CSE 574

INTRODUCTION TO MACHINE LEARNING

PROGRAMMING ASSIGNMENT 3

CLASSIFICATION AND REGRESSION

BY

HARIKRISHNA RANGINEENI 50248741 hrangine

AJAYSAI POTLURI 50246594 ajaysaip

UNIVERSITY AT BUFFALO

Spring 2018

CONTENTS

1. INTRODUCTION TO LOGISTIC REGRESSION AND SVM

1.1. LOGISTIC REGRESSION

Logistic regression is a discriminative classifier which tries to learn linear boundary. Hence, it is a model for classification rather than for regression. This model is based on Log Loss where it trains data based on Maximum Likelihood Approach. Logistic regression is better than linear regression in way that it can handle classification of multiple classes by directly calculating posterior probability for each outcome. Hence, it's also known as MaxEnt Classifier or Maximum entropy Classifier. To achieve multiclass classification, logistic model uses One-vsRest approach or One-VS-Other approach as discussed later.

1.2. SVM

SVM or Maximum Margin Classifier is a quadratic optimizer that tries to minimize error through better generalizability and by increasing Margin. SVM is used to improve the accuracy of classification and because of its ability to deal with high dimensional data. Also, unlike Logistic regression, SVM classifies two classes by finding the hyper-plane for the data and maximizing the margin between the two different classes. When training an SVM, we need to make a number of decisions: how to process the data, what kernel to use and finally setting the parameters of SVM and the kernel. We will be experimenting with three different parameters for this assignment, the kernel, gamma and C. The theoretical explanations behind these choices are explained as below:

Kernel:

Kernel provides various options such as Linear, RBF, poly and others. RBF and poly are useful for nonlinear hyper-plane. Linear kernel is used in cases where we have large number of features because it is more likely that data is linearly separable in high dimensional space. When using RBF, care should be taken to cross-validate the parameters so as to avoid over-fitting.

Gamma:

It is the kernel co-efficient for RBF kernel. For higher values of gamma, it will try to better fit as per training data set and make mistakes in classifying validation and test data due to over-fitting.

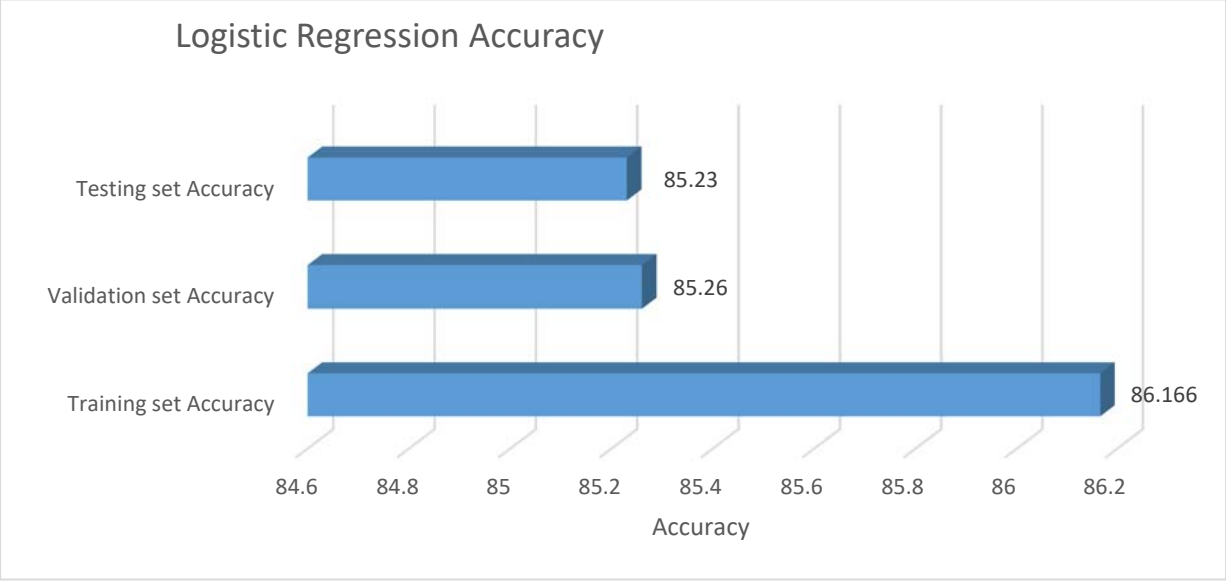
C:

Penalty parameter C of the error term, it controls the trade-off between a smooth decision boundary and classifying the training points correctly. A low C makes the decision surface smooth whereas a high C classifies the training samples giving the model freedom to select more samples as support vectors. As the C value is increased, the margin between the hyper-planes around the decision boundary decreases.

An effective combination of these parameters should always be looked into at the cross-validations core to avoid over-fitting.

