Alphabet Soup Charity Analysis

Paul Keller

**OVERVIEW**

The purpose of this analysis is to create a neural network model that could give a fair prediction as to whether a proposal to the non-profit organization Alphabet Soup would be successful or unsuccessful. This type of model is useful so that funds can be allocated to successful projects in order to maximize the utility the proposal yields.

The initial neural network model used all features in the data set, except for the name and of the entity identification number. This neural network had two hidden layers. The first layer had 80 nodes, and the second layer had 30 nodes. The activation function for each of these layers was a rectified linear unit function. The activation function for the outer layer was a sigmoid function. Below are the accuracy and loss generated by the model:

A screenshot of a computer code

Description automatically generated

After numerous attempts to produce a neural network model that had at least 75% accuracy, the best that could be achieved was a level of about 60%. This model was the best of four models. The activation functions of each layer remain the same as the first model. The difference between the models was that the first hidden layer had 40 nodes, and the second hidden layer had 25 nodes. Below are the accuracy and loss generated by the model:

A screenshot of a computer code

Description automatically generated

**RESULTS**

**Data Preprocessing**

* The target for the model is a binary output for whether the proposal is successful, a value of 1, or unsuccessful, a value of 0.
* The features for the model are application type, affiliation, classification type, case of use, organization, status, income amount, special consideration of application, and the asking amount.
* The variables that were removed from the dataset were the organization name and its identification number.

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

* The number of hidden layers used for both models was two. The activation functions for both hidden layers were a rectified linear unit function for both models and the activation function for the outer layer was a sigmoid function for both models. The difference was the number of nodes. For the first model, the first layer had 80 nodes, and the second layer had 30 nodes. For the second model, the first layer had 40 nodes, and the second layer had 25 nodes. The sigmoid function seemed to be the only function yielding accuracy results of greater than 50% for the outer layer. The hyperbolic tangent function consistently gave poor accuracy scores. The number of hidden layers only seemed to matter if there was at least one. Beyond that, the number of layers did not have much impact on the accuracy of the model. The first layer generally needed more nodes than the second layer to be more accurate, but not by too much.
* The target performance of the model was improved, but not to the 75% level. The first model had a 53% level of accuracy, and the improved model had a 60% level of accuracy.
* The improvement of the model came in reducing the number of nodes in each of the hidden layers.

**SUMMARY**

While the model was improved with changes, it was not to a satisfactory level of accuracy. If it were to be adapted again, the first place that should be addressed is dropping more features from the dataset. Perhaps a good place to start would be the special considerations feature.