Department of Computer Science & Engineering



**CSE 574 – MACHINE LEARNING**

Classification and Regression

Programming Assignment – 3

Presented By:

Pratik **Agarwal** [50290570]

Sourav **Bihani** [50290969]

Tannu **Priya** [50290768]

Group - 23

Table of Contents:

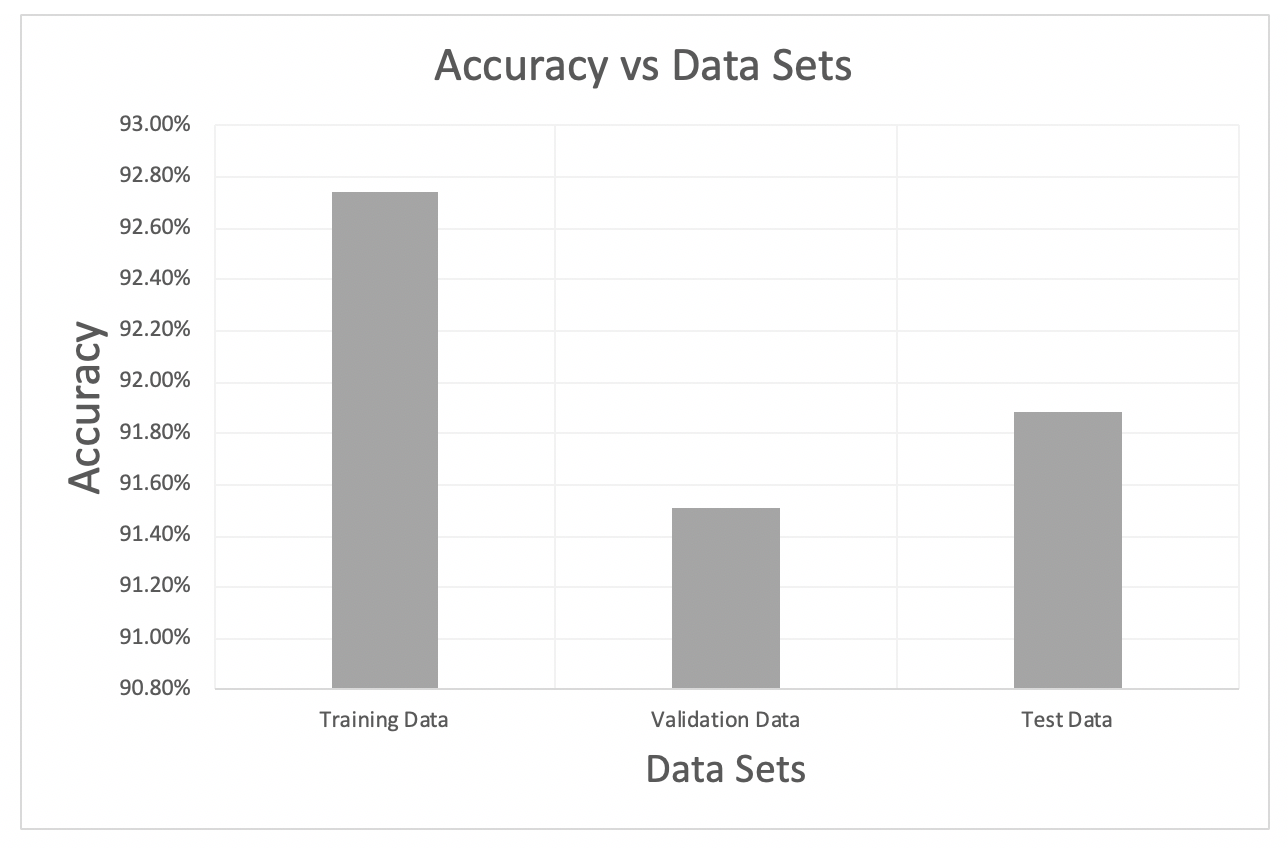
1. Problem 1………………………………………………………………………………3
2. Problem 2………………………………………………………………………………8
3. Problem 3…………………………………………………………………………….10

**Problem 1**: Implementation of Logistic Regression

The following table shows various accuracies obtained in Logistic Regression against varied data set:

|  |  |
| --- | --- |
| Data | Accuracy |
| Training Data | 92.746 % |
| Validation Data | 91.51 % |
| Test Data | 91.88 % |

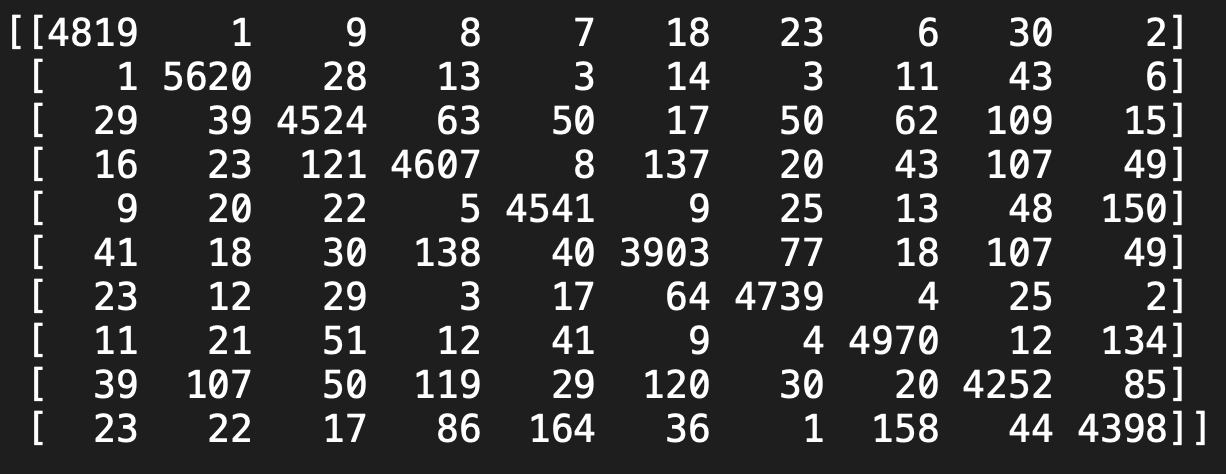
The total time taken by our code for Logistic Regression is 808.7755 seconds.



