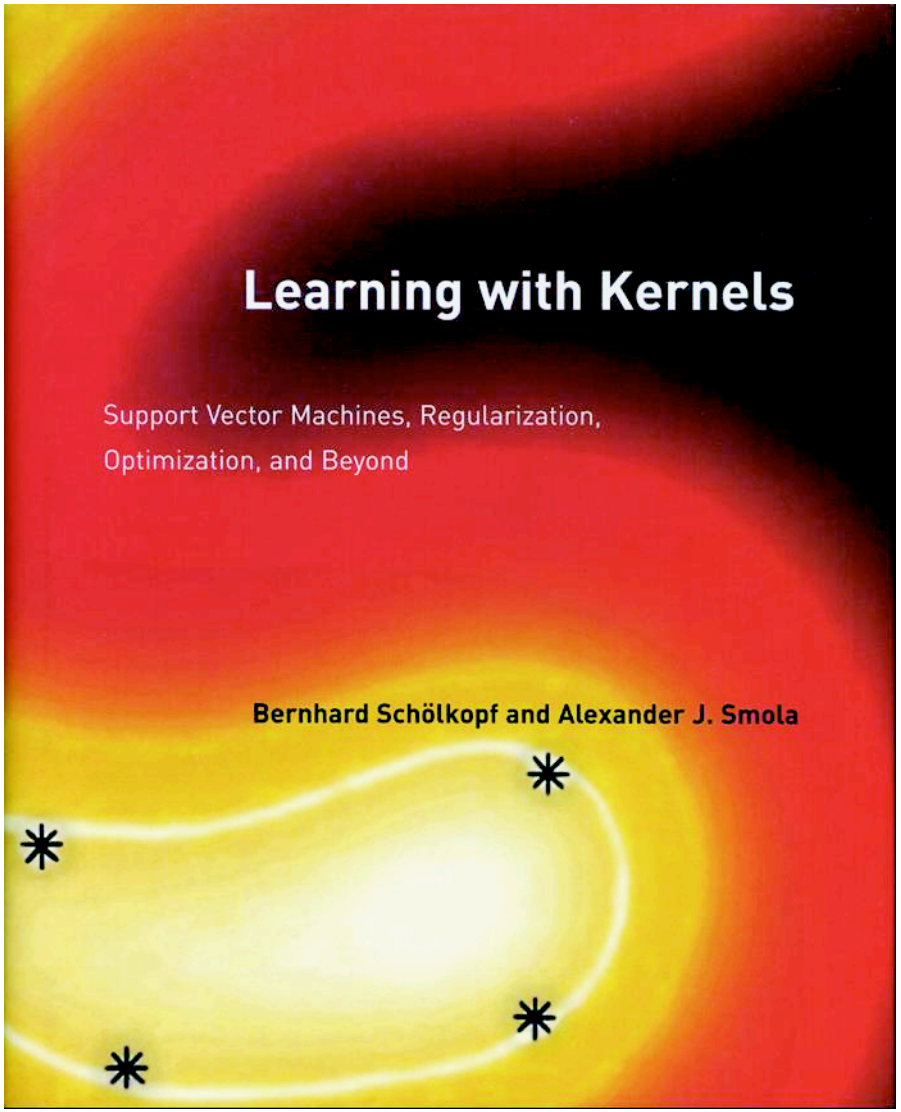
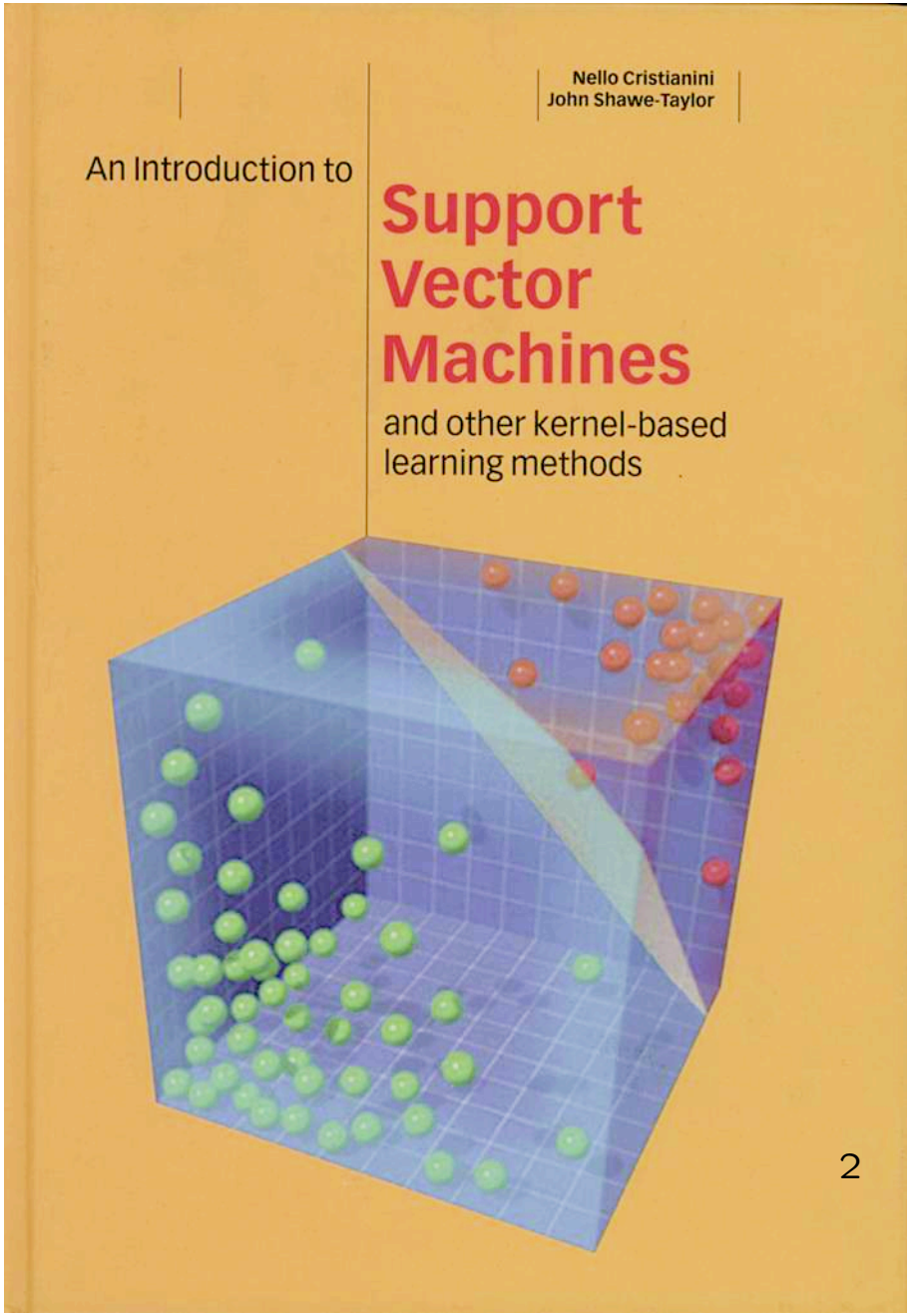
kernels (II)

presented by Virgil Pavlu

based on work by
Bernhard Scholkopf
Alexander Smola
Nello Cristianini
John Shaw-Taylor
Thorsten Joachims

where to read





this lecture

kernels

mercer conditions

SVM with kernels

designing kernels

feature extraction : kernel PCA

data similarities & dot product

- measurement of data similarities : a fundamental problem in ML
- reflects a priori knowledge of the problem/data
- dot product : a natural measure for similarity
  $\langle \mathbf{x} \cdot \mathbf{y} \rangle = \sum_i x_i \cdot y_i$
- dot product amounts to being able to carry all geometric constructions formulated in terms of angles, lengths and distances

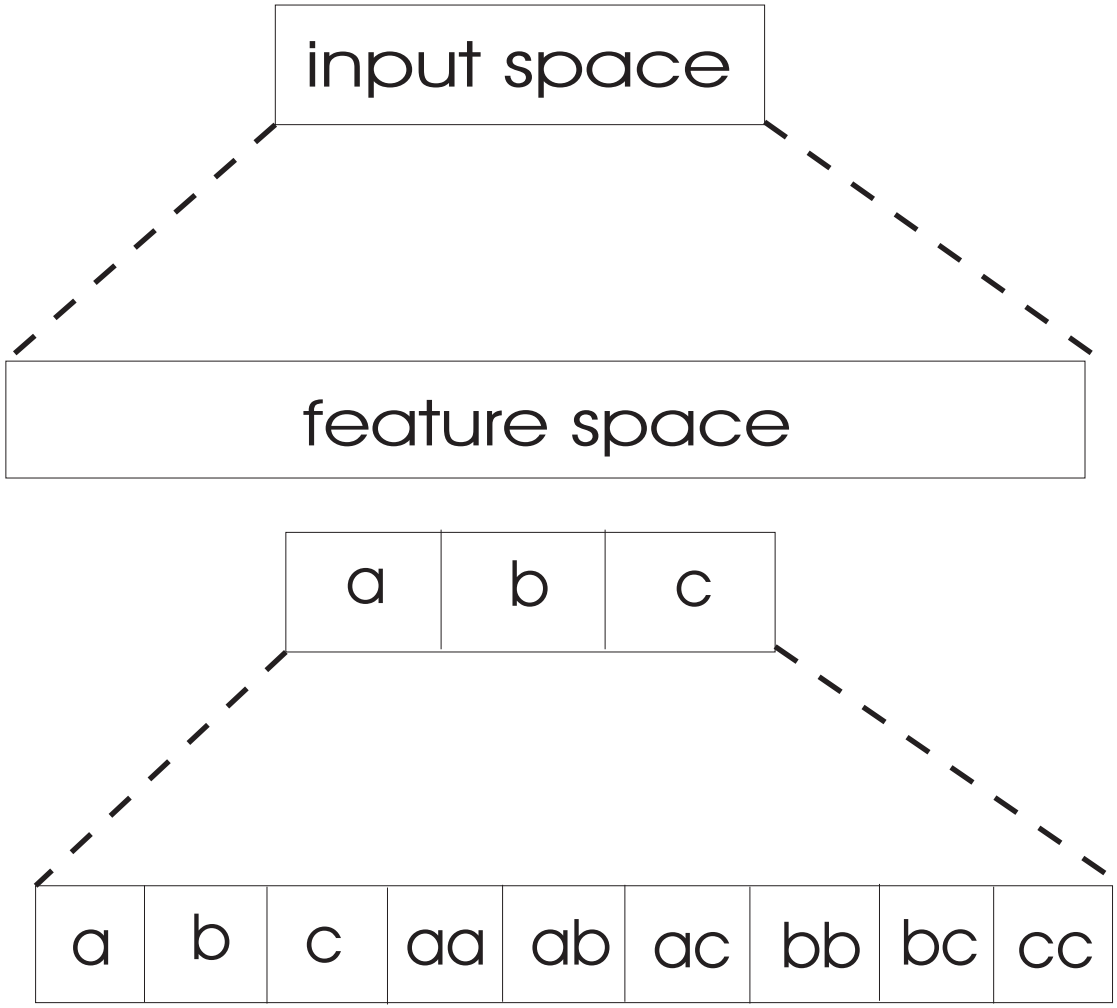
  $\cos(\mathbf{x}, \mathbf{y}) = \frac{\langle \mathbf{x} \cdot \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|}$ $\qquad$ $\|\mathbf{x}\| = \sqrt{\langle \mathbf{x} \cdot \mathbf{x} \rangle}$

feature space

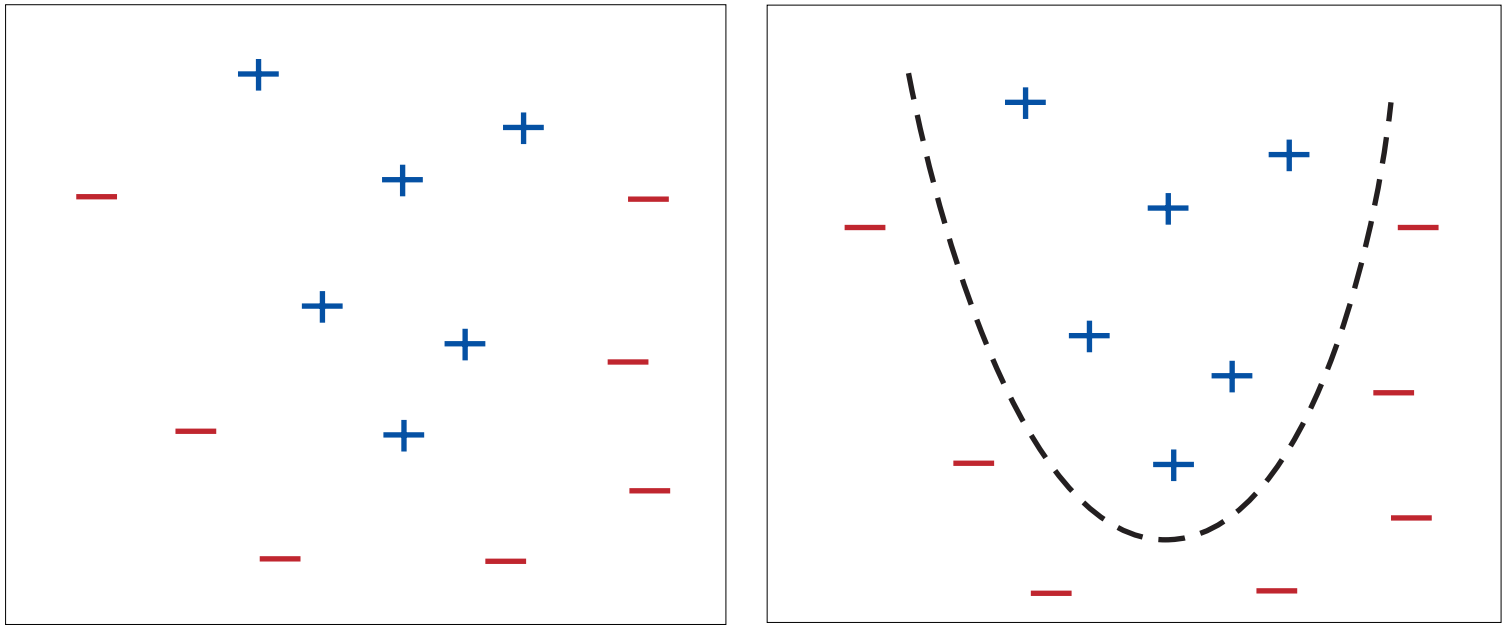
- general measure for similarity
  $k : X \times X \rightarrow \mathbf{R}$, symetric $k(\mathbf{x}, \mathbf{y}) = k(\mathbf{y}, \mathbf{x})$

