REPORT ASSIGNMENT-3

DECISION TREES

a)

a) 1. Computed the median of attributes (independently)in the training data without ignoring the duplicates.
b) Replaced each value of the attribute by 0/1 depending upon the median threshold.
c) Build a decision tree using information gain as metric for choosing attribute to split.
d) In case of tie chose the attribute that comes first.
e) Build a tree data-structure in Python and used it to store the decision tree.

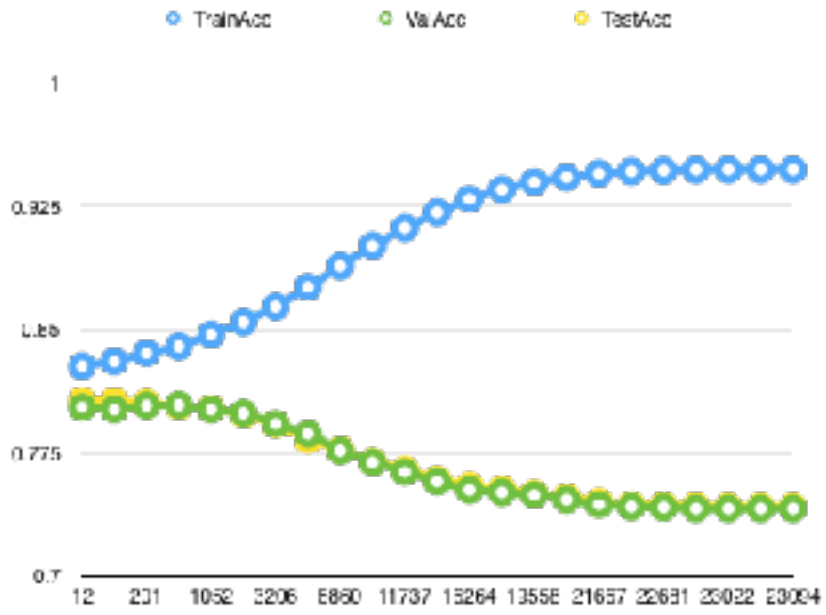
Obtained the following curve (**number of nodes in BFS order**) :

1. The train accuracy increases as we increase the number of nodes of the trees we start overfitting the training data hence when we test on the train data itself we get higher accuracies.

2. The test and validation accuracies show similar nature as both are unseen by the tree. The accuracies for them decreases as we in increase the number of nodes.

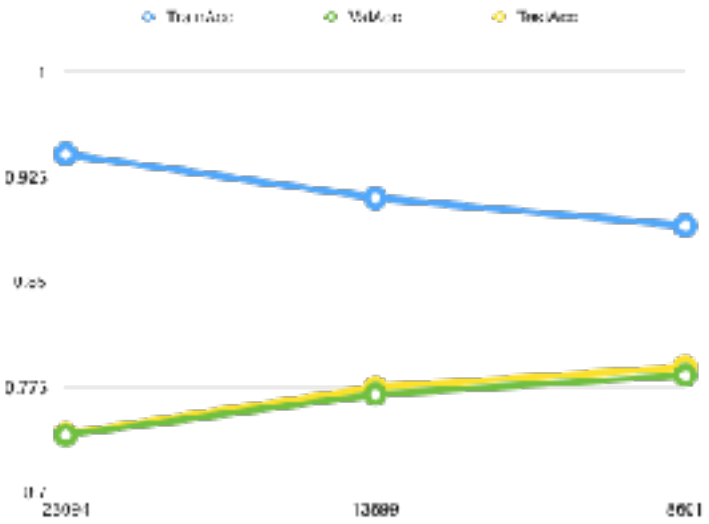
3. The Validation acc did slightly increase at first best obtained at a height of 4. This tree could be said to be one which has recognised the patterns of the train set without overfitting it and will be able to generalise well.

4. However then the tree just learns to represent the training data in a better fashion and fails to generalise on test and validation sets.

