Support Vector Machines

1. Importance of SVM

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

2. Mathematical Techniques Used

1. Linearly separable case considered

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

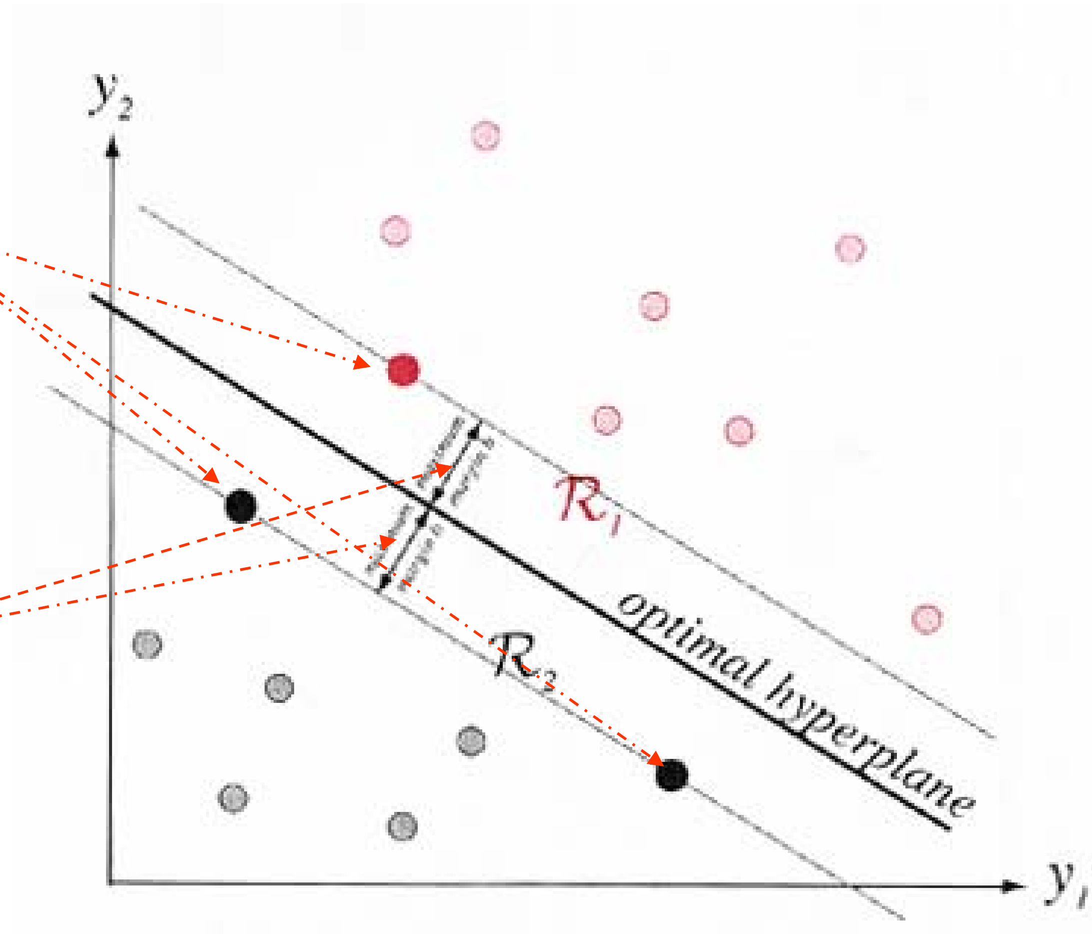
2. To handle non-linear separability:
   - Preprocessing data to represent in much higher-dimensional space than original feature space
   - Kernel trick reduces computational overhead

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

- Why maximize margin?
- Motivation found in computational learning theory or statistical learning theory (PAC learning-VC dimension)
- Insight given as follows (Tong, Koller 2000):
  - Model distributions for each class using Parzen density estimators using Gaussian kernels with common parameter σ^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 point and 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 $r = \dfrac{g(x)}{\|w\|}$

Proof: Let $x = x_p + r\dfrac{w}{\|w\|}$ where r is the distance from x to the plane,

