CSCI 635: Introduction to Machine Learning
Homework 4: Nonlinear Prediction

Arjun Ravikumar - ar4038

Question1:

a: Tackling the XOR Problem

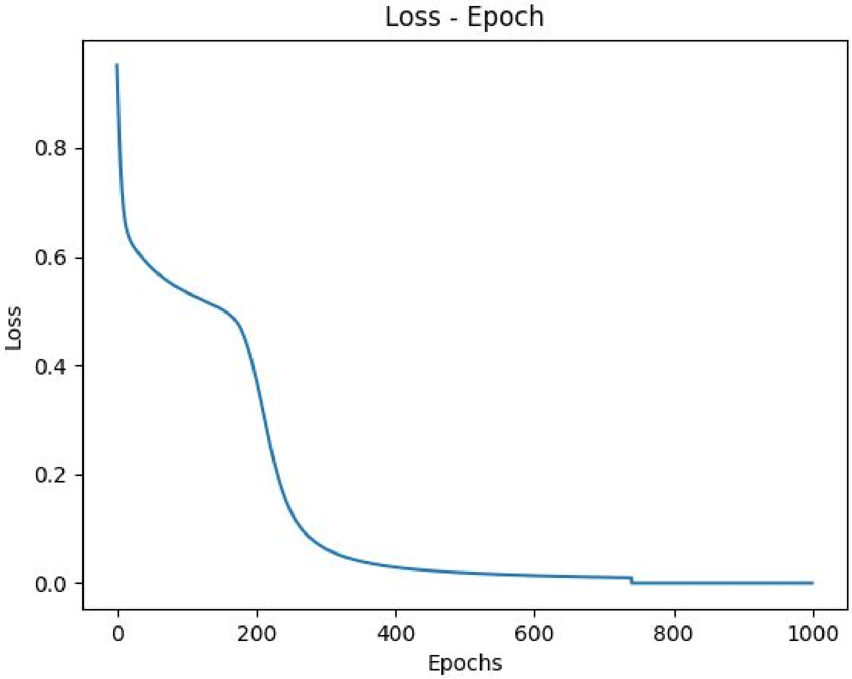
Received 100% accuracy.

Hyper parameters received:

```
Weights layer1:[[-5.17222875  4.83515359]
 [ 6.1091599  -6.03317647]]
Bias layer1:[[-2.6722741 ]
 [-3.53215311]]
Weights layer2:[[-5.04712753 -5.1621159 ]
 [ 6.20767651  5.85798548]]
Bias layer2:[[-0.32247223]
 [-5.7925941 ]]
```

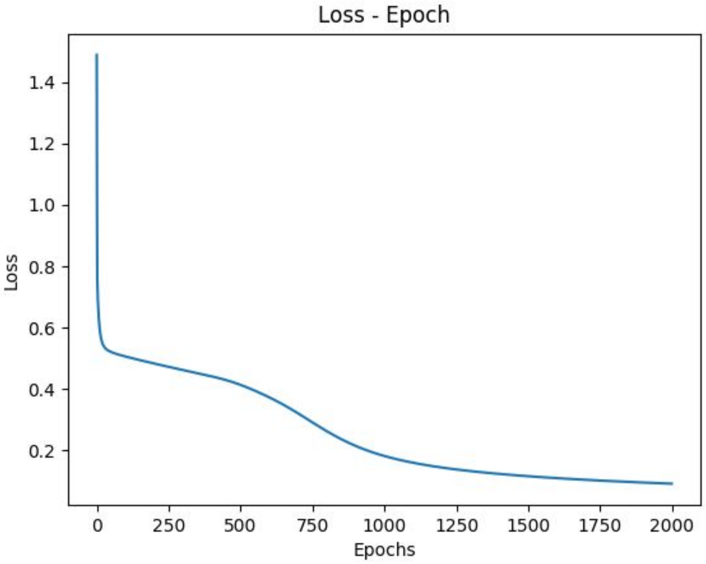
Final Accuracy: 100.0%

As compared to previous HW this model’s performance is much better compared to the model in the previous HW. Hence was easily able to get a 100% accuracy on the data as compared to 50% in the previous HW.

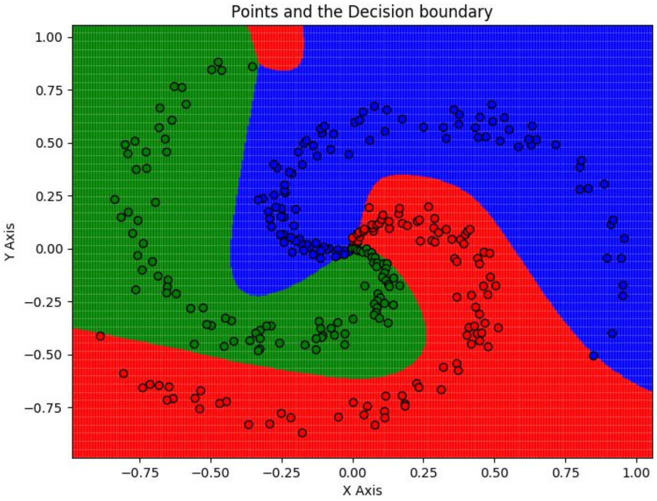


b: Softmax Regression & the Spiral Problem

Loss-Epoch Graph -



Decision boundary-



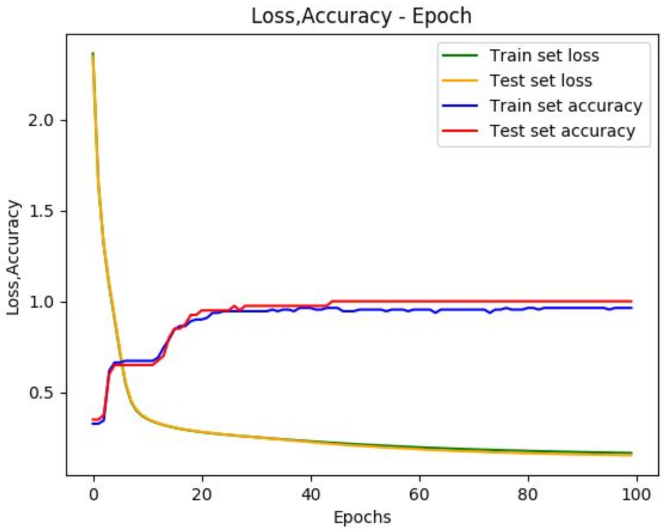
Final Value obtained:
Final Accuracy: 96.33333333333334%
Weights Hidden : [[0.3155565   1.05501139]
 [9.02173582 -7.84976643]

```
 [ 9.52911874 -1.67941488]
 [-2.70754044 -4.82335052]
 [ 2.1343388   9.5523382 ]
 [-8.43367917 -3.97150079]
 [-9.24415913 -3.32119787]
 [ 6.78723105  4.2409275 ]]
Weights Output: [[-0.44264471  0.84039111  0.01033611  4.25785563 -7.81111563
6.44754436
  -5.45587357  6.81670995]
 [ 0.72757517 -7.55103055  7.00826191 -4.43770523  3.81282234 -8.22125621
   3.55752963  0.9649094 ]
 [-2.23379879  7.39690821 -7.02905749 -1.70867085  4.31395338  5.29282536
   1.46590641 -0.54144358]]
Bias Hidden: [[ 3.28916118]
 [ 0.40473812]
 [ 3.75066351]
 [-3.45199974]
 [ 5.57475069]
 [ 3.8586822 ]
 [ 0.28290456]
 [ 0.05295108]]
Bias Output: [[ 2.58663996]
 [ 1.10752542]
 [-2.80815335]]
```

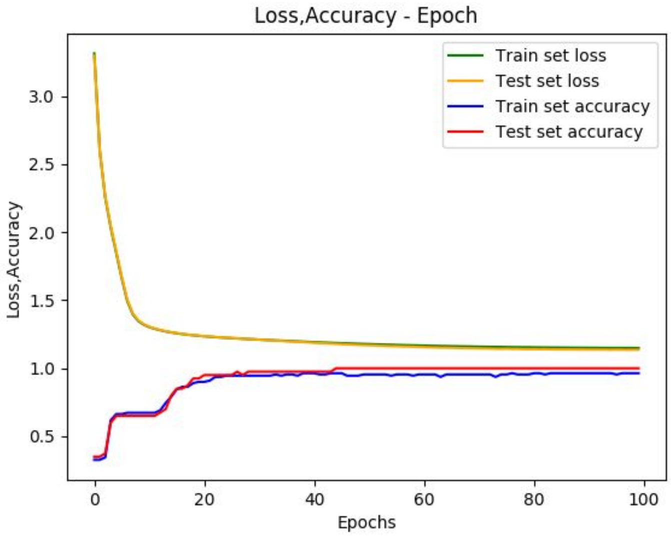
The decision boundary better wraps the images and gives a much better accuracy of 96.33% accuracy as compared to the accuracy of around 55% for the previous HW.

c: The IRIS Maximum Entropy Model

Loss Accuracy Graph before L2 weight decay



Loss Accuracy Graph after L2 weight decay

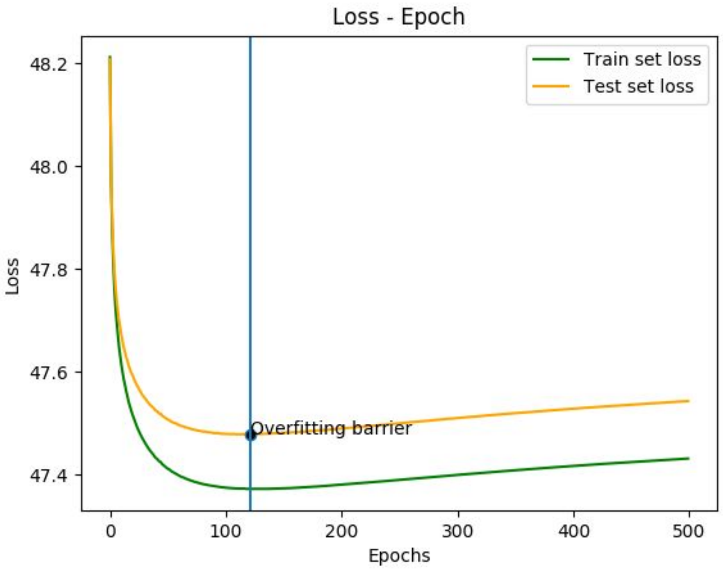


```
Accuracy on Test Data:100.0 %
Loss on Test Data:1.1361061752723411
Weight hidden: [[ 3.24847002 -1.22395629 -1.05560375 -2.14525238]
 [ 1.54056541 -4.71941018  3.43627562 -1.52953745]
 [ 0.13714852 -0.84568272  2.7908013  -4.14187327]
 [-1.14251998 -1.00255106  1.91412497 -2.30457662]
 [-0.34641953 -1.75642598  0.08353637  1.16554255]
 [-2.60821829  1.913447    1.82579189  1.12490877]
 [ 1.79999141 -1.36830835 -0.24691934 -1.87160655]
 [ 0.20119028  1.55889501 -1.25505601 -0.89747367]]
Weight output: [[-4.44198445e-03 -1.92616116e+00 -1.19471221e+00  3.38537826e+00
   1.01542580e-01 -1.27323459e+00  7.60607984e-01  4.61285092e+00]
 [ 6.85013164e-01  1.95326511e+00  1.09795656e+00 -6.49888406e-01
  -2.28475833e+00 -6.56142055e-01  2.68037622e-02  1.55408022e+00]
 [ 8.54042448e-01  1.29424851e+00 -2.26045648e-01  1.71520199e+00
  -1.50900102e+00  2.46243622e+00  2.02490397e-01 -1.38936015e+00]]
Bias hidden: [[-1.37451474]
 [-1.73047895]
 [-1.44292882]
 [-0.11323415]
 [-2.23495371]
 [ 0.33836677]
 [ 3.31923372]
 [ 1.67506703]]
Bias output: [[1.35615584]
 [0.29346648]
 [1.43952932]]
```

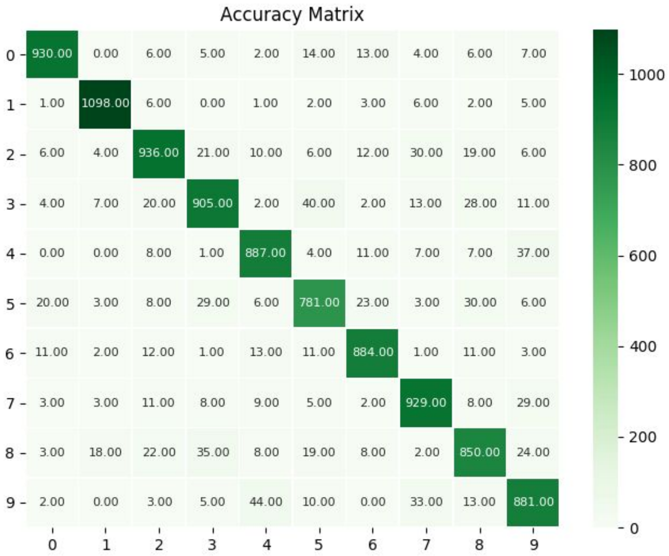
As the number of epochs increase the model tends to over fit the data. I was able to achieve a 100% accuracy on the Test Data on MLP as compared to 65% accuracy on previous HW.

2: Image Categorization with MLPs

Loss - Epoch graph

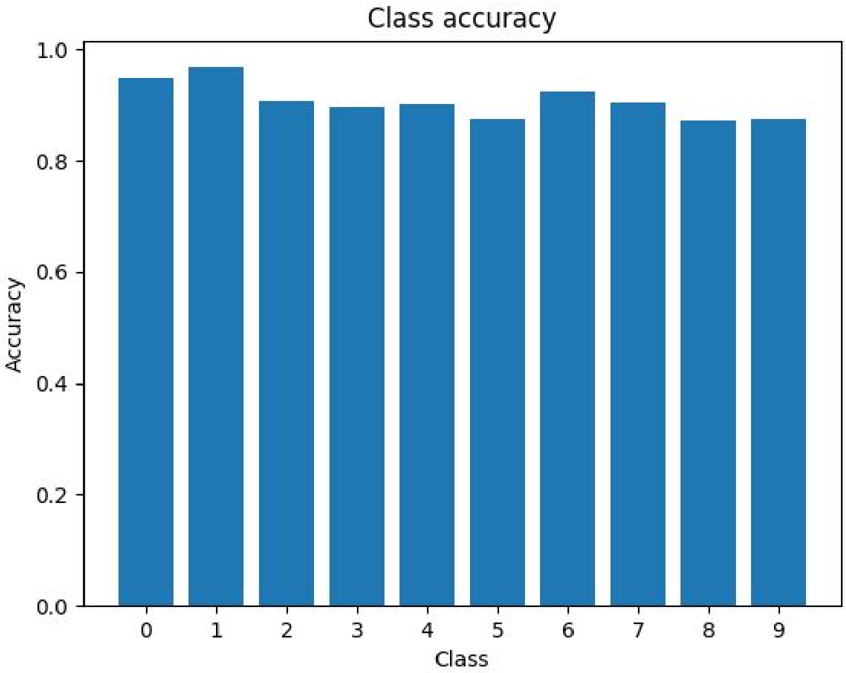


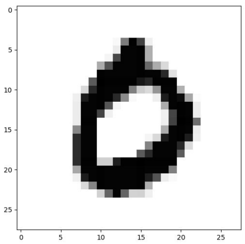
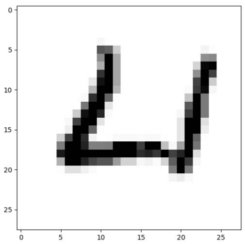
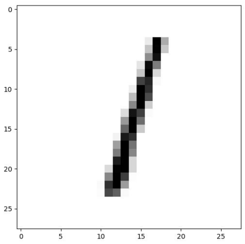
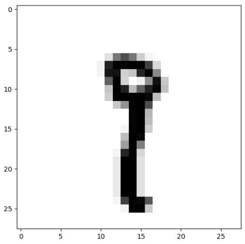
Accuracy Matrix for Validation Data

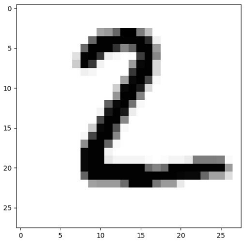
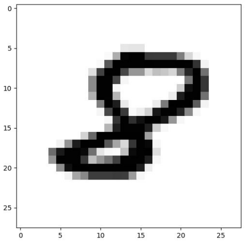
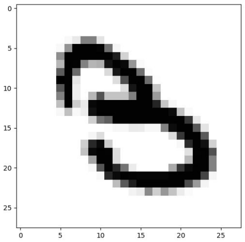
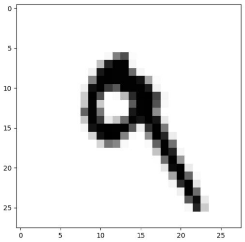
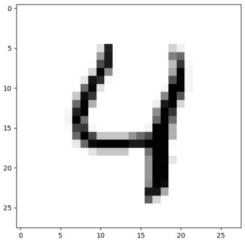
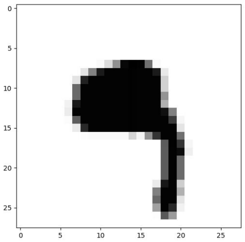
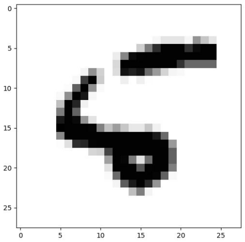
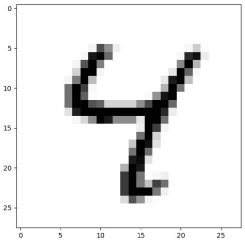
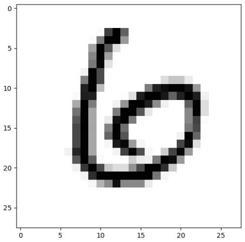
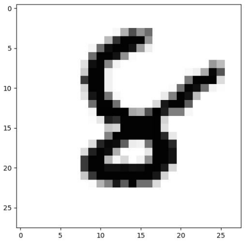


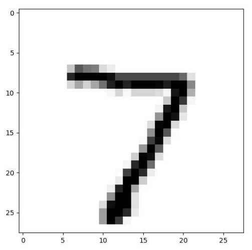
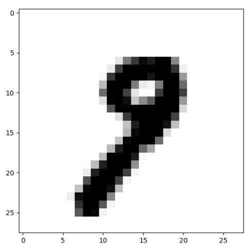
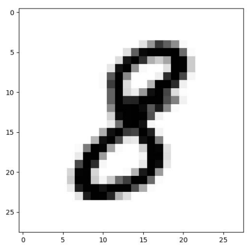
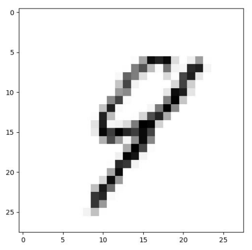
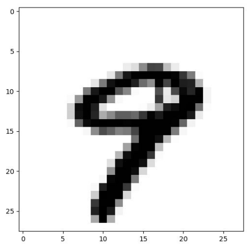
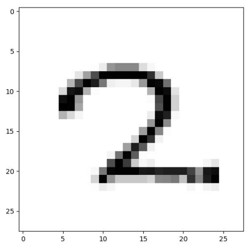
	0	1	2	3	4	5	6	7	8	9
0	930.00	0.00	6.00	5.00	2.00	14.00	13.00	4.00	6.00	7.00
1	1.00	1098.00	6.00	0.00	1.00	2.00	3.00	6.00	2.00	5.00
2	6.00	4.00	936.00	21.00	10.00	6.00	12.00	30.00	19.00	6.00
3	4.00	7.00	20.00	905.00	2.00	40.00	2.00	13.00	28.00	11.00
4	0.00	0.00	8.00	1.00	887.00	4.00	11.00	7.00	7.00	37.00
5	20.00	3.00	8.00	29.00	6.00	781.00	23.00	3.00	30.00	6.00
6	11.00	2.00	12.00	1.00	13.00	11.00	884.00	1.00	11.00	3.00
7	3.00	3.00	11.00	8.00	9.00	5.00	2.00	929.00	8.00	29.00
8	3.00	18.00	22.00	35.00	8.00	19.00	8.00	2.00	850.00	24.00
9	2.00	0.00	3.00	5.00	44.00	10.00	0.00	33.00	13.00	881.00

Accuracy for each class in the Validation dataset



Class	Correct	Incorrect
0		
1		

2	0 5 10 15 20 25	0 5 10 15 20 25
3	0 5 10 15 20 25	0 5 10 15 20 25
4	0 5 10 15 20 25	0 5 10 15 20 25
5	0 5 10 15 20 25	0 5 10 15 20 25
6	0 5 10 15 20 25	0 5 10 15 20 25

7		
8		
9		

The overall final accuracy of the training data was 99.97% after 500 epochs with 300 hidden nodes. The overall accuracy of testing data was 91.55% after 500 epochs. I found that the best epoch for early stopping was 121 as this was the node with the least testing loss after which the phenomenon of overfitting happened. The model was easily able to predict digit one but it had issues with predicting some other digits.

The model had most difficulty in prediction between 4 and 9 also some 6 and 0. The least accurate prediction was of the class 5 as it was getting mixed 3 and 8 a lot. The matrix given above clearly shows the distribution of data. Similarly 8 also got a lot of misclassification as 3 and 5. But overall the model predicted well with the test model which was provided by MNIST with an accuracy of more than 91%.