Kernel Methods

Terminology

$\phi : \mathbb{R}^d \to \mathbb{R}^p$ feature map

attributes x features $\phi(x)$

What to do if p is very large?

$$\phi(x) = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \\ x_1^2 \\ \vdots \\ x_i x_j \\ \vdots \\ x_d^2 \\ x_1^3 \\ \vdots \\ x_i x_j x_k \\ \vdots \\ x_d^3 \end{bmatrix} \begin{matrix} \} d \\ \} d^2 \\ \} d^3 \end{matrix}$$

$$\theta^T \phi(x) = \_ \cdot 1 + \_ x_1 + \_ x_2 + \_\, x_i x_j + \_\, x_i x_j x_k$$

LMS using kernel trick

$$\min_\theta \frac{1}{2} \sum_{i=1}^n \left(y^{(i)} - \theta^T \phi(x^{(i)})\right)^2$$

Gradient Descent Loop:

$$\theta := \theta + \alpha \sum_{i=1}^n \left(y^{(i)} - \theta^T \phi(x^{(i)})\right) \phi(x^{(i)})$$

Problem: $\phi(x)$ is high dimensional

$p = 1 + d + d^2 + d^3 \quad O(d^3)$

$d \sim 10^3 \Rightarrow p \sim 10^9$

Runtime for 1 iteration of GD is $O(np)$

Key observation:

If θ initialized as 0, then at any time,

$\theta = \sum_{i=1}^n \beta_i \phi(x^{(i)})$ for $\beta_1 \dots \beta_n \in \mathbb{R}$

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

New algo: update β

$$\beta_i := \beta_i + \alpha\Big(y^{(i)} - \underbrace{\sum_{j=1}^{n}}_{n} \beta_j \underbrace{\langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle}_{p}\Big)$$

① Precompute $\langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$

$$\langle a, b \rangle = \sum_{i=1}^{n} a_i b_i$$

② $\langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$ can be computed faster than by explicitly computing $\phi(\cdot)$

e.g. cubic polynomials

$$\phi(x) = \begin{array}{cc} \begin{bmatrix} 1 \\ x_i \\ x_i x_j \\ x_i x_j x_k \end{bmatrix} & \begin{array}{c} 1 \\ d \\ d^2 \\ d^3 \end{array} \end{array}$$

$$\langle \phi(x), \phi(z) \rangle = [1 \cdots x_i \cdots x_i x_j \cdots x_i x_j x_k] \begin{bmatrix} 1 \\ \vdots \\ z_i \\ \vdots \\ z_i z_j \\ \vdots \\ z_i z_j z_k \end{bmatrix}$$

$$= 1 + \sum_{i=1}^{d} x_i z_i + \overbrace{\sum_{i=1, j=1}^{d} x_i x_j z_i z_j}^{p} + \overbrace{\sum_{\underbrace{i=1, j=1, k=1}_{O(d^3)}}^{n} x_i x_j x_k z_i z_j z_k}^{q}$$

But: $\displaystyle\sum_{i=1, j=1}^{n} u_i w_j = \Big(\sum_{i=1}^{d} u_i\Big)\Big(\sum_{j=1}^{d} w_j\Big)$

$$p = \Big(\sum_{i=1}^{d} x_i z_i\Big)\Big(\sum_{i=1}^{d} x_j z_j\Big) = \langle x, z \rangle^2 \quad O(d) \text{ time}$$

$$q = \Big(\sum_{i=1}^{d} x_i z_i\Big)\Big(\sum_{j=1}^{d} x_j z_j\Big)\Big(\sum_{k=1}^{d} x_k z_k\Big) = \langle x, z \rangle^3 \quad O(d) \text{ time}$$

$$\langle \phi(x), \phi(z) \rangle = 1 + \langle x, z \rangle + \langle x, z \rangle^2 + \langle x, z \rangle^3 \quad O(d) \text{ time}$$

In general:

$$K(x,z) = \langle \Phi(x), \Phi(z) \rangle$$

$K(\cdot,\cdot)$ is a kernel function

$K: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

Compute $K(x^{(i)}, x^{(j)}) \quad \forall i,j$

n^2 entries $O(n^2 d)$ time

Loop {

$$\beta_i := \beta_i + \alpha\left(y^{(i)} - \sum_{j=1}^{n} \beta_j \langle \Phi(x^{(i)}), \Phi(x^{(j)}) \rangle\right)$$

$$= \beta_i + \alpha\left(y^{(i)} - \sum_{j=1}^{n} \beta_j K(x^{(i)}, x^{(j)})\right)$$

