CZ4041/SC4000: Machine Learning

Additional Notes: Nonseparable & Kernel SVMs

Li Boyang, Albert

School of Computer Science and Engineering, NTU, Singapore

Acknowledgements: some figures are adapted from the lecture notes of the book "Introduction to Data Mining" (Chap. 5). Slides are modified from the version prepared by Dr. Sinno Pan.

Linear SVMs: Separable Case

- Optimization problem of linear SVM

$$\boxed{\min_{\boldsymbol{w},b} \frac{\|\boldsymbol{w}\|_2^2}{2}}$$ Maximize margin

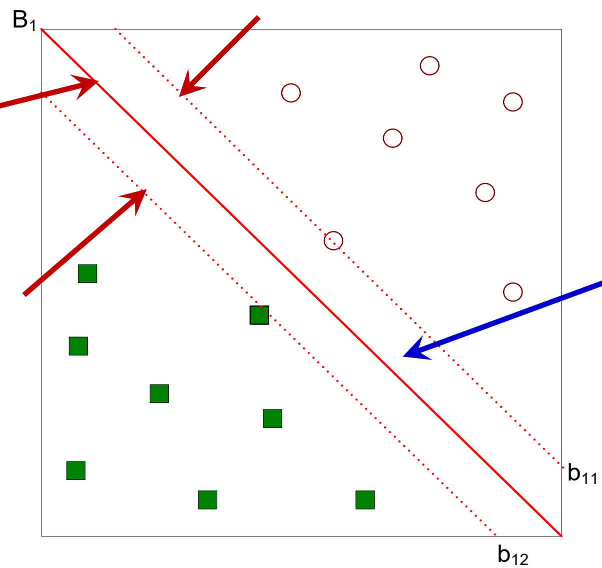
$$\boxed{\text{s.t.} \quad y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1,\ i = 1, \ldots, N}$$

Perfectly separate training data

$\boldsymbol{w} \cdot \boldsymbol{x} + b = 1$

$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$

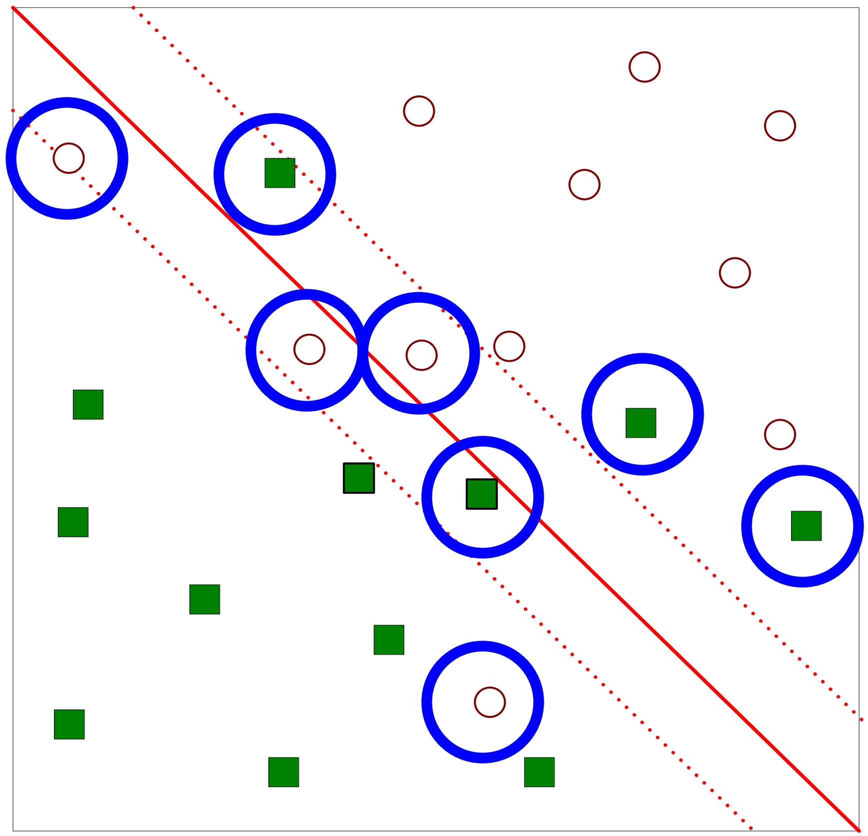
$\boldsymbol{w} \cdot \boldsymbol{x} + b = -1$



No training data locate within the margin

Linear SVMs: Nonseparable Case

- What if the problem is not separable?



