Machine Learning
Lecture 3

Review of the Previous Lecture

Linear Regression

The learning problem

- Features:
  - Living area, #bedroom, distance to work place …
  - Denote as x = $[x_1, x_2, ..., x_n]^T$
- Target:
  - Price
  - Denoted as y
- Training set:

$$X=\begin{bmatrix} --(x^{(1)})^T-- \\ --(x^{(2)})^T-- \\ \vdots \\ --(x^{(m)})^T-- \end{bmatrix}=\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{bmatrix} \qquad Y=\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

m: #examples/samples
n: #features

Linear Regression

- Assume that Y (target) is a linear function of X (features):

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Here, the θ_i 's are the **parameters** (also called **weights**) parameterizing the space of linear functions mapping from $\mathcal{X}$ to $\mathcal{Y}$ When there is no risk of confusion, we will drop the subscript θ in $h_\theta(x)$, and write it more simply as $h(x)$. To simplify our notation, we also introduce the convention of letting $x_0 = 1$ (this is the **intercept term**), so that

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T \underline{x}$$

$\emptyset(x)$

Pre-processing of features or feature extraction

The Least Mean Square (LMS) method

- The Cost Function:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

- Consider a gradient descent algorithm:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

The normal equations

- To minimize J, we set its derivatives to zero, and obtain the normal equations:

$$X^T X \theta = X^T \vec{y}$$

- Thus, the value of that minimizes J is given in closed form by the equation:

$$\theta = (X^T X)^{-1} X^T \vec{y}$$

Regularized least squares

The total error function:

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\mathbf{w}^\top \mathbf{w}$$

$$\mathbf{w} = (\lambda \mathbf{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\top \mathbf{t}$$

Regularization has the advantage of limiting the model complexity (the appropriate number of basis functions).
This is replaced with the problem of finding a suitable value of the regularization coefficient .

Linear Classification Models

Model Description

- Hypothesis

$$P(y=1|x;\theta) = h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$$

$$P(y=0|x;\theta) = 1 - h_\theta(x)$$

- Compact Form

$$P(y|x;\theta) = (h_\theta(x))^y (1-h_\theta(x))^{1-y}$$

- Parameters θ

Maximum Likelihood Estimation

- (Conditional) Likelihood

$$\begin{aligned} L(\theta) &= p(\vec{y}|X;\theta) \\ &= \prod_{i=1}^{m} p(y^{(i)}|x^{(i)};\theta) \\ &= \prod_{i=1}^{m} (h_\theta(x^{(i)}))^{y^{(i)}} (1-h_\theta(x^{(i)}))^{1-y^{(i)}} \end{aligned}$$

- Log-likelihood

$$\begin{aligned} \iota(\theta) &= \log L(\theta) \\ &= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log(1-h(x^{(i)})) \end{aligned}$$

Cross-Entropy

Gradient Ascent

- Gradient

$$\begin{aligned} \frac{\partial \iota(\theta)}{\partial \theta_j} &= \sum_{i=1}^{m} (y^{(i)} \frac{1}{h_\theta(x^{(i)})} - (1-y^{(i)}) \frac{1}{1-h_\theta(x^{(i)})}) \frac{\partial}{\partial \theta_j} h_\theta(x^{(i)}) \\ &= \sum_{i=1}^{m} (y^{(i)} \frac{1}{h_\theta(x^{(i)})} - (1-y^{(i)}) \frac{1}{1-h_\theta(x(i))}) h_\theta(x^{(i)})(1-h_\theta(x^{(i)})) \frac{\partial}{\partial \theta_j} \theta^T x^{(i)} \\ &= \sum_{i=1}^{m} (y^{(i)}(1-h_\theta(x^{(i)})) - (1-y^{(i)}) h_\theta(x^{(i)})) x_j \\ &= \sum_{i=1}^{m} (y - h_\theta(x^{(i)})) x_j \end{aligned}$$

- Gradient Ascent Method

$$\theta_j := \theta_j + \alpha \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)})) x_j^{(i)}$$

The Newton-Raphson method

- In LR the θ is vector-valued, thus we need the following generalization:

$$\theta := \theta - H^{-1}\nabla_\theta \iota(\theta)$$

Here, $\nabla_\theta \iota(\theta)$ is, as usual, the vector of partial derivatives of $\iota(\theta)$ with respect to the θ_i's; and H is an n-by-n matrix (actually, n + 1-by-n + 1, assuming that we include the intercept term) called the Hessian, whose entries are given by

$$H_{ij} = \frac{\partial^2 \iota(\theta)}{\partial\theta_i \partial\theta_j}$$

Newton's Method for Logistic Regression

- Problem

$$\arg\min_\theta J(\theta) = \frac{1}{m}\sum_{i=1}^{m} -y^{(i)}\log h_\theta(x^{(i)}) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))$$

- Gradient and Hessian Matrix

$$\nabla J(\theta) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j$$

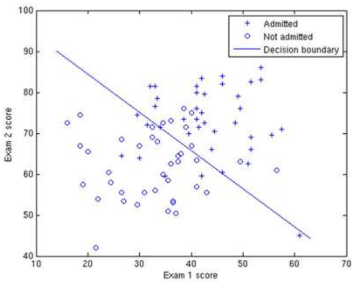
$$H = \frac{1}{m}\sum_{i=1}^{m} h_\theta(x^{(i)})(1-h_\theta(x^{(i)}))x^{(i)}(x^{(i)})^T$$

- Weight updating using Newton's method

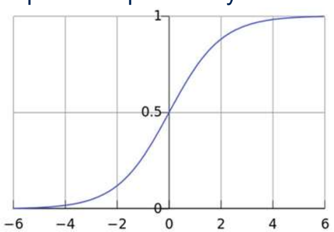
$$\theta^{(t+1)} = \theta^{(t)} - H^{-1}\nabla J(\theta^{(t)})$$

A Linear Classification Model

- Logistic regression has a linear decision boundary (hyperplane)



