Practical 4

Problem Statement:

Implement K-Means clustering on Iris.csv dataset. Determine the number of clusters using the elbow method.

Dataset: Iris.csv Dataset.

Dataset link: https://www.kaggle.com/datasets/uciml/iris

Objective:

1. Understand working of K-Means clustering algorithm on the Iris dataset
2. determine the optimal number of clusters using the elbow method

Outcome:

After implementing K-Means clustering on the Iris.csv dataset and using the elbow method, students will effectively determine the optimal number of clusters and understand the practical application of this clustering algorithm on real-world datasets.

Theory:

K Means algorithm is a centroid-based clustering (unsupervised) technique. This technique groups the dataset into k different clusters having an almost equal number of points. Each of the clusters has a centroid point which represents the mean of the data points lying in that cluster. The idea of the K-Means algorithm is to find k-centroid points and every point in the dataset will belong to either of the k-sets having minimum Euclidean distance.

Types of Support Vector Machine Algorithms

1. K-Means Clustering:

K-Means clustering is a partitioning method that divides a dataset into K distinct, non-overlapping clusters. The objective is to minimize the distance between data points within the same cluster and maximize the distance between clusters.

- **Strengths:**
  Simple and easy to implement.
  Scalable for large datasets.
- **Weaknesses:**
  The number of clusters K must be specified beforehand.
  Sensitive to initial centroid placement, which can lead to local optima.
  Assumes clusters are spherical and equally sized.

2. Elbow Method:

The Elbow method is a heuristic used in determining the optimal number of clusters K for the K-Means clustering algorithm. The method involves visualizing the variation explained as a function of the number of clusters, and picking the "elbow" of the curve as the number of clusters to use.

Key Points:

1.WCSS Calculation: For each value of K, compute the within-cluster sum of squares (WCSS). This represents the total distance of data points from their respective cluster centroids.
2.Plotting: Plot the curve of WCSS against the number of clusters K.
3.Elbow Identification: The point where the rate of decrease of WCSS sharply changes (representing an "elbow" in the graph) is considered an optimal value for K.

Rationale:
As the number of clusters increases, the total variance or WCSS will naturally decrease as data points will be closer to their respective centroids. However, after a certain number of clusters, the reduction in WCSS will be marginal, indicating that adding more clusters may not provide much additional value. The "elbow" of the curve represents the point of diminishing returns.

Limitations:

- The "elbow" may not always be clear and distinct.
- It's a heuristic, so it might not always yield the absolute optimal number of clusters.

Requirements:

1) Get the Iris dataset

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset.

2) Importing Libraries

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:
Matplotlib: The second library is matplotlib, which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot. This library is used to plot any type of charts in Python for the code. It will be imported as below:
import matplotlib.pyplot as mpt
Here we have used mpt as a short name for this library.
Seaborn: It is a library that uses Matplotlib underneath to plot graphs. It will be used to visualize random distributions
import seaborn as sns
Pandas: The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. It will be imported as below:
Here, we have used pd as a short name for this library. Consider the below image

```
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
```

3) Importing the Datasets

Now we need to import the datasets which we have collected for our machine learning project.

```
data = pd.read_csv('Iris.csv')### loading data
```

4) Data understanding and exploration

Here we understand and visualize missing values, if necessary, remove outliers, and converting data types as needed.

a) Display the data types of each column in the DataFrame

```
data.info()
```

b) Describe the distribution of your data

```
numbers.describe(include = 'all')
```

5) Data Preparation

a) scaling the features

```
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

6) Implementing K-Means Clustering:

```
from sklearn.cluster import KMeans

# Applying kmeans to the dataset
kmeans = KMeans(n_clusters=3, init='k-means++', random_state=42)
y_kmeans = kmeans.fit_predict(X)
```

7) Determining the Number of Clusters Using the Elbow Method:

```
import matplotlib.pyplot as plt

wcss = []  # Within-cluster sum of squares

for i in range(1, 11):

    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)

    kmeans.fit(X)

    wcss.append(kmeans.inertia_)
```

8) Visualizing the Clusters:

```
plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s=100, c='red',
label='Cluster 1')
plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], s=100, c='blue',
label='Cluster 2')
plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s=100, c='green',
label='Cluster 3')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=300,
c='yellow', label='Centroids'))
```

Algorithm:

K-Means Clustering

1. Initialization: Select K data points as initial centroids, either randomly or using some heuristic.
2. Assignment: Assign each data point to the nearest centroid, forming K clusters.
3. Update: Calculate the mean of all data points within each cluster and move the centroid to that mean location.
4. Convergence: Repeat the assignment and update steps until centroids no longer change, or a set number of iterations is reached.
The measure of "distance" is usually the Euclidean distance, but other distance metrics can also be used.

Elbow Method:

1. Computing the sum of squared distances from each point to its assigned center for different values of K. This is often referred to as within-cluster sum of squares (WCSS).
2. Plotting the curve of WCSS as a function of the number of clusters.
3. Observing the "elbow" point in the plot, where the rate of decrease sharply changes, indicating an optimal value for K.

The underlying idea is that increasing the number of clusters will naturally reduce the distance from data points to their respective centroids, leading to a decrease in WCSS. However, after a certain point, the reduction in WCSS becomes marginal, providing diminishing returns in clustering quality. This point, where adding another cluster doesn't give much better fit to the data, is the "elbow".

Conclusion:

We have learned about K Means clustering algorithm from scratch to implementation. First, we have seen clustering and then finding K value in K Means. For that we have seen Elbow method.

Oral Questions:

1. What is clustering?
2. Where is clustering used in real life?
3. When to use k-means and K medians?
4. What is difference between classification and clustering?