# Chapter 12

# Bayesian Classifier

# Chapter List

- Bayes Decision Theory
- Maximum Likelihood Estimation
- Naïve Bayes Classifier
- EM Algorithm

# Chapter List

- Bayes Decision Theory
- Maximum Likelihood Estimation
- Naïve Bayes Classifier
- EM Algorithm

# Bayes Decision Theory

- Bayesian decision theory is a fundamental decision-making approach under the probability framework.
  - When all relevant probabilities were known, Bayesian decision theory makes optimal classification decisions based on the probabilities and costs of misclassifications.

# Bayes Decision Theory

- Bayesian decision theory is a fundamental decision-making approach under the probability framework.
  - When all relevant probabilities were known, Bayesian decision theory makes optimal classification decisions based on the probabilities and costs of misclassifications.
- Let us assume that there are $N$ distinct class labels, that is, $y = \{c_1, c_2, \ldots, c_N\}$. Let $\lambda_{ij}$ denote the cost of misclassifying a sample of class $c_j$ as class $c_i$. Then, with the posterior probability $P(c_i \mid \boldsymbol{x})$ we can calculate the expected loss of classifying a sample $\mathbf{x}$ as class $c_i$, that is, the conditional risk of the sample $\mathbf{x}$:

$$R(c_i \mid \mathbf{x}) = \sum_{j=1}^{N} \lambda_{ij} P(c_j \mid \mathbf{x}) \qquad (7.1)$$

- Our task is to find a decision rule $h : X \mapsto Y$ that minimizes the overall risk:

$$R(h) = \mathbf{E}_x [R(h(\mathbf{x}) \mid \mathbf{x})] \qquad (7.2)$$

# Bayes Decision Theory

- The overall risk $R(h)$ is minimized when the conditional risk $R(h(\mathbf{x}) \mid \mathbf{x})$ of each sample $\mathbf{x}$ is minimized.

# Bayes Decision Theory

- The overall risk $R(h)$ is minimized when the conditional risk $R(h(\mathbf{x}) \mid \mathbf{x})$ of each sample $\mathbf{x}$ is minimized.

- This leads to the Bayes decision rule: to minimize the overall risk, classify each sample as the class that minimizes the conditional risk $R(c \mid \mathbf{x})$

$$h^*(x) = \underset{c \in y}{\operatorname{argmin}}\, R(c \mid x)$$

  - where $h^*$ is called the Bayes optimal classifier, and its associated overall risk $R(h^*)$ is called the Bayes risk.
  - $1 - R(h^*)$ is the best performance that can be achieved by any classifiers, that is, the theoretically achievable upper bound of accuracy for any machine learning models.

# Bayes Decision Theory

- To be specific, if the objective is to minimize the misclassification rate, then the misclassification loss $\lambda_{ij}$ can be written as

$$\lambda_{i,j} = \begin{cases} 0, & \text{if } i = j; \\ 1, & \text{otherwise,} \end{cases}$$

# Bayes Decision Theory

- To be specific, if the objective is to minimize the misclassification rate, then the misclassification loss $\lambda_{ij}$ can be written as

$$\lambda_{i,j} = \begin{cases} 0, & \text{if } i = j; \\ 1, & \text{otherwise}, \end{cases}$$

- and the conditional risk is

$$R(c \mid \mathbf{x}) = 1 - P(c \mid \mathbf{x})$$

# Bayes Decision Theory

- To be specific, if the objective is to minimize the misclassification rate, then the misclassification loss $\lambda_{ij}$ can be written as

$$\lambda_{i,j} = \begin{cases} 0, & \text{if } i = j; \\ 1, & \text{otherwise}, \end{cases}$$

- and the conditional risk is

$$R(c \mid \mathbf{x}) = 1 - P(c \mid \mathbf{x})$$

- Then, the Bayes optimal classifier that minimizes the misclassification rate is

$$h^*(x) = \underset{c \in y}{\operatorname{argmax}}\, P(c \mid x)$$

  - which classifies each sample $\mathbf{x}$ as the class that maximizes its posterior probability $P(c \mid \mathbf{x})$.

# Bayes Decision Theory

- We can see that the Bayes decision rule relies on the posterior probability $P(c \mid \mathbf{x})$ .

- However, it's often difficult to obtain in practice. The task of machine learning is then to accurately estimate the posterior probability $P(c \mid \mathbf{x})$ from the training samples.
- Generally speaking, there are two strategies :
  - discriminative models
    - Given $\mathbf{x}$, predict $c$ by estimating $P(c \mid \mathbf{x})$ directly.
    - Decision trees, BP neural networks and support vector machines.
  - generative models
    - estimate the joint probability $P(\mathbf{x}, c)$ first and then estimate $P(c \mid \mathbf{x})$
    - For generative models, we must evaluate:

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x}, c)}{P(\mathbf{x})}$$

# Bayes Decision Theory

- For generative models, we must evaluate:

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x}, c)}{P(\mathbf{x})}$$

# Bayes Decision Theory

- For generative models, we must evaluate:

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x}, c)}{P(\mathbf{x})}$$

- According to Bayes' theorem, $P(c \mid \mathbf{x})$ can be written as:

$$P(c \mid \mathbf{x}) = \frac{P(c)P(\mathbf{x} \mid c)}{P(\mathbf{x})}$$

# Bayes Decision Theory

- For generative models, we must evaluate:

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x}, c)}{P(\mathbf{x})}$$

- According to Bayes' theorem, $P(c \mid \mathbf{x})$ can be written as:

$$P(c \mid \mathbf{x}) = \frac{P(c) P(\mathbf{x} \mid c)}{P(\mathbf{x})}$$

the prior probability represents the proportion of each class in the sample, which can be estimated by the frequency of each class in the training set

# Bayes Decision Theory

- For generative models, we must evaluate:

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x}, c)}{P(\mathbf{x})}$$

- According to Bayes' theorem, $P(c \mid \mathbf{x})$ can be written as:

$$P(c \mid \mathbf{x}) = \frac{P(c)P(\mathbf{x} \mid c)}{P(\mathbf{x})}$$

the prior probability represents the proportion of each class in the sample, which can be estimated by the frequency of each class in the training set

the evidence factor, which is independent of the class

# Bayes Decision Theory

- For generative models, we must evaluate:

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x}, c)}{P(\mathbf{x})}$$

- According to Bayes' theorem, $P(c \mid \mathbf{x})$ can be written as:

$$P(c \mid \mathbf{x}) = \frac{P(c)\, P(\mathbf{x} \mid c)}{P(\mathbf{x})}$$

$P(\mathbf{x} \mid c)$: the class-conditional probability, also known as the likelihood, of the sample $\mathbf{x}$ with respect to class $c$

$P(c)$: the prior probability represents the proportion of each class in the sample, which can be estimated by the frequency of each class in the training set

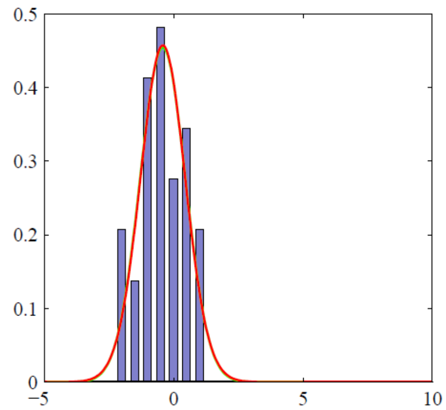
$P(\mathbf{x})$: the evidence factor, which is independent of the class

# Chapter List

- Bayes Decision Theory
- **Maximum Likelihood Estimation**
- Naïve Bayes Classifier
- EM Algorithm

# Maximum Likelihood Estimation

- A general strategy of estimating the class-conditional probability is to hypothesize a fixed form of probability distribution, and then estimate the distribution parameters using the training samples.




- let $P(\mathbf{x} \mid c)$ denote class-condition probability of class $c$ ,
  - suppose $P(\mathbf{x} \mid c)$ has a fixed form determined by a parameter vector $\boldsymbol{\theta}_c$. Then, the task is to estimate $\boldsymbol{\theta}_c$ from a training set $D$.

# Maximum Likelihood Estimation

- The training process of probabilistic models is the process of parameter estimation. There are two different ways of thinking about parameters:

  - **(The Frequentist school)** Parameters have unknown but fixed values, and hence they can be determined by some approaches such as optimizing the likelihood function.

  - **(The Bayesian school)** Parameters are unobserved random variables following some distribution, and hence we can assume prior distributions for the parameters and estimate posterior distribution from observed data.

# Maximum Likelihood Estimation

- Let $D_c$ denote the set of class $c$ samples in the training set $D$, and further suppose the samples are *i.i.d.* samples. Then, the likelihood of $D_c$ for a given parameter $\boldsymbol{\theta}_c$ is:

$$P(D_c \mid \boldsymbol{\theta_c}) = \prod_{\mathbf{x} \in D_c} P(\mathbf{x} \mid \boldsymbol{\theta_c})$$

  - Applying the MLE to $\boldsymbol{\theta}_c$ is about finding a parameter value $\hat{\theta}_c$ that maximizes the likelihood $P(D_c \mid \theta_c)$. Intuitively, the MLE aims to find a value of $\theta_c$ that maximizes the "*likelihood*" that the data will present.

# Maximum Likelihood Estimation

- Let $D_c$ denote the set of class $c$ samples in the training set $D$, and further suppose the samples are *i.i.d.* samples. Then, the likelihood of $D_c$ for a given parameter $\boldsymbol{\theta}_c$ is:

$$P(D_c \mid \boldsymbol{\theta_c}) = \prod_{\mathbf{x} \in D_c} P(\mathbf{x} \mid \boldsymbol{\theta_c})$$

  - Applying the MLE to $\boldsymbol{\theta}_c$ is about finding a parameter value $\hat{\theta}_c$ that maximizes the likelihood $P(D_c \mid \theta_c)$. Intuitively, the MLE aims to find a value of $\theta_c$ that maximizes the "likelihood" that the data will present.

