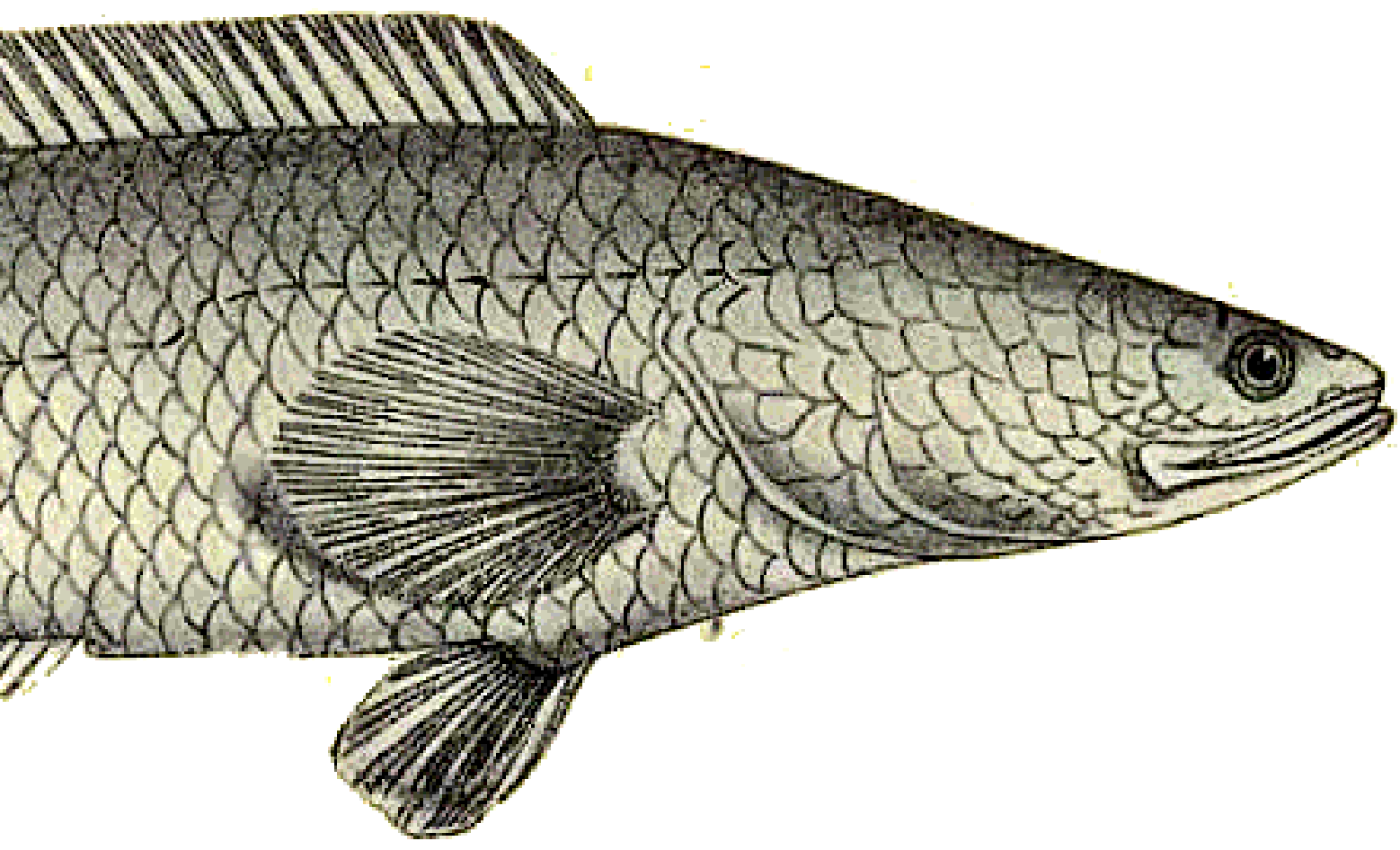
GAUSSIAN
MIXTURE MODEL

ARIHARASUDHAN



Intro : Expectation Maximization

Vanakkam! It's Ari, eager to uncover the yet-to-be-opened boxes of knowledge in machine learning. Before we delve into the Gaussian Mixture Model, we need to understand the Expectation-Maximization Algorithm. EM is a statistical technique used for solving problems related to parameter estimation in probabilistic models, especially when dealing with incomplete or missing data.

