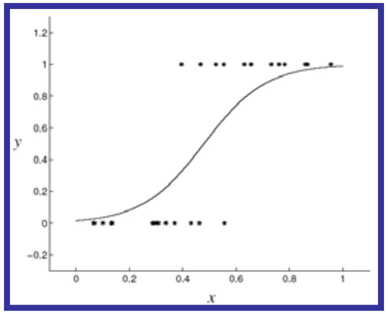
Machine Learning

Generative verses discriminative classifier

Eric Xing

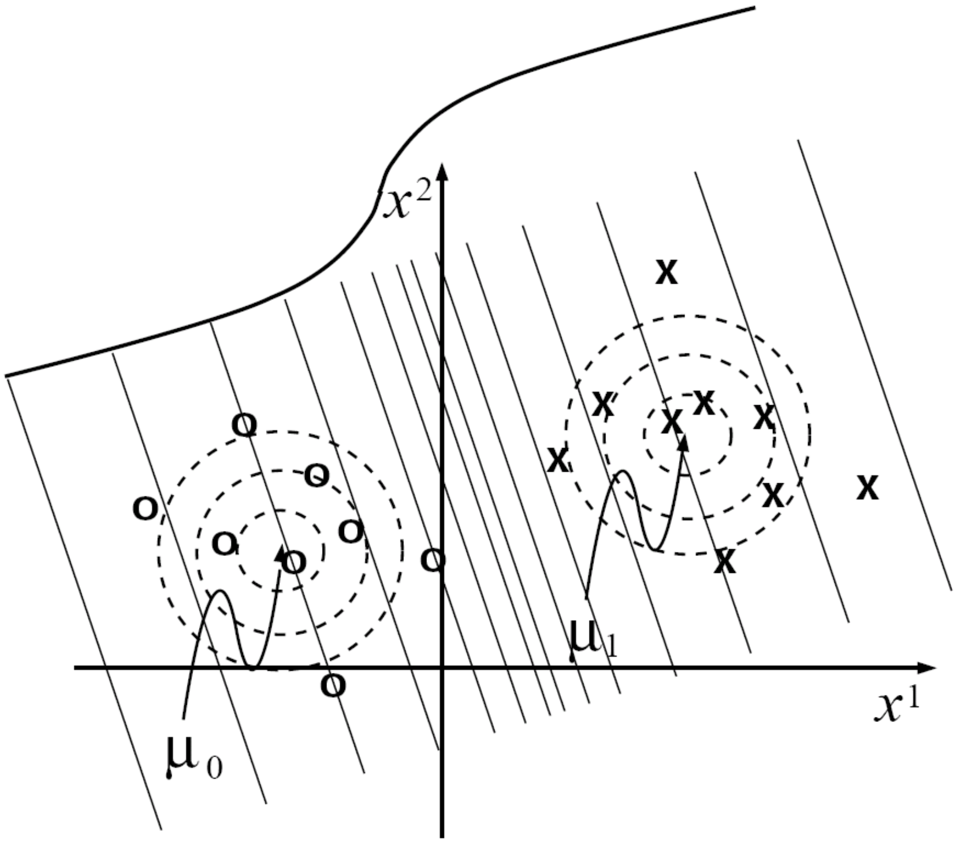
Lecture 2, August 12, 2010



Reading:

Generative and Discriminative classifiers

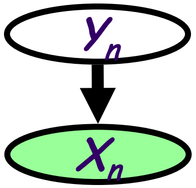
- Goal: Wish to learn f: X $\rightarrow$ Y, e.g., P(Y|X)
- Generative:
  - Modeling the joint distribution of all data
- Discriminative:
  - Modeling only points at the boundary



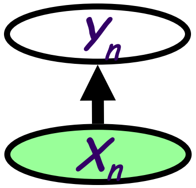
Generative vs. Discriminative Classifiers

- Goal: Wish to learn f: X $\rightarrow$ Y, e.g., P(Y|X)

- Generative classifiers (e.g., Naïve Bayes):
  - 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)

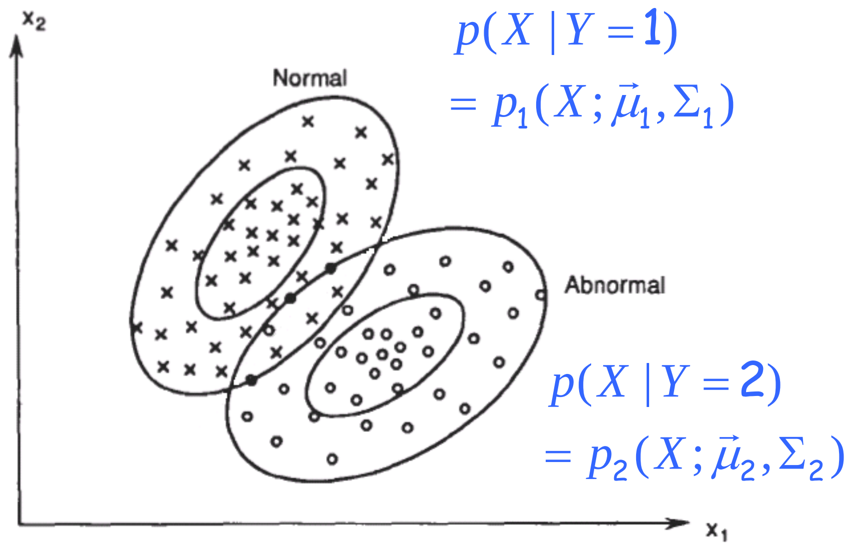


- Discriminative classifiers (e.g., logistic regression)
  - Directly assume some functional form for P(Y|X)
    This is a ***'discriminative'*** model of the data!
  - Estimate parameters of P(Y|X) directly from training data



Suppose you know the following ...

- Class-specific Dist.: P(X|Y)



Bayes classifier:

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

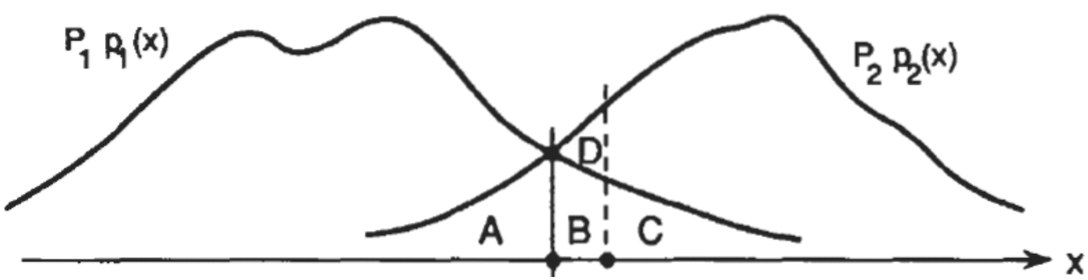
- Class prior (i.e., "weight"): P(Y)

- This is a generative model of the data!

Optimal classification

- **Theorem:** Bayes classifier is optimal!
  - That is

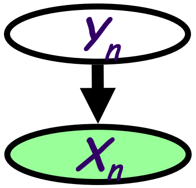
$$error_{true}(h_{Bayes})) \leq error_{true}(h), \ \forall h(\mathbf{x})$$



- How to learn a Bayes classifier?
  - Recall density estimation. We need to estimate P(X|y=k), and P(y=k) for all k

Gaussian Discriminative Analysis

- learning f: X $\rightarrow$ Y, where
  - X is a vector of real-valued features, $\mathbf{X}_n = < X_{n,1} \ldots X_{n,\mathrm{m}} >$
  - Y is an indicator vector
- What does that imply about the form of P(Y|X)?
  - The joint probability of a datum and its label is:



$$p(\mathbf{x}_n, y_n^k = 1 \mid \mu, \sigma) = p(y_n^k = 1) \times p(\mathbf{x}_n \mid y_n^k = 1, \mu, \sigma)$$
$$= \pi_k \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\tfrac{1}{2\sigma^2}(\mathbf{x}_n - \mu_k)^2\right\}$$

  - Given a datum $\mathbf{x}_n$, we predict its label using the conditional probability of the label given the datum:

$$p(y_n^k = 1 \mid \mathbf{x}_n, \mu, \sigma) = \frac{\pi_k \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\tfrac{1}{2\sigma^2}(\mathbf{x}_n - \mu_k)^2\right\}}{\sum_{k'} \pi_{k'} \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\tfrac{1}{2\sigma^2}(\mathbf{x}_n - \mu_{k'})^2\right\}}$$

