

Announcements





Kernels

Machine Learning – CSE546

Kevin Jamieson

University of Washington

November 6, 2018

Machine Learning Problems

- Have a bunch of iid data of the form:

$$\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d \quad y_i \in \mathbb{R}$$

- Learning a model's parameters:

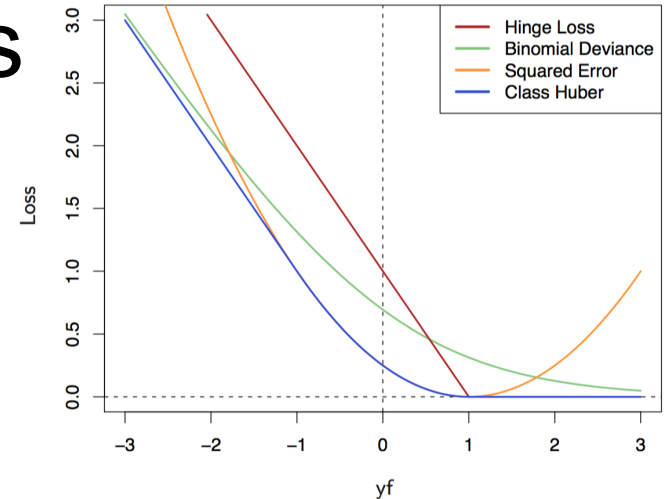
Each $\ell_i(w)$ is convex.

$$\sum_{i=1}^n \ell_i(w)$$

Hinge Loss: $\ell_i(w) = \max\{0, 1 - y_i x_i^T w\}$

Logistic Loss: $\ell_i(w) = \log(1 + \exp(-y_i x_i^T w))$

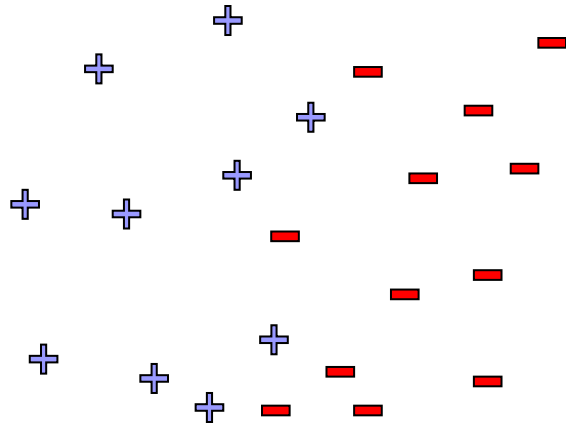
Squared error Loss: $\ell_i(w) = (y_i - x_i^T w)^2$



All in terms of inner products! Even nearest neighbor can use inner products!

What if the data is not linearly separable?

Use features of features
of features of features....



$$\phi(x) : \mathbb{R}^d \rightarrow \mathbb{R}^p$$

Feature space can get really large really quickly!

Dot-product of polynomials

$\Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) =$ polynomials of degree exactly d

$$d = 1 : \phi(u) = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad \langle \phi(u), \phi(v) \rangle = u_1 v_1 + u_2 v_2$$

Dot-product of polynomials

$\Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) =$ polynomials of degree exactly d

$$d = 1 : \phi(u) = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad \langle \phi(u), \phi(v) \rangle = u_1 v_1 + u_2 v_2$$

$$d = 2 : \phi(u) = \begin{bmatrix} u_1^2 \\ u_2^2 \\ u_1 u_2 \\ u_2 u_1 \end{bmatrix} \quad \langle \phi(u), \phi(v) \rangle = u_1^2 v_1^2 + u_2^2 v_2^2 + 2u_1 u_2 v_1 v_2$$

Dot-product of polynomials

$\Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) =$ polynomials of degree exactly d

$$d = 1 : \phi(u) = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad \langle \phi(u), \phi(v) \rangle = u_1 v_1 + u_2 v_2$$

$$d = 2 : \phi(u) = \begin{bmatrix} u_1^2 \\ u_2^2 \\ u_1 u_2 \\ u_2 u_1 \end{bmatrix} \quad \langle \phi(u), \phi(v) \rangle = u_1^2 v_1^2 + u_2^2 v_2^2 + 2u_1 u_2 v_1 v_2$$

General d :

Dimension of $\phi(u)$ is roughly p^d if $u \in \mathbb{R}^p$

Kernel Trick

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

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

Kernel Trick

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

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

$$\hat{\alpha} = \arg \min_{\alpha} \sum_{i=1}^n (y_i - \sum_{j=1}^n \alpha_j \langle x_j, x_i \rangle)^2 + \lambda \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j \langle x_i, x_j \rangle$$

Kernel Trick

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

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

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

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

Why regularization?

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

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

Why regularization?

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

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

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

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

Common kernels

- Polynomials of degree exactly d

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

- Polynomials of degree up to d

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

- Gaussian (squared exponential) kernel

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

- Sigmoid

$$K(\mathbf{u}, \mathbf{v}) = \tanh(\eta \mathbf{u} \cdot \mathbf{v} + \nu)$$

Mercer's Theorem

- When do we have a valid Kernel $K(x, x')$?

