Logistic Regression

Naïve Bayes Review

- Optimal Classifier: $f^*(x) = \arg\max_y P(y|x)$
- NB Assumption: $P(X_1 ... X_d|Y) = \prod_{i=1}^{d} P(X_i|Y)$
- NB Classifier:

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

- Assume parametric form for $P(X_i|Y)$ and $P(Y)$
  - Estimate parameters using MLE/MAP and plug in

Generative vs. Discriminative Classifiers

Generative classifiers (e.g. Naïve Bayes)

- Assume some functional form for P(X,Y) (or P(X|Y) and P(Y))
- Estimate parameters of P(X|Y), P(Y) directly from training data
- Use Bayes rule to calculate P(Y|X)

Why not learn P(Y|X) directly? Or better yet, why not learn the decision boundary directly?

Discriminative classifiers (e.g. Logistic Regression)

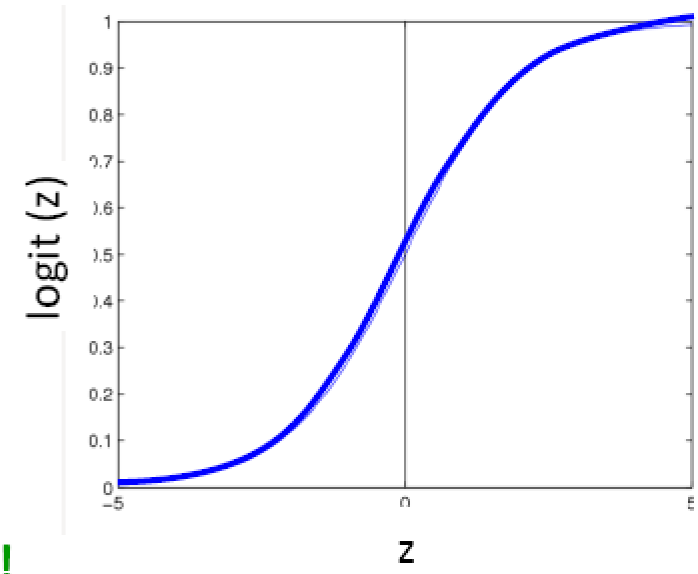
- Assume some functional form for P(Y|X) or for the decision boundary
- Estimate parameters of P(Y|X) directly from training data

Logistic Regression

Assumes the following functional form for P(Y|X):

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

Logistic function applied to a linear function of the data



Logistic function (or Sigmoid): $\frac{1}{1+exp(-z)}$

Features can be discrete or continuous!

Is Logistic Regression a Linear Classifier?

Assumes the following functional form for P(Y|X):

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

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

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

$$\Rightarrow \boxed{w_0 + \sum_i w_i X_i \quad \overset{0}{\underset{1}{\gtrless}} \quad 0}$$

Logistic Regression is a Linear Classifier!

Assumes the following functional form for P(Y|X):

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

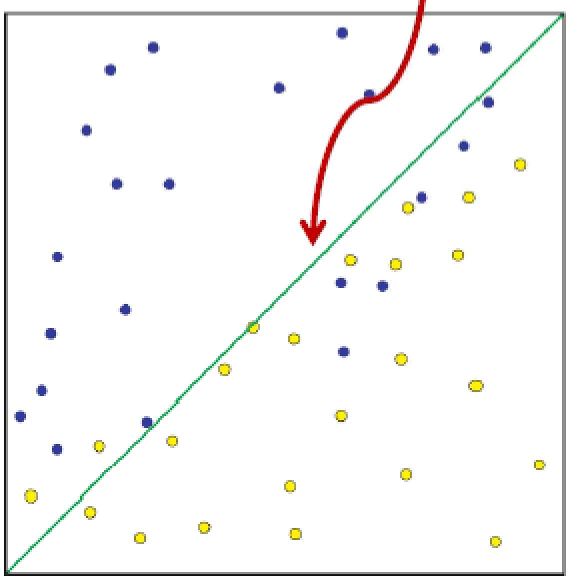
Decision boundary:

$$w_0 + \sum_i w_i X_i = 0$$

$$P(Y=0|X) \overset{0}{\underset{1}{\gtrless}} P(Y=1|X)$$

$$w_0 + \sum_i w_i X_i \overset{0}{\underset{1}{\gtrless}} 0$$

(Linear Decision Boundary)



Logistic Regression for more than 2 classes

- Logistic regression in more general case, where $Y \in \{y_1,\dots,y_K\}$

for $k<K$

$$P(Y = y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{d} w_{ki}X_i)}{1 + \sum_{j=1}^{K-1} \exp(w_{j0} + \sum_{i=1}^{d} w_{ji}X_i)}$$

for $k=K$ (normalization, so no weights for this class)

$$P(Y = y_K|X) = \frac{1}{1 + \sum_{j=1}^{K-1} \exp(w_{j0} + \sum_{i=1}^{d} w_{ji}X_i)}$$

Is the decision boundary still linear?

Training Logistic Regression

We'll focus on binary classification:

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

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

How to learn the parameters w_0, w_1, … w_d?

Training Data $\{(X^{(j)}, Y^{(j)})\}_{j=1}^{n}$ $\quad X^{(j)} = (X_1^{(j)}, \ldots, X_d^{(j)})$

Maximum Likelihood Estimates

$$\widehat{\mathbf{w}}_{MLE} = \arg\max_{\mathbf{w}} \prod_{j=1}^{n} P(X^{(j)}, Y^{(j)} \mid \mathbf{w})$$

But there is a problem …

Don't have a model for P(X) or P(X|Y) – only for P(Y|X)

Training Logistic Regression

How to learn the parameters w_0, w_1, … w_d?

Training Data $\{(X^{(j)}, Y^{(j)})\}_{j=1}^{n}$ $X^{(j)} = (X_1^{(j)}, \ldots, X_d^{(j)})$

Maximum (Conditional) Likelihood Estimates

$$\widehat{\mathbf{w}}_{MCLE} = \arg\max_{\mathbf{w}} \prod_{j=1}^{n} P(Y^{(j)} \mid X^{(j)}, \mathbf{w})$$

Expressing Conditional Log Likelihood

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

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

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

Maximizing Conditional Log Likelihood

$$\max_{\mathbf{w}} \; l(\mathbf{w}) \equiv \ln \prod_j P(y^j | \mathbf{x}^j, \mathbf{w})$$

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

Good news: $l(\mathbf{w})$ is concave function of $\mathbf{w}$ $\rightarrow$ no locally optimal solutions

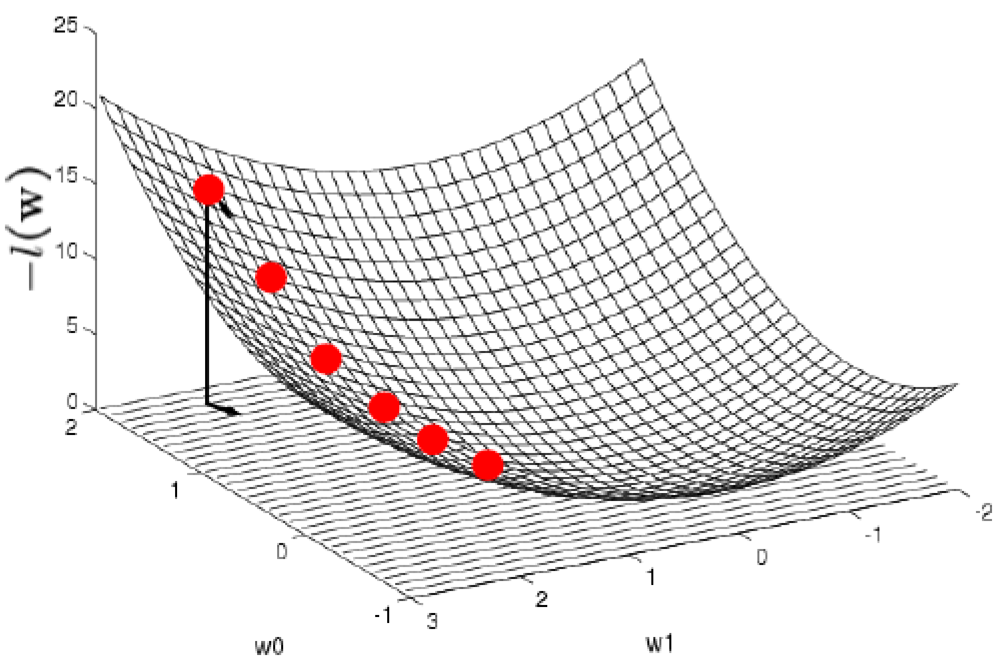
Bad news: no closed-form solution to maximize $l(\mathbf{w})$