Conditional Independence

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

$$(\forall i, j, k) P(X = i | Y = j, Z = k) = P(X = i | Z = k)$$

Which we often write

$$P(X \mid Y, Z) = P(X \mid Z)$$

- e.g.,

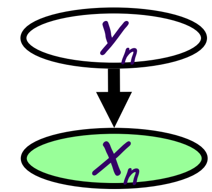
$$P(Thunder | Rain, Lightning) = P(Thunder | Lightning)$$

- Equivalent to:

$$P(X, Y \mid Z) = P(X \mid Z) P(Y \mid Z)$$

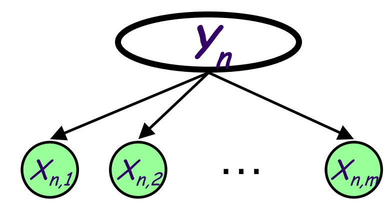
Naïve Bayes Classifier

- When X is multivariate-Gaussian vector:
  - The joint probability of a datum and it label is:



$$p(\mathbf{x}_n, y_n^k = 1 \mid \vec{\mu}, \Sigma) = p(y_n^k = 1) \times p(\mathbf{x}_n \mid y_n^k = 1, \vec{\mu}, \Sigma)$$
$$= \pi_k \frac{1}{(2\pi|\Sigma|)^{1/2}} \exp\left\{-\tfrac{1}{2}(\mathbf{x}_n - \vec{\mu}_k)^T \Sigma^{-1} (\mathbf{x}_n - \vec{\mu}_k)\right\}$$

  - The naïve Bayes simplification



$$p(\mathbf{x}_n, y_n^k = 1 \mid \mu, \sigma) = p(y_n^k = 1) \times \prod_j p(x_{n,j} \mid y_n^k = 1, \mu_{k,j}, \sigma_{k,j})$$
$$= \pi_k \prod_j \frac{1}{(2\pi\sigma_{k,j}^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma_{k,j}^2}(x_{n,j} - \mu_{k,j})^2\right\}$$

  - More generally: $p(\mathbf{x}_n, y_n \mid \eta, \pi) = p(y_n \mid \pi) \times \prod_{j=1}^{m} p(x_{n,j} \mid y_n, \eta)$
    - Where $p(. \mid .)$ is an arbitrary conditional (discrete or continuous) 1-D density

The predictive distribution

- Understanding the predictive distribution

$$p(y_n^k = 1 \mid x_n, \vec{\mu}, \Sigma, \pi) = \frac{p(y_n^k = 1, x_n \mid \vec{\mu}, \Sigma, \pi)}{p(x_n \mid \vec{\mu}, \Sigma)} = \frac{\pi_k N(x_n, \mid \mu_k, \Sigma_k)}{\sum_{k'} \pi_{k'} N(x_n, \mid \mu_{k'}, \Sigma_{k'})} \quad *$$

- Under naïve Bayes assumption:

$$p(y_n^k = 1 \mid x_n, \vec{\mu}, \Sigma, \pi) = \frac{\pi_k \exp\left\{ -\sum_j \left(\frac{1}{2\sigma_{k,j}^2} (x_n^j - \mu_k^j)^2 - \log \sigma_{k,j} - C \right) \right\}}{\sum_{k'} \pi_{k'} \exp\left\{ -\sum_j \left(\frac{1}{2\sigma_{k',j}^2} (x_n^j - \mu_{k'}^j)^2 - \log \sigma_{k',j} - C \right) \right\}} \quad **$$

- For two class (i.e., K=2), and when the two classes haves the same variance, ** turns out to be a logistic function

$$p(y_n^1 = 1 \mid x_n) = \frac{1}{1 + \frac{\pi_2 \exp\left\{ -\sum_j \left(\frac{1}{2\sigma_j^2} (x_n^j - \mu_2^j)^2 - \log \sigma_j - C \right) \right\}}{\pi_1 \exp\left\{ -\sum_j \left(\frac{1}{2\sigma_j^2} (x_n^j - \mu_1^j)^2 - \log \sigma_j - C \right) \right\}}} = \frac{1}{1 + \exp\left\{ -\sum_j \left(x_n^j \frac{1}{\sigma_j^2} (\mu_1^j - \mu_2^j) + \frac{1}{\sigma_j^2} ([\mu_1^j]^2 - [\mu_2^j]^2) \right) + \log \frac{(1-\pi_1)}{\pi_1} \right\}}$$

$$= \frac{1}{1 + e^{-\theta^T x_n}}$$

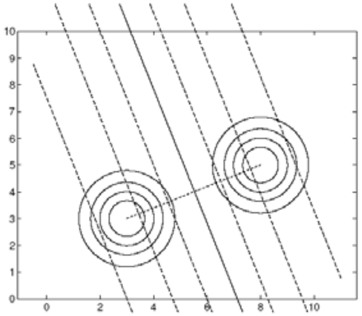
The decision boundary

- The predictive distribution

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

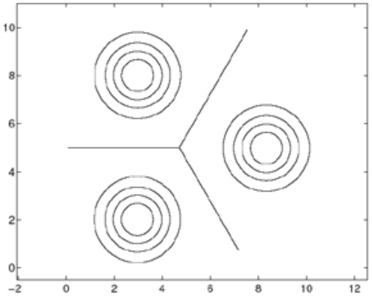
- The Bayes decision rule:

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



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

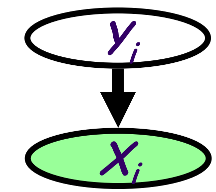
$$p(y_n^k = 1 \mid x_n) = \frac{e^{-\theta_k^T x_n}}{\sum_j e^{-\theta_j^T x_n}}$$



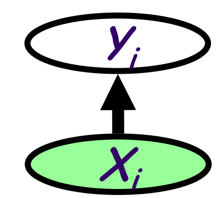
Generative vs. Discriminative Classifiers

- Goal: Wish to learn f: X $\rightarrow$ Y, e.g., P(Y|X)

- Generative classifiers (e.g., Naïve Bayes):
  - 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)



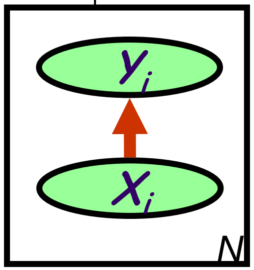
- Discriminative classifiers:
  - Directly assume some functional form for P(Y|X)
    This is a '***discriminative***' model of the data!
  - Estimate parameters of P(Y|X) directly from training data



Linear Regression

- The data:

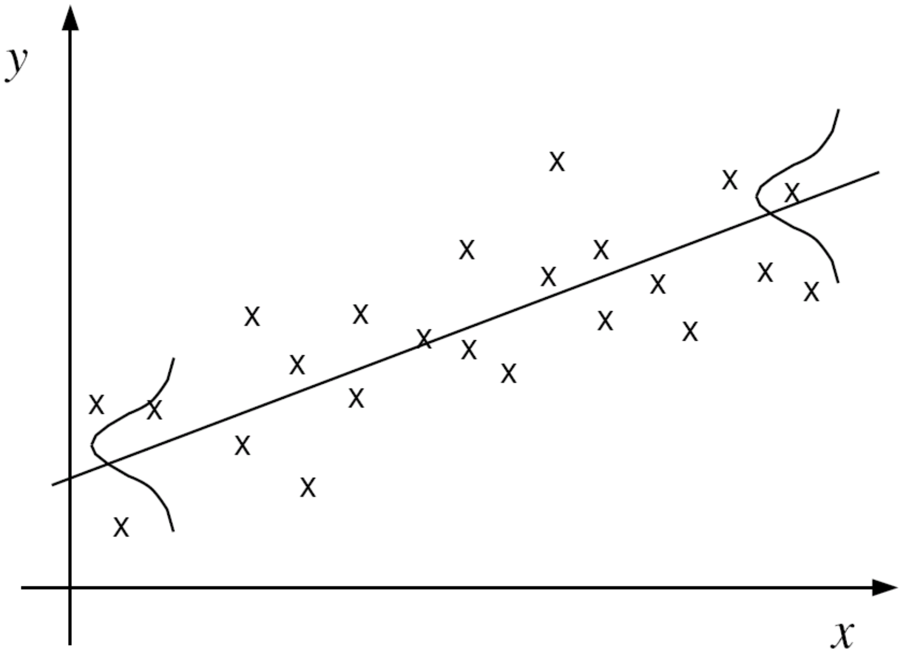
$$\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots, (x_N, y_N)\}$$



- Both nodes are observed:
  - X is an input vector
  - Y is a response vector

    (we first consider y as a generic continuous response vector, then we consider the special case of classification where y is a discrete indicator)

- A regression scheme can be used to model $p(y|x)$ directly, rather than $p(x,y)$



