CS 6375

ASSIGNMENT 1

Names of students in your group:

1. Rusheel Koushik Gollakota (Net ID – rxg220072)
2. Sai Naga Viswa Chaitanya Basava (Net ID – sxb220302)

Number of free late days used: \_0\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
Note: You are allowed a **total** of 4 free late days for the **entire semester**. You can use at most 2 for each assignment. After that, there will be a penalty of 10% for each late day.

Please list clearly all the sources/references that you have used in this assignment.

Data used for this assignment is hosted at this [link](https://github.com/chaitanya-basava/CS6375-004-Assignment-1-data/blob/main/abalone.data).

Below are the top experiment results of both parts of the assignment. Additional experiment results can be found within the log files. ManualModel.txt contains results for part 1 question. LinearModel.txt contains results for part 2 question.

1. Following are the top 5 experiment results of the model built without using libraries (part 1) -

lr: 0.1, threshold: 1e-07, max\_itrs: 10000  
train\_mse: 0.4756, test\_mse: 0.4606  
train\_adj\_r\_sqrd: 0.5361, test\_adj\_r\_sqrd: 0.4867  
----------------------------------------------------------------------------------------------------  
lr: 0.01, threshold: 1e-07, max\_itrs: 50000  
train\_mse: 0.4763, test\_mse: 0.4613  
train\_adj\_r\_sqrd: 0.5354, test\_adj\_r\_sqrd: 0.486  
----------------------------------------------------------------------------------------------------  
lr: 0.01, threshold: 1e-07, max\_itrs: 10000  
train\_mse: 0.4806, test\_mse: 0.4664  
train\_adj\_r\_sqrd: 0.5312, test\_adj\_r\_sqrd: 0.4803  
----------------------------------------------------------------------------------------------------  
lr: 0.01, threshold: 1e-06, max\_itrs: 10000  
train\_mse: 0.4832, test\_mse: 0.4695  
train\_adj\_r\_sqrd: 0.5286, test\_adj\_r\_sqrd: 0.4768  
----------------------------------------------------------------------------------------------------  
lr: 0.0001, threshold: 1e-08, max\_itrs: 500000  
train\_mse: 0.4853, test\_mse: 0.4717  
train\_adj\_r\_sqrd: 0.5266, test\_adj\_r\_sqrd: 0.4744

----------------------------------------------------------------------------------------------------

1. Following are the top 5 experiment results of the model built using libraries (part 2) -

lr: 0.0001, threshold: 1e-07, max\_itrs: 10000  
train\_mse: 0.4756, test\_mse: 0.4604  
train\_adj\_r\_sqrd: 0.5361, test\_adj\_r\_sqrd: 0.487  
--------------------------------------------------

lr: 0.0001, threshold: 1e-06, max\_itrs: 10000  
train\_mse: 0.4758, test\_mse: 0.4605  
train\_adj\_r\_sqrd: 0.5359, test\_adj\_r\_sqrd: 0.4869  
--------------------------------------------------  
lr: 0.001, threshold: 1e-06, max\_itrs: 10000  
train\_mse: 0.4764, test\_mse: 0.4606  
train\_adj\_r\_sqrd: 0.5353, test\_adj\_r\_sqrd: 0.4868  
--------------------------------------------------

lr: 0.001, threshold: 1e-05, max\_itrs: 10000  
train\_mse: 0.4766, test\_mse: 0.4608  
train\_adj\_r\_sqrd: 0.5351, test\_adj\_r\_sqrd: 0.4866  
--------------------------------------------------  
lr: 0.0001, threshold: 1e-05, max\_itrs: 10000  
train\_mse: 0.4779, test\_mse: 0.4628  
train\_adj\_r\_sqrd: 0.5338, test\_adj\_r\_sqrd: 0.4843  
--------------------------------------------------

The term referred above are as follows,

**lr:** learning rate

**threshold:** training iteration termination threshold

**max\_itrs:** number of iterations

**train\_mse:** mse (mean squared error) loss of training

**test\_mse:** mse (mean squared error) loss on test data

**train\_adj\_r\_sqrd:** adjusted r squared value on training data

**test\_adj\_r\_sqrd:** adjusted r squared value on test data

lr, threshold and max\_itrs are the hyperparameters that have been tuned during these experiments, using our own implementation of grid search.

**Ques:** Are you satisfied that you have found the best solution? Explain.

**Ans:** Yes, we are satisfied with the results of our best solution.

Based on the results from our experiments – A lower MSE indicates a better fit on the data, and a higher R-Squared value indicates that the model explains more variance in the data.

The experiment in part-1 with following hyper-parameter setting - lr: 0.1, threshold: 1e-07, max\_itrs: 10000 gave the best mse value and an adjusted r-squared value, which is comparable to the best model produced by the SGDRegressor implementation in scikit-learn library (part-2). The differences in performance metrics between the two sets of hyperparameters are relatively small, test set MSE of our implementation was 0.4606, while the test set MSE of SGDRegressor was 0.4604 (~0.002 difference). Plot (7) also indicates proper convergence of train and test losses over the iterations.

Plots generate during data exploration and the experiments.

1. Features correlation with the target variable.

A blue bar graph with white text

Description automatically generated

From this graph, we can observe that the features provided are very much in correlation with the target variable. This imply that the model would be able to efficiently leverage the relations within data, which in turn can help boost the prediction’s accuracy and interpretability. Hence, we choose to avoid doing any complex feature engineering steps and instead only make use of the existing correlations to build our models.

1. Top 5 important features based on the weights learnt by our implementation

A blue bar graph with black text

Description automatically generated

1. Plot of individual features again the corresponding reference line of the best model from our implementation.

A group of graphs with red lines

Description automatically generated with medium confidence

1. Top 5 important features based on the weights learnt by scikit-learn’s implementation

A blue and white bar graph

Description automatically generated

1. Plot of individual features again the corresponding reference line of the best model from scikit-learn’s implementation.

A graph of data with red line

Description automatically generated with medium confidence

1. Training convergence graph comparison between scikit-learn and our implementation.

A graph with a red line

Description automatically generated

1. Train vs Test cost function of our implementation’s best model

A graph with a red line

Description automatically generated

1. Train vs Test cost function of scikit-learn implementation’s best model

A graph of a graph

Description automatically generated

**Ques:** Are you satisfied that the package has found the best solution. How can you check. Explain.

**Ans:** We arrived at the best model after running extensive experiments with various combinations, also both the MSE and adjusted R-Squared values of train and test sets are very much close and there is no evidence of underfitting or overfitting. The train vs test loss curve (8) also indicates that the losses have converged. So, we are satisfied with the model produced by the package.

Also, when comparing with our model, from plots (2), (3), (4), (5) and (6) we have a clear observation that both our implementation and the scikit learn implementation have learned almost the same weights. This is especially highlighted in plot (2) and (4), where the feature importance order according to the model coefficients is almost similar. The loss convergence as observed in plots (6) and (7) also indicate how both the implementations had a similar learning curve, except the scikit-learn version (~1600 iterations) converged faster compared to ours (~4500 iterations).

Hence, we are satisfied with both our models due to the similarity in their performances.