Learning From Data
Lecture 4: Generative Learning Algorithms

Yang Li yangli@sz.tsinghua.edu.cn

October 9, 2020

Today's Lecture

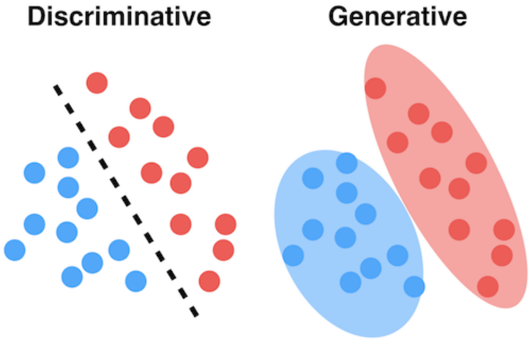
Supervised Learning (Part II)

- Discriminative & Generative Models
- Gaussian Discriminant Analysis
- Naïve Bayes

Discriminative & Generative Models

Two Learning Approaches

Classify input data x into two classes $y \in \{0, 1\}$

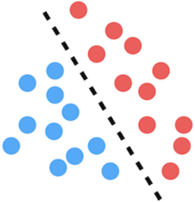


Discriminate between classes of data points

Model the underlying distribution of the data

Discriminative Learning Algorithms

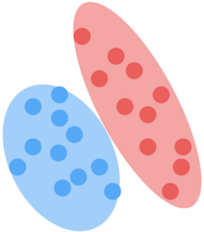
A class of learning algorithms that try to learn the **conditional probability** $p(y|x)$ directly or learn mappings directly from $\mathcal{X}$ to $\mathcal{Y}$.



- e.g. linear regression, logistic regression, k-Nearest Neighbors ...

Generative Learning Algorithms

A class of learning algorithms that model the **joint probability** $p(x, y)$.



- Equivalently, generative algorithms model $p(x|y)$ and $p(y)$
- $p(y)$ is called the **class prior**
- Learned models are transformed to $p(y|x)$ later to classify data using Bayes' rule

Bayes Rule

The posterior distribution on y given x:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

Bayes Rule

The posterior distribution on y given x:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

Make predictions in a generative model:

$$\begin{aligned}\arg_{y} p(y|x) &= \arg_{y} \frac{p(x|y)p(y)}{p(x)} \\ &= \arg_{y} p(x|y)p(y)\end{aligned}$$

No need to calculate $p(x)$.

Generative Models

Generative classification algorithms:

- Continuous input: Gaussian Discriminant Analysis
- Discrete input: Naïve Bayes

Gaussian Discriminant Analysis

Gaussian Discriminant Analysis: Overview

Goal

Binary classification with input in $\mathcal{X} = \mathbb{R}^n$ and label in $\mathcal{Y} = \{0, 1\}$

Main steps

1. Select a *data generating distribution* .

$$y \sim Bernoulli(\phi)$$
$$x|y = 0 \sim N(\mu_0, \Sigma), x|y = 1 \sim N(\mu_1, \Sigma)$$

2. Estimate model parameters ϕ, μ_0 ,μ_1 and Σ from training data.
3. For any new sample x', predict its label by computing $p(y|x = x'; \phi, \mu_0, \mu_1, \Sigma)$

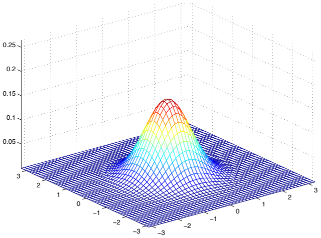
Multivariate Normal Distribution

Multivariate normal (or multivariate Gaussian) distribution
$N(\mu, \Sigma)$

- $\mu \in \mathbb{R}^n$ is the mean vector,
- $\Sigma \in \mathbb{R}^{n \times n}$ is the covariance matrix. Σ is symmetric and SPD.