Linear Regression

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

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

  - let's assume a vacuous "feature" X_0=1 (this is the intercept term, why?), and define the feature vector to be:

$$\mathbf{x} = [1, x_1, x_2]$$

  - then we have the following general representation of the linear function:

$$\hat{y} = \mathbf{x}^T \theta$$

- Our goal is to pick the optimal θ . How!
  - We seek θ that minimize the following **cost function**:

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{n} (\hat{y}_i(\vec{x}_i) - y_i)^2$$

The Least-Mean-Square (LMS) method

- Consider a **gradient descent** algorithm:

$$\theta_j^{t+1} = \theta_j^{t} - \alpha \frac{\partial}{\partial \theta_j} J(\theta)\Big|_t$$

- Now we have the following descent rule:

$$\theta_j^{t+1} = \theta_j^{t} + \alpha \sum_{i=1}^{n} (y_i - \vec{\mathbf{x}}_i^T \theta^t) x_i^j$$

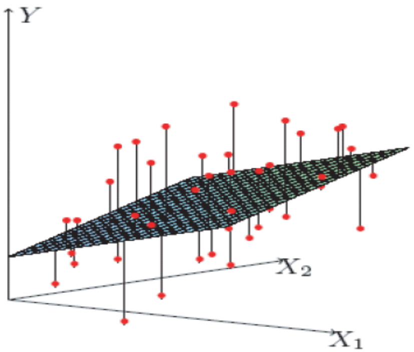
- For a single training point, we have:

$$\theta_j^{t+1} = \theta_j^{t} + \alpha (y_i - \vec{\mathbf{x}}_i^T \theta^t) x_i^j$$

  - This is known as the LMS update rule, or the Widrow-Hoff learning rule
  - This is actually a "**stochastic**", "**coordinate**" descent algorithm
  - This can be used as a **on-line** algorithm

Probabilistic Interpretation of LMS

- Let us assume that the target variable and the inputs are related by the equation:



$$y_i = \theta^T \mathbf{x}_i + \varepsilon_i$$

where ε is an error term of unmodeled effects or random noise

- Now assume that ε follows a Gaussian $N(0,\sigma)$, then we have:

$$p(y_i \mid x_i;\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i - \theta^T \mathbf{x}_i)^2}{2\sigma^2} \right)$$

- By independence assumption:

$$L(\theta) = \prod_{i=1}^{n} p(y_i \mid x_i;\theta) = \left(\frac{1}{\sqrt{2\pi}\sigma} \right)^n \exp\left(-\frac{\sum_{i=1}^{n} (y_i - \theta^T \mathbf{x}_i)^2}{2\sigma^2} \right)$$

Probabilistic Interpretation of LMS, cont.

- Hence the log-likelihood is:

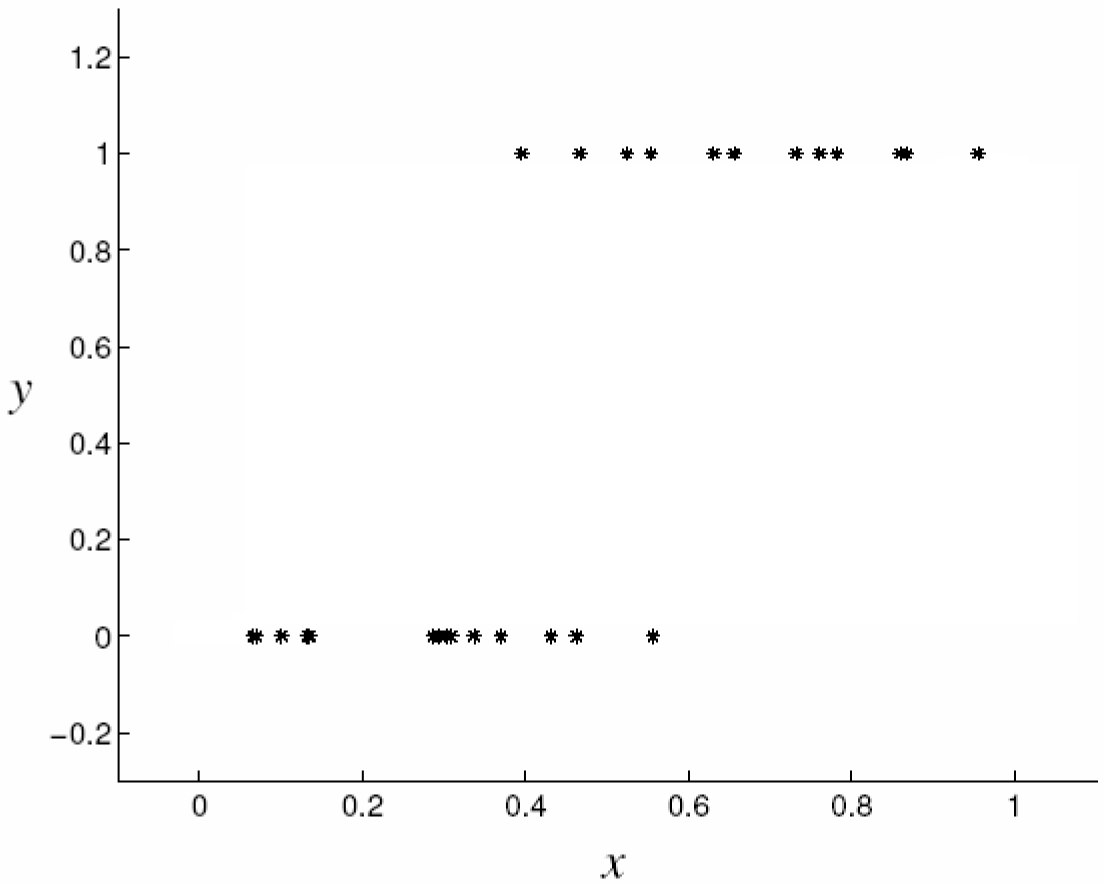
$$l(\theta) = n \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2}\frac{1}{2}\sum_{i=1}^{n}(y_i - \theta^T \mathbf{x}_i)^2$$

- Do you recognize the last term?

Yes it is: $$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}(\mathbf{x}_i^T\theta - y_i)^2$$

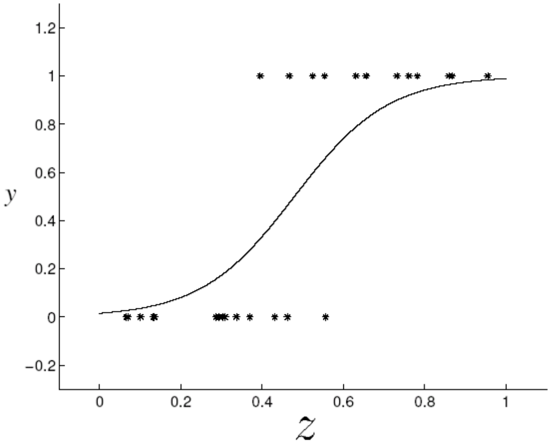
- Thus under independence assumption, LMS is equivalent to MLE of θ !

Classification and logistic regression



The logistic function

$$g(z) = \frac{1}{1 + e^{-z}}$$



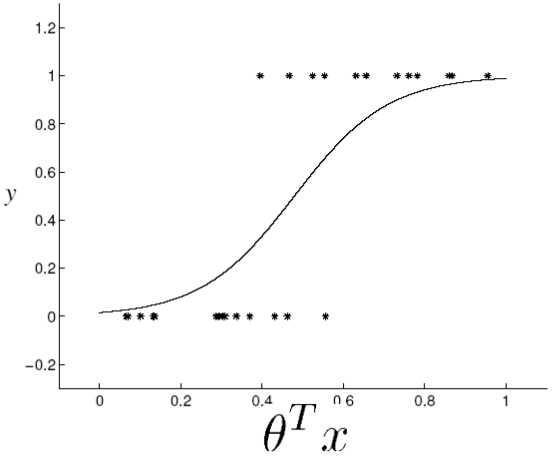
Logistic regression (sigmoid classifier)

- The condition distribution: a Bernoulli

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

where μ is a logistic function

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



- We can used the brute-force gradient method as in LR

- But we can also apply generic laws by observing the $p(y|x)$ is an exponential family function, more specifically, a generalized linear model (see future lectures …)

Training Logistic Regression: MCLE

- Estimate parameters θ=<θ_0, θ_1, ... θ_m> to maximize the **conditional likelihood** of training data

- Training data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$

- Data likelihood = $\prod_{i=1}^{N} P(x_i, y_i; \theta)$

- Data conditional likelihood = $\prod_{i=1}^{N} P(x_i | y_i; \theta)$

$$\theta = \arg\max_{\theta} \ln \prod_{i} P(y_i | x_i; \theta)$$

Expressing Conditional Log Likelihood

$$l(\theta) \equiv \ln \prod_i P(y_i|x_i;\theta) = \sum_i \ln P(y_i|x_i;\theta)$$

