Lab-3

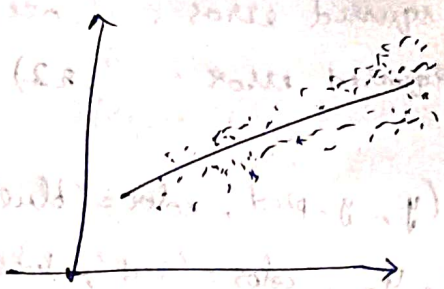
(1) Linear Regression

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

def estimate_coef(x,y):
    n = np.size(x)
    m_x = np.mean(x)
    m_y = np.mean(y)
    SS_xy = np.sum((x-m_x)*(y-m_y))
    SS_xx = np.sum((x-m_x)**2)
    b_1 = SS_xy / SS_xx
    b_0 = m_y - b_1*m_x
    return (b_0, b_1)

def plot_regression_line(x,y,b):
    plt.scatter(x,y, color = 'm', marker = "o", s=30)
    y_pred = b[0] + b[1]*x
    plt.plot(x, y_pred, color = 'g')
    plt.xlabel('x')
    plt.ylabel('y')
    plt.title('Linear Regression')
    plt.show()

file_path = input("Enter the path to csv file: ")
df = pd.read_csv(filepath)

x = df.iloc[:, 0].values
y = df.iloc[:, 1].values

b = estimate_coef(x,y)
print(f"Estimated coefficients: \n b_0 = {b[0]}. \n
                                 b_1 = {b[1]}")

plot_regression_line(x,y,b)
```

Output:

Enter the path to the csv file
/content/tvmarketing.csv

b_0 = 7.032593
b_1 = 0.047536



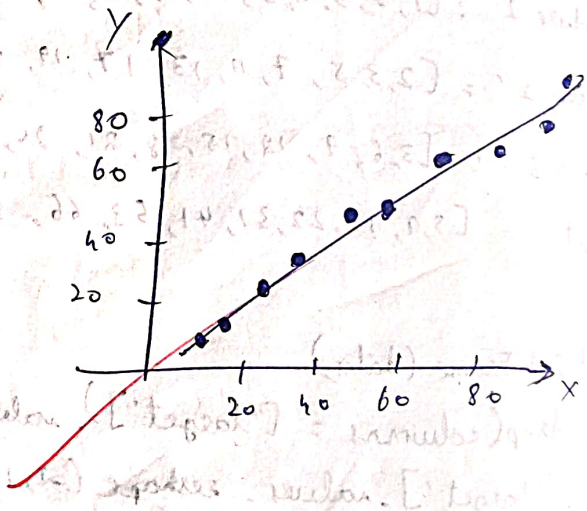
Multiple linear regression

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

data = {
    "Feature 1" : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    "Feature 2" : [2, 3, 5, 7, 11, 13, 17, 19, 23, 29],
    "Feature 3" : [3, 6, 9, 12, 15, 18, 21, 24, 27, 30],
    "Target" : [5, 9, 15, 22, 31, 41, 53, 66, 80, 96]
}

df = pd.DataFrame(data)
X = df.drop(columns = ['target']).values
Y = df['target'].values.reshape(-1, 1)

X = np.hstack((np.ones(X.shape[0], 1)), X))
beta = np.linalg.solve(X.T @ X + 0.01 * np.identity(X.shape[1]),
                       X.T @ y)

y_pred = X @ beta

mse = np.mean((y - y_pred) ** 2)
total_var = np.sum((y - np.mean(y)) ** 2)
exp_var = np.sum((y_pred - np.mean(y)) ** 2)
R2 = exp_var / total_var
```

```
print ("Model coefficients : ", beta[1:].flatten())
print ("Intercept : ", beta[0][0])
print ("Mean squared error : ", mse)
print ("R-squared error : ", r2)

plt.scatter(y, y_pred, color='blue')
plt.plot(y, y, color='red', linestyle='--')
plt.xlabel("Actual values")
plt.ylabel("predicted values")
plt.show()
```

Output :-

Model coefficients : [0.402 3.313 0.120]
Intercept : -3.159950
MSE : 5.36291281
R-square : 0.99353281



3. Logistic regression

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

def sigmoid(z):
    return 1/(1+np.exp(-z))

def compute_cost(x, y, theta):
    m = len(y)
    h = sigmoid(x @ theta)
    cost = (-1/m)*np.sum(y*log(h) + (1-y)*
    np.log(1-h)
    return cost
```

```
def gradient_descent(x, y, theta, alpha, iteration):
    m = len(y)
    cost_history = []
    for _ in range(it):
        grad = (1/m) * T@(sigmoid(x@theta)-y)
        theta -= alpha grad
    return theta, cost_history

def predict(x, theta):
    return (sigmoid(x@theta) >= 0.5), astype(int)

accuracy = np.mean(y_pred == y)
print("Accuracy: {accuracy:.2f}")
```

Output:

Accuracy: 1