Lecture 27: More Unsupervised Learning: Generative Models, Mixture of Gaussians, EM Algorithm, K-Means etc

Instructor: Prof. Ganesh Ramakrishnan

Discriminative & Generative Classification Models

- Goal in classification: Assign an input x with feature vector $\phi(\mathbf{x}) \in \Re^m$ to one of K discrete classes C_k where $k \in 1, ..., K$.
- Discriminative Models (so far): Directly model $P(C_i|\phi(\mathbf{x}))$. *E.g.:* Logistic Regression and Neural Networks
- Generative Models: Model $P(\phi(\mathbf{x})|C_i)$ for each i
  - Continuous Attributes $\Rightarrow P(\phi(\mathbf{x})|C_i) \sim \mathcal{N}(\mu_i, \Sigma_i)$ for Gaussian Discriminant Analysis
  - Discrete Attributes $\Rightarrow P(\phi(\mathbf{x})|C_i) \sim Mult(p_1, \ldots, p_m)$ for multivariate Bernoulli Naive Bayes[1]
  - Obtain the posterior using Bayes Rule

[1] Tutorial 10.

Discriminative & Generative Classification Models

- Goal in classification: Assign an input x with feature vector $\phi(\mathbf{x}) \in \Re^m$ to one of K discrete classes C_k where $k \in 1, ..., K$.
- Discriminative Models (so far): Directly model $P(C_i|\phi(\mathbf{x}))$. *E.g.:* Logistic Regression and Neural Networks
- Generative Models: Model $P(\phi(\mathbf{x})|C_i)$ for each i
  - Continuous Attributes $\Rightarrow P(\phi(\mathbf{x})|C_i) \sim \mathcal{N}(\mu_i, \Sigma_i)$ for Gaussian Discriminant Analysis
  - Discrete Attributes $\Rightarrow P(\phi(\mathbf{x})|C_i) \sim Mult(p_1, \ldots, p_m)$ for multivariate Bernoulli Naive Bayes[1]
  - Obtain the posterior using Bayes Rule $P(C_i|\phi(\mathbf{x})) = \frac{P(\phi(\mathbf{x})|C_i)P(C_i)}{\sum_j P(\phi(\mathbf{x})|C_j)P(C_j)}$

[1] Tutorial 10.

Gaussian (Quadratic) Discriminant Analysis

1. A canonical example of Generative Model
2. Example K class case:
$$P(\phi(\mathbf{x})|C_1) = \mathcal{N}(\mu_1, \Sigma_1)$$
$$P(\phi(\mathbf{x})|C_i) = \mathcal{N}(\mu_i, \Sigma_i)$$
$$P(\phi(\mathbf{x})|C_K) = \mathcal{N}(\mu_K, \Sigma_K)$$
3. Assumption: $\phi(\mathbf{x})$ is generated using **exactly one** $\mathcal{N}(\mu_i, \Sigma_i)$
4. In the case of $K = 2$, decision surface will be $\{\phi(\mathbf{x}) \mid P(C_1|\phi(\mathbf{x})) = P(C_2|\phi(\mathbf{x}))\}$. The surface will be **quadratic**
5. Hence, this classifier is also referred to as **Quadratic Discriminant** Analysis (QDA)

Gaussian (Quadratic) Discriminant Analysis[2]

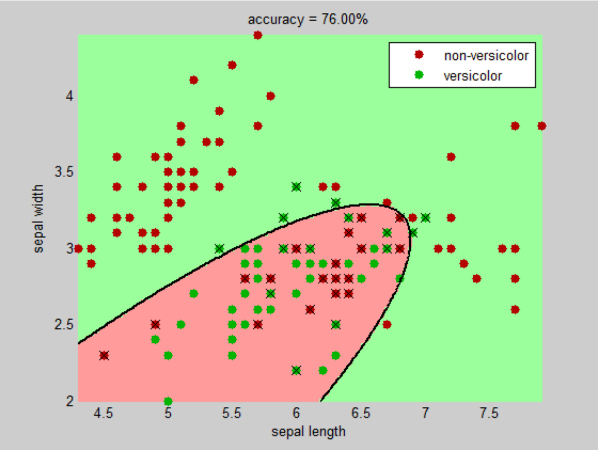

Figure: Illustration of Quadratic Discriminant Analysis

[2] `http://i.stack.imgur.com/0KYBH.png`

Why Quadratic Separating Surface?

- If $\phi(\mathbf{x}) \sim \mathcal{N}(\mu_i, \Sigma_i)$ (where $\phi(\mathbf{x}) \in \Re^m$) then

$$\Pr(\phi(\mathbf{x}) \mid C_i) = \frac{1}{(2\pi)^{\frac{m}{2}} |\Sigma_i|^{\frac{1}{2}}} \, exp \frac{-(\phi(\mathbf{x}) - \mu_i)^T \Sigma_i^{-1} (\phi(\mathbf{x}) - \mu_i)}{2}$$

- So, the separating surface is $\phi(\mathbf{x})$ such that $\{\phi(\mathbf{x}) \mid P(C_1|\phi(\mathbf{x})) = P(C_2|\phi(\mathbf{x}))\}$ that is, $\{\phi(\mathbf{x}) \mid P(\phi(\mathbf{x}) \mid C_1)P(C_1) = P(\phi(\mathbf{x}) \mid C_2)P(C_2)\}$ that is, after taking logs, $\phi(\mathbf{x})$ such that

Why Quadratic Separating Surface?

- If $\phi(\mathbf{x}) \sim \mathcal{N}(\mu_i, \Sigma_i)$ (where $\phi(\mathbf{x}) \in \Re^m$) then

$$\Pr(\phi(\mathbf{x}) \mid C_i) = \frac{1}{(2\pi)^{\frac{m}{2}}|\Sigma_i|^{\frac{1}{2}}} exp \frac{-(\phi(\mathbf{x}) - \mu_i)^T \Sigma_i^{-1}(\phi(\mathbf{x}) - \mu_i)}{2}$$

- So, the separating surface is $\phi(\mathbf{x})$ such that $\{\phi(\mathbf{x}) \mid P(C_1|\phi(\mathbf{x})) = P(C_2|\phi(\mathbf{x}))\}$ that is, $\{\phi(\mathbf{x}) \mid P(\phi(\mathbf{x}) \mid C_1)P(C_1) = P(\phi(\mathbf{x}) \mid C_2)P(C_2)\}$ that is, after taking logs, $\phi(\mathbf{x})$ such that

$$-(\phi(\mathbf{x}) - \mu_1)^T \Sigma_1^{-1}(\phi(\mathbf{x}) - \mu_1) + (\phi(\mathbf{x}) - \mu_2)^T \Sigma_2^{-1}(\phi(\mathbf{x}) - \mu_2) = b$$

  where b contains terms independent of $\phi(\mathbf{x})$.
- This is indeed a quadratic equation!

Maximum Likelihood estimates for QDA

Assuming test point $\mathbf{x}$ belongs to exactly one class, $\Rightarrow$ find C^* such that,

$$C^* = \underset{i}{\operatorname{argmax}} \log[P(\mathbf{x}|C_i)P(C_i)] = \underset{i}{\operatorname{argmax}} \log[\mathcal{N}(\mathbf{x}|\mu_i, \Sigma_i)P(C_i)] \tag{1}$$

We can obtain MLE $\hat{\mu}_i$, $\hat{\Sigma}_i$ and $\widehat{\Pr}(C_i)$ by extending[3] derivations for Multivariate Gaussian and use in (2)

- Setting $\nabla_{\mu_i} LL = 0$, and $\nabla_{\Sigma_i} LL = 0$:

[3] Recap from `lecture-06-unannotated.pdf` as well as extra (optional) accompanying this lecture

Maximum Likelihood estimates for QDA

Assuming test point $\mathbf{x}$ belongs to exactly one class, $\Rightarrow$ find C^* such that,

$$C^* = \underset{i}{\text{argmax}} \ \log[P(\mathbf{x}|C_i)P(C_i)] = \underset{i}{\text{argmax}} \ \log[\mathcal{N}(\mathbf{x}|\mu_i, \Sigma_i)P(C_i)] \tag{1}$$

We can obtain MLE $\hat{\mu}_i$, $\hat{\Sigma}_i$ and $\widehat{\Pr}(C_i)$ by extending[3] derivations for Multivariate Gaussian and use in (2)

- Setting $\nabla_{\mu_i} LL = 0$, and $\nabla_{\Sigma_i} LL = 0$: $\hat{\mu}_i = \frac{1}{m_i}\sum_{j=1}^{m_i} \phi(\mathbf{x}_j^i)$ and

  $$\hat{\Sigma}_i = \frac{1}{m_i}\sum_{j=1}^{m_i}(\phi(\mathbf{x}_j^i) - \hat{\mu}_i)(\phi(\mathbf{x}_j^i) - \hat{\mu}_i)^T$$ ....called the empirical co-variance matrix in statistics

- Also setting $\nabla_{\Pr(C_i)} LL = 0$, $\widehat{\Pr}(C_i) = \frac{m_i}{\sum_{j=1}^{K} m_j}$
- $\hat{\mu}_i \sim N(\mu_i, \Sigma_i)$ and since $\text{E}[\hat{\mu}_i] = \mu_i$, $\hat{\mu}_i$ is an unbiased estimator. [Extra optional slides]
- **Naive Bayes Classifier: Each Σ_i assumed to be diagonal**

[3] Recap from `lecture-06-unannotated.pdf` as well as extra (optional) accompanying this lecture

Bayesian estimation for QDA

Assuming test point $\mathbf{x}$ belongs to exactly one class, $\Rightarrow$ find C^* such that,

$$C^* = \underset{i}{\operatorname{argmax}} \ \log[P(\mathbf{x}|C_i)P(C_i)] = \underset{i}{\operatorname{argmax}} \ \log[\mathcal{N}(\mathbf{x}|\mu_i, \Sigma_i)P(C_i)] \tag{2}$$

We can obtain MAP $\hat{\mu}_i$, $\hat{\Sigma}_i$ and $\widehat{\Pr}(C_i)$ by extending[4] derivations for Multivariate Gaussian and use in (2)

- Extending to Bayesian setting[5] for multivariate case with fixed (non-probabilistic) Σ_i
$$\phi(\mathbf{x} \mid C_i) \sim \mathcal{N}(\mu_i, \Sigma_i), \ \mu_i \sim \mathcal{N}(\mu_i^0, \Sigma_i^0) \ \Rightarrow \Pr(\mu_i|\mathcal{D}) = \mathcal{N}(\mu_i^{m_i}, \Sigma_i^{m_i})$$

$$\left(\Sigma_i^{m_i}\right)^{-1} = \left(\Sigma_i^0\right)^{-1} + m_i \left(\Sigma_i\right)^{-1}$$

$$\left(\Sigma_i^{m_i}\right)^{-1} \mu_i^{m_i} = m_i \left(\Sigma_i\right)^{-1} \hat{\mu}_{mle} + \left(\Sigma_i^0\right)^{-1} \mu_i^0$$

MAP estimates $\mu_i^{m_i}$ and $\Sigma_i^{m_i}$ are obtained by solving above linear system.

- As before, $\widehat{\Pr}(C_i) = \frac{m_i}{\sum_{j=1}^{K} m_j}$

[4] Recap from `lecture-06-unannotated.pdf` as well as extra (optional) accompanying this lecture

[5] `https://en.wikipedia.org/wiki/Multivariate_normal_distribution#Bayesian_inference`

Tutorial 10

- Suppose, in our generative model, the points from each class are generated using a multivariate Gaussian with a different mean μ_i for each class C_i, but a shared covariance matrix Σ:

$$P(\phi(\mathbf{x})|C_i) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} \exp \frac{-(\phi(\mathbf{x}) - \mu_i)^T \Sigma^{-1} (\phi(\mathbf{x}) - \mu_i)}{2}$$

- Show that the Maximum Likelihood estimates are:

$$\hat{\mu}_i = \frac{1}{m_i} \sum_{\mathbf{x} \in C_i} \phi(\mathbf{x})$$

$$\hat{\Sigma} = \frac{1}{m} \sum_{i=1}^{K} \sum_{\mathbf{x} \in C_i} (\phi(\mathbf{x}) - \mu_i)(\phi(\mathbf{x}) - \mu_i)^T$$

- In fact, this has a **Linear** separating surface and is therefore called **Linear Discriminant Analysis** (LDA)

Tutorial 10: Linear Discriminant Analysis[6]

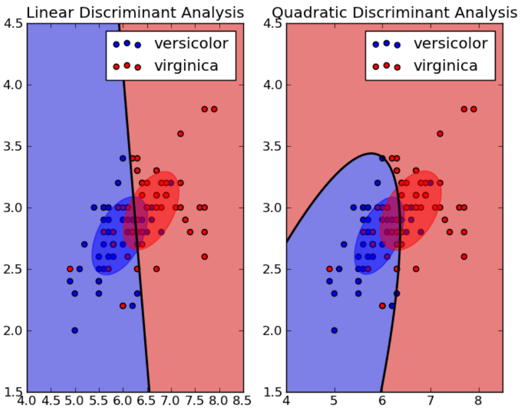

Figure: Illustration of Linear vs. Quadratic Discriminant Analysis

[6] http://scikit-learn.sourceforge.net/0.6/auto_examples/plot_lda_vs_qda.html

Unsupervised Mixture of Gaussians

1. Recall assumption: $\phi(\mathbf{x})$ is generated using **exactly one** $\mathcal{N}(\mu_i, \Sigma_i)$
2. What if this assumption was violated?

Unsupervised Mixture of Gaussians

1. Recall assumption: $\phi(\mathbf{x})$ is generated using **exactly one** $\mathcal{N}(\mu_i, \Sigma_i)$
2. What if this assumption was violated?
   - What if an example $\phi(\mathbf{x})$ belongs to multiple classes (Gaussians)?

$$\Pr(\phi(\mathbf{x})|C_p) = \mathcal{N}(\mu_p, \Sigma_p)$$

$$\Pr(\phi(\mathbf{x})|C_q) = \mathcal{N}(\phi(\mathbf{x}) \mid \mu_q, \Sigma_q)$$

   - What if the membership of an example to the different classes is not known?

$$\Pr(\phi(\mathbf{x})) = \sum_{i=1}^{K} \Pr(\phi(\mathbf{x}), C = z_i) = \sum_{i=1}^{K} \Pr(C = z_i)\mathcal{N}(\phi(\mathbf{x}), \mu_i, \Sigma_i)$$

Unsupervised Mixture

- $Z \in \{z_1, z_2, \ldots, z_k\}$: Multinomial variable indicating mixture component & $K =$ number of (hidden) classes or mixture components
- $\phi(\mathbf{x})$: Random variable (vector), with distribution specified, conditioned on different values z_i of Z

$$\Pr(\phi(\mathbf{x}) \mid z_i; \theta_i) \sim f_i(x; \theta_i)$$

- The finite mixture model is defined as

[7] Proportion of the population in subpopulation i.

Unsupervised Mixture

- $Z \in \{z_1, z_2, \ldots, z_k\}$: Multinomial variable indicating mixture component & K = number of (hidden) classes or mixture components
- $\phi(\mathbf{x})$: Random variable (vector), with distribution specified, conditioned on different values z_i of Z

$$\Pr(\phi(\mathbf{x}) \mid z_i; \theta_i) \sim f_i(x; \theta_i)$$

- The finite mixture model is defined as

$$\Pr(\phi(\mathbf{x})) = \sum_{i=1}^{K} \Pr(z_i) f_i(\mathbf{x}; \theta_i) = \sum_{i=1}^{K} \pi_i f_i(\mathbf{x}; \theta_i)$$

  $\pi = [\pi_1, \pi_2, \ldots, \pi_k]$ and $\theta = [\theta_1, \theta_2, \ldots, \theta_k]$ are the paramaters of the mixture model, with a fixed value of k.

- Quantities $\Pr(z_i) = \pi_i$ are mixing weights[7]

[7]Proportion of the population in subpopulation i.

Example: Gaussian Mixture Model (GMM)

- The density of each mixture component is Gaussian with $\theta_i = (\mu_i, \Sigma_i)$.

$$f_i(\phi(\mathbf{x}); \theta_i) = \mathcal{N}(\phi(\mathbf{x}) \mid \mu_i, \Sigma_i)$$

- $\Pr(\phi(\mathbf{x}))$ is then called a mixture of Gaussian

$$\Pr(\phi(\mathbf{x}) \mid z_i; \theta_i) \sim f_i(x; \theta_i)$$

  Gaussian Mixture Model (GMM) is itself NOT a Gaussian!

- Supervised setting: We learnt (μ_i, Σ_i) using Maximum Likelihood/MAP when a (unique) z was observed for an x
- Unsupervised setting: Learning parameters $\theta_i = (\mu_i, \Sigma_i)$ in the presence of incomplete data (only instances of $\phi(\mathbf{x})$)

Parameter Estimation for Mixture Models[8]

- Decomposition of the joint distribution

$$\Pr(\phi(\mathbf{x}), z; \theta) = \Pr(z) \Pr(\phi(\mathbf{x}) \mid z, \theta)$$

- The (log) likelihood to be maximized:

$$LL(\pi, \theta; \phi(\mathbf{x})) = \frac{1}{m} \sum_{j=1}^{m} \log \Pr\left(\phi\left(\mathbf{x}^{(j)}\right); \theta\right) = \frac{1}{m} \sum_{j=1}^{m} \log \left[\sum_{l=1}^{K} \pi_l f_l \left(\phi\left(\mathbf{x}^{(j)}\right); \theta_l\right)\right]$$

s.t. $\pi_l \geq 0$ and $\sum_{l=1}^{K} \pi_l = 1$.

