Lecture Slides for

INTRODUCTION TO

Machine Learning

ETHEM ALPAYDIN
© The MIT Press, 2004

alpaydin@boun.edu.tr
http://www.cmpe.boun.edu.tr/~ethem/i2ml

CHAPTER 10:

Linear Discrimination

CHAPTER 13:

Support Vector Machines

Likelihood- vs. Discriminant-based Classification

- Likelihood-based: Assume a model for $p(\mathbf{x}|C_i)$, use Bayes' rule to calculate $P(C_i|\mathbf{x})$

  Choose C_i *if* $g_i(\mathbf{x}) = \log P(C_i|\mathbf{x})$ is maximum

- Discriminant-based: Assume a model for the discriminant $g_i(\mathbf{x}|\Phi_i)$; no density estimation
  - Estimating the boundaries is enough; no need to accurately estimate the densities inside the boundaries

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Linear Discriminant

- Linear discriminant:

$$g_i(\boldsymbol{x} \mid \boldsymbol{w}_i, w_{i0}) = \boldsymbol{w}_i^T \boldsymbol{x} + w_{i0} = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$$

- Advantages:
  - Simple: O(d) space/computation
  - Knowledge extraction: Weighted sum of attributes; positive/negative weights, magnitudes (credit scoring)
  - Optimal when $p(\boldsymbol{x}|C_i)$ are Gaussian with shared cov matrix; useful when classes are (almost) linearly separable

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Generalized Linear Model

- **Quadratic discriminant** (O(d^2))

$$g_i(\boldsymbol{x} \mid \mathbf{W}_i, \boldsymbol{w}_i, w_{i0}) = \boldsymbol{x}^T \mathbf{W}_i \boldsymbol{x} + \boldsymbol{w}_i^T \boldsymbol{x} + w_{i0}$$

- Instead of higher complexity, we can still use a linear classifier if we use higher-order (product) terms.

- Map from ***x*** to ***z*** using nonlinear basis functions and use a linear discriminant in ***z***-space

$$z_1 = x_1, \; z_2 = x_2, \; z_3 = x_1^2, \; z_4 = x_2^2, \; z_5 = x_1 x_2$$

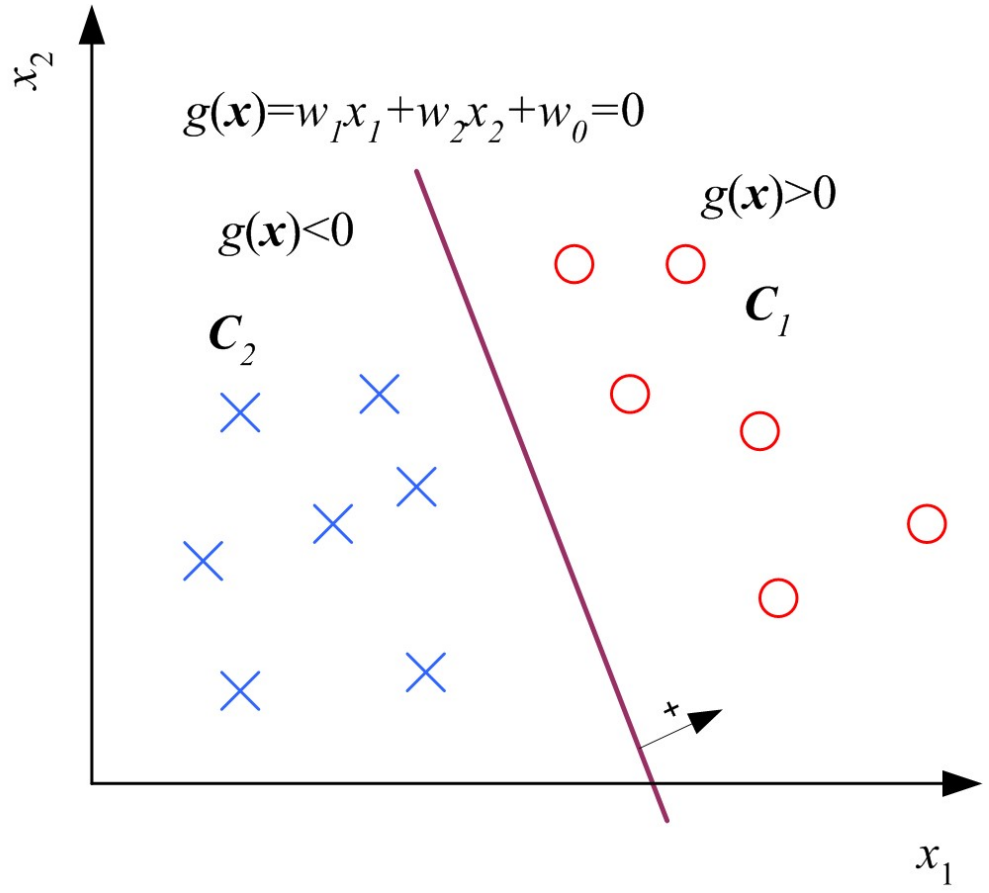
- The linear function defined in the **z** space corresponds to a non-linear function in the **x** space.

$$g_i(\boldsymbol{x}) = \sum_{j=1}^{k} w_{ij} \phi_j(\boldsymbol{x})$$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Two Classes

choose C_1 if $g_1(x) > g_2(x)$
C_2 if $g_2(x) > g_1(x)$



Define:

$$g(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x})$$
$$= \left(\boldsymbol{w}_1^T\boldsymbol{x} + w_{10}\right) - \left(\boldsymbol{w}_2^T\boldsymbol{x} + w_{20}\right)$$
$$= \left(\boldsymbol{w}_1 - \boldsymbol{w}_2\right)^T\boldsymbol{x} + \left(w_{10} - w_{20}\right)$$
$$= \boldsymbol{w}^T\boldsymbol{x} + w_0$$

$$\text{choose } \begin{cases} C_1 & \text{if } g(\boldsymbol{x}) > 0 \\ C_2 & \text{otherwise} \end{cases}$$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

A Bit of Geometry

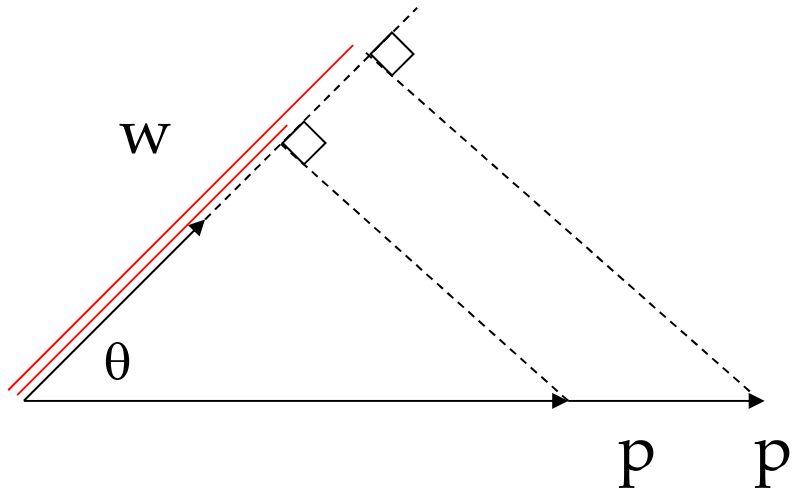
- Dot Product and Projection

- $\langle w,p \rangle = w^T p = \|w\|\|p\|\text{Cos}\theta$

- proj. of p onto w

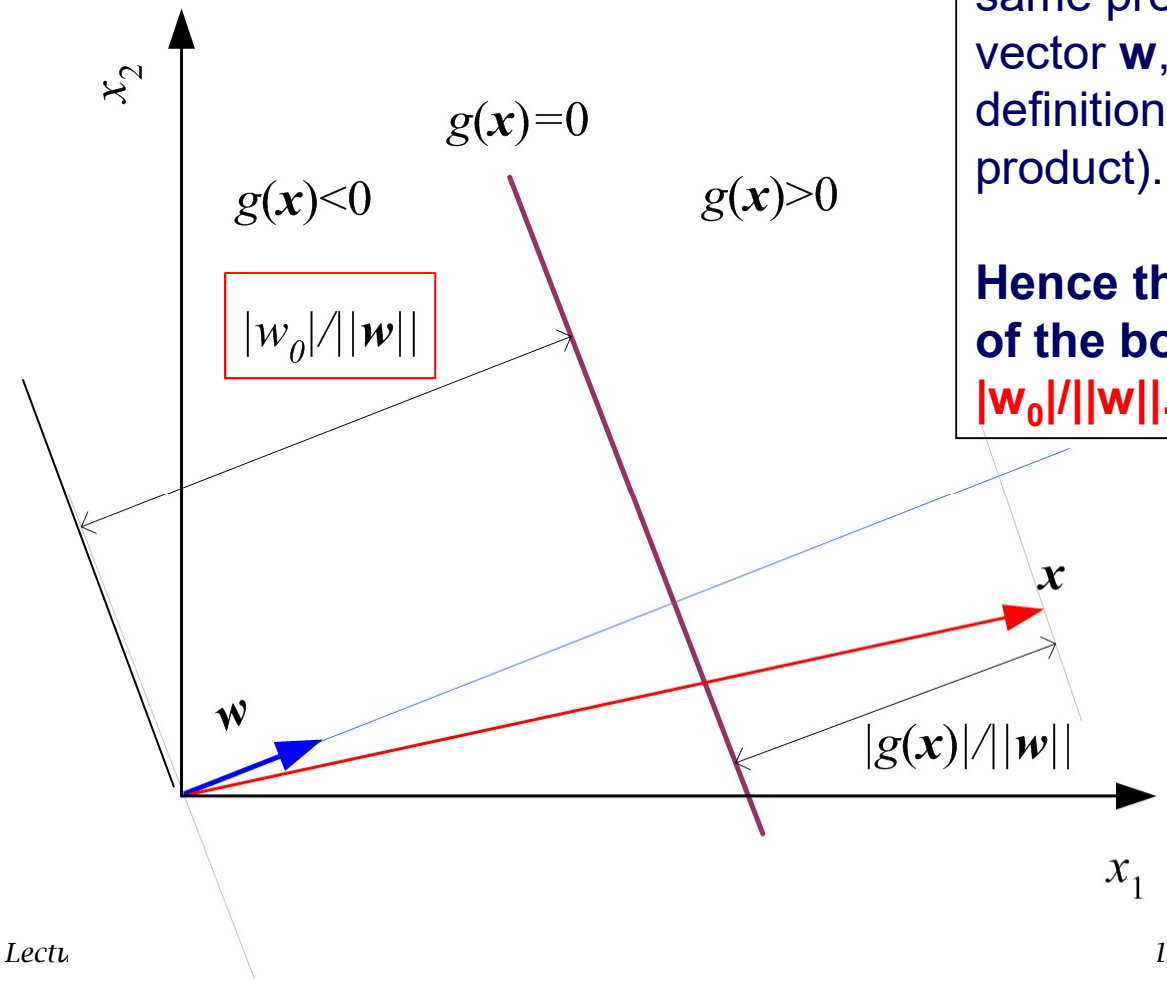
$$= \|p\|\text{Cos}\theta$$

$$= w^T.p/\|w\|$$



Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Geometry



The points **x** on the separating hyperplane have $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x}+w_0=0$. Hence for the points on the boundary $\mathbf{w}^T\mathbf{x}=-w_0$.

Thus, these points also have the same projection onto the weight vector **w**, namely $\mathbf{w}^T\mathbf{x}/||\mathbf{w}||$ (by definition of projection and dot product). But this is equal to $-w_0/||\mathbf{w}||$.

Hence the perpendicular distance of the boundary to the origin is $|w_0|/||\mathbf{w}||$.

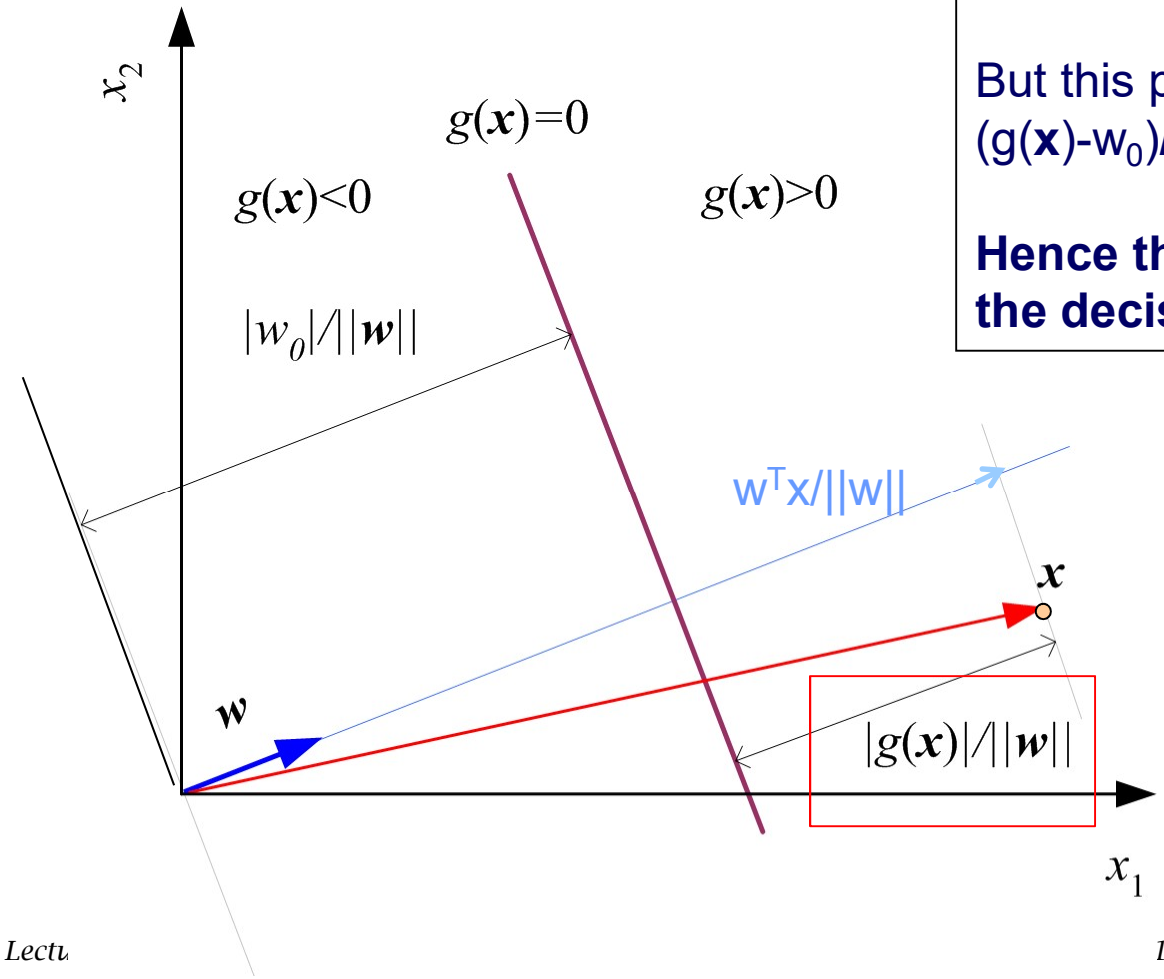
Geometry

In general, any point **x** have $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x}+w_0$.

When we consider **x**'s projection onto **w**, we have $\mathbf{w}^T\mathbf{x}/||\mathbf{w}||$ by definition of projection and dot product.

But this projection is equal to $(g(\mathbf{x})-w_0)/||\mathbf{w}||$ since $\mathbf{w}^T\mathbf{x}= g(\mathbf{x})-w_0$.

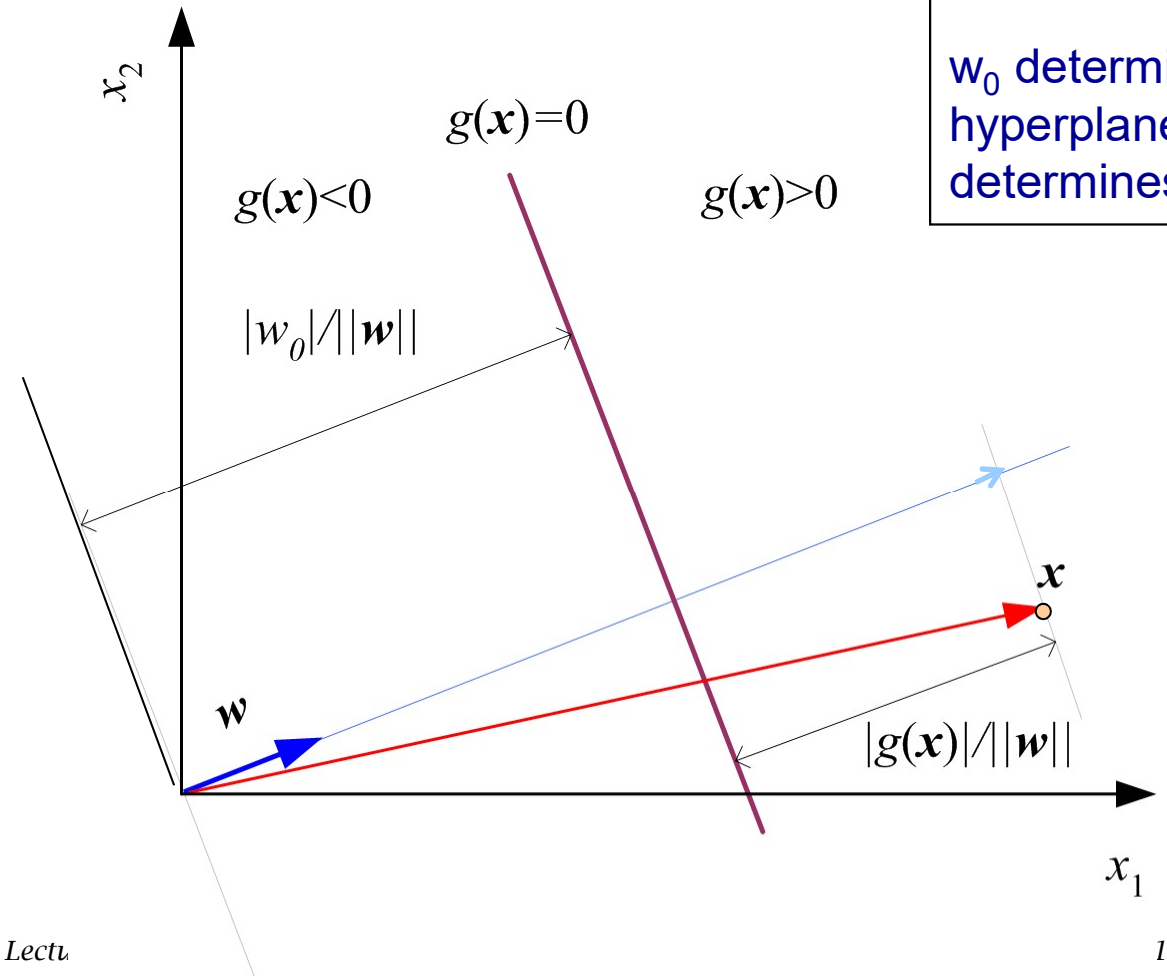
Hence the distance of any point x to the decision boundary is $|g(x)|/||w||$.



