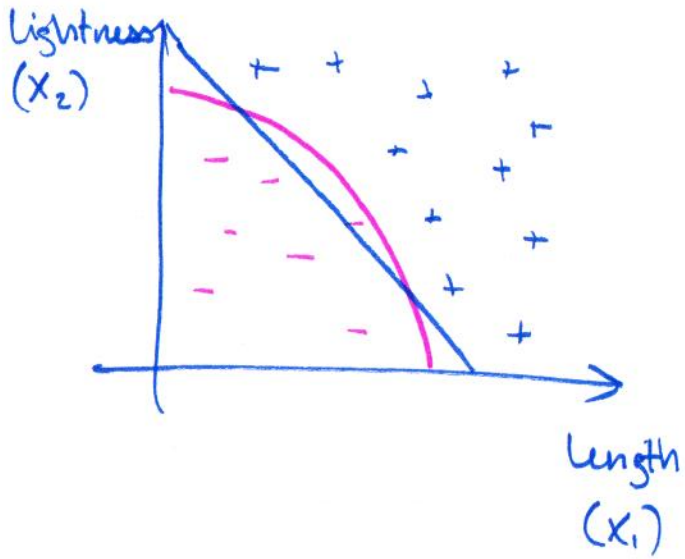
Lecture 25: Kernel Regression



$$\hat{y} = w_1 \cdot x_1 + w_2 \cdot x_2 \cancel{+ w_0} = \underline{w}^T \underline{x}$$

$$\tilde{y} = \text{sign}(\hat{y})$$

— or —

$$\hat{y} = w_1 x_1 + w_2 \cdot x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2$$

$$\Rightarrow \tilde{x} = \phi(x) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix} \Rightarrow \hat{y} = \underline{w}^T \underline{\tilde{x}}$$

Kernel methods formalize this idea.

original feature vector x $\xrightarrow[\text{map to}]{}$ $\phi(x)$ "higher dimensional feature space"

$x \in \mathbb{R}^p$ $\quad$ $\phi(x) \in \mathbb{R}^d$ $\quad$ $d > p$

goal: find weights s.t. $\hat{y} = w^T \phi(x)$ gives good predictions. (*)

when ϕ corresponds to quadratic monomials of p-dim vectors, then $d = O(p^2)$

Kernel methods allow us to do (*) without computing $\phi(x_i)$ explicitly for all training samples x_i.

Recall Ridge Regression

$$\min_w \|y - Xw\|_2^2 + \lambda \|w\|_2^2$$

Let $X = U\Sigma V^T$ (full SVD) $\Rightarrow U^T = U^{-1} = I$, $VV^T = V^TV = I$

$X \in \mathbb{R}^{n \times p}$

$$\Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix}$$

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

$$= (V\Sigma^T\Sigma V^T + \lambda VV^T)^{-1} V\Sigma^T U^T y$$

$$= V(\Sigma^T\Sigma + \lambda I)^{-1} V^T V \Sigma^T U^T y$$

$$= \underline{V}\underline{(\Sigma^T\Sigma + \lambda I)^{-1} \Sigma^T U^T y}$$

$\Rightarrow$ $\hat{w}$ = weighted sum of columns of V
= weighted sum of rows of X $\Rightarrow \hat{w} = X^T\alpha$ for some $\alpha \in \mathbb{R}^n$

First, note

$$(\Sigma^T\Sigma + \lambda I)^{-1}\Sigma^T = \Sigma^T(\Sigma\Sigma^T + \lambda I)^{-1}$$

Then $\hat{w} = V\Sigma^T(\Sigma\Sigma^T + \lambda I)^{-1}U^T y$

($U^TU = I$)

$$= \underbrace{V\Sigma^T U^T}_{= X^T}\underbrace{U(\Sigma\Sigma^T + \lambda I)^{-1}U^T y}_{= \alpha}$$

$$= X^T U(\Sigma\Sigma^T + \lambda I)^{-1}U^T y$$

$$= X^T(U\Sigma\Sigma^T U^T + \lambda I)^{-1} y$$

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

$$\Rightarrow \boxed{\hat{w} = X^T\alpha, \text{ where } \alpha = (XX^T + \lambda I)^{-1} y}$$

$$\hat{y} = x^T\hat{w}$$

$$= x^T X^T \alpha$$

$$= \sum_{i=1}^{n} \alpha_i \, x^T x_i$$

Let $K = XX^T$ (linear) (kernel matrix)

