Handwritten Digits Classification: Multi-Layer Perceptron Neural Network, Naïve Bayes and Hybrid Bayes-Neural Network

Avinav Sharan, Himanshu Sharma, Aayush Uppal
CSE 574

1. Introduction

Classifiers for MNIST handwritten digits dataset: Multi-layer perceptron neural network, naive bayes classifier and a hybrid-model of naive bayes and mlp-nn: "bayes-neural network" are implemented, analyzed and compared. Effects of various hyper-parameters of the neural network on its accuracy are also observed.

We have used each pixel of the input image as an input to neural network. A single hidden layer model with sigmoid units is used. The weights of neural network are learnt using back-propagation algorithm and regularization is implemented to prevent over-fitting.

2. Dataset

MNIST dataset of handwritten digits consists of 60000 sample images for training and 10000 sample images for testing with digits 0 to 9. Each image is 28x28 gray-scale image (represented as 1x784 row vector) with values between 0 and 255.

3. Approach

3.1. Pre-Processing and Feature Selection

For every image in dataset, each pixel value is normalized. 60000 training samples are partitioned into 50000 training samples and 10000 validation samples.

We have implemented two procedures to remove non-informative pixels from the image. We can observe that the in the image of 28x28 informative pixels are only present in the center. Therefore, we have constructed global_min_x and global_max_x vertical lines in the 28x28 image and removed the pixels present on the left of global_min_x and on the right of global_max_x. global_min_x represents minimum value of x-axis of image, where pixel(x,y) = black. Similarly, global_max_x is constructed for max value for x. The minimum/maximum value are global in a sense that it is unique for the complete dataset.

3.2. Standard Deviation Approach

In order to efficiently reduce running time ensuring no loss of features we employed a standard deviation metric across column computation. In the first step we reshaped the input image to 28*28 image and then compressed using pan column mean value. This was done recursively over 10 random images each from 0 to 9. Now across the mean values for 10 image samples we calculate standard deviation. Furthermore this is done over 10 separate iterations and the mean of all standard deviations is used to select usable columns. The global minimum and global maximum value are set using a FindLimits function and then finally resized using a transformation for pixel values of corresponding columns with standard deviation above threshold.

The number of features are reduced to approximately 600 by global_min and global_max by standard deviation method.

3.3. Classification

Multilayer Perceptron Neural Network

The neural network implemented consists of one hidden layer and sigmoid units at hidden layer and output nodes.

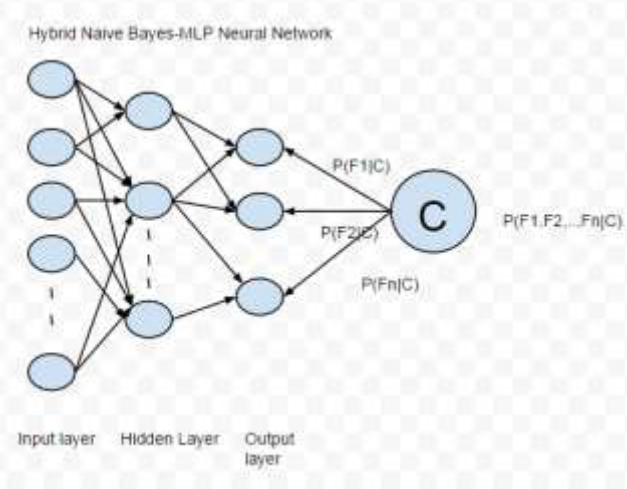
Features are extracted from input image and is fed to neural network. Number of hidden nodes used are in the range of [20,60].

Naive Bayes

For Naive Bayes classifier, the normalized image is further converted to binary by thresholding on 0.3 value. Each image pixel is considered as a feature and each label of the digit is considered as a class. Conditional probability tables for each feature is calculated by counting and normalizing what each feature takes given a class P(F1,F2,...,Fn|C). Prior of class is considered as uniform and therefore ignored. Therefore, probability of class given a set of features P(C|F1,F2..Fn) is assumed to be directly related only to P(F1,F2,...,Fn|C). Log-likelihood values are used as the product of probabilities diminish. Training Data is used to learn the CPTs.

Hybrid Model: Bayes-Neural Network

The network is built on the motivation that intermediate layers or output layer of the neural network can be considered as the features extracted of the input. These features are then considered impendent nodes of naive bayesian network (as shown in figure). The nodes of the neural network can be considered as a continuous random variables which we have discretized to a R.V, [0,1,...9]. The continuous values lie in the range of -infinity to +infinity (before sigmoid), is translated by the minimum value of feature vector of image (to remove negative values), normalized, scaled to 10, and then floor of the value.

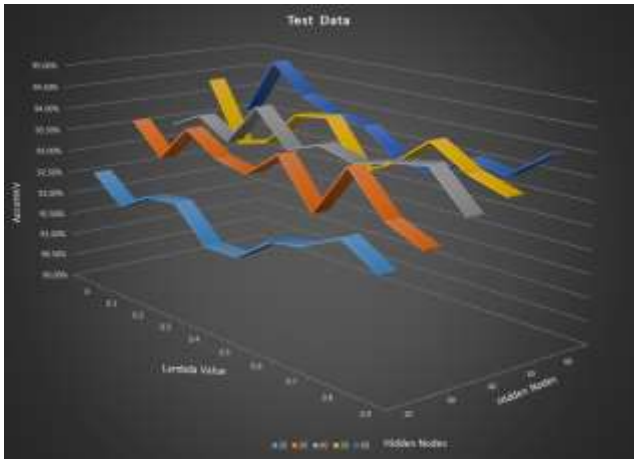


4. Analysis and Results

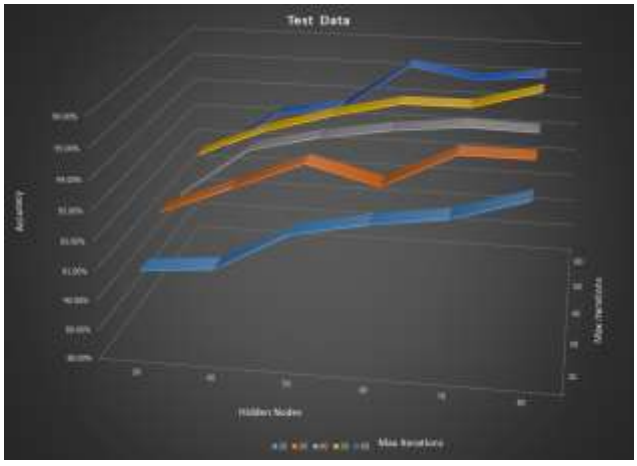
Color For multi-layer perceptron neural network, we analyzed accuracy and time complexity of the network by varying number of hidden nodes, regularization parameter and maximum number of iterations for converging algorithm. Maximum accuracy of the classifier is achieved at 96.45% on test and 97.53% on training. We can observe from the plots that accuracy increases with the increase in number of hidden nodes. But after a certain value of hidden layer nodes, accuracy decreases. This observation comes from the fact that hidden layer nodes can be considered as features of the image and **as we increase the number of features, accuracy increases**, till a point when they start to get redundant. Regularization Parameter does not affect significantly, as there is no significant over-fitting of the data. **Increase in the number of iterations** of the converging algorithm increases the accuracy as the gradient error is more converged. We also observe that **sigmoid unit is a critical part of the network**, as the neural network is **not able to classify with hyper-planes** and gives an accuracy of only 9% on test data.

We can observe from the confusion matrix that the network has major confusions in classifying “5”,”4” and “7”. “5” is confused with 3 (3.92%), “4” is confused with “9” (3.05%) and "7" with "9" (2.43%). We can intuitively think about the confusions due to the symmetry of the structures.

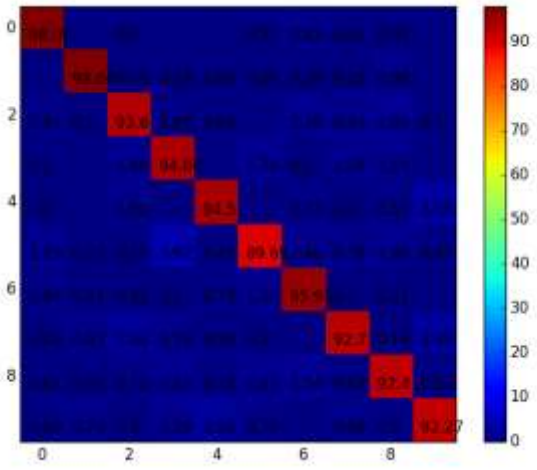
Naive Bayes classifier gives an accuracy of 84% on test and train data. We can observe from the confusion matrix that the major misclassifications are "4" with "9" (13%) , "5" with "3" (13%) and "8" with "3" (7.5%).



Test Data / Hidden Nodes / Lambda Value



Test Data / Hidden Nodes / Maximum Iterations



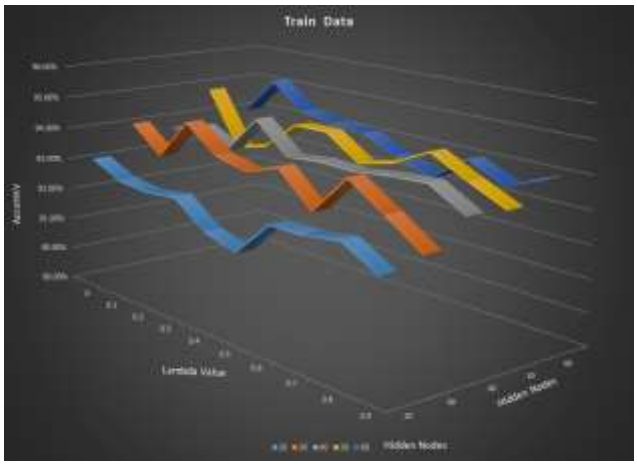
Confusion matrix for neural network

The bayes-neural network increases the accuracy of the naive bayes network from 84% to 94% on training data and 84% to 90% on test data. The connection with output node is found to be more efficient than at hidden nodes (90% and 85%). The hybrid model is found to be less efficient than the multi-layer perceptron neural network.

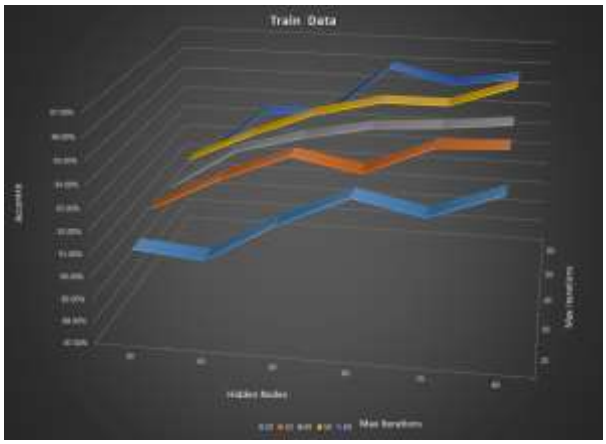
5. Conclusion

Major confusions for the classifiers are analyzed. **Multi-layer perceptron neural network is found to be most efficient for the given data with 96.45% on test data and 97.53% on train data with 80 hidden nodes, 0.1 as regularization parameter and 100 as maximum number of iterations**. **Higher value of Regularization parameter is used when accuracy on training data is much higher compared to validation data, thus overcoming overfitting. We observe that the hyper-parameter values can be determined from empirical observation, we find that the number of hidden nodes should be limited between number of output and number of input nodes and its optimal value can be determined by iterative approach.** The hybrid model is found to be less efficient than MLP-NN with 90% efficiency on test data, but it needs to be further analyzed with noisy dataset and considering continuous random variables.

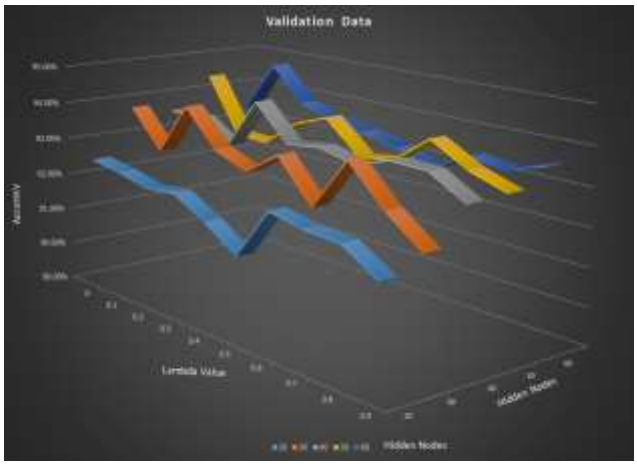
Graphical Analysis



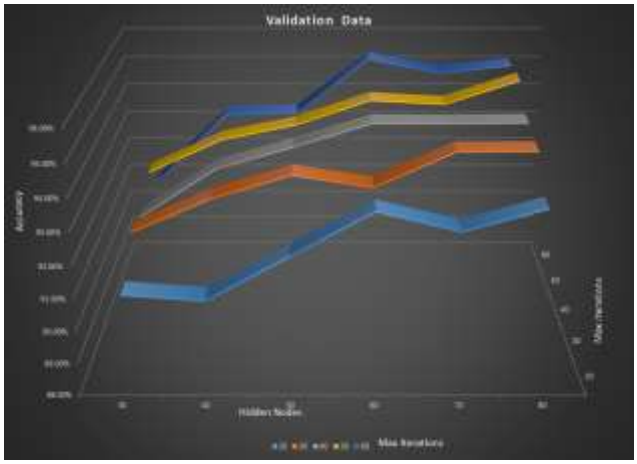
Training Data / Hidden Nodes / Lambda Value



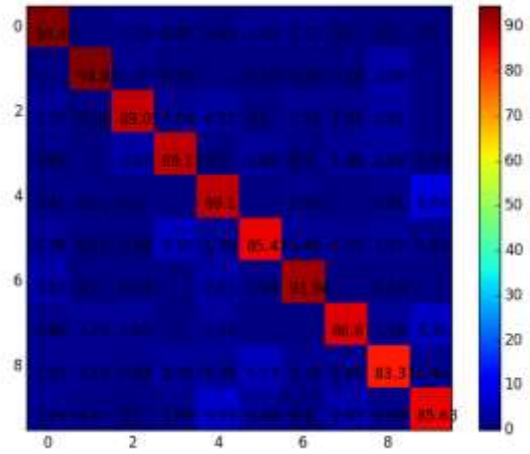
Training Data / Hidden Nodes / Maximum Iterations



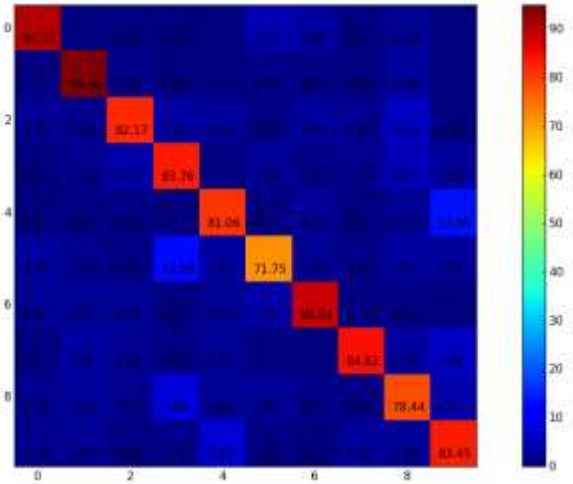
Validation Data / Hidden Nodes / Lambda Value



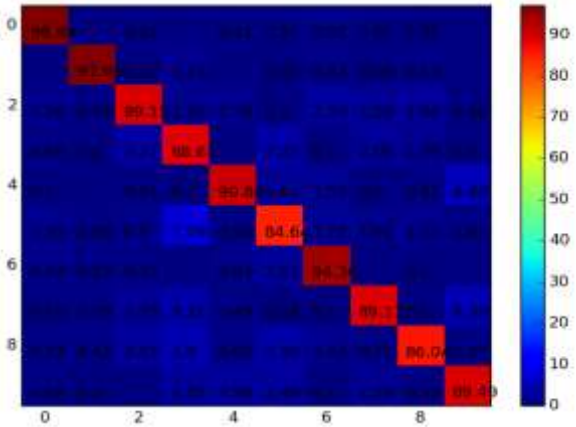
Validation Data / Hidden Nodes / Maximum Iterations



Confusion Matrix for hybrid model Neural Network output from hidden layer



Confusion Matrix for Naïve Bayes



Confusion Matrix for hybrid model Neural Network output from output