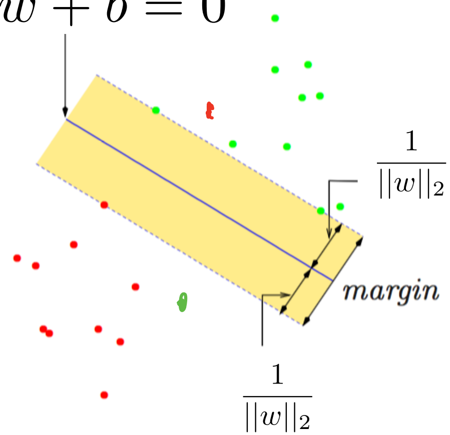
Kernels

What if the data is not linearly separable?

- $x^T w + b = 0$



Some points don't satisfy margin constraint:

$$\min_{w,b} ||w||_2^2$$

$$y_i(x_i^T w + b) \geq 1 \quad \forall i$$

Two options:

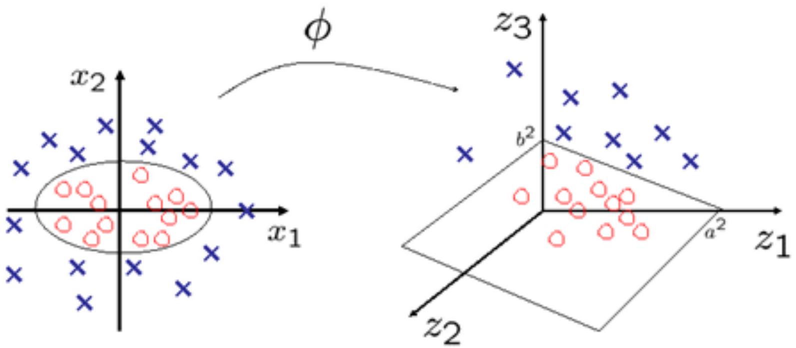
1. Introduce slack to this optimization problem
2. **Lift to higher dimensional space**

What if the data is not linearly separable?

Use features of features of features…

poly - feature

dim(ϕ) is exp in degree



How do we deal with high-dimensional lifts/data?

A fundamental trick in ML: use kernels

A function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a *kernel* for a map ϕ if $K(x, x') = \phi(x) \cdot \phi(x')$ for all x, x'.

1 2 $\langle x_i, x_j \rangle$

So, if we can represent our algorithms/decision rules as dot products
and we can find a kernel for our feature map
then we can avoid explicitly dealing with ϕ(x).

Examples of Kernels

- **Polynomials of degree exactly d**

$$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}\cdot\mathbf{v})^p$$

- **Polynomials of degree up to d**

$$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}\cdot\mathbf{v}+1)^p$$

- **Gaussian (squared exponential) kernel** (RBF)

$$K(\mathbf{u},\mathbf{v}) = \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||^2}{2\sigma^2}\right)$$

- **Sigmoid**

$$K(u,v) = \tanh(\gamma\cdot u^T v + r)$$

The Kernel Trick

$\{(x_j, y_j)\}_{j=1}^{n}$

1. **Pick a kernel K**

2. **For a linear predictor, show** $w = \sum_{i} \alpha_i x_i$ $\sum_{j=1}^{n}$, $\alpha \in \mathbb{R}^n$

Change loss function/decision rule to only access data through dot products $\langle x_i, x_j \rangle$

Given a new data $x_{new} \in \mathbb{R}^d$

$w^T x_{new} = \sum_{j=1}^{n} \alpha_j \langle x_j, x_{new} \rangle$

$\Rightarrow$ find α

$\alpha^T \begin{pmatrix} K(x_1, x_{new}) \\ \vdots \\ K(x_n, x_{new}) \end{pmatrix}$

Substitute $K(x_i, x_j)$ for $x_i^T x_j$

$\parallel$

Prediction: $\sum_{j=1}^{n} \alpha_j K(x_j, x_{new})$

The Kernel Trick for regularized least squares

$$\widehat{w} = \arg\min_{w} \sum_{i=1}^{n} (y_i - x_i^T w)^2 + \lambda ||w||_w^2$$

There exists an $\alpha \in \mathbb{R}^n$: $\widehat{w} = \sum_{i=1}^{n} \alpha_i x_i$

$$\begin{aligned}
\widehat{\alpha} &= \arg\min_{\alpha} \sum_{i=1}^{n} (y_i - \sum_{j=1}^{n} \alpha_j \langle x_j, x_i \rangle)^2 + \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \langle x_i, x_j \rangle \\
&= \arg\min_{\alpha} \sum_{i=1}^{n} (y_i - \sum_{j=1}^{n} \alpha_j K(x_i, x_j))^2 + \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j K(x_i, x_j) \\
&= \arg\min_{\alpha} ||\mathbf{y} - \mathbf{K}\alpha||_2^2 + \lambda \alpha^T \mathbf{K} \alpha
\end{aligned}$$

$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$$

Why regularization?

