

idea 2 - supervised (labels/tags correlated with desired clusters)

BETTER EVAL

same label \implies same cluster

same cluster \Rightarrow same/similar labels

\searrow small label variance

* deal with granularity/specificity

- clusters
- tags/labels

Soft K-means

$$\pi_{ik} = \text{prob}(x_i \rightarrow \text{cluster } k)$$

fixed i (datapoint x_i)

$$\sum_k \pi_{ik} = 1$$

$$(\pi_{ik} \quad k=1:K)$$

dist for x_i

#

$$(\pi_{jk} \quad k=1:K)$$

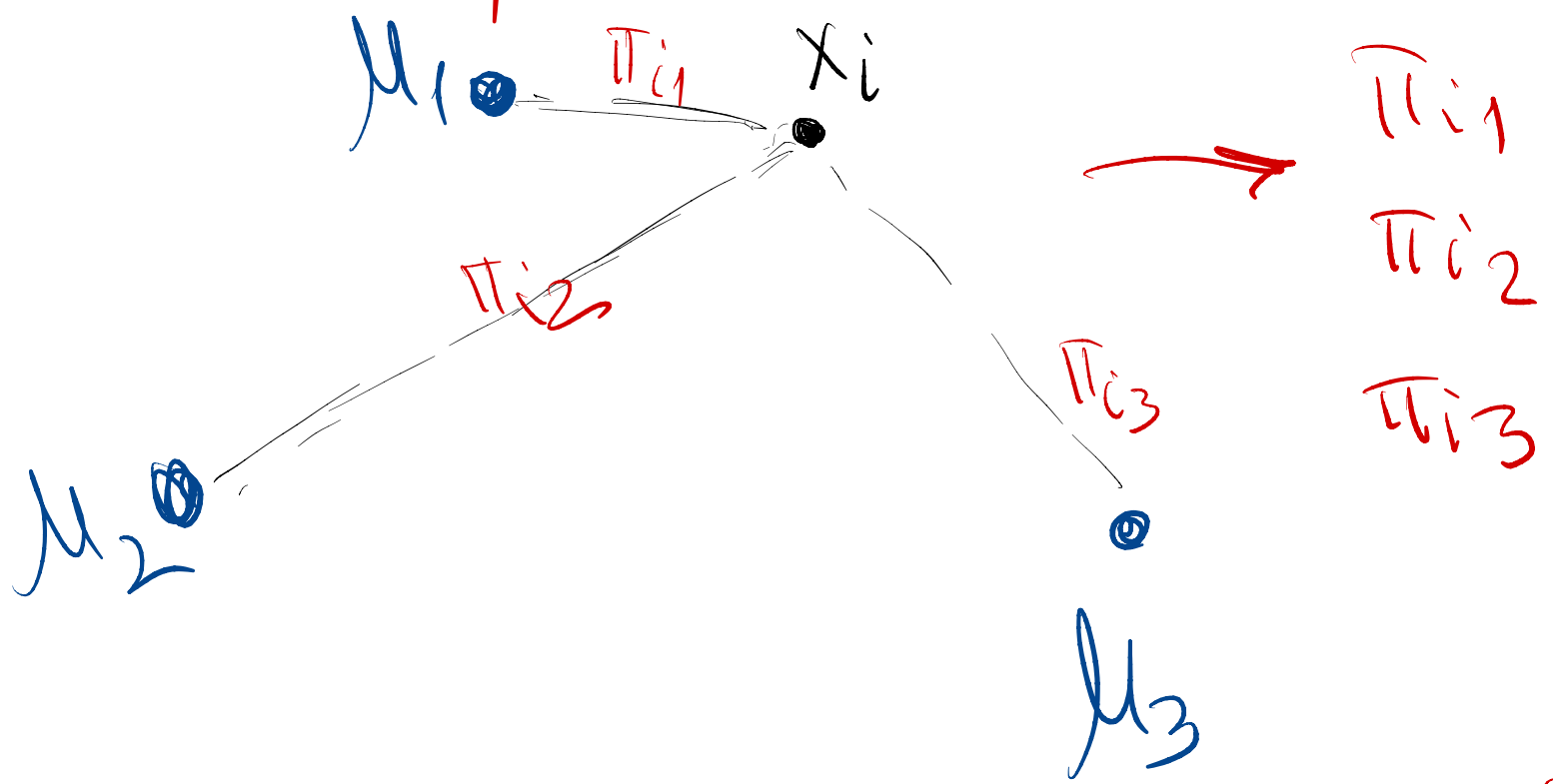
dist for x_j

distance



probability?
likelihood

score
history



need assumption probab = function(score)
 Gauss/Normal assumpt probab $\approx e^{-\text{dist}^2 / \text{dist}}$

$$\exp\left\{-\frac{1}{\beta} \text{dist}^2\right\} = \left(e^{1/\beta}\right)^{-\text{dist}^2}$$

$$= e^{-\frac{\text{dist}^2}{\beta}}$$

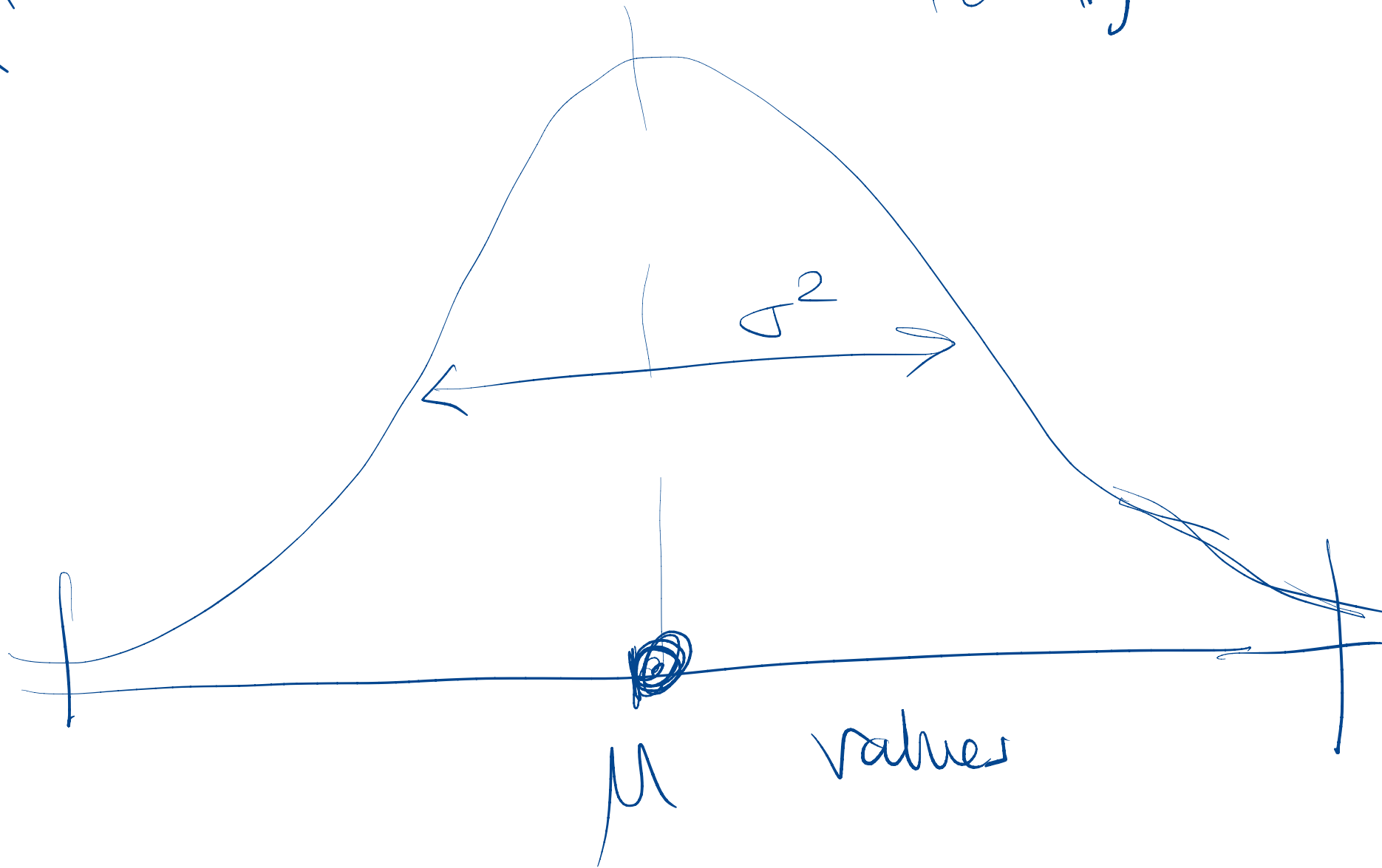
new base
→ scaled dist

$$\sigma^2 = \text{var}$$
$$\sigma = \text{std}$$

$$\mathcal{N}(\mu, \sigma, x) = \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\sqrt{2\sigma^2\pi}}$$

Probab/density

≈ Histogram
(area)



sandipanweb

Simply Data Science

Hard & Soft Clustering with K-means, Weighted K-means and GMM-EM in Python

MARCH 19, 2017APRIL 7, 2017 / SANDIPAN DEY

The following problems appeared as a project in the *edX course ColumbiaX: CSMM.102x Machine Learning*.

In this assignment the following clustering algorithms will be implemented:

1. Hard clustering with **K-means**
2. Soft clustering with
 - a. **Weighted K-means**
 - b. **Gaussian mixture models with Expectation Maximization.**

Some datasets with n data points $\{x_1, \dots, x_n\}$ will be used for testing the algorithms, where each $x_i \in \mathbb{R}^d$.

Hard Clustering

1. Each point is assigned to a one and only one cluster (hard

assignment).

2. With **K-means** we try to find K centroids $\{\mu_1, \dots, \mu_K\}$ and the corresponding assignments of each data point $\{c_1, \dots, c_n\}$ where each $c_i \in \{1, \dots, K\}$ and c_i indicates which of the K clusters the observation x_i belongs to. The objective function that we seek to minimize can be written as

$$\mathcal{L} = \sum_{i=1}^n \sum_{k=1}^K \mathbf{1}(c_i = k) \|x_i - \mu_k\|^2$$

Soft Clustering

$\phi_i(k) =$ membership probability
pr ($x_i \rightarrow$ cluster k)

We called this like in previous notes.

