SVM: Support Vector Machines

1. Lecture Notes for Chapter 5, Introduction to Data Mining, By Tan, Steinbach, Kumar
2. Lecture notes for Chapter 3, The Top Ten Algorithms in Data Mining, By Hui Xue, Qiang Yang, and Songcan Chen
3. Lecture notes by Peter Belhumeur, Columbia University

Edited by Wei Ding

Introduction

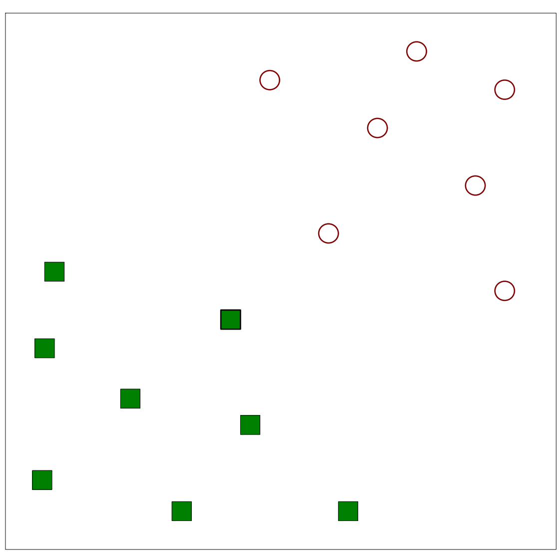

V. Vapnik

- Support vector machines (SVMs), including support vector classifier (SVC) and support vector regressor (SVR), are among the most robust and accurate methods in data mining algorithms.
- SVMs, which were originally developed by Vapnik in the 1990s, have a sound theoretical foundation rooted in statistical learning theory, require only as few as a dozen examples for training, and are often insensitive to the number of dimensions.

Support Vector Classifier

- For a two-class linearly separable learning task, the aim of SVC is to find a hyperplane that can separate two classes of given samples with a maximal margin which has been proved able to offer the best generalization ability.
- Generalization ability refers to the fact that a classifier not only has good classification performance on the training data, but also guarantees high predictive accuracy for the future data from the same distribution as the training data.

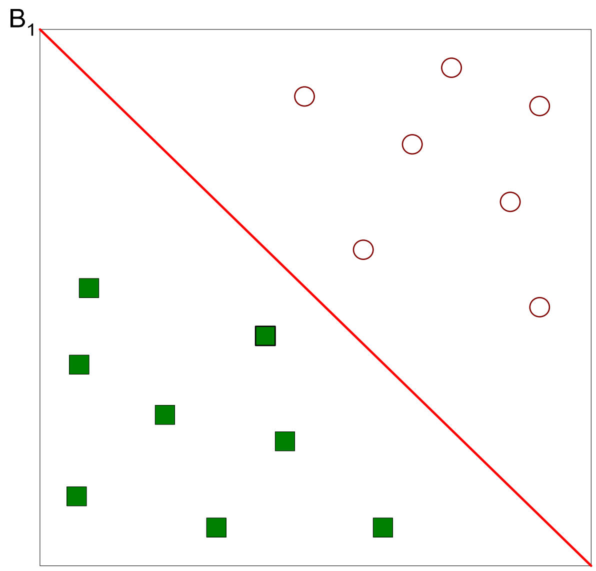
Support Vector Machines



- Find a linear hyperplane (decision boundary) that will separate the data

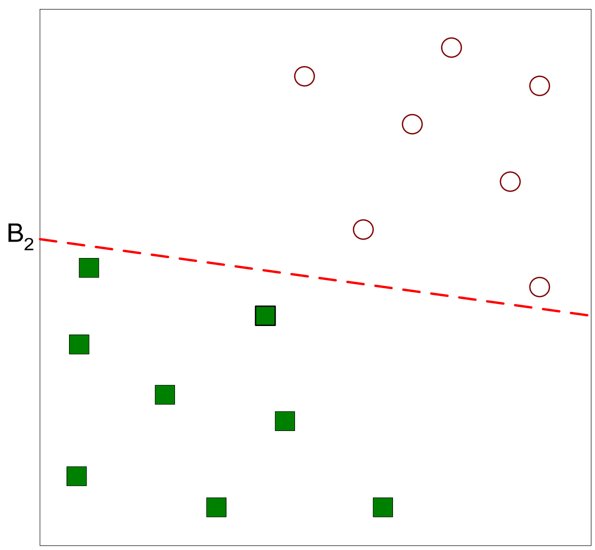
© Tan,Steinbach, Kumar Introduction to Data Mining 4/18/2004 ‹#›

Support Vector Machines



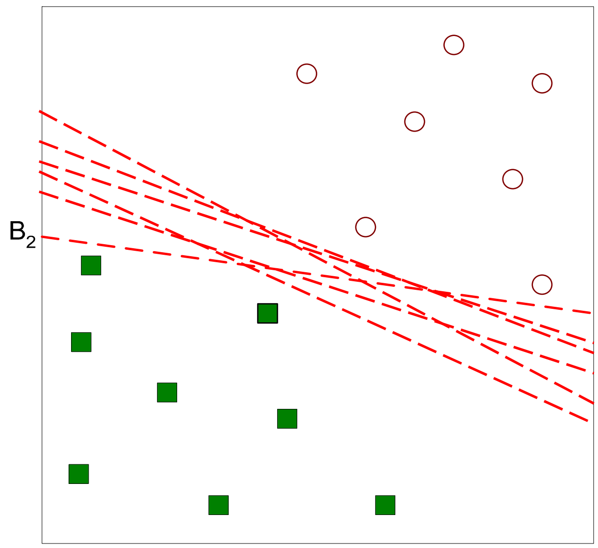
- One Possible Solution

Support Vector Machines



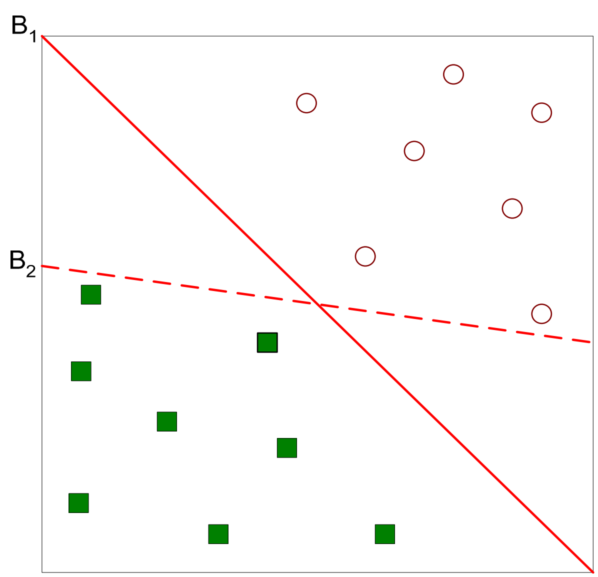
- Another possible solution

Support Vector Machines



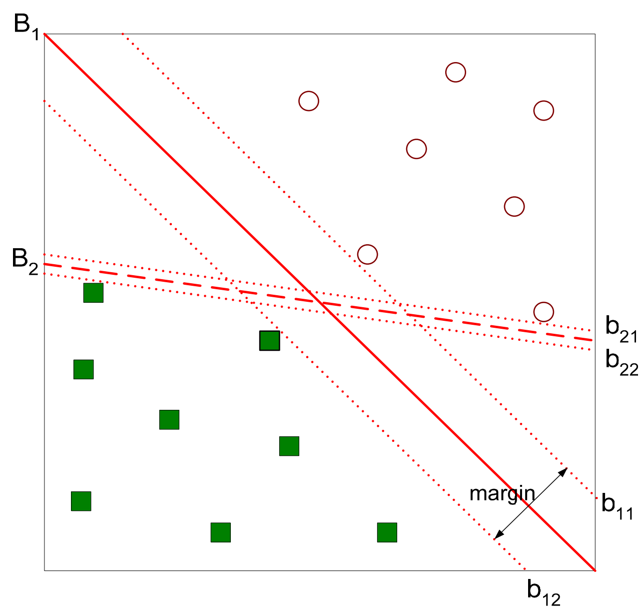
- Other possible solutions

Support Vector Machines



- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



- Find hyperplane maximizes the margin => B1 is better than B2

Margin

- Intuitively, a margin can be defined as the amount of space, or separation, between the two classes as defined by a hyperplane.
- Geometrically, the margin corresponds to the shortest distance between the closest data points to any point on the hyperplane.

