

# Support Vector Machine

Il-Chul Moon  
Dept. of Industrial and Systems Engineering  
KAIST

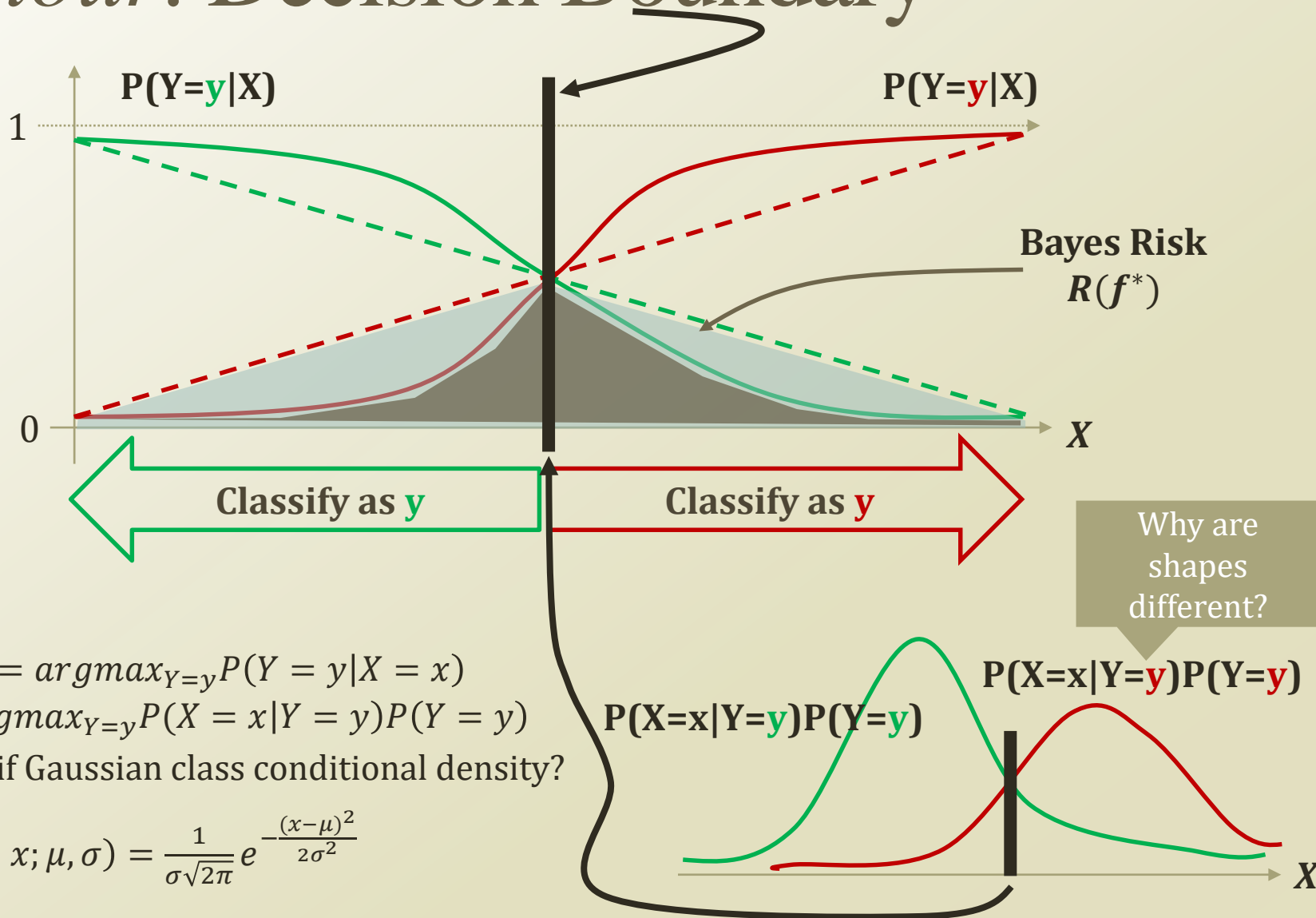
[icmoon@kaist.ac.kr](mailto:icmoon@kaist.ac.kr)

# Weekly Objectives

- Learn the support vector machine classifier
  - Understand the maximum margin idea of the SVM
  - Understand the formulation of the optimization problem
- Learn the soft-margin and penalization
  - Know how to add the penalization term
  - Understand the difference between the log-loss and the hinge-loss
- Learn the kernel trick
  - Understand the primal problem and the dual problem of SVM
  - Know the types of kernels
  - Understand how to apply the kernel trick to SVM and logistic regression

# SUPPORT VECTOR MACHINE

# Detour: Decision Boundary

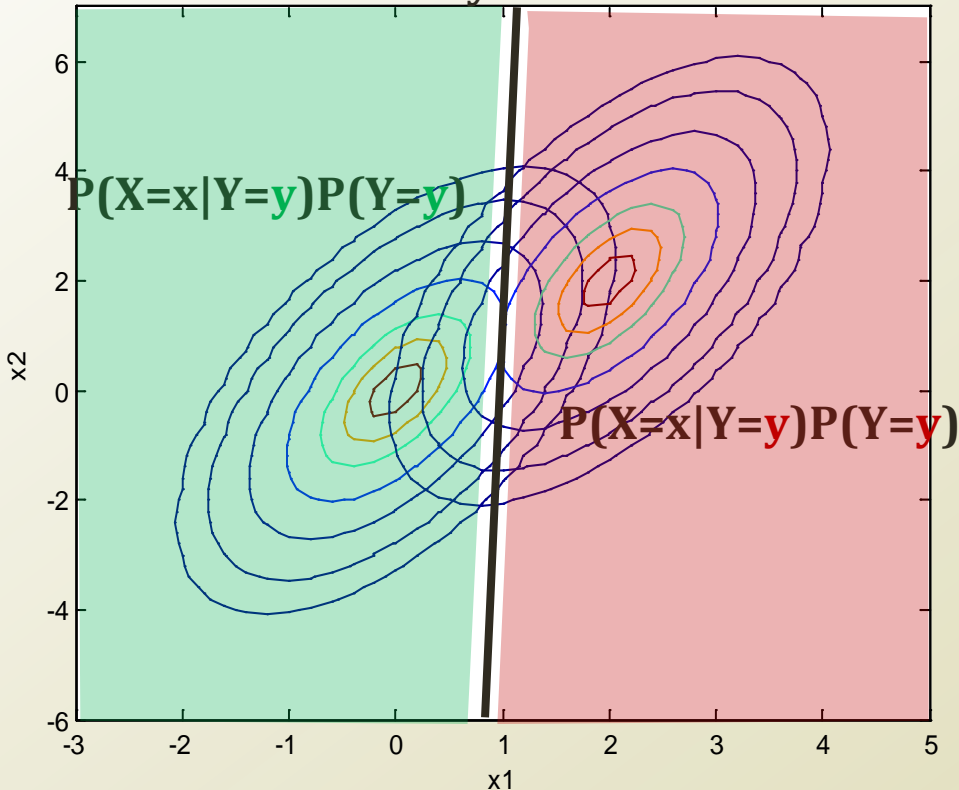


- $f^*(x) = \operatorname{argmax}_{Y=y} P(Y = y|X = x)$   
 $= \operatorname{argmax}_{Y=y} P(X = x|Y = y)P(Y = y)$
- What-if Gaussian class conditional density?

- $P(X = x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

# Detour: Decision Boundary in Two Dimension

Decision Boundary in Two Dimensions



$$f^*(x) = \operatorname{argmax}_{Y=y} P(Y = y|X = x) \\ = \operatorname{argmax}_{Y=y} P(X = x|Y = y)P(Y = y)$$

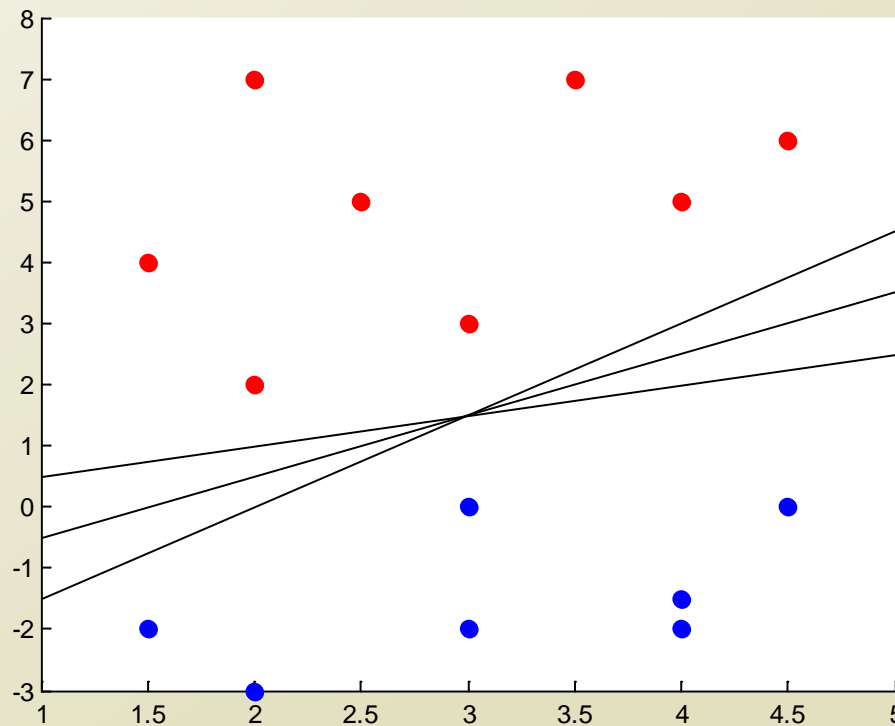
- Two multivariate normal distribution for the class conditional densities
- Decision boundary
  - A linear line
- Linear decision boundary
- Any problem in the real world applications?
  - Observing the combination of  $x_1$  and  $x_2$

$$P(X = x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

$$\longrightarrow P(X = (x_1, x_2)|Y = y) = \frac{1}{\sqrt{2\pi|\Sigma_y|}} \exp\left(-\frac{(x - \mu_y)\Sigma_y^{-1}(x - \mu_y)'}{2}\right)$$

