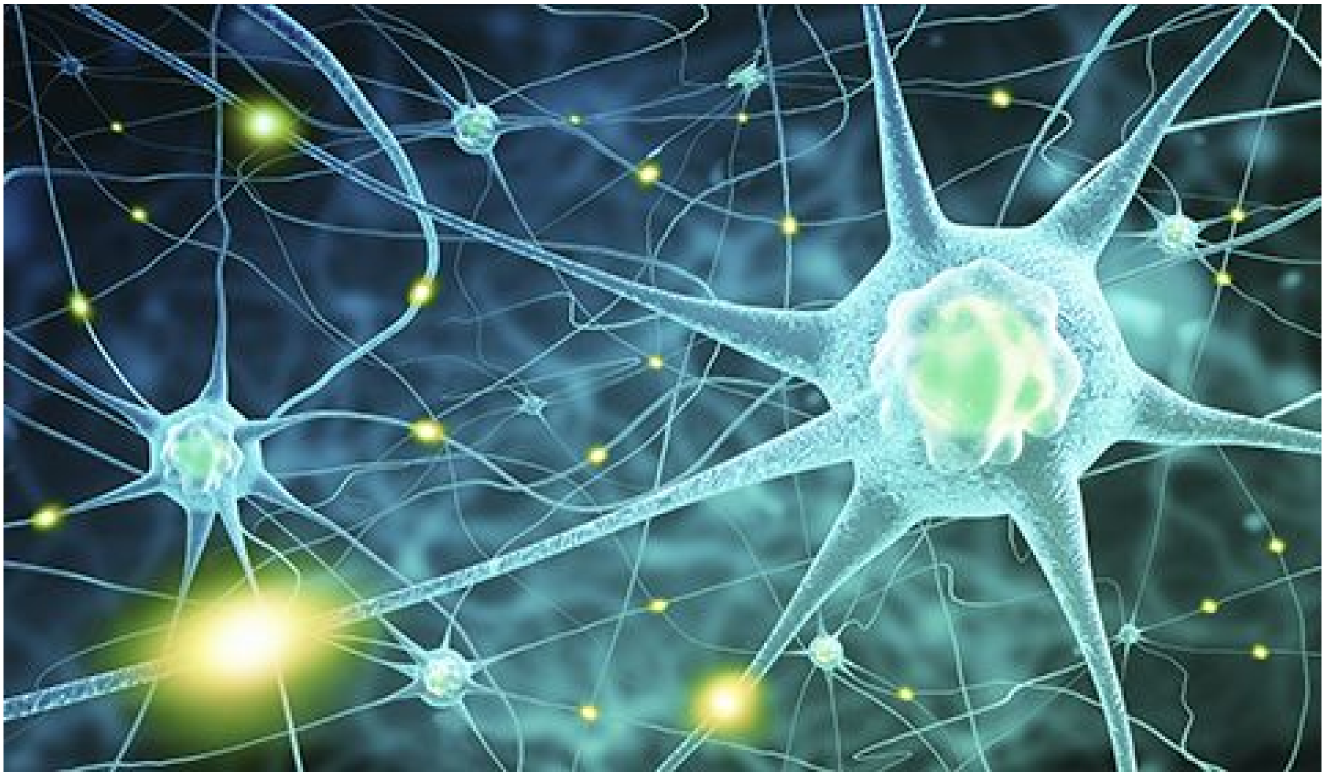
Assignment 3 Part 2

Deep Learning - CS577



Akshay R

A20442409

Task:

Problem Statement 1:

Multi-Class classification of letter recognition dataset using a fully connected neural network. The network must be tested on different loss functions then on different optimizers. The best model obtained must be tested on different regularization techniques then ensembled to give a final model.

Problem Statement 2:

Regression of abalone dataset using a fully connected neural network. The network must be tested on different loss functions then on different optimizers. The best model obtained must be tested on different regularization techniques then ensembled to give a final model.

Proposed Solution:

The letter recognition dataset has 20000 records with 16 features with no missing values. The given data is pre-processed for passing them into a neural network. First the input is tested on categorical_crossentropy, kullback_leibler_divergence and squared_hinge loss with Adam as the optimizer and the best model is selected among them.

