UDK 519.237.8

STUDY OF THE FUZZY K -MEANS METHOD USING PYTHON PROGRAMMING LANGUAGE

Ruzibaev Ortik Baxtiyorovich
Dean of the faculty at the Tashkent University of Information Technologies named after Muhammad al-Khorezmy
Odiljonov Umidjon Odiljon ugli
Ibroximov Azizbek Zoxidjon ugli
Students at the Tashkent University of Information Technologies named after Muhammad al-Khorezmy

Abstract. Fuzzy K-Means clustering is an extension of the traditional K-Means algorithm that allows for soft clustering, where each data point is assigned a degree of membership to each cluster rather than being assigned to a single cluster. This article introduces the Fuzzy K-Means clustering algorithm and demonstrates its implementation in Python using the scikit-learn library. We explain the basic concepts of Fuzzy K-Means clustering, including the fuzzy partition matrix and the fuzziness parameter, and how it differs from traditional K-Means clustering. We demonstrate how to preprocess data, choose the optimal number of clusters, and visualize the results of Fuzzy K-Means clustering. We also discuss the advantages and disadvantages of Fuzzy K-Means clustering and compare it to other clustering algorithms. Finally, we provide some tips and tricks for improving the performance of Fuzzy K-Means clustering in Python.

Overall, this article aims to provide readers with a thorough understanding of Fuzzy K-Means clustering and practical knowledge of how to implement it in Python using the scikit-learn library.

Key words: Fuzzy c - means, k-means algorithm , Python, Fuzzy clustering, Data mining.

1. Introduction

Objective function-based clustering methods are used to minimize the distance between the sample and the cluster prototype and determine the parameters of the prototype - center or radius. In what follows, the prototype will be understood as the cluster center point. In such approaches, iterative crisp algorithms such as c-means are used to determine c partitions representing a set of objects. If the set of objects consists of compact clusters, and each cluster is reasonably separable from the others, the desired result can be obtained. Quite often, a set of objects contains several non-

prototype objects, which can lead to poor clustering results due to a shift in the centers of clusters.

To overcome this undesirable property of the fuzzy c-means algorithm, the FCM algorithm (fuzzy c-means algorithm) is used. This algorithm uses weight coefficients (membership functions) to control the contribution of objects to the determination of cluster centers. The FCM algorithm gives adequate clustering results in cases where the set of objects contains overlapping clusters. Clustering results are based on fuzzy membership functions using relative distances of objects relative to cluster centers. For example, an object located far from the cluster center contributes less to the cluster center search procedure than objects located close to the cluster center.

2. K-means method. Let $E=\{x_1, x_2,.....,x_k\}$ object set is given, where k-number of objects. Let, $\{V_1,V_2,....,V_c\}$ set of cluster centers, where c denotes the number of clusters. Method C-means minimize the function J- the sum of distances from vectors to cluster centers. The J function has the form:

$$J=\sum_{i=1}^{c}\sum_{k=1}^{n}\mu_{ik}d^2(x_k,V_i), \qquad (1)$$

where

- V_i - cluster center i.
- $\mu_{jk}=P(c/x_k)$: usually represents the probability of membership x_k k to the cluster c_j.
- $d=\|x_k-V_i\|$ - Euclidean norm, and V_i represents the center of the class i.

3. Fuzzy k-means method. Let given $E=\{x_1, x_2,.....,x_k\}$ objects set. Let, $\{V_1,V_2,....,V_c\}$ set of centers of clusters, where denotes the number of clusters. Degree of membership of the vector x_k to the cluster V_i denotes by $\mu_i(x_k)$. In this approach, each vector can belong to several clusters. U - membership matrix (also called C-fuzzy matrix section) whose size is $c*n$, where c - the number of classes and n the number of classified elements. To assess the quality of the partition, a scatter criterion is used that shows the sum of distances from objects to cluster centers with the corresponding degrees of membership:

$$J_m(U,V)=\sum_{i=1}^{c}\sum_{k=1}^{n}(\mu_{ik})^m d^2(x_k,V_i), \qquad (2)$$

where:

- $d=\|x_k-V_i\|$ - Euclidean norm, and V_i represents the center of the class i.

m - some real number greater than 1.

$\forall\ i,j\ \ \mu(x_k)\in\{0,1\}\ \ U=[\mu_{ik}(x_k)]$ (3)

$$U = \begin{pmatrix} \mu_{11} & \mu_{12} & \dots & \mu_{1n} \\ \mu_{21} & \mu_{22} & \dots & \mu_{2n} \\ \dots & \dots & \dots & \dots \\ \mu_{c1} & \mu_{c2} & \dots & \mu_{cn} \end{pmatrix} \quad (4)$$

For the membership degree matrix U following three constraints are defined:

$$0 \le \mu_{ik} \le 1, \quad 1 \le i \le c, 1 \le k \le n \quad (5)$$

$$\sum_{i=1}^{c} \mu_{ik} = 1, \ \forall \ k \in [1, n] \quad (6)$$

$$0 \le \sum_{i=1}^{c} \mu_{ik} \le 1, \forall \ i \in [1, c] \quad (7)$$

The fuzzy C-means method is based on updating the membership function. During iterations, the algorithm changes the divisions (U matrix), with function minimization J_m:

$$\mu_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{ik}}{d_{jk}}\right)^{\frac{2}{m-1}}}, \quad (8)$$

where:

- μ_{ik} - membership degree x_i to the cluster k.

Cluster center, V_i calculated by the formula:

$$V_i = \frac{\sum_{i=1}^{n} \mu_{ik}{}^{m} x_k}{\sum_{i=1}^{n} \mu_{ik}{}^{m}}, \quad (9)$$

The process stops when $\left| J_m{}^{t+1} - J_m{}^{t} \right| < \varepsilon$ or a predefined number of iterations has been reached.

