Gaussian Mixture Models
&
Latent Variable Models

Learning Paradigms

- Supervised learning – Learning with a teacher
  - The dataset has labels – desired output is known
  - We can train the model till we get good results
  - Eg: Predict stock market price, classify email as spam or not
- Unsupervised learning – Learning without a teacher
  - The dataset is not labeled – do not know what results we are looking for
  - The algorithm has to figure out the pattern in the data
  - Eg. Clustering – group customers by purchasing behavior
- Reinforcement Learning - Learning with a critic
  - Reward or penalize the actions
  - Game playing and control applications

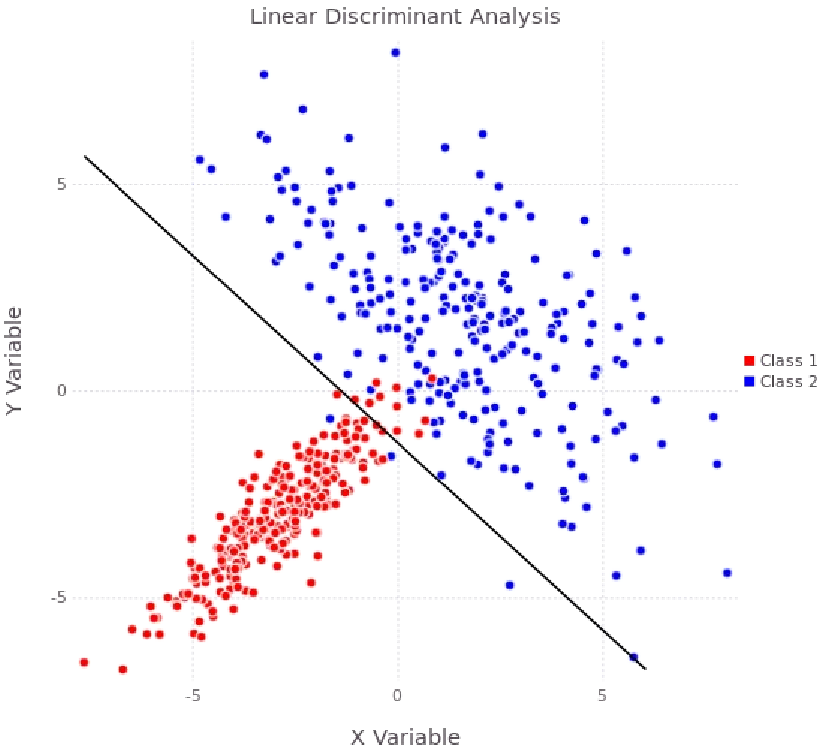
Supervised Learning Tasks

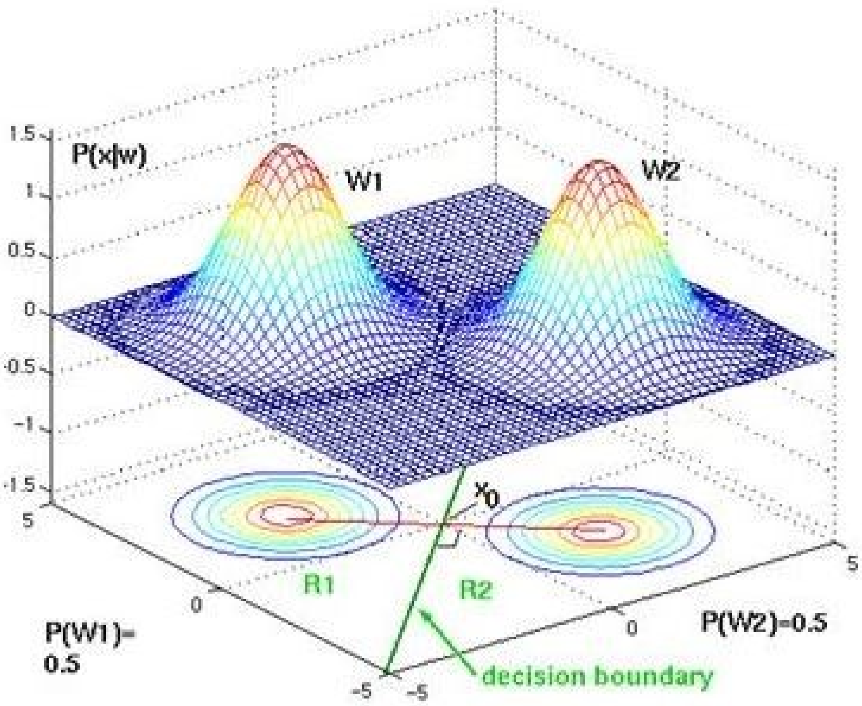
- Regression
  - Map input features to continuous output variables
  - Estimate weight of a person from his height
  - Predict stock-market price from past data
  - Create 3-D image of a person from his 2-D photograph
- Classification
  - Map input features to discrete/categorical output variables
  - Given the mammogram, predict whether the cancer is benign or malignant
  - Given the news article, check whether it is real or fake
  - Given the face image, verify whether it is the rightful owner or not

Pattern Classification

- Discriminant functions
  - Aims to learn the boundary that separates the classes
  - Linear discriminants, Fisher discriminants, perceptron, SVM & logistic regression
  - Criticized for being driven by boundary points, rather than structure in the data
- Statistical approaches
  - Aims to estimate the posterior probability of a class given observed data
  - Prior information can be easily incorporated

Discriminant vs Statistical

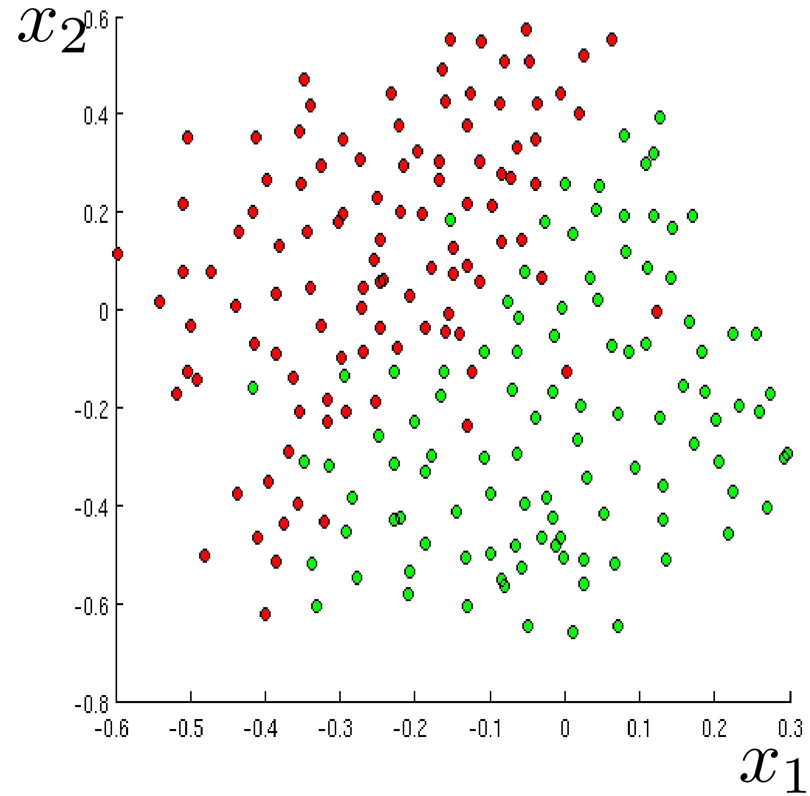




Pattern Classification

$$\overset{\text{Posterior}}{p(C_k/\mathbf{x})} = \frac{\overset{\text{Prior}}{p(C_k)} \ \ \overset{\text{Likelihood}}{p(\mathbf{x}/C_k)}}{p(\mathbf{x})}$$

- Generative Models
  - Estimate joint probability
- Discriminative Models
  - Estimate posterior probability



$$C^* = \arg\max_k p(C_k/\mathbf{x})$$

Generative vs Discriminative Models

- Estimate $p(\mathbf{x}, c_k)$
- Models the structure in the data
- Statistical: GMM, HMM
- Neural Net: Autoencoders, RBM
- Unlablled data can be used
- Captures variability in the data
- Priors and penalties can be controlled precisely
- Difficult to train
- Slightly inferior performance

- Estimate $p(C_k/\mathbf{x})$
- Captures the discriminating features
- Logistic Regression, CRF
- MLP, DNN, CNN
- Need lot of labeled data
- Criticized as blackbox approach
- Lacks elegance
- Priors, alternate penalty
- Easy to train
- Superior performance

