

# W2. Discrete Classification - Logistic Regression

Guang Cheng

University of California, Los Angeles

guangcheng@ucla.edu

Week 2

# Classification

- A typical dataset in classification  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ .

# Classification

- A typical dataset in classification  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ .
  - $\mathbf{x}_i$ : the covariate vector of  $i$ -th instance

# Classification

- A typical dataset in classification  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ .
  - $\mathbf{x}_i$ : the covariate vector of  $i$ -th instance
  - $y_i \in \{-1, 1\}$ : binary label of  $i$ -th instance

# Classification

- A typical dataset in classification  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ .
  - $\mathbf{x}_i$ : the covariate vector of  $i$ -th instance
  - $y_i \in \{-1, 1\}$ : binary label of  $i$ -th instance
- **Question:** Can we directly minimize the averaged 0-1 loss?

$$\text{Training Error : } \frac{1}{n} \sum_{i=1}^n I(f(\mathbf{x}_i) \neq y_i)$$

- A typical dataset in classification  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ .
  - $\mathbf{x}_i$ : the covariate vector of  $i$ -th instance
  - $y_i \in \{-1, 1\}$ : binary label of  $i$ -th instance
- **Question:** Can we directly minimize the averaged 0-1 loss?

$$\text{Training Error : } \frac{1}{n} \sum_{i=1}^n I(f(\mathbf{x}_i) \neq y_i)$$

- **Answer:** No, the 0-1 loss function is non-convex and discontinuous, so (sub)gradient methods cannot be applied.

# Classification - surrogate loss

- We can replace the 0-1 loss by other loss functions, say surrogate loss

$$\frac{1}{n} \sum_{i=1}^n I(f(\mathbf{x}_i) \neq y_i) \Rightarrow \frac{1}{n} \sum_{i=1}^n L(f(\mathbf{x}_i), y_i) = \frac{1}{n} \sum_{i=1}^n \phi(f(\mathbf{x}_i) y_i)$$

# Classification - surrogate loss

- We can replace the 0-1 loss by other loss functions, say surrogate loss

$$\frac{1}{n} \sum_{i=1}^n I(f(\mathbf{x}_i) \neq y_i) \Rightarrow \frac{1}{n} \sum_{i=1}^n L(f(\mathbf{x}_i), y_i) = \frac{1}{n} \sum_{i=1}^n \phi(f(\mathbf{x}_i)y_i)$$

- Hinge loss:  $\phi(x) = \max\{0, 1 - x\}$



# Classification - surrogate loss

- We can replace the 0-1 loss by other loss functions, say surrogate loss

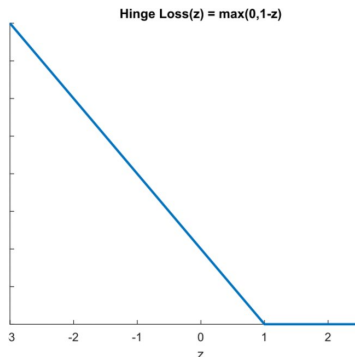
$$\frac{1}{n} \sum_{i=1}^n I(f(\mathbf{x}_i) \neq y_i) \Rightarrow \frac{1}{n} \sum_{i=1}^n L(f(\mathbf{x}_i), y_i) = \frac{1}{n} \sum_{i=1}^n \phi(f(\mathbf{x}_i) y_i)$$

- Hinge loss:  $\phi(x) = \max\{0, 1 - x\}$
- Logistic loss  $\phi(x) = \log(1 + \exp(-x))$

# One surrogate loss - Hinge Loss

- Definition of Hinge loss:

$$L_{\text{hinge}}(f(\mathbf{x}_i), y_i) = \begin{cases} 1 - f(\mathbf{x}_i)y_i & \text{if } f(\mathbf{x}_i)y_i \leq 1 \\ 0, & \text{if } f(\mathbf{x}_i)y_i > 1 \end{cases}$$



# Why Hinge Loss?

- Let  $\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})$ . The expected hinge loss: hinge risk.

$$\begin{aligned} R_{\text{hinge}}(f) &= \mathbb{E}_{\mathbf{X}, Y} [L_{\text{hinge}}(f(\mathbf{X}), Y)] \\ &= \mathbb{E}_{\mathbf{X}} \left[ \eta(\mathbf{X})(1 - f(\mathbf{X}))_+ + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X}))_+ \right] \end{aligned}$$

# Why Hinge Loss?

- Let  $\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})$ . The expected hinge loss: hinge risk.

$$\begin{aligned} R_{\text{hinge}}(f) &= \mathbb{E}_{\mathbf{X}, Y} [L_{\text{hinge}}(f(\mathbf{X}), Y)] \\ &= \mathbb{E}_{\mathbf{X}} \left[ \eta(\mathbf{X})(1 - f(\mathbf{X}))_+ + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X}))_+ \right] \end{aligned}$$

- Suppose that  $f(\mathbf{X}) \in [-1, 1]$ , for any  $\mathbf{X}$ , we have (pls verify in class)

$$\begin{aligned} &\eta(\mathbf{X})(1 - f(\mathbf{X})) + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X})) \\ &= \eta(\mathbf{X}) - 2\eta(\mathbf{X})f(\mathbf{X}) + 1 + f(\mathbf{X}) - \eta(\mathbf{X}) \\ &= f(\mathbf{X})(1 - 2\eta(\mathbf{X})) + 1. \end{aligned}$$

# Why Hinge Loss?

- Let  $\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})$ . The expected hinge loss: hinge risk.

$$\begin{aligned} R_{\text{hinge}}(f) &= \mathbb{E}_{\mathbf{X}, Y} [L_{\text{hinge}}(f(\mathbf{X}), Y)] \\ &= \mathbb{E}_{\mathbf{X}} \left[ \eta(\mathbf{X})(1 - f(\mathbf{X}))_+ + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X}))_+ \right] \end{aligned}$$

- Suppose that  $f(\mathbf{X}) \in [-1, 1]$ , for any  $\mathbf{X}$ , we have (pls verify in class)

$$\begin{aligned} &\eta(\mathbf{X})(1 - f(\mathbf{X})) + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X})) \\ &= \eta(\mathbf{X}) - 2\eta(\mathbf{X})f(\mathbf{X}) + 1 + f(\mathbf{X}) - \eta(\mathbf{X}) \\ &= f(\mathbf{X})(1 - 2\eta(\mathbf{X})) + 1. \end{aligned}$$

- The optimal function  $f_{\text{hinge}}^*$  minimizing  $R_{\text{hinge}}(f)$  (why?)

