DS107-Unsupervised Models - K-Means

By: eng. Esraa Madhi

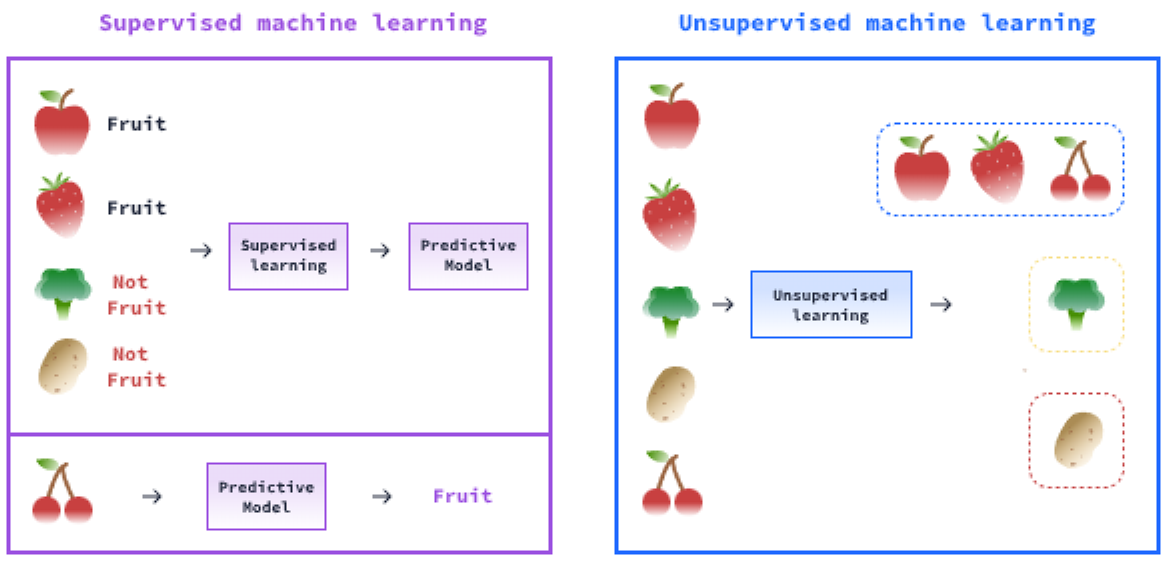
What is Unsupervised Learning?

https://youtu.be/5yeJ03crTrI

https://youtu.be/5yeJ03crTrI

Unsupervised Learning Approaches - Clustering:

The clustering approach works to **group non-labeled data based on similarities and differences**. Unlike supervised learning, clustering algorithms **discover natural groupings in data.**



What is K Means Algorithm?

https://youtu.be/mEGYznC6F9s

https://youtu.be/mEGYznC6F9s

The goal of K-means is to **divide a dataset into K clusters**, where each cluster contains **similar data points**. The number of clusters, K, is a **hyperparameter** that is specified by the user.

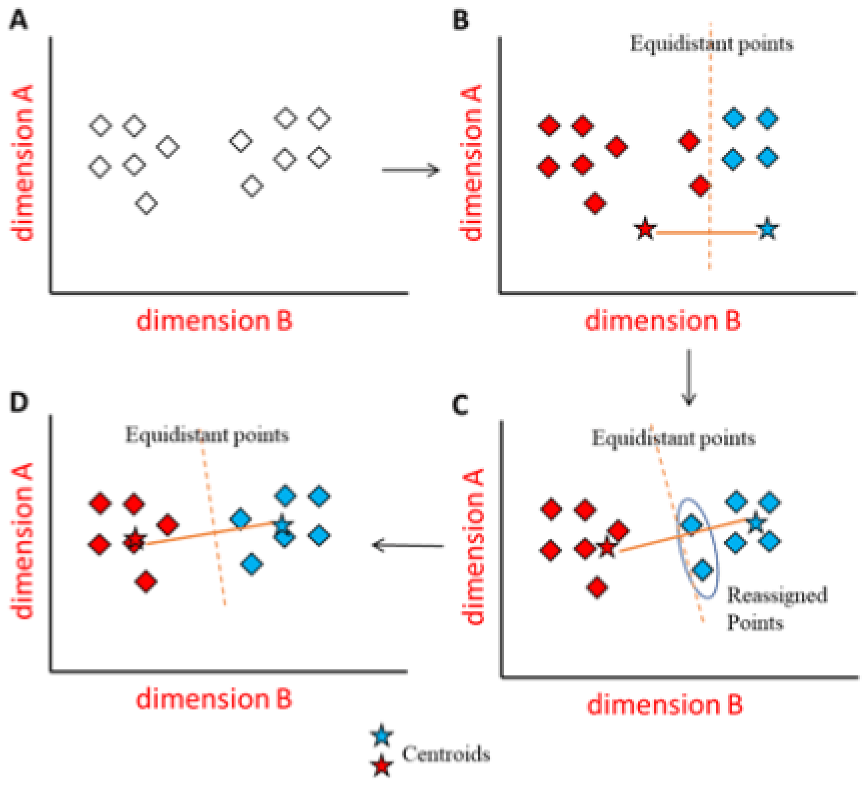
How K-Means Works?

