Deep Learning Assignment: Module 21 Charity Funding

In order to predict if applicants would be funded by Alphabet Soup, deep learning and neural networks were implemented.

Initially, the dataset removed irrelevant information which was automatically determined to be “NAME” and “EIN”, using the remaining columns for the model. “NAME” was returned in a later test as the accuracy of the model proved to be below the desired 75% without it. “CLASSIFICATION” and “APPLICATION\_TYPE” values were replaced with ‘Other’ due to high fluctuation. Data was split into training and testing sets of data. With the target variable being “IS\_SUCCESSFUL”, it was verified by the value of 1 meaning yes and 0 meaning no. Analyzed “APPLICATION” data and binned “CLASSIFICATION” data.

Neural Network was applied on each model in multiple (3) layers. Thus dictating the number of hidden nodes.

# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.

number\_input\_features = len(X\_train\_scaled[0])

hidden\_nodes\_layer1=7

hidden\_nodes\_layer2=14

hidden\_nodes\_layer3=21

nn = tf.keras.models.Sequential()

# First hidden layer

nn.add(tf.keras.layers.Dense(units = hidden\_nodes\_layer1, input\_dim = number\_input\_features, activation = 'relu'))

# Second hidden layer

nn.add(tf.keras.layers.Dense(units = hidden\_nodes\_layer2, activation = 'relu'))

# Output layer

nn.add(tf.keras.layers.Dense(units = 1, activation = 'sigmoid'))

A three-layer training model generated 477 parameters and using 100 epochs gave an accuracy of 72.97%. After adjusting the epochs and random state it did raise the accuracy to 73.15% but it was still under the desired 75%.

Model: "sequential\_3"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

dense\_9 (Dense) (None, 7) 350

dense\_10 (Dense) (None, 14) 112

dense\_11 (Dense) (None, 1) 15

=================================================================

Total params: 477

Trainable params: 477

Non-trainable params: 0

# Evaluate the model using the test data

model\_loss, model\_accuracy = nn.evaluate(X\_test\_scaled,y\_test,verbose=2)

print(f"Loss: {model\_loss}, Accuracy: {model\_accuracy}")

268/268 - 0s - loss: 0.5486 - accuracy: 0.7297 - 221ms/epoch - 823us/step

Loss: 0.5485827922821045, Accuracy: 0.72967928647995

Upon readding “NAME” into the dataset, the 75% accuracy was obtained with a 78.93% with 3,298 parameters.

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

dense (Dense) (None, 7) 3171

dense\_1 (Dense) (None, 14) 112

dense\_2 (Dense) (None, 1) 15

=================================================================

Total params: 3,298

Trainable params: 3,298

Non-trainable params: 0

# Evaluate the model using the test data

model\_loss, model\_accuracy = nn.evaluate(X\_test\_scaled,y\_test,verbose=2)

print(f"Loss: {model\_loss}, Accuracy: {model\_accuracy}")

268/268 - 0s - loss: 0.4638 - accuracy: 0.7893 - 320ms/epoch - 1ms/step

Loss: 0.46382376551628113, Accuracy: 0.7892711162567139