CLASSIFICATION AND REGRESSION

CSE 574 : INTRODUCTION TO MACHINE LEARNING

Programming Assignment 3

GROUP 29

Prashanth Seralathan(pseralat)

Shreya Ravi Kumar(sr264)

Shreya Ravi Hegde(shegde3)

INTRODUCTION

In this programming assignment we implement Logistic Regression using binary logistic regression classifier and multi-class logistic regression classifier to classify the handwritten digits. We perform the same classification with SVM classifier provided by SK learn library, The SVM kernel is implemented with varying parameters of kernel, gamma values, C values. We analyze the performance of SVM, Logistic Regression with Binary Classifier and Logistic Regression with Multi-class Classifier in terms of accuracy of the classification.

OBSERVATIONS AND RESULTS

LOGISTIC REGRESSION WITH BINARY CLASSIFIER vs LOGISTIC REGRESSION WITH MULTI-CLASS CLASSIFIER

Logistic Regression generally works on the concept of discriminative classification. Discriminative classification requires less training data and the VC dimension is smaller. The basis of Logistic regression is about applying nonlinear transformation to the input data with respect to weights.

The accuracy values obtained through Logistic Regression is listed as follows,

Classification	Training Accuracy	Validation Accuracy	Test Accuracy
Logistic Regression - Binary Classifier	86.096%	85.26%	85.19%
Logistic Regression - Multi-Class Classifier	93.39%	92.43%	92.67%

Binary Classifier:

Applying Nonlinear transformation:

$$p(\mathbf{y}|\mathbf{w}) = \prod_{n=1}^{N} \theta_n^{y_n}(1-\theta_n)^{1-y_n}$$

$$\theta_n = \sigma(\mathbf{w}^T\mathbf{x}_n)$$

Log-Likelihood Error Calculation:

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

Gradient Error function:

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

Multiclass Logistic Regression is a logistic regression model that can be used to predict multiple values. In this model, we now only need to build 1 classifier to classify 'k' classes at the same time unlike a logistic regression model with a binary classifier. For the multi-class classifier, the posterior probabilities are given by a **softmax transformation** of linear functions of the feature variables.

SoftMax:

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

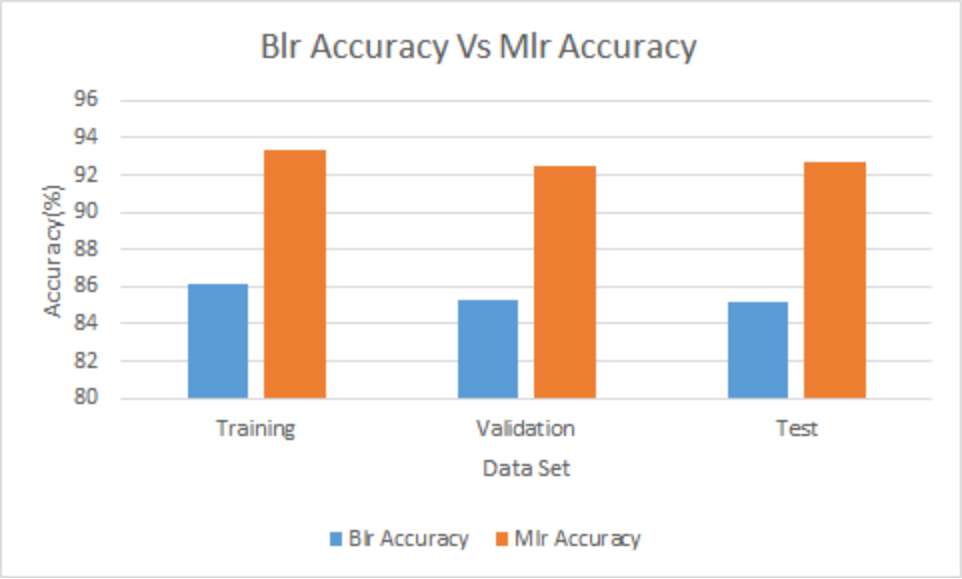
Error Function:

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

Gradient of Error Function:

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

The accuracy values improved when Multi-Class classifier is used. The Accuracy comparison plot can be seen as follows,



The reason for Logistic Regression with Multi-Class classifier performing better is because of the presence of soft-max function which enables in better prediction when dealing with multiple-classes in the input data. In our example of Handwritten digits classification we have 10 classes, i.e 0-9.

One vs Rest in Binary Logistic Classifier works based on the concept that at one particular instance, one of the classes is taken as positive and all the other classes are taken as negative. This step is repeated for all the classes to derive with boundaries that are equal to number of classes, the boundaries formed can misclassify the data as it can't recognize the data that falls in the intersection area of the classifying boundaries. **One vs All** classifier takes all classes into consideration when deciding the boundary, in-effect we would be having a classifier hyperplane that draws a boundary that separates all the classes and thus it provides better accuracy when compared to One vs Rest classifier.

Hence considering the large dataset with lots of features in it, it's better to use Multi-Class Logistic Regression.

SVM - SUPPORT VECTOR MACHINES

Support Vector Machine(SVM) also works based on the concept of classification of the data based on the hyperplane. But SVM has an additional constraint when choosing the hyperplane that separates multiple classes in the data, The goal of a support vector machine is to find the **optimal separating hyperplane** which **maximizes the margin** of the training data.

Finding the optimal hyper-plane would involve choosing the hyper-plane in such a way that it maximizes the separation distance between multiple classes of data, This can be termed as **Maximum-Margin classifiers**. This is extremely useful as it provide good generalizability.

