



# CIS 678 - Machine Learning

Clustering Algorithms



# Clustering Algorithms

- **k-means:** Centroid Based
- **Hierarchical clustering:** Distance connectivity based
- **GMM:** Distribution based
- **DBSCAN:** Density Based



# Clustering Algorithms

- k-means: Centroid Based
- **Hierarchical clustering:** Distance connectivity based
- **GMM:** Distribution based
- **DBSCAN:** Density Based

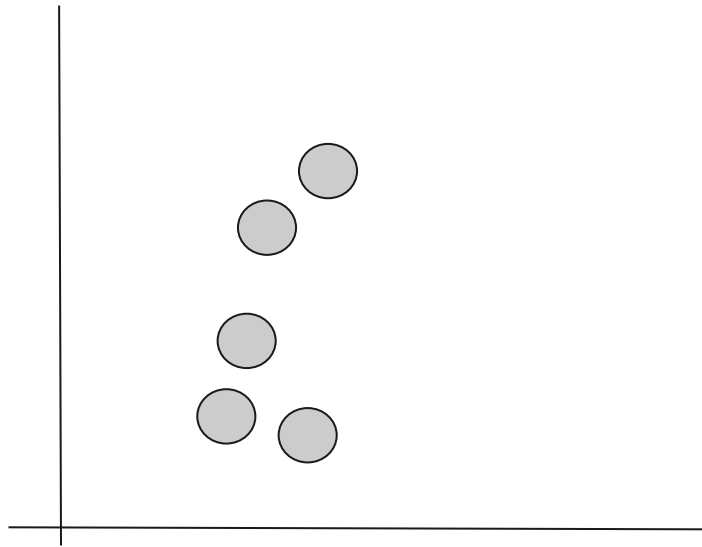


# k-means Clustering

- Centroid based
- Only works for numeric data only
- Explicit k-centroid inputs (initialization)
- Iterative algorithm

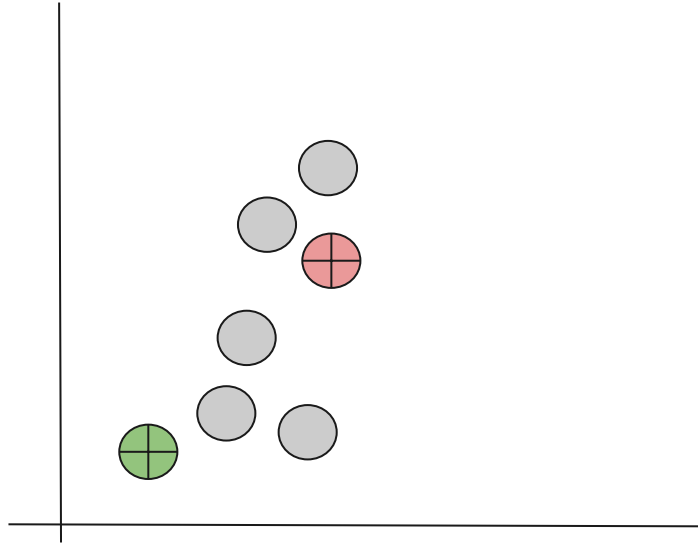


## k-means Clustering



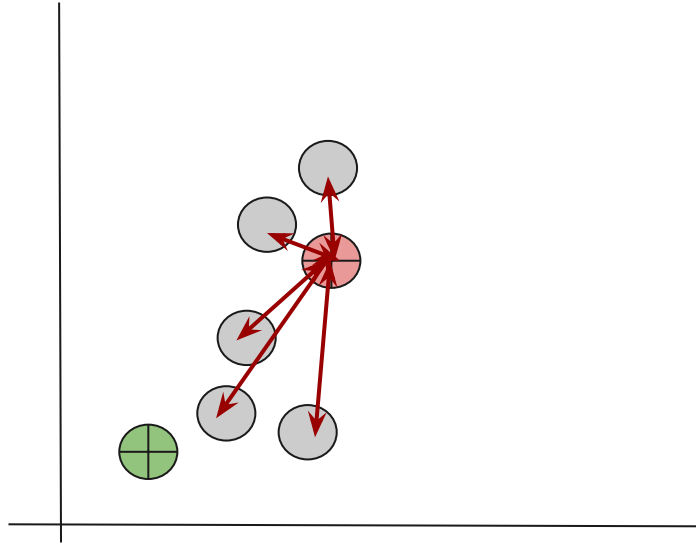
- 5 given data points of forms  $(x_i, y_i)$ : numeric

# k-means Clustering



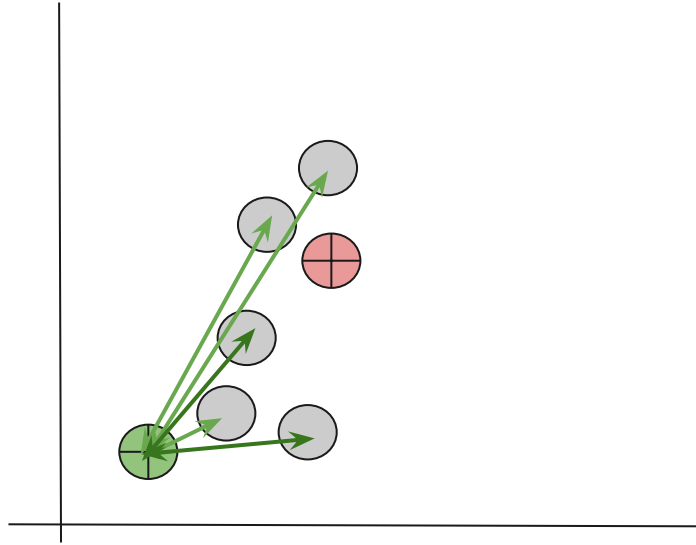
- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, .. k$