- But with a nonlinear activation function (Sigmoid function) to model the posterior probability

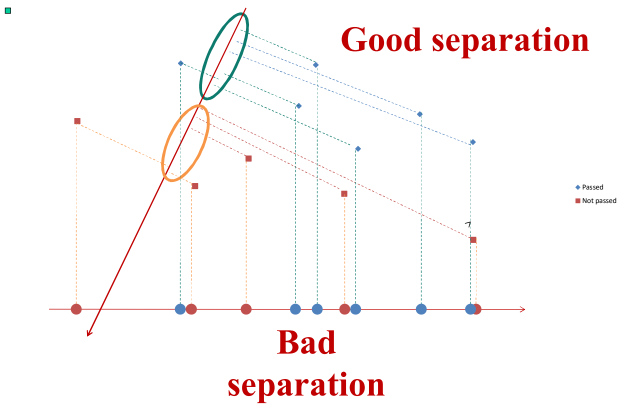


$$h(x) = \frac{1}{1 + exp^{-\theta^T x}}$$

Purpose

- Discriminant Analysis classifies objects in two or more groups according to linear combination of features
- Dimensionality reduction
  - Which set of features can best determine group membership of the object?
- Classification
  - What is the classification rule or model to best separate those groups?

Method (1)



Fisher’s linear discriminant analysis

- Let us now consider Fisher’s LDA projection for dimensionality reduction, considering the two-class case first
- We seek a projection vector a that can be used to compute scalar projections y = $\mathbf{a}^{\mathrm{T}}\mathbf{x}$ for input vectors x
- This vector is obtained by computing the means of each class, μ_1 and μ_2, and then computing two special matrices
- The between-class scatter matrix is calculated as

$$\mathbf{S}_B = (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^{\mathrm{T}}$$

(note the use of the outer product of two vectors here, which gives a matrix)

- The within-class scatter matrix is

$$\mathbf{S}_W = \sum_{i:c_i=1} (\mathbf{x}_i - \boldsymbol{\mu}_1)(\mathbf{x}_i - \boldsymbol{\mu}_1)^{\mathrm{T}} + \sum_{i:c_i=2} (\mathbf{x}_i - \boldsymbol{\mu}_2)(\mathbf{x}_i - \boldsymbol{\mu}_2)^{\mathrm{T}}$$

Fisher’s LDA: the solution vector

- The solution vector a for FLDA is found by maximizing the “Rayleigh quotient”

$$J(\mathbf{a}) = \frac{\mathbf{a}^{\mathrm{T}}\mathbf{S}_B\mathbf{a}}{\boxed{\mathbf{a}^{\mathrm{T}}\mathbf{S}_W\mathbf{a} \text{ + \textbackslash eta ||a||**2}}}$$

- This leads to the solution

$$\mathbf{a} = \mathbf{S}_W^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)$$

(Sw+\eta I)

19

Bayesian Learning

Bayes, Thomas (1763) An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, **53:370-418**

Your first consulting job

- A billionaire asks you a question:
  — He says: I have a coin, if I flip it, what's the probability it will fall with the head up?
  — You say: Please flip it a few times:

  — You say: The probability is : **3/5**
  — **He says: Why???**
  — You say: Because …

Bernoulli Distribution

- Data, D = 

$$D = \{x_i\}_{i=1}^{n}, X_i \epsilon \{H, T\}$$

- P(Heads) = θ, P(Tails) = 1- θ,
- Flips are i.i.d.:
  - Independent events
  - Identically distributed according to Bernoulli distribution

Choose θ that maximizes the probability of observed data

Maximum Likelihood Estimation

- Choose θ that maximizes the probability of observed data

$$\hat{\theta}_{MLE} = \arg\max_{\theta} P(D|\theta)$$

- MLE of probability of head:

$$\hat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T}$$ = **3/5** "Frequency of heads"

Number of heads   Number of tails

Maximum Likelihood Estimation

- Choose θ that maximizes the probability of observed data

$$\hat{\theta}_{MLE} = \arg\max_{\theta} P(D|\theta)$$

$$= \arg\max_{\theta} \prod_{i=1}^{n} P(X_i|\theta)$$ Independent draws

$$= \arg\max_{\theta} \prod_{i:X_i=H} \theta \prod_{i:X_i=T} (1-\theta)$$ Identically distributed

$$= \arg\max_{\theta} \underbrace{\theta^{\alpha_H}(1-\theta)^{\alpha_T}}_{J(\theta)}$$

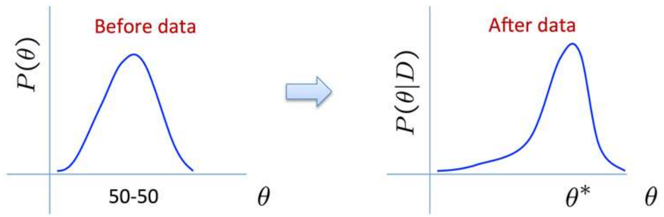
Maximum Likelihood Estimation

- Choose θ that maximizes the probability of observed data

$$\hat{\theta}_{MLE} = \arg\max_{\theta} P(D|\theta)$$
$$= \arg\max_{\theta} \underbrace{\theta^{\alpha_H}(1-\theta)^{\alpha_T}}_{J(\theta)}$$
$$\frac{\partial J(\theta)}{\partial \theta} = \alpha_H \theta^{\alpha_H - 1}(1-\theta)^{\alpha_T} - \alpha_T \theta^{\alpha_H}(1-\theta)^{\alpha_T - 1}$$
$$= (\alpha_H(1-\theta) - \alpha_T\theta)(\theta^{\alpha_H-1}(1-\theta)^{\alpha_T-1})$$
$$\alpha_H(1-\theta) - \alpha_T\theta|_{\theta=\hat{\theta}_{MLE}} = 0$$
$$\boxed{\hat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T}}$$

What about prior knowledge?

