

# Support Vector Machines

[srihari@buffalo.edu](mailto:srihari@buffalo.edu)

# SVM Discussion Overview

1. Importance of SVMs
2. Overview of Mathematical Techniques Employed
3. Margin Geometry
4. SVM Training Methodology
5. Overlapping Distributions
6. Dealing with Multiple Classes
7. SVM and Computational Learning Theory
8. Relevance Vector Machines

# 1. Importance of SVMs

- SVM is a discriminative method that brings together:
  - 1.computational learning theory
  - 2.previously known methods in linear discriminant functions
  - 3.optimization theory
- Widely used for solving problems in classification, regression and novelty detection

# Alternative Names for SVM

- Also called Sparse kernel machines
  - Kernel methods predict based on linear combinations of a kernel function evaluated at the training points, e.g., Parzen Window
  - Sparse because not all pairs of training points need be used
- Also called Maximum margin classifiers

## 2. Mathematical Techniques Used

### 1. Linearly separable case considered

since appropriate nonlinear mapping  $f$  to a high dimension two categories are always separable by a hyperplane

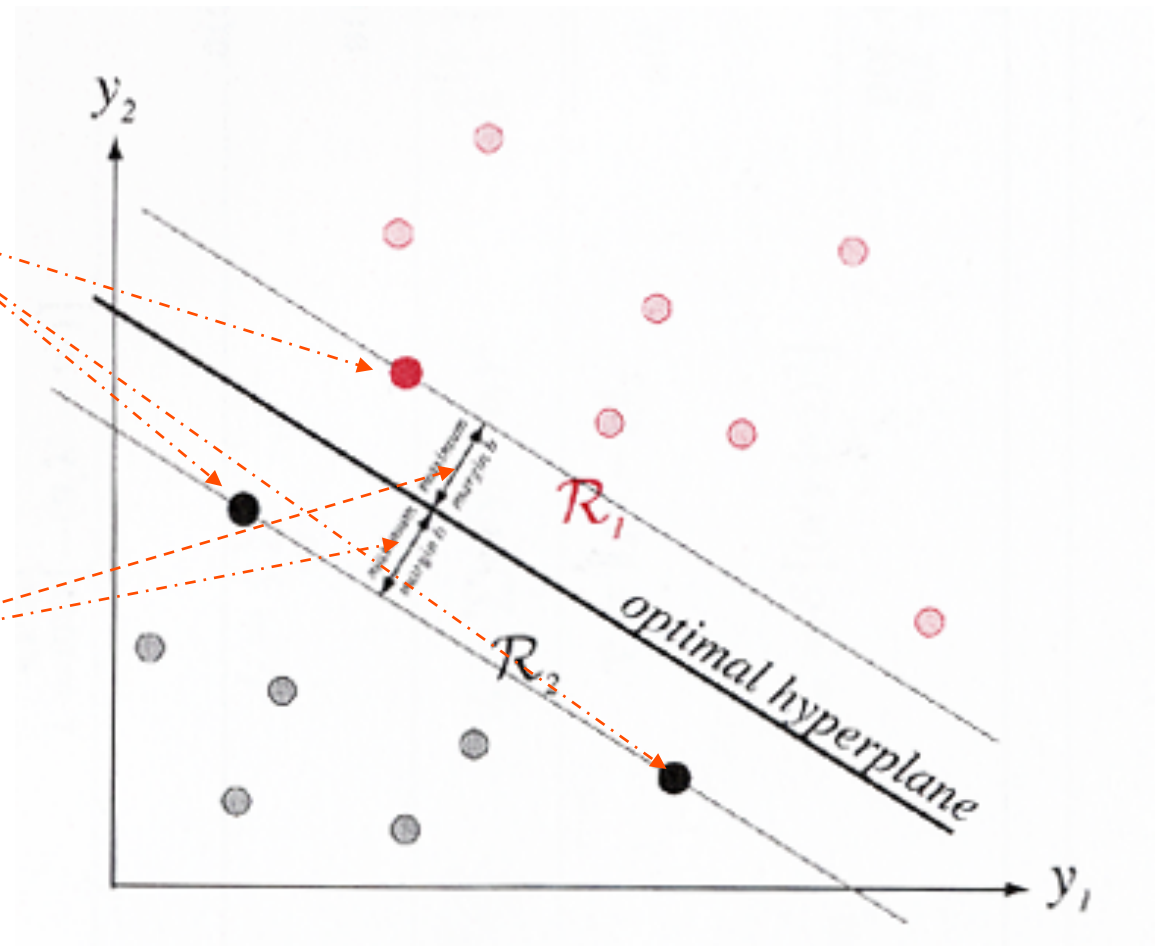
### 2. To handle non-linear separability

- Preprocessing data to represent in much higher-dimensional space than original feature space
- Kernel trick reduces computational overhead

$$k(\mathbf{y}_j, \mathbf{y}_k) = \mathbf{y}_j^t \cdot \mathbf{y}_k = \phi(\mathbf{x}_j)^t \cdot \phi(\mathbf{x}_k)$$

# 3. Support Vectors and Margin

- Support vectors are those nearest patterns at distance  $b$  from hyperplane
- SVM finds hyperplane with maximum distance  
(margin distance  $b$ )  
from nearest training patterns



Three support vectors are shown as solid dots

# Margin Maximization

- Motivation: computational learning theory
  - or statistical learning theory (PAC learning-VC dimension)
- Insight (Tong, Koller 2000):
  - Model distributions for each class using Parzen density estimators using Gaussian kernels with common parameter  $\sigma^2$
  - Instead of optimum boundary, determine best hyperplane relative to learned density model
    - As  $\sigma^2 \rightarrow 0$  optimum hyperplane has maximum margin
  - Hyperplane becomes independent of data points that are not support vectors

# Distance from arbitrary plane

- Hyperplane:**  $g(x) = w^t \bullet x + w_0$  where  $w$  is the weight vector and  $w_0$  is bias

