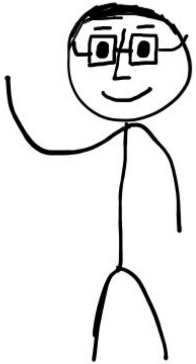
Cluster Analysis

Mohit Deshpande

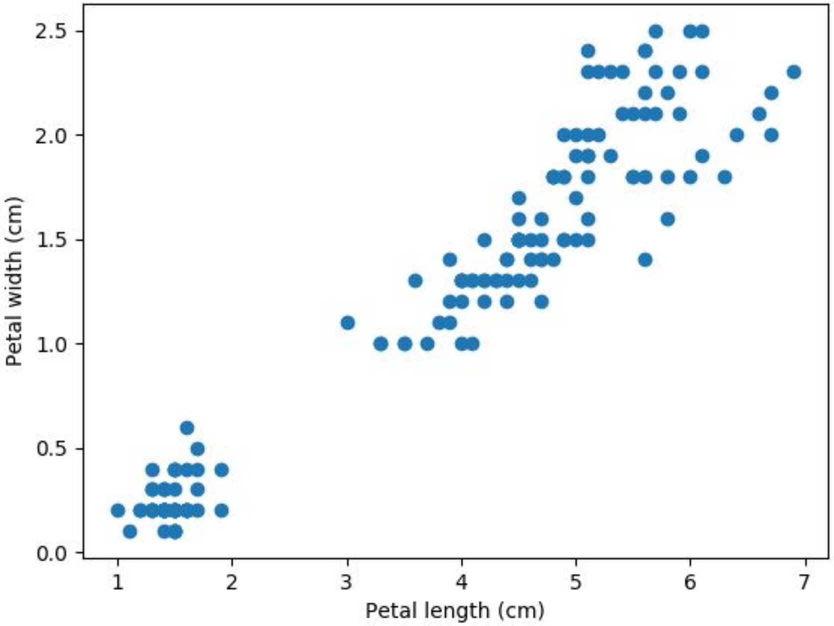
Cluster Analysis

1. Intro to Cluster Analysis
2. k-means Clustering
3. Density-based Spatial Clustering of Applications with Noise (DBSCAN)
4. Hierarchical Agglomerative Clustering (HAC)



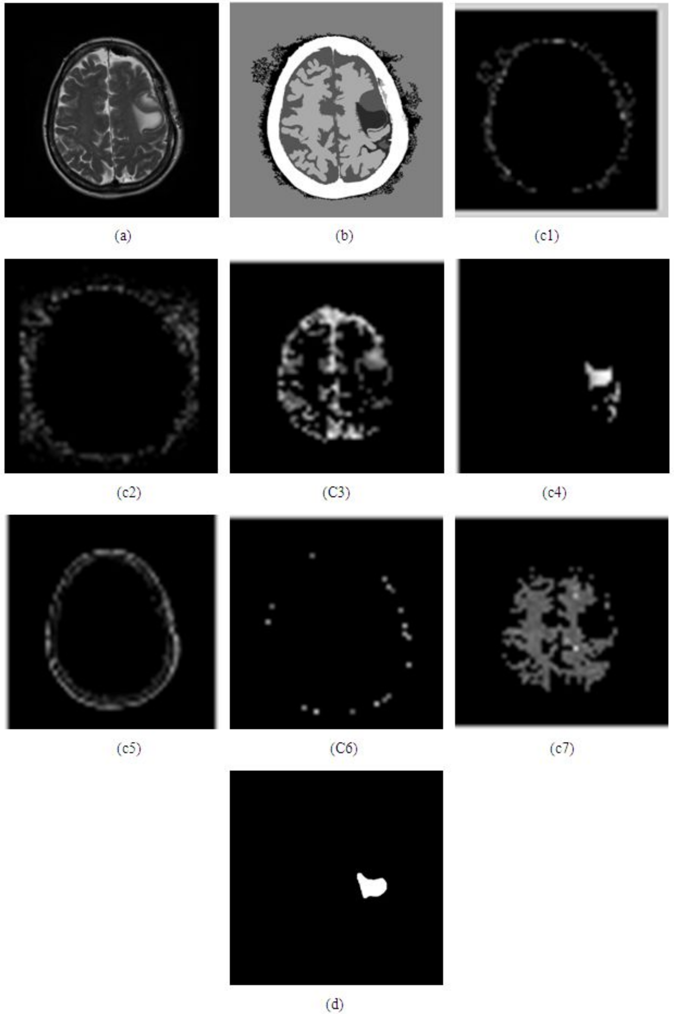
Intro to Cluster Analysis

Cluster Analysis

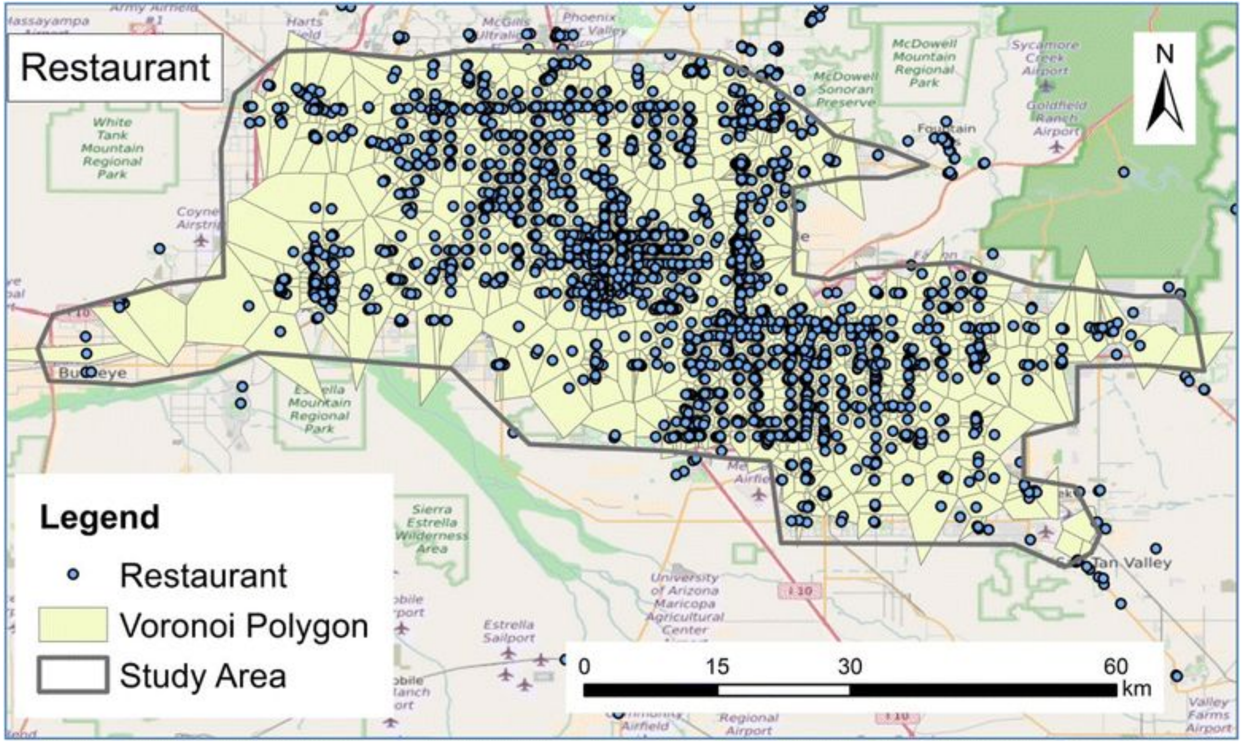


Cluster Analysis

- **Clustering**: grouping data into clusters such that the data in each cluster have similar attributes or properties
- Useful across a wide variety of fields and applications
  - Market analysis and segmentation
  - Medical imaging
  - Recommender systems
  - Geospatial data
  - Anomaly detection



A Comparison between Different Segmentation Techniques used in Medical Imaging by Mustafa and Hassan



Spatial analysis of users-generated ratings of yelp venues by Sun and Paule

Cluster Analysis

- Many algorithms exist to detect clusters
  - Parameters: vary for each algorithm
  - Input: set of data points $x_1, \ldots, x_n$ (any dimensionality, e.g., 2D, 3D, 100D, etc.)
  - Output: cluster assignment (each data point belongs to a cluster or other other group)
- **Unsupervised Machine Learning**: no “correct” labels to our data

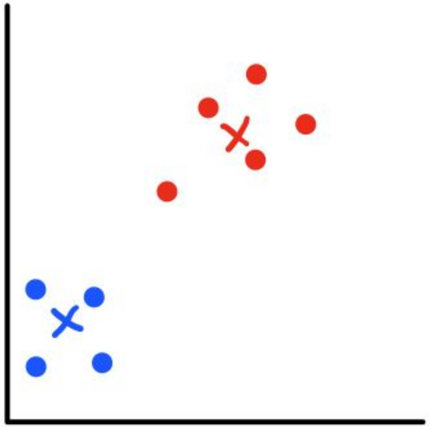
Cluster Analysis

1. Intro to Cluster Analysis
2. k-means Clustering
3. Density-based Spatial Clustering of Applications with Noise (DBSCAN)
4. Hierarchical Agglomerative Clustering (HAC)

k-means Clustering

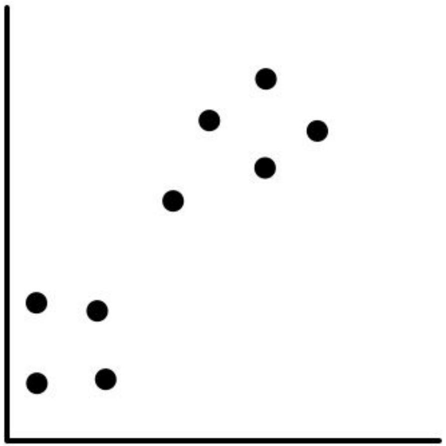
k-means Clustering

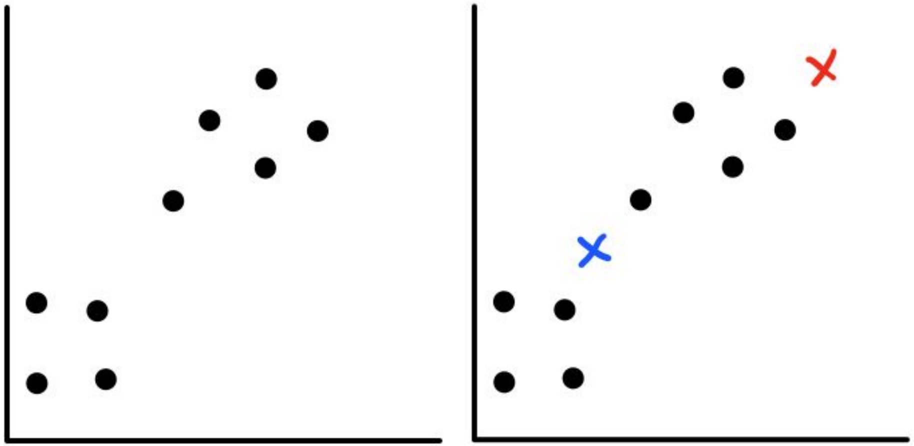
- Separate data into *k* disjoint clusters; minimize within-cluster sum-of-squares
- Only parameter is number of clusters *k*
- Very popular, well-known, and simple clustering algorithm
- **Cluster center/centroid**: a point that represents the cluster

