Clustering Use Cases

Monday 6 May 2024 8:11 PM

→ Customer segmentation

→ Data Analysis

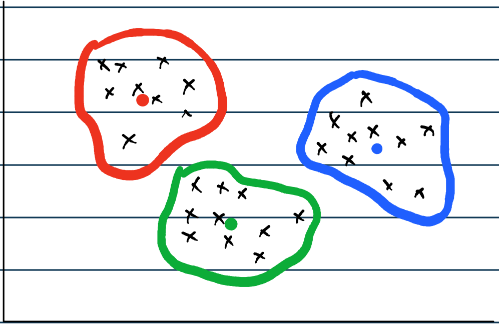
→ Semi Supervised Learning → google photos

→ Image segmentation

K-means

Monday 6 May 2024 8:18 PM

1. Decide k clusters
2. Initialize centroids
3. start iterating
   - Assign clusters → euclidean distance of every point with centroid
   - Move centroid $(\bar{x}, \bar{y})$
   - Check & stop → Previous centroid position = Current centroid position.



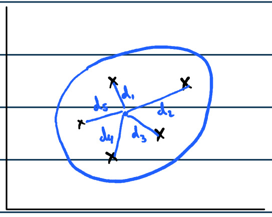
↑ Lloyd algorithm

→ Quality of clustering :

Sklearn.cluster

1) Elbow Method : → Ambigious

WCSS - within cluster sum of squared distances (inertia)



$WCSS = d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2$ ↓ better

wcss vs cluster graph [elbow curve] [choose elbow point]

→ the point from which the wcss doesn't decrease as rapidly

→ Not gaining much info after clustering beyond a certain point.

→ We need to find optimal cluster value for which the variance within a cluster is not too high or low.

2) Silhouette score :

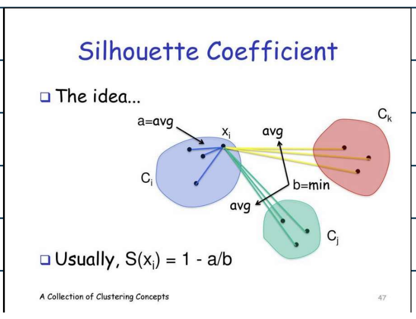
→ General Metric used to quantify the quality of clustering

→ Why is measuring the quality of clustering hard? → unsupervised → (No ground truth) No labels

→ Compactness of a cluster & separation b/w clusters
(cohesion) (tightly packed) (separation)

→ ideal : ↓ cohesion ↑ separation

→



Silhouette Coefficient

❑ The idea...

❑ Usually, $S(x_i) = 1 - a/b$

A Collection of Clustering Concepts 47

→ $S_{(i)} = \frac{\uparrow b_{(i)} - \overset{\downarrow}{a_{(i)}}}{\max(a_{(i)}, b_{(i)})} \uparrow$ → for each data point

$a_{(i)}$ → average distance with other points within same cluster

$b_{(i)}$ → average distance with other points which are part of each other clusters [choose the smaller cluster average]

→ close to +1 : Indicates that the data point is far each other clusters [choose the smaller cluster average]

→ Close to +1: Indicates that the data point is far away from the neighboring clusters

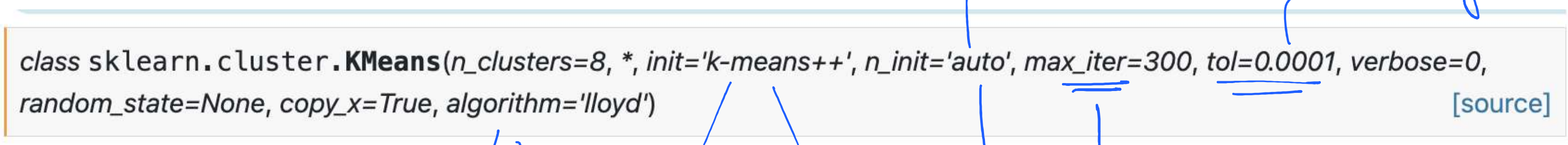
→ Close to 0: Indicates that the data point is on or very close to the decision boundary b/w two neighboring clusters

→ Close to -1: Indicates that the data point may have been assigned to the wrong cluster.