- (1) Each point is assigned to all the clusters with different weights or probabilities (soft assignment).

- (2) With **Weighted K-means** we try to compute the weights $\phi_i(k)$ for each data point i to the cluster k as minimizing the following objective:

$$\mathcal{L} = \sum_{i=1}^n \sum_{k=1}^K \phi_i(k) \frac{\|x_i - \mu_k\|^2}{\beta}$$

$$\phi_i(k) > 0. \quad \sum_{k=1}^K \phi_i(k) = 1. \quad \beta > 0.$$

- (3) With **GMM-EM** we can do soft clustering too. The **EM** algorithm can be used to learn the parameters of a Gaussian mixture model. For this model, we assume a generative process for the data as follows:

$$x_i | c_i \sim \text{Normal}(\mu_{c_i}, \Sigma_{c_i}), \quad c_i \sim \text{Discrete}(\pi).$$

In other words, the i -th observation is first assigned to one of K clusters according to the probabilities in vector π , and the value of observation x_i is then generated from one of K multivariate Gaussian distributions, using the mean and covariance indexed by c_i . The EM algorithm seeks to maximize

$$p(x_1, \dots, x_n | \pi, \mu, \Sigma) = \prod_{i=1}^n p(x_i | \pi, \mu, \Sigma)$$

over all parameters $\pi, \mu_1, \dots, \mu_K, \Sigma_1, \dots, \Sigma_K$ using the cluster assignments c_1, \dots, c_n as the hidden data.

The following figures show the algorithms that are going to be implemented for clustering.

HARD CLUSTERING MODELS

Taken from the slides of the edX course ColumbiaX: CSMM.102x Machine Learning

Review: K-means clustering algorithm

Given: Data x_1, \dots, x_n , where $x \in \mathbb{R}^d$

Goal: Minimize $\mathcal{L} = \sum_{i=1}^n \sum_{k=1}^K \mathbb{1}\{c_i = k\} \|x_i - \mu_k\|^2$.

$$\phi_i(k) = \begin{cases} 1 & \text{if } x_i \rightarrow \text{cluster } k \\ 0 & \text{otherwise} \end{cases}$$

► Iterate until values no longer changing

1. Update c : For each i , set $c_i = \arg \min_k \|x_i - \mu_k\|^2$
2. Update μ : For each k , set $\mu_k = (\sum_i x_i \mathbb{1}\{c_i = k\}) / (\sum_i \mathbb{1}\{c_i = k\})$

WEIGHTED K-MEANS (SOFT CLUSTERING EXAMPLE)

Weighted K-means clustering algorithm

Given: Data x_1, \dots, x_n , where $x \in \mathbb{R}^d$

Goal: Minimize $\mathcal{L} = \sum_{i=1}^n \sum_{k=1}^K \phi_i(k) \frac{\|x_i - \mu_k\|^2}{\beta}$ over ϕ_i and μ_k

Conditions: $\phi_i(k) > 0$ and $\sum_{k=1}^K \phi_i(k) = 1$. Set parameter $\beta > 0$.

► Iterate the following

1. Update ϕ : For each i , update the word allocation weights

$$\phi_i(k) = \frac{\exp\{-\frac{1}{\beta} \|x_i - \mu_k\|^2\}}{\sum_j \exp\{-\frac{1}{\beta} \|x_i - \mu_j\|^2\}}, \text{ for } k = 1, \dots, K$$

prob membership

2. Update μ : For each k , update μ_k with the *weighted* average

$$\mu_k = \frac{\sum_i x_i \phi_i(k)}{\sum_i \phi_i(k)}$$

weighted avg

Taken from the slides of the edX course ColumbiaX: CSMM.102x Machine Learning

EM FOR THE GMM

Taken from the slides of the edX course ColumbiaX: CSMM.102x Machine Learning

Algorithm: Maximum likelihood EM for the GMM

Given: x_1, \dots, x_n where $x \in \mathbb{R}^d$

Goal: Maximize $\mathcal{L} = \sum_{i=1}^n \ln p(x_i | \pi, \mu, \Sigma)$.

► Iterate until incremental improvement to \mathcal{L} is “small”

1. **E-step:** For $i = 1, \dots, n$, set

probab membership pr (cluster=k | data = x_i)

$$\phi_i(k) = \frac{\pi_k \mathcal{N}(x_i | \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(x_i | \mu_j, \Sigma_j)}, \text{ for } k = 1, \dots, K$$

source k generator distribution d dim

2. **M-step:** For $k = 1, \dots, K$, define $n_k = \sum_{i=1}^n \phi_i(k)$ and update the values

cluster k size or E[# of points]

$$\pi_k = \frac{n_k}{n}, \quad \mu_k = \frac{1}{n_k} \sum_{i=1}^n \phi_i(k) x_i, \quad \Sigma_k = \frac{1}{n_k} \sum_{i=1}^n \phi_i(k) (x_i - \mu_k)(x_i - \mu_k)^T$$

same as before $\sum_i \phi_i(k)$

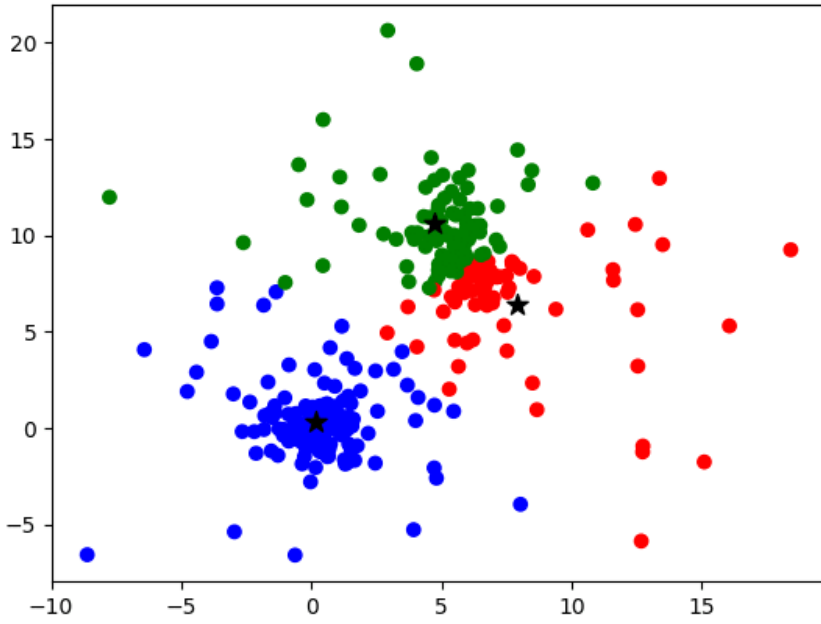
estimate of "width" for gaussian k covariance matrix

Comment: The updated value for μ_k is used when updating Σ_k .

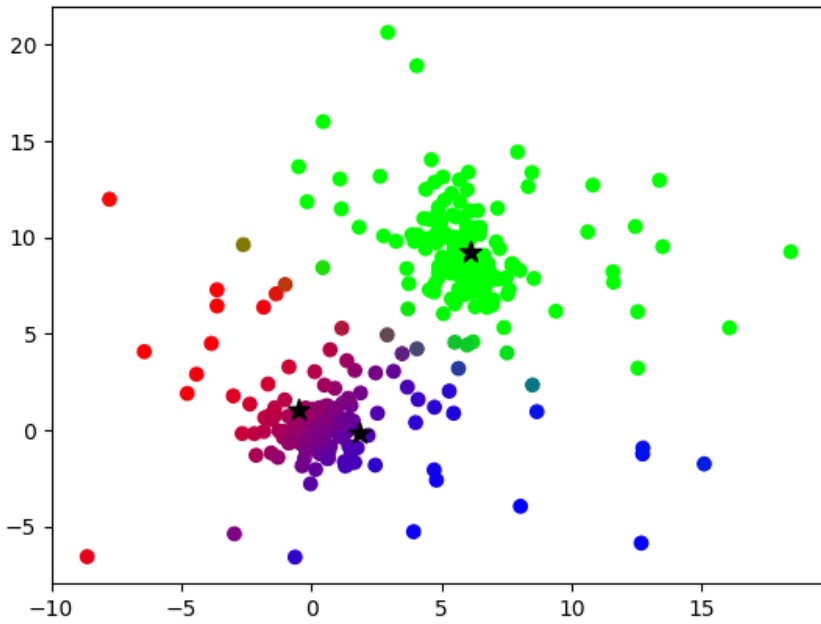
The following animations show the output of the clustering algorithms (and how they converge with different iterations) on a few datasets (with $k=3$ clusters), the weighted k-means is run with the stiffness-parameter $\beta=10$.

