

COMPUTER GRAPHICS AND IMAGE PROCESSING ASSIGNMENT

By
Madhavan K K(106110049)
Harikripa O C(106110032)
S V Arun Teja(106110083)

IMAGE CLASSIFICATION

RECOGNIZING HANDWRITTEN DIGITS

The project is a classifier which identifies what digit is written on an image. The classifier uses the logisitic regression technique. It works as follows.

Grey scale Images with a resolution of 20 x 20 are converted into matrices. A large number of such images are used to train the classifier. Once training is completed, the classifier is made to classify the test data. Based on the number of matches between the classifier's output and the expected one, percentage of accuracy is calculated. This classifier achieved 94.9% accuracy.

Logistic Regression

This is a short writeup on logistic regression.

Logisitic regression is a learning algorithm used for classification problems. It's output is either 0(not part of a class) or 1(part of a class). Logisitic regression works as follows.

Assume that the data we want to classify is x . Now, x is a vector ($n \times 1$) having n features. In the current problem, x has the value of each pixel in the image. Based on these features, the data is classified. The algorithm tries to find an optimal set of parameters ($\theta[n \times 1]$), that has the best accuracy. Each feature in the data, is multiplied with the corresponding value in θ , and they are summed up. This value is given as input to the hypothesis function.

Logistic regression uses a hypothesis function $h_{\theta}(x)$. $h_{\theta}(x)$ represents the value of the function when θ is the parameter and x is the input example. The hypothesis function ranges from 0-1.

To ensure the output value lies between 0-1, the sigmoid function is used
sigmoid function : $g(z) = 1/(1+e^{-z})$

The hypothesis is calculated as follows,

hypothesis : $h_{\theta}(x) = g(\theta^T x)$

$\theta^T x$ multiplies each parameter in θ with the corresponding feature in x and sums the values.

The classifier uses it as follows,

if $h_{\theta}(x) \geq 0.5$, output 1
0 otherwise

The optimal value for θ , is found by minimizing a cost function on the training examples. Let $y(0 \text{ or } 1)$ be the actual output of the training example x .

$$\text{Cost}(h_{\theta}(x), y) = -y \cdot \log(h_{\theta}(x)) - (1-y) \cdot \log(1-h_{\theta}(x))$$

$$\text{Cost}(h_{\theta}(x), y) = -\log(h_{\theta}(x)) \quad \text{if } y = 1$$

$$\text{Cost}(h\theta(x), y) = -\log(1 - h_{\theta}(x)) \quad \text{if } y = 0$$

The more our hypothesis is off from y , the larger the cost function output. If our hypothesis is equal to y , then our cost is 0.

$$\text{Cost}(h_{\theta}(x), y) = 0, \text{ if } h_{\theta}(x) = y$$

$\text{Cost}(h\theta(x), y) \rightarrow \infty$ if $y=0$ and $h_{\theta}(x) \rightarrow 1$
 $\text{Cost}(h\theta(x), y) \rightarrow \infty$ if $y=1$ and $h_{\theta}(x) \rightarrow 0$

The overall cost function, is a sum of the individual costs for each training example. This is what we have to minimize.

$J(\theta) = \sum_i (\text{Cost}(h_{\theta}(x(i)), y(i))) / m$, $i = 1$ to m where,
 m is number of training examples
 $x(i)$ is the i -th training example
 $y(i)$ is the actual output of the i -th training example

By using the technique gradient, this function is minimized, and optimum value of θ is obtained. Gradient descent works by moving down the gradient of the function till a minimum is reached.

Classifying the image

To classify the image, we train 10 classifiers, one for each digit, on the same training example set.

After finding θ for each classifier, we run all the classifiers on the test data. The classifier that returns the best output (i.e) value closest to 1, is chosen.

Datasets

The training data is a 5000x400 matrix. Each image is converted into a 20x20 matrix. This is reshaped into a 1x400 matrix. 5000 such images are used to create the 5000x400 training set. This is stored in "ex3data1.mat"

The expected output for each of these is stored in $y(5000 \times 1)$. The value of y ranges from 1 to 10 indicating what digit the corresponding image is (10 corresponds to 0). This is also a part of "ex3data1.mat".

The test data (X_{test}) is a 1000*400 matrix (i.e) 1000 images converted into 1x400 matrices. This is stored in "testInput.csv".

The test data output (y_{test}) is a 1000*1 matrix (i.e) Actual output for 1000 images. This is stored in "testOutput.csv".

The training and testing data was obtained from a Stanford open dataset. One sample image for each digit is in the folder "sample images".

How to use the program

Octave is a language similar to Matlab. It is an open source version of Matlab. Most programs that run on octave can also be run on Matlab.

Run octave in linux terminal in the directory containing the file classifier.m.

Type "classifier" (without quotes) in the terminal and press enter.

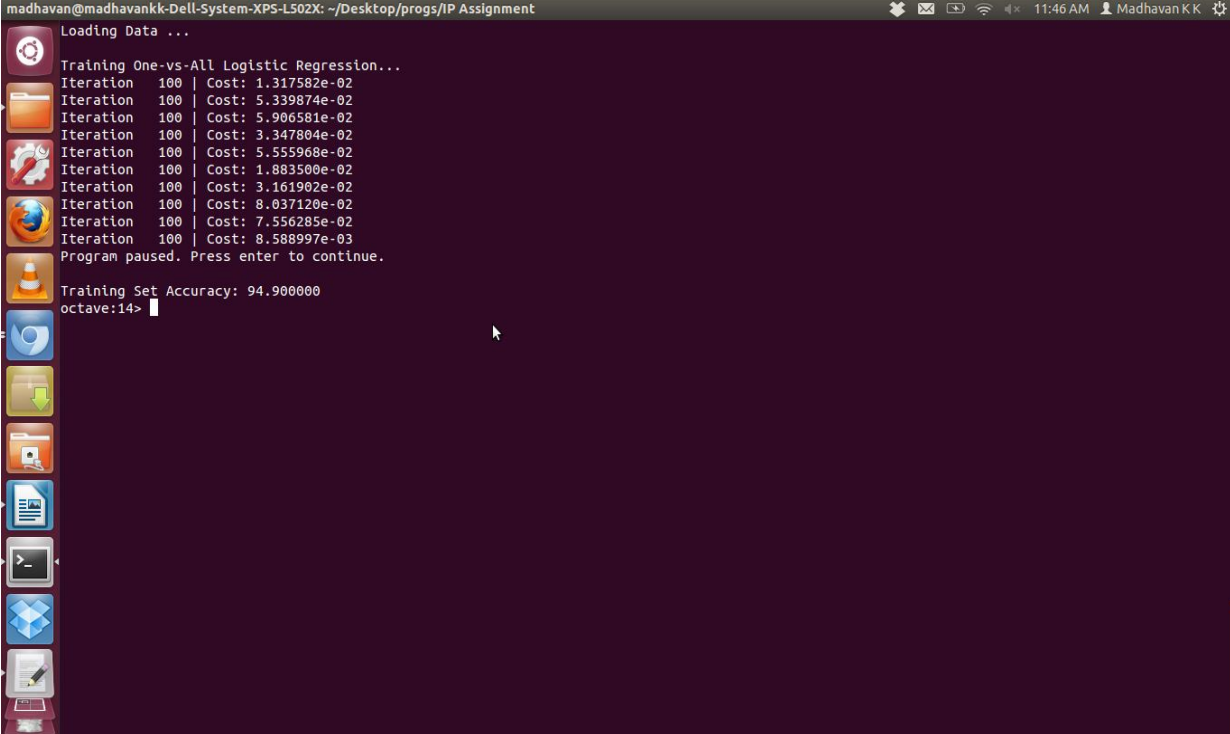
The minimum value of the cost function for each classifier will be printed on the screen.

The final output indicating the accuracy will then be printed on the screen. The predictions are stored in "predictedOutput.csv"

The number of iterations of gradient descent is limited to 100. If the program takes too long to run, this value can be changed in the file oneVsAll.m, line 19. Decreasing the value will also decrease the accuracy of the classifier.

The description of the function of each file is written in the file itself. The main controlling file is classifier.m

Sample Output



```
madhavan@madhavankk-Dell-System-XPS-L502X: ~/Desktop/progs/IP Assignment
Loading Data ...
Training One-vs-All Logistic Regression...
Iteration 100 | Cost: 1.317582e-02
Iteration 100 | Cost: 5.339874e-02
Iteration 100 | Cost: 5.906581e-02
Iteration 100 | Cost: 3.347804e-02
Iteration 100 | Cost: 5.555968e-02
Iteration 100 | Cost: 1.883500e-02
Iteration 100 | Cost: 3.161902e-02
Iteration 100 | Cost: 8.037120e-02
Iteration 100 | Cost: 7.556285e-02
Iteration 100 | Cost: 8.588997e-03
Program paused. Press enter to continue.
Training Set Accuracy: 94.900000
octave:14>
```