# Decision Boundary without Prob.

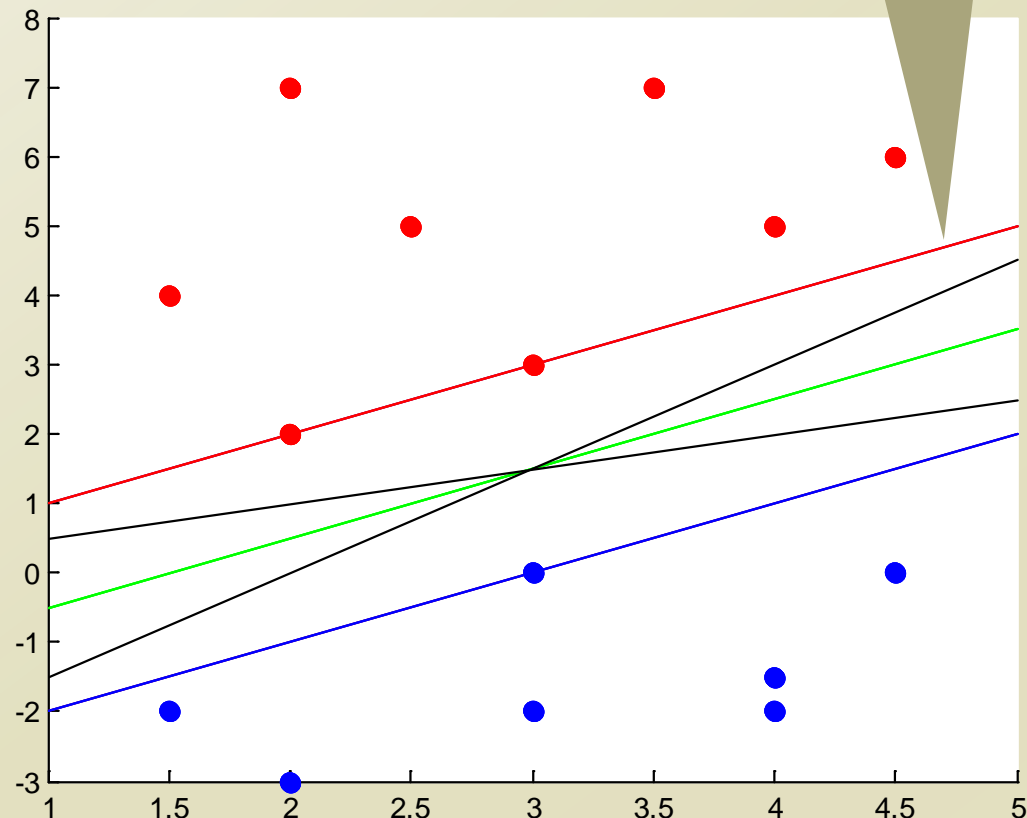
- Which is a better decision boundary?
  - Without considering the probability distribution?
- Which points are at the front line?



# Decision Boundary with Margin

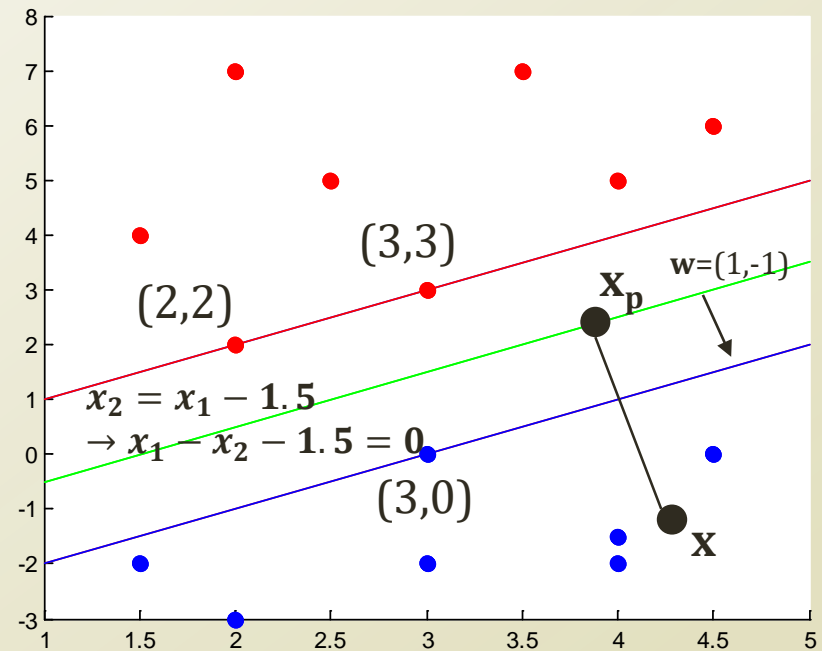
- Decision boundary with maximum margin
  - Between the points close to the boundary
  - How many points?
- Decision boundary line
  - $\mathbf{w} \cdot \mathbf{x} + b = 0$
  - Positive case
    - $\mathbf{w} \cdot \mathbf{x} + b > 0$
  - Negative case
    - $\mathbf{w} \cdot \mathbf{x} + b < 0$
  - Confidence level
    - $(\mathbf{w} \cdot \mathbf{x}_j + b)y_j$
- Margin?
  - Perpendicular distance from the closest point to the decision boundary

How many parameters?



# Margin Distance

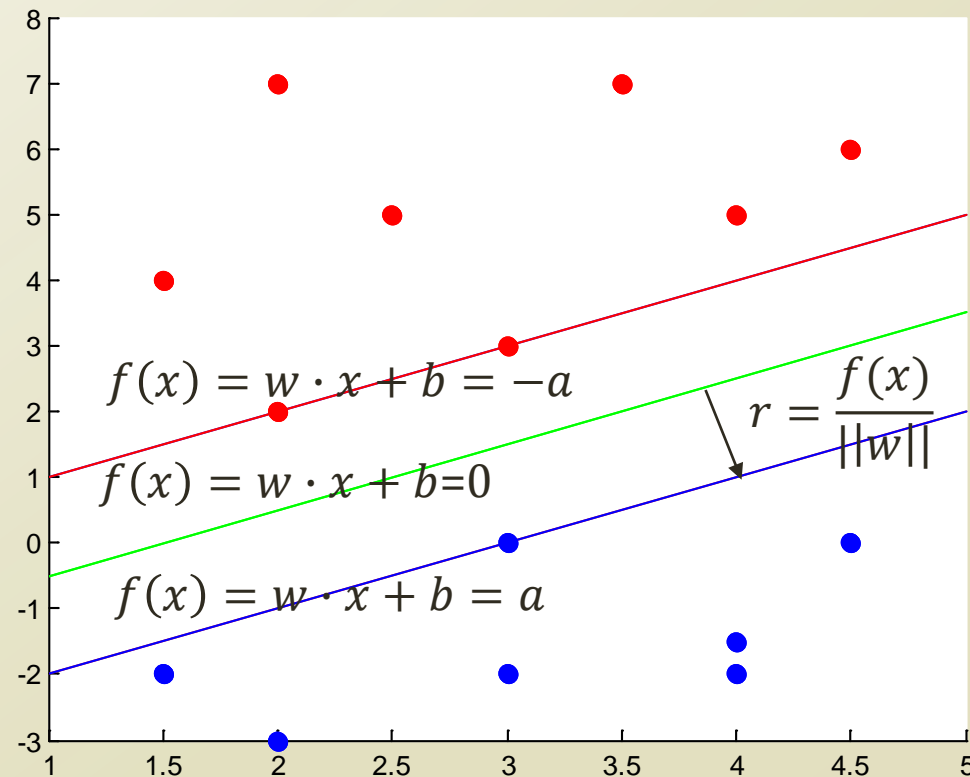
- Let's say
  - $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$
  - A point  $\mathbf{x}$  on the boundary has
    - $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b = 0$
  - A positive point  $\mathbf{x}$  has
    - $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b = a, a > 0$
- We are going to measure the distance
  - between an arbitrary point  $\mathbf{x}$  and a point  $\mathbf{x}_p$  on the boundary and on the perpendicular line from  $\mathbf{x}$  to the boundary
  - $x = x_p + r \frac{w}{||w||}, f(x_p) = 0$
  - $f(x) = w \cdot x + b = w \left( x_p + r \frac{w}{||w||} \right) + b = \underline{wx_p + b} + r \frac{w \cdot w}{||w||} = r ||w||$
- The distance is  $r = \frac{f(x)}{||w||}$





