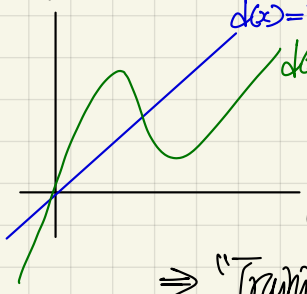
§ Kernel Regression.

(*) High-dim feature spaces

(*) Popular Kernels

- Higher dimensional feature spaces extend regressions.



let $x=[x_1 \cdots x_M]' \subseteq \mathbb{R}^M$

Consider. $d(x)=\phi^T(x)W$, where $\phi(x) \subseteq \mathbb{R}^P$ $(P>M)$

eg $x=[x_1\ x_2]'$

$\phi^T(x)=[x_1^2\ x_2^2\ \sqrt{2}x_2x_1\ x_1\ x_2\ 1]$

$\Rightarrow$ "Training data": $x^i, d^i;\ i=1,\cdots,N$

$\Rightarrow$ "Find W": $\min_W \sum_{i=1}^{N}(d^i-\phi^T(x^i)W)^2+\lambda\|W\|_2^2$ (Ridge)

Defining $d\equiv[d^1\cdots d^N]'$

$\Phi\equiv[\phi(x^1)\cdots\phi(x^N)]'$ $N\times P$

$\Rightarrow \min_W \|d-\Phi W\|_2^2+\lambda\|W\|_2^2$

$\Rightarrow W=(\Phi'\Phi+\lambda I)^{-1}\Phi' d$ (Standard Ridge)

- Regression is a weighted sum of "kernels".

$$d(x)=\phi^T(x)W=\phi^T(x)\underbrace{(\Phi'\Phi+\lambda I)^{-1}}_{P\times P}\Phi' d$$

Since $(\Phi'\Phi+\lambda I)^{-1}\Phi'=\Phi'(\Phi\Phi'+\lambda I)^{-1}$

we have $d(x)=\phi^T(x)\Phi'\underbrace{(\Phi\Phi'+\lambda I)^{-1}}_{N\times N}d$

① $[\Phi\Phi']_{ij}=\phi^T(x^i)\phi(x^j)$

② $[\phi^T(x)\Phi]_j=\phi^T(x)\phi(x^j)$

$\Rightarrow$ define Kernel: $K(u,v)=\phi^T(u)\phi(v)$

$\Rightarrow$ let $\alpha=[\alpha_1\cdots\alpha_N]'$ then $d(x)=\phi^T(x)\Phi'\alpha=\sum_{i=1}^{N}\alpha_i\phi^T(x)\phi(x^i)$

$=(\Phi\Phi'+\lambda I)^{-1}d$ (easier computation)

$=\sum_{i=1}^{N}\alpha_i K(x,x^i)$ (*) weighted sum of $K(x,x^i)$

- Kernel methods find $d(x)$ without computing $\phi(x)$

We have $d(x) = \sum_{i=1}^{N} \alpha_i K(x, x^i)$ where $\alpha \equiv (\Phi\Phi^T + \lambda I)^{-1} d$

$\equiv (R + \lambda I)^{-1} d$

$[R]_{ij} = \phi^T(x^i)\phi(x^j) = K(x^i, x^j)$

$\Rightarrow$ R can be computed efficiently!

eg. R being "Monomials of degree q": $\phi(x) \to x_1^q, x_1^{q-1}x_2, \cdots x_3^{q-5}x_6^2 x_8^3$ (power: degree sum to q)

then $K(u,v) = \phi^T(u)\phi(v) = (u^T v)^q$ (inner prod. raise to q power)

$O(p)$ $O(M)$

$\Rightarrow p = \frac{(q+M-1)!}{q!(M-1)!}$ terms ($\uparrow$ when M or q $\uparrow$)

@ Suppose $\begin{cases} M=10, q=5 \to p \sim 3000 \\ M=100, q=5 \to p \sim 10^8 \end{cases}$

Computing: $O(p)$ vs $O(M)$

Memory: $O(Np)$ vs $O(N^2)$

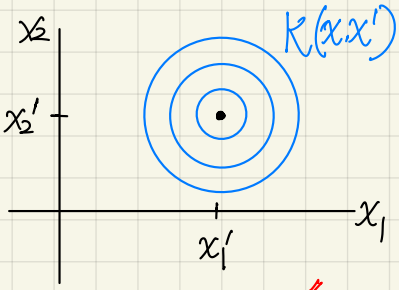
- Popular choices of Kernels (depend on similarity of u, v)

Note that $u^T v = \|u\|_2 \|v\|_2 \cos\theta$

(i) Monomial of degree q: $K(u,v) = (u^T v)^q$

(ii) Polynomials up to degree q: $K(u,v) = (u^T v + 1)^q$

★ (iii) Gaussian/Radial Kernel: $K(u,v) = \exp\left(\frac{-\|u-v\|_2^2}{2\sigma^2}\right)$ "how close u & v are"



* Not required explicit form of $\phi(x)$
* An infinite polynomial orders
* Smoothness: Controlled by σ

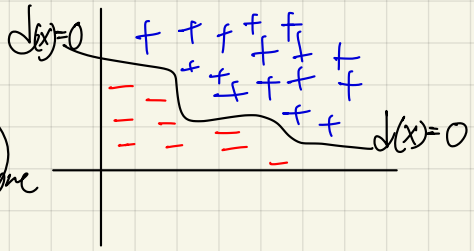
$\Rightarrow$ desired choice of Kernel since it generalizes regression with high-dim spaces (semiparametric)

- Kernel Regression Considerations.

$$d(x) = \phi^T(x) w \longleftrightarrow d(x) = \sum_{i=1}^{N} \alpha_i K(x, x^i) \quad \text{(Kernel)}$$

(+) Store & compute: w $(p \times 1)$ vs. α $(N \times 1)$

(*) Binary classifier: $\text{sign}(d(x))$ (more complex with high-dim space)



(*) WANT: Avoid overfitting with high-dim feature spaces
[challenge of working with high-dim]

$\Rightarrow$ Choose σ^2 parameter.

