STA 314: Statistical Methods for Machine Learning I

Lecture 10 - Probabilistic Models

Chris J. Maddison

University of Toronto

Today

- Wrapping up inference and decision-making.
- Gaussian generative models.

MLE Recap

- Last time we discussed the maximum likelihood estimation view of machine learning:
- Specify a family of distributions $p(\mathbf{x}|\theta)$ parameterized by $\theta \in \Theta$.
- Observe a data set $\mathcal{D} = \{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\}$.
- Under an IID assumption, MLE corresponds to

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \Theta} \sum_{i=1}^{N} \log p(\mathbf{x}^{(i)}|\theta)$$

MLE issue: Data Sparsity

- Maximum likelihood has a pitfall: if you have too little data, it can overfit.
- E.g., what if you flip the coin twice and get H both times?

$$\theta_{\mathrm{ML}} = \frac{N_H}{N_H + N_T} = \frac{2}{2+0} = 1$$

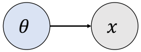
- Because it never observed T, it assigns this outcome probability 0. This problem is known as data sparsity.

Bayesiam Parameter Estimation

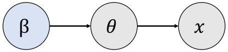
- Somehow we want to reflect our uncertainty in the true value of θ.
- Maybe the problem was that we summarized $\mathcal{D}$ in a single setting of the parameters $\hat{\boldsymbol{\theta}}_{\text{MLE}}$
- What if we summarized using a distribution? This will allow us to reflect that fact that we want to consider a variety of possible parameters weighted by some probability. This is the spirit behind Bayesian inference.

Bayesian Parameter Estimation

- In maximum likelihood, the observations are treated as random variables, but the parameters are not.



- The Bayesian approach treats the parameters as random variables as well. β is the set of parameters in the prior distribution of θ.



- To define a Bayesian model, we need to specify two distributions:
  - The prior distribution $p(\boldsymbol{\theta})$, which encodes our beliefs about the parameters *before* we observe the data
  - The likelihood $p(\mathcal{D} \mid \boldsymbol{\theta})$, same as in maximum likelihood

Bayesian Parameter Estimation

- The posterior distribution is the distribution that we will use to summarize $\mathcal{D}$.
- Using Bayes' Rule:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{p(\boldsymbol{\theta})p(\mathcal{D} \mid \boldsymbol{\theta})}{\int p(\boldsymbol{\theta}')p(\mathcal{D} \mid \boldsymbol{\theta}')\,\mathrm{d}\boldsymbol{\theta}'}.$$

- We rarely ever compute the denominator explicitly. In general, it is computationally intractable.

Bayesian Parameter Estimation

- Let's revisit the coin example. We already know the likelihood:

$$L(\theta) = p(\mathcal{D}|\theta) = \theta^{N_H}(1-\theta)^{N_T}$$

- It remains to specify the prior $p(\theta)$.
  - We can choose an uninformative prior, which assumes as little as possible. A reasonable choice is the uniform prior.
  - But our experience tells us 0.5 is more likely than 0.99. One particularly useful prior that lets us specify this is the beta distribution:

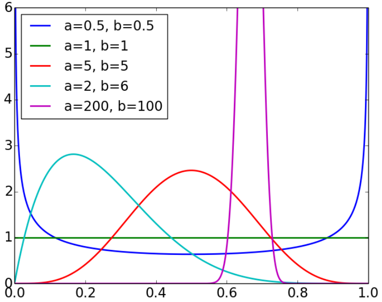
$$p(\theta; a, b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\,\theta^{a-1}(1-\theta)^{b-1}.$$

  - This notation for proportionality lets us ignore the normalization constant:

$$p(\theta; a, b) \propto \theta^{a-1}(1-\theta)^{b-1}.$$

Bayesian Parameter Estimation

- Beta distribution for various values of a, b:



- Some observations:
  - The expectation $\mathbb{E}[\theta] = a/(a + b)$ (easy to derive).
  - The distribution gets more peaked when a and b are large.
  - The uniform distribution is the special case where $a = b = 1$.
- The beta distribution is used for is as a prior for the Bernoulli distribution.

