Support Vector Machine

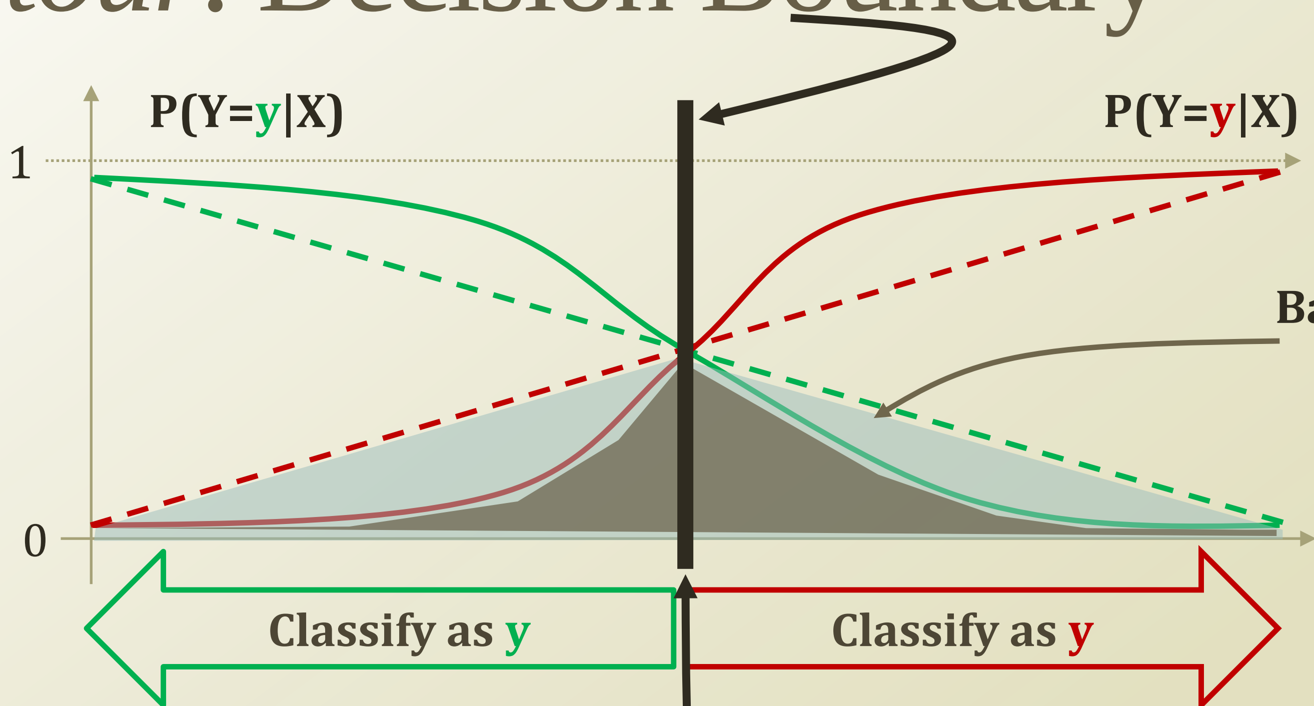
Il-Chul Moon
Dept. of Industrial and Systems Engineering
KAIST
icmoon@kaist.ac.kr

Weekly Objectives

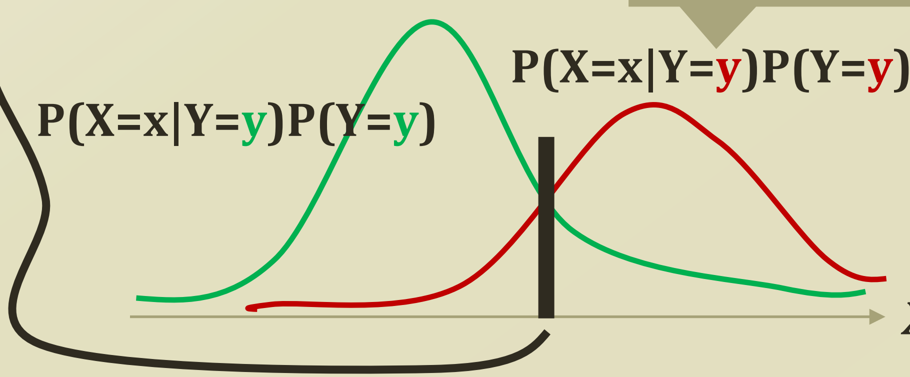
- Learn the support vector machine classifier
  - Understand the maximum margin idea of the SVM
  - Understand the formulation of the optimization problem
- Learn the soft-margin and penalization
  - Know how to add the penalization term
  - Understand the difference between the log-loss and the hinge-loss
- 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

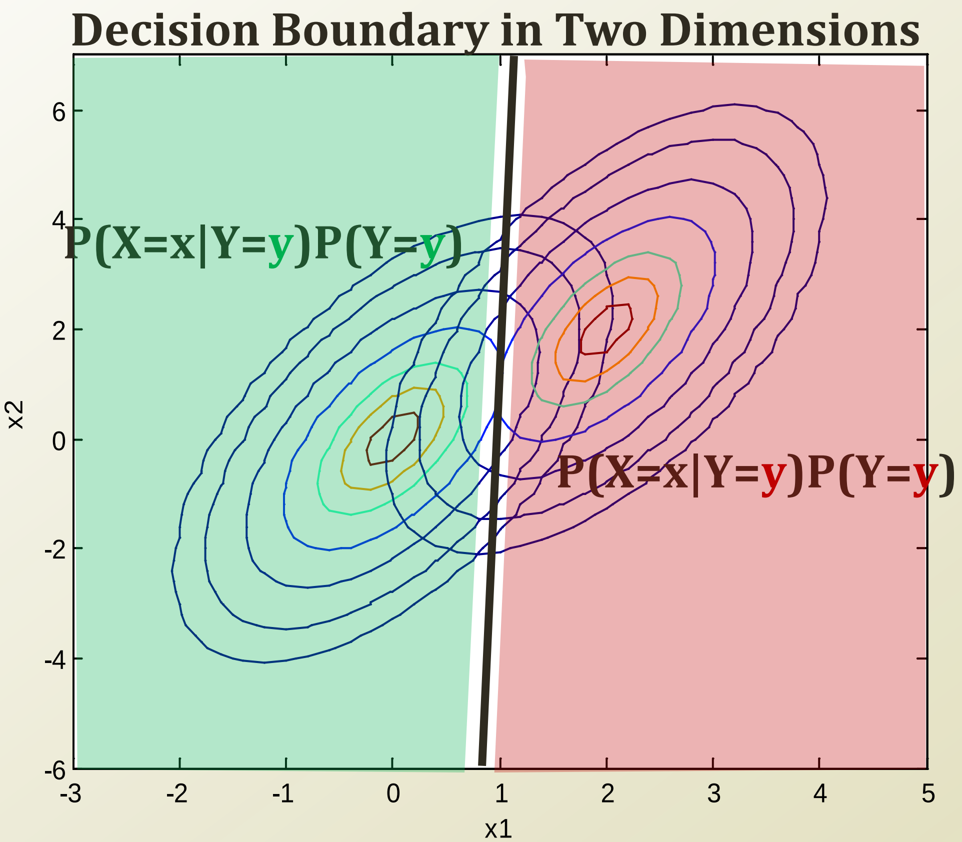


Why are shapes different?



