Machine Learning 10-601

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

September 29, 2011

Today:
- Gaussian Naïve Bayes
  - real-valued X_i's
  - Brain image classification
- Logistic regression

Readings:

Required:
- Mitchell: "Naïve Bayes and Logistic Regression"
  (available on class website)

Optional
- Bishop 1.2.4
- Bishop 4.2

Estimating Parameters

- Maximum Likelihood Estimate (MLE): choose θ that maximizes probability of observed data $\mathcal{D}$

$$\widehat{\theta} = \arg\max_{\theta} \; P(\mathcal{D} \mid \theta)$$

- Maximum a Posteriori (MAP) estimate: choose θ that is most probable given prior probability and the data

$$\widehat{\theta} = \arg\max_{\theta} \; P(\theta \mid \mathcal{D})$$

$$= \arg\max_{\theta} \; = \frac{P(\mathcal{D} \mid \theta) P(\theta)}{P(\mathcal{D})}$$

Recently:

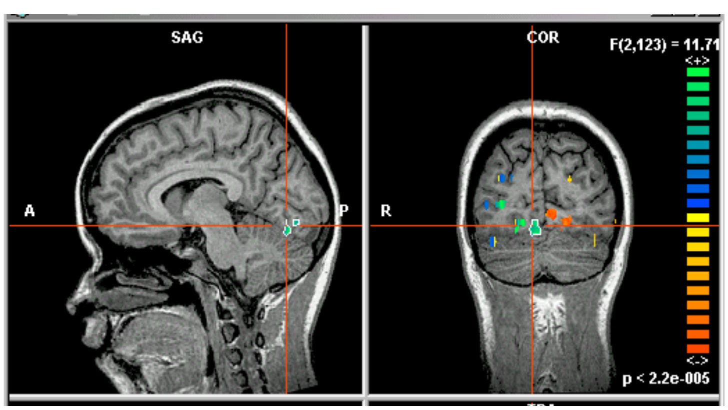
- Bayes classifiers to learn P(Y|X)
- MLE and MAP estimates for parameters of P
- Conditional independence
- Naïve Bayes → make Bayesian learning practical
- Text classification

Today:

- Naïve Bayes and continuous variables X_i:
  - Gaussian Naïve Bayes classifier
- Learn P(Y|X) directly
  - Logistic regression, Regularization, Gradient ascent
- Naïve Bayes or Logistic Regression?
  - Generative vs. Discriminative classifiers

What if we have continuous X_i ?

Eg., image classification: X_i is real-valued i^{th} pixel



What if we have continuous X_i ?

Eg., image classification: X_i is real-valued i^{th} pixel

Naïve Bayes requires $P(X_i \mid Y=y_k)$, but X_i is real (continuous)

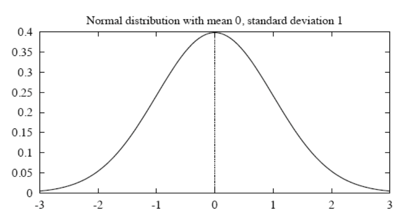
$$P(Y = y_k|X_1 \ldots X_n) = \frac{P(Y = y_k) \prod_i P(X_i|Y = y_k)}{\sum_j P(Y = y_j) \prod_i P(X_i|Y = y_j)}$$

Common approach: assume $P(X_i \mid Y=y_k)$ follows a Normal (Gaussian) distribution

Gaussian Distribution

(also called "Normal")

p(x) is a *probability density function*, whose integral (not sum) is 1



$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

The probability that X will fall into the interval (a, b) is given by

$$\int_a^b p(x)dx$$

- Expected, or mean value of X, $E[X]$, is

$$E[X] = \mu$$

- Variance of X is

$$Var(X) = \sigma^2$$

- Standard deviation of X, σ_X, is

$$\sigma_X = \sigma$$

What if we have continuous X_i ?

Gaussian Naïve Bayes (GNB): assume

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

Sometimes assume variance σ

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

Gaussian Naïve Bayes Algorithm – continuous X_i (but still discrete Y)

- Train Naïve Bayes (examples)

  for each value y_k

  estimate $\pi_k \equiv P(Y = y_k)$

  for each attribute X_i estimate $P(X_i|Y = y_k)$

  - conditional mean μ_{ik} , variance σ_{ik}

- Classify (X^{new})

$$Y^{new} \leftarrow \arg\max_{y_k} \; P(Y = y_k) \prod_i P(X_i^{new}|Y = y_k)$$

$$Y^{new} \leftarrow \arg\max_{y_k} \; \pi_k \prod_i \mathcal{N}(X_i^{new}; \mu_{ik}, \sigma_{ik})$$

Q: how many parameters must we estimate?

