MODULE 3

Unsupervised learning

- Used on unlabeled data
- Goal is to learn about data's underlying structure

K-means

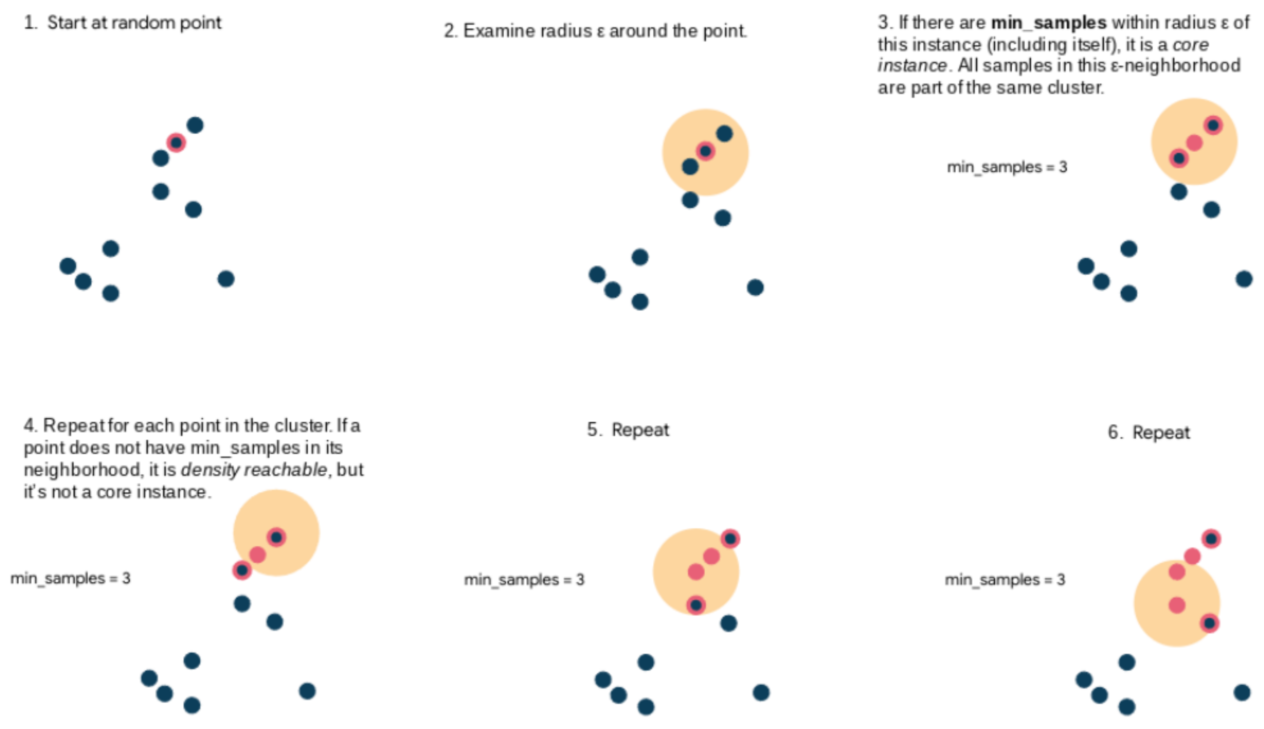
Unsupervised learning technique that groups unlabeled data into K clusters based on similarity

- Partitioning algorithm
- Steps:
  1. Initiate k centroid
  2. Assign all points to their nearest centroid
  3. Recalculate the centroid of each cluster based on points assigned
  4. Repeat step 2 and step 3 until reach convergence (stable point)
- Run the model with different centroid initialization to avoid poor clustering due to model converging in local minima
- Does not allow unassigned outliers

K-means++ randomly initializes centroids in the data based on probability calibration. It ensure that centroid aren't initially placed very close together avoiding convergence in local minima.

DBSCAN (density-based spatial clustering of applications with noise)

Searches the data space for continuous regions of high density. In other words, find clusters based on density



1. Start at random point

2. Examine radius ε around the point.

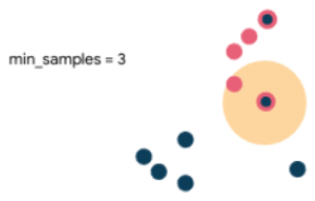
3. If there are **min_samples** within radius ε of this instance (including itself), it is a *core instance*. All samples in this ε-neighborhood are part of the same cluster.

4. Repeat for each point in the cluster. If a point does not have min_samples in its neighborhood, it is *density reachable*, but it's not a core instance.