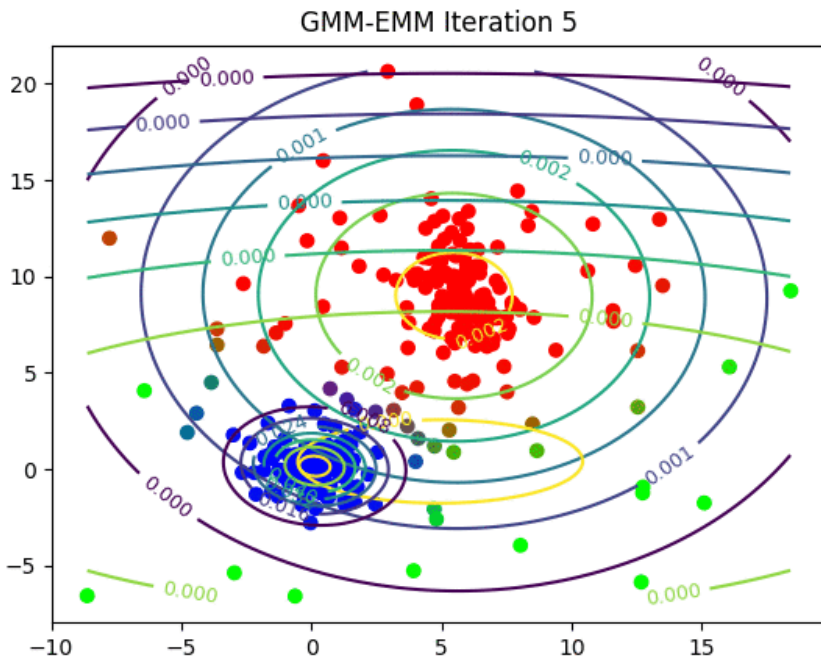
Dataset1

Kmeans Iteration 8



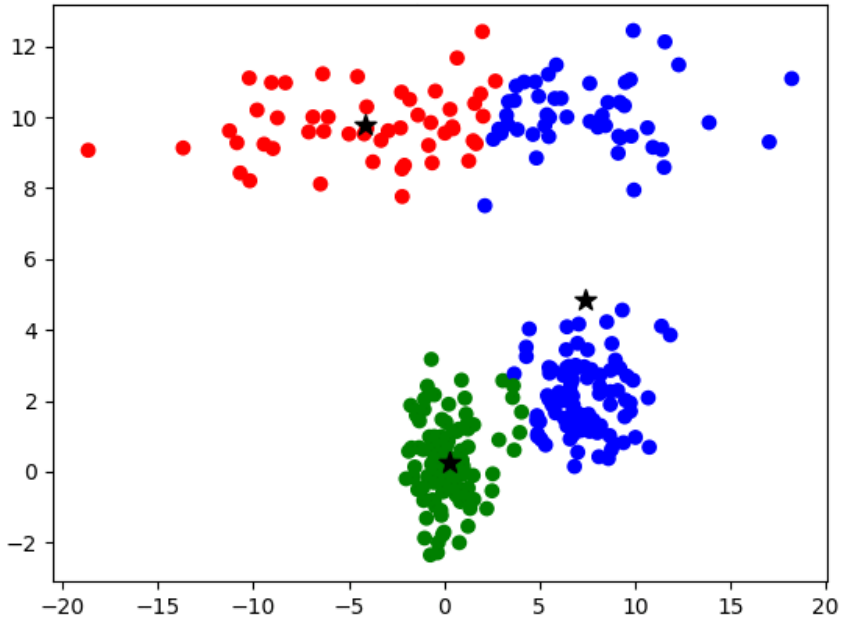
Weighted Kmeans Iteration 6



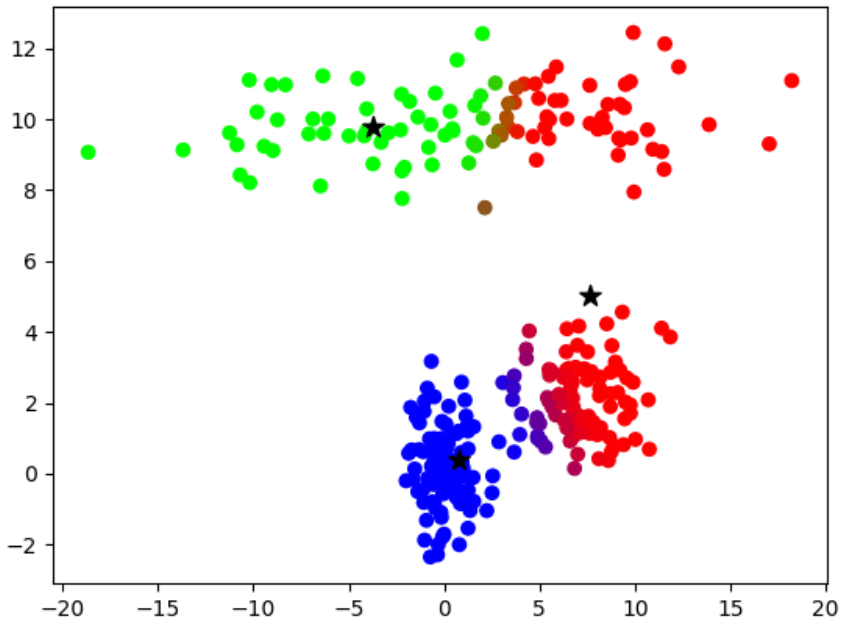


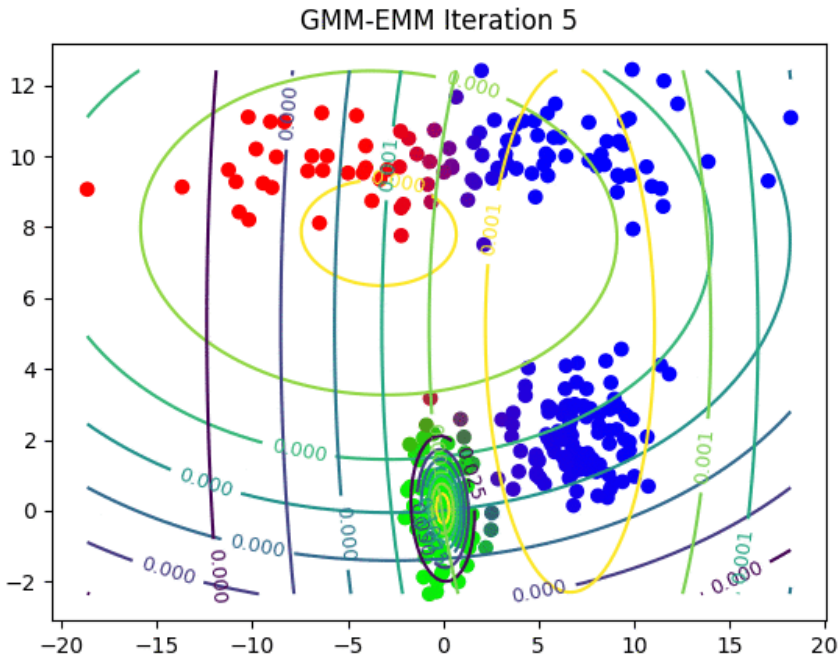
Dataset2

Kmeans Iteration 5



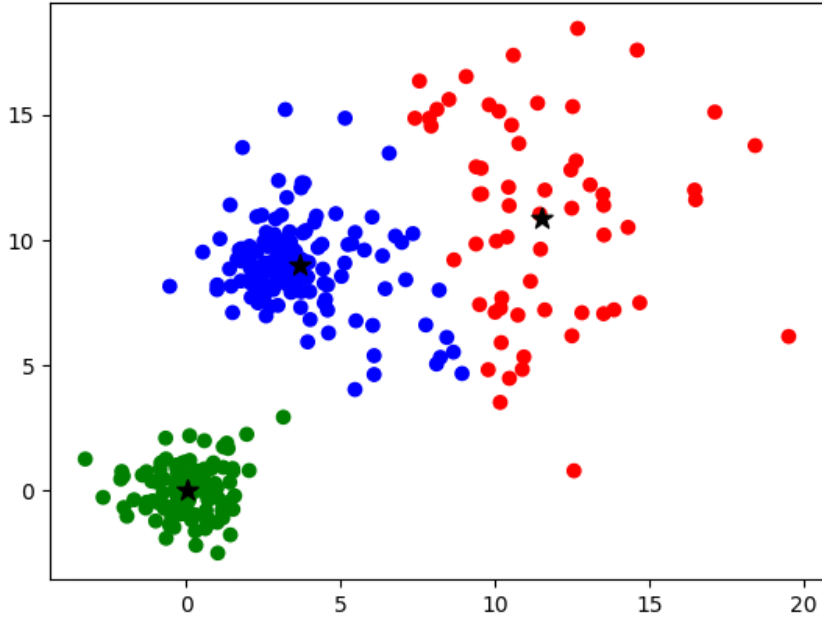
