CIS 635 - Knowledge Discovery & Data Mining

Clustering Algorithms

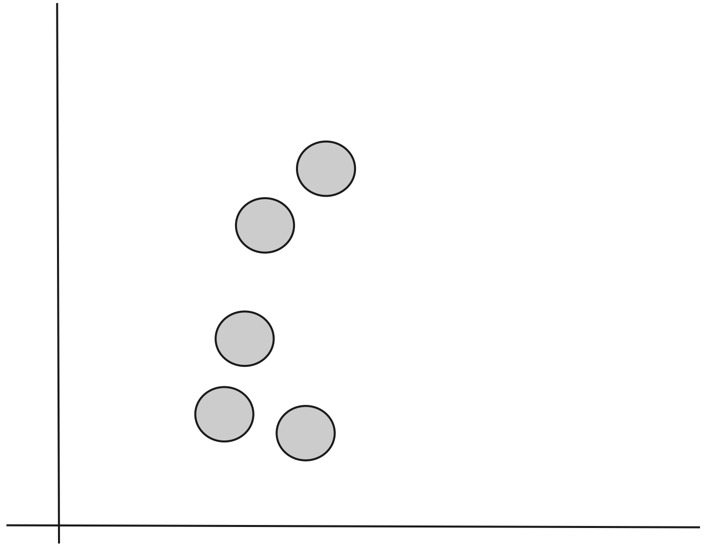
Clustering Algorithms

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

k-means Clustering

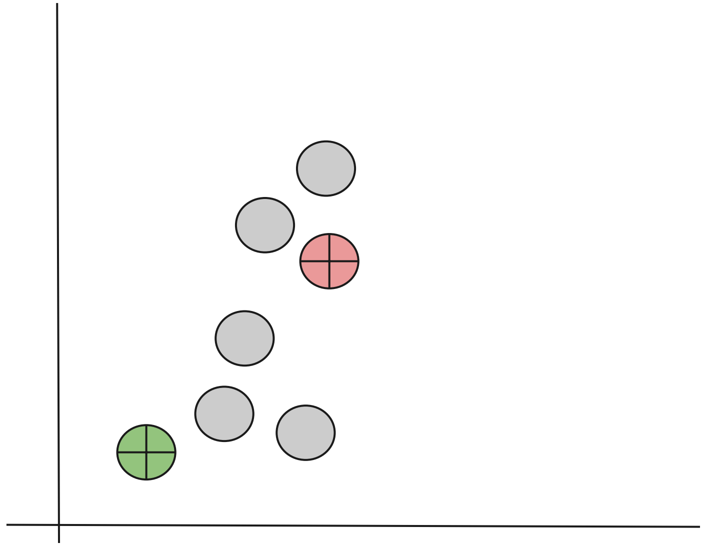
- Centroid based
- Only works for numeric data only
- Explicit k-centroid inputs (initialization)
- Iterative algorithm

k-means Clustering



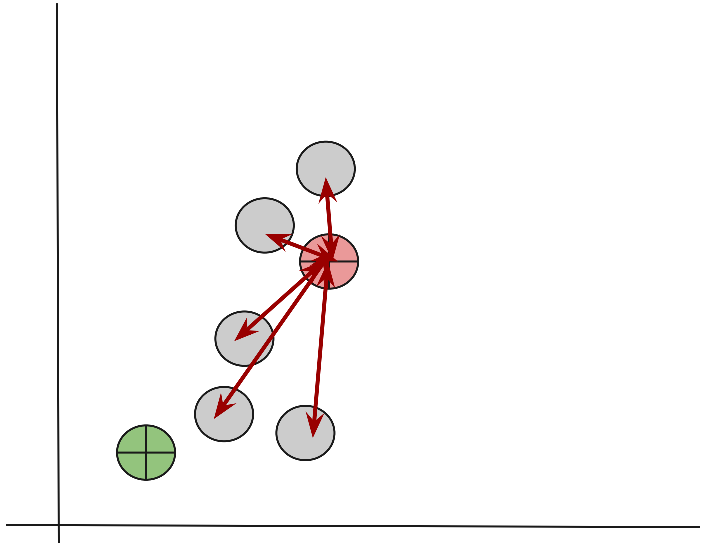
- 5 given data points of forms (x_i, y_i): numeric

