Introduction to Machine Learning

Lecture 6
Supervised Learning

- Least Squares Method
- Linear Regression
- Logistic Regression

- Some materials are courtesy of Vibhave Gogate and Tom Mitchell.
- All pictures belong to their creators.

Supervised Learning

- Given some training examples $< x, f(x) >$ and an unknown function f.
- Find a good approximation of f.

Recap

- Step 1. Select the features of x to be used.
- Step 2. Select a hypothesis space H: a set of candidate functions to approximate f.
- **Step 3. Select a measure to evaluate the functions in H.**
- Step 4. Use a machine learning algorithm to find the best function in H according to your measure.

Supervised Learning

- Step 3. Select a **measure** to evaluate the functions in H.

- What are the measures we have used?

Recap

- Concept learning: if there hypothesis is consistent with data.
- Decision tree: information gain
- Bayesian learning: select the most probable hypothesis or classification

Not consistent?
Do not want probabilities?
Error-driver approaches!

Least Squares Method (LSM)

- Given some training examples $< x, f(x) >$ and an unknown function f.
- Find a good approximation of f.

- Suppose we have the training data D.
- Suppose we are considering a hypothesis space H.
- For each $h \in H$, let us define the error over D as
- $ERROR(h) = \sum_{x \in D} |f(x) - h(x)|^2$

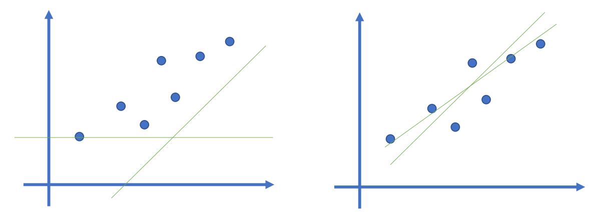
Sum of individual error

- If h is the true function, ideally $ERROR(h) = 0$

- **LSM** method: select the h in H such that the error is **minimized**.

Linear Regression

- **Setting**: x and $f(x)$ are two real numbers
- **Linear Regression**: assume they have the relationship $f(x) = ax + b$ for two constants a, b.
- We have applied some prior knowledge.



Linear Regression

- **Setting**: x and $f(x)$ are two real numbers
- **Linear Regression**: assume they have the relationship $f(x) = ax + b$ for two constants a, b.

- We have applied some prior knowledge.

- Apply LSM to decide a and b.

- Deciding a and b is a procedure to search the hypothesis space.

Linear Regression

- Suppose the training data is $(x_1,\ y_1), (x_2,\ y_2), \dots, (x_n,\ y_n)$
- For each pair of a and b, the error is
- $ERROR(a, b) = \sum |y_i - a \cdot x_i - b|^2$

- By calculus or algebra, the a and b that can minimize the above error is

$$\hat{a} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2} \quad \text{and} \quad \hat{b} = \bar{y} - \hat{a} \cdot \bar{x}$$

where $\bar{x} = \frac{\sum x_i}{n}$ and $\bar{y} = \frac{\sum y_i}{n}$

Linear Regression

- Some mathematics:
- $ERROR(h) = \sum |y_i - a \cdot x_i - b|^2$

- **Apply Lagrange multiplier.**

- Partial derivatives.
- $\frac{\partial\ ERROR(h)}{\partial\ b} = -2 \sum (y_i - a \cdot x_i - b)$
- $\frac{\partial\ ERROR(h)}{\partial\ a} = -2 \sum x_i \cdot (y_i - a \cdot x_i - b)$

- Solve $\frac{\partial\ ERROR(h)}{\partial\ b} = 0$ and $\frac{\partial\ ERROR(h)}{\partial\ a} = 0$ (try it yourself)

Linear Regression

- **Linear Regression**: assume they have the relationship $f(x) = ax + b$ for two constants a, b.

$$\hat{a} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2} \quad \text{and} \quad \hat{b} = \bar{y} - \hat{a} \cdot \bar{x}$$

- Look at the solution $y = \hat{a} \cdot x + \hat{b}$
- If the training data is already on a line, the produced solution is exactly that line and error=0.
- The mean point $(\bar{x}, \bar{y})$ must on the line $y = \hat{a} \cdot x + \hat{b}$
- LSM is good.