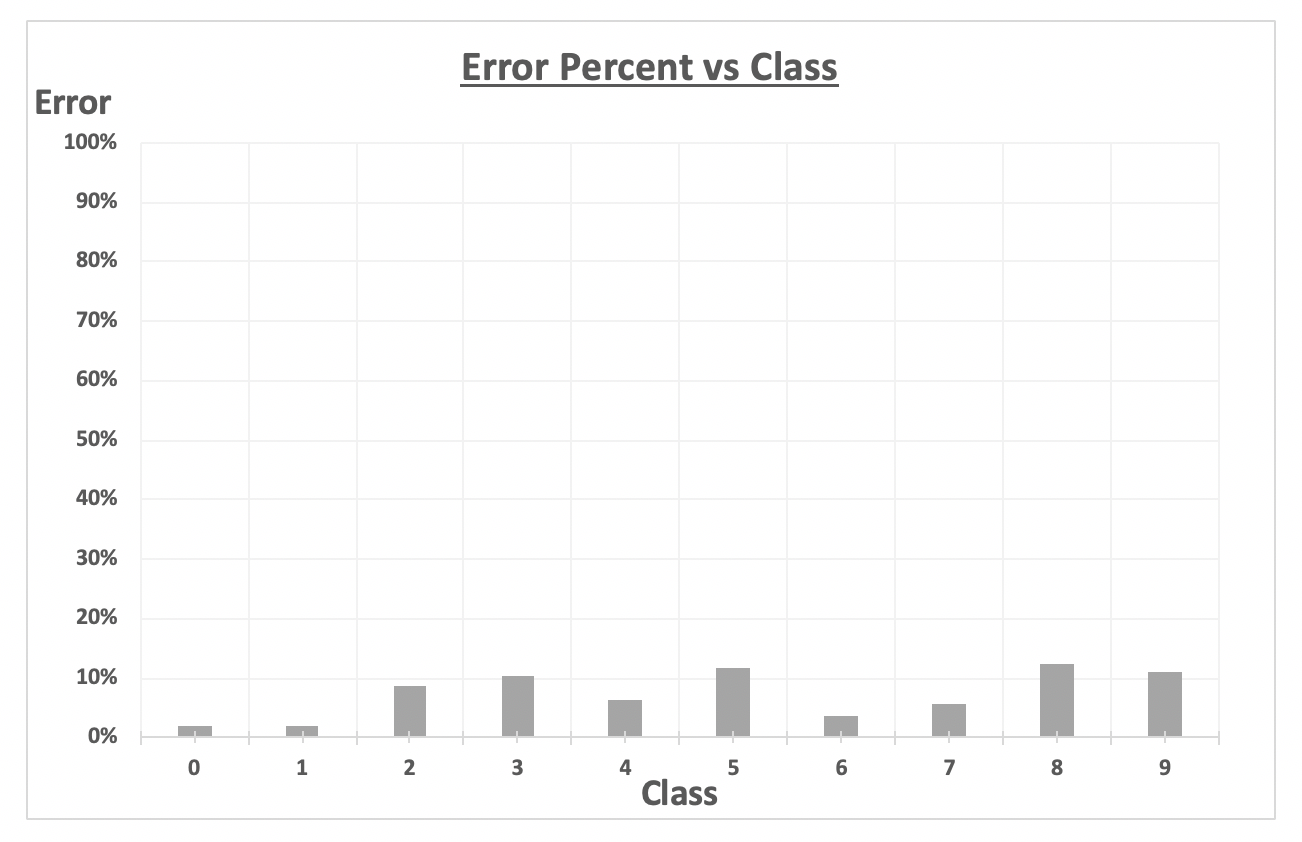
Category wise error:

1. Training Data:

Confusion Matrix:

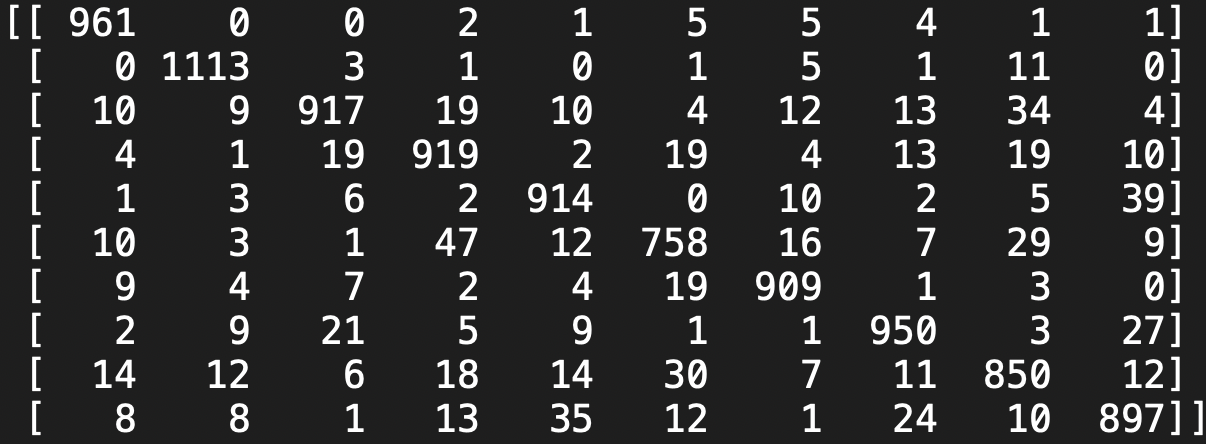


|  |  |
| --- | --- |
| Class | Error % |
| 0 | 2.11% |
| 1 | 2.12% |
| 2 | 8.75% |
| 3 | 10.21% |
| 4 | 6.22% |
| 5 | 11.72% |
| 6 | 3.64% |
| 7 | 5.60% |
| 8 | 12.35% |
| 9 | 11.13% |

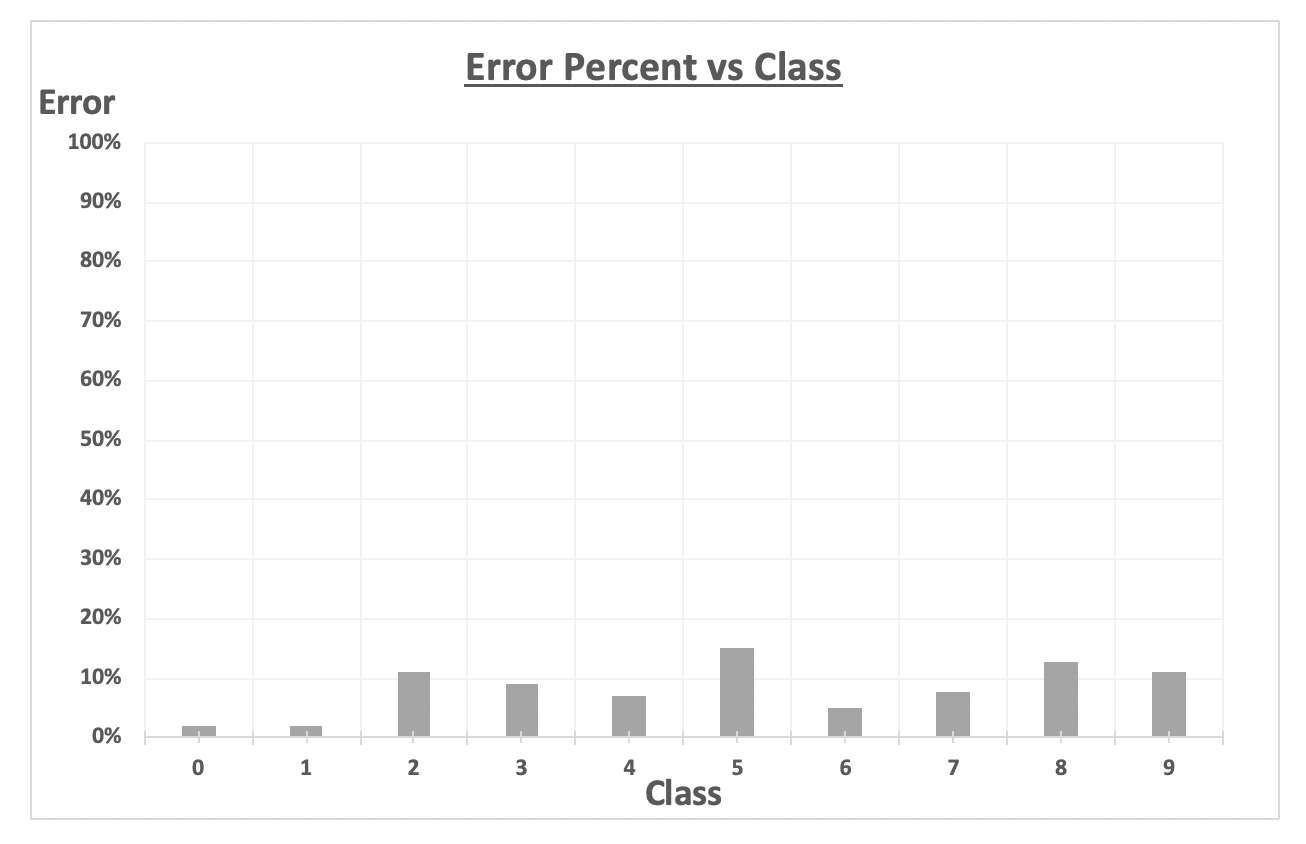


1. Test Data:

Confusion Matrix:



|  |  |
| --- | --- |
| Class | Error |
| 1 | 1.94% |
| 2 | 1.94% |
| 3 | 11.14% |
| 4 | 9.01% |
| 5 | 6.92% |
| 6 | 15.02% |
| 7 | 5.11% |
| 8 | 7.59% |
| 9 | 12.73% |
| 10 | 11.10% |



Our Implementation of Linear Regression incorporates building 10 binary classifiers (10 classes) to distinguish between different classes.

