Deep Learning

Regression and Classification

Kyungwoo Song

Contents

- Linear Statistical Model
- Logistic Regression
- Naïve Bayes
- Decision Tree
- Random Forest

Source:

Linear Statistical Model

Linear Statistical Model

Review

Definition

A **linear statistical model** relating a random response Y to a set of independent variables $x_1, \dots, x_k$ is of the form $Y = \beta_0 + \beta_1 x + \beta_2 x_2 + \cdots + \beta_k x_k + \epsilon$, where $\beta_0, \dots, \beta_k$ are **unknown** parameters, ϵ is a random variable, and the variables $x_1, \dots, x_k$ assume known values. We will assume that $E[\epsilon] = 0$ and hence that $E(Y) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$

Note

- $E(Y) = \beta_0 + \beta_1 x$
  - $E(Y)$ is a linear function of x and linear function of β_0 and β_1
- $E(Y) = \beta_0 + \beta_1 x^2$
  - $E(Y)$ is not a linear function of x but linear function of β_0 and β_1
- **Linear statistical model for Y: $E(Y)$ is a linear function of the unknown**
- **parameters β_0, β_1 and not necessarily a linear function of x**
- $Y = \beta_0 + \beta_1 (\ln x) + \epsilon$ is a linear model

Source:

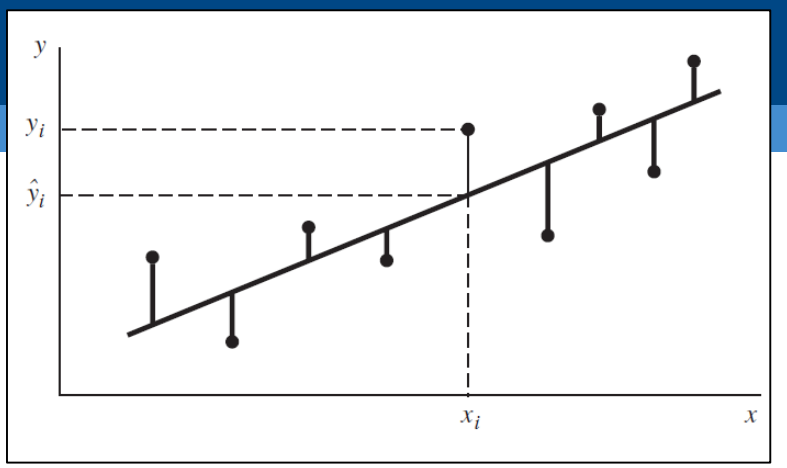
SSE

Review

simple vs multiple

$E(Y) = \beta_0 + \beta_1 x$

- $Y = \beta_0 + \beta_1 x + \epsilon$ where $E(\epsilon) = 0$
- We want to find estimators, $\hat{\beta}_0,\ \hat{\beta}_1$



Least-square method: fitting a line through a set of n data points

- Minimize the sum of squares of the vertical deviation from the fitted line
- Fitted line: $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$
- Sum of squares of the vertical deviation: $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}\left(y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)\right)^2$

SSE, sum of squares for error

If SSE possesses a minimum, it will occur for values of $\hat{\beta}_0$ and $\hat{\beta}_1$ s.t. $\frac{\partial SSE}{\partial \hat{\beta}_0} = 0$ and $\frac{\partial SSE}{\partial \hat{\beta}_1} = 0$

Source:

SSE

Review

- $\frac{\partial SSE}{\partial \hat{\beta}_0} = \frac{\partial\left\{\sum_{i=1}^{n}\left[y_i-(\hat{\beta}_0+\hat{\beta}_1 x_i)\right]^2\right\}}{\partial \hat{\beta}_0} = -\sum_{i=1}^{n} 2\left[y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i\right)\right] = 0$
  - $\Rightarrow \hat{\beta}_0 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\beta}_1 x_i) = \bar{y} - \hat{\beta}_1 \bar{x}$
- $\frac{\partial SSE}{\partial \hat{\beta}_1} = \frac{\partial\left\{\sum_{i=1}^{n}\left[y_i-(\hat{\beta}_0+\hat{\beta}_1 x_i)\right]^2\right\}}{\partial \hat{\beta}_1} = -\sum_{i=1}^{n} 2\left[y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i\right)\right] x_i = 0$
  - $\sum_{i=1}^{n} 2\left[y_i - \bar{y} + \hat{\beta}_1 \bar{x} - \hat{\beta}_1 x_i\right] x_i = 0$
  - $\sum_{i=1}^{n}(y_i - \bar{y}) x_i = \hat{\beta}_1 \sum_{i=1}^{n}(x_i - \bar{x})\, x_i$
  - $\Rightarrow \hat{\beta}_1 = \frac{\sum_{i=1}^{n}(y_i-\bar{y})x_i}{\sum_{i=1}^{n}(x_i-\bar{x})x_i}$

$$= \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}$$

$$= \frac{\sum_{i=1}^{n} x_i y_i - \frac{1}{n}\sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2}$$

Why?

$\sum_{i=1}^{n}(y_i - \bar{y})x_i = \sum_{i=1}^{n}(y_i - \bar{y})x_i - \bar{x}\sum_{i=1}^{n}(y_i - \bar{y})$

$= \sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})$

Fact: $\sum_{i=1}^{n}(y_i - \bar{y}) = 0$

$\hat{\beta}_1 = \frac{S_{xy}}{S_{xx}}$ where $S_{xy} = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$ and

$S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2$

Source:

Example

Example

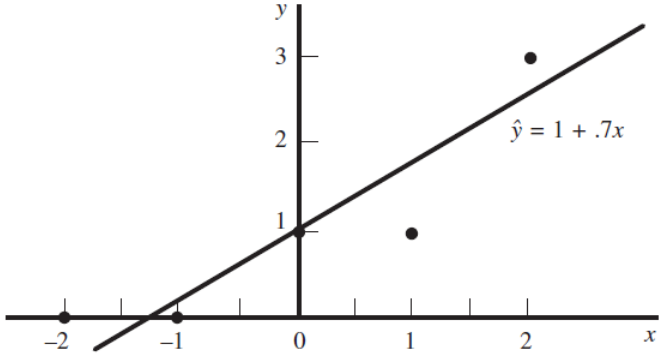
Use the method of least squares to fit a straight line to the $n = 5$ data points given in Table.

Table 11.1 Data for Example 11.1

x	y
−2	0
−1	0
0	1
1	1
2	3

Solution

- Fitted line: $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$
- $\hat{\beta}_1 = \frac{\sum_{i=1}^{n} x_i y_i - \frac{1}{n}\sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2} = 0.7$
- $\hat{\beta}_0 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\beta}_1 x_i) = \bar{y} - \hat{\beta}_1 \bar{x} = \frac{5}{5} - (0.7) * 0 = 1$



Source:

Multiple linear regression

Linear Statistical Model

지금까지 우리는 simple linear regression에 대해서만 살펴봄.

Multiple linear regression에 대해서는 어떻게 분석할 수 있을까?

- Linear model: $Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$
- We make n independent observations $y_1, .., y_n$
  - $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \epsilon_i$
  - x_{ij}: jth independent ith observation

예시) 대출가능 금액을 판단하고자 할 때, 여러가지 변수가 고려되어야 할 수 있음.
x_1: 신용등급, x_2: 대출희망금액, x_3: 자산 등

Another expression

- $Y = X\beta + \epsilon$
- $\hat{Y} = X\hat{\beta}$
- $e = Y - \hat{Y} = Y - X\hat{\beta}$

For simple case

- $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}\left(y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)\right)^2$

1

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_0 & x_{11} & x_{12} & \cdots & x_{1k} \\ x_0 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ x_0 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

Source:

Multiple linear regression

Linear Statistical Model

$$Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$$

- $Y = X\beta + \epsilon$
- $\hat{Y} = X\hat{\beta}$
- $e = Y - \hat{Y} = Y - X\hat{\beta} \in R^{n\times 1}$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_0 & x_{11} & x_{12} & \cdots & x_{1k} \\ x_0 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ x_0 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

$n \times k$

Sum of squared error: $e'e = \left(Y - X\hat{\beta}\right)'\left(Y - X\hat{\beta}\right) \in R^{1\times 1}$

- $e'e = (Y' - \hat{\beta}'X')(Y - X\hat{\beta})$

  $= Y'Y - \hat{\beta}'X'Y - Y'X\hat{\beta} + \hat{\beta}'X'X\hat{\beta}$

  $= Y'Y - 2\hat{\beta}'X'Y + \hat{\beta}'X'X\hat{\beta}$
- $\frac{\partial e'e}{\partial \hat{\beta}} = -2X'Y + 2X'X\hat{\beta} = 0$
- $\hat{\beta} = (X'X)^{-1}X'Y$

$\hat{\beta}'X'Y = Y'X\hat{\beta}$

$\hat{\beta}'X'Y, y'X\hat{\beta} \in R^{1\times 1}$

Source:

Multiple linear regression

Linear Statistical Model

- $Y = X\beta + \epsilon$
- $\hat{\beta} = (X'X)^{-1}X'Y$

x	y
-2	0
-1	0
0	1
1	1
2	3

Example

- Solve with matrix operations.

$$\mathbf{Y} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 3 \end{bmatrix}, \quad \text{and} \quad \mathbf{X} = \begin{matrix} x_0 & x_1 \\ \begin{bmatrix} 1 & -2 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{bmatrix} \end{matrix}.$$

It follows that

$$\mathbf{X'X} = \begin{bmatrix} 5 & 0 \\ 0 & 10 \end{bmatrix}, \quad \mathbf{X'Y} = \begin{bmatrix} 5 \\ 7 \end{bmatrix}, \quad (\mathbf{X'X})^{-1} = \begin{bmatrix} 1/5 & 0 \\ 0 & 1/10 \end{bmatrix}.$$

Thus,

$$\hat{\beta} = (\mathbf{X'X})^{-1}\mathbf{X'Y} = \begin{bmatrix} 1/5 & 0 \\ 0 & 1/10 \end{bmatrix}\begin{bmatrix} 5 \\ 7 \end{bmatrix} = \begin{bmatrix} 1 \\ .7 \end{bmatrix},$$

or $\hat{\beta}_0 = 1$ and $\hat{\beta}_1 = .7$. Thus,

$$\hat{y} = 1 + .7x,$$

Inverse Matrix

if $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

then $A^{-1} = \frac{1}{ad - bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

Source:

Inverse Matrix

Multiple linear regression

- $Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$
  - $\hat{Y} = X\hat{\beta}$
  - $\hat{\beta} = (X'X)^{-1}X'Y$
- 여기서 가장 어려운 부분이 무엇인가요?
  - $\hat{\beta} = (X'X)^{-1}X'Y$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_0 & x_{11} & x_{12} & \cdots & x_{1k} \\ x_0 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ x_0 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

$n \times k$

if $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

then $A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

- 1) Invertible 할까요?
  - 즉, $X'X$의 역행렬이 존재할까요?
  - 우리는 역행렬이 존재하지 않는 행렬도 알고 있습니다.
  - 예시) $X = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$ 라고 하면, X의 역행렬은 존재하지 않습니다.
  - 그렇다면, $X'X$는 어떨까요?
  - $X = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$이면, $X'X = \begin{bmatrix} 5 & 10 \\ 10 & 20 \end{bmatrix}$이 되고, 그러면 $X'X$ 의 역행렬 또한 존재하지 않습니다.
  - 만약 X가 full-rank matrix이면, $X'X$ 는 역행렬이 존재합니다.

❖이런것들은 선형대수학 관련 내용을 다룰 때, 잠깐 같이 다룰 예정입니다.

Source:

Inverse Matrix

Multiple linear regression

- $Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$
  - $\hat{Y} = X\hat{\beta}$
  - $\hat{\beta} = (X'X)^{-1}X'Y$
- 여기서 가장 어려운 부분이 무엇인가요?
  - $\hat{\beta} = (X'X)^{-1}X'Y$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_0 & x_{11} & x_{12} & \cdots & x_{1k} \\ x_0 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ x_0 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

$n \times k$

if $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

then $A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

- 2) 계산하기는 간편할까요?
  - 지금까지 우리가 직접 역행렬을 구한 예시는 2 by 2 행렬이었습니다.
  - 조금 더 행렬이 커지면 어떻게 될까요?
  - k by k행렬에 대해서 역행렬을 구하는 시간복잡도는 $O(k^3)$
    - ❖쉽게 생각해, k가 커질수록, 세제곱에 비례해서 더 많은 시간이 걸린다는 뜻입니다.
    - ❖굉장히 비효율적입니다.
  - 우리가 분석하고 싶은 데이터 X의 크기가 n by k라고 하면, $X'X$는 k by k가 되고, $(X'X)^{-1}$를 계산하기 위한 시간복잡도는 $O(k^3)$ 입니다.
  - 만약 우리가 분석하고 싶은 데이터가 굉장히 많은 feature (k)를 가지고 있을 경우에는 문제가 될 수 있겠죠?
    - ❖n: 데이터 개수, k: feature 개수

Source:

Gradient Descent

Multiple linear regression

- $Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$
  - $\hat{Y} = X\hat{\beta}$
  - $\hat{\beta} = (X'X)^{-1}X'Y$
- 여기서 가장 어려운 부분이 무엇인가요?
  - $\hat{\beta} = (X'X)^{-1}X'Y$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_0 & x_{11} & x_{12} & \cdots & x_{1k} \\ x_0 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ x_0 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

$n \times k$

if $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

then $A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

- 2) 계산하기는 간편할까요?
  - 그럼 어떻게 해야할까요? Gradient Descent!
  - 우리가 $\hat{\beta}$을 구한 과정을 다시 살펴보겠습니다.
  - Sum of squared error: $e'e = \left(Y - X\hat{\beta}\right)'\left(Y - X\hat{\beta}\right) \in R^{1\times 1}$
  - $e'e = \left(Y' - \hat{\beta}'X'\right)\left(Y - X\hat{\beta}\right)$
    $= Y'Y - \hat{\beta}'X'Y - Y'X\hat{\beta} + \hat{\beta}'X'X\hat{\beta}$
    $= Y'Y - 2\hat{\beta}'X'Y + \hat{\beta}'X'X\hat{\beta}$
  - $\frac{\partial e'e}{\partial \hat{\beta}} = -2X'Y + 2X'X\hat{\beta} = 0$
  - $\hat{\beta} = (X'X)^{-1}X'Y$

우리는 최적의 $\hat{\beta}$ 을 찾기 위해서, $\frac{\partial e'e}{\partial \hat{\beta}} = 0$으로 설정하였습니다.
그러지말고, $\frac{\partial e'e}{\partial \hat{\beta}}$ 값 자체만을 써 보겠습니다.

Source:

Gradient Descent

Multiple linear regression

- $Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$
  - $\hat{Y} = X\hat{\beta}$
  - $\hat{\beta} = (X'X)^{-1}X'Y$
- 여기서 가장 어려운 부분이 무엇인가요?
  - $\hat{\beta} = (X'X)^{-1}X'Y$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_0 & x_{11} & x_{12} & \cdots & x_{1k} \\ x_0 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ x_0 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix},$$

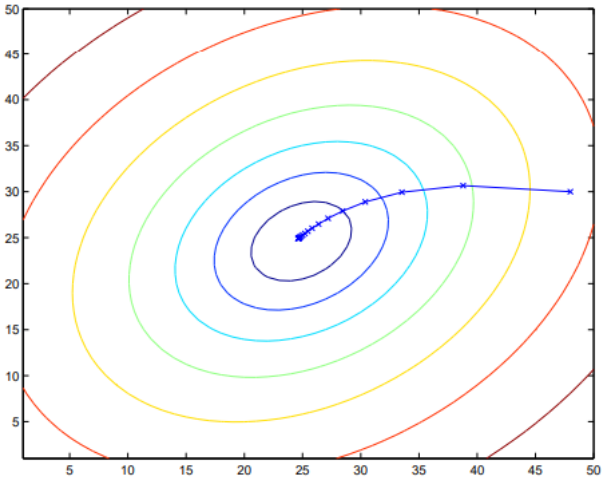
$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

$n \times k$

if $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

then $A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

- 2) 계산하기는 간편할까요?
  - 그럼 어떻게 해야할까요? Gradient Descent!
  - $\frac{\partial e'e}{\partial \hat{\beta}} = -2X'Y + 2X'X\hat{\beta}$
  - $\hat{\beta}^{(0)} = 0.0$ (초기값 설정)
  - $\hat{\beta}^{(t+1)} = \hat{\beta}^{(t)} - \eta \frac{\partial e'e}{\partial \hat{\beta}}$에 활용합니다.
    - ❖ η: learning rate (얼마나 나아갈 것인가)
    - ❖ $\frac{\partial e'e}{\partial \hat{\beta}}$: gradient (어느 방향으로 나아갈 것인가)
    - ❖ $\hat{\beta}^{(0)}$ 기반으로 $\frac{\partial e'e}{\partial \hat{\beta}}$ 계산하여 $\hat{\beta}^{(1)}$ 업데이트
    - ❖ $\hat{\beta}^{(1)}$ 기반으로 $\frac{\partial e'e}{\partial \hat{\beta}}$ 계산하여 $\hat{\beta}^{(2)}$ 업데이트 …



Source: https://see.stanford.edu/materials/aimlcs229/cs229-notes1.pdf

Gradient Descent

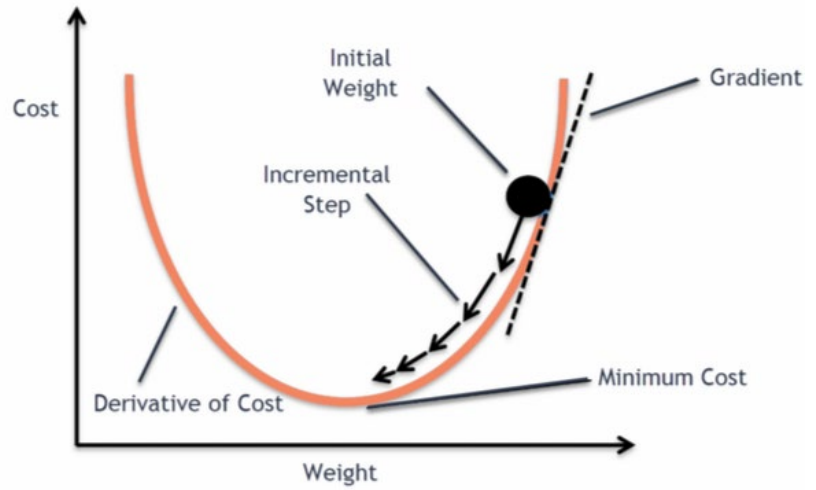
Multiple linear regression

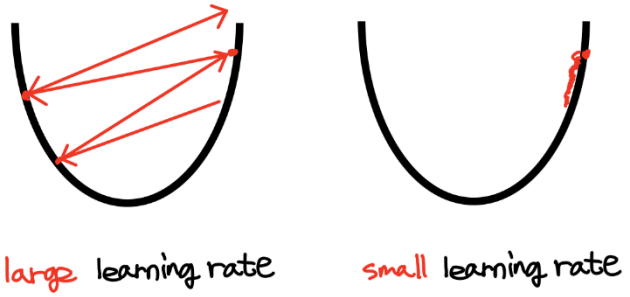
- 2) 계산하기는 간편할까요?
  - 그럼 어떻게 해야할까요? Gradient Descent!
  - $\frac{\partial e'e}{\partial \hat{\beta}} = -2X'Y + 2X'X\hat{\beta}$
  - $\hat{\beta}^{(0)} = 0.0$ (초기값 설정)
  - $\hat{\beta}^{(t+1)} = \hat{\beta}^{(t)} - \eta \frac{\partial e'e}{\partial \hat{\beta}}$에 활용합니다.
    - η: learning rate (얼마나 나아갈 것인가)
    - $\frac{\partial e'e}{\partial \hat{\beta}}$: gradient (어느 방향으로 나아갈 것인가)
    - $\hat{\beta}^{(0)}$ 기반으로 $\frac{\partial e'e}{\partial \hat{\beta}}$ 계산하여 $\hat{\beta}^{(1)}$ 업데이트
    - $\hat{\beta}^{(1)}$ 기반으로 $\frac{\partial e'e}{\partial \hat{\beta}}$ 계산하여 $\hat{\beta}^{(2)}$ 업데이트 …

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_0 & x_{11} & x_{12} & \cdots & x_{1k} \\ x_0 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ x_0 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

$n \times k$





Gradient Descent를 쓰면, exact solution($\frac{\partial e'e}{\partial \hat{\beta}} = 0$)이 존재하지 않는 경우에도 최적화가 가능하겠죠?!

Source: https://icim.nims.re.kr/post/easyMath/70 https://blog.clairvoyantsoft.com/the-ascent-of-gradient-descent-23356390836f

Logistic Regression

- Machine Learning: data를 기반으로 최적화
  - 정말로 Machine이 Learning 하는 것으로 해석할 수 있겠죠?
  - 앞에서 우리가 살펴본 Linear Regression도 Machine Learning의 일종이라고 볼 수 있겠죠?
    - ❖주어진 data에 대해서 에러를 $(e'e)$ 가장 최소화 할 수 있는 방향으로 $\hat{\beta}$가 학습됨

- 앞에서 살펴본 Linear Regression은 어떠한 경우에 활용 가능할까요?
  - 정말 말 그대로 regression을 하고 싶은 경우
  - 즉, x가 주어졌을 때, 실수 y를 예측하는 경우
  - 예시) 강수량 예측 등

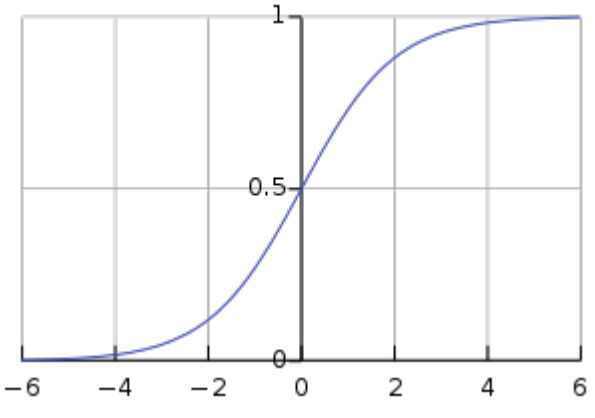
- 그렇다면, 이러한 문제는 어떻게 풀어야 할까요? (Classification, 분류)
  - 예시) 공장에서 제품을 생산하는데, 각 제품이 정상인지 비정상인지 판단
  - 예시) 카드사용기록을 통해서, 도난된 카드의 사용이력인지, 아니면 원 주인의 정상적인 사용이력인지 여부 판단
  - ⇒ Logistic Regression

Source:

Logistic Function

Logistic Regression

- Logistic regression으로 들어가기전에, logistic function에 대해서 먼저 살펴보겠습니다.
  - Logistic function: 쉽게 생각해서, input으로는 $-\infty$부터 ∞까지 받고, output으로는 0과 L 사이의 값을 도출.
  - Logistic function: $f(x) = \frac{L}{1+e^{-k(x-x_0)}}$
    - ❖x_0: midpoint
    - ❖L: curve's maximum value
    - ❖k: steepness of the curve
  - 좀 더 쉬운 예시를 살펴보겠습니다.
    - ❖Standard logistic function, $f(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x}$
      - ➢ Sigmoid function 으로도 불립니다.
    - ❖x_0가 0이고, L이 1인 경우입니다.
    - ❖즉 중간지점이 0이고, 최대치는 1인 경우입니다.
    - ❖Classification (분류)에 딱 알맞지 않나요?!
    - ❖1에 가까우면 정상, 0에 가까우면 비정상등으로 분류 할 수 있겠죠?



Source: https://en.wikipedia.org/wiki/Logistic_function

Logistic Regression

Logistic Regression

- 일단 우리는 $Y = 0$ 또는 $Y = 1$
  - Binary classification
  - Note) Multi-class classification: $Y = 0,1,2,3, \ldots$
    - ❖예제) 주어진 식물의 품종 분류하기

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} x_0 & x_{11} & x_{12} & \cdots & x_{1k} \\ x_0 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ x_0 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

$n \times k$

- Linear Regression에서는
  - $y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$
  - $y = \vec{x}\beta + \epsilon$
  - $\hat{y} = \vec{x}\hat{\beta}$
  - 여기서 $\vec{x}, \beta$ 등에 대해서는 아무런 제약이 없습니다.
  - 즉, y는 0과 1사이뿐만 아니라, 그 어떤값이든 될 수 있습니다.
  - ⇒ Classification에 적용하기는 쉽지 않음

편의상 $n = 1$, 즉 데이터가 1개만 있다고 가정하겠습니다.
즉, $\vec{x} = [x_0, x_1, \ldots, x_k]$

- Logistic Regression에서는
  - $\hat{y} = P(y = 1|\vec{x})$을 구하고 싶은 것
    - ❖즉, 제품이 정상일 확률 (또는 비정상일 확률)을 알 수 있다면, 분류 가능
    - ❖$P(y = 1|\vec{x}) = \vec{x}\hat{\beta}$라고 할 수 있을까요?
    - ❖RHS (우변)은 굉장히 큰 양수도 나올수가 있습니다.
    - ❖반면 LHS (좌변)은 0과 1사이의 값입니다. 좌변과 우변의 scale이 맞지 않죠?

우리가 배운 조건부확률이 나오죠?

0과1사이?! 어 이런거 배웠는데?

