Lecture Note for K-Means

Wang Yueqi
Software Engineering 2018
Sun Yat-sen University
November 22, 2020
wangyq95@mail2.sysu.edu.cn

Abstract

Clustering is an unsupervised learning technique, which aims to cluster a set of objects so that objects in the same cluster are more similar to each other than objects in other clusters. The k-means clustering algorithm is one of the most popular and widely used clustering algorithms, which is based on centroid clustering. The k-means clustering algorithm seeks to minimize the average distance between objects in the same cluster. In this article, I will introduce more information about the clustering algorithm (most importantly the k-means algorithm). In addition, in order to better illustrate the k-means clustering algorithm, I will also introduce other clustering algorithms for comparison. I also performed some experiments, through which we can better understand the k-means clustering algorithm.

1. Introduction

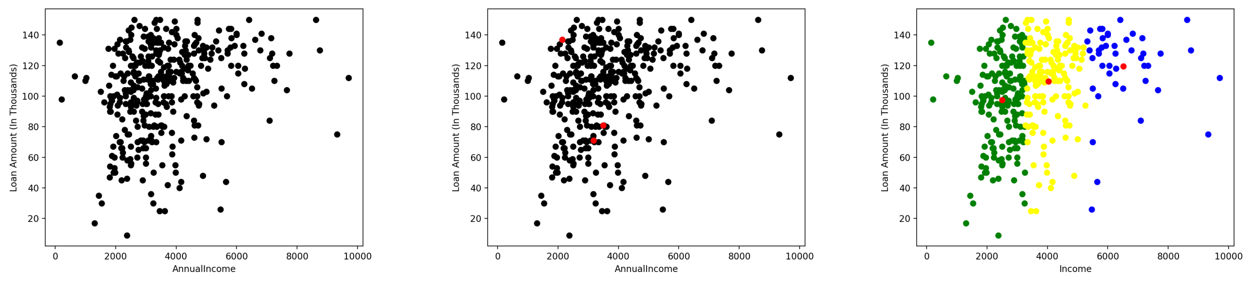
Clustering is one of the most important techniques in data mining and many other fields, which seek to group a set of objects so that objects in the same cluster are more similar to each other than objects in other clusters. Most clustering algorithms can be classified based on their clustering model. However, since there may be more than 100 published clustering algorithms, not all algorithms can be easily classified into exact categories, and there is no objectively correct clustering algorithm. Unless a clustering model is selected first for mathematical reasons, it is usually necessary to conduct experiments to choose the clustering algorithm that is most suitable for a particular problem. Clustering algorithms can be divided into hierarchical clustering, centroid-based clustering, distribution-based clustering, density-based clustering, etc. In this article, I will focus more on the k-means algorithm, which is the most popular one of the centroid-based clustering algorithms. According to Wikipedia [1], the concept of k-means was first used by James MacQueen in 1967, and then returned to Hugo Steinhaus in 1956. Although the standard k-means algorithm was not published in journals until 1982, it was originally proposed by Stuart Lloyd in 1957 as a technique for pulse-code modulation In 1965, Edward W. Forgy published the same method, which is why the k-means algorithm is sometimes called the Lloyd-Forgy algorithm.

Actually, k-means clustering is a vector quantization method, which originally started from signal processing. Its purpose is to divide n objects into k clusters, where each objects belongs to the cluster with the closest average (cluster center or cluster centroid) as the cluster prototype. There are a number of prototype-based clustering techniques, but two of the most prominent are k-means and k-medoid. The k-means defines the prototype by the centroid, which is usually the mean of a set of points, and is usually applied to objects in a continuous n-dimensional space. K-medoid uses medoid to define the prototype. The medoid is the most representative point in a set of points, and since it only requires the proximity of a pair of objects, it can be applied to a wide range of data. [2]

In this article, I will mainly focus on the k-means clustering algorithm. First introduce the concept and principle of k-means, and then analyze its specific algorithm steps. Also, a brief analysis of the advantages and disadvantages of k-means will be carried out. In addition, I will briefly introduce some other clustering algorithms for comparison. I also performed some experiments to verify the feasibility of k-means, and compare it with other clustering algorithms.

The main structure of this article is as follows:

In Section 2, I introduce the related work of k-means clustering algorithm and discuss clustering analysis as well as some concepts which will be used frequently. Section 3 is about k-means clustering algorithm. I use Section 4 to make a brief introduction of its applications. Next, Section 5 is to introduce other popular clustering algorithms. Section 6 is about experiments, related settings, and results to further illustrate k-means. Finally, I use Section 7 to make a brief conclusion and review the studies.



(a) Initialization (b) Randomly select 3 centroids (c) After k-means cluster

Figure 1. An example of k-means

2. Related work

K-means is a fundamental problem in machine learning, and has a rich history. The k-means algorithm was first been defined exactly by Inaba et al. [3], with the running time of $O(n^{kd})$. Since then, a number of polynomial time approximation schemes have been developed. Kanungo et al. [4] proposed an algorithm with the running time of $O(n^3\epsilon^{-d})$ and is $(9+\epsilon)$-competitive. Kanungo et al. also proposed a way of combining their work with Lloyd's algorithm. D. Arthur et al. [5] proposed a way of initializing k-means by choosing random starting centers with very specific probabilities. Mettu et al. [6] attempted to minimize the average distance to cluster centers. And they proposed a simple but powerful sampling technique called successive sampling, which can rapidly identify a small set of points.

2.1. Clustering Analysis

The concept of cluster cannot be precisely defined, which is one of the reasons why there are so many clustering algorithms. A variety of clustering algorithms have been developed in many fields for different types of applications. Nor algorithm or model is appropriate for all types of data, clusters and applications. Different researchers use different clustering models, and for each of these clustering models, different algorithms can be given again. Understanding these cluster models is the key to understanding the differences among various algorithms.

Clustering analysis is the task of grouping a set of objects so that objects in the same cluster are more similar to each other than objects in other clusters. Clustering analysis itself is not a specific algorithm, but a general task to be solved. It can be implemented through various algorithms, which differ greatly in understanding what constitutes a cluster and how to find them efficiently.

The rest part of this section focuses on the issues related to the different types of clustering algorithm. Through the discussion of various types of clustering algorithm, we could further understand the key issues of cluster analysis.

2.1.1 Connectivity-Based Clustering

In connectivity-based clustering, which also known as hierarchical clustering, clusters are formed by the relationship between data points. Connectivity-based clustering algorithms connect objects to form clusters based on distance. The cluster can be roughly described by the maximum distance required to connect the parts of the cluster. At different distances, different clusters will be formed.

Hierarchical clustering techniques are very important in clustering methods. Like k-means, these methods have a relatively long history compared to many other clustering algorithms that are still widely used. In order to generate hierarchical clusters, two basic methods are usually used:

- **Agglomerative:** This is a "bottom-up" approach: starting with the points as individual clusters, and merge the closest pair of clusters at each step.
- **Divisive:** This is a "top-down" approach: start with a cluster that contains everything, and then split the cluster until each step is left with only singleton clusters of individual points.

Agglomerative clustering works in a "bottom-up" manner, which means each point is initially treated as a single element cluster (leaf). In each step of the algorithm, the two most similar clusters are merged into a new cluster (node). Repeat this process until all points are members of only one cluster, which is the root.

Divisive is the inverse of agglomerative clustering, which works in a "top-down" manner. It starts from the root, where all points are contained in one single cluster. In each step of iteration, the most heterogeneous cluster is divided into two. Repeat this process until all points are in their respective clusters.

2.1.2 Centroid-Based Clustering

In centroid-based clustering, clusters are represented by their centroids. When the number of clusters is fixed to k, the k-means clustering algorithm can be formally defined as: find the k cluster centroids and assign other data points to the nearest centroids to minimize the squared distances from the cluster.

Most k-means-type algorithms require the number of clusters - k - to be specified in advance, which is considered to be one of the biggest shortcomings of these algorithms. In addition, because algorithms always assign objects to the nearest centroid, they tend to use clusters of similar size. This usually results in incorrectly cutting the boundaries of the cluster.

2.1.3 Distribution-Based Clustering

In distribution-based clustering, it is assumed that data consists of distributions, such as Gaussian distribution. Directly, clusters can be divided by the distributions to which objects are most likely to belong. This method is very similar to the artificial generation of data: it is done by sampling random data from the distribution.

Although this method seems to be excellent, it will encounter some problems, specifically overfitting. Overfitting is a modeling error that occurs when the clustering result is too closely fit to a limited set of data points. This is easy to understand that more complex clustering result will be able to interpret the data better, which is the reason why choosing the correct method is difficult. Another problem is that there may not be a precisely defined mathematical model in real world scenario, which causes additional burdens.

2.1.4 Density-Based Clustering

In density-based clustering, a cluster is determined by density, and areas with high density are grouped into the same cluster. Corresponding, data points in sparse areas are considered as noise or border points. As long as it can connect dense areas, it can be distributed in any shape.

2.1.5 Grid-Based Clustering

The grid-based technique is used for a multi-dimensional dataset. Grid-based clustering scans the datasets, divides the data space into several grid cells according to selected attributed, and divides the sample points into corresponding cells, and finally forms clusters according to the density of the cells. This method is more fast and effective in low-dimensional space. There are two types of grid-based clustering methods: STING and CLIQUE. The specific steps of the grid-based clustering algorithm are as follows:

1. Partition data space into a finite number of cells.
2. Randomly select a cell, which should not be traversed beforehand.
3. Calculate the cell density this cell.
4. If the density of this cell is greater than threshold density
   (a) Mark this cell as a new cluster
   (b) Calculate the density of all the neighboring cells of this cell
   (c) If the density of a neighboring cell is greater than threshold density, then add the cell in the cluster and repeat steps 4.2 and 4.3 till there is no neighbor with a density greater than threshold density
5. Repeat steps 2,3 and 4 till all the cells are traversed.

3. K-means Algorithm

3.1. The Basic K-means Algorithm

To begin with, I will introduce the basic k-means clustering, which is very simple. We first select K initial centroids, where K is a parameter of our own choice, that is, the required number of clusters. And we assign each point to its nearest centroid. As a result, the cluster is consist of a group of points assigned to the centroid. Next, we update the centroid of each cluster based on the points assigned to the cluster. Repeat this procedure until no point change clusters. Detailed k-means algorithm is described as Algorithm 1. There comes another problem that since most of the convergence occurs in early steps, it is common to replace the conditions in the last line of Algorithm 1 with a weaker condition, for example, repeat until only 1% of the points change clusters.

Algorithm 1 Basic K-means Algorithm

```
Select $K$ points as initial centroids.
repeat
    From $K$ clusters by assigning each point to its closest
    centroid.
    Recompute the centroid of each cluster.
until Centroids do not change.
```

Next, I will formally define the k-means problem. For k-means problem, an integer k and a set of n data points $x \in X$ are given, and we wish to choose k centroids C so as to minimize the objective function: $\phi = \sum_{x \in X} \min_{c \in C} ||x - c||^2$. Then we can define a cluster by grouping points according to which centroid is assigned to. And Algorithm 2 gives a more specific version of k-means algorithm.

An example of k-means is shown in Figure 1. The dataset used here is a Loan Prediction dataset [7]. Figure 1(a) shows all the points in this dataset, then in Figure 1(b), I set $K = 3$ to randomly select 3 centroids. Figure 1(c)

Algorithm 2 Standard K-means Algorithm

```
Given an initial set of $k$ means (centroids) $C = c_1, c_2, ..., c_k$
repeat
    for each $i \in 1, ..., k$ do
        set the cluster $C_i$ to be the set of points in $X$ that are closer to $c_i$ than they are to $c_j, \forall j \neq i$.
        set $c_i$ to be the centroid of all points in $C_i$ : $c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$.
    end for
until Centroids $C$ no longer change.
```

shows the final result after several rounds of k-means clustering. We can see clearly all the points are divided into 3 parts and assigned to a centroid.

I will introduce each of the steps in the basic k-means algorithm with more details.

3.1.1 Choosing Initial Centroids

Choosing the suitable initial centroids is a key step of the basic k-means algorithm. In short, we randomly choose the initial centroids, but the resulting clusters are usually not good. Similarly, the number of clusters is also important, which will affect the final cluster. Here, I introduce other methods of choosing centroids, which are also widely used.

- Choose K points with the distance as far as possible. First, randomly select a point as the first initial cluster center point, then select the point furthest from the point as the second initial cluster center point, and then select the point with the largest distance from the first two points as the closest point. The center point of the third initial cluster, and so on, until K initial cluster center points are selected.

- Use hierarchical clustering or Canopy algorithm.The commonly used hierarchical clustering algorithms are BIRCH and ROCK, which will not be introduced here. Canopy does not require the parameter K to do the initialization. Although Canopy is relatively inferior in accuracy compared to other algorithm, it converge very quickly. Therefore, Canopy clustering can be used to perform roughly clustering, and then obtain the parameter K first. With given K, k-means will further complete the clustering.

After years of development, a new method of initializing k-means called k-means++ has been developed, which can ensure that the initialization of the centroids is more intelligent and improve the quality of clustering. Except for initialization, the rest of the k-means++ algorithm is the same as k-means. Specific steps are described as follows:

- Randomly select the first centroid.

- For each point compute its distance from the nearest, previously chosen centroid.

- Select the next centroid from the points such that the probability of choosing a point as centroid is directily proportional to its distance from the nearest, previously chosen centroid.

- Repeat step 2 and 3 until k centroids have been sampled.

3.1.2 Assigning Points to the Closest Centroid

To assign points to the closest centroid, we need to measure the distance between points. Manhattan distance (L1) can be used for Euclidean data, while the Jaccard measure is usually used for documents. Euclidean distance (L2) is often used for data points in Euclidean space, while cosine similarity is more suitable for documents. The formal definitions of these methods are as follows:

$$Manhattan(x, y) = \sum_{i=1}^{n} |x_i - y_i| \tag{1}$$

$$Jaccard(A, B) = \frac{|A \cup B| - |A \cap B|}{A \cup B} \tag{2}$$

$$EuclideanDistance = \sqrt{\sum_{n}^{i=1} (x_i - y_j)^2} \tag{3}$$

$$CosineSimilarity = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \sqrt{\sum_{i=1}^{n} (B_i)^2}} \tag{4}$$

3.1.3 Centroids and Objective Functions

The step of "recompute the centroid of each cluster" depends on the proximity measurement for the data and the clustering goal. The goal of the clustering can be expressed as objective function, which depends on the distance between these points and the centroid of the cluster.

In basic k-means algorithm, lots of options can be used for the centroid and objective function that are guaranteed to converge. Table 1 shows some possible choices.

3.2. Strengths and Weaknesses of K-means

K-means algorithm is efficient and quite simple to implement. Similarly, k-means can be used for multiple data types, can be extended to large datasets, and can guarantee convergence simultaneously. In addition, k-means generalizes to clusters of different shapes and sizes.

Table 1. Choices for proximity, centroids, and objective functions.

Proximity Function	Centroid	Objective Function
Manhattan (L1)	median	Minimize sum of the L1 distance of an object to its cluster centroid
Squared Euclidean (L22)	mean	Minimize sum of the squared L2 distance of an object to its cluster centroid
cosine	mean	Maximize sum of the cosine similarity of an object to its cluster centroid
Bregman divergence	mean	Minimize sum of the Bregman divergence of an object to its cluster centroid

However, k-means algorithm is not suitable for all types of data, such as non-globular clusters or clusters of different sizes and densities. K-means is also difficult to deal with data that contains outliers. An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. In order to solve outliers, the detection and elimination of outliers may be significant. Similarly, when the number of dimension increases, the performance of k-means is not very suitable.

Another limitation of k-means is its cluster model. The concept is based on spherical clusters that are separable so that the mean converges toward the cluster center, which makes k-means perform better on only specific data distribution types. For example, in two half-moon classification, the k-means algorithm converges to an inferior result, which will be shown in the experimental section.

In addition, since the parameter of clusters K is a predefined number, an unsuitable selection of K will generate poor results. And like many other algorithms, once the k-means algorithm converges to a local minimum, it would generate erroneous results.

4. Applications

The k-means algorithm is very popular and widely used in various applications. Due to the long history of k-means, it has already been successfully used in data mining, computer vision, image segmentation and many other field. Generally, it is used as a preprocessing step for other algorithms to find the initial configuration. In this section, only some of its application will be introduced, and I won't go into details here.

