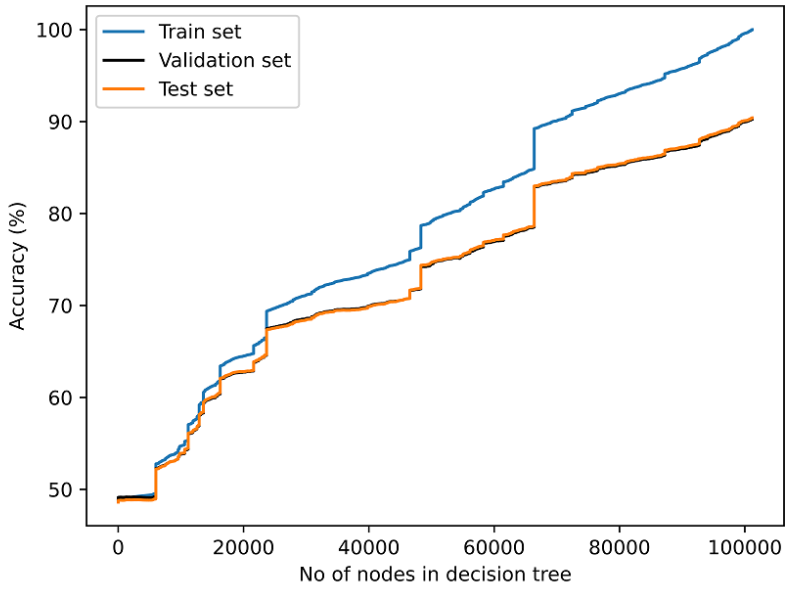
Question 1: Decision trees / Random forests

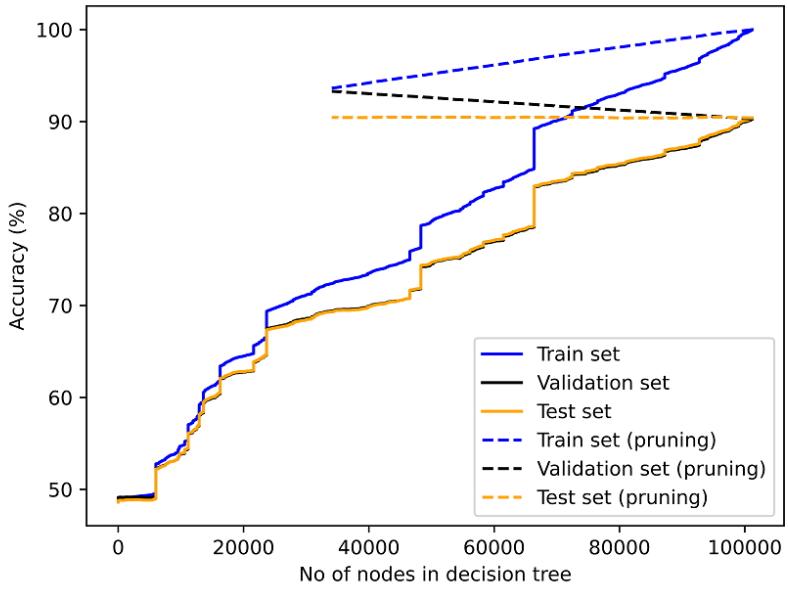
a) Decision tree:
Training time: 848s
101193 total nodes
Train set accuracy: 100.0%
Val set accuracy: 90.28010379006231%
Test set accuracy: 90.4069492152862%



It is observed that the accuracies keep increasing as we grow the tree, and reaches 100% for train set. The sudden jumps in between are because the tree was grown in a depth-first fashion, so whenever we move to a new branch there are a lot of examples which are split on an attribute, so the change in accuracy is likely to be high.

b) Pruning is performed as follows: For each node, first prune the left and right subtree recursively. Then check if the cumulative correctly classified examples in the subtrees are lesser than correctly classified examples considering the majority element at the node. If yes, then the subtrees will be pruned.