→ Average silhoutte score ↑ better quality

→ Average silhoutte score vs # of clusters

→ Cluster vs silhoutte score (histogram of silhoutte scores organized in descending order, grouped by cluster)



```
class sklearn.cluster.KMeans(n_clusters=8, *, init='k-means++', n_init='auto', max_iter=300, tol=0.0001, verbose=0,
random_state=None, copy_x=True, algorithm='lloyd')                                        [source]
```

→ kmeans output strongly depends on the initlizaton of centroids.

→ convergence limit

→ loop limit

→ How many times to perform the kmeans algo? calculates inertia for n iterations & selects best solution.

k-means++

random

lloyd

elkan (accelerated kmeans)

Prediction

→ assign cluster based on closest centroid

Assumptions

1) Spherical cluster shape; uniformity in all directions

2) Similar Cluster size

3) equal variance of cluster

4) Clusters are well separated

5) # of clusters is predefined

6) Large n & Small k

Limitations

1) Determining the optimal # of clusters is not straightforward & often requires domain knowledge or methods like the elbow method

2) Requires clusters of similar sizes

3) Kmeans requires clusters to be of similar variance

4) Assumption of spherical clusters & similar size clusters might not be the case in real-world data.

5) Vulnerability to Outliers

6) Hard clustering

7) High-Dimensional Challenges [euclidean distance not reliable in higher dimensions]

8) Sensitive to Scale

K-means++ (initialization technique)

Thursday 9 May 2024 6:28 PM

→ K-means final result depends (sensitive) strongly on the initialization of centroids

→ random initilization
- optimal
- sub-optimal

→ This method tends to spread the inital centroids, which can lead to better clustering results compared to selecting the inital centroids randomly. KMeans ++ can often lead to faster convergence & better clustering.

→ Algorithm [Research]

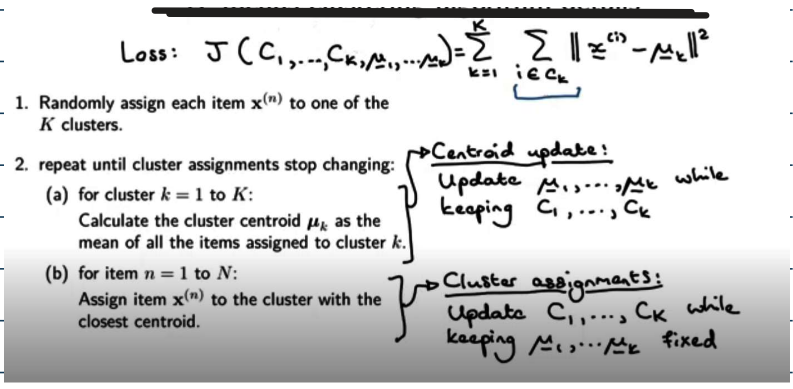
Mathematical Formulation

Thursday 9 May 2024 7:17 PM

$$J = \underset{\underbrace{\mu_1,\mu_2\ldots\mu_k}_{\text{centroids}}}{\operatorname{argmin}} \underbrace{\sum_{i=1}^{k}\sum_{x\in S_i} \|x-\mu_i\|^2}_{\text{Inertia}} \quad \text{such that } S_i\cap S_j=\{\}$$

↳ current cluster

very hard to solve



[redacted]

Loss: $J(C_1,\ldots,C_K,\underline{\mu}_1,\ldots\underline{\mu}_K)=\sum_{k=1}^{K}\underbrace{\sum_{i\in C_k}}\|\underline{x}^{(i)}-\underline{\mu}_k\|^2$

1. Randomly assign each item $\mathbf{x}^{(n)}$ to one of the K clusters.
2. repeat until cluster assignments stop changing:
   (a) for cluster $k=1$ to K:
   Calculate the cluster centroid μ_k as the mean of all the items assigned to cluster k.
   (b) for item $n=1$ to N:
   Assign item $\mathbf{x}^{(n)}$ to the cluster with the closest centroid.