Bayesian Parameter Estimation

- Computing the posterior distribution:

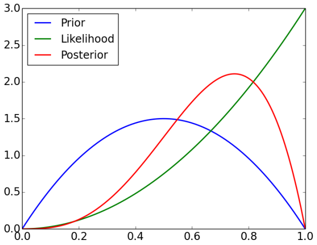
$$\begin{aligned} p(\boldsymbol{\theta} \,|\, \mathcal{D}) &\propto p(\boldsymbol{\theta}) p(\mathcal{D} \,|\, \boldsymbol{\theta}) \\ &\propto \left[\theta^{a-1}(1-\theta)^{b-1}\right]\left[\theta^{N_H}(1-\theta)^{N_T}\right] \\ &= \theta^{a-1+N_H}(1-\theta)^{b-1+N_T}. \end{aligned}$$

- This is just a beta distribution with parameters $N_H + a$ and $N_T + b$.
- The parameters a and b of the prior can be thought of as pseudo-counts.
  - The reason this works is that the prior and likelihood have the same functional form. This phenomenon is known as conjugacy (conjugate priors), and it's very useful.

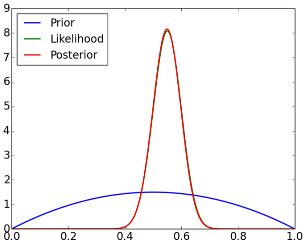
Bayesian Parameter Estimation

Bayesian inference for the coin flip example:

Small data setting
$N_H = 2, \ N_T = 0$



Large data setting
$N_H = 55, \ N_T = 45$



When you have enough observations, the data overwhelm the prior.

Bayesian Parameter Estimation

- What do we actually do with the posterior?
- The posterior predictive distribution is the distribution over future observables given the past observations. We compute this by marginalizing out the parameter(s):

$$p(\mathcal{D}' \mid \mathcal{D}) = \int p(\boldsymbol{\theta} \mid \mathcal{D}) p(\mathcal{D}' \mid \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}. \tag{1}$$

- For the coin flip example:

$$\begin{aligned}
\theta_{\mathrm{pred}} &= \Pr(\mathbf{x}' = H \mid \mathcal{D}) \\
&= \int p(\theta \mid \mathcal{D}) \Pr(\mathbf{x}' = H \mid \theta) \, \mathrm{d}\theta \\
&= \int \mathrm{Beta}(\theta; N_H + a, N_T + b) \cdot \theta \, \mathrm{d}\theta \\
&= \mathbb{E}_{\mathrm{Beta}(\theta; N_H + a, N_T + b)}[\theta] \\
&= \frac{N_H + a}{N_H + N_T + a + b},
\end{aligned} \tag{2}$$

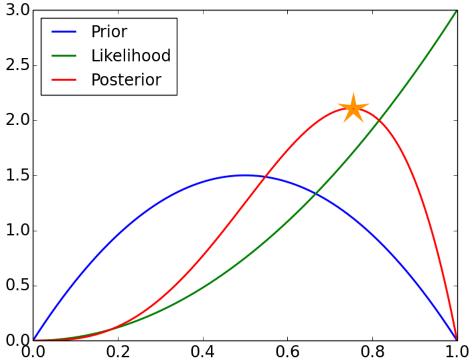
Bayesian Parameter Estimation

- Maybe we can summarize the posterior using a single value?
- One option is to use the posterior expectation of θ.
- For the coin flip example it coincides with the probability of heads:

$$\mathbb{E}[\theta \,|\, \mathcal{D}] = \frac{N_H + a}{N_H + N_T + a + b}$$

Maximum A-Posteriori Estimation

- Another option is Maximum a-posteriori (MAP) estimation: find the most likely parameter settings under the posterior to summarize the posterior.



Maximum A-Posteriori Estimation

- This converts the Bayesian parameter estimation problem into a maximization problem

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_{\mathrm{MAP}} &= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta} \,|\, \mathcal{D}) \\
&= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta}, \mathcal{D}) \\
&= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta})\, p(\mathcal{D} \,|\, \boldsymbol{\theta}) \\
&= \arg\max_{\boldsymbol{\theta}} \; \log p(\boldsymbol{\theta}) + \log p(\mathcal{D} \,|\, \boldsymbol{\theta})
\end{aligned}$$

- We already saw an example of this in the homework.

Maximum A-Posteriori Estimation

- Joint probability in the coin flip example:

$$\begin{aligned}\log p(\theta, \mathcal{D}) &= \log p(\theta) + \log p(\mathcal{D} \mid \theta) \\ &= \text{Const} + (a-1)\log\theta + (b-1)\log(1-\theta) + N_H \log\theta + N_T \log(1-\theta) \\ &= \text{Const} + (N_H + a - 1)\log\theta + (N_T + b - 1)\log(1-\theta)\end{aligned}$$

- Maximize by finding a critical point

$$0 = \frac{\mathrm{d}}{\mathrm{d}\theta} \log p(\theta, \mathcal{D}) = \frac{N_H + a - 1}{\theta} - \frac{N_T + b - 1}{1 - \theta}$$

- Solving for θ,

$$\hat{\theta}_{\mathrm{MAP}} = \frac{N_H + a - 1}{N_H + N_T + a + b - 2}$$

Maximum A-Posteriori Estimation

Comparison of estimates in the coin flip example:

	Formula	$N_H = 2, N_T = 0$	$N_H = 55, N_T = 45$
$\hat{\theta}_{\mathrm{ML}}$	$\frac{N_H}{N_H+N_T}$	1	$\frac{55}{100} = 0.55$
$\mathbb{E}[\theta \mid \mathcal{D}]$	$\frac{N_H+a}{N_H+N_T+a+b}$	$\frac{4}{6} \approx 0.67$	$\frac{57}{104} \approx 0.548$
$\hat{\theta}_{\mathrm{MAP}}$	$\frac{N_H+a-1}{N_H+N_T+a+b-2}$	$\frac{3}{4} = 0.75$	$\frac{56}{102} \approx 0.549$

$\hat{\theta}_{\mathrm{MAP}}$ assigns nonzero probabilities as long as $a, b > 1$.

Recap

- We took a probabilistic perspective on parameter estimation.
- We modeled a biased coin as a Bernoulli random variable with parameter θ, which we estimated using:
  - maximum likelihood estimation:
    $\hat{\theta}_{\mathrm{ML}} = \max_\theta p(\mathcal{D} \,|\, \theta)$
  - expected Bayesian posterior:
    $\mathbb{E}[\theta \,|\, \mathcal{D}]$ where $p(\theta \,|\, \mathcal{D}) \propto p(\boldsymbol{\theta})p(\mathcal{D} \,|\, \theta)$ by Bayes' Rule.
  - Maximum a-posteriori (MAP) estimation:
    $\hat{\theta}_{\mathrm{MAP}} = \arg\max_\theta \;\; p(\theta \,|\, \mathcal{D})$
- We also saw parameter estimation in context of a Naïve Bayes classifier.
- Today we will continuing developing the probabilistic perspective:
  - Gaussian Discriminant Analysis: Use Gaussian generative model of the data for classification
  - Gaussian Mixture Model: Gaussian generative model view of clustering

Motivation

- Generative models - model $p(\mathbf{x}|t=k)$
- Instead of trying to separate classes, try to model what each class "looks like".
- Recall that $p(\mathbf{x}|t=k)$ may be very complex

$$p(x_1, \cdots, x_d, y) = p(x_1|x_2, \cdots, x_d, y) \cdots p(x_{d-1}|x_d, y) p(x_d, y)$$

- Naive bayes used a conditional independence assumption. What else could we do? Choose a simple distribution.
- Today we will discuss fitting Gaussian distributions to our data.
- First, a review of our setting and MLE in Gaussians.

Multivariate Data

- Multiple measurements (sensors)
- d inputs/features/attributes
- N instances/observations/examples

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_d^{(N)} \end{bmatrix}$$

Multivariate Parameters

- Mean

$$\mathbb{E}[\mathbf{x}] = [\mu_1, \cdots, \mu_d]^T$$

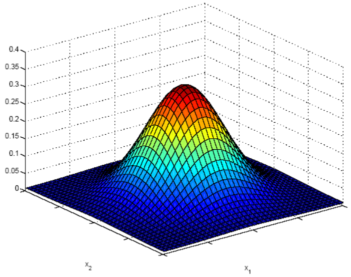
- Covariance

$$\Sigma = Cov(\mathbf{x}) = \mathbb{E}[(\mathbf{x} - \mu)^T(\mathbf{x} - \mu)] = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1d} \\ \sigma_{12} & \sigma_2^2 & \cdots & \sigma_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{d1} & \sigma_{d2} & \cdots & \sigma_d^2 \end{bmatrix}$$

Multivariate Gaussian Distribution

- $\mathbf{x} \sim \mathcal{N}(\mu, \Sigma)$, a Gaussian (or normal) distribution defined as

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right]$$



- Mahalanobis distance $(\mathbf{x}-\mu_k)^T\Sigma^{-1}(\mathbf{x}-\mu_k)$ measures the distance from $\mathbf{x}$ to μ in terms of Σ
- It normalizes for difference in variances and correlations

