INFO 510 Bayesian Modelling and Inference

Lecture 7 – Models in Bayesian inference; Gaussian Mixture Models

Dr. Kunal Arekar
College of Information
University of Arizona, Tucson

September 17, 2025

Overview of different models

Bayesian modeling isn't one-size-fits-all

Different models used, depending on the complexity of the situation and
the nature of the data.

- **Simple Bayesian Model:** Direct updating of beliefs with new data.
- **Conjugate Models:** Prior and data fit together like puzzle pieces, making updates easy.
- **Hierarchical Models:** Multiple layers of related data, balancing local and global information.
- **Non-Parametric Models:** Flexible models that grow with the data.
- **Latent Variable Models:** Inference about hidden variables from observable data.
- **Bayesian Networks:** Graphical representation of complex interdependencies.

Simple Bayesian Model

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

E.g., Deciding Whether to Bring an Umbrella when you step out

Email Spam Detection

Medical Diagnosis (Cancer)

Conjugate Models:

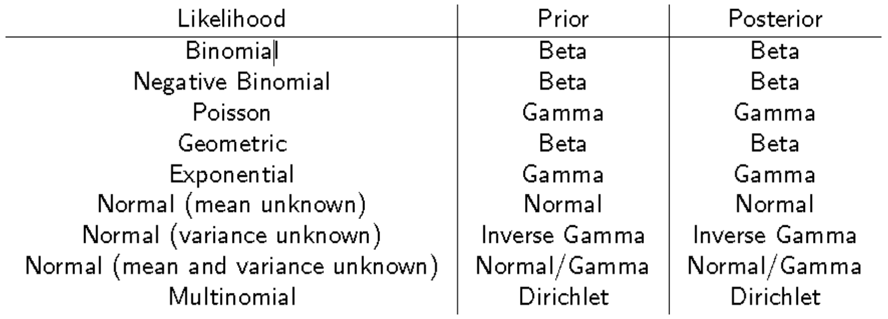
Prior and data fit together like puzzle pieces, making updates easy.

conjugate prior - a prior distribution when combined with a certain likelihood, produces a posterior distribution of the same family as the prior.

A conjugate prior ensures that the posterior distribution is in the same "family" of distributions as the prior.

If the prior and the posterior belong to the same parametric family, then the prior is said to be conjugate for the likelihood

For e.g., If prior distribution is ‘Beta’ then after the analysis the posterior probability will also be ‘Beta’ distribution



Likelihood	Prior	Posterior
Binomial	Beta	Beta
Negative Binomial	Beta	Beta
Poisson	Gamma	Gamma
Geometric	Beta	Beta
Exponential	Gamma	Gamma
Normal (mean unknown)	Normal	Normal
Normal (variance unknown)	Inverse Gamma	Inverse Gamma
Normal (mean and variance unknown)	Normal/Gamma	Normal/Gamma
Multinomial	Dirichlet	Dirichlet

Why Use Conjugate Priors?

Computational Simplicity – easy to compute
Conjugate priors allow us to avoid complicated integration to calculate posterior distribution by ensuring the posterior is easy to compute.

Closed-Form Solutions – you don’t need to rely on complex numerical methods to approximate the posterior solve the problem faster

Intuitive Prior-Posterior Updating –

E.g. parameters of the Beta distribution (α and β) update directly based on the data
So, if you observe more heads than tails, your Beta distribution shifts naturally toward higher values of θ (probability of heads).

Common Conjugate Priors in Practice

Conjugate priors depend on the type of likelihood you're working with.

- **Binomial Likelihood ↔ Beta Prior**

Used in scenarios with binary outcomes, like coin flips or success/failure data

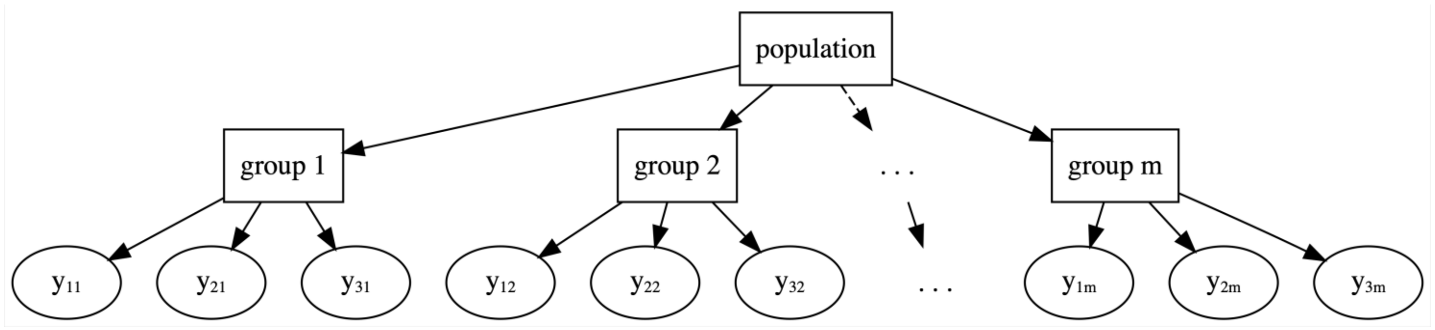
- **Normal Likelihood ↔ Normal Prior**

For problems involving continuous data where the likelihood is normally distributed

- **Poisson Likelihood ↔ Gamma Prior**

Used when you're counting events that happen over time (like the number of cars passing by per minute)

Hierarchical Models:



Multiple layers of related data, balancing local and global information

model complex, multi-layered structures in data

real-world problems often involve multiple levels of uncertainty and variation

Hierarchical models allow us share information between different groups or levels in the data in a structured way.