→ Centroid update:
Update $\underline{\mu}_1,\ldots,\underline{\mu}_k$ while keeping $C_1,\ldots,C_k$

→ Cluster assignments:
Update $C_1,\ldots,C_K$ while keeping $\underline{\mu}_1,\ldots\underline{\mu}_k$ fixed

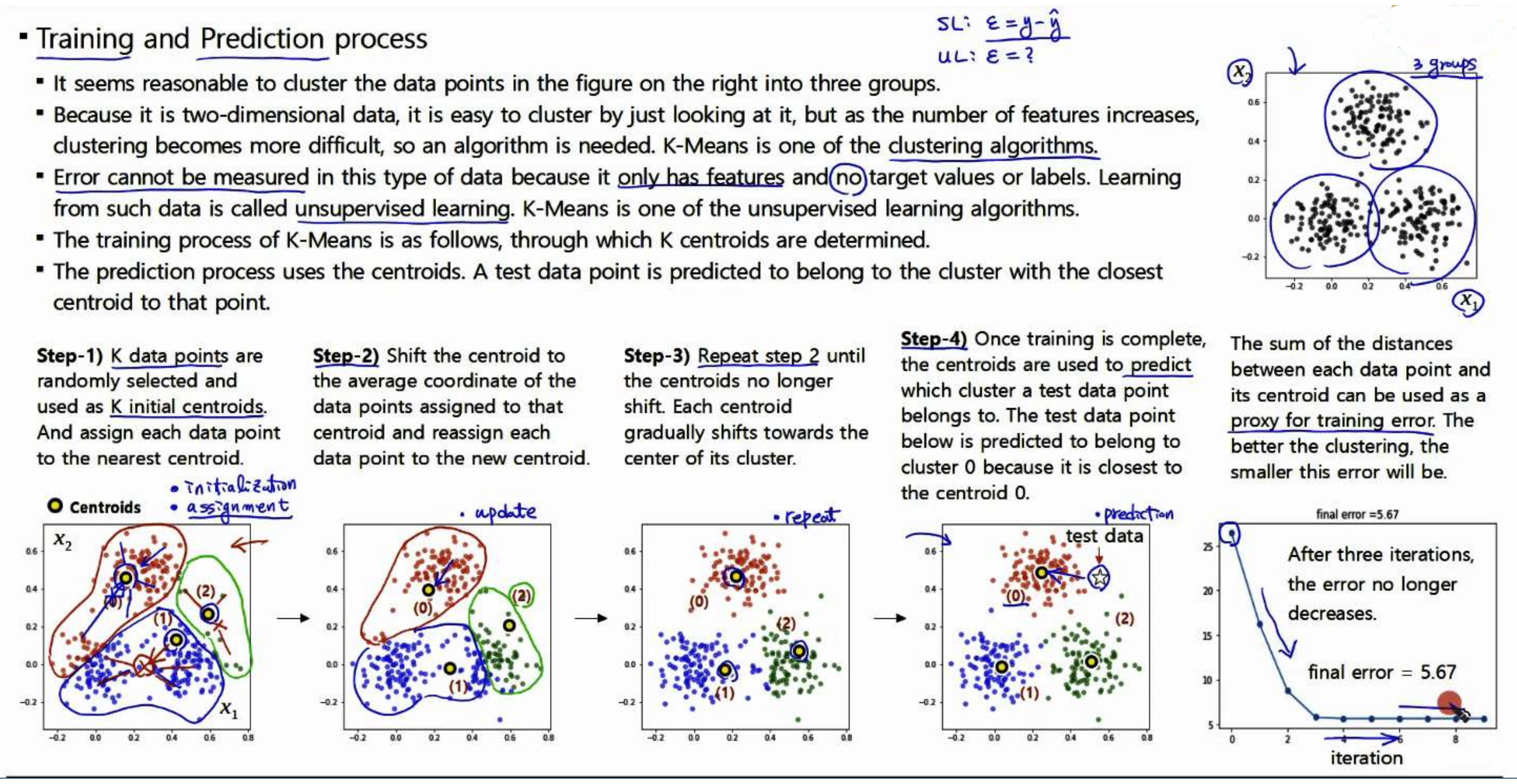
→ NP-hard

→ Non-convex function

→ lloyd's algorithm is a way around (sub-optimal) (local minima)

K-means Revisited

Friday 24 May 2024 2:34 AM



Training and Prediction process

SL: $\varepsilon = y - \hat{y}$

UL: $\varepsilon = ?$

- It seems reasonable to cluster the data points in the figure on the right into three groups.
- Because it is two-dimensional data, it is easy to cluster by just looking at it, but as the number of features increases, clustering becomes more difficult, so an algorithm is needed. K-Means is one of the clustering algorithms.
- Error cannot be measured in this type of data because it only has features and no target values or labels. Learning from such data is called unsupervised learning. K-Means is one of the unsupervised learning algorithms.
- The training process of K-Means is as follows, through which K centroids are determined.
- The prediction process uses the centroids. A test data point is predicted to belong to the cluster with the closest centroid to that point.