- Sufficient:

$K(x, x')$ is a valid kernel if there exists $\phi(x)$ such that $K(x, x') = \phi(x)^T \phi(x')$

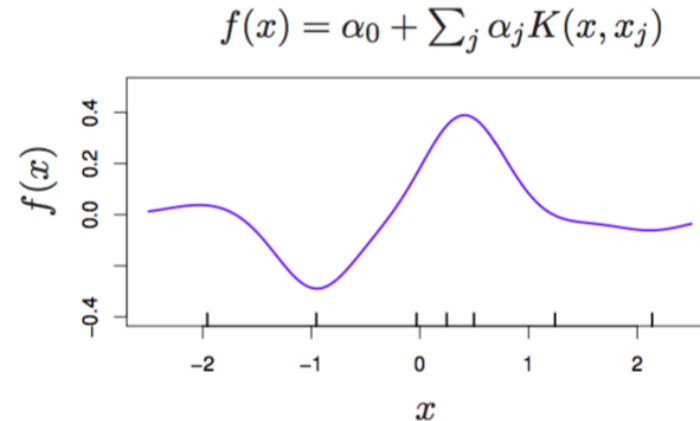
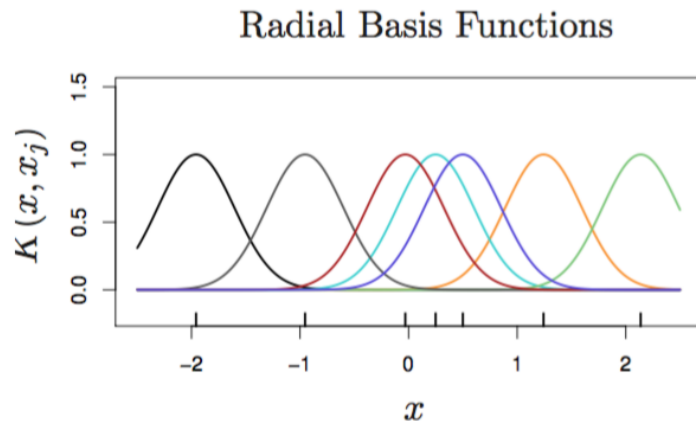
- Mercer's Theorem:

$K(x, x')$ is a valid kernel if and only if \mathbf{K} is symmetric and positive semi-definite for any pointset (x_1, \dots, x_n) where $\mathbf{K}_{i,j} = K(x_i, x_j)$.

RBF Kernel

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

- Note that this is like weighting “bumps” on each point like kernel smoothing but now we **learn** the weights

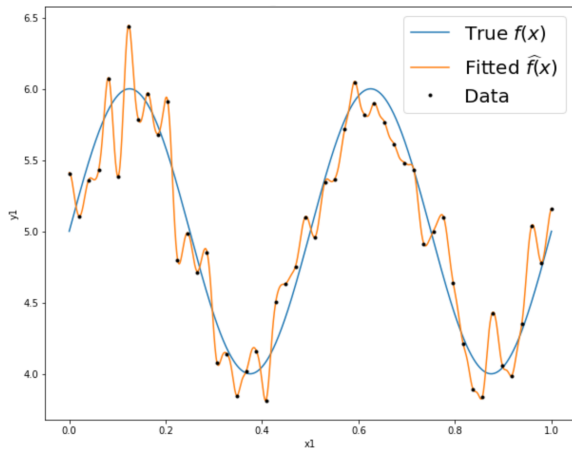


RBF Kernel

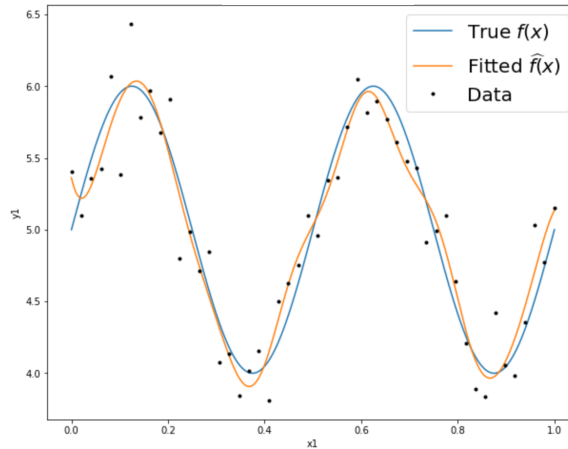
$$K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{\|\mathbf{u} - \mathbf{v}\|_2^2}{2\sigma^2}\right)$$

The bandwidth sigma has an enormous effect on fit:

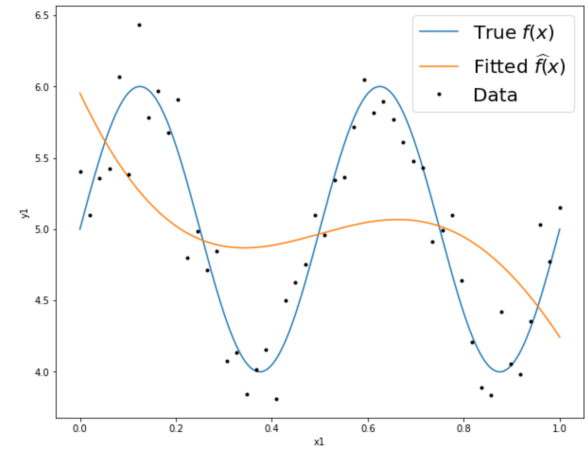
$$\sigma = 10^{-2} \quad \lambda = 10^{-4}$$



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



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



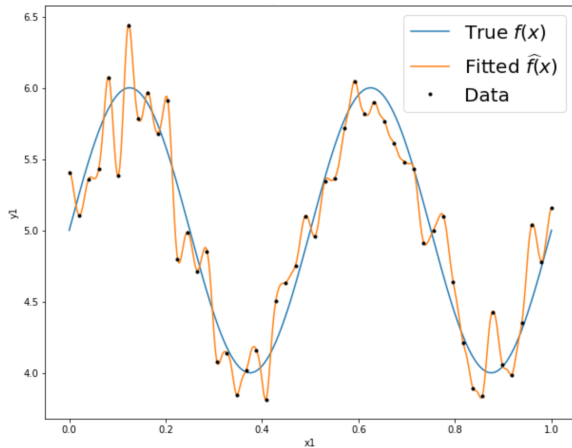
$$\hat{f}(x) = \sum_{i=1}^n \hat{\alpha}_i K(x_i, x)$$

RBF Kernel

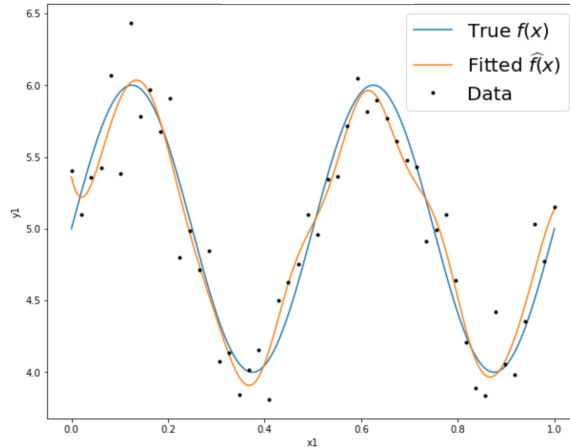
$$K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{\|\mathbf{u} - \mathbf{v}\|_2^2}{2\sigma^2}\right)$$

The bandwidth sigma has an enormous effect on fit:

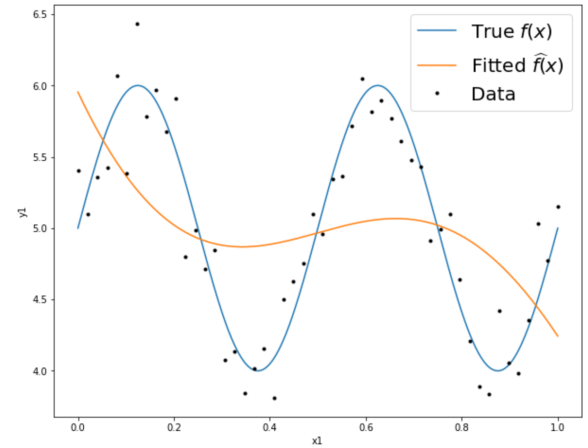
$$\sigma = 10^{-2} \quad \lambda = 10^{-4}$$



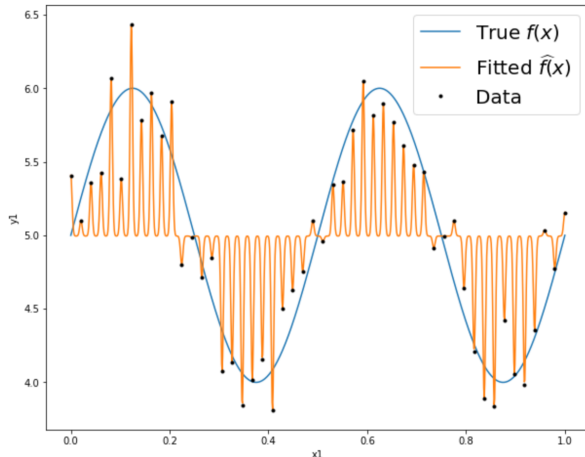
$$\sigma = 10^{-1} \quad \lambda = 10^{-4}$$



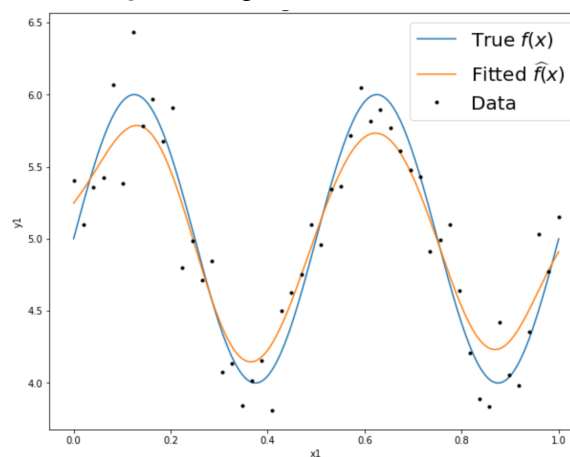
$$\sigma = 10^{-0} \quad \lambda = 10^{-4}$$



