CSCI-567: Machine Learning

Prof. Victor Adamchik

U of Southern California

July 27, 2020

Your model is only as good as your data.

Outline

General EM algorithm

EM is an algorithm to solve MLE with latent variables (not just GMM), i.e. find the maximizer of

$$P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \,;\boldsymbol{\theta})$$

Directly solving the objective is intractable. Instead we optimize the lower bound

$$P(\boldsymbol{\theta}) \geq F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right)$$

where

$$F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right) = \sum_{n=1}^{N}\sum_{k=1}^{K}\left(q_n(k)\ln p(\boldsymbol{x}_n, z_n = k \,;\boldsymbol{\theta}^{(t)}) - q_n(k)\ln q_n(k)\right)$$

General EM algorithm

Step 0 Initialize $\boldsymbol{\theta}^{(1)}$, $t = 1$

Step 1 (E-Step) **update the posterior of latent variables**

$$q_n^{(t)}(z_n = k) = p(z_n = k \mid \boldsymbol{x}_n \,;\boldsymbol{\theta}^{(t)})$$

and obtain **Expectation** of complete likelihood

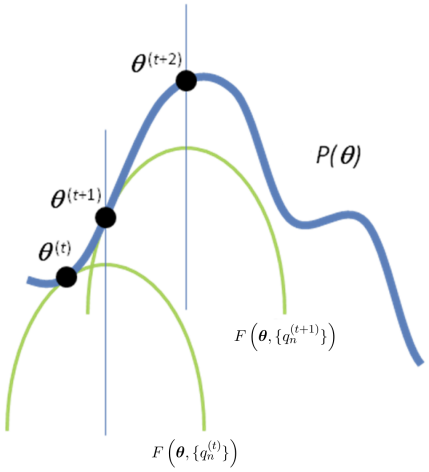
$$Q(\boldsymbol{\theta} \,;\boldsymbol{\theta}^{(t)}) = \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}}\left[\ln p(\boldsymbol{x}_n, z_n \,;\boldsymbol{\theta})\right]$$

Step 2 (M-Step) **update the model parameter** via **Maximization**

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \,;\boldsymbol{\theta}^{(t)})$$

Step 3 $t \leftarrow t + 1$ and return to Step 1 if not converged

Pictorial explanation



$P(\boldsymbol{\theta})$ is non-concave, but $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)})$ often is concave and easy to maximize.

$$\begin{aligned} P(\boldsymbol{\theta}^{(\mathrm{t}+1)}) &\geq F\left(\boldsymbol{\theta}^{(\mathrm{t}+1)}; \{q_n^{(t)}\}\right) \\ &\geq F\left(\boldsymbol{\theta}^{(\mathrm{t})}; \{q_n^{(t)}\}\right) \\ &= P(\boldsymbol{\theta}^{(\mathrm{t})}) \end{aligned}$$

So EM always increases the objective value and will converge to some local maximum (similar to K-means).

Apply EM to learn GMMs

E-Step:

$$q_n^{(t)}(z_n = k) = p\left(z_n = k \mid \boldsymbol{x}_n; \boldsymbol{\theta}^{(t)}\right)$$

This computes the "soft assignment" $\gamma_{nk} = q_n^{(t)}(z_n = k)$, i.e. conditional probability of $\boldsymbol{x}_n$ belonging to cluster k.

$$\begin{aligned} \gamma_{nk} = p\left(z_n = k \mid \boldsymbol{x}_n\right) &= \frac{p(z_n = k)p(\boldsymbol{x}_n \mid z_n = k)}{p(\boldsymbol{x}_n)} \\ &= \frac{p(z_n = k)p(\boldsymbol{x}_n \mid z_n = k)}{\sum_j^K p(z_n = j)p(\boldsymbol{x}_n \mid z_n = j)} \\ &= \frac{\omega_k N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)}{\sum_j^K \omega_j N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)} \end{aligned}$$

Apply EM to learn GMMs

M-Step:

$$\begin{aligned} \operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n; \boldsymbol{\theta})\right] \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(z_n; \boldsymbol{\theta}) + \ln p(\boldsymbol{x}_n | z_n; \boldsymbol{\theta})\right] \\ &= \operatorname*{argmax}_{\{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}} \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left(\ln \omega_k + \ln N(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right) \end{aligned}$$