Good news: concave functions easy to optimize (unique maximum)

Optimizing concave/convex function

- Conditional likelihood for Logistic Regression is concave
- Maximum of a concave function = minimum of a convex function

Gradient Ascent (concave)/ Gradient Descent (convex)



Gradient:

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

Update rule:

Learning rate, η>0

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

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left. \frac{\partial l(\mathbf{w})}{\partial w_i} \right|_t$$

Gradient Ascent for Logistic Regression

Gradient ascent algorithm: iterate until change < ε

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

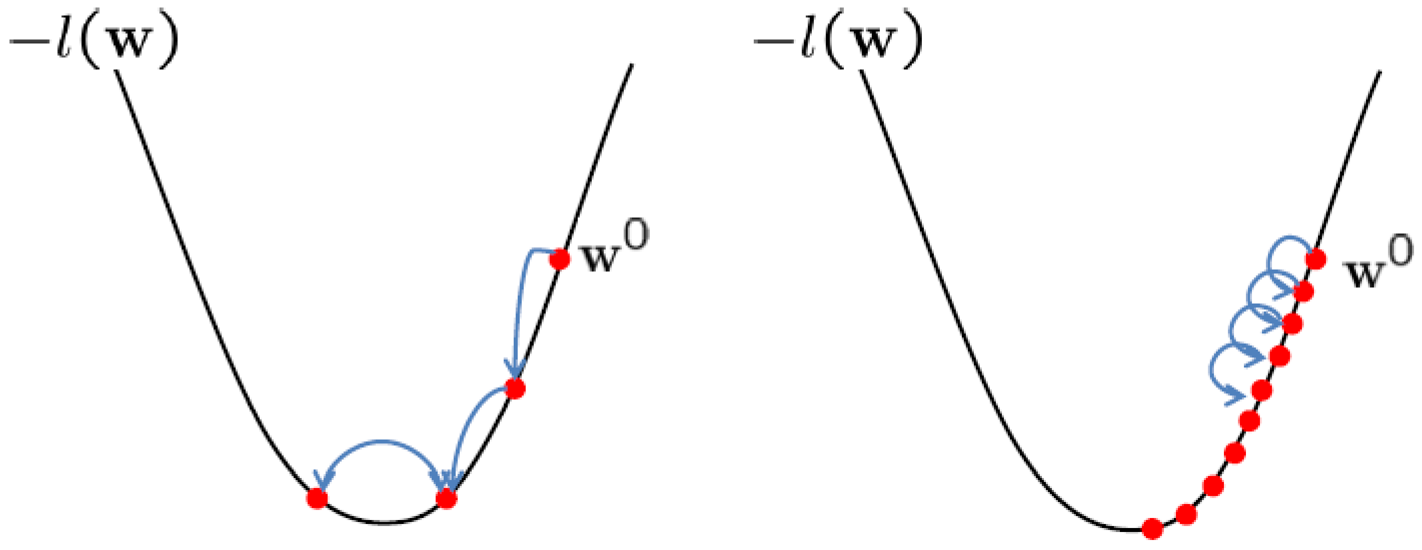
For i=1,…,d,

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \underbrace{\hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})}_{\text{Predict what current weight thinks label Y should be}}]$$

repeat

- Gradient ascent is simplest of optimization approaches
  - e.g., Newton method, Conjugate gradient ascent, IRLS (see Bishop 4.3.3)

