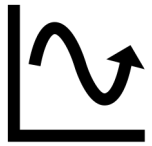
K-Means Clustering

INF2179H - ML with Applications in Python

Main Learning Types

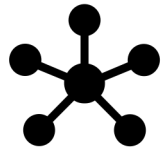
Supervised



Regression
prediction of a numeric value

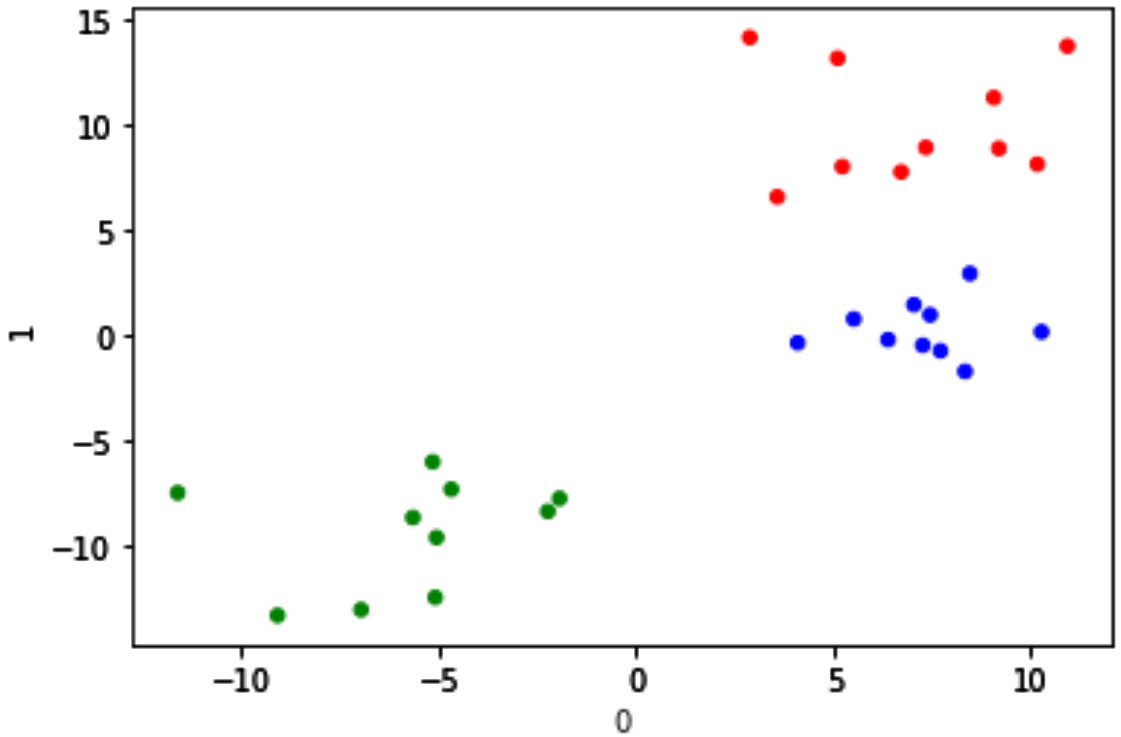
Classification
predict what class an instance of data should fall into

