Overview

The “Module_21_Deep-Learning-Challenge” repository contains the code and files for creating a binary classification model to predict if an Alphabet Soup-funded organisation will be successful based on the features in the dataset. The dataset was provided as a CSV file, which contains information about more than 34,000 organization’s that have received funding from the fictional foundation, along with several columns of metadata about each organisation.

Results

<u>First Attempt</u>

In the first attempt, we used a model called "Resources/AlphabetSoupCharity.h5," and it had an accuracy score of 72.83%. This means that around 72.83% of the model's predictions matched the actual values in the dataset.

Cutoff Values:

- We made some choices to simplify the data and help the model perform better:
- "Application Type" Column: We grouped together rare categories in the "Application Type" column. If a category had fewer than 500 entries, we combined it with other similar categories under a new label called "Other." This was done to make the data less complex and avoid overfitting.
- "CLASSIFICATION" Column: Similar to the "Application Type" column, we combined rare categories in the "CLASSIFICATION" column. If a category had fewer than 100 entries, we grouped it with other similar categories under a new label called "Other." This was necessary because the "CLASSIFICATION" column had many different categories, and this grouping helped improve the model's performance.

Neural Network Architecture:

- The neural network model was designed with these settings:
- Layers: The model had 2 hidden layers. More hidden layers allow the model to learn intricate patterns from the data, which could help it perform better.
- Layer 1: The first hidden layer had 80 neurons and used the 'relu' activation function. This activation function efficiently deals with certain training issues and helps the model learn faster.
- Layer 2: The second hidden layer had 30 neurons and used the 'sigmoid' activation function. This function is commonly used in problems where we need to classify things into two groups, like in our case.

Epochs:

- We trained the model for 100 epochs. This means the model went through the dataset 100 times during training. Choosing 100 epochs balanced the time needed for the model to learn without overfitting. If we had trained for too few epochs, the model might not have learned enough, and if we had trained for too many epochs, the model might have memorized the data and not performed well on new, unseen data.

Although the first attempt showed promise, we know there's room for improvement.

Second Attempt

In the second attempt, we used a model called ”Resources/AlphabetSoupCharity_Optimisation.h5," and it had a slightly better accuracy score of 72.91% compared to the first attempt's score of 72.83%. This means that around 72.91% of the model's predictions matched the actual values in the dataset.

Changes Made:

1. Neural Network Architecture: In the second attempt, we made some adjustments to the neural network's architecture to see if it could further enhance the model's performance.
   a. Layer 1: We reduced the number of neurons in the first hidden layer to 14 and kept the 'relu' activation function. This change was made to simplify the model and potentially prevent overfitting.
   b. Layer 2: The second hidden layer now had 7 neurons instead of 30, and we continued to use the 'sigmoid' activation function. This decision was taken based on experimentation and the nature of our binary classification problem.

2. Evaluation: After implementing the changes, we reevaluated the model's performance, and it showed a slight improvement in accuracy compared to the first attempt.

Reasons for the Changes:

1. Neural Network Architecture: We reduced the number of neurons in both hidden layers to simplify the model. This was done to avoid overfitting, which can occur when the model becomes too complex, by reducing the number of neurons, the model becomes less prone to overfitting and may improve its ability to perform on unseen data.

Despite the slight improvement in accuracy in the second attempt, we recognise that there is still potential for further enhancement.

Third Attempt

In the third attempt, we used a model called "Resources/AlphabetSoupCharity_Optimisationv2.h5," and it showed a significant improvement in accuracy, reaching 78.71%. This means that around 78.71% of the model's predictions matched the actual values in the dataset, which is better than our desired accuracy of 75% or higher.

Changes Made:

1. Restored "Name" Column:
   In the third attempt, we added back the "Name" column to the dataset. Surprisingly, this simple change played a role in improving the model's accuracy.
2. Neural Network Architecture:
   We made adjustments to the neural network's structure to improve its performance:
   - Layers: We added a third hidden layer to the model. This helps the model learn more complex patterns from the data, potentially making it perform better.
   - Layer 1: The first hidden layer now had 64 neurons and used the 'relu' activation function, which helps the model learn faster and handle certain training challenges.
   - Layer 2: The second hidden layer had 32 neurons and used the 'relu' activation function.
   - Layer 3: The third hidden layer had 16 neurons and used the 'relu' activation function as well.

Reasons for the Changes:

1. Restored "Name" Column:
   Adding back the "Name" column contributed to the improved accuracy. Although it might not directly relate to the model's predictions, having this column in the dataset seemed to have a positive impact.

2. Neural Network Architecture:
   By adding a third hidden layer and adjusting the number of neurons in each layer, we aimed to strike a balance between model complexity and performance. More layers and neurons help the model learn intricate relationships within the data. We stuck with the 'relu' activation function as it effectively addresses training issues and speeds up learning.

Overall, the combination of restoring the "Name" column and fine-tuning the neural network's architecture resulted in a significant accuracy improvement in the third attempt. With an accuracy of 78.71%, we have surpassed our desired target of 75%.

Summary

In the first attempt, the model achieved an accuracy score of 72.83%, with efforts to simplify the data and prevent overfitting through cutoff values and neural network settings. The second attempt saw a slight improvement in accuracy to 72.91%, with adjustments to the neural network architecture. However, it was the third attempt that showed significant progress, reaching an accuracy of 78.71%. The success was attributed to restoring the "Name" column and fine-tuning the neural network by adding a third hidden layer with specific neuron configurations. These changes resulted in a model that surpassed the desired accuracy target of 75%, showcasing its improved performance.