

Artificial Intelligence for Medicine II

Spring 2025

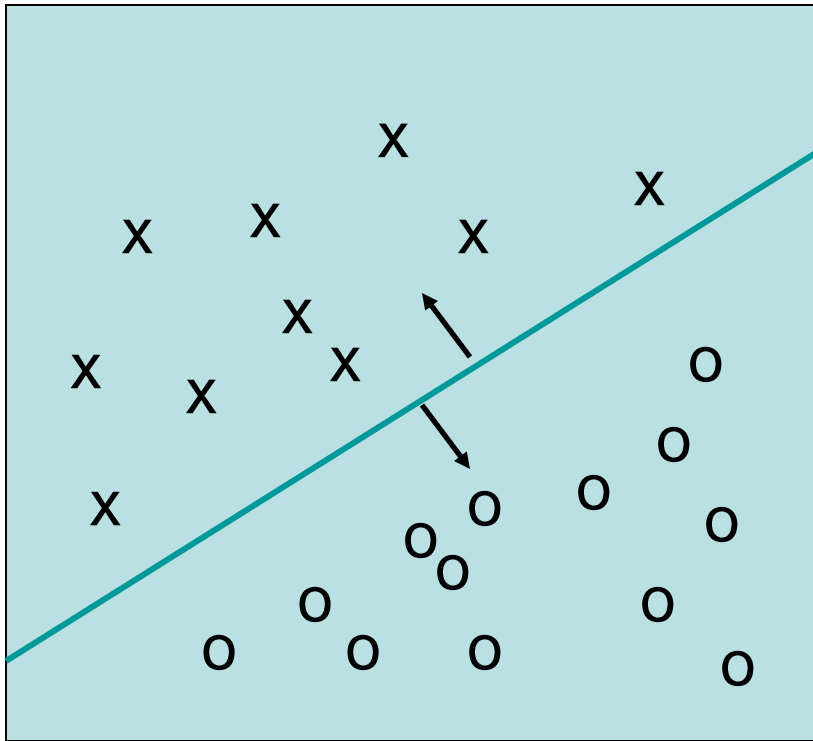
Lecture 61: Supervised Learning Support Vector Machines (SVM)

(Many slides adapted from Bing Liu, Han, Kamber & Pei; Tan, Steinbach, Kumar and the web)

Classification: A Mathematical Mapping

- Classification:
 - predicts categorical class labels
- E.g., Personal homepage classification
 - $x_i = (x_1, x_2, x_3, \dots)$, $y_i = +1$ or -1
 - x_1 : # of a word “homepage”
 - x_2 : # of a word “welcome”
- Mathematically
 - $x \in X = \mathbb{R}^n$, $y \in Y = \{+1, -1\}$
 - We want a function $f: X \rightarrow Y$