$\forall\ v \neq 0,\ v^T K v > 0$

generally, dim of $\phi(x)$ dimension $> n$

Typically, $\mathbf{K} \succ 0$. What if $\lambda = 0$?

$$\widehat{\alpha} = \arg\min_{\alpha} ||\mathbf{y} - \mathbf{K}\alpha||_2^2 + \lambda\alpha^T\mathbf{K}\alpha$$

Why regularization?

X: $n \times d$ data

y: n labels

$$\hat{y} = \begin{pmatrix} K(x_1, x_1) & \cdots & K(x_1, x_n) \\ & \ddots & \\ & \cdots & K(x_n, x_n) \end{pmatrix} \hat{\alpha}$$

Typically, $\mathbf{K} \succ 0$. What if $\lambda = 0$?

$$\widehat{\alpha} = \arg\min_{\alpha} ||\mathbf{y} - \mathbf{K}\alpha||_2^2 + \cancel{\lambda\alpha^T\mathbf{K}\alpha}$$

$$\forall x, x' \quad K(x, x') = K(x', x)$$

Unregularized kernel least squares can (over) fit **any data**!

$$\widehat{\alpha} = \mathbf{K}^{-1}\mathbf{y}$$

$\hat{y}$: predictions on y

$\in \mathbb{R}^n$

$$\hat{y}_i = \begin{pmatrix} K(x_1, x_i) \\ \vdots \\ K(x_n, x_i) \end{pmatrix}^T \hat{\alpha}$$

$$\hat{y} = K\hat{\alpha} = KK^{-1}y = y$$

The Kernel Trick for SVMs

$$w = \sum_{j=1}^{n} \alpha_j x_j$$

$$\min_{w,b} \frac{1}{n} \sum_{i=1}^{n} \max\{0, 1 - y_i(x_i^T w + b)\} + \lambda \|w\|_2^2$$

$$\Leftrightarrow \min_{\alpha, b} \frac{1}{n} \sum_{i=1}^{n} \max\{0, 1 - y_i(\sum_{j=1}^{n} \alpha_j \langle x_i, x_j \rangle + b)\} + \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \langle x_i, x_j \rangle$$

replace $\langle x_i, x_j \rangle \longrightarrow K(x_i, x_j)$

$$\Rightarrow \min_{\alpha, b} \frac{1}{n} \sum_{i=1}^{n} \max\{0, 1 - y_i(\sum_{j=1}^{n} \alpha_j K(x_i, x_j) + b)\} + \lambda \alpha^T K \alpha$$

convex in α, b

$\hat{\alpha}, \hat{b}$ are learned

Decision rule: x_{new}

$$\sum_{j=1}^{n} \hat{\alpha}_j K(x_j, x_{new}) + \hat{b}$$

$> 0 \Rightarrow$ predict 1

$< 0 \Rightarrow$ predict 0

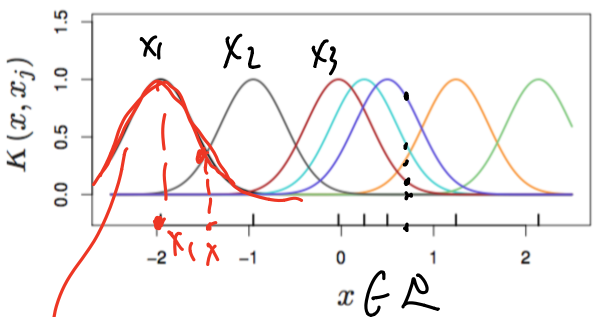
RBF Kernel

$$K(\overset{x_1}{\mathbf{u}},\overset{x}{\mathbf{v}}) = \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||_2^2}{2\sigma^2}\right)$$

$x, x_1 \in \mathbb{R}$

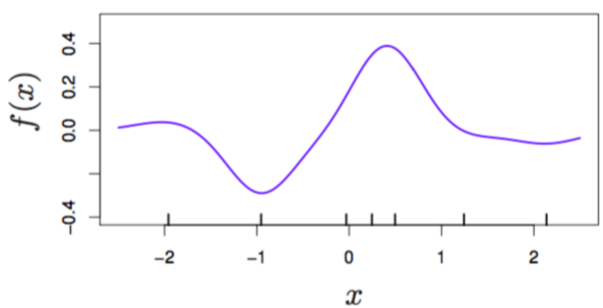
This is like weighting “bumps” on each point

Radial Basis Functions



$K(x_1, x)$

$f(x) = \overset{b}{\cancel{\alpha_0}} + \sum_j \alpha_j K(x, x_j)$



RBF Kernel

$$K(\mathbf{u},\mathbf{v}) = \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||_2^2}{2\sigma^2}\right)$$

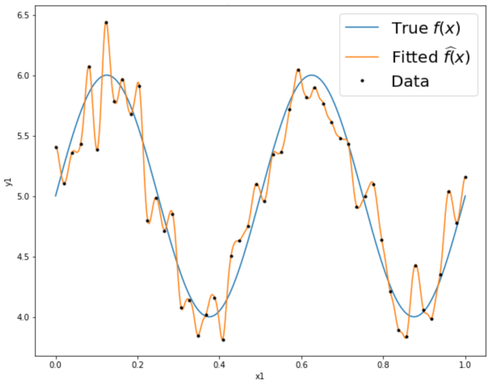
$\sigma \to 0$

$\exp(-\infty) = 0$

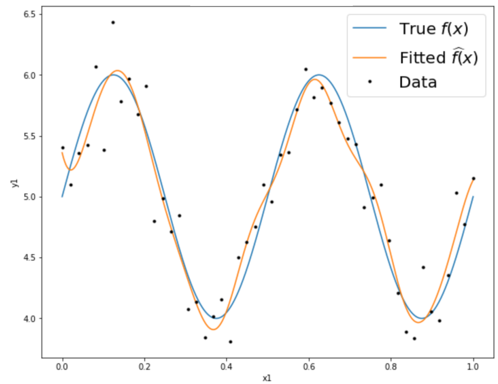
$\exp(0) = 1$

The bandwidth sigma has an enormous effect on fit:

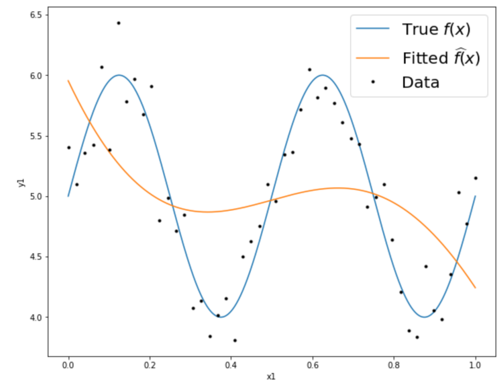
$\sigma = 10^{-2} \quad \lambda = 10^{-4}$



$\sigma = 10^{-1} \quad \lambda = 10^{-4}$



$\sigma = 10^{-0} \quad \lambda = 10^{-4}$



bias-variance: $\uparrow \sigma$: larger bias $\downarrow$ variance

as $\sigma \to 0$, only $x = x_i$, $K(x_i, x) \neq 0$, $x \neq x_i$, $K(x_i, x) = 0$

$$\widehat{f}(x) = \sum_{i=1}^{n} \widehat{\alpha}_i K(x_i, x)$$

$\sigma \to \infty$, $K(x_i, x) = 1$, $\forall i$

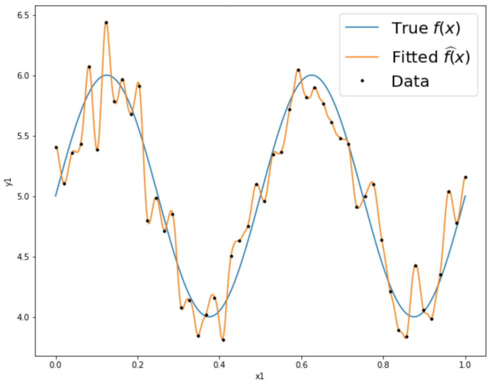
average over all y

RBF Kernel

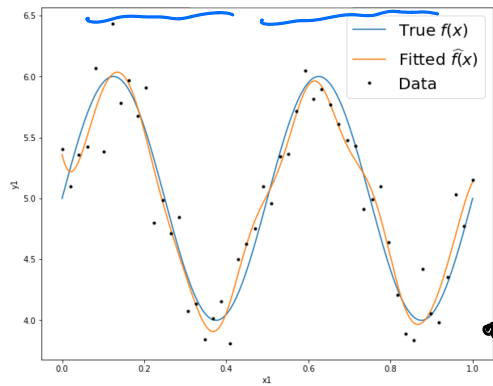
$$K(\mathbf{u},\mathbf{v}) = \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||_2^2}{2\sigma^2}\right)$$