Unsupervised



Clustering
group similar items together
- No label -

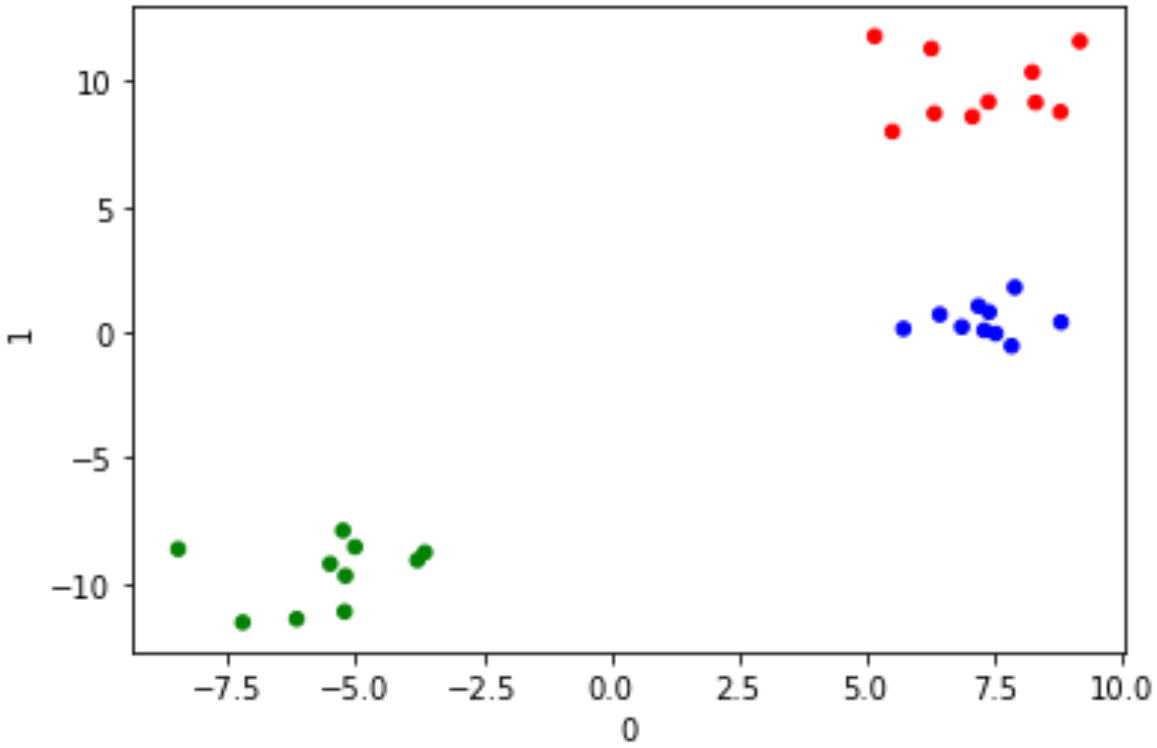
Clustering



Clustering

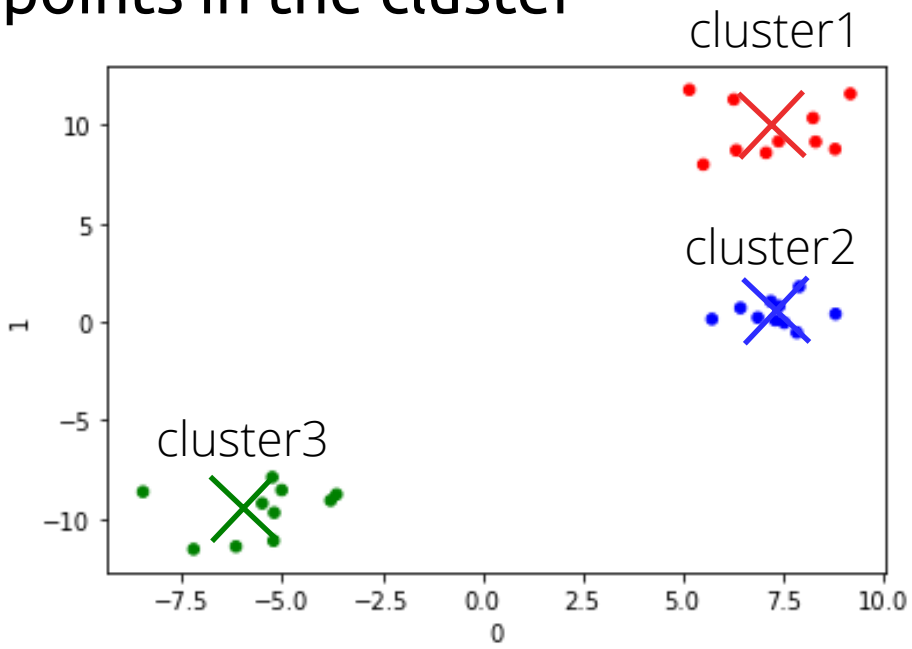
- Unsupervised
  - = No target variable = No class
- Use Case
  - Customer Segmentation
  - Educational data mining
  - Recommender systems
  - Document Analysis

Kmeans



Kmeans

- k is the number of desired clusters (input)
- Results are non-deterministic
- Centroid
  - Center of all the points in the cluster