Hierarchical Models

Common mean

$$\begin{tikzcd}
 & m \arrow[dl] \arrow[d] \arrow[dr] & \\
Y_1 & Y_2 & Y_3
\end{tikzcd}$$

Hierarchical

$$\begin{tikzcd}
 & m \arrow[dl] \arrow[d] \arrow[dr] & \\
\theta_1 \arrow[d] & \theta_2 \arrow[d] & \theta_3 \arrow[d] \\
Y_1 & Y_2 & Y_3
\end{tikzcd}$$

Independent

$$\begin{tikzcd}
m_1 \arrow[d] & m_2 \arrow[d] & m_3 \arrow[d] \\
Y_1 & Y_2 & Y_3
\end{tikzcd}$$

- Hierarchical modeling is a statistical method used to handle data with multiple levels of variability or nested structures
- They are sometimes called multilevel models or mixed-effects models
- Hierarchical models are powerful because they:

Capture group structure: Allow you to understand the impact of groups (e.g., schools) on individual outcomes (e.g., student scores)

Increase statistical efficiency: They borrow strength from group-level data to help estimate parameters even when individual-level data is sparse

Model complex variability: Enable you to account for different sources of variation at multiple levels of the data

E.g.

A teacher looking at test scores from multiple classes in a school district.

Each class has its own average performance

district-wide factors that influence all classes

variability within each class (some students do better than others)

variability across classes (some teachers are more effective than others)

A **non-hierarchical model** might treat each class as independent, but they are not they come from same school district

Hierarchical Bayesian Models (multilevel models) pool information across different levels, borrowing strength from similar groups to make better estimates, particularly when data is sparse or noisy

Non-Parametric Models

Flexible models that grow with the data

Imagine you’re at a party, and you don’t know how many people will show up

You don’t want to limit yourself to a fixed number of chairs, so you keep adding chairs as people arrive

In non-parametric models, we don’t assume a fixed number of parameters in advance

the model grows in complexity as more data arrives

Commonly used non-parametric model

Dirichlet Process: The Core of Non-Parametric Bayesian Models

distribution over distributions

It allows us to model situations where we expect an unknown number of clusters or groups in our data.

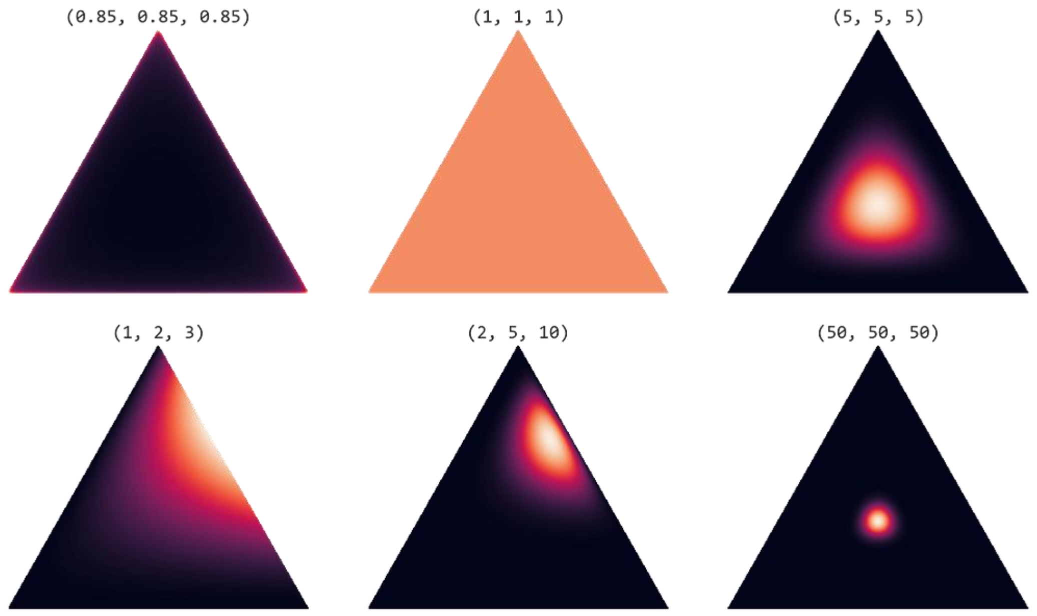
$$G \sim DP(a, G_0)$$

G_0 (Base distribution) - Represents the "center" distribution or prior guess for the data distribution

α (concentration parameter) - Determines how likely it is to create new clusters. A higher α means a greater chance of new clusters forming

Dirichlet process draws distributions around the base distribution the way a normal distribution draws real numbers around its mean

$$(G(A_1), G(A_2), \ldots, G(A_k)) \sim \mathrm{Dir}\,(\alpha G_0(A_1), \alpha G_0(A_2), \ldots, \alpha G_0(A_k))$$



Dirichlet distribution for different values of α

*parameter α governs the shapes of the distribution

*As the sum of all α increases, the distribution becomes more tightly concentrated around the centre

Other Non-Parametric Models

Model	Primary Use	Key Feature
Dirichlet Process (DP)	Clustering	Infinite mixture model
Pitman-Yor Process (PYP)	Power-law clustering	Rich-get-richer effect
Beta Process (BP)	Feature allocation	Binary feature matrices
Indian Buffet Process (IBP)	Feature allocation	Infinite latent features
Gaussian Process (GP)	Function approximation	Distribution over functions
Gamma Process	Modeling counts	Basis for DP via normalization
Normalized Random Measures (NRMs)	Flexible clustering	Generalization of DP
Hierarchical Dirichlet Process	Grouped data clustering	Shared clusters across groups
Kingman’s Coalescent	Population genetics	Genealogical trees
Dependent Dirichlet Process	Time/space-evolving data	Dependency structure over random measures

Advantages of Non-Parametric Bayesian Models

- **Flexibility:** The model grows with your data. You're not forced to decide on the number of parameters or clusters ahead of time

- **Adaptability:** These models can adapt to complex, evolving data structures, making them well-suited for real-world applications where the data isn't neatly divided into a fixed number of categories

- **Uncertainty Estimation:** Like all Bayesian models, non-parametric Bayesian models provide uncertainty estimates, which are crucial for making more informed decisions

Challenges

- **Inference is Hard:** Making predictions with non-parametric models often requires approximations or advanced algorithms like Markov Chain Monte Carlo (MCMC), which can be slow to compute.

