DEEP-LEARNING-CHALLENGE

Ibsar Hashmi

MODULE 21 EDEX UOFT BOOTCAMP

Table of Contents

Overview of the Analysis 2

Results 2

Data Preprocessing 2

Target and Features 2

Compiling, Training, and Evaluating the Model 3

Neural Network Architecture 3

Model Performance 5

Attempts to Increase Model Performance 5

Summary 5

Recommendation 5

Overview of the Analysis

The purpose of this analysis was to develop a deep learning model for Alphabet Soup to predict the success of funding applicants based on various features. The goal was to create an accurate and efficient model that could assist in making informed decisions about funding allocations to various organizations.

Results

Data Preprocessing

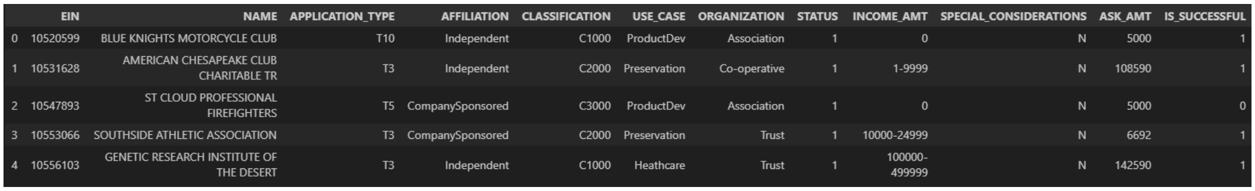
Target and Features

Target Variable(s): The target variable for the model is "IS_SUCCESSFUL".

Feature Variable(s): The features used for the model include "CLASSIFICATION", "APPLICATION_TYPE", "AFFILIATION", "USE_CASE", "ORGANIZATION", "STATUS", "INCOME_AMT", "SPECIAL_CONSIDERATIONS", and "ASK_AMT."

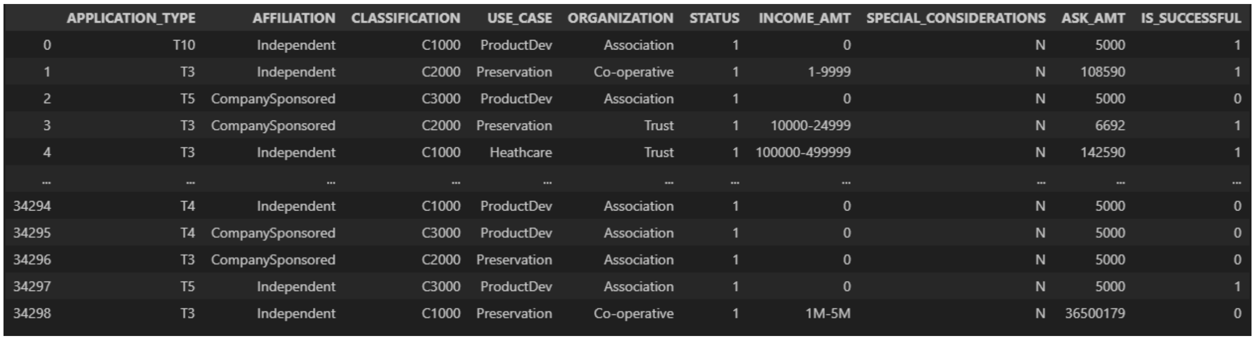
Variables Removed: "EIN" and "NAME" variables were removed.

Initial DataFrame:



	EIN	NAME	APPLICATION_TYPE	AFFILIATION	CLASSIFICATION	USE_CASE	ORGANIZATION	STATUS	INCOME_AMT	SPECIAL_CONSIDERATIONS	ASK_AMT	IS_SUCCESSFUL
0	10520599	BLUE KNIGHTS MOTORCYCLE CLUB	T10	Independent	C1000	ProductDev	Association	1	0	N	5000	1
1	10531628	AMERICAN CHESAPEAKE CLUB CHARITABLE TR	T3	Independent	C2000	Preservation	Co-operative	1	1-9999	N	108590	1
2	10547893	ST CLOUD PROFESSIONAL FIREFIGHTERS	T5	CompanySponsored	C3000	ProductDev	Association	1	0	N	5000	0
3	10553066	SOUTHSIDE ATHLETIC ASSOCIATION	T3	CompanySponsored	C2000	Preservation	Trust	1	10000-24999	N	6692	1
4	10556103	GENETIC RESEARCH INSTITUTE OF THE DESERT	T3	Independent	C1000	Heathcare	Trust	1	100000-499999	N	142590	1

Final DataFrame:



	APPLICATION_TYPE	AFFILIATION	CLASSIFICATION	USE_CASE	ORGANIZATION	STATUS	INCOME_AMT	SPECIAL_CONSIDERATIONS	ASK_AMT	IS_SUCCESSFUL
0	T10	Independent	C1000	ProductDev	Association	1	0	N	5000	1
1	T3	Independent	C2000	Preservation	Co-operative	1	1-9999	N	108590	1
2	T5	CompanySponsored	C3000	ProductDev	Association	1	0	N	5000	0
3	T3	CompanySponsored	C2000	Preservation	Trust	1	10000-24999	N	6692	1
4	T3	Independent	C1000	Heathcare	Trust	1	100000-499999	N	142590	1
...	...	...	...	...	...	...	...	...	...	...
34294	T4	Independent	C1000	ProductDev	Association	1	0	N	5000	0
34295	T4	CompanySponsored	C3000	ProductDev	Association	1	0	N	5000	0
34296	T3	CompanySponsored	C2000	Preservation	Association	1	0	N	5000	0
34297	T5	Independent	C3000	ProductDev	Association	1	0	N	5000	1
34298	T3	Independent	C1000	Preservation	Co-operative	1	1M-5M	N	36500179	0

Compiling, Training, and Evaluating the Model

Neural Network Architecture

Neurons, Layers, and Activation Functions:

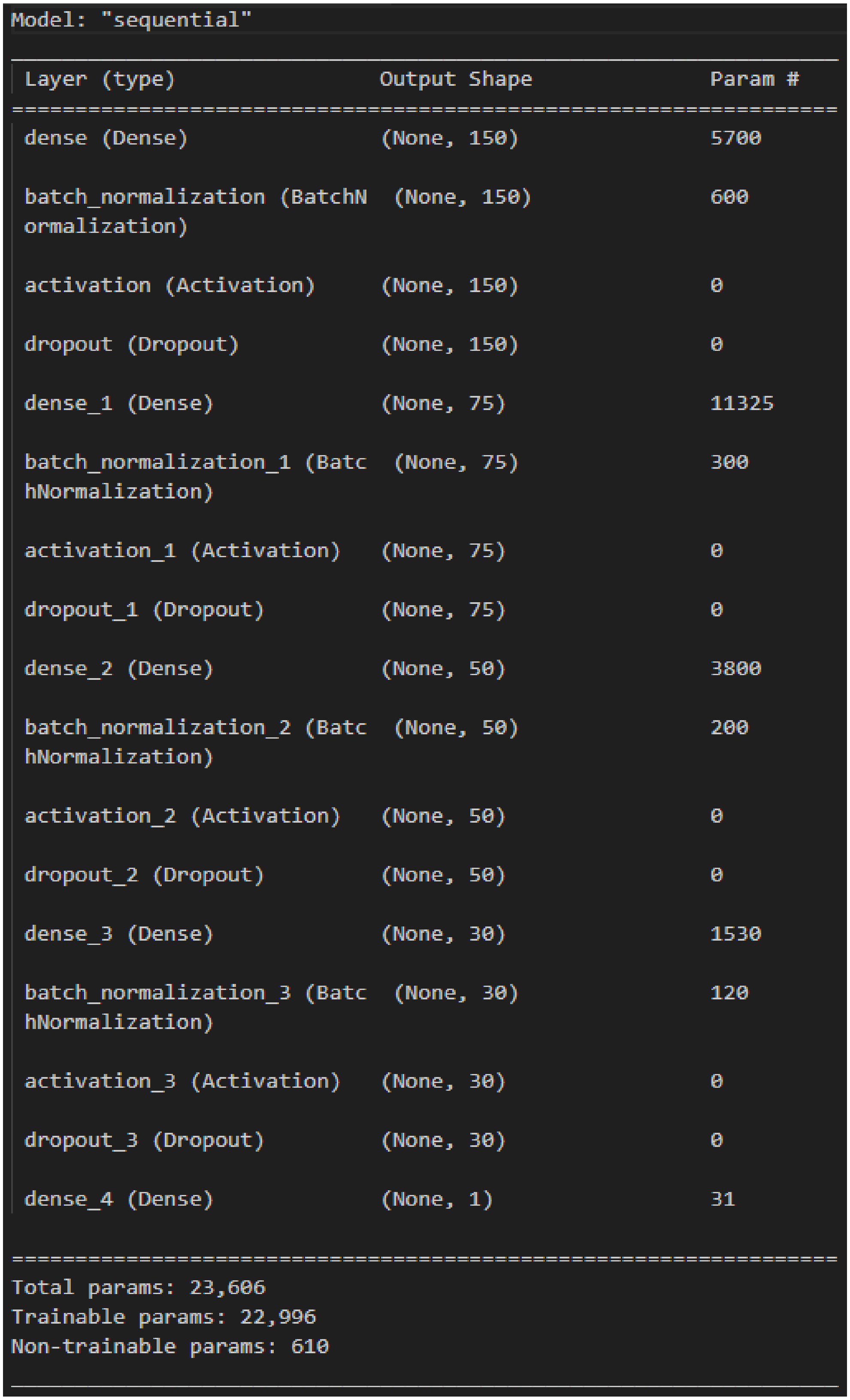
Number of neurons in each layer: 4+1 layers, with 150, 75, 50, 30, and 1 neuron/s respectively. The final layer is the output layer.

