

Machine Learning Basics & Linear Models

Weizi Li

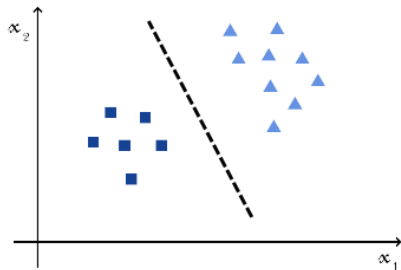
Department of Computer Science
University of Memphis



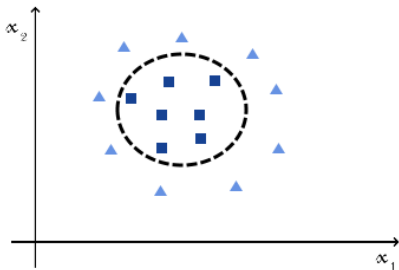
- Linear Classification
- Margins, Regularization, and Optimization
- Overfitting vs. Underfitting
- Bias-Variance Trade-off
- Linear Regression

Linear Classification

- $\theta \cdot X + \theta_0 = 0$
- X : both a vector and a point in high-dimensional space

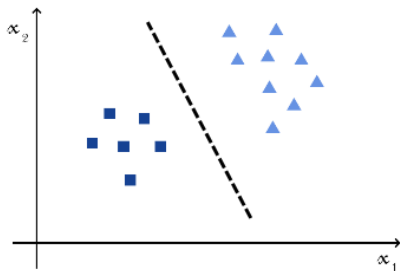


A linear decision boundary

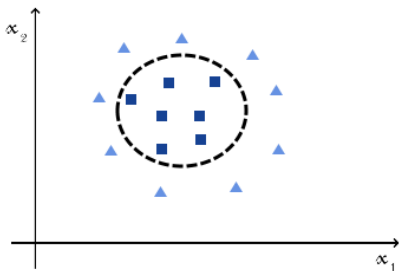


A non-linear decision boundary

- θ : also a vector, determine the orientation of the decision boundary
 - ▶ Consider $\theta \cdot X = 0$ (a linear classifier goes through the origin), θ is perpendicular to X , thus determine the orientation



A linear decision boundary

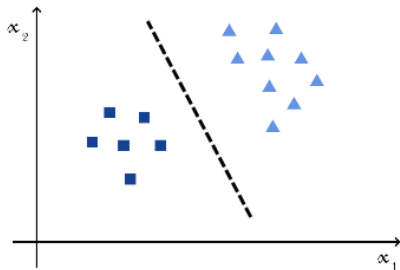


A non-linear decision boundary

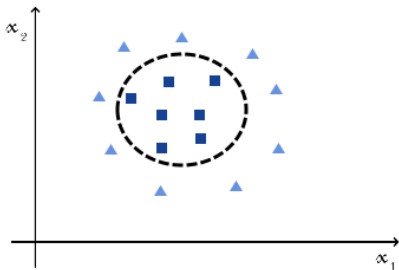
- Note that different θ can result in the same linear classifier, i.e., $c\theta \cdot X = \theta \cdot X = 0$

- $\theta \cdot X + \theta_0 = 0$

- θ_0 : determine the location of the decision boundary



A linear decision boundary

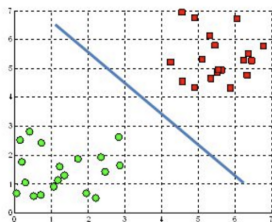


A non-linear decision boundary

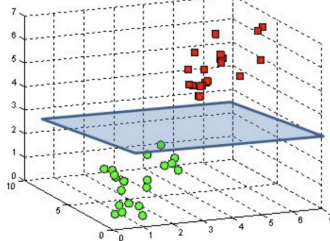
- Why it's called linear classifier?
- Linear operations
 - ▶ Addition
 - ▶ Multiplication with a constant
- $\theta \cdot X$: only linear operations conducted on the elements of X

- Linear classifier go beyond a “line”

