UOM DS Module 21 – Deep Learning Challenge

Robert Bentz

Robert Bentz

**OVERVIEW**

The purpose of this machine (deep) learning analysis is to develop a tool for successfully choosing the most likely to succeed candidates, from a pool of charitable funding requests. From numerous data points (inputs) try to predict success rate for the applicants using the tool should the applicants receive funding from Alphabet Soup.

**SUMMARY FINDINGS**

Overall it appears the accuracy of the machine learning models is quite similar, with the ‘tanh’ activation having only slightly better results. The accuracy results are changing at the third decimal place for the most part. Depending upon what the expectation is, using these models one can expect a few failures from the applicants. Safe, but not very safe, predictions in practice.

STATUS and SPECIAL\_CONSIDERATIONS are dropped in several models due to very low change in response. Only two response types with one response being well under 1%. A consideration to weight those marginal responses into a different model may yield increasing accuracy, but my instinct would say very low contribution regardless. Those two columns were dropped to decrease model complexity to reduce modeling time.

There is a great deal of data so an option to increase the accuracy may be to change the ratio of training to testing of the data. Investigate marginally increasing the training values and evaluate the change in testing accuracy.

Another idea to increase the model accuracy could be not more data, but an additional data category. A data stream that was not supplied but may be readily available – and most likely to improve accuracy – would be the number of employees and possibly locations. This would address the amount of resources an applicant has at its disposable and increase the likelihood of success. The many small hands do mighty works statement comes to mind. In many cases, if the employee count was not requested, the information is public for many types of organizations and can be added to the data set.

**SUMMARY TABLE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Activation** | **Epochs** | **Neurons** | **Results** |
| Original (nn2)  Dimension=43 | 1st layer – relu  2nd – relu  Output – sigmoid | 80 | 80  30  1 | Loss=0.5516  Accuracy=0.7272 |
| nn2 (new model)  Dimension=43 | 1st layer – relu  2nd – relu  Output – sigmoid | 50 | 40  15  1 | Loss=0.5525  Accuracy=0.7262 |
| nn3  Dimension=42 | 1st layer – relu  2nd – relu  Output – sigmoid | 100 | 80  30  1 | Loss=0.5561  Accuracy=0.7245 |
| nn5  Dimension=40 | 1st layer – relu  2nd – relu  Output – sigmoid | 100 | 100  40  1 | Loss=0.5575  Accuracy=0.7254 |
| nn8  Dimension=40 | 1st layer – tanh  2nd – tanh  Output – sigmoid | 100 | 80  30  1 | Loss=0.5526  Accuracy=0.7255 |
| nn12  Dimension=40 | 1st layer – relu  2nd – relu  3rd – relu  4th – tanh  Output – sigmoid | 150 | 80  40  20  10  1 | Loss=0.5615  Accuracy=0.7312 |
| Kerastuner  Dimension=52 | Automated  1st – tanh  2nd – tanh  3rd – tanh  4th – tanh  5th - tanh | 80 | 21  21  1  16  26  26 | Loss=0.5493  Accuracy=0.7308 |

All models were compiled with:

Loss = binary\_crossentropy

Optimizer = adam

Metrics = accuracy

Increasing the hidden layer densities did not appear to have much effect from the original file settings other than epoch=40. Epoch at 80 and 100 showed more effect than changing units in the hidden layers. The activation model showed slight variation for conditions tested.

See below for model specifics.

**FIRST MODEL (nn)**

*Activation relu*

The first model (nn) was set to parameters to obtain starter code results.

Using the first model as the baseline for other comparisons.

First model - sequential neural net, tensor flow, keras.

First hidden layer using 80 units, activation is 'relu' with input dimension at 43.

Second hidden layer using 30 units, activation is 'relu'.

Output layer has 1 unit, activation is set to 'sigmoid'.

Model compiled for loss='binary\_crossentropy', optimizer='adam', and metrics='accuracy'.

Model was fit using epochs=100.

Model loss at 0.5516.

Model accuracy at 0.7272.

Model saved as 'AlphabetSoupCharity.h5'.

**SECOND MODEL (nn2)**

*Activation relu*

The second model (nn2) was set with most parameters at half the first model.

Second model - sequential neural net, tensor flow, keras.

First hidden layer using 40 units, activation is 'relu' with input dimension at 43.

Second hidden layer using 15 units, activation is 'relu'.

Output layer has 1 unit, activation is set to 'sigmoid'.

Model compiled for loss='binary\_crossentropy', optimizer='adam', and metrics='accuracy'.

