CMU 10-715: Homework 2

Soft Support Vector Machine Theory and Implementation

DUE: Sept. 26, 2020, 11:59 PM.

Instructions:

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books, again after you have thought about the problems on your own. Please don't search for answers on the web, previous years' homeworks, etc. (please ask the TAs if you are not sure if you can use a particular reference). There are two requirements: first, cite your collaborators fully and completely (e.g., "Alice explained to me what is asked in Question 4.3"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only.

- **Submitting your work:** Assignments should be submitted as PDFs using Gradescope unless explicitly stated otherwise. Each derivation/proof should be completed on a separate page. Submissions can be handwritten, but should be labeled and clearly legible. Else, submission can be written in LaTeX. Your code will be evaluated with Gradescope Autograder. There is no limit on the number of submissions to Gradescope.

- **Late days:** For each homework you get three late days to be used only when anything urgent comes up. No points will be deducted for using these late days. We will consider an honor system where we will rely on you to use the late days appropriately.

- **Skeleton codes:** The python files, data.py and soft_svm.py can be found at `https://github.com/ShenghaoWu/10715/tree/master/hw2`

1 Soft Support Vector Machine Theory [40]

Consider the primal problem for the soft support vector machine (soft SVM). Where $y_i \in \{-1, 1\}$ are the labels, $\mathbf{x}_i \in \mathbb{R}^p$, $i = 1, \ldots, n$ are the features (features already include the bias term), $\xi_i \in \mathbb{R}^+$ are the slack variables.

$$\begin{aligned} \underset{\mathbf{w}, \xi_i}{\text{minimize}} \quad & \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{n} \xi_i & \\ \text{subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i) \geq 1 - \xi_i & i = 1, \ldots, n \\ & \xi_i \geq 0 & i = 1, \ldots, n \end{aligned} \tag{1}$$

(a) (30 points) Show that the soft SVM problem can be written as a regularized Hinge Loss problem:

$$\underset{\mathbf{w}}{\text{minimize}} \quad \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{n} \max\left(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i)\right) \tag{2}$$

(b) (10 points) Find an expression for the subgradient of the regularized Hinge Loss problem from equation (2).

2 Implementation of the Soft SVM [60 points]

Algorithm 1: Gradient Descent algorithm

Input: Number of steps T, learning rate α and batch size b
Initialize parameters $\mathbf{w}_0 = 0$, $b_0 = 0$, step $t = 0$;
for t *in* $1, \ldots, T$ **do**
  Sample a batch $\mathbf{b} = \{(x_i, y_i)\}$ with $|\text{batch}| = b$;
  Compute the subgradient $\hat{\nabla} f(w) = 1/b \sum_{i \in \mathbf{b}} \nabla f(x_i, y_i)$;
  Update the parameters $\mathbf{w}_{t+1} = \mathbf{w}_{t+1} - \alpha \hat{\nabla} f(w)$;
end

(a) (0 points) For the data provided by data.py, write a function to prepare data that concatenates a constant to the original data $\mathbf{X} = [\mathbb{1} | \mathbf{X}_{original}]$ so that the bias term is implicit in the weights $\mathbf{w}$, and change y to $\{-1, 1\}$.

(b) (5 points) Complete the code of the Soft SVM Loss method. This method includes both the Hinge Loss and the L2 regularization term.

(c) (5 points) Complete the code of the subgradient method of the Soft SVM Loss, that you derived in question (1.b).

(d) (5 points) Complete the code of the predict method.

(e) (10 points) Complete the sampling method which takes as input $\mathbf{X}$ and y, batch size, and returns a sampled batch $\mathbf{X}$.

(f) (15 points) Complete the train method of the soft SVM using the gradient descent algorithm. The train method should keep track of the train and test SVM loss trajectories (regularization + hinge), the train and test accuracy trajectories and the computational time of each step.

Train your soft SVM classifiers on the data from data.py. Report the following plots in your pdf full gradient, stochastic gradient (batch size of 1) and mini batch of batch size 128.

(g) (5 points) Plot the train and test **losses** trajectories against iterations.

(h) (5 points) Plot the train and test **losses** trajectories against time.

(i) (5 points) Plot the train and test **accuracy** trajectories against iterations.

(j) (5 points) Plot the train and test **accuracy** trajectories against time.