formal k-means clustering

Input: Dataset $x_1, \dots, x_n$, number of clusters K

Output: Cluster centers $c_1, \dots, c_k$

Goal: Minimize:

$$\text{cost}(c_1, \dots, c_k) = \sum_i \min_k\big(\text{dist}(x_i, c_k)\big)$$

The **objective** of **K-Means** Clustering is to **minimize** the **distance** between a point $\boldsymbol{x_i}$ and its nearest cluster center $\boldsymbol{c_k}$ summing all computed distances together. The **goal** to determine the cluster centers that minimize the **total distance** between all points $\boldsymbol{x_i}$ and the cluster centers (**Global Objective**).

Unlike Support Vector Machines and Logistic Regression, **K-Means** Clustering cannot optimize in a single step. **Global Minimization**: Try all possible assignments of m points to $\boldsymbol{K}$ clusters:

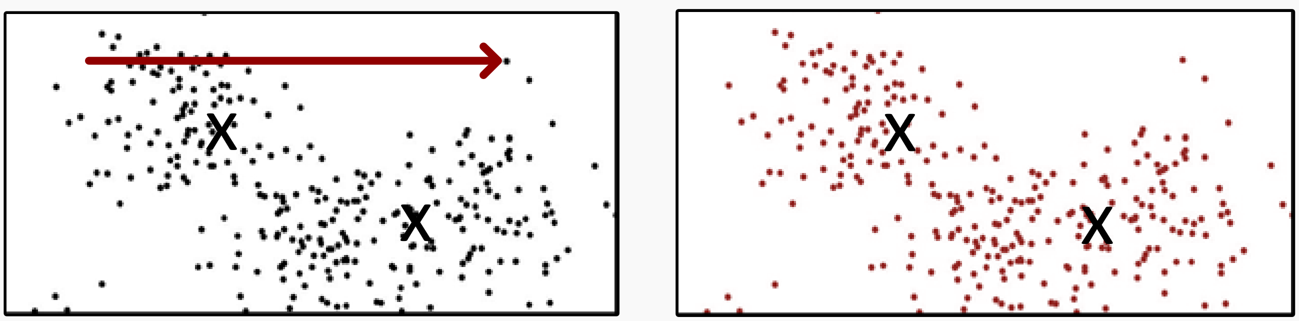
$$\text{Combinations}(m, K) = \frac{1}{K!} \sum_{k=1}^{K} (1)^{K-k} \binom{K}{k} k^n$$

$$\text{Combinations}(10, 4) = 34{,}000, \text{Combinations}(19, 4) = 10^{10}, \dots, \text{noncomputable}$$

Thus, **K-Means** Clustering is an approximation due the cluster assignments being indeterminable.

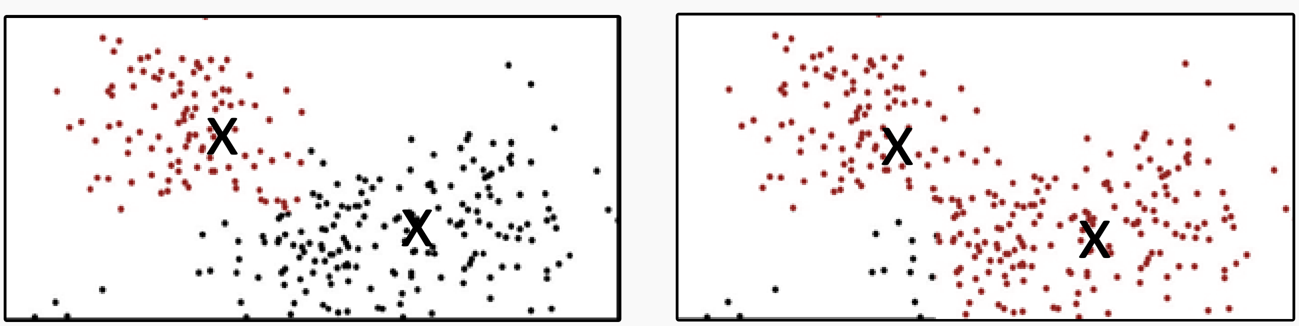
The **cost function** computes a sum over the points of distance to the nearest cluster center:

$$\text{cost}(c_1, \dots, c_k) = \sum_i \min_k\big(\text{dist}(x_i, c_k)\big)$$



The clusters can be summed at once from left to right as illustrated in the example above:

$$\text{cost}(c_1, \dots, c_k) = \sum_k \sum_{i: x_i \text{ is in cluster}_k} \text{dist}(x_i, c_k)$$



The clusters can equally be summed over by adding them up in order of the clusters illustrated above:

Illustrating a more general cost by focusing on the cluster assignments as well as the clusters c_k:

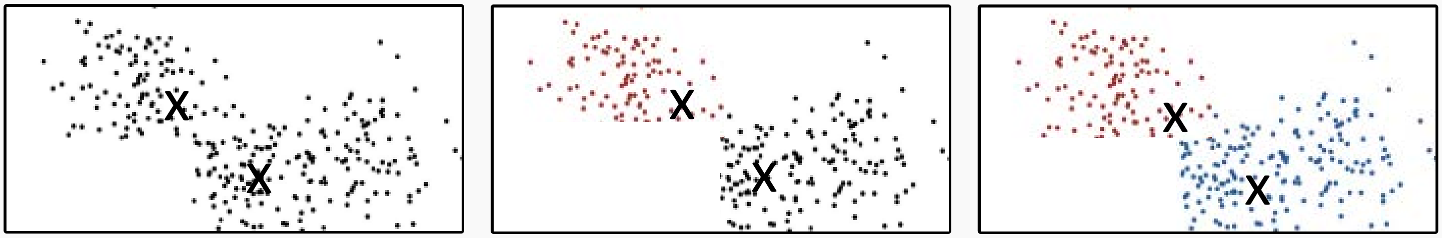
$$\text{cost}(\text{cluster}_1, \text{cluster}_2, \dots, \text{cluster}_k, c_1, \dots, c_K) = \sum_k \sum_{i:x_i \text{ is in cluster}_k} \text{dist}(x_i, c_k)$$

If the expression above was treated as just a functions of the cluster assignments $\mathbf{cluster}_k$ only:

$$\text{cost}(\mathbf{cluster_1}, \mathbf{cluster_2}, \dots, \mathbf{cluster}_k, c_1, \dots, c_K) = \sum_k \sum_{i:x_i \text{ is in } \mathbf{cluster_k}} \text{dist}(x_i, c_k)$$

The best way to assign the $\mathbf{clusters}_k$ is to assign all of the points to the nearest cluster center c_k:

$$\min_{\mathbf{cluster_1}, \mathbf{cluster_2}, \dots, \mathbf{cluster}_k} \text{cost}(\mathbf{cluster_1}, \mathbf{cluster_2}, \dots, \mathbf{cluster}_k, c_1, \dots, c_K)$$

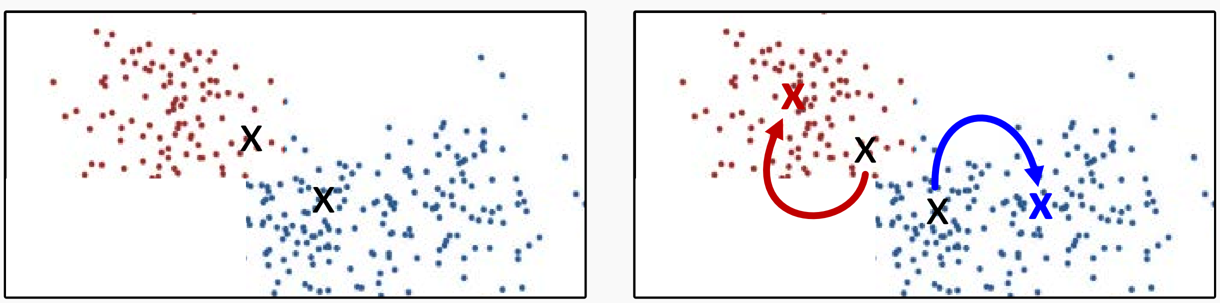


Alternatively, if the generalize cost function focused strictly on assigning the reference points c_k only:

$$\text{cost}(\text{cluster}_1, \text{cluster}_2, \dots, \text{cluster}_\text{k}, \boldsymbol{c_1, \dots, c_K}) = \sum_k \sum_{i:x_i \text{ is in } \mathbf{cluster}_k} dist(x_i, \boldsymbol{c_k})$$

The best way to minimize **cost** is to assign the centers to the middle of points assigned to that cluster:

$$\min_{c_1, c_2, \dots, c_K} \text{cost}(\mathbf{cluster_1}, \mathbf{cluster_2}, \dots, \mathbf{cluster_k}, \boldsymbol{c_1, \dots, c_K})$$



The above illustration assigns cluster centers by minimizing average distance to the reference points.

Thus the **generalize cost function** below is dependent on both the cluster assignments $\mathbf{cluster}_k$ and the cluster centers c_K.

$$\text{cost}(\mathbf{cluster_1}, \mathbf{cluster_2}, \dots, \mathbf{cluster}_k, \boldsymbol{c_1, \dots, c_K}) = \sum_k \sum_{i:x_i \text{ is in } \mathbf{cluster_k}} \text{dist}(x_i, \boldsymbol{c_k})$$

Input: number of clusters $\boldsymbol{K}$, randomly initialize centers c_k

Until converged:

Assign all points to the closest cluster center

$$\min_{\mathbf{cluster_1}, \mathbf{cluster_2}, \dots, \mathbf{cluster}_k} \text{cost}(\mathbf{cluster_1}, \mathbf{cluster_2}, \dots, \mathbf{cluster}_k, c_1, \dots, c_K)$$