- Problem:

[8] Section 7.8 onwards of cs725/notes/classNotes/misc/CaseStudyWithProbabilisticModels.pdf

Parameter Estimation for Mixture Models[8]

- Decomposition of the joint distribution

$$\Pr(\phi(\mathbf{x}), z; \theta) = \Pr(z)\Pr(\phi(\mathbf{x}) \mid z, \theta)$$

- The (log) likelihood to be maximized:

$$LL(\pi, \theta; \phi(\mathbf{x})) = \frac{1}{m}\sum_{j=1}^{m} \log \Pr\left(\phi\left(\mathbf{x}^{(j)}\right); \theta\right) = \frac{1}{m}\sum_{j=1}^{m} \log\left[\sum_{l=1}^{K} \pi_l f_l\left(\phi\left(\mathbf{x}^{(j)}\right); \theta_l\right)\right]$$

s.t. $\pi_l \geq 0$ and $\sum_{l=1}^{K} \pi_l = 1$.

- Problem: log cannot be distributed over a summation!!

[8] Section 7.8 onwards of cs725/notes/classNotes/misc/CaseStudyWithProbabilisticModels.pdf

Parameter Estimation for **Gaussian** Mixture Models

- Need to maximize $LL(\pi, \mu, \Sigma; \phi(\mathbf{x})) = \frac{1}{m}\sum_{j=1}^{m} \log\left[\sum_{l=1}^{K} \pi_l \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_l, \Sigma_l\right)\right]$ s.t $\pi_i \geq 0$ and $\sum_{l=1}^{K} \pi_l = 1$.
- Write down the necessary optimality conditions for this maximization problem, subject to its associated inequality and linear equality constraints
- Setting gradient w.r.t each μ_i to 0 we get:

Parameter Estimation for **Gaussian** Mixture Models

- Need to maximize $LL(\pi, \mu, \Sigma; \phi(\mathbf{x})) = \frac{1}{m}\sum_{j=1}^{m} \log\left[\sum_{l=1}^{K} \pi_l \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_l, \Sigma_l\right)\right]$ s.t $\pi_i \geq 0$ and $\sum_{l=1}^{K} \pi_l = 1$.
- Write down the necessary optimality conditions for this maximization problem, subject to its associated inequality and linear equality constraints
- Setting gradient w.r.t each μ_i to 0 we get:

$$\sum_{j=1}^{m} \frac{\pi_i \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_i, \Sigma_i\right)}{\left[\sum_{l=1}^{K} \pi_l \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_l, \Sigma_l\right)\right]} \Sigma_i^{-1}\left(\phi\left(\mathbf{x}^{(j)}\right) - \mu_i\right) = 0$$

Σ_i^{-1} is non-singular and therefore remaining expression must be 0.

The EM Trick

$$\sum_{j=1}^{m} \frac{\pi_i \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_i, \Sigma_i\right)}{\left[\sum_{l=1}^{K} \pi_l \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_l, \Sigma_l\right)\right]} \left(\phi\left(\mathbf{x}^{(j)}\right) - \mu_i\right) = 0 \tag{3}$$

- No way to solve this in closed form to get a clean MLE estimate for μ_i!

The EM Trick

$$\sum_{j=1}^{m} \frac{\pi_i \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_i, \Sigma_i\right)}{\left[\sum_{l=1}^{K} \pi_l \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_l, \Sigma_l\right)\right]} \left(\phi\left(\mathbf{x}^{(j)}\right) - \mu_i\right) = 0 \tag{3}$$

- No way to solve this in closed form to get a clean MLE estimate for μ_i!
- Note that $\frac{\pi_i \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_i, \Sigma_i\right)}{\left[\sum_{l=1}^{K} \pi_l \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_l, \Sigma_l\right)\right]} = \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)$ and comprises the **E-Step**.
- Pretending as if $\Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)$ is independent of μ_i and Σ_i in (3),

The EM Trick

$$\sum_{j=1}^{m} \frac{\pi_i \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_i, \Sigma_i\right)}{\left[\sum_{l=1}^{K} \pi_l \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_l, \Sigma_l\right)\right]} \left(\phi\left(\mathbf{x}^{(j)}\right) - \mu_i\right) = 0 \tag{3}$$

- No way to solve this in closed form to get a clean MLE estimate for μ_i!
- Note that $\frac{\pi_i \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_i, \Sigma_i\right)}{\left[\sum_{l=1}^{K} \pi_l \mathcal{N}\left(\phi\left(\mathbf{x}^{(j)}\right); \mu_l, \Sigma_l\right)\right]} = \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)$ and comprises the **E-Step**.
- Pretending as if $\Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)$ is independent of μ_i and Σ_i in (3),
- We get the **M-Step**: $\mu_i = \frac{\sum_{j=1}^{m} \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right) \phi\left(\mathbf{x}^{(j)}\right)}{\sum_{j=1}^{m} \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)}$

M-Step using (Approximate) Necessary Optimality conditions for GMM

M-Step or the Maximization Step

$$\mu_i = \frac{\sum_{j=1}^{m} \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right) \phi\left(\mathbf{x}^{(j)}\right)}{\sum_{j=1}^{m} \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)}$$

$$\Sigma_i = \frac{\sum_{j=1}^{m} \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right) \left(\phi\left(\mathbf{x}^{(j)}\right) - \mu_i\right) \left(\phi\left(\mathbf{x}^{(j)}\right) - \mu_i\right)^T}{\sum_{j=1}^{m} \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)}$$

$$\pi_i = \frac{1}{m} \sum_{j=1}^{m} \Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)$$

E-Step using Bayes Rule for GMM

E-Step or the Expectation Step

For the posterior $\Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right)$

$$\Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right)\right) = \frac{\pi_i \mathcal{N}(\phi(\mathbf{x}); \mu_i, \Sigma_i)}{\sum_{l=1}^{K} \pi_l \mathcal{N}(\phi(\mathbf{x}); \mu_l, \Sigma_l)}$$

Revisiting E and M-Steps for GMM

Example 2 EM algorithm [Bishop book[1] and its web site]

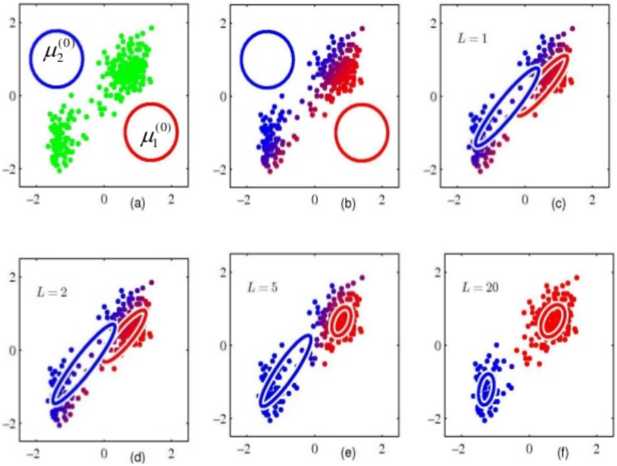

Figure: Illustration of EM on Mixture of Gaussians

EM More Formally: Reflections on the E and M Steps

- Necessary Optimality conditions do not yield any closed form solution
- Instead, one can continuously alternate between the **E-Step** and the **M-Step** until convergence
- This is the idea behind the **EM Algorithm**
- We will explain the EM Algorithm for the more general complete data loglikelihood formulation

$$LL(\theta; \phi(\mathbf{x}), \mathbf{z}) = \frac{1}{m} \log \Pr(\phi(\mathbf{x}), \mathbf{z}; \theta)$$

  and show its convergence

The EM Algorithm: More generally

- Given a predictive distribution $q(\mathbf{z}|\phi(\mathbf{x}))$, the expected complete data log-likelihood is

$$LL_E(\theta; \phi(\mathbf{x})) = \sum_{\mathbf{z}} q(\mathbf{z}|\phi(\mathbf{x})) \log \Pr(\phi(\mathbf{x}), \mathbf{z}; \theta)$$

  is an auxilliary function that gives a lower bound on the actual log-likelihood we want to optimize
- The actual log-likelihood under iid assumption is:

$$LL(\theta; \phi(\mathbf{x})) = \frac{1}{m} \sum_{i=1}^{m} \log \left\{ \sum_{\mathbf{z}} \Pr(\phi\left(\mathbf{x}^{(i)}\right), \mathbf{z}; \theta) \right\}$$

Lower-bound Theorem

For all θ and every possible distribution $q(\mathbf{z}|\phi(\mathbf{x}))$:

$$LL(\theta;\phi(\mathbf{x})) \geq LL_E(\theta;\phi(\mathbf{x})) + \frac{1}{m}H(q)$$

Equality holds if and only if

$$q(\mathbf{z}|\phi(\mathbf{x})) = \Pr(\mathbf{z}|\phi(\mathbf{x});\theta)$$

Proof: **(Optional)**

[9] That is, invoking Jensen's inequality

Lower-bound Theorem

For all θ and every possible distribution $q(\mathbf{z}|\phi(\mathbf{x}))$:

$$LL(\theta;\phi(\mathbf{x})) \geq LL_E(\theta;\phi(\mathbf{x})) + \frac{1}{m}H(q)$$

Equality holds if and only if

$$q(\mathbf{z}|\phi(\mathbf{x})) = \Pr(\mathbf{z}|\phi(\mathbf{x});\theta)$$

Proof: **(Optional)**

$$LL(\theta;\phi(\mathbf{x})) = \frac{1}{m}\log\left\{\sum_{\mathbf{z}} q(\mathbf{z}|\phi(\mathbf{x}))\frac{\Pr(\mathbf{z}|\phi(\mathbf{x});\theta)}{q(\mathbf{z}|\phi(\mathbf{x}))}\right\}$$

