Classification and Regression

CSE574 Introduction to Machine Learning
Spring 2019

Programming Assignment 3

Group No : 33
Manasi Kulkarni (Student No : 50288702)
Priyanka Pai (Student No : 50295903)
Kundan Kumar (Student No : 50301604)

Abstract

This project involves classifying handwritten digits into labels using logistic regression. We have implemented binary logistic regression as well as multi-class classification using logistic regression. This project also involves implementation of Support Vector Machines to compute accuracy on the randomly sampled data from the provided dataset using linear kernel function and radial basis function with varying gamma setting and C values.

1. Concepts and definitions

1.1 Logistic Regression

Unlike linear regression, logistic regression is atomic i.e. either the event happens or it does not happen. It uses binary data. Thus, in simple logistic regression or Binomial Logistic Regression, the output is either 0 or 1. Another type of logistic regression is called Multinomial Logistic Regression, where the output consists of multiple values. To find relationship between variables, logistic regression uses natural logarithm function, which can be used to predict future results.

1.2 Support Vector Machines

SVMs are used to find decision boundaries(also known as hyperplanes) to classify the data points. Support vectors are the points which lie close to the decision boundary. Using them, we can maximize the margin of the classifier. SVMs are widely used in classification of images, handwritten digits and text. The implementation learns the SVM model and makes predictions with respective accuracies for training, validation and test data.

Gamma Parameter :

The gamma parameter determines the influence of a single training example. Small gamma implies high variance and large gamma implies low variance.

C Parameter:

The C parameter gives us the changes in the misclassification on the train data. The smaller value of C would mean that the margin classifier is greater and a large value of C means that the margin is smaller and the hyperplane has all the points in the training data

Logistic regression considers all the points within a dataset for finding a separating hyperplane;on the other hand,SVM just works with the samples which are close to each other and close to the margin(which are called support vectors)and tries to provide the maximum possible distance between the hyperplane.

• In general, logistic regression learns data separation using hyperplane, while SVM learns the maximum margin: this justifies the performance difference in the first instance.

• Logistic regression works well with a low number of input features, while SVM performs better when the number of input dimensions is high, like in this case(716 pixels).

• Logistic regression works well when the input data does not give direct information on the output. In this case, the predictors are pixels and directly determine the response. Therefore, SVM performs better in this case.

Implementation

1. Implementation of Binary Logistic Regression

Logistic regression is used when we have 2 dependent variables. It is used to explain the relationship between dependent and independent variables. These independent variables can be binomial, multinomial or ordinal.

The weights are trained using gradient descent, which is used to minimize the errors.

The formula is for calculating error in the training phase is:

$$E(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^{N}\{y_n \ln \theta_n + (1-y_n)\ln(1-\theta_n)\}$$

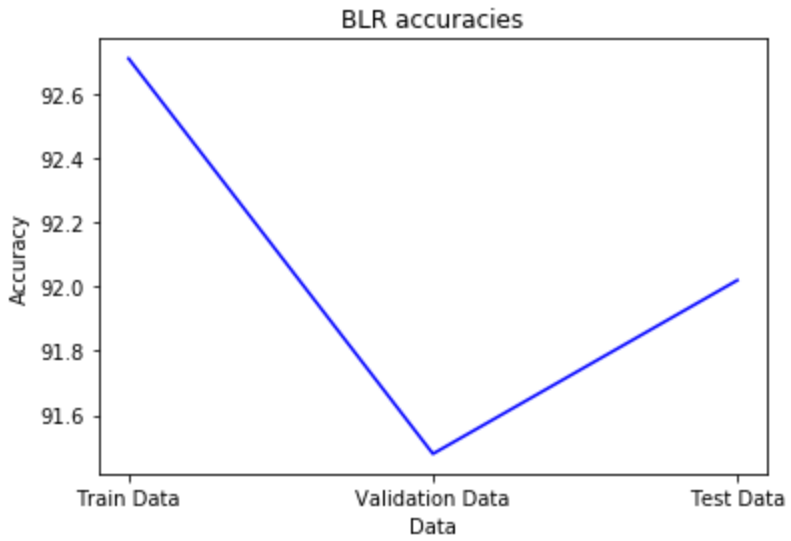
The calculation of error gradiance is done using:

$$\nabla E(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}(\theta_n - y_n)\mathbf{x}_n$$

We have implemented binary logistic regression on MNIST dataset, for classifying handwritten digits. The results obtained are as follows :

Data	Accuracy in %
Training data	92.71
Validation data	91.48
Test data	92.02

From the results obtained, it can be observed that the performance of our model is good on training data, validation data and test data.



