Deep Learning: Charity Funding Predictor

By: Stephen Jackson, 11/17/2022

Utilizing deep learning and neural networks, I examined charity data provided by AlphabetSoup to determine if applicants would be successfully funded given that 34,000 organization were previously successful.

# Data Processing

For the initial model, I removed the irrelevant data; this meant I dropped the “EIN” and “NAME” columns. The remaining columns and their data were considered features for the model. I will state that I did add “NAME” back in the second run and will explain my reasoning. For the “CLASSIFICATION” and “APPLICATION\_TYPE” columns, I counted the number of occurrences to allow for binning and better data model processing. In each case, I grouped rare occurrences into an “Other” category by setting a cut-off value and verified that the binning was successful.

The target variable for the model used the values from the “IS\_SUCCESSFUL” column where a value of “1” indicated a “yes” and “0” indicated a “no. I then used the “pd.get\_dummies()” function to encode any other categorical variables and applied StandardScaler to further transform the data to attempt to optimize the run and meet the targeted predictive accuracy of 75%. Once processed, I split the data into training and testing sets.

# Compiling, Training, and Evaluating the Model

For each model, I applied a multi-layered neural network with the first utilizing two. The number of features dictated the number of hidden nodes. The first model generated 8,191 parameters and an accuracy score of 73.1% when run 100 times. Screenshots of the first model results are below.

Text

Description automatically generated

Graphical user interface, application

Description automatically generated

Results of First Model:

A screenshot of a computer

Description automatically generated with medium confidence

# Optimization

Since the initial model failed to hit the target accuracy score, I added the “NAME” column back into the dataset and performed the same methods as in the first model for a second attempt. I also added a hidden layer and increased the nodes which resulted in 482,817 parameters. This time, the model produced an accuracy score of 79.55%. Screenshots of the second model results are below.

Text

Description automatically generated

Graphical user interface, application

Description automatically generated

Results of Second Model:

A screenshot of a computer

Description automatically generated with medium confidence