- Billionaire says: Wait, I know that the coin is "close" to 50-50. What can you do for me now?
- **You say: I can learn it the Bayesian way…**
- Rather than estimating a single θ, we obtain a distribution over possible values of θ



Prior Distribution

- What about prior?
  — Represents expert knowledge
  — Simple posterior form
- Uninformative priors:
  — Uniform distribution
- Conjugate priors:
  — Closed-form representation of posterior
  — P(θ) and P(θ|D) have the same form

Conjugate Prior

- P(θ) and P(θ|D) have the same form

Eg. 1 Coin flip problem

Likelihood is ~ Binomial

$$P(\mathcal{D} \mid \theta) = \binom{n}{\alpha_H} \theta^{\alpha_H}(1-\theta)^{\alpha_T}$$

If prior is Beta distribution,

$$P(\theta) = \frac{\theta^{\beta_H - 1}(1-\theta)^{\beta_T - 1}}{B(\beta_H, \beta_T)} \sim Beta(\beta_H, \beta_T)$$

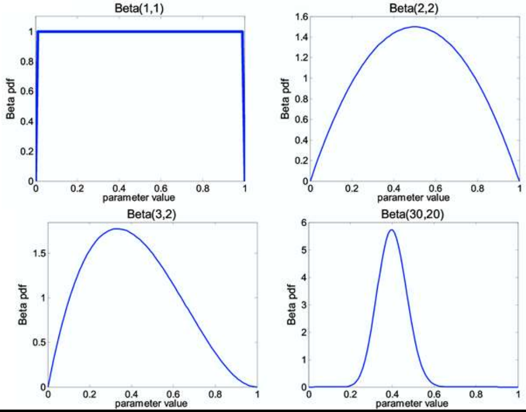
Then posterior is Beta distribution

$$P(\theta|D) \sim Beta(\beta_H + \alpha_H, \beta_T + \alpha_T)$$

For Binomial, conjugate prior is Beta distribution

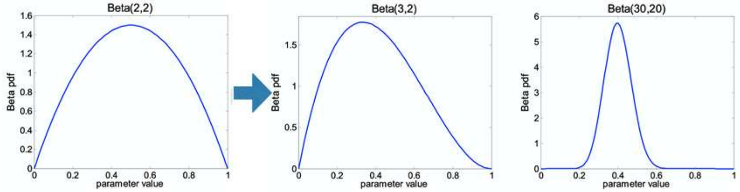
Beta Distribution

$Beta(\beta_H, \beta_T)$ More concentrated as values of β_H, β_T increase



Beta Conjugate Prior

$$P(\theta) \sim Beta(\beta_H, \beta_T) \quad P(\theta|D) \sim Beta(\beta_H + \alpha_H, \beta_T + \alpha_T)$$



As $n = \alpha_H + \alpha_T$ increases

As we get more samples, effect of prior is "washed out"

Conjugate Prior

- P(θ) and P(θ|D) have the same form

Eg. 2 Dice roll problem (6 outcomes instead of 2)

Likelihood is ~ Multinomial(θ = {θ , θ , ... , θ })

$$P(\theta|D) = \frac{n!}{\alpha_1!\alpha_2!...\alpha_k!}\theta_1^{\alpha_1}\theta_2^{\alpha_2}...\theta_k^{\alpha_k} \quad \sum_{i=1}^{k}\alpha_i = n \sum_{i=1}^{k}\theta_i = 1$$

If prior is Dirichlet distribution,

$$P(\theta) = \frac{\prod_{i=1}^{k}\theta_i^{\beta_i - 1}}{B(\beta_1, ..., \beta_k)} \sim Dirichlet(\beta_1, ..., \beta_k)$$

Then posterior is Dirichlet distribution

$$P(\theta|D) \sim Dirichlet(\beta_1 + \alpha_1, ..., \beta_k + \alpha_k)$$

For Multinomial, conjugate prior is Dirichlet distribution

Maximum A Posterior Estimation

- Choose θ that maximizes a posterior probability

$$\hat{\theta}_{MAP} = \arg\max_{\theta} P(\theta|D)$$
$$= \arg\max_{\theta} P(D|\theta)P(\theta)$$

- MAP estimate of probability of head:

$$P(\theta|D) \sim Beta(\beta_H + \alpha_H, \beta_T + \alpha_T)$$

$$\hat{\theta}_{MAP} = \frac{\alpha_H + \beta_H - 1}{\alpha_H + \beta_H + \alpha_T + \beta_T - 2}$$

Mode of Beta distribution

MLE vs. MAP

- Maximum Likelihood estimation (MLE)

  Choose value that maximizes the probability of observed data

$$\hat{\theta}_{MLE} = \arg\max_{\theta} P(D|\theta)$$

- Maximum a posteriori (MAP) estimation

  Choose value that is most probable given observed data and prior belief

$$\hat{\theta}_{MAP} = \arg\max_{\theta} P(\theta|D)$$
$$= \arg\max_{\theta} p(D|\theta)P(\theta)$$

MLE vs. MAP

$$\hat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T}$$

What if we toss the coin too few times?

- You say: Probability next toss is a head = 0
- Billionaire says: You're fired!

$$\hat{\theta}_{MAP} = \frac{\alpha_H + \beta_H - 1}{\alpha_H + \beta_H + \alpha_T + \beta_T - 2}$$

- Beta prior equivalent to extra coin flips
- As *n* → infinity, prior is "forgotten"
- But, for small sample size, prior is important!

Bayesians vs. Frequentists

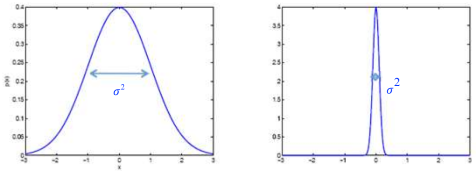
You are no good when sample is small

You give a different answer for different priors

What about continuous variables?

- Billionaire says: If I am measuring a continuous variable, what can you do for me?
- You say: Let me tell you about Gaussians...

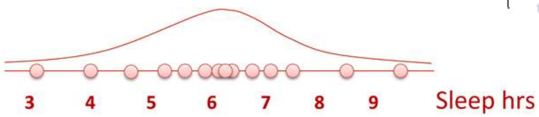
$$P(x|\mu,\alpha) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} = N(\mu, \sigma^2)$$



$\mu = 0$ $\mu = 0$

Gaussian Distribution

Data, D =



- Parameters: $\boldsymbol{\mu}$ – mean, σ^2 - variance
- Sleep hrs are i.i.d.:
  - Independent events
  - Identically distributed according to Gaussian distribution

Properties of Gaussians

- Affine transformation (multiplying by scalar and adding a constant)
  - X ~ $N(\mu,\sigma^2)$
  - Y = aX + b → Y ~ $N(a\mu+b, a^2\sigma^2)$
- Sum of Gaussians
  - X ~ $N(\mu_X,\sigma^2_X)$
  - Y ~ $N(\mu_Y,\sigma^2_Y)$
  - Z = X+Y → Z ~ $N(\mu_X+\mu_Y,\sigma^2_X+\sigma^2_Y)$

MLE for Gaussian mean and variance

- Choose $\theta = (\mu,\sigma^2)$ that maximizes the probability of observed data

$$\hat{\theta}_{MLE} = \arg\max_{\theta} P(D|\theta)$$

$$= \arg\max_{\theta} \prod_{i=1}^{n} P(X_i|\theta) \quad \text{Independent draws}$$

$$= \arg\max_{\theta} \prod_{i=1}^{n} \frac{1}{2\sigma\bullet} e^{-\frac{(X_i-\mu)^2}{2\sigma^2}} \quad \text{Identically distributed}$$

$$= \arg\max_{\theta=(\mu,\sigma^2)} \frac{1}{2\sigma\bullet} e^{-\sum_{i=1}^{n}\frac{(X_i-\mu)^2}{2\sigma^2}}$$

MLE for Gaussian mean and variance

$$\hat{\mu}_{MLE} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\hat{\sigma}^2_{MLE} = \frac{1}{n}\sum_{i=1}^{n}(x_i-\hat{\mu})^2$$

Note: MLE for the variance of a Gaussian is **biased**
- Expected result of estimation is **not** true parameter!
- Unbiased variance estimator:

$$\hat{\sigma}^2_{unbiased} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\hat{\mu})^2$$

MAP for Gaussian mean and variance

- Conjugate priors
  - — Mean: Gaussian prior
  - — Variance: Wishart Distribution

- Prior for mean:

$$P(\mu|\eta,\lambda) = \frac{1}{\lambda\sqrt{2\pi}} e^{\frac{-(\mu-\eta)^2}{2\lambda^2}}$$

Bayes Optimal Classifier & Naive Bayes

Optimal Classifier

Bayes Rule: $P(Y|X) = \dfrac{P(X|Y)P(Y)}{P(X)}$

$$P(Y=y|X=x) = \frac{P(X=x|Y=y)P(Y=y)}{P(X=x)}$$

Bayes classifier:

$$f^*(x) = \arg\max_{Y=y} P(Y=y|X=x)$$

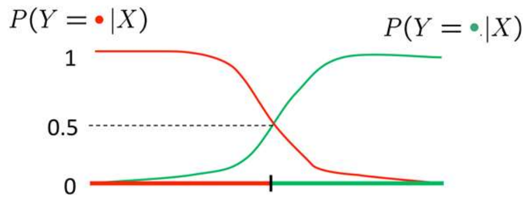
$$= \arg\max_{Y=y} \underbrace{P(X=x|Y=y)}_{\text{Class conditional density}} \underbrace{P(Y=y)}_{\text{Class prior}}$$

Optimal Classification

Optimal predictor: (Bayes classifier) $\boxed{f^* = \arg\min_f P(f(x) \neq Y)}$

Equivalently,

$$f^*(x) = \arg\max_{Y=y} P(Y=y|X=x)$$

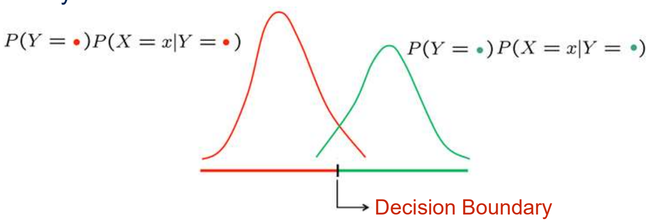


Example Decision Boundaries

- Gaussian class conditional densities (1-dimension/feature)

$$P(X = x|Y = y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp(-\frac{(x-\mu_y)^2}{2\sigma_y^2})$$

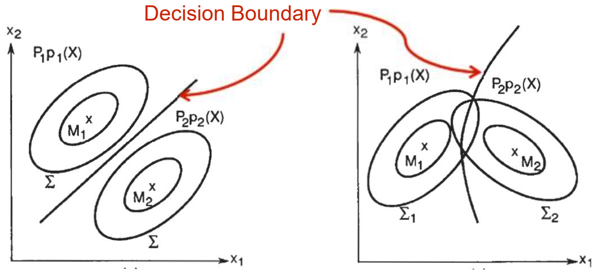
Binary Classification – two classes



Example Decision Boundaries

- Gaussian class conditional densities (2-dimension/feature)

$$P(X = x|Y = y) = \frac{1}{\sqrt{2\pi|\Sigma_y|}} exp(-\frac{(x-\mu_y)\Sigma_y^{-1}(x-\mu_y')}{2})$$



Learning the Optimal Classifier

Optimal classifier:

$$f^*(x) = \arg\max_{Y=y} P(Y = y|X = x)$$

$$= \arg\max_{Y=y} \underbrace{P(X = x|Y = y)}_{\text{Class conditional density}} \underbrace{P(Y = y)}_{\text{Class prior}}$$

Need to know Prior P(Y = y) for all y
Likelihood P(X=x|Y = y) for all x,y

Learning the Optimal Classifier

Task: Predict whether or not a picnic spot is enjoyable

Training Data: $X=(X_1\ \ X_2\ \ X_3\ \ \ldots\ \ \ldots\ \ X_d)$ Y

n rows

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Lets learn P(Y|X) – how many parameters?

Prior: P(Y = y) for all y — K-1 if K labels

Likelihood: P(X=x|Y = y) for all x,y — $(2^d - 1)K$ if d binary features

Learning the Optimal Classifier

Task: Predict whether or not a picnic spot is enjoyable

Training Data: $X=(X_1 \; X_2 \; X_3 \; \ldots \; \ldots \; X_d)$ $\quad Y$

n rows

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Lets learn P(Y|X) – how many parameters?

2^dK - 1 (K classes, d binary features)

Need n >> 2^dK – 1 number of training data to learn all parameters

Conditional Independence

- X is **conditionally independent** of Y given Z: probability distribution governing X is independent of the value of Y, given the value of Z

$$(\forall x, y, z) P(X = x | Y = y, Z = z) = P(X = x | Z = z)$$

- Equivalent to:

$$P(X, Y | Z) = P(X | Z) P(Y | Z)$$

- e.g. $P(Thunder | Rain, Lightning) = P(Thunder | Lightning)$

Note: does NOT mean Thunder is independent of Rain

Prediction using Conditional Independence

- Predict Lightening
- From two **conditionally independent** features
  - Thunder
  - Rain

parameters needed to learn likelihood given L
P(T, R | L) $\quad (2^2-1)2 = 6$

With conditional independence assumption
P(T, R | L) = P(T | L) P(R | L) $\quad (2-1)2 + (2-1)2 = 4$

Naïve Bayes Assumption

- Naïve Bayes assumption:
  - Features are conditionally independent given class:

$$P(X_1, X_2 | Y) = P(X_1 | X_2, Y) P(X_2 | Y)$$
$$= P(X_1 | Y) P(X_2 | Y)$$

  - More generally:

$$P(X_1 \ldots X_d | Y) = \prod_{i=1}^{d} P(X_i | Y)$$

- How many parameters now? **(2-1)dK vs. (2^d-1)K**
  - Suppose **X** is composed of *d* binary features

Naïve Bayes Classifier

- Given:
  - Class Prior P(Y)
  - d conditionally independent features **X** given the class **Y**
  - For each X_i, we have likelihood $P(X_i, Y)$
- Decision rule:

$$f_{NB}(x) = \arg\max_y P(x_1, ..., x_d|y)P(y)$$
$$= \arg\max_y \prod_{i=1}^{d} P(x_i|y)P(y)$$

- If conditional independence assumption holds, NB is optimal classifier! But worse otherwise.

Naïve Bayes Algo – Discrete features

- Training Data $\{(X^{(j)}, Y^{(j)})\}_{j=1}^{n}$ $\quad X^{(j)} = (X_1^{(j)}, ..., X_d^{(j)})$
- Maximum Likelihood Estimates
  - For Class Prior $\hat{P}(y) = \frac{\{\#j : Y^{(j)} = y\}}{n}$
  - For Likelihood $\frac{\hat{P}(x_i, y)}{\hat{P}(y)} = \frac{\{\#j : X_i^{(j)} = x_i, Y^{(i)} = y\}/n}{\{\#j : Y^{(j)} = y\}/n}$
- NB Prediction for test data $X = (x_1, ..., x_d)$

$$Y = \arg\max_y \hat{P}(y) \prod_{i=1}^{d} \frac{\hat{P}(x_i, y)}{\hat{P}(y)}$$

Subtlety 1 – Violation of NB Assumption

- Usually, features are not conditionally independent:

$$P(X_1, ..., X_d|Y) \neq \prod_i P(X_i|Y)$$

- Nonetheless, NB is the single most used classifier out there
  - NB often performs well, even when assumption is violated
  - [Domingos & Pazzani' 96] discuss some conditions for good performance

Subtlety 2 – Insufficient training data

- What if you never see a training instance where X_1=a when Y=b?
  - e.g., Y={SpamEmail}, X_1={'Earn'}
  - $P(X_1$=a | Y=b) = 0
- Thus, no matter what the values $X_2,...,X_d$ take:
  - $P(Y$=b | X_1=a,$X_2,...,X_d)$ = 0

$$\boxed{P(X_1 = a, X_2, ..., X_n|Y) = P(X_1 = a|Y) \prod_{i=2}^{d} P(X_i|Y)}$$

- What now???

MLE vs. MAP

$$\hat{\theta}_{MLE} = \frac{\alpha_H}{\alpha_H + \alpha_T}$$

What if we toss the coin too few times?

- You say: Probability next toss is a head = 0
- Billionaire says: You're fired!

$$\hat{\theta}_{MAP} = \frac{\alpha_H + \beta_H - 1}{\alpha_H + \beta_H + \alpha_T + \beta_T - 2}$$

- Beta prior equivalent to extra coin flips
- As $n \rightarrow$ infinity, prior is "forgotten"
- But, for small sample size, prior is important!

Naïve Bayes Algo – Discrete features

- Training Data $\{(X^{(j)}, Y^{(j)})\}_{j=1}^{n}$ $\quad X^{(j)} = (X_1^{(j)}, ..., X_d^{(j)})$
- Maximum A Posteriori Estimates – add m "virtual" examples

  Assume priors

  $$Q(Y = b) \qquad Q(X_i = a, Y = b)$$

  MAP estimate

  $$\hat{P}(X_i = a | Y = b) = \frac{\{\#j : X_i^{(j)} = a, Y^{(j)} = b\} + mQ(X_i = a, Y = b)}{\{\#j : Y^{(j)} = b\} + \underbrace{mQ(Y = b)}_{\text{\# virtual examples with Y = b}}}$$

  Now, even if you never observe a class/feature posterior probability never zero.

Case Study: Text Classification

- Classify e-mails
  - Y = {Spam,NotSpam}
- Classify news articles
  - Y = {what is the topic of the article?}
- Classify webpages
  - Y = {Student, professor, project, ...}
- What about the features **X**?
  - The Text!

Features X are entire document – X_i for i^{th} word in article

Article from rec.sport.hockey

```
Path: cantaloupe.srv.cs.cmu.edu!das-news.harvard.e
From: xxx@yyy.zzz.edu (John Doe)
Subject: Re: This year's biggest and worst (opinic
Date: 5 Apr 93 09:53:39 GMT

I can only comment on the Kings, but the most
obvious candidate for pleasant surprise is Alex
Zhitnik. He came highly touted as a defensive
defenseman, but he's clearly much more than that.
Great skater and hard shot (though wish he were
more accurate). In fact, he pretty much allowed
the Kings to trade away that huge defensive
liability Paul Coffey. Kelly Hrudey is only the
biggest disappointment if you thought he was any
good to begin with. But, at best, he's only a
mediocre goaltender. A better choice would be
Tomas Sandstrom, though not through any fault of
his own, but because some thugs in Toronto decided
```

NB for Text Classification

- P(X | Y) is huge!!!
  - Article at least 1000 words, **X**={X_1,…,X_{1000}}
  - X_i represents i^{th} word in document, i.e., the domain of X_i is entire vocabulary, e.g., Webster Dictionary (or more), 10,000 words, etc.
- NB assumption helps a lot!!!
  - P(X_i=x_i|Y=y) is just the probability of observing word x_i at the i^{th} position in a document on topic y

$$h_{NB}(x) = \arg\max_y P(y) \prod_{i=1}^{LengthDoc} P(x_i|y)$$

Bag of words model

- Typical additional assumption – **Position in document doesn't matter**: P(X_i=x_i|Y=y) = P(X_k=x_i | Y=y)
  - "Bag of words" model – order of words on the page ignored
  - Sounds really silly, but oren works very well!

$$\prod_{i=1}^{LengthDoc} P(x_i|y) = \prod_{w=1}^{W} P(w|y)^{count_w}$$

When the lecture is over, remember to wake up the person sitting next to you in the lecture room.

Bag of words model

- Typical additional assumption – **Position in document doesn't matter**: P(X_i=x_i|Y=y) = P(X_k=x_i | Y=y)
  - "Bag of words" model – order of words on the page ignored
  - Sounds really silly, but oren works very well!

$$\prod_{i=1}^{LengthDoc} P(x_i|y) = \prod_{w=1}^{W} P(w|y)^{count_w}$$

In is lecture lecture next over person remember room sitting the the the to to up wake when you

Bag of words approach

aardvark	0
about	2
all	2
Africa	1
apple	0
anxious	0
…	
gas	1
…	
oil	1
…	
Zaire	0

NB with Bag of Words for text classification

- **Learning phase**: using multiple training documents
  - — Class Prior P(Y)
  - — $P(X_i|Y)$
- **Test phase:**
  - — For each test document, use naïve Bayes decision rule:

$$h_{NB}(x) = \arg\max_y P(y) \prod_{i=1}^{LengthDoc} P(x_i|y)$$
$$= \arg\max_y P(y) \prod_{w=1}^{W} P(w|y)^{count_w}$$

Twenty news groups results

Given 1000 training documents from each group
Learn to classify new documents according to
which newsgroup it came from

comp.graphics	misc.forsale
comp.os.ms-windows.misc	rec.autos
comp.sys.ibm.pc.hardware	rec.motorcycles
comp.sys.mac.hardware	rec.sport.baseball
comp.windows.x	rec.sport.hockey
alt.atheism	sci.space
soc.religion.christian	sci.crypt
talk.religion.misc	sci.electronics
talk.politics.mideast	sci.med
talk.politics.misc	
talk.politics.guns	

Naive Bayes: 89% classification accuracy

What if features are continuous?

e.g., character recognition: X_i is intensity at i^{th} pixel

Gaussian Naïve Bayes (GNB):

$$P(X_i = x | Y = y_k) = \frac{1}{\sigma_{ik}\sqrt{2\pi}} e^{\frac{-(x-\mu_{ik})^2}{2\sigma_{ik}^2}}$$

Different mean and variance for each class k and each pixel i.

Sometimes assume variance
- is independent of Y (i.e., σ_i)
- or independent of X_i (i.e., σ_k)
- or both (i.e., σ)

Estimating parameters: Y discrete, X_i continuous

Maximum likelihood estimates: $\hat{\mu}_{MLE} = \frac{1}{N}\sum_{j=1}^{N} x_j$

$$\hat{\mu}_{ik} = \frac{1}{\Sigma_j \delta(Y^j = y_k)} \Sigma_j x_i^j \delta(Y^j = y_k)$$

i^{th} pixel in j^{th} training image

k^{th} class

j^{th} training image

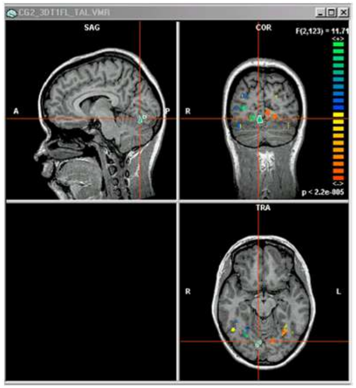
$$\boxed{\hat{\sigma}^2_{unbiased} = \frac{1}{N-1}\sum_{j=1}^{N}(x_j - \hat{\mu})^2}$$

$$\hat{\sigma}^2_{ik} = \frac{1}{\Sigma_j \delta(Y^j = y_k) - 1} \sum_j (x_i^j - \hat{\mu}_{ik})^2 \delta(Y^j = y_k)$$

Example: GNB for classifying mental States

~1 mm resolution
~2 images per sec
15,000 voxels/image
non-invasive, safe
measures Blood Oxygen Level Dependent (BOLD) response

[Mitchell et al.]



Gaussian Naive Bayes

Consider a GNB based on the following modeling assumptions:

- Y is boolean, governed by a Bernoulli distribution, with parameter $\pi = P(Y = 1)$
- $X = \langle X_1 \ldots X_n \rangle$, where each X_i is a continuous random variable
- For each X_i, $P(X_i|Y = y_k)$ is a Gaussian distribution of the form $N(\mu_{ik};\sigma_i)$
- For all i and $j \neq i$, X_i and X_j are conditionally independent given Y

Gaussian Naive Bayes

- Note here we are assuming the standard deviations σ_i vary from attribute to attribute, but do not depend on Y.
- We now derive the parametric form of P(Y|X) that follows from this set of GNB assumptions. In general, Bayes rule allows us to write

$$P(Y=1|X) = \frac{P(Y=1)P(X|Y=1)}{P(Y=1)P(X|Y=1)+P(Y=0)P(X|Y=0)}$$

Gaussian Naive Bayes (cont.)

- Dividing both the numerator and denominator by the numerator yields:

$$P(Y=1|X) = \frac{1}{1+\frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)}}$$

- or equivalently:

$$P(Y=1|X) = \frac{1}{1+exp(\ln\frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)})}$$

Gaussian Naive Bayes (cont.)

- Because of our conditional independence assumption we can write this

$$P(Y=1|X) = \frac{1}{1+exp(\ln\frac{P(Y=0)}{P(Y=1)} + \sum_i \ln\frac{P(X_i|Y=0)}{P(X_i|Y=1)})}$$
$$= \frac{1}{1+exp(\ln\frac{1-\pi}{pi} + \sum_i \ln\frac{P(X_i|Y=0)}{P(X_i|Y=1)})}$$
$$= \frac{1}{1+exp(\ln\frac{1-\pi}{\pi} + \sum_i(\frac{\mu_{i0}-\mu_{i1}}{\sigma_i^2}X_i + \frac{\mu_{i1}^2-\mu_{i0}^2}{2\sigma_i^2}))}$$
$$= \frac{1}{1+exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

where the weights $w_1 \ldots w_n$ are given by $w_i = \frac{\mu_{i0}-\mu_{i1}}{\sigma_i^2}$ and $w_0 = \ln\frac{1-\pi}{\pi} + \sum_i \frac{\mu_{i1}^2-\mu_{i0}^2}{2\sigma_i^2}$

also we have $P(Y=0|X) = 1 - P(Y=1|X) = \frac{exp(w_0+\sum_{i=1}^n w_i X_i)}{1+exp(w_0+\sum_{i=1}^n w_i X_i)}$

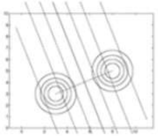
The decision boundary

- The predictive distribution:

$$p(y_n^1 = 1|x_n) = \frac{1}{1+exp\{-\sum_{j=1}^M \theta_j x_n^j - \theta_0\}} = \frac{1}{1+e^{-\theta^T x_n}}$$

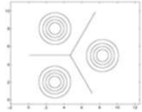
- The Bayes decision rule:

$$\ln\frac{p(y_n^1=1|x_n)}{p(y_n^2=1|x_n)} = \ln\left(\frac{1}{1+e^{-\theta^T x_n}} \Big/ \frac{e^{-\theta^T x_n}}{1+e^{-\theta^T x_n}}\right) = \theta^T x_n$$



- For multiple class (i.e., *K*>2), * correspond to a softmax function

$$p(y_n^k = 1|x_n) = \frac{e^{-\theta_k^T x_n}}{\sum_j e^{-\theta_j^T x_n}}$$



If assume variance of Xi is independent of Y,

then GNB has linear separating hyperplane.

Recall Logistic Regression

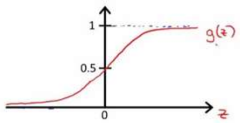
- The conditional distribution: a Bernoulli

$$p(y|x) = \mu(x)^y(1-\mu(x))^{1-y}$$

- Where μ is a logistic function

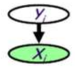
$$\mu(x) = \frac{1}{1+e^{-\theta^T x}} = p(y=1|x)$$

- What is the difference to NB?

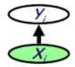


Naïve Bayes vs. Logistic Regression

- Naïve Baye — Generative classifier
  - Assume some functional form for P(X|Y), P(Y)
  - This is a ‘***generative***’ model of the data!
  - Estimate parameters of P(X|Y), P(Y) directly from training data
  - Use Bayes rule to calculate P(Y|X=x)



- Logistic Regression — Discriminative classifier
  - Directly assume some functional form for P(Y|X)
  - This is a ‘***discriminative***’ model of the data!
  - Estimate parameters of P(Y|X) directly from training data



Another Example of Generative Model: Gaussian Discriminant Analysis

- Gaussian discriminant analysis (GDA) is a simple generative learning algorithm
- In this model, p(x|y) is distributed according to a multivariate normal distribution

The multivariate normal distribution in n-dimensions, also called the multivariate Gaussian distribution, is parameterized by a **mean vector** $\mu \in \mathbb{R}^n$ and a **covariance matrix** $\Sigma \in \mathbb{R}^{n\times n}$, where $\Sigma \geq 0$ is symmetric and positive semi-definite. Also written "$\mathcal{N}(\mu, \Sigma)$", its density is given by:

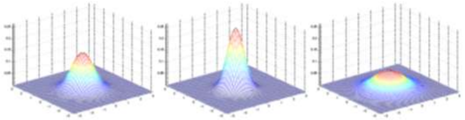
$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$

