Logistic Regression

Adapted from Tom Mitcell’s Lecure

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 ?

Logistic Regression

- **Consider learning f: X → Y, where**
  - X is a vector of real-valued, <X_1 ... X_n>
  - Y is boolean
  - Assume all $\mathbf{X_i}$ are conditionally idependent given Y
  - Model P($\mathbf{X_i}$ | Y =y_k) as Gaussian N (μ_{ik} , σ_i)
  - Model P(Y) as Bernoulli (π)
- **What does that imply about the form of P (Y|X) ?**

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

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

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

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

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

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

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

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

$$P(x \mid y_k) \quad = \frac{1}{1 + \exp\left(\left(\ln \frac{1-\pi}{\pi}\right) + \sum_i \ln \frac{P(X_i \mid Y=0)}{P(X_i \mid Y=1)}\right)}$$

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

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

$$P(x \mid y_k) \quad = \frac{1}{1+\exp\left(\left(\ln\frac{1-\pi}{\pi}\right)+\sum_i \ln\frac{P(X_i \mid Y=0)}{P(X_i \mid Y=1)}\right)}$$

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

$$P(x \mid y_k) \quad = \frac{1}{1 + \exp\left(\left(\ln \frac{1-\pi}{\pi} \right) + \sum_i \ln \frac{P(X_i \mid Y=0)}{P(X_i \mid Y=1)} \right)}$$

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

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

Very convenient

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

- implies

$$P(Y=0 \mid X<X_1,\ldots,X_n>) =$$

Very convenient

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

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

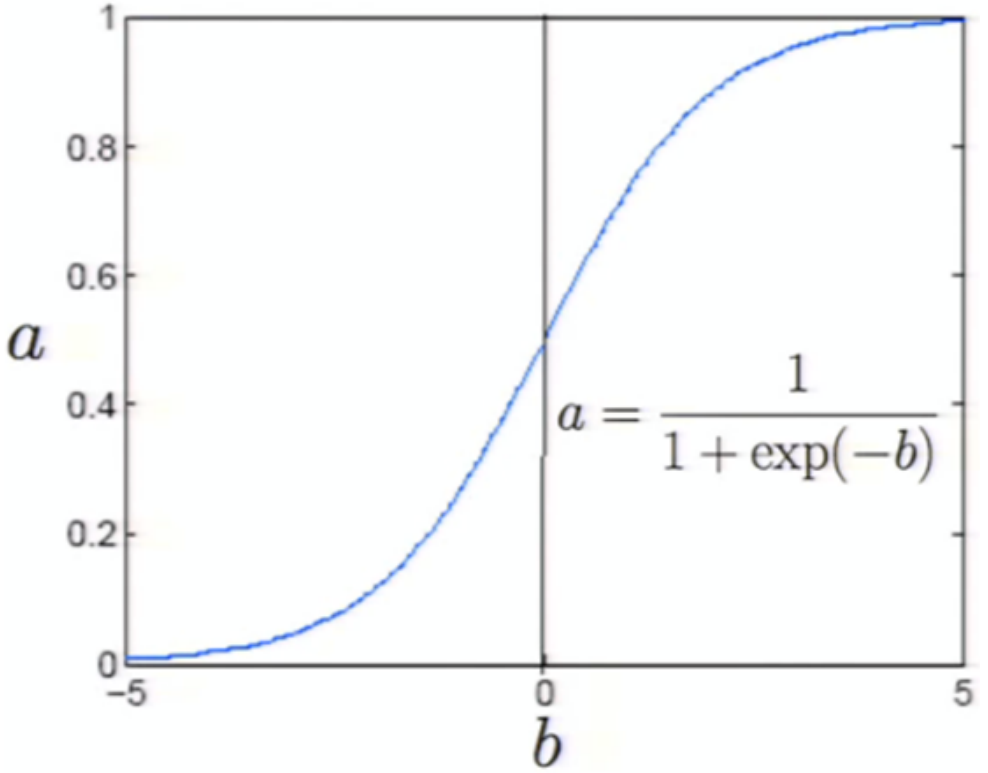
- implies

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

- implies

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

Logistic Fuction



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

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

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

Training Logistic Regression : MCLE

- We have L training examples : $\left\{ \left\langle X^1, Y^1 \right\rangle, \ldots, \left\langle X^L, Y^L \right\rangle \right\}$

- 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 <u>conditional</u> likehood estimate

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

Training Logistic Regression : MCLE

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

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

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

- Training data D = $\left\{ \left\langle X^1, Y^1 \right\rangle, \ldots, \left\langle X^L, Y^L \right\rangle \right\}$
- Data likehood = $\prod_l P(< X^l, Y^l > \mid W)$
- Data conditional likehood = $\prod_l P(Y^l \mid X^l, W)$