**Conclusion**:

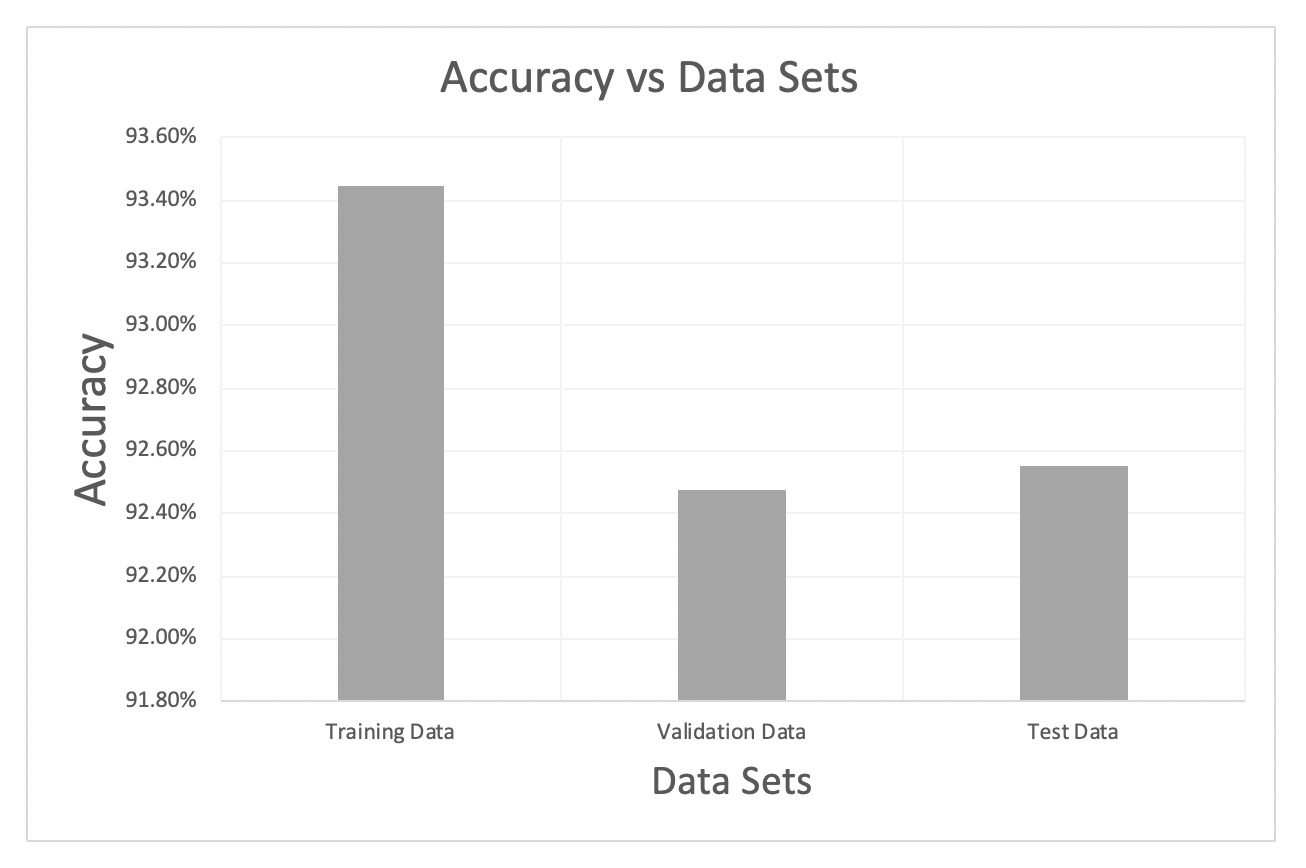
* The accuracy for the test data set is lower than the accuracy for training data set.
* There is a difference obtained between the training error and test error is due to the fact that Linear Regression progresses as a non-convex function.

**Problem 2**: Implementation of Multi-Class Logistic Regression

The following table shows various accuracies obtained in Multi-Class Logistic Regression against varied data set:

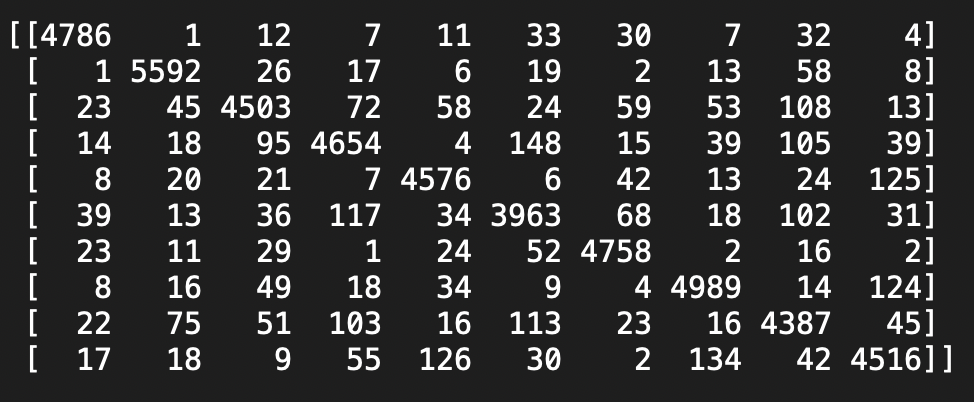
|  |  |
| --- | --- |
| Data | Accuracy |
| Training Data | 93.448 % |
| Validation Data | 92.479 % |
| Test Data | 92.55 % |

The total time taken by our code for Multi-Class Logistic Regression is 24.5829 seconds.



