**DEVELOP AN OCR SYSTEM USING A LINEAR PERCEPTRON**

**DATASET**

Dataset is the same as for assignment 4. This data is a subset of OCR data taken from

http://cmp.felk.cvut.cz/cmp/software/stprtool/index.html

**View an image**

You have 4 files: train2\_5.txt, train2\_5Labels.txt, test2\_5.txt, test2\_5Labels.txt.

Each row of train and test contains the image of a digit (either 2 or 5). Each image is a 16x16 image, stored in one row as a 256 dimensional vector. You can view any images as follows:

X =read.table(‘train2\_5.txt’)

r = X[1,] #first digit image, i.e., image in row 1

r = data.matrix(r)

dim(r) = c(16,16)

image(r) #view image

Similarly you can view any image you like for any row of train and test matrices

**ALGORITHM FOR STOCHASTIC GRADIENT DESCENT**

Dimensions of the data matrix are mxn (m rows and n column)

1. Add a column of ones to data matrix X (this will cater for bias unit).

2. Initialize all weights randomly #(n+1 weights because of the column of ones)

3. Repeat each epoch as: #i is a row index and j is a column

#index

a. for i = 1 to m # for each row

{

i. compute 𝑦̂𝑖=Σ𝑥𝑖𝑗𝑤𝑗𝑛𝑗=0́ #prediction for one row

ii. update the weights as: #each weight has to be updated. In

# R it can be done without a loop.

# Just do matrix/vector math

# j=0..n (note n+1 weights)

Δ𝑤𝑗←∝Δ𝑤𝑗+ (𝑦𝑖−𝑦𝑖̂)(𝑥𝑖𝑗) #alpha = momentum,

#eata = learning rate 𝑤𝑗 ← 𝑤𝑗+ Δ𝑤𝑗

}

**MODEL BUILDING & EVALUATION**

**Training part**

Write a function for training:

trainGradientDescent <- function(X,Y,learningRate,momentum)

The above function should return the weights when given the training data X, target values y and the learning rate and momentum. Don't forget to add a column of ones to X so that you can cater for the bias unit. Alternatively, you can keep a separate variable for bias.

**Test part**

Write a test function for getting predictions

testGradientDescent <- function (testX,regressionCoefficients)

This function should also not have any loops or iterations

**Evaluation part**

Read about the confusion matrix

https://en.wikipedia.org/wiki/Confusion\_matrix

Write an evaluation function that makes a confusion matrix

Now compute the following:

BalancedAccuracyRate (BAC) = (TP/totalPositiveLabels+TN/totalNegativeLabels)/2

**Main script: Bring it all together**

Write a main script that:

a. Reads the training data and builds a perceptron model. Next it gets predictions from the model using the training set as well as the test set.

b. Find a way of mapping the OCR labels to the predictions. So for example if your prediction is 10, then how will you map it to a label?

c. For the training data as well as the test data, make the confusion matrix for different values of lambda, as given in the report.

**Simulations**

You have to repeat for different values of learning rate and momentum and the number of iterations