Machine Learning
(機器學習)

Lecture 11: Support Vector Machine (2)

Hsuan-Tien Lin (林軒田)
htlin@csie.ntu.edu.tw

Department of Computer Science
& Information Engineering

National Taiwan University
(國立台灣大學資訊工程系)

Roadmap

1. When Can Machines Learn?
2. Why Can Machines Learn?
3. How Can Machines Learn?
4. How Can Machines Learn Better?
5. Embedding Numerous Features: Kernel Models

Lecture 11: Support Vector Machine (2)

- Kernel Trick
- Polynomial Kernel
- Gaussian Kernel
- Comparison of Kernels
- Motivation and Primal Problem
- Dual Problem
- Messages behind Soft-Margin SVM
- Soft-Margin SVM as Regularized Model

Dual SVM Revisited

goal: SVM **without dependence on $\tilde{d}$**

half-way done:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^T \mathrm{Q_D} \boldsymbol{\alpha} - \mathbf{1}^T \boldsymbol{\alpha} \\ \text{subject to} \quad & \mathbf{y}^T \boldsymbol{\alpha} = 0; \\ & \alpha_n \geq 0, \text{for } n = 1, 2, \ldots, N \end{aligned}$$

- $q_{n,m} = y_n y_m \mathbf{z}_n^T \mathbf{z}_m$: inner product in $\mathbb{R}^{\tilde{d}}$
- need: $\mathbf{z}_n^T \mathbf{z}_m = \mathbf{\Phi}(\mathbf{x}_n)^T \mathbf{\Phi}(\mathbf{x}_m)$ calculated faster than $O(\tilde{d})$

can we do so?

Fast Inner Product for Φ_2

2nd order polynomial transform

$$\Phi_2(\mathbf{x}) = (1, x_1, x_2, \ldots, x_d, x_1^2, x_1x_2, \ldots, x_1x_d, x_2x_1, x_2^2, \ldots, x_2x_d, \ldots, x_d^2)$$

—include both x_1x_2 & x_2x_1 for '**simplicity**' **:-)**

$$\begin{aligned}
\Phi_2(\mathbf{x})^T\Phi_2(\mathbf{x}') &= 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d}\sum_{j=1}^{d} x_i x_j x_i' x_j' \\
&= 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d} x_i x_i' \sum_{j=1}^{d} x_j x_j' \\
&= 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')(\mathbf{x}^T\mathbf{x}')
\end{aligned}$$

for Φ_2, transform + inner product can be carefully done in $O(d)$ instead of $O(d^2)$

Kernel: Transform + Inner Product

transform $\boldsymbol{\Phi} \Longleftrightarrow$ **kernel function**: $K_{\boldsymbol{\Phi}}(\mathbf{x}, \mathbf{x}') \equiv \boldsymbol{\Phi}(\mathbf{x})^T \boldsymbol{\Phi}(\mathbf{x}')$

$\boldsymbol{\Phi}_2 \Longleftrightarrow K_{\boldsymbol{\Phi}_2}(\mathbf{x}, \mathbf{x}') = 1 + (\mathbf{x}^T\mathbf{x}') + (\mathbf{x}^T\mathbf{x}')^2$

- quadratic coefficient $q_{n,m} = y_n y_m \mathbf{z}_n^T \mathbf{z}_m = y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$
- optimal bias b? from SV $(\mathbf{x}_s, y_s)$,

$$b = y_s - \mathbf{w}^T \mathbf{z}_s = y_s - \left(\sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n \right)^T \mathbf{z}_s = y_s - \sum_{n=1}^{N} \alpha_n y_n \Big(K(\mathbf{x}_n, \mathbf{x}_s) \Big)$$

- optimal hypothesis g_{SVM}: for test input $\mathbf{x}$,

$$g_{\text{SVM}}(\mathbf{x}) = \operatorname{sign}\left(\mathbf{w}^T \boldsymbol{\Phi}(\mathbf{x}) + b\right) = \operatorname{sign}\left(\sum_{n=1}^{N} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b \right)$$

kernel trick: plug in **efficient kernel function** to avoid dependence on $\tilde{d}$

Kernel SVM with QP

Kernel Hard-Margin SVM Algorithm

1. $q_{n,m} = y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$; $\mathbf{p} = -\mathbf{1}_N$; $(\mathrm{A}, \mathbf{c})$ for equ./bound constraints
2. $\boldsymbol{\alpha} \leftarrow \mathrm{QP}(\mathrm{Q}_\mathrm{D}, \mathbf{p}, \mathrm{A}, \mathbf{c})$
3. $b \leftarrow \left(y_s - \sum_{\text{SV indices } n} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_s) \right)$ with SV $(\mathbf{x}_s, y_s)$
4. return SVs and their α_n as well as b such that for new $\mathbf{x}$,

$$g_{\text{SVM}}(\mathbf{x}) = \text{sign}\left(\sum_{\text{SV indices } n} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b \right)$$

- 1: time complexity $O(N^2) \cdot$ (kernel evaluation)
- 2: QP with N variables and $N+1$ constraints
- 3 & 4: time complexity $O(\#\text{SV}) \cdot$ (kernel evaluation)

kernel SVM:

use computational shortcut to avoid $\tilde{d}$ & predict with SV only

Questions?

General Poly-2 Kernel

$$\Phi_2(\mathbf{x}) = (1, x_1, \ldots, x_d, x_1^2, \ldots, x_d^2) \quad \Leftrightarrow \quad K_{\Phi_2}(\mathbf{x}, \mathbf{x}') = 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$$

$$\Phi_2(\mathbf{x}) = (1, \sqrt{2}x_1, \ldots, \sqrt{2}x_d, x_1^2, \ldots, x_d^2) \quad \Leftrightarrow \quad K_2(\mathbf{x}, \mathbf{x}') = 1 + 2\mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$$

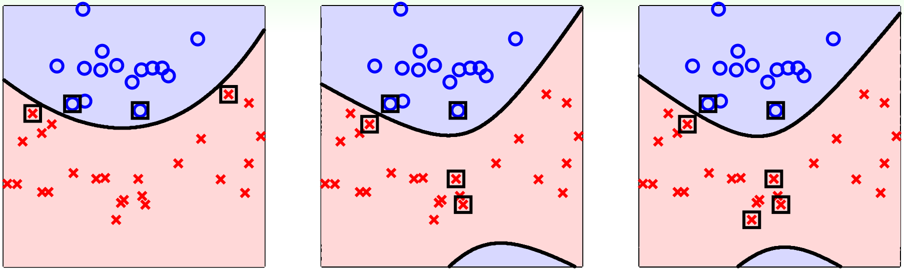
$$\Phi_2(\mathbf{x}) = (1, \sqrt{2\gamma}x_1, \ldots, \sqrt{2\gamma}x_d, \gamma x_1^2, \ldots, \gamma x_d^2)$$
$$\Leftrightarrow K_2(\mathbf{x}, \mathbf{x}') = 1 + 2\gamma\mathbf{x}^T\mathbf{x}' + \gamma^2(\mathbf{x}^T\mathbf{x}')^2$$

$$K_2(\mathbf{x}, \mathbf{x}') = (1 + \gamma\mathbf{x}^T\mathbf{x}')^2 \text{ with } \gamma > 0$$

- K_2: somewhat '**easier**' to calculate than K_{Φ_2}
- Φ_2 and Φ_2: equivalent **power**,
  different inner product $\Longrightarrow$ different **geometry**

K_2 commonly used

Poly-2 Kernels in Action



$(1 + 0.001\mathbf{x}^T\mathbf{x}')^2$ $\quad$ $1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$ $\quad$ $(1 + 1000\mathbf{x}^T\mathbf{x}')^2$

- g_{SVM} **different**, $\boxed{\text{SVs}}$ **different**
  —'hard' to say which is better before learning
- change of kernel $\Leftrightarrow$ change of **margin definition**

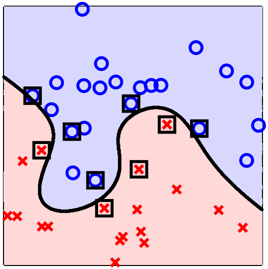
need selecting K, just like selecting $\boldsymbol{\Phi}$

