Chaitanya Machhindra Nawale
BFCOB215

ML Assignment No 1

PAGE NO.
DATE :

* Title :- To implement given problem using k-means clustering techniques.

* Objective :- To learn how to apply k-means clustering for given data points using SK-Learn

* Problem Statement :-
We have given a collection of 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-means clustering with initial centroids as m1 = p1 Cluster #1, and m2 = p6 Cluster #2.
Answer the following :-
1) Which cluster does p6 belong to ?
2) What is the population of cluster around m2?
3) What is updated value of m1 and m2 ?

* Outcomes :- Ability to apply k-means clustering technique to solve the given problem and find the optimal solution.

* Software & Hardware Requirement :-
Python 3, Jupyter Notebook, Scikit-learn.

* Theory :-
Clustering Basics
$x = \{x1', x2', \ldots, xn'\}$

Scanned by TapScanner

Where 'n' belongs to R^m
We assume that it's possible to find a criterion (not unique) so that each sample can be associated with a specific grouping.

$g_k = G(x_i)$ where $k = \{0, 1, \ldots, t\}$

Conventionally, each group is called a cluster and the process of finding the function G is called as clustering. Different clustering algorithms are based on alternative strategies to solve this problem, and can yield very different results.

K-means Clustering :-

The k-means algorithm is based on the conditions to decide the number of clusters through the assignment at k-initial centroids or means.

$$k^{(0)} = \{ u_1^{(0)}, u_2^{(0)}, \ldots, u_K^{(0)} \}$$

Then the distance between each sample and each centroid is compare and sample is assigned to the cluster where the distances is minimum. This approach is often called minimizing the inertia of the cluster, which is defined as

$$SS_{wi} = \sum_t \| x_t - u_i \|^2 \quad \forall i \in (1, k)$$

The process is iterate-once all the sample have been processed v One set of centroids is computed now considering the actual elements belonging to cluster and all the distances are computed. Consider the example with a

Scanned by TapScanner

PAGE NO.
DATE :

dummy dataset :-

```
From sklearn.cluster import KMeans
>>> km = KMeans (n_clusters=3)
>>> km.fit (X)
KMeans (algorithm='auto', copy_x=True,
init = 'k-means++', max_iter=300, n_clusters=3
n_init =10, n_jobs =1, precompute_distances = 'auto',
random_state = None, tol = 0.0001, vebose=0)
>>> print (km.cluster_centers_)
[[ 1.39014517, 1.38 5339937] [9.78
 [ 9.78 473454, 6.19 46332]
 [-5.47 807472, 3.73 913652] ]
```

Finding Optimal Number of clusters :-

One of the most disadvantages of k-means is related to the choice of optimal number of clusters. An excessively small value will determine large groupings that contain heterogeneous elements while a large number leads a scenario where it can be difficult to identify the difference among clusters.

Elbow method :-

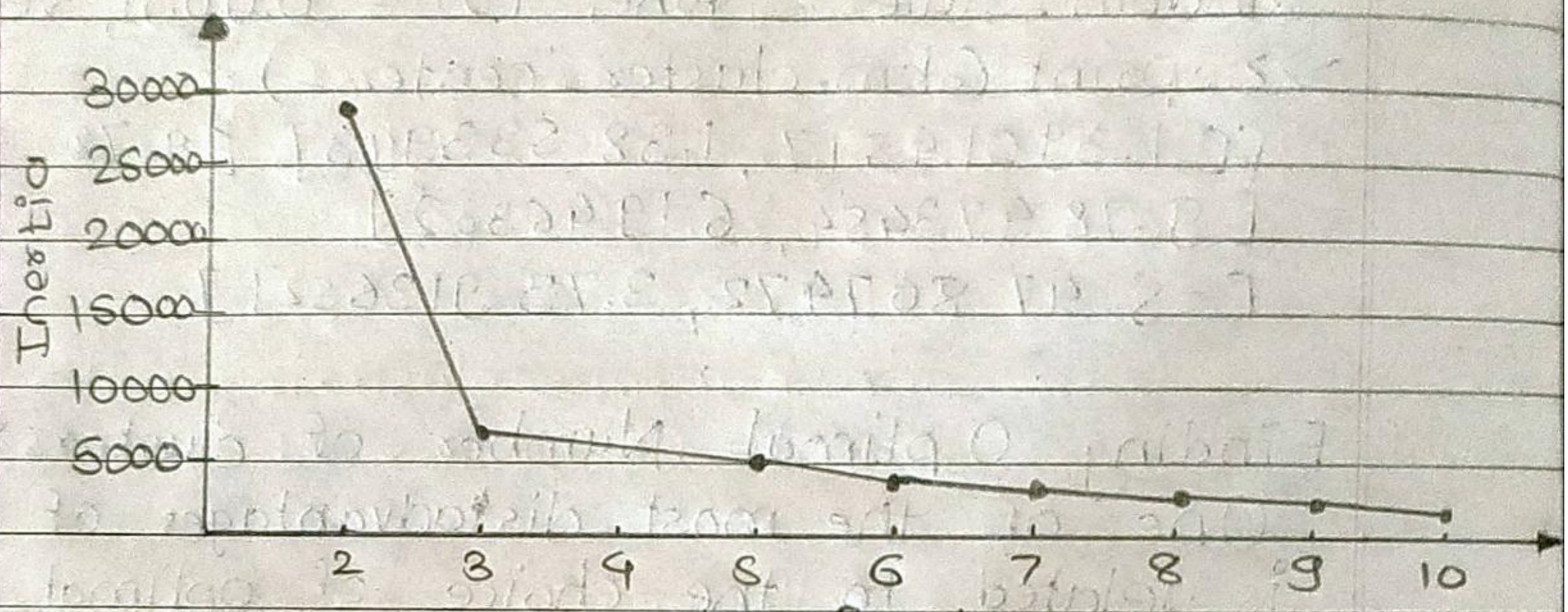
This method is based on the assumption that an approximate no. of clusters must produce a small inertia. However, this value reaches it minimum (OD) when no. of clusters is equal to no. of samples therefore we can't look for minimum but to a value which is tradeoff between inertia of clusters

Scanned by TapScanner

PAGE NO.
DATE :

```
>>> nbclusters =[2,3,4,5,6,7,8,9,10]
>>> inertias = []
for n in nb-clusters:
 km = Kmean (n_clusters =n)
 inertias append(km.inertia_)
```

Plotting the values, we get result as:



Scikit-learn datasets:

Scikit Learn provides some built-in datasets that can be used for testing purpose they are all available in the package scikit-learn. Datasets have common structure data instance variable contains the whole input set x, y target continuous the labels for classification

```
>>> from sklearn.datasets import load_iris
>>>  iris = load_iris()
>>> X = iris.data
>>> y = iris.target
>>> X.shape
    (150, 4)
```

Scanned by TapScanner

PAGE NO.
DATE :

```
>>> Y.shape
    (150,)
```

Scikit-learn also provides functions for creating dummy dataset from scratch: make_classification(), make_regression() make_blobs() they are very easy to use and its a best choice to test model without loading more complex dataset.

Conclusion : Thus we have successfully ~~implem~~ implemented given problem using k-means clustering technigque ~~to test model without loading more complex dataset~~ and used elbow method to find optimal number of clusters.

Scanned by TapScanner