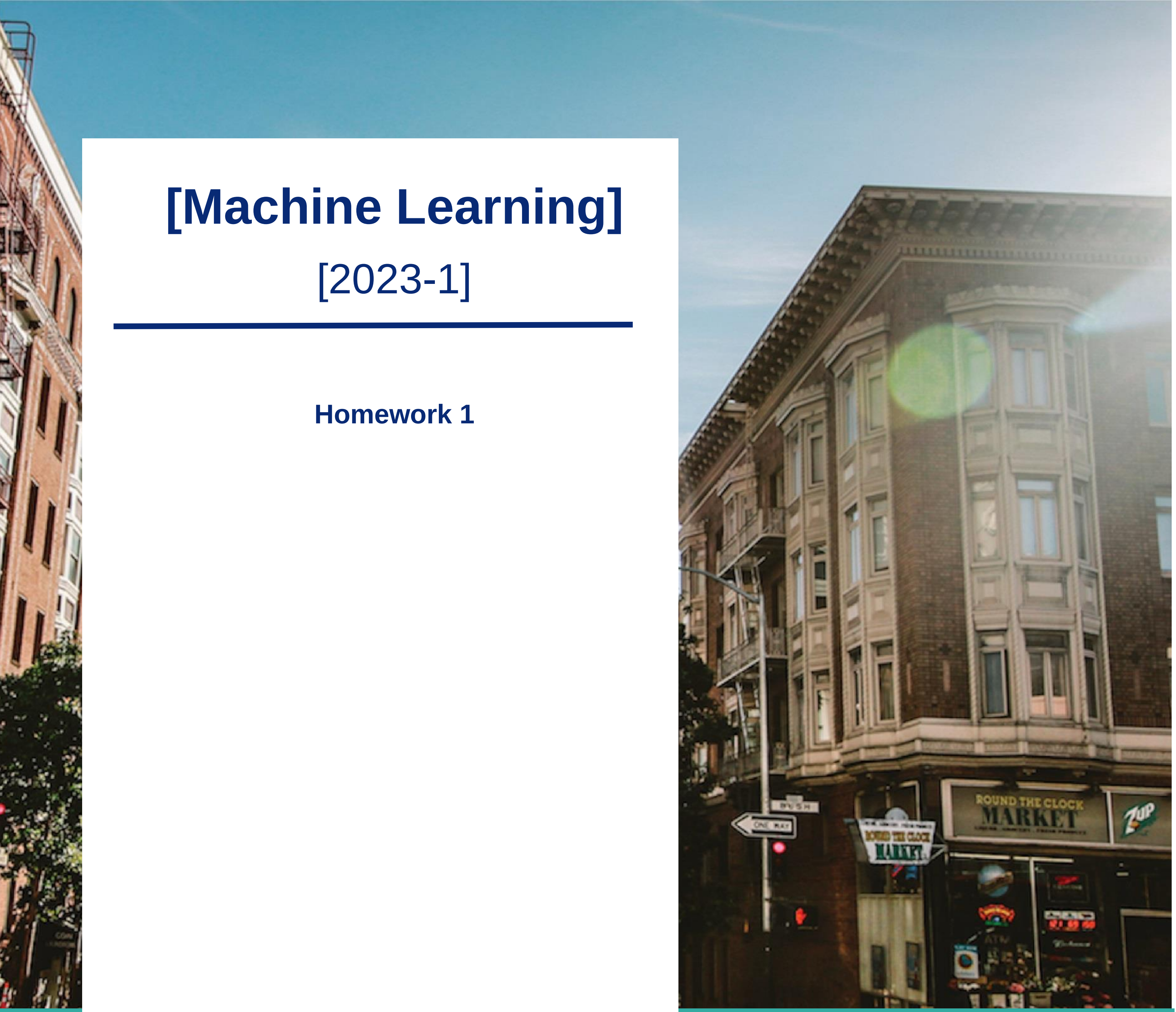


[Machine Learning]

[2023-1]

Homework 1

[Due Date] 2023.04.05

Student ID : 2018112007

Name : 이승현

Professor : Juntae Kim

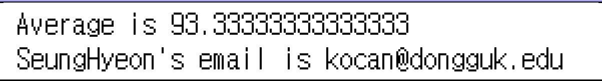
1. Write (python codes to solve each of the following problem, and attach the result and description. (20 pts)

1-1. Design a Student class in Python. It has the attributes **name**, **email**, **math_score**, **science_score**, **english_score**. Add methods **average()** to calculate the average score of a student and **print_email()** to print the email of the student.

Code

```
import numpy as np

class Student:
    def __init__(self, name, email, math_score, science_score, english_score):
        self.name = name
        self.email = email
        self.math_score = math_score
        self.science_score = science_score
        self.english_score = english_score

    def average(self):
        return (self.math_score + self.science_score + self.english_score) / 3

    def print_email(self):
        print(f"{self.name}'s email is {self.email}")

student1 = Student("SeungHyeon", "kocan@dongguk.edu", 95, 95, 90)
print(f"Average is {student1.average()}")
student1.print_email()
```

Result(Captured images)



```
Average is 93.33333333333333
SeungHyeon's email is kocan@dongguk.edu
```

1-2. Numpy:

$$X = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix}, \quad \text{err} = \begin{pmatrix} 0.3 \\ 0.2 \\ 0.1 \end{pmatrix},$$

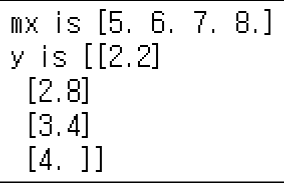
Compute $\text{mx} = (\qquad\qquad\qquad)$ where mx_i = mean of each column of X

Compute $y = \left(\quad \right)$ where $y_j = \sum_i (err_i \cdot x_{ij})$

Code

```
import numpy as np

def compute():
   X = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
   err = np.array([0.3, 0.2, 0.1])
   mx = np.mean(X, axis = 0)
   y = np.dot(err, X).reshape(4, 1)
   print(f"mx is {mx}")
   print(f"y is {y}")

compute()
```

Result(Captured images)



```
mx is [5. 6. 7. 8.]
y is [[2.2]
 [2.8]
 [3.4]
 [4. ]]
```

1-3. Pandas: Read data from From boston.csv (Boston Housing Price dataset), make a dataframe by selecting data with CRIM values < 1.0. Then from this data, compute "MEDV" column's mean, and show the distribution of "MEDV" using a Histogram.

