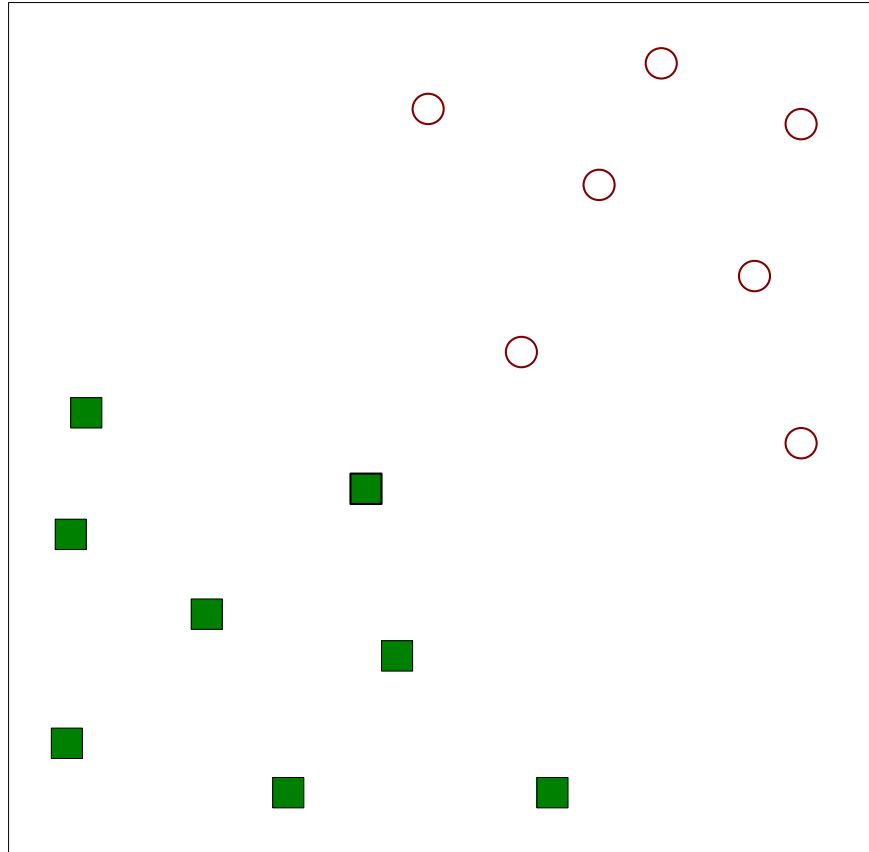


Data Mining

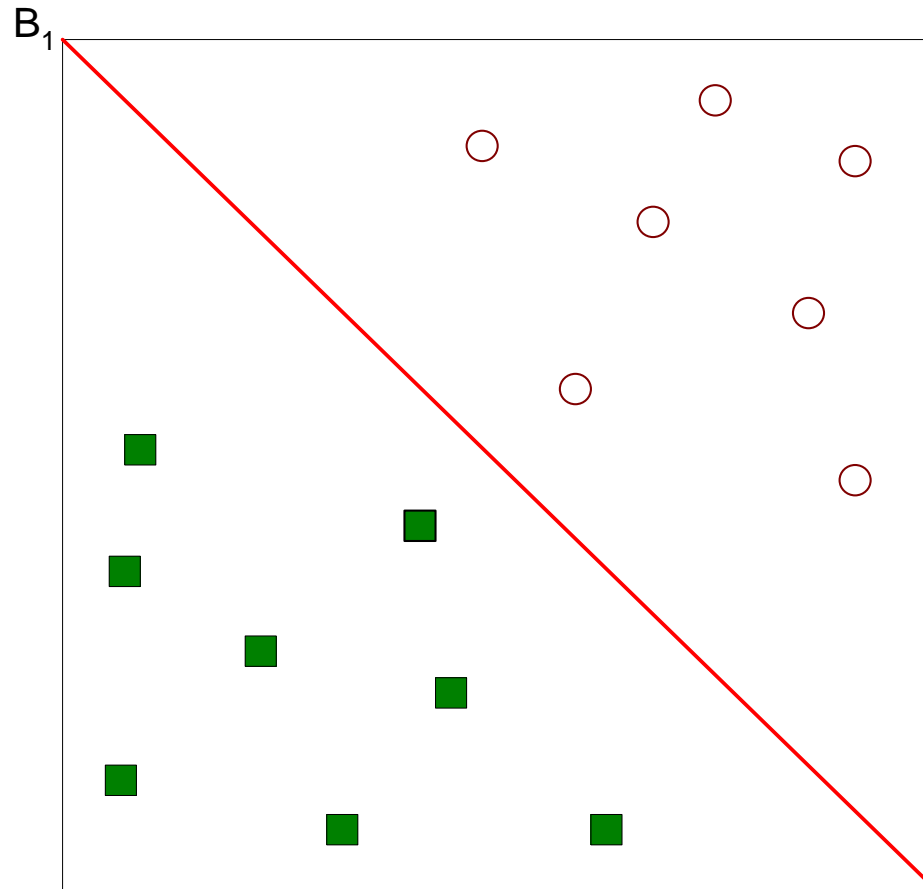
Support Vector Machines