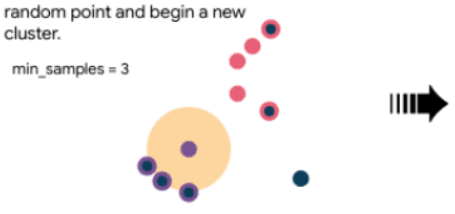
5. Repeat

6. Repeat

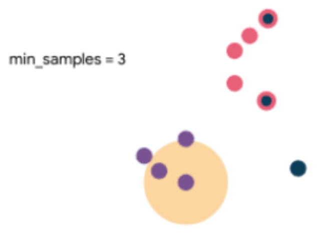
7. Repeat



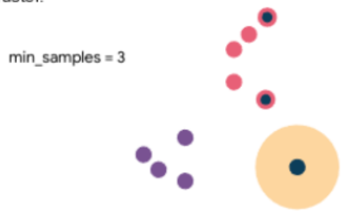
8. If there are no more points in the cluster, move to another random point and begin a new cluster.



11. Repeat



12. Points that don't have any core instances in their ε-neighborhood are considered noise and are not assigned to a cluster.

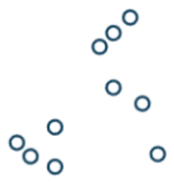


Where ε = radius of the search area, min_samples = number of samples in the search area

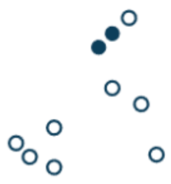
Agglomerative clustering

Works by first assigning every point to its own cluster, then progressively combing clusters based on intercluster distance.

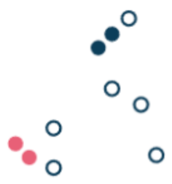
1. Each point is in its own cluster.



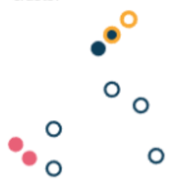
2. Merge the closest pair into a new cluster.



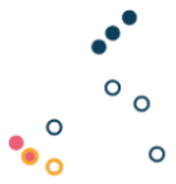
3. Repeat



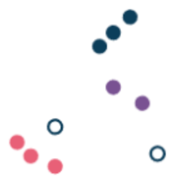
4. If one of the points in the next closest pair is already in a larger cluster, merge the unassigned point into the cluster



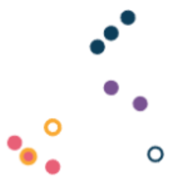
5. Continue



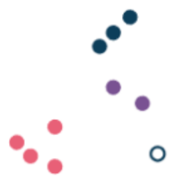
6. Continue



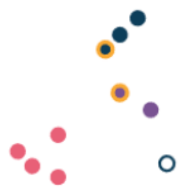
7. Continue



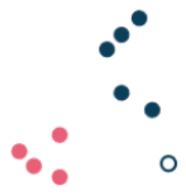
8. Continue (At this point, there are 4 clusters.)



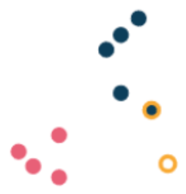
9. If the closest pairs are both part of larger clusters…



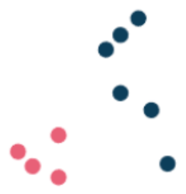
10. …merge the clusters. (Now there are 3 clusters.)



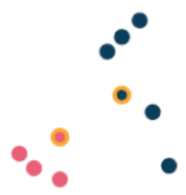
11. Continue



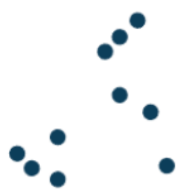
12. Continue (Now there are 2 clusters.)



13. Continue



14. One cluster (process cannot go further)

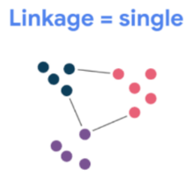


Agglomerative clustering requires to specify a desired number of clusters (distance threshold) which is the linkage distance above which clusters will not be merged.

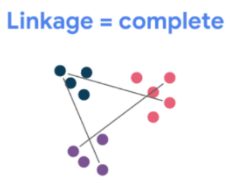
The model would converge into a single cluster every time.

Linkage – distances that determine whether or not to merge the clusters

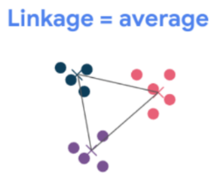
1. **Single:** Minimum pairwise distance between clusters



2. **Complete:** Maximum pairwise distance between clusters



3. **Average:** Distance between each cluster's centroid and other clusters' centroids



