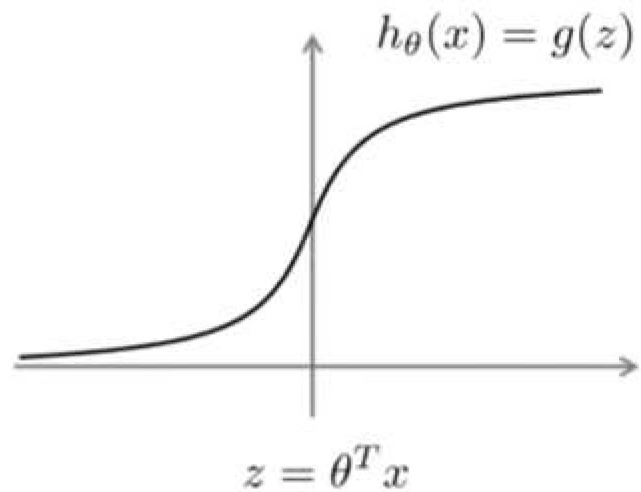
Support Vector Machine

Some slides were adapted/taken from various sources, including Prof. Andrew Ng's Coursera Lectures, Stanford University, Prof. Kilian Q. Weinberger's lectures on Machine Learning, Cornell University, Prof. Sudeshna Sarkar's Lecture on Machine Learning, IIT Kharagpur, Prof. Bing Liu's lecture, University of Illinois at Chicago (UIC), CS231n: Convolutional Neural Networks for Visual Recognition lectures, Stanford University, Patrick Winston, MIT OpenCourseWare and many more. We thankfully acknowledge them. Students are requested to use this material for their study only and **NOT** to distribute it.

Optimization objective

Alternative view of logistic regression

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$



If $y = 1$, we want $h_\theta(x) \approx 1$, $\theta^T x \gg 0$

If $y = 0$, we want $h_\theta(x) \approx 0$, $\theta^T x \ll 0$

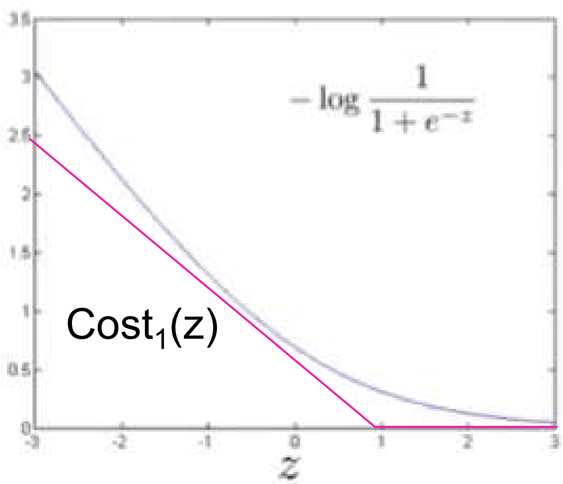
SVM

Alternative view of logistic regression

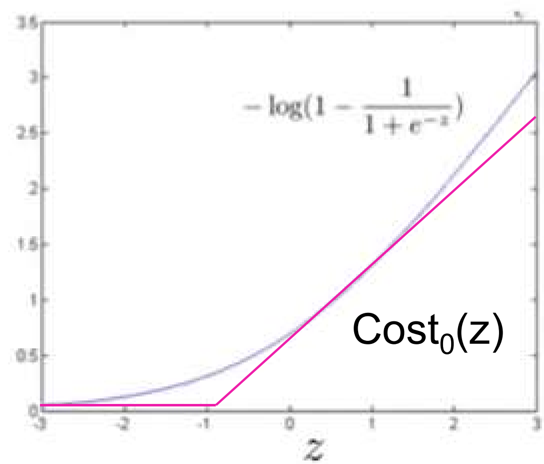
Cost of example: $-(y\log h_\theta(x) + (1-y)\log(1-h_\theta(x)))$

$$= -y\log\frac{1}{1+e^{-\theta^T x}} - (1-y)\log(1-\frac{1}{1+e^{-\theta^T x}})$$

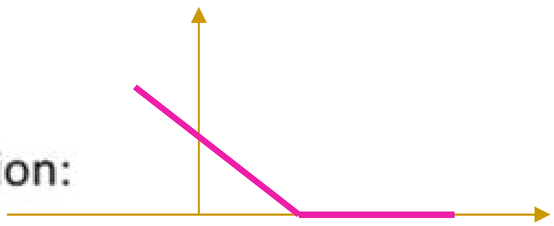
If $y=1$ (want $\theta^T x \gg 0$):

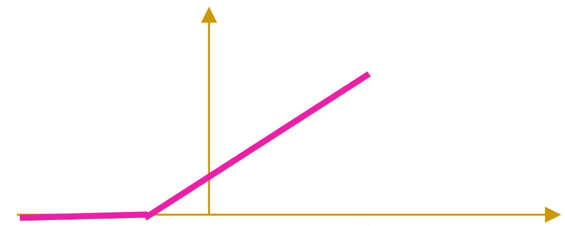


If $y=0$ (want $\theta^T x \ll 0$):



SVM





Logistic regression:

$$\min_\theta \frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\left(\underbrace{-\log h_\theta(x^{(i)})}_{Cost_1(\Theta^T x^{(i)})}\right) + (1-y^{(i)})\left(\underbrace{(-\log(1-h_\theta(x^{(i)}))}_{Cost_0(\Theta^T x^{(i)})}\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Support vector machine: $Cost_1(\Theta^T x^{(i)})$ $Cost_0(\Theta^T x^{(i)})$

$$\min_\Theta \frac{1}{m}\sum_{1}^{m} y^{(i)} Cost_1(\Theta^T x^{(i)}) + (1-y^{(i)}) Cost_0(\Theta^T x^{(i)}) + \frac{\lambda}{2m}\sum_{i=1}^{n}\Theta_j^2$$

A + λ B = C A + B where C = 1/ λ

$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1-y^{(i)}) cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{i=1}^{n}\theta_j^2$$

SVM

SVM hypothesis

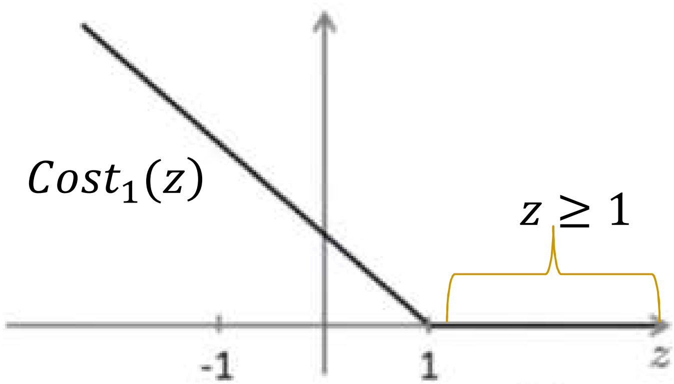
$$\min_{\theta} C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$

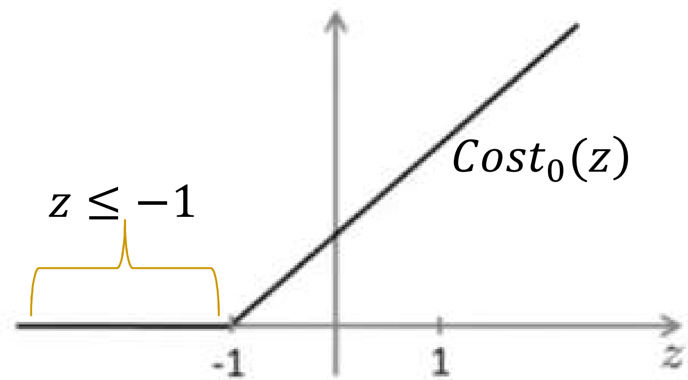
Hypothesis:

$$h_{\Theta}(x) = \begin{cases} 1 & if \;\; \Theta^T x \geq 0 \\ 0 & otherwise \end{cases}$$

SVM: As Large Margin Classifier

$$\min_{\theta} C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$





If $y = 1$, we want $\theta^T x \geq 1$ (not just ≥ 0)

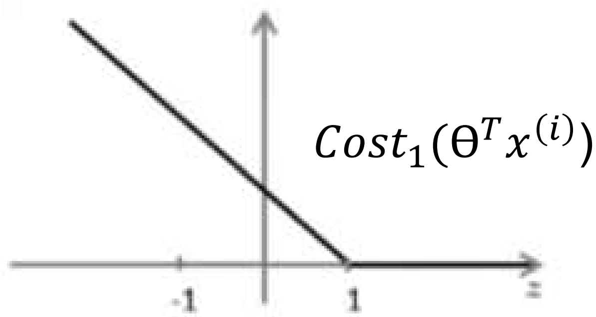
If $y = 0$, we want $\theta^T x \leq -1$ (not just < 0)

SVM Decision Boundary

$$\min_\theta C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$

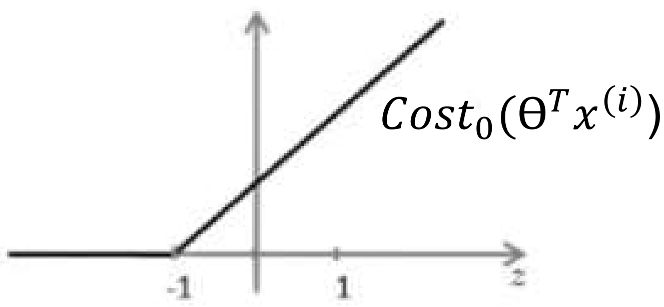
Whenever $y^{(i)} = 1$:

$$\Theta^T x^{(i)} \geq 1$$



Whenever $y^{(i)} = 0$:

$$\Theta^T x^{(i)} \leq -1$$



SVM Decision Boundary

$$\min_{\theta} C \sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$

$= 0$ as C is very big number

Whenever $y^{(i)} = 1$:

$$\Theta^T x^{(i)} \geq 1$$

$$\min_{\Theta} C \times 0 + \frac{1}{2} \sum_{i=1}^{n} \Theta_j^2 \text{ i.e. } \min_{\Theta} \frac{1}{2} \sum_{i=1}^{n} \Theta_j^2$$

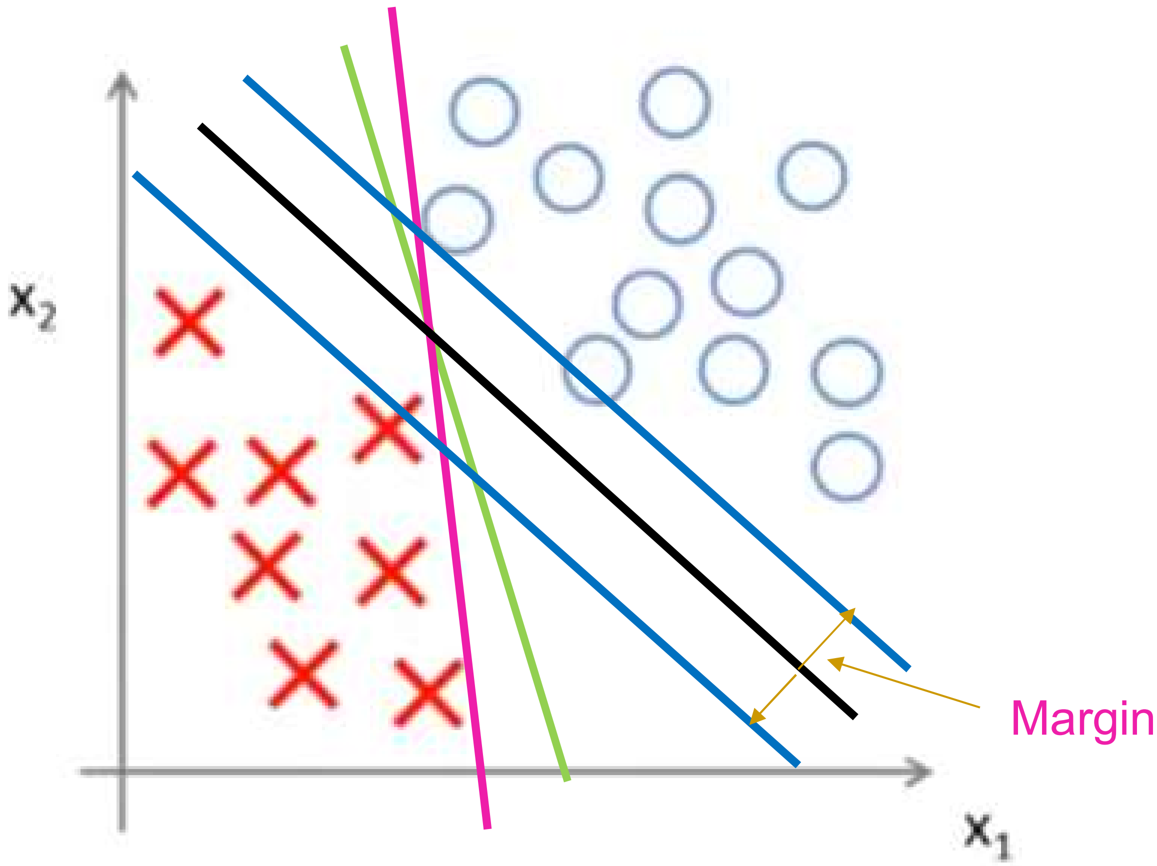
$$s.t. \quad \Theta^T x^{(i)} \begin{cases} \geq 1 & if \quad y^{(i)} = 1 \\ \leq -1 & if \quad y^{(i)} = 0 \end{cases}$$

Whenever $y^{(i)} = 0$:

$$\Theta^T x^{(i)} \leq -1$$

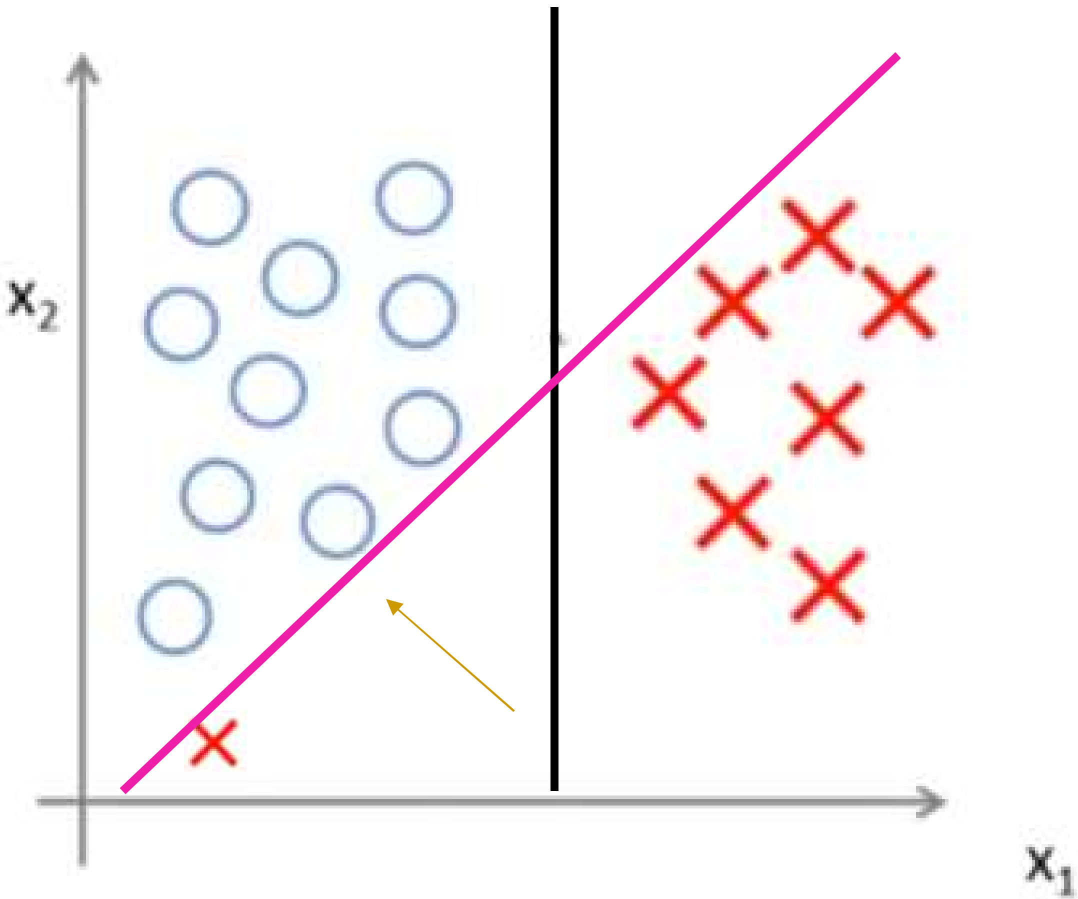
SVM Decision Boundary

Linearly separable case



Large Margin Classifier

In case of Outliers



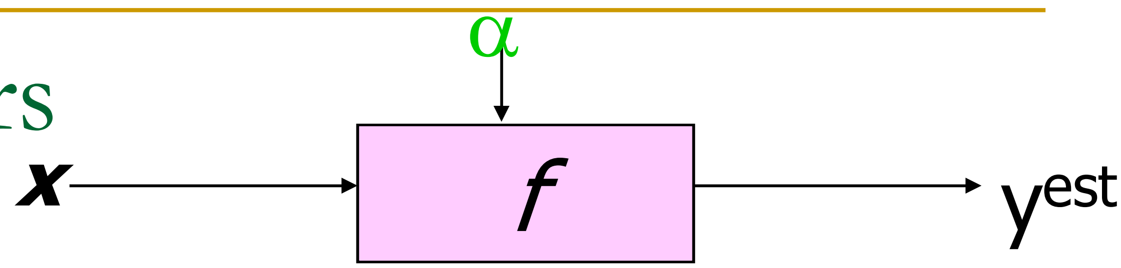
C is very large

Sensitive to outliers

to continue...

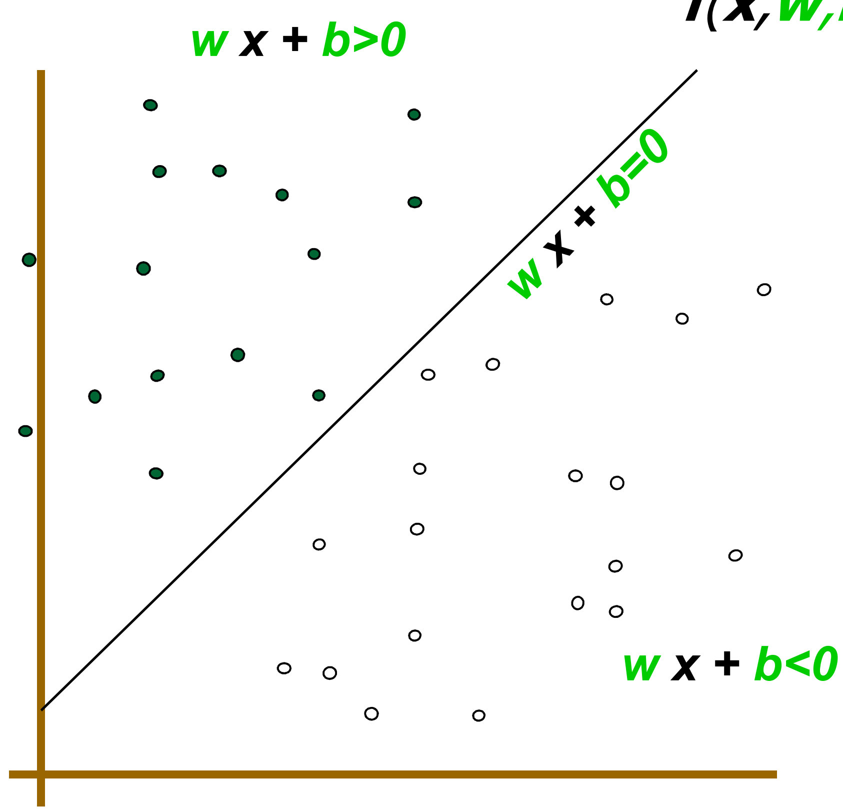
Support Vector Machine

Linear Classifiers



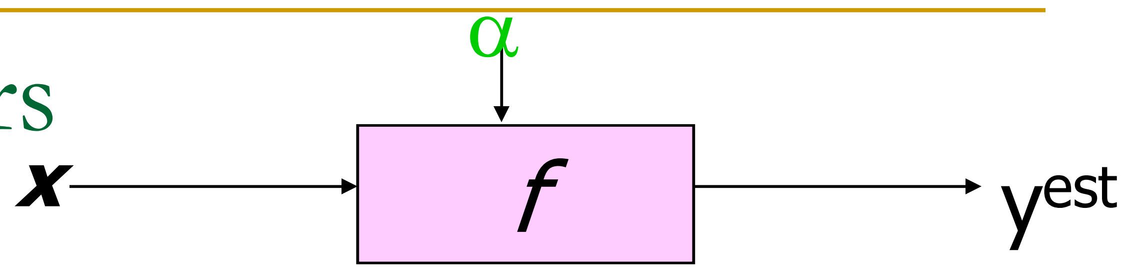
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$

- • denotes +1
- ○ denotes -1



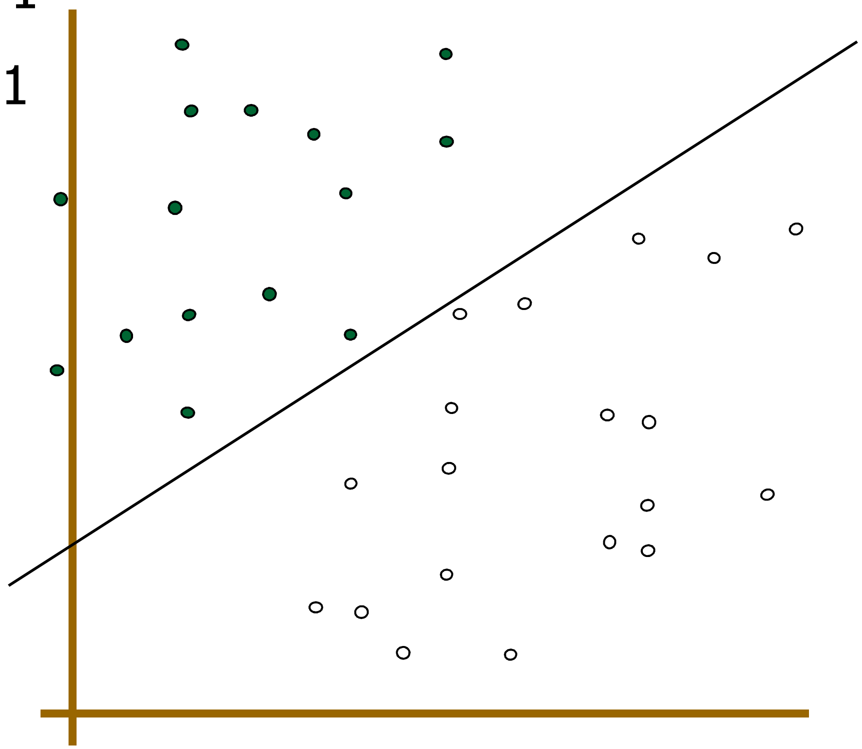
How would you classify this data?

Linear Classifiers



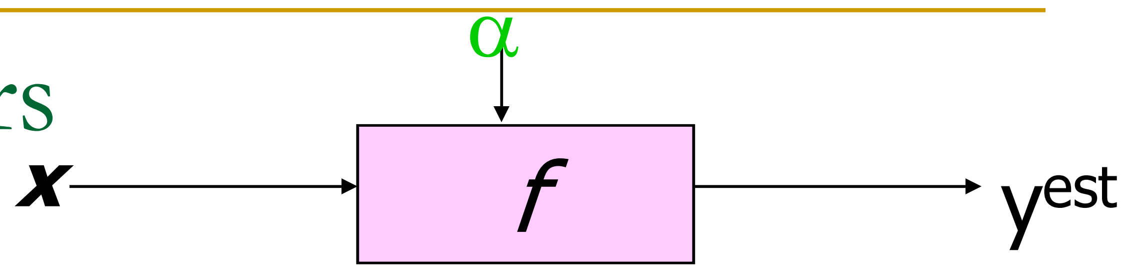
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$

- denotes +1
- denotes -1

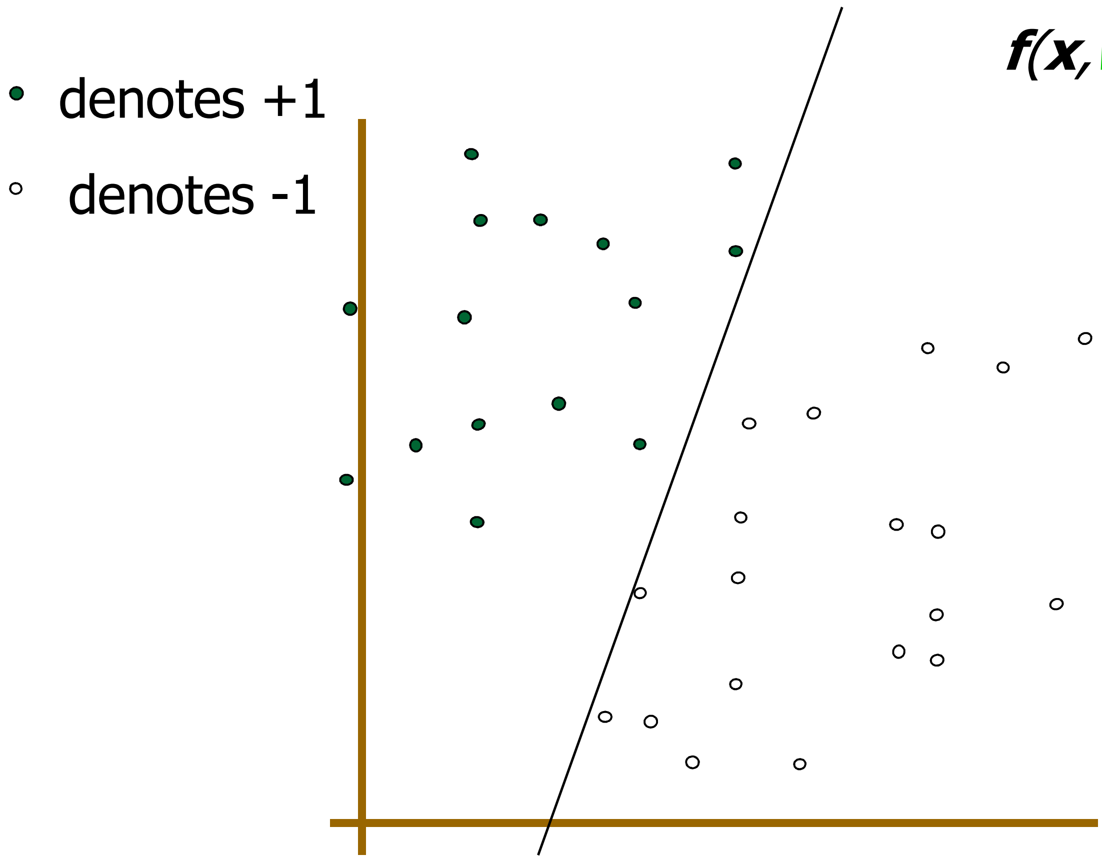


How would you classify this data?

Linear Classifiers

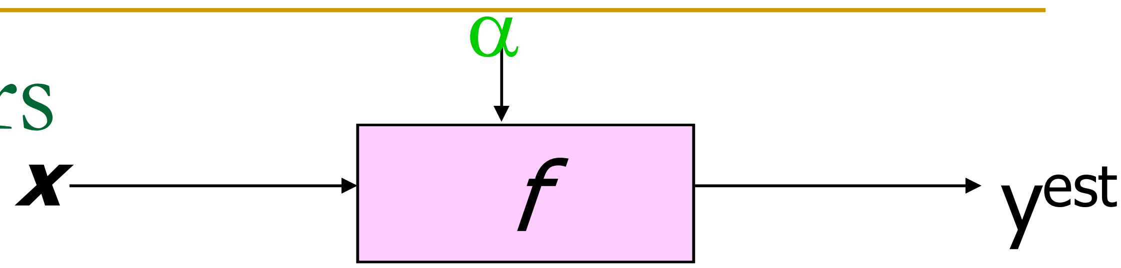


$$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$$

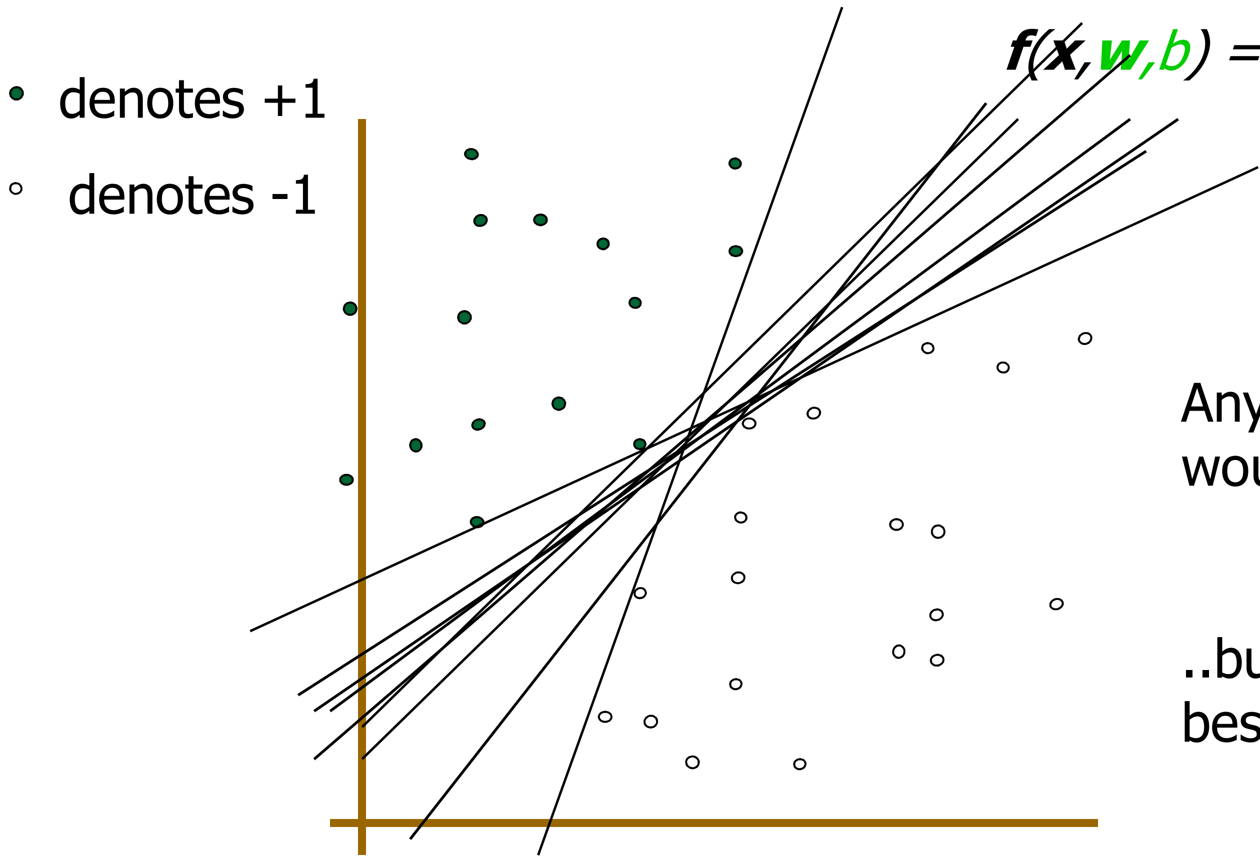


How would you classify this data?

Linear Classifiers



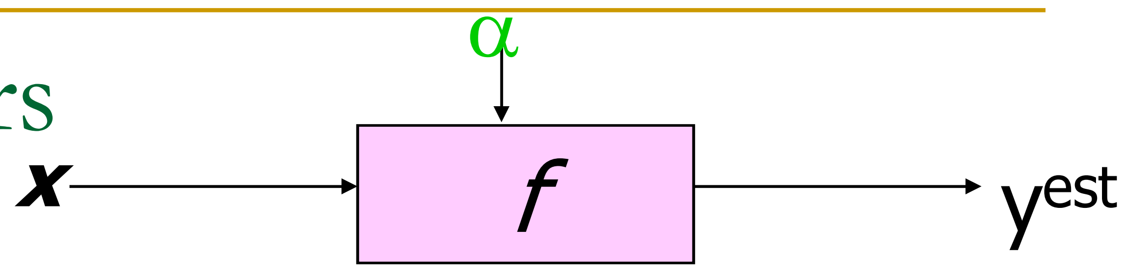
$$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}\ \mathbf{x} + b)$$



Any of these would be fine..

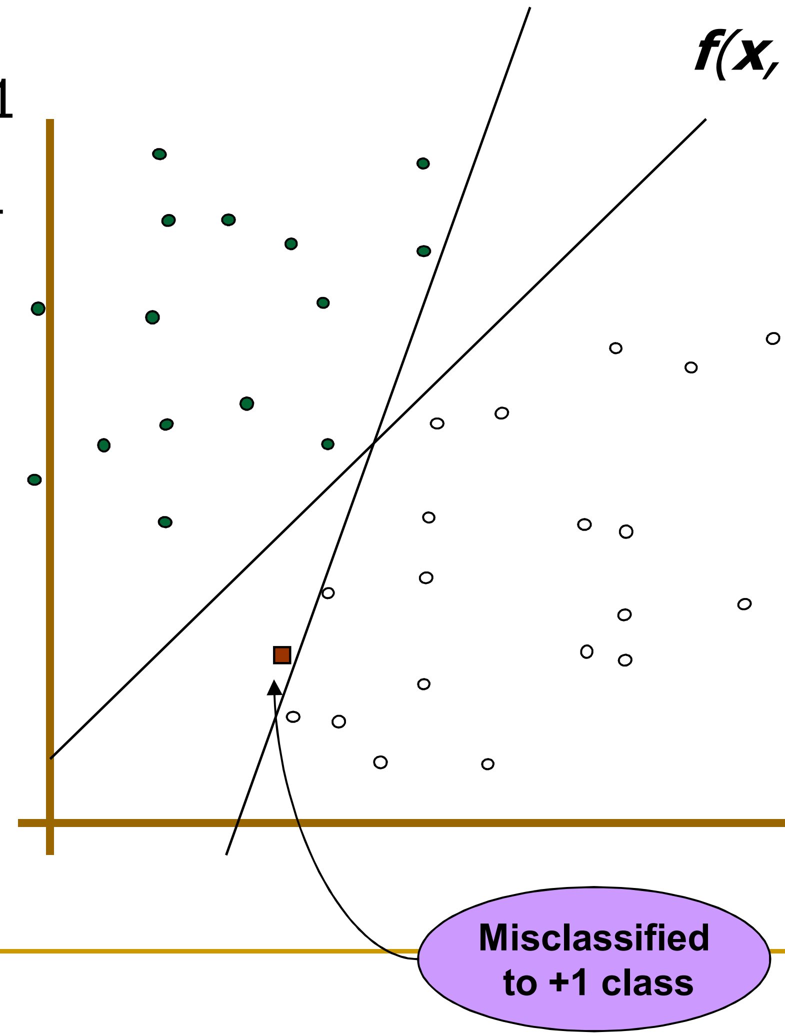
..but which is best?

Linear Classifiers



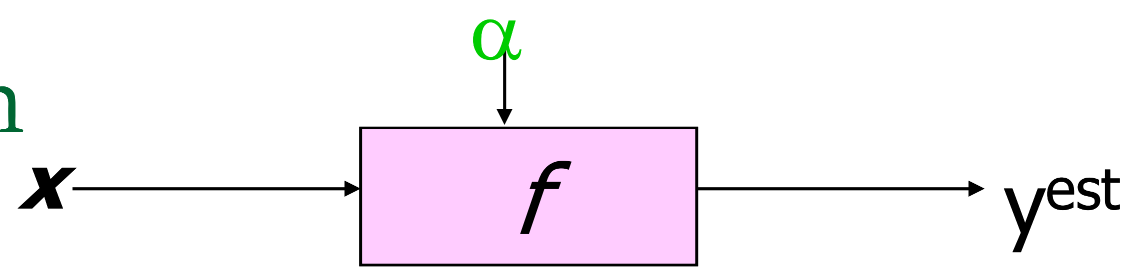
$$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$$

- • denotes +1
- ° denotes -1

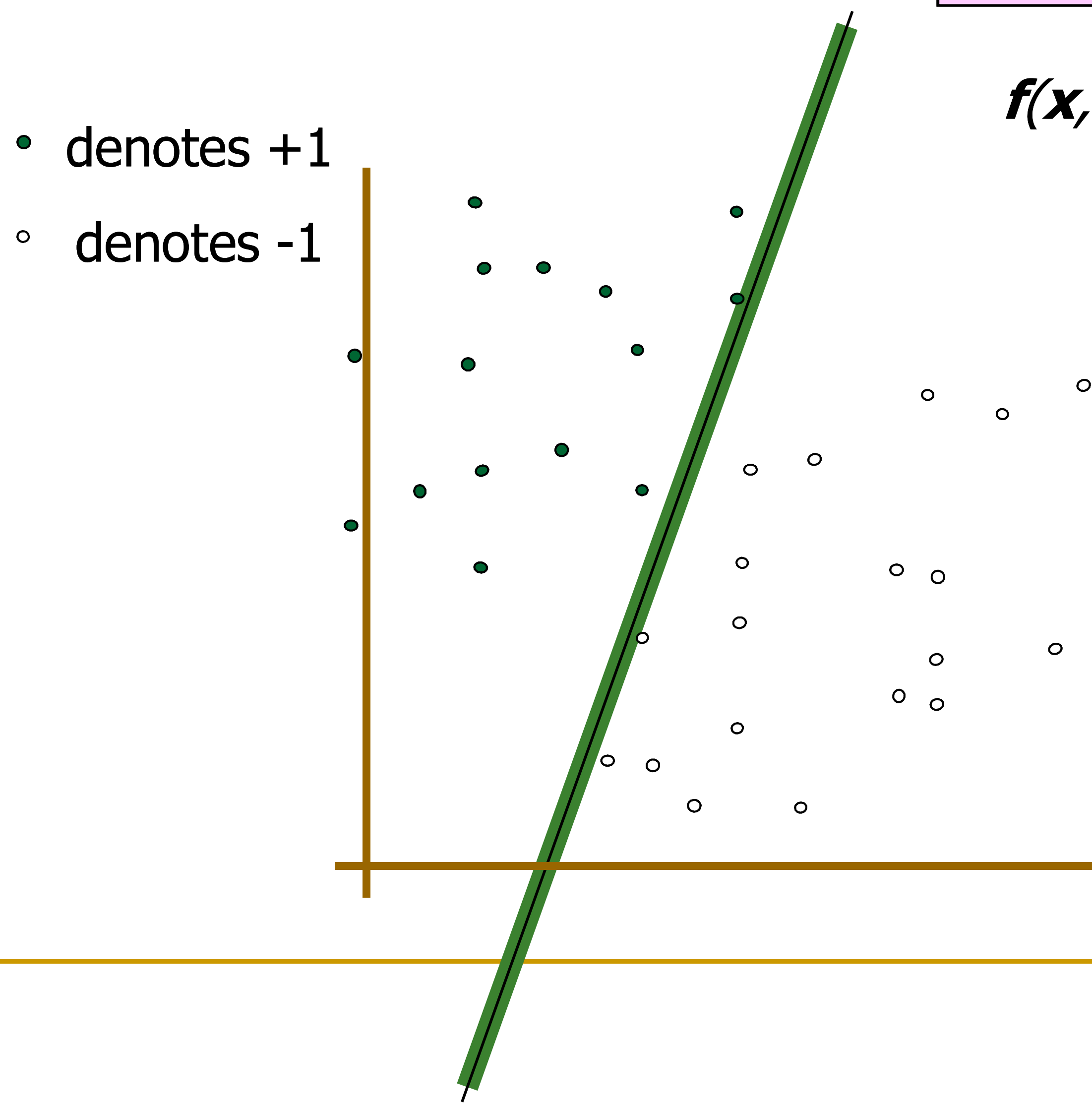


How would you classify this data?

