Lecture 12: 20 February, 2024

Madhavan Mukund

`https://www.cmi.ac.in/~madhavan`

Data Mining and Machine Learning
January–May 2022

Unsupervised learning

- Supervised learning requires labelled data

- Vast majority of data is unlabelled

- What insights can you get into unlabelled data?

"If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake ..."

- Yann LeCun
ACM Turing Award 2018
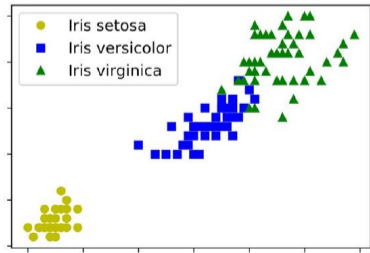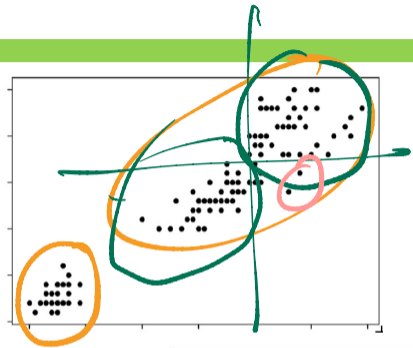
Applications

- Customer segmentation
  - Marketing campaigns
- Anomaly detection
  - Outliers
- Semi-supervised learning
  - Propagate limited labels
- Image segmentation
  - Object detection
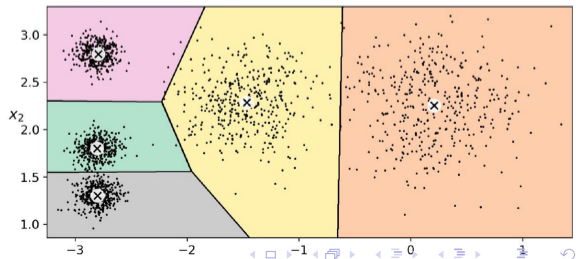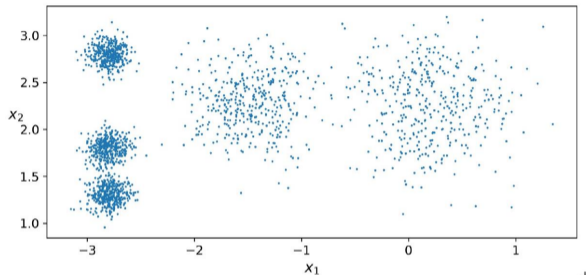

After a couple of classes

Clustering

- Find natural groups of data
- Define a distance measure
- Group together data that is close together
- Top down
  - Partition data into clusters

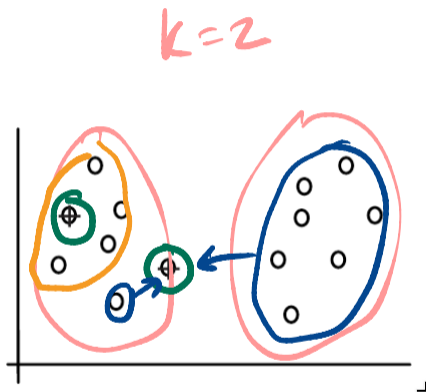- Bottom up
  - Group items into clusters
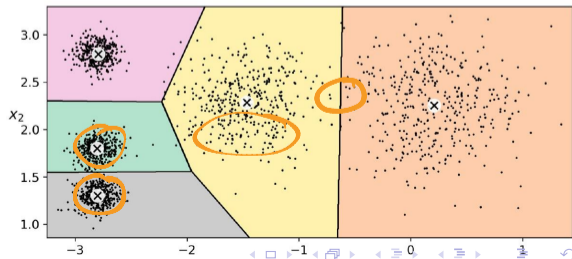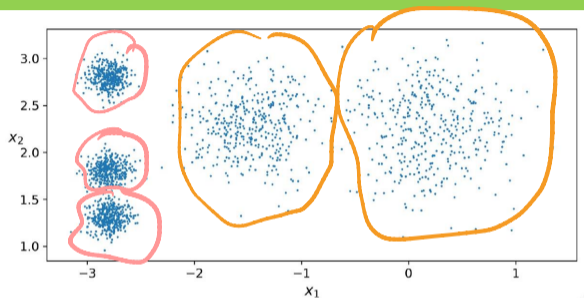
Top down clustering