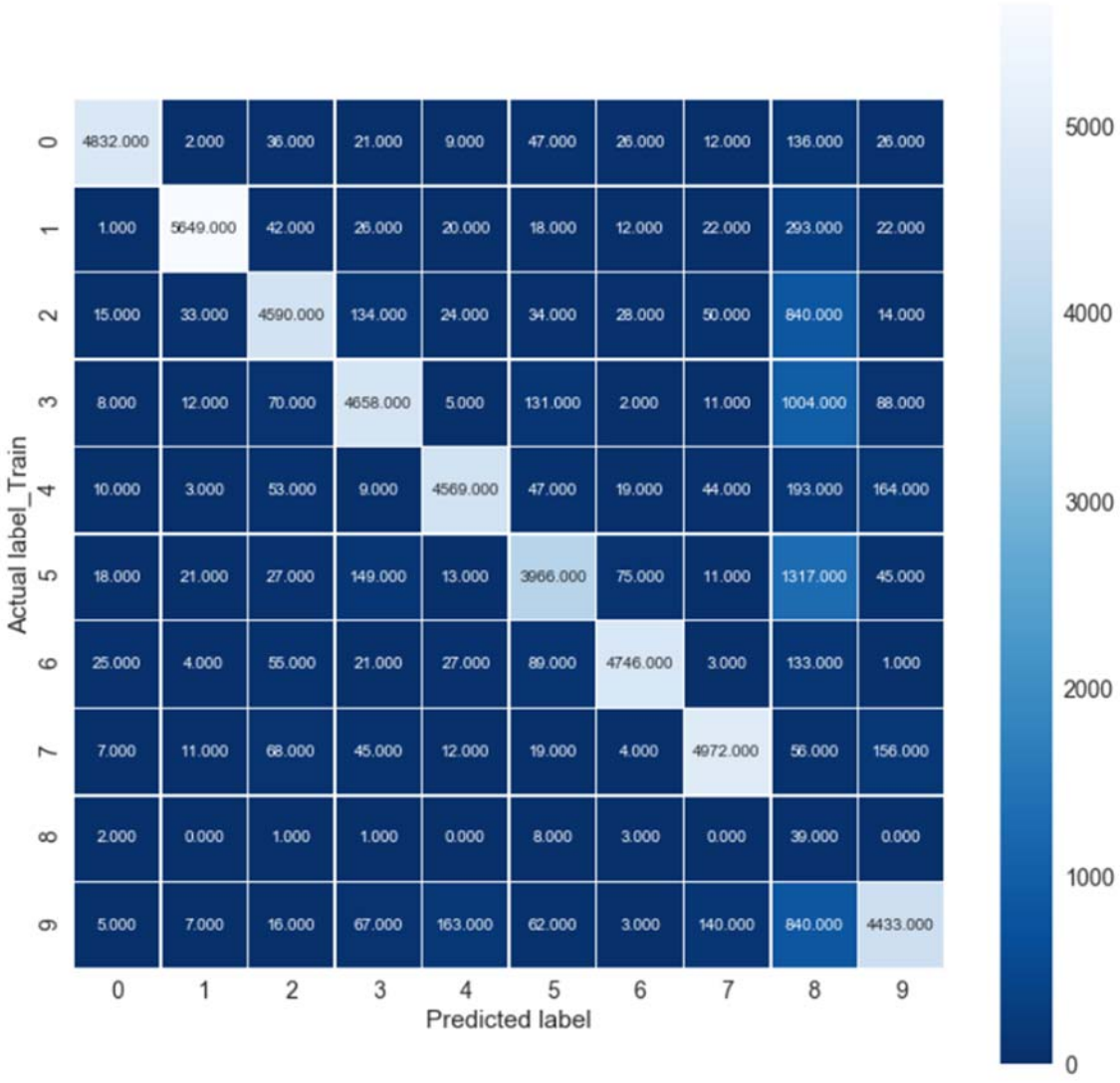
2. RESULTS OF LOGISTIC REGRESSION

For binary logistic regression, each time the algorithm optimizes the weights of one class. For example, after the first time of the optimization we can get the optimized weights in order to classify the digit “0”. Then we assemble the weights that can identify “0” into the first column of the weight matrix. Following the same procedures, we can get the ten weight columns with respect to the ten digits. However, we need to do the optimization ten times in order to classify the ten digits, which takes much time. Furthermore, each time the gradient descent algorithm only finds the minimum point for one class. Thus the minimum point is not the global minimum for the whole map of the ten classes. That’s why the classification accuracy is not ideal.

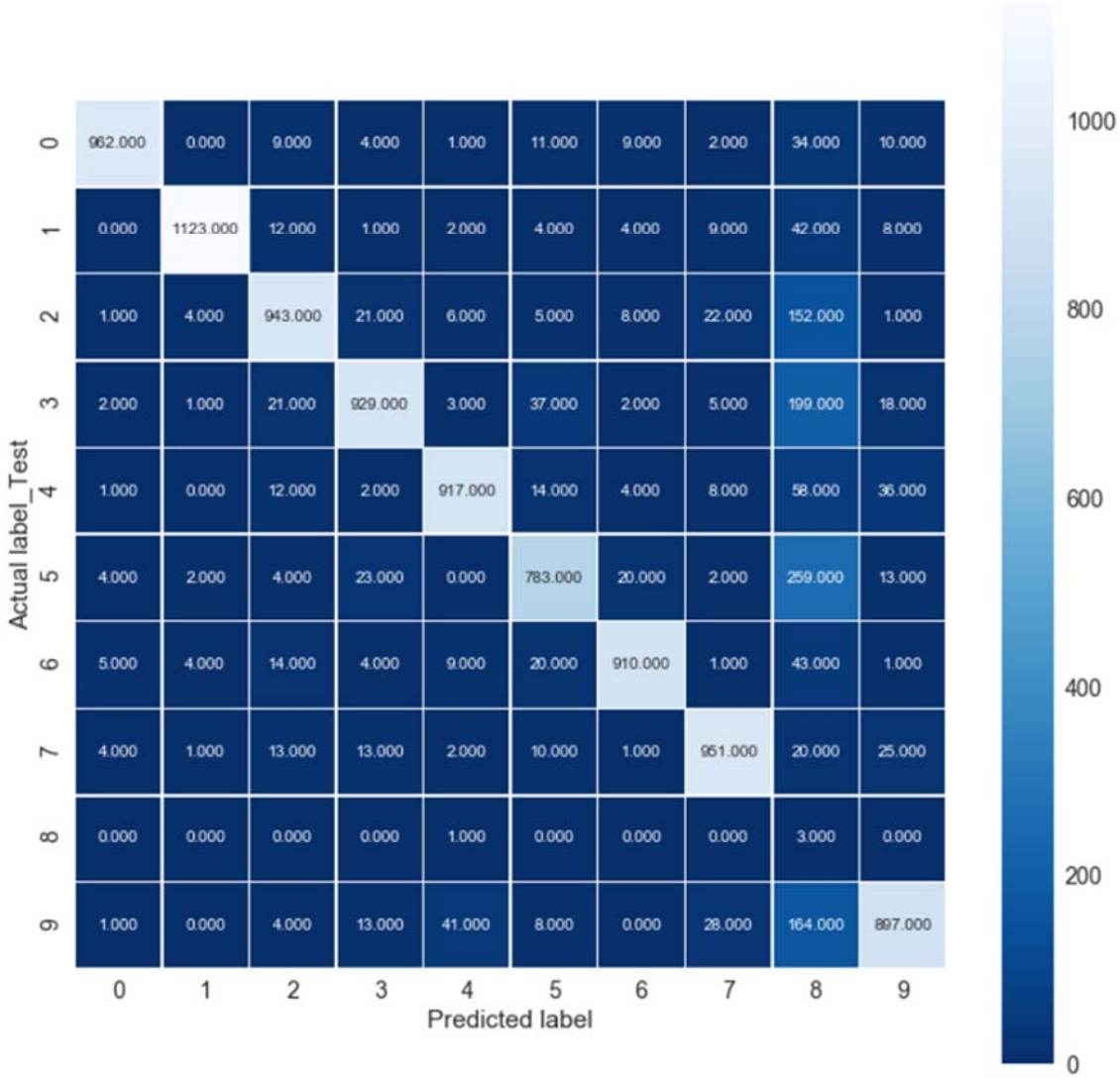



```
login as: hrangine
hrangine@metallica.cse.buffalo.edu's password:
Last login: Tue Apr 17 19:11:23 2018 from vpn205-245-164.vpn.buffalo.edu

>>System monitored for unauthorized usage and abuse.

> Hostname:              metallica.cse.buffalo.edu
> Operating System:      CentOS-6 x86_64
> System Use:            Compute server


metallica {~} > cd ML_PA2/
metallica {~/ML_PA2} > python script.py

 Training set Accuracy:86.166%

 Validation set Accuracy:85.26%

 Testing set Accuracy:85.23%
```

