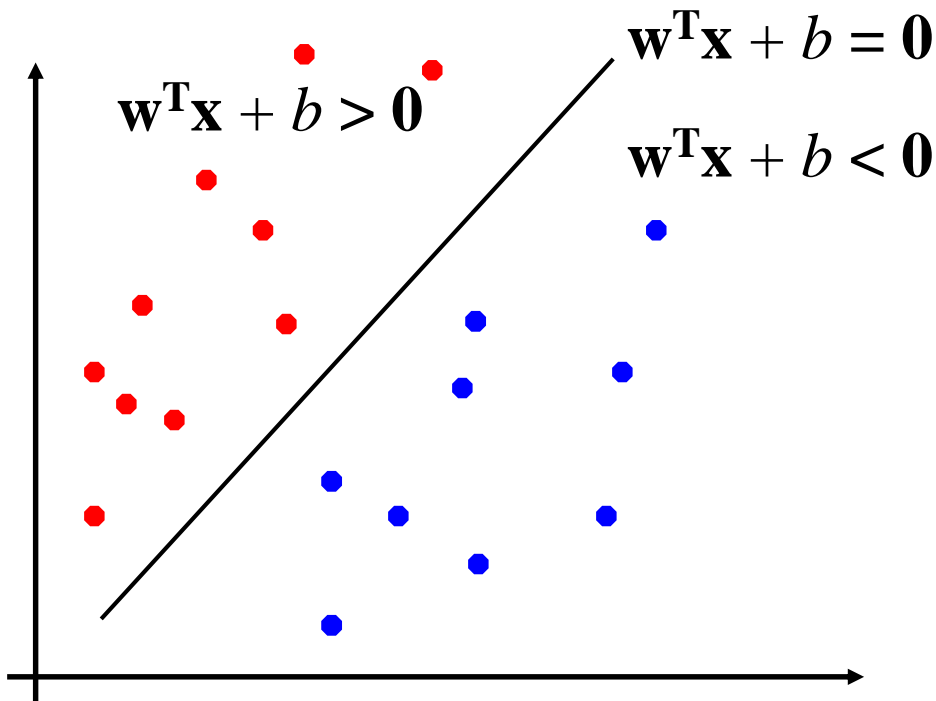


# Support Vector Machines

---

## Perceptron Revisited: Linear Separators

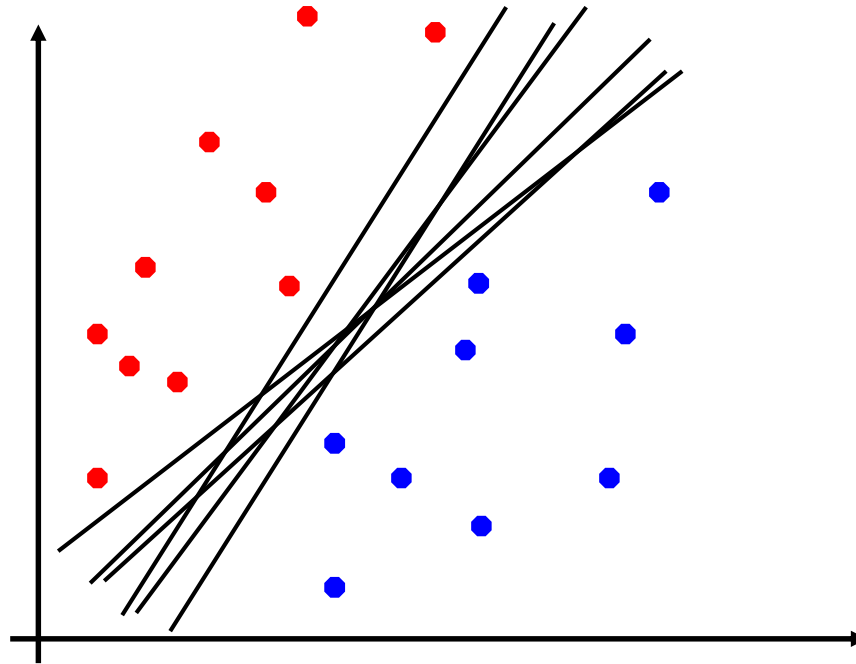
- Binary classification can be viewed as the task of separating classes in feature space:



$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + b)$$

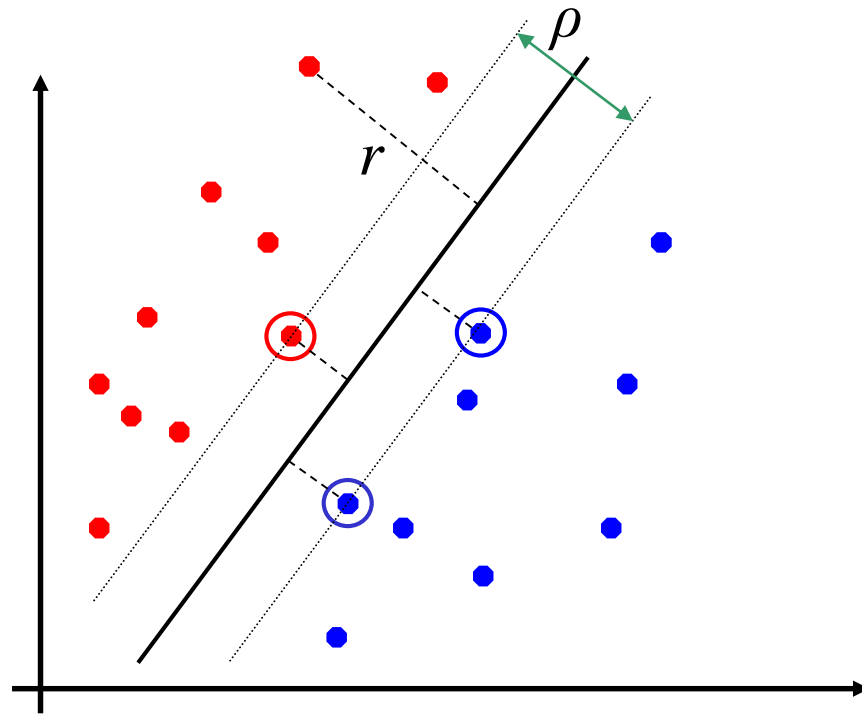
# Linear Separators

- Which of the linear separators is optimal?



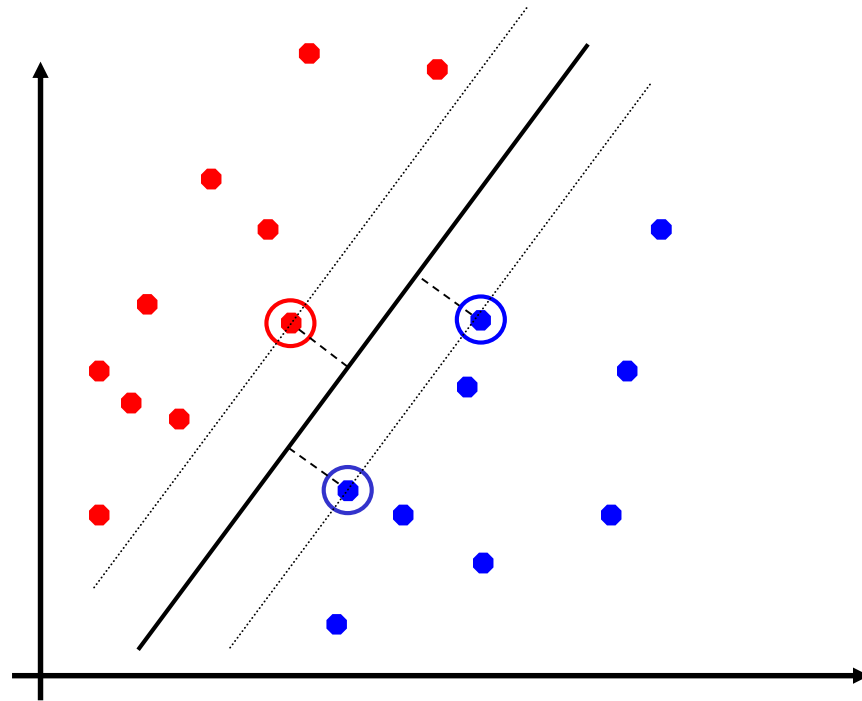
# Classification Margin

- 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*  $\rho$  of the separator is the distance between support vectors.



