**Alphabet Soup Charity Predictor Report**

Angela Angulo

**Overview**

The purpose of this analysis was to create a binary classifier algorithm, using machine learning and neural networks, that can be used to predict whether or not an applicant requesting funding from Alphabet Soup will be successful if granted. A CSV with metadata on over 34,000 organizations that have previously received funding from Alphabet Soup was used to create, train, test, and optimize the model in an attempt to achieve 75% or higher predictive accuracy.

**Results**

Data Processing

The target for the model was identified as the variable **IS\_SUCCESSFUL**, which denoted whether the funding granted to the organization was used effectively.

The variables that were considered features for the model were as follows:

* **APPLICATION\_TYPE**—Alphabet Soup application type
* **AFFILIATION**—Affiliated sector of industry
* **CLASSIFICATION**—Government organization classification
* **USE\_CASE**—Use case for funding
* **ORGANIZATION**—Organization type
* **INCOME\_AMT**—Income classification
* **SPECIAL\_CONSIDERATIONS**—Special consideration for application
* **ASK\_AMT**—Funding amount requested

The variables determined as neither targets nor features and removed from the input data were:

* **EIN—**Identification column
* **NAME—**Identification column
* **STATUS**—Active status

Compiling, Training, and Evaluating the Model

The original attempt to build the neural network model achieved a predictive accuracy score of 0.7299, failing to reach the desired target of 75%. Three attempts were made to improve the model’s performance:

* *Optimization Attempt #1* focused on reevaluating the input data and adjusting the model’s features, which reduced the number of features upon transformation from 43 to 32. The modifications resulted in a slightly increased accuracy score of 0.7304, and were thus kept in place for later attempts. Changes that were made include:
  + Dropping the STATUS column, as all but five entries had their status marked as ‘1’
  + Rebinning the data under APPLICATION\_TYPE such that all application types with under 1000 entries (increased from 500) were combined and marked as ‘Other’
  + Rebinning the data under INCOME\_AMT to combine the six categories with less than 1000 entries into two new categories, ‘1-24999’ and ‘1M-50M+’
  + Rebinning the data under AFFILIATION to combine the four categories with under 100 entries into the category ‘Other’
* *Optimization Attempt #2* focused on reconfiguring the number of hidden layers and neurons in the model. The adjustments resulted in an accuracy score of 0.7307, a minor increase from the first attempt. Changes that were made include:
  + Adding a third hidden layer with 75 nodes
  + Increasing the number of nodes in the first and second layers from 80 to 250 and from 30 to 125, respectively
* *Optimization Attempt #3* focused on the activation functions for the hidden layers. Changing the activation from ReLU to tanh decreased the accuracy score to 0.7285. Adjusting the number of epochs to the training regime did not improve this score, and thus was ultimately set at 100 similar to previous attempts.

Despite these attempts to improve its performance, none of the models were able to reach the desired target accuracy of 75%. The model with the best performance, Optimized Model #2, achieved a 0.7307 predictive accuracy. It contained 32 input features, three hidden layers with 250, 125, and 75 neurons respectively (chosen after testing various combinations for the number of layers and nodes) and using the ReLU activation function (most commonly used for classification models), and an output layer with one unit and relying on the Sigmoid activation function (necessary parameters for predicting binary classification).

**Summary**

Though the second attempt at optimization had the highest accuracy score of 0.7307, none of the modifications made were able to produce a model with 75% accuracy in predicting whether an applicant will be successful if provided funding from Alphabet Soup. Including more data, preprocessing the data differently, and/or continued testing for optimal hyperparameters (possibly with an automated process using KerasTuner) may result in an improved model. However, an alternative to this could be instead using a Logistic Regression model, which are commonly used for binary classification and would likely take less time to train and less processing power to run. Future analyses should compare the current neural network model with a Logistic Regression model to see which would be best to use.