Div: B
Roll No.: 422004
Gr. No.: 21810939
Name of Student: Aakanksha Bhondve
subject: Advanced Machine Learning

Assignment No. 2

Title: Write a program to do following:
We have given a collection of 8 points. P1=[0.1,0.6] P2=[0.15,0.71] P3=[0.08,0.9] P4=[0.16, 0.85] P5=[0.2,0.3] P6=[0.25,0.5] P7=[0.24,0.1] P8=[0.3,0.2]. Perform the k-mean clustering with initial centroids as m1=P1 =Cluster#1=C1 and m2=P8=cluster#2=C2. Answer the following

1] Which cluster does P6 belongs to?
2] What is the population of cluster around m2?
3] What is updated value of m1 and m2?

Theory:

1) Supervised learning algorithms:

Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y). Supervised learning can be further divided into two types of problems:

1. **Regression**
   Regression algorithms are used if there is a relationship between the input variable and the output variable. It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc. Below are some popular Regression algorithms which come under supervised learning:
   - Linear Regression
   - Regression Trees
   - Non-Linear Regression
   - Bayesian Linear Regression
   - Polynomial Regression

2. **Classification**

   Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-false, etc.

   - Random Forest
   - Decision Trees
   - Logistic Regression
   - Support vector Machines

2) Working of K-means algorithm:

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.

It is an iterative algorithm that divides the unlabelled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

The working of the K-Means algorithm is:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7: The model is ready.

Implementation link:

https://colab.research.google.com/drive/1fY4jhH9XE4q417aHQGeK_V5K2J2RbWkN#scrollTo=ee3dbfe0

Code & Output:

Assignment-2_422004.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

RAM Disk Editing

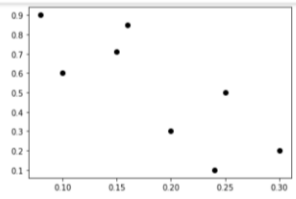
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
df=pd.DataFrame({'X':[0.1,0.15,0.08,0.16,0.2,0.25,0.24,0.3],
                 'y':[0.6,0.71,0.9,0.85,0.3,0.5,0.1,0.2]})
f1 = df['X'].values
f2 = df['y'].values
X = np.array(list(zip(f1, f2)))
print(X)
```

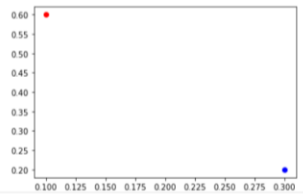
```
[[0.1  0.6 ]
 [0.15 0.71]
 [0.08 0.9 ]
 [0.16 0.85]
 [0.2  0.3 ]
 [0.25 0.5 ]
 [0.24 0.1 ]
 [0.3  0.2 ]]
```

```
C_x=np.array([0.1,0.3])
C_y=np.array([0.6,0.2])
centroids=C_x,C_y
```

```
colmap = {1: 'r', 2: 'b'}
plt.scatter(f1, f2, color='k')
plt.show()
```




```
plt.scatter(C_x[0],C_y[0], color=colmap[1])
plt.scatter(C_x[1],C_y[1], color=colmap[2])
plt.show()
```

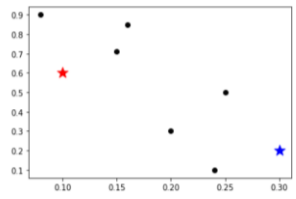


0s completed at 00:13

```
C = np.array(list((C_x, C_y)), dtype=np.float32)
print (C)
```

```
[[0.1 0.3]
 [0.6 0.2]]
```

```
plt.scatter(f1, f2, c='#050505')
plt.scatter(C_x[0], C_y[0], marker='*', s=200, c='r')
plt.scatter(C_x[1], C_y[1], marker='*', s=200, c='b')
plt.show()
```




```
from sklearn.cluster import KMeans
model=KMeans(n_clusters=2,random_state=0)
model.fit(X)
labels=model.labels_
print(labels)
```

0s completed at 00:13

```
labels=model.labels_
print(labels)
```

```
[1 1 1 1 0 0 0 0]
```

```
print('P6 belongs to cluster',model.labels_[5])
```

```
P6 belongs to cluster 0
```

```
count=0
for i in range(len(labels)):
    if (labels[i]==1):
        count=count+1

print('No of population around cluster 2:',count-1)
```

```
No of population around cluster 2: 3
```

```
new_centroids = model.cluster_centers_

print('Previous value of m1 and m2 is:')
print('M1==',centroids[0])
print('M1==',centroids[1])
```

```
Previous value of m1 and m2 is:
M1== [0.1 0.3]
M1== [0.6 0.2]
```

```
updated value of m1 and m2 is:
M1== [0.2475 0.275 ]
M1== [0.1225 0.765 ]
```

```
```