Gaussian Maximum Likelihood

- Suppose we want to model the distribution of highest and lowest temperatures in Toronto in March, and we've recorded the following observations

  (-2.5,-7.5) (-9.9,-14.9) (-12.1,-17.5) (-8.9,-13.9) (-6.0,-11.1)

- Assume they're drawn from a Gaussian distribution with mean $\boldsymbol{\mu}$, and covariance $\boldsymbol{\Sigma}$. We want to estimate these using data.
- Log-likelihood function:

$$\begin{aligned}\ell(\boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \log \prod_{i=1}^{N} \left[\frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x}^{(i)} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}^{(i)} - \boldsymbol{\mu}) \right\} \right] \\ &= \sum_{i=1}^{N} \log \left[\frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x}^{(i)} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}^{(i)} - \boldsymbol{\mu}) \right\} \right] \\ &= \sum_{i=1}^{N} \underbrace{- \log (2\pi)^{d/2}}_{\text{constant}} - \log |\boldsymbol{\Sigma}|^{1/2} - \frac{1}{2} (\mathbf{x}^{(i)} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}^{(i)} - \boldsymbol{\mu})\end{aligned}$$

Optional intuition building: why does $|\boldsymbol{\Sigma}|^{1/2}$ show up in the Gaussian density $p(\mathbf{x})$?

Hint: determinant is product of eigenvalues

Gaussian Maximum Likelihood

- Maximize the log-likelihood by setting the derivative to zero:

$$
\begin{aligned}
0 = \frac{\mathrm{d}\ell}{\mathrm{d}\boldsymbol{\mu}} &= -\sum_{i=1}^{N} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\mu}} \frac{1}{2} (\mathbf{x}^{(i)} - \boldsymbol{\mu})^{T} \boldsymbol{\Sigma}^{-1} (\mathbf{x}^{(i)} - \boldsymbol{\mu}) \\
&= -\sum_{i=1}^{N} \boldsymbol{\Sigma}^{-1} (\mathbf{x}^{(i)} - \boldsymbol{\mu}) = 0
\end{aligned}
$$

- Here we use the identity $\partial \mathbf{x}^{\top} \mathbf{A} \mathbf{x} / \partial \mathbf{x} = 2\mathbf{A}\mathbf{x}$ for symmetric $\mathbf{A}$.
- Solving we get $\hat{\boldsymbol{\mu}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}^{(i)}$. In general, “hat” means estimator
- This is just the sample mean of the observed values, or the empirical mean.

Gaussian Maximum Likelihood

- We can do a similar calculation for the covariance matrix Σ (we skip the details).
- Setting the *partial* derivatives to zero, just like before, we get:

$$
\begin{aligned}
0 = \frac{\partial \ell}{\partial \boldsymbol{\Sigma}} \implies \hat{\boldsymbol{\Sigma}} &= \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})^\top \\
&= \frac{1}{N}(\mathbf{X} - \mathbf{1}\boldsymbol{\mu}^\top)^\top(\mathbf{X} - \mathbf{1}\boldsymbol{\mu}^\top)
\end{aligned}
$$

  where $\mathbf{1}$ is an N-dimensional vector of 1s.
- This is called the empirical covariance and comes up quite often (e.g., PCA soon!)
- Derivation in multivariate case is tedious. No need to worry about it. But it is good practice to derive this in one dimension. See supplement (next slide).

Supplement: MLE for univariate Gaussian

$$
\begin{aligned}
0 = \frac{\partial \ell}{\partial \mu} &= -\frac{1}{\sigma^2} \sum_{i=1}^{N} \mathbf{x}^{(i)} - \mu \\
0 = \frac{\partial \ell}{\partial \sigma} &= \frac{\partial}{\partial \sigma} \left[\sum_{i=1}^{N} -\frac{1}{2} \log 2\pi - \log \sigma - \frac{1}{2\sigma^2} (\mathbf{x}^{(i)} - \mu)^2 \right] \\
&= \sum_{i=1}^{N} -\frac{1}{2} \frac{\partial}{\partial \sigma} \log 2\pi - \frac{\partial}{\partial \sigma} \log \sigma - \frac{\partial}{\partial \sigma} \frac{1}{2\sigma} (\mathbf{x}^{(i)} - \mu)^2 \\
&= \sum_{i=1}^{N} 0 - \frac{1}{\sigma} + \frac{1}{\sigma^3} (\mathbf{x}^{(i)} - \mu)^2 \\
&= -\frac{N}{\sigma} + \frac{1}{\sigma^3} \sum_{i=1}^{N} (\mathbf{x}^{(i)} - \mu)^2
\end{aligned}
$$

$$
\begin{aligned}
\hat{\mu}_{\mathrm{ML}} &= \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}^{(i)} \\
\hat{\sigma}_{\mathrm{ML}} &= \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}^{(i)} - \mu)^2}
\end{aligned}
$$

