Module 5: Classification

Basic Concepts: Risk and Measures of Predictive Accuracy

STAT/BIOSTAT 527, University of Washington
Emily Fox

May 23rd, 2013

The Optimal Prediction

- Assume we *know* the data-generating mechanism
- If our task is prediction, which summary of the distribution $Y \mid x$ should we report?

For x, what fcn $f(x)$ should we choose to predict Y if we can choose any $f(\cdot)$

- Taking a decision-theoretic framework, consider the ***expected loss*** (predictions are penalized by $L(\cdot,\cdot)$)

$$E_{X,Y}[L(Y, f(X))] = E_X\{E_{Y|X}[L(Y, f(x)) \mid X = x]\}$$

- $\hat{f}(\cdot)$ should min ↗
- can min. pointwise

Continuous Responses

- Expected loss $E_X \left\{ E_{Y|X} \left[L(Y, f(x)) \mid X = x \right] \right\}$
- Example: L_2 $L(Y, f(x)) = (Y - f(x))^2$

  Solution: $\hat{f}(x) = E[Y|x]$ ← focus in course so far

  Proofs: HW
- Example: L_1 $L(Y, f(x)) = |Y - f(x)|$

  Solution: $\hat{f}(x) = \text{median}(Y|x)$
- More generally: L_p $L(Y, f(x)) = \left\{ \int |Y - f(x)|^p \right\}^{1/p}$

Categorical Responses

- Expected loss $E_X \left\{ E_{Y|X} \left[L(Y, g(x)) \mid X = x \right] \right\}$
- Response: $Y \in \{G_1, \ldots, G_K\}$ # of classes
- Same setup, but need new loss function
- Can always represent loss function with K x K matrix L

  truth → $L_{jk} \triangleq L(j,k) = \begin{cases} 0 & j = k \\ \geq 0 & j \neq k \end{cases}$ ← pred. class
- L is zeros on the diagonal and non-negative elsewhere
- Typical loss function: zero-one (0-1)

  $L_{jk} \triangleq L(j,k) = \begin{cases} 0 & j = k \\ 1 & j \neq k \end{cases}$ unit cost for all possible mistakes

Optimal Prediction

- Expected loss

$$E_X\left\{E_{Y|X}\left[L(Y, g(x)) \mid X = x\right]\right\} =$$

$$= E_X\left\{\sum_{k=1}^{K} L(G_k, g(x)) \Pr(G_k \mid X = x)\right\}$$

- Again, can minimize pointwise

$$\hat{g}(x) = \arg\min_g \sum_{k=1}^{K} L(G_k, g) \Pr(G_k \mid X = x)$$

  - Example: $K=2$

$g=0$: $L(1,0)\Pr(G_1|x) + L(0,0)\Pr(G_0|x)$ → 0

$g=1$: $L(1,1)\Pr(G_1|x) + L(0,1)\Pr(G_0|x)$ → 0

$\hat{g}(x)=1$ if $L(1,0)\Pr(G_1|x) > L(0,1)[1 - \Pr(G_1|x)]$

$$\Pr(G_1|x) > \frac{L(0,1)}{L(0,1) + L(1,0)}$$

Optimal Prediction

$$\hat{g}(x) = \arg\min_g \sum_{k=1}^{K} L(\mathcal{G}_k, g) \Pr(\mathcal{G}_k \mid X = x)$$

- With 0-1 loss, we straightforwardly get the ***Bayes classifier***

$$\hat{g}(x) = \arg\min_g \left[1 - \Pr(g \mid X = x)\right]$$

or

$$\hat{g}(x) = G_k \text{ if } \Pr(G_k \mid X = x) = \max_g \Pr(g \mid X = x)$$

classify w/ most probable class

Optimal Prediction

$$\hat{g}(x) = \mathcal{G}_k \quad \text{if} \quad \Pr(\mathcal{G}_k \mid X = x) = \max_g \Pr(g \mid X = x)$$

- How to approximate the optimal prediction?
  - Don't actually have $p(Y \mid X = x)$
- Nearest neighbor approach
  - Look at *k*-nearest neighbors with majority vote to estimate

