Maximum Likelihood in Machine Learning

MATH 290J, UCLA
©Michael Lindstrom, 2020
Many machine learning algorithms require parameter estimation. In many cases this estimation is done using the principle of **maximum likelihood** whereby we seek parameters so as to maximize the probability the observed data occurred *given* the model with those prescribed parameter values.

Examples of where maximum likelihood comes into play includes, but is not limited to:

- linear and nonlinear regression
- binary classification with logistic regression
- feed forward neural networks to classify or fit data
- clustering via mixture of Gaussians (and kmeans to an extent)
Conditional Probability and Bayes’ Theorem

The **conditional probability** of event $A$ given that event $B$ happened is defined by

$$\Pr(A|B) = \frac{\Pr(A \land B)}{\Pr(B)}.$$ 

From this very definition, we uncover **Bayes’ Theorem**:

$$\Pr(A|B) = \frac{\Pr(A \land B)}{\Pr(B)} = \frac{\Pr(A \land B) \Pr(A)}{\Pr(A) \Pr(B)} = \frac{\Pr(B|A) \Pr(A)}{\Pr(B)}.$$
For the sake of introducing maximum likelihood, we consider fitting data to a model describing how the data are generated. We denote:

- $D$: the data/observations collected
- $M(\theta)$: the model $M$ chosen parameterized by parameters $\theta$

For example, if we believe values are chosen from the a normal distribution $\mathcal{N}(\mu = 3, \sigma^2 = 22)$ then $\theta = (\mu, \sigma^2)$ and $M$ is a normal distribution.
**Terminology**

We define the following terms:

**posterior** (probability): \( \Pr(M(\theta) | D) \), i.e., the probability \( M(\theta) \) is correct given the observed data.

**likelihood**: \( \Pr(D | M(\theta)) \), i.e., the probability the data are observed given the model and parameters are true.

**prior** (probability): \( \Pr(M(\theta)) \), i.e., the probability mass/density for \( M(\theta) \).

**Remark**: often expressions like \( \Pr(D | M(\theta)) \) are not probabilities! They could be probability densities, too. That doesn’t stop the general community from this sort of notation.
Problems

\textbf{posterior}: \Pr(M(\theta)|D)

\textbf{likelihood}: \Pr(D|M(\theta))

\textbf{prior}: \Pr(M(\theta))

Most would generally agree that the “best model and parameters” would occur when the \textbf{posterior} is maximal. The trouble is that we cannot directly calculate it!

But by Bayes’ Theorem we can write that

\[ \Pr(M(\theta)|D) = \frac{\Pr(D|M(\theta)) \Pr(M(\theta))}{\Pr(D)} \]
If we really wanted, we can express $\Pr(D)$ in terms of (many) likelihoods as

$$\Pr(D) = \int_M \Pr(D|m) d\mu(m)$$

where $m$ ranges over $M$, all possible $M(\theta)$, and $\mu$ is a measure on $M$.

This isn’t really necessary as in trying to maximize $\Pr(M(\theta)|D)$, it is only a normalization constant.
Frequentist vs Bayesian Perspective

So how to we maximize

\[ \text{Pr}(M(\theta)|D) \propto \text{Pr}(D|M(\theta)) \text{Pr}(M(\theta)) \]?

**Frequentist**: a **frequentist** would say, “knowing the prior does *not* make sense! How can we possibly know something about the probability density/mass of all possible models in existence with their associated sets of parameters? Let’s give up on the prior and focus on maximizing the likelihood!”

**Bayesian**: a **bayesian** person would say, “let’s make an assumption on the prior and then try to maximize the posterior.”
Least Squares

The classical **least squares** algorithm is the frequentist approach of estimating parameters. Let’s derive this famous result.

Let us denote $Y \in \mathbb{R}$ to be a random variable representing a measurement in an experiment. Given an input $x \in \mathbb{R}^n$, we assume

$$Y = f(x; \theta) + \epsilon$$

where

- $f$ is a model parameterized by $\theta$ and
- $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is a Gaussian random variable (experimental error/uncertainty).