Weighted Kmeans Iteration 5



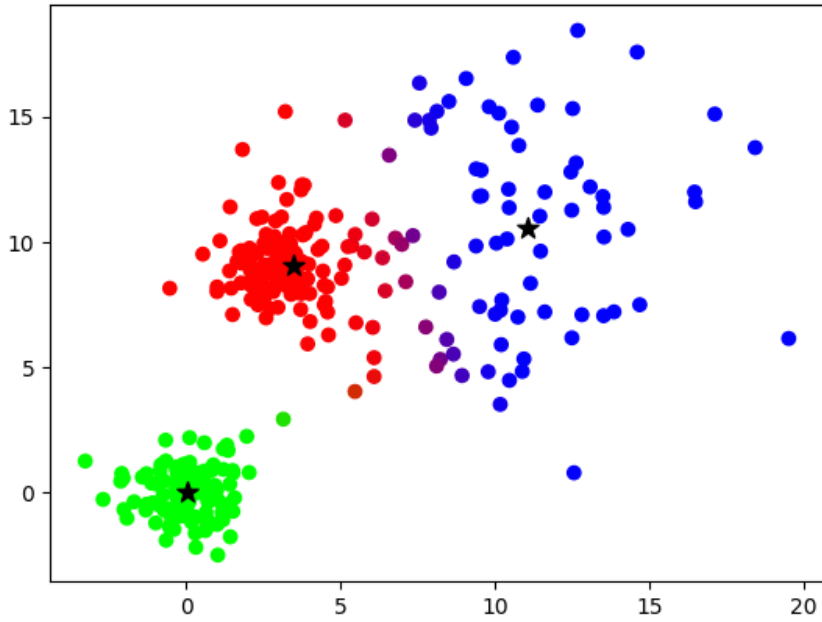


Dataset3

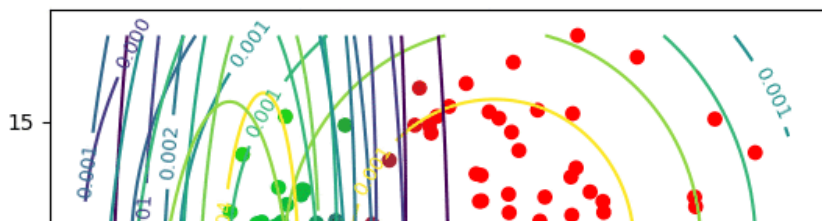
Kmeans Iteration 5

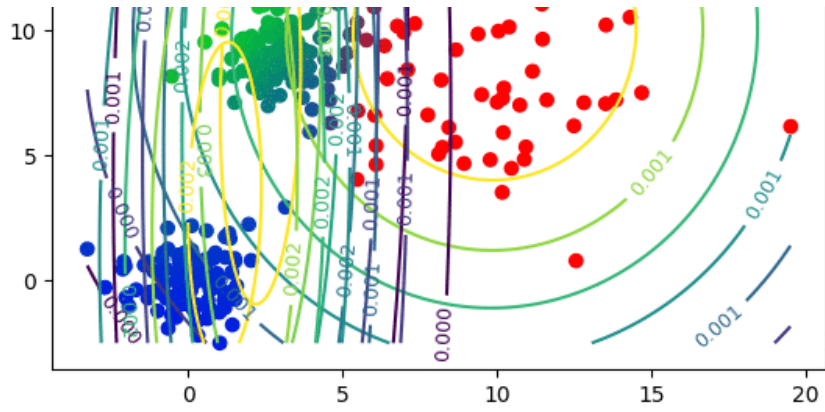


Weighted Kmeans Iteration 5

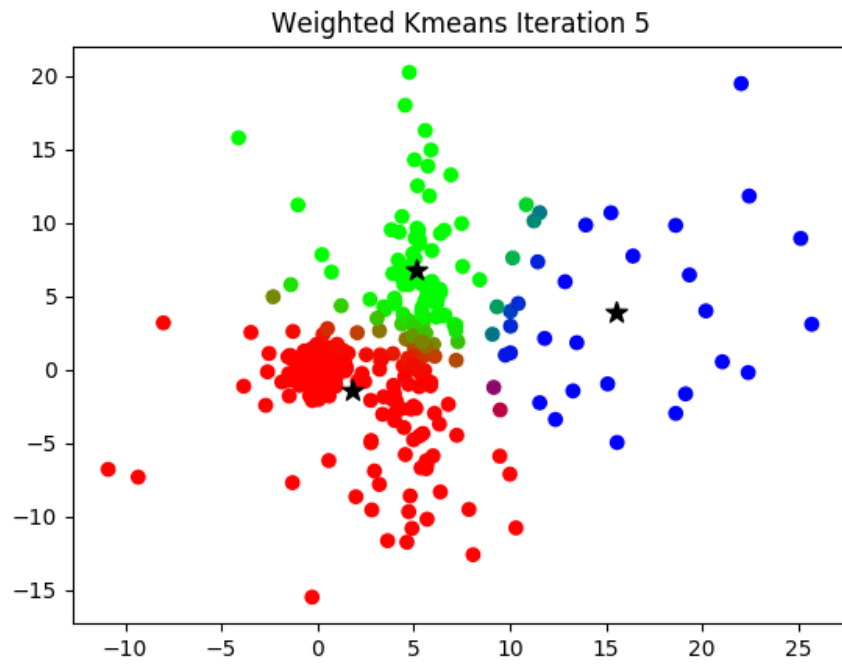
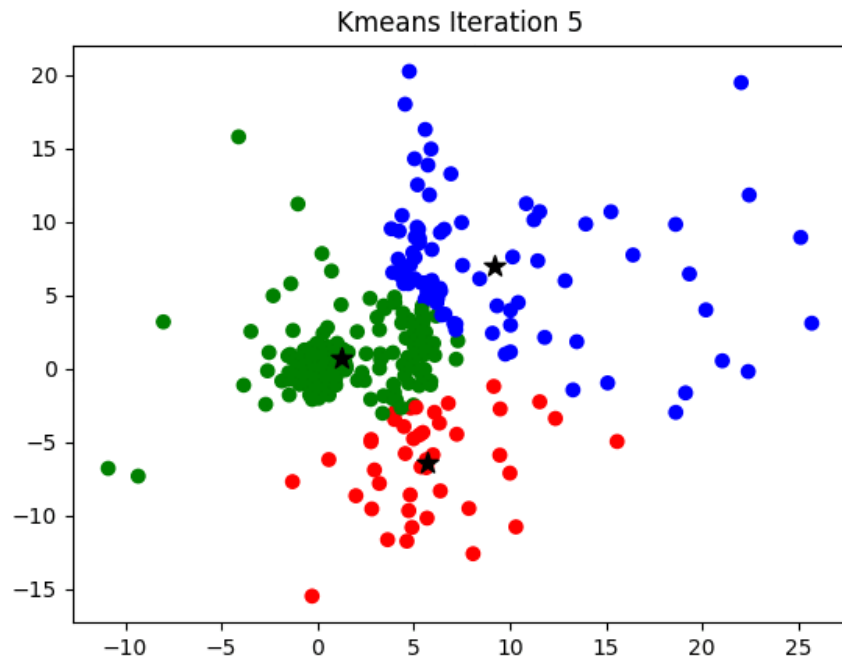