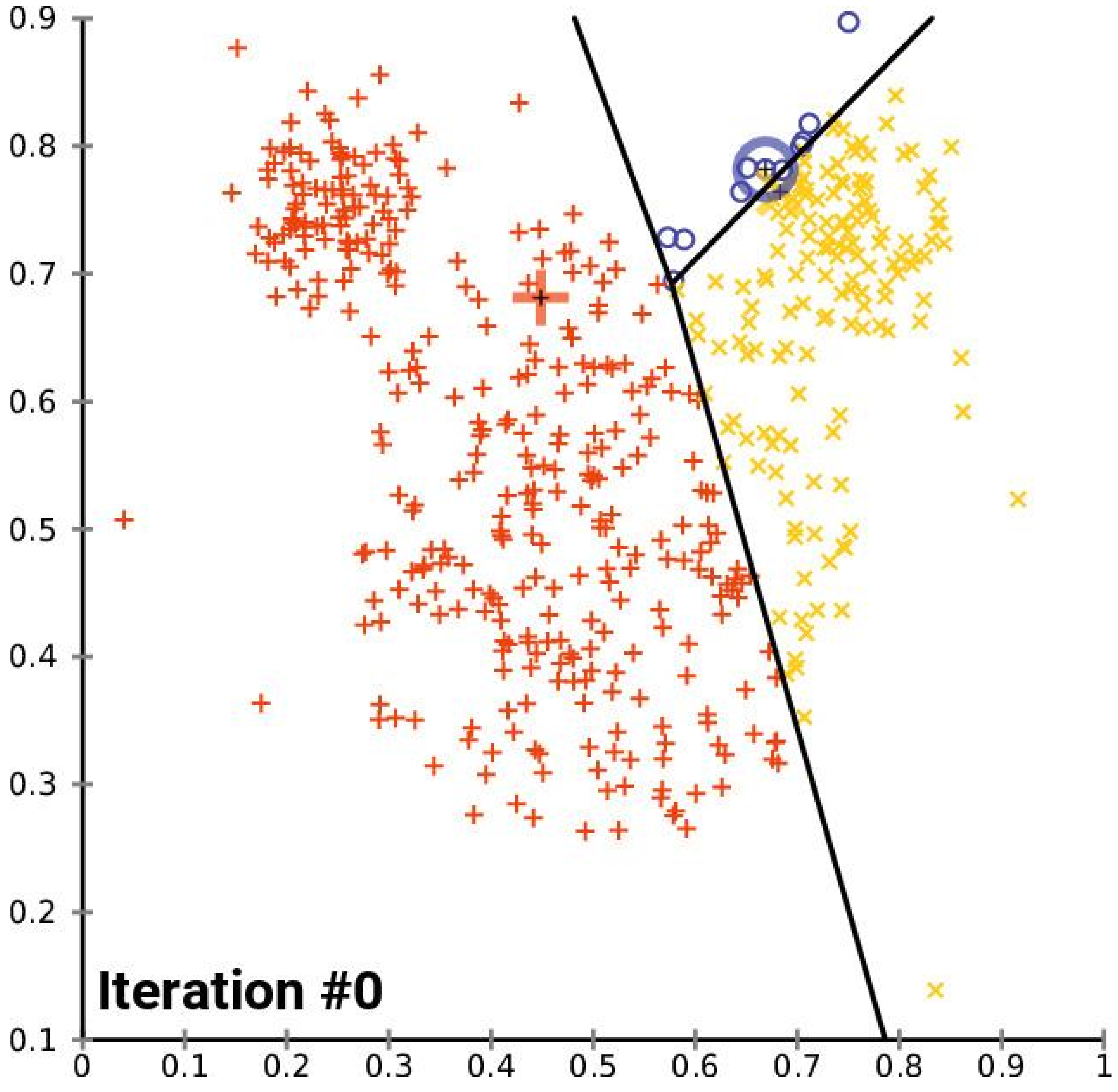
In Demo:

- https://www.naftaliharris.com/blog/visualizing-k-means-clustering/
- https://www.philippe-fournier-viger.com/tools/kmeans_demo.php
- https://user.ceng.metu.edu.tr/~akifakkus/courses/ceng574/k-means/
- http://alekseynp.com/viz/k-means.html
- http://ludovicstourm.com/KmeansDemo/

In Steps:

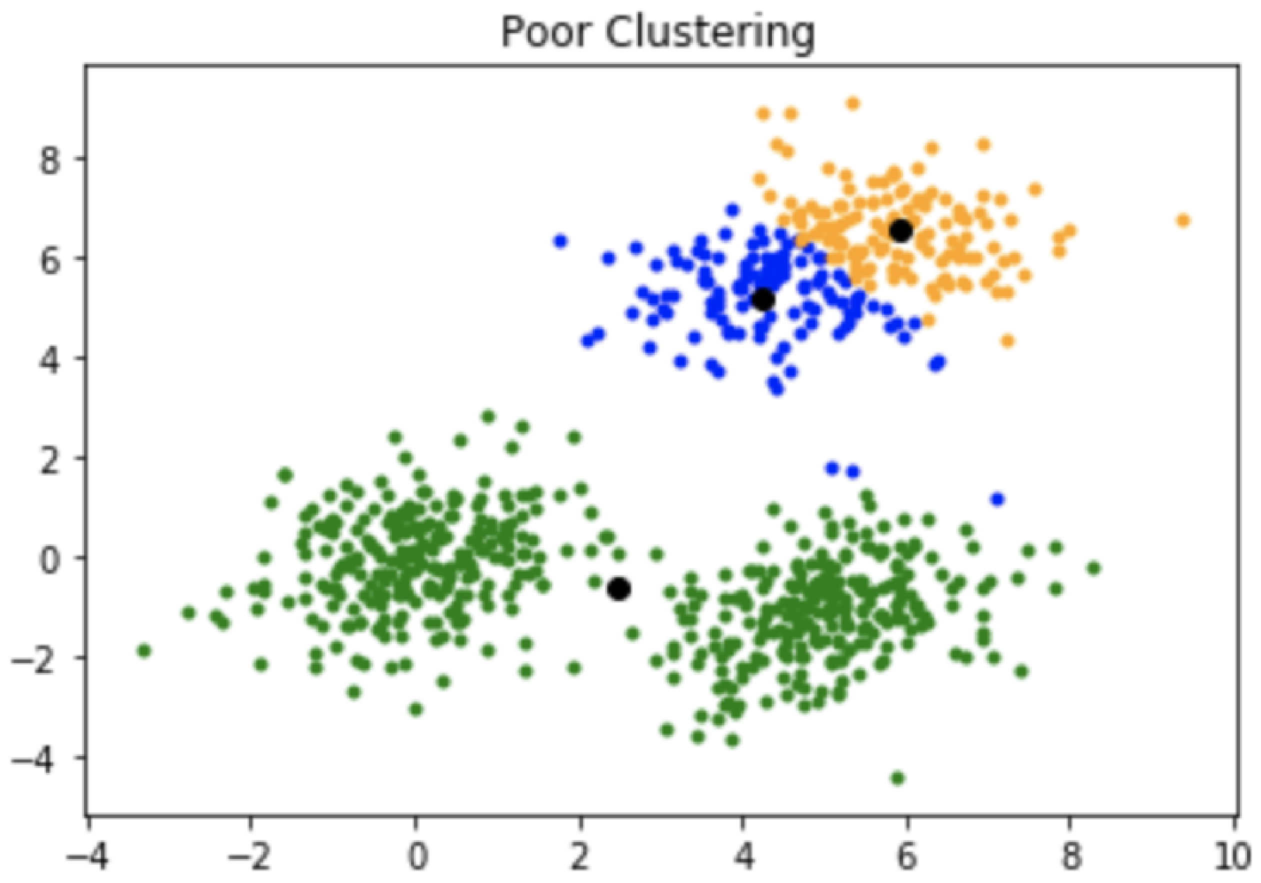
- The algorithm works by first initializing K centroids at random positions in the feature space. Each centroid represents the center of a cluster.
- Then, the algorithm iteratively performs the following steps:
  a. Assign each data point to the cluster whose centroid is closest to it. This is done by calculating the distance between the data point and each centroid and assigning the data point to the cluster whose centroid is closest.
  b. Recalculate the centroid of each cluster by taking the mean of all data points assigned to that cluster.
  c. Repeat steps 1 and 2 until the centroids **no longer move** or a **maximum number of iterations is reached.**





