CSE-574 Introduction to Machine Learning

Assignment #3

Classification and Regression Using Logistic Regression and Support Vector Machine

Group #41
Pruthvi Mulagala – 50208595
Madhav Jakkampudi - -50206563
Vinod Kumar Veparala – 50208038

PROBLEM 1

Logistic Regression

Logistic regression is a classification model which uses of a probabilistic function to estimate the likelihood of a given input to a particular class. Here in our experiment we have trained a handwritten digit classification using binominal and multinomial logistic regression and obtained the following accuracies.

Binary Logistic Regression

In the case of BLR, we train the input sample data by using 10 binary classifier one for each class and classify if the sample either belongs to the class or not by specifying values of either 0 or 1. By doing so, below are the list of accuracies we attained on the training, validation and the test data sets.

Training set Accuracy	84.868%
Validation set Accuracy	83.68%
Testing set Accuracy	84.17%

Here we notice that the accuracies aren't that encouraging on all the three data sets. This can be due to the fact that the number of input features are quite a hand full and BLR works best with lesser number of such features. Also LR works better well when the input data set doesn't give direct information of the output. Since the features in our in our input data set are pixels, they directly determine the output and hence the lesser accuracies. It worth noting here that there is no issue of over-fitting however.

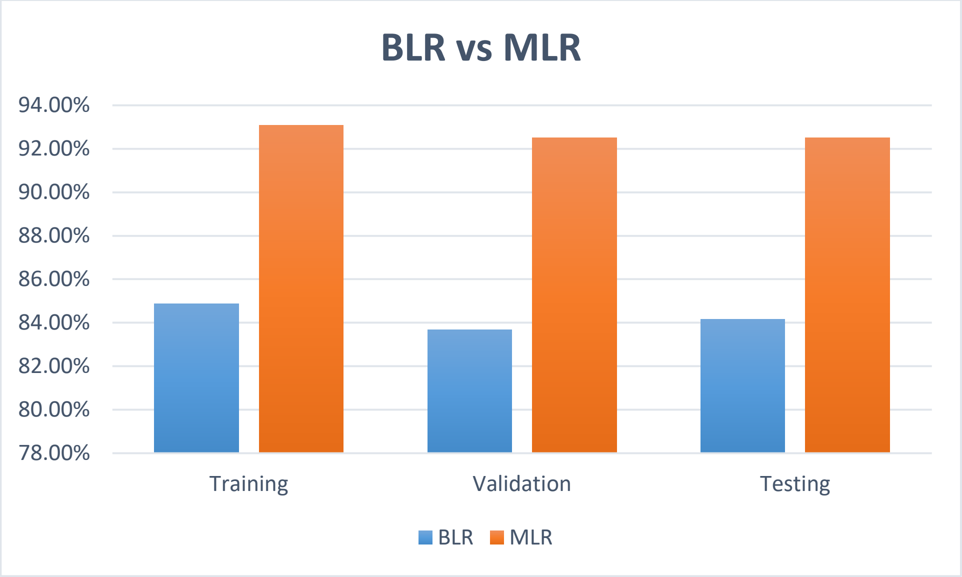
Direct Multi-class Logistic Regression

Unlike BLE, here in MLR we train the input sample data by using just one classifier which can classify the input to one of the 10 classes. Below are the list of accuracies we attained on the training, validation and the test data sets by following this regression.

Training set Accuracy	93.09%
Validation set Accuracy	92.5%
Testing set Accuracy	92.52%

By observing the accuracies, we notice an increase in the percentage accuracies of all the training, validation and the testing data and also the amount of time taken for training through MLR is much faster when compared to BLR. The increase in the accuracies even though this regression is still influenced by all the afore-mentioned reasons which are impacting the performance of BLR is explained in the next section. On a side note it is worth mentioning there is no issue of over fitting on this regression.

Comparison of BLR and MLR



In case of BLR, we employed a one-vs-all strategy which basically picks up the samples belonging to the class at hand as positive and all the other samples belonging to other classes as negative. This way we had to build a binary classifier for each class in the training data set and train them individually. Not only this approach is time consuming as it involves training each individual classifier but it is also inefficient as it highly relies on the distribution of the samples across the classes in the training data set. Even while training a single binary classifier the ratio of positive to negative samples decrease with an increase in the number of classes leading to inconsistencies in the accuracies by varying the training data set. On the other hand for MLR, there are no such issues because we compute the probability of a sample belonging to a particular class for all the classes at a single go with the usage of soft-max function and pick the class with the maximum probability. This way we not only train the data faster but also achieve better accuracies. However, even though MLR provides better accuracies as compared to BLR, it is important to notice the fact that MLR is computationally more expensive when compared to BLR and is less preferred for complex data sets.