$\Rightarrow$ with CV.

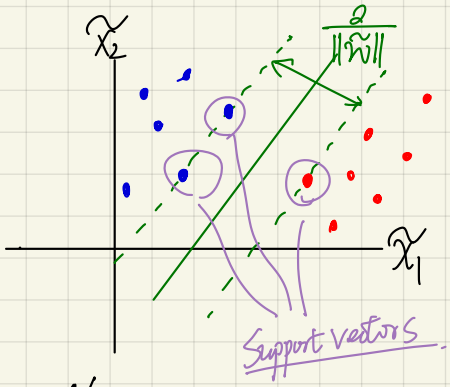
§ Kernel-based SVM.

(*) linear max margin classifier.

(*) Kernel version of Hinge Loss w/ Ridge Regression.

- SVM defines "max-margin" classifier.

$\Rightarrow \begin{cases} \tilde{x}'\tilde{w} + \underbrace{b}_{\text{const.}} = 0 \\ \text{or } x'w = 0 \end{cases}$ where $w = [\tilde{w}' \; b]'$ depends only on Support Vectors



Recall: kernel regression.

$$d(x) = \underbrace{\phi^T(x)}_{\text{"high-dim feature space"}} w = \sum_{j=1}^{N} \alpha_j \underbrace{K(x, x^j)}_{\text{"kernel"}}$$

Let $\phi^T(x) = x \Rightarrow K(x, x^j) = x'x^j$

$$\Rightarrow d(x) = \sum_{j=1}^{N} \alpha_j K(x, x^j) = \sum_{j=1}^{N} \alpha_j x'x^j = x' \underbrace{\sum_{j=1}^{N} \alpha_j x^j}_{w}$$

So, $w = \sum_{j=1}^{N} \alpha_j x^j$ $\Rightarrow$ all $\alpha_j = 0$ except for the support vectors!

- Use Kernels for "nonlinear" decision boundaries.

$\Rightarrow$ high-dim feature space: $x \to \phi(x)$

eg $\phi(x) = [x_1^2 \; x_2^2 \cdots x_2 x_4 \cdots x_{m-1} \; x_m \; 1]$

$\Rightarrow \hat{d}(x) = \text{sign}(\phi^T(x) w)$

Then, "Hinge loss with Ridge": $\min_w \left(\sum_{i=1}^{N} \left(1 - d^i \phi^T(x^i) w \right)_+ \right) + \lambda \|w\|_2^2$.

Claim. $w = \sum_{j=1}^{N} \phi(x^j) \alpha_j$. (*) (difference: $x^j \to \phi(x^j)$)

- "Kernel trick" replaces $\phi^T(x^i)\phi(x^i)$ with $K(x^i, x^i)$

From (★): $w = \sum_{j=1}^{N} \phi(x^j)\alpha_j$.

$\Rightarrow$ Hinge loss w/ Ridge: $\min_w \left(\sum_{i=1}^{N} \left(1 - d^i \phi^T(x^i) w\right)_+ \right) + \lambda \|w\|_2^2$.

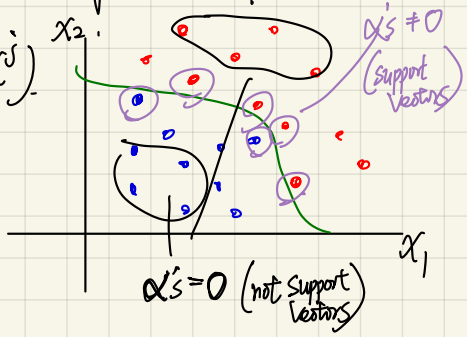
$$\Rightarrow \min_{\alpha} \left[\sum_{i=1}^{N} \left(1 - d^i \phi^T(x^i) \sum_{j=1}^{N} \alpha_j \phi(x^j)\right)_+ \right] + \lambda \sum_{i=1}^{N} \alpha_i \phi^T(x^i) \sum_{j=1}^{N} \alpha_j \phi(x^j)$$

$$= \min_{\alpha} \sum_{i=1}^{N} \left(1 - d^i \sum_{j=1}^{N} \alpha_j \underbrace{\phi^T(x^i)\phi(x^j)}_{K(x^i, x^j)}\right)_+ + \lambda \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j \underbrace{\phi^T(x^i)\phi(x^j)}_{K(x^i, x^j)}$$

$\Rightarrow$ SVM: $\min_{\alpha} \sum_{i=1}^{N} \left(1 - d^i \sum_{j=1}^{N} \alpha_j K(x^i, x^j)\right)_+ + \lambda \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j K(x^i, x^j)$

(★) It turns out, SVM has sparse α (in linear case + high-dim feature space)

- Decision boundary: $d(x) = 0 = \phi^T(x) w = \sum_{j=1}^{N} \alpha_j K(x, x^j)$
- Boundary (hinge loss) depends only on Support vectors
- Kernels ($K(u,v)$) measures similarity of vectors u, v



ex. Gaussian kernels: $K(u,v) = \exp\left(\frac{-\|u-v\|_2^2}{2\sigma^2}\right)$

$\rightarrow$ Solve α via Gradient Descent.