The algorithm stops **when the centroids stop moving** or when a **maximum number of iterations** is reached. The final clusters are determined by the final positions of the centroids.

Note: It is important to note that the K-Means algorithm **is sensitive to the initial positions of the centroids** and can sometimes converge to a local optimum instead of the global optimum. To overcome this issue, the algorithm is often **run multiple times** with different initial centroids and **the best solution is chosen.**



OR **use Kmeans++**:

Steps of selecting centroids in kmeans:

1. *Randomly select the first centroid from the data points.*
2. *For each data point compute its distance from the nearest, previously chosen centroid.*
3. *Select the next centroid from the data points such that the probability of choosing a point as centroid is directly proportional to its distance from the nearest, previously chosen centroid.* ***(i.e. the point having maximum distance from the nearest centroid is most likely to be selected next as a centroid)***
4. *Repeat steps 2 and 3* until all *k* centroids are chosen.

https://youtu.be/efKGmOH4Y_A

https://youtu.be/efKGmOH4Y_A

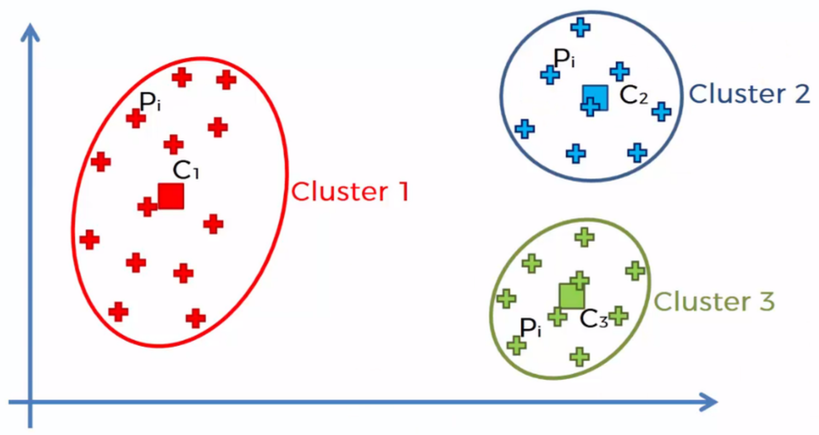
How to fine best K?

1. **The Elbow Method**

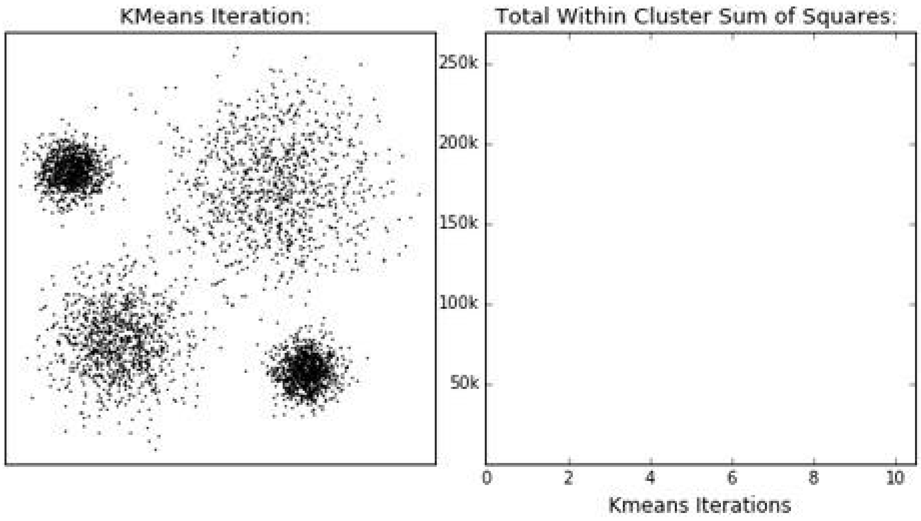
One popular technique is the Elbow Method. It involves running K-Means several times, each time with a different number of clusters, and then plotting the **Within-Cluster Sum of Squares (WCSS)** **(Inertia)** against the number of clusters. WCSS measures how tight your clusters are, **with lower values indicating that points are closer to their respective centroids.**