Linear Classification

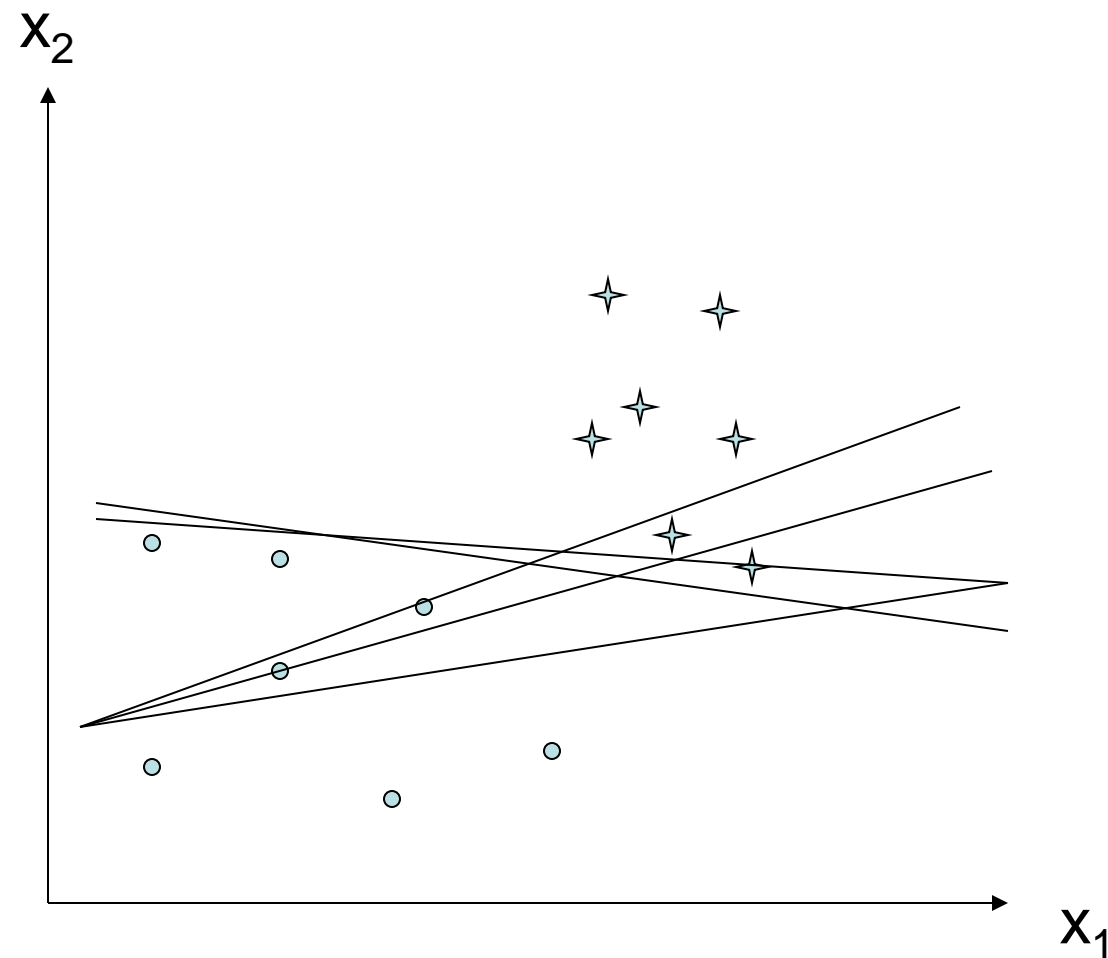


- Binary Classification problem
- The data above the red line belongs to class 'x'
- The data below red line belongs to class 'o'
- Examples: SVM, Perceptron, Probabilistic Classifiers

Discriminative Classifiers

- Advantages
 - prediction accuracy is generally high
 - As compared to Bayesian methods – in general
 - robust, works when training examples contain errors
 - fast evaluation of the learned target function
 - Bayesian networks are normally slow
- Criticism
 - long training time
 - difficult to understand the learned function (weights)
 - Bayesian networks can be used easily for pattern discovery
 - not easy to incorporate domain knowledge
 - Easy in the form of priors on the data or distributions