- $f^*(x) = argmax_{Y=y}P(Y=y|X=x)$
  $= argmax_{Y=y}P(X=x|Y=y)P(Y=y)$
- 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



$$f^*(x) = argmax_{Y=y}P(Y = y|X = x)$$
$$= argmax_{Y=y}P(X = x|Y = y)P(Y = y)$$

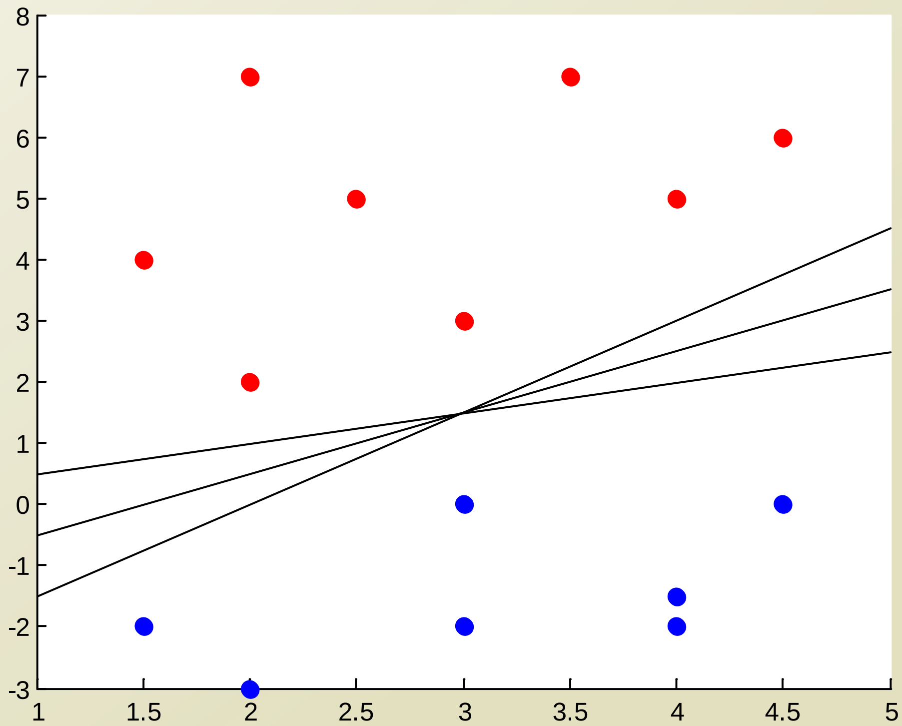
- Two multivariate normal distribution for the class conditional densities
- Decision boundary
  - A linear line
- Linear decision boundary
- Any problem in the real world applications?
  - Observing the combination of x_1 and x_2

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

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

Decision Boundary without Prob.

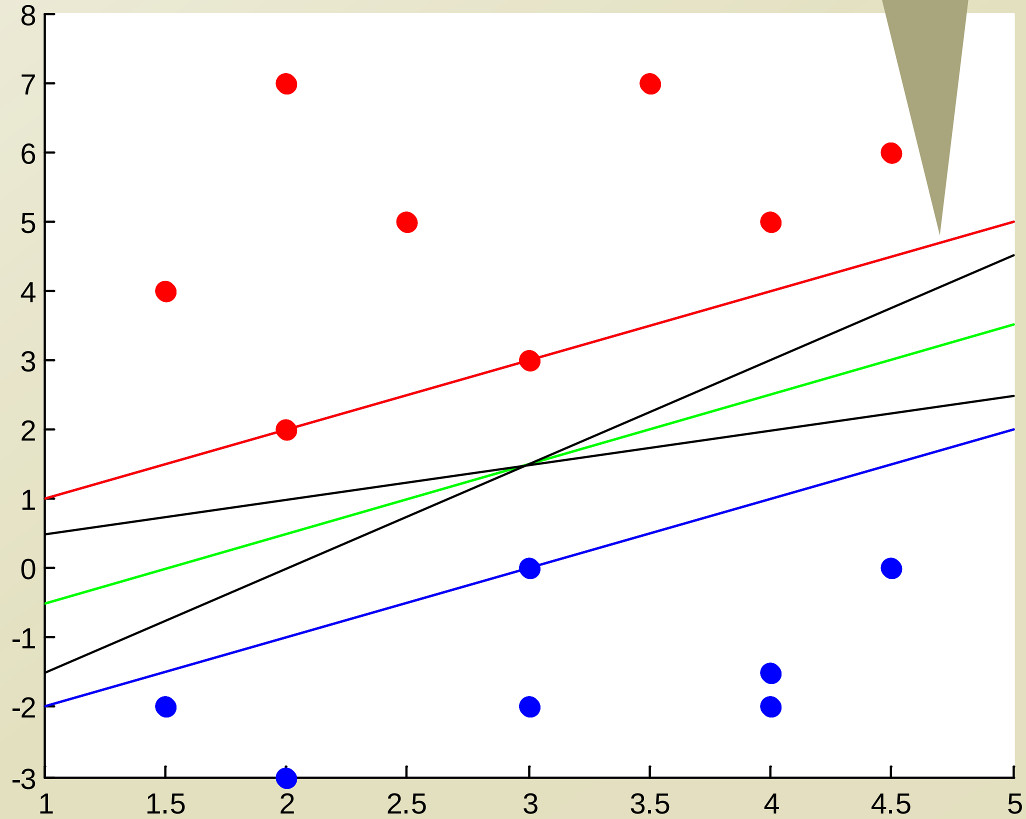
- Which is a better decision boundary?
  - Without considering the probability distribution?
- Which points are at the front line?



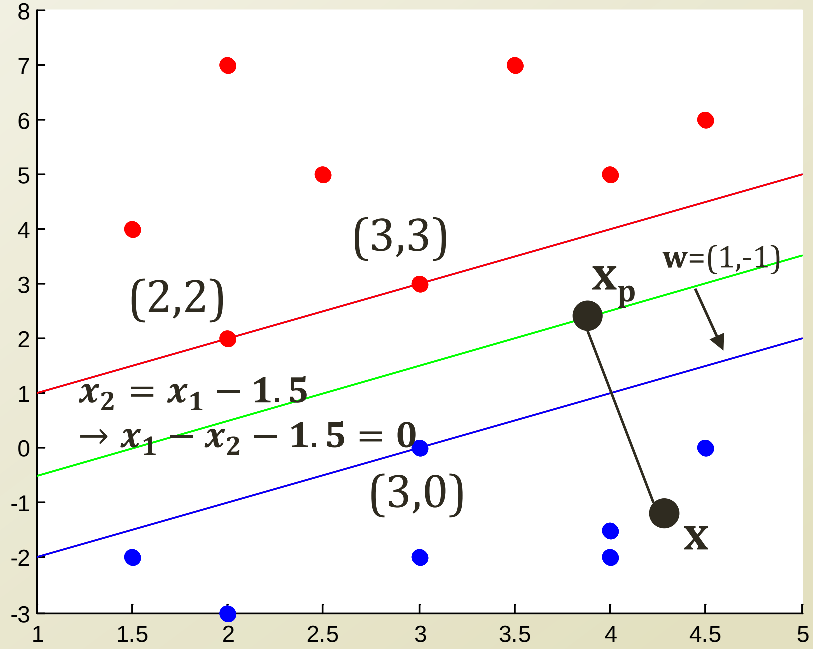
Decision Boundary with Margin

- 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?



