KMeans Clusturing Algorithm

KMeans algorithm is one of the most basic algorithms in unsupervised machine learning in which we only give the data and we let the machine divide the data into specific groups(clustures) depending on our algorithms.

So what does the KMeans algorithm look like?

Here is the simplified algorithm

*Choose the number of clusters you want to have

*Until the centers converge to a stable solution:

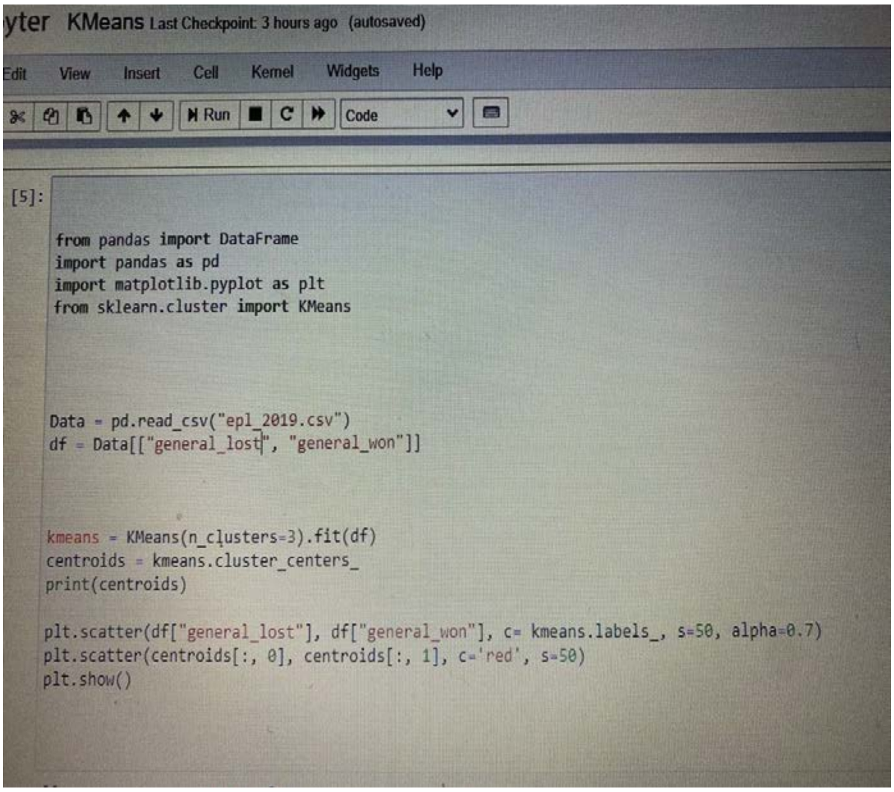
*Assign each data point to the nearest cluster center.

*Update each cluster by replacing it with the mean of all points assigned to that cluster in the previous step.

As we can see from the above algorithm ,the very first thing to do is deciding how many clusture centeres(catagories) we want to have. We continue by randomly placing our cluster centers around our data and start assigning the datas to their nearest respective clustures.However,we can see that this can't be our final result as the algorithm continuously reallocates the cluster centeres at a mid distance from all the datas assigned to it.This process keeps happening until our cluster centers give us constant results.Here is an image of what clustered data looks like   https://techwithtim.net/wp-content/uploads/2019/01/k-means-example.png.

We can use this algorithm in football to determine which teams perform well,average and which teams are in danger of being relegated from a league.

Below you will find an example based on the 2019/20 English Premier League which is based on the number of games won and number of games lost.


yter KMeans Last Checkpoint: 3 hours ago (autosaved)
Edit View Insert Cell Kernel Widgets Help
Run
Code
[5]:

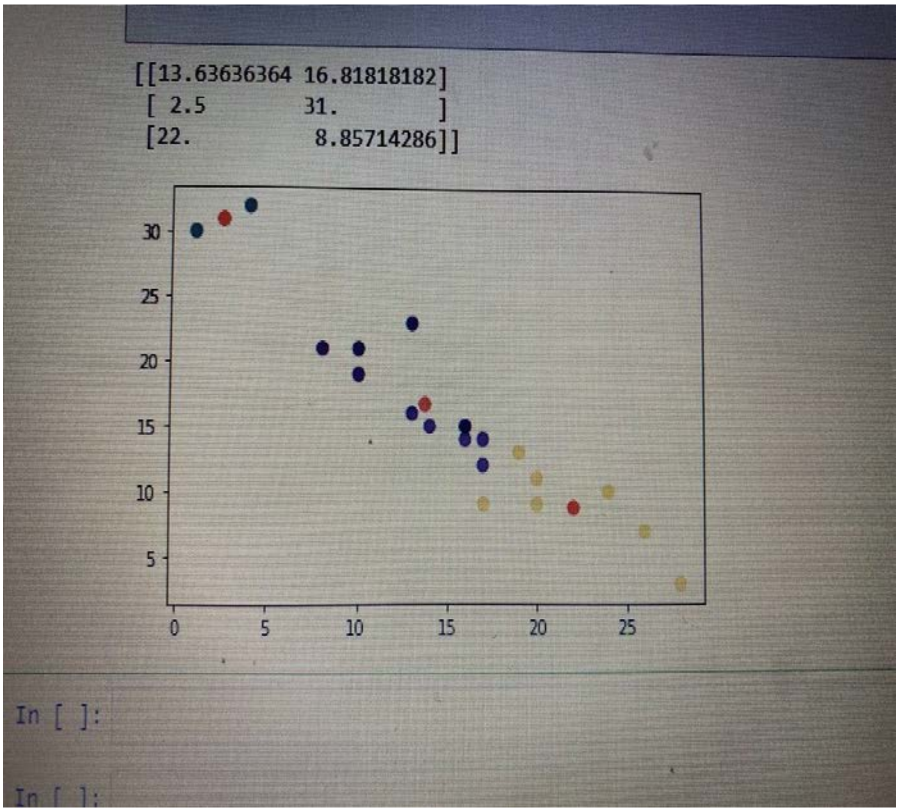
```
from pandas import DataFrame
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans



Data = pd.read_csv("epl_2019.csv")
df = Data[["general_lost", "general_won"]]


kmeans = KMeans(n_clusters=3).fit(df)
centroids = kmeans.cluster_centers_
print(centroids)

plt.scatter(df["general_lost"], df["general_won"], c= kmeans.labels_, s=50, alpha=0.7)
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', s=50)
plt.show()
```



```
[[13.63636364 16.81818182]
 [ 2.5        31.        ]
 [22.          8.85714286]]
```

As we can see from the graph, the red points represent the 3 centoids we have.

*Light Blue =Good performance

*Dark Blue =Average performance

*Yellow =Risk of relegation

This is a simple application of KMeans algorithm as there are more sophisticated data types we can use it on especially when we have more than two parameters to consider when implementing the algorithm.