**ASSIGNMENT 2 (MLP)**

**PROCESS MODELLING ASSIGNMENT :-**

**Loss function** : Mean Squared Error(MSE) is chosen to carry out the regression modeling. MSE take the difference between model’s predicted values and actual values, square it and then average it out across the whole dataset. It is applied to make sure that training model has no outlier predictions with huge error. As MSE square up the value it puts larger weight on such outlier values hence increasing the error value.

**Activation Function** : ReLu function is used for the activation of neurons output in both the hidden layers. It is non-linear in nature. Due to large no. of neurons are used in hidden layers makes the network dense. The advantage of using ReLu is, it will make the activation sparse and efficient by making some neurons dead by giving output as 0 for negative x values( actually making gradient to be zero) and its non bounded in nature.

A(x) = max(0, x) … if x > 0 otherwise 0.

**Kernel initializer** : The default kernel initializer is used in all the layers which is ‘glorot uniform’.

It helps to keep variance similar along all the layers. Xavier initialization ensure that weights are just right, for the signal to propagate through layers. It also helps in preserving the forward-propagated signal which is much more significant than backward propagation.

Pre-Processing Techniques Applied :

1. Libraries used :-

Tensorflow – used for training and creating large scale neural network.

Keras – it wraps efficient numerical computation for developing and evaluating neural network

Pandas – used for manipulation of data and analysis.

Numpy – a package for scientific computation.

Sklearn – this library contains efficient tools for statistical modelling.

1. Data read – Data used for the modelling is in ‘csv’ format and is read using pandas.
2. Removal of missing data- Nan values are removed from the data.
3. Scaling of data – A data columns named rate and uniformity are scaled by dividing/ 100 and multiplying by 10 respectively to improve the bias and variance.
4. Data splitting – data is splitted into 75% training data and .25 into testing set using sklearn library.

**Structured experiments :-**

Deep neural networks are hard to train. So a proper initialization is required to keep a check that it won’t become problematic while back propagating all the way till initial layers. In this process weights are assigned using glorot uniform function. A bad initialization can really hinder the learning of highly deep network. If we randomly initialize weights , the first layer will throw away the most of the information and a pinch of data will be left for subsequent layers for training. I trained neural network with constant initial weights (0 and 1). In both the cases no matter what inputs are fed to network if weights are same , all units in hidden layer will work in similar manner and choosing 0 as initial weight over 1 is even worse as it hinders the learning of model by great extent .

Hyper-parameters are important as they directly control the training algorithm and have direct performance on the model performance. The hyper-parameter for this model are tuned using keras tuner package. The model parameters are tuned using RandomSearch( ) with objective of focusing on ‘loss’ parameter i.e. MSE , the best value with the approach of minimizing it. The best model is chosen among all which has lower MSE value. Total of 5 different models are trained by tuning neurons , no of layers and learning rate. For a particular set of parameters model training is done once per trial execution. The hyper- parameter tuned for model are, no of hidden layer ranging from 2 to 10 layers a model, no of neurons ranging with min. value 32 to max. value 256 and. Learning rate to choose from values ( .01, .001, .0001). Below are the results after tuning the model:-

Text

Description automatically generated

Figure 2 : tuned parameters

Text

Description automatically generated with medium confidence

Figure 1 : hyper tuned model performance

The best model performance shows the MSE value of 2075.80

The model parameters after preforming random search tuning are as:

No of hidden layers : 10

No of neurons in each layer are shown in fig 2.

Learning rate= 0.01

Loss= 2075.80

There is no generalization regarding , whether the multi-output neural network is more efficient or a separate model for each output. So the basic rule which applies here is hit and trial. One has to check it itself which is works better. It has been shown empirically in research that a neural network might have some advantage with multiple outputs.

In this case a single output model is built to compare result with multi output model. I computed result on ‘ Uniformity ‘ output column( y2 ) by evaluating its MSE value and comparing to multi output model’s MSE value for the same output column (not with overall model score). The result for the above comparison is shown below.

