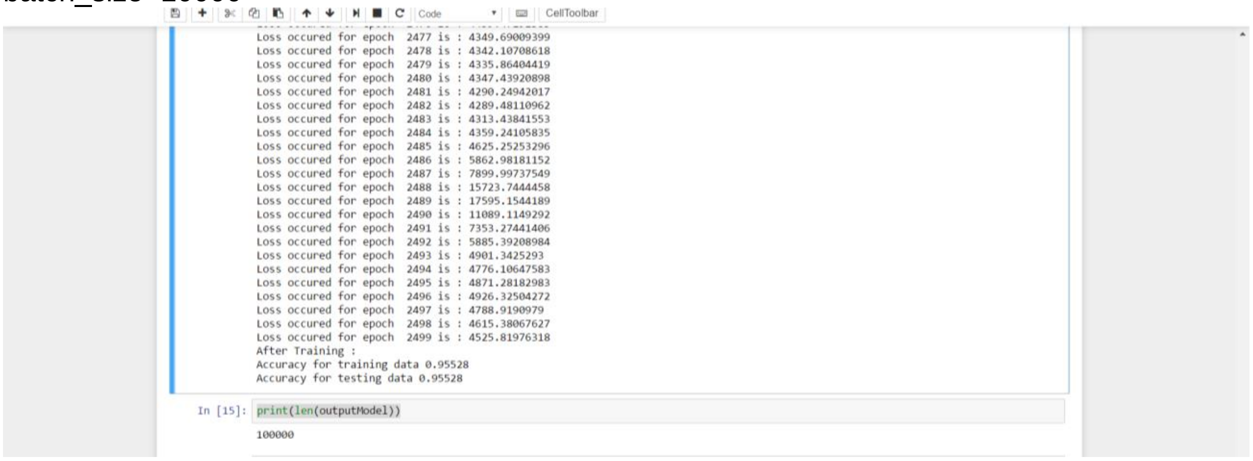
Venkata Sai Abhishek Gogu
U00825706
Soft Computing - Assignment 3

---

a) Detail, supported by actual performance data, how well your best network(s) performed. Do you consider the level of performance adequate to the real world task envisioned? Why or why not?

A) My network model has 1 input layer, 3 Hidden layer and 1 output layer. Also input layer consists of 10 input features, 10 neurons in hidden layer 1, 6 neurons in hidden layer 2, 3 neurons in hidden layer 3 and 1 output neuron. The 3 hidden Layers uses sigmoid as activation function and output layer uses tanh activation function.
Number of nodes in hidden layer 1 = 10
Number of nodes in hidden layer 2 = 8
Number of nodes in hidden layer 3 = 5
Output Classes = 1
batch_size=10000



```
Loss occured for epoch  2477 is : 4349.69009399
Loss occured for epoch  2478 is : 4342.10708618
Loss occured for epoch  2479 is : 4335.86404419
Loss occured for epoch  2480 is : 4347.43920898
Loss occured for epoch  2481 is : 4290.24942017
Loss occured for epoch  2482 is : 4289.48110962
Loss occured for epoch  2483 is : 4313.43841553
Loss occured for epoch  2484 is : 4359.24105835
Loss occured for epoch  2485 is : 4625.25253296
Loss occured for epoch  2486 is : 5862.98181152
Loss occured for epoch  2487 is : 7899.99737549
Loss occured for epoch  2488 is : 15723.7444458
Loss occured for epoch  2489 is : 17595.1544189
Loss occured for epoch  2490 is : 11089.1149292
Loss occured for epoch  2491 is : 7353.27441406
Loss occured for epoch  2492 is : 5885.39208984
Loss occured for epoch  2493 is : 4901.3425293
Loss occured for epoch  2494 is : 4776.10647583
Loss occured for epoch  2495 is : 4871.28182983
Loss occured for epoch  2496 is : 4926.32504272
Loss occured for epoch  2497 is : 4788.9190979
Loss occured for epoch  2498 is : 4615.38067627
Loss occured for epoch  2499 is : 4525.81976318
After Training :
Accuracy for training data 0.95528
Accuracy for testing data 0.95528

In [15]: print(len(outputModel))

100000
```

The outputs were 95.528% accurate. The level of performance isn’t highly adequate as my output had 4.51% flaws when compared to real world outputs. Though the level of performance could be increased with more neurons in hidden layers, but with increase in neurons the data seemed to become overfitted sometimes giving 100% results.

---

b) Explain what training and validation preprocessing steps you took. Why did you take them? How to you presume your preprocessing would help your training?
While looking at the raw data, it seemed to be highly scattered. I wanted the data to be scaled within a small range and take only the important features which are primarily contributing to the data cloud. So I did PCA to the data(both for training and validation).
Calculated covariance and eigen vectors for the data. I chose the top 10 features since they were contributing to 99.9903% of the whole data cloud. This seemed to give pretty bright results than when I did with all 90 dimensions.
Using input data and PCA, we calculated co-variance matrix. This matrix is then dot mutiplied with training data of x and also with testing data of x.So, that everything is lined up to the same axis.

“ For your case in particular you should do the following: Calculate the principal components WWs on the training dataset and then utilize the training sample WW to reduce the dimensions of the testing dataset. I say this because:

1. if you merged both your training and testing dataset to calculate your PC you will evidently utilize information from the testing set. This is clearly wrong.
2. If you did two independent PCAs you will be comparing data registered on different axes (if anything principal components are not sign-identifiable so the estimated parameters from them will also have the same issue). The axes over which you project your data should be the same, otherwise you are in a typical "*orange-apples situation*".

“

Reference - http://stats.stackexchange.com/questions/114560/pca-on-train-and-test-datasets-do-i-need-to-merge-them#114565

---

c) Explain why you chose the architecture you used. Note that this isn't any one universal answer to this question. There are, however, strategies you can use to narrow your choices. Some of these are mentioned in the book. Others can be gleaned from relevant literature available with a modest amount of research.
I chose the 3 hidden layers with 10,8,5 neurons in each respectively and one output neuron for classification. The reasons why I chose this architecture was it gave better results when compared to others. I tried various other architectures before landing with this one. One hidden layer with 20 neurons which gave me 56% accuracy. Two hidden layers with 20,5 neurons respectively which was oscillating with 65-70 % accuracy. Various architectures gave me different accuracy results, though the architecture isn’t the only reason for accuracy. Epochs also mattered a lot which I observed predominantly during the process.

---

d) What training termination conditions did you use? Why did you use them?

I used two termination conditions-

1. When the maximum epoch count meets the iteration, the function stops or
2. When the absolute difference between present epoch error and previous epoch error was less than 0.0001, the function stops.

First termination condition was used, because there were high chances for the training to happen infinite times. So this stops from being iterated infinitely
Second termination condition was used to stop the iteration as soon as the error reaches a minimum.

---

e) Explain how you chose to measure performance of your final products, including how you determined that the final products do indeed generalize to an acceptable level.
I measured performance by calculating cost(½*(y-y_hat)^2) and accuracy. The final product did have few wrong results but compared to other architectures the accuracy results seemed to give more generalised results with the chosen one. The computations were highly time consuming to

notice any errors in architecture and make changes for a better one. If there was more time I could’ve trained my data with more epochs, to bring out finer results.