To find $\omega_1, \ldots, \omega_K$, solve

$$\operatorname*{argmax}_{\boldsymbol{\omega}} \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \ln \omega_k$$

To find each $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$, solve

$$\operatorname*{argmax}_{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k} \sum_{n=1}^{N} \gamma_{nk} \ln N(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

M-Step (continued)

Solutions to previous two problems are very natural, for each k

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N}$$

i.e. (weighted) fraction of examples belonging to cluster k

$$\boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk} \boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

i.e. (weighted) average of examples belonging to cluster k

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

i.e (weighted) covariance of examples belonging to cluster k

GMM: putting it together

EM for clustering:

Step 0 Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

Step 1 (E-Step) **update the "soft assignment"** (fixing parameters)

$$\gamma_{nk} = p(z_n = k \mid \boldsymbol{x}_n)$$

Step 2 (M-Step) **update the model parameter** (fixing assignments)

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N} \qquad \boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk} \boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

Step 3 return to Step 1 if not converged

Connection to K-means

K-means is in fact a special case of EM for (a simplified) GMM:

Let $\boldsymbol{\Sigma}_k = \sigma^2 \boldsymbol{I}$ for some fixed σ, so only ω_k and $\boldsymbol{\mu}_k$ are parameters.

EM becomes K-means:

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \prod_{n=1}^{N} p(\boldsymbol{x}_n \,; \boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \prod_{n=1}^{N} \sum_{k=1}^{K} p(z_n = k) N(\boldsymbol{x}_n | \boldsymbol{\mu}_k)$$

If we assume hard assignments $p(z_n = k) = 1$, if $k = C(n)$, then

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \prod_{n=1}^{N} p(\boldsymbol{x}_n \,; \boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \prod_{n=1}^{N} N(\boldsymbol{x}_n | \boldsymbol{\mu}_{C(n)})$$

$$= \operatorname*{argmax}_{\boldsymbol{\theta}} \prod_{n=1}^{N} \exp\left(\frac{-1}{2\sigma^2} \|\boldsymbol{x}_n - \boldsymbol{\mu}_{C(n)}\|_2^2 \right) = \operatorname*{argmin}_{\boldsymbol{\mu}, C} \sum_{n=1}^{N} \|\boldsymbol{x}_n - \boldsymbol{\mu}_{C(n)}\|_2^2$$

GMM is a soft version of K-means and it provides a probabilistic interpretation of the data.

Outline

Problem 1

Which of the following statements of the Expectation-Maximization (EM) algorithm is true?

(A) Before running the EM algorithm, we need to choose the step size.

(B) EM always converges to the global optimum of the likelihood.

(C) In EM, the lower-bound for the log-likelihood function we maximize is always non-concave.

(D) None of the above.

Problem 2

Which of the following statements of Gaussian Mixture Model (GMM) is true?

(A) GMM is a non-parametric method that all the training samples need to be stored.

(B) GMM is a probabilistic model that can be used to explain how data is generated.

(C) When learning a GMM, the labels of the samples are available.

(D) None of the above

Problem 3

Which of the following statements of the Expectation-Maximization (EM) algorithm is correct?

(A) The EM algorithm maximizes the expectation of latent variables $E[z_n]$.

(B) In the expectation (E) step, we can use an arbitrary distribution $q_n(z_n)$ to maximize the lower bound of the log-likelihood $P(\Theta)$.

(C) In the maximization (M) step, the objective likelihood is guaranteed to be increased by updating the model parameters (if not converged).

(D) None of above.

Problem 4

Which of the following statements of Gaussian Mixture Model (GMM) is correct?

(A) The parameters of a GMM can be learned via maximum-likelihood estimation (MLE).

(B) Gradient descent cannot be used to learn a GMM. not changed since this is the incorrect answer

(C) GMM is a supervised learning method.

(D) None of above.

Problem 5

Which of the following statements of GMM and K-means is correct?

(A) Given a set of data points and a fixed number of clusters K, applying GMM and K-means will always result in same cluster centroids.

(B) Given a set of data points and a fixed number of clusters K, applying GMM and K-means will always result in different cluster centroids.

(C) Given a learned GMM, we assign a data point to a cluster if the distance from the data point to its centroid is the smallest.

(D) Given a learned K-means model, we assign a data point to a cluster if the distance from the data point to its centroid is the smallest.