- Since the product of a sequence can lead to underflow, we often use the log-likelihood instead:

$$\begin{aligned} LL(\boldsymbol{\theta_c}) &= \log P(D_c \mid \boldsymbol{\theta_c}) \\ &= \sum_{\mathbf{x} \in D_c} \log P(\mathbf{x} \mid \boldsymbol{\theta}_c) \end{aligned}$$

- and the MLE of $\boldsymbol{\theta}_c$ is $\hat{\boldsymbol{\theta}}_c$ :

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{c}} = \underset{\boldsymbol{\theta}_c}{\arg\max}\, LL(\boldsymbol{\theta}_c)$$

# Maximum Likelihood Estimation

- For example, suppose the features are continuous and the probability density function follows the Gaussian distribution $p(\mathbf{x} \mid c) \sim N(\boldsymbol{\mu}_c, \boldsymbol{\sigma}_c^2)$ , then the MLE of the parameters $\boldsymbol{\mu}_c$ and $\boldsymbol{\sigma}_c^2$ are

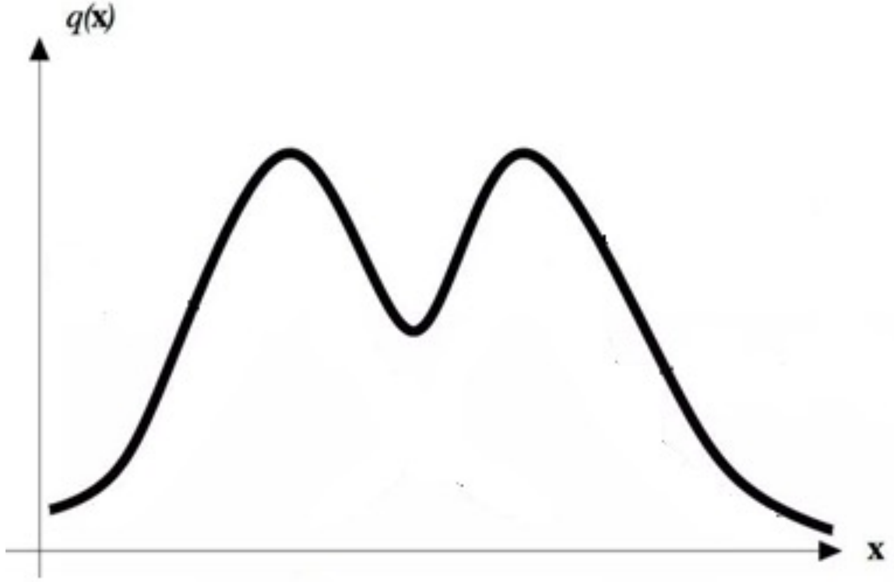
$$\hat{\boldsymbol{\mu}}_c = \frac{1}{|D_c|} \sum_{\mathbf{x} \in D_c} \mathbf{x}$$

$$\hat{\boldsymbol{\sigma}}_c^2 = \frac{1}{|D_c|} \sum_{\mathbf{x} \in D_c} (\mathbf{x} - \hat{\boldsymbol{\mu}}_c)(\mathbf{x} - \hat{\boldsymbol{\mu}}_c)^{\mathrm{T}}$$

- In other words, the estimated mean of Gaussian distribution obtained by the MLE is the sample mean, and the estimated variance is the mean of $(\mathbf{x} - \hat{\boldsymbol{\mu}}_c)(\mathbf{x} - \hat{\boldsymbol{\mu}}_c)^{\mathrm{T}}$ .

# Maximum Likelihood Estimation

- Such kind of parametric methods simplify the estimation of posterior probabilities, but the accuracy of estimation heavily relies on whether the hypothetical probability distribution matches the unknown ground-truth data distribution. In practice, a "guessed" probability distribution could incur misleading results.



# Chapter List

- Bayes Decision Theory
- Maximum Likelihood Estimation
- Naïve Bayes Classifier
- EM Algorithm

# Naïve Bayes Classifier

- The difficulty of estimating the posterior probability $P(c \mid \mathbf{x})$ : it is not easy to calculate the class-conditional probability $P(\mathbf{x} \mid c)$ from the training samples since $P(\mathbf{x} \mid c)$ is the joint probability on all attributes.
  - For example, $d$ binary attributes $\rightarrow 2^d$ possible values, $2^d$ >> the number of samples
- To avoid this, the Naïve Bayes classifier makes the "attribute conditional independence assumption": for any class, assume all attributes are independent of each other.
- With the independence assumption, we have:

$$P(c \mid \mathbf{x}) = \frac{P(c)P(\mathbf{x} \mid c)}{P(\mathbf{x})} = \frac{P(c)}{P(\mathbf{x})} \prod_{i=1}^{d} P(x_i \mid c)$$

where $d$ is the number of attributes.

# Naïve Bayes Classifier

$$P(c \mid \mathbf{x}) = \frac{P(c)P(\mathbf{x} \mid c)}{P(\mathbf{x})} = \frac{P(c)}{P(\mathbf{x})} \prod_{i=1}^{d} P(x_i \mid c)$$

# Naïve Bayes Classifier

$$P(c \mid \mathbf{x}) = \frac{P(c)P(\mathbf{x} \mid c)}{P(\mathbf{x})} = \frac{P(c)}{P(\mathbf{x})} \prod_{i=1}^{d} P(x_i \mid c)$$

Since $P(x)$ is the same for all classes, from the Bayes decision rule, we have

$$h_{nb}(\mathbf{x}) = \operatorname*{argmax}_{c \in y} P(c) \prod_{i=1}^{d} P(x_i \mid c)$$

which is the formulation of the Naïve Bayes classifier.

# Naïve Bayes Classifier

- To train a Naïve Bayes classifier, we compute the prior probability $P(c)$ from the training set $D$ and then compute the conditional probability $P(x_i \mid c)$ for each attribute.
  - Let $D_c$ denote a subset of $D$ containing all samples of class $c$ Then, The prior probability can be estimated by

$$P(c) = \frac{|D_c|}{|D|}$$

  - For discrete attributes, let $D_{c,x_i}$ denote a subset of $D_c$ containing all samples taking the value $x_i$ on the $i$-th attribute. Then, the conditional probability $P(x_i \mid c)$ can be estimated by

$$P(x_i \mid c) = \frac{|D_{c,x_i}|}{|D_c|}$$

  - For continuous features, suppose $p(x_i \mid c) \sim N(\mu_{c,i}, \sigma_{c,i}^2)$ , where $\mu_{c,i}$ and $\sigma_{c,i}^2$ are, respectively, the mean and variance of the $i$-th feature of class $c$. Then, we have

$$P(x_i \mid c) = \frac{1}{\sqrt{2\pi}\sigma_{c,i}} \exp(-\frac{(x_i - \mu_{c,i})^2}{2{\sigma_{c,i}}^2})$$

# Laplace (add-1) Smoothing

- If a feature value has never appeared together with a particular class, it becomes problematic to use the probability.
- For example, given a testing sample with *sound = crisp*, the Naïve Bayes classifier trained on the watermelon data set will predict 0. The classification result will always be *ripe = false* regardless of the values of other features.

$$P_{crisp|true} = \frac{0}{8} = 0$$

| ID | color | root | sound | texture | umbilicus | surface | ripe |
|---|---|---|---|---|---|---|---|
| 1 | green | curly | muffled | clear | hollow | hard | true |
| 2 | dark | curly | dull | clear | hollow | hard | true |
| 3 | dark | curly | muffled | clear | hollow | hard | true |
| 4 | green | curly | dull | clear | hollow | hard | true |
| 5 | light | curly | muffled | clear | hollow | hard | true |
| 6 | green | slightly curly | muffled | clear | slightly hollow | soft | true |
| 7 | dark | slightly curly | muffled | slightly blurry | slightly hollow | soft | true |
| 8 | dark | slightly curly | muffled | clear | slightly hollow | hard | true |
| 9 | dark | slightly curly | dull | slightly blurry | slightly hollow | hard | false |
| 10 | green | straight | crisp | clear | flat | soft | false |
| 11 | light | straight | crisp | blurry | flat | hard | false |
| 12 | light | curly | muffled | blurry | flat | soft | false |
| 13 | green | slightly curly | muffled | slightly blurry | hollow | hard | false |
| 14 | light | slightly curly | dull | slightly blurry | hollow | hard | false |
| 15 | dark | slightly curly | muffled | clear | slightly hollow | soft | false |
| 16 | light | curly | muffled | blurry | flat | hard | false |
| 17 | green | curly | dull | slightly blurry | slightly hollow | hard | false |

# Laplace (add-1) Smoothing

- To avoid "removing" the information carried by other features, a common choice is the Laplace smoothing.
  - Let $N$ denote the number of distinct classes in the training set $D$, $N_i$ denote the number of distinct values the $i$-th feature can take. Then, we write smoothed version of prior probability and conditional probability as:

$$\hat{P}(c) = \frac{|D_c| + 1}{|D| + N},$$

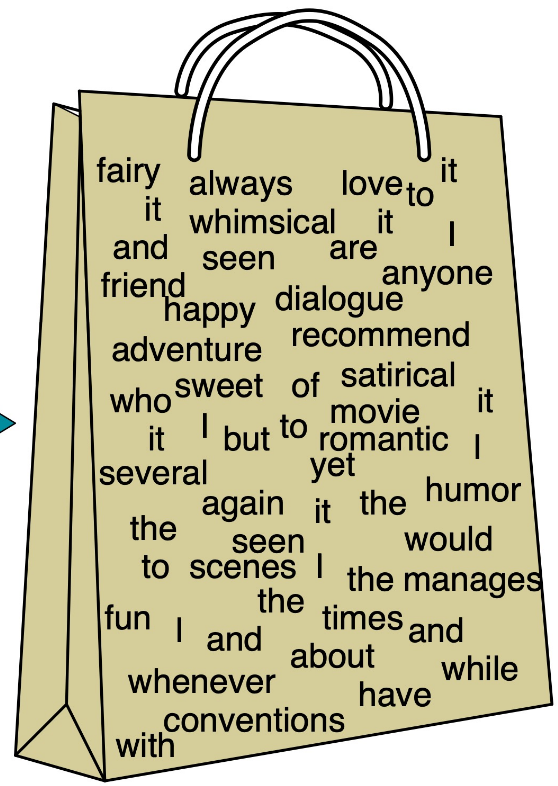
Why?

$$\hat{P}(x_i \mid c) = \frac{|D_{c,x_i}| + 1}{|D_c| + N_i}$$

# Text Classification

## The Bag of Words Representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!




| | |
|---|---|
| it | 6 |
| I | 5 |
| the | 4 |
| to | 3 |
| and | 3 |
| seen | 2 |
| yet | 1 |
| would | 1 |
| whimsical | 1 |
| times | 1 |
| sweet | 1 |
| satirical | 1 |
| adventure | 1 |
| genre | 1 |
| fairy | 1 |
| humor | 1 |
| have | 1 |
| great | 1 |
| … | … |

# Multinomial Distribution

Suppose one does an experiment of extracting $n$ balls of $k$ different colors from a bag, replacing the extracted balls after each draw. Balls of the same color are equivalent. Denote the variable which is the number of extracted balls of color $i$ ($i = 1, \dots, k$) as $X_i$, and denote as $p\_i$ the probability that a given extraction will be in color $i$.

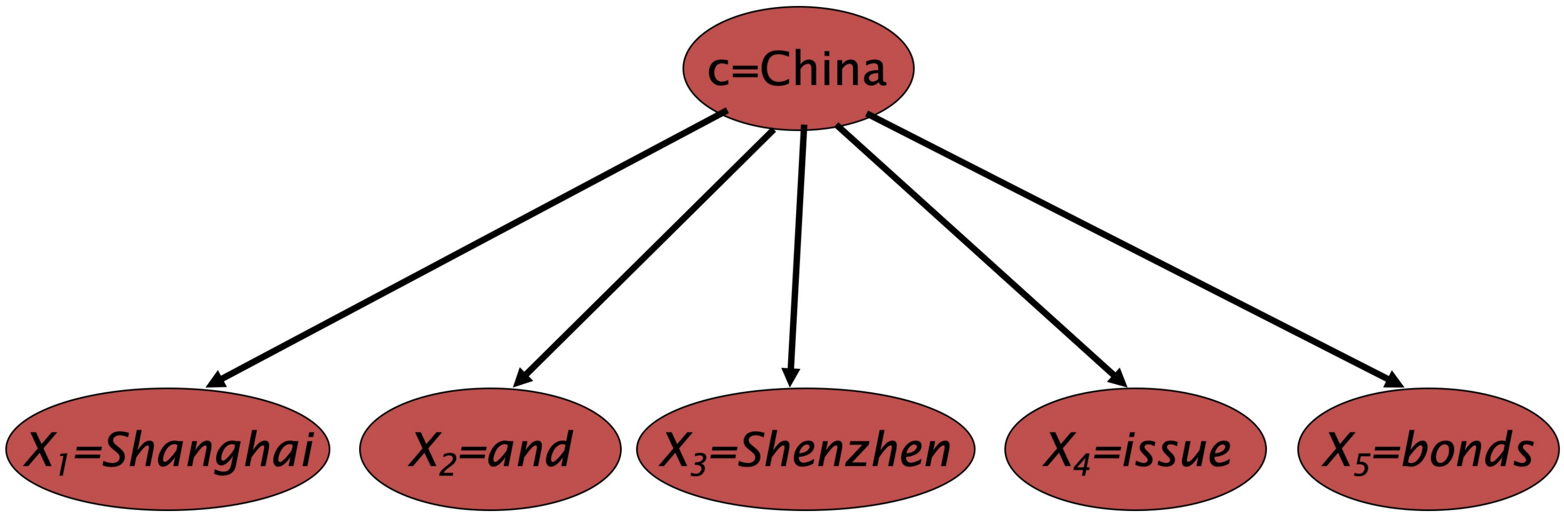
The **probability mass function** of this multinomial distribution is:

$$f(x_1, \ldots, x_k; n, p_1, \ldots, p_k) = \Pr(X_1 = x_1 \text{ and } \ldots \text{ and } X_k = x_k)$$
$$= \begin{cases} \dfrac{n!}{x_1! \cdots x_k!} p_1^{x_1} \times \cdots \times p_k^{x_k}, & \text{when } \sum_{i=1}^{k} x_i = n \\ \\ 0 & \text{otherwise,} \end{cases}$$

for non-negative integers $x_1, \dots, x_k$.

# Generative Model for Naive Bayes

$$P(x_i \mid c)$$



# Text Classification

Consider a naive Bayes model with the classes positive (+) and negative (-) and the following model parameters:

| w | P(w\|+) | P(w\|-) |
|---|---|---|
| I | 0.1 | 0.2 |
| love | 0.1 | 0.001 |
| this | 0.01 | 0.01 |
| fun | 0.05 | 0.005 |
| film | 0.1 | 0.1 |
| ... | ... | ... |

$$P(\text{“I love this fun film”}|+) = 0.1 \times 0.1 \times 0.01 \times 0.05 \times 0.1 = 0.0000005$$
$$P(\text{“I love this fun film”}|-) = 0.2 \times 0.001 \times 0.01 \times 0.005 \times 0.1 = .0000000010$$

Note that this is just the likelihood part of the naive Bayes model.

# Text Classification

To apply the naive Bayes classifier to text, we need to consider word positions, by simply walking an index through every word position in the document:

$$\text{positions} \leftarrow \text{all word positions in test document}$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{i \in positions} P(w_i|c)$$

Naive Bayes calculations are done in log space, to avoid underflow and increase speed

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} \log P(c) + \sum_{i \in positions} \log P(w_i|c)$$

Naive Bayes is a **linear classifiers**.

# Training the Naive Bayes Classifier

Let $N_c$ be the number of documents in our training data with class $c$ and $N_{doc}$ be the total number of documents. Then:

$$\hat{P}(c) = \frac{N_c}{N_{doc}}$$

$$\hat{P}(w_i|c) = \frac{count(w_i, c)}{\sum_{w \in V} count(w, c)}$$

$$\hat{P}(w_i|c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} (count(w, c) + 1)} = \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + |V|}$$

# Text Classification

| | Doc | Words | Class |
|---|---|---|---|
| Training | 1 | Chinese Beijing Chinese | c |
| | 2 | Chinese Chinese Shanghai | c |
| | 3 | Chinese Macao | c |
| | 4 | Tokyo Japan Chinese | j |
| Test | 5 | Chinese Chinese Chinese Tokyo Japan | ? |

**Priors:**

$P(c) = ?$

$P(j) = ?$

**Conditional Probabilities:**

$P(Chinese|c) = ?$

$P(Tokyo|c) = ?$

$P(Japan|c) = ?$

$P(Chinese|j) = ?$

$P(Tokyo|j) = ?$

$P(Japan|j) = ?$

**Choosing a class:**

$P(c|d5) = ?$

$P(j|d5) = ?$

$$\hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

# Text Classification

Priors:

$$P(c) = \frac{3+1}{4+2} = \frac{2}{3}$$

$$P(j) = \frac{1+1}{4+2} = \frac{1}{3}$$

# Text Classification

**Conditional Probabilities:**

P(Chinese|$c$) = (5+1) / (8+6) = 6/14 = 3/7

P(Tokyo|$c$) = (0+1) / (8+6) = 1/14

P(Japan|$c$) = (0+1) / (8+6) = 1/14

P(Chinese|$j$) = (1+1) / (3+6) = 2/9

P(Tokyo|$j$) = (1+1) / (3+6) = 2/9

P(Japan|$j$) = (1+1) / (3+6) = 2/9

# Text Classification

**Choosing a class:**

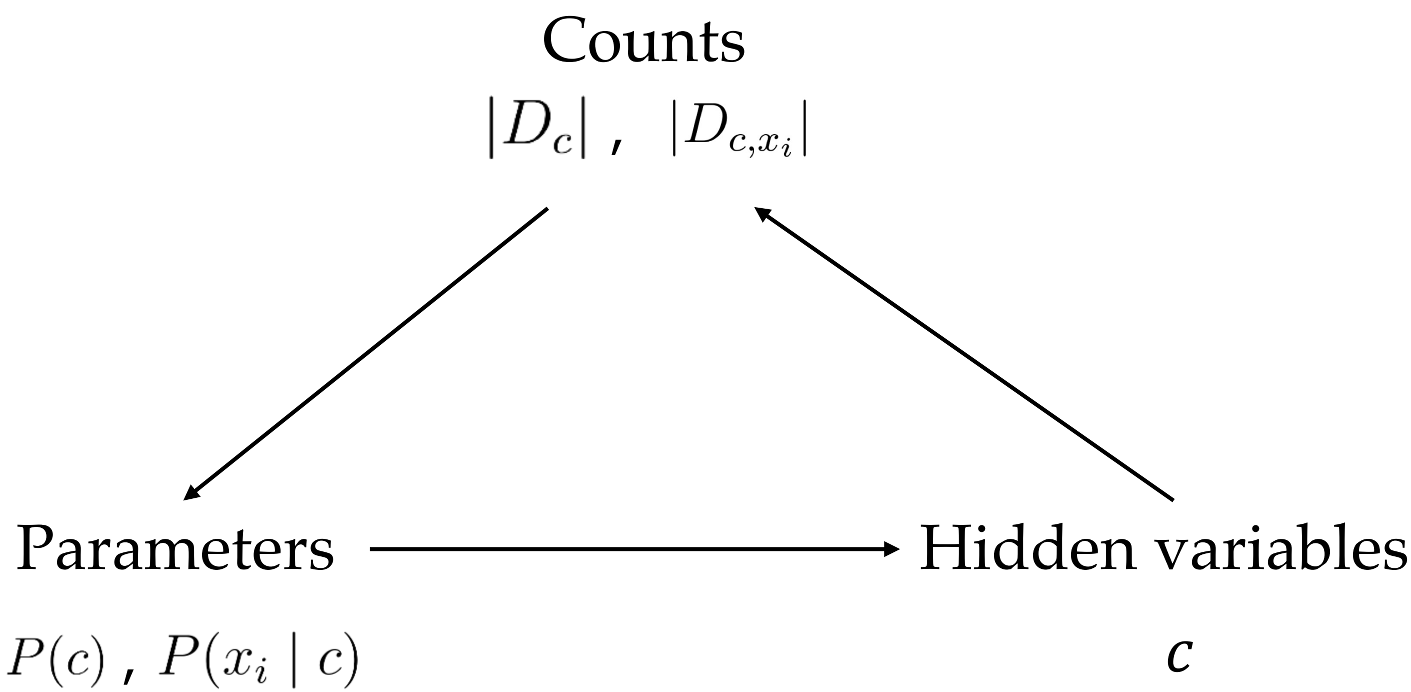
$$P(c|d5) \propto \frac{2}{3} * \left(\frac{3}{7}\right)^3 * \frac{1}{14} * \frac{1}{14} \approx 0.00027$$

$$P(j|d5) \propto \frac{1}{3} * \left(\frac{2}{9}\right)^3 * \frac{2}{9} * \frac{2}{9} \approx 0.00018$$

# Chapter List

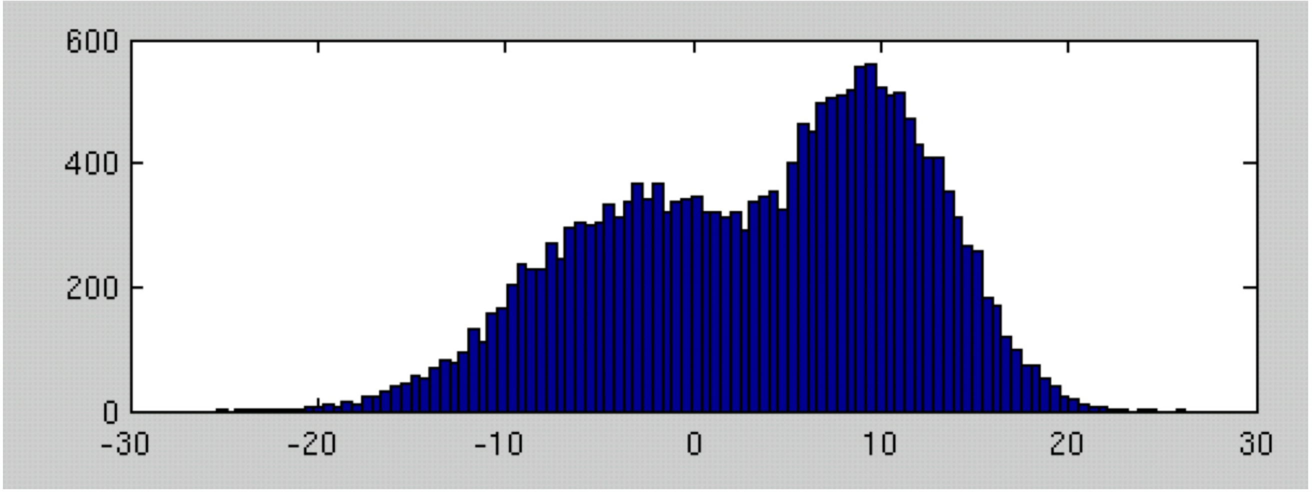
- Bayes Decision Theory
- Maximum Likelihood Estimation
- Naïve Bayes Classifier
- **EM Algorithm**

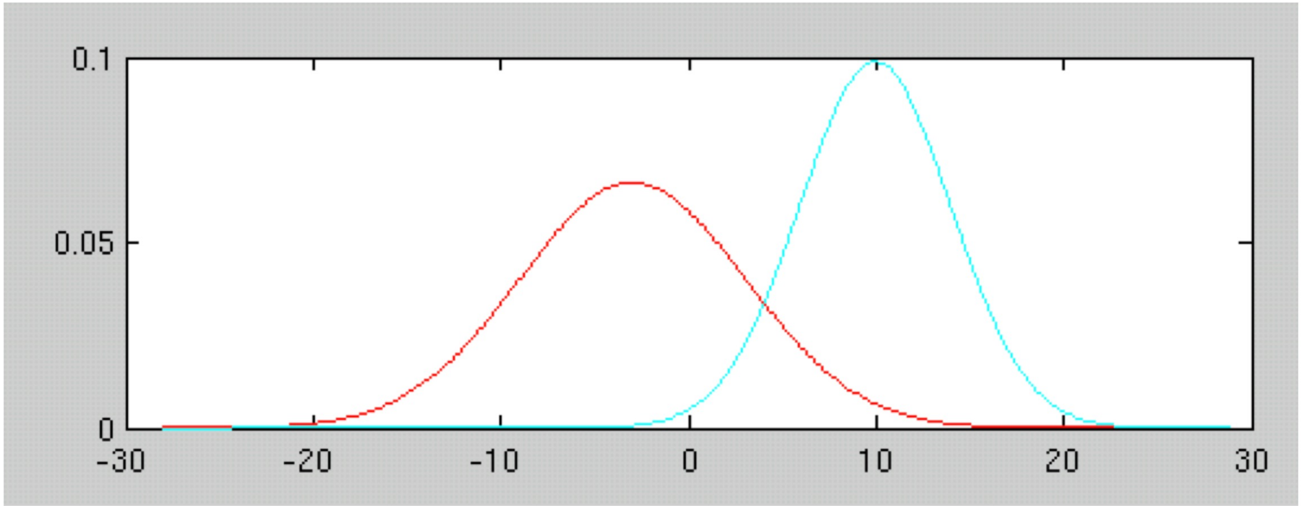
# Dealing with Hidden Variables




Observed variables: $x$

# Gaussian Mixture-Model



# Gaussian Mixture-Model

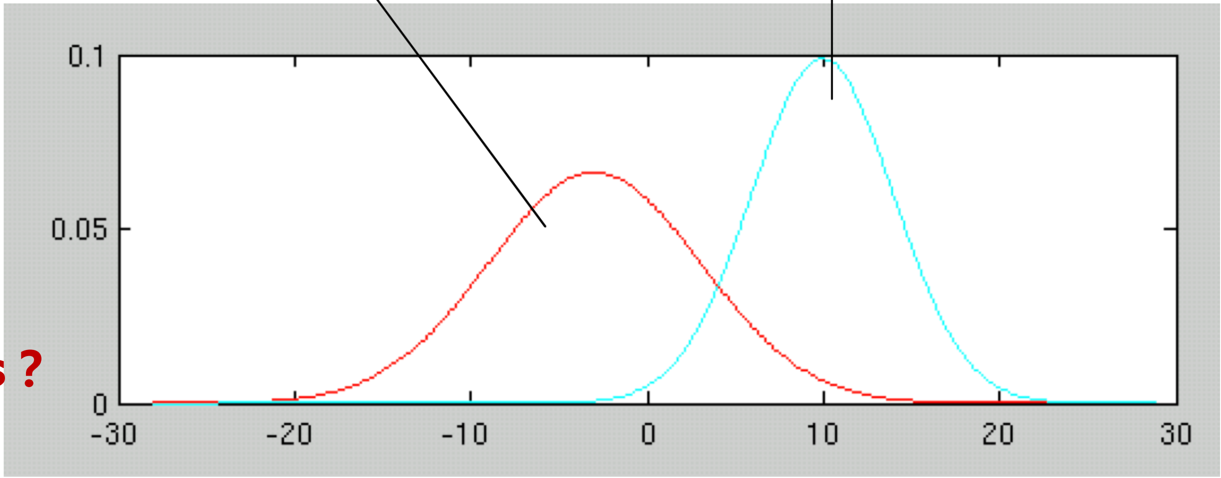
$$\mathrm{P}(Z_i = 1) = \tau_1 \text{ and } \mathrm{P}(Z_i = 2) = \tau_2 = 1 - \tau_1$$

$$X_i \mid (Z_i = 1) \sim \mathcal{N}_d(\boldsymbol{\mu}_1, \Sigma_1) \qquad X_i \mid (Z_i = 2) \sim \mathcal{N}_d(\boldsymbol{\mu}_2, \Sigma_2);$$

GMM with 2 gaussians

**Hidden Variable ?**

**Model Parameters ?**




PDF of Guassian: $$\frac{1}{(2\pi)^{d/2} |\Sigma_j|^{1/2}} e^{-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1} (x-\mu_j)}$$

# (Soft) EM Algorithm

The EM algorithm seeks to find the maximum likelihood estimate of the marginal likelihood by iteratively applying these two steps:

- **E-step (Expectation):** infer the current conditional distribution $P(\mathbf{Z} \mid \mathbf{X}, \Theta^t)$ based on $\Theta^t$, and compute the expectation of the log-likelihood function with respect to $\mathbf{Z}$:

$$Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^t) = \mathbb{E}_{\mathbf{Z}|\mathbf{X},\boldsymbol{\Theta}^t} LL(\boldsymbol{\Theta} \mid \mathbf{X}, \mathbf{Z})$$

- **M-step (Maximization):** find the parameters that maximize the expected log-likelihood, that is,

$$\boldsymbol{\Theta}^{t+1} = \underset{\boldsymbol{\Theta}}{\arg\max}\, Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^t)$$

# Gaussian Mixture-Model

The likelihood function

$$L(\theta;\mathbf{x},\mathbf{z}) = p(\mathbf{x},\mathbf{z} \mid \theta) = \prod_{i=1}^{n}\prod_{j=1}^{2} \left[f(\mathbf{x}_i;\boldsymbol{\mu}_j,\Sigma_j)\tau_j\right]^{\mathbb{I}(z_i=j)},$$