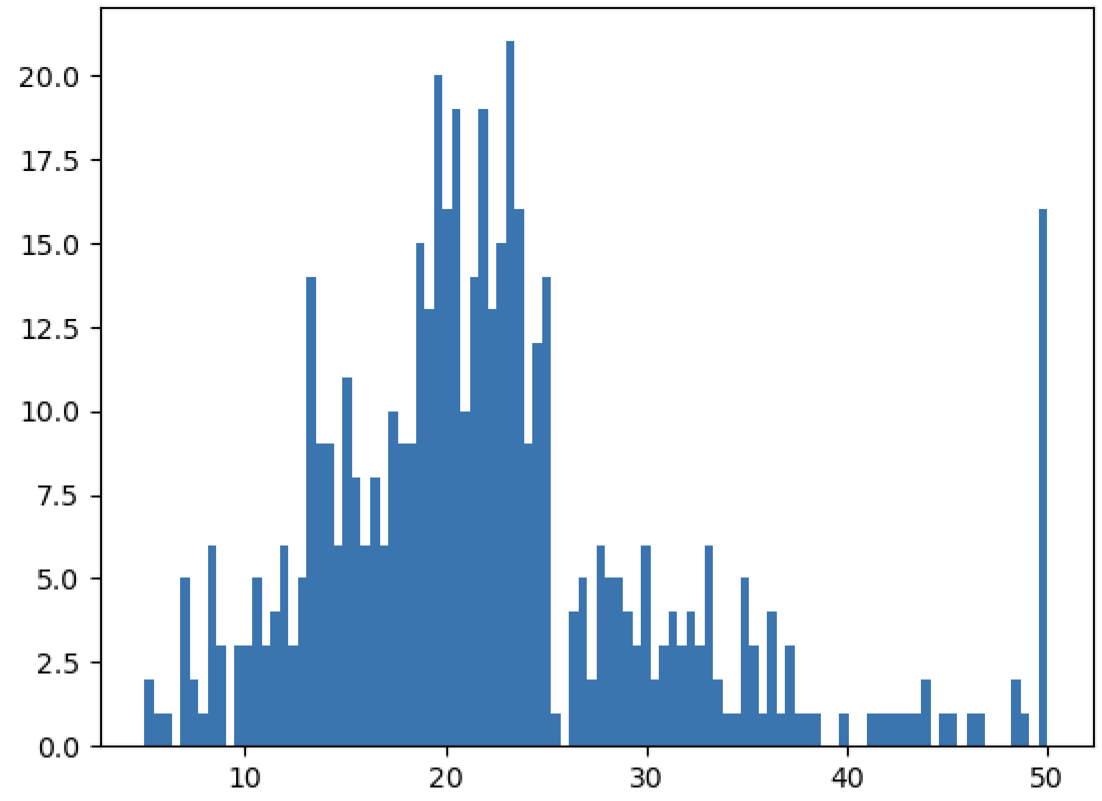
Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

def pandas():
   df = pd.read_csv("boston.csv")
   df.columns = ['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX',
'PTRATIO', 'B', 'LSTAT', 'MEDV']
   df[df['CRIM'] < 1.0]
```

```
    print(f"mean is {np.mean(df.MEDV)}")

    plt.hist(df.MEDV, bins = 100)
    plt.show()

pandas()
```

Result(Captured images)

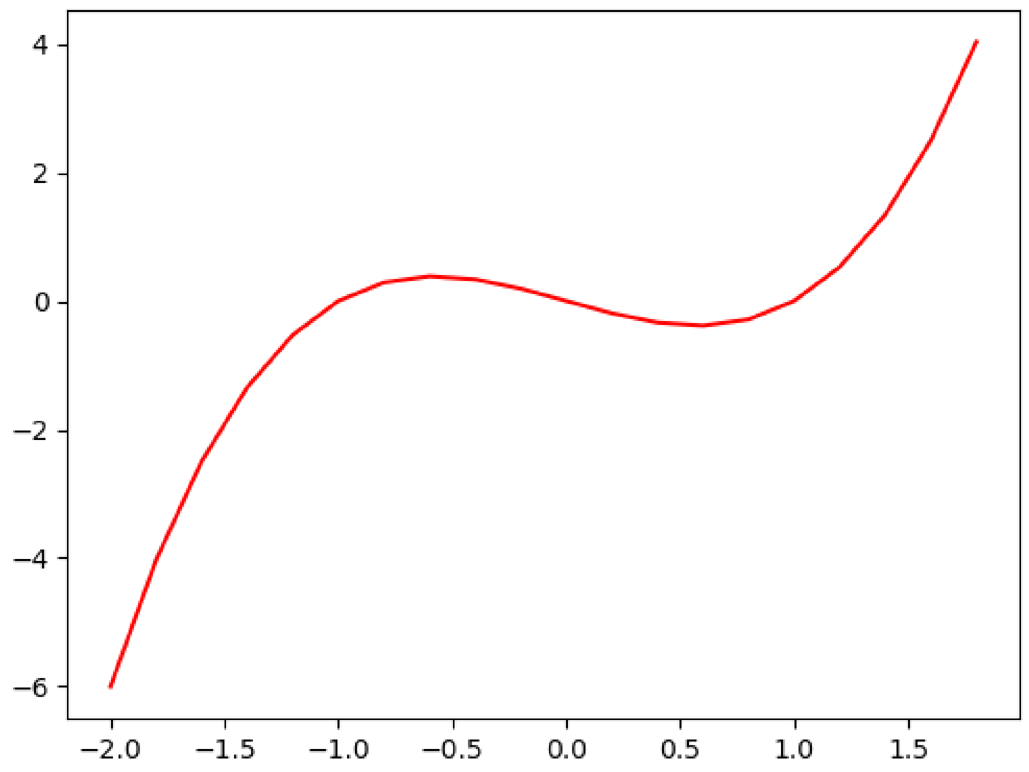
```
mean is 22.532806324110677
```



1-4. Matplotlib : Plot the graph of $y = x^3 - x$ for $x = -2$ to $+2$ with red color.

Code

