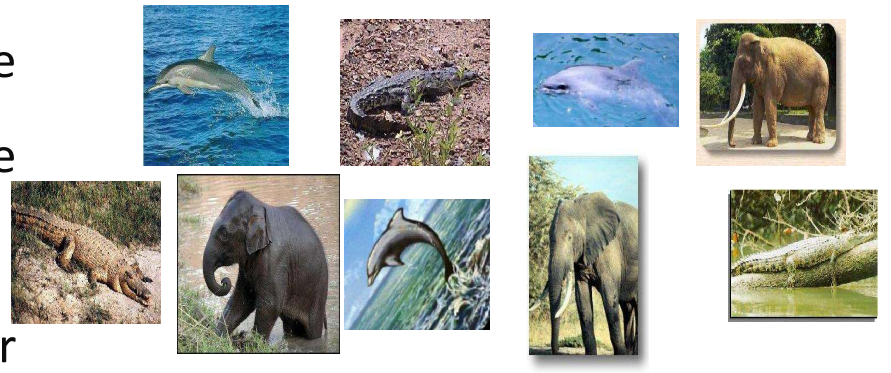
Unsupervised Learning

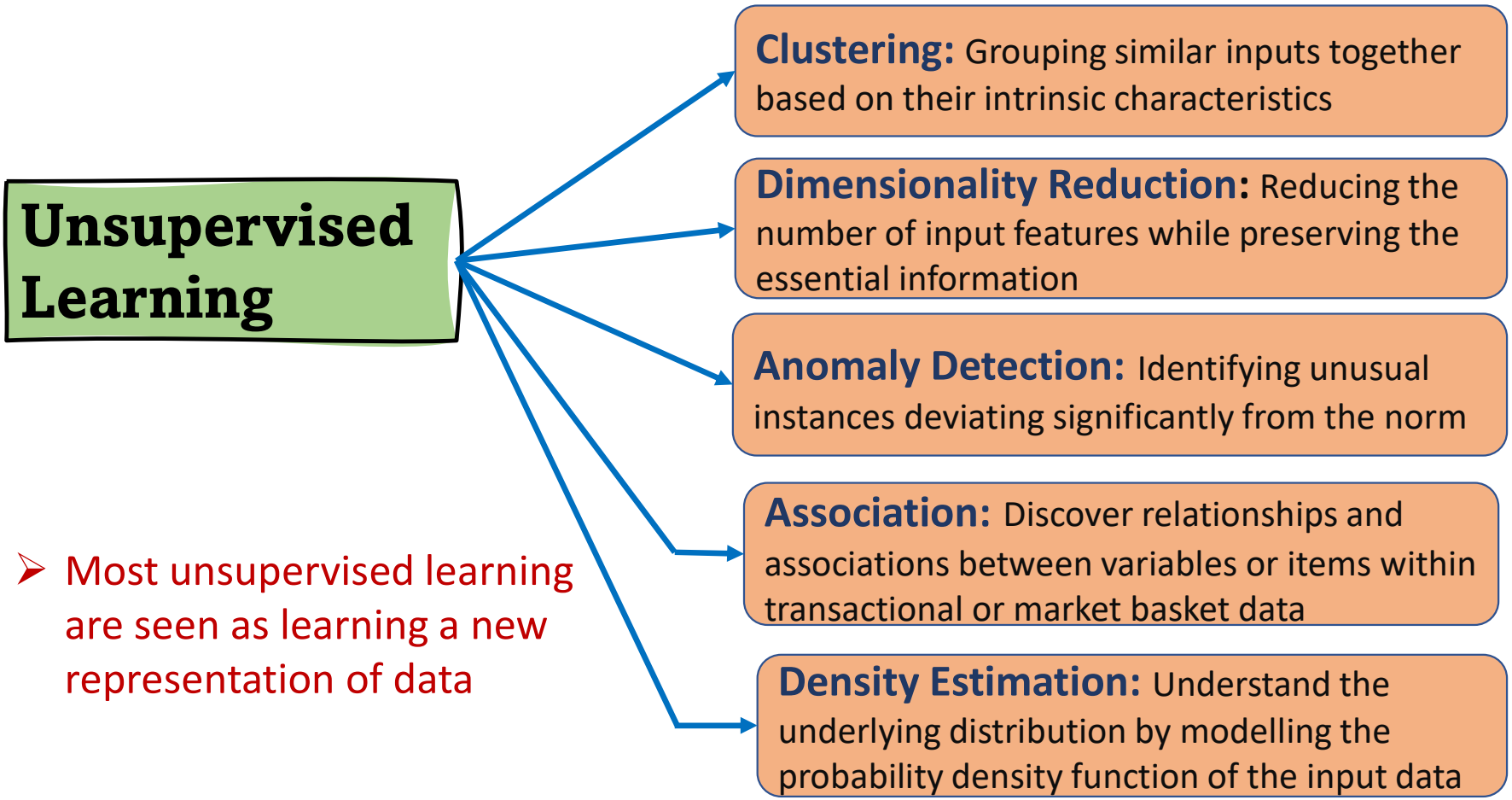
K-Means Clustering

Unsupervised Learning

- Models are trained using unlabeled dataset and are allowed to learn interesting/useful structures in the data without any supervision.
- The data can be categorized based on their similarities and dissimilarities.
- Unsupervised methods are used to find features which can be useful for categorization.
- Used to gain insight into the nature or structure of the data.



