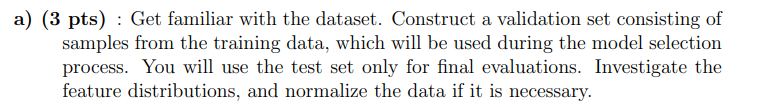
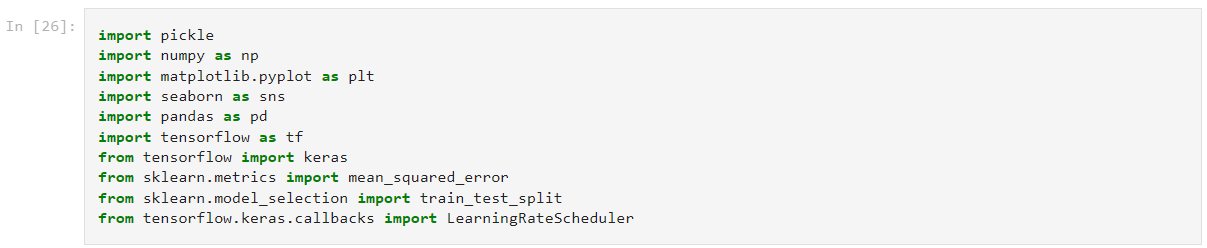
**Deep Learning KU (708.220) WS23**

**Assignment 2: Training Neural Networks**



Code:



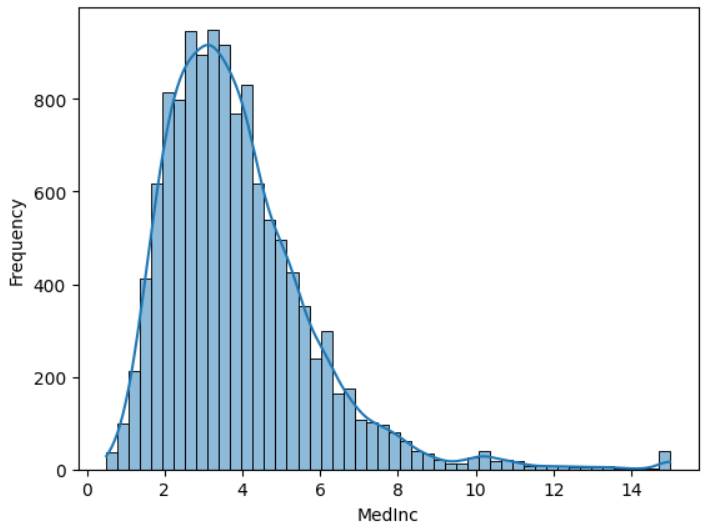


At task number a, it is needed to understand the structure of the data and getting familiar with data set. First ,program loads the dataset using the pickle module and extracts the training&test sets “x\_train”,”y\_train”,”x\_test”,”y\_test” and displays the dimensions of the set. After loading and extraction, the shape of the training data printed. The print statements show the values of the first example in the training set (x\_train[0]) and its corresponding label (y\_train[0]).

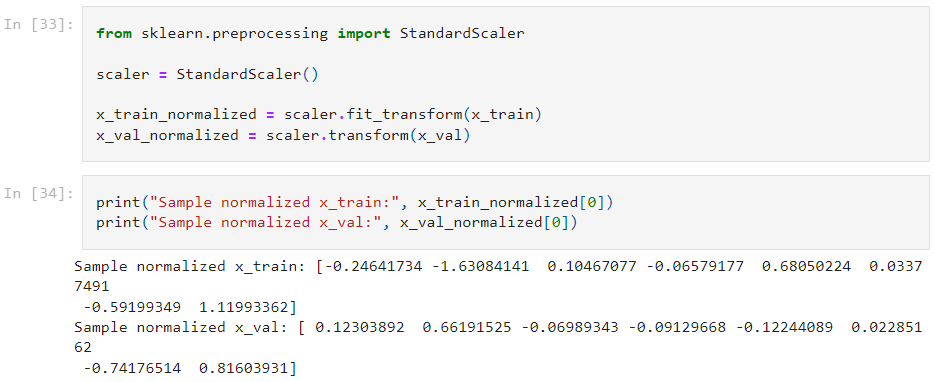
According to task, its essential to construct a validation set. To make this, the training data splitted into training and validation sets with the test size 0.2. Which determines the proportion of the dataset, in this case 0.2 refers to original dataset and remaining 0.8 refers to training . As a test, we defined random\_state as 42 means data split will be same everytime when we execute the code. This will provide us the consistency of the results and we’ll be able to reach better understanding of the results.