Graphical user interface, text, application

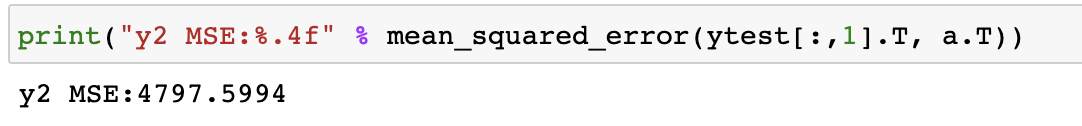
Description automatically generatedFigure 3 : single output model 

Figure 4 : MSE values after rescaling data

*As shown in fig. 3 and fig. 4 the MSE value for y2 output is 4797.59 in single output model and 2379.5 in multi output model. Hence , clearly the multi output layer model is performing better.*

**Generalized demonstrated in design network : -**

Graphical user interface, text

Description automatically generatedThe model performance is completely and effectively evaluated through various checkpoints. Initially model was trained by feeding the input data without normalization(rescaling) which resulted in higher accuracy but the mean square error was quite high. Later, two output features Rate and Uniformity are rescaled by a definitive values resulting in the better performance of the model. Rescaling of values impacted the accuracy by little but a lot of improvement is seen in MSE values as shown in fig 4. Overall model achieved accuracy of 90.2 for the training data.

Figure 5 : MSE values without rescaling data

By comparing values in Fig. 4 and fig. 5 its very evident the model outperforms after rescaling of values.

No technique such as regularization or cross validation has been applied. Bias variance relationship is well maintained as model is not showing any signs of overfitting or under fitting the data. Cross validation is avoided as by tuning the parameter improves the model performance which eventually refined the accuracy. The bias and variance relation of all outputs is shown below :-

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generatedChart, line chart

Description automatically generatedChart, line chart

Description automatically generated

A little bias is shown by model while making prediction in y4 output i.e. selectivity otherwise in rest three bias variance trade-off is well maintained.

**NON SEPARABLE CLASSIFICATION : MULTI FONT CHARACTER RECOGNITION**

**Loss Function :** The loss function used for this multi font classification is binary cross entropy. The advantage of using it is if an observation belongs to certain class , it should not influence the decision of another class. As clear by name binary class entropy loss it sets up binary classification between classes for every output.

**Activation Function** : ReLu function is used for the activation of neurons output in both the hidden layers. It is non-linear in nature. Due to large no. of neurons are used in hidden layers makes the network dense. The advantage of using ReLu is, it will make the activation sparse and efficient by making some neurons dead by giving output as 0 for negative x values( actually making gradient to be zero) and its non bounded in nature.

The other activation function used is ‘softmax’. It is used to squash the raw class scores into positive values that sum to 1, so that cross entropy can be applied on this

Pre-Processing Techniques Applied :

1. Libraries used :-

Tensorflow – used for training and creating large scale neural network.

Keras – it wraps efficient numerical computation for developing and evaluating neural network

Pandas – used for manipulation of data and analysis.

Numpy – a package for scientific computation.

Sklearn – this library contains efficient tools for statistical modelling.

1. Data read – Data used for the modelling is in ‘csv’ format and is read using pandas. For testing and training separate dataset is available.

A neural network is constructed using sklearn library of python. Network consist of two hidden layer with 256 and 64 neurons respectively in first and second layer. The activation function used for both the layers is ReLu. The output layer has 26 neuron output to which softmax activation is applied so that raw result is presented in a good presentable form. The sum of all the output neuron for particular prediction is 1. The fig 1. represent the model summary details.

Figure : model details

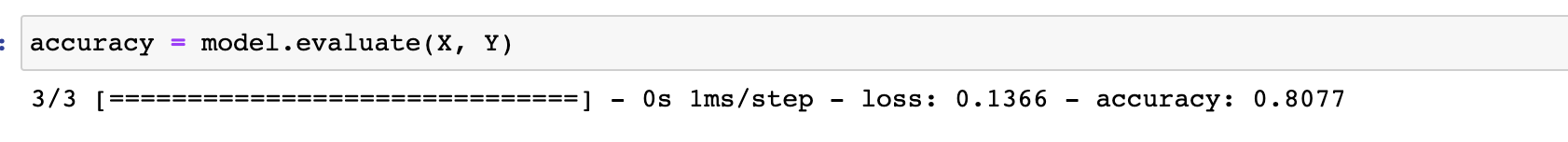
Table

Description automatically generated

The model is trained using training dataset provided using 100 epochs and batch size of 10.

the classification loss is .1366 i.e. binary crossentropy and accuracy on training data is .8077 as shown. below in fig 2.

Figure : model performance



The model performance is also evaluated for each epoch for which result is shown below.

the loss increases as the no of epochs increase for validation set, for training set it gradually decreases with increasing epochs which shows the model is trained perfectly.

The accuracy of the training data almost close to 1 which is a good sign for modelling the model.both results are shown below in fig3.

Figure : model performance for each epoch

Chart, scatter chart

Description automatically generated

Now , the model is tested on the test dataset which is showing a loss of 0.165 and accuracy of 0.693. Graphical user interface, text, application, email

Description automatically generated

There was no class imbalance in the data set so nothing is required to sort it out. and no regularization technique has been applied to it.