Source: https://ratsgo.github.io/machine%20learning/2017/04/02/logistic/

Logistic Regression

- Logistic Regression에서는
  - $\hat{y} = P(y = 1|\vec{x})$을 구하고 싶은 것
    - ❖즉, 제품이 정상일 확률 (또는 비정상일 확률)을 알 수 있다면, 분류 가능
    - ❖$P(y = 1|\vec{x}) = \vec{x}\hat{\beta}$라고 할 수 있을까요?
    - ❖RHS (우변)은 음수와 양수 모두 나올수가 있습니다. 좌변과 우변의 scale이 맞지 않죠?
  - 이번에는 비율로 생각해보겠습니다.
    - ❖$\frac{P(y=1|\vec{x})}{P(y=0|\vec{x})} = \frac{P(y=1|\vec{x})}{1-P(y=1|\vec{x})} = \vec{x}\hat{\beta}$ 로 하면 어떨까요?
      - ➢ 일단, $\frac{P(y=1|\vec{x})}{P(y=0|\vec{x})}$를 우리는 odds 라고 부릅니다.
      - ➢ 분모: 일어나지 않을 확률, 분자: 일어날 확률
    - ❖RHS와 마찬가지로, LHS도 매우 큰 양수도 표현할 수 있게 되었습니다.
    - ❖하지만, LHS는 음수가 아니지만, RHS는 여전히 음수가 될 수 있습니다.
    - ❖그렇다면 이번에는 좌변에 log를 적용볼까요?
  - $\log\frac{P(y=1|\vec{x})}{1-P(y=1|\vec{x})} = \vec{x}\hat{\beta}$
    - ❖이제는 좌변과 우변 모두 $-\infty$부터 ∞를 표현할 수 있게 되었습니다.
    - ❖우리가 구하고 싶은 것은 무엇이라고 했죠? $P(y = 1|\vec{x})$

Source: https://ratsgo.github.io/machine%20learning/2017/04/02/logistic/

Logistic Regression

Logistic Regression

- Logistic Regression에서는
  - $\hat{y} = P(y = 1|\vec{x})$을 구하고 싶은 것
  - $\log \frac{P(y=1|\vec{x})}{1-P(y=1|\vec{x})} = \vec{x}\hat{\beta}$
    - ❖이제는 좌변과 우변 모두 $-\infty$부터 ∞를 표현할 수 있게 되었습니다.
    - ❖우리가 구하고 싶은 것은 무엇이라고 했죠? $P(y = 1|\vec{x})$
  - $\frac{P(y=1|\vec{x})}{1-P(y=1|\vec{x})} = \exp(\vec{x}\hat{\beta})$
  - $P(y = 1|\vec{x}) = \exp(\vec{x}\hat{\beta})\left(1 - P(y = 1|\vec{x})\right)$
  - $P(y = 1|\vec{x})\left(1 + \exp(\vec{x}\hat{\beta})\right) = \exp(\vec{x}\hat{\beta})$
  - $P(y = 1|\vec{x}) = \frac{\exp(\vec{x}\hat{\beta})}{1+\exp(\vec{x}\hat{\beta})}$

이 과정은, 단순 계산이죠?!

- 앞에서 배운 logistic function 기억나시나요?
  - Standard logistic function, $f(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x}$
  - 왜 이름이 logistic regression인지 감이 잡히시나요?

Source: https://ratsgo.github.io/machine%20learning/2017/04/02/logistic/

Learning

Logistic Regression

- 자 그러면, 남은게 무엇일까요? 학습이겠죠?
  - $P(y=1|\vec{x}) = \frac{\exp(\vec{x}\hat{\beta})}{1+\exp(\vec{x}\hat{\beta})}$
  - 우리가 학습해야 하는것: $\hat{\boldsymbol{\beta}}$
  - 우리에게 주어진 것: training dataset, test dataset
  - Training dataset: $(\vec{x}_{tr}, y_{tr})$
  - Test dataset: $(\vec{x}_{te})$
- 이러한 상황을 **Supervised Learning** 이라고 합니다.
  - 문제집 (Training)의 문제 $(\vec{x}_{tr})$에 답지가 (y_{tr}) 주어졌고, 그것으로 공부를 열심히 해서, 본 시험 (Test)의 문제 $(\vec{x}_{te})$를 잘 푸는것을 목표로 함
  - 즉, 우리는 $\vec{x}_{tr}$과 y_{tr}을 가지고, $\hat{\beta}$를 최적화해야 합니다.
  - 그리고 그러한 $\hat{\beta}$가 $\vec{x}_{te}$에 대해서도 잘 하기를 희망합니다. (그 과정에서 $\vec{x}_{te}$는 미리 알 수 없습니다. 수능문제를 미리 알고, 문제집으로 공부할 수 없는것처럼…)
  - Note) **Unsupervised Learning**: y_{tr}이 없는 상황
    - ❖즉, $\vec{x}_{tr}$만 가지고 열심히 공부해서, $\vec{x}_{te}$를 잘 푸는것을 목표로 함
  - Note) **Semi-supervised Learning**: y_{tr}이 일부만 있는 상황
    - ❖즉, 문제집의 모든 문제에 대해 답이 있는것은 아니고, 일부만 있는 상황이며, 문제집으로 공부를 열심히 해서 본 시험의 문제를 잘 푸는것을 목표로 함

Source: https://web.stanford.edu/~jurafsky/slp3/5.pdf

Learning

Logistic Regression

- 학습을 위해서는, 목적식이 필요합니다.
  - 비유) 문제집을 공부할 때 목적식은, 문제집의 답을 다 맞추는 것
- Linear Regression의 경우에는
  - Mean Squared Error
  - $(y - \hat{y})^2$
  - 정답지 y와 우리의 예측치 $\hat{y}$ 차이의 제곱을 최소화 (MSE)
- Logistic Regression의 경우에는
  - Cross-Entropy Loss (CE)
  - $L_{CE} = -(y \log \hat{y} + (1 - y) \log(1 - \hat{y}))$
    - ❖$\hat{y} = P(y = 1|\vec{x}) = \frac{\exp(\vec{x}\hat{\beta})}{1+\exp(\vec{x}\hat{\beta})}$
  - CE는 어떻게 작동할까요?
    - ❖L_{CE}가 낮아지려면, y가 1일 때 $\hat{y}$도 높은 값을 가져야 함 (1에 가까운 값)
    - ❖L_{CE}가 낮아지려면, y가 0일 때 $\hat{y}$도 낮은 값을 가져야 함 (0에 가까운 값)
  - 우리가 학습해야하는 것: $\hat{\beta}$
  - $\hat{\beta}$ 는 L_{CE}를 최소화하도록 최적화 되어야 한다.
  - ⇒ Gradient Descent!

Source: https://web.stanford.edu/~jurafsky/slp3/5.pdf

Learning

Logistic Regression

- $L_{CE} = -(y \log \hat{y} + (1-y) \log(1-\hat{y})$
  - $\hat{y} = P(y=1|\vec{x}) = \frac{\exp(\vec{x}\hat{\beta})}{1+\exp(\vec{x}\hat{\beta})}$
  - $\hat{\beta}$ 는 L_{CE}를 최소화하도록 최적화 되어야 한다.
  - ⇒ Gradient Descent!
- $\frac{\partial L_{CE}}{\partial \hat{\beta}}$를 계산해야 하는데… 그 전에 몇가지 짚고 넘어가겠습니다.
  - $\frac{\exp(x)}{1+\exp(x)}$ 를 $\sigma(x)$라고 편의상 적겠습니다. (sigmoid를 나타내는 기호로 많이 활용)

    ❖$\sigma(x)$를 x에 대해서 미분하면 어떻게 될까요?

    ❖$\frac{\partial \sigma(x)}{\partial x} = \frac{\exp(x)(1+\exp(x)) - \exp(x)\exp(x)}{(1+\exp(x))^2} = \frac{\exp(x)}{(1+\exp(x))^2} = \frac{1}{1+\exp(x)} \frac{\exp(x)}{1+\exp(x)} = (1-\sigma(x))\sigma(x)$
  - $\hat{y} = P(y=1|\vec{x}) = \frac{\exp(\vec{x}\hat{\beta})}{1+\exp(\vec{x}\hat{\beta})}$를 $\sigma(\vec{x}\hat{\beta})$라고 편의상 적겠습니다.

    ❖$\sigma(\vec{x}\hat{\beta})$를 $\hat{\beta}$에 대해서 미분하면 어떻게 될까요? $\vec{x}\hat{\beta} = t$ 라고 하겠습니다.

    ❖$\frac{\partial \sigma(\vec{x}\hat{\beta})}{\partial \hat{\beta}} = \frac{\partial t}{\partial \hat{\beta}} \frac{\partial \sigma(t)}{\partial t} = \vec{x}^T(1-\sigma(t))\sigma(t) = \vec{x}^T\left(1-\sigma(\vec{x}\hat{\beta})\right)\sigma(\vec{x}\hat{\beta})$
- 그럼 이제, 정말 우리가 원하는 $\frac{\partial L_{CE}}{\partial \hat{\beta}}$를 계산해보겠습니다.
  - $L_{CE} = -(y \log \hat{y} + (1-y) \log(1-\hat{y})$

Source: https://web.stanford.edu/~jurafsky/slp3/5.pdf

Learning

Logistic Regression

- 그럼 이제, 정말 우리가 원하는 $\frac{\partial L_{CE}}{\partial \hat{\beta}}$를 계산해보겠습니다.
  - $L_{CE} = -(y \log \hat{y} + (1-y) \log(1-\hat{y})$
  - $L_{CE} = -(y \log \sigma(\vec{x}\hat{\beta}) + (1-y) \log(1-\sigma(\vec{x}\hat{\beta}))$

  - $\frac{\partial L_{CE}}{\partial \hat{\beta}} = \frac{\partial \hat{y}}{\partial \hat{\beta}} \frac{\partial L_{CE}}{\partial \hat{y}} = \frac{\partial \sigma(\vec{x}\hat{\beta})}{\partial \hat{\beta}} \frac{\partial L_{CE}}{\partial \hat{y}} = \vec{x}^T \left(1-\sigma(\vec{x}\hat{\beta})\right) \sigma(\vec{x}\hat{\beta}) \frac{\partial L_{CE}}{\partial \hat{y}}$
    - ❖ $\frac{\partial L_{CE}}{\partial \hat{y}} = -\left(y \frac{1}{\hat{y}} - (1-y) \frac{1}{1-\hat{y}}\right) = -\left(y \frac{1}{\sigma(\vec{x}\hat{\beta})} - (1-y) \frac{1}{1-\sigma(\vec{x}\hat{\beta})}\right) = -\left(\frac{y-\sigma(\vec{x}\hat{\beta})}{\sigma(\vec{x}\hat{\beta})(1-\sigma(\vec{x}\hat{\beta}))}\right)$
  - $\frac{\partial L_{CE}}{\partial \hat{\beta}} = \vec{x}^T \left(1-\sigma(\vec{x}\hat{\beta})\right) \sigma(\vec{x}\hat{\beta}) \times -\left(\frac{y-\sigma(\vec{x}\hat{\beta})}{\sigma(\vec{x}\hat{\beta})(1-\sigma(\vec{x}\hat{\beta}))}\right)$
  - $\frac{\partial L_{CE}}{\partial \hat{\beta}} = \vec{x}^T \left(\sigma(\vec{x}\hat{\beta}) - y\right)$
- 이후에는 앞 linear regression에서의 gradient descent와 동일합니다.
  - $\hat{\beta}^{(0)} = 0.0$ (초기값 설정)
  - $\hat{\beta}^{(t+1)} = \hat{\beta}^{(t)} - \eta \frac{\partial L_{CE}}{\partial \hat{\beta}}$
    - ❖ $\hat{\beta}^{(0)}$ 기반으로 $\frac{\partial L_{CE}}{\partial \hat{\beta}}$ 계산하여 $\hat{\beta}^{(1)}$ 업데이트
    - ❖ $\hat{\beta}^{(1)}$ 기반으로 $\frac{\partial L_{CE}}{\partial \hat{\beta}}$ 계산하여 $\hat{\beta}^{(2)}$ 업데이트 …

Source:

Testing

Logistic Regression

- 아하 교수님. Logistic Regression 이제 알겠습니다.
  - Logistic Regression은 다음과 같이 모델링 되는거였고
    - ❖$P(y=1|\vec{x}) = \frac{\exp(\vec{x}\hat{\beta})}{1+\exp(\vec{x}\hat{\beta})}$
  - 이를 학습하기 위해서 CE loss를 사용했고
    - ❖$L_{CE} = -(y\log\hat{y} + (1-y)\log(1-\hat{y})$ where $\hat{y} = P(y=1|\vec{x}) = \frac{\exp(\vec{x}\hat{\beta})}{1+\exp(\vec{x}\hat{\beta})}$
  - CE loss로 실제로 $\hat{\beta}$를 최적화 할 때는 아래와 같이 했고
    - ❖$\frac{\partial L_{CE}}{\partial \hat{\beta}} = \vec{x}^T(\sigma(\vec{x}\hat{\beta}) - y)$
    - ❖Gradient Descent 활용
- 그럼 Test는 어떻게 하나요?
  - 즉, 최적화가 완료된 $\hat{\beta}$가 있다고 할 때, 새로운 data x_{te}에 대해서 y값은 어떻게 예측하나요?
  - $P(y=1|\vec{x}) = \frac{\exp(\vec{x}\hat{\beta})}{1+\exp(\vec{x}\hat{\beta})}$ 이것을 기억하시면 됩니다. 이게 0.5보다 크면 1, 그렇지 않으면 0으로 분류하게 됩니다.
  - $\vec{x}\hat{\beta} = 0$일때는 $P(y=1|\vec{x}) = 0.5$, $\vec{x}\hat{\beta} > 0$이면 $P(y=1|\vec{x}) > 0.5$, $\vec{x}\hat{\beta} < 0$이면 $P(y=1|\vec{x}) < 0.5$가 됩니다.
  - 즉, $\vec{x}\hat{\beta}$의 값만 구해서, 0보다 큰지 작은지만 판단하면 됩니다 :)

Source:

Naïve Bayes

Naive Bayes

Naive Bayes

- 이번에는, 또 다른 방식의 분류기인 Naïve Bayes에 대해서 살펴보겠습니다.
  - Naïve Bayes는 조건부확률, 독립, Bayes Rule 기반입니다.