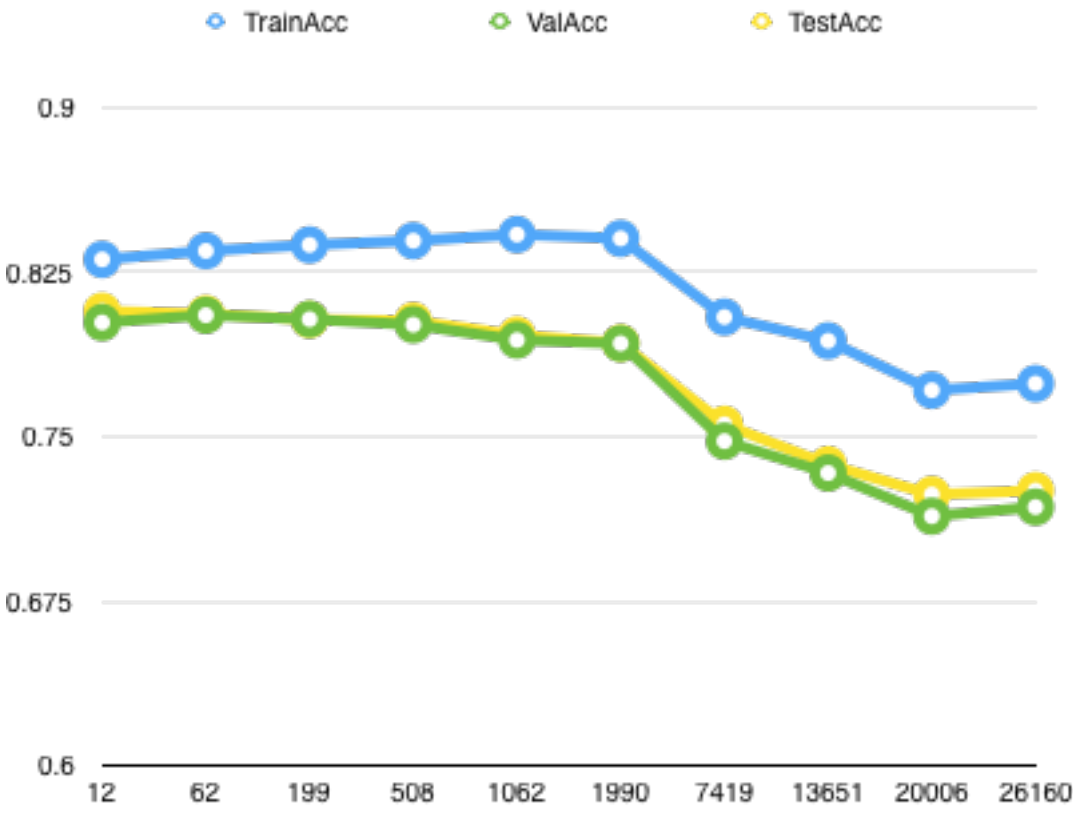


b) Did greedy post-pruning of the tree built previously, iteratively picking up a node that gives maximum increase in accuracy on validation set, and repeated this process until further pruning leads to decrease in accuracy.

prune log file in submission



c) No pre-processing is done in the beginning. At any given internal node of the tree, a numerical attribute is considered for a two way split by calculating the median attribute value from the data instances coming to that node, and then computing the information gain



d) Validation Accuracy :

Min sample leaf is an integer grater than one signifies the minimum number of samples a node must have if its a leaf.

S.NO	MIN_SAMPLES_LEAF	MIN_SAMPLES_SPLIT	MAX_DEPTH	VALIDATION ACC.
1	92	92	-	0.805
2	2	5	3	0.805
3	2	5	6	0.8051677
4	22	92	4	0.806
5	2	5	h>6	Decreases

log of different values : In Submission

At higher values of min_samples_leaf and min_samples_split the validation accuracy is independent of height(above a certain height).