Since log is a strictly concave function[9]

$$LL(\theta;\phi(\mathbf{x})) \geq \underbrace{\frac{1}{m}\sum_{\mathbf{z}} q(\mathbf{z}|\phi(\mathbf{x}))\log\Pr(\phi(\mathbf{x}),\mathbf{z};\theta)}_{LL_E(\theta;\phi(\mathbf{x}))} - \underbrace{\frac{1}{m}\sum_{\mathbf{z}} q(\mathbf{z}|\phi(\mathbf{x}))\log q(\mathbf{z}|\phi(\mathbf{x}))}_{H(q)}$$

[9] That is, invoking Jensen's inequality

Proof continued **(Optional)**

Equality holds if and only if $\frac{\Pr(\mathbf{z}|\phi(\mathbf{x});\theta)}{q(\mathbf{z}|\phi(\mathbf{x}))}$ is a constant, that is,

$$q(\mathbf{z}|\phi(\mathbf{x})) \propto \Pr(\phi(\mathbf{x}), \mathbf{z}; \theta) = \Pr(\mathbf{z}|\phi(\mathbf{x}); \theta) \Pr(\phi(\mathbf{x}); \theta) \propto \Pr(\mathbf{z}|\phi(\mathbf{x}); \theta)$$

This can happen if and only if $q(\mathbf{z}|\phi(\mathbf{x})) = \Pr(\mathbf{z}|\phi(\mathbf{x}); \theta)$.

$\square$

EM Algo as Coordinate Descent on Lower Bound

$$\max_{\theta} LL(\theta;\phi(\mathbf{x})) \geq \max_{\theta} \max_{q} LL_E(\theta;\phi(\mathbf{x})) + \frac{1}{m}H(q)$$

The EM algorithm is simply coordinate ascent on the auxilliary function $LL_E(\theta;\phi(\mathbf{x})) + \frac{1}{m}H(q)$.

- **Expectation Step** t: $q^{(t+1)} = \underset{q}{\text{argmax}}\ LL_E(\theta^{(t)};\phi(\mathbf{x})) + \frac{1}{m}H(q)$

  $= \underset{q}{\text{argmax}}\ -D\left(q(\mathbf{z}|\phi(\mathbf{x}))||\Pr(\mathbf{z}|\phi(\mathbf{x});\theta^{(t)})\right) + \log\left\{\phi(\mathbf{x});\theta^{(t)}\right\}$
- Since, $LL_E(\theta^{(t)};\phi(\mathbf{x})) + \frac{1}{m}H(q) \leq \log\left\{\phi(\mathbf{x});\theta^{(t)}\right\}$, maximum value is attained for $q(\mathbf{z}|\phi(\mathbf{x})) = \Pr(\mathbf{z}|\phi(\mathbf{x});\theta^{(t)})$
- Thus, the E-step can be summarized by

$$q^{(t+1)}(\mathbf{z}|\phi(\mathbf{x})) = \Pr(\mathbf{z}|\phi(\mathbf{x});\theta^{(t)}) \tag{4}$$

Special Case: Revisiting E-step for GMM (Tutorial 10)

Initialize $\mu_i^{(0)}$ to different random values and $\Sigma_i^{(0)}$ to I

For the posterior $Pr\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \mu, \Sigma\right)$

$$Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \mu, \Sigma\right) = \frac{\pi_i^{(t)} \mathcal{N}\left(\phi(\mathbf{x}); \mu_i^{(t)}, \Sigma_i^{(t)}\right)}{\sum_{l=1}^{K} \pi_l^{(t)} \mathcal{N}\left(\phi(\mathbf{x}); \mu_l^{(t)}, \Sigma_l^{(t)}\right)}$$

EM Algo as Coordinate Descent on Lower Bound

$$\max_{\theta} LL(\theta; \phi(\mathbf{x})) \geq \max_{\theta} \max_{q} LL_E(\theta; \phi(\mathbf{x})) + \frac{1}{m} H(q)$$

The EM algorithm is simply coordinate ascent on the auxilliary function $LL_E(\theta; \phi(\mathbf{x})) + \frac{1}{m} H(q)$.

- **Maximization Step** t: Since $H(q)$ is independent of θ, $\theta^{(t+1)} = \underset{\theta}{\operatorname{argmax}}\ LL_E(\theta; \phi(\mathbf{x})) + \frac{1}{m} H(q^{(t+1)}) = \underset{\theta}{\operatorname{argmax}} \sum_{\mathbf{z}} q(\mathbf{z}|\phi(\mathbf{x})) \log \Pr(\phi(\mathbf{x}), \mathbf{z}; \theta)$
- Like ordinary maximum likelihood estimation problem, but using predicted values of $\mathbf{z}$.
- The M-step may not have a closed form solution, in which case, it may be required to resort to approximation techniques.

Special Case: Revisiting **M-step** for GMM (Tutorial 10)

$$\mu_i^{(t+1)} = \frac{\sum_{j=1}^{m} Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \theta\right)\phi\left(\mathbf{x}^{(j)}\right)}{\sum_{j=1}^{m} Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \theta\right)}$$

$$\Sigma_i^{(t+1)} = \frac{\sum_{j=1}^{m} Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \theta\right)\left(\phi\left(\mathbf{x}^{(j)}\right) - \mu_i^{(t+1)}\right)\left(\phi\left(\mathbf{x}^{(j)}\right) - \mu_i^{(t+1)}\right)^T}{\sum_{j=1}^{m} Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \theta\right)}$$

$$\pi_i^{(t+1)} = \frac{1}{m}\sum_{j=1}^{m} Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \theta\right)$$

EM for GMM: Summary

1. Initialize $\mu_i^{(0)}$ to different random values and $\Sigma_i^{(0)}$ to I. Let $t = 0$.
2. Compute $Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \theta\right)$ using $\mu_i^{(t)}$ and $\Sigma_i^{(t)}$
3. Compute $\pi_i^{(t+1)}$ and $\mu_i^{(t+1)}$ using $Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \theta\right)$ and $\Sigma_i^{(t+1)}$ using $Pr^{(t+1)}\left(z_i \middle| \phi\left(\mathbf{x}^{(j)}\right), \theta\right)$ and $\mu_i^{(t+1)}$
4. If parameters have changed significantly, increment t by 1 and go back to Step 2.

K-Means Clustering Algorithm or Hard EM

1. Initialize $\mu_i^{(0)}$ to different random values. Let $t = 0$.
2. Posterior $\Pr(z_i \mid \phi\left(\mathbf{x}^{(j)}\right), \theta) \in [0,1]$ replaced by $P_{i,j} \in \{0,1\}$. Compute cluster memberships $P_{i,j}$ that minimize the sum of squared distance of points to existing centroids
3. Compute $\mu_i^{(t+1)}$ that minimize the sum of squared distance of points to the centroid of the cluster assigned in the previous iteration
4. If parameters have changed, increment t by 1 and go back to Step 2.

K-Means Clustering Algorithm or Hard EM

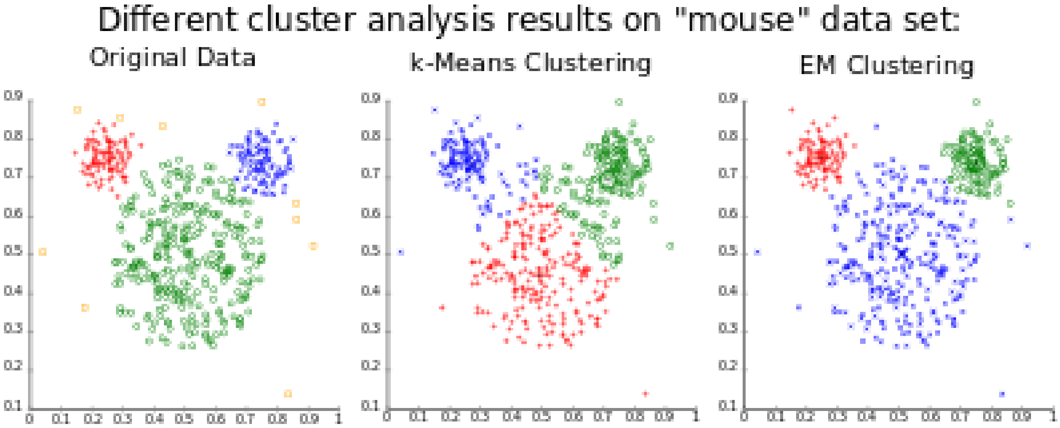

Figure: Comparison of K-Means with EM (Mixture of Gaussians). Source: Wikipedia

K-Means Clustering Algorithm or Hard EM

1. Initialize $\mu_i^{(0)}$ to different random values. Let $t = 0$.
2. $Pr^{(t+1)} \in \underset{P}{\text{argmin}} \sum_{j=1}^{m} \sum_{l=1}^{K} P_{l,j} \|\phi\left(\mathbf{x}^{(j)}\right) - \mu_l^{(t)}\|^2$

   Solution: For each $j \in [1..n]$, $i^* = \underset{l}{\text{argmin}} \|\phi\left(\mathbf{x}^{(j)}\right) - \mu_l^{(t)}\|^2$, $Pr_{i^*,j}^{(t+1)} = 1$ and $P_{l,j}^{(t+1)} = 0$ for $l \neq i^*$.

