****

**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.

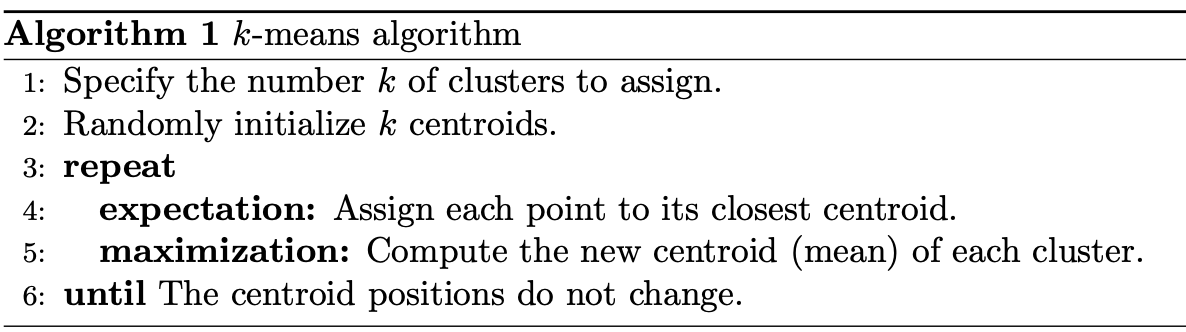
**How to implement K-Means Clustering**

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.

**Kmeans Algorithm**

The iterative Kmeans algorithm attempts to divide the dataset into K unique, non-overlapping subgroups (clusters), each of which contains a single data point. While keeping the clusters as distinct (far) apart as possible, it aims to make the intra-cluster data points as comparable as possible. It distributes data points to clusters in a way that minimizes the sum of the squared distances between the data points and the cluster centroid, which is the average value of all the data points in the cluster. The homogeneity (similarity) of the data points within a cluster increases as the amount of variance within the cluster decreases.The way kmeans algorithm works is as follows:

1. Specify number of clusters K.
2. Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement.
3. Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn’t changing.

* Compute the sum of the squared distance between data points and all centroids.
* Assign each data point to the closest cluster (centroid).
* Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.

Expectation-Maximization is the method that kmeans uses to tackle the issue. The data points are assigned to the closest cluster in the E-step. The centroid of each cluster is calculated in the M-step. Here is a breakdown of the mathematical procedure (feel free to skip it).

The purposeful action is:

Diagram

Description automatically generated with low confidence

if data point xi is a member of cluster k, then wik=1; else, wik=0. Additionally, k is the cluster's centroid for xi.

It is a two-part minimization issue. First, we fix k and decrease J w.r.t. wik. Then we treat wik fixed and minimize J w.r.t. Technically, we update cluster assignments after differentiating J w.r.t. wik (E-step). After recalculating the centroids based on the cluster assignments from the previous phase, we distinguish J w.r.t. k. (M-step). Consequently, E-step is:Text, letter

Description automatically generated

In other words, assign the data point xi to the closest cluster judged by its sum of squared distance from cluster’s centroid.

And M-step is:

A picture containing diagram

Description automatically generated

Which translates to recomputing the centroid of each cluster to reflect the new assignments.

Few things to note here:

* Since almost always the features in any dataset would have different units of measurements, such as age vs income, it is advised to standardize the data to have a mean of zero and a standard deviation of one. This is because clustering algorithms, including kmeans, use distance-based measurements to determine the similarity between data points.
* Different initializations may result in different clusters since the kmeans method may get trapped in a local optimum and not converge to a global optimum due to its iterative nature and the random initialization of centroids at the beginning of the algorithm. Therefore, it is advised to execute the method with several centroids' initializations and select the results of the run that produced the smallest sum of squared distance.
* Assignment of examples isn’t changing is the same thing as no change in within-cluster variation:

Background pattern

Description automatically generated with low confidence

**Evaluation Methods**

Clustering analysis lacks a reliable evaluation metric that we may use to compare the results of various clustering algorithms, in contrast to supervised learning where we have the ground truth to assess the model's performance. There is no correct answer in terms of how many clusters we should have in any problem because kmeans demands k as an input and doesn't learn it from data. Domain expertise and intuition may be helpful occasionally, although this is not frequently the case. Since clusters are employed in the downstream modeling in the cluster-predict methodology, we may assess how well the models are performing based on various K clusters.