Density function:

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)}$$



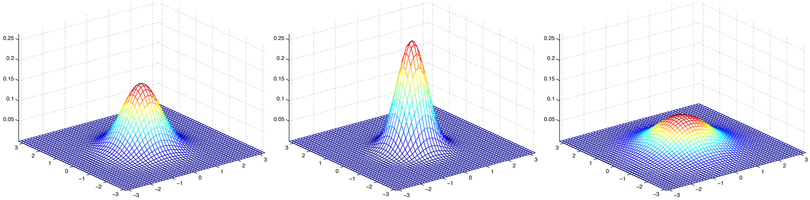
Multivariate Normal Distribution

Let $X \in \mathbb{R}^n$ be a random vector. If $X \sim N(\mu, \Sigma)$,

$$\mathbb{E}[X] = \int_x p(x; \mu, \Sigma) dx = \mu$$

$$\text{Cov}(X) = \mathbb{E}\left[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T\right] = \Sigma$$

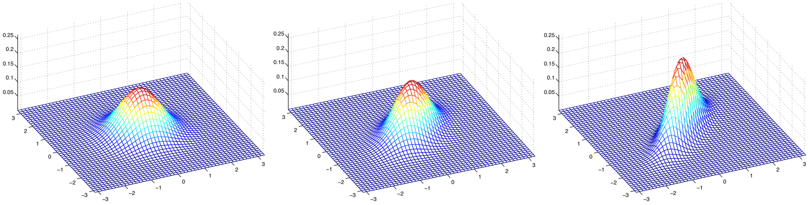
Gaussian Discriminative Analysis



$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} 0.6 & 0 \\ 0 & 0.6 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Diagonal entries of Σ controls the "spread" of the distribution

Gaussian Discriminative Analysis



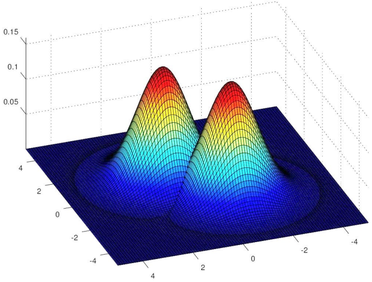
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

The distribution is no longer oriented along the axes when off-diagonal entries of Σ are non-zero.

Gaussian Discriminant Analysis (GDA) Model

Given parameters $\phi, \mu_0, \mu_1, \Sigma$,

$$y \sim \text{Bernoulli}(\phi)$$
$$x|y = 0 \sim \mathcal{N}(\mu_0, \Sigma)$$
$$x|y = 1 \sim \mathcal{N}(\mu_1, \Sigma)$$



Probability density functions:

$$p(y) = \phi^y (1 - \phi)^{1-y}$$
$$p(x|y = 0) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{\left(-\frac{1}{2}(x-\mu_0)^T \Sigma^{-1}(x-\mu_0)\right)}$$
$$p(x|y = 1) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{\left(-\frac{1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1)\right)}$$

Log likelihood of the data:

$$
\begin{aligned}
l(\phi, \mu_0, \mu_1, \Sigma) &= \log \prod_{i=1}^{m} p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma) \\
&= \log \prod_{i=1}^{m} p(x^{(i)} | y^{(i)}; \mu_0, \mu_1, \Sigma) p(y^{(i)}; \phi)
\end{aligned}
$$

Log likelihood of the data:

$$
\begin{aligned}
l(\phi, \mu_0, \mu_1, \Sigma) &= \log \prod_{i=1}^{m} p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma) \\
&= \log \prod_{i=1}^{m} p(x^{(i)} | y^{(i)}; \mu_0, \mu_1, \Sigma) p(y^{(i)}; \phi)
\end{aligned}
$$

Log likelihood of the data:

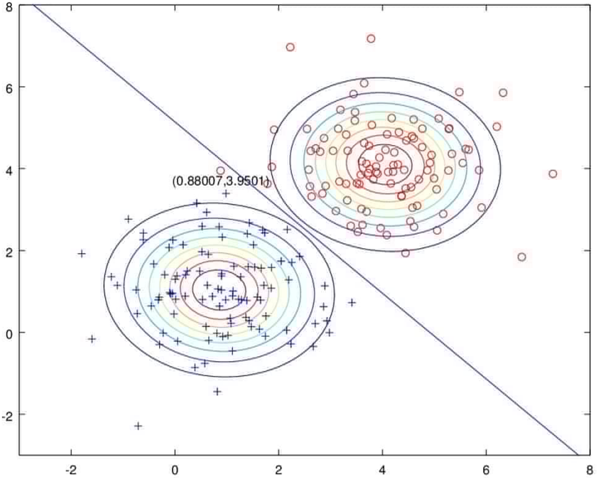
$$
\begin{aligned}
l(\phi, \mu_0, \mu_1, \Sigma) &= \log \prod_{i=1}^{m} p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma) \\
&= \log \prod_{i=1}^{m} p(x^{(i)} | y^{(i)}; \mu_0, \mu_1, \Sigma) p(y^{(i)}; \phi)
\end{aligned}
$$