Before (L_2): $E[Y|x] \approx \text{avg}(y_i \mid x_i \in \text{nbhd}(x))$

$$\Pr(G_m | X = x) \approx \frac{1}{k} \sum_{x_i \in \text{nbhd}(x)} \mathbb{1}(y_i = m)$$

classify w/ largest $P(G_m | X = x)$

(most common label of k-NN)

Optimal Prediction

$$\hat{g}(x) = \mathcal{G}_k \quad \text{if} \quad \Pr(\mathcal{G}_k \mid X = x) = \max_g \Pr(g \mid X = x)$$

- How to approximate the optimal prediction?
  - Don't actually have $p(Y \mid X = x)$
- Model-based approach
  - Introduce indicators for each class:
  - Consider squared-error loss: $\hat{f}(X) = E[Y \mid X]$

$Y_k = 1$ if $Y = G_k$

$Y = [0\ 0\ 1\ 0\ 0 \ldots 0]$

$$E[Y_k | x] = \Pr(Y = G_k | x)$$

  - Bayes classifier is equivalent to standard regression and L_2 loss, followed by classification to largest fitted value

$\hat{f}(x) = \begin{bmatrix} 0.1 \\ 0.07 \\ 0.9 \\ \vdots \\ 0 \end{bmatrix}$ ← largest ⇒ class 3

  - Works in theory, but not in practice…Will look at many other approaches (e.g., logistic regression)

Measuring Accuracy of Classifier

- For a given classifier, how do we assess how well it performs?
- For 0-1 loss, the generalization error is

$$E_{X,Y}[g(x) \neq Y] = Pr_{X,Y}(g(x) \neq Y)$$

with empirical estimate

$$\frac{1}{m}\sum_{i=1}^{m} \mathbb{1}(g(x_i) \neq y_i)$$

(chosen classifier)

- Consider binary response and some useful summaries

Eg. $Y = \begin{cases} 0 & \text{if true state is no disease} \\ 1 & \text{if disease} \end{cases}$

decision rule $g(x) = \begin{cases} 0 & \text{if predict no disease} \\ 1 & \text{predict disease} \end{cases}$

Measuring Accuracy of Classifier

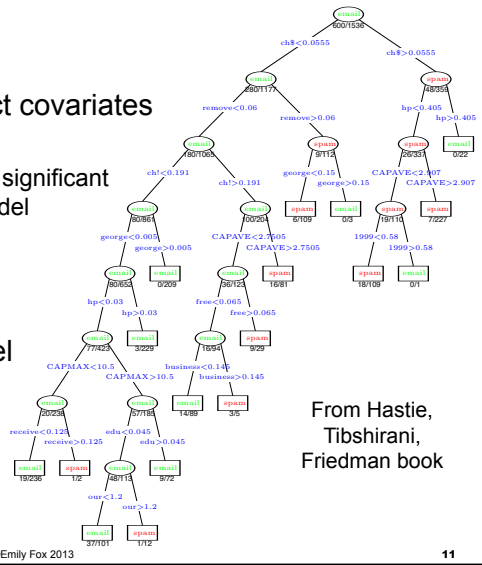
- Sensitivity: prob. of pred. disease for a diseased individual
  $Pr(g(x)=1 \mid Y=1)$
- Specificity: prob. of pred. disease-free when disease-free
  $Pr(g(x)=0 \mid Y=0)$
- False positive rate:
  $Pr(g(x)=1 \mid Y=0)$
- True positive rate:
  $Pr(g(x)=1 \mid Y=1)$
- Connections:
  sensitivity = TPR specifity = 1 - FPR

what if we increase $L(0,1)$? inc. specificity

Classification Tree Spam Example

- Resulting tree of size 17
- Note that there are 13 distinct covariates split on by the tree
  - 11 of these overlap with the 16 significant predictors from the additive model previously explored
- Overall error rate (9.3%) is higher than for additive model

	Predicted	
True	email	spam
email	57.3%	4.0%
spam	5.3%	33.4%



From Hastie, Tibshirani, Friedman book

Classification Tree Spam Example

- Think of ***spam*** and ***email*** as presence and absence of disease
- Using equal losses
  - Sensitivity = $100 \frac{33.4}{33.4+5.3} = 86.3\%$
  - Specificity = $100 \frac{57.3}{57.3+4.0} = 93.4\%$