In this article, we'll discuss two indicators that could help us understand k:

* Elbow method
* Silhouette analysis

# **Elbow Method**

**Elbow** method gives us an idea on what a good k number of clusters would be based on the sum of squared distance (SSE) between data points and their assigned clusters’ centroids. We pick k at the spot where SSE starts to flatten out and forming an elbow. We’ll use the geyser dataset and evaluate SSE for different values of k and see where the curve might form an elbow and flatten out.

Chart, line chart

Description automatically generated

# The graph demonstrates that k=2 is a reasonable choice. Because the curve is monotonically declining, it can be challenging to determine how many clusters to utilize because there may be no elbow or a clear point where the curve begins to flatten out.

# **Silhouette Analysis**

**Silhouette analysis** can be used to determine the degree of separation between clusters. For each sample:

* Compute the average distance from all data points in the same cluster (ai).
* Compute the average distance from all data points in the closest cluster (bi).
* Compute the coefficient:

A picture containing table

Description automatically generated

The coefficient can take values in the interval [-1, 1].

* If it is 0 –> the sample is very close to the neighboring clusters.
* It it is 1 –> the sample is far away from the neighboring clusters.
* It it is -1 –> the sample is assigned to the wrong clusters.

Therefore, we want the coefficients to be as big as possible and close to 1 to have a good clusters. We’ll use here geyser dataset again because its cheaper to run the silhouette analysis and it is actually obvious that there is most likely only two groups of data points.

A picture containing diagram

Description automatically generated

Chart

Description automatically generated

Chart, funnel chart

Description automatically generated

As the above plots show, n\_clusters=2 has the best average silhouette score of around 0.75 and all clusters being above the average shows that it is actually a good choice. Also, the thickness of the silhouette plot gives an indication of how big each cluster is. The plot shows that cluster 1 has almost double the samples than cluster 2. However, as we increased n\_clusters to 3 and 4, the average silhouette score decreased dramatically to around 0.48 and 0.39 respectively. Moreover, the thickness of silhouette plot started showing wide fluctuations. The bottom line is: Good n\_clusters will have a well above 0.5 silhouette average score as well as all of the clusters have higher than the average score.

# **Drawbacks**

If clusters resemble spheres, the Kmeans algorithm does a decent job at capturing the structure of the data. The centroid is always surrounded by a lovely, spherical form. This means that kmeans performs poorly at grouping the data the moment the clusters have a complex geometric shape. We'll give three examples of situations when kmeans will fall short.

First, even when two data points clearly belong in the same cluster, the kmeans algorithm forbids them from sharing a cluster. The data points on two separate horizontal lines shown below show how kmeans tries to group the half of each horizontal line's data points together.

Chart

Description automatically generated

Given that their shapes are non-spherical, Kmeans believes that point 'B' is closer to point 'A' than point 'C'. Points "A" and "B" will therefore be in the same cluster, however point "C" will be in a distinct cluster. Because it doesn't segregate related points, the Single Linkage hierarchical clustering approach gets this right (note).

Second, data from multivariate normal distributions with various means and standard deviations will be generated. As a result, there would be three groups of data, each with a different multivariate normal distribution (different mean/standard deviation). There will be one group with a lot more data points than the other two put together. The data will then be run using kmeans with K=3 to see if the data can be accurately clustered.To make the comparison easier, I am going to plot first the data colored based on the distribution it came from. Then I will plot the same data but now colored based on the clusters they have been assigned to.

Chart, scatter chart

Description automatically generated

Looks like kmeans couldn’t figure out the clusters correctly. Since it tries to minimize the within-cluster variation, it gives more weight to bigger clusters than smaller ones. In other words, data points in smaller clusters may be left away from the centroid in order to focus more on the larger cluster.

Last, we’ll generate data that have complicated geometric shapes such as moons and circles within each other and test kmeans on both of the datasets.

Chart, scatter chart

Description automatically generated

As expected, kmeans couldn’t figure out the correct clusters for both datasets. However, we can help kmeans perfectly cluster these kind of datasets if we use kernel methods. The idea is we transform to higher dimensional representation that make the data linearly separable (the same idea that we use in SVMs). Different kinds of algorithms work very well in such scenarios such as SpectralClustering, see below:

Chart, scatter chart

Description automatically generated

**Applications of K-means clustering:**

Almost every industry can benefit from the usage of K-means clustering, from banking to cyber security, document clustering to image segmentation. It is often used with data that is continuous, quantitative, and has fewer dimensions. Some of the K-means usage cases are listed below:

**Customer Segmentation**: Clustering aids marketers in expanding their customer base, focusing on key markets, and segmenting consumers according to past purchases, interests, or activity tracking. For particular campaigns, organizations can target particular clusters or groups of clients thanks to segmentation.

**Document Classification**: The goal of the k-means method is to group documents into numerous categories based on tags, subjects, and the content of the documents. This is a relatively common classification problem. In order to represent each document as a vector during the first processing of the documents, term frequency is employed to find frequently used terms that aid in document classification. The document vectors are subsequently grouped to reveal document group similarity.

**Identifying crime localities**: The type of crime, the location of the crime, and the relationship between the two can provide valuable insight into crime-prone areas within a city or locality when data on crimes is available for specific places in a metropolis.

**Delivery store optimization**: By combining k-means to determine the ideal number of launch sites and a genetic algorithm to resolve the truck route as a traveling salesman issue, it is possible to optimize the process of good delivery utilizing truck drones.

**Rideshare data analysis**: The publicly accessible Uber ride information collection offers a wealth of useful information about traffic, transit times, popular pickup locations, and more. Planning for future cities can be aided by understanding the traffic patterns in cities through analysis of this data.

**Call record detail analysis**: The data that telecom firms record about a customer's calls, messages, and online behavior is called a call detail record. When combined with demographic data about the customer, this information offers deeper insights into their demands. The K-means clustering technique can be used to understand client groupings based on their usage patterns over time.

**Insurance fraud detection**: Machine learning offers a wide range of uses in the identification of fraud in the auto, healthcare, and insurance industries. It is possible to separate new claims based on their proximity to clusters that suggest fraudulent tendencies using historical data on fraudulent claims. The ability to spot frauds is essential since insurance fraud might potentially cost a business millions of dollars.

**Cyber-Profiling criminals**: Cyber-profiling is the process of gathering information from people and groups in order to find meaningful relationships. The concept of cyber-profiling comes from criminal profiles, which give information to the investigative branch for categorizing the types of criminals who were at the crime scene, or from cyber-profiling individuals in a specific environment depending on the user data preferences.

**Automatic clustering of IT alerts**: Large company IT infrastructure technology components like network, storage, or databases produce a lot of alert signals; these messages need to be manually assessed for priority for subsequent operations because they could indicate operational concerns. Data clustering can help in failure prediction and can shed light on different alert types and mean times to repair.

K-Means By attempting to group comparable pixels in the image together and producing clusters, clustering can also be utilized for performing image segmentation. The several clusters that are created are the various items in an image. K-means clustering is also used in recommendation engines. For instance, in a music streaming app, similar genres or song types are clustered together for a user based on their listening habits, and the app can then suggest the songs that are the closest in similarity.

# **Conclusion**

Kmeans clustering is one of the most widely used clustering methods, and practitioners frequently use it as their first line of defense when tackling clustering problems to determine the general structure of the dataset. Data points are categorized using kmeans into unique, non-overlapping groupings. When the clusters are somewhat spherical in shape, it works quite well. It suffers, nevertheless, since cluster geometric shapes diverge from spherical shapes. Additionally, it requires a predefined number of clusters rather than learning it from the data. Knowing the underlying assumptions of algorithms and approaches will help you become a better practitioner by giving you a better understanding of the advantages and disadvantages of each technique. This will help you decide when to use each method and under what circumstances. In this post, we covered both strength, weaknesses, and some evaluation methods related to kmeans.

Below are the main takeaways:

* Scale/standardize the data when applying kmeans algorithm.
* Elbow method in selecting number of clusters doesn’t usually work because the error function is monotonically decreasing for all ks.
* Kmeans gives more weight to the bigger clusters.
* Kmeans assumes spherical shapes of clusters (with radius equal to the distance between the centroid and the furthest data point) and doesn’t work well when clusters are in different shapes such as elliptical clusters.
* If there is overlapping between clusters, kmeans doesn’t have an intrinsic measure for uncertainty for the examples belong to the overlapping region in order to determine for which cluster to assign each data point.
* Kmeans may still cluster the data even if it can’t be clustered such as data that comes from uniform distributions.