Overview of the Analysis:

The purpose of this project was to create a tool for a fictitious company called Alphabet Soup Charity. The tool is to help select applicants for funding with the best chance of success in their ventures. To create the tool a data set of over 34,000 previous applications was used to train and test a neural network.

Results:

Data Preprocessing

- The variable that is the target for this model is the "**IS_SUCCESSFUL**" column as it indicates
- The variable(s) are the features for this model are the "**APPLICATION_TYPE**", "**AFFILIATION**", "**CLASSIFICATION**", "**USE_CASE**", "**ORGANIZATION**", "**STATUS**", "**INCOME_AMT**", "**SPECIAL_CONSIDERATIONS**", and "**ASK_AMT**"
- The variable(s) that should be removed from the input data because they are neither targets nor features and the ones identifying the organization submitting the application: "**EIN**", and "**NAME**"

Compiling, Training, and Evaluating the Model

- Three layers were selected for this model as it is somewhat complex and takes it from a basic neural network to more complex. The two hidden networks activation were "relu" as this is a standard activation to use and the output layer was sigmoid.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 80)                3520

 dense_1 (Dense)             (None, 30)                2430

 dense_2 (Dense)             (None, 1)                 31

=================================================================
Total params: 5981 (23.36 KB)
Trainable params: 5981 (23.36 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

- This model did NOT achieve the target model performance of 75% accuracy.

```
# 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 - 1s - loss: 0.5594 - accuracy: 0.7291 - 506ms/epoch - 2ms/step
Loss: 0.5594168901443481, Accuracy: 0.7290962338447571
```

- To increase model performance the following modifications were made:
  - Removing two columns (reducing number of features) that had responses that were the same for the majority applying organizations ("**STATUS**" and "**SPECIAL_CONSIDERATIONS**")
    - total number of rows = 34299
    - Status with a "1" = 34294 (99.98%)
    - Special Considerations with an "N" = 34272 (99.92%)
  - More neurons were added to the two hidden layers
  - An additional hidden layer with "relu" activation was added

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 100)               4100

 dense_1 (Dense)             (None, 50)                5050

 dense_2 (Dense)             (None, 25)                1275

 dense_3 (Dense)             (None, 1)                 26

=================================================================
Total params: 10451 (40.82 KB)
Trainable params: 10451 (40.82 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

  - The modifications did not improve the model and did not achieve 75^ accuray.

```
# 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 - 1s - loss: 0.5627 - accuracy: 0.7287 - 549ms/epoch - 2ms/step
Loss: 0.5627292990684509, Accuracy: 0.7287463545799255
```

Summary:

The model selected came close to 75% on the training data (about 74% for both the original model and the optimized version of the model). Evaluation of the test data in both cases resulted in about 73% accuracy. It might be necessary to do some hyper parameter optimization to check if there are outliers. Additionally it might be beneficial to separate the different types of organization based on some of the factors that were used as features, for instance, there ma be more consistency within a subcategory such that there is higher accuracy in the prediction.