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

Expressing Conditional Log Likehood

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

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

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

$$l(W) = \sum_l Y^l \ln P(Y^l = 1 \mid X^l, W) + (1 - Y^l) \ln P(Y^l = 0 \mid X^l, W)$$

$$= \sum_l Y^l \ln \frac{P(Y^l = 1 \mid X^l, W)}{P(Y^l = 0 \mid X^l, W)} + \ln P(Y^l = 0 \mid X^l, W)$$

$$= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + \exp(w_0 + \sum_i^n w_i X_i^l))$$

Maximizing Conditional Log Likelihood

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

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

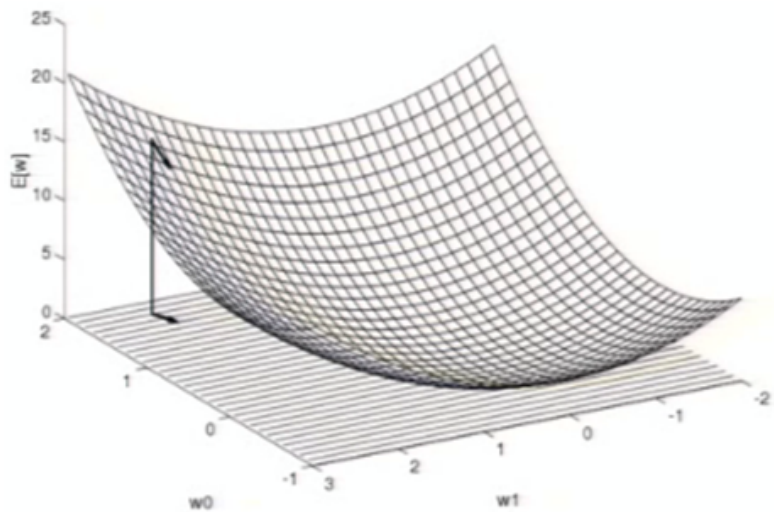
$$\begin{aligned} l(W) &\equiv \ln \prod_l P(Y^l|X^l,W) \\ &= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1+exp(w_0+\sum_i^n w_i X_i^l)) \end{aligned}$$

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

Gradient Descent

- Gradient

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



- Training rule :

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

- i.e., $\Delta w_i = -\eta \dfrac{\partial E}{\partial w_i}$

Maximize Conditional Log Likelihood : Gradient Ascent

$$l(W) \equiv \ln \prod_l P(P^l \mid X^l, W)$$

$$= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + \exp(w_0 + \sum_i^n w_i X_i^l))$$

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

Gradient ascent algoritm : iterate until change < ε

For all i repeat

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

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

Regression

So far, we've been interested in learning P(Y|X) where Y has discrete values (called 'classification')

What if Y is continuous ? (calles 'regression')

- Predict weight from gender, height, age, ...

- Predict Google stock price today from Google, Yahoo, MSFT prices yesterday

- Predict each pixel intensity in robot's current camera image, from previous image and previous action

Regression

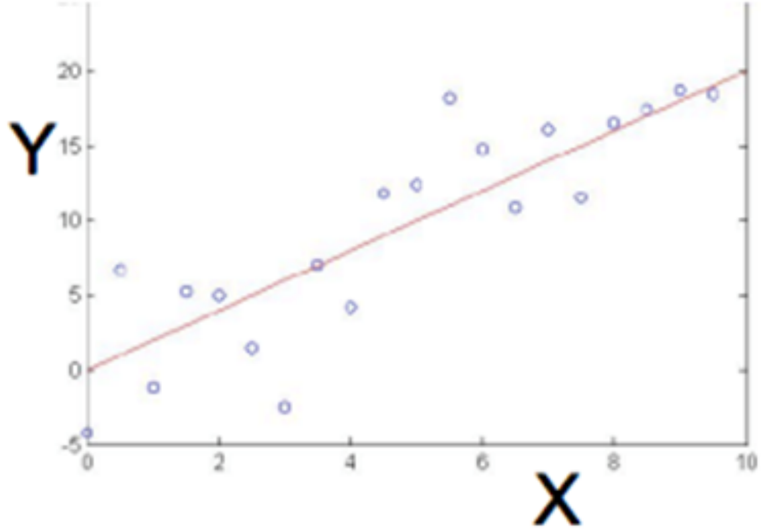
Wish to learn $f:X \rightarrow Y$, where Y is real, given $\{<x^1,y^1> \ldots <x^n,y^n>\}$

Approach :

