Clustering

Dr. Rahul Kottath

Clustering (Unsupervised Learning)

Given: Examples: $< x_1 , x_2 , \ldots x_n >$

Find: A natural clustering (grouping) of the data

Example Applications:

Identify similar energy use customer profiles

<**x**> = time series of energy usage

Identify anomalies in user behavior for computer security

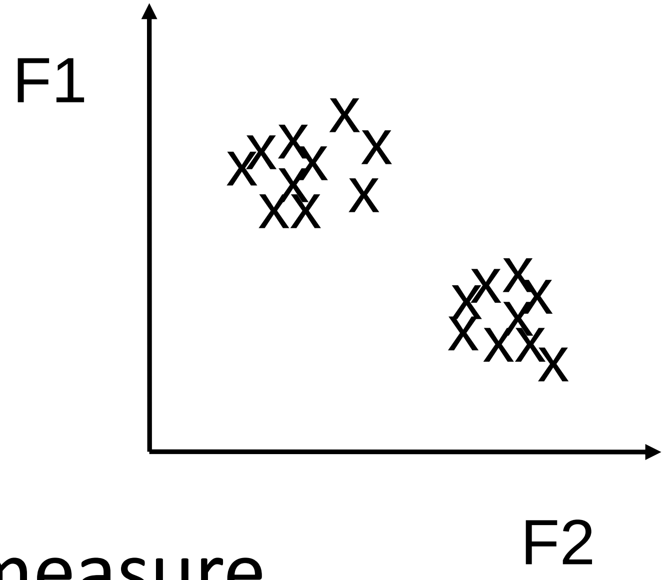
<**x**> = sequences of user commands

Why cluster?

- Labeling is expensive
- Gain insight into the structure of the data
- Find prototypes in the data

Goal of Clustering

- Given a set of data points, each described by a set of attributes, find clusters such that:
  - Inter-cluster similarity is maximized
  - Intra-cluster similarity is minimized



- Requires the definition of a similarity measure

What is Similarity?

Similarity is hard to define, but…
"*We know it when we see it*"

What properties should a distance measure have?

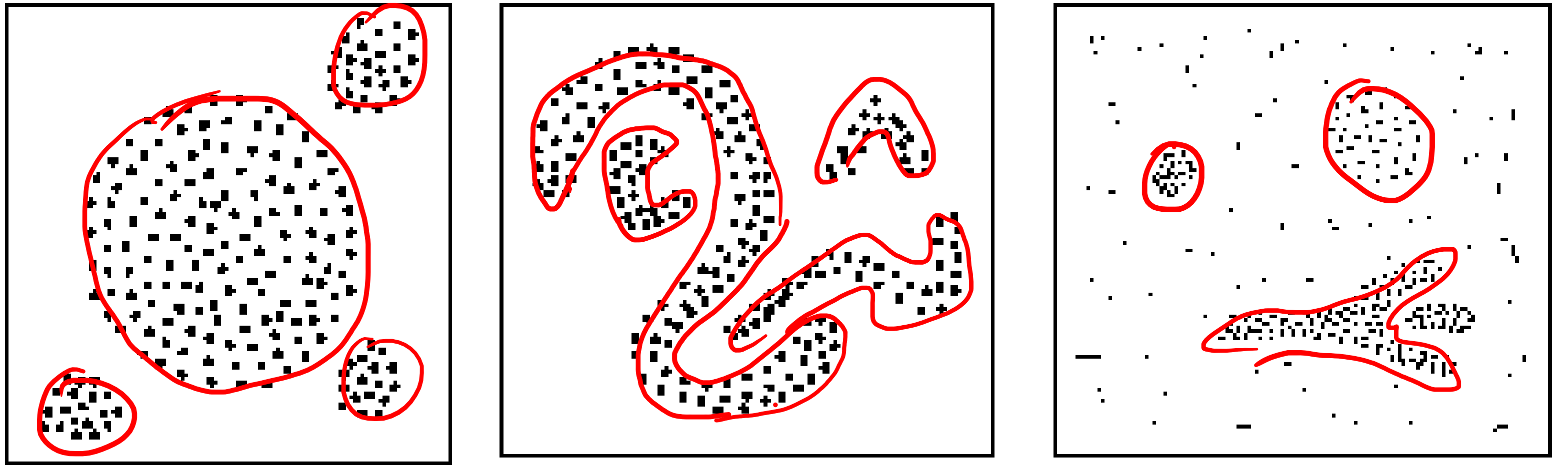
- D(A,B) = D(B,A) *Symmetry*
- D(A,A) = 0 *Constancy of Self-Similarity*
- D(A,B) = 0 iif A= B *Positivity (Separation)*
- D(A,B) $\leq$ D(A,C) + D(B,C) *Triangular Inequality*

Density-Based Clustering Methods

DBSCAN

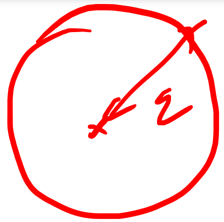
- Clustering based on density (local cluster criterion), such as density-connected points
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise (outliers)
  - One scan
  - Need density parameters as termination condition

