# Clustering: K-means vs GMM

Liang Liang

# K-means Clustering

- $N$ data points $\{x_1, \dots, x_N\}$, $x_n \in \mathcal{R}^M$
- Find $K$ cluster centers $\{\mu_1, \dots, \mu_K\} \in \mathcal{R}^M$ such that the loss is minimized

$$L = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} \alpha_{(n,k)} \|x_n - \mu_k\|^2$$

$\alpha_{(n,k)} = 1$ if and only if $x_n$ is assigned to the cluster-k

a $N$-by-$K$ assignment matrix $\begin{bmatrix} \alpha_{(1,1)} & \cdots & \alpha_{(1,K)} \\ \cdots & \cdots & \cdots \\ \alpha_{(N,1)} & \cdots & \alpha_{(N,K)} \end{bmatrix}$, $\alpha_{(n,k)} = 0 \ or \ 1$

$\sum_{k=1}^{K} \alpha_{(n,k)} = 1$ because $x_n$ is assigned to only one cluster

# The k-means algorithm

- Initialization: the user inputs $K$, and the algorithm initializes random centers $\{\mu_1, \dots, \mu_K\}$, $\mu_k \in \mathcal{R}^M$
- In each iteration:
  - step-1: assign each data point $x_n$ to its nearest cluster, *we* get $\alpha_{(n,k)}$

$$cluster\_label(x_n) = arg \min_{k \in \{1,\dots,K\}} \|x_n - \mu_k\|^2$$

  - step-2: move center $\mu_k$ to the average location of data points in cluster-k

$$\mu_k = \frac{1}{N_k} \sum x_n \; (x_n \; in \; cluster\_k)$$

where $N_k$ is the number of data points in the cluster-k

# The k-means algorithm

- step-1: assign each data point $x_n$ to its nearest cluster, $we$ get $\alpha_{(n,k)}$

$$cluster\_label(x_n) = arg \min_{k \in \{1,\ldots,K\}} \|x_n - \mu_k\|^2$$

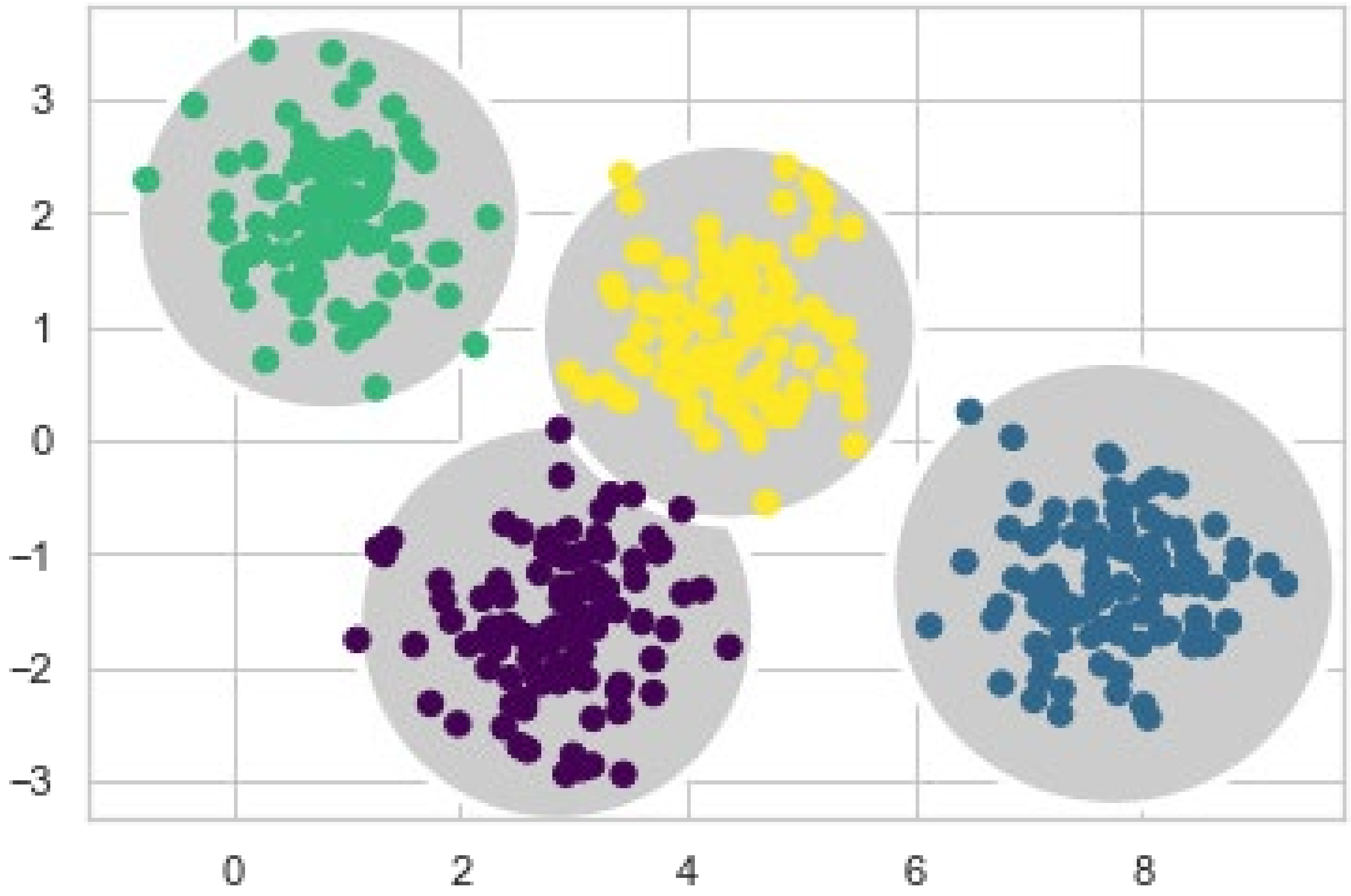
Example: Let's assume there are three clusters (K=3)

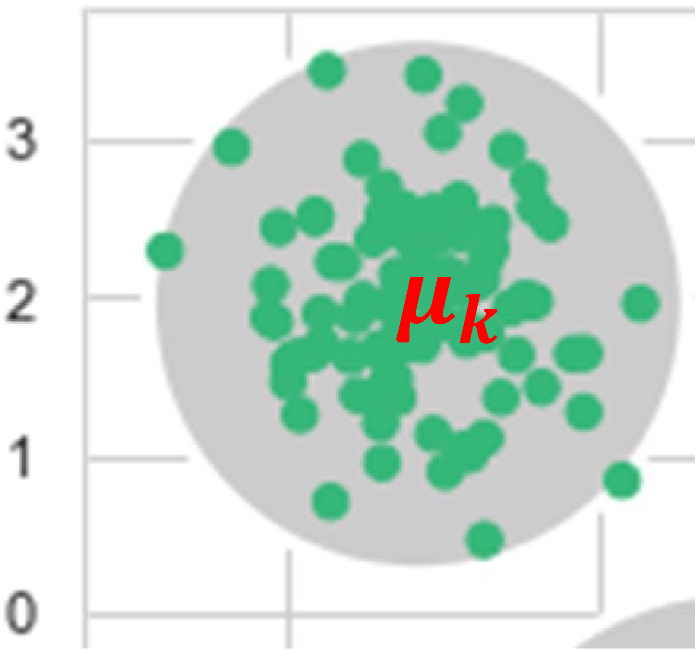
| Data Point | *Cluster 1* | *Cluster 2* | *Cluster 3* |
|---|---|---|---|
| $x_1$ | $\alpha_{(1,1)} = 1$ | $\alpha_{(1,2)} = 0$ | $\alpha_{(1,3)} = 0$ |
| $x_2$ | $\alpha_{(2,1)} = 0$ | $\alpha_{(2,2)} = 1$ | $\alpha_{(2,3)} = 0$ |
| $x_3$ | $\alpha_{(3,1)} = 0$ | $\alpha_{(3,2)} = 1$ | $\alpha_{(3,3)} = 0$ |