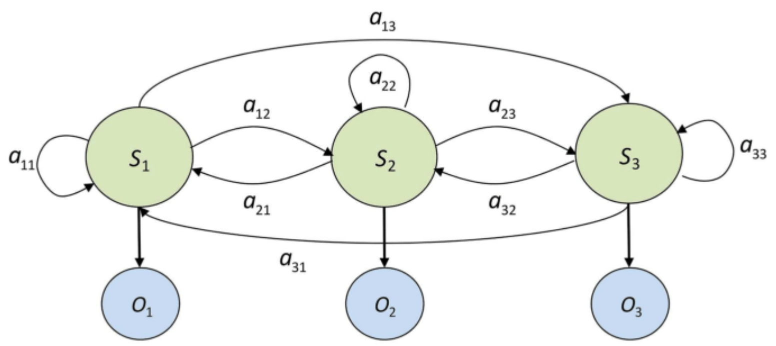
- **Interpretability:** The flexibility of these models can sometimes make them harder to interpret compared to simpler parametric models.

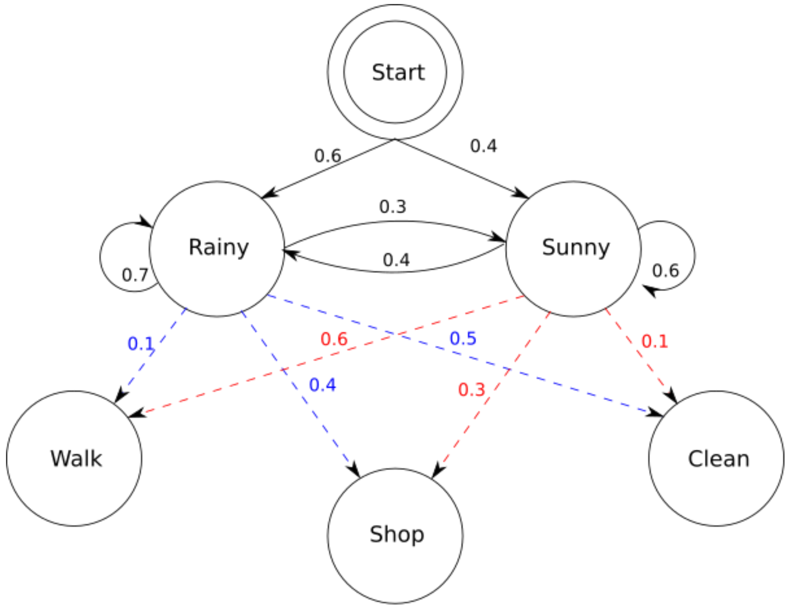
Latent Variable Models

Inference about hidden variables from observable data.

we can’t directly observe everything about a system, but we know there’s something "hidden" affecting what we do observe

E.g. Hidden Markov Model

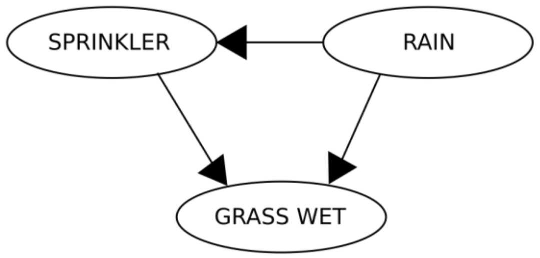




Bayesian Networks

Graphical model that represents a complex system of variables and their dependencies

	SPRINKLER	
RAIN	T	F
F	0.4	0.6
T	0.01	0.99



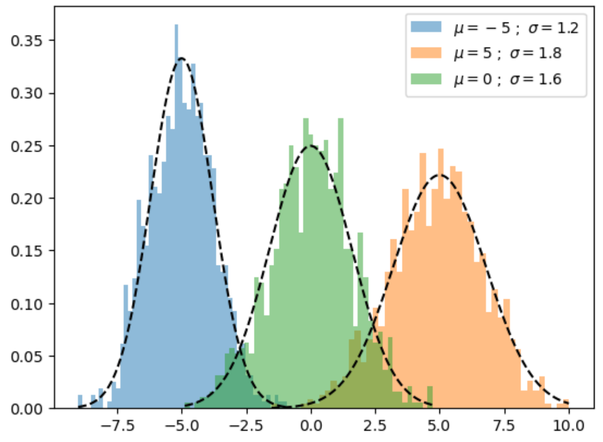
RAIN	
T	F
0.2	0.8

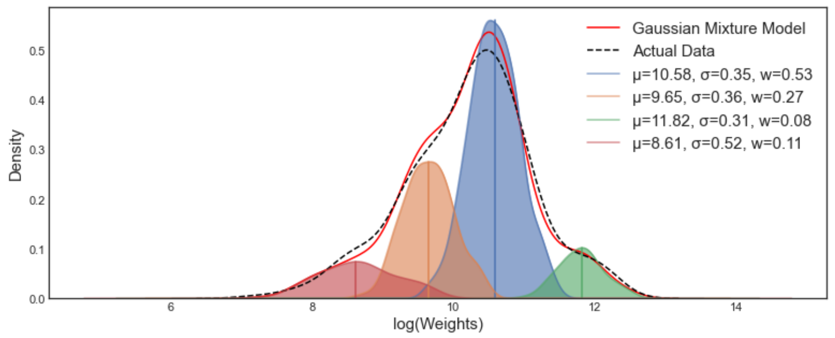
		GRASS WET	
SPRINKLER	RAIN	T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Gaussian Mixture Model (GMM)

A Gaussian Mixture Model (GMM) assumes that the data comes from a mixture of several Gaussian distributions, each representing a cluster.