Estimating Parameters: Y discrete, X_i continuous

Maximum likelihood estimates:

jth training example

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

ith feature

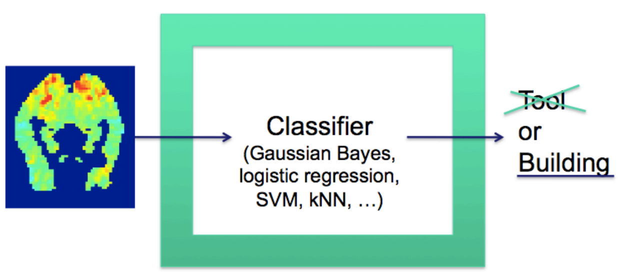
kth class

δ()=1 if ($Y^j=y_k$) else 0

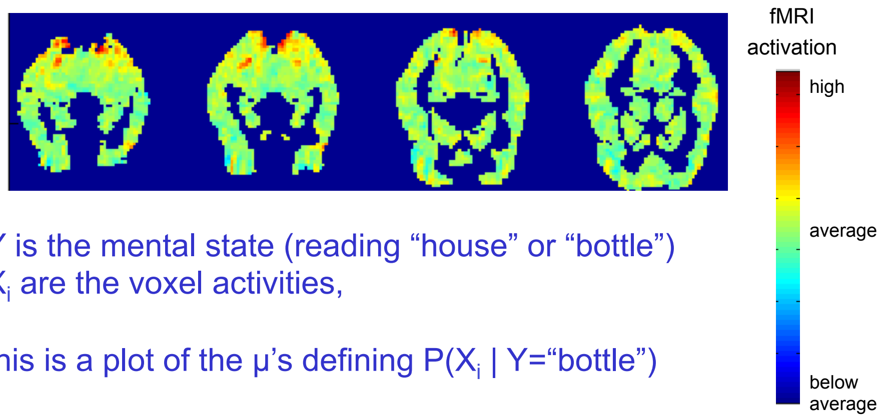
$$\hat{\sigma}_{ik}^2 = \frac{1}{\sum_j \delta(Y^j = y_k)} \sum_j (X_i^j - \hat{\mu}_{ik})^2 \delta(Y^j = y_k)$$

GNB Example: Classify a person's cognitive state, based on brain image

- reading a sentence or viewing a picture?
- reading the word describing a "Tool" or "Building"?
- answering the question, or getting confused?



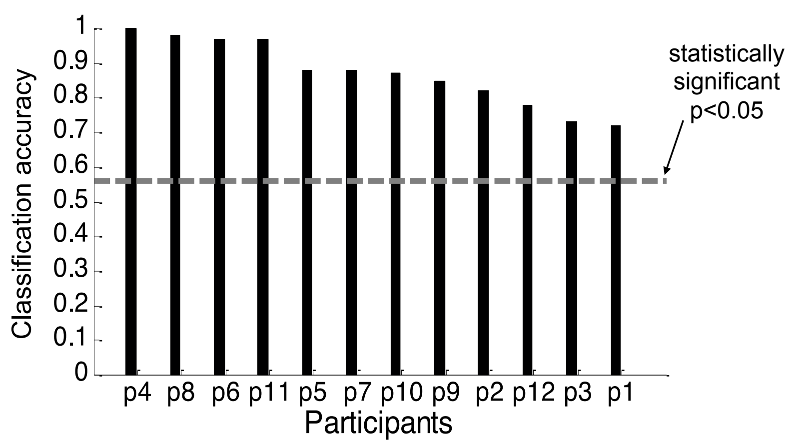
Mean activations over all training examples for Y="bottle"



Y is the mental state (reading "house" or "bottle")
X_i are the voxel activities,