# Why Hinge Loss?

- Let  $\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})$ . The expected hinge loss: hinge risk.

$$\begin{aligned} R_{\text{hinge}}(f) &= \mathbb{E}_{\mathbf{X}, Y} [L_{\text{hinge}}(f(\mathbf{X}), Y)] \\ &= \mathbb{E}_{\mathbf{X}} \left[ \eta(\mathbf{X})(1 - f(\mathbf{X}))_+ + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X}))_+ \right] \end{aligned}$$

- Suppose that  $f(\mathbf{X}) \in [-1, 1]$ , for any  $\mathbf{X}$ , we have (pls verify in class)

$$\begin{aligned} &\eta(\mathbf{X})(1 - f(\mathbf{X})) + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X})) \\ &= \eta(\mathbf{X}) - 2\eta(\mathbf{X})f(\mathbf{X}) + 1 + f(\mathbf{X}) - \eta(\mathbf{X}) \\ &= f(\mathbf{X})(1 - 2\eta(\mathbf{X})) + 1. \end{aligned}$$

- The optimal function  $f_{\text{hinge}}^*$  minimizing  $R_{\text{hinge}}(f)$  (why?)
  - If  $\eta(\mathbf{X}) < 1/2$ , hinge loss is minimized at  $f(\mathbf{X}) = -1$

# Why Hinge Loss?

- Let  $\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})$ . The expected hinge loss: hinge risk.

$$\begin{aligned} R_{\text{hinge}}(f) &= \mathbb{E}_{\mathbf{X}, Y} [L_{\text{hinge}}(f(\mathbf{X}), Y)] \\ &= \mathbb{E}_{\mathbf{X}} \left[ \eta(\mathbf{X})(1 - f(\mathbf{X}))_+ + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X}))_+ \right] \end{aligned}$$

- Suppose that  $f(\mathbf{X}) \in [-1, 1]$ , for any  $\mathbf{X}$ , we have (pls verify in class)

$$\begin{aligned} &\eta(\mathbf{X})(1 - f(\mathbf{X})) + (1 - \eta(\mathbf{X}))(1 + f(\mathbf{X})) \\ &= \eta(\mathbf{X}) - 2\eta(\mathbf{X})f(\mathbf{X}) + 1 + f(\mathbf{X}) - \eta(\mathbf{X}) \\ &= f(\mathbf{X})(1 - 2\eta(\mathbf{X})) + 1. \end{aligned}$$

- The optimal function  $f_{\text{hinge}}^*$  minimizing  $R_{\text{hinge}}(f)$  (why?)
  - If  $\eta(\mathbf{X}) < 1/2$ , hinge loss is minimized at  $f(\mathbf{X}) = -1$
  - If  $\eta(\mathbf{X}) > 1/2$ , hinge loss is minimized at  $f(\mathbf{X}) = 1$

# Why Hinge Loss?

- The optimal classifier (i.e., Bayes classifier) of Binary loss is defined as

$$f^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2) = \begin{cases} 1 & \text{if } \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) > 1/2 \\ 0 & \text{if } \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) < 1/2 \end{cases}$$



# Why Hinge Loss?

- The optimal classifier (i.e., Bayes classifier) of Binary loss is defined as

$$f^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2) = \begin{cases} 1 & \text{if } \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) > 1/2 \\ 0 & \text{if } \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) < 1/2 \end{cases}$$

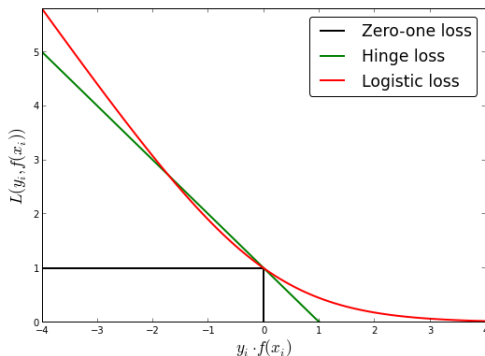
- **Observation:**

- (i)  $f_{\text{hinge}}^*$  is exactly the Bayes classifier defined above;
- (ii) The hinge loss is a convex function, which makes it possible to minimize the training error in practice.

# Another surrogate loss - Logistic Loss

- Definition of Logistic loss:

$$L_{\log}(f(\mathbf{x}_i), y_i) = \log(1 + \exp(-f(\mathbf{x}_i)y_i))$$



# Why Logistic Loss?

- The logistic risk:

$$\begin{aligned} R_{\log}(f) &= \mathbb{E}_{\mathbf{X}, Y} \left[ \log \left( 1 + \exp(-f(\mathbf{X})Y) \right) \right] \\ &= \mathbb{E}_{\mathbf{X}} \left[ \eta(\mathbf{X}) \log \left( 1 + \exp(-f(\mathbf{X})) \right) + (1 - \eta(\mathbf{X})) \log \left( 1 + \exp(f(\mathbf{X})) \right) \right] \end{aligned}$$

# Why Logistic Loss?

- The logistic risk:

$$\begin{aligned} R_{\log}(f) &= \mathbb{E}_{\mathbf{X}, Y} \left[ \log \left( 1 + \exp(-f(\mathbf{X})Y) \right) \right] \\ &= \mathbb{E}_{\mathbf{X}} \left[ \eta(\mathbf{X}) \log \left( 1 + \exp(-f(\mathbf{X})) \right) + (1 - \eta(\mathbf{X})) \log \left( 1 + \exp(f(\mathbf{X})) \right) \right] \end{aligned}$$

- Take the derivative with respect to  $f$ , **pls verify the following in class**

$$\begin{aligned} & -\eta(\mathbf{X}) \frac{\exp(-f(\mathbf{X}))}{1 + \exp(-f(\mathbf{X}))} + (1 - \eta(\mathbf{X})) \frac{\exp(f(\mathbf{X}))}{1 + \exp(f(\mathbf{X}))} \\ &= -\eta(\mathbf{X}) \frac{1}{1 + \exp(f(\mathbf{X}))} + (1 - \eta(\mathbf{X})) \frac{\exp(f(\mathbf{X}))}{1 + \exp(f(\mathbf{X}))} \\ &= \frac{\exp(f(\mathbf{X}))}{1 + \exp(f(\mathbf{X}))} - \eta(\mathbf{X}) = 0 \iff f_{\log}^*(\mathbf{X}) = \log \frac{\eta(\mathbf{X})}{1 - \eta(\mathbf{X})} \end{aligned}$$

# Connection between binary loss and surrogate losses

- The Bayes classifier  $f^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$

# Connection between binary loss and surrogate losses

- The Bayes classifier  $f^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$
- The optimal classifier of Hinge risk  $f_{\text{hinge}}^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$

# Connection between binary loss and surrogate losses

- The Bayes classifier  $f^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$
- The optimal classifier of Hinge risk  $f_{\text{hinge}}^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$
- The optimal classifier of Logistic risk  $f_{\text{log}}^*(\mathbf{x}) = \log \frac{\eta(\mathbf{x})}{1-\eta(\mathbf{x})}$

# Connection between binary loss and surrogate losses

- The Bayes classifier  $f^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$
- The optimal classifier of Hinge risk  $f_{\text{hinge}}^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$
- The optimal classifier of Logistic risk  $f_{\text{log}}^*(\mathbf{x}) = \log \frac{\eta(\mathbf{x})}{1-\eta(\mathbf{x})}$
- **Question:** what is the connection between these optimal classifiers?



