CSE 574 - Introduction to Machine Learning

May 3, 2017

Programming Assignment 3: Classification and Regression

Group 64

Debanjan Paul - 50208716

Jay Bakshi - 50206954

Amaan Akhtarali Modak – 50206525

Introduction

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.

Datasets - MNIST

Tasks -

1. Implement Logistic Regression and infer from the prediction results.
2. Use the Support Vector Machine toolbox at sklearn.svm.SVM to perform classification.
3. Implement the gradient descent minimization of Multi-class Logistic Regression using softmax function.

Logistic Regression

Logistic regression is a probabilistic statistical classification model. It is classified into two types namely binomial and multinomial. It includes all the points within a dataset to find the separating hyperplane. It works well when there are lesser number of input features. The output in case of binary logistic regression can take only two types, either True or False whereas Multinomial logistic regression the output can have more than two types.

We're applying the logistic regression on the MNIST data which is images of handwritten digits, so that the algorithm can correctly classify them into corresponding labels.

The number of classes here are the 10-digits. To be able to use Binary classifier on this data, we will employ One-vs-All strategy with this implementation of BLR - Binary Logistic Regression.

Binary Logistic Regression (BLR)

The following equation is used to calculate 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 following equation is used to calculate the gradient of the error function:

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

Multinomial Logistic Regression (MLR)

The following equation is used to calculate the error function:

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

The following equation is used to calculate the gradient of the error function:

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

Results:

BINOMIAL LOGISTIC REGRESSION

Training data accuracy: 84.908%

Validation data accuracy: 83.74%

Testing data accuracy: 84.2%

MULTINOMIAL LOGISTIC REGRESSION

Training data accuracy: 93.284%

Validation data accuracy: 92.58%

Testing data accuracy: 92.55%

Support Vector Machine

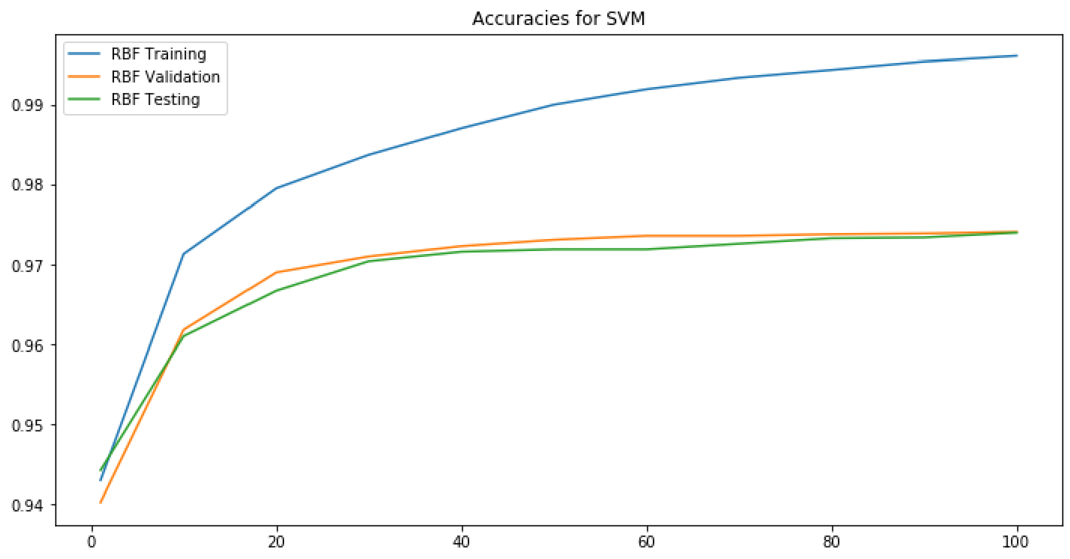
A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

The accuracy is obtained for training, testing and validation data by using various parameters listed as follows:

1. Using linear kernel.
2. Using radial basis function with value of gamma setting to 1.
3. Using radial basis function with value of gamma setting to default.
4. Using radial basis function with value of gamma setting to default and varying value of C.

The gamma parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. So Gamma value 1 means, every single training example is important and it’s essential to classify every single example correctly.

Not only it takes a lot of time to classify all the training examples, the impact of noise present in the training data is also high. This results in the poor performance in validation and test data. This is a classic over fitting example. The SVM basically memorizes all the points.



The accuracy increases with increase in the value of C. C indicates the tolerance in misclassifying data. A smaller-margin hyperplane is obtained for larger values of C. A larger-margin hyperplane is obtained for smaller values of C.

C value specifies the tolerance level in misclassifying the training data. So smaller value of C causes SVM to look for a larger margin separating hyper plane even if it means misclassifying some training points. A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly by giving the model freedom to select more samples as support vectors.

Higher C value indicates smaller margin separating hyper plane. In higher C values, the training error will be mostly zero.

Results:

LINEAR BASIS FUNCTION

Training set Accuracy:97.286%

Validation set Accuracy:93.64%

Testing set Accuracy:93.78%

RADIAL BASIS FUNCTION (GAMMA = 1)

Training data accuracy: 100.00%

Validation data accuracy: 15.48%

Testing data accuracy: 17.14%

RADIAL BASIS FUNCTION (GAMMA = 0)

Training data accuracy: 94.294%

Validation data accuracy: 94.02%

Testing data accuracy: 94.42%

RADIAL BASIS FUNCTION (GAMMA = 0 AND VARYING VALUES OF C)

C	Training Accuracy	Validation Accuracy	Testing Accuracy
1	94.29%	94.02%	94.42%
10	97.13%	96.18%	96.10%
20	97.95%	96.90%	96.67%
30	98.37%	97.10%	97.04%
40	98.71%	97.23%	97.19%
50	99.00%	97.31%	97.19%
60	99.20%	97.38%	97.16%
70	99.34%	97.36%	97.26%
80	99.44%	97.39%	97.33%
90	99.54%	97.36%	97.34%
100	99.61%	97.41%	97.40%

Comparison Between Logistic Regression and SVM

It can be seen from the given results, that SVM performs much better than Logistic Regression for all cases, except for when Gamma value is set to 1 in the Radial Basis Function (RBF). This is because when Gamma=1, there is overfitting in SVM, where the accuracy values for testing and validation are extremely poor. But, generally, SVM works better than Logistic Regression in most cases.

The main reason for this could be that the number of features in our data set is high. SVM works much better on data sets with a large number of features, whereas Logistic Regression works better when the number of features is low.

Another reason is that Logistic Regression learns any compatible hyperplane that separates the data, but SVM learns the best one, which is the one with the maximum margin.

Conclusion

Logistic regression is great when the training data has fewer features ie. lesser dimensions. SVMs do better when there's a higher number of dimensions, and especially on problems where the predictors determine the responses. In SVM only points near the boundary influence the decision boundary but in logistic regression, all the points influence it. In this MNIST data set, SVM is the best choice because the data is not linearly separable has low noise and higher dimension of data.