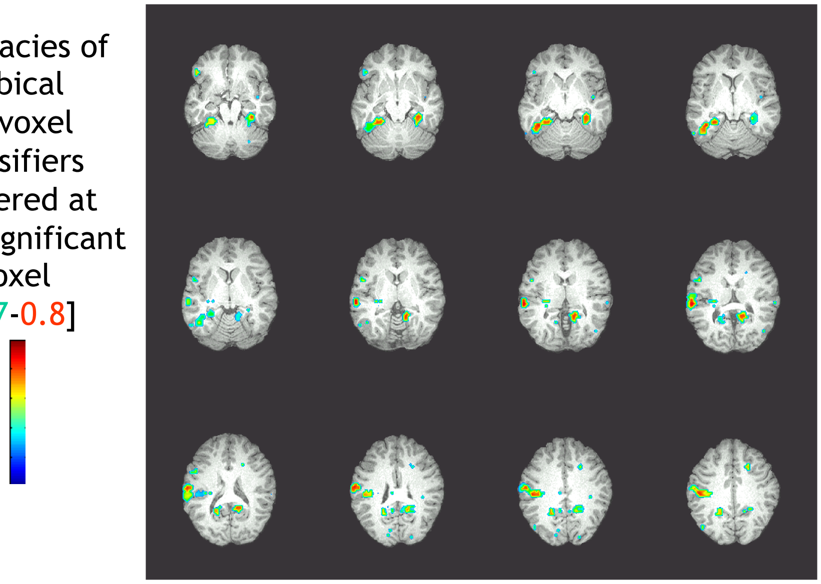
this is a plot of the μ's defining $P(X_i \mid Y=\text{"bottle"})$

Classification task: is person viewing a "tool" or "building"?



Where is information encoded in the brain?

Accuracies of cubical 27-voxel classifiers centered at each significant voxel [0.7-0.8]



Naïve Bayes: What you should know

- Designing classifiers based on Bayes rule
- Conditional independence
  - What it is
  - Why it's important
- Naïve Bayes assumption and its consequences
  - Which (and how many) parameters must be estimated under different generative models (different forms for P(X|Y))
    - and why this matters
- How to train Naïve Bayes classifiers
  - MLE and MAP estimates
  - with discrete and/or continuous inputs X_i

Questions to think about:

- Can you use Naïve Bayes for a combination of discrete and real-valued X_i?
- How can we easily model just 2 of n attributes as dependent?
- What does the decision surface of a Naïve Bayes classifier look like?
- How would you select a subset of X_i's?

Logistic Regression

Required reading:
• Mitchell draft chapter (see course website)

Recommended reading:
• Ng and Jordan paper (see course website)

Machine Learning 10-601

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

September 29, 2011

Logistic Regression

Idea:

- Naïve Bayes allows computing P(Y|X) by learning P(Y) and P(X|Y)

- Why not learn P(Y|X) directly?

• Consider learning f: X → Y, where
  - X is a vector of real-valued features, $< X_1 \ldots X_n >$
  - Y is boolean
  - assume all X_i are conditionally independent given Y
  - model $P(X_i \mid Y = y_k)$ as Gaussian $N(\mu_{ik},\sigma_i)$
  - model P(Y) as Bernoulli (π)

• What does that imply about the form of P(Y|X)?

$$P(Y=1|X=<X_1,...X_n>) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

Derive form for P(Y|X) for continuous X_i

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

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

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

$$= \frac{1}{1 + \exp(\ (\ln \frac{1-\pi}{\pi}) + \boxed{\sum_i \ln \frac{P(X_i|Y=0)}{P(X_i|Y=1)}})}$$

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

$$\boxed{\sum_i \left(\frac{\mu_{i0} - \mu_{i1}}{\sigma_i^2} X_i + \frac{\mu_{i1}^2 - \mu_{i0}^2)}{2\sigma_i^2} \right)}$$

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

Very convenient!

$$P(Y=1|X=<X_1, ...X_n>) = \frac{1}{1 + exp(w_0 + \sum_i w_i X_i)}$$

implies

$$P(Y=0|X=<X_1, ...X_n>) =$$

implies

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

implies

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

Very convenient!