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

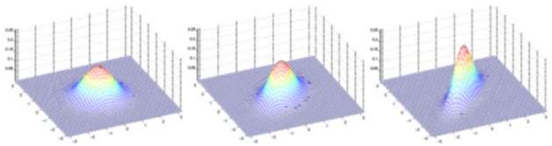
$$\mathrm{Cov}(X) = \Sigma.$$

Some Examples

- Here're some examples of what the density of a Gaussian distribution looks like:

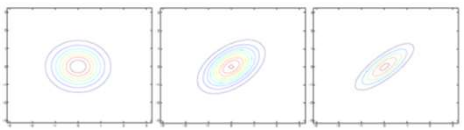


mean = $\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma = I$ mean = $\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma = 0.6I$ mean = $\begin{bmatrix}0\\0\end{bmatrix}$, $\Sigma = 2I$



The figures above show Gaussians with mean 0, and with covariance matrices respectively

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



The Gaussian Discriminant Analysis Model

- When we have a classification problem in which the input features x are continuous-valued random variables, we can then use the Gaussian Discriminant Analysis (GDA) model, which models p(x|y) using a multivariate normal distribution. The model is:

$$\begin{aligned} y &\sim \mathrm{Bernoulli}(\phi) \\ x|y=0 &\sim \mathcal{N}(\mu_0, \Sigma) \\ x|y=1 &\sim \mathcal{N}(\mu_1, \Sigma) \end{aligned}$$

Writing out the distributions, this is:

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

The Gaussian Discriminant Analysis model

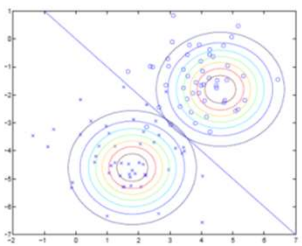
- Here, the parameters of our model are φ, , μ_0 and μ_1. (Note that while there're two different mean vectors μ_0 and μ_1, this model is usually applied using only one covariance matrix .) The log-likelihood of the data is given by

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

The Gaussian Discriminant Analysis model

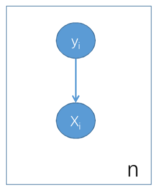
- By maximizing ℓ with respect to the parameters, we find the maximum likelihood estimate of the parameters to be:

$$
\begin{aligned}
\phi &= \frac{1}{m} \sum_{i=1}^{m} 1\{y^{(i)} = 1\} \\
\mu_0 &= \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 0\} x^{(i)}}{\sum_{i=1}^{m} 1\{y^{(i)} = 0\}} \\
\mu_1 &= \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} x^{(i)}}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\}} \\
\Sigma &= \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T..
\end{aligned}
$$

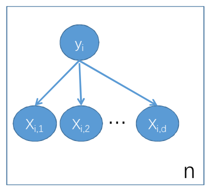


- Note that the two Gaussians have contours that are the same shape and orientation, since they share a covariance matrix Σ, but they have different means μ_0 and μ_1.
- Also shown in the figure is the straight line giving the decision boundary at which $p(y = 1|x) = 0.5$. On one side of the boundary, we'll predict $y = 1$ to be the most likely outcome, and on the other side, we'll predict $y = 0$.

GNB vs. GDA



GDA



GNB

Discriminative Model vs. Generative Model

Hypothesis - Learning - Decision

- Discriminative Model
  - Directly Modeling Predictive Function

$$y = f(x)$$

Examples: Perceptron, SVMs

  - Modeling Conditional Distribution

$$p(y|x)$$

Examples: Logistic Regression

- Generative Model (Modeling Joint Distribution)

$$p(x,y) = p(y)p(x|y)$$

Examples: Naïve Bayes, GDA

Hypothesis - Learning - Decision

- Discriminative Model
  - Modeling Predictive Function

$$\theta^* = \arg\max_\theta J(\theta)$$

Optimizing some loss functions, such as least mean square (LMS), cross entropy (CE), Maximum Margin, etc.

  - Modeling Conditional Distribution

$$\theta^* = \arg\max_\theta \sum_i \log p(y^{(i)}|x^{(i)})$$

MLE, MAP (for conditional distribution)

- Generative Model (Modeling Joint Distribution)

$$\theta^* = \arg\max_\theta \sum_i \log p(x^{(i)}, y^{(i)})$$

MLE, MAP, Bayesian Inference (for conditional distribution)

Hypothesis - Learning - Decision

- Discriminative Model
  - Conditional Distribution

$$\arg\max_y p(y|x)$$

  - Predictive Function

$$y = f(x)$$

- Generative Model
  - Bayes Formula

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

$$\Downarrow$$

$$\arg\max_y p(y|x) = \arg\max_y p(x,y) = \arg\max_y p(x|y)p(y)$$

What You Should Know

- We can use Bayes rule as the basis for designing learning algorithms by using the training data to learn estimates of P(X|Y) and P(Y). This type of classifier is called a generative classifier, because we can view the distribution P(X|Y) as describing how to generate random instances X conditioned on the target attribute Y.
- Learning Bayes classifiers typically requires an unrealistic number of training instances. The Naive Bayes classifier assumes all attributes describing X are conditionally independent given Y. This assumption dramatically reduces the number of parameters that must be estimated to learn the classifier.
- When X is a vector of discrete-valued attributes, Naive Bayes learning algorithms can be viewed as linear classifiers. The same statement holds for Gaussian Naive Bayes classifiers if the variance of each feature is assumed to be independent of the class

What You Should Know (cont.)

- Logistic Regression is a function approximation algorithm that uses training data to directly estimate P(Y|X), in contrast to Naive Bayes. In this sense, Logistic Regression is often referred to as a discriminative classifier because we can view the distribution P(Y|X) as directly discriminating the value of the target value Y for any given instance X.
- Logistic Regression is a linear classifier over X. The linear classifiers produced by Logistic Regression and Gaussian Naive Bayes are identical in the limit as the number of training examples approaches infinity, provided the Naive Bayes assumptions hold. However, if these assumptions do not hold, the Naive Bayes bias will cause it to perform less accurately than Logistic Regression, in the limit.