For loop created to repeat over each feature, 8 feature in total, and creates a histogram using Seaborn’s “histplot” function. After setting the bins and adding kernal density, plots displayed.

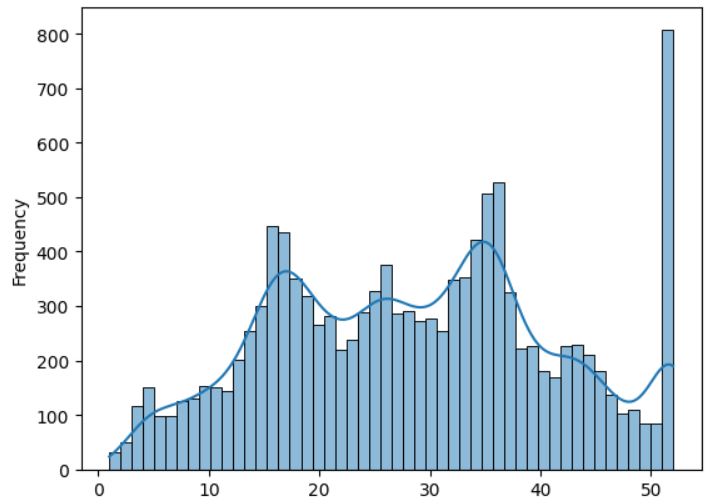
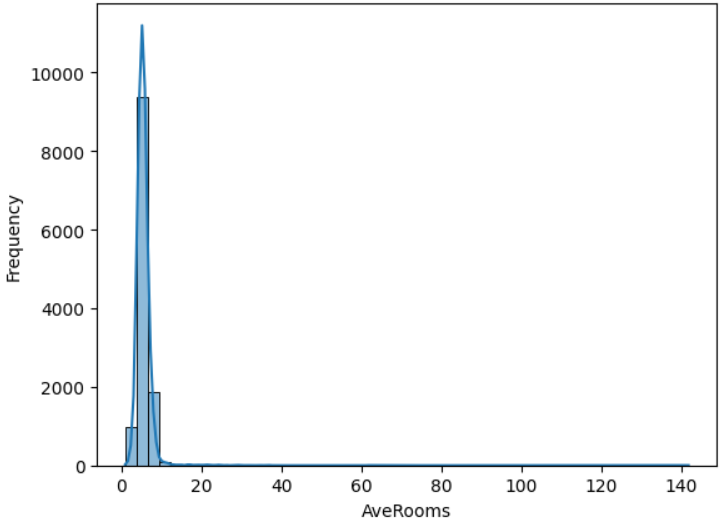
Below histogram of the median income , histogram displayed below shows us the distribution of the median income values. Other feature histograms will be shown at the bottom.

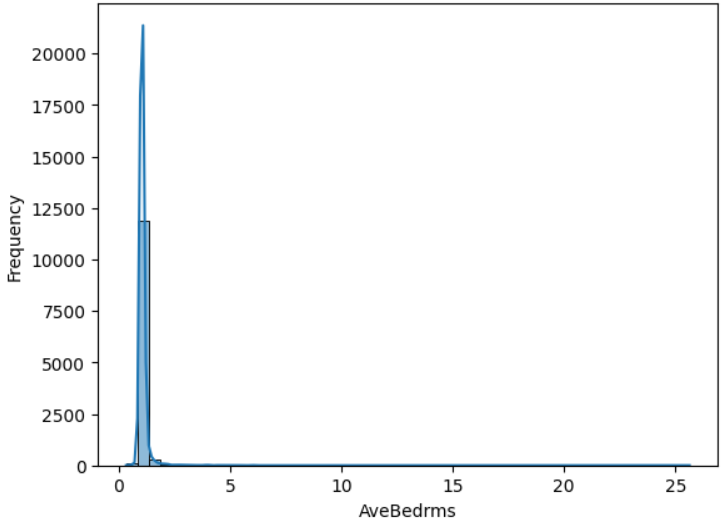
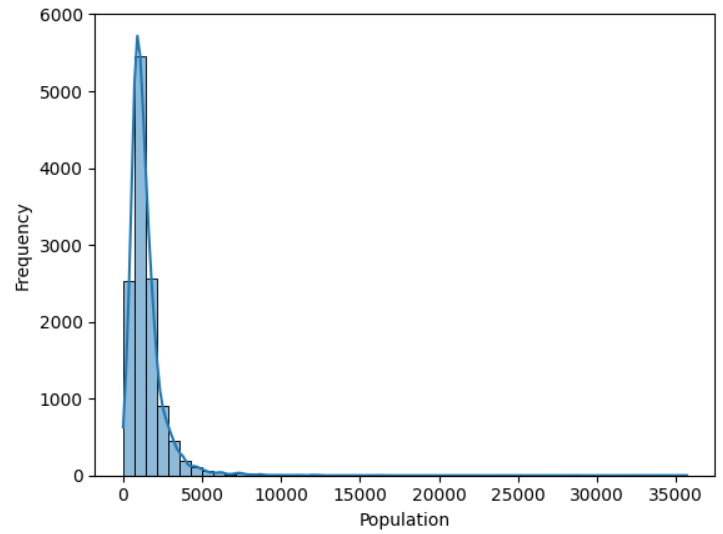