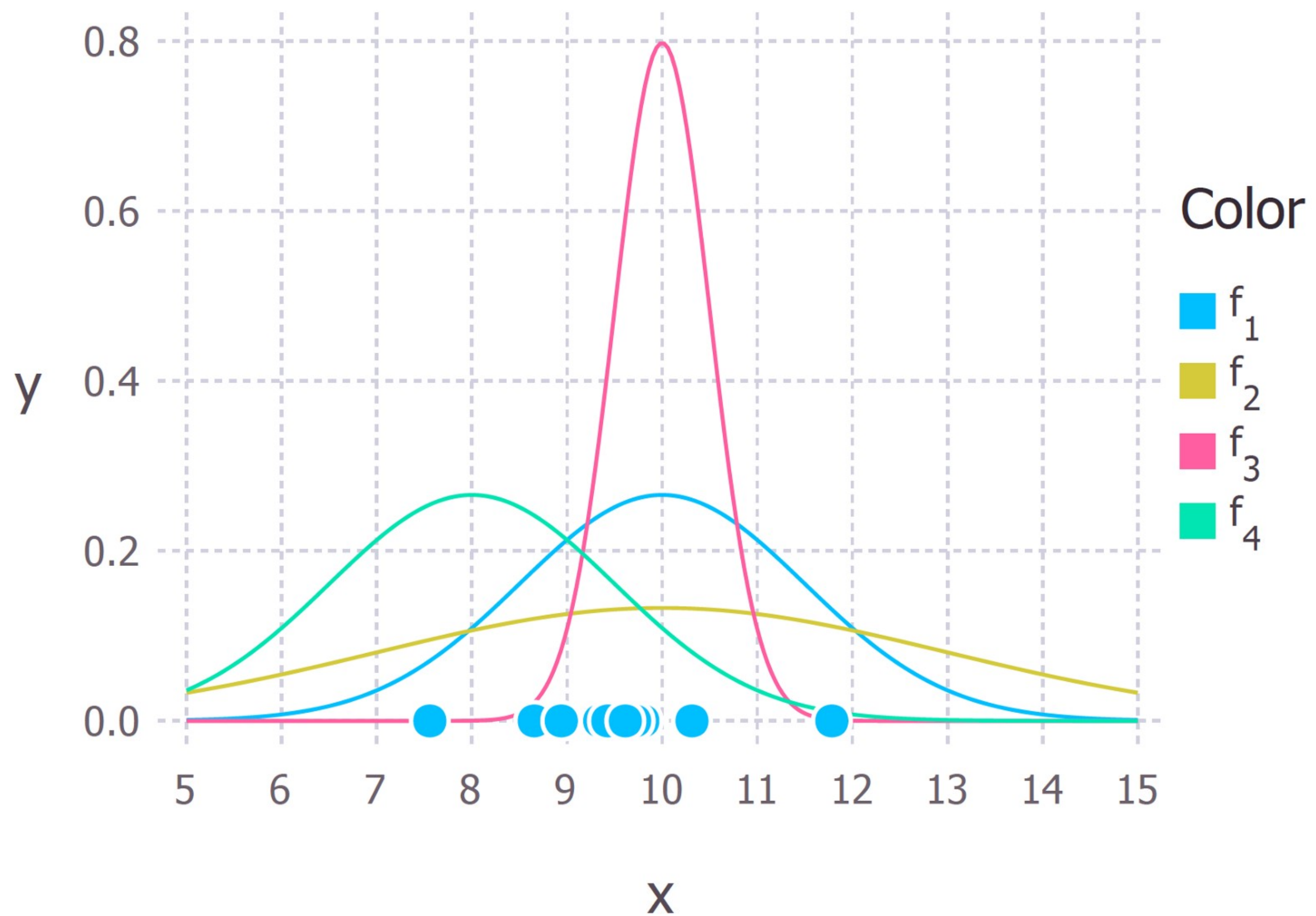
Generative Modeling

- Need to estimate either joint density or class conditional density
- Assume that the dataset X is generated from an parametric probability density function
- Estimate the parameters of the probability density function
- Gaussian density function can be parameterized by mean and variance

Maximum Likelihood Estimation

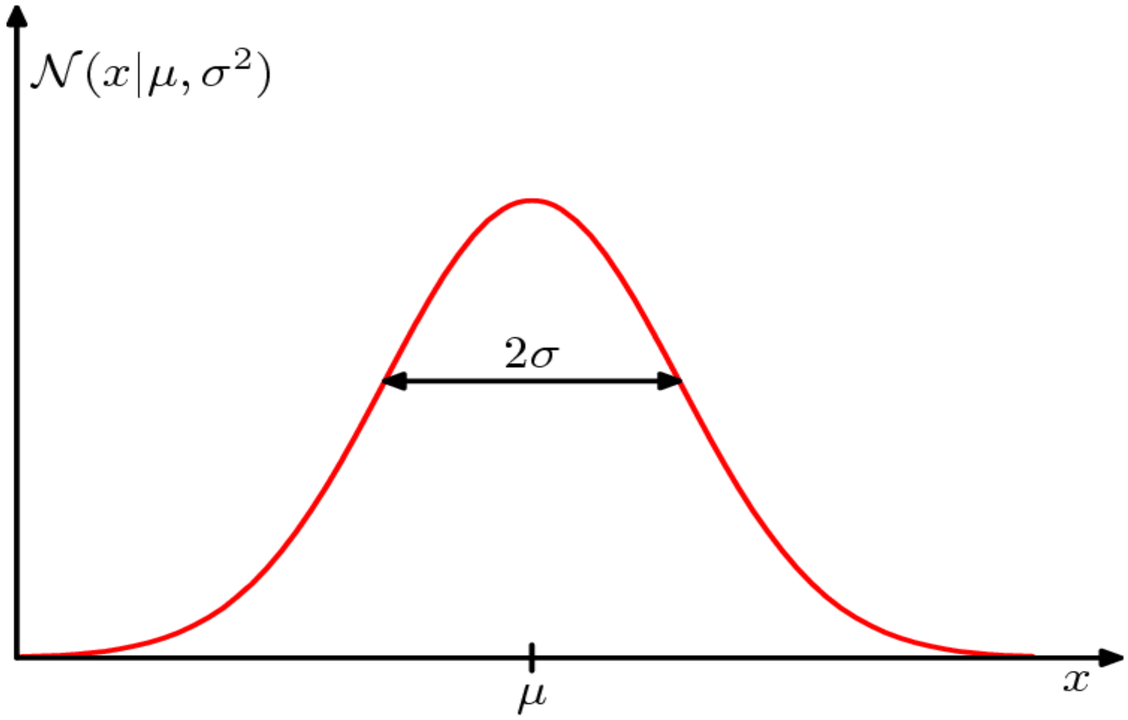
- Maximize the likelihood formulation
- Setting the gradient of the complete data likelihood to zero we can find the closed form solution.
- Since log() is a monotonic function, we can maximize the logarithm of likelihood
- Domonstrate ML estimate in the context of Bernoulli experiment

ML Estimate for Gaussian



1-D Gaussian

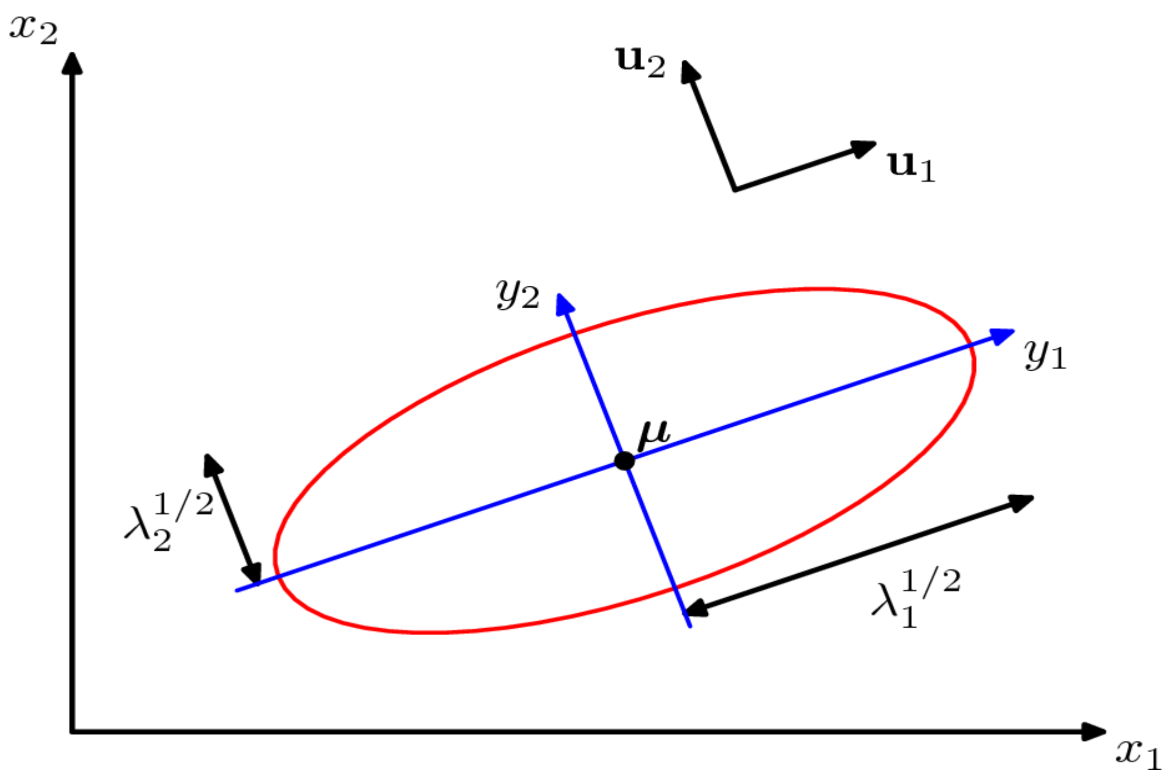
$$\text{Normal}(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ - \frac{1}{2\sigma^2}(x - \mu)^2 \right\}$$



Parameters to be estimated are the mean (μ) and variance (σ)

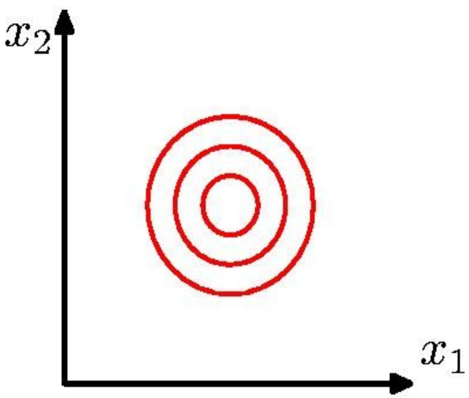
Multivariate Gaussian

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi|\boldsymbol{\Sigma}|)^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$



Multivariate Gaussian

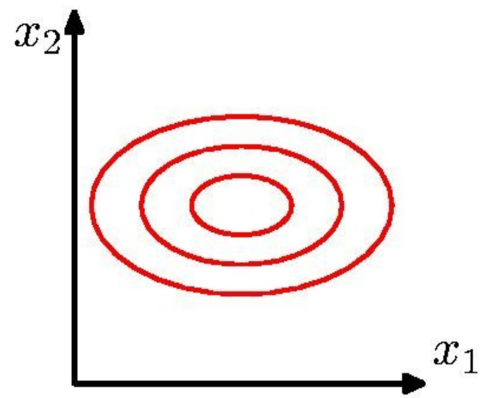
- Spherical



(c)

$$\Sigma = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix}$$

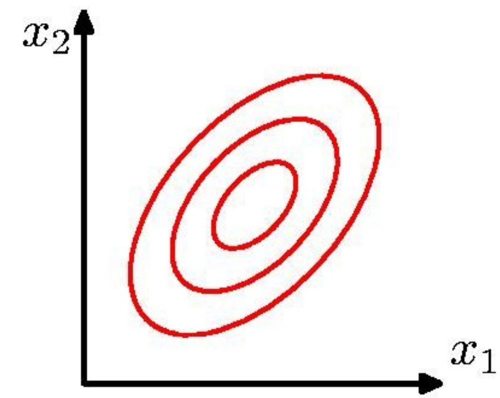
- Diagonal



(b)

$$\Sigma = \begin{pmatrix} \sigma_1{}^2 & 0 \\ 0 & \sigma_2{}^2 \end{pmatrix}$$

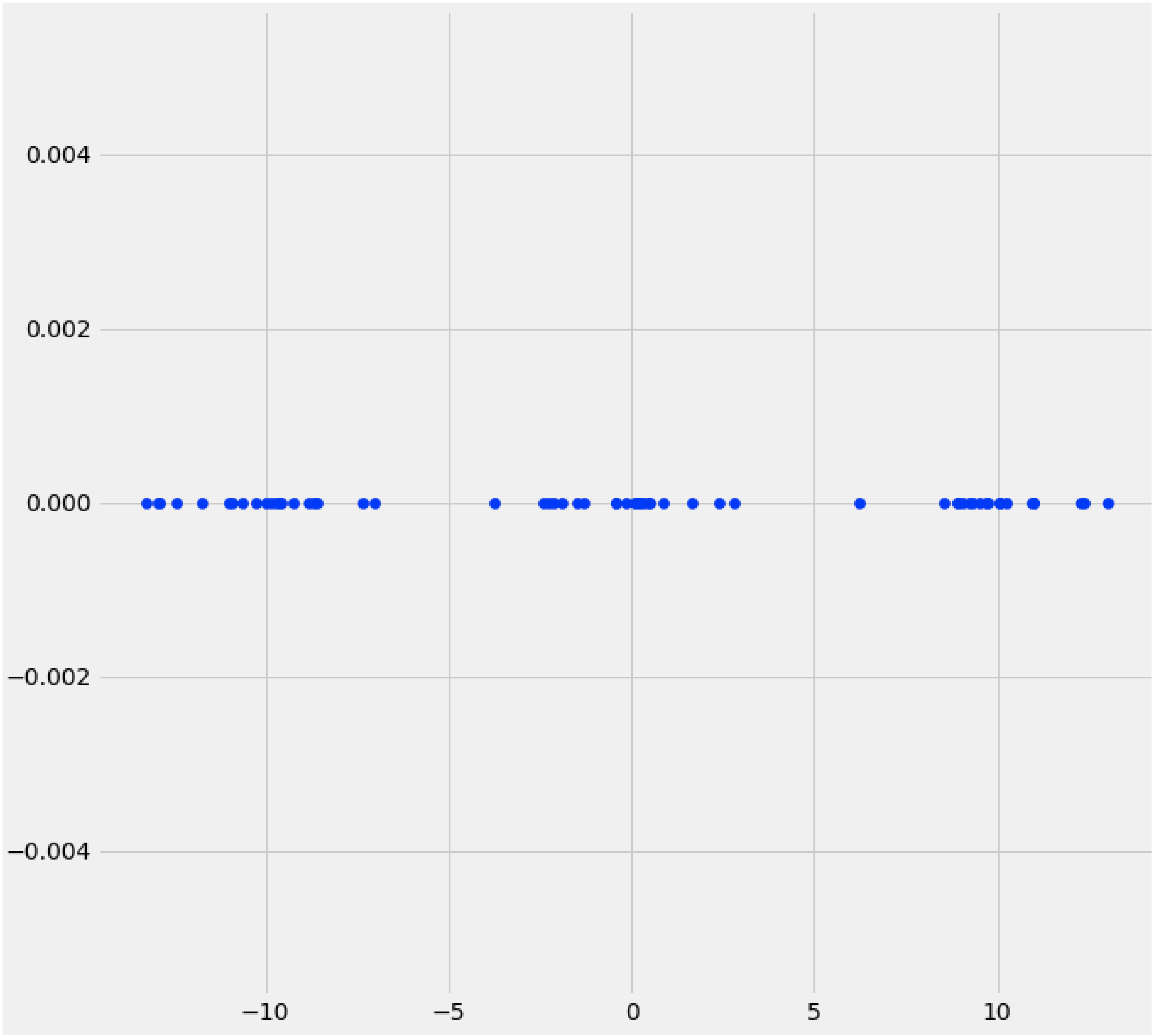
- Full



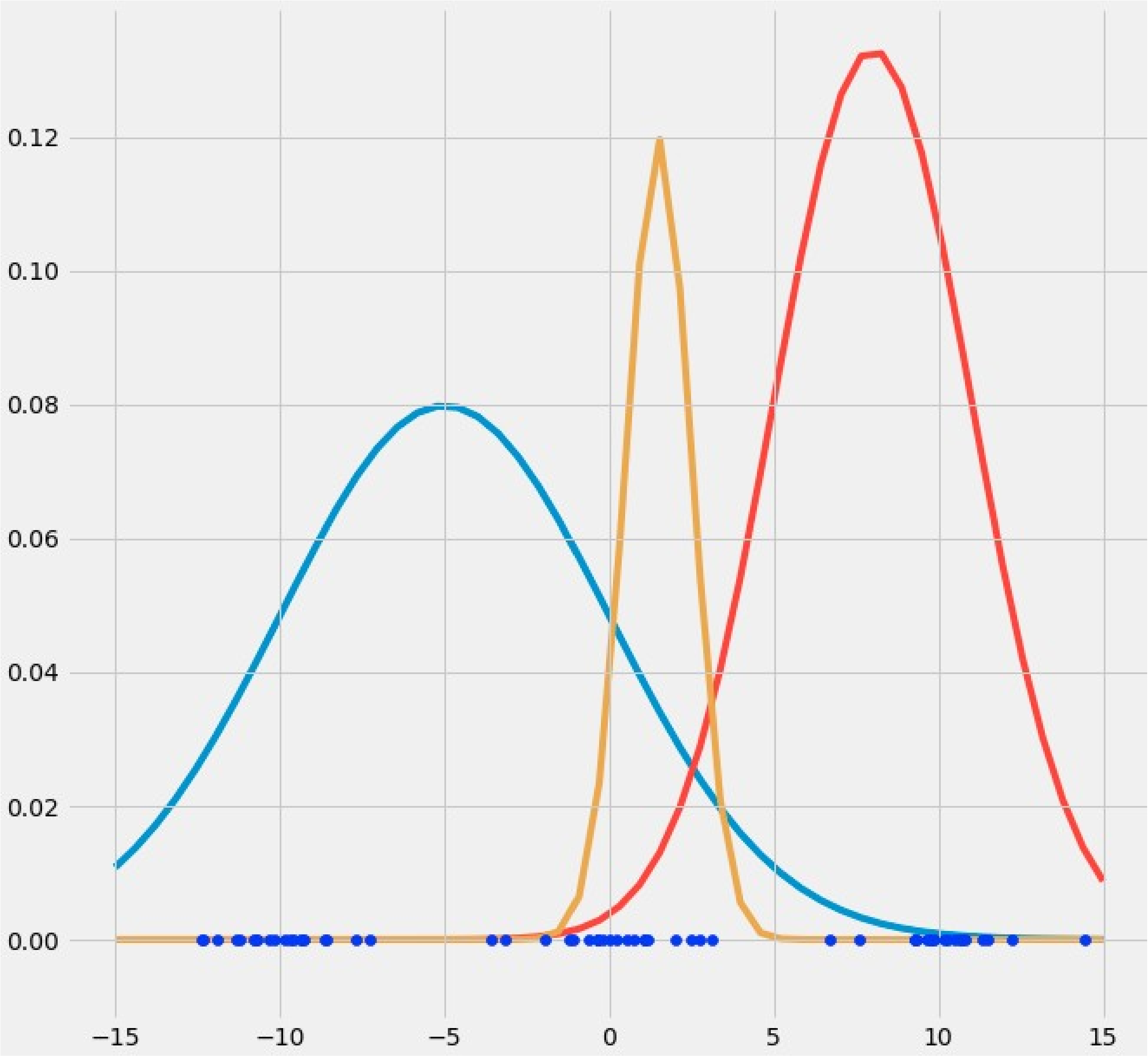
(a)

$$\Sigma = \begin{pmatrix} \sigma_{11}{}^2 & \sigma_{12}{}^2 \\ \sigma_{12}{}^2 & \sigma_{22}{}^2 \end{pmatrix}$$

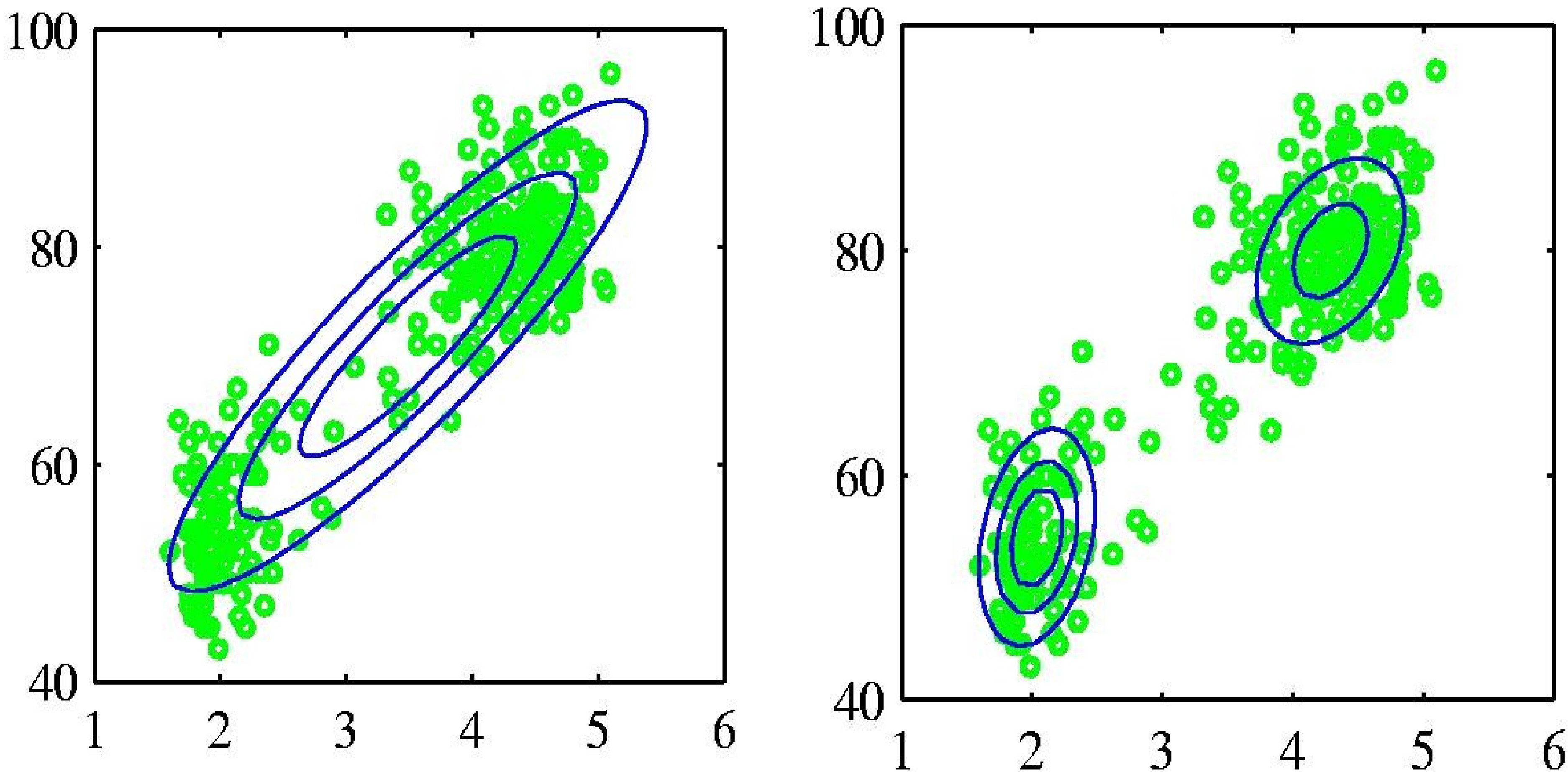
When One Gaussian Is Not Enough



When one Gaussian is not enough



2-D Case



Real world datasets are rarely unimodal!