General Polynomial Kernel

$$
\begin{aligned}
K_2(\mathbf{x}, \mathbf{x}') &= (\zeta + \gamma \mathbf{x}^T \mathbf{x}')^2 \text{ with } \gamma > 0, \zeta \geq 0 \\
K_3(\mathbf{x}, \mathbf{x}') &= (\zeta + \gamma \mathbf{x}^T \mathbf{x}')^3 \text{ with } \gamma > 0, \zeta \geq 0 \\
&\vdots \\
K_Q(\mathbf{x}, \mathbf{x}') &= (\zeta + \gamma \mathbf{x}^T \mathbf{x}')^Q \text{ with } \gamma > 0, \zeta \geq 0
\end{aligned}
$$

- embeds $\mathbf{\Phi}_Q$ specially with parameters (γ, ζ)
- allows computing large-margin polynomial classification without dependence on $\tilde{d}$

SVM + Polynomial Kernel: Polynomial SVM



10-th order polynomial with margin 0.1

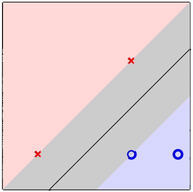
Special Case: Linear Kernel

$$K_1(\mathbf{x}, \mathbf{x}') = (0 + 1 \cdot \mathbf{x}^T\mathbf{x}')^1$$

$$\vdots$$

$$K_Q(\mathbf{x}, \mathbf{x}') = (\zeta + \gamma\mathbf{x}^T\mathbf{x}')^Q \text{ with } \gamma > 0, \zeta \geq 0$$

- K_1: just **usual inner product**, called **linear kernel**
- 'even easier': can be solved (often in primal form) **efficiently**



linear first, remember? :-)

Questions?

Kernel of Infinite Dimensional Transform

infinite dimensional $\boldsymbol{\Phi}(\mathbf{x})$? Yes, if $K(\mathbf{x}, \mathbf{x}')$ **efficiently computable**!

when $\mathbf{x} = (x)$, $K(x, x')$

$$
\begin{aligned}
K(x, x') &= \exp(-(x - x')^2) \\
&= \exp(-(x)^2)\exp(-(x')^2)\exp(2xx') \\
&\overset{\text{Taylor}}{=} \exp(-(x)^2)\exp(-(x')^2)\left(\sum_{i=0}^{\infty}\frac{(2xx')^i}{i!}\right) \\
&= \sum_{i=0}^{\infty}\left(\exp(-(x)^2)\exp(-(x')^2)\sqrt{\frac{2^i}{i!}}\sqrt{\frac{2^i}{i!}}(x)^i(x')^i\right) \\
&= \boldsymbol{\Phi}(x)^T\boldsymbol{\Phi}(x')
\end{aligned}
$$

with infinite dimensional $\boldsymbol{\Phi}(x) = \exp(-x^2) \cdot \left(1, \sqrt{\frac{2}{1!}}x, \sqrt{\frac{2^2}{2!}}x^2, \ldots\right)$

more generally, **Gaussian kernel** $K(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2\right)$ with $\gamma > 0$

Hypothesis of Gaussian SVM

Gaussian kernel $K(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2\right)$

$$
\begin{aligned}
g_{\mathrm{SVM}}(\mathbf{x}) &= \mathrm{sign}\left(\sum_{\mathrm{SV}} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b\right) \\
&= \mathrm{sign}\left(\sum_{\mathrm{SV}} \alpha_n y_n \exp\left(-\gamma\|\mathbf{x} - \mathbf{x}_n\|^2\right) + b\right)
\end{aligned}
$$

- linear combination of Gaussians centered at SVs $\mathbf{x}_n$
- also called Radial Basis Function (RBF) kernel

Gaussian SVM:
find α_n to combine Gaussians centered at $\mathbf{x}_n$
& achieve large margin in infinite-dim. space

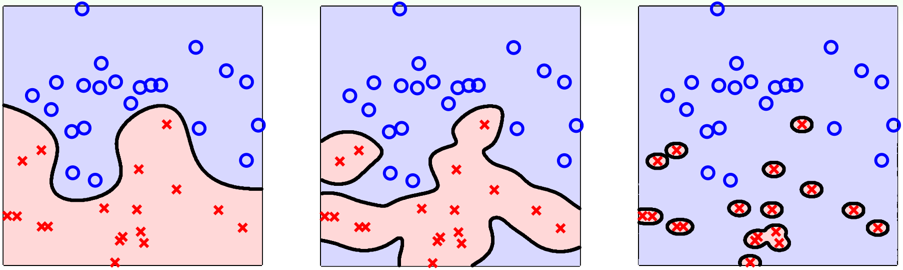
Support Vector Mechanism

	large-margin **hyperplanes** **+ higher-order transforms** with **kernel trick**
#	**not many**
boundary	**sophisticated**

- transformed vector $\mathbf{z} = \mathbf{\Phi}(\mathbf{x}) \Longrightarrow$ efficient kernel $K(\mathbf{x}, \mathbf{x}')$
- store optimal $\mathbf{w} \Longrightarrow$ store a few SVs and α_n

new possibility by Gaussian SVM:
infinite-dimensional linear classification, with generalization 'guarded by' large-margin **:-)**

Gaussian SVM in Action



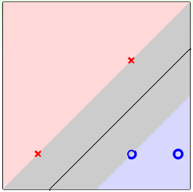
$\exp(-1\|\mathbf{x}-\mathbf{x}'\|^2)$ $\exp(-10\|\mathbf{x}-\mathbf{x}'\|^2)$ $\exp(-100\|\mathbf{x}-\mathbf{x}'\|^2)$

- large $\gamma \Longrightarrow$ sharp Gaussians $\Longrightarrow$ 'overfit'?
- **warning: SVM can still overfit :-(**

Gaussian SVM: need careful selection of γ

Questions?

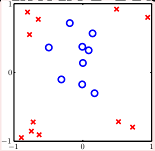
Linear Kernel: Cons and Pros



$$K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$$

Cons

- restricted
  —**not always separable?!**

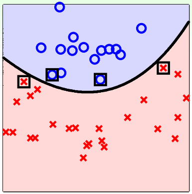


Pros

- safe—**linear first, remember? :-)**
- fast—with **special QP solver** in primal
- very explainable—**w and SVs** say something

linear kernel: an important **basic** tool

Polynomial Kernel: Cons and Pros



$$K(\mathbf{x}, \mathbf{x}') = (\zeta + \gamma \mathbf{x}^T \mathbf{x}')^Q$$

Cons

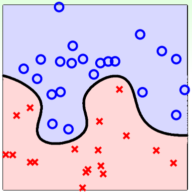
- **numerical difficulty** for large Q
  - $|\zeta + \gamma \mathbf{x}^T \mathbf{x}'| < 1$: $K \to 0$
  - $|\zeta + \gamma \mathbf{x}^T \mathbf{x}'| > 1$: $K \to$ big
- three parameters (γ, ζ, Q) —**more difficult to select**

Pros

- **less restricted** than linear
- strong physical control —'knows' **degree Q**

polynomial kernel: perhaps **small-Q only**
—sometimes efficiently done by **linear on $\boldsymbol{\Phi}_Q(\mathbf{x})$**

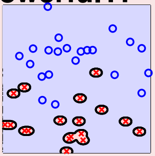
Gaussian Kernel: Cons and Pros



$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x}-\mathbf{x}'\|^2)$$

Cons

- **mysterious**—no **w**
- **slower** than linear
- **too powerful?!**



Pros

- **more powerful than linear/poly.**
- bounded—**less numerical difficulty than poly.**
- one parameter only—**easier to select than poly.**

Gaussian kernel: **one of most popular** but shall **be used with care**

Other Valid Kernels

- kernel represents **special** similarity: $\boldsymbol{\Phi}(\mathbf{x})^T\boldsymbol{\Phi}(\mathbf{x}')$
- any similarity $\Longrightarrow$ valid kernel? **not really**
- necessary **& sufficient** conditions for valid kernel: **Mercer's condition**
  - symmetric
  - let $k_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$, the matrix K