Category wise error:

1. Training Data:

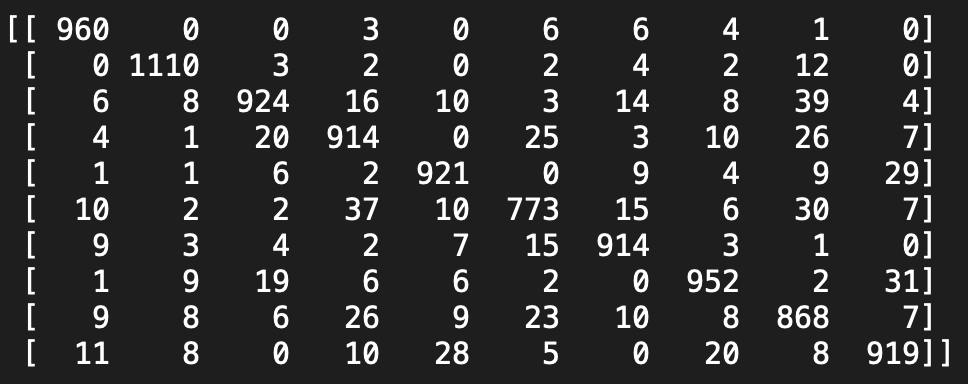
Confusion Matrix:

|  |  |
| --- | --- |
| Class | Error % |
| 0 | 2.78% |
| 1 | 2.61% |
| 2 | 9.18% |
| 3 | 9.30% |
| 4 | 5.49% |
| 5 | 10.36% |
| 6 | 3.25% |
| 7 | 5.24% |
| 8 | 9.57% |
| 9 | 8.75% |