We shall denote

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1}^N$$

to be experimental points with inputs $x^{(i)}$ and measured value of $Y$ given by $y^{(i)}$. 
Least Squares

Model with Gaussian error.
Least Squares

For ease of notation, denote $\epsilon^{(i)} = y^{(i)} - f(x^{(i)}; \theta)$. Each $\epsilon^{(i)}$ is a random variable with pdf (probability density function)

$$p(z) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{z^2}{2\sigma^2}\right).$$

Assuming they are iid (independent identically distributed), to maximize the likelihood we want to maximize

$$L = \Pr(D|M(\theta))$$

$$= \prod_{i=1}^{N} \Pr(\epsilon^{(i)} = y^{(i)} - f(x^{(i)}; \theta))$$

$$= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(y^{(i)} - f(x^{(i)}; \theta))^2}{2\sigma^2}\right).$$
Least Squares

Often one seeks to maximize the log likelihood or minimize the negative log likelihood. Thus we wish to minimize

\[-\mathcal{L} = - \log L\]

\[= \sum_{i=1}^{N} \left( \log(\sqrt{2\pi\sigma}) + \frac{(y^{(i)} - f(x^{(i)}; \theta))^2}{2\sigma^2} \right)\]

\[= \frac{N}{2} \log(2\pi) + N \log \sigma + \frac{1}{2\sigma^2} \sum_{i=1}^{N} (y^{(i)} - f(x^{(i)}; \theta))^2\]

The value of \(-\mathcal{L}\) is minimized when

- \(\theta\) minimizes \(\sum_{i=1}^{N} (y^{(i)} - f(x^{(i)}; \theta))^2\) and
- \(\sigma^2 = \frac{1}{N} \sum_{i=1}^{N} (y^{(i)} - f(x^{(i)}; \theta))^2\) (yes, technically \(\sigma\) is a model parameter, too).
Aside: Too Many Parameters?

It is certainly possible to overfit data using maximum likelihood. Imagine fitting a polynomial of degree $d$ through $N$ points in the plane. The least squares error could be zero once $d = N - 1$.

The **Aikake Information Criterion (AIC)** is a means to penalize models with too many parameters. When comparing models, one compares their AIC values

$$\text{AIC} = 2d - 2 \log L^*$$

where $d$ is the number of parameters in a model and $L^*$ is the maximum likelihood for that model. The model with the lower AIC is often preferred: higher $d$ is bad unless $\log L^*$ can increase enough to compensate.
When Maximum Likelihood Isn’t So Good

While maximum likelihood is often a good approach, in certain cases, it can lead to a heavily biased estimates for parameters, i.e., in expectation, the estimates are off. Here is a trivial example.

Suppose our model posits that $X \sim U([0, \alpha])$ is a random variable uniformly distributed on $[0, \alpha]$, i.e., the pdf is

$$p(x) = \begin{cases} 
\frac{1}{\alpha}, & 0 \leq x \leq \alpha \\
0, & \text{otherwise}. 
\end{cases}$$
We are given the set of sample points \( D = \{ x_1, x_2, \ldots, x_N \} \). Given the data, what estimate do we place on \( \alpha \)?

We adopt the **indicator function** notation. We write \( \mathbb{1}_{x_i \leq \alpha} \) to represent the value 1 if \( x_i \leq \alpha \) and 0 otherwise, etc.
When Maximum Likelihood Isn’t So Good

Assuming iid,

$$
\Pr(D|\alpha) = \prod_{i=1}^{N} \left( \frac{1}{\alpha} \mathbb{1}_{x_i \leq \alpha} \right)
$$

$$
= \frac{1}{\alpha^N} \mathbb{1}_{x_1 \leq \alpha, x_2 \leq \alpha, \ldots, x_N \leq \alpha}
$$

$$
= \frac{1}{\alpha^N} \mathbb{1}_{\max\{x_1, \ldots, x_N\} \leq \alpha}
$$

Since $\alpha^{-N}$ is monotonically decreasing in $\alpha$, it is maximal when $\alpha$ is as small as possible. But from the $\mathbb{1}_{\max\{x_1, \ldots, x_N\} \leq \alpha}$ term, $\alpha$ can be no smaller than $\max\{x_1, \ldots, x_N\}$ or else the likelihood is 0 whence the maximum likelihood $\alpha = \max\{x_1, \ldots, x_N\}$. 
When Maximum Likelihood Isn’t So Good

Is this estimate any good? Given $N$ iid points $X_i$ sampled from $U([0, \alpha])$, we can calculate $\mathbb{E}(Y = \max\{X_1, \ldots, X_N\})$.