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



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



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

RBF Kernel

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

Basis representation in 1d?

$$[\phi(x)]_i = \frac{1}{\sqrt{i!}} e^{-\frac{x^2}{2}} x^i \quad \text{for } i = 0, 1, \dots$$

$$\begin{aligned}\phi(x)^T \phi(x') &= \sum_{i=0}^{\infty} \left(\frac{1}{\sqrt{i!}} e^{-\frac{x^2}{2}} x^i \right) \left(\frac{1}{\sqrt{i!}} e^{-\frac{(x')^2}{2}} (x')^i \right) \\ &= e^{-\frac{x^2 + (x')^2}{2}} \sum_{i=0}^{\infty} \frac{1}{i!} (xx')^i \\ &= e^{-|x-x'|^2/2}\end{aligned}$$

If n is very large, allocating an n -by- n matrix is tough. Can we truncate the above sum to approximate the kernel?

RBF kernel and random features

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

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

Recall HW1 where we used the feature map:

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

$$\begin{aligned} \mathbb{E}\left[\frac{1}{p} \phi(x)^T \phi(y)\right] &= \frac{1}{p} \sum_{k=1}^p \mathbb{E}[2 \cos(w_k^T x + b_k) \cos(w_k^T y + b_k)] \\ &= \mathbb{E}_{w,b}[2 \cos(w^T x + b) \cos(w^T y + b)] \end{aligned}$$

RBF kernel and random features

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

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

Recall HW1 where we used the feature map:

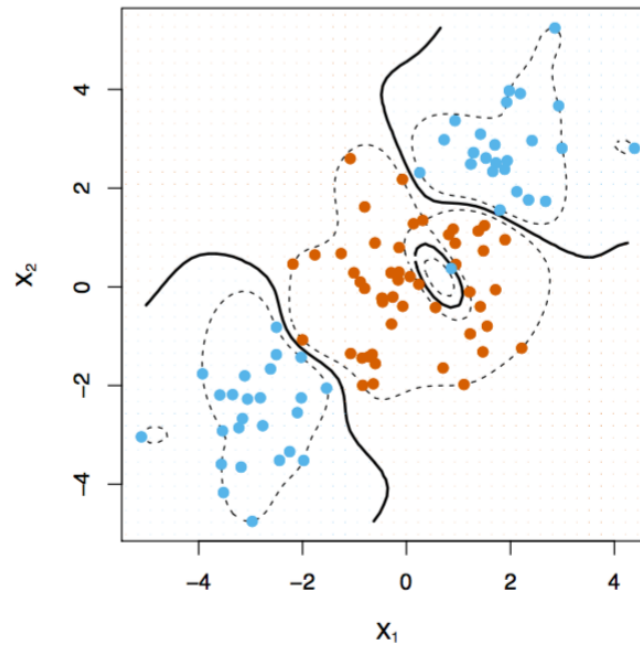
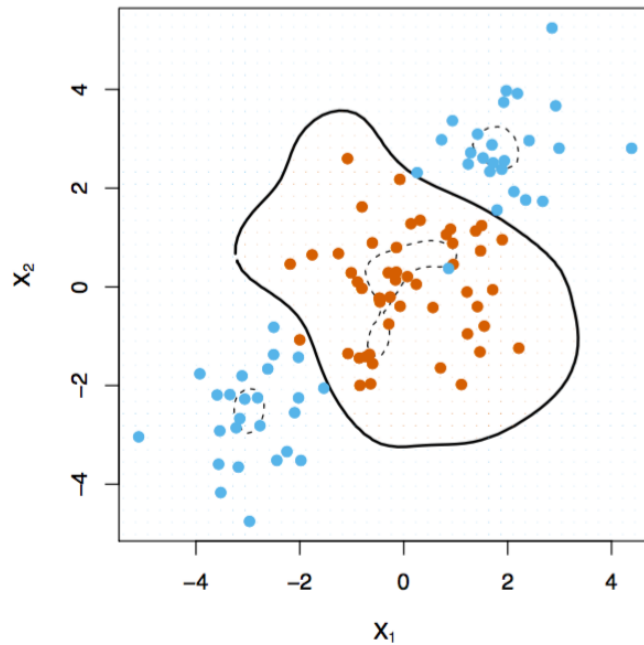
$$\phi(x) = \begin{bmatrix} \sqrt{2} \cos(w_1^T x + b_1) \\ \vdots \\ \sqrt{2} \cos(w_p^T x + b_p) \end{bmatrix} \quad \begin{aligned} w_k &\sim \mathcal{N}(0, 2\gamma I) \\ b_k &\sim \text{uniform}(0, \pi) \end{aligned}$$

$$\begin{aligned} \mathbb{E}\left[\frac{1}{p} \phi(x)^T \phi(y)\right] &= \frac{1}{p} \sum_{k=1}^p \mathbb{E}[2 \cos(w_k^T x + b_k) \cos(w_k^T y + b_k)] \\ &= \mathbb{E}_{w,b}[2 \cos(w^T x + b) \cos(w^T y + b)] \\ &= e^{-\gamma \|x-y\|_2^2} \end{aligned}$$