k-means Clustering



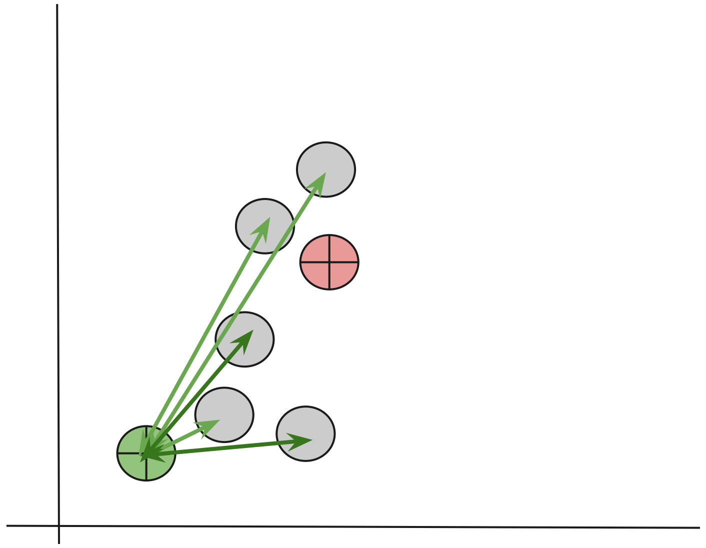
- 5 given data points of forms (x_i, y_i): numeric
- **For k = 2, the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means $(m_x, m_y)_j$ *where j = 1, .. k***

k-means Clustering



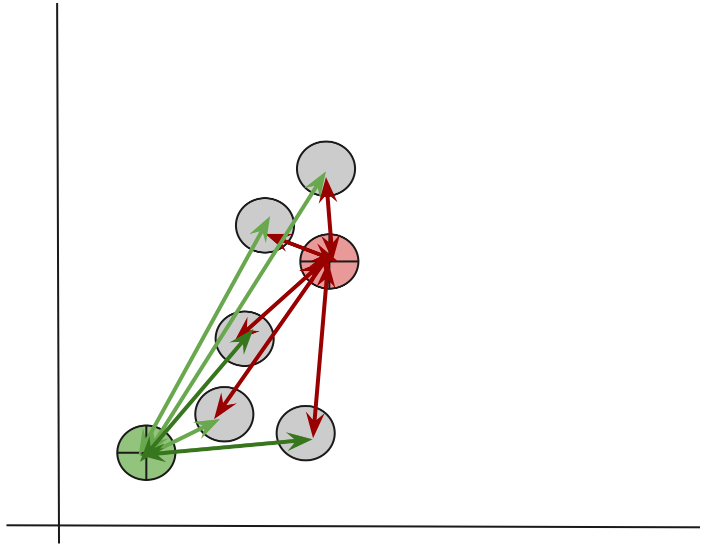
- 5 given data points of forms (x_i, y_i): numeric
- For k = 2, the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means $(m_x, m_y)_j$ *where* $j = 1, .. k$
- **Compares distance between the means $(m_x, m_y)_j$ and all given (5) data points and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

k-means Clustering



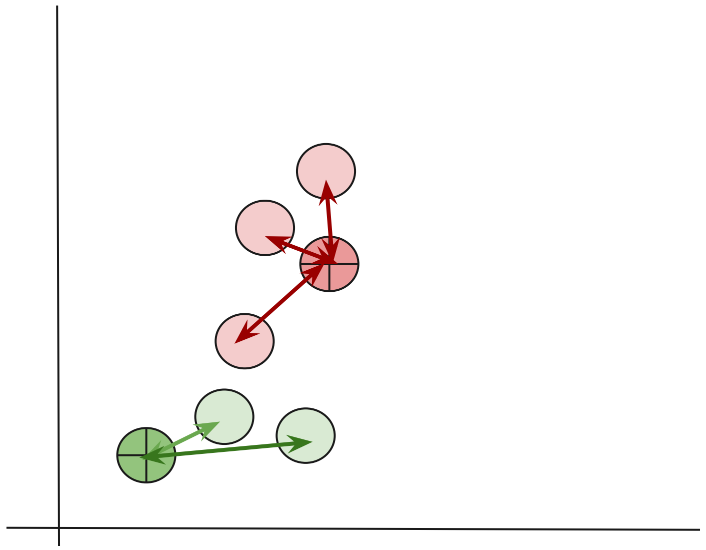
- 5 given data points of forms *(x_i, y_i)*: numeric
- For k = 2, the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means *(m_x, m_y)$_j$ where j = 1, .. k*
- **Compares distance between the means *(m_x, m_y)$_j$* and all given (5) data points and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

k-means Clustering



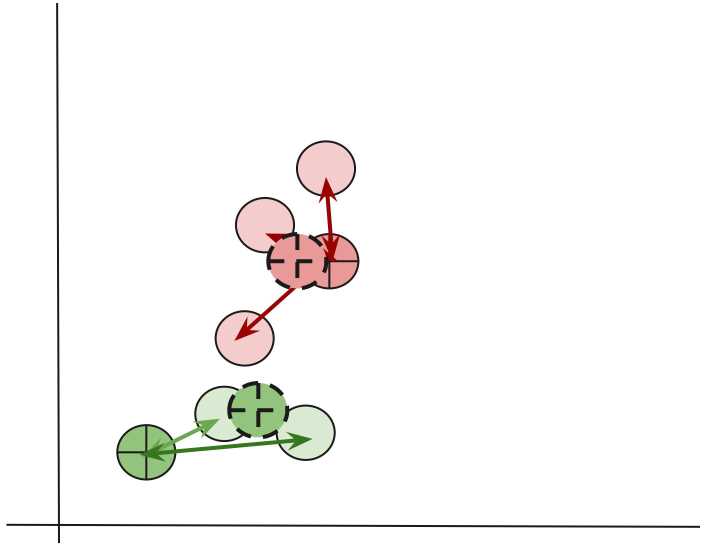
- 5 given data points of forms (x_i, y_i): numeric
- For k = 2, the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means $(m_x, m_y)_j$ *where* $j = 1, .. k$
- **Compares distance between the means $(m_x, m_y)_j$ and all given (5) data points and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

k-means Clustering



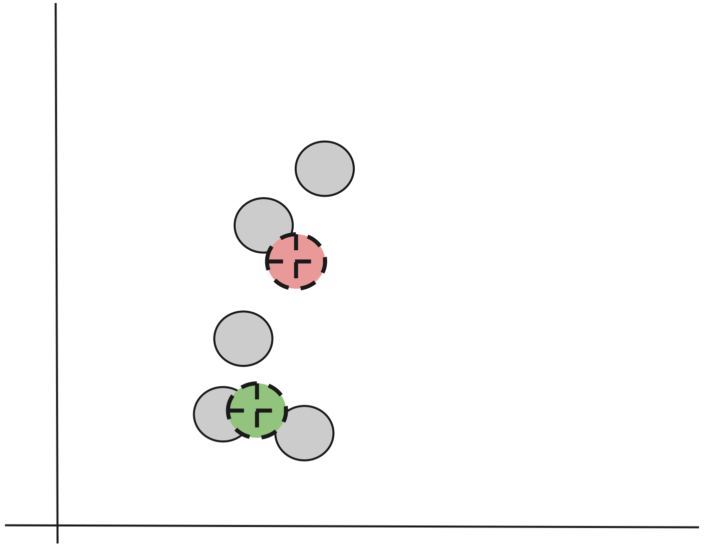
- 5 given data points of forms (x_i, y_i): numeric
- For k = 2, the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means $(m_x, m_y)_j$ *where* $j = 1, .. k$
- Compares distance between the means $(m_x, m_y)_j$ and all given (5) data points and for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

k-means Clustering



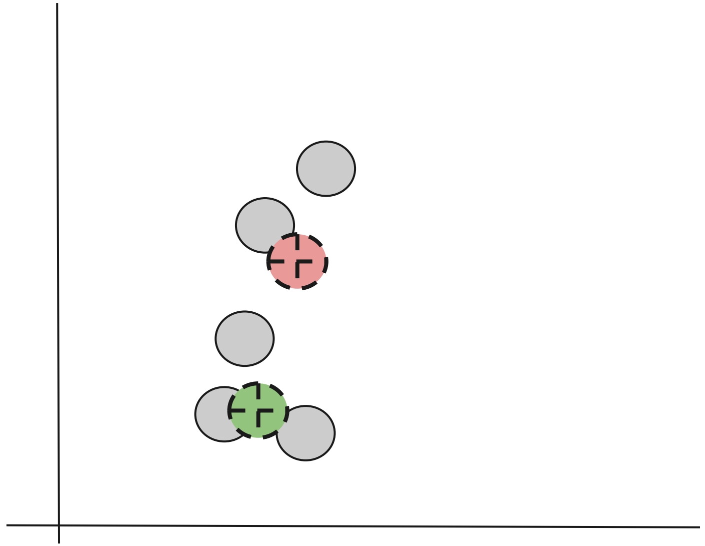
- 5 given data points of forms (x_i, y_i): numeric
- For k = 2, the algorithm starts with two random (users can also provide based on their analysis/intuition) initials means $(m_x, m_y)_j$ *where j = 1, .. k*
- Compares distance between the means $(m_x, m_y)_j$ and all given (5) data points and for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

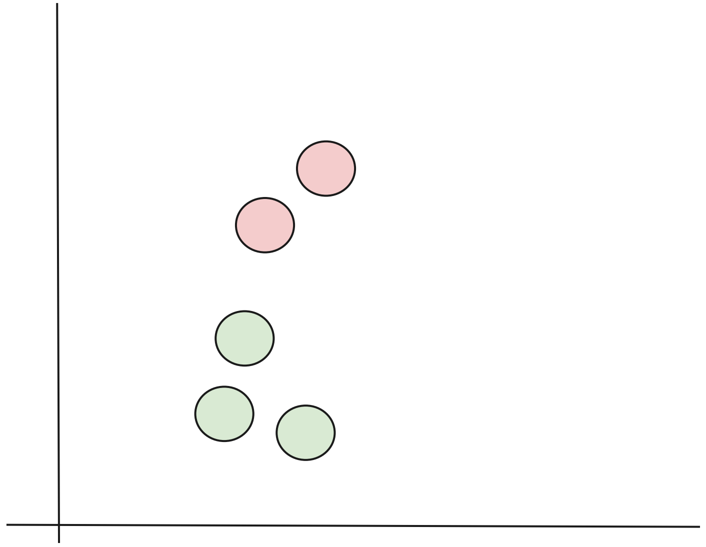
k-means Clustering



- 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, .. 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