Conclusion:

We can observe that using multi-class Logistic Regression provides better testing, validation and training accuracies as compared to the binomial Logistic Regression where we label the class to be either True/False

PROBLEM 2

SUPPORT VECTOR MACHINES

In SVM, in contrast to Logistic regression where a separating hyper-plane is chosen by considering all the points of the dataset, here we chose only the points which are close to the margin (support vectors) and try to provide the maximum possible distance between the points and the hyper-plane. Below are the list of accuracies across the training, validation and the testing data set using both Linear and Radial Basis Function.

Linear Kernel

As the name suggests, using linear SVM kernel we separate the data set using linear hyper plane. It is useful for classifying multi-dimensional data and is less prone to over-fitting issues. Below are the list of training, validation and testing accuracies we attained for the given data set.

Training set Accuracy	97.286%
Validation set Accuracy	93.64%
Testing set Accuracy	93.78%

Radial Basis Function

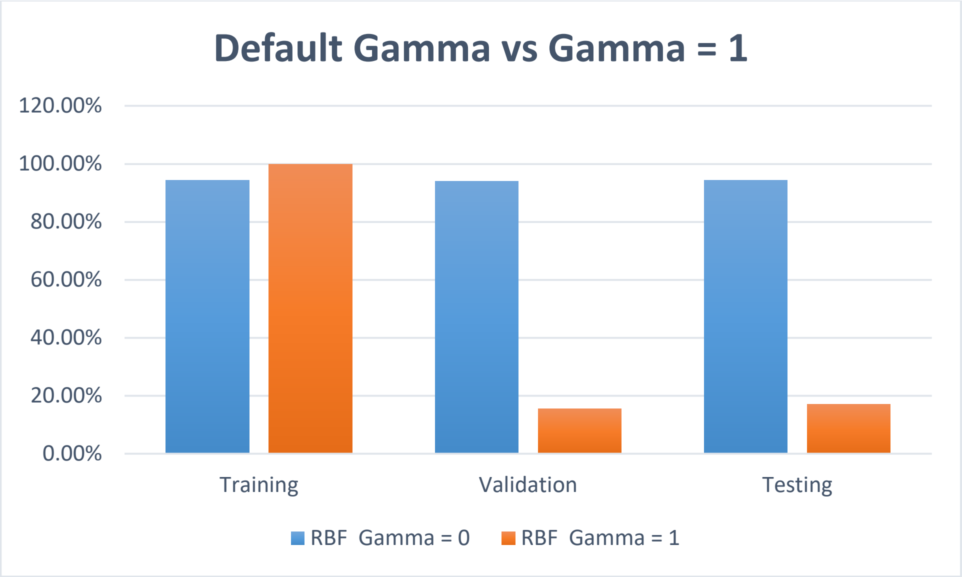
RBF is a non-linear SVM which is characterized by a couple of parameters namely Gamma and C values with Gaussian radial basis function kernel. The gamma parameter controls the influence of each training example on the learned hyper plane whereas the C parameter governs the impact of error on the training examples which in turn is used to control the complexity of the learned hyper-plane. Below are the list of training, validation, testing accuracies for a various range of gamma and C values.

With Gamma = 1

Training set Accuracy	100.0%
Validation set Accuracy	15.48%
Testing set Accuracy	17.14%

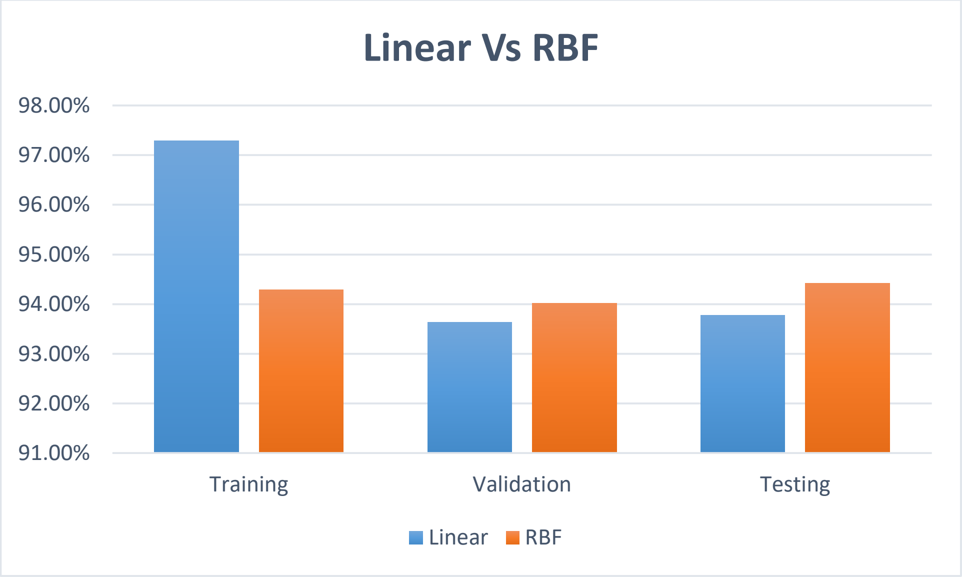
With default Gamma (default gamma = 1/number of features – 716 in our case)

Training set Accuracy	94.294%
Validation set Accuracy	94.02%
Testing set Accuracy	94.42%



Comparing the above accuracies we obtained for SVM using RBF with default gamma and gamma =1, a clear scenario of **over-fitting** is noticed for higher values of gamma(1) and for low values of gamma the results are much more encouraging. This is because gamma controls the influence of each training example on the learned hyper plane.

Comparison of Linear Vs RBF SVM Kernels

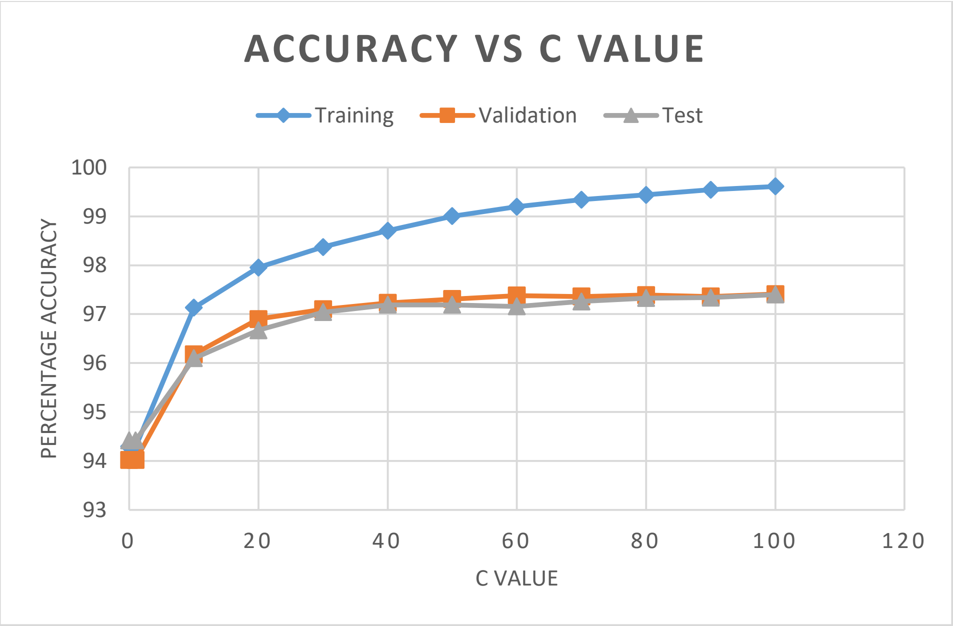


From the above accuracies we notice that even though the training accuracies are better for linear kernels, the validation and testing accuracies are better for RBF kernels with default gamma. This can be attributed to the fact that the boundaries obtained by RBF classify the data sets better when compared to linear boundaries. However it is worth noticing here that this scenario is noticed for lower values of Gamma till the point where over-fitting doesn't creep in.

With default Gamma varying ‘C’ values from 1, 10, 20 … 100

As already mentioned above, C value determines the penalty of the error term on each training sample. With lower values of C, we have lower weight values for each term and therefore a larger error value is accepted during the training phase. This implies higher margin hyper-plane even though it means misclassifying some of the samples and hence we notice lesser accuracies. Similarly with an increase in the value of C, the weight values increase for each term and therefore a smaller error value is accepted during the training phase. This implies lower margin hyper-plane with lesser misclassification of the samples. Below given are the list of accuracies of training, validation and testing data with the variation of C values.

Values	Accuracy in %		
C	Training	Validation	Test
0	94.294	94.02	94.42
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	99.542	97.36	97.34
100	99.612	97.41	97.4



From the above variation of accuracies with the increase in the value of C, we can observe that for the training data the accuracy increases steeply till C = 10 and the steepness decrease thereafter with a minimal increase in the accuracy with the increment in C values. As for the validation and the testing data, the accuracies increase till a C value of 40 similar to that of the training data but thereafter the accuracy decreases a bit and almost remain constant for all the next C values. This can be attributed to the over-fitting issue which have crept in with an increase in C value for RBF kernels after a value of 40.

Conclusion:

Logistic regression is preferred over SVM when the training data has fewer features/dimensions and in problems where the predictors doesn’t determine the response.

Also, unlike Logistic regression where all the samples in the data set determine the decision boundary, in SVM only the points near the decision boundary (support vectors) influence it. Since the data we worked on is our experiment is mnist data with a significant input features, we noticed better performances for SVM over Logistic regression.