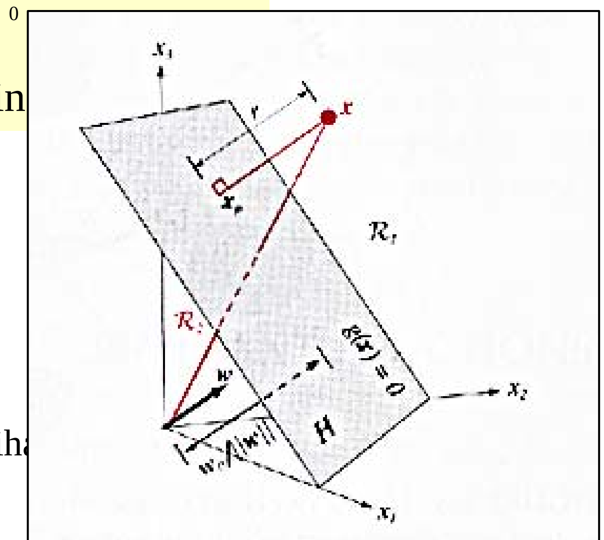
$$g(x) = w^t\left(x_p + r\frac{w}{\|w\|}\right) + w_0 = w^t x_p + w_0 + r\frac{w^t w}{\|w\|} = \underbrace{g(x_p)}_{=0} + r\frac{\|w\|^2}{\|w\|} = r\|w\| \qquad \text{QED}$$

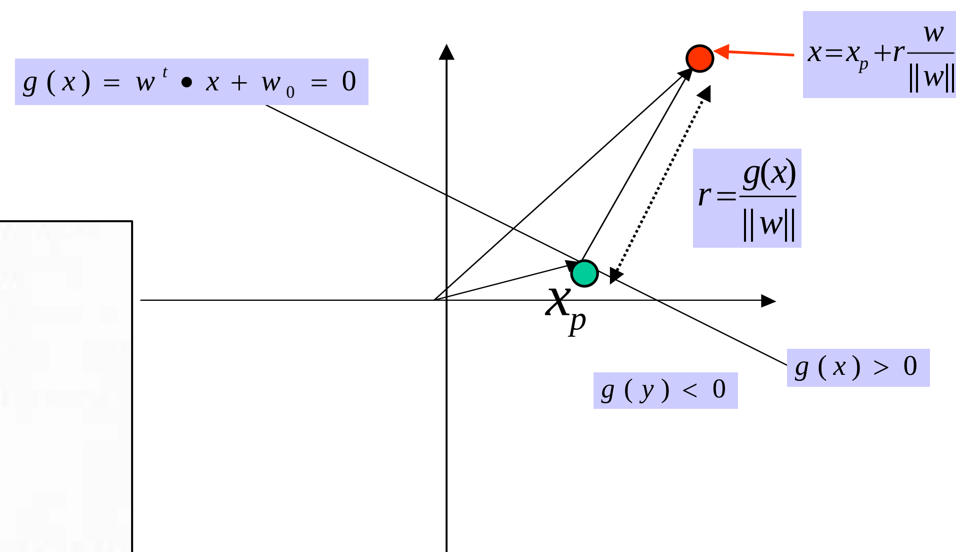
Corollary: Distance of origin to plane is

