CSE 574 - Introduction to Machine Learning
Programming Assignment 3
Project Report

Submitted by:
Anupriya Goyal(50287108)
Sravanthi Adibhatla(50288587)
Sagar Pokale(50288055)

1. Logistic Regression:

We trained the logistic regression classifier using the entire MNIST training set and obtained the following accuracies:

- Training set Accuracy : **92.716 %**
- Validation set Accuracy : **91.44%**
- Testing set Accuracy : **92.03%**

The aforementioned accuracies were along expected lines.
Furthermore, we computed the train and test accuracies for each of the individual classes with the help of a confusion matrix to get a better picture of the error for each category.

Following are the training and test accuracies for each of the classes:

CLASS	**TRAIN ACCURACY %**	**TEST ACCURACY %**
Class 0	97.8	98.061
Class 1	97.9	98.149
Class 2	91.1	89.0503
Class 3	89.807	91.089
Class 4	93.907	93.380
Class 5	88.305	85.6502
Class 6	96.299	94.885

Class 7	94.33	92.607
Class 8	87.548	87.063
Class 9	88.765	89.098

2. Multi-Class Logistic Regression:

We trained a multi-class logistic regression classifier using the entire MNIST training set and obtained the following accuracies:
Training set Accuracy : **92.018 %**
Validation set Accuracy : **91.53 %**
Test set Accuracy : **91.79 %**

Using the same methodology used in Binary Logistic Regression, we calculated accuracies for each of the 10 classes. Furthermore, we computed the train and test accuracies for each of the individual classes with the help of a confusion matrix to get a better picture of the error for each category.
Following are the accuracy values :

CLASS	**TRAIN ACCURACY %**	**TEST ACCURACY %**
Class 0	97.460	99.183
Class 1	97.387	97.621
Class 2	90.116	89.631
Class 3	89.1833	90.99
Class 4	93.515	93.38
Class 5	86.7903	85.201
Class 6	95.933	94.572
Class 7	93.770	91.820
Class 8	87.793	87.371
Class 9	86.786	88.00

Inference from logistic regression and multi-class logistic regression:
Ideally, the multi-class logistic regression accuracy should be more than one-vs-all category logistic regression accuracy. But in our case, the one-vs-all category logistic regression has better accuracy because our data set is small and number of classes defined are less.

3. Support Vector Machines:

Following are the accuracies for the SVM trained on the **sample** training dataset for different parameters:

SVM with linear kernel:

- Training set accuracy for linear kernel: **92.856 %**
- Validation set accuracy for linear kernel: **91.46 %**
- Test set accuracy for linear kernel: **91.81 %**

SVM with RBF kernel and gamma=1:

- Training set accuracy for SVM with RBF kernel and gamma=1: **32.896 %**
- Validation set accuracy for SVM with RBF kernel and gamma=1: **16.19 %**
- Test set accuracy for SVM with RBF kernel and gamma=1: **17.77 %**

SVM with RBF and default parameters:

- Training set accuracy for SVM with RBF kernel : **91.816 %**
- Validation set accuracy for SVM with RBF kernel: **91.98 %**
- Test set accuracy for SVM with RBF kernel: **92.25 %**

SVM with RBF and different values of C:

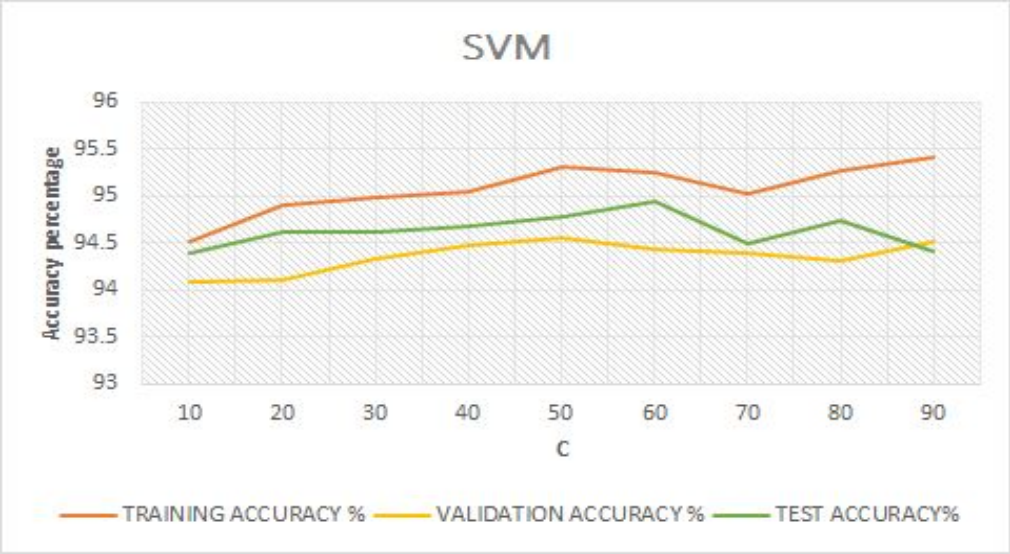
- Training set accuracy for SVM with RBF kernel and C=1 : **91.918 %**
- Validation set accuracy for SVM with RBF kernel and C=1: **92.05 %**
- Test set accuracy for SVM with RBF kernel and C=1: **92.53 %**

As we can see from the above observations, it is clear that linear kernel the testing accuracy is around **91.8** and radial kernel testing accuracy of around **92.25** percent when other parameters are default.
When RBF kernel is used with Gamma = 1, the **validation and testing accuracies are very low**, almost around 16 to 17 percent, respectively. This is because using gamma=1 led to overfitting of the model on training data. Because we used comparatively high Gamma opposed to the sklearns default value (1/N) our learnt model became **high bias and low-variance model**. As a result,our model has poor performance for data that is ‘different’ from the training data.

TABLE - ACCURACY FOR SVM WITH RBF AND VARYING C :

C VALUE	TRAINING ACCURACY %	VALIDATION ACCURACY %	TEST ACCURACY%
10	94.518	94.09	94.4
20	94.902	94.11	94.61
30	94.98	94.33	94.62
40	95.05	94.47	94.68
50	95.32	94.55	94.79
60	95.26	94.44	94.94
70	95.026	94.39	94.5
80	95.274	94.31	94.74
90	95.408	94.51	94.41
100	95.158	94.18	94.68

Below is the plot which clearly depicts that as C increases, accuracy also increases. But after a point, the accuracy decreases a bit and again increases. Because C in our case controls the influence of individual support vectors and this helps the model 'ignore' the error/cost of one or two misclassified vectors. Therefore, we are considering that point before the accuracy decreases as **optimal C value i.e. at 60**.



From the above experiments for **sample 10,000 dataset,** we can see that, C= 60, gives us the best validation and test accuracies i.e almost **94.44 % and 94.94 %**

Following are the accuracies for the SVM trained on the **entire** training dataset for different parameters:

SVM with linear kernel:

- Training set accuracy for linear kernel: **97.286 %**
- Validation set accuracy for linear kernel: **93.64 %**
- Test set accuracy for linear kernel: **93.78 %**

SVM with RBF kernel and gamma=1:

- Training set accuracy for SVM with RBF kernel and gamma=1: **100 %**
- Validation set accuracy for SVM with RBF kernel and gamma=1: **15.48 %**
- Test set accuracy for SVM with RBF kernel and gamma=1: **17.14 %**

SVM with RBF and default parameters:

- Training set accuracy for SVM with RBF kernel : **94.294 %**
- Validation set accuracy for SVM with RBF kernel: **94.02%**
- Test set accuracy for SVM with RBF kernel: **94.42%**

SVM with RBF and different values of C:

- Training set accuracy for SVM with RBF kernel and C=1 : **94.294 %**
- Validation set accuracy for SVM with RBF kernel and C=1: **94.02%**
- Test set accuracy for SVM with RBF kernel and C=1: **94.42%**