At small values of min_samples_leaf and min_samples_split the validation accuracy is decreases with height due to overfitting. (above a certain height).

Best Accuracy 0.806 is obtained for height 4, min_sample_leaves = 22 , min_sample_split = 92 .

The accuracy obtained here is slightly grater than that obtained in part b,c.

Validation 0.804333333333
Train 0.831166666667
Test 0.810833333333

e)
First appended all the samples of train, test and validation sets into 1 array and then applied one-hot coding using “pd.get_dummies”
and then again divided the appended samples into three(train,test and validation).

Validation 0.798333333333
Train 0.8305
Test 0.809166666667

The accuracy’s obtained are slightly lower in all three cases of part d.

The accuracy obtained here is slightly grater than that obtained in part b,c.

f)

	TREES	BOOTSTRAP	N_FEATURES	ACC
	1	F	80	0.8016
	19	F	40	0.8035
	21	F	50	0.8045
	7	F	30	0.806

log file in Submission

Even with very less features with bootstrap = False we obtain significant accuracy

With medium n_features(40-50) we tend to get higher accuracies.

Maximum accuracy was obtained for 7 trees with bootstrap = False and N_features = 30

We tend to get higher accuracies with bootstrap = True probably allowing the model to fit without overfitting the train data

```
Validation  0.805
Train 0.828
Test 0.806333333333
```

NEURAL NETWORKS

a) One hot encoder code made. Added the encoded train and test files in the drive.
link : **https://drive.google.com/open?id=1cZ5xsNaz-08-0x6UZ2VFJtHfTACg2KUM**

b) Implemented a generic code of neural network back-propagation algorithm.

c)

Stopping Criteria :
Difference in J(THEETA) across epochs is less than epsilon.(0.00001) or epochs greater than than 350

Hidden layer units : **5**
Train Acc : 0.64 Train Time: 59s
Test Acc : 0.63 Test Time: 11.89s

Confusion Matrix:

```
[432535, 217078, 14807, 9809, 2808, 1731, 358, 61, 13, 8]
[81164, 216012, 34021, 11825, 1170, 319, 1102, 175, 4, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **10**
Train Acc : 0.8257 Train Time: 70 s
Test Acc : 0.812 Test Time: 12

Confusion Matrix:
```
[486856, 86374, 2001, 3323, 3590, 1944, 44, 7, 14, 8]
[26843, 346716, 46827, 18311, 388, 106, 1416, 229, 3, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **15**
Train Acc : 0.83 Train Time: 88.8s
Test Acc : 0.809 Test Time: 12.58s

Confusion Matrix:
```
[475646, 78285, 820, 1241, 3799, 1887, 20, 11, 15, 7]
[38053, 354805, 48008, 20393, 179, 163, 1440, 225, 2, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **20**
Train Acc : 0.9229 Train Time: 103s
Test Acc : 0.9207 Test Time: 13.25s

Confusion Matrix:
```
[512623, 1917, 67, 410, 3924, 2041, 1, 41, 16, 8]
[1076, 431173, 48761, 21224, 54, 9, 1459, 195, 1, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **25**
Train Acc :0.92 Train Time: 144s
Test Acc : 0.91 Test Time: 11.97 s

Confusion Matrix:
```
[511416, 4355, 19, 463, 3853, 2038, 0, 31, 15, 8]
[2283, 428735, 48809, 21171, 125, 12, 1460, 205, 2, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

d)
Hidden layer units : **5**
Train Acc : 0.63 Train Time: 54s
Test Acc : 0.62 Test Time: 11.89s

Confusion Matrix:
```
[427484, 224315, 13218, 9077, 2917, 1727, 214, 55, 13, 7]
[86215, 208775, 35610, 12557, 1061, 323, 1246, 181, 4, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **10**
Train Acc : 0.79 Train Time: 76s
Test Acc : 0.77 Test Time: 11.89s