Kmeans Algorithm

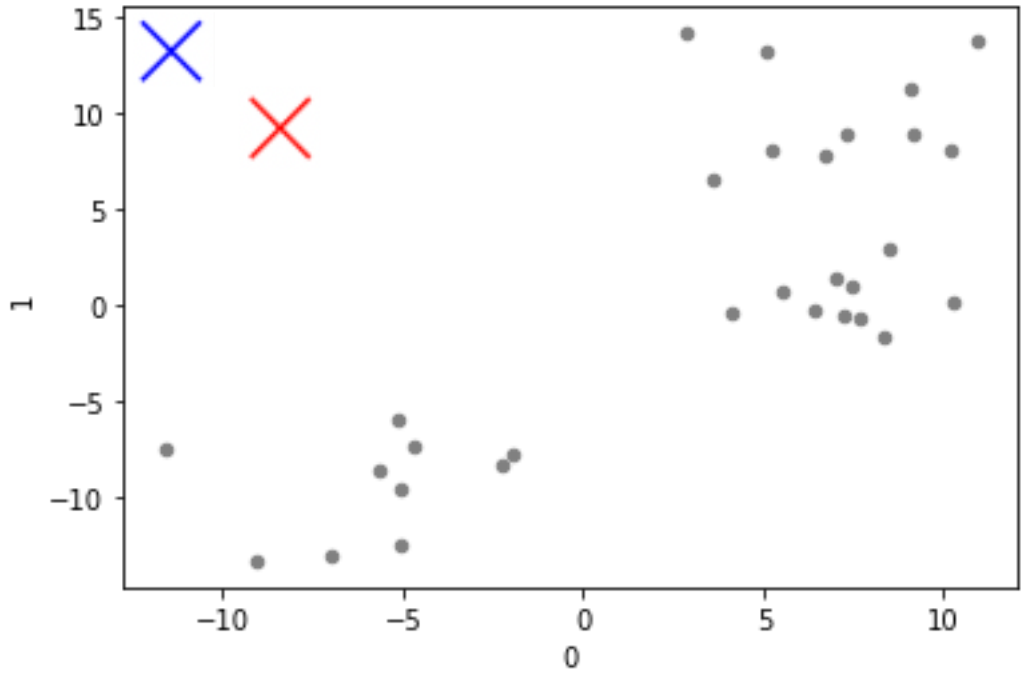
- *Create k points for starting centroids (e.g., randomly)*
- *While any point has changed cluster assignment*
  - *for every point in our dataset:*
    - *for every centroid*
      - *calculate the distance between the centroid and assign the point to the cluster with the lowest distance*
  - *for every cluster calculate the mean of the points in that cluster assign the centroid to the mean*

Initialization Strategies in Kmeans

- Initialization is the "art" of choosing the initial centroids
- It has an important impact on the clustering results

Initialization Illustration

What's the issue with this initialization?



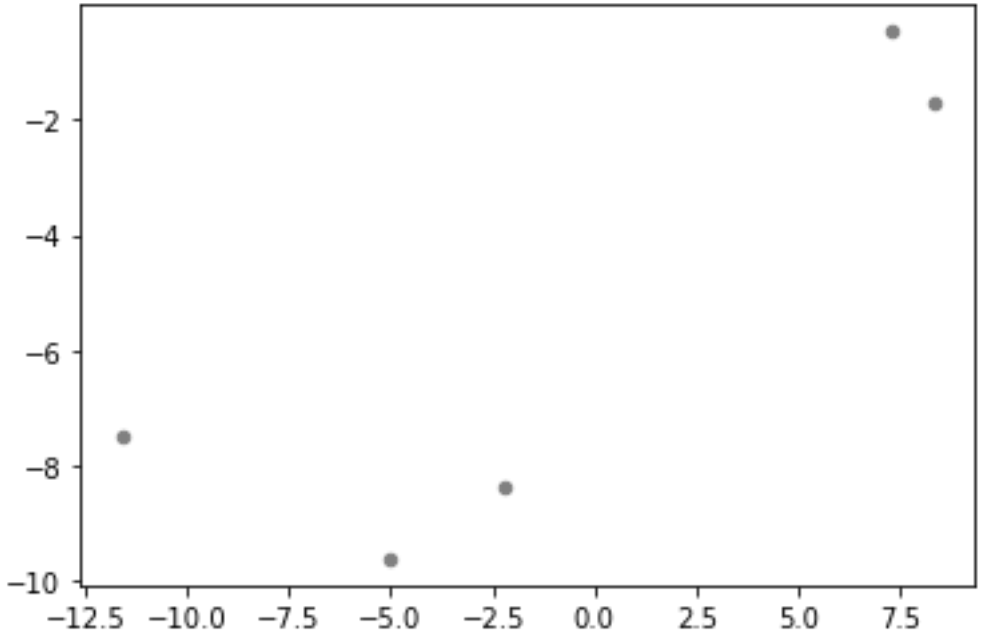
All points will be assigned to the red centroid and the algorithm stops! (converging to a suboptimal solution)

- *While any point has changed cluster assignment*
  - *for every point in our dataset:*
    - *for every centroid*
      - *calculate the distance between the centroid and assign the point to the cluster with the lowest distance*
  - *for every cluster calculate the mean of the points in that cluster assign the centroid to the mean*

Initialization Strategies

- *https://medium.com/analytics-vidhya/comparison-of-initialization-strategies-for-k-means-d5ddd8b0350e*
- Initialization has an impact on clustering
- Strategies
  1. Random: Chooses any k points (may not be landing on any datapoints!)
  2. Forgy Initialization: Chooses any k points **from the data**
  3. kmeans++ Initialization: [next slide]

Kmeans++



Pseudo Algorithm [1]

1. Choose a point from the data (i.e., Forgy)
2. For each data point x not chosen yet, compute $D(x)$, the distance between x and the nearest center that has already been chosen.
3. Choose one new data point at random with probability proportional to $D(x)^2$
4. Repeat Steps 2 and 3 until k centers have been chosen.