Density-Based Clustering Methods



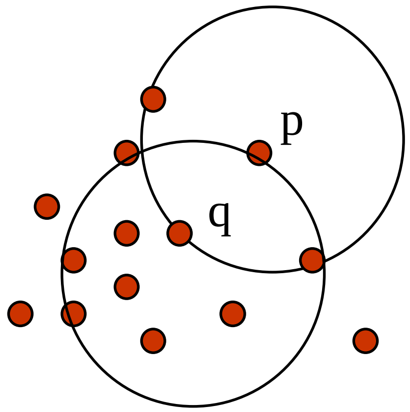
- Clustering based on density (local cluster criterion), such as density-connected points
- Each cluster has a considerable higher density of points than outside of the cluster

Density-Based Clustering: Background



- Two parameters*:*
  - ε: Maximum radius of the neighbourhood
  - ***MinPts***: Minimum number of points in an Eps-neighbourhood of that point
- $N_\varepsilon(p)$: $\{q \text{ belongs to } D \mid dist(p,q) <= \varepsilon\}$
- Directly density-reachable**:** A point ***p*** is directly density-reachable from a point ***q*** wrt. ε, ***MinPts*** if
  - 1) ***p*** belongs to $N_\varepsilon(q)$
  - 2) core point condition:

    $|N_\varepsilon(q)| >= MinPts$



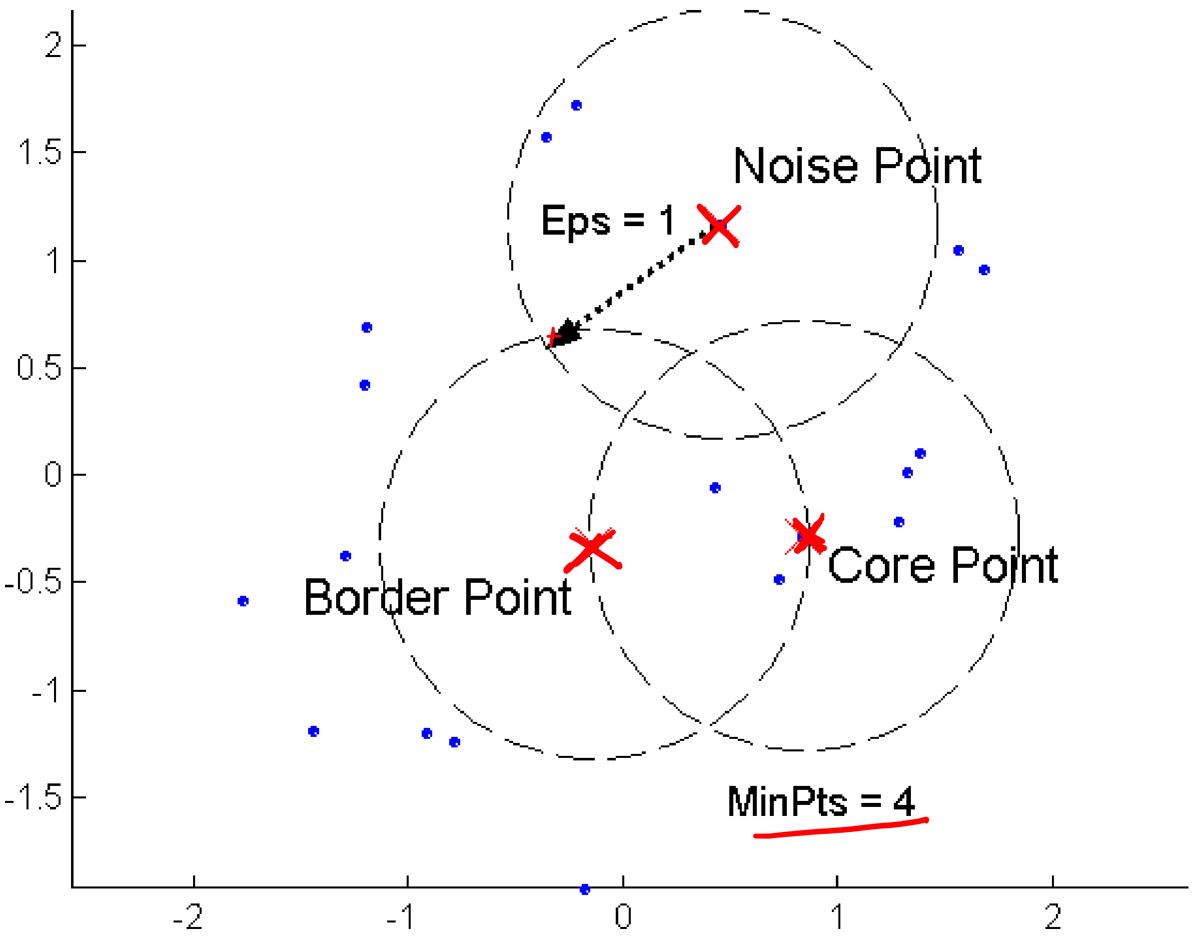
MinPts = 5

ε = 1 cm

DBSCAN: Core, Border, and Noise Points

DBSCAN

Density based spatial clustering of application with noise



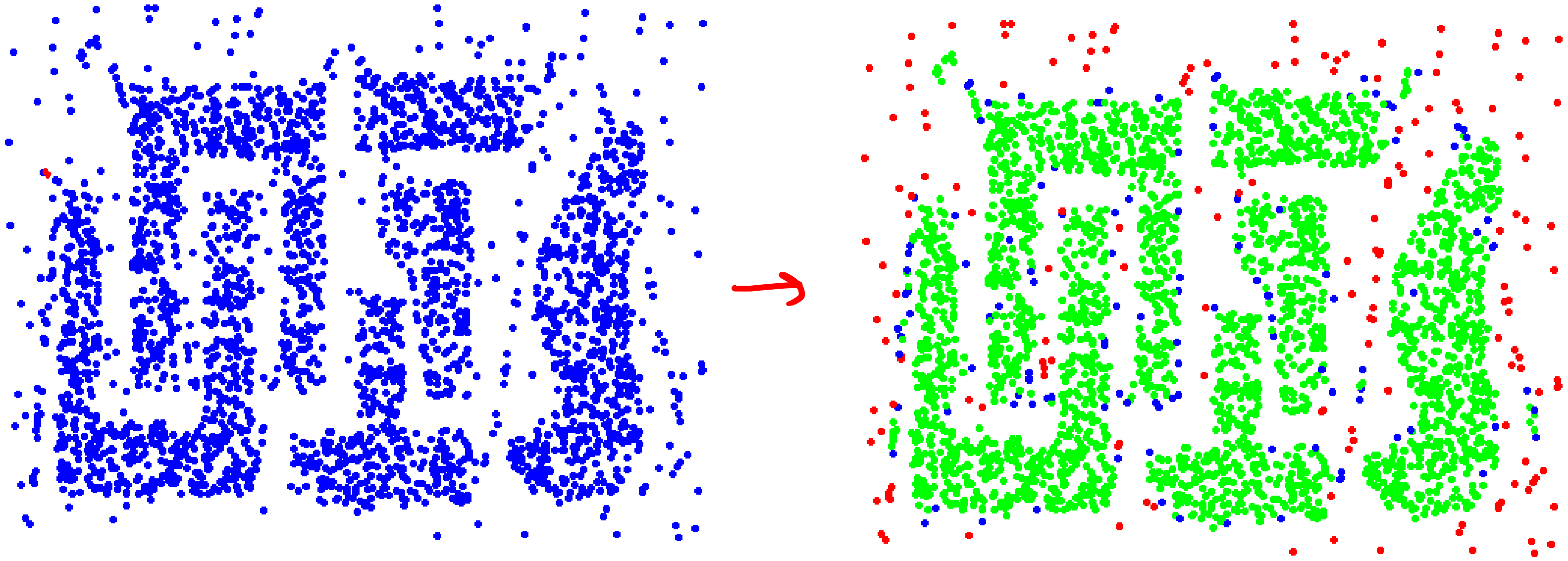
ε

Min Pts

ε

3 < 4

DBSCAN: Core, Border and Noise Points

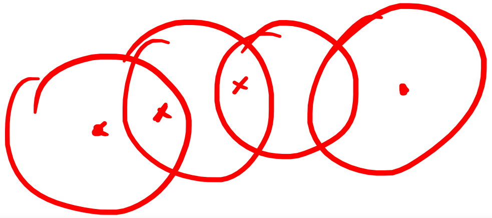


Original Points

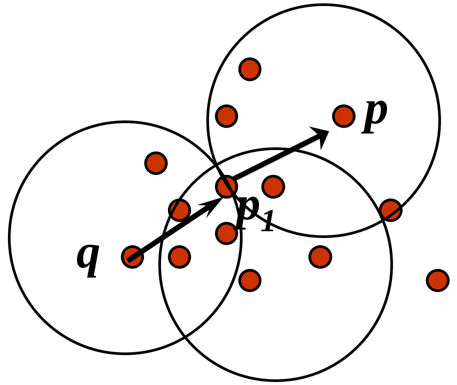
Point types: core, border and noise

Eps = 10, MinPts = 4

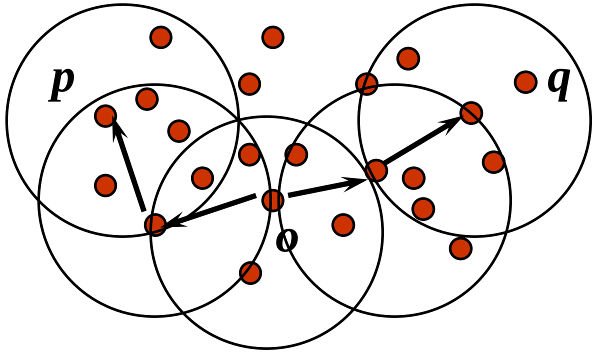
Density-Based Clustering



- Density-reachable:
  - A point p is density-reachable from a point q wrt. ε, *MinPts* if there is a chain of points $p_1, \ldots, p_n, p_1 = q, p_n = p$ such that p_{i+1} is directly density-reachable from p_i



- Density-connected
  - A point p is density-connected to a point q wrt. ε, *MinPts* if there is a point o such that both, p and q are density-reachable from o wrt. ε and *MinPts*.