Effect of step-size η



Large η => Fast convergence but larger residual error
Also possible oscillations

Small η => Slow convergence but small residual error

That's all M(C)LE. 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
- Corresponds to ***Regularization***
  - Helps avoid very large weights and overfitting
  - More on this later in the semester

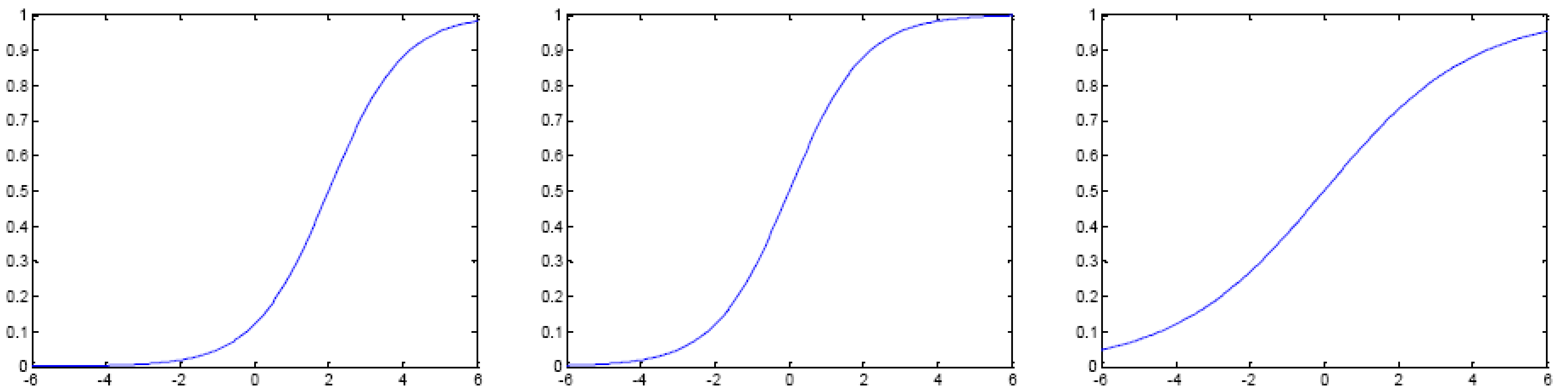
- M(C)AP estimate

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

Understanding the sigmoid

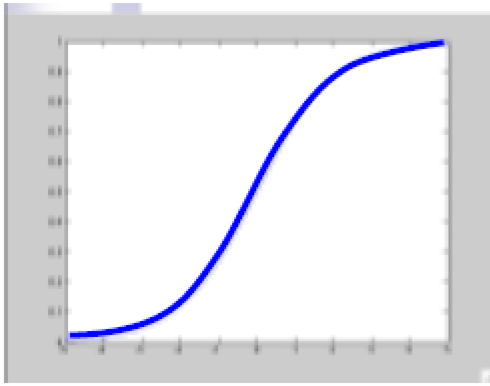
$$g(w_0 + \sum_i w_i x_i) \quad = \quad \frac{1}{1 + e^{w_0 + \sum_i w_i x_i}}$$