K Means Clustering

- Data items are points in n dimensions

  - $(x_1, x_2, \ldots, x_n)$

- Partition into K clusters

  - Fix K in advance

- Each cluster is represented by its geometric centre

  - *Centroid*, or *mean*

  - Hence "K means"

K Means Algorithm

- Choose K points initially as random centroids

- In each iteration

  - Assign each point to nearest centroid

  - Recompute centroids

- Termination

  - Clusters stabilize

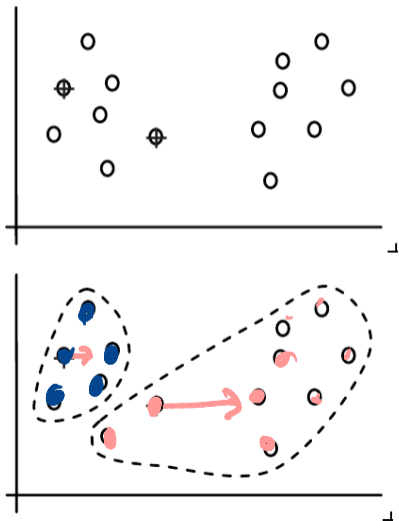 - Sum square distance is below threshold

K Means Algorithm

- Choose K points initially as random centroids
- In each iteration
  - Assign each point to nearest centroid
  - Recompute centroids
- Termination
  - Clusters stabilize
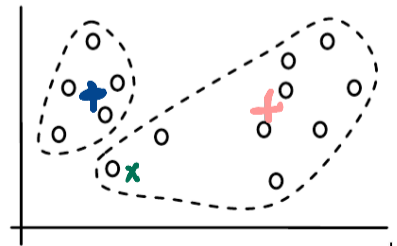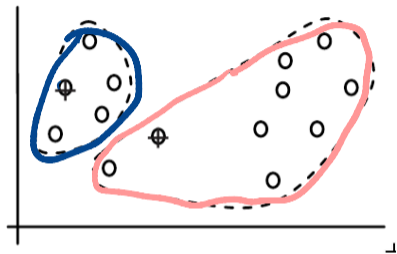 - Sum square distance is below threshold

K Means Algorithm

- Choose K points initially as random centroids
- In each iteration
    - Assign each point to nearest centroid
    - Recompute centroids
- Termination
    - Clusters stabilize
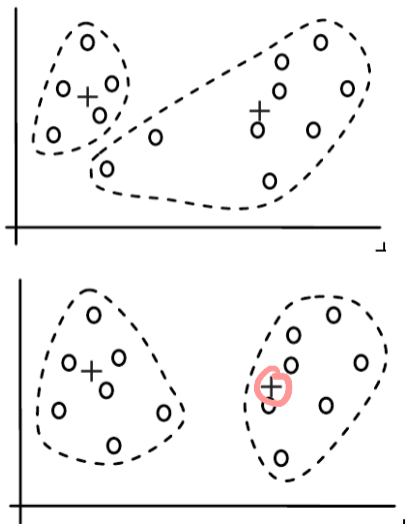   - Sum square distance is below threshold

K Means Algorithm

- Choose K points initially as random centroids
- In each iteration
  - Assign each point to nearest centroid
  - Recompute centroids
- Termination
  - Clusters stabilize
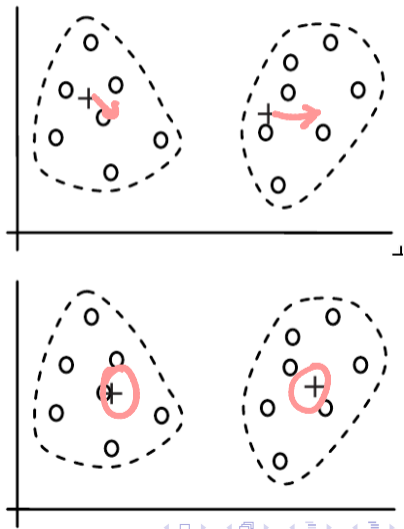 - Sum square distance is below threshold

K Means Algorithm

- Choose K points initially as random centroids
- In each iteration
  - Assign each point to nearest centroid
  - Recompute centroids
- Termination
  - Clusters stabilize
  - Sum square distance is below threshold

- Choose K points initially as random centroids
- In each iteration
    - Assign each point to nearest centroid
    - Recompute centroids
- Termination
    - Clusters stabilize, or
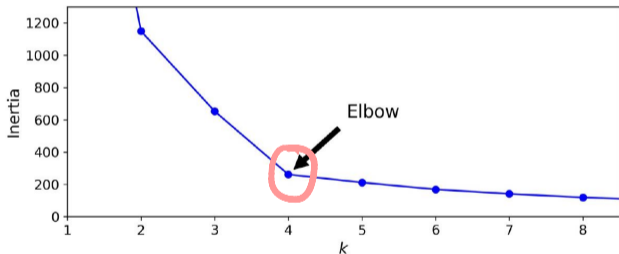   - Sum square distance is below threshold
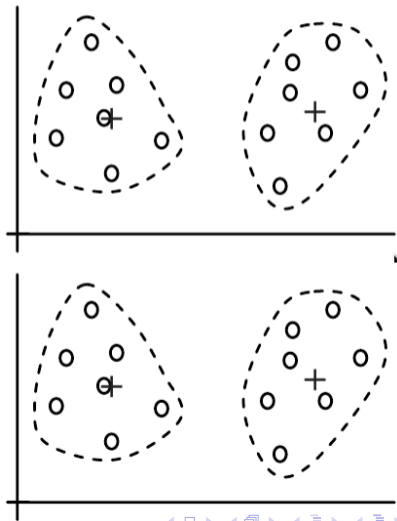
Evaluating clustering

- How "tight" are the clusters?

- Mean squared distance from centroids – *inertia*

$$\frac{1}{n}\sum_{j=1}^{K}\sum_{x \in C_j} dist(x, centroid_j)^2$$

- Plot inertia for different values of K and look for optimum

- Can also use change in inertia threshold to stop iterations

K Means Algorithm

Advantages

- Efficient – each iteration makes a single pass over data

  - Incrementally compute centroid

Disadvantages

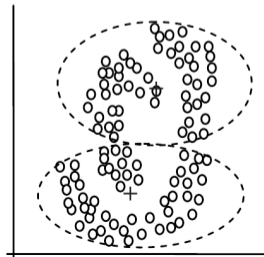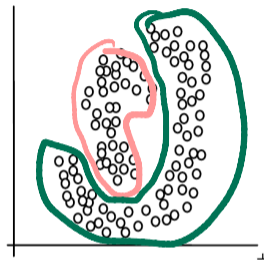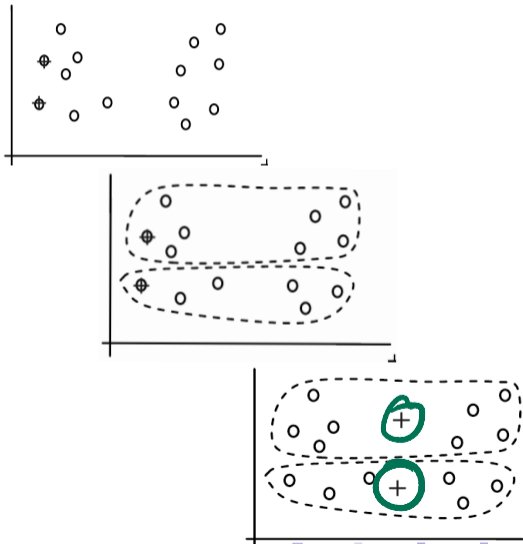- Can only find clusters that look like ellipses