A hyperplane in \mathbb{R}^2 is a line



A hyperplane in \mathbb{R}^3 is a plane



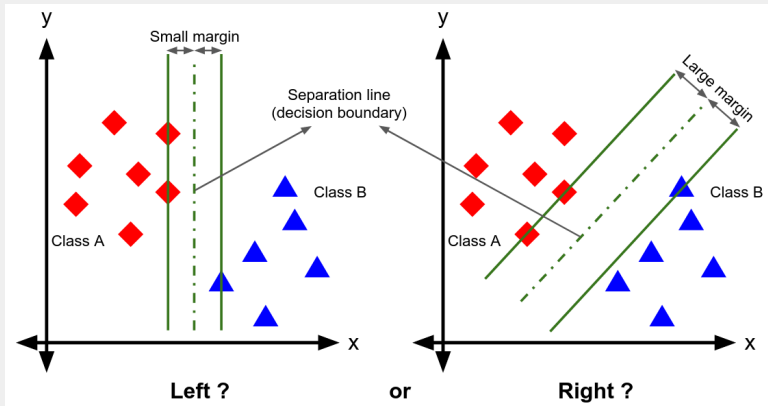
- For n training examples $S_n = \{(X^{(i)}, y^{(i)}), i = 1, \dots, n\}$ if there exist θ and θ_0 such that $y^{(i)}(\theta \cdot X^{(i)} + \theta_0) > 0$ for all $i = 1, \dots, n$, we call these training examples linearly separable
 - ▶ $X^{(i)}$: i th training example
 - ▶ $y^{(i)}$: i th training example's label (+ or -)
 - ▶ $\theta \cdot X^{(i)} + \theta_0$: result of the classifier (+ or -)
- $y^{(i)}$ and $\theta \cdot X^{(i)} + \theta_0$ should have the same sign if the classifier works correctly

- Occurs when classifier makes a mistake
- For linear classification, occurs when data are not linearly separable, otherwise there exist algorithms guarantee no error
- Denote h as the classifier, n as the total number of training examples, we have the classification error:

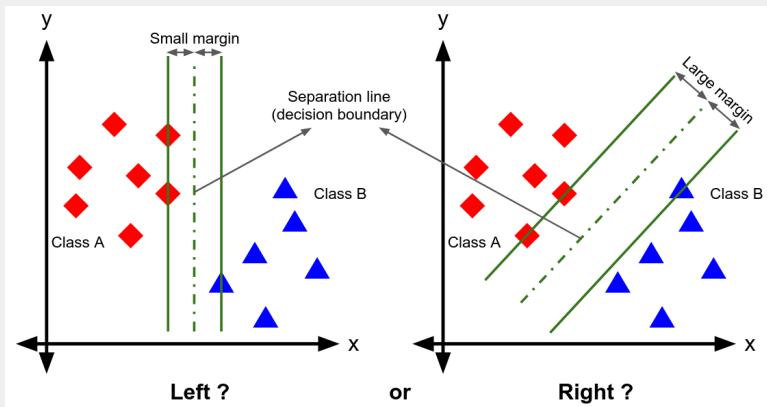
$$\epsilon_n(h) = \frac{1}{n} \sum_{i=1}^n [[h(X^{(i)}) \neq y^{(i)}]]$$

Margins, Regularization, and Optimization

- Large margin: more robust classifier (against noise), less accurate



- Small margin: less robust classifier, more accurate



- Hinge loss encourages large margins
- Denote ζ as the distance from the margin boundary to the decision boundary:
 - ▶ When $y^{(i)}(\theta \cdot X^{(i)} + \theta_0) > \zeta$, loss = 0
 - ▶ When $y^{(i)}(\theta \cdot X^{(i)} + \theta_0) \leq \zeta$, loss = $\zeta - y^{(i)}(\theta \cdot X^{(i)} + \theta_0)$

- Recall the decision boundary of linear classification

$$\theta \cdot X + \theta_0 = 0$$

- Then, the margin boundary of the decision boundary

$$\theta \cdot X + \theta_0 = \zeta$$

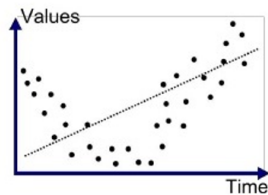
- If divide both sides by $\|\theta\|$, the decision boundary won't change but the margin boundary ($\frac{\zeta}{\|\theta\|}$) will decrease if $\|\theta\|$ increases
- In general, the goal of regularization is to obtain a more robust classifier, thus prefer large margins (small $\|\theta\|$)

- Objective function

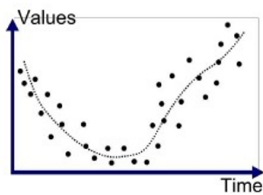
$$J(\theta, \theta_0) = \epsilon_n(h) + \frac{\lambda}{2} \|\theta\|^2$$

- λ : regularization parameter (a hyperparameter)

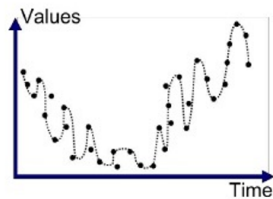
Overfitting vs. Underfitting



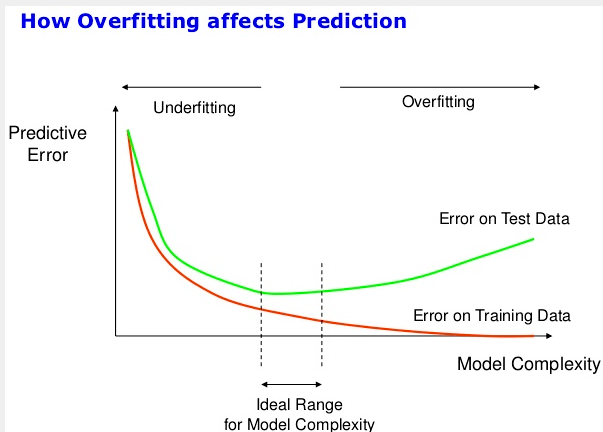
Underfitted



Good Fit/Robust



Overfitted

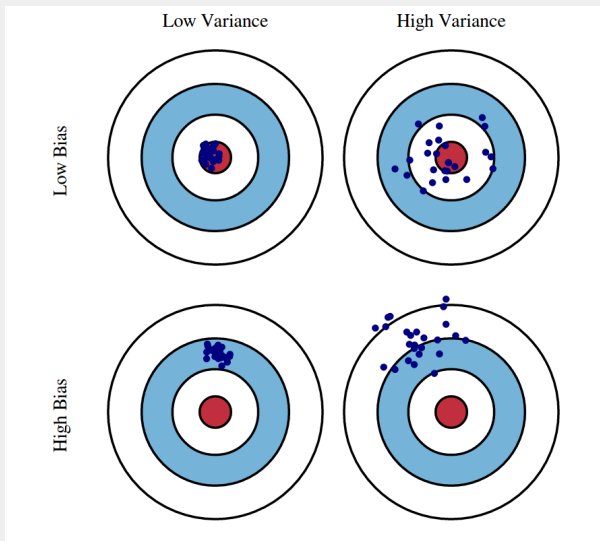


- Recall the objective function $J(\theta, \theta_0) = \epsilon_n(h) + \frac{\lambda}{2} \|\theta\|^2$
- Increase λ will encourage smaller θ (less complex model), thus prevent overfitting

Bias-Variance Trade-off

- Assume the underlying true relationship between X and y is $y = f(X) + \epsilon$
- X, y : data and label
- f : actual mapping (unknown)
- ϵ : irreducible error (unknown)
- \hat{f} : trained model depends on the training set (not fixed)

- $\mathbb{E} \left[\left(y - \hat{f}(x) \right)^2 \right] = \left(f(x) - \mathbb{E} \left[\hat{f}(x) \right] \right)^2 + \mathbb{E} \left[\left(\hat{f}(x) - \mathbb{E} \left[\hat{f}(x) \right] \right)^2 \right] + \mathbb{E} [\epsilon^2]$
- $\left(f(x) - \mathbb{E} \left[\hat{f}(x) \right] \right)^2$: bias
- $\mathbb{E} \left[\left(\hat{f}(x) - \mathbb{E} \left[\hat{f}(x) \right] \right)^2 \right]$: variance
- $\mathbb{E} [\epsilon^2]$: irreducible error
- Because the left-hand side and $\mathbb{E} [\epsilon^2]$ are fixed, when bias increases variance decreases, and vice versa