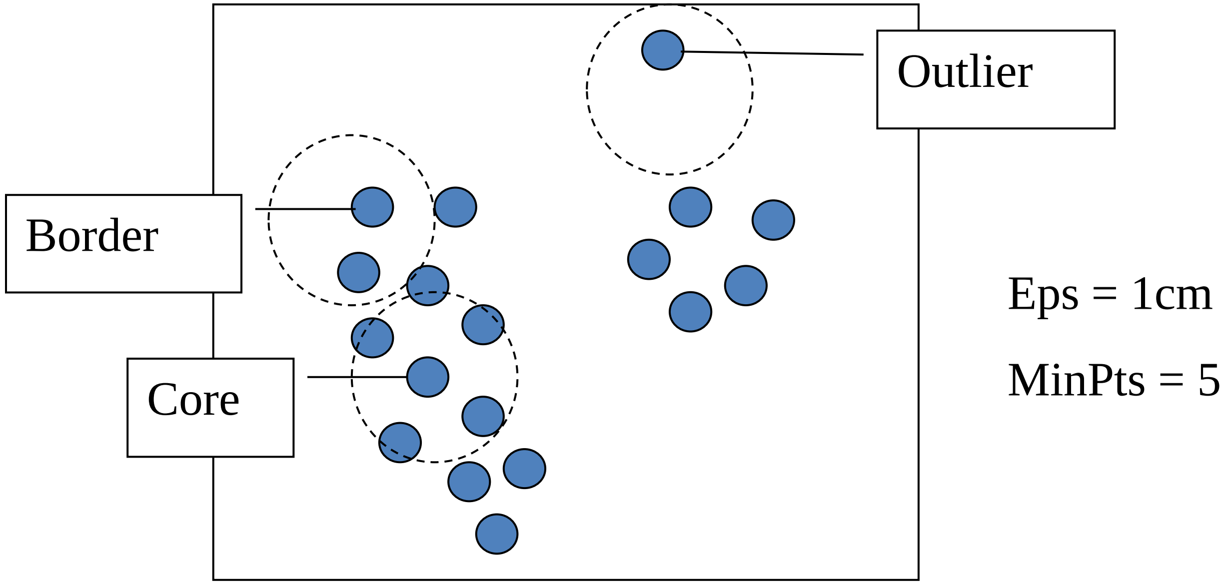


DBSCAN: Density Based Spatial Clustering of Applications with Noise

- Relies on a *density-based* notion of cluster: A *cluster* is defined as a maximal set of density-connected points
- Discovers clusters of arbitrary shape in spatial databases with noise