$$P(Y=1|X=<X_1,...X_n>) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

implies

$$P(Y=0|X=<X_1,...X_n>) = \frac{exp(w_0+\sum_i w_i X_i)}{1+exp(w_0+\sum_i w_i X_i)}$$

implies

$$\frac{P(Y=0|X)}{P(Y=1|X)} = exp(w_0+\sum_i w_i X_i)$$

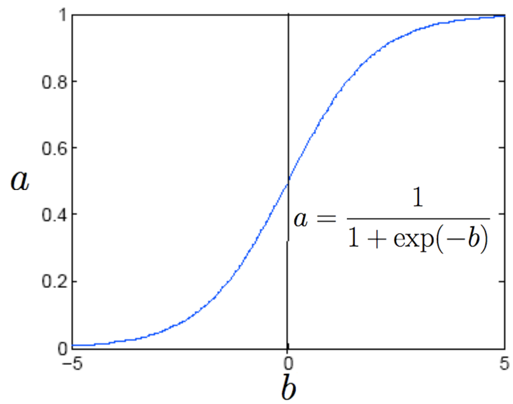
linear classification rule!

implies

$$\ln \frac{P(Y=0|X)}{P(Y=1|X)} = w_0+\sum_i w_i X_i$$

$$\ln \frac{P(Y=0|X)}{P(Y=1|X)} = w_0+\sum_i w_i X_i$$

Logistic function



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

Logistic regression more generally

- Logistic regression when Y not boolean (but still discrete-valued).
- Now $y \in \{y_1 \ldots y_R\}$: learn R-1 sets of weights

for $k<R$ $$P(Y=y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{n} w_{ki} X_i)}{1+\sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)}$$

for $k=R$ $$P(Y=y_R|X) = \frac{1}{1+\sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)}$$

Training Logistic Regression: MCLE

- we have L training examples: $\{\langle X^1, Y^1\rangle, \ldots \langle X^L, Y^L\rangle\}$
- maximum likelihood estimate for parameters W

$$W_{MLE} = \arg\max_W P(< X^1, Y^1 > \ldots < X^L, Y^L > |W)$$
$$= \arg\max_W \prod_l P(< X^l, Y^l > |W)$$

- maximum conditional likelihood estimate

Training Logistic Regression: MCLE

- Choose parameters W=<w_0, ... w_n> to maximize conditional likelihood of training data

where
$$P(Y = 0|X, W) = \frac{1}{1 + exp(w_0 + \sum_i w_i X_i)}$$
$$P(Y = 1|X, W) = \frac{exp(w_0 + \sum_i w_i X_i)}{1 + exp(w_0 + \sum_i w_i X_i)}$$

- Training data D = $\{\langle X^1, Y^1\rangle, \ldots \langle X^L, Y^L\rangle\}$
- Data likelihood = $\prod_l P(X^l, Y^l|W)$
- Data conditional likelihood = $\prod_l P(Y^l|X^l, W)$

$$W_{MCLE} = \arg\max_W \prod_l P(Y^l|W, X^l)$$

Expressing Conditional Log Likelihood

$$l(W) \equiv \ln \prod_l P(Y^l|X^l, W) = \sum_l \ln P(Y^l|X^l, W)$$

$$P(Y=0|X,W) = \frac{1}{1 + exp(w_0 + \sum_i w_i X_i)}$$

$$P(Y=1|X,W) = \frac{exp(w_0 + \sum_i w_i X_i)}{1 + exp(w_0 + \sum_i w_i X_i)}$$

$$\begin{aligned} l(W) &= \sum_l Y^l \ln P(Y^l = 1|X^l, W) + (1 - Y^l) \ln P(Y^l = 0|X^l, W) \\ &= \sum_l Y^l \ln \frac{P(Y^l = 1|X^l, W)}{P(Y^l = 0|X^l, W)} + \ln P(Y^l = 0|X^l, W) \\ &= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + exp(w_0 + \sum_i^n w_i X_i^l)) \end{aligned}$$

Maximizing Conditional Log Likelihood

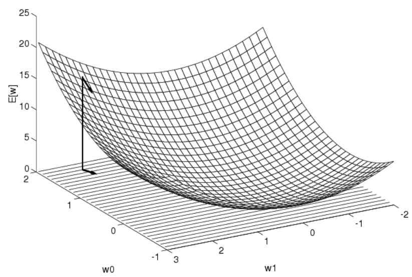
$$P(Y=0|X,W) = \frac{1}{1 + exp(w_0 + \sum_i w_i X_i)}$$

$$P(Y=1|X,W) = \frac{exp(w_0 + \sum_i w_i X_i)}{1 + exp(w_0 + \sum_i w_i X_i)}$$

$$\begin{aligned} l(W) &\equiv \ln \prod_l P(Y^l|X^l, W) \\ &= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + exp(w_0 + \sum_i^n w_i X_i^l)) \end{aligned}$$