- symmetry is too general, we want something that feels like dot product
  $\exists \Phi : X \rightarrow \mathbf{H}$ mapping function
  $k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})$
  where **H**=feature space (Hilbert space, supports dot product)

Φ extends the attribute space



non-linear data separation

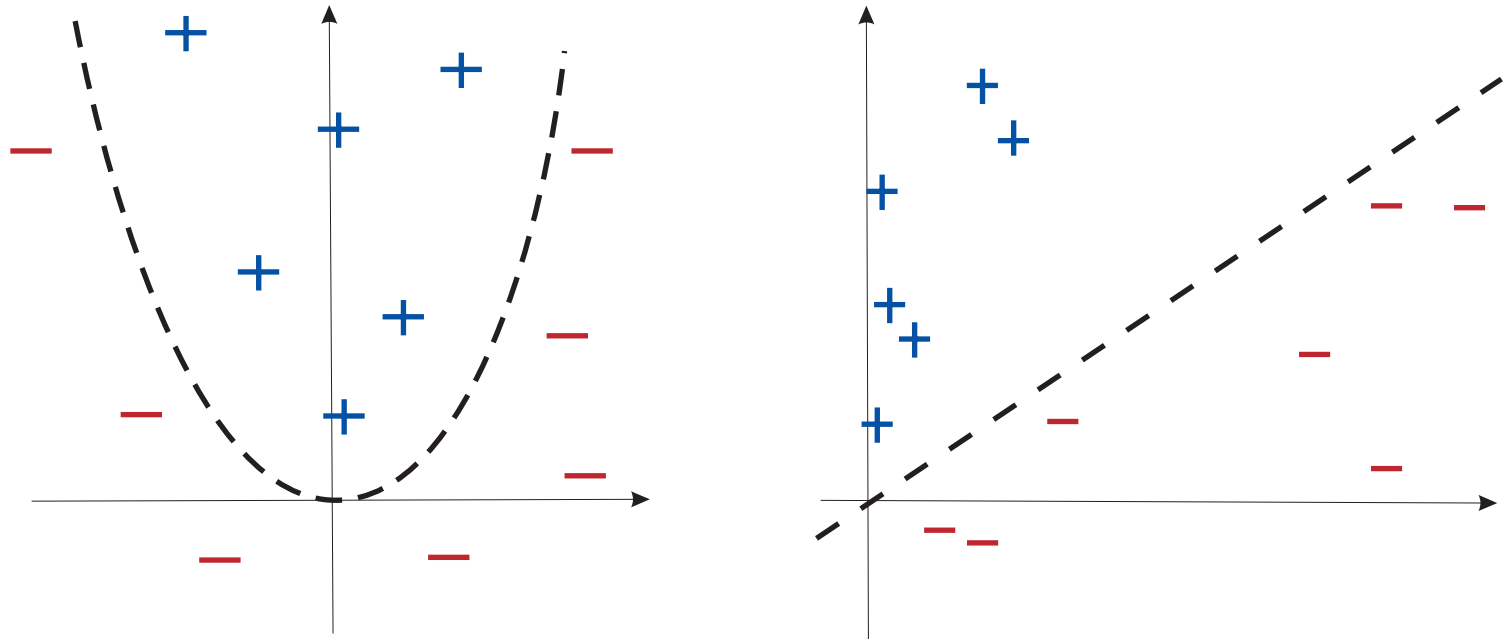


- i.e. when linear classifiers fail

- using a non-linear mapping Φ and a linear classifier in the feature space *may* succeed

feature space : example

input space : $\mathbf{x} = (x_1, x_2)$ (2 attributes)
feature space : $\Phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, 1)$ (6 attributes)



kernels

$\exists \Phi : X \to \mathbf{H}$, $k : X \times X \to \mathbf{R}$

$k(\mathbf{x}, \mathbf{y}) = k(\mathbf{y}, \mathbf{x})$
$k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})$

H= feature space, Φ= map(feature) function

- for which k there exits Φ ?
- given k, if Φ exists, it may be not unique

$\longrightarrow$

this lecture

kernels

mercer conditions

SVM with kernels

designing kernels

feature extraction : kernel PCA

linear algebra

- $\langle x \cdot Ay \rangle = \langle A^T x \cdot y \rangle$. A is **symmetric** if $A = A^T$. then $\langle x \cdot Ay \rangle = \langle Ax \cdot y \rangle$

- A is **positive definite** if A is symmetric and satisfies
$\langle x \cdot Ax \rangle = x^T Ax = \sum_{i,j} x_i a_{ij} x_j \geq 0, \forall x$

- A is **unitary (orthogonal)** if $A^T = A^{-1}$ or $AA^T = I$. then
$\langle Ax \cdot Ay \rangle = \langle A^T Ax \cdot y \rangle = \langle A_{-1} Ax \cdot y \rangle = \langle x \cdot y \rangle$

- $\det(A) \neq 0 \Leftrightarrow A$ has full rank $\Leftrightarrow \exists A^{-1}$

more linear algebra

• λ is **eigenvalue** of matrix A if there is a non-zero vector x(**eigenvector**) such that $Ax = \lambda x$. then $\det(A - \lambda I) = 0$. eigenvectors are **linear independent** if eigenvalues are different

• $\det(A) = \prod_i \lambda_i$. if a matrix is **triangular/diagonal** then its eigenvalues are exactly the diagonal entries

• if the eigenvectors $V = (v_1^T, ..., v_n^T)$ are linear independent and form an **orthonormal** base and $D = [\lambda_1, ..., \lambda_n]$ diagonal matrix then $V^{-1}AV = V^TAV = D \Leftrightarrow A = VDV^T = VDV^{-1}$(diagonalization). any symmetric matrix can be diagonalized

• **SVD** if A is $m \times n$ then $A = Q_1 M Q_2^T$; Q_1,Q_2 orthogonal, M diagonal

kernel characterization

data dependent - X finite

theorem if the Gram matrix $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ is positive definite then k is a dot product : $\exists \Phi$ such that $k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})$

proof K positive definite $\Rightarrow K = SDS^T$ (diagonalization)
where S is orthogonal and D is diagonal with non-negative entries
then $k(\mathbf{x}_i, \mathbf{x}_j) = (SDS^T)_{ij} = \langle S_i \cdot DS_j \rangle = \langle \sqrt{D} S_i \cdot \sqrt{D} S_j \rangle$
take $\Phi(\mathbf{x}_i) = \sqrt{D} S_i$

kernel characterization (converse)

data dependent - X finite

theorem if the kernel k is a dot product $\exists \Phi$, $k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})$ then the Gram matrix $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ is positive definite

proof for any $\alpha \in \mathbf{R}^m$

$$\sum_{i,j=1}^{m} \alpha_i \alpha_j K_{ij} = \langle \sum_{i=1}^{m} \alpha_i \Phi(\mathbf{x}_i), \sum_{j=1}^{m} \alpha_j \Phi(\mathbf{x}_j) \rangle = \| \sum_{i=1}^{m} \alpha_i \Phi(\mathbf{x}_i) \|^2 \geq 0$$

