Alphabet Soup Analysis

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. Using machine learning and neural networks, use the features in the provided dataset (charity_data) to create a binary classifier that can predict whether applicants will be successful if funded by Alphabet Soup.

Results:

1. Data Preprocessing
   - What variable(s) are the target(s) for your model?
     - We want to target successful ventures in this project.
   - What variable(s) are the features for your model?
     - All other data was defined as features within the model (APPLICATION_TYPE, AFFILIATION, CLASSIFICATION, USE_CASE, ORGANIZATION, STATUS, INCOME_AMT, SPECIAL_CONSIDERATIONS, ASK_AMT).

```
In [10]: # Split our preprocessed data into our features and target arrays
         X = application_dummies.drop('IS_SUCCESSFUL', axis=1).values
         y = application_dummies['IS_SUCCESSFUL'].values

         # Split the preprocessed data into a training and testing dataset
         X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 42)
```

   - What variable(s) should be removed from the input data because they are neither targets nor features?
     - We removed EIN and NAME (identification columns) as unnecessary data.
2. Compiling, Training, and Evaluating the Model
   - How many neurons, layers, and activation functions did you select for your neural network model, and why?
     - I selected 2 layers in addition to the outer layer with values of 22, 18, and 13 because to me they seemed like a good balance. With this I got an accuracy of 0.7290. I also tried two other options for the values, 16, 13, and 7 which came back at 0.7263 and 80, 65, and 40 which came back at 0.7277. They were all almost identical, but I went with the one that was on average the best.

```
In [12]: # Define the model
         #  YOUR CODE GOES HERE
         number_input_features = X_train_scaled.shape[1]

         # Deep neural net- the number of input features and hidden nodes for each layer
         hidden_node_layer1 = 22
         hidden_node_layer2 = 18
         hidden_node_layer3 = 13
```

- Were you able to achieve the target model performance?
  - The first run through I achieved a model performance value of 0.7290 accuracy, so I did not reach the goal of 75%.

```
In [15]: # Evaluate the model using the test data
         model_loss, model_accuracy = nn.evaluate(X_test_scaled,y_test,verbose=2)
         print(f"Loss: {model_loss}, Accuracy: {model_accuracy}")

         268/268 - 0s - loss: 0.5579 - accuracy: 0.7290 - 254ms/epoch - 949us/step
         Loss: 0.5579401850700378, Accuracy: 0.7289795875549316
```

- What steps did you take in your attempts to increase model performance?
  - To achieve an increased model performance, I first tried binning a couple of the other columns including USE_CASE and ASK_AMT with no success. The accuracy remained almost the exact same value.
  - Next, I tried something different, I added the name column back into the data frame and tried binning that column. I figured, if there were companies that had multiple ventures could be important in predicting success based on previous experiences and the success and failure rates of those. At first, I set the name_count to 50 and achieved a model accuracy of 0.7608. Just to see what would happen, I then set the name_count to 15 and increased the accuracy to 0.7736, both of which helped us hit our goal.

```
In [4]: # Look at APPLICATION_TYPE value counts for binning
        name_count = dropped_df.value_counts("NAME")
        name_count

Out[4]: NAME
        PARENT BOOSTER USA INC                          1260
        TOPS CLUB INC                                    765
        UNITED STATES BOWLING CONGRESS INC               700
        WASHINGTON STATE UNIVERSITY                      492
        AMATEUR ATHLETIC UNION OF THE UNITED STATES INC  408
                                                        ...
        FUSION ARTS UNLIMITED INC                          1
        FURRST AND FURRMOST POOCH SANCTUARY INC            1
        FURNISHING HOPE AZ INC                             1
        FUNNY RIVER EMERGENCY SERVICES                     1
        ZURICH PUBLISHING FOUNDATION INC                   1
        Length: 19568, dtype: int64

In [5]: # Choose a cutoff value and create a list of application types to be replaced
        # use the variable name `application_types_to_replace`
        name_to_replace = list(name_count[name_count<15].index)

        # Replace in dataframe
        for name in name_to_replace:
            dropped_df['NAME'] = dropped_df['NAME'].replace(name,"Other")

        # Check to make sure binning was successful
        dropped_df['NAME'].value_counts()

Out[5]: Other                                  21751
        PARENT BOOSTER USA INC                  1260
        TOPS CLUB INC                            765
        UNITED STATES BOWLING CONGRESS INC       700
        WASHINGTON STATE UNIVERSITY              492
                                                ...
        CALIFORNIA FEDERATION OF WOMENS CLUBS     15
        JAMESTOWNE SOCIETY                        15
        THETA TAU                                 15
        LAKE TRABIS ATHLETIC BOOSTER CLUB INC     15
        INTERNATIONAL ASSOCIATION OF LIONS CLUB   15
        Name: NAME, Length: 161, dtype: int64
```

```
In [17]: # Evaluate the model using the test data
         model_loss, model_accuracy = nn.evaluate(X_test_scaled,y_test,verbose=2)
         print(f"Loss: {model_loss}, Accuracy: {model_accuracy}")

         268/268 - 0s - loss: 0.4709 - accuracy: 0.7736 - 251ms/epoch - 936us/step
         Loss: 0.4708895683288574, Accuracy: 0.7736443281173706
```

Summary:

The deep learning models should have multiple layers. The number of neurons didn't add as much value when it comes to improving the averages as the adjustment of features did. The model reached the desired accuracy value of 75%, but I am not confident that an accuracy of 75% is something that I would want to place confidence in when it comes to decision making on revenue based decisions for a company.