Classifier Margin

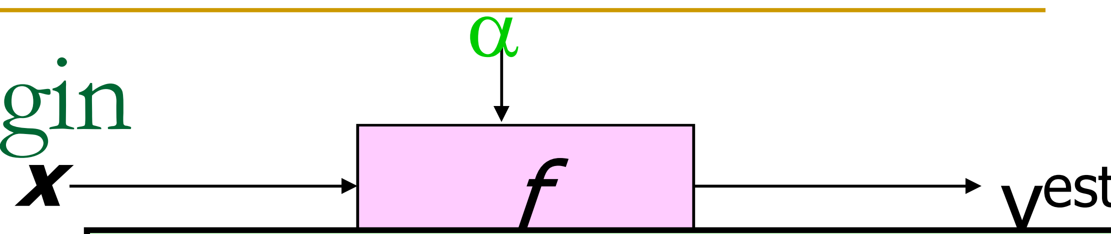


$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\cdot\mathbf{x} + b)$



Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

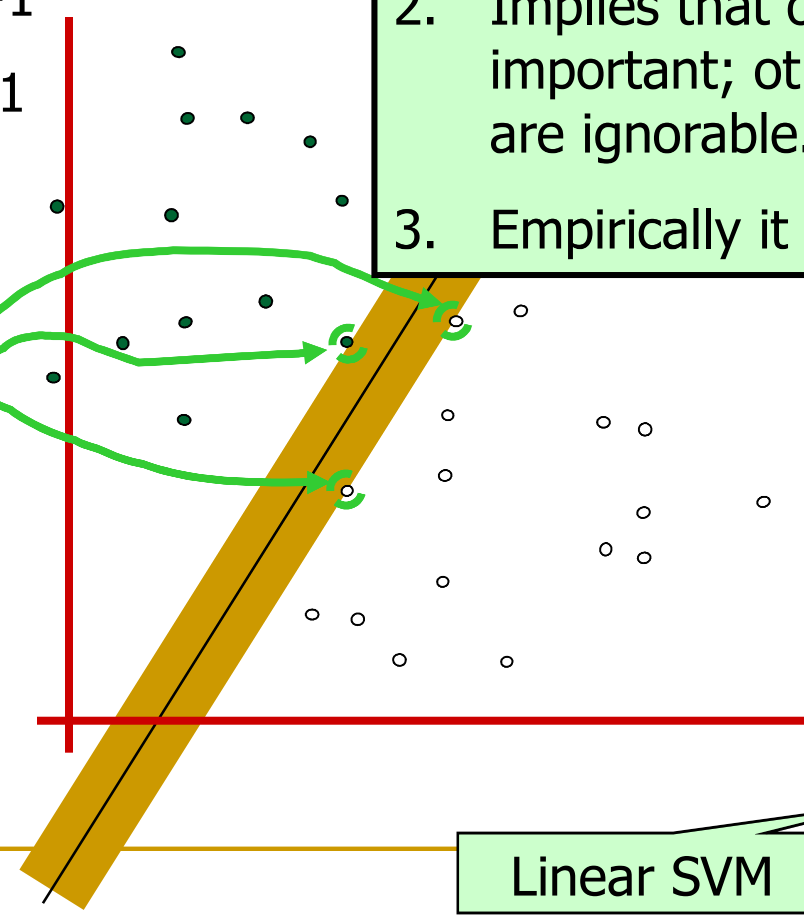
Maximum Margin



1. Maximizing the margin is good according to intuition and PAC* theory
2. Implies that only support vectors are important; other training examples are ignorable.
3. Empirically it works very very well.

• denotes +1

° denotes -1



Support Vectors are those datapoints that the margin pushes up against

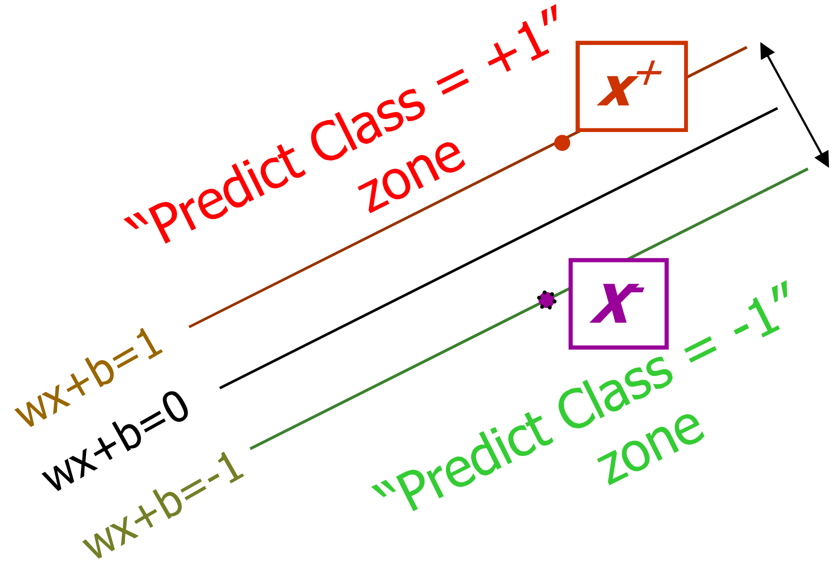
linear classifier with the, um, maximum margin.

This is the simplest kind of SVM (Called an LSVM)

Linear SVM

*Probability Approximately Correct

Linear SVM Mathematically



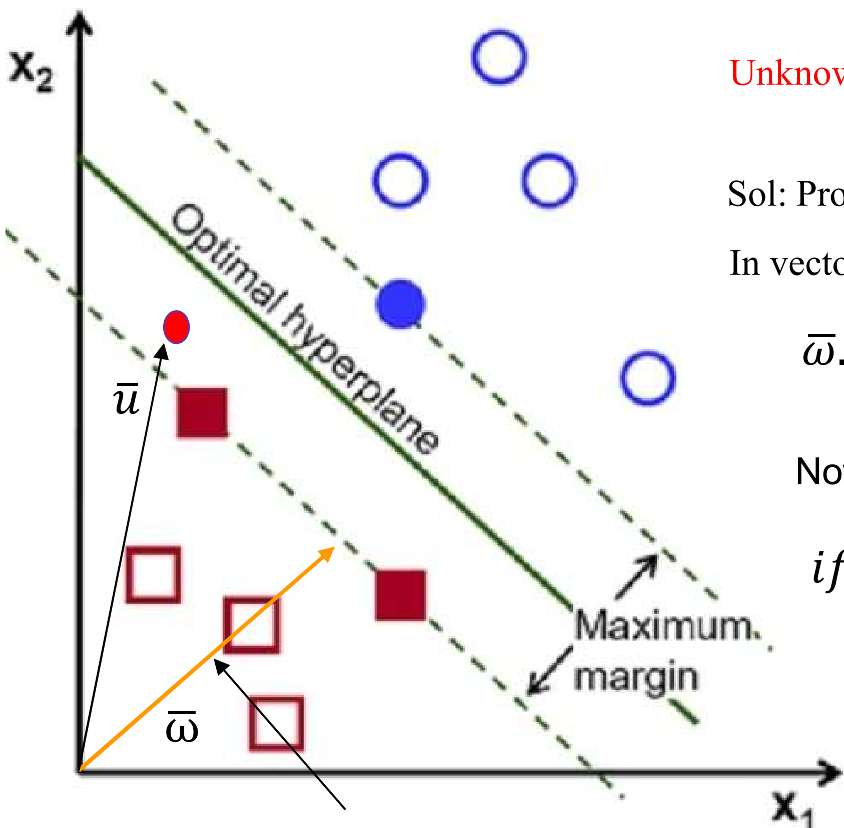
M=Margin Width

What we know:

- $\boldsymbol{w} \cdot \boldsymbol{x}^+ + b = +1$
- $\boldsymbol{w} \cdot \boldsymbol{x}^- + b = -1$
- $\boldsymbol{w} \cdot (\boldsymbol{x}^+ - \boldsymbol{x}^-) = 2$

$$M = \frac{(x^+ - x^-) \cdot w}{|w|} = \frac{2}{|w|}$$

Linear SVM Mathematically



Perpendicular to the median

Unknown ($\bar{u}$) is left side or right side of the DB?

Sol: Project the vector ($\bar{u}$) on the perpendicular of the DB

In vector representation,

$\bar{\omega}.\bar{u} \geq C$ Here dot product is taking a projection of $\bar{u}$ on $\overline{\omega}$

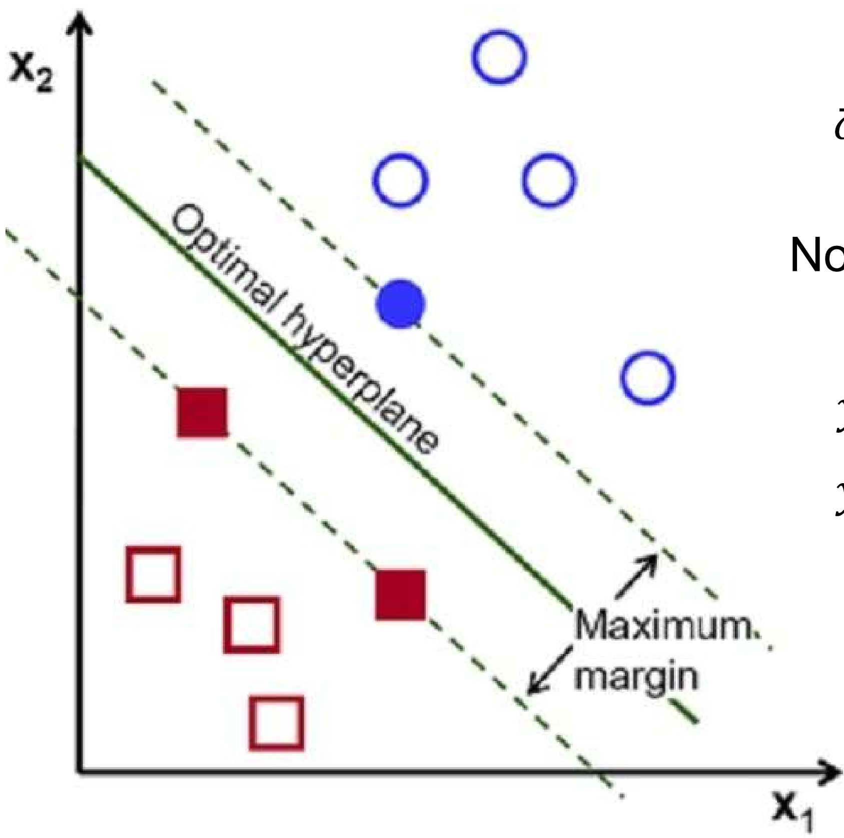
Now, without loss of generality, we can write

$if \quad \bar{\omega}.\bar{u} + b \geq 0 \quad then + (\bigcirc)$: Decision rule

But the problem is, we don't know b and $\bar{\omega}$ only we know that $\bar{\omega}$ is perpendicular to DB, but there may be many such $\bar{\omega}$ of different lengths.

C=-b

Linear SVM Mathematically



$$\bar{\omega}.\bar{x}_+ + b \geq 1$$

$$\bar{\omega}.\bar{x}_- + b \geq -1$$

Now, y_i such that $y_i = +1 \ for + samples$

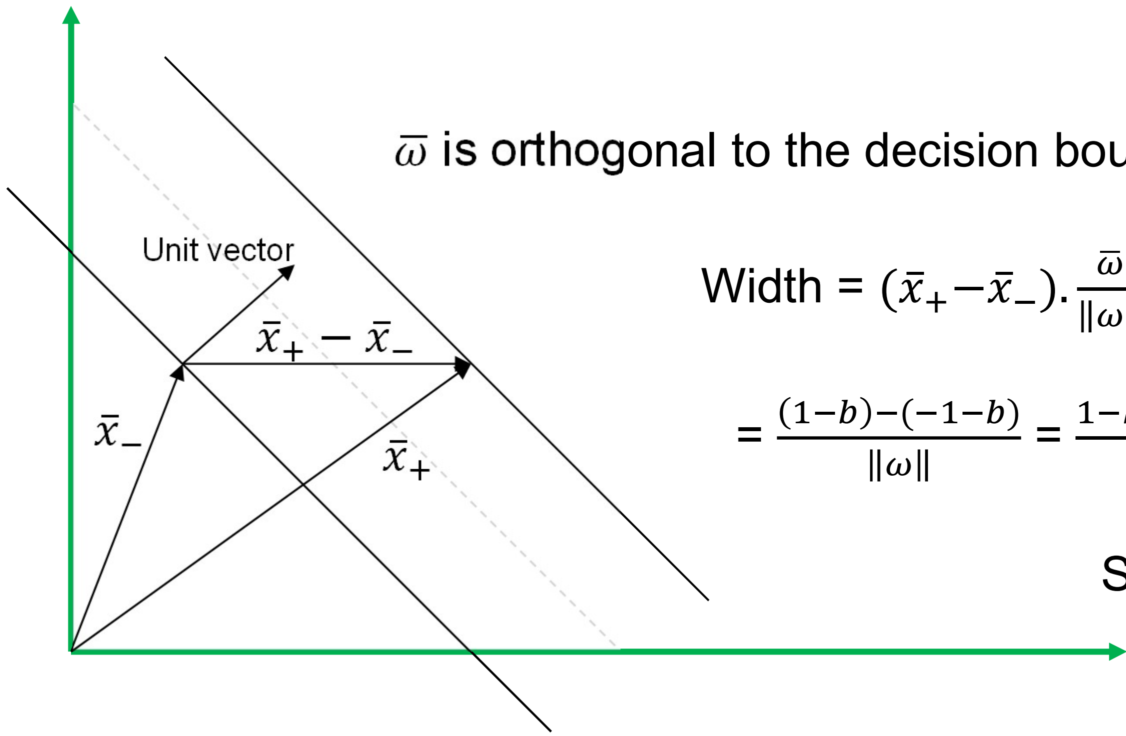
$y_i = -1 \ for - samples$

$$y_i(\bar{\omega}.\bar{x}_i + b) \geq 1$$

$$y_i(\bar{\omega}.\bar{x}_i + b) - 1 \geq 0$$

$if \ y_i(\bar{\omega}.\bar{x}_i + b) - 1 = 0$

Data ($\bar{x}_i$) is on the DB



$\overline{\omega}$ is orthogonal to the decision boundary

Width = $(\bar{x}_+ - \bar{x}_-).\frac{\overline{\omega}}{\|\omega\|} = \frac{\overline{\omega}x_+ - (\overline{\omega}x_-)}{\|\omega\|}$

$= \frac{(1-b)-(-1-b)}{\|\omega\|} = \frac{1-b+1+b}{\|\omega\|} = \frac{2}{\|\omega\|}$

Since $\overline{\omega}.\bar{x}_+ + b \geq 1$

$\overline{\omega}.\bar{x}_- + b \geq -1$

Linear SVM Mathematically

- Goal: **1) Correctly classify all training data**

$$wx_i + b \geq 1 \quad \text{if } y_i = +1$$
$$wx_i + b \leq 1 \quad \text{if } y_i = -1$$
$$y_i(wx_i + b) \geq 1 \quad \text{for all i}$$

2) Maximize the Margin $M = \frac{2}{|w|}$

same as minimize $\frac{1}{2} w^t w$

- **We can formulate a Quadratic Optimization Problem and solve for w and b**

- Minimize $\Phi(w) = \frac{1}{2} w^t w$

  subject to $y_i(wx_i + b) \geq 1 \qquad \forall i$

Solving the Optimization Problem

Find $\mathbf{w}$ and b such that

$\mathbf{\Phi}(\mathbf{w}) = ½\, \mathbf{w}^{\mathrm{T}}\mathbf{w} = \frac{1}{2}\, \|\mathbf{w}\|^2$ is minimized;

and for all $\{(\mathbf{x_i}, y_i)\}$: $y_i\,(\mathbf{w}^{\mathbf{T}}\mathbf{x_i} + b) \geq 1$

- This is a optimization problem for *convex quadratic* objective subject to *linear constraints*.
- Quadratic optimization problems are a well-known class of mathematical programming problems, and can be solved using Quadratic Programming (QP)
- The solution involves constructing a *dual problem* where a *Lagrange multiplier* α_i is associated with every constraint in the primary problem:

Solving the Optimization Problem

- Lagrange duality to get the optimization problem's dual form
  - Allow us to use kernels to get optimal margin classifier to work efficiently in very high dimensional spaces
  - Allow us to derive a very efficient algorithm for solving the above optimization problems

Lagrangian Duality

- The primal problem

$$\begin{aligned} &\mathbf{min}_w \quad f(w) \\ &\mathbf{s.t.}\ g_i(w) \le 0, i = 1,2, \dots \dots k \\ &\qquad h_i(w) = 0, i = 1,2, \dots \dots l \end{aligned}$$

- The generalized Lagrangian

$$L(w, \alpha, \beta) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w) + \sum_{i=1}^{l} \beta_i h_i(w)$$

where the α_i's ($\alpha_i \ge 0$) and $\beta_i{}'s$ are called Lagrange multipliers

Lagrangian Duality

- Lemma:

$$max_{\alpha,\beta,\alpha_{i\geq 0}}\, L(w,\alpha,\beta) = \begin{cases} f(w) & if\ \ w\, satisfies\, the\, primal\, constraints \\ \infty & otherwise \end{cases}$$

- So, the primal can be re-written

$$min_w\ \ max_{\alpha,\beta,\alpha_{i\geq 0}}\, L(w,\alpha,\beta)$$

Lagrangian Duality: Dual Formation

- The primal problem:

$$p^* = min_w \ max_{\alpha,\beta,\alpha_{i\geq 0}} L(w,\alpha,\beta)$$

- The dual problem:

$$d^* = max_{\alpha,\beta,\alpha_{i\geq 0}} min_w \ L(w,\alpha,\beta)$$

- Theorem (weak duality):

$$max_{\alpha,\beta,\alpha_{i\geq 0}} min_w \ L(w,\alpha,\beta) \leq min_w \ max_{\alpha,\beta,\alpha_{i\geq 0}} L(w,\alpha,\beta)$$

i.e. $d^* \leq p^*$

- Theorem (strong duality): iff there exists a saddle point of $L(w,\alpha,\beta)$, we have $d^* = p^*$

The KKT Conditions

- iff there exists some saddle point of $L(w,\alpha,\beta)$, then it satisfies "Karush-Kuhn-Tucker" (KKT) conditions:

$$\frac{\partial}{\partial w_i} L(w,\alpha,\beta) = 0, \qquad i = 1 \dots \dots k$$

$$\frac{\partial}{\partial \beta_i} L(w,\alpha,\beta) = 0, \qquad i = 1 \dots \dots l$$

$$\alpha_i g_i(w) = 0, i = 1, \dots \dots, m$$

$$g_i(w) \le 0, i = 1, \dots \dots, m$$

$$\alpha_i \ge 0, \qquad i = 1, \dots \dots, m$$

- Theorem: If $w^*, \alpha^* and\ b^*$ satisfy the KKT conditions, then it is also a solution to the primal and the dual problem

Solving the Optimization Problem

Find $\mathbf{w}$ and b such that

$\boldsymbol{\Phi}(\mathbf{w}) = ½\ \mathbf{w}^{\mathrm{T}}\mathbf{w} = \frac{1}{2}\ \|\mathbf{w}\|^2$ is minimized;

and for all $\{(\mathbf{x_i}, y_i)\}$: $y_i\,(\mathbf{w^T x_i} + b) \geq 1$

- This is a optimization problem for *convex quadratic* objective subject to *linear constraints*.
- Quadratic optimization problems are a well-known class of mathematical programming problems, and can be solved using Quadratic Programming (QP)
- The solution involves constructing a *dual problem* where a *Lagrange multiplier* α_i is associated with every constraint in the primary problem:

Lagrangian Duality for SVM

- The primal problem

$$\min_w \quad f(w)$$
$$\text{s.t.} \quad g_i(w) \le 0, i = 1,2,\dots\dots k$$
$$h_i(w) = 0, i = 1,2,\dots\dots l$$

- The generalized Lagrangian

$$L(w,\alpha,\beta) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w) + \sum_{i=1}^{l} \beta_i h_i(w)$$

where the α_i's ($\alpha_i \ge 0$) and $\beta_i{}'s$ are called Lagrange multipliers

We don't have the equality constraints, so we only deal with α_is

The KKT Conditions

- iff there exists some saddle point of $L(w, \alpha, \beta)$, then it satisfies "Karush-Kuhn-Tucker" (KKT) conditions:

$$\frac{\partial}{\partial w_i} L(w, \alpha, \beta) = 0, \qquad i = 1 \ldots \ldots k$$

$$\frac{\partial}{\partial \beta_i} L(w, \alpha, \beta) = 0, \qquad i = 1 \ldots \ldots l$$

$$\alpha_i g_i(w) = 0, i = 1, \ldots \ldots, m$$

$$g_i(w) \leq 0, i = 1, \ldots \ldots, m$$

$$\alpha_i \geq 0, \qquad i = 1, \ldots \ldots, m$$

- Theorem: If $w^*, \alpha^* and\ b^*$ satisfy the KKT conditions, then it is also a solution to the primal and the dual problem

Support Vectors

$$\alpha_i g_i(w) = 0, i = 1, \dots \dots, m$$
$$g_i(w) \leq 0, i = 1, \dots \dots, m$$

- If $\alpha_i > 0$, then $g_i(w) = 0$
- Only a few α_i's are non-zero.
- The training data points having non-zero α_i are called support vectors

Solving the SVM Optimization Problem

Find **w** and b such that

$\mathbf{\Phi}(\mathbf{w}) = ½\, \mathbf{w}^T\mathbf{w} = \frac{\mathbf{1}}{\mathbf{2}}\, \|\mathbf{w}\|^{\mathbf{2}}$ is minimized;

and for all $\{(\mathbf{x_i}, y_i)\}$: $y_i\, (\mathbf{w^T x_i} + b) \geq 1$

Quadratic programming with linear constraints.

Lagrangian Function

$$minimize\ L_p(\mathbf{w}, b, \alpha_i) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i (y_i (w^T x_i + b) - 1)$$

$$s.t.\ \alpha_i \geq 0$$

Where p denotes *primal*

Solving the SVM Optimization Problem

$$minimize\ L_p(\boldsymbol{w}, b, \alpha_i) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i(y_i(w^T x_i + b) - 1)$$

$$s.t.\ \alpha_i \geq 0$$

Minimizing L_p w.r.t. $\boldsymbol{w}$ and b with fixed α

$$\frac{\partial L_p}{\partial \boldsymbol{w}} = 0 \longrightarrow w = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i$$

$$\frac{\partial L_p}{\partial b} = 0 \longrightarrow \sum_{i=1}^{n} \alpha_i y_i = 0$$

Solving the SVM Optimization Problem

$$L_p(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \left(\boldsymbol{x}_i^T \boldsymbol{x}_j\right) - b\sum_{i=1}^{m} \alpha_i y_i$$

$$L_p(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \left(\boldsymbol{x}_i^T \boldsymbol{x}_j\right)$$

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

The Dual Problem

- Now we have the following dual optimization problem

$$max_\alpha \, J(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \left(\boldsymbol{x}_i^T \boldsymbol{x}_j\right)$$

$$s.t. \ \alpha_i \geq 0, i = 1, \dots \dots, k$$

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

- This is a quadratic programming problem
  - A global maximum of α_i can always be found

Support Vector Machine

- Once we have Lagrange multipliers $\{\alpha_j\}$, we can reconstruct the parameter vector w as a weighted combination of the training examples

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i \qquad \boldsymbol{w} = \sum_{i \in SV} \alpha_i y_i \boldsymbol{x}_i$$

- For testing with a new data **z**

  Compute $\boldsymbol{w}^T \boldsymbol{z} + b = \sum_{i \in SV} \alpha_i y_i (\boldsymbol{x}_i^T z) + b$

  and classify **z** as class 1 if the sum is positive, and as class 2 otherwise (Note: w need not to be formed explicitly)

Solving the Optimization Problem

- The discriminant function is:

$$g(x) = \boldsymbol{w}^T \boldsymbol{x} + b = \sum_{i \in SV} \alpha_i \boldsymbol{x}_i^T \boldsymbol{x} + b$$

$where\ \boldsymbol{x}_i^T \boldsymbol{x}$ = dot product of $\boldsymbol{x}_i^T$ and $\boldsymbol{x}$

- It relies on a dot product between a test point and the support vectors $\boldsymbol{x}_i$
- Solving the optimization problem involved computing the dot product of $\boldsymbol{x}_i^T \boldsymbol{x}$ between all pairs of training points. It is scalar value.
- The optimal **w** is a linear combination of a small number of data points

The Dual Problem

- Now we have the following dual optimization problem

$$max_{\alpha}\, J(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \left(\boldsymbol{x}_i^T \boldsymbol{x}_j\right)$$

$$s.t.\ \alpha_i \geq 0, i = 1, \dots \dots, k$$

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

- This is a quadratic programming problem
  - A global maximum of α_i can always be found

to continue...

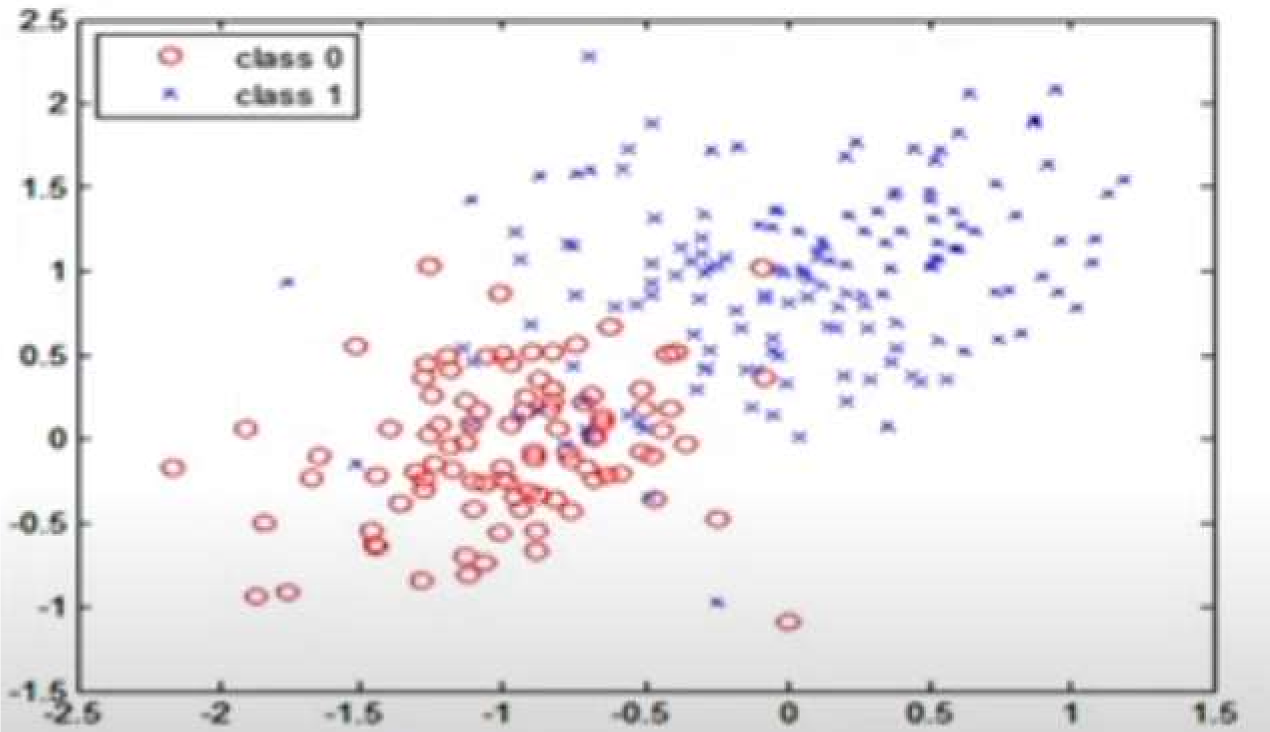
Linear SVM Formulation

Find $\mathbf{w}$ and b such that
$\mathbf{\Phi}(\mathbf{w}) = \frac{1}{2}\, \mathbf{w}^{\mathrm{T}}\mathbf{w} = \frac{1}{2}\, \|\mathbf{w}\|^2$ is minimized;
and for all $\{(\mathbf{x_i}, y_i)\}$: $y_i\, (\mathbf{w^T x_i} + b) \geq 1$

Find $\alpha_1 \ldots \alpha_N$ such that
$\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j}$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $\alpha_i \geq 0$ for all α_i

Assumption: Training examples are linearly separable

Limitations of Linear SVM



- What if data is not linearly separable?

Or

- Noisy data points?

Extend the definition of maximum margin to allow the non-separating planes.

The Optimization Problem Solution

- The solution has the form:

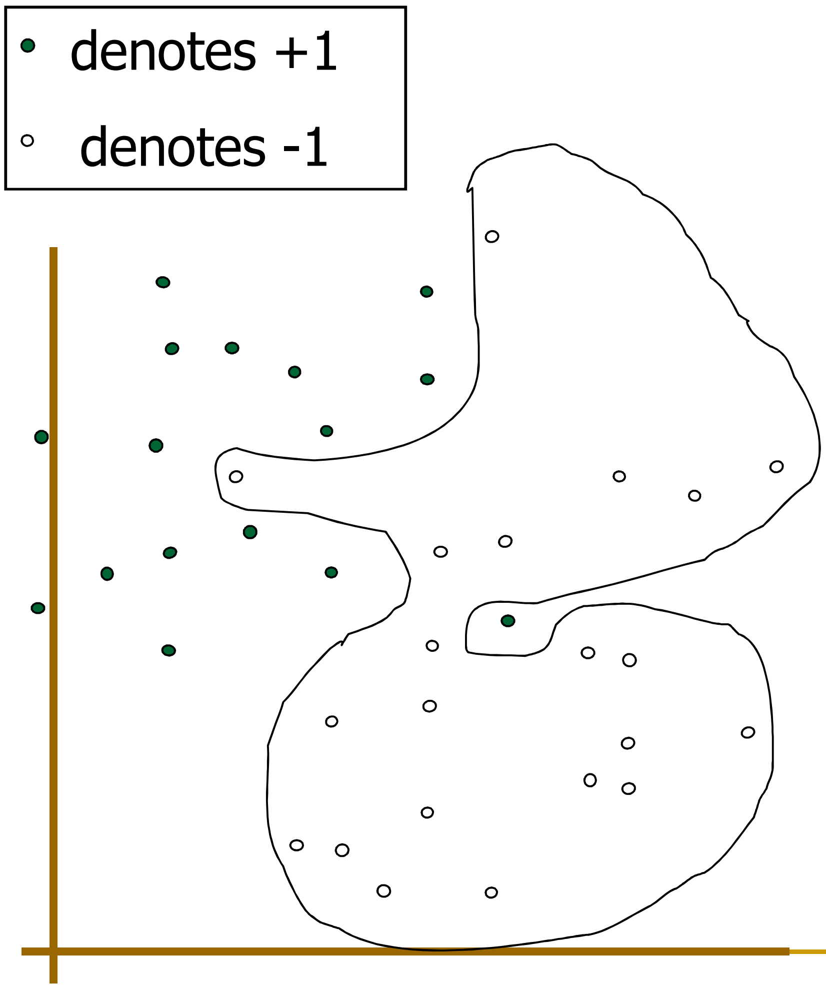
$$\mathbf{w} = \Sigma\alpha_i y_i \mathbf{x_i} \qquad b = y_k - \mathbf{w}^{\mathrm{T}}\mathbf{x_k} \text{ for any } \mathbf{x_k} \text{ such that } \alpha_k \neq 0$$

- Each non-zero α_i indicates that corresponding $\mathbf{x_i}$ is a support vector.
- Then the classifying function will have the form:

$$f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x_i}^{\mathrm{T}}\mathbf{x} + b$$

- Notice that it relies on an *inner product* between the test point **x** and the support vectors $\mathbf{x_i}$ – we will return to this later.
- Also keep in mind that solving the optimization problem involved computing the inner products $\mathbf{x_i}^{\mathrm{T}}\mathbf{x_j}$ between all pairs of training points.

Dataset with noise



- **Hard Margin: So far we require all data points be classified correctly**

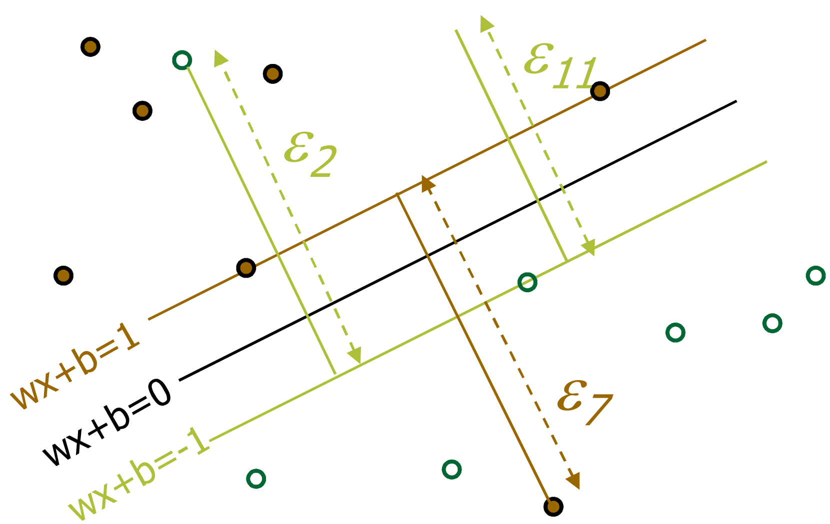
  - No training error
- **What if the training set is noisy?**

  - Solution 1: use very powerful kernels

OVERFITTING!

Soft Margin Classification

***Slack variables ξi* can be added to allow misclassification of difficult or noisy examples.**



What should our quadratic optimization criterion be?

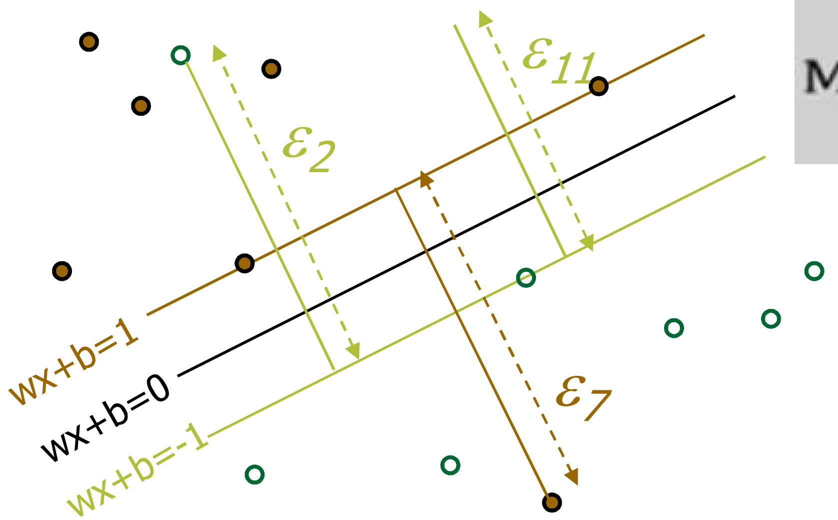
Minimize

$$\frac{1}{2}\mathbf{w}.\mathbf{w} + C \text{ . \# Training errors}$$

$$\frac{1}{2}\mathbf{w}.\mathbf{w} + C\sum_{k=1}^{R}\varepsilon_k$$

The problem is that it is no longer a quadratic optimization problem and we can’t use QP solver to solve it.

Soft Margin Classification



$$M = \frac{2}{\sqrt{w.w}}$$

Minimize

$$w.w + C\sum_{k=1}^{m} \xi_k$$

m constraints

$$\left\{ \begin{array}{l} w.x_k + b \geq 1 - \xi_k \text{ if } y_k = 1 \\ w.x_k + b \leq -1 + \xi_k \text{ if } y_k = -1 \end{array} \right\}$$

$$\equiv$$

$$y_k(w.x_k + b) \geq 1 - \xi_k, \quad \text{k=1,...,m}$$

$$\xi_k \geq 0, \quad \text{k=1,...,m}$$

- C controls the relative importance of maximizing the margins and fitting the training data.
- Controls overfitting

Lagrangian

$$L(w, b, \xi, \alpha, \beta)$$
$$= \frac{1}{2} w.w + C \sum_{i=1}^{m} \xi_i$$
$$+ \sum_{i=1}^{m} \alpha_i [y_i (x.w + b) - 1 + \xi_i] - \sum_{i=1}^{m} \beta_i \xi_i$$

α_i's and β_i's are Lagrange multipliers (≥ 0).

Dual Formulation

Find $\alpha_1, \alpha_2, \ldots, \alpha_m$ s.t.

$$\max_{\alpha} \mathrm{J}(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

Linear SVM

$$\text{s.t.} \quad \alpha_i \geq 0, \quad i = 1, \ldots, m$$
$$\sum_{i=1}^{m} \alpha_i y_i = 0.$$

Noise Accounted

$$\text{s.t.} \quad 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, m$$
$$\sum_{i=1}^{m} \alpha_i y_i = 0.$$

Soft Margin Classifier

- x_i with non-zero α_i will be support vectors.
- Solution to the dual problem is:

$$w = \sum_{i=1}^{m} \alpha_i y_i x_i$$

$$b = y_k(1 - \xi_k) - \sum_{i=1}^{m} \alpha_i y_i x_i x_k$$

for any k s.t. $\alpha_k > 0$

For classification,

$$f(x) = \sum_{i=1}^{m} \alpha_i y_i x_i . x \ + b$$

(no need to compute w explicitly)

Hard Margin v.s. Soft Margin

- **The old formulation:**

> Find $\mathbf{w}$ and b such that
> $\mathbf{\Phi}(\mathbf{w}) = \frac{1}{2}\, \mathbf{w}^T\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
> $y_i\, (\mathbf{w^T x_i} + \mathrm{b}) \geq 1$

- **The new formulation incorporating slack variables:**

> Find $\mathbf{w}$ and b such that
> $\mathbf{\Phi}(\mathbf{w}) = \frac{1}{2}\, \mathbf{w}^T\mathbf{w} + C\Sigma\xi_i$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
> $y_i\, (\mathbf{w^T x_i} + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i

- **Parameter *C* can be viewed as a way to control overfitting.**

Linear SVMs: Overview

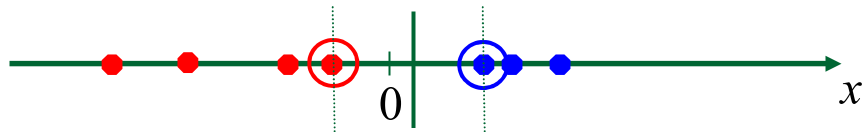
- **The classifier is a *separating hyperplane.***
- **Most "important" training points are support vectors; they define the hyperplane.**
- **Quadratic optimization algorithms can identify which training points $\mathbf{x}_i$ are support vectors with non-zero Lagrangian multipliers α_i.**
- **Both in the dual formulation of the problem and in the solution training points appear only inside dot products:**

Find $\alpha_1 \ldots \alpha_N$ such that
$Q(\alpha) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $0 \leq \alpha_i \leq C$ for all α_i

$$f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x}_i^T\mathbf{x} + b$$

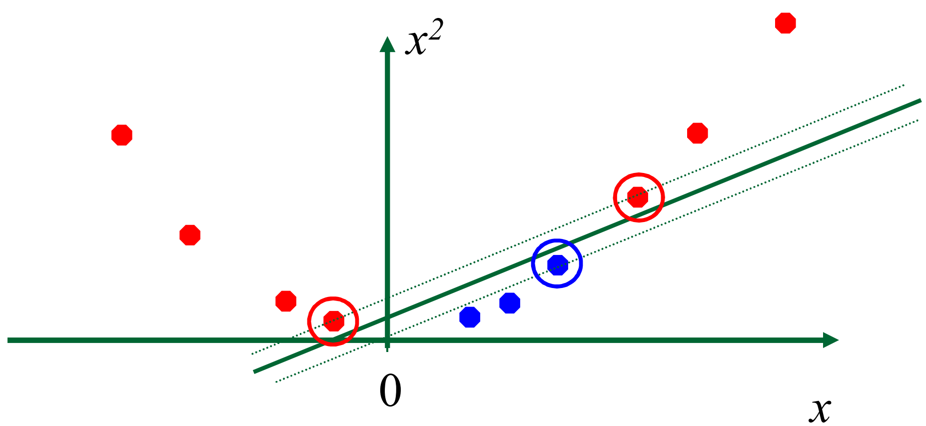
Non-linear SVMs

- Datasets that are linearly separable with some noise work out great:



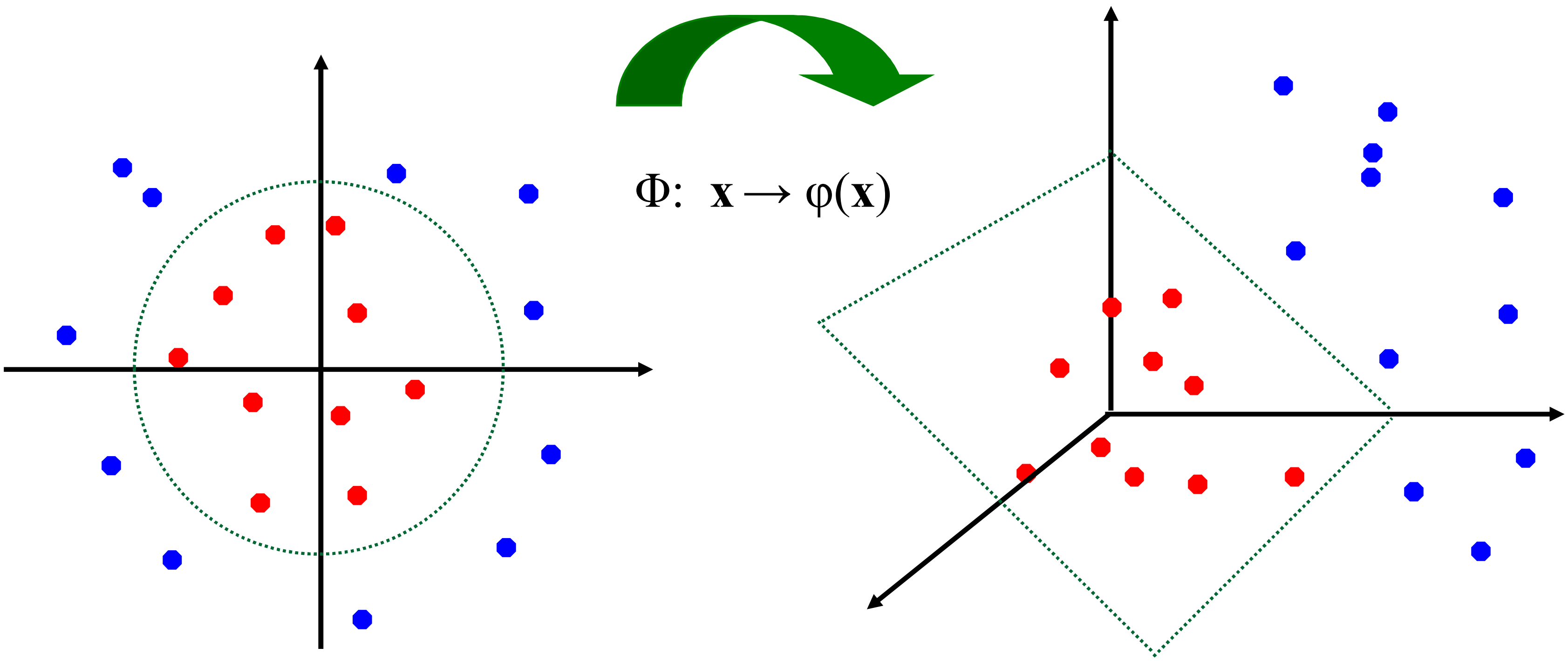
- But what are we going to do if the dataset is just too hard?