so K is positive definite

mercer theorem

theorem[Mercer] Let $\mathcal{X}$ be a compact subset of $\mathbf{R}^n$. Suppose $\mathcal{K}$ is a continuous symmetric function such that

$$\int_{\mathcal{X}} \int_{\mathcal{X}} \mathcal{K}\Big(\mathbf{x}, \mathbf{z}\Big) f\Big(\mathbf{x}\Big) f\Big(\mathbf{z}\Big) d\mathbf{x} d\mathbf{z} \geq 0$$

for all $f \in \mathcal{L}_2\Big(\mathcal{X}\Big)$. Then, $\mathcal{K}\Big(\mathbf{x}, \mathbf{z}\Big)$ can be expanded in a uniformly convergent series

$$\mathcal{K}\Big(\mathbf{x}, \mathbf{z}\Big) = \sum_{j=1}^{\infty} \lambda_j \phi_j\Big(\mathbf{x}\Big) \phi_j\Big(\mathbf{z}\Big)$$

in terms of the eigenfunctions $\phi_j \in \mathcal{L}_2\Big(\mathcal{X}\Big)$ of $\Big(\mathcal{T}_{\mathcal{K}} f\Big)\Big(\cdot\Big) = \int_{\mathcal{X}} \mathcal{K}\Big(\cdot, \mathbf{x}\Big) f\Big(\mathbf{x}\Big) d\mathbf{x}$ normalized so that $||\phi_j||_{\mathcal{L}_2} = 1$ and positive associated eigenvalues $\lambda_j \geq 0$.

valid kernels

- $\mathcal{K}\big(\mathbf{x},\mathbf{z}\big) = \mathcal{K}_1\big(\mathbf{x},\mathbf{z}\big) + \mathcal{K}_2\big(\mathbf{x},\mathbf{z}\big)$
- $\mathcal{K}\big(\mathbf{x},\mathbf{z}\big) = a\mathcal{K}_1\big(\mathbf{x},\mathbf{z}\big)$
- $\mathcal{K}\big(\mathbf{x},\mathbf{z}\big) = \mathcal{K}_1\big(\mathbf{x},\mathbf{z}\big)\mathcal{K}_2\big(\mathbf{x},\mathbf{z}\big)$
- $\mathcal{K}\big(\mathbf{x},\mathbf{z}\big) = f\big(\mathbf{x}\big)f\big(\mathbf{z}\big)$
- $\mathcal{K}\big(\mathbf{x},\mathbf{z}\big) = \mathcal{K}_3\big(\phi\big(\mathbf{x}\big),\phi\big(\mathbf{z}\big)\big)$

$p\big(x\big)$ a polynomial with positive coefficients

- $\mathcal{K}\big(\mathbf{x},\mathbf{z}\big) = p\big(\mathcal{K}_1\big(\mathbf{x},\mathbf{z}\big)\big)$
- $\mathcal{K}\big(\mathbf{x},\mathbf{z}\big) = exp\big(\mathcal{K}_1\big(\mathbf{x},\mathbf{z}\big)\big)$
- $\mathcal{K}\big(\mathbf{x},\mathbf{z}\big) = exp\big(-||\mathbf{x}-\mathbf{z}||^2/\sigma^2\big)$

dot product kernels

$k(\mathbf{x}, \mathbf{y}) = k(\langle \mathbf{x}, \mathbf{y} \rangle)$

theorem

•the function k of the dot product kernel must satisfy
$k(t) \geq 0, k'(t) \geq 0$ and $k'(t) + tk''(t) \geq 0 \ \forall t \geq 0$
in order to be a positive definite kernel. that may still be insufficient

• if k s a power series expansion

$$k(t) = \sum_{n=0}^{\infty} a_n t^n$$

then k is a positive definite kernel iff $\forall n, a_n \geq 0$

this lecture

kernels

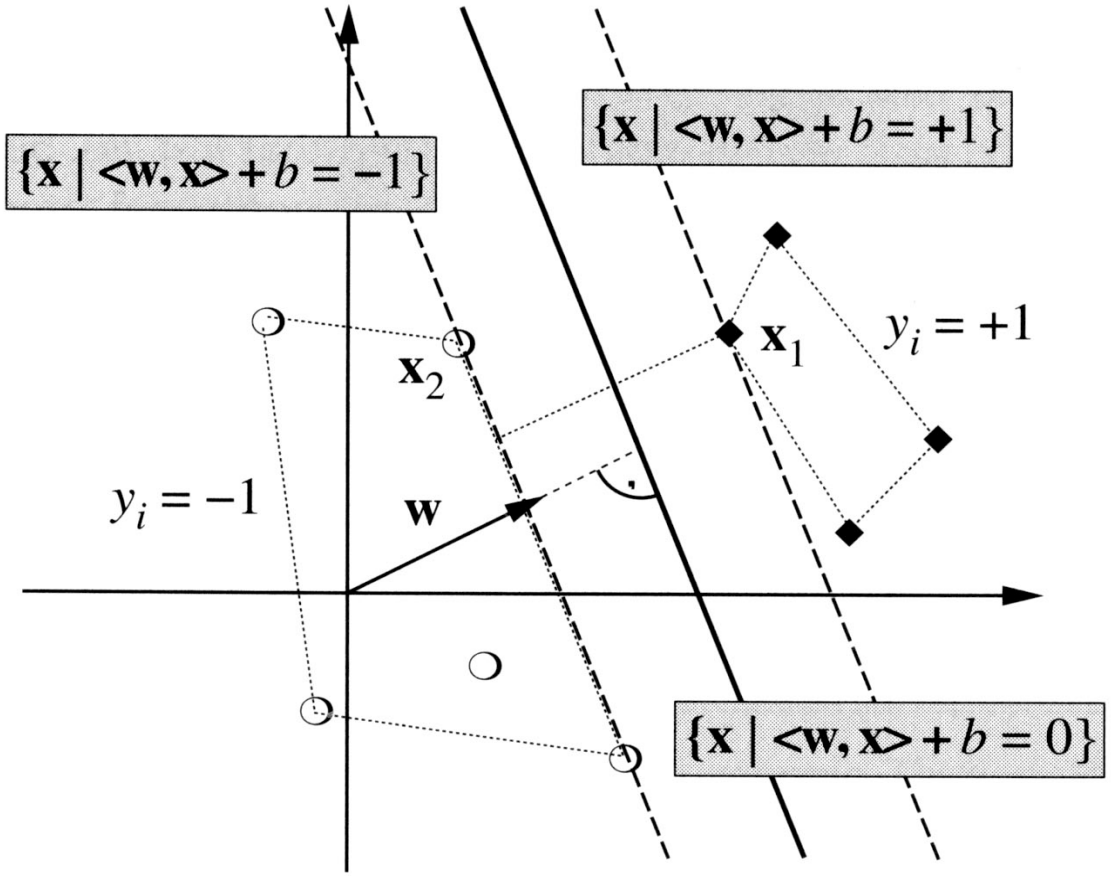
mercer conditions

SVM with kernels

designing kernels

feature extraction : kernel PCA

SVMs



Note:

$$\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = +1$$
$$\langle \mathbf{w}, \mathbf{x}_2 \rangle + b = -1$$

$$\Rightarrow \quad \langle \mathbf{w}, (\mathbf{x}_1 - \mathbf{x}_2) \rangle = 2$$

$$\Rightarrow \left\langle \frac{\mathbf{w}}{\|\mathbf{w}\|}, (\mathbf{x}_1 - \mathbf{x}_2) \right\rangle = \frac{2}{\|\mathbf{w}\|}$$

plug a kernel into SVM

after a long discussion on optimization theory...