2.1. TRAIN AND TEST ERROR

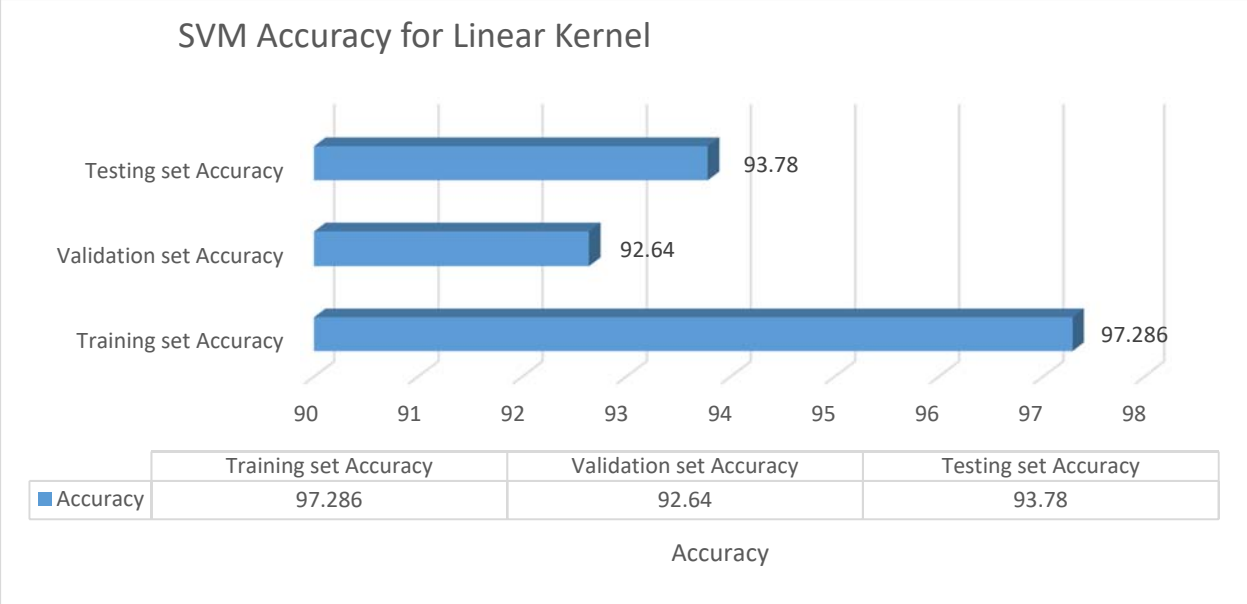


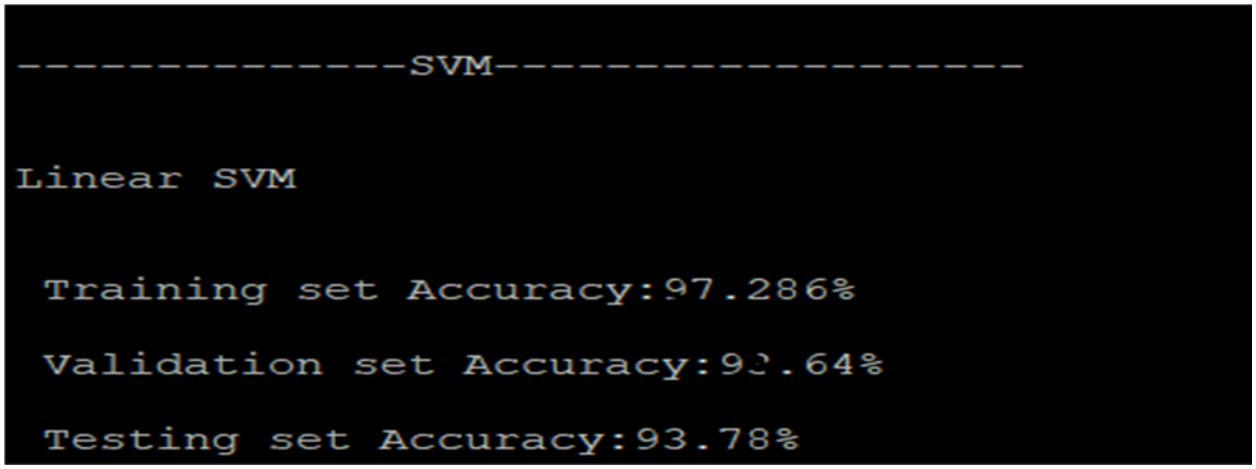
The prediction for each class is around 90% correct except for the digit'8' where the prediction is just 0.6% when we have used the train_data. The prediction for each class is around 90% even in test data except for digit '8'. On a whole the TestError is bit high for when compared to Train Error because the model is more biased when done on train data. The errors for both data are high for digits '5' and '8'.


