Report on the Neural Network Model for Alphabet Soup

Overview of the Analysis

The primary goal of this analysis was to develop an effective deep learning model using neural networks for Alphabet Soup. My objective was to predict which applicants are likely to succeed if funded. By doing so, I aimed to make more informed decisions about allocating resources, ultimately improving the organization's success rate and overall impact.

Results

Data Preprocessing

- Target Variable: I have identified the target variable for my model as 'IS_SUCCESSFUL,' which signifies whether applicants were successful if funded (1 for successful, 0 for not successful).

- Features: The features used in my model encompass a variety of columns from the input data, including 'APPLICATION_TYPE,' 'AFFILIATION,' 'CLASSIFICATION,' 'USE_CASE,' 'ORGANIZATION,' 'STATUS,' 'INCOME_AMT,' and 'SPECIAL_CONSIDERATIONS.'

- Variables Removed: During the data preprocessing stage, I excluded 'EIN' and 'NAME' columns from the input data. These columns were not relevant as neither targets nor features for my model.

Compiling, Training, and Evaluating the Model

- Neurons and Layers:

- Notebook 1: In my initial approach, I designed a neural network with three hidden layers, each containing a different number of neurons. The first hidden layer had 9 neurons, followed by the second layer with 5 neurons, and the third layer with 2 neurons. The output layer consisted of a single neuron with a sigmoid activation function.

- Notebook 2: In my second model, I opted for a neural network with two hidden layers, featuring 8 neurons in the first layer and 5 neurons in the second layer. The output layer retained one neuron with a sigmoid activation function.

- Notebook 3: My third model introduced more complexity with four hidden layers. The first layer contained 9 neurons, followed by the second layer with 5 neurons, the third layer with 3 neurons, and the fourth layer with 2 neurons. The output layer consisted of one neuron with a sigmoid activation function.

Number of Epochs:

- Notebook 1: The initial model underwent training for 100 epochs.

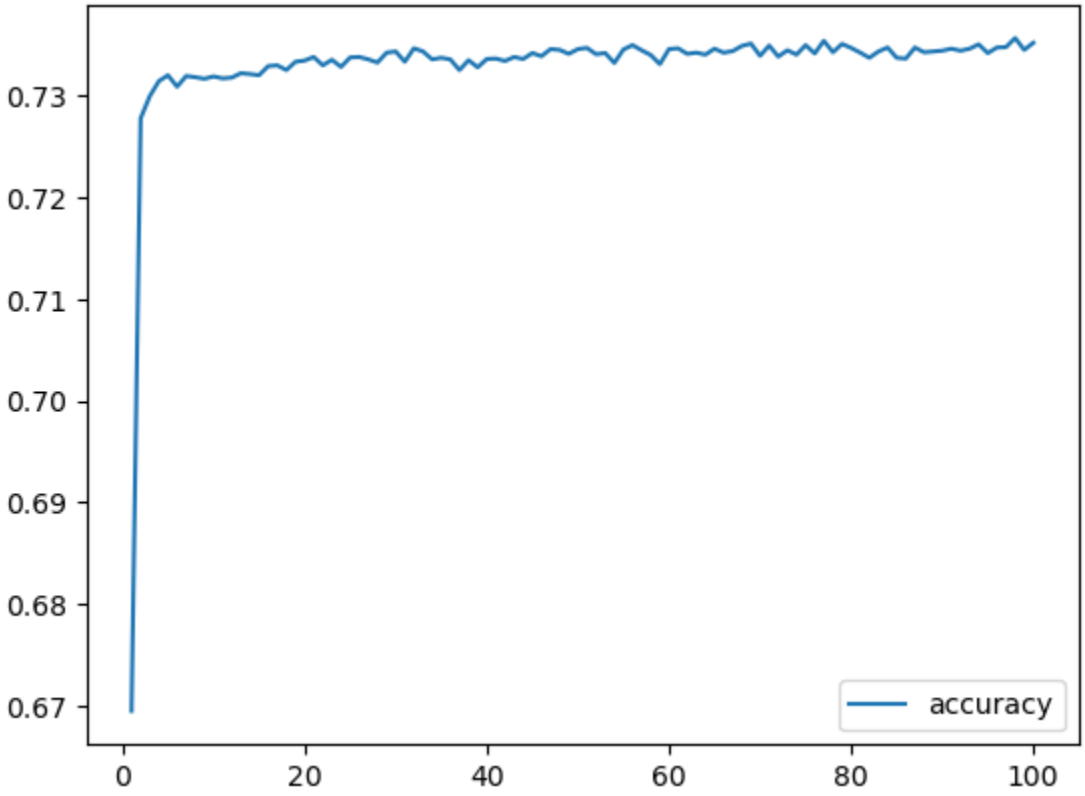
- Notebook 2: My second model was trained for 70 epochs.

- Notebook 3: The third model was trained over 50 epochs.

Model Performance:

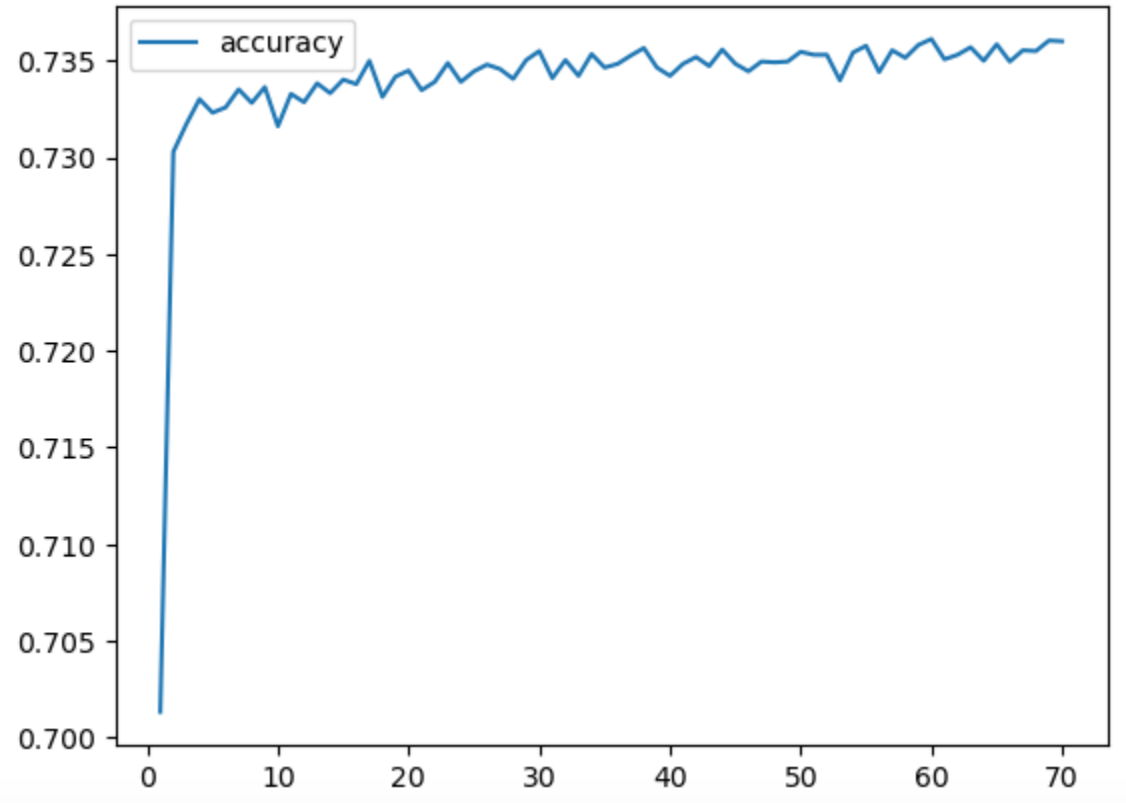
- Notebook 1: The first model achieved an accuracy of approximately 72.64%, although it fell short of my target

model performance.