Good news: $l(W)$ is concave function of W
Bad news: no closed-form solution to maximize $l(W)$

Gradient Descent



Gradient

$$\nabla E[\vec{w}] \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \cdots \frac{\partial E}{\partial w_n}\right]$$

Training rule:

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}]$$

i.e.,

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

Maximize Conditional Log Likelihood: Gradient Ascent

$$\begin{aligned} l(W) &\equiv \ln \prod_l P(Y^l | X^l, W) \\ &= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + exp(w_0 + \sum_i^n w_i X_i^l)) \end{aligned}$$

$$\frac{\partial l(W)}{\partial w_i} = \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W))$$

Maximize Conditional Log Likelihood: Gradient Ascent

$$l(W) \equiv \ln \prod_l P(Y^l | X^l, W)$$
$$= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + exp(w_0 + \sum_i^n w_i X_i^l))$$

$$\frac{\partial l(W)}{\partial w_i} = \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W))$$

Gradient ascent algorithm: iterate until change < ε

For all i, repeat

$$w_i \leftarrow w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W))$$

That's all for M(C)LE. How about MAP?

- One common approach is to define priors on W
  - Normal distribution, zero mean, identity covariance
- Helps avoid very large weights and overfitting
- MAP estimate

$$W \leftarrow \arg\max_W \ \ln \ P(W) \prod_l P(Y^l | X^l, W)$$

- let's assume Gaussian prior: W ~ N(0, σ)

MLE vs MAP

- Maximum conditional likelihood estimate

$$W \leftarrow \arg\max_W \ \ln \prod_l P(Y^l|X^l, W)$$

$$\boxed{w_i \leftarrow w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1|X^l, W))}$$

- Maximum a posteriori estimate with prior W~N(0,σI)

$$W \leftarrow \arg\max_W \ \ln[P(W) \ \prod_l P(Y^l|X^l, W)]$$

$$\boxed{w_i \leftarrow w_i - \eta\lambda w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1|X^l, W))}$$

MAP estimates and Regularization

- Maximum a posteriori estimate with prior W~N(0,σI)

$$W \leftarrow \arg\max_W \ \ln[P(W) \ \prod_l P(Y^l|X^l, W)]$$

$$\boxed{w_i \leftarrow w_i - \eta\lambda w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1|X^l, W))}$$

called a "<u>regularization</u>" term
• helps reduce overfitting, especially when training data is sparse
• keep weights nearer to zero (if P(W) is zero mean Gaussian prior), or whatever the prior suggests
• used very frequently in Logistic Regression

The Bottom Line

- Consider learning f: X → Y, where
  - X is a vector of real-valued features, < X_1 … X_n >
  - Y is boolean
  - assume all X_i are conditionally independent given Y
  - model P(X_i | Y = y_k) as Gaussian N(μ_{ik},σ_i)
  - model P(Y) as Bernoulli (π)

- Then P(Y|X) is of this form, and we can directly estimate W

$$P(Y = 1|X =< X_1, ...X_n >) = \frac{1}{1 + exp(w_0 + \sum_i w_i X_i)}$$

- Furthermore, same holds if the X_i are boolean
  - trying proving that to yourself

Generative vs. Discriminative Classifiers

Training classifiers involves estimating f: X → Y, or P(Y|X)

Generative classifiers (e.g., Naïve Bayes)
- Assume some functional form for P(X|Y), P(X)
- Estimate parameters of P(X|Y), P(X) directly from training data
- Use Bayes rule to calculate P(Y|X= x_i)

Discriminative classifiers (e.g., Logistic regression)

- Assume some functional form for P(Y|X)
- Estimate parameters of P(Y|X) directly from training data

Use Naïve Bayes or Logisitic Regression?

Consider

- Restrictiveness of modeling assumptions

- Rate of convergence toward asymptotic hypothesis
  - How does increasing number of features n influence need for larger training set?