[Rahimi, Recht NIPS 2007]
“NIPS Test of Time Award, 2018”

RBF Classification

$$\hat{w} = \sum_{i=1}^n \max\{0, 1 - y_i(b + x_i^T w)\} + \lambda \|w\|_2^2$$
$$\min_{\alpha, b} \sum_{i=1}^n \max\{0, 1 - y_i(b + \sum_{j=1}^n \alpha_j \langle x_i, x_j \rangle)\} + \lambda \sum_{i,j=1}^n \alpha_i \alpha_j \langle x_i, x_j \rangle$$



Wait, infinite dimensions?

- Isn't everything separable there? How are we not overfitting?
- Regularization! Fat shattering $(R/\text{margin})^2$

String Kernels

Example from Efron and Hastie, 2016

Amino acid sequences of different lengths:

x1 IPTSALVKETLALLSTHRTLLIANETLRIPVPVHKNHQLCTEEIFQGIGTLESQTVQGGTV
ERLFKNLSLIKKYIDGQKKKCGEERRRVNQFLDY**LQE**FLGVMNTEWI

x2 PHRRDLCSRSIWLARKIRSDLTALTESYVKHQGLWSELTEAER**LQEN**LQAYRTFHVLLA
RLLEDQQVHFTPTEGDFHQAIHTLLLQVAAFAYQIEELMILLEYKIPRNEADGMLFEKK
LWGLKV**LQE**LSQWTVRSIHDLRFISSHQTGIP

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

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



Principal Component Analysis

Machine Learning – CSE546

Kevin Jamieson

University of Washington

November 6, 2018

Linear projections

Given $x_1, \dots, x_n \in \mathbb{R}^d$, for $q \ll d$ find a compressed representation with $\lambda_1, \dots, \lambda_n \in \mathbb{R}^q$ such that $x_i \approx \mu + \mathbf{V}_q \lambda_i$ and $\mathbf{V}_q^T \mathbf{V}_q = I$

$$\min_{\mu, \mathbf{V}_q, \{\lambda_i\}_i} \sum_{i=1}^n \|x_i - \mu - \mathbf{V}_q \lambda_i\|_2^2$$

Linear projections

Given $x_1, \dots, x_n \in \mathbb{R}^d$, for $q \ll d$ find a compressed representation with $\lambda_1, \dots, \lambda_n \in \mathbb{R}^q$ such that $x_i \approx \mu + \mathbf{V}_q \lambda_i$ and $\mathbf{V}_q^T \mathbf{V}_q = I$

$$\min_{\mu, \mathbf{V}_q, \{\lambda_i\}_i} \sum_{i=1}^n \|x_i - \mu - \mathbf{V}_q \lambda_i\|_2^2$$

Fix \mathbf{V}_q and solve for μ, λ_i :

$$\mu = \bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$
$$\lambda_i = \mathbf{V}_q^T (x_i - \bar{x})$$

Which gives us:

$$\min_{\mathbf{V}_q} \sum_{i=1}^N \|(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})\|^2.$$

$\mathbf{V}_q \mathbf{V}_q^T$ is a *projection matrix* that minimizes error in basis of size q

Linear projections

$$\sum_{i=1}^N \|(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})\|_2^2$$

$$\Sigma := \sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T$$
$$\mathbf{V}_q^T \mathbf{V}_q = I_q$$

Linear projections

$$\sum_{i=1}^N \|(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})\|_2^2$$

$$\Sigma := \sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T$$
$$\mathbf{V}_q^T \mathbf{V}_q = I_q$$

$$\min_{\mathbf{V}_q} \sum_{i=1}^N \|(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})\|_2^2 = \min_{\mathbf{V}_q} \text{Tr}(\Sigma) - \text{Tr}(\mathbf{V}_q^T \Sigma \mathbf{V}_q)$$

Eigenvalue decomposition of $\Sigma =$

Linear projections

$$\sum_{i=1}^N \|(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})\|_2^2$$

$$\Sigma := \sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T$$
$$\mathbf{V}_q^T \mathbf{V}_q = I_q$$

$$\min_{\mathbf{V}_q} \sum_{i=1}^N \|(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})\|_2^2 = \min_{\mathbf{V}_q} \text{Tr}(\Sigma) - \text{Tr}(\mathbf{V}_q^T \Sigma \mathbf{V}_q)$$

Eigenvalue decomposition of $\Sigma =$

\mathbf{V}_q are the first q eigenvectors of Σ

Minimize reconstruction error and capture the most variance in your data.

