Step 4: Write a Report on the Neural Network Model

Report on the performance of the deep learning model created for Alphabet Soup.

1. **Overview** of the analysis:

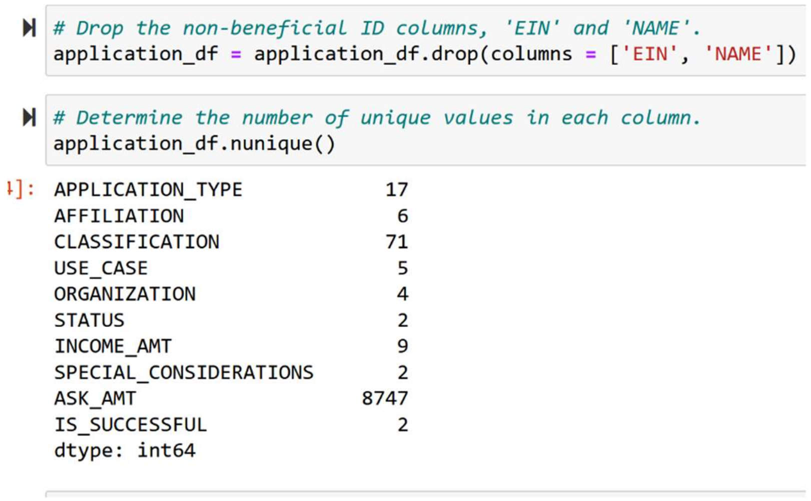
The nonprofit foundation Alphabet Soup wants a tool that can help it select the applicants for funding with the best chance of success in their ventures. With our knowledge of machine learning and neural networks, we have used the features in the provided dataset to create a binary classifier that can predict whether applicants will be successful if funded by Alphabet Soup.

Dataset received, a CSV file, contains more than 34,000 organizations that have received funding from Alphabet Soup over the years.

2. **Results**:

- Data Preprocessing

Variable that were neither targets or features for the dataset were the Columns that were dropped: EIN, NAME due to having little to no impact on the outcome



```
# Drop the non-beneficial ID columns, 'EIN' and 'NAME'.
application_df = application_df.drop(columns = ['EIN', 'NAME'])
```

```
# Determine the number of unique values in each column.
application_df.nunique()
```

```
]:  APPLICATION_TYPE            17
    AFFILIATION                  6
    CLASSIFICATION              71
    USE_CASE                     5
    ORGANIZATION                 4
    STATUS                       2
    INCOME_AMT                   9
    SPECIAL_CONSIDERATIONS       2
    ASK_AMT                   8747
    IS_SUCCESSFUL                2
    dtype: int64
```

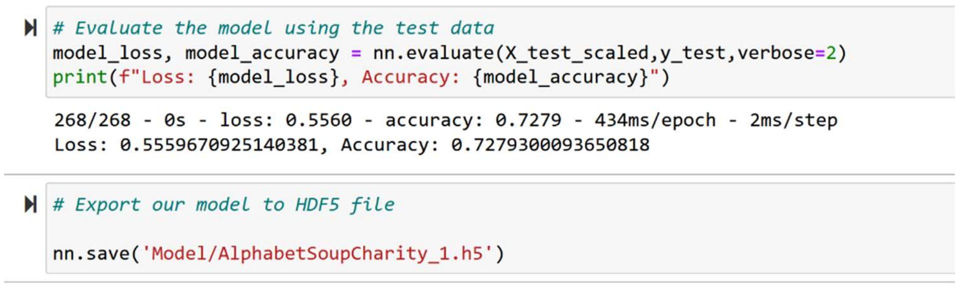
Target variable for the my model: IS_SUCCESSFUL Column

Features variable for the model: The rest of the above listed columns except the IS_SUCCESSFUL

- Compiling, Training, and Evaluating the Model

For the neural network model we had 2 hidden layers at 80 and 30 neurons, with the uutput layer. The first and second hidden layer have the "relu" activation function and the output layer is "sigmoid.

Using TensorFlow, this model did not achieve the a target predictive accuracy higher than 75%. The accuracy for this model was 72%



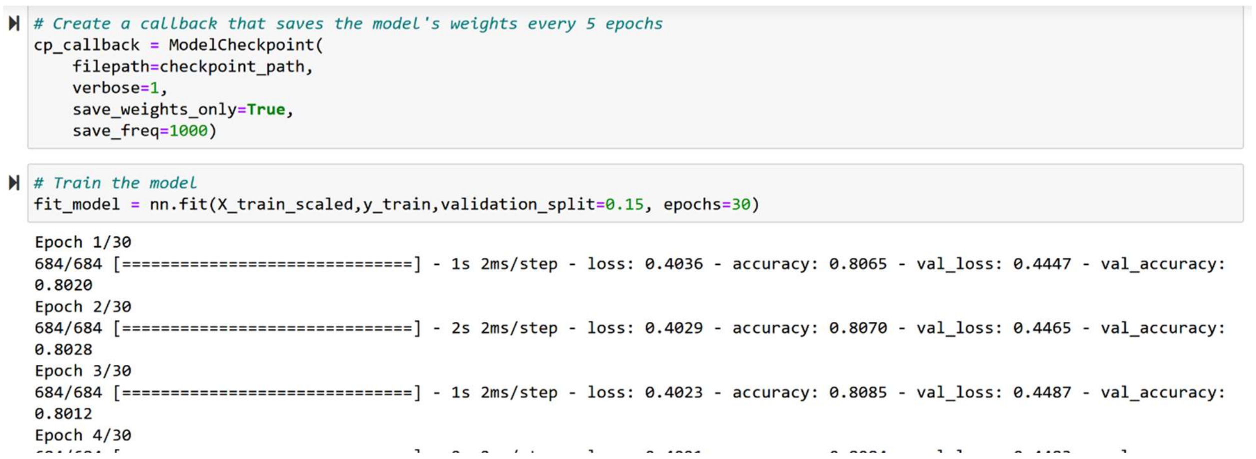
```
# Evaluate the model using the test data
model_loss, model_accuracy = nn.evaluate(X_test_scaled,y_test,verbose=2)
print(f"Loss: {model_loss}, Accuracy: {model_accuracy}")
```

```
268/268 - 0s - loss: 0.5560 - accuracy: 0.7279 - 434ms/epoch - 2ms/step
Loss: 0.5559670925140381, Accuracy: 0.7279300093650818
```

```
# Export our model to HDF5 file

nn.save('Model/AlphabetSoupCharity_1.h5')
```

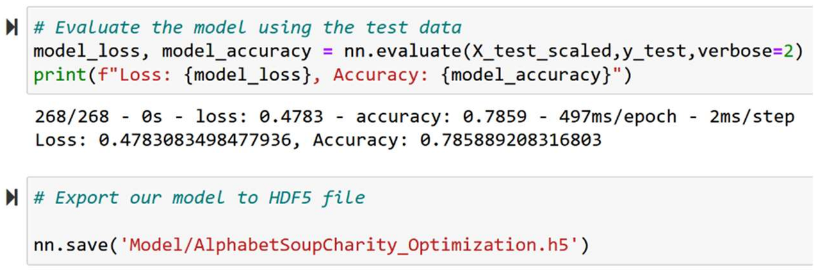
We've made further adjustments and parameters of value counts, set model's weight every 5 epochs, and hidden layers values and were able to successful reach the target accuracy of 78.6%



```
# Create a callback that saves the model's weights every 5 epochs
cp_callback = ModelCheckpoint(
    filepath=checkpoint_path,
    verbose=1,
    save_weights_only=True,
    save_freq=1000)
```

```
# Train the model
fit_model = nn.fit(X_train_scaled,y_train,validation_split=0.15, epochs=30)
```

```
Epoch 1/30
684/684 [==============================] - 1s 2ms/step - loss: 0.4036 - accuracy: 0.8065 - val_loss: 0.4447 - val_accuracy:
0.8020
Epoch 2/30
684/684 [==============================] - 2s 2ms/step - loss: 0.4029 - accuracy: 0.8070 - val_loss: 0.4465 - val_accuracy:
0.8028
Epoch 3/30
684/684 [==============================] - 1s 2ms/step - loss: 0.4023 - accuracy: 0.8085 - val_loss: 0.4487 - val_accuracy:
0.8012
Epoch 4/30
```



```
# 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: 0.4783 - accuracy: 0.7859 - 497ms/epoch - 2ms/step
Loss: 0.4783083498477936, Accuracy: 0.785889208316803
```

```
# Export our model to HDF5 file

nn.save('Model/AlphabetSoupCharity_Optimization.h5')
```

3. **Summary**:

After optimization the model was able to achieve accuracy score of 78.6%. The loss of accuracy from the first attempt could be attributed to the a possibility that the model was overfitted. There are additional features that could be removed (such as USE_CASE), and different activation functions for the outer layer to fir the model and test the model for higher accuracy.