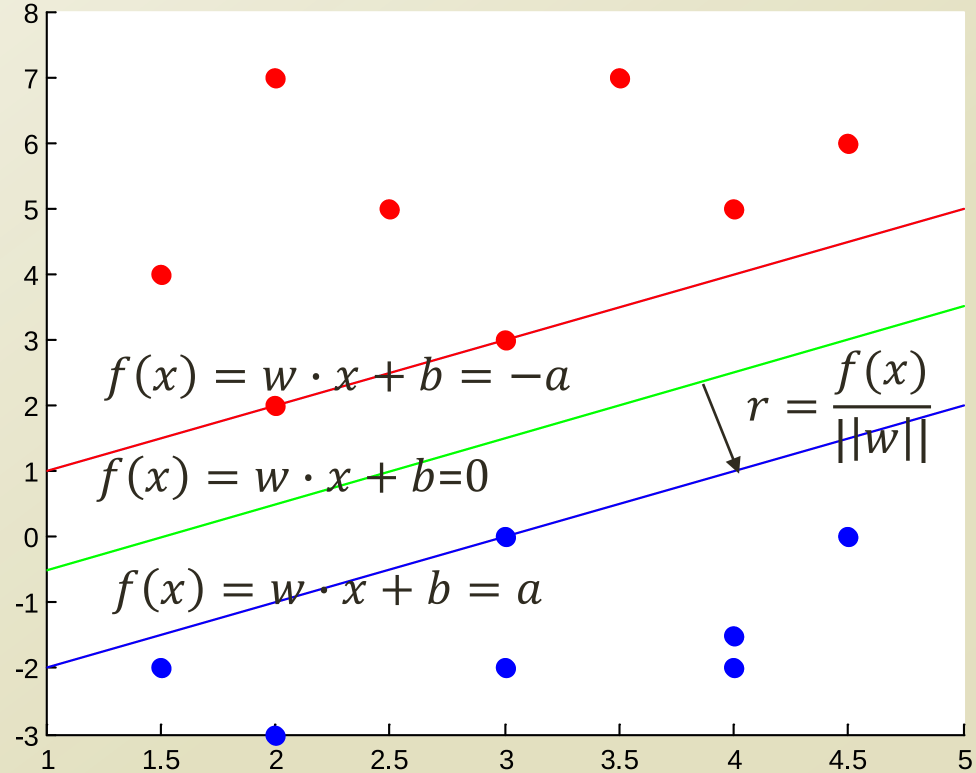
Margin Distance



- Let's say
  - $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$
  - A point **x** on the boundary has
    - $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b = 0$
  - A positive point **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 **x** and a point $\mathbf{x_p}$ on the boundary and on the perpendicular line from **x** to the boundary
  - $x = x_p + r\frac{w}{||w||}, f(x_p) = 0$
  - $f(x) = w \cdot x + b = w\left(x_p + r\frac{w}{||w||}\right) + b = \overline{wx_p + b} + r\frac{w \cdot w}{||w||} = r||w||$
- The distance is $r = \frac{f(x)}{||w||}$

Maximizing the Margin

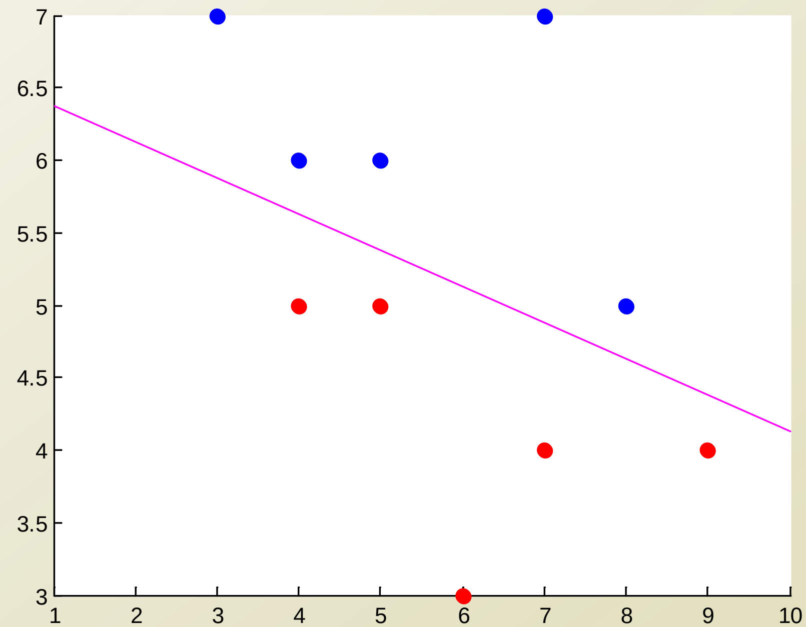
- Good decision boundary?
  - Maximum margin!
  - $r = \frac{a}{||w||}$
  - Need to consider the both side
- Optimization problem?
  - $max_{w,b} 2r = \frac{2a}{||w||}$
    $s.t.(wx_j + b)y_j \geq a, \forall j$
- a is an arbitrary number and can be normalized
  - $min_{w,b} ||w||$
    $s.t.(wx_j + b)y_j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

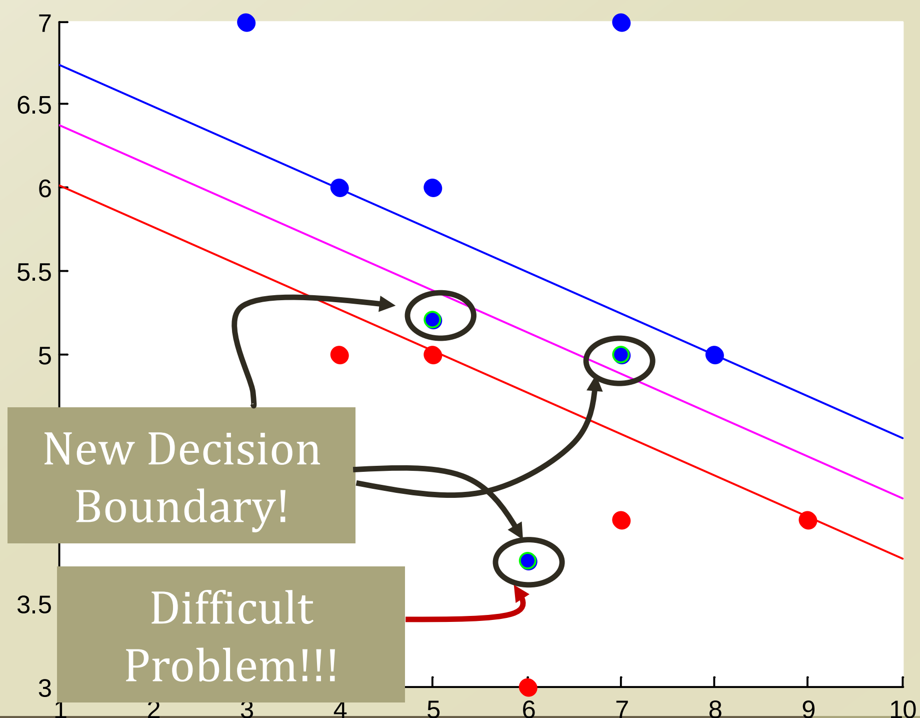


This becomes a quadratic optimization problem. Why?

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
  - $min_{w,b}||w||$
    $s.t.\left(wx_j+b\right)y_j \geq 1, \forall j$

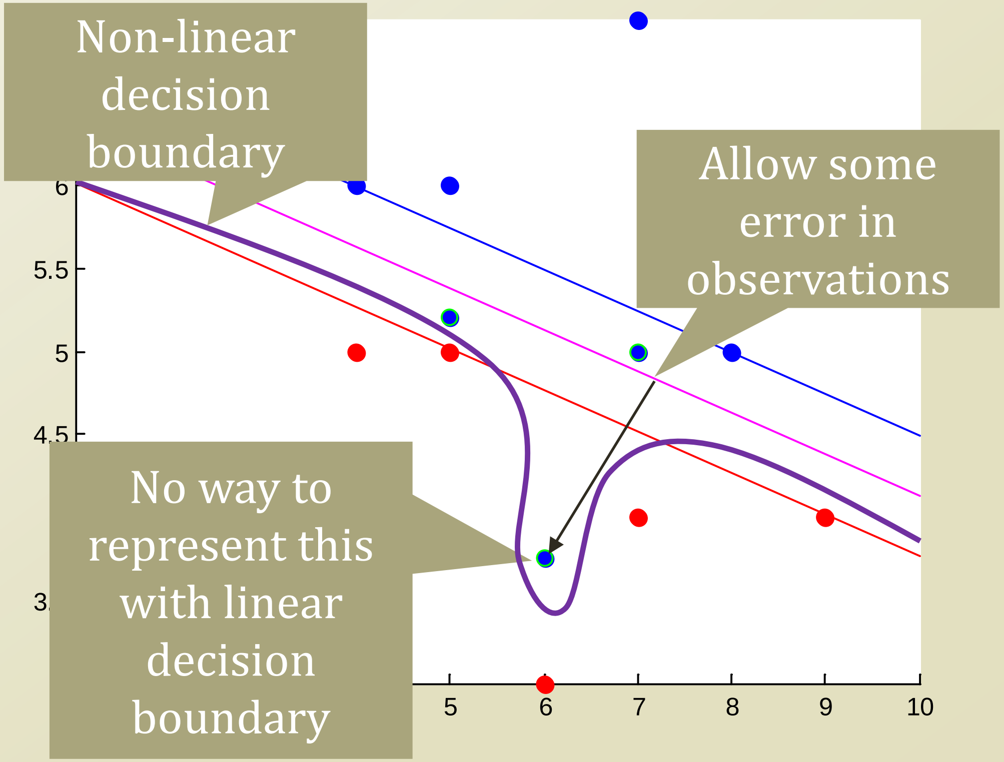




SOFT MARGIN

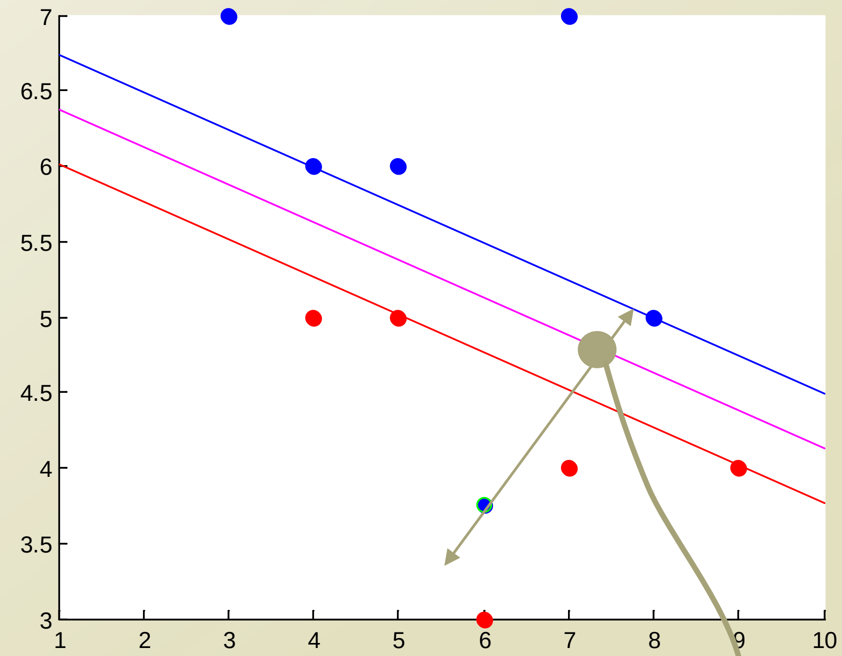
“Error” Cases in SVM

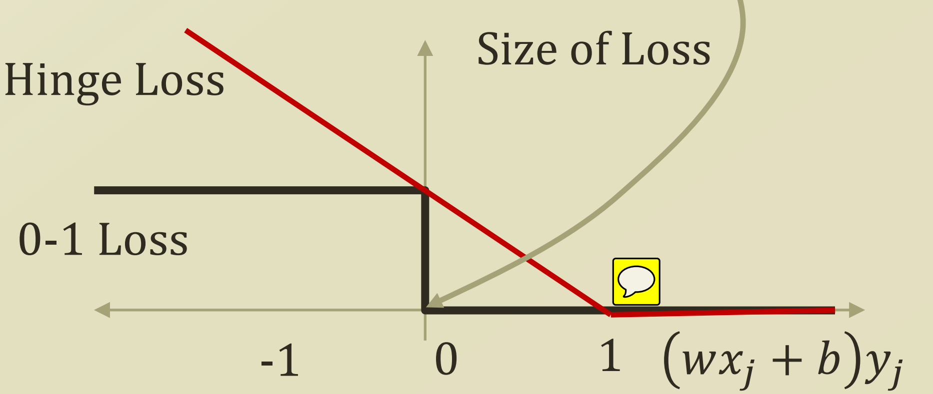
- Data points that are
  - Impossible to classify with a linear decision boundary
- So called, “error” cases…
- How to manage these?
  - Option 1
    - Make decision boundary more complex
    - Go to non-linear
    - Any problem?
  - Option 2
    - Admit there will be an “error”
    - Represent the error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



"Error" Handling in SVM

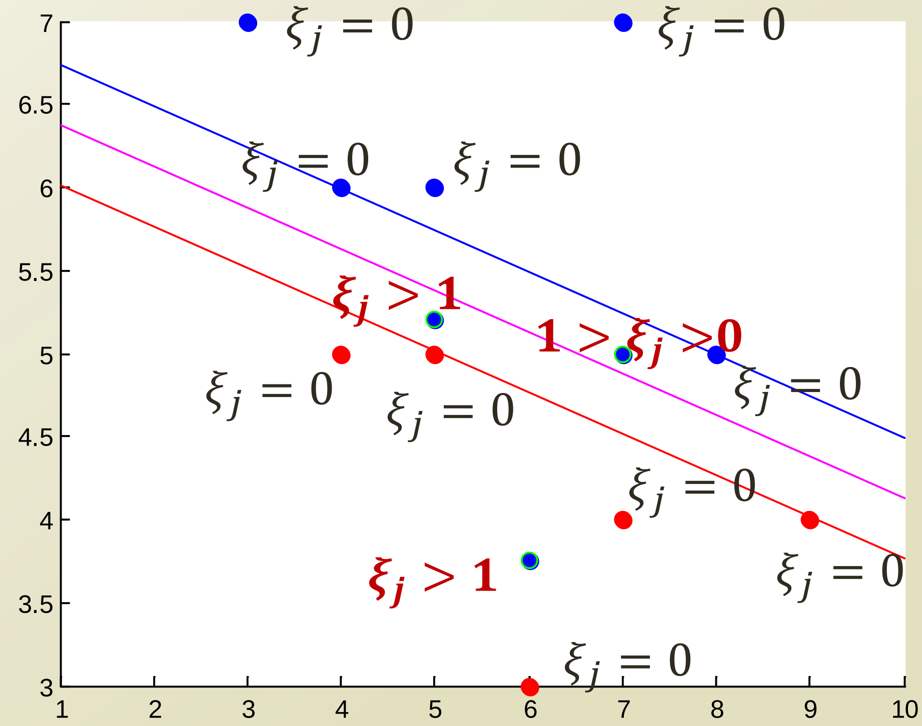
- How to handle
- Option 1)
  - Counting the error cases and reduce the counts
  - $min_{w,b}||w|| + C \times \#_{error}$
    $s.t. (wx_j + b)y_j \geq 1, \forall j$
  - Any problem?
- Option 2)
  - Introduce a slack variable
    - $\xi_j > 1$ when mis-classified
  - $min_{w,b}||w|| + C\sum_j \xi_j$
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

