Course ID 80250993 Dates: 9/16-12/23/2021 @ I-205
Tencent/VooV Meeting 521 4678 6257 (passwd 1205)

# Lecture 6
# Support Vector Machines

Xuegong Zhang
Oct. 14, 2020

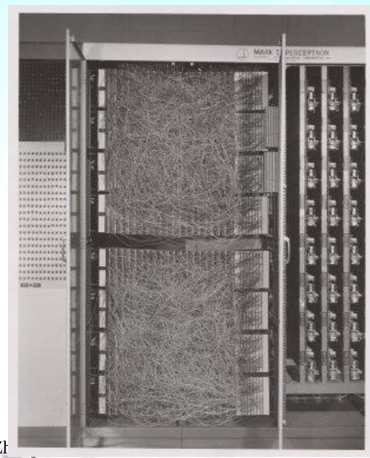


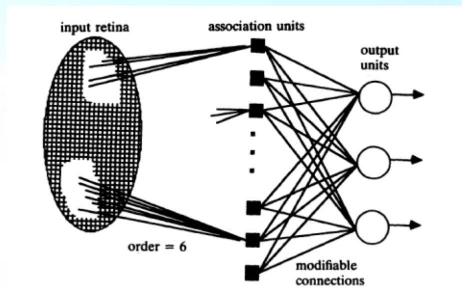
# 6.1
# Large Margin and Optimal Hyperplane

# Perceptron

Frank Rosenblatt, *The Perceptron – a perceiving and recognizing automaton*, Report 85-460-1, Cornell Aeronautical Laboratory, Jan. 1957

$$y = \mathrm{sgn}\left(\sum_{i=1}^{d} w_i x_i + w_0\right)$$



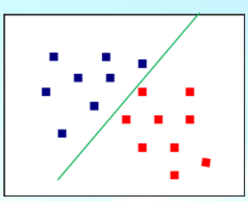


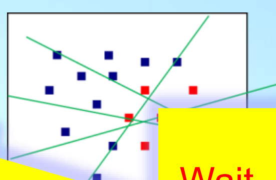

https://en.wikipedia.org/wiki/Perceptron

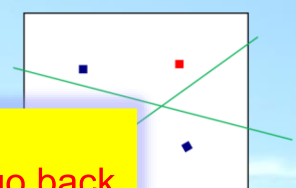
M. Olazaran, A sociological study of the official history of the perceptrons controversy, *Social Studies of Science*, 1996



# Linearly non-separable cases







Linearly separable

Not perfectly sepa
but can achieve s
reasonable separation
with a linear classifier.

Wait, let's go back to consider the easiest case.

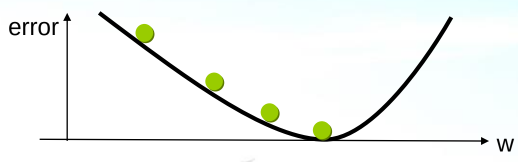
nable linear
r approximation.

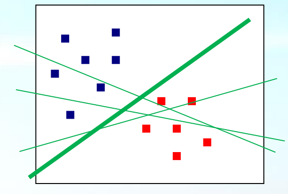
# Perceptron

- To train the machine $y = \mathrm{sgn}(\sum_{i=1}^{d} w_i x_i + w_0)$ with data $\{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$.
- Goal: to optimize an objective function $J_P(\boldsymbol{\alpha}) = \sum_{\mathbf{y}_j \in \mathcal{Y}^k}(-\boldsymbol{\alpha}^T \mathbf{y}_j)$
  - Learning algorithm
    - *Basic Gradient Descent*

$$\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}(k) - \rho_k \nabla J$$

Any question here?



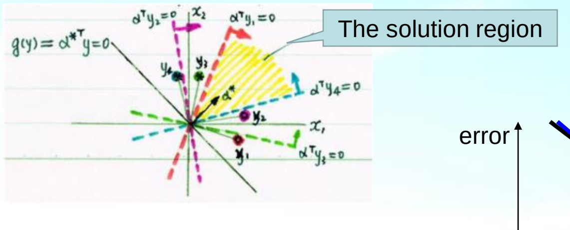



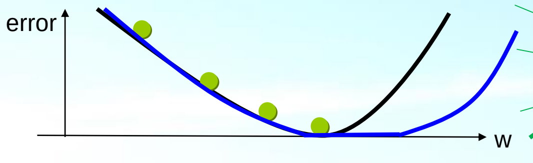


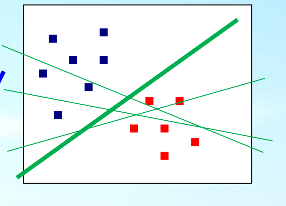
# Question of Perceptron

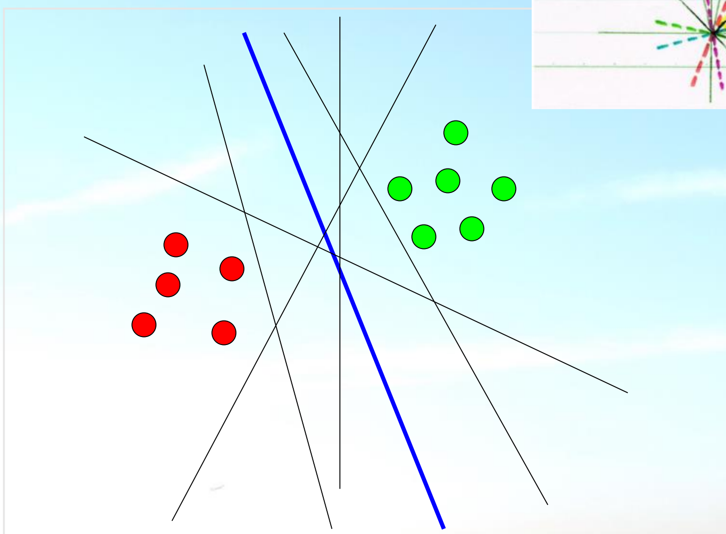
$$g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0 = \boldsymbol{\alpha}^T \boldsymbol{y} \qquad \min\ J_P(\boldsymbol{\alpha}) = \sum_{\mathbf{y}_j \in \mathcal{Y}^k}(-\boldsymbol{\alpha}^T \mathbf{y}_j)$$

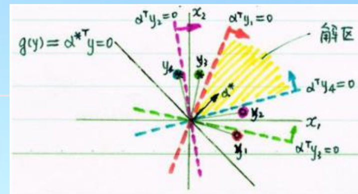
- There are multiple solutions, which one is the best?

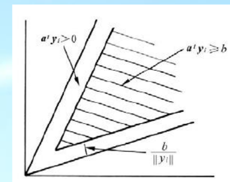


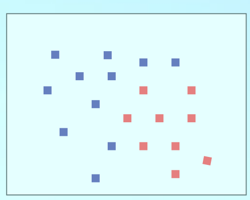




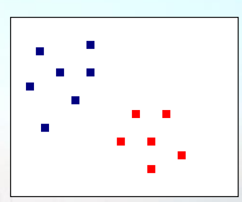




- How can we get a “best” linear classifier?



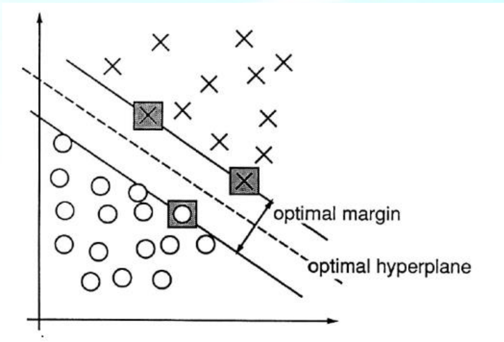
- What is “the best” when no one is perfect?
  - A basic principle that differentiates different methods.
- What is “the best” when everyone looks perfect?



# The Optimal Hyperplane

For sample set $(\mathrm{x}_1, y_1), \cdots, (\mathrm{x}_l, y_l)$, $\mathrm{x} \in R^d$, $y \in \{+1, -1\}$ that can be separated with a hyperplane $(\boldsymbol{w} \cdot \boldsymbol{x}) + b = 0$, **the optimal hyperplane** is defined as the linear decision function (hyperplane) $f(\boldsymbol{x}) = \mathrm{sgn}\big((\boldsymbol{w} \cdot \boldsymbol{x}) + b\big)$ with **maximal margin** between the vectors (samples) of the two classes among all hyperplanes that separate the two classes without error.

Vapnik




Separation:

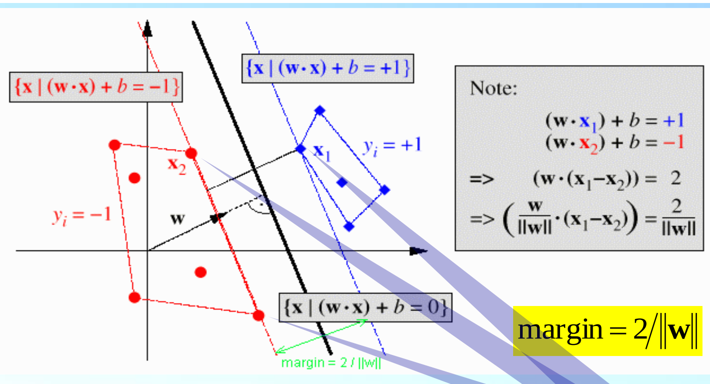
$$(\boldsymbol{w} \cdot \boldsymbol{x}_i) + b > 0, \quad \text{if} \quad y_i = 1$$
$$(\boldsymbol{w} \cdot \boldsymbol{x}_i) + b < 0, \quad \text{if} \quad y_i = -1$$
$$y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) > 0, \qquad \mathrm{i} = 1, \cdots l$$

D

To fix the scale, we require

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1, \qquad \mathrm{i} = 1, \cdots l$$






Support Vectors
(the ones with "=")

- Normalization

$$(\boldsymbol{w} \cdot \boldsymbol{x}_i) + b \geq 1, \quad \text{if} \quad y_i = 1$$
$$(\boldsymbol{w} \cdot \boldsymbol{x}_i) + b \leq -1, \quad \text{if} \quad y_i = -1$$

i.e., $y_i[(\boldsymbol{w} \cdot \boldsymbol{x}_i) + b] \geq 1, \quad i = 1, \cdots, l$

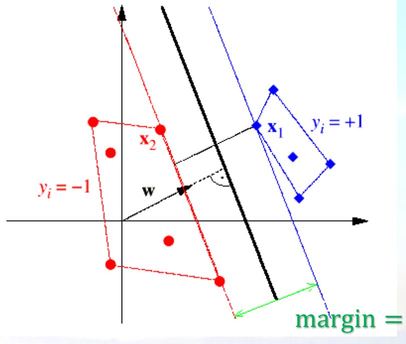
-- The canonical form of the separating hyperplane

# The Optimal Hyperplane

min $\Phi(\boldsymbol{w}) = \frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{w})$ *w.r.t.* $\boldsymbol{w}$

s.t. $y_i[(\boldsymbol{x}_i\cdot\boldsymbol{w})+b] \geq 1, \quad i = 1,2,\cdots,l$

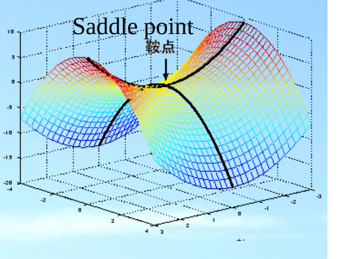
for training samples $(y_1,\boldsymbol{x}_1),\cdots,(y_l,\boldsymbol{x}_l),\ y\in\{-1,1\}$