the primal problem

minimize $\langle \mathbf{w} \cdot \mathbf{w} \rangle$

subject to $y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \forall i$

the dual problem

maximize $P(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$

subject to $\sum_{i=1}^{m} y_i \alpha_i = 0, \alpha_i \geq 0, \forall i$

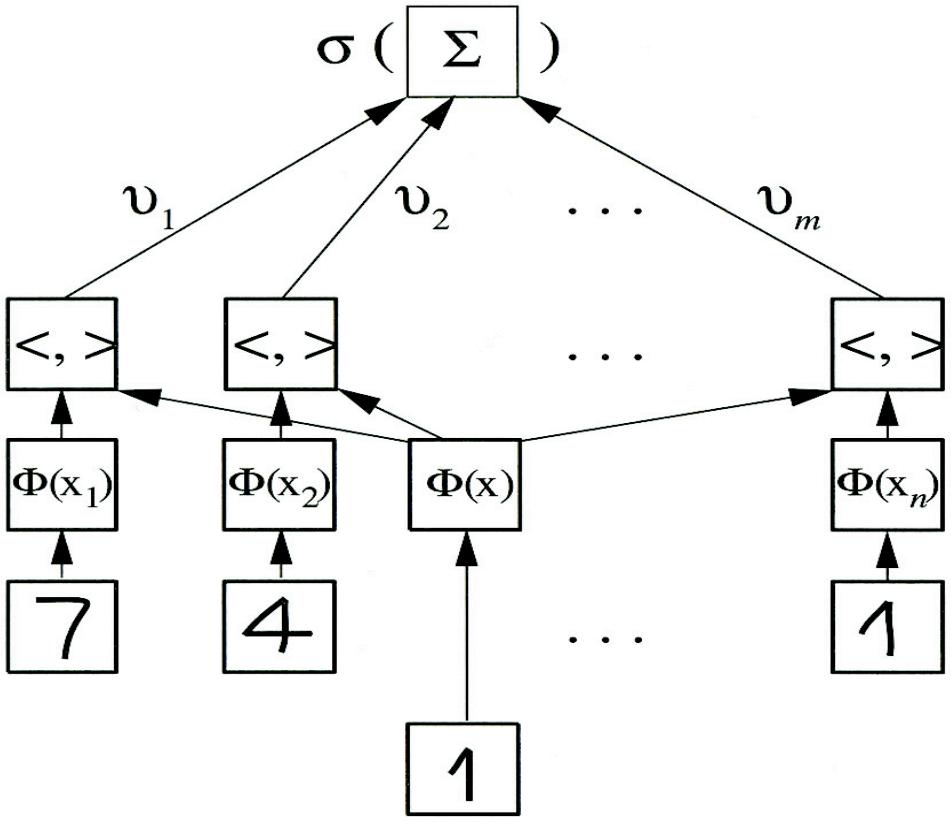
kernel trick replace the dot product $\langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$ with a kernel $k(\mathbf{x}_i, \mathbf{x}_j)$:

maximize

$$P(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j)$$

subject to $\sum_{i=1}^{m} y_i \alpha_i = 0, \alpha_i \geq 0, \forall i$

SVM with kernels



output $\sigma\ (\Sigma\ \upsilon_i\ k\ (x,x_i))$

weights

dot product $\langle\Phi(x),\Phi(x_i)\rangle = k(x,x_i)$

mapped vectors $\Phi(x_i)$, $\Phi(x)$

support vectors x_1 ... x_n

test vector x

the kernel trick

maximize

$$P(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j)$$

subject to $\sum_{i=1}^{m} y_i \alpha_i = 0, \alpha_i \geq 0, \forall i$

- we need only the kernel k ,**not** Φ - thats good...
- any algorithm that only depends on dot products (rotationally invariant) can be kernelized
- any algorithm that is formulated in terms of positive definite kernel(s) supports a kernel-replace
- math was around for long time (1940s Kolgomorov, Aronszajn, Schoenberg) but the practical importance was underestimated

SVM, concept class, good kernels

C a concept class = set of concepts

a kernel is **complete** if it is "fine-grained" enough
$k(\mathbf{x}_i, \cdot) = k(\mathbf{x}_j, \cdot) \Rightarrow c(\mathbf{x}_i) = c(\mathbf{x}_j), \forall c \in C$

a kernel is **correct(linear-good** wrt to C) if an SVM with perfect separation can be learned with it
$\forall c \in C, \exists \mathbf{w}$ such that $\sum_i w_i k(\mathbf{x}_i, \mathbf{x}) \geq 0 \Leftrightarrow c(\mathbf{x})$

this lecture

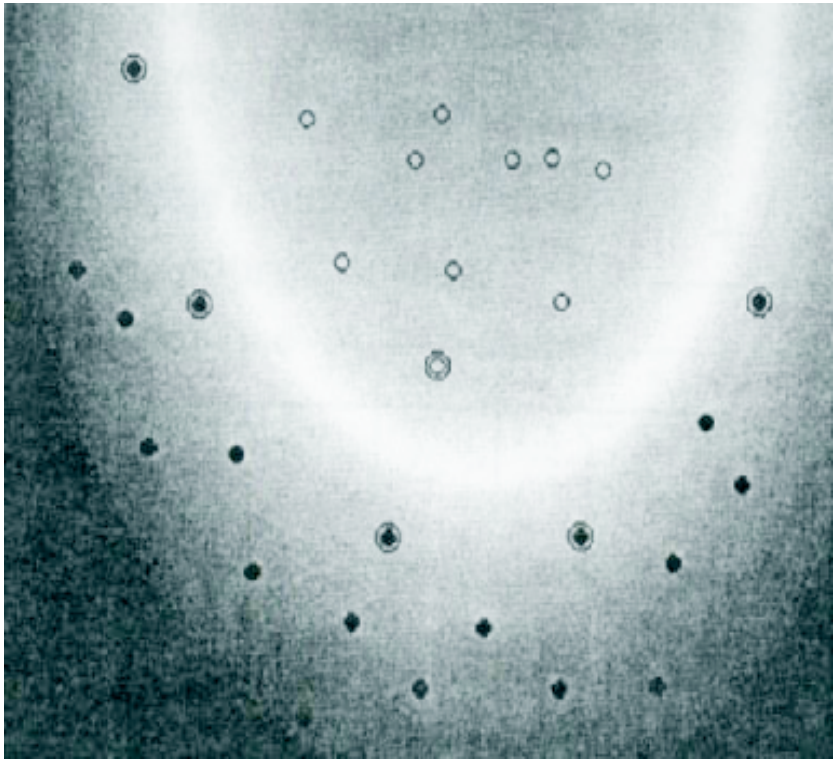
kernels
mercer conditions
SVM with kernels
designing kernels
feature extraction : kernel PCA

