Deep-Learning-Challenge

Overview:

The nonprofit Foundation Alphabet Soup wanted a tool to help them select applicants for funding. They needed binary classifier to predict whether applicants will be successful if funded by Alphabet Soup.

Data Preprocessing:

Unnecessary metrics such as EIN and Name were removed form the dataset and all remaining metrics were considered in the model. Both Classification and Application type were features for the model.

. What variables are the targets for your model?
. What variables are the features for your model?
. What variables should be removed from the input data because they are neither targets nor features?

Compiling, Training, and Evaluating the Model:

ATTEMPT 1

The first attempt (Model/AlphabetSoupCharity1.h5) resulted in an accuracy score of 72.8%. This was the highest lowest accuracy score of the three models. This means that 72.8% of the model's predicted values align with the dataset's true values.

The hyper parameters used were:

layers = 3
layer1 = 10 neurons : activation function = 'relu'
layer2 = 8 neurons : activation function = 'sigmoid'
layers3 = 6 neurons : activation function = 'sigmoid'
epochs = 30

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 10)	440
dense_1 (Dense)	(None, 8)	88
dense_2 (Dense)	(None, 6)	54
dense_3 (Dense)	(None, 1)	7

===
Total params: 589
Trainable params: 589
Non-trainable params: 0
_____

268/268 - 1s - loss: 0.5530 - accuracy: 0.7279 - 697ms/epoch - 3ms/step
Loss: 0.5529545545578003, Accuracy: 0.7279300093650818

OPTIMIZATION - MODEL

ATTEMPT 1

The first attempt (Models/AlphabetSoupCharity_Optimization) resulted in an accuracy score of 73.9%. This means that 73.9% of the model's predicted values align with the dataset's true values.

The hyper parameters used were:

layers = 2
layer1 = 80 neurons : activation function = 'relu'
layer2 = 80 neurons : activation function = 'relu'
epochs = 100

Model: "sequential"
_____

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 80)	9120
dense_1 (Dense)	(None, 80)	6480
dense_2 (Dense)	(None, 1)	81

===
Total params: 15,681
Trainable params: 15,681
Non-trainable params: 0

215/215 - 1s - loss: 0.5626 - accuracy: 0.7388 - 869ms/epoch - 4ms/step
Loss: 0.562642514705658, Accuracy: 0.7387754917144775

ATTEMPT 2

The first attempt (Models/AlphabetSoupCharity_Optimization) resulted in an accuracy score of 73.94%. This was the highest accuracy score of the two models. This means that 73.94% of the model's predicted values align with the dataset's true values.

The hyperparameters used were:

layers = 3
layer1 = 80 neurons : activation function = 'relu'
layer2 = 30 neurons : activation function = 'relu'
layers3 = 20 neurons : activation function = 'relu'
epochs = 75

Model: "sequential_1"
_____

Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 80)	9120
dense_4 (Dense)	(None, 30)	2430
dense_5 (Dense)	(None, 20)	620
dense_6 (Dense)	(None, 1)	21

Total params: 12,191
Trainable params: 12,191
Non-trainable params: 0

215/215 - 1s - loss: 0.5615 - accuracy: 0.7394 - 1s/epoch - 5ms/step
Loss: 0.5614603161811829, Accuracy: 0.7393586039543152

ATTEMPT 3 - Utilizing an Automated Optimizer (such as a hyper-parameter tuner):

Automated optimizers, like hyper-parameter tuners, systematically explore various combinations of hyper-parameters, such as activation functions, number of layers, number of neurons, and epochs. This exploration can help identify the most optimal combination of hyper parameters for your specific problem, potentially leading to higher accuracy.

```
# Import the keras-tuner library
import keras-tuner as kt

tuner = kt.Hyperband(
```

```
    create_model,
    objective="val_accuracy",
    max_epochs=20,
    hyperband_iterations=2)

    Trial 60 Complete [00h 01m 24s]
val_accuracy: 0.7374635338783264

Best val_accuracy So Far: 0.7409620881080627
Total elapsed time: 00h 27m 32s
```