Hannah Kollmann

Neural Network Model Analysis

Overview

The purpose of this analysis was to create a binary classifier that can predict whether applicants will be successful if funded by the nonprofit foundation, Alphabet Soup. The model utilizes a dataset of over 34,000 organizations that have received funding from Alphabet Soup, containing metadata about each organization.

Results

Data Preprocessing

- **Target variable(s) for the model:** The target variable for the model was "IS_SUCCESSFUL" with 'TRUE' result being the desired outcome.
- **Feature variable(s) for the model:** The featured variables for the model include "APPLICATION_TYPE," "AFFILIATION," "CLASSIFICATION," "USE_CASE," "ORGANIZATION," "STATUS," "INCOME_AMT," "SPECIAL_CONSIDERATIONS," AND "ASK_AMT."
- **Variable(s) removed from the input data:** "EIN" and "NAME" columns were originally removed from the input data as they are identification columns and not useful as features or targets. However, "NAME" was brought back in the last model.

Compiling, Training, and Evaluating the Model

- **Neurons, layers, and activation functions selected for the neural network model and why:** In this project, I ran approximately 5 different models using different combinations to compile, train, and evaluate the model (Resource: AlphabetSoupCharity file). These models removed the EIN and NAME columns and with applying different neurons and layers and binning achieved an accuracy of 73%.

  Figure 1: AlphabetSoupCharity Model

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 42)	2,100
dense_1 (Dense)	(None, 21)	903
dense_2 (Dense)	(None, 1)	22

Total params: 3,025 (11.82 KB)
Trainable params: 3,025 (11.82 KB)
Non-trainable params: 0 (0.00 B)

The original model (see Figure 1) consists of three hidden layers with 42, 21, and 1 neurons (respectively) and ReLU activation functions. The output layer uses a sigmoid activation function for binary classification. This structure was chosen to provide a balance between complexity and the potential for overfitting, while maintaining the

ability to learn complex patterns in the data. As stated before, this model did not achieve the desired accuracy of 75%. It achieved just over 73% (see Figure 2).

Figure 2: AlphabetSoupCharity Results

```
268/268 - 0s - 2ms/step - accuracy: 0.7319 - loss: 0.5734
Loss: 0.5733826756477356, Accuracy: 0.7318950295448303
```

Next, I created a different model which only removed the EIN column (Resource: AlphabetSoupCharity_Optimization file). The model did great, achieving a performance of nearly 80% accuracy!

Figure 3: AlphabetSoupCharity_Optimization

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 42)	19,026
dense_1 (Dense)	(None, 21)	903
dense_2 (Dense)	(None, 1)	22

Total params: 19,951 (77.93 KB)
Trainable params: 19,951 (77.93 KB)
Non-trainable params: 0 (0.00 B)

Figure 4: AlphabetSoupCharity_Optimization Results

```
268/268 - 1s - 2ms/step - accuracy: 0.7963 - loss: 0.4739
Loss: 0.4738933742046356, Accuracy: 0.7962682247161865
```

The featured variables for this model did not change from the first model. Only the "EIN" variable was removed. The model consists of three hidden layers with 14, 7, and 1 neurons, respectively, The output layer uses a sigmoid activation function for binary classification and used ReLU for other layers. Multiple attempts were also made in this model to optimize it, however, I ended up keeping most things (e.g., adjusting input data) the same.

- **Steps taken in attempts to increase model performance:** To increase model performance, the following steps were taken:
  - Dropping additional irrelevant columns from the input data.
  - Adding or removing hidden layers
  - Using different activation functions for the hidden layers.
  - Increasing or decreasing the number of epochs in the training regimen.

Summary

Hannah Kollmann

The deep learning model achieved the desired performance of 79.63% accuracy in predicting the success of Alphabet Soup-funded organizations. Several attempts were made to optimize the model which is what ultimately lead to this improved performance. Another recommendation is a Random Forest Classifier, which employs a decision tree to classify samples. It could provide better accuracy.