polynomial kernel

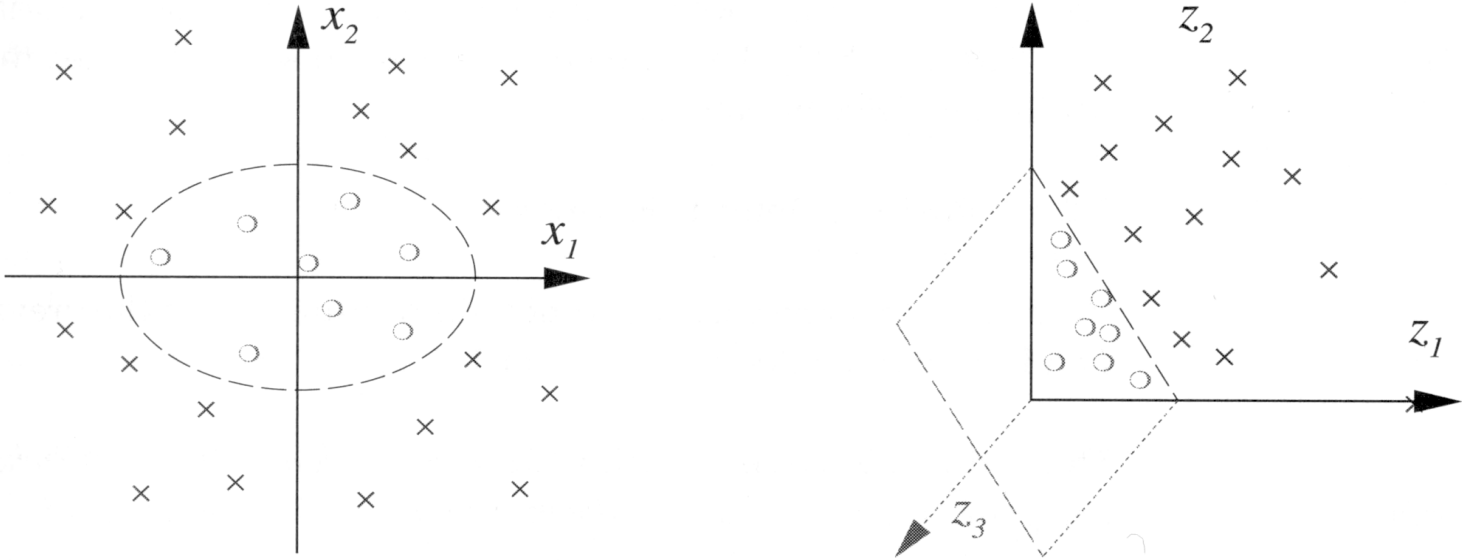
theorem define the map $\mathbf{x} \rightarrow C_d(\mathbf{x})$ where $C_d(\mathbf{x})$ the vector consisting in all possible d^{th} degree ordered products of the entries of $\mathbf{x}$=$(x_1, x_2, ..., x_N)$ then $\langle C_d(\mathbf{x}), C_d(\mathbf{y})\rangle = \langle \mathbf{x}, \mathbf{y}\rangle^d$

$k(\mathbf{x}, \mathbf{y}) = (\langle \mathbf{x}, \mathbf{y}\rangle + c)^d$
polynomial kernel

• invariant to group of all orthogonal transformations(rotations, mirroring)



polynomial kernel : toy example



use the map $\mathbf{x}=(x_1, x_2) \rightarrow \Phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$
ellipse from 2D-input space becomes hyperplane into 3D-feature space

note $C_2(\mathbf{x}) = (x_1^2, x_2^2, x_1x_2, x_2x_1)$ maps data in a 4D-feature space but it generates the same kernel
$k(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle = \langle C_2(\mathbf{x}), C_2(\mathbf{y}) \rangle = x_1^2y_1^2 + x_2^2y_2^2 + 2x_1y_1x_2y_2$

Gaussian RadialBasisFunction kernel

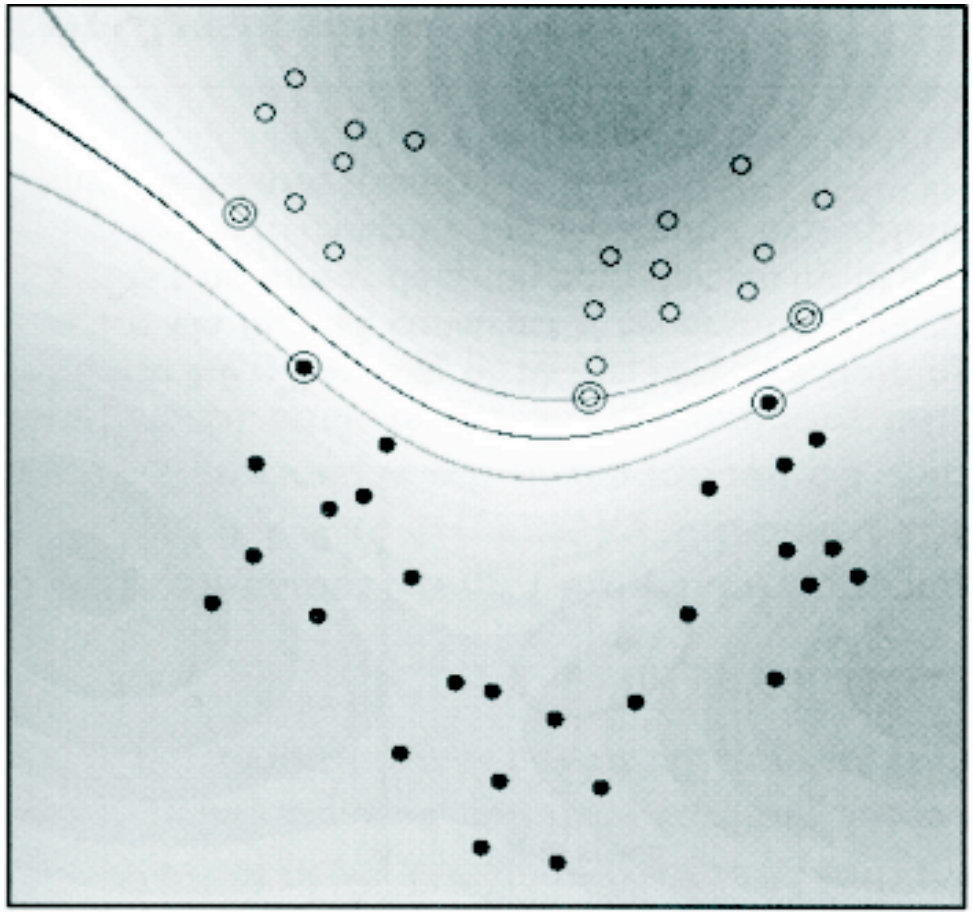
$k(\mathbf{x},\mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2})$