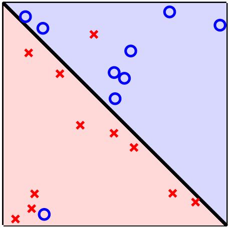
$$
\begin{aligned}
&= \begin{bmatrix} \boldsymbol{\Phi}(\mathbf{x}_1)^T\boldsymbol{\Phi}(\mathbf{x}_1) & \boldsymbol{\Phi}(\mathbf{x}_1)^T\boldsymbol{\Phi}(\mathbf{x}_2) & \ldots & \boldsymbol{\Phi}(\mathbf{x}_1)^T\boldsymbol{\Phi}(\mathbf{x}_N) \\ \boldsymbol{\Phi}(\mathbf{x}_2)^T\boldsymbol{\Phi}(\mathbf{x}_1) & \boldsymbol{\Phi}(\mathbf{x}_2)^T\boldsymbol{\Phi}(\mathbf{x}_2) & \ldots & \boldsymbol{\Phi}(\mathbf{x}_2)^T\boldsymbol{\Phi}(\mathbf{x}_N) \\ \ldots & \ldots & \ldots & \ldots \\ \boldsymbol{\Phi}(\mathbf{x}_N)^T\boldsymbol{\Phi}(\mathbf{x}_1) & \boldsymbol{\Phi}(\mathbf{x}_N)^T\boldsymbol{\Phi}(\mathbf{x}_2) & \ldots & \boldsymbol{\Phi}(\mathbf{x}_N)^T\boldsymbol{\Phi}(\mathbf{x}_N) \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \ldots & \mathbf{z}_N \end{bmatrix}^T \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \ldots & \mathbf{z}_N \end{bmatrix} \\
&= \mathrm{Z}\mathrm{Z}^T \text{ must **always** be **positive semi-definite**}
\end{aligned}
$$

define your own kernel: possible, **but hard**

Questions?

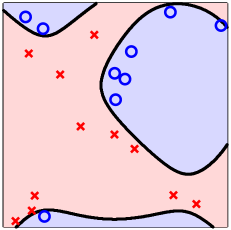
Cons of Hard-Margin SVM

recall: SVM can still overfit :-(



$\mathbf{\Phi}_1$

- part of reasons: $\mathbf{\Phi}$
- other part: **separable**



$\mathbf{\Phi}_4$

if always insisting on **separable** ($\Longrightarrow$ **shatter**), have power to **overfit to noise**

Give Up on Some Examples

want: **give up** on some noisy examples

min.-error perceptron

$$\min_{b,\mathbf{w}} \sum_{n=1}^{N} [\![y_n \neq \text{sign}(\mathbf{w}^T\mathbf{z}_n + b)]\!]$$

hard-margin SVM

$$\min_{b,\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{s.t.} \quad y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1 \text{ for all } n$$

combination:

$$\min_{b,\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \cdot \sum_{n=1}^{N} [\![y_n \neq \text{sign}(\mathbf{w}^T\mathbf{z}_n + b)]\!]$$

$$\text{s.t.} \quad y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1 \text{ for } \textbf{correct } n$$

$$y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq -\infty \text{ for } \textbf{incorrect } n$$

C: trade-off of large margin & noise tolerance

Soft-Margin SVM (1/2)

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \cdot \sum_{n=1}^{N} [\![y_n \neq \text{sign}(\mathbf{w}^T\mathbf{z}_n + b)]\!]$$

$$\text{s.t.} \quad y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1 - \infty \cdot [\![y_n \neq \text{sign}(\mathbf{w}^T\mathbf{z}_n + b)]\!]$$

- $[\![\cdot]\!]$: non-linear, **not QP anymore :-(**
  —what about dual? kernel?
- cannot distinguish **small error** (slightly away from fat boundary) or **large error** (a...w...a...y... from fat boundary)

- record '**margin violation**' by ξ_n—**linear constraints**
- penalize with **margin violation** instead of **error count**
  —**quadratic objective**

soft-margin SVM: $$\min_{b,\mathbf{w},\boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \cdot \sum_{n=1}^{N} \xi_n$$

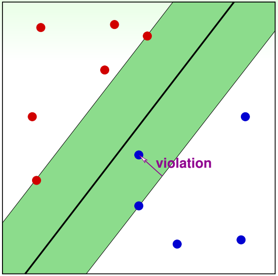
$$\text{s.t.} \quad y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1 - \xi_n \text{ and } \xi_n \geq 0 \text{ for all } n$$

Soft-Margin SVM (2/2)

- record '**margin violation**' by ξ_n
- penalize with **margin violation**

$$\min_{b,\mathbf{w},\boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \cdot \sum_{n=1}^{N} \xi_n$$

$$\text{s.t.} \quad y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1 - \xi_n \text{ and } \xi_n \geq 0 \text{ for all } n$$



- parameter C: trade-off of large margin & margin violation
  - large C: want less margin violation
  - small C: want large margin
- QP of $\tilde{d} + 1 + N$ variables, $2N$ constraints

next: remove dependence on $\tilde{d}$ by soft-margin SVM primal $\Rightarrow$ **dual**?