The distance of any point x to the decision boundary is g(x)/||w||

Positive if x is on the positive side
Negative if x is on the negative side

w_0 determines the location of the hyperplane w.r.t the origin and **w** determines its orientation



How to Deal with Multi-class Problems

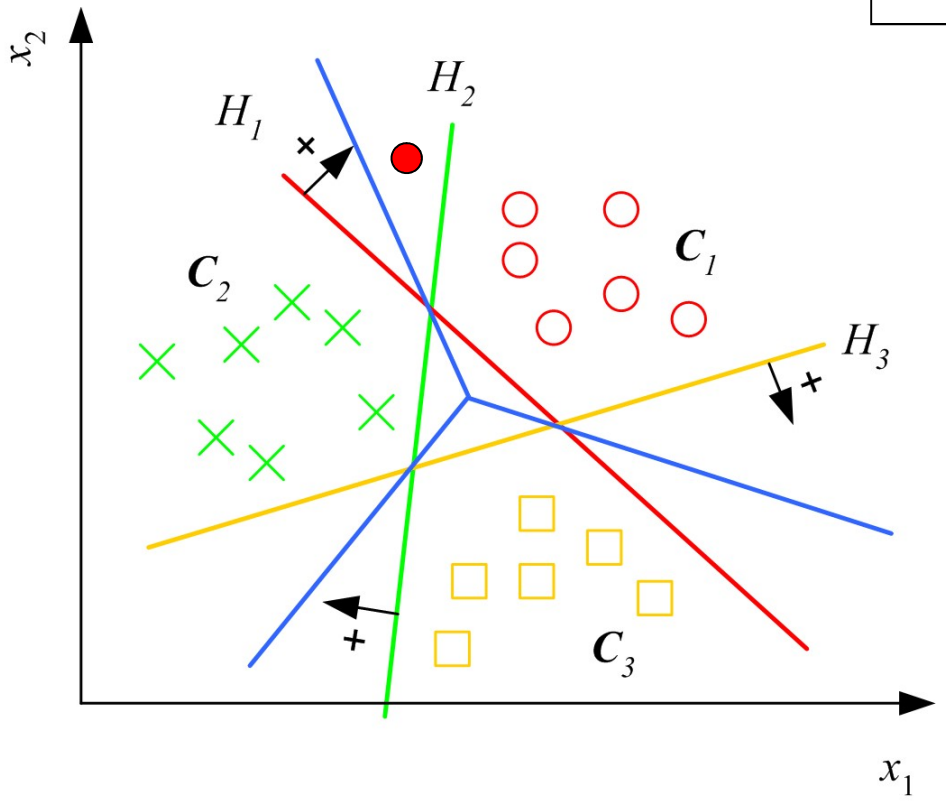
- When K > 2
  - Combine K two-class problems, each one separating one class from all other classes

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Multiple Classes

$$g_i(\boldsymbol{x} \mid \boldsymbol{w}_i, w_{i0}) = \boldsymbol{w}_i^T \boldsymbol{x} + w_{i0}$$

How to train?
How to decide on a test?



Choose C_i if

$$g_i(\boldsymbol{x}) = \max_{j=1}^{K} g_j(\boldsymbol{x})$$

Convex decision regions based on g_is (indicated with blue)
dist is $|g_i(x)|/||wi||$

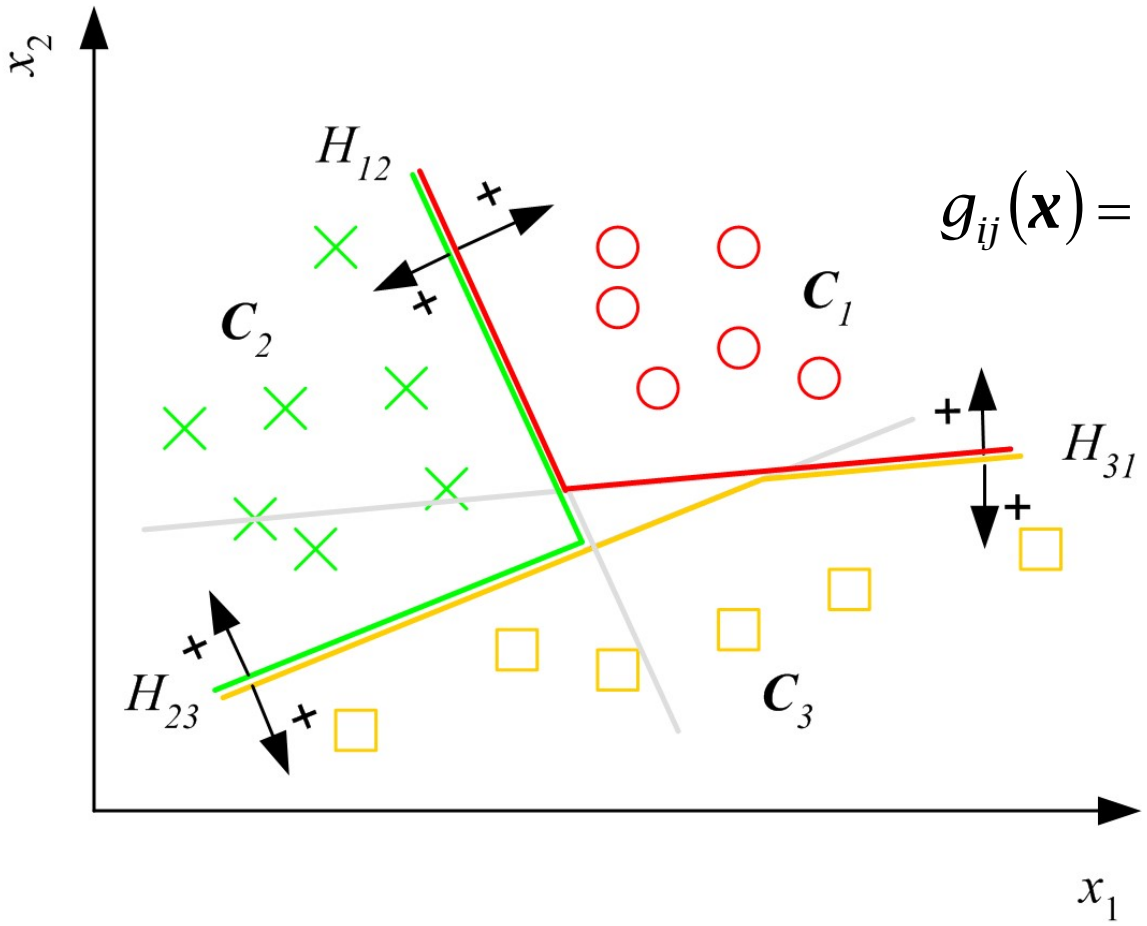
Assumes that classes are linearly separable:
reject may be used

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Pairwise Separation

If the classes are not linearly separable:

$$g_{ij}\left(\boldsymbol{x} \mid \boldsymbol{w}_{ij}, w_{ij0}\right) = \boldsymbol{w}_{ij}^T \boldsymbol{x} + w_{ij0}$$



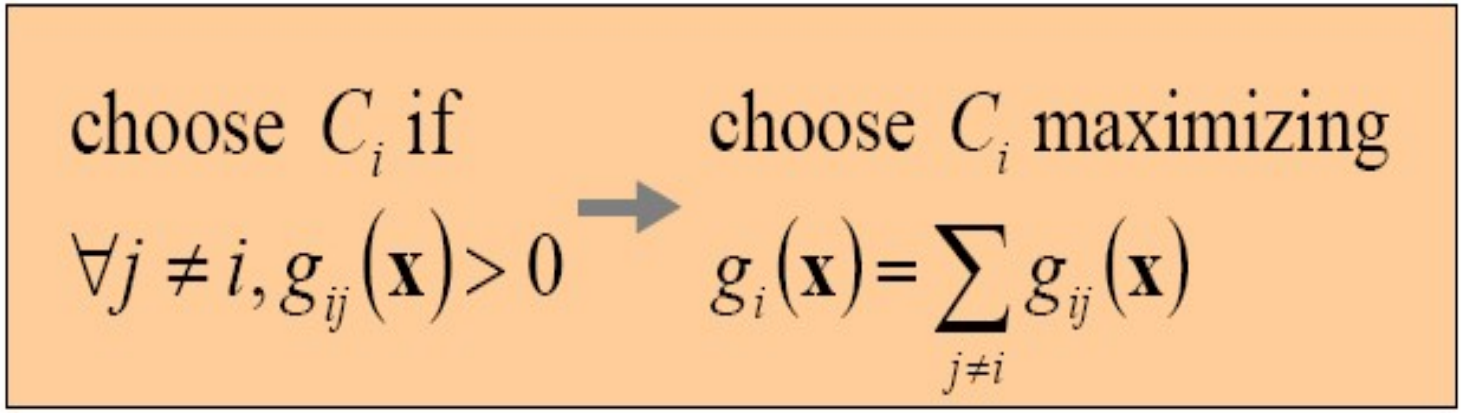
$$g_{ij}(\boldsymbol{x}) = \begin{cases} > 0 & \text{if } \boldsymbol{x} \in C_i \\ \leq 0 & \text{if } \boldsymbol{x} \in C_j \\ \text{don't care} & \text{otherwise} \end{cases}$$

choose C_i if $\forall j \neq i, g_{ij}(\boldsymbol{x}) > 0$

Uses k(k-1)/2 linear discriminants

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

- None of the classes may satisfy that condition
- Use instead:



Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Learning the Discriminants

As we have seen before, when $p(\boldsymbol{x} \mid C_i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma})$, the optimal discriminant is a linear one:

$$g_i\left(\boldsymbol{x} \mid \boldsymbol{w}_i, w_{i0}\right) = \boldsymbol{w}_i^T \boldsymbol{x} + w_{i0}$$

$$\boldsymbol{w}_i = \Sigma^{-1}\mu_i \quad w_{i0} = -\frac{1}{2}\mu_i^T \Sigma^{-1}\mu_i + \log P\left(C_i\right)$$

So, estimate μ_i and Σ from data, and plug into the g_i's to find the linear discriminant functions.

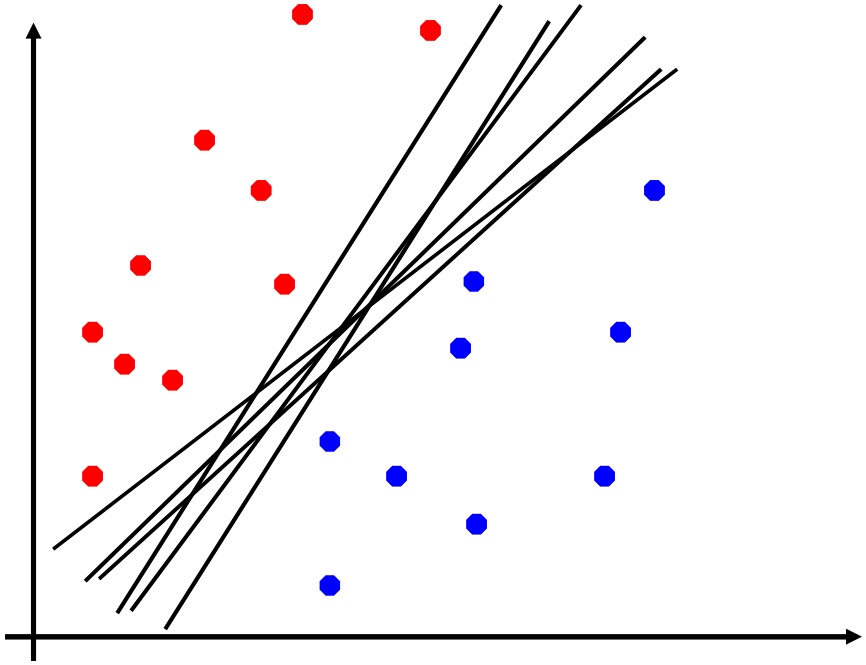
Of course any way of learning can be used.

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Support Vector Machines

Linear Separators

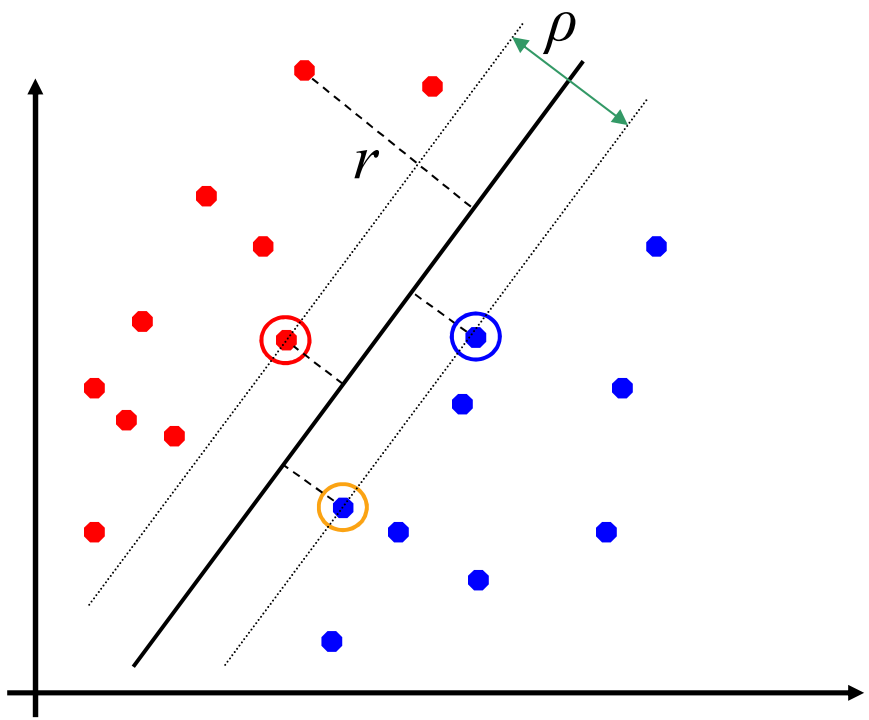
- Which of the linear separators is optimal?



Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Classification Margin

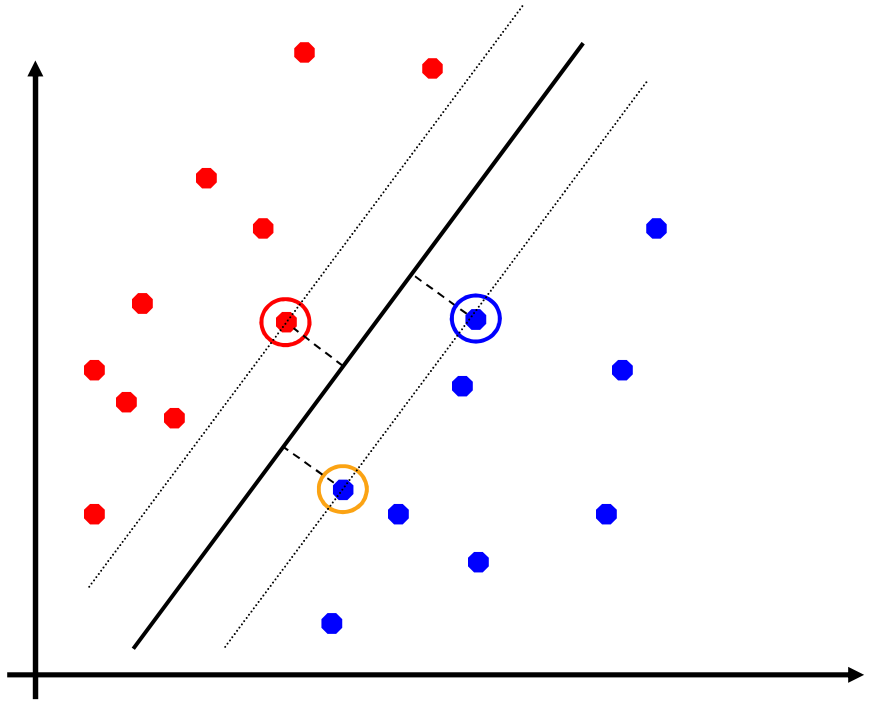
- Distance from example $\mathbf{x}_i$ to the separator is $r = \frac{\mathbf{w}^T \mathbf{x}_i + b}{\|\mathbf{w}\|}$
- Examples closest to the hyperplane are ***support vectors***.
- ***Margin*** ρ of the separator is the distance between support vectors from two classes.



Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Maximum Margin Classification

- Maximizing the margin is good according to intuition.
- Implies that only support vectors matter; other training examples are ignorable.



Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

SVM as 2-class Linear Classifier

(Cortes and Vapnik, 1995; Vapnik, 1995)

$$\mathsf{X} = \left\{\mathbf{x}^t, r^t\right\}_t \text{ where } r^t = \begin{cases} +1 & \text{if } \mathbf{x}^t \in C_1 \\ -1 & \text{if } \mathbf{x}^t \in C_2 \end{cases}$$

find $\mathbf{w}$ and w_0 such that

$$\mathbf{w}^T\mathbf{x}^t + w_0 \geq +1 \text{ for } r^t = +1$$

$$\mathbf{w}^T\mathbf{x}^t + w_0 \leq -1 \quad \text{for } r^t = -1$$

Note the condition >= 1 (not just 0). We can always do this if the classes are linearly separable by rescaling w and w0, without affecting the separating hyperplane: $\mathbf{w}^T\mathbf{x}^t + w_0 = 0$

r^t is 1 for x^t if x belongs to class +1
r^t is -1 for x^t if x belongs to class -1

Optimal separating hyperplane:
Separating hyperplane maximizing the margin

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Optimal Separating Hyperplane

Must satisfy:

$$\mathbf{w}^T\mathbf{x}^t + w_0 \geq +1 \text{ for } r^t = +1$$

$$\mathbf{w}^T\mathbf{x}^t + w_0 \leq -1 \text{ for } r^t = -1$$

which can be rewritten as

$$r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \geq +1$$

(Cortes and Vapnik, 1995; Vapnik, 1995)

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Maximizing the Margin

Distance from the discriminant to the closest instances on either side is called the margin

In general this relationship holds (geometry): $d = \frac{|g(x)|}{\|\mathbf{w}\|}$

So, for the support vectors, we have:

$$d = \begin{cases} \frac{1}{\|\mathbf{w}\|} \\ \\ \frac{|-1|}{\|\mathbf{w}\|} \end{cases}$$

$$\rho = 2d = \frac{2}{\|\mathbf{w}\|}$$

To maximize margin, minimize the Euclidian norm of the weight vector w

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

- The SVM formulation starts as an optimization function subject to some constraints (red box)
- Then the constraints are added into the optimization by adding extra variables (called Lagrange multipliers)

$$\min \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \geq +1, \forall t$$

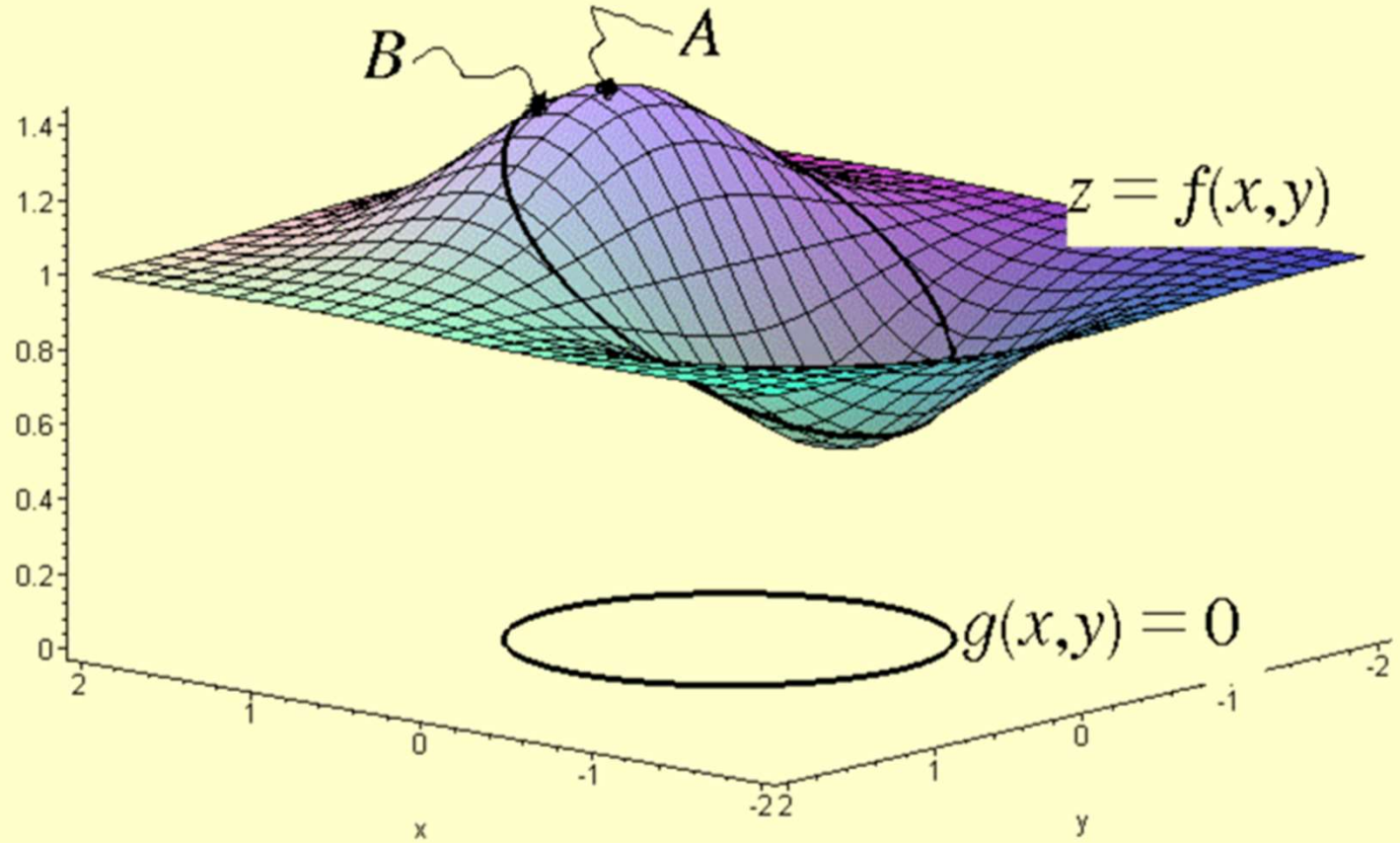
$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N}\alpha^t\left[r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) - 1\right]$$

$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N}\alpha^t r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) + \sum_{t=1}^{N}\alpha^t$$

Unconstrained problem using Lagrange multipliers (+)

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

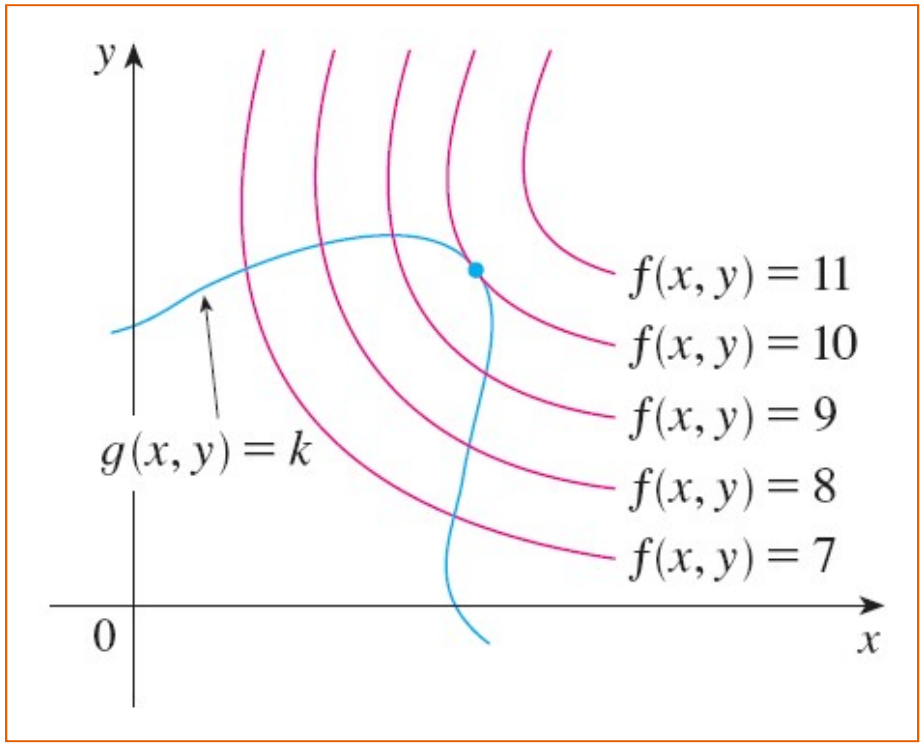
In the figure below we have illustrated an extreme value problem with constraints. The point *A* is the largest value of the function *z=f(x,y)* while the point *B* is the largest value of the function under the *constraint g(x,y)=0*.



Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

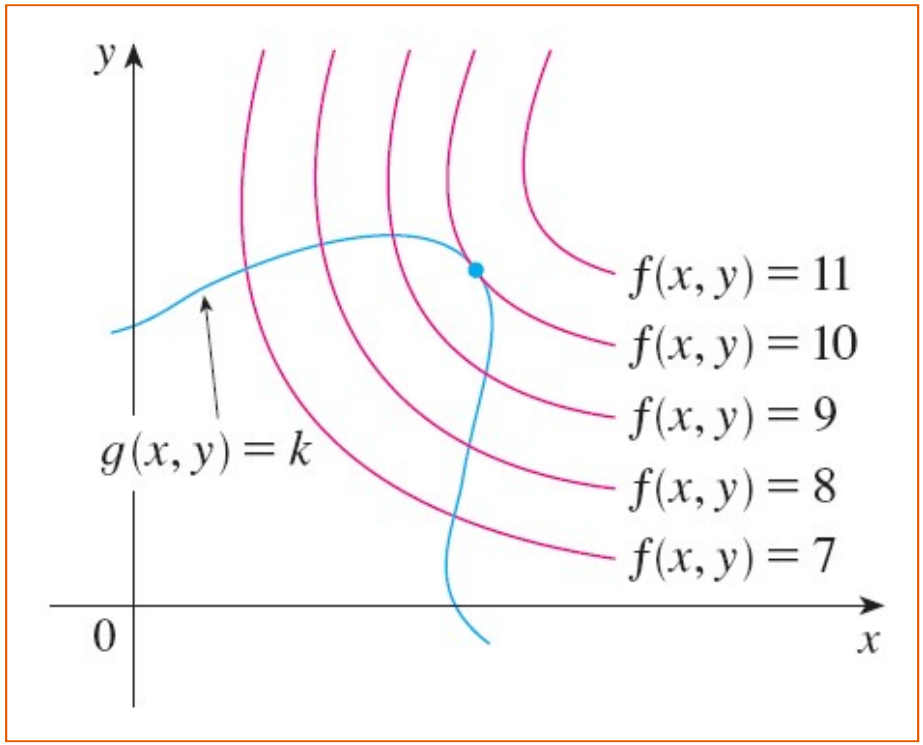
LAGRANGE MULTIPLIERS—TWO VARIABLES

- The figure shows this curve together with several level curves of *f*.
- These have the equations $f(x, y) = c$, where $c = 7, 8, 9, 10, 11$



LAGRANGE MULTIPLIERS—TWO VARIABLES

- To maximize $f(x, y)$ subject to $g(x, y) = k$ is to find:
- The largest value of c such that the level curve $f(x, y) = c$ intersects $g(x, y) = k$.



$$\min \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N} \alpha^t r^t \left(\mathbf{w}^T \mathbf{x}^t + w_0\right) + \sum_{t=1}^{N} \alpha^t$$

At the minimum, the following minimum requirements have to be satisfied:

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{t=1}^{N} \alpha^t r^t \mathbf{x}^t$$

$$\frac{\partial L_p}{\partial w_0} = 0 \Rightarrow \sum_{t=1}^{N} \alpha^t r^t = 0$$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Going from primal to dual form

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N} \alpha^t \left[r^t \left(\mathbf{w}^T \mathbf{x}^t + w_0 \right) - 1 \right]$$

$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N} \alpha^t r^t \left(\mathbf{w}^T \mathbf{x}^t + w_0 \right) + \sum_{t=1}^{N} \alpha^t$$

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{t=1}^{N} \alpha^t r^t \mathbf{x}^t$$

$$\frac{\partial L_p}{\partial w_0} = 0 \Rightarrow \sum_{t=1}^{N} \alpha^t r^t = 0$$

$$L_d = \frac{1}{2}\left(\boldsymbol{w}^T \boldsymbol{w} \right) - \boldsymbol{w}^T \sum_t \alpha^t r^t \boldsymbol{x}^t - w_0 \sum_t \alpha^t r^t + \sum_t \alpha^t$$

$$= -\frac{1}{2}\left(\boldsymbol{w}^T \boldsymbol{w} \right) + \sum_t \alpha^t$$

$$= -\frac{1}{2} \sum_t \sum_s \alpha^t \alpha^s r^t r^s \left(\boldsymbol{x}^t \right)^T \boldsymbol{x}^s + \sum_t \alpha^t$$

$$\text{subject to } \sum_t \alpha^t r^t = 0 \text{ and } \alpha^t \geq 0, \forall t$$

- Maximize L_d with respect to α^t only
- Quadratic programming problem
- Thanks to the convexity of the problem, optimal value of $L_p = L_d$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Calculating the parameters $\mathbf{w}$ and w_0

- Once we solve for α^t, *we see that*
  - *most of them* are 0
  - only a small number have $\alpha^t > 0$ and the corresponding x^ts are called the **support vectors**

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N} \alpha^t \left[r^t \left(\mathbf{w}^T \mathbf{x}^t + w_0 \right) - 1 \right]$$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Calculating the parameters $\mathbf{w}$ and w_0

Once we have the α^t s, we can reconstruct the weight vectors:

$$\mathbf{w} = \sum_{t=1}^{N} \alpha^t r^t \mathbf{x}^t = \sum_{t \in SV} \alpha^t r^t \mathbf{x}^t$$

where *SV* is the set of the Support Vectors.

After we compute **w**, from any x^t (or better averaging over all of them) we can compute w_0:

$$w_0 = r^t - \mathbf{w}^T \mathbf{x}^t$$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

- We make decisions by comparing each new example $\mathbf{x}$ with **only** the support vectors $\{\mathbf{x}_i\}_{i \in SV}$:

$$\hat{y} = \text{sign}\left(\sum_{i \in SV} \hat{\alpha}_i \, y_i \, (\mathbf{x}_i^T \mathbf{x}) + \hat{w}_0 \right)$$

Choose C1 if > 0
C2 otherwise

41

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Not-Linearly Separable Case

- The non-separable case cannot find a feasible solution using the previous approach
  - The objective function (L_D) grows arbitrarily large.
- Relax the constraints, but only when necessary
  - Introduce a further cost for this

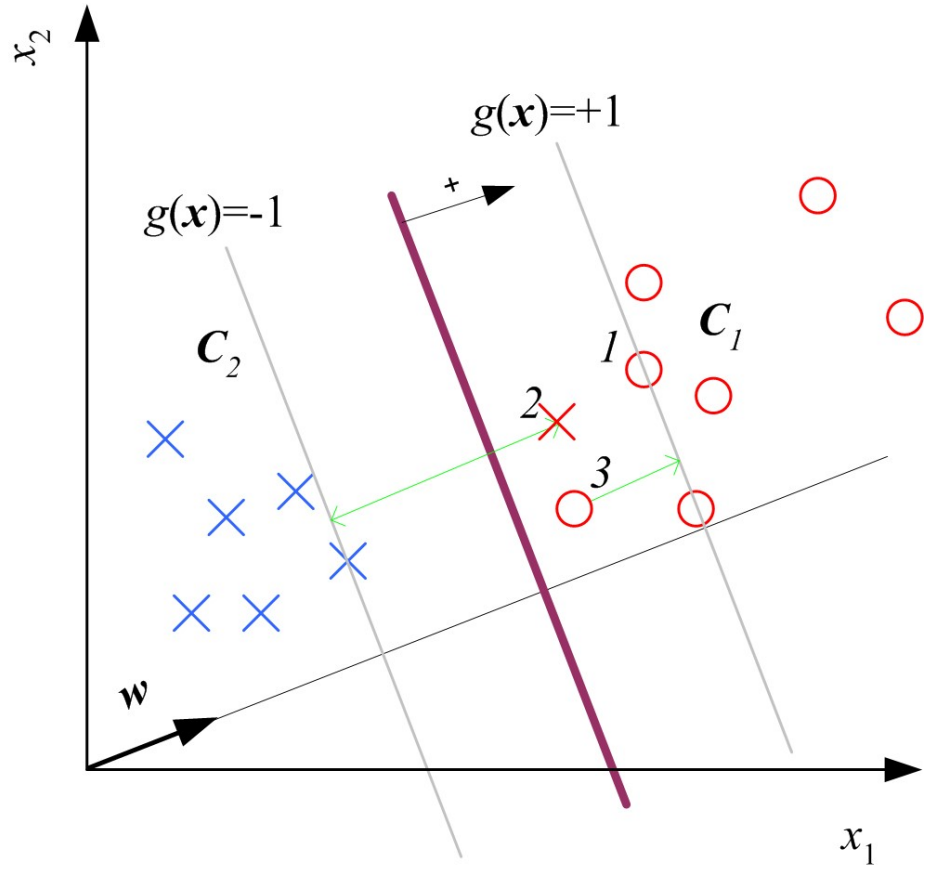
Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Soft Margin Hyperplane

- Not linearly separable

$$r^t\left(\mathbf{w}^T x^t + w_0\right) \geq 1 - \xi^t$$

$$\xi^t \geq 0$$



Three cases (shown in fig):

Case 1: $\xi^t = 0$

Case 2: $\xi^t \geq 1$

Case 3: $0 \leq \xi^t < 1$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Soft Margin Hyperplane

- Define Soft error $\sum_t \xi^t$

Upper bound on the number of training errors

- New primal to minimize is

Lagrange multipliers to enforce positivity of ξ

$$L_p = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_t \xi^t - \sum_t \alpha^t \left[r^t\left(\boldsymbol{w}^T x^t + w_0\right) - 1 + \xi^t\right] - \sum_t \mu^t \xi^t$$

- Parameter *C* can be viewed as a way to control overfitting: it "trades off" the relative importance of maximizing the margin and fitting the training data.
- μ^t are the new Lagrange parameters to guarantee positivity of ξ^t

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Soft Margin Hyperplane

- New dual is the same as the old one

$$L_d = -\frac{1}{2}\sum_t \sum_s \alpha^t \alpha^s r^t r^s \left(\mathbf{x}^t\right)^T \mathbf{x}^s + \sum_t \alpha^t$$

subject to

$$\sum_t \alpha^t r^t = 0 \text{ and } 0 \le \alpha^t \le C, \forall t$$

As in the separable case, instances that are not support vectors vanish with their at=0 and the remaining define the boundary.