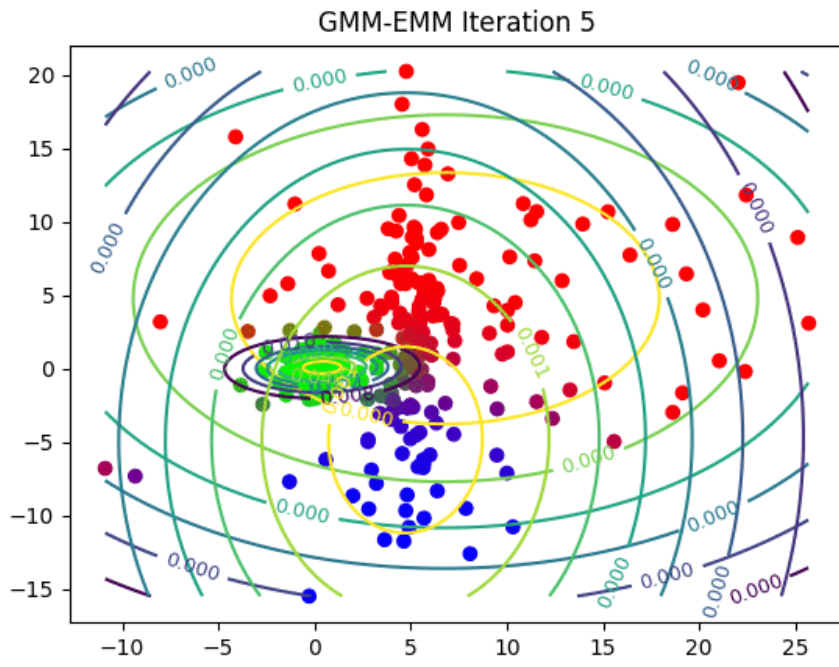
GMM-EMM Iteration 5





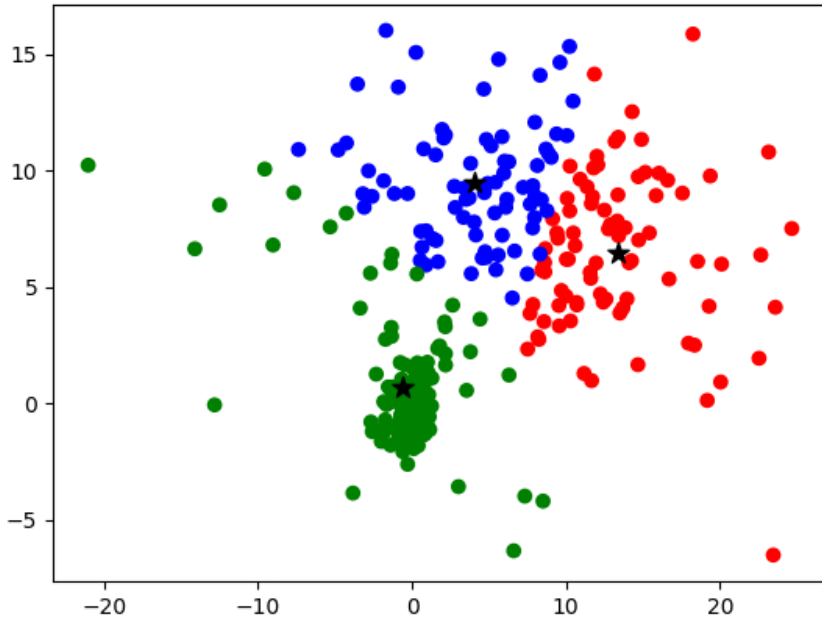
Dataset4



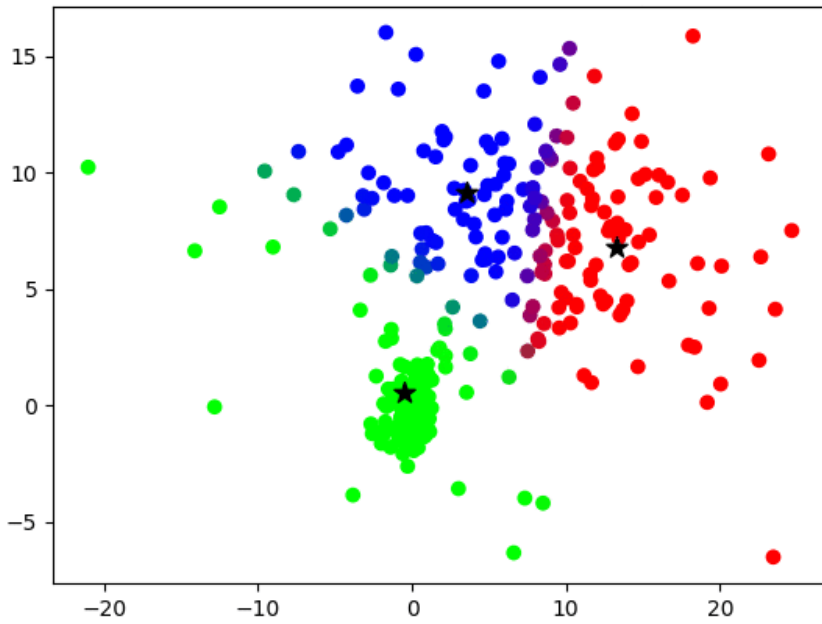


Dataset5

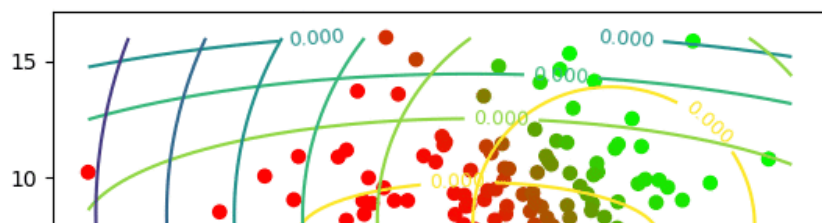
Kmeans Iteration 5

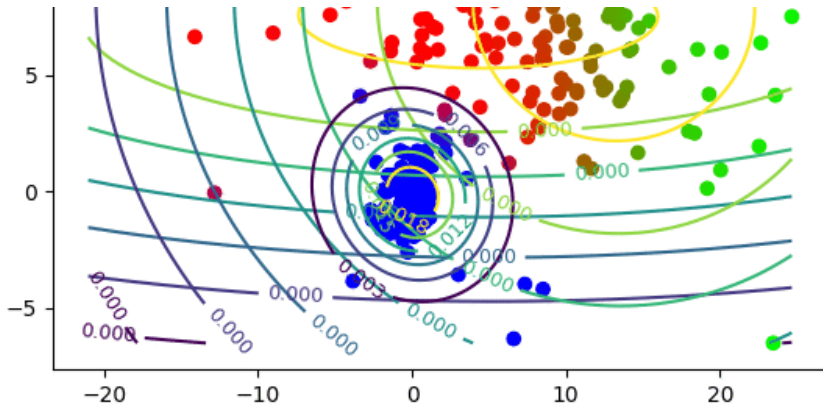


Weighted Kmeans Iteration 5

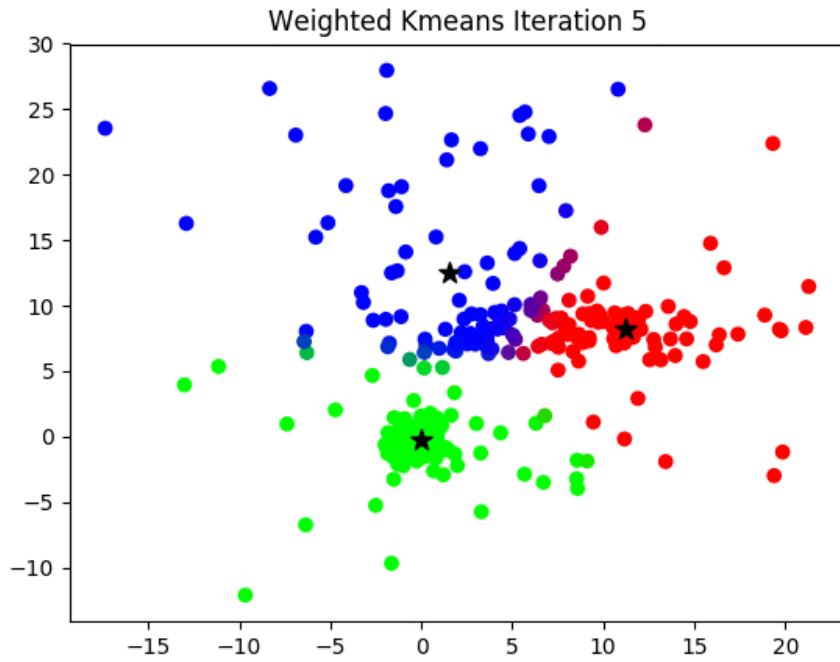
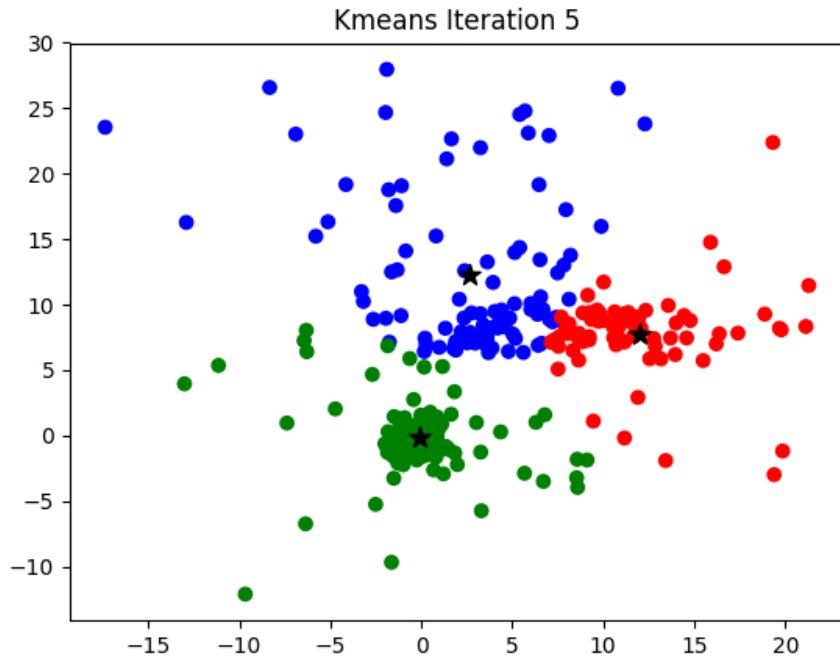