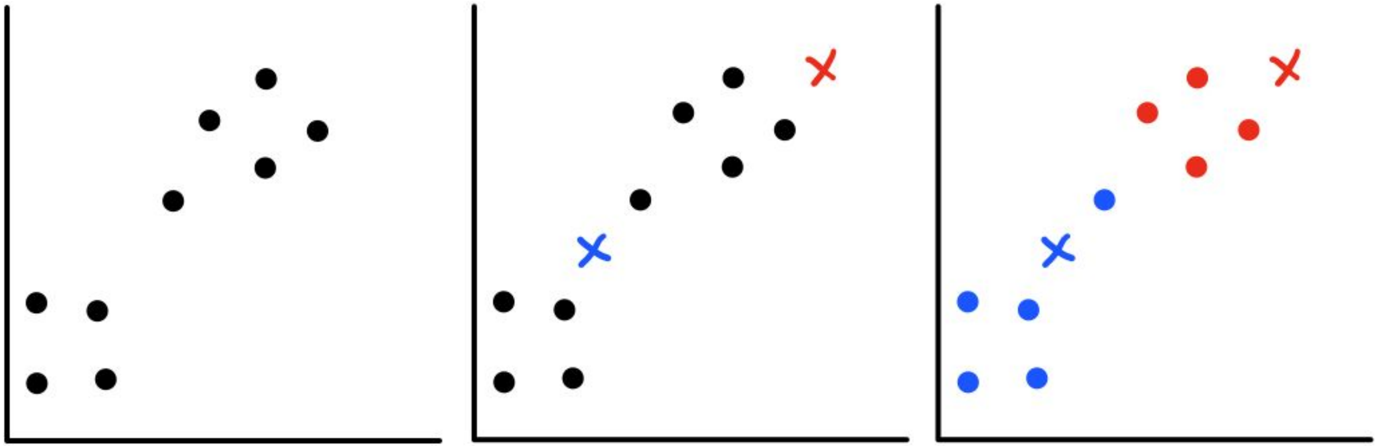


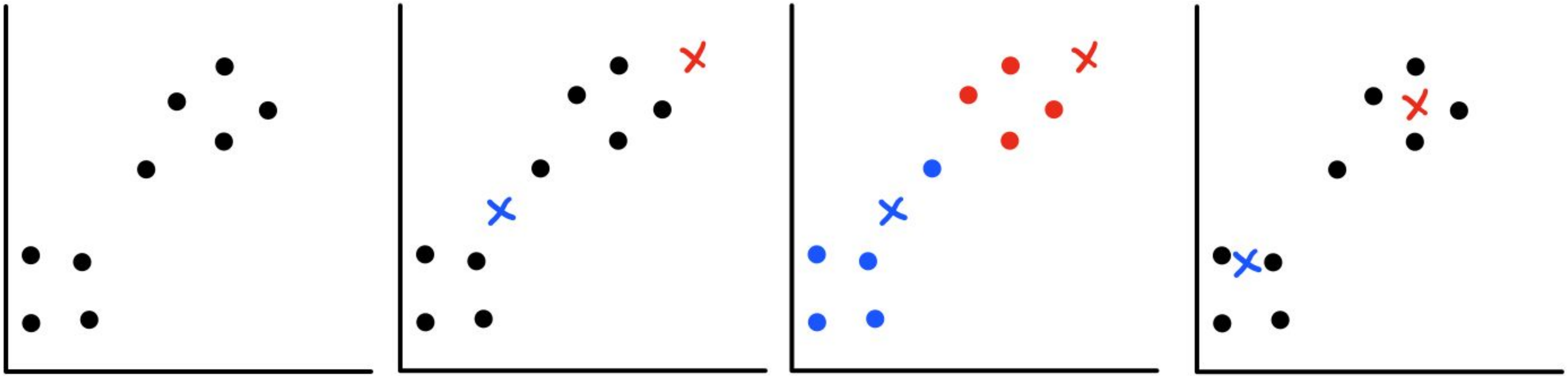
k-means Clustering Algorithm

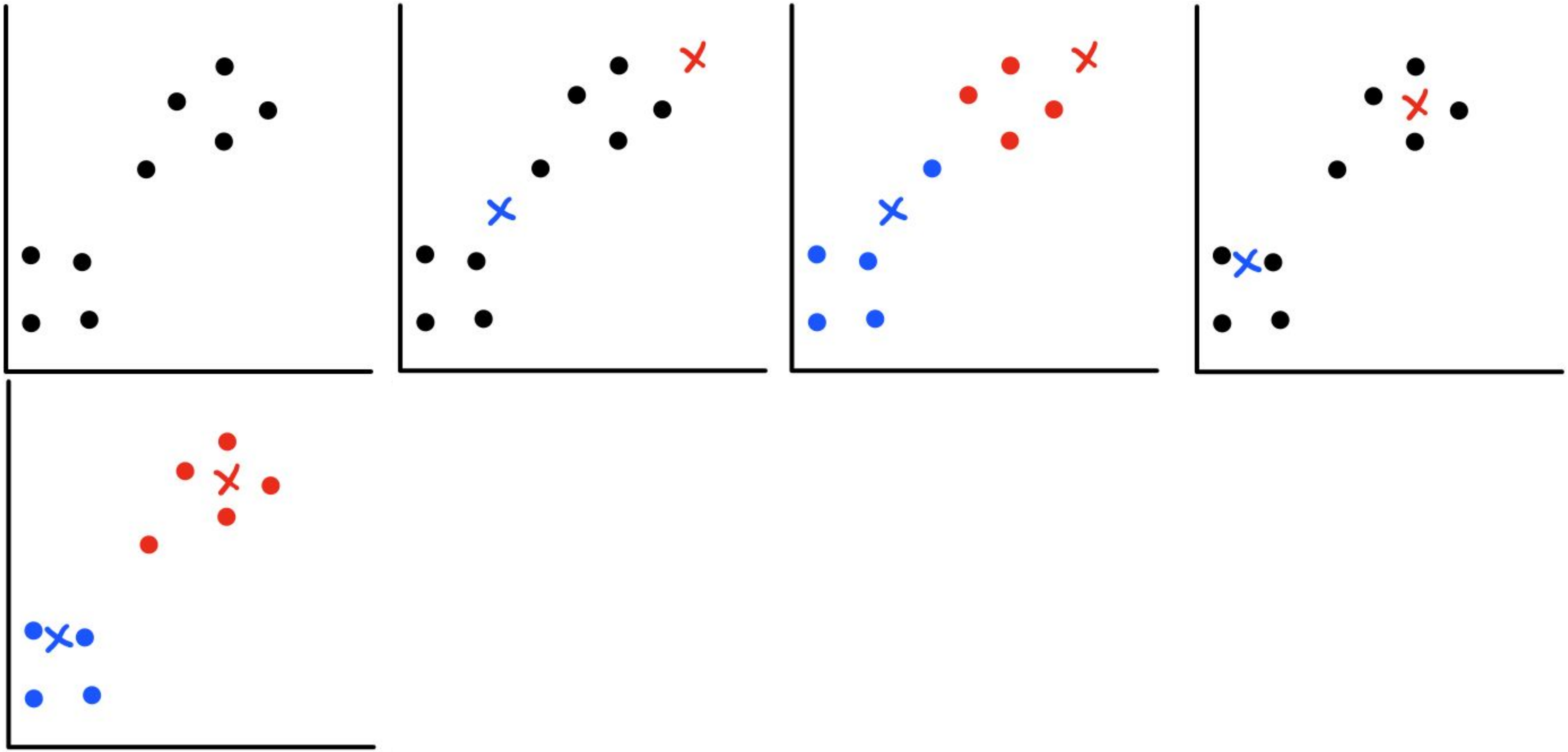
1. Randomly initialize cluster centers
2. For each point, assign it to its nearest cluster
3. Update the cluster centroids by taking the mean of the points assigned to it
4. Go back to 2 until convergence