# Maximizing the Margin

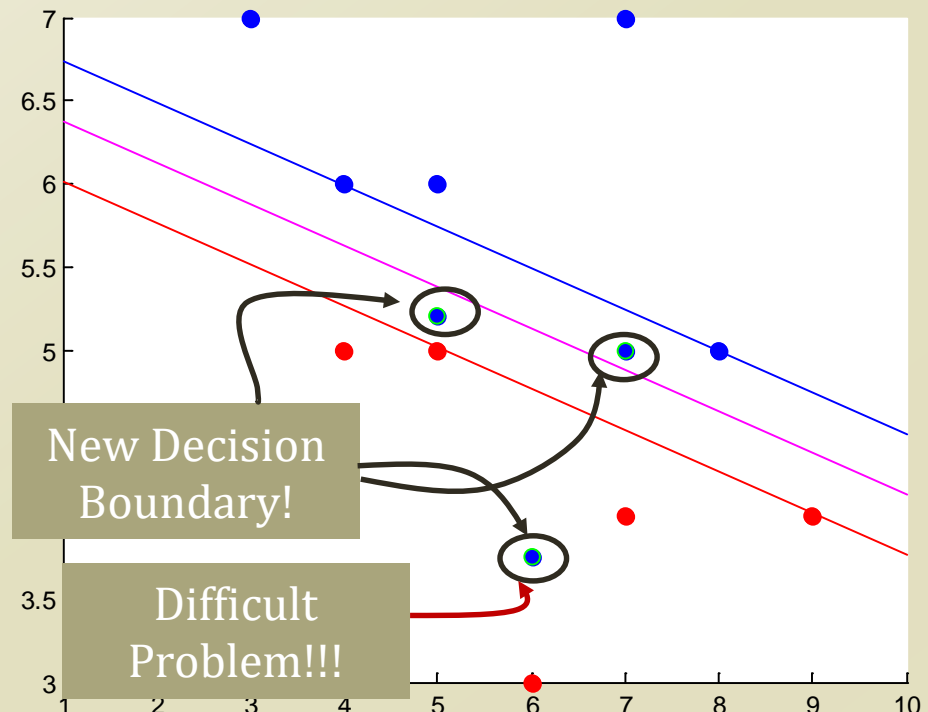
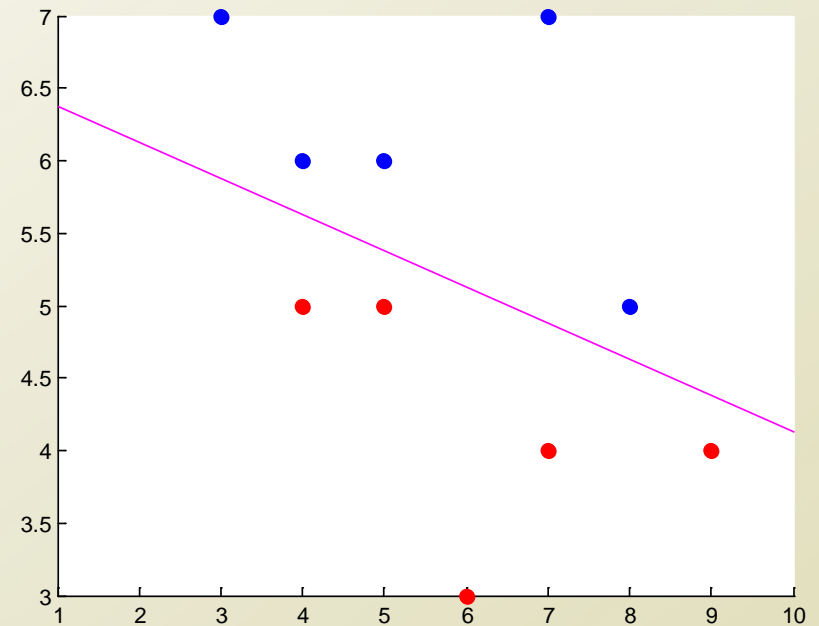
- Good decision boundary?
  - Maximum margin!
  - $r = \frac{a}{||w||}$
  - Need to consider the both side
- Optimization problem?
  - $\max_{w,b} 2r = \frac{2a}{||w||}$   
 $s.t. (wx_j + b)y_j \geq a, \forall j$
- $a$  is an arbitrary number and can be normalized
  - $\min_{w,b} ||w||$   
 $s.t. (wx_j + b)y_j \geq 1, \forall j$



This becomes a quadratic optimization problem. Why?

# Support Vector Machine with Hard Margin

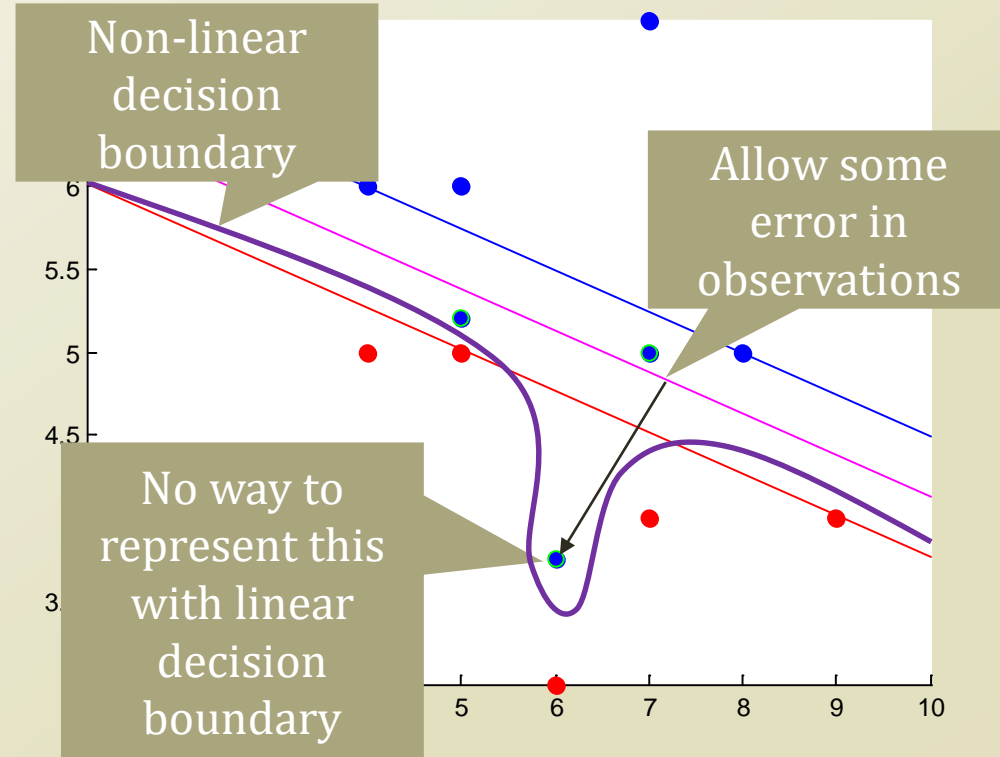
- Support Vector Machine (SVM)
  - Constructs a set of hyperplanes to have the largest distance to the nearest training data point of any class.
- Hard margin
  - No error cases are allowed
  - What If there is an error case?
- Let's implement the hard margin SVM
  - $\min_{w,b} ||w||$   
 $s.t. (wx_j + b)y_j \geq 1, \forall j$



# SOFT MARGIN

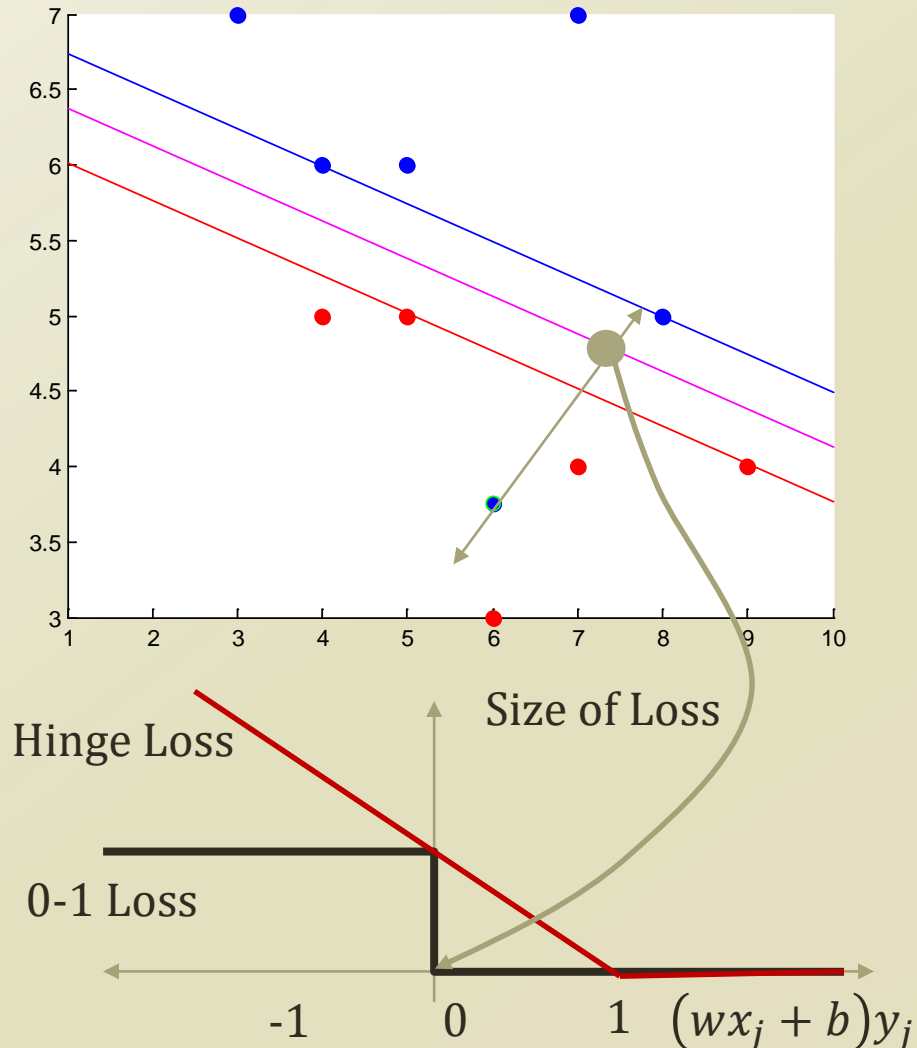
# “Error” Cases in SVM

- Data points that are
  - Impossible to classify with a linear decision boundary
- So called, “error” cases...
- How to manage these?
  - Option 1
    - Make decision boundary more complex
    - Go to non-linear
    - Any problem?
  - Option 2
    - Admit there will be an “error”
    - Represent the error in our problem formulation.
    - Try to reduce the error as well.
    - Any problem?



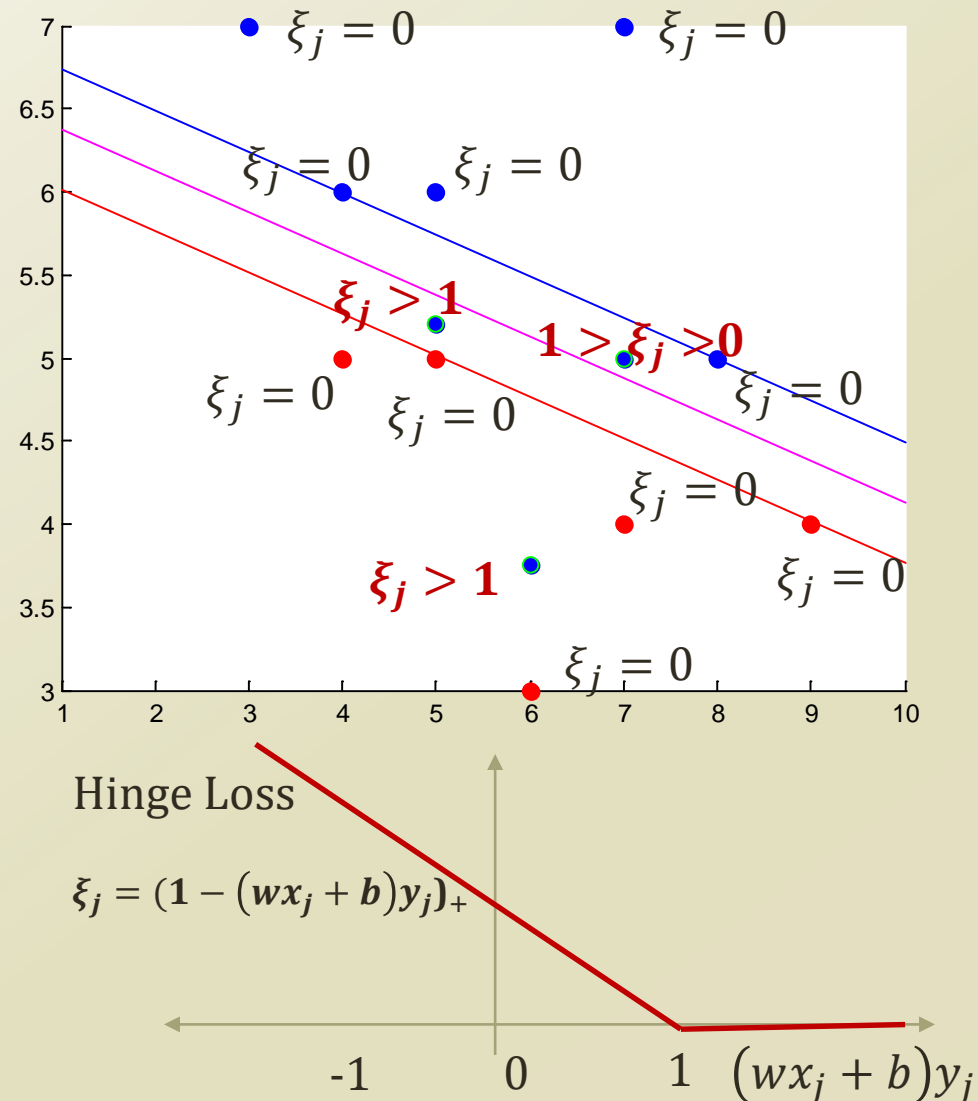
# “Error” Handling in SVM

- How to handle
- Option 1)
  - Counting the error cases and reduce the counts
  - $\min_{w,b} ||w|| + C \times \#_{error}$   
 $s.t. (wx_j + b)y_j \geq 1, \forall j$
  - Any problem?
- Option 2)
  - Introduce a slack variable
    - $\xi_j > 1$  when mis-classified
  - $\min_{w,b} ||w|| + C \sum_j \xi_j$   
 $s.t. (wx_j + b)y_j \geq 1 - \xi_j, \forall j$   
 $\xi_j \geq 0, \forall j$
  - Any problem?
- $C$  = trade-off parameter



# Soft-Margin SVM

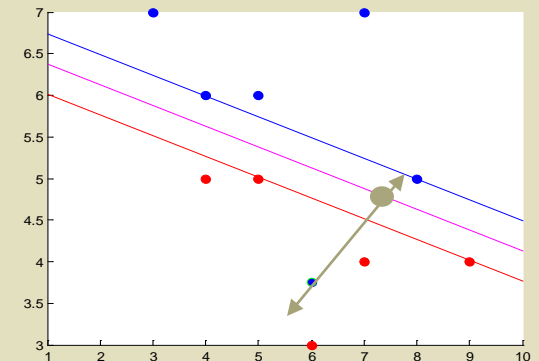
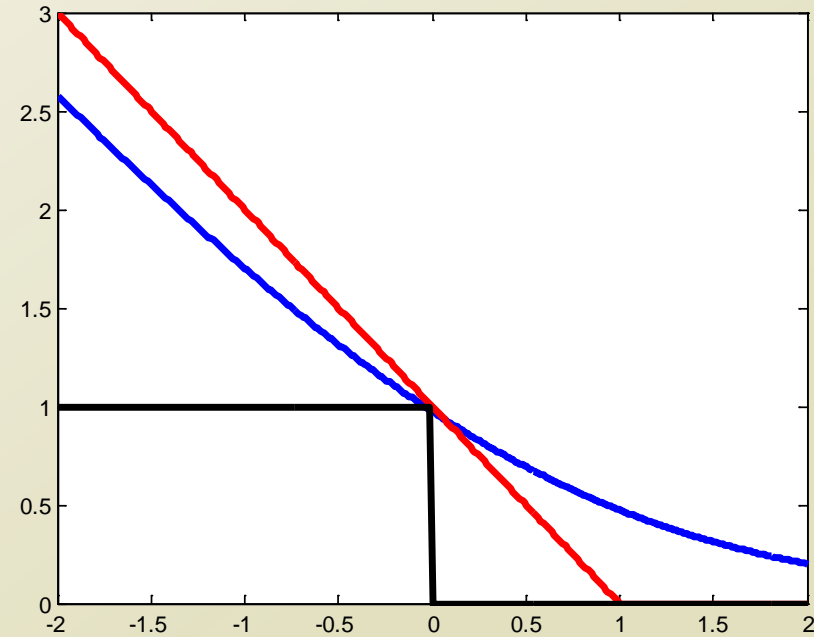
- $\min_{w,b} ||w|| + C \sum_j \xi_j$   
 s. t.  
 $(wx_j + b)y_j \geq 1 - \xi_j, \forall j$   
 $\xi_j \geq 0, \forall j$
- We soften the constraints
  - By adding a slack variable
- Instead, we penalize the misclassification cases in the objective function
  - $C \sum_j \xi_j$
- How to recover the hard-margin SVM?



# Comparison to Logistic Regression

- Loss function
  - $\xi_j = \text{loss}(f(x_j), y_j)$
- SVM loss function: Hinge Loss
  - $\xi_j = (1 - (wx_j + b)y_j)_+$
- Logistic Regression  
loss function: Log Loss
  - $\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{1 \leq i \leq N} \log(P(Y_i | X_i; \theta))$   

$$= \underset{\theta}{\operatorname{argmax}} \sum_{1 \leq i \leq N} \{Y_i X_i \theta - \log(1 + e^{X_i \theta})\}$$
  - $\xi_j = -\log(P(Y_j | X_j, w, b)) = \log(1 + e^{(wx_j + b)y_j})$
- Which loss function is preferable?
  - Around the decision boundary?
  - Overall place?