The solution: the **saddle point** of the Lagrangian

$$L(\boldsymbol{w},b,\alpha) = \frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{w}) - \sum_{i=1}^{l}\alpha_i\{[(\boldsymbol{x}_i\cdot\boldsymbol{w})+b]y_i - 1\}$$




i.e.,

$$\min_{\boldsymbol{w},b}\max_{\alpha} L(\boldsymbol{w},b,\alpha) = \frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{w}) - \sum_{i=1}^{l}\alpha_i\{[(\boldsymbol{x}_i\cdot\boldsymbol{w})+b]y_i - 1\}$$

At the saddle point, we have

$$\frac{\partial L(\boldsymbol{w}_0,b_0,\alpha^0)}{\partial b} = 0 \quad \Longrightarrow \quad \sum_{i=1}^{l}\alpha_i^0 y_i = 0$$

$$\frac{\partial L(\boldsymbol{w}_0,b_0,\alpha^0)}{\partial \boldsymbol{w}} = 0 \quad \Longrightarrow \quad \boldsymbol{w}_0 = \sum_{i=1}^{l} y_i\alpha_i^0\boldsymbol{x}_i$$

H

R

$$\min_{\boldsymbol{w},b}\max_{\alpha} L(\boldsymbol{w},b,\alpha)=\frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{w})-\sum_{i=1}^{l}\alpha_i\{[(\boldsymbol{x}_i\cdot\boldsymbol{w})+b]y_i-1\}$$

We have

(1) For the optimal hyperplane, $\alpha_i^0$ must meet

$$\sum_{i=1}^{l}\alpha_i^0 y_i=0,\quad \alpha_i^0\geq 0,\quad i=1,\cdots,l$$

(2) and $\boldsymbol{w}_0$ must be the linear combination of training samples as:

$$\boldsymbol{w}_0=\sum_{i=1}^{l}y_i\alpha_i^0\boldsymbol{x}_i\,,\quad \alpha_i^0\geq 0,\quad i=1,\cdots,l$$



$$\min_{\boldsymbol{w},b}\max_{\alpha} L(\boldsymbol{w},b,\alpha)=\frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{w})-\sum_{i=1}^{l}\alpha_i\{[(\boldsymbol{x}_i\cdot\boldsymbol{w})+b]y_i-1\}$$

(3) Only the support vectors have non-zero coefficients $\alpha_i^0$ in $\boldsymbol{w}_0$.

This is because, according to the Kühn-Tucker theorem (Karush-Kühn-Tucker (KKT) conditions), at the saddle point, the following equalities hold:

$$\alpha_i^0\{[(\boldsymbol{x}_i\cdot\boldsymbol{w}_0)+b_0]y_i-1\}=0,\quad i=1,\cdots,l$$

Consider $y_i[(\boldsymbol{x}_i\cdot\boldsymbol{w})+b]\geq 1,\ i=1,2,\cdots,l$, $\alpha_i\neq 0$ only for cases where $y_i[(\boldsymbol{x}_i\cdot\boldsymbol{w})+b]=1$, which we call "support vectors" or SVs.

- The solution:

$$\boldsymbol{w}_0=\sum_{\mathrm{SV}s}y_i\alpha_i^0\boldsymbol{x}_i\,,\quad \alpha_i^0\geq 0$$

Substituting $\boldsymbol{w}_0 = \sum_{i=1}^{l} y_i \alpha_i^0 \boldsymbol{x}_i$ into the Lagrangran, we get

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j)$$

- ***The dual problem*** of the optimal hyperplane

$$\max_{\boldsymbol{\alpha}} \quad W(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j)$$

s.t

$$\alpha_i \geq 0, \quad i = 1, \cdots, l$$

and $\sum_{i=1}^{l} \alpha_i y_i = 0$

- The original problem of is called the *Primal Problem*.

min $\Phi(\boldsymbol{w}) = \frac{1}{2}(\boldsymbol{w} \cdot \boldsymbol{w})$ *w.r.t.* $\boldsymbol{w}$

*s.t.* $y_i[(\mathbf{x}_i \cdot \mathbf{w}) + b] \geq 1, \quad i = 1,2,\cdots,l$

for training samples $(y_1, \mathbf{x}_1), \cdots, (y_l, \mathbf{x}_l),\ y \in \{-1,1\}$



At the solution $\alpha_0 = (\alpha_1^0, \cdots, \alpha_l^0)$, we have

$$\|\boldsymbol{w}_0\|^2 = 2W(\alpha_0) = \sum_{\mathrm{SVs}} \alpha_i^0 \alpha_j^0 (\boldsymbol{x}_i \cdot \boldsymbol{x}_j) y_i y_j = \sum_{i=1}^{l} \alpha_i^0$$

and the decision function is

$$f(\boldsymbol{x}) = \mathrm{sign}\left(\sum_{\mathrm{SVs}} y_i \alpha_i^0 (\boldsymbol{x}_i \cdot \boldsymbol{x}) + b_0\right)$$

The threshold $b_0$ can be obtained from support vectors of the two classes:

$$b_0 = \frac{1}{2}[(\boldsymbol{w}_0 \cdot \boldsymbol{x}^*(1)) + (\boldsymbol{w}_0 \cdot \boldsymbol{x}^*(-1))]$$

## 1-minute break

Any questions?

D



## Questions

- Why can we call it “optimal”?
- What if the samples are not separable?
- Can we build optimal nonlinear machines like this?

# 6.2 Generalized Optimal Hyperplane



## Non-separable Cases

For *l* samples of two classes: $(y_1, \boldsymbol{x}_1), \cdots, (y_l, \boldsymbol{x}_l),\ y \in \{-1,1\}$

$$\min\ \Phi(\boldsymbol{w}) = \frac{1}{2}(\boldsymbol{w} \cdot \boldsymbol{w}) \quad \textit{w.r.t. } \boldsymbol{w}$$

$$\text{s.t.} \qquad y_i[(\boldsymbol{x}_i \cdot \boldsymbol{w}) + b] \geq 1, \quad i = 1,2,\cdots,l$$

- Not linearly separable ←→ the inequalities cannot be met by all samples
- Introducing a slack variable $\xi_i \geq 0$, we define new constraints

$$y_i((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) \geq 1 - \xi_i, \quad i = 1,2,\cdots,l$$

- Function $F_\sigma(\xi) = \sum_{i=1}^{l} \xi_i^\sigma,\ \sigma > 0$ can reflect the severity the original constraints are violated

## Soft-margin Optimal Hyperplane (Generalized optimal hyperplane)

min $\quad \Phi(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + C\left(\sum_{i=1}^{l} \xi_i\right)$ w.r.t. $\boldsymbol{w}$

s.t. $\quad y_i((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) \geq 1 - \xi_i, \quad i = 1,2,\cdots,l$

where parameter $C$ controls the penalty on errors.

D

The solution can be obtained with the same Lagrange optimization technology.

H



**The dual problem of the generalized optimal hyperplane**

Weights of the generalized optimal hyperplane are

$$\boldsymbol{w} = \sum_{i=1}^{l} \alpha_i y_i \boldsymbol{x}_i$$

where parameters $\alpha_i, i = 1,\cdots,l$ are the solution of the quadratic optimization problem

$$\max_{\alpha} \; W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

s.t.

$$\sum_{i=1}^{l} y_i \alpha_i = 0 ,$$

$$0 \leq \alpha_i \leq C, \quad i = 1,\cdots,l$$

H

Similar to the linearly-separable cases, at the solution, we have

$$\alpha_i^0\{[(\boldsymbol{x}_i \cdot \boldsymbol{w}_0) + b_0]y_i - 1 + \xi_i\} = 0, \quad i = 1, \cdots, l$$

Only some $\alpha_i$ are non-zero. They corresponds to those that the equality holds in

$$y_i((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) \geq 1 - \xi_i, \quad i = 1,2, \cdots, l$$

- The SVs also include the wrongly classified samples and their $\alpha_i = C$.



## Summary on the generalized optimal hyperplane

**The Primal Problem:**

$$\min \quad \psi(\boldsymbol{w}, \boldsymbol{\xi}) = \frac{1}{2}(\boldsymbol{w} \cdot \boldsymbol{w}) + C\left(\sum_{i=1}^{n} \xi_i\right)$$

$$\text{s.t.} \quad y_i[(\boldsymbol{w} \cdot \boldsymbol{x}_i) + b] - 1 + \xi_i \geq 0, \quad \xi_i \geq 0, \quad i = 1, \cdots, l$$

**The Dual Problem:**

$$\max_{\alpha} \quad W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j)$$

s.t.

$$\sum_{i=1}^{l} y_i \alpha_i = 0,$$

and

$$0 \leq \alpha_i \leq C, \quad i = 1, \cdots, l$$

**The Solution:**

$$f(\mathbf{x}) = \operatorname{sgn}\left\{\sum_{i=1}^{n} \alpha_i^* y_i (\boldsymbol{x}_i \cdot \boldsymbol{x}) + b^*\right\}$$

单选题 1分

设置

# 5-min break

A Yes, I’m back to my computer.

B Sorry, not yet.

Xuegong Zhang

提交

25

# Questions

- Why can we call it “optimal”?
- What if the samples are not separable?
- Can we build optimal nonlinear machines like this?

Xuegong Zhang

26

# 6.3
# Nonlinear Classification via the Kernel Trick

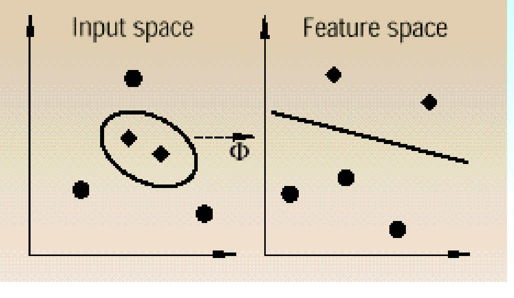


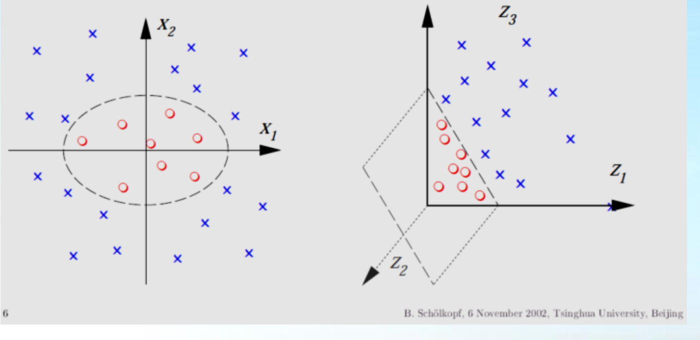
## Generalization to nonlinear machines

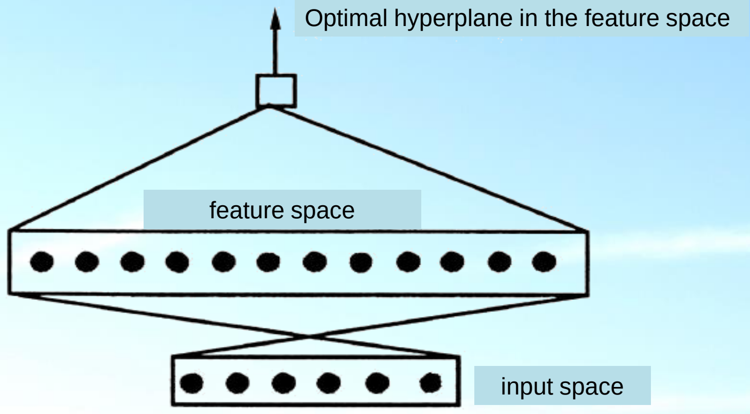
Generalized linear discriminant

$g(x) = c_0 + c_1 x + c_2 x^2 \xrightarrow{z = [1 \quad x \quad x^2]^T} g(\boldsymbol{z}) = \boldsymbol{w} \cdot \boldsymbol{z}$

Nonlinear transform $\boldsymbol{z} = \Phi(\boldsymbol{x})$






Idea:

- Mapping the original input space into a new feature space by nonlinear transformation, and finding the optimal hyperplane in the feature space



For example:

For d-dimensional samples, to construct a 2nd-order polynomial decision function, we can transform the original $X \subset R^{\mathrm{d}}$ space to a feature space $Z \subset R^{D}$ with $N$=$d$($d$+3)/2 coordinates:

$$
\begin{array}{ll}
z^1 = x^1, \cdots, z^d = x^d & d \text{ coordinates,} \\
z^{d+1} = (x^1)^2, \cdots, z^{2d} = (x^d)^2 & d \text{ coordinates,} \\
z^{2d+1} = x^1 x^2, \cdots, z^D = x^d x^{d-1} & d(d\text{-}1)/2 \text{ coordinates.}
\end{array}
$$

The linear discriminant function constructed in the feature space is a quadratic discriminant function in the original space.

