Logistic Regression

Aarti Singh

Machine Learning 10-315
Sept 15, 2021

Discriminative Classifiers

Bayes Classifier:

$$f^*(x) = \arg\max_{Y=y} P(Y = y|X = x)$$
$$= \arg\max_{Y=y} P(X = x|Y = y)P(Y = y)$$

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

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

Today we will see one such classifier – **Logistic Regression**

Logistic Regression

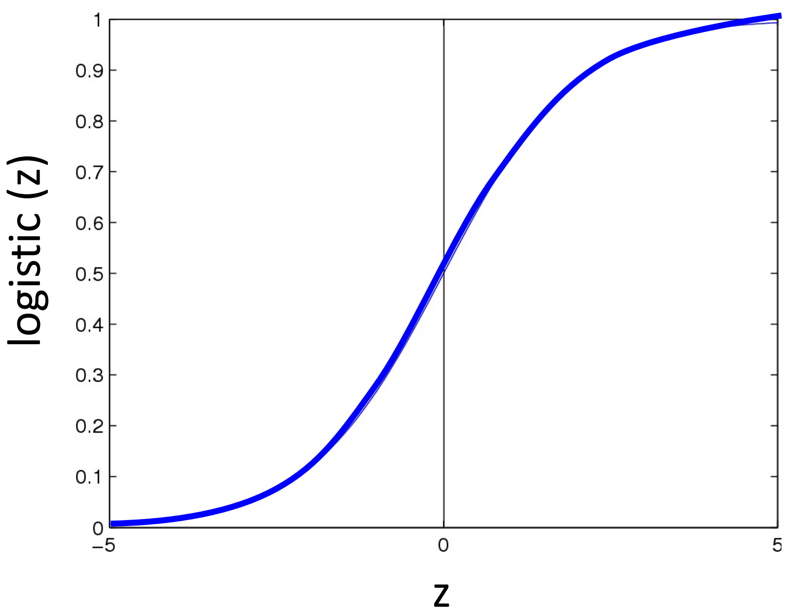
Not really regression

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), σ(z) = : $\frac{1}{1+exp(-z)}$



Features can be discrete or continuous!

Logistic Regression is a Linear Classifier!

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

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

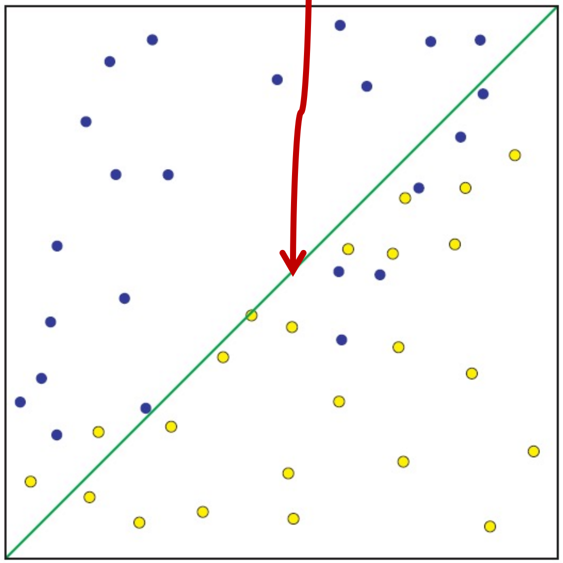
Decision boundary:

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

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

(Linear Decision Boundary)

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



Training Logistic Regression

How to learn the parameters w_0, w_1, … w_d? (d features)

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? (d features)

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

Maximum (Conditional) Likelihood Estimates

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

Discriminative philosophy – Don't waste effort learning P(X), focus on P(Y|X) – that's all that matters for classification!

Expressing Conditional log Likelihood

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

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

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

Expressing Conditional log Likelihood

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

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

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

Good news: $l(\mathbf{w})$ is concave function of **w** !

Expressing Conditional log Likelihood

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

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

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

Good news: *l*(**w**) is concave function of **w** !

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

Good news: can use iterative optimization methods (gradient ascent)

That's M(C)LE. How about M(C)AP?

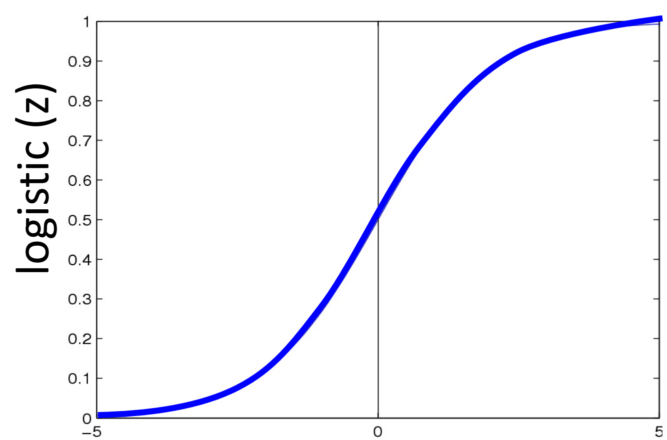
$$p(\mathbf{w} \mid Y, \mathbf{X}) \;\propto\; P(Y \mid \mathbf{X}, \mathbf{w})p(\mathbf{w})$$

- Define priors on **w**
  - Common assumption: Normal distribution, zero mean, identity covariance
  - "Pushes" parameters towards zero

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

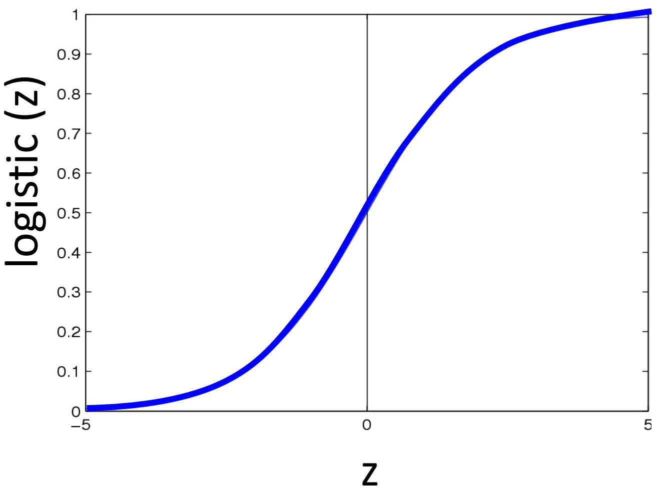
Zero-mean Gaussian prior

Logistic function (or Sigmoid), σ(z) = : $\dfrac{1}{1 + exp(-z)}$



➢ What happens if we scale z by a large constant?

Logistic function (or Sigmoid), $\sigma(z) = : \dfrac{1}{1 + exp(-z)}$



- Poll: What happens if we scale z (equivalently weights w) by a large constant?

A) The logistic classifier decision boundary shifts towards class 1
B) The logistic classifier decision boundary remains same
C) The logistic classifier tries to separate the data perfectly
D) The logistic classifier allows more mixing of labels on each side of decision boundary

That's M(C)LE. How about M(C)AP?

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

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

- M(C)AP estimate

Zero-mean Gaussian prior

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

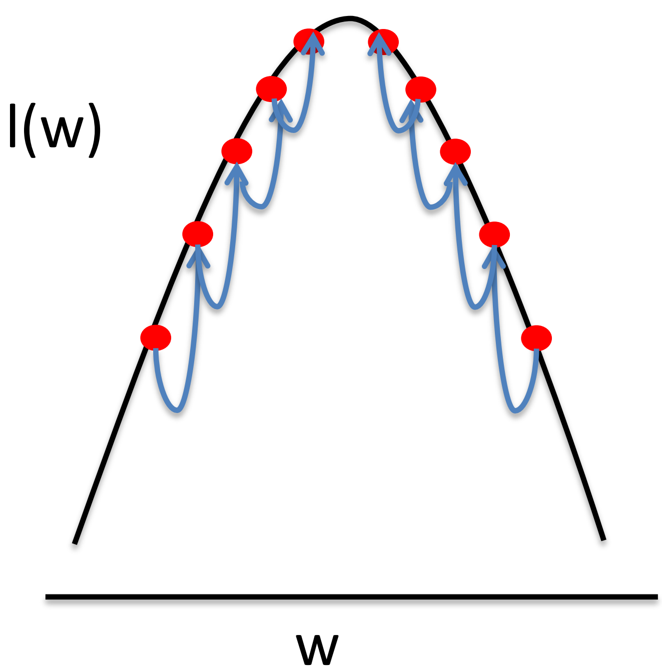
$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{j=1}^{n} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w}) - \underbrace{\sum_{i=1}^{d} \frac{w_i^2}{2\kappa^2}}_{\boxed{\text{Penalizes large weights}}}$$

Still concave objective!

Iteratively optimizing concave function

- Conditional likelihood for Logistic Regression is concave
- Maximum of a concave function can be reached by

Gradient Ascent Algorithm



Initialize: Pick **w** at random

Gradient:

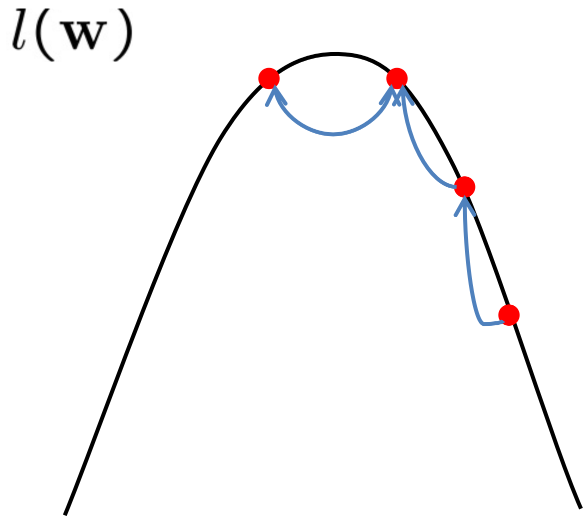
$$\nabla_{\mathbf{w}} l(\mathbf{w}) = \left[\frac{\partial l(\mathbf{w})}{\partial w_0}, \ldots, \frac{\partial l(\mathbf{w})}{\partial w_d}\right]'$$

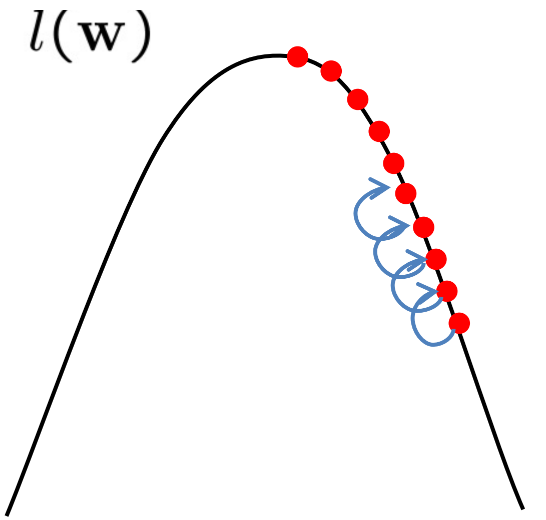
Update rule: Learning rate, η>0

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

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

Effect of step-size η





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

Small η => Slow convergence but small residual error

Gradient Ascent for M(C)LE

Gradient ascent rule for w_0:

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \left. \frac{\partial l(\mathbf{w})}{\partial w_0} \right|_t$$

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

Gradient Ascent for M(C)LE

Gradient ascent rule for w_0:

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \left. \frac{\partial l(\mathbf{w})}{\partial w_0} \right|_t$$

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

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

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

Gradient Ascent for M(C)LE Logistic Regression

Gradient ascent algorithm: iterate until change < ε

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

For i=1,…,d,

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

repeat

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

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

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

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

$$\propto \frac{-w_i}{\kappa^2}$$

$\boxed{\text{Extra term Penalizes large weights}}$

Penalization = Regularization

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

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

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

Logistic Regression for more than 2 classes

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

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)

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

Predict $f^*(x) = \arg\max_{Y=y} P(Y = y|X = x)$

Is the decision boundary still linear?

Comparison with Gaussian Naïve Bayes

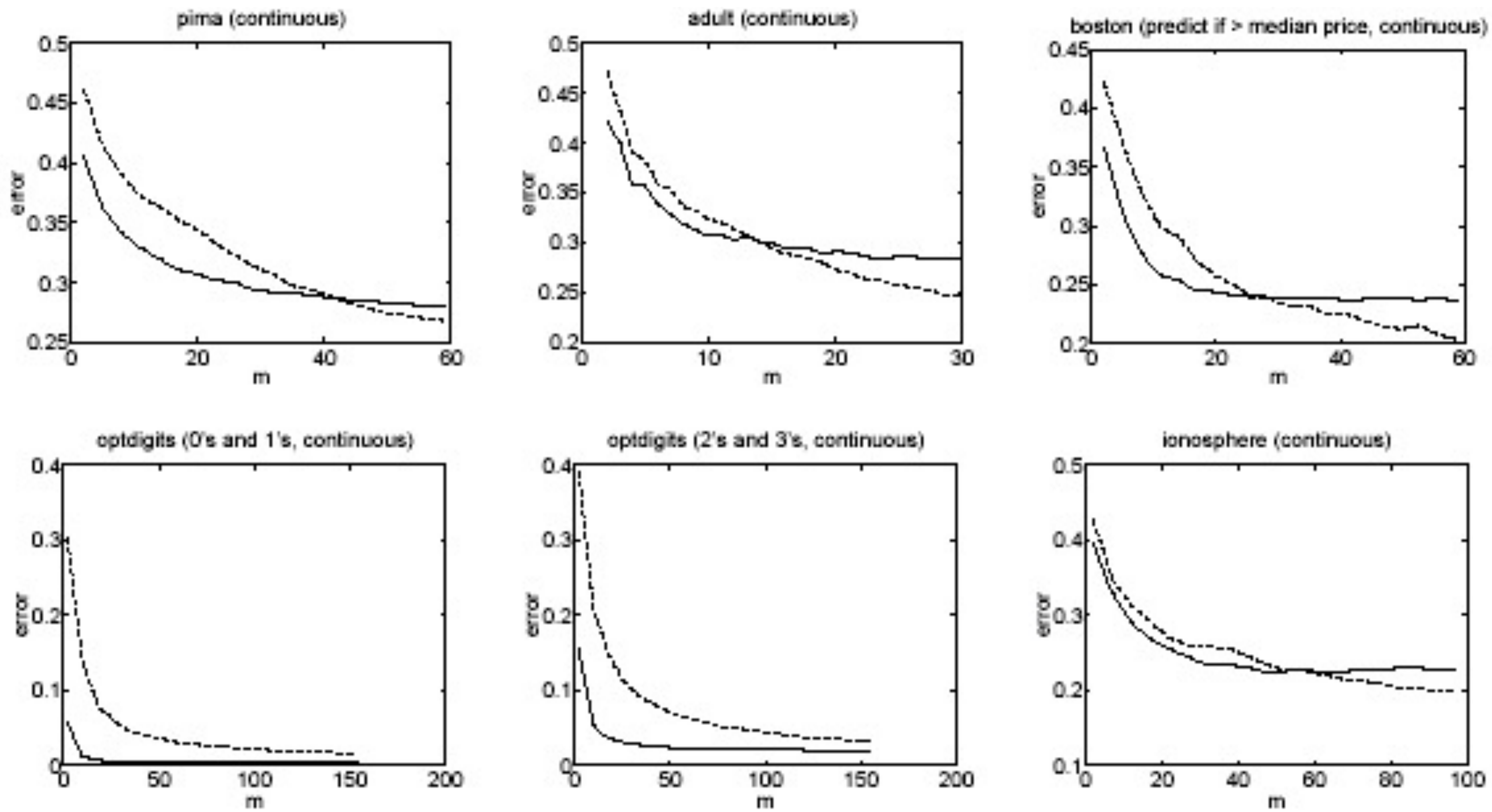
Gaussian Naïve Bayes vs. Logistic Regression

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

- Representation equivalence (both yield linear decision boundaries)
  - **But only in a special case!!!** (GNB with class-independent variances)
  - **LR makes no assumptions about** P(**X**|Y) **in learning**!!!
  - **Optimize different functions (MLE/MCLE) or (MAP/MCAP)! Obtain different solutions**

Experimental Comparison (Ng-Jordan'01)

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



More in Paper…

—— Naïve Bayes ----- Logistic Regression

Gaussian Naïve Bayes vs. Logistic Regression

- If conditional independence assumption holds, then GNB has lower large sample error

- If conditional independence assumption DOES NOT hold, then GNB has higher large sample error
  - But if converges faster (to its higher large sample error) as the parameter estimates are not coupled

What you should know

- LR is a linear classifier
- LR optimized by maximizing conditional likelihood or conditional posterior
  - no closed-form solution
  - concave ! global optimum with gradient ascent
- 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 ! 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