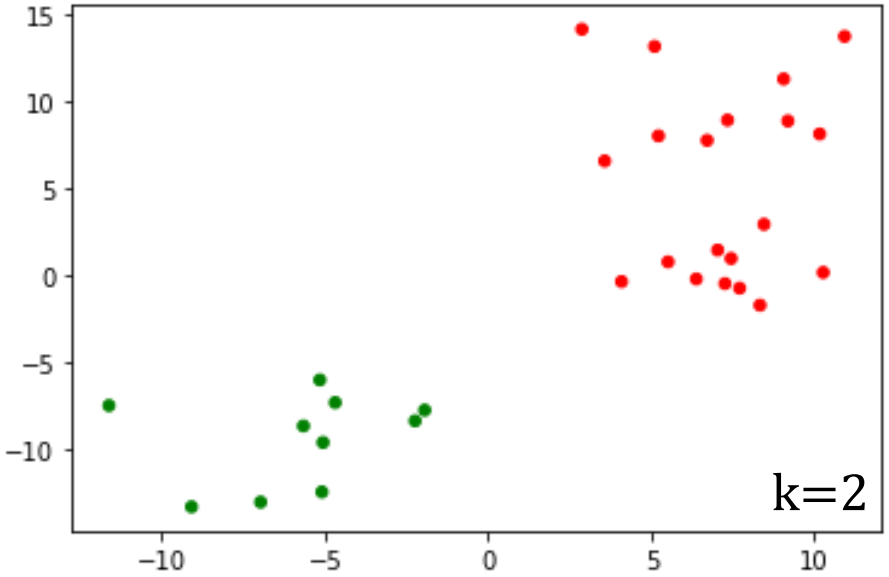
https://en.wikipedia.org/wiki/K-means%2B%2B

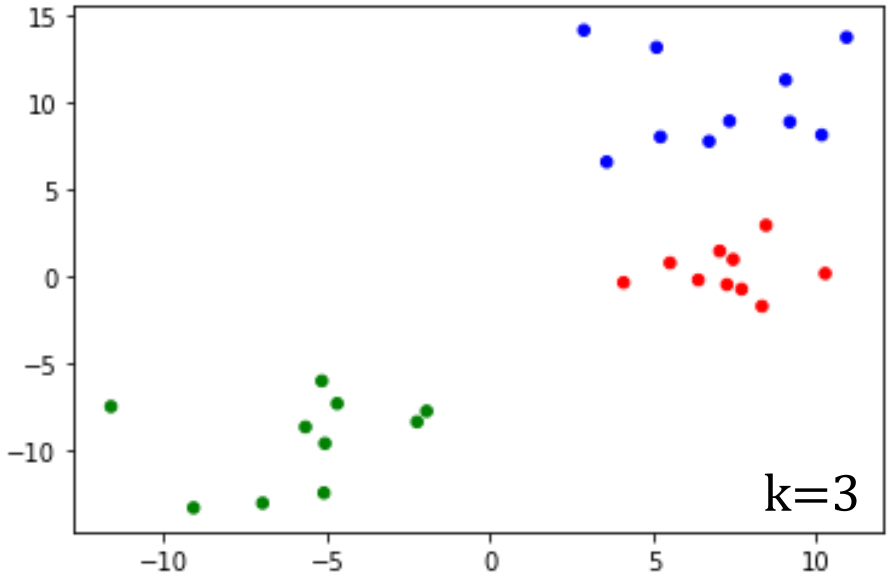
Kmeans++

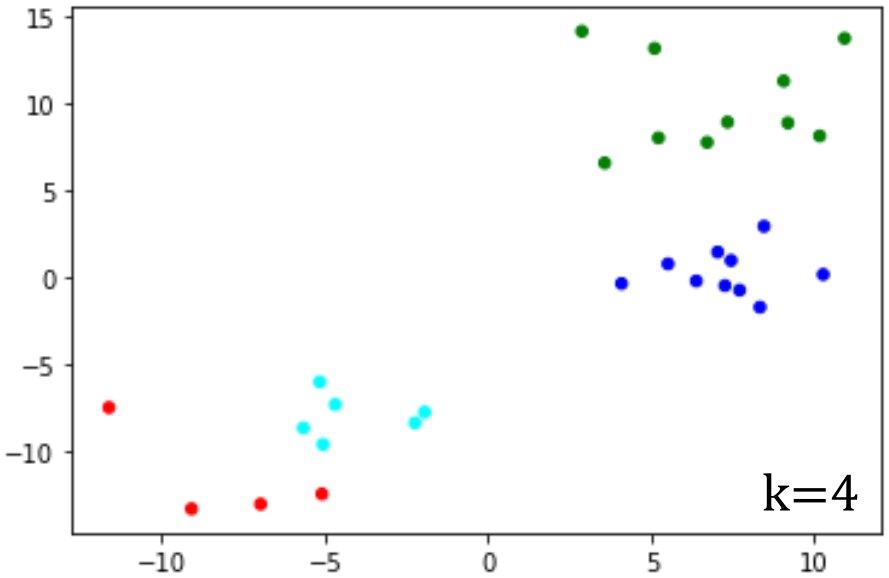
- Let's say we have a dataset of 6 points and want to cluster them into 2 groups using K-means++.
- Our 2D dataset is: {(1, 3), (2, 5), (3, 4), (6, 2), (7, 3), (8, 1)}
- We randomly select the first centroid from the dataset, let's say it's point (1, 3).
- We calculate the distance between each data point and the first centroid and choose the second centroid **randomly** with a probability proportional to the distance squared to (1, 3).
- Let's say (6, 2) is selected as the second centroid.
- We then use these centroids to cluster the remaining data points using the standard K-means algorithm.

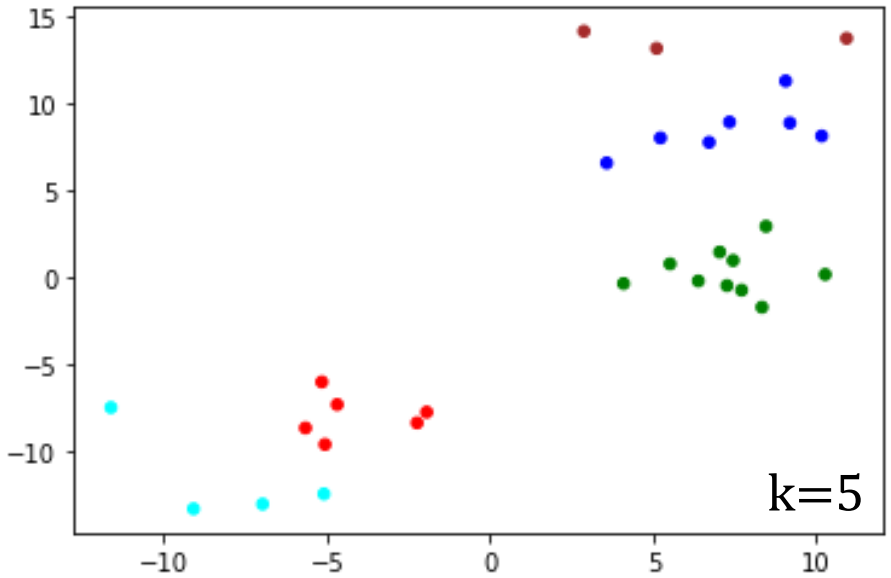
How to choose k?

Which one is best?