Linear Kernel

Training Data Accuracy : 97.268%
Validation Data Accuracy : 93.64%
Test Data Accuracy : 93.78%

Linear kernel is useful when the data is high dimensional and features are informative. We performed SVM on the MNIST data which is a high dimensional data but the features(pixels) are less informative, hence we obtain low accuracy values on performing SVM with linear kernel and higher accuracies when performing SVM using non-linear kernel such as radial basis kernel.

Radial Basis Kernel

- Gamma = Default value
  Training Data Accuracy : 94.294%
  Validation Data Accuracy : 94.02%
  Test Data Accuracy : 94.42%

- Gamma = 1
  Training Data Accuracy : 100%
  Validation Data Accuracy : 15.48%
  Test Data Accuracy : 17.14%

Low accuracies are observed in case of gamma = 1 which shows a classic case of overfitting since the training test accuracy obtained was 100% and very low values of validation and test accuracies since large values of gamma leads to high bias and low variance.

C is a parameter for the soft margin cost function which controls the influence of each individual support vector, i.e it controls the cost of misclassification on the data. Gamma is the free parameter of the Gaussian radial basis function. Large gamma leads to high bias and low variance and low gamma leads to low bias and high variance.

The summarised values of accuracies obtained for different hyperparameters values are as follows :

Kernel	C	Training Data Accuracy	Validation Data Accuracy	Test Data Accuracy	Gamma
Linear	Default	97.286	93.64	93.78	Default
RBF	Default	100	15.48	17.14	1
RBF	Default	94.294	94.02	94.42	Default
RBF	1	94.294	94.02	94.42	Default
RBF	10	97.132	96.18	96.1	Default
RBF	20	97.952	96.9	96.67	Default
RBF	30	98.372	97.1	97.04	Default
RBF	40	98.706	97.23	97.19	Default
RBF	50	99.002	97.31	97.19	Default
RBF	60	99.196	97.38	97.16	Default
RBF	70	99.34	97.36	97.26	Default
RBF	80	99.438	97.39	97.33	Default
RBF	90	99.542	97.36	97.34	Default
RBF	100	99.612	97.41	97.4	Default

Linear kernel is used when the number of features is larger than number of observations and Gaussian kernel when number of observations is larger than number of features. In general the real world distribution of data tends to be Gaussian and hence Gaussian kernel results(RBF) are better than linear kernel SVM for the given MNIST data.

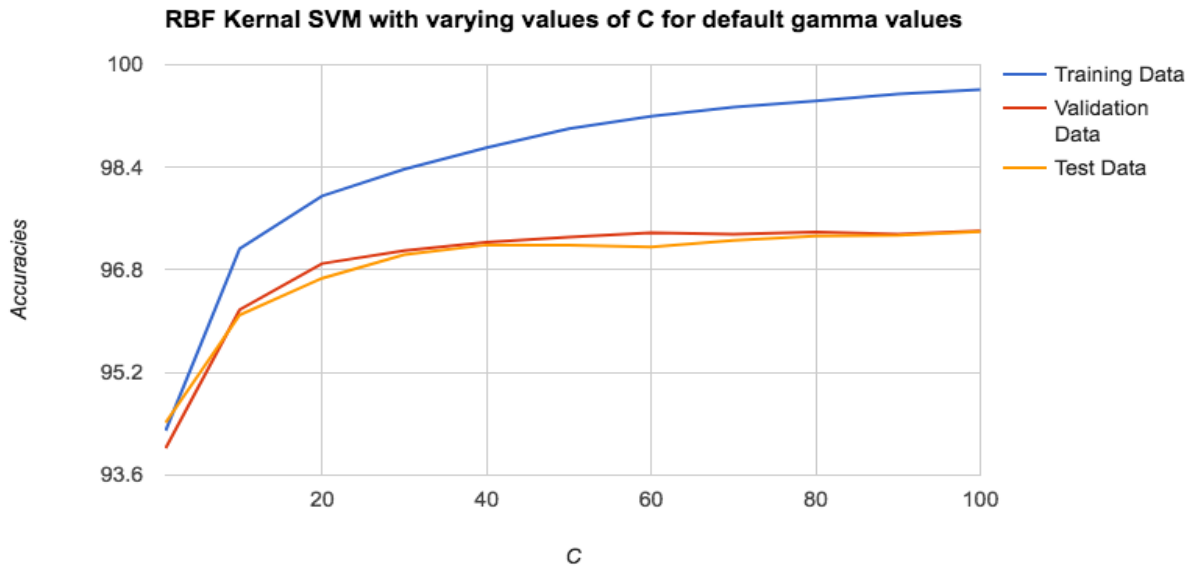


Fig . Plot of the accuracy values for different values of C from 1 to 100 in increments of 10

We can see from the above plot that increasing the value of C is increasing the accuracy values. If we choose too high a value of C, it will have high penalty for non-separable points and we may overfit the support vectors and choosing too values can result in underfitting.

CONCLUSION

In this programming assignment we performed Binary Logistic regression, Multi-class Logistic regression and Support Vector machines(SVM) using Linear and Radial basis kernels for different values of hyperparameters. SVM gave the best accuracy in comparison with logistic regression, this provides an example that Maximum Margin Classifier, i.e., SVM provides better performance in classification.

References :

[1]https://www.quora.com/What-are-C-and-gamma-with-regards-to-a-support-vector-machine
[2]http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html
[3]https://en.wikipedia.org/wiki/Multiclass_classification#One-vs.-rest
[4]https://en.wikipedia.org/wiki/Binary_classification