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

since $\mathbf{g}(0) = \mathbf{w}^t 0 + \mathbf{w}_0 = \mathbf{w}_0$

Thus w_0=0 implies that plane passes through origin





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 ω_1 or ω_2

- Thus if $g(y)=0$ is a separating hyperplane then

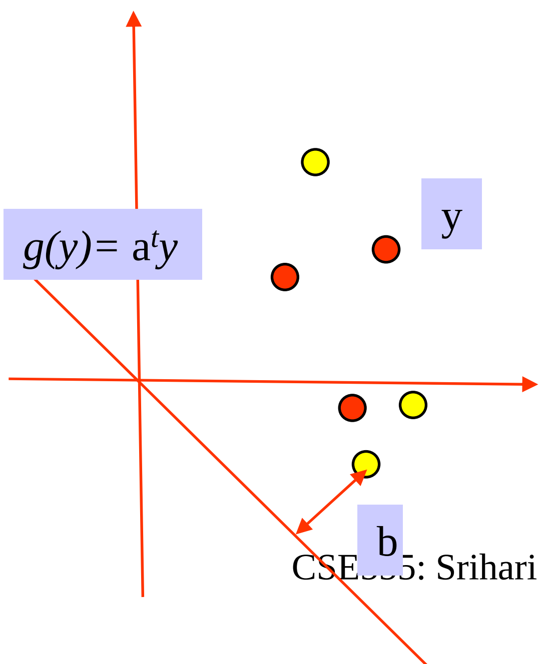
$$z_k\, g(y_k) \geq 0,\ k = 1,.., n$$

- Since distance of a point y to hyperplane $g(y)=0$ is

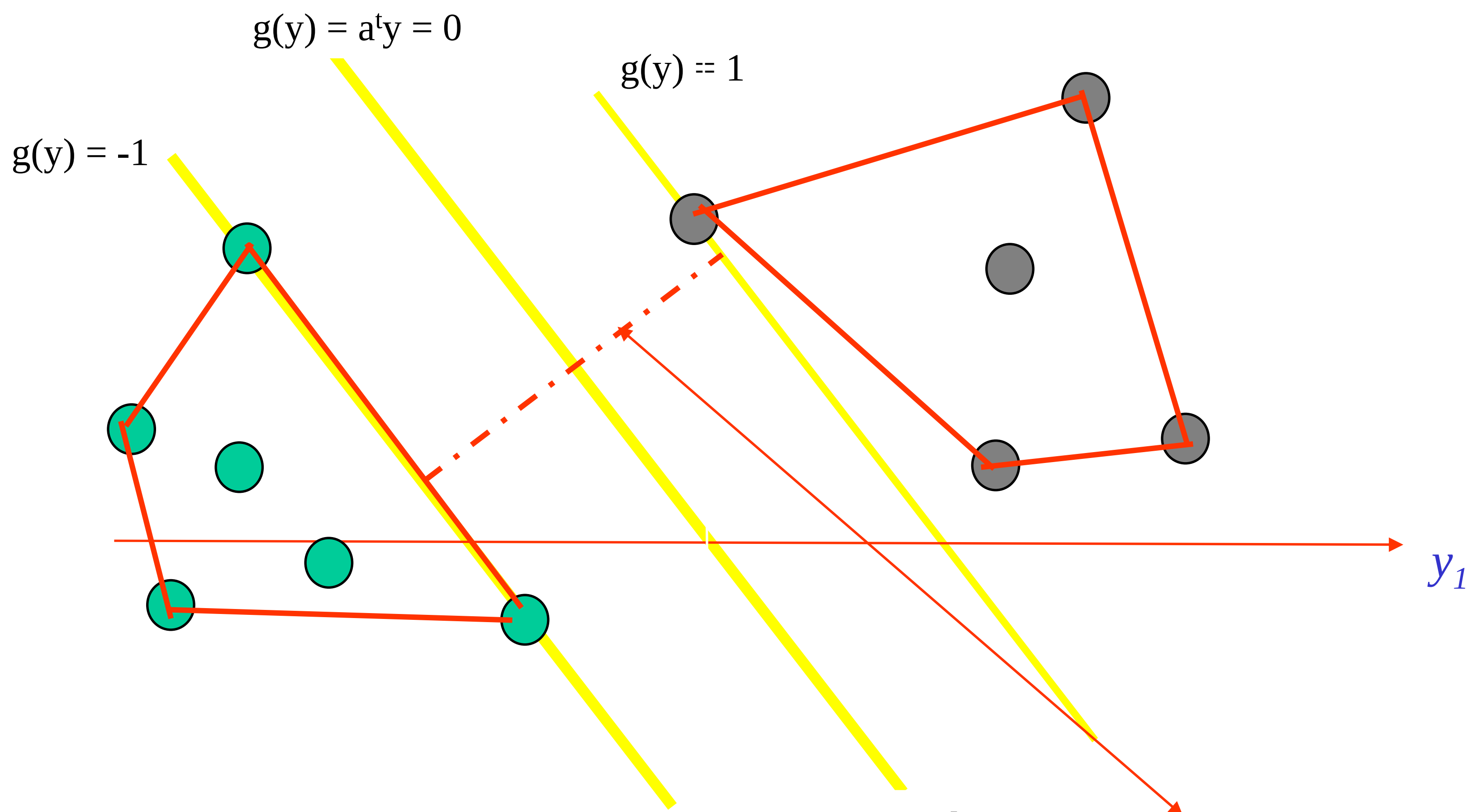
$$\frac{g(y)}{\| a \|}$$

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



Optimal hyperplane is orthogonal to..

Shortest line connecting Convex hulls, which has length = $2/||a||$

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

Kernel Function: key property

- If kernel function is chosen with property

$$K(\mathbf{x},\mathbf{y}) = (\phi(\mathbf{x}). \quad \phi(\mathbf{y}))$$

then computational expense of increased dimensionality is avoided.

- Polynomial kernel $K(\mathbf{x},\mathbf{y}) = (\mathbf{x}.\mathbf{y})^d$

can be shown (next slide) to correspond to a map ϕ into the space spanned by **all** products of exactly d dimensions.