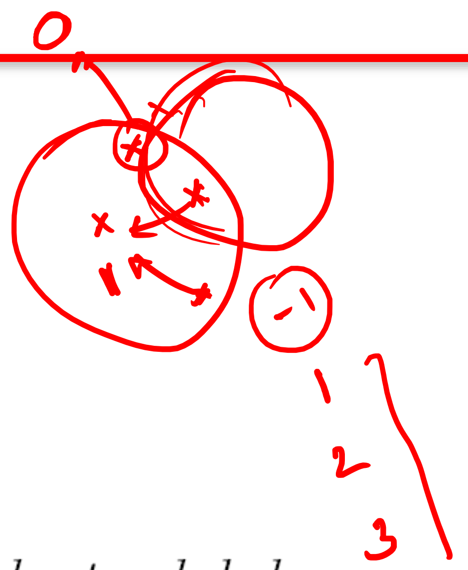
-1 → Noise

0
1
2
} labels clustering



DBSCAN Algorithm

- Eliminate noise points
- Perform clustering on the remaining points



$current\_cluster\_label \leftarrow 1$

for all core points **do**

    if the core point has no cluster label **then**

        $current\_cluster\_label \leftarrow current\_cluster\_label + 1$

        Label the current core point with cluster label $current\_cluster\_label$

    end if

    for all points in the Eps-neighborhood, except i^{th} the point itself **do**

        if the point does not have a cluster label **then**

            Label the point with cluster label $current\_cluster\_label$

        end if

    end for

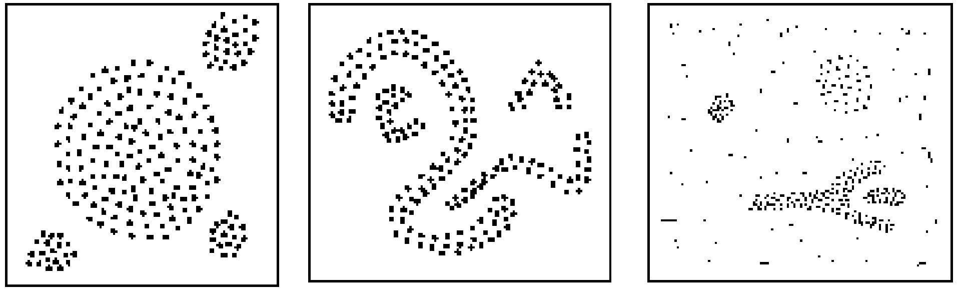
end for

DBSCAN Properties

- Generally takes O(nlogn) time

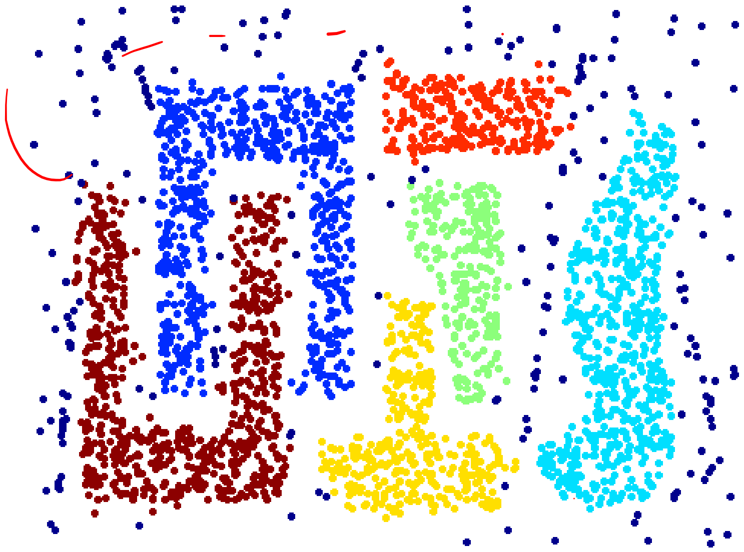
$O(n^2)$

- Still requires user to supply Minpts and ε
- Advantage
  - Can find points of arbitrary shape
  - Requires only a minimal (2) of the parameters



When DBSCAN Works Well

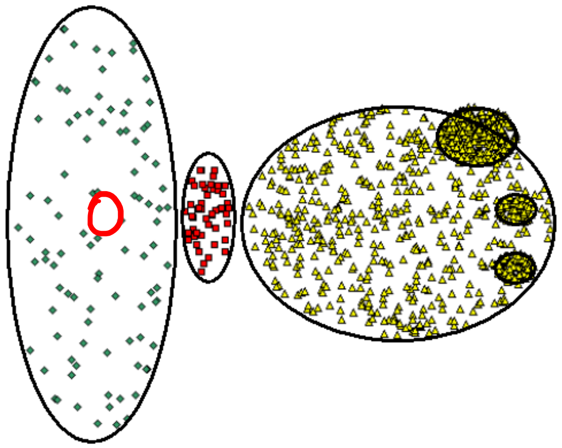
Original Points



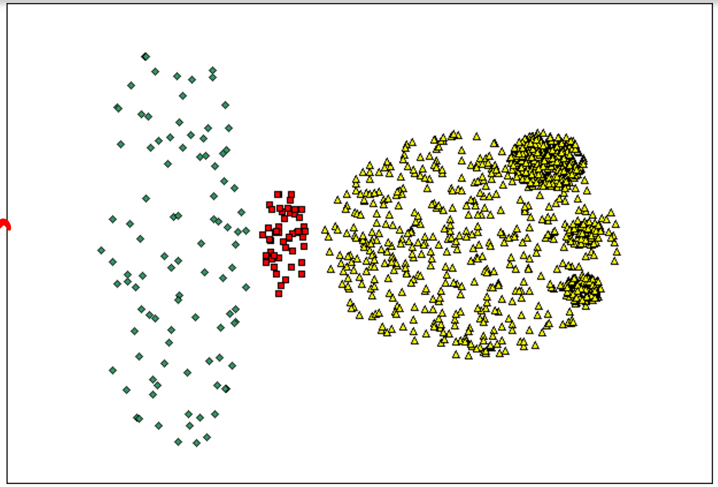
Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

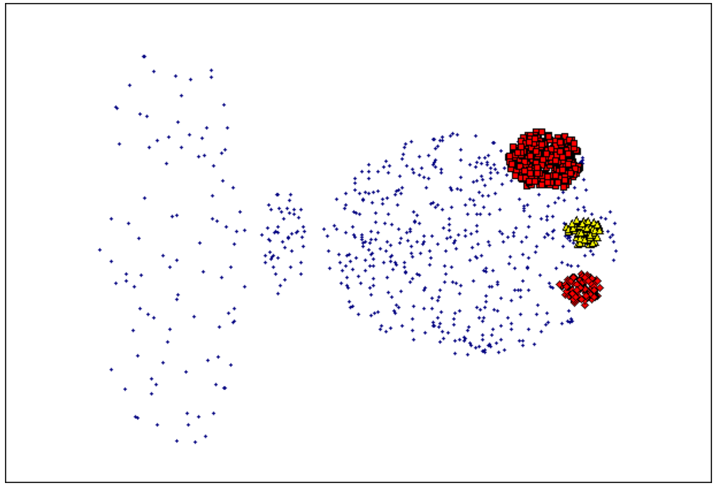
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=large value).

