The final accuracy on test data was 82.33%. Precision and recall table is shown below

	precision	recall	f1-score	support
0.0	0.85	0.95	0.89	4714
1.0	0.66	0.37	0.47	1286
avg / total	0.81	0.82	0.80	6000

Confusion matrix is displayed below

Predicted Class

True class

	0	1
0	4469	245
1	815	471

The final architecture was decided after just lot of trial and errors. I tried increasing no. of layers, no. of neurons in each layer adding dropout(to prevent overfitting) but could not get an accuracy better than 81.7%. I went with ReLu activation for each layer because that generally works well.

As far as parameters go, learning rate was fixed as that only affects the convergence rate and not the convergence point itself. For no. of epochs, I tried 10 first but that was too small, hence increased it and finally settled on around 50. For batch size of 10 the training speed was too slow, increasing it to 50 fairly increases the speed by decreasing the no. of batches per epoch. I But for neurons and no. of layers, I started out with a small number and then kept on increasing until the accuracy saturated on validation set. So, yes I used parameter sweep for deciding on the no. of layers and no. of neurons.

The only interesting thing was how dropout fixes the overfitting problem. After a few epochs validation accuracy saturated while training accuracy kept on increasing. This signified a overfitting problem. Adding dropout after every layer fixes this greatly.

The problem was simple enough. Wanted to achieve higher accuracy but could not.