	Predicted	
True	email	spam
email	57.3%	4.0%
spam	5.3%	33.4%

From Hastie, Tibshirani, Friedman book

- By varying L_{01} and L_{10}, can increase/decrease sensitivity and decrease/increase specificity of rule
- Which do we want here? avoid marking 'email' as 'spam'
- How? $L_{01} > L_{10} = 1$ ... high specificity
- Change in rule at leaf: 'spam' if proportion of 'spam' at leaf $\geq \frac{L_{01}}{L_{10}+L_{01}}$

ROC Curves

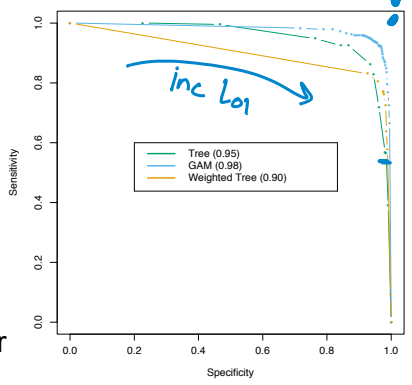
- ***Receiver operating characteristic*** (ROC) curve summarizes tradeoff between sensitivity and specificity
  - Plot of sensitivity vs. specificity as a function of params of classification rule

want this

- Example: vary L_{01} in [0.1,10]
  - Want specificity near 100%, but in this case sensitivity drops to about 50%

- Summary = area under the curve
  - Tree = 0.95
  - GAM = 0.98 ← wins

- Instead of Bayes rule at leaf, better to account for unequal losses in constructing tree … see book



From Hastie, Tibshirani, Friedman book

What you need to know

- Again, goal framed as minimizing expected loss

- Loss here is summarized by K x K matrix L
  - Common choice = 0-1 loss

- Bayes classifier chooses most probable class

- Measures of predictive performance:
  - Sensitivity, specificity, true positive rate, false positive rate
  - ROC curve and area under the curve

Readings

- Wakefield – 10.3.2, 10.4.2, 12.8.4
- Hastie, Tibshirani, Friedman – 9.2.3, 9.2.5, 2.4

Module 5: Classification

Linear Methods: Logistic Regression

STAT/BIOSTAT 527, University of Washington
Emily Fox
May 23rd, 2013

Link Functions

Focus on $y \in \{0,1\}$ or, generally $y \in \{1, \ldots, k\}$

- Estimating p(*Y*|***X***): Why not use standard linear regression?

$$p(y|x) = \beta_0 + \sum_j \beta_j h_j(x)$$

range $(-\infty, \infty)$ BAD

$P(y=k|x) \in [0,1]$

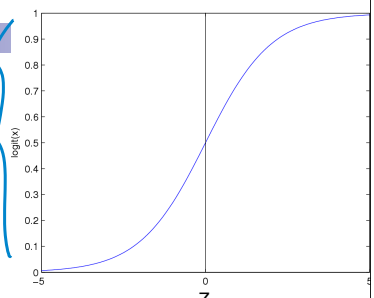
- Combing regression and probability?
  - Need a mapping from real values to [0,1]
  - A link function!

$g: \mathbb{R} \rightarrow [0,1]$

many options, but here's a useful one...

Logistic Regression

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



- Learn p(*Y*|***X***) directly
  - Assume a particular functional form for link function
  - Sigmoid applied to a linear function of the input features:

$$p(y=0 \mid x, \beta) = \frac{1}{1+\exp(\beta_0 + \sum_j \beta_j x_j)}$$

↑ choice

z: linear, just like in standard regression

$\mathbb{R}$

$\beta_0 + \sum_j \beta_j x_j$ not bounded could be neg.

after logistic fcn, output is in $[0,1]$

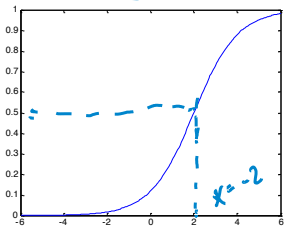
Covariates can be discrete or continuous!

Understanding the Sigmoid

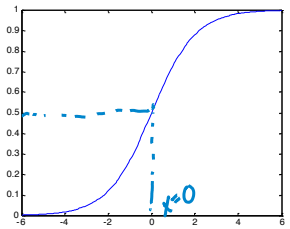
$$g(\beta_0 + \sum_j \beta_j x_j) = \frac{1}{1 + \exp(\beta_0 + \sum_j \beta_j x_j)}$$

d=1 (x)

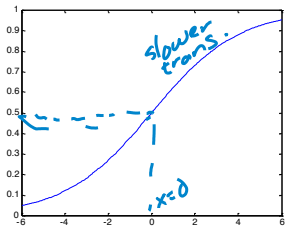
β_0=-2, β_1=-1



β_0=0, β_1=-1

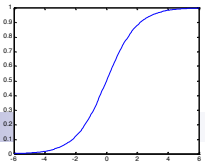


β_0=0, β_1=-0.5



Logistic Regression – a Linear classifier

$$\frac{1}{1 + exp(-z)}$$



$$P(y=0 \mid x, \beta) = g(\beta_0 + \sum_j \beta_j x_j) = \frac{1}{1 + \exp(\beta_0 + \sum_j \beta_j x_j)}$$

$\beta_0 + \sum_j \beta_j x_j > 0$

$\Rightarrow g(\cdot) < 0.5$

$\Rightarrow P(y=0 \mid x, \beta) < 0.5$

$\Rightarrow$ predict class 1

$\beta_0 + \sum_j \beta_j x_j = 0$ hyperplane

$\beta_0 + \sum_j \beta_j x_j < 0$

$\Rightarrow g(\cdot) > 0.5$

$\Rightarrow$ predict class 0

Very convenient!

$$p(y = 0 \mid x, \beta) = \frac{1}{1 + \exp(\beta_0 + \sum_j \beta_j x_j)}$$

implies

$$p(y = 1 \mid x, \beta) = \frac{\exp(\beta_0 + \sum_j \beta_j x_j)}{1 + \exp(\beta_0 + \sum_j \beta_j x_j)}$$

$1 - P(y=0 \mid x)$

Examine ratio:

$$\frac{p(y = 1 \mid x, \beta)}{p(y = 0 \mid x, \beta)} = \exp(\beta_0 + \sum_j \beta_j x_j) > 1$$

$\Rightarrow$ class 1 wins, else class 0 (under 0-1 loss)

linear classification rule!

implies

log odds

$$\log \frac{p(y = 1 \mid x, \beta)}{p(y = 0 \mid x, \beta)} = \beta_0 + \sum_j \beta_j x_j$$

linear

$> 0 \Rightarrow$ class 1 wins, as before

Loss Function: Conditional Likelihood

- Have a bunch of iid data of the form:

X: \$=0.05, CAPAVE=2 → Y: 'email'; \$=0.9, CAPAVE=10 → 'spam'

(x_1, y_1) (x_2, y_2) ⋮

(x_i, y_i) iid $i = 1, \dots, n$

$= (D_X, D_Y)$

- Discriminative (logistic regression) loss function:
  Conditional Data Likelihood

$$\arg\max_\beta p(D_Y \mid D_X, \beta) = \arg\max_\beta \log \prod_{i=1}^n p(y_i \mid x_i, \beta)$$

$$= \arg\max_\beta \sum_i \log p(y_i \mid x_i, \beta)$$

$$\log p(D_Y \mid D_X, \beta) = \sum_{i=1}^n \log p(y_i \mid x_i, \beta)$$

Expressing Conditional Log Likelihood

$$p(y=0 \mid x,\beta) = \frac{1}{1+\exp(\beta_0+\sum_j \beta_j x_j)}$$

$$p(y=1 \mid x,\beta) = \frac{\exp(\beta_0+\sum_j \beta_j x_j)}{1+\exp(\beta_0+\sum_j \beta_j x_j)}$$

$$l(\beta) = \sum_i \log p(y_i \mid x_i, \beta)$$

$$= \sum_i \begin{cases} \log p(y=1 \mid x_i,\beta) & y_i=1 \\ \log p(y=0 \mid x_i,\beta) & y_i=0 \end{cases}$$

$$l(\beta) = \sum_i y_i \log p(y=1 \mid x_i,\beta) + (1-y_i)\log p(y=0 \mid x_i,\beta)$$

$$= \sum_i y_i \log \frac{\exp(\beta_0+\sum \beta_j x_j)}{1+\exp(\beta_0+\sum \beta_j x_j)} + (1-y_i)\frac{1}{1+\exp(\beta_0+\sum \beta_j x_j)}$$

$$= \sum_i y_i(\beta_0+\sum \beta_j x_j) - \log\left(1+\exp(\beta_0+\sum \beta_j x_j)\right)$$

Maximizing Conditional Log Likelihood

$$p(y=0 \mid x,\beta) = \frac{1}{1+\exp(\beta_0+\sum_j \beta_j x_j)}$$

$$p(y=1 \mid x,\beta) = \frac{\exp(\beta_0+\sum_j \beta_j x_j)}{1+\exp(\beta_0+\sum_j \beta_j x_j)}$$

$$l(\beta) = \sum_i \log p(y_i \mid x_i, \beta)$$

$$= \sum_i y_i(\beta_0+\sum_j \beta_j x_{ij}) - \log(1+\exp(\beta_0+\sum_j \beta_j x_{ij}))$$

fixed in training data

$x \in \mathbb{R}^d$

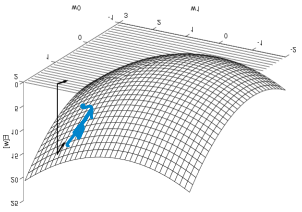
Good news: $l(\boldsymbol{\beta})$ is concave function of $\boldsymbol{\beta}$, no local optima problems

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

Good news: concave functions easy to optimize

Optimizing Concave Function – Gradient Ascent

- Conditional likelihood for logistic regression is concave
- Find optimum with gradient ascent



Gradient: $\nabla_\beta l(\beta) = \left[\frac{\partial l(\beta)}{\partial \beta_0}, \ldots, \frac{\partial l(\beta)}{\partial \beta_d}\right]'$

Step size, $\eta > 0$

Update rule: $\Delta\beta = \eta \nabla_\beta l(\beta)$

$$\beta_j^{(t+1)} \leftarrow \beta_j^{(t)} + \eta \frac{\partial l(\beta)}{\partial \beta_j}$$

new est. prev. est. stepsize

- Gradient ascent is simplest of optimization approaches
  - e.g., Conjugate gradient ascent can be much better

Often, esp. proofs, η gets smaller w/ iterations

e.g. $\eta_t = \frac{\alpha}{t}$ ← const.

Maximize Conditional Log Likelihood: Gradient ascent

$\frac{\partial}{\partial s} \ln f(s) = \frac{f'(s)}{f(s)}$ $\quad \frac{\partial}{\partial s} e^{f(s)} = f'(s) e^{f(s)}$

$$l(\beta) = \sum_{i=1}^{n} y_i (\beta_0 + \sum_{j=1}^{d} \beta_j x_{ij}) - \log(1 + \exp(\beta_0 + \sum_j \beta_j x_{ij}))$$

$$\nabla l(\beta): \quad \frac{\partial l(\beta)}{\partial \beta_j} = \sum_{i=1}^{n} y_i x_{ij} - \frac{x_{ij} \exp(\beta_0 + \sum_j \beta_j x_{ij})}{1 + \exp(\beta_0 + \sum_j \beta_j x_{ij})}$$

$j > 0$

$p(y=1 \mid x_i, \beta)$

$$= \sum_i x_{ij} (y_i - \hat{p}(y=1 \mid x_j, \beta))$$

weighted by contribution of j^{th} covariate to point i

how far is my prediction from the truth

Gradient Ascent for LR

start w/ $\beta^{(0)}$ (e.g. 0)

revisit soon

Gradient ascent algorithm: iterate until change < ε

$$\beta_0^{(t+1)} \leftarrow \beta_0^{(t)} + \eta \sum_i \left(y_i - \hat{p}(y = 1 \mid x_i, \beta^{(t)}) \right)$$

For *j*=1,…,*d*,

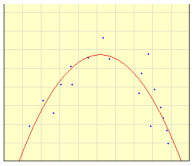
$$\beta_j^{(t+1)} \leftarrow \beta_j^{(t)} + \eta \sum_i x_{ij} \left(y_i - \hat{p}(y = 1 \mid x_i, \beta^{(t)}) \right)$$

repeat

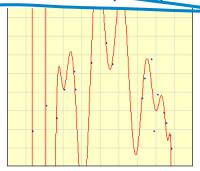
Regularization in Linear Regression

- Overfitting usually leads to very large parameter choices, e.g.:

-2.2 + 3.1 X – 0.30 X^2



-1.1 + 4,700,910.7 X – 8,585,638.4 X^2 + …

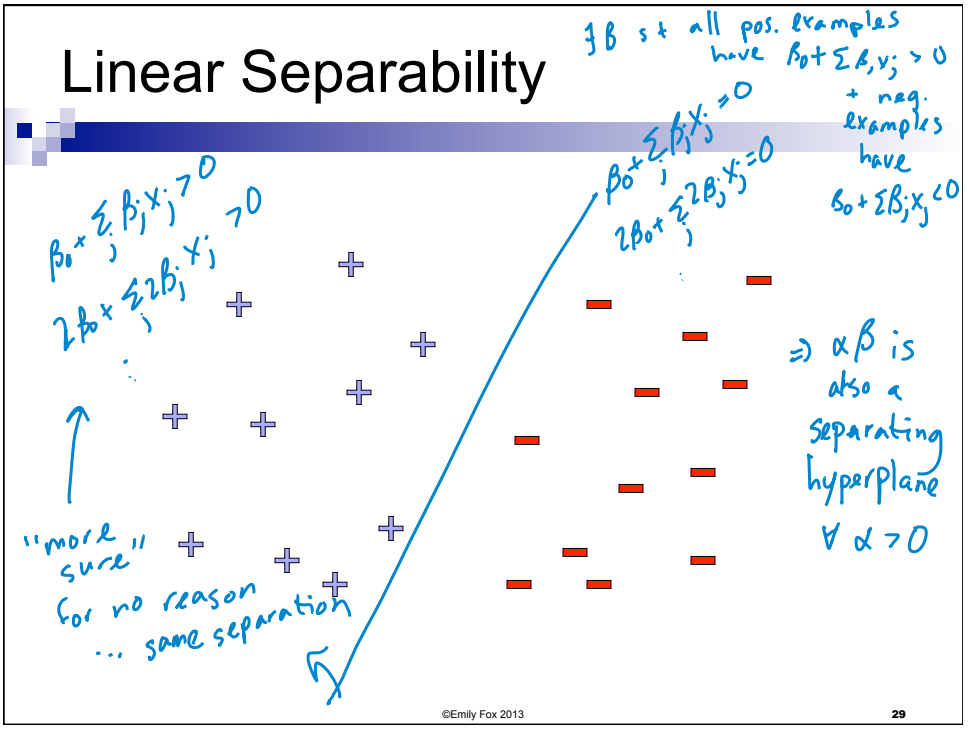


even for n ≈ p, p large

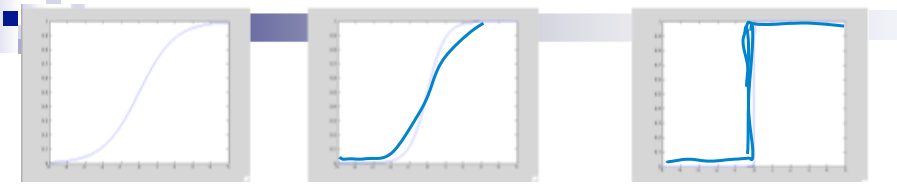
- ***Regularized*** or ***penalized*** regression aims to impose a "complexity" penalty by penalizing large weights
  - "Shrinkage" method

$$\hat{\beta} = \arg\min_{\beta} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta^T x_i))^2 + \lambda ||\beta||$$

Linear Separability



Large Parameters → Overfitting



$$\frac{1}{1+e^{-x}} \qquad \frac{1}{1+e^{-2x}} \qquad \frac{1}{1+e^{-100x}}$$

- If data is linearly separable, weights go to infinity

$$p(y=0 \mid \beta, x) = \frac{1}{1+e^{\beta_0 + \sum_j \beta_j x_j}}$$ increases as $\|\beta\| \to \infty$

