# **Deep Learning Challenge Report**

## Introduction & Overview

The non-profit foundation Alphabet Soup wants a tool that can help it select the applicants for funding with the best chance of success in their ventures. The challenge develops a deep learning model that can predict whether applicants will be successful if funded by Alphabet Soup. I was given a CSV file that I read into Pandas. This file contained more than 34,000 organisations that have received funding from Alphabet Soup over the years.

### **Data Pre-processing**

The data needed to be pre-processed so that it can be input into the deep learning model. Categorical variables are transformed into numerical variables via one hot encoding and binning. The data was then split into a training set and a testing set. Both sets were then scaled.

The target variable (y) was IS\_SUCCESSFUL. The data was split into training and test subsets. This tells us whether a funded organisation is successful in its venture and Alphabet Soup's money was used effectively.

These variables are the features for the model:

* **APPLICATION\_TYPE**—Alphabet Soup application type
* **AFFILIATION**—Affiliated sector of industry
* **CLASSIFICATION**—Government organisation classification
* **USE\_CASE**—Use case for funding
* **ORGANIZATION**—Organisation type
* **STATUS**—Active status
* **INCOME\_AMT**—Income classification
* **SPECIAL\_CONSIDERATIONS**—Special considerations for application
* **ASK\_AMT**—Funding amount requested.

**These two variables are removed from the input data:**

* EIN
* NAME

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

### The general rule of thumb is that having more hidden layers and neurons and having activation functions of the hidden layers that of the output layer give better accuracy. So, I experimented with increasing nodes and neurons, with changing other Parameters to get a better Accuracy, still all result came below the 75%. I couldn’t be able to achieve the target model performance 75%. In my third attempt of optimisation, I reach close to 73% accuracy though.

* + Steps taken to increase model performance.
    - Increasing or decreasing the number of values for each bin.
    - Add more neurons to a hidden layer.
    - Add more hidden layers.
    - Use different activation functions for the hidden layers.
    - Add number of epochs to the training regimen.
* **Results**
* Before optimisation, there are 2 hidden layers, the first of which has 80 neurons and the second of which has 30 neurons. All hidden layers use the relu function and the output layer uses the sigmoid function. Loss is 0.5575041770935059 and accuracy is 0.724897980690024. Epoch is 100.

A picture containing text, screenshot, font, number

Description automatically generated



* In my first attempt to optimise, I added the number of neurons of the first layer to 90 and of the second layer to 70 neurons. Third layer to 50 neurons. Loss was higher at 0.5603310465812683 and accuracy improve slightly to 0.7279300093650818

A picture containing text, screenshot, font, number

Description automatically generated­­­



* In the second attempt in optimisation, I added a third layer with 129 neurons ­­­­using a tanh function and increased the neurons in the second layer to 90 using tanh function again. Third layer using LeakyRelu and fourth layer using re­lu with output layer using sigmoid. Epoch is 100. Loss worsens slightly further to 0.572106122970581, and accuracy gone down slightly to 0.7258309125900269.

A screenshot of a computer

Description automatically generated with low confidence



* In my third optimisation attempt, added a first layer with 130 neurons, second layer with 100 neurons using relu function, third layer with 70 neurons using LeakyRelu and fourth layer with 40 neurons using sigmoid function. Loss is 0.5630198121070862 slightly better compared to last attempt and but with less accuracy of 0.7254810333251953 compared to previous attempt. Number of epochs I used in this attempt is 120.

A picture containing text, screenshot, font, number

Description automatically generated



## Summary and Conclusion

* The pre-optimisation model has a loss of 0.5575041770935059 and accuracy of 0.724781334400177. The first attempt in optimisation model has Loss of 0.5570106506347656 and accuracy improves to 0.7279300093650818 which is better compared to all other attempts of optimisation and pre optimisation. As we can see with all three optimisation attempts Still, I couldn’t achieve the target model performance of 75% accuracy despite having attempted several times to increase the model performance. With all attempts I could reach close to 73% accuracy rate only. I recommend considering different model to reach the target 75% model performance.