CSE 574 INTRODUCTION TO MACHINE LEARNING

PROGRAMMING ASSIGNMENT 3

CLASSIFICATION AND REGRESSION

Team 8

Deepti Chavan (deeptisu)
Kamalakannan Kumar (kkumar2)
Sushmita Sinha (ssinha7)

TABLE OF CONTENTS

PROBLEM 1: LOGISTIC REGRESSION

	Accuracy	
Training Set	**Validation Set**	**Testing Set**
84.976%	83.74%	84.27%

Logistic regression tries to find a hyper plane which separates the different points in a data set such that the data set gets classified. It considers all the points in the data set and classifies them. It creates 10 binary classification models, optimizes the algorithm for each class, and then merges the models, distinguishing one class from all other classes.

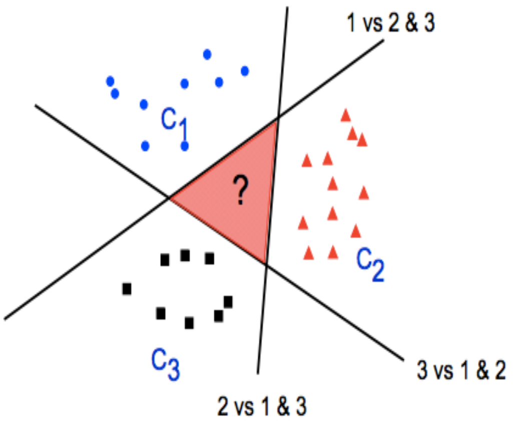
Logistic regression happens to give better results when we don't have pre determined output classes and low number of input features are present.

PROBLEM 2: DIRECT MULTI CLASS LOGISTIC REGRESSION

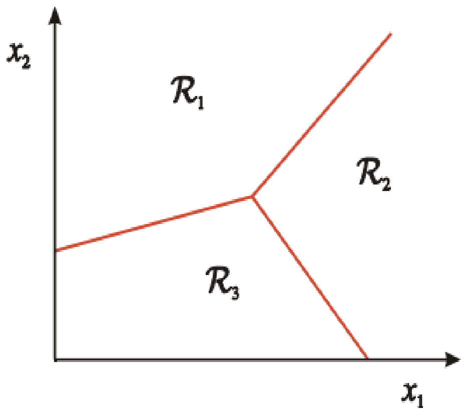
	Accuracy	
Training Set	**Validation Set**	**Testing Set**
93.11%	92.51%	92.47%

As we increase the number of classification classes, we increase the number of hyper-planes that we try to separate the data in. One-vs-all strategy tries to change multiple classes into two classes, and construct one logistic classifier for each class. Hence it in-turn uses a 2 class Logistic regression internally. It creates multiple binary classification models, optimizes the algorithm for each class, and then merges the models.

Multiclass logistic regression has C weight vectors to learn. Multiclass logistic regression tries to create a logistic regression model that can be used to predict multiple values. Hence we obtain a better accuracy than one-vs-all strategy where the regression tries to classify using one classifier classifying into 10 different classes simultaneously.

