Project Report 3

Handwritten Digit Recognition

CS 574 – Fall 2016
Raj Jaysukh Patel:

(UBIT Name: rajjaysu UBIT : 50208278)

1. Introduction

The project is based on classification of Handwritten digits using Logistic Regression, Neural Network with Single hidden layer and Convolution neural network using tensorflow on MNIST dataset. And test that on USPS data.

2. MNIST Data

The data is extracted from mnist.pkl file using pickle library. The dataset is divided into three parts as: Training set with 50,000 images, Validation set with 10,000 images and Test set with 10,000 images. The image has 28 X 28 pixels, which are stored in matrix as 784 features for each row(Image).

3. Logistic Regression

I have used 1-to-K coding scheme, where t = $[t_1, \ldots t_k]$. X for each iteration will be 784 X 1 and have appended 1 for the bias.

The multiclass regression model is represented as follows:

$$p\left(\mathcal{C}_k|\mathbf{x}\right) = y_k\left(\mathbf{x}\right) = \frac{\exp\left(a_k\right)}{\sum_j \exp\left(a_j\right)}$$

Where the activation function is given by $a_k = \mathbf{w}_k^\top \mathbf{x} + b_k.$

I have used stochastic gradient descent to update the weights and ultimately find the optimum weights of features as follows

$$\mathbf{w}_j^{t+1} = \mathbf{w}_j^t - \eta \nabla_{\mathbf{w}_j} E\left(\mathbf{x}\right)$$

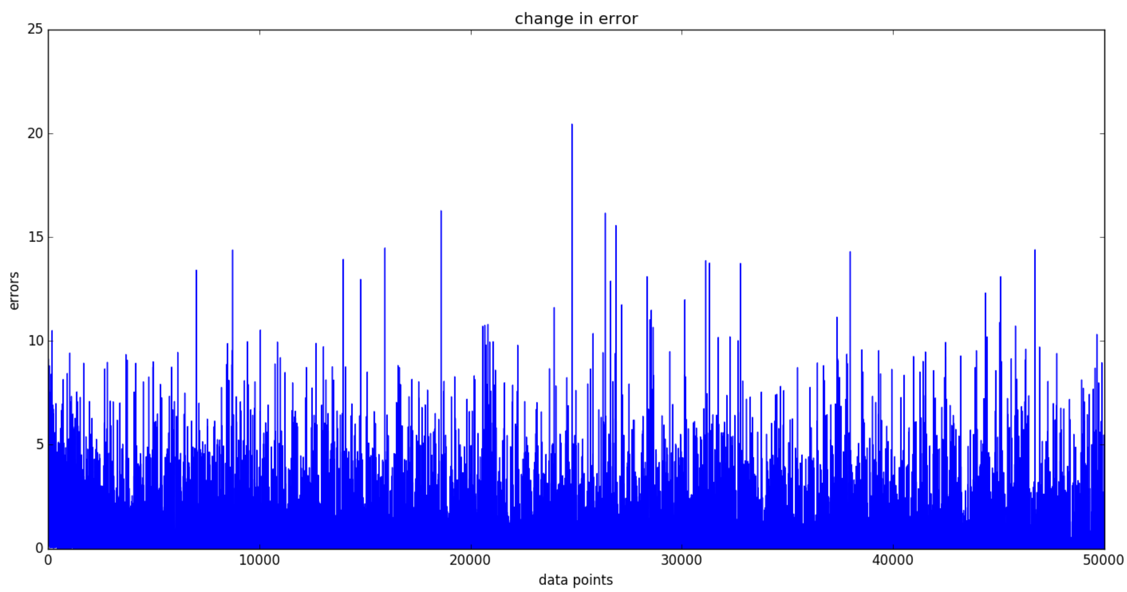
Where the gradient of error is calculated as $\nabla_{\mathbf{w}_j} E\left(\mathbf{x}\right) = \left(y_j - t_j\right)\mathbf{x}$

Here the I have used cross entropy error function which is as follows:

$$E(\mathbf{x}) = -\sum_{k=1}^{K} t_k \ln y_k$$

I have looped over various learning rate (eta) for and trained over validation set to find optimum value.

The error over various images in training set is as follows.



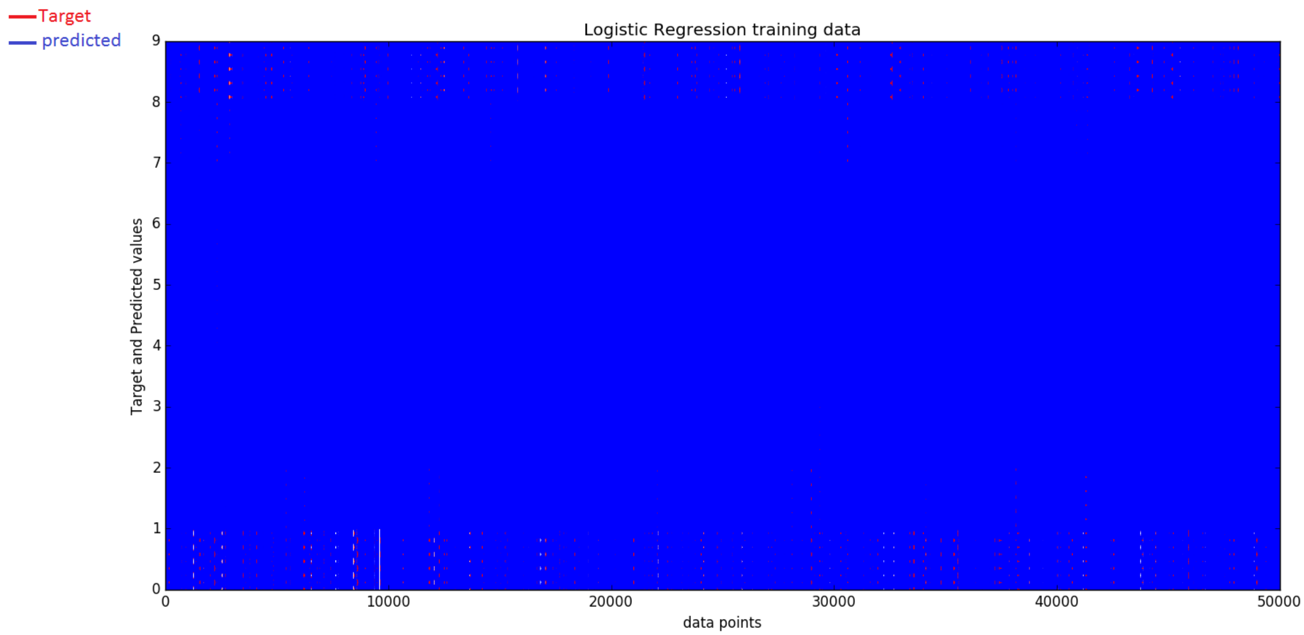
Accuracy of Images classified (In percentage)

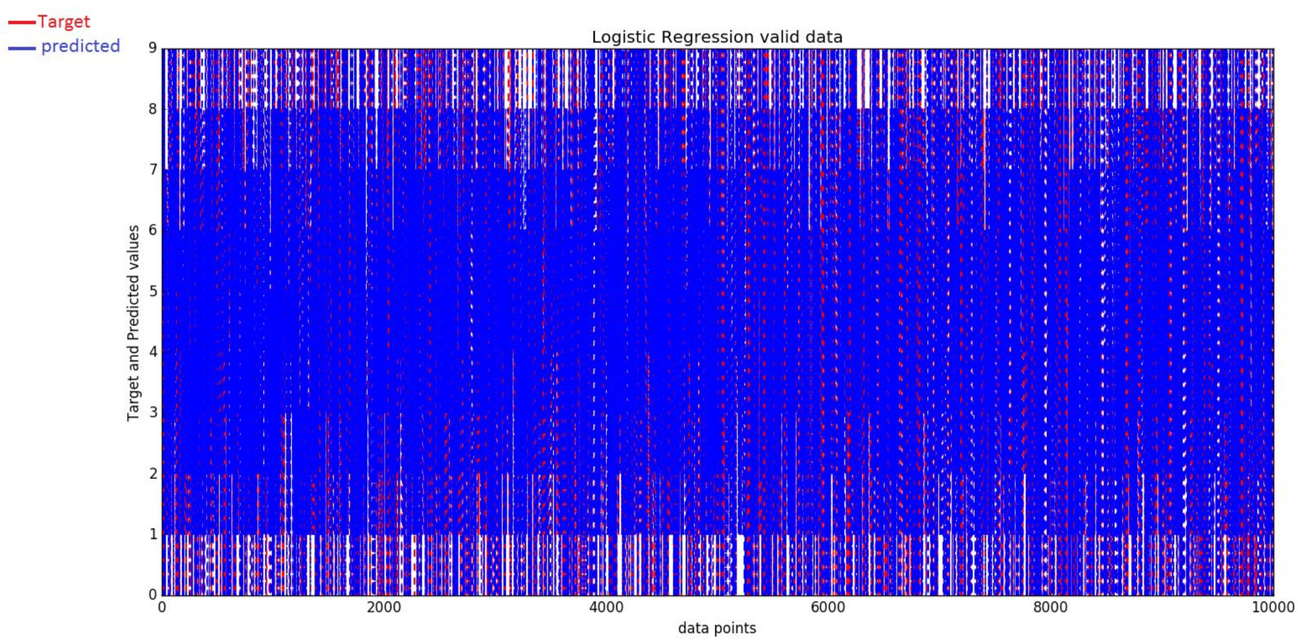
Accuracy of Training Data: 91.26%

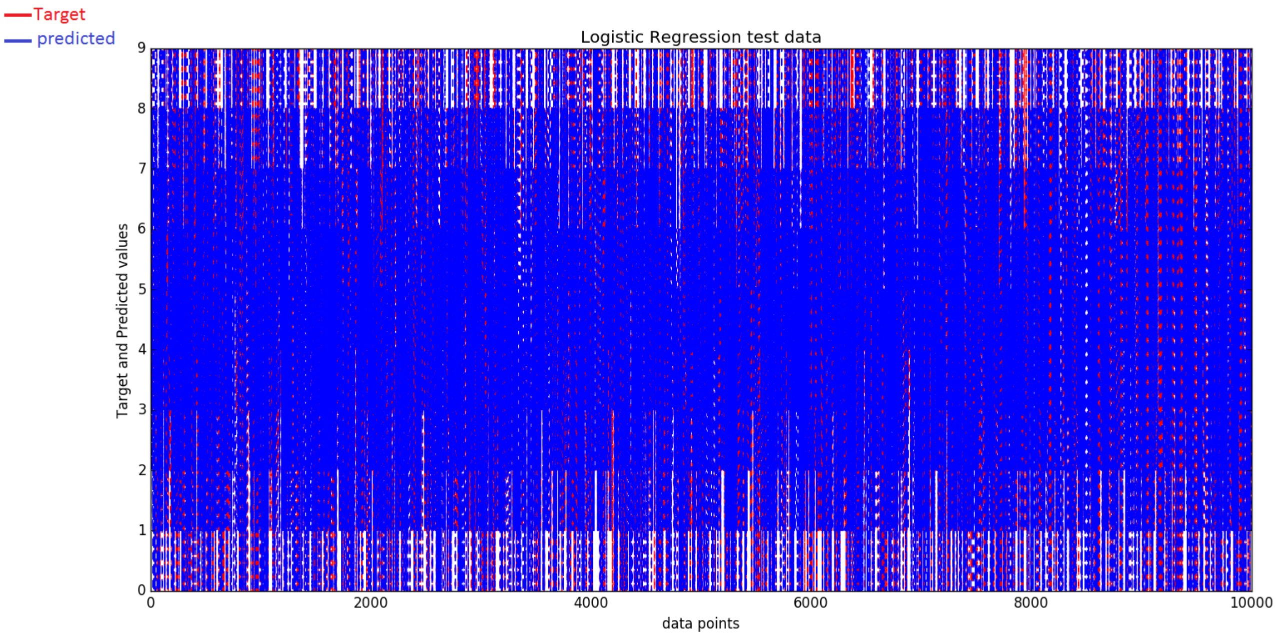
Accuracy of Test Data: 90.57%

Accuracy of Valid Data: 90.43%

The figures below show the difference between original image and predicted images for different image sets.


Target
predicted
Logistic Regression training data
Target and Predicted values
data points
0
1
2
3
4
5
6
7
8
9
10000
20000
30000
40000
50000


Target
predicted
Logistic Regression valid data
Target and Predicted values
data points
0
1
2
3
4
5
6
7
8
9
2000
4000
6000
8000
10000



From the figures above it is clear that training error is less than validation and testing errors as almost every point is overlapping in the graph as compared to other two sets.

4. Single Layer Neural Network

In single layer neural network, I have used one hidden layer. I used feed forward propagation to predict the result as follows:

$$z_j = h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + b_j^{(1)}\right)$$

Z is the activation function for the hidden layer, I used sigmoid function for calculating activation function as follows:

$$\sigma(a) = \frac{1}{1 + \exp(-a)}.$$

And the activation function for output layer is calculated as follows:

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + b_k^{(2)}$$

And the predictions are calculated as follows:

$$y_k = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

I have used stochastic gradient descent to update the weights as follows:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla_{\mathbf{w}} E(\mathbf{x})$$

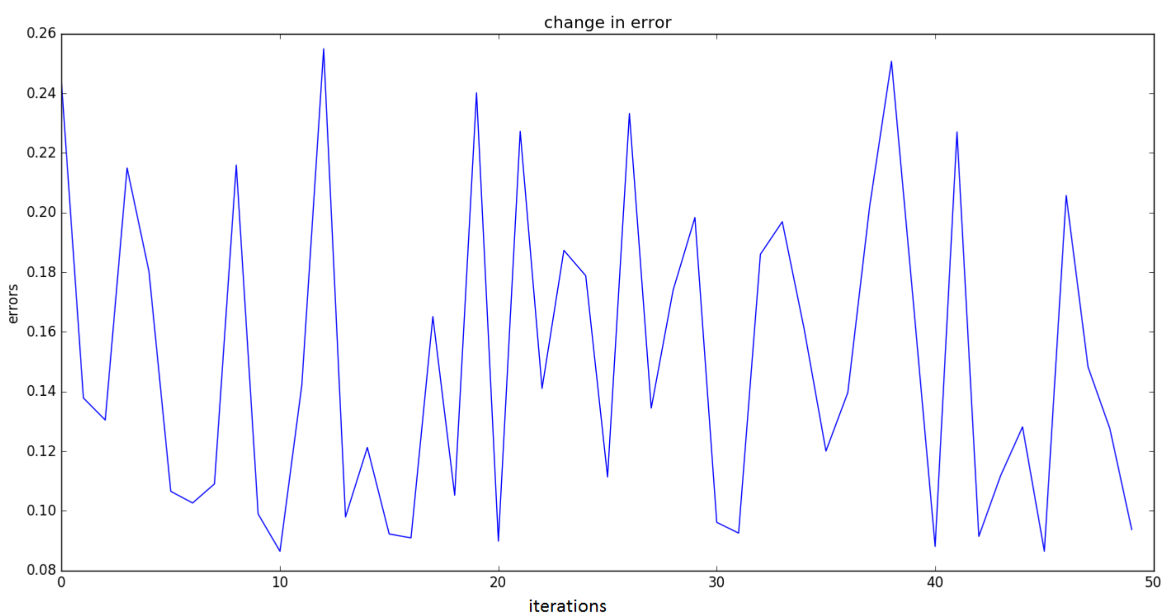
Where the gradient of the error is calculated for different layer as follows:

$$\frac{\partial E}{\partial w_{ji}^{(1)}} = \delta_j x_i, \qquad \frac{\partial E}{\partial w_{kj}^{(2)}} = \delta_k z_j$$

Where

$$\delta_k = y_k - t_k \qquad \delta_j = h'(a_j) \sum_k w_{kj} \delta_k$$

Error over various epochs is as follows:

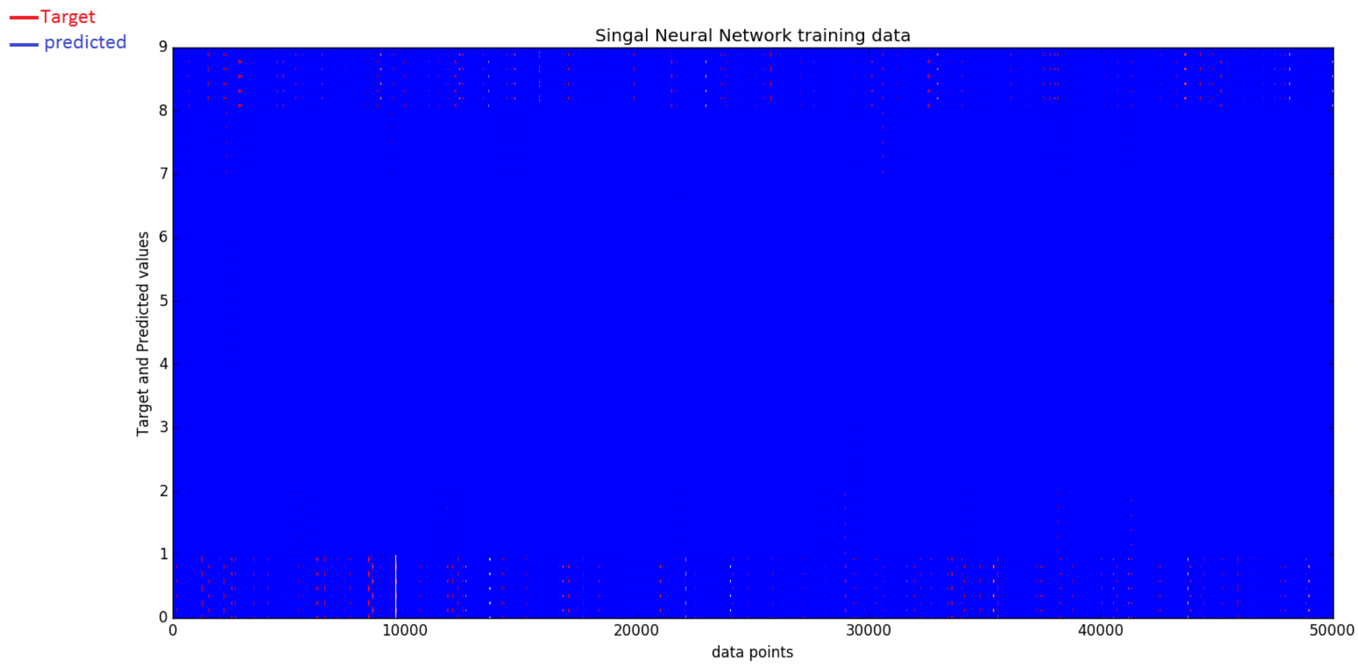


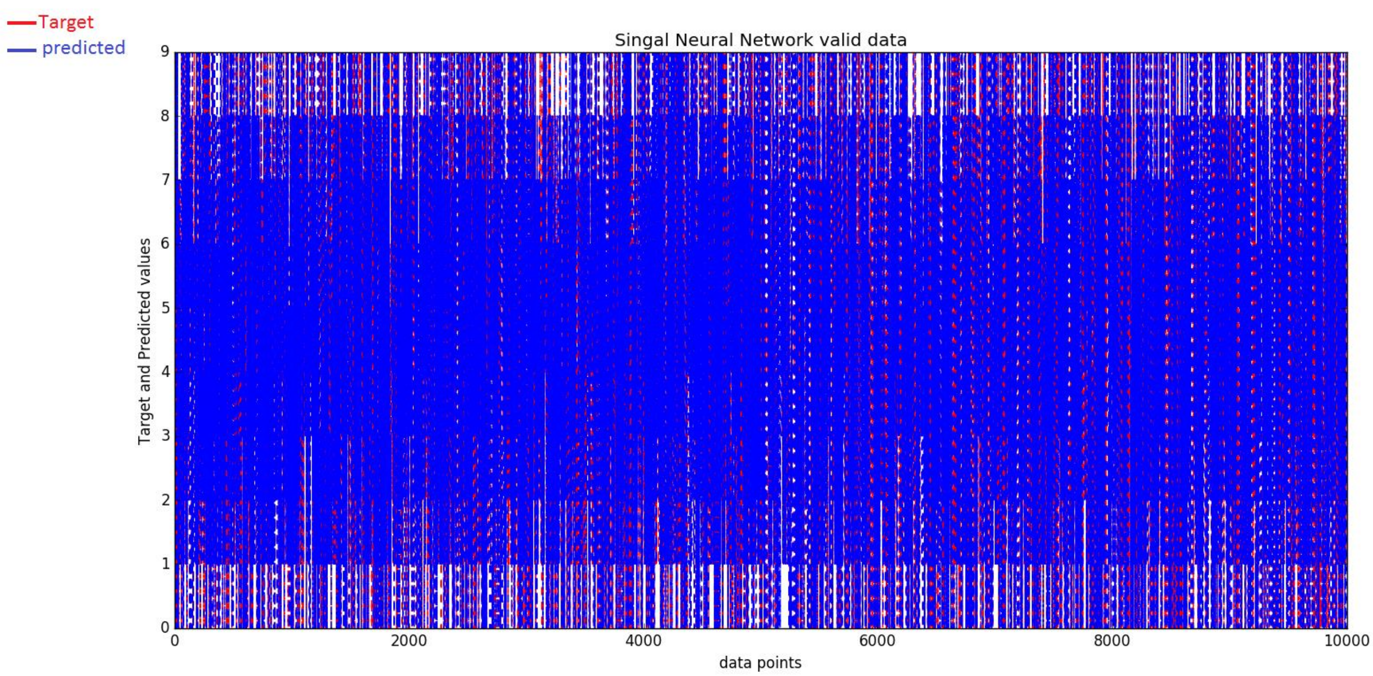
Accuracy of Training Data: 91.882%

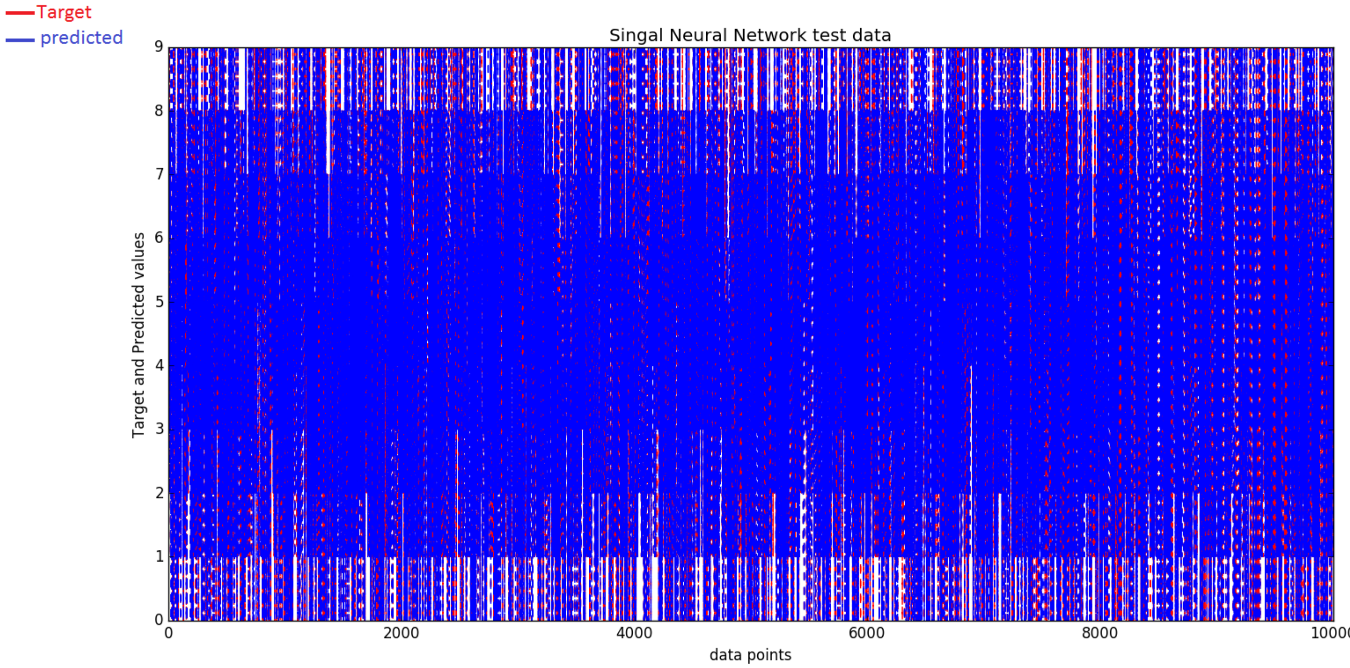
Accuracy of Test Data: 91.2%

Accuracy of Valid Data: 91.15%

The figures below show the difference between original image and predicted images for different image sets.







From the figures above it is clear that training error is less than validation and testing errors as almost every point is overlapping in the graph as compared to other two sets.

5. Convolutional Neural Network

For the convolutional neural network, I have used publicly available open sourced package Tensorflow.

Accuracy of Training Data is ≈ 100

Accuracy of Test Data is ≈ 100

6. USPS Data

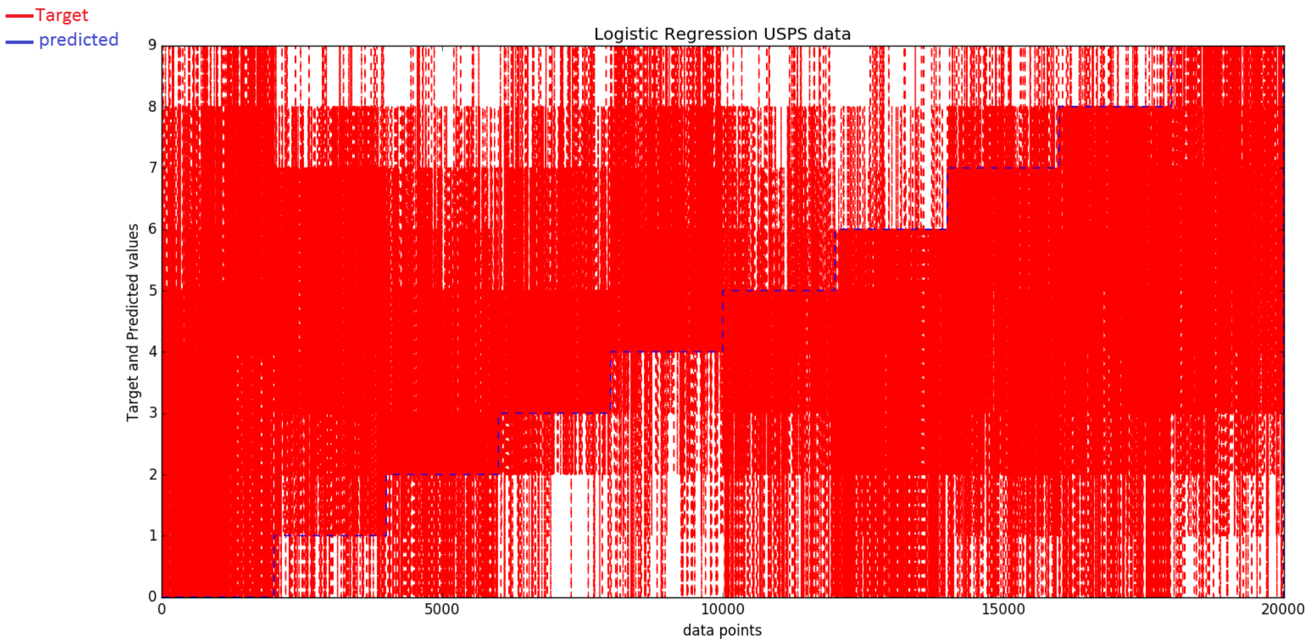
I have used 20000 USPS handwritten digits images as another testing data to test the accuracy of models trained by MNIST data over USPS data set. For that I have rescaled each image of the USPS image to 28 X 28 and normalized by 256. After this transformation I observed that the USPS images and MNIST images were representing the digit in opposite form. MNIST data represented higher value for higher for portion of digits and vice versa and so one subtracted by pixel values of the USPS image.

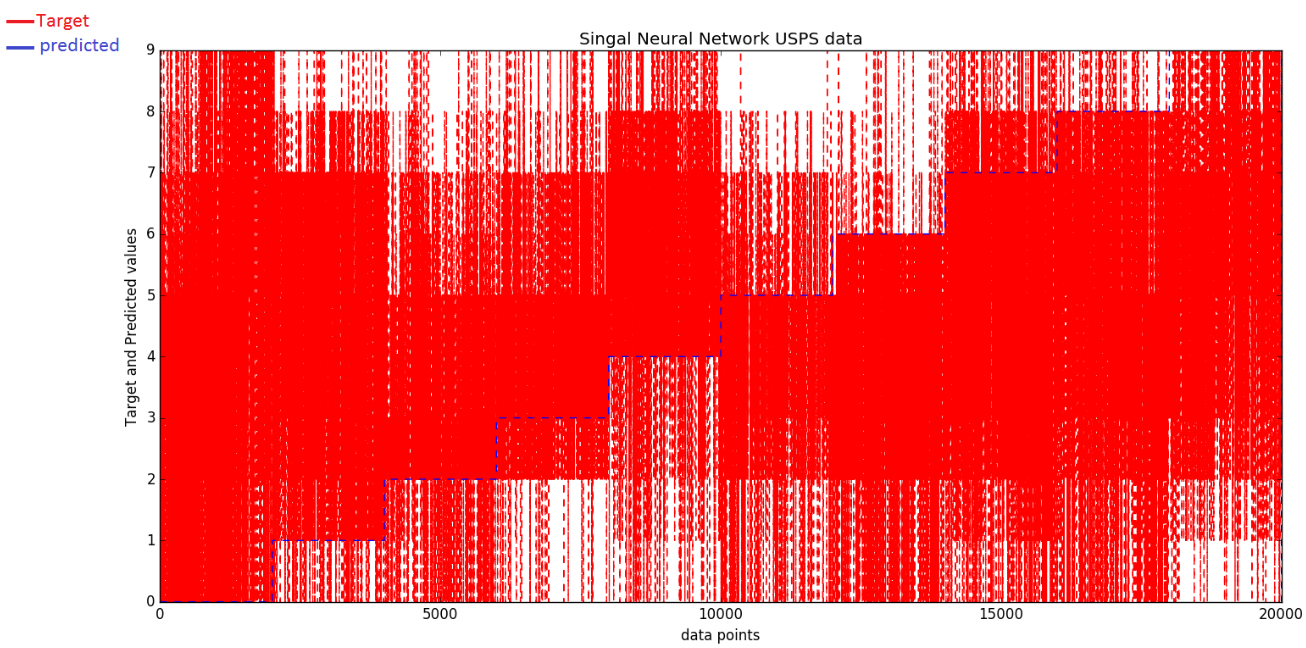
Accuracy for USPS data by Logistic Regression is 33.87%

Accuracy for USPS data Single Layer Neural Network is 34.09%

Accuracy for USPS data by Convolution Neural Network is 61.1%

The figures below show the difference between original image and predicted images for different models.





From the above figures it is clear that there are hardly any overlapping predictions and most of them are wrong predictions. This indicates that models trained on one data set may not give correct predictions on other data sets even though it is giving good predictions on test set of same data set. Therefore, my findings supports “No Free Lunch” Theorem.

7. Conclusion

Different machine learning techniques give different performance. For this project Convolutional Neural Network gave highest accuracy among all. Single Layer Neural Network give better performance than Logistic Regression.

It is also clear that above result supports No Free Lunch Theorem. Because the models trained on one data set may not give correct predictions on other data set.