Module 21 – Deep Learning Challenge / Lori Andler

**Neural Network Model Report**

**Overview of the Analysis**

The nonprofit foundation Alphabet Soup wants a tool that can help it select the applicants for funding with the best chance of success in their ventures. With your knowledge of machine learning and neural networks, you’ll use the features in the provided dataset to create a binary classifier that can predict whether applicants will be successful if funded by Alphabet Soup.

From Alphabet Soup’s business team, you have received a CSV containing more than 34,000 organizations that have received funding from Alphabet Soup over the years. Within this dataset are a number of columns that capture metadata about each organization, such as:

* **EIN** and **NAME**—Identification columns
* **APPLICATION\_TYPE**—Alphabet Soup application type
* **AFFILIATION**—Affiliated sector of industry
* **CLASSIFICATION**—Government organization classification
* **USE\_CASE**—Use case for funding
* **ORGANIZATION**—Organization type
* **STATUS**—Active status
* **INCOME\_AMT**—Income classification
* **SPECIAL\_CONSIDERATIONS**—Special considerations for application
* **ASK\_AMT**—Funding amount requested
* **IS\_SUCCESSFUL**—Was the money used effectively

For this challenge, I used Python in google collab to create a tool to help the nonprofit foundation Alpahbet Soup can use to select the applicants for funding with the best chance of success in their ventures. A CSV containing more than 34,000 organizaitons that have received funding from AlphabetSoup over the years was provided.

**Results**

* **Data preprocessing**
  + Using knowledge of Pandas and scikit-learn's StandardScaler() was used to preprocess the data set.
  + - The target for the model was the "IS\_SUCCESSFUL" column.
  + - The columns "EIN" and "NAME" were dropped with the remaining feature columns remaining:
  + - APPLICATION\_TYPE
  + - AFFILIATION
  + - CLASSIIFICATION
  + - USE\_CASE
  + - ORGANIZATION
  + - STATUS
  + - INCOME\_AMT
  + - SPECIAL CONSIDERATIONS
  + - ASK\_AMT
* **Compiling Training and Evaluating the Model**
  + This **original model,** dropped "EIN" and "NAME" columns and determined the unique values in the remaining columns. Binning was used on the CLASSIFICATION value. The dataframe was converted to numeric with pd.get\_dummies. The preprocessed data was split in the X and y values. The StandardScaler() was used to fit and transform the model.

- Cutoff value for APPLICATION\_TYPE was set to 10

- Cutoff value for CLASSIFICATION was set to 1000

- First hidden layer used 10 units and activation of "relu"

- Second hidden layer used 5 units and activation of "relu"

- Output layer used activation of "sigmoid"

- epochs = 50

* + **Optimization #1**

Dropped "EIN" and "NAME" columns and determined the unique values in the remaining columns. Binning was used on the CLASSIFICATION value. The dataframe was converted to numeric with pd.get\_dummies. The preprocessed data was split in the X and y values. The StandardScaler() was used to fit and transform the model.

- Cutoff value for APPLICATION\_TYPE was set to 500

- Cutoff value for CLASSIFICATION was set to 700

- First hidden layer used 10 units and activation of "relu"

- Second hidden layer used 5 units and activation of "sigmoid"

- Added a third hidden layer using 3 units and activation of "sigmoid"

- Output layer used activation of "sigmoid"

- epochs = 100

* + **Optimization #2**

Dropped "EIN" and "NAME" columns and determined the unique values in the remaining columns. Binning was used on the CLASSIFICATION value. The dataframe was converted to numeric with pd.get\_dummies. The preprocessed data was split in the X and y values. The StandardScaler() was used to fit and transform the model.

- Cutoff value for APPLICATION\_TYPE was set to 1000

- Cutoff value for CLASSIFICATION was set to 1500

- First hidden layer used 10 units and activation of "leaky\_relu"

- Second hidden layer used 5 units and activation of "leaky\_relu"

- Added a third hidden layer using 3 units and activation of "sigmoid"

- Output layer used activation of "sigmoid"

- epochs = 125

* + **Optimization #3**

Dropped "EIN" and "NAME" columns and determined the unique values in the remaining columns. Binning was used on the CLASSIFICATION value. The dataframe was converted to numeric with pd.get\_dummies. The preprocessed data was split in the X and y values. The StandardScaler() was used to fit and transform the model.

- Also dropped "SPECIAL\_CONSIDERATIONS" and "ASK\_AMT" columns

- Cutoff value for APPLICATION\_TYPE remained at the original 10

- Cutoff value for CLASSIFICATION remained at the original 1000

- First hidden layer used 20 units and activation of "relu"

- Second hidden layer used 5 units and activation of "sigmoid"

- Added a third hidden layer using 3 units and activation of "sigmoid"

- Output layer used activation of "sigmoid"

- epochs = 100

**Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Loss** | **Accuracy** | **Accuracy difference from original** |
| Original Model | 0.5521 | 0.7271 | -- |
| Optimization #1 | 0.5531 | 0.7247 | -0.0024 |
| Optimization #2 | 0.5552 | 0.7247 | -0.0024 |
| Optimization #3 | 0.5525 | 0.7254 | -0.0017 |

Even with the optimization changes, the models decreased in accuracy. I was not able to attain the recommended 0.75 target on accuracy. The changes that were made had minimal impact and the model might need more data to make the training more precise.