- How about… mapping data to a higher-dimensional space:



Non-linear SVMs: Feature spaces

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



Non- Linear SVM

- Computational cost

Find $\alpha_1 \ldots \alpha_N$ such that
$Q(\alpha) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \boxed{x_i^T x_j}$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $0 \le \alpha_i \le C$ for all α_i

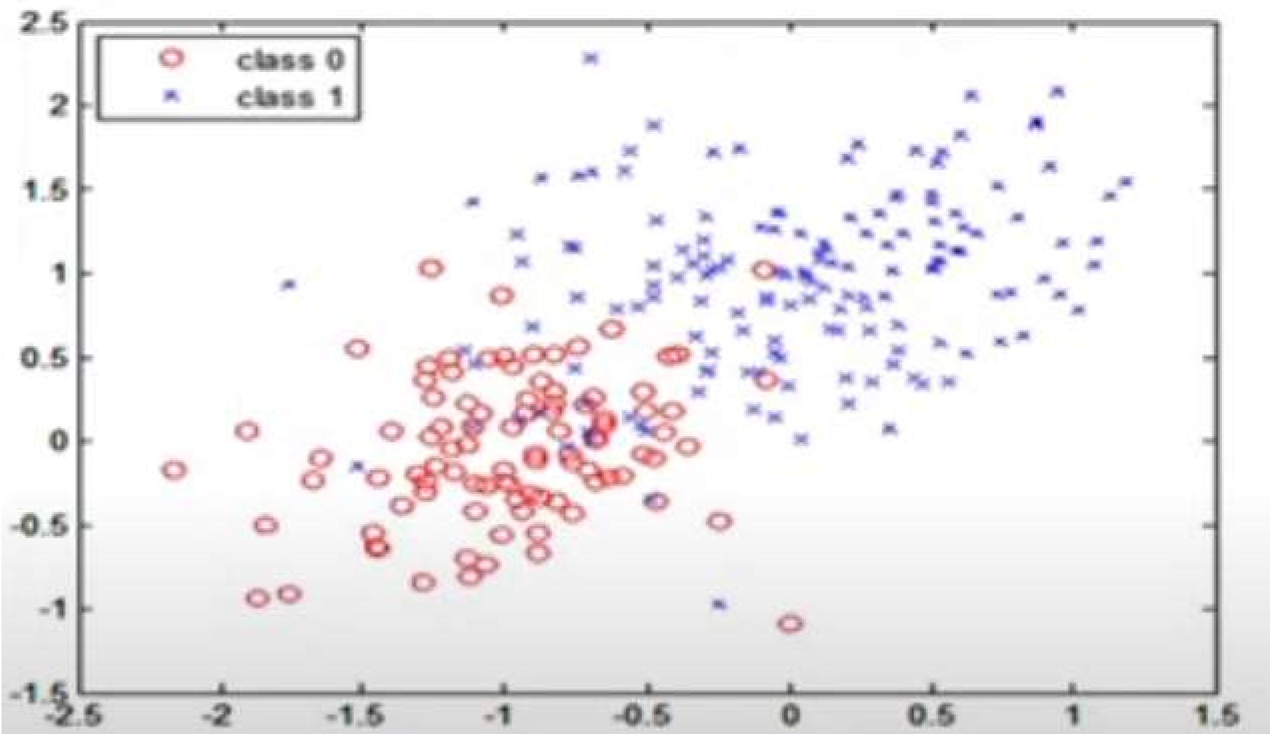
$O(d^2)$

Find $\alpha_1 \ldots \alpha_N$ such that
$Q(\alpha) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \boxed{\varphi(x_i^T)\varphi(x_j)}$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $0 \le \alpha_i \le C$ for all α_i

$O(D^2)$

If D>>d, computational cost will increases substantially

Limitations of Linear SVM



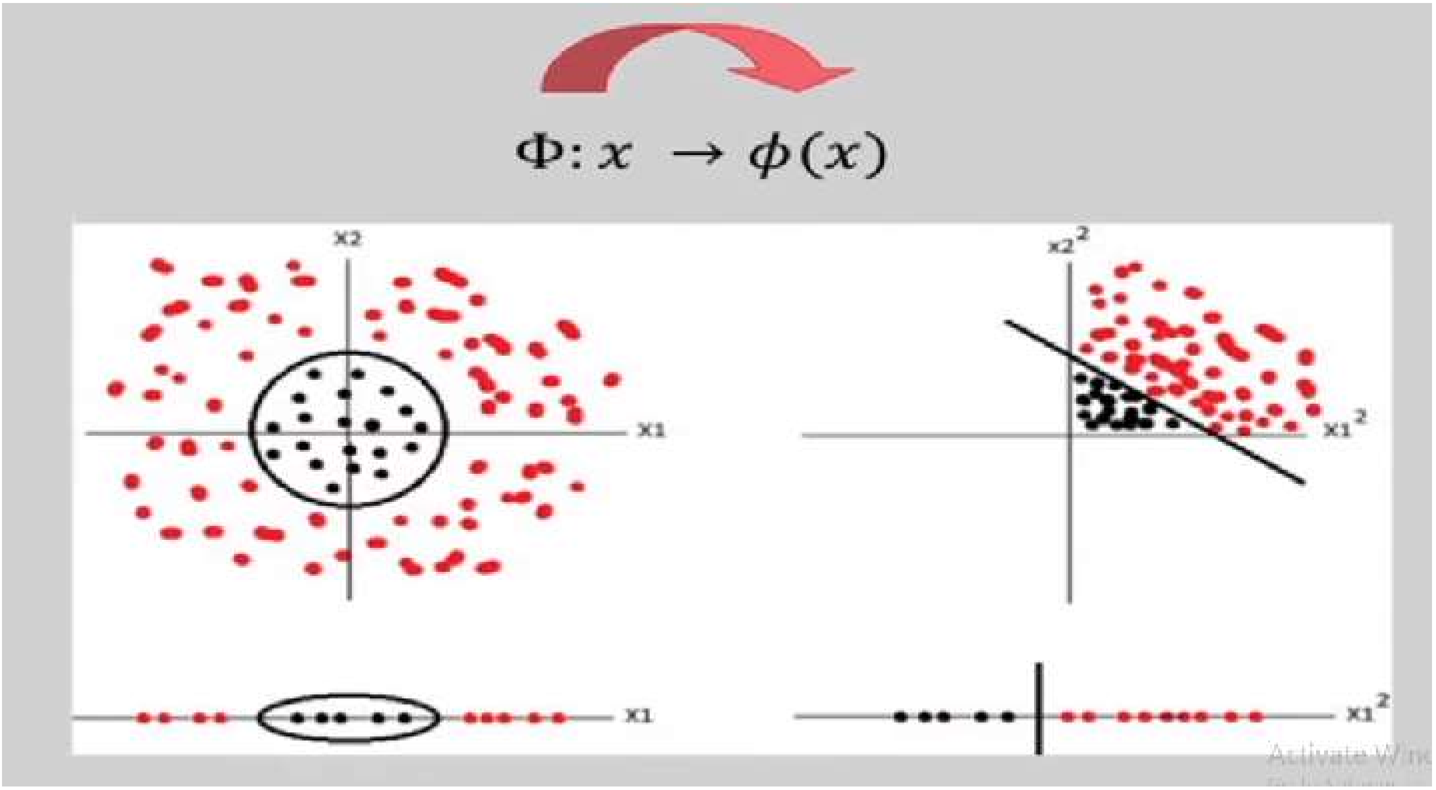
- What if data is not linearly separable?

Or

- Noisy data points?

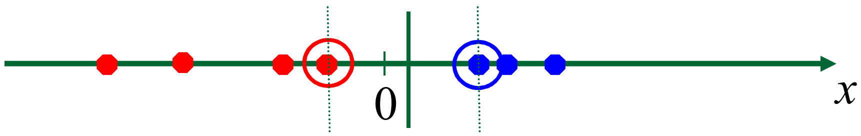
Extend the definition of maximum margin to allow the non-separating planes.

Non Linear SVM: Feature Space

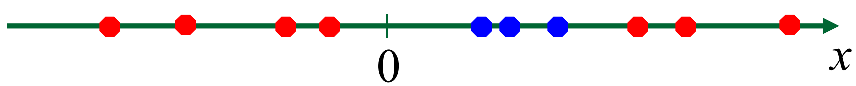


Non-linear SVMs

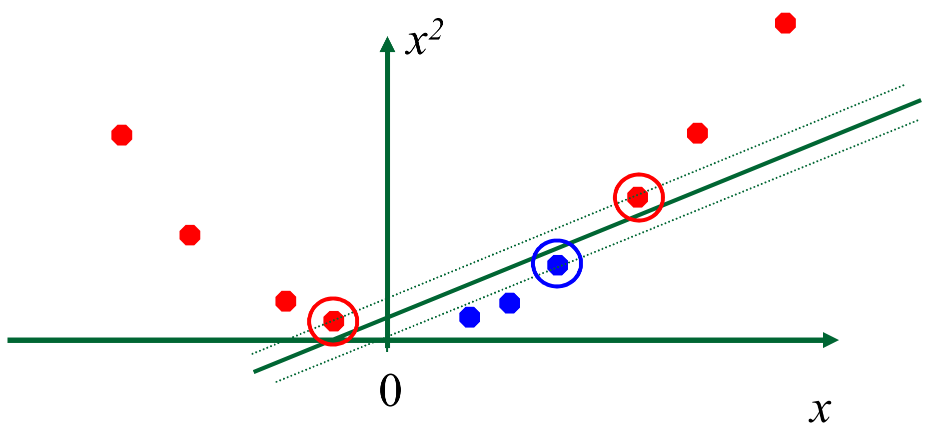
- Datasets that are linearly separable with some noise work out great:



- But what are we going to do if the dataset is just too hard?

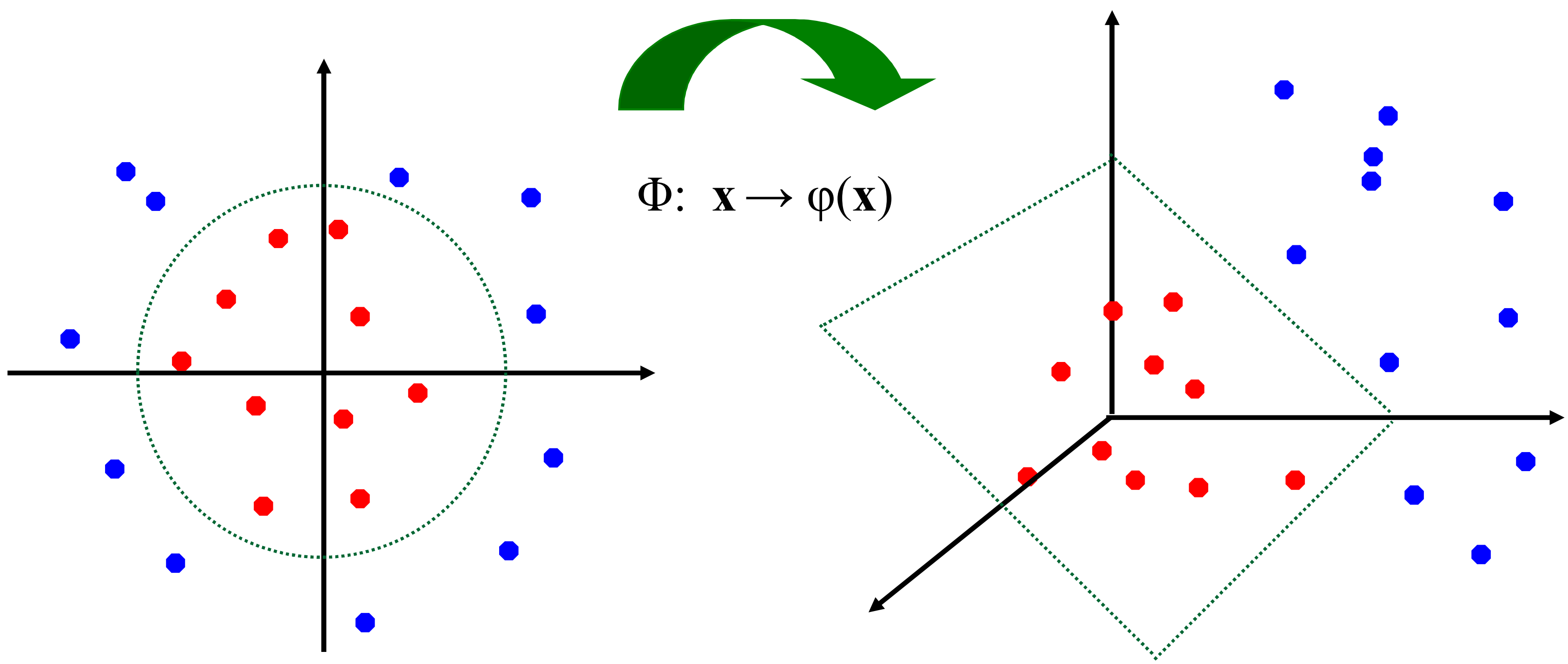


- How about… mapping data to a higher-dimensional space:



Non-linear SVMs: Feature spaces

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



The “Kernel Trick”

- **The linear classifier relies on dot product between vectors $K(\mathbf{x}_i,\mathbf{x}_j)=\mathbf{x}_i^T\mathbf{x}_j$**
- **If every data point is mapped into high-dimensional space via some transformation Φ: x → φ(x), the dot product becomes:**

$$K(\mathbf{x}_i,\mathbf{x}_j)= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$$

- **A *kernel function* is some function that corresponds to an inner product in some expanded feature space.**
- **Example:**

2-dimensional vectors x=[x_1 x_2]; let $K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^T\mathbf{x}_j)^2$,

Need to show that $K(\mathbf{x}_i,\mathbf{x}_j)= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$:

$$
\begin{aligned}
K(\mathbf{x}_i,\mathbf{x}_j)&=(1 + \mathbf{x}_i^T\mathbf{x}_j)^2, \\
&= 1+ x_{i1}^2x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2} \\
&= [1\ \ x_{i1}^2\ \ \sqrt{2}\, x_{i1}x_{i2}\ \ x_{i2}^2\ \ \sqrt{2}x_{i1}\ \ \sqrt{2}x_{i2}]^T\, [1\ \ x_{j1}^2\ \ \sqrt{2}\, x_{j1}x_{j2}\ \ x_{j2}^2\ \ \sqrt{2}x_{j1}\ \ \sqrt{2}x_{j2}] \\
&= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j), \quad \text{where } \varphi(\mathbf{x}) = [1\ \ x_1^2\ \ \sqrt{2}\, x_1x_2\ \ x_2^2\ \ \sqrt{2}x_1\ \ \sqrt{2}x_2]
\end{aligned}
$$

What Functions are Kernels?

- **For some functions $K(x_i,x_j)$ checking that**

  $K(x_i,x_j)= \varphi(x_i)^T\varphi(x_j)$ can be cumbersome.

- **Mercer's theorem:**

  Every semi-positive definite symmetric function is a kernel

- **Semi-positive definite symmetric functions correspond to a semi-positive definite symmetric Gram matrix:**

K=

$K(\mathbf{x}_1,\mathbf{x}_1)$	$K(\mathbf{x}_1,\mathbf{x}_2)$	$K(\mathbf{x}_1,\mathbf{x}_3)$	…	$K(\mathbf{x}_1,\mathbf{x}_N)$
$K(\mathbf{x}_2,\mathbf{x}_1)$	$K(\mathbf{x}_2,\mathbf{x}_2)$	$K(\mathbf{x}_2,\mathbf{x}_3)$		$K(\mathbf{x}_2,\mathbf{x}_N)$
…	…	…	…	…
$K(\mathbf{x}_N,\mathbf{x}_1)$	$K(\mathbf{x}_N,\mathbf{x}_2)$	$K(\mathbf{x}_N,\mathbf{x}_3)$	…	$K(\mathbf{x}_N,\mathbf{x}_N)$

Examples of Kernel Functions

- Linear: $K(\mathbf{x_i},\mathbf{x_j}) = \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j}$

- Polynomial of power p: $K(\mathbf{x_i},\mathbf{x_j}) = (1 + \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j})^p$

- Gaussian (radial-basis function network):

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\frac{\|\mathbf{x_i} - \mathbf{x_j}\|^2}{2\sigma^2}\right)$$

- Sigmoid: $K(\mathbf{x_i},\mathbf{x_j}) = \tanh(\beta_0 \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j} + \beta_1)$

Kernel Functions

- Kernel function can be thought of a similarity measure between the input objects.

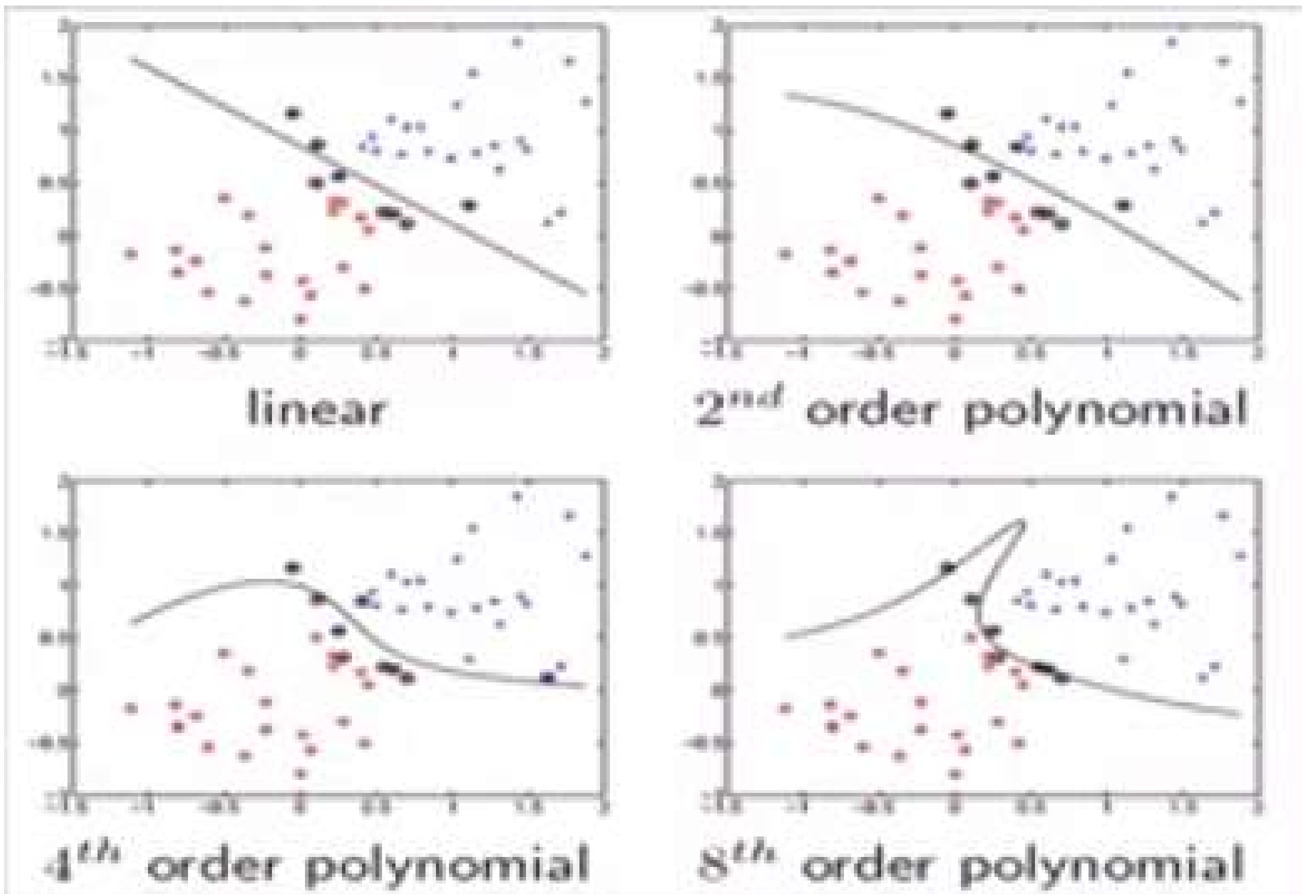
- Not all similarity measure can be used as Kernel function

- Mercer's condition state that any positive semi-definite kernel K(X,Y) i.e.

$$\sum_{i,j} K(x_i x_j) c_i c_j \geq 0$$

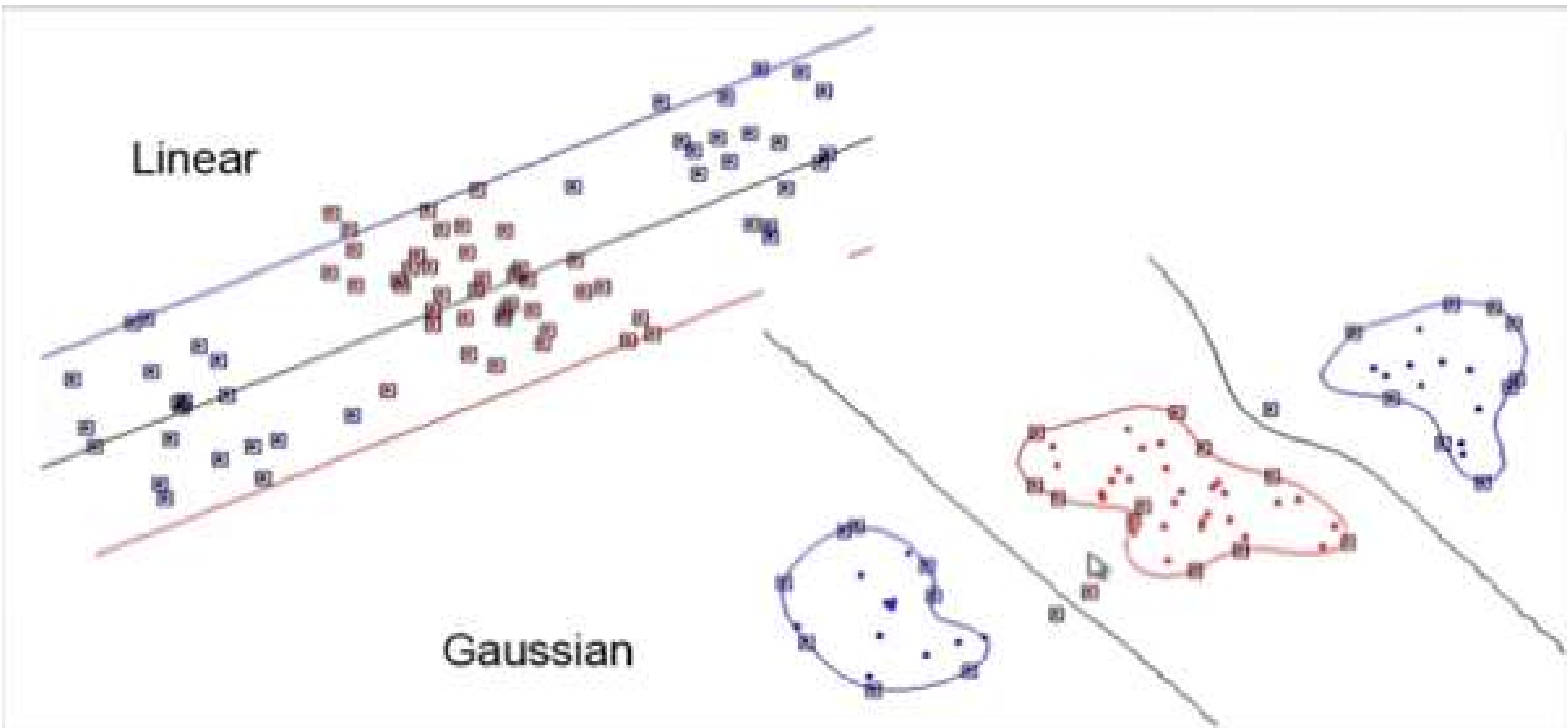
- Can be expressed as a dot product in a high dimensional space.

SVM Examples



Non Linear SVM

Gaussian Kernels



Non-linear SVMs Mathematically

- **Dual problem formulation:**

Find $\alpha_1 \ldots \alpha_N$ such that
$Q(\alpha) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j K(\mathrm{x}_i, \mathrm{x}_j)$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $\alpha_i \geq 0$ for all α_i

- **The solution is:**

$$f(\mathrm{x}) = \Sigma\alpha_i y_i K(\mathrm{x}_i, \mathrm{x}_j) + b$$

- **Optimization techniques for finding α_i's remain the same!**

Nonlinear SVM - Overview

- SVM locates a separating hyperplane in the feature space and classify points in that space

- It does not need to represent the space explicitly, simply by defining a kernel function

- The kernel function plays the role of the dot product in the feature space.

SVM: Summary

- Flexibility in choosing a similarity function

- Sparseness of solution when dealing with large data sets
  - only support vectors are used to specify the separating hyperplane

- Ability to handle large feature spaces
  - complexity does not depend on the dimensionality of the feature space

- Overfitting can be controlled by soft margin approach

- Nice math property: a simple convex optimization problem which is guaranteed to converge to a single global solution

- Feature Selection

SVM Applications

- SVM has been used successfully in many real-world problems

  - text (and hypertext) categorization

  - image classification

  - bioinformatics (Protein classification, Cancer classification)

  - hand-written character recognition

Weakness of SVM

- It is sensitive to noise

  - A relatively small number of mislabeled examples can dramatically decrease the performance

- It only considers two classes

  - how to do multi-class classification with SVM?

  - Answer:

  1) with output arity m, learn m SVM's

    - SVM 1 learns "Output==1" vs "Output != 1"
    - SVM 2 learns "Output==2" vs "Output != 2"
    - :
    - SVM m learns "Output==m" vs "Output != m"

  2)To predict the output for a new input, just predict with each SVM and find out which one puts the prediction the furthest into the positive region.

to continue…