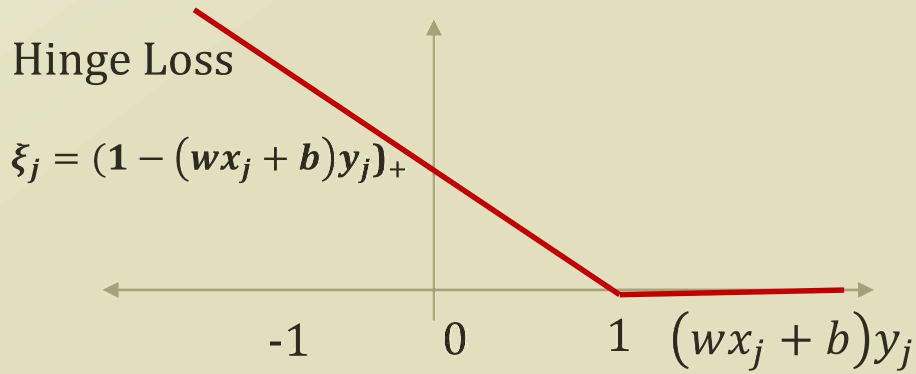




Soft-Margin SVM

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



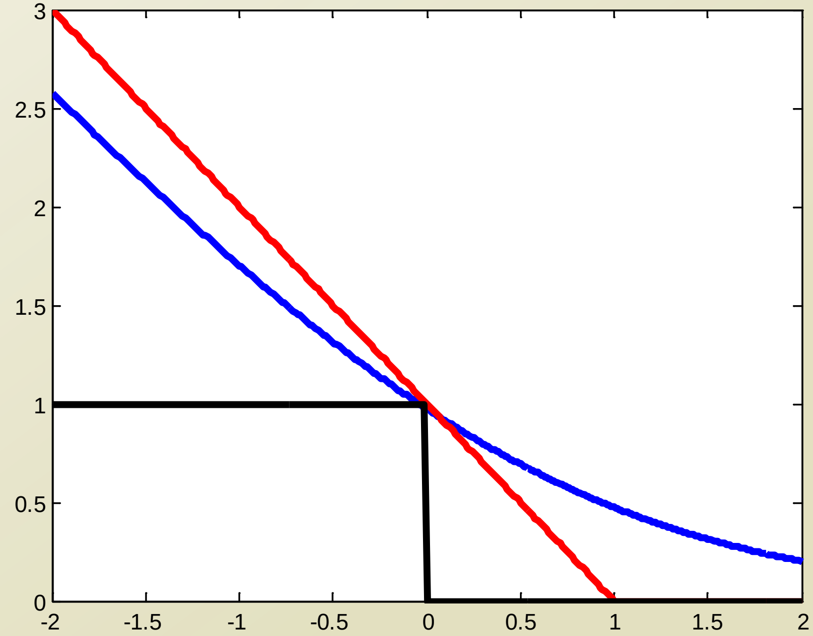


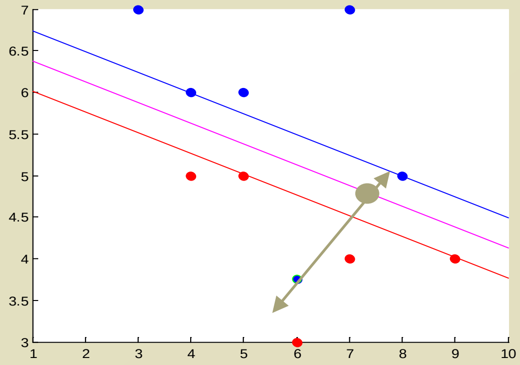
Comparison to Logistic Regression

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

$$= argmax_\theta \sum_{1 \le i \le N} \{Y_i X_i \theta - \log(1 + e^{X_i \theta})\}$$

  - $\xi_j = -\log\left(P(Y_j | X_j, w, b)\right) = \log\left(1 + e^{(wx_j + b)y_j}\right)$
- Which loss function is preferable?
  - Around the decision boundary?
  - Overall place?





Strength of the Loss Function

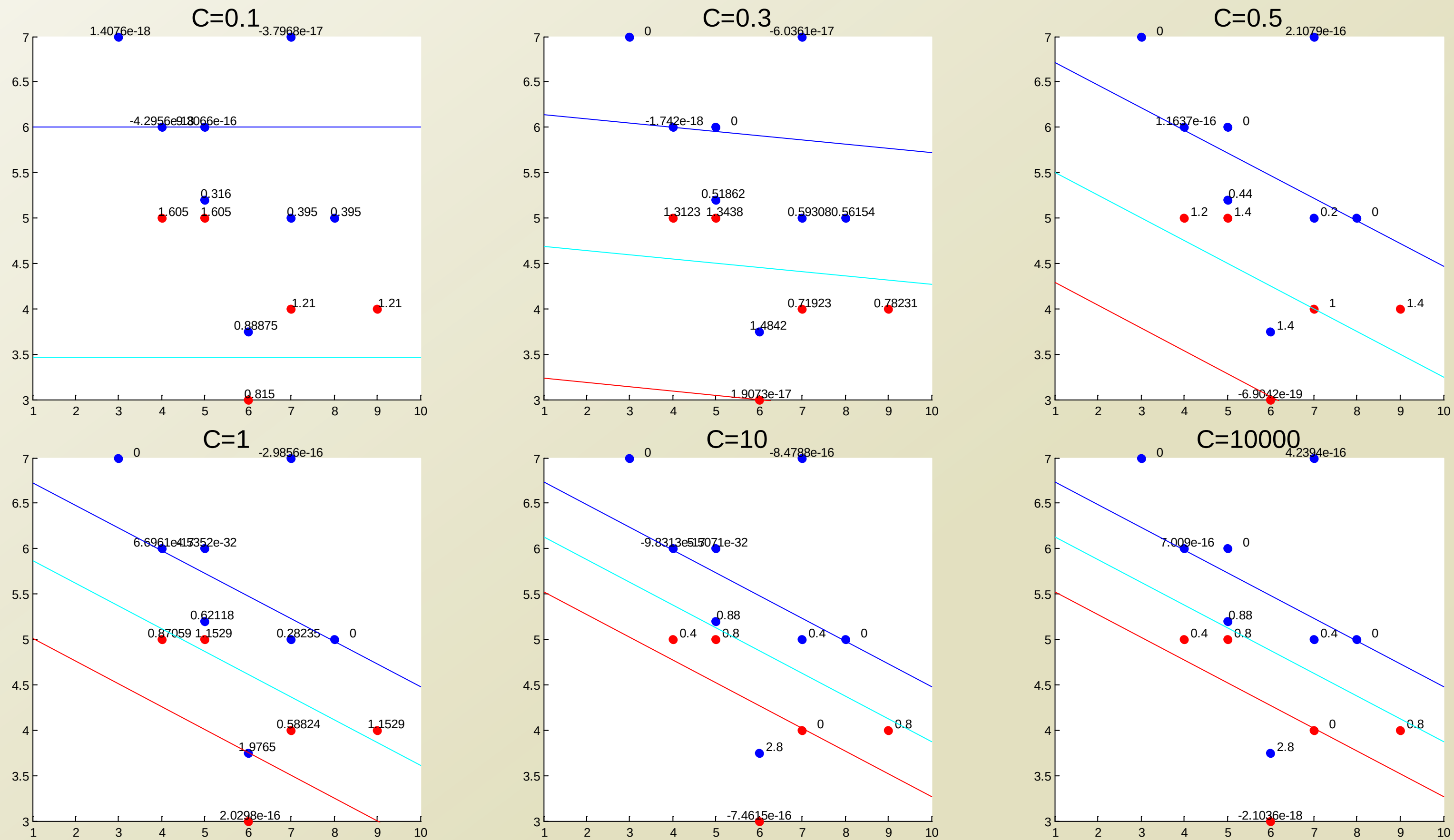
- $min_{w,b,\xi_j} ||w|| + C \sum_j \xi_j$

  $s.t.$

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

  $\xi_j \geq 0, \forall j$

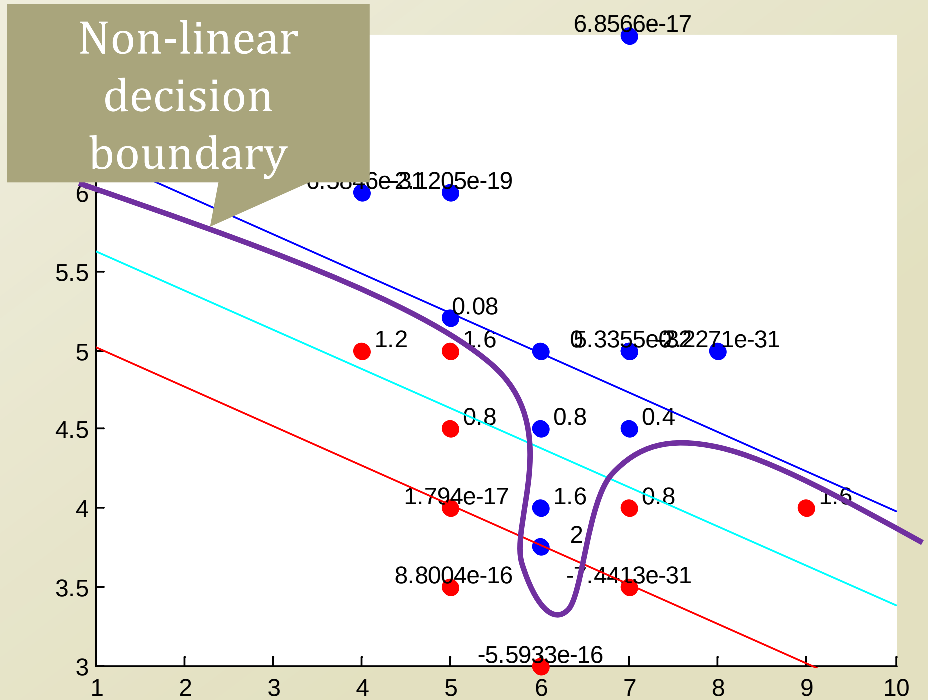
- Let's implement the model
- How does the decision boundary evolves over the variations of C?



KERNEL TRICK

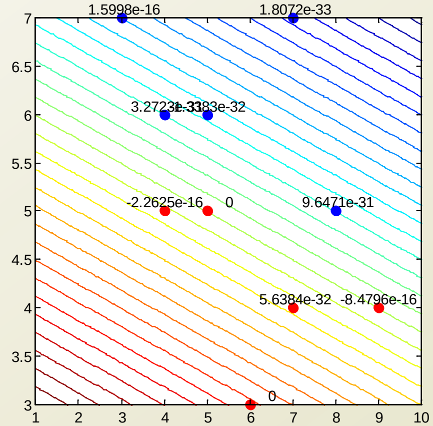
Enough of Studying SVM?

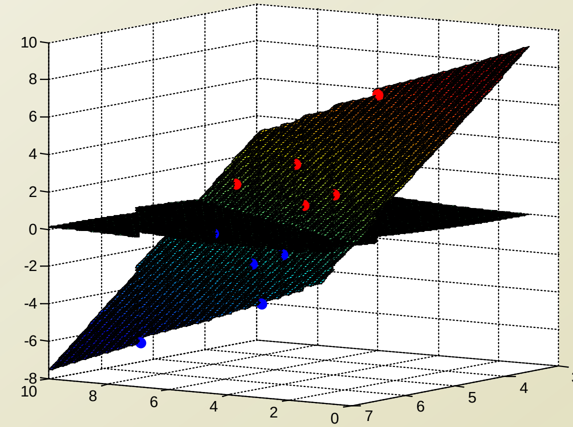
- You can train the SVM when you even have "error" cases
  - Use a soft-margin to handle such errors
- However, this does not change the complexity of the decision boundary
- In the real world, there are situations which require complex decision boundary…
  - Option 1
    - Make decision boundary more 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.

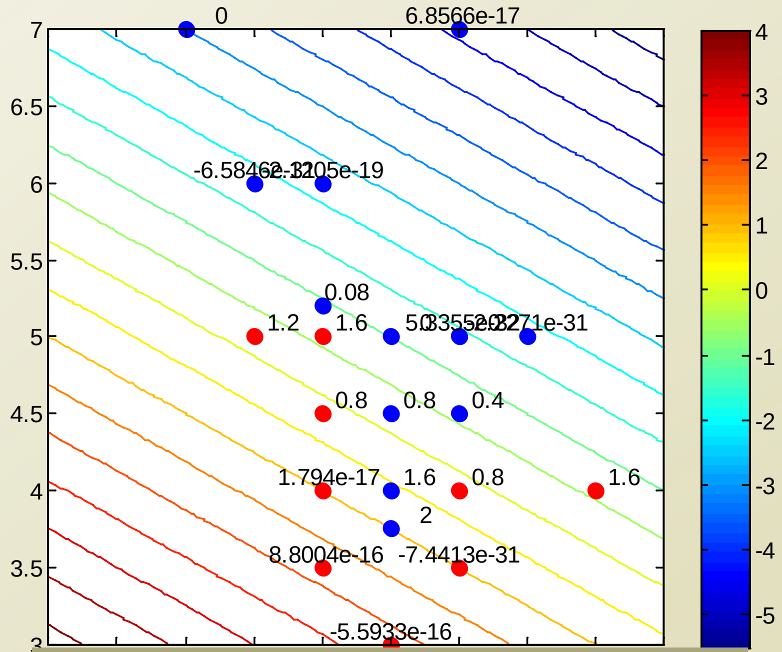
- $min_{w,b,\xi_j} ||w|| + C\sum_j \xi_j$
  $s.t.$
  $(w\varphi(x_j) + b)y_j \geq 1 - \xi_j, \forall j$
  $\xi_j \geq 0, \forall j$
- $\varphi(< x_1, x_2 >) =$
  $< x_1, x_2, {x_1}^2, {x_2}^2, x_1x_2, {x_1}^3, {x_2}^3, {x_1}^2x_2, x_1{x_2}^2 >$

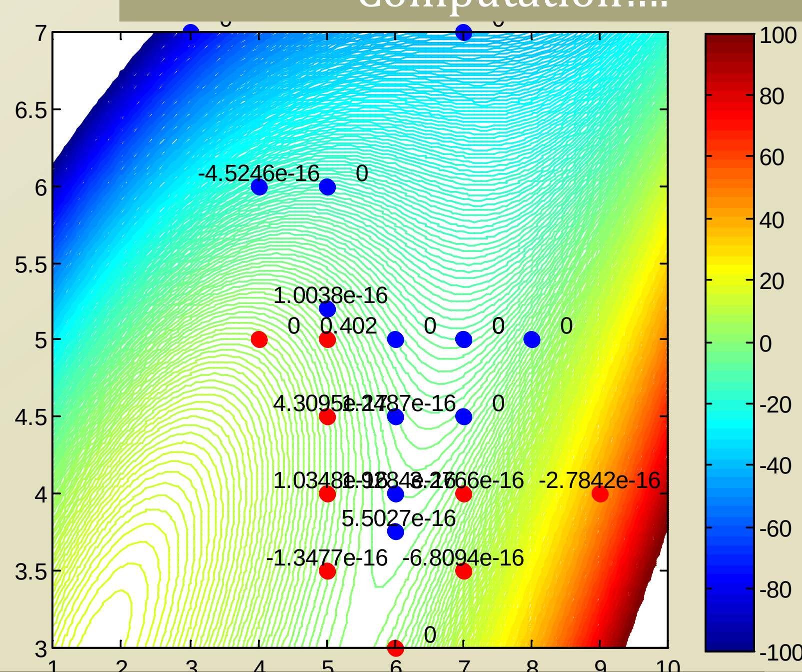




Linearly Separable Dataset

Any problem???
# of Params, Representation, Computation….





Linear Decision Boundary on

Rethinking the Formulation

- SVM turns
  - Classification → 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: $d(\alpha, \beta) = inf_{x \in X} L(x, \alpha, \beta) = min_x L(x, \alpha, \beta)$
  - $max_{\alpha \geq 0, \beta} L(x, \alpha, \beta) = \begin{cases} f(x): if\ x\ is\ feasible \\ \infty: otherwise \end{cases}$
  - $min_x f(x) \rightarrow min_x max_{\alpha \geq 0, \beta} L(x, \alpha, \beta)$
- 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	Lagrange Dual Problem
$min_x f(x)$ $s.t.\ g(x) \leq 0, h(x) = 0$	$max_{\alpha>0,\beta} d(\alpha,\beta)$ $s.t. \alpha > 0$
$min_x max_{\alpha\geq 0,\beta} L(x,\alpha,\beta)$	$max_{\alpha\geq 0,\beta} min_x L(x,\alpha,\beta)$

- Weak duality theorem
  - $d(\alpha,\beta) \leq f(x^*) \quad for \quad \forall\alpha, \forall\beta$
  - $d^* = max_{\alpha\geq 0,\beta} min_x L(x,\alpha,\beta) \leq min_x max_{\alpha\geq 0,\beta} L(x,\alpha,\beta) = p^*$
    - 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-Kunh-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
$g(x^*) < 0 \Rightarrow \alpha^* = 0$
→ Complementary Slackness

Primal Problem

$min_x f(x)$
$s.t.\ g(x) \leq 0,$
$h(x) = 0$

Strong Duality
$d^* = p^*$

Always

KKT Condition

For convex optimization

Primal and dual problems are equivalent for the constrained convex optimization

Dual Problem of SVM

Primal Problem

$$\min_x f(x)$$
$$s.t.\ g(x) \le 0,$$
$$h(x) = 0$$

Lagrange Prime Function

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

Primal Problem of Linearly Separable SVM

$$\min_{w,b} ||w||$$
$$s.t. (wx_j + b)y_j \ge 1, \forall j$$

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

- 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
  - $\alpha_j \ge 0, for\ \forall j$

Dual Problem of Linearly Separable SVM

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

KKT Condition to Eliminate the Duality Gap

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = 0\ ,\ \frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = 0$$
$$\alpha_i \ge 0, \forall i$$
$$\alpha_i \left((wx_j + b)y_j - 1 \right) = 0, \forall i$$

Dual Representation of SVM

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

Dual Problem of Linearly Separable SVM

$$max_{\alpha\geq 0} min_{w,b} \frac{1}{2}w \cdot w - \sum_j \alpha_j\left[(wx_j + b)y_j - 1\right]$$

$$s.t. \alpha_j \geq 0, for \; \forall j$$

KKT Condition to Eliminate the Duality Gap

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

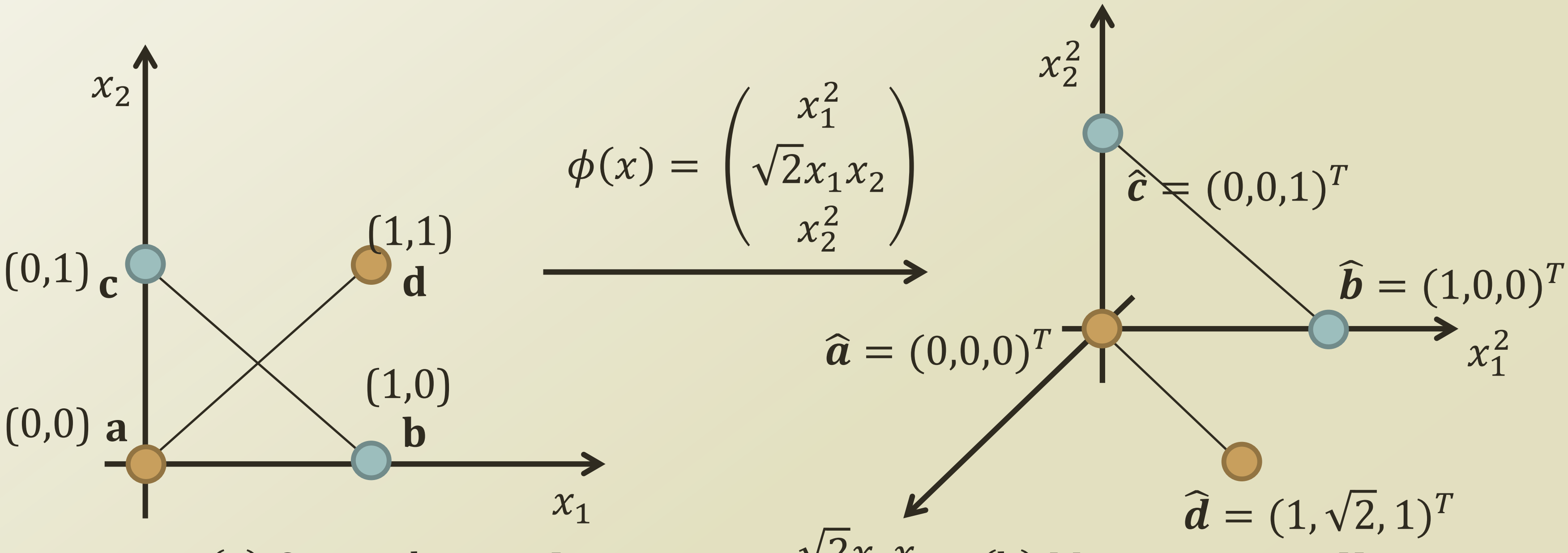
$$\alpha_i \geq 0, \forall i$$

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

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \qquad \sum_{i=1}^{N} \alpha_i y_i = 0$$

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: $\boldsymbol{e_1, e_2, e_3,\ldots,e_n}$ → Linearly independent and generate $\boldsymbol{R^n}$
- Expanding the Basis through Space mapping function $\phi : L \rightarrow H$
  - Or, transformation function, etc…
- Any problem????
  - Feature space becomes bigger and bigger….



(a) Original space, L (b) Mapping space, H

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)
  - $K\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = \varphi(\boldsymbol{x_i}) \cdot \varphi(\boldsymbol{x_j})$

- Some common kernels are following :
  - Polynomial(homogeneous)
    - $k\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = (\boldsymbol{x_i} \cdot \boldsymbol{x_j})^d$
  - Polynomial(inhomogeneous)
    - $k\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = (\boldsymbol{x_i} \cdot \boldsymbol{x_j} + 1)^d$
  - Gaussian kernel function, a.k.a. Radial Basis Function
    - $k\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = \exp\left(-\gamma\left\|\boldsymbol{x_i} - \boldsymbol{x_j}\right\|^2\right)$
      - For $\gamma > 0$. Sometimes parameterized using $\gamma = \frac{1}{2\sigma^2}$
  - Hyperbolic tangent, a.k.a. Sigmoid Function
    - $k\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = \tanh\left(\kappa \boldsymbol{x_i} \cdot \boldsymbol{x_j} + c\right)$
    - For some(not every) $\kappa > 0$ and c < 0

Polynomial Kernel Function

- Imagine we have
  - $\mathbf{x}=<x_1, x_2>$ and $\mathbf{z}=<z_1, z_2>$
  - Polynomial Kernel Function of degree 1
    - $K(<x_1, x_2>, <z_1, z_2>) = <x_1, x_2> \cdot <z_1, z_2> = x_1 z_1 + x_2 z_2 = \mathbf{x} \cdot \mathbf{z}$
  - Polynomial Kernel Function of degree 2
    - $K(<x_1, x_2>, <z_1, z_2>) = <x_1^2, \sqrt{2} x_1 x_2, x_2^2> \cdot <z_1^2, \sqrt{2} z_1 z_2, z_2^2>$
    - $= x_1^2 z_1^2 + 2 x_1 x_2 z_1 z_2 + x_2^2 z_2^2 = (x_1 z_1 + x_2 z_2)^2 = (\mathbf{x} \cdot \mathbf{z})^2$
  - Polynomial Kernel Function of degree 3
    - $K(<x_1, x_2>, <z_1, z_2>) = (\mathbf{x} \cdot \mathbf{z})^3$
  - Polynomial Kernel Function of degree n
    - $K(<x_1, x_2>, <z_1, z_2>) = (\mathbf{x} \cdot \mathbf{z})^n$
- Do we need to express and calculate the transformed coordinate values for ***x*** and ***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

- $max_{\alpha \geq 0} \sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \varphi(x_i) \varphi(x_j)$
- $max_{\alpha \geq 0} \sum_j \alpha_j - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j K(x_i, x_j)$
  - $\alpha_i \left((wx_j + b) y_j - 1 \right) = 0, C > \alpha_i > 0$
    - $\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \varphi(\boldsymbol{x}_i)$
    - $b = y_j - \sum_{i=1}^{N} \alpha_i y_i \varphi(\boldsymbol{x}_i) \varphi(x_j)$
  - $\sum_{i=1}^{N} \alpha_i y_i = 0$
  - $C \geq \alpha_i \geq 0, \forall i$
- 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

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

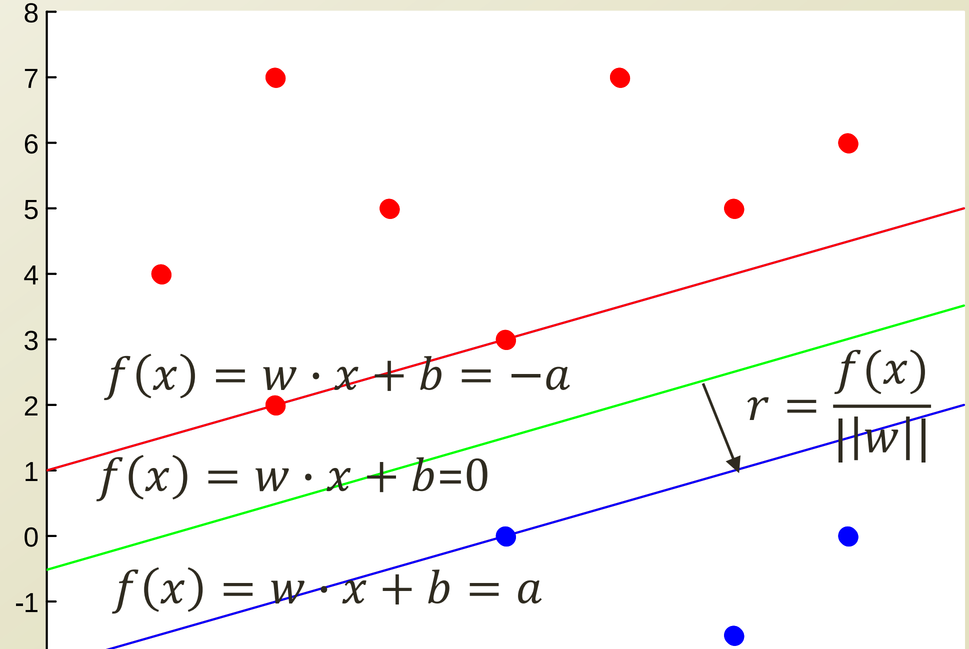
$$C \geq \alpha_i \geq 0, \forall i$$

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

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

Classification with SVM Kernel Trick

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



$$sign(w \cdot \varphi(x) + b) = sign\left(\sum_{i=1}^{N} \alpha_i y_i \varphi(\boldsymbol{x}_i) \cdot \varphi(x) + y_j - \sum_{i=1}^{N} \alpha_i y_i \varphi(\boldsymbol{x}_i)\,\varphi(x_j)\right)$$

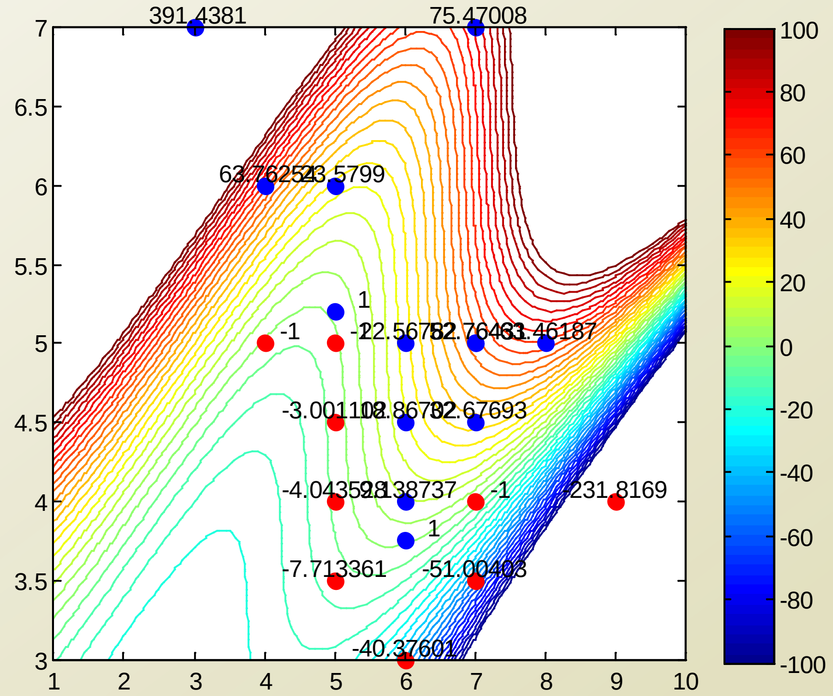
$$= sign\left(\sum_{i=1}^{N} \alpha_i y_i K(\boldsymbol{x}_i, x) + y_j - \sum_{i=1}^{N} \alpha_i y_i K(\boldsymbol{x}_i, x_j)\right)$$

$$0 < \alpha_j < C$$

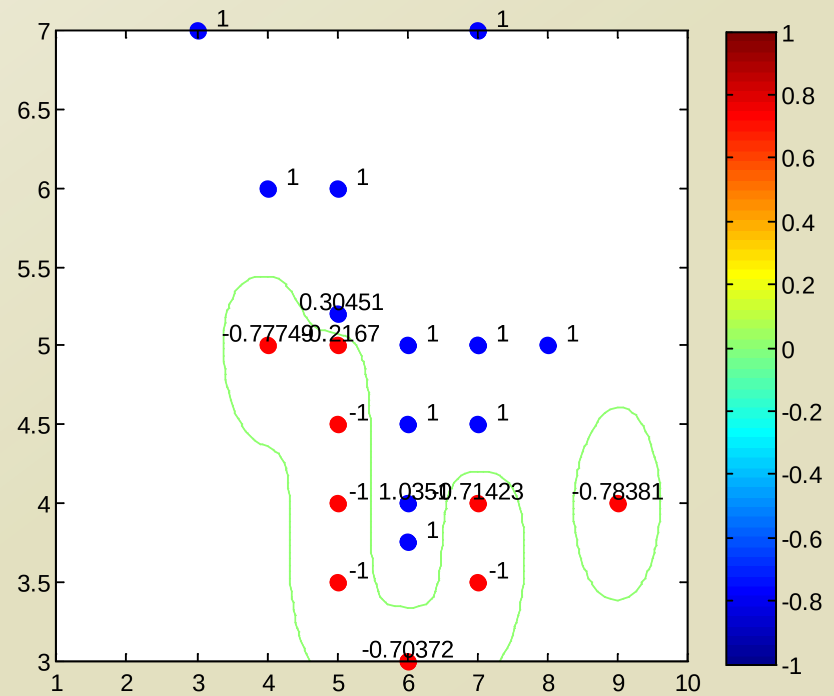
SVM with Various Kernels

- 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 γ=4



Logistic Regression with Kernel

- Logistic regression
  - $P(Y|X) = \frac{1}{1+e^{-\dot{\theta}^T x}}$
  - Finding the MLE of θ
- Can we kernelize the logistic regression?
  - $\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \varphi(\boldsymbol{x}_i)$
  - $P(Y|X) = \frac{1}{1+e^{-\dot{\theta}^T x}} = \frac{1}{1+e^{\sum_{i=1}^{N} \alpha_i y_i \varphi(x_i)\varphi(x)+b}} = \frac{1}{1+e^{\sum_{i=1}^{N} \alpha_i y_i K(x_i, x)+b}}$
- Problem changes
  - From finding θ to finding α_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