This **E step** corresponds with setting up this function for Q:

$$\begin{aligned}
Q(\theta \mid \theta^{(t)}) &= \operatorname{E}_{\mathbf{Z}\mid\mathbf{X}=\mathbf{x};\theta^{(t)}}\left[\log L(\theta;\mathbf{x},\mathbf{Z})\right] \\
&= \operatorname{E}_{\mathbf{Z}\mid\mathbf{X}=\mathbf{x};\theta^{(t)}}\left[\log \prod_{i=1}^{n} L(\theta;\mathbf{x}_i,Z_i)\right] \\
&= \operatorname{E}_{\mathbf{Z}\mid\mathbf{X}=\mathbf{x};\theta^{(t)}}\left[\sum_{i=1}^{n} \log L(\theta;\mathbf{x}_i,Z_i)\right] \\
&= \sum_{i=1}^{n} \operatorname{E}_{Z_i\mid X_i=x_i;\theta^{(t)}}\left[\log L(\theta;\mathbf{x}_i,Z_i)\right] \\
&= \sum_{i=1}^{n}\sum_{j=1}^{2} P(Z_i = j \mid X_i = \mathbf{x}_i;\theta^{(t)}) \log L(\theta_j;\mathbf{x}_i,j)
\end{aligned}$$

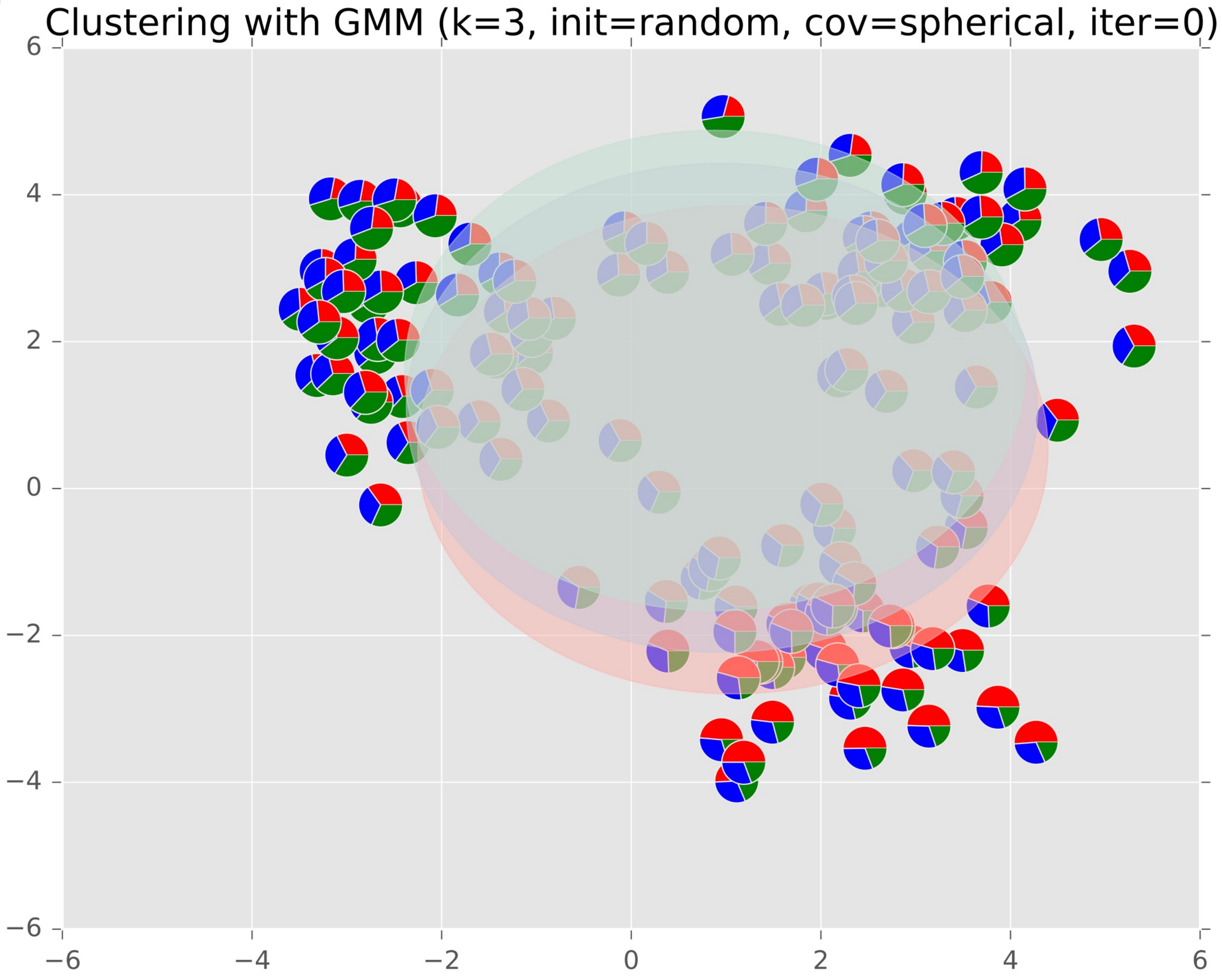
# Gaussian Mixture-Model

**M step**

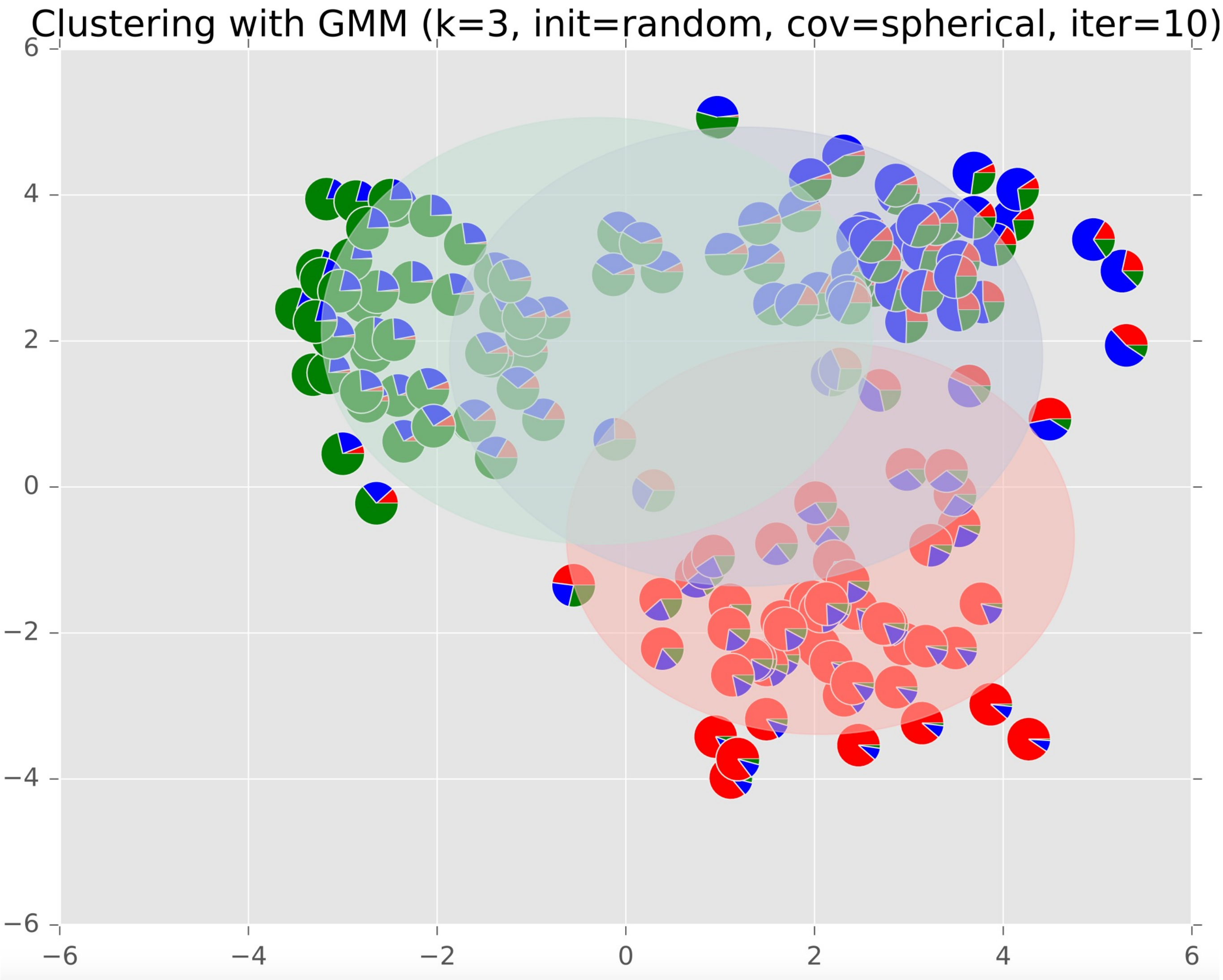
$$\boldsymbol{\tau}^{(t+1)} = \arg\max_{\boldsymbol{\tau}} Q(\theta \mid \theta^{(t)})$$

$$(\boldsymbol{\mu}_1^{(t+1)}, \Sigma_1^{(t+1)}) = \arg\max_{\boldsymbol{\mu}_1, \Sigma_1} Q(\theta \mid \theta^{(t)})$$