  - In general, leads to overfitting: regularization → $\|\beta\|_2^2$ ★, $\|\beta\|_1$, ⋮
- Penalizing high weights can prevent overfitting…

Regularized Conditional Log Likelihood

- Add regularization penalty, e.g., L_2:

$$l(\beta) = \log \prod_{i=1}^{n} p(y_i \mid x_i, \beta) - \frac{\lambda}{2}||\beta||_2^2$$

log-like

$\sum_j \beta_j^2$

- Practical note about β_0:

don't regularize ... want to be able to put decision boundary anywhere

- Gradient of regularized likelihood:

just as before

$$\frac{\partial l(\beta)}{\partial \beta_j} = \frac{\partial}{\partial \beta_j} \log \prod_i p(y_i \mid x_i, \beta) - \lambda \beta_j$$

Standard v. Regularized Updates

- Maximum conditional likelihood estimate

$$\hat{\beta} = \arg\max_{\beta} \log \prod_{i=1}^{n} p(y_i \mid x_i, \beta)$$

w/o regularization

$$\beta_j^{(t+1)} \leftarrow \beta_j^{(t)} + \eta \sum_i x_{ij} \left(y_i - \hat{p}(y = 1 \mid x_i, \beta^{(t)}) \right)$$

- Regularized maximum conditional likelihood estimate

$$\hat{\beta} = \arg\max_{\beta} \log \prod_{i=1}^{n} p(y_i \mid x_i, \beta) - \frac{\lambda}{2} \sum_{j=1}^{d} \beta_j^2$$

pushes toward 0

$$\beta_j^{(t+1)} \leftarrow \beta_j^{(t)} + \eta \left\{ -\lambda \beta_j^{(t)} + \sum_i x_{ij} \left(y_i - \hat{p}(y = 1 \mid x_i, \beta^{(t)}) \right) \right\}$$

Stopping Criterion

β^* is opt. soln (no closed-form)

$$l(\beta) = \log \prod_{i=1}^{n} p(y_i \mid x_i, \beta) - \frac{\lambda}{2}||\beta||_2^2$$

- When do we stop doing gradient ascent?

$l(\beta^*) - l(\beta^{(t)}) < \epsilon$

- Because $l(\beta)$ is strongly concave:
  - i.e., because of some technical condition

$$l(\beta^*) - l(\beta) \leq \frac{1}{2\lambda}||\nabla l(\beta)||_2^2$$

don't know

- Thus, stop when: $\frac{1}{2\lambda}||\nabla l(\beta^{(t)})||_2^2 < \epsilon$

Digression: Logistic Regression for $K > 2$

- Logistic regression in more general case (K classes), where Y in $\{1,\ldots,K\}$

$(K-1)(d+1)$ params (when $K=2 \rightarrow d+1$ params)

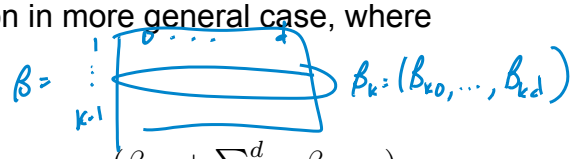
$\forall k \in \{1,\ldots,K\}$

$p(y=k \mid x, \beta) \propto e^{\beta_{k0} + \sum_j \beta_{kj} x_j}$

$p(y=K \mid x, \beta) = 1 - \sum_{k'=1}^{K} p(y=k' \mid x, \beta)$

Digression: Logistic Regression for *K* > 2

- Logistic regression in more general case, where *Y in* {1,…,*K*}



for *k*<*K*

$$p(y = k|\mathbf{x}, \beta) = \frac{\exp(\beta_{k0} + \sum_{j=1}^{d} \beta_{kj} x_j)}{1 + \sum_{k'=1}^{K-1} \exp(\beta_{k'0} + \sum_{j=1}^{d} \beta_{k'j} x_j)}$$

normalization const

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

$$p(y = K|\mathbf{x}, \beta) = \frac{1}{1 + \sum_{k'=1}^{K-1} \exp(\beta_{k'0} + \sum_{j=1}^{d} \beta_{k'j} x_j)}$$

Estimation procedure is basically the same as what we derived! slightly longer derivation

35