Linear SVM: Separable Case

- A linear SVM is a classifier that searches for a hyperplane with the largest margin, which is why it is often known as a maximal margin classifier.
- N training examples
- Each example $(\mathbf{x_i}, y_i)$ $(i=1,2,\ldots,N)$, $\mathbf{x_i}=(x_{i1},x_{i2},\ldots,x_{id})^T$ corresponds to the attribute set for the i_{th} example (that is, each object is represented by d attributes), y_i is in {-1,1} that denotes its class label.
- The decision boundary of a linear classifier can be written in the following form: **w.x** + b = 0 (where the weight vector **w** and bias b are the parameters of the model)

Linear Discriminant Function

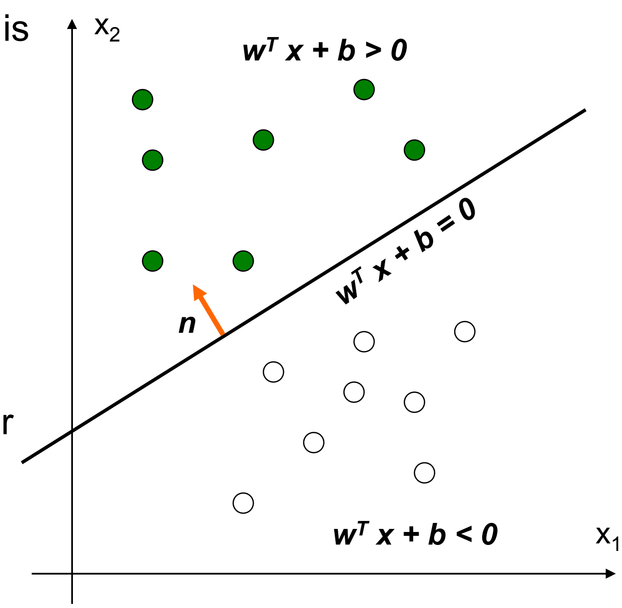
- Discriminant function g(x) is a linear function:

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$$

- A hyper-plane in the feature space

- (Unit-length) normal vector of the hyper-plane:

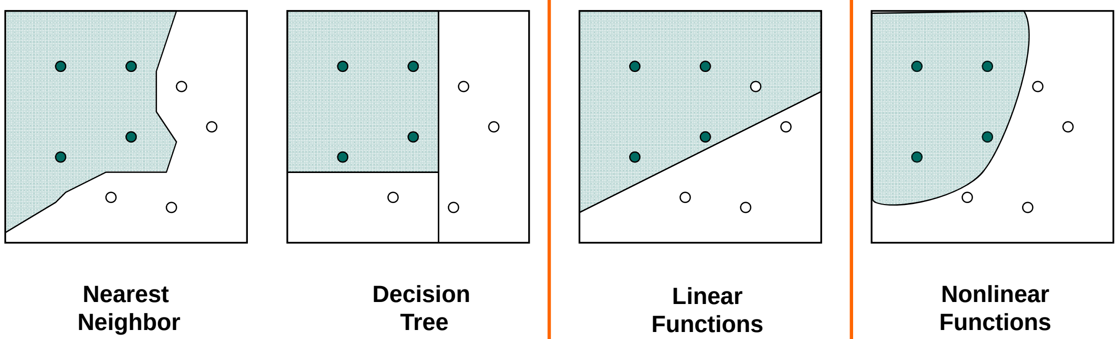
$$\mathbf{n} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$$



The vector w defines a direction perpendicular to the hyperplane, while varying the value of b moves the hyperplane parallel to itself.

Discriminant Function

- It can be arbitrary functions of x, such as:



$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$$

Large Margin Linear Classifier

- We know that

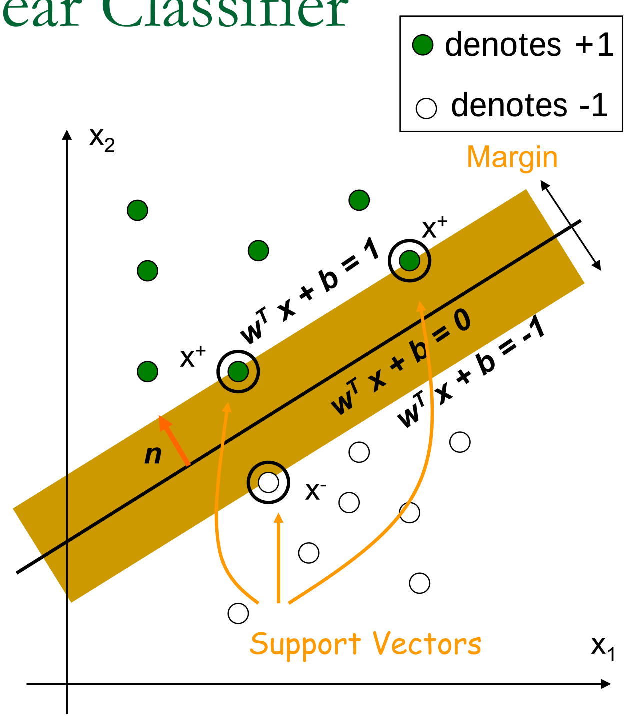
$$\mathbf{w}^T\mathbf{x}^+ + b = 1$$

$$\mathbf{w}^T\mathbf{x}^- + b = -1$$

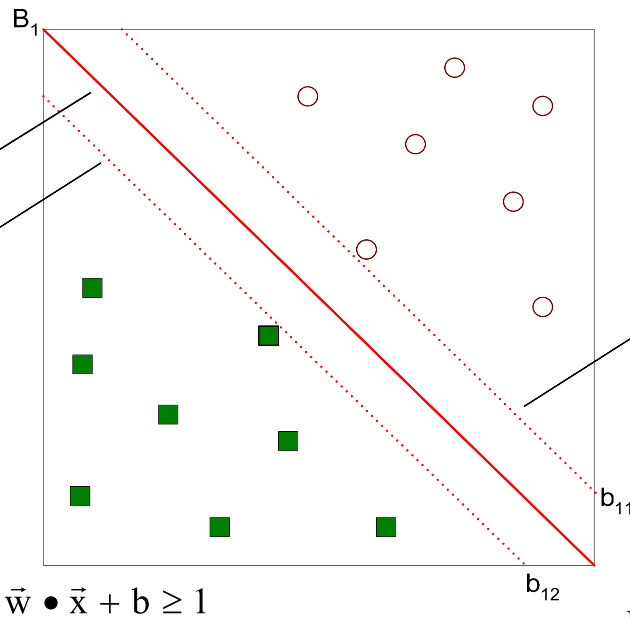
- The margin width is:

$$M = (\mathbf{x}^+ - \mathbf{x}^-)\cdot\mathbf{n}$$

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



Support Vector Machines



$\vec{w} \bullet \vec{x} + b = 0$

$\vec{w} \bullet \vec{x} + b = -1$

$\vec{w} \bullet \vec{x} + b = +1$

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\vec{w}\|^2}$$

Support Vector Machines

- We want to maximize: $\text{Margin} = \frac{2}{\|\vec{w}\|^2}$
  - Which is equivalent to minimizing: $L(w) = \frac{\|\vec{w}\|^2}{2}$
  - But subjected to the following constraints:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 \end{cases}$$

    - This is a constrained optimization problem
      - Numerical approaches to solve it (e.g., lagrange multiplier)

Solving the Optimization Problem

Quadratic programming with linear constraints

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$

$$\Updownarrow$$

Lagrangian Function

$$\text{minimize} \quad L_p(\mathbf{w}, b, \alpha_i) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \right)$$

$$\text{s.t.} \quad \alpha_i \geq 0$$

Solving the Optimization Problem

$$\text{minimize} \quad L_p(\mathbf{w}, b, \alpha_i) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \right)$$

$$\text{s.t.} \quad \alpha_i \geq 0$$

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

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

Solving the Optimization Problem

$$\text{minimize} \;\; L_p(\mathbf{w}, b, \alpha_i) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right)$$

$$\text{s.t.} \quad \alpha_i \geq 0$$

Lagrangian Dual Problem

$$\text{maximize} \;\; \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

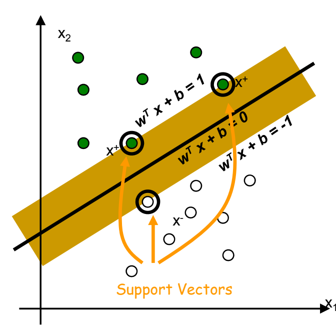
$$\text{s.t.} \quad \alpha_i \geq 0 \text{ , and } \sum_{i=1}^{n} \alpha_i y_i = 0$$