# Connection between binary loss and surrogate losses

- The Bayes classifier  $f^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$
- The optimal classifier of Hinge risk  $f_{\text{hinge}}^*(\mathbf{x}) = \text{sign}(\eta(\mathbf{x}) - 1/2)$
- The optimal classifier of Logistic risk  $f_{\text{log}}^*(\mathbf{x}) = \log \frac{\eta(\mathbf{x})}{1-\eta(\mathbf{x})}$
- **Question:** what is the connection between these optimal classifiers?
- **Answer:** They are consistent in sign in the sense that signs of  $f^*, f_{\text{hinge}}^*, f_{\text{log}}^*$  are always the same, e.g., always positive as long as  $\eta(\mathbf{x}) > 1/2$ .

# Logistic Regression – model details

- To estimate  $f_{log}^*(\mathbf{x})$ , we need to impose an assumption on the form of:

$$\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{x})$$

# Logistic Regression – model details

- To estimate  $f_{log}^*(\mathbf{x})$ , we need to impose an assumption on the form of:

$$\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{x})$$

- In logistic regression, it is often assumed that

$$\eta(\mathbf{x}) = \frac{\exp(\beta_0 + \beta^T \mathbf{x})}{1 + \exp(\beta_0 + \beta^T \mathbf{x})}, \quad (1)$$

where

# Logistic Regression – model details

- To estimate  $f_{log}^*(\mathbf{x})$ , we need to impose an assumption on the form of:

$$\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{x})$$

- In logistic regression, it is often assumed that

$$\eta(\mathbf{x}) = \frac{\exp(\beta_0 + \beta^T \mathbf{x})}{1 + \exp(\beta_0 + \beta^T \mathbf{x})}, \quad (1)$$

where

- $\mathbf{x} = (x_1, \dots, x_p)^T$  is a  $p$ -dimensional predictor

# Logistic Regression – model details

- To estimate  $f_{log}^*(\mathbf{x})$ , we need to impose an assumption on the form of:

$$\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{x})$$

- In logistic regression, it is often assumed that

$$\eta(\mathbf{x}) = \frac{\exp(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}{1 + \exp(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}, \quad (1)$$

where

- $\mathbf{x} = (x_1, \dots, x_p)^T$  is a  $p$ -dimensional predictor
- $\beta_0$  and  $\boldsymbol{\beta} = (\beta_1, \dots, \beta_p)$  are unknown parameters to be estimated

# Logistic Regression – model details

- To estimate  $f_{log}^*(\mathbf{x})$ , we need to impose an assumption on the form of:

$$\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{x})$$

- In logistic regression, it is often assumed that

$$\eta(\mathbf{x}) = \frac{\exp(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}{1 + \exp(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}, \quad (1)$$

where

- $\mathbf{x} = (x_1, \dots, x_p)^T$  is a  $p$ -dimensional predictor
- $\beta_0$  and  $\boldsymbol{\beta} = (\beta_1, \dots, \beta_p)$  are unknown parameters to be estimated
- $\boldsymbol{\beta}^T \mathbf{x} = \sum_{i=1}^p \beta_i x_i$

# Rational behind logistic loss: log odds ratio

- By reformulating (1), we have obtained

$$\exp(\beta_0 + \beta^T \mathbf{x}) = \frac{\mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})}{1 - \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})} = \frac{\mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})}{\mathbb{P}(Y = 0 | \mathbf{X} = \mathbf{x})},$$

where the last term above is the ratio between the conditional probability of  $Y = 1$  and that of  $Y = 0$  on  $\mathbf{X} = \mathbf{x}$ , i.e., “odds ratio.”

# Rational behind logistic loss: log odds ratio

- By reformulating (1), we have obtained

$$\exp(\beta_0 + \beta^T \mathbf{x}) = \frac{\mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})}{1 - \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})} = \frac{\mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})}{\mathbb{P}(Y = 0 | \mathbf{X} = \mathbf{x})},$$

where the last term above is the ratio between the conditional probability of  $Y = 1$  and that of  $Y = 0$  on  $\mathbf{X} = \mathbf{x}$ , i.e., “odds ratio.”

- In other words, we can claim that the log-odds is assumed to be linear with respect to  $\beta$ :

$$\beta_0 + \beta^T \mathbf{x} = \log \left( \frac{\mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})}{\mathbb{P}(Y = 0 | \mathbf{X} = \mathbf{x})} \right)$$

**Interpretability:**  $\beta_i$  can then be interpreted as the average change in the log-odds ratio given by a one-unit increase in  $x_i$



# Maximum likelihood estimation

- Likelihood function  $L(\beta_0, \beta)$ :

$$L(\beta_0, \beta) = \prod_{i=1}^n \left( \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) \right)^{y_i} \left( \mathbb{P}(Y = 0 | \mathbf{X} = \mathbf{x}) \right)^{1-y_i}$$

# Maximum likelihood estimation

- Likelihood function  $L(\beta_0, \beta)$ :

$$L(\beta_0, \beta) = \prod_{i=1}^n \left( \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) \right)^{y_i} \left( \mathbb{P}(Y = 0 | \mathbf{X} = \mathbf{x}) \right)^{1-y_i}$$

- Logarithm of  $L(\beta_0, \beta)$  (pls verify in class):

$$\begin{aligned} \log L(\beta_0, \beta) &= \sum_{i=1}^n \left[ y_i \log \left( \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x}) \right) + (1 - y_i) \log \left( \mathbb{P}(Y = 0 | \mathbf{X} = \mathbf{x}) \right) \right] \\ &= \sum_{i=1}^n \left[ y_i (\beta_0 + \beta^T \mathbf{x}) - \log \left( 1 + \exp(\beta_0 + \beta^T \mathbf{x}) \right) \right] \end{aligned}$$

# Gradient descent/ascent in the computation

Estimate  $\beta_0$  and  $\beta$  (Gradient Ascent):

$$\begin{aligned}\beta_0^{(t+1)} &\leftarrow \beta_0^{(t)} + \lambda \sum_{i=1}^n \left[ y_i - \frac{\exp(\beta_0^{(t)} + \beta^{(t)T} \mathbf{x})}{1 + \exp(\beta_0^{(t)} + \beta^{(t)T} \mathbf{x})} \right] \\ \beta^{(t+1)} &\leftarrow \beta^{(t)} + \lambda \sum_{i=1}^n \left[ y_i - \frac{\exp(\beta_0^{(t)} + \beta^{(t)T} \mathbf{x})}{1 + \exp(\beta_0^{(t)} + \beta^{(t)T} \mathbf{x})} \right] \mathbf{x}_i\end{aligned}$$