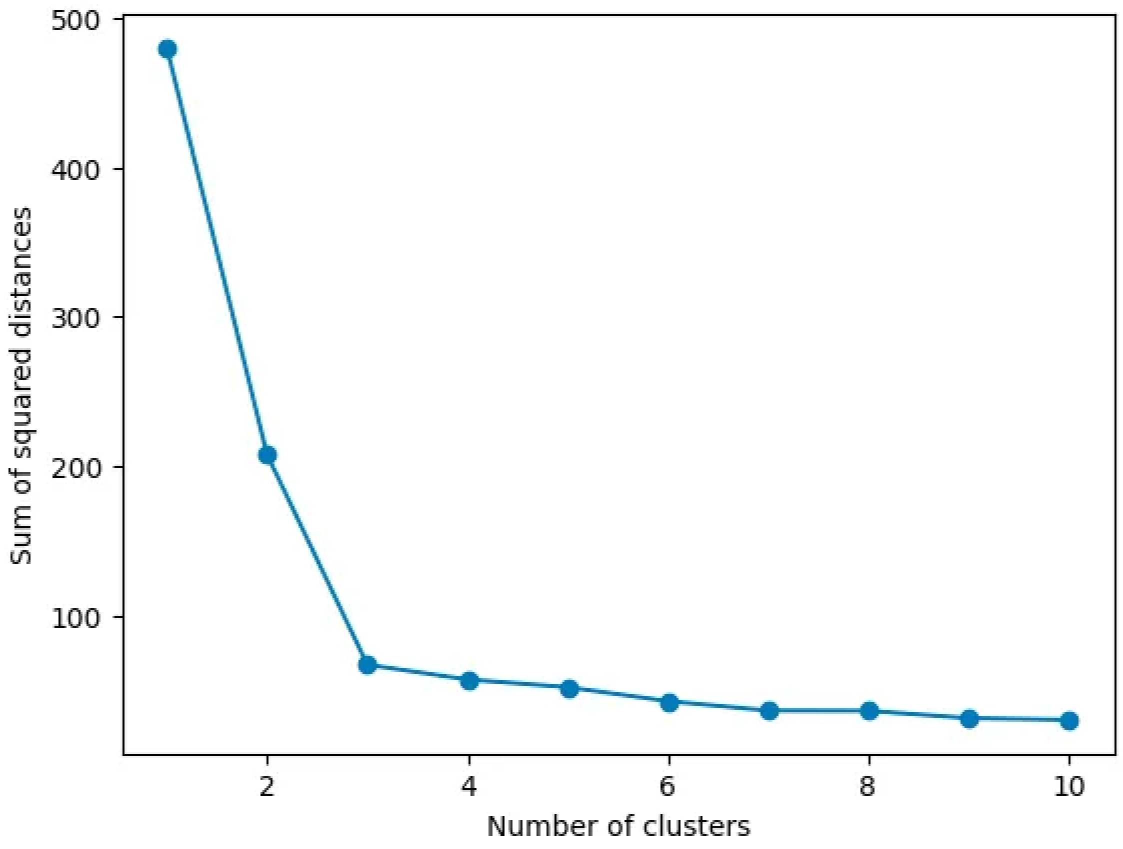
$$WCSS = \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \| \mathbf{x} - \mu_i \|^2$$

- ($\sum_{i=1}^{k}$): This is the summation over all clusters, from 1 to (k), where (k) is the number of clusters.
- ($\sum_{\mathbf{x} \in S_i}$): This is the summation over all data points ($\mathbf{x}$)that belong to the cluster (S_i).
- ($\| \mathbf{x} - \mu_i \|^2$): This represents the squared Euclidean distance between a data point ($\mathbf{x}$)and the centroid (μ_i)of the cluster (S_i) The norm ($\| \cdot \|$)typically denotes the Euclidean norm in this context.