K-Means Clustering Algorithm or Hard EM

1. Initialize $\mu_i^{(0)}$ to different random values. Let $t = 0$.
2. $Pr^{(t+1)} \in \underset{P}{\operatorname{argmin}} \sum_{j=1}^{m} \sum_{l=1}^{K} P_{l,j} \|\phi\left(\mathbf{x}^{(j)}\right) - \mu_l^{(t)}\|^2$

   Solution: For each $j \in [1..n]$, $i^* = \underset{l}{\operatorname{argmin}} \|\phi\left(\mathbf{x}^{(j)}\right) - \mu_l^{(t)}\|^2$, $Pr_{i^*,j}^{(t+1)} = 1$ and $P_{l,j}^{(t+1)} = 0$ for $l \neq i^*$.
3. $\mu^{(t+1)} = \underset{\mu}{\operatorname{argmin}} \sum_{j=1}^{m} \sum_{l=1}^{K} P_{l,j}^{(t+1)} \|\phi\left(\mathbf{x}^{(j)}\right) - \mu_l\|^2$

   Solution: $\mu_i^{(t+1)} = \dfrac{\sum_{j=1}^{m} P_{i,j}^{(t+1)} \phi\left(\mathbf{x}^{(j)}\right)}{\sum_{j=1}^{m} P_{i,j}^{(t+1)}}$.
4. If any parameter $P_{i,j}$ has changed, increment t by 1 and go back to **Step 2**.

K-Means Clustering Algorithm or Hard EM (Tutorial 10)

1. **Claim:** The K-Means Clustering algorithm will converge in a finite number of iterations
2. **Proof Sketch:** At each iteration, the K-Means algorithm reduces the objective $\sum_{j=1}^{m}\sum_{l=1}^{K} P_{l,j}\|\phi\left(\mathbf{x}^{(j)}\right) - \mu_l\|^2$ and stops when this objective does not reduce any further.
3. *Hint1:* $P^{(t+1)} = \underset{P}{\text{argmin}} \sum_{j=1}^{m}\sum_{l=1}^{K} P_{l,j}\|\phi\left(\mathbf{x}^{(j)}\right) - \mu_l^{(t)}\|^2$
4. *Hint2:* $\mu^{(t+1)} = \underset{\mu}{\text{argmin}} \sum_{j=1}^{m}\sum_{l=1}^{K} P_{l,j}^{(t+1)}\|\phi\left(\mathbf{x}^{(j)}\right) - \mu_l\|^2$
5. *Hint3:* Only a finite number of combinations of $P_{i,j}$ are possible.

Disadvantages of K-means & Alternatives

1. Fixed value of K: Right value of K critical to success
2. Sometimes problem owing to the wrong initialization of μ_j's
3. Mean in "no-man's land": Lack of robustness to outliers

Variants of K-means[10]

1. *K-mediods:* Assumption is Cluster's centroid coincides with one of the points. That is, $\mu_i = \phi\left(\mathbf{x}^{(j)}\right)$ for some value of j.
   $\Rightarrow$ Each step of the K-mediod algorithm is $K(n-1)n \sim \mathcal{O}(Kn^2)$
2. *K-modes:* For discrete valued attributes:

$$x[\mu_i]_q = \underset{v \in \{1,\dots V_q\}}{\text{argmax}} \sum_{\mathbf{x}^{(j)} \in C_i} \delta\left(\phi_q\left(\mathbf{x}^{(j)}\right), v\right) \quad \forall q = 1 \dots m$$

[10]For more details read Chapter 7 of Jiawei Han's book

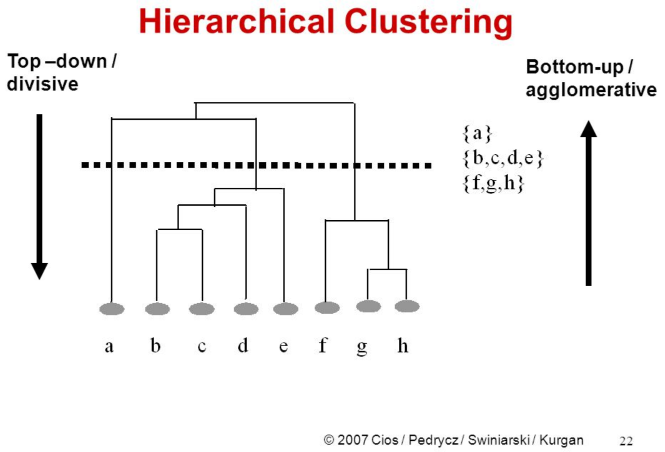

Figure: Bottom-up and Top-down Hierarchical Clustering

Hierarchical Clustering: Two Choices

1. Bottom-up (agglomerative)
2. Top-down (divisive)

Main idea: Iteratively merge clusters that are closest (or break clusters that are furthest apart): **NEED A NOTION OF DISTANCE BETWEEN POINTS**

Distance Measures

Denoted by d_{ij} (or s_{ij} respectively): is distance between any two datapoints i and j.

1. Mahalanobis Distance (discussed for Gaussian): $||\phi(\mathbf{x}) - \mu||_2^2 \Rightarrow (\phi(\mathbf{x}^{(i)}) - \phi(\mathbf{x}^{(j)}))^T \Sigma^{-1} (\phi(\mathbf{x}^{(i)}) - \phi(\mathbf{x}^{(j)}))$. EM algorithm has this in some sense.
2. If $\phi(\mathbf{x})$ are numeric / ordinal (optionally normalized to $\|\phi(x_i) - \phi\left(\mathbf{x}^{(j)}\right)\|_p = 1$):

$$\|\phi(\mathbf{x})\|_p = \left(\sum_{l=1}^{m} (\phi_l(x_i) - \phi_l\left(\mathbf{x}^{(j)}\right))^p\right)^{1/p}$$

   1. $p = 1$: Manhattan distance
   2. $p = 2$: Euclidean distance
   3. $p > 2$: Minkowski distance

Distance Measures (binary features)

1. If $\phi(\mathbf{x})$ are binary, measures based on contingency matrix defined over any two features ϕ_i and ϕ_j.
$$M = \begin{bmatrix} \#(i=1, j=1) = p & \#(i=1, j=0) = q \\ \#(i=0, j=1) = r & \#(i=0, j=0) = s \end{bmatrix}$$
if $p+q+r+s = n$, some symmetric and asymmetric measures
    1. $d_{ij} = \frac{q+r}{n}$: symmetric
    2. $d_{ij} = \frac{q+r}{p+s}$: symmetric (odd's ratio)
    3. $d_{ij} = 1 - (p/n)$: asymmetric
    4. $d_{ij} = 1 - (s/n)$: asymmetric
    5. $d_{ij} = 1 - (\frac{p}{p+q+r})$: asymmetric
    (Jaccard distance: refer: `http://en.wikipedia.org/wiki/Jaccard_index`)

Distance Measures (non-binary categorical features)

1. If $\phi(\mathbf{x})$ are discrete then :
   - $d_{ij} = 1 - \frac{\#(\phi_k(i)=\phi_k(j))}{n}$: Symmetric measure
   - Expand ϕ to multiple binary features $\phi_1 \ldots \phi_k$, if the original ϕ, takes k values. Now we can have the various symmetric and asymmetric measures defined for binary features above.
2. If $\phi(\mathbf{x})$ is a combination of numeric/ordinal and discrete

$$tot\_d_{ij} = w_1 * d_{ij}^{discrete} + w_2 * d_{ij}^{num/ordinal} \quad \text{s.t. } w_1 + w_2 = 1, w_1, w_2 \in [0, 1]$$

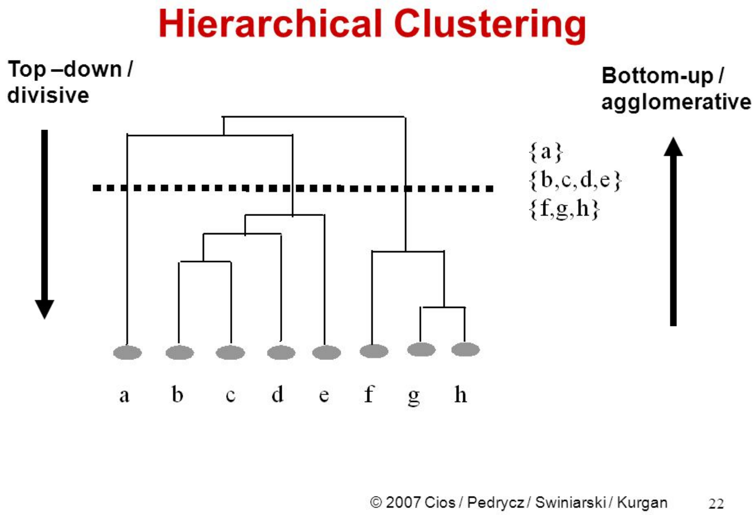

Figure: Bottom-up and Top-down Hierarchical Clustering

Bottom-up Hierarchical Clustering

1. Initially every point is a cluster of its own
2. Iteratively merge closes clusters (single-link, complete-link, average distance): Merge clusters that have the least mutual distance. For top-down: Which clusters to break.
3. When to stop merging clusters (closely linked to the distance measure). Stop when the distance between two clusters is $> \theta$ (some threshold). For top-down: When to stop splitting the clusters.

