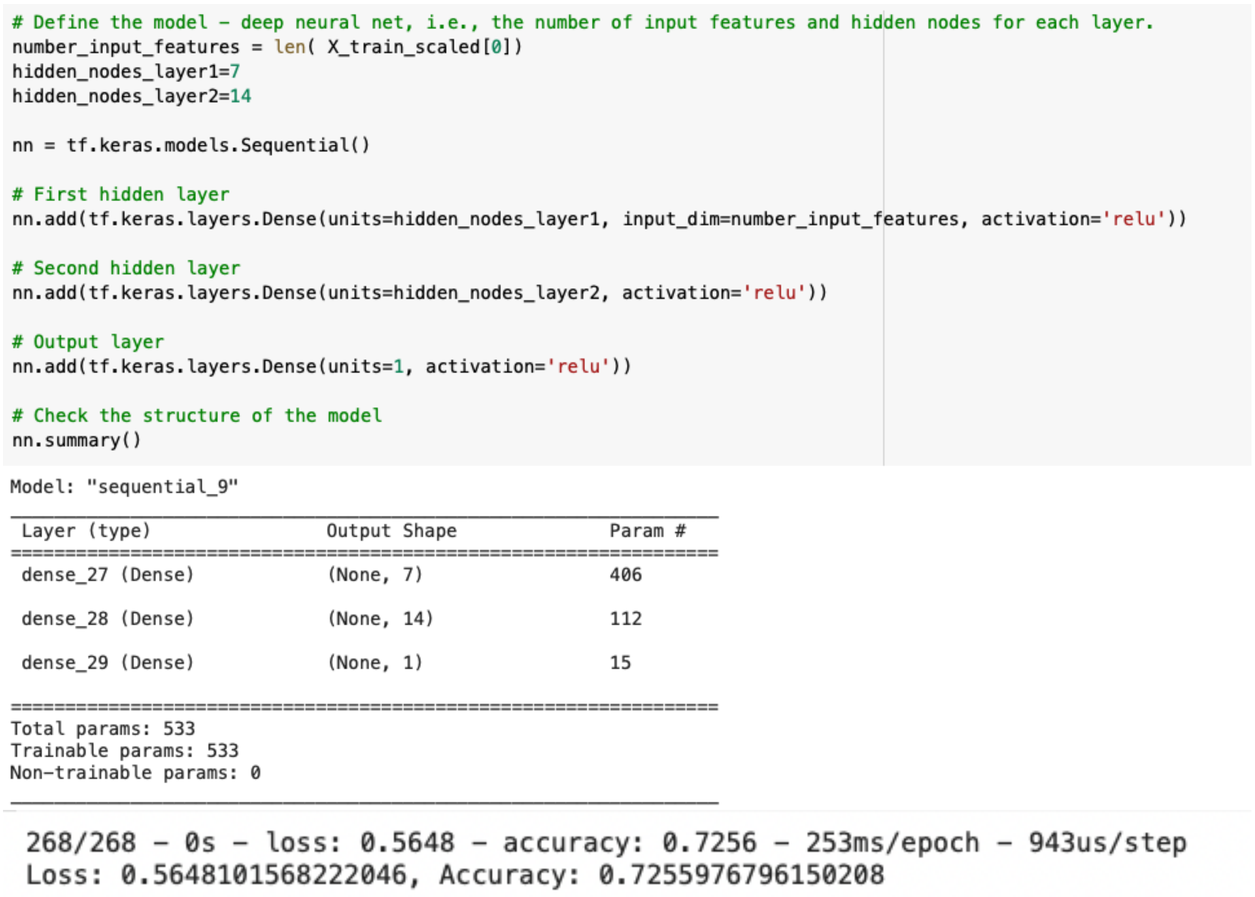
Deep Learning Challenge Analysis

Overview

The nonprofit foundation Alphabet Soup wants a tool that can help it select the applicants for funding with the best chance of success in their ventures. With the knowledge of machine learning and neural networks, we use the features in the provided dataset to create a binary classifier that can predict whether applicants will be successful if funded by Alphabet Soup.

Results

There were three layers for the model after applying Neural Networks with two hidden layers. 533 parameters were created by a three-layer training model. The first attempt was 72.56% which was under the desired 75%.