Then the best loss is tested with Adadelta, Adagrad, Adam, Adamax, Nadam and rmsprop optimizers and the best model is selected among them based on their accuracy and convergence time. Then the best loss and optimizer is used to create various models with different regularization techniques and then ensembled to give the final model.

This Abalone dataset is a regression problem . The number of observations for each class is not balanced. There are 4,177 observations with 8 input variables and 1 output variable.

Loss functions used here are mean square error, log cosh, mean absolute error and optimizers used here are Rmsprop,Adam, Adagrad, Adadelta, Adamax and Nadam Then the best loss and optimizer is used to create various models with different regularization techniques and then ensembled to give the final model.

Implementation, methodology with results and discussion:

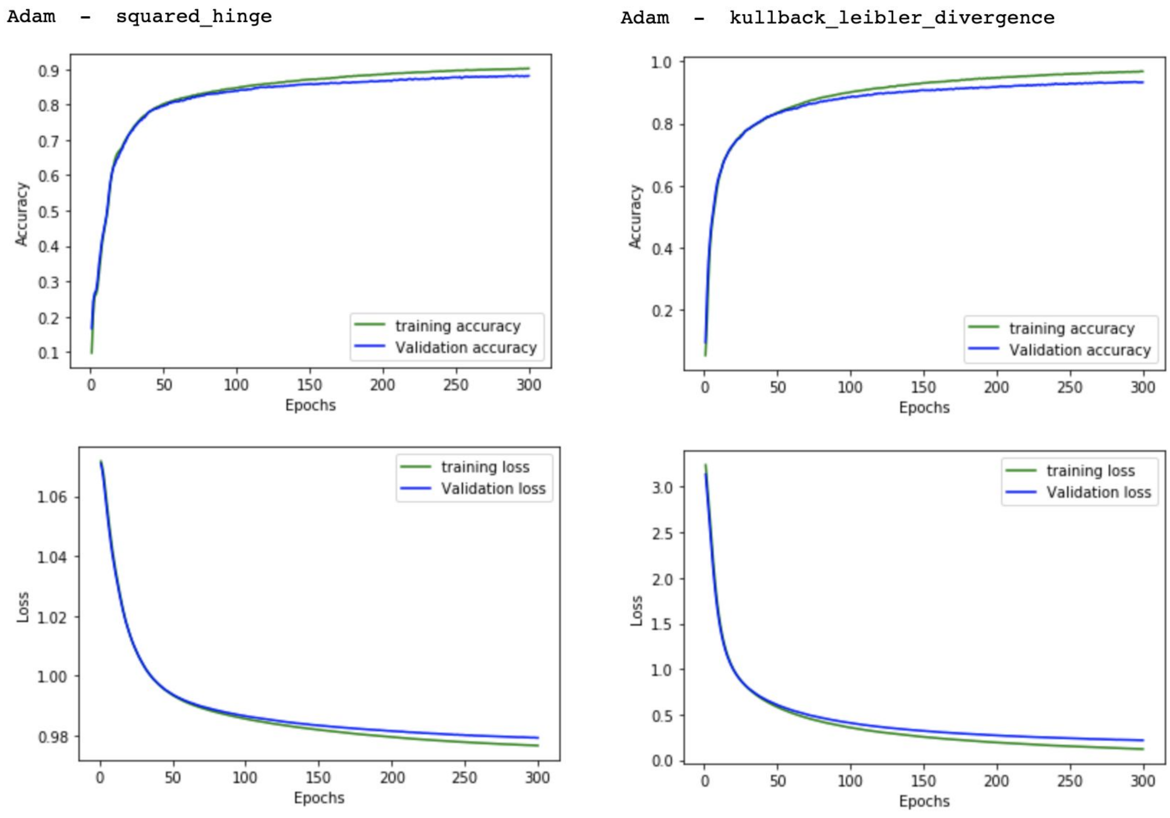
The dataset obtained is first divided into features and labels with the last column being the label and first three columns are the feature variables. The label columns are one hot encoded into class 0 to 16 respectively based on the type of label.

First the data is split into train and test data points. This is then normalized by first taking the mean and standard deviation of the train data points, then subtracting the train data by its mean value and then dividing by its standard deviation The same procedure is followed for the test data. This procedure is followed for both datasets.

The resulting data sets are further divided into train and validation datasets for training.