- 먼저 그 전에, 조건부 확률을 잠깐 다시 보고 넘어가겠습니다.
- The probability of an event will sometimes depend upon other things
  - Event A: The person will contract lung cancer
  - Event B: The person is a smoker
  - Suppose we know event B is true, what is the probability of A, given B?
    - ❖**Conditional probability**
    - ❖$P(A|B) = \frac{P(A \cap B)}{P(B)}$ provided $P(B) > 0$
- Example
  - For a balanced die tossing, A: Observe a 1, B: Observe an odd number
  - $P(A) = \frac{1}{6}$ and $P(B) = \frac{3}{6}$
  - $P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{1}{3}$
  - $P(B|A) = \frac{P(A \cap B)}{P(A)} = 1$

Source:

Naive Bayes

Naive Bayes

- 조건부 확률 봤으니, 이제, 독립도 한번만 더 복습하고 넘어가겠습니다.
- Two events A and B might not be dependent
- Two events A and B are said to be **independent** if any one of the following holds
  - $P(A|B) = P(A)$
  - $P(B|A) = P(B)$
  - $P(A \cap B) = P(A)P(B)$
- Example
  - For a single die tossing, A: Observe an odd number, B: Observe an even number, C: Observe a 1 or 2
  - A and B are dependent
    - ❖$P(A|B) \neq P(A)$
  - A and C are independent
    - ❖$P(A|C) = P(A)$

또 다른 표현: $P(A|B,C) = P(A|C)$
뒤에서 다시 나옵니다. 꼭 기억해주세요!

- **Conditional Independent**
  - A and B are conditionally independent given C if and only if $P(A \cap B|C) = P(A|C)P(B|C)$

Source:

Naive Bayes

Naive Bayes

- 이제 마지막으로, Bayes' Rule을 살펴보도록 하겠습니다.
- **Partition**
  - For some positive integer k, let the sets $B_1, \dots, B_k$ be such that, $S = B_1 \cup B_2 \dots \cup B_k$ and $B_i \cap B_j = \emptyset$ for $i \neq j$. Then the collection of sets $\{B_1, \dots, B_k\}$ is said to be a partition of S
- **The Law of Total Probability**
  - $\{B_1, \dots, B_k\}$ is a partition of S such that
  - $P(B_i) > 0$ for $i = 1,2, \dots, k$.
  - Then for any event A, $P(A) = \sum_{i=1}^{k} P(A|B_i)P(B_i)$
- **Bayes' Rule (useful for reversing)**
  - $P(B_j|A) = \frac{P(B_j \cap A)}{P(A)}$ (by conditional probability)

$$= \frac{P(B_j \cap A)}{\sum_{i=1}^{k} P(A|B_i)P(B_i)} \text{ (by the law of total probability)}$$

- Corollary) $P(B|A) = \frac{P(B \cap A)}{P(A)} = \frac{P(B \cap A)}{P(A \cap B) + P(A \cap \bar{B})} = \frac{P(B \cap A)}{P(A|B)P(B) + P(A|\bar{B})P(\bar{B})}$

Source:

Naive Bayes

Naive Bayes

- 이제 본론으로 들어와서, Naïve Bayes에 대해서 살펴보겠습니다.
  - Naïve: 순진해 빠진, 순진한

여러분들은
Independent Variable
또는 독립변수로
더 익숙하실 것 같네요.

- 결국 우리가 하고 싶은것은 무엇인가요?
  - Data input $\vec{x}$가 주어졌을 때, 해당 data에 대한 class를 예측하는 것
    - ❖$\vec{x} = (x_1, \dots, x_n)$ 으로 표현 가능하며, 총 n개의 feature가 존재
    - ❖주어진 $\vec{x}$의 class가 y_1일지, y_2일지, …, y_K일지 추정 (총 K개의 선택지 중, 한가지를 고르는 것)
  - 즉, 우리가 구하고 싶은 것을 수식화 하면
    - ❖$p(y_k|\vec{x})$를 구하고 싶은 것
    - ❖$\vec{x}$가 주어졌을 때, 해당 데이터의 class가 y_k일 확률 값.
    - ❖이러한 확률값을 y_1 부터 y_K 까지 모두 구할 수 있다면, 가장 큰 확률 값을 가지는 class로 선택하면 되겠죠?

소문자 k는 index를,
대문자 K는 class 개수를 의미

Class가 무엇인지는 기억나시죠?
분류 할 때의 정답 후보 군이라고 생각하시면 되겠습니다.
예를 들어, 주어진 이미지가 강아지인지, 고양이인지 맞추고자 하면, class는 강아지, 고양이 2개 입니다.

  - 조건부 확률, $p(y_k|\vec{x})$를 구하자!

Source: https://en.dict.naver.com/#/entry/enko/a26d050d60b04d2a9e08f8773303cba0

Naive Bayes

Naive Bayes

- 조건부 확률, $p(y_k|\vec{x})$를 구하자!
  - 다시 쓰면, $p(y_k|x_1, \dots, x_n)$을 구하자!
  - 우리가 왜 조건부 확률을 그동안 배웠는지 아시겠죠?

- $p(y_k|\vec{x})$ 을 우리가 배운 Bayes Rule 로 접근해보겠습니다.
  - $p(y_k|\vec{x}) = \frac{p(\vec{x}|y_k)p(y_k)}{p(\vec{x})}$ 가 됩니다.

우리가 왜 Bayes Rule을 배웠는지 아시겠죠?!

    - ❖$p(\vec{x}|y_k)$: Likelihood
    - ❖$p(y_k)$: Prior
    - ❖$p(\vec{x})$: Evidence
  - 여기서 우리의 목표는 "분류" 자체이며, $p(\vec{x})$는 신경쓰지 않겠습니다.
    - ❖나중에 머신러닝을 깊게 배우면, 매우 중요해집니다 :)
  - 즉, $p(y_k|\vec{x}) \propto p(\vec{x}|y_k)p(y_k)$

- 자 그럼 우리가 이제 우리가 구해야 하는 것은, 다음 2개 입니다.
  - $p(\vec{x}|y_k)$
  - $p(y_k)$

Source: https://en.dict.naver.com/#/entry/enko/a26d050d60b04d2a9e08f8773303cba0

Naive Bayes

Naive Bayes

- 우리가 구하고 싶은 것: $p(y_k|\vec{x})$
  - $p(y_k|\vec{x}) = \frac{p(\vec{x}|y_k)p(y_k)}{p(\vec{x})}$
  - $p(y_k|\vec{x}) \propto p(\vec{x}|y_k)p(y_k)$
- 자, 그런데 $\vec{x}$가 무엇이라고 했죠?
  - $\vec{x} = (x_1, \dots, x_n)$
  - 예시)
    - ❖Task: 학점 예측
    - ❖Class: A+, A, B+, B, C+, C, D+, D, F
      - ➢ y_1: A+, y_2: A, …
    - ❖Feature: 중간고사 점수, 기말고사 점수, 과제, 출석
      - ➢ x_1: 중간고사 점수, x_2: 기말고사 점수, …

- 즉, 우리가 진짜로 구해야 하는 것은
  - $p(x_1, \dots, x_n|y_k)p(y_k)$
  - 두개 중에, 무엇이 더 구하기 어려워보이나요?

Source:

Naive Bayes

- 즉, 우리가 진짜로 구해야 하는 것은
  - $p(x_1, \dots, x_n|y_k)p(y_k)$
  - 두개 중에, 무엇이 더 구하기 어려워보이나요?
- $p(x_1, \dots, x_n|y_k)$ 를 다시 써보겠습니다.
  - 그 전에, 쉬운것 부터 보고 가죠.
  - $p(x_1, x_2, x_3\ |y_k) = \frac{p(x_1,x_2,x_3,y_k)}{p(y_k)}$

$$= \frac{p(x_1|x_2,x_3,y_k)p(x_2,x_3,y_k)}{p(y_k)}$$

$$= \frac{p(x_1|x_2,x_3,y_k)p(x_2|x_3,y_k)p(x_3,y_k)}{p(y_k)}$$

$$= \frac{p(x_1|x_2,x_3,y_k)p(x_2|x_3,y_k)p(x_3|y_k)p(y_k)}{p(y_k)}$$

$$= p(x_1|x_2,x_3,y_k)p(x_2|x_3,y_k)p(x_3|y_k)$$

규칙성이
보이지 않나요?

- 규칙성을 찾으셨나요?
  - $p(x_1, \dots, x_n|y_k)$도 그럼 우리가 위와 같이 써볼 수 있겠죠?
  - $p(x_1|x_2, \dots, x_n, y_k)p(x_2|x_3, \dots, x_n, y_k) \dots p(x_n|y_k)$
  - 주어진 형태가, 비교적 복잡하지 않나요? 간단히 할 수 있는 방법 없을까요?

Source:

Naive Bayes

Naive Bayes

- 즉, 우리가 진짜로 구해야 하는 것은
  - $p(x_1, \dots, x_n|y_k)p(y_k)$
- $p(x_1, \dots, x_n|y_k)$
  - $p(x_1|x_2, \dots, x_n, y_k)p(x_2|x_3, \dots, x_n, y_k) \dots p(x_n|y_k)$
  - Conditional Independence!
  - 만약 y_k가 주어졌을 때, $x_1, x_2, \dots, x_n$이 서로 독립이라면?
  - $p(x_1|x_2, \dots, x_n, y_k)p(x_2|x_3, \dots, x_n, y_k) \dots p(x_n|y_k)$ 는 아래와 같이 간략화 됩니다.
  - $p(x_1|y_k)p(x_2|y_k) \dots p(x_n|y_k)$
- 아니 교수님. 여기서 y_k가 주어졌을 때, $x_1, x_2, \dots, x_n$이 서로 독립이라는 것이 보장이 되나요? 왜 그런거죠?
  - 보장 안됩니다.
  - 그냥 우리는 그럴 것이라 "가정" 하는 것입니다.
  - 그래도 되냐고요?
  - 그래서 이름이 **Naïve** Bayes 입니다.

A and B are conditionally independent given C if and only if $P(A \cap B|C) = P(A|C)P(B|C)$
또 다른 표현: $P(A|B, C) = P(A|C)$

Source:

Naive Bayes

Naive Bayes

- 즉, 우리가 진짜로 구해야 하는 것은
  - $p(x_1, \dots, x_n|y_k)p(y_k)$
    - ❖$p(x_1, \dots, x_n|y_k) = p(x_1|y_k)p(x_2|y_k) \dots p(x_n|y_k)$
  - 이번에는, $p(y_k)$를 구해보겠습니다.

- $p(y_k)$
  - 사실, 구할 것이 없어요…
  - Training Set 에서 $p(y_1)$,…, $p(y_K)$를 구할 수 있겠죠?
  - 어떻게 구할까요? 그냥 count!
  - 아니면, 단순하게, $1/K$ 로 설정하기도 합니다.
  - 나중에는, 여러가지 분포로 접근 할 수도 있습니다.

- 정리하면,
  - $p(y_k|\vec{x}) \propto p(\vec{x}|y_k)p(y_k)$
  - $p(x_1|y_k)p(x_2|y_k) \dots p(x_n|y_k)p(y_k)$

Source:

Example

Naive Bayes

- 정리하면,
  - $p(y_k|\vec{x}) \propto p(\vec{x}|y_k)p(y_k)$
  - $p(x_1|y_k)p(x_2|y_k) \dots p(x_n|y_k)p(y_k)$
- Task
  - 골프를 칠 것인가, 말 것인가?
  - (전 아직 한번도 쳐본적 없지만…)
  - Class: Yes, No
- Test $\vec{x}$
  - (Rainy, Cool, High, True)
  - $p(\vec{x}|Yes)$
    - ❖$p(Rainy|Yes) = \frac{2}{9}$
    - ❖$p(Cool|Yes) = p(High|Yes) = p(True|Yes) = \frac{3}{9}$
  - $p(Yes) = \frac{9}{14}$
- $\Rightarrow p(Yes|\vec{x}) \propto \frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{9}{14} \approx 0.00529$
- $\Rightarrow p(No|\vec{x}) \approx 0.02057$ (유사하게 구해보세요)

Training Set

Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Test Set

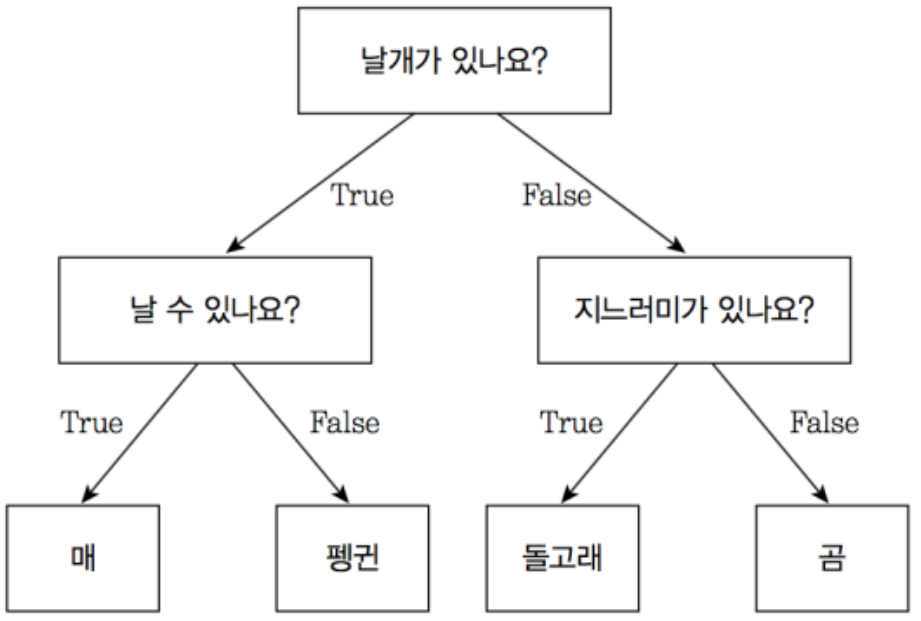
Rainy, Cool, High, True 일 때
Play Golf 는?

Source: https://www.saedsayad.com/naive_bayesian.htm

Decision Tree

Decision Tree (Motivation)

- Logistic Regression 외에, 또 다른 Classification Algorithm 은 없을까?
  - 그냥 Attribute (input feature) 기반으로 분류할 수는 없을까?
  - 즉, 기온이 30도가 넘으면, 비가 온다고 하자.
  - 이렇게 간단한 알고리즘이 잘 작동할 수 있을까?
    - ❖조금 더 고도화가 필요하긴 합니다!
    - ❖하지만, 생각보다 괜찮습니다!



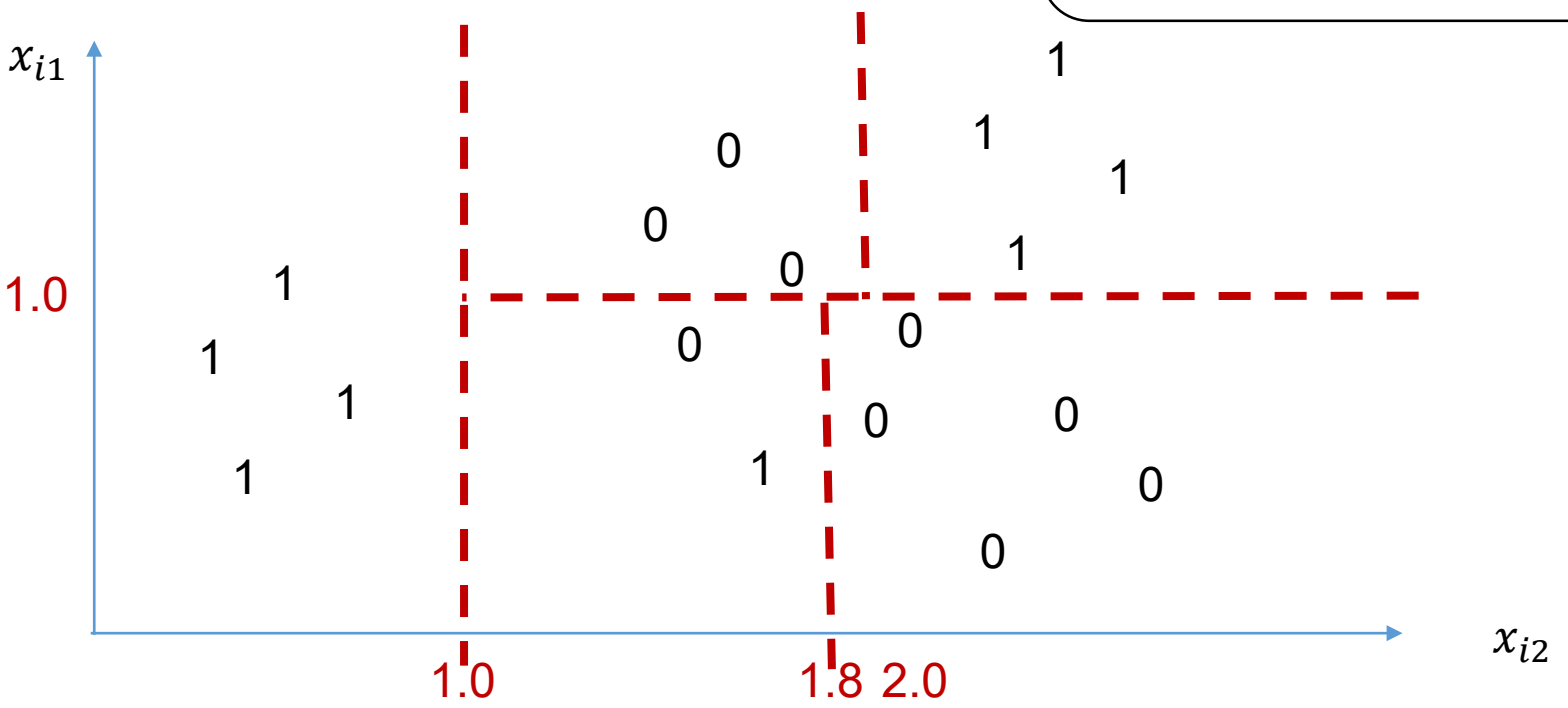
Source: https://bkshin.tistory.com/entry/%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9D-4-%EA%B2%B0%EC%A0%95-%ED%8A%B8%EB%A6%ACDecision-Tree

Decision Tree (Classification)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - $(x_1, y_1), \ldots, (x_n, y_n)$ 이 주어져있고, 이를 통해서 Tree를 구성해보자.
  - 가정1) Binary Tree 구성
  - 가정2) 각 leaf 에서, 가장 최소화된 error 를 가져와.
  - Decision Tree는 Split을 해나가는 것!

Binary Tree
각각의 node가 최대 2명의 children을 가지는 경우

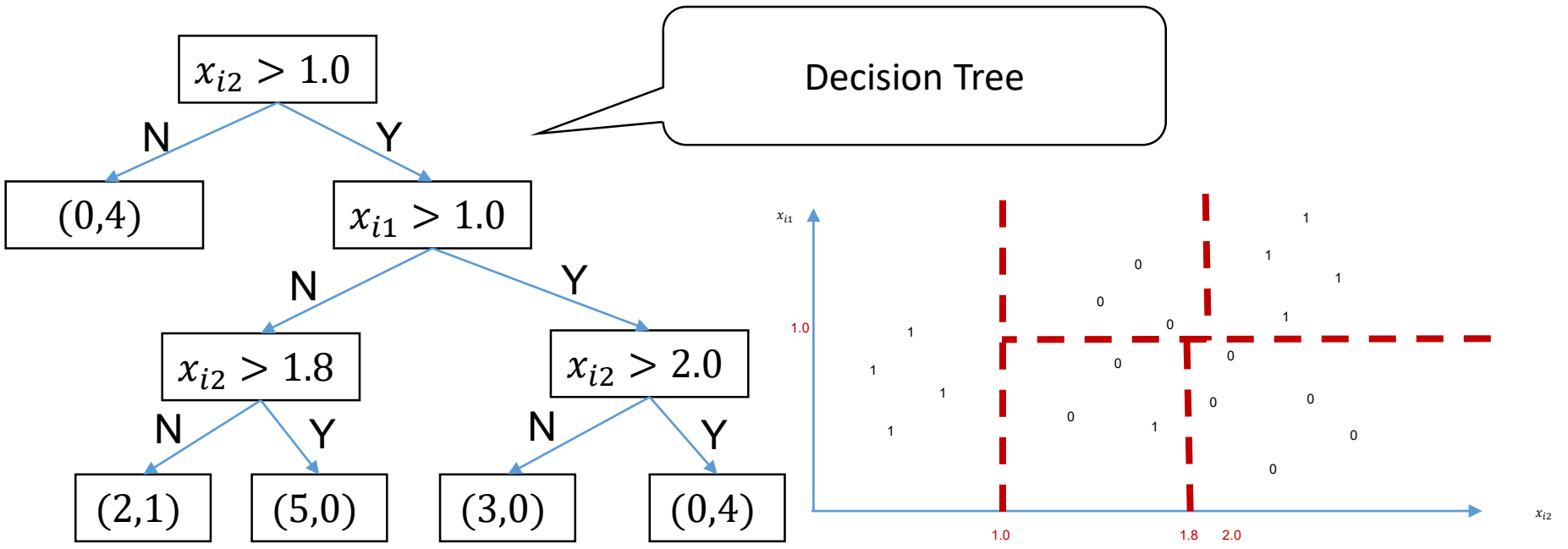


Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Classification)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - $(x_1, y_1), \ldots, (x_n, y_n)$ 이 주어져있고, 이를 통해서 Tree를 구성해보자.
  - 가정1) Binary Tree 구성
  - 가정2) 각 leaf 에서, 가장 최소화된 error 를 가지자. (Training Error)
  - Decision Tree는 Split을 해나가는 것!

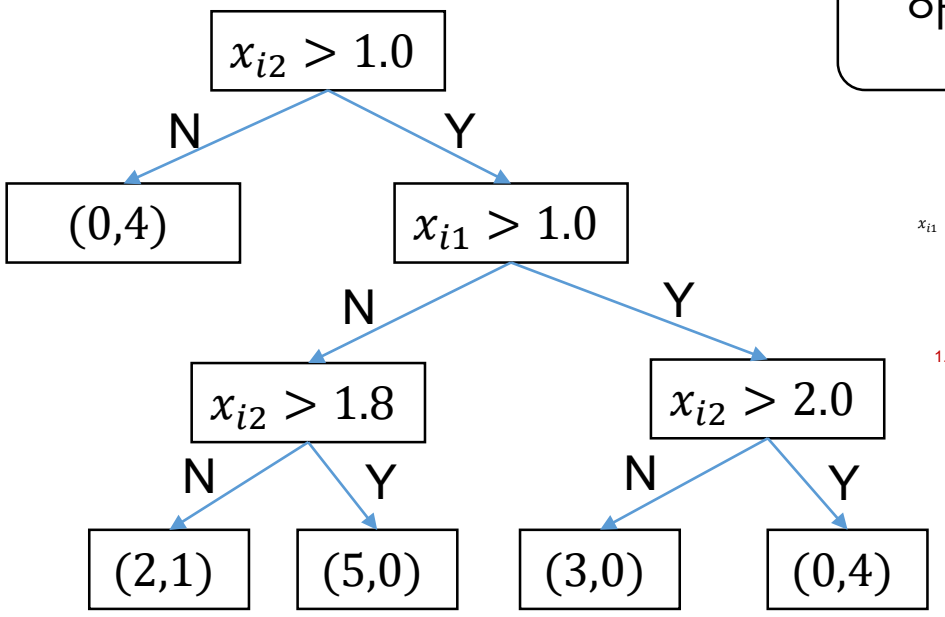


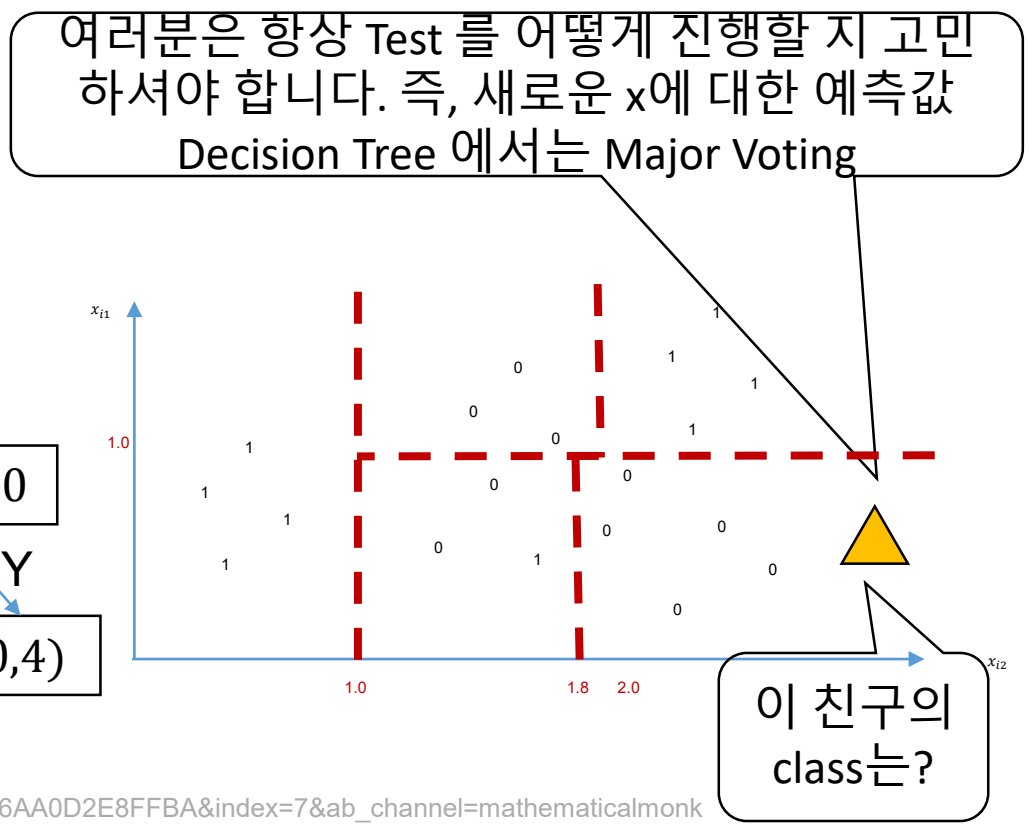
Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Classification)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - $(x_1, y_1), \ldots, (x_n, y_n)$ 이 주어져있고, 이를 통해서 Tree를 구성해보자.
  - 가정1) Binary Tree 구성
  - 가정2) 각 leaf 에서, 가장 최소화된 error 를 가지자. (Training Error)
  - Decision Tree는 Split을 해나가는 것!



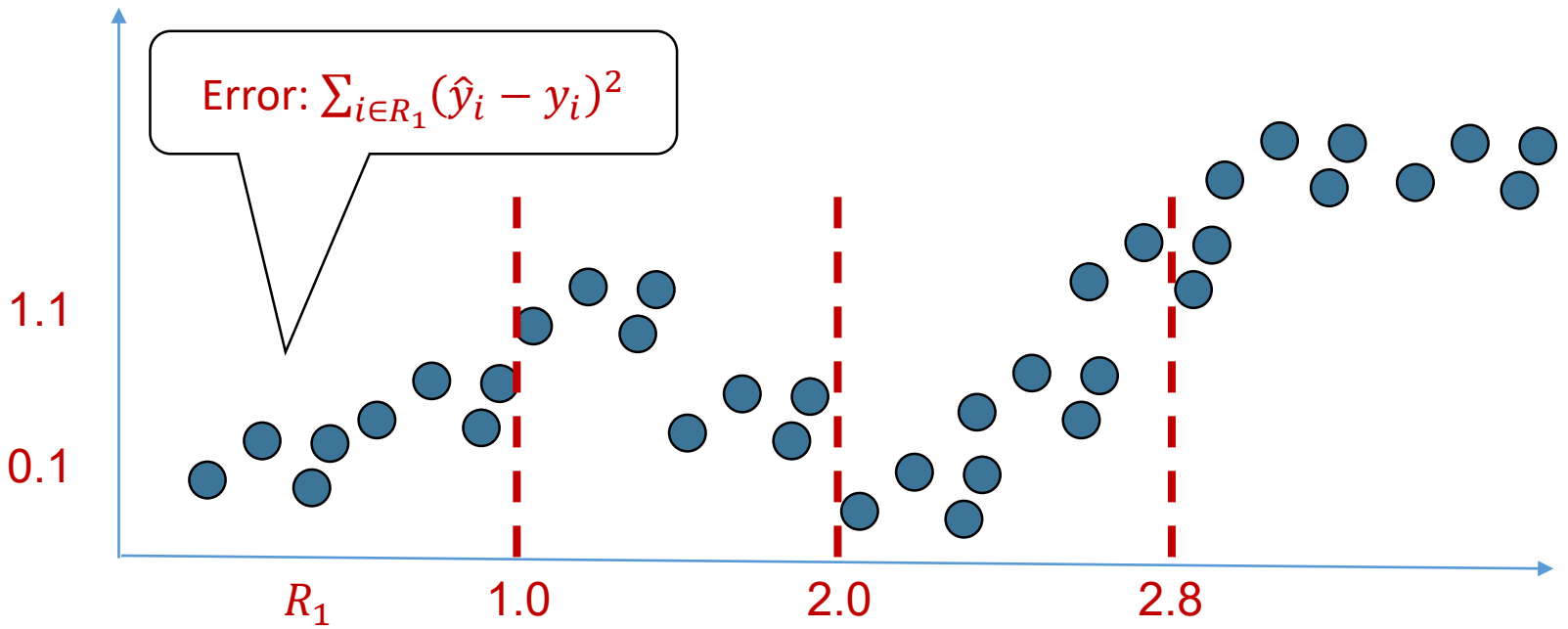


Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Regression)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - $(x_1, y_1), \ldots, (x_n, y_n)$ 이 주어져있고, 이를 통해서 Tree를 구성해보자.
  - 가정1) Binary Tree 구성
  - 가정2) 각 leaf 에서, 가장 최소화된 error 를 가지자. (Training Error)
  - Decision Tree는 Split을 해나가는 것!
- Decision Tree로 Regression 도 가능할까요?

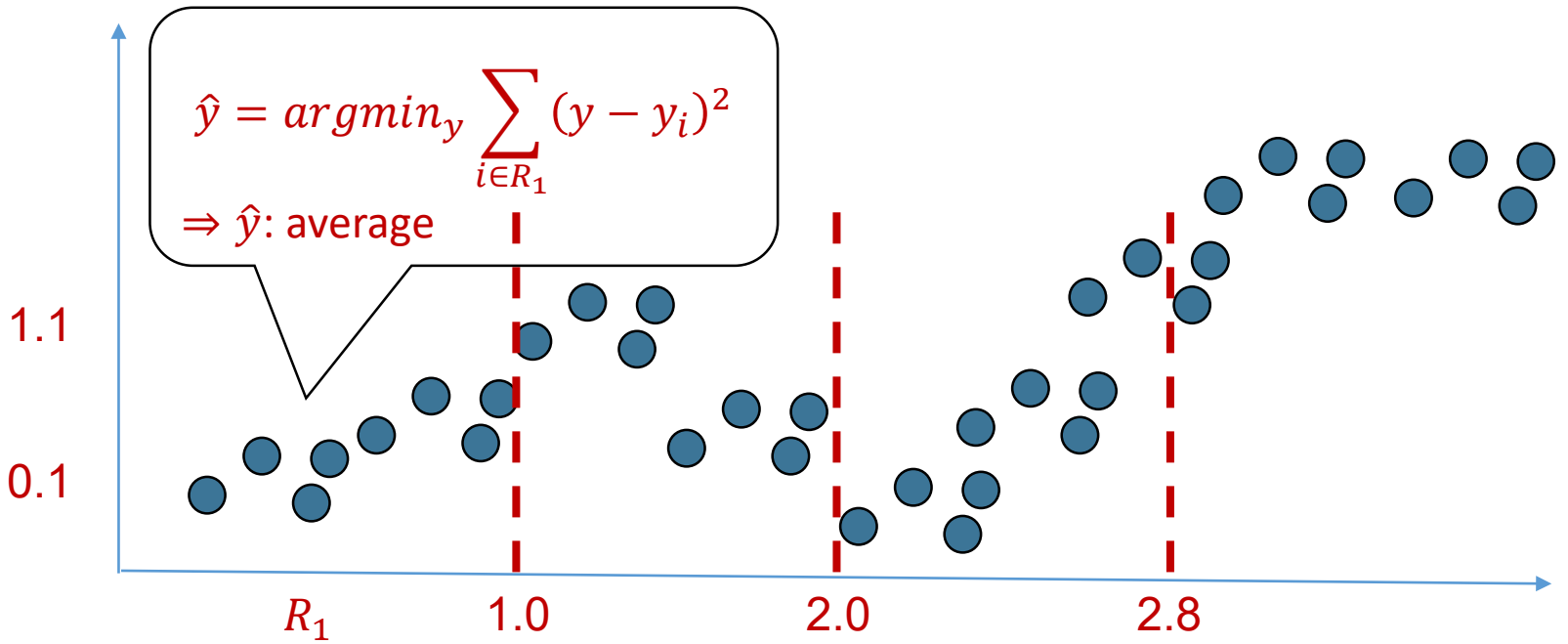


Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Regression)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - $(x_1, y_1), \ldots, (x_n, y_n)$ 이 주어져있고, 이를 통해서 Tree를 구성해보자.
  - 가정1) Binary Tree 구성
  - 가정2) 각 leaf 에서, 가장 최소화된 error 를 가지자. (Training Error)
  - Decision Tree는 Split을 해나가는 것!
- Decision Tree로 Regression 도 가능할까요?

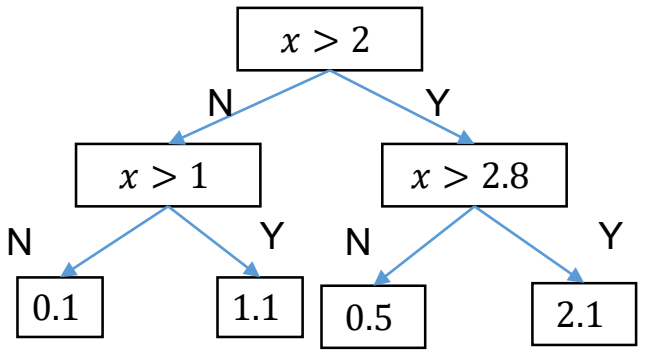


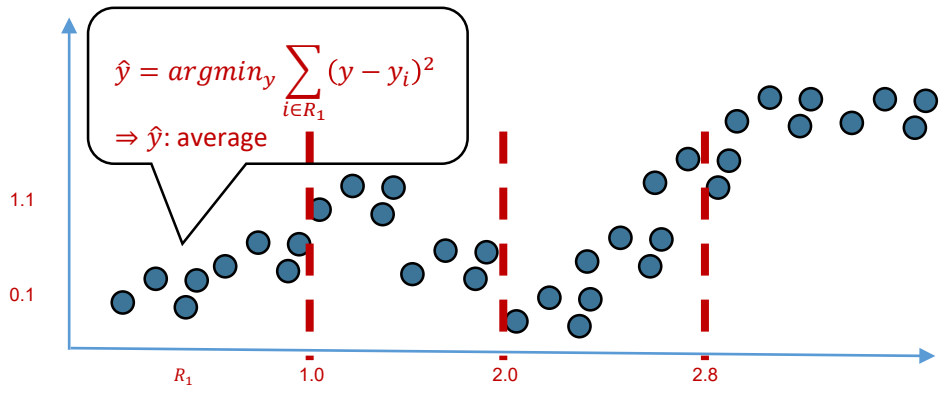
Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Regression)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - $(x_1, y_1), \ldots, (x_n, y_n)$ 이 주어져있고, 이를 통해서 Tree를 구성해보자.
  - 가정1) Binary Tree 구성
  - 가정2) 각 leaf 에서, 가장 최소화된 error 를 가지자. (Training Error)
  - Decision Tree는 Split을 해나가는 것!
- Decision Tree로 Regression 도 가능할까요?



