***Deep Learning Challenge: Charity Funding Predictor***

***Overview:***

The non-profit foundation Alphabet Soup wants to create an algorithm to predict whether or not applicants for funding will be successful. With knowledge of machine learning and neural networks, we must use the features in the provided dataset to create a binary classifier that is capable of predicting whether applicants will be successful if funded by Alphabet Soup.

**Results:**

Initially, I dropped the EIN and NAME from the dataset, per the instructions, assuming they both were not relevant as features. I then determined a cutoff of values for APPLICATION and CLASSIFICATION columns, binning those values below the cuttoff into a bin labeled ‘other’. This allows for the use of these columns as features while the cuttoff deceases the number of columns after the get\_dummies() function is applied, which I then did. The data was then split for training and testing, and a neural network (see figure below) was then created and ran. The results were consistently at or around .732 accuracy. I then saved the model as AlphabetSoupCharity.h5.

Seeing that the results were below the goal of at least .750 accuracy, model optimization was performed. The number of hidden layers was adjusted, along with the number of nodes, using various figures. This alone did not cause much of a change in the accuracy of +-.732. Next, the activation functions in the inner layers of the model were tried. In total, relu, selu, and tanh were used with not much of a difference in outcome. Finally, the number of epochs was adjusted from 50, to 75, to 100, to 150. Using 50 epochs worsened the results and using 75 to 150 resulted in roughly similar results. Next, changing the cuttoff figure for the APPLICATION and CLASSIFICATION columns so that more or fewer values were included in the ‘other’ bin were tried, with little change in the model accuracy.

Finally, the NAMES column, which was omitted entirely as irrelevant from the start, was reincluded as a feature after looking at the number of values and binning the low value counts into a ‘other’ bin, similar to what was done for APPLICATION and CLASSIFICATION. Doing this greatly increased the column count from 46 to 452 after the get\_dummies() function was applied. While this decision may have increased the processing time slightly, it significantly increased the accuracy to .781 (using relu as the activation function for the inner layers). Trying selu and tanh increased the accuracy to .788. An optimized model was then saved as AlphabetSoupCharity\_Optimization.h5.

**Compiling, Training, and Evaluating the Model:**

There were three layers total for each model after applying Neural Networks. The number of hidden nodes were dictated by the number of features.

page1image28594864