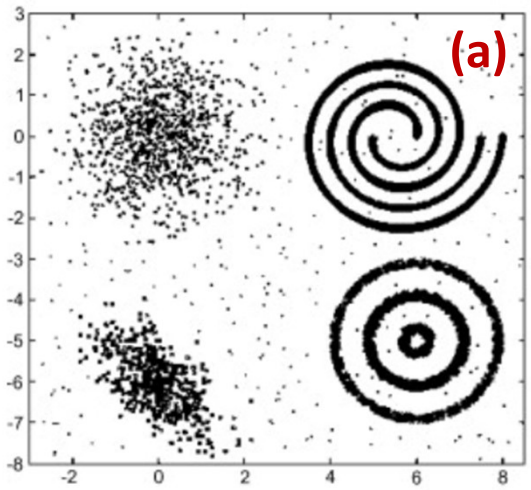
Types of Unsupervised Learning

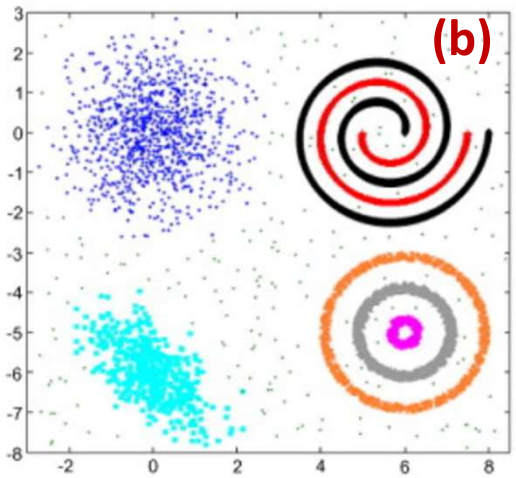


- Most unsupervised learning are seen as learning a new representation of data

Clustering

- Clustering is the partitioning of N unlabelled samples $\boldsymbol{x_1, x_2, \ldots, x_N}$ of the dataset into K "homogeneous" **partitions**, based on some notion of **similarity**





The seven clusters in (a) (denoted by seven different colors in (b)) differ in shape, size, and density.

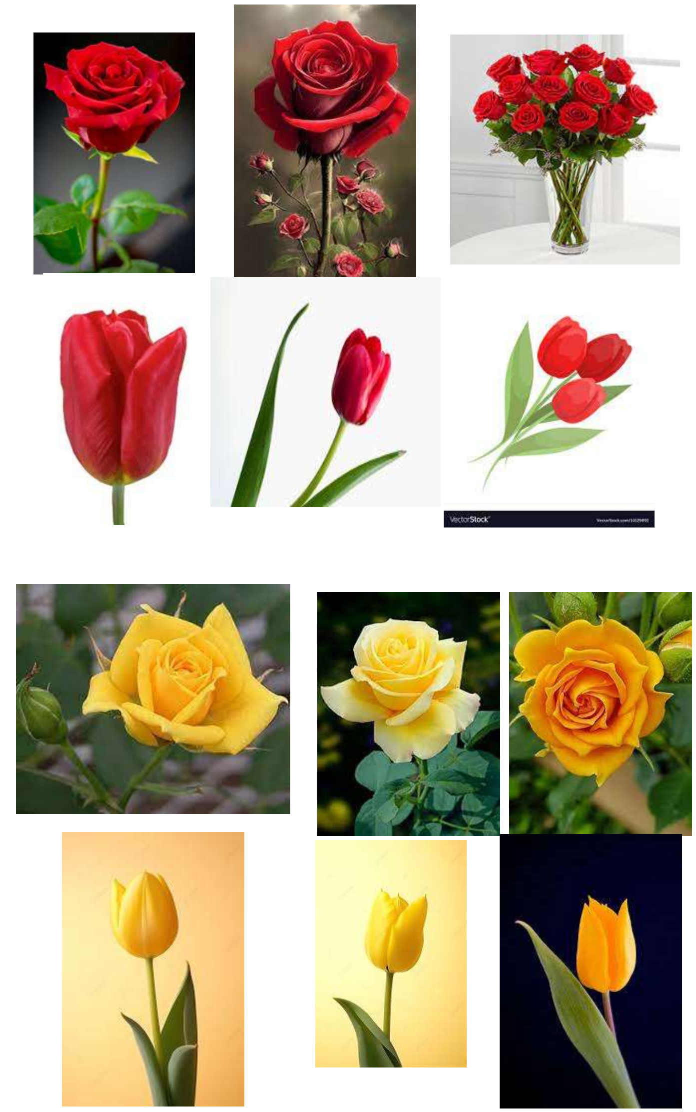
- *K* groups clustering achieved based on a measure of *similarity*
  - Similarities between objects within the cluster are high
  - Similarities between objects in different clusters are low

Clustering

- Similarity can be subjective, as without labels similarity can be hard to define

- Clustering colors based on shades: Two shades of blue might be perceived as very similar by one person and quite different by another.
- Music: Some group songs based on rhythm, while others focus on lyrical themes or emotional content
- Text documents: Grouping due to different interpretations of semantic meaning, sentiment, or specific keywords.
- Images: Visual similarity, group images of animals together based on their shapes, while another might prioritize color or background.