Pictures

\mathbf{V}_q are the first q eigenvectors of Σ

\mathbf{V}_q are the first q *principal components*

$$\Sigma := \sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T$$

Linear projections

Given $x_i \in \mathbb{R}^d$ and some $q < d$ consider

$$\min_{\mathbf{V}_q} \sum_{i=1}^N \|(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})\|^2.$$

where $\mathbf{V}_q = [v_1, v_2, \dots, v_q]$ is orthonormal:

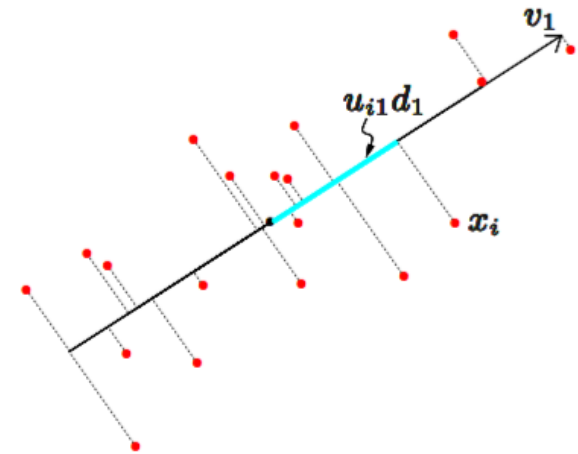
$$\mathbf{V}_q^T \mathbf{V}_q = I_q$$

\mathbf{V}_q are the first q eigenvectors of Σ

\mathbf{V}_q are the first q principal components

Principal Component Analysis (PCA) projects $(\mathbf{X} - \mathbf{1}\bar{x}^T)$ down onto \mathbf{V}_q

$$(\mathbf{X} - \mathbf{1}\bar{x}^T) \mathbf{V}_q = \mathbf{U}_q \text{diag}(d_1, \dots, d_q) \quad \mathbf{U}_q^T \mathbf{U}_q = I_q$$



$$\Sigma := \sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T$$

Singular Value Decomposition (SVD)

Theorem (SVD): Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ with rank $r \leq \min\{m, n\}$. Then $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ where $\mathbf{S} \in \mathbb{R}^{r \times r}$ is diagonal with positive entries, $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, $\mathbf{V}^T\mathbf{V} = \mathbf{I}$.

$$\mathbf{A}^T \mathbf{A} v_i =$$

$$\mathbf{A}\mathbf{A}^T u_i =$$

Singular Value Decomposition (SVD)

Theorem (SVD): Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ with rank $r \leq \min\{m, n\}$. Then $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ where $\mathbf{S} \in \mathbb{R}^{r \times r}$ is diagonal with positive entries, $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, $\mathbf{V}^T\mathbf{V} = \mathbf{I}$.

$$\mathbf{A}^T \mathbf{A} v_i = \mathbf{S}_{i,i}^2 v_i$$

$$\mathbf{A}\mathbf{A}^T u_i = \mathbf{S}_{i,i}^2 u_i$$

\mathbf{V} are the first r eigenvectors of $\mathbf{A}^T \mathbf{A}$ with eigenvalues $\text{diag}(\mathbf{S})$
 \mathbf{U} are the first r eigenvectors of $\mathbf{A}\mathbf{A}^T$ with eigenvalues $\text{diag}(\mathbf{S})$

Linear projections

Given $x_i \in \mathbb{R}^d$ and some $q < d$ consider

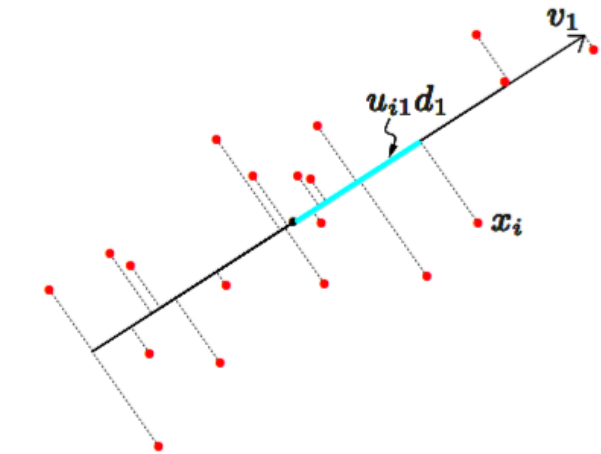
$$\min_{\mathbf{V}_q} \sum_{i=1}^N \|(x_i - \bar{x}) - \mathbf{V}_q \mathbf{V}_q^T (x_i - \bar{x})\|^2.$$

where $\mathbf{V}_q = [v_1, v_2, \dots, v_q]$ is orthonormal:

$$\mathbf{V}_q^T \mathbf{V}_q = I_q$$

\mathbf{V}_q are the first q eigenvectors of Σ

\mathbf{V}_q are the first q principal components



$$\Sigma := \sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T$$

Principal Component Analysis (PCA) projects $(\mathbf{X} - \mathbf{1}\bar{x}^T)$ down onto \mathbf{V}_q