NN Perceptron & Winnow



- Vector: x, w

- Scalar: x, y, w

Input: $\{(x_1, y_1), \dots\}$

Output: classification function $f(x)$

$f(x_i) > 0$ for $y_i = +1$

$f(x_i) < 0$ for $y_i = -1$

$f(x) \Rightarrow wx + b = 0$

or $w_1x_1 + w_2x_2 + b = 0$

- Perceptron: update W additively

- Winnow: update W multiplicatively

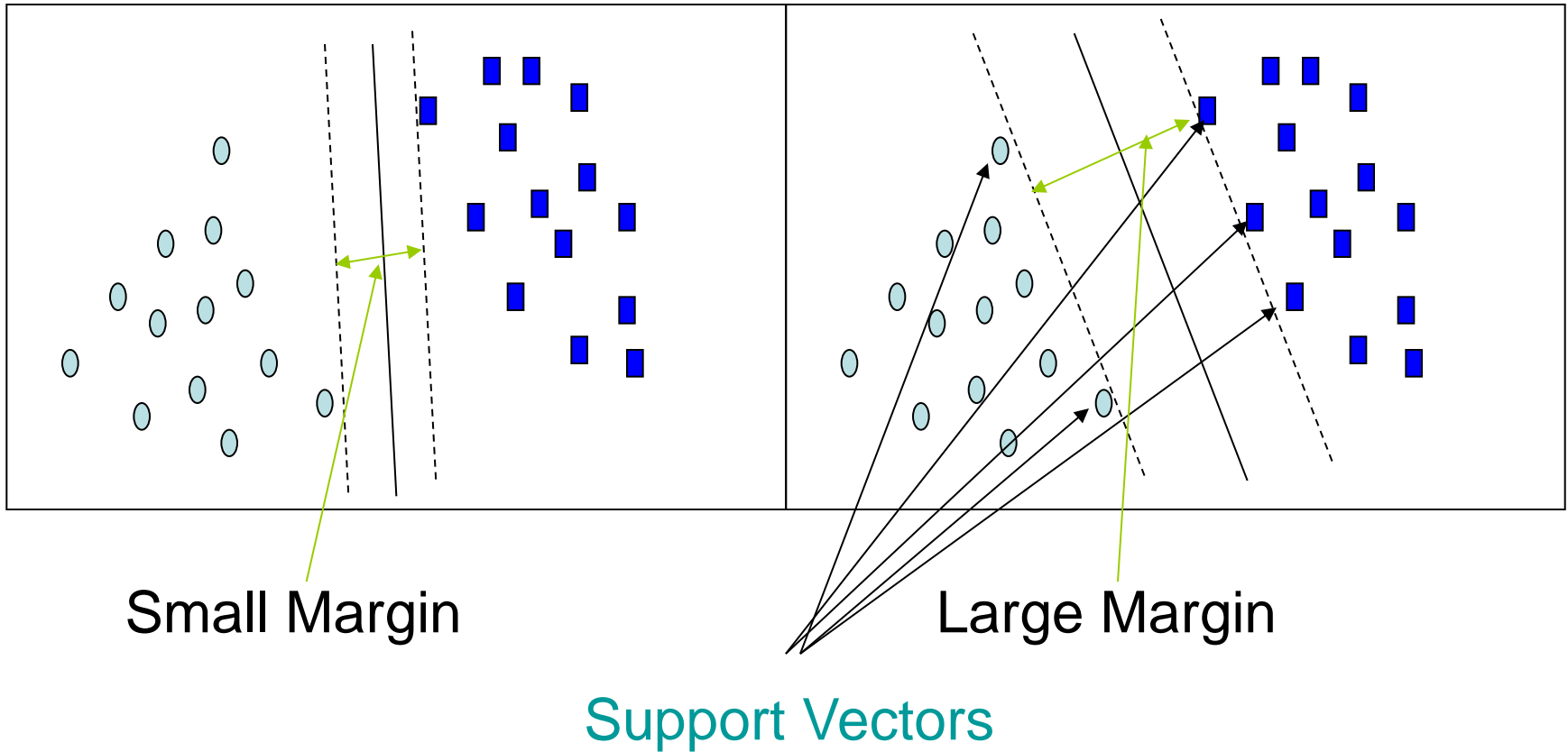
SVM—Support Vector Machines

- A new classification method for both linear and nonlinear data
- It uses a nonlinear mapping to transform the original training data into a higher dimension
- With the new dimension, it searches for the linear optimal separating hyperplane (i.e., “decision boundary”)
- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM finds this hyperplane using support vectors (“essential” training tuples) and margins (defined by the support vectors)

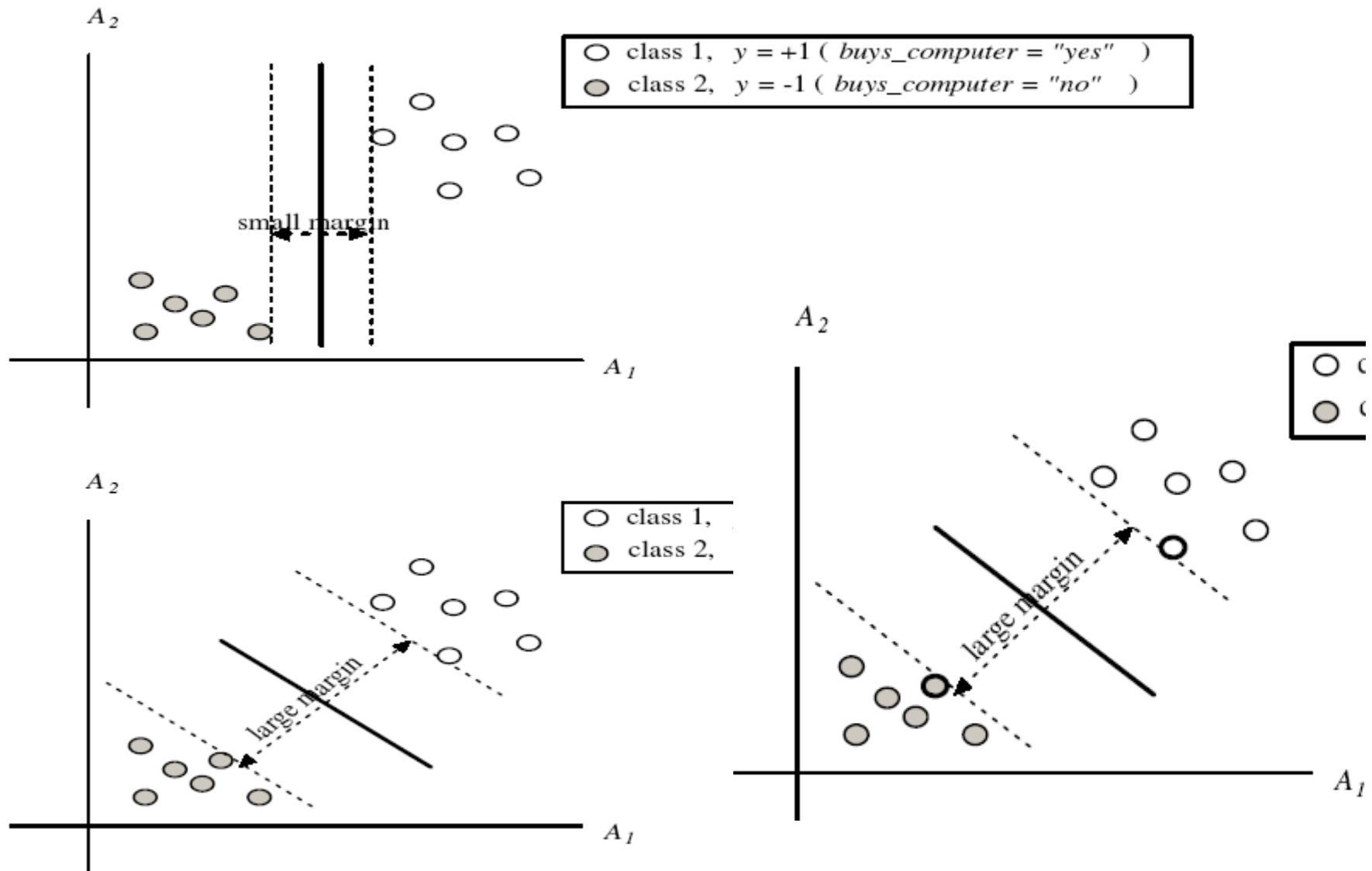
SVM—History and Applications

- Vapnik and colleagues (1992)—groundwork from Vapnik & Chervonenkis' statistical learning theory in 1960s
- Features: training can be slow but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)
- Used both for classification and prediction
- Applications:
 - handwritten digit recognition, object recognition, speaker identification, benchmarking time-series prediction tests

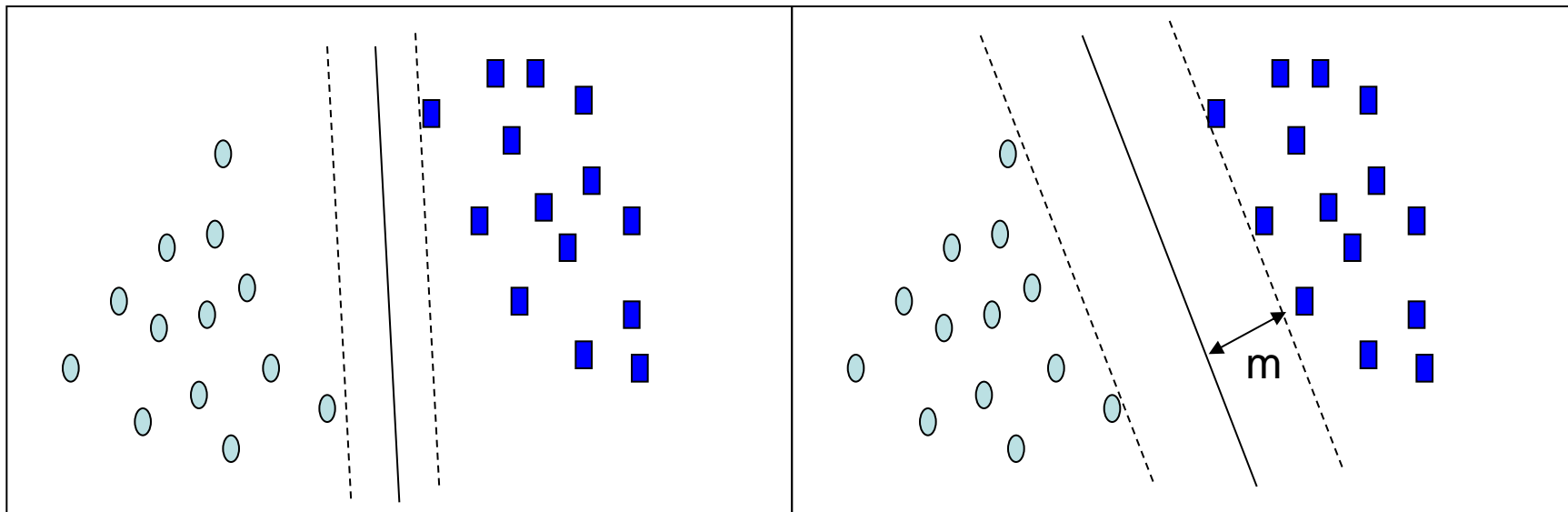
SVM—General Philosophy



SVM—Margins and Support Vectors



SVM—When Data Is Linearly Separable



Let data D be $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_{|D|}, y_{|D|})$, where \mathbf{X}_i is the set of training tuples associated with the class labels y_i

There are infinite lines (hyperplanes) separating the two classes but we want to find the best one (the one that minimizes classification error on unseen data)

SVM searches for the hyperplane with the largest margin, i.e., **maximum marginal hyperplane** (MMH)

SVM—Linearly Separable

- A separating hyperplane can be written as

$$\mathbf{W} \bullet \mathbf{X} + b = 0$$

where $\mathbf{W} = \{w_1, w_2, \dots, w_n\}$ is a weight vector and b a scalar (bias)

- For 2-D it can be written as

$$w_0 + w_1 x_1 + w_2 x_2 = 0$$

- The hyperplane defining the sides of the margin:

$$H_1: w_0 + w_1 x_1 + w_2 x_2 \geq 1 \quad \text{for } y_i = +1, \text{ and}$$

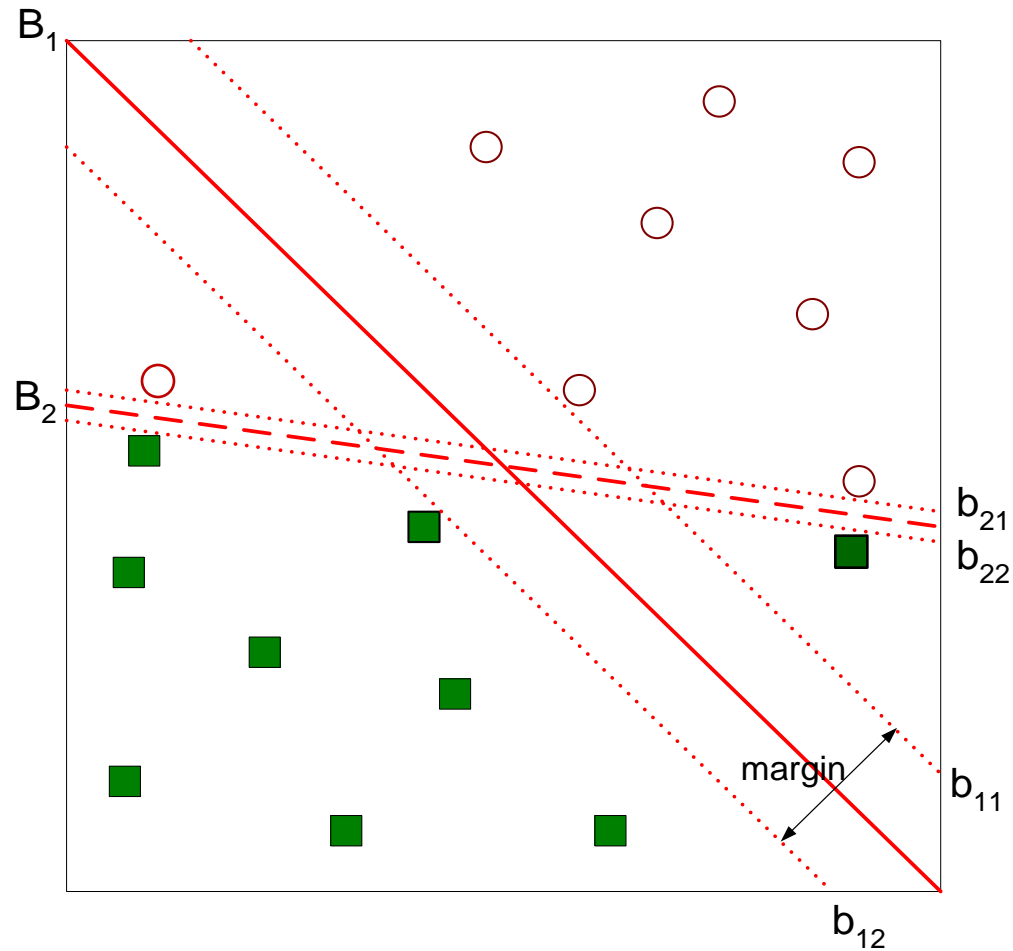
$$H_2: w_0 + w_1 x_1 + w_2 x_2 \leq -1 \quad \text{for } y_i = -1$$

- Any training tuples that fall on hyperplanes H_1 or H_2 (i.e., the sides defining the margin) are **support vectors**
- This becomes a **constrained (convex) quadratic optimization** problem: Quadratic objective function and linear constraints \rightarrow *Quadratic Programming (QP)* \rightarrow Lagrangian multipliers

Why Is SVM Effective on High Dimensional Data?

- The complexity of trained classifier is characterized by the # of support vectors rather than the dimensionality of the data
- The support vectors are the essential or critical training examples — they lie closest to the decision boundary (MMH)
- If all other training examples are removed and the training is repeated, the same separating hyperplane would be found
- The number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality
- Thus, an SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high

