**CSE 574 Programming Assignment 3**
**Classification And Regression**

Group number 12
Bhavika Jain - 50170168
Chen Song - 50060333
Pranav Jain - 50169659

**Objective** : Implement Logistic Regression and use the Support Vector Machine tool to classify hand-written digit images and compare the performance of these methods.

Please find the results below:

**I ) Logistic Regression(BLR) :**

Below are the results of performing Logistic Regression (binary logistic regression) on the given dataset:

Training Accuracy % : 86.294%

Validation Set Accuracy % : 85.33%

Testing Set Accuracy% : 85.38%

**II) SVM(Support Vector Machine):**

Please find the results below:

1) **Using Linear Kernel:**

Training Accuracy % : 97.286%

Validation Set Accuracy % : 93.64%

Testing Set Accuracy% : 93.78%

**2) Radial Basis Function:**

The **gamma** parameter defines how far the influence of a single training example reaches, with low values meaning 'far' and high values meaning 'close'. The **gamma** parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.

**a) Using Radial Basis Function(Gamma = 1)**

Training Accuracy % : 100.0%

Validation Set Accuracy % : 15.48%

Testing Set Accuracy% : 17.14%

**b) Using Radial Basis Function(Gamma = 0)**

Training Accuracy % : 94.294%

Validation Set Accuracy % : 94.02%

Testing Set Accuracy% : 94.42%

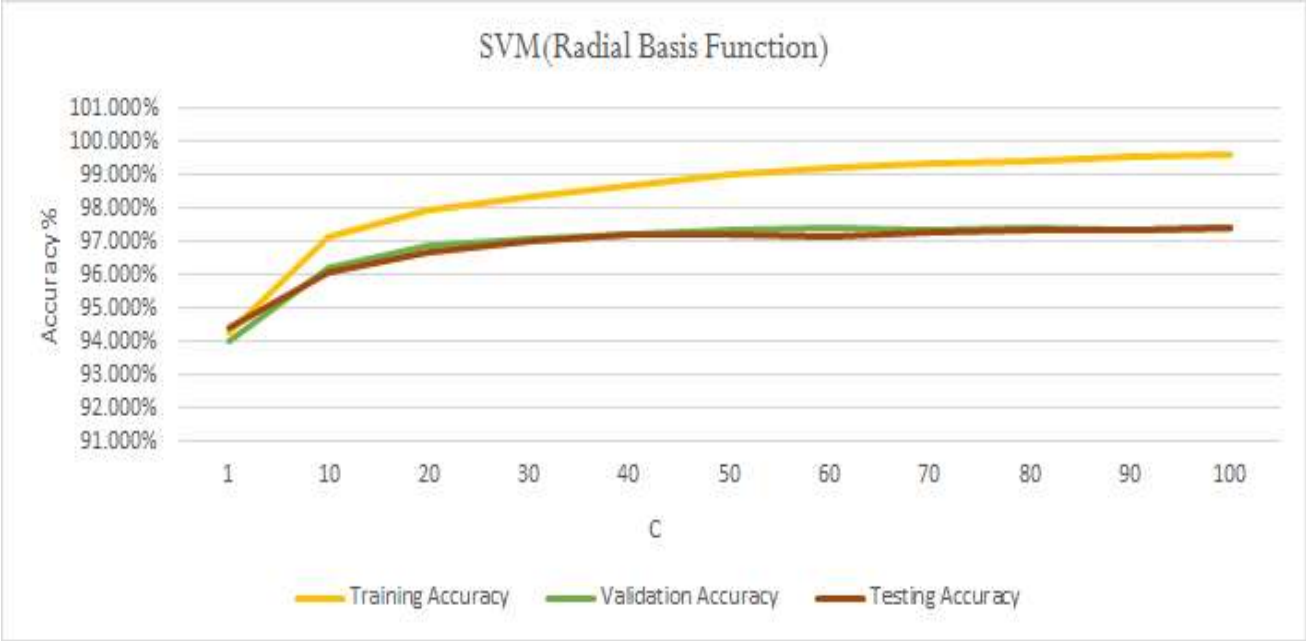
**c) Using radial basis function with value of gamma setting to default and varying value of C (1, 10, 20, 30, ..., 100).**

The **C** parameter trades off misclassification of training examples against simplicity of the decision surface. For large values of **C**, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of **C** will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

Below are the results for different values of C on Training, Testing and Validation data:

| **C** | **Training Accuracy** | **Validation Accuracy** | **Testing Accuracy** |
|---|---|---|---|
| 1 | 94.294% | 94.020% | 94.420% |
| 10 | 97.132% | 96.180% | 96.100% |
| 20 | 97.952% | 96.900% | 96.670% |
| 30 | 98.372% | 97.100% | 97.040% |
| 40 | 98.706% | 97.230% | 97.190% |
| 50 | 99.002% | 97.310% | 97.190% |
| 60 | 99.196% | 97.380% | 97.160% |
| 70 | 99.340% | 97.360% | 97.260% |
| 80 | 99.438% | 97.390% | 97.330% |
| 90 | 99.542% | 97.360% | 97.340% |
| 100 | 99.612% | 97.410% | 97.400% |

Plot for various for values of C and the accuracy obtained on each of Training, Testing and Validation dataset :




**III) Direct Multi-class Logistic Regression :**

**Below are the results obtained:**

Training Accuracy % : 93.39%

Validation Set Accuracy % : 92.43%

Testing Set Accuracy% : 92.67%

**IV) Comparison between Logistic regression and SVM:**

It can be seen from the results reported above that SVM performs better than Logistic regression for all the setups except for RBF with Gamma = 1. This is due to overfitting in SVM, wherein it performs poorly on test and validation data. But overall, it performs better than Logistic Regression.

The reasons for this could be since the number of features are large - SVM performs better here as compared to Logistic regression, which performs better in case of input data with less features. Also, Logistic regression stops after it finds a compatible hyperplane while SVM finds the best one.

The plot of different values of C vs accuracy % indicates that as the value of C increases, the accuracies also increase. Thus C can be used to limit the error term for the training examples.