University at Buffalo
The State University of New York

CSE 574 Introduction to Machine Learning

Programming Assignment 3

Classification and Regression

Submitted by:

Debika Dutt: **50170009**

Mahesh Venkataramaiah: **50170332**

Vardhana Srinivas Kulkarni: **50169374**

Group# 7

The goal of this assignment is to classify the handwritten digits using Logistic regression and Support Vector Machine tool. The dataset used is the MNIST dataset and the data is divided into training set, validation set and testing set.

1. Logistic Regression:

We use a binary logistic regression to classify the digits into one of the ten possible digits. Since binary logistic regression can take two classes and in this problem and there are ten classes, we calculate using ten binary classifiers and then the classes can be distinguished from one other based on the output of binary classifiers. Gradient descent is used to minimize the error. The following equations are used to calculate the error and the gradient of the error function

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

The above equation gives the total error

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

2. SVM:

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

The operation of the SVM algorithm is based on finding the hyperplane that gives the largest minimum distance to the training examples. Twice, this distance receives the important name of margin within SVM's theory. Therefore, the optimal separating hyperplane maximizes the margin of the training data

Optimization Formulation

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \frac{\|\mathbf{w}\|^2}{2} \\ \text{subject to} \quad & y_n(\mathbf{w}^\top \mathbf{x}_n + b) \geq 1, \; n = 1, \ldots, N. \end{aligned}$$

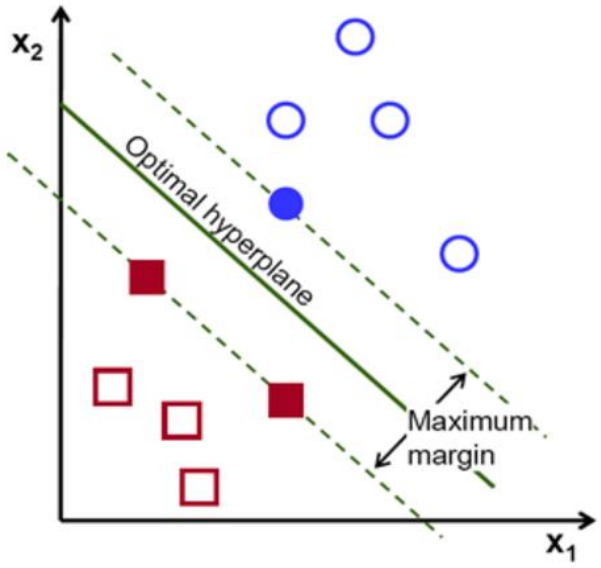

Fig.1 SVM Optimal Hyperplane

The above figure shows that it finds the optimal hyperplane gives the minimum distance between the training points or support vectors and finds the margin parameter which will try to maximize the space between two classes of training data points.

3. Direct Multi-class Logistic Regression

In this method, Logistic regression is extended to solve the multi-class classification problem. The posterior probability is given by a softmax transformation as shown in the previous equation.

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

The error function and the gradient of the error function with respect to parameters in W are as follows.

Error Function:

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

Gradient of Error Function with respect to W:

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

4. Results:

1. **Binary Logistic Regression**

   Training set Accuracy: 86.336%

   Validation set Accuracy: 85.52%

   Testing set Accuracy: 85.63%

2. **Direct Multi-class Logistic Regression**

   Training set Accuracy:93.39%

   Validation set Accuracy:92.43%

   Testing set Accuracy:92.67%

3. **SVM Accuracy**

   - **SVM with linear kernel**

   Training set Accuracy:97.286%

   Validation set Accuracy:93.64%

   Testing set Accuracy:93.78%

   - **SVM with radial basis function, gamma = 1**

   Training set Accuracy:100.0%

   Validation set Accuracy:15.48%

   Testing set Accuracy:17.14%

   - **SVM with radial basis function, gamma = 0 (default value)**

   Training set Accuracy:94.294%

   Validation set Accuracy:94.02%

   Testing set Accuracy:94.42%

SVM gives high accuracy when using a Gaussian kernel which has the flexibility to transform the data into a space of any dimension. Linear kernels are more preferred when number of features is larger than the observations while Gaussian kernels are used when number of observations is larger than number of features.

Gamma controls the influence of each training example on the learned hyperplane. When Gamma has high values, i.e Gamma = 1, despite getting an accuracy of 100% on the Training dataset, we have very low accuracies for both Validation and Test dataset.
Again, we observe that when Gamma = 0, the results improve on both Validation and Test set.

C	Training Accuracy	Validation Accuracy	Testing Accuracy
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

Table 1. Accuracy of SVM using Gaussian kernel for different C values

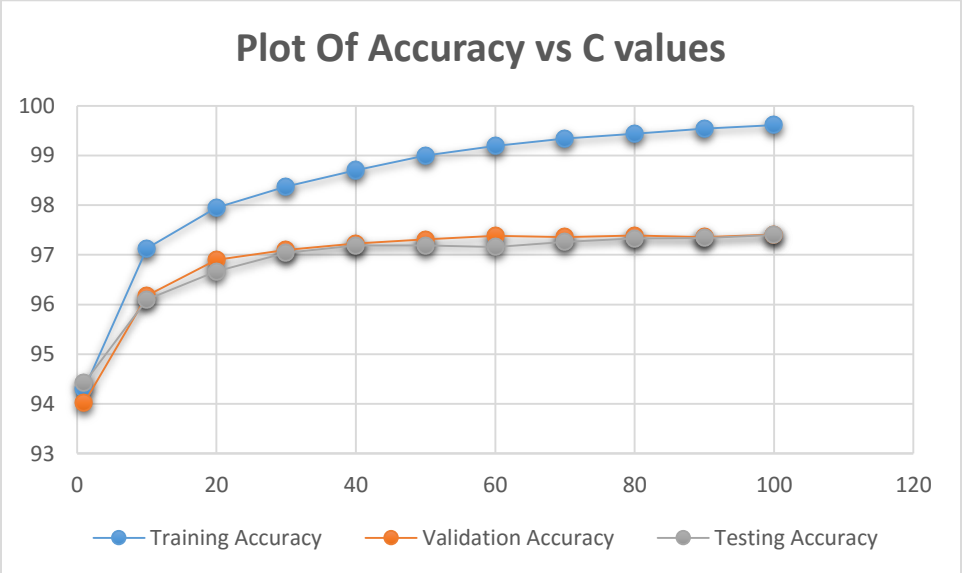

Fig.2 Accuracy of SVM using Gaussian kernel for different C values

From the above plot of accuracy of SVM for different values of C, using Gaussian kernel, we can say that the accuracy is higher for higher values of C. This is because C controls the impact of margin and the margin error.
When C has low values, the weight of each error term is low as well and a larger margin hyperplane is created.
When C has higher values, the weight of each error tem increases as well, and a smaller margin hyperplane is created and less samples are misclassified. So the accuracy of data increases.
Hence, we can infer that we should be carefully choose the value of C, also called the penalty factor. If C is large, we may have a high penalty for non-separable points and store many support vectors that may result in overfitting. It the value of C is small, we may under-fit.

5. Reference

http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html