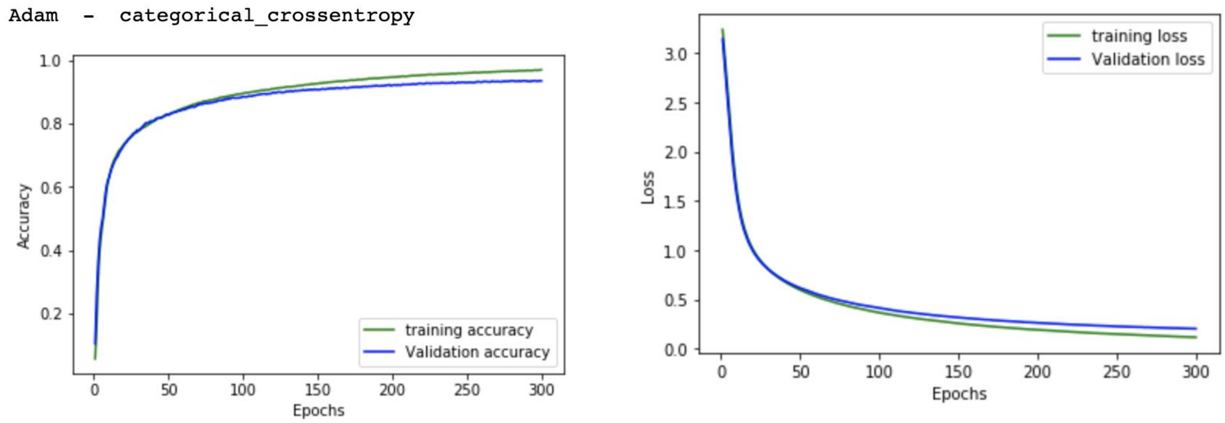
The data obtained is passed through a model created with 64 units in the first layer and 128 units in the second with 26 output neurons for each letter. This is the basic model. The activation at each layer is relu except softmax at the output. Adam is used with loss functions categorical_crossentropy, kullback_leibler_divergence and squared_hinge loss.

The results are as follows



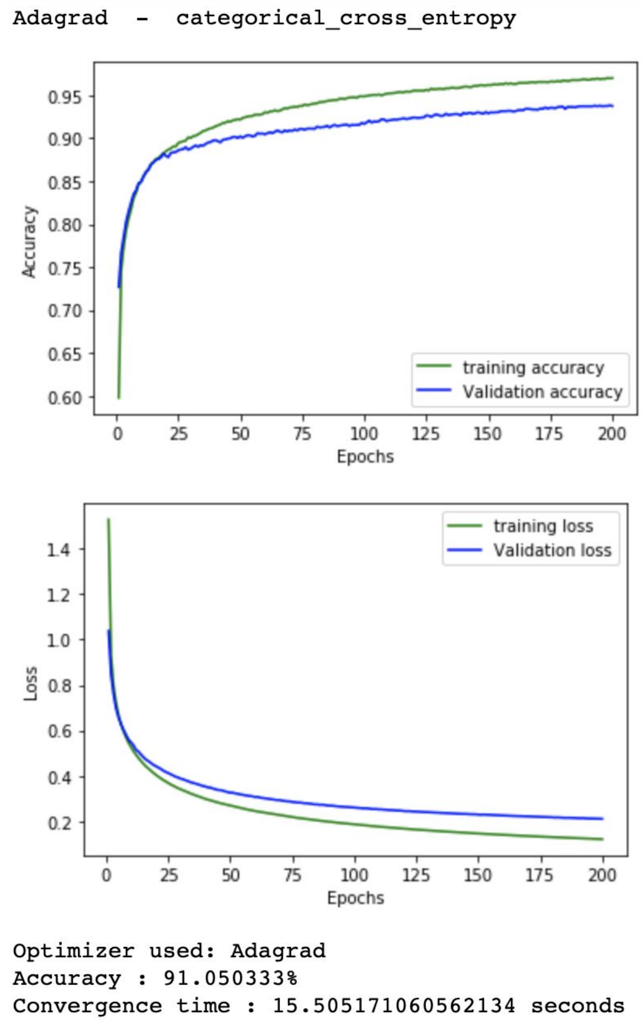
Adam - squared_hinge

Adam - kullback_leibler_divergence



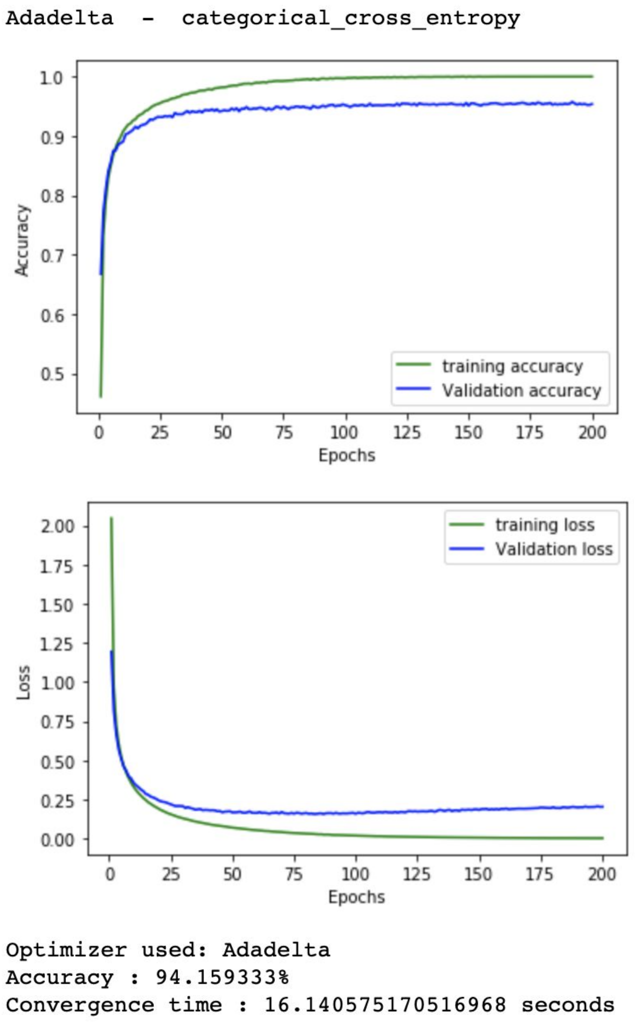
```
Adam  -  categorical_crossentropy
```

Categorical cross entropy is chosen. Now we fix categorical cross entropy as the loss and test out different optimizers. The results are as follows:



```
Adagrad  -  categorical_cross_entropy
```

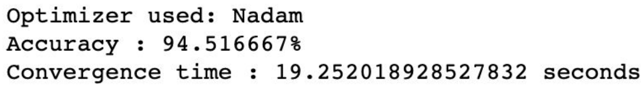
```
Optimizer used: Adagrad
Accuracy : 91.050333%
Convergence time : 15.505171060562134 seconds
```



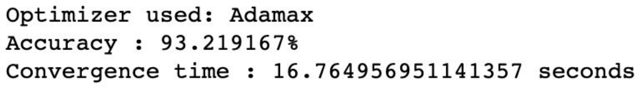
```
Adadelta  -  categorical_cross_entropy
```

```
Optimizer used: Adadelta
Accuracy : 94.159333%
Convergence time : 16.140575170516968 seconds
```

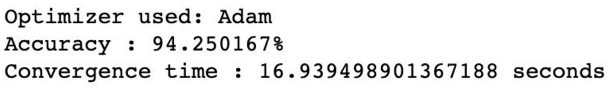
Results for other optimizers



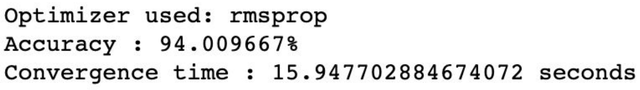
```
Optimizer used: Nadam
Accuracy : 94.516667%
Convergence time : 19.252018928527832 seconds
```



```
Optimizer used: Adamax
Accuracy : 93.219167%
Convergence time : 16.764956951141357 seconds
```



```
Optimizer used: Adam
Accuracy : 94.250167%
Convergence time : 16.939498901367188 seconds
```

