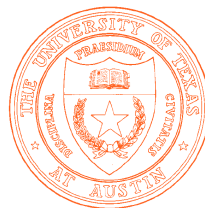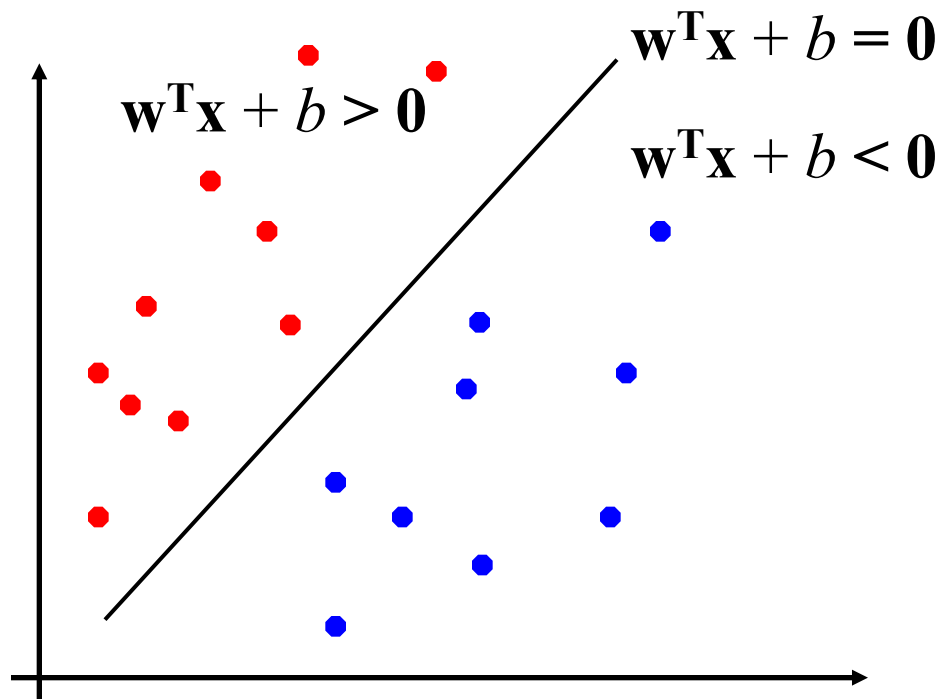# Support Vector Machines

Machine Learning Group

Department of Computer Sciences

University of Texas at Austin

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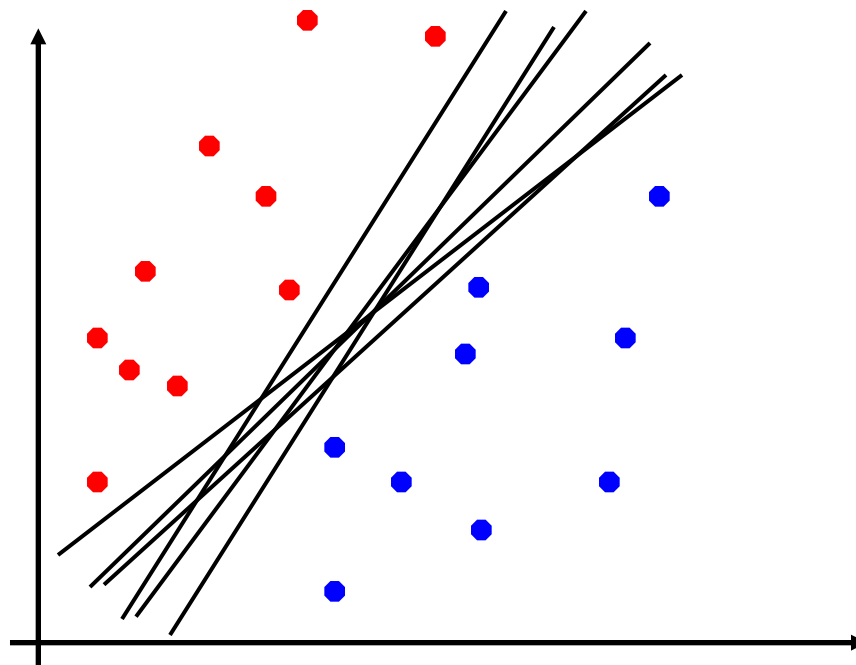
- Binary classification can be viewed as the task of separating classes in feature space:

$$\mathbf{w^T x} + b = 0$$

$$\mathbf{w^T x} + b > \mathbf{0}$$

$$\mathbf{w^T x} + b < \mathbf{0}$$

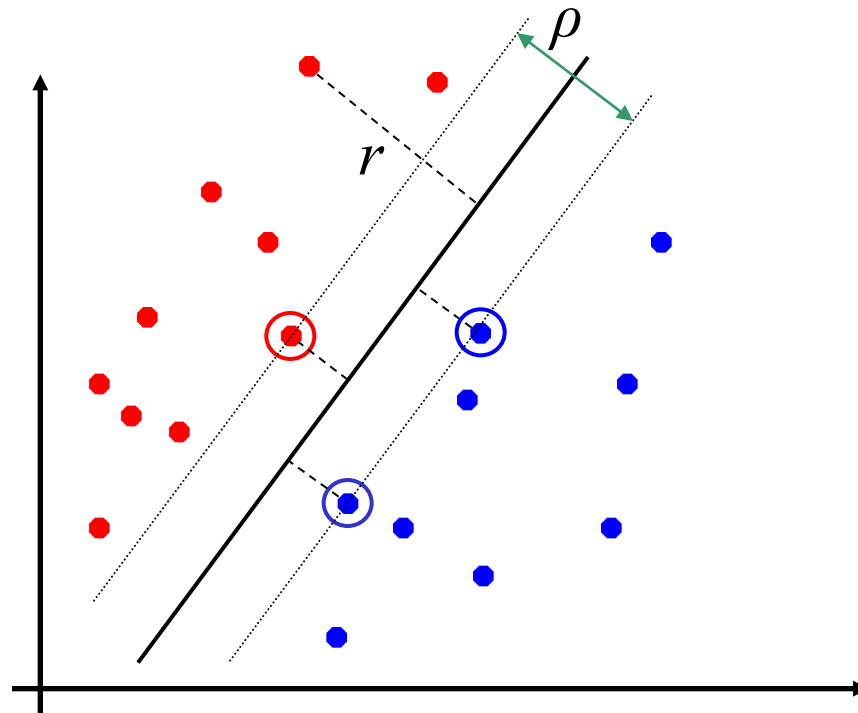$$f(\mathbf{x}) = \text{sign}(\mathbf{w^T x} + b)$$

Linear Separators

- Which of the linear separators is optimal?

Classification Margin

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

# 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.
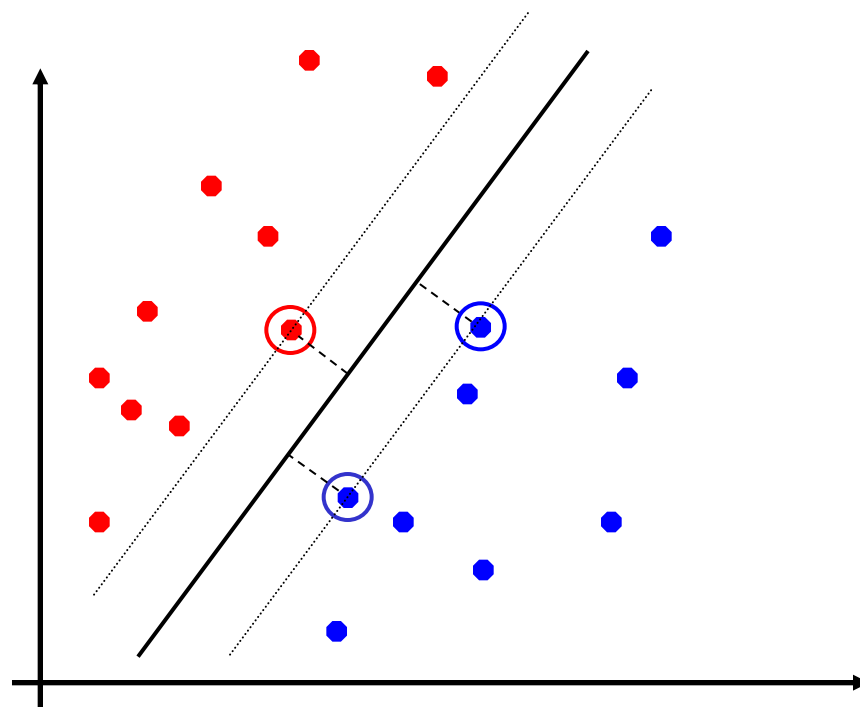
Soft Margin Classification

- What if the training set is not linearly separable?
- *Slack variables* ξ_i can be added to allow misclassification of difficult or noisy examples, resulting margin called *soft*.
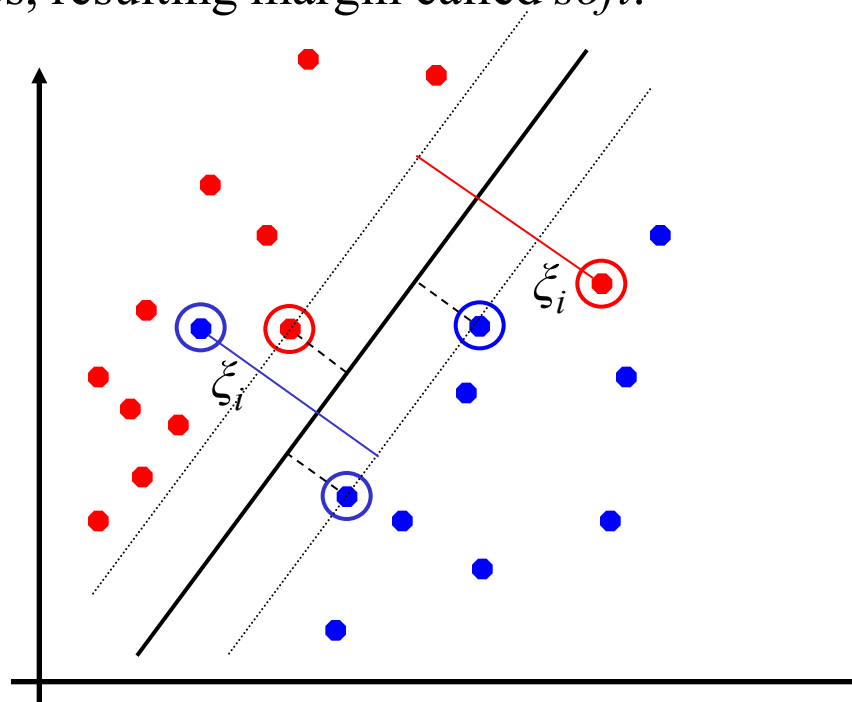
Linear SVMs: Overview

- 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 α_i.

- Both in the dual formulation of the problem and in the solution training points appear only inside inner products:

Find $\alpha_1...\alpha_N$ such that
$\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x}_i^{\mathbf{T}}\mathbf{x}_j$ is maximized and
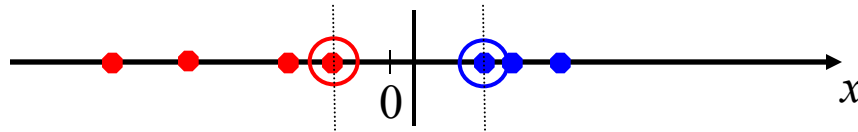(1) $\Sigma\alpha_i y_i = 0$
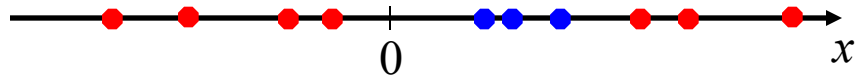(2) $0 \le \alpha_i \le C$ for all α_i

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

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?
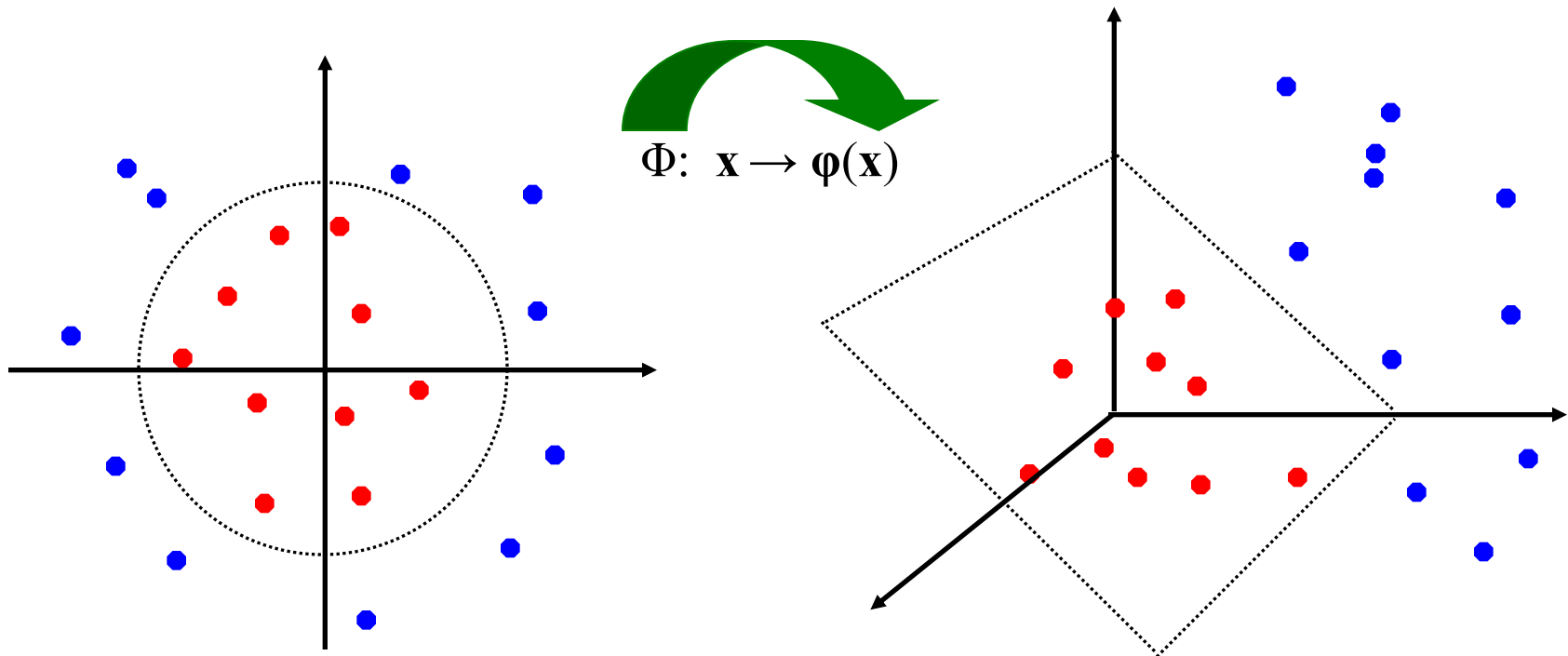

- How about… mapping data to a higher-dimensional space:
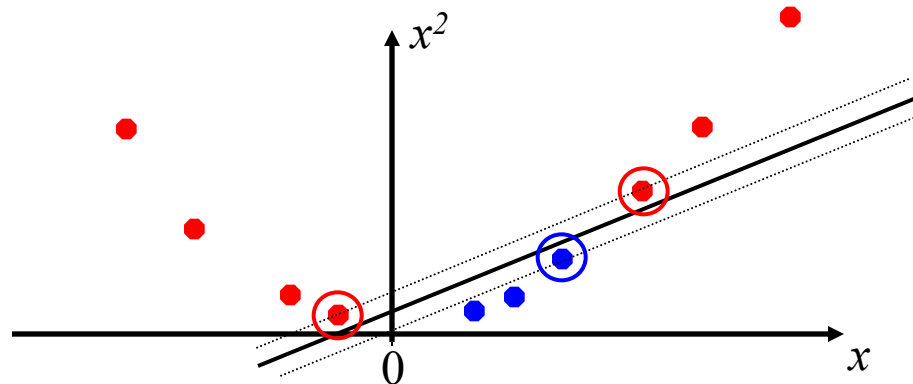
Non-linear SVMs: Feature spaces

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

$$\Phi:\ \mathbf{x} \rightarrow \boldsymbol{\varphi}(\mathbf{x})$$

The "Kernel Trick"

- The linear classifier relies on inner product between vectors $K(\mathbf{x}_i,\mathbf{x}_j)=\mathbf{x}_i^{\mathbf{T}}\mathbf{x}_j$
- If every datapoint is mapped into high-dimensional space via some transformation $\Phi: \mathbf{x} \rightarrow \varphi(\mathbf{x})$, the inner product becomes:
$$K(\mathbf{x}_i,\mathbf{x}_j)= \varphi(\mathbf{x}_i)^{\mathbf{T}}\varphi(\mathbf{x}_j)$$
- A *kernel function* is a function that is eqiuvalent to an inner product in some feature space.
- Example:

  2-dimensional vectors $\mathbf{x}=[x_1 \ x_2]$; let $K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^{\mathbf{T}}\mathbf{x}_j)^2$,

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

  $K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^{\mathbf{T}}\mathbf{x}_j)^2 = 1+ x_{i1}^2x_{j1}^2 + 2\ x_{i1}x_{j1}\ x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}=$
  $= [1 \ \ x_{i1}^2 \ \sqrt 2\ x_{i1}x_{i2} \ \ x_{i2}^2 \ \sqrt 2 x_{i1} \ \sqrt 2 x_{i2}]^{\mathbf{T}} [1 \ \ x_{j1}^2 \ \sqrt 2\ x_{j1}x_{j2} \ \ x_{j2}^2 \ \sqrt 2 x_{j1} \ \sqrt 2 x_{j2}] =$
  $= \varphi(\mathbf{x}_i)^{\mathbf{T}}\varphi(\mathbf{x}_j), \quad$ where $\varphi(\mathbf{x}) = [1 \ \ x_1^2 \ \sqrt 2\ x_1x_2 \ \ x_2^2 \ \sqrt 2 x_1 \ \sqrt 2 x_2]$

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

Examples of Kernel Functions

- Linear: $K(\mathbf{x}_i,\mathbf{x}_j)= \mathbf{x}_i^\mathbf{T}\mathbf{x}_j$
  - Mapping Φ: $\mathbf{x} \to \boldsymbol{\varphi}(\mathbf{x})$, where $\boldsymbol{\varphi}(\mathbf{x})$ is $\mathbf{x}$ itself

- Polynomial of power p: $K(\mathbf{x}_i,\mathbf{x}_j)= (1+ \mathbf{x}_i^\mathbf{T}\mathbf{x}_j)^p$
  - Mapping Φ: $\mathbf{x} \to \boldsymbol{\varphi}(\mathbf{x})$, where $\boldsymbol{\varphi}(\mathbf{x})$ has $\binom{d+p}{p}$ dimensions

- Gaussian (radial-basis function): $K(\mathbf{x}_i,\mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \,_j\|^2}{2\sigma^{\;!}}}$
  - Mapping Φ: $\mathbf{x} \to \boldsymbol{\varphi}(\mathbf{x})$, where $\boldsymbol{\varphi}(\mathbf{x})$ 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.

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 α_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.