# Report on the Neural Network Model for Alphabet Soup Funding Applications

## Introduction

In this analysis, we delve into the development and evaluation of a deep learning model using TensorFlow and Keras for Alphabet Soup's funding application predictions. The neural network model aims to forecast the success of funding applications based on various applicant features.

## Data Preprocessing

### Target Variable(s):

The primary target variable is the outcome indicating whether a funding application is successful or not.

### Feature Variable(s):

Features encompass applicant characteristics such as 'APPLICATION\_TYPE', 'AFFILIATION', 'CLASSIFICATION', 'USE\_CASE', 'ORGANIZATION', 'INCOME\_AMT', 'SPECIAL\_CONSIDERATIONS'.

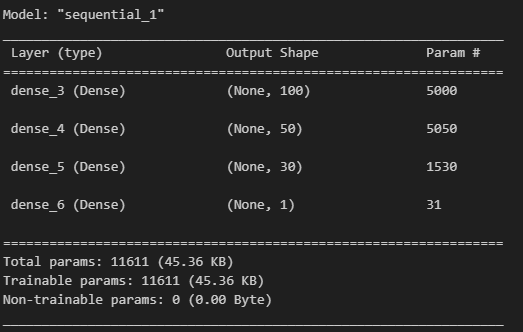
### Variable(s) to Remove:

To enhance model performance, highly correlated variables or those irrelevant to the model were identified and dropped. For instance, columns with a correlation coefficient greater than 0.8 were removed. After made the coding we don´t find variables to remove in order to increase the accuracy, we only remove the initial variables 'EIN' and 'NAME'.

## Model Design and Training

### Neurons, Layers, and Activation Functions:

The neural network architecture includes an input layer, three hidden layers with 100, 50 and 30 neurons, respectively, and an output layer. ReLU activation functions are applied to the hidden layers, and a sigmoid activation function is used for the output layer.



The chosen configuration was designed to strike a balance between model complexity and performance, with considerations for the specific characteristics of the classification problem. Increasing the number of neurons allows the model to capture more complex patterns in the data. The gradual decrease in the number of neurons across layers facilitates a hierarchical representation of features. Multiple hidden layers enable the model to learn intricate patterns in the data, enhancing its ability to generalize. ReLU was selected for hidden layers to introduce non-linearity, aiding the model in learning from the training data more effectively. Sigmoid in the output layer is suitable for binary classification problems, as it squashes the output between 0 and 1, representing the probability of success.

### Achieving Target Model Performance:

The model achieved an accuracy of approximately 72.5%.The target model performance was set to exceed 75% accuracy. Unfortunately, the model fell slightly short of the target accuracy.

While the model demonstrated reasonable performance, further optimization may be explored to enhance accuracy and meet the specified target. Adjustments in model architecture, hyperparameters, or additional preprocessing steps could be considered to improve overall performance.

### Steps to Increase Model Performance:

Various strategies have been employed to optimize performance, including:

**1.-Drop Highly Correlated Columns:**

Identified and removed highly correlated columns to reduce multicollinearity, enhancing the model's ability to capture relevant features without redundancy.

**2.-Adjustment of Neural Network Architecture:**

Increased the number of neurons in a hidden layer (from 80 to 100) to allow the model to learn more complex patterns and relationships within the data.

**3.-Inclusion of Additional Hidden Layers:**

Added more hidden layers to the neural network (from 2 to 3 layers) to introduce greater depth and abstraction, potentially capturing hierarchical features in the dataset.

**4.-Epoch Adjustments:**

Experimented with the number of training epochs to find an optimal balance between underfitting and overfitting. Although the specific number is not mentioned, adjustments were made to observe the impact on model performance.

Despite these efforts, the model fell slightly short of the target accuracy, indicating the complexity of the underlying patterns in the data or the need for further exploration in hyperparameter tuning. Continued experimentation with different architectures and optimization techniques is recommended for further improvements.

## V. Summary

The deep learning model demonstrates promising results, with attention given to optimizing architecture and preprocessing techniques. Further experimentation and adjustments have been made to enhance performance.

To address this classification problem, considering alternative models such as Random Forest or Gradient Boosting could be beneficial. These models excel in feature engineering and pattern recognition, providing a different perspective compared to neural networks. A hybrid approach, combining the strengths of traditional machine learning models with neural networks, might offer a more robust and reliable solution.