**Deep Learning Project**

**Charity Funding Predictor**

**By Karla R. Ellsworth**

**Aug 18, 2022**

**OVERVIEW**

I created an algorithm to predict whether or not applicants for funding will be successful. Deep Learning and Neural Networks were used in this project to create a binary classifier that is capable of predicting if applicants will be successful or not if funded by Alphabet Soup.

**RESULTS**

**. Data PreProcessing**

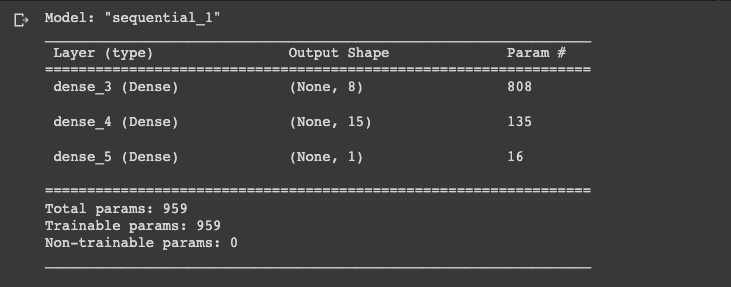
The dataset was used removing EIN which was not relevant information (it was nor a target nor a feature). The remaining columns were considered features for our model. Then the number of unique values for each column was determined, and for NAME, CLASSIFICATION and APPLICATION\_TYPE the number of data points for each unique value was determined. This number of data points for each unique value was used to pick a cutoff point to bin ‘rare’ categorical variables together in a new value, ‘Other’. The binning was checked then to verified if it was successful. Finally, I use pd.get\_dummies to encode categorical variables. Once these steps were completed, the data was splitted into training and testing datasets. Our Target Variable was IS\_SUCCESSFUL, this variable had the values 0 & 1: 1 meaning ‘Yes’, and 0 meaning ‘No’.

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

Using my knowledge of TensorFlow, I designed a Neural Network (Deep Learning Model) to create a binary classification model that predicts if an Alphabet Soup-funded organization will be successful based on the features in the dataset.

On the final model: I ended up using two layers, using the activation functions: ‘relu’ and ‘sigmoid’. The two layer training model generated 959 parameters.

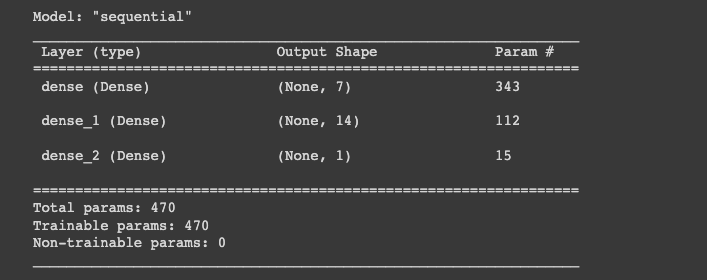
I was able to achieve the target model performance, achieving 76.30% (1% over the target).





**Optimization Steps**

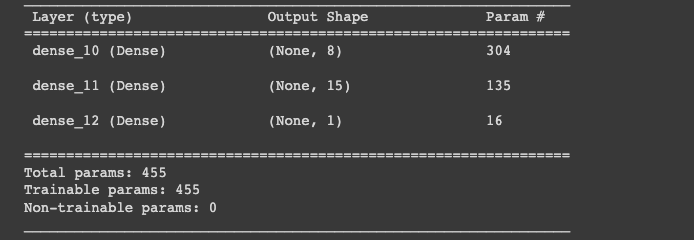
On the first attempt I removed the NAME from the dataset (It wasn’t use to determine unique values, and cutoff points to bin its rare categorical variables in a new value - ‘Other’). Then the number of unique values for each column was determined, and for CLASSIFICATION and APPLICATION\_TYPE the number of data points for each unique value was determined. This number of data points for each unique value was used to pick a cutoff point to bin ‘rare’ categorical variables together in a new value, ‘Other’.Two-layers were also used with 100 epochs on the training regimen. This produced an accuracy of 72.91%, with 470 parameters.

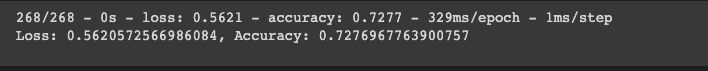




On my second attempt, I removed more columns from the original dataset: ’EIN', 'NAME', 'ORGANIZATION', 'USE\_CASE', 'SPECIAL\_CONSIDERATIONS'. Then the number of unique values for each column was determined, and for CLASSIFICATION and APPLICATION\_TYPE the number of data points for each unique value was determined. This number of data points for each unique value was used to pick a cutoff point to bin ‘rare’ categorical variables together in a new value, ‘Other’.

Two-layers were also used, increasing the number of epochs to 120 on the training regimen. This produced an accuracy of 72.76%, with 455 parameters. Which was still under the desired target model performance.

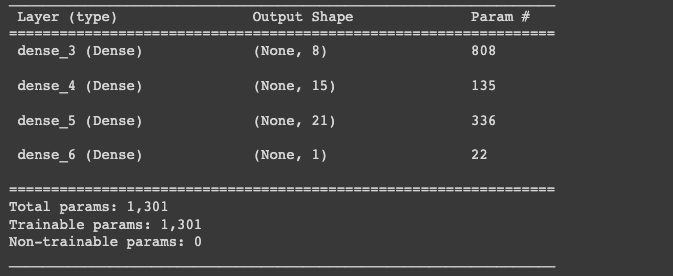


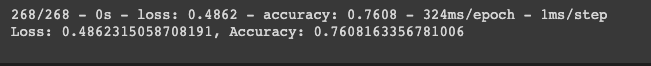


On my third attempt, I achieved the desired target - results are shown right before Optimization Steps section of this document.

I also did one more attempt trying to achieve a better performance. On my fourth attempt I removed the NAME from the dataset (It wasn’t use to determine unique values, and cutoff points to bin its rare categorical variables in a new value - ‘Other’). Then the number of unique values for each column was determined, and for CLASSIFICATION and APPLICATION\_TYPE the number of data points for each unique value was determined. This number of data points for each unique value was used to pick a cutoff point to bin ‘rare’ categorical variables together in a new value, ‘Other’.

This time three-layers were used with 100 epochs on the training regimen. My hypothesis was that increasing the number of layers, was going to improve to performance but to my dismay, this wasn’t the case. As the Accuracy of this model decrease from the best model, generating an accuracy of 76.08% with 1301 parameters.





**Summary**

The optimization efforts taken were successful, as I was able to achieve the target model performance. There are several other approaches that could have been taken to address this problem, but a classic approach to classification problems is to try fitting a logistic regression model to the data to predict the outcome. Deep Learning models should have multiple layers, since it is machine based, it ‘teaches’ a computer to filter inputs through the layers to learn how to predict and classify information. Although sometimes, as on my last attempt, having extra layers doesn’t necessary improve the performance of the model.