ISSUES:

1. Can't undo clustering decision.
2. Lack of flexibility in choice of clustering algorithm
3. Do not scale well.

ADVANTAGE: Easy to visualize. So a choosen *k* from hierarchical clustering can be used in k-means or any other clustering algorithm run from scratch.
Some of the algoritms studied here were *Birch clustering* and *Chameleon*.

Extra Slides: Derivation of MLE and MAP for GDA, Another Generative Distribution with MLE and MAP: Multinomial Distribution, Multinomial Naive Bayes, Frameworks for Multilabel Classification

Multinomial distribution

- Multinomial distribution is similar to the binomial distribution but for a variable that could assume one of t possible values $V_1, V_2 \dots V_t$
- *Eg:* In the case of the toss of dice, $t = 6$
- $\Pr(X = V_j) = \mu_j$
- Given n iid observations of a multinomial random variables, with m_i being the number of times $X = V_i$ was observed, the likelihood will be:

Multinomial distribution

- Multinomial distribution is similar to the binomial distribution but for a variable that could assume one of t possible values $V_1, V_2 \dots V_t$
- *Eg:* In the case of the toss of dice, $t = 6$
- $\Pr(X = V_j) = \mu_j$
- Given n iid observations of a multinomial random variables, with m_i being the number of times $X = V_i$ was observed, the likelihood will be:

$$L(n_1, \dots, n_t; \mu_1, \dots, \mu_t) = \frac{n!}{n_1! \cdots n_t!} \mu_1^{n_1} \cdots \mu_t^{n_t} \tag{5}$$

Finding the conjugate prior

Question: What will be conjugate priors for μ_j's, the parameters of Multinomial?

Dirichlet Prior for Multinomial

$$P\left(\mu_1, \ldots \mu_t | \alpha_1, \ldots \alpha_t\right) \propto \prod_{i=1}^{t} \mu_i^{\alpha_i - 1} \tag{6}$$

- Normalizing (to make the prior a density function):

Dirichlet Prior for Multinomial

$$P\left(\mu_1, \ldots \mu_t | \alpha_1, \ldots \alpha_t\right) \propto \prod_{i=1}^{t} \mu_i^{\alpha_i - 1} \tag{6}$$

- Normalizing (to make the prior a density function):

$$\int_{\mu_1} \ldots \int_{\mu_t} P\left(\mu_1, \ldots \mu_n | \alpha_1, \ldots \alpha_t\right) = 1$$

$$P\left(\mu_1, \ldots \mu_t | \alpha_1, \ldots \alpha_t\right) = \frac{\Gamma(\sum_{l=1}^{t} \alpha_l)}{\prod_{l=1}^{t} \Gamma(\alpha_i)} \prod_{l=1}^{t} \mu_l^{\alpha_l - 1} \tag{7}$$

  which, is Dir($\alpha_1 \ \ldots \ \alpha_t$) - the **Dirichlet Distribution**.
  Recall $\Gamma(n) = (n-1)!$ when $n \in \mathcal{N}$

- ... a generalization of Beta distribution, just as multinomial is generalization of Bernoulli distribution

Dirichlet as Generalization of *Beta*(α, β)

1. $Dir(\mu_1, \mu_2, \ldots, \mu_t; \alpha_1, \ldots, \alpha_t) = \frac{\mu_1^{\alpha_1 - 1} \ldots \mu_t^{\alpha_t - 1}}{B(\alpha_1, \ldots, \alpha_t)}$ is the Dirichlet conjugate prior for multinomial/categorical distributions
2. $\mathbf{E}_{Dir(\alpha_1, \ldots, \alpha_t)}[\mu_l] = \frac{\alpha_l}{\sum_{l=1}^{t} \alpha_l}$
3. $Dir(1, \ldots, 1)$ is the uniform distribution!

Posterior Probability for Multinomial

$$P\left(\mu_1, \ldots \mu_t | x_1, \ldots x_n\right) = \frac{P\left(x_1, \ldots x_n | \mu_1, \ldots \mu_t\right) P(\mu_1, \ldots \mu_t)}{P(x_1, \ldots x_n)}$$

$$P\left(\mu_1, \ldots \mu_t | x_1, \ldots x_n\right) = \frac{\Gamma(\sum_{j=1}^{t} \alpha_j + n)}{\prod_{j=1}^{t} \Gamma(\alpha_j + \sum_{k=1}^{n} X_{k,j})} \prod_{j=1}^{t} \mu_j^{(\alpha_j - 1 + \sum_{k=1}^{n} X_{k,j})} \tag{8}$$

Summary for Multinomial

- For multinomial, the mean at maximum likelihood is given by:

$$\hat{\mu}_l = \frac{\sum_{j=1}^{m} X_{j,l}}{n} \tag{9}$$

- Conjugate prior follows $\text{Dir}(\alpha_1 \ldots \alpha_n)$
- Posterior is $\text{Dir}(\ldots \alpha_l + \sum_{j=1}^{m} X_{j,l} \ldots)$
- The expectation of μ for $\text{Dir}(\alpha_1 \ldots \alpha_n)$ is given by:

Summary for Multinomial

- For multinomial, the mean at maximum likelihood is given by:

$$\hat{\mu}_l = \frac{\sum_{j=1}^{m} X_{j,l}}{n} \tag{9}$$

- Conjugate prior follows $\text{Dir}(\alpha_1 \ldots \alpha_n)$
- Posterior is $\text{Dir}(\ldots \alpha_l + \sum_{j=1}^{m} X_{j,l} \ldots)$
- The expectation of μ for $\text{Dir}(\alpha_1 \ldots \alpha_n)$ is given by:

$$E[\mu]_{Dir(\alpha_1 \ldots \alpha_n)} = \left[\frac{\alpha_1}{\sum \alpha_l} \ldots \frac{\alpha_1}{\sum \alpha_l} \right] \tag{10}$$

- The expectation of μ for $\text{Dir}(\ldots \alpha_l + \sum_{j=1}^{m} X_{j,l} \ldots)$ is given by:

Summary for Multinomial

- For multinomial, the mean at maximum likelihood is given by:

$$\hat{\mu}_l = \frac{\sum_{j=1}^{m} X_{j,l}}{n} \tag{9}$$

- Conjugate prior follows $\mathrm{Dir}(\alpha_1 \ldots \alpha_n)$
- Posterior is $\mathrm{Dir}(\ldots \alpha_l + \sum_{j=1}^{m} X_{j,l} \ldots)$
- The expectation of μ for $\mathrm{Dir}(\alpha_1 \ldots \alpha_n)$ is given by:

$$E[\mu]_{Dir(\alpha_1 \ldots \alpha_n)} = \left[\frac{\alpha_1}{\sum \alpha_l} \ldots \frac{\alpha_1}{\sum \alpha_l}\right] \tag{10}$$

- The expectation of μ for $\mathrm{Dir}(\ldots \alpha_l + \sum_{j=1}^{m} X_{j,l} \ldots)$ is given by:

$$E[\mu]_{Dir(\ldots \alpha_l + \sum_{k=1}^{n} X_{j,l} \ldots)} = \left[\frac{\alpha_1 + \sum_j X_{j,1}}{\sum \alpha_l + n} \ldots \frac{\alpha_l + \sum_j X_{j,l}}{\sum \alpha_l + n} \ldots\right] \tag{11}$$

(Multinomial) Naive Bayes

- $< \mathbf{x}^{(j)}, C_i >$: Tuple with example $\mathbf{x}^{(j)}$ belonging to class C_i. $\Pr(C_i)$ is prior probability of class C_i.
- $\phi_1\left(\mathbf{x}^{(j)}\right), \ldots, \phi_m\left(\mathbf{x}^{(j)}\right)$: The feature vector for $\mathbf{x}^{(j)}$
- $P(\phi_q(\mathbf{x})|C_i) \sim Mult(\mu^q_{1,i} \ldots \mu^q_{t_q,i})$; that is, each feature ϕ_q follows multinomial distribution Bayes
  1. $[V^1_1 \ldots V^1_{t_1}] \ldots [V^q_1 \ldots V^q_{t_q}] \ldots [V^m_1 \ldots V^m_{t_m}]$: Set of values that could be taken by each of $\phi_1, \phi_2 \ldots \phi_m$ respectively
  2. $[\mu^1_{1,i} \ldots \mu^1_{t_1,i}] \ldots [\mu^q_{1,i} \ldots \mu^q_{t_q,i}] \ldots [\mu^m_{1,i} \ldots \mu^m_{t_m,i}]$: Parameters for each of $\phi_1, \phi_2 \ldots \phi_m$ respectively for class C_i
- $P(\phi_1(\mathbf{x}) \ldots \phi_m(\mathbf{x})|C_i) = \prod_{q=1}^{m} P(\phi_q(\mathbf{x})|C_i)$: Feature are independent given the class

ML for Naive Bayes

ML Estimators: $\left[\hat{\mu}_{ML}, \hat{P}r_{ML}(C_i)\right]$. or more simply $\left[\hat{\mu}, \hat{P}r(C_i)\right]$

$$\hat{\mu}, \hat{P}r(C) = \underset{\mu, Pr(C_i)}{\text{argmax}} \prod_{k=1}^{n} Pr(c(X_k)) * \prod_{q=1}^{m} Pr(\phi_q(X_k)|c(X_k))$$

$$= \underset{\mu, Pr(C)}{\text{argmax}} \prod_{i=1}^{|C|} \left(Pr(C_i)\right)^{\#C_i} * \prod_{q=1}^{m} \prod_{j=1}^{t_q} \left(\mu_{j,i}^{q}\right)^{n_{j,i}^{q}}$$

where,

$$\#C_i = \text{ No. of times } c(X_k) = C_i \text{ across all } k\text{'s in the dataset}$$

$$n_{j,i}^{q} = \text{ No. of times } \phi_q(X_k) = V_j \text{ and } c(X_k) = C_i \text{ across all the } k\text{'s}$$

$$n_{j,i}^{q} = \sum_{k} \delta\left(\phi_q(X_k), V_j^q\right) \delta\left(c(Xk_i), C_i\right)$$

$$Pr(c(X_k) = \sum_{i=1}^{|C|} \delta\big(c(X_k), C_i\big) Pr(C_i)$$

$$Pr(\phi_q(X_k)|c(X_k)) = \sum_{j=1}^{t_q} \delta\big(\phi_q(X_k), V_j^q\big) * \mu_{j,c(X_k)}^q$$

$$Pr(c(X_k)) = \sum_{i=1}^{|C|} \delta\big(c(X_k), C_i\big) Pr(C_i)$$

$$Pr(\phi_q(X_k)|c(X_k)) = \sum_{j=1}^{t_q} \delta\big(\phi_q(X_k), V_j^q\big) * \mu_{j,c(X_k)}^q$$

So, the final log-likelihood objective function is:

$$\boxed{\operatorname*{argmax}_{\mu, Pr(c)} \left[\sum_{i=1}^{|c|} (\# C_i) \log Pr(C_i) + \sum_{j=1}^{m} \sum_{j=1}^{t_q} n_{j,i}^q \log(\mu_{j,i}^q) \right]} \tag{12}$$

such that $\sum_{i=1}^{|c|} Pr(C_i) = 1$, $\sum_{j=1}^{t_q} \mu_{j,i}^q = 1 \quad \forall q, i$, $Pr(C_i) \in [0,1] \quad \forall i$ and $\mu_{j,i}^q \in [0,1] \quad \forall q, i, j$

Solving Naive Bayes through KKT Conditions

Intuitively, working out the KKT conditions on the above objective function, we get the Maximum Likelihood Naive Bayes estimators as follows

$$\hat{\mu}_{j,i}^{q} = \frac{n_{j,i}^{q}}{\sum_{j'=1}^{n} n_{j',i}^{q}}$$

$$\widehat{Pr}_{c_i} = \frac{\#C_i}{\sum_{i'} \#C_{i'}}$$

Tutorial 10

Can you now do Bayesian Inference for Naive Bayes using the Dirichlet Conjugate Prior for each $\phi_q(\mathbf{x})$?

Derivation of MAP and Maximum Likelihood Estimates for Multivariate Gaussian: Recapped from `https://www.cse.iitb.ac.in/~cs725/notes/lecture-slides/lecture-06-unannotated.pdf`

Likelihood estimates for each class C_i

Let $\mathcal{D}_i \subseteq \mathcal{D}$ the subset of data points that belong to class C_i. Let $\mathcal{D}_i = \mathbf{x}_1^i \ldots \mathbf{x}_{m_i}^i$

- $LL(\mathbf{x}_1^i \ldots \mathbf{x}_{m_i}^i | \mu_i, \Sigma_i) = -\frac{m}{2} ln(2\pi) - \frac{m}{2} ln[|\Sigma_i|] - \frac{1}{2} \sum_{j=1}^{m_i} ((\phi(\mathbf{x}_j^i) - \mu_i)^T \Sigma_i^{-1} (\phi(\mathbf{x}_j^i) - \mu_i))$.
- Setting $\nabla_{\mu_i} LL = 0$, and $\nabla_{\Sigma_i} LL = 0$ for each i individually , we get

Likelihood estimates for each class C_i

Let $\mathcal{D}_i \subseteq \mathcal{D}$ the subset of data points that belong to class C_i. Let $\mathcal{D}_i = \mathbf{x}_1^i ... \mathbf{x}_{m_i}^i$

- $LL(\mathbf{x}_1^i ... \mathbf{x}_{m_i}^i | \mu_i, \Sigma_i) = -\frac{m}{2} ln(2\pi) - \frac{m}{2} ln[|\Sigma_i|] - \frac{1}{2} \sum_{j=1}^{m_i} ((\phi(\mathbf{x}_j^i) - \mu_i)^T \Sigma_i^{-1} (\phi(\mathbf{x}_j^i) - \mu_i))$.
- Setting $\nabla_{\mu_i} LL = 0$, and $\nabla_{\Sigma_i} LL = 0$ for each i individually , we get
  1. $\nabla_{\mu_i} LL = \left[-\frac{1}{2} \sum_{j=1}^{m_i} 2(\phi(\mathbf{x}_j^i) - \mu_i) \right] \Sigma_i^{-1} = 0$
  2. Since Σ_i is invertible, $\sum_{j=1}^{m_i} (\phi(\mathbf{x}_j^i) - \mu_i) = 0$ ie, $\hat{\mu}_i = \frac{1}{m_i} \sum_{j=1}^{m_i} \phi(\mathbf{x}_j^i)$
  3. $\hat{\Sigma}_i = \frac{1}{m_i} \sum_{j=1}^{m_i} (\phi(\mathbf{x}_j^i) - \hat{\mu}_i)(\phi(\mathbf{x}_j^i) - \hat{\mu}_i)^T$

Estimates based on all n $\left(= \sum_{i=1}^{K} m_i\right)$ instances

- $\Pr\left((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\right) = \Pr\left(\mathbf{x}_1 \ldots \mathbf{x}_n \mid y_1 \ldots y_n\right) \Pr\left(y_1 \ldots y_n\right)$
  $= \prod_{i=1}^{K} \Pr\left(\mathbf{x}_1^i \ldots \mathbf{x}_{m_i}^i \mid \mu_i, \Sigma_i\right) \Pr(C_i)^{m_i} \Rightarrow$
- $LL\left((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\right) = \sum_{i=1}^{K} LL(\mathbf{x}_1^i \ldots \mathbf{x}_{m_i}^i \mid \mu_i, \Sigma_i) + m_i \log \Pr(C_i)$
  $= \left(\sum_{i=1}^{K} -\frac{m}{2} \ln(2\pi|\Sigma_i|) + \frac{1}{2} \sum_{j=1}^{m_i} ((\phi(\mathbf{x}_j^i) - \mu_i)^T \Sigma_i^{-1} (\phi(\mathbf{x}_j^i) - \mu_i))\right) + \sum_{i=1}^{K} m_i \log \Pr(C_i)$

Estimates based on all $n \left(= \sum_{i=1}^{K} m_i\right)$ instances

- $LL\left((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\right)$
$= \left(\sum_{i=1}^{K} -\frac{m}{2}\ln(2\pi|\Sigma_i|) - \frac{1}{2}\sum_{j=1}^{m_i}((\phi(\mathbf{x}^i_j) - \mu_i)^T\Sigma_i^{-1}(\phi(\mathbf{x}^i_j) - \mu_i))\right) + \sum_{i=1}^{K} m_i \log \Pr(C_i)$
- Like before, setting $\nabla_{\mu_i} LL = 0$, and $\nabla_{\Sigma_i} LL = 0$: $\hat{\mu}_i = \frac{1}{m_i}\sum_{j=1}^{m_i}\phi(\mathbf{x}^i_j)$ and
$\hat{\Sigma}_i = \frac{1}{m_i}\sum_{j=1}^{m_i}(\phi(\mathbf{x}^i_j) - \hat{\mu}_i)(\phi(\mathbf{x}^i_j) - \hat{\mu}_i)^T$
- Also setting $\nabla_{\Pr(C_i)} LL = 0$, $\widehat{\Pr(C_i)} = \frac{m_i}{\sum_{j=1}^{K} n_j}$

Conjugate Prior & MAP for Univariate Gaussian

RECAP:

- $P(\mathbf{x}) \sim \mathcal{N}(\mu, \sigma^2)$
- The conjugate prior for mean of univariate gaussian distribution in the case that σ^2 is known is
  $P(\mu) = \mathcal{N}(\mu_0, \sigma_0^2)$
- $P(\mu | x_1 ... x_n) = \mathcal{N}(\mu_n, \sigma_n^2)$
- $\mu_n = \left(\frac{\sigma^2}{n\sigma_0^2 + \sigma^2} \mu_0 \right) + \left(\frac{n\sigma_0^2}{n\sigma_0^2 + \sigma^2} \hat{\mu}_{mle} \right)$
- $\frac{1}{\sigma_n^2} = \frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}$

