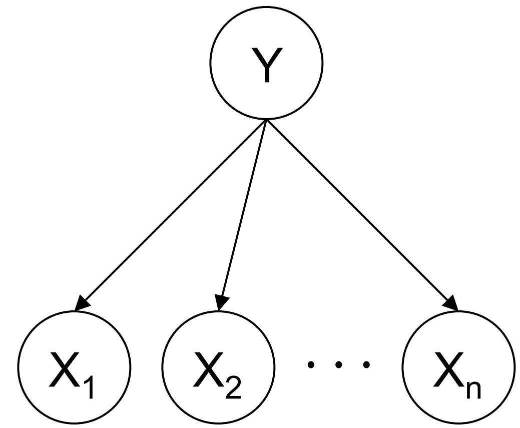
# Introduction to Bayesian methods (continued) - Lecture 17

David Sontag
New York University

Slides adapted from Luke Zettlemoyer, Carlos Guestrin, Dan Klein, and Vibhav Gogate

# The Naïve Bayes Classifier

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




- Decision rule:

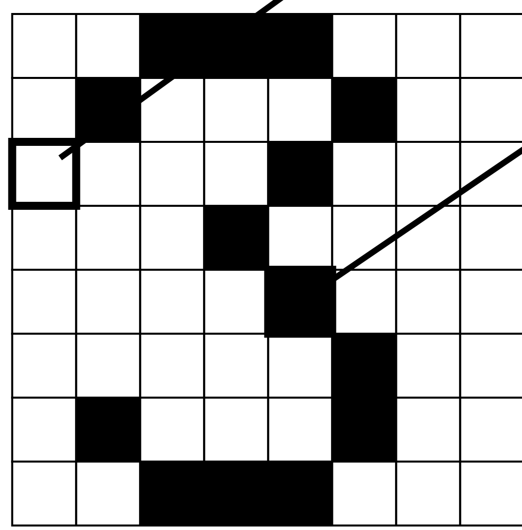
$$y^* = h_{NB}(\mathbf{x}) = \arg\max_y P(y)P(x_1, \ldots, x_n \mid y)$$
$$= \arg\max_y P(y)\prod_i P(x_i|y)$$

If certain assumption holds, NB is optimal classifier!
(they typically don't)

# What has to be learned?

$P(Y)$

| | |
|---|---|
| 1 | 0.1 |
| 2 | 0.1 |
| 3 | 0.1 |
| 4 | 0.1 |
| 5 | 0.1 |
| 6 | 0.1 |
| 7 | 0.1 |
| 8 | 0.1 |
| 9 | 0.1 |
| 0 | 0.1 |



$P(F_{3,1} = on|Y)$

| | |
|---|---|
| 1 | 0.01 |
| 2 | 0.05 |
| 3 | 0.05 |
| 4 | 0.30 |
| 5 | 0.80 |
| 6 | 0.90 |
| 7 | 0.05 |
| 8 | 0.60 |
| 9 | 0.50 |
| 0 | 0.80 |

$P(F_{5,5} = on|Y)$

| | |
|---|---|
| 1 | 0.05 |
| 2 | 0.01 |
| 3 | 0.90 |
| 4 | 0.80 |
| 5 | 0.90 |
| 6 | 0.90 |
| 7 | 0.25 |
| 8 | 0.85 |
| 9 | 0.60 |
| 0 | 0.80 |

# MLE for the parameters of NB

- Given dataset
  - Count(A=a,B=b) $\leftarrow$ number of examples where A=a and B=b
- MLE for discrete NB, simply:
  - Prior:

$$P(Y=y) = \frac{Count(Y=y)}{\sum_{y'} Count(Y=y')}$$

  - Observation distribution:

