**Overview:**

The nonprofit foundation Alphabet Soup wants a tool that can help it select the applicants for funding with the best chance of success in their ventures. With your knowledge of machine learning and neural networks, you’ll use the features in the provided dataset to create a binary classifier that can predict whether applicants will be successful if funded by Alphabet Soup.

**Results:**

We first removed the EIN and NAME columns and considered the remaining columns features for the model. The data was then split for training and testing sets. The target variable for the model was labeled “IS\_SUCCESSFUL” as per the starter code instructions and has the value of 1 for yes and 0 for no. I looked at APPLICATION\_TYPE value counts for binning and decided to bin based on application types that had more than 800. These were replaced with “Other”.

A screen shot of a computer program

Description automatically generated

“CLASSIFICATION value was also used for binning as well. The classifications that had less than 1000 were replaced with “Other”. I chose to use this value only because it was implied in the Starter code to use less than 1000 by looking at the screenshot below.

A screenshot of a computer

Description automatically generated

Categorical data was converted to numeric with pd.get dummies.

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

Once the data was scaled a neural network was applied on each model. I used 3 layers since it was explained in class that usually 3 layers is good enough while 2 layers can present inaccuracies.

A screenshot of a computer program

Description automatically generated

A three-layer training model generated 5900 parameters. The first attempt was already very accurate at 100% meeting the requirements for 75% or more. There was a loss of 2.9 so I changed the units to 80, 30 and 1 which was the original output of the starter code and the loss improved. Being able to see the output on the code made it easier to determine with all honesty.

A screen shot of a computer code

Description automatically generated

Optimization: The second attempt achieved an accuracy of almost 100% but with 9440 parameters. I replaced the application types with values over 500 instead of 800 and I also updated the units in each layer to 80, 30 and 1. Deep learning models should have multiple layers as mentioned before, since it helps learn how to predict and data.

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated