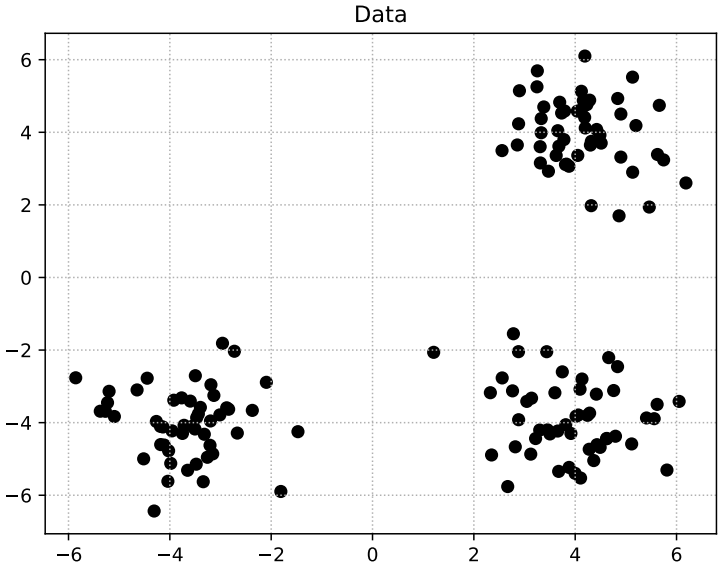
Clustering

Daniil Merkulov

Introduction to Data Science. Skoltech

Idea

General problem



Intuition with k-means

K-means

Algorithm

The k-means algorithm is a popular clustering method used to partition n data points into k clusters. Each data point belongs to the cluster with the closest mean.

1. **Initialization**: Randomly select k data points (or seed them in some other manner) to serve as the initial centroids.

Algorithm

The k-means algorithm is a popular clustering method used to partition n data points into k clusters. Each data point belongs to the cluster with the closest mean.

1. **Initialization**: Randomly select k data points (or seed them in some other manner) to serve as the initial centroids.
2. **Assignment**: Assign each data point to the nearest centroid. The distance is typically computed using the Euclidean distance, though other metrics can be used. Mathematically, assign each data point x_i to the nearest centroid c_j using the formula:

$$s(i) = \text{argmin}_j \|x_i - c_j\|^2$$

where $s(i)$ is the cluster to which data point x_i is assigned.

Algorithm

The k-means algorithm is a popular clustering method used to partition n data points into k clusters. Each data point belongs to the cluster with the closest mean.

1. **Initialization**: Randomly select k data points (or seed them in some other manner) to serve as the initial centroids.
2. **Assignment**: Assign each data point to the nearest centroid. The distance is typically computed using the Euclidean distance, though other metrics can be used. Mathematically, assign each data point x_i to the nearest centroid c_j using the formula:
$$s(i) = \text{argmin}_j \|x_i - c_j\|^2$$
where $s(i)$ is the cluster to which data point x_i is assigned.
3. **Update**: Recompute the centroid of each cluster as the mean of all points currently assigned to that cluster. For each cluster j:
$$c_j = \frac{1}{|S(j)|} \sum_{i \in S(j)} x_i$$
where $S(j)$ is the set of data points assigned to cluster j.

Algorithm

The k-means algorithm is a popular clustering method used to partition n data points into k clusters. Each data point belongs to the cluster with the closest mean.

1. **Initialization**: Randomly select k data points (or seed them in some other manner) to serve as the initial centroids.
2. **Assignment**: Assign each data point to the nearest centroid. The distance is typically computed using the Euclidean distance, though other metrics can be used. Mathematically, assign each data point x_i to the nearest centroid c_j using the formula:
$$s(i) = \text{argmin}_j \|x_i - c_j\|^2$$
where $s(i)$ is the cluster to which data point x_i is assigned.
3. **Update**: Recompute the centroid of each cluster as the mean of all points currently assigned to that cluster. For each cluster j:
$$c_j = \frac{1}{|S(j)|} \sum_{i \in S(j)} x_i$$
where $S(j)$ is the set of data points assigned to cluster j.
4. **Convergence**: Repeat steps 2 and 3 until the centroids no longer change significantly or some other stopping criteria is met.

Algorithm

Objective Function:

The k-means algorithm aims to minimize the within-cluster sum of squares (WCSS), given by:

$$J = \sum_{j=1}^{k} \sum_{i \in S(j)} \|x_i - c_j\|^2$$

Where: - J is the objective function value (WCSS). - k is the number of clusters. - x_i is a data point. - c_j is the centroid of cluster j.