After pruning:
Nodes left: 34088
Train set accuracy: 93.63030730973506%
Val set accuracy: 93.286709265635%
Test set accuracy: 90.44396807768796%



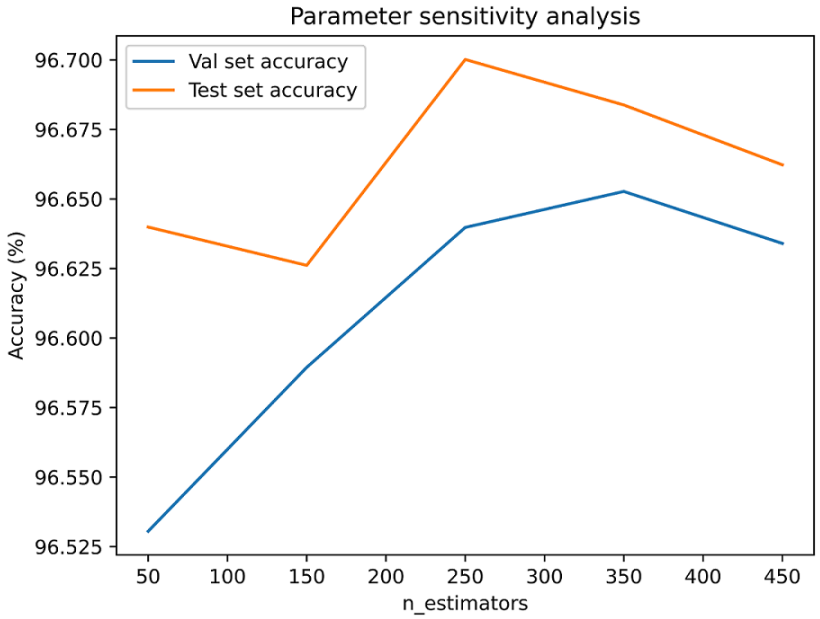
We observe that the major changes are in val set accuracy, which increases, and the train set accuracy, which decreases as expected. The test set accuracy increases only marginally.

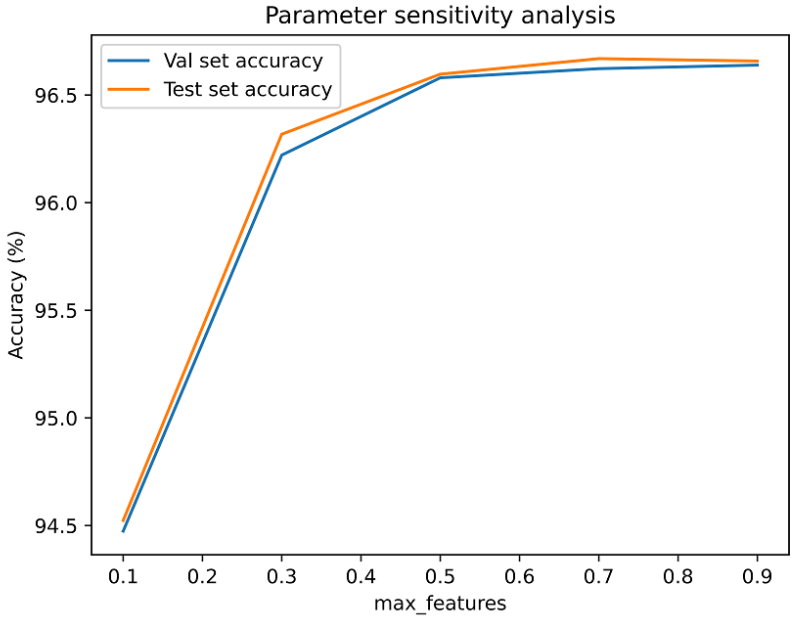
c) Optimal set of parameters:
$n\_estimators$: 350
$max\_features$: 0.7
$min\_samples\_split$: 2