w_0=-2, w_1=-1 $\qquad$ w_0=0, w_1=-1 $\qquad$ w_0=0, w_1=-0.5

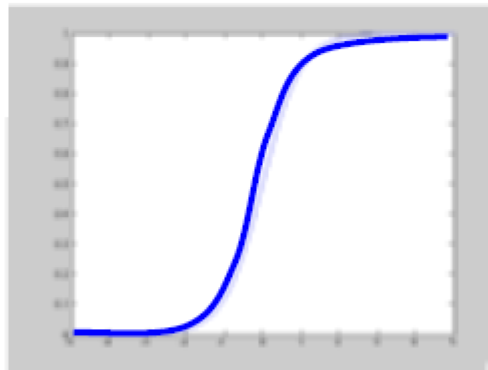


$$z = w_0 + \sum_i w_i x_i$$

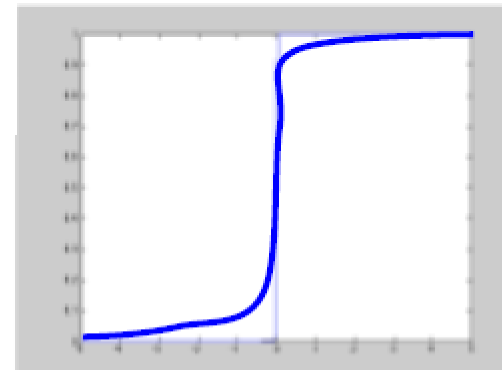
Large weights → Overfitting



$$\frac{1}{1+e^{-x}}$$

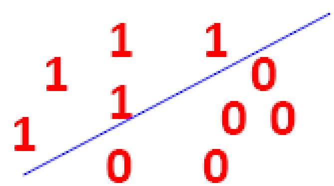


$$\frac{1}{1+e^{-2x}}$$



$$\frac{1}{1+e^{-100x}}$$

- Large weights lead to overfitting:



- Penalizing high weights can prevent overfitting…
  – again, more on this later in the semester

M(C)AP - Regularization

- Regularization

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

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

Zero-mean Gaussian prior

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{j=1}^{n} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w}) - \underbrace{\sum_{i=1}^{d} \frac{w_i^2}{2\kappa^2}}_{\boxed{\text{Penalizes large weights}}}$$

M(C)AP - Gradient

- Gradient

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

Zero-mean Gaussian prior

$$\frac{\partial}{\partial w_i} \ln \left[p(\mathbf{w}) \prod_{j=1}^{n} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$\underbrace{\frac{\partial}{\partial w_i} \ln p(\mathbf{w})}_{\propto \frac{-w_i}{\kappa^2}} + \underbrace{\frac{\partial}{\partial w_i} \ln \left[\prod_{j=1}^{n} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]}_{\text{Same as before}}$$

Extra term Penalizes large weights

M(C)LE vs. M(C)AP

- Maximum conditional likelihood estimate

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

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

- Maximum conditional a posteriori estimate

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

$$\boxed{w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left\{ -\frac{1}{\kappa^2} w_i^{(t)} + \sum_j x_i^j [y^j - P(Y = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})] \right\}}$$

Connection to Gaussian Naïve Bayes

There are several distributions that can lead to a linear decision boundary.

As another example, consider a generative model (GNB):

$$Y \sim \text{Bernoulli}(\pi)$$

$$P(X_i|Y=y) = \frac{1}{\sqrt{2\pi\sigma_{i,y}^2}}\, e^{\frac{-(X_i-\mu_{i,y})^2}{2\sigma_{i,y}^2}}$$

Gaussian class conditional densities

Assume variance is independent of class, i.e. $\sigma_{i,0}^2 = \sigma_{i,1}^2$

Connection to Gaussian Naïve Bayes

$$P(X_i|Y=y) = \frac{1}{\sqrt{2\pi\sigma_i^2}}\, e^{\frac{-(X_i-\mu_{i,y})^2}{2\sigma_i^2}}$$

Using conditionally independent assumption,

$$\log\frac{P(X|Y=0)}{P(X|Y=1)} = \log\prod_{i=1}^{d}\frac{P(X_i|Y=0)}{P(X_i|Y=1)}$$

Decision boundary:

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

$$= \underbrace{\log\frac{1-\pi}{\pi} + \sum_i \frac{\mu_{i,1}^2-\mu_{i,0}^2}{2\sigma_i^2}}_{\text{Constant term}} + \underbrace{\sum_i \frac{\mu_{i,0}-\mu_{i,1}}{\sigma_i^2}X_i}_{\text{First-order term}} =: w_0 + \sum_i w_i X_i$$

Gaussian Naïve Bayes vs. Logistic Regression

Set of Gaussian Naïve Bayes parameters (feature variance independent of class label)	⟺	Set of Logistic Regression parameters

- Representation equivalence
  - **But only in a special case!!!** (GNB with class-independent variances)
- But what's the difference???
- **LR makes no assumptions about** P(**X**|Y) **in learning**!!!
- **Loss function!!!**
  - Optimize different functions $\rightarrow$ Obtain different solutions

Naïve Bayes vs. Logistic Regression

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

Number of parameters:

- NB: 4d +1 $\pi, (\mu_{1,y}, \mu_{2,y}, \ldots, \mu_{d,y}), (\sigma^2_{1,y}, \sigma^2_{2,y}, \ldots, \sigma^2_{d,y})$ y = 0,1
- LR: d+1 $w_0, w_1, \ldots, w_d$

Estimation method:

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

Generative vs. Discriminative

[Ng & Jordan, NIPS 2001]

Given **infinite data** (asymptotically),

If conditional independence assumption holds,
Discriminative and generative NB perform similar.

$$\epsilon_{\mathrm{Dis},\infty} \sim \epsilon_{\mathrm{Gen},\infty}$$

If conditional independence assumption does NOT holds,
Discriminative outperforms generative NB.

$$\epsilon_{\mathrm{Dis},\infty} < \epsilon_{\mathrm{Gen},\infty}$$

Generative vs. Discriminative

Given **finite data** (n data points, d features),

[Ng & Jordan, NIPS 2001]

$$\epsilon_{\mathrm{Dis},n} \leq \epsilon_{\mathrm{Dis},\infty} + O\left(\sqrt{\frac{d}{n}}\right)$$

$$\epsilon_{\mathrm{Gen},n} \leq \epsilon_{\mathrm{Gen},\infty} + O\left(\sqrt{\frac{\log d}{n}}\right)$$

Naïve Bayes (generative) requires n = O(log d) to converge to its asymptotic error, whereas Logistic regression (discriminative) requires n = O(d).

Why? “Independent class conditional densities”

* parameter estimates not coupled – each parameter is learnt independently, not jointly, from training data.

Naïve Bayes vs. Logistic Regression

Verdict

Both learn a linear decision boundary.

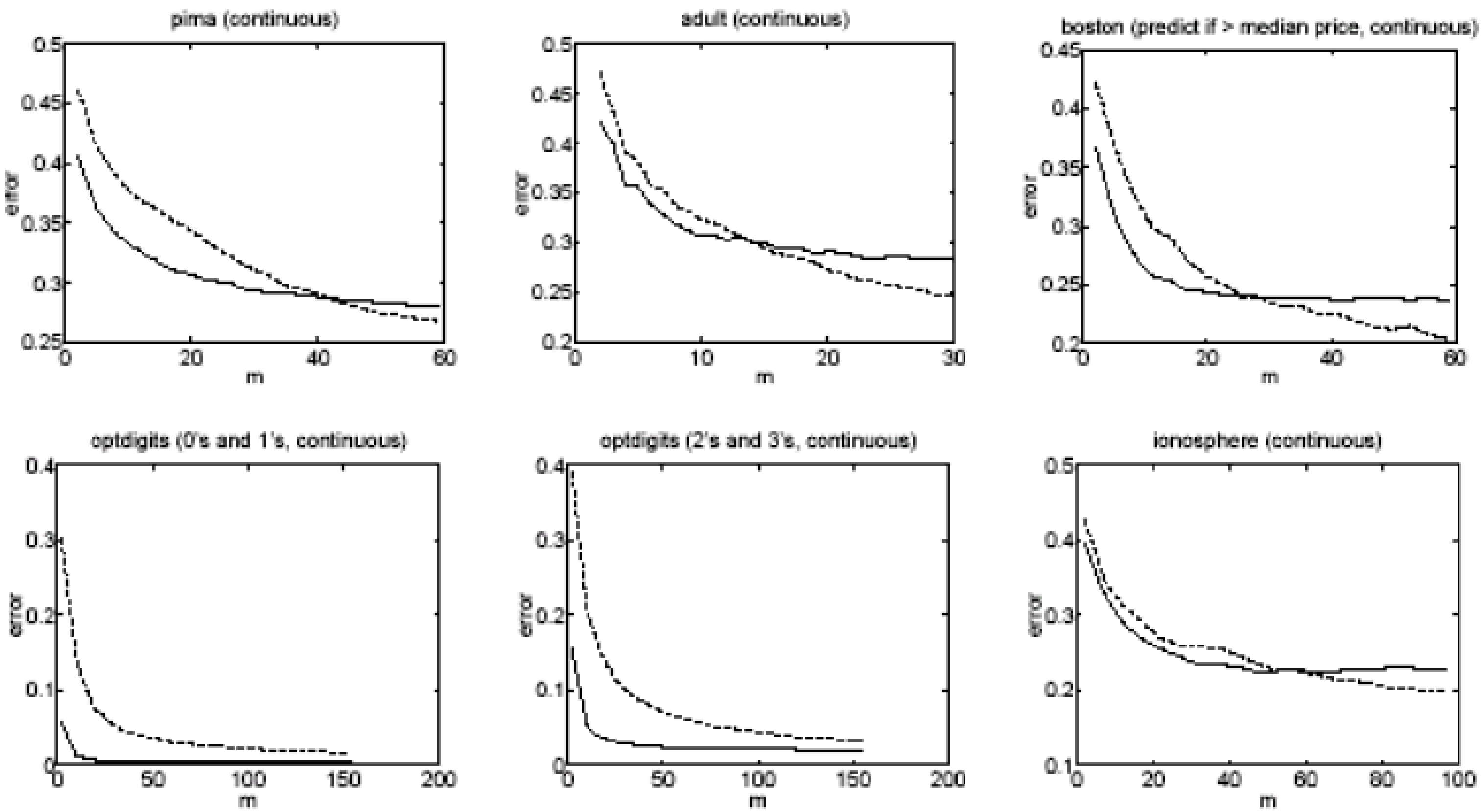
Naïve Bayes makes more restrictive assumptions and has higher asymptotic error,

BUT

converges faster to its less accurate asymptotic error.

Experimental Comparison (Ng-Jordan'01)

- UCI Machine Learning Repository 15 datasets, 8 continuous features, 7 discrete features



More in Paper…

—— Naïve Bayes ----- Logistic Regression

What you should know

- LR is a linear classifier
  - decision rule is a hyperplane
- LR optimized by conditional likelihood
  - no closed-form solution
  - concave $\rightarrow$ global optimum with gradient ascent
  - Maximum conditional a posteriori corresponds to regularization
- Gaussian Naïve Bayes with class-independent variances representationally equivalent to LR
  - Solution differs because of objective (loss) function
- In general, NB and LR make different assumptions
  - NB: Features independent given class $\rightarrow$ assumption on P(**X**|Y)
  - LR: Functional form of P(Y|**X**), no assumption on P(**X**|Y)
- Convergence rates
  - GNB (usually) needs less data
  - LR (usually) gets to better solutions in the limit