Data Mining

Project Report

Student 1 : ابراهيم رفاعي كمال عبدالفتاح محمد

ID :20191480247

Student 2: احمد ايهاب محمد عبدالفتاح

ID :20191565981

Dataset link

https://www.kaggle.com/sohommajumder21/fifa-2018-world-cup-players/code

Dataset Description:

the dataset we used it's a data about world club it contains

Hierarchical Clustering

is an algorithm that groups similar objects into groups called **clusters**. The endpoint is a set of **clusters**, where each **cluster** is distinct from each other **cluster**, and the objects within each **cluster** are broadly similar to each other.

Code implementation:

- Imported all required libraries

```
In [1]: # 1st thing to do is to import needed libraries
        import numpy as np
        import pandas as pd
        import scipy.cluster.hierarchy as sch
        from sklearn.cluster import AgglomerativeClustering
        import matplotlib.pyplot as plt
```

- Then used read method to read data set

```
wc_data= pd.read_csv('all_wc_18_players_fifa.csv')
```

- In this step we had used shape method to know the features of dataset

```
In [3]: #now show number of coulmns and rows
        wc_data.shape

Out[3]: (736, 12)
```

- Now we used head method to show dataset content:

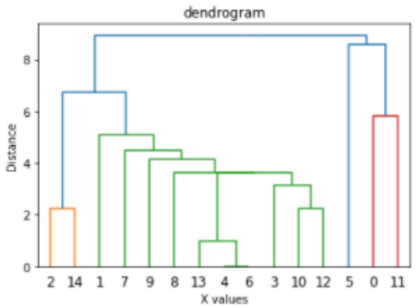
```
In [18]: # show dataset content
         wc_data.head

Out[18]: <bound method NDFrame.head of           team  number position  birth_date        shirt_name  \
         0    Argentina       1       GK  1986-02-10            GUZMÁN
         1    Argentina       2       DF  1987-03-18          MERCADO
         2    Argentina       3       DF  1992-08-31       TAGLIAFICO
         3    Argentina       4       DF  1986-09-20          ANSALDI
         4    Argentina       5       MF  1986-01-30            BIGLIA
         ..         ...     ...      ...         ...               ...
         731    Uruguay      19       DF  1990-10-07         S. COATES
         732    Uruguay      20       FW  1990-03-19  J. URRETAVISCAYA
         733    Uruguay      21       FW  1987-02-14         E. CAVANI
         734    Uruguay      22       DF  1987-04-07        M. CACERES
         735    Uruguay      23       GK  1983-03-25          M. SILVA

                              club  height  weight league        age  \
         0               Tigres UANL     192      90    MEX  32.339726
         1                Sevilla FC     181      81    ESP  31.241096
         2                  AFC Ajax     169      65    NED  25.786301
         3                 Torino FC     181      73    ITA  31.731507
         4                   AC Milan     175      73    ITA  32.369863
         ..                       ...     ...     ...    ...        ...
         731              Sporting CP     196      89    POR  27.684932
         732            CF Monterrey      172      66    MEX  28.238356
         733  Paris Saint-Germain FC      188      78    FRA  31.328767
         734                SS Lazio      178      75    ITA  31.186301
         735        CR Vasco da Gama      187      82    BRA  35.221918

                               name  caps
         0              Nahuel Guzmán     6
         1           Gabriel Mercado     20
         2         Nicolás Tagliafico     4
         3           Cristian Ansaldi     5
         4               Lucas Biglia    57
         ..                       ...   ...
         731         Sebastián Coates     30
         732  Jonathan Urretaviscaya      4
         733           Edinson Cavani   101
         734           Martín Cáceres    76
         735             Martín Silva    11

         [736 rows x 12 columns]>
```

- Use iloc method to select number of features that we need:

```
In [5]: #now use iloc method to select number of feature that we need.
        data = wc_data.iloc[:,6:8].values
```

- Now we used dendrogram to graph our selected features to pretend it as clusters

```
In [6]: # using dendrogram to draw our single Linkage method for first 15 columns.
        dendrogram=sch.dendrogram(sch.linkage(data[:15],'single'))
        plt.title('dendrogram')
        plt.xlabel('X values')
        plt.ylabel('Distance')
        plt.show()
```

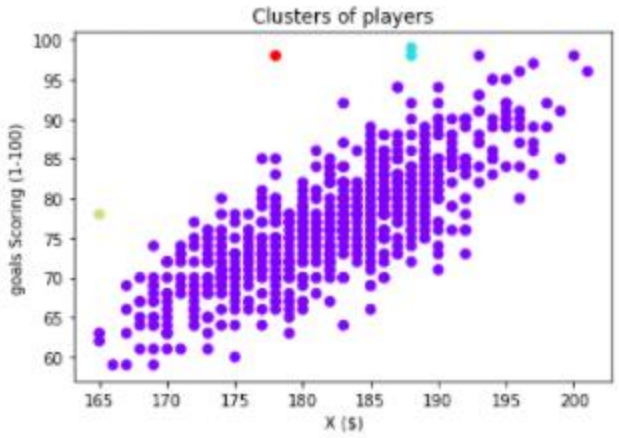


- Then we cluster and predict our results into groups and show it in array :

```
In [7]: # now we cluster &predict our Data into groups and fitting it
        cluster = AgglomerativeClustering(n_clusters=4, affinity='euclidean',linkage='single')
        cluster.fit_predict(data)

Out[7]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int64)
```