$x_1$ is assigned to cluster 1
$x_2$ and $x_3$ are assigned to cluster 2

**"hard" assignment matrix** $[\alpha_{(n,k)}]$

# Visualize 2D clusters from K-means






The smallest circle that contains all of the points in the **cluster-k**

The radius of the circle

$$r = max\{d(x_1, \mu_k), d(x_2, \mu_k), \dots, d(x_{N_k}, \mu_k)\}$$

$x_1, x_2, \dots, x_{N_k}$ are in cluster-k

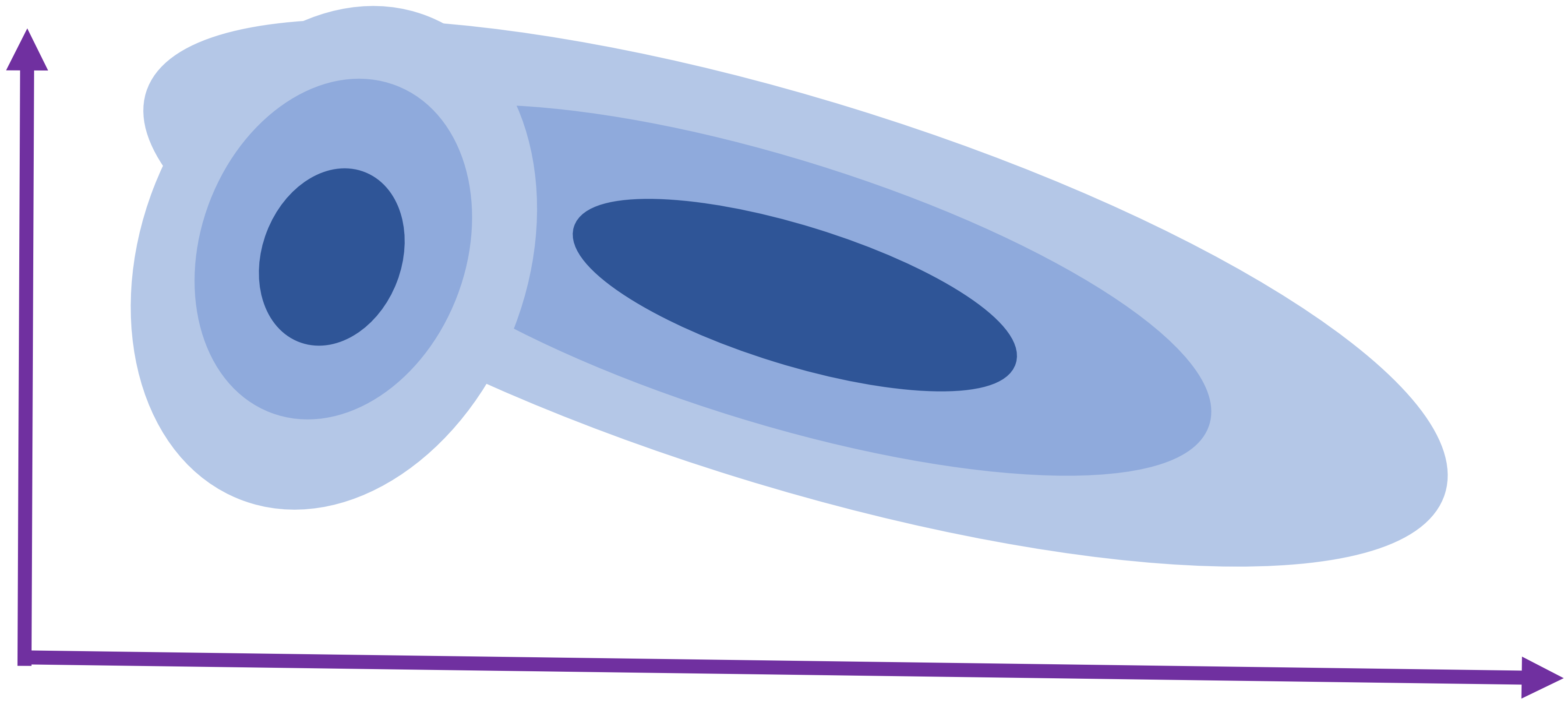
# Gaussian Mixture Model (GMM)

- Mixture of $K$ Gaussians in *M-dim*: $x, \mu \in \mathcal{R}^M$, $\Sigma \in \mathcal{R}^{M\times M}$

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

- $\pi_k = P(\{X \in \ cluster_k\})$, *prior probability,* and $\sum_{k=1}^{K} \pi_k = 1$
- $\pi_k$ is called the weight of a Gaussian component in sk-learn

# Visualize a 2D GMM that has 2 Gaussians



# GMM for data distribution (PDF) modeling and clustering

- **Input**: data samples

  $K$, the number of Gaussian components/clusters

- **Initialization**: initial values of $\mu_k, \Sigma_k, \pi_k$ for $k$=1 to $K$

two methods for initialization (available in sk-learn):

(1) random

(2) use k-means to get $K$ clusters

for cluster-k, compute $\mu_k, \Sigma_k$ only use $\{x_n\}$ in cluster-k

$$\mu_k = \frac{1}{N_k} \sum x_n$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N_k} (x_n - \mu_k)(x_n - \mu_k)^T$$

## GMM for data distribution (PDF) modeling and clustering

- **E-step**: compute $\gamma_{(n,k)}$, probability of $x_n$ belonging to cluster-**k**

$$\gamma_{(n,\boldsymbol{k})} = \frac{\boldsymbol{\pi_k} \mathcal{N}(\boldsymbol{x_n} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}{\sum_{j=1}^{K} \boldsymbol{\pi_j} \mathcal{N}(\boldsymbol{x_n} | \boldsymbol{\mu_j}, \boldsymbol{\Sigma_j})}$$

where $\sum_{k=1}^{K} \gamma_{(n,k)} = 1$

the soft assignment matrix $[\gamma_{(n,k)}]$ can be used for clustering

# GMM for data distribution (PDF) modeling and clustering

- **E-step**: compute $\gamma_{(n,k)}$, the probability of $x_n$ belonging to cluster-k

$$\gamma_{(n,k)} = \frac{\pi_k \mathcal{N}(x_n|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n|\mu_j,\Sigma_j)} \text{ where } \sum_{k=1}^{K} \gamma_{(n,k)} = 1$$

the soft assignment matrix $[\gamma_{(n,k)}]$ can be used for clustering

Example: Let's assume there are three Gaussians in a GMM

| **Data Point** | *Cluster 1* | *Cluster 2* | *Cluster 3* |
|---|---|---|---|
| $x_1$ | $\gamma_{(1,1)} = 0.9$ | $\gamma_{(1,2)} = 0.06$ | $\gamma_{(1,3)} = 0.04$ |
| $x_2$ | $\gamma_{(2,1)} = 0.03$ | $\gamma_{(2,2)} = 0.9$ | $\gamma_{(2,3)} = 0.07$ |
| $x_3$ | $\gamma_{(3,1)} = 0.5$ | $\gamma_{(3,2)} = 0.2$ | $\gamma_{(3,3)} = 0.3$ |