Maximum likelihood estimate of the parameters:

$$
\begin{aligned}
\phi &= \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = 1\} \\
\mu_b &= \frac{\sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = b\} x^{(i)}}{\sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = b\}} \text{ for } b = 0, 1 \\
\Sigma &= \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T
\end{aligned}
$$

Maximum likelihood estimation of GDA

GDA finds a linear decision boundary at which
$p(y = 1|x) = p(y = 0|x) = 0.5$



GDA and Logistic Regression

$p(y = 1|x; \phi, \mu_0, \mu_1, \Sigma)$ can be written in the form:

$$p(y = 1|x; \phi, \Sigma, \mu_0, \mu_1) = \frac{1}{1 + e^{-\theta^T x}}$$

GDA and Logistic Regression

$p(y=1|x;\phi,\mu_0,\mu_1,\Sigma)$ can be written in the form:

$$p(y=1|x;\phi,\Sigma,\mu_0,\mu_1) = \frac{1}{1+e^{-\theta^T x}}$$

where

$$\theta = \begin{bmatrix}\theta_1 \\ \theta_2\end{bmatrix} = \begin{bmatrix}\Sigma^{-1}(\mu_0 - \mu_1) \\ \log\frac{1-\phi}{\phi} - \frac{1}{2}(\mu_0^T\Sigma^{-1}\mu_0 - \mu_1^T\Sigma^{-1}\mu_1)\end{bmatrix}, x = \begin{bmatrix}x_1 \\ \vdots \\ x_n \\ 1\end{bmatrix}$$

GDA and Logistic Regression

$p(y=1|x;\phi,\mu_0,\mu_1,\Sigma)$ can be written in the form:

$$p(y=1|x;\phi,\Sigma,\mu_0,\mu_1)=\frac{1}{1+e^{-\theta^T x}}$$

where

$$\theta=\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix}=\begin{bmatrix}\Sigma^{-1}(\mu_0-\mu_1)\\ \log\frac{1-\phi}{\phi}-\frac{1}{2}(\mu_0^T\Sigma^{-1}\mu_0-\mu_1^T\Sigma^{-1}\mu_1)\end{bmatrix}, x=\begin{bmatrix}x_1\\ \vdots\\ x_n\\ 1\end{bmatrix}$$

Similarly,

$$p(y=0|x;\phi,\Sigma,\mu_0,\mu_1)=\frac{1}{1+e^{\theta^T x}}$$

If $p(x|y)\sim\mathcal{N}(\mu,\Sigma)$, $p(y|x)$ is a logistic function.

GDA and Logistic Regression

GDA

- Maximizes the **joint likelihood** $\prod_{i=1}^{m} p(x^{(i)}, y^{(i)})$
- Modeling assumptions: $x|y{=}b \sim \mathcal{N}(\mu_b, \Sigma)$, $y \sim \text{Bernoulli}(\phi)$
- When modeling assumptions are correct, GDA is **asymptotically efficient** and **data efficient**

Logistic Regression

- Maximizes the **conditional likelihood** $\prod_{i=1}^{m} p(y^{(i)}|x^{(i)})$
- Modeling assumptions: $p(y|x)$ is a logistic function; no restriction on $p(x)$
- More robust and less sensitive to incorrect modeling assumptions.

Naïve Bayes

Naïve Bayes: Motivationg Example

A simple generative learning algorithm for discrete input variables

Example: Spam filter (document classification)

Classify email messages x to spam ($y = 1$) and non-spam ($y = 0$) classes.