K Means Algorithm

Advantages

- Efficient – each iteration makes a single pass over data
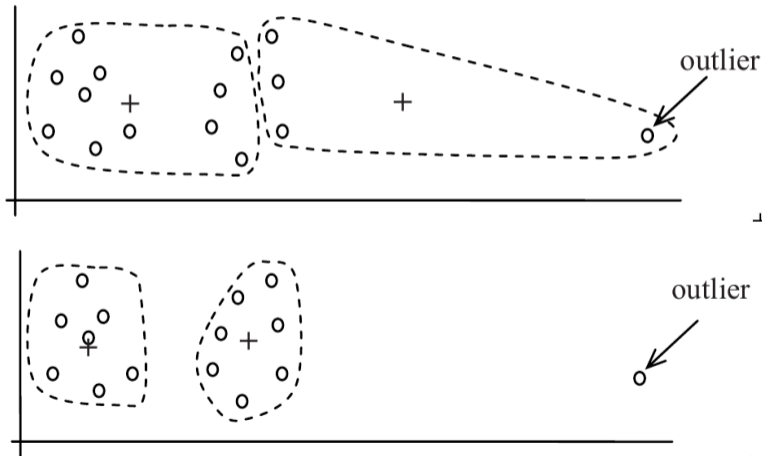
  - Incrementally compute centroid

Disadvantages

- Can only find clusters that look like ellipses

- Choice of initial random centroid matters
  - Repeat and check

- Anomalous values
  - Far away from all centroids
- But clustering with outliers distorts clusters
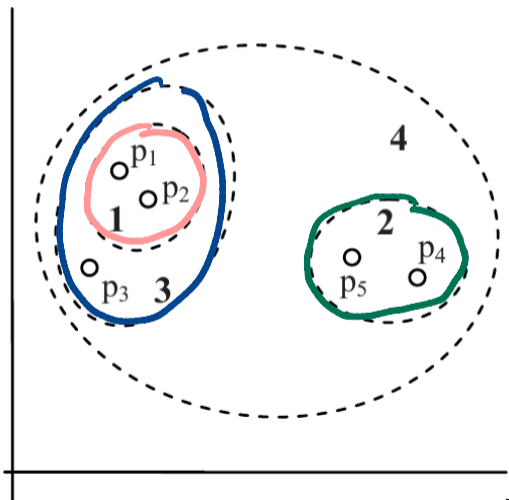- How to identify outliers before clustering?

c_m_i

Clustering

- K Means clustering can only find clusters that look like ellipses
- Instead, build clusters bottom up, by merging clusters

Hierarchical clustering

- Initially, each item is a singleton cluster
- At each step, merge nearest clusters

Hierarchical Clustering

- Initially, each item is a singleton cluster
- At each step, merge nearest clusters
- Can represent process using a tree – dendrogram
- Choose appropriate level in dendrogram for final clustering
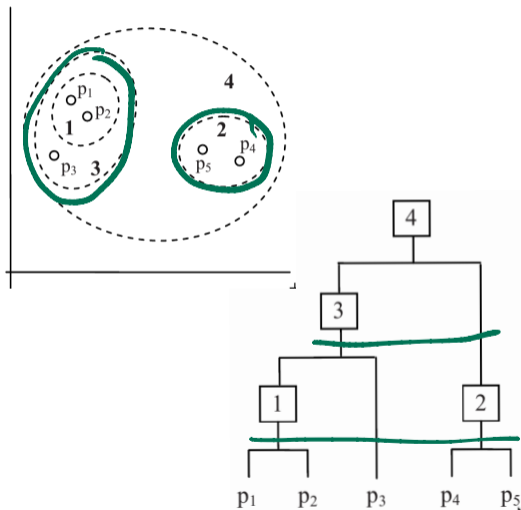
Hierarchical Clustering