Change cluster centers to be in the middle of its points

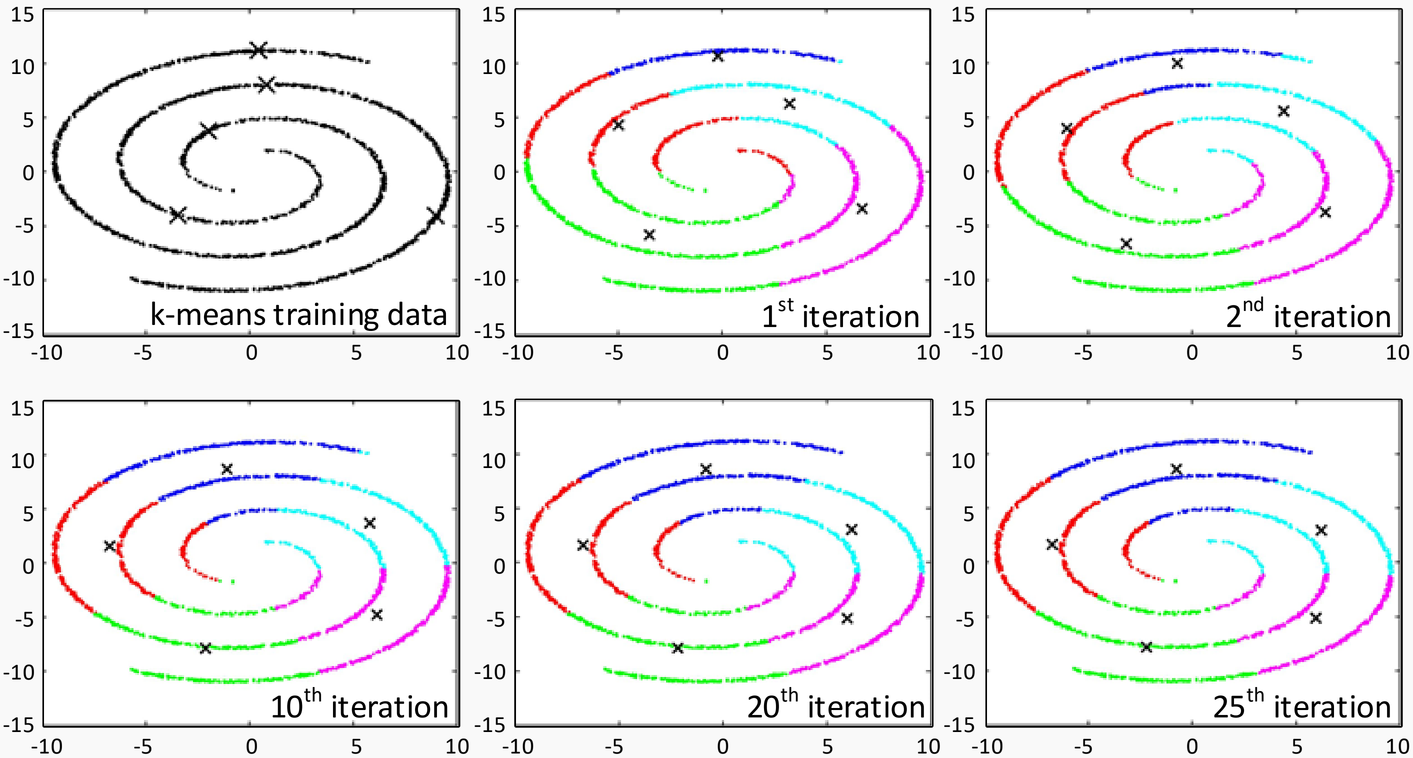
$$\min_{c_1, c_2, \dots, c_K} \text{cost}(\mathbf{cluster_1}, \mathbf{cluster_2}, \dots, \mathbf{cluster_k}, \boldsymbol{c_1, \dots, c_K})$$

K-Means Clustering is functions in alternating iterations of assigning clusters and centering them (**alternation minimization**).

K-means does not always achieve its goal of minimizing the cost function (**Global Minimization**):

$$\text{cost}(\mathbf{cluster_1}, \mathbf{cluster_2}, \ldots, \mathbf{cluster}_k, c_1, \ldots, c_K)$$

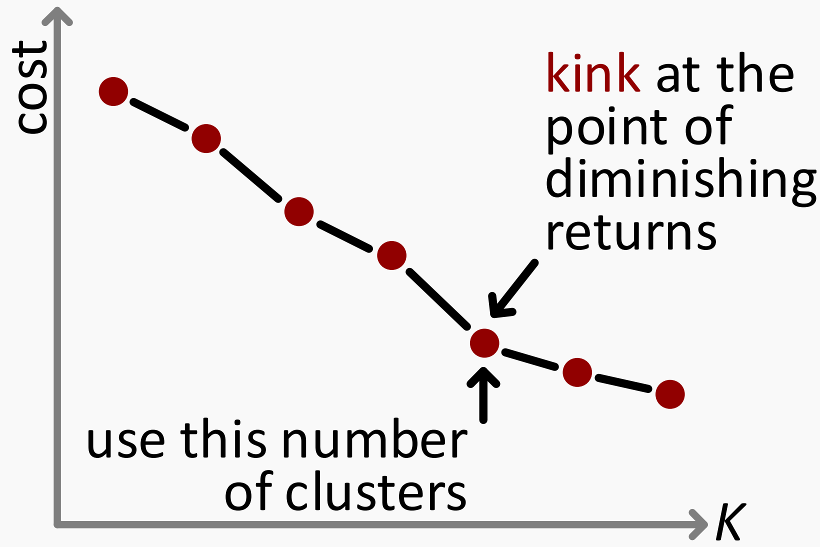
K-Means often requires multiple replicates and is possible that K-means goal is not the correct one:



The above illustration simulates the K-Means optimization algorithm on a set of data. The first image illustrates the **random initialization** of **cluster centers**. After many iterations, the model has "**converged**". It is evident the 25 iterations in the above experiment did not yield meaningful results.

choosing k for k-means clusting

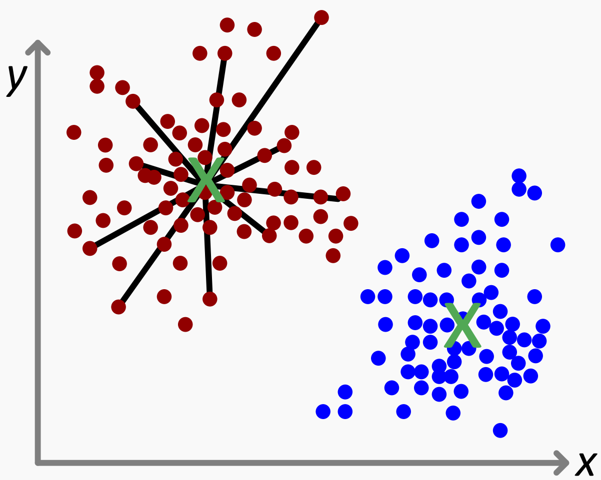
There are several acceptable options for choosing the optimal amount of clusters $\boldsymbol{K}$.

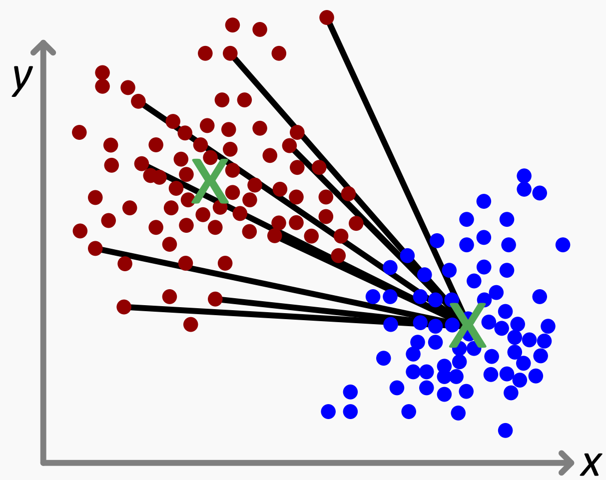


One of the more widely used practical applications is to plot the number of clusters $\boldsymbol{K}$ against the **cost** and examine for the point of **diminishing returns**. As the number of clusters increases, the cost functions will decrease. After a certain threshold, the decrease in cost is insignificant to the additional of new clusters $\boldsymbol{K}$.

Another option for optimal number of clusters $\boldsymbol{K}$ examines the ratio of **average distance of the assigned cluster center** to the **average distance of the other cluster centers**.

First compute the distance to the closest assigned cluster center;
Then compare the latter computation to the average of the distance to the other centers.





Ideally, the two separate computations above should differ from each other; implying that the clusters are not only tightly modeled, but far away and clearly separated from other clusters.

In application, the alternate metric to **choosing the number of clusters $\boldsymbol{K}$** is not ideal for $\boldsymbol{K} > \mathbf{2}$.

K-Means Clustering Summary

- Popular clustering algorithm, computationally efficient
- Performs alternating minimization on a cost function
- Does not always fully minimize the cost function
  (multiple replicates might be needed for a good solution)
- Can use the cost function to evaluate whether one replicate is better than another
- Can use cost function to help choose the number of clusters
- Does not work well for highly non-spherical clusters
- Euclidean distance is often used, but other distances can equally be used