Bayes Classifier

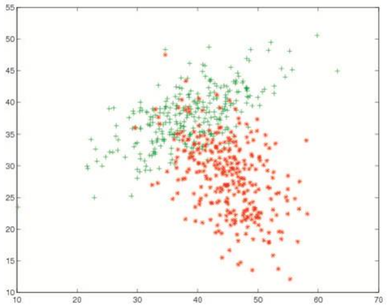
- Let's take a step back...
- Bayes Classifier

$$
\begin{aligned}
h(\mathbf{x}) &= \arg\max p(t = k|\mathbf{x}) = \arg\max \frac{p(\mathbf{x}|t = k)p(t = k)}{p(\mathbf{x})} \\
&= \arg\max p(\mathbf{x}|t = k)p(t = k)
\end{aligned}
$$

- Talked about Discrete **x**, what if **x** is continuous?

Classification: Diabetes Example

- Observation per patient: White blood cell count & glucose value.



- How can we model $p(x|t = k)$? Multivariate Gaussian

Gaussian Discriminant Analysis (Gaussian Bayes Classifier)

- Gaussian Discriminant Analysis in its general form assumes that $p(\mathbf{x}|t)$ is distributed according to a multivariate normal (Gaussian) distribution
- Multivariate Gaussian distribution:

$$p(\mathbf{x}|t = k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1}(\mathbf{x} - \boldsymbol{\mu}_k)\right]$$

  where $|\Sigma_k|$ denotes the determinant of the matrix, and d is dimension of $\mathbf{x}$
- Each class k has associated mean vector $\boldsymbol{\mu}_k$ and covariance matrix Σ_k
- Σ_k has $\mathcal{O}(d^2)$ parameters - could be hard to estimate (more on that later).

Learning

- Learn the parameters for each class using maximum likelihood
- Assume the prior is Bernoulli (we have two classes)

$$p(t|\phi) = \phi^t(1-\phi)^{1-t}.$$

- You can compute the MLE in closed form (good exercise!)

$$\hat{\phi} = \frac{1}{N}\sum_{n=1}^{N} 1[t^{(n)} = 1]$$

$$\hat{\boldsymbol{\mu}}_k = \frac{\sum_{n=1}^{N} 1[t^{(n)} = k] \cdot \mathbf{x}^{(n)}}{\sum_{n=1}^{N} 1[t^{(n)} = k]}$$

$$\hat{\mathbf{\Sigma}}_k = \frac{1}{\sum_{n=1}^{N} 1[t^{(n)} = k]} \sum_{n=1}^{N} 1[t^{(n)} = k](\mathbf{x}^{(n)} - \hat{\mu}_{t^{(n)}})(\mathbf{x}^{(n)} - \hat{\mu}_{t^{(n)}})^T$$

Gaussian Discriminant Analysis (Gaussian Bayes Classifier)

- GDA (GBC) decision boundary is based on class posterior.
- Make decisions by comparing class probabilities:

$$\begin{aligned}\log p(t_k|\mathbf{x}) &= \log p(\mathbf{x}|t_k) + \log p(t_k) - \log p(\mathbf{x}) \\ &= -\frac{d}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}_k^{-1}| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k) \\ &\quad + \log p(t_k) - \log p(\mathbf{x})\end{aligned}$$