**Issues:**

- Complicated transformation
- Dimensionality increasing

What problems we may meet with this approach?

**(1) Can we handle the computation in feature space?**

(technical issue)

- e.g, a 4th or 5th order polynomial function on 200d data (like 16x16 pixels) needs feature spaces of ~billion dimensions.

(curse of dimensionality)

**(2) Will the "optimal hyperplane" still be optimal or good in the feature space?**

(conceptual issue)

- Can it still work or can it generalize?

Let's leave it for future classes.



The Algorithm:

$$\min \quad \psi(\boldsymbol{w},\boldsymbol{\xi}) = \frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{w}) + C\left(\sum_{i=1}^{n}\xi_i\right)$$

s.t. $\quad y_i[(\boldsymbol{w}\cdot\boldsymbol{x}_i)+b]-1+\xi_i \geq 0, \quad \xi_i \geq 0, \quad i=1,\cdots,l$

$$\max_{\alpha} \quad W(\alpha) = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j y_i y_j(\boldsymbol{x}_i\cdot\boldsymbol{x}_j)$$

s.t.

$$\sum_{i=1}^{l} y_i\alpha_i = 0\,,$$

and

$$0 \leq \alpha_i \leq C, \quad i=1,\cdots,l$$

The Solution:

$$f(\mathbf{x}) = \operatorname{sgn}\left\{\sum_{i=1}^{n}\alpha_i^* y_i(\boldsymbol{x}_i\cdot\boldsymbol{x}) + b^*\right\}$$

$\boldsymbol{z} = \Phi(\boldsymbol{x})$

Look carefully

Only the inner-products of the vectors are involved!

$$\max_{\alpha} \quad Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

$$f(\mathbf{x}) = \operatorname{sgn}\left\{ \sum_{i=1}^{l} \alpha_i^* y_i (\mathbf{x}_i \cdot \mathbf{x}) + b^* \right\}$$

$$\boldsymbol{x} \rightarrow \Phi(\boldsymbol{x})$$

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j))$$

Actually we can directly define the kernel without defining the transformation, like:

$$\Phi(\mathbf{x}) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2) \longleftrightarrow$$

$$(\mathbf{x} \cdot \mathbf{y})^2 = \left( \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \right)^2$$

$$= \left( \begin{pmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{pmatrix} \cdot \begin{pmatrix} y_1^2 \\ \sqrt{2} y_1 y_2 \\ y_2^2 \end{pmatrix} \right)$$

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



# Support Vector Machine (SVM)

The Dual Problem

$$\max_{\alpha} \quad Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

$$s.t. \quad \sum_{i=1}^{l} y_i \alpha_i = 0, \quad 0 \le \alpha_i \le C, \quad i = 1, \cdots, l$$

Decision Function:

$$f(\boldsymbol{x}) = \operatorname{sgn}\left( \sum_{i=1}^{l} \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b^* \right)$$

## The Kernel Trick

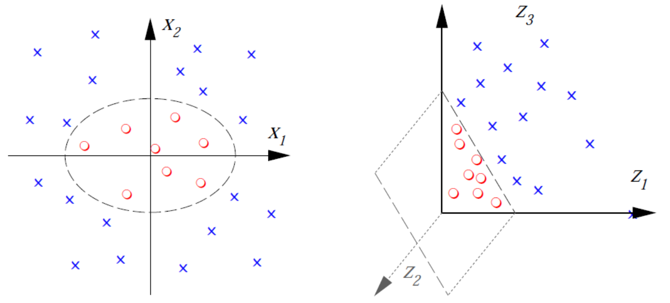
*(Boser, Guyon, and Vapnik, 1992)*

$$\begin{aligned}\langle \Phi(x), \Phi(y)\rangle &= (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)(y_1^2, \sqrt{2}\, y_1 y_2, y_2^2)^\top \\ &= \langle x, y\rangle^2 \\ &=: k(x, y)\end{aligned}$$

$\longrightarrow$ the dot product in $F$ can be computed in $\mathbb{R}^2$

$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$$

$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$







## The Kernel Trick

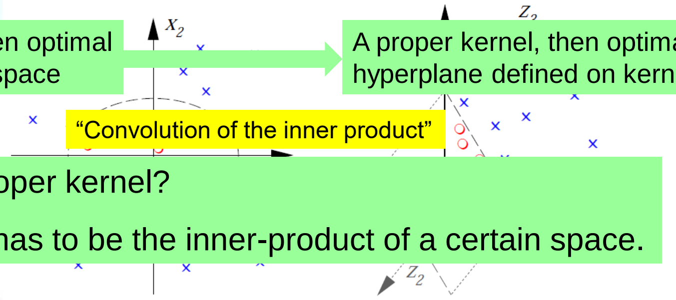
*(Boser, Guyon, and Vapnik, 1992)*

$$\begin{aligned}\langle \Phi(x), \Phi(y)\rangle &= (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)(y_1^2, \sqrt{2}\, y_1 y_2, y_2^2)^\top \\ &= \langle x, y\rangle^2 \\ &=: k(x, y)\end{aligned}$$

$\longrightarrow$ the dot product in $F$ can be computed in $\mathbb{R}^2$

$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$$

$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$

A proper transform, then optimal hyperplane in feature space

A proper kernel, then optimal hyperplane defined on kernels

"Convolution of the inner product"

What is a proper kernel?

—— It has to be the inner-product of a certain space.

According to the Hilbert-Schmidt theory, the inner product in a Hilbert space has an equivalent representation

$$(z_1, z_2) = \sum_{r=1}^{\infty} a_r z_r(x_1) z_r(x_2) \Leftrightarrow K(x_1, x_2),\ \ a_r \geq 0$$

## Mercer Theorem

To guarantee that a continuous symmetric function $K(u,v)$ in $L_2(C)$ has an expansion

$$K(u,v) = \sum_{k=1}^{\infty} a_k \psi_k(u) \psi_k(v)$$

with positive coefficients $a_k > 0$ (i.e., $K(u,v)$ describes an inner product in some feature space), it is necessary and sufficient that the condition

$$\int K(u,v) g(u) g(v) du dv > 0$$

be valid for all $g \in L_2(C)$ ($C$ being a compact subset of $R^d$).

R



## Mercer's Theorem

*If $k$ is a continuous kernel of a positive definite integral operator on $L_2(\mathcal{X})$ (where $\mathcal{X}$ is some compact space),*

$$\int_{\mathcal{X}} k(x, x') f(x) f(x')\, dx\, dx' \geq 0,$$

*it can be expanded as*

$$k(x, x') = \sum_{i=1}^{\infty} \lambda_i \psi_i(x) \psi_i(x')$$

*using eigenfunctions $\psi_i$ and eigenvalues $\lambda_i \geq 0$ [34].*

In that case

$$\Phi(x) := \begin{pmatrix} \sqrt{\lambda_1}\psi_1(x) \\ \sqrt{\lambda_2}\psi_2(x) \\ \vdots \end{pmatrix}$$

satisfies $\langle \Phi(x), \Phi(x') \rangle = k(x, x')$.

B. Schölkopf, NIPS, 3 December 2001

R

## Positive Definite Kernels

Let $\mathcal{X}$ be a nonempty set. The following two are equivalent:

- $k$ is *positive definite (pd)*, i.e., $k$ is symmetric, and for
  - any set of training points $x_1, \ldots, x_m \in \mathcal{X}$ and
  - any $a_1, \ldots, a_m \in \mathbb{R}$

  we have

$$\sum_{i,j} a_i a_j K_{ij} \geq 0, \quad \text{where } K_{ij} := k(x_i, x_j)$$

- there exists a map $\Phi$ into a dot product space $\mathcal{H}$ such that

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle$$

$\mathcal{H}$ is a so-called *reproducing kernel Hilbert space*.

Special case of positive definite kernels: "Mercer kernels"

R



- Got lost?
- It's ok. Let's skip it and just be assured that there is a theorem to safeguard the choosing of kernels.

D

Given training data $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)$, choose some kernel $K(\cdot,\cdot)$ and solve the dual problem:

$$\max_{\alpha} \quad Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

s.t. $\quad \sum_{i=1}^{n} y_i \alpha_i = 0$

and $\quad 0 \leq \alpha_i \leq C, \quad i = 1, \cdots, l$

to get solution $f(\boldsymbol{x}) = \mathrm{sgn}(\sum_{i=1}^{l} \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b^*)$

# Most commonly used kernels

- Inner product: $K(\boldsymbol{x}, \boldsymbol{x}_i) = (\boldsymbol{x} \cdot \boldsymbol{x}_i)$ --- Linear SVM
- Polynomial: $K(\boldsymbol{x}, \boldsymbol{x}_i) = [(\boldsymbol{x} \cdot \boldsymbol{x}_i) + 1]^q$

  --- SVM version of polynomial discriminant
- Radial Basis Function (RBF):

$$K(\boldsymbol{x}, \boldsymbol{x}_i) = \exp\left\{-\frac{|\boldsymbol{x} - \boldsymbol{x}_i|^2}{\sigma^2}\right\}$$

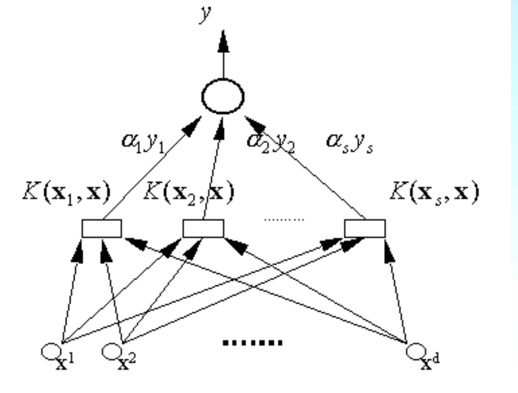
  --- SVM version of RBF Network
- Sigmoid: $K(\boldsymbol{x}, \boldsymbol{x}_i) = \tanh(v(\boldsymbol{x} \cdot \boldsymbol{x}_i) + c)$

  --- SVM version of 3-layered MLP (one hidden layer)
  (with certain constrains on the choice of $v$ and $c$)



# The "Support Vector Network"

Ref. $g_k(\mathbf{x}) \equiv y_k = f\left(\sum_j w_{jk} \boxed{f\left(\sum_i w_{ij} x_i + w_{j0}\right)} + w_{k0}\right)$