Confusion Matrix:
```
[462192, 98840, 2518, 3258, 3138, 1860, 67, 5, 13, 8]
```

```
[51507, 334245, 46255, 18376, 840, 190, 1390, 231, 4, 0]
[0, 5, 55, 0, 0, 0, 3, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **15**
Train Acc : 0.85 Train Time: 115s
Test Acc : 0.82 Test Time: 11.89s

Confusion Matrix:

```
[470119, 59373, 1346, 502, 3118, 1893, 15, 1, 15, 8]
[43580, 373656, 47133, 20685, 860, 157, 1403, 220, 2, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 61, 349, 447, 0, 0, 42, 15, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Able to predict some data of class 4 also along with 1 and 2.

Hidden layer units : **20**
Train Acc : 0.96 Train Time: 117s
Test Acc : 0.94 Test Time: 12s

Confusion Matrix:

```
[504117, 9496, 81, 20, 3847, 2009, 0, 0, 17, 8]
[9582, 416644, 4225, 8477, 131, 41, 27, 17, 0, 0]
[0, 6531, 43880, 12514, 0, 0, 1425, 211, 0, 0]
[0, 419, 642, 623, 0, 0, 8, 8, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Able to predict class 3 also very well along with 1 and 2.The power of network increased

Hidden layer units : **25**
Train Acc : 092` Train Time: 102s

Test Acc : 0.91 Test Time: 11.89s

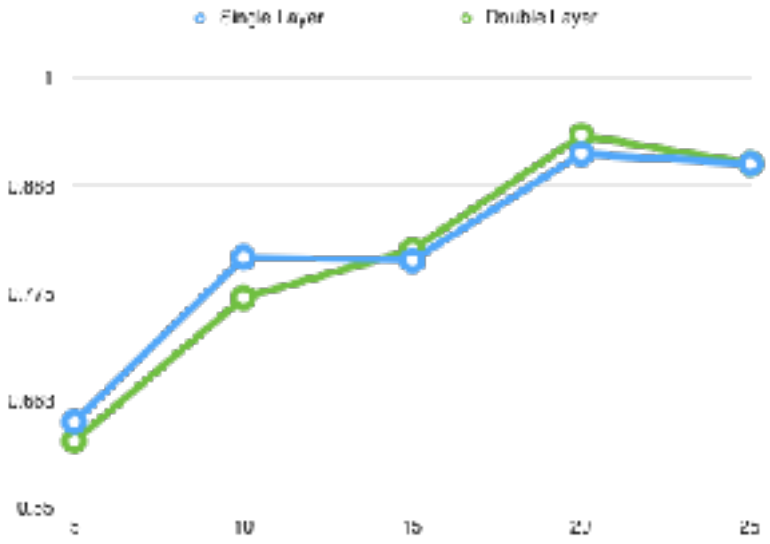
Confusion Matrix:

```
[509654, 5011, 37, 96, 3868, 2017, 2, 12, 17, 8]
[4045, 428079, 48791, 21538, 110, 33, 1458, 224, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

With increase in the number of hidden layer units the accuracy of both train and test data increased. Due to fact that network has more representation power. However increasing the units too much might lead to overfitting.

With increase in the number of layers we observed an increase in both training and test accuracies, more layers provided the network with more representational power. Extreme high layers might not generalise well.

This is plot of test accuracies. The train accuracies are just 0.1-0.3 above the test accuracy in all the cases, hence not shown to maintain the brevity.



e)
Hidden layer units : **5 2 layers**
Train Acc : 0.6437 Train Time: 36s
Test Acc : 0.6237 Test Time: 11.89s

Confusion Matrix:

```
[397024, 190334, 11208, 5801, 2647, 1548, 214, 30, 10, 2]
[116675, 242756, 37620, 15833, 1331, 502, 1246, 206, 7, 6]
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **10 2 layers**
Train Acc : 0.807 Train Time: 81s
Test Acc : 0.781 Test Time: 11.89s

Confusion Matrix:

```
[469476, 101113, 1257, 2550, 3335, 1877, 7, 1, 13, 8]
[44223, 331977, 47571, 19084, 643, 173, 1453, 235, 4, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **15 2 layers**
Train Acc : 0.895 Train Time: 142s
Test Acc : 0.885 Test Time: 11.89s

Confusion Matrix:

```
[511558, 36727, 151, 215, 3966, 2039, 0, 2, 17, 7]
[2141, 396363, 48677, 21419, 12, 11, 1460, 234, 0, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **20 2 layers**
Train Acc : 0.92 Train Time: 131s
Test Acc : 0.91 Test Time: 11.89s

Confusion Matrix:

```
[510424, 4476, 31, 32, 3845, 2032, 0, 0, 16, 7]
[3275, 428614, 48797, 21602, 133, 18, 1460, 236, 1, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **25 2 layers**
Train Acc : 0.91 Train Time: 86s
Test Acc : 0.90 Test Time: 11.89s

Confusion Matrix:
```
[507962, 13147, 111, 50, 3960, 2032, 0, 3, 17, 8]
[5737, 419943, 48717, 21584, 18, 18, 1460, 233, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **5**
Train Acc : 0.65 Train Time: 26s
Test Acc : 0.64 Test Time: 11.89s

Confusion Matrix:
```
[401082, 177145, 8212, 7075, 2861, 1618, 160, 32, 12, 7]
[112617, 255945, 40616, 14559, 1117, 432, 1300, 204, 5, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **10**
Train Acc : 0.79 Train Time: 56s
Test Acc : 0.781 Test Time: 11.89s

Confusion Matrix:
```
[471668, 102372, 1506, 2768, 3792, 1883, 8, 4, 17, 8]
[42031, 330718, 47322, 18866, 186, 167, 1452, 232, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **15 2 eta updates**
Train Acc : 0.910 Train Time: 72s
Test Acc : 0.904 Test Time: 11.89s

Confusion Matrix:
```
[511083, 17298, 163, 304, 3972, 2039, 4, 8, 17, 8]
[2616, 415792, 48665, 21330, 6, 11, 1456, 228, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **20 3 eta updates**
Train Acc : 0.919 Train Time: 79s
Test Acc : 0.912 Test Time: 11.89s

Confusion Matrix:
```
[510460, 8268, 36, 67, 3910, 2034, 0, 4, 16, 6]
[3239, 424822, 48792, 21567, 68, 16, 1460, 232, 1, 2]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : **25 5 Eta updates**
Train Acc : 0.922 Train Time: 58s
Test Acc : 0.919 Test Time: 11.89s

Confusion Matrix:
```
[512545, 2776, 1, 178, 3968, 2046, 0, 13, 17, 8]
[1154, 430314, 48827, 21456, 10, 4, 1460, 223, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

f) **RELU**

Hidden layer units **: 5** **Eta updates – 6**
Train Acc : 0.60 Train Time: 21
Test Acc : 0.597 Test Time:

Confusion Matrix:
[436297, 257016, 21580, 12820, 3442, 1754, 697, 125, 14, 7]
[77402, 176074, 27248, 8814, 536, 296, 763, 111, 3, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Hidden layer units **: 10** **Eta updates – 5**
Train Acc : 0.74 Train Time: 81
Test Acc : 0.73 Test Time:

Confusion Matrix:
[500031, 191827, 11310, 9869, 3775, 2004, 407, 91, 15, 8]
[13668, 241262, 29446, 1786, 203, 46, 355, 22, 2, 0]
[0, 1, 8072, 9979, 0, 0, 698, 123, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Hidden layer units **: 15** **Eta updates – 5**
Train Acc : 0.66 Train Time: 86
Test Acc : 0.616 Test Time:

Confusion Matrix:
[379228, 179050, 11511, 7356, 2739, 1543, 323, 68, 11, 5]
[134471, 254040, 37317, 14278, 1239, 507, 1137, 168, 6, 3]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Hidden layer units **: 20** **Eta updates – 5**
Train Acc : 0.921 Train Time: 135
Test Acc : 0.915 Test Time:

Confusion Matrix:
[510754, 5296, 21, 258, 3970, 2038, 13, 6, 17, 8]
[2945, 427794, 48807, 21376, 8, 12, 1447, 230, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Hidden layer units **: 25** **Eta updates – 3**
Train Acc : 0.922 Train Time: 64s
Test Acc : 0.916 Test Time:

Confusion Matrix:
[510943, 4848, 76, 156, 3912, 2038, 16, 5, 17, 8]
[2756, 428216, 48752, 21478, 66, 12, 1444, 231, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 26, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Hidden layer units **: 5 2 layers Eta updates –5**
Train Acc : 0.51 Train Time: 82
Test Acc : 0.506 Test Time:

Confusion Matrix:
[362361, 276100, 28548, 13848, 2730, 1451, 842, 146, 12, 6]
[151338, 156990, 20280, 7786, 1248, 599, 618, 90, 5, 2]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Hidden layer units **: 10 2 layers Eta updates – 0**
Train Acc : 0.70 Train Time: 172
Test Acc : 0.693 Test Time:

Run time overflow encountered after 140 epochs when J(theeta) was 0.0002.
Confusion Matrix:

```
[422893, 145337, 10644, 7040, 3596, 1691, 368, 60, 15, 7]
[90806, 287753, 38184, 14594, 382, 359, 1092, 176, 2, 1]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units **: 15 2 layers Eta updates – 5**
Train Acc : 0.69 Train Time: 390
Test Acc : 0.649 Test Time:

Run time overflow encountered after 207 epochs when just about to change the learning rate i.e. J_THEETA within tol.
Confusion Matrix:

```
[358696, 125308, 7617, 6180, 3230, 1531, 255, 43, 13, 8]
[155003, 307782, 41211, 15454, 748, 519, 1205, 193, 4, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units **: 20 2 layers Eta updates – 5**
Train Acc : 0.942 Train Time: 610
Test Acc : 0.937 Test Time:

Run time overflow encountered after 306 epochs when just about to change the learning rate i.e. J_THEETA within tol.
Confusion Matrix:

```
[512448, 6880, 268, 220, 3918, 2044, 10, 0, 17, 8]
[1251, 425481, 26240, 20741, 60, 6, 1210, 231, 0, 0]
[0, 604, 22273, 93, 0, 0, 234, 1, 0, 0]
[0, 100, 41, 580, 0, 0, 6, 4, 0, 0]
[0, 9, 5, 0, 0, 0, 0, 0, 0, 0]
[0, 14, 0, 0, 0, 0, 0, 0, 0, 0]
```

```
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 2, 0, 0, 0, 0, 0, 0, 0, 0]
```

Hidden layer units : 25 2 layers Eta updates – 5
Train Acc : 0.962 Train Time: 241
Test Acc : 0.961

Confusion Matrix:

```
[513595, 159, 0, 0, 3972, 2046, 0, 0, 16, 6]
[98, 432441, 9394, 21633, 4, 2, 450, 236, 0, 2]
[6, 490, 39434, 1, 2, 2, 1010, 0, 1, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Comments:

a) I encounter runtime overflow error at times when learning rate was about to adapt.
b) Obtained the best model for 2 layer perceptron with 25 units each. Was able to detect 3 classes .
c) In the 2 layer perceptron with 20 units each was able to detect 4 classes correctly.
d) The accuracies both train and test are better than corresponding previous parts .
e) The train time increased by about 1.5 times for certain architectures(2 layer 20, 25 units).

Sigmoid Relu

This is plot of test accuracies. The train accuracies are just 0.1-0.3 above the test accuracy in all the cases, hence not shown to maintain the brevity.