Extra Tutorial:
3:30 → 4:30 Friday (ICCS 193)

Monday, September 22, 2014 3:22 PM

Review Kernel Trick

(training) $X = \begin{bmatrix} & \\ & \end{bmatrix}$ $N \times d$ → basis expansion → $\Phi(X) = \begin{bmatrix} & \\ & \end{bmatrix}$ $N \times d^{100}$ based on Φ size

(testing) $\hat{X} = \begin{bmatrix} & \\ & \end{bmatrix}$ $T \times d$ → $\Phi(\hat{X}) = \begin{bmatrix} & \\ & \end{bmatrix}$ $T \times d^{100}$

Ridge regression

$\hat{Y} = \underbrace{\Phi(\hat{X})}_{T \times d^{100}} \underbrace{w_{MAP}}_{d^{100} \times 1}$

$w_{MAP} = \left(\Phi(X)^T \Phi(X)\right)^{-1} \Phi(X)^T Y$

$= \Phi(\hat{X}) \left(\Phi(X)^T \Phi(X) + \lambda I\right)^{-1} \Phi(X)^T Y$ ($d^{100} \times d^{100}$, $d^{100} \times N$, $N \times 1$)

$= \underbrace{\Phi(\hat{X})}_{T \times N} \Phi(X)^T \left(\underbrace{\Phi(X)\Phi(X)^T}_{N \times N} + \lambda I\right)^{-1} \underbrace{Y}_{N \times 1}$

$= \underbrace{K(\hat{X}, X)}_{T \times N} \left(\underbrace{K(X, X)}_{N \times N} + \lambda I\right)^{-1} \underbrace{Y}_{N \times 1}$

matrix inversion lemma (now based on examples, not features)
replace gram with kernel (under conditions)

If you can compute $K(X, X)$ and $K(\hat{X}, X)$,
no need for $\Phi(X)$, $\Phi(\hat{X})$ (don't have to store them)

$\Phi(X)\,\Phi(X)^T = \begin{bmatrix} \phi(\bar{x}_1)^T \phi(\bar{x}_1) & \phi(\bar{x}_1)^T \phi(\bar{x}_2) & \cdots \\ \phi(\bar{x}_2)^T \phi(\bar{x}_1) & \ddots & \\ \vdots & & \end{bmatrix}$ $\quad K(X, X) = \begin{bmatrix} k(\bar{x}_1, \bar{x}_1) & k(\bar{x}_1, \bar{x}_2) & \cdots \\ k(\bar{x}_2, \bar{x}_1) & \ddots & \\ \vdots & & \end{bmatrix}$

Valid Kernels

mercers theorum

Q: When does there exist feature map ϕ, such that for kernel k we have $\phi(\bar{x}_i)^T \phi(\bar{x}_j) = k(\bar{x}_i, \bar{x}_j)$??

A: If $k(X_1, X_2) \succeq 0$ (positive semi definite) for all X_1, X_2
(may be hard to show)

Checking a valid Kernel

Assume k is valid:

- ck $c > 0$
- $k + k'$
- $k(\phi(x), \phi(x'))$
- $\exp(k)$
- $q(k)$ (polynomial) non negative coefficients
- $f(x)\, k(x, x')\, f(x')$

Course project ideas
- take your application, design a kernel
- scale up to large 'N'
- learning a kernel (MKL) multiple kernel learning

remember for assignment: laplace prior

Search over Feature Combinations

-NP Hard
Greedy Methods:

- forward selection: start with empty model, add best next (BIC, Cross Validation)
  (sub-modular: best approximation)
- backwards selection: start with all, remove the worst
- stagewise: combine forward + backwards

L1 - Regularization

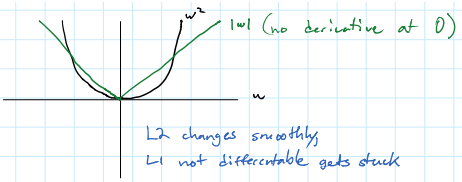
$\|\bar{w}\|_1 = \sum_{i=1}^{d} |w_i|$

$$\underset{w}{\operatorname{argmin}} \; \frac{1}{2}\|Xw - Y\|^2 + \lambda \|w\|_1$$

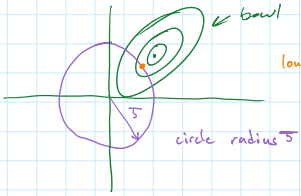
quadratic — linear

- regularization: protect against overfitting (but not unique)
- encourages variables to be exactly 0
  - i.e. not dependant on

$$Xw = \begin{bmatrix} & j & \\ & \vdots & \\ & \text{no longer matters} & \end{bmatrix} \begin{bmatrix} \\ \\ \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ -j \end{bmatrix}$$



$$\underset{w}{\operatorname{argmin}} \; \frac{1}{2}\|Xw - Y\|^2 \quad \text{s.t.} \quad \|w\|_1 \leq \tau$$



(Smooth) + (non smooth) — still easy optimization

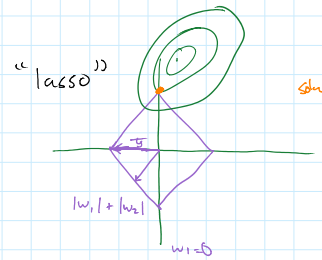
but seperable

$$f(x) = \sum_{i=1}^{d} f_i(x_i)$$

Elastic net: $\frac{\lambda_2}{2}\|w\|^2 + \lambda_1 \|w\|_1$

small, for unique

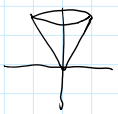
unique — sparse



Group L_1 Regularization: $\lambda \sum_g \|w_g\|$ — w_g is a sub vector

selects "groups" of variables

L_1 of groups



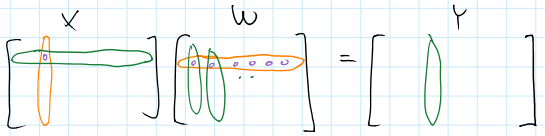
still non differentiable at 0

$[\ldots\ldots \underbrace{0\,0\,0}_{w_g} \ldots]$

$$X = \begin{bmatrix} & \\ & \\ & \end{bmatrix} \quad N \times d \qquad Y = \begin{bmatrix} & \\ & \\ & \end{bmatrix} \quad N \times c$$

"multiple regression"

$$W = \begin{bmatrix} & \\ & \end{bmatrix} \qquad XW = Y$$



- Nuclear-norm regularization
  - sum of singular values
  - => low rank matrices (many go to zero)
- Structured Sparcity ($w_2 \neq 0$ only if $w_1 \neq 0$, etc)
  - challenging, but good project (potential publication)
  - patterns of variables

Supervised Learning

- generative $p(y_i, x_i)$
  - Naive Bays
  - GDA
- discriminative $p(y_i | x_i)$
  - least squares
  - [logistic regression]
- discriminant function
  - knn
  - [support vector machine]

Classification:

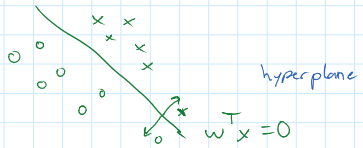
features: $\bar{x}_i \in \mathbb{R}^d$

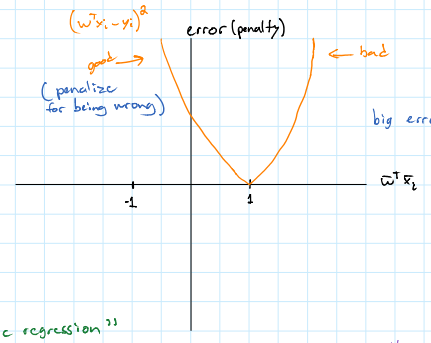
target: $y_i \in \{-1, 1\}$

Linear Classifier:

$$\hat{y}_i = \text{sign}(w^T \bar{x}_i)$$

sign: $w^T x_i > 0 \Rightarrow \hat{y}_i = 1$ (1 if positive)
$w^T x_i < 0 \Rightarrow \hat{y}_i = -1$ (-1 if negative)

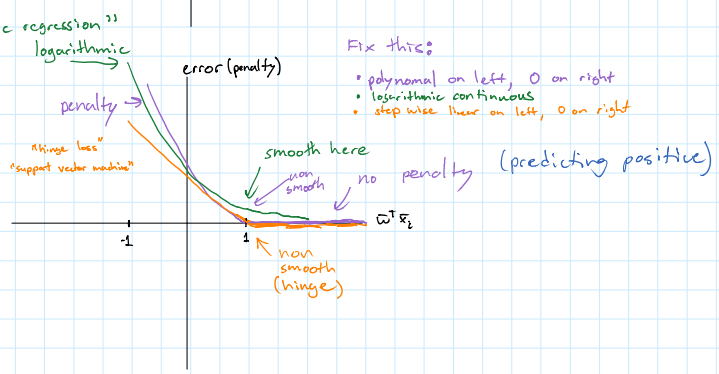




$\sum_{i=1}^{n} f_i(w^T x_i, y_i)$

big error if predicting negative if classifier positive

penalize for being really right?
extremely likely should not have extreme penalty



Odds ratio:

$$\frac{p(\bar{y}_i = 1 \mid \bar{x}_i, \bar{w})}{p(y_i = -1 \mid \bar{x}_i, \bar{w})}$$

Log odds ratio:

$$\log\left(\frac{p(\bar{y}_i = 1 \mid \bar{x}_i, \bar{w})}{p(y_i = -1 \mid \bar{x}_i, \bar{w})}\right) = \bar{w}^T \bar{x}_i$$

$$\frac{p(\bar{y}_i = 1 \mid \bar{x}_i, \bar{w})}{p(y_i = -1 \mid \bar{x}_i, \bar{w})} = \exp(\bar{w}^T x_i)$$

$p(y_i = -1 \mid \bar{x}_i, \bar{w}) = 1 - p(y_i = 1 \mid \bar{x}_i, \bar{w})$

$$p_1 = (1 - p_1)\exp(\bar{w}^T \bar{x}_i)$$

$$p_1(1 + \exp(\bar{w}^T \bar{x}_i)) = \exp(\bar{w}^T x_i)$$

$$p_1 = \frac{\exp(\bar{w}^T \bar{x}_i)}{1 + \exp(\bar{w}^T \bar{x}_i)} = \frac{1}{1 + \exp(-\bar{w}^T \bar{x}_i)}$$

"Sigmoid Function"

$$p_2 = \frac{1}{1 + \exp(\bar{w}^T \bar{x}_i)}$$