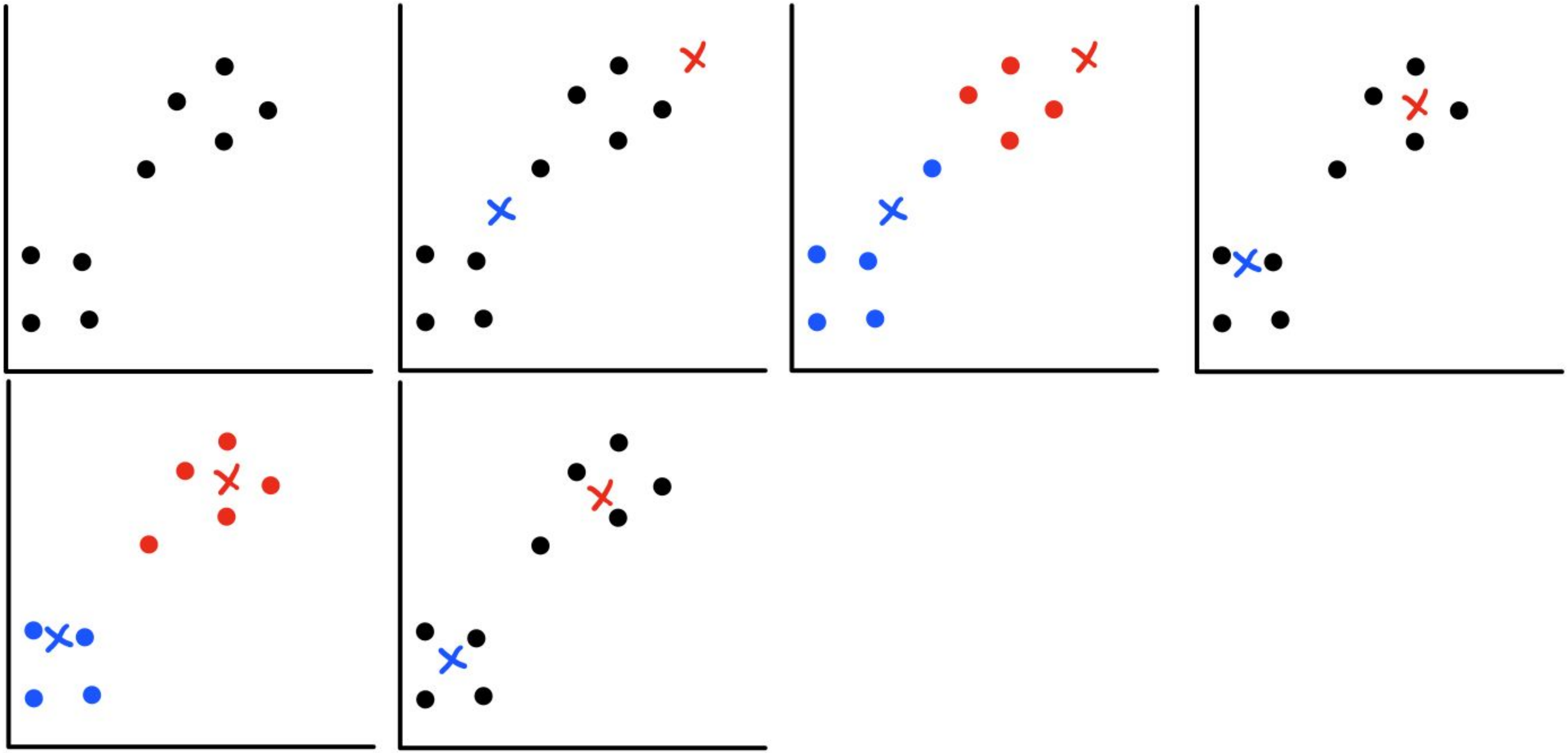


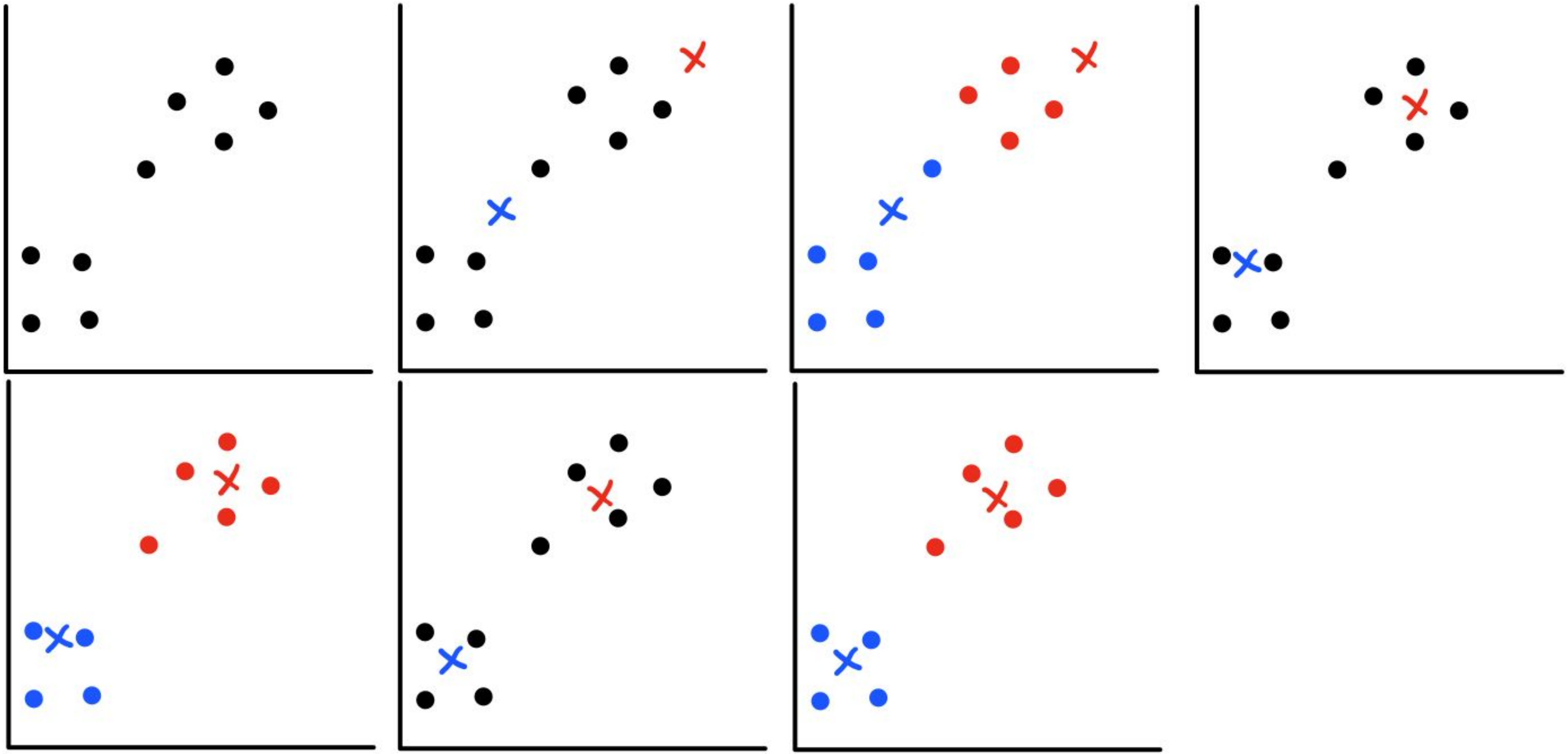












Convergence

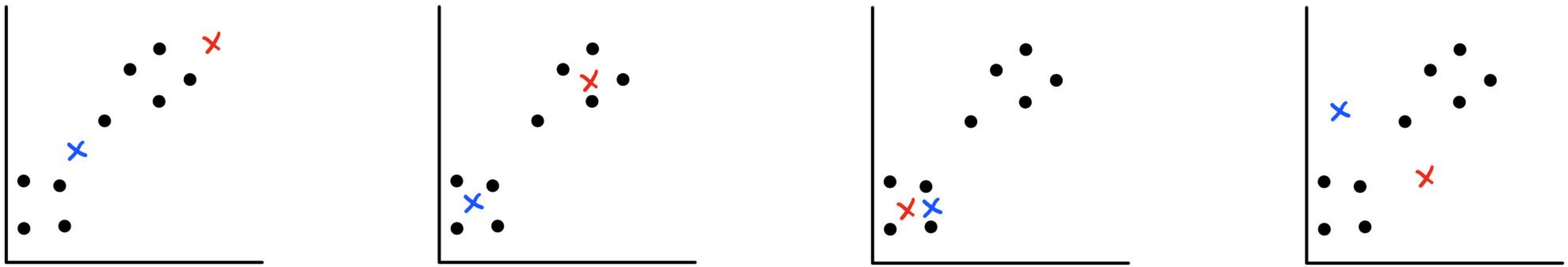
- **Convergence**: cluster centroids don't move or move a very small amount

$$\min \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i - \mu_j||^2$$

- Mathematically guaranteed to converge in a finite number of iterations
- However, may not converge to best clustering

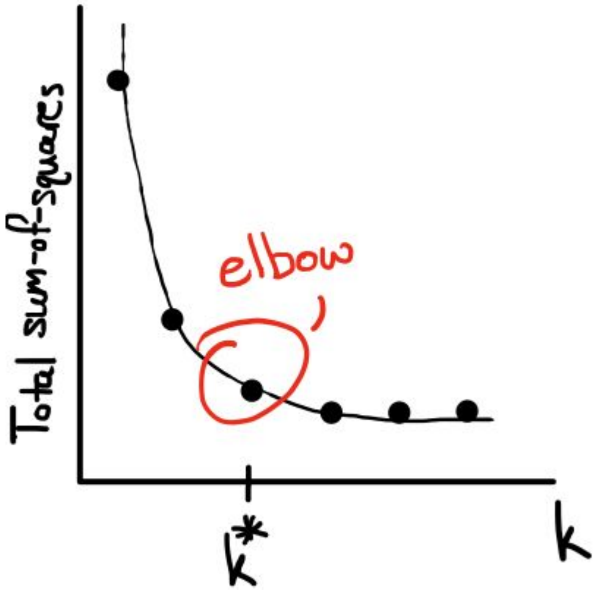
k-means Clustering

- Sensitive to cluster centroid initialization
  - Assign each cluster to a random data point
  - Choose the k points that are the farthest away from each other
  - Repeat k-means many times and pick the average of the clusters



k-means Clustering

- How many clusters to use?
  - Plot your data points and try various *k* values
  - Use the “elbow” method
- **Elbow method**
  - Run clustering algorithm for some *k*
  - For each cluster, compute within-cluster sum-of-squares between centroid and data
  - Sum up for all clusters
  - Repeat for different values of *k*



k-means Clustering

Advantages

- Widely known and used
- Simple algorithm; easy to implement
- Guaranteed convergence

Disadvantages

- Algorithmically slow
- Can converge to local minima
  - May not converge to optimal/best solution
- Not robust against varying cluster shapes
  - Same parameters used for each cluster

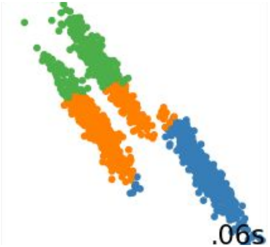
Cluster Analysis

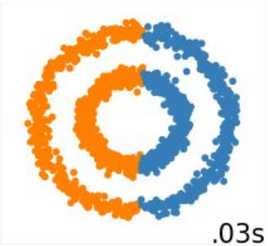
1. Intro to Cluster Analysis ✓
2. k-means Clustering ✓
3. Density-based Spatial Clustering of Applications with Noise (DBSCAN)
4. Hierarchical Agglomerative Clustering (HAC)

Density-based Spatial Clustering of Applications with Noise (DBSCAN)