$$P(X_i=x|Y=y) = \frac{Count(X_i=x, Y=y)}{\sum_{x'} Count(X_i=x', Y=y)}$$

# What about if there is missing data?

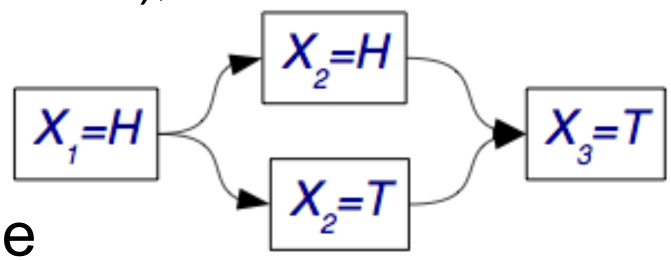
- One of the key strengths of Bayesian approaches is that they can naturally handle missing data
- Suppose don't have value for some attribute $X_i$
  - applicant's credit history unknown
  - some medical test not performed on patient
  - how to compute $P(X_1=x_1 \ldots X_j=? \ldots X_d=x_d \mid y)$
- Easy with Naïve Bayes
  - ignore attribute in instance where its value is missing
  - compute likelihood based on observed attributes
  - no need to "fill in" or explicitly model missing values
  - based on conditional independence between attributes

$$P(x_1 \ldots \boxed{X_j} \ldots x_d | y) = \prod_{\boxed{i \neq j}}^{d} P(x_i | y)$$

[Slide from Victor Lavrenko and Nigel Goddard]

# What about if there is missing data?

- Ex: three coin tosses: Event = {$X_1$=H, $X_2$=?, $X_3$=T}
  - event = head, unknown (either head or tail), tail
  - event = {H,H,T} + {H,T,T}
  - *P*(event) = P(H,H,T) + P(H,T,T)




- General case: $X_j$ has missing value

$$P(x_1 \ldots \boxed{x_j} \ldots x_d | y) = P(x_1|y) \cdots \boxed{P(x_j|y)} \cdots P(x_d|y)$$

$$\sum_{\boxed{x_j}} P(x_1 \ldots \boxed{x_j} \ldots x_d|y) = \sum_{x_j} P(x_1|y)\cdots \boxed{P(x_j|y)} \cdots P(x_d|y)$$

$$= P(x_1|y)\cdots \boxed{\left[\sum_{x_j} P(x_j|y)\right]} \cdots P(x_d|y)$$

$$= P(x_1|y)\cdots \boxed{1}\cdots P(x_d|y)$$

[Slide from Victor Lavrenko and Nigel Goddard]

# Naive Bayes = Linear Classifier

- **Theorem**: assume that $x_i \in \{0, 1\}$ for all $i \in [1, N]$. Then, the Naive Bayes classifier is defined by

$$\mathbf{x} \mapsto \operatorname{sgn}(\mathbf{w} \cdot \mathbf{x} + b),$$

[Slide from Mehyrar Mohri]

# Outline of lectures

- Review of probability
- Maximum likelihood estimation

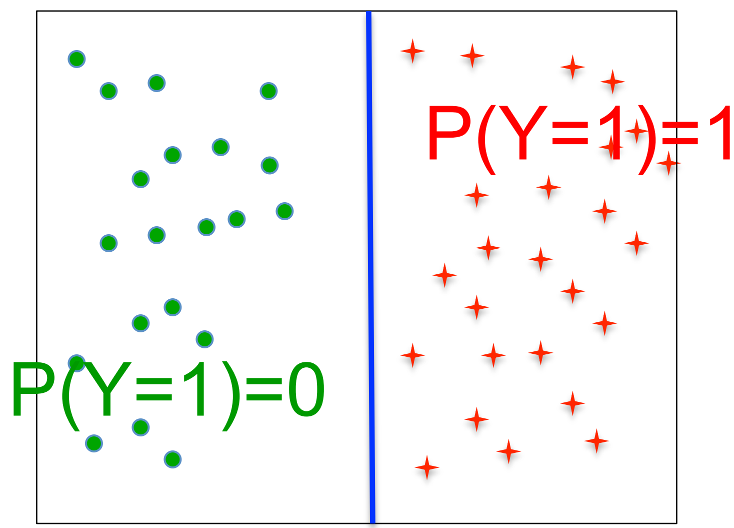
2 examples of Bayesian classifiers:

