Report

Name: Alberto Hernandez

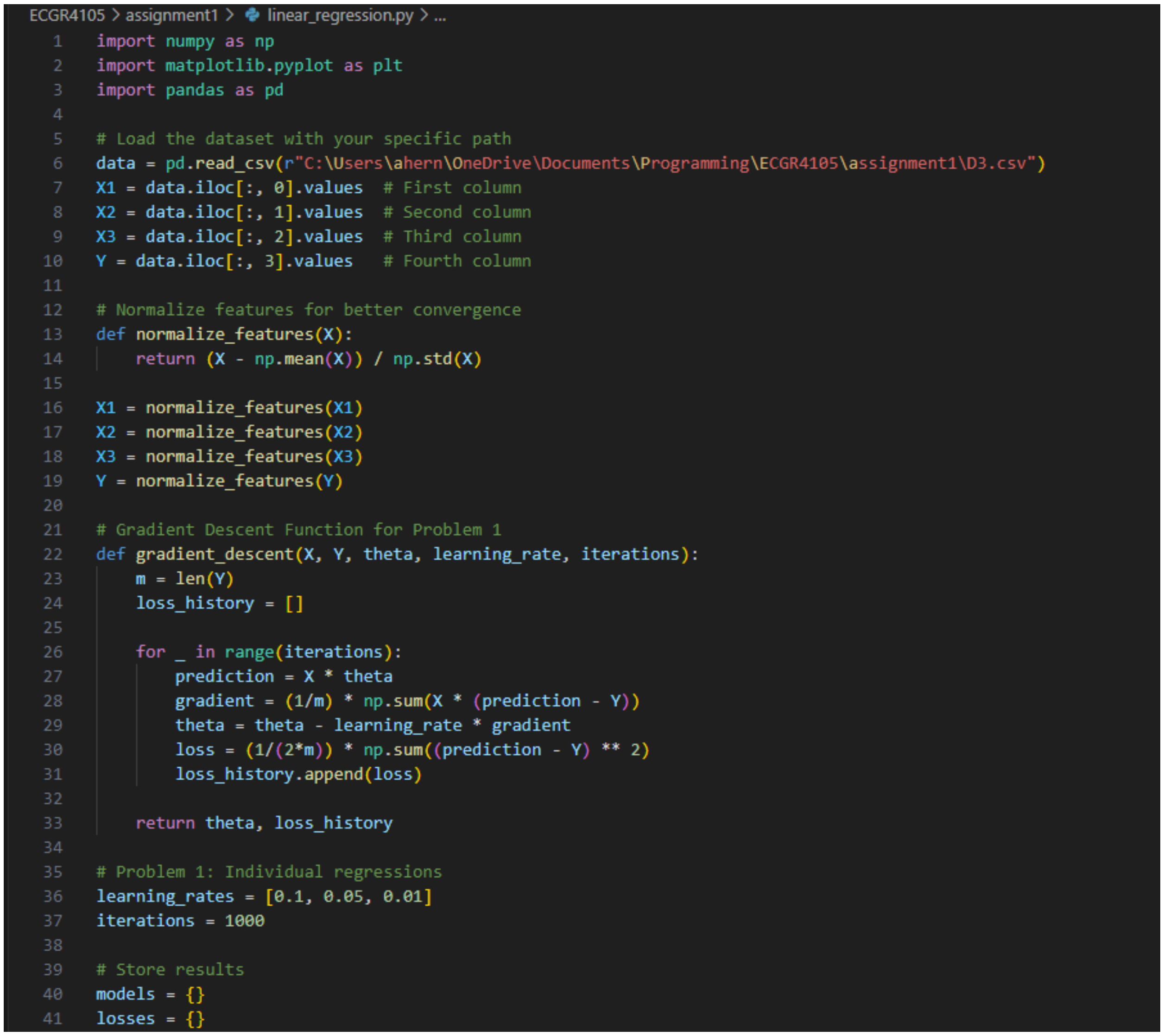
Student ID: 801294761

Assignment Number: 1

GitHub Repository: https://github.com/alberto-hdz/ECGR4105-Assignment1.git

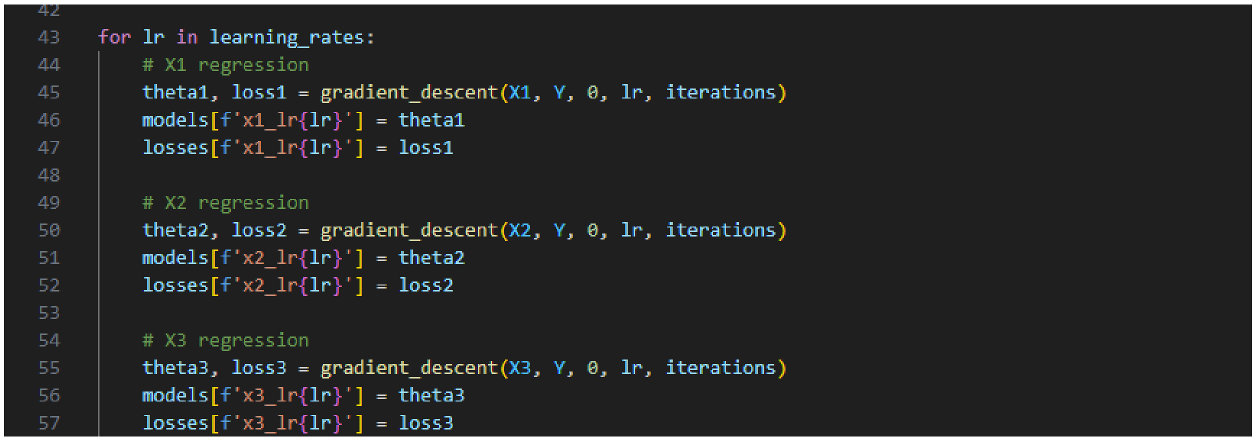
Problem 1

Source Code



ECGR4105 > assignment1 > linear_regression.py > ...

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

# Load the dataset with your specific path
data = pd.read_csv(r"C:\Users\ahern\OneDrive\Documents\Programming\ECGR4105\assignment1\D3.csv")
X1 = data.iloc[:, 0].values  # First column
X2 = data.iloc[:, 1].values  # Second column
X3 = data.iloc[:, 2].values  # Third column
Y = data.iloc[:, 3].values   # Fourth column

# Normalize features for better convergence
def normalize_features(X):
    return (X - np.mean(X)) / np.std(X)

X1 = normalize_features(X1)
X2 = normalize_features(X2)
X3 = normalize_features(X3)
Y = normalize_features(Y)

# Gradient Descent Function for Problem 1
def gradient_descent(X, Y, theta, learning_rate, iterations):
    m = len(Y)
    loss_history = []

    for _ in range(iterations):
        prediction = X * theta
        gradient = (1/m) * np.sum(X * (prediction - Y))
        theta = theta - learning_rate * gradient
        loss = (1/(2*m)) * np.sum((prediction - Y) ** 2)
        loss_history.append(loss)

    return theta, loss_history

# Problem 1: Individual regressions
learning_rates = [0.1, 0.05, 0.01]
iterations = 1000

# Store results
models = {}
losses = {}
```



```python
for lr in learning_rates:
    # X1 regression
    theta1, loss1 = gradient_descent(X1, Y, 0, lr, iterations)
    models[f'x1_lr{lr}'] = theta1
    losses[f'x1_lr{lr}'] = loss1

    # X2 regression
    theta2, loss2 = gradient_descent(X2, Y, 0, lr, iterations)
    models[f'x2_lr{lr}'] = theta2
    losses[f'x2_lr{lr}'] = loss2