- Recall the logistic function: $\mu = \frac{1}{1+e^{-\theta^T x}}$

  and conditional likelihood: $P(y|x) = \mu(x)^y (1-\mu(x))^{1-y}$

$$\begin{aligned} l(\theta) = \sum_i \ln P(y_i|x_i;\theta) &= \sum_i y_i \ln u(x_i) + (1-y_i)\ln(1-\mu(x_i)) \\ &= \sum_i y_i \ln \frac{u(x_i)}{1-\mu(x_i)} + \ln(1-\mu(x_i)) \\ &= \sum_i y_i \theta^T x_i - \theta^T x_i + \ln(1+e^{-\theta^T x_i}) \\ &= \sum_i (y_i - 1)\theta^T x_i + \ln(1+e^{-\theta^T x_i}) \end{aligned}$$

Maximizing Conditional Log Likelihood

- The objective:

$$
\begin{aligned}
l(\theta) &= \ln \prod_i P(y_i|x_i;\theta) \\
&= \sum_i (y_i - 1)\theta^t x_i + \ln(1 + e^{-\theta^T x_i})
\end{aligned}
$$

- Good news: *l(θ)* is concave function of θ

- Bad news: no closed-form solution to maximize *l(θ)*

The Newton's method

- Finding a zero of a function

$$\theta^{t+1} := \theta^t - \frac{f(\theta^t)}{f'(\theta^t)}$$

The Newton's method (con'd)

- To maximize the conditional likelihood $l(\theta)$:

$$l(\theta) = \sum_i (y_i - 1)\theta^T x_i + \ln(1 + e^{-\theta^T x_i})$$

since l is convex, we need to find θ^* where $l'(\theta^*)=0$!

- So we can perform the following iteration:

$$\theta^{t+1} := \theta^t + \frac{l'(\theta^t)}{l''(\theta^t)}$$

The Newton-Raphson method

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

$$\theta^{t+1} := \theta^t + H^{-1}\nabla_{\theta^t} l(\theta^t)$$

- ∇ is the gradient operator over the function

- H is known as the Hessian of the function

The Newton-Raphson method

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

$$\theta^{t+1} := \theta^t + H^{-1}\nabla_{\theta^t} l(\theta^t)$$

- ∇ is the gradient operator over the function

$$\nabla_\theta l(\theta) = \sum_i (y_i - u_i) x_i = \mathbf{X}^T(\mathbf{y} - \mathbf{u})$$

- H is known as the Hessian of the function

$$H = \nabla_\theta \nabla_\theta l(\theta) = \sum_i u_i(1-u_i) x_i x_i^T = \mathbf{X}^T\mathbf{R}\mathbf{X}$$

$$\text{where} \quad R_{ii} = u_i(1-u_i)$$

Iterative reweighed least squares (IRLS)

- Recall in the least square est. in linear regression, we have:

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

which can also derived from Newton-Raphson

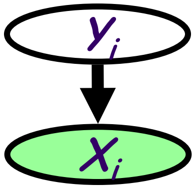
- Now for logistic regression:

$$\begin{aligned}\theta^{t+1} &= \theta^t + H^{-1}\nabla_{\theta^t} l(\theta^t) \\ &= \theta^t - (\mathbf{X}^T\mathbf{R}\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{u}-\mathbf{y}) \\ &= (\mathbf{X}^T\mathbf{R}\mathbf{X})^{-1}\{\mathbf{X}^T\mathbf{R}\mathbf{X}\theta^t - \mathbf{X}^T(\mathbf{u}-\mathbf{y})\} \\ &= (\mathbf{X}^T\mathbf{R}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{R}\mathbf{z}\end{aligned}$$

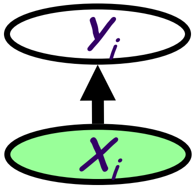
Generative vs. Discriminative Classifiers

- Goal: Wish to learn f: X $\rightarrow$ Y, e.g., P(Y|X)

- Generative classifiers (e.g., Naïve Bayes):
  - 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)



- Discriminative classifiers:
  - 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



Naïve Bayes vs Logistic Regression

- Consider Y boolean, X continuous, X=<X[1] ... X[m]>
- Number of parameters to estimate:

NB:

$$p(y \mid \mathbf{x}) = \frac{\pi_k \exp\left\{ -\sum_j \left(\frac{1}{2\sigma_{k,j}^2} (x_j - \mu_{k,j})^2 - \log \sigma_{k,j} - C \right) \right\}}{\sum_{k'} \pi_{k'} \exp\left\{ -\sum_j \left(\frac{1}{2\sigma_{k',j}^2} (x_j - \mu_{k',j})^2 - \log \sigma_{k',j} - C \right) \right\}} \quad **$$

LR:

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

- Estimation method:
  - NB parameter estimates are uncoupled
  - LR parameter estimates are coupled

Naïve Bayes vs Logistic Regression

- Asymptotic comparison (# training examples $\rightarrow$ infinity)

- when model assumptions correct
  - NB, LR produce identical classifiers

- when model assumptions incorrect
  - LR is less biased – does not assume conditional independence
  - therefore expected to outperform NB

Naïve Bayes vs Logistic Regression

- Non-asymptotic analysis (see [Ng & Jordan, 2002])
- convergence rate of parameter estimates – how many training examples needed to assure good estimates?

  NB order log m (where m = # of attributes in X)

  LR order m

- NB converges more quickly to its (perhaps less helpful) asymptotic estimates

Some experiments from UCI data sets

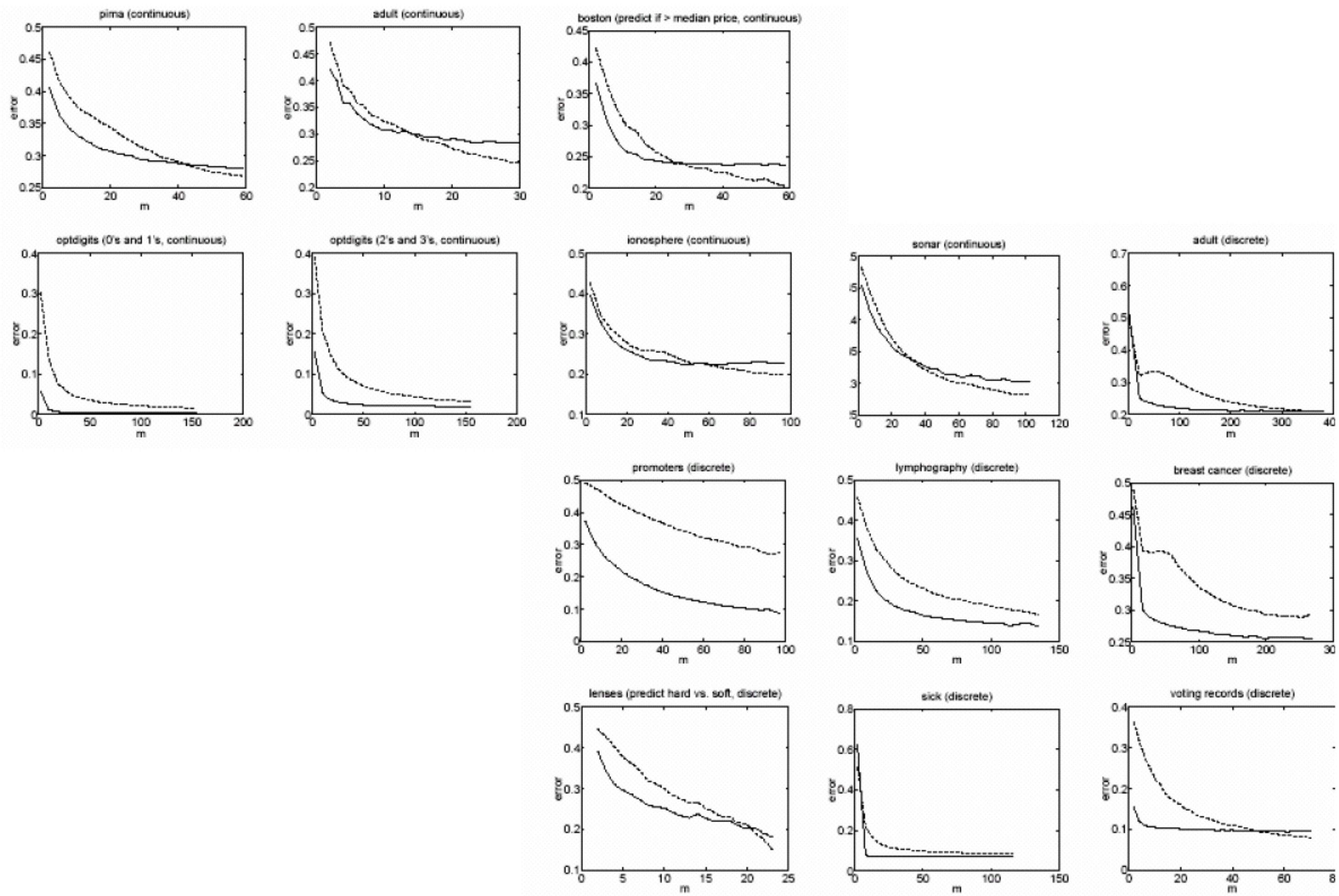
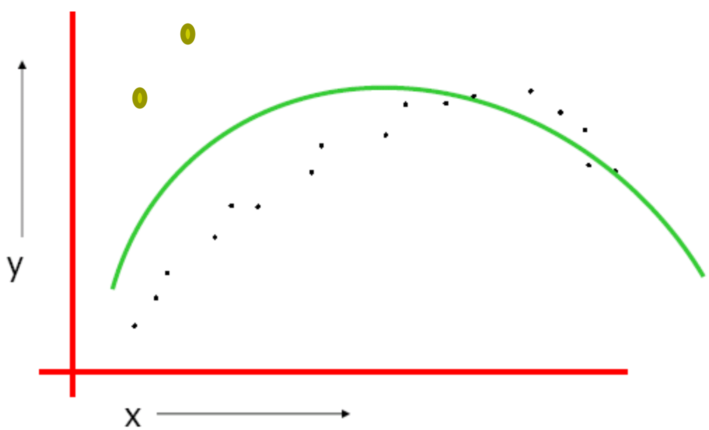


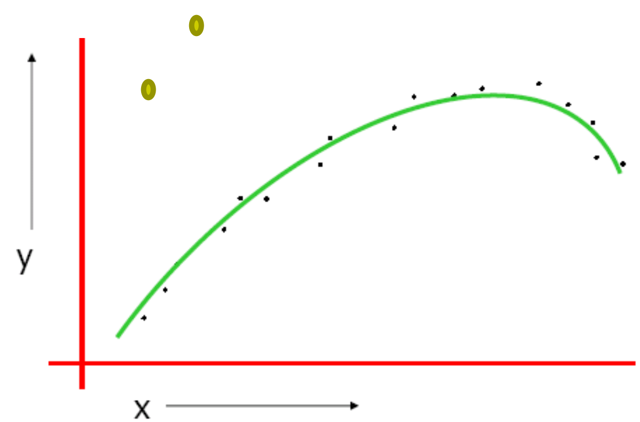
Figure 1: Results of 15 experiments on datasets from the UCI Machine Learnin repository. Plots are of generalization error vs. m (averaged over 1000 randon train/test splits). Dashed line is logistic regression; solid line is naive Bayes.

Robustness

- The best fit from a quadratic regression



- But this is probably better …



Bayesian Parameter Estimation

- Treat the distribution parameters θ also as a *random variable*
- The *a posteriori* distribution of θ after seem the data is:

$$p(\theta \mid D) = \frac{p(D \mid \theta)\, p(\theta)}{p(D)} = \frac{p(D \mid \theta)\, p(\theta)}{\int p(D \mid \theta)\, p(\theta)\, d\theta}$$

This is Bayes Rule

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}$$

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**

The prior p(.) encodes our prior knowledge about the domain

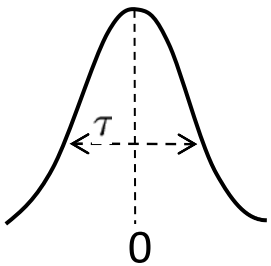
Regularized Least Squares and MAP

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

$$\widehat{\beta}_{\text{MAP}} = \arg\max_{\beta} \underbrace{\log p(\{(X_i, Y_i)\}_{i=1}^n | \beta, \sigma^2)}_{\text{log likelihood}} + \underbrace{\log p(\beta)}_{\text{log prior}}$$

I) Gaussian Prior

$$\beta \sim \mathcal{N}(0, \tau^2\mathbf{I}) \qquad p(\beta) \propto e^{-\beta^T\beta/2\tau^2}$$



$$\widehat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \underbrace{\lambda}_{\text{constant}(\sigma^2, \tau^2)} \|\beta\|_2^2$$

Ridge Regression

Closed form: HW

constant(σ^2, τ^2)

Prior belief that β is Gaussian with zero-mean biases solution to “small” β

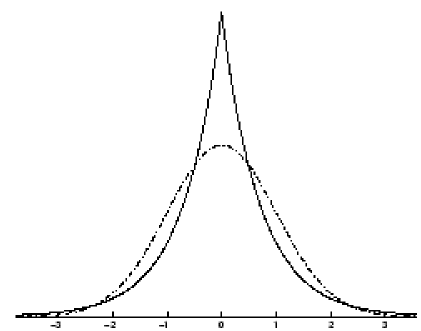
Regularized Least Squares and MAP

What if $(\mathbf{A}^T\mathbf{A})$ is not invertible ?

$$\widehat{\beta}_{\text{MAP}} = \arg\max_{\beta} \underbrace{\log p(\{(X_i, Y_i)\}_{i=1}^n | \beta, \sigma^2)}_{\text{log likelihood}} + \underbrace{\log p(\beta)}_{\text{log prior}}$$

II) Laplace Prior

$$\beta_i \overset{iid}{\sim} \text{Laplace}(0, t) \qquad p(\beta_i) \propto e^{-|\beta_i|/t}$$



$$\widehat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \underset{\substack{\downarrow \\ \text{constant}(\sigma^2, t)}}{\lambda} ||\beta||_1$$

Lasso

Closed form: HW

Prior belief that β is Laplace with zero-mean biases solution to "small" β

Ridge Regression vs Lasso

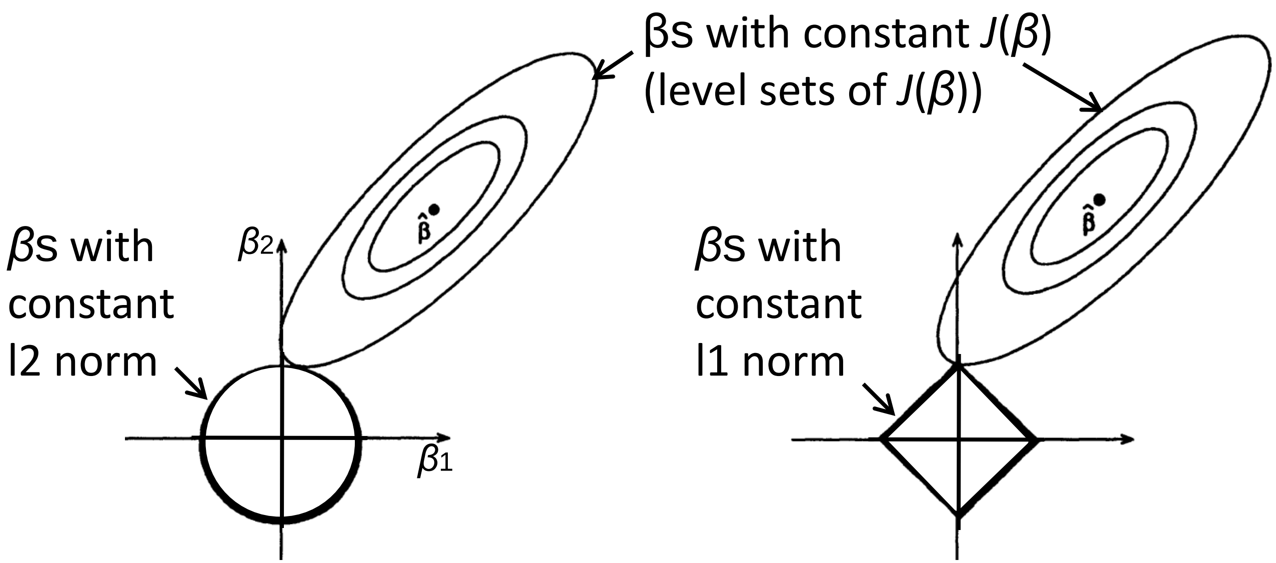
$$\min_{\beta}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) + \lambda \mathrm{pen}(\beta) = \min_{\beta} J(\beta) + \lambda \mathrm{pen}(\beta)$$

Ridge Regression:

$$\mathrm{pen}(\beta) = \|\beta\|_2^2$$

Lasso:

$$\mathrm{pen}(\beta) = \|\beta\|_1$$

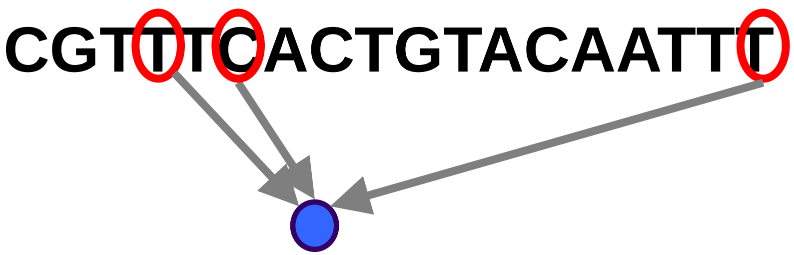


Lasso (l1 penalty) results in sparse solutions – vector with more zero coordinates
Good for high-dimensional problems – don't have to store all coordinates!

Case study: predicting gene expression

The genetic picture

causal SNPs



a univariate phenotype:

i.e., the expression intensity of a gene

Association Mapping as Regression

	Phenotype (BMI)	Genotype
Individual 1	2.5	..C.....T..C........T... ..C.....A..C........T...
Individual 2	4.8	..G .....A..G........A... ..C.....T..C........T...
⋮		
Individual N	4.7	..G .....T..C........T... ..G .....T..G........T...

Benign SNPs **Causal SNP**

Association Mapping as Regression

	Phenotype (BMI)	Genotype
Individual 1	2.5	. . 0 . . . . . 1 . . 0 . . . . 0 . . .
Individual 2	4.8	. . 1 . . . . . 1 . . 1 . . . . 1 . . .
⋮		
Individual N	4.7	. . 2 . . . . . 2 . . 1 . . . . 0 . . .

$$\mathbf{y}_i = \sum_{j=1}^{J} x_{ij} \beta_j$$

SNPs with large $|\beta_j|$ are relevant

© Eric Xing @ CMU, 2006-2008 40

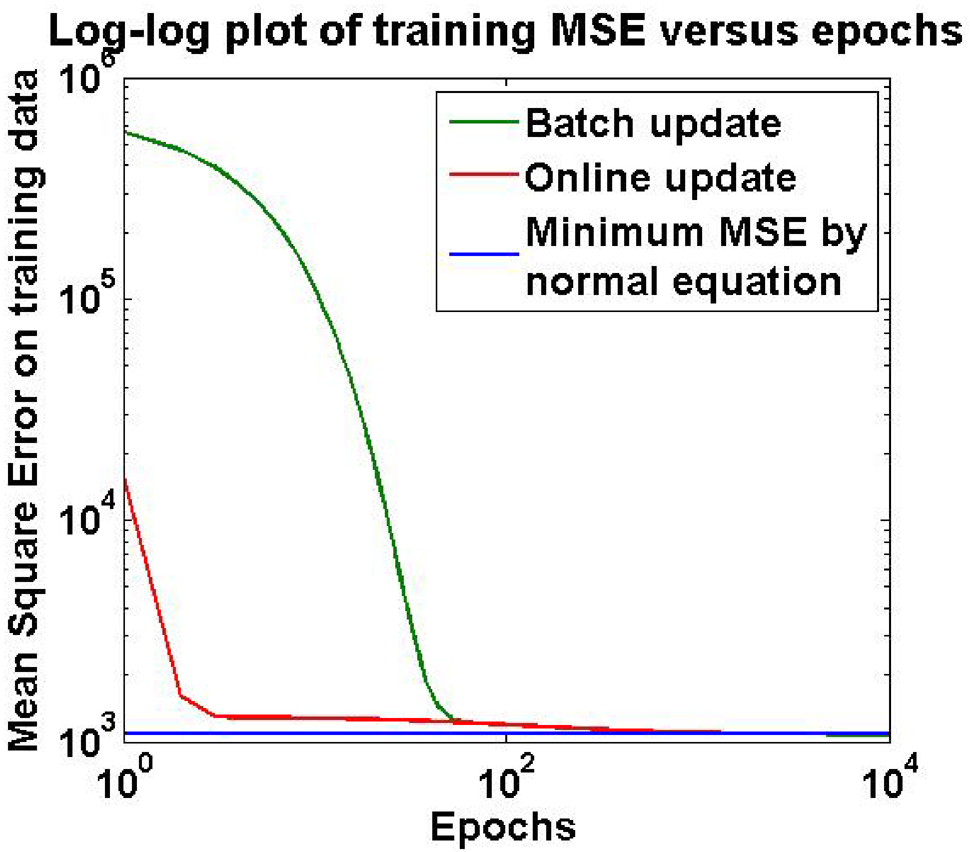
Experimental setup

- Asthama dataset
  - 543 individuals, genotyped at 34 SNPs
  - Diploid data was transformed into 0/1 (for homozygotes) or 2 (for heterozygotes)
  - X=543x34 matrix
  - Y=Phenotype variable (continuous)
- A single phenotype was used for regression

- Implementation details
  - Iterative methods: Batch update and online update implemented.
  - For both methods, step size α is chosen to be a small fixed value (10^{-6}). This choice is based on the data used for experiments.
  - Both methods are only run to a maximum of 2000 epochs or until the change in training MSE is less than 10-4

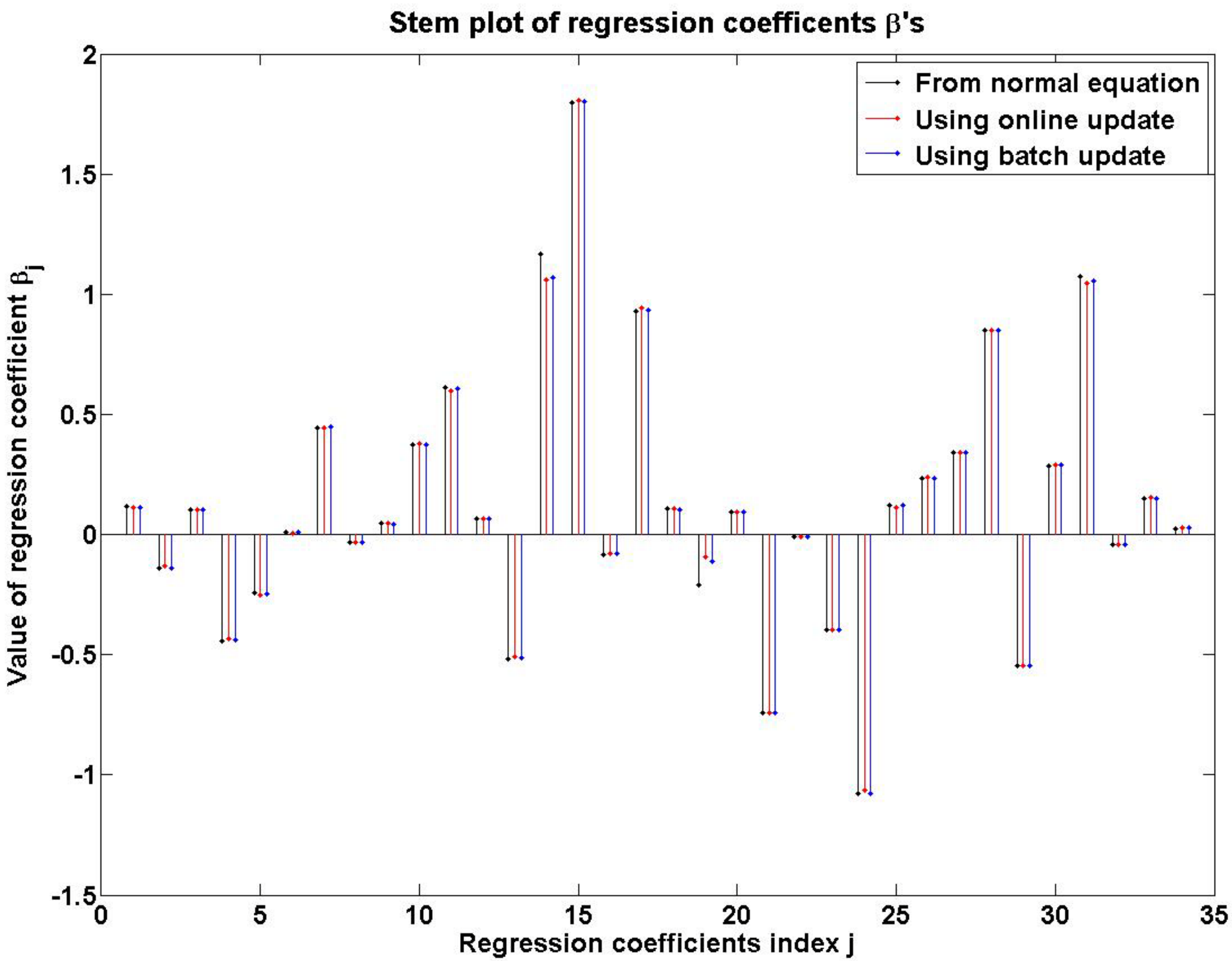
Convergence Curves

- For the batch method, the training MSE is initially large due to uninformed initialization
- In the online update, N updates for every epoch reduces MSE to a much smaller value.