```
import numpy as np
import matplotlib.pyplot as plt

def graph():
    x = np.arange(-2, 2, 0.2)
    plt.plot(x, x ** 3 - x, color = 'red')
    plt.show()
```

graph()
Result(Captured images)


2. Explain what Supervised Learning, Unsupervised Learning, and Reinforcement Learning are, and describe the differences. (10 pts)

Your Answer
지도 학습 : 지도 학습(Supervised Learning)은 데이터에 대한 레이블(Label)이 주어진 상태에서 컴퓨터를 학습시키는 방법입니다. 즉, (데이터(data), 레이블(label)) 형태로 학습을 진행하는 방법입니다. 지도 학습에는 데이터에 대해 여러 개의 값 중 하나의 답을 도출해내는 분류(classification)와 데이터 분석을 통해 특징으로 답을 도출해내는 회귀(regression)의 방법이 있습니다. 비지도 학습 : 데이터에 대한 레이블(Label)이 주어지지 상태에서 컴퓨터를 학습시키는 방법론입니다. 즉, 데이터에 대한 명시적인 정답 없이 (데이터(data)) 형태로 학습을 진행하는 방법입니다. 대표적인 종류로는 클러스터링(Clustering), Dimensionality Reduction, Hidden Markov Model 등이 있습니다.

강화학습 : 에이전트가 주어진 환경(state)에 대해 어떤 행동(action)을 취하고 이로부터 어떤 보상(reward)을 얻으면서 학습을 진행합니다. 이때, 에이전트는 보상(reward)을 최대화(maximize)하도록 학습이 진행됩니다.

지도 학습과 비지도 학습의 차이는 학습 데이터의 형태로 지도 학습은 학습데이터를 (Data, label) 형태로 제공하고 학습하기 때문에 data 에 따른 label 의 패턴을 학습하고 예측을 하는 반면에, 비지도 학습은 (data) 형태로 제공하기 때문에, data 의 패턴만 학습하여 예측을 하거나 결과 값을 만들어냅니다. 그리고 지도 학습과 비지도 학습은 환경에 변화가 없는 정적인 환경에서 학습 데이터가 주어져 학습을 진행한다면, 강화학습은 변화되는 환경으로부터 reward 를 받아 학습한다는 점에서 차이를 보입니다.

3. Describe the concept of "overfitting", and explain how you can prevent overfitting in supervised learning. (10 pts)

Your Answer
overfitting 은 학습 데이터를 과하게 학습시켜 학습데이터에는 정확도가 높지만, 테스트 데이터에 대해서는 오차가 증가하는 현상입니다. Overfitting 을 막는 방법은 1. 데이터의 양을 늘려 데이터의 일반적인 패턴을 학습시킨다. 2. 모델의 complexity 를 줄입니다. 3. Regularization 을 적용하여 weight 의 수를 줄이고 복잡도를 감소시킵니다. 4. 드롭 아웃을 적용하여 신경망의 일부를 사용하지 않습니다.

4. Describe the sigmoid function $y = \phi(x) = ?$ and show that $dy/dx = y\,(1 - y)$. (10 pts)

Your Answer
y = $\phi(x)$ = 1 / (1 + e^(-x)) dy / dx = e^(-x) / ((1 + e^(-x)) ^ 2) = 1 + e^(-x) – 1 / ((1 + e^(-x)) ^ 2) = (1 + e^(-x) / ((1 + e^(-x)) ^ 2) – 1 / ((1 + e^(-x)) ^ 2) = 1 / (1 + e^(-x)) -1 / ((1 + e^(-x)) ^ 2) = 1 / (1 + e^(-x)) (1 - 1 / (1 + e^(-x))) = y(1 – y)

$$y = \phi(x) = \frac{1}{1+e^{-x}}$$

$$\frac{dy}{dx} = -\frac{-e^{-x}}{(1+e^{-x})^2} = \frac{e^{-x}}{(1+e^{-x})^2}$$

$$= \frac{1+e^{-x}-1}{(1+e^{-x})^2}$$

$$= \frac{1+e^{-x}}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2}$$

$$= \frac{1}{(1+e^{-x})} - \frac{1}{(1+e^{-x})^2}$$

$$= \frac{1}{(1+e^{-x})}\left(1 - \frac{1}{(1+e^{-x})^2}\right)$$

$$= y(1-y)$$

5. For $\mathrm{X} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 2 \end{bmatrix}, \mathrm{y} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \mathrm{w} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, b = 0,$

