**Report on the Neural Network Model**

**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. We have used machine learning and neural networks in the provided dataset to create a binary classifier that can predict whether applicants will be successful if funded by Alphabet Soup.

From Alphabet Soup’s business team, we have received a CSV containing more than 34,000 organizations that have received funding from Alphabet Soup over the years.

**Purpose**

The purpose of this challenge is to get at least 75% accuracy by in neural network model. To achieve this, we have to try different hidden layers, activation method and number of neurons.

**Data Preprocessing**

1. **Read the csv file and drop the EIN and NAME columns from the data frame.**
2. **Identify the number of unique values for each column.**
3. **APPLICATION\_TYPE and CLASSIFICATION column has more than 10 unique values.**
4. **Decide a cutoff point to bin "rare" categorical variables together in a new value, other and then check if the binning was successful.**
5. **Use get\_dummies() to encode the categorical values.**
6. **Split the preprocessed data into a features array, x and a target array y. Use these arrays and the train\_test\_split function to split the data into training and testing datasets.**
7. **Scale the training and testing features datasets by creating a standardscaler instance, fitting it to the training data, then using the transform function.**

### Compile, Train, and Evaluate the Model

1. **Created neural network model by assigning the number of input features and nodes for each layer using TensorFlow and Keras.**
2. **In the first trial created 2 hidden layers with the same activation(relu) for the hidden layers and different activation(sigmoid) for hidden layers.**

**hidden\_node\_layer1 = 8**

**hidden\_node\_layer2 = 5**

1. **For the first trial the loss is 55% and accuracy is 72.6%**
2. **Created new notebook for optimization and tried 4 more trials building the neural network model. Increased the hidden layers, tried different activation. But I could not achieve 75% accuracy. So, commented out the trials.**
3. **Tried whether Random Forest method will work for this data still I could achieve only 70% accuracy.**
4. **I will further work on the cut off value to get a 75% accuracy.**

**Results**

**Data Preprocessing**

* What variable(s) are the target(s) for your model?

IS\_SUCCESSFUL column is the target for the model.

* What variable(s) are the features for your model?

All columns other than IS\_SUCCESSFUL, EIN and Name are the features of the model.

* What variable(s) should be removed from the input data because they are neither targets nor features

EIN and Name column has been dropped from the input data, these are neither targets nor features.

Compiling, Training, and Evaluating the Model

* How many neurons, layers, and activation functions did you select for your neural network model, and why?

Trial 1

**hidden\_node\_layer1 = 8**

**hidden\_node\_layer2 = 5**

**activation\_ node\_layer1 = “relu”**

**activation\_ node\_layer2 = “relu”**

**activation\_ output = “sigmoid”**

**loss = 56%**

**Accuracy = 72%**

**In the first trial the accuracy is 72% and below is the plot showing the accuracy**

A graph with blue lines

Description automatically generated

Trial 2

hidden\_node\_layer1 = 10

hidden\_node\_layer2 = 7

hidden\_node\_layer3 = 5

activation\_node\_layer1 = “relu”

activation\_node\_layer2 = “tanh”

activation\_node\_layer3 = “relu”

activation\_ output = “sigmoid”

loss = 56%

Accuracy = 72%

Trial 3

hidden\_node\_layer1 = 15

hidden\_node\_layer2 = 10

hidden\_node\_layer3 = 5

hidden\_node\_layer4 = 3

hidden\_node\_layer5 = 2

activation\_node\_layer1 = “relu”

activation\_node\_layer2 = “relu”

activation\_node\_layer3 = “relu”

activation\_node\_layer4 = “relu”

activation\_node\_layer5 = “relu”

activation\_ output = “sigmoid”

loss = 56%

Accuracy = 72%

Trial 4

hidden\_node\_layer1 = 40

hidden\_node\_layer2 = 35

hidden\_node\_layer3 = 30

hidden\_node\_layer4 = 25

hidden\_node\_layer5 = 20

hidden\_node\_layer6 = 15

hidden\_node\_layer7 = 10

hidden\_node\_layer8 = 5

activation\_node\_layer1 = “relu”

activation\_node\_layer2 = “relu”

activation\_node\_layer3 = “relu”

activation\_node\_layer4 = “relu”

activation\_node\_layer5 = “relu”

activation\_node\_layer6 = “relu”

activation\_node\_layer7 = “relu”

activation\_node\_layer8 = “relu”

activation\_ output = “sigmoid”

loss = 56%

Accuracy = 72%

Trial 5

hidden\_node\_layer1 = 100

hidden\_node\_layer2 = 50

hidden\_node\_layer3 = 10

activation\_node\_layer1 = “relu”

activation\_node\_layer2 = “tanh”

activation\_node\_layer3 = “sigmoid”

activation\_ output = “sigmoid”

loss = 55%

Accuracy = 72%

Even after 5 trials there is no improvement in the accuracy **and below is the plot showing the accuracy for trial 5.**

A graph with a line

Description automatically generated

* Were you able to achieve the target model performance?

I was not able to achieve the target model performance of 75%. The current model shows only 72% accuracy.

* What steps did you take in your attempts to increase model performance?

In each trial tried to increase the hidden layers, change the activation method and increased the number of neurons for each hidden layer. Still only 72% accuracy in the current model.

**Summary**

**After many attempts I could not achieve the 75% accuracy, so tried to run it using Random Forest model, but it gave only 70% accuracy. As a next step planning to reduce the feature and increase the cut off value.**