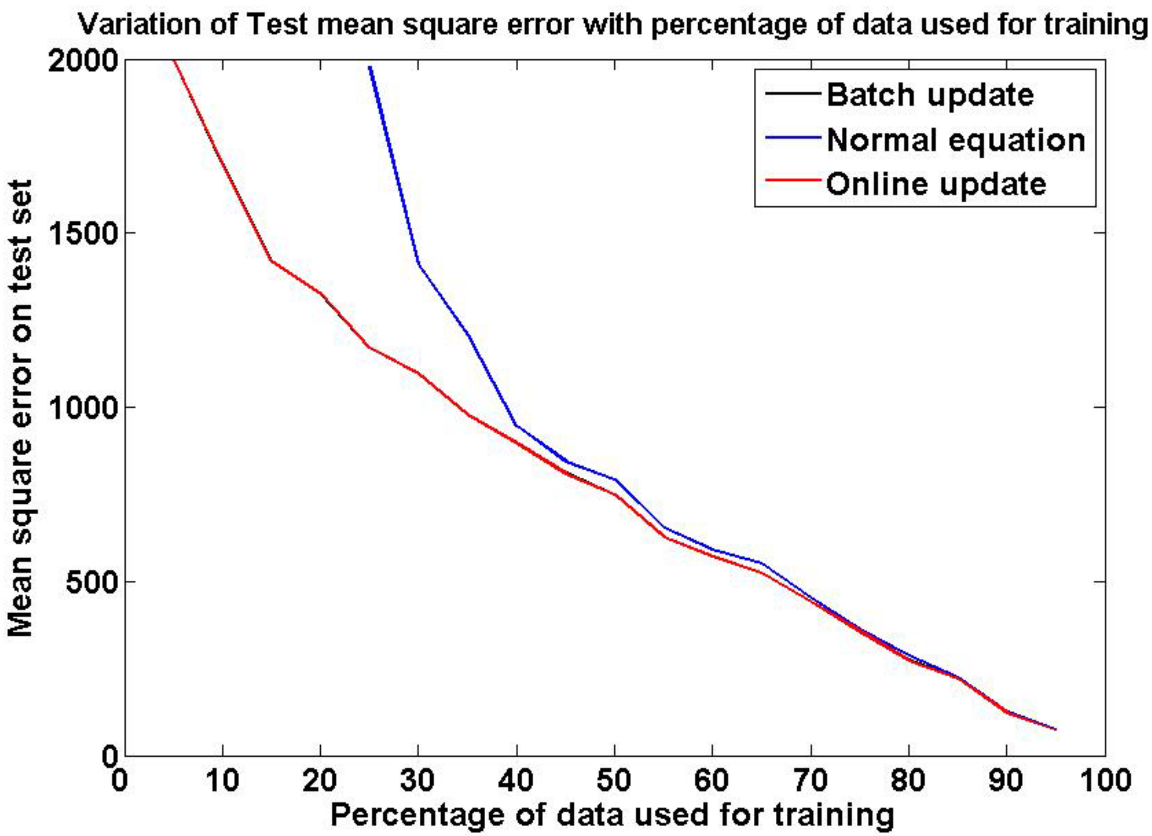


© Eric Xing @ CMU, 2006-2008
42

The Learned Coefficients



Performance vs. Training Size



- The results from B and O update are almost identical. So the plots coincide.
- The test MSE from the normal equation is more than that of B and O during small training. This is probably due to overfitting.
- In B and O, since only 2000 iterations are allowed at most. This roughly acts as a mechanism that avoids overfitting.

Summary

- Naïve Bayes classifier
  - What's the assumption
  - Why we use it
  - How do we learn it
- Logistic regression
  - Functional form follows from Naïve Bayes assumptions
  - For Gaussian Naïve Bayes assuming variance
  - For discrete-valued Naïve Bayes too
  - But training procedure picks parameters without the conditional independence assumption
- Gradient ascent/descent
  - – General approach when closed-form solutions unavailable
- Generative vs. Discriminative classifiers
  - – Bias vs. variance tradeoff

Appendix

Parameter Learning from *iid* Data

- Goal: estimate distribution parameters θ from a dataset of N independent, identically distributed (*iid*), fully observed, training cases

$$D = \{x_1, \ldots, x_N\}$$

- Maximum likelihood estimation (MLE)
  1. One of the most common estimators
  2. With iid and full-observability assumption, write $L(\theta)$ as the likelihood of the data:

$$\begin{aligned} L(\theta) &= P(x_1, x_2, \ldots, x_N; \theta) \\ &= P(x;\theta)P(x_2;\theta), \ldots, P(x_N;\theta) \\ &= \prod_{i=1}^{N} P(x_i;\theta) \end{aligned}$$

  3. pick the setting of parameters most likely to have generated the data we saw:

$$\theta^* = \arg\max_\theta L(\theta) \quad = \arg\max_\theta \log L(\theta)$$

Example: Bernoulli model

- Data:
  - We observed *N* ***iid*** coin tossing: *D*={1, 0, 1, …, 0}
- Representation:

Binary r.v: $x_n = \{0,1\}$

- Model: $P(x) = \begin{cases} 1-\theta & \text{for } x=0 \\ \theta & \text{for } x=1 \end{cases} \quad \Rightarrow \quad P(x) = \theta^x (1-\theta)^{1-x}$

- How to write the likelihood of a single observation x_i ?

$$P(x_i) = \theta^{x_i} (1-\theta)^{1-x_i}$$

- The likelihood of dataset *D*={x_1, …,x_N}:

$$P(x_1, x_2, \dots, x_N \mid \theta) = \prod_{i=1}^{N} P(x_i \mid \theta) = \prod_{i=1}^{N} \left(\theta^{x_i} (1-\theta)^{1-x_i} \right) = \theta^{\sum_{i=1}^{N} x_i} (1-\theta)^{\sum_{i=1}^{N} 1-x_i} = \theta^{\#\text{head}} (1-\theta)^{\#\text{tails}}$$

Maximum Likelihood Estimation

- Objective function:

$$\ell(\theta; D) = \log P(D \mid \theta) = \log \theta^{n_h} (1-\theta)^{n_t} = n_h \log \theta + (N - n_h) \log(1-\theta)$$

- We need to maximize this w.r.t. θ

- Take derivatives wrt θ

$$\frac{\partial \ell}{\partial \theta} = \frac{n_h}{\theta} - \frac{N - n_h}{1-\theta} = 0 \quad \Rightarrow \quad \widehat{\theta}_{MLE} = \frac{n_h}{N} \quad \text{or} \quad \widehat{\theta}_{MLE} = \frac{1}{N} \sum_i x_i$$

Frequency as sample mean

- Sufficient statistics
  - The counts, n_h, where $n_k = \sum_i x_i$, are **sufficient statistics** of data D

Overfitting

- Recall that for Bernoulli Distribution, we have

$$\widehat{\theta}_{ML}^{head} = \frac{n^{head}}{n^{head} + n^{tail}}$$

- What if we tossed too few times so that we saw zero head?
  We have $\widehat{\theta}_{ML}^{head} = 0,$ and we will predict that the probability of seeing a head next is zero!!!

- The rescue: *"smoothing"*
  - Where n' is know as the pseudo- (imaginary) count

$$\widehat{\theta}_{ML}^{head} = \frac{n^{head} + n'}{n^{head} + n^{tail} + n'}$$

  - But can we make this more formal?

Bayesian Parameter Estimation

- Treat the distribution parameters θ also as a *random variable*
- The *a posteriori* distribution of θ after seem the data is:

$$p(\theta \mid D) = \frac{p(D \mid \theta)\, p(\theta)}{p(D)} = \frac{p(D \mid \theta)\, p(\theta)}{\int p(D \mid \theta)\, p(\theta)\, d\theta}$$

This is Bayes Rule

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}$$

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**

The prior p(.) encodes our prior knowledge about the domain

Frequentist Parameter Estimation

Two people with different priors $p(\theta)$ will end up with different estimates $p(\theta|D)$.