# Maximum Margin Classification

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



## Linear SVM Mathematically

- Let training set  $\{(\mathbf{x}_i, y_i)\}_{i=1..n}$ ,  $\mathbf{x}_i \in \mathbf{R}^d$ ,  $y_i \in \{-1, 1\}$  be separated by a hyperplane with margin  $\rho$ . Then for each training example  $(\mathbf{x}_i, y_i)$ :

$$\begin{aligned} \mathbf{w}^T \mathbf{x}_i + b &\leq -\rho/2 & \text{if } y_i = -1 \\ \mathbf{w}^T \mathbf{x}_i + b &\geq \rho/2 & \text{if } y_i = 1 \end{aligned} \quad \Leftrightarrow \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq \rho/2$$

- For every support vector  $\mathbf{x}_s$  the above inequality is an equality. After rescaling  $\mathbf{w}$  and  $b$  by  $\rho/2$  in the equality, we obtain that distance between each  $\mathbf{x}_s$  and the hyperplane is  $r = \frac{y_s(\mathbf{w}^T \mathbf{x}_s + b)}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}$
- Then the margin can be expressed through (rescaled)  $\mathbf{w}$  and  $b$  as:

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

## Linear SVMs Mathematically (cont.)

- Then we can formulate the *quadratic optimization problem*:

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

$$\rho = \frac{2}{\|\mathbf{w}\|} \text{ is maximized}$$

and for all  $(\mathbf{x}_i, y_i), i=1..n :$   $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$

Which can be reformulated as:

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

$$\Phi(\mathbf{w}) = \|\mathbf{w}\|^2 = \mathbf{w}^T \mathbf{w} \text{ is minimized}$$

and for all  $(\mathbf{x}_i, y_i), i=1..n :$   $y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$

# Solving the Optimization Problem

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

$\Phi(\mathbf{w}) = \mathbf{w}^T \mathbf{w}$  is minimized

and for all  $(\mathbf{x}_i, y_i)$ ,  $i=1..n$  :  $y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$

- Need to optimize a *quadratic* function subject to *linear* constraints.
- Quadratic optimization problems are a well-known class of mathematical programming problems for which several (non-trivial) algorithms exist.
- The solution involves constructing a *dual problem* where a *Lagrange multiplier*  $\alpha_i$  is associated with every inequality constraint in the primal (original) problem:

Find  $\alpha_1 \dots \alpha_n$  such that

$Q(\boldsymbol{\alpha}) = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$  is maximized and

(1)  $\sum \alpha_i y_i = 0$

(2)  $\alpha_i \geq 0$  for all  $\alpha_i$



# The Optimization Problem Solution

- Given a solution  $\alpha_1 \dots \alpha_n$  to the dual problem, solution to the primal is:

$$\mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i \quad b = y_k - \sum \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_k \quad \text{for any } \alpha_k > 0$$