# k-means Clustering



- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- **Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$**

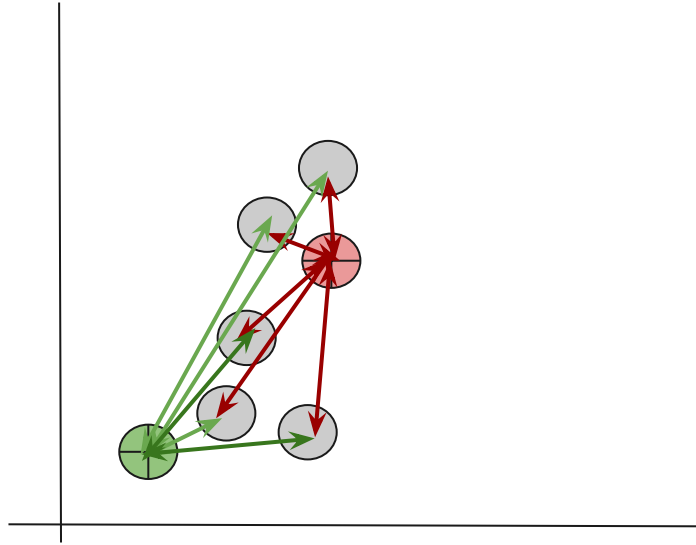
# k-means Clustering



- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- **Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$**

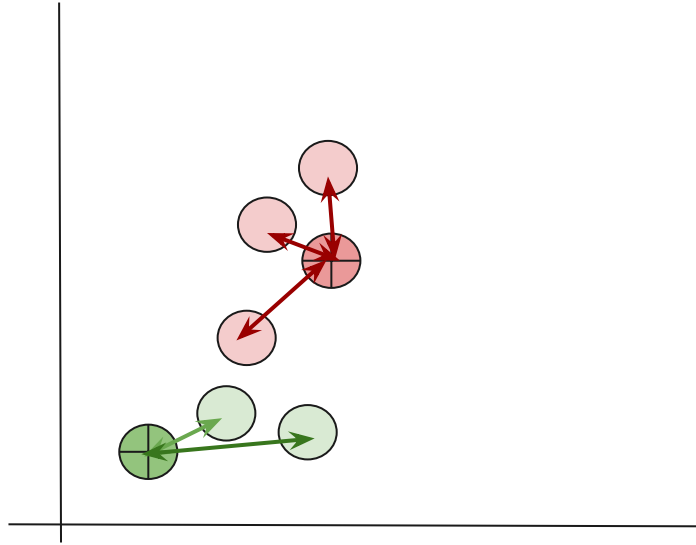


# k-means Clustering



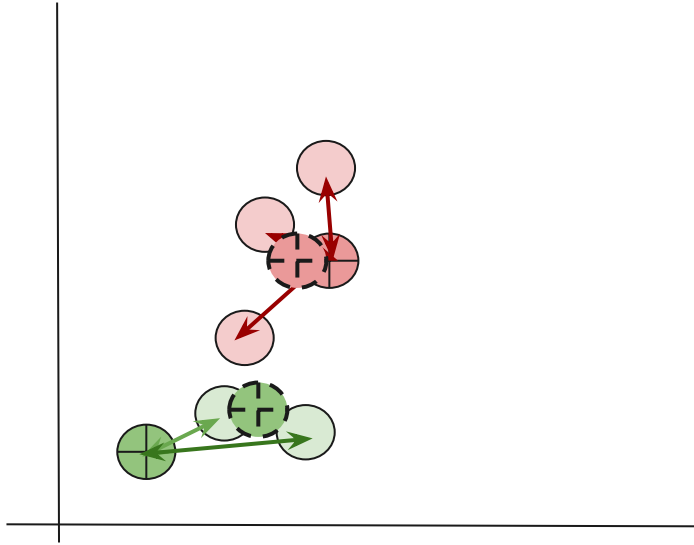
- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- **Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$**

# k-means Clustering



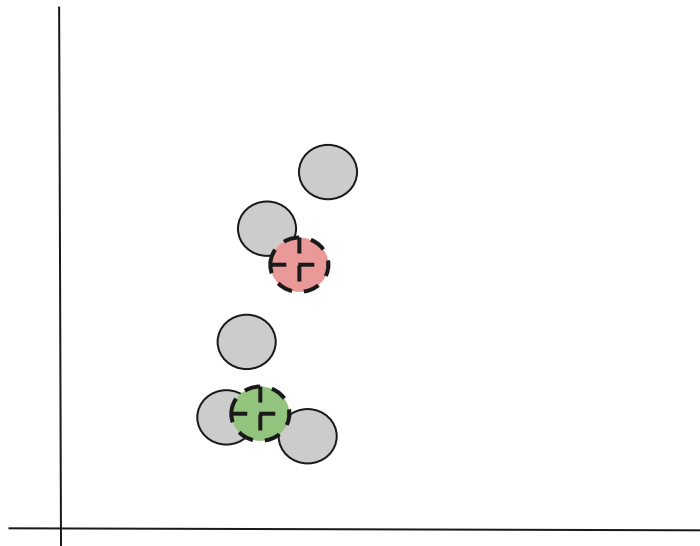
- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$
- **Assign cluster labels based on the distances measured.**

