FIFA 22 Players

Phase 1

The purpose of this project is to mine the players dataset from FIFA 22 and create a model for determining the best position for a player depending on the feat features.

The dataset contains information on the skills a player needs to handle for predicting growth of a player based on all the current values of the attributes/skills.

The dataset was downloaded from https://www.kaggle.com/datasets/aryan25-11/fifa-22-complete-player-dataset

The dataset contains 19239 rows and 106 columns. 104 of the rows are FIFA registered players. So we can consider this dataset as the whole universe of players for the project.

The dataset contains attributes related to different aspects of a player such as overall ability as well as some other features. i.e. value, wage, nationality, club, contract, etc. Some of the attributes are specific to the game and are not directly related to the real-world attributes of a player. For example, the attribute "ball control" is a measure of how good a player is with the ball. This is not the case in real-world football where ball control is a FIFA rating or skill or a measure of a player. Some of such attributes are specific to the game and are not directly related to real-world attributes. Such attributes are dropped in the process of cleaning the data.

Import libraries

```
%matplotlib inline
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns, sklearn, sklearn.preprocessing, sklearn
from sklearn.model_selection import train_test_split, cross_validation, sklearn
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_absolute_error, r2_score
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import GradientBoostingRegressor, AdaBoostRegressor, ExtraTreesRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR, LinearSVR
from sklearn.linear_model import Lasso, ElasticNet, Ridge, BayesianRidge, LassoLars, OrthogonalMatchingPursuit, PassiveAggressiveRegressor, SGDRegressor, TheilSenRegressor, RANSACRegressor, HuberRegressor
from sklearn.model_selection import cross_val_score, KFold
from sklearn.preprocessing import StandardScaler, MinMaxScaler, RobustScaler
from sklearn.pipeline import Pipeline
from sklearn.decomposition import PCA
```

Import CSV file

```
df = pd.read_csv('/kaggle/input/fifa-22-complete-dataset/fifa22.csv')
df.to_csv(path_wd+'/fifa22.csv', index=False)
df.head(5)
df.shape
```

	Name	FullName	Age	Photo	Height	Weight	PhotoURL	Nationality	Overall	Potential	Hits
0	158023	L. Messo	Lionel Messi	35	170	72	https://cdn.sofifa.net/players/158/023/22_60.png	Argentina	93	93	
1	20801	C. Ronaldo dos Santos Aveiro	Cristiano Ronaldo	37	187	83	https://cdn.sofifa.net/players/020/801/22_60.png	Portugal	91	91	
2	190871	Neymar da Silva Santos Júnior	Neymar Jr	29	175	68	https://cdn.sofifa.net/players/190/871/22_60.png	Brazil	91	91	
3	207865	R. Lewandowski	Robert Lewandowski	32	185	81	https://cdn.sofifa.net/players/207/865/22_60.png	Poland	91	91	
4	212831	Alisson Ramses Becker	Alisson	29	191	81	https://cdn.sofifa.net/players/212/831/22_60.png	Brazil	91	91	

```
(19239, 106)
```

```
# Replace the NaN values of each column
df.fillna(0,inplace=True)
df.isnull.sum(axis=0).sort_values
Unique\nCount, 22\n22 players, 1\n, 19239
Name\nCount, 19239\nLuis Ortiz, 2\n, 19237
FullName\nCount, 19239\nLuis Ortiz, 2\n, 19237
Age\n\n\n
```

```
# of people with combinations of variables dataset / they have the same Regions
# sorted by the number of people in descending order / Regions with the most people
count_of_combinations_df = df[['Region', 'Country']].groupby(['Region', 'Country'])
                           .size().sort_values(ascending=False)
                           .reset_index().head(20)
count_of_combinations_df.shape, type(count_of_combinations_df)
count_of_combinations_df.dtypes, type(count_of_combinations_df.dtypes)
count_of_combinations_df.columns, type(count_of_combinations_df.columns)
```

```
((20, 3), pandas.core.frame.DataFrame)
(Region    object
Country   object
dtype: object, pandas.core.series.Series)
(Index(['Region', 'Country', 0], dtype='object'), pandas.core.indexes.base.Index)
```

	Region	Country	0
0	Region8	Indonesia	14,6250
1	Region8	Philippines	13,9750
2	Region8	Thailand	13,7500
3	Region8	Mauritius	13,7500
4	Region8	Kenya	13,6667
5	Region8	Malaysia	13,6667
6	Region8	Cambodia	13,6667
7	Region8	Tanzania	13,6667
8	Region8	Nepal	13,6667
9	Region8	Vietnam	13,6667
10	Region8	Laos	13,6667
11	Region8	Ethiopia	13,6667
12	Region8	Ghana	13,6667
13	Region8	Cameroon	13,6667
14	Region8	Uganda	13,5000
15	Region8	Zambia	13,5000
16	Region8	Mozambique	13,5000
17	Region8	Madagascar	13,5000
18	Region8	Malawi	13,5000
19	Region8	Lesotho	13,5000
20	Region8	Botswana	13,5000
21	Region8	Swaziland	13,5000
22	Region8	Burundi	13,5000
23	Region8	Zimbabwe	13,5000
24	Region8	Rwanda	13,5000
25	Region8	Comoros	13,5000
26	Region8	Mauritania	13,4167
27	Region8	Senegal	13,4167
28	Region8	Gambia	13,4167
29	Region8	Guinea	13,4167
30	Region8	Liberia	13,4167
31	Region8	Sierra Leone	13,4167
32	Region8	Togo	13,4167
33	Region8	Burkina Faso	13,4167
34	Region8	Mali	13,4167
35	Region8	Niger	13,4167
36	Region8	Chad	13,4167
37	Region8	Central African Republic	13,4167
38	Region8	Congo	13,4167
39	Region8	Congo (Kinshasa)	13,4167
40	Region8	Angola	13,4167
41	Region8	Mozambique	13,4167
42	Region8	Zambia	13,4167
43	Region8	Malawi	13,4167
44	Region8	Tanzania	13,4167
45	Region8	Uganda	13,4167
46	Region8	Kenya	13,4167
47	Region8	Rwanda	13,4167
48	Region8	Burundi	13,4167
49	Region8	Zimbabwe	13,4167
50	Region8	Botswana	13,4167
51	Region8	Lesotho	13,4167
52	Region8	Swaziland	13,4167
53	Region8	Madagascar	13,4167
54	Region8	Comoros	13,4167
55	Region8	Mauritius	13,4167
56	Region8	Ethiopia	13,4167
57	Region8	Ghana	13,4167
58	Region8	Cameroon	13,4167
59	Region8	Nepal	13,4167
60	Region8	Vietnam	13,4167
61	Region8	Laos	13,4167
62	Region8	Cambodia	13,4167
63	Region8	Malaysia	13,4167
64	Region8	Indonesia	13,4167
65	Region8	Philippines	13,4167
66	Region8	Thailand	13,4167
67	Region8	Mauritius	13,4167
68	Region8	Kenya	13,4167
69	Region8	Malaysia	13,4167
70	Region8	Cambodia	13,4167
71	Region8	Tanzania	13,4167
72	Region8	Nepal	13,4167
73	Region8	Vietnam	13,4167
74	Region8	Laos	13,4167
75	Region8	Ethiopia	13,4167
76	Region8	Ghana	13,4167
77	Region8	Cameroon	13,4167
78	Region8	Uganda	13,4167
79	Region8	Zambia	13,4167
80	Region8	Mozambique	13,4167
81	Region8	Madagascar	13,4167
82	Region8	Malawi	13,4167
83	Region8	Lesotho	13,4167
84	Region8	Botswana	13,4167
85	Region8	Swaziland	13,4167
86	Region8	Burundi	13,4167
87	Region8	Zimbabwe	13,4167
88	Region8	Rwanda	13,4167
89	Region8	Comoros	13,4167
90	Region8	Mauritania	13,4167
91	Region8	Senegal	13,4167
92	Region8	Gambia	13,4167
93	Region8	Guinea	13,4167
94	Region8	Liberia	13,4167
95	Region8	Sierra Leone	13,4167
96	Region8	Togo	13,4167
97	Region8	Burkina Faso	13,4167
98	Region8	Mali	13,4167
99	Region8	Niger	13,4167
100	Region8	Chad	13,4167
101	Region8	Central African Republic	13,4167
102	Region8	Congo	13,4167
103	Region8	Congo (Kinshasa)	13,4167
104	Region8	Angola	13,4167
105	Region8	Mozambique	13,4167
106	Region8	Zambia	13,4167
107	Region8	Malawi	13,4167
108	Region8	Tanzania	13,4167
109	Region8	Uganda	13,4167
110	Region8	Kenya	13,4167
111	Region8	Rwanda	13,4167
112	Region8	Burundi	13,4167
113	Region8	Zimbabwe	13,4167
114	Region8	Botswana	13,4167
115	Region8	Lesotho	13,4167
116	Region8	Swaziland	13,4167
117	Region8	Madagascar	13,4167
118	Region8	Comoros	13,4167
119	Region8	Mauritius	13,4167

```
# of people with combinations of variables dataset / they have the same Regions
# sorted by the number of people in descending order / Regions with the most people
count_of_combinations_df = df[['Region', 'Country']].groupby(['Region', 'Country'])
                           .size().sort_values(ascending=False)
                           .reset_index()
count_of_combinations_df.shape, type(count_of_combinations_df)
count_of_combinations_df.dtypes, type(count_of_combinations_df.dtypes)
count_of_combinations_df.columns, type(count_of_combinations_df.columns)
```

```
((119
```

60
40
20
0
−20
−40
−60
−80
−100
−120
−140
−160
−180
−200
−220
−240
−260
−280
−300
−320
−340
−360
−380
−400
−420
−440
−460
−480
−500
−520
−540
−560
−580
−600
−620
−640
−660
−680
−700
−720
−740
−760
−780
−800

(Height)

(Weight)

(Height)

(Weight)

(Height)

(Weight)

```
In [17]: df.to_csv('Sensor.csv')
         df_pivot = df.pivot(index='Id', columns='Field', value='Measurement')
         df_pivot.to_csv('Sensor_pivot.csv')
         df_pivot.shape, df.shape, df.columns, df_pivot
         df_pivot.isnull().sum()
         data = df[['Measurement', 'Id', 'Sensorposition']].dropna().copy()
         data['Sensorposition'] = data['Sensorposition'].astype('category')
         data['Id'] = data['Id'].astype('category')
         pd.crosstab(data['Id'], data['Sensorposition'], dtype=int).shape
```

```
In [18]: from tqdm import tqdm
         from sklearn.model_selection import train_test_split, GridSearchCV
         from sklearn.preprocessing import StandardScaler
         from sklearn.decomposition import PCA
         from sklearn.cluster import KMeans, DBSCAN
         from sklearn.metrics import silhouette_score, pairwise_distances
```

```
In [19]: !pwd
```

```
/content/drive/My Drive
```