Conjugate Prior & MAP for Multivariate Gaussian

- Rearranging terms for $\mu \sim \mathcal{N}(\mu_0, {\sigma^2}_0)$ and $x \sim \mathcal{N}(\mu, \sigma^2)$

$$\frac{1}{\sigma_n^2} = \frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}$$
$$\frac{\mu_n}{\sigma_n^2} = \frac{n}{\sigma^2}\hat{\mu}_{mle} + \mu_0$$

  such that $\Pr(\mu|D) \sim \mathcal{N}(\mu_n, {\sigma_n}^2)$. Here n/σ^2 is due to noise in observation while $1/\sigma_0^2$ is due to uncertainity in μ

- Extending to Bayesian setting[11] for multivariate case with fixed Σ

$$\phi(\mathbf{x}) \sim \mathcal{N}(\mu, \Sigma),\ \mu \sim \mathcal{N}(\mu_0, \Sigma_0)\ \Rightarrow \Pr(\mu|D) = \mathcal{N}(\mu_n, \Sigma_n)$$

$$\Sigma_n^{-1} = \Sigma_0^{-1} + n\Sigma^{-1}$$
$$\Sigma_n^{-1}\mu_n = n\Sigma^{-1}\hat{\mu}_{mle} + \Sigma_0^{-1}\mu$$

  MAP estimates μ_n and Σ_n obtained by solving above linear system.

[11] https://en.wikipedia.org/wiki/Multivariate_normal_distribution#Bayesian_inference

Extensions

1. Recall assumption: $\phi(\mathbf{x})$ is generated using **exactly one** $\mathcal{N}(\mu_i, \Sigma_i)$
2. What if this assumption were violated?

Extensions

1. Recall assumption: $\phi(\mathbf{x})$ is generated using **exactly one** $\mathcal{N}(\mu_i, \Sigma_i)$
2. What if this assumption were violated?
   - **Supervised Multi-labeled**: What if an example $\phi(\mathbf{x})$ is **known to** belong to multiple classes (Gaussians)?

$$P(\phi(\mathbf{x})|C_p) = \mathcal{N}(\mu_p, \Sigma_p)$$

$$P(\phi(\mathbf{x})|C_q) = \mathcal{N}(\mu_q, \Sigma_q)$$

Extensions

1. Recall assumption: $\phi(\mathbf{x})$ is generated using **exactly one** $\mathcal{N}(\mu_i, \Sigma_i)$
2. What if this assumption were violated?
   - **Supervised Multi-labeled**: What if an example $\phi(\mathbf{x})$ is **known to** belong to multiple classes (Gaussians)?

$$P(\phi(\mathbf{x})|C_p) = \mathcal{N}(\mu_p, \Sigma_p)$$

$$P(\phi(\mathbf{x})|C_q) = \mathcal{N}(\mu_q, \Sigma_q)$$

   - **Unsupervised Mixture (of Gaussians)**:

$$\Pr(\phi(\mathbf{x})) = \sum_{i=1}^{K} \Pr(\phi(\mathbf{x}), C = z_i) = \sum_{i=1}^{K} \Pr(C = z_i)\mathcal{N}(\mu_i, \Sigma_i)$$

Supervised Multi-labeled

Building a K-class discriminant by combining a number of two-class discriminants

- one-versus-the-rest: In this approach, $K-1$ classifiers are constructed, each of which separates the points in a particular class C_k from points not in that classes
- one-versus-one: In this method, $\binom{K}{2}$ binary discriminant functions are introduced, one for every possible pair of classes.

Can you think of problems with each of the above?

Multi-labeling and Nil-labeling

Attempting to construct a K class discriminant from a set of two class discriminants can lead to multi-labeled and nil-labeled regions. Multilabeled regions marked with '?'.

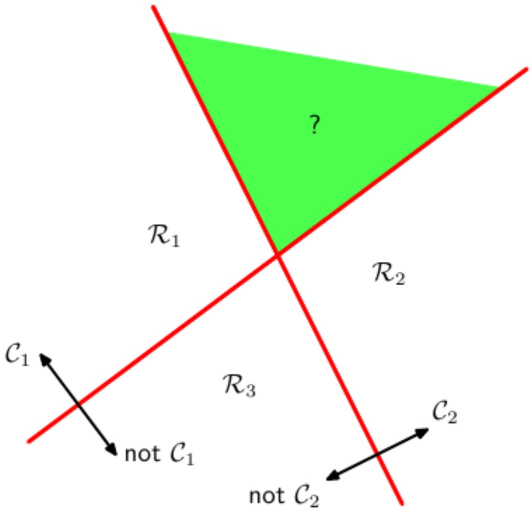

Figure: Illustrates multi-labeling and nil-labeling ($\mathcal{R}_3$ has no label) in one-versus-rest case

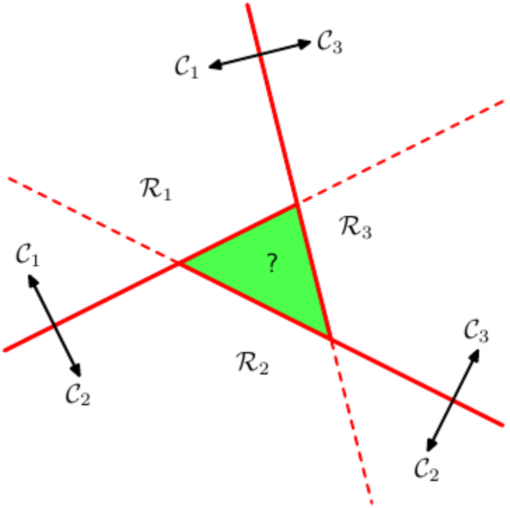

Figure: Illustrates the multi-labeling in one-versus-one case, but at the cost of complexity

OPTIONAL: Unbiased Estimators

- Estimator $e(\theta)$ is called an unbiased estimator of θ if $E[e(\theta)] = \theta$
- If $e_i(\theta), e_2(\theta), ..., e_k(\theta)$ are unbiased estimators and $\sum_{i=1}^{K} \lambda_i = 1$ then $\sum_{i=1}^{K} \lambda_i e_i(\theta)$ is also unbiased estimator
- $E(\hat{\Sigma}_i) = \frac{n_i - 1}{n_i} \Sigma_i \Rightarrow \hat{\Sigma}_i$ is a biased estimator.
- An unbiased estimator for Σ_i is therefore $\frac{1}{n_i - 1} \sum_{j=1}^{n_i} (\mathbf{x}_j^i - \hat{\mu}_i)(\mathbf{x}_j^i - \hat{\mu}_i)^T$

OPTIONAL: Sufficient statistic

- s is a sufficient statistic for θ if $\Pr(D|s,\theta)$ is independent of θ
  $\Leftrightarrow$ iff $\Pr(D|\theta)$ can be written as $\Pr(D|\theta) = g(s,\theta)h(D)$.

OPTIONAL: Sufficient statistic

- s is a sufficient statistic for θ if $\Pr(D|s,\theta)$ is independent of θ
  $\Leftrightarrow$ iff $\Pr(D|\theta)$ can be written as $\Pr(D|\theta) = g(s,\theta)h(D)$.
- For Gaussian, $\hat{\mu}_i = \dfrac{1}{m}\sum_{j=1}^{n_i}\phi(x_i)$ is a sufficient statistic for $\theta = \mu_i$ because:
  $\Pr(D|\mu_i) = g(\hat{\mu}_i, \mu_i)h(D)$, where

OPTIONAL: Sufficient statistic

- s is a sufficient statistic for θ if $\Pr(D|s,\theta)$ is independent of θ
  $\Leftrightarrow$ iff $\Pr(D|\theta)$ can be written as $\Pr(D|\theta) = g(s,\theta)h(D)$.
- For Gaussian, $\hat{\mu}_i = \frac{1}{m}\sum_{j=1}^{n_i}\phi(x_i)$ is a sufficient statistic for $\theta = \mu_i$ because:
  $\Pr(D|\mu_i) = g(\hat{\mu}_i, \mu_i)h(D)$, where

$$\Pr(D|\mu_i) = \prod_{j=1}^{n_i}\frac{1}{(2\pi)^{\frac{m}{2}}|\Sigma_i|^{\frac{1}{2}}} exp\left(\frac{-(\phi(x_j^i)-\mu_i)^T\Sigma_i^{-1}(\phi(x_j^i)-\mu_i)}{2}\right)$$

$$g(\hat{\mu}_{mle},\mu_i) = \exp\left(-\frac{n_i}{2}\mu_i^T\Sigma_i^{-1}\mu_i + \mu_i^T\Sigma_i^{-1}\left(n_i\hat{\mu}_i\right)\right)$$

$$h(x_1^i, x_2^i \ldots x_{n_i}^i) = \frac{1}{2\pi^{nm/2}|\Sigma_i|^{n_i/2}}\exp\left(-1/2\sum_{j=1}^{n_i}\phi^T(x_j^i)\Sigma_i^{-1}\phi(x_i^i)\right)$$