Homework 3: Nonlinear Prediction

Ketan Kokane
kk7471

December 2019

1 Prob 1

1.1 Q1a

1.1.1 Cost function

Figure 1 shows the cost as a function of Epochs when training on XOR dataset. It can be seen that the cost with MLP pretty quickly converges.

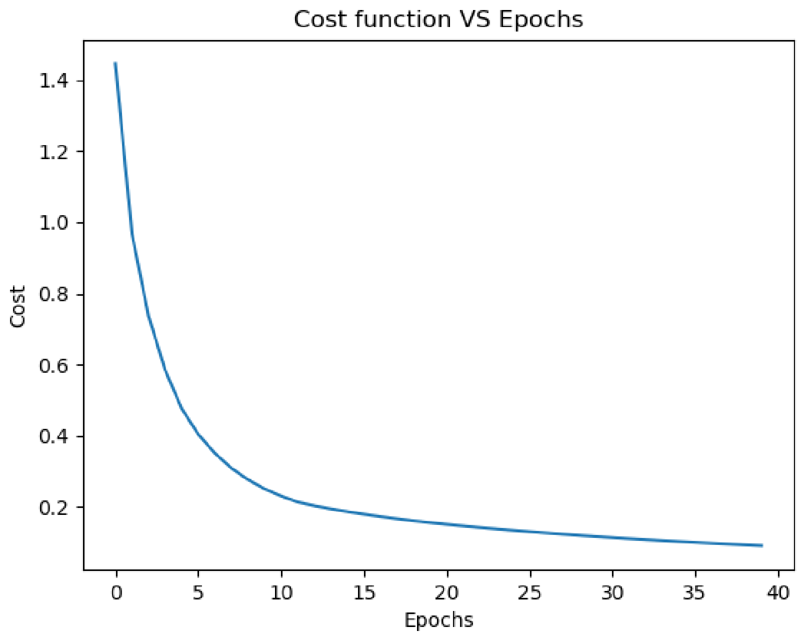


Figure 1: Cost as the function of epochs XOR dataset

1.1.2 Decision boundary

Figure 2 plots the decision boundary generated by the training MLP for the XOR dataset. The feature space is cut up non-linearly to classify the points correctly.

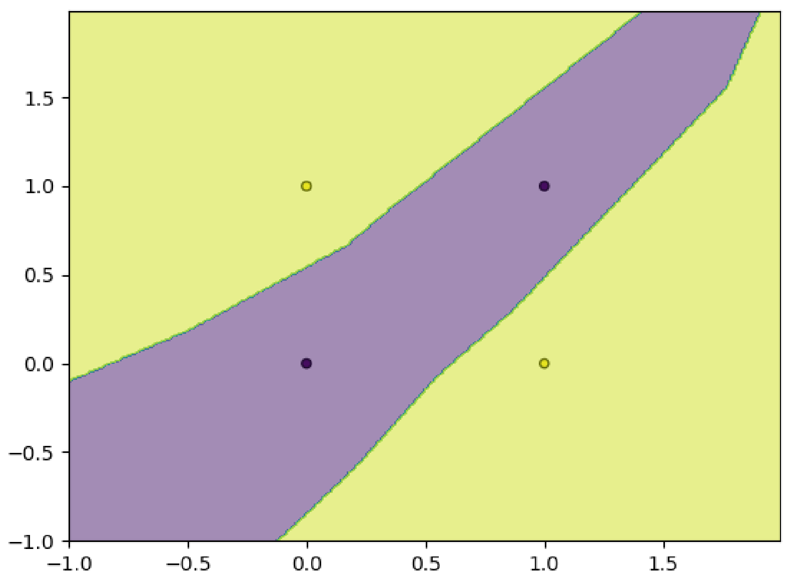


Figure 2: Non linear decision boundary for XOR problem

1.1.3 Gradient checking

Performed gradient checking for Network with 2 input neurons, 2 neurons in the hidden layer and 2 output neurons and tabulated the output of the numerical and analytical gradient in table 1.

Dimensions of the weight and bias matrices:

- w1 = 2 * 2 (weights between input and hidden layer)
- b1 = 1 * 2 (bias weight to hidden layer)
- w2 = 2 * 2 (weights between hiddden output layer)
- b2 = 1 * 2 (bias weight to output layer)

Analytic gradient	Numerical Gradient
-0.003230662630748782	-0.0032306604215406945
0.773361940459516	0.773361952088018
2.0229090559844565	2.022909060962532
0.9107522565070525	0.9107522569616788
2.0229090559844565	2.022909060962532
1.5095620686134126	1.5095620926614506
1.6874243686517687	1.687424371479111
-1.6874243686517687	-1.687424371479111
1.9438135257994205	1.9438135303246948
-1.9438135257985323	-1.9438135303246946
1.6515416365434632	1.6515416680359287
-1.6515416365434632	-1.6515416680359285

Table 1: Gradient checking result

1.1.4 Contrast with model constructed in HW2

The model accuracy of the linear model implemented in HW2 was 50% and accuracy for non-linear model implemented in this HW is 100%. The notable difference is because by using the linear model to classify non-linear feature space was not possible. As the model capacity was low of earlier model to handle.

1.2 Q1b

As seen in the previous homework, the data set (spiral.dat) is not linearly seperable, so implemented MultiLayer perceptron to classify the non-lienar dataset. During training experimented with learning rate ranging from 0.1 to 0.00001, number of hidden nodes in the hidden layer and number of epochs.

Final selected hyperparameters for which the model performed the best are learning rate = $1e^{-3}$, number of hidden nodes in the hidden layer = 12 and number of epochs = 20000.

1.2.1 Cost function vs Epoch

Figure 3 shows the graph of cost as a function of Epoch. The graph shows the cost function started to converge pretty soon. To optimize the performance, calculated the cost after every 50 epochs.

1.2.2 Decision Boundary

Figure 4 shows the decision boundary of the spiral dataset generated by the model. The model splits the data space non linearly and somewhat in spiraling way.

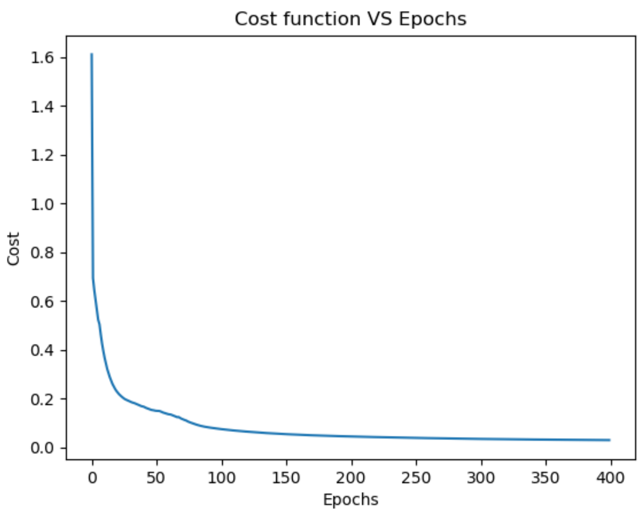


Figure 3: Cost function vs Epoch for MLP trained on Spiral Dataset

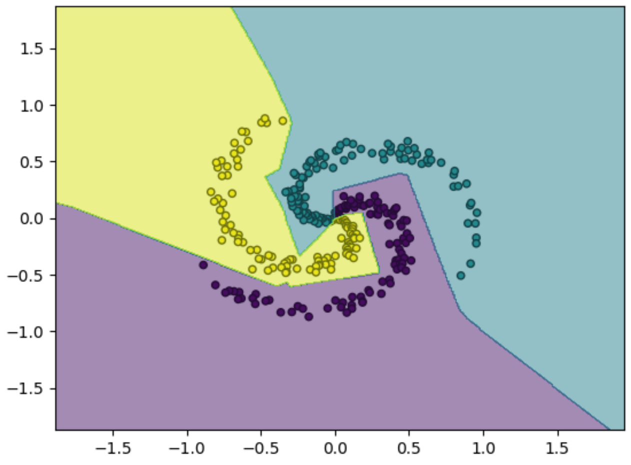


Figure 4: Decision Boundary of spiral data set with MLP

1.2.3 classification report

Table 2 shows the classification report of the model trained on spiral dataset.
Accuracy of the model = 99%.

class label	precision	recall	f1-score	support
0	0.98	1.00	0.99	100
1	1.00	0.99	0.99	100
2	1.00	0.99	0.99	100

Table 2: Classification report of Sprial.dat

1.2.4 Contrast with Linear model trained in HW2

The model accuracy of the linear model implemented in HW2 was 52% and accuracy for non-linear model implemented in this HW is 99%. The notable difference is because by using the linear model to classify non-linear feature space was not possible. As the model capacity was low of earlier model to handle.

1.3 Q1c

Figure 5 shows the cost on training and validation set as a function of epochs. Due to mini batching the cost of training set bounces between 0 and 1 but the cost on the validation set remains constantly low.

Experimented on different batch size ranging from 2 to 8.

1.3.1 Cost as Function of Epoch

Figure 5 shows the cost as the function of Epochs.

1.3.2 Accuracy

Model's accuracy on training set = 99%
Model's accuracy on validation set = 97%

1.3.3 Constract with implementation from HW2

The model accuracy for the both model is pretty much similar.

2 Prob 2

2.1 Cost function

Figure 6 Shows the cost (cross entropy) of training and validation set superimposed on each other as a function of epoch.

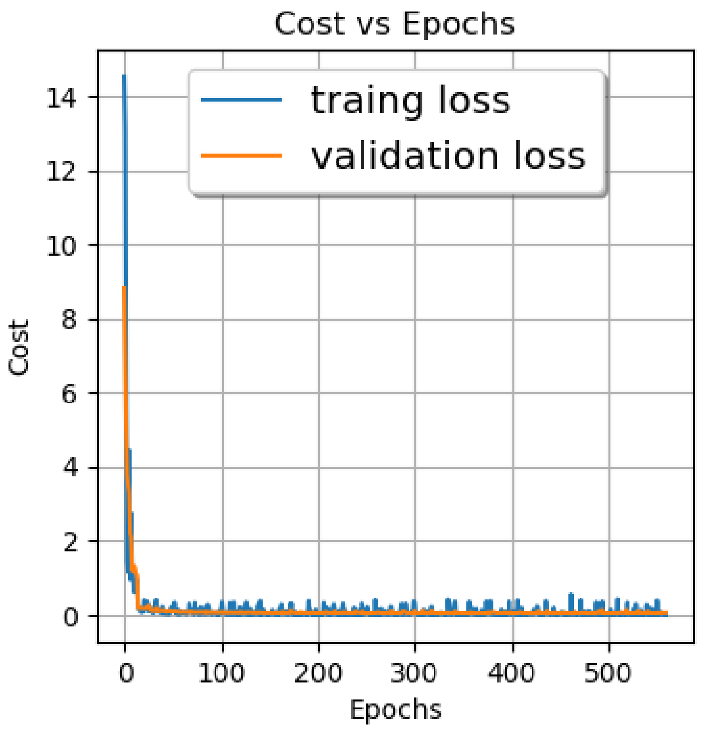


Figure 5: Cost on Training and validation set on IRIS Dataset

2.2 Accuracy score

Accuracy score on Training set: 95%
Accuracy score on Validation set: 92%

2.3 Confusion matrix as Heat map

On careful study of confusion matrix in both training Figure 7 and validation Figure 8 set the digit 2 has the lowest accuracy. My assumption why this is the case is because the images of digit 2 looks similar to digit 5 and 8, thus many of the incorrect digit 2 are mis-classified as digit 5 and 8.

Another important result I witnessed is that non of the instances of digit 1 images are mis-classified as digit 4 and 5 as none of this digit has any similarity. Which to me suggests that the results of the classifiers are comprehensible.

A major issue during the training of the MNIST dataset on already validated MLP implementation was with the softmax function due to data overflow. To mitigate this issue normalized the dataset by dividing each value by 255.

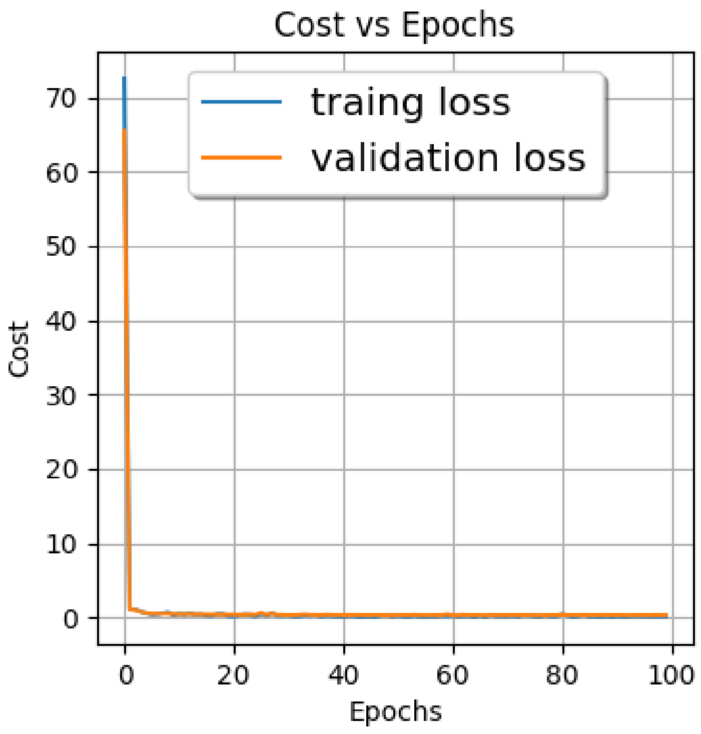


Figure 6: Training and validation cost as function of epoch on MNIST

The phenomena witnessed when the number of epochs are increased is over-fitting when the model starts memorising the noise in data and thus performs exceptionally well on training set and extremely poor on validation set. In this model did not see any over-fitting as the model performed equally better on training and validation set, thus did not experiment much with the regularization strength term.

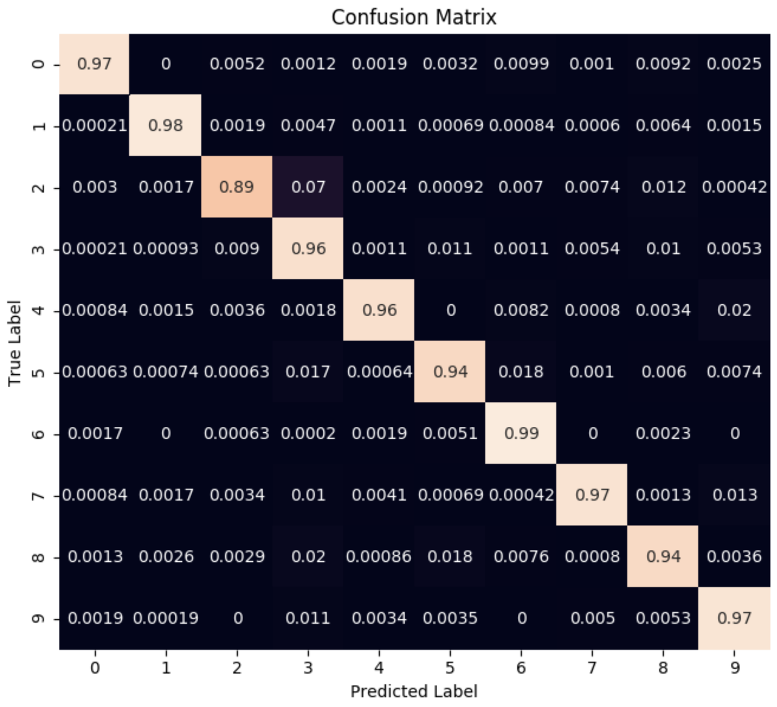
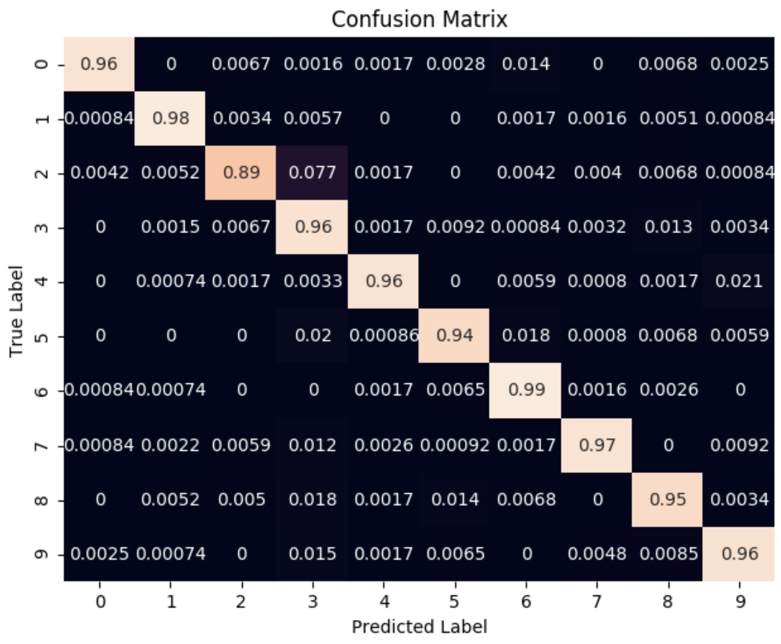


Figure 7: Confusion matrix of training set



	0	1	2	3	4	5	6	7	8	9
0	0.96	0	0.0067	0.0016	0.0017	0.0028	0.014	0	0.0068	0.0025
1	0.00084	0.98	0.0034	0.0057	0	0	0.0017	0.0016	0.0051	0.00084
2	0.0042	0.0052	0.89	0.077	0.0017	0	0.0042	0.004	0.0068	0.00084
3	0	0.0015	0.0067	0.96	0.0017	0.0092	0.00084	0.0032	0.013	0.0034
4	0	0.00074	0.0017	0.0033	0.96	0	0.0059	0.0008	0.0017	0.021
5	0	0	0	0.02	0.00086	0.94	0.018	0.0008	0.0068	0.0059
6	0.00084	0.00074	0	0	0.0017	0.0065	0.99	0.0016	0.0026	0
7	0.00084	0.0022	0.0059	0.012	0.0026	0.00092	0.0017	0.97	0	0.0092
8	0	0.0052	0.005	0.018	0.0017	0.014	0.0068	0	0.95	0.0034
9	0.0025	0.00074	0	0.015	0.0017	0.0065	0	0.0048	0.0085	0.96

Figure 8: Confusion matrix of validation set