# Now, we are ready to do classification

We have obtained the estimate for  $\beta_0$  and  $\beta$ , denoted as  $\hat{\beta}_0$  and  $\hat{\beta}$ , based on which we can estimate  $P(Y = 1|\mathbf{X})$  as follows:

$$\hat{\eta}(\mathbf{x}) = \frac{\exp(\hat{\beta}_0 + \hat{\beta}^T \mathbf{x})}{1 + \exp(\hat{\beta}_0 + \hat{\beta}^T \mathbf{x})}$$

# Now, we are ready to do classification

We have obtained the estimate for  $\beta_0$  and  $\beta$ , denoted as  $\hat{\beta}_0$  and  $\hat{\beta}$ , based on which we can estimate  $P(Y = 1|\mathbf{X})$  as follows:

$$\hat{\eta}(\mathbf{x}) = \frac{\exp(\hat{\beta}_0 + \hat{\beta}^T \mathbf{x})}{1 + \exp(\hat{\beta}_0 + \hat{\beta}^T \mathbf{x})}$$

Then we make predictions by (recall that  $\eta(\mathbf{x}) = P(Y = 1|\mathbf{X})$ )

$$\hat{f}(\mathbf{x}) = \begin{cases} 1, & \text{if } \hat{\eta}(\mathbf{x}) > 1/2 \\ 0, & \text{if } \hat{\eta}(\mathbf{x}) < 1/2 \end{cases}$$

If  $\hat{\eta}(\mathbf{x}) = 1/2$ , then just randomly assign a label to it.

# Example

Dataset  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{5000}$ , where  $\mathbf{x}_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4})$ .

- Features are generated from uniform distribution  
 $x_{il} \sim \text{Unif}(0, 2), l = 1, 2, 3, 4$ .

# Example

Dataset  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{5000}$ , where  $\mathbf{x}_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4})$ .

- Features are generated from uniform distribution  $x_{il} \sim \text{Unif}(0, 2)$ ,  $l = 1, 2, 3, 4$ .
- $\beta_0 = 0.5$  and  $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3, \beta_4)$  with  $\beta_i \sim \text{Unif}(-1, 1)$

# Example

Dataset  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{5000}$ , where  $\mathbf{x}_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4})$ .

- Features are generated from uniform distribution  $x_{il} \sim \text{Unif}(0, 2), l = 1, 2, 3, 4$ .
- $\beta_0 = 0.5$  and  $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3, \beta_4)$  with  $\beta_i \sim \text{Unif}(-1, 1)$
- Model:

$$Y_i \sim \text{Bernoulli}\left(\frac{\exp(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}{1 + \exp(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}\right),$$

which means

$$P(Y_i = 1 | \mathbf{X}) = \frac{\exp(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}{1 + \exp(\beta_0 + \boldsymbol{\beta}^T \mathbf{x})}$$



# Python Codes – data generation

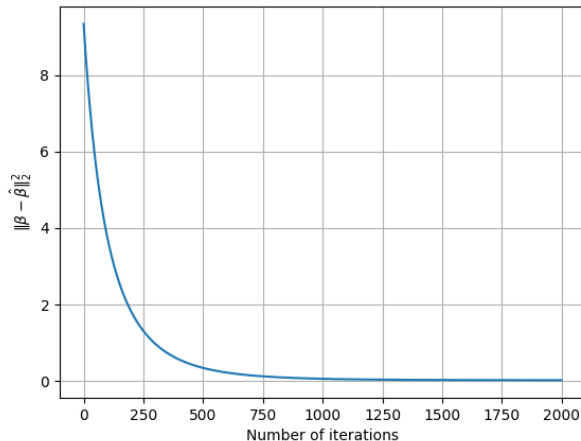
```
import numpy as np
np.random.seed(2)
n,p = 5000,4 # Set training datasize and dimension of features
X = np.random.uniform(-1,1,[n,p]) # Generation of features
beta = np.random.uniform(0,2,4) # Generation of parameters
beta_0 = 0.5 # Set the intercept term to 0.5
logOdd = (X * beta).sum(axis=1)+beta_0 # Log-odds
Prob = np.exp(logOdd)/(1+np.exp(logOdd)) # Probability
Y = np.array(Prob - np.random.uniform(0,1,n)>0,dtype=int) # Generate labels
```

```

Beta_0_hat = 0. # Initialization of intercept term
Beta_hat = np.zeros(p) # Initialization of beta
lamb = 0.1 # Learning rate
Error = [] # Error set
for i in range(2000): # Iterations of gradient ascent
    logOdd_hat = (X * Beta_hat).sum(axis=1) + Beta_0_hat
    Beta_0_hat = Beta_0_hat + lamb * np.mean(Y - np.exp(logOdd_hat) / (1 + np.exp(logOdd_hat)))
    Beta_hat = Beta_hat + lamb * ((Y - np.exp(logOdd_hat) / (1 + np.exp(logOdd_hat))) * X.T).mean(axis=1)
    Error.append(np.linalg.norm(Beta_hat - beta)**2)
import matplotlib.pyplot as plt
plt.plot(np.arange(0, 2000), Error)
plt.xlabel('Number of iterations')
plt.ylabel('$\\|\\beta - \\hat{\\beta}\\|_2^2$')
plt.grid()

```

# Example: gradient ascent for logistic regression



# Over/under-fitting problem

- Overfitting occurs when a model learns the training data too well, capturing noise and making it perform poorly on new, unseen data.

# Over/under-fitting problem

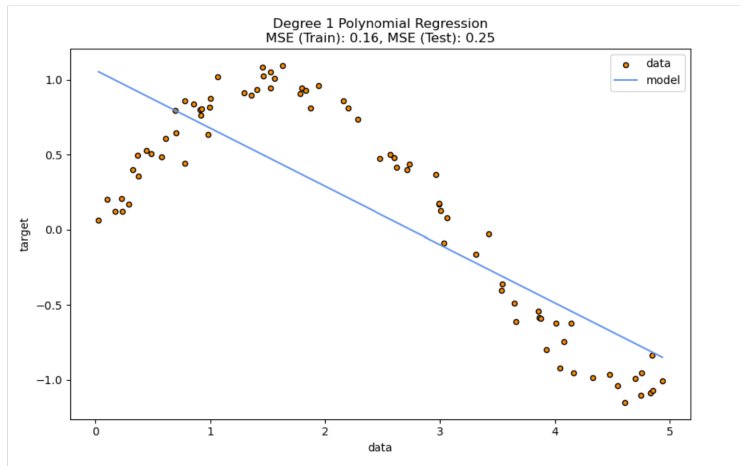
- Overfitting occurs when a model learns the training data too well, capturing noise and making it perform poorly on new, unseen data.
- Underfitting, on the other hand, happens when a model is too simple to capture the underlying patterns in the data, resulting in poor performance on both the training and test data.

# Over/under-fitting problem

- Overfitting occurs when a model learns the training data too well, capturing noise and making it perform poorly on new, unseen data.
- Underfitting, on the other hand, happens when a model is too simple to capture the underlying patterns in the data, resulting in poor performance on both the training and test data.
- Let's use a simple example with polynomial regression and visualize the above with the out-of-sample (OOS) metrics.

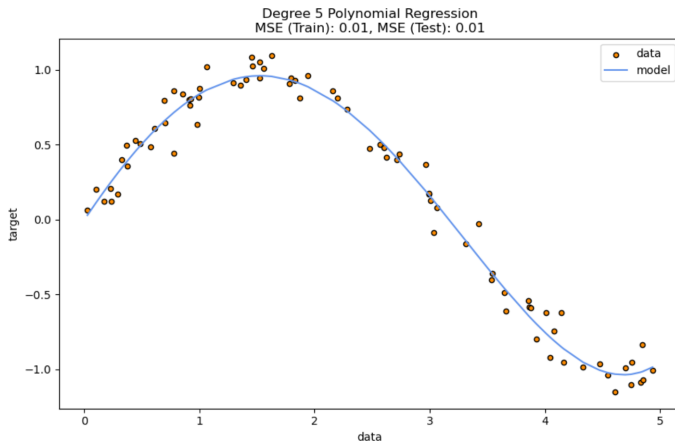
# An example - underfitting

Degree 1: Underfitting (Too simple) - The model is not able to capture the underlying pattern in the data.



# An example - good fit

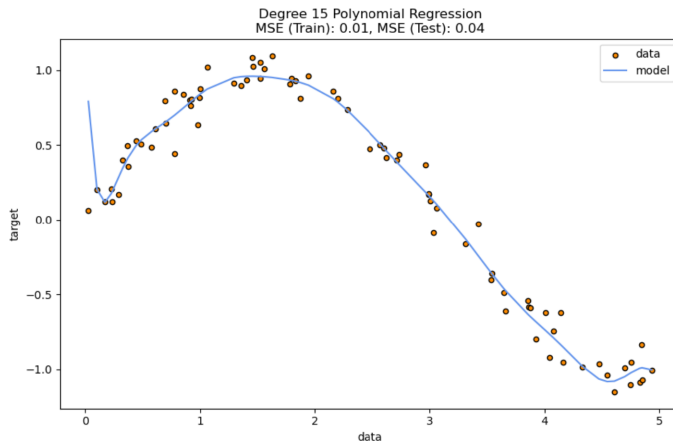
Degree 5: Good fit - The model captures the underlying pattern well and generalizes to the test data.





# An example - overfitting

Degree 15: Overfitting (Too complex) - The model fits the training data too closely, capturing noise and performing poorly on new data.



# Confusion matrix – measure the performance of classification

- In Machine Learning, to measure the performance of the classification model, such as logistic regression, we use the confusion matrix.

# Confusion matrix – measure the performance of classification

- In Machine Learning, to measure the performance of the classification model, such as logistic regression, we use the confusion matrix.
- A confusion matrix is a matrix that displays the number of accurate and inaccurate classification outcomes for each input instance  **$X$** .

# Classification outcomes

To measure the performance of classification, we have the following metrics:

- true positives (TP): occurs when the model accurately predicts a positive data point, i.e.,  $\hat{y} = y = 1$ .

# Classification outcomes

To measure the performance of classification, we have the following metrics:

- true positives (TP): occurs when the model accurately predicts a positive data point, i.e.,  $\hat{y} = y = 1$ .
- true negatives (TN): occurs when the model accurately predicts a negative data point, i.e.,  $\hat{y} = y = -1$ .

# Classification outcomes

To measure the performance of classification, we have the following metrics:

- true positives (TP): occurs when the model accurately predicts a positive data point, i.e.,  $\hat{y} = y = 1$ .
- true negatives (TN): occurs when the model accurately predicts a negative data point, i.e.,  $\hat{y} = y = -1$ .
- false positives (FP): occurs when the model predicts a positive data point incorrectly, i.e.,  $\hat{y} = 1$  but  $y = -1$ .

# Classification outcomes

To measure the performance of classification, we have the following metrics:

- true positives (TP): occurs when the model accurately predicts a positive data point, i.e.,  $\hat{y} = y = 1$ .
- true negatives (TN): occurs when the model accurately predicts a negative data point, i.e.,  $\hat{y} = y = -1$ .
- false positives (FP): occurs when the model predicts a positive data point incorrectly, i.e.,  $\hat{y} = 1$  but  $y = -1$ .
- false negatives (FN): occurs when the model predicts a negative data point incorrectly, i.e.,  $\hat{y} = -1$  but  $y = 1$ .

# An example : dog recognition

Dog:  $Y = 1$  & Not Dog:  $Y = -1$  (pls verify this table in class!)

index	actual	predicted	Result
1	Dog	Dog	TP
2	Dog	Not Dog	FN
3	Dog	Dog	TP
4	Not Dog	Not Dog	TN
5	Dog	Dog	TP
6	Not Dog	Dog	FP
7	Dog	Dog	TP
8	Dog	Dog	TP
9	Not Dog	Not Dog	TN
10	Not Dog	Not Dog	TN



# An example - counts

Pls take a min to count....

- Actual Dog Counts = ?

# An example - counts

Pls take a min to count....

- Actual Dog Counts = ?
- Actual Not Dog Counts = ?

# An example - counts

Pls take a min to count....

- Actual Dog Counts = ?
- Actual Not Dog Counts = ?
- True Positive Counts = ?

# An example - counts

Pls take a min to count....

- Actual Dog Counts = ?
- Actual Not Dog Counts = ?
- True Positive Counts = ?
- False Positive Counts = ?

# An example - counts

Pls take a min to count....

- Actual Dog Counts = ?
- Actual Not Dog Counts = ?
- True Positive Counts = ?
- False Positive Counts = ?
- True Negative Counts = ?

# An example - counts

Pls take a min to count....

- Actual Dog Counts = ?
- Actual Not Dog Counts = ?
- True Positive Counts = ?
- False Positive Counts = ?
- True Negative Counts = ?
- False Negative Counts = ?