Questions?

Lagrange Dual

primal: $$\min_{b,\mathbf{w},\boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \cdot \sum_{n=1}^{N} \xi_n$$

s.t. $y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1 - \xi_n$ and $\xi_n \geq 0$ for all n

Lagrange function with Lagrange multipliers α_n and β_n

$$\mathcal{L}(b, \mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \cdot \sum_{n=1}^{N} \xi_n + \sum_{n=1}^{N} \alpha_n \cdot \left(1 - \xi_n - y_n(\mathbf{w}^T\mathbf{z}_n + b)\right) + \sum_{n=1}^{N} \beta_n \cdot (-\xi_n)$$

want: Lagrange dual

$$\max_{\alpha_n \geq 0,\ \beta_n \geq 0} \left(\min_{b,\mathbf{w},\boldsymbol{\xi}} \quad \mathcal{L}(b, \mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \right)$$

Simplify ξ_n and β_n

$$\max_{\alpha_n \geq 0,\ \beta_n \geq 0} \left(\min_{b, \mathbf{w}, \boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \cdot \sum_{n=1}^{N} \xi_n + \sum_{n=1}^{N} \alpha_n \cdot (1 - \xi_n - y_n(\mathbf{w}^T\mathbf{z}_n + b)) + \sum_{n=1}^{N} \beta_n \cdot (-\xi_n) \right)$$

- $\frac{\partial \mathcal{L}}{\partial \xi_n} = 0 = C - \alpha_n - \beta_n$
- no loss of optimality if solving with implicit constraint $\beta_n = C - \alpha_n$ and explicit constraint $0 \leq \alpha_n \leq C$: β_n removed

ξ can also be removed :-), like how we removed b

$$\max_{0 \leq \alpha_n \leq C,\ \beta_n = C - \alpha_n} \left(\min_{b, \mathbf{w}, \xi} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{n=1}^{N} \alpha_n (1 - y_n(\mathbf{w}^T\mathbf{z}_n + b)) \cancel{+ \sum_{n=1}^{N} (C - \alpha_n - \beta_n) \cdot \xi_n} \right)$$

Other Simplifications

$$\max_{0 \le \alpha_n \le C,\ \beta_n = C - \alpha_n} \left(\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{n=1}^{N} \alpha_n (1 - y_n(\mathbf{w}^T\mathbf{z}_n + b)) \right)$$

familiar? :-)

- inner problem **same as hard-margin SVM**
- $\frac{\partial \mathcal{L}}{\partial b} = 0$: no loss of optimality if solving with constraint $\sum_{n=1}^{N} \alpha_n y_n = 0$
- $\frac{\partial \mathcal{L}}{\partial w_i} = 0$: no loss of optimality if solving with constraint $\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n$

standard dual can be derived
using the same steps as Lecture 10

Standard Soft-Margin SVM Dual

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \mathbf{z}_n^T \mathbf{z}_m - \sum_{n=1}^{N} \alpha_n$$

$$\text{subject to} \quad \sum_{n=1}^{N} y_n \alpha_n = 0;$$

$$0 \le \alpha_n \le C, \text{ for } n = 1, 2, \ldots, N;$$

$$\text{implicitly} \quad \mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n;$$

$$\beta_n = C - \alpha_n, \text{ for } n = 1, 2, \ldots, N$$

—only difference to hard-margin: upper bound on α_n

another (convex) **QP**,
with N variables & $2N + 1$ constraints

Questions?

Kernel Soft-Margin SVM

Kernel Soft-Margin SVM Algorithm

1. $q_{n,m} = y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$; $\mathbf{p} = -\mathbf{1}_N$; $(\mathrm{A}, \mathbf{c})$ for equ./lower-bound/**upper-bound** constraints
2. $\alpha \leftarrow \mathrm{QP}(\mathrm{Q_D}, \mathbf{p}, \mathrm{A}, \mathbf{c})$
3. $b \leftarrow ?$
4. return SVs and their α_n as well as b such that for new $\mathbf{x}$,

$$g_{\mathrm{SVM}}(\mathbf{x}) = \mathrm{sign}\left(\sum_{\text{SV indices } n} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b \right)$$

- almost the same as hard-margin
- more flexible than hard-margin —**primal/dual always solvable**

remaining question: step 3?