The cdf (cumulative distribution function) for $Y$,

$$F(y) = \Pr(Y \leq y) = \prod_{i=1}^{N} \Pr(X_i \leq y) = \prod_{i=1}^{N} \left( \begin{cases} 0, & y < 0 \\ y/\alpha, & 0 \leq y \leq \alpha \\ 1, & y > \alpha \end{cases} \right)$$

yielding a pdf

$$f(y) = F'(y) = \frac{N}{\alpha^N} y^{N-1} 1_{0 \leq y \leq \alpha}.$$ Integrating, we calculate $\mathbb{E}(Y) = \int_{0}^{\alpha} yf(y)\,dy = \frac{N}{N+1} \alpha$.

So as $N$ increases, the estimate is better and better. But it tends to underestimate the true value.
Logistic Regression

Logistic regression is a supervised learning algorithm (we know some ground truths ahead of time and these are used to “train” the algorithm). In its basic form, it is used to classify a binary output: “cat” vs “not cat”, “cancerous” vs “benign”, etc.

As as a model, we denote \( x = (1, x_1, x_2, ..., x_n) \in \mathbb{R}^{n+1} \) to be a features vector (a 1 plus the values of \( n \) properties used to make a prediction plus). The 1 is useful later.

We assume there is a Bernoulli random variable \( Y \in \{0, 1\} \) to indicate a negative/positive result we wish to describe where

\[
Y \sim \text{Bernoulli}(p(x; \theta)),
\]

i.e., given an \( x \), we can say

\[
\Pr(Y = 1) = 1 - \Pr(Y = 0) = p(x; \theta).
\]

The parameters are represented by \( \theta \).
Logistic Regression

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>rent $</td>
<td></td>
</tr>
<tr>
<td># thefts</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>record 1</td>
<td>1</td>
</tr>
<tr>
<td>record 2</td>
<td>0</td>
</tr>
<tr>
<td>record N</td>
<td>0</td>
</tr>
<tr>
<td>1 1000</td>
<td>3</td>
</tr>
<tr>
<td>1 2300</td>
<td>3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1 1600</td>
<td>0</td>
</tr>
</tbody>
</table>

Idea of using data to make predictions on a binary outcome.

**Remark:** often data are normalized before being placed in a logistic regression fit. Thus, we may convert all values to their $z$-scores or divide all values by the $\| \cdot \|_\infty$ value.
Logistic Regression

In logistic regression, we choose

\[
\theta = \begin{pmatrix}
\theta_0 \\
\theta_1 \\
\vdots \\
\theta_n
\end{pmatrix} \in \mathbb{R}^{n+1}
\]

and then let

\[
p(x; \theta) = \sigma(x\theta) = \sigma(\theta_0 + x_1\theta_1 + \ldots + x_n\theta_n)
\]

where \(\sigma\) is the \textbf{logistic} or \textbf{sigmoid} function

\[
\sigma(z) = \frac{\exp(z)}{1 + \exp(z)}.
\]
Logistic Regression

Plot of sigmoid function.
Logistic Regression

Given a data matrix $X \in \mathbb{R}^{N \times (n+1)}$ storing $N$ records of features with corresponding ground truths stored in $y \in \{0, 1\}^N$, we assume each $y_i$ is the realization of a Bernoulli trial with $x = X_i$: the $i^{th}$ row of $X$.