Compute followings by hand: (20 pts)

$$\hat{\mathbf{y}} = sigmoid(\mathbf{X} \cdot \mathbf{w} + b)$$

$$\frac{\partial J}{\partial \mathbf{w}} = \frac{1}{m}\mathbf{X}^{\mathrm{T}}(\hat{\mathbf{y}} - \mathbf{y})$$

$$X \cdot W = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1\times 1 + 2\times 0 \\ 2\times 1 + 3\times 0 \\ 3\times 1 + 2\times 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$X \cdot W + b = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + 0 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$\hat{y} = \begin{bmatrix} \frac{1}{1+e^{-1}} \\ \frac{1}{1+e^{-2}} \\ \frac{1}{1+e^{-3}} \end{bmatrix} = \begin{bmatrix} 0.73 \\ 0.88 \\ 0.95 \end{bmatrix}$$

$$X^T = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 2 \end{bmatrix}, \quad \hat{y} - y = \begin{bmatrix} 0.73 - 1 \\ 0.88 - 1 \\ 0.95 - 0 \end{bmatrix}$$

$$= \begin{bmatrix} -0.27 \\ -0.12 \\ 0.95 \end{bmatrix}$$

$$X^T \cdot (\hat{y} - y) = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 2 \end{bmatrix} \cdot \begin{bmatrix} -0.27 \\ -0.12 \\ 0.95 \end{bmatrix}$$

$$= \begin{bmatrix} 1\times(-0.27) + 2\times(-0.12) + 3\times 0.95 \\ 2\times(-0.27) + 3\times(-0.12) + 2\times 0.95 \end{bmatrix}$$

$$= \begin{bmatrix} 2.34 \\ 1 \end{bmatrix}$$

$$\frac{\partial J}{\partial W} = \frac{1}{3} \times \begin{bmatrix} 2.34 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.78 \\ 0.33 \end{bmatrix}$$

6. The heart_disease.csv dataset represents 13 attributes of a patient and the presence of heart disease. Meaning of attributes are as below. The ‘num’ is the target value, 0 mean no disease, 1~4 means different types of disease.

- age: age in years

- sex: sex (1 = male; 0 = female)
- cp: chest pain type
- trestbps: resting blood pressure (in mm Hg on admission to the hospital)
- chol: serum cholestoral in mg/dl
- fbs: fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
- restecg: resting electrocardiographic results
  (0: normal, 1: ST-T wave abnormality, 2: left ventricular hypertrophy)
- thalach: maximum heart rate achieved
- exang: exercise induced angina (1 = yes; 0 = no)
- oldpeak = ST depression induced by exercise relative to rest
- slope: the slope of the peak exercise ST segment (1: upsloping, 2: flat, 3: downsloping)
- ca: number of major vessels (0-3) colored by flourosopy
- thal: 3 = normal; 6 = fixed defect; 7 = reversable defect
- num: diagnosis of heart disease

