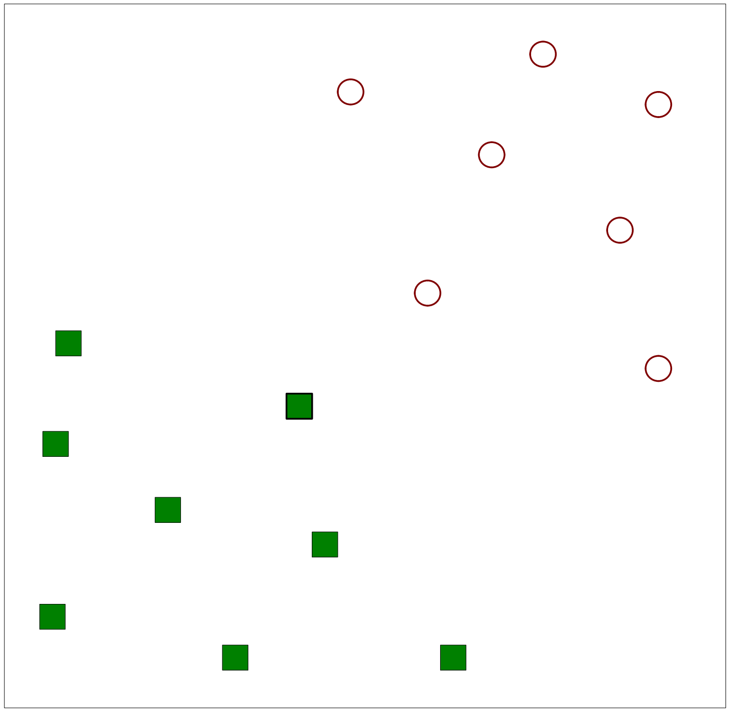
Support Vector Machines

CS 584 Data Mining (Spring 2022)

Prof. Sanmay Das
George Mason University

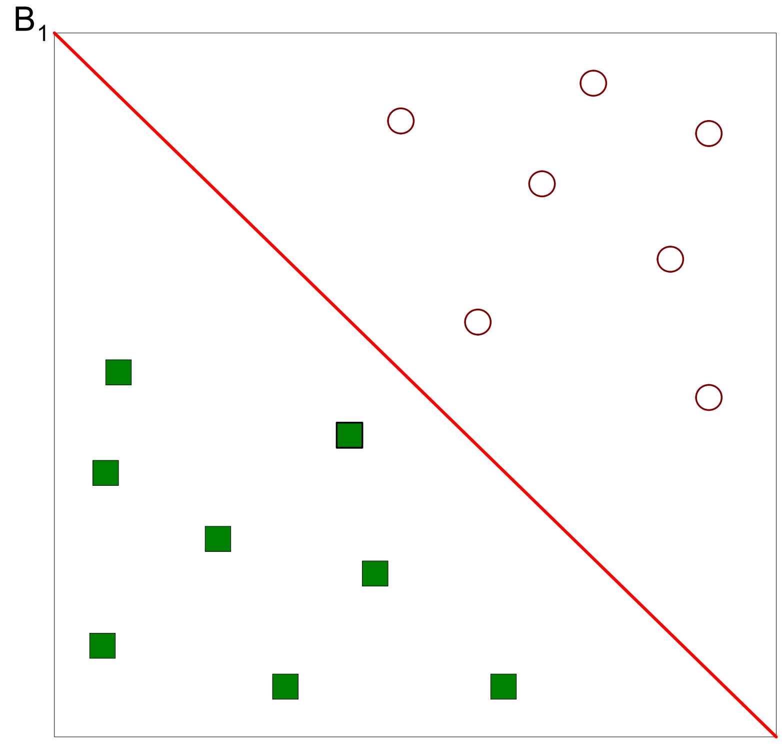
Slides are adapted from the available book slides developed by Tan, Steinbach and Kumar

Support Vector Machines



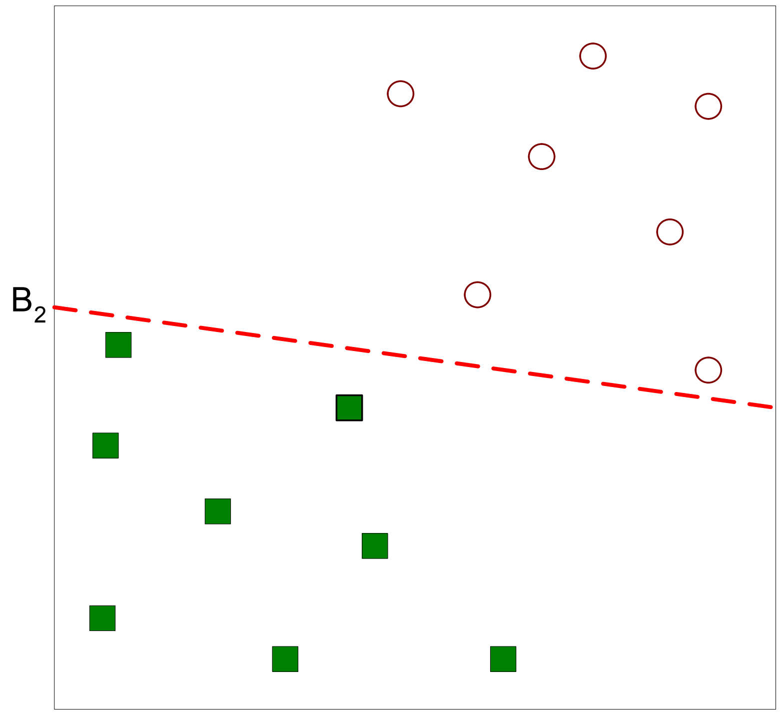
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