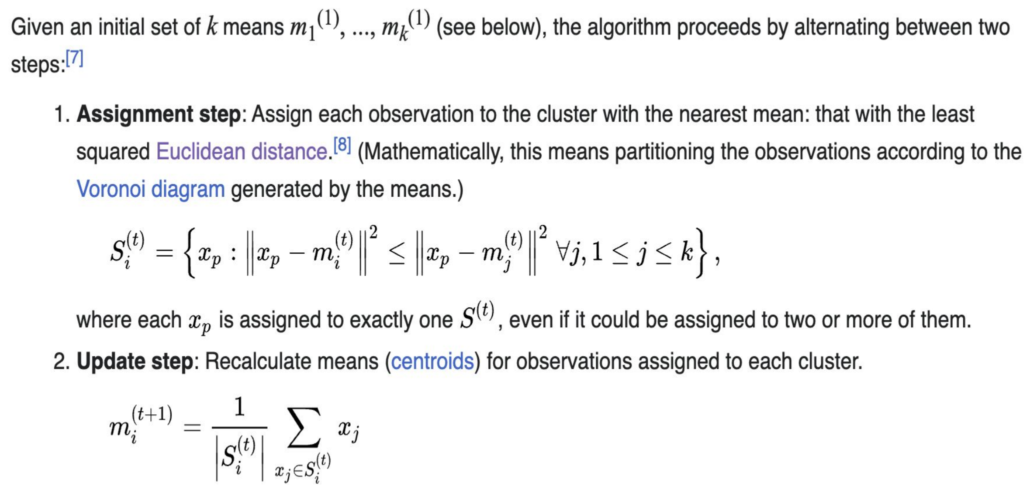
k-means Clustering



- 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, .. 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



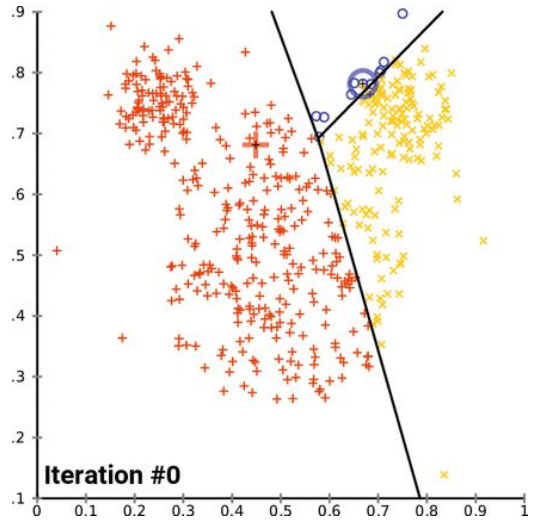
Given an initial set of k means $m_1{}^{(1)}$, ..., $m_k{}^{(1)}$ (see below), the algorithm proceeds by alternating between two steps:[7]

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

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

   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.

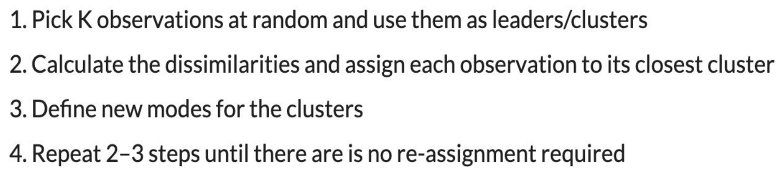
   $$m_i^{(t+1)} = \frac{1}{\left|S_i^{(t)}\right|} \sum_{x_j \in S_i^{(t)}} x_j$$



k-means [wiki]

k-modes is the categorical equivalent!