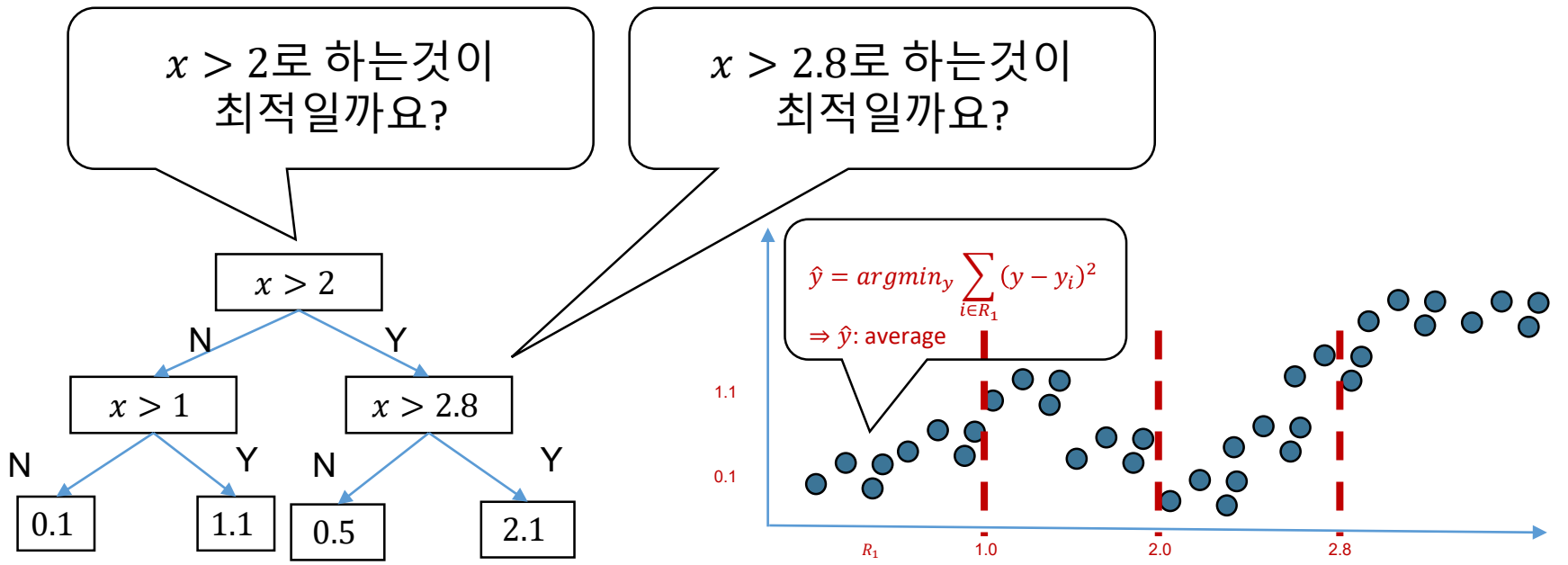


Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Growing)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - 어떻게 하면, Tree 를 효과적으로 키울 수 있을까요? (Growing Tree)
  - 우리는 Decision Tree 를 굉장히 많은 방식으로 키울 수 있습니다.
  - 즉, 기준을 어떻게 잡아야 가장 효과적인 Tree 를 만들 수 있을까요?
    - ❖일단, Regression Case 부터 살펴보겠습니다.

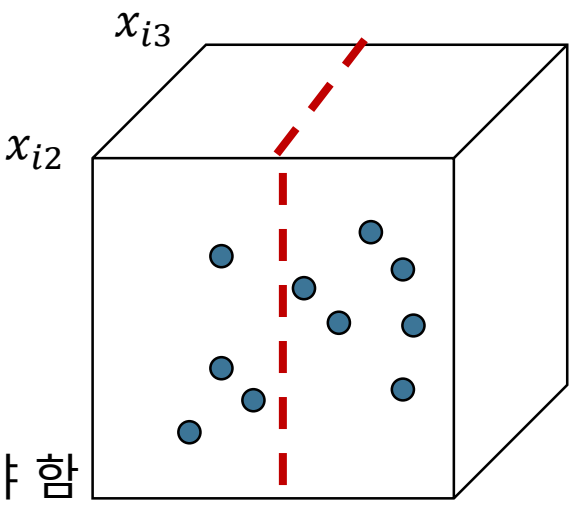


Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Growing)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - 어떻게 하면, Tree 를 효과적으로 키울 수 있을까요? (Growing Tree)
  - 우리는 Decision Tree 를 굉장히 많은 방식으로 키울 수 있습니다.
  - 즉, 기준을 어떻게 잡아야 가장 효과적인 Tree 를 만들 수 있을까요?
    - ❖Greedy 하게 접근
  - 우리가 정해야 하는 것
    - ❖축 (어떠한 feature 를 가지고 분류할 것인가)
    - ❖기준 (분류할 기준점은 어디인가)
- $x = (x_{i1}, \dots, x_{id})$
- First Split
  - $\min_y \sum_{i:x_{ij}>s}(y - y_i)^2 + \min_y \sum_{i:x_{ij}\le s}(y - y_i)^2$
  - 즉, 우리는 위 loss가 minimize 되는 j와 s를 잡아야 함



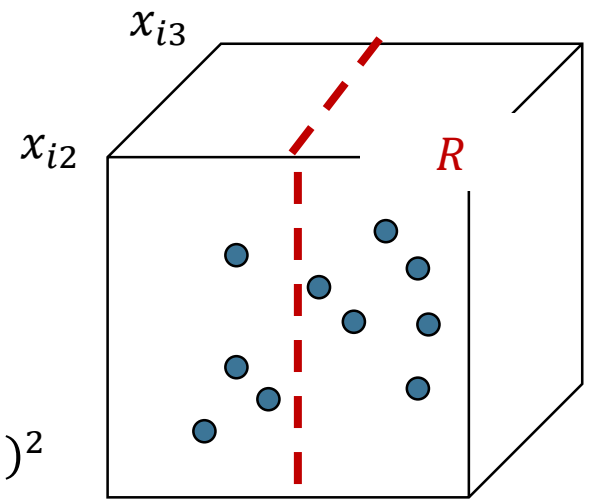
$$\begin{matrix} s \\ j = 1 \end{matrix}$$

Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Growing)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - 어떻게 하면, Tree 를 효과적으로 키울 수 있을까요? (Growing Tree)
  - 우리는 Decision Tree 를 굉장히 많은 방식으로 키울 수 있습니다.
  - 즉, 기준을 어떻게 잡아야 가장 효과적인 Tree 를 만들 수 있을까요?
    - ❖Greedy 하게 접근
  - 우리가 정해야 하는 것
    - ❖축 (어떠한 feature 를 가지고 분류할 것인가)
    - ❖기준 (분류할 기준점은 어디인가)
- $x = (x_{i1}, \dots, x_{id})$
- Second Split; First Split과 유사함
  - 단, 고려해주는 영역의 위치가 변화됨
  - $\min_y \sum_{i:x_{ij}>s, \boldsymbol{x_i \in R}} (y - y_i)^2 + \min_y \sum_{i:x_{ij} \le s, \boldsymbol{x_i \in R}} (y - y_i)^2$
  - 즉, 우리는 위 loss가 minimize 되는 j와 s를 잡아야 함
- Third Split …



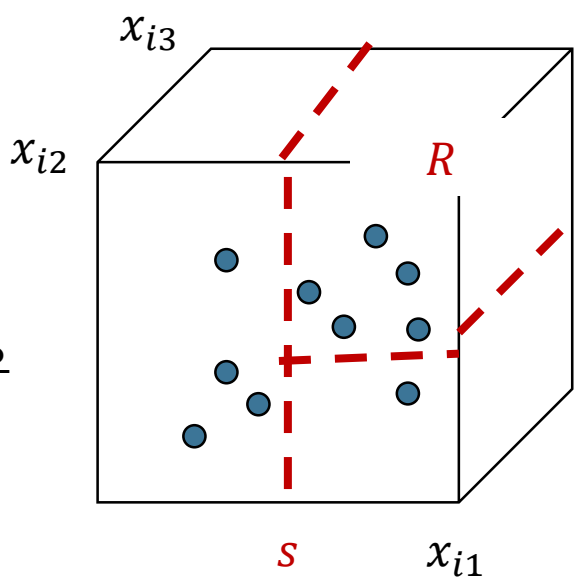
$$\begin{matrix} s \\ j = 1 \end{matrix} \qquad x_{i1}$$

Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Growing)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - 어떻게 하면, Tree 를 효과적으로 키울 수 있을까요? (Growing Tree)
  - 우리는 Decision Tree 를 굉장히 많은 방식으로 키울 수 있습니다.
  - 즉, 기준을 어떻게 잡아야 가장 효과적인 Tree 를 만들 수 있을까요?
    - ❖Greedy 하게 접근
  - 우리가 정해야 하는 것
    - ❖축 (어떠한 feature 를 가지고 분류할 것인가)
    - ❖기준 (분류할 기준점은 어디인가)
- 언제까지 split 해야할까요?
  - a)주어진 영역에 point 가 1개만 주어져 있는 경우
  - b)각 영역에 m개의 point 이하가 주어져 있는 경우
- Greedy Approach
  - Global Optimum 이라는 보장은 없습니다.



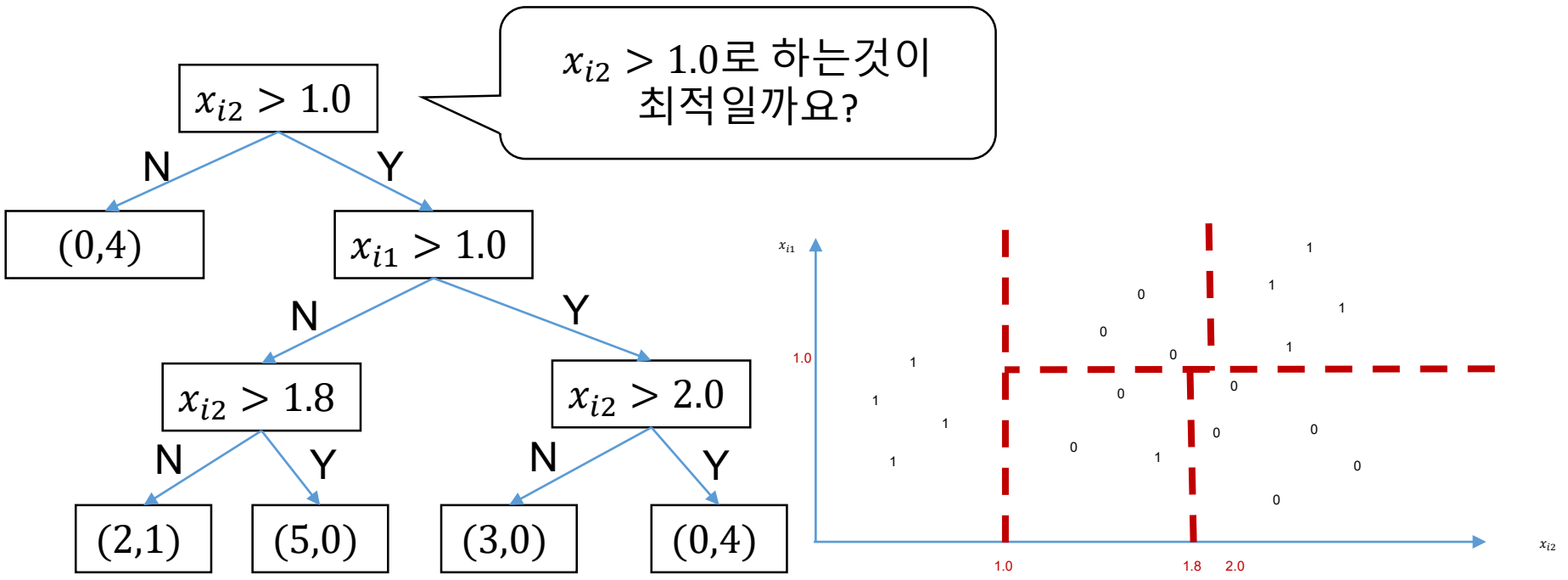
$$\begin{matrix} s \\ j = 1 \end{matrix} \qquad x_{i1}$$

Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Growing)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - 어떻게 하면, Tree 를 효과적으로 키울 수 있을까요? (Growing Tree)
  - 우리는 Decision Tree 를 굉장히 많은 방식으로 키울 수 있습니다.
  - 즉, 기준을 어떻게 잡아야 가장 효과적인 Tree 를 만들 수 있을까요?
    - ❖이제는, Classification Case 를 살펴보겠습니다.