Solving for b

hard-margin SVM

complementary slackness:

$$\alpha_n(1 - y_n(\mathbf{w}^T\mathbf{z}_n + b)) = 0$$

- SV ($\alpha_s > 0$)
  $\Rightarrow b = y_s - \mathbf{w}^T\mathbf{z}_s$

soft-margin SVM

complementary slackness:

$$\alpha_n(1 - \xi_n - y_n(\mathbf{w}^T\mathbf{z}_n + b)) = 0$$

$$(C - \alpha_n)\xi_n = 0$$

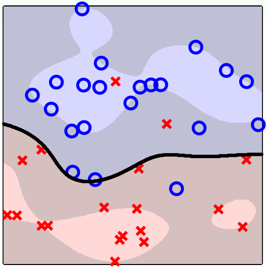
- SV ($\alpha_s > 0$)
  $\Rightarrow b = y_s - y_s\xi_s - \mathbf{w}^T\mathbf{z}_s$
- free ($\alpha_s < C$)
  $\Rightarrow \xi_s = 0$

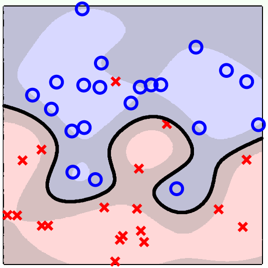
solve unique b with free SV $(\mathbf{x}_s, y_s)$:

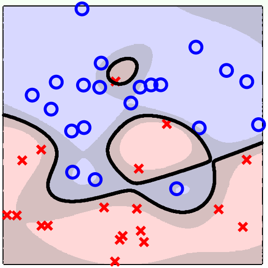
$$b = y_s - \sum_{\text{SV indices } n} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_s)$$

—range of b otherwise

Soft-Margin Gaussian SVM in Action







$C = 1$ $C = 10$ $C = 100$

- large $C \Longrightarrow$ less noise tolerance $\Longrightarrow$ 'overfit'?
- **warning: SVM can still overfit :-(**

soft-margin Gaussian SVM:
need careful selection of (γ, C)

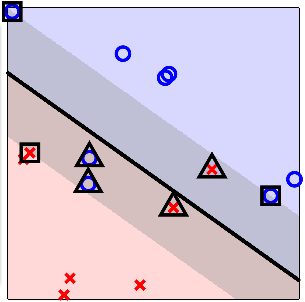
Physical Meaning of α_n

complementary slackness:

$$\alpha_n(1-\xi_n - y_n(\mathbf{w}^T\mathbf{z}_n + b)) = 0$$

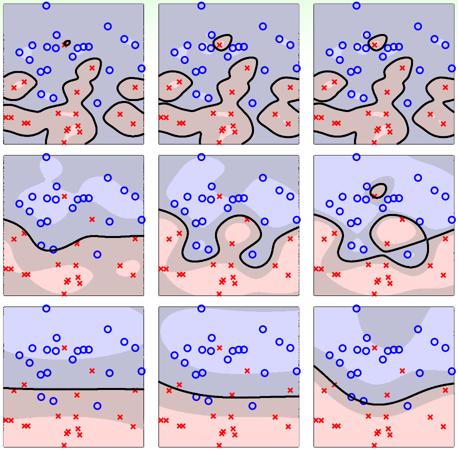
$$(C - \alpha_n)\xi_n = 0$$

- non SV ($0 = \alpha_n$): $\xi_n = 0$, 'away from'/on fat boundary
- $\square$ free SV ($0 < \alpha_n < C$): $\xi_n = 0$, on fat boundary, locates b
- $\triangle$ bounded SV ($\alpha_n = C$): $\xi_n =$ violation amount, 'violate'/on fat boundary



α_n can be used for **data analysis**

Practical Need: Model Selection



- complicated even for (C, γ) **of Gaussian SVM**
- more combinations if including other kernels or parameters

how to select? **validation :-)**

Questions?