- Naïve Bayes
- **Logistic regression**

[Next several slides adapted from:
Vibhav Gogate, Luke Zettlemoyer, Carlos Guestrin, and Dan Weld]

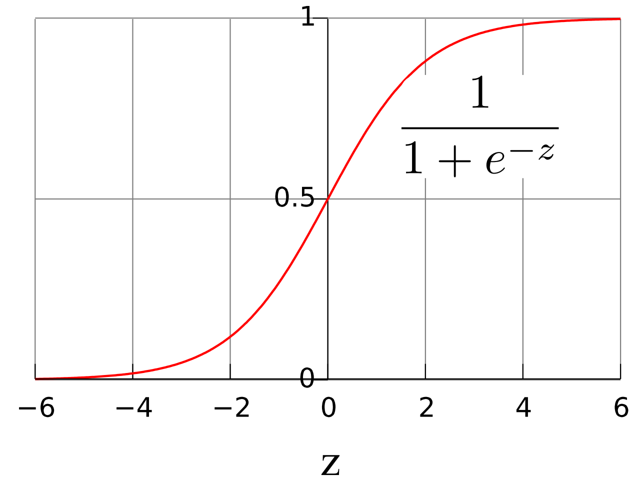
# Logistic Regression

- Learn P(Y|**X**) directly!
  - Assume a particular functional form
  - Linear classifier? On one side we say P(Y=1|X)=1, and on the other P(Y=1|X)=0
  - ***But, this is not differentiable (hard to learn)… doesn't allow for label noise...***



# Logistic Regression

Logistic function (Sigmoid):




- Learn P(Y|**X**) directly!
  - Assume a particular functional form
  - Sigmoid applied to a linear function of the data:

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

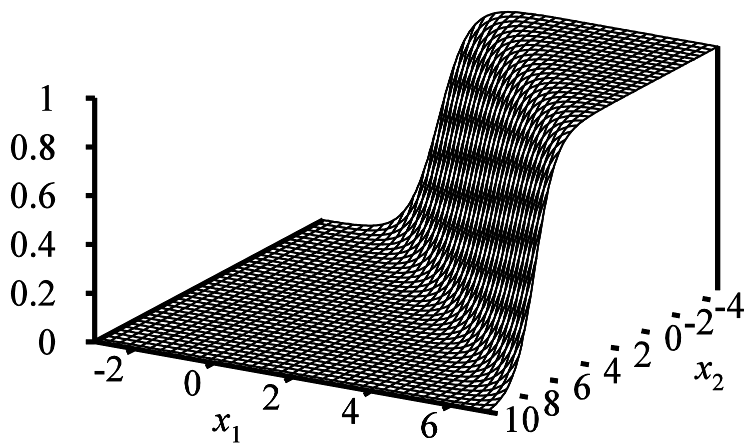
$$P(Y=0|X) = \frac{\exp(w_0+\sum_{i=1}^{n} w_i X_i)}{1+\exp(w_0+\sum_{i=1}^{n} w_i X_i)}$$

**Features can be discrete or continuous!**

# Logistic Function in n Dimensions

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

Sigmoid applied to a linear function of the data:




**Features can be discrete or continuous!**

# Logistic Regression: decision boundary

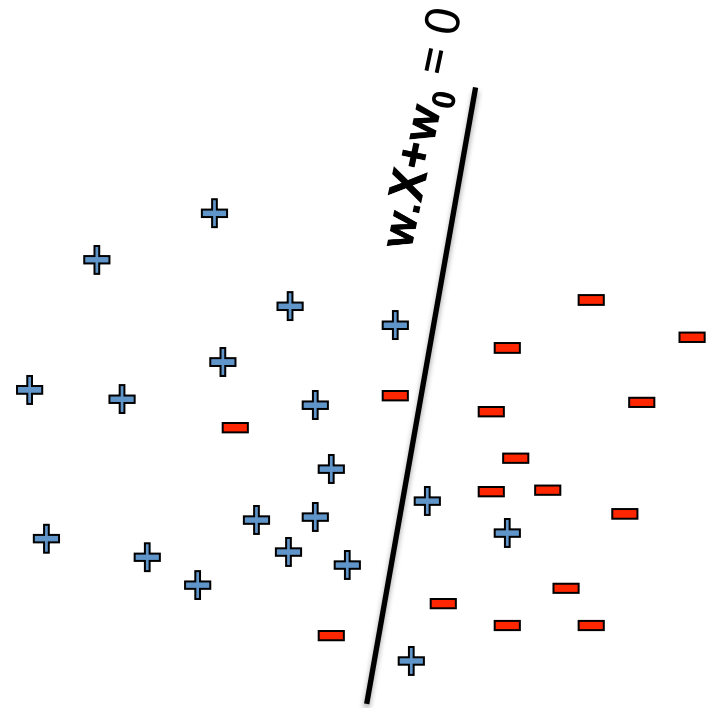
$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_{i=1}^n w_i X_i)} \qquad P(Y=0|X) = \frac{\exp(w_0 + \sum_{i=1}^n w_i X_i)}{1+\exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

- Prediction: Output the Y with highest P(Y|X)
  - For binary Y, output Y=0 if

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

$$1 < \exp(w_0 + \sum_{i=1}^n w_i X_i)$$

$$0 < w_0 + \sum_{i=1}^n w_i X_i$$




A Linear Classifier!

# Likelihood vs. Conditional Likelihood

Generative (Naïve Bayes) maximizes **Data likelihood**

$$\begin{aligned} \ln P(\mathcal{D} \mid \mathbf{w}) &= \sum_{j=1}^{N} \ln P(\mathbf{x}^j, y^j \mid \mathbf{w}) \\ &= \sum_{j=1}^{N} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w}) + \sum_{j=1}^{N} \ln P(\mathbf{x}^j \mid \mathbf{w}) \end{aligned}$$

Discriminative (Logistic Regr.) maximizes **Conditional Data Likelihood**

$$\ln P(\mathcal{D}_Y \mid \mathcal{D}_{\mathbf{X}}, \mathbf{w}) = \sum_{j=1}^{N} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w})$$

Focuses only on learning P(Y|**X**) - all that matters for classification

# Maximizing Conditional Log Likelihood

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

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

$$\begin{aligned} l(\mathbf{w}) &\equiv \ln \prod_j P(y^j|\mathbf{x}^j,\mathbf{w}) \\ &= \sum_j y^j(w_0+\sum_i^n w_i x_i^j) - \ln(1+exp(w_0+\sum_i^n w_i x_i^j)) \end{aligned}$$

**0 or 1!**

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

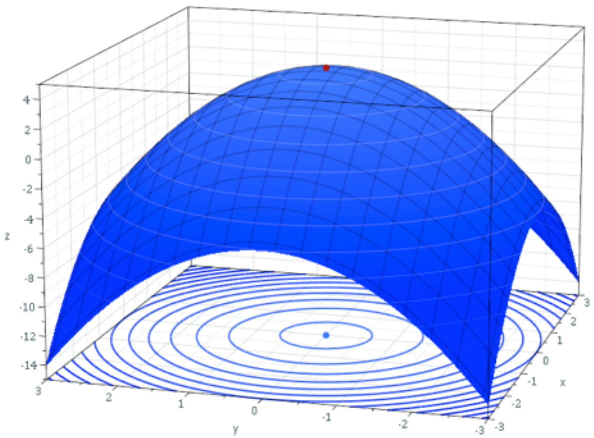
Good news: *l*(**w**) is concave function of **w**$\rightarrow$