The goal of the k-means algorithm is to find the set of centroids $\{c_1, c_2, ..., c_k\}$ that minimize J.

Notes:

- The k-means algorithm does not guarantee a global optimum solution. The final result might depend on the initial centroids.

Algorithm

Objective Function:

The k-means algorithm aims to minimize the within-cluster sum of squares (WCSS), given by:

$$J = \sum_{j=1}^{k} \sum_{i \in S(j)} \|x_i - c_j\|^2$$

Where: - J is the objective function value (WCSS). - k is the number of clusters. - x_i is a data point. - c_j is the centroid of cluster j.

The goal of the k-means algorithm is to find the set of centroids $\{c_1, c_2, ..., c_k\}$ that minimize J.

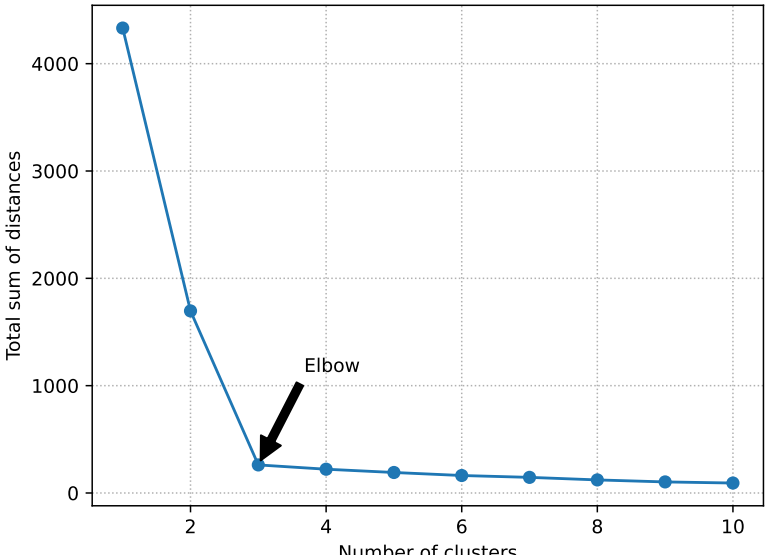
Notes:

- The k-means algorithm does not guarantee a global optimum solution. The final result might depend on the initial centroids.
- To improve the chances of finding a global optimum, the algorithm can be run multiple times with different initializations and the best result (i.e., the one with the lowest WCSS) can be selected.