The Gaussian distribution, commonly referred to as the normal distribution, is a fundamental probability distribution that is defined by two parameters: the mean (μ) and the variance (σ^2).





Mixture model

A mixture model assumes that the overall distribution of data isn’t generated by just one process but rather by a combination of several processes. In the case of GMMs, the data is assumed to be a combination (or "mixture") of several different Gaussian distributions.

Each Gaussian component in the mixture represents a different underlying process or subgroup within the data.

Mixture Weights/mixing coefficients

Each Gaussian in the mixture has a weight, which represents how much that Gaussian contributes to the overall data distribution. The weights of all Gaussians in the mixture must sum to 1.

These weights reflect the proportion of the data that is generated by each Gaussian.

Generative Models

- We model the joint distribution as,

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

- But in unsupervised clustering we do not have the class labels z.
- What can we do instead?

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

- This is a mixture model

It says:
the probability of observing a data point x is a weighted sum of the probabilities of x under each component (i.e., each value of z), weighted by how likely that component is (i.e., $p(z)$)

Most common mixture model: Gaussian mixture model (GMM)

- A GMM represents a **distribution** as

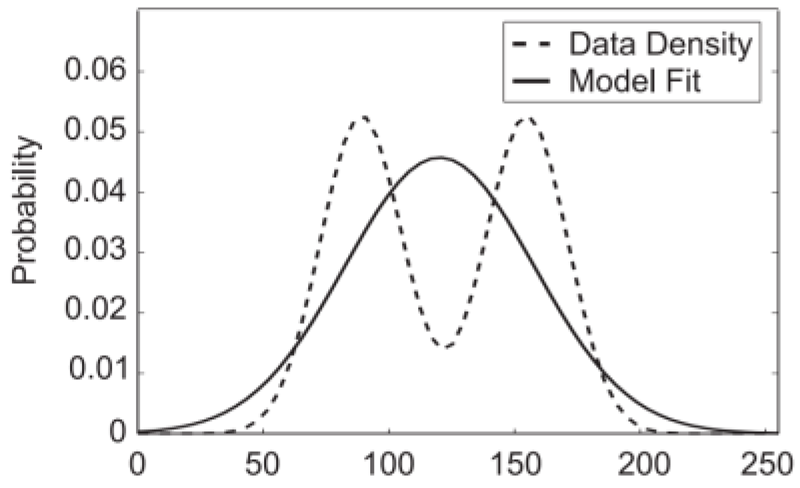
$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

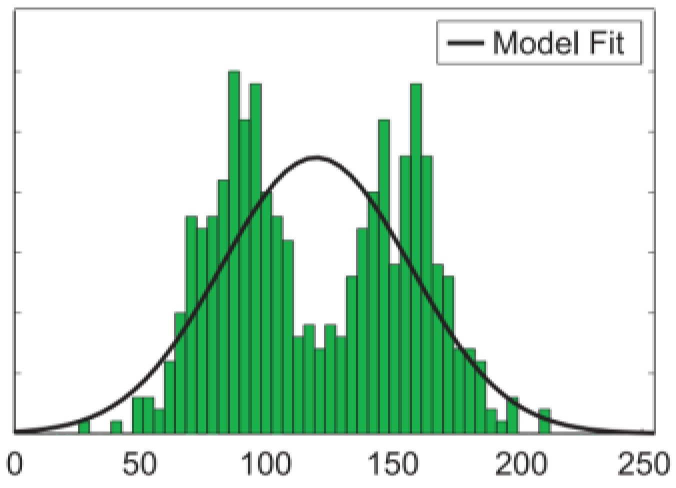
  with π_k the mixing coefficients, where:

$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{and} \quad \pi_k \geq 0 \quad \forall k$$

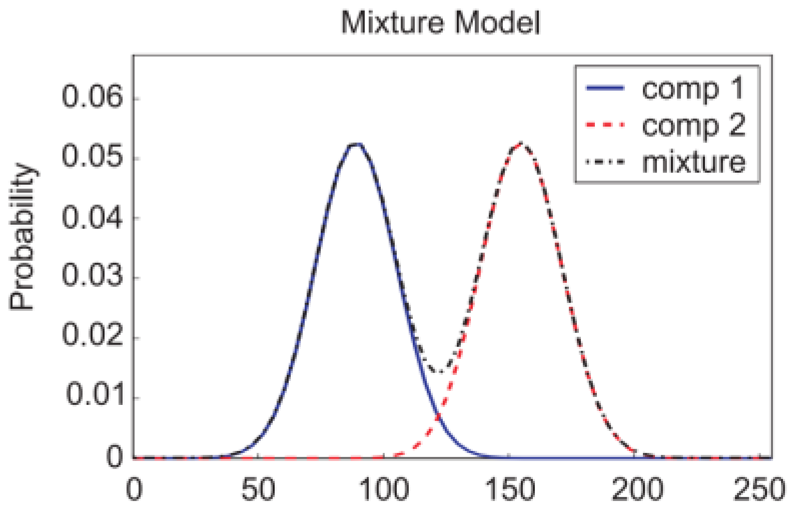
- GMM is a density estimator
- GMMs are **universal approximators of densities** (if you have enough Gaussians). Even diagonal GMMs are universal approximators.
- In general mixture models are very powerful, but harder to optimize

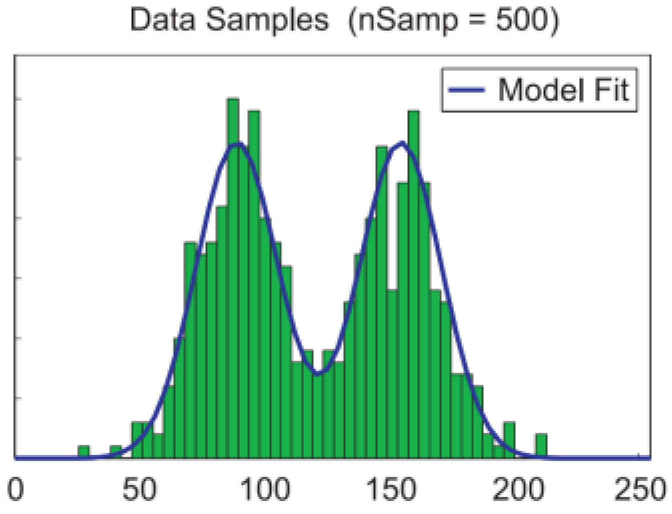
- If you fit a Gaussian to data:



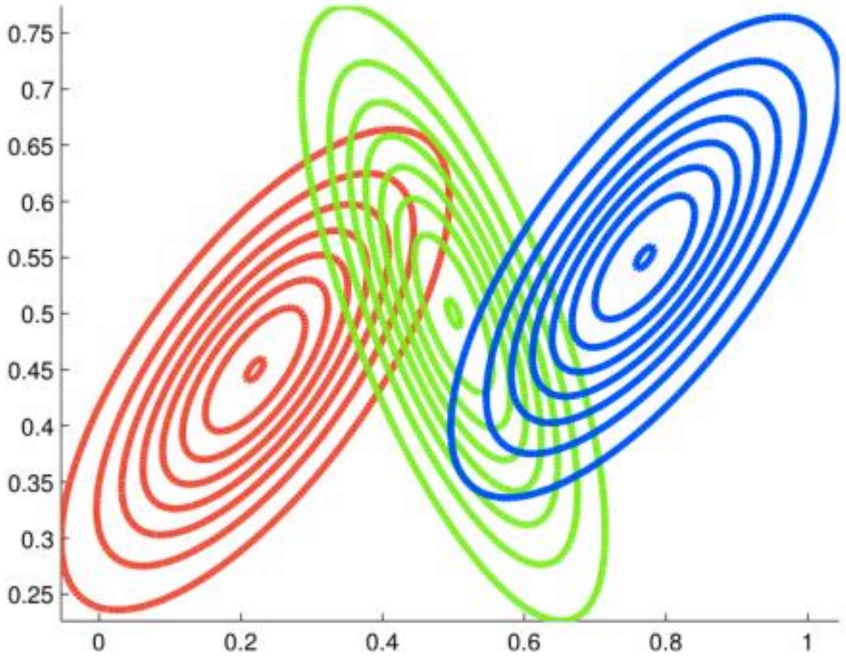


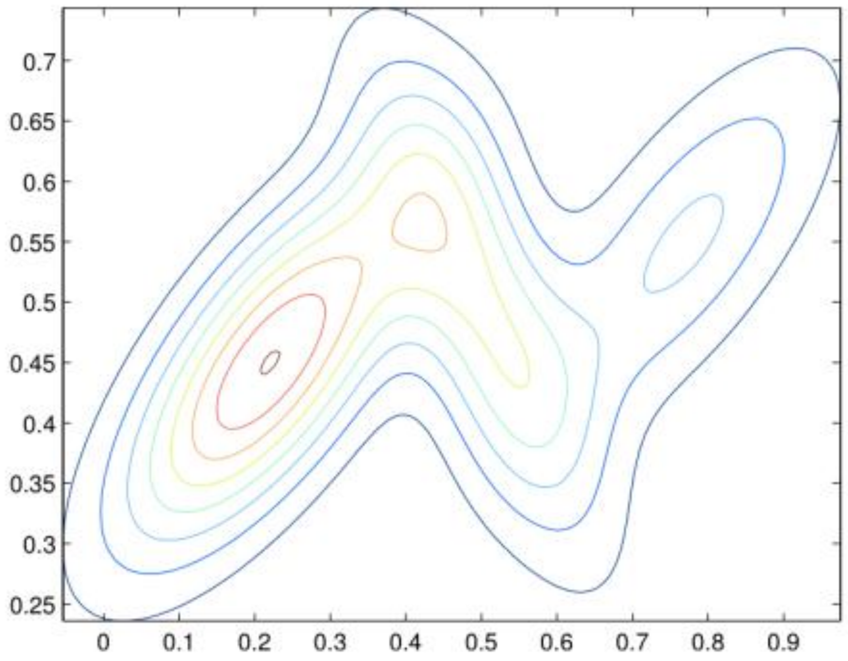
- Now, we are trying to fit a GMM (with $K = 2$ in this example):





Visualizing a Mixture of Gaussians – 2D Gaussians





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

x: This represents the observed data point. In practice, x is often a vector (i.e., it can have multiple features or dimensions).

K: The number of Gaussian components (or clusters). The model assumes that there are K different Gaussian distributions that are combined to generate the data.

πk: These are the mixing coefficients, also known as the prior probabilities for each Gaussian component. Each π_k represents the proportion of the total data that belongs to the k-th Gaussian component. They must satisfy the following conditions:

$$0 \le \pi_k \le 1$$

(each mixing coefficient is a probability)

(the total sum of the mixing coefficients must equal 1)

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

the Gaussian (or normal) distribution for the k-th component. It is parameterized by:

μk: the mean vector of the k-th Gaussian distribution

Σk: the covariance matrix of the k-th Gaussian distribution, which captures how spread out the data is and whether different dimensions of the data are correlated

What does this mean?

Imagine you have a set of points that were drawn from different bell curves (Gaussians) with different shapes (variances) and centers (means)

The GMM says, “Each data point x comes from a mixture of these different bell curves, and we don’t know which one exactly, but there’s some probability for each curve.”

Mixing Coefficients (πk) – how likely a data point is to come from each of the different Gaussians.

Gaussian Distributions (N(x|μk,Σk) –

> different shapes of the bell curves
> tall and narrow - data to be close to the mean;
> wide and spread out - capturing a larger variance in the data

The Sum – The GMM models the overall probability distribution as a weighted sum (mixture) of these individual Gaussian components

Steps in GMM Training

Expectation-Maximization (EM) Algorithm: To train a GMM, the Expectation-Maximization (EM) algorithm is typically used. This is an iterative process where:

- **E-step (Expectation):** You estimate the probability that each data point x belongs to each of the K Gaussian components.

  Based on the current parameters of the Gaussian distributions, assign probabilities to each data point. Updating our guess which Gaussian generated which data point

- **M-step (Maximization):** You update the parameters of the Gaussian components (means μk, covariances Σk, and mixing coefficients πk) based on those estimated probabilities.

  You repeat these steps until the model converges (i.e., when the parameters stop changing significantly).

Benefits of the EM Algorithm

Efficiency: While EM can be slow to converge, it’s relatively simple and straightforward.

Adaptability: EM works well even when the data is noisy, as it updates the parameters iteratively.

E-step (Expectation):

For each data point, calculate the probability that it belongs to each Gaussian

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

This $\boldsymbol{\gamma_{ik}}$ is called the responsibility—how responsible Gaussian k is for data point i

M-step (Maximization):

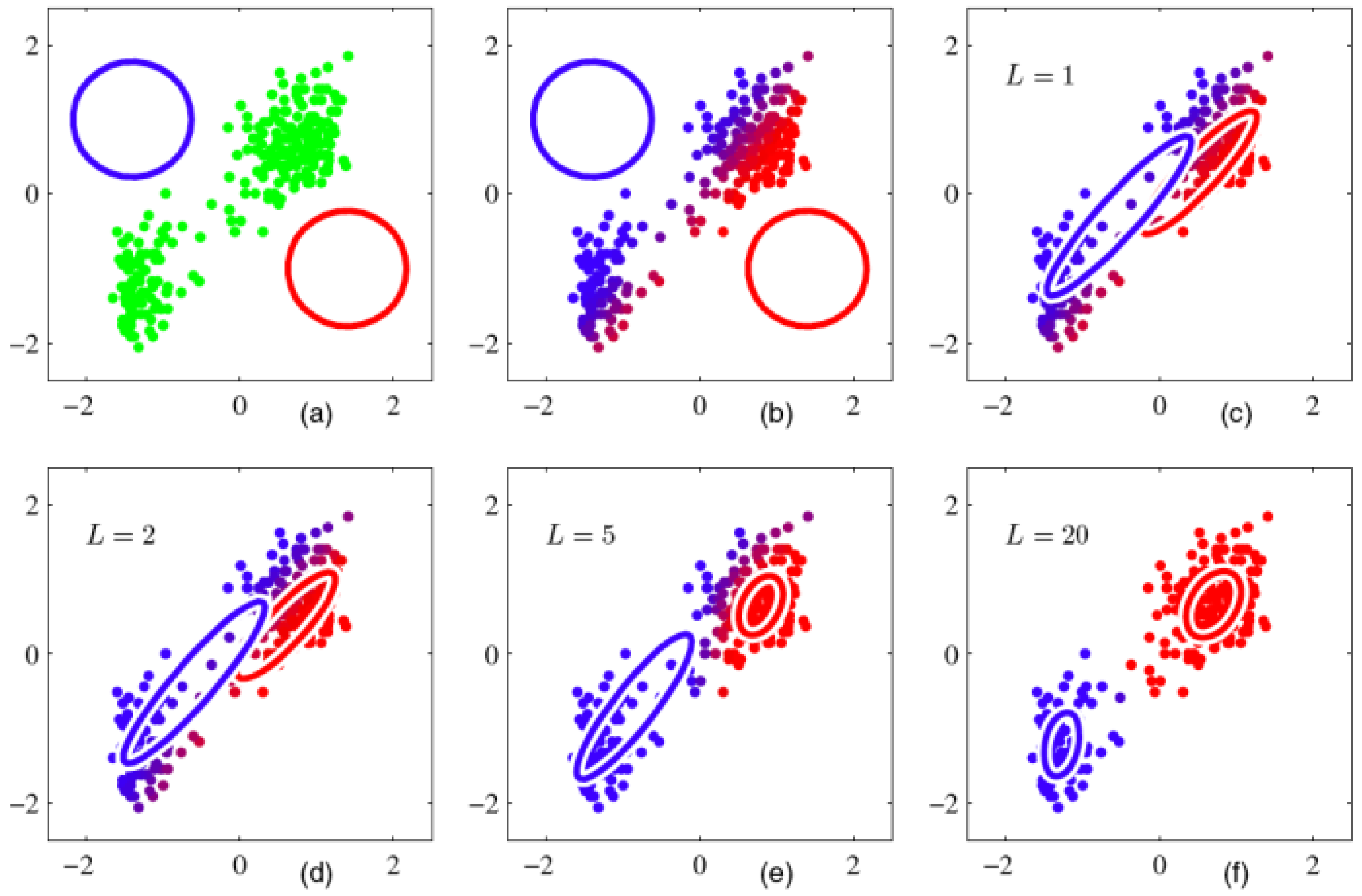
Re-estimate the parameters given current responsibilities

Updated mean
$$\mu_k = \frac{\sum_{i=1}^{N} \gamma_{ik}\, x_i}{\sum_{i=1}^{N} \gamma_{ik}}$$

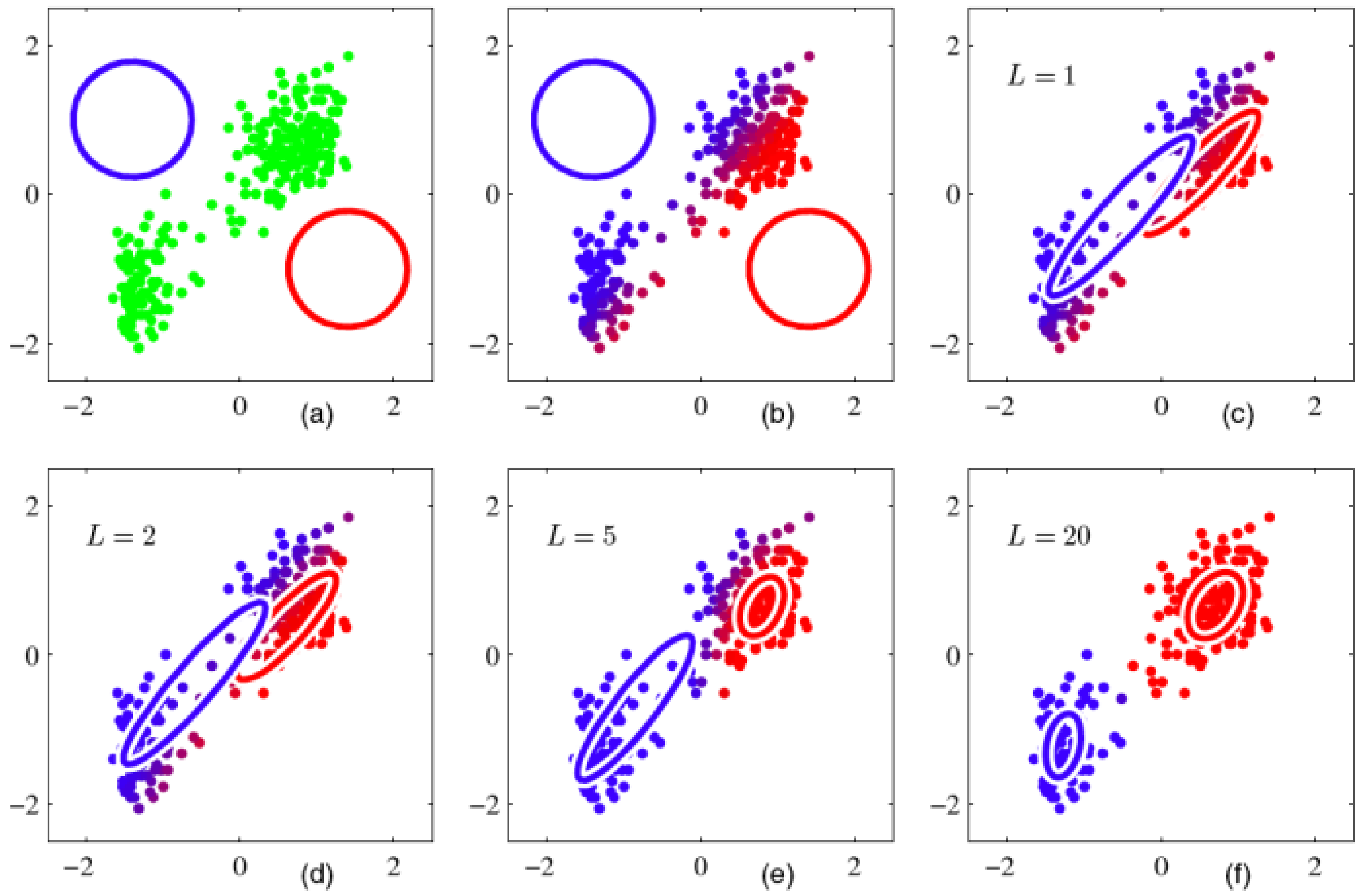
Updated covariance
$$\Sigma_k = \frac{\sum_{i=1}^{N} \gamma_{ik} (x_i - \mu_k)(x_i - \mu_k)^T}{\sum_{i=1}^{N} \gamma_{ik}}$$

Updated responsibilities
$$\pi_k = \frac{1}{N} \sum_{i=1}^{N} \gamma_{ik}$$

Evaluate log likelihood and check for convergence
$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$



2
0
−2
−2
0
2
(a)
(b)
(c)
(d)
(e)
(f)
$L = 1$
$L = 2$
$L = 5$
$L = 20$



(a)
(b)
(c)
(d)
(e)
(f)
$L = 1$
$L = 2$
$L = 5$
$L = 20$
−2
0
2

EM Algorithm for GMM

Initialize the means μ_k, covariances Σ_k and mixing coefficients π_k

Iterate until convergence:

- E-step: Evaluate the responsibilities given current parameters

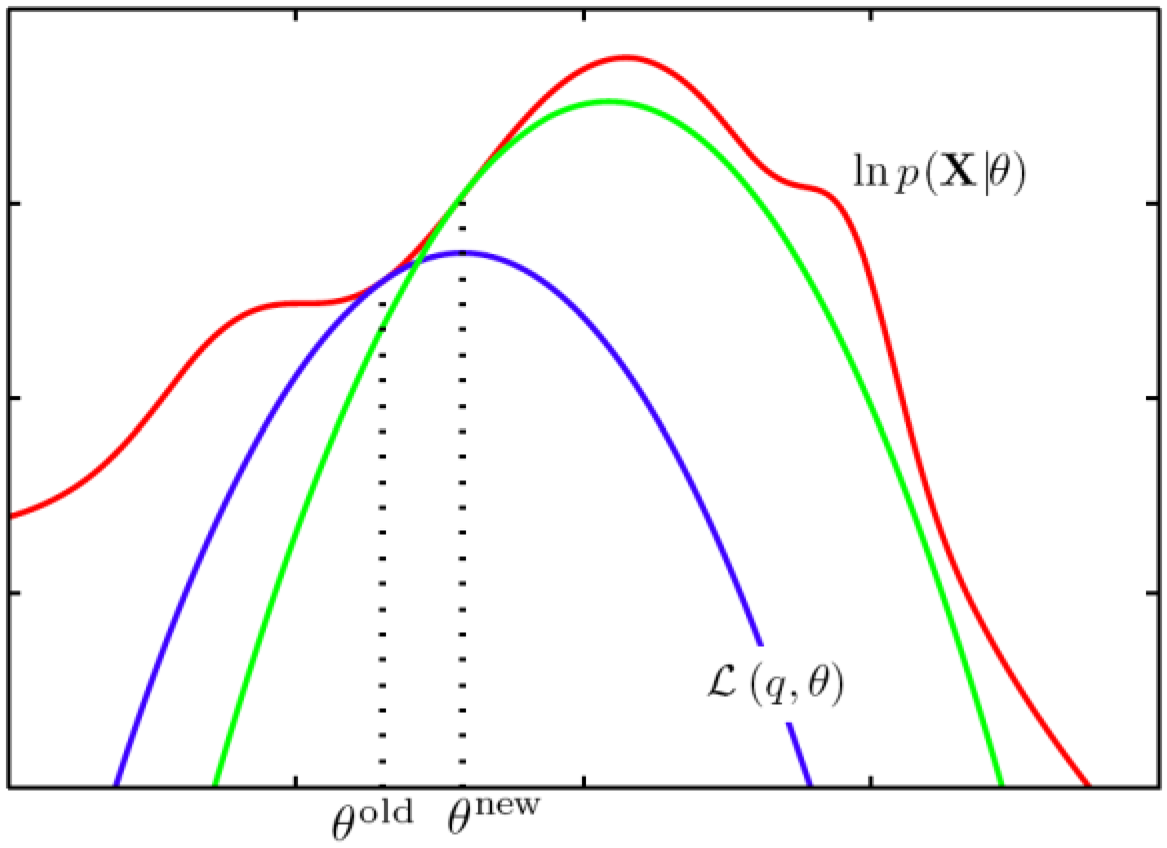
$$\gamma_k^{(n)} = p(z^{(n)}|\mathbf{x}) = \frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}^{(n)}|\mu_j, \Sigma_j)}$$

- M-step: Re-estimate the parameters given current responsibilities

$$\begin{aligned}
\mu_k &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} \mathbf{x}^{(n)} \\
\Sigma_k &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T \\
\pi_k &= \frac{N_k}{N} \quad \text{with} \quad N_k = \sum_{n=1}^{N} \gamma_k^{(n)}
\end{aligned}$$

- Evaluate log likelihood and check for convergence

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



The EM algorithm involves alternately computing a lower bound on the log likelihood for the current parameter values and then maximizing this bound to obtain the new parameter values.

Relation to k-Means

The K-Means Algorithm:

1. Assignment step - Assign each data point to the closest cluster
2. Refitting step - Move each cluster center to the center of gravity of the data assigned to it

The EM Algorithm:

1. E-step - Compute the posterior probability over z given our current Model
2. M-step - Maximize the probability that it would generate the data it is currently responsible for.

Why Gaussian Mixture Models?

Versatility: In contrast to a single Gaussian (which assumes data follows one smooth curve), GMMs allow us to model situations where data is generated from multiple sources.

Clustering: GMMs naturally lead to clustering, which helps in finding groups in the data. E.g., identify different species of plants from height and leaf size data.

Soft Clustering: Unlike some clustering algorithms that assign each point to a single group, GMMs allow for uncertainty. A point might belong to multiple clusters, with some probability assigned to each. It’s like saying, "This plant is likely 70% of Species A, but there's a 30% chance it's of Species B."

Why GMMs Are Important in Bayesian Modeling

Flexible Priors: GMMs provide flexible, multi-modal priors that capture complex data distributions.

Likelihoods: They serve as powerful likelihood models, especially when data comes from multiple underlying processes.

Bayesian Parameter Estimation: GMMs allow for posterior distributions over parameters, integrating uncertainty into the model.

Latent Variables: GMMs naturally introduce latent variables, making them ideal for modeling hidden structures in the data.

Model Comparison: Bayesian techniques help select the appropriate number of components in GMMs, improving model fit.

Nonparametric Extensions: Bayesian nonparametrics, like Dirichlet Process GMMs, allow for models that adapt in complexity as more data becomes available.

Challenges with GMMs

Choosing the Right Number of Components:
How many Gaussians should we use to model the data? If we pick too few, we may miss important patterns. If we pick too many, we may overfit the data, meaning we model noise rather than actual structure.

Local Optima in EM: EM can get stuck in local optima, meaning it may not find the best solution to fitting the data, just a “good enough” one.

Assumption of Gaussianity: GMM assumes that the underlying distributions are Gaussian (bell-shaped). However, real-world data might not always fit this assumption.

Applications of Gaussian Mixture Models

Clustering: GMMs are widely used for clustering data when you expect that the data might come from different groups (e.g., customer segmentation, image segmentation)

Density Estimation: GMMs can model the underlying distribution of data, making them useful for estimating the probability of certain events occurring

Anomaly Detection: By modeling the "normal" data distribution with GMMs, any data points that fall far from the fitted distribution can be flagged as anomalies (e.g., fraud detection)

Speech and Image Recognition: GMMs are used in recognizing patterns in audio and visual data by modeling the underlying distribution of features in these domains