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

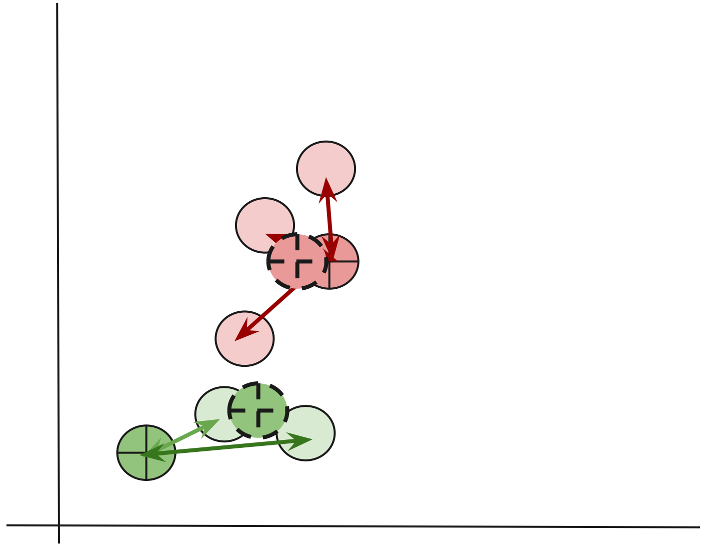


1. Pick K observations at random and use them as leaders/clusters
2. Calculate the dissimilarities and assign each observation to its closest cluster
3. Define new modes for the clusters
4. Repeat 2–3 steps until there are is no re-assignment required

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

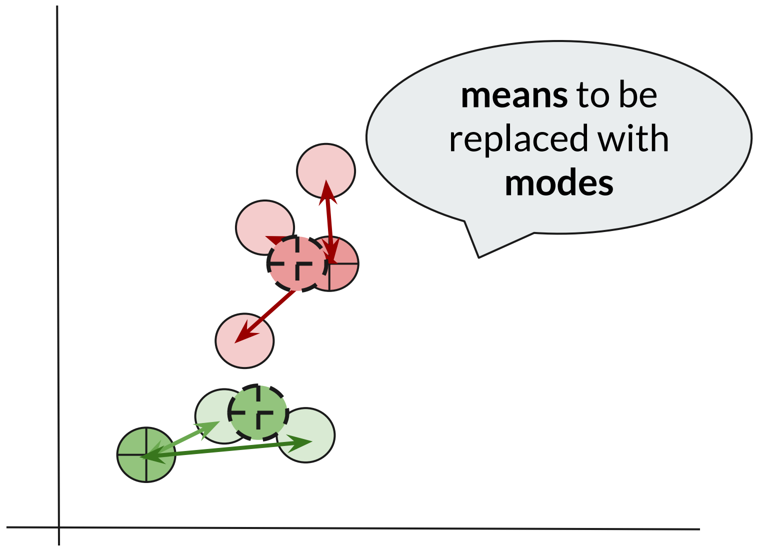
[k-modes]

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* $j = 1, .. k$
- 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:**
    - $m_x = 1/|x_i|\sum x_i$
    - $m_y = 1/|y_i|\sum y_i$

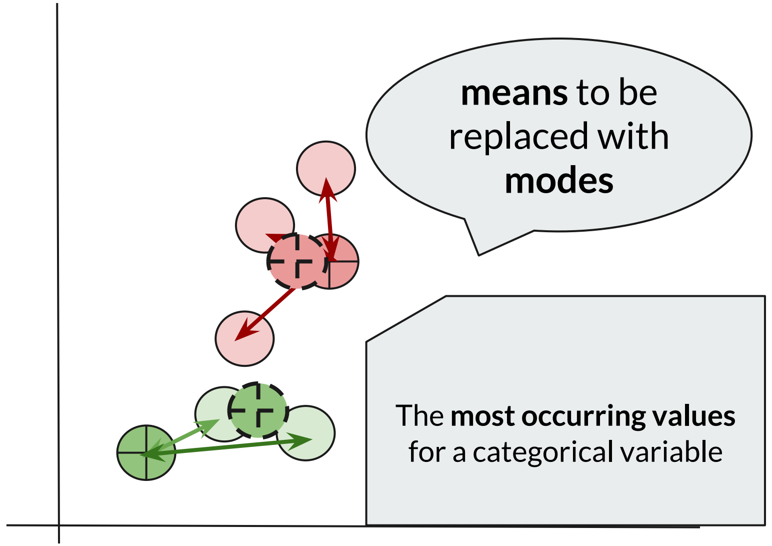
k-modes Clustering



- 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, .. 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:**
  - $m_x = 1/|x_i|$ **mode**(x_i)
  - $m_y = 1/|y_i|$ **mode**(x_i)

[k-modes]

k-modes Clustering


means to be replaced with **modes**
The **most occurring values** for a categorical variable

- 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, .. 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| mode(x_i)**
  - **m_y = 1/|y_i| mode(x_i)**

k-modes

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

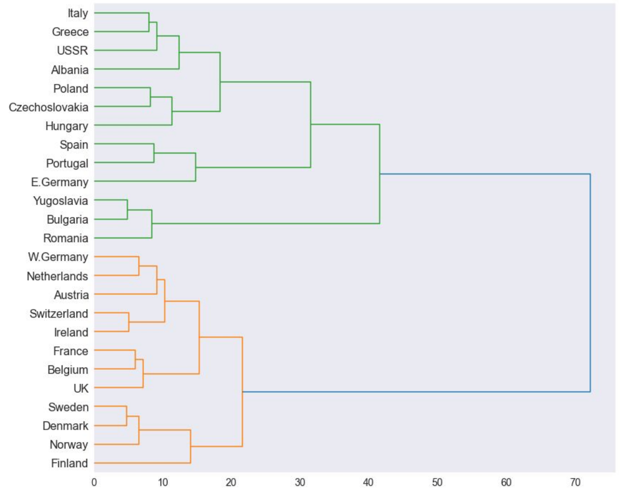
- ~~**k-means**: Centroid Based~~
- **Hierarchical clustering**: Distance connectivity based
- **GMM**: Distribution based
- **DBSCAN**: Density Based

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 on cluster, and recursively splits into two/multiple clusters.

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



Gaussian Mixture Model (GMM)

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

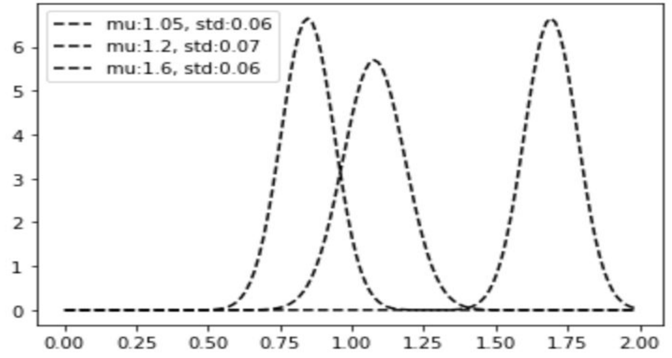
$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

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



Gaussian Mixture Model (GMM)

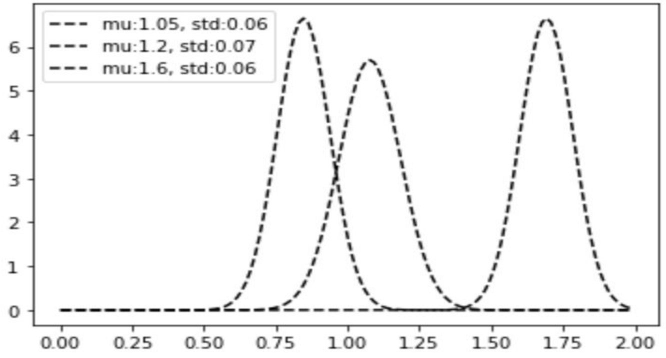
Basic idea from the model title itself:

- **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

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

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

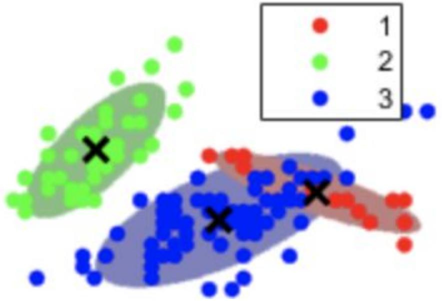


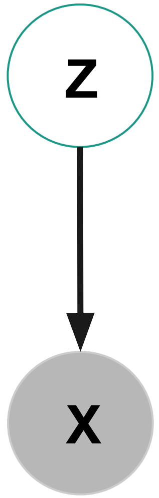
$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k,\, \sigma_k)$$

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

Gaussian Mixture Model (GMM)

- Fig at the right shows the graphical model (B Net) of GMM
- X: feature vector; in our example case a vector with apples (size, color)
- 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)





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

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

$$\boxed{0 \leqslant \pi_k \leqslant 1} \quad \boxed{\sum_{k=1}^{K} \pi_k = 1} \quad \boxed{p(\mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k}.}$$

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
- X: feature vector; in our example case a vector with apples (size, color)
- 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)

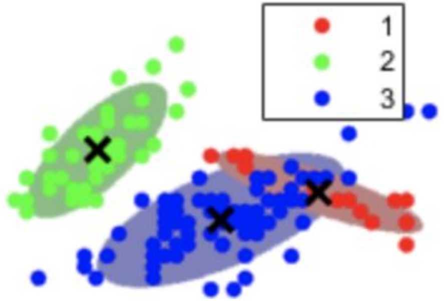
$$p(\mathbf{x}, \mathbf{z}) \;=\; p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$$

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

$$\boxed{0 \leqslant \pi_k \leqslant 1}$$

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

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

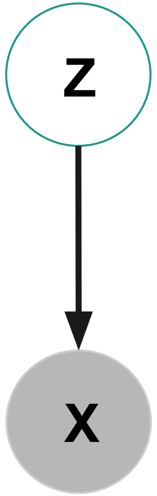


$$\begin{tikzcd} Z \arrow[d] \\ X \end{tikzcd}$$

Model parameters(all *k* s)

$$\pi, \mu, \Sigma$$

Gaussian Mixture Model (GMM)



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

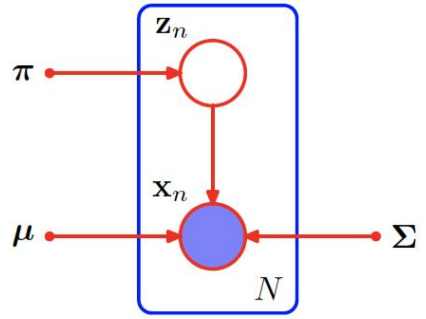
$$
\begin{aligned}
p(z_k = 1|\mathbf{x}) &= \frac{p(z_k = 1)p(\mathbf{x}|z_k = 1)}{\displaystyle\sum_{j=1}^{K} p(z_j = 1)p(\mathbf{x}|z_j = 1)} \\
&= \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\displaystyle\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j,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

Maximum Likelihood Learning

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

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

$$\log p(X) = \log(p(Z)p(X|Z))$$
$$= \log p(Z) + \log p(X|Z)$$

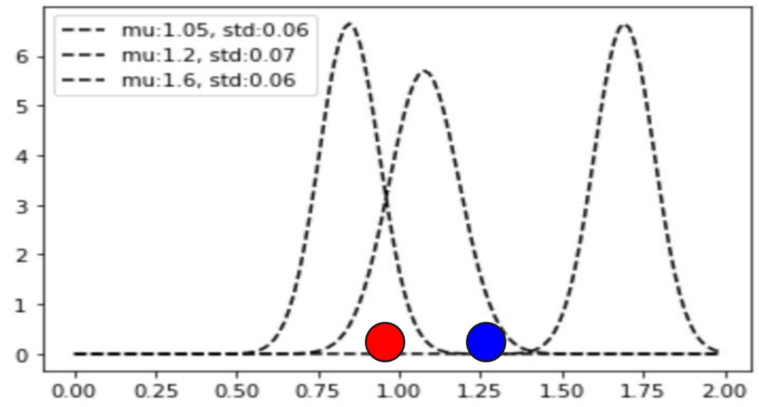


- Some popular techniques:
  - Expectation Maximization algorithm
  - We can also use gradient based optimization techniques

Why do we need GMM ?

- For clustering data points (like other methods - k-means, hierarchical clustering)
- Soft clustering: cluster assignment probability scores, ***p(Z|X)***
- It 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)***

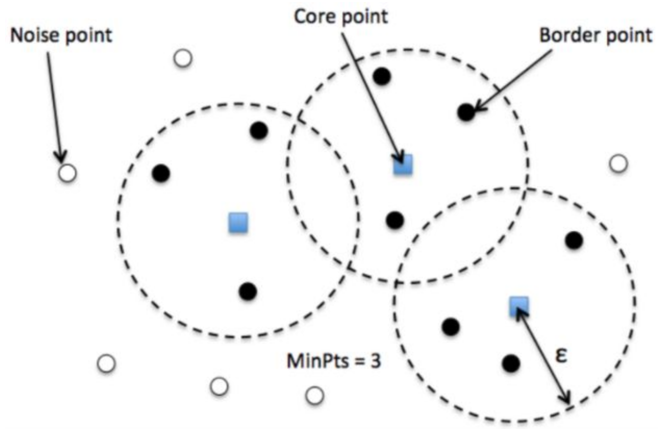
Gmm 3 components



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** (ε): 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 'ε'** 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

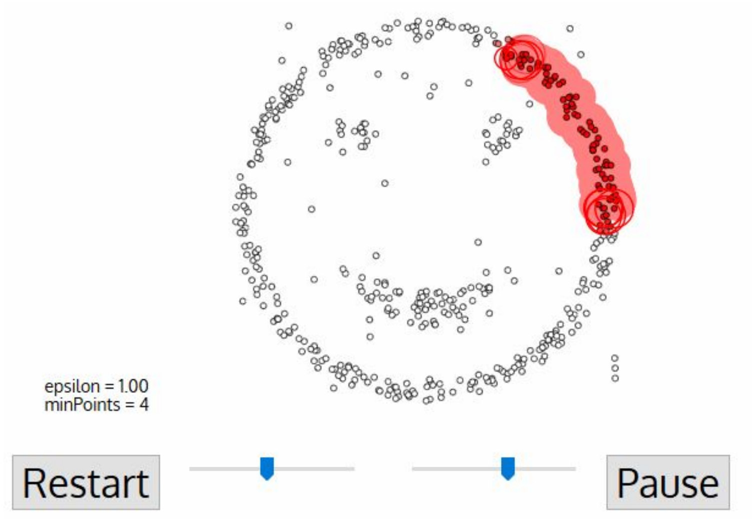
Model details with visual demonstration



"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**."