As I mentioned above, because k-means is derived from signal processing, one of the most significant applications of k-means is vector quantization, and k-means is still used in this field now. Vector quantization is a classic quantization, and its working principle is to divide a large set of vectors into groups with a similar number of closest points. Each group is represented by its centroid, which is the reason why k-means is appropriate in vector quantization. The k-means algorithm is very suitable in dealing with color quantization and can produce competitive results. The purpose of color quantization is to reduce the image palette of an image to a fixed number of colors.

Another important application of k-means is image segmentation. Image segmentation is the task of dividing an image into multiple segments, which aims to simplify the image for easy analysis. In order to complete this work, it is obvious that k-means can be used. The k-means algorithm could help divide the digital image into different clusters (segments).

Moreover, in both supervised learning and unsupervised learning, the k-means algorithm has been used as a feature learning step. Feature learning is an important field in machine learning, which allows a system to find representations on its own. However, defining specific feature by using the real-world data is very complicated. Therefore, it is important to use some other algorithms (such as k-means) to help discover features or representations. For example, after k clusters of a group of unlabeled input data are generated by the k-means algorithm, the centroid of these clusters can be used to generated features. This application has already been successfully used in actual usage.

5. DBSCAN

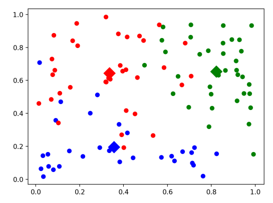
In this section, I will introduce DBSCAN clustering algorithm which is also popular and widely used. Because in the subsequent experimental section, I will compare the DBSCAN algorithm and the k-means algorithm, this part will be a little more detailed.

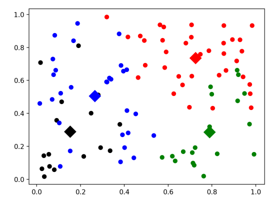
Density-based spatial clustering of applications with noise (DBSCAN) is also a clustering algorithm proposed by Ester et al. [8] It is a simple and effective density-based clustering non-parametric algorithm, but it illustrates concepts that are very significant for any density-based clustering method.

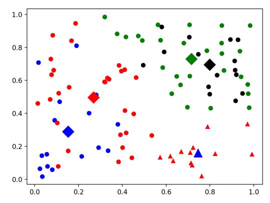
In order to describe the DBSCAN algorithm, some definitions should be given first.

- **Core points:** A point is a core point if there are at least MinPts within a distance of Eps, where MinPts and Eps are parameters given by the user.
- **Border points:** A border point is not the core point, but falls near the core point. A border point can near several core points.
- **Noise points:** A noise point is any point that is neither a core point nor a border point.

The detailed steps of DBSCAN are as follows:


(a) k=3, inertia = 12.9215


(b) k=4, inertia = 10.0651


(c) k=5, inertia = 8.2758

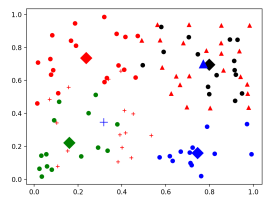
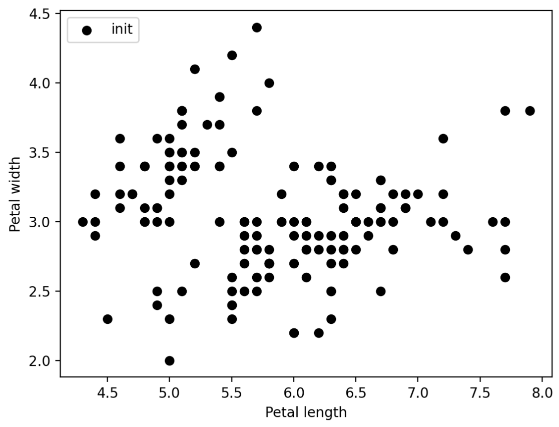
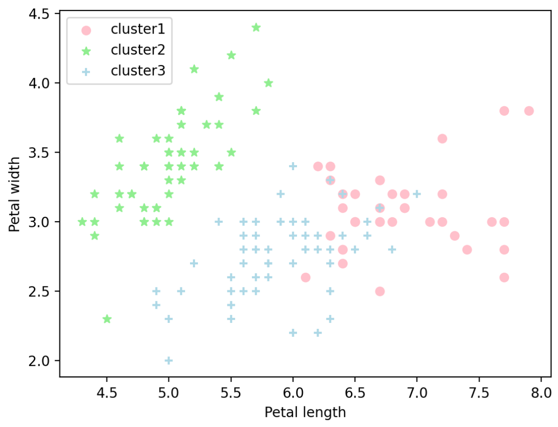

