## Deep Learning and Neural Network

AlphabetSoup Analysis

**Overview**

In this analysis, I attempted to create a deep learning and neural network model to assist the charity, AlphabetSoup, determine which funding applicants would successfully utilize any allocated funds. The organization provided a csv file with previous funding partners who had received various amounts for various charities and whether those funds were used successfully. This file included the fund amounts both applied for and allocated, the purpose (use case) of those funds, and organization data like name, affiliation, and classification.

Second the categorical data was transformed into numerical data and the values pivoted. For example, instead of having the application types listed by row, they were pivoted so that each value has a unique column, and the row now contains a True or False. From here I was able to build an experimental model the train and test.

**Results**

**Data Preprocessing:**

* A cleanup of the csv was required. To limit the amount of data being fed into the model, unique data like EIN and Name were removed from the file. These variables will not be used as Targets or Features.
* A count of the unique data values for the remaining columns was generated so that any outlying values that occurred infrequently could be grouped together in a new category called “other.”
  + Any occurrence within the column “Application\_Type” that occurred less than 500 times.
  + Any occurrence within the column "Classification” that occurred less than 10 times.
* Categorical data was transformed into numerical data. *For example, instead of having the application types listed by row, they were pivoted so that each value has a unique column, and the row now contains a True or False.*
* Model Target: Is\_Successful
* Model Features: Application\_Type, Affiliation, Classification, Use\_case, Organization, Status, Income\_Amt, Special\_Considerations, Ask\_Amt

**Compiling, Training, and Evaluating the Model:**

* Base Model Variables and Functions:
  + Random State = 78
  + 2 Hidden Layers:
    - H\_layer1 = 80
    - H\_layer2 = 30
    - Both Hidden layers used “relu” activation function
  + Output layer uses Sigmoid activation function.
  + Compilation:
    - Loss=”binary\_crossentropy”
    - Optimizer=”adam”
    - Metrics=”accuracy”
  + Training: 50 epochs
  + RESULTS
    - Loss: 55.24%
    - Accuracy: 72.57%
  + Saved to “AlphabetSoupCharity.h5”

**Model Optimization**

**Overall adjustments:**

* Random State for all models changed from 78 to 100.
* The Ask\_Amt column was binned reducing the number of variables from 8700+ to 12.

**Model #1**

* + 2 Hidden Layers:
    - H\_layer1 = 80
    - H\_layer2 = 30
    - Activation Function = “leaky\_relu”
  + Output layer uses Sigmoid activation function.
  + Compilation:
    - Loss=”binary\_crossentropy”
    - Optimizer=”adamax”
    - Metrics=”accuracy”
  + Training: 100 epochs
  + RESULTS
    - Loss: 56.51%
    - Accuracy: 72.66%
  + Saved to “AlphabetSoupCharity\_opt\_model\_1.h5”

**Model #2**

* + 3 Hidden Layers:
    - H\_layer1 = 50
    - H\_layer2 = 25
    - H\_layer3 = 12
    - Activation Function = “relu”
  + Output layer uses Sigmoid activation function.
  + Compilation:
    - Loss=”binary\_crossentropy”
    - Optimizer=”adam”
    - Metrics=”binary\_accuracy”
  + Training: 100 epochs
  + RESULTS
    - Loss: 58.96%
    - Accuracy: 72.48%
  + Saved to “AlphabetSoupCharity\_opt\_model\_2.h5”

**Model #3**

* + 3 Hidden Layers:
    - H\_layer1 = 7
    - H\_layer2 = 14
    - H\_layer3 = 28
    - Activation Function = “relu”
  + Output layer uses Sigmoid activation function.
  + Compilation:
    - Loss=”binary\_crossentropy”
    - Optimizer=”adamax”
    - Metrics=”binary\_accuracy”
  + Training: 100 epochs
  + RESULTS
    - Loss: 56.07%
    - Accuracy: 72.35%
  + Saved to “AlphabetSoupCharity\_opt\_model\_3.h5”