

# Machine Learning 10-601

Tom M. Mitchell  
Machine Learning Department  
Carnegie Mellon University

October 4, 2011

## Today:

- Logistic regression
- Generative/Discriminative classifiers

## Readings:

### Required:

- Mitchell: "Naïve Bayes and Logistic Regression"  
(available on class website)
- Bishop: Chapt. 3 through 3.2

### Optional

- Ng & Jordan (see class website)

- Consider learning  $f: X \rightarrow Y$ , where
  - $X$  is a vector of real-valued features,  $\langle X_1 \dots X_n \rangle$
  - $Y$  is boolean
  - assume all  $X_i$  are conditionally independent given  $Y$
  - model  $P(X_i | Y = y_k)$  as Gaussian  $N(\mu_{ik}, \sigma_i)$
  - model  $P(Y)$  as Bernoulli ( $\pi$ )
- What does that imply about the form of  $P(Y|X)$ ?

$$P(Y = 1 | X = \langle X_1, \dots, X_n \rangle) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

## Training Logistic Regression: MCLE

- Choose parameters  $W = \langle w_0, \dots, w_n \rangle$  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  $D = \{\langle X^1, Y^1 \rangle, \dots, \langle X^L, Y^L \rangle\}$
- 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

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

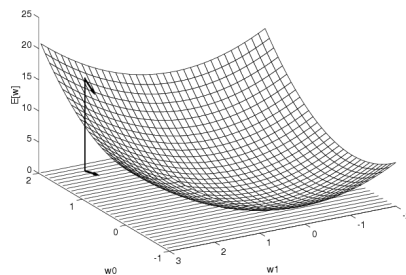
$$P(Y = 1|X, W) = \frac{\exp(w_0 + \sum_i w_i X_i)}{1 + \exp(w_0 + \sum_i w_i X_i)}$$

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

Good news:  $l(W)$  is convex function of  $W$

Bad news: no closed-form solution to maximize  $l(W)$

## Gradient Descent



Gradient

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

Training rule:

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

i.e.,

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

## Gradient Descent:

**Batch gradient:** use error  $E_D(\mathbf{w})$  over entire training set  $D$

Do until satisfied:

1. Compute the gradient  $\nabla E_D(\mathbf{w}) = \left[ \frac{\partial E_D(\mathbf{w})}{\partial w_0} \dots \frac{\partial E_D(\mathbf{w})}{\partial w_n} \right]$
2. Update the vector of parameters:  $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla E_D(\mathbf{w})$

**Stochastic gradient:** use error  $E_d(\mathbf{w})$  over single examples  $d \in D$

Do until satisfied:

1. Choose (with replacement) a random training example  $d \in D$
2. Compute the gradient just for  $d$ :  $\nabla E_d(\mathbf{w}) = \left[ \frac{\partial E_d(\mathbf{w})}{\partial w_0} \dots \frac{\partial E_d(\mathbf{w})}{\partial w_n} \right]$
3. Update the vector of parameters:  $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla E_d(\mathbf{w})$

Stochastic approximates Batch arbitrarily closely as  $\eta \rightarrow 0$

Stochastic can be much faster when  $D$  is very large

Intermediate approach: use error over subsets of  $D$

## 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

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

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

Gradient ascent algorithm: iterate until change  $< \epsilon$

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 prior on weights  $W = \langle w_0, w_1, \dots, w_n \rangle$
- 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: each  $w_i \sim N(0, \sigma)$

$$p(w_i) = \frac{1}{Z} \exp \left( -\frac{(w - 0)^2}{2\sigma^2} \right)$$

## MLE vs MAP

- Maximum conditional likelihood estimate

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

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

- MAP estimate with Gaussian prior  $p(w_i) = \frac{1}{Z} \exp\left(-\frac{(w_i - 0)^2}{2\sigma^2}\right)$

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

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

called a “regularization” term

## The Bottom Line

- Consider learning  $f: X \rightarrow Y$ , where
  - $X$  is a vector of real-valued features,  $\langle X_1 \dots X_n \rangle$
  - $Y$  is boolean
  - assume all  $X_i$  are conditionally independent given  $Y$
  - model  $P(X_i | Y = y_k)$  as Gaussian  $N(\mu_{ik}, \sigma_i)$
  - model  $P(Y)$  as Bernoulli ( $\pi$ )
- Then  $P(Y|X)$  is of this form, and we can directly estimate  $W$ 

$$P(Y = 1 | X = \langle X_1, \dots, X_n \rangle) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)}$$
- Furthermore, same holds if the  $X_i$  are boolean
  - trying proving that to yourself

## Generative vs. Discriminative Classifiers

Training classifiers involves estimating  $f: X \rightarrow Y$ , 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 Logistic 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,  $X = \langle X_1 \dots X_n \rangle$

Number of parameters to estimate:

- NB:

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

- LR:

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

## Naïve Bayes vs Logistic Regression

Consider  $Y$  boolean,  $X_i$  continuous,  $X = \langle X_1 \dots X_n \rangle$

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]

Recall two assumptions deriving form of LR from GNBayes:

1.  $X_i$  conditionally independent of  $X_k$  given  $Y$
2.  $P(X_i | Y = y_k) = N(\mu_{ik}, \sigma_i)$ ,  $\leftarrow$  not  $N(\mu_{ik}, \sigma_{ik})$

Consider three learning methods:

- GNB (assumption 1 only)
- GNB2 (assumption 1 and 2)
- LR

Which method works better if we have infinite training data, and...

- Both (1) and (2) are satisfied
- Neither (1) nor (2) is satisfied
- (1) is satisfied, but not (2)

## G.Naïve Bayes vs. Logistic Regression

[Ng & Jordan, 2002]

Recall two assumptions deriving form of LR from GNBayes:

1.  $X_i$  conditionally independent of  $X_k$  given  $Y$
2.  $P(X_i | Y = y_k) = N(\mu_{ik}, \sigma_i)$ ,  $\leftarrow$  not  $N(\mu_{ik}, \sigma_{ik})$

Consider three learning methods:

- GNB (assumption 1 only) -- decision surface can be non-linear
- GNB2 (assumption 1 and 2) -- decision surface linear
- LR -- decision surface linear, trained differently

Which method works better if we have infinite training data, and...

- Both (1) and (2) are satisfied: LR = GNB2 = GNB
- Neither (1) nor (2) is satisfied: LR > GNB2, GNB > GNB2
- (1) is satisfied, but not (2) : GNB > LR, LR > GNB2

## G.Naïve Bayes vs. Logistic Regression

[Ng & Jordan, 2002]

What if we have only finite training data?

They converge at different rates to their asymptotic ( $\infty$  data) error

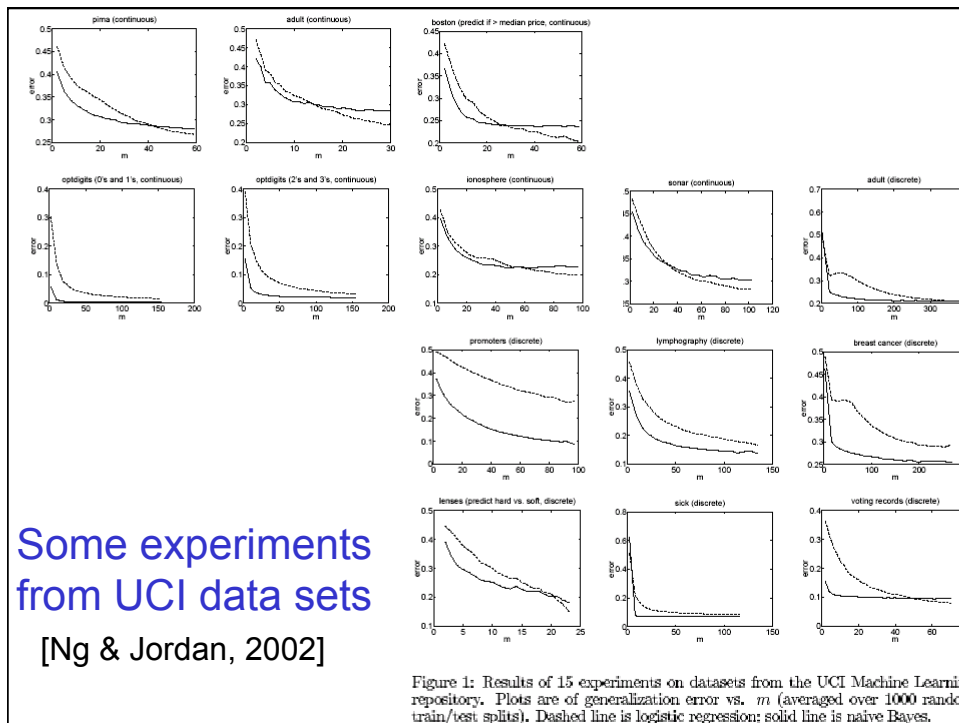
Let  $\epsilon_{A,n}$  refer to expected error of learning algorithm A after n training examples

Let d be the number of features:  $\langle X_1 \dots X_d \rangle$

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

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

So, GNB requires  $n = O(\log d)$  to converge, but LR requires  $n = O(d)$



## Naïve Bayes vs. Logistic Regression

The bottom line:

GNB2 and LR both use linear decision surfaces, GNB need not

Given infinite data, LR is better or equal to GNB2 because *training procedure* does not make assumptions 1 or 2 (though our derivation of the form of  $P(Y|X)$  did).

But GNB2 converges more quickly to its perhaps-less-accurate asymptotic error

And GNB is both more biased (assumption1) and less (no assumption 2) than LR, so either might beat the other

## 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