# An example - counts

- Actual Dog Counts = 6

# An example - counts

- Actual Dog Counts = 6
- Actual Not Dog Counts = 4



# An example - counts

- Actual Dog Counts = 6
- Actual Not Dog Counts = 4
- True Positive Counts = 5

# An example - counts

- Actual Dog Counts = 6
- Actual Not Dog Counts = 4
- True Positive Counts = 5
- False Positive Counts = 1

# An example - counts

- Actual Dog Counts = 6
- Actual Not Dog Counts = 4
- True Positive Counts = 5
- False Positive Counts = 1
- True Negative Counts = 3

# An example - counts

- Actual Dog Counts = 6
- Actual Not Dog Counts = 4
- True Positive Counts = 5
- False Positive Counts = 1
- True Negative Counts = 3
- False Negative Counts = 1

# An example - construct the confusion matrix

- Construct the Confusion Matrix

		Actual	
		Dog	Not Dog
Predicted	Dog	True Positive (TP =5)	False Positive (FP=1)
	Not Dog	False Negative (FN =1)	True Negative (TN=3)

# An example - construct the confusion matrix

- Construct the Confusion Matrix

		Actual	
		Dog	Not Dog
Predicted	Dog	True Positive (TP =5)	False Positive (FP=1)
	Not Dog	False Negative (FN =1)	True Negative (TN=3)

- How to use the Confusion Matrix for assessing a classification model's performance ?

# Classification metrics based on confusion matrix: Accuracy

- Accuracy is used to measure the performance of the model. It is the ratio of total correct instances to the total instances.

# Classification metrics based on confusion matrix: Accuracy

- Accuracy is used to measure the performance of the model. It is the ratio of total correct instances to the total instances.
- $$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$



# Classification metrics based on confusion matrix: Accuracy

- Accuracy is used to measure the performance of the model. It is the ratio of total correct instances to the total instances.
- $$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$
- For the above case: Accuracy = ?

# Classification metrics based on confusion matrix: Accuracy

- Accuracy is used to measure the performance of the model. It is the ratio of total correct instances to the total instances.
- $\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$
- For the above case: Accuracy = ?
- For the above case: Accuracy =  $(5+3)/(5+3+1+1) = 8/10 = 0.8$

# Classification metrics based on confusion matrix: Precision

- Precision is a measure of how accurate a model's positive predictions are. Basically, it answers the question “What proportion of positive identifications was actually correct?”

# Classification metrics based on confusion matrix: Precision

- Precision is a measure of how accurate a model's positive predictions are. Basically, it answers the question “What proportion of positive identifications was actually correct?”
- It is defined as the ratio of true positive predictions to the total number of positive predictions made by the model.

# Classification metrics based on confusion matrix: Precision

- Precision is a measure of how accurate a model's positive predictions are. Basically, it answers the question “What proportion of positive identifications was actually correct?”
- It is defined as the ratio of true positive predictions to the total number of positive predictions made by the model.
- $\text{Precision} = \frac{TP}{TP+FP}$

# Classification metrics based on confusion matrix: Precision

- Precision is a measure of how accurate a model's positive predictions are. Basically, it answers the question “What proportion of positive identifications was actually correct?”
- It is defined as the ratio of true positive predictions to the total number of positive predictions made by the model.
- $\text{Precision} = \frac{TP}{TP+FP}$
- For the above case: Precision = ?

# Classification metrics based on confusion matrix: Precision

- Precision is a measure of how accurate a model's positive predictions are. Basically, it answers the question “What proportion of positive identifications was actually correct?”
- It is defined as the ratio of true positive predictions to the total number of positive predictions made by the model.
- $\text{Precision} = \frac{TP}{TP+FP}$
- For the above case: Precision = ?
- For the above case: Precision =  $5/(5+1) = 5/6 = 0.8333$

# Classification metrics based on confusion matrix: Recall

- Recall measures the effectiveness of a classification model in identifying all relevant instances from a dataset. Basically, it answers the question “What proportion of actual positives was identified correctly?”



# Classification metrics based on confusion matrix: Recall

- Recall measures the effectiveness of a classification model in identifying all relevant instances from a dataset. Basically, it answers the question “What proportion of actual positives was identified correctly?”
- It is defined as the ratio of the number of true positive (TP) instances to the sum of true positive and false negative (FN) instances.

# Classification metrics based on confusion matrix: Recall

- Recall measures the effectiveness of a classification model in identifying all relevant instances from a dataset. Basically, it answers the question “What proportion of actual positives was identified correctly?”
- It is defined as the ratio of the number of true positive (TP) instances to the sum of true positive and false negative (FN) instances.
- $\text{Recall} = \frac{TP}{TP+FN}$

# Classification metrics based on confusion matrix: Recall

- Recall measures the effectiveness of a classification model in identifying all relevant instances from a dataset. Basically, it answers the question “What proportion of actual positives was identified correctly?”
- It is defined as the ratio of the number of true positive (TP) instances to the sum of true positive and false negative (FN) instances.
- $\text{Recall} = \frac{TP}{TP+FN}$
- For the above case: Recall = ?

# Classification metrics based on confusion matrix: Recall

- Recall measures the effectiveness of a classification model in identifying all relevant instances from a dataset. Basically, it answers the question “What proportion of actual positives was identified correctly?”
- It is defined as the ratio of the number of true positive (TP) instances to the sum of true positive and false negative (FN) instances.
- $\text{Recall} = \frac{TP}{TP+FN}$
- For the above case:  $\text{Recall} = ?$
- For the above case:  $\text{Recall} = 5/(5+1) = 5/6 = 0.8333$

# Classification metrics based on confusion matrix: F-1 Score

- F1-score is used to evaluate the overall performance of a classification model. It is the harmonic mean of precision and recall

# Classification metrics based on confusion matrix: F-1 Score

- F1-score is used to evaluate the overall performance of a classification model. It is the harmonic mean of precision and recall
- The F1 score is named as such because it is a combination of two other metrics: precision (P) and recall (R). The "F" in F1 stands for "F-measure" or "F-score," and the "1" indicates that it is computed as the harmonic mean of precision and recall.

# Classification metrics based on confusion matrix: F-1 Score

- F1-score is used to evaluate the overall performance of a classification model. It is the harmonic mean of precision and recall
- The F1 score is named as such because it is a combination of two other metrics: precision (P) and recall (R). The "F" in F1 stands for "F-measure" or "F-score," and the "1" indicates that it is computed as the harmonic mean of precision and recall.
- The harmonic mean is often used to calculate the average of the ratios or rates.

# Classification metrics based on confusion matrix: F-1 Score

- F1-score is used to evaluate the overall performance of a classification model. It is the harmonic mean of precision and recall
- The F1 score is named as such because it is a combination of two other metrics: precision (P) and recall (R). The "F" in F1 stands for "F-measure" or "F-score," and the "1" indicates that it is computed as the harmonic mean of precision and recall.
- The harmonic mean is often used to calculate the average of the ratios or rates.
- The harmonic mean can be expressed as the reciprocal of the arithmetic mean of the reciprocals of the given set of observations.



# Classification metrics based on confusion matrix: F-1 Score

- F1-score is used to evaluate the overall performance of a classification model. It is the harmonic mean of precision and recall
- The F1 score is named as such because it is a combination of two other metrics: precision (P) and recall (R). The "F" in F1 stands for "F-measure" or "F-score," and the "1" indicates that it is computed as the harmonic mean of precision and recall.
- The harmonic mean is often used to calculate the average of the ratios or rates.
- The harmonic mean can be expressed as the reciprocal of the arithmetic mean of the reciprocals of the given set of observations.
- For example, harmonic mean of 1, 4, 4 is

$$\left(\frac{1^{-1} + 4^{-1} + 4^{-1}}{3}\right)^{-1} = 2$$

# Classification metrics based on confusion matrix: F-1 Score

- Pls verify this in class:  $F1\text{-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

# Classification metrics based on confusion matrix: F-1 Score

- Pls verify this in class:  $F1\text{-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
- For the above case: F1-Score=?

# Classification metrics based on confusion matrix: F-1 Score

- **Pls verify this in class:**  $F1\text{-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
- For the above case: F1-Score=?
- For the above case:  $F1\text{-Score} = (2 * 0.8333 * 0.8333) / (0.8333 + 0.8333) = 0.8333$

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Problem statement

- You have a binary classification model used to predict whether an email is spam (positive class) or not spam (negative class). After testing the model on a dataset of 100 emails, you get the following results:

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Problem statement

- You have a binary classification model used to predict whether an email is spam (positive class) or not spam (negative class). After testing the model on a dataset of 100 emails, you get the following results:
  - 40 emails are correctly identified as spam (True Positives).

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Problem statement

- You have a binary classification model used to predict whether an email is spam (positive class) or not spam (negative class). After testing the model on a dataset of 100 emails, you get the following results:
  - 40 emails are correctly identified as spam (True Positives).
  - 10 emails are incorrectly identified as spam (False Positives).

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Problem statement

- You have a binary classification model used to predict whether an email is spam (positive class) or not spam (negative class). After testing the model on a dataset of 100 emails, you get the following results:
  - 40 emails are correctly identified as spam (True Positives).
  - 10 emails are incorrectly identified as spam (False Positives).
  - 30 emails are correctly identified as not spam (True Negatives).



# Exercise: Confusion Matrix and Classification Metrics Calculation

## Problem statement

- You have a binary classification model used to predict whether an email is spam (positive class) or not spam (negative class). After testing the model on a dataset of 100 emails, you get the following results:
  - 40 emails are correctly identified as spam (True Positives).
  - 10 emails are incorrectly identified as spam (False Positives).
  - 30 emails are correctly identified as not spam (True Negatives).
  - 20 emails are incorrectly identified as not spam (False Negatives).

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Your task

- Construct a confusion matrix from these results.

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Your task

- Construct a confusion matrix from these results.
- Calculate the following metrics:

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Your task

- Construct a confusion matrix from these results.
- Calculate the following metrics:
  - Accuracy

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Your task

- Construct a confusion matrix from these results.
- Calculate the following metrics:
  - Accuracy
  - Precision

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Your task

- Construct a confusion matrix from these results.
- Calculate the following metrics:
  - Accuracy
  - Precision
  - Recall

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Your task

- Construct a confusion matrix from these results.
- Calculate the following metrics:
  - Accuracy
  - Precision
  - Recall
  - F-1 Score

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:



# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:
  - True Positives (TP): 40

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:
  - True Positives (TP): 40
  - False Positives (FP): 10

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:
  - True Positives (TP): 40
  - False Positives (FP): 10
  - True Negatives (TN): 30

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:
  - True Positives (TP): 40
  - False Positives (FP): 10
  - True Negatives (TN): 30
  - False Negatives (FN): 20

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:
  - True Positives (TP): 40
  - False Positives (FP): 10
  - True Negatives (TN): 30
  - False Negatives (FN): 20
- Calculate Accuracy:  $\text{Accuracy} = \frac{40+30}{40+30+10+20} = 0.7$

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:
  - True Positives (TP): 40
  - False Positives (FP): 10
  - True Negatives (TN): 30
  - False Negatives (FN): 20
- Calculate Accuracy:  $\text{Accuracy} = \frac{40+30}{40+30+10+20} = 0.7$
- Precision =  $\frac{40}{40+10} = 0.8$

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:
  - True Positives (TP): 40
  - False Positives (FP): 10
  - True Negatives (TN): 30
  - False Negatives (FN): 20
- Calculate Accuracy:  $\text{Accuracy} = \frac{40+30}{40+30+10+20} = 0.7$
- Precision =  $\frac{40}{40+10} = 0.8$
- Recall =  $\frac{40}{40+20} = \frac{2}{3}$

# Exercise: Confusion Matrix and Classification Metrics Calculation

## Solution

- confusion matrix:
  - True Positives (TP): 40
  - False Positives (FP): 10
  - True Negatives (TN): 30
  - False Negatives (FN): 20
- Calculate Accuracy:  $\text{Accuracy} = \frac{40+30}{40+30+10+20} = 0.7$
- Precision =  $\frac{40}{40+10} = 0.8$
- Recall =  $\frac{40}{40+20} = \frac{2}{3}$
- F-1 Score =  $2 \times \frac{0.8 \times 0.667}{0.8 + 0.667} \approx 0.727$



# Application of logistic regression: Banknote authentication dataset

- The Banknote authentication dataset is used for the task of classifying whether a banknote is authentic or not based on certain features extracted from images.

# Application of logistic regression: Banknote authentication dataset

- The Banknote authentication dataset is used for the task of classifying whether a banknote is authentic or not based on certain features extracted from images.
- Variables meaning

# Application of logistic regression: Banknote authentication dataset

- The Banknote authentication dataset is used for the task of classifying whether a banknote is authentic or not based on certain features extracted from images.
- Variables meaning
  - $\beta_0$ : the intercept term;

# Application of logistic regression: Banknote authentication dataset

- The Banknote authentication dataset is used for the task of classifying whether a banknote is authentic or not based on certain features extracted from images.
- Variables meaning
  - $\beta_0$ : the intercept term;
  - $\beta_1, \beta_2, \beta_3, \beta_4$  are the coefficients associated with each feature.

# Application of logistic regression: Banknote authentication dataset

- The Banknote authentication dataset is used for the task of classifying whether a banknote is authentic or not based on certain features extracted from images.
- Variables meaning
  - $\beta_0$ : the intercept term;
  - $\beta_1, \beta_2, \beta_3, \beta_4$  are the coefficients associated with each feature.
  - $x_1$  represents the Variance of the Wavelet Transformed image.

# Application of logistic regression: Banknote authentication dataset

- The Banknote authentication dataset is used for the task of classifying whether a banknote is authentic or not based on certain features extracted from images.
- Variables meaning
  - $\beta_0$ : the intercept term;
  - $\beta_1, \beta_2, \beta_3, \beta_4$  are the coefficients associated with each feature.
  - $x_1$  represents the Variance of the Wavelet Transformed image.
  - $x_2$  represents the Skewness of the Wavelet Transformed image.

# Application of logistic regression: Banknote authentication dataset

- The Banknote authentication dataset is used for the task of classifying whether a banknote is authentic or not based on certain features extracted from images.
- Variables meaning
  - $\beta_0$ : the intercept term;
  - $\beta_1, \beta_2, \beta_3, \beta_4$  are the coefficients associated with each feature.
  - $x_1$  represents the Variance of the Wavelet Transformed image.
  - $x_2$  represents the Skewness of the Wavelet Transformed image.
  - $x_3$  represents the Kurtosis of the Wavelet Transformed image.

# Application of logistic regression: Banknote authentication dataset

- The Banknote authentication dataset is used for the task of classifying whether a banknote is authentic or not based on certain features extracted from images.
- Variables meaning
  - $\beta_0$ : the intercept term;
  - $\beta_1, \beta_2, \beta_3, \beta_4$  are the coefficients associated with each feature.
  - $x_1$  represents the Variance of the Wavelet Transformed image.
  - $x_2$  represents the Skewness of the Wavelet Transformed image.
  - $x_3$  represents the Curtosis of the Wavelet Transformed image.
  - $x_4$  represents the Entropy of the image.



# Application of logistic regression: Banknote authentication dataset

- The logistic regression model makes the following assumption:

$$P(\text{authentic}) = \frac{1}{1 + \exp\{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4\}}$$

# Application of logistic regression: Banknote authentication dataset

- The logistic regression model makes the following assumption:

$$P(\text{authentic}) = \frac{1}{1 + \exp\{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4\}}$$

- This probability is then used to make a classification decision.

# Application of logistic regression: Banknote authentication dataset

- The logistic regression model makes the following assumption:

$$P(\text{authentic}) = \frac{1}{1 + \exp\{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4\}}$$

- This probability is then used to make a classification decision.
  - If  $P(\text{authentic}) \geq 0.5$ , the model predicts the banknote as authentic (Class 1)

# Application of logistic regression: Banknote authentication dataset

- The logistic regression model makes the following assumption:

$$P(\text{authentic}) = \frac{1}{1 + \exp\{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4\}}$$

- This probability is then used to make a classification decision.
  - If  $P(\text{authentic}) \geq 0.5$ , the model predicts the banknote as authentic (Class 1)
  - If  $P(\text{authentic}) < 0.5$ , the model predicts the banknote as not authentic (Class 0)

# Application of logistic regression: Results

Pls verify confusion matrix, precision, recall.... by hand after class

	Variance	Skewness	Curtosis	Entropy	Class
0	3.62160	8.6661	-2.8073	-0.44699	0
1	4.54590	8.1674	-2.4586	-1.46210	0
2	3.86600	-2.6383	1.9242	0.10645	0
3	3.45660	9.5228	-4.0112	-3.59440	0
4	0.32924	-4.4552	4.5718	-0.98880	0

Accuracy: 0.98

Confusion Matrix:

```
[[144  4]
 [ 2 125]]
```

Classification Report:

	precision	recall	f1-score	support
0	0.99	0.97	0.98	148
1	0.97	0.98	0.98	127
accuracy			0.98	275
macro avg	0.98	0.98	0.98	275
weighted avg	0.98	0.98	0.98	275

# Statsmodels: Python package

- Statsmodels is primarily focused on statistical modeling and hypothesis testing. It provides tools for estimating and testing various statistical models, including linear regression, logistic regression, time-series analysis, and more.

# Statsmodels: Python package

- Statsmodels is primarily focused on statistical modeling and hypothesis testing. It provides tools for estimating and testing various statistical models, including linear regression, logistic regression, time-series analysis, and more.
- Statsmodels includes modules for performing hypothesis tests, constructing confidence intervals, and fitting different types of statistical models with an emphasis on providing detailed statistical information.

# Statsmodels: Python package

- Statsmodels is primarily focused on statistical modeling and hypothesis testing. It provides tools for estimating and testing various statistical models, including linear regression, logistic regression, time-series analysis, and more.
- Statsmodels includes modules for performing hypothesis tests, constructing confidence intervals, and fitting different types of statistical models with an emphasis on providing detailed statistical information.
- Statsmodels is commonly used in academic research, econometrics, and any scenario where a detailed statistical analysis is required.



# Statsmodels: Linear regression example

```
import statsmodels.api as sm
import numpy as np

# Generate some random data for demonstration
np.random.seed(42)
X = np.random.rand(100, 2)
y = 3 * X[:, 0] + 2 * X[:, 1] + 1 + 0.1 * np.random.randn(100)

# Add a constant term to the independent variable
X = sm.add_constant(X)

# Create a Linear model
model = sm.OLS(y, X)
results = model.fit()

# Print detailed statistical summary
print(results.summary())
```

# Statsmodels: output

```

=====
                        OLS Regression Results
=====
Dep. Variable:          y      R-squared:          0.991
Model:                  OLS    Adj. R-squared:      0.991
Method:                 Least Squares    F-statistic:    5655.
Date:                  Thu, 21 Dec 2023    Prob (F-statistic): 3.86e-101
Time:                  10:50:14    Log-Likelihood:  89.304
No. Observations:      100    AIC:             -172.6
Df Residuals:          97    BIC:             -164.8
Df Model:               2
Covariance Type:       nonrobust
=====

```

	coef	std err	t	P> t	[0.025	0.975]
const	0.9772	0.026	37.651	0.000	0.926	1.029
x1	3.0339	0.033	91.615	0.000	2.968	3.100
x2	2.0355	0.035	57.426	0.000	1.965	2.106

```

=====
Omnibus:                5.986    Durbin-Watson:      2.104
Prob(Omnibus):           0.050    Jarque-Bera (JB):    5.624
Skew:                    0.439    Prob(JB):             0.0601
Kurtosis:                3.761    Cond. No.             5.22
=====

```

# Scikit-learn: Python Lib (a collection of Python packages)

- Purpose: Scikit-learn is a versatile machine learning library that provides tools for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and more.

# Scikit-learn: Python Lib (a collection of Python packages)

- Purpose: Scikit-learn is a versatile machine learning library that provides tools for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and more.
- Functionality: Scikit-learn focuses on providing a consistent interface for various machine learning algorithms, making it easy to train models, perform feature engineering, and evaluate model performance.

# Scikit-learn: Python Lib (a collection of Python packages)

- Purpose: Scikit-learn is a versatile machine learning library that provides tools for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and more.
- Functionality: Scikit-learn focuses on providing a consistent interface for various machine learning algorithms, making it easy to train models, perform feature engineering, and evaluate model performance.
- Use Cases: Scikit-learn is widely used in industry for building and deploying machine learning models in areas such as image recognition, natural language processing, and predictive analytics.