A Polynomial Kernel Function

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

The feature space mapping is :

$$\varphi(\mathrm{x}) = \left(x_1{}^2, \sqrt{2}x_1x_2, x_2{}^2\right)^t$$

Then inner product is

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

same

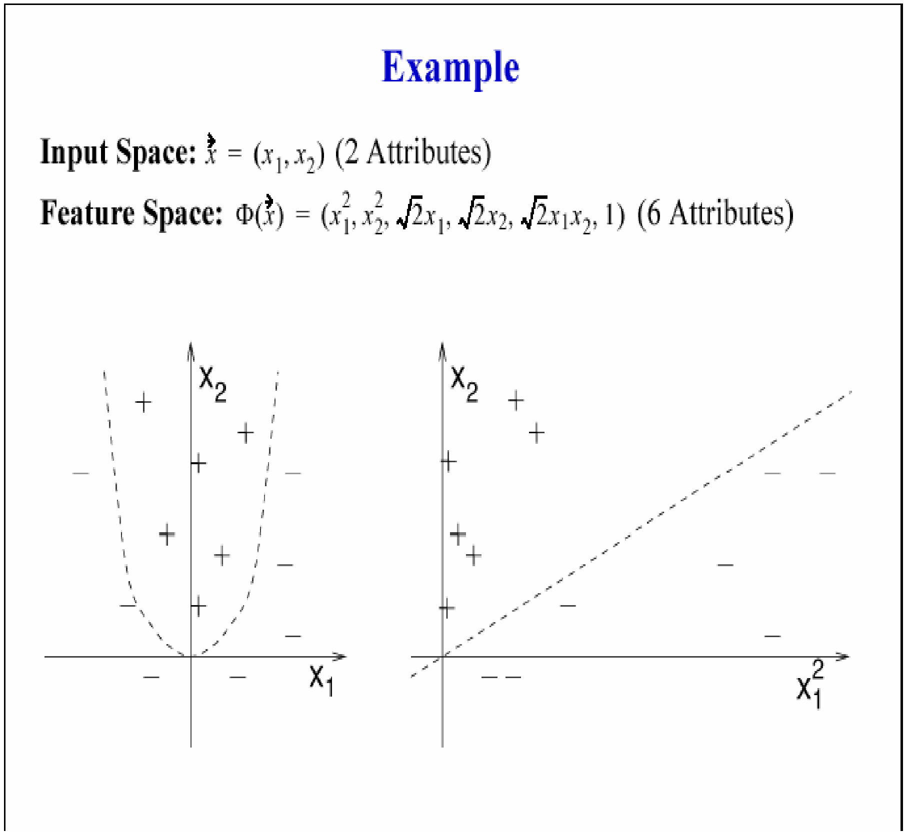
Polynomial kernel function to compute the same value is

$$K(\mathrm{x},\mathrm{y}) = (\mathrm{x}.\mathrm{y})^2 = \left((x_1, x_2)(y_1, y_2)^t\right)^2 = (x_1y_1 + x_2y_2)^2$$

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

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

Another Polynomial (quadratic) kernel function



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

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

where

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

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

SVM with Kernels

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

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

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

New hypotheses spaces through new Kernels:

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

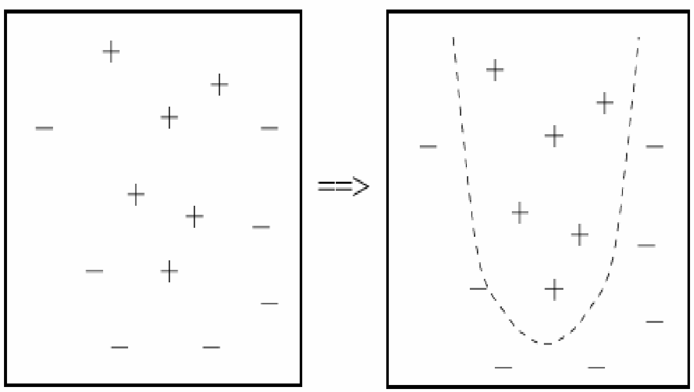
Polynomial: $K(\vec{x}_i, \vec{x}_j) = [\vec{x}_i \cdot \vec{x}_j + 1]^d$

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 $\phi(.)$ 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

  $\boldsymbol{y}_k = \phi(\boldsymbol{x}_k)$, *for* $k=1,.., n$
- First choose the non-linear ϕ 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

