Introduction to Machine Learning *Mohit Khatwani, AJ75499*

HW3

Please note that only PDF submissions are accepted. We encourage using LaTeX to produce your writeups. You'll need *mydefs.sty* and *notes.sty* which can be downloaded from the course page.

Gradient descent algorithm:

1. Implement the gradient descent algorithm for binary SVM.
2. Similar to the previous homework, train and test it for classifying digits "1" and "6" in MNIST dataset.
3. Plot the accuracy on the test set wrt. the number of iterations. Here, processing each data-point is considered one iteration. You don't need to use all training data if it takes a long time to run.
   Figure 1 shows accuracy plot with respect to number of iterations for digits 1 and 6.

   (a) Play with the hyper-parameter C to see its effect in the accuracy and overfitting. For instance, try very large and very small values for it.
   For very large values of Hyper-parameter C the accuracy is decreased drastically.
   Small values of C does not have any effect on accuracy of test set. Optimal value of C can be given as 1/(number of epochs)

   (b) Choosing the learning rate may be a little tricky. One popular strategy is to reduce it promotional to $\frac{1}{t}$ where t is the iteration number.
4. Visualize the learned model in the image form to see if it makes sense. Note that the weight vector has both positive and negative values, so you may plot those on two separate planes.
   Figure 2 and Figure 3 show weight vector visualization for digits 1 and 6 respectively.
   Weight vector comes to be combination of two digits. It is separated and displayed on two different planes.
5. Sort the data before training so that all "1"s appear before "6"s and plot the accuracy wrt iterations. Is this faster or slower in training?
   Figure 4 shows accuracy plot with number of iterations for sorted data in digits 1 and 6.
   It is slower in training because the model is fully trained after more number of iterations than previously shuffled data.
6. Implement a function for 1-vs-all multi-class SVM that calls the binary SVM function.
7. Train and test it on all 10 digits in MNIST and report the confusion matrix as well as the average precision. $conf(i, j)$ is the number of images that are from category i and are classified into category j. You should normalize this so that all rows sum to one and then average accuracy is the average of its diagonal values.
   Average precision comes to be 0.9558
8. Look at top mistakes and show images along with ground truth label and the predicted label to see if they make sense.
   Figure 5 shows top mistakes for every number (0-9).
   Label for every image is presented as (i,j) where i is actual label of that image and j in predicted label.
   This output makes sense as numbers 1,2,9,8 are shown to be mis-classified as 7,7,4,3 respectively.

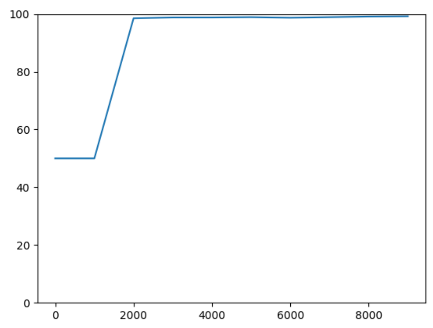

Figure 1: Accuracy plot with respect to number of iterations for 1 and 6

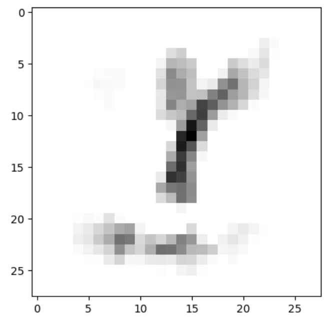

Figure 2: Weight Vector visualization for digit 1

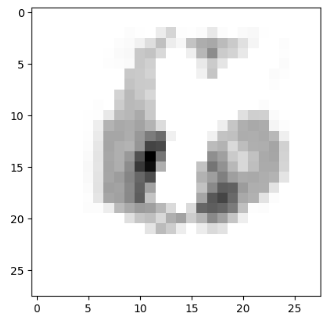

Figure 3: Weight Vector visualization for digit 6

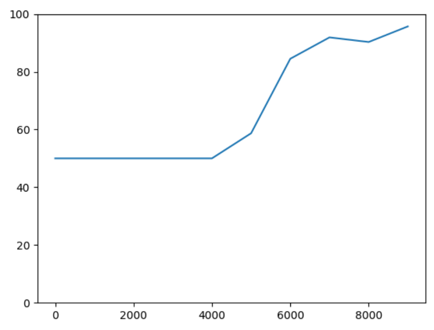

Figure 4: Accuracy plot with respect to number of iterations for 1 and 6 with sorted training data

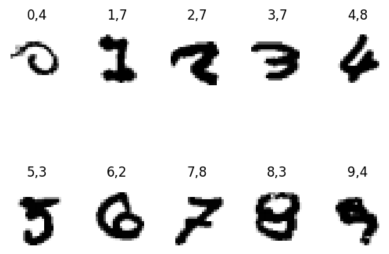

Figure 5: Worst set of images with their actual and predicted labels