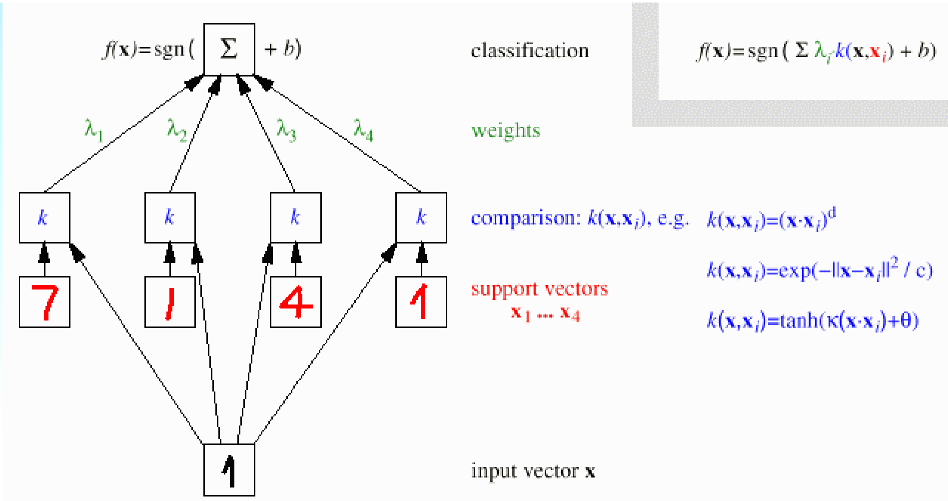

Output (decision function)

$$y = \text{sign}\left(\sum_{i=1}^{s} \alpha_i y_i \boxed{K(\boldsymbol{x}_i, \boldsymbol{x})} + b\right)$$

weights $w_i = \alpha_i y_i$

inner-products with $s$ support vectors $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_s$

Input vector $\boldsymbol{x} = (x^1, x^2, \cdots, x^d)$

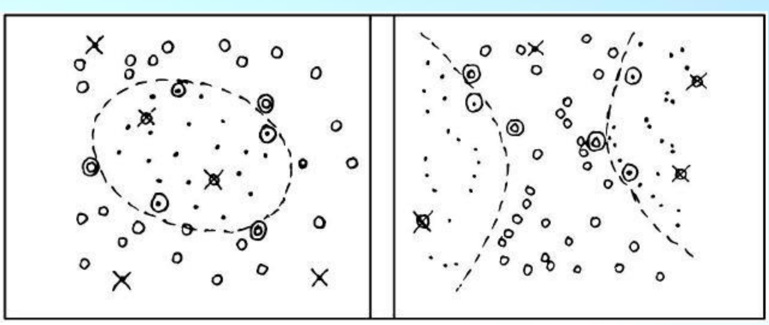


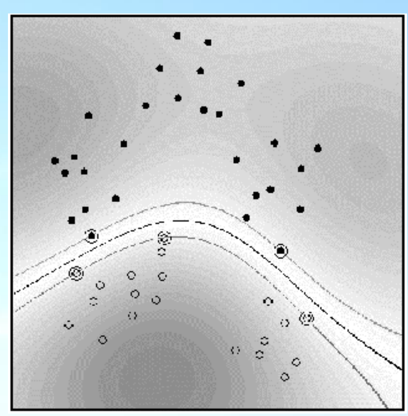

An intuitive understanding

Templates → Similarity → Weighted Voting

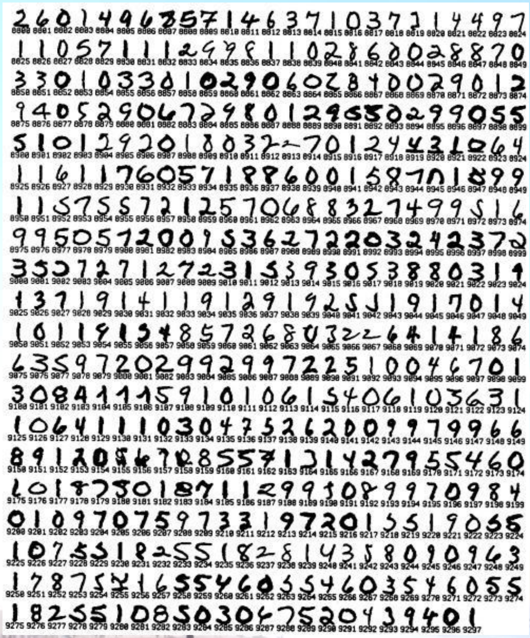
Distance/correlation… → Kernel





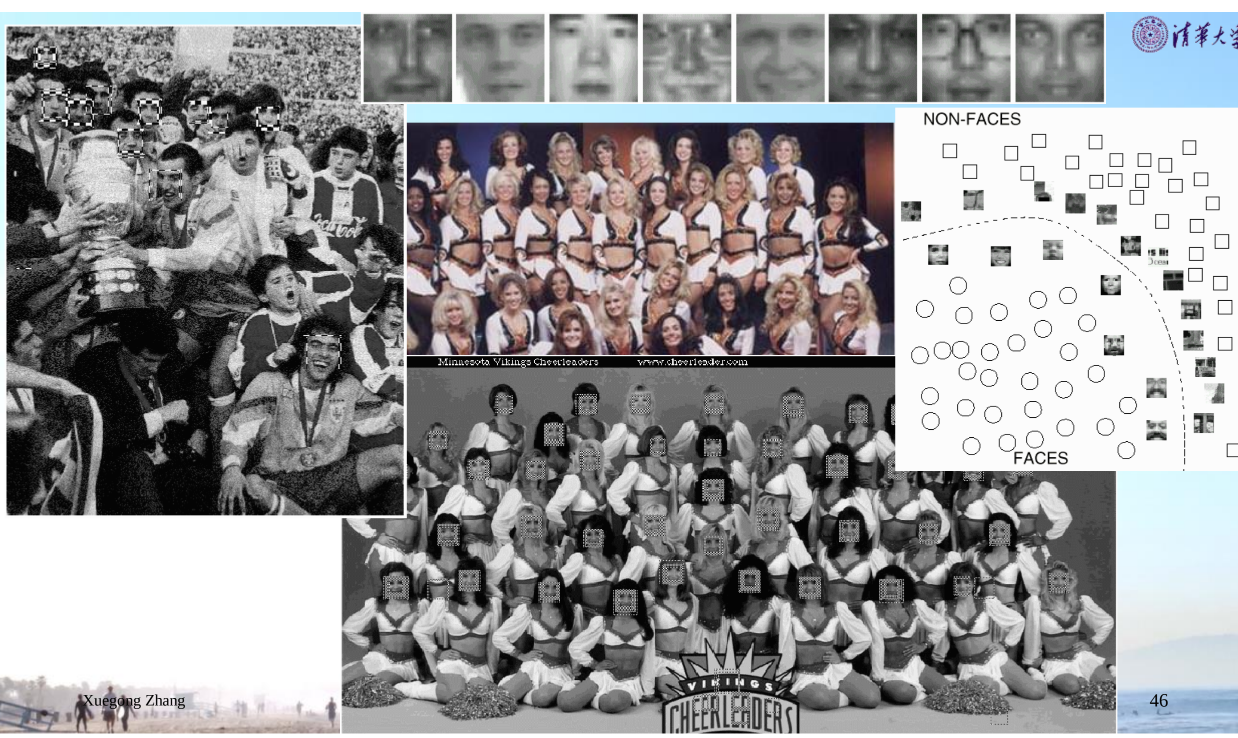


## Former Bell Lab's experiments on USPS data



| Method | Test Error |
|---|---|
| Human | 2.5% |
| Decision Tree | 16.2% |
| 2-layer MLP | 5.9% |
| 5-layer MLP | 5.1% |
| SVM with 3 types of kernels | 4.0%<br>4.1%<br>4.1% |





# 5-second break



# 6.4
# SVM Training Algorithms

### Basic concepts of ML: Perceptron

- How can we make a learning machine?
  - It needs a teacher.
    - → The model: $y = \mathrm{sgn}(\sum_{i=1}^{d} w_i x_i + w_0)$
  - We need materials to train it. / It needs materials to learn from.
    - → Training data: $\{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$, $\boldsymbol{x}_j \in R^{d+1}, y_j \in \{-1,1\}$
  - We need to tell what is the goal of the learning.
    - → Objective function: $\min J_P(\boldsymbol{\alpha}) = \sum_{\mathbf{y}_j \in \mathcal{Y}^k} (-\boldsymbol{\alpha}^T \mathbf{y}_j)$
  - We need to tell it how to learn.
    - → Learning algorithm: $\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}(k) - \rho_k \nabla J = \boldsymbol{\alpha}(k) + \rho_k \sum_{\mathbf{y}_j \in Y^k} \mathbf{y}_j$

### Basic concepts of ML: Logistic Regression

- How can we make a learning machine?
  - It needs a teacher.
    - → The model: $h(\boldsymbol{x}) = \theta(\boldsymbol{w}^T \boldsymbol{x})$
  - We need materials to train it. / It needs materials to learn from.
    - → Training data: $\{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$, $\boldsymbol{x}_j \in R^{d+1}, y_j \in \{-1,1\}$
  - We need to tell what is the goal of the learning.
    - → Objective function: $\min E(\boldsymbol{w}) = \frac{1}{N} \sum_{j=1}^{N} \ln\left(1 + e^{-y_j \boldsymbol{w}^T \boldsymbol{x}_j}\right)$
  - We need to tell it how to learn.
    - → Learning algorithm: $\boldsymbol{w}(k+1) = \boldsymbol{w}(k) - \rho_k \nabla E$

### Basic concepts of ML: Linear Regression

- How can we make a learning machine?
  - It needs a teacher.
    - → The model: $f(\boldsymbol{x}) = \sum_{i=0}^{d} w_i x_i = \boldsymbol{w}^T \boldsymbol{x}$
  - We need materials to train it. / It needs materials to learn from.
    - → Training data: $\{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$, $\boldsymbol{x}_j \in R^{d+1}, y_j \in R$
  - We need to tell what is the goal of the learning.
    - → Objective function: $\min E = \frac{1}{N} \sum_{j=1}^{N} \left(f(\boldsymbol{x}_j) - y_j\right)^2$
  - We need to tell it how to learn.
    - → Learning algorithm: $\boldsymbol{w}(k+1) = \boldsymbol{w}(k) - \rho_k \nabla E$

How to train a SVM?

### Basic concepts of ML: MLP

- How can we make a learning machine?
  - It needs a teacher.
    - → The model:: $g(\mathbf{x}) = f\left(\sum_j w_{jk} f\left(\sum_i w_{ij} x_i + w_{j0}\right) + w_{k0}\right)$
  - We need materials to train it. / It needs materials to learn from.
    - → Training data: $\{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$, $\boldsymbol{x}_j \in R^{d+1}, y_j \in R$
  - We need to tell what is the goal of the learning.
    - → Objective function: $\min E = \frac{1}{2} \sum_{j=1}^{N} \left(g(\boldsymbol{x}_j) - y_j\right)^2$
  - We need to tell it how to learn.
    - → Learning algorithm: $\boldsymbol{w}(k+1) = \boldsymbol{w}(k) - \rho_k \nabla E$ via the BP algorithm



## The optimization problem of SVM

$$f(\mathbf{x}) = \mathrm{sign}\left(\sum y_i \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) - b\right)$$

$$\max_{\alpha} \quad W(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

s.t.

$$\sum_{i=1}^{l} y_i \alpha_i = 0$$

$$0 \le \alpha_i \le C, \quad i = 1, \cdots, l$$

A quadratic programming problem with equality and inequality constraints.

→ Convex Optimization

https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf

# Exterior Point Method for nonlinear programming

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) & \\ \text{s.t.} \quad & g_i(\boldsymbol{x}) \geq 0 & i \in I \\ & h_i(\boldsymbol{x}) = 0 & i \in E \end{aligned}$$

Define a penalty function

$$T(\boldsymbol{x}, M_k) = f(\boldsymbol{x}) + M_k \left( \sum_{i \in E} [h_i(\boldsymbol{x})]^2 \right) + M_k \left( \sum_{i \in I} [\min(0, g_i(\boldsymbol{x}))]^2 \right)$$

- Increasing the penalty factor $M_k$ during optimization
- Using the conjugate gradient method to solve the quadratic programming problem.



# Algorithms for SVM

$$\max_{\alpha} \quad W(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

- The complexity of the problem
  - does not depend on the dimensionality of samples
  - key component: $K(x_i, x_j)$, a $l \times l$ matrix
  - huge requirement on memory and computation for large sample sizes
- Key ideas of solutions
  - Dividing the problem on all samples to sub-problems with fewer samples
- An old-styled resource for SVM
  - www.kernel-machine.org or www.kernel-machines.com

## Some early references on the implementation

- Osuna, E., Freund, R., Girosi, F., Support Vector Machines: Training and Applications. *MIT AI Memo* 1602, March, 1997
- Edgar Osuna et al. Training Support Vector Machines: an Application to Face Detection, *CVPR'97*
- John C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. *Advances in Kernel Methods-Support Vector Learning*. Cambridge, MA: MIT Press,1999, pp.185-208

R



## Early Popular SVM Packages

R

- SVM*light*
  - Thorsten Joachims, Making large-scale SVM learning practical, in B. Schoekopf et al eds. *Advances in Kernel Methods- Support Vector Learning*, MIT Press, 1998
  - http://svmlight.joachims.org/ (version 6.02, 14.08.2008)
  - https://www.cs.cornell.edu/people/tj/svm_light/
- SVMTorch
  - Ronan Collobert & Samy Bengio, SVMTorch: support vector machines for large-scale regression problems, *JMLR*, 1:143-160, 2001
  - http://bengio.abracadoudou.com/SVMTorch.html
  - http://www.torch.ch/
- LibSVM
  - Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1--27:27, 2011
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm (Version 3.23 released on July 15, 2018)

# 5-second break

Xuegong Zhang 55

# 6.5 SVM for Classification

Xuegong Zhang 56

## Support Vector Machine (SVM)

Training samples $(y_1, \boldsymbol{x}_1), \cdots, (y_l, \boldsymbol{x}_l),\ y \in \{-1,1\}$

The Dual Problem

$$\max_{\alpha} \quad Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

$$s.t. \quad \sum_{i=1}^{l} y_i \alpha_i = 0, \quad 0 \le \alpha_i \le C, \quad i = 1, \cdots, l$$

Decision Function:

$$f(\boldsymbol{x}) = \operatorname{sgn}\left( \sum_{i=1}^{l} \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b^* \right)$$



## Questions

- Why can we call it “optimal”?
- What if the samples are not separable?
- Can we build optimal nonlinear machines like this?

# Why largest margin is optimal?

- Generalization: the expected performance of a machine on future samples after being trained on limited samples
  - The difference between the expected risk and empirical risk
- **Statistical Learning Theory**
  - Large margin

    ➔Low VC dimension

    ➔Low complexity

    ➔High generalization ability

$$R(w) \le R_{emp}(w) + \Phi\left(\frac{h}{n}\right)$$

Let's leave it for future classes.

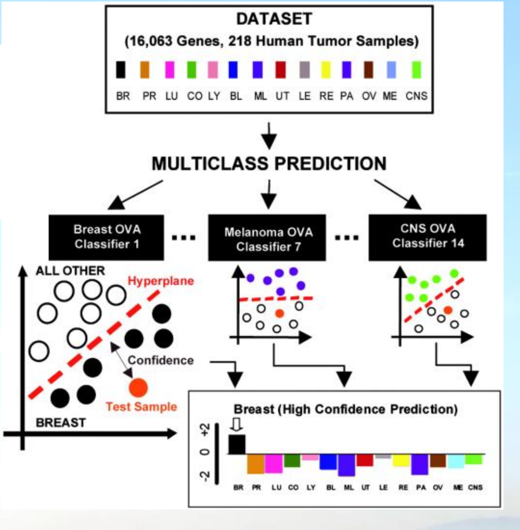


# Example:

SVM in multi-class classification of cancers with microarray data

- 14 tumor classes
- Methods: SVM, Recursive Feature Elimination, etc.
  - they concluded SVM performs the best
- Multi-class problem:

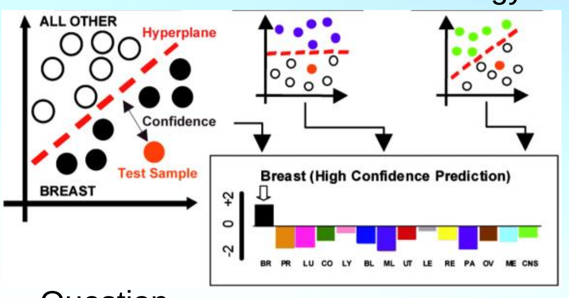
**multiple one-over-all (OVA) binary classifiers**

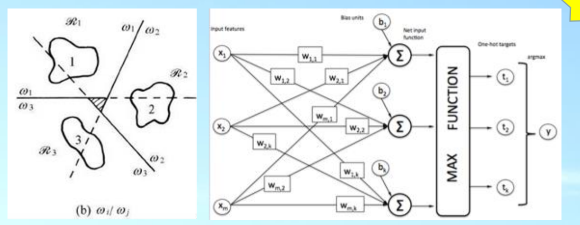



S. Ramaswamy et. al. Multiclass cancer diagnosis using tumor gene expression signatures, *PNAS*, **98**(26): 15149-15154, 2001

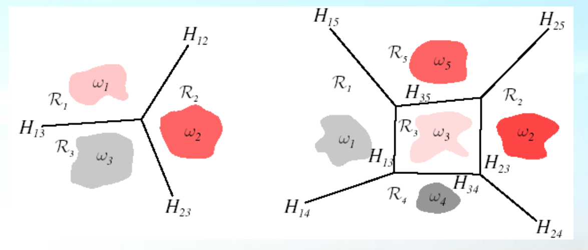
# Multi-class classification

- One-vs-all or one-over-all strategy




Can we build a multi-class SVM?




- Question
  - Since each SVM classifier is built independently, with its own scaling factor, are the output values of different SVM classifiers comparable?



# Multicategory SVM

- Yoonkyung Lee, Yi Lin & Grace Wahba, Multicategory Support Vector Machines, *Technical Report No. 1043, Dept of Statistics, Univ of Wisconsin, Madison*, Sept. 29, 2001
  - slides adopted from Lee's presentations

## SVM in Regularization Framework

Wahba (1998)

- $f(\mathbf{x}) = h(\mathbf{x}) + b$ with $h \in \mathcal{H}_K$ a Reproducing Kernel Hilbert space (RKHS) with reproducing kernel $K$.
- Classification rule : $\phi(\mathbf{x}) = sign(f(\mathbf{x}))$
- SVM scheme : to find $f$ minimizing

$$\frac{1}{n}\sum_{i=1}^{n}(1 - y_i f(\mathbf{x}_i))_+ + \lambda \|h\|^2_{\mathcal{H}_K}$$

$\lambda$ is a tuning parameter which balances the data fit and the complexity of $f(\mathbf{x})$.

## Multicategory SVM

- Class codes :

  $\mathbf{y}_i$ is represented by a $k$-dimensional vector with 1 in the $j$th coordinate and $-\frac{1}{k-1}$ elsewhere if example $i$ falls into class $j$.
  For example, when $k=3$,

$$\mathbf{y}_i = \begin{cases} (1, -\frac{1}{2}, -\frac{1}{2}) & \text{for class 1} \\ (-\frac{1}{2}, 1, -\frac{1}{2}) & \text{for class 2} \\ (-\frac{1}{2}, -\frac{1}{2}, 1) & \text{for class 3} \end{cases}$$

- Separating functions :

  $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_k(\mathbf{x}))$ with sum-to-zero constraint,
  $\sum_{j=1}^{k} f_j(\mathbf{x}) = 0$ for any $\mathbf{x} \in R^d$, and
  $f_j(\mathbf{x}) = h_j(\mathbf{x}) + b_j$ with $h_j \in \mathcal{H}_K$.

- Multicategory SVM formulation :

  Find $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_k(\mathbf{x}))$, with sum-to-zero constraint, minimizing

$$\frac{1}{n}\sum_{i=1}^{n} L(\mathbf{y}_i) \cdot (\mathbf{f}(\mathbf{x}_i) - \mathbf{y}_i)_+ + \frac{\lambda}{2}\sum_{j=1}^{k} \|h_j\|^2_{\mathcal{H}_K}$$

  where $L(\mathbf{y}_i)$ = the $j$th row of the cost matrix $C$ if $\mathbf{y}_i$ indicates class $j$.
  $C$ is defined as $k$ by $k$ matrix with 0 on the diagonal, and 1 elsewhere.
  For example, when $k=3$,

$$C = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

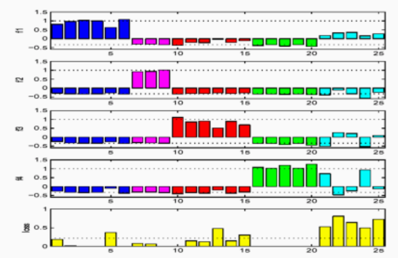


Figure 4: Predicted decision vectors $(f_1, f_2, f_3, f_4)$ at the test samples. EWS: $(1, -1/3, -1/3, -1/3)$, BL: $(-1/3, 1, -1/3, -1/3)$, NB: $(-1/3, -1/3, 1, -1/3)$, and RMS: $(-1/3, -1/3, -1/3, 1)$. The colors indicate the true class identities of the test samples.

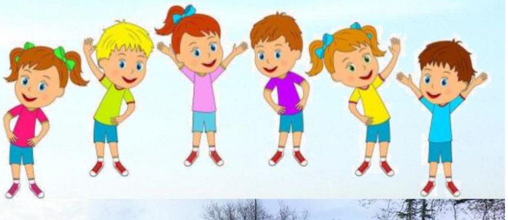
Xuegong Zhang 63

单选题 1分 设置

# 5-min break

A Yes, I’m back to my computer.

B Sorry, not yet.

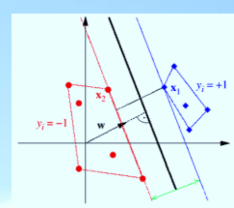


Xuegong Zhang 提交 64

# 6.6
# SVM for Regression
# (Support Vector Regression)



## The primal problem of linear SVR



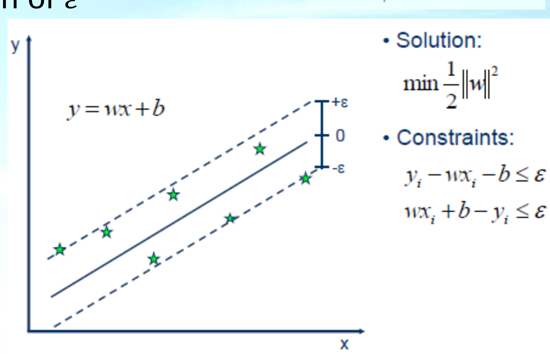

- The task
  - Using function $f(\mathbf{x}) = \boldsymbol{w} \cdot \boldsymbol{x} + b$
    to fit samples $\{\boldsymbol{x}_i, y_i\}$, $\boldsymbol{x}_i \in R^d, y_i \in R, i = 1, \cdots, l$
- If all samples that can be fit within precision of $\varepsilon$

$$\begin{cases} y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i - b \leq \varepsilon, \\ \boldsymbol{w} \cdot \boldsymbol{x}_i + b - y_i \leq \varepsilon, \end{cases} \quad i = 1, \cdots, l \ ,$$

minimize

$$\frac{1}{2}\|\boldsymbol{w}\|^2$$

to obtain the optimal regression.




https://www.saedsayad.com/support_vector_machine_reg.htm

## The primal problem of linear SVR

Given samples $\{\boldsymbol{x}_i, y_i\}, \boldsymbol{x}_i \in R^d, y_i \in R, i = 1, \cdots, l$

$$\min \ \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\text{s.t.} \quad \begin{cases} y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i - b \le \varepsilon, \\ \boldsymbol{w} \cdot \boldsymbol{x}_i + b - y_i \le \varepsilon, \end{cases} \quad i = 1, \cdots, l$$

The primal problem of SVM

$$(y_1, x_1), \cdots, (y_l, x_l), y \in \{-1,1\}$$

$$\min \quad \Phi(w) = \frac{1}{2}(w \cdot w)$$

$$\text{s.t.} \quad y_i[(x_i \cdot w) - b] \ge 1, \quad i = 1,2, \cdots, l$$



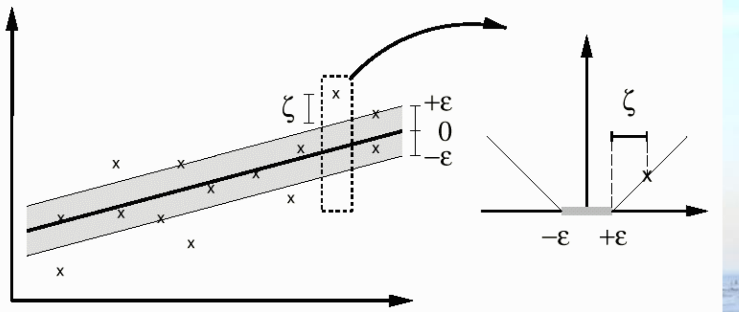
## To allow for errors:

- Introducing slack variables $\xi_i \ge 0, \xi_i^* \ge 0,$

$$\min \ \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

$$\text{s.t.} \quad \begin{cases} y_i - \boldsymbol{w} \cdot \boldsymbol{x}_i - b \le \varepsilon + \xi_i{}^*, \\ \boldsymbol{w} \cdot \boldsymbol{x}_i + b - y_i \le \varepsilon + \xi_i, \end{cases} \quad i = 1, \cdots, l$$

where parameter $C$ controls the penalty to samples that exceed the accuracy threshold $\varepsilon$.

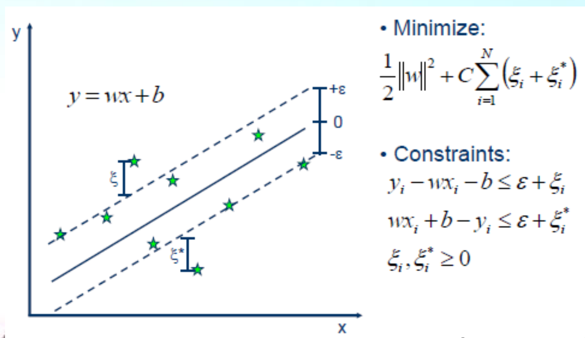


## The Primal Problem of SVR

$$\min \quad \Phi(\boldsymbol{w}, \xi^*, \xi) = \frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{w}) + C\left(\sum_{i=1}^{l}\xi_i^* + \sum_{i=1}^{l}\xi_i\right)$$

s.t.

$$y_i - (\boldsymbol{w}\cdot\boldsymbol{x}_i) - b \le \varepsilon + \xi_i^*, \qquad i = 1,\cdots,l$$
$$(\boldsymbol{w}\cdot\boldsymbol{x}_i) + b - y_i \le \varepsilon + \xi_i, \qquad \mathrm{i} = 1,\cdots,l$$
$$\xi_i^* \ge 0, \qquad \xi_i \ge 0, \qquad i = 1,\cdots,l$$



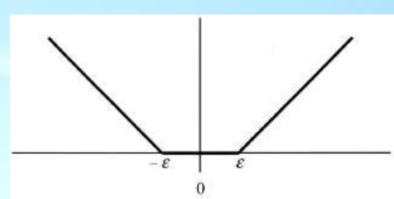

https://www.saedsayad.com/support_vector_machine_reg.htm



## $\varepsilon$-insensitive loss function

$$L(y, f(\boldsymbol{x}, \boldsymbol{\alpha})) = |y - f(\boldsymbol{x}, \boldsymbol{\alpha})|_\varepsilon$$

$$|y - f(\boldsymbol{x}, \boldsymbol{\alpha})|_\varepsilon = \begin{cases} 0, & \text{if } |y - f(\boldsymbol{x}, \boldsymbol{\alpha})| \le \varepsilon \\ |y - f(\boldsymbol{x}, \boldsymbol{\alpha})| - \varepsilon, & \text{else} \end{cases}$$




The objective function of SVR can be re-written as:

$$\min \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C \cdot R_{emp}^{\varepsilon}(f)$$

$$R_{emp}^{\varepsilon}(f) := \frac{1}{l}\sum_{i=1}^{l}|y_i - f(\boldsymbol{x}_i)|_\varepsilon$$

## The Lagrangian

$$L(w,b,\alpha)=\frac{1}{2}(w\cdot w)-\sum_{i=1}^{l}\alpha_i\{[(x_i\cdot w)-b]y_i-1\}$$

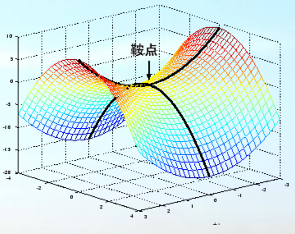
- The solution to the SVR primal problem is the saddle point of the Lagrangian:

$$L(\boldsymbol{w},b,\boldsymbol{\xi}^*,\boldsymbol{\xi}\,;\boldsymbol{\alpha}^*,\boldsymbol{\alpha},\boldsymbol{\gamma},\boldsymbol{\gamma}^*)$$

$$=\frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{w})+C\sum_{i=1}^{l}(\xi_i^*+\xi_i)-\sum_{i=1}^{l}\alpha_i[y_i-(\boldsymbol{w}\cdot\boldsymbol{x}_i)-b+\varepsilon+\xi_i]$$

$$-\sum_{i=1}^{l}\alpha_i^*[(\boldsymbol{w}\cdot\boldsymbol{x}_i)+b-y_i+\varepsilon+\xi_i^*]-\sum_{i=1}^{l}(\gamma_i^*\xi_i^*+\gamma_i\xi_i)$$

- The minimal point w.r.t. $\boldsymbol{w},b,\boldsymbol{\xi}^*,\boldsymbol{\xi}$ and the maximal point w.r.t. $a_i^*\geq 0, a_i\geq 0, \gamma_i^*\geq 0, \gamma_i\geq 0, i=1,\cdots,l$






## The Dual Problem of SVR

Following the same procedure as for SVM classification, we can get

$$\boldsymbol{w}=\sum_{i=1}^{l}(\alpha_i^*-\alpha_i)\boldsymbol{x}_i$$

where $a_i^*, a_i$ are the solution of

$$\max\quad W(\boldsymbol{\alpha},\boldsymbol{\alpha}^*)=-\varepsilon\sum_{i=1}^{l}(\alpha_i^*+\alpha_i)+\sum_{i=1}^{l}y_i(\alpha_i^*-\alpha_i)-\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)(\boldsymbol{x}_i\cdot\boldsymbol{x}_j)$$

s.t.

$$\textstyle\sum_{i=1}^{l}\alpha_i^*=\sum_{i=1}^{l}\alpha_i$$
$$0\leq\alpha_i^*\leq C,\quad i=1,\cdots,l$$
$$0\leq\alpha_i\leq C,\quad i=1,\cdots,l$$

And the regression function is

$$f(\boldsymbol{x})=(\boldsymbol{w}\cdot\boldsymbol{x})+b=\sum_{i=1}^{l}(\alpha_i^*-\alpha_i)(\boldsymbol{x}_i\cdot\boldsymbol{x})+b^*$$

## Support Vectors in SVR

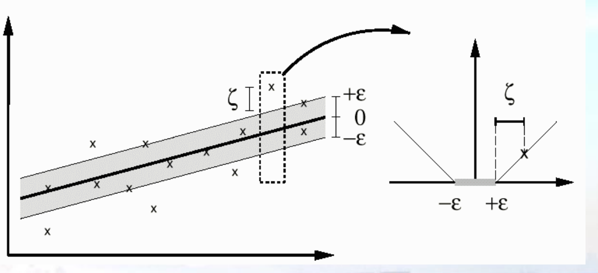
- $\alpha_i^* \alpha_i = 0$ for any sample
- Only samples with $\alpha_i^{(*)} = C$ falls outside the *ε-insensitive tube*
- Samples with $\alpha_i^{(*)} \in (0, C)$ have $\xi_i^{(*)} = 0$ and $|y_i - f(\boldsymbol{x}_i)| = \varepsilon$.

Corresponds to error samples in classification

Corresponds to boundary samples in classification

The $b$ can be obtained on these samples:

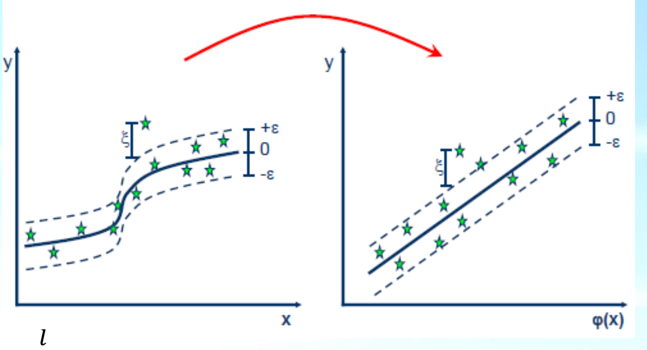
$$b = y_i - < \boldsymbol{w}, \boldsymbol{x}_i > -\varepsilon \qquad \text{for} \quad \alpha_i \in (0, C)$$
$$b = y_i - < \boldsymbol{w}, \boldsymbol{x}_i > +\varepsilon \qquad \text{for} \quad \alpha_i^* \in (0, C)$$






## Nonlinear SVR with the kernel trick

- Construct nonlinear regression with SVM using kernels




$$f(\boldsymbol{x}) = (\boldsymbol{w} \cdot \boldsymbol{x}) + b = \sum_{i=1}^{l} (\alpha_i^* - \alpha_i)(\boldsymbol{x}_i \cdot \boldsymbol{x}) + b^* \Rightarrow f(\boldsymbol{x}; \boldsymbol{v}, \boldsymbol{\beta}) = \sum_{i=1}^{N} \beta_i K(\boldsymbol{x}, \boldsymbol{v}_i) + b$$

https://www.saedsayad.com/support_vector_machine_reg.htm

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} \beta_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b, \quad \beta_i = \alpha_i^* - \alpha_i, \quad i = 1, \cdots, l$$

The dual problem of SVR

$$\max \quad W(\boldsymbol{\alpha}, \boldsymbol{\alpha}^*) = -\varepsilon \sum_{i=1}^{l} (\alpha_i^* + \alpha_i) + \sum_{i=1}^{l} y_i (\alpha_i^* - \alpha_i) - \frac{1}{2} \sum_{i,j=1}^{l} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

s.t.

$$\sum_{i=1}^{l} \alpha_i^* = \sum_{i=1}^{l} \alpha_i$$
$$0 \le \alpha_i^* \le C, \quad i = 1, \cdots, l$$
$$0 \le \alpha_i \le C, \quad i = 1, \cdots, l$$

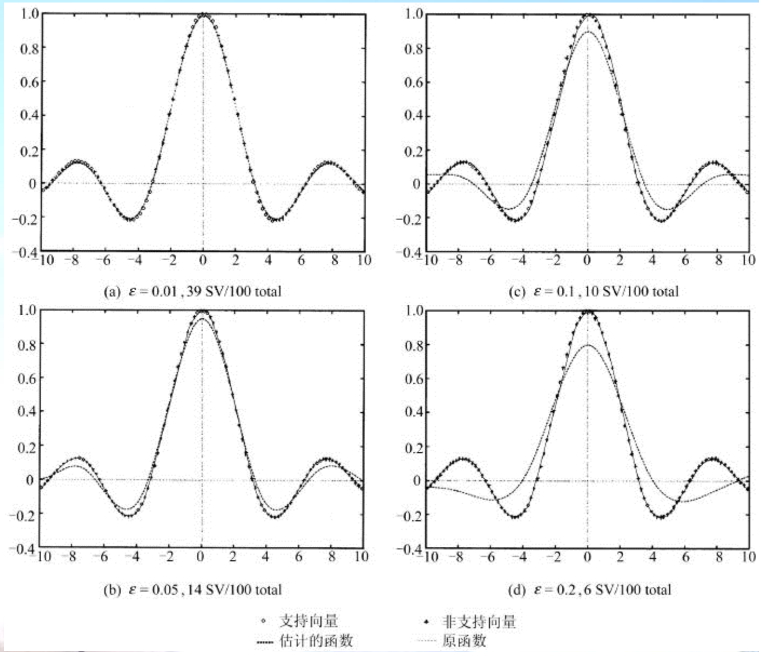


# SVR examples on toy data (1)

- Data from 1D sinc function $f(x) = \frac{\sin(x-10)}{x-10}$
- Regression function $y = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) K_1(x, x_i) + b$
- The kernel $K_1(x, x_i) = 1 + x_i x + \frac{1}{2} |x - x_i| (x \wedge x_i)^2 + \frac{(x \wedge x_i)^3}{3}$

where $(x \wedge x_i) := \min(x, x_i)$

The number of SVs decreases with relaxed requirement on accuracy.




Xuegong Zhang 77

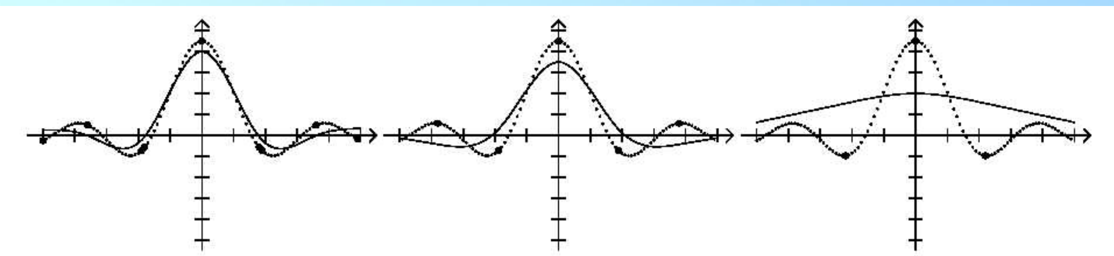


**Figure 9.4** Left to right: regression (solid line), data points (small dots) and SVs (big dots) for an approximation of sinc $x$ (dotted line) with $\varepsilon = 0.1, 0.2$, and $0.5$. Note the decrease in the number of SVs.

Xuegong Zhang 78

# SVR examples on toy data (2)

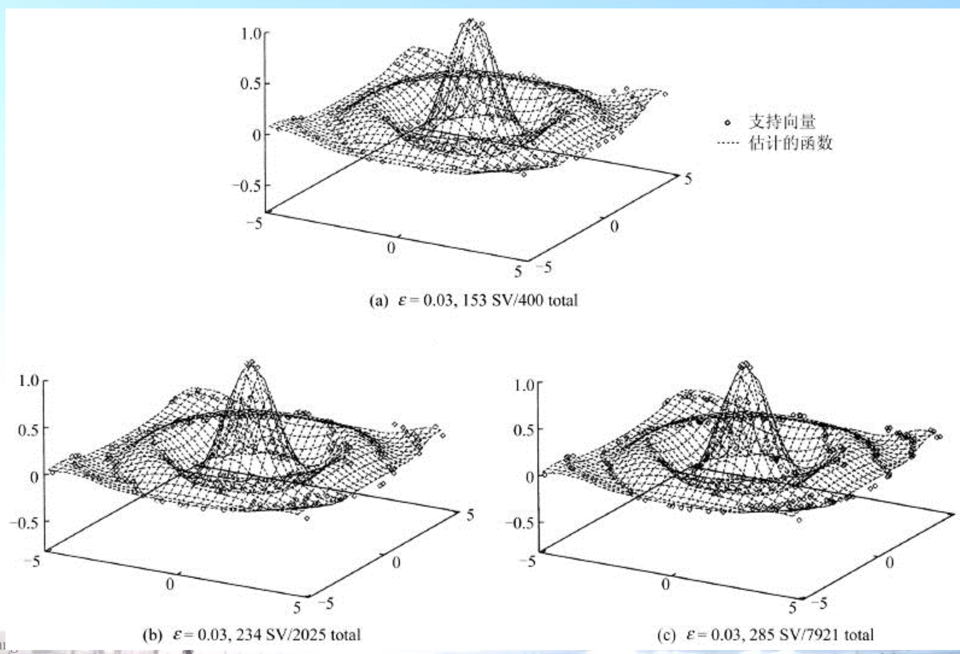
- Data from 2D sinc function $f(x) = \frac{\sin\sqrt{(x-10)^2+(y-10)^2}}{\sqrt{(x-10)^2+(y-10)^2}}$
- Regression function $y = \sum_{i=1}^{N}(\alpha_i^* - \alpha_i)K(x, x_i)\,K(y, y_i) + b$
- The kernel $K(x, y\,; x_i, y_i) = K(x, x_i)K(y, y_i)$

$$= \left(1 + xx_i + \frac{1}{2}|x - x_i|(x \wedge x_i)^2 + \frac{(x \wedge x_i)^3}{3}\right) \times \left(1 + yy_i + \frac{1}{2}|y - y_i|(y \wedge y_i)^2 + \frac{(y \wedge y_i)^3}{3}\right)$$

where $(x \wedge x_i) := \min(x, x_i)$



number of SVs increased <2 fold when number of training samples increased ~20 fold

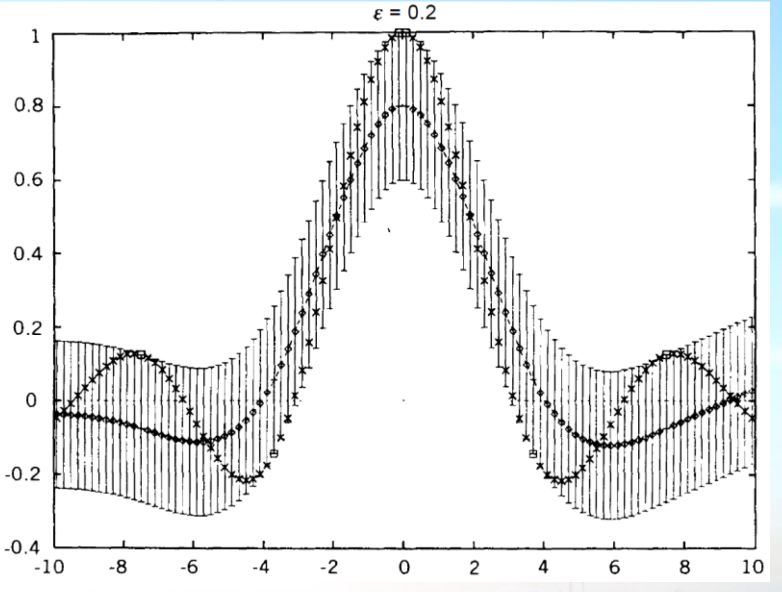



(a) $\varepsilon = 0.03$, 153 SV/400 total

(b) $\varepsilon = 0.03$, 234 SV/2025 total

(c) $\varepsilon = 0.03$, 285 SV/7921 total

## The ε-tube model of function approximation






## 5-second break

# 6.7
# Kernel Machines



## The idea of **kernel machines**

- Any linear method, if it only involves the inner-product of its input vectors, it can be nonlinearized by adopting the kernel trick.
- Large margin is sometimes desirable for good generalization as the kernel trick raises the dimensionality. → "**Large Margin Machines**"
  - **Kernel Fisher's Discriminant (KFD):** S. Mika, G. Ratsch, J. Weston, B. Scholkopf and K.-R. Muller, Fisher discriminant analysis with kernels, *Neural Networks for Signal Processing IX*, pp.41-48, IEEE, 1999
  - **Kernel MSE:** Jianhua Xu, Xuegong Zhang, Yanda Li, Kernel MSE algorithm: a unified framework for KFD, LS-SVM and KRR, *Proceedings of IJCNN'01*, pp.1486-1491
  - **Kernel Pocket:** Jianhua Xu, Xuegong Zhang, Yanda Li, Large margin kernel pocket algorithm, ibid., pp. 1480-1485
  - **Kernel Nearest Neighbor:** Kai Yu, Liang Ji, Xuegong Zhang, Kernel nearest-neighbor algorithm, *Neural Processing Letters*, 15: 147-156, 2002

R

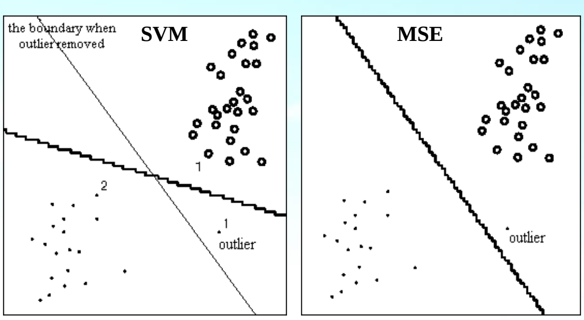
# 6.8
# Variations of SVM

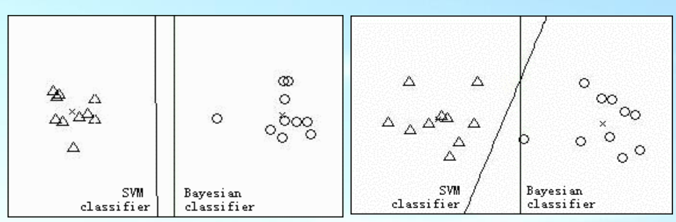


## 6.8.1 Central Support Vector Machines (CSVM)

X. Zhang, Using class-center vectors to build support vector machines, *Neural Networks for Signal Processing*, IX, pp. 3-11, 1999

- Observation: SVM can be sensitive to noise when sample size is very small




Examples of synthetic samples from two normal distributions

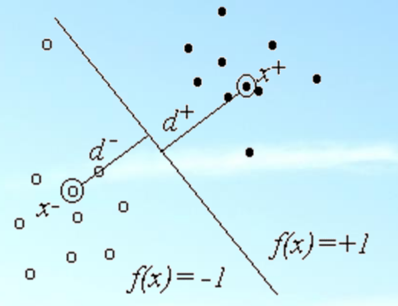
## Central Margin

$$d = d^+ + d^- = \frac{\sum_{i=1}^{n} l_i y_i (\mathbf{w}\cdot\mathbf{x}_i)}{\|\mathbf{w}\|}$$

$$l_i = 1/n^- \quad \text{if} \quad y_i = -1$$

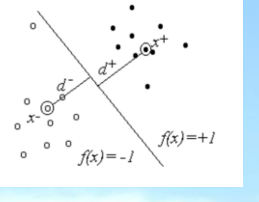
$$l_i = 1/n^+ \quad \text{if} \quad y_i = +1$$

$$\text{normalization: } \sum_{i=1}^{n} l_i y_i (\mathbf{w}\cdot\mathbf{x}_i) = 1$$






## CSVM (Linearly separable case)




min $\quad \psi(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}(\mathbf{w}\cdot\mathbf{w})$

subject to $\quad \sum_{i=1}^{n} l_i y_i (\mathbf{w}\cdot\mathbf{x}_i) = 1$

and $\quad y_i[(\mathbf{w}\cdot\mathbf{x}_i) + b] - \varepsilon \ge 0, \quad i = 1, \cdots, n$

where $\varepsilon \ge 0$ is a constant indicating the least distance that the samples should be away from the separation hyperplane.