$x_1$ and $x_3$ are assigned to cluster 1

$x_2$ is assigned to cluster 2

**soft assignment matrix** $[\gamma_{(n,k)}]$

# GMM for data distribution (PDF) modeling and clustering

After E-step, we obtain the soft assignment matrix $\left[\gamma_{(n,k)}\right]$

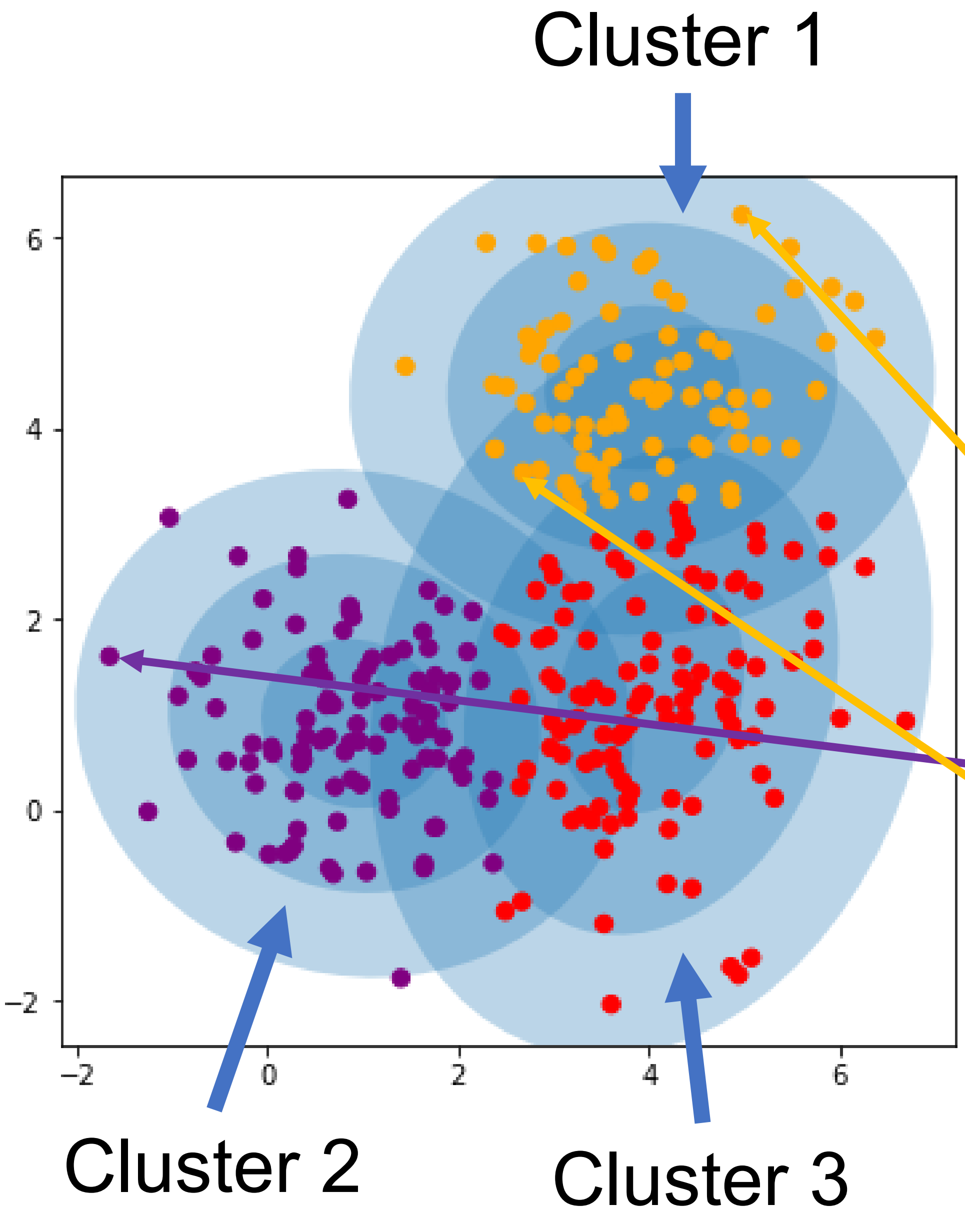
- **M-step**: update the parameters $\mu_k, \Sigma_k, \pi_k$ for $k$=1 to $K$

$$\mu_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma_{(n,k)}x_n$$ weighted average of **all** data points

$$\Sigma_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma_{(n,k)}(x_n - \mu_k)(x_n - \mu_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

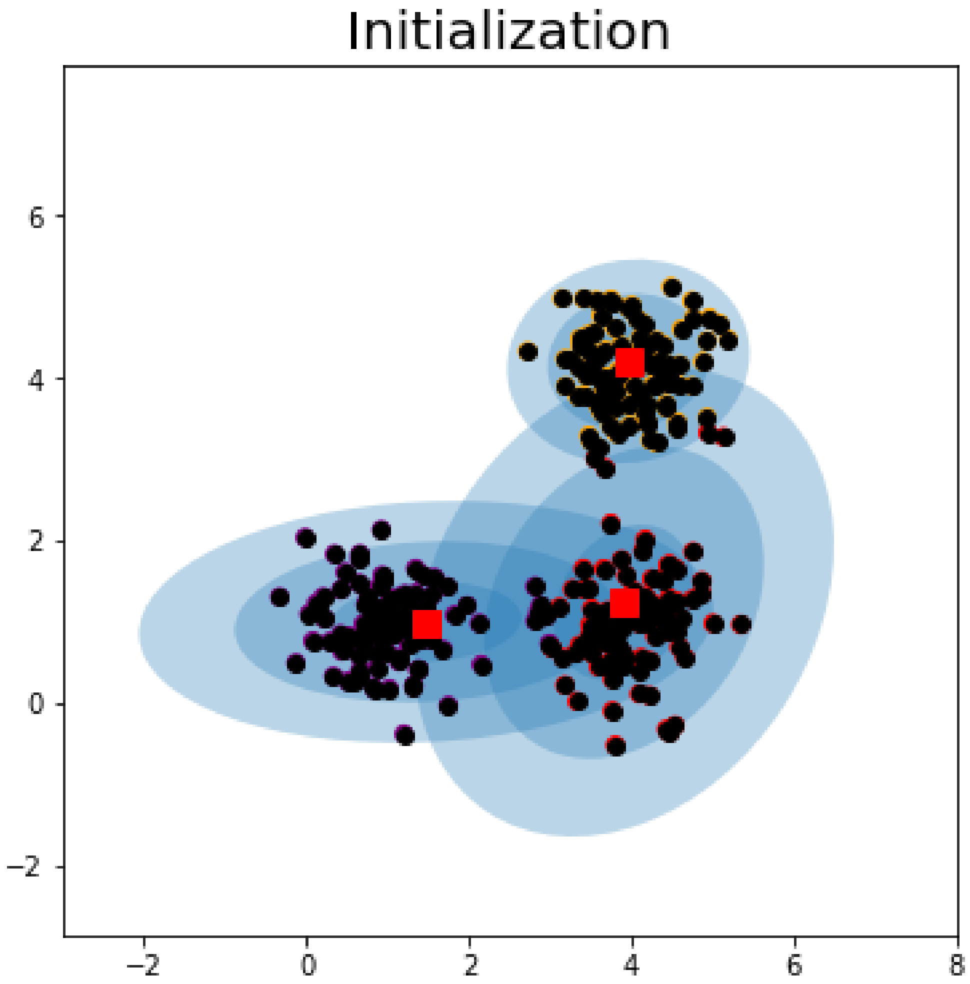
$$N_k = \sum_{n=1}^{N}\gamma_{(n,k)}$$



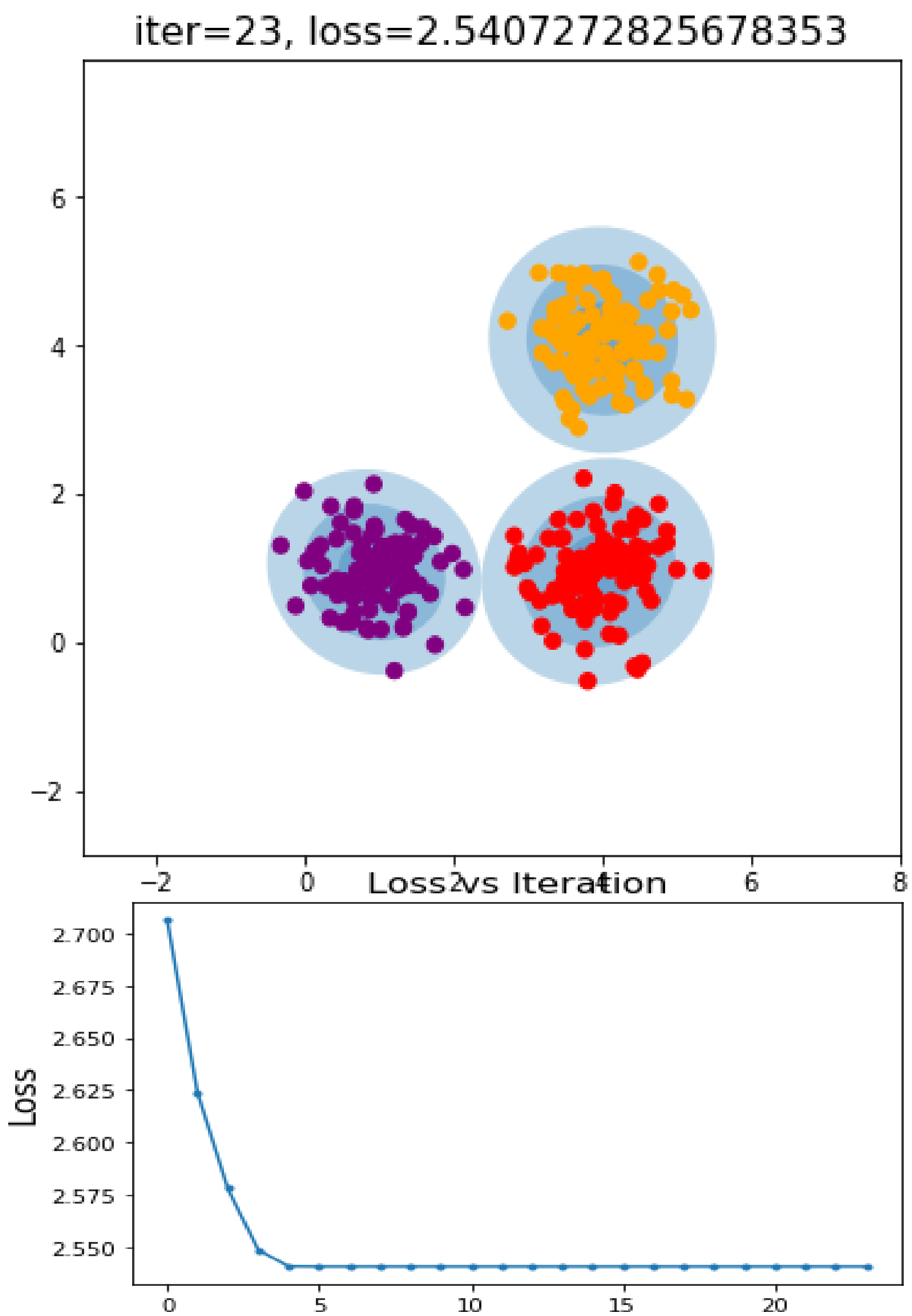

| Data Point | *Cluster* 1 | *Cluster* 2 | *Cluster* 3 |
|---|---|---|---|
| $x_1$ | 0.9 | 0.06 | 0.04 |
| $x_2$ | 0.03 | 0.9 | 0.07 |
| $x_3$ | 0.5 | 0.2 | 0.3 |

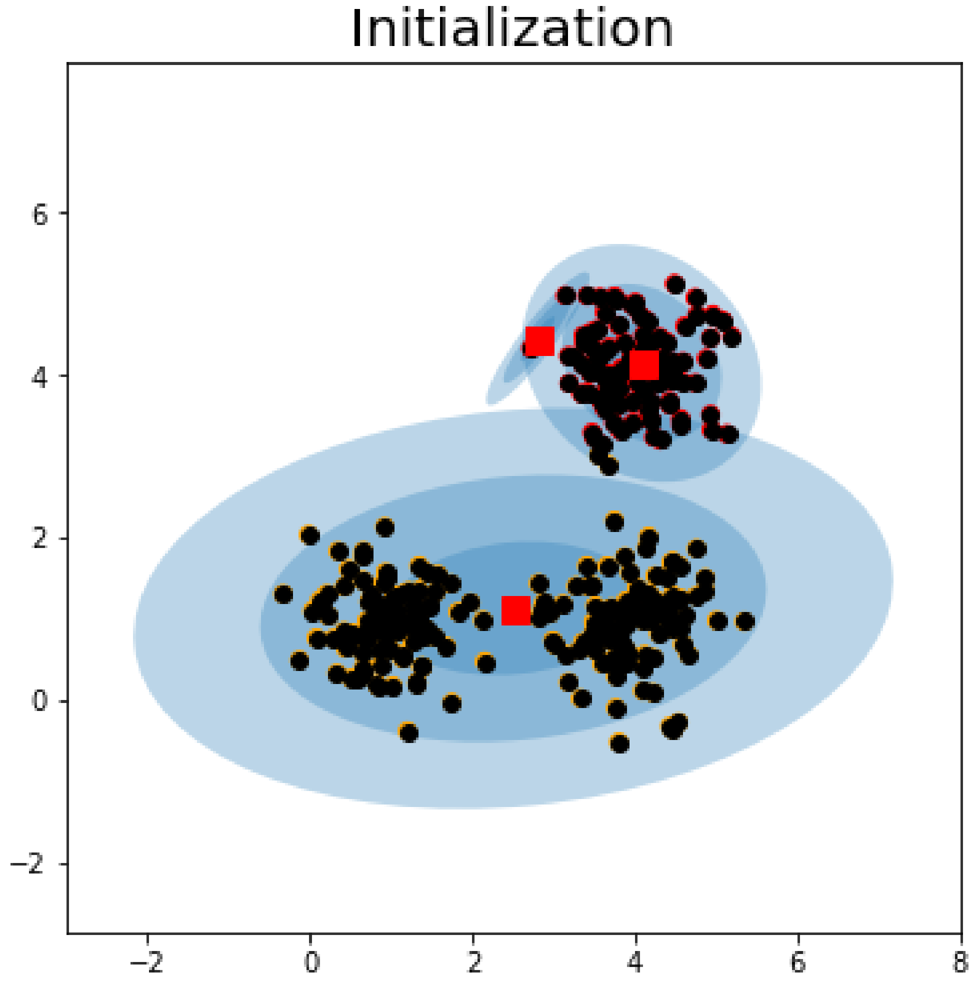
$\gamma_{(n,k)}$ is also called membership of $x_n$ in cluster-k

See Kmeans_2D_sim.ipynb
GMM_2D_sim.ipynb
GMM_DE_Generative_Model.ipynb

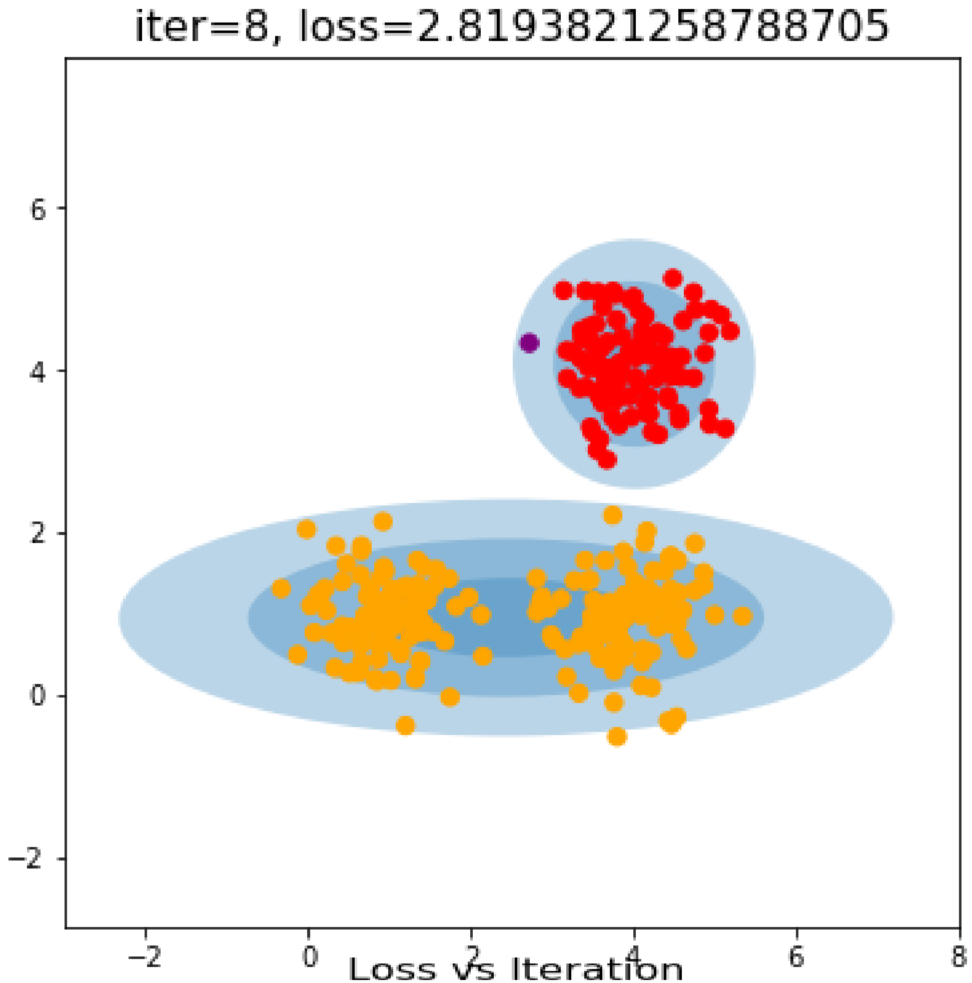


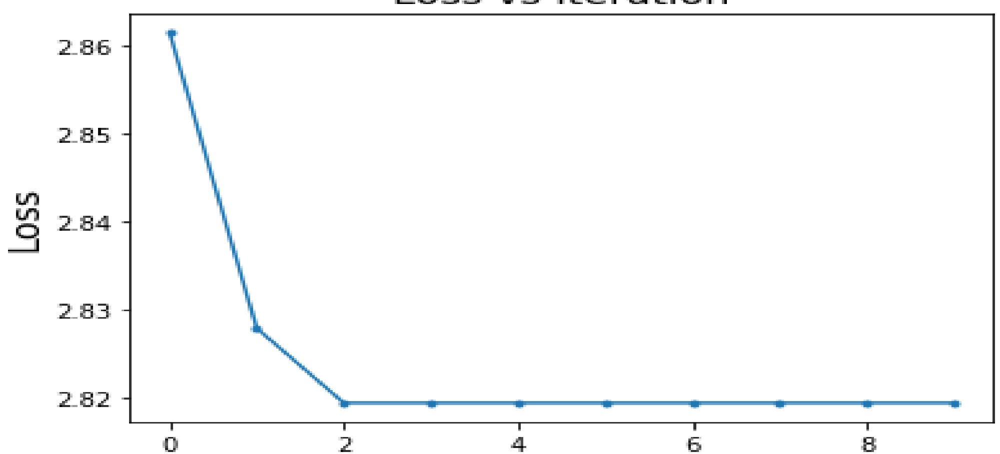

Random initialization

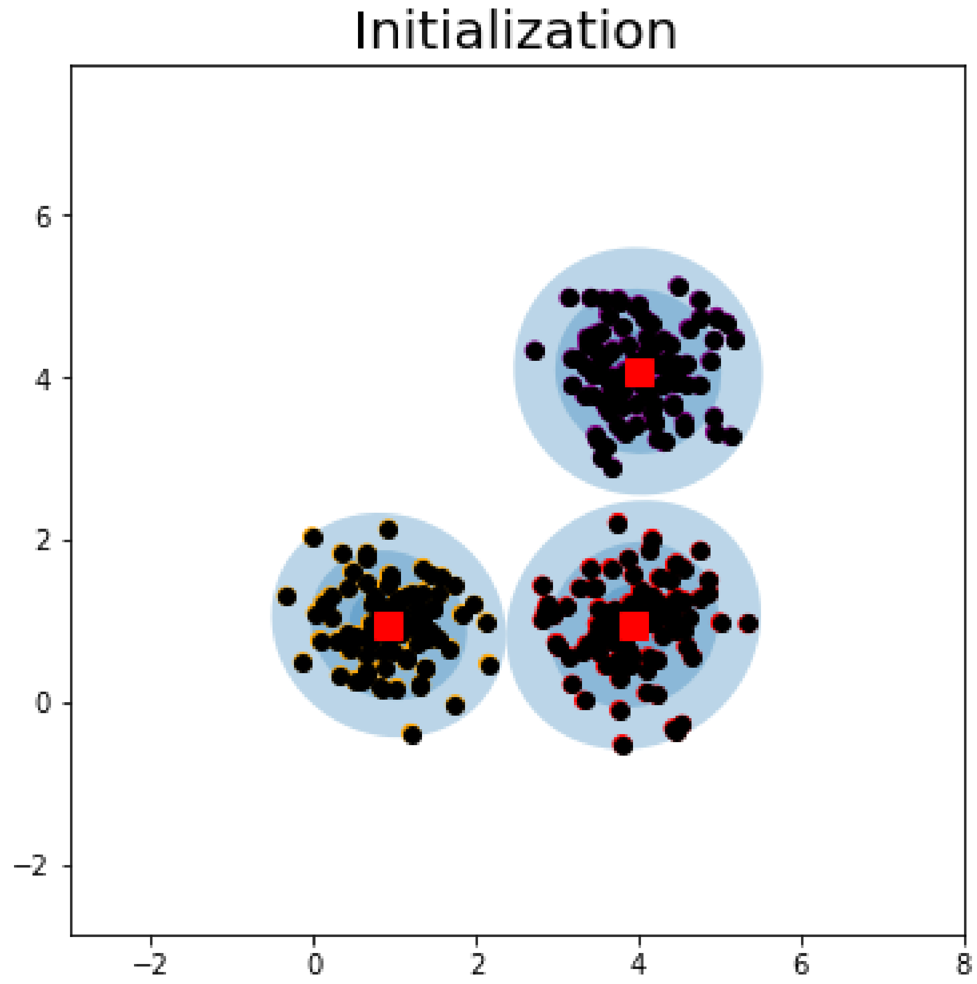


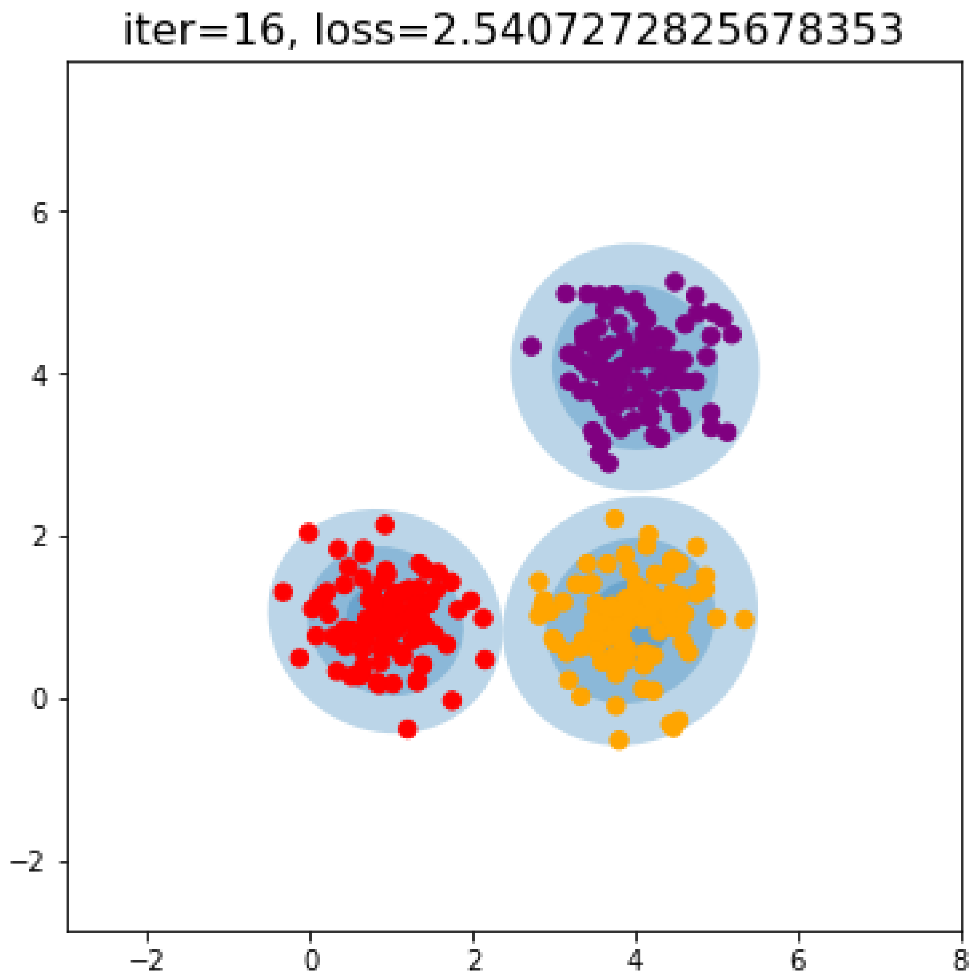



Random initialization

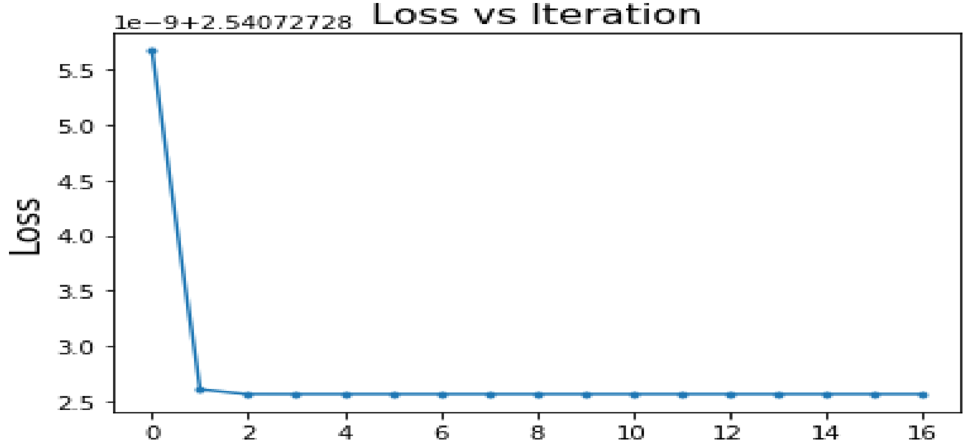


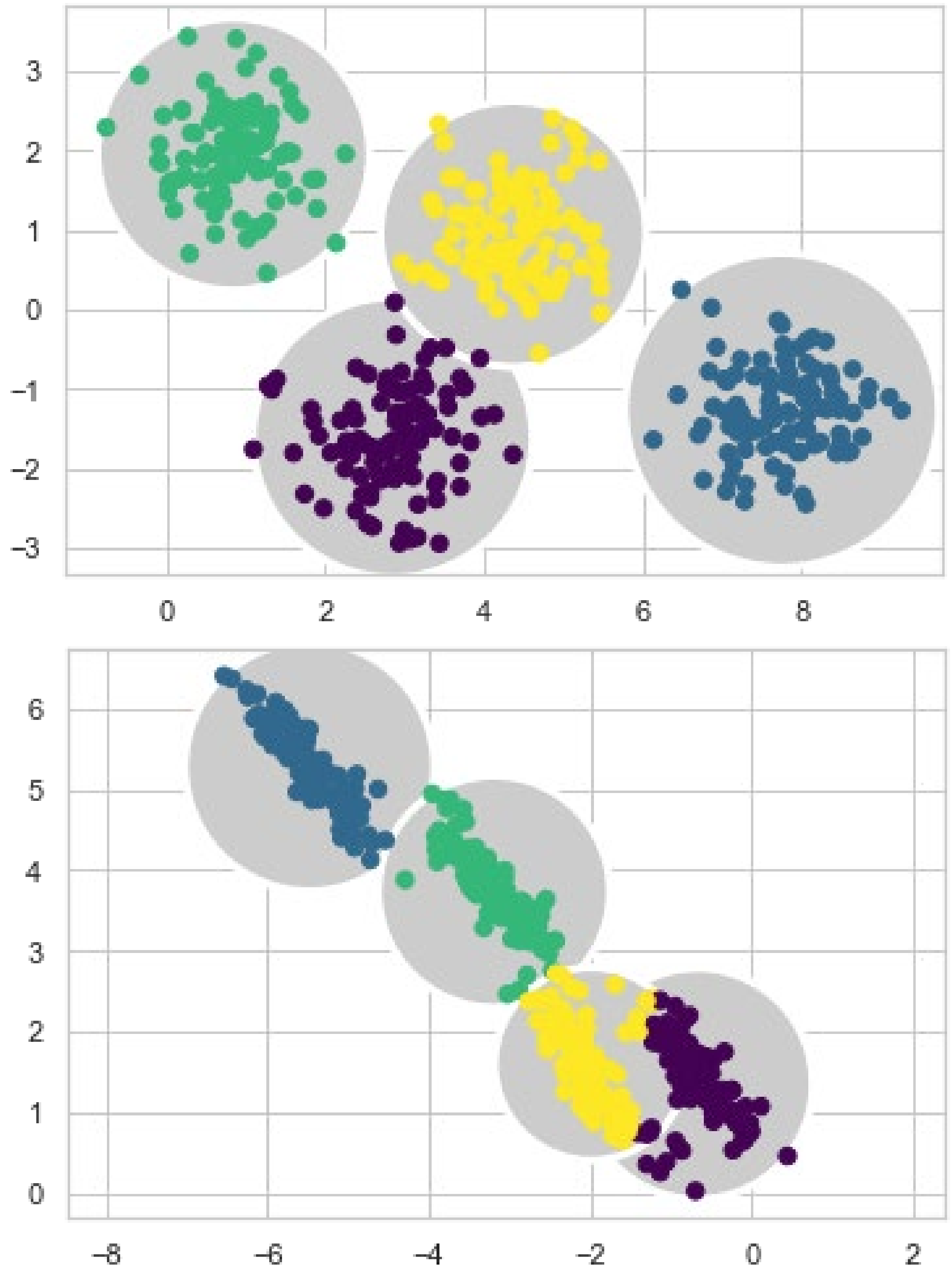


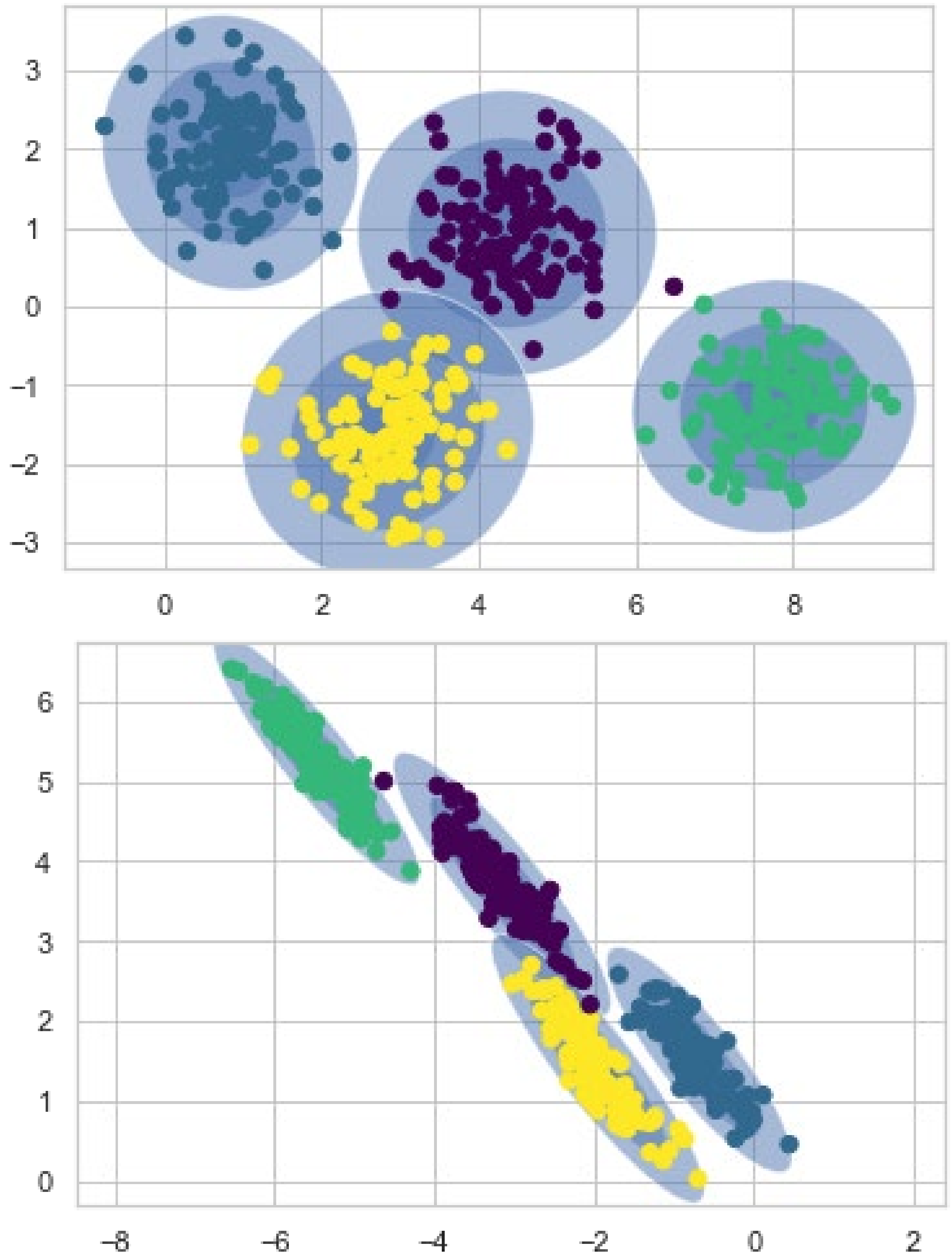

kmeans-based initialization



# K-means Clustering




# GMM Clustering



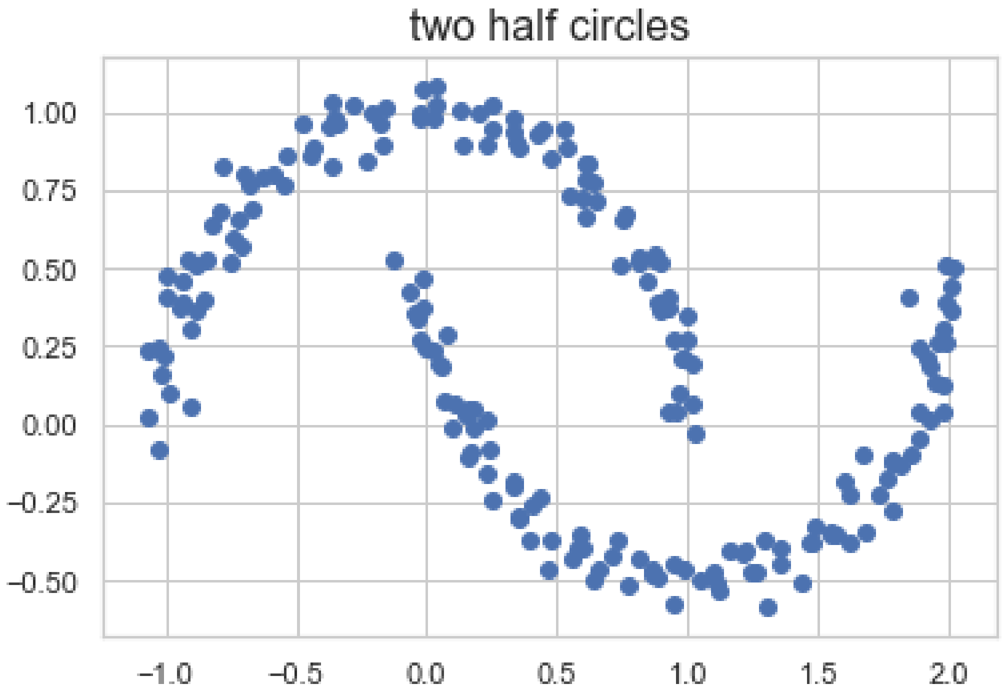
GMM is a probability density function (PDF)

to model data distribution

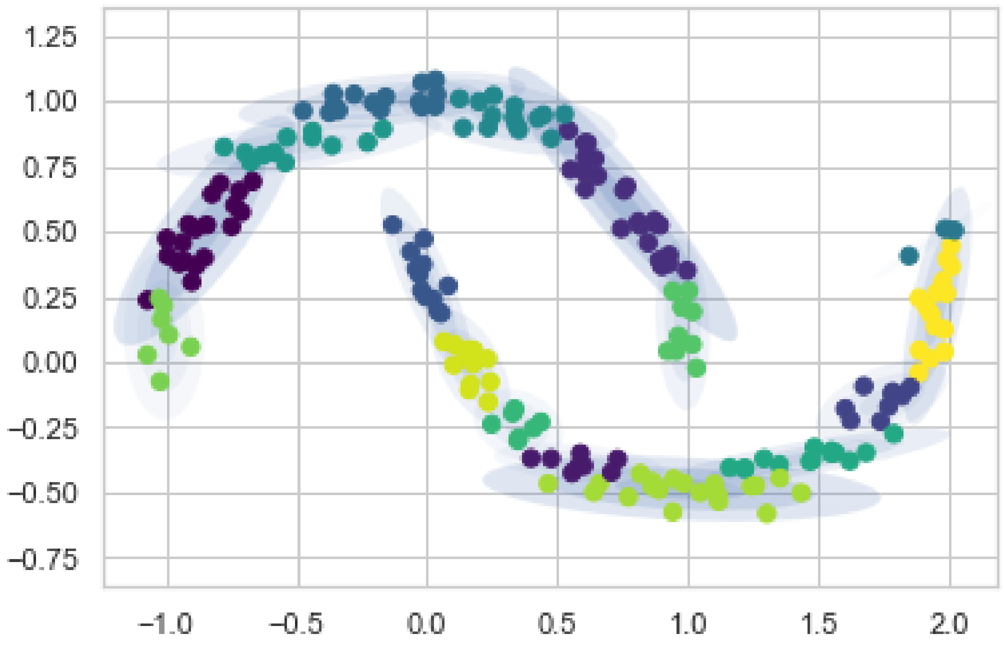
to estimate PDF of data

assuming data samples come from some unknown PDF

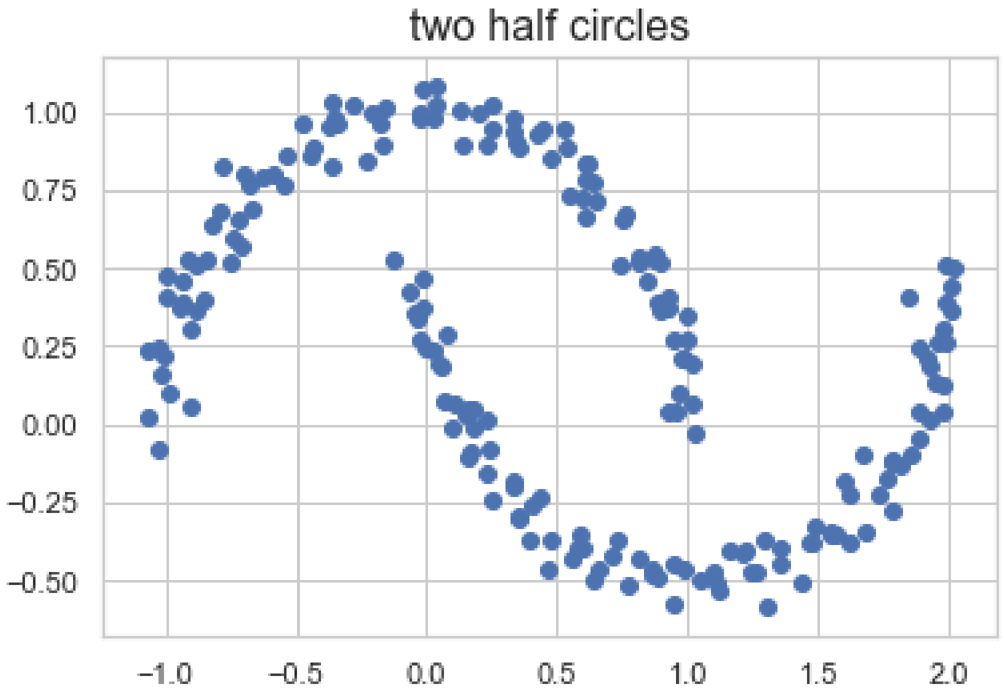
# Input Data



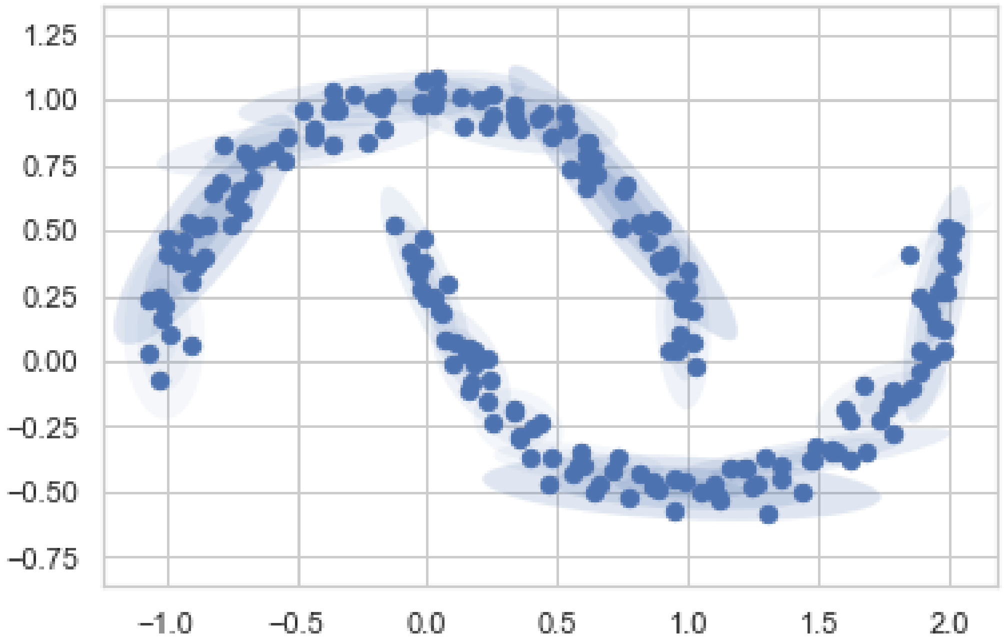

# Use a GMM to do clustering

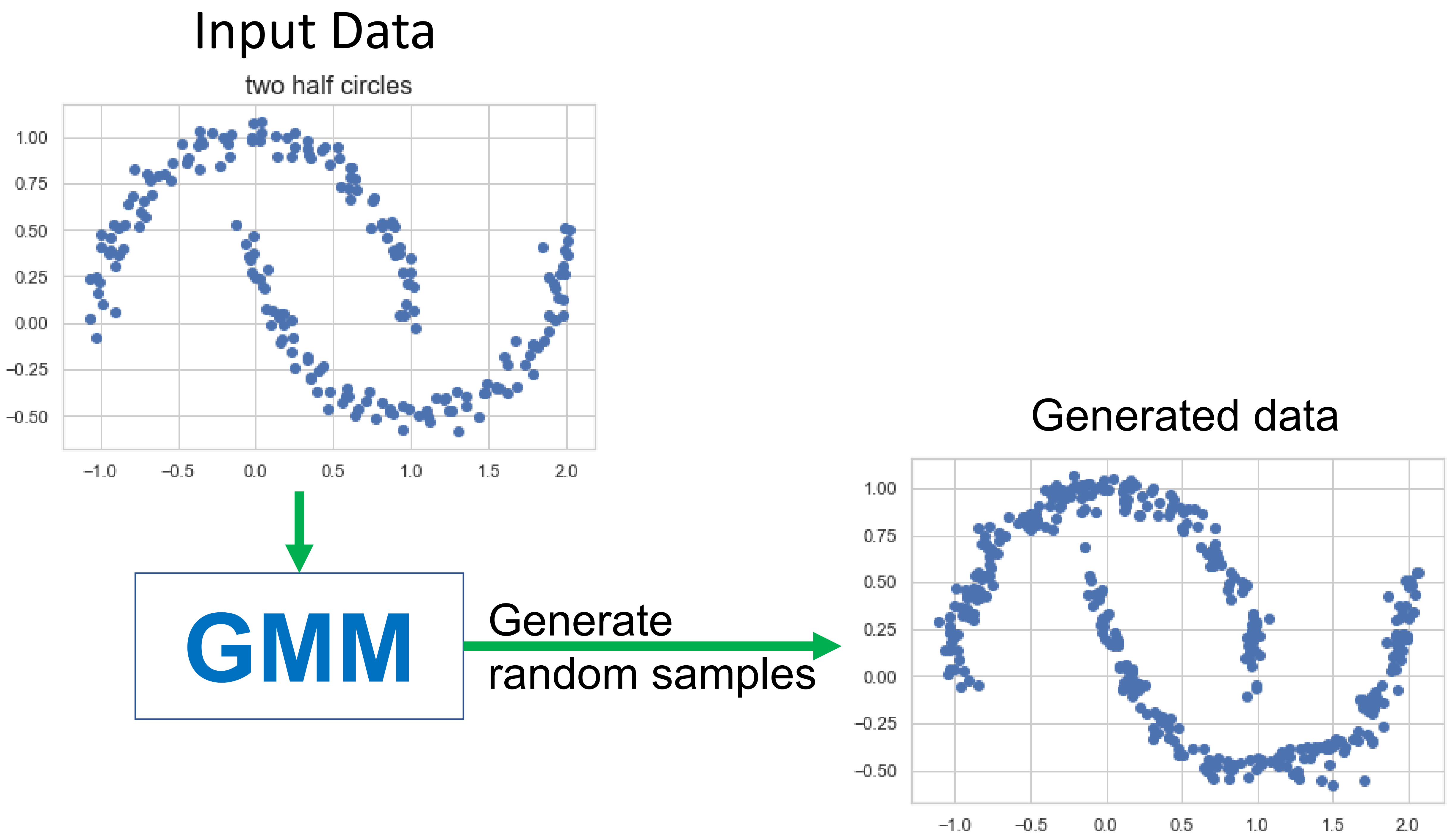


# Input Data




# Use a GMM as an estimation of the “true” PDF of the data





Input Data
two half circles
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
GMM
Generate
random samples
Generated data
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

## Combine GMM and K-means:
## => modified k-means for clustering: faster than GMM

- **E-step (GMM)**: compute the soft assignment $\gamma_{(n,k)}$, the probability of sample $x_n$ belonging to cluster-k

$$\gamma_{(n,k)} = \frac{\pi_k \mathcal{N}(x_n|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n|\mu_j,\Sigma_j)} \text{ where } \sum_{k=1}^{K} \gamma_{(n,k)} = 1$$

- **E-step (modified k-means)**: compute the soft assignment $\gamma_{(n,k)}$, and convert it to hard assignment $\alpha_{(n,k)}$.

  e.g. K=2 soft $\gamma_{(n,1)} = 0.1$, $\gamma_{(n,2)} = 0.9$

  => hard $\alpha_{(n,1)} = 0$, $\alpha_{(n,2)} = 1$

## Combine GMM and K-means:
## => modified k-means for clustering: faster than GMM

- **M-step (modified k-means)**: update the parameters

for each cluster-k, we only use the data points in cluster-k to update the parameters of the cluster-k: $\mu_k, \Sigma_k, \pi_k$ (for $k$=1 to $K$)

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \alpha_{(n,k)} x_n$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \alpha_{(n,k)} (x_n - \mu_k)(x_n - \mu_k)^T$$

$$\pi_k = \frac{N_k}{N} \text{, where } N_k = \sum_{n=1}^{N} \alpha_{(n,k)}$$

$\alpha_{(n,k)} = 1$ if and only if $x_n$ is assigned to the cluster-k

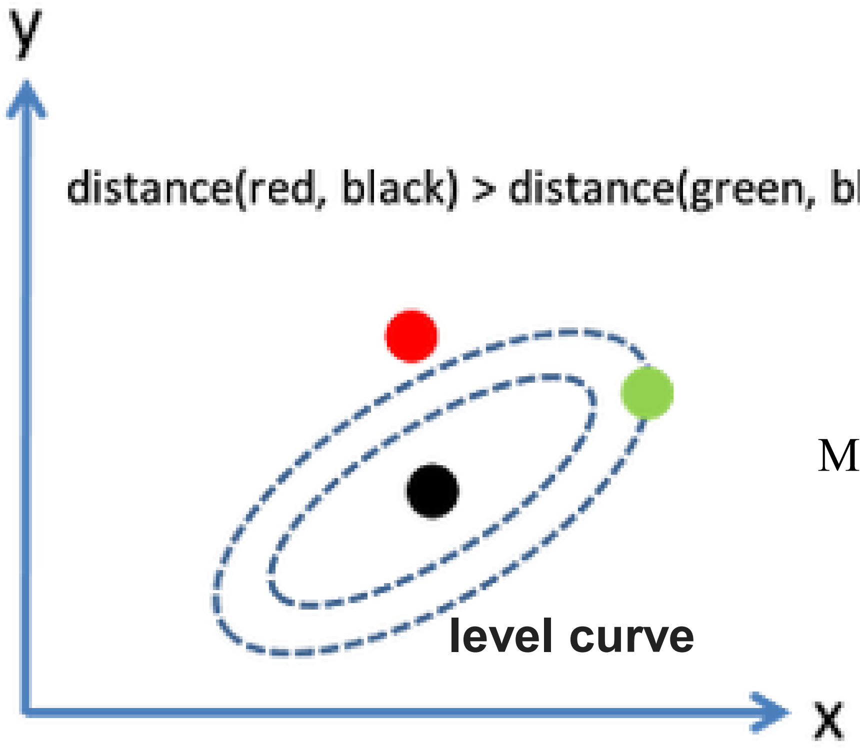
$\alpha_{(n,k)} = 0$ otherwise

# The distance function of the modified k-means

- **The distance function is**

$$d_k(x_n, \mu_k) = \sqrt{(x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k)}$$

**a.ka.** Mahalanobis distance




Mahalanobis distance in 2D space

https://lylelin317.wordpress.com/2014/03/11/distance-and-similarity-measuring-methods-in-machine-learning/