Alphabet Soup Report

Purpose

This project helps the non-profit company Alphabet soup determine which of the loan applications are most likely to succeed at their ventures. With these results, Alphabet soup can be sure that its efforts have the biggest impact.

Results

Variables

IS_SUCCESSFUL—Was the money used effectively
The following parameters are the targets for the model:

The following parameters are the features of the model:

- **APPLICATION_TYPE**—Alphabet Soup application type
- **AFFILIATION**—Affiliated sector of industry
- **CLASSIFICATION**—Government organization classification
- **USE_CASE**—Use case for funding
- **ORGANIZATION**—Organization type
- **STATUS**—Active status
- **INCOME_AMT**—Income classification
- **SPECIAL_CONSIDERATIONS**—Special considerations for application
- **ASK_AMT**—Funding amount requested
-

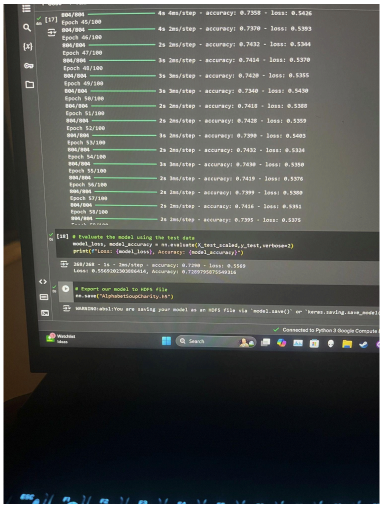
The following parameters were removed because they are neither targets nor features
EIN and **NAME**—Identification columns

Training

In creating the optimized model I decided to go with the following parameters
- 5 hidden layers (1 with 100 neurons, 3 with 50 neurons, and 1 with 30 neurons)
    - Tanh activation functions throughout

I decided to go with this structure because I assumed that more hidden layers, and more neurons per layer would lead to more accurate models. I picked the tanh activation function before and have worked with sigmoid and relu before, so I wanted to try out the tanh function.

My assumptions about how to build a better model were incorrect because the optimized model had an accuracy of 78.5% while the original model had an accuracy of 72.9%. This means the optimized model performed worse than the original.



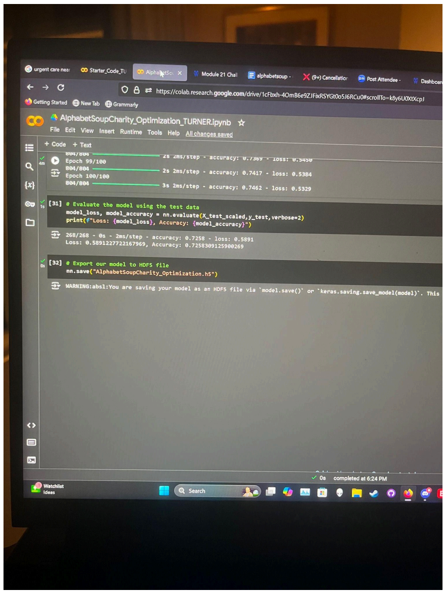
```
[17]
804/804 ━━━━━━━━━━━━━━━━━━━━ 4s 4ms/step - accuracy: 0.7358 - loss: 0.5426
Epoch 45/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 4s 2ms/step - accuracy: 0.7370 - loss: 0.5393
Epoch 46/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7432 - loss: 0.5344
Epoch 47/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.7414 - loss: 0.5370
Epoch 48/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 3s 3ms/step - accuracy: 0.7420 - loss: 0.5355
Epoch 49/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 3s 3ms/step - accuracy: 0.7340 - loss: 0.5430
Epoch 50/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7418 - loss: 0.5388
Epoch 51/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7428 - loss: 0.5359
Epoch 52/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.7390 - loss: 0.5403
Epoch 53/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7432 - loss: 0.5324
Epoch 54/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 3s 3ms/step - accuracy: 0.7430 - loss: 0.5350
Epoch 55/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.7419 - loss: 0.5376
Epoch 56/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7399 - loss: 0.5380
Epoch 57/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7416 - loss: 0.5351
Epoch 58/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7395 - loss: 0.5375

[18] # 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 - 1s - 2ms/step - accuracy: 0.7290 - loss: 0.5569
Loss: 0.5569202303886414, Accuracy: 0.7289795875549316

     # Export our model to HDF5 file
     nn.save("AlphabetSoupCharity.h5")

WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model
```



```
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7369 - loss: 0.5450
Epoch 99/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7417 - loss: 0.5384
Epoch 100/100
804/804 ━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.7462 - loss: 0.5329

[31] # 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 - 2ms/step - accuracy: 0.7258 - loss: 0.5891
Loss: 0.5891227722167969, Accuracy: 0.7258309125900269

[32] # Export our model to HDF5 file
     nn.save("AlphabetSoupCharity_Optimization.h5")

WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This
```

Alterations

I also tried universal sigmoid activation functions throughout the model (with the same number of layers and neurons), but the accuracy was significantly worse, so the details are not reported.

Summary

The potential for neural networks to help in this situation is evident. With enough tuning, Alphabet Soup can be confident that their loans will impact the world. However, I would recommend further refinements to the network before rolling it out to the field. Since neither model met the accuracy threshold of 75%, Alphabet Soup may find the current model ineffective. Therefore, I would wait until we clear the accuracy threshold before using the program.