Problem 6

Which of the following is not correct?

(A) We can generate novel samples of data from a Gaussian Mixture Model.

(B) Naive Bayes classier (both continuous or discrete input) is a linear classier.

(C) Parameters of Logistic Regression can be derived in closed form from parameters of Naive Bayes classifier.

(D) Generative models can perform better when there is less labelled data for training compared to discriminative models.

Problem 7

Let x be a one-dimensional random variable distributed according to a mixture of two Gaussian distributions

$$P(x) = \omega_1 N\left(x\middle|\mu_1, \sigma_1^2\right) + \omega_2 N(x|\mu_2, \sigma_2^2)$$

M-step: derive the updates for the parameters ω_k and σ_k^2.

Solution

Outline

Density estimation

Observe what we have done indirectly for clustering with GMMs is:

Given a training set $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, **estimate a density function p that could have generated this dataset** (via $\boldsymbol{x}_n \overset{i.i.d.}{\sim} p$).

We say that a random variable x has a probability distribution $p(x)$.

This is exactly the problem of *density estimation*, another important unsupervised learning problem.

Useful for many downstream applications

- we have seen clustering already, will see more applications today
- these applications also *provide a way to measure quality of the density estimator*

Parametric generative models

Parametric estimation assumes a generative model parametrized by $\boldsymbol{\theta}$:

$$p(\boldsymbol{x}) = p(\boldsymbol{x}\,; \boldsymbol{\theta})$$

here $p(x)$ is a common (predefined) probability distribution. Examples:

- GMM: $p(\boldsymbol{x}\,; \boldsymbol{\theta}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ where $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$
- Multinomial for 1D examples with K possible values

$$p(x = k\,; \boldsymbol{\theta}) = \theta_k$$

  where $\boldsymbol{\theta}$ is a distribution over K elements.

Size of $\boldsymbol{\theta}$ is independent of the training set size, so it's parametric.

Parametric methods

Again, we apply **MLE** to learn the parameters $\boldsymbol{\theta}$:

$$\operatorname*{argmax}_{\boldsymbol{\theta}} = \sum_{n=1}^{N} \ln p(x_n\,; \boldsymbol{\theta})$$

For some cases this is intractable and we can use EM to approximately solve MLE (e.g. GMMs).

For some other cases this admits a simple closed-form solution (e.g. multinomial).

MLE for multinomial

$$\begin{aligned}\operatorname*{argmax}_{\boldsymbol{\theta}} &= \sum_{n=1}^{N} \ln p(x = x_n\,; \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \theta_{x_n} \\ &= \sum_{k=1}^{K} \sum_{n: x_n = k} \ln \theta_k = \sum_{k=1}^{K} z_k \ln \theta_k\end{aligned}$$

where $z_k = |\{n : x_n = k\}|$ is **the number of examples with value k**.

The solution (verify yourself!) is simply

$$\theta_k = \frac{z_k}{N} \propto z_k,$$

i.e. the fraction of examples with value k.

Nonparametric models

Can we estimate *without* assuming a fixed generative model?

Kernel density estimation (KDE) is a common approach for nonparametric density estimation (without a pre-defined distribution).

Here "kernel" means something different from what we have seen for "kernel function".

The scikit-learn library provides the KernelDensity class that implements KDE.

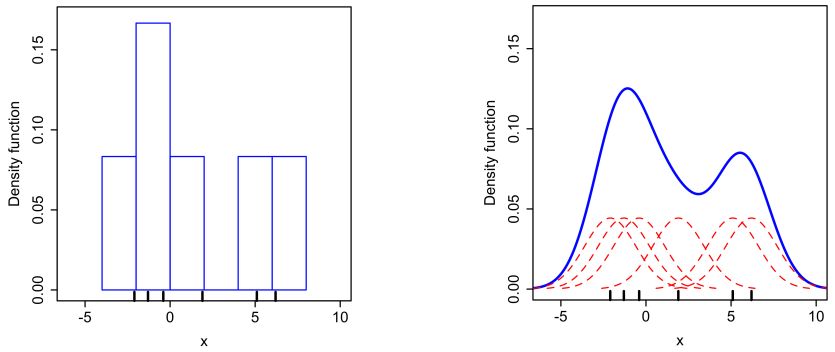
We focus on the 1D (continuous) case.

High level idea

picture from Wikipedia

KDE is closely related to a **histogram**. A histogram is a plot that involves first grouping the observations into bins and counting the number of events that fall into each bin. To construct KDE,

- for each data point, create a "hump" (via a kernel)
- sum up all the humps; more data - a higher hump



Kernel

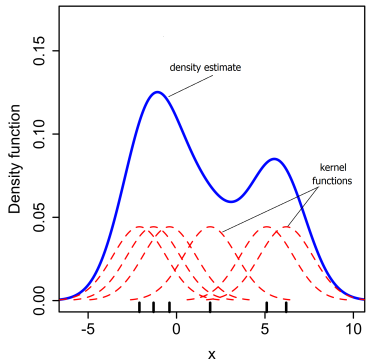
KDE with **a kernel** $K(x)$: $\mathbb{R} \to \mathbb{R}$ centered at x_n:

$$p(x) = \frac{1}{N} \sum_{n=1}^{N} K(x - x_n)$$

Many choices for K, for example, $K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$, the standard Gaussian density

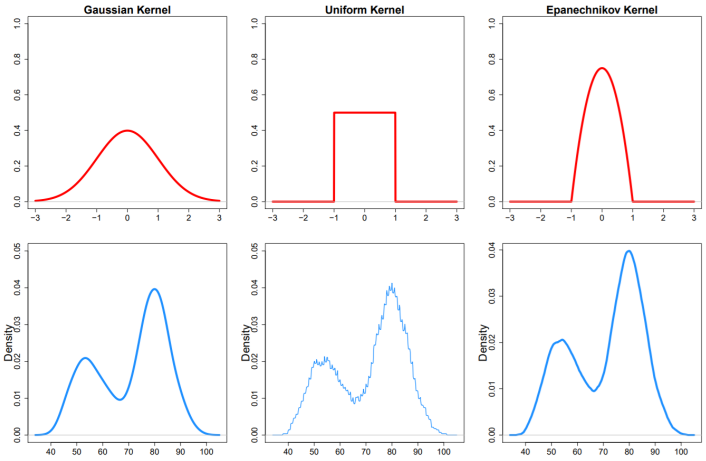
Properties of a kernel:

- symmetry: $K(x) = K(-x)$
- $\int_{-\infty}^{\infty} K(x)dx = 1$, this insures p is a density function.



Different kernels $K(x)$

$$\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \qquad \frac{1}{2}\mathbb{I}[|x| \le 1] \qquad \frac{3}{4}\max\{1 - x^2, 0\}$$



Bandwidth

If $K(x)$ is a kernel, then for any $h > 0$

$$K_h(u) \triangleq \frac{1}{h} K\left(\frac{x}{h}\right) \qquad \text{(stretching the kernel)}$$

can be used as a kernel too (verify the two properties yourself)

So, general KDE is determined by both the kernel K and the bandwidth h

$$p(x) = \frac{1}{N}\sum_{n=1}^{N} K_h\left(x - x_n\right) = \frac{1}{Nh}\sum_{n=1}^{N} K\left(\frac{x - x_n}{h}\right)$$

- x_n controls the center of each hump
- h controls the width/variance of the humps

Effect of bandwidth

picture from Wikipedia

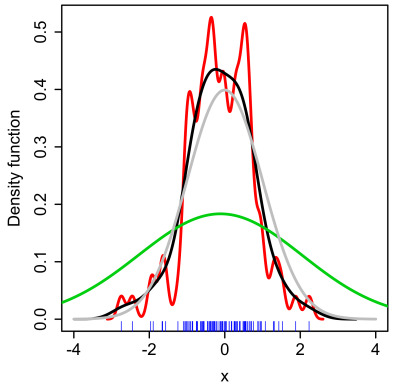
A larger h will smooth a density.

A small h will yield a density that is spiky and very hard to interpret.

Assume Gaussian kernel.

Gray curve is ground-truth

- Red: $h = 0.05$
- Black: $h = 0.337$
- Green: $h = 2$



Bandwidth selection

Selecting h is a deep topic

- one can also do cross-validation based on downstream applications
- there are theoretically-motivated approaches

  Find a value of h that minimizes the error between the estimated density and the true density:

$$\mathbb{E}\left[\left(p_{KDE}(x) - p(x)\right)^2\right] = \mathbb{E}\left[p_{KDE}(x) - p(x)\right]^2 + Var\left[p_{KDE}(x)\right]$$

  This expression is an example of the bias-variance tradeoff, which we saw in the earlier lecture.

Summary

This was a gentle introduction to probability density estimation.

- Histogram provides a fast and reliable way to visualize the probability density of data.
- Parametric probability density estimation involves selecting a common distribution and estimating the parameters for the density function from data.
- Nonparametric probability density estimation involves using an algorithm (KDE) to fit a model to the arbitrary distribution of data.

Outline

Bayes optimal classifier

Suppose the data $(\boldsymbol{x}_n, y_n)$ is drawn from a joint distribution $p(x, y)$, the **Bayes optimal classifier** is

$$f^*(\boldsymbol{x}) = \operatorname*{argmax}_{c \in [\mathsf{C}]} p(c \mid \boldsymbol{x})$$

i.e. predict the class with the largest conditional probability.

$p(x, y)$ is of course unknown, but we can estimate it, which is *exactly a density estimation problem!*

Observe that

$$p(\boldsymbol{x}, y) = p(y)p(\boldsymbol{x} \mid y)$$

To estimate $p(\boldsymbol{x} \mid y = c)$ for some $c \in [\mathsf{C}]$, we are doing density estimation using data with label $y = c$.

Discrete features

For a label $c \in [\mathsf{C}]$,

$$p(y = c) = \frac{|\{n : y_n = c\}|}{N}$$

For each possible value k of a discrete feature d,

$$p(x_d = k \mid y = c) = \frac{|\{n : x_{nd} = k, y_n = c\}|}{|\{n : y_n = c\}|}$$

Continuous features

If the feature is continuous, we can do

- parametric estimation, e.g. via a Gaussian

$$p(x_d = x \mid y = c) = \frac{1}{\sqrt{2\pi}\sigma_{cd}} \exp\left(-\frac{(x - \mu_{cd})^2}{2\sigma_{cd}^2}\right)$$

  where μ_{cd} and σ_{cd}^2 are the empirical mean and variance of feature d among all examples with label c.

- or nonparametric estimation, e.g. via a kernel K and bandwidth h:

$$p(x_d = x \mid y = c) = \frac{1}{|\{n : y_n = c\}|} \sum_{n : y_n = c} K_h(x - x_{nd})$$

How to predict?

Using Naive Bayes assumption:

$$p(\boldsymbol{x} \mid y = c) = \prod_{d=1}^{\mathsf{D}} p(x_d \mid y = c)$$

the **prediction** for a new example x is

$$\begin{aligned}
\operatorname*{argmax}_{c \in [\mathsf{C}]} p(y = c \mid \boldsymbol{x}) &= \operatorname*{argmax}_{c \in [\mathsf{C}]} \frac{p(\boldsymbol{x} \mid y = c) p(y = c)}{p(\boldsymbol{x})} \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(p(y = c) \prod_{d=1}^{\mathsf{D}} p(x_d \mid y = c) \right) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln p(y = c) + \sum_{d=1}^{\mathsf{D}} \ln p(x_d \mid y = c) \right)
\end{aligned}$$

Naive Bayes

For **discrete features**, plugging in previous MLE estimations gives

$$\begin{aligned}
&\operatorname*{argmax}_{c \in [\mathsf{C}]} p(y = c \mid x) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln p(y = c) + \sum_{d=1}^{\mathsf{D}} \ln p(x_d \mid y = c) \right) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln |\{n : y_n = c\}| + \sum_{d=1}^{\mathsf{D}} \ln \frac{|\{n : x_{nd} = x_d, y_n = c\}|}{|\{n : y_n = c\}|} \right)
\end{aligned}$$

Naive Bayes

For **continuous features** with a Gaussian model,

$$\begin{aligned}
&\operatorname*{argmax}_{c \in [\mathsf{C}]} p(y = c \mid \boldsymbol{x}) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln p(y = c) + \sum_{d=1}^{\mathsf{D}} \ln p(x_d \mid y = c) \right) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln |\{n : y_n = c\}| + \sum_{d=1}^{\mathsf{D}} \ln \left(\frac{1}{\sqrt{2\pi}\sigma_{cd}} \exp\left(-\frac{(x_d - \mu_{cd})^2}{2\sigma_{cd}^2} \right) \right) \right) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln |\{n : y_n = c\}| - \sum_{d=1}^{\mathsf{D}} \left(\ln \sigma_{cd} + \frac{(x_d - \mu_{cd})^2}{2\sigma_{cd}^2} \right) \right)
\end{aligned}$$

Connection to logistic regression

Let us fix the variance for each feature to be σ (i.e. not a parameter of the model any more), then the prediction becomes

$$\begin{aligned}
&\operatorname*{argmax}_{c \in [\mathsf{C}]} p(y = c \mid \boldsymbol{x}) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln |\{n : y_n = c\}| - \sum_{d=1}^{\mathsf{D}} \left(\ln \sigma + \frac{(x_d - \mu_{cd})^2}{2\sigma^2} \right) \right) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln |\{n : y_n = c\}| - \frac{\|\boldsymbol{x}\|_2^2}{2\sigma^2} - \sum_{d=1}^{\mathsf{D}} \frac{\mu_{cd}^2}{2\sigma^2} + \sum_{d=1}^{\mathsf{D}} \frac{\mu_{cd}}{\sigma^2} x_d \right) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(w_{c0} + \sum_{d=1}^{\mathsf{D}} w_{cd} x_d \right) = \operatorname*{argmax}_{c \in [\mathsf{C}]} \boldsymbol{w}_c^{\mathrm{T}} \boldsymbol{x} \qquad (\textit{linear classifier!})
\end{aligned}$$

where we denote $w_{c0} = \ln |\{n : y_n = c\}| - \sum_{d=1}^{\mathsf{D}} \frac{\mu_{cd}^2}{2\sigma^2}$ and $w_{cd} = \frac{\mu_{cd}}{\sigma^2}$.

Connection to logistic regression

You can verify

$$p(y = c \mid x) \propto e^{\boldsymbol{w}_c^{\mathrm{T}} \boldsymbol{x}}$$

This is exactly the **softmax** function, the same model we used for a probabilistic interpretation of logistic regression!

So what is different then? They **learn the parameters in different ways**:

- both via MLE, one on $p(y = c \mid x)$, the other on $p(x, y)$
- solutions are different: logistic regression has no closed-form, naive Bayes admits a simple closed-form

Two different modeling paradigms

Suppose the training data is from an *unknown* joint probabilistic model $p(\boldsymbol{x}, y)$. There are two kinds of classification models in machine learning — generative models and discriminative models.

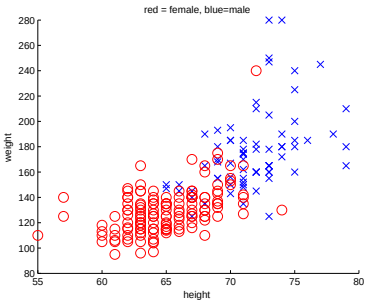
Differences in *assuming* models for the data

- the generative approach requires we specify the model for the joint distribution (such as Naive Bayes), and thus, maximize the *joint* likelihood $\sum_n \log p(\boldsymbol{x}_n, y_n)$
- the discriminative approach (discriminative) requires only specifying a model for the conditional distribution (such as logistic regression), and thus, maximize the *conditional* likelihood $\sum_n \log p(y_n | \boldsymbol{x}_n)$
- Sometimes, modeling by discriminative approach is easier
- Sometimes, parameter estimation by generative approach is easier

Generative model v.s discriminative model

	Discriminative model	Generative model
Example	logistic regression	naive Bayes
Model	conditional $p(y \mid x)$	joint $p(x, y)$ (might have same $p(y \mid x)$)
Learning	MLE	MLE
Accuracy	usually better for large N	usually better for small N
Remark		more flexible, can generate data after learning

Determining sex (man or woman) based on measurements

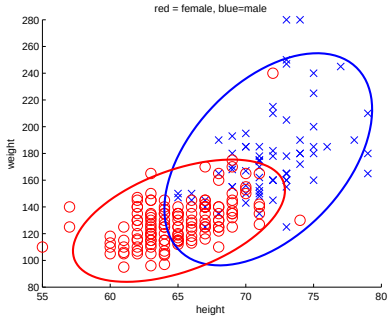


Example: Generative approach

Propose a model of the joint distribution of (x = height, y =sex)

our data

Sex	Height
1	6′
2	5′2”
1	5′6”
1	6′2”
2	5.7”
...	...

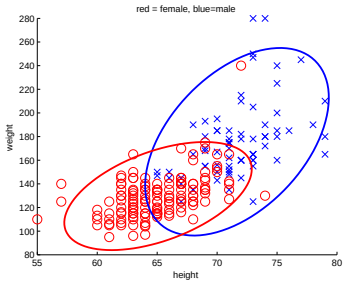


Intuition: we will model how heights vary (according to a Gaussian) in each sub-population (male and female).
Note: This is similar to Naive Bayes for detecting spam emails.

Model of the joint distribution

$$p(x,y) = p(y)p(x|y)$$
$$= \begin{cases} p_1 \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} & \text{if } y = 1 \\ p_2 \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}} & \text{if } y = 2 \end{cases}$$

where $p_1 + p_2 = 1$ represents two *prior* probabilities that x is given the label 1 or 2 respectively. $p(x|y)$ is assumed to be Gaussians.



Parameter estimation

Likelihood of the training data $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$ with $y_n \in \{1, 2\}$

$$\log P(\mathcal{D}) = \sum_n \log p(x_n, y_n)$$
$$= \sum_{n:y_n=1} \log \left(p_1 \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x_n-\mu_1)^2}{2\sigma_1^2}} \right)$$
$$+ \sum_{n:y_n=2} \log \left(p_2 \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(x_n-\mu_2)^2}{2\sigma_2^2}} \right)$$

Maximize the likelihood function

$$(p_1^*, p_2^*, \mu_1^*, \mu_2^*, \sigma_1^*, \sigma_2^*) = \operatorname{argmax} \log P(\mathcal{D})$$

Decision boundary

The decision boundary between two classes is defined by

$$p(y = 1|x) \geq p(y = 2|x)$$

which is equivalent to

$$p(x|y = 1)p(y = 1) \geq p(x|y = 2)p(y = 2)$$

Namely,

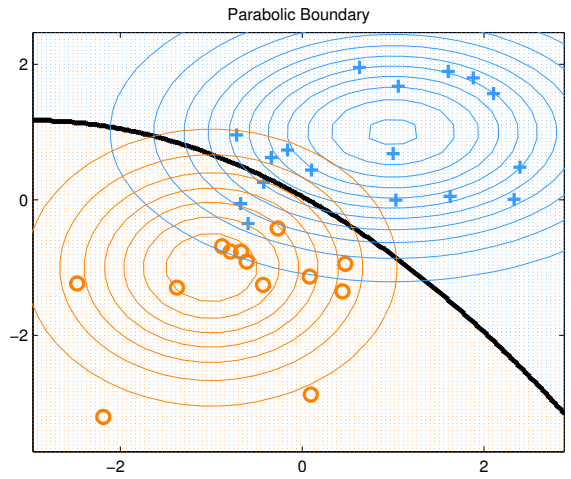
$$-\frac{(x-\mu_1)^2}{2\sigma_1^2} - \log\sqrt{2\pi}\sigma_1 + \log p_1 \geq -\frac{(x-\mu_2)^2}{2\sigma_2^2} - \log\sqrt{2\pi}\sigma_2 + \log p_2$$

It is quadratic in x. It follows (for some a, b and c, that

$$ax^2 + bx + c \geq 0$$

The decision boundary is *not linear*!

Example of nonlinear decision boundary



Note: the boundary is characterized by a quadratic function, giving rise to the shape of parabolic curve.

A special case

What if we assume the two Gaussians have the same variance?

We will get a *linear* decision boundary

From the previous slide:

$$-\frac{(x-\mu_1)^2}{2\sigma_1^2} - \log\sqrt{2\pi}\sigma_1 + \log p_1 \geq -\frac{(x-\mu_2)^2}{2\sigma_2^2} - \log\sqrt{2\pi}\sigma_2 + \log p_2$$

Setting $\sigma_1 = \sigma_2$, we obtain

$$bx + c \geq 0$$

Note: equal variances across two different categories could be a very strong assumption.

For example, the plot suggests that the *male* population has slightly bigger variance (i.e., bigger eclipse) than the *female* population.

Outline

Markov Models

Markov models are powerful **probabilistic models** to analyze sequential data. A.A.Markov (1856-1922) introduced the Markov chains in 1906 when he produced the first theoretical results for stochastic processes. They are now commonly used in

- text or speech recognition
- stock market prediction
- bioinformatics
- ...