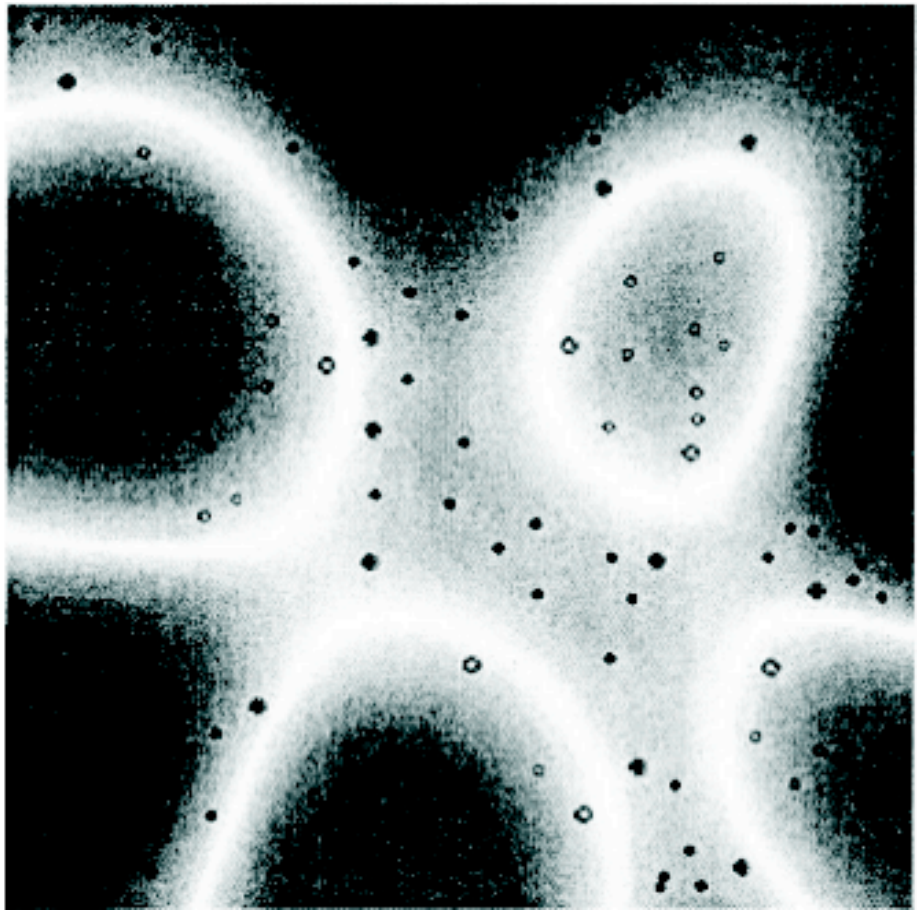
more general $k(\mathbf{x},\mathbf{y}) = f(d(\mathbf{x},\mathbf{y}))$
where d is a metric on X and f is a function on $\mathbf{R}_0^+$; usually d arises from dot product $d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|$

- invariant on translations $k(\mathbf{x},\mathbf{y}) = k(\mathbf{x}+\mathbf{z},\mathbf{y}+\mathbf{z})$
- $\cos(\angle(\Phi(\mathbf{x}),\Phi(\mathbf{y}))) = \langle\Phi(\mathbf{x}),\Phi(\mathbf{y})\rangle = k(\mathbf{x},\mathbf{y}) \geq 0 \Rightarrow$ enclosed angle between any 2 mapped points is smaller than $\pi/2$

theorem if $X = \{\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_m\}$ all distinct and $\sigma > 0$ then the matrix $K_{ij} = \exp(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2})$ has full rank $\Rightarrow \Phi(\mathbf{x}_1),\Phi(\mathbf{x}_2),...,\Phi(\mathbf{x}_m)$ are linear independent

RBF kernel SVM





comparison

	SVM	KFD	RBF	AB	AB_R
Banana	11.5±0.07	**10.8±0.05**	**10.8±0.06**	12.3±0.07	*10.9±0.04*
Breast Cancer	*26.0±0.47*	**25.8±0.46**	27.6±0.47	30.4±0.47	26.5±0.45
Diabetes	*23.5±0.17*	**23.2±0.16**	24.3±0.19	26.5±0.23	23.8±0.18
German	**23.6±0.21**	*23.7±0.22*	24.7±0.24	27.5±0.25	24.3±0.21
Heart	**16.0±0.33**	*16.1±0.34*	17.6±0.33	20.3±0.34	16.5±0.35
Image	*3.0±0.06*	3.3±0.06	3.3±0.06	**2.7±0.07**	**2.7±0.06**
Ringnorm	1.7±0.01	**1.5±0.01**	1.7±0.02	1.9±0.03	*1.6±0.01*
F. Sonar	**32.4±0.18**	*33.2±0.17*	34.4±0.20	35.7±0.18	34.2±0.22
Splice	10.9±0.07	10.5±0.06	*10.0±0.10*	10.1±0.05	**9.5±0.07**
Thyroid	4.8±0.22	**4.2±0.21**	4.5±0.21	*4.4±0.22*	4.6±0.22
Titanic	**22.4±0.10**	23.2±0.20	23.3±0.13	*22.6±0.12*	*22.6±0.12*
Twonorm	3.0±0.02	**2.6±0.02**	2.9±0.03	3.0±0.03	*2.7±0.02*
Waveform	*9.9±0.04*	*9.9±0.04*	10.7±0.11	10.8±0.06	**9.8±0.08**

Fisher kernel

- knowledge about objects in form of a generative probability model
- deals with mising/incomplete data, uncertainty, variable length

family of generative models (density functions)
$p(x|\theta)$, smoothly parametrized by $\theta = (\theta^1, ..., \theta^r)$; $l(x, \theta) = \ln p(x|\theta)$

score $V_\theta(x) := (\delta_{\theta^1} l(x, \theta), ..., \delta_{\theta^r} l(x, \theta)) = \nabla_\theta l(x, \theta) = \nabla_\theta \ln p(x|\theta)$

Fisher information matrix $I := \mathbf{E}_p[V_\theta(x) V_\theta(x)^T]$
$I_{ij} = \mathbf{E}_p[\delta_{\theta^i} \ln p(x|\theta) \cdot \delta_{\theta^j} \ln p(x|\theta)]$, $\mathbf{E}_p$ is called *Fisher information metric*

Fisher kernel
$K_I(x, y) := V_\theta(x)^T I^{-1} V_\theta(y)$
natural kernel M positive definite matrix
$K_M^{nat}(x, y) := V_\theta(x)^T M^{-1} V_\theta(y)$

[information] diffusion kernel

- local relationships

the exponential of a suared matrix H is
$e^{\beta H} = \lim_{n\to\infty}(1+\frac{\beta H}{n})^n = I + \beta H + \frac{\beta^2}{2!}H^2 + \frac{\beta^3}{3!}H^3 + ...$

exponential kernel $K_\beta = e^{\beta H}$, $\frac{\delta K_\beta}{\delta\beta} = HK_\beta$(heat eq)

diffusion kernel on graph : consider
$H_{ij} = 1$ if i j ; $-d_i$ (degree) if $i = j$; 0 otherwise
$w^T Hw = -\sum_{i,j\in E}(w_i - w_j)^2$ negative semidefinite
$-H$=Laplacian of the graph

two approaches to kernel design

model driven - encodes knowledge about domain

- polynomial,Gaussian
- from generative models : Fisher kernel
- local relationships : diffusion kernel

syntax driven - exploits structure of the problem

- terms : convolution kernel
- text classification : string kernel
- tree kernel

- **particulary useful for non-vectorial data**

convolution kernel

kernel between composite objects building on similarities of resp. parts $k_d : X_d \times X_d \to \mathbf{R}$, R-relation. define the R-convolution kernel

$$(k_1 \star k_2 \star ... \star k_D)(\mathbf{x}, \mathbf{y}) := \sum_R \prod_{d=1}^{D} k_d(x_d, y_d)$$

where the sum runs over all possible decompositions of $\mathbf{x} \to (x_1, x_2, ..., x_D)$ and of $\mathbf{y} \to (y_1, y_2, ..., y_D)$ s.t. $R(\mathbf{x}, x_1, x_2, ..., x_D)$ and $R(\mathbf{y}, y_1, y_2, ..., y_D)$