DBSCAN

- k-means is unable to handle different cluster shapes



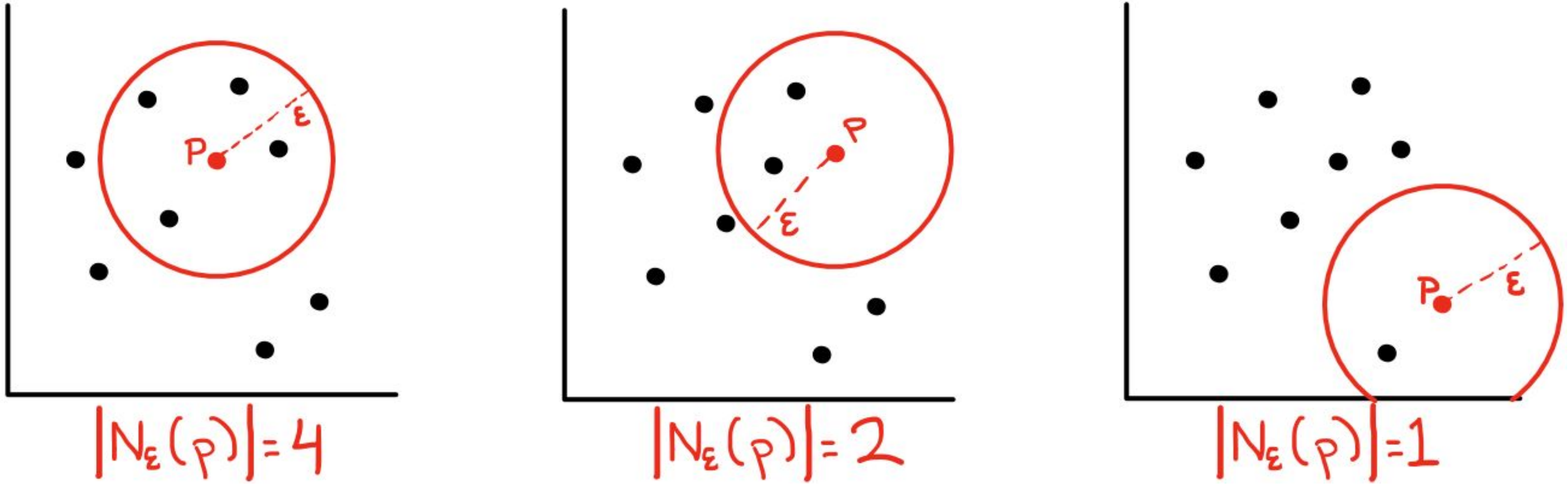


- DBSCAN is a *density-based* approach
  - Groups together points in high-density regions
  - Ignore outliers/noise in low-density regions

ε-neighborhoods

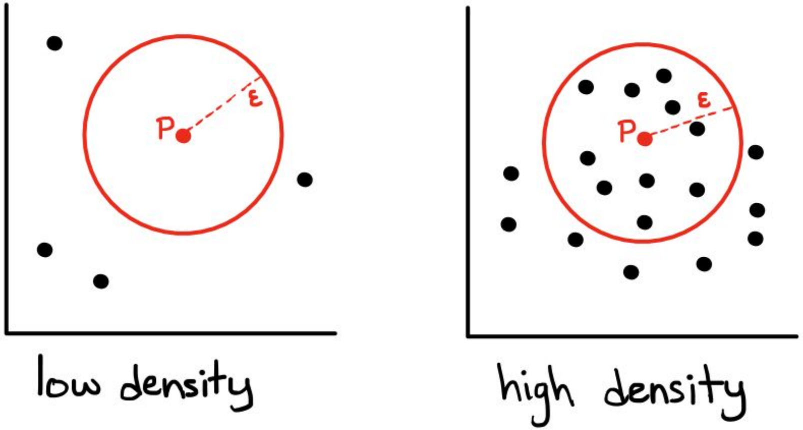
ε-neighborhood of p: set of all points at most ε away from p

$$N_\varepsilon(p) = \{q \mid q \neq p \text{ and } d(p,q) \leq \varepsilon\}$$

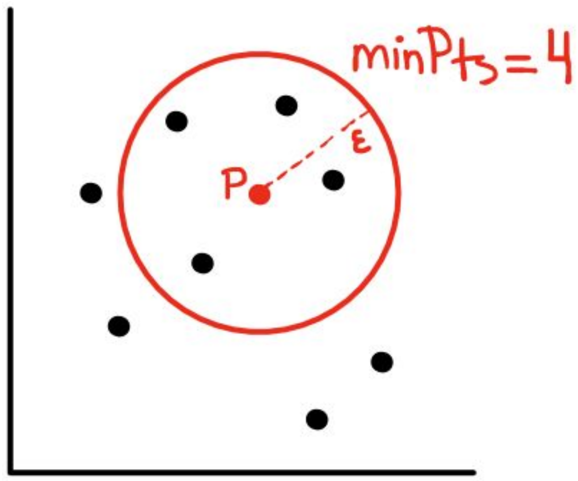


ε-neighborhoods

- Regions of high-density have more points in their ε-neighborhoods
- Define minPts as a parameter to denote high density
  - If there are at least minPts in the ε-neighborhood, then this is “high-density”
  - If there aren’t, then this ε-neighborhood is “low-density”



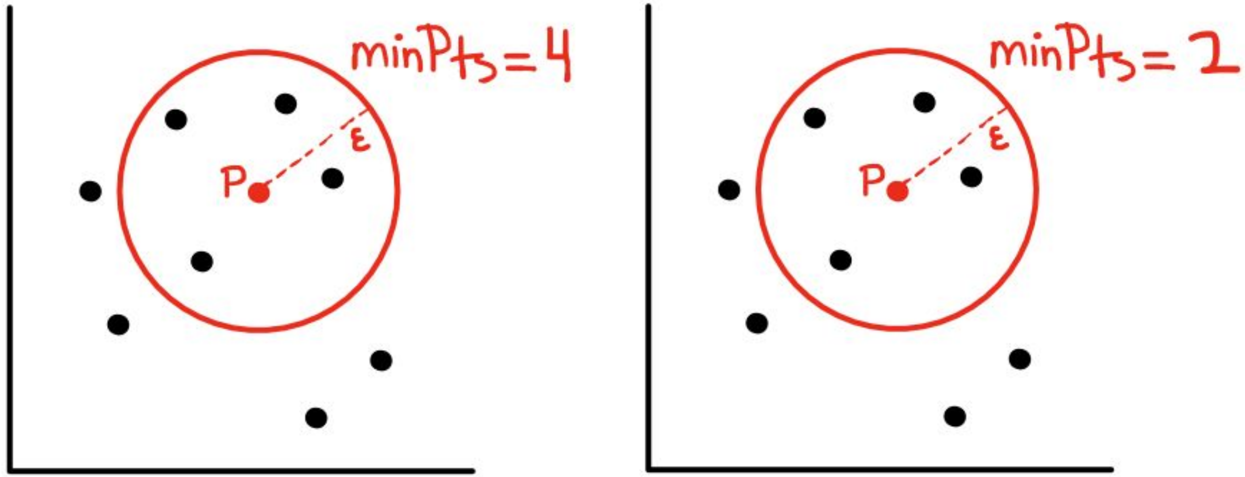
DBSCAN



- DBSCAN has 2 parameters
  - ε: size of the neighborhood
  - minPts: density requirement of the neighborhood
  - No parameter for the number of clusters! Inferred from the data
- Use these parameters to define clusters of high-density regions
- DBSCAN labels each point as a core point, border point, or outlier/noise point

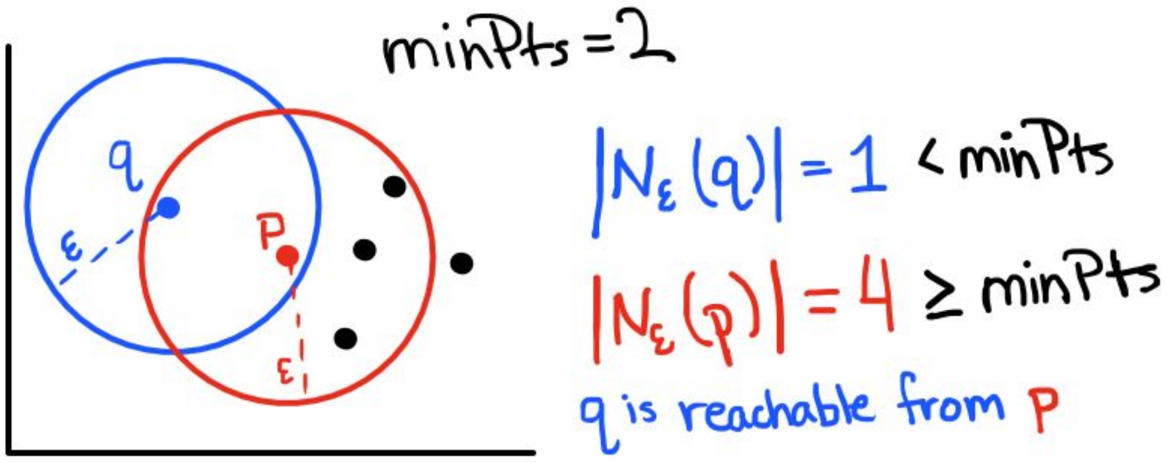
DBSCAN

- p is a **core point** if it has at least minPts points in its ε-neighborhood
- Core points are the foundation of the clusters
- Adjusting both ε and minPts affects the minimum density requirement



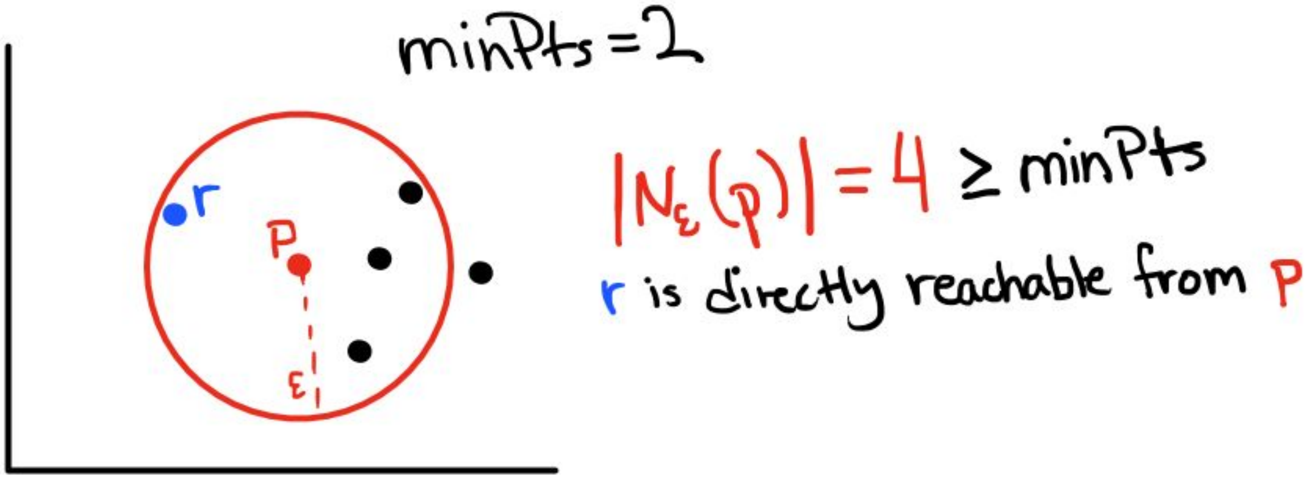
DBSCAN

- q is a **border point** if it is **reachable** from some core point p
- Border points define the *borders* around clusters



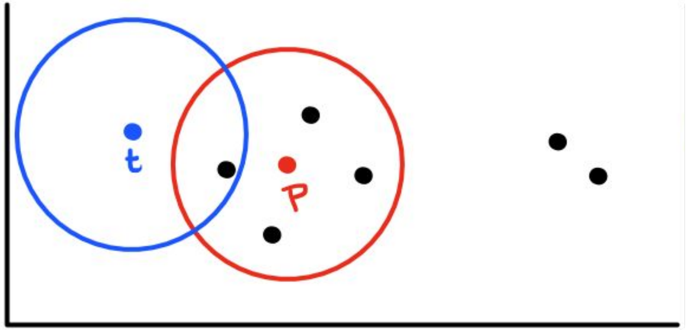
Reachability

r is **directly reachable/density-reachable** from p if r is in the ε-neighborhood of p and p is a core point.