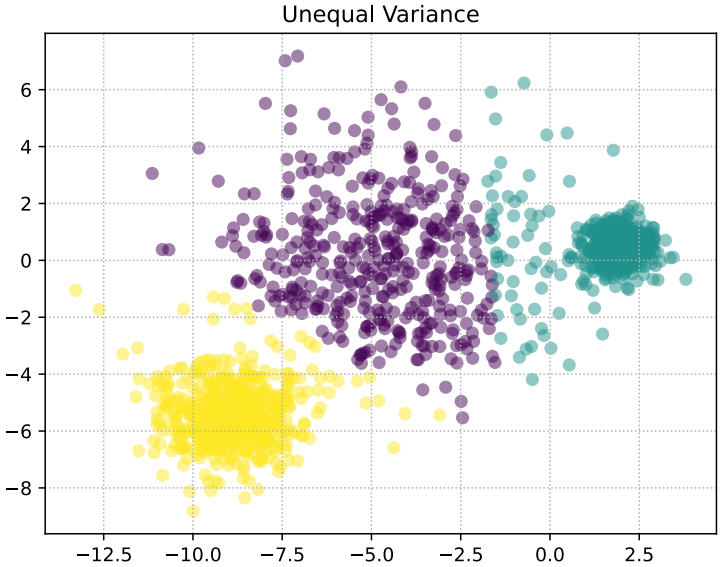
Traps. Initialization

Traps. The number of clusters

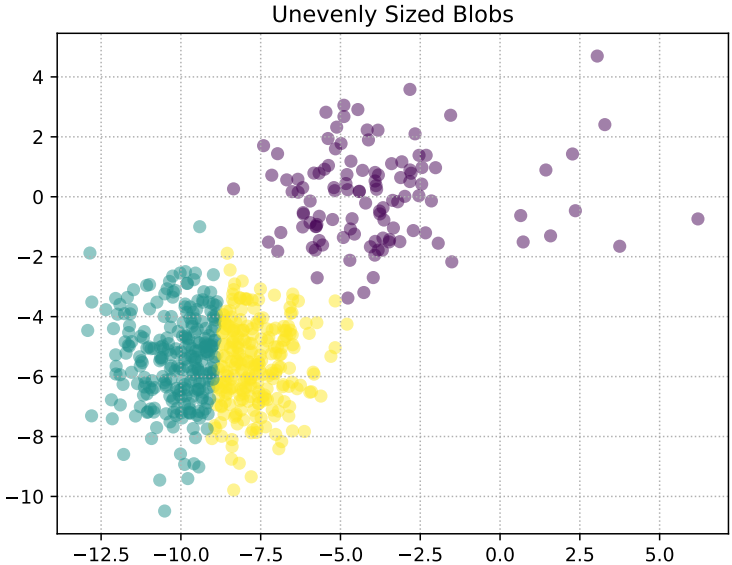
Elbow method



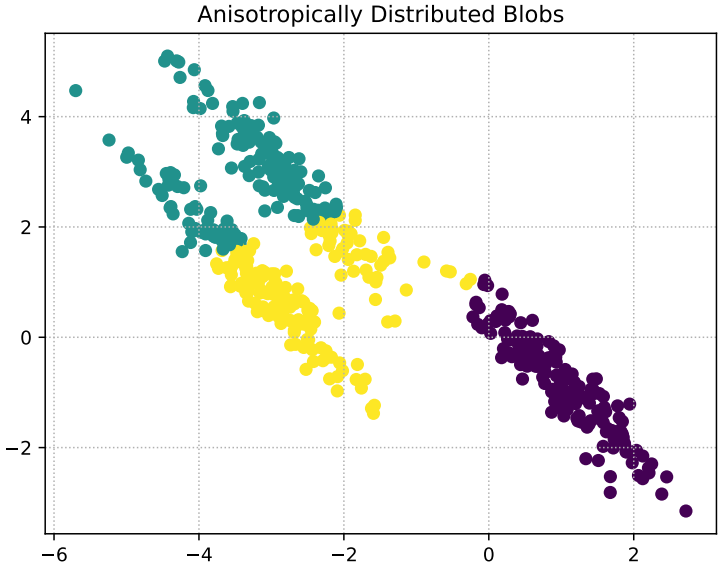
Traps. Different variances



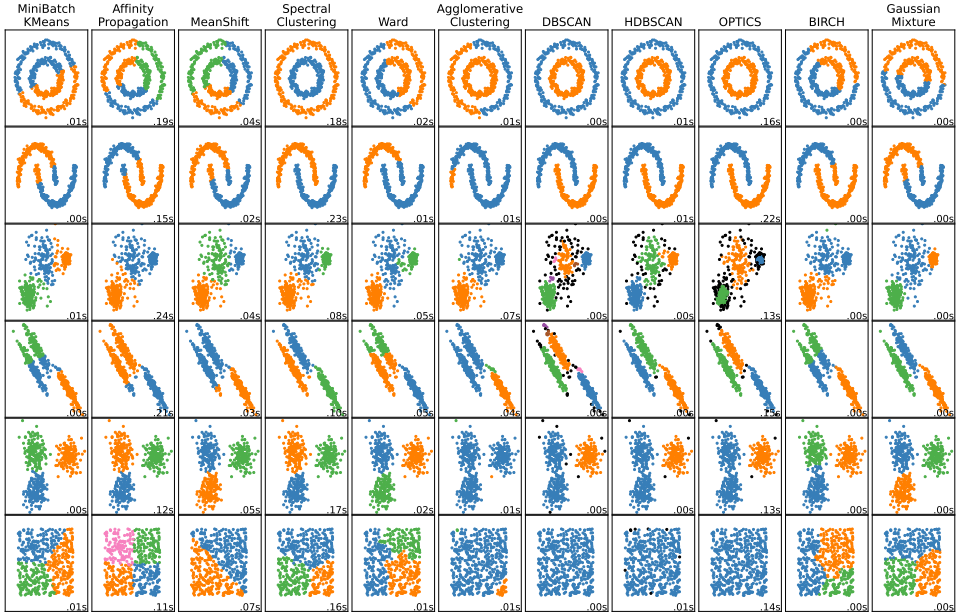
Traps. Different sizes



Traps. Anisotropic data



Another clustering methods



Practice

Hierarchical Clustering

Idea

- One nice feature of hierarchical clustering is that we can visualize the results as a dendrogram, a hierarchical tree.