# k-means Clustering



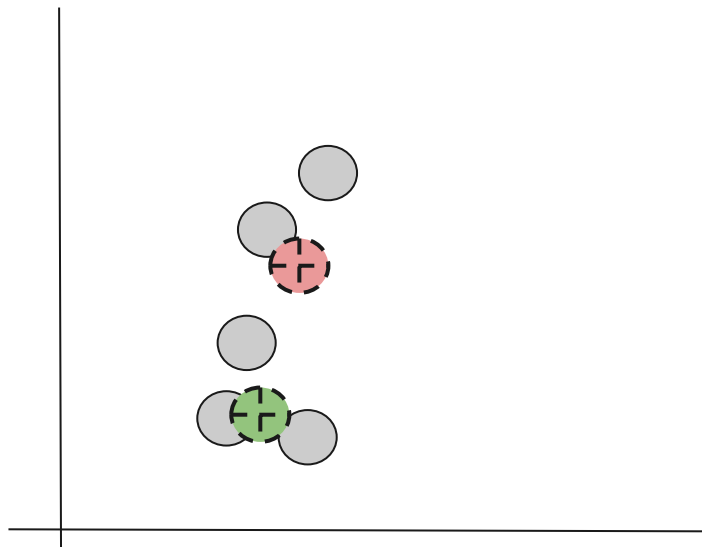
- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$
- Assign cluster labels based on the distances measured.
- **Update the two means:**
  - $m_x = 1/|x_i| \sum x_i$
  - $m_y = 1/|y_i| \sum y_i$

# k-means Clustering



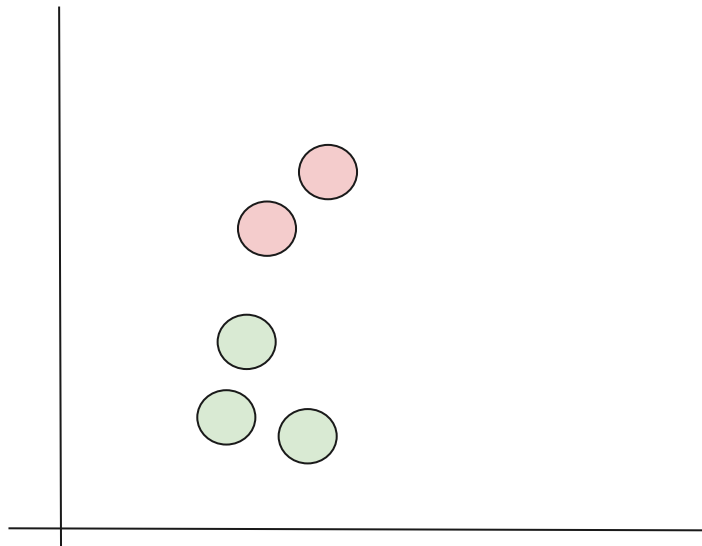
- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$
- Assign cluster labels based on the distances measured.
- **Update the two means:**
  - $m_x = 1/|x_i| \sum x_i$
  - $m_y = 1/|y_i| \sum y_i$

# k-means Clustering



- 5 given data points of forms  $(x_i, y_i)$ : numeric
  - For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
  - Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$
  - Assign cluster labels based on the distances measured.
  - **Update the two means:**
    - $m_x = 1/|x_i| \sum x_i$
    - $m_y = 1/|y_i| \sum y_i$
- Iterate until the means  $(m_x, m_y)_j$  don't move; or a predefined number of iterations are run

# k-means Clustering



- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$
- Assign cluster labels based on the distances measured.
- **Update the two means:**
  - $m_x = 1/|x_i| \sum x_i$
  - $m_y = 1/|y_i| \sum y_i$

- Iterate until the means  $(m_x, m_y)_j$  don't move; or a predefined number of iterations are run
- **Likely the final cluster configuration**

# k-means Clustering

Given an initial set of  $k$  means  $m_1^{(1)}, \dots, m_k^{(1)}$  (see below), the algorithm proceeds by alternating between two steps:<sup>[7]</sup>

1. **Assignment step:** Assign each observation to the cluster with the nearest mean: that with the least squared [Euclidean distance](#).<sup>[8]</sup> (Mathematically, this means partitioning the observations according to the [Voronoi diagram](#) generated by the means.)

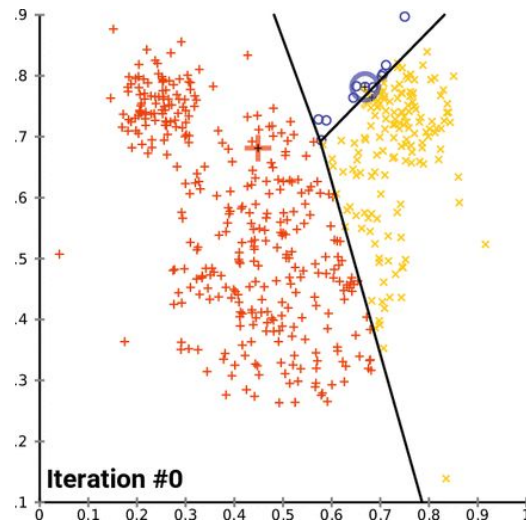
$$S_i^{(t)} = \left\{ x_p : \|x_p - m_i^{(t)}\|^2 \leq \|x_p - m_j^{(t)}\|^2 \forall j, 1 \leq j \leq k \right\},$$

where each  $x_p$  is assigned to exactly one  $S^{(t)}$ , even if it could be assigned to two or more of them.

