Assignment Report 2 - Machine Learning COL774

Piyush Kaul - 2015EEY7544

March 15, 2016

1 ANS-1 AD CLASSIFIER USING SVM

Download and install the CVX package. Express the SVM dual problem (with a linear kernel) in the a form that the CVX package can take. You will have to think about how to express the SVM dual objective in the form $\alpha^T Q\alpha + b^T\alpha + c$ matrix where Q is an $m \times m$ matrix (m being the number of training examples), b is an m-sized column vector and c is a constant. For your optimization problem, remember to use the constraints on $i's$ in the dual. Use C = 1. Report the set of support vectors obtained from your optimization.

1.1 PART A

Number of Support Vectors obtained is dependent on cutoff threshold.

1. For threshold 10^{-1} number of support vectors is 219
2. For threshold 10^{-2} number of support vectors is 329
3. For threshold 10^{-3} number of support vectors is 372
4. For threshold 10^{-4} number of support vectors is 374
5. For threshold 10^{-5} number of support vectors is 377
6. For threshold 10^{-6} number of support vectors is 368
7. For threshold 10^{-7} number of support vectors is 382

   From the above, the number of support vectors is roughly 375.

1.2 PART B

Q. Calculate the weight vector w and the intercept term b using the solution in the part above. Classify the test examples as "ad" or "nonad". Report the average accuracy obtained.

Ans. Accuracy is 11 errors out of 779 = 98.5879%

1.3 PART C

Q. Now solve the dual SVM problem using a Gaussian kernel with the bandwidth parameter $= 2.5 * 10^{-4}$. Think about how the Q matrix will be represented. What are the set of support vectors in this case? Note that you may not be able to explicitly store the weight vector (w) or the intercept term (b) in this case. Use your learned model to classify the test examples and report the accuracies obtained. How do these compare with the ones obtained with the linear SVM?

Ans. Accuracy is 1 errors out of 779 = 99.86% The number of support vectors is 460 with threshold 0.23. These support vectors are the first 460 features. The number of support vectors is 2500 with threshold smaller than 0.23.

1.4 PART D

Q. Now train an SVM on this dataset using the LibSVM library, available for download from LibSVM. Repeat the parts above using a linear Kernel as well as a Gaussian kernel with $= 2.5 * 10^{-4}$. Use C = 1 in both cases, as before. Report the set of support vectors obtained as well as the test set accuracies for both linear as well as the Gaussian kernel setting. How do these compare with the numbers obtained using the CVX package. Comment.

Ans.

1. Number of Support Vectors for Linear Kernel with libSVM is 329. Accuracy = 98.5879%
2. Number of Support Vectors for Gaussian Kernel with libSVM is 968. Accuracy = 100%
3. Accuracy is roughly same between libSVM and CVX based SVM.

1.5 PART E

Q. The first 3 attributes in the data are continuous valued, depicting the geometry of the image. Some of these values were missing (unknown) in the original dataset. These missing values were replaced with 0 in the dataset provided to you. What other criteria (other than setting all of them to 0), could be used to initialize these values? Report the accuracies obtained using linear and Gaussian kernels with $= 2.5 * 10^{-4}$ after your modifications. Use C = 1 in both cases, as before. Describe the rationale behind your modifications and comment on the results obtained. Note: Do not submit the CVX or LibSVM code. You should only submit the code that you wrote by yourself (including wrapper code, if any) to solve this problem.

Ans. The first 3 attributes can be replaced with average of these attributes across all instances. This gives slightly better results for Linear Kernel case at 98.7% detection rate

2 ANS-2 NEURAL NETWORK

Q. In this problem, you are given the MNIST handwritten digit dataset1 (mnist all.mat) that contains 60K training and 10K testing examples of handwritten digits. Each example in the dataset is represented by 784 features corresponding to (28 x 28) pixel values ([0, 255]). The classes are 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9 corresponding to each digit. Examples are partitioned based on the class to which they belong. We will implement the neural network learning algorithm (using backpropagation) to recognize the digits given the pixel values

2.1 PART A

Q. Write a script to visualize the digits in the data. Your script should take an example index and display the image corresponding to the gray scale pixel values.
Ans. Following API is created

