Luke Fuller
Ga Tech Data Science & Analytics
Week 21

Creating a Charity Fund Predictor using Deep Learning

For Week 21, we were tasked with using Deep Learning with Neural Networks to determine whether applicants would obtain success if funded by Alphabet Soup, an Investment Group that focuses on charities.

In the Data Processing Phase, we were given a dataset and asked to remove irrelevant data points from our model specifically EIN and NAME fields. The remaining columns became features for our data model. Contradictory to the instructions the NAME field was added back in the second test to achieve 77% accuracy. CLASSIFICATION and APPLICATION_TYPE was replaced with 'Other due to high volatility and influence. The data was then split into training and testing sets. The target variable for the model is "IS_SUCCESSFUL" and is verified by the value, a binary flag of 1(Yes) and 0(No) was used. CLASSIFICATION's value was used for binning. Each unique value used several data point as a cutoff point to bin "rare" categorical variables together in a new value, 'Other'. Afterwards checked to see if binning was successful. Categorical variables were decided by pd.get_dummies ().

Compiling, Training, and Evaluating was phase two in our process. A Neural Network was applied on each model in three layers. The number of features dictated the number of hidden nodes.

```
# First hidden layer
nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer1, input_dim=number_in

# 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'))
```

After an unsuccessful 73% accuracy score, we decided to add the NAME field back into the dataset to increase the total amount of parameters considered.

```
[21] # 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: 0.5493 - accuracy: 0.7310 - 317ms/epoch - 1ms/step
     Loss: 0.5493265986442566, Accuracy: 0.7309620976448059
```

```
[ ]  # 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: 0.4621 - accuracy: 0.7896 - 370ms/epoch - 1ms/step
     Loss: 0.462077260017395, Accuracy: 0.7896209955215454
```

These types of Machine Deep Learning models have multiple layer to help teach a computer to filter input layers to learn prediction and classification