Linear Regression Multivariable

- **Setting**: $(x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a real number.
- **Linear Regression**: assume they have the relationship

$f(x_i) = \omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0$ for some constants ω_i.

- How to find the parameters (coefficients)?
- $ERROR(\boldsymbol{\omega}) = \sum\left|y_i - (\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0)\right|^2 = \sum\left|y_i - \boldsymbol{\omega}^T \boldsymbol{x_i}\right|^2$

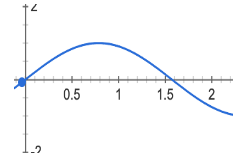
$$\boxed{\begin{aligned}\boldsymbol{\omega} &= (\omega_1, \dots, \omega_k, \omega_0) \\ \boldsymbol{x_i} &= (x_{i,1}, x_{i,2}, \dots, x_{i,k}, 1)\end{aligned}} \qquad \boxed{X = \begin{pmatrix} x_1 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{pmatrix} \quad Y = \begin{pmatrix} y_1 \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{pmatrix}}$$

The $\boldsymbol{\omega}$ that can minimize the error is $\boldsymbol{\omega} = (X^T X)^{-1} X^T Y$

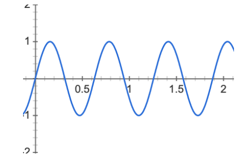
Linear Regression Regularization

- **Regularization**

- Reduce overfitting: simple model, small parameters (absolute value)



$\sin ax, a = 2$



$\sin ax, a = 10$

Range $x = [0,2], y = [-2,2]$

Linear Regression Regularization

- **Regularization**

- Reduce overfitting: simple model, small parameters

Linear Regression Regularization

- **Regularization**

- Reduce overfitting: simple model, small parameters

New cost function: $ERROR_R(\boldsymbol{\omega})$= $ERROR\ (\boldsymbol{\omega})$+regularization

$$ERROR_R(\boldsymbol{\omega}) = \sum\left|y_i - \boldsymbol{\omega}^T \boldsymbol{x_i}\right|^2 + \frac{\lambda}{2}\sum|\omega_j|^q$$

Minimizing $ERROR_R(\boldsymbol{\omega})$ implicitly minimizes $\frac{\lambda}{2}\sum|\omega_j|^q$

$q = 1$: L1 regularization (Lasso)
$q = 2$: L2 regularization

Linear Regression Regularization

- Input $(\boldsymbol{x}, y)$, $\boldsymbol{x}$ is a real vector and y is a real number.

- Assume $y = \omega_1 x_1 + \cdots + \omega_k x_k + \omega_0$

- Define $ERROR(\boldsymbol{\omega})$ over the training data
- Define regularization term.
- Minimize $ERROR_R(\boldsymbol{\omega})$ = $ERROR(\boldsymbol{\omega})$ + regularization.
- Done

Logistic Regression

- **Setting**: $(x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.
- $H(x) = 1$ if $\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0 > 0$
- $H(x) = 0$ if $\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0 \le 0$
- Not smooth.

Logistic Regression

- **Setting**: $(x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.
- $H(x) = 1$ if $\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0 > 0$
- $H(x) = 0$ if $\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0 \le 0$
- Not smooth.

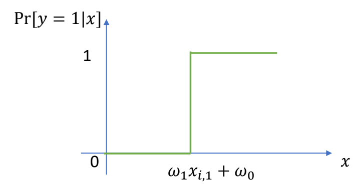
- Assume a form of $\Pr[y = 1|x]$. ($\Pr[y = 0|x]$ =1−$\Pr[y = 1|x]$)
- Classify it as 1 if $\Pr[y = 1|x] \ge \Pr[y = 0|x]$.