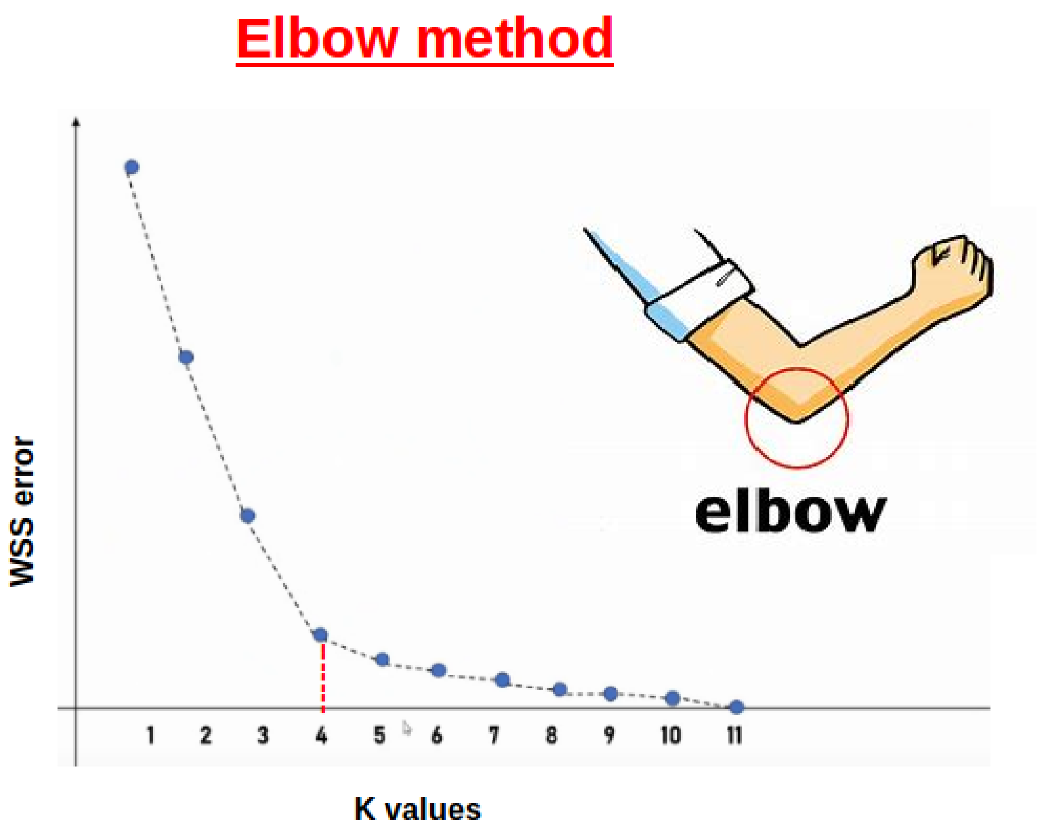


$$\mathrm{WCSS} = \sum_{\mathrm{P_i\ in\ Cluster\ 1}} \mathrm{distance}(\mathrm{P_i}, \mathrm{C_1})^2 + \sum_{\mathrm{P_i\ in\ Cluster\ 2}} \mathrm{distance}(\mathrm{P_i}, \mathrm{C_2})^2 + \sum_{\mathrm{P_i\ in\ Cluster\ 3}} \mathrm{distance}(\mathrm{P_i}, \mathrm{C_3})^2$$





As you increase *k*, WCSS decreases because the points are closer to centroids. However, there's a point where adding more clusters doesn't give you much bang for your buck. This point, where the rate of decrease sharply changes, looks like an elbow on the graph. In the graph above, the elbow point is 3



2. **The Silhouette coefficient**

How to calculate distance?

Kmeans uses the Euclidean distance as a measure of similarity. Other distance measures such as Manhattan distance or Cosine similarity can also be used.

Difference between KNN and K-mean

Many people get confused between these two statistical techniques- K-mean and K-nearest neighbor. See the difference between them below:

1. K-mean is an unsupervised learning technique **(no dependent variable)** whereas KNN is a supervised learning algorithm **(dependent variable exists)**
2. K-mean is a clustering technique which tries to split data points into **K-clusters** such that the points in each cluster tend to be near each other whereas K-nearest neighbor tries to determine the classification of a point, combines the **classification** of the K nearest points

---

Resources:

- https://thedatascientist.com/machine-learning-tutorial-a-practical-guide-of-unsupervised-learning-algorithms/