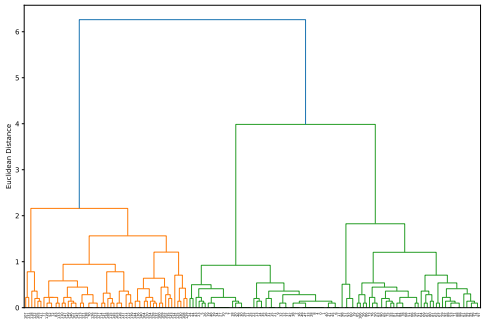


Figure 7: hi

Idea

- One nice feature of hierarchical clustering is that we can visualize the results as a dendrogram, a hierarchical tree.
- Using the visualization, we can then decide how “deep” we want to cluster the dataset by setting a “depth” threshold

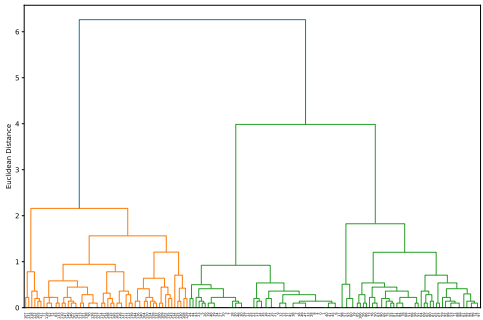


Figure 7: hi

Idea

- One nice feature of hierarchical clustering is that we can visualize the results as a dendrogram, a hierarchical tree.
- Using the visualization, we can then decide how “deep” we want to cluster the dataset by setting a “depth” threshold
- Or in other words, we don’t need to make a decision about the number of clusters upfront.

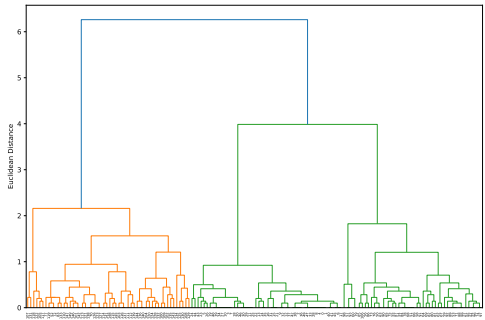


Figure 7: hi

Agglomerative and divisive hierarchical clustering

- Furthermore, we can distinguish between 2 main approaches to hierarchical clustering: Divisive clustering and agglomerative clustering.

We will study **agglomerative** clustering.

Agglomerative and divisive hierarchical clustering

- Furthermore, we can distinguish between 2 main approaches to hierarchical clustering: Divisive clustering and agglomerative clustering.
- In agglomerative clustering, we start with a single sample from our dataset and iteratively merge it with other samples to form clusters - we can see it as a bottom-up approach for building the clustering dendrogram.

We will study **agglomerative** clustering.

Agglomerative and divisive hierarchical clustering

- Furthermore, we can distinguish between 2 main approaches to hierarchical clustering: Divisive clustering and agglomerative clustering.
- In agglomerative clustering, we start with a single sample from our dataset and iteratively merge it with other samples to form clusters - we can see it as a bottom-up approach for building the clustering dendrogram.
- In divisive clustering, however, we start with the whole dataset as one cluster, and we iteratively split it into smaller subclusters - a top-down approach.

We will study **agglomerative** clustering.

How to compute similarity between clusters?

- Now, the next question is how we measure the similarity between samples.

How to compute similarity between clusters?

- Now, the next question is how we measure the similarity between samples.
- One approach is the familiar Euclidean distance metric that we already used via the K-Means algorithm.

How to compute similarity between clusters?

- Now, the next question is how we measure the similarity between samples.
- One approach is the familiar Euclidean distance metric that we already used via the K-Means algorithm.
- As a refresher, the distance between 2 m-dimensional vectors $\mathbf{p}$ and $\mathbf{q}$ can be computed as:

$$\begin{aligned} \mathrm{d}(\mathbf{q}, \mathbf{p}) &= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_m - p_m)^2} \\ &= \sqrt{\sum_{j=1}^{m} (q_j - p_j)^2} = \|\mathbf{q} - \mathbf{p}\|_2 \end{aligned}$$

How to compute similarity between clusters?