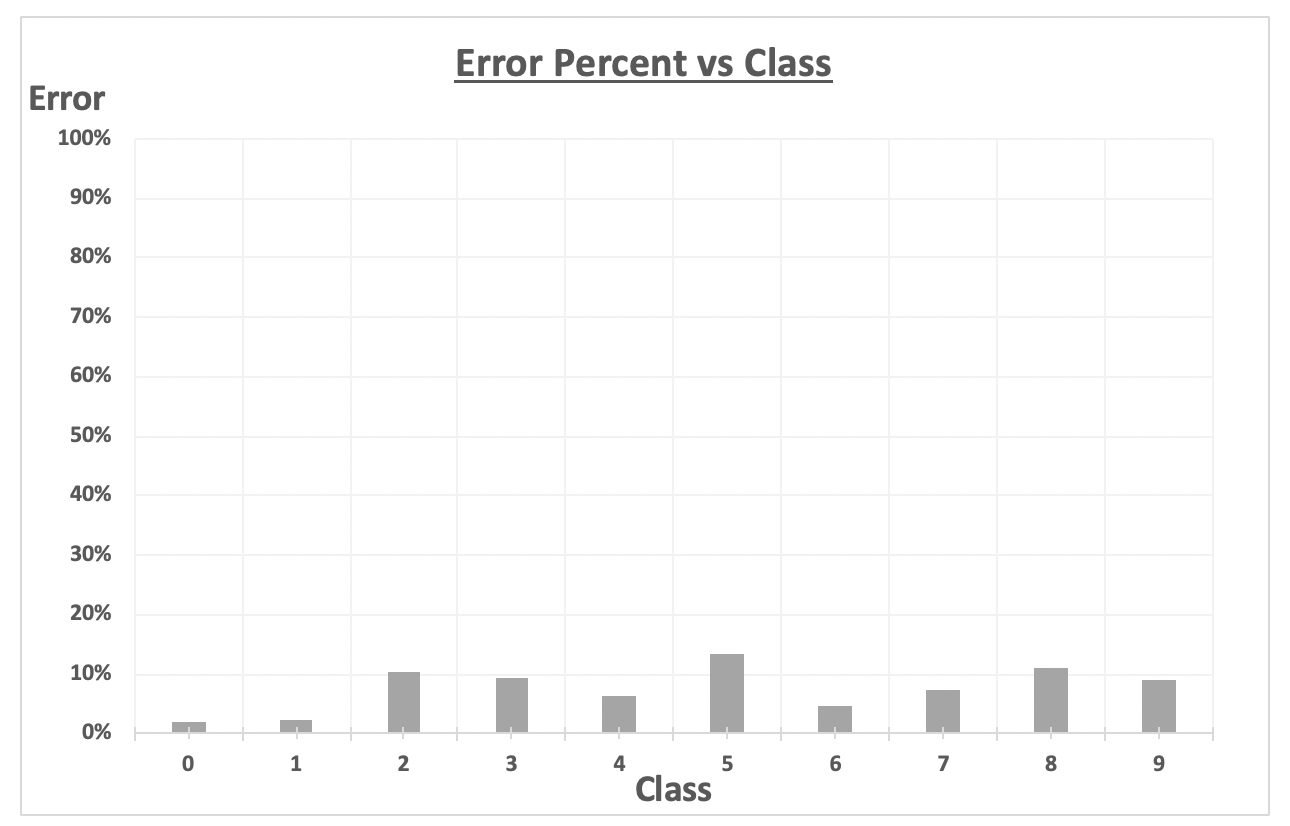


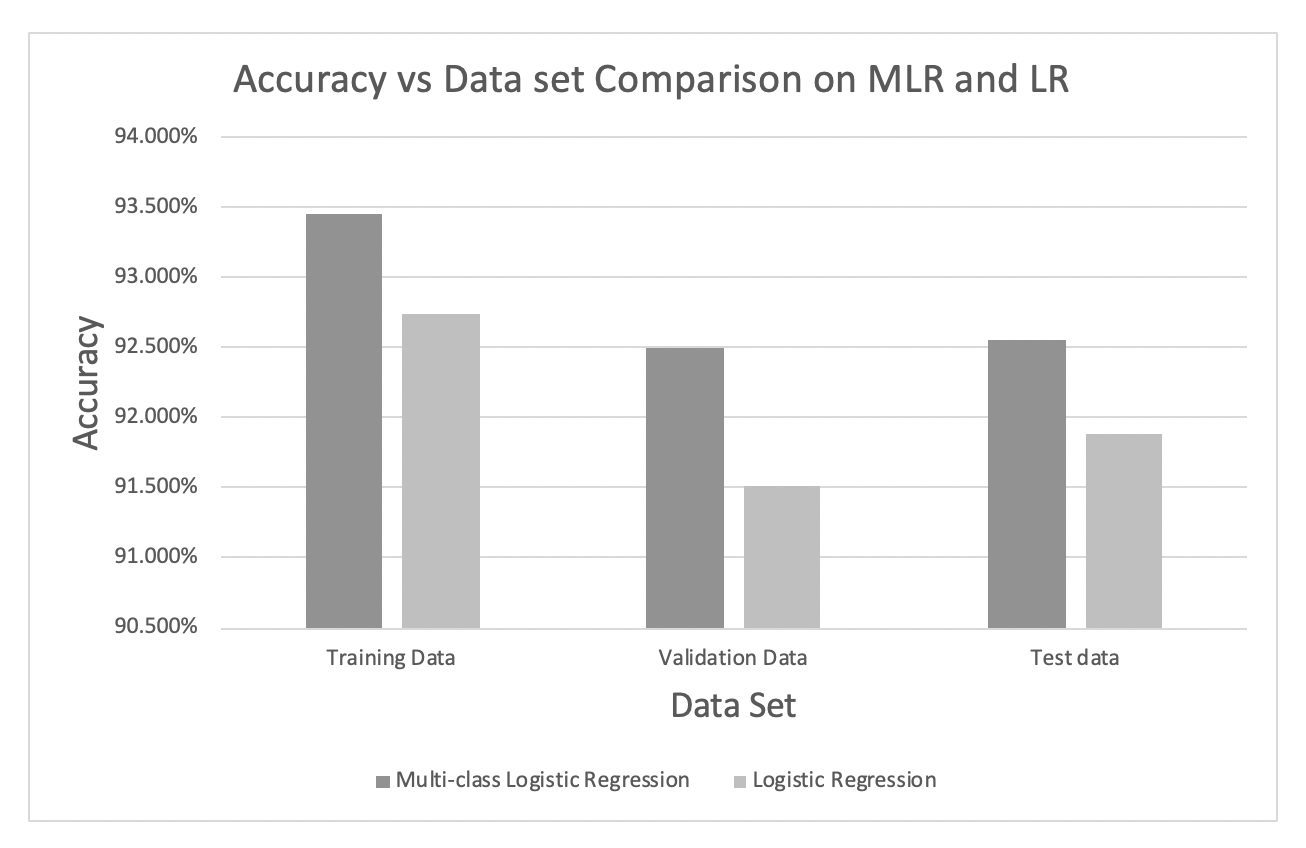
1. Test Data:

Confusion Matrix:



|  |  |
| --- | --- |
| Class | Error % |
| 0 | 2.04% |
| 1 | 2.20% |
| 2 | 10.47% |
| 3 | 9.50% |
| 4 | 6.21% |
| 5 | 13.34% |
| 6 | 4.59% |
| 7 | 7.39% |
| 8 | 10.88% |
| 9 | 8.92% |



Conclusion:

Comparison:

|  |  |
| --- | --- |
| Linear Regression | Multi-class Linear Regression |
| Time taken in Linear Regression is more than Multi-class Linear Regression as optimization is performed 10 times (one for each class – digits 0 to 9) | Multi-class Linear Regression takes lesser time than Linear Regression as the classifier can run at once for all the classes at once. |
| This performs worse than Multi-class Linear Regression as Linear Regression is primarily used for Binary Classification. | This performs better than Linear regression if we consider the one vs all Logistic Regression strategy. |

**Problem 3**: Implementation of Support Vector Machines

1. SVM using Linear Kernel:

The following table shows various accuracies obtained in SVM against varied data set:

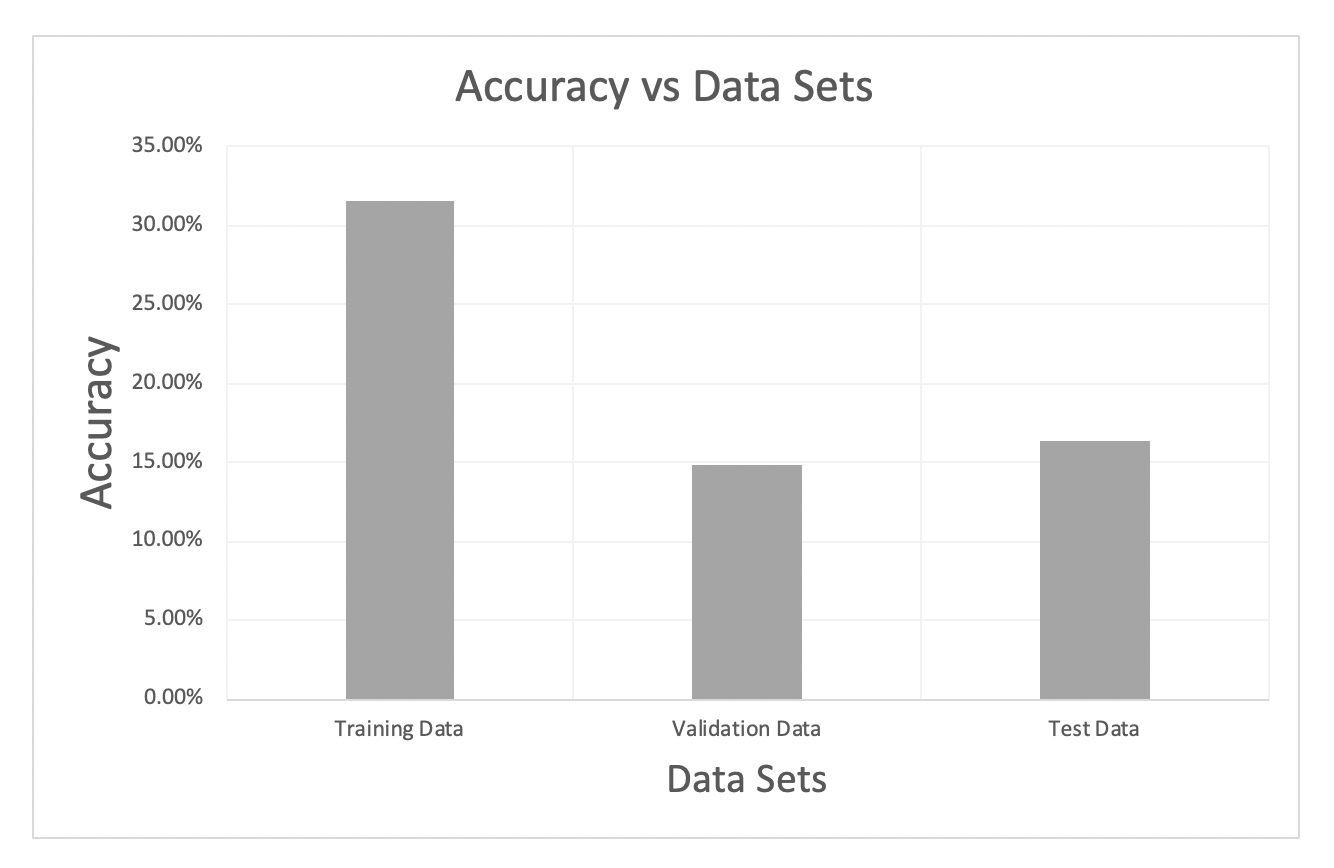
|  |  |
| --- | --- |
| Data | Accuracy |
| Training Data | 92.874 % |
| Validation Data | 91.58 % |
| Test Data | 92.07 % |

The total time taken by our code for SVM is 165.5910 seconds.

1. SVM with radial basis function for gamma = 1

The following table shows various accuracies obtained in SVM against varied data set:

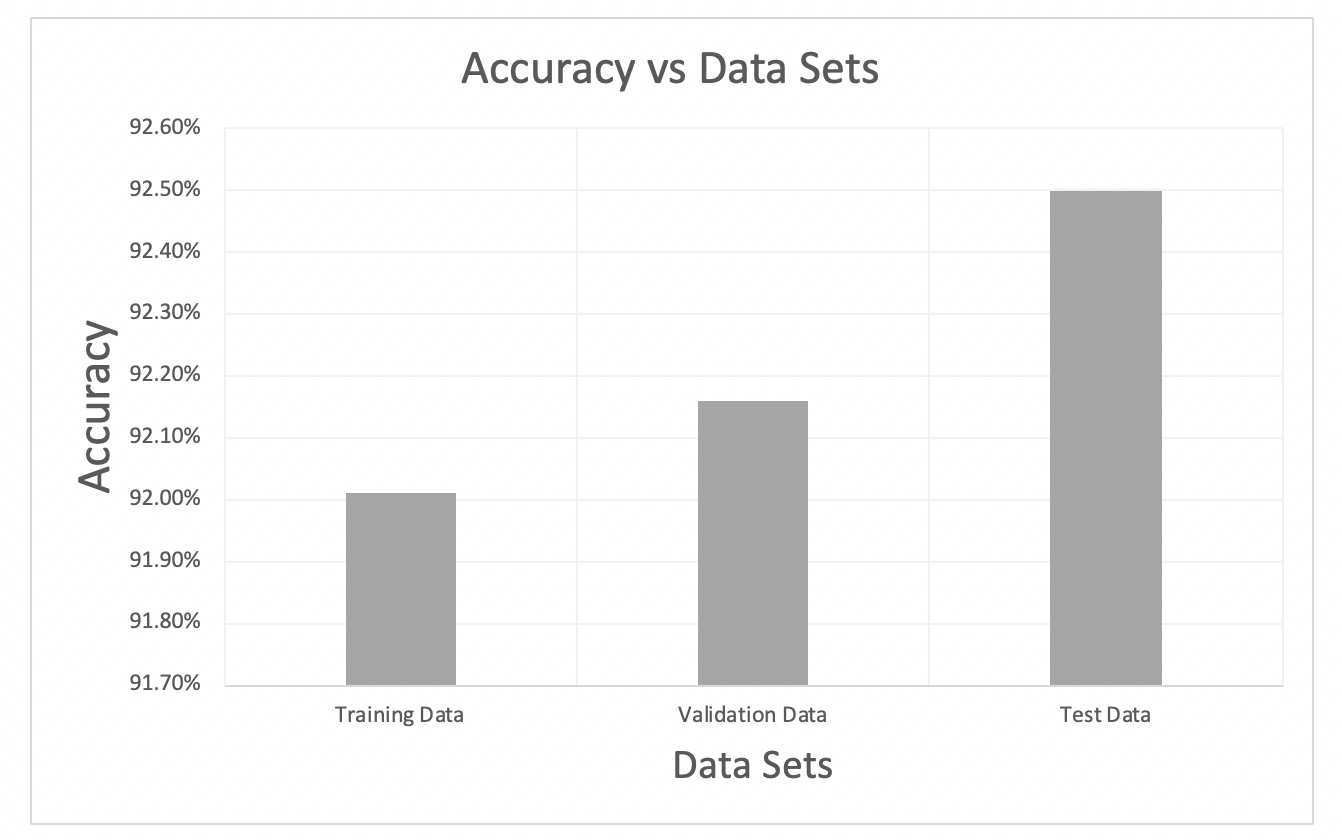
|  |  |
| --- | --- |
| Data | Accuracy |
| Training Data | 31.5419 % |
| Validation Data | 14.8799 % |
| Test Data | 16.400 % |

The total time taken by our code for SVM is 719.6619 seconds.

1. SVM with radial basis function for default gamma

The following table shows various accuracies obtained in SVM against varied data set:

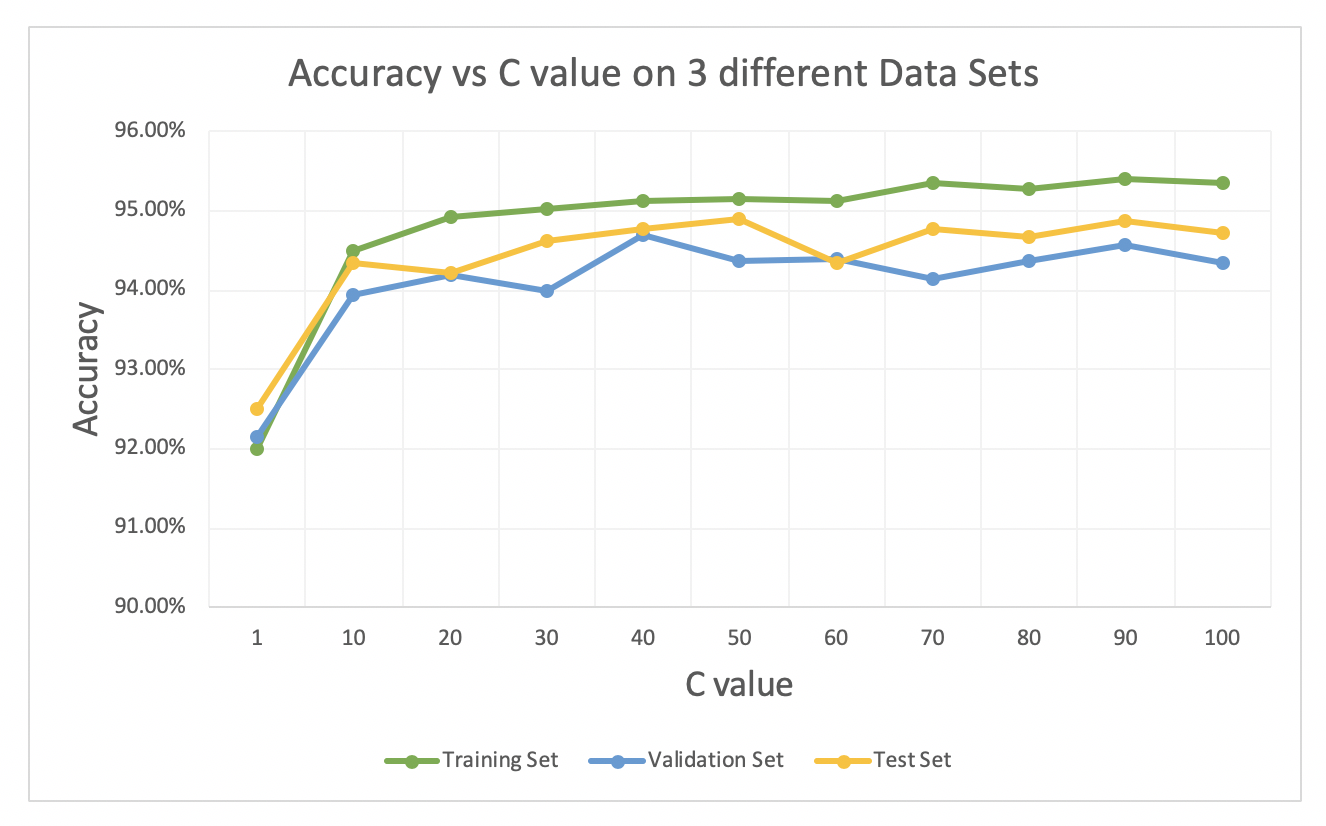
|  |  |
| --- | --- |
| Data | Accuracy |
| Training Data | 92.01 % |
| Validation Data | 92.16 % |
| Test Data | 92.5 % |

The total time taken by our code for SVM is 325.9381 seconds.

1. SVM with radial basis function for default gamma and varying value of C

The following table shows various accuracies obtained in SVM against varied values of C with gamma set to default:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| C | Training Set | Validation Set | Test Set | Time (seconds) |
| 0 | 92.01% | 92.16% | 92.50% | 325.938 |
| 10 | 94.50% | 93.93% | 94.35% | 205.322 |
| 20 | 94.92% | 94.20% | 94.21% | 178.46 |
| 30 | 95.02% | 93.99% | 94.62% | 179.701 |
| 40 | 95.13% | 94.69% | 94.78% | 172.672 |
| 50 | 95.154% | 94.36% | 94.89% | 171.063 |
| 60 | 95.13% | 94.39% | 94.35% | 174.321 |
| 70 | 95.34% | 94.13% | 94.77% | 417.445 |
| 80 | 95.27% | 94.37% | 94.67% | 200.509 |
| 90 | 95.40% | 94.58% | 94.86% | 185.77 |
| 100 | 95.34% | 94.35% | 94.73% | 191.635 |

The total time taken by our code for SVM is 2402.836 seconds

Conclusion:

We computed the Optimal SVM on whole training dataset with radial basis function kernel and value of gamma setting to default and C = 50. We used C = 50 as taking gamma = 1 led to overfitting and in the default case led to underfitting.