No local maxima

Concave functions easy to optimize

# Optimizing concave function – Gradient ascent

- Conditional likelihood for Logistic Regression is concave $\rightarrow$



Gradient: $$\nabla_{\mathbf{w}} l(\mathbf{w}) = [\frac{\partial l(\mathbf{w})}{\partial w_0}, \ldots, \frac{\partial l(\mathbf{w})}{\partial w_n}]'$$

Learning rate, η>0

Update rule: $$\Delta \mathbf{w} = \eta \nabla_{\mathbf{w}} l(\mathbf{w})$$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \frac{\partial l(\mathbf{w})}{\partial w_i}$$

# Maximize Conditional Log Likelihood: Gradient ascent

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

$$l(\mathbf{w}) \;=\; \sum_j y^j (w_0 + \sum_i^n w_i x_i^j) - \ln(1 + exp(w_0 + \sum_i^n w_i x_i^j))$$

$$\frac{\partial l(w)}{\partial w_i} = \sum_j \left[ \frac{\partial}{\partial w_i} y^j (w_0 + \sum_i w_i x_i^j) - \frac{\partial}{\partial w_i} \ln \left( 1 + \exp(w_0 + \sum_i w_i x_i^j) \right) \right]$$

$$= \sum_j \left[ y^j x_i^j - \frac{x_i^j \exp(w_0 + \sum_i w_i x_i^j)}{1 + \exp(w_0 + \sum_i w_i x_i^j)} \right]$$

$$= \sum_j x_i^j \left[ y^j - \frac{\exp(w_0 + \sum_i w_i x_i^j)}{1 + \exp(w_0 + \sum_i w_i x_i^j)} \right]$$

$$\boxed{\frac{\partial l(w)}{\partial w_i} = \sum_j x_i^j \left( y^j - P(Y^j = 1 | x^j, w) \right)}$$

# Gradient Ascent for LR

Gradient ascent algorithm: (learning rate $\eta$ > 0)

do:

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \sum_j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})]$$

For i=1 to n: (iterate over features)

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w})]$$

until "change" < ε

Loop over training examples
(could also do stochastic GD)

# That's all MLE. How about MAP?

$$p(\mathbf{w} \mid Y, \mathbf{X}) \;\propto\; P(Y \mid \mathbf{X}, \mathbf{w})p(\mathbf{w})$$

- One common approach is to define priors on **w**
  - Normal distribution, zero mean, identity covariance
  - "Pushes" parameters towards zero

$$p(\mathbf{w}) = \prod_i \frac{1}{\kappa\sqrt{2\pi}} \; e^{\frac{-w_i^2}{2\kappa^2}}$$

- ***Regularization***
  - Helps avoid very large weights and overfitting

- MAP estimate:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln\left[p(\mathbf{w})\prod_{j=1}^{N} P(y^j \mid \mathbf{x}^j, \mathbf{w})\right]$$

# MAP as Regularization

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[ p(\mathbf{w}) \prod_{j=1}^{N} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right] \qquad p(\mathbf{w}) = \prod_i \frac{1}{\kappa\sqrt{2\pi}} \; e^{\frac{-w_i^2}{2\kappa^2}}$$

- Adds log p(w) to objective:

$$\ln p(w) \propto -\frac{\lambda}{2} \sum_i w_i^2 \qquad\qquad \frac{\partial \ln p(w)}{\partial w_i} = -\lambda w_i$$

  - Quadratic penalty: drives weights towards zero
  - Adds a negative linear term to the gradients

**Quadratic penalty on weights, just like with SVMs!**

$$\ell(D;\vec{w}) = \sum_{i=1}^{N} \log\left(\frac{1}{1+e^{-y_i(w_0+\vec{w}\cdot\vec{x}_i)}}\right)$$

$y_i \in \{+1,-1\}$

MAP estimation of LR:

$$\max_{\vec{w}} \sum_{i=1}^{N} \lg\left(\frac{1}{1+e^{-y_i(\vec{w}\cdot\vec{x}_i)}}\right) - \lambda\|\vec{w}\|^2$$

$$\|\|$$

$$\min_{\vec{w}} \underbrace{\sum_{i=1}^{N} \lg\left(1+e^{-y_i(\vec{w}\cdot\vec{x}_i)}\right)}_{\text{logit loss}} + \lambda\|w\|^2$$