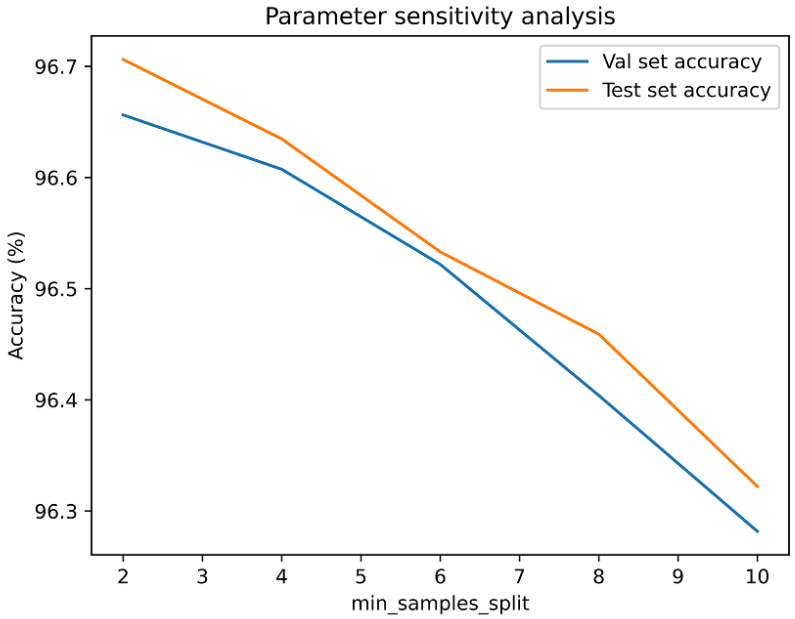
For optimal parameters:
Out-of-bag accuracy: 96.63371504006386%
Train accuracy: 100.0%
Val accuracy: 96.63544818762713%
Test accuracy: 96.68552045937827%

We observe that these accuracy figures are better than what were observed for a single decision tree in part (b)

d) The parameter sensitivity analysis is as follows:


Parameter sensitivity analysis
Val set accuracy
Test set accuracy
96.700
96.675
96.650
96.625
96.600
96.575
96.550
96.525
Accuracy (%)
50
100
150
200
250
300
350
400
450
n_estimators


Parameter sensitivity analysis
Val set accuracy
Test set accuracy
96.5
96.0
95.5
95.0
94.5
Accuracy (%)
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
max_features



We observe that our random forest classifier is most sensitive to the value of $max\_features$, since varying that lead to a difference of 2% accuracy. Other 2 parameters also influence the accuracy, but to a relatively lesser extent.

Question 2: Neural networks

a) The neural network was implemented, according to the formulae derived for forward and backward propagation in class.
For SGD, the convergence criteria used was: when the absolute difference of average cost between any 2 iterations $\leq \xi$. This value of ξ changed depending on the network architecture and learning rate.
The maximum epochs were set to be 100 in case the convergence was taking too long.

b) For learning rate 0.001, ξ was set to 10^{-4}. It was observed that the learning was happening really slowly, so the max epoch limit of 100 hit each time leading to low accuracies.

1 hidden unit
Time to train: 69.36635994911194s
Train accuracy: 17.14666666666667
Test accuracy: 14.85

10 hidden units
Time to train:

348.72922563552856s Train accuracy: 56.175
Test accuracy: 49.47

50 hidden units
Time to train: 1371.4055676460266s Train accuracy: 64.248333333333
Test accuracy: 55.93

100 hidden units
Time to train: 2608.280740261078s
Train accuracy: 64.05166666666666
Test accuracy: 53.68

500 hidden units
Time to train: 10587.649795293808s
Train accuracy: 78.14
Test accuracy: 68.18

For learning rate=0.1, the convergence happened much quicker, giving high accuracies. ξ was set to 10^{-3} here

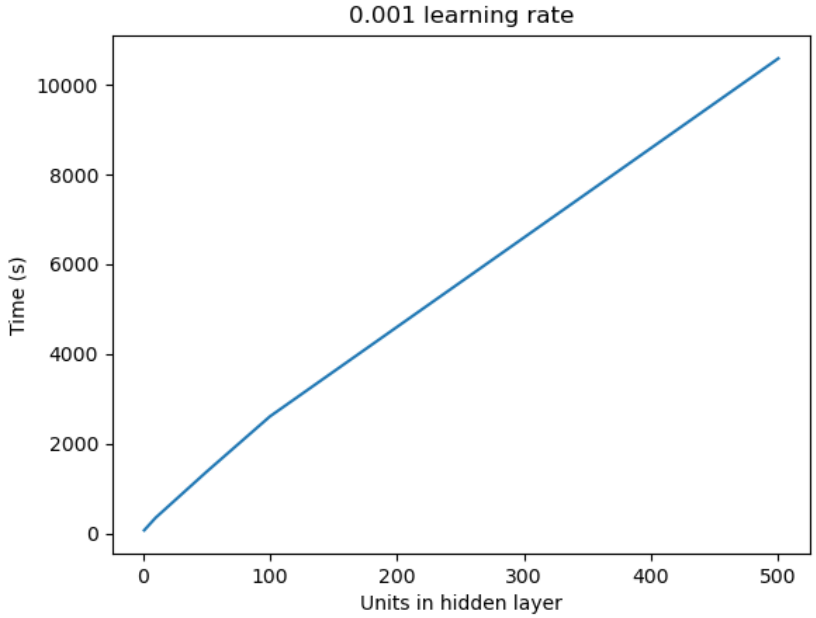
Hidden layer: 1 units
Time to train: 12.84850263595581s
Train accuracy: 20.323333333333334
Test accuracy: 19.47

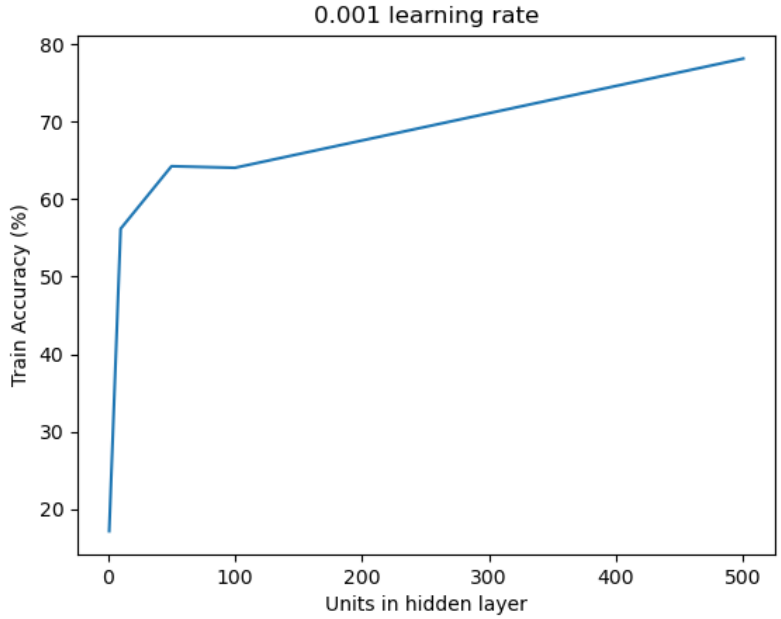
Hidden layer: 10 units
Time to train: 88.31778812408447s
Train accuracy: 95.23
Test accuracy: 87.71

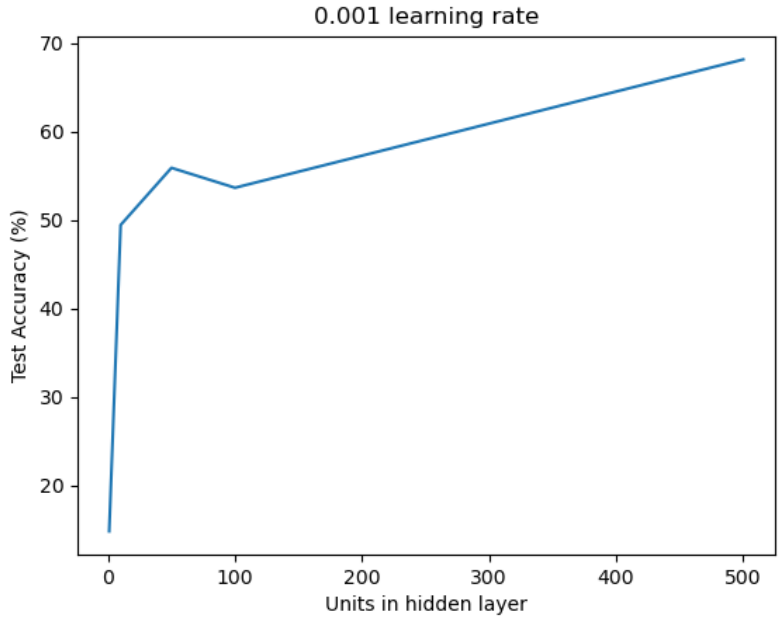
Hidden layer: 50 units
Time to train: 337.90894627571106s
Train accuracy: 95.35833333333333
Test accuracy: 88.05

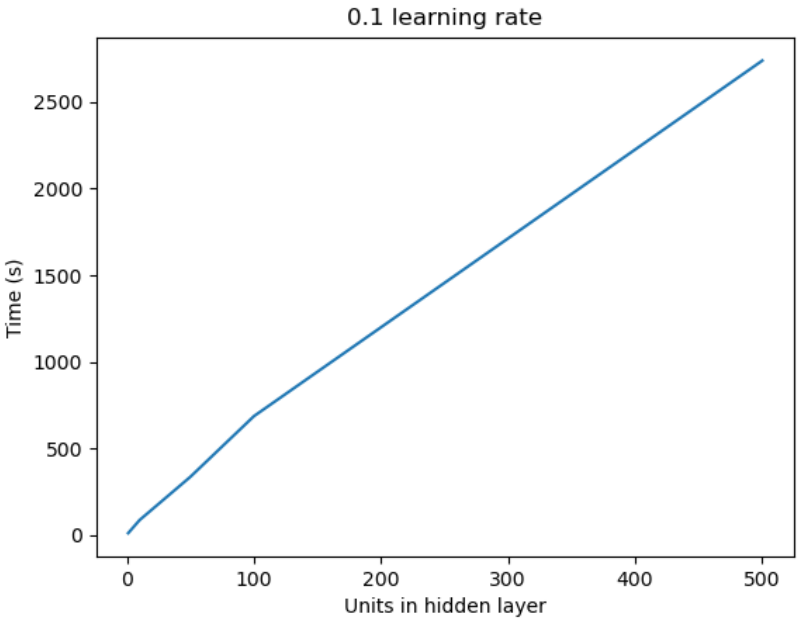
Hidden layer: 100 units
Time to train: 687.6106009483337s
Train accuracy: 95.19666666666667
Test accuracy: 87.84

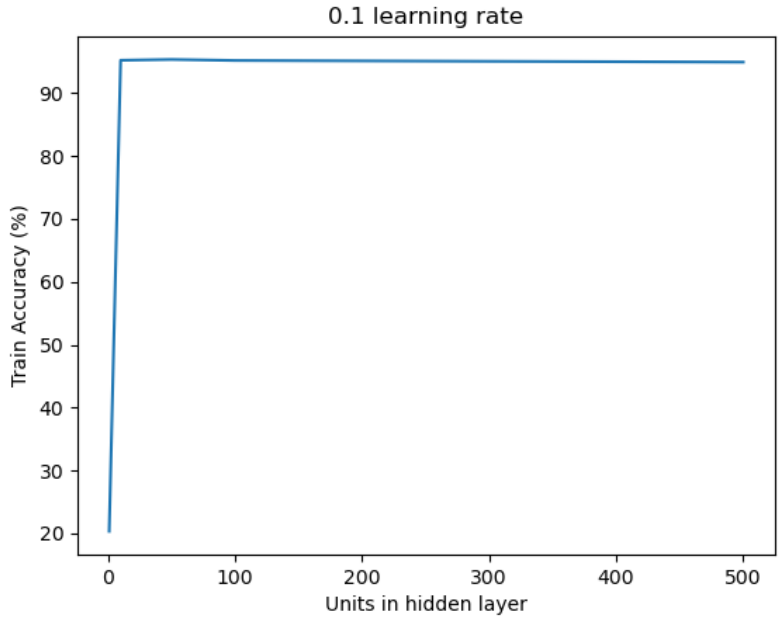
Hidden layer: 500 units
Time to train: 2737.724682569504s
Train accuracy: 94.94
Test accuracy: 87.56

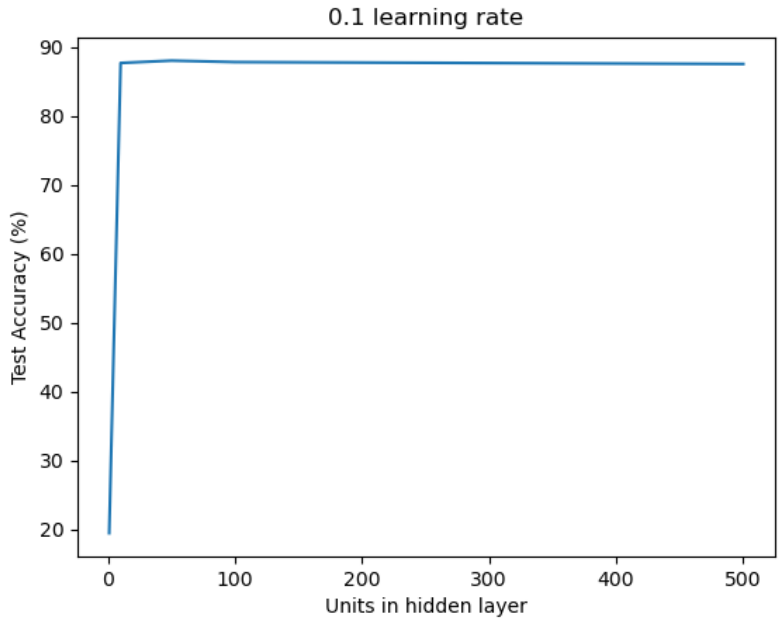





0.001 learning rate
Test Accuracy (%)
70
60
50
40
30
20
0
100
200
300
400
500
Units in hidden layer


0.1 learning rate
Time (s)
2500
2000
1500
1000
500
0
0
100
200
300
400
500
Units in hidden layer





c) For adaptive learning rate, the convergence happened quicker and training time reduced significantly. High accuracies were also achieved. Here ξ was set to 10^{-3}

1 hidden unit
Time to train: 15.262976169586182s
Train accuracy: 23.43166666666667

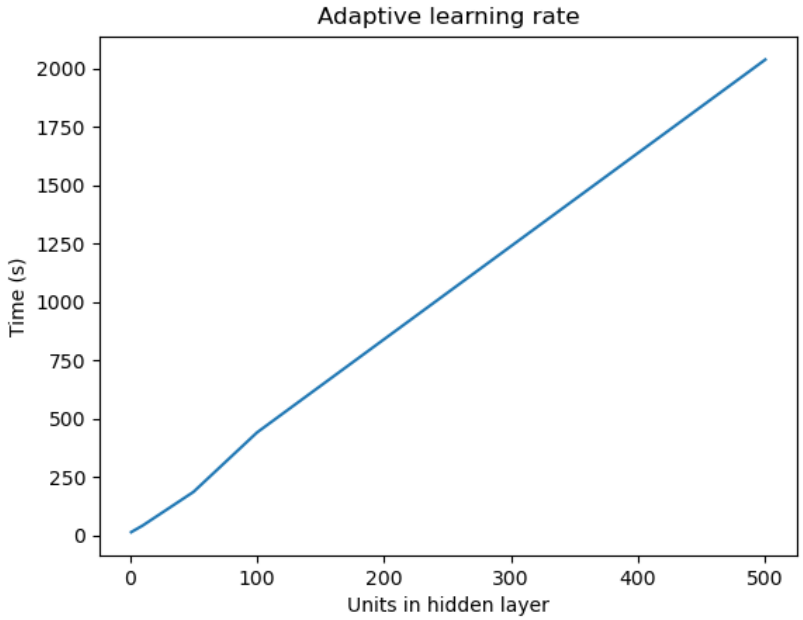
Test accuracy: 23.88

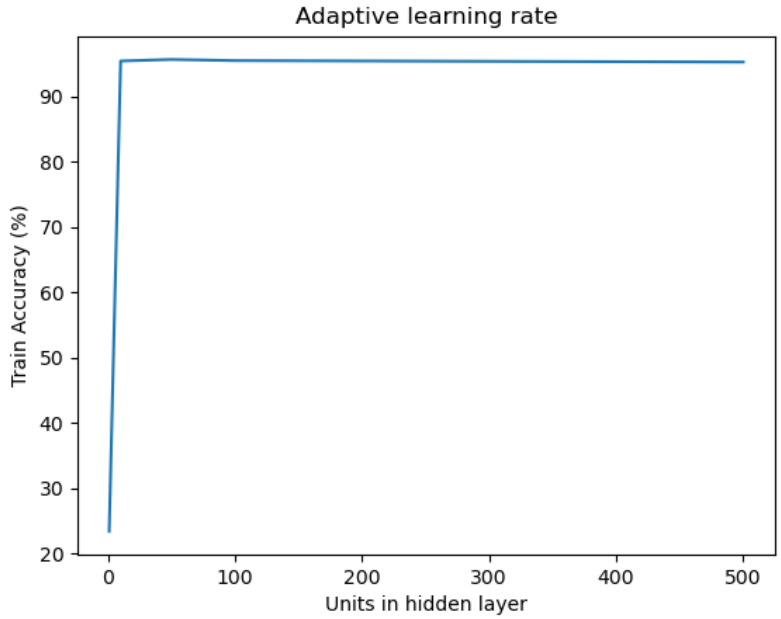
10 hidden units
Time to train: 43.299896001815796s
Train accuracy: 95.37333333333333
Test accuracy: 87.63

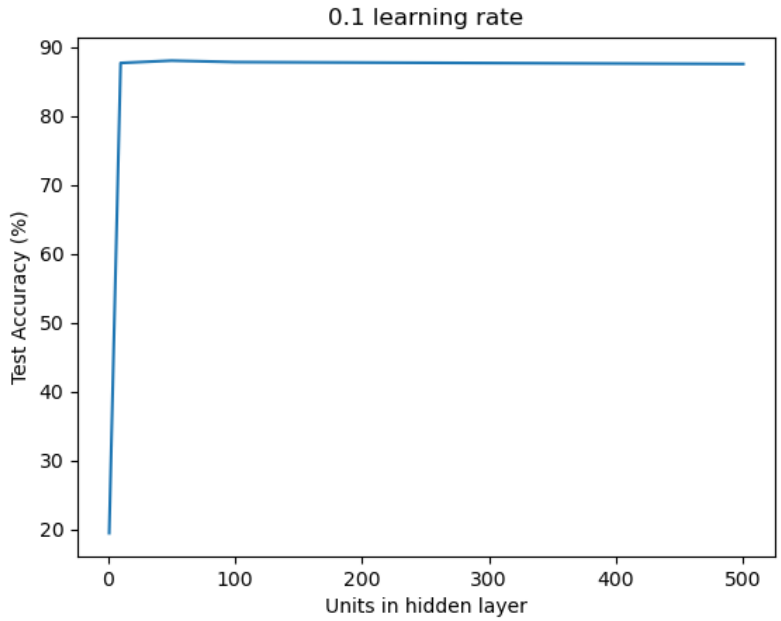
50 hidden units
Time to train: 187.31457471847534s
Train accuracy: 95.60666666666667
Test accuracy: 88.54

100 hidden units
Time to train: 441.55437660217285s
Train accuracy: 95.43333333333334
Test accuracy: 88.24

500 hidden units
Time to train: 2036.9717137813568s
Train accuracy: 95.215
Test accuracy: 88.0







d) (100, 100) hidden layer architecture with Relu activation:
Time to train: 238.3533968925476s
Train accuracy: 97.36333333333333
Test accuracy: 90.9

(100, 100) hidden layer architecture with Sigmoid activation:
Time to train: 575.7539095878601s

Train accuracy: 95.07666666666667
Test accuracy: 87.72

Thus ReLu not only reduced the training time considerably but increased the prediction accuracies as well (compared to using sigmoid in hidden layers, as well as compared to the earlier parts with sigmoid)

e) Using scikit's MLP classifier:
Time to train: 276.08580470085144s
Train accuracy: 99.395
Test accuracy: 93.17999999999

Thus it performs better than our ReLu classifier in part (d), with roughly similar training times.