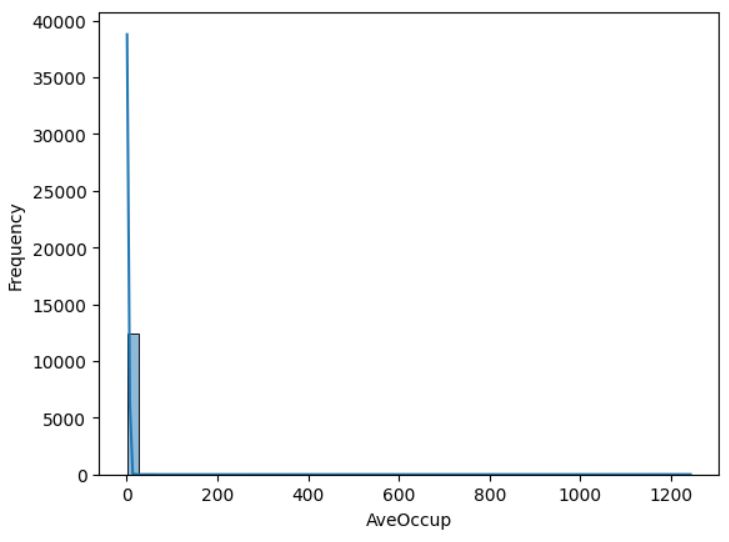
The x axis represents the median income and the y axis represents the frequency. As we mentioned above, histogram gives us idea about the distribution of median incomes in the dataset. In this case, we can come to conclusion that histogram is right-skewed. It is clear to define it as right-skewed since there are more districts with lower median incomes and less with higher median incomes

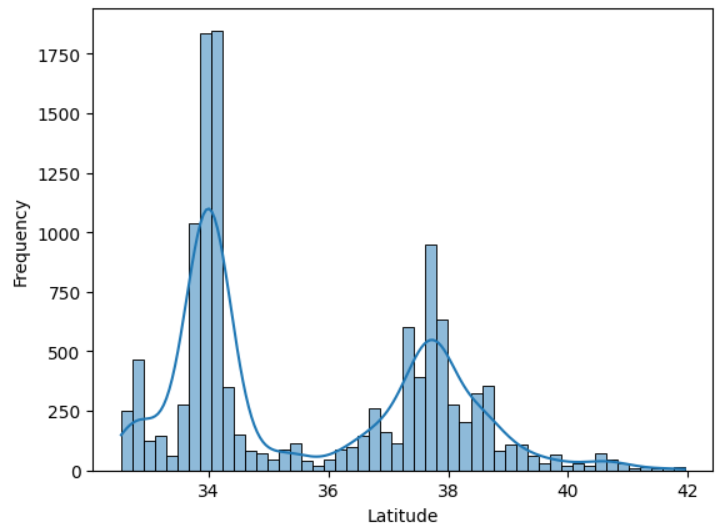
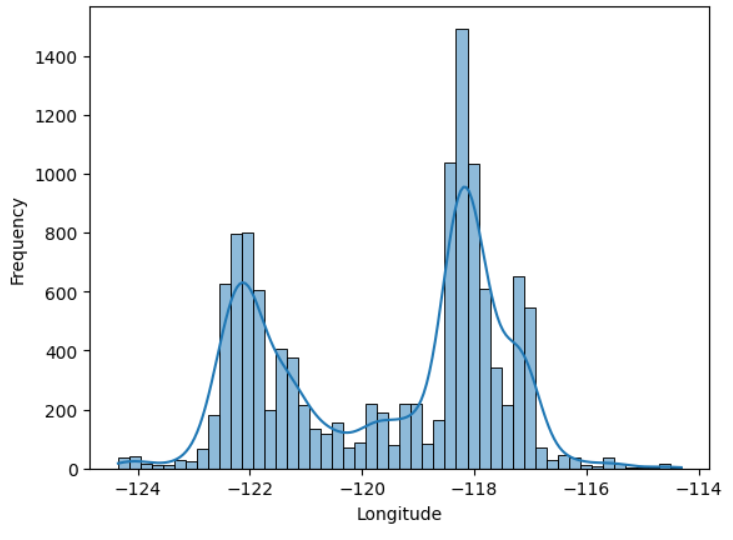


Lastly, data is normalized. “StandardScaler()” is used to ensure that all inputs have the same scale. Normalized data printed and displayed. Below, remaining histograms for other 7 features can be seen.

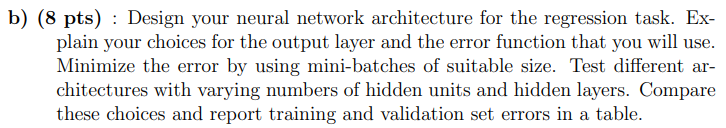
 



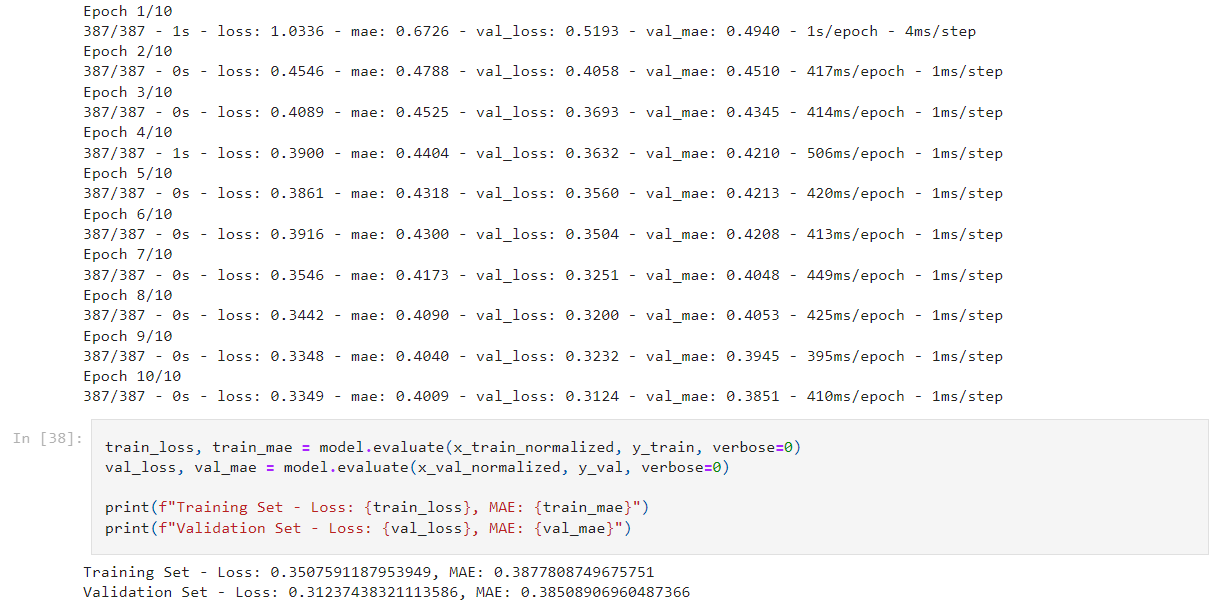
 

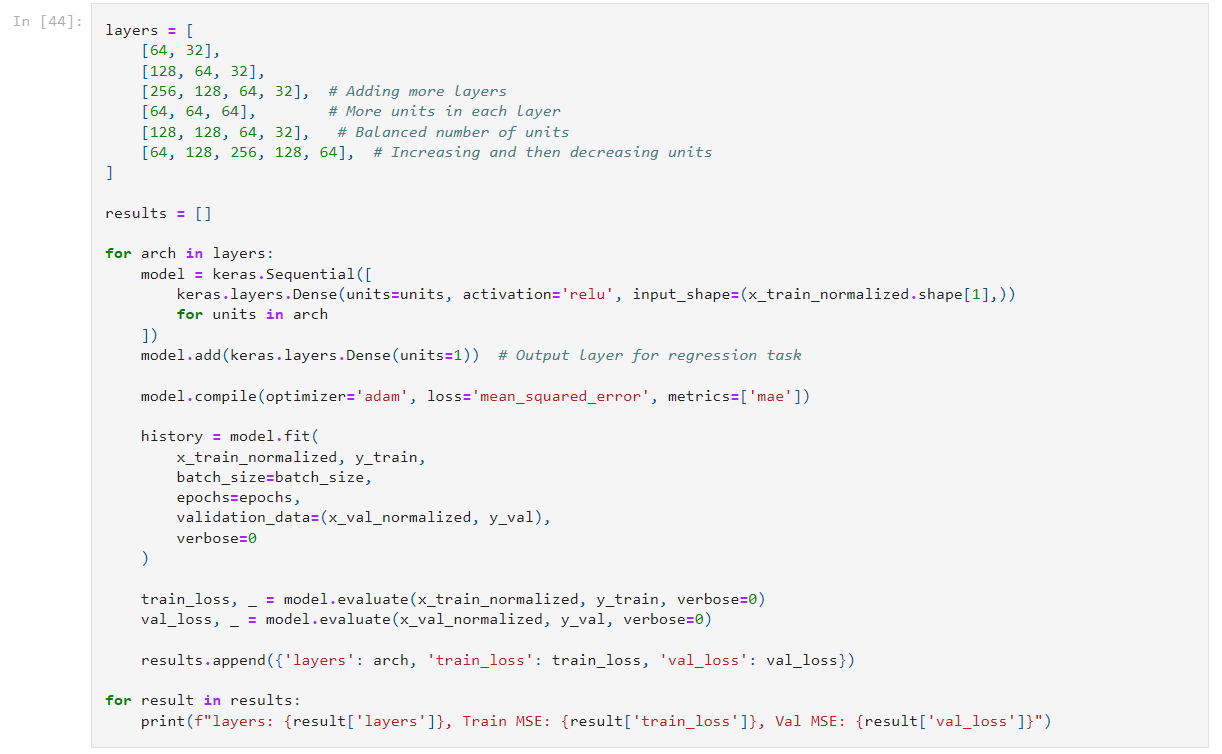
Lastly, data is normalized. “StandardScale()” is used to ensure that all inputs have the same scale. Normalized data printed and displayed.

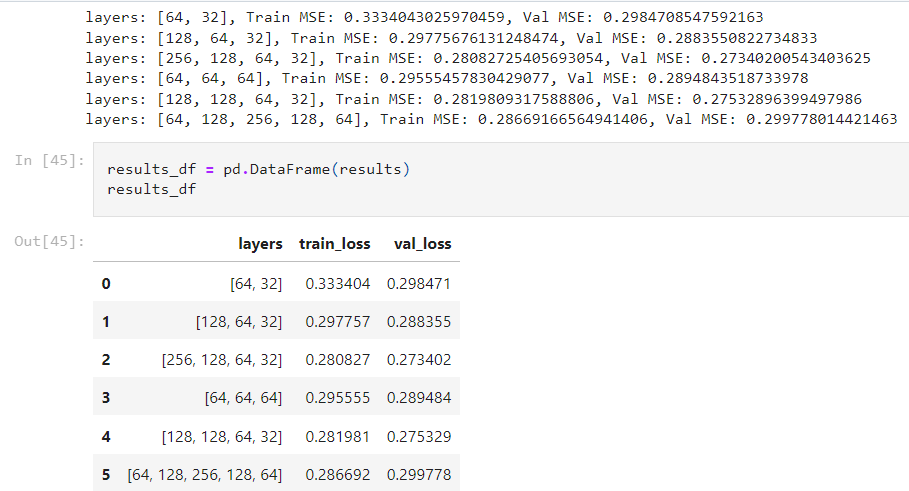


Code:







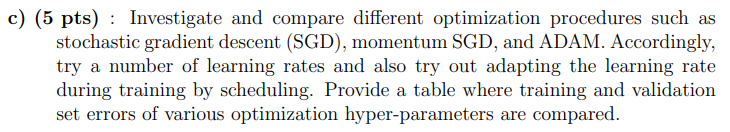


**Interperations:**

Since we have regression task like predicting the house prices, we choose **output layer** as single dense layer with one unit. We believe it is an apporpriate choice to predict for the median house value. **For Error function**, we choose MSE(Mean Squared Error). Reason behind it is MSE ables to penalizes larger errors better than smaller errors and in our case since we have a regression task, it will be important to accurately predict the magnitude of the target variable. For house price prediction, we need to accurately estimate the difference between predicted and actual values which is critical. By choosing the combination of single dense output layer with one unit and the MSE function suits well for our task to predict house prices

**Mini-batch** training applied to minimize the error, which is efficient to apply it on large datasets. Batch size selected as 32 to balance computational effiency and model convergence. Different architectures tested and all models are compiled with the help of Adam optimizer and the chosen MSE function and it trained for 10 epochs on the normalized training set by using mini batches.

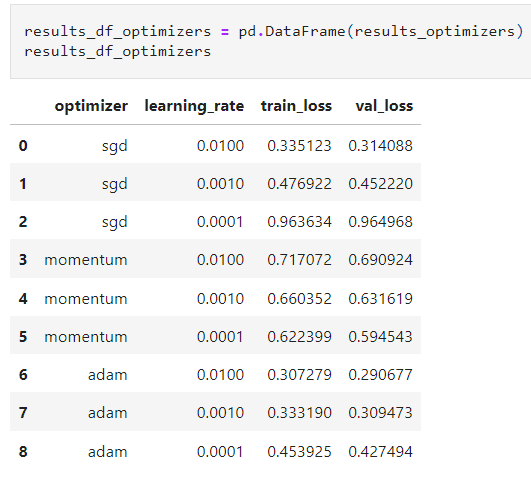
To compare the training and validation set MSE for each architecture are displayed on the table as its needed at task. When we examine the table, we can say that models generally had reasonable performance with MSE values range approximately from 0.27 to 0.33. As a architectural comparison, [256,128,64,32] hidden units performs the best in terms of MSE value. Lower MSE value indicates better performance. In this case [256,128,64,32] has the best performance among other models. We can say that more complex model performs better.

