**REPORT**

**Overview**

The purpose of this analysis is to create a binary classifier using a deep learning neural network to predict the success rate of applicants for funding from Alphabet Soup, a nonprofit organization. The provided dataset contains information on over 34,000 organizations, including metadata such as application type, affiliated sector of industry, government organization classification, use case for funding, income classification, funding amount requested, and whether the money was used effectively. The analysis involves preprocessing the dataset by dropping unnecessary columns, encoding categorical variables, and splitting the data into training and testing datasets. The neural network model is then designed, trained, and evaluated to determine its loss and accuracy. Finally, the model is optimized using various methods such as adjusting input data, adding more neurons and hidden layers, using different activation functions, and adjusting the number of epochs. The ultimate goal is to achieve a predictive accuracy higher than 75% and save the optimized model as an HDF5 file.

**Results**:

**Data Pre-processing**

* What variable(s) are the target(s) for your model?

The target variable(s) for the model is IS\_SUCCESSFUL, as it represents the binary classification outcome variable of whether a charity donation was successful or not.

* What variable(s) are the feature(s) for your model?

The feature variables for the model are all the other columns in the DataFrame, excluding IS\_SUCCESSFUL.

* What variable(s) should be removed from the input data because they are neither targets nor features?

As part of the preprocessing phase for the analysis, I have identified that the EIN, NAME or Identification Columns do not contain relevant information for our predictive model. I excluded these variables from the feature and target selection, as it is not pertinent to the analysis.

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

* **How many neurons, layers, and activation functions did you select for your neural network model, and why?**

For this neural network model, I selected four hidden layers with 80, 50, 30 and 10 neurons respectively. I chose this combination after few iterations with different numbers of neurons and layers and verifying the accuracy.

For the activation functions, I chose ‘Relu’ for all the hidden layer to introduce non-linearity in the model and improve its performance. Finally, for the output layer, I used sigmoid activation function to ensure the output is between 0 and 1, which is needed for binary classification.

The initial Model gave an accuracy of 73%

* **Were you able to achieve the target model performance?**

I did not get the desired result with the first Model. I tried 3 more Optimizations of the Model.

In the first optimized run I did the following changes to the original:

1. Drop Columns 'EIN','NAME','SPECIAL\_CONSIDERATIONS’
2. Increase the Layer to 5 of values – 100, 80, 60. 40, 20

This however still resulted in an Accuracy of 73%

In the second optimization I made the following changes

1. DROP Columns -'EIN','NAME','SPECIAL\_CONSIDERATIONS'
2. Change the layers to 100, 75, 50, 25

The result was still around 73%

In the Third Optimization I achieved the desired accuracy with the following:

1. DROP Column -'EIN'
2. Keep the layers as 100, 75, 50, 25
3. I used ‘Relu’ for the first layer and ‘tanh’ for the remaining layers
4. Sigmoid was used as output

A screenshot of a computer

Description automatically generated

The Accuracy was 77%

A screenshot of a computer code

Description automatically generated

Keeping the NAME column improved the model accuracy. A cutoff value was chosen, and a list of names to replace was created using value counts. Any name that had less than 10 occurrences was replaced with "Other". Similarly, a cutoff value was chosen for the CLASSIFICATION column and any classification with fewer than 2000 occurrences was replaced with "Other".

**Summary**

In summary, the deep learning model using TensorFlow and Keras was able to achieve a predictive accuracy of 77% in classifying the success of organizations funded by Alphabet Soup based on their features. The model underwent several optimization attempts, including dropping columns, binning categorical variables, adding hidden layers and neurons, and trying different activation functions, among other adjustments. While the target predictive accuracy of 75% was achieved, it required significant optimization attempts to reach that level.

One recommendation for solving this classification problem would be to try a Random Forest Classifier or a Support Vector Machine (SVM) for a different type of model. These models have been shown to be effective in binary classification problems and may be able to achieve a higher accuracy without the need for extensive optimization attempts. Additionally, they can handle both numerical and categorical variables and can handle outliers and imbalanced datasets well, which may be present in this dataset. It may be worth exploring these alternative models as a potential solution to the classification problem.

**Note: The notebooks were created and tested using Google Colab. Some package installations and paths may need to be modified if running in a different environment. Also make sure that a new runtime is used for each notebook and close any already running sessions to prevent the execution from crashing due to exceeding the RAM.**

**Appendix**

**Screen shots of the models that were used are in this attached file.**

****