}

$K \in \mathbb{R}^{n \times n}$ kernel matrix

$K_{ij} = K(x^{(i)}, x^{(j)})$

$\beta := \beta + \alpha(\vec{y} - K\beta)$ $\quad$ $O(n^2)$ time

Test time

Given x, predict $\theta^T \Phi(x)$

$$\theta^T \Phi(x) = \left(\sum_{i=1}^{n} \beta_i \Phi(x^{(i)})\right)^T \Phi(x)$$

$$= \sum_{i=1}^{n} \beta_i \langle \Phi(x^{(i)}), \Phi(x) \rangle$$

$$= \sum_{i=1}^{n} \beta_i K(x^{(i)}, x)$$

linear in # exs
independent of p

Time complexity

Training: Preprocessing $O(n^2 d)$
Training $O(n^2) \times$ # iters

Test time: $O(nd)$ assuming $K(\cdot,\cdot)$ can be computed in $O(d)$ time

Deeper observation

→ Only thing needed is $K(\cdot,\cdot)$

function K is valid kernel fn

if $\exists \Phi$. $K(x,z) = \langle \Phi(x), \Phi(z) \rangle$

Other algos can also be kernelized

e.g. perceptron, LR

→ algo for linear $\theta^T x$

→ replace x by $\Phi(x)$

→ rewrite algo s.t. only depends on $\langle \Phi(x), \Phi(z) \rangle$

Kernel fns:

$K(x,z) = 1 + x^T z + (x^T z)^2 + (x^T z)^3$

$K(x,z) = (x^T z)^2$

$K(x,z) = (x^T z + c)^2$ $\qquad \Phi(x) = \begin{bmatrix} c \\ \sqrt{2c}\, x_1 \\ \vdots \\ x_i x_j \end{bmatrix}$

polynomial kernel

$K(x,z) = (x^T z + c)^k \quad \sim \binom{d+k}{k}$ monomials

$K(x,z) = \exp\left(\frac{-\|x-z\|^2}{2\sigma^2}\right) = \langle \Phi(x), \Phi(z) \rangle$

Φ: ∞-dimensional

Kernel validity

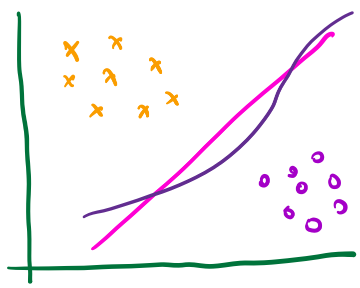
Necessary condition:

n exs $x^{(1)} \ldots x^{(n)}$

kernel matrix $K_{ij} = K(x^{(i)}, x^{(j)})$

kernel matrix is PSD ($K \succeq 0$; $z^T K z \geq 0 \quad \forall z \in \mathbb{R}^n$)

Support Vector Machines (SVM)



$\{x : w^T x + b = 0\}$ by reg

kernel methods w/ feature map

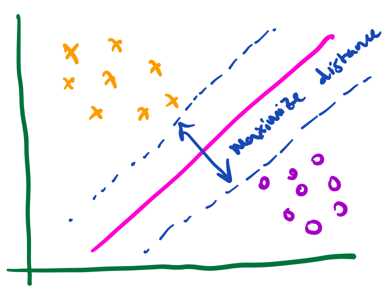
with SVM: let $y^{(i)} \in \{-1, 1\}$

Find w, b s.t.

If $y^{(i)} = 1, \quad w^T x^{(i)} + b > 0 \qquad (1)$

$\quad\; y^{(i)} = -1, \quad w^T x^{(i)} + b < 0 \qquad (2)$

Choose w, b that give most separation



amongst all (w, b) pairs that satisfy (1), (2) $\left(y^{(i)}(w^T x^{(i)} + b) > 0 \;\forall i\right)$

$$\max_{w,b} \left[\min_i \text{dist}(x^{(i)}, \text{decision boundary}) \right]$$

$$= \max_{w,b} \left[\min_i \frac{|w^T x^{(i)} + b|}{\|w\|_2} \right]$$

$$= \max_{w,b} \left[\min_{i \in \{1 \dots n\}} \frac{y^{(i)}(w^T x^{(i)} + b)}{\|w\|_2} \right] \Leftarrow \text{objective function}$$

Note: scaling invariant

Non-trivial facts (need KKT condition)

① optimal soln w^*, b^* satisfies

$$w^* = \sum_{i=1}^{n} \alpha_i x^{(i)} y^{(i)} \qquad \alpha_i \in \mathbb{R}^+$$

② $\alpha_i = (\alpha_1 \cdots \alpha_n)$ is optimal soln of program

$$W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle$$

$$\text{s.t. } \alpha_i \geq 0$$

$$\sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$

$$w^* = \sum_{i=1}^{n} \alpha_i \phi(x^{(i)}) y^{(i)}$$

test time: $w^{*T} \phi(x) = \sum_{i=1}^{n} \alpha_i \langle \phi(x^{(i)}), \phi(x) \rangle y^{(i)}$

$$= \sum_{i=1}^{n} \alpha_i K(x^{(i)}, x) y^{(i)}$$