- This point we will visualize our results using scatter plot:

```
In [8]: # visulize our data using scatter plot.
        plt.scatter(data[:,0], data[:,1], c=cluster.labels_, cmap='rainbow')
        plt.title('Clusters of players')
        plt.xlabel('X ($)')
        plt.ylabel('goals Scoring (1-100)')

Out[8]: Text(0, 0.5, 'goals Scoring (1-100)')
```



K-medoids

k-medoids is a general version of k-means where we calculate with it medoid also medoid make minimize distance between every point and its medoid.

Medoid has initial point, but that initial point should be existing in the data set we used.

code Implementation:

- We used:

```
In [56]: !pip install scikit-learn-extra
```

To install sickit - learn-extra to import k-medoids.

So, the result is:

```
Collecting scikit-learn-extra
  Downloading scikit_learn_extra-0.2.0-cp38-cp38-win_amd64.whl (381 kB)
Requirement already satisfied: numpy>=1.13.3 in c:\users\20109\anaconda3\lib\site-packages (from scikit-learn-extra) (1.19.2)
Requirement already satisfied: scipy>=0.19.1 in c:\users\20109\anaconda3\lib\site-packages (from scikit-learn-extra) (1.5.2)
Requirement already satisfied: scikit-learn>=0.23.0 in c:\users\20109\anaconda3\lib\site-packages (from scikit-learn-extra) (0.
23.2)
Requirement already satisfied: joblib>=0.11 in c:\users\20109\anaconda3\lib\site-packages (from scikit-learn>=0.23.0->scikit-le
arn-extra) (0.17.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\20109\anaconda3\lib\site-packages (from scikit-learn>=0.23.0->s
cikit-learn-extra) (2.1.0)
Installing collected packages: scikit-learn-extra
Successfully installed scikit-learn-extra-0.2.0
```

- We used:

```
import numpy as np
```

- To read the file we uploaded as follows.

```
wc_data= pd.read_csv('all_wc_18_players_fifa.csv')
```

- To show data of my dataset we use:

```
In [12]: wc_data.head
```

- To determine features that needed to use k-medoids as we choose height and weight as features we use :

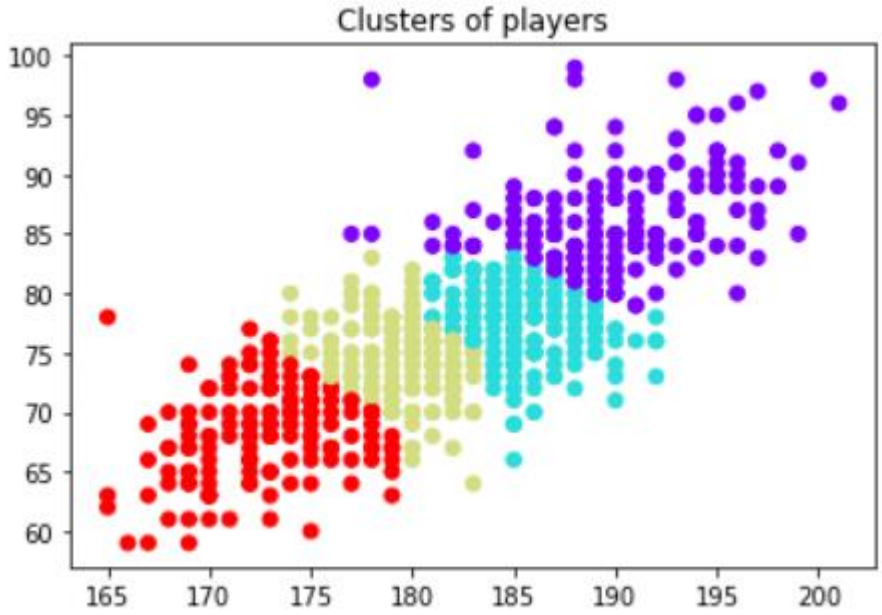
```
data = shopping_data.iloc[:, 3:5].values
```

- To create object from k-medoid class where the used distance is Manhattan and initial point is random that can choose any point not special point as start, read data and return label we use :

```
In [14]: cluster = KMedoids(n_clusters=4, metric="manhattan",init="random") #random_state=33

         cluster.fit_predict(data)
```

- To scatter data and plot it in a graph as follows we used :

```
plt.scatter(data[:,0], data[:,1], c=cluster.labels_, cmap='rainbow')
```



Where it shows cluster of players by using features weight and height

X label is height and y label is weight

We notice from graph as we go max of top or max of bottom that data become more spread.

Also as data become less spread so most of data concentrate at that point as 180,190 and so on