Solution of the dual problem :

$$\mathbf{w}^* = \sum_{i=1}^{n} (\alpha_i^* + \beta^* l_i) y_i \mathbf{x}_i = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i + \beta^*(\mathbf{x}^+ - \mathbf{x}^-)$$

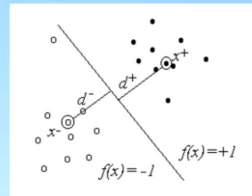
- Intuition: *maximizing the central margin while keeping all the training data not only correctly classified but also be away from the separation hyperplane by at least certain small distance*.

# CSVM (Linearly non-separable case)

$$\min \psi(\mathbf{w}, \xi) = \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + C(\textstyle\sum_{i=1}^{n} \xi_i)$$

subject to $\sum_{i=1}^{n} l_i y_i (\mathbf{w} \cdot \mathbf{x}_i) = 1$

and $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) + \xi_i \geq \varepsilon > 0, \quad \xi_i \geq 0, \quad i = 1, \ldots, n$




where $\varepsilon \geq 0$ is the constant indicating the least distance that the correctly-classified samples should be away from the separation hyperplane.

Solution to the dual problem:

$$\mathbf{w}^* = \sum_{i=1}^{n} (\alpha_i^* + \beta^* l_i) y_i \mathbf{x}_i = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i + \beta^* (\mathbf{x}^+ - \mathbf{x}^-)$$

- Intuition: *maximizing the central margin while keeping smallest possible mistakes (samples that are misclassified or within the boundary zone are regarded as possible mistakes).*

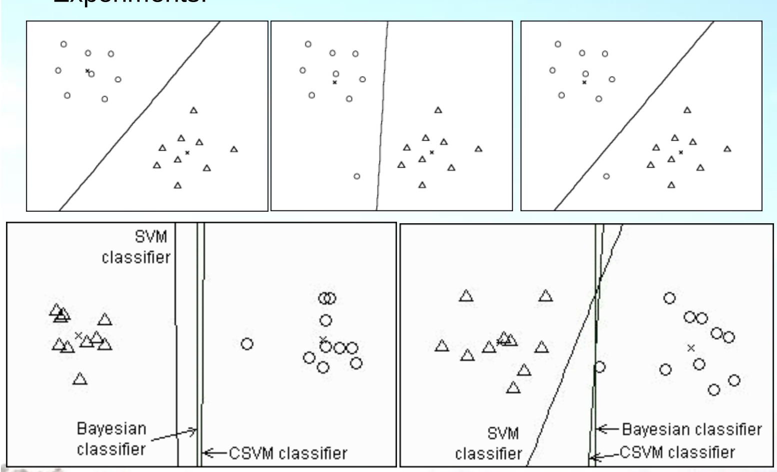


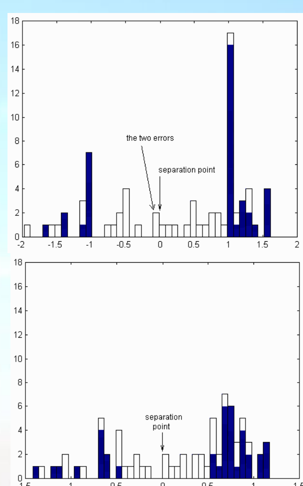
# CSVM implementation

Simplified calculation: $\boldsymbol{w}^{\text{csvm}} = (1-\lambda)\boldsymbol{w}^{\text{svm}} + \lambda(\boldsymbol{x}^+ - \boldsymbol{x}^-)$

Kernel version: $f(\boldsymbol{x}) = \text{sgn}((1-\lambda)\sum_{i=1}^{n} \alpha_i^{\text{svm}} y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + \lambda \sum_{i=1}^{n} l_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b^*)$

Experiments:

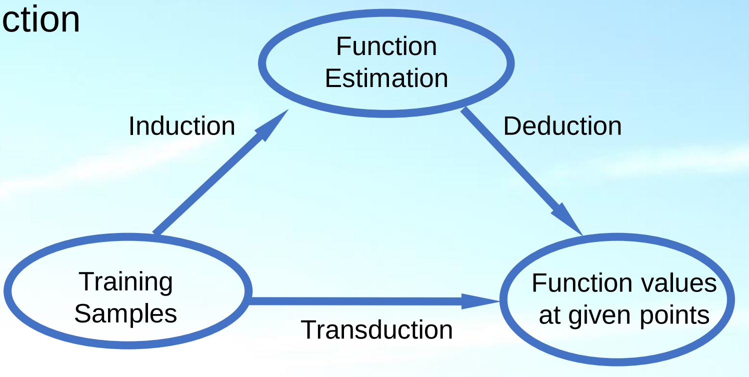



Experiment on a small sample high-dimensional gene expression data

# 6.8.2 Transduction and Semi-supervised SVM

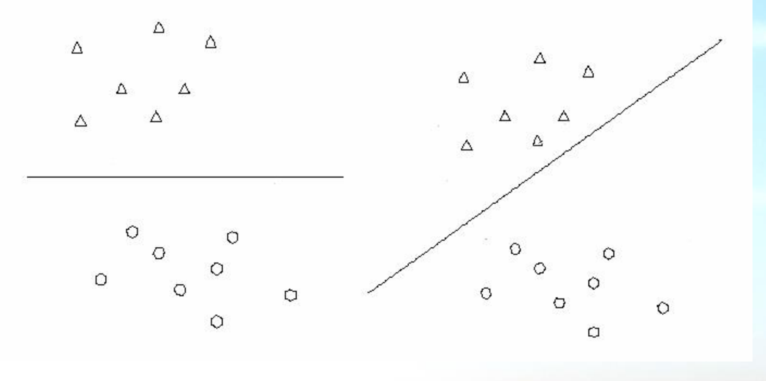
- Transduction






- Semi-supervised SVM ($S^3VM$)
- Idea: to consider the unknown test samples in the training

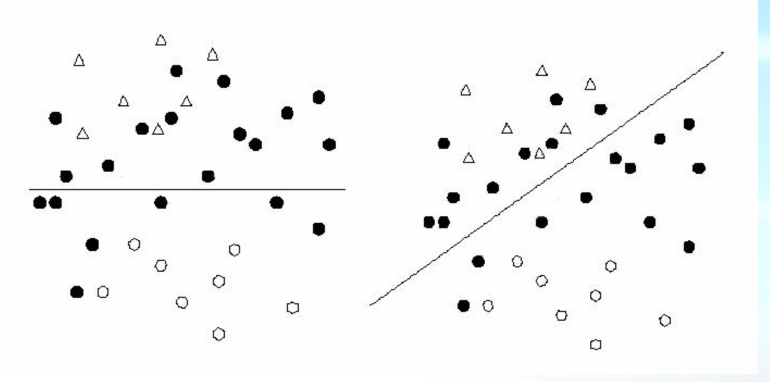
Bennett, K. & Demiriz, A., Semi-supervised support vector machines. in: M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, Advances in Neural Information Processing Systems 11, MIT Press, Cambridge, MA, 1999



- Semi-supervised SVM ($S^3$VM)
- Idea: to consider the unknown test samples in the training

Bennett, K. & Demiriz, A., Semi-supervised support vector machines. in: M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, Advances in Neural Information Processing Systems 11, MIT Press, Cambridge, MA, 1999

D

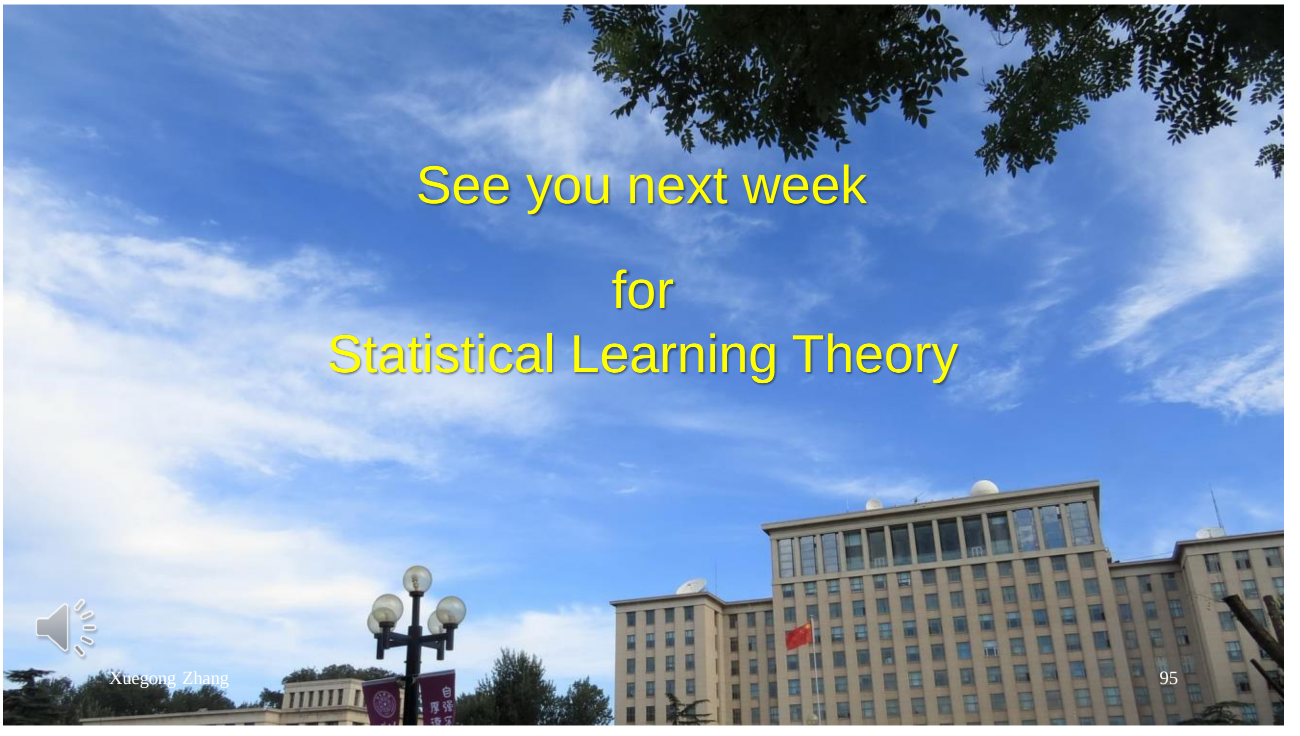




# Homework

- Problems (Pr4)
  1. Derive the SVM dual problem for linearly non-separable cases.
  2. Practice with SVM on a 2-D toy example.
- Deadline:
  - Oct. 20 (Wednesday), 23:00
- Computer exercises (Ex3)
  - Study the SVM package in scikit-learn.
  - Experiment on the medical dataset
- Deadline:
  - Oct. 27 (Wednesday), 23:00

D


See you next week
for
Statistical Learning Theory
Xuegong Zhang
95