Wrap-Up

Hard-Margin Primal

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$
$$\text{s.t.} \quad y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1$$

Soft-Margin Primal

$$\min_{b,\mathbf{w},\boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n$$
$$\text{s.t.} \quad y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1 - \xi_n, \xi_n \geq 0$$

Hard-Margin Dual

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^T\mathrm{Q}\boldsymbol{\alpha} - \mathbf{1}^T\boldsymbol{\alpha}$$
$$\text{s.t.} \quad \mathbf{y}^T\boldsymbol{\alpha} = 0$$
$$0 \leq \alpha_n$$

Soft-Margin Dual

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^T\mathrm{Q}\boldsymbol{\alpha} - \mathbf{1}^T\boldsymbol{\alpha}$$
$$\text{s.t.} \quad \mathbf{y}^T\boldsymbol{\alpha} = 0$$
$$0 \leq \alpha_n \leq C$$

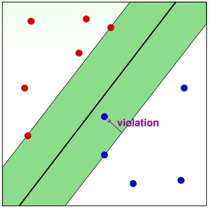
soft-margin preferred in practice;
linear: LIBLINEAR; non-linear: LIBSVM

Slack Variables ξ_n

- record '**margin violation**' by ξ_n
- penalize with **margin violation**

$$\min_{b,\mathbf{w},\boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \cdot \sum_{n=1}^{N} \xi_n$$

$$\text{s.t.} \quad y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 1 - \xi_n \text{ and } \xi_n \geq 0 \text{ for all } n$$



on any $(b, \mathbf{w})$, $\xi_n =$ **margin violation** $= \max\big(1 - y_n(\mathbf{w}^T\mathbf{z}_n + b), 0\big)$

- $(\mathbf{x}_n, y_n)$ violating margin: $\xi_n = 1 - y_n(\mathbf{w}^T\mathbf{z}_n + b)$
- $(\mathbf{x}_n, y_n)$ not violating margin: $\xi_n = 0$

'unconstrained' form of soft-margin SVM:

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C \sum_{n=1}^{N} \max\big(1 - y_n(\mathbf{w}^T\mathbf{z}_n + b), 0\big)$$

Unconstrained Form

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N} \max\left(1 - y_n(\mathbf{w}^T\mathbf{z}_n + b), 0\right)$$

familiar? :-)

$$\min \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum \widehat{\mathrm{err}}$$

just L2 regularization

$$\min \quad \frac{\lambda}{N}\mathbf{w}^T\mathbf{w} + \frac{1}{N}\sum \mathrm{err}$$

with shorter $\mathbf{w}$, another parameter, and special err

why not solve this? :-)

- not QP, **no (?) kernel trick**
- $\max(\cdot, 0)$ **not differentiable**, harder to solve

SVM as Regularized Model

	minimize	constraint
regularization by constraint	E_{in}	$\mathbf{w}^T\mathbf{w} \leq C$
hard-margin SVM	$\mathbf{w}^T\mathbf{w}$	$E_{\text{in}} = 0$ [and more]
L2 regularization	$\frac{\lambda}{N}\mathbf{w}^T\mathbf{w} + E_{\text{in}}$	
soft-margin SVM	$\frac{1}{2}\mathbf{w}^T\mathbf{w} + CN\widehat{E_{\text{in}}}$	

large margin $\Longleftrightarrow$ fewer hyperplanes $\Longleftrightarrow$ L2 regularization of short $\mathbf{w}$

soft margin $\Longleftrightarrow$ special $\widehat{\text{err}}$

larger C or C $\Longleftrightarrow$ smaller λ $\Longleftrightarrow$ less regularization

viewing SVM as regularized model:

allows **extending/connecting** to other learning models

Questions?

Summary

1 Embedding Numerous Features: Kernel Models

Lecture 11: Support Vector Machine (2)

- Kernel Trick
  kernel as shortcut of transform + inner product
- Polynomial Kernel
  embeds specially-scaled polynomial transform
- Gaussian Kernel
  embeds infinite dimensional transform
- Comparison of Kernels
  linear for efficiency or Gaussian for power
- Motivation and Primal Problem
  add margin violations ξ_n
- Dual Problem
  upper-bound α_n by C
- Messages behind Soft-Margin SVM
  bounded/free SVs for data analysis
- Soft-Margin SVM as Regularized Model
  L2-regularization with hinge error measure