Assignment 1 (Things that have been tried and Observations made)

1. Normalised both training and testing data based on the mean and variance of data in the training set.
2. Split the training data into two parts so that one part can be used for validation(to tune hyperparameters), used test data only for final result comparison.
3. Compared the performance of the model with and without Data Normalisation.

Observation : Data Normalisation model a lot when the model has less number of layers. It helps the training data distribution to be close to origin and with less variance. So the model tends to converge quickly in the case where data has been normalised and takes more time if the data is not normalised. Here the validation accuracy of the model after 10 epochs in both cases is considered to compare them and there is drastic difference between the results where normalised data reaches global minimum whereas the with not normalised data it tends to learn very slowly.

4. Tuning the learning rate by taking five different learning rate (0.1, 0.01,0.001,0.0001,0.00001)

Observation : The model converges fast with learning rate of 0.001 and 0.0001. And doesn't perform well for other cases. This may be because if learning rate is low it takes lot of time to converge. If it is high then it fluctuates about the minimum and doesn't converge. Here all the model parameters are fixed and only learning rate is varied in experimental setup

5. Compared the effect of different activation functions with single hidden layer (for fixed Number of epochs)

Observation : Tanh, Relu and Sigmoid activation functions are considered. Here all the activation functions gave almost same result.

Sigmoid performed little better than tanh and relu. But relu converged little faster. The reason is because there is single hidden layer so vanishing gradient problem is not present here. The gap between them increases and relu performs better than tanh and sigmoid if we increase the number of layers.

4. Compared performance by varying number of neurons in the hidden layer.

   Observation : Here the model with only one hidden layer is considered and number of neurons in that layer are changed. 512, 784, 1024 neurons are considered in hidden layer. Almost all performed well and there is not much difference expect the converging time.

5. Tried with one, two, three, four and five hidden layers for fixed number of epochs (Demonstrated overfitting, Time for the model to train, No of parameters for the model)

   Observation : The problem is little simple and one hidden layer is enough to learn the mapping function and as we increase the number of hidden layers the model complexity increases and it overfits and takes more time to converge.

6. Tried different optimisation algorithms like RMSProp, Adagrad, Adadelta, Adam and Stochastic Gradient Descent by keeping same learning rate.

   Observation : All the optimisation algorithms converged within 10 epochs. Adam and RmsProp have converged faster than other optimisation algorithms. Adagrad also converged at the same rate. But Adadelta took lot of time to converge

7. Plotted graphs for Comparison of different variations of the models.

All the comparisons are plotted using matplotlib and all the results are stored in CSV files.