- Special case:
- $\Pr[y = 1|x] = 1$ if $\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0 > 0$
- $\Pr[y = 1|x] = 0$ if $\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0 \le 0$

Logistic Regression

- **Setting**: $(x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

- Special case:
- $\Pr[y = 1|x] = 1$ if $\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0 > 0$
- $\Pr[y = 1|x] = 0$ if $\omega_1 x_{i,1} + \cdots + \omega_k x_{i,k} + \omega_0 \le 0$



$\Pr[y = 1|x]$ is not differentiable.

Lagrange multiplier and Gradient Descent requires computing partial derivatives.

We prefer smooth functions.

Logistic Regression

- **Setting**: $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

- Suppose $\Pr[y_i|x_i]$ follows the following distribution

$\exp(x) = e^x$

$$\Pr[y_i = 0|x_i] = \frac{1}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

implies

ω_j are parameters

$$\Pr[y_i = 1|x_i] = \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

Classification Rule:

Classify a new instance x as 1 iff $\Pr[y_i = 1|x_i] \geq \Pr[y_i = 0|x_i]$

Logistic Regression

- **Setting**: $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

$$\Pr[y_i = 0|x_i] = \frac{1}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

$$\Pr[y_i = 1|x_i] = \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

Classification Rule:
Classify a new instance x as 1 iff $\Pr[y_i = 1|x_i] \geq \Pr[y_i = 0|x_i]$

$$\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j}) \geq 1 \quad \Rightarrow \quad \omega_o + \sum_j \omega_j \cdot x_{i,j} \geq 0$$

A linear classifier!

Logistic Regression

- **Setting**: $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

$$\Pr[y_i = 0|x_i] = \frac{1}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

$$\Pr[y_i = 1|x_i] = \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

How to learn the parameter ω? Bayesian learning.

MAP= argmax $\Pr[\boldsymbol{\omega}|D]$=argmax $\Pr[D|\boldsymbol{\omega}]$ $\Pr[\boldsymbol{\omega}]$

No prior, MAP= MLE=argmax $\Pr[D|\boldsymbol{\omega}]$.

Logistic Regression

- **Setting**: $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

$$\Pr[y_i = 0|x_i] = \frac{1}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

$$\Pr[y_i = 1|x_i] = \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

argmax $\Pr[D|\boldsymbol{\omega}]$=argmax $\ln \prod \Pr[y_i|x_i, \boldsymbol{\omega}]$=argmax $\sum \ln \Pr[y_i|x_i, \boldsymbol{\omega}]$

Note: $\ln \Pr[y_i|x_i, \boldsymbol{\omega}] = \underbrace{y_i \ln \Pr[y_i = 1|x_i, \boldsymbol{\omega}]}_{\text{If } y_i = 1} + \underbrace{(1 - y_i) \ln \Pr[y_i = 0|x_i, \boldsymbol{\omega}]}_{\text{If } y_i = 0}$

Logistic Regression

- **Setting**: $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

$$\Pr[y_i = 0|x_i] = \frac{1}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

$$\Pr[y_i = 1|x_i] = \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

Solve an optimization problem

Note: $\ln \Pr[y_i|x_i, \boldsymbol{\omega}] = y_i \ln \Pr[y_i = 1|x_i, \boldsymbol{\omega}] + (1 - y_i) \ln \Pr[y_i = 0|x_i, \boldsymbol{\omega}]$

argmax $\sum \ln \Pr[y_i|x_i, \boldsymbol{\omega}]$

=argmax $\sum[\, y_i \ln \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})} + (1 - y_i) \ln \frac{1}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}]$

Next page

Logistic Regression

- **Setting**: $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

=argmax $\sum[y_i \ln \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})} + (1 - y_i) \ln \frac{1}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}]$