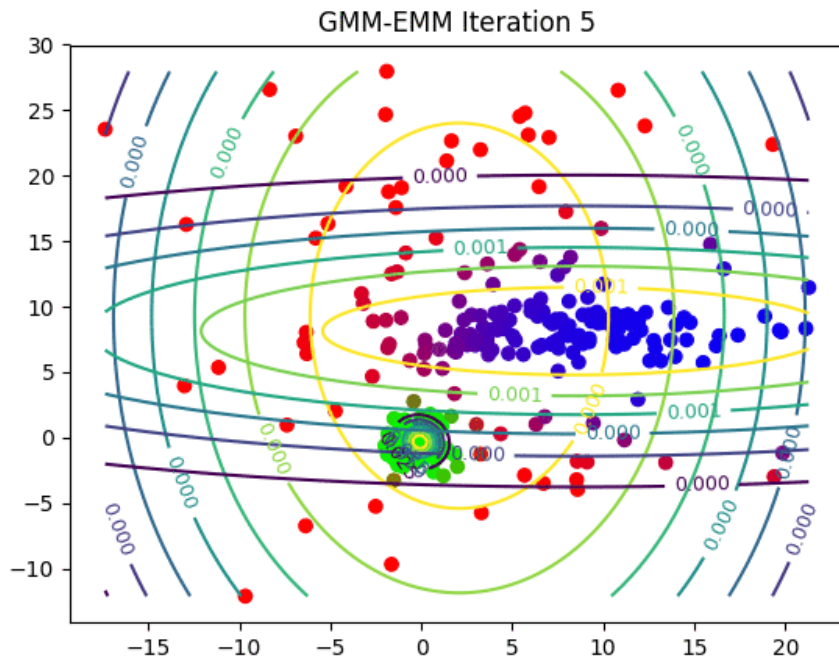
GMM-EMM Iteration 5





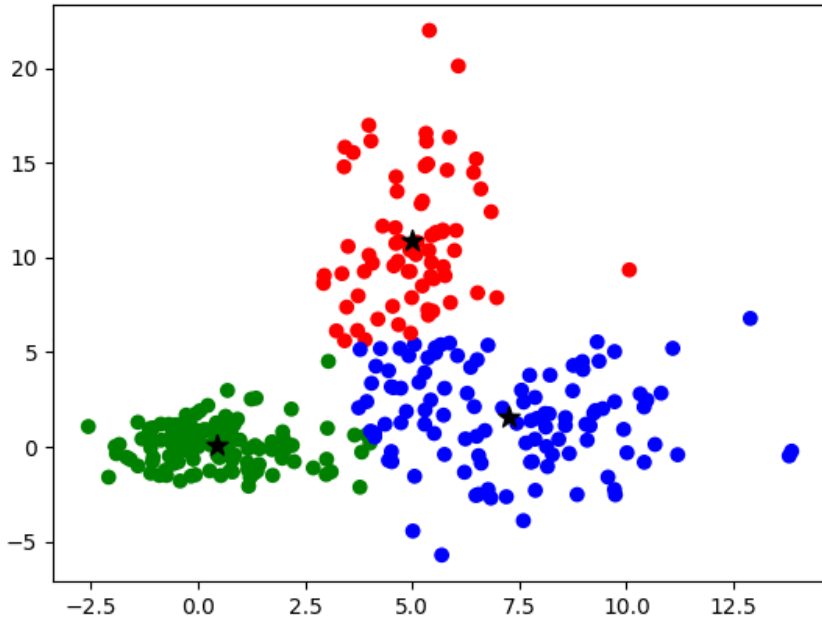
Dataset6



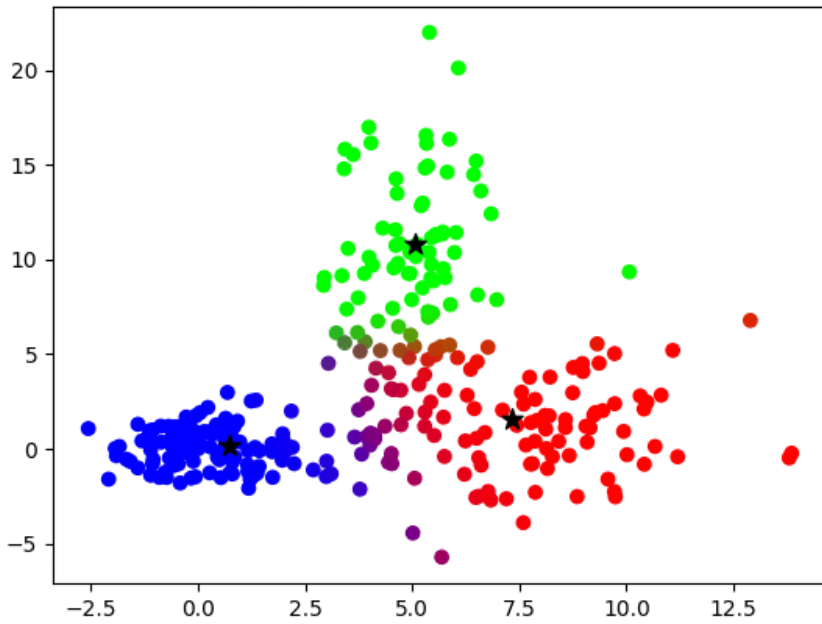


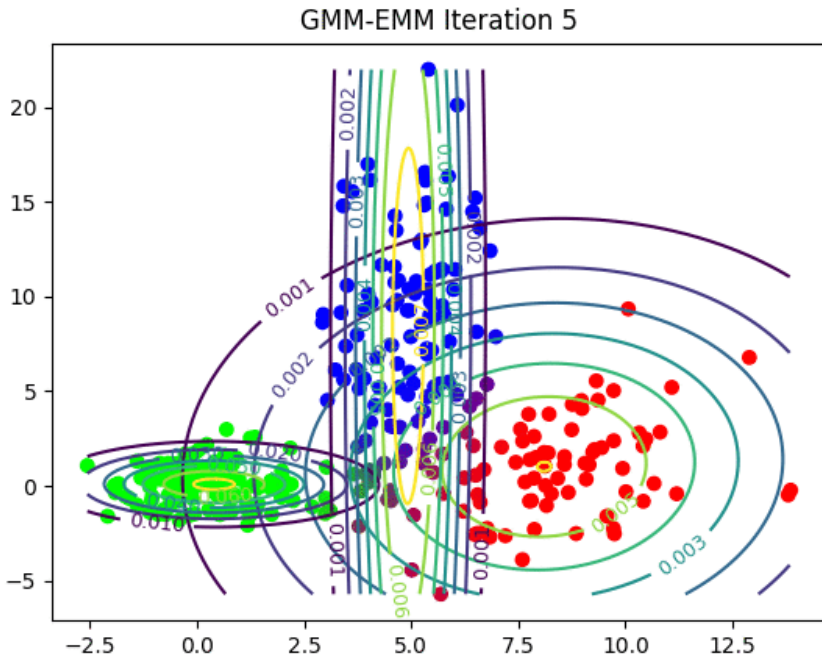
Dataset7

Kmeans Iteration 5

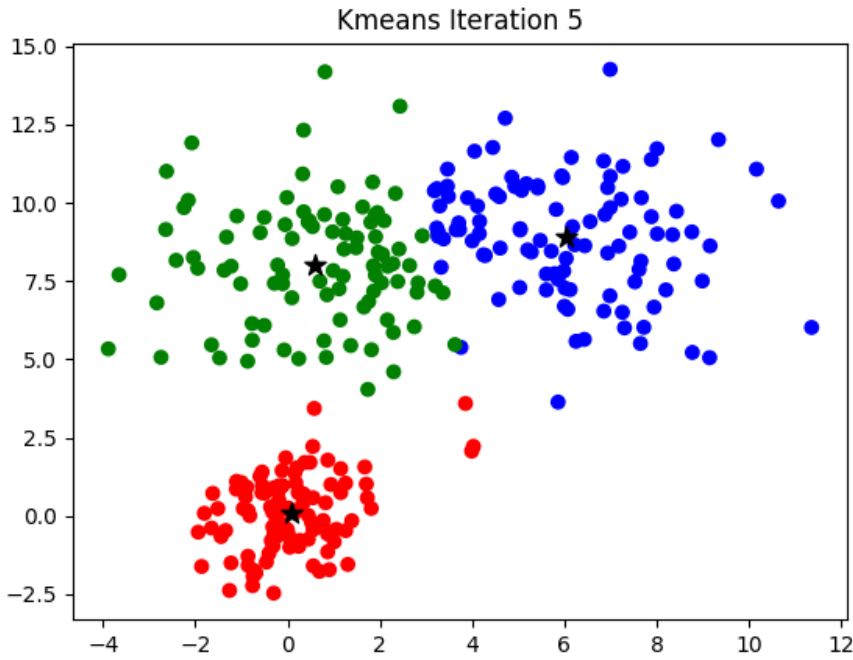


Weighted Kmeans Iteration 5

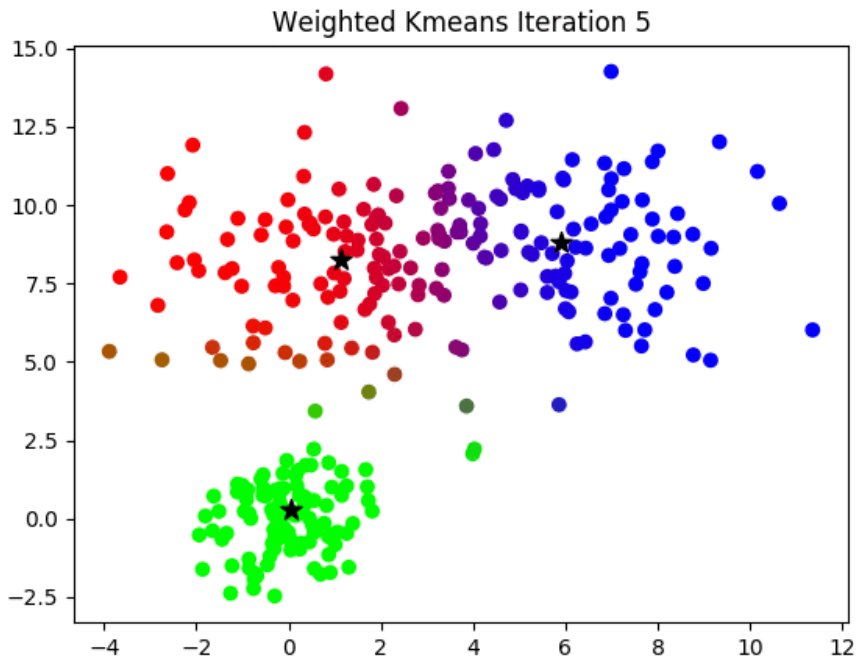




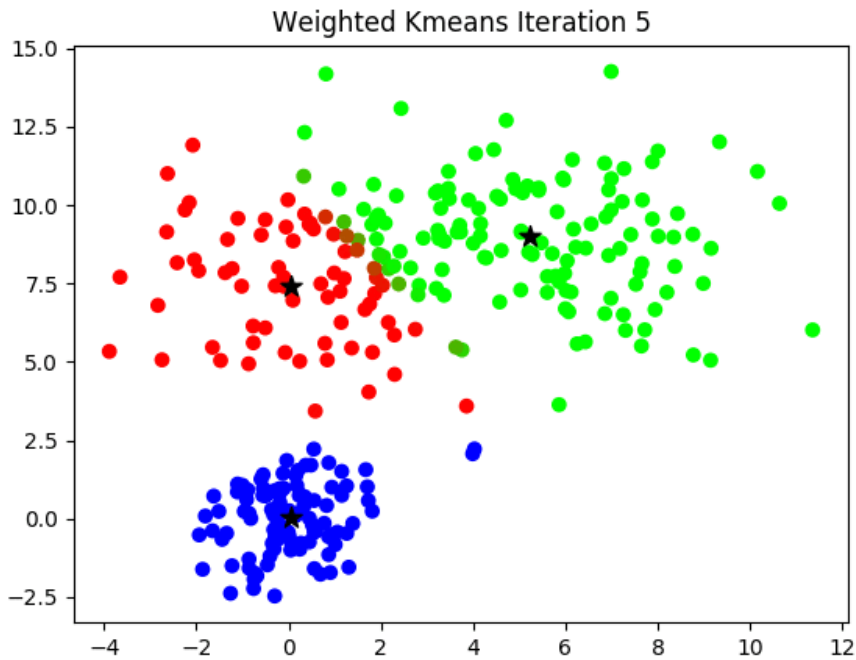
Dataset8

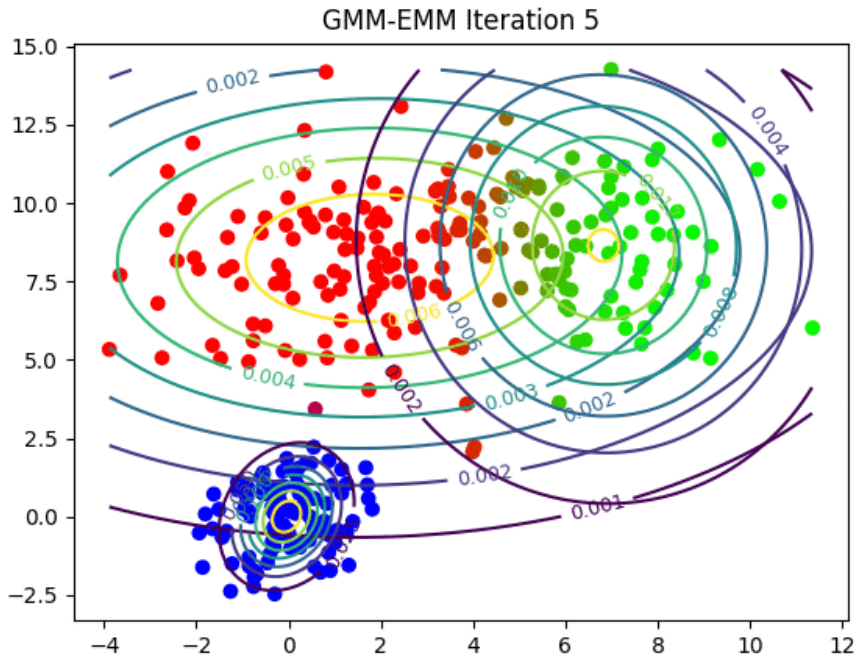


The next animation shows the results with **weighted K-means** with stiffness parameter **beta=10**