Reachability

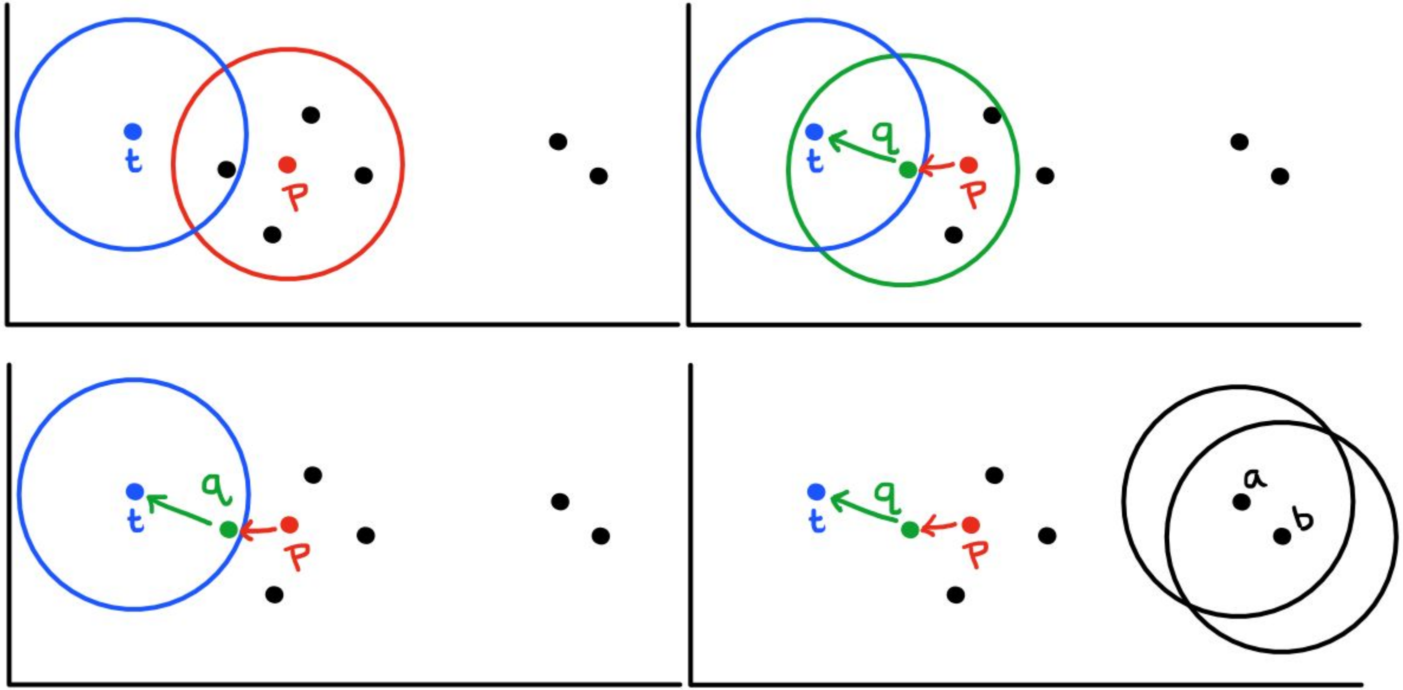
t is **reachable/density-reachable** from p if there exists some sequence of core points connecting p to t through their ε-neighborhoods.



minPts = 2

Reachability

Outliers or **noise points** are neither core or border points.



minPts = 2

q is directly reachable from p

t is directly reachable from q

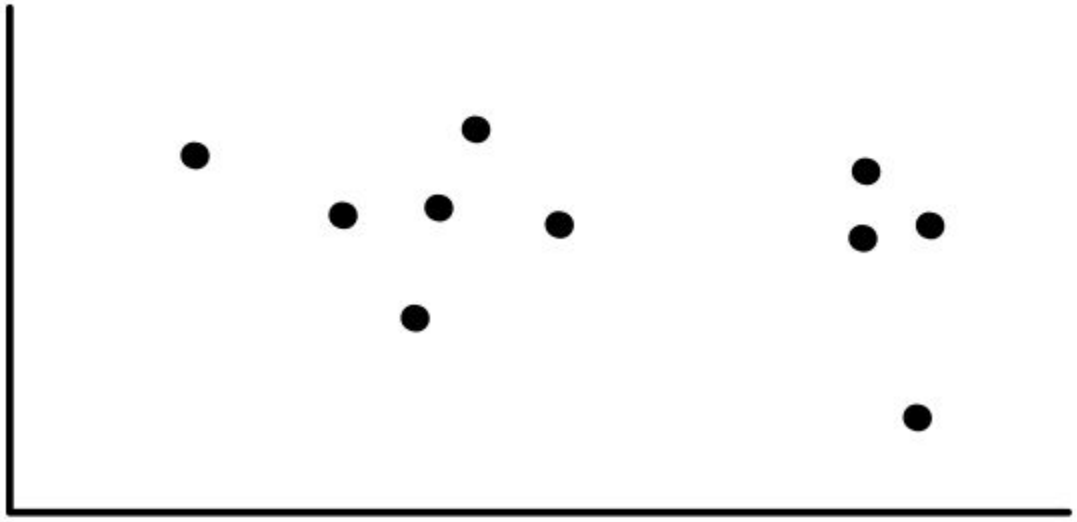
t is reachable from p

a,b are outliers

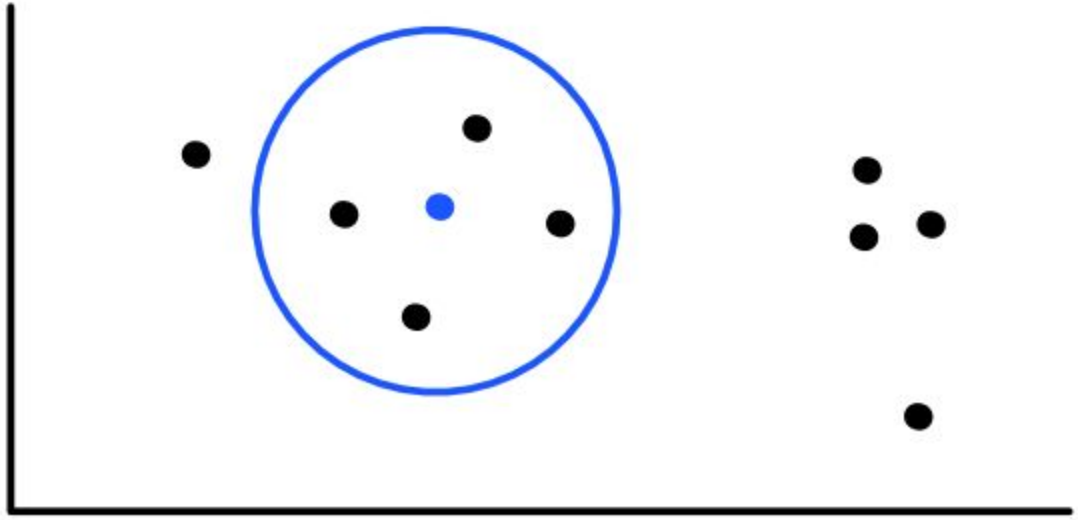
DBSCAN Algorithm

1. Pick a point p that hasn’t been selected or labeled yet
2. Check the number of points in p’s ε-neighborhood
    a. If it is less than minPts, mark p as an outlier for now and go back to 1
    b. If it is at least minPts, mark p as a core point and start a new cluster at p
3. Find all **reachable** points from p
    a. Mark some point q as core point if q has at least minPts in its ε-neighborhood
    b. Mark some point q as border point if q does not have at least minPts in their ε-neighborhood but is reachable from p
4. Go back to 1 and repeat until each point is labeled

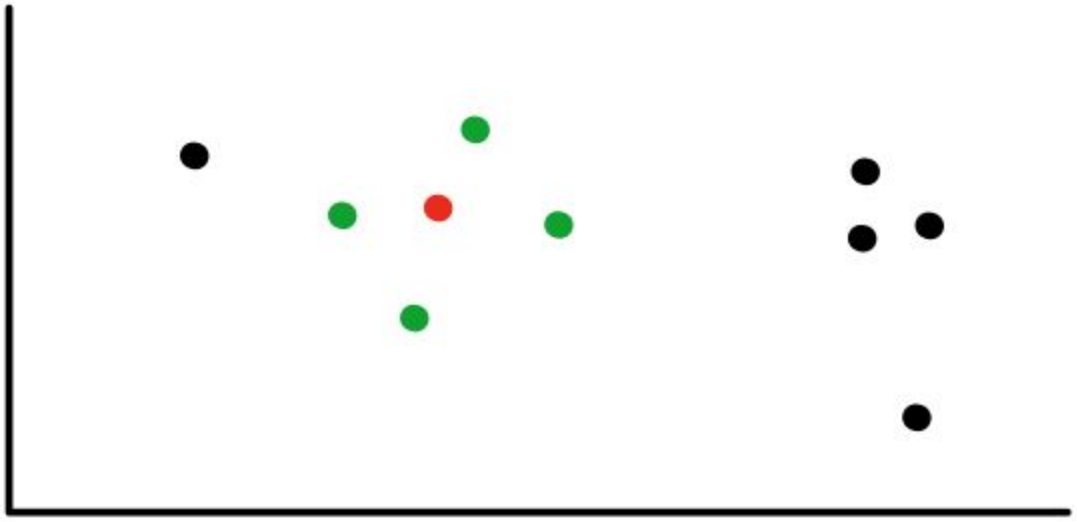
DBSCAN Algorithm



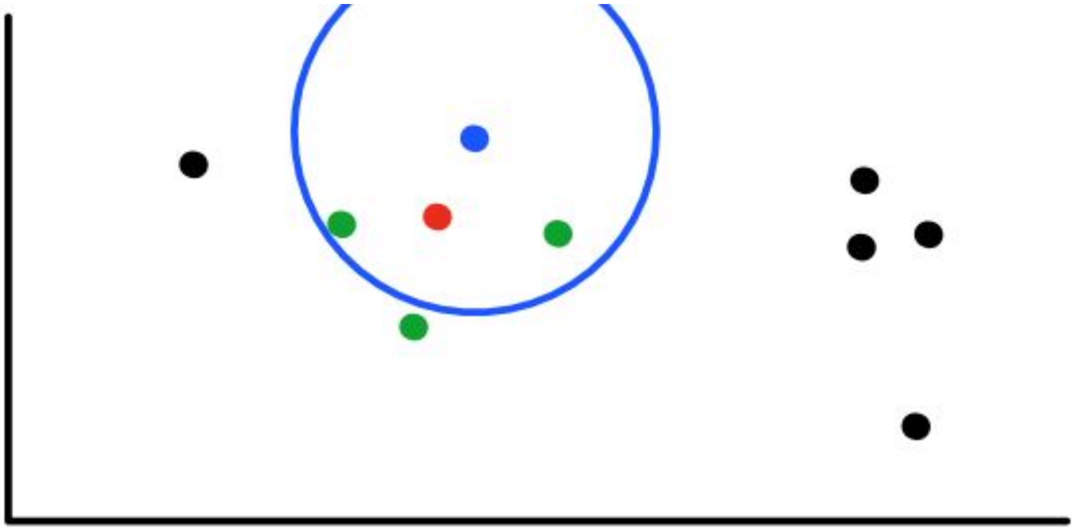
DBSCAN Algorithm



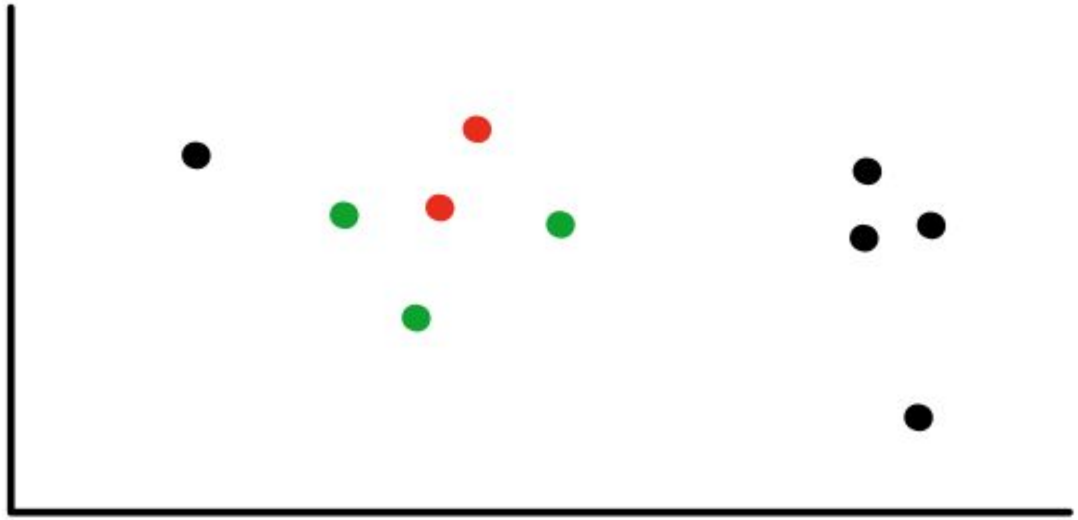
DBSCAN Algorithm



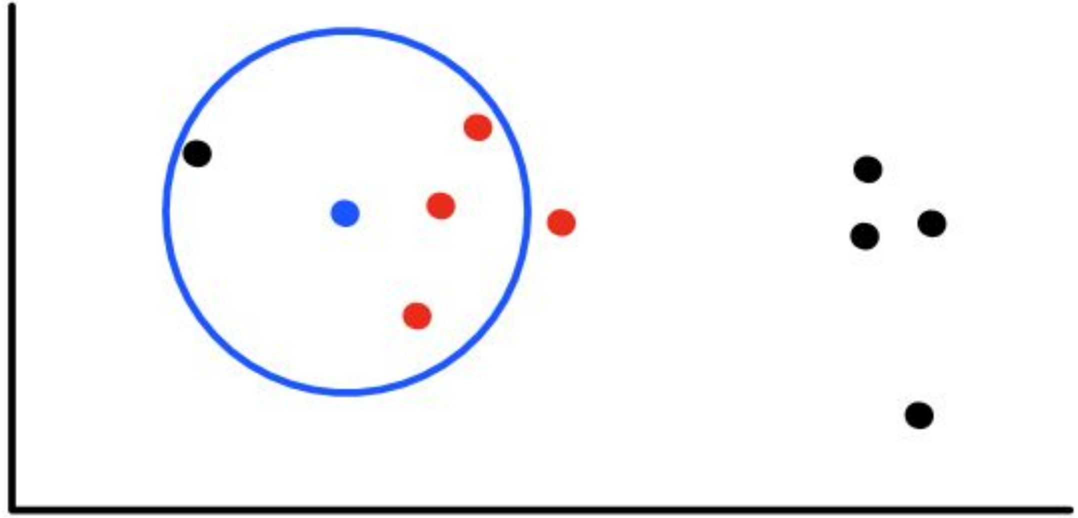
DBSCAN Algorithm



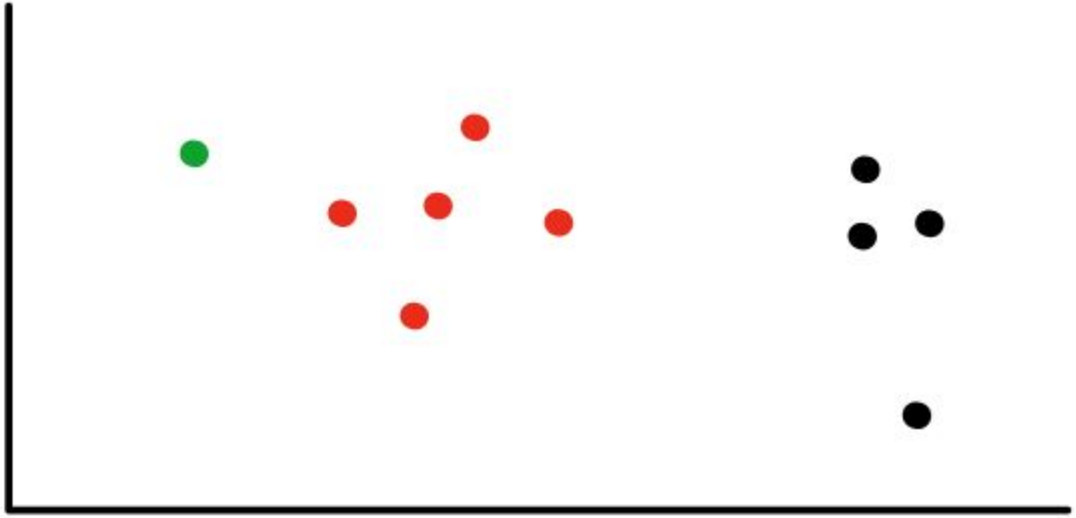
DBSCAN Algorithm



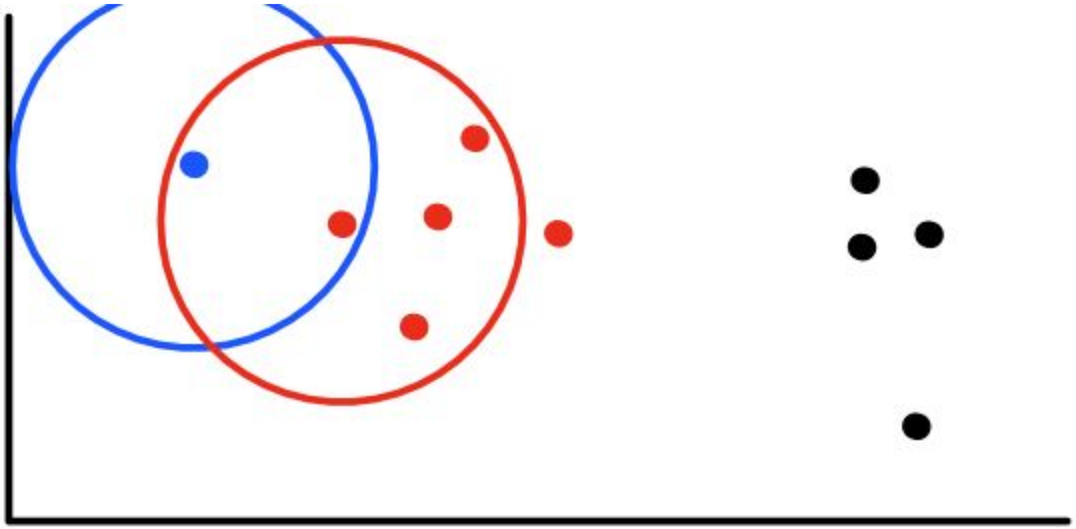
DBSCAN Algorithm



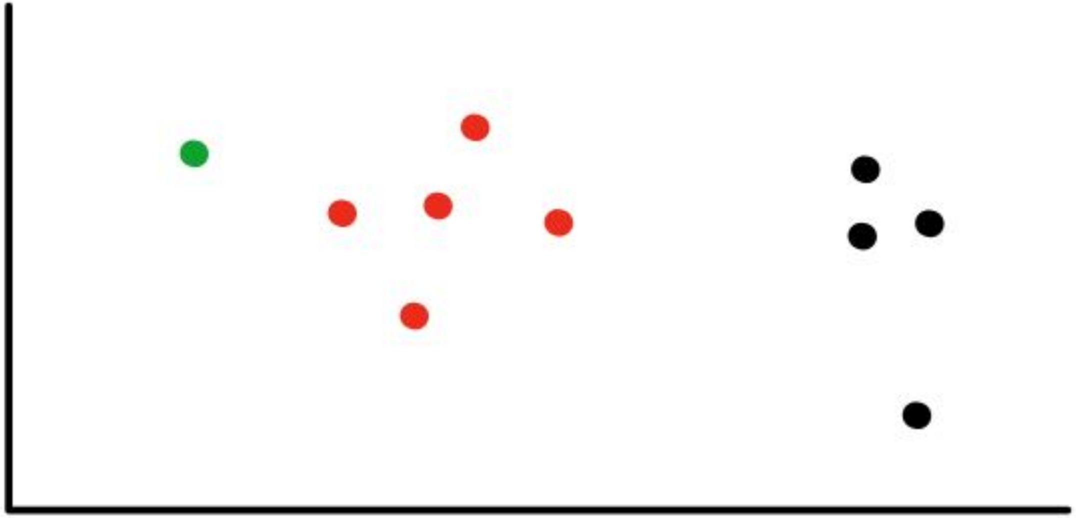
DBSCAN Algorithm



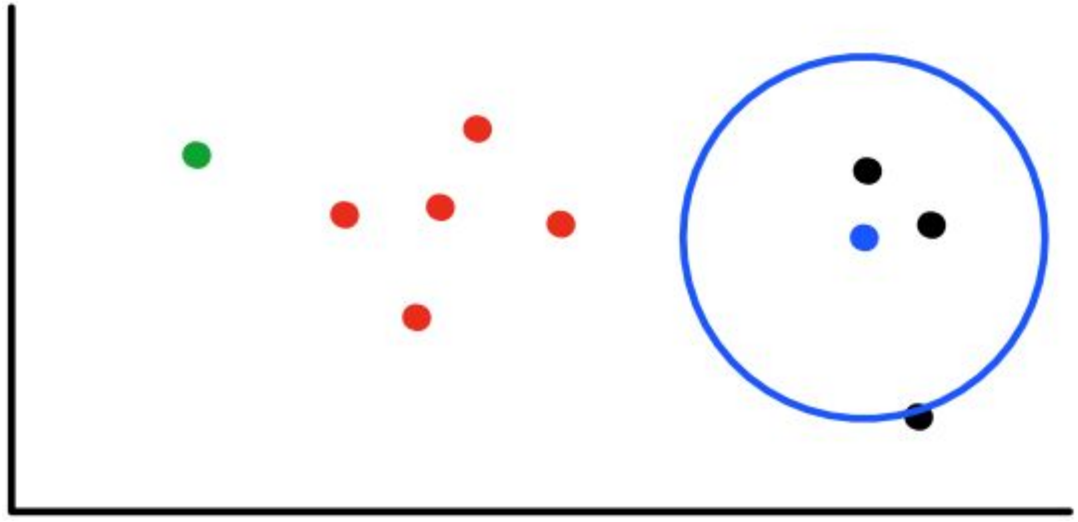
DBSCAN Algorithm



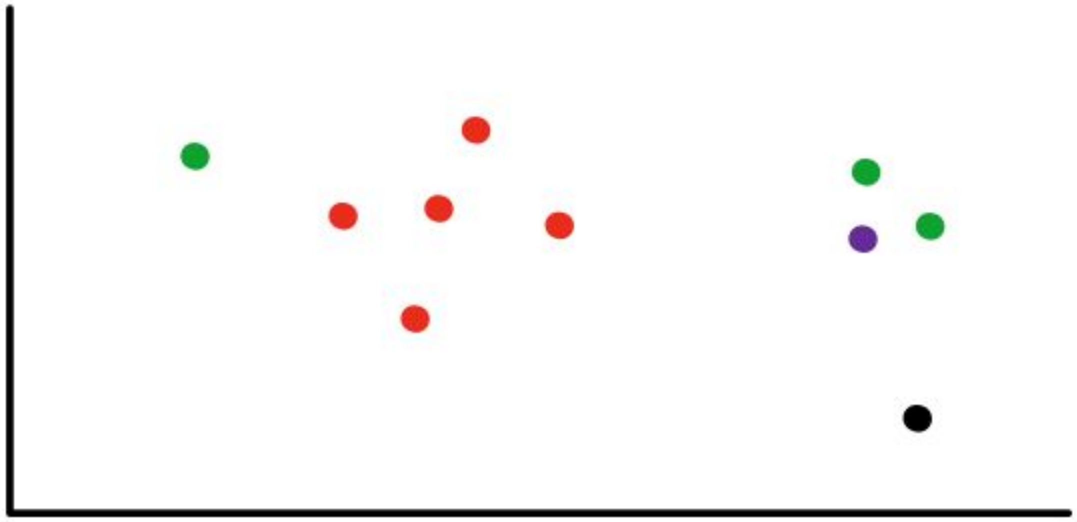
DBSCAN Algorithm



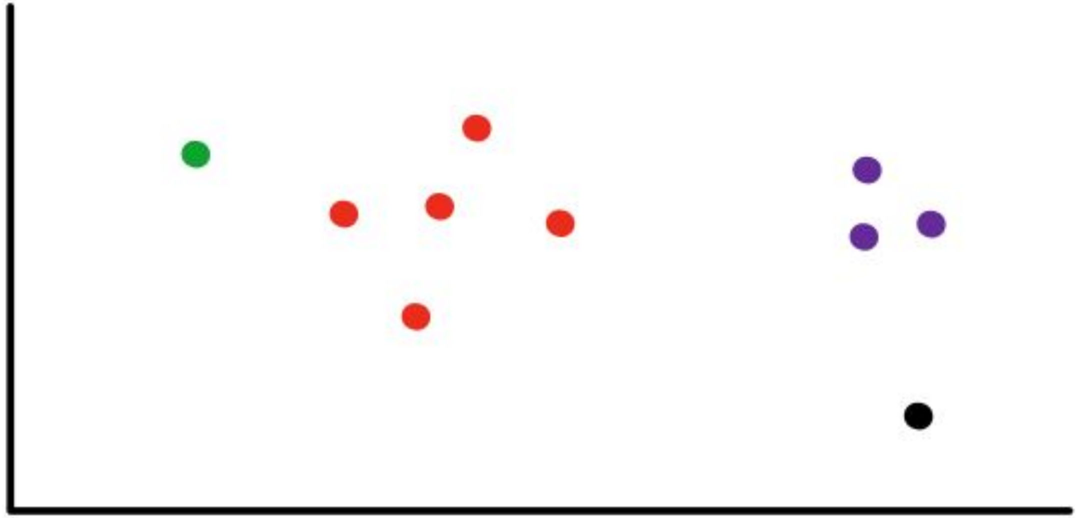
DBSCAN Algorithm