	Logistic Regression	**Multiclass Logistic Regression**
1	Dependent Variable (DV) is categorical.	Dependent variable has more than two outcome categories
2	Model is used to estimate the probability of a binary response based on one or more predictor (or independent) variables (features).	Model that is used to predict the probabilities of the different possible outcomes of a categorically distributed dependent variable, given a set of independent variables (which may be real-valued, binary -valued, categorical-valued, etc.

Confusion Matrix :

Confusion Matrix for blr Training Set

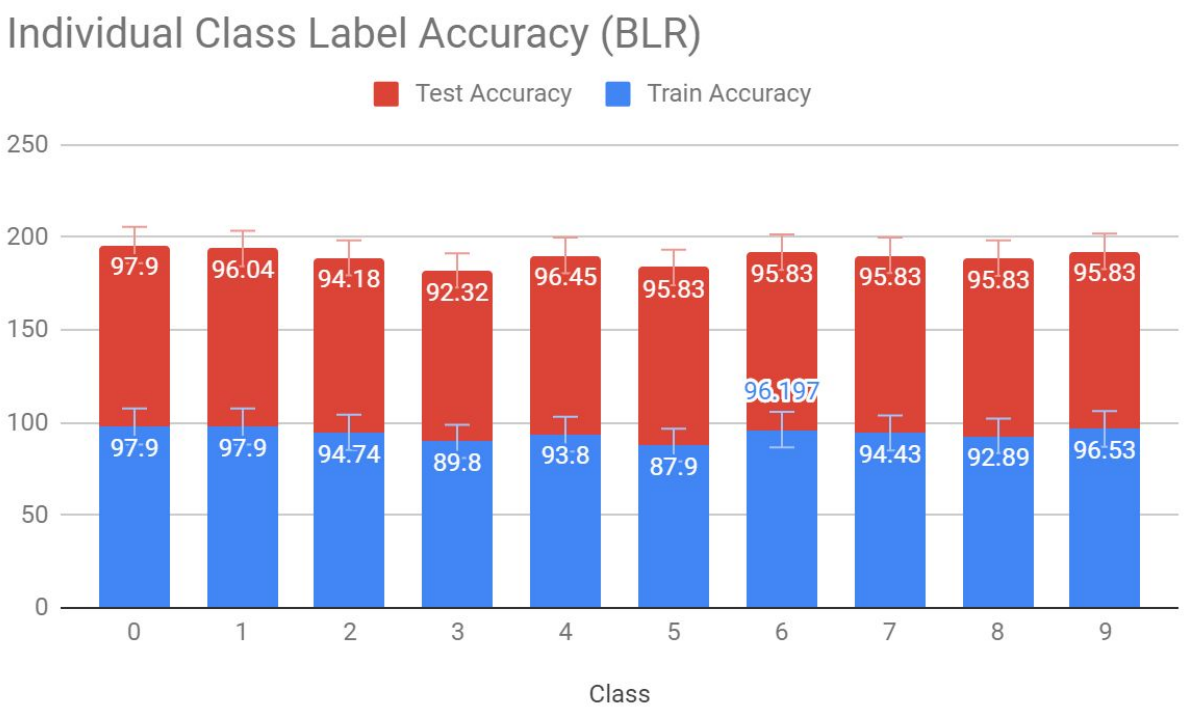
```
[[4821    1    9    7    8   18   22    5   30    2]
 [   1 5627   28   10    3   16    2   11   38    6]
 [  31   36 4528   64   47   16   50   62  109   15]
 [  19   21  121 4608    8  134   20   43  104   53]
 [  10   20   21    6 4545    9   25   12   46  148]
 [  44   18   32  135   42 3889   83   17  110   51]
 [  25   13   31    3   22   65 4731    4   22    2]
 [  11   21   47   11   40   13    3 4972   11  136]
 [  36  107   54  115   27  124   32   19 4255   82]
 [  25   19   14   82  166   37    1  160   44 4401]]
```

Confusion Matrix for blr Testing Set

```
[[ 960  0       1    3   1       5    5   3       1    1]
 [   0 1116    3       1       0    1   4       1    9   0]
 [   9   10  918   18   10      4   12   11   36       4]
 [   4    0   17  922    2   19  4   14   18   10]
 [   1    3       5       2  918 0   10    2      4   37]
 [  10    2       1   44   13  757   17    9   30       9]
 [   9    4       7       2    4   19  909    1   3       0]
 [   2    9   21  6    7   2       1  951 2   27]
 [  12   15      6   19   14   28       7   12  850   11]
 [   8    8       1   12   35   12       1   22   10  900]]
```