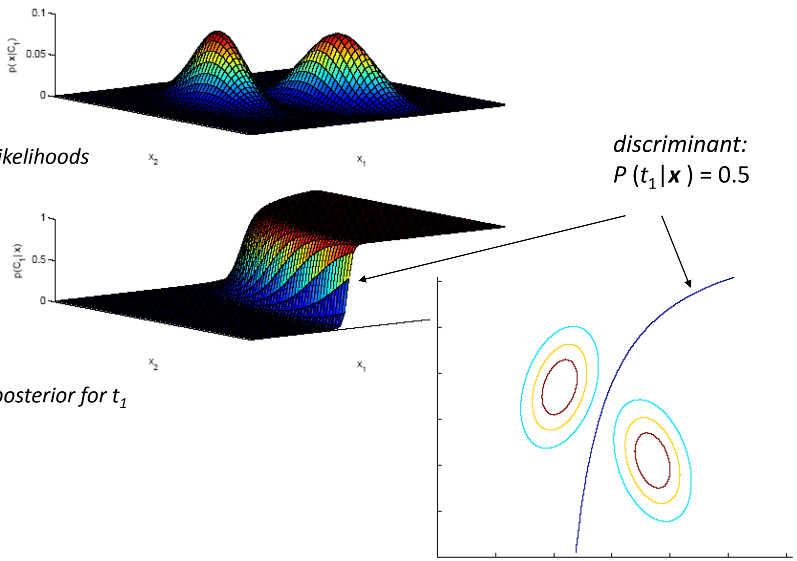
- Decision boundary ($\log p(t_k|\mathbf{x}) = \log p(t_l|\mathbf{x})$):

$$(\mathbf{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k) = (\mathbf{x}-\boldsymbol{\mu}_\ell)^T\boldsymbol{\Sigma}_\ell^{-1}(\mathbf{x}-\boldsymbol{\mu}_\ell) + C_{k,l}$$

$$\mathbf{x}^T\boldsymbol{\Sigma}_k^{-1}\mathbf{x} - 2\boldsymbol{\mu}_k^T\boldsymbol{\Sigma}_k^{-1}\mathbf{x} = \mathbf{x}^T\boldsymbol{\Sigma}_\ell^{-1}\mathbf{x} - 2\boldsymbol{\mu}_\ell^T\boldsymbol{\Sigma}_\ell^{-1}\mathbf{x} + C_{k,l}$$

- Quadratic relation in $\mathbf{x}$ $\Longrightarrow$ quadratic (conic) decision boundary
- So sometimes called "Quadratic Discriminant Analysis" (QDA)

Decision Boundary



Simplifying the Model

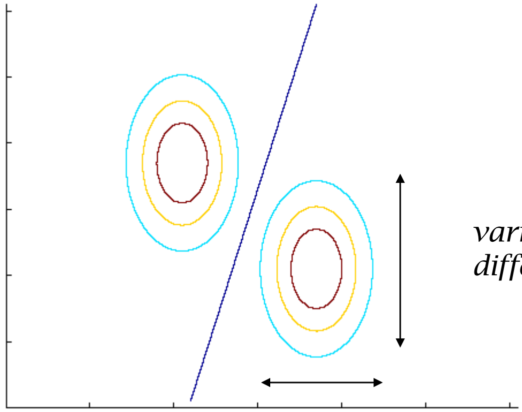
What if $\mathbf{x}$ is high-dimensional?

- For Gaussian Bayes Classifier, if input $\mathbf{x}$ is high-dimensional, then covariance matrix has many parameters $O(d^2)$
- Save some parameters by using a shared covariance for the classes, i.e. $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}_l$.
- MLE in this case:

$$\hat{\Sigma} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}^{(n)} - \mu_{t^{(n)}})(\mathbf{x}^{(n)} - \mu_{t^{(n)}})^T$$

- Linear decision boundary (at home: verify this mathematically!).
  - In Scikit-Learn this is called "Linear Discriminant Analysis" (LDA)

Decision Boundary: Shared Variances (between Classes)



variances may be different

Gaussian Discriminative Analysis vs Logistic Regression

- Binary classification: If you examine $p(t = 1|\mathbf{x})$ under GDA and assume $\Sigma_0 = \Sigma_1 = \Sigma$, you will find that it looks like this:

$$p(t|\mathbf{x}, \phi, \mu_0, \mu_1, \Sigma) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x})}$$

  where $\mathbf{w}$ is an appropriate function of $(\phi, \mu_0, \mu_1, \Sigma)$, $\phi = p(t = 1)$.
- GDA is similar to logistic regression (LR), parameter estimates are computed differently.
- When should we prefer GDA to LR, and vice versa?

Gaussian Discriminative Analysis vs Logistic Regression

- GDA is a generative model, LR is a discriminative model.
- GDA makes stronger modeling assumption: assumes class-conditional data is multivariate Gaussian.
- If this is true, GDA is asymptotically efficient (best model in limit of large N)
- But LR is more robust, less sensitive to incorrect modeling assumptions (what loss is it optimizing?)
- Many class-conditional distributions lead to logistic classifier.
- When these distributions are non-Gaussian (true almost always), LR usually beats GDA

A Generative View of Clustering

What if we do not observe the targets?