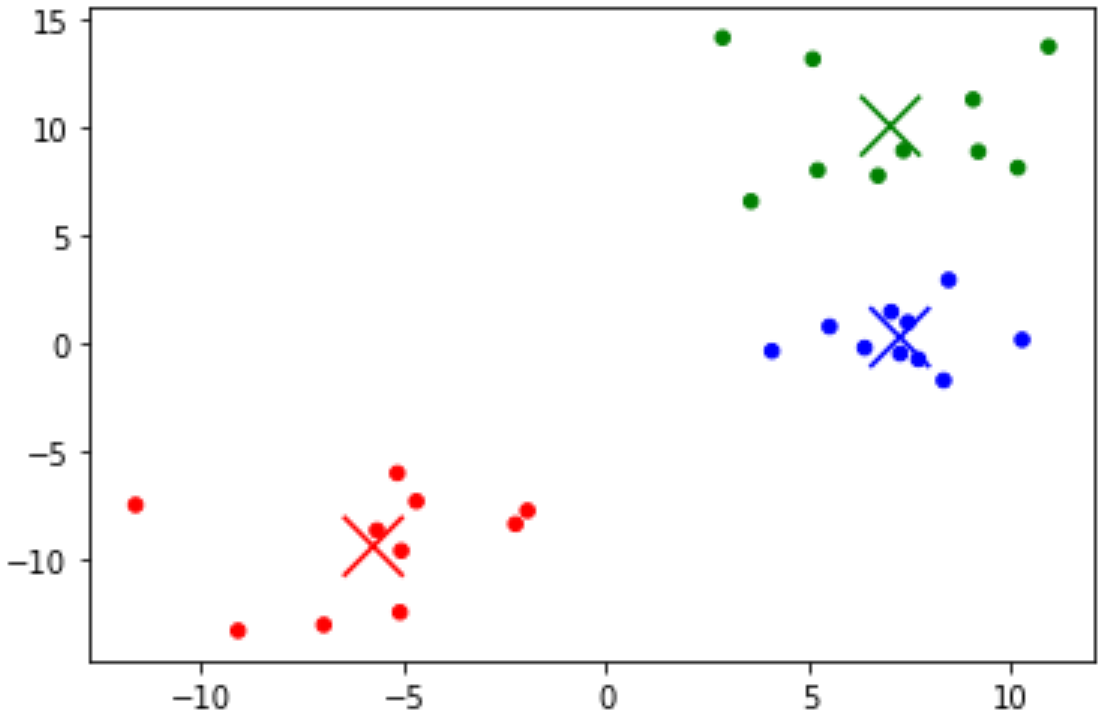
Methods to Choose the Best k

- Elbow method
- Silhouette Score

Elbow Method

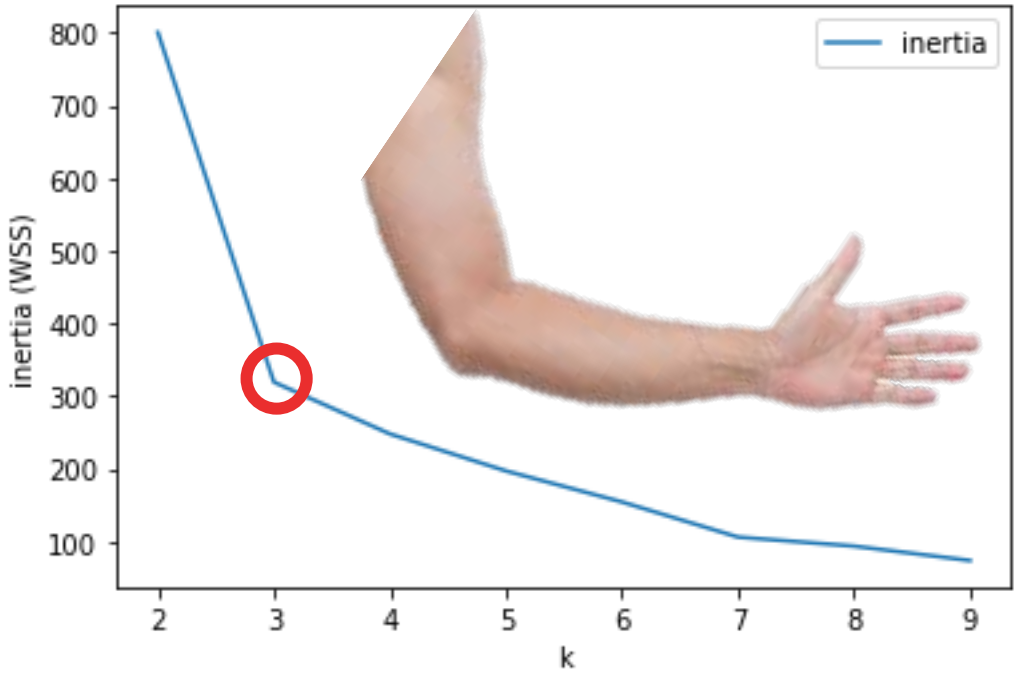
Within-Cluster-Sum of Squared Errors (WSS)

- Distance of all points to its centroid (called inertia in scikit-learn)



Elbow Method

- The elbow is a heuristic helping in choosing a good k



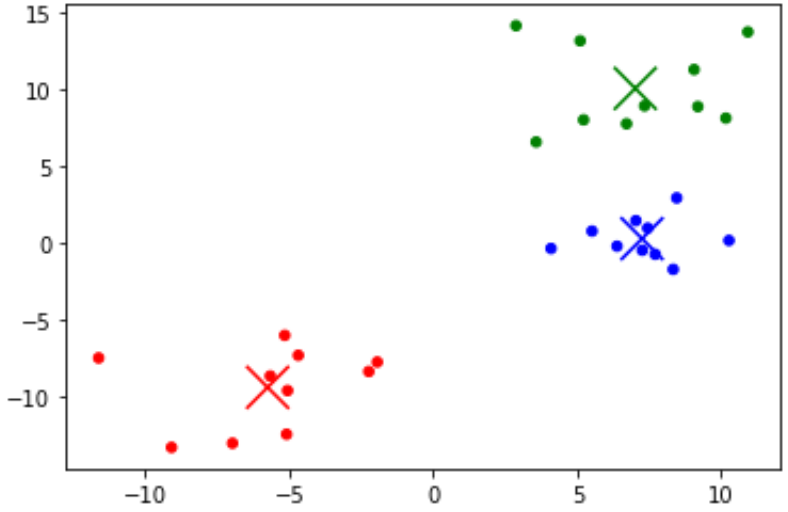
Pros and cons

- Pros:
  - Easy to compute
  - Quick Heuristic
- Cons:
  - The elbow is not always easy to spot

Silhouette Score

More robust approach than the Elbow Method

Silhouette Score



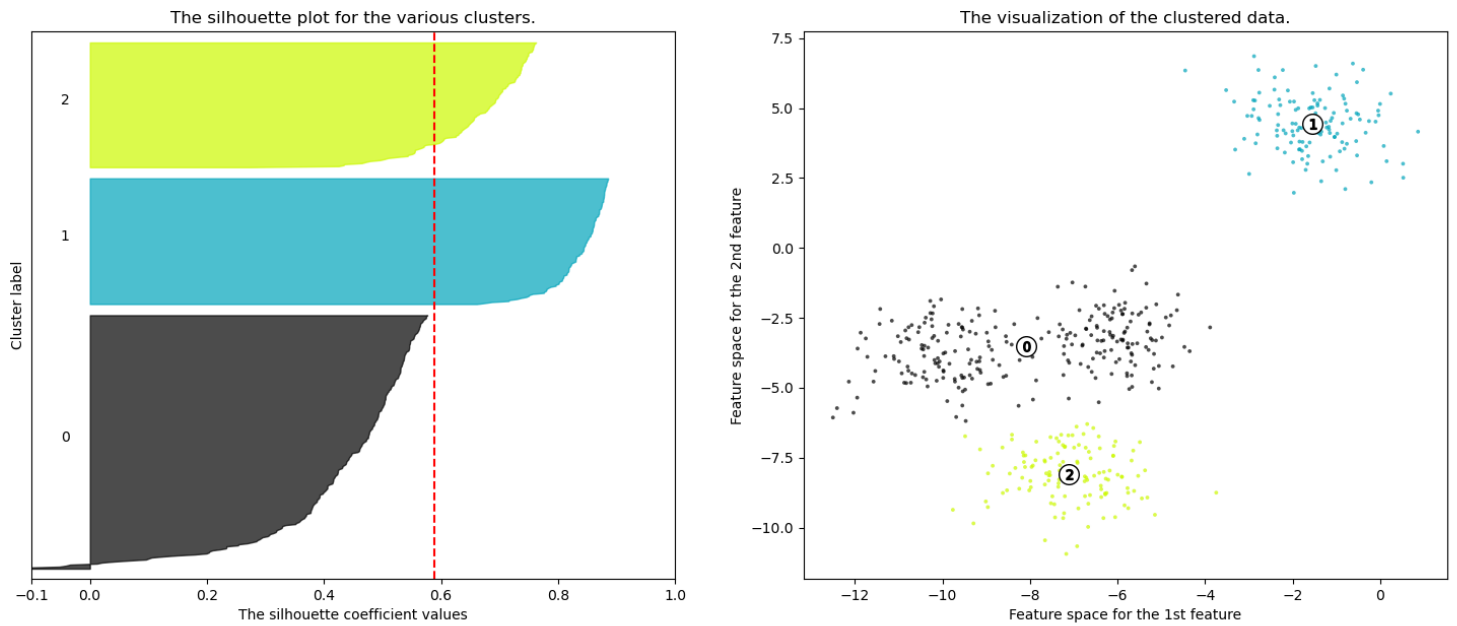
- Silhouette score:
  - Determines how well an observation, o, lies within its cluster