2. **Update step:** Recalculate means ([centroids](#)) for observations assigned to each cluster.

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

[k-means \[wiki\]](#)





# k-modes is the categorical equivalent!

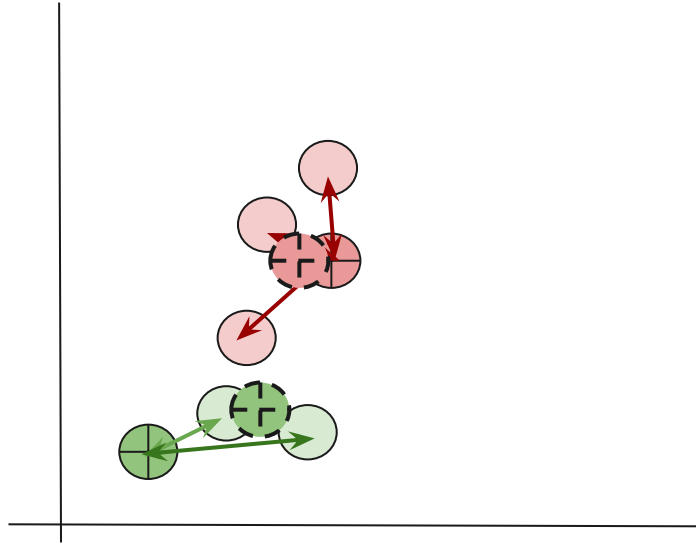
- Centroid based
- Only works for categorical data
- Explicit k-centroid inputs (initialization)
- Iterative algorithm
- Only

1. Pick K observations at random and use them as leaders/clusters
2. Calculate the dissimilarities and assign each observation to its closest cluster
3. Define new modes for the clusters
4. Repeat 2-3 steps until there are no re-assignment required

- One simple metric: number of categorical value match
- Whiteboarding

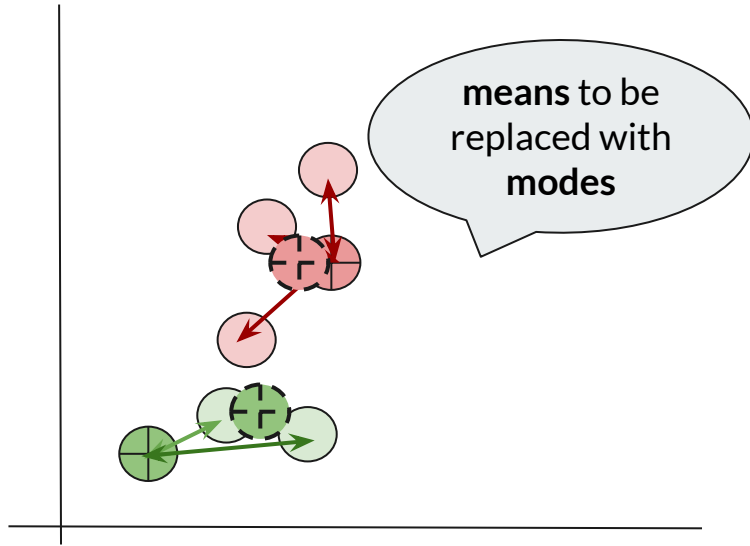


# k-means Clustering



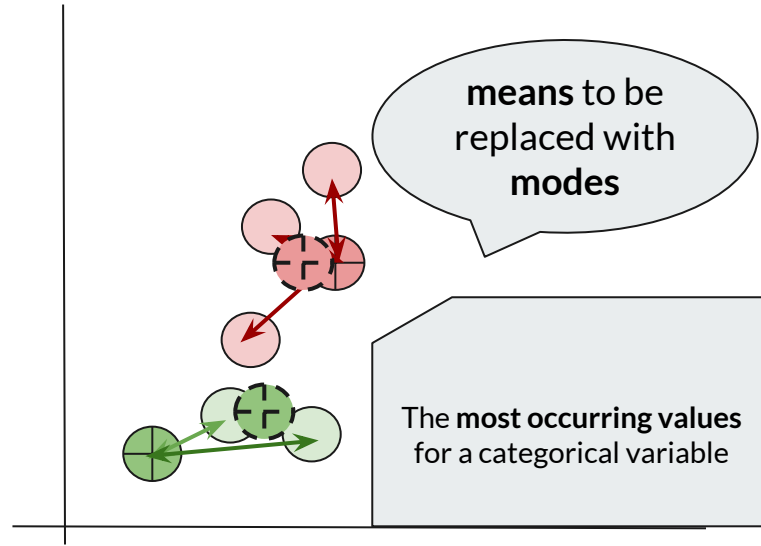
- 5 given data points of forms  $(x_i, y_i)$ : numeric
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- Compares distance between the means  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$
- Assign cluster labels based on the distances measured.
- **Update the two means:**
  - $m_x = 1/|x_i| \sum x_i$
  - $m_y = 1/|y_i| \sum y_i$

# k-modes Clustering



- 5 given data points of forms  $(x_i, y_i)$ : **categorical**
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initials modes  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- Compares distance between the modes  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$
- Assign cluster labels based on the distances measured.
- **Update the two modes:**
  - $m_x = 1/|x_i| \text{ mode}(x_i)$
  - $m_y = 1/|y_i| \text{ mode}(x_i)$

# k-modes Clustering



- 5 given data points of forms  $(x_i, y_i)$ : **categorical**
- For  $k = 2$ , the algorithm starts with two random (users can also provide based on their analysis/intuition) initial modes  $(m_x, m_y)_j$  where  $j = 1, \dots, k$
- Compares distance between the modes  $(m_x, m_y)_j$  and all given (5) data points and for all  $j$
- Assign cluster labels based on the distances measured.
- **Update the two modes:**
  - $m_x = 1/|x_i| \text{ mode}(x_i)$
  - $m_y = 1/|y_i| \text{ mode}(y_i)$



## How about when you have mixed data types?

- Either you have to convert data in one type
  - What could be the challenges?



## How about when you have mixed data types?

- Either you have to convert data in one type
  - What could be the challenges?
- Or, you have to have an algorithm that updates centroids



# Clustering Algorithms

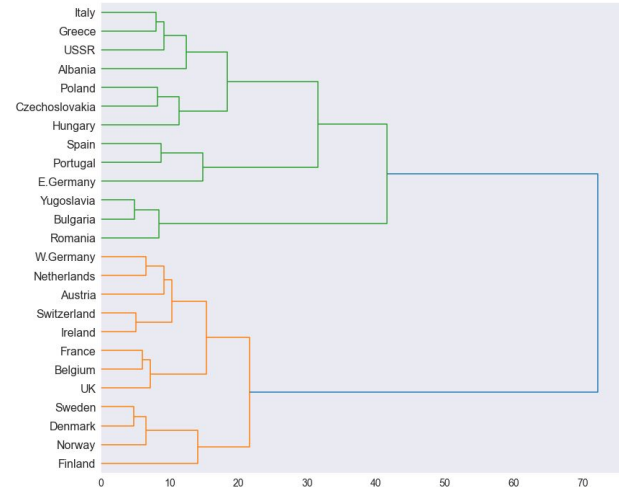
- K-means: Centroid Based
- Hierarchical clustering: Distance connectivity based
- GMM: Distribution based
- DBSCAN: Density Based

# Hierarchical clustering

**Agglomerative:** This is a “bottom up” approach. Each observation starts as a new cluster, and pairs of clusters are merged as one moves up the hierarchy.

**Divisive:** This is a “top down” approach. All data starts as one cluster, and recursively splits into two/multiple clusters.

Grouping countries according to their protein consumption.  
(dendrogram graph below)





# Clustering Algorithms

- K-means: Centroid Based
- Hierarchical clustering: Distance connectivity based
- GMM: Distribution based
- DBSCAN: Density Based



# Gaussian Mixture Model (GMM)

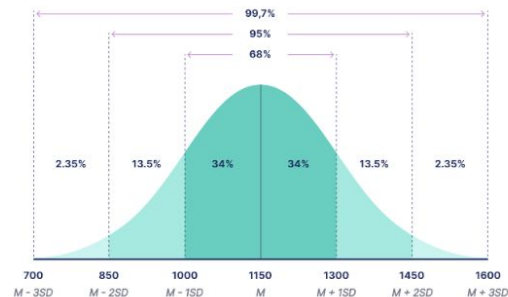
Basic idea from the model title itself:

- **Gaussian/Normal distribution:** distribution modeling some continuous variables such as: population height/weight in a certain region, yearly sales of a business etc.

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Using the empirical rule in a normal distribution



# Gaussian Mixture Model (GMM)

Basic idea from the model title itself:

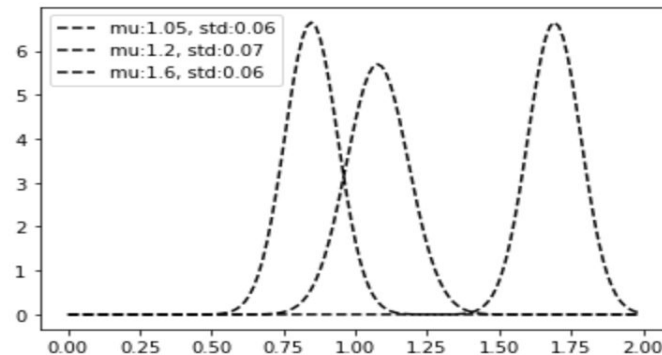
- **Gaussian/Normal distribution:** distribution modeling some continuous variables such as: population height/weight in a certain region, yearly sales of a business etc.

An example problem

- Display the weight distribution of grade 5,6 and 10 students
- Choose an x (confusing between g 5 and 6) and explain through words

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



# Gaussian Mixture Model (GMM)

Basic idea from the model title itself:

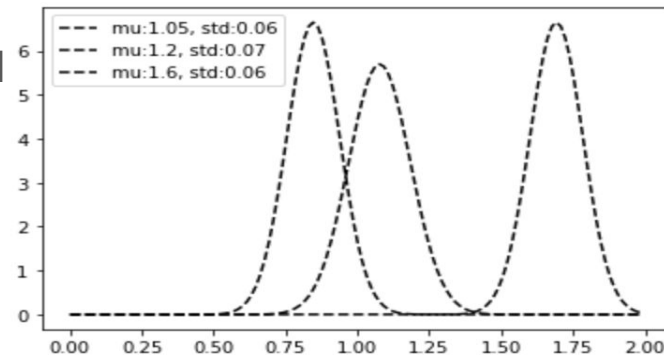
- **Gaussian/Normal distribution:** distribution modeling some continuous variables such as: population height/weight in a certain region, yearl sales of a business etc.
- **Mixture:** more than one object/component

$$p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x | \mu_k, \sigma_k)$$

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



# Gaussian Mixture Model (GMM)

- Fig at the right shows the graphical model (B Net) of GMM
- $\mathbf{X}$ : feature vector; in our example case a vector with apples (size, color)
- $\mathbf{Z}$ : encoding of clusters (1-of-K is 1, rest are 0s),  $K$  is the number of clusters.
- Essentially GMM models(learns) the joint distribution (an example of what we call a generative model)

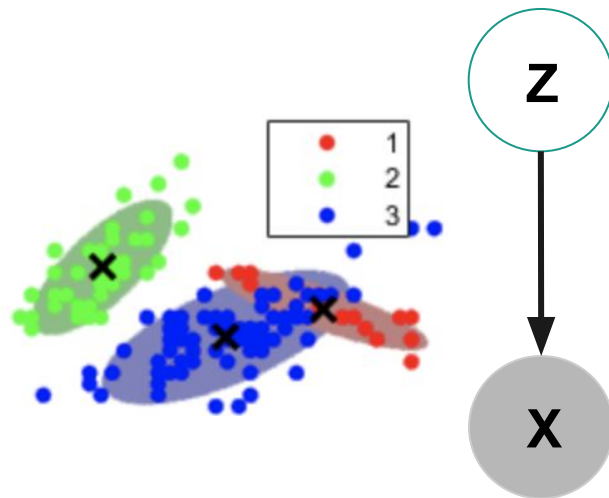
$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$$

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

$$0 \leq \pi_k \leq 1$$

$$\sum_{k=1}^K \pi_k = 1$$

$$p(\mathbf{z}) = \prod_{k=1}^K \pi_k^{z_k}$$



Model parameters(all  $k$  s)

$$\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$$

# Gaussian Mixture Model (GMM)

- Fig at the right shows the graphical model (B Net) of GMM
- $\mathbf{X}$ : feature vector; in our example case a vector with apples (size, color)
- $\mathbf{Z}$ : encoding of clusters (1-of-K is 1, rest are 0s),  $K$  is the number of clusters.
- Essentially GMM models(learns) the joint distribution (an example of what we call a generative model)

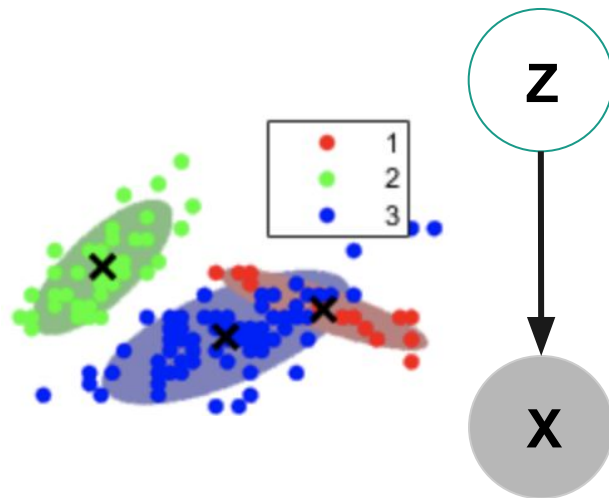
$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$$

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

$$0 \leq \pi_k \leq 1$$

$$\sum_{k=1}^K \pi_k = 1$$

$$p(\mathbf{z}) = \prod_{k=1}^K \pi_k^{z_k}$$



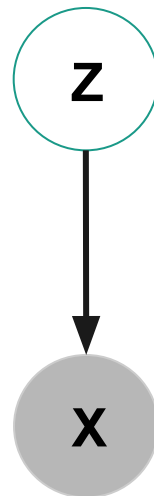
Model parameters(all  $k$  s)

$$\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$$

# Gaussian Mixture Model (GMM)

- If given model parameters, using Bayes rule, we can estimate the class conditional probabilities for a given query  $\mathbf{X}$

$$\begin{aligned} p(z_k = 1 | \mathbf{x}) &= \frac{p(z_k = 1)p(\mathbf{x} | z_k = 1)}{\sum_{j=1}^K p(z_j = 1)p(\mathbf{x} | z_j = 1)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \end{aligned}$$



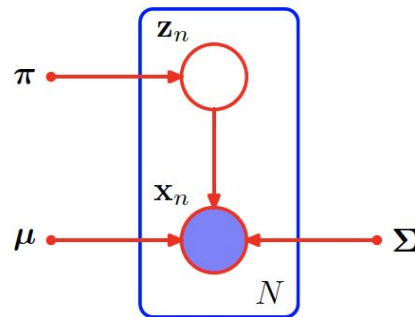
# Maximum Likelihood Learning

- Start with random parameter initializations
- Optimize the following likelihood function for  $N$  data points

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$

- Some popular techniques:
  - Expectation Maximization algorithm
  - We can also use **gradient based optimization techniques**

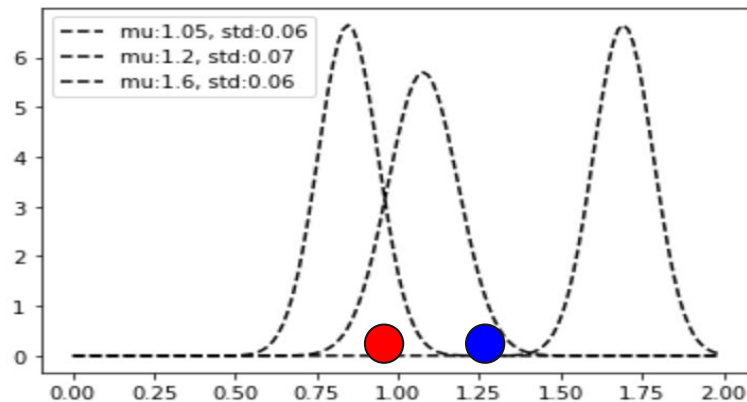
$$\begin{aligned} \log p(X) &= \log(p(Z)p(X|Z)) : \\ &= \log p(Z) + \log p(X|Z) \end{aligned}$$



## Why do we need GMM ?

- Hard cluster assignments (k-means, hierarchical clustering)

Gmm 3 components

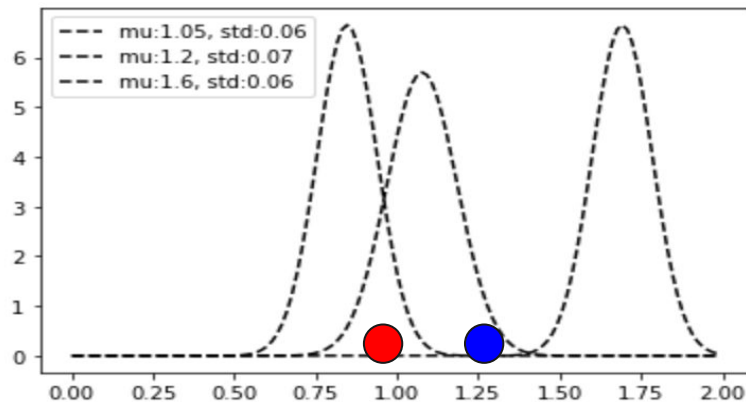




## Why do we need GMM ?

- Hard cluster assignments (k-means, hierarchical clustering)
- Soft assignments: cluster assignment probability scores,  $p(Z|X)$

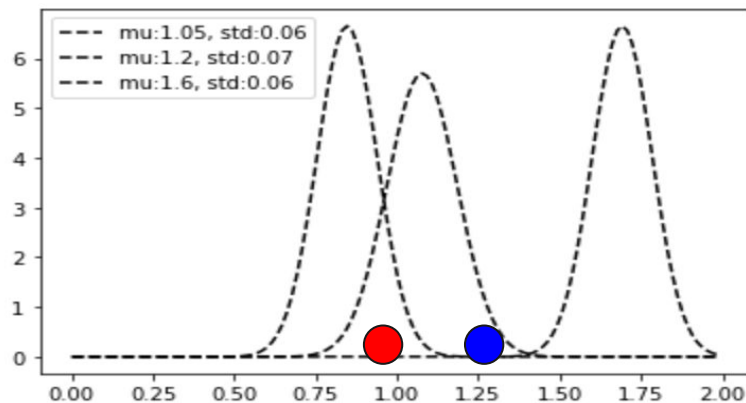
Gmm 3 components



## Why do we need GMM ?

- Hard cluster assignments (k-means, hierarchical clustering)
- Soft assignments: cluster assignment probability scores,  $p(Z|X)$ 
  - GMM offers us a probability distribution over the (features & clusters) space, and this can be used as a part of a larger/complex modeling tasks,  $P(X, Z)$

Gmm 3 components





# GMM

[Notebook presentation](#)

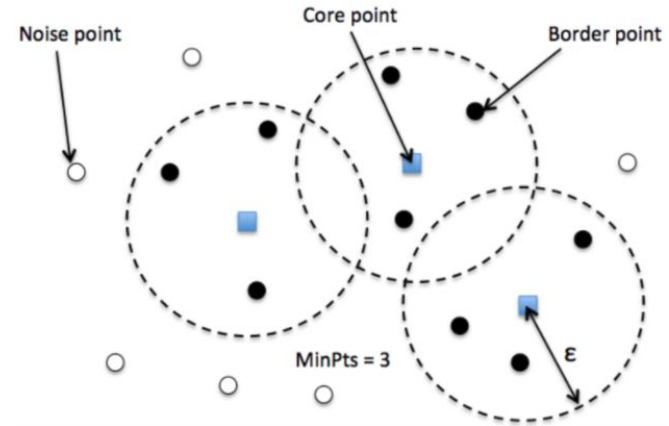


# Clustering Algorithms

- K-means: Centroid Based
- Hierarchical clustering: Distance connectivity based
- GMM: Distribution based
- DBSCAN: Density Based

# DBSCAN

- Density-Based Spatial Clustering of Applications with Noise
- Two parameters:
  - **minPts**: The minimum number of points (a threshold) clustered together to be considered dense.
  - **eps** ( $\epsilon$ ): A distance measure that will be used to locate the points in the neighborhood of any point.
- The algorithm proceeds by arbitrarily picking up a point in the dataset.
- If there are at least '**minPoint**' points within a **radius of ' $\epsilon$ '** to the point then we consider all these points to be part of the same cluster.
- The clusters are then expanded by recursively repeating the neighborhood calculation for each neighboring point

