Huberized support vector machine

(Due in class Feb-16-2018)

1 Problem description

The original support vector machine uses the hinge loss function $\max(0, 1-t)$, which is non-differentiable and cause trouble on designing efficient solvers. One way to conquer the trouble is to smooth the loss function by the huberized hinge loss function, defined as

$$h_\delta(t) = \begin{cases} 0, & \text{if } t > 1, \\ \frac{(1-t)^2}{2\delta}, & \text{if } 1-\delta < t \le 1, \\ 1 - t - \frac{\delta}{2}, & \text{if } t \le 1-\delta \end{cases}$$

where $\delta > 0$ is a smoothing parameter. With the smooth loss function, one can find the decision boundary by solving the huberized support vector machine (HSVM):

$$\min_{b,\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} h_\delta\big(y_i(b + \mathbf{x}_i^\top \mathbf{w})\big),$$

where $\big\{(\mathbf{x}_i, y_i)\big\}_{i=1}^N$ is the data set with the label $y_i \in \{+1, -1\}$. For high-dimensional data set, if there is not enough training samples, regularizer is used. One commonly used regularizer is the so-called elastic net regularized. It yields the elastic-net regularized HSVM:

$$\min_{b,\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} h_\delta\big(y_i(b + \mathbf{x}_i^\top \mathbf{w})\big) + \lambda_1 \|\mathbf{w}\|_1 + \frac{\lambda_2}{2}\|\mathbf{w}\|_2^2, \tag{1}$$

where $\lambda_1 \ge 0$ and $\lambda_2 \ge 0$ are parameters balancing the regularizers and the loss function.

2 Requirements

Include every item below in a single report and attach your code.

1. Write the gradient of the first term in the objective of (1).

2. Develop two solvers for (1) using the proximal gradient and also accelerated proximal gradient methods. Your solvers should be general enough to treat δ, λ_1 and λ_2 as inputs.

3. For each of the given data sets, run both your solvers to 500 iterations and plot distance of the objective values to the optimal value in terms of iteration number and also running time. [To obtain the "optimal" objective value, you can run the accelerated proximal gradient solver to sufficiently many (say 10,000) iterations, and pick the best one among the last 100 iterates.]

4. Use the training data in each of the given data sets to learn a classification function, i.e., $(\mathbf{w}, b)$, and then use the learned classification function to classify the samples in the testing data. Suppose the solution from the model is $(\bar{\mathbf{w}}, \bar{b})$. Then for a testing sample $\mathbf{x}$, its label can be predicted as $\text{sign}(\mathbf{x}^\top \bar{\mathbf{w}} + \bar{b})$.

   Report the prediction accuracy, i.e., the ratio between the number of correctly predicted samples and the number of all testing samples. [To have a higher accuracy, you need to tune the hyperparameters δ, λ_1 and λ_2.]