Parameter Estimation? What is it?

Parameter estimation is a fundamental concept in statistics and data analysis. It involves the process of determining the values of unknown parameters in a statistical model based on observed data.

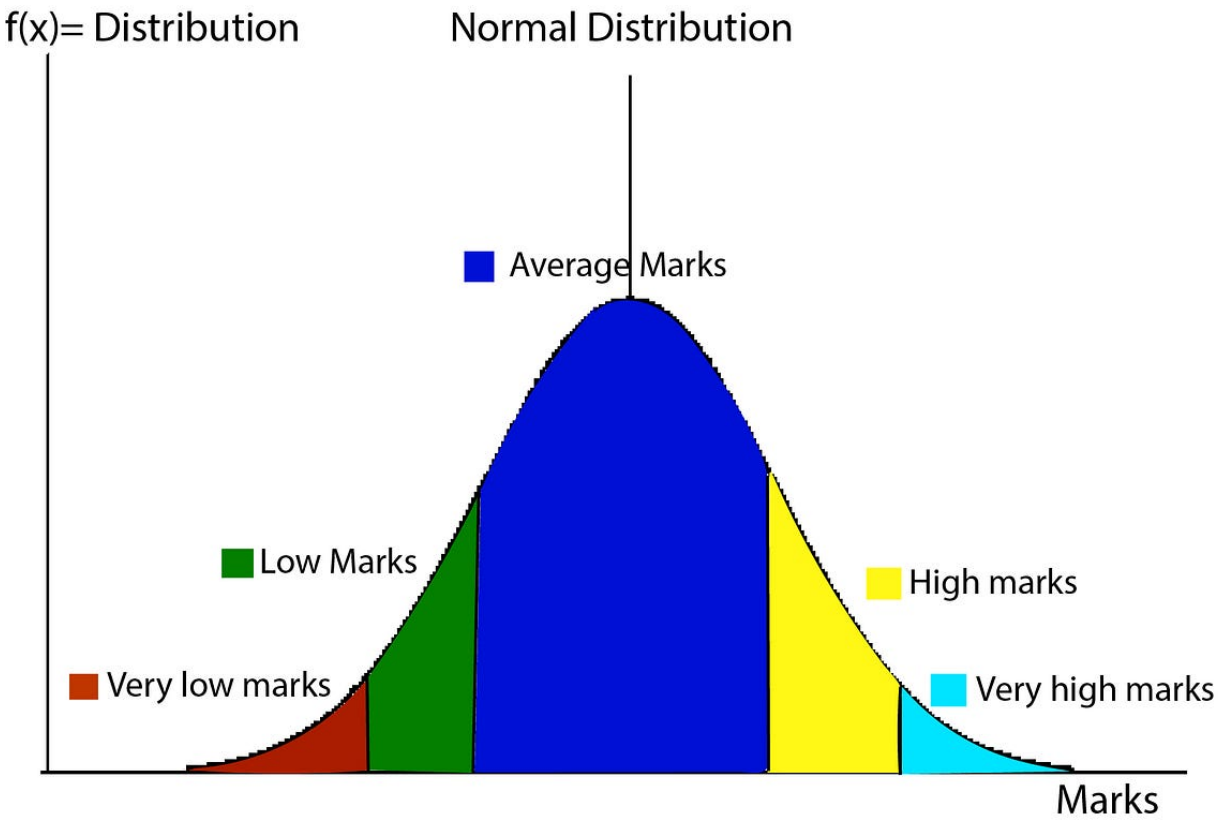
These parameters describe various characteristics of the model, such as the mean, variance, shape, or other properties of a probability distribution.

Example: Estimating the Mean

If we have a dataset consisting of the following weight scores of fishes,

[85, 92, 88, 78, 95, 90, 87, 89, 84, 91].

Our goal is to estimate the mean (average) weight score of the entire class of fishes using parameter estimation. In this case, the parameter we want to estimate is the population mean (μ). We can assume that the weight scores are normally distributed with an unknown mean (μ) and an unknown standard deviation (σ). The normal distribution is a common model for representing continuous data like weight scores.



To estimate the population mean (μ), we can use the sample mean ($\bar{x}$) as an estimator. The sample mean is calculated as the sum of all observations divided by the number of observations.

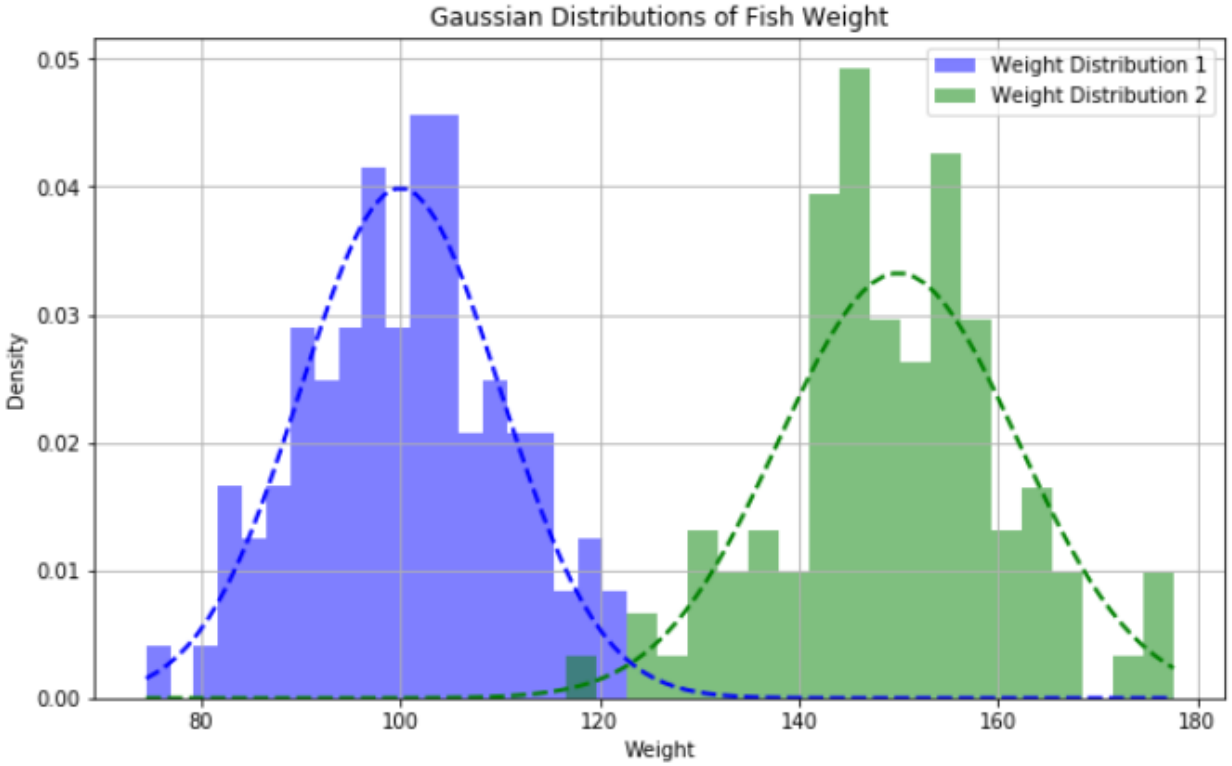
$$\bar{x} = (85 + 92 + 88 + 78 + 95 + 90 + 87 + 89 + 84 + 91) / 10 = 89.9$$

Using the formula for the sample mean, we find that the estimated mean weight score for the fishes is 89.9.

In many cases, it's important to assess the uncertainty associated with our parameter estimate. This can be done by calculating a confidence interval. For example, we might find that the 95% confidence interval for the population mean is [87.2, 92.6]. This means you are 95% confident that the true population mean falls within this interval. Based on our estimation and confidence interval, we can conclude that we estimate the mean weight score of the class to be approximately 89.9, with a high degree of confidence that it falls within the range of 87.2 to 92.6. In this example, parameter estimation involves using the sample mean as an estimator for the population mean.

Expectation Maximization

EM can be understood through a simple example involving a mixture of two Gaussian distributions. Imagine we have a dataset of fish weights, but we don't know the underlying distribution of the data. We suspect that the data is generated by a mixture of two Gaussian distributions but with unknown means and standard deviations. Our goal is to estimate these parameters.



Initially, we make some initial guesses for the means and standard deviations of the two Gaussian distributions. Let's call these initial guesses for the parameters as θ.

Expectation Step : Then, We calculate the probability that each data point belongs to each of the two Gaussian distributions based on the current estimates of the parameters (θ). This is done using the conditional probability formula from the Gaussian distribution.