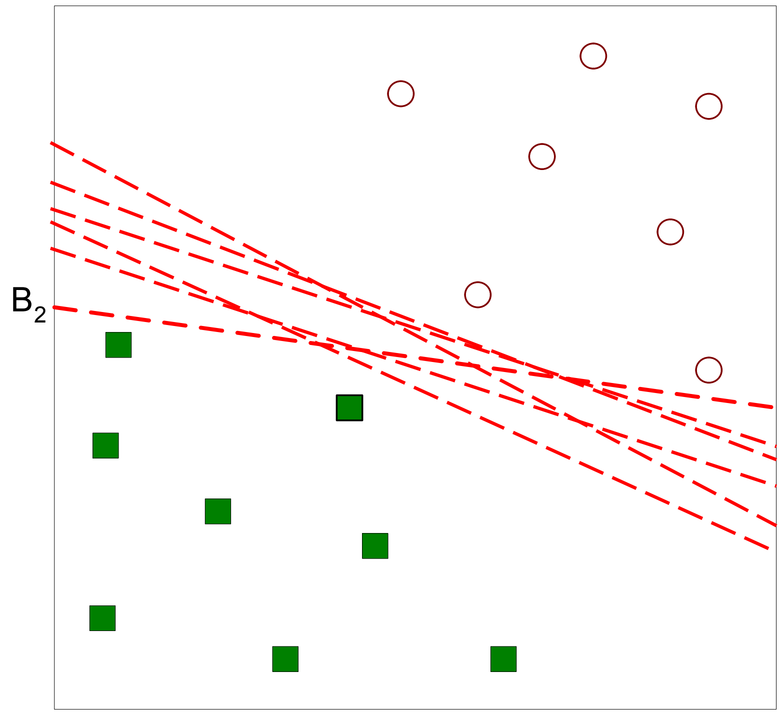
- One Possible Solution

Support Vector Machines



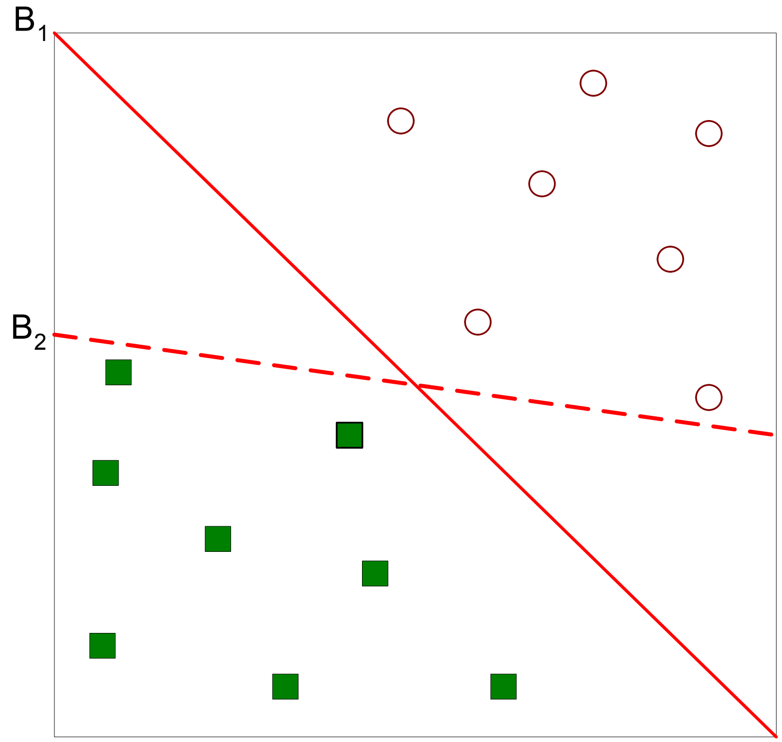
- Another possible solution

Support Vector Machines



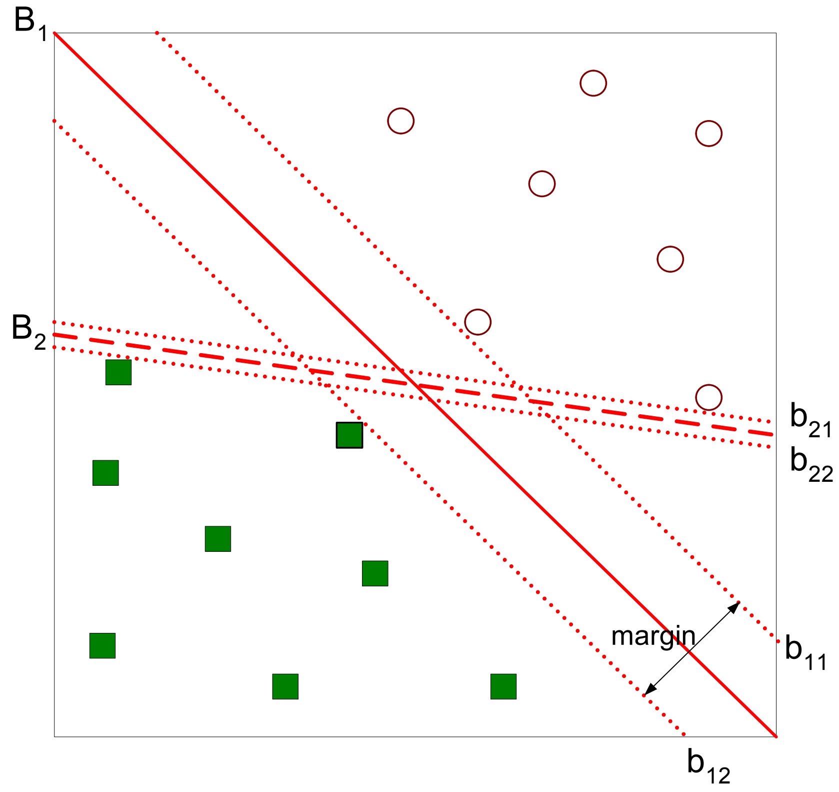
- Other possible solutions

Support Vector Machines



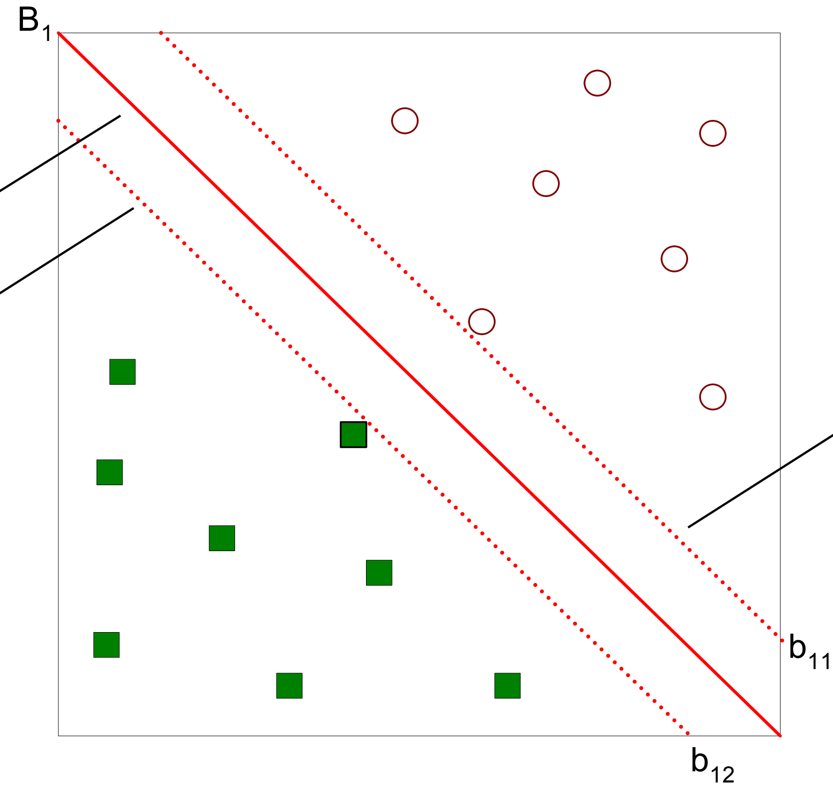
- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



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

Support Vector Machines



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

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

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

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

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

Linear SVM

- Linear model:

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

- Learning the model is equivalent to determining the values of $\vec{w}$ and b
  - How to find them 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 constraints:

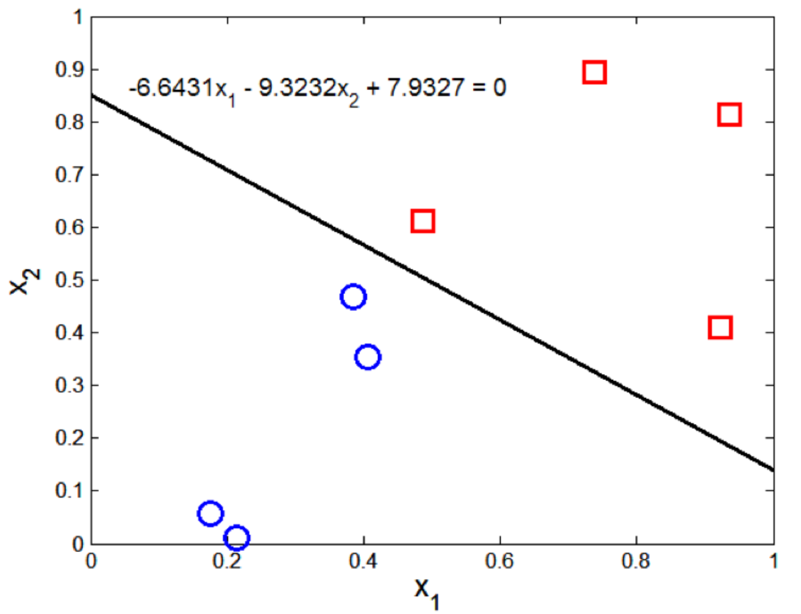
$$y_i = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 \end{cases}$$

  or $y_i(\mathrm{w} \bullet \mathrm{x}_i + b) \geq 1, \quad i = 1,2,\ldots,N$

  - This is a constrained optimization problem
    - Solve it using Lagrange multiplier method
    - Yields a quadratic programming problem

- We get the dual of the original problem:
  - $\max_{\lambda_i} \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j \mathbf{x}_i^T \mathbf{x}_j$
  - Subject to $\sum_{i=1}^{n} \lambda_i y_i = 0$
  - And $\lambda_i \geq 0$
- Cool things:
  - Most of the λ_i will be 0: *Sparsity*
  - Objective only involves dot products of training examples $(\mathbf{x}_i, \mathbf{x}_j)$
  - Can recover $\mathbf{w}, b$ easily

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

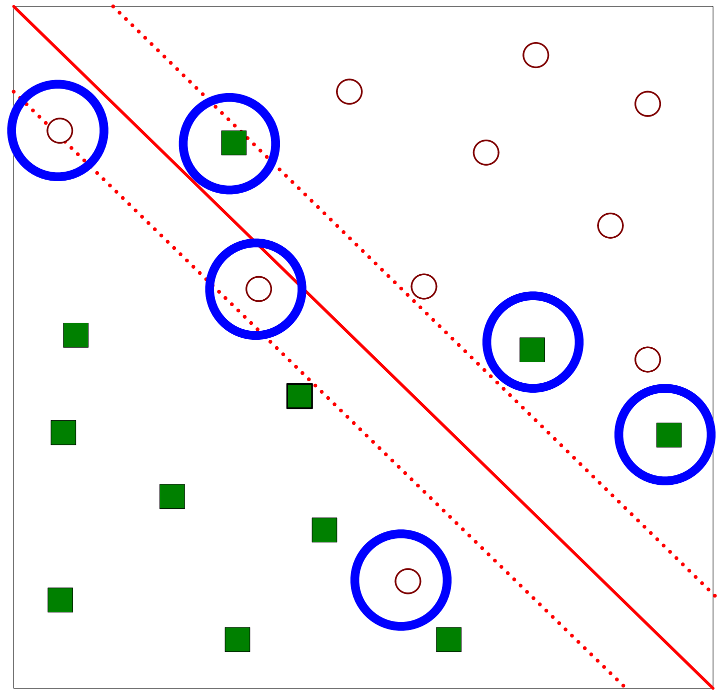
Learning Linear SVM

- 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 **w** and *b* are found? Given a test record, x_i

