**Challenge 21**

**1. Overview**

Using a Google Colab notebook, Tensorflow library, and the charity data CSV dataset, I built a machine learning model to predict whether or not the grants provided by the nonprofit foundation Alphabet Soup will be successful.

**2**. **Results**

* **During Data Preprocessing**:

What variable(s) are the target(s) for your model?

The target variable for my model is IS\_SUCCESSFUL.

What variable(s) are the features for your model?

status, ask\_amt, is\_successful, affiliation, classification, application\_type, use\_case, organization, income\_amt, special\_considerations.

What variable(s) should be removed from the input data because they are neither targets nor features? EIN & NAME since they are individual identifiers that are neither targets nor features.

1. Dropped columns EIN and Name.

A screenshot of a computer

Description automatically generated

2. Identified the categorical columns as APPLICATION\_TYPE and CLASSIFICATION.

A screenshot of a computer code

Description automatically generated

3. Next, I pulled unique values to determine if I had to use bucketing. Both Application Type and Classification categories have more than 10 values, so I used bucketing to reduce the number of values in each categorical column to less than 10 and place the rest under the group “other.”

4. Used get.dummies to encode categorical columns.

5. Split the preprocessed data into training and test data.

6. Scale the model. Take X and transform it to predict y, where X is the input that the neuron will transform into y, the output or prediction.

* **Throughout compile, train and evaluate the model**:

1. How many neurons, layers, and activation functions did you select for your neural network model, and why?

The first model (nn\_model1) with two hidden layers, 32 and 64 neurons, respectively, two activation functions, ReLU and sigmoid, and 50 epochs seemed reasonable given the dataset contains 44 input features. ReLU to explore non-linearity and sigmoid because this is a binary classification model.

Saved and exported the results to an HDF5 file and named it AlphabetSoupCharity.h5.

2. Were you able to achieve the target model’s performance?

I achieved an accuracy score of 72.48%, which is far from the target predictive accuracy of at least 75%.

**Optimize the model**:

3. What steps did you take in your attempts to increase model performance?

**Adding hidden layer**:

To increase model performance, I tried to optimize the first model with a second model (nn\_model2), adding an additional layer for a total of 3 layers with numbers of neurons 32, 64, and 64. I also added a learning rate of 0.001. This second model’s loss went down to 55.36%, and model accuracy went up to 72.54%, a little higher than the first model.

**Adding hidden layer and increasing epochs**:

This is the best model but still lower than the target predictive accuracy of at least 75%. I tried to optimize the second model with a third model (nn\_model3) with four hidden layers (one additional layer) and the number of neurons 64, 32, 16, and 8. I kept the learning rate at 0.001 and increased the number of epochs to 100. This model’s loss went up to 55.48%, and model accuracy went up to 72.64%, higher than the first and second models.

A screenshot of a computer program

Description automatically generated A screenshot of a computer

Description automatically generated

**Increasing the number of neurons and increasing epochs**:

For my final model, the fourth model(nn\_model4), I tried optimizing the third model by increasing the number of neurons to 128, 256, and 512 on the second, third, and fourth hidden layers. I used the same number of hidden layers, 4, with a learning rate of 0.001, and increased the number of epochs to 150. This model’s loss increased to 56.51%, and model accuracy decreased to 72.47%.

Saved and exported the results to an HDF5 file and named it AlphabetSoupCharity\_Optimization.h5.

**3. Summary:**

My first neural network model (nn\_model1) was 72.48% accurate in predicting successful funding projects**. The accuracy of my model increased when the number of hidden layers at nn\_model2 was increased.**The best model (nn\_model3) resulted from increasing layers to 4 and epochs to 100.

Increasing the number of layers improved the models’ performance, as proved by nn\_model2 and nn\_model3 increased accuracies. The increase in the number of neurons for each model may or may not lead to an improvement in accuracy. In this case, increasing the number of neurons and epochs did not lead to an improved accuracy in nn\_model4.

I would suggest a Random Forest Classifier because it can be used to determine feature importance, giving insights into what features are most important for this problem. Also, consider looking at other metrics like precision, recall, and F1 score; accuracy alone may not always be the best metric to evaluate model performance.

Source

Xpert. (2024). AI Learning Assistant. EdX.