Naïve Bayes vs Logistic Regression

Consider Y boolean, X_i continuous, $X=<X_1 \ldots X_n>$

Number of parameters to estimate:

- NB:

- LR:

$$P(Y=0|X,W) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

$$P(Y=1|X,W) = \frac{exp(w_0+\sum_i w_i X_i)}{1+exp(w_0+\sum_i w_i X_i)}$$

Naïve Bayes vs Logistic Regression

Consider Y boolean, X_i continuous, $X=<X_1 \dots X_n>$

Number of parameters:

- NB: 4n +1
- LR: n+1

Estimation method:

- NB parameter estimates are uncoupled
- LR parameter estimates are coupled

G.Naïve Bayes vs. Logistic Regression

[Ng & Jordan, 2002]

- Generative and Discriminative classifiers

- *Asymptotic comparison* (# training examples → infinity)
  - when conditional independence assumptions correct
    - GNB, LR produce identical classifiers
  - when conditional independence assumptions incorrect
    - LR is less biased – does not assume cond indep.
    - therefore expected to outperform GNB when both given infinite training data

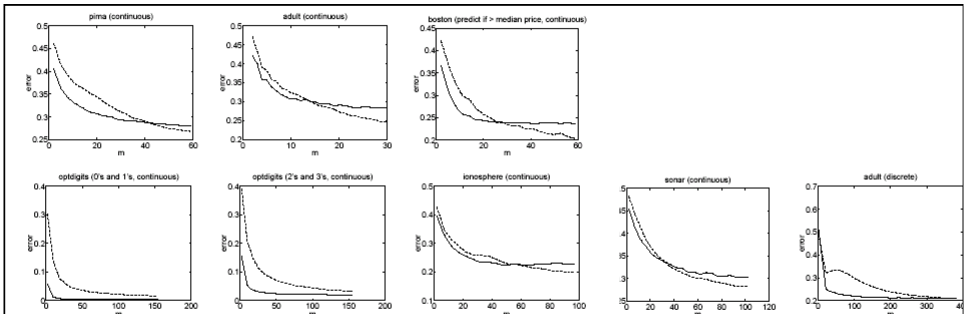
Naïve Bayes vs. Logistic Regression

- Generative and Discriminative classifiers

- *Non-asymptotic analysis* (see [Ng & Jordan, 2002])
    - convergence rate of parameter estimates – how many training examples needed to assure good estimates?
        - GNB order log n (where n = # of attributes in X)
        - LR order n
        - both scale as 1/sqrt(m) (where m = # train examples)

→ GNB converges more quickly to its (perhaps less accurate) asymptotic estimates

→ Why: Because LR's parameter estimates are coupled, but GNB's are not



Some experiments from UCI data sets

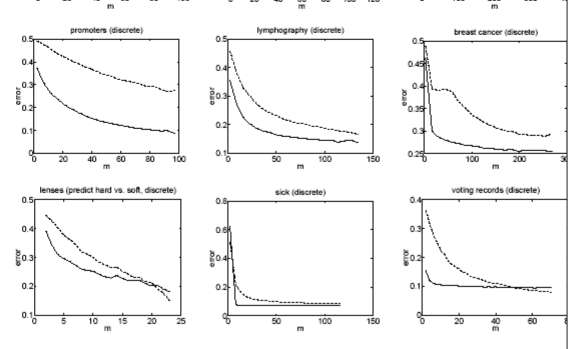


Figure 1: Results of 15 experiments on datasets from the UCI Machine Learnin repository. Plots are of generalization error vs. m (averaged over 1000 rando train/test splits). Dashed line is logistic regression; solid line is naive Bayes.

Summary: Naïve Bayes and Logistic Regression

- Modeling assumptions
  - Naïve Bayes more biased (cond. indep)
  - Both learn linear decision surfaces
- Convergence rate (n=number features in <X_1 … X_n>)
  - Naïve Bayes requires O(log n) examples
  - Logistic regression requires O(n) examples
- Bottom line
  - Naïve Bayes converges faster to its (potentially too restricted) final hypothesis

What you should know:

- Logistic regression
  - Functional form follows from Naïve Bayes assumptions
    - For Gaussian Naïve Bayes assuming variance $\sigma_{i,k} = \sigma_i$
    - For discrete-valued Naïve Bayes too
  - But training procedure picks parameters without making conditional independence assumption
  - MLE training: pick W to maximize P(Y | X, W)
  - MAP training: pick W to maximize P(W | X,Y)
    - 'regularization'
    - helps reduce overfitting

- Gradient ascent/descent
  - General approach when closed-form solutions unavailable

- Generative vs. Discriminative classifiers
  - Bias vs. variance tradeoff