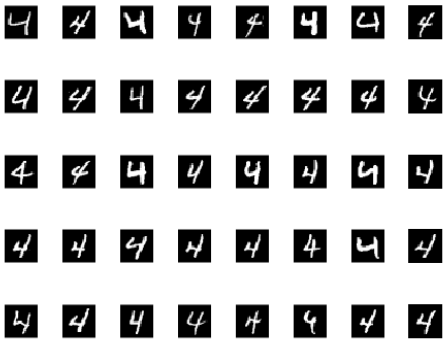
```
function show_mnist(digit, index, m, n)
```

Here
digit is number of the digit,
index is single index or range of indices
m and *n* are dimension of matrix of images in case index is range

The following commands shows the usage to display images for digit 4 for indices 1 to 40 in a 5 by 8 grid. Images displayed are shown below

```
show_mnist(4,1:40, 5,8);
```

Figure 2.1: Digit visualization script output



2.2 PART B

Q. Extract the data for classes 3 and 8 from the original mnist all.mat file to create a new mnist bin38.mat file for binary classification. Train a neural network with one hidden layer (with 100 units) using the backpropagation algorithm. You should implement the algorithm from first principles and not use any existing matlab modules. Use the stochastic gradient version of the algorithm. Use a variable learning rate given as $\alpha_t = 1/\sqrt{t}$ where t denotes the learning iteration. Choose an appropriate stopping criteria based on the change in value of the error function. Report the stopping criteria that you chose

Ans. Stopping Criteria used is that the error in any given iteration should is larger or equal to average error in previous 5 iterations i.e.

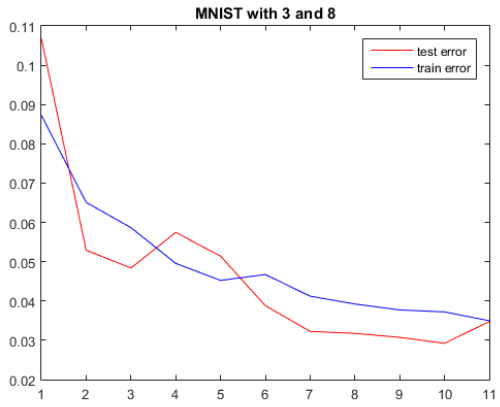
1. testerr(end) >= mean(testerr(end-5:end-1))

2.3 PART C

Q. Report your accuracies over the test set using the learned network. Also report the training times of your algorithm.
Ans. Detection rate of 91.6% was achieved.

Training time requireds is 50 seconds per iterations. Eleven iterations were required, hence simulation ran approximately 9 minutes.

Figure 2.2: Training and Test Error



2.4 PART D

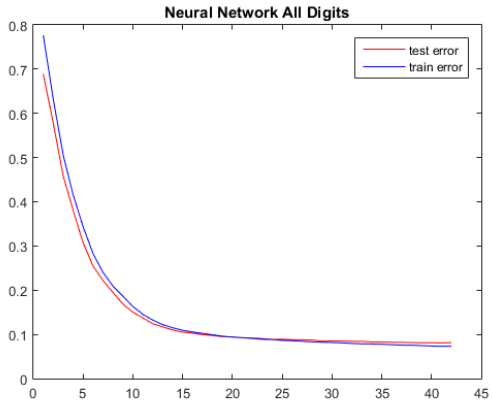
Q. Train the neural network classifier on the original multiclass MNIST dataset with the same experimental settings as in the binary case above. How many output units would you need in this case? Report the accuracies over the test set. Do you see any difference in training times

compared to the binary setting?
Ans.

1. 10 outputs units were used.

2. With 100 hidden units we get the following results. Detection rate is 807 errors for 10000 images. i.e. 91.93% detection rate. Time taken is 42 iterations with 100 sec per iteration i.e. 70 minutes. To speed-up the algorithm minibatch with variable size was added. For minibatch size 100, training takes 30 sec per iteration i.e. total of 21 min minutes.

Figure 2.3: Training and Test Error with 100 Neurons. All Digits 0-9



3. With 1000 hidden units we get the following results. Detection rate is 506 errors for 10000 images. i.e. 94.94%. Time taken is 30 sec per epoch after conversion to mini-batch algorithm. Since it took 81 epochs to converge this amounts to 40 minutes.

Figure 2.4: Training and Test Error with 1000 Neurons. All Digits 0-9