James Helm
CS 461
UMKC Fall 2021

For my neural network creation I primarily followed tutorials I found both on Medium and in Tensorflow/Keras API documentation, the links of which can be found in the neural net code submitted with this report. Through those tutorials I was able to extend the span of the topics to cover the assignment and, along with some trial and error in batch configuration, I've been getting some relatively good loss results.

Initially the program iterates through and formats the input csv file into a dataframe that is then manipulated through Pandas. Non-numeric columns are translated using one-hotting and dummy columns. The thousand sample dataframe is then separated into three numpy arrays: seventy percent training, fifteen percent validation, and fifteen percent testing. From there the Keras compiler and optimizer are set up, the latter of which implements gradient descent and an exponential decay learning rate to mitigate the chances of overfitting in creating the model. A fit function trains the model using the training sets, and an evaluate function tests the model and outputs the accuracy of the test. Finally, the program outputs the losses and the accuracy percentage estimates.

From a structural stance, the model implements 6 total layers: an input layer, two hidden layers, and 3 output layers. According to the formula I found in my research and some trial and error, 32 neurons in the two hidden layers gave the best and most consistent results. The input layer itself has 16 neurons, one for each input being tested, and the output layers have 1 neuron each.

I would have preferred a larger dataset so I could get even further consistency, but as it stands the model, calculated with mean absolute error, is roughly 88 to 92 percent accurate during testing. While I'm skeptical of this accuracy, I'm not seeing anything in the program code that would skew that data.

For future research and investigation, I would like to look into implementing the capability to enter in custom data for testing, run it by the model, and have it predict what grade each of the tests will be, given the model. This, however, goes beyond the scope of the assignment. I'd also like to pry open the black box that is the Keras API and see the inner workings for deeper understanding and hypertuning.