Name - Rohila Gudipati

UIN - 01058204

Compile and execution of the code:

- 1. The system needs to have python version 2.7 or up
- 2. Install scikit-learn using the following command, if doesn't exist sudo pip install scikit-learn
- 3. Download the neural py file and data file named digitsData. The data file needs to have a statement 'EOF' at the end of file
- 4. Execute this python file using the following command

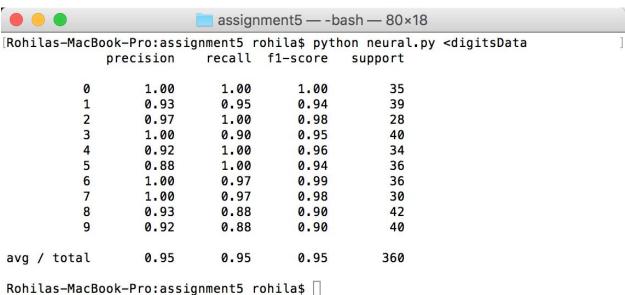
Python neural.py <digitsData

5. The classification report of the trained model is displayed on console.

Output

1. Training data - 80%, Testing Data - 20%

 $Hidden\ nodes = 10$



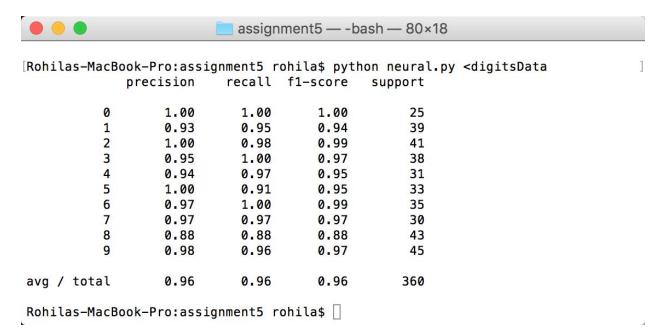
2. Training data - 80%, Testing Data - 20%

Hidden nodes = 100

		assign	ment5 — -b	ash — 80×18		
[Rohilas-MacBe	ook-Pro:assi	gnment5 r	ohila\$ pyth	on neural.py	<digitsdata< td=""><td>]</td></digitsdata<>]
	precision	recall	f1-score	support		
0	1.00	1.00	1.00	33		
1	0.93	0.97	0.95	29		
	A. A					
2	0.97	1.00	0.99	34		
3	0.94	0.94	0.94	32		
4	0.97	1.00	0.99	39		
5	0.98	0.91	0.94	46		
6	1.00	1.00	1.00	35		
7	0.98	0.98	0.98	42		
8	0.91	0.91	0.91	34		
9	0.94	0.94	0.94	36		
avg / total	0.96	0.96	0.96	360		
Rohilas-MacB	ook-Pro:assi	gnment5 r	ohila\$ ∏			

3. Training data - 80%, Testing Data - 20%

 $Hidden\ nodes = 500$



4. Training data - 80%, Testing Data - 20%

$Hidden\ nodes = 1000$

ohilas-MacB		The second second second	The second secon	Commence of the Commence of th	<pre><digitsdata< pre=""></digitsdata<></pre>
	precision	recall	f1-score	support	
0	1.00	1.00	1.00	31	
1	0.92	1.00	0.96	35	
2	1.00	1.00	1.00	38	
3	1.00	1.00	1.00	37	
4	0.97	0.97	0.97	34	
5	1.00	0.98	0.99	41	
6	1.00	0.94	0.97	36	
7	1.00	0.94	0.97	35	
8	0.95	0.95	0.95	37	
9	0.95	1.00	0.97	36	
avg / total	0.98	0.98	0.98	360	

Analysis

I have used 1024 features to build the neural network model and have used 80% of data for training and 20% for testing. I have trained the model for various sets of hidden nodes, that is 10, 100, 500, 1000 and the accuracy of prediction is seen to be increasing with increase in number of hidden node. We can see this in above outputs, the precision of model is 0.95 when the number of hidden nodes are 10, where as precision is 0.98 when the number of hidden nodes are 1000. I have used the MLP classifier which is Multi-layer perceptron algorithm that trains using back propagation. Also, all parts of code is explained clearly as comment in the neural.py file.