Deep Learning Challenge

Overview

The purpose of this analysis was to develop a machine learning model for Alphabet Soup, a non-profit foundation, that can reliably predict the success of applicants if funded.

Results

Data Preprocessing

In the subject model, the target variable which is what is being predicted is the column 'IS_SUCCESSFUL'. The features are all the other columns which provide information used by the model to make predictions, these columns include – 'APPLICATION_TYPE', 'AFFILIATION', 'CLASSIFICATION', 'USE_CASE', 'ORGANIZATION', 'STATUS', 'INCOME_AMT', 'SPECIAL_CONSIDERATIONS' and 'ASK_AMT'. The variables/ columns 'EIN' and "NAME' which are identification columns were removed from the input data.

Compiling, Training, and Evaluating the Model

In the model's design, three layers were selected. The initial hidden layer, equipped with 80 neurons, was structured to capture the dataset's intricate patterns. The subsequent hidden layer, with 30 neurons, further refined these patterns. Lastly, the output layer had a single neuron, dedicated to the binary classification of determining an applicant's success.

```
# 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[0])
hidden_nodes_layer1 =  80
hidden_nodes_layer2 = 30

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"
_________________________________________________________________
 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 neural network model achieved an accuracy of 73.1% which was close to the target model performance of 75%.

```
# 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: 0.5629 - accuracy: 0.7314 - 582ms/epoch - 2ms/step
Loss: 0.5629305839538574, Accuracy: 0.7314285635948181
```

Optimization

In an attempt to optimize the model and achieve target model performance, the column ‘NAME’ which was previously excluded from the initial model was reincorporated as a feature. As a result, the model's accuracy significantly increased to 79.6%, up from the original 73.1%.

```
# 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: 0.5043 - accuracy: 0.7956 - 588ms/epoch - 2ms/step
Loss: 0.504274308681488, Accuracy: 0.7955685257911682
```

Summary

The deep learning model, structured with three layers, was proficient at capturing intricate patterns from the dataset, achieving target performance after optimization. While the model is effective, for future attempts, it may be beneficial to try Random Forest, a model known to handle a variety of data without extensive preprocessing. It combines multiple decision trees, to produce better classification results. Given the dataset's complexity and the binary nature of the classification, the Random Forest could be a more straightforward and equally effective solution.