Slack variables $\xi_i \geq 0$ need to be introduced to absorb errors

Slack Variables

- For Separable Case:

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

OR

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1$$

- For Nonseparable Case:

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

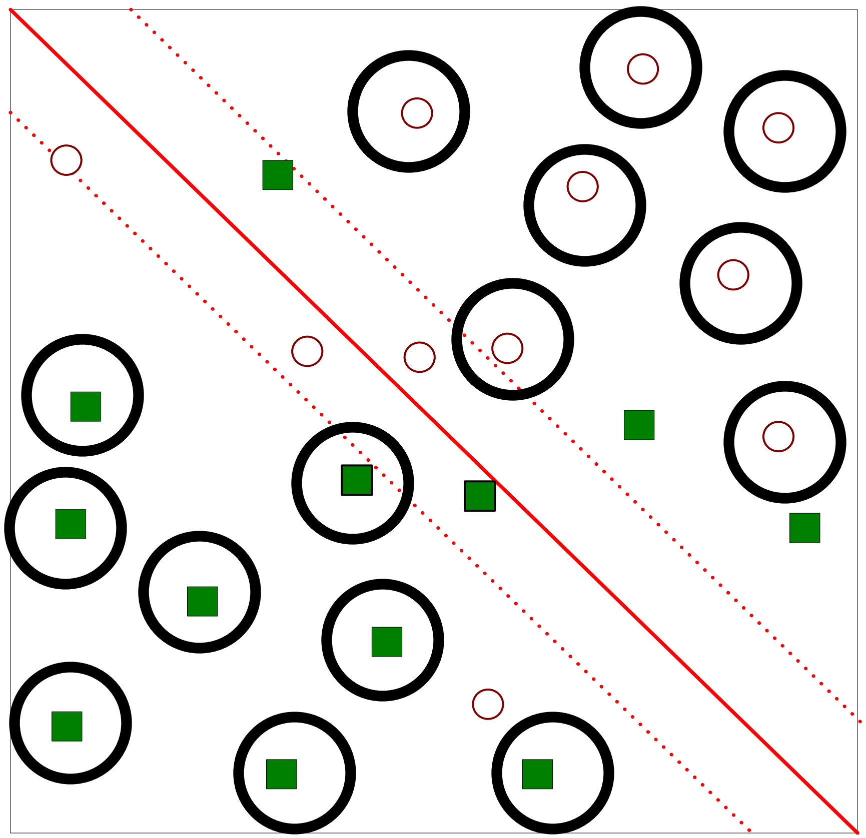
$\xi_i \geq 0$

Slack variables

OR

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

Slack Variables (cont.)



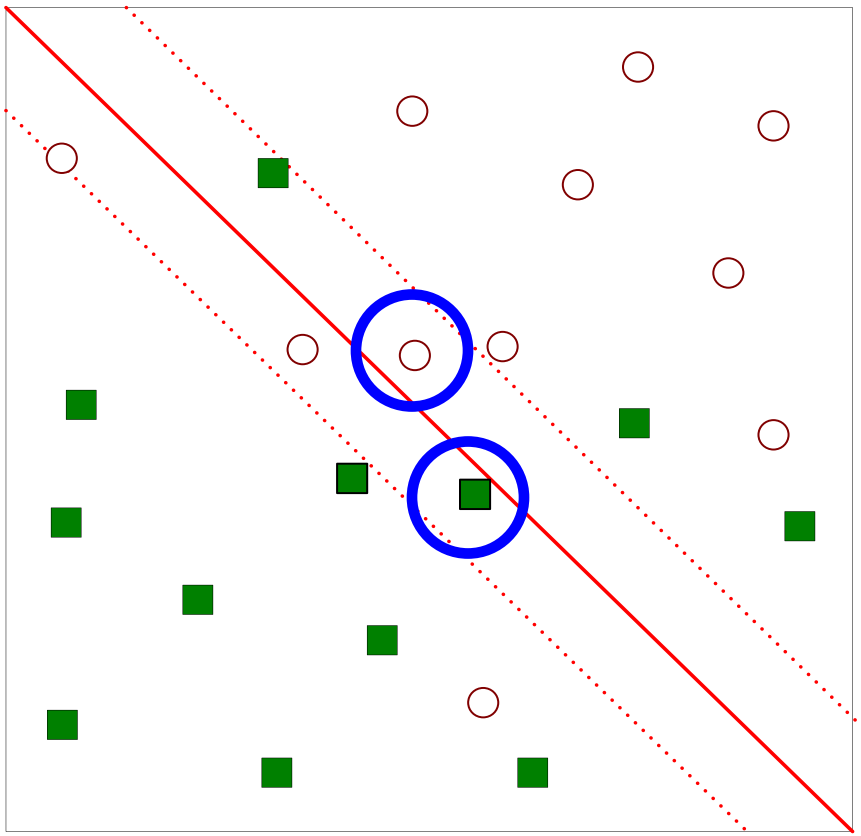
If $\xi_i = 0$, there is no problem with $\boldsymbol{x}_i$

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

$$\Downarrow$$

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1$$

Slack Variables (cont.)