Support Vector Machines



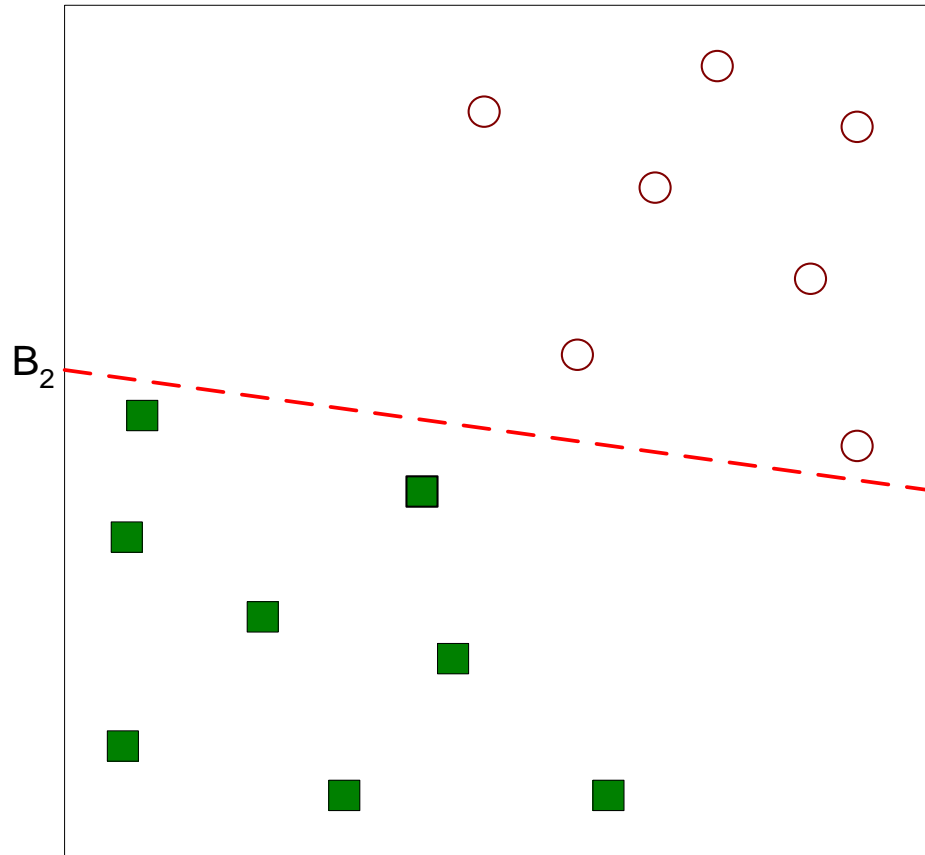
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