- Lemma:** Distance from  $x$  to the plane is

$$r = \frac{g(x)}{\|w\|}$$

Proof: Let  $x = x_p + r \frac{w}{\|w\|}$

where  $r$  is the distance from  $x$  to the plane,

$$= 0$$

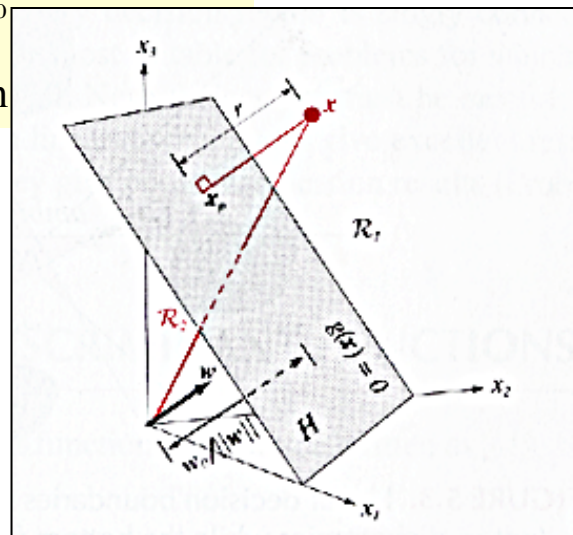
$$g(x) = w^t \left( x_p + r \frac{w}{\|w\|} \right) + w_0 = w^t x_p + w_0 + r \frac{w^t w}{\|w\|} = g(x_p) + r \frac{\|w\|^2}{\|w\|} = r \|w\| \quad \text{QED}$$

**Corollary:** Distance of origin to plane is

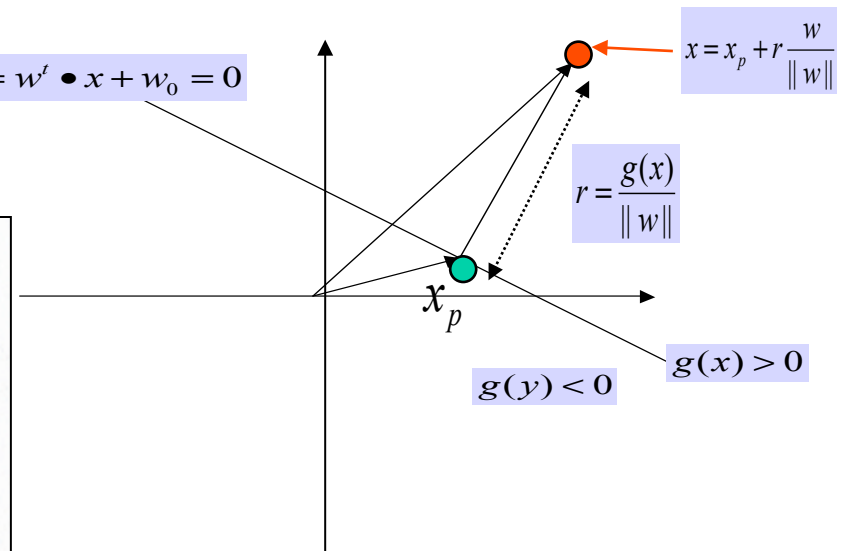
$$r = g(0)/\|w\| = w_0/\|w\|$$

since  $g(0) = w^t 0 + w_0 = w_0$

Thus  $w_0=0$  implies that plane passes through origin



$$g(x) = w^t \bullet x + w_0 = 0$$





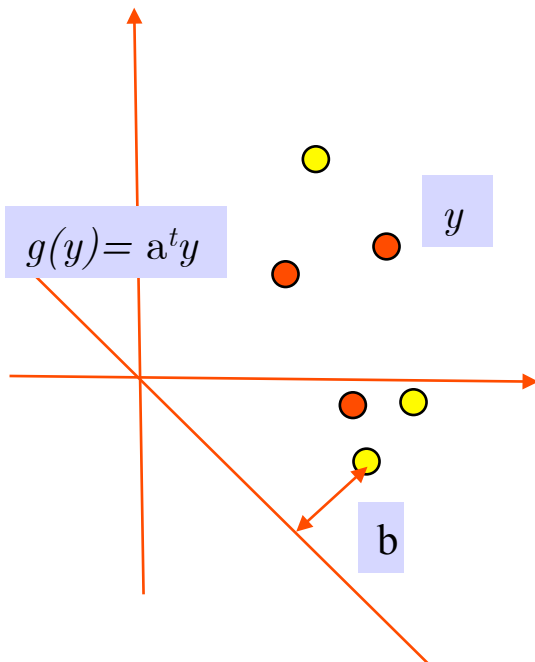
# Choosing a margin

- Augmented space:  $g(y) = a^t y$  by choosing  $a_0 = w_0$  and  $y_0 = 1$ , i.e, plane passes through origin
- For each of the patterns, let  $z_k = \pm 1$  depending on whether pattern  $k$  is in class  $\omega_1$  or  $\omega_2$
- Thus if  $g(y)=0$  is a separating hyperplane then
 
$$z_k g(y_k) \geq 0, \quad k = 1, \dots, n$$
- Since distance of a point  $y$  to hyperplane  $g(y)=0$  is

