Programming Assignment

For this assignment, you are asked to write a computer program to implement linear regression with gradient descent.

Implementation language: You can use either Matlab or Python to write the program.

Documentation requirement: Your program should be well-documented with comment statements to enhance the readability of the program. You may want to include a README file in your submission, in which you would include your NAME, specify the data files needed to run your code, and the main function(s) to call for part (A) and part (B) below.

Your programming task:

A. Write a program for linear regression (with one input variable) by gradient descent. For this part, your program reads an input file KCSmall2.csv (which has NO column heading) which has two columns (x, y). The program trains on the input data with gradient descent to learn the parameter vector $\theta = (\theta_0, \theta_1)^T$ such that for each data pair (x, y) in the input file, the predicted value $\hat{y} = \theta_0 + \theta_1 * x$ would approximate well the target y value. Namely the program outputs an estimated optimal value for $\theta$ minimizing the loss (error) function
$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(y^i - h_\theta(x^i))^2$ with $h_\theta(x^i) = \theta^T * x^i$ (the dot product of $\theta$ with $x^i$)

The loss function $J(\theta)$ is called "Mean Squared Error" (MSE), $m$ $(=100 \text{ here})$ is the number of training examples and $(x^i, y^i)$ is the i-th training data point pair, with $x^i$ being expanded by adding the zero-th feature $x_0^i = 1$ .

Note that with gradient descent, the update quantity for the parameter vector $\theta_j$ ($for\ j$ = 0,1) is given by the formula:

$$\triangle \theta_j = \frac{\eta}{m}\sum_{i=1}^{m}(y^i - h_\theta(x^i)) * x_j^i$$

Desired output of the program:

(1) First, plot the data points in a 2-d diagram (scatter plot), with horizontal axis for the input variable x, and vertical axis corresponding to the output variable y. The x values are house living areas in 1000 square feet (namely, x=2.5 represents 2500 square feet of living area for a house), and the y values represent house prices in 10,000 dollars. Each data point should be plotted on the diagram using the marker "X" (a cross).

(2) Print the loss function value $J(\theta)$ for $\theta$ = (0,0) and for $\theta$ = (−1 , 20). The correct output values are about 1806.551 and 330.099 respectively. This would allow you to check if your loss function is implemented correctly.

(3) Plot the loss function curve for 4 trials (so the program generates 4 figures for this part – do NOT plot them together in one figure! For each plot figure, the horizontal axis corresponds to the number of iterations (epochs) and the vertical axis corresponds to the values for the loss function $J(\theta)$ . For each trial, the program starts fresh with $\theta$ = (0,0) as the initial value, and run 15 iterations (epochs). The learning rate $\eta$ value for each trial is different, and is taken from the following list: (0.01, 0.1, 0.2, 0.4). Your figure should be titled to indicate the learning rate value for the specific trial. Note that when executing your program once, it should produce all these 4 figures through a loop over the list of (0.01, 0.1, 0.2, 0.4). It is NOT acceptable that your submitted code would just run for ONE trial with one $\eta$ value! Submitting other plotted figures produced by (you) running your program multiple times with different $\eta$ values would NOT help here – the instructor would execute your program and see if it produces plots of all 4 figures in one run.

(4) For each trial, the program should print out the final $\theta$ vector learned, along with the associated loss value $J(\theta)$ for the final $\theta$. The program should also print out the predicted y values for the x=3.5 (3500 square feet living area) and x = 7.0, using the final learned $\theta$ vector for each trial. Again, running your program once, it should print out the desired final $\theta$, $J(\theta)$ and the predicted y values for all 4 trials.

B. Here you should extend the program from part A to handle multivariate linear regression in which there are more than one input variables. For this part, your program reads an input file (KCSmall_NS2.csv) which has 4 columns (n_bed, liv_area, lot_area, h_price). The values of the variables have NOT been scaled, as a result the range of the variables differ significantly. So after reading the data, you would need to first standardize the input variables (but NOT the output variable h_price). Standardization means to replace each value $v$ for feature A by ( $\frac{v-\bar{A}}{\sigma_A}$ ), namely, first subtract mean value of A from $v$, and then divide the result by sample standard deviation of A. After that, you should ADD the column of all 1's (corresponding to the dummy input variable to be multiplied by $\theta_0$) to the input variable matrix X. The loss function $J(\theta)$ is defined in the same way as in part A. The update formula $\triangle\ \theta_j$ is the same as well, except that j ranges from 0 to 3, as we now have 3 input variables. Please note that you do NOT perform standardization to the target variable y (h_price).

Your program would run gradient descent to learn an estimated optimal value for the parameter vector $\theta$ minimizing the loss function $J(\theta)$ .

Desired output of the program:

(1) Print out the first 5 rows of the raw input data including the target variable (to make sure the data loading works correctly), also check m=100 (the number of input rows equals 100).

(2) Print out the first 5 rows of standardized input data X (after adding the dummy 0th column) and (non-standardized) target variable y. This is to make sure your inputs to gradient descent are correct.

(3) Print out the cost $J(\theta)$ value for $\theta$ = (0, 0, 0, 0).

(4) Perform 5 trials, with each trial starting fresh from $\theta$ = (0, 0, 0, 0) as the initial parameter value, and running gradient descent for n = 50 iterations. Each trial uses a different learning rate value taken from the list (0.01, 0.1, 0.5, 1.0, 1.5) . Plot the loss function curve for each trial – in 5 figures. Similar to part A, you should set the title for each figure indicating the learning rate $\eta$ for that particular trial. Same as in part A, executing your program once should generate all 5 plotted figures.

(5) For each of the 5 trials, print out the final $\theta$ value, the associated loss value $J(\theta)$, and the predicted y value for the input n_bed = 3, liv_area=2000, lot_area=8550 using the final $\theta$ value learned after 50 iterations. Note that you would need to first standardize these 3 input values using the mean and standard deviation values obtained from the training data, and add the 0th column value 1. Then the final learned $\theta$ is applied to the transformed input to produce the prediction. Same as in part A, running your program once, it should print out the desired final $\theta$, $J(\theta)$ and the predicted y value for all 5 trials.