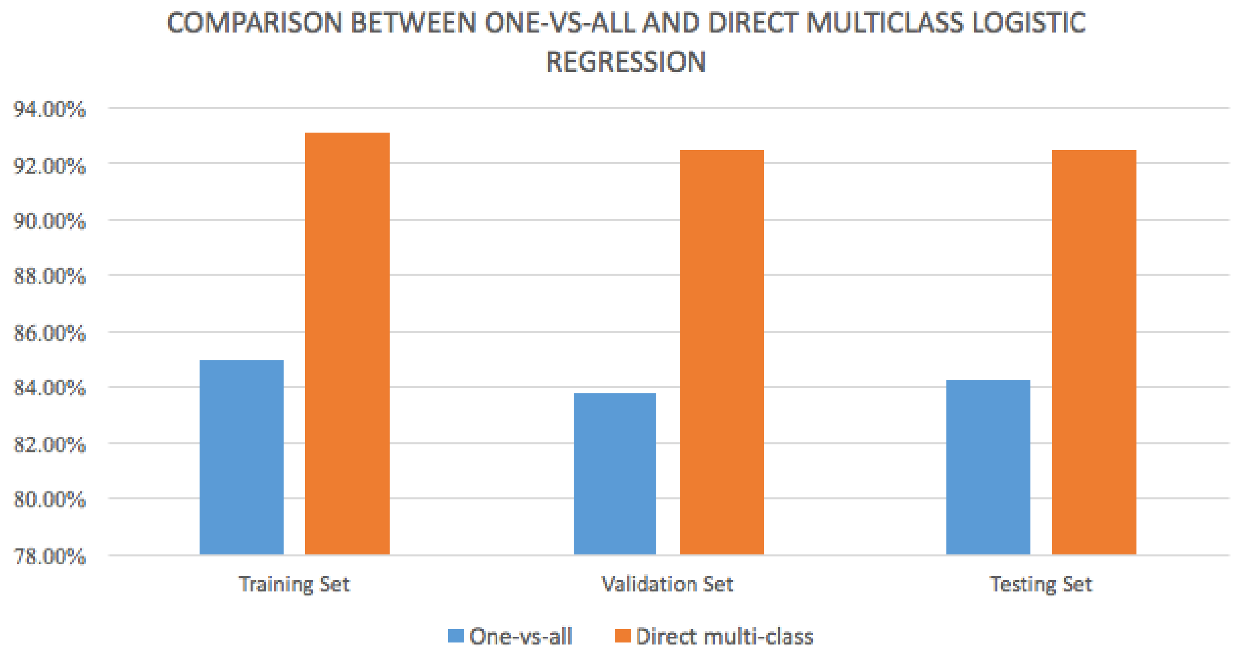


One-vs-all strategy



Direct multiclass strategy

Fig source - http://www.robots.ox.ac.uk/~az/lectures/ml/2011/lect4.pdf



COMPARISON BETWEEN ONE-VS-ALL AND DIRECT MULTICLASS LOGISTIC REGRESSION

PROBLEM 3: SUPPORT VECTOR MACHINES

We can observe that SVM performs better than Logistic regression as SVM only learns from the support vectors and not all the data points. The support vectors separate the classes in such a way that support vectors are present on the maximum margin and hence performs better in testing phase. Thus the decision boundary learnt in SVM is not just the one that separates the classes but is the best boundary separating the classes maximizing the margin around the decision boundary.

	SVM with Linear Kernel	
Training Set	**Validation Set**	**Testing Set**
97.286%	93.64%	93.78%

Linear kernel tries to separate the points in a linear space. Linear kernel performs better when the data is linearly separable.