Individual class label accuracy for BLR :

Class	Train Accuracy	Test Accuracy
0	97.9	97.9
1	97.9	96.04
2	94.74	94.18
3	89.8	92.32
4	93.8	96.45
5	87.9	95.83
6	96.197	95.83
7	94.43	95.83
8	92.89	95.83
9	96.53	95.83



2. Implementation of Multi-class Logistic Regression

For implementing MLR, we have used the following formulae :

$$P(y = C_k|\mathbf{x}) = \frac{\exp(\mathbf{w}_k^T \mathbf{x})}{\sum_j \exp(\mathbf{w}_j^T \mathbf{x})}$$

The error for the k(=10) classes is calculated as follows:

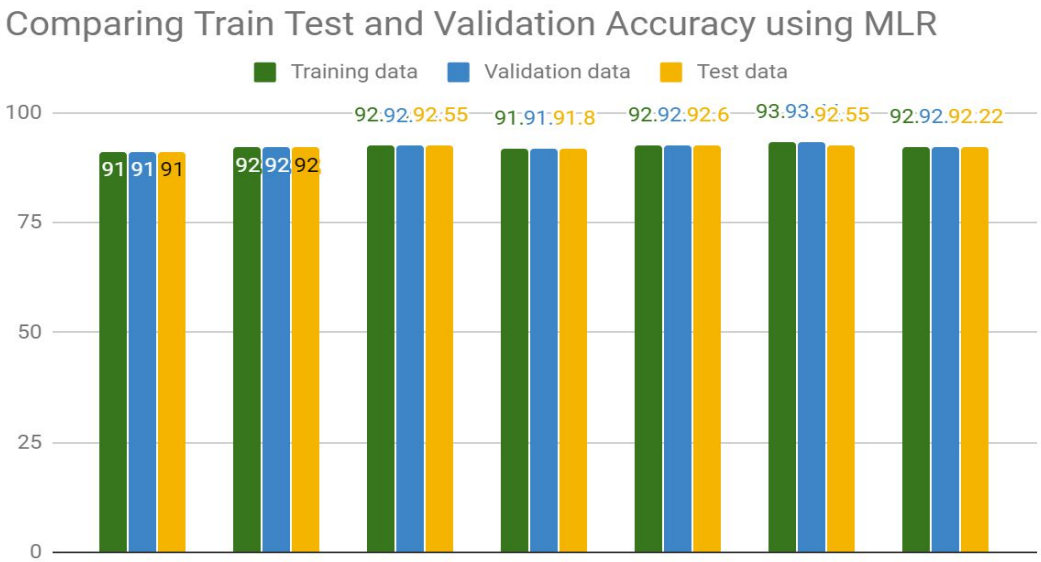
$$E(\mathbf{w}_1, \cdots, \mathbf{w}_K) = -\ln\ P(\mathbf{Y}|\mathbf{w}_1, \cdots, \mathbf{w}_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk} \ln\ \theta_{nk}$$

The below formula is used for evaluating the gradiance:

$$\frac{\partial E(\mathbf{w}_1, \cdots, \mathbf{w}_K)}{\partial \mathbf{w}_k} = \sum_{n=1}^{N} (\theta_{nk} - y_{nk})\mathbf{x}_n$$

For our dataset, we got the following accuracies on training data, validation data and test data (Plot for first 7 values) :

Data	Accuracy in %
Training data	93.44
Validation data	92.48
Test data	92.55



Multinomial regression is better in terms of accuracy in comparison with binomial regression because each input belongs exactly to 1 class alone.

Also, from the below graph we can conclude the same.

Confusion Matrix for mlr Training Set

```
[[4786    1   12    7   11   33   30    7   32    4]
 [   1 5592   26   17    6   19    2   13   58    8]
 [  23   45 4503   72   58   24   59   53  108   13]
 [  14   18   95 4654    4  148   15   39  105   39]
 [   8   20   21    7 4576    6   42   13   24  125]
 [  39   13   36  117   34 3963   68   18  102   31]
 [  23   11   29    1   24   52 4758    2   16    2]
 [   8   16   49   18   34    9    4 4989   14  124]
 [  22   75   51  103   16  113   23   16 4387   45]
 [  17   18    9   55  126   30    2  134   42 4516]]
```

Confusion Matrix for mlr Testing Set

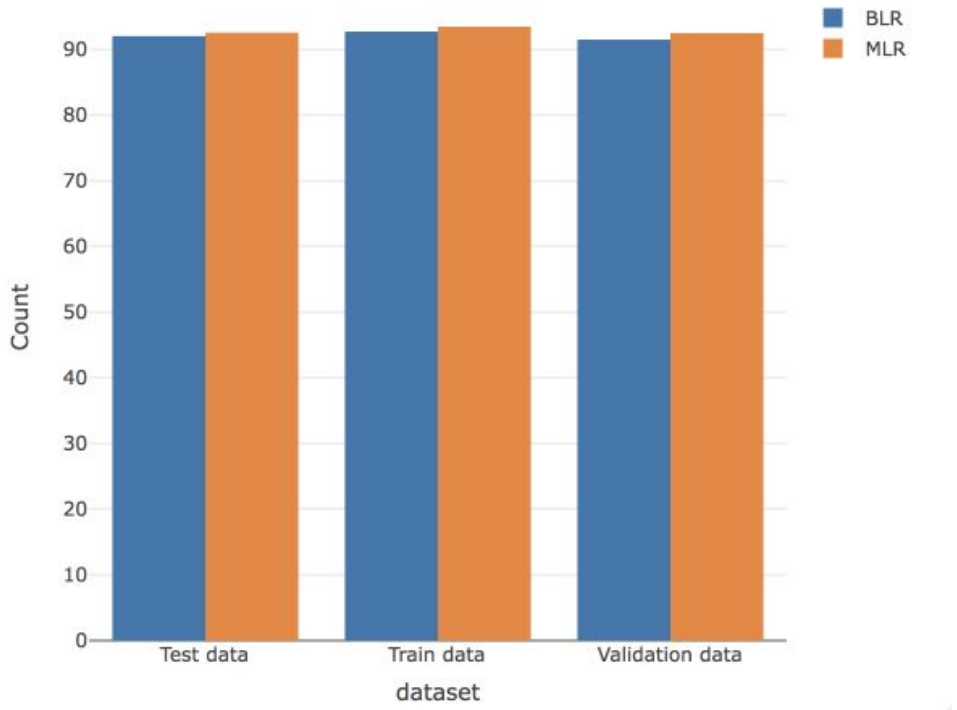
```
[[ 960    0    0    3    0    6    6    4    1    0]
 [   0 1110    3    2    0    2    4    2   12    0]
 [   6    8  924   16   10    3   14    8   39    4]
 [   4    1   20  914    0   25    3   10   26    7]
```

```
[  1   1   6   2 921   0   9   4   9  29]
[ 10   2   2  37  10 773  15   6  30   7]
[  9   3   4   2   7  15 914   3   1   0]
[  1   9  19   6   6   2   0 952   2  31]
[  9   8   6  26   9  23  10   8 868   7]
[ 11   8   0  10  28   5   0  20   8 919]]
```

Individual class label accuracy for MLR :

Class Label	Train Data	Test Data
0	97.9	96.3
1	92.46	98.5
2	88.9	97.65
3	92.46	91.86
4	97.68	97.7
5	95.8	97.4
6	93.92	97.1
7	92.04	96.8
8	90.16	96.5
9	88.28	96.2

The comparison of BLR and MLR is as follows :



Comparison of Binary and Multi-class Logistic Regression:

Multiclass Regression in a way is a binary logistic model. In simpler terms , if we consider binary logistic regression to be a one-vs-rest approach its performance is comparatively lesser than the one-vs-all approach that we take for multiclass regression

3. Implementation of Support Vector Machines

Support Vector Machines are models used for classification and regression. SVM assigns new points or examples to either of the categories, which makes it non-probabilistic. It is a linear model, as the classification is done based on a linear boundary. SVMs can also be used for non-linear classification with the help of kernel trick, where inputs are mapped to higher dimensional space.

Few parameters are used for non-linear SVM with a Gaussian radial basis function kernel. They are C and gamma. C is used to control the influence of individual support vectors. It is useful to avoid misclassification of data. Gamma is a free parameter; small gamma implies high variance(large margin) and thus in turn indicates more influence of a support vector. Large gamma implies low variance and thus less influence of a support vector.