Finding the optimal $\theta \in \mathbb{R}^{n+1}$ for logistic regression amounts to maximizing the likelihood:

$$L = \prod_{i=1}^{N} \Pr(Y = y_i | X_i, \theta)$$

nice trick

$$\approx \prod_{i=1}^{N} \Pr(Y = 0 | X_i, \theta)^{y_i=0} \Pr(Y = 1 | X_i, \theta)^{y_i=1}$$
Logistic Regression

The log likelihood is

\[ L = \sum_{i=1}^{N} \mathbb{1}_{y_i=0} \log(1 - \sigma(X_i; \theta)) + \mathbb{1}_{y_i=1} \log(\sigma(X_i; \theta)) \]

\[ = \sum_{i \text{ s.t. } y_i=0} \log(1 - \sigma(X_i; \theta)) + \sum_{i \text{ s.t. } y_i=1} \log(\sigma(X_i; \theta)) \]

**Note:** The this value is hurt a lot when the algorithm is really sure that \( Y_i = 1 \) (\( \sigma \approx 1 \)) but \( y_i = 0 \) (\( \log(1 - \sigma) \downarrow -\infty \)). The same story applies when the algorithm believes \( Y_i = 0 \) but \( y_i \) is in fact 1.
Logistic Regression

Remarks: Mathematically, \( \sigma \in (0, 1) \). But through numerical roundoff errors, this can become 0 or 1. This will screw up computations. So from a practical perspective, it can be useful to define:

\[
\sigma(z) = \begin{cases} 
\epsilon, & \text{if } \frac{\exp(z)}{1+\exp(z)} \leq \epsilon \\
1 - \epsilon, & \text{if } \frac{\exp(z)}{1+\exp(z)} \geq 1 - \epsilon \\
\frac{\exp(z)}{1+\exp(z)}, & \text{otherwise}
\end{cases}
\]

for some \( 0 < \epsilon \ll 1 \). Pick \( \epsilon = 10^{-12} \), say.

The trick with the indicator function is quite useful: it allows us to write simpler sums that are not directly using the values of the response variable \( y_i \).
Logistic Regression

Finding the likelihood maximizing $\theta$ can be done with a method such as gradient descent upon $-\log \mathcal{L}$.

If we wish to find

$$\hat{\theta} = \arg\min_{\theta} (-\log \mathcal{L}(\theta))$$

we pick an initial guess $\theta^{(0)}$. Then denote

$$G(\theta) = \nabla_{\theta}(-\log \mathcal{L}(\theta)) \in \mathbb{R}^{n+1}.$$ 

We recursively define

$$\theta^{(i+1)} = \theta^{(i)} - \alpha G(\theta^{(i)})$$

where $0 < \alpha$ is a learning rate. Usually $\alpha \ll 1$, maybe 0.01 or something.
Logistic Regression

For predictions, one can vary a tolerance threshold $0 < \tau < 1$ such that we predict $Y = 0$ when $p(x; \theta) < \tau$ and $Y = 1$ otherwise. The choice of $\tau = 0.5$ is intuitive but not always the right choice. Generally as $\tau$ varies, there is a tradeoff between true positives (model predicts a positive outcome and observations confirm that) and false positives (observation results in a negative outcome but the model predicts a positive outcome).

If $\tau \downarrow 0$ then the model always predicts a positive outcome: the true positive rate is 100% (but so is the false positive rate - not good).

If $\tau \uparrow 1$ then the model always predicts a negative outcome: the false positive rate is 0% (but so is the true positive rate - not good).
Logistic Regression

One concern with logistic regressions is if they have predictive power in an unbalanced dataset: 95% of cases are positive, say. In that case, always predicting positive, regardless of the inputs would yield an accuracy of 95%.

To evaluate predictive power (besides validating against more data), we begin by imagining a perfect logistic regression algorithm.

For a perfect regression, we should be able to sort the $N$ data points into

$x_1, \ldots, x_m$ where $y_1 = 0, \ldots, y_m = 0$ with $p(x_1; \theta) \leq \ldots \leq p(x_m; \theta) < \tau$ and $x_{m+1}, \ldots, x_N$ where $y_{m+1} = 1, \ldots, y_N = 1$ with $\tau \leq p(x_{m+1}; \theta) \leq \ldots \leq p(x_N; \theta)$. 
For different $\tau$, perfect (and useless) classifiers will change their predictions. Perfect classifiers can perfectly separate outcomes based on the $p$'s and useless classifiers mix everything up.
Logistic Regression

In general then, as $\tau$ ranges on $[0, 1]$, we should see an ROC (receiver operator characteristic) curve moving from $(1, 1)$ to $(0, 1)$ then to $(0, 0)$ in the true positive vs false positive space.

A random regression where no insights can be drawn would mix up the positive and negative cases tracing a curve from $(1, 1)$ to $(0, 0)$.