# Strength of the Loss Function

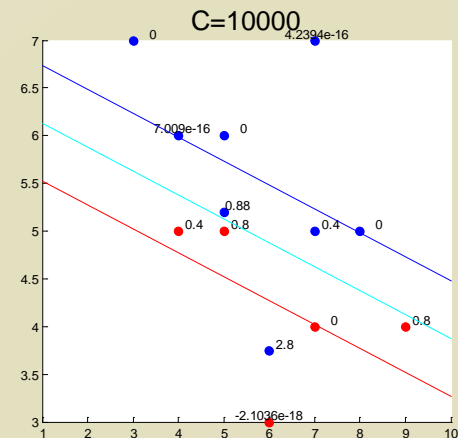
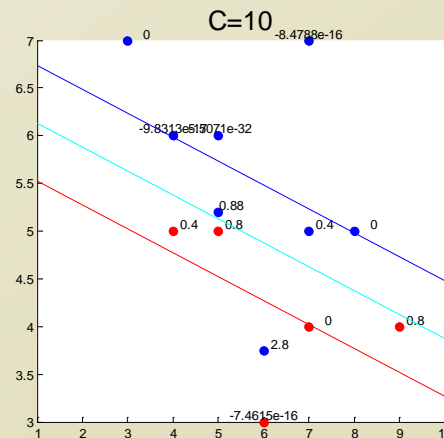
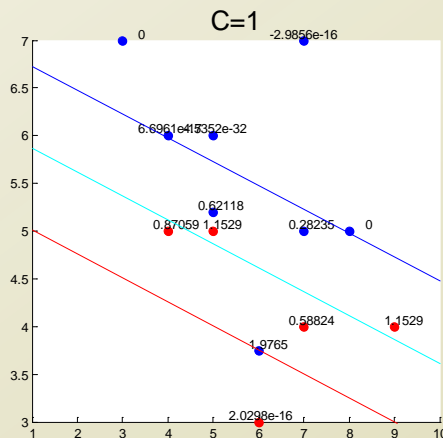
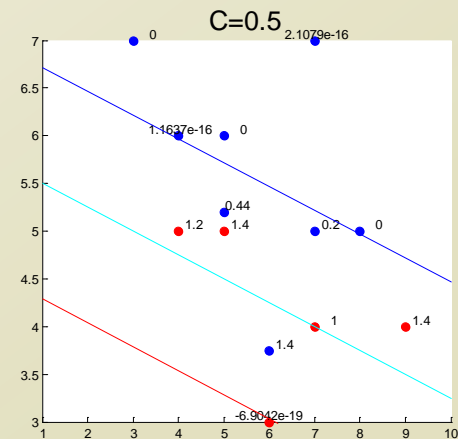
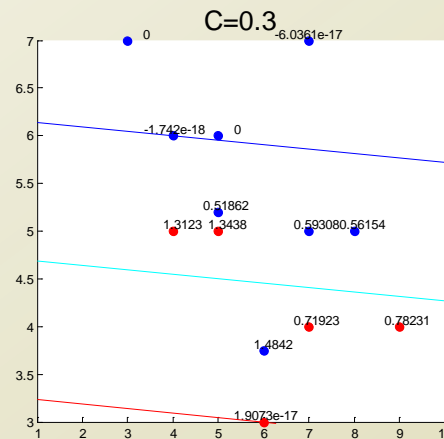
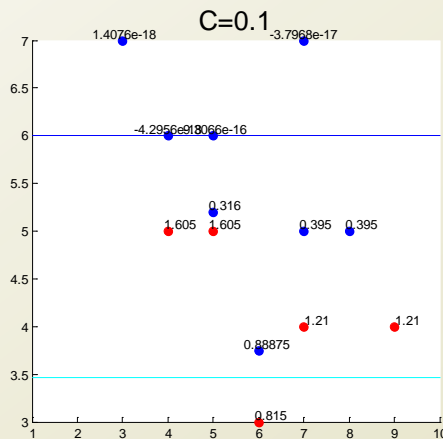
- $\min_{w,b,\xi_j} ||w|| + C \sum_j \xi_j$

s. t.

$$(wx_j + b)y_j \geq 1 - \xi_j, \forall j$$

$$\xi_j \geq 0, \forall j$$

- Let's implement the model
- How does the decision boundary evolve over the variations of C?

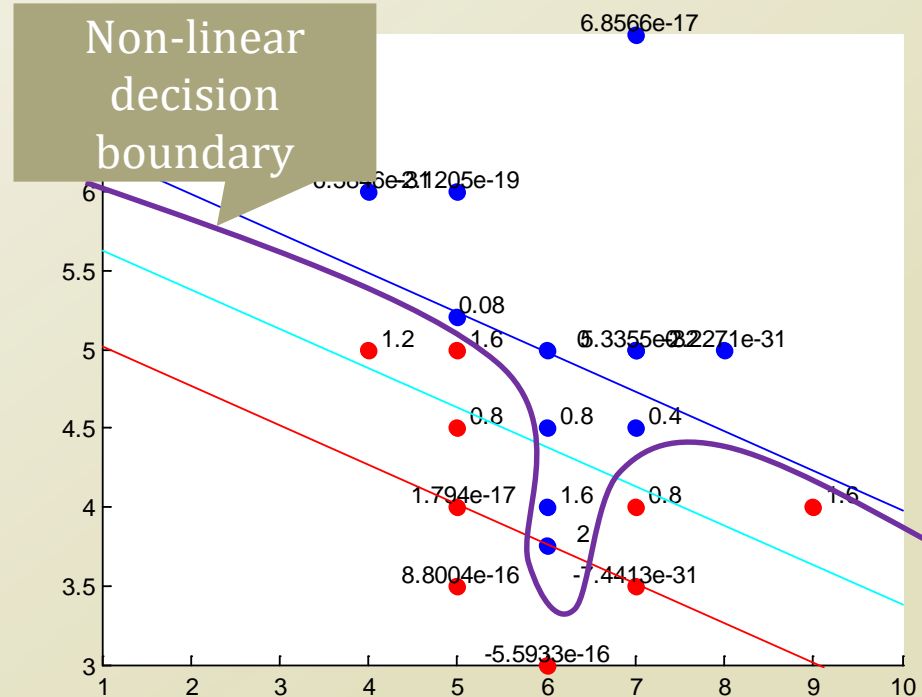




# KERNEL TRICK

# Enough of Studying SVM?

- You can train the SVM when you even have “error” cases
  - Use a soft-margin to handle such errors
- However, this does not change the complexity of the decision boundary
- In the real world, there are situations which require complex decision boundary...
  - Option 1
    - Make decision boundary more complex
    - Go to non-linear
  - Option 2
    - Admit there will be an “error”
    - Represent the error in our problem formulation.



# Feature Mapping to Expand Dim.

- $$\min_{w,b,\xi_j} ||w|| + C \sum_j \xi_j$$

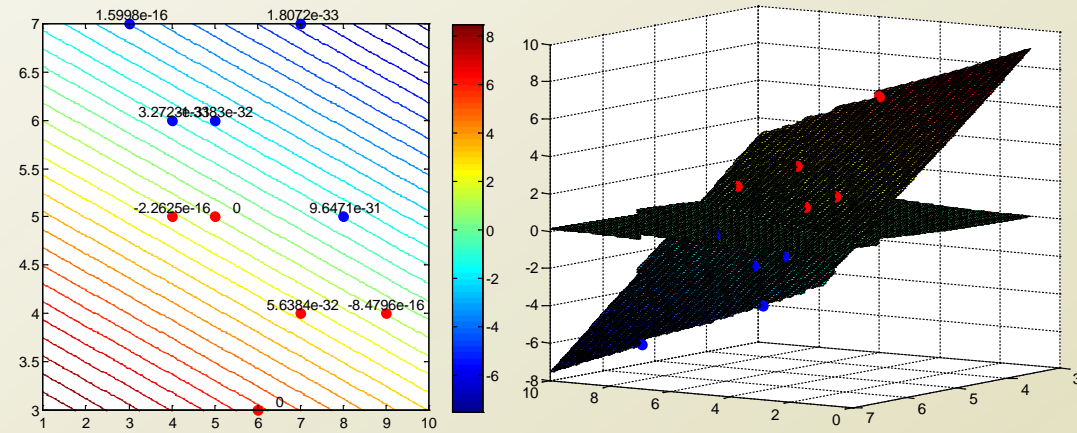
$$s.t.$$

$$(w\phi(x_j) + b)y_j \geq 1 - \xi_j, \forall j$$

$$\xi_j \geq 0, \forall j$$

- $$\phi(< x_1, x_2 >) =$$

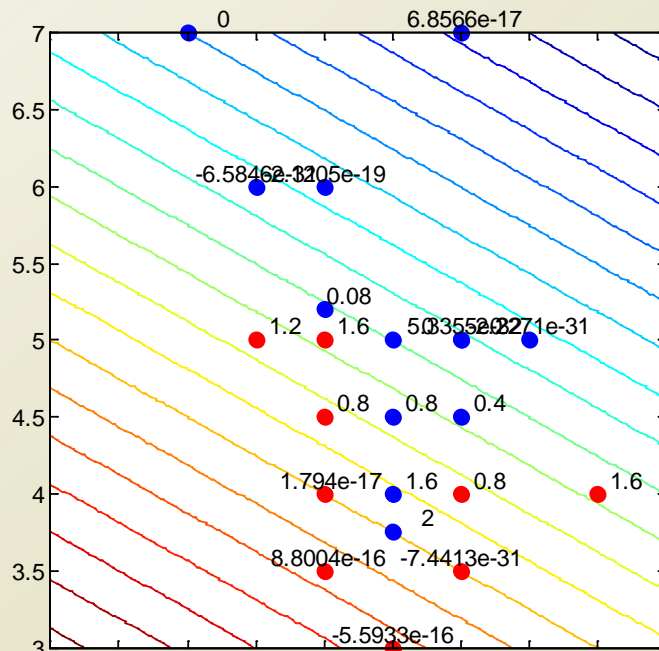
$$< x_1, x_2, x_1^2, x_2^2, x_1x_2, x_1^3, x_2^3, x_1^2x_2, x_1x_2^2 >$$