Actual label_Test
Predicted label
0 1 2 3 4 5 6 7 8 9
962.000 0.000 9.000 4.000 1.000 11.000 9.000 2.000 34.000 10.000
0.000 1123.000 12.000 1.000 2.000 4.000 4.000 9.000 42.000 8.000
1.000 4.000 943.000 21.000 6.000 5.000 8.000 22.000 152.000 1.000
2.000 1.000 21.000 929.000 3.000 37.000 2.000 5.000 199.000 18.000
1.000 0.000 12.000 2.000 917.000 14.000 4.000 8.000 58.000 36.000
4.000 2.000 4.000 23.000 0.000 783.000 20.000 2.000 259.000 13.000
5.000 4.000 14.000 4.000 9.000 20.000 910.000 1.000 43.000 1.000
4.000 1.000 13.000 13.000 2.000 10.000 1.000 951.000 20.000 25.000
0.000 0.000 0.000 0.000 1.000 0.000 0.000 0.000 3.000 0.000
1.000 0.000 4.000 13.000 41.000 8.000 0.000 28.000 164.000 897.000
1000
800
600
400
200
0

3. RESULTS OF SVM

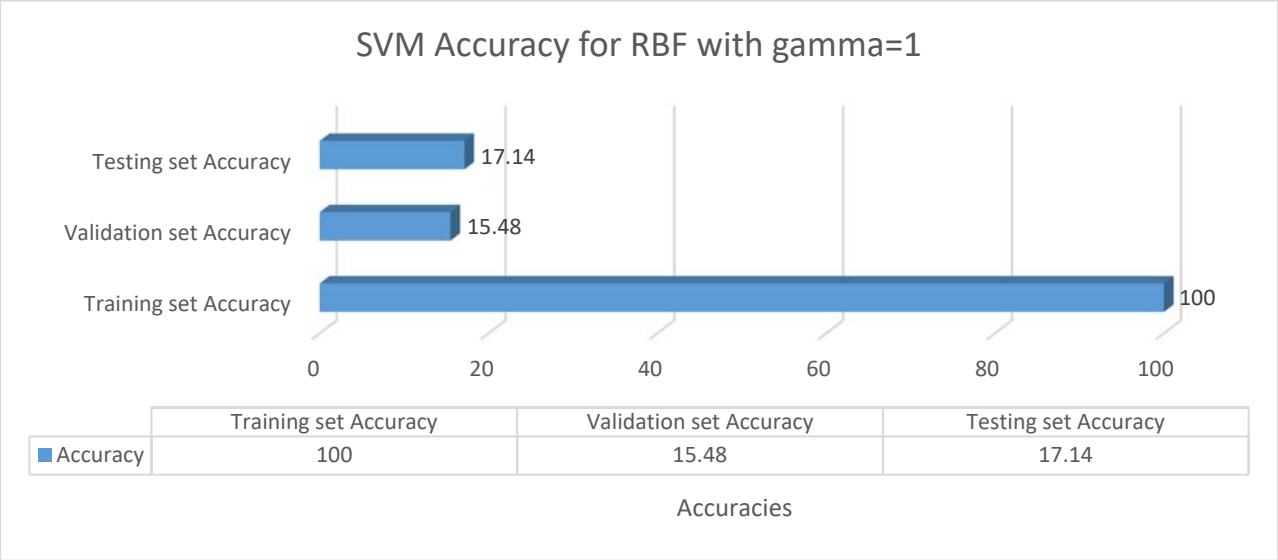
3.1. LINEAR KERNEL

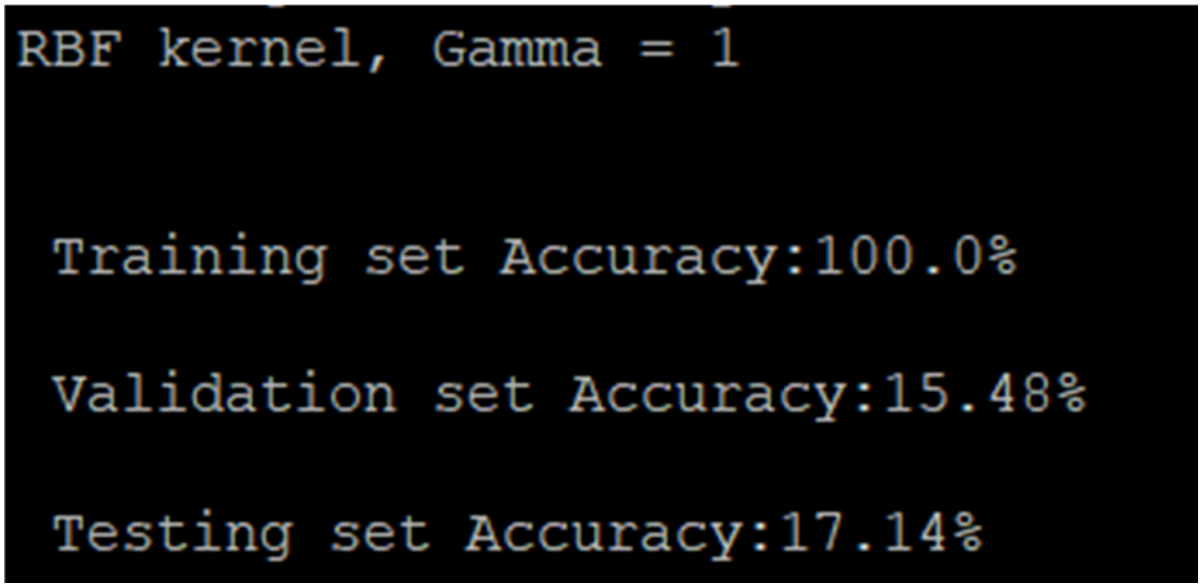




For linear kernel, the feature space is still the input space. The accuracies are pretty good, which means the classes of the data are almost linear separable. There is no need to use the SVM in a higher order feature space to complete the classification.

