1. **Overview**:

The purpose of the analysis is that 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 my knowledge of machine learning and neural networks, I 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.

1. **Results**:

* Data Preprocessing
  + *What variables are the target for your model?*

IS\_SUCCESSFUL

* + *What variables are the features of your model?*

Cut off values and binning for APPLICATION\_TYPE and binning for CLASSIFICATION.

* + *What variable(s) should be removed from the input data because they are neither targets nor features?*

I dropped EIN, NAME for the first model. For the second optimization, I added the NAME.

* *Compiling, Training, and Evaluating the Model*

* + How many neurons, layers, and activation functions did you select for your neural network model, and why?

For the first model, I gave two hidden layers 10 and 5. The percentage of accuracy is only 73. For the second optimization, I used the hidden layers 75 and 30. The accuracy is 79%, That’s pretty much a good prediction.

* + Were you able to achieve the target model performance?

Yes, I was able to achieve the target model performance.

* + What steps did you take in your attempts to increase model performance?

For the first model, I dropped EIN and Name, to increase model performance, I dropped only EIN. Also, replace some names with others and bin the classification. Total parameters 36, 286.

1. **Summary**:

For the optimization:

# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.

units\_1 = 75

units\_2 = 30

input\_features = len(X\_train\_scaled[0])

nn\_model1 = tf.keras.models.Sequential()

# First hidden layer

nn\_model1.add(tf.keras.layers.Dense(units=units\_1, input\_dim = input\_features, activation = "relu"))

# Second hidden layer

nn\_model1.add(tf.keras.layers.Dense(units=units\_2, activation = "relu"))

# Output layer

nn\_model1.add(tf.keras.layers.Dense(units=1, activation="sigmoid"))

# Check the structure of the model

nn\_model1.summary()

Model: "sequential\_14"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

dense\_39 (Dense) (None, 75) 33975

dense\_40 (Dense) (None, 30) 2280

dense\_41 (Dense) (None, 1) 31

=================================================================

Total params: 36,286

Trainable params: 36,286

Non-trainable params: 0

**Accuracy:**

268/268 - 0s - loss: 0.4641 - accuracy: 0.7907 - 490ms/epoch - 2ms/step

Loss: 0.4641447961330414, Accuracy: 0.7906705737113953

The hidden layers are found from the features.

79% accuracy is a good prediction whether applicants will be successful if funded by alphabet soup.