$K_{ij} = x_i^T x_j$

$\Rightarrow \hat{w} = (K + \lambda I)^{-1} y$

Now work in "feature space"

Let ϕ be a nonlinear ~~fcnn~~ function mapping from $\mathbb{R}^p$ to $\mathbb{R}^d$

define matrix $\Phi = [\phi(x_1) \quad \phi(x_2) \quad \dots \quad \phi(x_n)]$

e.g. if $\phi(x) = x$, then $\Phi = X$

e.g. if $\phi\left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}\right) = \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 x_3 \\ x_2^2 \\ x_2 x_3 \\ x_3^2 \end{bmatrix}$

Regression in Kernel space:

$$\hat{w} = \underset{w}{\operatorname{argmin}} \| y - \Phi w \|_2^2 + \lambda \| w \|_2^2$$

$$\hat{w} = \Phi^T \alpha, \quad \alpha = \left(\Phi \Phi^T + \lambda I \right)^{-1} y$$

define $K = \Phi \Phi^T$, $K_{ij} = \phi(x_i)^T \phi(x_j)$

$$\Rightarrow \alpha = \left(K + \lambda I \right)^{-1} y$$

$K(x, x_i)$

$$K = \left(X X^T \right)^{\odot q} \in \mathbb{R}^{n \times n}$$

new sample prediction

$$\hat{y} = \sum_{i=1}^{n} \alpha_i \, \phi(x)^T \phi(x_i)$$

e.g. $\underline{x} \in \mathbb{R}^2 \Rightarrow x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$$\phi(x) = \begin{bmatrix} x_1^2 \\ \sqrt{2}\, x_1 x_2 \\ x_2^2 \end{bmatrix}$$

$$\phi(x_i)^T \phi(x_j) = x_{i1}^2 x_{j1}^2 + 2 x_{i1} x_{i2} x_{j1} x_{j2} + x_{i2}^2 x_{j2}^2$$

$$= \left(x_{i1} x_{j1} + x_{i2} x_{j2} \right)^2$$

$$= \left(x_i^T x_j \right)^2$$

if $\phi(x) \in \mathbb{R}^d$, then computing this takes $O(d)$ time

if $x_i \in \mathbb{R}^p$ then computing this takes $O(p)$ time

Recall $p << d$

for ϕ above, $d \approx O(p^2)$

Popular Kernels:

Kernel function: $\phi(x_i)^T \phi(x_j) = K(x_i, x_j)$

- Polynomials of degree q:

$$K(x_i, x_j) = (x_i^T x_j)^q$$

- Polynomials of degree up to q:

$$K(x_i, x_j) = (x_i^T x_j + 1)^q$$

- Gaussian/Radial kernels

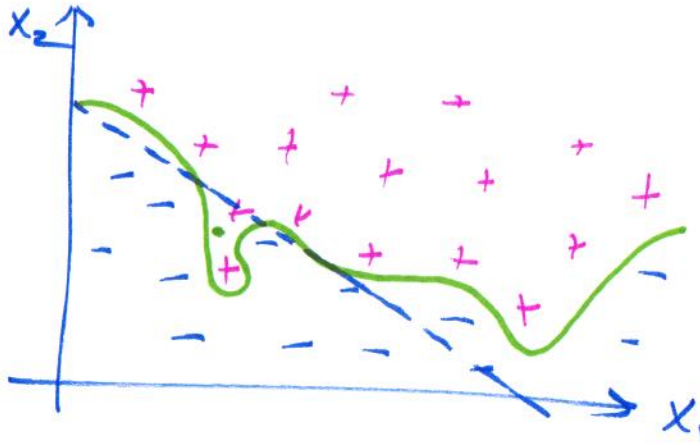
$$K(x_i, x_j) = \exp\left(\frac{-\|x_i - x_j\|_2^2}{2\sigma^2}\right), \quad d = \infty$$

Choosing a kernel:

(A). Kernel regression gives linear boundary in high-dimensional feature space

$\Leftrightarrow$ nonlinear boundary in original space.

(B) Looking for perfect classifier kernel can be bad



risk "overfitting"