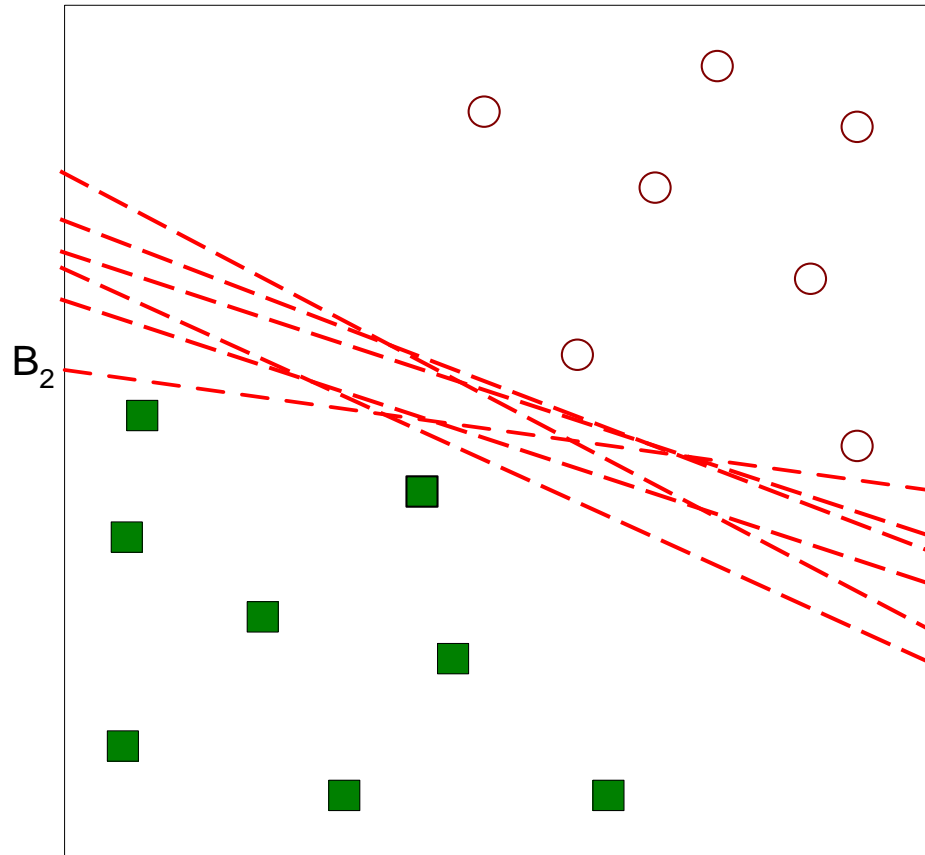
- One Possible Solution

Support Vector Machines



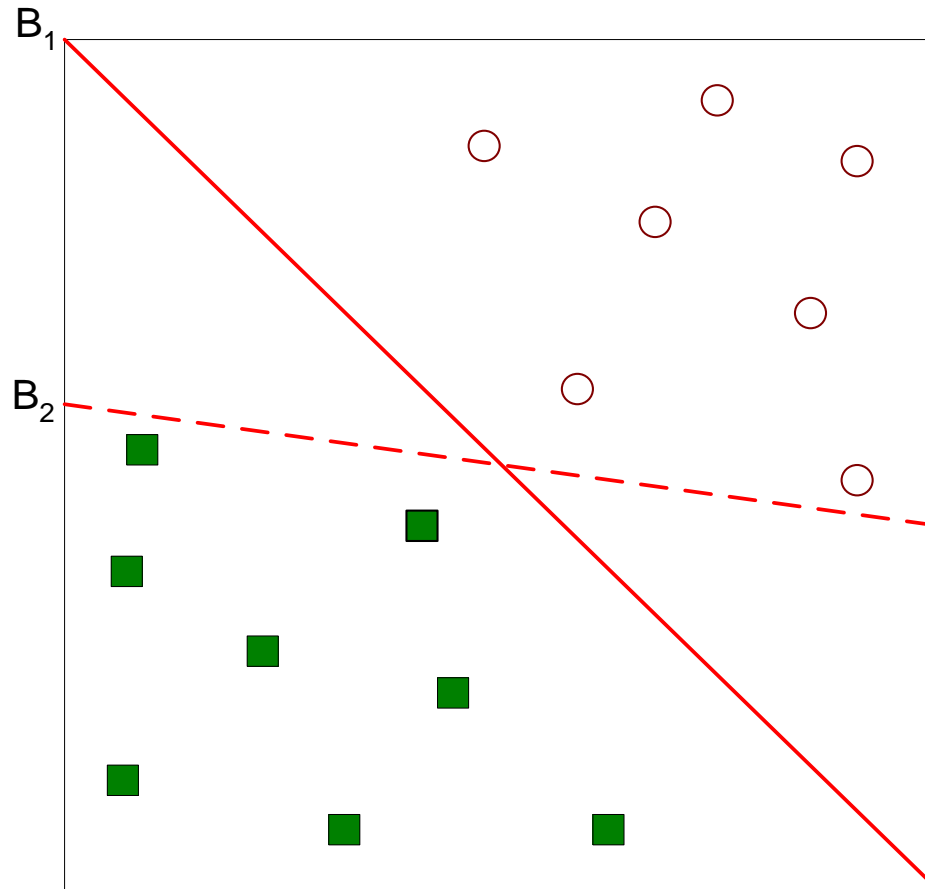
- Another possible solution

Support Vector Machines



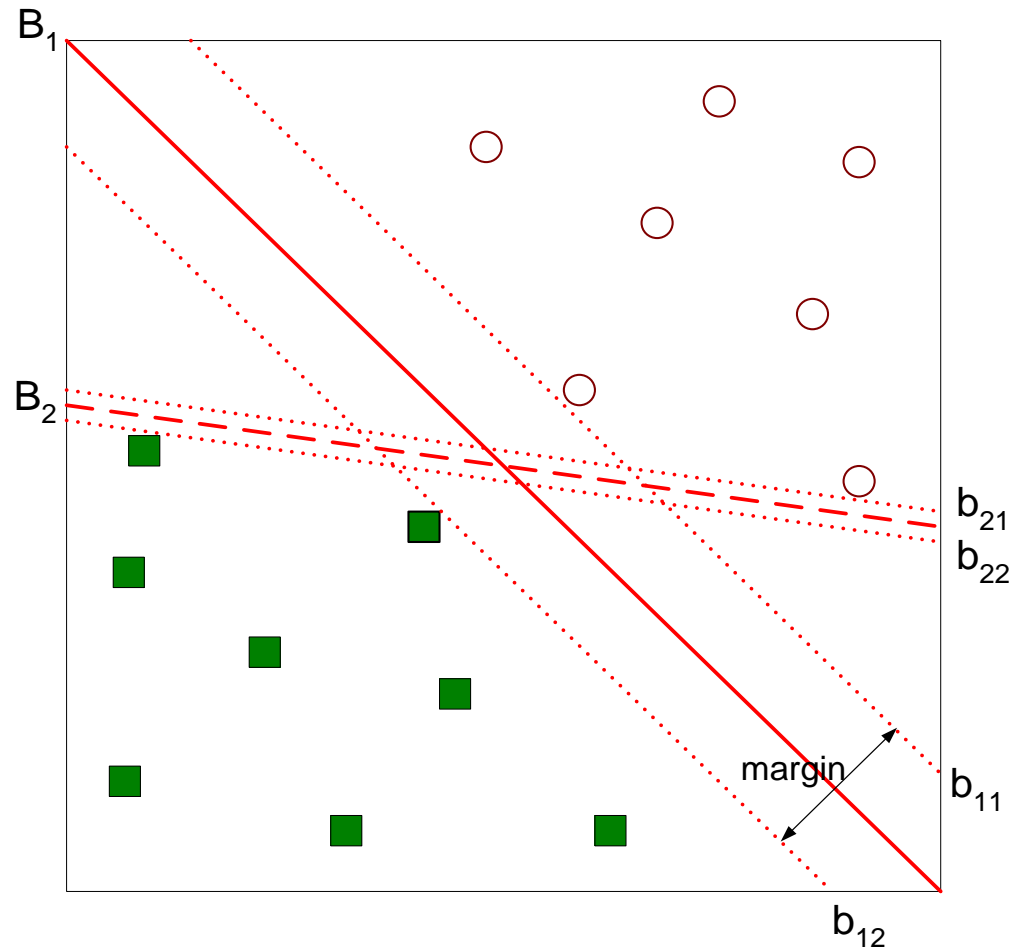
- Other possible solutions

Support Vector Machines



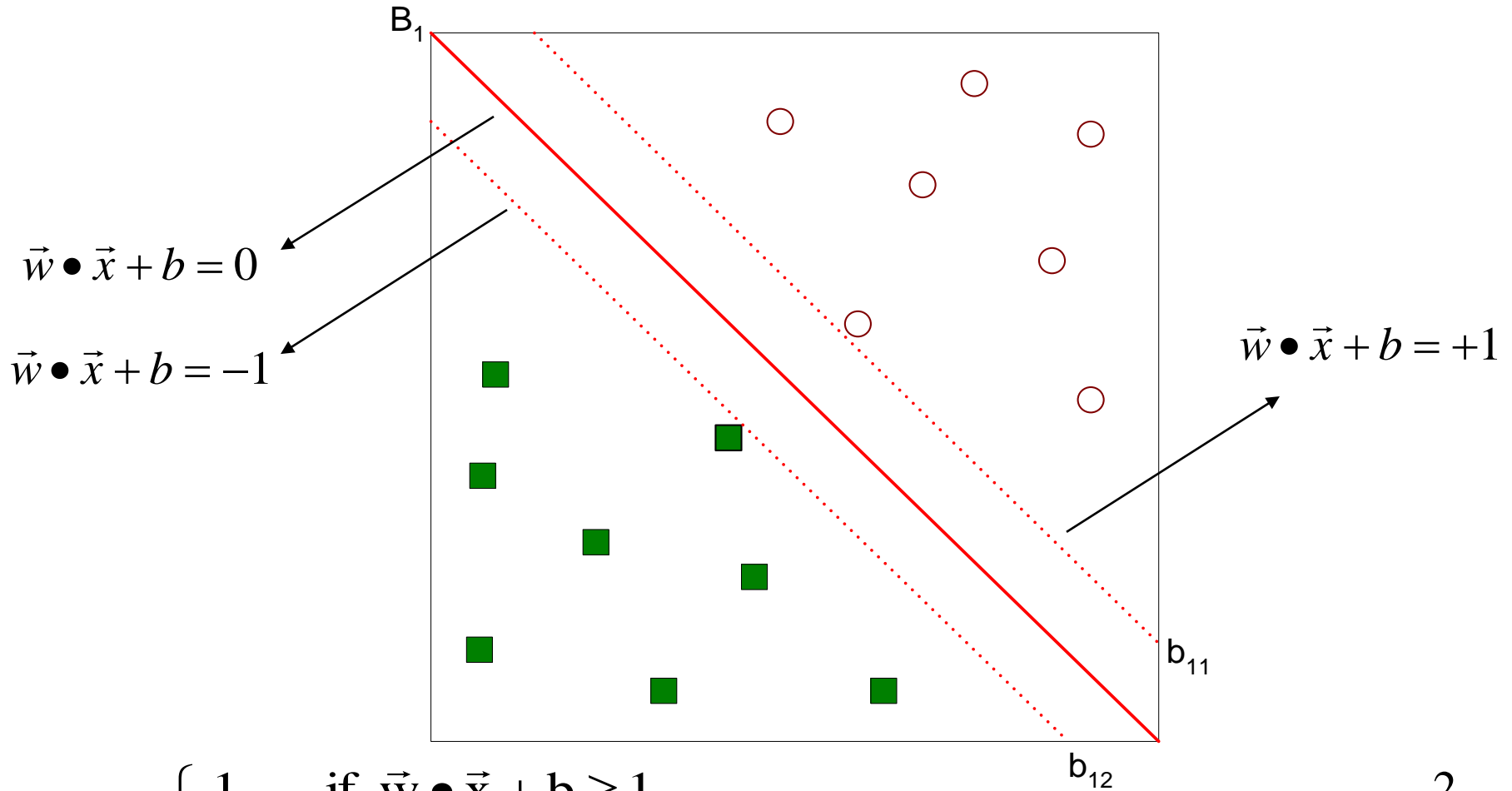
- Which one is better? B_1 or B_2 ?
- How do you define better?

Support Vector Machines



- Find hyperplane **maximizes** the margin \Rightarrow B1 is better than B2

Support Vector Machines



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

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

Linear SVM

- Linear model:

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

- Learning the model is equivalent to determining the values of \vec{w} and b
 - How to find \vec{w} and b from training data?

Learning Linear SVM

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

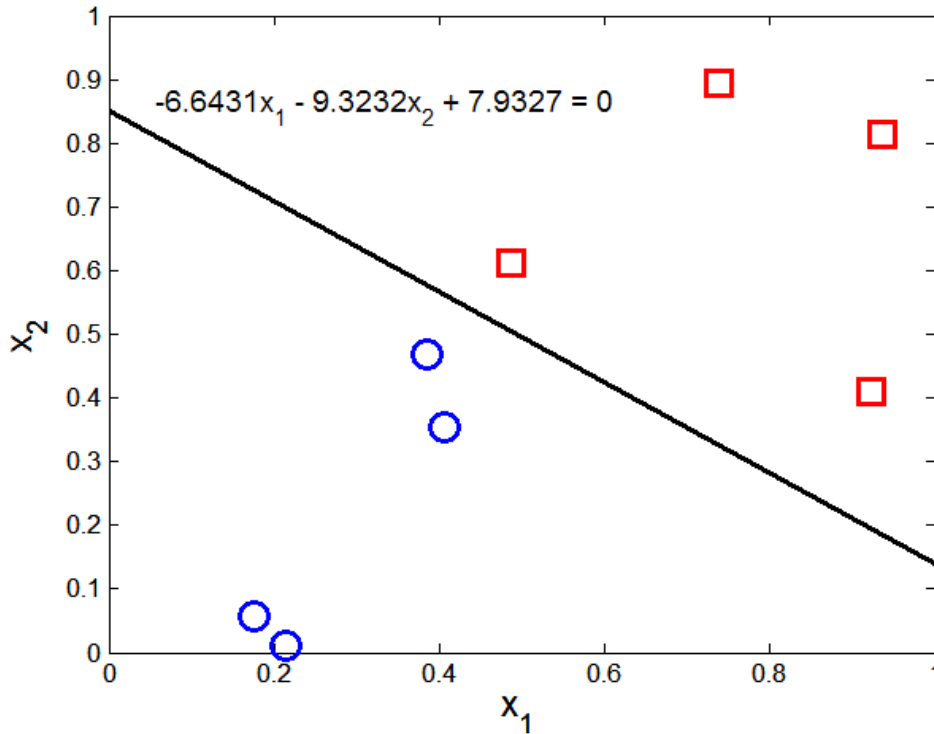
$$y_i = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 \end{cases}$$

or

$$y_i(w \bullet x_i + b) \geq 1, \quad i = 1, 2, \dots, N$$

- ◆ This is a constrained optimization problem
 - Solve it using Lagrange multiplier method

Example of Linear SVM



Support vectors

x1	x2	y	λ
0.3858	0.4687	1	65.5261
0.4871	0.611	-1	65.5261
0.9218	0.4103	-1	0
0.7382	0.8936	-1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	-1	0
0.2146	0.0099	1	0

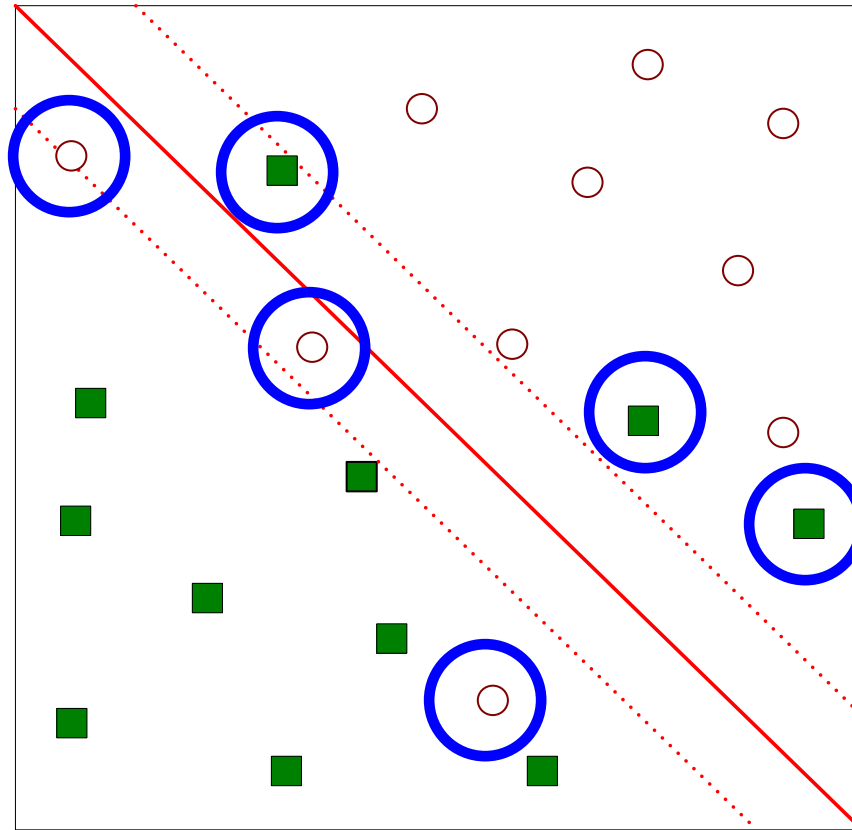
Learning Linear SVM

- Decision boundary depends only on support vectors
 - If you have data set with same support vectors, decision boundary will not change
 - How to classify using SVM once \mathbf{w} and b are found? Given a test record, \mathbf{x}_i

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

Support Vector Machines

- What if the problem is not linearly separable?



Support Vector Machines

- What if the problem is not linearly separable?
 - Introduce slack variables

- ◆ Need to minimize:

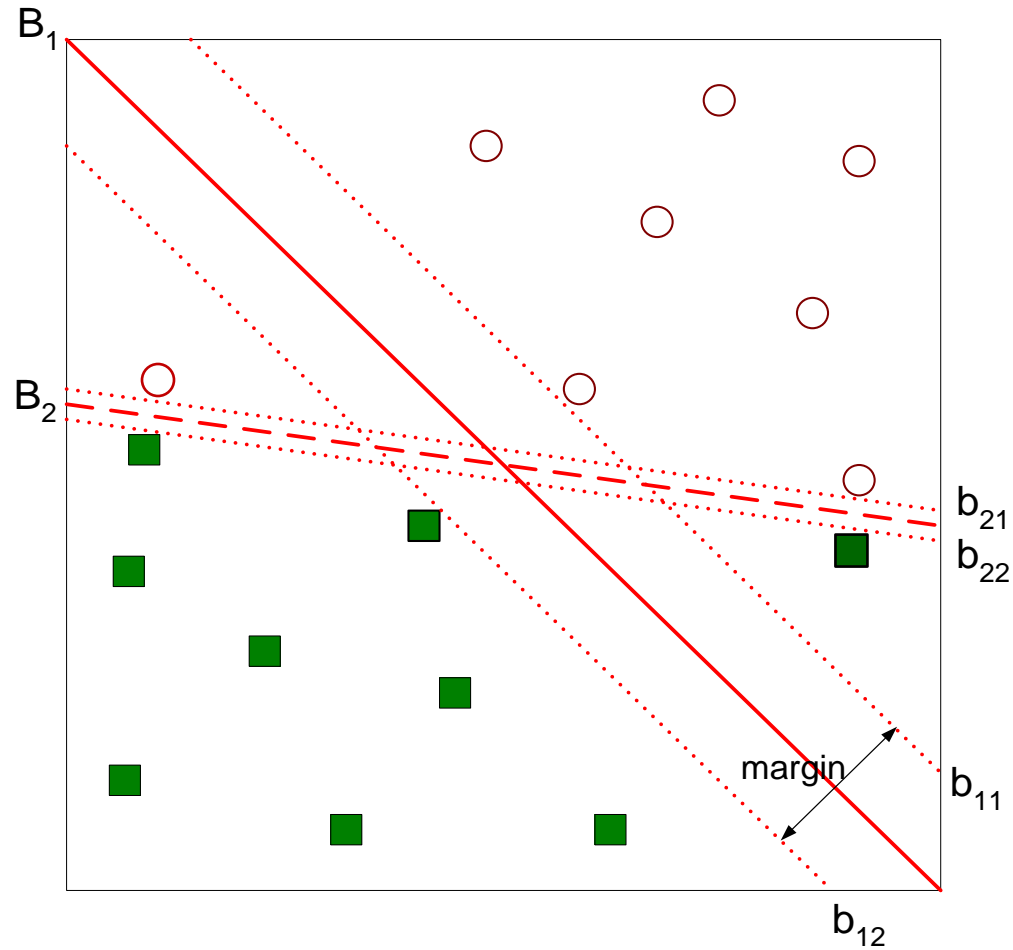
$$L(w) = \frac{\|\vec{w}\|^2}{2} + C \left(\sum_{i=1}^N \xi_i^k \right)$$

- ◆ Subject to:

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

- ◆ If k is 1 or 2, this leads to similar objective function as linear SVM but with different constraints (see textbook)

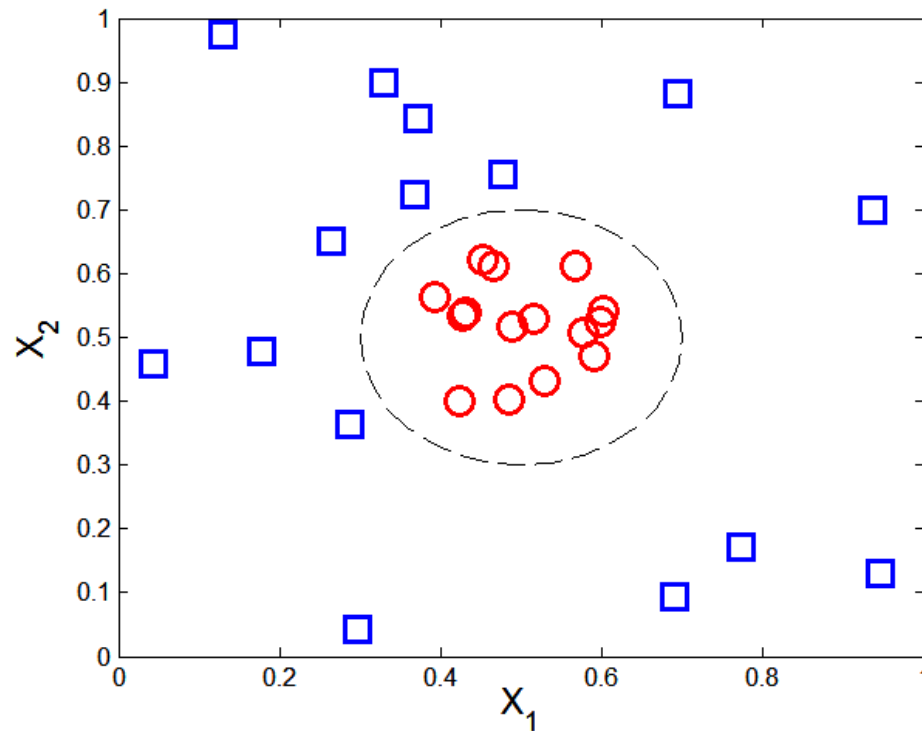
Support Vector Machines



- Find the hyperplane that optimizes both factors

Nonlinear Support Vector Machines

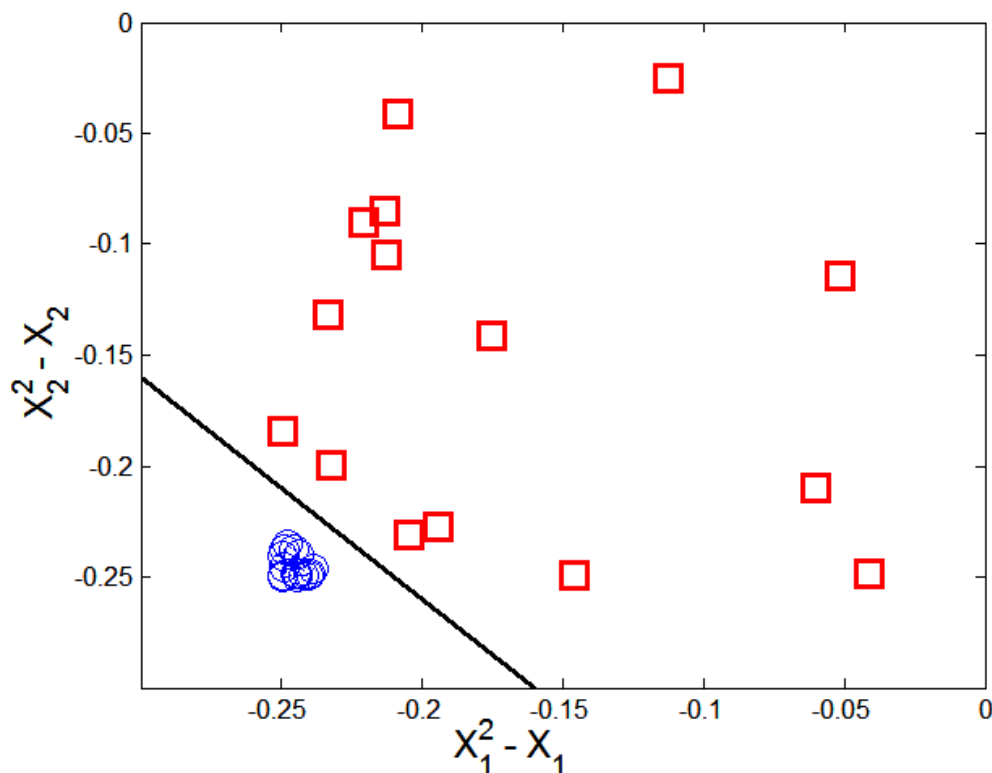
- What if decision boundary is not linear?



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

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:

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

Learning Nonlinear SVM

- Optimization problem:

$$\min_w \frac{\|\mathbf{w}\|^2}{2}$$

subject to $y_i(\mathbf{w} \cdot \Phi(\mathbf{x}_i) + b) \geq 1, \forall \{(\mathbf{x}_i, y_i)\}$

- Which leads to the same set of equations (but involve $\Phi(\mathbf{x})$ instead of \mathbf{x})

$$L_D = \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) \quad \mathbf{w} = \sum_i \lambda_i y_i \Phi(\mathbf{x}_i)$$
$$\lambda_i \{ y_i (\sum_j \lambda_j y_j \Phi(\mathbf{x}_j) \cdot \Phi(\mathbf{x}_i) + b) - 1 \} = 0,$$

$$f(\mathbf{z}) = \text{sign}(\mathbf{w} \cdot \Phi(\mathbf{z}) + b) = \text{sign}(\sum_{i=1}^n \lambda_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{z}) + b).$$

Learning NonLinear SVM

- Issues:
 - What type of mapping function Φ should be used?
 - How to do the computation in high dimensional space?
 - ◆ Most computations involve dot product $\Phi(x_i) \bullet \Phi(x_j)$
 - ◆ Curse of dimensionality?

Learning Nonlinear SVM

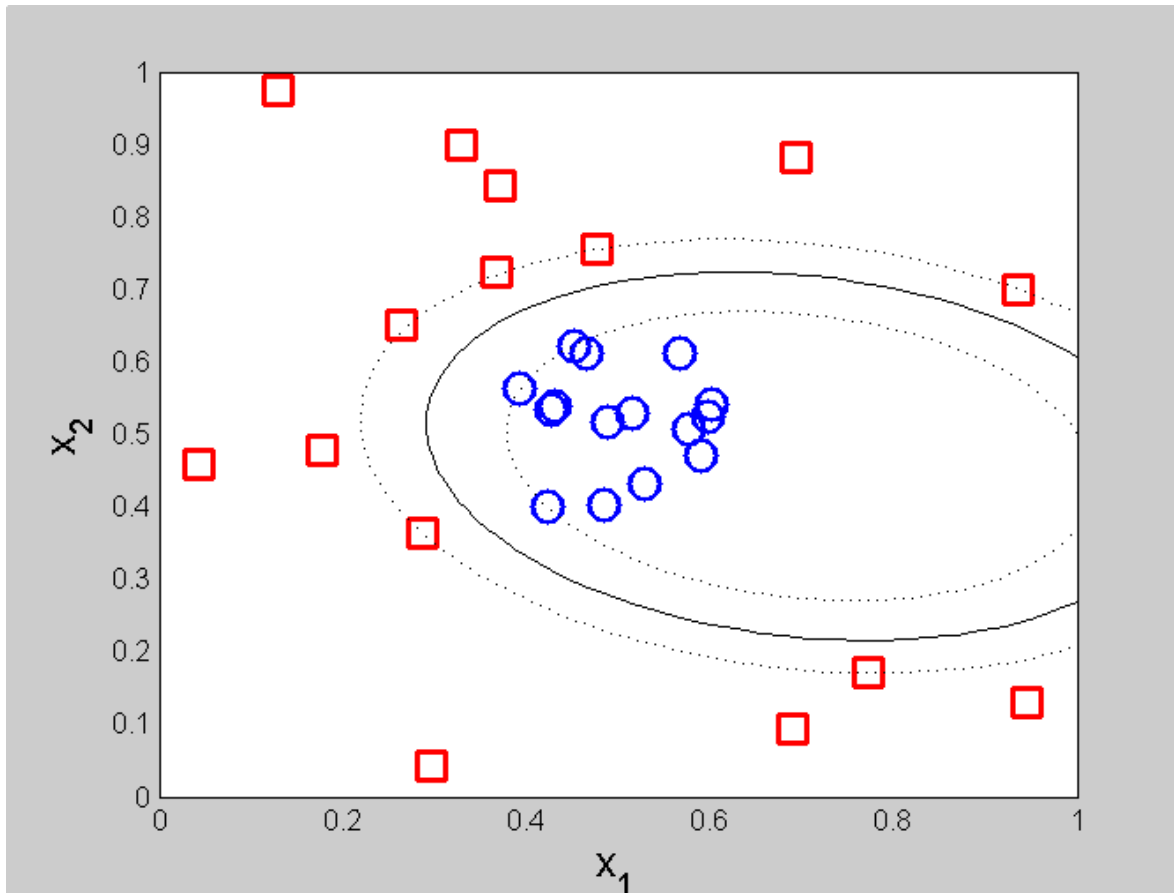
- Kernel Trick:
 - $\Phi(\mathbf{x}_i) \bullet \Phi(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$
 - $K(\mathbf{x}_i, \mathbf{x}_j)$ is a kernel function (expressed in terms of the coordinates in the original space)
 - ◆ Examples:

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$$

$$K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x} - \mathbf{y}\|^2 / (2\sigma^2)}$$

$$K(\mathbf{x}, \mathbf{y}) = \tanh(k\mathbf{x} \cdot \mathbf{y} - \delta)$$

Example of Nonlinear SVM



**SVM with polynomial
degree 2 kernel**

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 (see textbook)

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 better than many other techniques
- The user needs to provide the type of kernel function and cost function
- Difficult to handle missing values
- What about categorical variables?