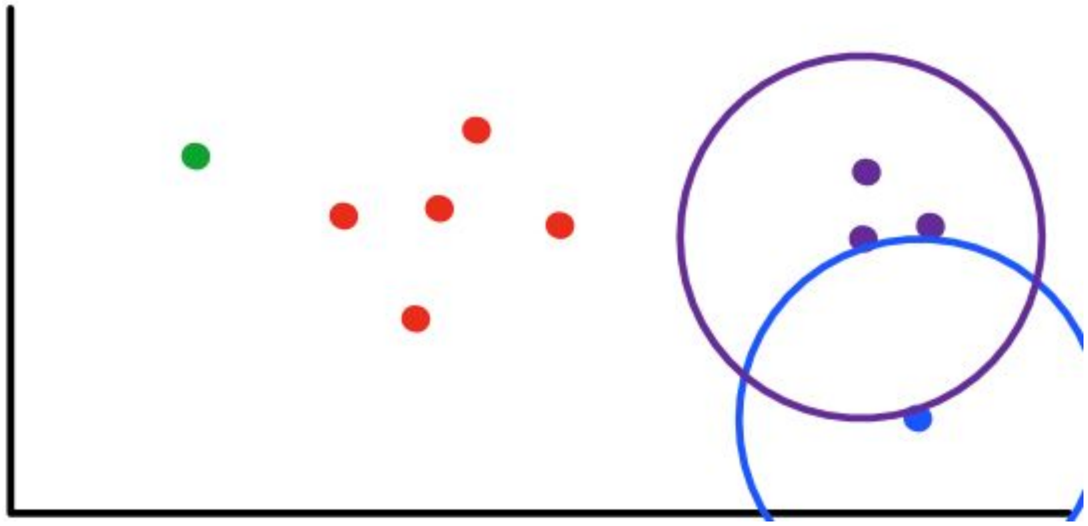
DBSCAN Algorithm



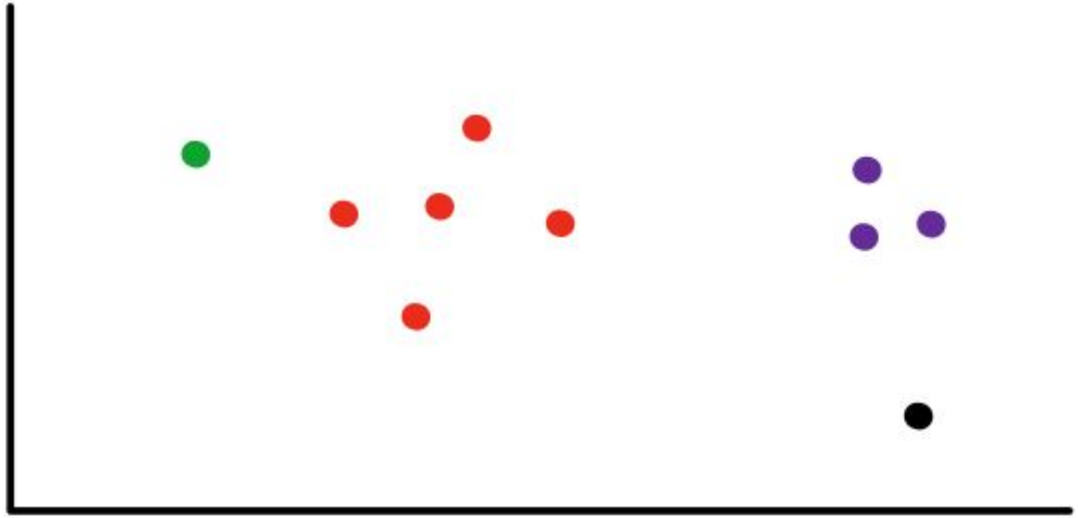
DBSCAN Algorithm



DBSCAN Algorithm



DBSCAN Algorithm



DBSCAN

Advantages

- Robust to noise and outliers
- Number of clusters inferred from the data
- Correctly groups arbitrary cluster shapes

Disadvantages

- Very sensitive to parameters
- Unable to handle varying densities
  - Same density parameters for all points
  - E.g., Two clusters with vastly different densities abide by the same min density
- Quality dependent on the distance metric
  - Usually use Euclidean distance
  - Worse with higher-dimensional data

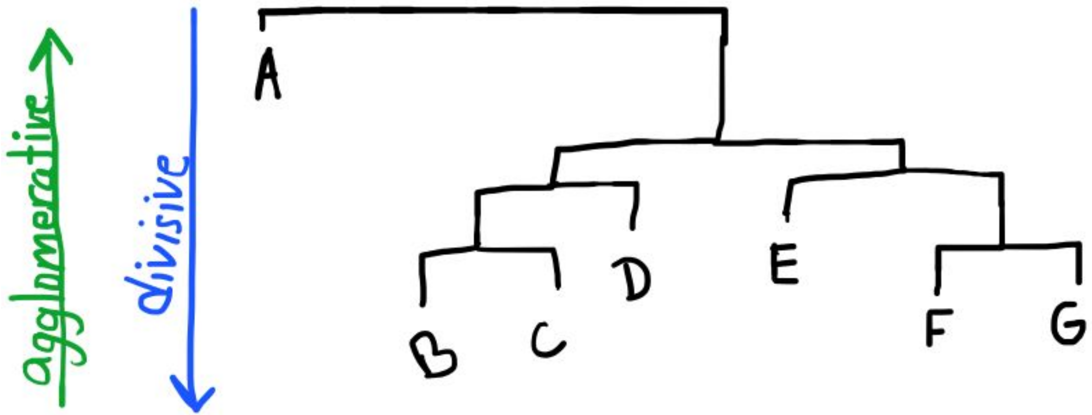
Cluster Analysis

1. Intro to Cluster Analysis ✓
2. k-means Clustering ✓
3. Density-based Spatial Clustering of Applications with Noise (DBSCAN) ✓
4. Hierarchical Agglomerative Clustering (HAC)

Hierarchical Agglomerative Clustering (HAC)

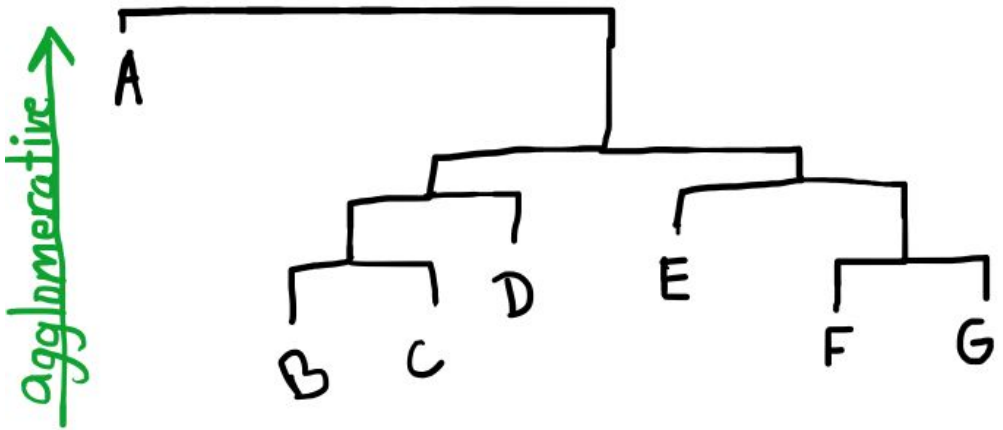
Hierarchical Agglomerative Clustering (HAC)

- **Hierarchical clustering**: build a tree structure/hierarchy of the clusters
  - **Agglomerative**: each point is its own cluster initially and we group them recursively
  - **Divisive**: all points are one cluster and we split them recursively
- Tree structure is a nice human-interpretable visualization
- Each split in the tree is a segmentation of the data



Hierarchical Agglomerative Clustering (HAC)

- **Hierarchical Agglomerative Clustering**: each point is its own cluster initially
- Use a similarity metric to merge clusters together
- Construct a tree/dendogram of the clusters
- Only parameter is the similarity metric

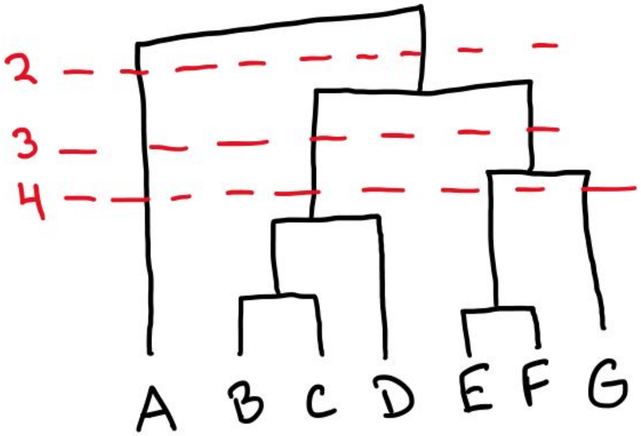


Hierarchical Agglomerative Clustering (HAC)

1. Assign each point to its own cluster
2. Find the two “closest” clusters using the similarity metric and merge them
3. Go to 2 until all clusters are merged into one cluster

Hierarchical Agglomerative Clustering (HAC)

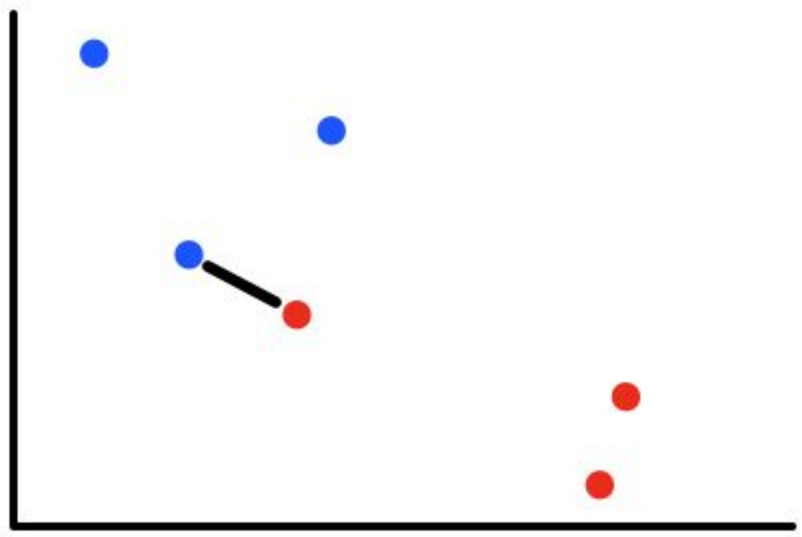
- May not need pre-defined number of clusters
- For flat clustering, we need number of clusters
- Given number of clusters, points may be assigned based on several metrics
  - Minimize clusters variance
  - Splits in dendogram



Linkage Metrics

Single linkage: distance between the closest pair

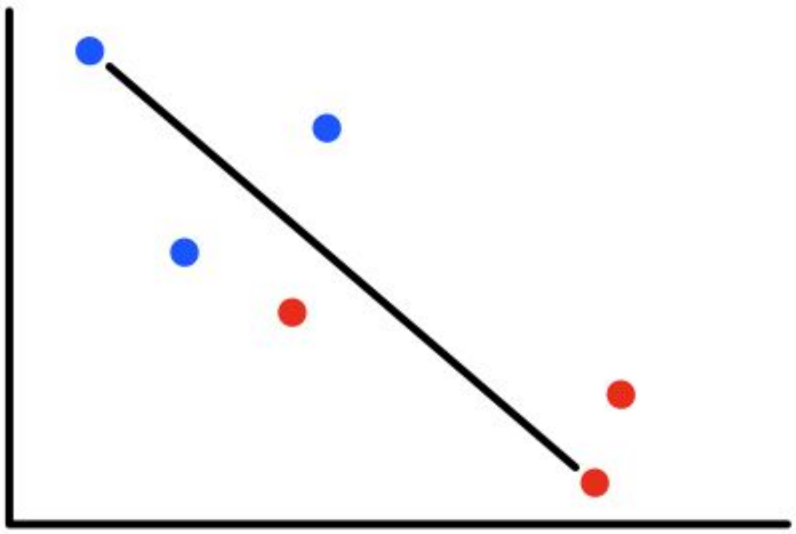
$$d_{SL}(X, Y) = \min_{i,j} d(X_i, Y_j)$$



Linkage Metrics

Complete linkage: distance between the farthest pair

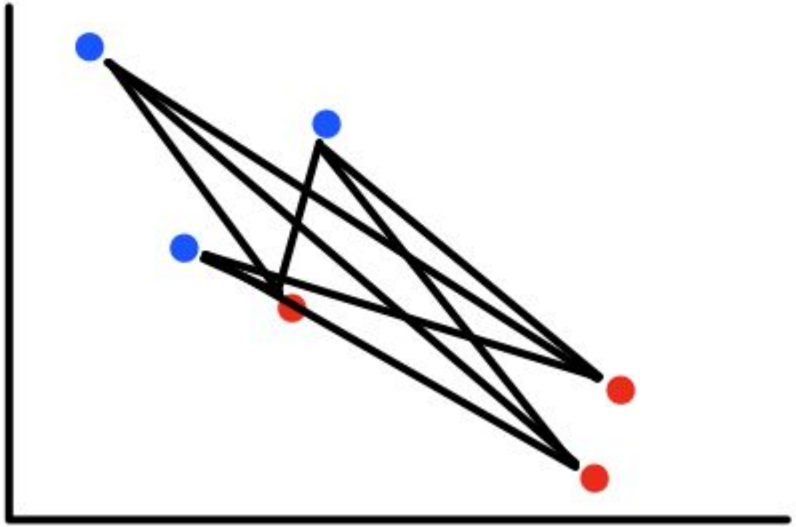
$$d_{CL}(X, Y) = \max_{i,j} d(X_i, Y_j)$$



Linkage Metrics

Average linkage: averaged distance between all pairs

$$d_{AL}(X,Y) = \frac{1}{|X||Y|}\sum_i \sum_j d(X_i, Y_j)$$



Linkage Metrics

- **Single linkage**: may produce chaining, i.e., sequence of close/similar clusters grouped early
- **Complete linkage**: may not merge together close groups because of outliers
- **Average linkage**: compromise between single and complete
  - Depends on the closeness/similarities being on the same scale

Hierarchical Agglomerative Clustering

Advantages

- Simple algorithm; easy to implement
- Constructs a human-interpretable structure for cluster groupings

Disadvantages

- Susceptible to noise or outliers
- Cluster groupings early drastically affect final grouping
- Forces hierarchical structure on data that might not be hierarchical

Cluster Analysis

1. Intro to Cluster Analysis ✓
2. k-means Clustering ✓
3. Density-based Spatial Clustering of Applications with Noise (DBSCAN) ✓
4. Hierarchical Agglomerative Clustering (HAC) ✓