Homework 2, CSCI 5525 2018

Paul Schrater

October 11, 2018

1 Support vector machine implementation using Convex Optimization

1. (a) (15 pts) Implement a linear SVM with slack variables in dual form. See the separate file optimizers.txt on how to install and call the optimizers. Write a prediction function that can be used to "predict" an input point, that is, compute the functional margin of the point. Look in the file svmtest for a simple skeleton that shows how to read files and plot results.

   (b) (15 pts) Run your SVM implementation on the MNIST-13 dataset. The dataset contains two classes labeled as 1 and 3 in the first column of the csv file we provided. All other columns are data values. Compute test performance using 10-fold cross-validation on random 80-20 splits. Report your results and explain what you find for the following manipulations.

      i. Try $C = 0.01, 0.1, 1, 10, 100$ and show your results, both graphically and by reporting the number of mistakes on the training and validation data sets.

      ii. What is the impact of test performance as C increases?

      iii. What happens to the geometric margin $1/||w||$ as C increases?

      iv. What happens to the number of support vectors as C increases?

   (c) (10 pts) Answer the following questions about the dual SVM approach:

      i. The value of C will typically change the resulting classifier and therefore also affects the accuracy on test examples.

      ii. The quadratic programming problem involves both w, b and the slack variables ξ_i. We can rewrite the optimization problem in terms of w, b alone. This is done by explicitly solving for the optimal values of the slack variables $\xi_i = \xi_i(w, b)$ as functions of w, b. Then the resulting minimization problem over w, b can be formally written as:

$$\frac{1}{2}||w||^2 + C \sum_{i=1:N} \xi_i(w, b)$$

where the first (regularization) term biases our solution towards zero in the absence of any data and the remaining terms give rise to the loss functions, one loss function per training point, encouraging correct classification. The values of these slack variables, as functions of w, b, are "loss-functions". a) Derive the functions $\xi_i(w, b)$ that determine the optimal ξ_i. The equations should take on a familiar form. b) Are all the margin constraints satisfied with these expressions for the slack variables?

1.1 Procedure

Use the code templates provided in the moodle, and any additional instructions posted by the TA.

1.2 Reporting

Code: You will have to submit code for myDualSVM(filename, C) (main file). This main file has input: (1) a filename (including extension and absolute path) containing the dataset and (2) the value of the C parameter: (1) the average test error and standard deviations printed to the terminal (stdout). The function take the inputs in this order and display the output via the terminal. The filename will correspond to a plain text file for a dataset, with each line corresponding to a data point: the first entry will be the label and the rest of the entries will be feature values of the data point. The data for the plots can be written in a file tmp in the same directory, but you do not have to explain how you did the plots from this data, and can use an iPython notebook to . You can submit additional files/functions (as needed) which will be used by the main file. Put comments in your code so that one can follow the key parts and steps in your code.

Summary of methods and results Describe your algorithm, and create plots that show the number of support vectors vs. C and test performance vs. C, with standard deviations.

2 Support vector machine implementation on Primal problem using Stochastic Gradient Descent

The goal is to evaluate the performance of optimization algorithms for SVMs which work on the primal. For both these problems, I EXPECT you to work in groups of two - please name your teammate and include for your submission. The first part of this problem is based on following paper: "Pegasos: Primal Estimated sub-GrAdient SOlver for SVM," by S. Shalev-Shawtrz, Y. Singer, and N. Srebro. Please read Section 2 of the paper. The algorithm discussed in the paper is a (stochastic) mini-batch (projected) sub-gradient descent method for the SVM non-smooth convex optimization problem. For the experiments, we will assume $b = 0$ for the affine form $f(x) = wTx + b$ (you can see related discussions in Sections 4 and 5).

The second part of the homework is an experimental attempt to use a softened hinge loss called "soft-plus" to create an analytically computable gradient. Recall that the primal problem has the form:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{N} \sum_{i=1:N} \max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i) + \lambda ||w||^2$$

We can use the softplus function $\text{softplus}_a(x) = a \log(1 + \exp(x/a))$, for which $\max(0, x) = \lim_{a \to 0} \text{softplus}_a(x)$. The softplus function has an analytic derivative, which will allow deriving a straightforward gradient descent algorithm.

For the homework, do the following:

1. (30 points) Implement the Pegasos algorithm and evaluate its performance on the MNIST-13 dataset. The evaluation will be based on the optimization performance, and not classification accuracy. So, we will use the entire dataset for training (optimization). For the runtime results, run the code 5 times and report the average runtime and standard deviation. The runs will be repeated with different choices of mini-batch size (see below).

2. (30 points) Implement the gradient descent algorithm and evaluate its performance on the MNIST-13 dataset. You will need to do two things. First derive the gradient.

   $$\nabla \mathcal{L}(\mathbf{w}) = \frac{1}{N} \sum_{i=1:N} \nabla \left[a \log(1 + \exp\left((1 - y_i \mathbf{w}^T \mathbf{x}_i)/a\right) + \lambda ||w||^2 \right]$$

   with respect to $\mathbf{w}$ using the chain rule. Second, implement your gradient descent into stochastic gradient descent code.

   Evaluation will be based on the optimization performance, and not classification accuracy. So, we will use the entire dataset for training (optimization). For the runtime results, run the code 5 times and report the average runtime and standard deviation. The runs will be repeated with different choices of epoch size (see below). You will have to submit (i) summary of methods and results report, and (ii) code for each algorithm:

   Summary of methods and results: Pegasos works with a mini-batch A_t of size k in iteration t. When $k = 1$, we get (projected) stochastic gradient descent (SGD), and when $k = n$, we get (projected) gradient descent (GD), since the entire dataset is considered in each iteration. For the runs, we will consider the following values of k: (a) k = 1, (b) k = 20 (1% of data), (c) k = 200 (10% of data), (d) k = 1000 (50%

of data), and (e) k = 2000 (100% of data). For fixed percent mini-batches, pick the corresponding percentages from each class, e.g., for 10%, pick 10% of samples from each of the two classes. For each choice of k as above, report the average runtime of Pegasos over 5 runs on the entire dataset, along with the standard deviation. For each choice of k, plot the primal objective function value for each run with increasing number of iterations. Thus, there will be a figure with 5 plots (different runs) overlayed for each choice of k, and a separate figure for each k. For your softplus code, match the minibatch structure of your Pegasos code. For the softplus runs, use values of k: (a) k = 1, (b) k = 20 (1% of data), (c) k = 200 (10% of data), (d) k = 1000 (50% of data), and (e) k = 2000 (100% of data). For fixed percent mini-batches, pick the corresponding percentages from each class, e.g., for 10%, pick 10% of samples from each of the two classes. For each choice of k as above, report the average runtime of softplus over 5 runs on the entire dataset, along with the standard deviation. For each choice of k, plot the primal objective function value for each run with increasing number of iterations. Thus, there will be a figure with 5 plots (different runs) overlayed for each choice of k, and a separate figure for each k. Thus, there will be a figure with 5 plots (different runs) overlayed for each choice of k, and a separate figure for each k.

Termination condition: Please use ktot - the total number of gradient computations - as the termination condition. For this question, set ktot = 100n, where n is your data size (number of data points). The algorithm terminates if ktot is reached.

Code: You will have to submit code for (1) myPegasos(filename, k, numruns) (main file), and (2) mySoftplus(filename, m, numruns) (main file). The main files have input: a filename (including extension and absolute path) containing the dataset, (2) mini-batch size k for Pegasos and softplus, and (3) the number of runs, and output: (1) the average runtime and standard deviations printed to the terminal (stdout). The functions must take the inputs in this order and display the output via the terminal. The filename will correspond to a plain text file for a dataset, with each line corresponding to a data point: the first entry will be the label and the rest of the entries will be feature values of the data point. The data for the plots can be written in a file tmp in the same directory, but (i) we will not be using this data for doing plots, and (ii) you do not have to explain how you did the plots from this data. You can submit additional files/functions (as needed) which will be used by the main file. Put comments in your code so that one can follow the key parts and steps in your code.

Instructions: Follow the rules strictly. If we cannot run your functions, you get 0 points. Things to submit hw2.pdf: The report that contains the solutions to Problems 1,2, and 3, including the summary of methods and results. myDualSVM: Code for Question 1. myPegasos and mySoftplus: Code for Question 2. README.txt: README file that contains your name, student ID, your partner's name and ID, your email, instructions on how to your run program, any assumptions you?re making, and any other necessary details. Any other files, except the data, which are necessary for your program.