<https://github.com/Christian-Martens-UNCC/ECGR-4106/tree/main/Homework_0-Multi_Layer_Perceptron>

Problem 1:

1. It does not appear that my network needs more epochs for full training as the training and validation accuracy have converged and are pretty high. There does appear to be some very slight overfitting at the very end of the training since the training accuracy is climbing slightly while the testing accuracy is pretty much stationary.

Text

Description automatically generatedChart, line chart

Description automatically generated

1. The training results are very similar to the baseline. The validation and training accuracies are more inconsistent but there also seems to be less overfitting compared to the baseline model. During other training sessions, this method also resulted in a ~2% increase in validation accuracy.

Graphical user interface, text, application

Description automatically generatedText

Description automatically generated

Chart, line chart

Description automatically generated

1. The baseline model and the weight decay model both performed similarly to the dropout model. This method was consistently faster than the other two however, likely due to the reduced number of computations that needed completing. During other training sessions, this method generally performed with the highest validation accuracy, sometimes reaching as high as 90%.

Text

Description automatically generatedText

Description automatically generated

Chart, line chart

Description automatically generated

1. The training times for these both models were not much better than the untrained models. As we can see, there was not much change between the weights of the baseline model and the weights of the weight decay and dropout models. This method did allow us to further train our model, which was used to confirm that training had completed and that the models was slowly drifting towards overfitting if they continued for additional epochs.

Chart

Description automatically generatedChart

Description automatically generated

Problem 2:

1. When we standardize the continuous numerical features of a dataset, the features are condensed such that all the features are relatively the same size compared to one another. This makes sure that the model doesn’t get dominated by features with large input values and allows features with smaller input values to contribute to training the model. This overall helps with generalization.
2. The model complexity was increased by adding in more hidden layers with more neurons. It is hard to compare training times since the model in class was ran on a different computer, but it appears that my model has a lower validation loss than the model in the lecture. I would expect the model to take longer to train given it has more layers and thus more computations.

Chart

Description automatically generatedText

Description automatically generated

1. This model trained significantly faster than the previous model (roughly a 10% decrease in training time). Additionally, we can see that these methods did not increase the validation loss.

Graphical user interface, text, application

Description automatically generated

Chart

Description automatically generated

1. My predictions to Kaggle were not as good as I would’ve liked. Had I more time, I would’ve gone back and tried other ways of implementing Pytorch so that the regression is more accurate. My predictions were all very high compared to what I was expecting.