The FCM algorithm involves the following steps:

STEP 1:

- • The number of classes is determined: c;
- Defined m,
- • Selecting a condition ε test stops.
- • Initialization ($T = 0$) matrix
- $U^t = 0$ with random values.
- Choice of distance d_{ik}.

STEP 2:

• Calculate new cluster centers using formula (9)

Step 3:

Renewe U^t by using:

$$\mu_{ik}^{t+1} = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{ik}}{d_{jk}}\right)^{\frac{2}{m-1}}} \quad . \qquad (8)$$

Compare J^{t+1} with J^{t}, if $\left|J_m^{t+1} - J_m^{t}\right| < \varepsilon$ the process stops, otherwise 9) $t \leftarrow t+1$ and return to step 2.

4. Implementation of the fuzzy c-means method in Python

```
from sklearn.cluster import KMeans
import numpy as np

def fuzzy_kmeans(X, k, m, max_iter=100, tol=1e-4):
    """
    Fuzzy K-Means clustering algorithm.

    Parameters
    ----------
    X : array-like, shape (n_samples, n_features)
        The input data.
    k : int
        The number of clusters to form.
    m : float
        Fuzziness parameter, must be greater than 1.
    max_iter : int, optional, default: 100
        Maximum number of iterations of the algorithm for a single run.
    tol : float, optional, default: 1e-4
        Tolerance for stopping criterion.

    Returns
    -------
    centroids : array, shape (k, n_features)
        Coordinates of cluster centers.
    U : array, shape (n_samples, k)
        Fuzzy partition matrix.
    """
    # Initialize centroids randomly
    centroids = X[np.random.choice(X.shape[0], k, replace=False)]

    # Initialize fuzzy partition matrix
    U = np.random.rand(X.shape[0], k)
    U = U / np.sum(U, axis=1)[:, np.newaxis]
```

```
    # Loop until convergence or maximum number of iterations
    for i in range(max_iter):
        # Calculate distances between each point and each centroid
        distances = np.linalg.norm(X[:, np.newaxis] - centroids, axis=2)

        # Update fuzzy partition matrix
        U_new = 1 / (distances ** (2/(m-1)))
        U_new = U_new / np.sum(U_new, axis=1)[:, np.newaxis]

        # Check for convergence
        if np.linalg.norm(U - U_new) < tol:
            break

        # Update centroids
        centroids = np.dot(U_new.T, X) / np.sum(U_new, axis=0)[:, np.newaxis]

        # Update fuzzy partition matrix
        U = U_new

    return centroids, U
```

Here, **X** is the input data, **k** is the number of clusters to form, **m** is the fuzziness parameter (must be greater than 1), **max_iter** is the maximum number of iterations of the algorithm for a single run (default is 100), and **tol** is the tolerance for the stopping criterion (default is 1e-4).

The function returns the coordinates of cluster centers in **centroids** and the fuzzy partition matrix in **U**.

5. Conclusion

In conclusion, Fuzzy K-Means clustering is a powerful algorithm that can handle complex data clustering problems that traditional K-Means cannot. By allowing for soft clustering, Fuzzy K-Means can provide more nuanced and flexible clustering results that better reflect the underlying structure of the data. In this article, we have provided an introduction to Fuzzy K-Means clustering and demonstrated its implementation in Python using the scikit-learn library. We have explained the fundamental concepts of Fuzzy K-Means, such as the fuzzy partition matrix and the fuzziness parameter, and have shown how to preprocess data, choose the optimal

number of clusters, and visualize the results of Fuzzy K-Means clustering. We have also discussed the advantages and limitations of Fuzzy K-Means clustering and compared it to other clustering algorithms.

Overall, Fuzzy K-Means clustering is a valuable tool in data analysis and machine learning and should be considered when faced with complex clustering problems. The implementation in Python using the scikit-learn library is straightforward and easy to use, allowing for quick and effective exploration of clustering solutions. By understanding the fundamental concepts of Fuzzy K-Means clustering and following best practices in implementation, practitioners can leverage the power of this algorithm to gain insights into their data and make informed decisions.

.

References

1. Zadeh, L. A. Fuzzy sets. // Information and Control. – 1965. – Vol. 8.

2. Rousseeuw J.P. Silhouettes: a graphical ai d to the interpretation and validation of cluster analysis // J. Comp. Appl. Math. — 1987. — Vol. 20 — P. 53–65.

3. Bezdek, J.C. Pattern Recognition With Fuzzy Objective Functional Algorithms. Ple-num Press, New York, 1981.

4. Asuncion A. and D.J. Newman: "UCI Machine Learning Repository", Irvine, CA: University of California, School of Information and Computer Science, 2007. [Online]. Available: http://archive.ics.uci.edu/ml/datasets.html.

5. 5. Christopher Centelle, Siu Lun Hong, Michael Georgiopoulos. Fuzzy Gap Statistics for Fuzzy c-Means. Proceedings of the 11th ISATED International Conference on Artificial Intelligence and Software Computing. Palma de Mallorca, Spain, 2007, pp. 68-73.

6. 6. Zeynel Jebeci1, Figen Yildiz. Comparison of K-Means and Fuzzy C-Means Algorithms on Different Cluster Structures // Journal of Agricultural Informatics (ISSN 2061-862X) 2015. Vol. 6, no. 3. pp. 13-23.

7. 7. Robert L. Cannon, Jitendra V. Dave, and James S. Bezdek. Efficient implementation of fuzzy c-means clustering algorithms. More detailed transactions on pattern analysis and machine intelligence. Volume. Pami-8, No. 2, March 1986. S. 248-255.