Alphabet Soup Analysis

MODULE 21 CHALLENGE

RAE ANN GREGG

3.11.24

Alphabet Soup Overview

The purpose of this project is to create a tool for Alphabet Soup to help select applicants for funding with the best chance of success in their ventures.

The provided dataset was used to create a binary classifier using a machine learning program to predict whether applicants will be successful if funded by Alphabet Soup.

The provided dataset included more than 34,000 organizations that have received funding from Alphabet Soup over the years.

The following information was provided in the dataset for use in the program to predict probability of success with at least 75% accuracy:

- **EIN** and **NAME**—Applicants
- **APPLICATION_TYPE**—Alphabet Soup application type
- **AFFILIATION**—Affiliated sector of industry such as Independent or Company Sponsored
- **CLASSIFICATION**—Government organization classification
- **USE_CASE**—Use case for funding such as Product Development, Preservation, Healthcare, etc.
- **ORGANIZATION**—Organization type such as Association, Co-operative or Trust
- **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

Alphabet Soup Analysis Results

Data Preprocessing

- The focus of this model is to predict the probability of success for an applicant if they receive funding.
- The IS_SUCCESSFUL data was the target variable used for the model.
- The other columns were used as feature variables for the model except the EIN column. I originally also removed the NAME column as suggested in the instructions. However, the accuracy increased when I included the NAME column and binned the NAME data.
- The EIN column was removed since it was only introducing noise into the model and was not needed to predict the probability of success for funding.
- I used binning/bucketing with the APPLICATION_TYPE, CLASSIFICATION, and NAME columns to combine the rare occurrence categories to optimize the model.
- The categorical data was converted into numeric data using get_dummies.
- The data was split into training and testing data and scaled for a more balanced distribution.

Alphabet Soup Analysis Results

Compiling, Training, and Evaluating the Model

- I did 5 optimizations to get to the target of over 75% accuracy
  - **Original Mode**
    - Dropped EIN and Name Columns
    - 1st layer – 80 neurons with ReLU activation
    - 2nd layer – 30 neurons with ReLU activation
    - Output layer – 1 neuron with Sigmoid activation
    - 100 epochs
    - Accuracy – 51.43%
  - **Optimization 1** – added a hidden layer and increased the nodes
    - Dropped EIN and Name Columns
    - 1st layer – 80 neurons with ReLU activation
    - 2nd layer – 40 neurons with ReLU activation
    - 3rd layer – 20 neurons with ReLU activation
    - Output layer – 1 neuron with Sigmoid activation
    - 100 epochs
    - Accuracy – 72.92%

Alphabet Soup Analysis Results

Compiling, Training, and Evaluating the Model

- I did 5 optimizations to get to the target of over 75% accuracy
  - **Optimization 2** – increased the nodes and the epochs
    - Dropped EIN and Name Columns
    - 1st layer – 100 neurons with ReLU activation
    - 2nd layer – 50 neurons with ReLU activation
    - 3rd layer – 25 neurons with ReLU activation
    - Output layer – 1 neuron with Sigmoid activation
    - 150 epochs
    - Accuracy – 72.96%
  - **Optimization 3** – adjusted to fewer bins and reduced the epochs back to 100
    - Dropped EIN and Name Columns
    - Changed app count cut off from 500 to 50
    - Changed class count cut off from 1000 to 100
    - 1st layer – 100 neurons with ReLU activation
    - 2nd layer – 50 neurons with ReLU activation
    - 3rd layer – 25 neurons with ReLU activation
    - Output layer – 1 neuron with Sigmoid activation
    - 100 epochs
    - Accuracy – 72.91% - slightly less than Optimization 2

Alphabet Soup Analysis Results

Compiling, Training, and Evaluating the Model

- I did 5 optimizations to get to the target of over 75% accuracy
  - **Optimization 4** – created more bins
    - Dropped EIN and Name Columns
    - Changed app count cut-off from 500 to 1000
    - Changed class count cut-off from 1000 to 2000
    - 1st layer – 100 neurons with ReLU activation
    - 2nd layer – 50 neurons with ReLU activation
    - 3rd layer – 25 neurons with ReLU activation
    - Output layer – 1 neuron with Sigmoid activation
    - 100 epochs
    - Accuracy – 72.75% - slightly less than earlier Optimizations
  - **Optimization 5** – Did not drop the NAME column and binned the NAME data
    - Dropped Name Column
    - Changed app count cut-off back to 500
    - Changed class count cut-off back to 1000
    - 1st layer – 100 neurons with ReLU activation
    - 2nd layer – 50 neurons with ReLU activation
    - 3rd layer – 25 neurons with ReLU activation
    - Output layer – 1 neuron with Sigmoid activation
    - 100 epochs
    - Accuracy – 78.83% - achieved over the target accuracy of 75%

Alphabet Soup Analysis Results Summary

Overall, by using machine learning, we can predict with almost 79% accuracy which campaigns will be successful. This will help us fund campaigns more efficiently.

It would be ideal to continue to try other models to improve the accuracy even more.

The greatest gains in accuracy were gained from adding layers and neurons and including the NAME data. Continuing to adjust the layers, neurons and binning could further increase the prediction accuracy.

The Tanh model could also be used to try to increase the accuracy since it will normalize the data.