Solving the Optimization Problem

- From KKT condition, we know the optimum:

$$\alpha_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right) = 0$$

- Thus, only support vectors have $\alpha_i \neq 0$



- The solution has the form:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = \sum_{i \in SV} \alpha_i y_i \mathbf{x}_i$$

get b from $y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 = 0$,

where $\mathbf{x}_i$ is support vector

Solving the Optimization Problem

- The linear discriminant function is:

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

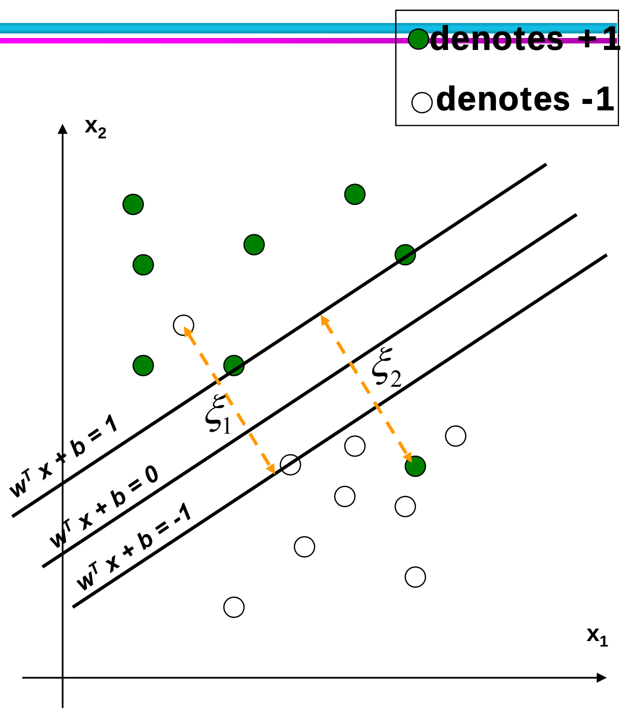
- Notice it relies on a *dot product* between the test point ***x*** and the support vectors $\boldsymbol{x}_i$

- Also keep in mind that solving the optimization problem involved computing the dot products $\boldsymbol{x}_i^T \boldsymbol{x}_j$ between all pairs of training points

Large Margin Linear Classifier

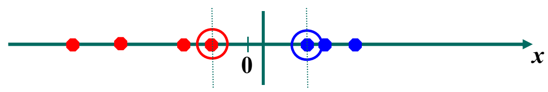
- What if data is not linear separable? (noisy data, outliers, etc.)

- **Slack variables ξ_i can be added to allow mis-classification of difficult or noisy data points**



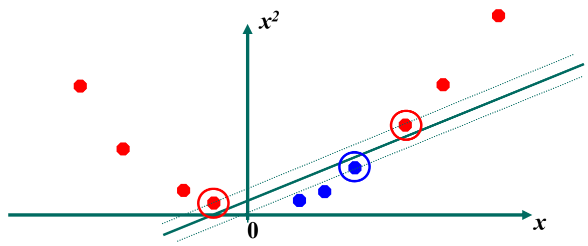
Non-linear SVMs

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



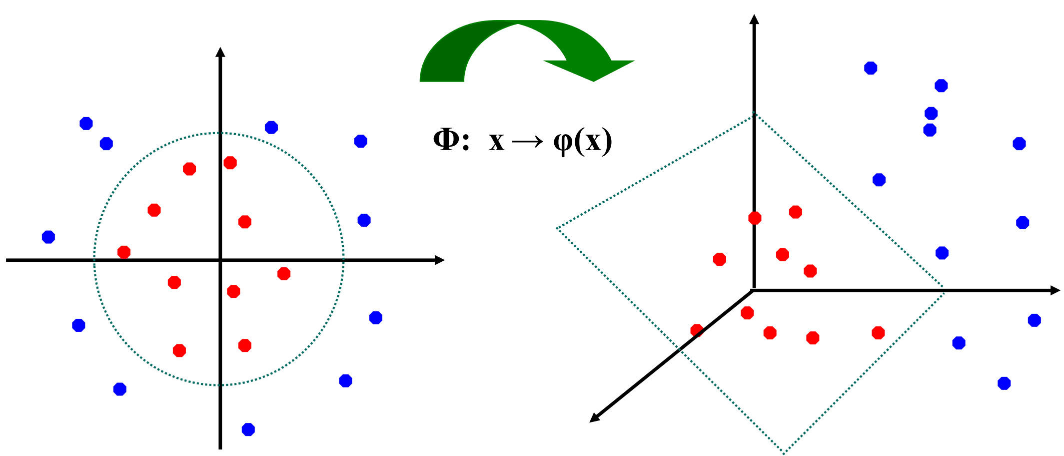
- But what are we going to do if the dataset is just too hard?

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



Non-linear SVMs: Feature Space

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



Nonlinear SVMs: The Kernel Trick

- With this mapping, our discriminant function is now:

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

- No need to know this mapping explicitly, because we only use the dot product of feature vectors in both the training and test.

- A *kernel function* is defined as a function that corresponds to a dot product of two feature vectors in some expanded feature space:

$$K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

Nonlinear SVMs: The Kernel Trick

- An example:

2-dimensional vectors $\mathbf{x}=[x_1\ \ x_2]$;

let $K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^T\mathbf{x}_j)^2$,

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

$K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^T\mathbf{x}_j)^2$,

$= 1+ x_{i1}^2x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}$

$= [1\ \ x_{i1}^2\ \ \sqrt{2}\, x_{i1}x_{i2}\ \ x_{i2}^2\ \ \sqrt{2}x_{i1}\ \ \sqrt{2}x_{i2}]^T\, [1\ \ x_{j1}^2\ \ \sqrt{2}\, x_{j1}x_{j2}\ \ x_{j2}^2\ \ \sqrt{2}x_{j1}\ \ \sqrt{2}x_{j2}]$

$= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$, where $\varphi(\mathbf{x}) = [1\ \ x_1^2\ \ \sqrt{2}\, x_1x_2\ \ x_2^2\ \ \sqrt{2}x_1\ \ \sqrt{2}x_2]$

This slide is courtesy of *www.iro.umontreal.ca/~pift6080/documents/papers/svm_tutorial.ppt*

Nonlinear SVMs: The Kernel Trick

- **Examples of commonly-used kernel functions:**
  - **Linear kernel:** $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
  - **Polynomial kernel:** $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$
  - **Gaussian (Radial-Basis Function (RBF)) kernel:**

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{2\sigma^2}\right)$$

  - **Sigmoid:**

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$$

- **In general, functions that satisfy *Mercer's condition* can be kernel functions.**

Nonlinear SVM: Optimization

- **Formulation: (Lagrangian Dual Problem)**

$$\text{maximize} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

such that

$$0 \le \alpha_i \le C$$

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

- **The solution of the discriminant function is**

$$g(\mathbf{x}) = \sum_{i \in \mathrm{SV}} \alpha_i K(\mathbf{x}_i, \mathbf{x}) + b$$

- **The optimization technique is the same.**

Kernel Trick

- The significance of the kernel is that we may use it to construct the optimal hyperplane in the feature space without having to consider the concrete form of the transformation φ.

Support Vector Machine: Algorithm

- 1. Choose a kernel function
- 2. Choose a value for *C*
- 3. Solve the quadratic programming problem (many software packages available)
- 4. Construct the discriminant function from the support vectors

Some Issues

- Choice of kernel

  - Gaussian or polynomial kernel is default

  - if ineffective, more elaborate kernels are needed

  - domain experts can give assistance in formulating appropriate similarity measures

- Choice of kernel parameters

  - e.g. σ in Gaussian kernel

  - σ is the distance between closest points with different classifications

  - In the absence of reliable criteria, applications rely on the use of a validation set or cross-validation to set such parameters.

- Optimization criterion – Hard margin v.s. Soft margin

  - a lengthy series of experiments in which various parameters are tested

Summary: Support Vector Machine

- 1. Large Margin Classifier
  – Better generalization ability & less over-fitting

- 2. The Kernel Trick
  – Map data points to higher dimensional space in order to make them linearly separable.
  – Since only dot product is used, we do not need to represent the mapping explicitly.

Summary

- SVM has its roots in statistical learning theory
- It has shown promising empirical results in many practical applications, from handwritten digit recognition to text categorization
- Works very well with high-dimensional data and avoids the curse of dimensionality problem
- A unique aspect of this approach is that it represents the decision boundary using a subset of the training examples, known as the support vectors.