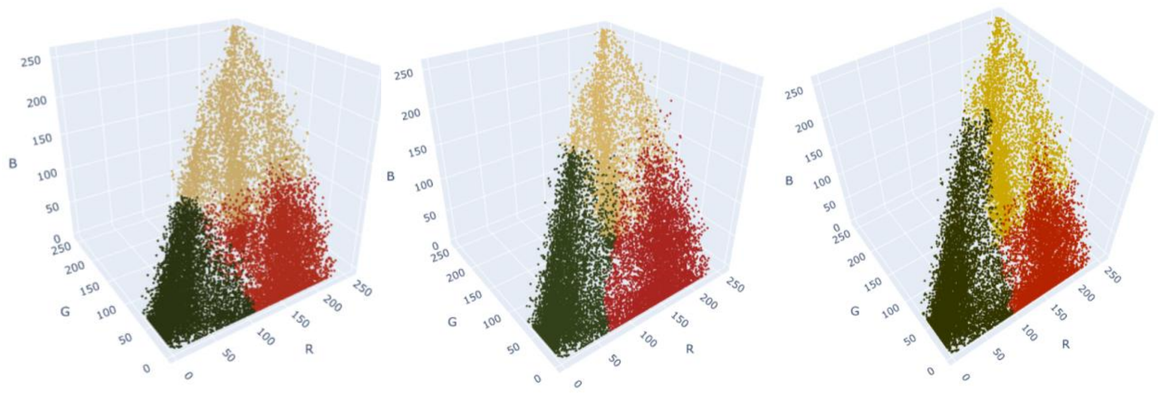
The agglomerative clustering algorithm will stop when one of the following condition is met:

1. Reached a specified number of clusters
2. Reached an intercluster distance threshold

Hyperparameters

- **n_clusters:** the number of clusters you want in the final model
- **linkage:** the linkage method to use to determine which clusters to merge
- **affinity:** the metric used to calculate the distance between clusters (default = Euclidean distance)
- **distance_threshold:** the distance above which clusters will not be merged

K-means **DBSCAN** **Agglomerative clustering**

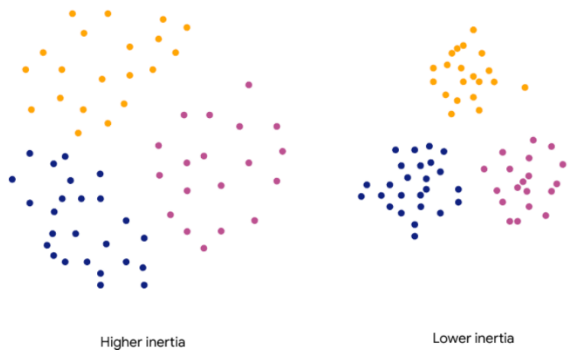


Inertia

- Sum of the square distances between each observation and its nearest centroid
- Measurement of how compact the clusters are in a model
- The more compact the clusters, the lower the inertia
- Helps to decide the optimal k value using **elbow method**

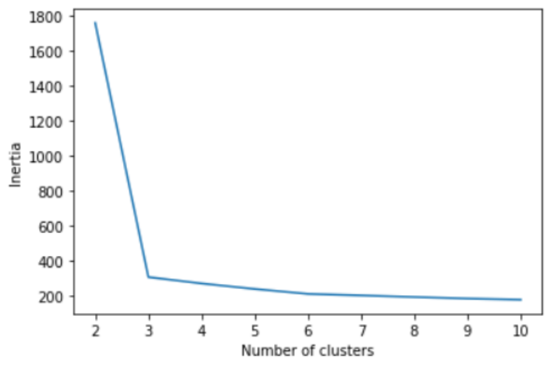
$$Inertia = \sum_{i=1}^{n}(x_i - c_k)^2$$

Where, x_i = location of a particular observations, c_k = location of the centroid k, which is the cluster to which point x_i is assigned



Elbow method

With K-means model, its done by comparing between number of clusters and inertia



Find the part of the curve with the sharpest bend

Silhouette Analysis

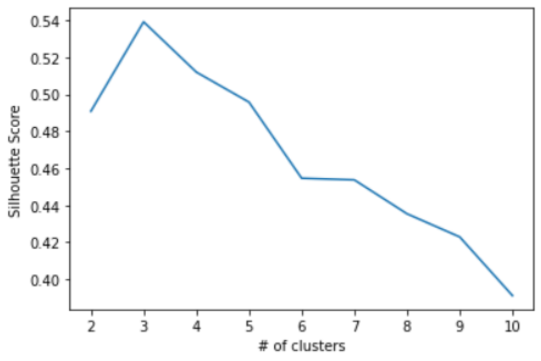
Comparison of different models' silhouette scores.

Silhouette score

- The mean of the silhouette coefficients of all the observations in the model
- The value are in range [-1,1]

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

Where a = mean distance between the instance an each other instance in the same cluster,
b = mean distance from the instance to each instance in the nearest other cluster (excluding the cluster that the instance is assigned to)



In the example above, it is probably best grouped into 3 clusters because the silhouette score is close to 1.