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \ge 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \le -1 \end{cases}$$

Support Vector Machines

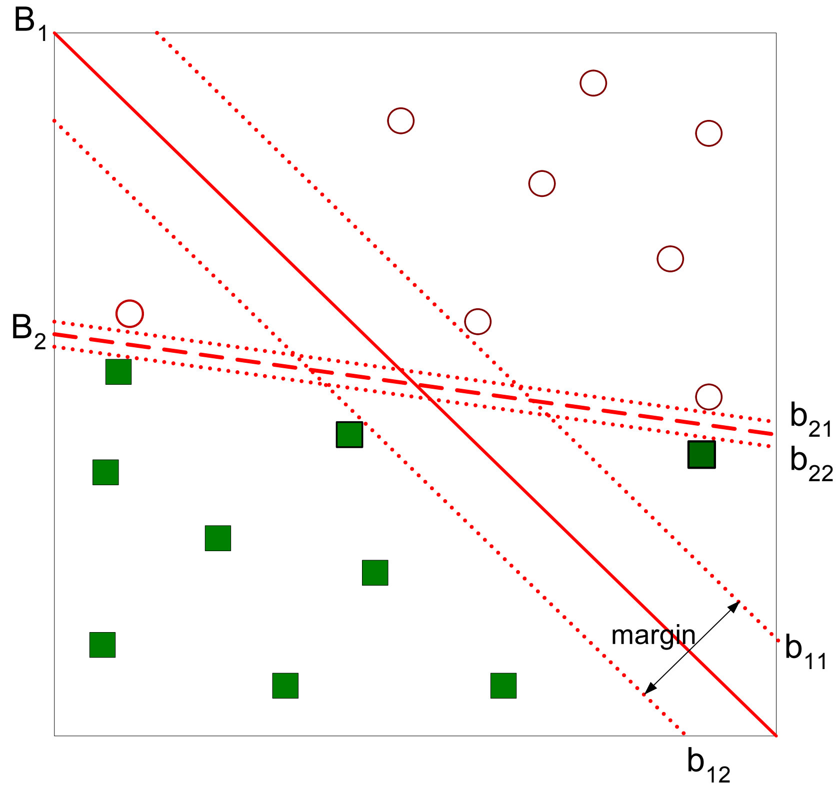
- What if the problem is not linearly separable?



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:
      - $y_i(\mathbf{w} \,.\, \mathbf{x}_i + b) \geq \xi_i \quad \forall i$
      - $\xi_i \geq 0 \quad \forall i$
  - Dual:
    - $\max\limits_{\lambda_i} \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$
    - Subject to $\sum_{i=1}^{n} \lambda_i y_i = 0$
    - And $0 \leq \lambda_i \leq C$

Support Vector Machines



- Find the hyperplane that optimizes both factors

Another View of SVMs

$$\hat{y} = \mathbf{w}^T\mathbf{x} + b$$
$$\text{where } y \in \{+1, -1\}$$

Now define the hinge loss as $\text{Loss}(y, \hat{y}) = \max(0, 1 - y\hat{y})$

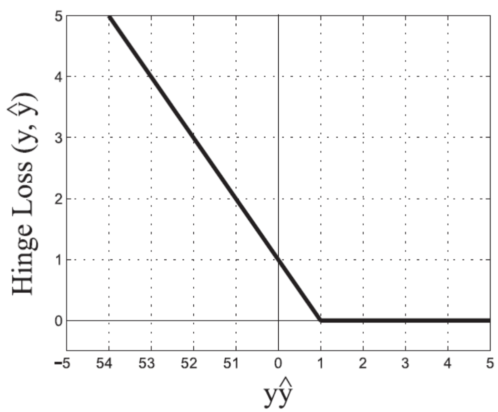


Figure 4.37. Hinge loss as a function of $y\hat{y}$.

The optimization problem is equivalent to:

$$\min_{\mathbf{w},b} C \sum_{i=1}^{n} \text{Loss}(y_i, \hat{y}_i) + \frac{\|w\|^2}{2}$$

Training error

Complexity penalty