If $0 < \xi_i < 1$, $\boldsymbol{x}_i$ is correctly classified but in the margin

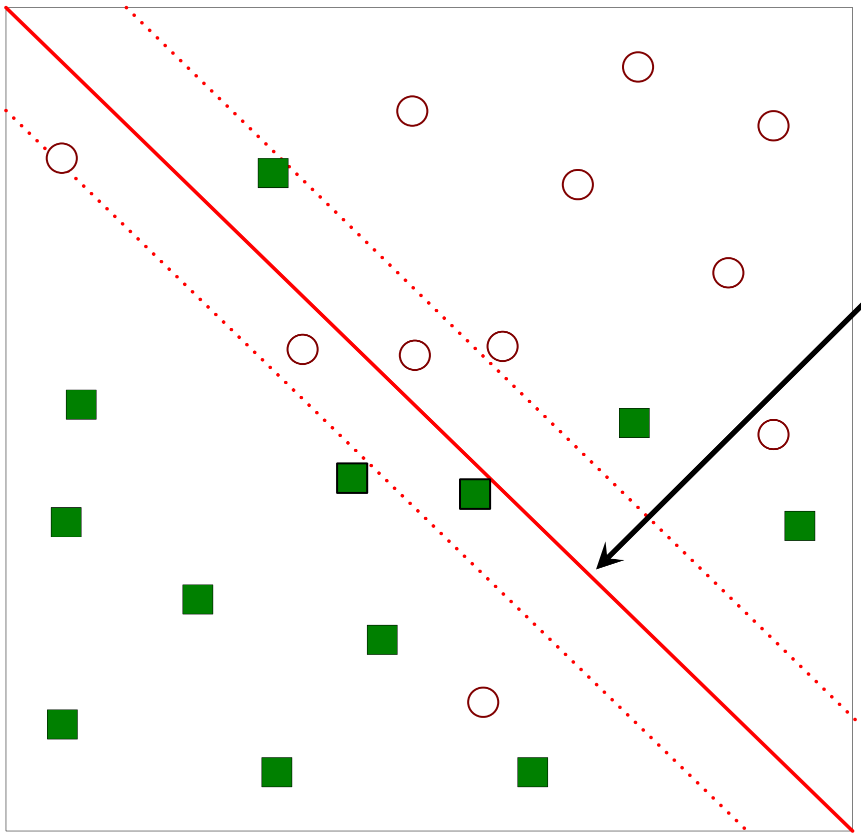
$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

$$\Downarrow$$

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq k$$

$$0 < k < 1$$

Slack Variables (cont.)