For each data point xi and Gaussian distribution j, Calculate the probability that xi belongs to Gaussian j: P(j | xi, θ)

This step assigns a "responsibility" to each Gaussian distribution for each data point, indicating how likely it is that the data point came from that Gaussian.

Maximization Step : We update the estimates of the parameters (θ) based on the responsibilities calculated in the E-step. You do this to maximize the likelihood of the observed data under the current model. Calculate new estimates for the means and standard deviations of the two Gaussians.

Mean of Gaussian 1

$$\mu 1_{new} = \Sigma (P(1 \mid x_i, \theta) * x_i) / \Sigma P(1 \mid x_i, \theta)$$

Mean of Gaussian 2

$$\mu 2_{new} = \Sigma (P(2 \mid x_i, \theta) * x_i) / \Sigma P(2 \mid x_i, \theta)$$

Standard deviation of Gaussian 1:

$$\sigma 1_{new} = sqrt(\Sigma (P(1 \mid x_i, \theta) * (x_i - \mu 1_{new})^2) / \Sigma P(1 \mid x_i, \theta))$$

Standard deviation of Gaussian 2:

$$\sigma 2_{new} = sqrt(\Sigma (P(2 \mid x_i, \theta) * (x_i - \mu 2_{new})^2) / \Sigma P(2 \mid x_i, \theta))$$

Repeat the E-step and M-step until the estimates of the parameters (θ) converge, meaning they stop changing significantly.

Once the algorithm converges, we have estimates for the means and standard deviations of the two Gaussian distributions. We can now use these estimates to describe the underlying mixture model that generated our data. The EM algorithm iteratively refines the estimates of the model parameters by alternately computing the responsibilities of each data point for each component and then updating the parameters based on these responsibilities. It's a powerful technique for solving problems involving hidden or incomplete data, like clustering or density estimation.
Let’s discuss how it works with a simple Dataset,
[1.2, 1.4, 2.0, 3.5, 4.2, 5.1, 6.0, 6.5]

Initial Parameters

Initial mean for Gaussian 1 ($\mu 1$) = 2
Initial mean for Gaussian 2 ($\mu 2$) = 5
Initial standard deviation for Gaussian 1 ($\sigma 1$) = 1
Initial standard deviation for Gaussian 2 ($\sigma 2$) = 1.5
Equal initial probabilities for both Gaussians:

$$P(1) = P(2) = 0.5$$

Iteration 1 (E-step and M-step)

Calculate the responsibilities (probabilities that each data point belongs to each Gaussian distribution) based on the current parameter estimates. Use the Gaussian probability density function formula.

$$P(1 \mid x_i, \theta) = (1 / (\sigma_1 * \sqrt{(2\pi)})) * \exp(-(x_i-\mu_1)^2 / (2 * \sigma_1^2))$$
$$P(2 \mid x_i, \theta) = (1 / (\sigma_2 * \sqrt{(2\pi)})) * \exp(-(x_i - \mu_2)^2 / (2 * \sigma_2^2))$$

For each data point x_i, calculate these probabilities for both Gaussians.

Example:

For $x_i = 1.2$:

$P(1 \mid 1.2, \theta) = (1 / (1 * \sqrt{(2\pi)})) * \exp(-(1.2 - 2)^2 / (2 * 1^2)) = 0.24197$ (approx)

$P(2 \mid 1.2, \theta) = (1 / (1.5 * \sqrt{(2\pi)})) * \exp(-(1.2 - 5)^2 / (2 * 1.5^2)) = 0.00021$ (approx)

Calculate these probabilities for all data points. Update the parameter estimates based on the responsibilities calculated in the E-step.

Mean of Gaussian 1 ($\mu 1new$):

$\mu_{1new} = \Sigma (P(1 \mid x_i, \theta) * x_i) / \Sigma P(1 \mid x_i, \theta)$

Calculate this for all data points, and then take the average.

