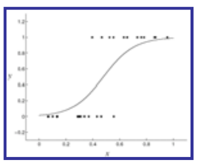
Machine Learning

10-701/15-781, Fall 2012

Generative verses discriminative classifier

Eric Xing and Aarti Singh

Lecture 4, September 19, 2012

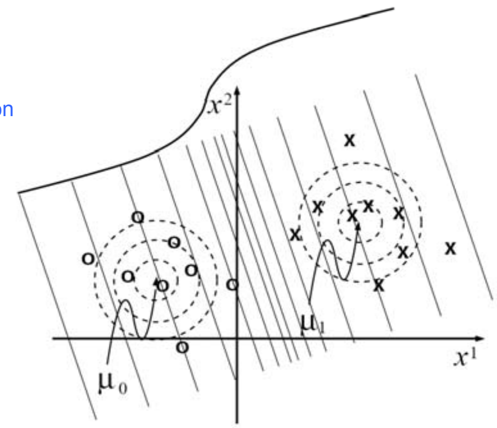


Reading:

- How was your hw? Save at least 10 hours for it.
- About project
- About team formation

Generative vs. Discriminative classifiers

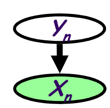
- Goal: Wish to learn f: X → Y, e.g., P(Y|X)
- Generative:
  - Modeling the joint distribution of all data
- Discriminative:
  - Modeling only points at the boundary



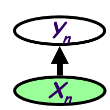
© Eric Xing @ CMU, 2006-2012 3

Learning Generative and Discriminative Classifiers

- Goal: Wish to learn f: X → 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)



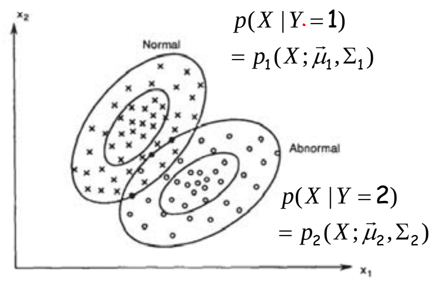
- Discriminative classifiers (e.g., logistic regression)
  - Directly assume some functional form for P(Y|X)
    This is a '***discriminative***' model of the data!
  - Estimate parameters of P(Y|X) directly from training data



© Eric Xing @ CMU, 2006-2012 4

Suppose you know the following …

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



Bayes classifier:

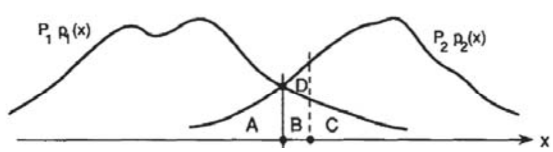
$$\boxed{P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}}$$

- 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

Learning Bayes Classifier

- Training data (discrete case):

X: Sky, Temp, Humid, Wind, Water, Forecst; Y: EnjoySpt

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

- Learning = estimating P(X|Y), and P(Y)
- Classification = using Bayes rule to calculate P(Y | X_{new})

Parameter learning from *iid* data: The Maximum Likelihood Est.

- 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:

$$L(\theta) = P(x_1, x_2, \ldots, x_N; \theta)$$
$$= P(x;\theta)P(x_2;\theta), \ldots, P(x_N;\theta)$$
$$= \prod_{n=1}^{N} P(x_n;\theta)$$

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

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

How hard is it to learn the optimal classifier?

- How do we represent these? How many parameters?
  - Prior, P(Y):
    - Suppose Y is composed of *k* classes

$$\overbrace{\qquad\qquad\qquad\qquad\qquad\qquad}^{X}\ \overbrace{\qquad}^{Y}$$

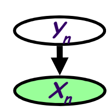
Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

  - Likelihood, P(**X**|Y):
    - Suppose **X** is composed of *n* binary features

- Complex model $\rightarrow$ High variance with limited data!!!

Gaussian Discriminative Analysis

- learning f: X $\rightarrow$ Y, where
  - X is a vector of real-valued features, $\mathbf{X}_n = < X_n^1, \ldots X_n^m >$
  - 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:

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

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

$$p(y_n^k = 1 \mid \mathbf{x}_n, \mu, \Sigma) = \frac{\pi_k \frac{1}{(2\pi|\Sigma|)^{1/2}} \exp\left\{-\tfrac{1}{2}(\mathbf{x}_n - \vec{\mu}_k)^T \Sigma^{-1} (\mathbf{x}_n - \vec{\mu}_k)\right\}}{\sum_{k'} \pi_{k'} \frac{1}{(2\pi|\Sigma|)^{1/2}} \exp\left\{-\tfrac{1}{2}(\mathbf{x}_n - \vec{\mu}_k)^T \Sigma^{-1} (\mathbf{x}_n - \vec{\mu}_{k'})\right\}}$$

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:

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

The Naïve Bayes assumption

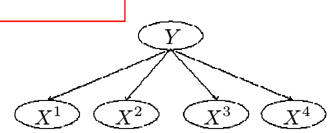
- Naïve Bayes assumption:
  - Features are conditionally independent given class:

$$\begin{aligned} P(X_1, X_2|Y) &= P(X_1|X_2, Y) P(X_2|Y) \\ &= P(X_1|Y) P(X_2|Y) \end{aligned}$$

  - More generally:

$$\boxed{P(X^1 ... X^n|Y) = \prod_i P(X^i|Y)}$$

- How many parameters now?
  - Suppose **X** is composed of *m* binary features



The Naïve Bayes Classifier

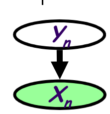
- Given:
  - Prior P(Y)
  - *m* conditionally independent features **X** given the class Y
  - For each X_n, we have likelihood $P(X_n|Y)$
- Decision rule:

$$y^* = h_{NB}(\mathbf{x}) = \arg\max_y P(y)P(x^1, \ldots, x^m \mid y) = \arg\max_y P(y)\prod_i P(x^i|y)$$

- If assumption holds, NB is optimal classifier!

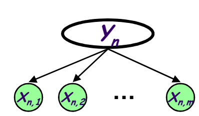
The A Gaussian Discriminative Naïve Bayes Classifier

- 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



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

  - 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:

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

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

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

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

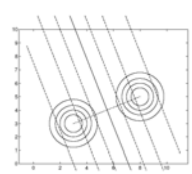
The decision boundary

- The predictive distribution

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

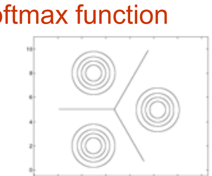
- The Bayes decision rule:

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



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

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



Summary: The Naïve Bayes Algorithm

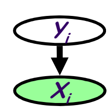
- Train Naïve Bayes (examples)
  - for each* value y_k
  - estimate $\pi_k \equiv P(Y = y_k)$
  - for each* value x_{ij} of each attribute X_i
  - estimate $\theta_{ijk} \equiv P(X^i = x_{ij} | Y = y_k)$
- Classify (X_{new})

$$Y^{new} \leftarrow \arg\max_{y_k} P(Y = y_k) \prod_i P(X^i = x_{ij} | Y = y_k)$$

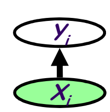
$$Y^{new} \leftarrow \arg\max_{y_k} \pi_k \prod_i \theta_{ijk}$$

Generative vs. Discriminative Classifiers

- Goal: Wish to learn f: X → 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



Recall the predictive law under NB

Recall the NB 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:

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

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

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

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

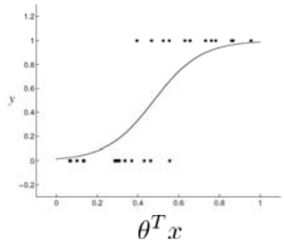
Logistic regression (sigmoid classifier)

- The condition distribution: a Bernoulli

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

where μ is a logistic function

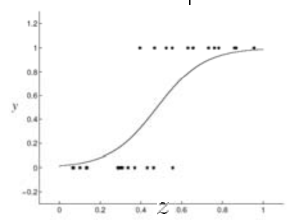
$$\mu(x) = \frac{1}{1 + e^{-\theta^T x}} = p(y = 1 \mid x)$$



- In this case, learning p(y|x) amounts to learning ...?
- What is the difference to NB?

The logistic function

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



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

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

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

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

Maximizing Conditional Log Likelihood

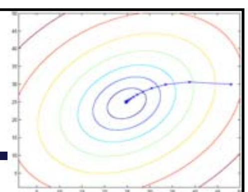
- The objective:

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

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

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

Gradient Ascent



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

- Property of sigmoid function:

$$\mu = \frac{1}{1+e^{-t}} \qquad \frac{d\mu}{dt} = \mu(1-\mu)$$

- The gradient:

$$\frac{\partial l(\theta)}{\partial \theta_j} =$$

The gradient ascent algorithm iterate until change < ε

For all *i*, $\theta_j \leftarrow \theta_j + \eta \sum_i (y_i - P(y_i = 1|x_i;\theta))x_i^j$

repeat

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

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

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

- This is also known as Iterative reweighed least squares (IRLS)

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:

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

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

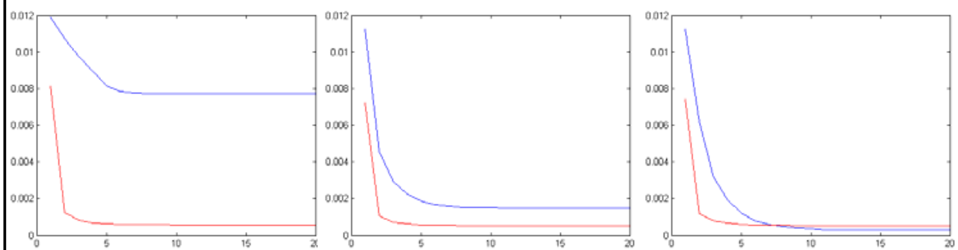
- Now for logistic regression:

$$\theta^{t+1} = (\mathbf{X}^T\mathbf{R}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{R}\mathbf{z}$$

$$\text{where} \quad \mathbf{z} = \mathbf{X}\theta^t - \mathbf{R}^{-1}(\mathbf{u}-\mathbf{y})$$

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

Convergence curves



alt.atheism vs. comp.graphics | rec.autos vs. rec.sport.baseball | comp.windows.x vs. rec.motorcycles

Legend: - X-axis: Iteration #; Y-axis: error
- In each figure, red for IRLS and blue for gradient descent

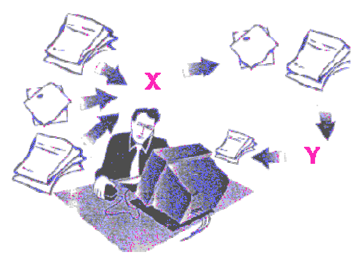
Logistic regression: practical issues

- NR (IRLS) takes $O(N+d^3)$ per iteration, where N = number of training cases and d = dimension of input x, but converge in fewer iterations
- Quasi-Newton methods, that approximate the Hessian, work faster.
- Conjugate gradient takes $O(Nd)$ per iteration, and usually works best in practice.
- Stochastic gradient descent can also be used if N is large c.f. perceptron rule:

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

the world of TOTAL

All About The Company
Global Activities
Corporate Structure
TOTAL's Story
Upstream Strategy
Downstream Strategy
Chemicals Strategy
TOTAL Foundation
Homepage

all about the **company**

Our energy exploration, production, and distribution operations span the globe, with activities in more than 100 countries.

At TOTAL, we draw our greatest strength from our fast-growing oil and gas reserves. Our strategic emphasis on natural gas provides a strong position in a rapidly expanding market.

Our expanding refining and marketing operations in Asia and the Mediterranean Rim complement already solid positions in Europe, Africa, and the U.S.

Our growing specialty chemicals sector adds balance and profit to the core energy business.

→

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

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 often works very well!

$$P(y) \prod_{i=1}^{LengthDoc} P(x^i|y) \quad \textbf{or} \quad P(y) \prod_{k=1}^{LengthVol} P(w^k|y)$$

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 often works very well!

$$P(y) \prod_{i=1}^{LengthDoc} P(x^i|y) \quad \textbf{or} \quad P(y) \prod_{k=1}^{LengthVol} P(w^k|y)$$

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

NB with Bag of Words for text classification

- Learning phase:
  - Prior P(Y)
    - Count how many documents you have from each topic (+ prior)
  - $P(X^i|Y)$
    - For each topic, count how many times you saw word in documents of this topic (+ prior)
- Test phase:
  - For each document $\mathbf{x}_{new}$
    - Use naïve Bayes decision rule

$$h_{NB}(\mathbf{x}_{\text{new}}) = \arg\max_y P(y) \prod_{i=1}^{LengthDoc} P(x^i_{\text{new}}|y)$$

Back to our 20 NG Case study

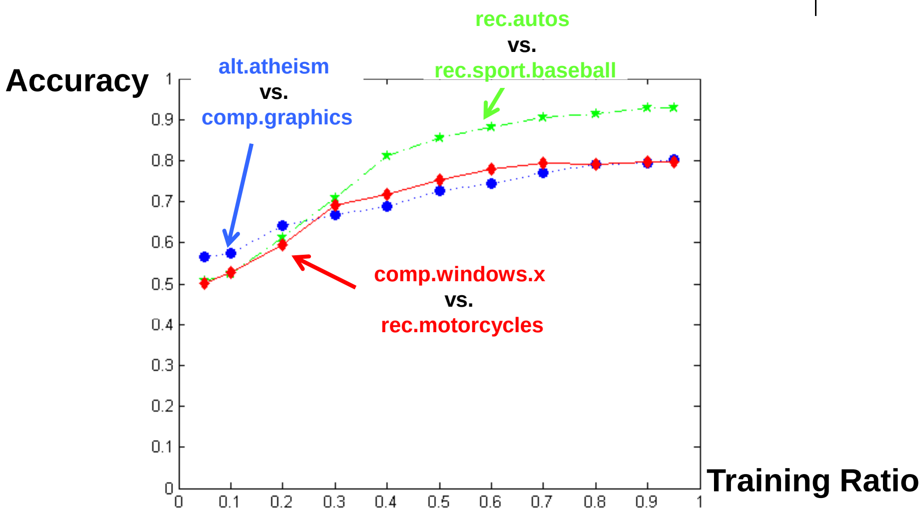
- Dataset
  - 20 News Groups (20 classes)
  - 61,118 words, 18,774 documents

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

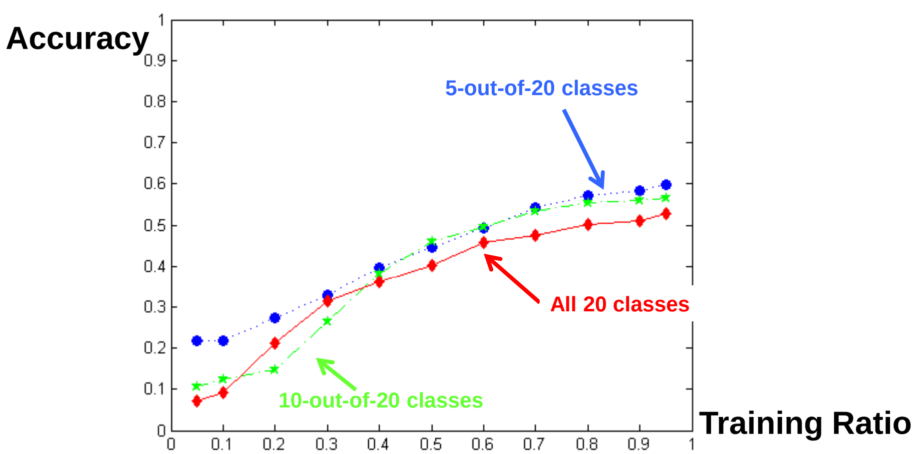
- Experiment:
  - Solve only a two-class subset: 1 vs 2.
  - 1768 instances, 61188 features.
  - Use dimensionality reduction on the data (SVD).
  - Use 90% as training set, 10% as test set.
  - Test prediction error used as accuracy measure.

$$Accuracy = \frac{\sum_{i \in test\ set} \mathbb{I}(predict_i == true\ label_i)}{\#\ of\ test\ samples}$$