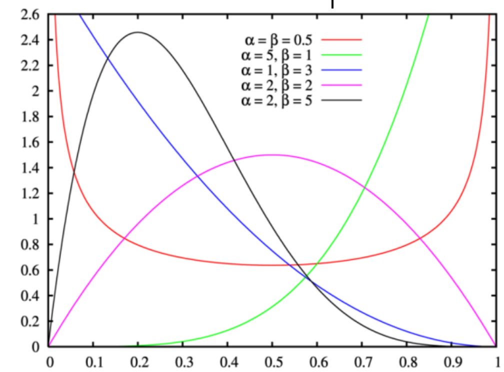
- Frequentists dislike this “subjectivity”.
- Frequentists think of the parameter as a fixed, unknown constant, not a random variable.
- Hence they have to come up with different "objective" **estimators** (ways of computing from data), instead of using Bayes’ rule.
  - These estimators have different properties, such as being “unbiased”, “minimum variance”, etc.
  - The maximum likelihood estimator, is one such estimator.

Discussion

θ **or** $p(\theta)$**, this is the problem!**

Bayesian estimation for Bernoulli

- Beta distribution:



$$P(\theta;\alpha,\beta)=\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}=B(\alpha,\beta)\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

  - When x is discrete $\Gamma(x+1)=x\Gamma(x)=x!$

- Posterior distribution of θ:

$$P(\theta \mid x_1,\dots,x_N)=\frac{p(x_1,\dots,x_N \mid \theta)\,p(\theta)}{p(x_1,\dots,x_N)}\propto \theta^{n_h}(1-\theta)^{n_t}\times\theta^{\alpha-1}(1-\theta)^{\beta-1}=\theta^{n_h+\alpha-1}(1-\theta)^{n_t+\beta-1}$$

  - Notice the isomorphism of the posterior to the prior,
  - such a prior is called a **conjugate prior**
  - α *and* β are hyperparameters (parameters of the prior) and correspond to the number of “virtual” heads/tails (pseudo counts)

Bayesian estimation for Bernoulli, con'd

- Posterior distribution of θ:

$$P(\theta \mid x_1,\ldots,x_N) = \frac{p(x_1,\ldots,x_N \mid \theta)\, p(\theta)}{p(x_1,\ldots,x_N)} \propto \theta^{n_h}(1-\theta)^{n_t} \times \theta^{\alpha-1}(1-\theta)^{\beta-1} = \theta^{n_h+\alpha-1}(1-\theta)^{n_t+\beta-1}$$

- Maximum *a posteriori* (MAP) estimation:

$$\theta_{MAP} = \arg\max_{\theta} \log P(\theta \mid x_1,\ldots,x_N)$$

- Posterior mean estimation:

Bata parameters can be understood as pseudo-counts

$$\theta_{Bayes} = \int \theta p(\theta \mid D) d\theta = C \int \theta \times \theta^{n_h+\alpha-1}(1-\theta)^{n_t+\beta-1} d\theta = \frac{n_h+\alpha}{N+\alpha+\beta}$$

- Prior strength: A=α+β
  - A can be interoperated as the size of an imaginary data set from which we obtain the **pseudo-counts**

Effect of Prior Strength

- Suppose we have a uniform prior ($\alpha=\beta=1/2$),
  and we observe $\vec{n} = (n_h = 2, n_t = 8)$
- Weak prior A = 2. Posterior prediction:

$$p(x = h \mid n_h = 2, n_t = 8, \vec{\alpha} = \vec{\alpha}' \times 2) = \frac{1+2}{2+10} = 0.25$$

- Strong prior A = 20. Posterior prediction:

$$p(x = h \mid n_h = 2, n_t = 8, \vec{\alpha} = \vec{\alpha}' \times 20) = \frac{10+2}{20+10} = 0.40$$

- However, if we have enough data, it washes away the prior. e.g., $\vec{n} = (n_h = 200, n_t = 800)$. Then the estimates under weak and strong prior are $\frac{1+200}{2+1000}$ and $\frac{10+200}{20+1000}$, respectively, both of which are close to 0.2

Example 2: Gaussian density

- Data:
  - We observed *N* ***iid*** real samples:
    D={-0.1, 10, 1, -5.2, …, 3}
- Model: $P(x) = \left(2\pi\sigma^2\right)^{-1/2} \exp\left\{-(x-\mu)^2 / 2\sigma^2\right\}$
- Log likelihood:

$$\ell(\theta; D) = \log P(D \mid \theta) = -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2}\sum_{n=1}^{N}\frac{(x_n-\mu)^2}{\sigma^2}$$

- MLE: take derivative and set to zero:

$$\frac{\partial \ell}{\partial \mu} = (1/\sigma^2)\sum_n (x_n - \mu)$$

$$\frac{\partial \ell}{\partial \sigma^2} = -\frac{N}{2\sigma^2} + \frac{1}{2\sigma^4}\sum_n (x_n-\mu)^2$$

$$\Rightarrow$$

$$\mu_{MLE} = \frac{1}{N}\sum_n (x_n)$$

$$\sigma^2_{MLE} = \frac{1}{N}\sum_n (x_n - \mu_{ML})^2$$

MLE for a multivariate-Gaussian

- It can be shown that the MLE for μ *and* Σ is

$$x_n = \begin{pmatrix} x_n^1 \\ x_n^2 \\ \vdots \\ x_n^K \end{pmatrix}$$

$$\mu_{MLE} = \frac{1}{N}\sum_n (x_n)$$

$$\Sigma_{MLE} = \frac{1}{N}\sum_n (x_n - \mu_{ML})(x_n - \mu_{ML})^T = \frac{1}{N} S$$

$$X = \begin{pmatrix} --- x_1^T --- \\ --- x_2^T --- \\ \vdots \\ --- x_N^T --- \end{pmatrix}$$

where the scatter matrix is

$$S = \sum_n (x_n - \mu_{ML})(x_n - \mu_{ML})^T = \left(\sum_n x_n x_n^T\right) - N\mu_{ML}\mu_{ML}^T$$

- The sufficient statistics are $\Sigma_n x_n$ and $\Sigma_n x_n x_n^T$.
- Note that $X^T X = \Sigma_n x_n x_n^T$ may not be full rank (eg. if $N < D$), in which case Σ_{ML} is not invertible

Bayesian estimation

- Normal Prior:

$$P(\mu) = \left(2\pi\sigma_0^2\right)^{-1/2} \exp\left\{-(\mu-\mu_0)^2 / 2\sigma_0^2\right\}$$

- Joint probability:

$$P(x,\mu) = \left(2\pi\sigma^2\right)^{-N/2} \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2\right\}$$
$$\times\left(2\pi\sigma_0^2\right)^{-1/2} \exp\left\{-(\mu-\mu_0)^2 / 2\sigma_0^2\right\}$$

- Posterior:

$$P(\mu \mid x) = \left(2\pi\tilde{\sigma}^2\right)^{-1/2} \exp\left\{-(\mu-\tilde{\mu})^2 / 2\tilde{\sigma}^2\right\}$$

where $\tilde{\mu} = \frac{N/\sigma^2}{N/\sigma^2 + 1/\sigma_0^2}\bar{x} + \frac{1/\sigma_0^2}{N/\sigma^2 + 1/\sigma_0^2}\mu_0$, and $\tilde{\sigma}^2 = \left(\frac{N}{\sigma^2} + \frac{1}{\sigma_0^2}\right)^{-1}$

Sample mean

Bayesian estimation: unknown μ, known σ

$$\mu_N = \frac{N/\sigma^2}{N/\sigma^2 + 1/\sigma_0^2}\bar{x} + \frac{1/\sigma_0^2}{N/\sigma^2 + 1/\sigma_0^2}\mu_0\,, \qquad \tilde{\sigma}^2 = \left(\frac{N}{\sigma^2} + \frac{1}{\sigma_0^2}\right)^{-1}$$

- The posterior mean is a convex combination of the prior and the MLE, with weights proportional to the relative noise levels.
- The precision of the posterior $1/\sigma^2_N$ is the precision of the prior $1/\sigma^2_0$ plus one contribution of data precision $1/\sigma^2$ for each observed data point.

- Sequentially updating the mean
  - $\mu* = 0.8$ (unknown), $(\sigma^2)* = 0.1$ (known)
  - Effect of single data point

    $$\mu_1 = \mu_0 + (x - \mu_0)\frac{\sigma_0^2}{\sigma^2 + \sigma_0^2} = x - (x - \mu_0)\frac{\sigma_0^2}{\sigma^2 + \sigma_0^2}$$

  - Uninformative (vague/ flat) prior, $\sigma^2_0 \rightarrow \infty$

    $$\mu_N \rightarrow \mu_0$$