= argmax $\sum[y_i(\omega_0 + \sum_j \omega_j \cdot x_{i,j}) - \ln(1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j}))] = l(\boldsymbol{\omega})$

$$l(\boldsymbol{\omega}) = \sum[y_i(\omega_0 + \sum_j \omega_j \cdot x_{i,j}) - \ln(1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j}))]$$

Logistic Regression

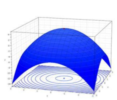
- **Setting**: $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

=argmax $\sum[y_i \ln \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})} + (1 - y_i) \ln \frac{1}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}]$

= argmax $\sum[y_i(\omega_0 + \sum_j \omega_j \cdot x_{i,j}) - \ln(1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j}))] = l(\boldsymbol{\omega})$

$$l(\boldsymbol{\omega}) = \sum[y_i(\omega_0 + \sum_j \omega_j \cdot x_{i,j}) - \ln(1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j}))]$$

Lagrange multiplier or Gradient Ascent?
Bad news: no closed-form to maximize $l(\boldsymbol{\omega})$
Good news: the function is concave.



Logistic Regression

- **Setting**: $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$ is a real vector and y_i is a **binary value**.

$$l(\boldsymbol{\omega}) = \sum[y_i(\omega_0 + \sum_j \omega_j \cdot x_{i,j}) - \ln(1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j}))]$$

$$\frac{\partial\, l(\boldsymbol{\omega})}{\partial\, \omega_j} = \sum y_i x_{i,j} - \frac{x_{i,j} \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})} = \sum x_{i,j}(y_i - \Pr[y_i = 1|x_i])$$

for $j > 0$

- $\omega_j = \omega_j + \eta \frac{\partial\, l(\boldsymbol{\omega})}{\partial\, \omega_j}$
- η: learning rate

Do it yourself for $j = 0$.

Logistic Regression

- **How does the updating rule force the parameter to fit the data?**

$$\frac{\partial\, l(\boldsymbol{\omega})}{\partial\, \omega_j} = \sum y_i x_{i,j} - \frac{x_{i,j} \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1 + \exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})} = \sum x_{i,j}(\underline{y_i - \Pr[y_i = 1|x_i]})$$

for $j > 0$

Difference between observed value and predicted probability.

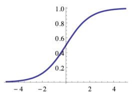
- $\omega_j = \omega_j + \eta \frac{\partial\, l(\boldsymbol{\omega})}{\partial\, \omega_j}$
- η: learning rate

Classify it as 1 if $\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j}) \geq 1$

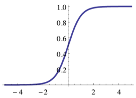
Logistic Regression

- **Regularization**

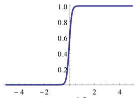
$$\frac{1}{1+\exp(-ax)}$$



a=1



a=5



a=10

- Small parameters → smooth curves
- Maximum likelihood solution: prefers higher weights
- – higher likelihood of (properly classified) examples close to decision boundary
- – larger influence of corresponding features on decision
- Regularization: penalize high weights

Logistic Regression

- **Regularization**
- Method 1: add a term to the objective function, like we did for linear regression

$$l_R(\omega) = l(\omega) - \frac{\lambda}{2}\sum|\omega_j|^2$$

- A hard constraint when we calculate the parameter
- We are maximizing $l_R(\omega)$ so we add a negative term $-\frac{\lambda}{2}\sum|\omega_j|^2$

Logistic Regression

- **Regularization**
- Method 2: assume a prior distribution on the parameter. (so far we assume no prior)

MAP= argmax $\Pr[\boldsymbol{\omega}|D]$=argmax $\Pr[D|\boldsymbol{\omega}]\Pr[\boldsymbol{\omega}]$.

