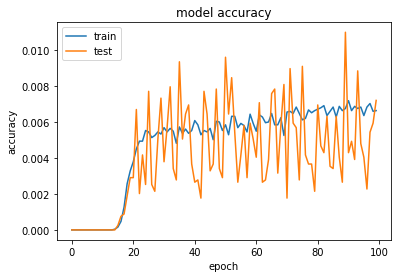
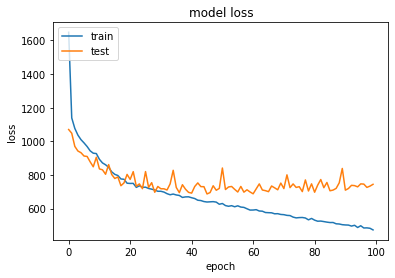
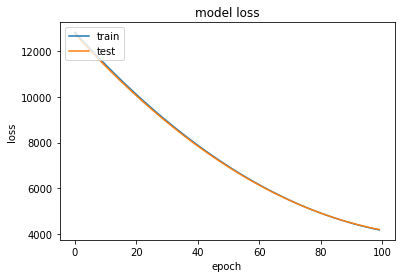
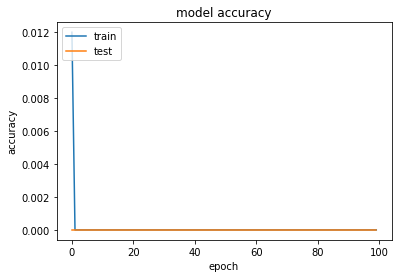
**#ANN Lab - Susara Thenuwara 179418b**

**##** **Model accuracy and loss graph with apply any initializers**



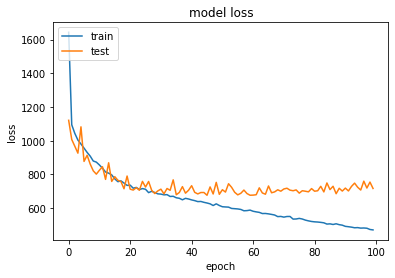
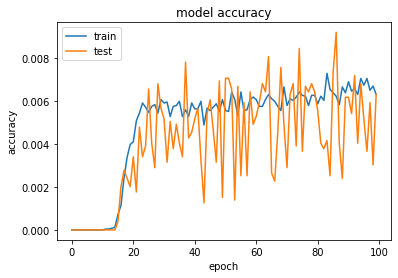
When increasing the epochs error decreased and accuracy increased.

**##Weight initialize to zero**



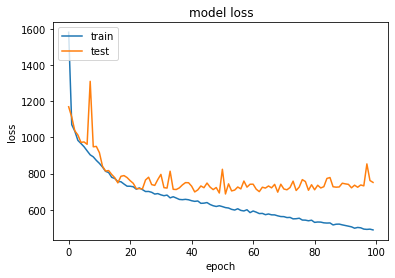
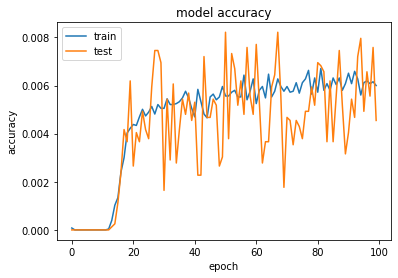
This is not suitable for this dataset. the first have the same values and symmetric features at the initialization. No variation or different features can learn this is not good procedure and not able to learn faster. All the hidden neurons have the symmetric feature.

**##Bias initialization zero**



This perfectly work as it increases the accuracy with the number of epochs and error decreased.

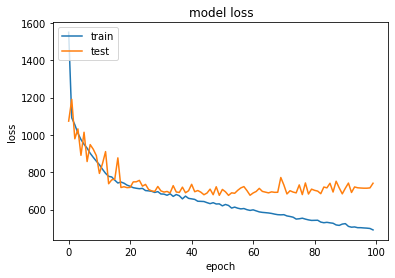
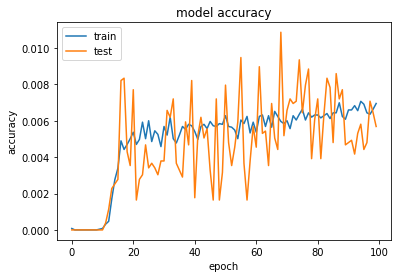
**##Weight initialize to random normal**



Weights are initialized with normal distribution randomly. Therefore, model work perfect but not perfect like truncated normal bcoz the weight initialization rage are larger. Accuracy is less than truncated normal weight initializing.

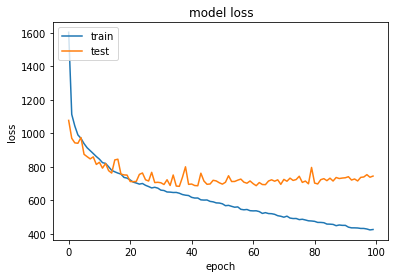
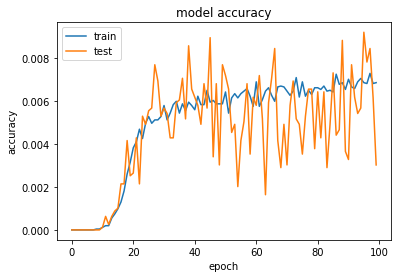
**##Weight initialize to truncated normal**

We need ‘W’ in circular form as much as possible then lean smoothly and faster. In order to do that What we need to do is W have in same scale but different, truncate the normal distribution to **limited range and initializes** the weights from that range, mean zero, sigma is 0.05. but it will give different accuracy at different experiments.



We define the standard deviation 0.05 and this work perfectly. It depends on weight values it initialized. Model accuracy increased over the number of epochs and error decreased. We need to initialize the weight in same range but different.

**##He initialization to weights**



this also work perfectly rather than going to random truncated normal. Sigma based on the number of neurons in the previous layer. Model accuracy increased over the number of epochs and error decreased. Sigma based on the number of neurons in the previous layer. Variance proportional to the number of neurons in the previous layer. It gives variation of weights as a function of hidden neuron in the previous layer. It gives minimum variance require t for the weight values to learn the features.

**##With dropouts**

With dropouts it seems to increase the test accuracy over time. This is because we are dropping the final hidden layer neurons by comparing keep probability and it eventually make the network to generalize well and increase the accuracy.

**##L2 Regularization without Dropouts**

By adding L2 regularization it obtained higher accuracy over testing batches, i.e. 0.55. This is because its adding a upper limit to the weight update an keep inside the balancing interval to memorize the features rather than making it to memorize noisy features without keeping a limit

**##L2 Regularization with Dropouts**

Although it is expected to have a higher accuracy (atleast than 0.55 with L2 regularization) surprisingly here we got rather a same accuracy as with dropouts. This can be due to the number of neurons per layer and having the less number of layers.