- Now, the next question is how we measure the similarity between samples.
- One approach is the familiar Euclidean distance metric that we already used via the K-Means algorithm.
- As a refresher, the distance between 2 m-dimensional vectors $\mathbf{p}$ and $\mathbf{q}$ can be computed as:

$$\begin{aligned} \mathrm{d}(\mathbf{q}, \mathbf{p}) &= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_m - p_m)^2} \\ &= \sqrt{\sum_{j=1}^{m} (q_j - p_j)^2} = \|\mathbf{q} - \mathbf{p}\|_2 \end{aligned}$$

- However, that's the distance between 2 samples.

How to compute similarity between clusters?

- Now, the next question is how we measure the similarity between samples.
- One approach is the familiar Euclidean distance metric that we already used via the K-Means algorithm.
- As a refresher, the distance between 2 m-dimensional vectors $\mathbf{p}$ and $\mathbf{q}$ can be computed as:

$$\begin{aligned} \mathrm{d}(\mathbf{q}, \mathbf{p}) &= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_m - p_m)^2} \\ &= \sqrt{\sum_{j=1}^{m} (q_j - p_j)^2} = \|\mathbf{q} - \mathbf{p}\|_2 \end{aligned}$$

- However, that's the distance between 2 samples.
- Now, how do we compute the similarity between subclusters of samples?

How to compute similarity between clusters?

- Now, the next question is how we measure the similarity between samples.
- One approach is the familiar Euclidean distance metric that we already used via the K-Means algorithm.
- As a refresher, the distance between 2 m-dimensional vectors $\mathbf{p}$ and $\mathbf{q}$ can be computed as:

$$\begin{aligned} \mathrm{d}(\mathbf{q}, \mathbf{p}) &= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_m - p_m)^2} \\ &= \sqrt{\sum_{j=1}^{m} (q_j - p_j)^2} = \|\mathbf{q} - \mathbf{p}\|_2 \end{aligned}$$

- However, that's the distance between 2 samples.
- Now, how do we compute the similarity between subclusters of samples?
- I.e., our goal is to iteratively merge the most similar pairs of clusters until only one big cluster remains.

How to compute similarity between clusters?

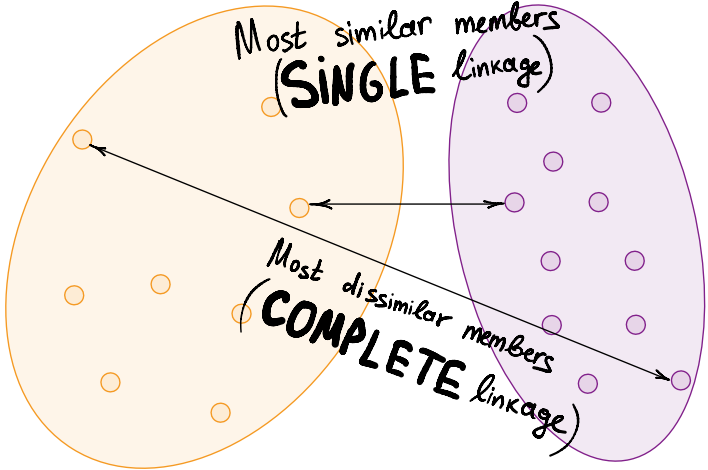
- Now, the next question is how we measure the similarity between samples.
- One approach is the familiar Euclidean distance metric that we already used via the K-Means algorithm.
- As a refresher, the distance between 2 m-dimensional vectors $\mathbf{p}$ and $\mathbf{q}$ can be computed as:

$$\begin{aligned} \mathrm{d}(\mathbf{q}, \mathbf{p}) &= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_m - p_m)^2} \\ &= \sqrt{\sum_{j=1}^{m} (q_j - p_j)^2} = \|\mathbf{q} - \mathbf{p}\|_2 \end{aligned}$$

- However, that's the distance between 2 samples.
- Now, how do we compute the similarity between subclusters of samples?
- I.e., our goal is to iteratively merge the most similar pairs of clusters until only one big cluster remains.
- There are many different approaches to this, for example single and complete linkage.

Single and complete linkage

- In single linkage, we take the pair of the most similar samples (based on the Euclidean distance, for example) in each cluster, and merge the two clusters which have the most similar 2 members into one new, bigger cluster.



Single and complete linkage

- In single linkage, we take the pair of the most similar samples (based on the Euclidean distance, for example) in each cluster, and merge the two clusters which have the most similar 2 members into one new, bigger cluster.
- In complete linkage, we compare the pairs of the two most dissimilar members of each cluster with each other, and we merge the 2 clusters where the distance between its 2 most dissimilar members is smallest.