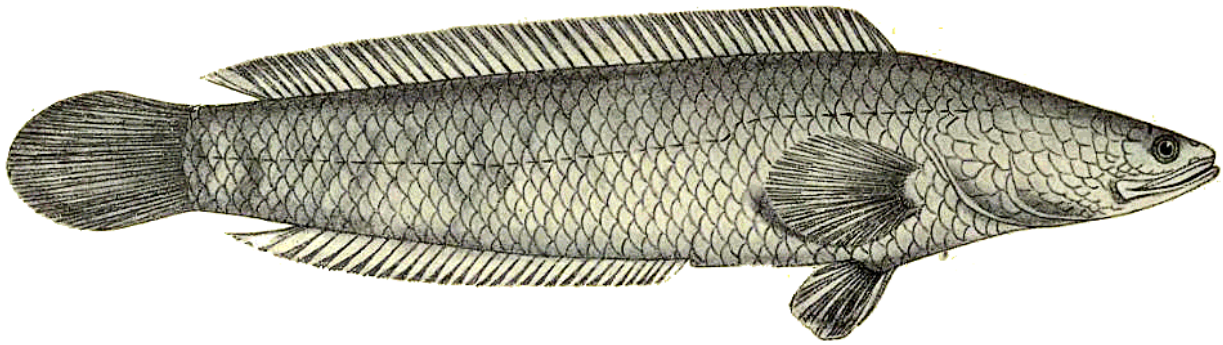
Mean of Gaussian 2 (μ_{2new}):
$\mu_{2new} = \Sigma (P(2 | x_i, \theta) * x_i) / \Sigma P(2 | x_i, \theta)$
Calculate this for all data points, and then take the average.

Standard deviation of Gaussian 1 (σ_{1new}):
$\sigma_{1new} = sqrt(\Sigma (P(1 | x_i, \theta) * (x_i - \mu_{1new})^2) / \Sigma P(1 | x_i, \theta))$
Calculate this for all data points, and then take the average.

Standard deviation of Gaussian 2 (σ_{2new}):
$\sigma_{2new} = sqrt(\Sigma (P(2 | x_i, \theta) * (x_i - \mu_{2new})^2) / \Sigma P(2 | x_i, \theta))$
Calculate this for all data points, and then take the average.

Update the probabilities for both Gaussians (P(1) and P(2)) based on the number of data points assigned to each Gaussian.

Repeat the E-step and M-step until the parameter estimates (μ1new, μ2new, σ1new, σ2new) and the probabilities (P(1) and P(2)) stop changing significantly. This convergence criterion helps determine when to stop iterating. The algorithm will converge to parameter estimates that best fit the given dataset with a mixture of two Gaussian distributions.



An Example : Image Segmentation

Image segmentation is a great example of how the Expectation-Maximization (EM) algorithm can be applied in practice. Image segmentation is the process of dividing an image into multiple regions or segments, where each segment corresponds to a meaningful part of the image, such as objects, boundaries, or regions of interest. EM can be used for image segmentation when the underlying distribution of pixel intensities in an image is complex and not easily modeled by a single Gaussian distribution.

Let’s imagine we have a grayscale image that we want to segment into different regions based on pixel intensity. Each region might correspond to a different object or feature in the image. We assume that the pixel intensities in each segment follow a Gaussian distribution, but there may be multiple Gaussian distributions because different objects or regions in the image may have different intensity characteristics. When we talk about pixel intensities in an image following a Gaussian distribution, we are making an assumption about how the brightness or gray levels of pixels are distributed within different regions of the image. The Gaussian distribution is a common statistical distribution that looks like a bell-shaped curve.

We assume the pixel intensities in each part of the image (each segment) are spread out in a way that resembles this bell-shaped curve. Now, why do we consider that there might be multiple Gaussian distributions? This is because different parts of an image can have different characteristics in terms of brightness or color. For example, in a photograph, the sky might be a certain shade of blue, the grass might be a different shade of green, and objects like cars or people may have their own distinct colors or brightness levels. So, instead of assuming that all the pixel intensities in the image follow a single Gaussian distribution, we consider the possibility that there could be several different Gaussian distributions, each corresponding to a different part of the image with its

own characteristic color or brightness. In other words, we allow for the idea that there are different "groups" or "clusters" of pixel intensities in the image, and each group follows its own Gaussian distribution. The Expectation-Maximization (EM) algorithm helps us identify these different groups or clusters of pixel intensities and estimate the parameters (mean and standard deviation) of the Gaussian distribution for each cluster. By doing this, we can segment the image into different regions, each associated with one of these Gaussian distributions. These regions often correspond to different objects or parts of the scene captured in the image.

EM Algorithm for Segmentation :

Initialize the parameters (mean and standard deviation) for each Gaussian component. These initial values can be arbitrary or based on some prior knowledge. For each pixel in the image, calculate the probability that it belongs to each Gaussian component based on the current estimates of the parameters. This step assigns a probability or "responsibility" to each Gaussian component for each pixel. Update the parameters (mean and standard deviation) for each Gaussian component based on the responsibilities calculated in the E-step. These updates aim to maximize the likelihood of the observed pixel intensities under the current model.

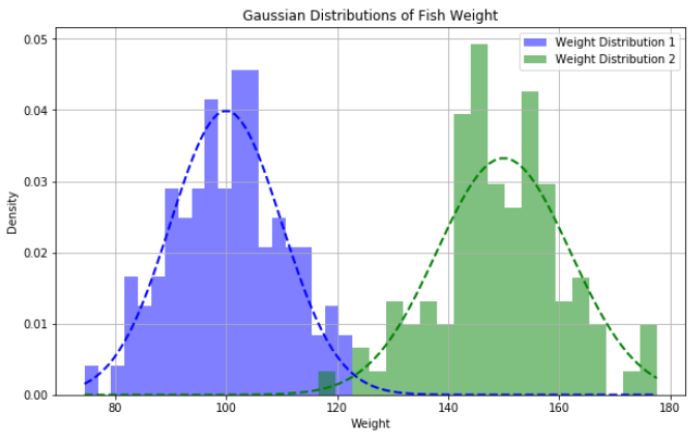
Continue iterating between the E-step and M-step until the estimates of the parameters converge. After the EM algorithm converges, we have estimates for the means and standard deviations of the Gaussian components. We can then use these parameters to assign each pixel to the Gaussian component with the highest responsibility. This assigns each pixel to a specific segment or region in the image. The result is a segmented image where each region corresponds to a different object or feature in the original image. EM allows us to handle cases where the pixel intensity distribution is not uniform across the image, making it a apt technique for image segmentation tasks.

The Gaussian Mixture Model

GMM, is a probabilistic model used for representing complex data distributions. It is particularly useful when dealing with data that doesn't fit a single simple distribution but appears to be a combination of multiple underlying distributions.

Mixture of Gaussians

A GMM assumes that the data is generated by a mixture of multiple Gaussian (normal) distributions. Each Gaussian component represents one of the underlying distributions in the data.



Parameters

A GMM is characterized by its parameters, which include the number of Gaussian components, the mean and standard deviation of each component, and the mixing coefficients (weights).

Probability Density Function (PDF)

The probability density function of a GMM is a weighted sum of the PDFs of its Gaussian components. It describes how likely each data point is to have been generated by the mixture of Gaussians.

Working of GMM

First, let’s initialize the parameters of the GMM. This includes specifying the number of Gaussian components (K), initializing the means (μ), standard

deviations (σ), and mixing coefficients (π) for each component.

Expectation-Maximization

GMMs are typically trained using the EM algorithm, similar to what we discussed earlier. The EM algorithm iteratively refines the parameter estimates based on the observed data.

Segmentation &Density Estimation:

Once the GMM is trained, it can be used for various purposes.

Density Estimation: GMMs provide a way to estimate the probability density of new data points. This is useful for tasks like anomaly detection or generative modeling.

Clustering: GMMs can be used for clustering data points into groups, where each group corresponds to one of the Gaussian components. Each group represents a cluster of similar data points.

K-Means Clustering and GMM

Gaussian Mixture Model (GMM) and K-Means clustering are both popular techniques for clustering data, but they have different characteristics and advantages. Here are some merits of GMM over K-Means clustering.

1. GMM is a probabilistic model that assigns a probability distribution (typically Gaussian) to each data point, allowing for soft clustering. This means that instead of assigning each point to a single cluster as in K-Means, GMM assigns probabilities to

each cluster, indicating the likelihood that a point belongs to each cluster. This can better capture complex data distributions where points may belong to multiple clusters to varying degrees.

2. K-Means assumes that clusters are spherical and equally sized, as it uses Euclidean distance to calculate cluster assignments. GMM, on the other hand, models clusters using Gaussian distributions, which can capture more flexible and complex cluster shapes, including ellipsoidal and unevenly sized clusters.

3. GMM can accommodate noisy data by modeling it as a combination of multiple Gaussian distributions. This allows it to handle data with varying levels of noise and uncertainty.

MERCI