**Deep Learning – Charity Funding Predictor**

**OVERVIEW:** The goal of this project is to create an algorithm using machine learning and neural networks to predict whether applicants will be successful if funded by the non-profit foundation.

**PROCESS:**

1. Read in CSV - Read in the CSV file using pandas. The file contained more than 34,000 organizations that have received funding from the [fictional] foundation, along with several columns of metadata about each organization.
2. Pre-processing
   1. Dropped the non-beneficial columns
   2. Found the number of data points for each unique value for each of the columns that had more than 10 unique values: APPLICATION\_TYPE and CLASSIFICATION, and in another attempt, ASK\_AMT
   3. Chose varying cutoff points for each model
   4. Used “pd.get\_dummies()” to convert categorical data to numeric
   5. Divided the data into target arrays and feature arrays
   6. Applied the “train\_test\_split” to create a testing and training dataset
   7. And finally, using the “StandardScaler” to scale the training and testing sets
3. Compiling, Training, and Evaluated the Model

The model was required to achieve a target predictive accuracy higher than 75%. I made 3 attempts using machine learning and neural networks. Each model resulted in an accuracy rate between 73 and 74%.

**MODELS:**

Attempt 1 (Initial Attempt):

With this model, I used two layers of neurons and both activation functions used were sigmoid.

APPLICATION\_TYPE cutoff = 450

CLASSIFICATION cutoff = 500

Loss: 0.553830623626709, Accuracy: 0.731195330619812

**Hyperparameters used:**

layer1: 20 neurons; activation function = sigmoid

layer2: 10 neurons; activation function = sigmoid

epochs=100

A loss value of 55% indicates that the model can be further optimized. The accuracy percent shows that **73%** of the model's predicted values align with the true values in the original dataset. Additional attempts could be made to try to reach 75% accuracy.

Attempt 2 (Optimization Attempt 1):

With this model, I changed the cutoffs, replaced another variable, and added an additional layer.

APPLICATION\_TYPE cutoff = 500

CLASSIFICATION cutoff = 1,000

Loss: 0.5530775189399719, Accuracy: 0.7313119769096375

**Hyperparameters used:**

layer1: 10 neurons; activation function = relu

layer2: 8 neurons; activation function = sigmoid

layer3: 6 neurons; activation function = sigmoid

epochs=25

This model did not result in much improvement at all. A loss value of 55% indicates that the model can be further optimized. The accuracy percent shows that **73%** of the model's predicted values align with the true values in the original dataset. Additional attempts could be made to try to reach 75% accuracy.

Attempt 3 (Optimization Attempt 2):

With this model, I used the keras tuner hyperband module to implement hyperparameter optimization.

APPLICATION\_TYPE cutoff = 500

CLASSIFICATION cutoff = 1,000

**Accuracy:** 0.7399417161941528

epochs=25

This model did not result in much improvement but is a bit closer to 75% accuracy. The accuracy percent shows that **74%** of the model's predicted values align with the true values in the original dataset. Additional attempts could be made to try to reach 75% accuracy, but since this is my third attempt, I will circle back to this at a later date.

My final model would be the recommended model from my attempts due to its higher accuracy rate. However, in an ideal scenario, I would be able to explore additional optimization techniques to reach an accuracy of 75% or higher.