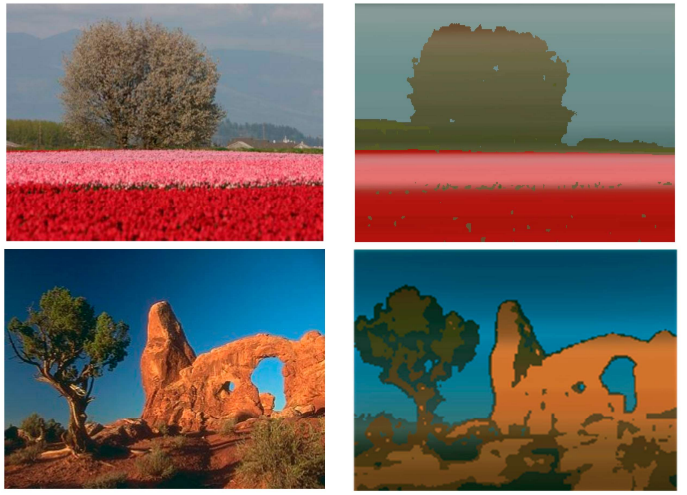
Example



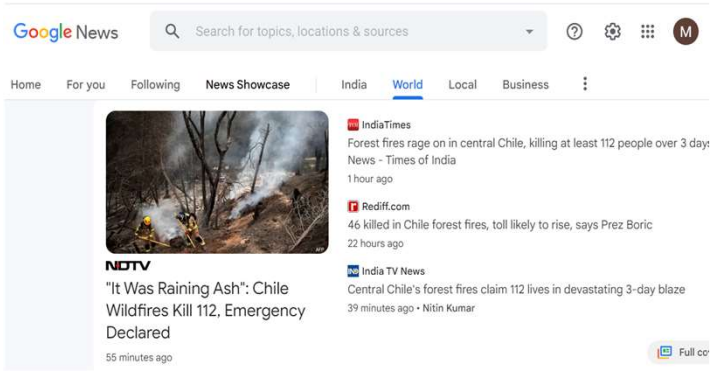
Clustering - Applications

- Image Segmentation: Break up the image into meaningful or perceptually similar regions [clustering pixels]
- Documents clustered to generate topical hierarchies for efficient information access or retrieval
- Customer segmentation for efficient marketing to group services delivery engagements for workforce management and planning
- Cluster the search results and present them in a more organized way to the user

Image Source: https://doi.org/10.3390/jimaging5030038

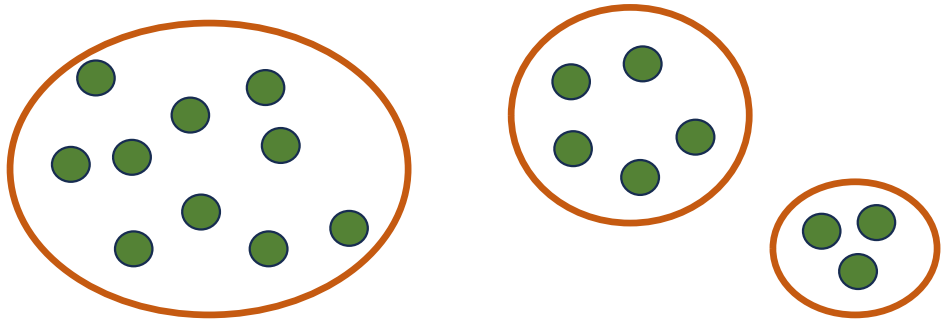


(a) (b)

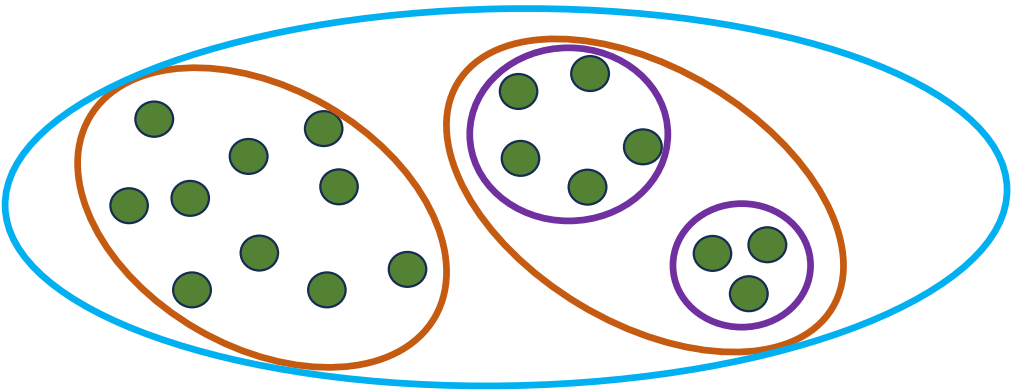


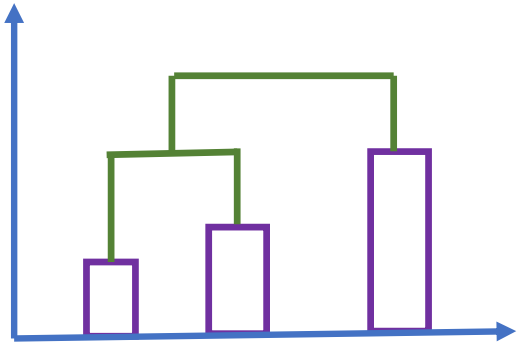
Clustering - Types

- **Partitioning Clustering**: Divides the dataset into a set number of non-overlapping clusters



- **Hierarchical Clustering**: Divides the dataset into a tree-like structure of clusters. Begins with individual data points as separate clusters and merges or splits them iteratively.



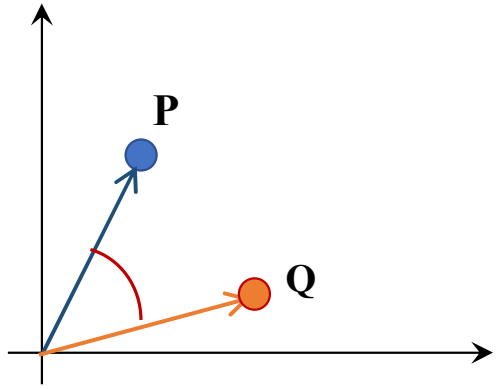


Clustering: Similarity Measures

- Vectors: Cosine distance.

$$\boldsymbol{s(P,Q) = \cos\theta = \frac{\mathbf{P}.\mathbf{Q}}{\|\mathbf{P}\|\|\mathbf{Q}\|}}$$

$$d(P,Q) = 1 - s(P,Q)$$



- Quantifies the angle and direction between the vectors rather than their magnitude

Clustering: Similarity Measures

- Jaccard distance: Measure of dissimilarity between two sets of features or attributes in datasets.
- The Jaccard Distance (JD) between two sets A and B is calculated as:

  JD = 1 – Jaccard Similarity (A,B)

- The Jaccard Similarity coefficient between set A and B:

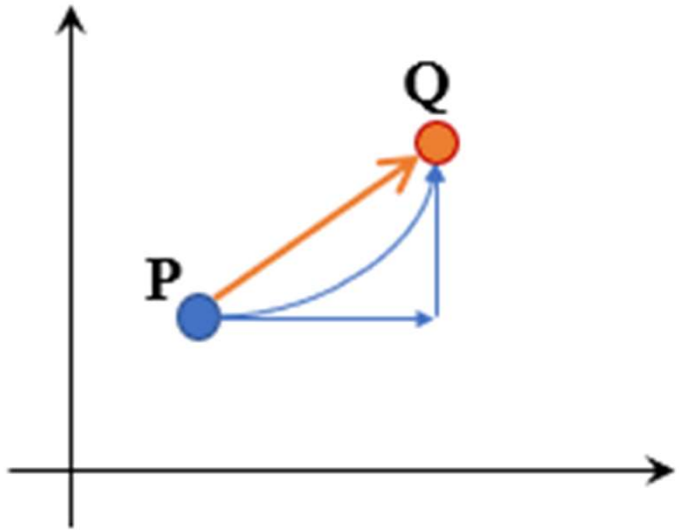
$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

- The Jaccard distance ranges from 0 to 1, where 0 indicates identical sets (perfect similarity), and 1 indicates completely dissimilar sets (no common elements).

Clustering: Similarity Measures

- Points: Minkowski distance
  - q=2: Euclidean distance
  - q=1: City-block distance