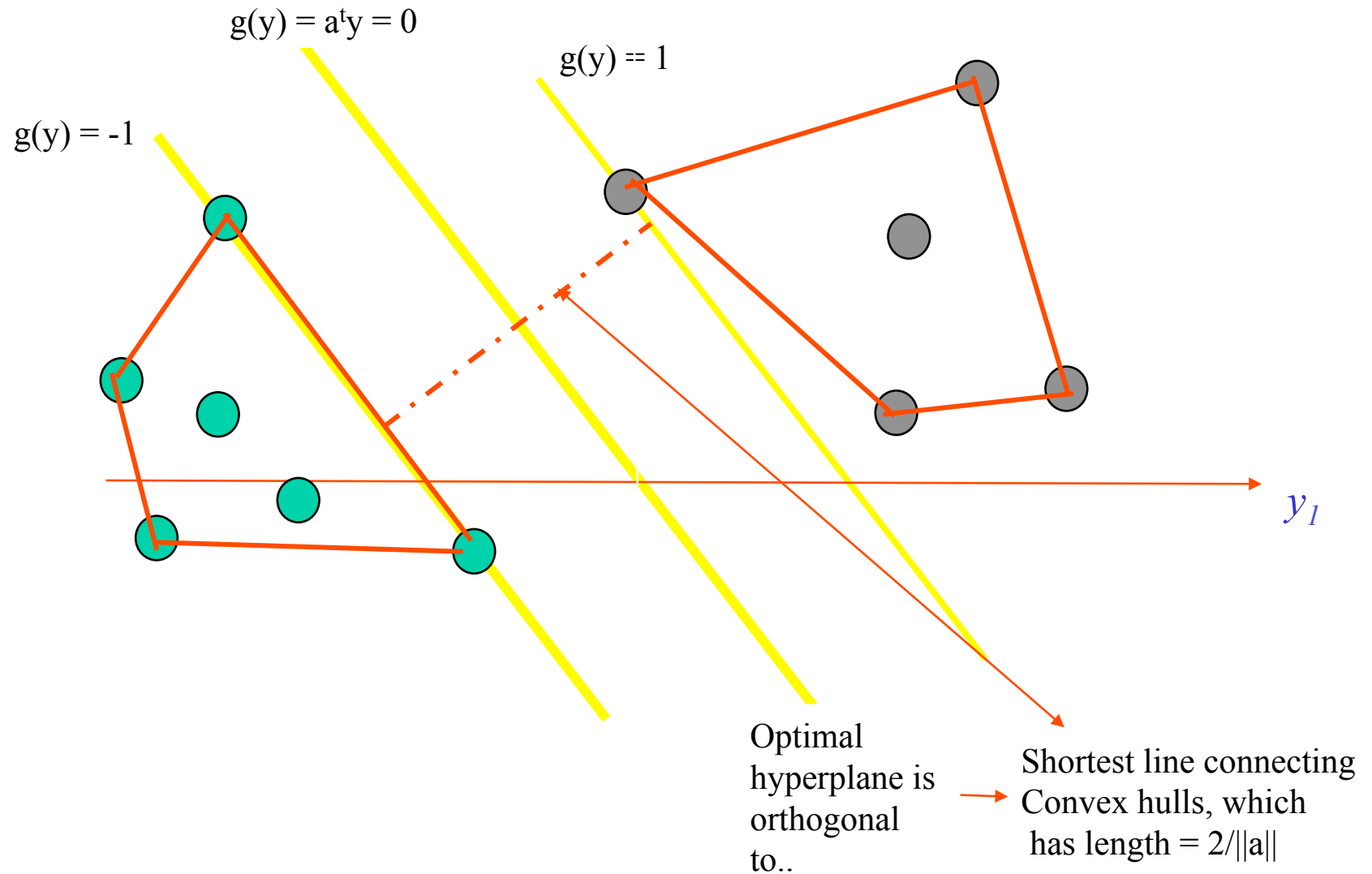
$$\frac{g(y)}{\|a\|}$$

we could require that hyperplane be such that all points are at least distant  $b$  from it, i.e.,

$$\frac{z_k g(y_k)}{\|a\|} \geq b$$



# SVM Margin geometry



# Statement of Optimization Problem

- The goal is to find the weight vector  $\mathbf{a}$  that satisfies

$$\frac{z_k g(y_k)}{\|\mathbf{a}\|} \geq b, \quad k = 1, \dots, n$$

while maximizing  $b$

- To ensure uniqueness we impose the constraint

$$b \|\mathbf{a}\| = 1 \quad \text{or} \quad b = 1/\|\mathbf{a}\|$$

- Which implies that  $\|\mathbf{a}\|^2$  is to be minimized
- Support vectors are (transformed) training patterns which represent equality in above equation
  - Called a quadratic optimization problem since we are trying to minimize a quadratic function subject to a set of linear inequality constraints

# 4. SVM Training Methodology

1. Training is formulated as an optimization problem
  - Dual problem is stated to reduce computational complexity
  - Kernel trick is used to reduce computation
2. Determination of the model parameters corresponds to a **convex optimization** problem
  - Solution is straightforward (local solution is a global optimum)
3. Makes use of **Lagrange** multipliers

# Joseph-Louis Lagrange 1736-1813



- French Mathematician
- Born in Turin, Italy
- Succeeded Euler at Berlin academy
- Narrowly escaped execution in French revolution due to Lavoisier who himself was guillotined
- Made key contributions to calculus and dynamics

# SVM Training: Optimization Problem

*optimize*

$$\arg \min_{\mathbf{a}, b} \frac{1}{2} \|\mathbf{a}\|^2$$

*subject to constraints*

$$z_k \mathbf{a}^t \mathbf{y}_k \geq 1, \quad k = 1, \dots, n$$

- Can be cast as an unconstrained problem by introducing Lagrange undetermined multipliers with one multiplier  $\alpha_k$  for each constraint

- The Lagrange function is 
$$\mathbf{L}(\mathbf{a}, \alpha) = \frac{1}{2} \|\mathbf{a}\|^2 - \sum_{k=1}^n \alpha_k [z_k \mathbf{a}^t \mathbf{y}_k - 1]$$

# Optimization of Lagrange function

- The Lagrange function is

$$\mathbf{L}(\mathbf{a}, \alpha) = \frac{1}{2} \|\mathbf{a}\|^2 - \sum_{k=1}^n \alpha_k [z_k \mathbf{a}^t \mathbf{y}_k - 1]$$

- We seek to minimize  $L ( )$ 
  - with respect to the weight vector  $\mathbf{a}$  and  $\alpha_k \geq 0$
  - maximize it w.r.t . the undetermined multipliers
- Last term represents the goal of classifying the points correctly
- Karush-Kuhn-Tucker construction shows that this can be recast as a maximization problem which is computationally better

# Dual Optimization Problem

- Problem is reformulated as one of maximizing

$$L(\alpha) = \sum_{k=1}^n \alpha_k - \frac{1}{2} \sum_{k=1}^n \sum_{j=1}^n \alpha_k \alpha_j z_k z_j k(y_j, y_k)$$

- Subject to the constraints

$$\sum_{k=1}^n \alpha_k z_k = 0 \quad \alpha_k \geq 0, \quad k = 1, \dots, n$$

given the training data

- where the kernel function is defined by

$$k(y_j, y_k) = y_j^t \cdot y_k = \phi(x_j)^t \cdot \phi(x_k)$$



# Solution of Dual Problem

- Implementation:
  - Solved using quadratic programming
  - Alternatively, since it only needs inner products of training data
    - It can be implemented using kernel functions
    - which is a crucial property for generalizing to non-linear case
- The solution is given by  $\mathbf{a} = \sum_k \alpha_k \mathbf{z}_k y_k$

# Summary of SVM Optimization Problems

## Dual Optimization Problem

$$\begin{aligned} \text{Primal OP: minimize } P(\vec{w}, b, \vec{\xi}) &= \frac{1}{2} \vec{w} \cdot \vec{w} + C \sum_{i=1}^n \xi_i \\ \text{s. t. } y_i [\vec{w} \cdot \vec{x}_i + b] &\geq 1 - \xi_i \text{ and } \xi_i \geq 0 \end{aligned}$$

**Lemma:** The solution  $w^\circ$  can always be written as a linear combination

$$\vec{w}^\circ = \sum_{i=1}^n \alpha_i y_i \vec{x}_i \quad \alpha_i \geq 0$$

of the training data.

$$\begin{aligned} \text{Dual OP: maximize } D(\vec{\alpha}) &= \left( \sum_{i=1}^n \alpha_i \right) - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \\ \text{s.t. } \sum_{i=1}^n \alpha_i y_i &= 0 \quad \text{and} \quad 0 \leq \alpha_i \leq C \end{aligned}$$

$\implies$  positive semi-definite quadratic program

Different  
Notation here!