(d) k=6, inertia = 6.5751

Figure 2. K-means on random dataset



(a) Iris dataset



(b) Iris after k-means

Figure 3. K-means on Iris dataset

- Mark all points as core, border, or noise points.
- Eliminate noise points.
- Keep the edges between all core points at Eps distance from each other.
- Divide each group of connected core points into a separate cluster.
- Assign each border point to one of its associated core point clusters.

Because DBSCAN uses a density-based definition of a cluster, it is relatively resistant to noise and can handle clusters of arbitrary shapes and sizes. Thus, DBSCAN can find many clusters that could not be found using K-means.

However, when the density of the cluster changes severely, DBSCAN may run into trouble. High-dimensional data is also troublesome because it is difficult to define density for such data. In addition, when the nearest neighbor calculation needs to calculate all paired proximity, DBSCAN costs more, which is usually the case for high-dimensional data.

6. Experiments

In this section, I will demonstrate the implementation and performance of k-means algorithm through some experiments. All methods are implemented by python 3.8.3 with some related libraries.

6.1. K-means

To begin with, I performed a simple experiment. In this experiment, I randomly generated a dataset with a sample of 100 and a feature number 3. Then I used k-means method to cluster these data. For comparison purposes, I set the parameter k with 3, 4, 5 and 6. Figure 2 shows the results. It is obvious that Figure 2(a) works best when $k = 3$, which fits the settings.

We should notice that although inertia measures how well a dataset was cluster by k-means, it cannot be used as the only evaluation of k-means' performance. Because it is calculated by measuring the distance between each data point and its centroid, squaring the distance, and summing these squares across one cluster. As k increases, the inertia would also increase, which does not mean this cluster result is the best.