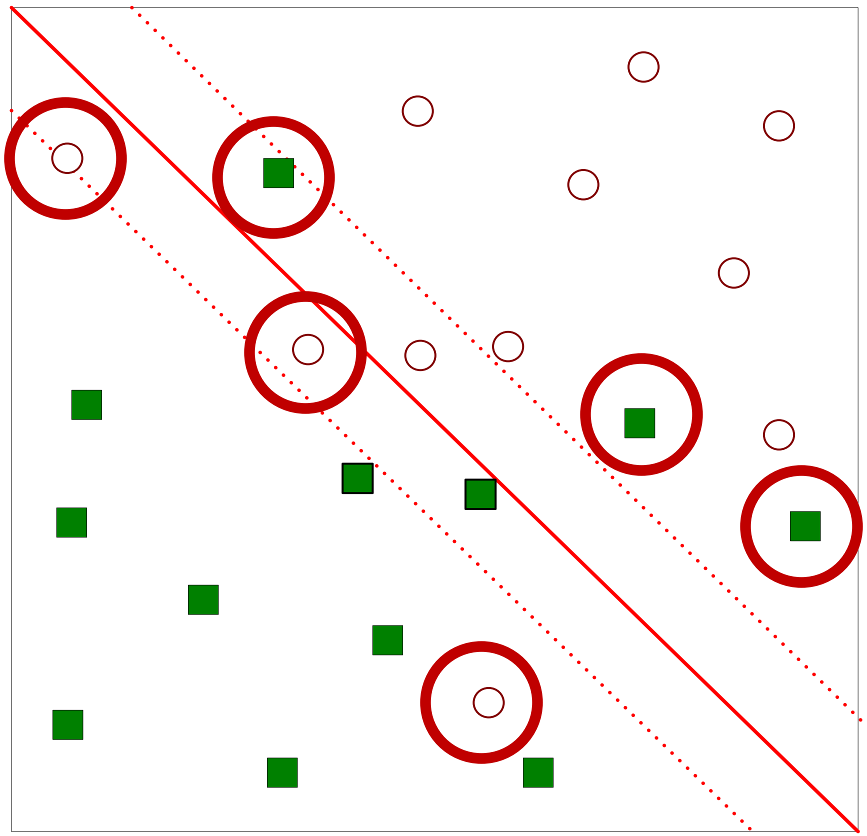


If $\xi_i = 1$, $\boldsymbol{x}_i$ is on the decision boundary (random guess)

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 0$$

Slack Variables (cont.)



If $\xi_i > 1$, $\boldsymbol{x}_i$ is misclassified

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i$$

$$y_i \times (\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq -k$$

$$k > 0$$

Can be negative

$$\underline{\boldsymbol{w} \cdot \boldsymbol{x}_i + b} \geq -k, \text{ if } y_i = 1$$

$$\underline{\boldsymbol{w} \cdot \boldsymbol{x}_i + b} \leq k, \text{ if } y_i = -1$$

Can be positive

Soft Error

- The number of misclassifications is $\#\{\xi_i > 1\}$
- The number of nonseparable points is $\#\{\xi_i > 0\}$
- Soft errors:

$$\sum_i \xi_i$$

Linear SVMs: Nonseparable Case

- Linear SVMs with soft errors:

$$\min_{\boldsymbol{w},b,\xi_i} \frac{\|\boldsymbol{w}\|_2^2}{2} + C\left(\sum_{i=1}^{N} \xi_i\right)$$

$$\text{s.t.} \quad y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1 - \xi_i, \; i = 1, \ldots, N, \quad \xi_i \geq 0$$

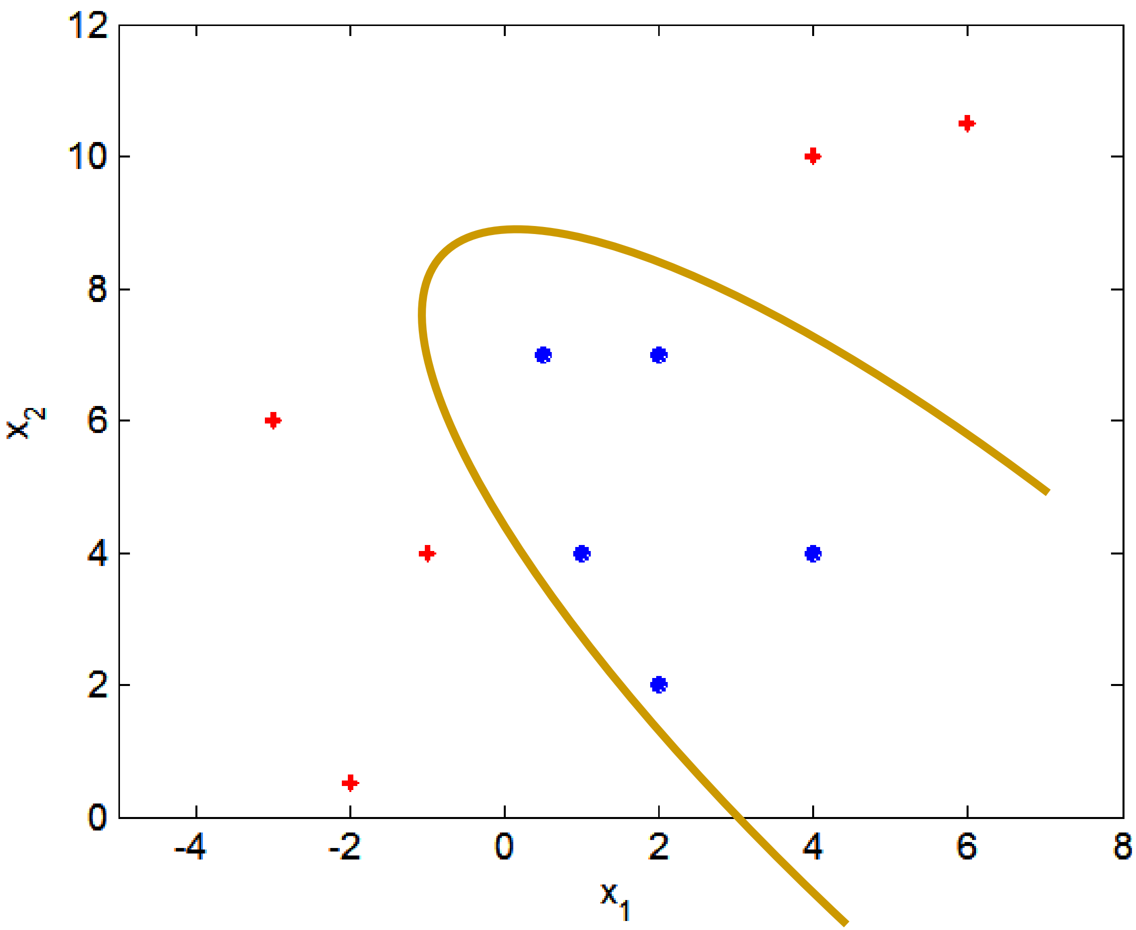
Penalize the decision boundary with large values of slack variables