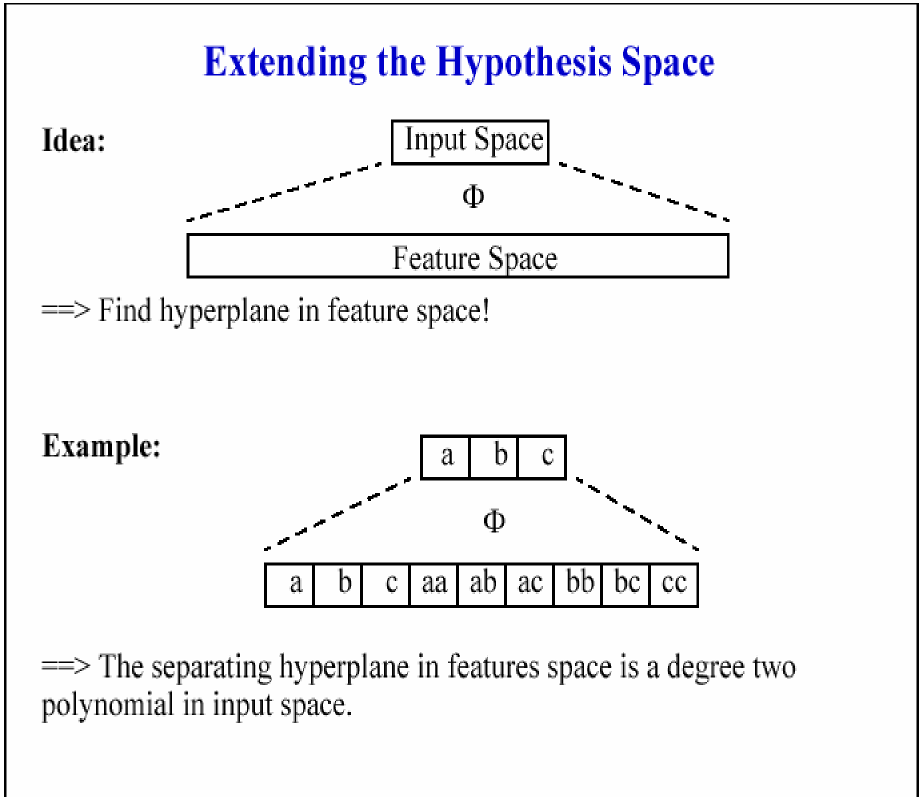
- Mapping each input point x by map

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

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

$$g(x) = a_1 y_1 + a_2 y_2 + a_3 y_3$$



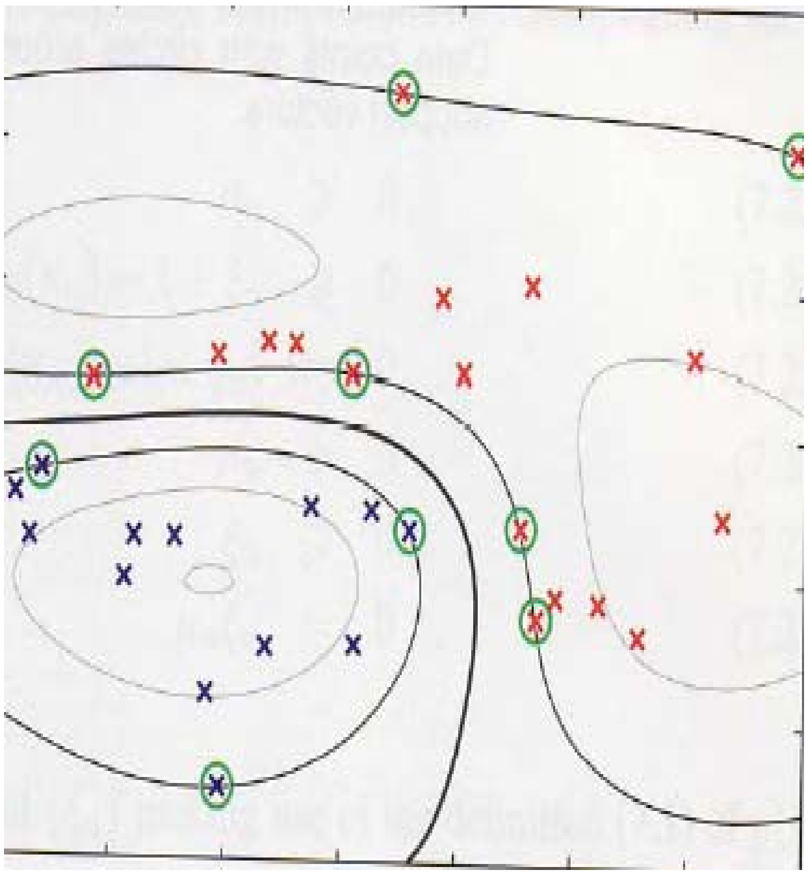
Pattern Transformation using Kernels

- 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

$$\mathbf{y}_\mathbf{k} = \Phi(\mathbf{x}_\mathbf{k})$$

  - 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

Demo

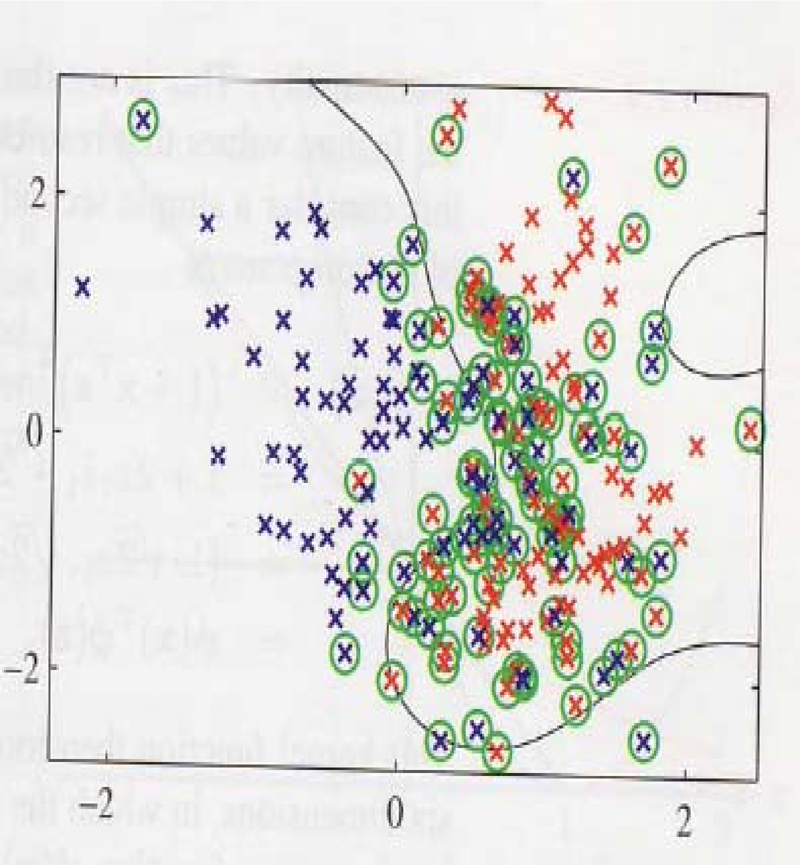
http://www.csie.ntu.edu.tw/~cjlin/libsvm/

Svmtoy.exe

5. Overlapping Class Distributions

- We assumed training data are linearly separable in the mapped space y
- Resulting SVM will give exact separation in input space x although decision boundary will be nonlinear
- In practice class-conditional distributions will overlap
- In which case exact separation of training data will lead to poor generalization
