8th Jan

TUTORIAL

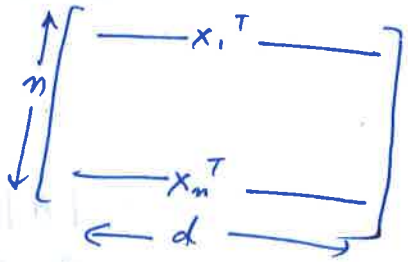
'n' points

$x_i \in R^{d \times 1}$

$w \in R^{d \times 1}$

$y_i \in R$ (a single number)

$Y \in R^{n \times 1}$

$X =$ 

Matrix algebra

derivative of inner product (vector $\rightarrow w$): $\frac{\partial}{\partial w}\left[a^T w\right] = \frac{\partial}{\partial w}\left[\sum_{i=1}^{d} a_i w_i\right]$

derivative wrt each component of the vector & then stack them up

$$= \begin{bmatrix} \frac{\partial}{\partial w_1}\left[\sum_{i=1}^{d} a_i w_i\right] \\ \vdots \\ \frac{\partial}{\partial w_d}\left[\sum_{i=1}^{d} a_i w_i\right] \end{bmatrix} = \begin{bmatrix} a_1 \\ \vdots \\ a_d \end{bmatrix} = a$$

$$\frac{\partial^2 f}{\partial w^2} = \frac{\partial}{\partial w}\left[\frac{\partial f}{\partial w}\right] \Rightarrow \frac{\partial^2 [w^T A w]}{\partial w^2} = \frac{\partial}{\partial w}\left[(A + A^T) w\right] = A + A^T$$

$$\nabla^2 f = \begin{bmatrix} \frac{\partial^2 f}{\partial w_1^2} & \frac{\partial^2 f}{\partial w_1 \partial w_2} & \cdots & \frac{\partial^2 f}{\partial w_1 \partial w_d} \\ & \ddots & & \\ & & & \frac{\partial^2 f}{\partial w_d^2} \end{bmatrix} \begin{matrix} \uparrow \\ d \\ \downarrow \end{matrix}$$

$\leftarrow \quad d \quad \rightarrow$

$$\frac{\partial}{\partial x}\left[x^T A x\right] = (A + A^T)$$

$$\frac{\partial}{\partial x}\left[a^T x b\right] = ab^T$$

Matrix cookbook for other rules.

$$tr(A) = \sum_i A_{ii} = \sum_i \lambda_i$$

Trace is sum of diagonal elements / eigen values.

$$\lambda_i = eig(A)$$

$$|A| = \prod \lambda_i$$

$$tr(ABC) = tr(CAB) = tr(BCA)$$ [cyclically equal]

$$\nabla f(w) = 0$$

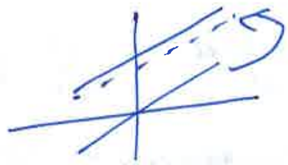
$$\Rightarrow \quad w = (X^T X)^{-1} X^T y$$

In MATLAB:

$Ax = b$

$x = A \backslash B$

Adding bias:



to be able to increase expressive power, that is to incorporate the y intercept term, we add a bias variable.

$$\hat{y_i} = w^T x_i + w_0$$

$$= \begin{bmatrix} w_0 \\ w \end{bmatrix}^T \begin{bmatrix} 1 & x_i \end{bmatrix}$$

$$\hat{y_i} = w^T \begin{bmatrix} 1 & x_i \end{bmatrix}$$

$$\boxed{w = \text{argmin} \, \| X_{bias} w - y \|^2}$$ (*)

$$X_{bias} = \underbrace{\begin{bmatrix} 1 & - x_1 \rightarrow \\ \vdots & \\ 1 & - x_n \rightarrow \end{bmatrix}}_{d+1} \Bigg\updownarrow n$$

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^p \\ & & & & \\ 1 & x_n & x_n^2 & \cdots & x_n^p \end{bmatrix} \rightarrow \text{if } d=1,\ X_{poly} \in R^{n \times (p+1)}$$

if $d=2$ $\begin{bmatrix} 1 & x_{1,1} & x_{1,2} & x_{1,1}^2 & x_{1,2}^2 & x_{1,1}x_{1,2} \end{bmatrix}$

$p \rightarrow$ degree of polynomial for basis expansion

$d \rightarrow$ no. of features in X $[\ x_i \in R^{d \times 1}\ ]$

$x_{i,j} \rightarrow j$th feature of the ith point.

if $d > 1 \rightarrow$ use polynomial kernel.

RBF kernel $\rightarrow \infty$ basis expansion.

As $p \uparrow$, model complexity $\uparrow$, training err $\downarrow$, generalization $\rightarrow$ worse, test error $\rightarrow$ high $\Big\}$ $\rightarrow$ overfitting

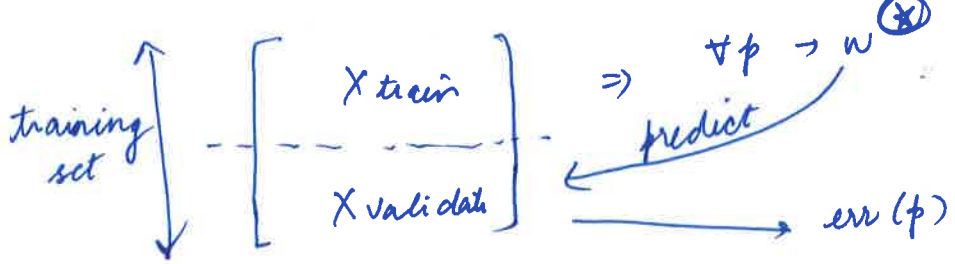
simple model, training error $\uparrow$, generalization $\rightarrow$ good, test err $\uparrow$ $\Big\}$ $\rightarrow$ underfitting

$\therefore$ we have so much data, our model will not overfit, even the very complex ones (deep neural n/w s for instance have become popular now because we have a lot of data now).

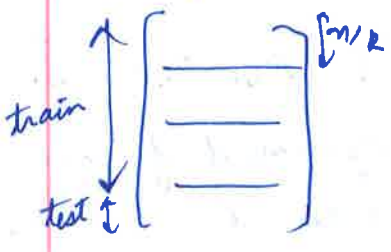
<< REFER or READ : VC- Dimension, Rademacher complexity >>

CROSS VALIDATION - choosing 'p':

split your training set into X_{train} & $X_{validate}$ & $X_{validate}$ will be used as a proxy for the 'test' set.



K-fold cross validation



$\Rightarrow$ do this k times. We will get 'k' errors. Average over all these 'k' errors $\Rightarrow$ $\frac{\text{sum of errors}}{k}$

This is to be done for all 'p's'

Choose the 'p' with the minimum average error.

$p^* = \text{argmin}$ (all avg errors)

Once you get $p^* \rightarrow$ train $\rightarrow$ get 'w'

Regularization

$$\min_w \|Xw - y\|^2 + \underbrace{\frac{\lambda}{2}\|w\|^2}_{L2 \text{ regularization}}$$

$$(X^TX + \lambda I)^{-1} X^T y$$

improves conditioning number.

$\lambda = 1e^{-2}$

condition number tells you how easy it is to invert a matrix

- unique solution.