

# CS5489

## Lecture 3.2: Support Vector Machines: Part I

Kede Ma

City University of Hong Kong (Dongguan)



香港城市大學（東莞）  
City University of Hong Kong  
(Dongguan)

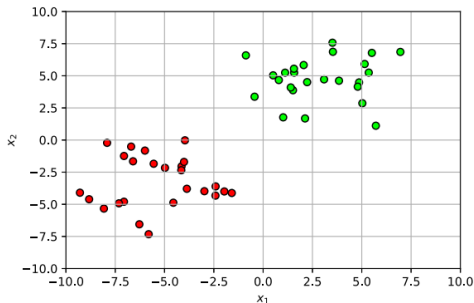
Slide template by courtesy of Benjamin M. Marlin

# Overview

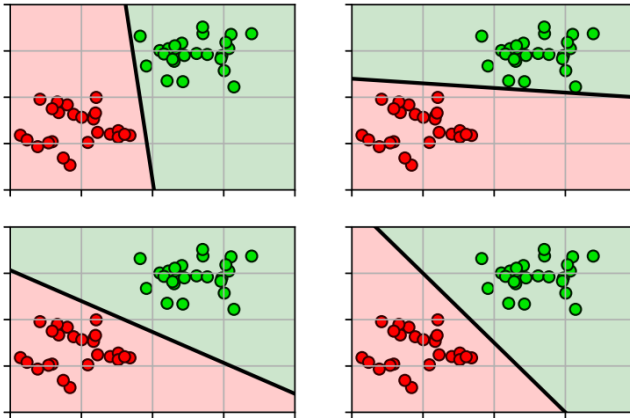
- To date we've seen one example of a discriminative linear classifier
  - I.e., logistic regression, which used a maximum likelihood framework to learn the separating hyperplane
- Today we'll introduce a second example, support vector machines (SVMs)
  - A purely geometric approach

# Linearly Separable Data

- For now, assume the training data is **linearly separable**
  - The two classes in the training data can be separated by a line (hyperplane)
  - Example: iris dataset

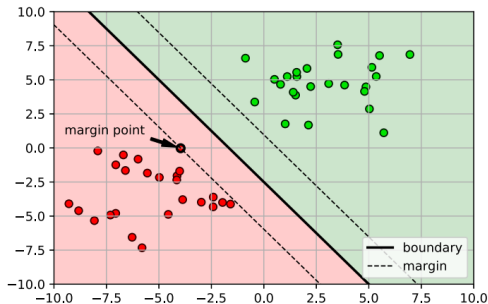


- There are many possible solutions



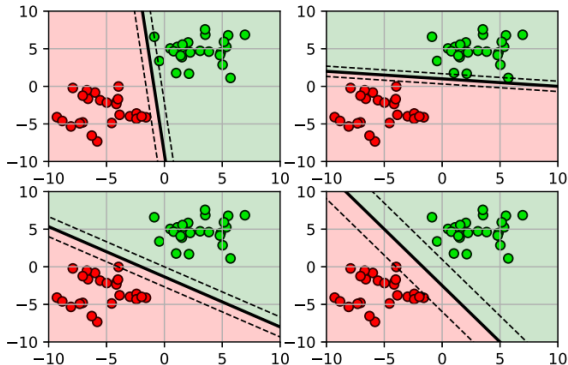
# Maximum Margin Principle

- Define the distance between the separating line and the closest point as the **margin**
  - Think of this space as the “amount of wiggle room” for accommodating errors in estimating  $\mathbf{w}$



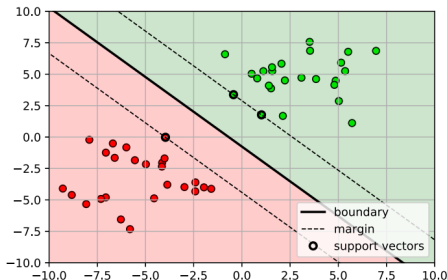
# Maximum Margin Principle

- **Idea:** the best separating line is the one that maximizes the margin
  - I.e., puts the most distance between the closest points and the decision boundary



# Maximum Margin Principle

- The solution...
  - By symmetry, there should be at least one margin point on each side of the boundary
  - The points on the margins are called the support vectors
    - The points support (define) the margin



# Computing the Margin

- What is the margin (i.e., distance) of  $\mathbf{w}^T \mathbf{x} + b = 0$  with respect to a point  $(\mathbf{x}^{(i)}, y^{(i)})$ ?

$$d^{(i)} = \frac{y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b)}{\|\mathbf{w}\|_2}$$

- This is called the **geometric** margin
- The numerator  $y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b)$  is called the **functional** margin
- What is the margin of  $\mathbf{w}^T \mathbf{x} + b = 0$  with respect to a training set  $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)}), i = 1, \dots, M\}$ ?

$$d = \min\{d^{(i)}\}_{i=1}^M = \min_{(\mathbf{x}, y) \in \mathcal{D}} \frac{y(\mathbf{w}^T \mathbf{x} + b)}{\|\mathbf{w}\|_2}$$



# Maximizing the Margin

- The maximum margin solution is found by solving

$$\max_{\mathbf{w}, b} \left( \frac{1}{\|\mathbf{w}\|_2} \min_{(\mathbf{x}, y) \in \mathcal{D}} y(\mathbf{w}^T \mathbf{x} + b) \right)$$

- Notice that if we make the rescaling  $\mathbf{w} \rightarrow \gamma \mathbf{w}$  and  $b \rightarrow \gamma b$ , then the objective function is unchanged
- We can use this freedom to set

$$\min_{(\mathbf{x}, y) \in \mathcal{D}} y(\mathbf{w}^T \mathbf{x} + b) = 1$$

- Or equivalently,

$$y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1, \forall i = 1, \dots, M$$

# SVM (Primal Form, Hard-Margin)

- Given a training set  $\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^M$ , optimize:

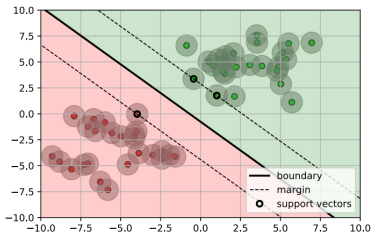
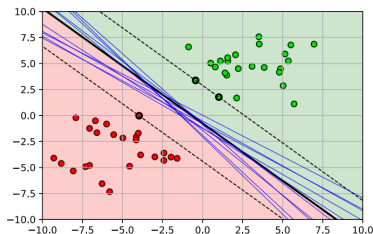
$$\min_{\mathbf{w}, b} \quad \frac{1}{2} \|\mathbf{w}\|_2^2$$

$$\text{subject to } y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1, \quad i = 1, \dots, M$$

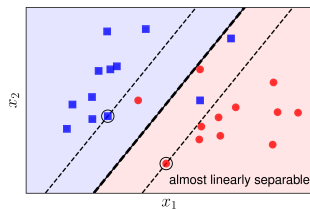
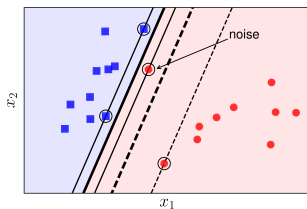
- The objective minimizes the inverse of the margin distance, i.e., maximizes the **geometric** margin
- The inequality constraints ensure that all points are either on or outside of the **functional** margin
  - The functional margin is set to be distance of 1 from the boundary
- Prediction:
  - Given a new data point  $\mathbf{x}^*$ , use sign of linear function to predict class
    - $y^* = \text{sign}(\mathbf{w}^T \mathbf{x}^* + b)$

# Why is Maximizing the Margin Good?

- The true  $\mathbf{w}$  is uncertain
  - Maximizing the margin allows the most uncertainty (wiggle room) for  $\mathbf{w}$ , while keeping all the points correctly classified
- The data points are uncertain
  - Maximizing the margin allows the most wiggle of the points, while keeping all the points correctly classified

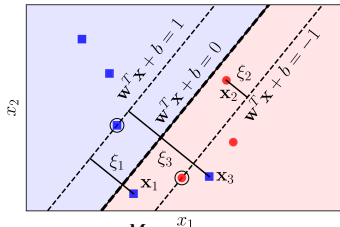


# Is Hard-Margin Enough?



- Solution: use the same linear classifier
  - Allow some training samples to violate margin
    - I.e., are inside the margin (or even mis-classified)
  - Define “slack” variable  $\xi_i \geq 0$ 
    - $\xi_i = 0$  means sample is outside of margin area (no slack)
    - $\xi_i > 0$  means sample is inside of margin area (slack)

# SVM (Primal Form, Soft-Margin)

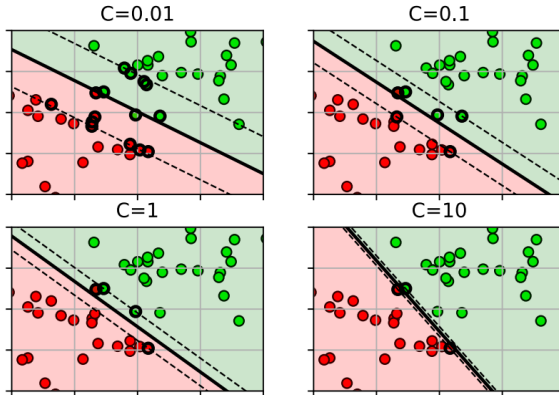


$$\min_{\mathbf{w}, b} \quad \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^M \xi_i$$

subject to  $y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, \dots, M$

- Introduces a parameter  $C$  as penalty for violating the margin
  - Smaller value means allow more violations (less penalty)
  - Larger value means don't allow violations (more penalty)

# The Effect of $C$



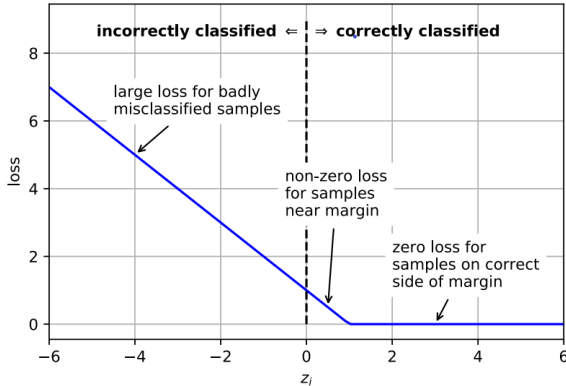
# Loss Function

- In the case of primal-form SVM with soft-margin, it is equivalent to minimizing the function:

$$\sum_{i=1}^M \max(0, 1 - y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b)) + \frac{1}{C} \|\mathbf{w}\|_2^2,$$

- Hinge loss function  $\ell_h(z) = \max(0, 1 - z)$ 
  - Note:  $\max(a, b)$  returns whichever value ( $a$  or  $b$ ) is larger
- It takes tens of years from hinge loss to SVM

# Hinge Loss





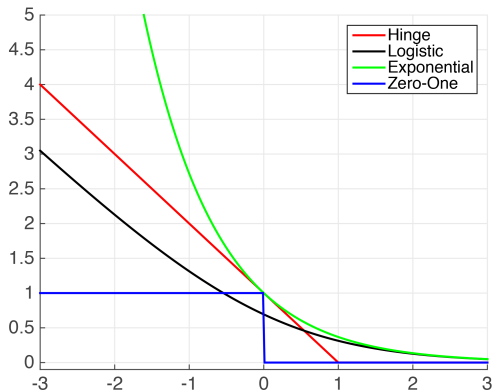
# Zero-One Loss

- Both the logistic loss and the hinge loss are convex relaxations of the zero-one loss:

$$\ell_{01}(z) = \mathbb{I}[z \leq 0]$$

- The average zero-one loss over a data set is exactly the classification error rate
- This is the loss function we'd like to minimize, but this generally isn't computationally feasible, thus the need for surrogate loss functions

# Loss Function Comparison



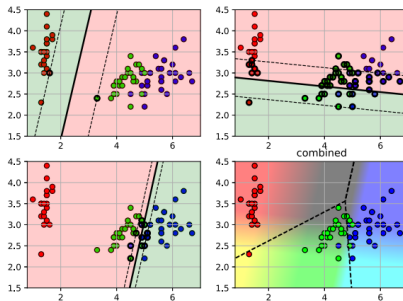
- Exponential loss:  $\ell_e(z) = \exp(-z)$ 
  - Used in the development of other machine learning algorithms, e.g., AdaBoost

# Multiclass SVM

- “1-vs-1” multiclass classification
  - Train binary classifiers on all pairs of classes
    - 3-class example: 1 vs 2, 1 vs 3, 2 vs 3
  - To label a sample, pick the class with the most votes among the binary classifiers
  - **Problem:** 1v1 classification is very slow when there are a large number of classes
    - If there are  $C$  classes, need to train  $C(C - 1)/2$  binary classifiers!
- “1-vs-rest” multiclass classification
  - Train 1-vs-rest binary classifiers
    - 3-class example: 1 vs {2,3}, 2 vs {1,3}, 3 vs {1,2}
  - To label a sample, pick the class with the largest geometric margin
    - $y^* = \arg \max_{c \in \{1, \dots, C\}} (\mathbf{w}_c^T \mathbf{x}^* + b_c) / \|\mathbf{w}_c\|_2$

# Example: 3-Class Iris Dataset

## ■ Decision boundaries for each binary classifier



# SVM Summary

## ■ Classifier:

- Linear function  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$
- Given new sample  $\mathbf{x}^*$ , predict  $y^* = \text{sign}(\mathbf{w}^T \mathbf{x}^* + b)$

## ■ Training:

- Maximize the margin of the training data
  - I.e., maximize the separation between the points and the decision boundary
- Allow some training samples to violate the margin
  - Use cross-validation to pick the hyperparameter  $C$

# Summary

## ■ Linear classifiers:

- Separate the data using a linear surface (hyperplane)
- $y = \text{sign}(\mathbf{w}^T \mathbf{x} + b)$

## ■ Two formulations:

- Logistic regression - maximize the probability of the data
- Support vector machine - maximize the margin of the hyperplane

## ■ Loss functions:

- Logistic regression - push the classification boundary as far as possible from all points
- Support vector machine - ensure a margin of 1 between boundary and the closest point

# Summary

## ■ Advantages:

- SVM works well on high-dimensional features ( $N$  large), and has low generalization error
- LR has well-defined probabilities

## ■ Disadvantages:

- Decision surface can only be linear!
  - Next lecture we will see how to deal with non-linear decision surfaces