1. Choose some parameterized form for $P(Y|X; \theta)$ (θ is the vector of parameters)

2. Derive learning algoritma as MLE or MAP estimate for θ

Choose parameterized form for P(Y|X; θ)



Assume Y is some deterministic f(X), plus random noise

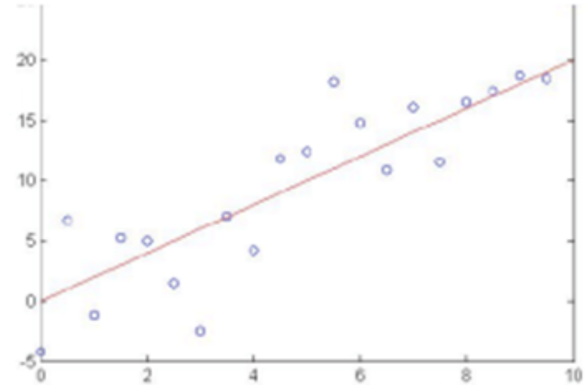
$$y = f(x) + \in \quad \text{where} \quad \in \sim N(0, \sigma)$$

Therefore Y is a random variable that follows the distribution

$$p(y \mid x) = N(f(x), \sigma)$$

And the expected value of y for any given x is f(x)

Training Linear Regression



$$p(y \mid x; W) = N(w_0 + w_1 x, \sigma)$$

How can we learn W from the training data ?

Learn Maximum Conditional Likelihood Estimate!

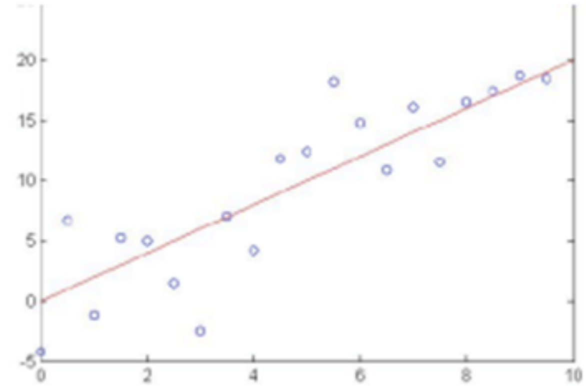
$$W_{MLCE} = \arg\max_W \prod_l p(y^l \mid x^l, W)$$

$$W_{MLCE} = \arg\max_W \sum_l \ln p(y^l \mid x^l, W)$$

where

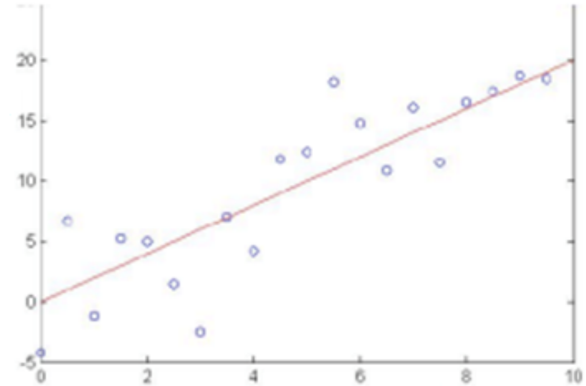
$$p(y \mid x; W) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{y - f(x;W)}{\sigma}\right)^2}$$

Training Linear Regression



$$p(y \mid x; W) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{y - f(x;W)}{\sigma}\right)^2}$$

Training Linear Regression



Learn Maximum Conditional Likelihood Estimate!

$$W_{MLCE} = \arg\min_{W} \sum_{l} (y - f(x;W))^2$$

Can we derive gradient descent rule for training ?

$$\frac{\partial \sum_l (y - f(x;W))^2}{\partial w_i} = \sum_l 2(y - f(x;W)) \frac{\partial (y - f(x;W))}{\partial w_i}$$

$$= \sum_l -2(y - f(x;W)) \frac{\partial f(x;W)}{\partial w_i}$$

How about MAP instead of MLE estimate?

$$W = \arg\max_W \ \lambda R(W) + \sum_l \ln P(Y^l | X^l; W)$$

$$R(W) = ||W||_2^2 = \sum_i w_i^2$$

Regression – What you should know

Under general assumption $p(y|x;W) = N(f(x;W), \sigma)$

1. MLE corresponds to minimizing sum of squared prediction errors

2. MAP estimate minimizes SSE plus sum of squared weights

3. Again, learning is an optimization problem once we choose our objective function
   - maximize data likelihood
   - maximize posterior prob of W

4. Again, we can use gradient descent as a general learning algorithm
   - as long as our objective fn is differentiable wrt W
   - though we might learn local optima ins

5. Almost nothing we said here required that f(x) be linear in x