****

**SUBJECT: PROGRAM STRUCTURE AND ALGORITHMS**

**TOPIC: K-MEANS CLUSTERING ARTICLE**

**NAME: MEGHANSHU BHATT**

**NUID: 002961585**

**K-Means Clustering Algorithm**

An unsupervised machine learning technique called k-means clustering is used to find groups of data objects in a dataset. Although there are many alternative clustering techniques, k-means is one of the most established and user-friendly. These characteristics make k-means clustering in Python relatively simple to implement, even for inexperienced programmers and data scientists.

**This article will teach you:**

* K-means clustering's definition
* When to analyze your data using k-means clustering
* How to use scikit-learn to construct k-means clustering in Python
* How to choose an appropriate amount of clusters

**What Is Clustering?**

Data can be divided into groups, or clusters, using a variety of processes called clustering. In a broad sense, clusters are collections of data objects that share more similarities with one another than with those in other clusters. Clustering aids in spotting two characteristics of data:

* Meaningfulness
* Usefulness

**Meaningful** clusters expand domain knowledge. For instance, scientists used clustering in gene expression studies in the medical field. The clustering findings revealed patient populations that react differentially to medicinal interventions.

**Useful** clusters, on the other hand, serve as an intermediate step in a data pipeline. Businesses, for instance, employ clustering to categorize their clientele. The consumer groups with comparable purchase histories that are created because of clustering can then be used by firms to develop targeted advertising campaigns.

There are numerous additional uses for clustering, including social network analysis and document clustering. Since these applications are applicable to almost every industry, clustering is a valuable ability for experts dealing with data in all industries.

**Clustering Techniques Overview**

There are a variety of methods you may use to perform clustering; there are even entire categories of clustering algorithms. These groups each have particular advantages and disadvantages. Accordingly, depending on the input data, some clustering algorithms will produce more logical cluster designations.

Because there are so many options, choosing the right clustering algorithm for your dataset can be challenging. The properties of the clusters, the dataset's attributes, the quantity of outliers, and the number of data objects are some significant variables that influence this choice.

By examining three common categories of clustering algorithms, you'll learn how these elements assist in determining which strategy is most suitable.

* Partitional clustering
* Hierarchical clustering
* Density-based clustering

Before diving into k-means, it is worthwhile to have a high-level look at these categories. To put k-means' position within the spectrum of clustering algorithms, you'll discover the advantages and disadvantages of each group.

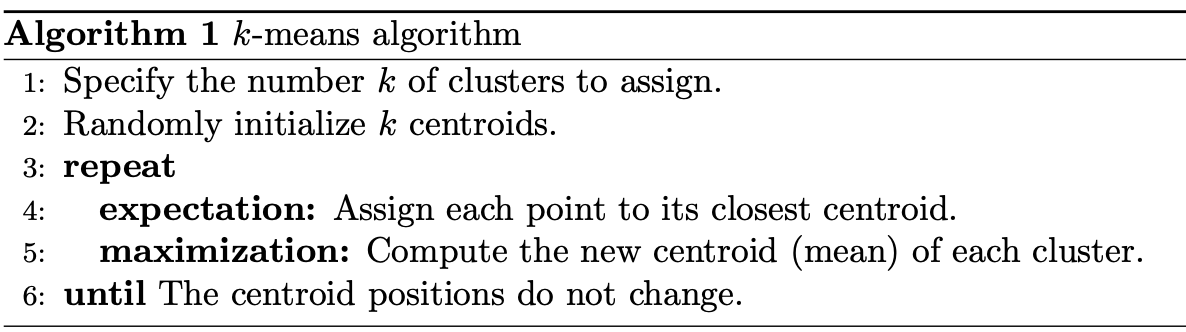
**How to implement K-Means Clustering**

You will be given a step-by-step walkthrough of the traditional k-means algorithm in this part. Writing a Python k-means clustering pipeline requires a fundamental understanding of the algorithm's workings. Your decision over whether to use k-means to solve your clustering problem will be aided by the knowledge you gain in this section.

**How to Interpret the K-Means Algorithm**

Traditional k-means just needs a few steps. First, choose k centroids at random, where k is equal to the number of clusters you want to use. The center of a cluster is represented by centroids, which are data points.

The algorithm's major component operates via a two-step procedure known as expectation-maximization. Each data point is assigned to the closest centroid during the expectation stage. The new centroid is then determined by computing the mean of all the points for each cluster during the maximizing step. The k-means algorithm in its traditional form looks like this:



After the centroids converge or match the assignment from the previous iteration, the quality of the cluster assignments is assessed by determining the sum of the squared errors (SSE). The squared Euclidean distances between each point and its nearest centroid are added up to form the SSE. Given that this is a measure of error, k-means seeks to reduce this value.

The centroids and SSE updates over the first five iterations of two separate runs of the k-means algorithm on the same dataset are shown in the figure below:

Chart, scatter chart

Description automatically generated

This diagram is meant to demonstrate how crucial it is to initialize the centroids. It also emphasizes the use of SSE as a benchmark for clustering effectiveness. The expectation-maximization stage is continued until the centroid locations attain convergence and remain unchanged after selecting a number of clusters and the initial centroids.

The k-means approach is nondeterministic because of the random initialization stage, which results in different cluster allocations if the process is applied twice to the same dataset. The complete k-means algorithm is typically initialized multiple times by researchers, and the initialization with the lowest SSE is used to determine the cluster assignments.