Number of layers: 4 layers plus output layer.

Activation functions: activation functions used inclusive of output layer include 'relu', 'relu', 'relu', 'relu', and 'sigmoid'. Other functions tried included ‘tanh’ and ‘leaky_relu.’

Explanation: the increase and adjustment of layers was based on trial and error to achieve the best accuracy with minimal loss. Model accuracy remained unchanged, however, even against adding two additional neural network layers into the model and trying different activation methods while accounting for overfitting by including a dropout rate, and Batch Normalization. Learning rate of the optimizer was adjusted to 0.001 and to 0.01 as well.

The following is the model build:



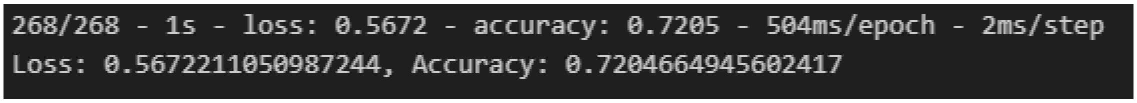
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 dense (Dense)               (None, 150)               5700      
                                                                 
 batch_normalization (BatchN  (None, 150)              600       
 ormalization)                                                   
                                                                 
 activation (Activation)     (None, 150)               0         
                                                                 
 dropout (Dropout)           (None, 150)               0         
                                                                 
 dense_1 (Dense)             (None, 75)                11325     
                                                                 
 batch_normalization_1 (Batc  (None, 75)               300       
 hNormalization)                                                 
                                                                 
 activation_1 (Activation)   (None, 75)                0         
                                                                 
 dropout_1 (Dropout)         (None, 75)                0         
                                                                 
 dense_2 (Dense)             (None, 50)                3800      
                                                                 
 batch_normalization_2 (Batc  (None, 50)               200       
 hNormalization)                                                 
                                                                 
 activation_2 (Activation)   (None, 50)                0         
                                                                 
 dropout_2 (Dropout)         (None, 50)                0         
                                                                 
 dense_3 (Dense)             (None, 30)                1530      
                                                                 
 batch_normalization_3 (Batc  (None, 30)               120       
 hNormalization)                                                 
                                                                 
 activation_3 (Activation)   (None, 30)                0         
                                                                 
 dropout_3 (Dropout)         (None, 30)                0         
                                                                 
 dense_4 (Dense)             (None, 1)                 31        
                                                                 
=================================================================
Total params: 23,606
Trainable params: 22,996
Non-trainable params: 610
_________________________________________________________________
```

Model Performance

Achievement of Target Model Performance:

The model achieved an accuracy of approximately 72.05% and a loss of 0.5672 on the test dataset.

Time was 504ms/epoch with 2ms/step.



```
268/268 - 1s - loss: 0.5672 - accuracy: 0.7205 - 504ms/epoch - 2ms/step
Loss: 0.5672211050987244, Accuracy: 0.7204664945602417
```

Attempts to Increase Model Performance

Attempts to increase model performed included increasing the layers from 2 to 4, adjusting neurons per layer from an 80, 30, 1 configuration to a 150, 75, 50, 30, 1 configuration. This configuration included four hidden layer and a final output layer as opposed to two hidden layers followed by a final output layer. Batch Normalization was included in the optimization while different activations were tried including Tanh, Sigmoid, relu (current and final model), and leaky_relu. Dropout rates were included per layer to prevent overfitting. the optimizer 'Adam's learning rate was adjusted to 0.001, and back to 0.01 to see a change in accuracy and loss by the model. finally, epochs were adjusted between 50 to 150 (final being 150), and batch sizes were included with initial tests run on 128 batch sizes and final model kept at 64 batch sizes.

additionally, the data was preprocessed again and another category "Other" was included to remove any generalized classifications of entries and categorize them together under the "Other' classification. cutoff rates were tested at 1000 and then at 1500 for both classifications and application types to home in on improving the accuracy of the model.

Summary

In summary, the deep learning model developed for Alphabet Soup demonstrated moderate performance with an accuracy of 72.05%. The chosen architecture, including the number of neurons, layers, and activation functions, was based on trial and error. Despite achieving a satisfactory performance, there is room for improvement based on additional data preprocessing whereby the "IS_SUCCESSFUL" selection criteria could become more stringent based on scrutinizing other criteria/categories.

Recommendation

A different model, such as Gradient Boosting Machines model or a Neural Network with a different Architecture model might be better suited for this classification problem. This decision is based on the classification and data types available in the dataset which causes this case to become less of a binary decision-making problem and more of a complex and robust decision tree problem. Additionally, further exploration of the dataset and feature engineering could enhance the predictive power of the existing model better than 72.05% accuracy rate.