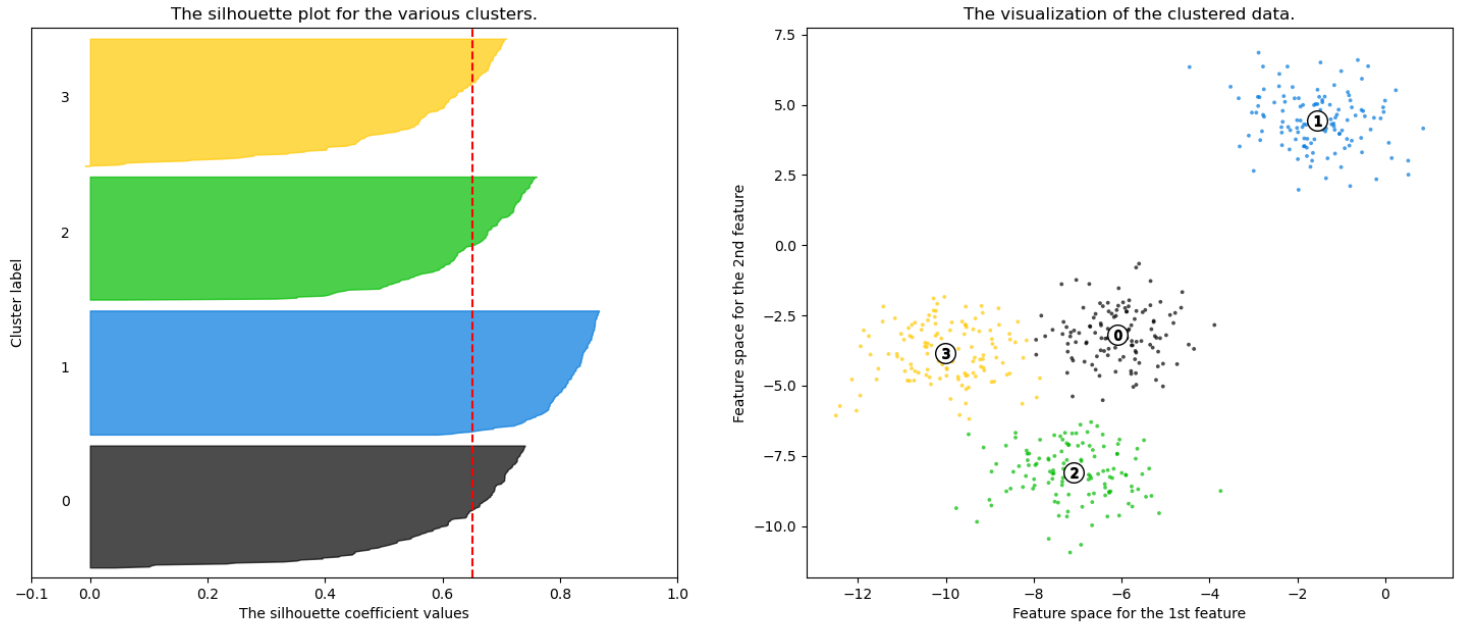
- Compute the average distance between the data point and all other data points within the same cluster (a).
- Compute the average distance between the data point and all data points in the nearest neighboring cluster (b).
- Compute the silhouette score for the data point as :

$$silhouetteScore = \frac{b(o) - a(o)}{\max(b(o), a(o))}$$

$$\text{a}() = \text{mean intra-cluster distance}$$

$$\text{b}() = \text{mean nearest-cluster distance}$$

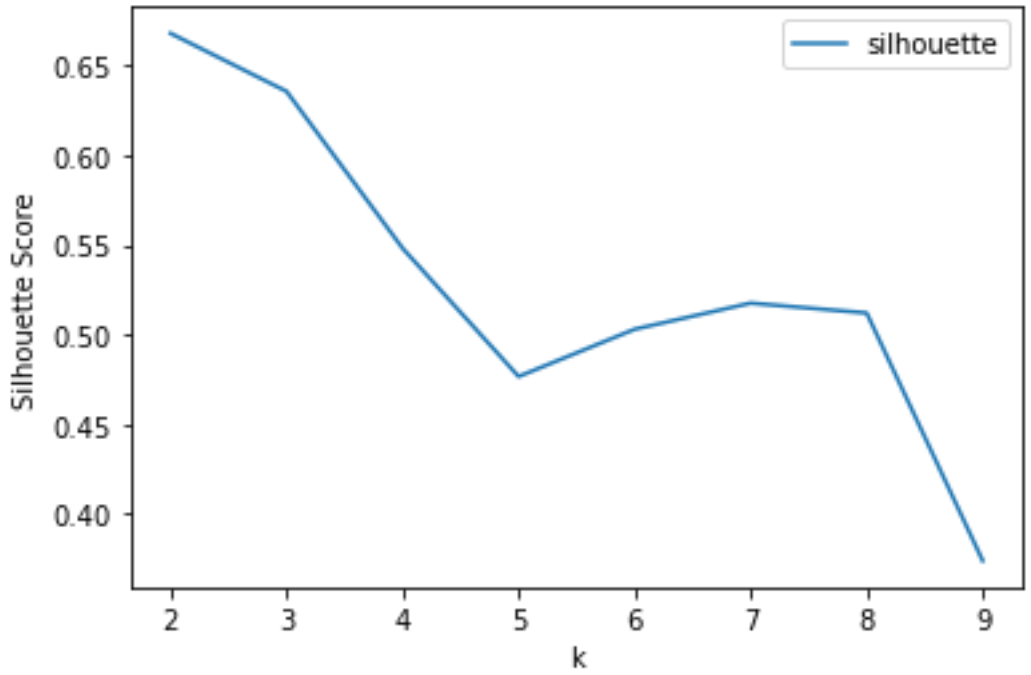




https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Comparing Silhouette for Various k

- Mean of the silhouette score of all observations
- The higher, the better



Pros and cons

- Pros:
  - Numbers are directly comparable for various k
  - The score is bounded between -1 for incorrect clustering and +1 for highly dense clustering.
- Cons:
  - More computation required

Wrap-up

- Kmeans (unsupervised)
- Initialization strategies
  - Random
  - Forgy
  - Kmeans++
- Methods to choose k
  - Elbow
  - Silhouette