In general, the ROC curve is somewhere between the two for a predictive model. The AUC (area under the curve) will be bigger than 0.5.
Logistic Regression

ROC curves. The AUC is the area under the curve as $\tau$ varies from 0 to 1.
Let’s consider another problem, an **unsupervised learning** problem (ground truth is not known). We want to group observation points into clusters.

As a model, we assume there exist $k$ different groups and each observation belongs to one of these groups. *We never know what group an observation truly belongs to!*
Mixture of Gaussians

We imagine a datum $X_i$ being generated as follows:

- Choose

$$Z_i \sim \text{Multinomial}(p_1, \ldots, p_k)$$

to be a cluster index so $Z_i \in \{1, 2, \ldots, k\}$ with $\Pr(Z_i = j) = p_j$ for $1 \leq j \leq k$. We say $Z_i$ is a **latent variable** because we never know it.

- After assigning a cluster, $j$, say, the observation is given a value in $\mathbb{R}^n$ according to a **multivariable Gaussian**

$$X_i \sim \mathcal{N}(\mu_j, \Xi_j)$$

where $\mu_j \in \mathbb{R}^n$ is the mean for cluster $j$ and $\Xi_j \in \mathbb{R}^{n \times n}$ is the **covariance** of points within cluster $j$, i.e.,

$$\Xi = \mathbb{E}((X_j - \mu_j) \otimes (X_j - \mu_j)).$$
Mixture of Gaussians

Case of 2 clusters with different means and covariances. Observations do come from one of the clusters but the real clustering is unknown.
Mixture of Gaussians

**Notation:** for brevity (context should make it clear), we may write

\( \Xi \) to represent all of \( \Xi_1, \ldots, \Xi_k \);

\( Z \) to represent all of \( Z_1, \ldots, Z_N \),

\( x \) to represent all of \( x_1, \ldots, x_N \);

etc.
Mixture of Gaussians

There is a lot we don’t know: the $\mu$’s, $\Xi$’s, and $p$’s! Putting that aside for now, we can try to come up with a likelihood.

We shall denote

$$
\rho(u; \mu_i, \Xi_i) = \frac{1}{(2\pi)^{n/2}|\Xi_i|^{n/2}} \exp\left(-\frac{1}{2} \langle u - \mu_i, \Xi_i^{-1}(u - \mu_i) \rangle \right)
$$

to be the density of the multivariable Gaussian in cluster $i$.

For a single observation $X_i$ (and playing fast and loose with densities and probabilities):

$$
\Pr(X_i = x_i|\rho, \mu, \Xi) = \sum_{\ell=1}^{k} \Pr(X_i = x_i|\mu_\ell, \Xi_\ell) \Pr(Z_i = \ell)
$$

$$
= \sum_{\ell=1}^{k} p_\ell \rho(x_i; \mu_\ell, \Xi_\ell)
$$

Not so bad...
Mixture of Gaussians

Now we consider our entire dataset. We have $N$ realizations of these random variables $x_1, \ldots, x_N$. The likelihood, assuming each observation is iid is:

$$L = \prod_{i=1}^{N} \left( \sum_{\ell=1}^{k} p_{\ell} \rho(x_i; \mu_\ell, \Xi_\ell) \right).$$

And the log likelihood is

$$\mathcal{L} = \sum_{i=1}^{N} \log \left( \sum_{\ell=1}^{k} p_{\ell} \rho(x_i; \mu_\ell, \Xi_\ell) \right).$$

This is not much of an improvement. Maximizing this is difficult: we can’t maximize analytically here and gradients are difficult to compute.