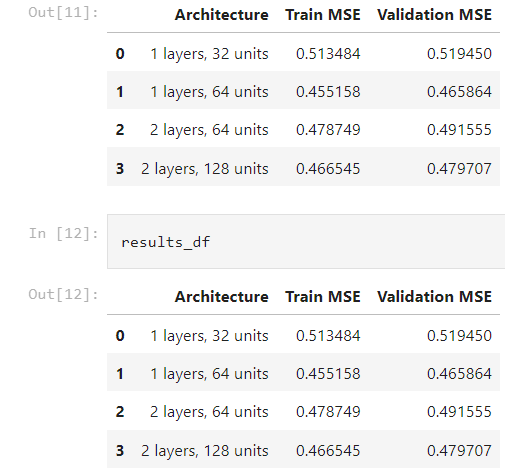
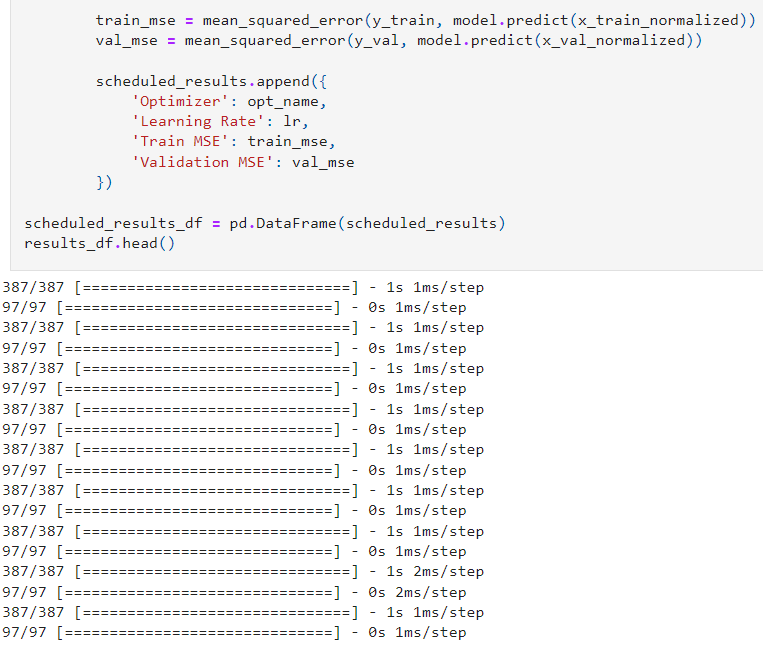
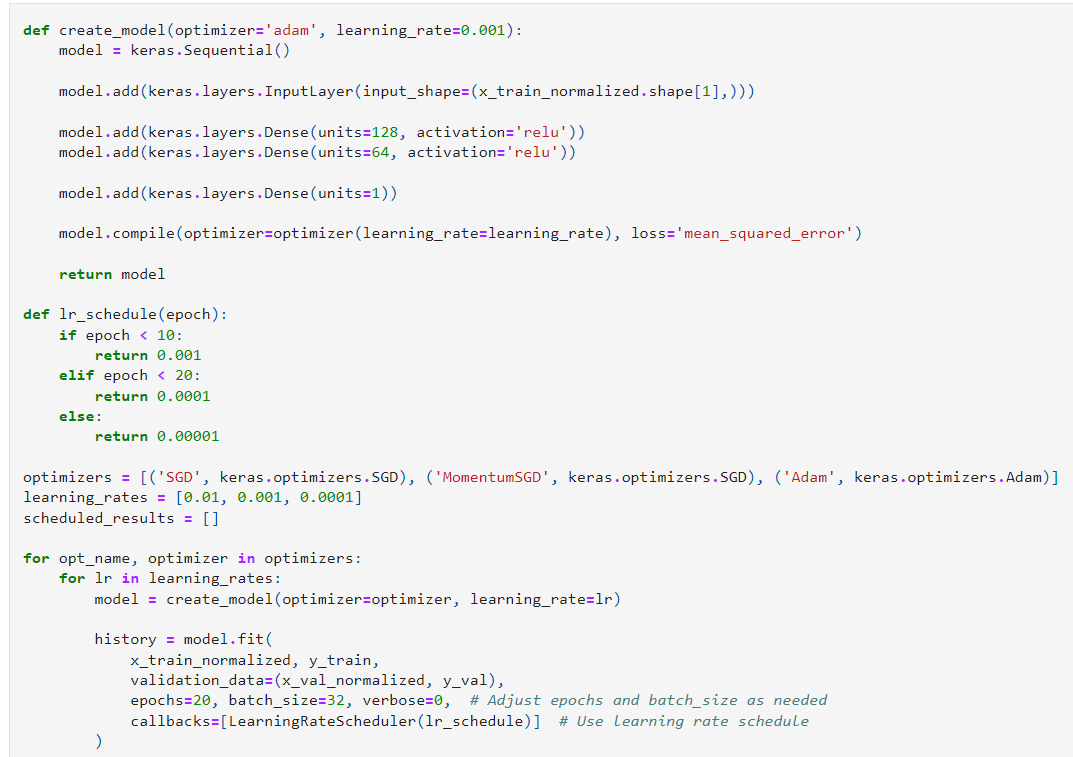


Code:









**Interperations:**

Different optimization procedures such as SGD,momentum SGD and ADAM investigated. Learning rate scheduling and architectural experiments have been done.

Obtained results are: Learning rates are used to control how fast or slow the model learns. When we look at the SGD and momentum SGD, smaller sensitivity rate often performs better. ADAM achieves faster convergence and lower validation loss. With a higher learning rate(0.1000) ADAM learns fast with low validation loss. In general after experiments, we come up conclusions that ADAM i a smart learner at our neural network , when we use ADAM as a optimizer, it adapts well without needing many adjustments. We can call it as flexible.

SGD optimizer works through data slowly but steadily. It doesn’t rush and makes small steps to minimize error. It can be very useful at large and complex data since it is harder to deal with it and by using SGD optimizer, we can feel more comfartable since it will handle the process more cautious. SGD momentum provided us good results especially at low learning rate but it was slower. ADAM consistenly provided lower losses compared to SGD and SGD momentum. It demonstrated better optimization results in training and validation ser errors.