The bandwidth sigma has an enormous effect on fit:

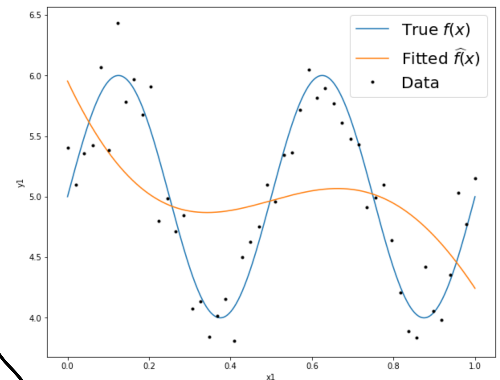
$\sigma = 10^{-2} \quad \lambda = 10^{-4}$



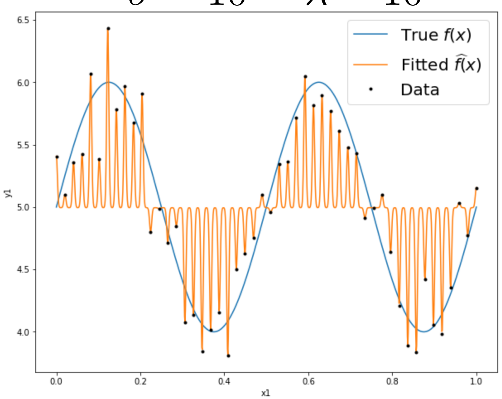
$\sigma = 10^{-1} \quad \lambda = 10^{-4}$



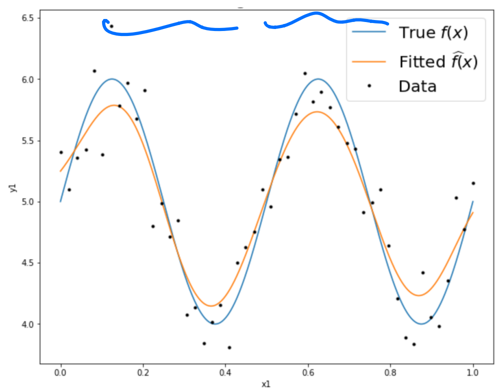
$\sigma = 10^{-0} \quad \lambda = 10^{-4}$



$\sigma = 10^{-3} \quad \lambda = 10^{-4}$



$\sigma = 10^{-1} \quad \lambda = 10^{-0}$

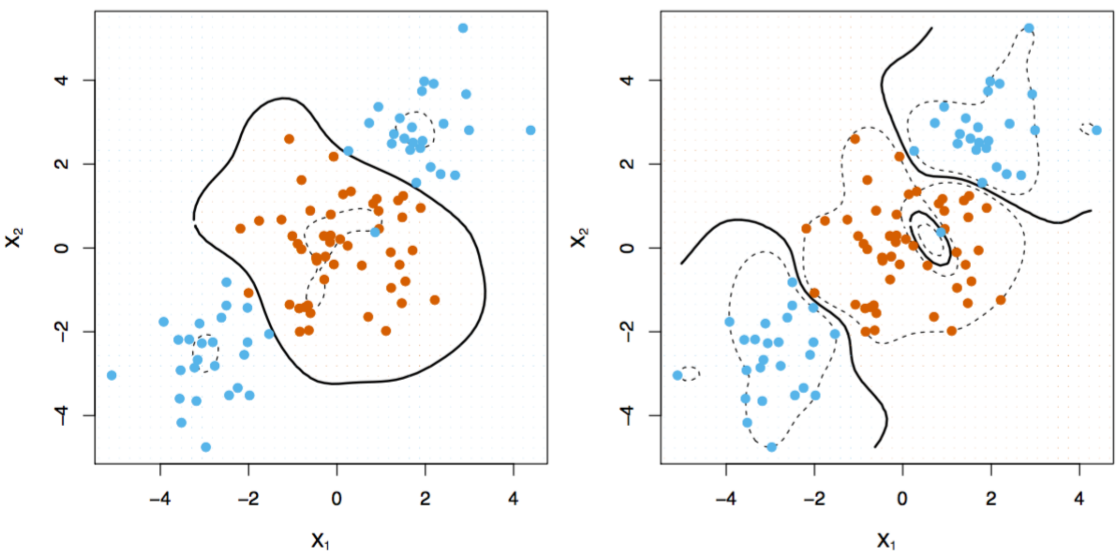


$$\widehat{f}(x) = \sum_{i=1}^{n} \widehat{\alpha}_i K(x_i, x)$$

RBF kernel and random features

$$\widehat{w} = \sum_{i=1}^{n} \max\{0, 1 - y_i(b + x_i^T w)\} + \lambda ||w||_2^2$$

$$\min_{\alpha, b} \sum_{i=1}^{n} \max\{0, 1 - y_i(b + \sum_{j=1}^{n} \alpha_j K(x_i, x_j))\} + \lambda \sum_{i,j=1}^{n} \alpha_i \alpha_j K(x_i, x_j)$$



dc score
6

RBF kernel and random features

$$K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{||\mathbf{u} - \mathbf{v}||_2^2}{2\sigma^2}\right)$$

If n is very large, allocating an n-by-n matrix is tough.

$$\alpha = (K + \lambda I)^{-1} y, \quad K \in \mathbb{R}^{n \times n}$$

n: 1m, 1b

not feasible

RBF: ϕ_∞, $\phi \approx \phi_\infty$, $\phi(x)^T\phi(x') \approx \phi_\infty(x)^T\phi(x') = K(x,x')$

RBF kernel and random features

n: 1b,

p: 10K

100K $\to$ $K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||_2^2}{2\sigma^2}\right)$, $p \ll n$

If n is very large, allocating an n-by-n matrix is tough.

use sampling from inf-dim feature space

$2\cos(\alpha)\cos(\beta) = \cos(\alpha+\beta) + \cos(\alpha-\beta)$

$e^{jz} = \cos(z) + j\sin(z)$

$\Rightarrow$ apply linear regression / SVM on ϕ

$$\phi(x) = \begin{bmatrix} \sqrt{2}\cos(w_1^T x + b_1) \\ \vdots \\ \sqrt{2}\cos(w_p^T x + b_p) \end{bmatrix} \in \mathbb{R}^p \qquad \begin{aligned} w_k &\sim \mathcal{N}(0, 2\gamma I) \\ b_k &\sim \text{uniform}(0, \pi) \end{aligned}$$

$$\mathbb{E}_{w,b}\left[\frac{1}{p}\phi(x)^T\phi(x')\right] = \frac{1}{p}\sum_{k=1}^{p}\mathbb{E}_{w,b}\left[2\cos(w_k^T x + b_k)\cos(w_k^T x' + b_k)\right]$$

$$= \mathbb{E}\left[\cos(w^T(x-x')) + \cos(w^T(x+x') + 2b)\right]$$

$$= e^{-\gamma\|x-y\|_2^2}$$

$x \in \mathbb{R}^d$

$w_k \in \mathbb{R}^d$, $\gamma = \frac{1}{2\sigma^2}$

[Rahimi, Recht NIPS 2007]

“NIPS Test of Time Award, 2018”

$x, \ \phi(x) \in \mathbb{R}^{\infty}$

$\Downarrow$

$K(x, x')$, $O(n^3)$

$\Downarrow$

sampled feame

$\tilde{\phi} \in \mathbb{R}^{p}$

$p << n$

Apply SVM on $\tilde{\phi}$

String Kernels

Example from Efron and Hastie, 2016

Amino acid sequences of different lengths:

x1

IPTSALVKETLALLSTHRTLLIANETLRIPVPVHKNHQLCTEEIFQGIGTLESQTVQGGTV
ERLFKNLSLIKKYIDGQKKKCGEERRRVNQFLDY**LQE**FLGVMNTEWI

x2

PHRRDLCSRSIWLARKIRSDLTALTESYVKHQGLWSELTEAER**LQE**NLQAYRTFHVLLA
RLLEDQQVHFTPTEGDFHQAIHTLLLQVAAFAYQIEELMILLEYKIPRNEADGMLFEKK
LWGLKV**LQE**LSQWTVRSIHDLRFISSHQTGIP

All subsequences of length 3 (of possible 20 amino acids) $20^3 = 8{,}000$

$$h^3_{\text{LQE}}(x_1) = 1 \text{ and } h^3_{\text{LQE}}(x_2) = 2.$$

Graph, time-series, etc - - -

Fixed Feature V.S. Learned Feature

linear function with fixed feature

$x \Rightarrow \phi(x)$

ϕ : hand-crafted , $\phi(x)^T w$

can we learn ϕ from data

$\Rightarrow$ neural net