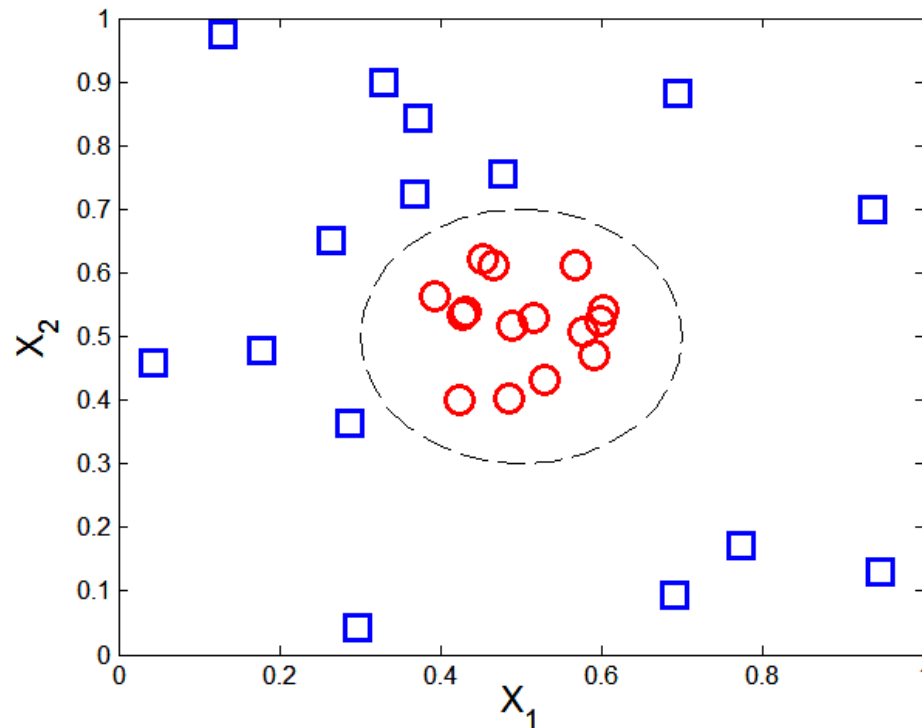
Support Vector Machines



- Find the hyperplane that optimizes both factors

Nonlinear Support Vector Machines

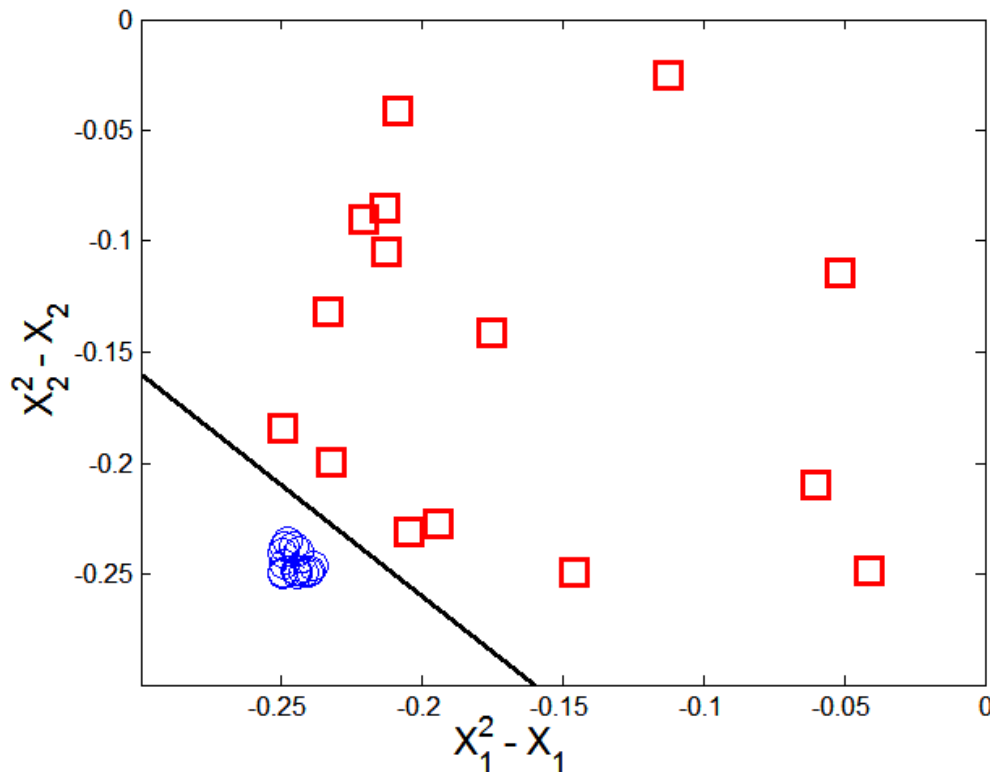
- What if decision boundary is not linear?



$$y(x_1, x_2) = \begin{cases} 1 & \text{if } \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2 \\ -1 & \text{otherwise} \end{cases}$$