    # X3 regression
    theta3, loss3 = gradient_descent(X3, Y, 0, lr, iterations)
    models[f'x3_lr{lr}'] = theta3
    losses[f'x3_lr{lr}'] = loss3
```

Results

The linear models found for each variable at different learning rates are as follows:

Learning Rate 0.1:

- X1: y = -0.8611x1
- X2: y = 0.2356x2
- X3: y = -0.2182x3

Learning Rate 0.05:

- X1: y = -0.8611x1
- X2: y = 0.2356x2
- X3: y = -0.2182x3

Learning Rate 0.01:

- X1: y = -0.8611x1
- X2: y = 0.2356x2
- X3: y = -0.2182x3

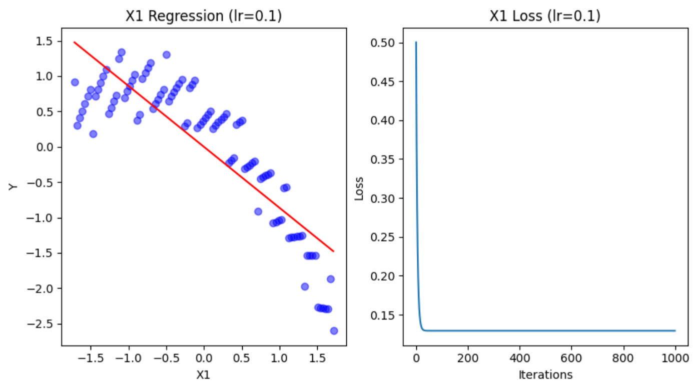
Final Losses:

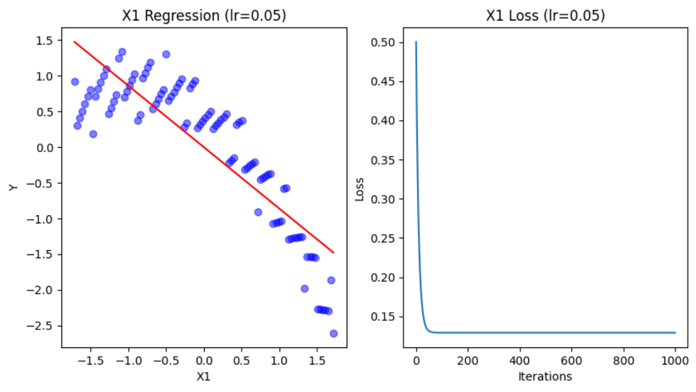
- Learning Rate 0.1: X1 = 0.1292, X2 = 0.4723, X3 = 0.4762
- Learning Rate 0.05: X1 = 0.1292, X2 = 0.4723, X3 = 0.4762
- Learning Rate 0.01: X1 = 0.1292, X2 = 0.4723, X3 = 0.4762

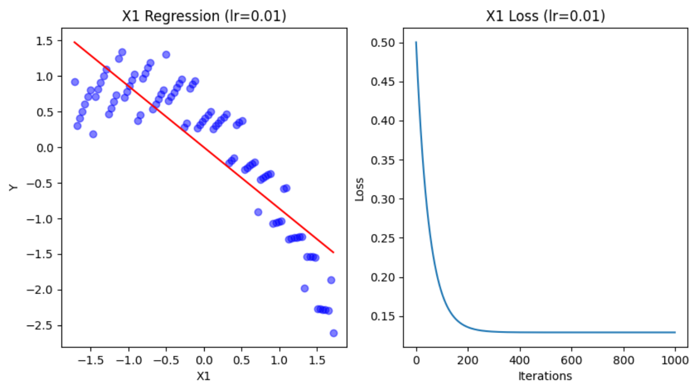
Plots

The figures below display the final regression model (left subplot) and loss over iterations (right subplot) for each variable at the specified learning rates:

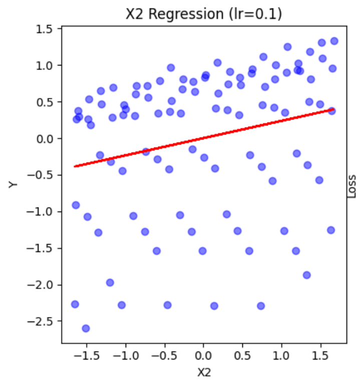
X1 Figures:

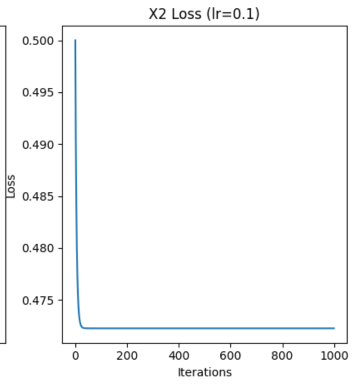


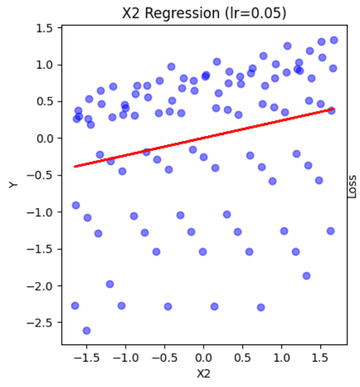


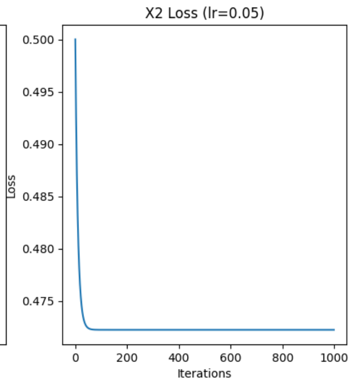


X2 Figures:

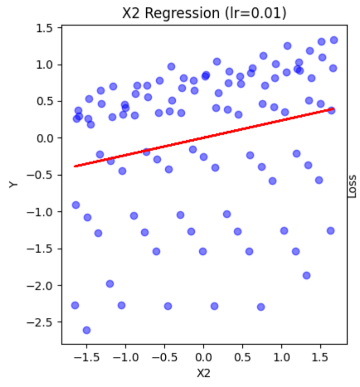

X2 Regression (lr=0.1)
Y
X2
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5

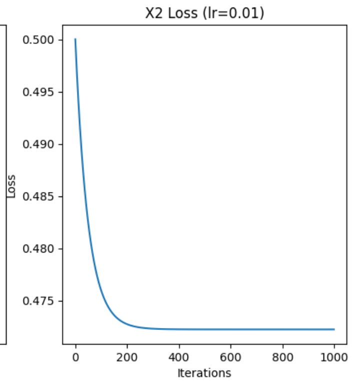

X2 Loss (lr=0.1)
Loss
Iterations
0.500
0.495
0.490
0.485
0.480
0.475
0 200 400 600 800 1000


X2 Regression (lr=0.05)
Y
X2
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5

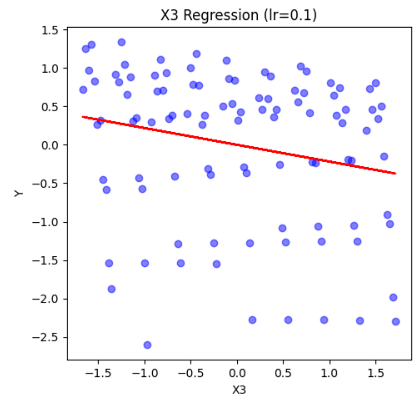

X2 Loss (lr=0.05)
Loss
Iterations
0.500
0.495
0.490
0.485
0.480
0.475
0 200 400 600 800 1000

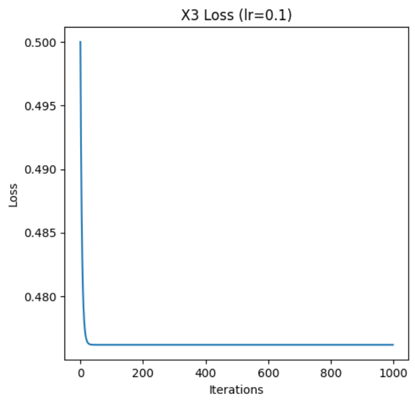
-

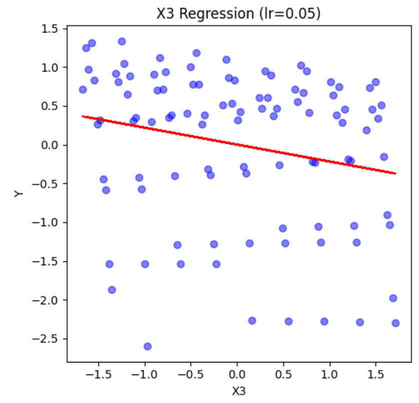

X2 Regression (lr=0.01)
Y
X2
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5

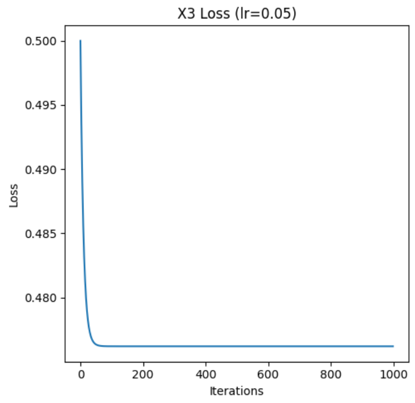

X2 Loss (lr=0.01)
Loss
Iterations
0.500
0.495
0.490
0.485
0.480
0.475
0 200 400 600 800 1000

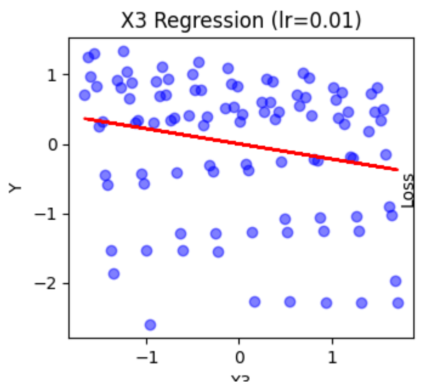
X3 Figures:

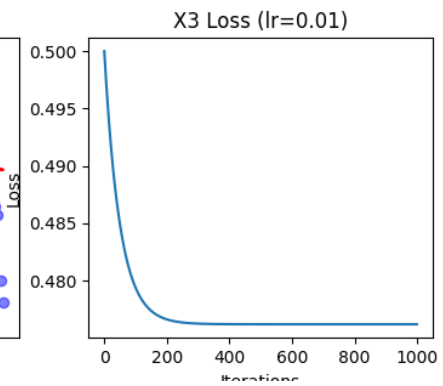

X3 Regression (lr=0.1)
Y
X3
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


X3 Loss (lr=0.1)
Loss
Iterations
0.500
0.495
0.490
0.485
0.480
0
200
400
600
800
1000


X3 Regression (lr=0.05)
Y
X3
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


X3 Loss (lr=0.05)
Loss
Iterations
0.500
0.495
0.490
0.485
0.480
0
200
400
600
800
1000


X3 Regression (lr=0.01)
Y
1
0
−1
−2
−1
0
1


X3 Loss (lr=0.01)
