ENPM808A - Introduction to Machine Learning
Learning Model Evaluation and Selection
Shon Cortes

Introduction:

The goal was to train multiple machine learning models using provided data and evaluate their performance. The best performing model was then selected as a deliverable. A data preprocessing procedure was first evaluated before training any models. The data was divided into four sections. The first 1080 columns consisted of laser range data covering 270 degree field of view. The next three sections are grouped into four columns; each section representing the final goal, local goal and the robots current position information respectively. The four columns in each section consist of the "x" and "y" positions as well as the orientation, in quaternion (qk, qr), for each of the sections. The final two columns are the labels which consist of the linear and angular velocities the robot will use to drive. The LIDAR data covered 1080 columns that spanned a 270 degree field of view. To condense this information, the LIDAR data was split into four sections each covering 67.5 degrees of information. The four sections were averaged then divided by the maximum to scale the data between zero and one and regularize the data. Further simplification of the feature data set was done by grouping the local goal and robot position information. These were grouped by finding the difference between the positions and orientations. Finding these deltas allowed for reducing these features from eight to four (Δx, Δy, Δqk, Δqr). The total dataset provided was approximately 25 gigabytes of data. This was difficult for my machine to handle so only a small subset of the training and test data was used. Models were then trained and compared after simplifying the feature set and regularizing the data.

Linear Regression:

The first model tested was a linear regression model. This method attempts to minimize the distance between the data and the linear separator function. Calculation of the pseudo inverse helps do this minimization and we can find our weight vector form here. Linear regression gives us the smallest in-sample error possible in one step. For this case the weight vector returned is a 9x2 matrix; one row for each of the features and one column for each of the outputs (linear and angular velocity). Our weights, accuracy score, and in/out of sample errors can be seen below:

Linear Regression Weights: [[0.00000000e+00 0.00000000e+00]
 [-6.41612957e-01 -1.29485114e-01]
 [5.79072690e-02 -1.35470964e-01]
 [5.31883990e-01 1.33194893e-01]
 [-7.12046332e-01 6.69709921e-02]
 [1.44051771e-02 1.80931171e-02]
 [-2.64140785e-02 1.66382072e-03]
 [-1.13852007e-01 1.45684322e-02]
 [-6.27623154e-03 -4.89986573e-04]]