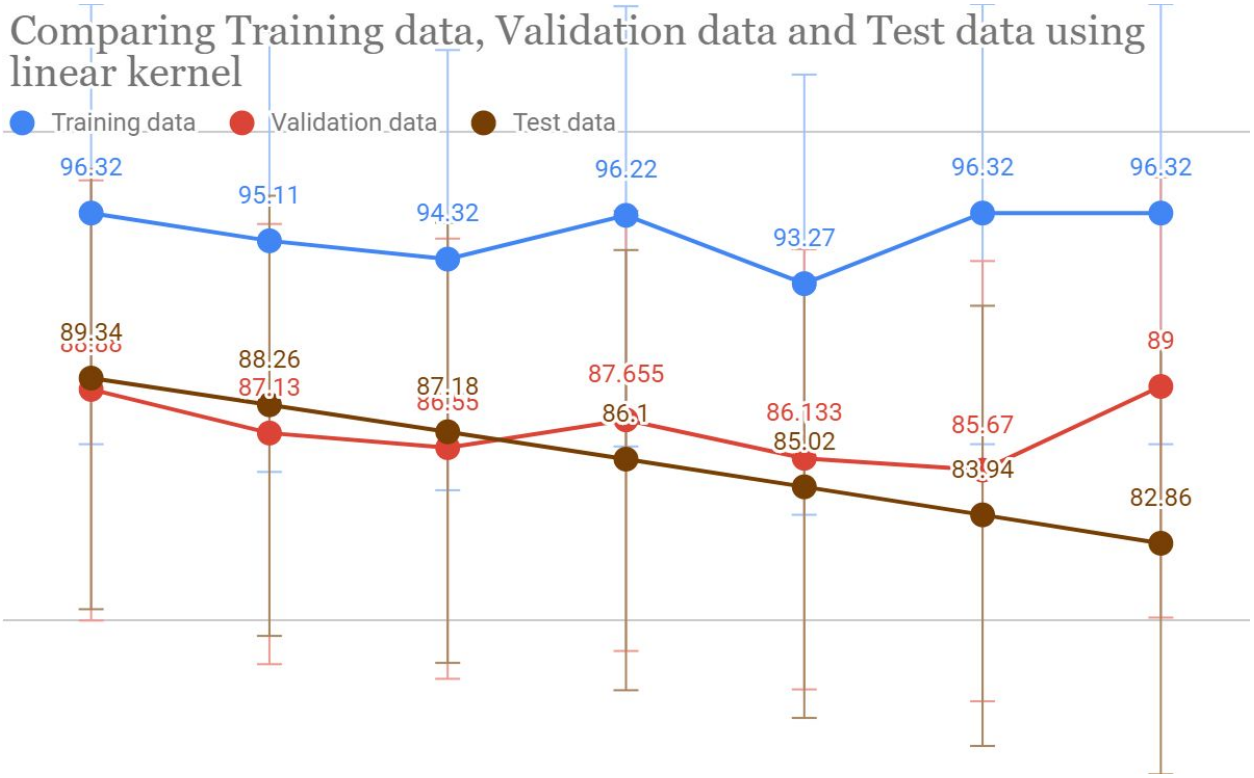
We have used Support Vector Machine tool in sklearn.svm to perform classification on our data set. We have randomly sampled 10000 samples to learn SVM models, as SVMs are difficult to scale well to large datasets.

We have calculated accuracies on training data, validation data and test data using the following parameters :

1. Using linear kernel (all other parameters are kept default)

Linear kernel is used when the number of features is more in a dataset. It is mostly used when the data can be separated linearly, meaning it can be separated using a single line. Following are the accuracies we got on training data, validation data and testing data using linear kernel function :

Data	Accuracy in %
Training data	96.32
Validation data	88.88
Test data	89.34



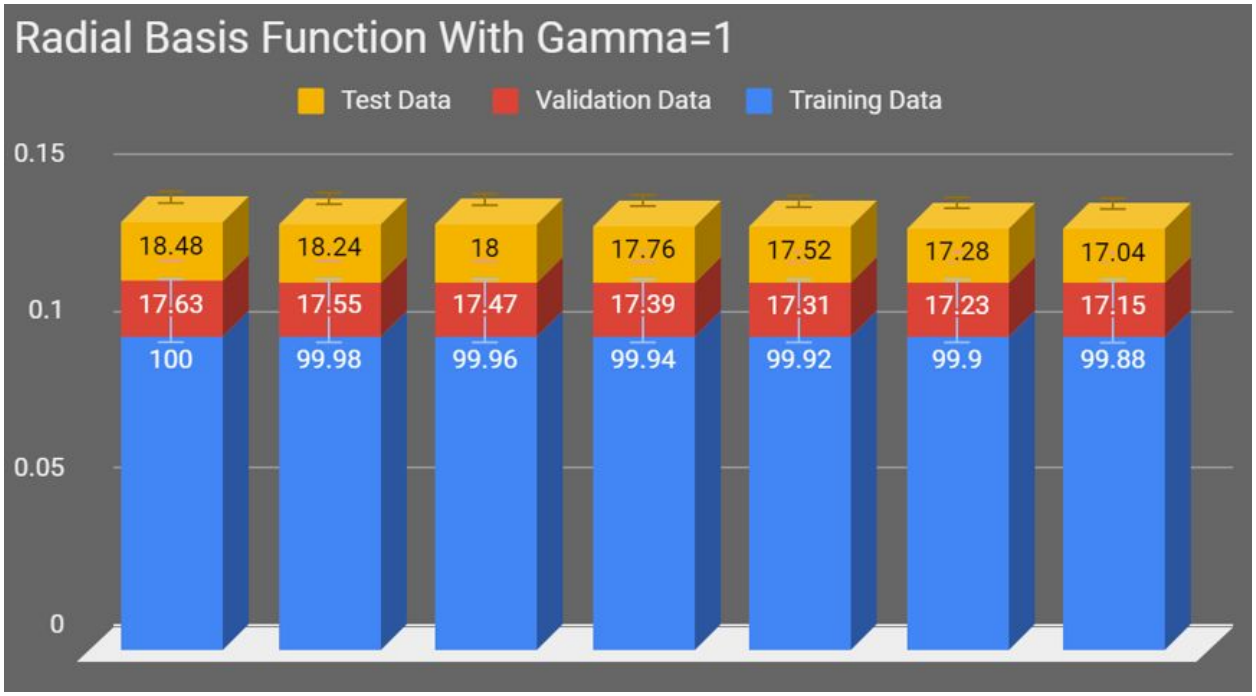
It can be observed that using linear kernel results in better results on the data sets. Our data contains large number of features. This also infers that our **data is linearly separable**.

2. Using radial basis function with value of gamma setting to 1 (all other parameters are kept default)

Radial basis function or RBF is one of the most popular kernel functions used in Support Vector Machine classification. It is a real-valued function. The value depends on the distance from the origin or any particular point considered as the center. The distance is normally calculated using Euclidean distance.

Here, we are using RBF for classification by setting gamma to 1. The results we obtained are as follows :

Data	Accuracy in %
Training data	100
Validation data	17.63
Test data	18.48



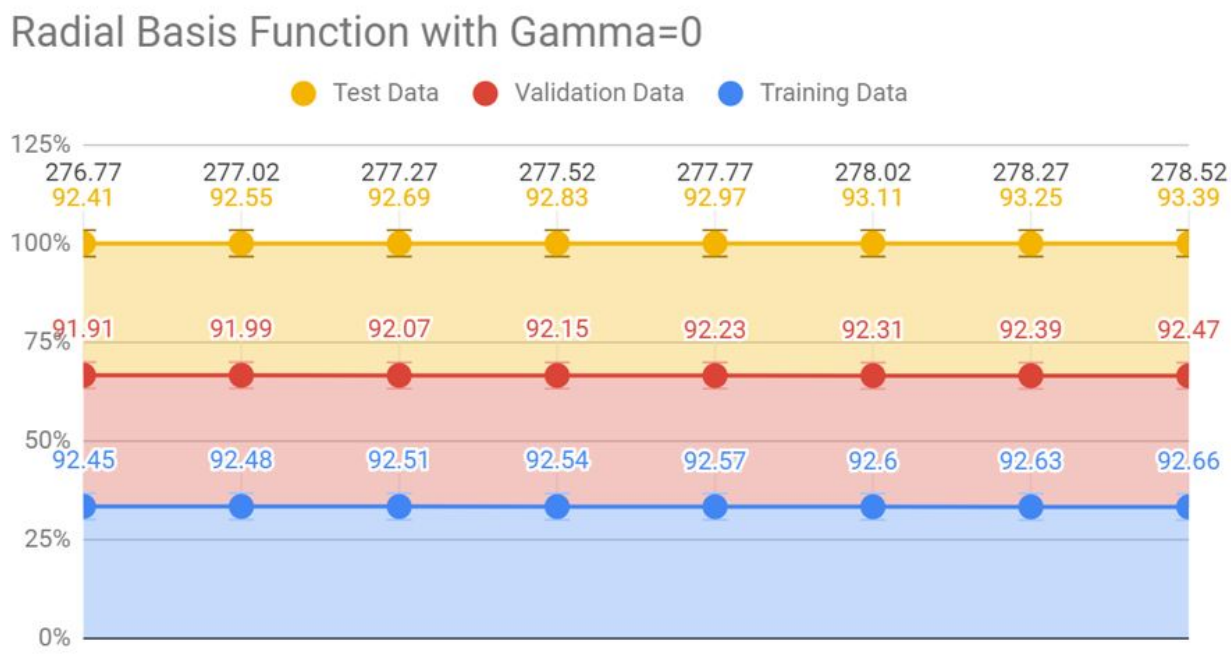
It can be observed that setting gamma value to 1 results in **overfitting of training data**.

3. Using radial basis function with value of gamma setting to default (all other parameters are kept default)

We tested SVM by setting gamma to default value. The results obtained in this case are as follows :

Data	Accuracy in %
Training data	92.45
Validation data	91.91
Test data	92.41

Setting gamma value to default, we obtain much better results on each of the data sets. It can be observed that there is **no overfitting** of data, as was the case when setting gamma value to 1.

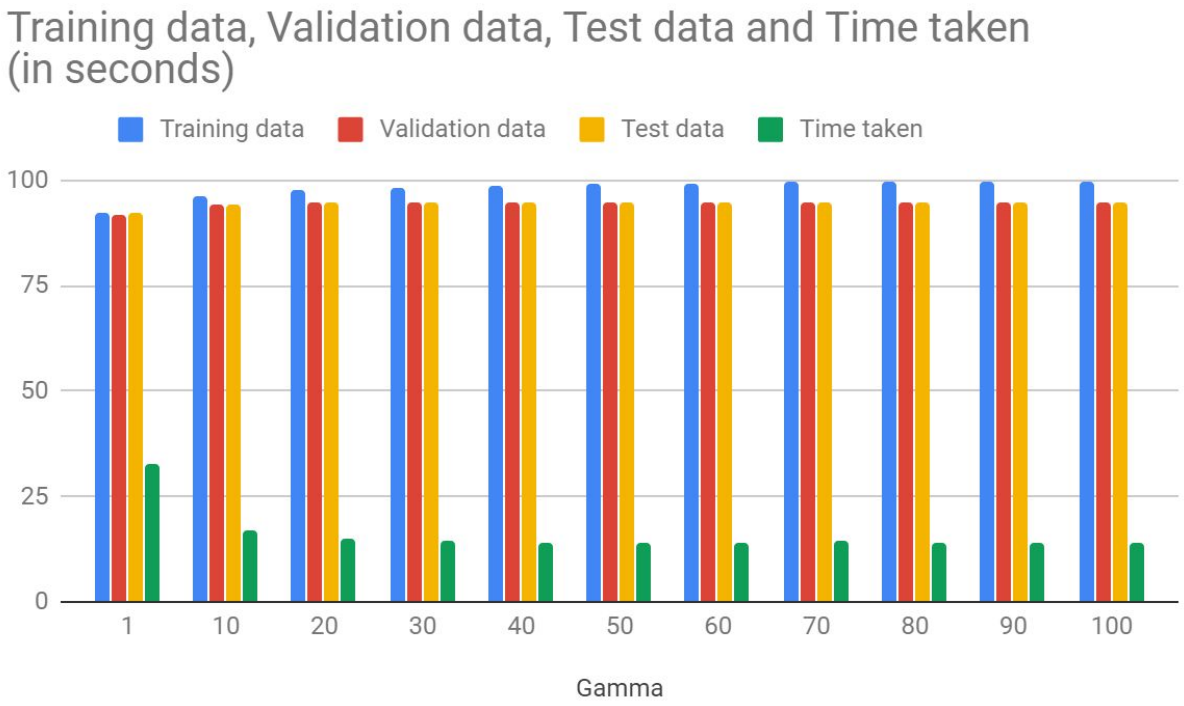


4. Using radial basis function with value of gamma setting to default and varying value of C : 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100