- A common method is to assume ω_i follow normal distribution, zero mean, identity variance. (push the parameter to 0)

$$\Pr[\omega_i] = \frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-\omega_i^2}{2\sigma^2}} \qquad \Pr[\boldsymbol{\omega}] = \prod\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-\omega_i^2}{2\sigma^2}}$$

- $\text{argmax}\ln\Pr[D|\boldsymbol{\omega}]\Pr[\boldsymbol{\omega}] = \text{argmax}\ln\Pr[D|\boldsymbol{\omega}] + \ln\Pr[\boldsymbol{\omega}] = l(\boldsymbol{\omega}) - \lambda\frac{\sum\omega_i^2}{2}$.

Similar effect as method 1!!

Logistic Regression

- **Regularization** $l_R(\omega) = l(\omega) - \frac{\lambda}{2}\sum|\omega_j|^2$
- Update rule
- $\frac{\partial\, l(\omega)}{\partial\, \omega_j} = \sum x_{i,j}(y_i - \Pr[y_i = 1|x_i])$ (see previous slides)
- $\frac{\partial\, l_R(\omega)}{\partial\, \omega_j} = \frac{\partial\, l(\omega)}{\partial\, \omega_j} - \lambda\omega_j$

When $\omega_j > 0$, it pushes ω_j to decrease
When $\omega_j < 0$, it pushes ω_j to increase

- $\omega_j = \omega_j + \eta\frac{\partial\, l(\omega)}{\partial\, \omega_j} - \eta\lambda\omega_j$

Logistic Regression vs Gaussian Naïve Bayes

- **Setting** $x_i = (x_{i,1}, \dots, x_{i,k})$ real-vector and y_i **Boolean value**.
- $\text{argmax}_y \Pr[y|x]$: the most probable classification of x

Logistic Regression vs Gaussian Naïve Bayes

- Setting $x_i = (x_{i,1}, \dots, x_{i,k})$ real-vector and y_i Boolean value.
- $\text{argmax}_y \Pr[y|x]$: the most probable classification of x
- **Logistic Regression: assume**

$$\Pr[y_i = 1|x_i] = \frac{1}{1+\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

$$\Pr[y_i = 0|x_i] = \frac{\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}{1+\exp(\omega_o + \sum_j \omega_j \cdot x_{i,j})}$$

- Estimate ω by data.

Logistic Regression vs Gaussian Naïve Bayes

- Setting $x = (x_1, \dots, x_k)$ real-vector and y Boolean value.
- **Under any model:**

$$\Pr[y=0|x] = \frac{\Pr[x|y=0]\Pr[y=0]}{\Pr[x]} \qquad \Pr[y=1|x] = \frac{\Pr[x|y=1]\Pr[y=1]}{\Pr[x]}$$

- Since $\Pr[y=1|x] + \Pr[y=0|x] = 1$, we have

$$\Pr[y=1|x] = \frac{\Pr[x|y=1]\Pr[y=1]}{\Pr[x|y=0]\Pr[y=0] + \Pr[x|y=1]\Pr[y=1]}$$

$$= \frac{1}{1+\exp\ln\frac{\Pr[x|y=0]\Pr[y=0]}{\Pr[x|y=1]\Pr[y=1]}} = \frac{1}{1+\exp(\ln\frac{\Pr[y=0]}{\Pr[y=1]} + \ln\frac{\Pr[x|y=0]}{\Pr[x|y=1]})}$$

So far, it is true for any model.

Logistic Regression vs Gaussian Naïve Bayes

- Setting $x = (x_1, \dots, x_k)$ real-vector and y Boolean value.

$$\Pr[y=1|x] = \frac{1}{1+\exp(\ln\frac{\Pr[y=0]}{\Pr[y=1]} + \ln\frac{\Pr[x|y=0]}{\Pr[x|y=1]})}$$

Under Naïve Bayes: $\Pr[x|y] = \prod\Pr[x_i|y]$, so we have

$$= \frac{1}{1+\exp(\ln\frac{\Pr[y=0]}{\Pr[y=1]} + \sum\ln\frac{\Pr[x_i|y=0]}{\Pr[x_i|y=1]})}$$

Under Gaussian Naïve Bayes with variance independent of class:

$$\Pr[x_i|y=1] = \frac{1}{\sigma_i\sqrt{2\pi}}e^{\frac{-(x_i-\mu_{i,1})^2}{2\sigma_i^2}}$$

$$\Pr[x_i|y=0] = \frac{1}{\sigma_i\sqrt{2\pi}}e^{\frac{-(x_i-\mu_{i,0})^2}{2\sigma_i^2}}$$

$$\sum\ln\frac{\Pr[x_i|y=0]}{\Pr[x_i|y=1]} = \sum(\frac{\mu_{i,0}-\mu_{i,1}}{\sigma_i^2}x_i + \frac{\mu_{i,0}^2-\mu_{i,1}^2}{2\sigma_i^2})$$

Logistic Regression vs Gaussian Naïve Bayes

- Setting $x = (x_1, \dots, x_k)$ real-vector and y Boolean value.

$$\Pr[y=1|x] = \frac{1}{1+\exp(\ln\frac{\Pr[y=0]}{1-\Pr[y=0]} + \sum(\frac{\mu_{i,0}-\mu_{i,1}}{\sigma_i^2}x_i + \frac{\mu_{i,0}^2-\mu_{i,1}^2}{2\sigma_i^2}))}$$

Take $\omega_0 = \ln\frac{\Pr[y=0]}{1-\Pr[y=0]} + \sum\frac{\mu_{i,0}^2-\mu_{i,1}^2}{2\sigma_i^2}$ and $\omega_i = \frac{\mu_{i,0}-\mu_{i,1}}{\sigma_i^2}$

$$\Pr[y=1|x] = \frac{1}{1+\exp(\omega_o + \sum\omega_i x_i)}$$

The same form as logistic regression!

Logistic Regression vs Gaussian Naïve Bayes

$\text{argmax}_y \Pr[y|x]$: the most probable classification of x

Naïve Bayes (NB)	**Logistic Regression (LR)**
• Use $\Pr[y|x] = \frac{\Pr[x|y]\Pr[y]}{\Pr[x]}$ • Assume $\Pr[x|y] = \prod \Pr[x_i|y]$ • Choose a representation of $\Pr[x_i|y]$ and $\Pr[y]$ • Learn $\Pr[x_i|y]$ and $\Pr[y]$ from data.	• Assume a representation of $\Pr[y|x]$ • Learn $\Pr[y|x]$ from data.

Logistic Regression vs Gaussian Naïve Bayes

$\text{argmax}_y \Pr[y|x]$: the most probable classification of x

Naïve Bayes (NB)	**Logistic Regression (LR)**
• Generative classifier • Can generate new data. We know $\Pr[x_i|y]$ and $\Pr[y]$	• Discriminative classifier • Cannot generate new data. We only know $\Pr[y|x]$

Logistic Regression vs Gaussian Naïve Bayes

$\text{argmax}_y \Pr[y|x]$: the most probable classification of x

Naïve Bayes (NB)	**Logistic Regression (LR)**
• Generally not linear classifier • When it is linear?	• Linear classifier

Gaussian NB with class independent variance is representationally equivalent to LR.

- When training data is infinite and the model is correct, they produce the identical classifier.
- When model is not correct, LR is less biased.

Summary

- **Least Squares Method**
  - The difference between the observed value and the predicted value
  - A measure of the hypothesis
  - Smooth functions
  - Optimization methods
- **Linear Regression**
  - Assume $y = ax + b$
  - y_i are real numbers.
- **Logistic Regression**
  - Assume $\Pr[y|x]$ follows a particular distribution.
  - Linear classifier: y_i are binary.
- **Regularization: penalize large parameters**
  - Hard constraint on objective function & Prior distribution
- **Naïve Bayes vs Logistic Regression**