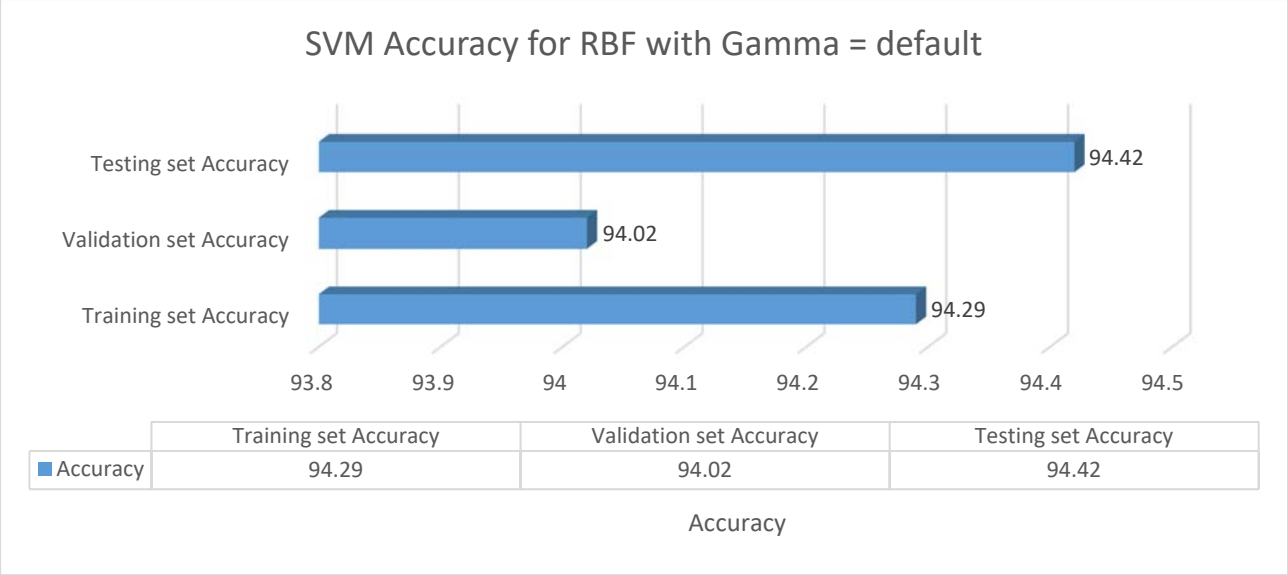
3.2. RBF KERNEL WITH Gamma=1

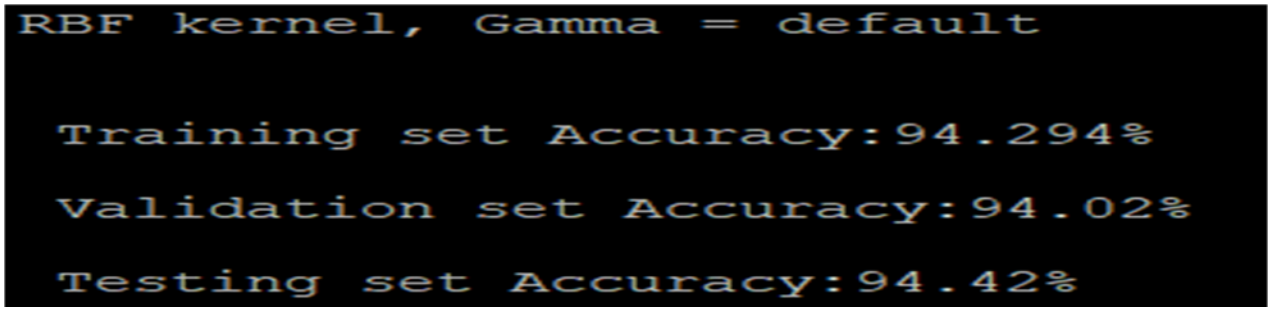




The order of the feature space of rbf kernel could be infinite. The behavior of the model is very sensitive to the gamma parameter. From the results of this case, the gamma is too large. The radius of the area of influence of the support vectors only includes the support vector itself and no amount of regularization with C will be able to prevent overfitting. Thus the training accuracy is 100%, but the prediction is very poor.

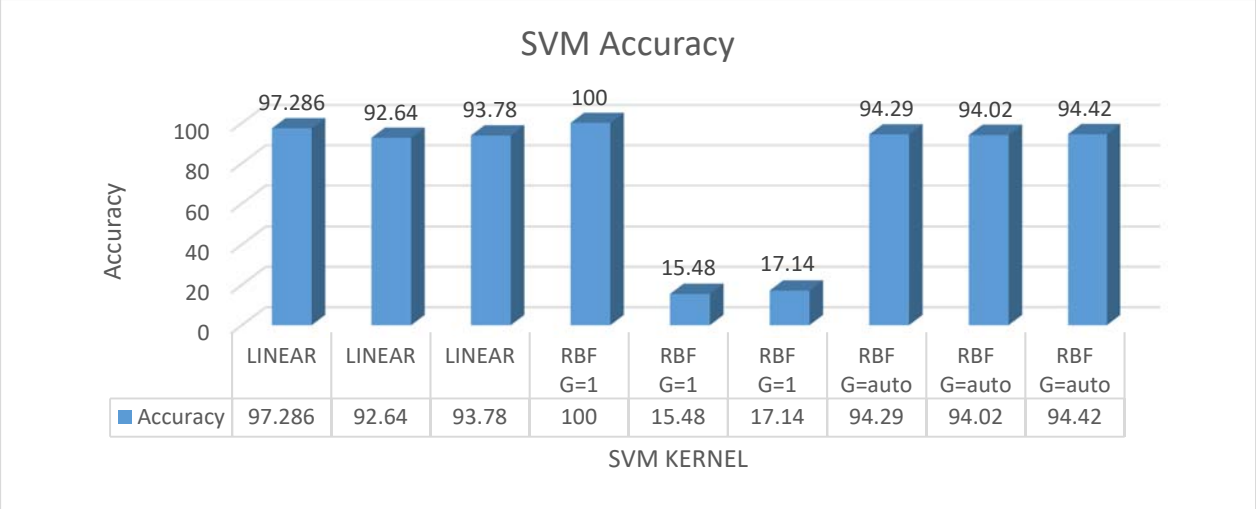
3.3. RBF KERNEL WITH Gamma=default





Following the last case, this time the gamma is not too large. Then the overfiitting disappears.

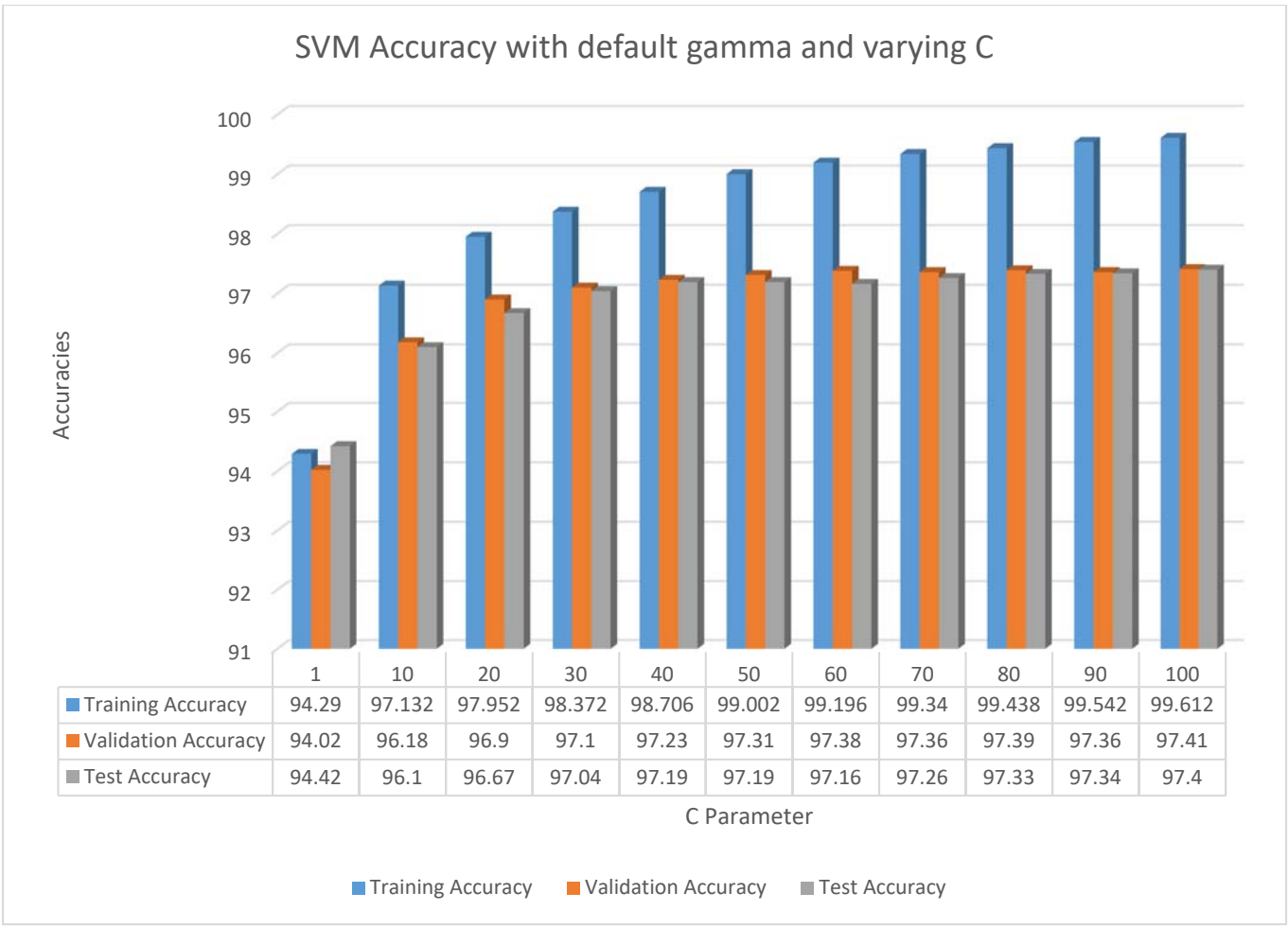
3.4 COMPARISION OF SVM WITH LINEAR AND NON LINEAR KERNELS



3.5. RBF KERNEL WITH Gamma=default AND VARYING C (1, 10, 20,……., 100)

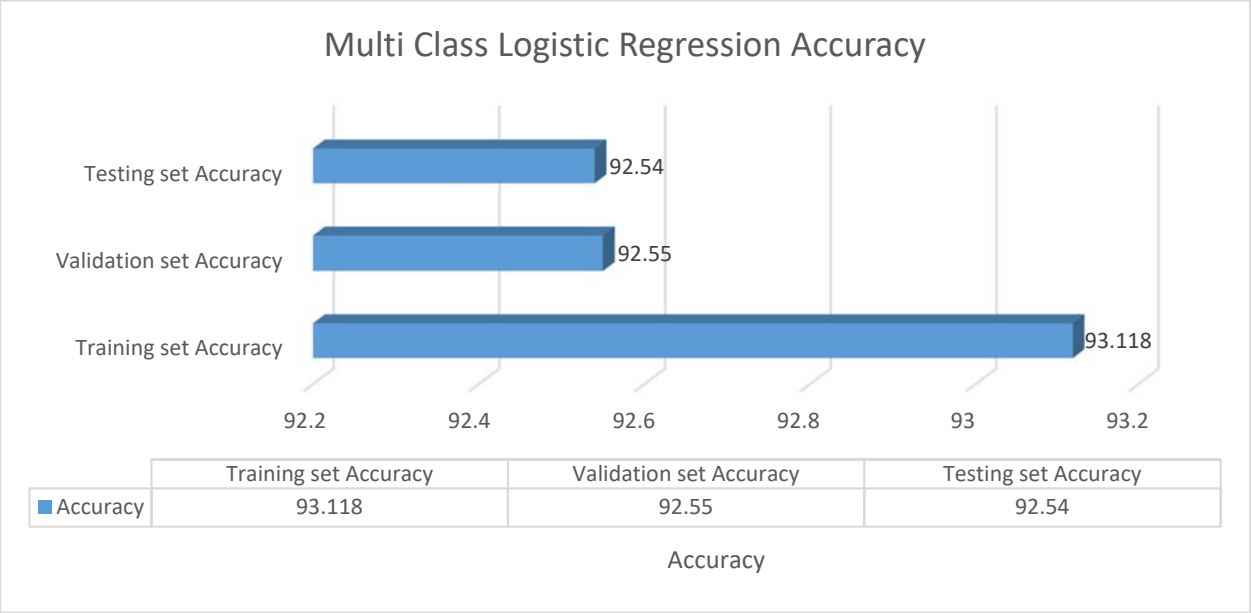
The results of this case are shown in the table and the graph. The C parameter trades off misclassification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly by giving the model freedom to select more samples as support vectors [1]. Thus with the incease of C, the accuracy of training data is closer to 100%, but the prediction doesn't improve when C is higher than 30.

```
RBF, Gamma = default, varying value of C (1; 10; 20; 30;...100)

for C =10                          for C =60


 Training set Accuracy:97.132%     Training set Accuracy:99.196%

 Validation set Accuracy:96.18%    Validation set Accuracy:97.38%

 Testing set Accuracy:96.1%        Testing set Accuracy:97.16%
for C =20                          for C =70


 Training set Accuracy:97.952%     Training set Accuracy:99.34%

 Validation set Accuracy:96.9%     Validation set Accuracy:97.36%

 Testing set Accuracy:96.67%       Testing set Accuracy:97.26%
for C =30                          for C =80


 Training set Accuracy:98.372%     Training set Accuracy:99.438%

 Validation set Accuracy:97.1%     Validation set Accuracy:97.39%

 Testing set Accuracy:97.04%       Testing set Accuracy:97.33%
for C =40                          for C =90


 Training set Accuracy:98.706%     Training set Accuracy:99.542%

 Validation set Accuracy:97.23%    Validation set Accuracy:97.36%

 Testing set Accuracy:97.19%       Testing set Accuracy:97.34%
for C =50                          for C =100


 Training set Accuracy:99.002%     Training set Accuracy:99.612%

 Validation set Accuracy:97.31%    Validation set Accuracy:97.41%

 Testing set Accuracy:97.19%       Testing set Accuracy:97.4%
```



	1	10	20	30	40	50	60	70	80	90	100
Training Accuracy	94.29	97.132	97.952	98.372	98.706	99.002	99.196	99.34	99.438	99.542	99.612
Validation Accuracy	94.02	96.18	96.9	97.1	97.23	97.31	97.38	97.36	97.39	97.36	97.41
Test Accuracy	94.42	96.1	96.67	97.04	97.19	97.19	97.16	97.26	97.33	97.34	97.4

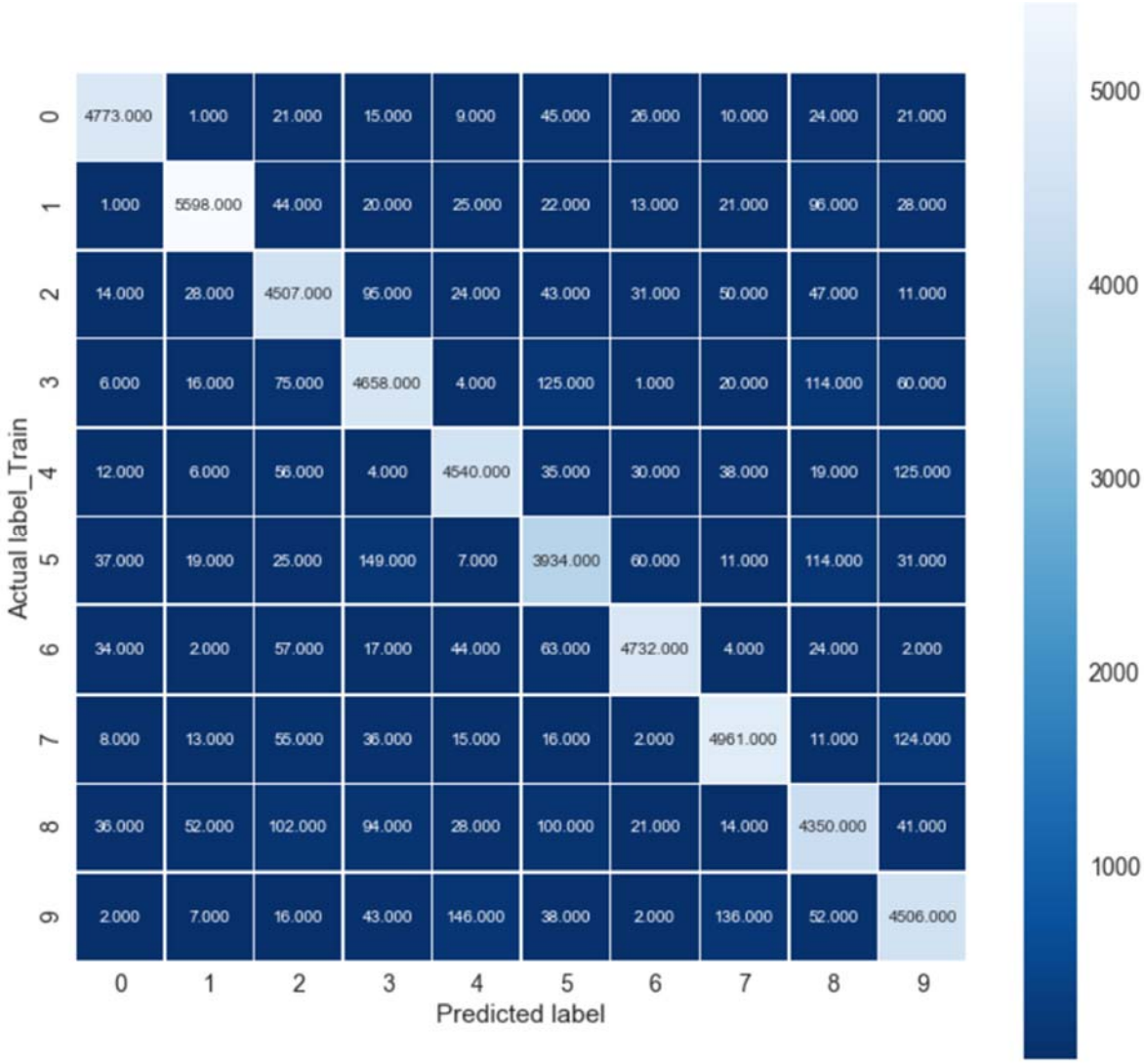
4. RESULTS OF MULTI CLASS LOGISTIC REGRESSION



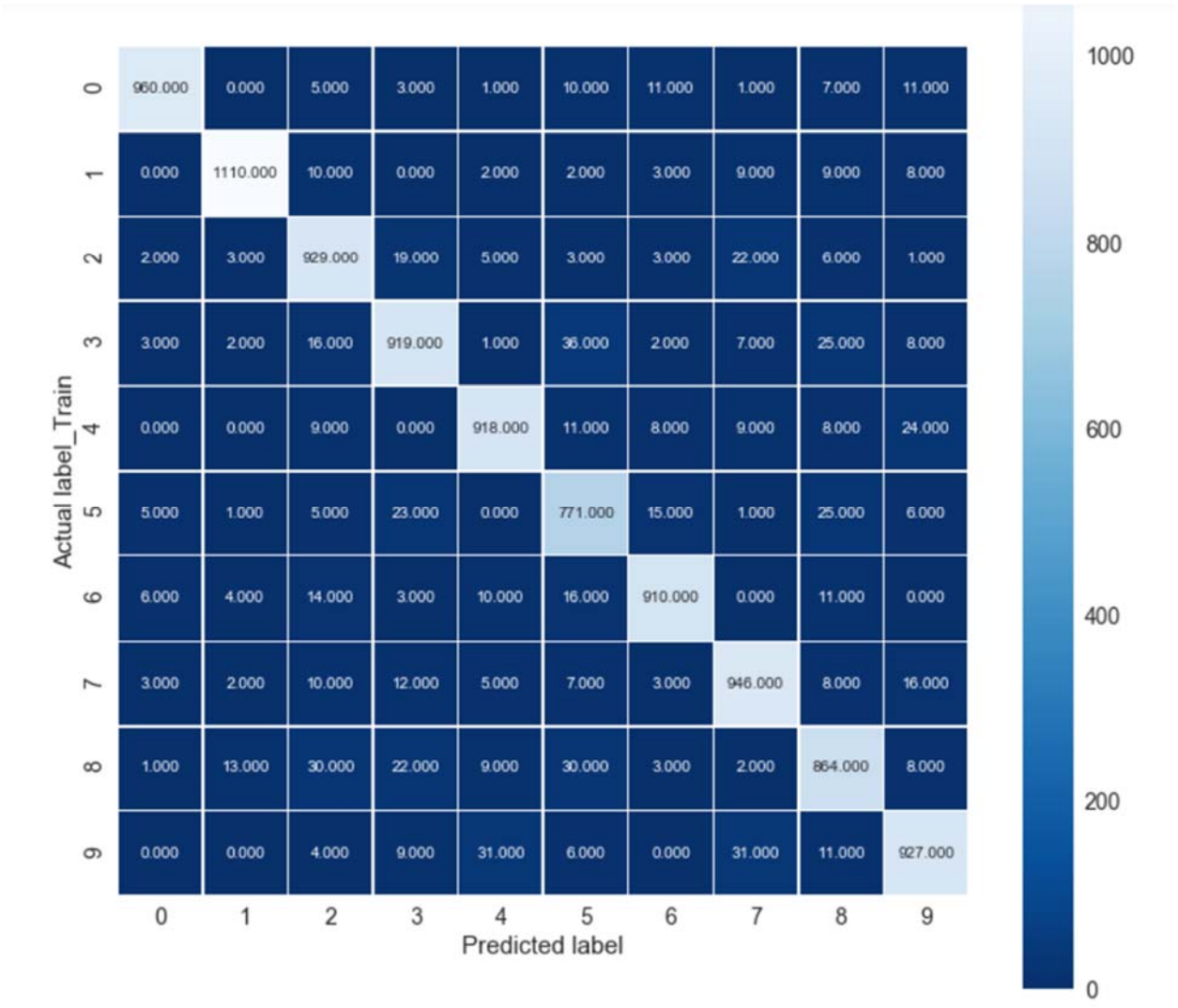

```
Training set Accuracy:93.118%
Validation set Accuracy:92.55%
Testing set Accuracy:92.54%
```

Following the discussion in Part 1, multi-class logistic regression improves the performance of the binary logistic regression not only for the accuracy but also for the run time. The art behind the multi-class logistic regression is very easy. Firstly, the cost function can include all of the weights of the ten classes by using the softmax function. Thus this time we can find the global minimum of the whole map of the ten classes. That's why the accuracies are increased. Secondly, this time we only need to do the optimization one time for the ten classes. Thus the run time of this algorithm is only about one tenth of the run time of the binary logistic regression.

4.1. TRAIN AND TEST ERROR



Even in Multiclass Logistic Regression the the prediction in each class seems to be around 90% for the train_data. It is similar even in the case of test data, but the test error is a bit high when comapred to the train error, this is because the model is biased when applied on train data compared to the test data. In both train data and test data the error was more for digit ’5’.



4.3. COMPARISION

Multiclass logistic regression performs better than one-vs-all strategy. This is expected since with given small size data of MNIST Handwritten digits and input digits are not closely co-related. Hence, In One-VS-Other approach, true probability for a digit clearly overpowers other digits in most cases rather than the case where a single digit is compared against all the digists.One-VS-Rest performs at par with One-V-Other in cases where data is discriminable with larger probabilities between each possible outcome.

Digit '8' is predicted 86% correctly in Multiclass Logistic Regression where as only 0.3% in logistic regression. This example shows that Multiclass Logistic regression outperforms logistic regression.