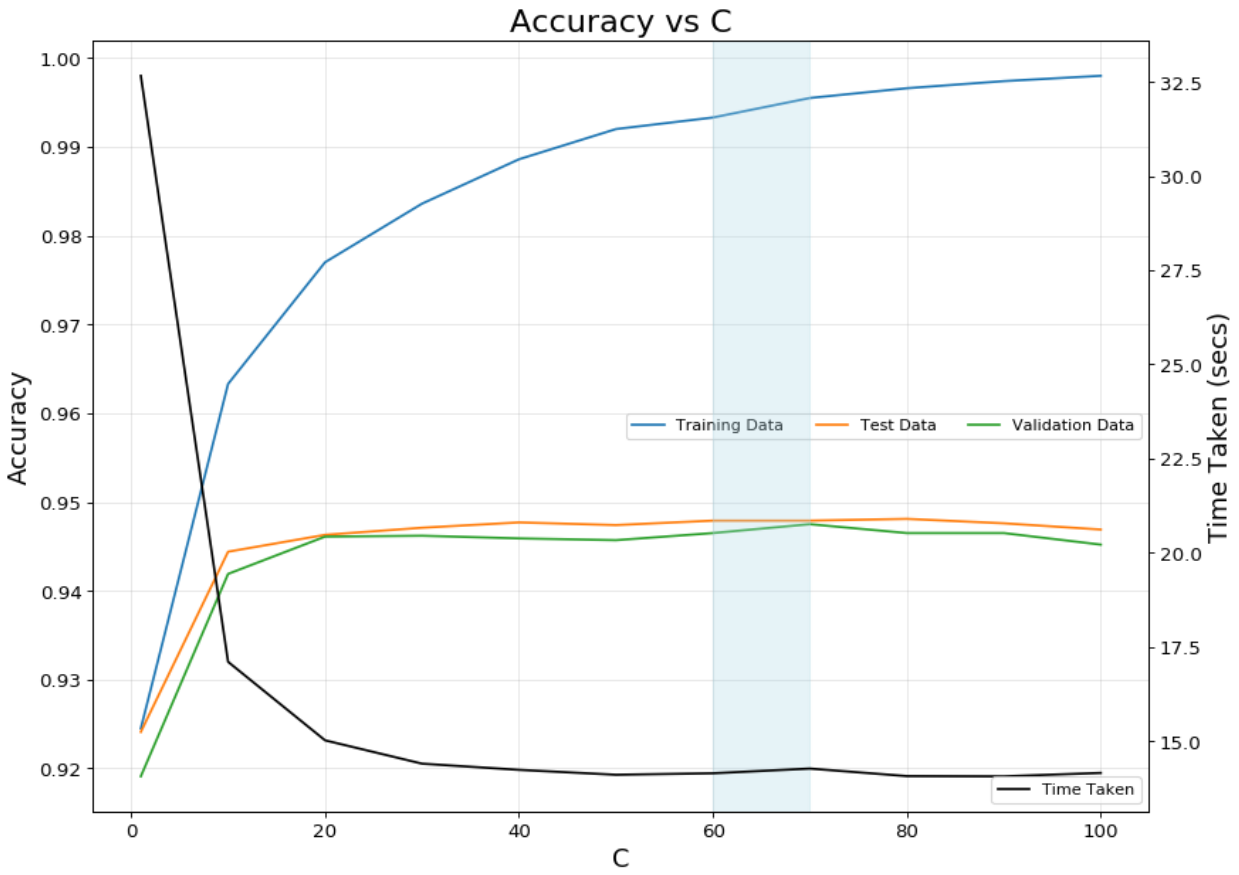
For the above mentioned values of C, our SVM generated the following accuracies on training data of 10000 samples, validation data and testing data :

Gamma	Accuracy			Time taken (in seconds)
	Training data	Validation data	Test data	
1	92.45	91.91	92.41	32.65
10	96.33	94.19	94.44	17.10
20	97.70	94.61	94.63	15.01
30	98.36	94.62	94.71	14.39
40	98.86	94.59	94.77	14.23
50	99.20	94.57	94.74	14.10
60	99.33	94.65	94.79	14.14

70	**99.55**	**94.75**	**94.79**	**14.26**
80	99.66	94.65	94.81	14.06
90	99.74	94.65	94.76	14.06
100	99.80	94.52	94.69	14.15



Following is the graph of accuracies wrt C as well as the time taken :



Conclusion :

Using SVM with the Gaussian kernel **gives higher accuracy** because of the flexibility to transform the data into a space of any dimension.

The decision between whether to choose Linear Kernel of Radial Basis Function is directly proportional to the number of Observations we have. Higher the number of Observations, we go for Gaussian Kernel

The value of C is directly proportional to the level of accuracy that we define.Error will be low when the weight that we choose is low.But this is fine when we are considering for training phase. Greater the margin , greater the range of misclassified samples.There might be more data that is overfit due to the higher range of values for C .The accuracy on validation and test set saturates pretty fast.

Gamma value controls the influence of each training data on the learned hyperplane.

When Gamma = 1, we have observed a case of overfitting on training data.

Similarly, the impact of setting higher C(penalty factor) value is the increase in the weight of each error term and a smaller margin hyperplane. Thus, accuracy increases with higher C values.

While choosing the optimal value for C, we should ensure that it does not cause either overfitting or underfitting of the model.