Step-1) K data points are randomly selected and used as K initial centroids. And assign each data point to the nearest centroid.

Step-2) Shift the centroid to the average coordinate of the data points assigned to that centroid and reassign each data point to the new centroid.

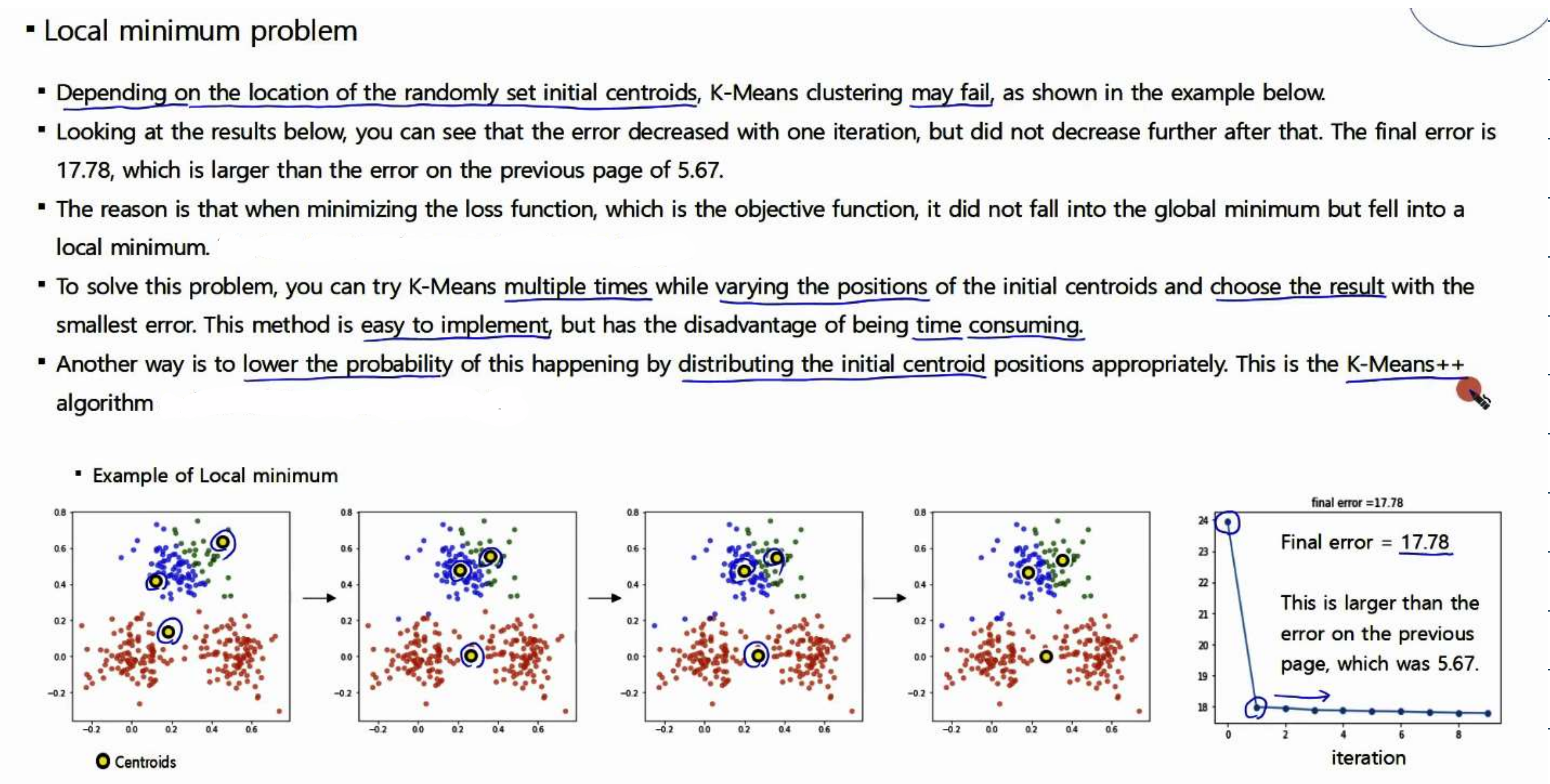
Step-3) Repeat step 2 until the centroids no longer shift. Each centroid gradually shifts towards the center of its cluster.