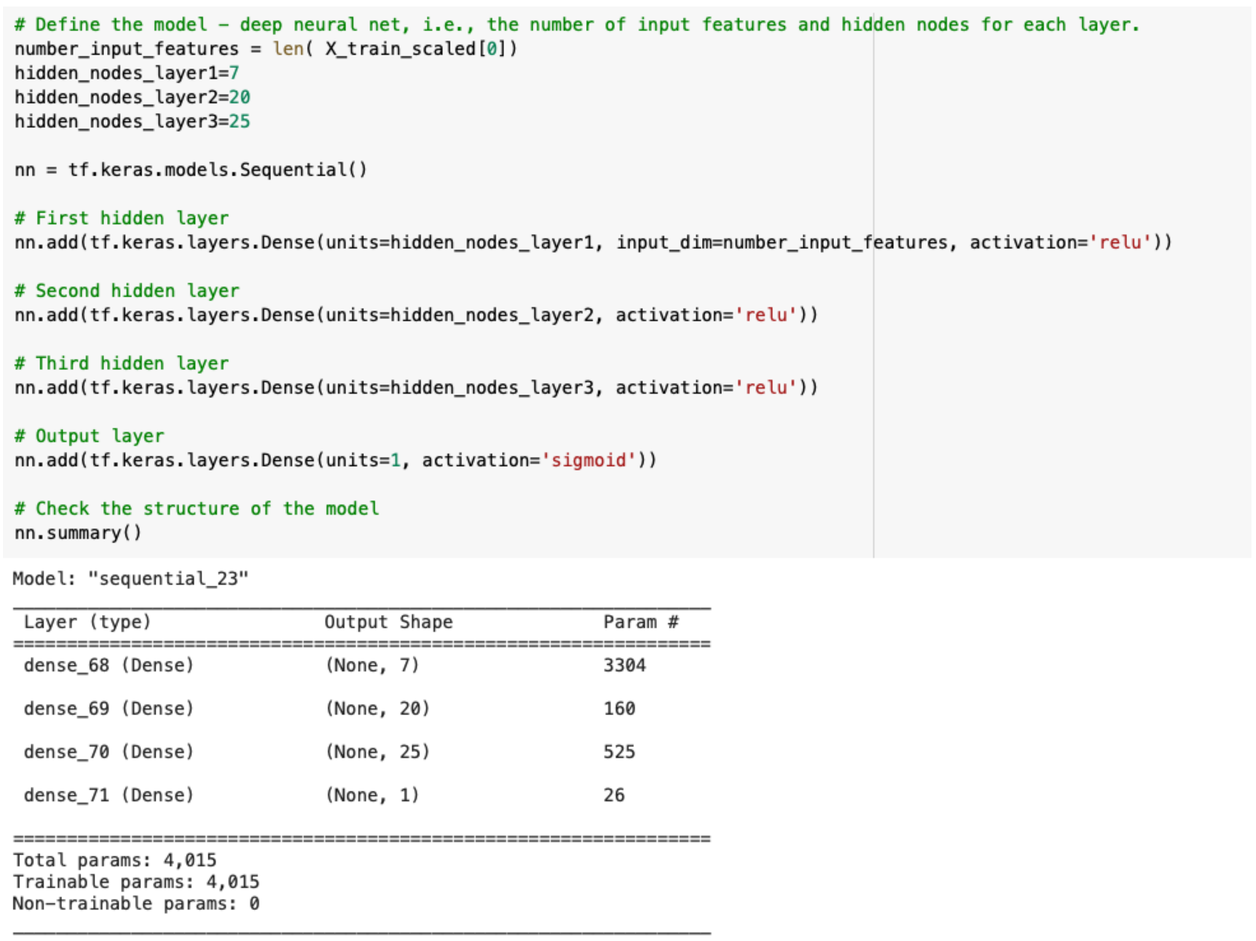
```
# Define the model - deep neural net, i.e., the number of input features and hidden nodes for each layer.
number_input_features = len( X_train_scaled[0])
hidden_nodes_layer1=7
hidden_nodes_layer2=14

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='relu'))

# Check the structure of the model
nn.summary()
```

```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_27 (Dense)            (None, 7)                 406

 dense_28 (Dense)            (None, 14)                112

 dense_29 (Dense)            (None, 1)                 15

=================================================================
Total params: 533
Trainable params: 533
Non-trainable params: 0
_________________________________________________________________
```

```
268/268 - 0s - loss: 0.5648 - accuracy: 0.7256 - 253ms/epoch - 943us/step
Loss: 0.5648101568222046, Accuracy: 0.7255976796150208
```

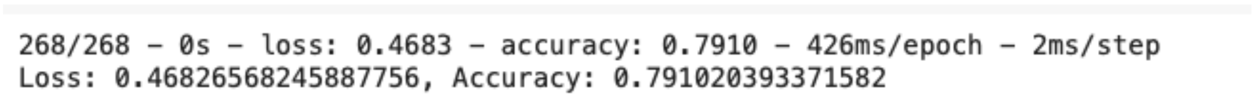
Optimization

The second attempt with “Name” column added, increased the number of values for each bin and added one more hidden layers, which leads to the accuracy 79.10% with 4015 parameters.



```python
# 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=20
hidden_nodes_layer3=25

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'))

# Third hidden layer
nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer3, activation='relu'))

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

# Check the structure of the model
nn.summary()
```

```
Model: "sequential_23"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_68 (Dense)            (None, 7)                 3304

 dense_69 (Dense)            (None, 20)                160

 dense_70 (Dense)            (None, 25)                525

 dense_71 (Dense)            (None, 1)                 26

=================================================================
Total params: 4,015
Trainable params: 4,015
Non-trainable params: 0
_________________________________________________________________
```



```
268/268 - 0s - loss: 0.4683 - accuracy: 0.7910 - 426ms/epoch - 2ms/step
Loss: 0.46826568245887756, Accuracy: 0.791020393371582
```