Results: Binary Classes

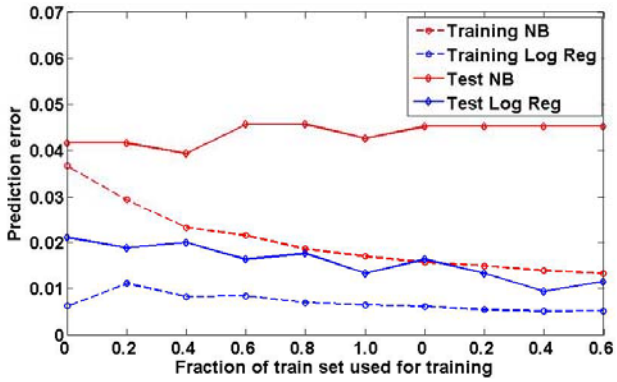


Results: Multiple Classes



NB vs. LR

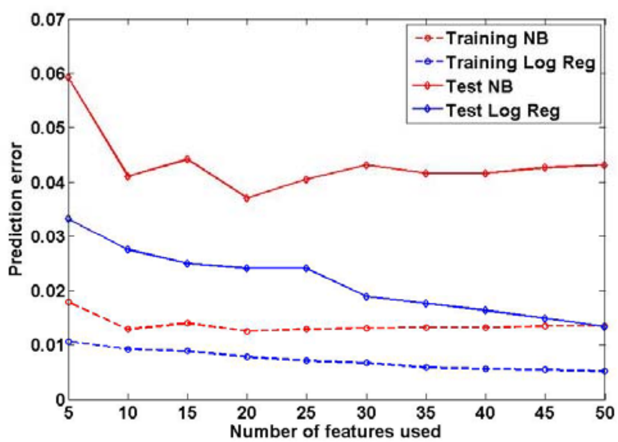
- Versus training size



- 30 features.
- A fixed test set
- Training set varied from 10% to 100% of the training set

NB vs. LR

- Versus model size



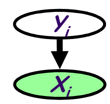
Number of dimensions of the data varied from 5 to 50 in steps of 5

The features were chosen in decreasing order of their singular values

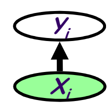
90% versus 10% split on training and test

Generative vs. Discriminative Classifiers

- Goal: Wish to learn f: X → 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:

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

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 → 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

Rate of convergence: logistic regression

- Let $h_{Dis,m}$ be logistic regression trained on n examples in m dimensions. Then with high probability:

$$\epsilon(h_{Dis,n}) \leq \epsilon(h_{Dis,\infty}) + O\left(\sqrt{\frac{m}{n}\log\frac{n}{m}}\right)$$

- Implication: if we want $\epsilon(h_{Dis,m}) \leq \epsilon(h_{Dis,\infty}) + \epsilon_0$ for some small constant ε_0, it suffices to pick order m examples

  → Convergences to its asymptotic classifier, in order m examples

  - result follows from Vapnik's structural risk bound, plus fact that the "VC Dimension" of an m-dimensional linear separators is m

Rate of convergence: naïve Bayes parameters

- Let any ε_1, δ>0, and any $n \geq 0$ be fixed.
  Assume that for some fixed $\rho_0 > 0$,
  we have that $\rho_0 \leq p(y = T) \leq 1 - \rho_0$

- Let $n = O((1/\epsilon_1^2)\log(m/\delta))$

- Then with probability at least 1-δ, after n examples:

  1. For discrete input, $|\hat{p}(x_i|y=b) - p(x_i|y=b)| \leq \epsilon_1$, $|\hat{p}(y=b) - p(y=b)| \leq \epsilon_1$ for all i and b
  2. For continuous inputs, $|\hat{\mu}_{i|y=b} - \mu_{i|y=b}| \leq \epsilon_1$, $|\hat{\sigma}^2_{i|y=b} - \sigma^2_{i|y=b}| \leq \epsilon_1$ for all i and b

Some experiments from UCI data sets

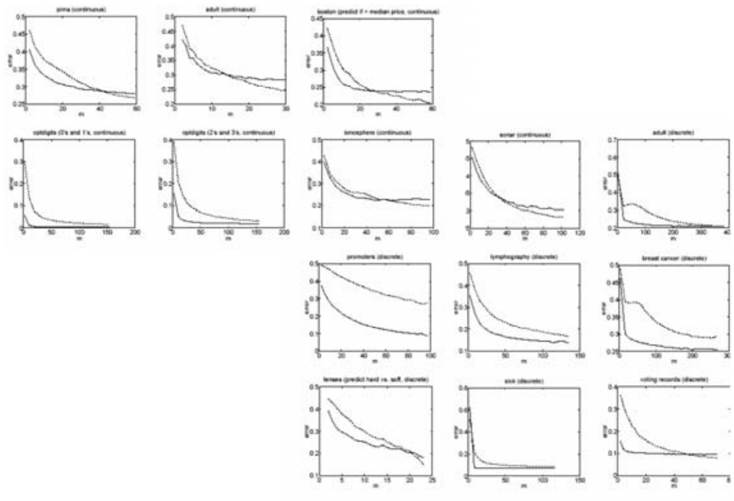


Figure 1: Results of 15 experiments on datasets from the UCI Machine Learnin repository. Plots are of generalization error vs. m (averaged over 1000 randor train/test splits). Dashed line is logistic regression; solid line is naive Bayes.

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