Gaussian Mixtures

- Linear super-position of Gaussians

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

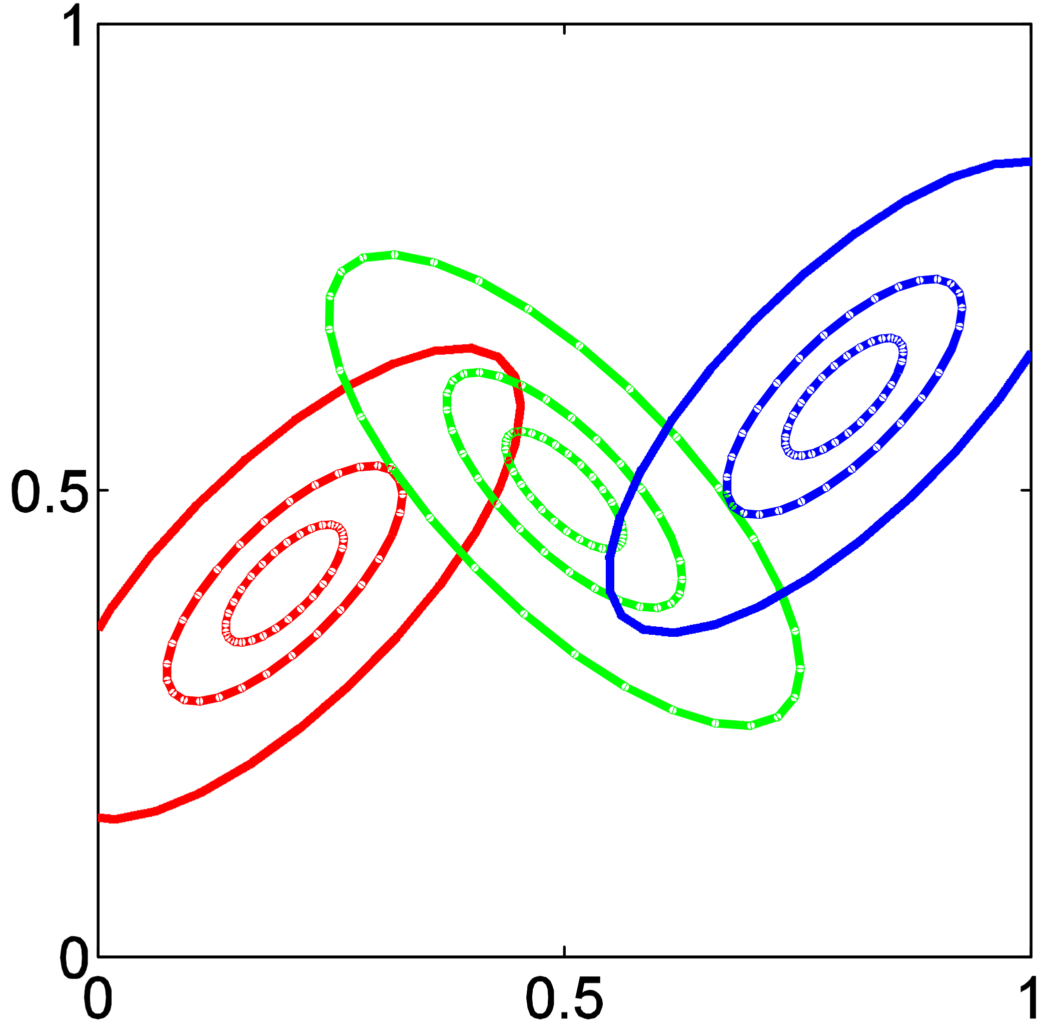
- Normalization and positivity require

$$\sum_{k=1}^{K} \pi_k = 1 \qquad 0 \leqslant \pi_k \leqslant 1$$

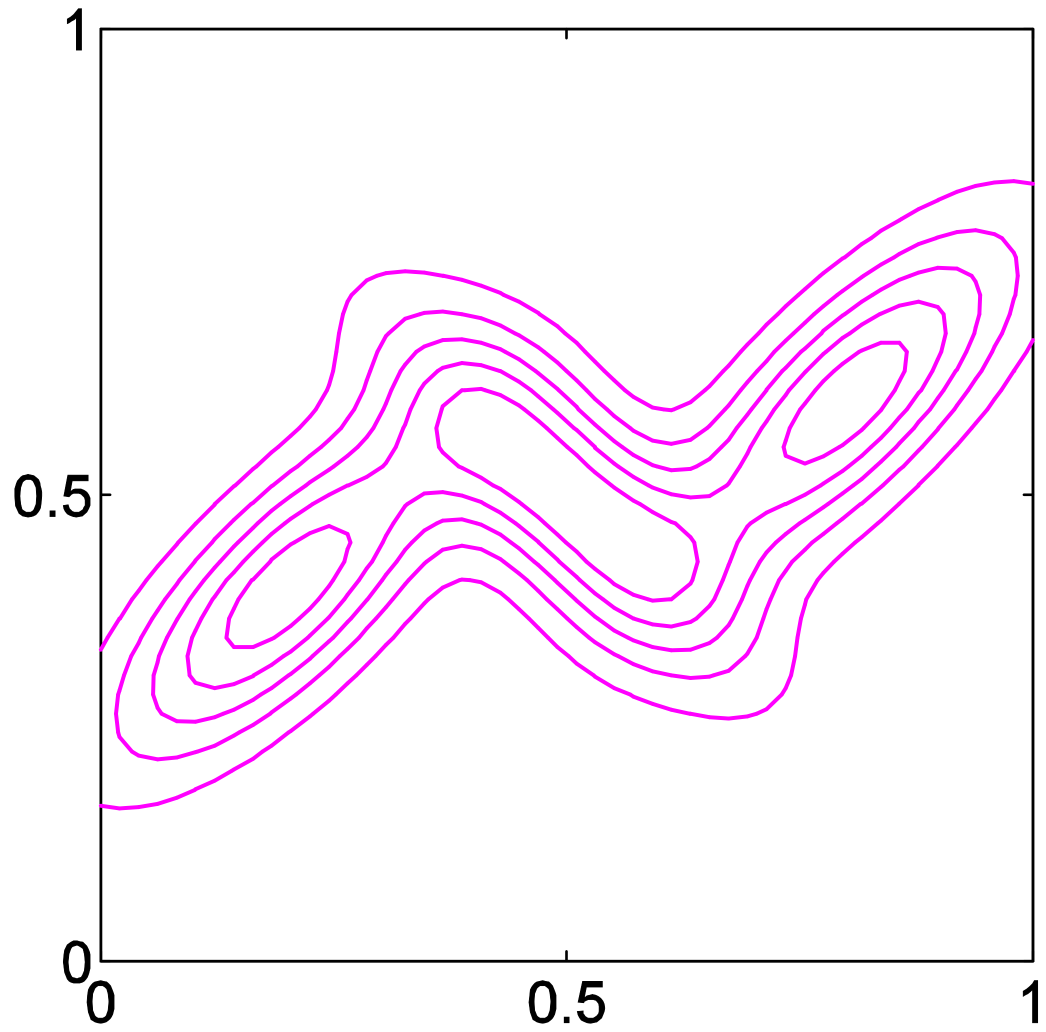
- Can inter[illegible]nts as prior probabilities

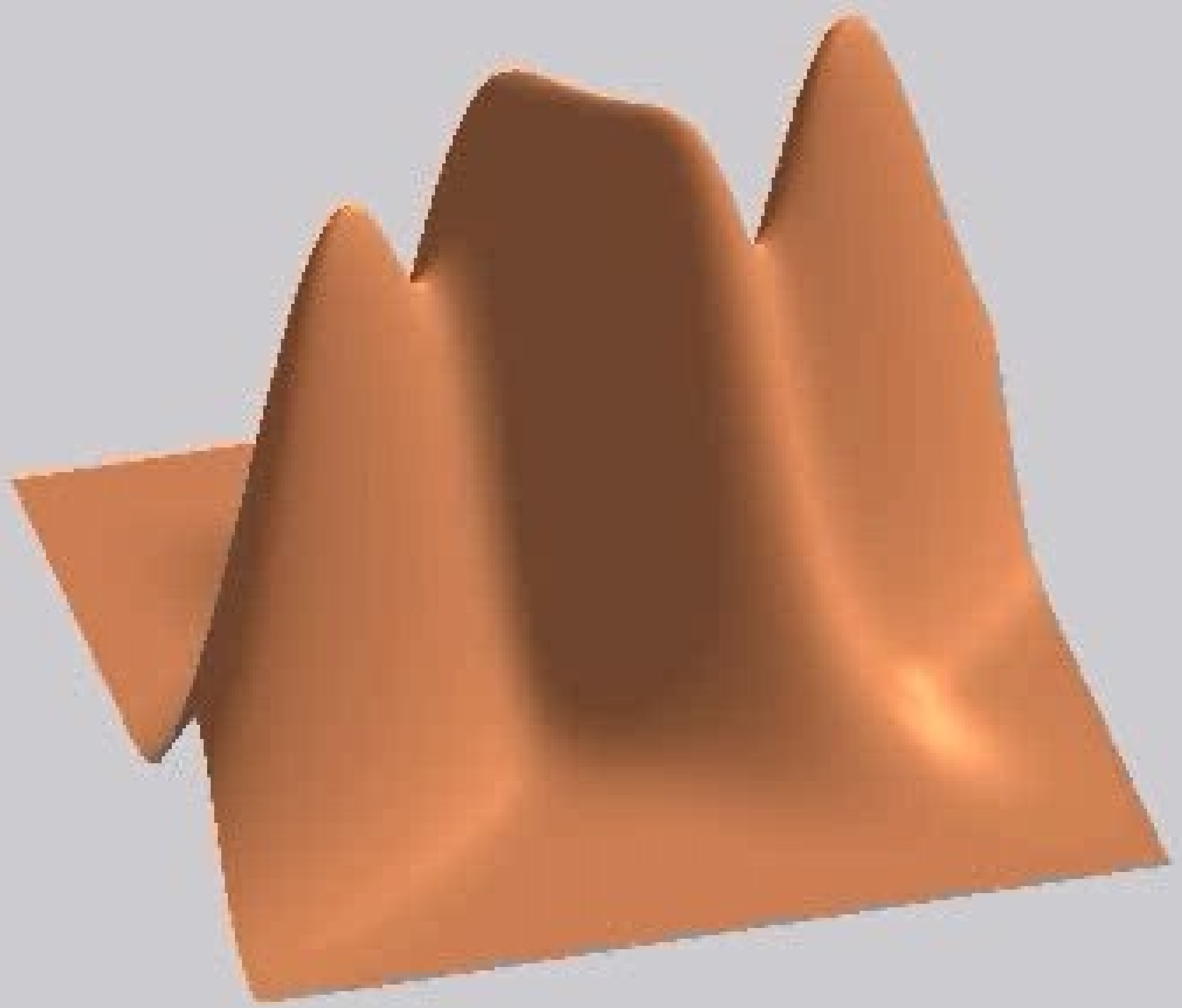
$$p(\mathbf{x}) = \sum_{k=1}^{K} p(k)p(\mathbf{x}|k)$$

Example: Mixture of 3 Gaussians



Contours of Probability Distribution





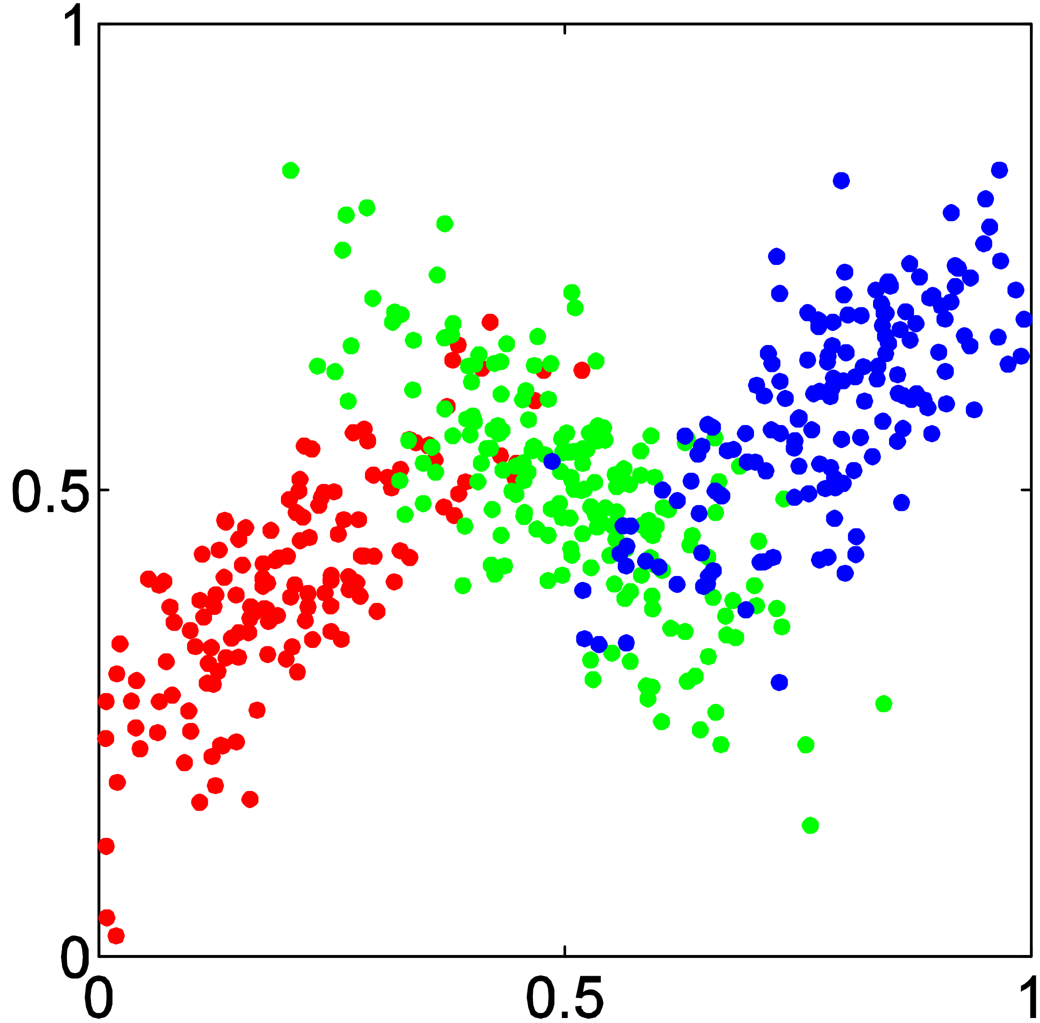
Sampling from the Gaussian

- To generate a data point:
  - first pick one of the components with probability π_k
  - then draw a sample $\mathbf{x}_n$ from that component
- Repeat these two steps for each new data point

Sampling from the Gaussian

- To generate a data point:
  - first pick one of the components with π_k probability $\mathbf{x}_n$
  - then draw a sample from that component
- Repeat these two steps for each new data point

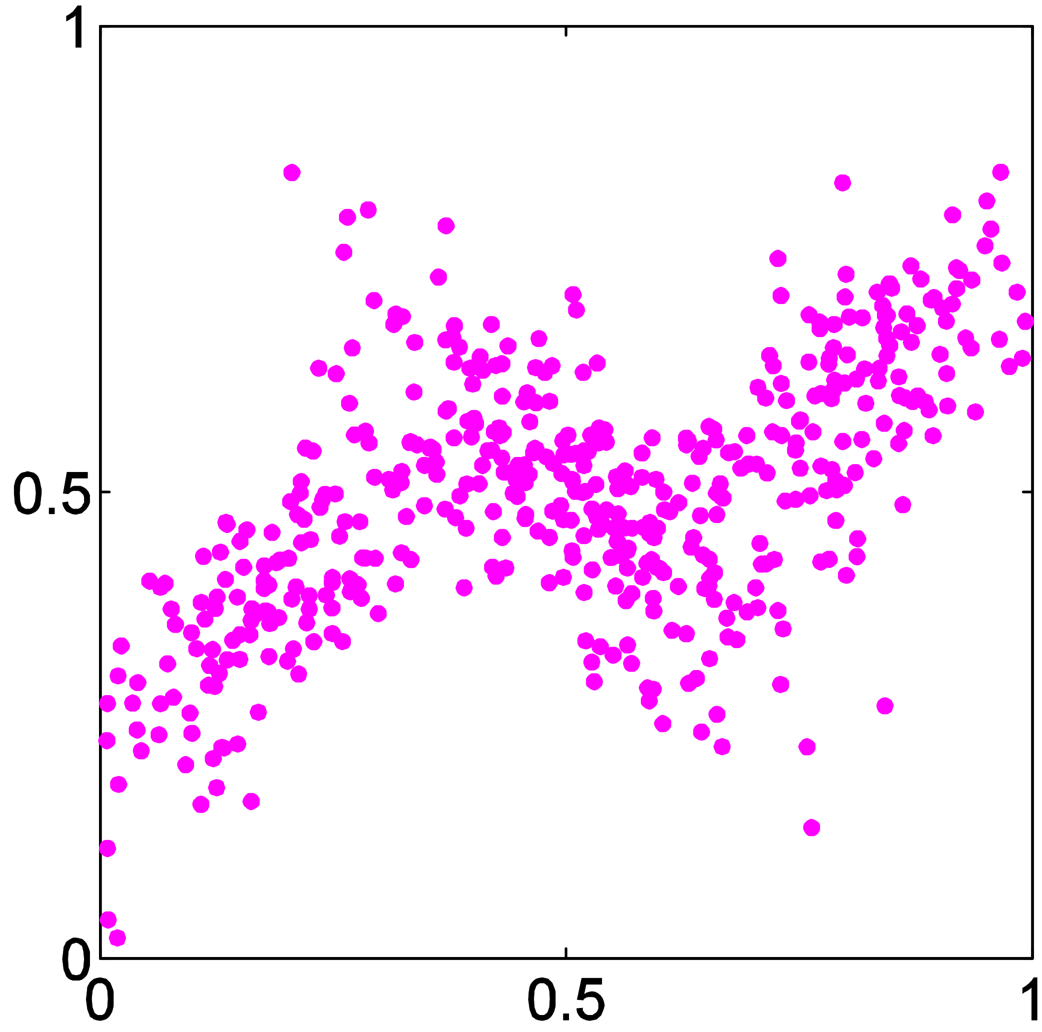
Synthetic Data Set



Fitting the Gaussian Mixture

- We wish to invert this process – given the data set, find the corresponding parameters:
  - mixing coefficients
  - means
  - covariances
- If we knew which component generated each data point, the maximum likelihood solution would involve fitting each component to the corresponding cluster
- Problem: the data set is unlabelled
- We shall refer to the labels as *latent* (= hidden) variables

Synthetic Data Set Without Labels



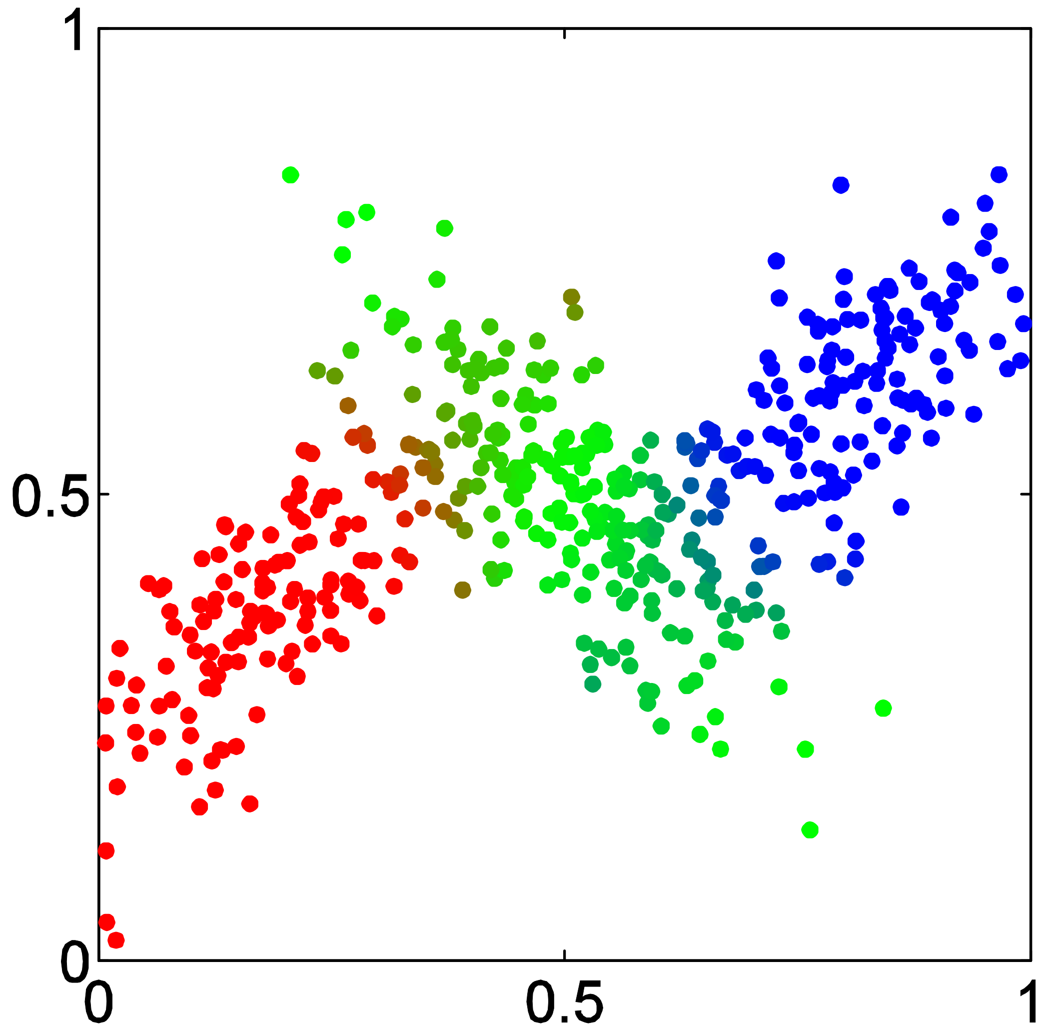
Posterior Probabilities

- We can think of the mixing coefficients as prior probabilities for the components
- For a given value of $\mathbf{x}$ we can evaluate the corresponding posterior probabilities, called *responsibilities*
- These are given from Bayes' theorem by

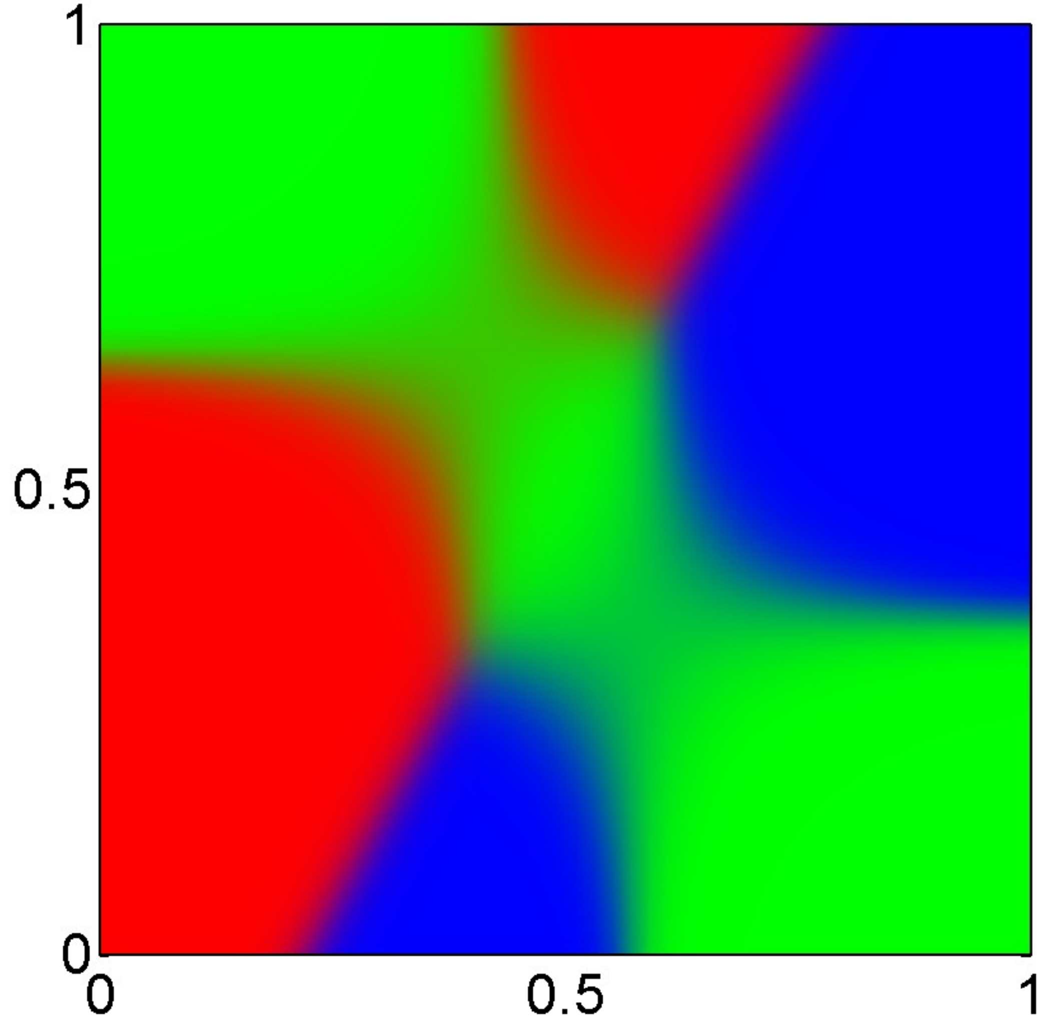
$$\gamma_k(\mathbf{x}) \equiv p(k|\mathbf{x}) = \frac{p(k)p(\mathbf{x}|k)}{p(\mathbf{x})}$$

$$= \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

Posterior Probabilities (colour coded)



Posterior Probability Map



ML for the GMM

- The log likelihood function takes the form

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

- Note: sum over components appears *inside* the log
- There is no closed form solution for maximum likelihood

Problems and Solutions

- How to maximize the log likelihood
  - solved by expectation-maximization (EM) algorithm
- How to avoid singularities in the likelihood function
  - solved by a Bayesian treatment
- How to choose number K of components
  - also solved by a Bayesian treatment

EM – Informal Derivation

- Let us proceed by simply differentiating the log likelihood $\boldsymbol{\mu}_j$
- Sett zero give

$$-\sum_{n=1}^{N} \underbrace{\frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_k \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}}_{\gamma_j(\mathbf{x}_n)} \boldsymbol{\Sigma}_j^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_j) = 0$$

giving

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)}$$

which is simply the weighted mean of the data

EM – Informal Derivation

- Similarly for the covariances

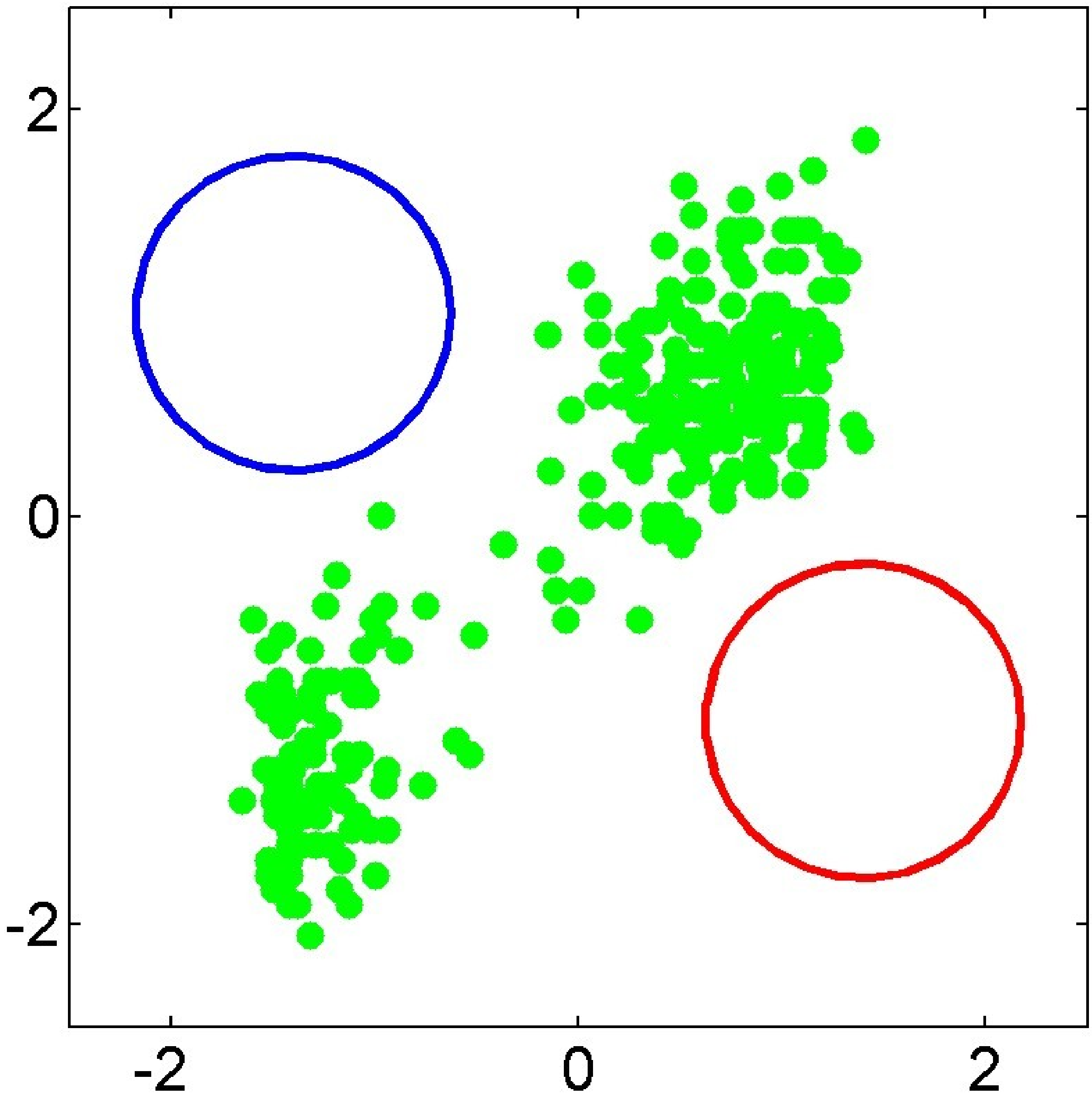
$$\boldsymbol{\Sigma}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)(\mathbf{x}_n - \boldsymbol{\mu}_j)(\mathbf{x}_n - \boldsymbol{\mu}_j)^{\mathsf{T}}}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)}$$

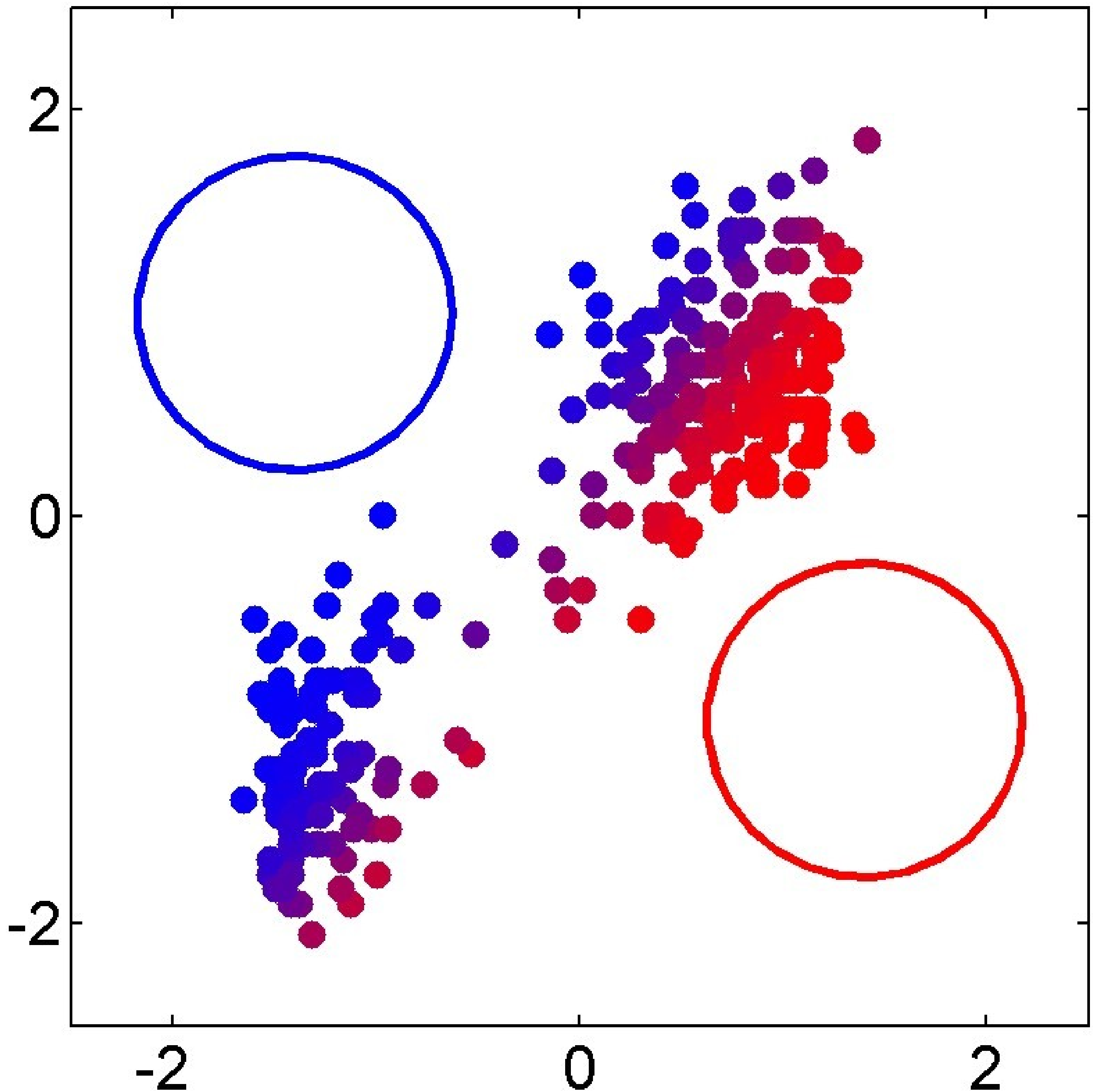
- For mixing coefficients use a Lagrange multiplier to give

$$\pi_j = \frac{1}{N} \sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)$$

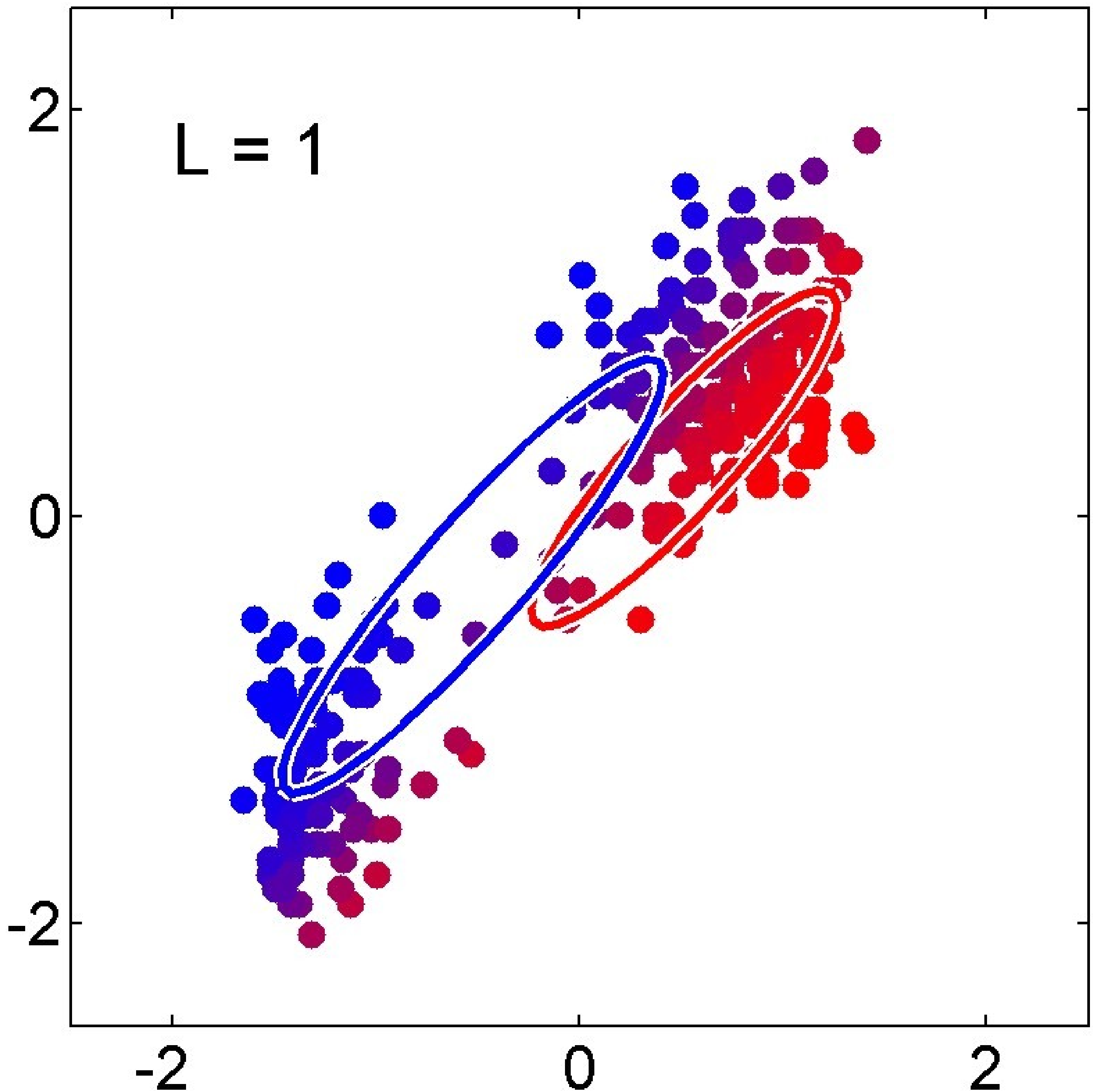
EM – Informal Derivation

- The solutions are not closed form since they are coupled
- Suggests an iterative scheme for solving them:
  - Make initial guesses for the parameters
  - Alternate between the following two stages:
    1. E-step: evaluate responsibilities
    2. M-step: update parameters using ML results

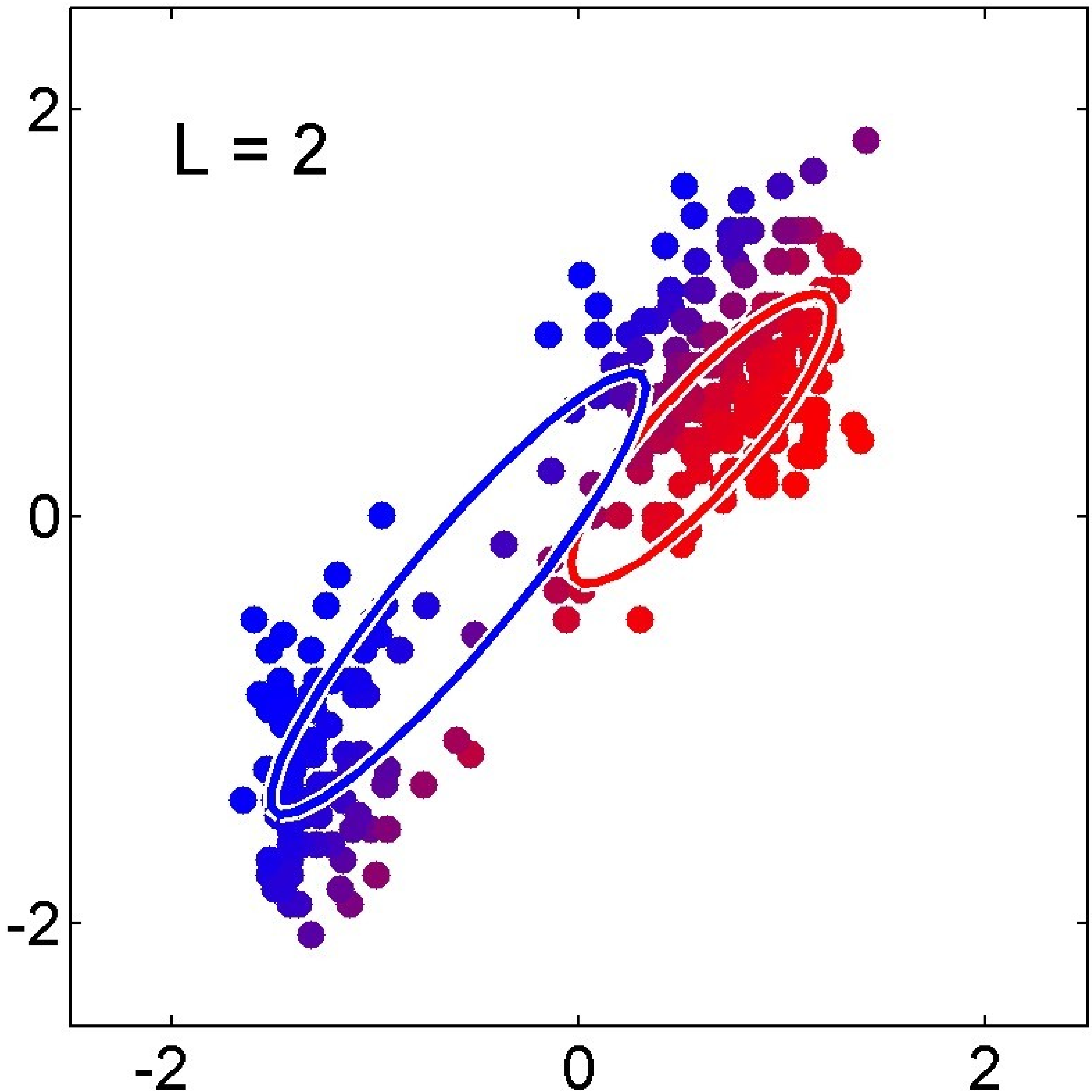

2
0
-2
-2
0
2

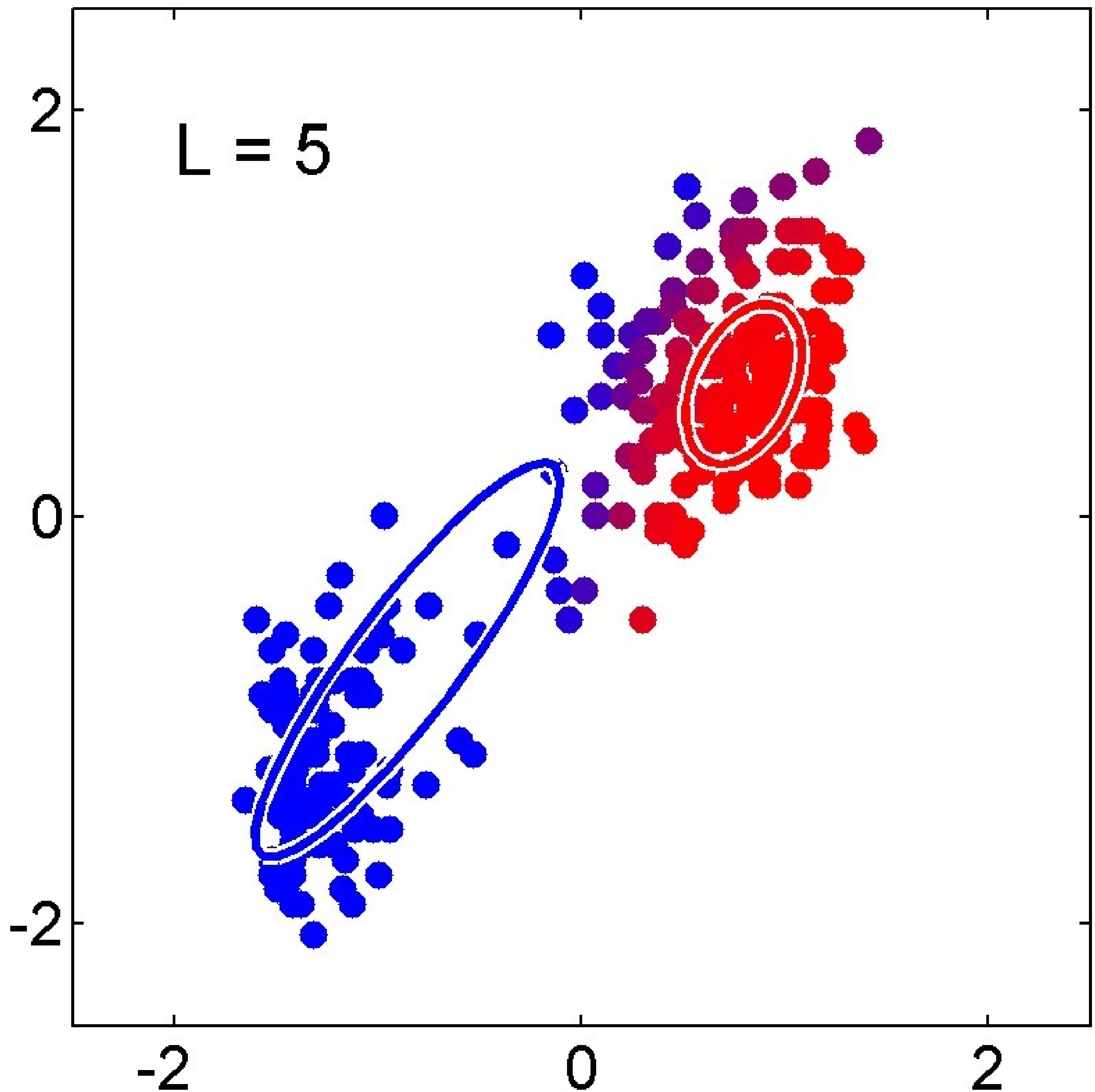


2
0
-2
-2
0
2

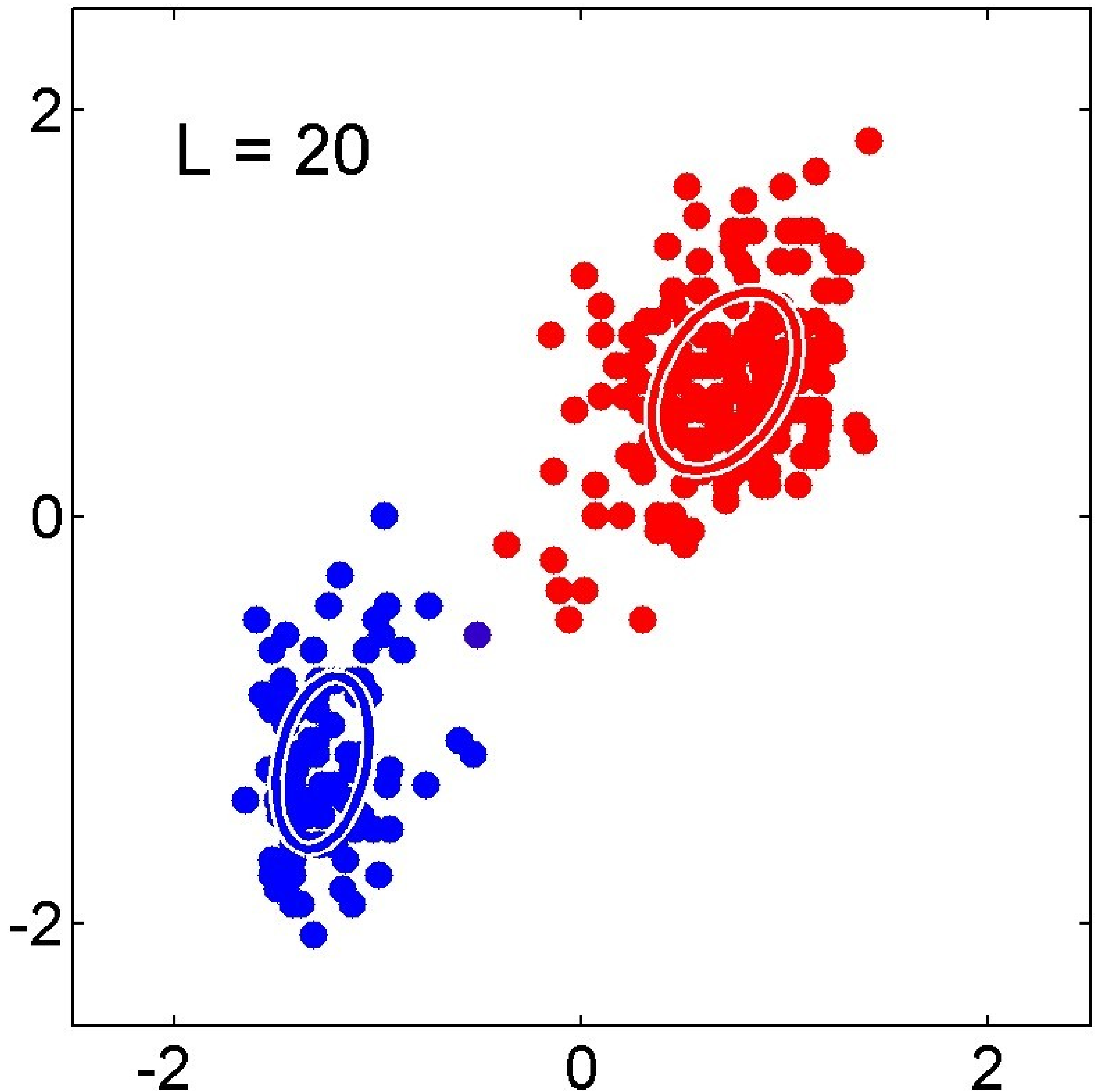


L = 1
2
0
-2
-2
0
2


L = 2
2
0
-2
-2
0
2



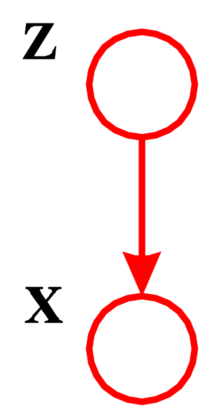
L = 5
2
0
-2
-2
0
2


L = 20
2
0
-2
-2
0
2

EM – Latent Variable Viewpoint

- Binary latent variables $\mathbf{z} = \{z_{kn}\}$ describing which component generated each data point
- Conditional distribution of observed variable

$$p(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k)^{z_k}$$



- Prior distribution of latent variables

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

- Marginalizing over the latent variables we obtain

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

Human Action Recognition

- Given a video, we need to recognize the human action in that
  - Bowling, batting and fielding
- Video clips could be of varying lenghts
  - Cannot compute a similarity score
- Solution: estimate & compare pdf of action clips
  - Fit a huge GMM (1024 or more) components
  - It captures all possible atomic attributes of human actions
  - Adapt the GMM to each clip, and estimate sufficient statistics
  - Compute similarity between sufficient stats

Thank You!