$C \geq 0$ is a parameter to tradeoff the impact of margin maximization and tolerable errors

Nonnegative ξ_i provides an estimate of the error of the decision boundary on the training example $\boldsymbol{x}_i$

Nonlinear SVMs

- What if decision boundary is not linear?



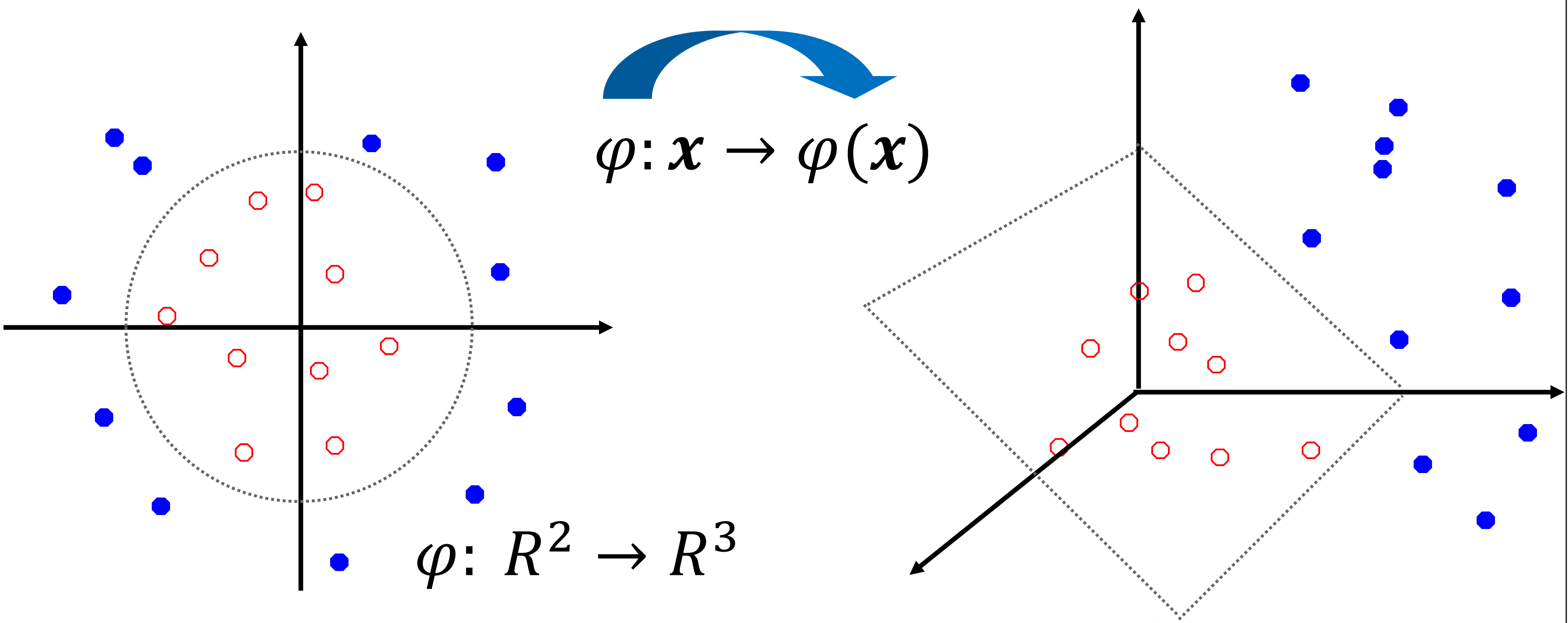
Kernel trick in the dual form

Nonlinear SVMs (cont.)

