Alphabet Soup Charity Funding Predictor Models

# Overview

The non-profit foundation Alphabet Soup wants to create an algorithm to predict whether or not applicants for funding will be successful. With the knowledge of machine learning and neural networks, the model will utilize the features in the Charity.csv, which is a dataset that consists of more than 34,000 organizations that have received funding from Alphabet Soup over the years, to create a binary classifier that is capable of predicting whether applicants will be successful if funded by Alphabet Soup.

# Results

After a few optimizations, the model achieved ~80% accuracy, as described below.

## Data Preprocessing

The data provided contained the following columns:

1. EIN
2. Name
3. Application Type
4. Affiliation
5. Classification
6. Use Case
7. Organization
8. Status
9. Income Amount
10. Special Considerations
11. Ask Amount
12. Is Successful

**Target Variables**

**The target variable: “Is Successful”, can be considered binary as a value of 1 indicates success and a value of 0 indicates the lack of success.**

**Feature Variables**

**During the initial development of the model, these columns were presumed to have at least some influence on success of funding:**

* Affiliation
* Application Type
* Ask Amount
* Classification
* Income Amount
* Organization
* Status
* Special Considerations
* Use Case

**Upon initial review of the data set, it appeared that these variables had some influence on the success of a funding application. After optimization, it was determined that Name was also a feature variable and was included in the model.**

**Unnecessary Variables**

**Initially, the columns EIN and Name were excluded from the model. However, after optimization, Name was included as a feature variable.**

## Training and Evaluation

**Neurons, Layers and Activation Function**

**The first version of the model had two additional hidden layers after the input layer. The hidden layers each contained 69 and 29 nodes, respectively. The hidden layers both used the Rectified Linear Unit (ReLU) activation function. The output layer had a single node and used the Sigmoid activation for the binary output.**

**After the optimization (step 3), the model had three hidden layers with 100, 30 and 10 nodes each. The first hidden layer utilized the ReLU activation function. The other two used the Sigmoid activation function. The output layer still had a single node and used the Sigmoid activation function.**

**Model Performance**

**Initial model performance ended up being lower than the accuracy threshold of 75:**

Text

Description automatically generated

**The optimized model achieved a higher accuracy score of ~78%:**

Text

Description automatically generated

**Optimization Process**

**The first step in optimizing the model was to re-add the Name column to be a feature variable because won’t an organization’s track record influence the success of funding?**

**Beyond the data itself, the model parameters were change as indicated:**

* **Hidden Layer 1**
  + **Nodes increased from 69 to 100**
* **Hidden Layer 2**
  + **Nodes increase from 29 to 30**
  + **Activation function changed from ReLU to Sigmoid**
* **Hidden Layer 3**
  + **Additional hidden layer with 10 nodes**
  + **Activation function set to Sigmoid**

# Summary

On one hand, the initial results of the model came below of the minimum accuracy threshold of 75%, on the other, the optimized model had an increased accuracy of 78%. The difference being that one of the few columns that was initially excluded was reintroduced into the model. Let it be heard, not all initial assumptions about the model can be correct, some may even be the detriment of the model.