Change the dataset for binary classification (change 1~4 values of 'num' to 1), then perform logistic regression and show 1) the cost function graph, 2) learned model, 3) training accuracy of the model, 4) prediction result for the patient with attribute values of [61, 0, 3, 154, 210, 1, 0, 130, 0, 1.5, 2, 2, 3]. Do NOT use scikit learn library. (30 pts)

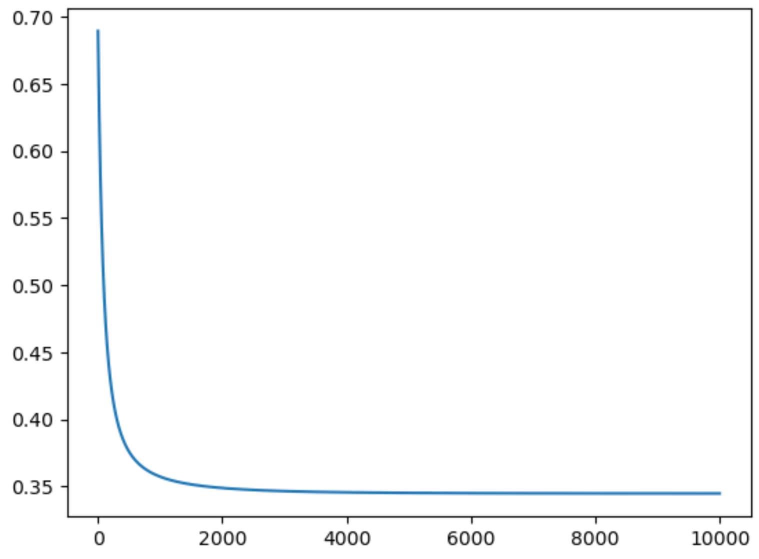
Code
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import axes3d

df = pd.read_csv("heart_disease.csv")
origin = df.values

X_train = origin[:, 0:13]
X_train = (X_train - np.mean(X_train, axis = 0)) / np.std(X_train, axis = 0)
y_train = origin[:, 13]
y_train[y_train > 1] = 1
```

```
def sigmoid(z):
    y_hat = 1. / (1 + np.exp(-np.clip(z, -250, 250)))
    return y_hat

def predict(x, w, b):
    y_hat = sigmoid(np.dot(x, w) + b)
    y = np.where(y_hat >= 0.5, 1, 0)
    return y

def compute_cost(X ,y, w, b):
    m = X.shape[0]
    cost = 0.0

    z = np.dot(X, w) + b
    y_hat = sigmoid(z)
    cost = -y * np.log(y_hat+1e-7) - (1-y) * np.log(1-y_hat+1e-7)
    cost = np.sum(cost) / m
    return cost

def compute_gradient(X, y, w, b):
    m, n = X.shape
    dj_dw = np.zeros((n, ))
    dj_db = 0.

    y_hat = sigmoid(np.dot(X, w) + b)
    err = y_hat - y
    dj_dw = np.dot(X.T, err) / m
    dj_db = np.sum(err) / m
    return dj_dw, dj_db

def gradient_descent(X, y, w, b, alpha, num_iters):
    J_hist = []

    for i in range(num_iters):
        dj_dw, dj_db = compute_gradient(X, y, w, b)
        w = w - alpha * dj_dw
        b = b - alpha * dj_db
        J_hist.append(compute_cost(X, y, w, b))
    return w, b, J_hist

w_init = np.zeros(X_train.shape[1])
b_init = 0.

alpha = 0.01
iterations = 10000

w_final, b_final, J_hist = gradient_descent(X_train, y_train, w_init, b_init, alpha,
iterations)

plt.plot(J_hist[:10000])

fig, ax = plt.subplots(subplot_kw={"projection": "3d"})

# plot the data points
ax.scatter(X_train[y_train == 0, 0], X_train[y_train == 0,4], y_train[y_train == 0],
marker='o', c='red')
ax.scatter(X_train[y_train == 1, 0], X_train[y_train == 1,4], y_train[y_train == 1],
marker='x', c='blue')

# compute y_hat for all meshgrid using learned w and b
x0 = np.arange(-3, 3, 0.1)
x1 = np.arange(-6, 6, 0.1)
x0, x1 = np.meshgrid(x0, x1)
y_hat = sigmoid(x0 * w_final[0] + x1 * w_final[4] + b_final)

# show the model by plotting y_hat
ax.plot_surface(x0, x1, y_hat, cmap=cm.coolwarm, alpha=0.5)

plt.xlabel('age')
plt.ylabel('chol')
plt.title('Prob. of y = 1')
plt.show()

y_pred = predict(X_train, w_final, b_final)
accuracy = np.sum(y_train == y_pred)/len(y_train)
```