- How to generalize linear decision boundary to become nonlinear?
- Key idea: transform $\boldsymbol{x}_i$ to a higher dimensional space to "make life easier"
  - Input space: the space the point $\boldsymbol{x}_i$ are located
  - Feature space: the space of $\varphi(\boldsymbol{x}_i)$ after transformation
- Assumption: in a higher dimensional space, it is easier to find a linear hyperplane to classify data

Feature Mapping

- The original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



$$(X_1, X_2) \rightarrow \left(X_1^2, \sqrt{2} X_2 X_1, X_2^2\right)$$

Nonlinear SVMs (cont.)

- Optimization problem of nonlinear SVMs

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

$$\text{s.t.} \quad y_i \times \left(\underline{\boldsymbol{w} \cdot \varphi(\boldsymbol{x}_i) + b}\right) \geq 1,\ i = 1, \ldots, N$$

$$\boldsymbol{w} \cdot \varphi(\boldsymbol{x}_i) + b = 0$$

Hyperplane in feature space

- Computation in the feature space can be costly because it is high dimensional
  - The feature space is typically very high dimensional!
- The kernel trick comes to rescue

Nonlinear SVM: Kernel Trick

- Suppose $\varphi(\cdot)$ is given as follows, mapping an instance from 2-dimensional space to 6-dimensional space:

$$\varphi([X_1, X_2]) = \left[1, \sqrt{2}X_1, \sqrt{2}X_2, X_1^2, X_2^2, \sqrt{2}X_1X_2\right]$$

- Given two data instances: $\boldsymbol{a} = [A_1, A_2]$ and $\boldsymbol{b} = [B_1, B_2]$

$$\varphi(\boldsymbol{a}) = \left[1, \sqrt{2}A_1, \sqrt{2}A_2, A_1^2, A_2^2, \sqrt{2}A_1A_2\right]$$

$$\varphi(\boldsymbol{b}) = \left[1, \sqrt{2}B_1, \sqrt{2}B_2, B_1^2, B_2^2, \sqrt{2}B_1B_2\right]$$

- Inner product of the two instances after feature mapping:

$$\varphi(\boldsymbol{a}) \cdot \varphi(\boldsymbol{b}) = 1 + 2A_1B_1 + 2A_2B_2 + A_1^2B_1^2 + A_2^2B_2^2 + 2A_1A_2B_1B_2$$

$$= (1 + A_1B_1 + A_2B_2)^2$$

Kernel Trick (cont.)

- Inner product of the two instances after feature mapping:

$$\boxed{\varphi(\boldsymbol{a}) \cdot \varphi(\boldsymbol{b})} = 1 + 2A_1B_1 + 2A_2B_2 + A_1^2B_1^2 + A_2^2B_2^2 + 2A_1A_2B_1B_2$$

$$= (1 + A_1B_1 + A_2B_2)^2$$

- So, if we define the kernel function as follows, there is no need to carry out $\varphi(\cdot)$ explicitly

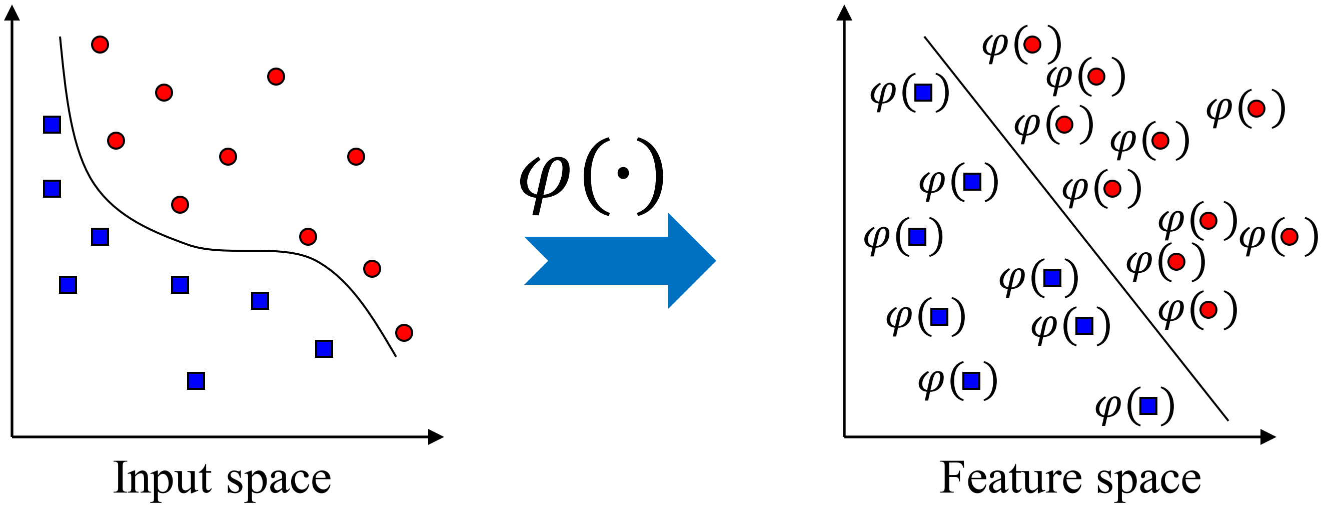
$$\boxed{k(\boldsymbol{a}, \boldsymbol{b})} = (1 + A_1B_1 + A_2B_2)^2 = (1 + \boldsymbol{a} \cdot \boldsymbol{b})^2$$