- proved valid if R is finite

ANOVA kernel (analysis of variance)

if $X = S^N$ and $k^{(i)}$ kernel on $S \times S$ for $i = 1, 2, ..., N$,the ANOVA kernel of order D is

$$k_D(\mathbf{x}, \mathbf{y}) := \sum_{1 \le i_1 < ... < i_D \le N} \prod_{d=1}^{D} k^{i_d}(x_{i_d}, y_{i_d})$$

string kernel - similarities between two documents

$\sum$=alphabet, $\sum^n$=set of all strings of length n

for a given index sequence $\mathbf{i} = (1 \leq i_1 < i_2 < ... < i_r \leq |s|)$

define $s(\mathbf{i}) := s(i_1)s(i_2)....s(i_r)$ and $l_s(\mathbf{i}) = i_r - i_1 + 1 \geq r$

example $s = \texttt{fast food}$,$\mathbf{i} = (2, 3, 9) \Rightarrow s(\mathbf{i}) = \texttt{asd}, l_s(\mathbf{i}) = 9 - 2 + 1 = 8$

$0 < \lambda \leq 1$ parameter, define $[\Phi_n(s)]$ a map with $|\sum^n|$ components

$$[\Phi_n(s)]_u = \sum_{\mathbf{i}:s(\mathbf{i})=u} \lambda^{l_s(\mathbf{i})}$$

example $[\Phi_3(\texttt{Nasdaq})]_{\texttt{asd}} = \lambda^3$, $[\Phi_3(\texttt{lass das})]_{\texttt{asd}} = 2\lambda^5$

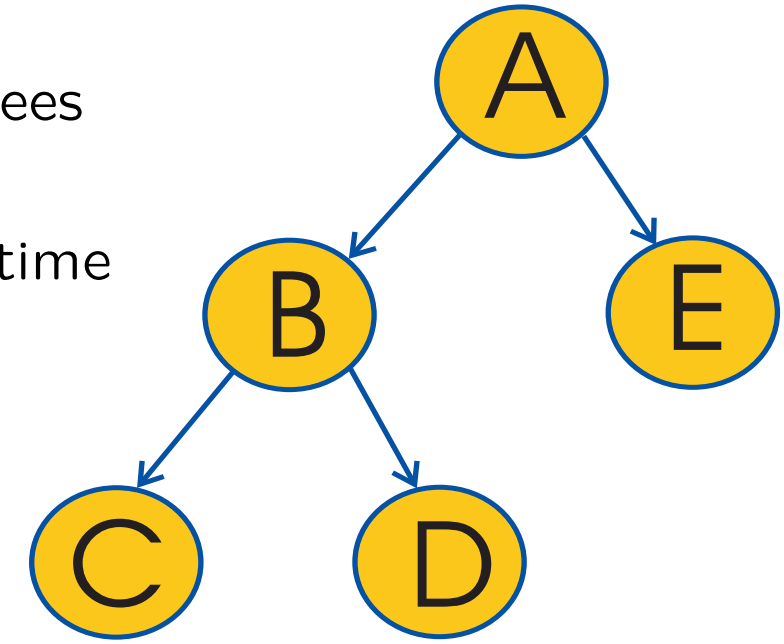
the kernel induced

$$k_n(s,t) = \sum_{u \in \sum^n} [\Phi_n(s)]_u [\Phi_n(t)]_u = \sum_{u \in \sum^n} \sum_{(\mathbf{i},\mathbf{j}):s(\mathbf{i})=t(\mathbf{j})=u} \lambda^{l_s(\mathbf{i})} \lambda^{l_t(\mathbf{j})}$$

$k := \sum_n c_n k_n$ linear combination of kernels on different substring-lengths

tree kernel

- encode a tree as a string by traversing in preorder and parenthesing
- substrings correspond to subset trees
- tag can be computed in loglinear time
- then use a string kernel



`tag(T)=(A(B(C)(D))(E))`

kernels correspond to

- similarity measure for the data
- linear representation of the data
- function space for learning
- covariance function for correlated observations
- prior over he set of functions

the kernel *is* the prior knowledge we have about the problem and its solution - no free lunch here

this lecture

kernels
mercer conditions
SVM with kernels
designing kernels
feature extraction : kernel PCA

Principal Component Analysis

technique for extracting structure from possible high-dim data sets

given **observations** $x_i \in \mathbf{R}^N, i = 1, ..., m$
centered : $\sum_{i=1}^{m} x_i = 0$
form the **covariance matrix** $C = \frac{1}{m} \sum_{j=1}^{m} x_j x_j^T$, positive definite

C can be diagonalized with non negative eigenvalues.To do this, solve the eigenvalue eq $\lambda v = Cv$ for $\lambda \geq 0$ and non-zero eigenvectors $v \in \mathbf{R}^N$
equation becomes

$$\lambda v = Cv = \frac{1}{m} \sum_{j=1}^{m} \langle x_j, v \rangle x_j$$

all v with $\lambda \neq 0$ lie in the span of $x_1 ....x_m$ hence the eigenvalue eq becomes $\lambda \langle x_i, v \rangle = \langle x_i, Cv \rangle, \forall i$

kernel PCA

$\Phi : X \to \mathbf{H}$ (possibly nonlinear) map, cenetred $\sum_{i=1}^{m} \Phi(x_i) = 0$.

the covariance matrix $C = \frac{1}{m} \sum_{j=1}^{m} \Phi(x_j)\Phi(x_j)^T$.

as in PCA , we need to find the eigenvalues and eigenvectors satisfying $\lambda v = Cv$.note that solutions lie in the spam of $\Phi(x_1), ..., \Phi(x_m)$ or $v = \sum_{i=1}^{m} \alpha_1 \Phi(x_i)$ and equation is equiv to $\lambda \langle \Phi(x_i), v \rangle = \langle \Phi(x_i), Cv \rangle, \forall i$ which becomes

$$\lambda \sum_{i=1}^{m} \alpha_i \langle \phi(x_n), \Phi(x_i) \rangle = \frac{1}{m} \sum_{i=1}^{m} \alpha_i \langle \Phi(x_n), \sum_{j=1}^{m} \Phi(x_j) \langle \Phi(x_j), \Phi(x_i) \rangle \rangle$$

if $K_{ij} = \langle \Phi(x_i), \Phi(x_j) \rangle$ (Gram matrix) then we need to find non-zero solutions of $m\lambda K\alpha = K^2\alpha$ which are between solutions of $m\lambda\alpha = K\alpha$

kernel PCA - properties

kernel PCA is the orthogonal basis transformation in **H** with following properties (assuming eigenvectors in descending order of eigenvalues):

•first q principal components(proj. on eigenvectors) carry more variance that any other q orthogonal directions
•the mean-squared approx. error when representing (x_i) by the q first principal components is minimal
• the principal components are uncorrelated
• the first q principal components have max mutual information
• connection with SVM : the n^{th} KPCA feature extractor ,scaled by $1/\lambda_n$is optimal among all feature extractions, in the sense that it has minimal weight vector norm in the RKHS **H**,
$\|v\|^2 = \sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j)$
subject to orthogonality and unit variance set of outputs when applied to training set (x_i)

important things not covered

- regularization
- kernel fisher discriminant
- bayesian kernel methods
- locality-improved kernels

END

don't look after this slide

bayesian kernel methods

kernels and Gaussian processes

kernel fisher discriminant