**Remark:** one of the chief difficulties is having a log of a sum. The fact we don’t know the $Z_i$’s is a big challenge!
Mixture of Gaussians

Suppose we knew the $Z_i$’s... Then

$$\Pr(X_i = x_i \land Z_i = z_i | \rho, \mu, \Xi) = \Pr(X_i = x_i | \mu_{z_i}, \Xi_{z_i}) \Pr(Z_i = z_i)$$

$$= \rho_{z_i} \rho(x_i; \mu_{z_i}, \Xi_{z_i}),$$

not a sum anymore. So if we knew all the $Z_i$’s then the complete likelihood and complete log likelihoods are given by

$$L^* = \prod_{i=1}^{N} \rho_{z_i} \rho(x_i; \mu_{z_i}, \Xi_{z_i})$$

$$L^* = \prod_{i=1}^{N} \prod_{\ell=1}^{k} \left( \rho_{\ell} \rho(x_i; \mu_{z_\ell}, \Xi_{z_\ell}) \right)^{z_i = \ell} \implies$$

$$\mathcal{L}^* = \log L^* = \sum_{i=1}^{N} \sum_{\ell=1}^{k} \mathbb{1}_{z_i = \ell} \left( \log \rho_{\ell} + \log \rho(x_i; \mu_{\ell}, \Xi_{\ell}) \right)$$
Mixture of Gaussians

It can be proven that by *maximizing the expected value of the complete log likelihood with respect to posterior of the latent variables, we also maximize the true likelihood*. We want to maximize

$$\mathbb{E}_{Z|X}(\mathcal{L}^*).$$

$\mathbb{E}_{Z|X}$ means to compute an expectation conditioned on the observed data $X$. In particular:

$$\mathbb{E}_{Z|X}(\mathcal{L}^*) = \sum_{i=1}^{N} \sum_{\ell=1}^{k} \mathbb{E}_{Z|X}(\mathbb{1}_{Z_i = \ell} (\log p_\ell + \log \rho(x_i; \mu_\ell, \Xi_\ell)))$$

$$= \sum_{i=1}^{N} \sum_{\ell=1}^{k} \mathbb{E}_{Z|X}(\mathbb{1}_{Z_i = \ell}) (\log p_\ell + \log \rho(x_i; \mu_\ell, \Xi_\ell))$$

$$= \sum_{i=1}^{N} \sum_{\ell=1}^{k} \Pr(Z_i = \ell|X_i = x_i) (\log p_\ell + \log \rho(x_i; \mu_\ell, \Xi_\ell))$$
Mixture of Gaussians

From Bayes, we can write

\[
\Pr(Z_i = \ell | X_i = x_i) = \frac{\Pr(X_i = x | Z_i = \ell) \Pr(Z_i = \ell)}{\Pr(X_i = x)} = \frac{p_{\ell} \rho(x_i; \mu_{\ell}, \Xi_{\ell})}{\sum_{\ell=1}^k p_{\ell} \rho(x_i; \mu_{\ell}, \Xi_{\ell})} := \gamma_{i,\ell}
\]

This means for fixed parameters, we have

\[
\mathbb{E}_{Z|X}(L^*) = \sum_{i=1}^N \sum_{\ell=1}^k \gamma_{i,\ell} (\log p_{\ell} + \log \rho(x_i; \mu_{\ell}, \Xi_{\ell})).
\]
Mixture of Gaussians

If the $\gamma$’s were fixed, it wouldn’t be hard to maximize this. Since the $\log p_\ell$ and $\log \rho(x_i; \mu_{z_\ell}, \Xi_{z_\ell})$ terms are decoupled, we can maximize $p$ separately from $\mu$ and $\Xi$. To maximize over $p$ we wish to:

$$\text{maximize } F(p) = \sum_{i=1}^{N} \sum_{\ell=1}^{k} \gamma_{i,\ell} \log p_\ell$$

subject to $G(p) = \sum_{\ell=1}^{k} p_\ell - 1 = 0$, $\min p \geq 0$.

The Lagrange system is

$$\nabla F = \lambda \nabla G$$

$$G(p) = 0$$
Mixture of Gaussians

We can compute

$$\partial_{p_j} F = \sum_{i=1}^{N} \sum_{\ell=1}^{k} \gamma_{i,\ell} \frac{\delta_{\ell,j}}{p_{\ell}} = \sum_{i=1}^{N} \frac{\gamma_{i,\ell}}{p_j}$$

and

$$\partial_{p_j} G = 1.$$ 

Given that $$\sum_{i=1}^{N} \frac{\gamma_{i,\ell}}{p_j} = \lambda$$ for $$j = 1, \ldots, k$$, we get $$p_j = \lambda^{-1} \sum_{i=1}^{N} \gamma_{i,j}$$. And by the $$G$$ constraint,

$$\sum_{j=1}^{k} p_j = 1 = \lambda^{-1} \sum_{i=1}^{N} \sum_{j=1}^{k} \gamma_{i,j}$$

giving $$\lambda = N$$ so that

$$p_j = \frac{1}{N} \sum_{i=1}^{N} \gamma_{i,j}.$$
Mixture of Gaussians

Maximizing over $\mu$ and $\Xi$ can be done, too, but the work is more cumbersome...