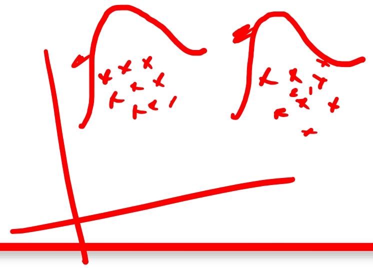


(MinPts=4, Eps=small value; min density increases)

- **Varying densities**
- **High-dimensional data**

Gaussian Mixture Models

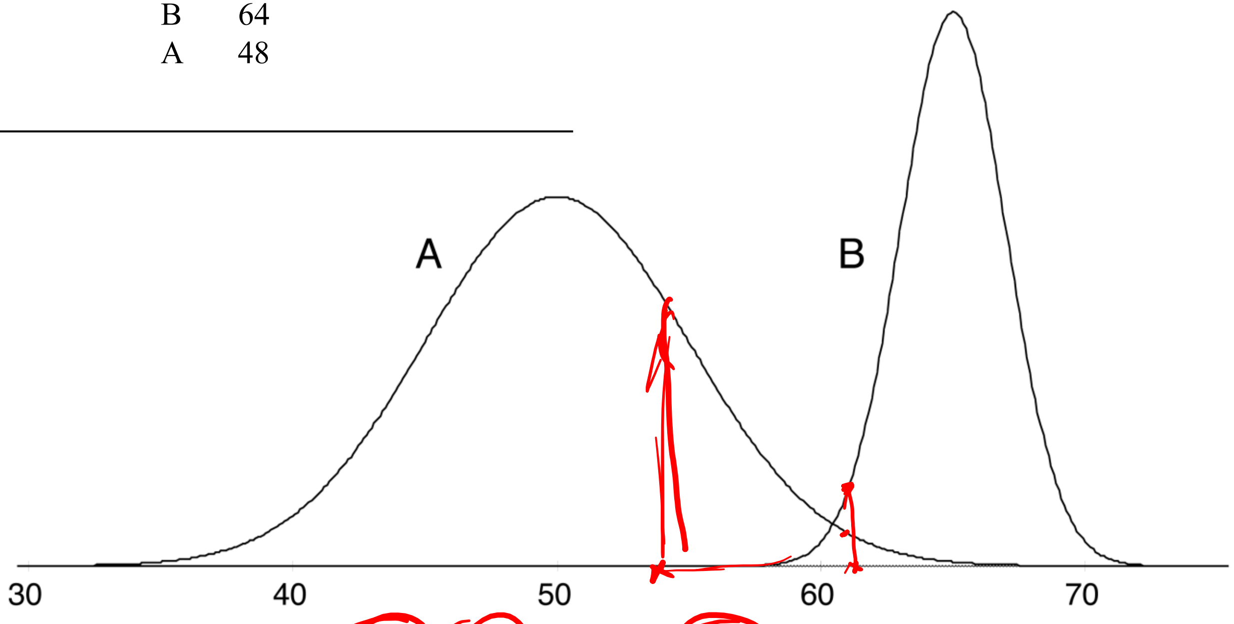
Finite mixtures



- Probabilistic clustering algorithms model the data using a mixture of distributions
  - Each cluster is represented by one distribution
- The distribution governs the probabilities of attributes values in the corresponding cluster
- They are called finite mixtures because there is only a finite number of clusters being represented
- Usually individual distributions are normal distribution
- Distributions are combined using cluster weights

A two-class mixture model

data											
A	51	B	62	B	64	A	48	A	39	A	51
A	43	A	47	A	51	B	64	B	62	A	48
B	62	A	52	A	52	A	51	B	64	B	64
B	64	B	64	B	62	B	63	A	52	A	42
A	45	A	51	A	49	A	43	B	63	A	48
A	42	B	65	A	48	B	65	B	64	A	41
A	46	A	48	B	62	B	66	A	48		
A	45	A	49	A	43	B	65	B	64		
A	45	A	46	A	40	A	46	A	48		



$\mu_A=50$, $\sigma_A=5$, $p_A=0.6$ $\mu_B=65$, $\sigma_B=2$, $p_B=0.4$

Using the mixture model

The probability of an instance x belonging to cluster A is:

$$\Pr[A \mid x] = \frac{\Pr[x \mid A]\Pr[A]}{\Pr[x]} = \frac{f(x;\mu_A,\sigma_A)\,p_A}{\Pr[x]}$$