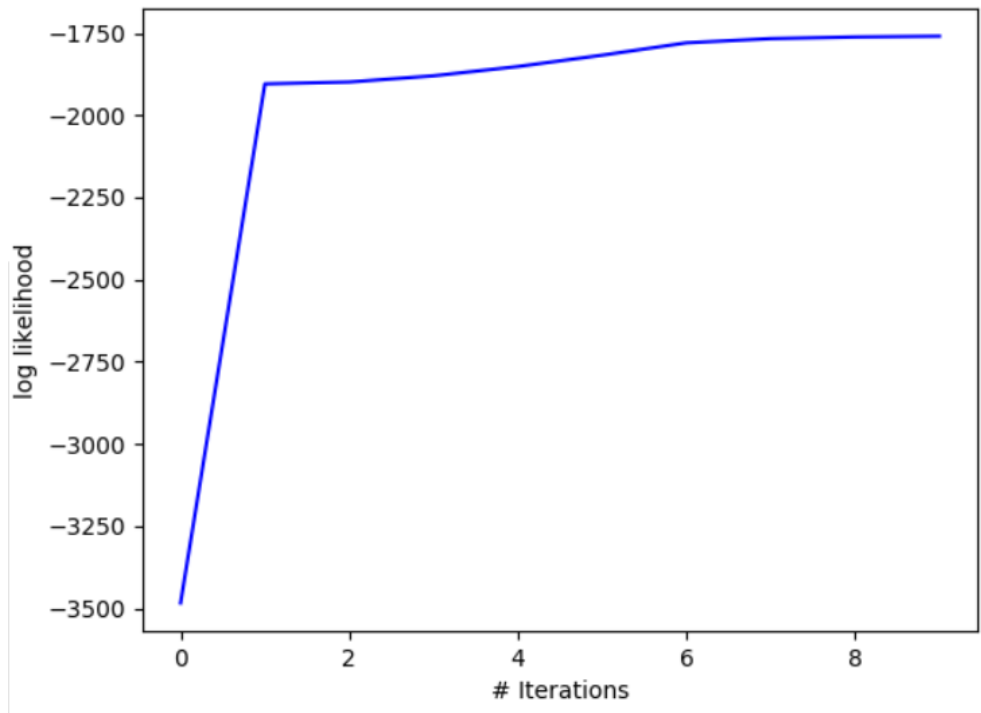


The next animation shows the results with **weighted K-means** with stiffness parameter **beta=1**

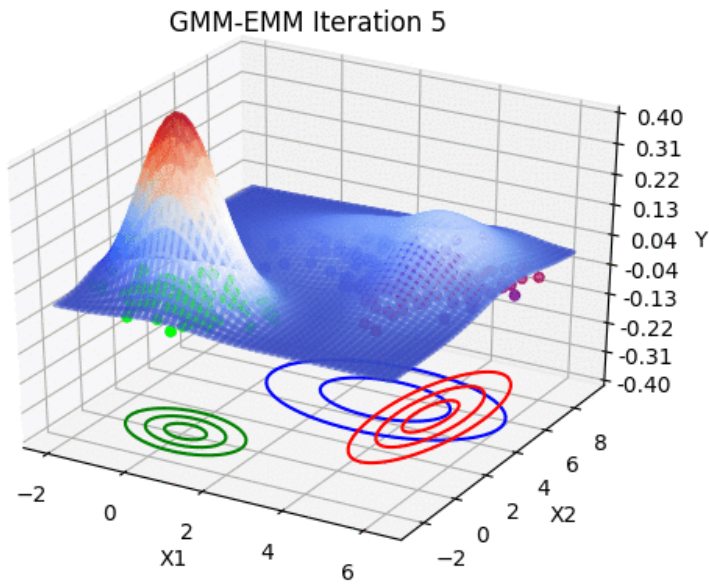
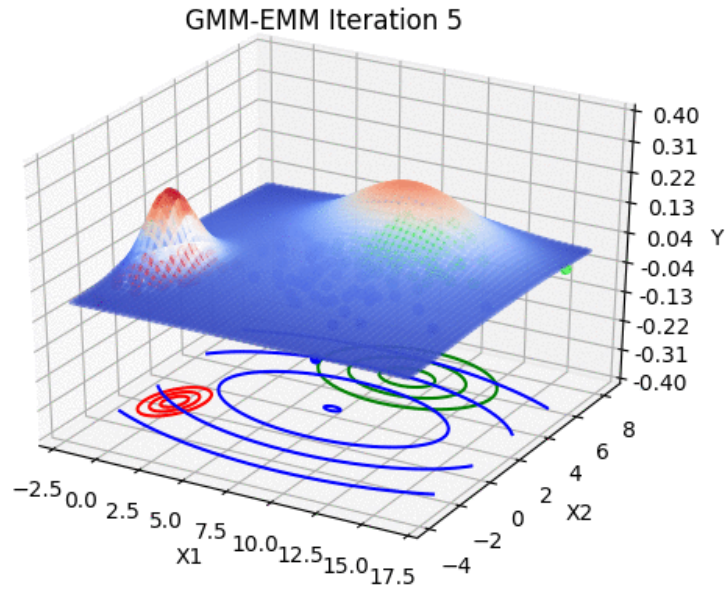


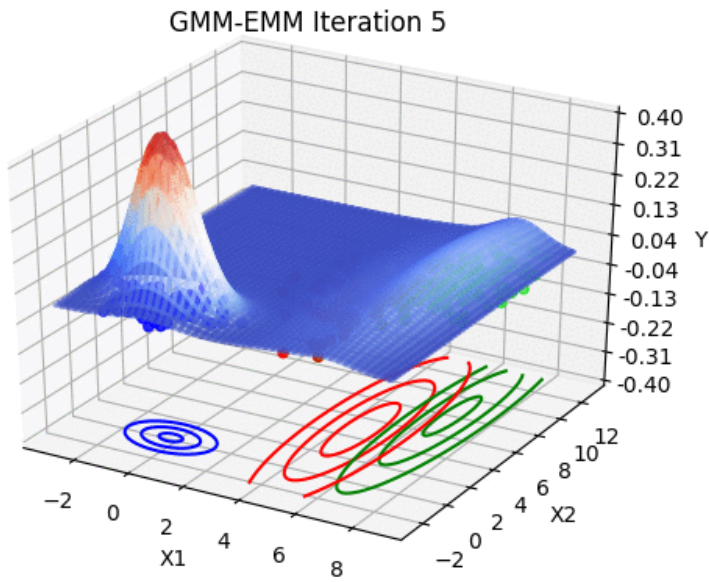
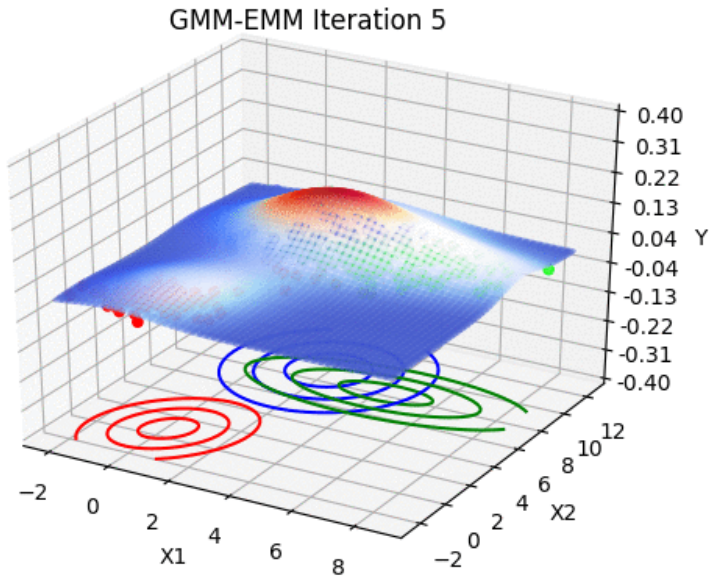


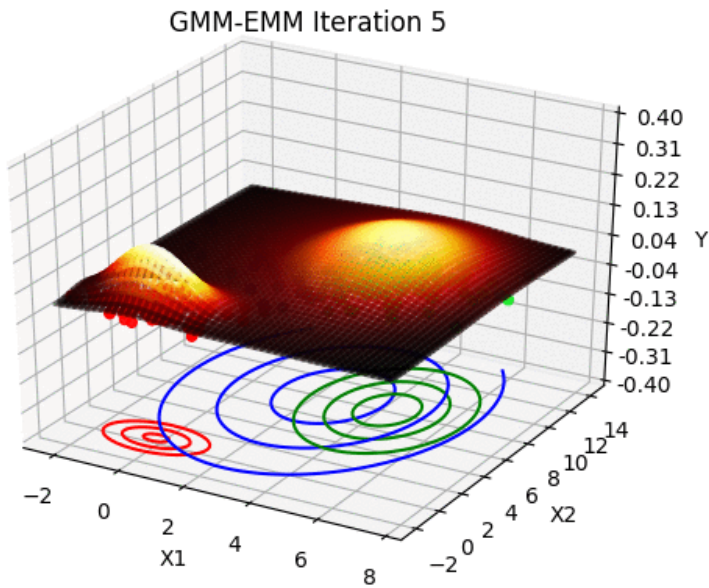
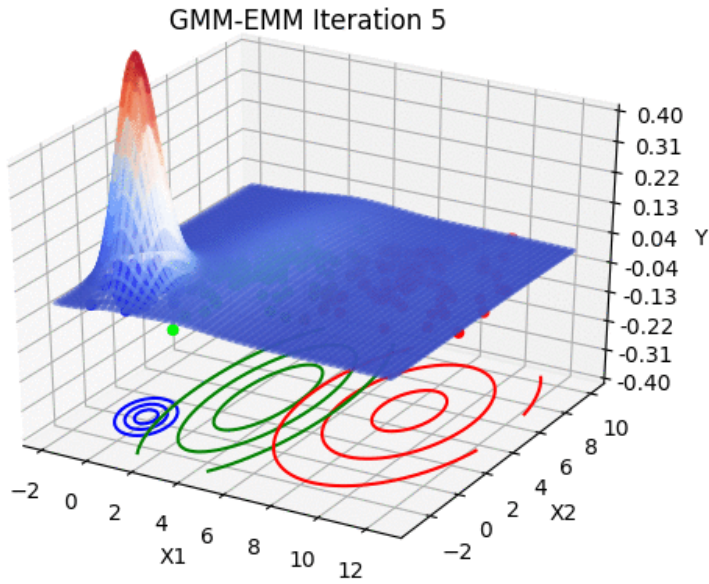
Here is how the **log-likelihood** for **EM** continuously increases over the iterations for a particular dataset:

