**Assignment Ques 2 Machine Learning**

**Ques.** What is k-means algorithm? Explain with an example.

## **Ans**. **Introduction to K-Means Algorithm**

K-means clustering algorithm computes the centroids and iterates until we it finds optimal centroid. It assumes that the number of clusters are already known. It is also called **flat clustering** algorithm. The number of clusters identified from data by algorithm is represented by ‘K’ in K-means.

## **Working of K-Means Algorithm**

We can understand the working of K-Means clustering algorithm with the help of following steps −

**Step 1** − First, we need to specify the number of clusters, K, need to be generated by this algorithm.

**Step 2** − Next, randomly select K data points and assign each data point to a cluster. In simple words, classify the data based on the number of data points.

**Step 3** − Now it will compute the cluster centroids.

**Step 4** − Next, keep iterating the following until we find optimal centroid which is the assignment of data points to the clusters that are not changing any more

* **4.1** − First, the sum of squared distance between data points and centroids would be computed.
* **4.2** − Now, we have to assign each data point to the cluster that is closer than other cluster (centroid).
* **4.3** − At last compute the centroids for the clusters by taking the average of all data points of that cluster.

K-means follows **Expectation-Maximization** approach to solve the problem. The Expectation-step is used for assigning the data points to the closest cluster and the Maximization-step is used for computing the centroid of each cluster.

The following two examples of implementing K-Means clustering algorithm will help us in its better understanding −

### **Example 1**

Let us move to another example in which we are going to apply K-means clustering on simple digits dataset. K-means will try to identify similar digits without using the original label information.

First, we will start by importing the necessary packages −

%matplotlib inline

import matplotlib.pyplot as plt

import seaborn as sns; sns.set()

import numpy as np

from sklearn.cluster import KMeans

Next, load the digit dataset from sklearn and make an object of it. We can also find number of rows and columns in this dataset as follows −

from sklearn.datasets import load\_digits

digits = load\_digits()

digits.data.shape

### **Output**

(1797, 64)

The above output shows that this dataset is having 1797 samples with 64 features.

We can perform the clustering as we did in Example 1 above −

kmeans = KMeans(n\_clusters = 10, random\_state = 0)

clusters = kmeans.fit\_predict(digits.data)

kmeans.cluster\_centers\_.shape

### **Output**

(10, 64)

The above output shows that K-means created 10 clusters with 64 features.

fig, ax = plt.subplots(2, 5, figsize=(8, 3))

centers = kmeans.cluster\_centers\_.reshape(10, 8, 8)

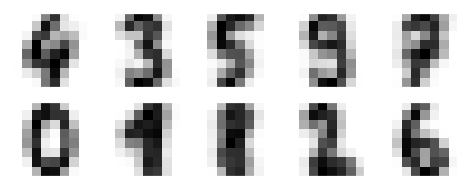
for axi, center in zip(ax.flat, centers):

axi.set(xticks=[], yticks=[])

axi.imshow(center, interpolation='nearest', cmap=plt.cm.binary)

### **Output**

As output, we will get following image showing clusters centers learned by k-means.



The following lines of code will match the learned cluster labels with the true labels found in them −

from scipy.stats import mode

labels = np.zeros\_like(clusters)

for i in range(10):

mask = (clusters == i)

labels[mask] = mode(digits.target[mask])[0]

Next, we can check the accuracy as follows −

from sklearn.metrics import accuracy\_score

accuracy\_score(digits.target, labels)

### **Output**

0.7935447968836951

The above output shows that the accuracy is around 80%.