A Generative View of Clustering

- We covered hard and soft k-means algorithm for clustering.
- Today: statistical formulation of clustering → principled, justification for updates
- We need a sensible measure of what it means to cluster the data well
  - This makes it possible to judge different methods
  - It may help us decide on the number of clusters
- An obvious approach is to imagine that the data was produced by a generative model
  - Then we adjust the model parameters to maximize the probability that it would produce exactly the data we observed

Latent Variable Models

- To incorporate the idea of clusters model a joint distribution,

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

  between the data and an unobserved cluster id $z \in \{1, \ldots, K\}$.
- The "label" or cluster id z is not observed, so we call it a latent variable.
- Because z is unobserved, we cannot just maximize $\log p(\mathbf{x}, z)$. Instead, we must maximize just the likelihood of the data $\mathbf{x}$:

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

- This is an instance of a mixture model or more generally, a latent variable model.

Gaussian Mixture Model (GMM)

Most common mixture model: Gaussian mixture model (GMM)

- A GMM represents a **distribution** as

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

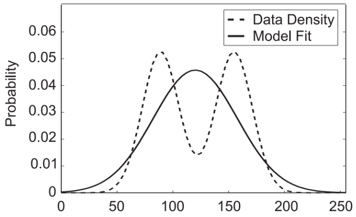
with π_k the mixing coefficients, where:

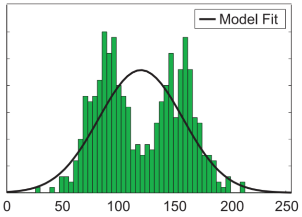
$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{and} \quad \pi_k \geq 0 \quad \forall k$$

- GMM is a density estimator
- In general mixture models are very powerful, but harder to optimize

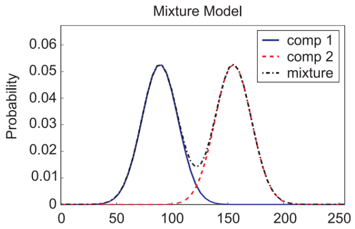
Visualizing a Mixture of Gaussians – 1D Gaussians

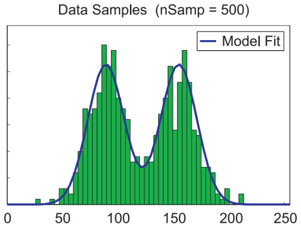
- If you fit a Gaussian to data:





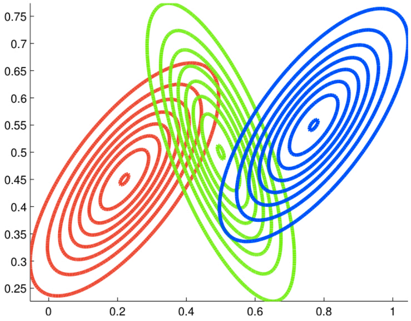
- Now, we are trying to fit a GMM (with $K = 2$ in this example):

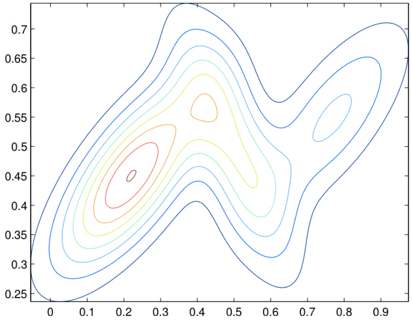


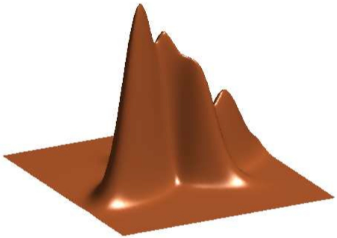


[Slide credit: K. Kutulakos]

Visualizing a Mixture of Gaussians – 2D Gaussians







Fitting GMMs: Maximum Likelihood

- Maximum likelihood maximizes

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

  w.r.t $\Theta = \{\pi_k, \mu_k, \Sigma_k\}$

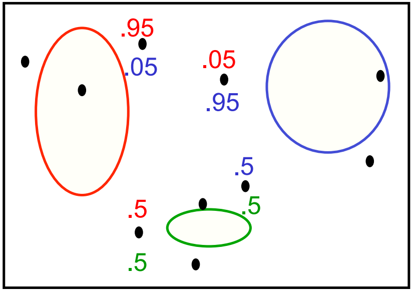
- Problems:
  - Singularities: Arbitrarily large likelihood when a Gaussian explains a single point
  - Identifiability: Solution is invariant to permutations
  - Non-convex
- How would you optimize this?
- Could try gradient descent, but don't forget to satisfy the constraints on π_k and Σ_k.

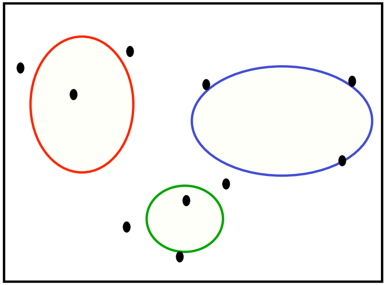
Expectation Maximization

- Typically a latent variable model is fit with the Expectation Maximization (EM) algorithm, or variants of it.
- The EM algorithm can be seen as a type of coordinate descent, just like K-means and our method for matrix completion.
- We won't go into details to justify the convergence of the algorithm, but I will show you the high-level algorithm for Gaussian mixture models and compare it to K-means.

Intuitively, How Can We Fit a Mixture of Gaussians?

1. E-step: Compute the posterior probability over z given our current model - i.e. how much do we think each Gaussian generates each datapoint.
2. M-step: Assuming that the data really was generated this way, change the parameters of each Gaussian to maximize the probability that it would generate the data it is currently responsible for.





Relation to k-Means

- The K-Means Algorithm:
  1. Assignment step: Assign each data point to the closest cluster
  2. Refitting step: Move each cluster center to the center of gravity of the data assigned to it
- The EM Algorithm:
  1. E-step: Compute the posterior probability over z given our current model
  2. M-step: Maximize the probability that it would generate the data it is currently responsible for.

EM Algorithm for GMM

- Initialize the means μ_k, covariances Σ_k and mixing coefficients π_k
- Iterate until convergence:
  - E-step: Evaluate the responsibilities given current parameters

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

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

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

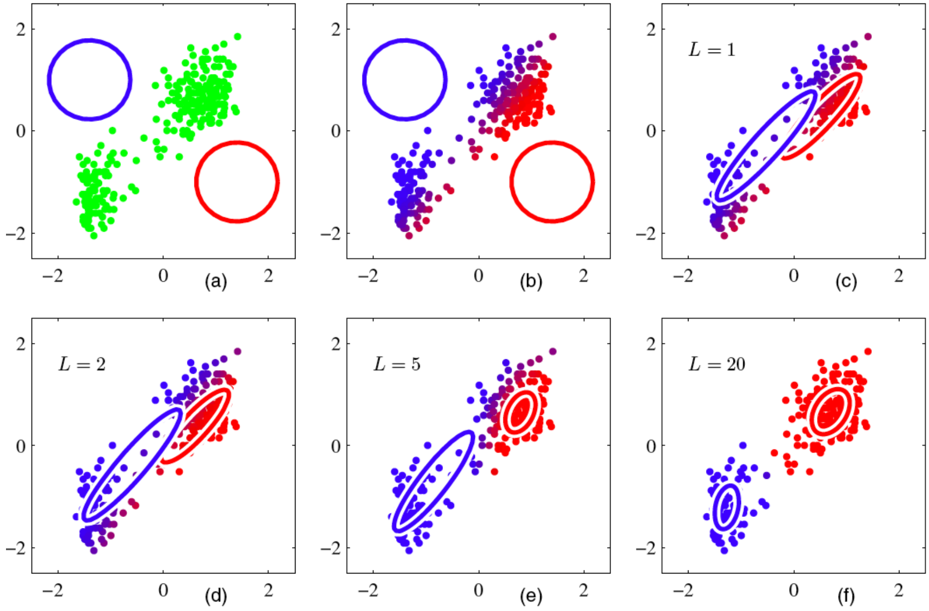
  - Evaluate log likelihood and check for convergence

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

EM Algorithm for GMM

- Can show that the EM algorithm monotonically improves the log-likelihood.
- Evaluate log likelihood and check for convergence

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



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

Mixture of Gaussians vs. K-means

- EM for mixtures of Gaussians is just like a soft version of K-means, with fixed priors and covariance
- Instead of hard assignments in the E-step, we do soft assignments based on the softmax of the squared Mahalanobis distance from each point to each cluster.
- Each center moved by weighted means of the data, with weights given by soft assignments
- In K-means, weights are 0 or 1.
- Confirm this at home!!!