In [40]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
```

In [41]:

```
import pandas as pd
football = pd.read_csv(r'/content/reformed_datset.csv', encoding='iso-8859-1')
football.head()
```

Out[41]:

	Unnamed: 0	Unnamed: 0.1	ID	Name	Age	Photo	Nationality	Flag	Overall	Potent
0	0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	
1	1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	
2	2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	
3	3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	
4	4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	

5 rows × 90 columns

In [42]:

```
df = football[['Name','Age',    'Overall',      'Potential','Value',    'Wage', 'Special'       ,       'Internationa
               'Weak Foot',     'Skill Moves',  'Jersey Number',        'Height','LS',
               'ST','RS','LW','LF','CF','RF','RW','LAM','CAM','RAM','LM','LCM','CM','RCM',
               'RM','LWB','LDM','CDM','RDM'  ,'RWB' ,'LB' ,'LCB','CB','RCB','RB','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','GKD
                'GKHandling','GKKicking','GKPositioning','GKReflexes','Release Clause'
        ]]

df = df[df.Overall > 86]
names = df.Name.tolist()

df = df.drop(['Name'], axis = 1)
df.head()
```

Out[42]:

	Age	Overall	Potential	Value	Wage	Special	International Reputation	Weak Foot	Skill Moves	Jersey Number	Height	LS	ST	RS	LW	LF
0	31	94	94	110.5	0.565	2202	5.0	4.0	4.0	10.0	170.18	57.81547	57.81547	57.81547	59.03765	58.71939
1	33	94	94	77.0	0.405	2228	5.0	4.0	5.0	7.0	187.96	57.81547	57.81547	57.81547	59.03765	58.71939
2	26	92	93	118.5	0.290	2143	5.0	5.0	5.0	10.0	175.26	57.81547	57.81547	57.81547	59.03765	58.71939
3	27	91	93	72.0	0.260	1471	4.0	3.0	1.0	1.0	193.04	57.81547	57.81547	57.81547	59.03765	58.71939
4	27	91	92	102.0	0.355	2281	4.0	5.0	4.0	7.0	154.94	57.81547	57.81547	57.81547	59.03765	58.71939

In [43]:

```
from sklearn import preprocessing

x = df.values # numpy array
mmscaler = preprocessing.MinMaxScaler()
xscaled = scaler.fit_transform(x)
Xnormalized = pd.DataFrame(xscaled)
```

In [44]:

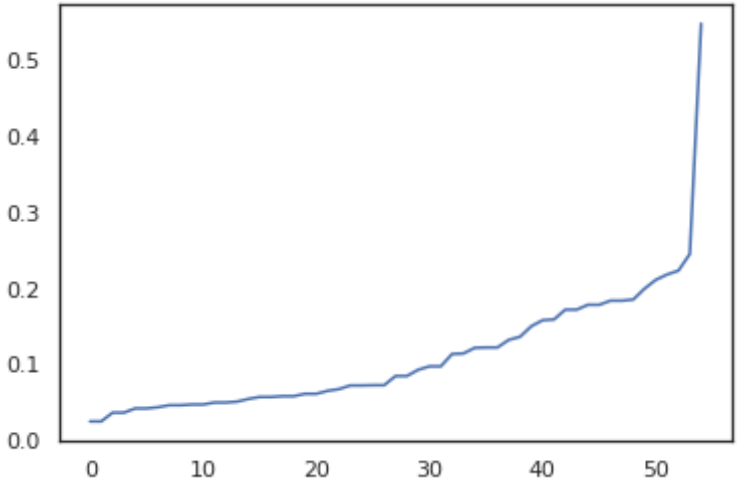
```
from sklearn.decomposition import PCA

pca = PCA(n_components = 2)
transform = pd.DataFrame(pca.fit_transform(Xnormalized))
```

In [45]:

```
from sklearn.neighbors import NearestNeighbors

# calculate the distance from each point to its closest neighbor
```

```
nn = NearestNeighbors(n_neighbors = 2)

# fit the nearest neighbor
nbr = nn.fit(transform)

# returns two arrays - distance to the closest n_neighbors points and index for each point
distances, indices = nbr.kneighbors(transform)

# sort the distance and plot it
distances = np.sort(distances, axis=0)
distances = distances[:,1]
plt.plot(distances)
```

Out[45]: [<matplotlib.lines.Line2D at 0x7f70bd5b0c50>]



In [47]:

```
from sklearn.cluster import DBSCAN

db= DBSCAN(eps=0.3, min_samples=5)

db_clusters = db.fit_predict(transform)

transform['cluster'] = db_clusters
transform['Name'] = names
transform.columns = ['x', 'y', 'cluster', 'Name']
transform.head()
```

Out[47]:

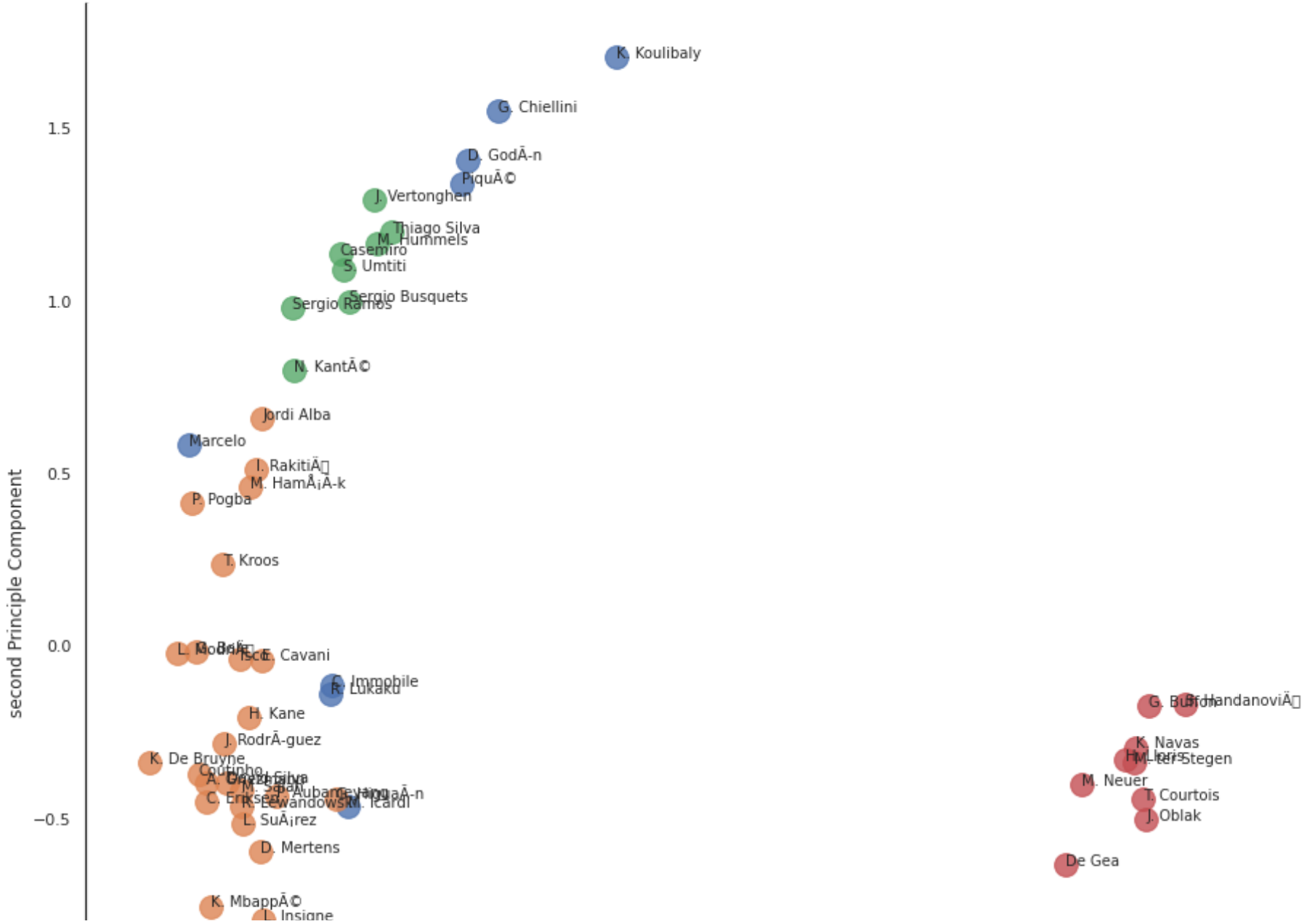
	x	y	cluster	Name
0	-1.237459	-1.390348	-1	L. Messi
1	-1.082082	-0.972013	0	Cristiano Ronaldo
2	-1.099776	-1.277386	-1	Neymar Jr
3	2.893399	-0.636772	2	De Gea
4	-1.184405	-0.340630	0	K. De Bruyne

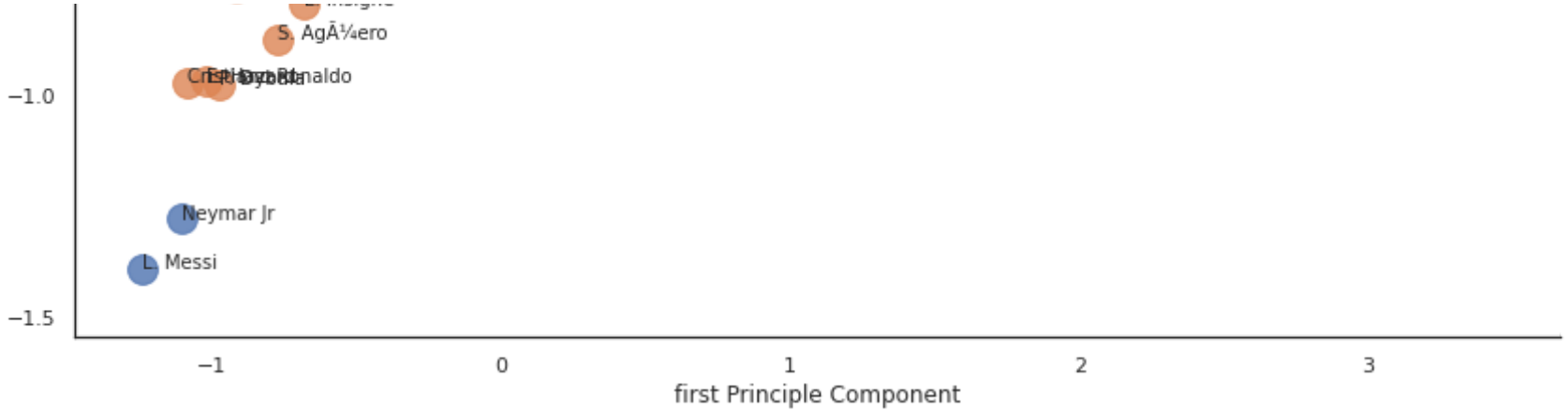
In [49]:

```
import matplotlib.pyplot as plt
import seaborn as sb
%matplotlib inline


ax = sb.lmplot(x="x", y="y", hue='cluster', data = transform, legend=False,
                   fit_reg=False, size = 12, scatter_kws={"s": 250})

texts = []
for x, y, s in zip(transform.x, transform.y, transform.Name):
    texts.append(plt.text(x, y, s))

plt.xlabel("first Principle Component")
plt.ylabel("second Principle Component")

plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/regression.py:581: UserWarning: The `size` parameter has been renamed
to `height`; please update your code.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.7/dist-packages/matplotlib/backends/backend_agg.py:214: RuntimeWarning: Glyph 135 missing from
current font.
  font.set_text(s, 0.0, flags=flags)
/usr/local/lib/python3.7/dist-packages/matplotlib/backends/backend_agg.py:214: RuntimeWarning: Glyph 141 missing from
current font.
  font.set_text(s, 0.0, flags=flags)
/usr/local/lib/python3.7/dist-packages/matplotlib/backends/backend_agg.py:183: RuntimeWarning: Glyph 135 missing from
current font.
  font.set_text(s, 0, flags=flags)
```

```
/usr/local/lib/python3.7/dist-packages/matplotlib/backends/backend_agg.py:183: RuntimeWarning: Glyph 141 missing from
current font.
  font.set_text(s, 0, flags=flags)
```





In [ ]:

ANALYSIS

1. 180-185 interval for height contains maximum number of players.

1. In scatter plot of Potential vs Wage, players having potential greater than 90 are showing non uniform distribution of wages. 1 player is found to have wage above 0.5. Another interesting observation made where the player with maximum potential has wage close to 0.1 only.

Thus, these 2 being the outliers clearly.

1. Pie chart for skill move showed that Move 1 is most favourable (close to 48%) and Move5 being the least.

1. Bar graph showing the top 20 countries having maximum number of players shows that England is leading with around 1700 players.

1. Another bar graph showing counts of different position shows ST is the most favourable position.

1. Histogram made on the basis of age shows that most players are of the age 18 to 27, with more density between 22 to 25.

CLUSTERS

Clusters are well formed as per the silhoutte score obtained for K-Means, however there are many outliers. Intraclass similarity is high, while

interclass similarity is low.

The best cluster is observed for class value k=3 (as per elbow method). Parameter used is WCSS.

Cluster formation is similar for both heirarchical and K-means clustering.

In divisive heirarchical clustering, the intra class similarity is obtained using dissimilarity matrix and euclidean distance.

The Dendodrams in both the methods look pretty similar, however clusters formed are better in divisive heirarchical clustering than aglomerative method.

Having silhoutte score less than 1 is a marker for good cluster formation. In our case, we got an average score of 0.4 ranging from 0.28 to 0.53.

From scatterplot distribution of clusters, it can be observed that for majority of cases, wage and value are proportional. However few outliers show that, certain high valued players are under paid.

Using nearest neighbour, we got final value of epsilon as 0.3 and minPts as 5 for DB scan.

In [ ]: