CSE 574 - Programming Assignment 3: Project Report (Team 50)

Utkarsh Kumar
50419745
utkarshk@buffalo.edu

Abhishek Kumar
50419133
akumar58@buffalo.edu

1 Introduction

In this assignment classification has been performed on the MNIST digit dataset using variants of logistic regression and SVM and performance has been evaluated in terms of classification accuracy. For logistic, two variations have been used - a) One vs all strategy & b) softmax multi-class logistic regression. For SVM classifier we have varied the kernel type, kernel coefficient gamma and regularization parameter C to find the optimum model on a smaller database, followed by application to the entire training set.
Preprocessing: The MNIST dataset has been divided into 50,000 training samples, 10,000 validation & test samples each. Using feature selection the data set has retained 715 features for training and testing. We evaluate model accuracy on the three data sets to evaluate a model.

2 Logistic Regression: One vs all strategy

As logistic regression is a binary classifier, to classify multiple classes we opt for one against all strategy where we consider one class at a time and train against it as true or 1, where as every other class is regarded as false or 0. In this experiment we evaluate class-wise accuracy and overall model accuracy. Table 1 shows class-wise model accuracy

Table 1: Logistic 1vall: Class Wise Accuracy(%)

Class	Train	Test
0	97	98
1	97	98
2	90	89
3	89	91
4	93	93
5	88	85
6	96	94
7	94	92
8	87	87
9	89	89

For certain classes, the problem of classification is non-linear, for example classes: [2,3,5,8,9]. Figures 1,2,3&4 show some of the miss-classification examples. Table 2 summarizes overall accuracy for the model.

Overall, the training, validation & test error are very close to each other which shows a robust model. For a class like [5], as the training accuracy is low, we also find test accuracy to be still lower. The next step in optimizing the classifier is to consider a softmax multi-class approach which will adjust weights for all classes simultaneously.

Table 2: Overall Accuracy(%)

Train	Validation	Test
92.7	91.5	92

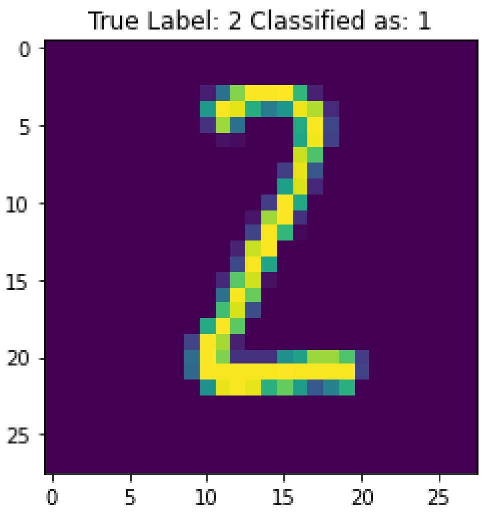

Figure 1:

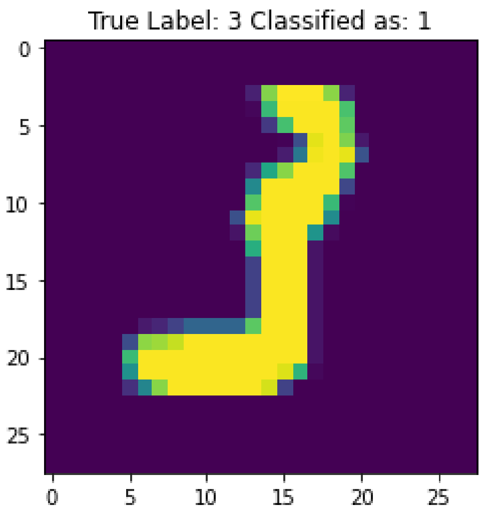

Figure 2:

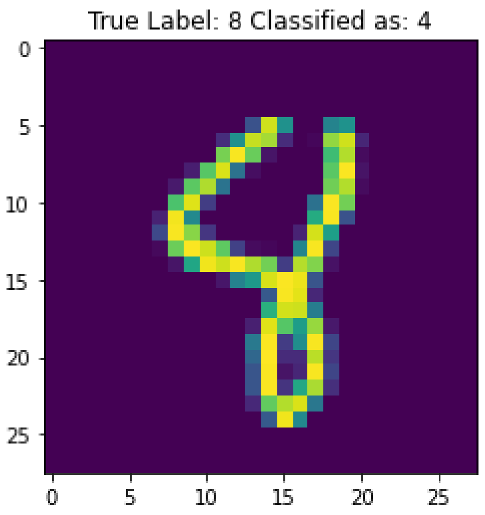

Figure 3:

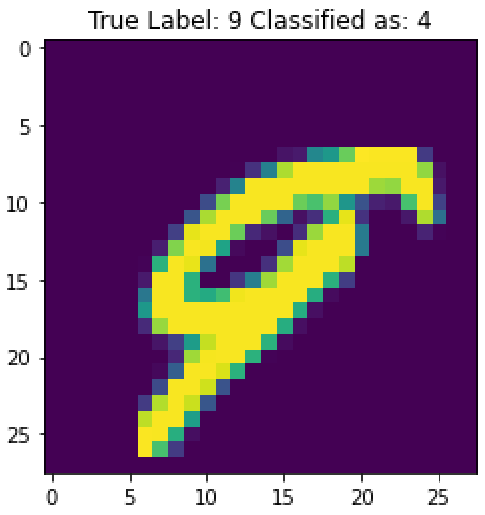

Figure 4:

3 Logistic Regression: Multi-class approach

In this approach we optimize for all weights simultaneously while building the model. The estimates are now given using softmax function which normalizes the probability for all cases for their sum to be = 1. Thus, the approach solves for all weights simultaneously. Table 3 & 4 show the class-wise and overall model accuracy respectively.

As with the previous model, the accuracy is similar for training and test sets however the overall accuracy increases.

Table 3: Class Wise Train & Test Accuracy(%)

Class	Train	Test
0	97	98
1	97	97
2	91	90
3	90	91
4	93	93
5	88	86
6	96	94
7	93	92
8	88	88
9	91	91

Table 4: Overall Accuracy(%)

Model	Train	Validation	Test
1 vs all	92.7	91.5	92
Softmax	93.1	92.4	92.5

4 Support Vector Machine

Unlike logistic, the SVM classifier can search for a non-linear classifier hyperplane in the data. However, since multiple parameters are provided to fit the model, we search for the optimum model systematically. Firstly, we sample a small number of points from our training set [10,000] and call is our model selection set. Using this, we test our model's fit on the full training, test and validation data. Using the four accuracy values, the optimum model is selected. The optimum model is then trained over the full data to compare with other classifiers.

4.1 Using linear kernel:

We first benchmark the model performance using a linear kernel, all other parameters remaining constant. Table 5 shows the accuracy for the model across the data sets:

Table 5: SVM: Linear Kernel (Acc %)

Model Selection Set	Full Training	Full Validation	Full Test
99.6	93.1	91.7	91.6

4.2 Using rbf kernal, varying gamma:

For non linear fit, we move to rbf kernel and vary gamma (Table 6): Moving from linear to rbf kernel improves training accuracy, however choosing gamma=1 over-fits the data leading to worse validation and test accuracy. Default gamma value on the other hand uses $\frac{1}{(no.features*X.var())}$ as value of gamma and fits much better on the data leading to consistently high training and test accuracy. We therefore continue with kernel='rbf' and gamma='default' to search for the optimum regularization parameter C

Table 6: SVM: Linear Kernel (Acc %)

Models	Model Selection Set	Full Training	Full Validation	Full Test
Linear Kernel	99.6	93.1	91.7	91.6
RBF Kernel (gamma=1.0)	100	34.6	17.1	18.1
RBF Kernel (gamma=default)	98.6	96.6	95.9	96.2

4.3 Using rbf kernal, default gamma, varying C:

In this experiment with vary C for values: [1 (default), 10 , 20 , 30 , 40 , 50 , 60 , 70 , 80 , 90 , 100] and plot accuracy values to find optimum (Figure 5). We find that the accuracy values increase when C is increased from 1 to 10, and then the value slightly decreases and remains constant there after. This is due to the fact that a higher C value penalizes more for errors, therefore creating hyper plane with lower margin, leading to an over-fit. A lower C value for similar reason under-fits and therefore an optimal value lies somewhere in between. From this we can conclude that our optimal SVM model is [kernel=rbf, gamma=default, C=10]

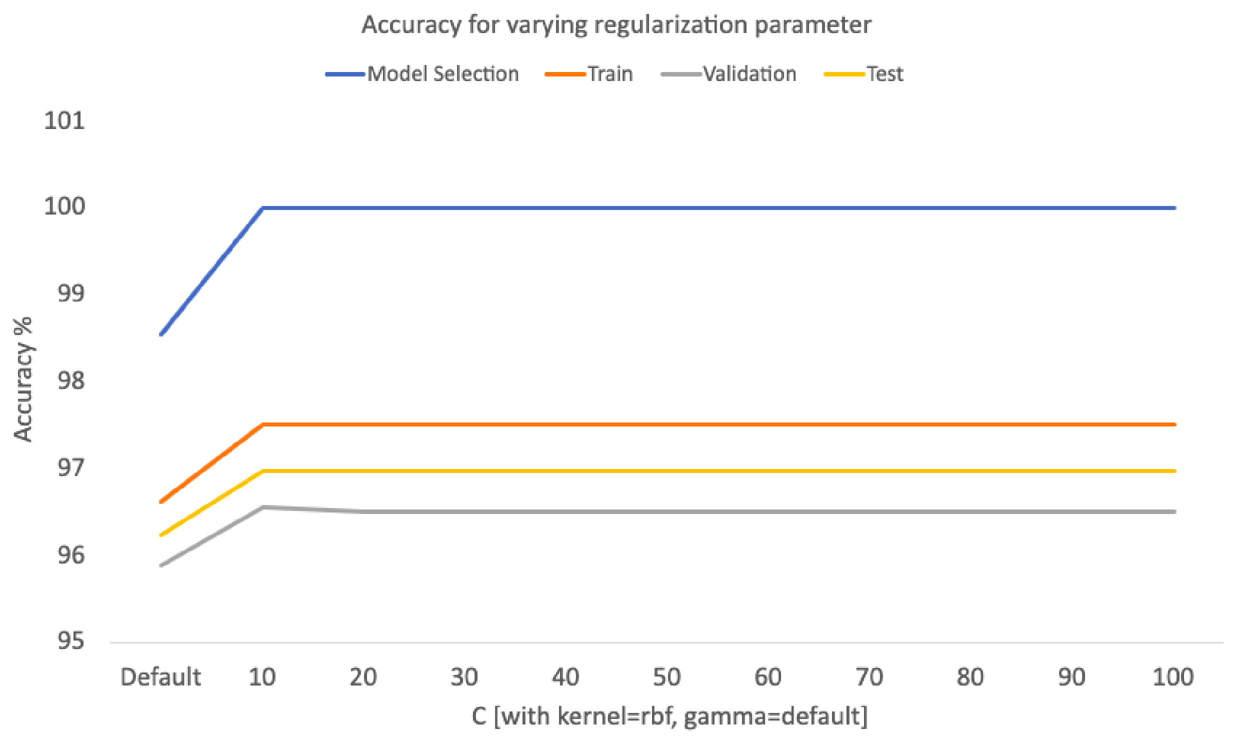

Figure 5:

4.4 Final Scoring and model comparisons:

Using the optimal SVM parameters [kernel=rbf, gamma=default, C=10], we train the final model on the entire training sample [N=50,000] and compare it with our logistic models (Table 7):

Table 7: Overall Accuracy(%)

Model	Train	Validation	Test
1 vs all	92.7	91.5	92
Softmax	93.1	92.4	92.5
Optimal SVM	99.9	98.5	98.3

From the table we can conclude that the SVM model works best for the given problem as it is non-linear in nature.