Grayson Bass
ID Number: 801074960
Homework 1
Github: https://github.com/gbass2/IntroToML

Problem 1

Part a:

The gradient descent and cost functions for problem 1 are the same as the previous homework. For the first problem, the input variables that were used were area, number of bedrooms number of bathrooms, number of stories, and the number of parking spaces. The output variable was price. Data set was split up into 70% training set and 30% validation set using the sklearn library. The only modification that was done to the gradient descent function was to include a cost history for the validation set. This allowed me to plot cost vs. iterations for both training and validation. Figure 1 shows a graph of cost vs. iterations. The figures can be found in the source code pdf file.

The learning rate to make the loss converge was 0.0000000001 and the number of iterations used was 1000. A very small learning rate was required because the parameter "area" is large compared to the other parameters and therefore the parameters were not explaining the output. Another approach would have been to remove area as an input variable.

The best parameters (theta) are [0.17254066 817.36415965 0.54640289 0.25046484 0.36083704 0.13700813]

Theta_1 is the parameter associated with area and since area is significantly higher than the other input variable, it had a strong effect on the model.

Part B:

The approach is the same as part a with the only difference being the number of parameters used. All the parameters in the housing dataset were used except for the furnishing status. The binary parameter (yes/no) were converted to either 0 or 1 before performing the gradient descent. The learning rate stayed the same at 0.0000000001 and the number of iterations was the same at 1000. Figure 2 shows the cost vs. the number of iterations for the training and validation when using the binary parameters.

The best parameters (theta) are [0.15156855 758.44655408 0.47504058 0.21538025 0.30823608 0.13517499 0.03630238 0.06591472 0.01009858 0.06618256
0.1200329 0.04394649]

The theta values decreased compared to the theta values in part a. Theta_1 is still significantly higher than the other input parameters, but it has decreased. The training cost has decreased ever so slightly from 1.57604747e+12 in part a to 1.57604736e+12 in part b. The area parameter is still having a strong effect on the model and feature scaling is needed.

Problem 2
Part a:

The gradient descent and cost functions stayed the same as problem 1. The change occurs in the input variables and output variables. For problem 2, both normalization and standardization were used. After the data is split into training and validations, the sklearn MinMaxScaler and StandardScaler were used to scale the data. Then the gradient descent is calculated for both with a learning rate of .01 and the number of iterations were 250 for normalization and 200 for standardization. Figure 3 shows the plot of the cost vs. the iterations for normalization Figure 4 shows the plot of the cost vs. the iterations for Stan dardization:

Compared to problem 1 part a, both normalization and standardization converged a lot quicker with a larger alpha. When setting the number of iterations to 10000, the validation loss for normalization starts to increase at around 2000 iterations showing that the model is starting to overfit. Whereas the validation loss for standardization remains relatively constant. The curve for the validation loss in standardization is better follows the curve of the training loss compared to the normalized model. Nomalization provided a lower cost than the standardized model. The loss when using normalization follows a similar curve to the non-scaled model in problem 1 part a except the non-scaled model does not start to overfit like the normalized model.

Theta for normalization = [0.18999258 0.07446712 0.09050375 0.04504933 0.08173306 0.0739338]
Theta for standardization = [0. 0.36176458 0.1171842 0.2921215 0.20710648 0.19452094]

Part b:

Problem 2 part b was completed by combining part a and problem 1 part b using all the input variables except furnishing status. The learning rate is set to .01 and the number of iterations for normalization is 150 and 200 for standardization. Figure 5 shows the plot of the cost vs. the iterations for normalization. Figure 6 shows the plot of the cost vs. the iterations for Standardization.

Compared to problem 1 part b, both normalization and standardization converged a lot quicker with a larger alpha. When setting the number of iterations to 10000, the validation loss for normalization starts to increase at 2000 iterations showing that the model is starting to overfit. Whereas the validation loss for standardization remains relatively constant. The curve for the validation loss in standardization is better follows the curve of the training loss compared to the normalized model. Normalization provided a lower cost than the standardized model. The loss when using normalization follows a similar curve to the non-scaled model in problem 1 part b except the non-scaled model does not start to overfit like the normalized model.

Theta for normalization = [0.10199588 0.03958284 0.0502271 0.0261717 0.04753288 0.09485726
0.03096009 0.04488565 0.0103418 0.06258119 0.04103593 0.03786835]

Theta for standardization = [0. 0.28136212 0.0922294 0.27227317 0.17908698 0.10572215
0.08817514 0.11084128 0.07192317 0.18832824 0.15747882 0.12163229]

Problem 3
Part a:

For problem 3, parameter penalty was added. The cost function was modified for the training set but not the validation set. The parameters (theta) were squared using numpy.square excluding theta_0. The result of this was then added to the square error and multiplied by $\frac{1}{2m}$. The rest of the code is the same as problem 2 part a. The learning rate of .01 and the number of iterations were 250 for normalization and 200 for standardization. Figure 7 shows the plot of the cost vs. the iterations for normalization using regularization Figure 8 shows the plot of the cost vs. the iterations for standardization using regularization.

Theta for normalization = [0.18775205 0.08421515 0.09442677 0.05463201 0.09050374 0.0830125]
Theta for standardization = 0. 0.37704584 0.10623171 0.30071024 0.21275141 0.19485301]

Part b

Part b was completed the using the cost functions from part a and the code from problem 2 part b. Figure 9 shows the cost vs the iterations for normalization using regularization. Figure 10 shows the plot of the cost vs. the iterations for standardization using regularization:

Theta for normalization = [0.10199588 0.03958284 0.0502271 0.0261717 0.04753288 0.09485726
0.03096009 0.04488565 0.0103418 0.06258119 0.04103593 0.03786835]

Theta for standardization = 0. 0.28136212 0.0922294 0.27227317 0.17908698 0.10572215
0.08817514 0.11084128 0.07192317 0.18832824 0.15747882 0.12163229]