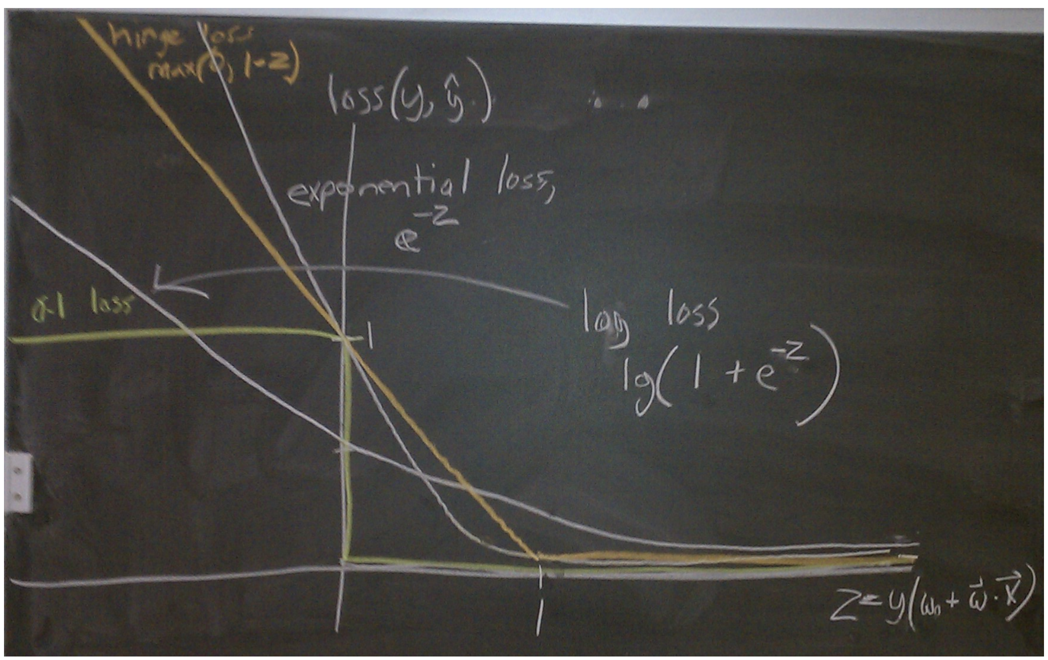
$$y_i(\vec{w}\cdot\vec{x}_i) \geq 1-\xi_i$$

SVM

$$\min_{\vec{w}} \underbrace{\sum_{i=1}^{N} \max\left(0,\, 1-y_i(\vec{w}\cdot\vec{x}_i)\right)}_{\text{hinge loss}} + \lambda\|w\|^2$$

Learning:

$$\min_{\vec{w}} \frac{1}{N}\sum_{i=1}^{N} L\left(y_i,\, \hat{y}(\vec{x}_i,\vec{w})\right) + R(\vec{w})$$



# Naïve Bayes *vs.* Logistic Regression

**Learning**: h:**X** $\mapsto$ Y

**X** – features

Y – target classes

**Generative**

- Assume functional form for
  - P(X|Y) **assume cond indep**
  - P(Y)
  - Est. params from train data
- Gaussian NB for cont. features
- Bayes rule to calc. P(Y|X= x):
  - P(Y | **X**) $\propto$ P(**X** | Y) P(Y)
- **Indirect** computation
  - Can generate a sample of the data
  - **Can easily handle missing data**

**Discriminative**

- Assume functional form for
  - P(Y|X) **no assumptions**
  - Est params from training data
- Handles discrete & cont features
- Directly calculate P(Y|X=x)
  - Can't generate data sample

# Naïve Bayes vs. Logistic Regression

[Ng & Jordan, 2002]