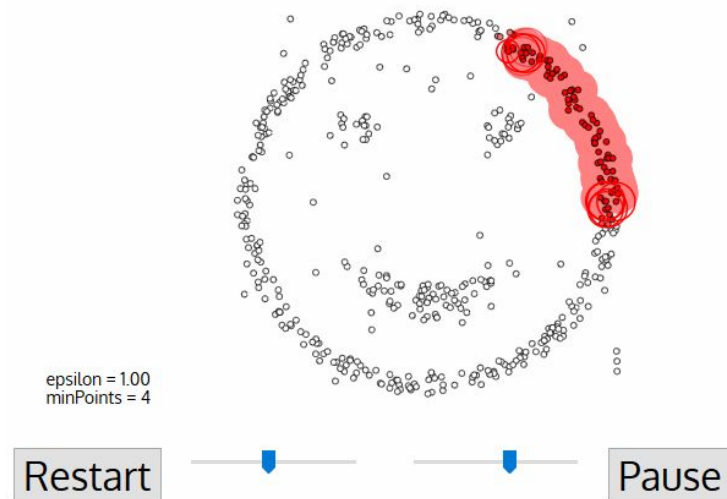


"The idea that a cluster in data space is a **contiguous region** of **high point density**, separated from other such clusters by contiguous regions of **low point density**."

[Model details with visual demonstration](#)

# DBSCAN

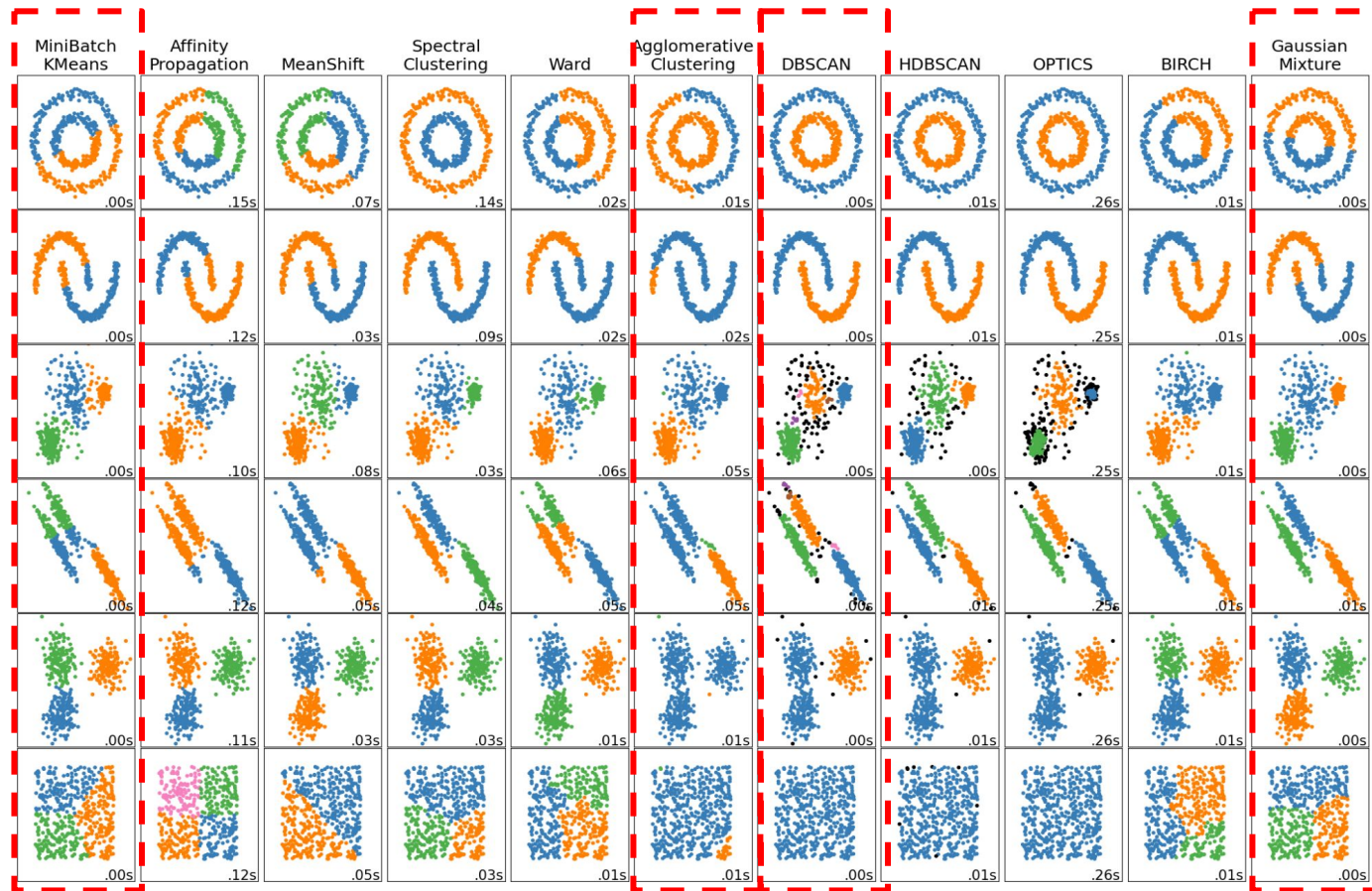
- Two parameters:
  - **minPts**: The minimum number of points (a threshold) clustered together to be considered dense.
  - **eps ( $\epsilon$ )**: A distance measure that will be used to locate the points in the neighborhood of any point.
- The algorithm proceeds by arbitrarily picking up a point in the dataset.
- If there are at least '**minPoint**' points within a **radius of ' $\epsilon$ '** to the point then we consider all these points to be part of the same cluster.
- The clusters are then expanded by recursively repeating the neighborhood calculation for each neighboring point



[Reference](#)



sklearn





# sklearn packages

[Clustering: sklearn](#)

[Clustering: sklearn \(API reference\)](#)