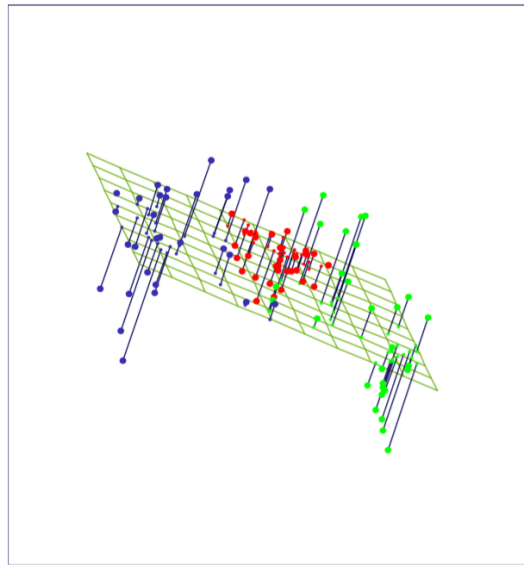
$$(\mathbf{X} - \mathbf{1}\bar{x}^T)\mathbf{V}_q = \mathbf{U}_q \text{diag}(d_1, \dots, d_q) \quad \mathbf{U}_q^T \mathbf{U}_q = I_q$$

Singular Value Decomposition defined as

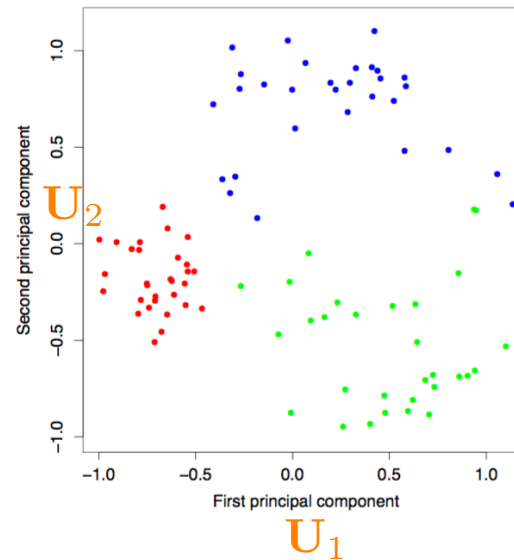
$$\mathbf{X} - \mathbf{1}\bar{x}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

Dimensionality reduction

V_q are the first q eigenvectors of Σ and SVD $\mathbf{X} - \mathbf{1}\bar{x}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T$



$$\mathbf{X} - \mathbf{1}\bar{x}^T$$



Dimensionality reduction

\mathbf{V}_q are the first q eigenvectors of Σ and SVD $\mathbf{X} - \mathbf{1}\bar{x}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T$

Handwritten 3's, 16x16 pixel image so that $x_i \in \mathbb{R}^{256}$

$$\begin{aligned}\hat{f}(\lambda) &= \bar{x} + \lambda_1 v_1 + \lambda_2 v_2 \\ &= \text{3} + \lambda_1 \cdot \text{3} + \lambda_2 \cdot \text{3}.\end{aligned}$$

$$(\mathbf{X} - \mathbf{1}\bar{x}^T)\mathbf{V}_2 = \mathbf{U}_2\mathbf{S}_2 \in \mathbb{R}^{n \times 2}$$

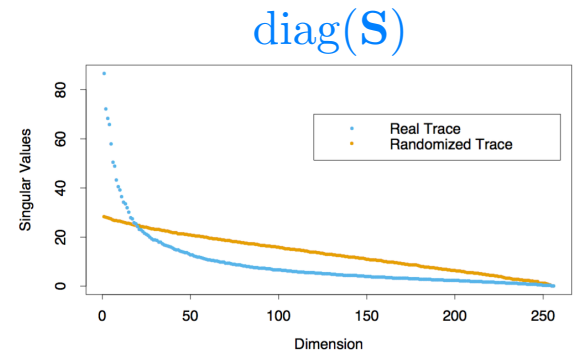
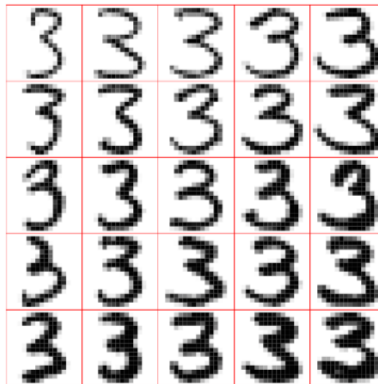
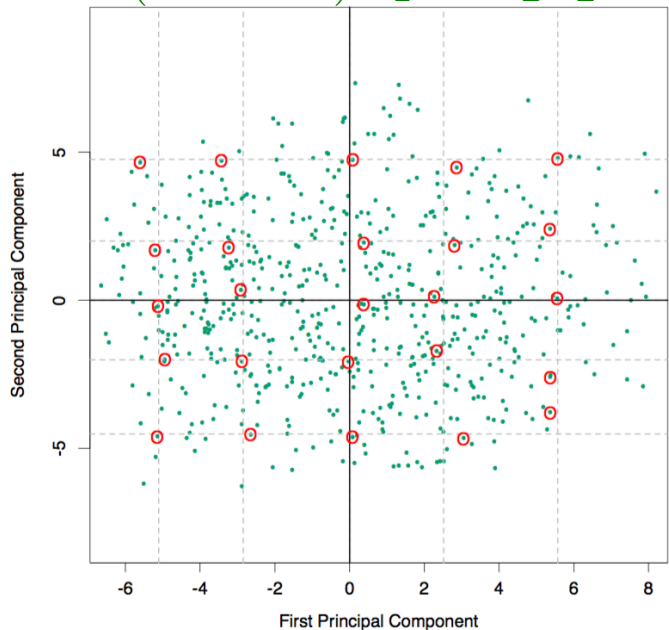