- This use of kernel function to avoid carrying out $\varphi(\cdot)$ explicitly is known as the kernel trick

Kernel Trick: General Idea

- If $\varphi(\cdot)$ satisfies some conditions, then we can find a function $k(\cdot,\cdot)$ such that

Kernel function → $\boxed{k(\boldsymbol{x}_i, \boldsymbol{x}_j)} = \varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}_j)$



$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{2\sigma^2}\right)$$

How to Apply Kernel Trick?

- Optimization problem for nonlinear SVMs (separable)

Primal Form

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

$$\text{s.t.} \quad y_i \times (\boldsymbol{w} \cdot \boxed{\varphi(\boldsymbol{x}_i)} + b) \geq 1, \; i = 1, \ldots, N$$

Instances in the feature space do not appear in the form of inner products

- The kernel trick is not applicable
- How about its dual form?

Lagrange multiplier method

Lagrange Multiplier Method: Idea

- Given: an objective $f(\boldsymbol{w})$ to be minimized, with a set of inequality constraints to be satisfied $h_i(\boldsymbol{w}) \leq 0, i = 1,2, \dots, q$

$$\min_{\boldsymbol{w}} \; f(\boldsymbol{w})$$
$$\text{s.t.} \quad h_i(\boldsymbol{w}) \leq 0, \; i = 1, \dots, q$$

- The Lagrangian for the optimization problem:

$$L(\boldsymbol{w}, \boxed{\boldsymbol{\lambda}}) = f(\boldsymbol{w}) + \sum_{i=1}^{q} \boxed{\lambda_i} h_i(\boldsymbol{w})$$

$\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_q)$

The Lagrange multipliers

The Dual Form (Separable)

- By using Lagrangian Multiplier method

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

Primal Form

$$\text{s.t.} \quad y_i \times (\boldsymbol{w} \cdot \varphi(\boldsymbol{x}_i) + b) \geq 1,\ i = 1, \dots, N$$

$$\max_{\boldsymbol{\lambda}} L_D(\boldsymbol{\lambda}) = -\left(\frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j \left(\varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}_j)\right) - \sum_{i=1}^{N} \lambda_i\right)$$

Dual Form

Dual Optimization Problem

- The dual Lagrangian involves only the Lagrange multipliers and the training data
- The negative sign in the dual Lagrangian transforms a minimization problem of the primal form to a maximization problem of the dual form
- The objective is to maximize $L_D(\lambda)$
  - Can be solved using numerical techniques such as quadratic programming

Dual Optimization Problem (cont.)

- Once the λ_i's are found, we can obtain the feasible solutions for $\boldsymbol{w}$ and b from

$$\boldsymbol{w} = \sum_{i=1}^{N} \lambda_i y_i \varphi(\boldsymbol{x}_i) \quad \textbf{AND} \quad \lambda_i(y_i(\boldsymbol{w} \cdot \varphi(\boldsymbol{x}_i) + b) - 1) = 0$$

- The decision boundary can be expressed as

$$\boldsymbol{w} \cdot \varphi(\boldsymbol{x}) + b = \left(\sum_{i=1}^{N} \boxed{\lambda_i} y_i \varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}) \right) + \boxed{b} = 0$$

If $\boldsymbol{x}_i$ is a support vector, then the corresponding $\lambda_i > 0$, otherwise, $\lambda_i = 0$

Dual Optimization Problem (cont.)