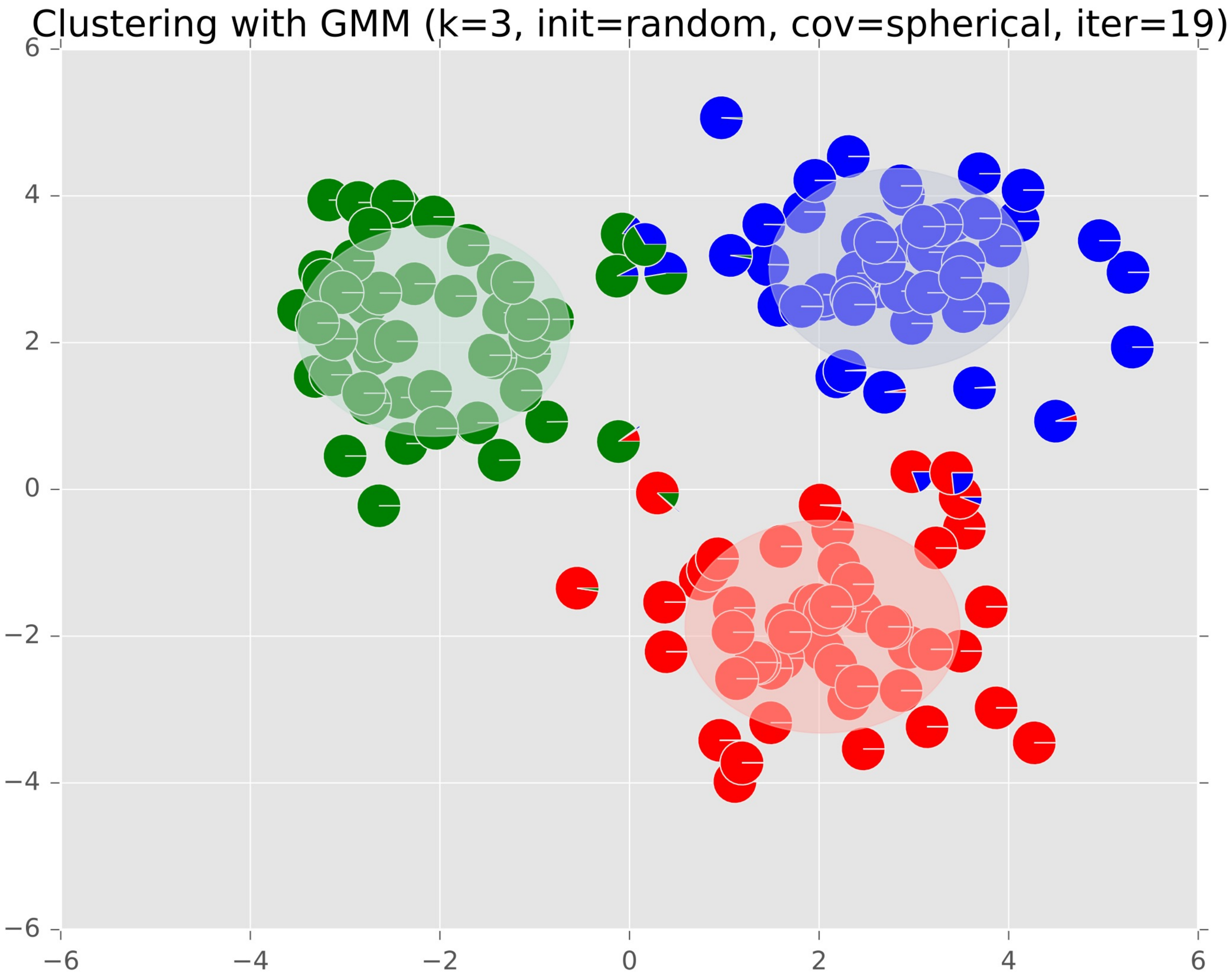
# Example: GMM


Clustering with GMM (k=3, init=random, cov=spherical, iter=0)
6
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6

# Example: GMM


Clustering with GMM (k=3, init=random, cov=spherical, iter=10)
6
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6

# Example: GMM



Clustering with GMM (k=3, init=random, cov=spherical, iter=19)
6
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6

# Relationships between MLE and Q-function

When the outputs are hidden variables, and if $z$ is known, we can turn EM algorithm to MLE in supervised settings.

- supposed that each $\boldsymbol{x}_i$ has a supervised label $y_i$
- defining

$$P\left(z \mid \boldsymbol{x}_i, \Theta^t\right) = \begin{cases} 1 \text{ if } z = y_i \\ 0 \text{ otherwise} \end{cases}$$

$$\begin{aligned} Q\left(\Theta \mid \Theta^t\right) &= \sum_{i=1}^{N} \sum_{\boldsymbol{z} \in Z} P\left(\boldsymbol{z} \mid \boldsymbol{x}_i, \Theta^t\right) \log P(\boldsymbol{x}_i, \boldsymbol{z} \mid \Theta) \\ &= \sum_{i=1}^{N} \log P(\boldsymbol{x}_i, y_i \mid \Theta) \end{aligned}$$

which is exactly the maximum log-likelihood training objective.

# Graphical interpretation

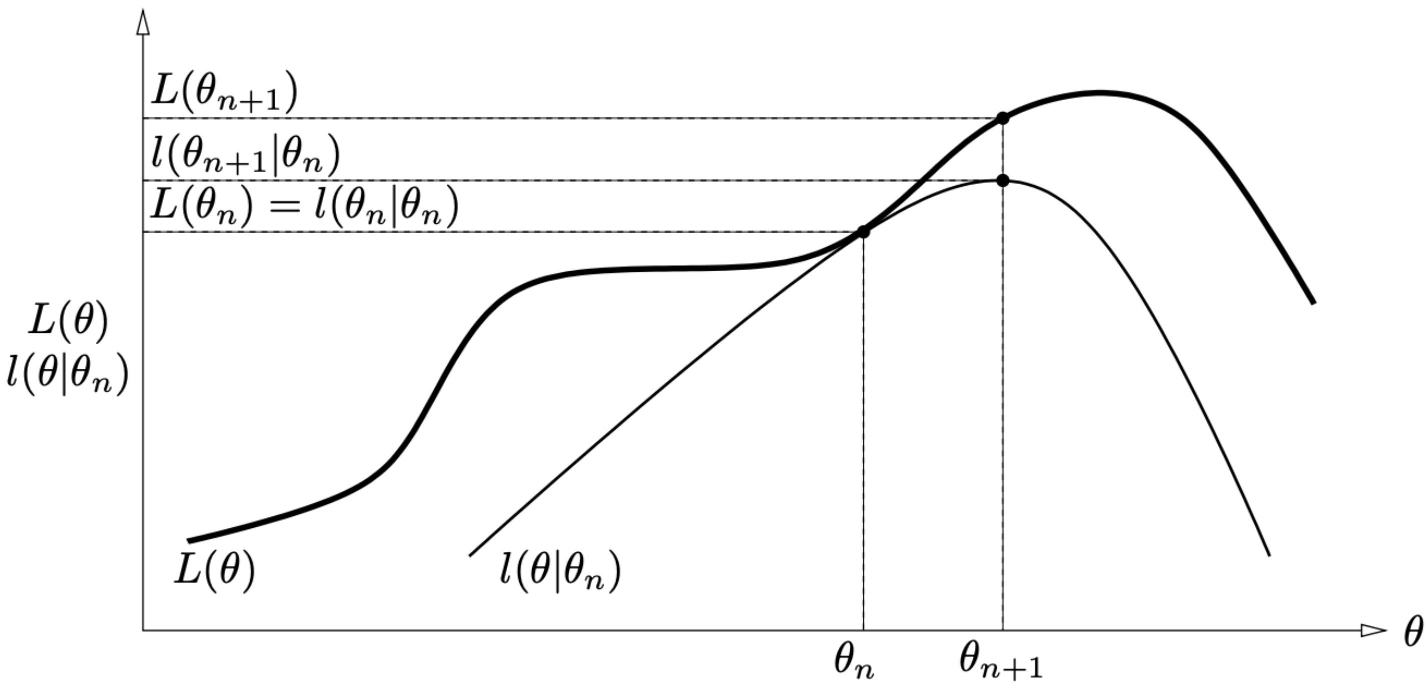



Figure 2: Graphical interpretation of a single iteration of the EM algorithm: The function $l(\theta|\theta_n)$ is upper-bounded by the likelihood function $L(\theta)$. The functions are equal at $\theta = \theta_n$. The EM algorithm chooses $\theta_{n+1}$ as the value of $\theta$ for which $l(\theta|\theta_n)$ is a maximum. Since $L(\theta) \geq l(\theta|\theta_n)$ increasing $l(\theta|\theta_n)$ ensures that the value of the likelihood function $L(\theta)$ is increased at each step.

# Convergence

EM is guaranteed to converge to a point with **zero gradient**.

$$\begin{aligned}
L(\theta) - L(\theta_n) &= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \cdot \frac{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \left( \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
\text{WHY?} \Rightarrow \quad &\geq \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left( \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left( \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)\mathcal{P}(\mathbf{X}|\theta_n)} \right) \\
&\triangleq \Delta(\theta|\theta_n)
\end{aligned}$$

# Convergence

**Theorem 2 (Jensen's inequality)** *Let $f$ be a convex function defined on an interval $I$. If $x_1, x_2, \ldots, x_n \in I$ and $\lambda_1, \lambda_2, \ldots, \lambda_n \geq 0$ with $\sum_{i=1}^n \lambda_i = 1$,*

$$f\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leq \sum_{i=1}^{n} \lambda_i f(x_i)$$

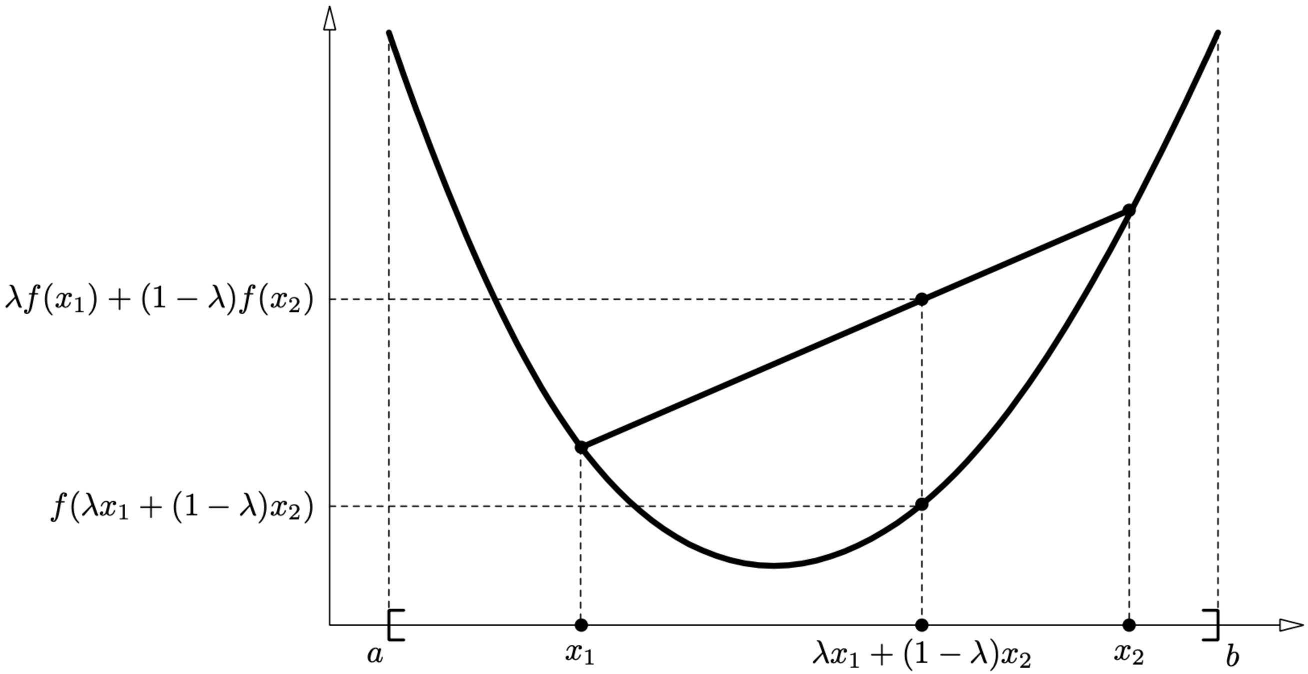



Figure 1: $f$ is *convex* on $[a, b]$ if $f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$ $\forall x_1, x_2 \in [a, b], \quad \lambda \in [0, 1]$.

# Convergence

Equivalently we may write,

$$L(\theta) \geq L(\theta_n) + \Delta(\theta|\theta_n)$$

and for convenience define,

$$l(\theta|\theta_n) \triangleq L(\theta_n) + \Delta(\theta|\theta_n)$$

so that

$$L(\theta) \geq l(\theta|\theta_n).$$

Additionally, observe that,

$$\begin{aligned}
l(\theta_n|\theta_n) &= L(\theta_n) + \Delta(\theta_n|\theta_n) \\
&= L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta_n)\mathcal{P}(\mathbf{z}|\theta_n)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)\mathcal{P}(\mathbf{X}|\theta_n)} \\
&= L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X},\mathbf{z}|\theta_n)}{\mathcal{P}(\mathbf{X},\mathbf{z}|\theta_n)} \\
&= L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln 1 \\
&= L(\theta_n),
\end{aligned}$$

# Convergence

$$
\begin{aligned}
\theta_{n+1} &= \arg\max_{\theta} \{l(\theta|\theta_n)\} \\
&= \arg\max_{\theta} \left\{ L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{X}|\theta_n)\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right\} \\
&\quad \text{Now drop terms which are constant w.r.t. } \theta \\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \right\} \\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \mathcal{P}(\mathbf{X},\mathbf{z}|\theta) \right\} \\
&= \arg\max_{\theta} \left\{ \mathrm{E}_{\mathbf{Z}|\mathbf{X},\theta_n} \{\ln \mathcal{P}(\mathbf{X},\mathbf{z}|\theta)\} \right\}
\end{aligned}
$$

$$
Q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^t) = \mathbb{E}_{\mathbf{Z}|\mathbf{X},\boldsymbol{\Theta}^t} LL(\boldsymbol{\Theta} \mid \mathbf{X}, \mathbf{Z})
$$

# Graphical interpretation

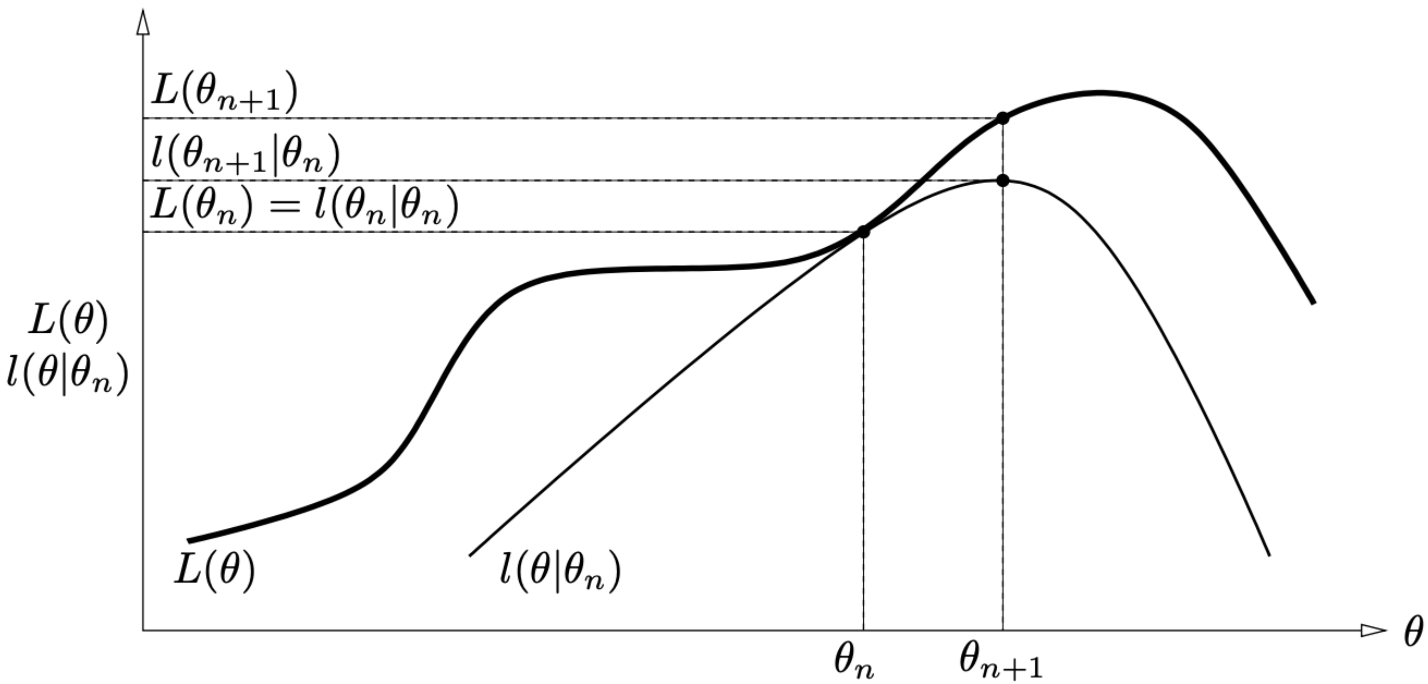
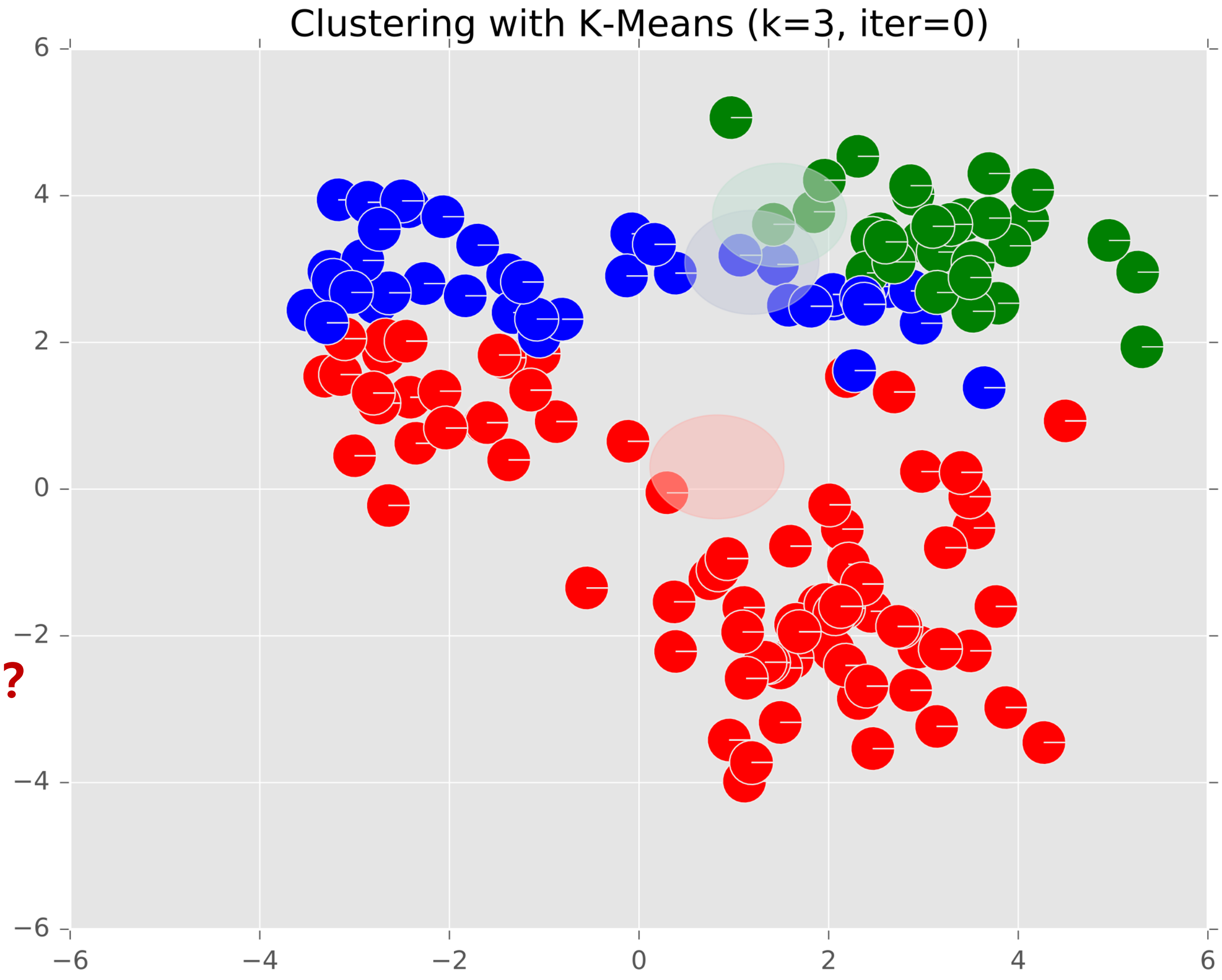