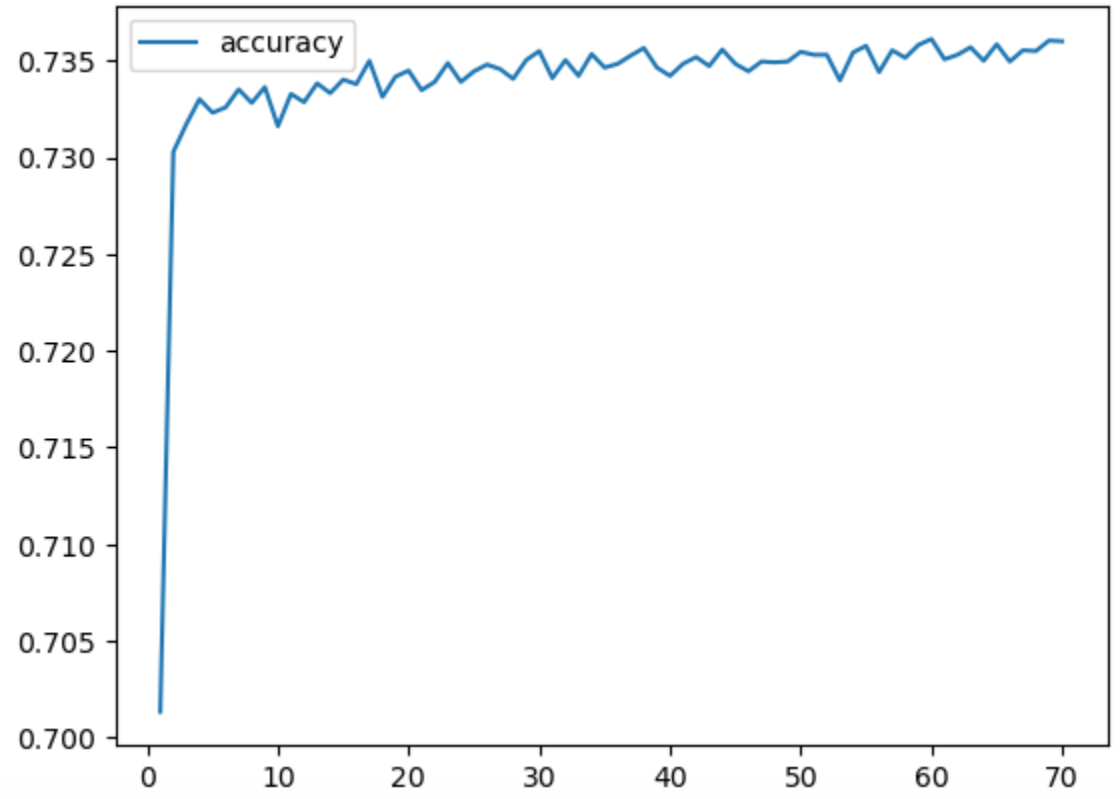
- Notebook 2: The second model yielded an accuracy of around 72.45%, which also did not reach the desired target

model performance.



- Notebook 3: The third model attained an accuracy of roughly 72.80%, yet it still did not meet the target model

performance.



Steps Taken to Improve Performance

Despite multiple attempts with different neural network configurations, such as the number of layers, neurons, and activation functions, I was unable to reach my target model performance. Potential next steps to boost model performance include:

- Fine-tuning hyperparameters, such as learning rates and batch sizes.

- Exploring various activation functions and layer combinations.

- Extending the number of training epochs.

- Investigating more advanced deep learning techniques like dropout and batch normalization.

Summary

In summary, my three deep learning models exhibited similar accuracy levels, around 72-73%. Unfortunately, these accuracies did not meet my target performance. While I experimented with various configurations, performance seemed to plateau in this range. A possible alternative approach is to explore ensemble models, such as Random Forest or Gradient Boosting, which frequently deliver superior results for classification tasks. This change may enhance the accuracy and predictive capabilities of

the model when coupled with the existing data and features. Further refinement and tuning are necessary to discover the most effective model for this task.