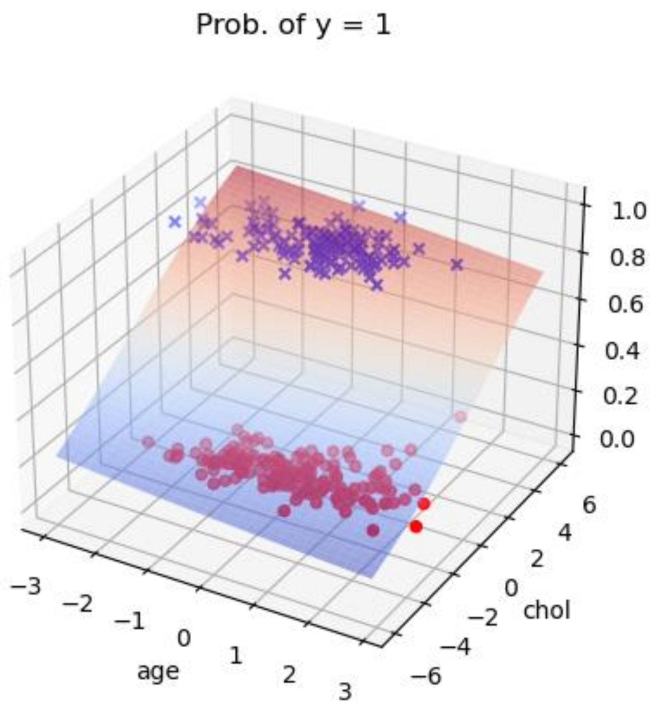
```
print("Accuracy on the training set =", accuracy)

X_test = np.array([61, 0, 3, 154, 210, 1, 0, 130, 0, 1.5, 2, 2, 3])
y_pred = predict(X_test, w_final, b_final)
print('class prediction = ', y_pred)
```

Result(Captured images)

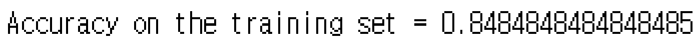
1) the cost function graph



2) learned model

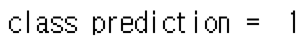


3) training accuracy of the model



```
Accuracy on the training set = 0.8484848484848485
```

4) prediction result for the patient with attribute values of [61, 0, 3, 154, 210, 1, 0, 130, 0, 1.5, 2, 2, 3]



```
class prediction =  1
```

Note

1. Submit the file to e-class as pdf
2. Specify your pdf file name as “hw1_<StudentID>_<Name>.pdf”

   Ex) hw1_2000123456_홍길동.pdf