$$d(x, x') = \left(\sum_{k=1}^{d} |x_k - x'_k|^q \right)^{1/q}$$



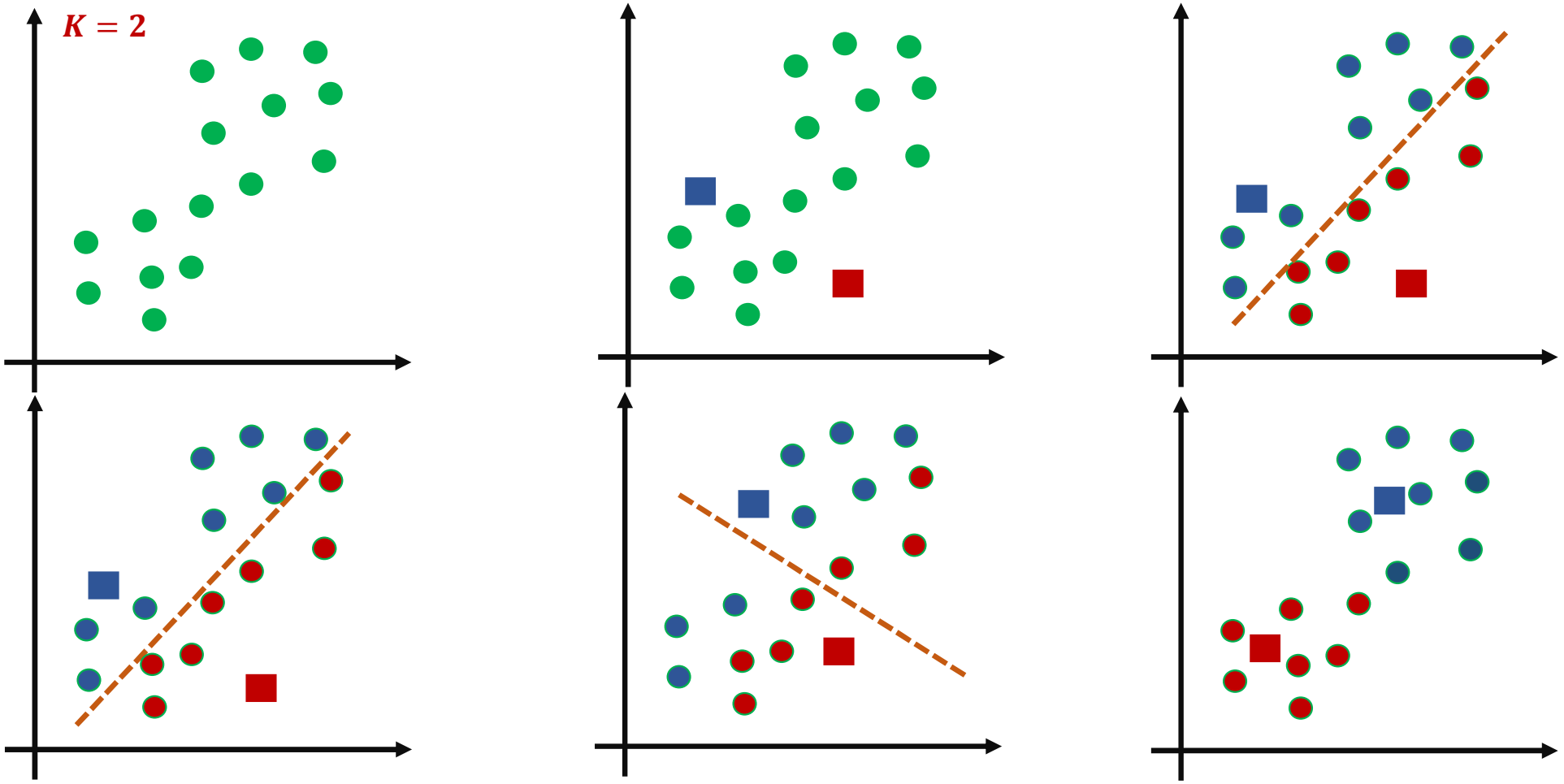
K-means Lloyd's Algorithm

- Algorithm organizes data into clusters such that there is high intra-cluster similarity and low inter-cluster similarity.
    - A datapoint will belong to one cluster, not several – resulting in a specific number of disjoint non-hierarchical clusters

- Assume we have inputs $\boldsymbol{x_1, x_2, \ldots, x_N}$ and a predefined value of K

1. Pick K random points as cluster centres (centroids).
2. Each point x_i is assigned to nearest cluster by calculating its distance to each centroid.
3. The new cluster centres are calculated by taking the average of the assigned points.
4. Steps 2 and 3 are repeated until none of the cluster assignments/means changes.

K-Means Algorithm



Squared Error-Loss Function

- Finds a partition such that the squared error between the empirical mean of a cluster and the points in the cluster is minimized

  Mean of cluster $(\boldsymbol{c_k}) = \boldsymbol{\mu_k} = \frac{\mathbf{1}}{\boldsymbol{N_{c_k}}} \sum_{\boldsymbol{x_i}\, \epsilon\, \boldsymbol{c_k}} \boldsymbol{x_i}$

- Let $\boldsymbol{\mu_1}, \boldsymbol{\mu_2}, \ldots, \boldsymbol{\mu_K}$ be the K cluster centroids/means

- Define $\delta_{ik}\, \epsilon\, \{0,1\}$, with $\boldsymbol{\delta_{ik} = 1}$, if x_i belongs to cluster k, otherwise 0
  - ✓ $\boldsymbol{\delta_i} = [\boldsymbol{\delta_{i1}}, \boldsymbol{\delta_{i2}}, \ldots, \boldsymbol{\delta_{iK}}]$ denotes a length K one-hot encoding of $\boldsymbol{x_i}$

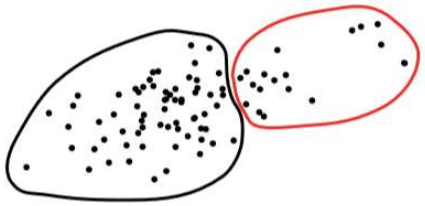
Loss Function

- The Loss Function for $\boldsymbol{x_i}$: $\boldsymbol{l(\mu, x_i, \delta_i) = \sum_{k=1}^{K} \delta_{ik} \|x_i - \mu_k\|^2}$
- The overall Loss Function: $\boldsymbol{L(\mu, x_i, \delta_i) = \sum_{i=1}^{N} \sum_{k=1}^{K} \delta_{ik} \|x_i - \mu_k\|^2 = \|X - \Delta \mu\|^2}$

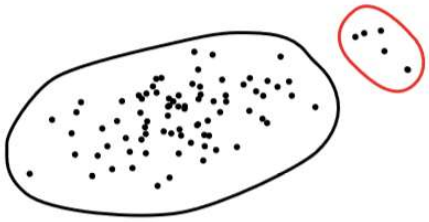
  ✓ $\boldsymbol{X}$ is $N \times D$
  ✓ $\boldsymbol{\Delta}$ is $N \times K$ (each row is a one-hot δ_i or equivalently $\delta_i \in \{1,2, \dots, K\}$)
  ✓ $\boldsymbol{\mu}$ is $K \times D$ (each row is a μ_k)

- $\boldsymbol{L(\mu, x_i, \delta_i) = \sum_{i=1}^{N} \sum_{i: z_i = k} \|x_i - \mu_k\|^2}$

Measures within cluster variance (Sum-of-Squared-Error)



$J_e = small$



$J_e = large$

Loss Function- Optimization

- $\arg\min_{\Delta,\mu} L(\mu, x_i, \delta_i) = \arg\min_{\Delta,\mu} \sum_{i=1}^{N} \sum_{k=1}^{K} \delta_{ik} \|x_i - \mu_k\|^2$,

  such that $\delta_{ik} \epsilon \{0,1\}$ and $\sum_{k=1}^{K} \delta_{ik} = 1$

- The K-means aims to minimize this overall loss function w.r.t. μ and δ
  - Initialize the K mean-vector μ_k randomly (e.g., choosing any K data points as the mean vectors)

- Expectation-step [Find the expected point associated with a cluster]: minimize L w.r.t. δ_{ik}
  - Set $\delta_{ik} = 1$ for cluster index k corresponding to the smallest $\|x_i - \mu_k\|^2$ i.e., closes cluster mean (centroid)

- Maximization step [Improves the estimation of the cluster using E-step] : minimize L w.r.t. μ_k
  - Set $\frac{\partial L}{\partial \mu_k} = 0 \Rightarrow \mu_k = \frac{\sum_{i=1}^{N} \delta_{ik} x_i}{\sum_{i=1}^{N} \delta_{ik}}$ i.e., re-computing the mean.

- Minimizing the K-Means objective function involves finding the optimal cluster assignments and centroids → is known to be an NP-hard problem. $O(NKDL)$

K-means - Hyperparameters

- The K-means algorithm requires three user-specified parameters:

  a. number of clusters K: Run K-means independently for different values of K and select the partition that appears the most meaningful depending on the problem

  b. cluster initialization: Different initializations leads to different final clustering as K-means only converges to local minima. Run, for a given K, with multiple different initial partitions and choose the partition with the smallest loss.

  c. distance metric: Typically Euclidean metric is used for computing the distance between points and cluster centers. K-means therefore finds spherical or ball-shaped clusters in data

Clustering: Evaluation

- Supervised classification have a variety of measures to evaluate how good our model is e.g., Accuracy, precision, recall

- Clustering being an unsupervised task, how to evaluate the “goodness” of the resulting clusters?

- But “clusters are in the eye of the beholder” - Estivill-Castro !

- If true, then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

Clustering: Evaluation (Elbow Method)

- SSE is good for comparing two clusters (average SSE/WCSS).

$$L(\mu, x_i, \delta_i) = \sum_{i=1}^{N} \sum_{x_i \epsilon\, k} \|x_i - \mu_k\|^2$$

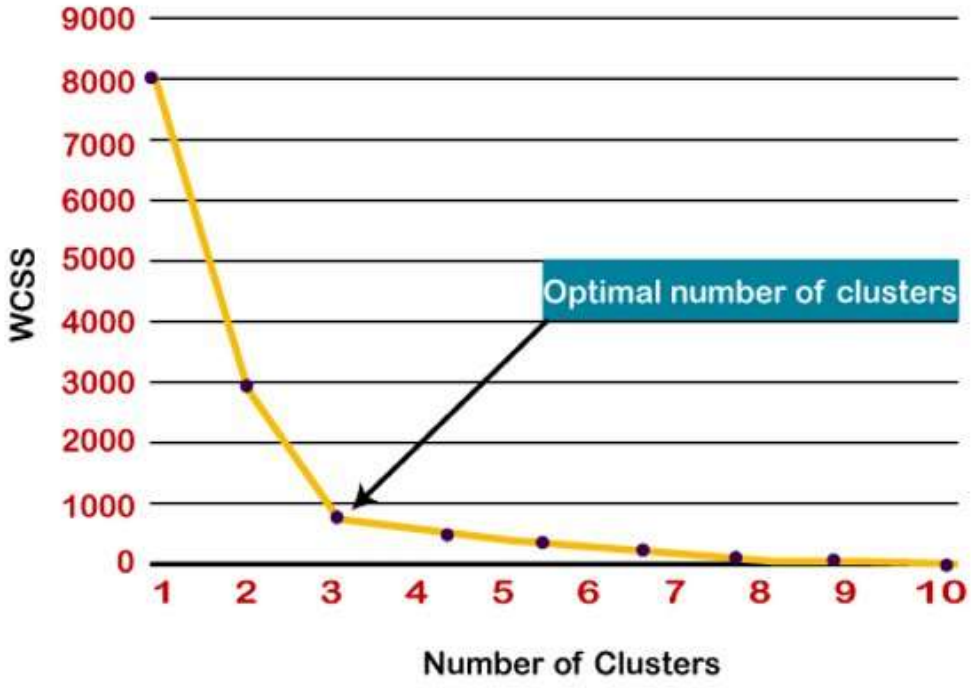
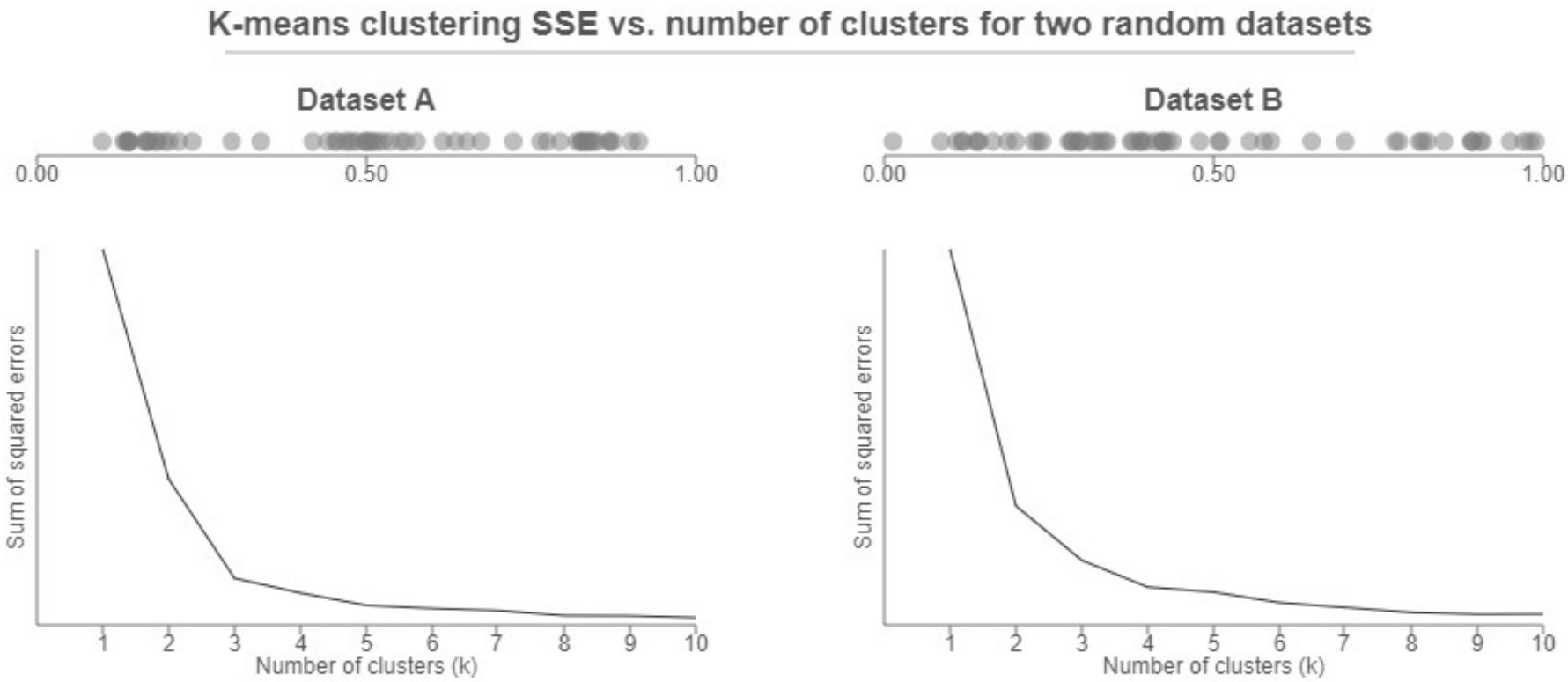


Image Source: Wikipedia



https://medium.com/analytics-vidhya/how-to-determine-the-optimal-k-for-k-means-708505d204eb

Silhouette Coefficient

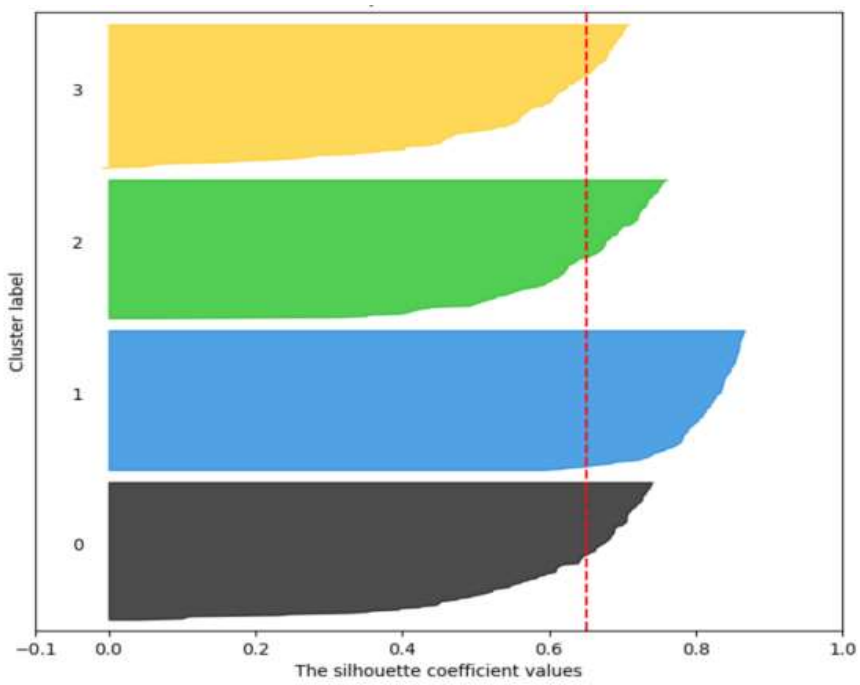
- The Silhouette value measures how similar a point is to its own cluster (cohesion) compared to other clusters (separation).
- The Silhouette function will compute the mean Silhouette Coefficient of all samples using the mean intra-cluster distance and the mean nearest-cluster distance for each sample.