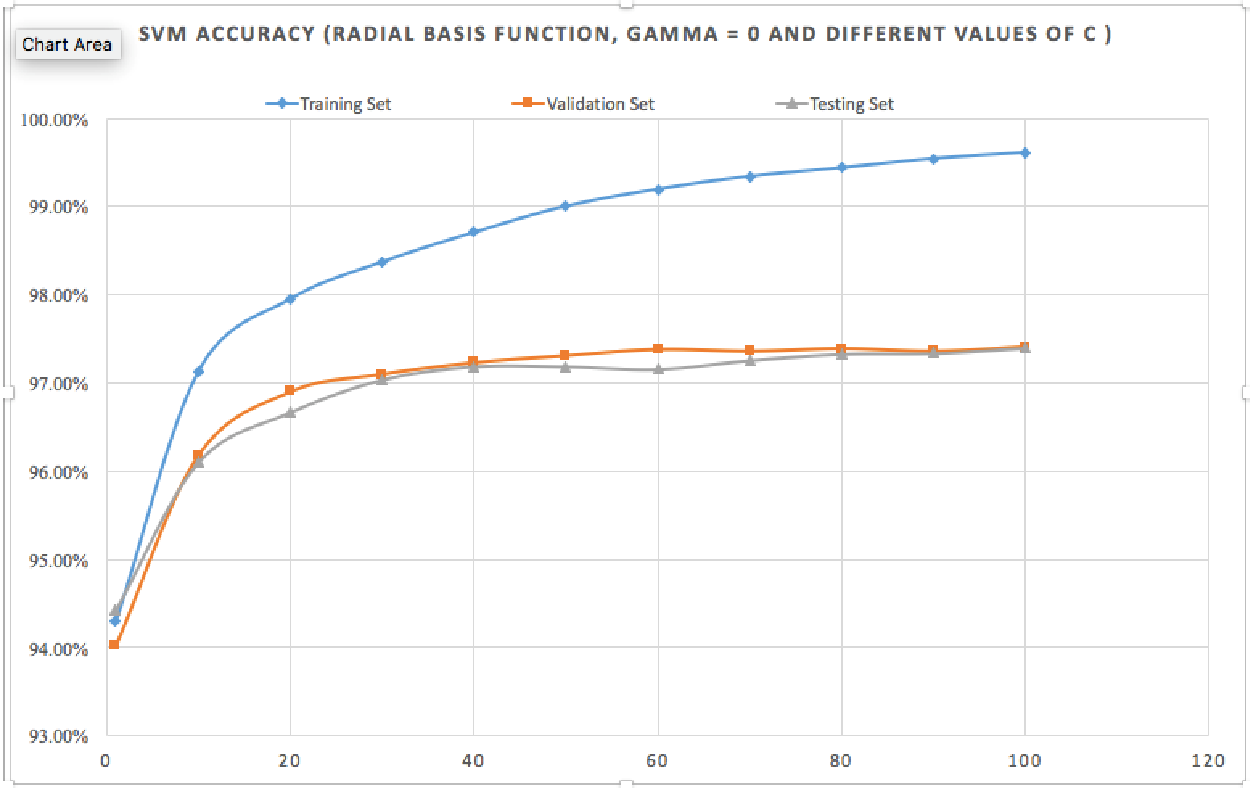
SVM Accuracy (radial basis function, gamma = 0)		
Training Set	**Validation Set**	**Testing Set**
94.294%	94.02%	94.42%

RBF kernel gives us better performance in classification as it uses normal curves around the data points, and sums these so that the decision boundary can be defined by a type of topology condition such as curves instead of a straight line. Hence in general, RBF or any other kernel (polynomial, sigmoid) gives better accuracy than linear kernel. Hyper- parameter gamma is set to default zero. Gamma acts as a regularization parameter and it controls the effect of each training example on the learned hyper-plane.

SVM Accuracy (radial basis function, gamma = 1)		
Training Set	**Validation Set**	**Testing Set**
100.0%	15.48%	17.14%

RBF kernel can be tuned using hyper-parameter, when set to 1 performs worse and is an example of over-fitting. It can be predicted because the training accuracy is 100% but the accuracy on validation and test set is very less as compared to when gamma is set to 0, or is close to zero.

SVM Accuracy (radial basis function, gamma = 0 and different values of C)			
C	**Training Set**	**Validation Set**	**Testing Set**
1	94.294%	94.02%	94.42%
10	97.132%	96.18%	96.1%
20	97.952%	96.9%	96.67%
30	98.372%	97.1%	97.04%
40	98.706%	97.23%	97.19%
50	99.002%	97.31%	97.19%
60	99.196%	97.38%	97.16%
70	99.34%	97.36%	97.26%
80	99.438%	97.39%	97.33%
90	99.54%	97.36%	97.34%
100	99.612%	97.41%	97.4%



As we increase the C value from 0 to 100, we see an improvement in the training accuracy. We can say that C is responsible for addition of penalty for every misclassified data and hence it reduces the error in the test phase. When the value of C is low, smaller amount of error is acceptable during the training phase of classification which in-turn creates a higher margin for classification and more examples can be misclassified. But when we increase the value of C, the margin is lowered and lesser examples are misclassified during the training phase. Hence the accuracy of the test set increases when C is increased but decreases when the C value is too high.

We can observe an example of over-fitting when we increase the complexity of hyper-planes when the value of C is too high (seen during C takes value between 40 and 80).