FIGURE 14.24. The 256 singular values for the digitized threes, compared to those for a randomized version of the data (each column of \mathbf{X} was scrambled).

Kernel PCA

\mathbf{V}_q are the first q eigenvectors of Σ and SVD $\mathbf{X} - \mathbf{1}\bar{x}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T$

$$(\mathbf{X} - \mathbf{1}\bar{x}^T)\mathbf{V}_q = \mathbf{U}_q\mathbf{S}_q \in \mathbb{R}^{n \times q}$$

$$\mathbf{J}\mathbf{X} = \mathbf{X} - \mathbf{1}\bar{x}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T \quad \mathbf{J} = \mathbf{I} - \mathbf{1}\mathbf{1}^T/n$$

$$(\mathbf{J}\mathbf{X})(\mathbf{J}\mathbf{X})^T =$$

Kernel PCA

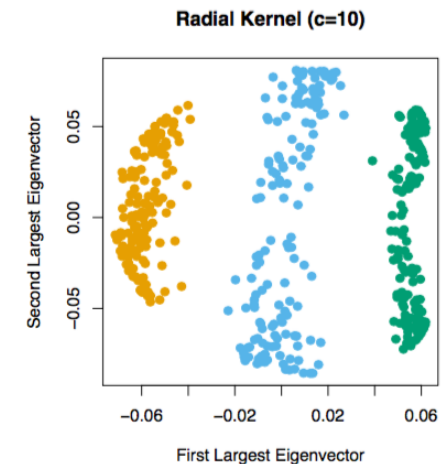
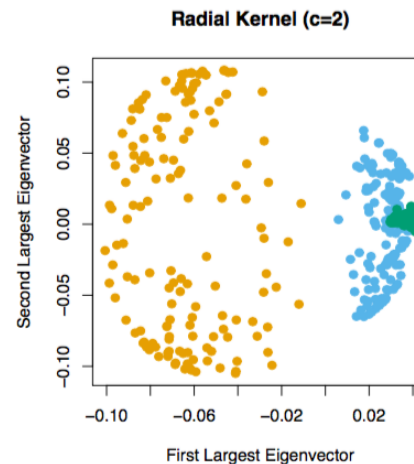
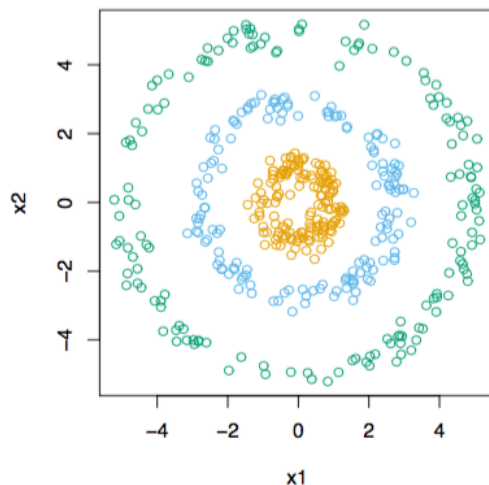
\mathbf{V}_q are the first q eigenvectors of Σ and SVD $\mathbf{X} - \mathbf{1}\bar{x}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T$

$$(\mathbf{X} - \mathbf{1}\bar{x}^T)\mathbf{V}_q = \mathbf{U}_q\mathbf{S}_q \in \mathbb{R}^{n \times q}$$

$$\mathbf{J}\mathbf{X} = \mathbf{X} - \mathbf{1}\bar{x}^T = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

$$\mathbf{J} = \mathbf{I} - \mathbf{1}\mathbf{1}^T/n$$

$$(\mathbf{J}\mathbf{X})(\mathbf{J}\mathbf{X})^T = \mathbf{U}\mathbf{S}^2\mathbf{U}^T$$



PCA Algorithm

PCA

input

A matrix of m examples $X \in \mathbb{R}^{m,d}$

number of components n

if ($m > d$)

$$A = X^T X$$

Let $\mathbf{u}_1, \dots, \mathbf{u}_n$ be the eigenvectors of A with largest eigenvalues

else

$$B = X X^T$$

Let $\mathbf{v}_1, \dots, \mathbf{v}_n$ be the eigenvectors of B with largest eigenvalues

for $i = 1, \dots, n$ set $\mathbf{u}_i = \frac{1}{\|X^T \mathbf{v}_i\|} X^T \mathbf{v}_i$

output: $\mathbf{u}_1, \dots, \mathbf{u}_n$