 75. 78. 77. 71. 71. 72.
 75. 79. 76. 79. 75. 81. 74. 70. 78. 79. 75. 76. 70. 80. 70. 69. 75. 72.
 76. 75. 71. 65. 71. 73. 73. 75. 81. 78. 73. 80. 75. 73. 76. 74. 76. 75.
 67. 73. 67. 70. 75. 64. 73. 82. 73. 80. 79. 77. 65. 64. 73. 73. 74. 79.
 75. 76. 78. 78. 77. 76. 75. 74. 74. 69. 80. 81. 75. 84. 75. 73. 70. 72.
 68. 63. 80. 76. 76. 75. 75. 77. 79. 77. 75. 71. 71. 82. 79. 76. 76. 80.
 79. 60. 71. 76. 68. 70. 70. 84. 76. 80. 85. 78. 71. 72. 73. 76. 78. 68.
 60. 71. 71. 64. 77. 76. 79. 75. 80. 70. 77. 70. 79. 68. 71. 76. 65. 74.
 75. 78. 74. 71. 72. 73. 71. 74. 72. 72. 68. 75. 77. 75. 75. 68. 74. 71.
 74. 71. 76. 72. 82. 74. 71. 68. 75. 70. 76. 77. 81. 72. 78. 77. 76. 77.
 73. 75. 80. 75. 72. 79. 71. 71. 81. 75. 73. 74.]
```

2nd Approach

Cluster result from Decision Tree





Predict values and the accuracy

Accuracy

- Accuracy can tell us the proportion of correctly classified instances out of the total number of instances in a dataset
- In our case, the accuracy varies because we decided to test randomly
- So we have accuracy of 12%-22%

Python code for accuracy

```python
# Perform one-hot encoding for categorical columns
onehot_encoder = OneHotEncoder(sparse=False, handle_unknown='ignore')
X_encoded = pd.DataFrame(onehot_encoder.fit_transform(X_encoded))

# Split the dataset into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.3, random_state=2)

# Initialize and train the Decision Tree Classifier
clf = DecisionTreeClassifier()
clf=clf.fit(X_train, y_train)

# Predict the response for the test dataset
y_pred = clf.predict(X_test)
print(y_pred)
# Evaluate the model accuracy
accuracy = metrics.accuracy_score(y_test, y_pred)

print("Accuracy:", accuracy)
```

2nd Approach

Visualization

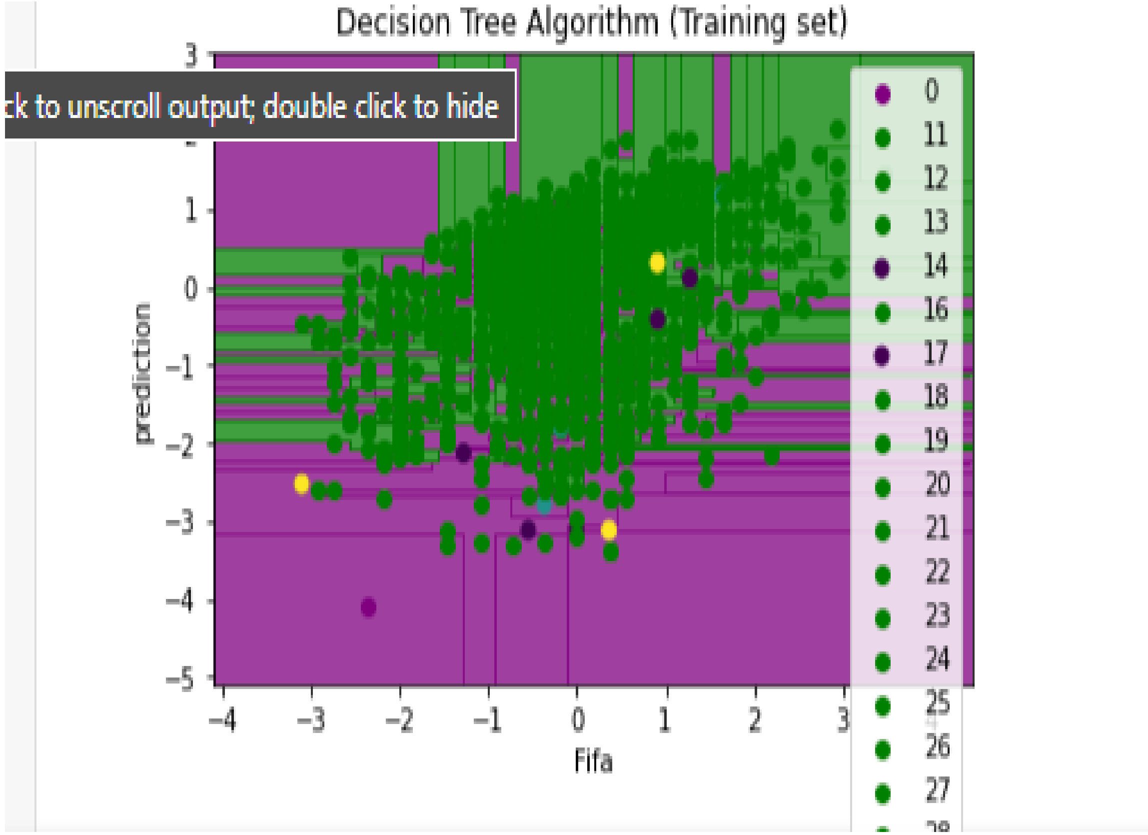
THE OUTPUT IS COMPLETELY DIFFERENT FROM THE REST OF THE CLASSIFICATION MODEL

AS WE CAN SEE THE TREE IS TRYING TO CAPTURE EACH DATASET,

WHICH IN THIS CASE CAUSES OVERFITTING

MANY OF THE GREEN AND PURPLE DATA ARE POINTING INTO EACH OTHER'S REGION

THESE ARE THE INCORRECT PREDICTIONS WHICH WE CAN FOCUS MORE WHEN IT COME TO CONFUSION MATRIX



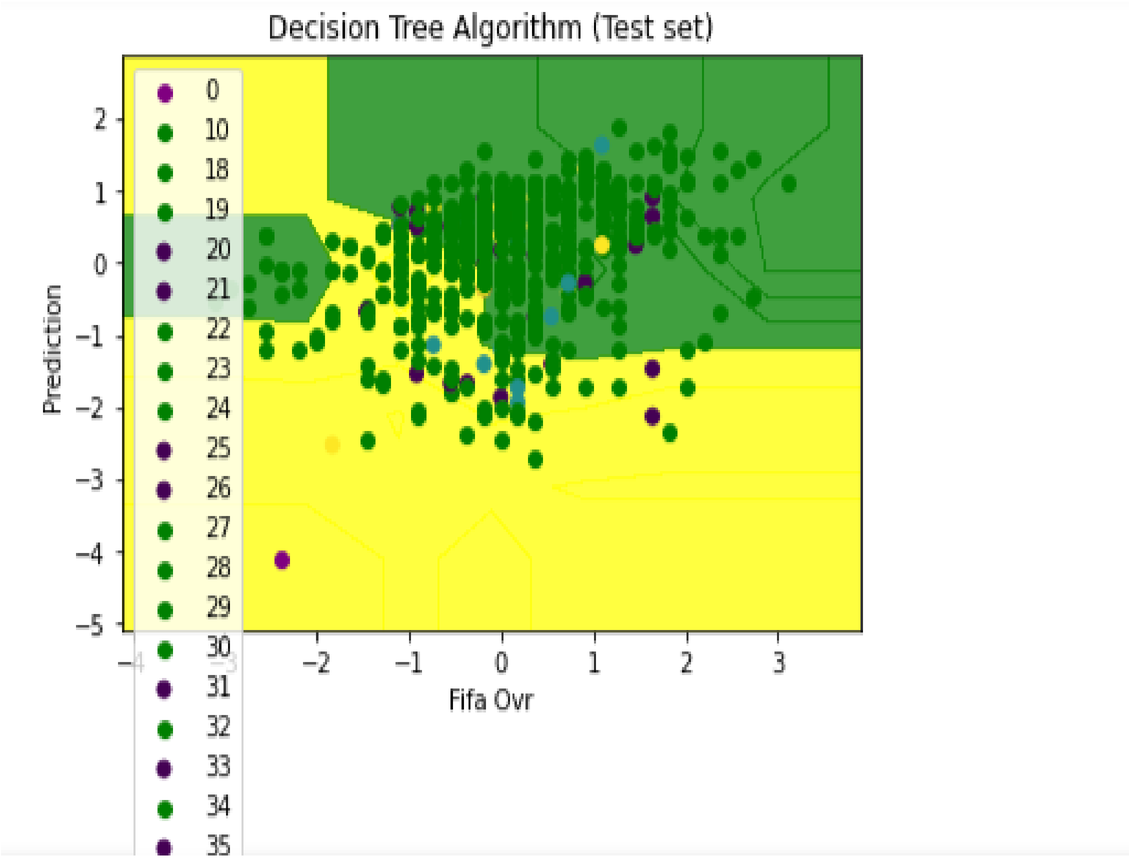
2nd Approach

Visualization

This is to see if the test set might say different, we also try to visualize the test set result

For any questions on this or if you want to understand more kindly get back to us per email

We see many intrusions in this case also



Evaluation & Challenges

- Understanding data
- Python code
- Research
- Decision taken

Conclusion

- Finally based on our research, we realized that we can't use FIFA as a digital twin of real-life football, but this is due to the data set that we had
- Considering the later development this might not be the case due to better technology and deeper research in the area of analysis

QUESTIONS?

THANK YOU