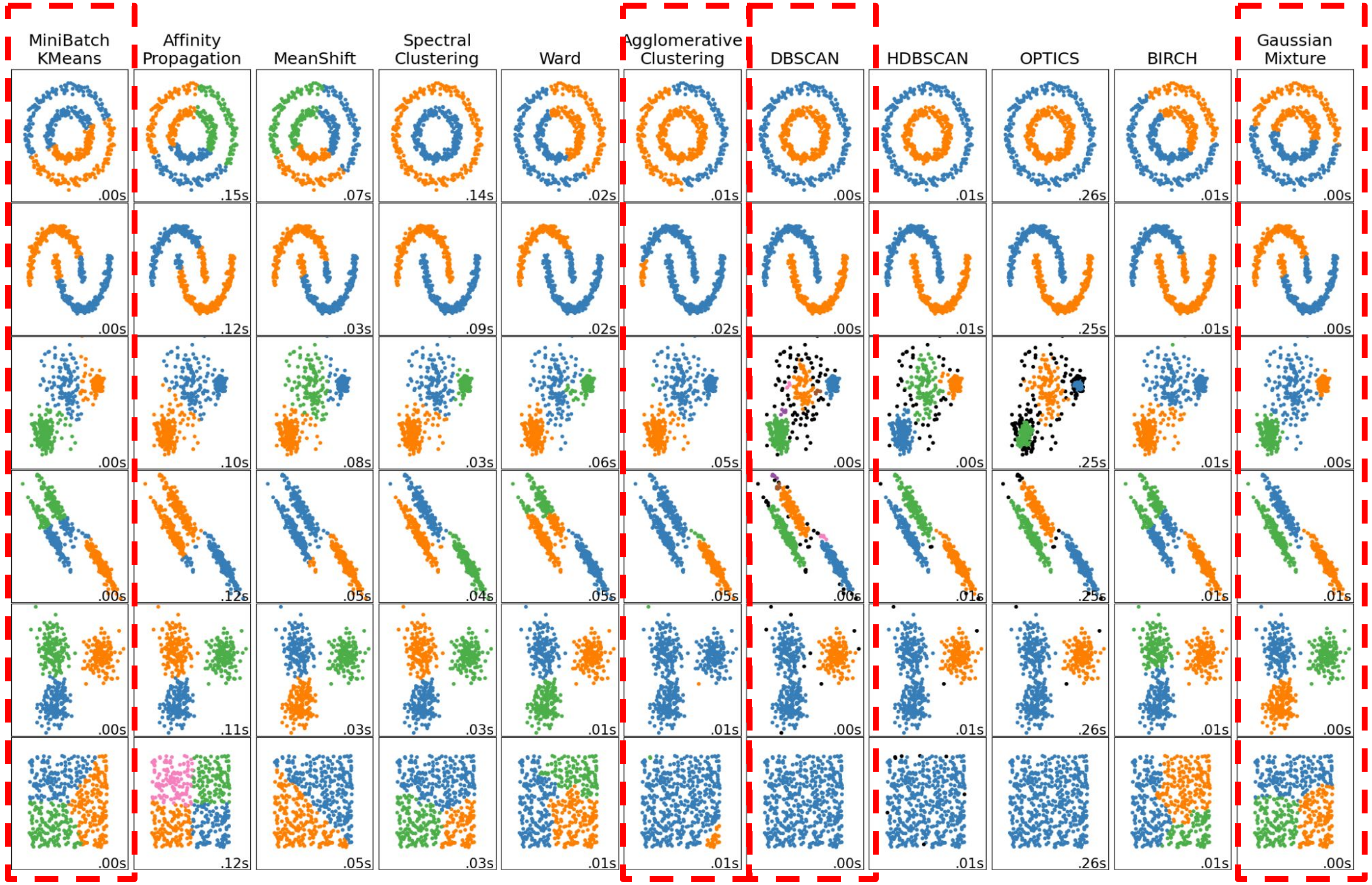
DBSCAN

- Two parameters:
  - minPts: The minimum number of points (a threshold) clustered together to be considered dense.
  - eps (ε): 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** ‘**ε**’ 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


epsilon = 1.00
minPoints = 4

Reference

sklearn



MiniBatch KMeans
Affinity Propagation
MeanShift
Spectral Clustering
Ward
Agglomerative Clustering
DBSCAN
HDBSCAN
OPTICS
BIRCH
Gaussian Mixture
.00s
.15s
.07s
.14s
.02s
.01s
.00s
.01s
.26s
.01s
.00s
.00s
.12s
.03s
.09s
.02s
.02s
.00s
.01s
.25s
.01s
.00s
.00s
.10s
.08s
.03s
.06s
.05s
.00s
.00s
.25s
.01s
.00s
.00s
.12s
.05s
.04s
.05s
.05s
.00s
.01s
.25s
.01s
.01s
.00s
.11s
.03s
.03s
.01s
.01s
.00s
.01s
.26s
.01s
.00s
.00s
.12s
.05s
.03s
.01s
.01s
.00s
.01s
.26s
.01s
.00s

sklearn packages

Clustering: sklearn

Clustering: sklearn (API reference)