Hello [redacted]
We need to confirm your info...
(1) FINAL MESSAGE: Payout Verification - $3000 PAYOUT is ready to be addressed in your Name and we want to be sure it gets to the right place. Click below to start the confirmation process. The sooner you act, the sooner it can be in your hands!
Raging Bull Casino

A sample spam email

Example: Spam Filter

Binary text features

Given a dictionary of size n, represent a message composed of dictionary words as $x \in \{0, 1\}^n$:

$$x_i = \begin{cases} 1 & i\text{-th dictionary word is in message} \\ 0 & \text{otherwise} \end{cases}$$

$$x = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \textit{a} \\ \textit{aardvark} \\ \vdots \\ \textit{casino} \\ \vdots \\ \textit{payout} \\ \vdots \\ \textit{zyzzyva} \end{matrix}$$

Naïve Bayes Model

Probability of observing email $x_1, \ldots, x_n$ given spam class y :

$$p(x_1, \ldots, x_n | y) = p(x_1|y)p(x_2|y, x_1), \ldots, p(x_n|y, x_1, \ldots, x_{n-1})$$

Naïve Bayes (NB) assumption

x_i's are conditionally independent given y:

$$p(x_i|y, x_1, \ldots, x_{i-1}) = p(x_i|y)$$

$$p(x_1, \ldots, x_n|y) = p(x_1|y)p(x_2|y) \ldots p(x_n|y) = \prod_{i=1}^{n} p(x_i|y)$$

Naïve Bayes Parameters

Multi-variate Bernoulli event model

$x|y$ generated from n independent Bernoulli trials

$$p(x, y) = p(y)p(x|y) = p(y)\prod_{i=1}^{n} p(x_i|y)$$

- $y \sim Bernoulli(\phi_y)$: assume email class (spam vs no-spam) is randomly generated with prior $p(y) = \phi_y^y(1-\phi_y)^{1-y}$
- $x_i|y = b \sim Bernoulli(\phi_{i|y=b}), b = 1, 2$: given $y = b$, each word x_i is included in the message independently with $p(x_i = 1|y = b) = \phi_{i|y=b}$. i.e.

$$p(x_i|y = b) = \phi_{i|y=b}^{x_i}(1 - \phi_{i|y=b})^{1-x_i}$$

Model parameters:

- ϕ_y
- $\phi_{i|y=1}, \phi_{i|y=0}$ for $i = 1, \ldots, n$

Naïve Bayes Parameter Learning

Likelihood of training data $(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$:

$$L(\phi_y, \phi_{j|y=0}, \phi_{j|y=1}) = \prod_{i=1}^{m} p(x^{(i)}, y^{(i)})$$

Maximum likelihood estimation of parameters:

$$\phi_y = \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = 1\}$$ *% of spam emails*

$$\phi_{j|y=b} = \frac{\sum_{i=1}^{m} \mathbf{1}\{x_j^{(i)} = 1, y^{(i)} = b\}}{\sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = b\}} \text{ for } b = 1, 0$$

% of spam(non-spam) emails containing jth dictionary word

Naïve Bayes Prediction

Given new example with feature x, compute the posterior probability

$$
\begin{aligned}
p(y=1|x) &= \frac{p(x|y=1)p(y=1)}{p(x)} \\
&= \frac{p(x|y=1)p(y=1)}{p(x|y=1)p(y=1)+p(x|y=0)p(y=0)} \\
&= \frac{\prod_{i=1}^{n} p(x_i|y=1)p(y=1)}{\prod_{i=1}^{n} p(x_i|y=1)p(y=1)+\prod_{i=1}^{n} p(x_i|y=0)p(y=0)}
\end{aligned}
$$

Choose label $y = 1$ (spam) if $p(y = 1|x) > T$ where $T \in [0, 1]$ is a threshold .. e.g. $T = 0.5$
T tradeoff between wrongly blocked non-spam (FPs) vs. wrongly blocked spams (FNs).

Laplace smoothing

Issue with Naïve Bayes prediction:

- Suppose word x_j hasn't been seen in the training data, $\phi_{j|y=1} =$

Laplace smoothing

Issue with Naïve Bayes prediction:

- Suppose word x_j hasn't been seen in the training data, $\phi_{j|y=1} = \phi_{j|y=0} = 0$
- Can not compute class posterior $p(y = 1|x) = \frac{0}{0}$.