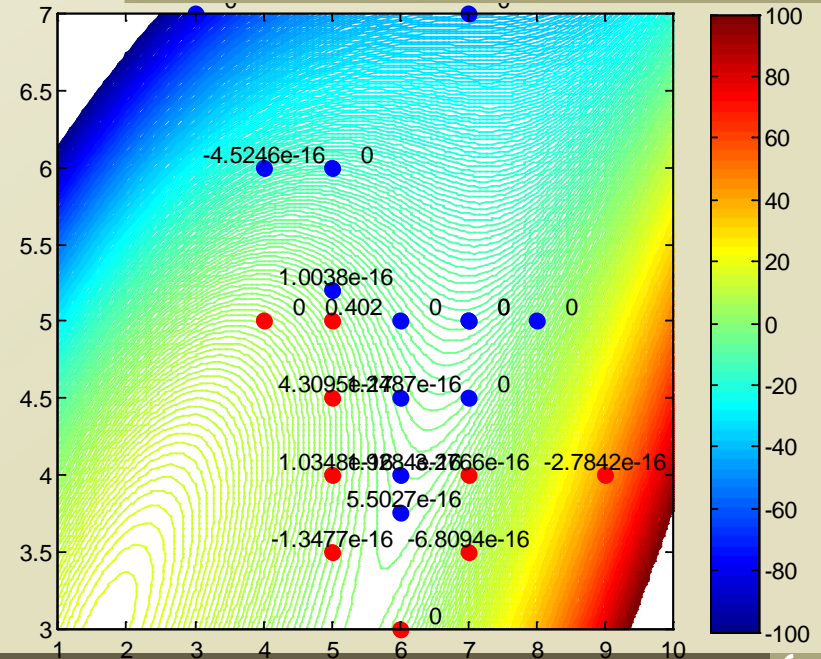
Linearly Separable Dataset

Any problem???

# of Params, Representation, Computation....



Linear Decision Boundary on



Industrial and Systems Engineering, KAIST

# Rethinking the Formulation


- SVM turns
  - Classification  $\rightarrow$  Constrained quadratic programming
- Constrained optimization
  - $\min_x f(x)$
  - $s.t. \ g(x) \leq 0, h(x) = 0$
- Lagrange method
  - Lagrange Prime Function:  $L(x, \alpha, \beta) = f(x) + \alpha g(x) + \beta h(x)$
  - Lagrange Multiplier:  $\alpha \geq 0, \beta$
  - Lagrange Dual Function:  $d(\alpha, \beta) = \inf_{x \in X} L(x, \alpha, \beta) = \min_x L(x, \alpha, \beta)$
  - $\max_{\alpha \geq 0, \beta} L(x, \alpha, \beta) = \begin{cases} f(x): \text{if } x \text{ is feasible} \\ \infty: \text{otherwise} \end{cases}$
  - $\min_x f(x) \rightarrow \min_x \max_{\alpha \geq 0, \beta} L(x, \alpha, \beta)$
- Take advantage of the formulation technique of the constrained optimization
  - Primal and Dual Problems!

*inf*: infimum “Greatest Lower Bound”  
 $\inf\{1,2,3\} = 1$

# Primal and Dual Problem


## Primal Problem

$$\begin{aligned} \min_x & f(x) \\ \text{s.t. } & g(x) \leq 0, h(x) = 0 \end{aligned}$$


$$\min_x \max_{\alpha \geq 0, \beta} L(x, \alpha, \beta)$$

## Lagrange Dual Problem

$$\begin{aligned} \max_{\alpha > 0, \beta} & d(\alpha, \beta) \\ \text{s.t. } & \alpha > 0 \end{aligned}$$


$$\max_{\alpha \geq 0, \beta} \min_x L(x, \alpha, \beta)$$

- Weak duality theorem
  - $d(\alpha, \beta) \leq f(x^*)$  for  $\forall \alpha, \forall \beta$
  - $d^* = \max_{\alpha \geq 0, \beta} \min_x L(x, \alpha, \beta) \leq \min_x \max_{\alpha \geq 0, \beta} L(x, \alpha, \beta) = p^*$ 
    - Maximizing the dual function provides the lower bound of  $f(x^*)$
  - Duality gap =  $f(x^*) - d(\alpha^*, \beta^*)$
- Strong duality
  - $d^* = \max_{\alpha \geq 0, \beta} \min_x L(x, \alpha, \beta) = \min_x \max_{\alpha \geq 0, \beta} L(x, \alpha, \beta) = p^*$
  - When Karush-Kuhn-Tucker (KKT) Conditions are satisfied

# KKT Condition and Strong Duality

- Strong duality

- $d^* = \max_{\alpha \geq 0, \beta} \min_x L(x, \alpha, \beta) = \min_x \max_{\alpha \geq 0, \beta} L(x, \alpha, \beta) = p^*$

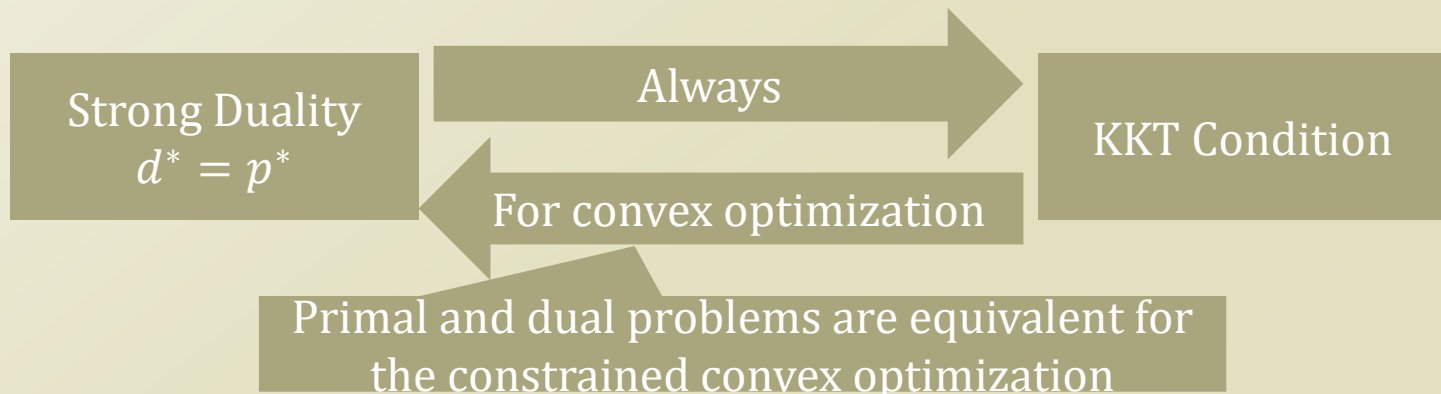
- Holds when KKT conditions are met

- $\nabla L(x^*, \alpha^*, \beta^*) = 0$
  - $\alpha^* \geq 0$
  - $g(x^*) \leq 0$
  - $h(x^*) = 0$
  - $\alpha^* g(x^*) = 0$

Active Constraint  
 $\alpha^* = 0 \Rightarrow g(x^*) = 0$   
 Inactive Constraint  
 $g(x^*) < 0 \Rightarrow \alpha^* = 0$   
 $\rightarrow$  Complementary Slackness

Primal Problem

$$\begin{aligned} \min_x & f(x) \\ \text{s.t. } & g(x) \leq 0, \\ & h(x) = 0 \end{aligned}$$



# Dual Problem of SVM

$$\begin{aligned} \min_x & f(x) \\ \text{s.t. } & g(x) \leq 0, \\ & h(x) = 0 \end{aligned}$$

## Lagrange Prime Function

$$L(x, \alpha, \beta) = f(x) + \alpha g(x) + \beta h(x)$$

## Primal Problem of Linearly Separable SVM

$$\begin{aligned} \min_{w,b} & ||w|| \\ \text{s.t. } & (wx_j + b)y_j \geq 1, \forall j \end{aligned}$$

$$\begin{aligned} \min_{w,b} \max_{\alpha \geq 0, \beta} & \frac{1}{2} w \cdot w - \sum_j \alpha_j [(wx_j + b)y_j - 1] \\ \text{s.t. } & \alpha_j \geq 0, \text{ for } \forall j \end{aligned}$$