with

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

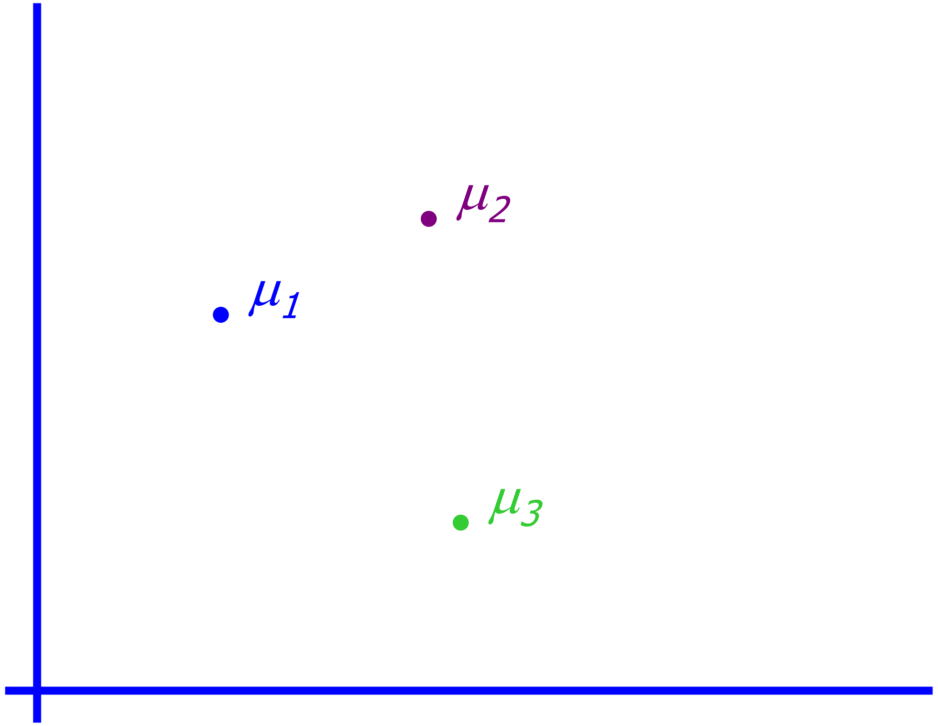
← Gaussian

The *likelihood* of an instance given the clusters is:

$$\Pr[x \mid \text{the distributions}] = \sum_i \Pr[x \mid \text{cluster}_i]\Pr[\text{cluster}_i]$$

The GMM assumption

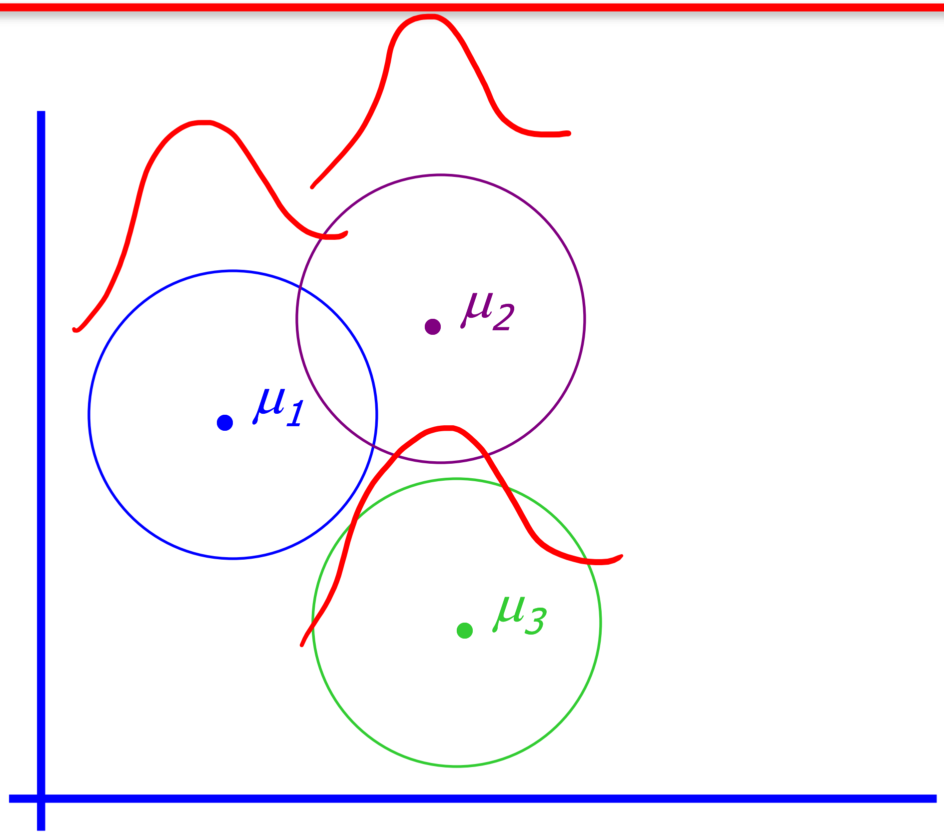
- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i



The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \mathbf{I}$

Assume that each datapoint is generated according to the following recipe:

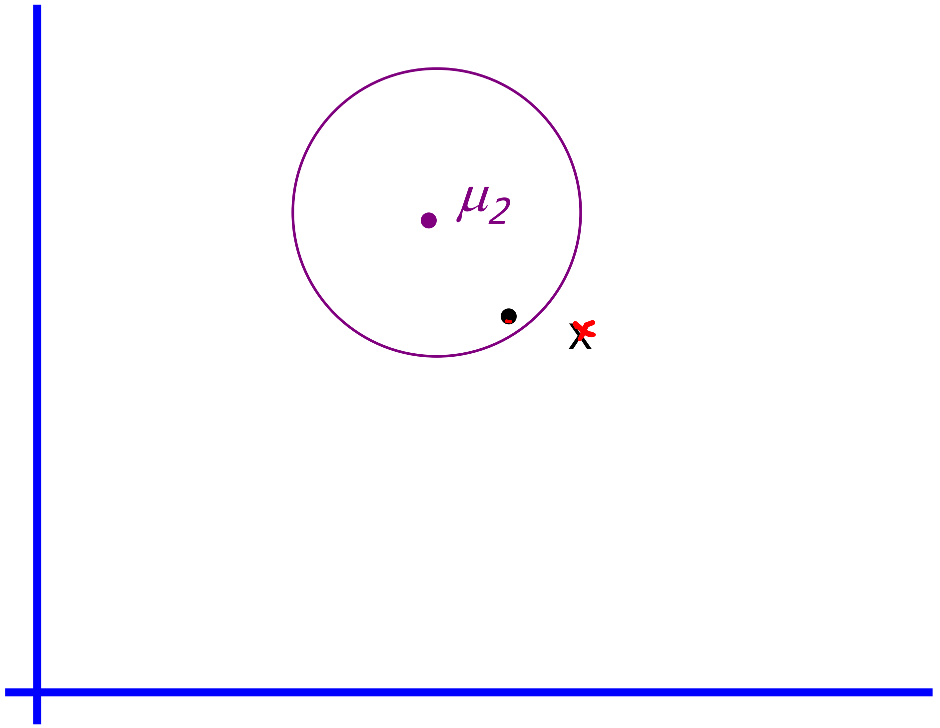


The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \boldsymbol{I}$

Assume that each datapoint is generated according to the following recipe:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.
2. Datapoint ~ N(μ_i, $\sigma^2 \boldsymbol{I}$)



Learning the clusters

(μ_1, σ_1) (μ_2, σ_2) $\cdots$ (μ_k, σ_k)

- Assume we know that there are *k* clusters
- To learn the clusters we need to determine their parameters
  - I.e. their means and standard deviations (μ, σ) GMM
- We actually have a performance criterion: the likelihood of the training data given the clusters
- Fortunately, there exists an algorithm that finds a local maximum of the likelihood

(Expectation Maximization)

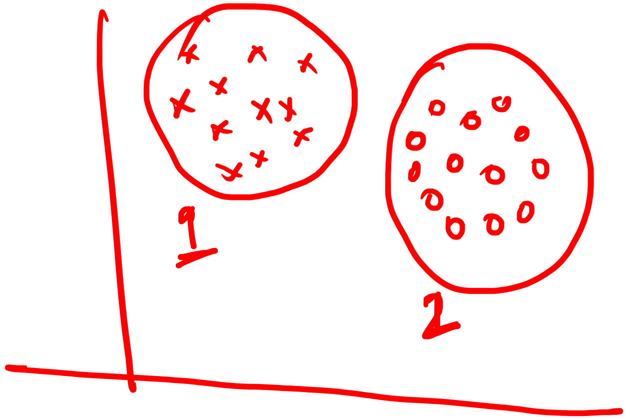
EM

- Hierarchical
- K Mean
- DBSCAN
- GMM.

Clustering performance evaluation

Cluster Validity

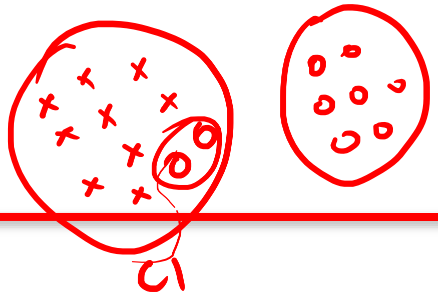
- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall
- For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?
- But "clusters are in the eye of the beholder"!
- Then why do we 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 performance evaluation

- Evaluating the performance of a clustering algorithm is not as trivial as counting the number of errors or the precision and recall of a supervised classification algorithm.
- In particular any evaluation metric should not take the absolute values of the cluster labels into account but rather if this clustering define separations of the data similar to some ground truth set of classes or satisfying some assumption such that members belong to the same class are more similar than members of different classes according to some similarity metric.

Homogeneity, completeness



Given the knowledge of the ground truth class assignments of the samples, it is possible to define some intuitive metric using conditional entropy analysis.

In particular Rosenberg and Hirschberg (2007) define the following two desirable objectives for any cluster assignment:

- **homogeneity**: each cluster contains only members of a single class.
- **completeness**: all members of a given class are assigned to the same cluster.

V-measure

Their harmonic mean called V-measure

$$v = \frac{(1+\beta) \times \text{homogeneity} \times \text{completeness}}{(\beta \times \text{homogeneity} + \text{completeness})}$$

Python Packages needed

- pandas
  - Data Analytics
- numpy
  - Numerical Computing
- matplotlib.pyplot
  - Plotting graphs
- Sklearn, Scipy
  - Clustering Classes

Implementation Using sklearn

Let's go to Jupyter Notebook!