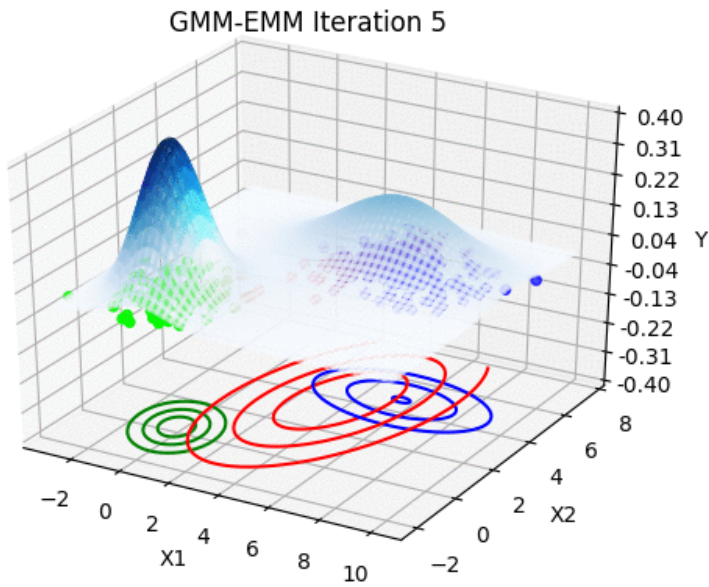
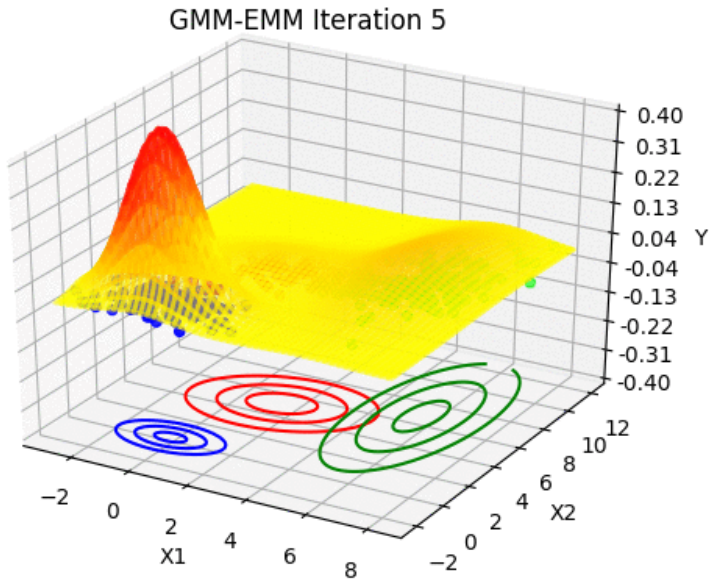


The following animations show **GMM EM convergence** on a few more datasets:

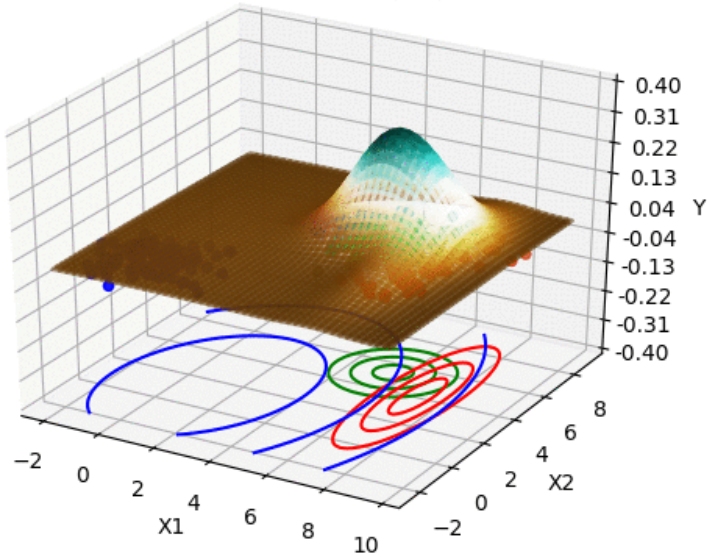




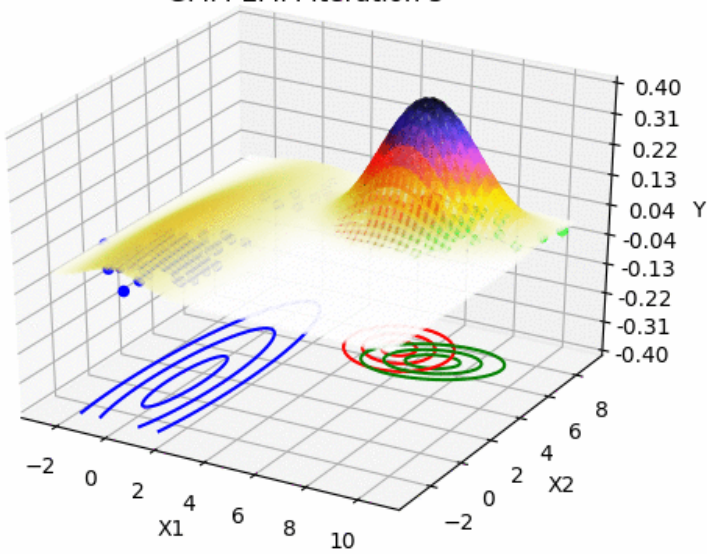




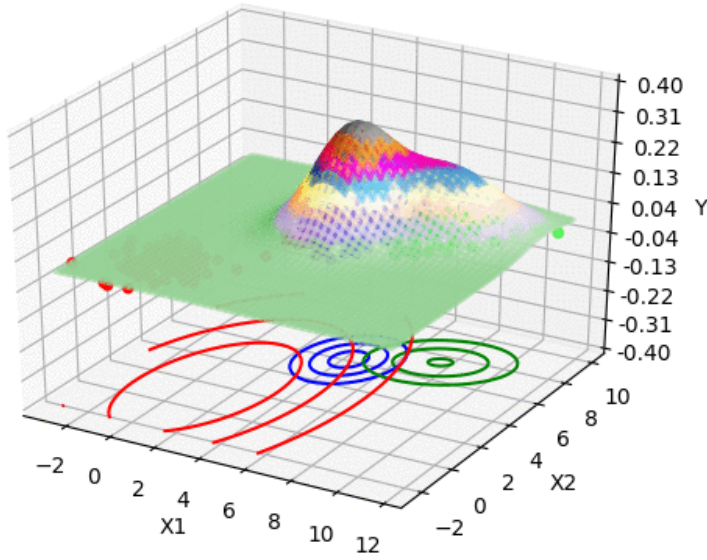
GMM-EMM Iteration 5



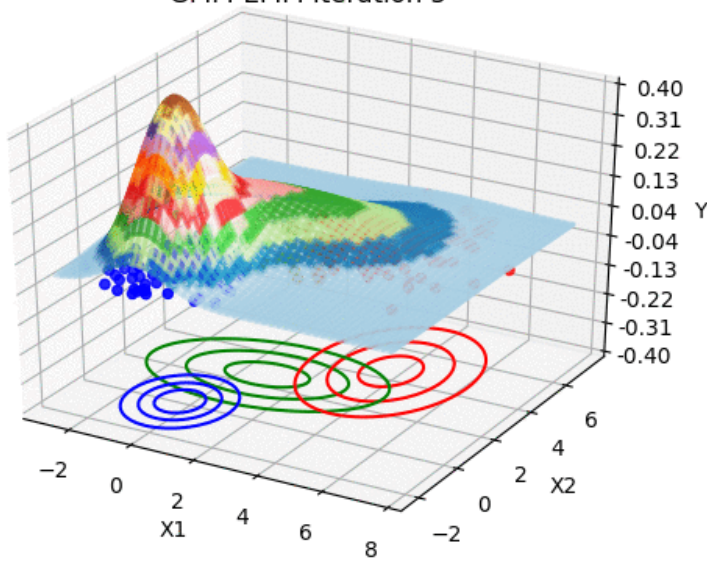
GMM-EMM Iteration 5

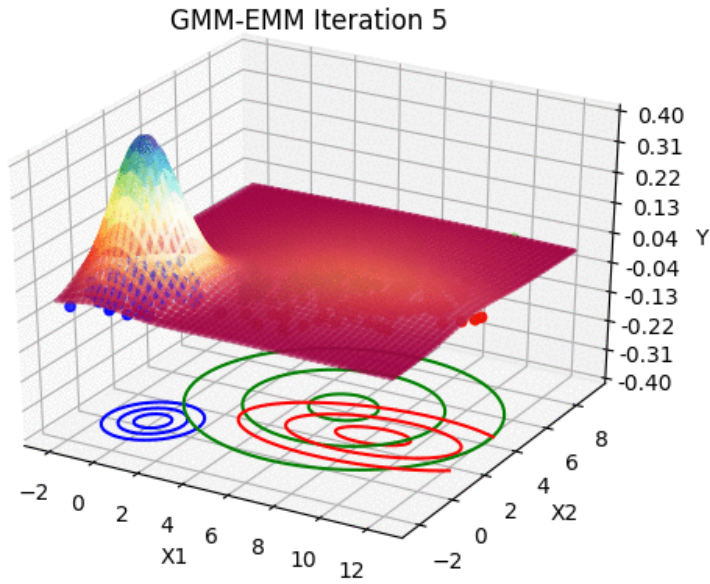


GMM-EMM Iteration 5



GMM-EMM Iteration 5





Advertisements

[ML](#), [Python](#), [Uncategorized](#)

[POWERED BY WORDPRESS.COM.](#)