Kingdom of Saudi Arabia
Ministry of Education
University of Jeddah
College of Computer Science and Engineering
Department of Computer Science and Artificial Intelligence

المملكة العربية السعودية
وزارة التعليم
جامعة جدّة
كلية علوم وهندسة الحاسب
قسم علوم الحاسب والذكاء الاصطناعي

Lab 8
CCAI 312 Pattern Recognition
Third Trimester 2023

Student Name: Ruba Khalid Alsulami

Student ID: 2110618

Part 1

Lab Assessment

Step1: Create a new notebook and name it "CCAI312_YOURSTUDENTID_Lab8_p1"

Step2: Generate the following data
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1)

```
[1] import numpy as np
    X = 2 * np.random.rand(100, 1)
    y = 4 + 3 * X + np.random.randn(100, 1)
```

Step3: Implement the mini-batch gradient descent algorithm,

```
def mini_batch_gradient_descent(X, y, batch_size=20, learning_rate=0.01, num_iterations=1000):
    # Initialize parameters
    b = 0
    m = 0

    # Loop over number of iterations
    for i in range(num_iterations):
        # Randomly select a batch of data points
        indices = np.random.randint(0, len(X), batch_size)
        X_batch = X[indices]
        y_batch = y[indices]

        # Compute gradients
        b_gradient = np.mean(2 * (m * X_batch + b - y_batch))
        m_gradient = np.mean(2 * X_batch * (m * X_batch + b - y_batch))

        # Update parameters
        b = b - learning_rate * b_gradient
        m = m - learning_rate * m_gradient

    return b, m
```

Step4: Report the learned parameters when the mini-batch size is 20. Repeat this step with 2 different mini-batch sizes and report the learned parameters.

```
b, m = mini_batch_gradient_descent(X, y, batch_size=20)
print("Learned Parameters (Batch Size = 20):")
print("Intercept: ", b)
print("Slope: ", m)
print("\n")
b, m = mini_batch_gradient_descent(X, y, batch_size=10)
print("Learned Parameters (Batch Size = 10):")
print("Intercept: ", b)
print("Slope: ", m)
print("\n")
b, m = mini_batch_gradient_descent(X, y, batch_size=50)
print("Learned Parameters (Batch Size = 50):")
print("Intercept: ", b)
print("Slope: ", m)
```

```
Learned Parameters (Batch Size = 20):
Intercept:  3.8875879457943863
Slope:  3.0491010484331458
```

```
Learned Parameters (Batch Size = 10):
Intercept:  3.89971722200207
Slope:  3.009020014856965
```

```
Learned Parameters (Batch Size = 50):
Intercept:  3.9014029471949856
Slope:  3.047229297611564
```

Step6: Submit a pdf document containing **your code and answers** to Blackboard, name the file as: CCAI312_YOURSTUDENTID_Lab6.pdf.

Part 2

Lab Assessment

1. Import the necessary libraries, including `pyplot` from `matplotlib` and `train_test_split` from `sklearn.model_selection`

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

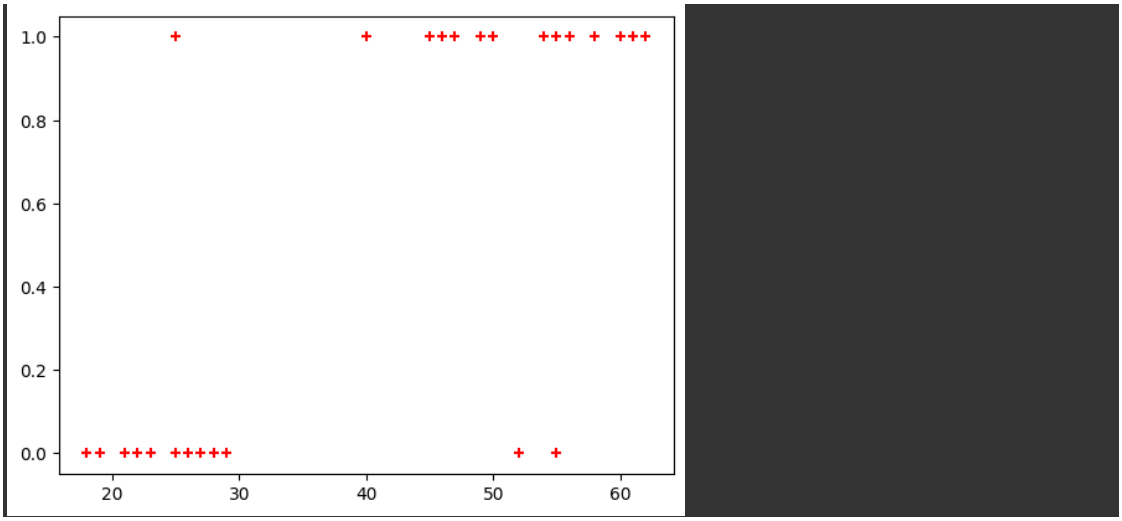
2. Load the insurance dataset "insurance_data.csv" and show the head of the it.

```
df = pd.read_csv("insurance_data.csv")
df
```

	age	bought_insurance
0	22	0
1	25	0
2	47	1
3	52	0
4	46	1
5	56	1
6	55	0
7	60	1
8	62	1
9	61	1
10	18	0
11	28	0
12	27	0
13	29	0
14	49	1
15	55	1
16	25	1
17	58	1
18	19	0
19	18	0
20	21	0
21	26	0
22	40	1
23	45	1
24	50	1
25	54	1
26	23	0

3. Plot the dataset to get an idea of the data and how it is distributed

```
plt.scatter(df.age,df.bought_insurance,marker='+',color='
red')
```

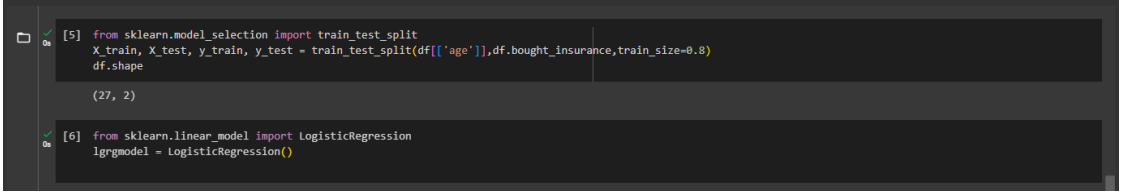


4. Import “train_test_split” package and split the data into training/testing sets and split the targets into training/testing sets

```
from sklearn.model_selection import train_test_split
df.shape
X_train, X_test, y_train, y_test =
train_test_split(df[['age']],df.bought_insurance,train_si
ze=0.8)
(27, 2)
```

5. Import the logistic regression package and create Logistic regression object.

```
from sklearn.linear_model import LogisticRegression
lgrgmodel = LogisticRegression()
```



6. Train the model using the training sets.

```
lgrgmodel.fit(X_train, y_train)
```

7. Make predictions using the testing set and show the results. Note that 1: bought insurance and 0: didn't bought insurance

```
lgrg_pred =lgrgmodel.predict(X_test)
```

8. Calculate the score (Accuracy as it is the default metric) of the test set.