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. This neural network will assist in helping Alphabet Soup select future successful candidates.

Results:

Data Preprocessing

- What variable(s) are the target(s) for your model?
  - The target variable would be if the campaign was successful listed in the data as (Is_Successfu)
- What variable(s) are the features for your model
  - Application type, classification are binned and the list of features:
    - APPLICATION TYPE
    - AFFILIATION
    - CLASSIFICATION
    - ORGANIZATION
    - STATUS
    - INCOME
    - SPECIAL CONSIDERATIONS
    - ASKING AMOUNT
    - USE CASE
- What variable(s) should be removed from the input data because they are neither targets nor features?
  - The variables that were removed as they were neither targets no features were the Tax ID "EIN", and the name of the company "NAME".

Compiling, Training, and Evaluating the Model

- How many neurons, layers, and activation functions did you select for your neural network model, and why?

```
# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.
num_input = len(X_train[0])
h1_1 = 10
h1_2 = 15

nn = tf.keras.models.Sequential()

# First hidden layer
nn.add(
    tf.keras.layers.Dense(units=h1_1, input_dim=num_input, activation="relu")
)
# Second hidden layer
nn.add(
    tf.keras.layers.Dense(units=h1_2, activation="relu")
)
# Output layer
nn.add(tf.keras.layers.Dense(units=1, activation="sigmoid"))

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

- 25 Neuron in total, 10 in the first layer and 15 in the second layer
- Two layers in total
- These were chosen base on assistance from the instructor along with trial and error.

Optimized Data

- I was not able to achieve the target model performance at the end it had reached 72.9% slight increase from 72.7%
- To increase performance, the following steps were taken:
  - The column “Use Case” was dropped
  - The Values of the classification decreased to 500
  - A third layer was added including a different node count

- Fewer Epochs where ran

```
[ ]  # Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer
     num_input = len(X_train[0])
     h1_1 = 10
     h1_2 = 15
     h1_3 = 10

     nn = tf.keras.models.Sequential()

     # First hidden layer
     nn.add(
         tf.keras.layers.Dense(units=h1_1, input_dim=num_input, activation="relu")
     )
     # Second hidden layer
     nn.add(
         tf.keras.layers.Dense(units=h1_2, activation="relu")
     )

     #Third hidden layer
     nn.add(
         tf.keras.layers.Dense(units=h1_3, activation="relu")
     )
     # Output layer
     nn.add(tf.keras.layers.Dense(units=1, activation="sigmoid"))
```

Summary:

We can determine the success of a given charity in moderately high accuracy. In both the initial trial and optimization, the learning model had a success rate between 72-73%. The ‘kerastuner’ ‘model’ may be helpful in the future in being able to determine the shape of the neural network. Using the “brute force” method was time consuming and overall, not very efficient, or effective. If recreating a model this would be the changes implemented.