To merge clusters, define distance between clusters

- Single link: distance between closest points

  - Creates chain effect

- Complete link: maximum of pairwise distances

- Average link: mean of pairwise distances

- All require $O(n^2)$ computation - expensive
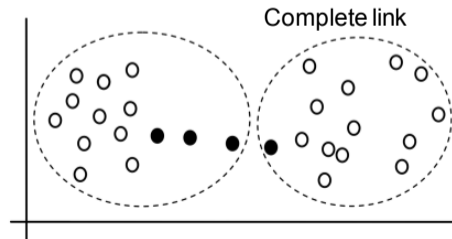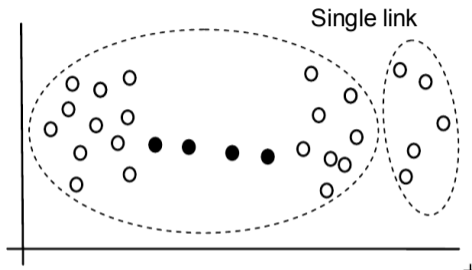
- How to identify odd shaped clusters?
- Cluster - group of points that are "close together"
- Identify "dense" neighbourhoods
- How do we formalize this?
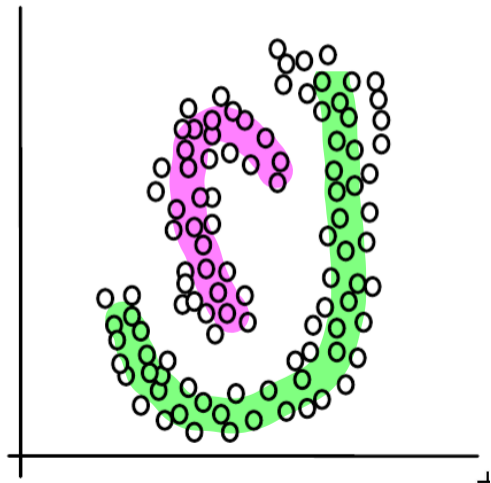
Density based clustering

- Construct a small ball around each point, radius *Eps*
- Identify a threshold for neighbours within ball, *MinPts*
- Core point – has at least *MinPts* neighbours inside *Eps* ball
- Connect each core point to all its neighbours
- Border points – attached to core points but not core themselves
- Noise – disconnected points
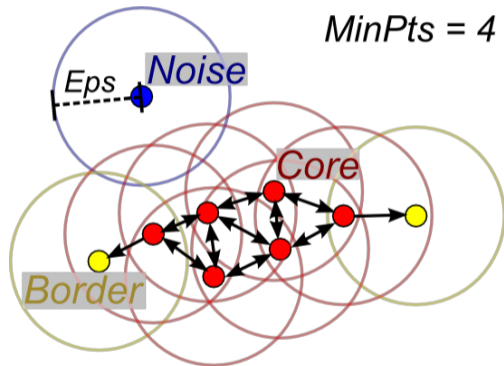
Density based clustering

- Formally, edges from core points to neighbours define a directed graph
- Border points are part of this graph, but cannot add edges to extend the graph
- Discard the edge directions
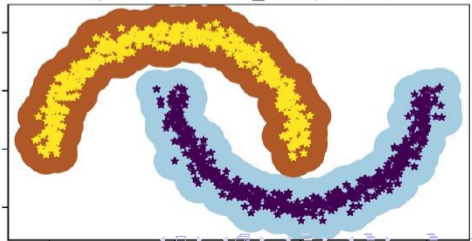- Connected components are clusters
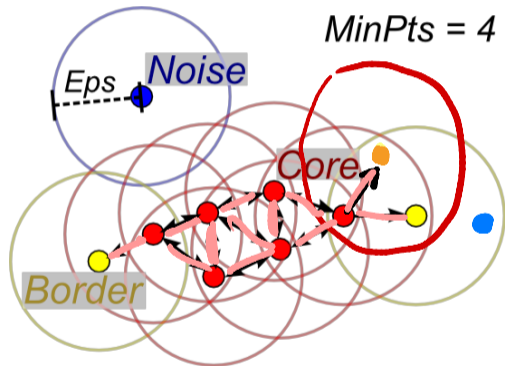
- Implementation of density based clustering available in Python and R

- Smaller value of *Eps* subdivides into small clusters

- Larger *Eps* groups larger clusters
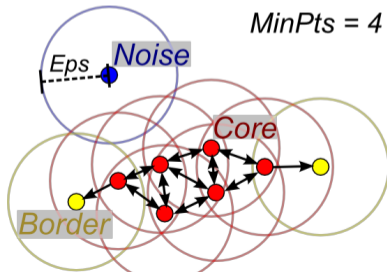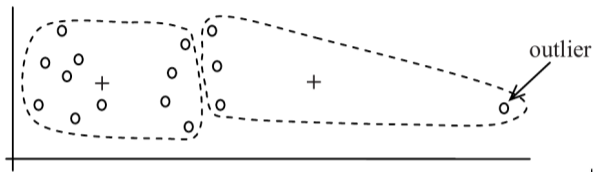

eps=0.05, min_samples=5
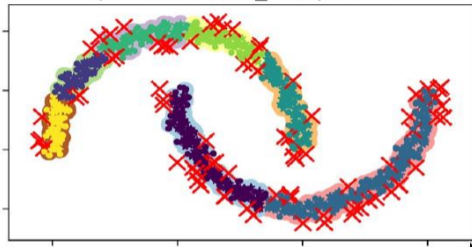
eps=0.20, min_samples=5

Outliers and clustering

- Outliers are points that lie outside natural clusters

- K Means – far away from all centroids

  - But outliers can distort the clustering process

- Density based clustering – not connected to any core point

  - But density is applied uniformly

outlier

MinPts = 4

Eps *Noise*

Core

*Border*

- An outlier is less dense than its nearest neighbours
- But difference in density may be local
- A distance metric to eliminate o_2 could make all of C_1 outliers
- C_1 has 400 points, C_2 has 100 points
- Larger distance would make ~~all of C_2 outliers with respect to C_1~~ o_2 part of C_2

C_1

C_2 o_2

o_1
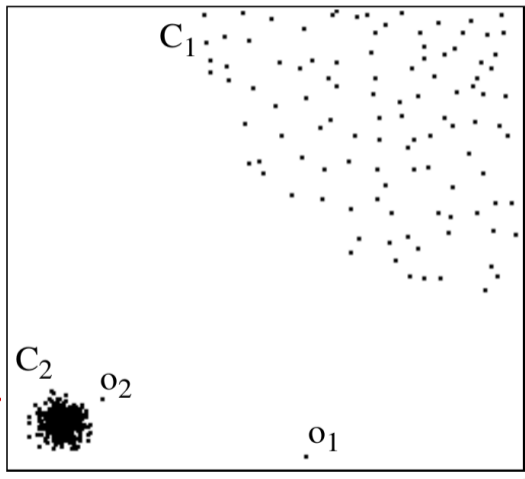
Outliers and density

- For clustering, we defined a radius *Eps* and looked for *MinPts* neighbours within that ball

- Instead, fix *MinPts* and find smallest ball with that many neighbours

- Compare *radius(p)* with radius of its neighours

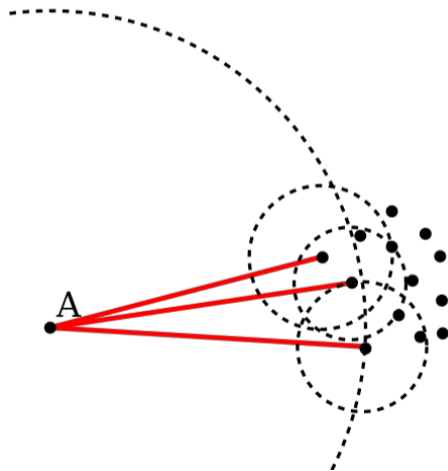- A is an outlier because its radius is much more than that of its neighbours
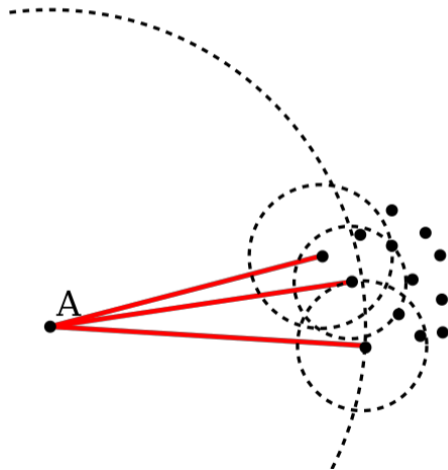
Outliers and density

- Local outlier factor *LOF(p)*

$$\frac{\text{Mean radius of } \textit{MinPts-neighbours}(p)}{\textit{radius}(p)}$$

- The smaller this ratio, the more likely that *p* is an outlier
- Comparison is local to neighbourhood, so this can deal with different densities across range of data

Outliers and density

- Local outlier factor *LOF(p)*

$$\frac{\text{Mean radius of } MinPts\text{-}neighbours(p)}{radius(p)}$$

- The smaller this ratio, the more likely that *p* is an outlier
- Comparison is local to neighbourhood, so this can deal with different densities across range of data