Neural Network Model Report

Overview

In this analysis, machine learning and neural networks were used to determine if organizations applying to receive funding from Alphabet Soup will be successful--if and when funding is granted. More than 34,000 organizations were analyzed who have received funding from Alphabet Soup in the past. The goal of the analysis was to train the machine to have above 75% accuracy for predicting successful applicants.

Results

Data Preprocessing

The target variable for the model is:

- IS_SUCCESSFUL

The variables that are the features of the model are:

- APPLICATION_TYPE
- CLASSIFICATION

The variables that were removed from the input because they were neither targets nor features are:

- EIN
- NAME (NAME was ultimately added back in the 2nd analysis)

Compiling, Training, and Evaluating the Model

- A three-layer model was used which generated 763 parameters. Within each layer, multiples of 7 were used for the number of neurons.
- Two different activation functions were used, ReLU and Sigmoid. These help demonstrate binary networks, which is appropriate for the desired IS_SUCCESSFUL output, where we are hoping to observe a binary “yes” or “no” of successful funding.

```
[ ] # Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.
    number_input_features = len(X_train_scaled[0])
    hidden_nodes_layer1=7
    hidden_nodes_layer2=14
    hidden_nodes_layer3=21
    nn = tf.keras.models.Sequential()

    nn = tf.keras.models.Sequential()

    nn = tf.keras.models.Sequential()

    # First hidden layer
    nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer1, input_dim=number_input_features, activation='relu'))

    # Second hidden layer
    nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer2, activation='relu'))

    # Output layer
    nn.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))

    # Check the structure of the model
    nn.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_3 (Dense)             (None, 7)                 763

 dense_4 (Dense)             (None, 14)                112

 dense_5 (Dense)             (None, 1)                 15

=================================================================
Total params: 890 (3.48 KB)
Trainable params: 890 (3.48 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Yes, we were able to achieve target model performance in the second attempt seen in AlphabetSoupCharity_Optimization.ipynb. The second attempt had 79% accuracy. The first attempt was unsuccessful, with an accuracy of 73%.

In order to achieve model performance the following changes were made:

- The binning value cutoff was changed from >1 to <50 to only include the most prevalent classifications.
- NAME was added back as one of the features.

Summary

In summary, we were able to achieve target performance by focusing on the most common classification types and adding in name as one of the features. These features helped train the model to predict with a higher level of accuracy whether applicants would be successful. A different model could solve the classification problem by using a model with greater correlation between input and output. This would involve more data cleanup and management in advance of data processing by the machine. Other activation functions may also be able to more accurately predict successful applicants.