Loss
0.500
0.495
0.490
0.485
0.480
0
200
400
600
800
1000

Problem 1 Questions

1. Linear Model for Each Explanatory Variable:

The models for each variable are reported above under "Results." For all learning rates (0.1, 0.05, 0.01), the models are:

X1: y = -0.8611x1

X2: y = 0.2356x2

X3: y = -0.2182x3

These were derived using gradient descent with theta initialized to zero, trained separately for each variable.

2. Plot the Final Regression Model and Loss Over Iteration:

The figures listed above show the regression lines (left subplots) and loss curves (right subplots) for X1, X2, and X3 at learning rates 0.1, 0.05, and 0.01.

3. Which Explanatory Variable Has the Lowest Loss (Cost) for Explaining the Output (Y)?

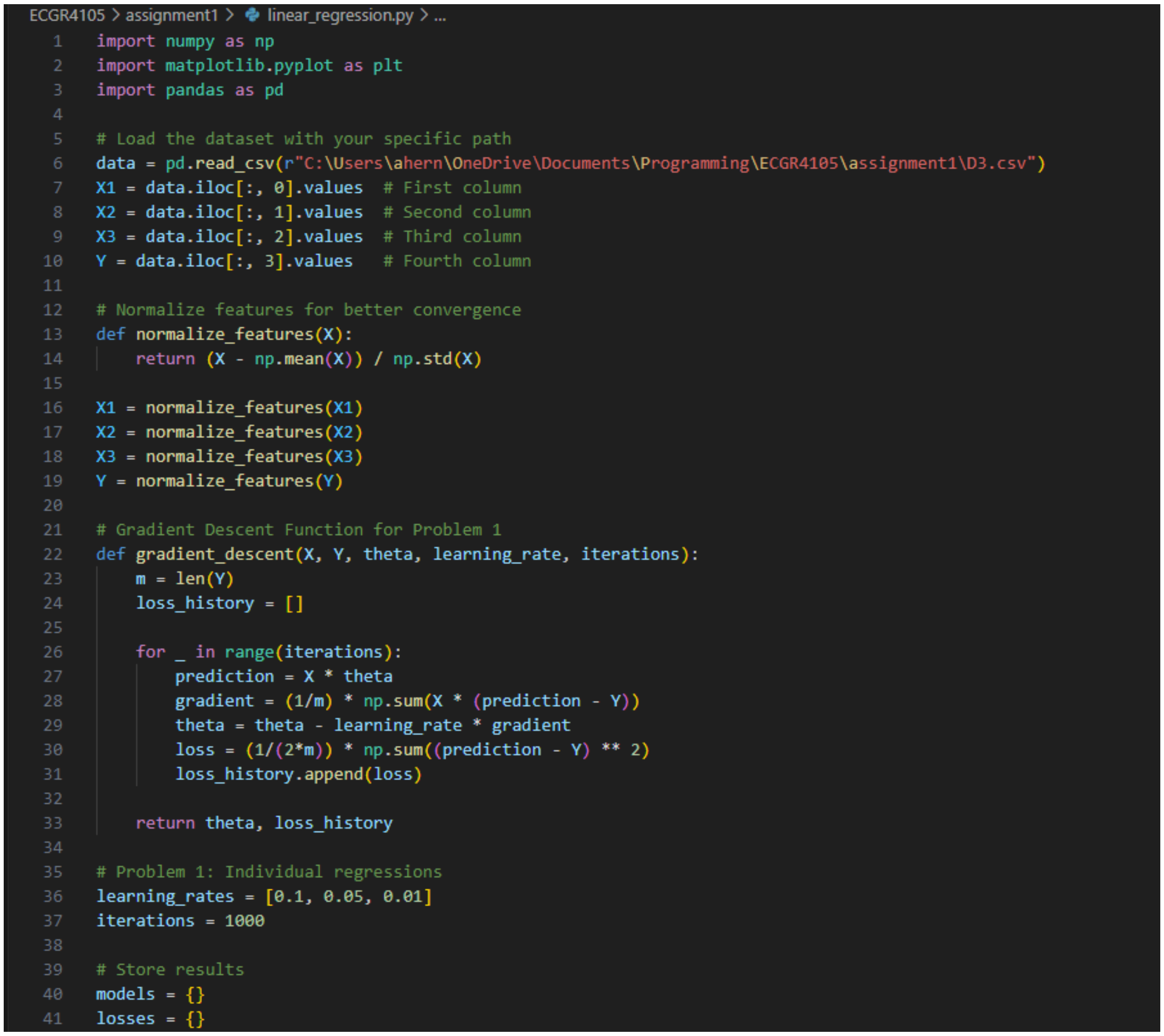
The explanatory variable X1 has the lowest loss of 0.1292 across all learning rates (0.1, 0.05, 0.01). Compared to X2 (loss = 0.4723) and X3 (loss = 0.4762), X1 explains the most variance in Y, making it the strongest predictor.

4. Impact of Different Learning Rates on Final Loss and Number of Training Iterations:

The final loss values and theta coefficients are the same across all learning rates (0.1, 0.05, 0.01) for each variable: X1 = 0.1292, X2 = 0.4723, X3 = 0.4762. This suggests that gradient descent converged to the optimal solution quickly regardless of the learning rate. At learning rate 0.1, the loss stabilized around 100-200 iterations, reflecting fast convergence due to larger steps. At learning rate 0.05, stabilization occurred around 150-300 iterations, slightly slower but still efficient. At learning rate 0.01, the loss took longer, stabilizing around 200-400 iterations, as smaller steps slowed the process. Higher learning rates (0.1) reduce the number of iterations needed for convergence without affecting the final loss, while lower rates (0.01) increase the iteration count but achieve the same result.

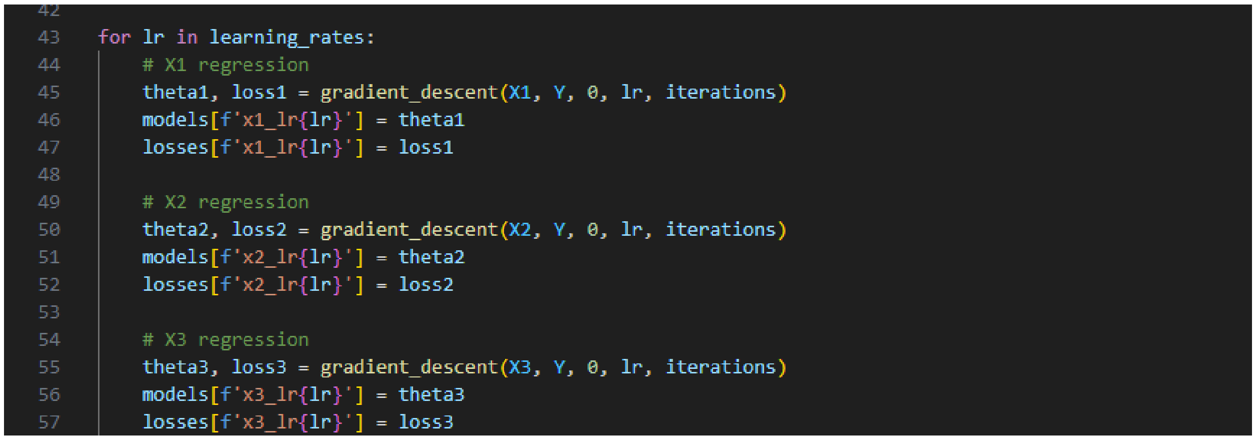
Problem 2

Source Code



ECGR4105 > assignment1 > linear_regression.py > ...

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

# Load the dataset with your specific path
data = pd.read_csv(r"C:\Users\ahern\OneDrive\Documents\Programming\ECGR4105\assignment1\D3.csv")
X1 = data.iloc[:, 0].values  # First column
X2 = data.iloc[:, 1].values  # Second column
X3 = data.iloc[:, 2].values  # Third column
Y = data.iloc[:, 3].values   # Fourth column

# Normalize features for better convergence
def normalize_features(X):
    return (X - np.mean(X)) / np.std(X)

X1 = normalize_features(X1)
X2 = normalize_features(X2)
X3 = normalize_features(X3)
Y = normalize_features(Y)

# Gradient Descent Function for Problem 1
def gradient_descent(X, Y, theta, learning_rate, iterations):
    m = len(Y)
    loss_history = []

    for _ in range(iterations):
        prediction = X * theta
        gradient = (1/m) * np.sum(X * (prediction - Y))
        theta = theta - learning_rate * gradient
        loss = (1/(2*m)) * np.sum((prediction - Y) ** 2)
        loss_history.append(loss)

    return theta, loss_history

# Problem 1: Individual regressions
learning_rates = [0.1, 0.05, 0.01]
iterations = 1000

# Store results
models = {}
losses = {}
```



```python

for lr in learning_rates:
    # X1 regression
    theta1, loss1 = gradient_descent(X1, Y, 0, lr, iterations)
    models[f'x1_lr{lr}'] = theta1
    losses[f'x1_lr{lr}'] = loss1

    # X2 regression
    theta2, loss2 = gradient_descent(X2, Y, 0, lr, iterations)
    models[f'x2_lr{lr}'] = theta2
    losses[f'x2_lr{lr}'] = loss2

    # X3 regression
    theta3, loss3 = gradient_descent(X3, Y, 0, lr, iterations)
    models[f'x3_lr{lr}'] = theta3
    losses[f'x3_lr{lr}'] = loss3
```

Results

The best linear model found with learning rate 0.1 is:

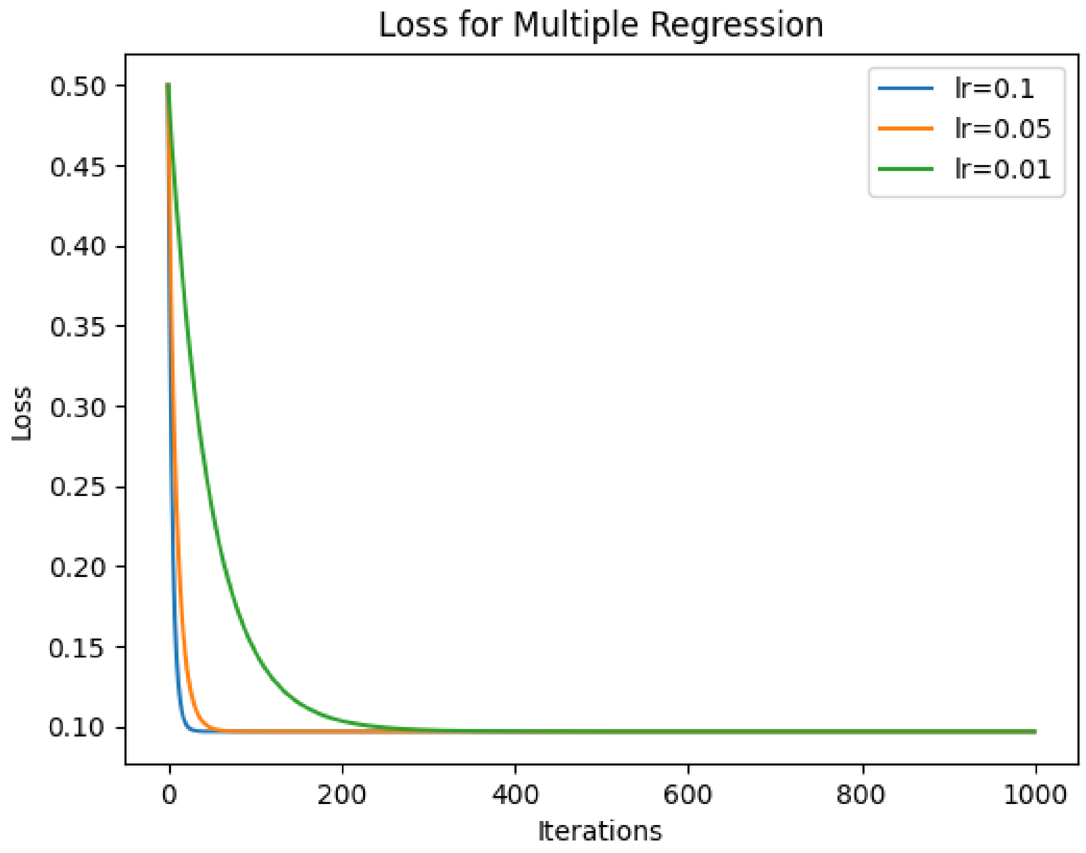
y = -0.8465x1 + 0.2250x2 - 0.1113x3 - 0.0000

Predictions for new values:

- (1, 1, 1): -0.7328
- (2, 0, 4): -2.1383
- (3, 2, 1): -2.2008

Plots

The figure below shows the loss over iterations for multiple regression across all learning rates:



Problem 2 Questions

1. Final Linear Model Found:

The best model, derived at learning rate 0.1 with theta initialized to zero, is:

y = -0.8465x1 + 0.2250x2 - 0.1113x3 - 0.0000

This model minimizes the loss when all three explanatory variables (X1, X2, X3) are combined.

2. Plot Loss Over the Iteration:

Completed earlier in report.

3. Impact of Different Learning Rates on Final Loss and Number of Training Iterations:

At learning rate 0.1, the loss dropped rapidly achieving the lowest final loss, which is why it was selected as the best model. At learning rate 0.05, the loss was slightly slower than 0.1. At learning rate 0.01 it took the longest due to smaller steps. Higher learning rates (0.1) reduce the number of iterations needed for convergence and, in this case, achieved a slightly better final loss, as lr=0.1 was chosen. Lower rates (0.01) increase the iteration count but still converge within 1000 iterations. Normalization ensured stability across all rates.

4. Predict the Value of Y for New (x1, x2, x3) Values:

Using the best model (y = -0.8465x1 + 0.2250x2 - 0.1113x3 - 0.0000), the predictions are:

For (1, 1, 1): -0.7328

For (2, 0, 4): -2.1383

For (3, 2, 1): -2.2008

These values were computed by normalizing the input data consistent with the training process and applying the model coefficients.