Homework 1

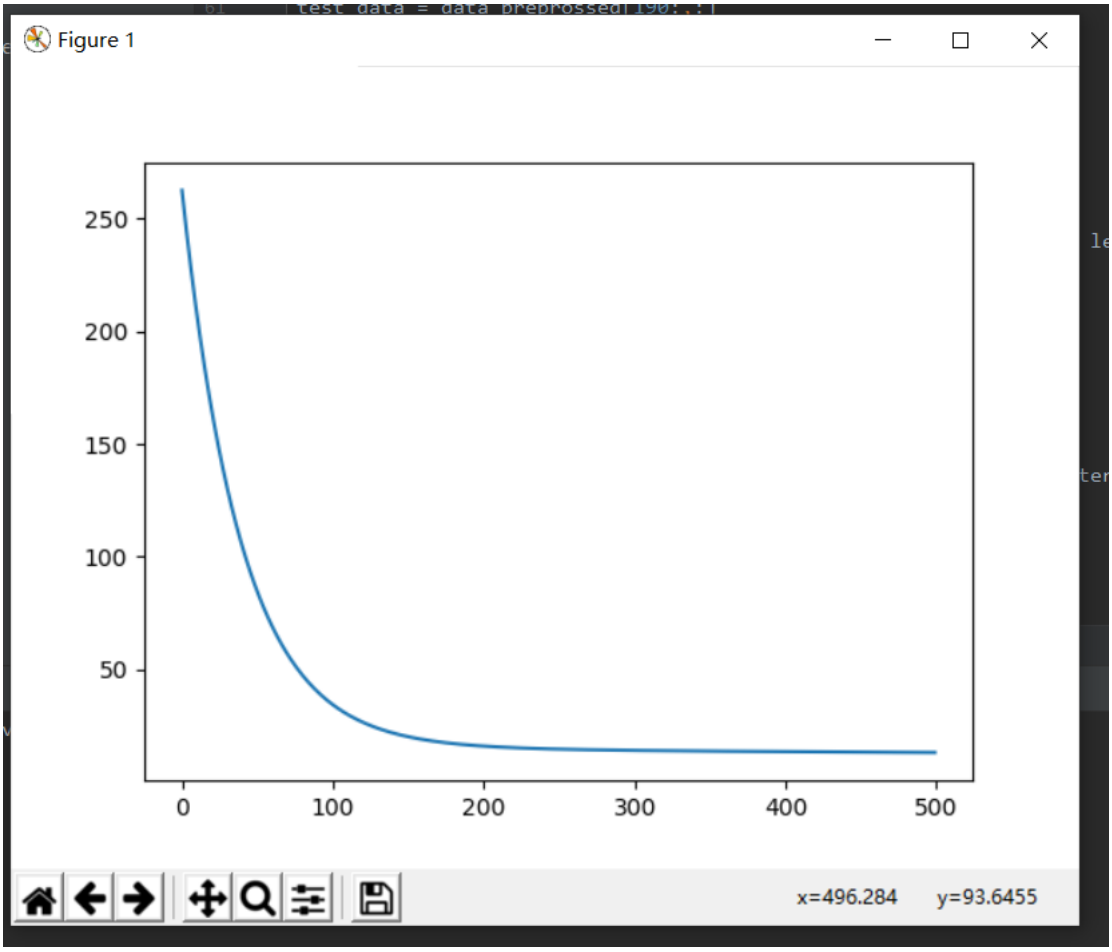
Z5048267 TAO Lipeng

Q1

Theta 0 is 10.11283406777275

Theta 1 is 8.271831294479673

Q2



Q3

3.640345489368778

Q4

3.9085603448797355

Q5

4.20042579511254

Q6

5.427909854899054

Q7

Rank 1, 3.9085603448797355

2, 4.20042579511254

3, 5.427909854899054

Therefore, TV model is best

```
import numpy as np
import csv
import matplotlib.pyplot as plt
import time
import math

def csv_reader(file_name):
        data = []
        with open(file_name) as file:
                csv_file = csv.reader(file)
                for row in csv_file:
                        data.append(row)
        return np.array(data[1:]).astype(float)

def pre_processing(data, axis_num):
        if axis_num != 'ALL':
                max_num = data[:,axis_num].max()
                min_num = data[:,axis_num].min()
                for i in range(len(data[:,axis_num])):
                        data[i,axis_num] = (data[i,axis_num] - min_num)/(max_num - min_num)
        else:
```

```
        for axis_num in range(1,4):
            max_num = data[:,axis_num].max()
            min_num = data[:,axis_num].min()
            for i in range(len(data[:,axis_num])):
                data[i,axis_num] = (data[i,axis_num] - min_num)/(max_num -
min_num)
    return data

def train(train_data,axis,theta_0,theta_1,learn_rate,max_iter):
    train_x = train_data[:,axis]
    train_y = train_data[:,4]
    loss_record = []
    for iter in range(max_iter):
        y_pred = theta_0 * 1 + theta_1 * train_x
        error = train_y - y_pred
        loss = np.mean(error*error)
        loss_record.append(loss)
        delta_theta_0 = np.mean(error*1)
        delta_theta_1 = np.mean(error*train_x)
        theta_0 = theta_0 + learn_rate * delta_theta_0
        theta_1 = theta_1 + learn_rate * delta_theta_1
    return theta_0,theta_1,loss_record

def evaluation(train_data,axis,theta_0,theta_1):
    train_x = train_data[:,axis]
    train_y = train_data[:,4]
    y_pred = theta_0 * 1 + theta_1 * train_x
    rmse = math.sqrt(np.mean((train_y - y_pred)**2))
    return rmse

def evaluation_T(test_data,axis,theta_0,theta_1):
    test_x = test_data[:,axis]
    test_y = test_data[:,4]
    y_pred = theta_0 * 1 + theta_1 * test_x
    rmse1 = math.sqrt(np.mean((test_y - y_pred)**2))
    return rmse1

data_raw = csv_reader('Advertising.csv')
data_preprossed = pre_processing(data_raw,'ALL')
train_data = data_preprossed[:190,:]
test_data = data_preprossed[190:,:]

theta_0 = -1
theta_1 = -0.5
```

```
learn_rate = 0.01
axis = 1
max_iter = 500
theta = train(train_data, axis,theta_0, theta_1, learn_rate,max_iter)
print(theta)
theta_0, theta_1, loss = train(train_data, axis,theta_0, theta_1, learn_rate,max_iter)
plt.plot(loss)
plt.show()
rmse = evaluation(train_data, axis,theta_0, theta_1)
print(rmse)
rmse1 = evaluation_T(test_data,axis,theta_0,theta_1)
print(rmse1)


theta_0 = -1
theta_1 = -0.5
learn_rate = 0.01
axis = 2
theta_0, theta_1, loss = train(train_data, axis,theta_0, theta_1, learn_rate,max_iter)
rmse1 = evaluation_T(test_data,axis,theta_0,theta_1)
print(rmse1)

theta_0 = -1
theta_1 = -0.5
learn_rate = 0.01
axis = 3
theta_0, theta_1, loss = train(train_data, axis,theta_0, theta_1, learn_rate,max_iter)
rmse1 = evaluation_T(test_data,axis,theta_0,theta_1)
print(rmse1)
```