- Therefore need to allow SVM to misclassify some training points

ν-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
- ν 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 fundamentally a two-class classfier
- Several suggested methods for combining multiple two-class classfiers
- Most widely 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 can be 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 is historically motivated and analyzed using a theoretical framework called computational learning theory
- Called Probably Approximately Correct or 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 Vapnik-Chernovenkis (VC) dimension which provides a measure of complexity of a space of functions

All dichotomies of 3 points in 2 dimensions are linearly separable

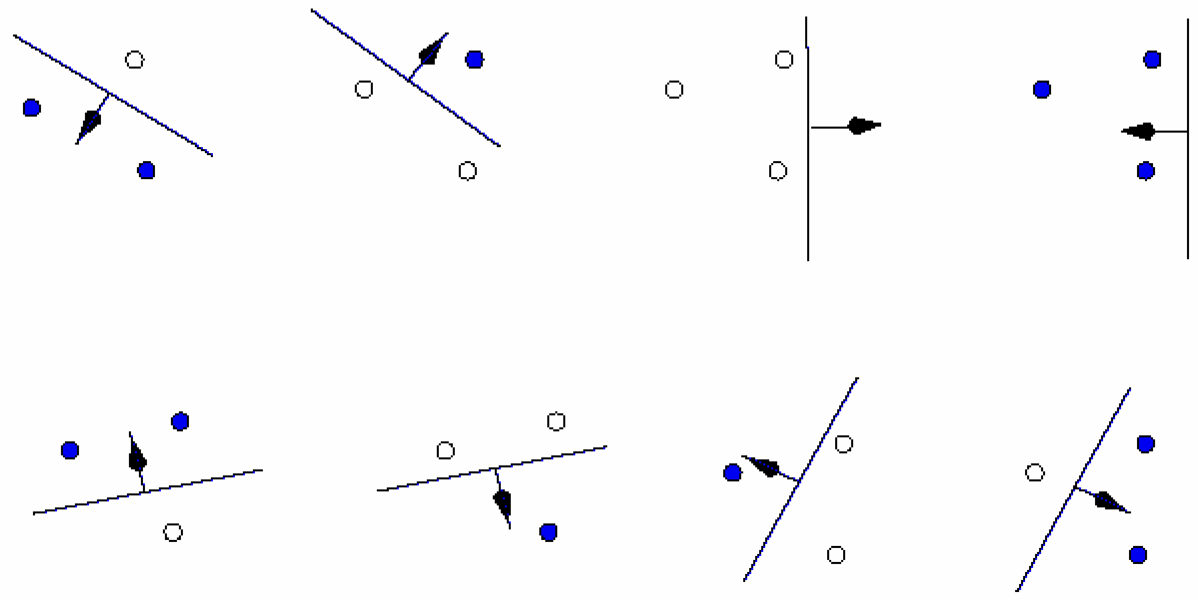
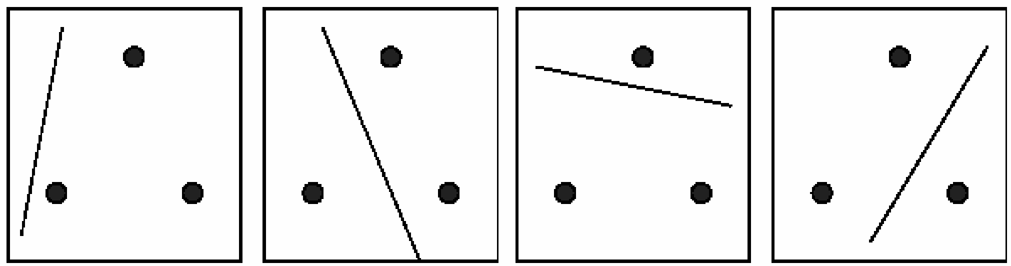


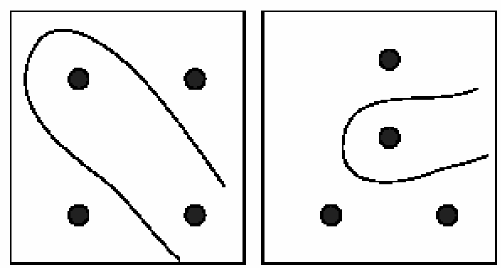
Figure 1. Three points in $\mathbf{R}^2$, shattered by oriented lines.

VC Dimension of Hyperplanes in R^2

- Three points in $\Re^2$ can be shattered with hyperplanes.



- Four points cannot be shattered.



VC dimension provides the complexity of a class of decision functions

=> Hyperplanes in $\Re^2$ -> $VCdim=3$

Hyperplanes in R^d ➔ VC Dimension = d+1

Fraction of Dichotomies that are linearly separable

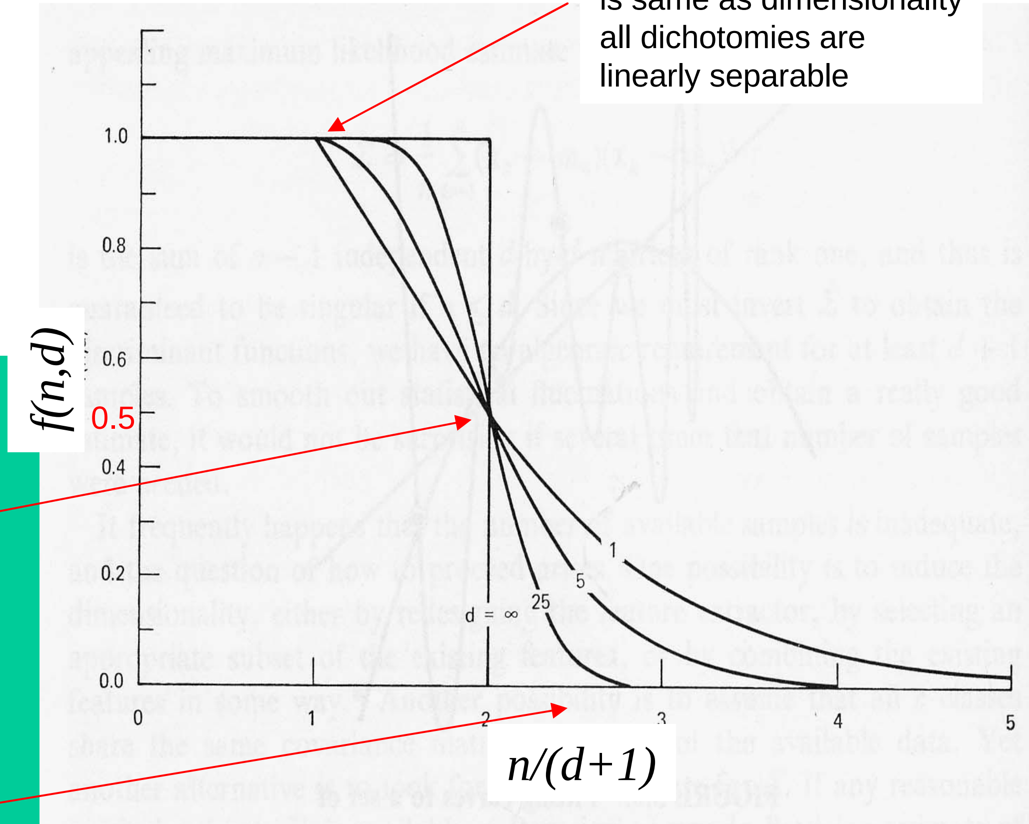
$$f(n,d) = \begin{cases} 1 & n \le d+1 \\ 2^{n} \sum_{i=0}^{d} \binom{n-1}{i} & n > d+1 \end{cases}$$

