CSE 574
INTRODUCTION TO MACHINE LEARNING

PROGRAMMING ASSIGNMENT 3

LOGISTIC REGRESSION
&
SUPPORT VECTOR MACHINE

ALEKYA KUMAR
(50249052)
TRUPTI JADHAV
(50249177)

The project is aimed at classifying Handwritten Digit images into correct corresponding labels using two methods – Logistic Regression and Support Vector Machines.

The data set has 60000 record and 784 features. The train set is split into train set of dimension 50000*784 and validation set of dimension 10000*784. There is also a test set that has 10000 records with 784 features.

Each digit image is represented by it's pixels with values ranging from 0 to 255. White represents a value of 0, whereas Black represents 255. All values in between 0 and 255 are represented by Gray.

Preprocessing was done on the data set by taking the Variance or standard deviation and setting it to a particular threshold. This step was done to eliminate features that are repeating and that give no meaning with their existence. This is called Feature Selection.

Firstly, we implement the classification methodology using Logistic Regression.

LOGISTIC REGRESSION :

Binary Logistic Classifier :

It is a classification method that starts with making Binary classification. It uses the same principle of regression models, except that the prediction that the classifier makes is a probability measure representing the class. This is accomplished by the use of a Sigmoid Function.

There are 10 digits in the data set which are the class labels. The labels associated with each digit can take one of 10 values, i.e; multiple classes, we can't use Binary Logistic Classifier. Instead, we go for a methodology called One Vs Rest – where there is a classifier for each class.

The Binary Logistic Classifier was implemented using the cross-entropy error function and the gradient of the error function. The error and the error gradient that we obtain, are fed into the minimizer and the new Weights or parameters are obtained.

Error Function:

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

Error Gradient Function:

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

Here we use Bayesian Logistic Classifier where, for the new set of weights, the posterior probability is calculated for each class label. The argument with the highest value corresponds to the probability of that value being the target.

The accuracies were calculated for the training, validation and test sets.

```
Training set Accuracy:84.902%

Validation set Accuracy:83.72%

Testing set Accuracy:84.13000000000001%
```

The error rate for each category of the target variable was calculated using the Confusion Matrix.

The error rate is displayed below, according to the categories.

Train Set Error : Confusion Matrix and Classification Metrics

```
 Training set Accuracy:84.902%
[[4827    1   15    9   11   20   27    6    2    5]
 [   2 5651   32   13    3   20    4   11    0    6]
 [  32   41 4592   70   52   27   56   70    1   17]
 [  19   24  131 4666    8  147   21   47    1   67]
 [   9   19   23    6 4567   14   25   14    0  165]
 [  49   19   38  129   45 3966   95   18    6   56]
 [  23   12   33    2   21   69 4746    4    4    4]
 [   9   20   49   11   42   14    4 4974    0  142]
 [ 120  295  893  982  194 1305  132   52   38  840]
 [  23   24   16   89  166   47    1  157    2 4424]]
             precision    recall  f1-score   support

        0.0       0.94      0.98      0.96      4923
        1.0       0.93      0.98      0.95      5742
        2.0       0.79      0.93      0.85      4958
        3.0       0.78      0.91      0.84      5131
        4.0       0.89      0.94      0.92      4842
        5.0       0.70      0.90      0.79      4421
        6.0       0.93      0.97      0.95      4918
        7.0       0.93      0.94      0.94      5265
        8.0       0.70      0.01      0.02      4851
        9.0       0.77      0.89      0.83      4949

avg / total       0.84      0.85      0.81     50000
```

Error rates :

0 – 1.95%
1 – 1.59%
2 – 7.3%
3 – 9.01%
4 – 5.6%
5 – 10.2%
6 – 3.5%
7 – 6.03%
8 – 99%
9 – 10.6%