Step-4) Once training is complete, the centroids are used to predict which cluster a test data point belongs to. The test data point below is predicted to belong to cluster 0 because it is closest to the centroid 0.

The sum of the distances between each data point and its centroid can be used as a proxy for training error. The better the clustering, the smaller this error will be.

After three iterations, the error no longer decreases.

final error = 5.67



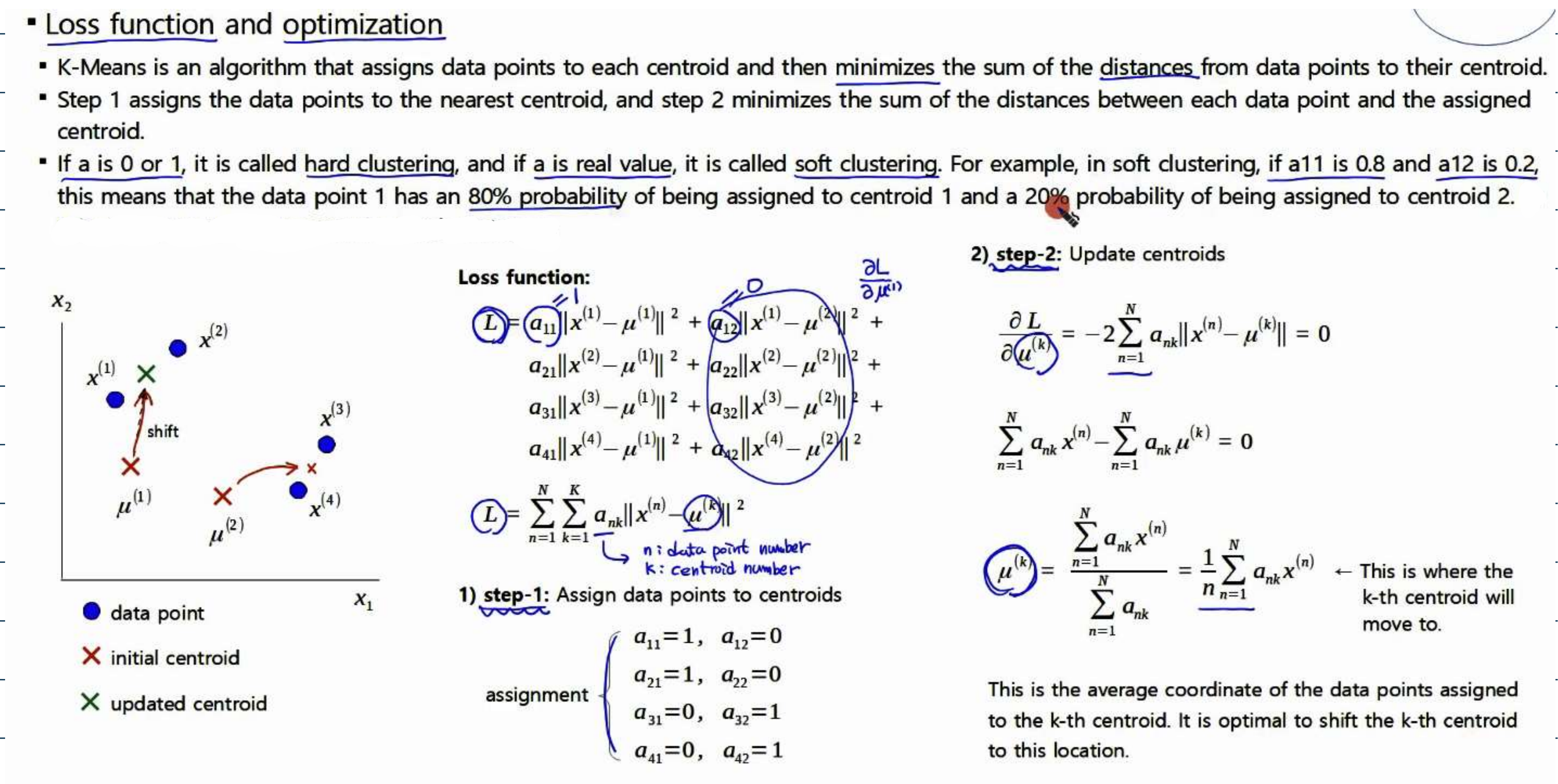
Local minimum problem

- Depending on the location of the randomly set initial centroids, K-Means clustering may fail, as shown in the example below.
- Looking at the results below, you can see that the error decreased with one iteration, but did not decrease further after that. The final error is 17.78, which is larger than the error on the previous page of 5.67.
- The reason is that when minimizing the loss function, which is the objective function, it did not fall into the global minimum but fell into a local minimum.
- To solve this problem, you can try K-Means multiple times while varying the positions of the initial centroids and choose the result with the smallest error. This method is easy to implement, but has the disadvantage of being time consuming.
- Another way is to lower the probability of this happening by distributing the initial centroid positions appropriately. This is the K-Means++ algorithm

  - Example of Local minimum

Final error = 17.78

This is larger than the error on the previous page, which was 5.67.



▪ Loss function and optimization

- K-Means is an algorithm that assigns data points to each centroid and then minimizes the sum of the distances from data points to their centroid.
- Step 1 assigns the data points to the nearest centroid, and step 2 minimizes the sum of the distances between each data point and the assigned centroid.
- If a is 0 or 1, it is called hard clustering, and if a is real value, it is called soft clustering. For example, in soft clustering, if a11 is 0.8 and a12 is 0.2, this means that the data point 1 has an 80% probability of being assigned to centroid 1 and a 20% probability of being assigned to centroid 2.

Loss function:

$\frac{\partial L}{\partial \mu^{(1)}}$

$$\begin{aligned} L = {} & \overset{=1}{a_{11}}\|x^{(1)}-\mu^{(1)}\|^2 + \overset{=0}{a_{12}}\|x^{(1)}-\mu^{(2)}\|^2 + \\ & a_{21}\|x^{(2)}-\mu^{(1)}\|^2 + a_{22}\|x^{(2)}-\mu^{(2)}\|^2 + \\ & a_{31}\|x^{(3)}-\mu^{(1)}\|^2 + a_{32}\|x^{(3)}-\mu^{(2)}\|^2 + \\ & a_{41}\|x^{(4)}-\mu^{(1)}\|^2 + a_{42}\|x^{(4)}-\mu^{(2)}\|^2 \end{aligned}$$

$$L = \sum_{n=1}^{N}\sum_{k=1}^{K} \underline{a_{nk}}\|x^{(n)}-\mu^{(k)}\|^2$$

↳ n: data point number
k: centroid number

1) step-1: Assign data points to centroids

$$\text{assignment}\begin{cases} a_{11}=1, & a_{12}=0 \\ a_{21}=1, & a_{22}=0 \\ a_{31}=0, & a_{32}=1 \\ a_{41}=0, & a_{42}=1 \end{cases}$$

2) step-2: Update centroids

$$\frac{\partial L}{\partial \mu^{(k)}} = -2\underline{\sum_{n=1}^{N}} a_{nk}\|x^{(n)}-\mu^{(k)}\| = 0$$

$$\sum_{n=1}^{N} a_{nk}x^{(n)} - \sum_{n=1}^{N} a_{nk}\mu^{(k)} = 0$$

$$\mu^{(k)} = \frac{\sum_{n=1}^{N} a_{nk}x^{(n)}}{\sum_{n=1}^{N} a_{nk}} = \underline{\frac{1}{n}\sum_{n=1}^{N}} a_{nk}x^{(n)}$$

← This is where the k-th centroid will move to.

This is the average coordinate of the data points assigned to the k-th centroid. It is optimal to shift the k-th centroid to this location.

[Learn Later]

Thursday 9 May 2024 11:38 PM

→ Minibatch Kmeans

→ Acclerated Kmeans (eclat)