Model was fit using epochs=50.

Model loss at 0.5525.

Model accuracy at 0.7262.

Model saved as 'AlphabetSoupCharity2.h5'.

**THIRD MODEL (nn3)**

*Activation relu*

The third model (nn3) was set with same parameters as first model (nn).

The difference is the STATUS column was removed due to <1% difference in values (all 1's).

Third model - sequential neural net, tensor flow, keras.

First hidden layer using 80 units, activation is 'relu' with input dimension at 42.

Second hidden layer using 30 units, activation is 'relu'.

Output layer has 1 unit, activation is set to 'sigmoid'.

Model compiled for loss='binary\_crossentropy', optimizer='adam', and metrics='accuracy'.

Model was fit using epochs=100.

Model loss at 0.5561.

Model accuracy at 0.7245.

Model saved as 'AlphabetSoupCharity3.h5'.

**FIFTH MODEL (nn5)**

*Activation relu*

The fifth model (nn5) was set with same parameters as first model (nn).

The difference is the STATUS and SPECIAL\_CONSIDERATIONS(y/n) columns were removed due to <1% difference in values (all 1's).

Fifth model - sequential neural net, tensor flow, keras.

First hidden layer using 100 units, activation is 'relu' with input dimension at 40.

Second hidden layer using 40 units, activation is 'relu'.

Output layer has 1 unit, activation is set to 'sigmoid'.

Model compiled for loss='binary\_crossentropy', optimizer='adam', and metrics='accuracy'.

Model was fit using epochs=100.

Model loss at 0.5575.

Model accuracy at 0.7254.

Model saved as 'AlphabetSoupCharity5.h5'.

**EIGHTH MODEL (nn8)**

*Activation tanh*

An additional model activation was tried, "tanh". Seemed to achieve marginally better performance than, "relu". A plot was also generated which seemed to indicate better accuracy way down the road of diminishing returns.

The eighth model (nn8) was set with same parameters as first model (nn).

The difference is the STATUS and SPECIAL\_CONSIDERATIONS(y/n) columns were removed due to <1% difference in values (all 1's).

Eighth model - sequential neural net, tensor flow, keras.

First hidden layer using 80 units, activation is 'tanh' with input dimension at 40.

Second hidden layer using 40 units, activation is 'tanh'.

Output layer has 1 unit, activation is set to 'sigmoid'.

Model compiled for loss='binary\_crossentropy', optimizer='adam', and metrics='accuracy'.

Model was fit using epochs=100.

Model loss at 0.5526.

Model accuracy at 0.7255.

Model saved as 'AlphabetSoupCharity5.h5'.

**TWELTH MODEL (nn12)**

*Mixed activation (relu/tanh)*

The twelth model (nn12) was set with same parameters as first model (nn).

The difference is the STATUS and SPECIAL\_CONSIDERATIONS(y/n) columns were removed due to <1% difference in values (all 1's). More layers were added and more epochs to attempt a better accuracy. Used a mix of activations, tanh and relu.

Twelth model - sequential neural net, tensor flow, keras.

First hidden layer using 80 units, activation is 'relu' with input dimension at 40.

Second hidden layer using 40 units, activation is 'relu'.

Third hidden layer using 20 units, activation is 'relu'.

Fourth hidden layer using 10 units, activation is 'tanh'.

Output layer has 1 unit, activation is set to 'sigmoid'.

Model compiled for loss='binary\_crossentropy', optimizer='adam', and metrics='accuracy'.

Model was fit using epochs=150.

Model loss at 0.5615.

Model accuracy at 0.7312.

Model saved as 'AlphabetSoupCharity11.h5' (yes, 11).

**TUNER APPROACH**

*Automated activation*

An automated tuner approach was also tried. Based on Xara's class code (21.2.5).

The Optimization Tuner2 was set to use kerastuner to run through various combinations and compare result values.

The STATUS and SPECIAL\_CONSIDERATIONS(y/n) columns were removed due to <1% difference in values (all 1's).

The tuner was give the activation options of relu and tanh. Kerastuner would determine first layer of neurons to use (calling model nn\_model5). Min value is 1 and max is 30 with an input dimension of 40.

The tuner also decided how many hidden layers and neurons, with input layer range of 1-8, value range 1-40 with a step size of 2.

Model had a sigmoid activation for the final layer.

Model compiled for loss='binary\_crossentropy', optimizer='adam', and metrics='accuracy'.

Model was fit using epochs=100, hyperband iterations at 2.

Model loss at 0.5543.

Model accuracy at 0.7306.