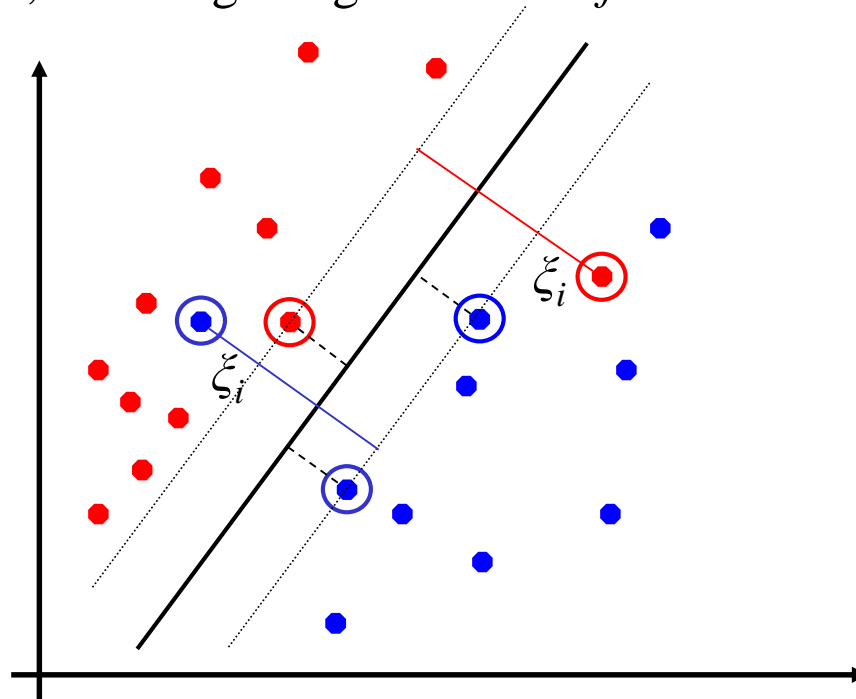
- Each non-zero  $\alpha_i$  indicates that corresponding  $\mathbf{x}_i$  is a support vector.
- Then the classifying function is (note that we don't need  $\mathbf{w}$  explicitly):

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

- Notice that it relies on an *inner product* between the test point  $\mathbf{x}$  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  $\mathbf{x}_i^T \mathbf{x}_j$  between all training points.

# Soft Margin Classification

- What if the training set is not linearly separable?
- *Slack variables*  $\xi_i$  can be added to allow misclassification of difficult or noisy examples, resulting margin called *soft*.



# Soft Margin Classification Mathematically

- The old formulation:

Find  $\mathbf{w}$  and  $b$  such that  
 $\Phi(\mathbf{w}) = \mathbf{w}^T \mathbf{w}$  is minimized  
and for all  $(\mathbf{x}_i, y_i), i=1..n$  :  $y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$

- Modified formulation incorporates slack variables:

Find  $\mathbf{w}$  and  $b$  such that  
 $\Phi(\mathbf{w}) = \mathbf{w}^T \mathbf{w} + C \sum \xi_i$  is minimized  
and for all  $(\mathbf{x}_i, y_i), i=1..n$  :  $y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$  ,  $\xi_i \geq 0$

- 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.

## Soft Margin Classification – Solution

- Dual problem is identical to separable case (would *not* be identical if the 2-norm penalty for slack variables  $C\sum \xi_i^2$  was used in primal objective, we would need additional Lagrange multipliers for slack variables):

Find  $\alpha_1 \dots \alpha_N$  such that

$Q(\boldsymbol{\alpha}) = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$  is maximized and

(1)  $\sum \alpha_i y_i = 0$

(2)  $0 \leq \alpha_i \leq C$  for all  $\alpha_i$

- Again,  $\mathbf{x}_i$  with non-zero  $\alpha_i$  will be support vectors.
- Solution to the dual problem is:

$$\mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i$$

$$b = y_k(1 - \xi_k) - \sum \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_k \quad \text{for any } k \text{ s.t. } \alpha_k > 0$$

Again, we don't need to compute  $\mathbf{w}$  explicitly for classification:

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

# Theoretical Justification for Maximum Margins

- Vapnik has proved the following:

*The class of optimal linear separators has VC dimension  $h$  bounded from above as*

$$h \leq \min \left\{ \left\lceil \frac{D^2}{\rho^2} \right\rceil, m_0 \right\} + 1$$

*where  $\rho$  is the margin,  $D$  is the diameter of the smallest sphere that can enclose all of the training examples, and  $m_0$  is the dimensionality.*

- Intuitively, this implies that regardless of dimensionality  $m_0$  we can minimize the VC dimension by maximizing the margin  $\rho$ .
- Thus, complexity of the classifier is kept small regardless of dimensionality.

# 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  $\alpha_i$ .
- Both in the dual formulation of the problem and in the solution training points appear only inside inner products:

Find  $\alpha_1 \dots \alpha_N$  such that

$\mathbf{Q}(\boldsymbol{\alpha}) = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$  is maximized and

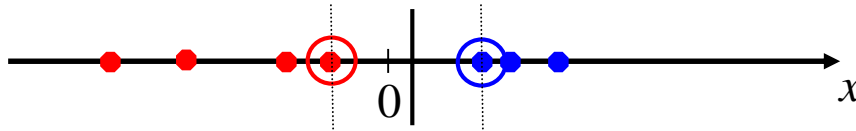
(1)  $\sum \alpha_i y_i = 0$

(2)  $0 \leq \alpha_i \leq C$  for all  $\alpha_i$

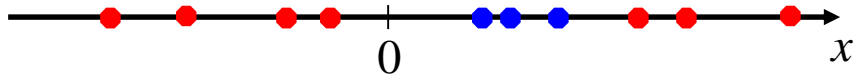
$$f(\mathbf{x}) = \sum \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

# Non-linear SVMs

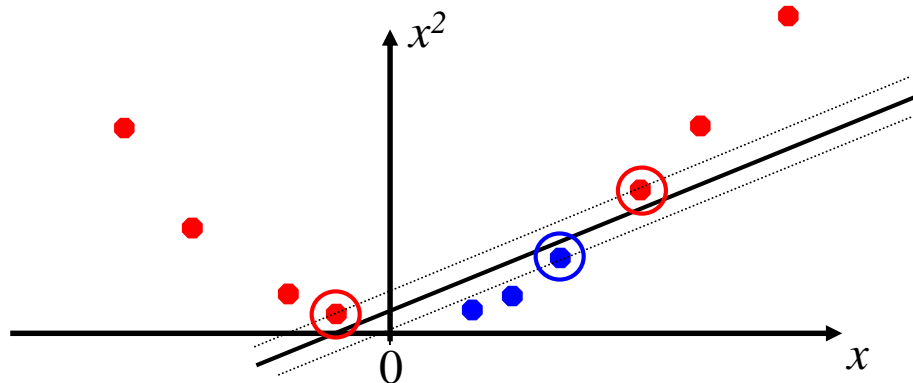
- Datasets that are linearly separable with some noise work out great:



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

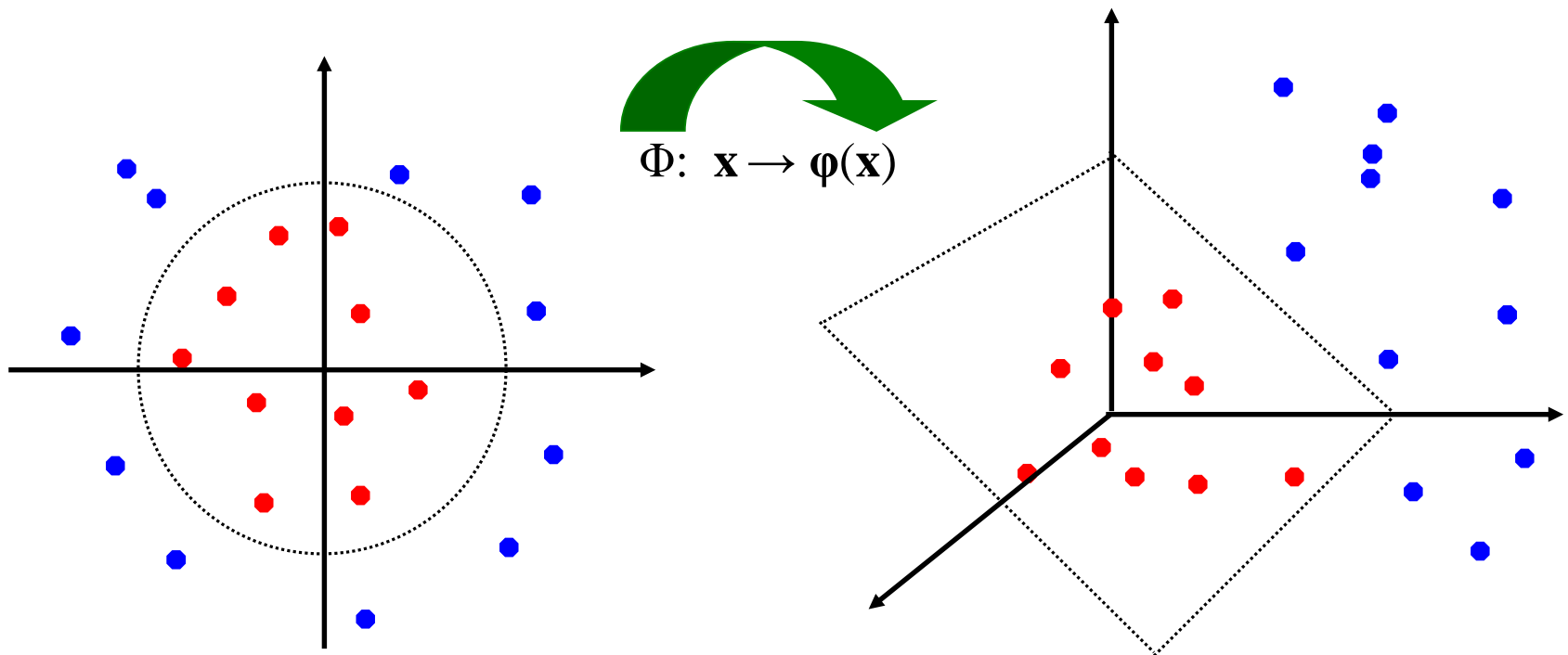


- How about... mapping data to a higher-dimensional space:



# Non-linear SVMs: Feature spaces

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





# The “Kernel Trick”

- The linear classifier relies on inner product between vectors  $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- If every datapoint is mapped into high-dimensional space via some transformation  $\Phi: \mathbf{x} \rightarrow \phi(\mathbf{x})$ , the inner product becomes:

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

- A *kernel function* is a function that is equivalent to an inner product in some feature space.
- 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  $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ :

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

- Thus, a kernel function *implicitly* maps data to a high-dimensional space (without the need to compute each  $\phi(\mathbf{x})$  explicitly).

# What Functions are Kernels?

- For some functions  $K(\mathbf{x}_i, \mathbf{x}_j)$  checking that  $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$  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:  $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$ 
  - Mapping  $\Phi: \mathbf{x} \rightarrow \boldsymbol{\phi}(\mathbf{x})$ , where  $\boldsymbol{\phi}(\mathbf{x})$  is  $\mathbf{x}$  itself
- Polynomial of power  $p$ :  $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$ 
  - Mapping  $\Phi: \mathbf{x} \rightarrow \boldsymbol{\phi}(\mathbf{x})$ , where  $\boldsymbol{\phi}(\mathbf{x})$  has  $\binom{d+p}{p}$  dimensions
- Gaussian (radial-basis function):  $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}$ 
  - Mapping  $\Phi: \mathbf{x} \rightarrow \boldsymbol{\phi}(\mathbf{x})$ , where  $\boldsymbol{\phi}(\mathbf{x})$  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  $d$  (the mapping is not *onto*), but linear separators in it correspond to *non-linear* separators in original space.

# Non-linear SVMs Mathematically

- Dual problem formulation:

Find  $\alpha_1 \dots \alpha_n$  such that

$Q(\alpha) = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$  is maximized and

(1)  $\sum \alpha_i y_i = 0$

(2)  $\alpha_i \geq 0$  for all  $\alpha_i$

- The solution is:

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

- Optimization techniques for finding  $\alpha_i$ 's remain the same!

## SVM applications

- SVMs were originally proposed by Boser, Guyon and Vapnik in 1992 and gained increasing popularity in late 1990s.
- SVMs are currently among the best performers for a number of classification tasks ranging from text to genomic data.
- 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  $\alpha_i$ 's at a time, e.g. SMO [Platt '99] and [Joachims '99]
- Tuning SVMs remains a black art: selecting a specific kernel and parameters is usually done in a try-and-see manner.