$$S = \frac{n-i}{\max(i,n)}$$

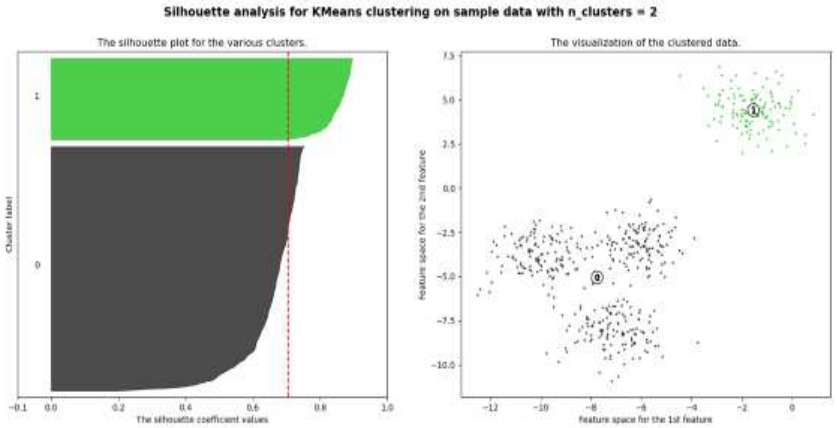
Here, i is intra-cluster distance and, n is mean nearest-cluster distance.

- The range of the Silhouette value is between +1 and -1. A **high value is desirable** and indicates that the point is placed in the correct cluster

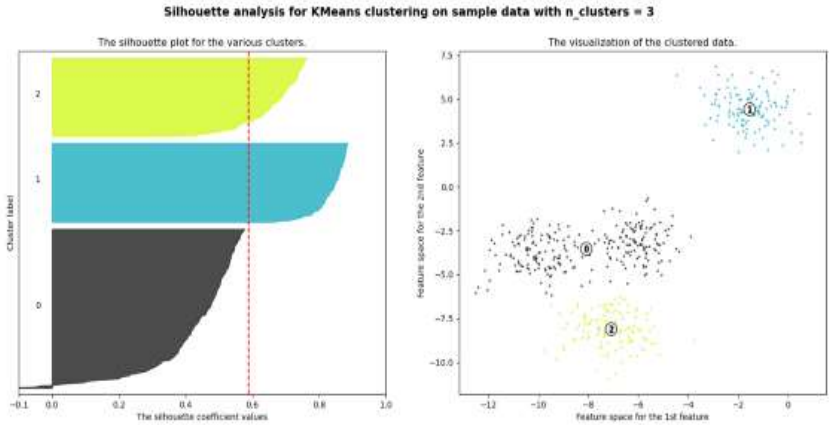
The cluster label is plotted on the y-axis, while the actual Silhouette Score on the x-axis. The size/thickness of the silhouettes is proportional to the number of samples inside that cluster



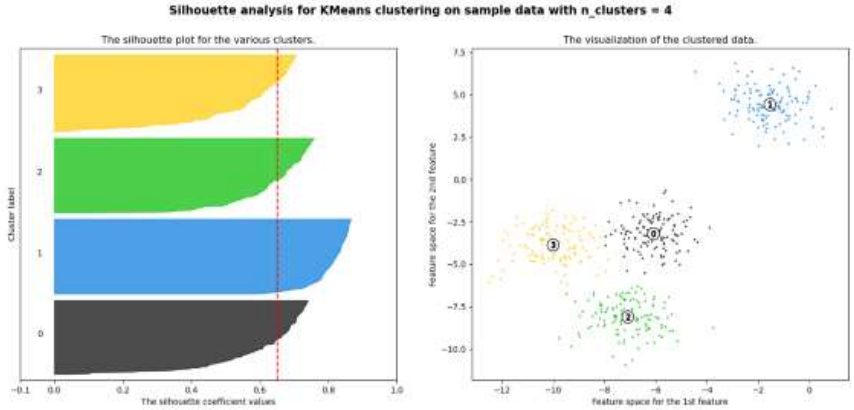
- For $n_{clusters}$ = 2, the average Silhouette Score is: 0.70



- For $n_{clusters}$ = 3, the average Silhouette Score is: 0.59



- For $n_{clusters}$ = 4, the average Silhouette Score is: 0.65



- For $n_{clusters}$ = 5, the average Silhouette Score is: 0.56

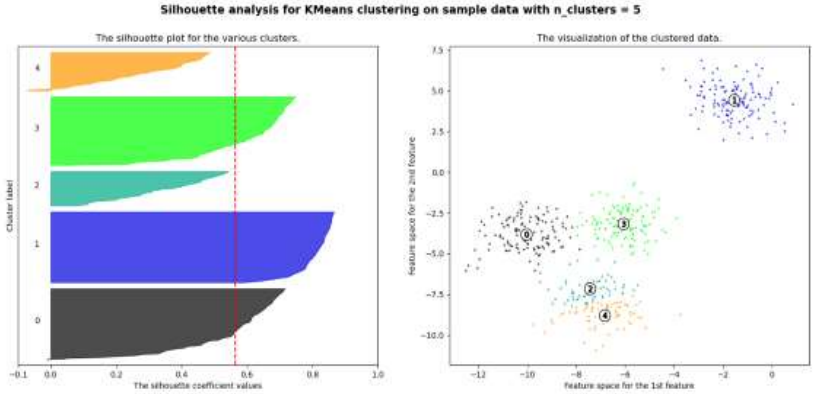


Image Source: https://towardsdatascience.com/