- Generative vs. Discriminative classifiers
- Asymptotic comparison
  (# training examples → infinity)
  - when model correct
    - NB, Linear Discriminant Analysis (with class independent variances), and Logistic Regression produce identical classifiers
  - when model incorrect
    - LR is less biased – does not assume conditional independence
      - **therefore LR expected to outperform NB**

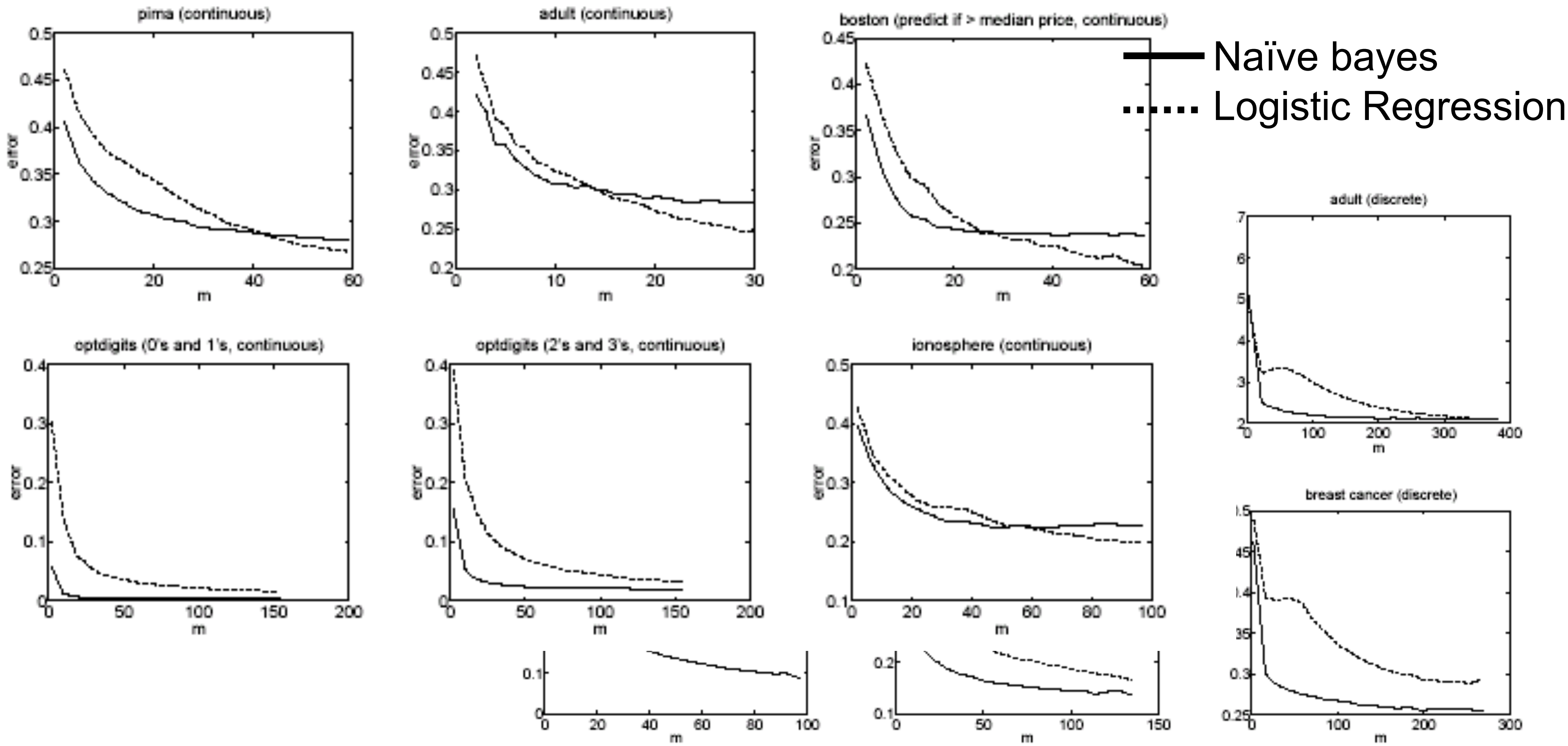
# Naïve Bayes vs. Logistic Regression

[Ng & Jordan, 2002]