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

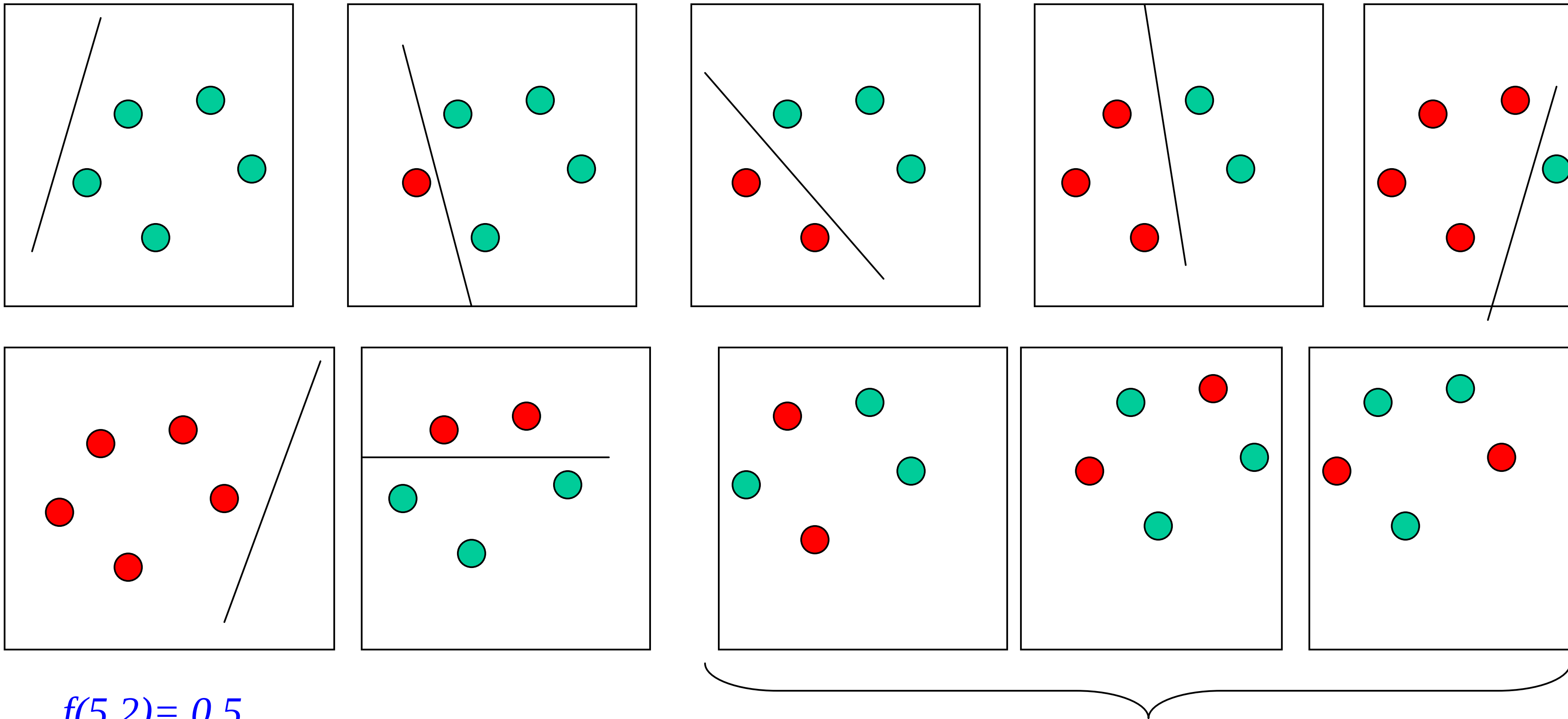
When no of points is same as dimensionality all dichotomies are linearly separable



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

Capacity of a line when d=2

Some Separable Cases



Some Non-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*

Possible method of training SVM

- Based on modification of Perceptron training rule given below

$$\overset{\downarrow}{\mathbf{y}_1},\ \mathbf{y}_2,\ \overset{\downarrow}{\mathbf{y}_3},\ \overset{\downarrow}{\mathbf{y}_1},\ \overset{\downarrow}{\mathbf{y}_2},\ \mathbf{y}_3,\ \mathbf{y}_1,\ \overset{\downarrow}{\mathbf{y}_2},\ \ldots$$

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

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

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

Instead of all misclassified samples, use worst classified samples

Support vectors are worst classified 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

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

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 ϕ
    - that well separates the data, then
      - expected number of support vectors is small
      - expected error rate is small