***Deep Learning Homework: Charity Funding Predictor***

**Overview**

Alphabet Soup is a non-profit foundation that provides funding to different organizations. Using the data from past funded projects and utilizing Deep Learning, the goal is to develop an algorithm that will help the foundation predict a project’s success prior to funding it.

**Results**

**Data Preprocessing**

The information from 34,000 past projects will be used as inputs in developing the prediction model. The column “IS\_SUCCESSFUL” is the target variable, this was used as the success/failure indicator in past projects. Two columns, “EIN” and “NAME” are being removed from the input data because they are only used to identify the requesting organizations. The remaining nine columns which includes “ASK\_AMT “, “USE\_CASE” and “APPLICATION\_TYPE” will comprise the features set. These are the information that will be analyzed and be the basis of the predicted outcome.

The dataset will be formatted before being fed to the deep learning module. Here is a picture of the formatted dataset.

Graphical user interface

Description automatically generated

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

The initial model is made up of 3 layers. I arbitrarily picked 2 Hidden Layers to start with and each layer has 23 neurons. The number of neurons is based on the mean of the inputs plus output nodes of the deep learning module. The mean came up to be 22 but I wanted to have an odd number of neurons, so I adjusted it to 23. The activation function I employed was the “Sigmoid”. I used this function after I ran a quick accuracy analysis between the “Sigmoid”, “Tanh” and “Relu” functions. I was going to optimize the model after the initial run, that is why I didn’t do a deeper analysis of the initial settings.

Graphical user interface, text, application, email

Description automatically generated

The initial model provided an accuracy score of 73.07% which is below the model performance target of 75%.

Text

Description automatically generated

I tried different optimization steps from increasing the number of epochs of the training regimen, comparing the different activation functions, and adjusting the number of neurons and layers, but all of these gave me comparable results to my initial model. In the end, I ran the Keras tuner to get the optimum number of layers and neurons and identify the activation function that will be used.

Text

Description automatically generated

Here is the result of the Keras tuner.

Text, letter

Description automatically generated

This model provided an accuracy score of 73.24% which is still below the model performance target of 75%.

Text

Description automatically generated

**Summary**

My best accuracy score was still below the target of 75% but going through the different optimization steps made me see the impact of adjusting the different parameters to the resulting model score. In hindsight, I think I should have analyzed more the impact of increasing my application type and classifications cut-offs so that more of the rare occurring events are grouped together. I also need to pay closer attention to the “ASK\_AMT” data, see if any outliers are present and thus needed to be removed. Another thing to look at, is to insert a stage before the Deep Learning portion, utilize a module like the PCA to decrease the number of input features. I think this type of hybrid modules will help improve the model tuning.