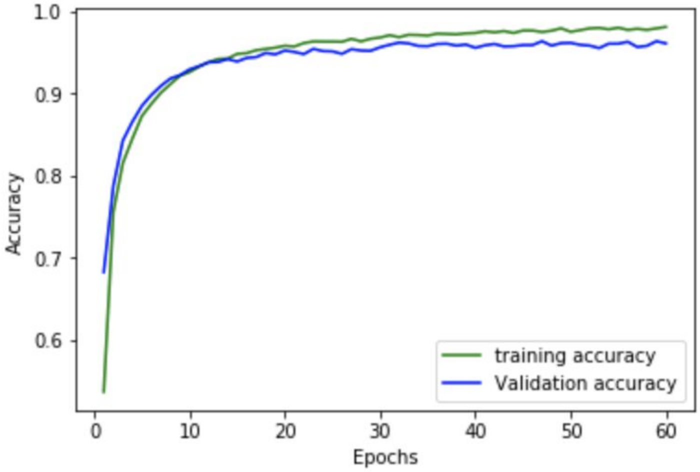


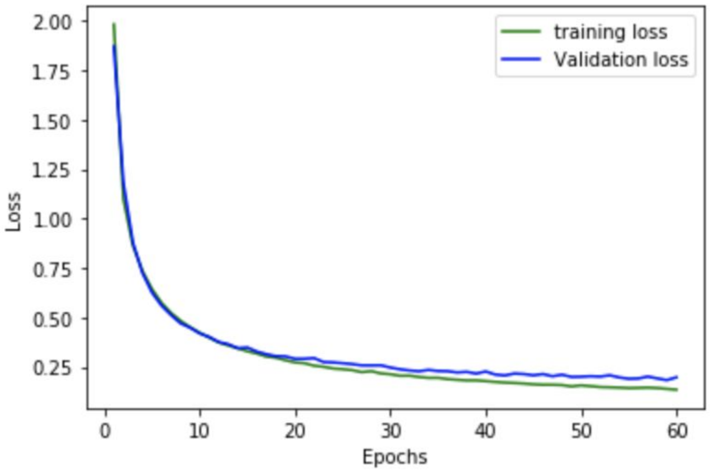
```
Optimizer used: rmsprop
Accuracy : 94.009667%
Convergence time : 15.947702884674072 seconds
```

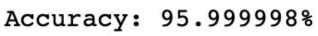
Based on the above results we choose RMSprop as the best optimizer. We then use three regularization techniques. First we build a model with weight decay, second we have a model with batch normalization and third we have a model with both normalization and dropout. These three models are ensembled and the average output is considered as the final output.

The results are as follows:

Model 1:

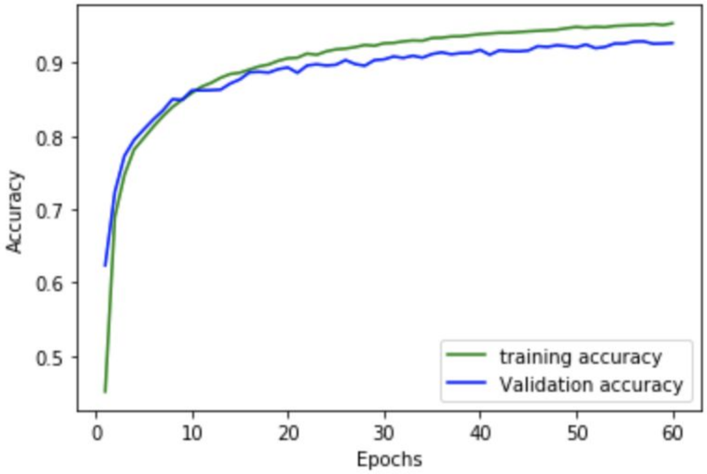


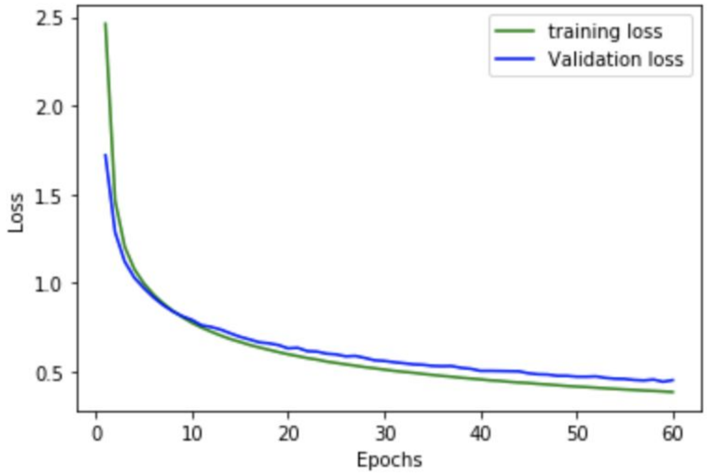




```
Accuracy: 95.999998%
```

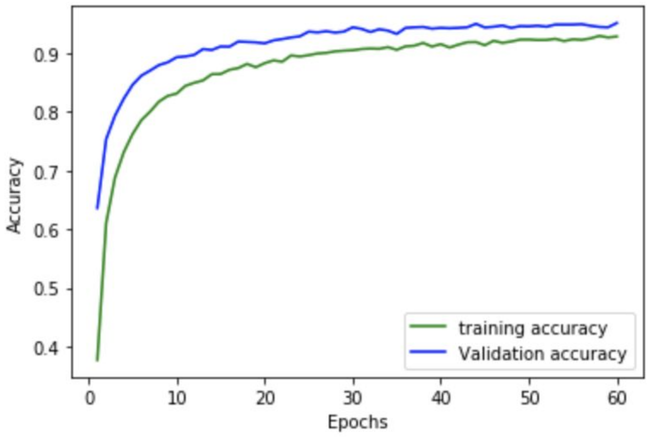
Model 2:

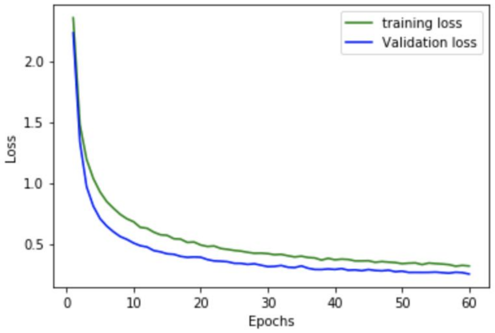




```
Accuracy: 92.460001%
```

Model 3:



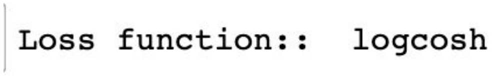


```
Accuracy: 94.720000%

Final accuracy  96.220000%
```

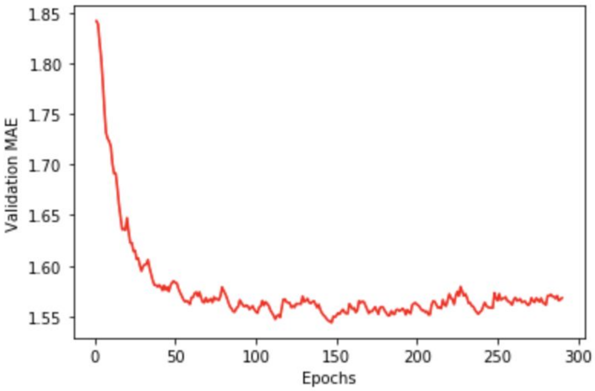
Finally we achieve an accuracy of 96%.

Similar procedure is followed for abalone dataset and the resulting loss and optimizers obtained with their statistics are as shown below.

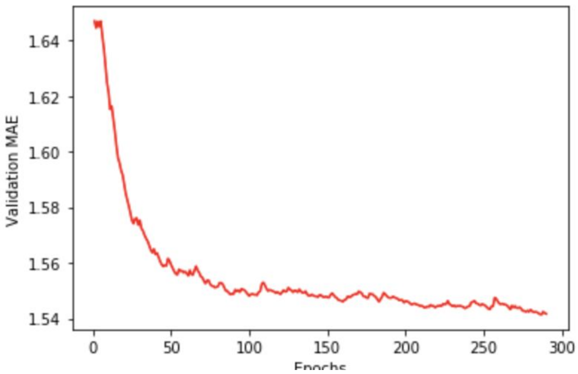


```
Loss function::  logcosh
```

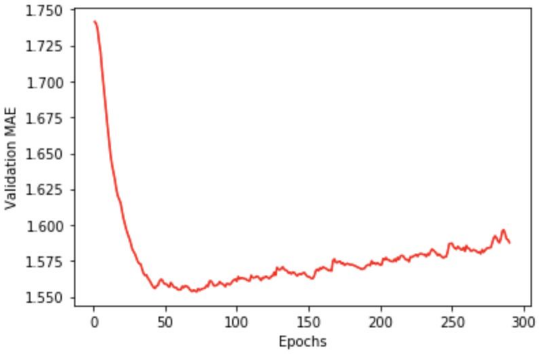
```
Adadelta - logcosh
```



```
Adagrad - logcosh
```