Figure 2: Graphical interpretation of a single iteration of the EM algorithm: The function $l(\theta|\theta_n)$ is upper-bounded by the likelihood function $L(\theta)$. The functions are equal at $\theta = \theta_n$. The EM algorithm chooses $\theta_{n+1}$ as the value of $\theta$ for which $l(\theta|\theta_n)$ is a maximum. Since $L(\theta) \geq l(\theta|\theta_n)$ increasing $l(\theta|\theta_n)$ ensures that the value of the likelihood function $L(\theta)$ is increased at each step.

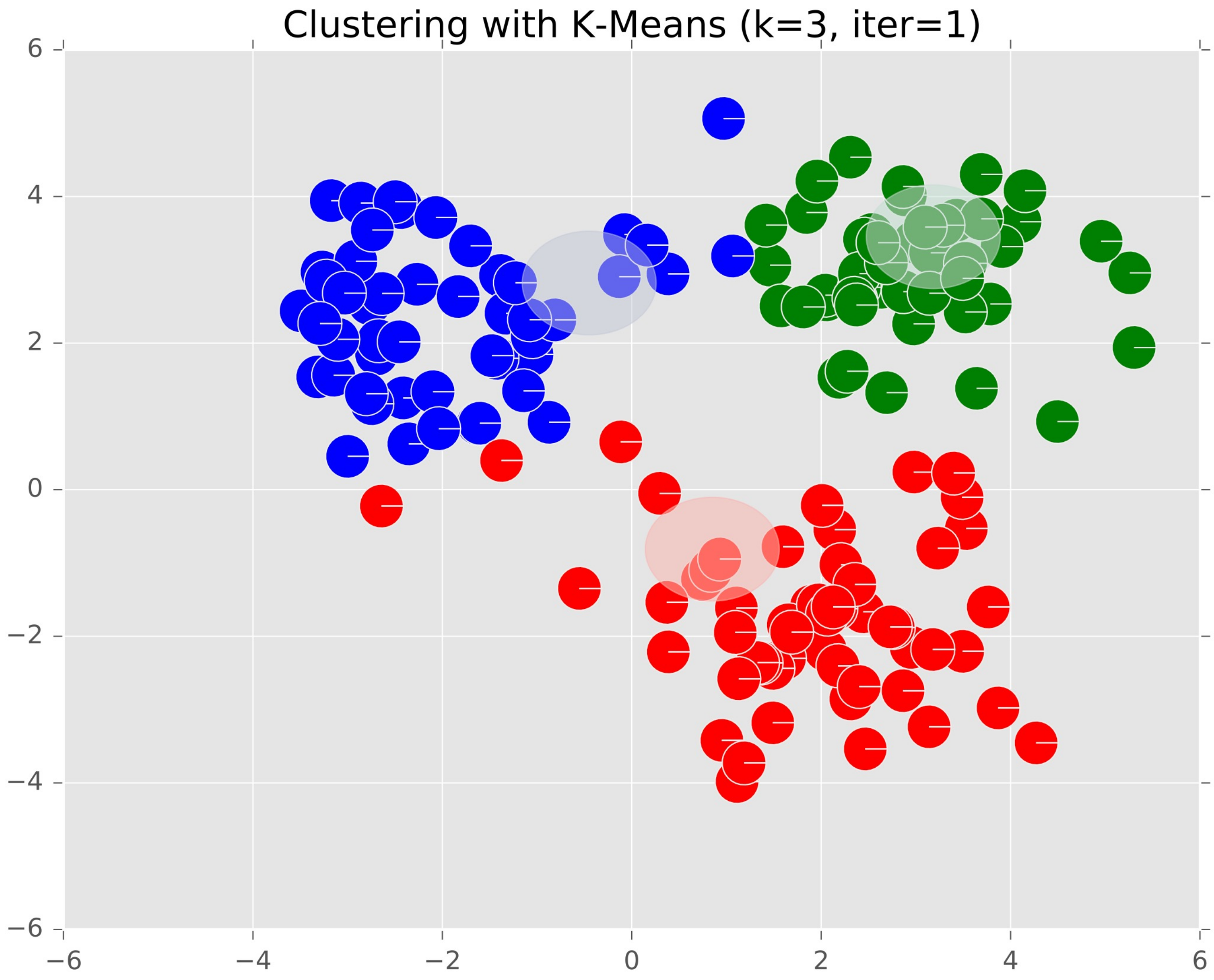
# Example: K-Means



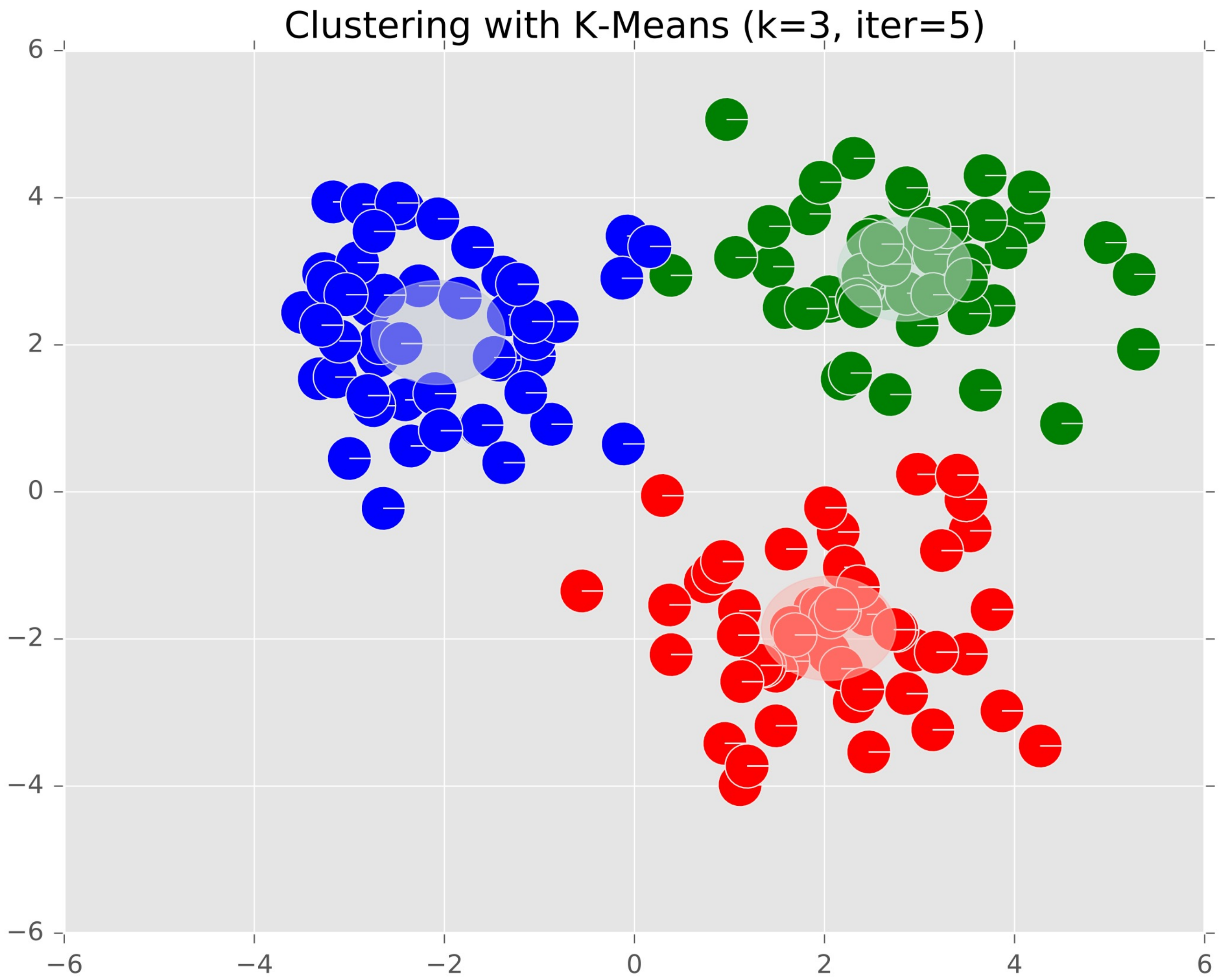

**Hidden Variable ?**

**Model Parameters ?**

# Example: K-Means



# Example: K-Means



# (Hard) EM Algorithm

**repeat**

  **Expectation step:**

  $\mathbf{Z}^t \leftarrow \arg\max_{\mathbf{Z}} \log P(\mathbf{X}, \mathbf{Z} | \Theta^t)$;

  **Maximisation step:**

  $\Theta^{t+1} \leftarrow \arg\max_{\Theta} \log P(\mathbf{X}, \mathbf{Z}^t | \Theta)$;

  $t \leftarrow t + 1$;

**until** $\textsc{Converge}(\mathbf{Z}, \Theta)$;