- Overfitting: low bias and high variance
- Underfitting: high bias and low variance
- Desired model: balancing bias and variance (neither overfit nor underfit)

Linear Regression

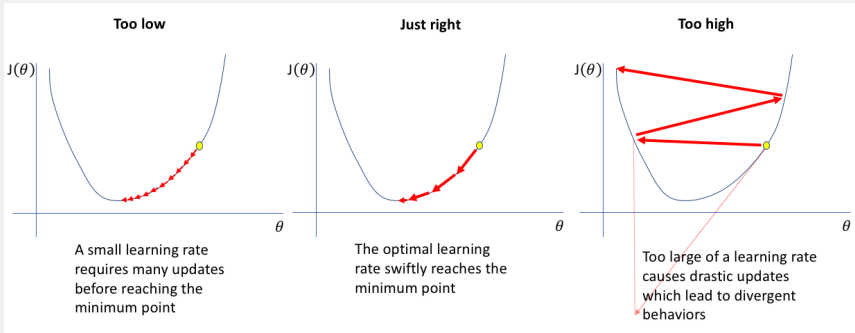
- Denote $h(X^{(i)}) = \theta \cdot X^{(i)} + \theta_0$
- Classification
 - ▶ y takes discrete values
 - ▶ Example loss: $\frac{1}{n} \sum_{i=1}^n [[h(X^{(i)}) \neq y^{(i)}]]$
- Regression
 - ▶ y takes continuous values
 - ▶ Example loss: $\frac{1}{n} \sum_{i=1}^n \frac{(y^{(i)} - h(X^{(i)}))^2}{2}$

- Compact representation: $\theta \cdot X + \theta_0 \rightarrow \theta' X'$ ($\theta' = [\theta, \theta_0]$, $X' = [X, 1]$)
- Step 1: initialize θ'
- Step 2: randomly select a training example (X', y)
- Step 3: update θ' via $\theta' \leftarrow \theta' + \alpha(y - \theta' \cdot X')X'$
 - ▶ α : learning rate
 - ▶ Where does this formula come from?
 - ▶ Why $+$?
- Step 4: if stopping criteria not meet, go to Step 2

- Compact representation: $\theta \cdot X + \theta_0 \rightarrow \theta' X'$ ($\theta' = [\theta, \theta_0]$, $X' = [X, 1]$)
- Step 1: initialize θ'
- Step 2: randomly select a training example (X', y)
- Step 3: update θ' via $\theta' \leftarrow \theta' + \alpha(y - \theta' \cdot X')X'$
 - ▶ α : learning rate
 - ▶ Where does this formula come from?
 - ▶ Why $+$?
- Step 4: if stopping criteria not meet, go to Step 2

- Compact representation: $\theta \cdot X + \theta_0 \rightarrow \theta' X'$ ($\theta' = [\theta, \theta_0]$, $X' = [X, 1]$)
- Step 1: initialize θ'
- Step 2: randomly select a training example (X', y)
- Step 3: update θ' via $\theta' \leftarrow \theta' + \alpha(y - \theta' \cdot X')X'$
 - ▶ α : learning rate
 - ▶ Where does this formula come from? $\nabla_{\theta} \frac{(y - \theta' \cdot X')^2}{2}$
 - ▶ Why +? we need to go the opposite direction of the gradient
- Step 4: if stopping criteria not meet, go to Step 2

- Small learning rate: slow learning
- Large learning rate: overshoot (miss the optimal point)





- Only applicable to linear problems (matrix operation is linear)
- Solution: $\theta' = \left(\frac{1}{n} \sum_{i=1}^n X^{(i)} (X^{(i)})^T \right)^{-1} \left(\frac{1}{n} \sum_{i=1}^n y^{(i)} X^{(i)} \right)$
- Pros: obtain the solution in one go
- Cons: 1) needs the matrix to be invertible, 2) matrix computation can be expensive