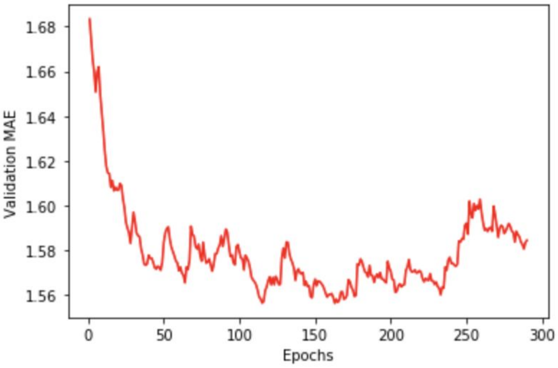


```
Adam - logcosh
```



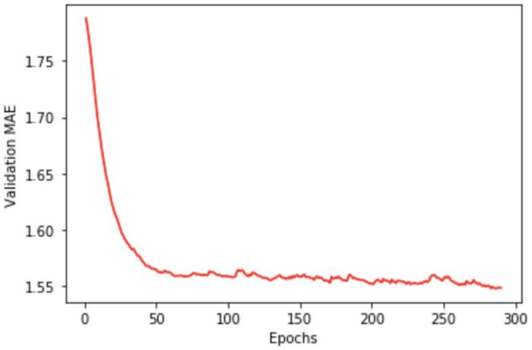
```
<Figure size 432x288 with 0 Axes>
```

```
Nadam - logcosh
```

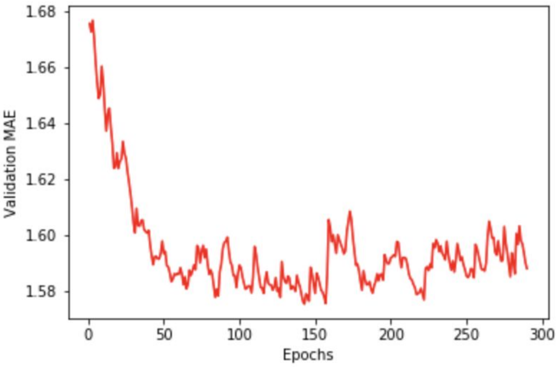


```
<Figure size 432x288 with 0 Axes>
```

```
Adamax - logcosh
```



```
rmsprop - logcosh
```



The converging time for different optimizers are as follows:

Adadelta : 2.964693069458008 seconds

Adagrad : 2.5420830249786377 seconds

Adam : 2.9216420650482178 seconds

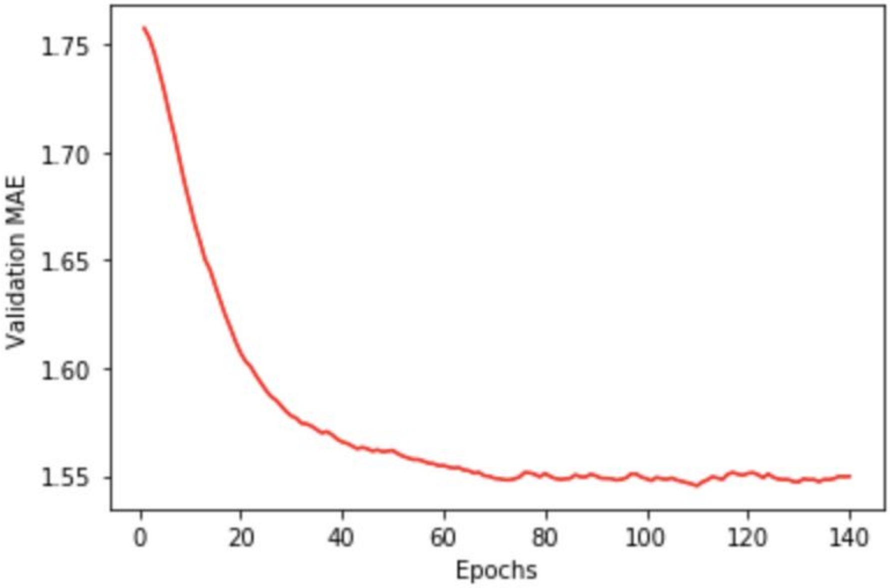
Adamax : 2.7293548583984375 seconds

Nadam :3.0427229404449463 seconds

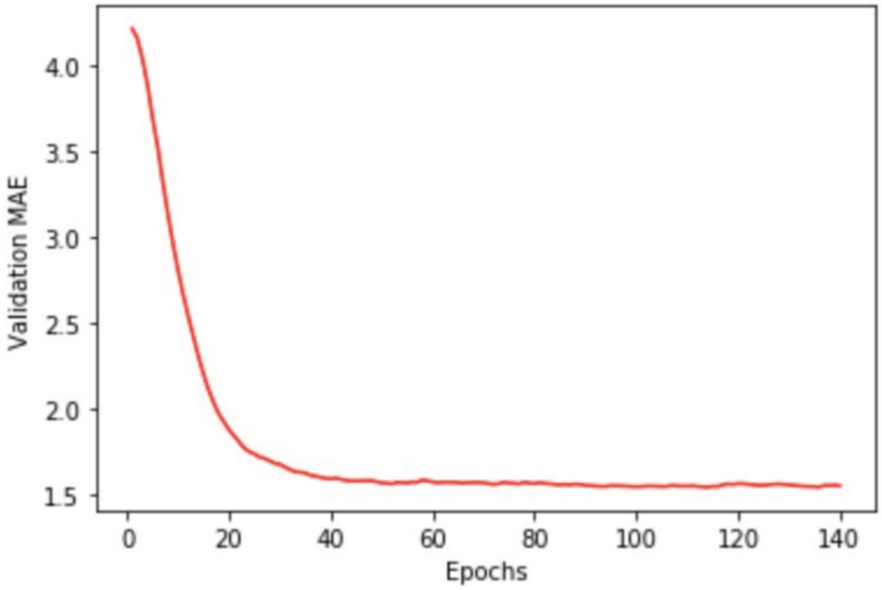
rmsprop :2.5547640323638916 seconds

Based on the converging time and the nature of the graph Adamax is chosen. Regularization methods followed during classification is followed for regression as well and the results of the resulting three models are as follows:

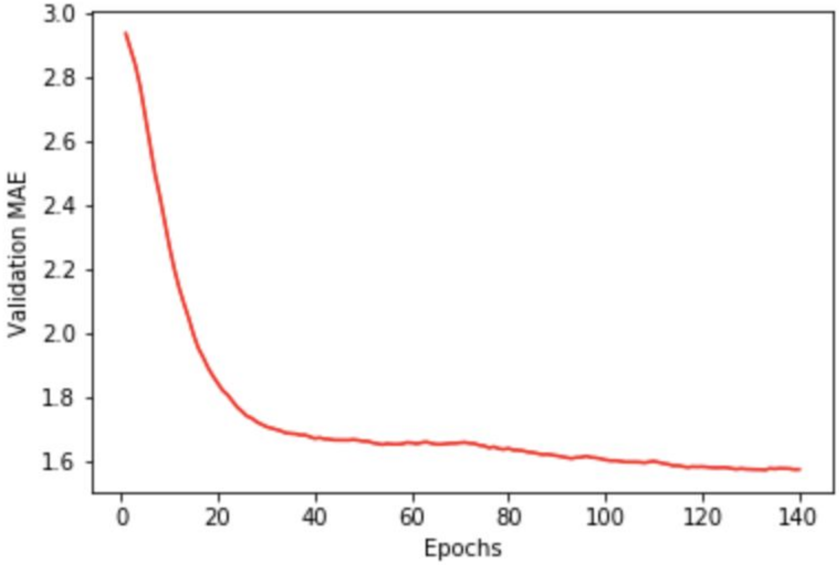
Model 1:



Model 2:



Model 3:



The three models are assembled and the final result is 1.4654536 rings for the test data.

References

- https://archive.ics.uci.edu/ml/datasets/Abalone
- https://archive.ics.uci.edu/ml/datasets/Letter+Recognition
- https://keras.io/optimizers/
- https://keras.io/losses/