Laplace smoothing

Issue with Naïve Bayes prediction:

- Suppose word x_j hasn't been seen in the training data, $\phi_{j|y=1} = \phi_{j|y=0} = 0$
- Can not compute class posterior $p(y = 1|x) = \frac{0}{0}$.

Laplace smoothing

Let $z \in \{1, \ldots, k\}$ be a multinomial random variable. Given m independent observations $z^{(1)} \ldots z^{(m)}$, maximum likelihood estimation of $\phi_j = p(z = j)$ with **Laplace smoothing** is

$$\phi_j = \frac{\sum_{i=1}^{m} 1\{z^{(i)} = j\} + 1}{m + k}$$

- $\phi_j \neq 0$ for all j
- $\sum_{j=1}^{k} \phi_j = 1$

Naïve Bayes with Laplace smoothing

Apply Laplace smoothing to $\phi_{j|y=b}$ for $b \in \{0, 1\}$

$$\phi_{j|y=b} = \frac{\sum_{i=1}^{m} \mathbf{1}\{x_j^{(i)} = 1, y^{(i)} = b\} + 1}{\sum_{i=1}^{m} \mathbf{1}\{y^i = b\} + 2}$$

In practice we don't apply Laplace smoothing to $\phi_y = p(y = 1)$, which is greater than 0.

Naïve Bayes and Multinomial Event Model

Alternative text representation

- $x_i \in \{1, \ldots, K\}$ where K is the dictionary size
- Represent email of n words as $x = \{x_1, \ldots, x_n\}$

"a free gift..." $\rightarrow$ $\{x_1 = 1, x_2 = 1300, x_3 = 2433, \ldots\}$

dictionary id	1	2	...	1300	...	2433	...
word	a	aa	...	free	...	gift	...

Naive Bayes and Multinomial Event Model

Multinomial event model

- first sampling $y \in \{0, 1\}$ from $p(y)$

$$y \sim Bernoulli(\phi_y)$$

- Select $x_1, x_2, \ldots, x_n$ independently from the same Multinomial distribution $p(x_i|y)$

$$x_i|y = b \sim Multinomial(\phi_{1|y=b}, \ldots, \phi_{K|y=b}), b = 0, 1$$
$$\phi_{k|y=b} = p(x_j = k|y = b) \text{ for all } j \in \{1, \ldots, n\}$$

For any word k in the dictionary, $\phi_{k|y}$ is the probability of k appear in an email given email class y

- Joint probability: $p(x_1, \ldots, x_n, y) = p(y) \prod_{i=1}^{n} p(x_i|y)$

Multinomial event model parameters

Assume $p(x_j = k|y)$ is the same for all j

- $\phi_y = p(y)$
- $\phi_{k|y=1} = p(x_j = k|y = 1)$ for $k = 1, \ldots, n$
- $\phi_{k|y=0} = p(x_j = k|y = 0)$ for $k = 1, \ldots, n$

Likelihood of training set $(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$:

$$
\begin{aligned}
L(\phi_y, \phi_{k|y=0}, \phi_{k|y=1}) &= \prod_{i=1}^{m} p(x^{(i)}, y^{(i)}) \\
&= \prod_{i=1}^{m} p(x_1^{(i)}, \ldots, x_n^{(i)}, y^{(i)}) \\
&= \prod_{i=1}^{m} p(y^{(i)}; \phi_y) \prod_{j=1}^{n_i} p(x_j{}^{(i)}|y; \phi_{k|y=0}, \phi_{k|y=1})
\end{aligned}
$$

where n_i is the # words in the i-th email.

Maximum likelihood estimation with Laplace smoothing

- $\phi_y = \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = 1\}$
- $\phi_{k|y=1} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n_i} \mathbf{1}\{x_j^{(i)} = k, y^{(i)} = 1\} + 1}{\sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = 1\} n_i + K}$
- $\phi_{k|y=0} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n_i} \mathbf{1}\{x_j^{(i)} = k, y^{(i)} = 0\} + 1}{\sum_{i=1}^{m} \mathbf{1}\{y^{(i)} = 0\} n_i + K}$

K is the dictionary size.