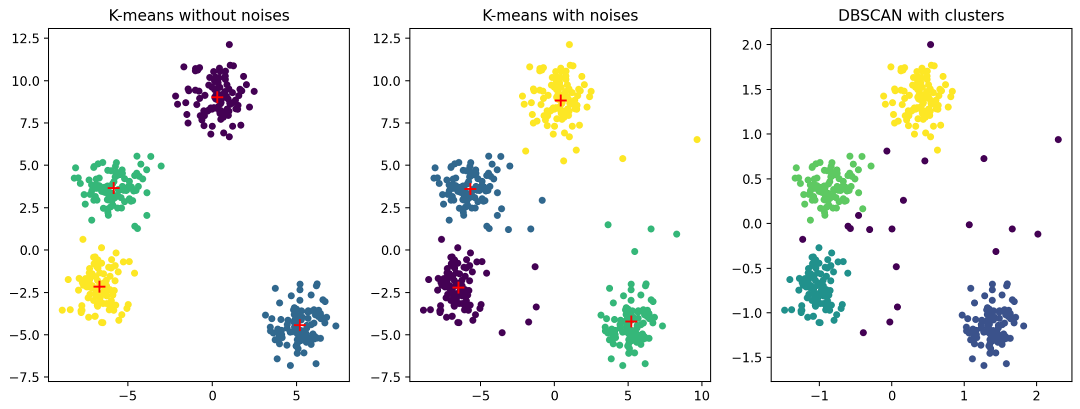


Figure 4. K-means and DBSCAN with noises

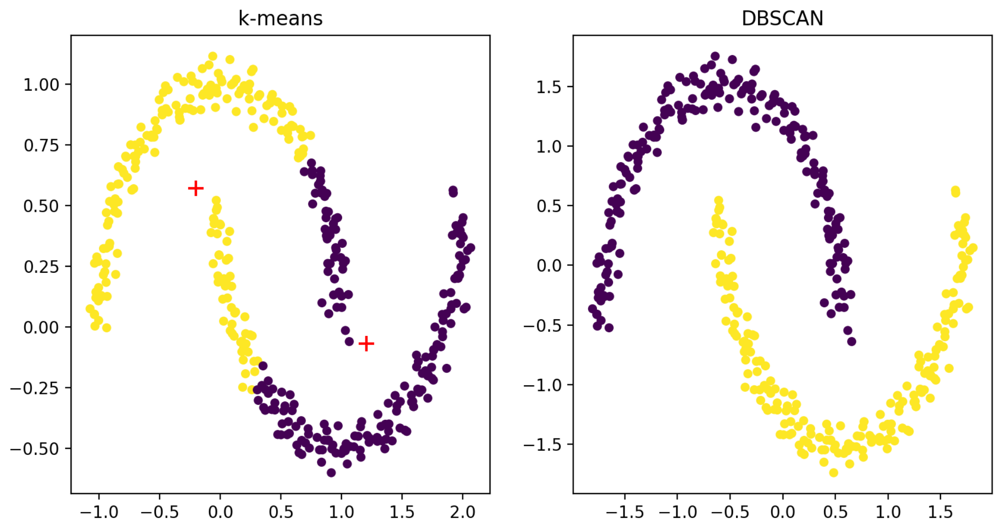


Figure 5. K-means and DBSCAN in moon dataset

In the next experiment, I run the k-means on Iris dataset to classify 3 classes of flowers using the features of flowers petal-length and petal-width. The result is shown as in Figure 3.

6.2. Comparison between K-means and DBSCAN

As I mentioned above, a big flaw of k-means is that it cannot distinguish which are noise or outliers. It can only assign each data point to a category Therefore, in this experiment, 20 noise points are added to the 4 clusters of 400 Gaussian sample points with a standard deviation of 1. While DBSCAN is a density-based algorithm, which can find noise points. The experiment compares the k-means clustering results with and without noises, as well as the clustering results of DBSCAN with noises. The results are shown in Figure 4, where the purple points are marked as noises. The results is obvious. Adding noises leads to a decline in the k-means clustering.

Another comparison experiment between k-means and DBSCAN focuses on the shape of sample points. The k-means algorithms has a good performance on convex data, which can be divided in to spherical clusters based on distance. But for datasets with non-convex shapes, such as circular data, DBSCAN is more suitable as density-based clustering algorithm. The results are shown in Figure 5.

7. Conclusion

In this article, I first introduced clustering algorithms and the classification of clustering algorithms. Then I mainly introduced the k-means clustering algorithm, which is a kind of centroid-based clustering algorithm. In addition to the introduction to the principle of the k-means clustering algorithm, I also discussed the specific operations of each step in the k-means clustering algorithm. Next, I briefly introduce the application of k-means algorithm and some other popular clustering algorithms. Finally, I did experiments related to the k-means algorithm to show the feasibility of the k-means algorithm and its clustering process.

References

[1] "K-means clustering," https://en.wikipedia.org/wiki/K-means_clustering.

[2] V. K. Pang-Ning Tan, *Introduction To Data Mining*. Pearson, 2005.

[3] H. Brönnimann and M. T. Goodrich, "Almost optimal set covers in finite vc-dimension: (preliminary version)," in *Proceedings of the Tenth Annual Symposium on Computational Geometry*, ser. SCG '94. New York, NY, USA: Association for Computing Machinery, 1994, p. 293–302. [Online]. Available: https://doi.org/10.1145/177424.178029

[4] KANUNGO and T, "A local search approximation algorithm for k-means clustering*1," *Computational Geometry*, vol. 28, no. 2-3, pp. 89–112, 2004.

[5] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, 2007.

[6] R. R. Mettu and C. G. Plaxton, "Optimal time bounds for approximate clustering: Theoretical advances in data clustering (guest editors: Nina mishra and rajeev motwani)," *Machine Learning*, vol. 56, 2004.

[7] "Loan prediction," https://datahack.analyticsvidhya.com/contest/practice-problem-loan-prediction-iii/?utm_source=blog&utm_medium=comprehensive-guide-k-means-clustering.

[8] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.