- For a test instance $\boldsymbol{x}^*$, it can be classified using

$$f(\boldsymbol{x}^*) = \operatorname{sign}\left(\sum_{i=1}^{N} \lambda_i y_i \varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}^*) + b\right)$$

Nonlinear SVM via Kernel Trick

Training: $$\max_{\lambda}\left(\sum_{i=1}^{N}\lambda_i-\frac{1}{2}\sum_{i,j}\lambda_i\lambda_j y_i y_j\boxed{\left(\varphi(\boldsymbol{x}_i)\cdot\varphi(\boldsymbol{x}_j)\right)}\right)$$

Decision boundary: $$\sum_{i=1}^{N}\lambda_i y_i\boxed{\left(\varphi(\boldsymbol{x}_i)\cdot\varphi(\boldsymbol{x}^*)\right)}+b=0$$

- The data points only appear as inner product
- As long as the inner product in the feature space can be calculated, no need for the explicit mapping

Nonlinear SVM via Kernel Trick (cont.)

- Replace inner product in feature space by kernel function

Training: $$\max_{\lambda}\left(\sum_{i=1}^{N}\lambda_i-\frac{1}{2}\sum_{i,j}\lambda_i\lambda_j y_i y_j \boxed{k(\boldsymbol{x}_i,\boldsymbol{x}_j)}\right)$$

Decision boundary: $$\sum_{i=1}^{N}\boxed{\lambda_i} y_i \boxed{k(\boldsymbol{x}_i,\boldsymbol{x}^*)}+b=0$$

$$k(\boldsymbol{x}_i,\boldsymbol{x}^*)=\varphi(\boldsymbol{x}_i)\cdot\varphi(\boldsymbol{x}^*)$$

If $\boldsymbol{x}_i$ is a support vector, then the corresponding $\lambda_i>0$, otherwise, $\lambda_i=0$

$$k(\boldsymbol{x}_i,\boldsymbol{x}_j)=\varphi(\boldsymbol{x}_i)\cdot\varphi(\boldsymbol{x}_j)$$

Kernel Functions: Examples

- Linear kernel

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

- Radial basis function kernel with width σ

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{2\sigma^2}\right)$$

- Polynomial kernel with degree d

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\boldsymbol{x}_i \cdot \boldsymbol{x}_j + 1\right)^d$$

Soft Margin Dual Form

- By using Lagrangian Multiplier method

$$\min_{\boldsymbol{w},b,\xi_i} \frac{\|\boldsymbol{w}\|_2^2}{2} + C\left(\sum_{i=1}^{N} \xi_i\right)$$

$$\text{s.t.} \quad y_i(\boldsymbol{w} \cdot \varphi(\boldsymbol{x}_i) + b) \geq 1 - \xi_i,\ i = 1, \ldots, N,$$
$$\xi_i \geq 0, i = 1, \ldots, N,$$

$$\min_{\boldsymbol{\lambda}} L_D(\boldsymbol{\lambda}) = -\left(\frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j \boxed{\left(\varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x}_j)\right)} - \sum_{i=1}^{N} \lambda_i\right)$$

$$\text{s.t.,}\ 0 \leq \lambda_i \leq C$$

Kernel trick can be applied

Soft Margin Dual Form (cont.)

$$\boldsymbol{w}\cdot\varphi(\boldsymbol{x})+b=\left(\sum_{i=1}^{N}\lambda_i y_i \overbrace{\boxed{\varphi(\boldsymbol{x}_i)\cdot\varphi(\boldsymbol{x})}}^{k(\boldsymbol{x}_i,\boldsymbol{x}_j)}\right)+b=0$$

Kernel trick can be applied

- For a test instance $\boldsymbol{x}^*$, it can be classified using

$$f(\boldsymbol{x}^*)=\operatorname{sign}\left(\sum_{i=1}^{N}\lambda_i y_i \underbrace{\boxed{\varphi(\boldsymbol{x}_i)\cdot\varphi(\boldsymbol{x}^*)}}_{k(\boldsymbol{x}_i,\boldsymbol{x}^*)}+b\right)$$

Popular Toolboxes of SVMs

- LIBSVM
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm/
- LIBLINEAR
  - http://www.csie.ntu.edu.tw/~cjlin/liblinear/

- SVM-light
  - http://svmlight.joachims.org/
- SVM-struct
  - http://www.cs.cornell.edu/People/tj/svm_light/svm_struct.html
- SVM-perf
  - http://www.cs.cornell.edu/People/tj/svm_light/svm_perf.html
- SVM-rank
  - http://www.cs.cornell.edu/People/tj/svm_light/svm_rank.html

Further Readings

- *A Tutorial on Support Vector Machines for Pattern Recognition,* by Christopher J. C. Burges, DMKD, 1998
- *Convex Optimization,* by Stephen Boyd and Lieven Vandenberghe, Cabridge University Press, 2004
- *Learning with Kernel,* by Bernhard Scholkopf and Alex Smola, The MIT Press, 2002
- *Statistical Learning Theory,* by Vladimir N. Vapnik, Wiley-Interscience, 1998

Thank you!