Quadratic term

# Kernel Function: key property

- If kernel function is chosen with property

$$K(\mathbf{x}, \mathbf{y}) = (f(\mathbf{x}) \cdot f(\mathbf{y}))$$

then computational expense of increased dimensionality is avoided.

- Polynomial kernel  $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^d$   
can be shown (next slide) to  
correspond to a map  $f$  into  
the space spanned by **all** products of exactly  $d$  dimensions.

# A Polynomial Kernel Function

Suppose  $\mathbf{x} = (x_1, x_2)$  is the input vector

The feature space mapping is :

$$\varphi(\mathbf{x}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

Then inner product is

$$\varphi(\mathbf{x})\varphi(\mathbf{y}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) (y_1^2, \sqrt{2}y_1y_2, y_2^2) = (x_1y_1 + x_2y_2)^2$$

Polynomial kernel function to compute the same value is

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^2 = ((x_1, x_2)(y_1, y_2)^t)^2 = (x_1y_1 + x_2y_2)^2$$

or  $K(\mathbf{x}, \mathbf{y}) = \varphi(\mathbf{x})\varphi(\mathbf{y})$

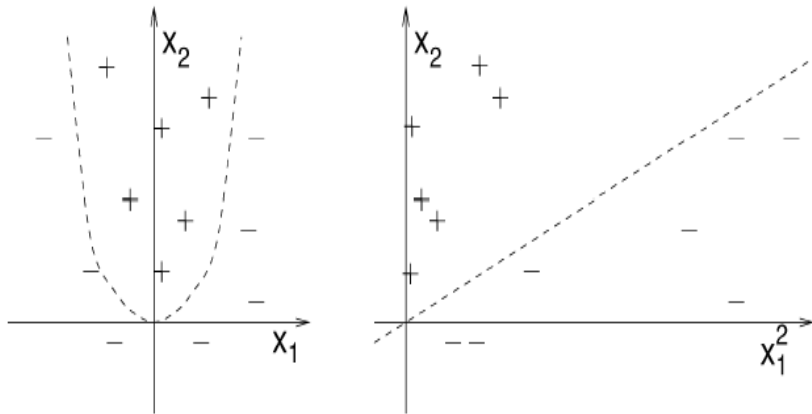
- Inner product  $\varphi(\mathbf{x})\varphi(\mathbf{y})$  needs computing six feature values and  $3 \times 3 = 9$  multiplications
- Kernel function  $K(\mathbf{x}, \mathbf{y})$  has 2 mults and a squaring

# Another Polynomial (quadratic) kernel function

## Example

**Input Space:**  $\mathbf{x} = (x_1, x_2)$  (2 Attributes)

**Feature Space:**  $\Phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, 1)$  (6 Attributes)



- $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^2$
- This one maps  $d = 2, p = 2$  into a *six-* dimensional space
- Contains all the powers of  $\mathbf{x}$

$$\mathbf{K}(\mathbf{x}, \mathbf{y}) = \varphi(\mathbf{x})\varphi(\mathbf{y})$$

where

$$\varphi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, 1)$$

- Inner product needs 36 multiplications
- Kernel function needs 4 multiplications

## SVM with Kernels

**Training:** maximize  $D(\vec{\alpha}) = \left( \sum_{i=1}^n \alpha_i \right) - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(\vec{x}_i, \vec{x}_j)$

s. t.  $\sum_{i=1}^n \alpha_i y_i = 0 \quad \text{und} \quad 0 \leq \alpha_i \leq C$

**Classification:** For new example  $x$   $h(\vec{x}) = \text{sign} \left( \sum_{x_i \in SV} \alpha_i y_i K(\vec{x}_i, \vec{x}) + b \right)$

**New hypotheses spaces through new Kernels:**

Linear:  $K(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$

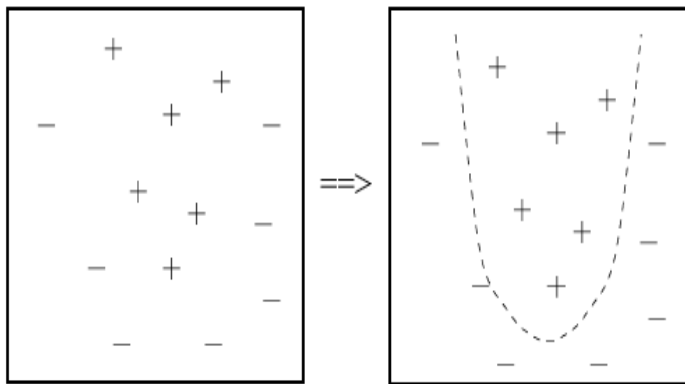
Polynomial:  $K(\vec{x}_i, \vec{x}_j) = [\vec{x}_i \cdot \vec{x}_j + 1]^d$

Radial Basis Functions:  $K(\vec{x}_i, \vec{x}_j) = \exp(-|\vec{x}_i - \vec{x}_j|^2 / \sigma^2)$

Sigmoid:  $K(\vec{x}_i, \vec{x}_j) = \tanh(\gamma(\vec{x}_i - \vec{x}_j) + c)$

# Non-Linear Case

## Non-Linear Problems



### Problem:

- some tasks have non-linear structure
- no hyperplane is sufficiently accurate

How can SVMs learn non-linear classification rules?

- Mapping function  $f(.)$  to a sufficiently high dimension
- So that data from two categories can always be separated by a hyperplane
- Assume each pattern  $x_k$  has been transformed to  $y_k = f(x_k)$ , for  $k=1, \dots, n$
- First choose the non-linear  $f$  functions
  - To map the input vector to a higher dimensional feature space
- Dimensionality of space can be arbitrarily high only limited by computational resources

# Mapping into Higher Dimensional Feature Space

- Mapping each input point  $\mathbf{x}$  by map

$$\mathbf{y} = \Phi(\mathbf{x}) = \begin{pmatrix} 1 \\ \mathbf{x} \\ \mathbf{x}^2 \end{pmatrix}$$

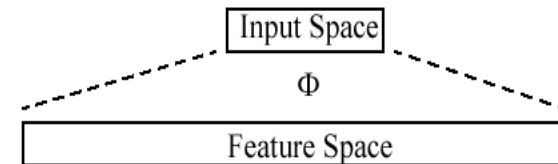
Points on 1-d line are mapped onto curve in 3-d.

- Linear separation in 3-d space is possible. Linear discriminant function in 3-d is in the form

$$g(\mathbf{x}) = a_1 y_1 + a_2 y_2 + a_3 y_3$$

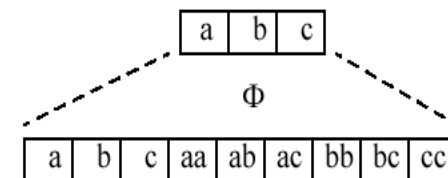
## Extending the Hypothesis Space

**Idea:**



==> Find hyperplane in feature space!

**Example:**



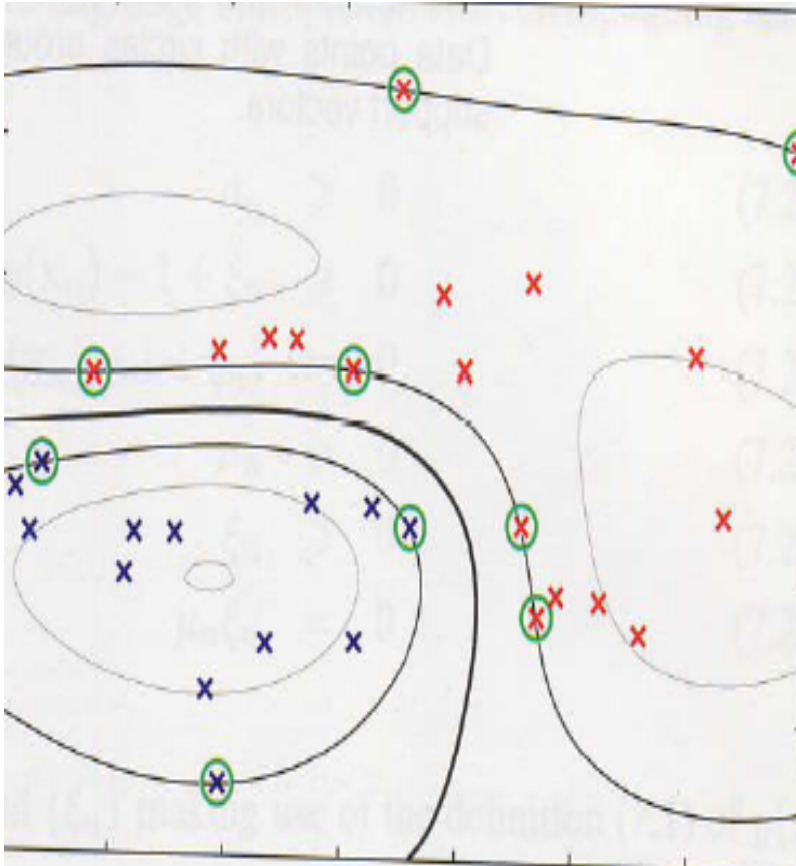
==> The separating hyperplane in features space is a degree two polynomial in input space.



# Pattern Transformation using Kernels

- Problem with high-dimensional mapping
  - Very many parameters
  - Polynomial of degree  $p$  over  $d$  variables leads to  $O(d^p)$  variables in feature space
    - Example: if  $d = 50$  and  $p = 2$  we need a feature space of size 2500
- Solution:
  - Dual Optimization problem needs only inner products
  - Each pattern  $x_k$  transformed into pattern  $y_k$  where
$$\mathbf{y}_k = \Phi(\mathbf{x}_k)$$
  - Dimensionality of mapped space can be arbitrarily high

# Example of SVM results

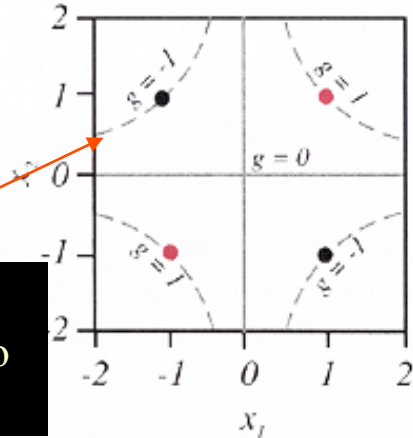


- Two classes in two dimensions
- Synthetic Data
- Shows contours of constant  $g(x)$
- Obtained from SVM with Gaussian kernel function
- Decision boundary is shown
- Margin boundaries are shown
- Support vectors are shown
- Shows sparsity of SVM

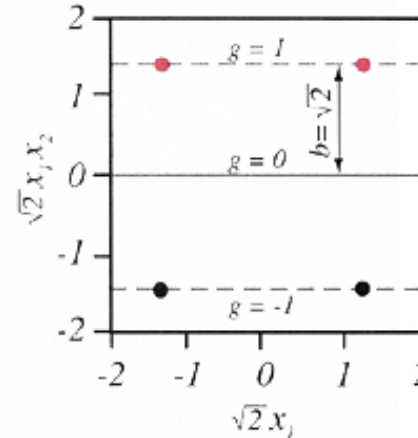
# SVM for the XOR problem

- XOR: binary valued features  $x_1, x_2$
- not solved by linear discriminant
- function  $f$  maps input  $x = [x_1, x_2]$  into six-dimensional feature space  $[x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2]$

Hyperbolas  
Corresponding to  
 $x_1 x_2 = \pm 1$



feature sub-space



Hyperplanes  
Corresponding to  
 $\sqrt{2}x_1 x_2 = \pm 1$

# SVM for XOR: maximization problem

- We seek to maximize

$$\sum_{k=1}^4 \alpha_k - \frac{1}{2} \sum_{k=1}^4 \sum_{j=1}^4 \alpha_k \alpha_j z_j z_k \mathbf{y}_j^t \mathbf{y}_k$$

- Subject to the constraints

$$\begin{aligned} \alpha_1 - \alpha_2 + \alpha_3 - \alpha_4 &= 0 \\ 0 \leq \alpha_k \quad k &= 1, 2, 3, 4, \end{aligned}$$

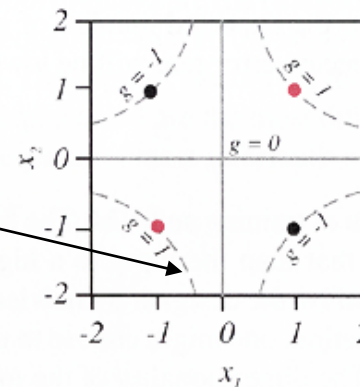
- From problem symmetry, at the solution

$$\alpha_1 = \alpha_3, \text{ and } \alpha_2 = \alpha_4$$

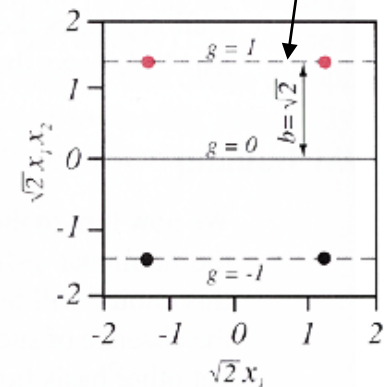
# SVM for XOR: maximization problem

- Can use iterative gradient descent
- Or use analytical techniques for small problem
- The solution is
 
$$\mathbf{a}^* = (1/8, 1/8, 1/8, 1/8)$$
- Last term of Optimizn Problem implies that all four points are support vectors (unusual and due to symmetric nature of XOR)
- The final discriminant function is  $g(x_1, x_2) = x_1 \cdot x_2$
- Decision hyperplane is defined by  $g(x_1, x_2) = 0$
- Margin is given by  $b = 1/\|\mathbf{a}\| = \sqrt{2}$

Hyperbolas  
Corresponding to  
 $x_1 x_2 = \pm 1$



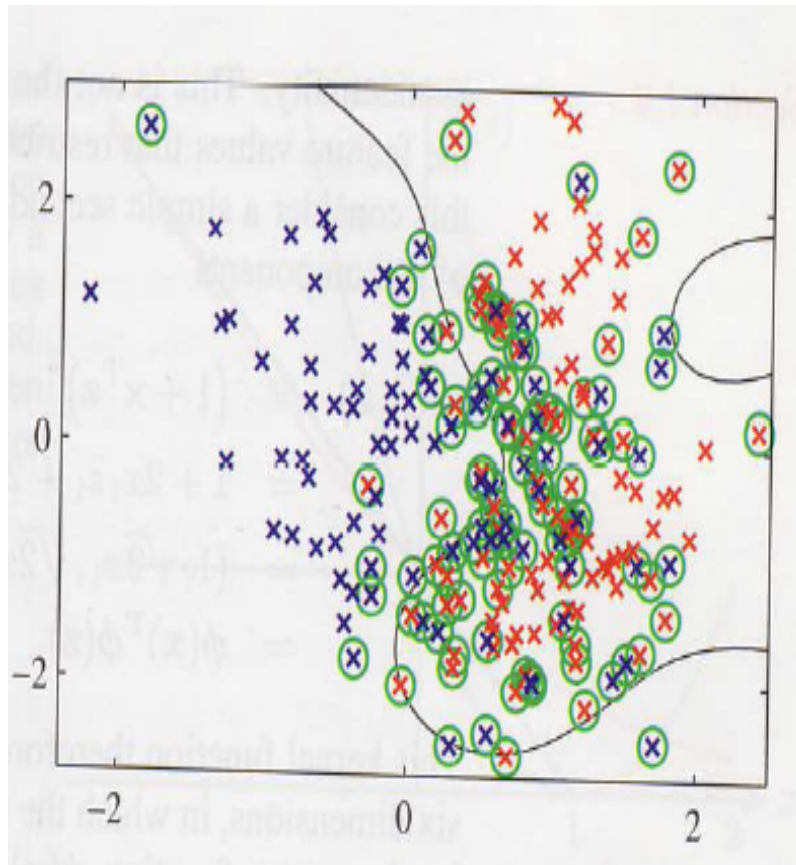
Hyperplanes  
Corresponding to  
 $\sqrt{2}x_1 x_2 = \pm 1$



## 5. Overlapping Class Distributions

- We assumed training data are linearly separable in the mapped space  $y$ 
  - Resulting SVM gives exact separation in input space  $x$  although decision boundary will be nonlinear
- In practice class-conditionals overlap
  - Exact separation leads to poor generalization
  - Therefore need to allow SVM to misclassify some training points

# n-SVM applied to non-separable data



- Support Vectors are indicated by circles
- Done by introducing slack variables
- With one slack variable per training data point
- Maximize the margin while softly penalizing points that lie on the wrong side of the margin boundary
- $\eta$  is an upper-bound on the fraction of margin errors (lie on wrong side of margin boundary)

## 6. Multiclass SVMs (one-versus rest)

- SVM is a two-class classifier
- Several suggested methods for combining multiple two-class classifiers
- Most used approach: *one versus rest*
  - Also recommended by Vapnik
  - using data from class  $C_k$  as the positive examples and data from the remaining  $k-1$  classes as negative examples
  - Disadvantages
    - Input assigned to multiple classes simultaneously
    - Training sets are imbalanced (90% are one class and 10% are another)– symmetry of original problem is lost



## Multiclass SVMs (one-versus one)

- Train  $k(k-1)/2$  different 2-class SVMs on all possible pairs of classes
- Classify test points according to which class has highest number of votes
- Again leads to ambiguities in classification
- For large  $k$  requires significantly more training time than one-versus rest
  - Also more computation time for evaluation
    - Can be alleviated by organizing into a directed acyclic graph (DAGSVM)

## 7. SVM and Computational Learning Theory

- SVM motivated using COLT
  - Called PAC learning framework
- Goal of PAC framework is to understand how large a data sets needs to be in order to give good generalizations
  - Key quantity in PAC learning is VC dimension which provides a measure of complexity of a space of functions
- Maximizing margin is main conclusion

# All dichotomies of 3 points in 2 dimensions are linearly separable

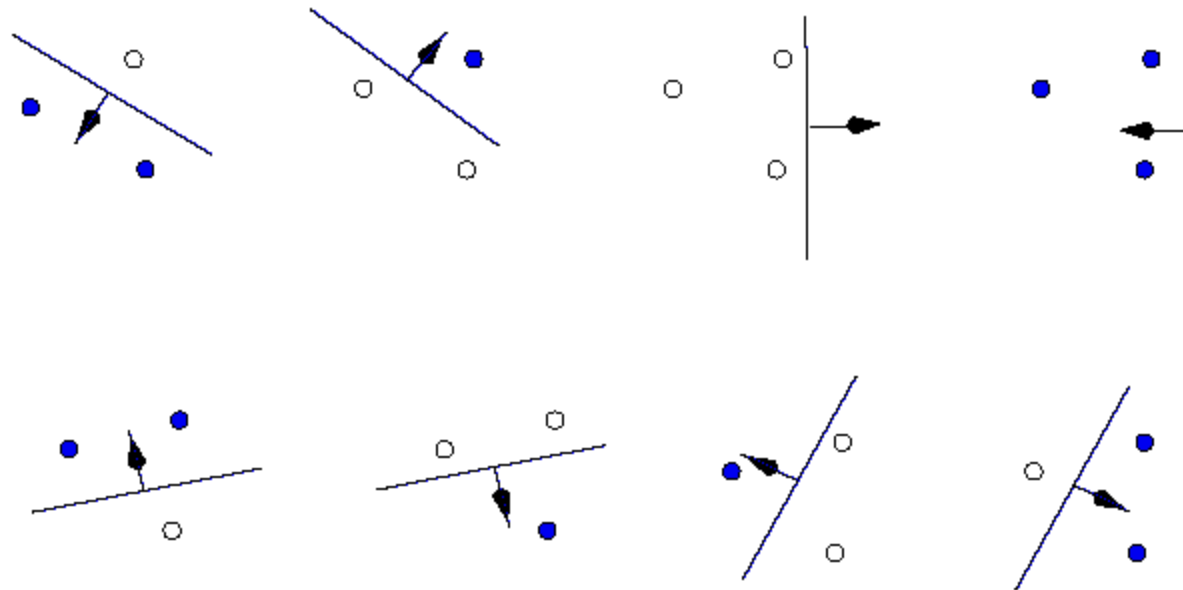
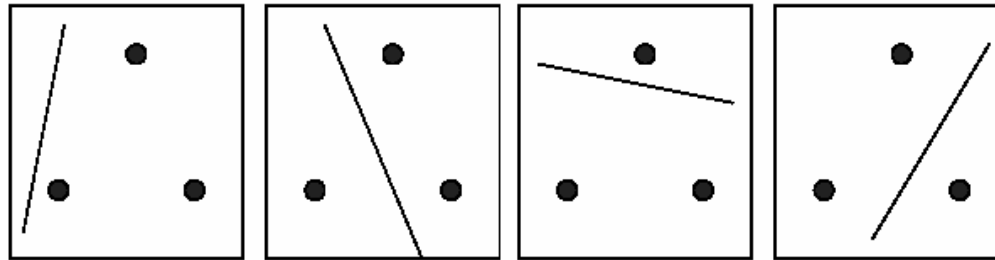


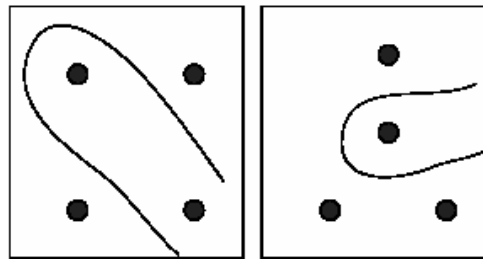
Figure 1. Three points in  $\mathbf{R}^2$ , shattered by oriented lines.

# VC Dimension of Hyperplanes in $\mathbb{R}^2$

- Three points in  $\mathbb{R}^2$  can be shattered with hyperplanes.



- Four points cannot be shattered.



VC dimension provides  
the complexity of a  
class of decision  
functions

$\Rightarrow$  Hyperplanes in  $\mathbb{R}^2 \rightarrow VCdim=3$

Hyperplanes in  $\mathbb{R}^d \rightarrow VC \text{ Dimension} = d+1$

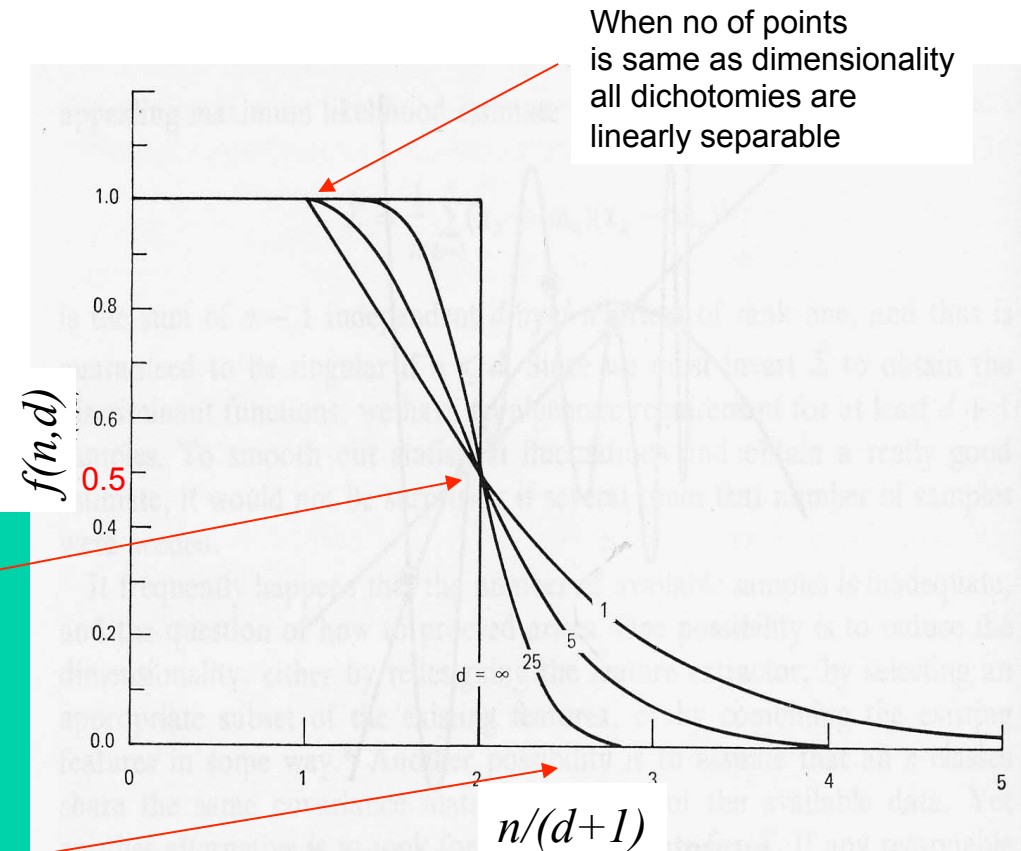
# Fraction of Dichotomies that are linearly separable

$$f(n, d) = \begin{cases} 2^{-n} \sum_{i=0}^d \binom{n-1}{i} & n \leq d+1 \\ 0 & n > d+1 \end{cases}$$

## Capacity of a hyperplane

At  $n = 2(d+1)$ , called the capacity of the hyperplane nearly one half of the dichotomies are still linearly separable

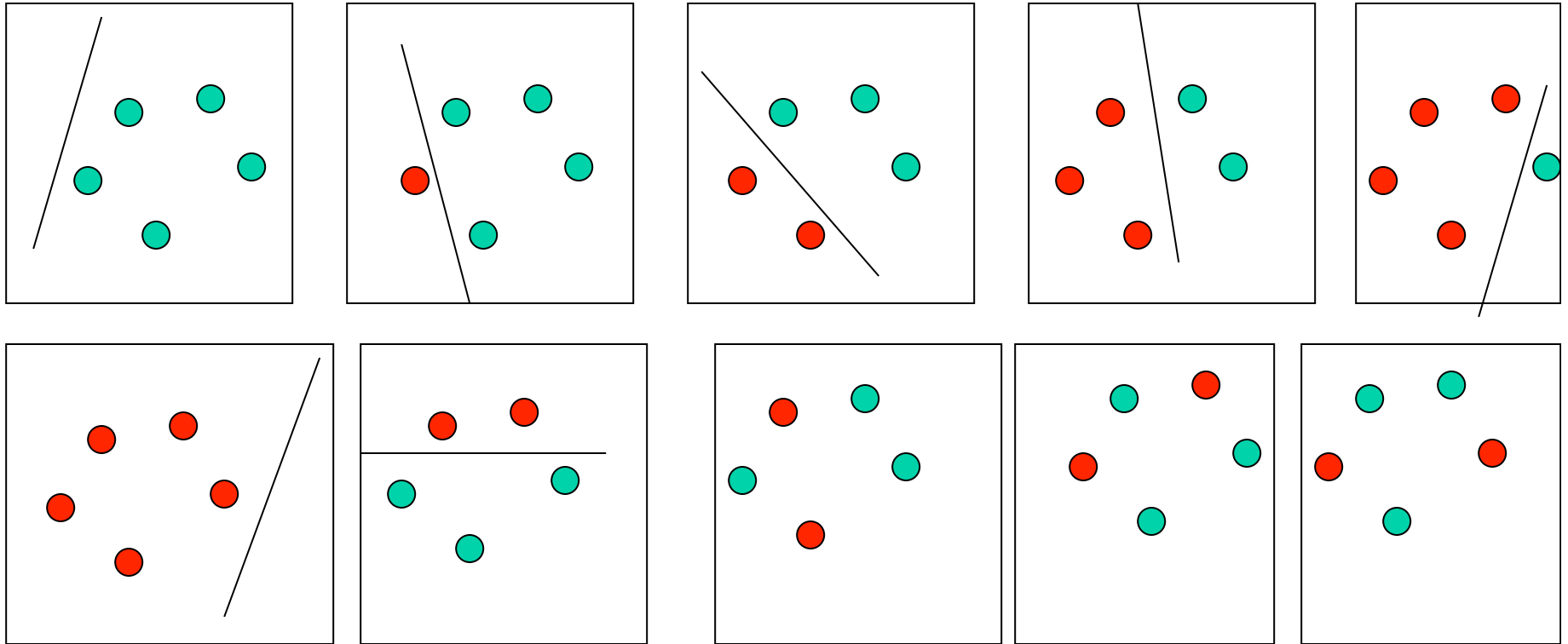
Hyperplane is not over-determined until number of samples is several times the dimensionality



Fraction of dichotomies of  $n$  points in  $d$  dimensions that are linear

# Capacity of a line when $d=2$

## Some Separable Cases



$f(5,2) = 0.5$ ,  
 i.e., half the dichotomies are linear  
 Capacity is achieved at  $n = 2d+1 = 5$

VC Dimension =  $d+1 = 3$

Some Non-separable Cases

# Possible method of training SVM

- Based on modification of Perceptron training rule given below

$$\vec{y}_1, y_2, \vec{y}_3, \vec{y}_1, \vec{y}_2, y_3, y_1, \vec{y}_2, \dots$$

$$\begin{aligned} \mathbf{a}(1) & \text{ arbitrary} \\ \mathbf{a}(k+1) &= \mathbf{a}(k) + \mathbf{y}^k \quad k \geq 1 \end{aligned}$$

## ■ Algorithm 4. (Fixed-Increment Single-Sample Perceptron)

```

1 begin initialize  $\mathbf{a}, k \leftarrow 0$ 
2   do  $k \leftarrow (k + 1) \bmod n$ 
3     if  $\mathbf{y}^k$  is misclassified by  $\mathbf{a}$  then  $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{y}^k$ 
4   until all patterns properly classified
5   return  $\mathbf{a}$ 
6 end

```

Instead of all misclassified samples, use worst classified samples

# Support vector samples

- Support vectors are
  - training samples that define the optimal separating hyperplane
  - They are the most difficult to classify
  - Patterns most informative for the classification task
- Worst classified pattern at any stage is the one on the wrong side of the decision boundary farthest from the boundary
- At the end of the training period such a pattern will be one of the support vectors
- Finding worst case pattern is computationally expensive
  - For each update, need to search through entire training set to find worst classified sample
  - Only used for small problems
  - More commonly used method is different



# Generalization Error of SVM

- If there *are*  $n$  training patterns
- Expected value of the generalization error rate is bounded according to

$$\mathcal{E}_n[P(\text{error})] \leq \frac{\mathcal{E}_n[\text{No. of Support Vectors}]}{n}$$

Expected value of error  $\leq$  expected no of support vectors/ $n$

- Where expectation is over all training sets of size  $n$  (drawn from distributions describing the categories)
- This also means that error rate on the support vectors will be  $n$  times the error rate on the total sample

- Leave one out bound

- If we have  $n$  points in the training set
- Train SVM on  $n-1$  of them
- Test on single remaining point
- If the point is a support vector then there will be an error
- If we find a transformation  $f$ 
  - that well separates the data, then
    - expected number of support vectors is small
    - expected error rate is small

## 8. Relevance Vector Machines

- Addresses several limitations of SVMs
  - SVM does not provide *a posteriori* probabilities
    - Relevance Vector Machines provide such output
  - Extension of SVM to multiclass is problematic
  - Complexity parameter C or  $\gamma$  that must be found using a hold-out method
  - Linear combinations of kernel functions centered on training data points that must be positive definite