- Linearly separable case
- Lagrange Prime Function

- $L(w, b, \alpha)$ 

$$= \frac{1}{2} w \cdot w - \sum_j \alpha_j [(wx_j + b)y_j - 1]$$

- Lagrange Multiplier
  - $\alpha_j \geq 0, \text{ for } \forall j$

## Dual Problem of Linearly Separable SVM

$$\begin{aligned} \max_{\alpha \geq 0} \min_{w,b} & \frac{1}{2} w \cdot w - \sum_j \alpha_j [(wx_j + b)y_j - 1] \\ \text{s.t. } & \alpha_j \geq 0, \text{ for } \forall j \end{aligned}$$

## KKT Condition to Eliminate the Duality Gap

$$\frac{\partial L(w,b,\alpha)}{\partial w} = 0, \frac{\partial L(w,b,\alpha)}{\partial b} = 0$$

$$\alpha_i \geq 0, \forall i$$

$$\alpha_i ((wx_j + b)y_j - 1) = 0, \forall i$$



# Dual Representation of SVM

- $L(w, b, \alpha) = \frac{1}{2}w \cdot w - \sum_j \alpha_j [(wx_j + b)y_j - 1]$ 
  - $= \frac{1}{2}ww - \sum_j \alpha_j y_j wx_j - b \sum_j \alpha_j y_j + \sum_j \alpha_j$
  - $= \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i x_j - \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i x_j - b \times 0 + \sum_j \alpha_j$
  - $= \sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i x_j$
- Again, a quadratic programming
- Once,  $\alpha_j$  is known
  - $w = \sum_{i=1}^N \alpha_i y_i x_i$
  - $\alpha_i ((wx_j + b)y_j - 1) = 0$
- Now, we can find out the  $w$  and  $b$  again.
  - Why is this better?
  - Most of  $\alpha_j$  are.....
  - Location of  $x$  is....
- Let's find out from the implementation...

## Dual Problem of Linearly Separable SVM

$$\begin{aligned} \max_{\alpha \geq 0} \min_{w, b} & \frac{1}{2}w \cdot w - \sum_j \alpha_j [(wx_j + b)y_j - 1] \\ \text{s. t. } & \alpha_j \geq 0, \text{ for } \forall j \end{aligned}$$

## KKT Condition to Eliminate the Duality Gap

$$\frac{\partial L(w, b, \alpha)}{\partial w} = 0, \frac{\partial L(w, b, \alpha)}{\partial b} = 0$$

$$\alpha_i \geq 0, \forall i$$

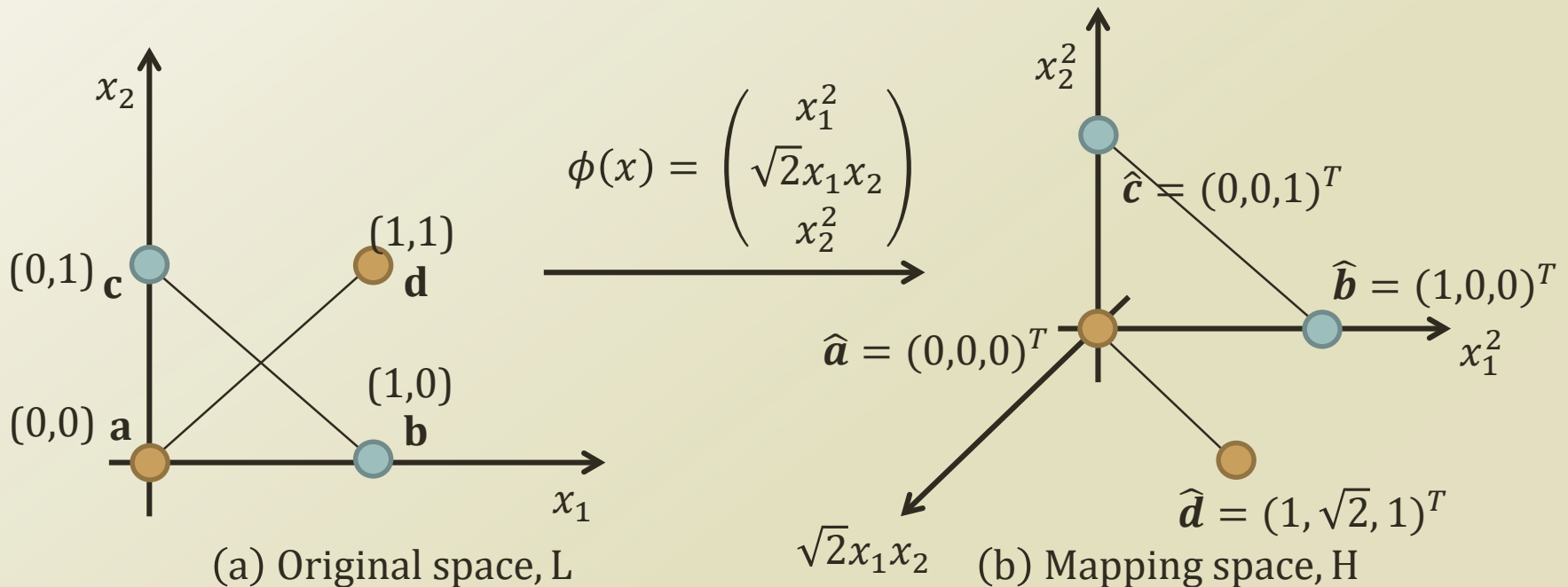
$$\alpha_i ((wx_j + b)y_j - 1) = 0, \forall i$$

$$w = \sum_{i=1}^N \alpha_i y_i x_i \qquad \sum_{i=1}^N \alpha_i y_i = 0$$



# Mapping Functions

- Suppose that there are non-linearly separable data sets...
- The non-linear separable case can be linearly separable when we increase the basis space
  - Standard basis:  $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \dots, \mathbf{e}_n \rightarrow$  Linearly independent and generate  $\mathbf{R}^n$
- Expanding the Basis through Space mapping function  $\phi : L \rightarrow H$ 
  - Or, transformation function, etc...
- Any problem????
  - Feature space becomes bigger and bigger....



# Kernel Function

- The kernel calculates the inner product of two vectors in a different space (preferably without explicitly representing the two vectors in the different space)
  - $K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$
- Some common kernels are following :
  - Polynomial(homogeneous)
    - $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^d$
  - Polynomial(inhomogeneous)
    - $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^d$
  - Gaussian kernel function, a.k.a. Radial Basis Function
    - $k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$ 
      - For  $\gamma > 0$ . Sometimes parameterized using  $\gamma = \frac{1}{2\sigma^2}$
  - Hyperbolic tangent, a.k.a. Sigmoid Function
    - $k(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\kappa \mathbf{x}_i \cdot \mathbf{x}_j + c)$
    - For some(not every)  $\kappa > 0$  and  $c < 0$

# Polynomial Kernel Function

- Imagine we have
  - $\mathbf{x} = \langle x_1, x_2 \rangle$  and  $\mathbf{z} = \langle z_1, z_2 \rangle$
  - Polynomial Kernel Function of degree 1
    - $K(\langle x_1, x_2 \rangle, \langle z_1, z_2 \rangle) = \langle x_1, x_2 \rangle \cdot \langle z_1, z_2 \rangle = x_1 z_1 + x_2 z_2 = \mathbf{x} \cdot \mathbf{z}$
  - Polynomial Kernel Function of degree 2
    - $K(\langle x_1, x_2 \rangle, \langle z_1, z_2 \rangle) = \langle x_1^2, \sqrt{2}x_1x_2, x_2^2 \rangle \cdot \langle z_1^2, \sqrt{2}z_1z_2, z_2^2 \rangle$
    - $= x_1^2 z_1^2 + 2x_1x_2z_1z_2 + x_2^2 z_2^2 = (x_1 z_1 + x_2 z_2)^2 = (\mathbf{x} \cdot \mathbf{z})^2$
  - Polynomial Kernel Function of degree 3
    - $K(\langle x_1, x_2 \rangle, \langle z_1, z_2 \rangle) = (\mathbf{x} \cdot \mathbf{z})^3$
  - Polynomial Kernel Function of degree  $n$ 
    - $K(\langle x_1, x_2 \rangle, \langle z_1, z_2 \rangle) = (\mathbf{x} \cdot \mathbf{z})^n$
- Do we need to express and calculate the transformed coordinate values for  $\mathbf{x}$  and  $\mathbf{z}$  to know the polynomial kernel of  $K$ ?
  - Do we need to convert the feature spaces to exploit the linear separation in the high order?
  - **Condition: only the inner product is computable with this trick**

# Dual SVM with Kernel Trick

- $\max_{\alpha \geq 0} \sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \varphi(x_i) \varphi(x_j)$
- $\max_{\alpha \geq 0} \sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j K(x_i, x_j)$ 
  - $\alpha_i \left( (w x_j + b) y_j - 1 \right) = 0, C > \alpha_i > 0$ 
    - $w = \sum_{i=1}^N \alpha_i y_i \varphi(x_i)$
    - $b = y_j - \sum_{i=1}^N \alpha_i y_i \varphi(x_i) \varphi(x_j)$
  - $\sum_{i=1}^N \alpha_i y_i = 0$
  - $C \geq \alpha_i \geq 0, \forall i$
- Dual formulation lets SVM utilize
  - Kernel trick
    - Reduced parameters to estimate
  - Only store alpha values instead of w
    - How many alpha values are needed?
    - Consider meaningful alphas

Dual Problem of Linearly Separable SVM

$$\max_{\alpha \geq 0} \sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i x_j$$

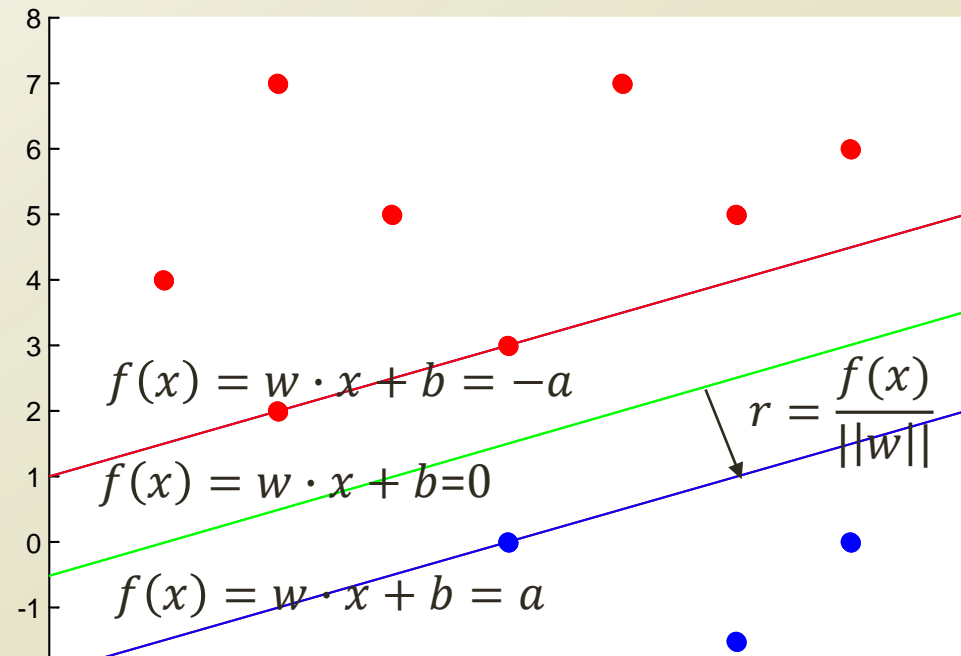
$$C \geq \alpha_i \geq 0, \forall i$$

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

$$\alpha_i \left( (w x_j + b) y_j - 1 \right) = 0, C > \alpha_i > 0$$

# Classification with SVM Kernel Trick

- Linear case
  - $\text{sign}(w \cdot x + b)$
  - $\min_{w,b} ||w||$ 
    - $(wx_j + b)y_j \geq 1, \forall j$
- Transformed case
  - $\text{sign}(w \cdot \varphi(x) + b)$
  - $\min_{w,b,\xi_j} ||w|| + C \sum_j \xi_j$ 
    - $(w\varphi(x_j) + b)y_j \geq 1 - \xi_j, \forall j$
    - $\xi_j \geq 0, \forall j$
- Kernel trick case
  - $\text{sign}(w \cdot \varphi(x) + b)$
  - $\max_{\alpha \geq 0} \sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j K(x_i, x_j)$ 
    - $w = \sum_{i=1}^N \alpha_i y_i \varphi(x_i)$
    - $b = y_j - w\varphi(x_j)$  when  $0 < \alpha_j < C$
    - $\sum_{i=1}^N \alpha_i y_i = 0$
    - $0 \leq \alpha_i \leq C, \forall i$

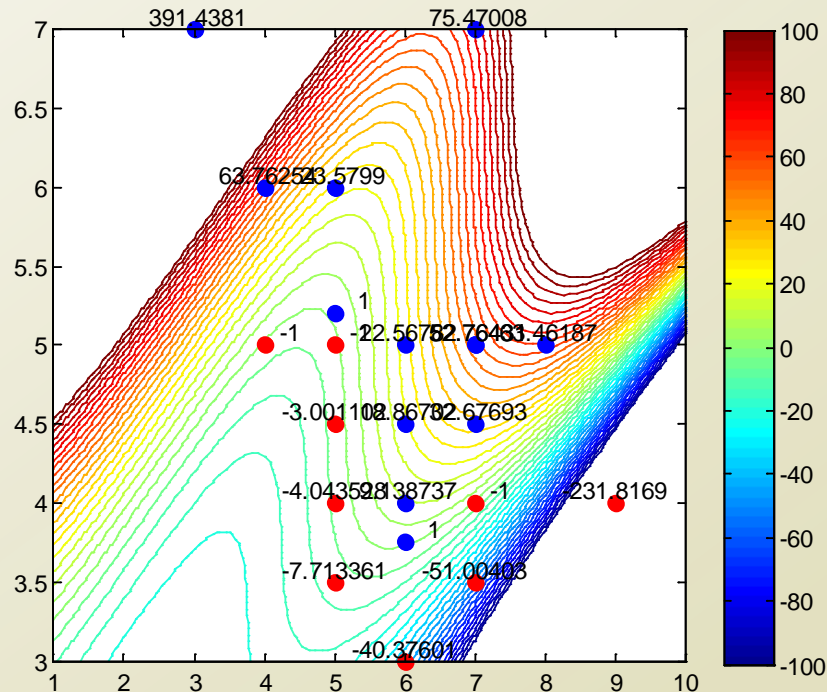


$$\begin{aligned} \text{sign}(w \cdot \varphi(x) + b) &= \text{sign} \left( \sum_{i=1}^N \alpha_i y_i \varphi(x_i) \cdot \varphi(x) + y_j - \sum_{i=1}^N \alpha_i y_i \varphi(x_i) \varphi(x_j) \right) \\ &= \text{sign} \left( \sum_{i=1}^N \alpha_i y_i K(x_i, x) + y_j - \sum_{i=1}^N \alpha_i y_i K(x_i, x_j) \right) \\ & \quad 0 < \alpha_j < C \end{aligned}$$

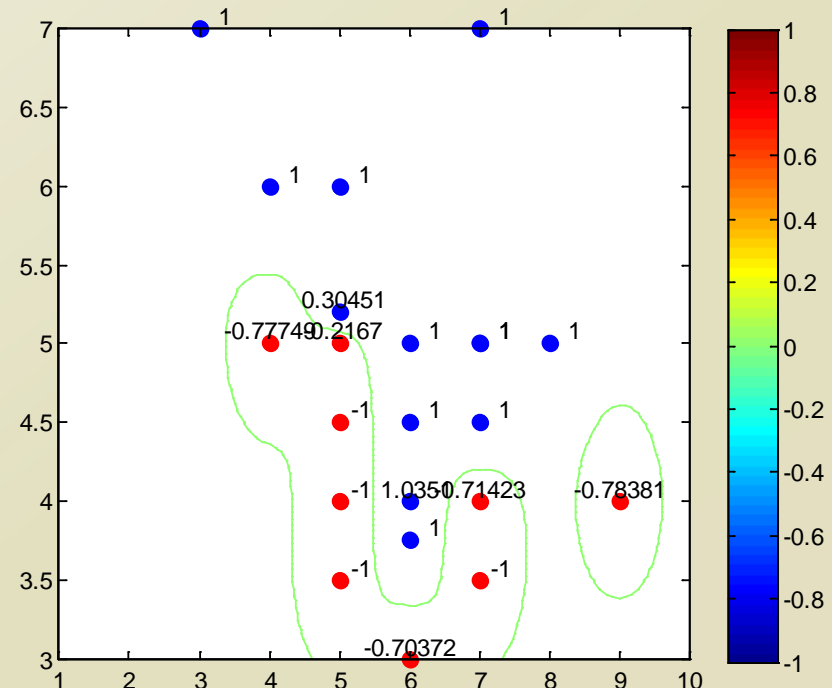
# SVM with Various Kernels

- SVM is very adaptable to the non-linearly separable cases with the kernel trick
  - Easy expand to the high dimension features (for free!)

## Polynomial Kernel with Degree 4



## RBF Kernel with $\gamma=4$



# Logistic Regression with Kernel

- Logistic regression

- $P(Y|X) = \frac{1}{1+e^{-\theta^T x}}$

- Finding the MLE of  $\theta$

- Can we kernelize the logistic regression?

- $\mathbf{w} = \sum_{i=1}^N \alpha_i y_i \varphi(\mathbf{x}_i)$

- $P(Y|X) = \frac{1}{1+e^{-\theta^T x}} = \frac{1}{1+e^{\sum_{i=1}^N \alpha_i y_i \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}) + b}} = \frac{1}{1+e^{\sum_{i=1}^N \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b}}$

- Problem changes

- From finding  $\theta$  to finding  $\alpha_i$
  - How to solve this problem?
    - In other words...
    - Is this a constrained optimization?
    - If not, what does it imply?

# Acknowledgement

- This slideset is greatly influenced
  - By Prof. Carlos Guestrin at CMU
  - By Prof. Eric Xing at CMU



# Further Readings

- Bishop Chapter 7 → 6