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

When we use the soft margin hyperplane

- If C is too large, too high a penalty for non-separable points (too many support vectors)
- If C is too small, we may have underfitting

Kernel Functions in SVM

Kernel Functions

- Instead of trying to fit a non-linear model, we can
  - map the problem to a new space through a non-linear transformation and
  - use a linear model in the new space

- Say we have the new space calculated by the basis functions $\boldsymbol{z} = \boldsymbol{\varphi}(\boldsymbol{x})$ where $z_j = \phi_j(\boldsymbol{x})$, $j=1,...,k$

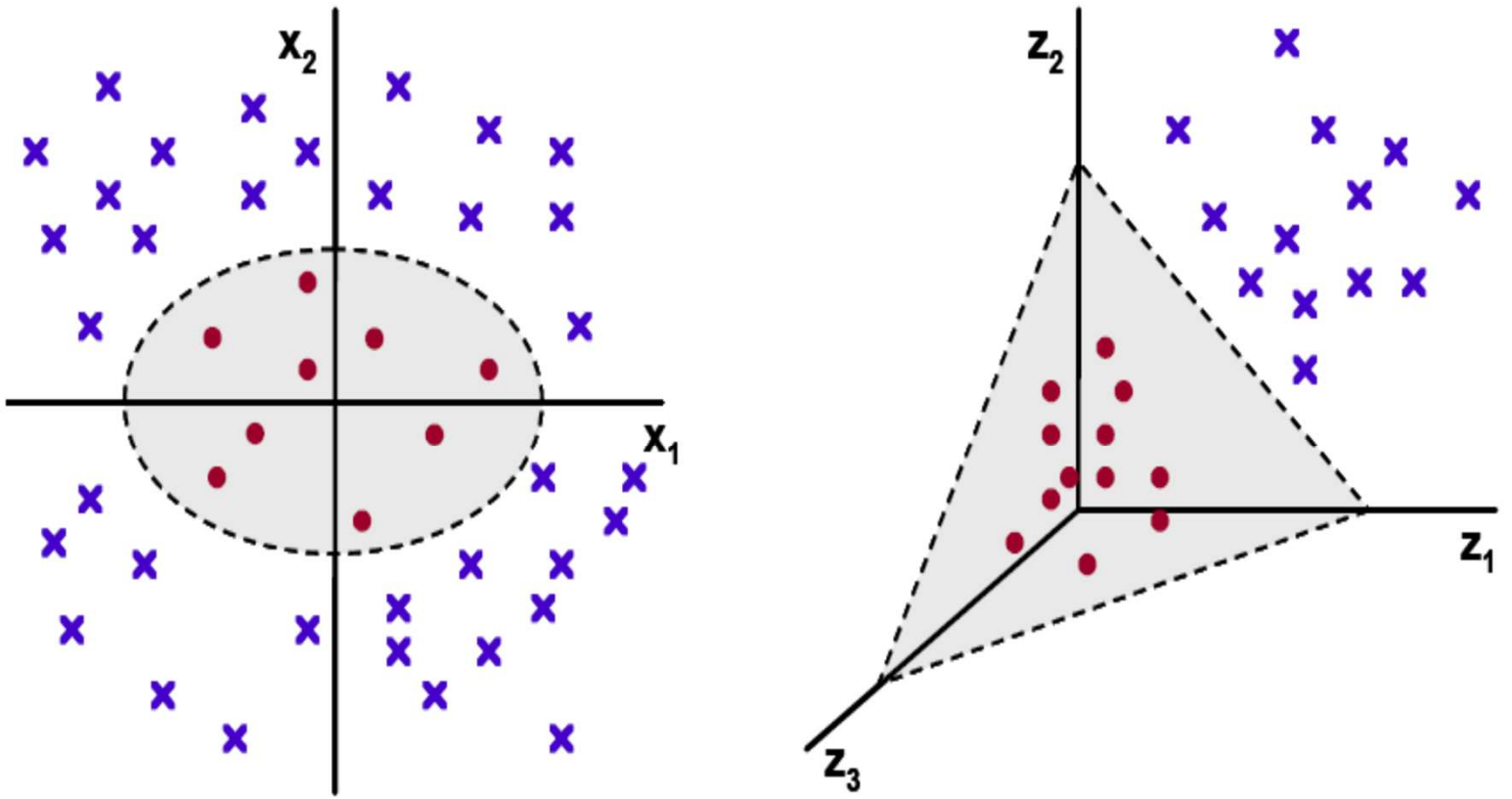
d-dimensional $\boldsymbol{x}$ space $\longrightarrow$ k-dimensional $\boldsymbol{z}$ space

$$\boldsymbol{\varphi}(\boldsymbol{x}) = [\varphi_1(x)\ \varphi_2(x)\ ...\ \varphi_k(x)]$$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

$$\varphi : \mathbb{R}^2 \to \mathbb{R}^3$$

$$(x_1, x_2) \mapsto (z_1, z_2, z_3) = \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right)$$



Kernel Functions

$$g(\mathbf{x}) = \sum_{k=1} w_k \varphi_k(\mathbf{x}) + b$$

$$g(\mathbf{x}) = \sum_{k=0} w_k \varphi_k(\mathbf{x})$$

if we assume $\varphi_0(\mathbf{x}) = 1$ for $\forall \mathbf{x}$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Kernel Machines

- Preprocess input $\boldsymbol{x}$ by basis functions

$$\boldsymbol{z} = \boldsymbol{\varphi}(\boldsymbol{x}) \qquad g(\boldsymbol{z}) = \boldsymbol{w}^T \boldsymbol{z}$$

$$g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\varphi}(\boldsymbol{x})$$

- SVM solution: Find Kernel functions $K(x,y)$ such that the inner product of basis functions are replaced by a Kernel function *in the original input space*

$$\boldsymbol{w} = \sum_t \alpha^t r^t \boldsymbol{z}^t = \sum_t \alpha^t r^t \boldsymbol{\varphi}\left(\boldsymbol{x}^t\right)$$

$$g\left(\boldsymbol{x}\right) = \boldsymbol{w}^T \boldsymbol{\varphi}\left(\boldsymbol{x}\right) = \sum_t \alpha^t r^t \boldsymbol{\varphi}\left(\boldsymbol{x}^t\right)^T \boldsymbol{\varphi}\left(\boldsymbol{x}\right)$$

$$g\left(\boldsymbol{x}\right) = \sum_t \alpha^t r^t K\left(\boldsymbol{x}^t, \boldsymbol{x}\right)$$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Kernel Functions

- Consider polynomials of degree q:

$$K(\mathbf{x},\mathbf{y}) = (\mathbf{x}^T\mathbf{y}+1)^2$$
$$= (x_1y_1 + x_2y_2 + 1)^2$$
$$= 1 + 2x_1y_1 + 2x_2y_2 + 2x_1x_2y_1y_2 + x_1^2y_1^2 + x_2^2y_2^2$$
$$\phi(\mathbf{x}) = \left[1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2\right]^T$$

(Cherkassky and Mulier, 1998)

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

- Let's look at the previous example in a bit more detail

$$\mathbf{x} \rightarrow \phi(\mathbf{x}) = [x_1^2 \;\; x_2^2 \;\; \sqrt{2}x_1x_2 \;\; \sqrt{2}x_1 \;\; \sqrt{2}x_2 \;\; 1]$$

- The SVM classifier deals only with inner products of examples (or feature vectors). In this example,

$$\begin{aligned}\phi(\mathbf{x})^T\phi(\mathbf{x}') &= x_1^2x_1'^2 + x_2^2x_2'^2 + 2x_1x_2x_1'x_2' + 2x_1x_1' + 2x_2x_2' + 1 \\ &= (1 + x_1x_1' + x_2x_2')^2 \\ &= \left(1 + (\mathbf{x}^T\mathbf{x}')\right)^2\end{aligned}$$

so the inner products can be evaluated without ever explicitly constructing the feature vectors $\phi(\mathbf{x})$!

- $K(\mathbf{x}, \mathbf{x}') = \left(1 + (\mathbf{x}^T\mathbf{x}')\right)^2$ is a *kernel function* (inner product in the feature space)

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Examples of Kernel Functions

Linear: $K(x_i,x_j)= x_i^T x_j$

Mapping Φ: $x \rightarrow \varphi(x)$, where $\varphi(x)$ is x itself

Polynomial of power p: $K(x_i,x_j)= (1+ x_i^T x_j)^p$

Mapping Φ: $x \rightarrow \varphi(x)$, where $\varphi(x)$ has $\binom{d+p}{p}$ dimensions

Gaussian (radial-basis function): $K(x_i,x_j) = e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2}}$

Mapping Φ: $x \rightarrow \varphi(x)$, where $\varphi(x)$ is infinite-dimensional: every point is mapped to a function (a Gaussian); combination of functions for support vectors is the separator.

Higher-dimensional space still has intrinsic dimensionality d, but linear separators in it correspond to non-linear separators in original space.

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

$$x = (x_1, x_2);$$

$$z = (z_1, z_2);$$

$$\langle x, z \rangle^2 = (x_1 z_1 + x_2 z_2)^2 =$$

$$= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 z_1 x_2 z_2 =$$

$$= \left\langle (x_1^2, x_2^2, \sqrt{2} x_1 x_2), (z_1^2, z_2^2, \sqrt{2} z_1 z_2) \right\rangle =$$

$$= \langle \phi(x), \phi(z) \rangle$$

www.support-vector.net

Other Kernel Functions

- Polynomials of degree q: $K(\mathbf{x}^t, \mathbf{x}) = (\mathbf{x}^T \mathbf{x}^t)^q$

$$K(\boldsymbol{x}^t, \boldsymbol{x}) = (\boldsymbol{x}^T \boldsymbol{x}^t + 1)^q$$

- Radial-basis functions: $$K(\boldsymbol{x}^t, \boldsymbol{x}) = \exp\left[-\frac{\|\boldsymbol{x}^t - \boldsymbol{x}\|^2}{\sigma^2}\right]$$

- Sigmoidal functions such as: $$K(\boldsymbol{x}^t, \boldsymbol{x}) = \tanh(2\boldsymbol{x}^T \boldsymbol{x}^t + 1)$$

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Sample solutions

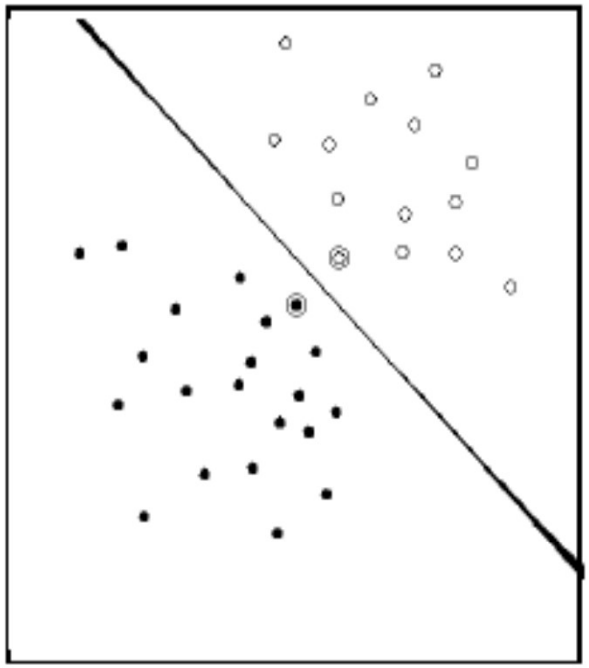


Figure 3: A simple two–class classification problem as solved by the Support Vector algorithm ($c_i = 1$ for all i; cf. Eq. 1). Note that the RBF centers (indicated by extra circles) are closest to the decision boundary.

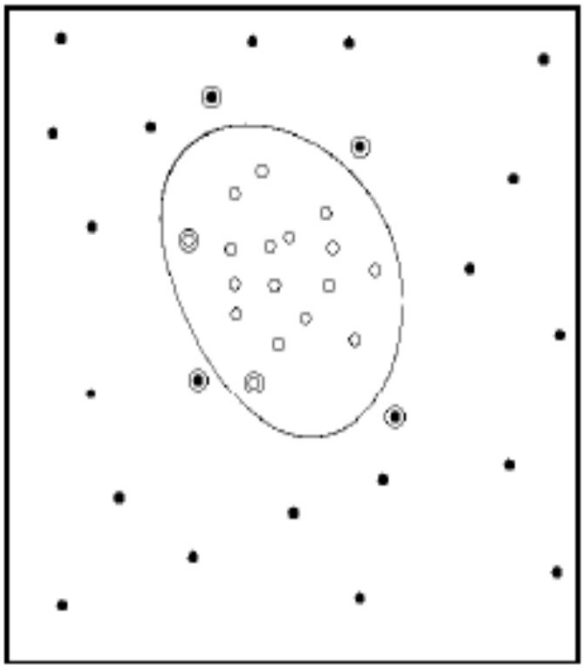


Figure 4: Two–class classification problem solved by the Support Vector algorithm ($c_i = 1$ for all i; cf. Eq. 1).

- Informally, kernel methods *implicitly* define the class of possible patterns by introducing a notion of similarity between data
  - Choice of similarity -> Choice of relevant features

- More formally, kernel methods exploit information about the inner products between data items
  - Many standard algorithms can be rewritten so that they only require inner products between data (inputs)
  - Kernel functions = inner products in some feature space (potentially very complex)
  - If kernel given, no need to specify what features of the data are being used
  - Kernel functions make it possible to use infinite dimensions
    - efficiently in time / space

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

SVM summary

- SVMs were originally proposed by Boser, Guyon and Vapnik in 1992 and gained increasing popularity in late 1990s.
- SVMs represent a general methodology for many PR problems: classification,regression, feature extraction, clustering, novelty detection, etc.
- SVMs can be applied to complex data types beyond feature vectors (e.g. graphs, sequences, relational data) by designing kernel functions for such data.
- SVM techniques have been extended to a number of tasks such as regression [Vapnik *et al.* '97], principal component analysis [Schölkopf *et al.* '99], etc.
- Most popular optimization algorithms for SVMs use *decomposition* to hill-climb over a subset of α_i's at a time, e.g. SMO [Platt '99] and [Joachims '99]

Advantages of SVMs

- There are no problems with local minima, because the solution is a Qaudratic Programming problem
- The optimal solution can be found in polynomial time
- There are few model parameters to select: the penalty term C, the kernel function and parameters (e.g., spread σ in the case of RBF kernels)
- The final results are stable and repeatable (e.g., no random initial weights)
- The SVM solution is sparse; it only involves the support vectors
- SVMs rely on elegant and principled learning methods
- SVMs provide a method to control complexity independently of dimensionality
- SVMs have been shown (theoretically and empirically) to have excellent generalization capabilities

Challenges

- **Can the kernel functions be selected in a principled manner?**
- **SVMs still require selection of a few parameters, typically through cross-validation**
- **How does one incorporate domain knowledge?**
  - Currently this is performed through the selection of the kernel and the introduction of “artificial” examples
- How interpretable are the results provided by an SVM?
- What is the optimal data representation for SVM? What is the effect of feature weighting? How does an SVM handle categorical or missing features?
- Do SVMs always perform best? Can they beat a hand-crafted solution for a particular problem?
- Do SVMs eliminate the model selection problem?

- More explanations or demonstrations can be found at:
  - http://www.support-vector-machines.org/index.html
  - Haykin Chp. 6 pp. 318-339
  - Burges tutorial (under/reading/)
    - Burges, CJC "A Tutorial on Support Vector Machines for Pattern Recognition" *Data Mining and Knowledge Discovery*, Vol 2 No 2, 1998.
  - http://www.dtreg.com/svm.htm

- **Software**
  - **SVM*light***, by Joachims, is one of the most widely used SVM classification and regression package. Distributed as C++ source and binaries for Linux, Windows, Cygwin, and Solaris. Kernels: polynomial, radial basis function, and neural (tanh).

  - **LibSVM http://www.csie.ntu.edu.tw/~cjlin/libsvm/** LIBSVM (Library for Support Vector Machines), is developed by Chang and Lin; also widely used. Developed in C++ and Java, it supports also multi-class classification, weighted SVM for unbalanced data, cross-validation and automatic model selection. It has interfaces for Python, R, Splus, MATLAB, Perl, Ruby, and LabVIEW. Kernels: linear, polynomial, radial basis function, and neural (tanh).

- Applet to play with:
  - http://lcn.epfl.ch/tutorial/english/svm/html/index.html

- Some non-SVM good applets:
  - http://cs.stanford.edu/people/karpathy/convnetjs/

SVM Applications

- Cortes and Vapnik 1995:
  - Handwritten digit classification
  - 16x16 bitmaps -> 256 dimensions
  - Polynomial kernel where q=3 -> feature space with 10^6 dimensions
  - No overfitting on a training set of 7300 instances
  - Average of 148 support vectors over different training sets

Expected test error rate:

$Exp_N[P(error)] = Exp_N[\#support\ vectors] / N$

(= 0.02 for the above example)

Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

- The main things to know are:
  - The aim of SVMs (maximizing the margins)
  - How it is specified as an optimization problem
  - Once solved, the optimization finds the αs, that in turn specifies the weights
    - αs also indicate which input data are the support vectors (those for which the corresponding α is non-zero)
  - The non-linearly separable case can be handled in a similar manner, with a slight extension that involves a parameter C
    - It is important to understand what C does in order to properly use SVM software
  - The use of kernels makes it possible to go to higher dimensional space, without actually computing those features explicitly; nor being badly affected by the curse of dimensionality