Linear Regression Out of Sample Error: 0.12049004638645117

Linear Regression In Sample Error: 0.07156458766910313

Linear Regression Out of Sample Accuracy Score: -0.15649031778890493

Linear Regression In Sample Accuracy Score: 0.1812224925740104

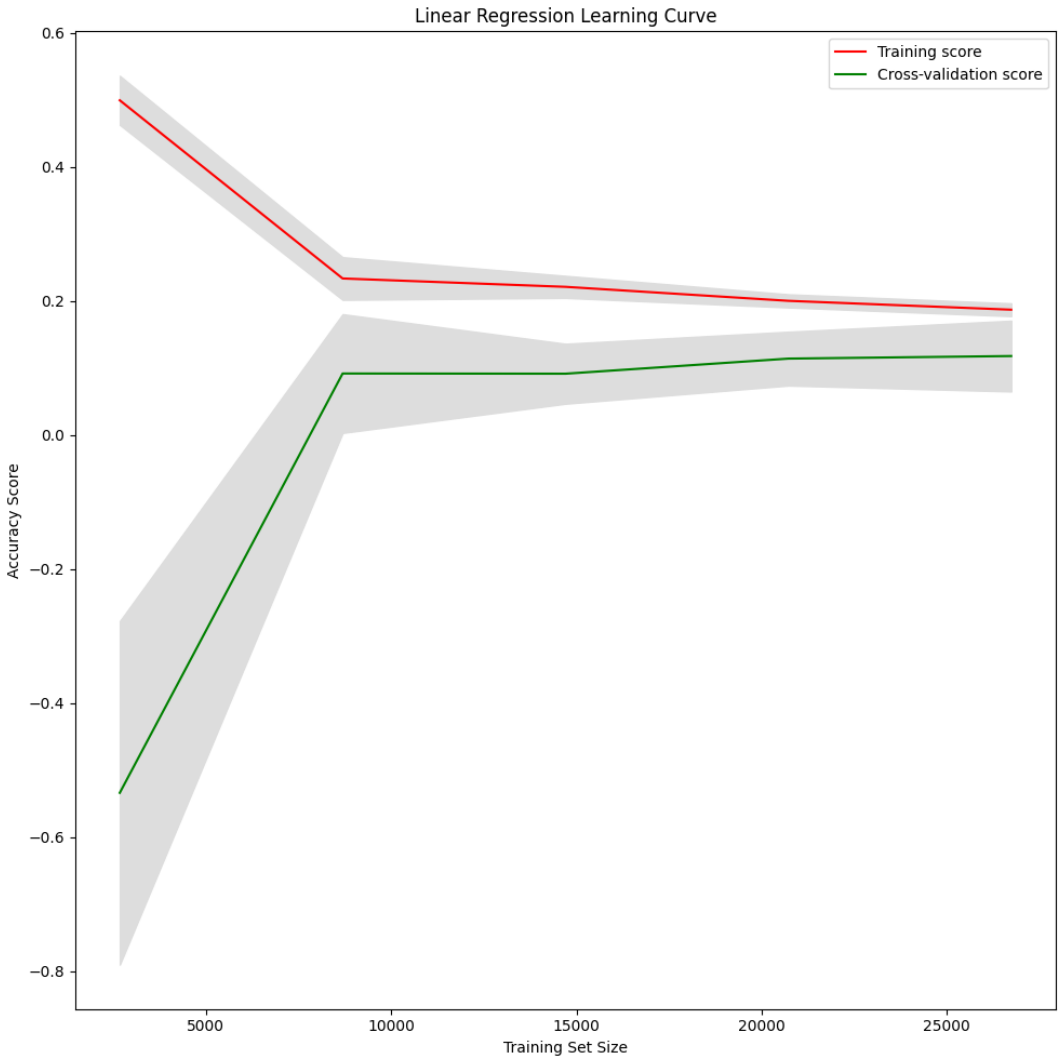

Figure 1: Linear Regression Learning Curve

The in-sample error is found by comparing the predicted labels with the expected labels from the training data. All errors were found by calculating the mean squared error. Our linear regression model had an in-sample error of 0.072. In other words this model was able to predict with a 92.8% success rate when using the training data as input. The success rate decreased to 87.9% when using the test data (out of sample error 0.120). The accuracy score given from the sklearn python library is the coefficient of determination of the prediction. This is a measure of how accurate the predictions made by the model are when compared to the actual data labels. The score normally ranges between zero and one where the best possible score achievable is a one. Negative values are possible and indicate the average of the data provided provides a better fit than the trained model. In other words the linear separator function obtained performs worse than a function defined by the average of the provided data. For this model we can see

the our of sample accuracy score is negative and the in sample score is closer to zero than one. This indicates the model did not perform well when compared to the average of the data according to this metric. Figure 1 shows the learning and validation curves converging which indicates that adding more training data may have a marginal effect on the model performance. The gray areas around the lines indicate the standard deviation.

Decision Tree Regression:

In decision tree regression we use our features to go through a series of “branches” that lead to a correct label. The model goes through a series of decisions, each decision leading to a branch, and ultimately attempts to correctly label the data. The max depth parameter was adjusted and compared for our model. The max depth parameter indicates the depth of the decision tree. The true maximum depth is N - 1 where N is the number of data points. By setting a user defined max depth we can limit the number of branches the model will split on. By default the number of nodes will expand to the true maximum depth. The score and in/out of sample errors for all three models can be seen below:

DTR Out of Sample Error (max depth= 1): 0.10358445541261233
DTR In Sample Error (max depth= 1): 0.09541276098000764
DTR Out of Sample Accuracy Score (max depth= 1): -0.029599475999058766
DTR In Sample Accuracy Score (max depth= 1): 0.0

DTR Out of Sample Error (max depth= 10): 0.16056006687573898
DTR In Sample Error (max depth= 10): 0.012906480492195158
DTR Out of Sample Accuracy Score (max depth= 10): -0.688004772502052
DTR In Sample Accuracy Score (max depth= 10): 0.8238384103059477

DTR Out of Sample Error (max depth= default): 0.1826566532802652
DTR In Sample Error (max depth= default): 1.5336673387216882e-07
DTR Out of Sample Accuracy Score (max depth= default): -0.9135885418106986
DTR In Sample Accuracy Score (max depth= default): 0.9999969728718665

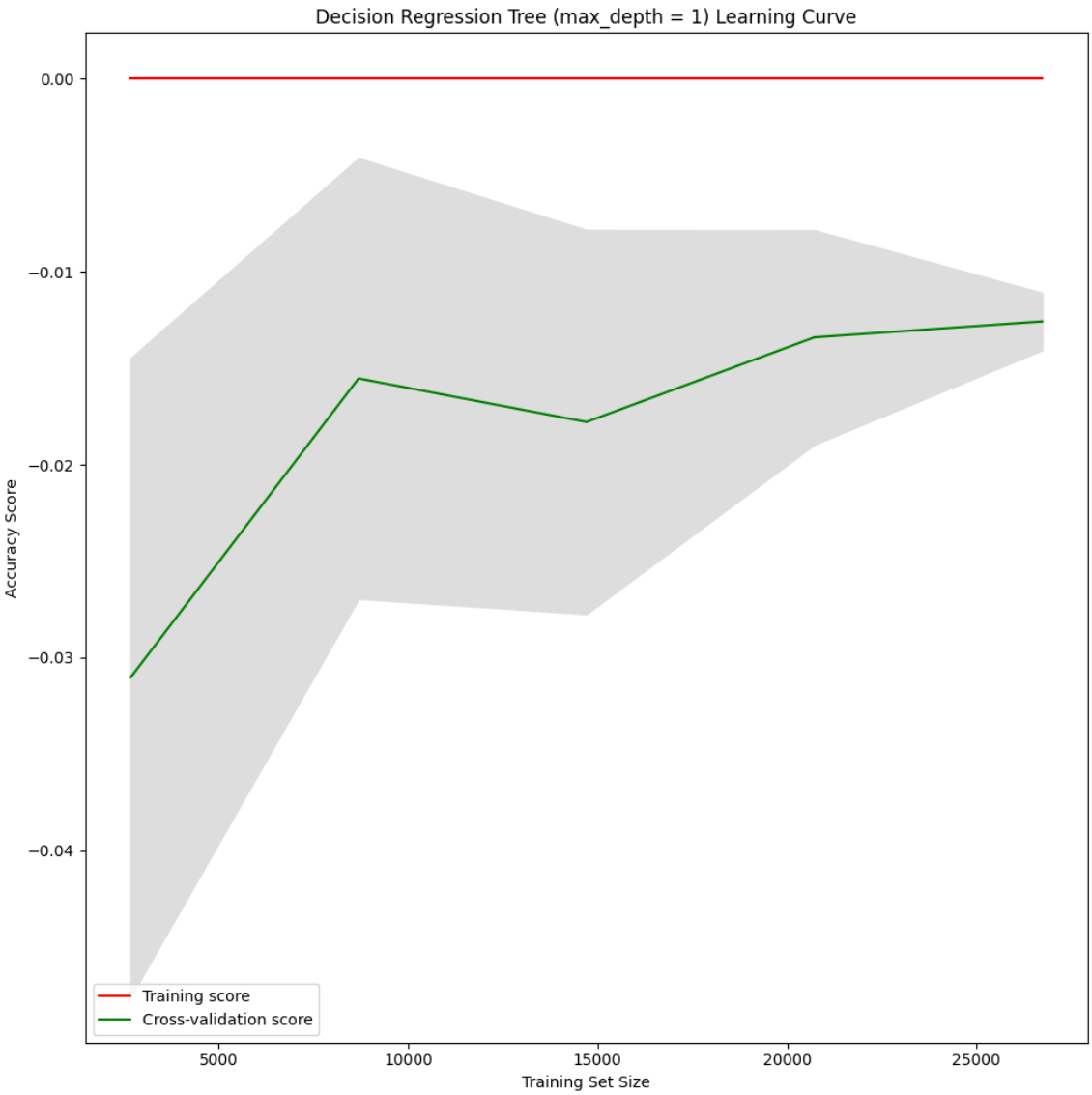

Figure 2: Decision Regression Tree (max depth = 1)

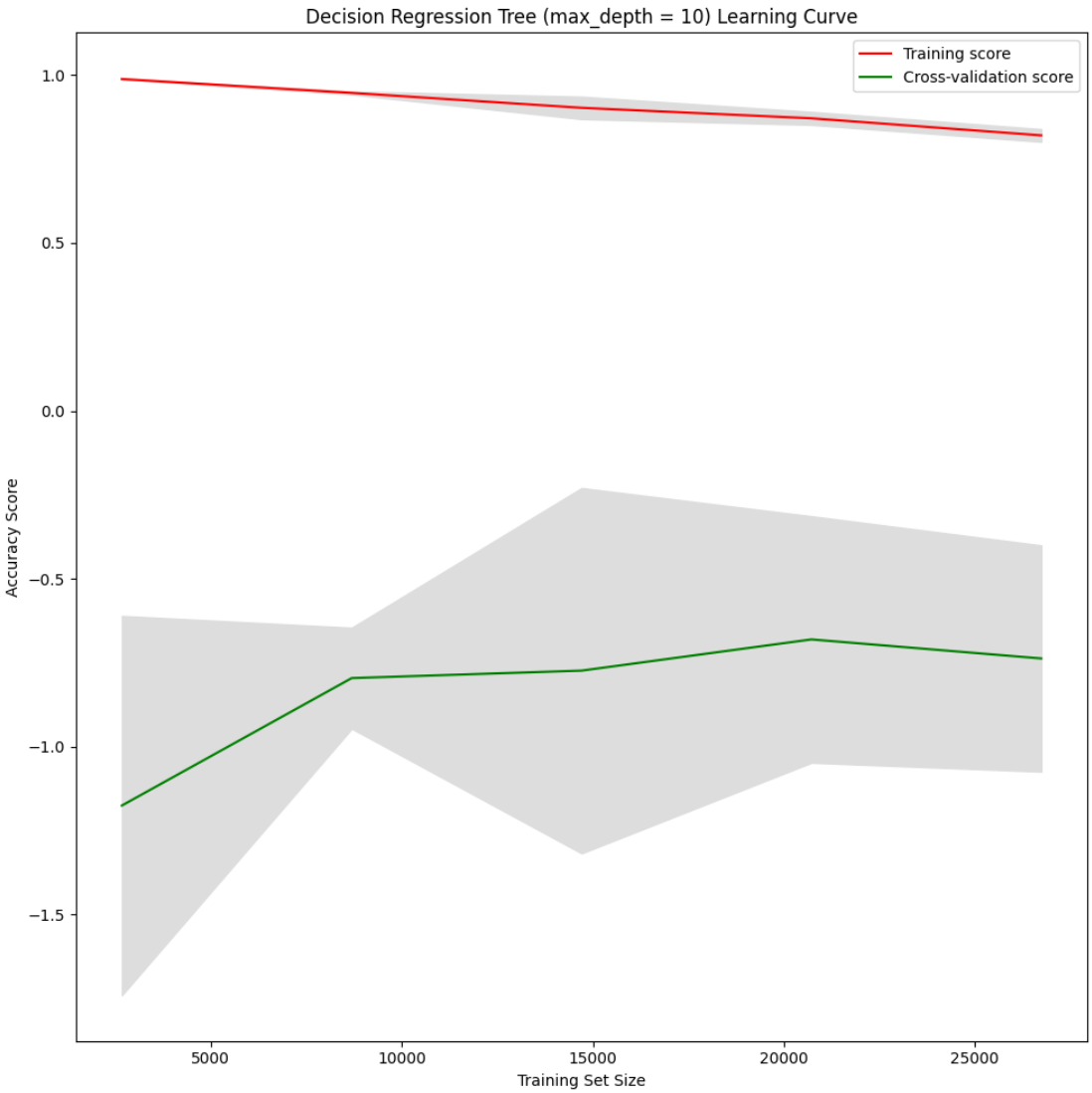

Figure 3: Decision Regression Tree (max depth = 10)

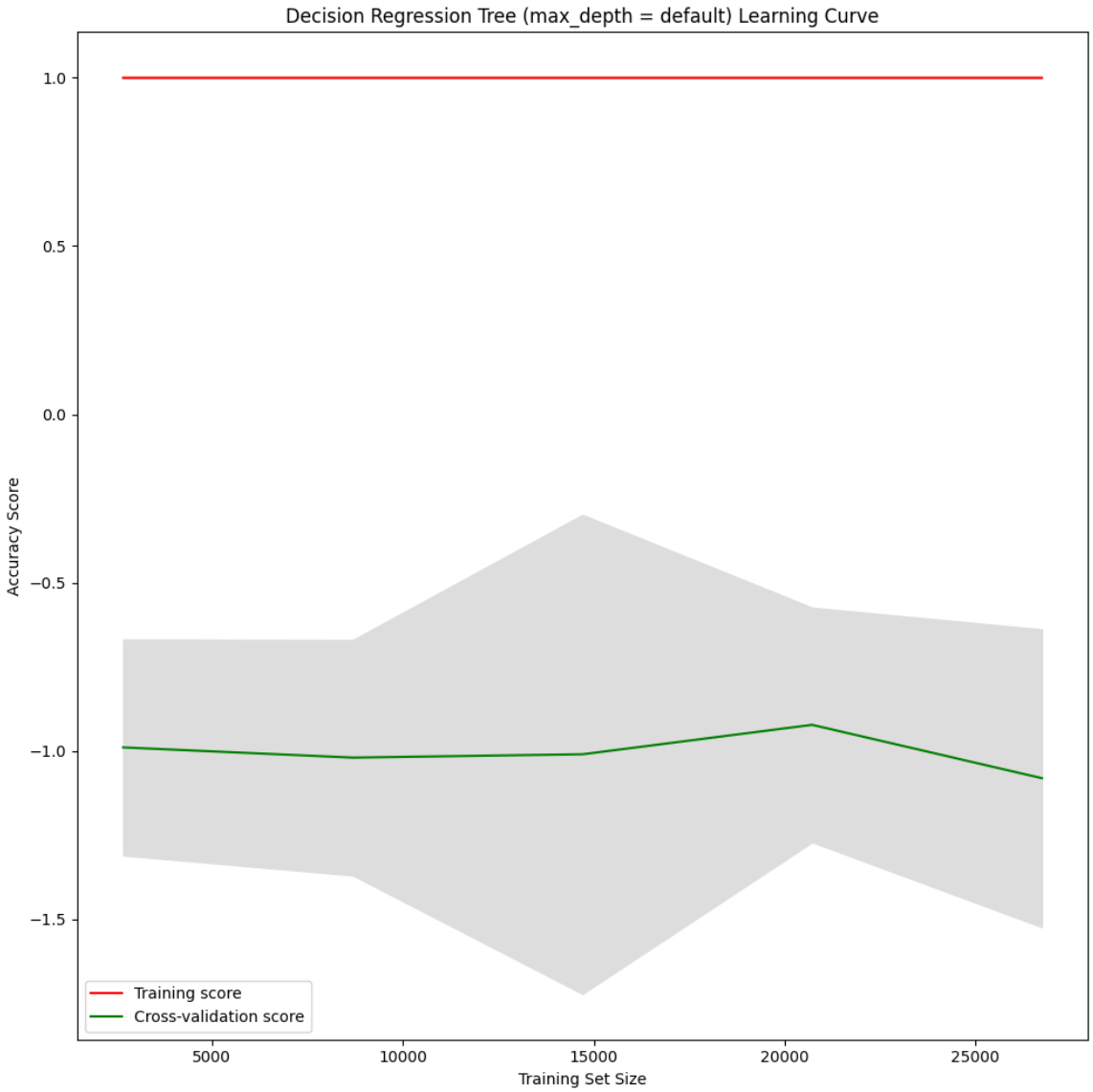

Figure 4: Decision Regression Tree (max depth = default)

The max depth did affect the mean square errors and the models generalizability. A model was fit using a max depth of 1, 10, and the default. Deeper decision trees allowed for better in sample error but worse out of sample error. The accuracy scores also got better with deeper decision trees. The default max depth did achieve a model with 0.999 which is close to the best achievable score; however when picking the most generalizable model, a priority was placed on picking the best out of sample error performer. When using a max depth of one, the out of sample error was 0.103 meaning the model has a 89.7% success rate. Figures 2 - 4 show that the learning and validation curves are not converging and the models would greatly benefit from training on more data.

Sequential Neural Network Using Tensorflow and Keras:

The sequential model uses the recurrent neural network algorithm for learning. It is optimal for sequential data to be used as inputs and outputs. The model weights were initialized using the HeUniform initializer which selects initial weights that are uniformly distributed between a set limit determined by the size of the input data tensor. A sigmoid activation function was used to map the outputs in each layer of the neural network. The sigmoid activation function outputs values between zero and one. The model was made with five layers and was compiled using the ADAM optimizer. ADAM optimizes gradient descent and handles noisy data well.

The RNN models' in and out of sample errors can be seen below along with a learning history:

Sequential Model Out of Sample Error(MSE): 0.1197641211242923
Sequential Model In Sample Error(MSE): 0.06441059488651525

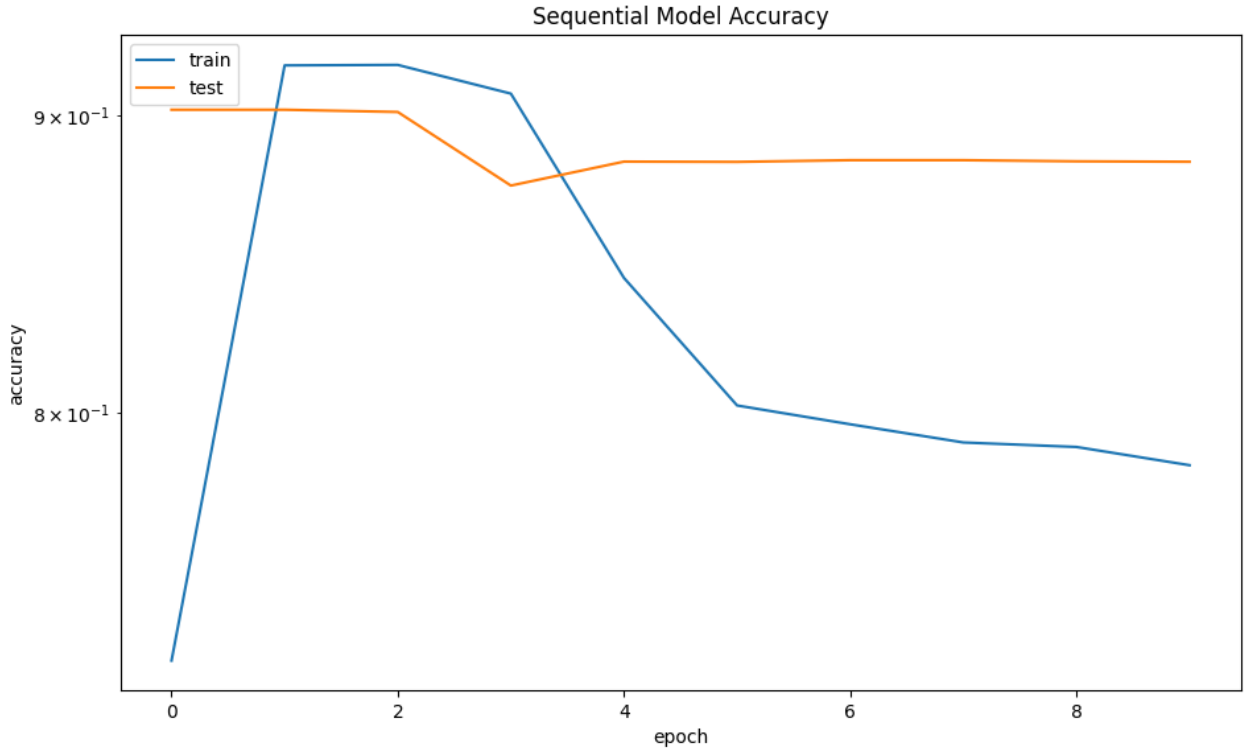

Figure 5: Sequential Model Accuracy

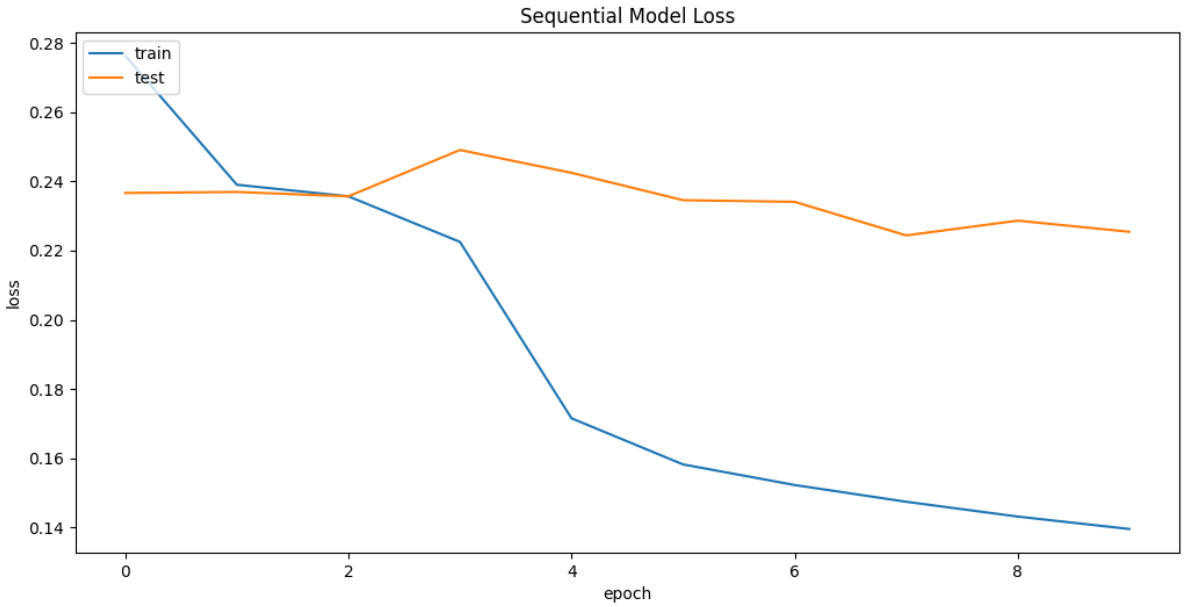

Figure 6: Sequential Model Loss

The trained model had a worse accuracy the more data it was trained on, as seen in Figure 5, however its out of sample performance was not greatly affected by the amount of data it was trained on. Figure 6 shows the model did experience less loss as more data was used for training. The out of sample error was 0.119 meaning the model had an 88.1% accuracy.

Model Comparison and Selection:

We strive for a model to be generalizable in optimal model selection. The model which generalizes best using out of sample data was chosen to be the best performing. Decision regression tree training using a maximum depth of one returned a model that successfully predicted the correct output 89.7% of the time. The learning and validation curves for this model indicate that more training data would improve the model performance. For future improvement, the model would be trained on the full data set provided. With the full training data set being used, all models should be re-evaluated as their performances may change as well.