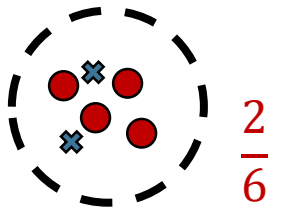


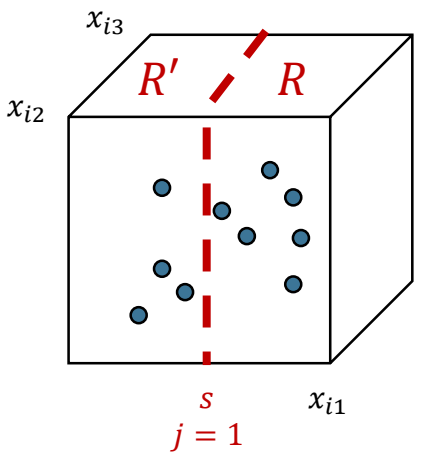
Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Growing)

Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - 어떻게 하면, Tree 를 효과적으로 키울 수 있을까요? (Growing Tree)
  - 우리는 Decision Tree 를 굉장히 많은 방식으로 키울 수 있습니다.
  - 즉, 기준을 어떻게 잡아야 가장 효과적인 Tree 를 만들 수 있을까요?
    - ❖ 이제는, Classification Case 를 살펴보겠습니다.
- $(x_1, y_1), \dots, (x_n, y_n)$
  - E_R: Fraction of points $x_i \in R$
    - ❖ $\min_y \frac{1}{N_R} \sum_{i:x_i \in R} I(y_i \neq y)$
      - ➢ I: indicator function
      - ➢ N_R: $\#\{i: x_i \in R\}$
    - ❖ 즉, majority 로 예측 했을 때, 잘못 분류될 데이터의 개수
- First Split
  - $E_{R(j,s)} + E_{R'}(j,s)$가 minimize 되는 j와 s 설정
    - ❖ $R(j,s) = \{x_i : x_{ij} > s\}$, $R'(j,s) = \{x_i : x_{ij} \leq s\}$
- Second Split …
- c)영역 내에 하나의 class만 존재 할 때까지 Split



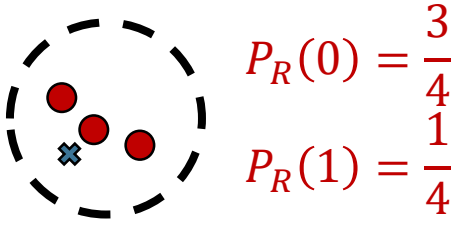


Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Decision Tree (Growing)

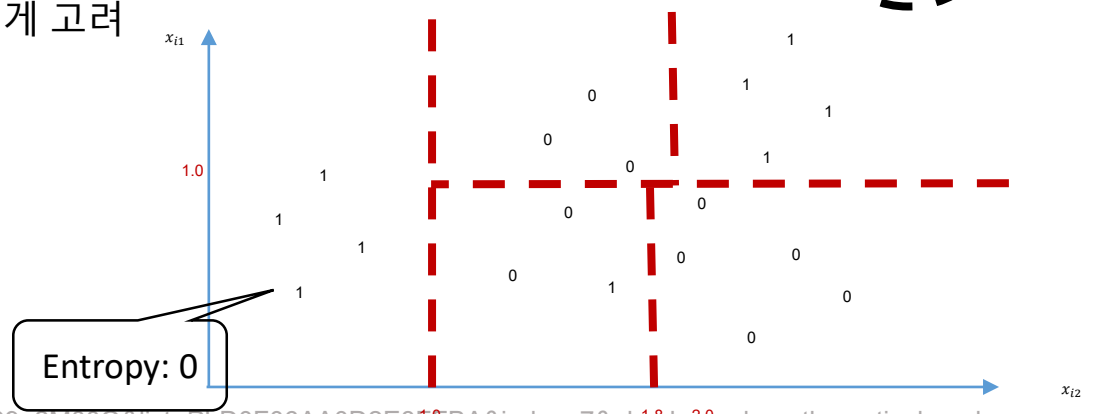
Decision Tree

- Decision Tree (CART, Classification And Regression Tree)
  - 어떻게 하면, Tree 를 효과적으로 키울 수 있을까요? (Growing Tree)
  - 우리는 Decision Tree 를 굉장히 많은 방식으로 키울 수 있습니다.
  - 즉, 기준을 어떻게 잡아야 가장 효과적인 Tree 를 만들 수 있을까요?
    - ❖ 이제는, Classification Case 를 살펴보겠습니다.
    - ❖ $\min_y \frac{1}{N_R} \sum_{i:x_i \in R} I(y_i \neq y)$
- 우리가 살펴본 것은, impurity measure 라고 해석할 수 있습니다.
- 이러한 impurity measure는 다양하게 존재합니다.
  - 1) Misclassification rate $E_R = \min_y \frac{1}{N_R} \sum_{i:x_i \in R} I(y_i \neq y)$
  - 2) Entropy
    - ❖ $H_R = -\sum_{y \in Y} P_R(y) \log P_R(y)$
    - ❖ Multi-class 까지도 자연스럽게 고려
  - 3) Gini index
    - ❖ $G_R = \sum_{y \in Y} P_R(y)(1 - P_R(y))$
      $= 1 - \sum_{y \in Y} P_R(y)^2$



$P_R(0) = \frac{3}{4}$

$P_R(1) = \frac{1}{4}$



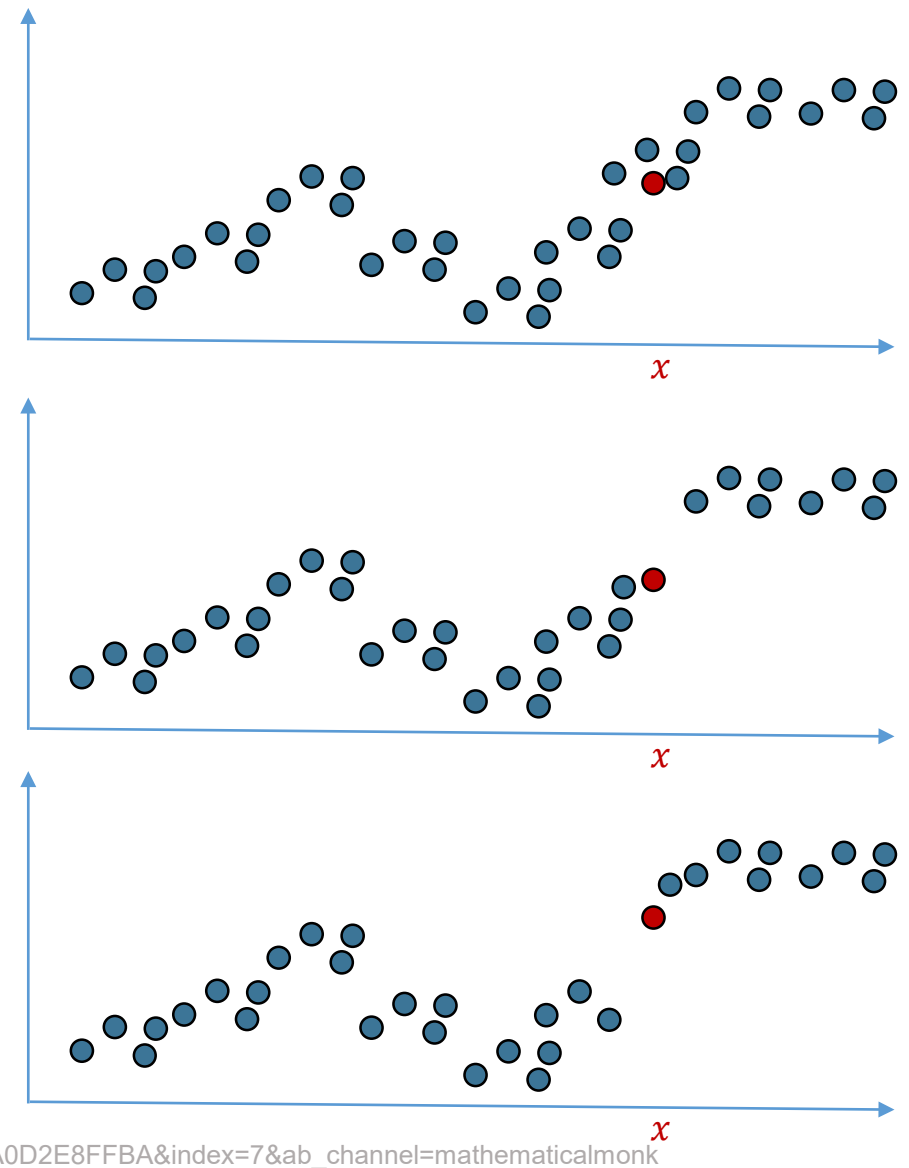
Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Random Forest

Bootstrap Aggregation (Bagging)

Bagging

- 실제로 Tree 모델을 활용해면, 매우 sensitive 한 것을 확인할 수 있습니다. ⇒ Bagging 을 활용해보자!
- Bagging
  - $(x_1, y_1), \dots, (x_n, y_n) \sim P$ i.i.d. $f(x) = y$
  - 여러 개의 data sample 을 통해 평균을 내고 싶다.
  - $\left(x_1^{(1)}, y_1^{(1)}\right), \dots, \left(x_n^{(1)}, y_n^{(1)}\right), (x_{te}, y^{(1)})$
  - …
  - $\left(x_1^{(m)}, y_1^{(m)}\right), \dots, \left(x_n^{(m)}, y_n^{(m)}\right) (x_{te}, y^{(m)})$
  - $Z = \frac{1}{m}\sum_{i=1}^{m} y^{(i)}$
  - $E[Z] = \frac{1}{m}\sum_{i=1}^{m} y = y$, **unbiased**
  - $E[(Z-y)^2] = \sigma^2(Z) = \frac{1}{m}\sigma^2(y)$



Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Bootstrap Aggregation (Bagging)

Bagging

- 하지만.. 우리가 m개의 data 분포를 얻을 수 있을까요?
  - 우리에게 주어지는 것은 1개의 data 분포 일뿐…
  - 근사를 하자!
  - 즉, 우리에게 주어진 empirical data $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$ 에서 uniform 하게!
  - 즉, 주어진 data 에서 random 하게 sampling

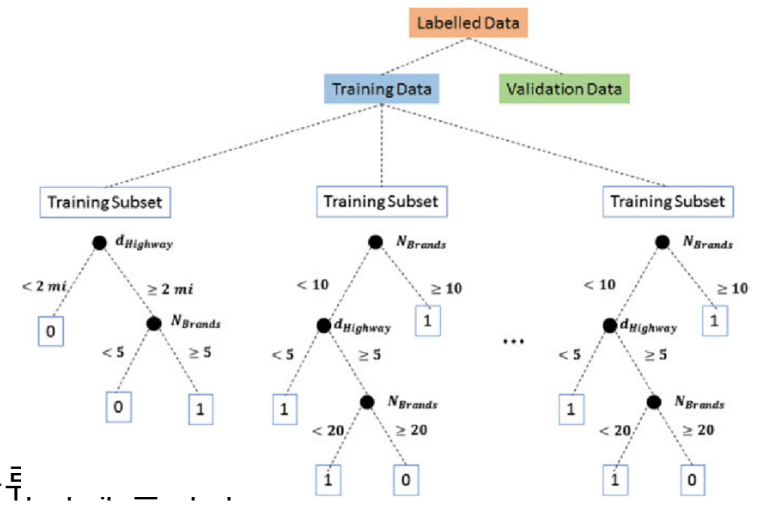
- Bagging for Classification
  - 그러면, 우리에게, m개의 prediction 값이 생기게 되는데, 어떻게 classification 할까?
  - 1) Majority vote
    - ❖ $C_1, \dots, C_m$
  - 2) Average Probabilities
    - ❖ $p^{(1)}, p^{(2)}, \dots, p^{(m)}$
    - ❖ $\frac{1}{m}\sum_{i=1}^{m} p^{(i)}(y)$

Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk

Random Forest

Random Forest

- $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_i \in R^d$
- Bootstrap
  - For $i = 1, \ldots, m$
    - ❖D 로부터 random하게, Bootstrap Sample D_i 추출
    - ❖Construct Tree T_i using D_i
      - ➢ 즉, D_i마다 T_i가 하나씩 만들어지게 되는 것입니다.
      - ➢ 즉, 전체 m개가 만들어지는 것이지요.
      - ➢ 단 여기서, 각 node마다, random subset of features 로 분할.
        - ✓ Random subspace
        - ✓ 왜 이름이 random forest 인지 아시겠죠?
      - ➢ 각 T_i는 서로 다른 tree 로 구성되게 됩니다.



- Aggregation
  - 새롭게 주어진 x에 대해서, m개의 결과물을 가질 수 있습니다.
  - 이러한 m개의 결과를 aggregate 해서, 최종적인 output 을 도출합니다.
    - ❖Majority vote, Average Probabilities

Source: https://www.youtube.com/watch?v=p17C9q2M00Q&list=PLD0F06AA0D2E8FFBA&index=7&ab_channel=mathematicalmonk https://www.researchgate.net/figure/Cartoon-representation-of-a-random-forest-classifier_fig1_337361248

Conclusion

- Linear Statistical Model
  - 행렬 기반으로 연산하는 방법
  - 역행렬 기반 연산의 단점
  - Gradient Descent

- Logistic Regression
  - Logistic Regression이 풀 수 있는 문제들 (Classification)
  - Logistic Regression의 목적식 (CE)
  - Logistic Regression을 학습하는 방법 (Gradient Descent)

- Decision Tree and Random Forest

- 각종 학습 환경
  - Supervised Learning
  - Unsupervised Learning
  - Semi-Supervised Learning

Source: