Deep Learning Project: Charity Funding Predictor Model

Intro:

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 your knowledge of machine learning and neural networks, you'll use the features in the provided dataset to create a binary classifier that can predict whether applicants will be successful if funded by Alphabet Soup.

Run through:
In the start of the processing we removed some of the columns that might be deemed to be irrelevant and in this initial run we decided on EIN and NAME columns to be surplus (we kept the NAME column during the second / Optimization run). The target Variable in the model is labeled IS_SUCCESSFUL with values of 1 for yes and 0 for no. To avoid huge fluctuations we employed binning and setting cutoff points to the APPLICATION and CLASSIFICATION values. Categorical values were encoded by the get_dummies() function.

Initial compiling, training and evaluating the model:
Total 435 params were created by the three layer training model. Accuracy level stood at ~ 73% and this was slightly below the 75% threshold.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 7)                 308

 dense_1 (Dense)             (None, 14)                112

 dense_2 (Dense)             (None, 1)                 15

=================================================================
Total params: 435 (1.70 KB)
Trainable params: 435 (1.70 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
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: 0.5592 - accuracy: 0.7313 - 530ms/epoch - 2ms/step
Loss: 0.5592454075813293, Accuracy: 0.7313119769096375
```

Optimization:
In this second run of the deep learning sequence we left the “NAME” column in the data set, and saw an increased overall accuracy (~ 79%) of the model.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 7)                 3171

 dense_1 (Dense)             (None, 14)                112

 dense_2 (Dense)             (None, 1)                 15

=================================================================
Total params: 3298 (12.88 KB)
Trainable params: 3298 (12.88 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
# 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: 0.4592 - accuracy: 0.7888 - 349ms/epoch - 1ms/step
Loss: 0.45915648341178894, Accuracy: 0.7888046503067017
```

Summary:

Multiple layers should be used in deep learning models, as it's the best way for the models to classify and predict information based on filtering inputs through various layers.