Nonlinear Support Vector Machines

- Transform data into higher dimensional space



$$x_1^2 - x_1 + x_2^2 - x_2 = -0.46.$$

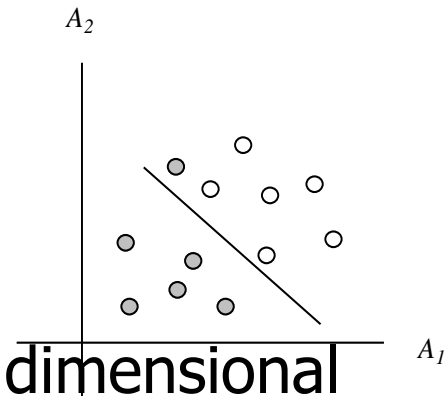
$$\Phi : (x_1, x_2) \longrightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1).$$

$$w_4x_1^2 + w_3x_2^2 + w_2\sqrt{2}x_1 + w_1\sqrt{2}x_2 + w_0 = 0.$$

Decision boundary:

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

SVM—Linearly Inseparable



- Transform the original input data into a higher dimensional space

Example 6.8 Nonlinear transformation of original input data into a higher dimensional space. Consider the following example. A 3D input vector $\mathbf{X} = (x_1, x_2, x_3)$ is mapped into a 6D space Z using the mappings $\phi_1(\mathbf{X}) = x_1, \phi_2(\mathbf{X}) = x_2, \phi_3(\mathbf{X}) = x_3, \phi_4(\mathbf{X}) = (x_1)^2, \phi_5(\mathbf{X}) = x_1x_2$, and $\phi_6(\mathbf{X}) = x_1x_3$. A decision hyperplane in the new space is $d(\mathbf{Z}) = \mathbf{WZ} + b$, where \mathbf{W} and \mathbf{Z} are vectors. This is linear. We solve for \mathbf{W} and b and then substitute back so that we see that the linear decision hyperplane in the new (\mathbf{Z}) space corresponds to a nonlinear second order polynomial in the original 3-D input space,

$$\begin{aligned} d(\mathbf{Z}) &= w_1x_1 + w_2x_2 + w_3x_3 + w_4(x_1)^2 + w_5x_1x_2 + w_6x_1x_3 + b \\ &= w_1z_1 + w_2z_2 + w_3z_3 + w_4z_4 + w_5z_5 + w_6z_6 + b \end{aligned} \quad \blacksquare$$

- Search for a linear separating hyperplane in the new space

SVM—Kernel functions

- Instead of computing the dot product on the transformed data tuples, it is mathematically equivalent to instead applying a kernel function $K(\mathbf{X}_i, \mathbf{X}_j)$ to the original data, i.e., $K(\mathbf{X}_i, \mathbf{X}_j) = \Phi(\mathbf{X}_i) \cdot \Phi(\mathbf{X}_j)$
- Typical Kernel Functions

Polynomial kernel of degree h : $K(X_i, X_j) = (X_i \cdot X_j + 1)^h$

Gaussian radial basis function kernel : $K(X_i, X_j) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$

Sigmoid kernel : $K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta)$

- SVM can also be used for classifying multiple (> 2) classes and for regression analysis (with additional user parameters)

Learning Nonlinear SVM

- Advantages of using kernel:
 - Don't have to know the mapping function Φ
 - Computing dot product $\Phi(x_i) \bullet \Phi(x_j)$ in the original space avoids curse of dimensionality
- Not all functions can be kernels
 - Must make sure there is a corresponding Φ in some high-dimensional space
 - Mercer's theorem

Characteristics of SVM

- The learning problem is formulated as a convex optimization problem
 - Efficient algorithms are available to find the global minima
 - Many of the other methods use greedy approaches and find locally optimal solutions
 - High computational complexity for building the model
- Robust to noise
- Overfitting is handled by maximizing the margin of the decision boundary,
- SVM can handle irrelevant and redundant attributes better than many other techniques
- The user needs to provide the type of kernel function and cost function
- Difficult to handle missing values

SVM vs. Neural Network

- SVM
 - Relatively new concept
 - Deterministic algorithm
 - Nice Generalization properties
 - Hard to learn – learned in batch mode using quadratic programming techniques
 - Using kernels can learn very complex functions
- Neural Network
 - Relatively old
 - Nondeterministic algorithm
 - Generalizes well but doesn't have strong mathematical foundation
 - Can easily be learned in incremental fashion
 - To learn complex functions—use multilayer perceptron (not that trivial)