IoT Analytics

Project 4 – Neural Networks

Name - Aditya Kadole

Student ID – 200321563

To determine the configuration of a feedforward neural network and its hyperparameters using a data set with five input variables and a single continuous output variable.

I have used two datasets

akadole1.csv is used for training neural network whereas akadole3.csv consists of only 300 test samples which is used for running regression model.

Task 1: Automatic grid search

The dataset consists of 2300 sample data points with 5 input variables X1, X2, X3, X4, X5 and an output variable Y.

The dataset was divided into 2000 samples as training dataset and the remaining 300 samples as testing data. We implemented it using Python using different frameworks and statistical packages. These data samples were normalized using MinMaxScaler().

We executed the training dataset by using the package GridSearchCV where we used different values for the number of epochs, batch size, number of nodes in the hidden layer and the learning rate.

Based on this search we obtained the best score values for different parameters as below

Epochs - 100

Batch size - 40

Learning rate – 0.001

Number of hidden nodes / Neurons – 16

```
'batch_size': 40, 'epochs': 100, 'learn_rate': 0.001, 'neurons': 16
```

For the above said parameters a part of console output is as below

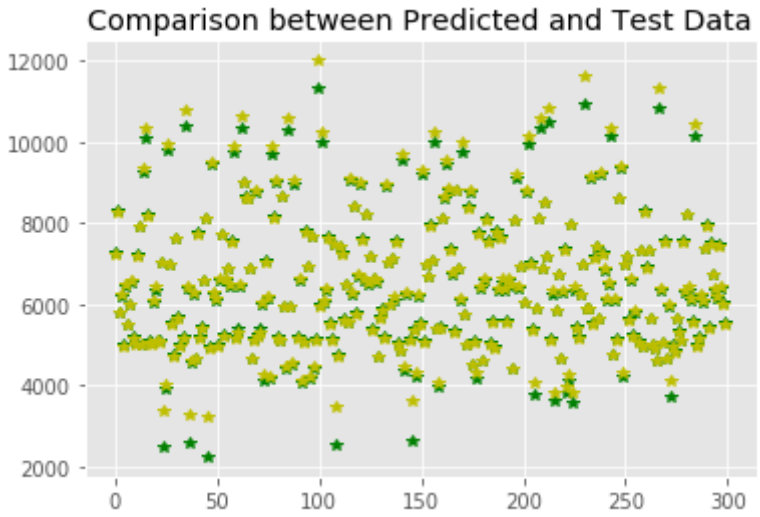
```
Epoch 94/100
2000/2000 [==============================] - 0s 25us/step - loss: 0.0056 - mae: 0.0056 - val_loss: 0.0062
- val_mae: 0.0062
Epoch 95/100
2000/2000 [==============================] - 0s 25us/step - loss: 0.0059 - mae: 0.0059 - val_loss: 0.0061
- val_mae: 0.0061
Epoch 96/100
2000/2000 [==============================] - 0s 27us/step - loss: 0.0061 - mae: 0.0061 - val_loss: 0.0061
- val_mae: 0.0061
Epoch 97/100
2000/2000 [==============================] - 0s 25us/step - loss: 0.0056 - mae: 0.0056 - val_loss: 0.0061
- val_mae: 0.0061
Epoch 98/100
2000/2000 [==============================] - 0s 25us/step - loss: 0.0056 - mae: 0.0056 - val_loss: 0.0060
- val_mae: 0.0060
Epoch 99/100
2000/2000 [==============================] - 0s 25us/step - loss: 0.0056 - mae: 0.0056 - val_loss: 0.0064
- val_mae: 0.0064
Epoch 100/100
2000/2000 [==============================] - 0s 26us/step - loss: 0.0057 - mae: 0.0057 - val_loss: 0.0061
- val_mae: 0.0061
```

We also calculated the Mean absolute error for the Test dataset

```
Mean absolute error of Test data: 73.25
```

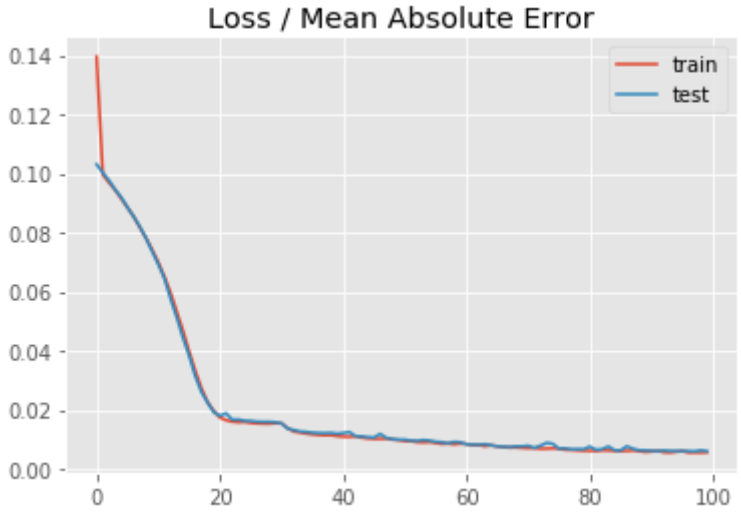
We got the SSE (Sum Squared Error) as **7180784.4886279795**

With the predicted test data values and the actual test data values we plotted the graph which is as shown below where Green indicated the predicted test data values and Yellow indicates the actual test data value.



From the graph we can see that predicted data samples are almost similar to the actual data samples. Hence, we can say that the neural network has trained good to get output similar to actual test data.

Then we plotted the graph between number of epochs and loss



From the graph plotted above we can say that as the number of epochs increases the loss of the data samples becomes constant and doesn't reach zero and also the neural network is trained good for the test data as the loss is minimized.

Task 2: Compare the trained neural network with multivariable regression

Before we go ahead with the comparison let us see the predicted output values of test data in multivariable regression and the actual test data values for 10 input values.

	Actual Test Data	Predicted Test Data
1	7214.2	7340.8
2	8286.2	8239.7
3	5786.2	5959.6
4	6226.6	6376.8
5	4953.7	5018.4
6	6507.0	6701.2
7	5479.6	5577.0
8	5999.7	6147.1
9	6593.2	6775.1
10	5136.3	5118.4

From the above table we can say that the predicted values of multivariable regression are close enough to the actual test data.

Now let us see the SSE (sum squared error) of multivariable regression on test data

15356694.279388031

	Neural Network	Multivariable Regression
Sum_Squared_Error	7180784.488	15356694.279

From the above values we can say that Neural network model has trained much better than the multivariable linear regression as the error is low comparatively. And regression has less parameters to estimate than the neural network for the same set of input variables. Neural networks usually require a large dataset for its optimization. We can also say that for non-linearity data samples neural network trains much better than regression model.

But sometimes we have to be careful with the neural networks model as training with large parameters may sometimes lead to overfitting.