To maximize $\mathbb{E}_{Z \mid X}(\mathcal{L}^*)$, we employ the famous EM (Expectation Maximization) algorithm:

- **E-step:** calculate $\gamma^{(t+1)}$ with fixed $p^{(t)}, \mu^{(t)}, \Xi^{(t)}$.
- **M-step:** with $\gamma^{(t+1)}$ fixed, let $(p^{(t+1)}, \mu^{(t+1)}, \Xi^{(t+1)}) = \arg \max_{p, \mu, \Xi} \mathbb{E}_{Z \mid X}(\mathcal{L}^*)$.
- Iterative between Expectation and Maximization until convergence.

The appropriate cluster for $x_i$ is $\ell = \arg \max_{\ell} \gamma_i, \ell$. 
In general $\mathbb{E}_{Z \mid X}(\mathcal{L}^*)$ gives a lower bound for $\mathcal{L}$. Iteratively, we can maximize $\mathcal{L}$. 

Mixture of Gaussians
Mixture of Gaussians

**Remarks:** The value \( k \) is a **hyperparameter** (we choose it ahead of time) although there are means of justifying what \( k \) should be.

The **EM** algorithm is very general and is often used in models where there are latent variables.

The **kmeans** algorithm can be thought of as a mixture of Gaussians where all of the covariance matrices \( \Xi \) are equal to \( \sigma^2 I \) where \( \sigma^2 \) is a variance and \( I \) is the identity: in other words, all the clusters are “spherically” symmetric with the same spread.
The basic kmeans algorithm clusters $N$ points $x_i \in \mathbb{R}^n, i = 1, ..., N$, into $k$ clusters. To implement:

1. Begin by randomly assigning each point to a cluster from 1 to $k$.
2. Calculate $\mu_1, ..., \mu_k$, the centre of mass of each cluster given the assignments.
3. For each point $x_i$, place it in the cluster index $\ell$ where $\ell = \arg \min_\ell \text{dist}(x_i, \mu_\ell)$.
4. Repeat (2) and (3) until convergence.

Step (3) can be thought of calculating $\gamma_{i,\ell}$ and “rounding” $\gamma_{i,\ell}$ up to 1 where it is maximal. Step (2) can be thought of as estimating the parameters $\mu_1, ..., \mu_k$ with the $\gamma$’s fixed.
Mixture of Gaussians

We can justify maximizing $\mathbb{E}_{Z \mid L}(L^*)$ to maximize $L$ as follows:

$$L(p, \mu, \Xi) = \log \Pr(X \mid p, \mu, \Xi) = \log \sum_{z \in \{1, 2, \ldots, k\}^N} \Pr(X \wedge (Z = z) \mid p, \mu, \Xi)$$

$$= \log \sum_{z \in \{1, 2, \ldots, k\}^N} \Pr(Z = z \mid X) \frac{\Pr(X \wedge (Z = z) \mid p, \mu, \Xi)}{\Pr(Z = z \mid X)}$$

Jensen

$$\geq \sum_{z \in \{1, 2, \ldots, k\}^N} \Pr(Z = z \mid X) \log \frac{\Pr(X \wedge (Z = z) \mid p, \mu, \Xi)}{\Pr(Z = z \mid X)}$$

$$= \sum_{z \in \{1, 2, \ldots, k\}^N} \left( \Pr(Z = z \mid X) \log \Pr(X \wedge (Z = z) \mid p, \mu, \Xi) \right)$$

$$- \Pr(Z = z \mid X) \log \Pr(Z = z \mid X))$$

$$= \mathbb{E}_{Z \mid L}(L^*(p, \mu, \Xi)) - \sum_{z \in \{1, 2, \ldots, k\}^N} \Pr(Z = z) \log \Pr(Z = z).$$