Test Set Error : Confusion Matrix and Classification Metrics

```
 Testing set Accuracy:84.13000000000001%
[[ 959    2    2    6    0    0    8    3    0    0]
 [   0 1122    6    2    0    0    3    0    2    0]
 [  95  154  661   25   19    0   31   30   17    0]
 [  54   48   18  842    0    0    8   30    4    6]
 [  23   66    2    1  793    0   36   12    2   47]
 [ 292  125    2  220   21  114   34   52   11   21]
 [  99   40    9    1    5    2  802    0    0    0]
 [  26   70   19    1    8    0    1  891    2   10]
 [  98  228   15  149   16    1   26   51  368   22]
 [  48   55    7   16   99    0    3  180    1  600]]
             precision    recall  f1-score   support

        0.0       0.93      0.98      0.95       980
        1.0       0.93      0.99      0.96      1135
        2.0       0.80      0.91      0.86      1032
        3.0       0.76      0.92      0.83      1010
        4.0       0.88      0.93      0.91       982
        5.0       0.71      0.88      0.78       892
        6.0       0.89      0.95      0.92       958
        7.0       0.92      0.92      0.92      1028
        8.0       0.67      0.00      0.00       974
        9.0       0.77      0.89      0.83      1009

avg / total       0.83      0.84      0.80     10000
```

Error Rates :

0 – 2.1%
1 – 1.1%
2 – 35.9%
3 – 16.6%
4 – 20.2%
5 – 87.2%
6 – 16.2%
7 – 13.3%
8 – 62.2%
9 – 40.5%

Multi-Class Logistic Classifier :

For the multi-class logistic classifier, the posterior probability is calculated using the soft-max transformation function.

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

Again, we calculate the cross-entropy error function and the gradient of the error function.

Error Function:

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

Error Gradient Function:

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

Using the error and error gradient, the new weights are calculated and further used for predicting the class label. The measure used for predicting the class label is the posterior probability of the function, just like the Binary Logistic Classifier.

The accuracies were calculated for Training, Validation and Test sets and displayed below.

From the accuracies we can see that, the Multi-class Logistic Regression is performing better than the Binary Logistic Regression thus implying that for these type of classifications, Multi-class Logistic Regression are more suitable.

```
Training set Accuracy:93.156%

Validation set Accuracy:92.49000000000001%

Testing set Accuracy:92.51%
```

The next method used for classification is the Support Vector Machines.

SUPPORT VECTOR MACHINES:

Support Vector Machines is one of the most popular methods used for classification of data.

There are several different kernel methods for performing the classification problem.

Linear and Radial Basis Kernels were used and their accuracy was computed for Training data, Validation data and Test data.

Linear Kernel:

Linear Kernel performs well when the number of features are really large. It implements the "one-vs-all" strategy training n_class models which is similar to Logistic Regression.

The parameters were kept default for this kernel and the predictions were made. The gamma value is taken as 1/no_of_features and Cost = 1. The training set gave an accuracy of 97% whereas the Validation and Test seem to perform equally good with 94% accuracy.

It can be inferred that the default parameters along with properties of linear kernel gives good accuracy.

```
Training set Accuracy:97.286%

Validation set Accuracy:93.64%

Test set Accuracy:93.78%
```

Radial Basis kernels pertain to two important parameters – C and Gamma . C is associated with Cost. A low cost makes the decision surface smooth, where as a high cost aims at classifying all the examples correctly. Gamma determines how much significance or how much each train example affects the others. If the value of Gamma value is large, the other examples or data points must be close enough by that much units to be affected.

Choosing proper values for Cost and Gamma are very critical in determining the performance of an SVM

Radial Basis Kernel with Gamma = 1:

The Radial Basis Kernel with Gamma = 1 seems to perform really well for the training set but it is performing poorly in the validation and test set with very low accuracies. This suggests that the SVM is overfitting the training set thus yielding very less accuracy for the other two. Clearly, this setting for gamma is not valid for this data set.

```
Training set Accuracy:100.0%

Validation set Accuracy:15.479999999999999%

Test set Accuracy:17.14%
```

Radial Basis Kernel with default settings:

For this type of kernel, the default settings were taken for gamma and cost. The default value of gamma is 1/no_of_features. A value of 0.0013 was taken for gamma and the default cost value was taken as 1. For these settings, the accuracy of Training, Test and Validation are almost the same. The value

```
Training set Accuracy:94.294%

Validation set Accuracy:94.02000000000001%

Test set Accuracy:94.42%
```

Radial Basis Kernel with varying Cost values:

Since running the script for 60000 values takes a really long time, the data set has been restricted to 10000 values for different settings of cost.

From the amount of data that we could obtain, we could infer that Training, Validation and Test data perform almost similar. Also, the cost accuracy across each Cost parameter is also similar for all the three data.

Training set Accuracy:21.33%

Validation set Accuracy:19.98%

Test set Accuracy:21.13%