- Generative vs. Discriminative classifiers
- Non-asymptotic analysis
  - convergence rate of parameter estimates,

    (**n = # of attributes in X**)
    - Size of training data to get close to infinite data solution
    - Naïve Bayes needs $O(\log n)$ samples
    - Logistic Regression needs $O(n)$ samples

  - Naïve Bayes converges more quickly to its ***(perhaps less helpful)*** asymptotic estimates




# Some experiments from UCI data sets

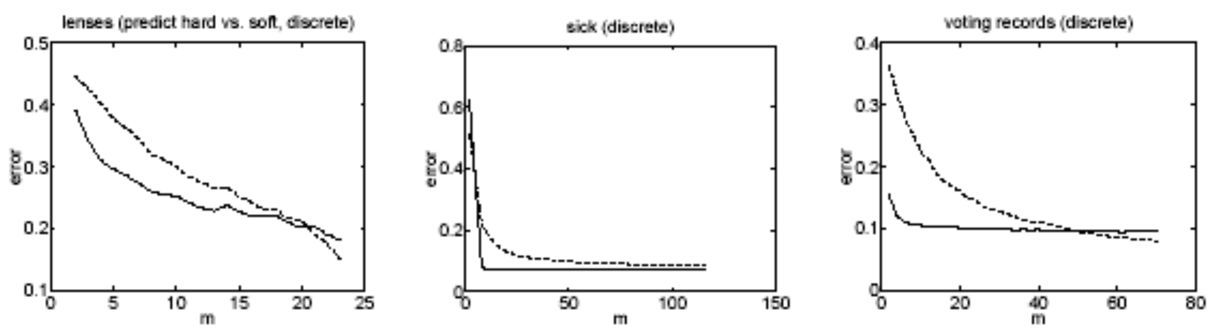



Figure 1: Results of 15 experiments on datasets from the UCI Machine Learning repository. Plots are of generalization error vs. $m$ (averaged over 1000 random train/test splits). Dashed line is logistic regression; solid line is naive Bayes.

# Logistic regression for discrete classification

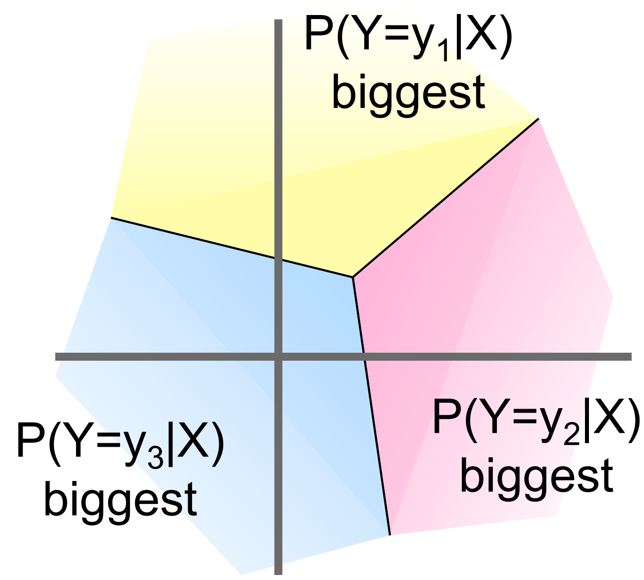
Logistic regression in more general case, where set of possible $Y$ is $\{y_1,\ldots,y_R\}$

- Define a weight vector $w_i$ for each $y_i$, $i=1,\ldots,R$

$$P(Y=1|X) \propto \exp(w_{10} + \sum_i w_{1i} X_i)$$

$$P(Y=2|X) \propto \exp(w_{20} + \sum_i w_{2i} X_i)$$

$$\ldots$$




- Also called "soft-max" loss