**COL774 Assignment 4**



Training accuracy: **34.25%**

Testing accuracy: **33.97%**



Training accuracy: **86.46%**

Testing accuracy: **80.84%**

Validation accuracy: **81.23%**



Training accuracy: **80.62%**

Testing accuracy: **77.06%**

Validation accuracy: **75.77%**



In this part, I implemented general Neural Network which takes following parameters as input:

* Training and testing data
* List of layers having perceptron counts
* Batch size for SGD

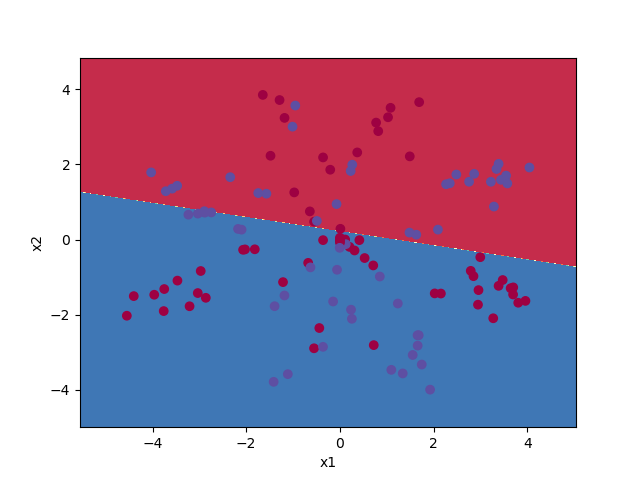
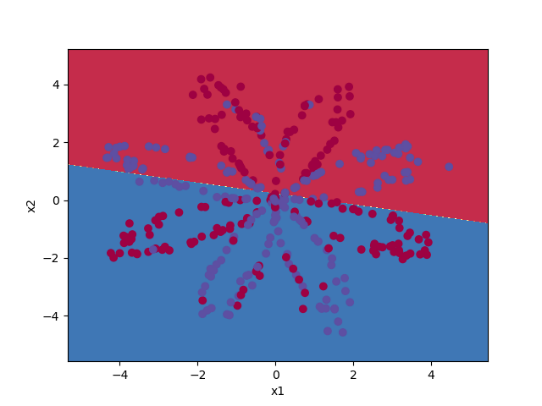


Using sklearn logistic regression learner,

Train set accuracy: **45.79%**

Test set accuracy: **38.33%**

These are the decision boundaries for train and test data respectively:

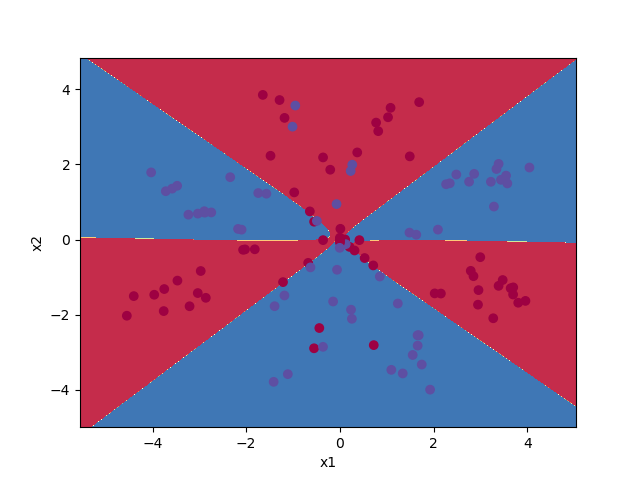
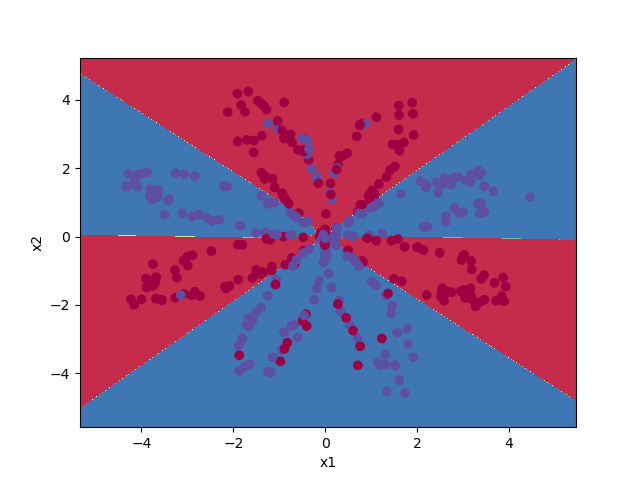
****

Parameters used:

Training accuracy: **89.21%**

Test set accuracy: **87.50%**

These are the decision boundaries for train and test data respectively:



Since logistic regression is a linear classifier and our test data is not linearly separable, so logistic regression has very less accuracy.

On the other hand, neural network also learns non-linear boundaries using activation functions, hence can classify the data better and hence have much better accuracy than logistic classifier.



|  |  |  |  |
| --- | --- | --- | --- |
| **Perceptron Count** | **1** | **2** | **3** |
| **Training Accuracy** | 48.16% | 56.58% | 77.63% |
| **Test Set Accuracy** | 40.83% | 55.83% | 69.17% |
| **Test set decision boundary** | C:\Users\Kapil\perceptroncount1.png | C:\Users\Kapil\Downloads\perceptroncount2.png | C:\Users\Kapil\Downloads\perceptroncount3.png |

|  |  |  |  |
| --- | --- | --- | --- |
| **Perceptron Count** | **10** | **20** | **40** |
| **Training Accuracy** | 89.47% | 88.16% | 88.42% |
| **Test Set Accuracy** | 84.17% | 83.33% | 83.33% |
| **Test set decision boundary** | C:\Users\Kapil\Downloads\perceptroncount10.png | C:\Users\Kapil\Downloads\perceptroncount20.png | C:\Users\Kapil\Downloads\perceptroncount40.png |

The optimal number of units are 10. For 10 units, the accuracy of test set is maximum. The accuracy of the test set is low for less number of units because of underfitting. As the units increase the accuracy increases and reaches maximum at 10 units and after that it starts overfitting the training data so the test set accuracy decreases.

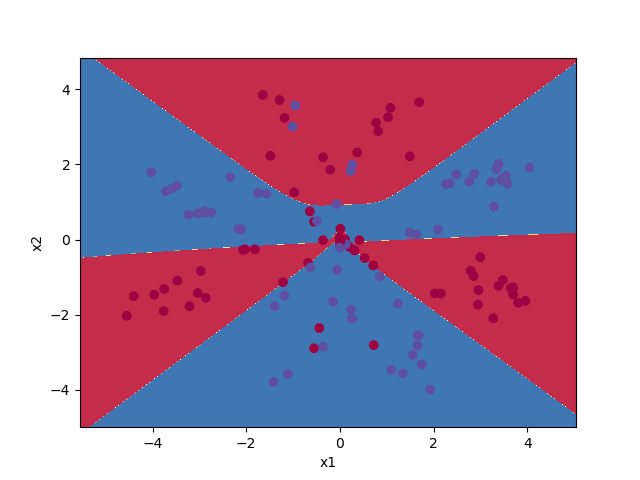


A network with 2 hidden layers each with 5 units.

Training accuracy: **87.63%**

Test set accuracy: **85.83%**

Decision boundary with test samples:



The test set accuracy using 2 hidden layers is better than using a single hidden layer with same number of total units. But they are very close to each other and depend on the type of data. 2 hidden layers can represent an arbitrary decision boundary and can approximate smooth mapping. So, it has slightly better accuracy. It may also decrease when the training data overfits.

1. python NeuralNets.py mnist\_data\MNIST\_train.csv mnist\_data\MNIST\_test.csv 1 100

Using libsvm:

Training accuracy: **100.00%**

Test set accuracy: **98.47%**

Using Neural Network with single perceptron:

Training accuracy: **50.00%**

Test set accuracy: **50.00%**

The accuracies of libsvm are almost perfect whereas accuracy of single perceptron is very low because single perceptron is under fitting the training data. Since, the data has a lot of features, a single perceptron is not able to learn it properly. A network of perceptrons will learn the model better.



Using neural network with single hidden layer and 100 units

Training accuracy: **98.19%**

Test set accuracy: **98.08%**

Stopping criteria is the increase in error for a validation set in successive iterations of SGD. This means, whenever, in successive iterations, the accuracy of validation set decreases, we stop.

There is a significant increase in accuracy compared to previous part. This is because we were earlier using only one perceptron whereas in this case we have 100 units. So, in previous part, model was underfitting that data. And as we increase the number of units, the fitting gets improved.



Using neural network with single hidden layer and 100 units and ReLU as activation unit:

Training accuracy: **97.19%**

Test set accuracy: **97.76%**

The accuracy in this case is not better than using sigmoid as the activation unit. This is possibly because of the problem with ReLU activation unit that when the input of a unit is non positive then it always outputs zero. In other words, that unit becomes dead (dead neuron).