K-Means

The following content is based on existing documentation on the topics covered as well as the author's personal experience.

Written by Jhoeliel Palma Salazar | LinkedIn – Data Scientist

Application of the K-Means Machine Learning Model and Implementation Examples in Python

Introduction:

In the field of machine learning and data science, data clustering is a fundamental task that seeks to organize samples into coherent groups. The K-Means algorithm is widely used for this task due to its simplicity and efficiency in segmenting unlabeled datasets. In this article, we will explore the functioning of the K-Means algorithm and provide practical examples of its implementation in Python.

1. Fundamentals of the K-Means Algorithm:

The K-Means algorithm is an unsupervised clustering approach that aims to partition a dataset into K groups, where K is a predetermined value specified by the user. The objective is to minimize the variance within each group and maximize the difference between the groups. The clustering process is performed iteratively and consists of the following steps:

1.1. Centroid Initialization: K points are randomly selected as initial centroids, representing the starting centers of the groups.

1.2. Point Assignment to Groups: Each data point is assigned to the group whose centroid is the nearest, using a distance measure, typically the Euclidean distance.

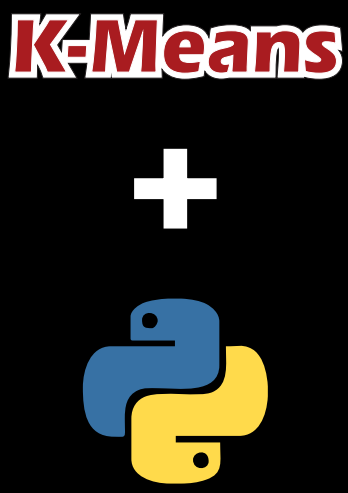
1.3. Centroid Update: The centroids of each group are recalculated as the average of all points assigned to that group.

1.4. Iteration: Steps 1.2 and 1.3 are repeated until the centroids converge or a maximum number of iterations is reached.

2. Example of Implementation in Python:

Below is an example of implementing the K-Means algorithm in Python using the scikit-learn library:

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs

# Generate example data
X, y = make_blobs(n_samples=300, centers=4, random_state=42)

# Visualize the example data
plt.scatter(X[:, 0], X[:, 1], s=50)
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title('Example Data Set')
plt.show()

# Create and fit the K-Means model
k = 4
kmeans = KMeans(n_clusters=k, random_state=42)
kmeans.fit(X)

# Get the labels and centroids
labels = kmeans.labels_
centroids = kmeans.cluster_centers_

# Visualize the clusters and found centroids
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
plt.scatter(centroids[:, 0], centroids[:, 1], marker='x', s=200, linewidths=2, c='red')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title('K-Means Clustering')
plt.show()
```



3. Complementary Feature of the K-Means Model: Dimensionality Reduction

In addition to its primary function of data clustering, the K-Means algorithm can also be used as a dimensionality reduction technique. This property is based on the fact that the centroids obtained at the end of the algorithm's convergence can be considered as a compressed representation of the original data. Instead of using the entire dataset, the centroids can be used as representative points to approximately reconstruct the original samples, thus reducing the dimensionality of the problem. This lesser-known feature can be beneficial in applications where feature reduction is essential for efficient analysis of high-dimensional data. However, it is important to note that, unlike other dimensionality reduction techniques, K-Means does not preserve a linear relationship with the original features and may not be suitable for all dimensionality reduction scenarios.

Conclusion:

The K-Means algorithm is a powerful tool for data clustering in unsupervised machine learning problems. Its implementation in Python, using the scikit-learn library, is straightforward and enables applying the algorithm to real datasets. K-Means' ability to find coherent clusters and its computational efficiency make it an attractive option for various applications in data science and the analysis of unlabeled data.

References:

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1(14), 281-297.
Scikit-learn. (2023). KMeans. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html.