CSE599s, Spring 2014, Online Learning Due: 05/06/2014

Programming Assignment 1

Instructors: Ofer Dekel, Brendan McMahan *TA: De (Dennis) Meng*

Submission: Programming assignments involve both mathematical derivation and coding. Your submission should include a write-up that describes your derivation and explains your code for each sub-problem. Your code should print out the final results as required by each problem. Please submit your write-up (pdf file) and source code files in a single compressed package named "Coding1_YourFirstName_YourLastName" to the dropbox
`https://catalyst.uw.edu/collectit/dropbox/summary/demeng/31445`.
You can post questions on the discussion board
`https://catalyst.uw.edu/gopost/board/demeng/36570/`.
If you have trouble with `Matlab`, feel free to email TA Dennis, `demeng@uw.edu`.

In this programming assignment, we will implement algorithms for the online convex optimization and analyze their regret. The algorithms are:

- `Follow-The-Leader (FTL)`
- `Follow-the-Regularized-Leader (FTRL)`
- `Online Gradient Descent (OGD)`.

1. *Online Linear Optimization in 1D*

   (a) Convex quadratic functions in 1D can be generally expressed as $f(w) = zw + \frac{\lambda}{2}w^2$, where w is the scalar variable, z and $\lambda \geq 0$ are scalar constants. When $\lambda = 0$, $f(w)$ reduces to a linear function. Implement an optimizer to minimize $f(w)$ (for both $\lambda = 0$ and $\lambda \neq 0$ cases) on the interval $W = [-1, +1]$. This optimizer will be used to implement `FTL` and `FRTL` later in this problem.

   (b) In the online convex optimization framework, after the player makes a prediction w_t on each round, the adversary chooses a loss function $f_t : W \to \mathbb{R}$ and the player suffers the loss $f_t(w_t)$. Consider two different adversaries:

      i. The adversary chooses the function $f_t(w) = z_t w$ with $z_t \in [-1, 1]$ chosen to maximize $f_t(w_t)$ (for $w = 0$, the adversary selects $z_t = 1$). Implement this adversary.

      ii. The adversary chooses the function $f_t(w) = z_t w$ with $z_t = +1$ or $z_t = -1$ with equal probability. Implement this adversary.

   (c) For each of the adversaries described above, implement `FTL` with $T = 1000$; use the initial strategy $w_1 = 0$. Report the cumulative loss $\sum_{t=1}^{T} f_t(w_t)$, the cumulative loss of the best hypothesis in hindsight $\min_{w \in W} \sum_{t=1}^{T} f_t(w)$, and the final regret. (For the randomized adversary, report the mean quantities from 10 runs, using different randomization for each run).

   (d) For each of the adversaries described above, implement `FTRL` with $T = 1000$ and the regularization function $R(w) = \frac{1}{2\eta}w^2$, where η is the learning rate chosen based on the formula discussed in class. Report the same quantities requested in part (c).

2. *Online SVM for Email Spam Classification*

   Consider the problem of binary linear prediction with hinge loss, which was discussed in Lecture 1 (see Lecture 1 notes). Recall that the hinge loss function is defined as

$$f_t(w) = \max\{0, 1 - y_t w^T x_t\},$$

where $x_t, w_t \in \mathbb{R}^n$ and $y_t \in \{-1, +1\}$. Say that our goal is to detect email spam. On each round t, the player chooses a spam detector w_t and the adversary chooses an instance (x_t, y_t), where x_t is a feature vector that represents an email and y_t is its binary label (spam or non-spam). The player incurs a loss $f_t(w_t)$ and receives the feedback (x_t, y_t). Implement `FRTL` and `OGD` for this problem.

Dataset. We will use a small email dataset adapted from the UCI Machine Learning Repository. This dataset contains 4601 emails, each represented by 57 features ($n = 57$). Additional information on this dataset is available at `https://archive.ics.uci.edu/ml/datasets/Spambase`. Assume that the emails arrive in sequence and are classified one after the other, so the player makes a total of $T = 4601$ predictions.

We have already preprocessed the dataset to have the following structure. The 4601 feature vectors are stored as rows of the matrix `spam_inst` $\in \mathbb{R}^{4601\times 57}$ and are normalized to have an ℓ_2 norm of one. Spam and non-spam emails are labeled by 1 and -1 respectively and the 4601 labels are stored in a column vector `spam_label` $\in \mathbb{R}^{4601}$.

You can load the data in Matlab using the command `load('spam_data.mat')`.

Optimization software. We will use the optimization software `MOSEK` to solve each step of the `FTRL` algorithm. `MOSEK` solves large-scale batch convex optimization problems using a state-of-the-art interior-point optimizer. You do not have to know too much about `MOSEK`, but simply download and install it from

`http://mosek.com/resources/download/`.

You can get a trial license or free academic license. The license information is available on the above webpage. The installation manual may be helpful:

`http://docs.mosek.com/7.0/toolsinstall/index.html`.

Feel free to contact the TA if you have trouble with the installation.

(a) Characterize the subgradient set of the hinge loss function $f_t(w)$. Find an upper bound on the ℓ_2 norm of the subgradients for the given dataset.

(b) We apply the algorithm `FTRL` with the regularization function $R(w) = \frac{1}{2\eta}\|w\|_2^2$ for online SVM. In each round t, the prediction w_t is made by solving a convex optimization problem

$$\underset{w}{\text{minimize}} \sum_{i=1}^{t-1} \max\{0, 1 - y_i w^T x_i\} + \frac{1}{2\eta}\|w\|_2^2.$$

This optimization problem can be formulated as a quadratic optimization problem

$$\begin{array}{ll} \underset{\xi, w}{\text{minimize}} & \displaystyle\sum_{i=1}^{t-1} \xi_i + \frac{1}{2\eta}\|w\|_2^2 \\ \text{subject to} & \xi_i \geq 0, \quad i = 1, \ldots, t-1 \\ & y_i w^T x_i \geq 1 - \xi_i, \quad i = 1, \ldots, t-1. \end{array}$$

where $w \in \mathbb{R}^n$ and $\xi \in \mathbb{R}^{t-1}$. You do not need to worry about how to solve this optimization problem; instead you can simply call the Matlab function `l2svm` provided in this homework package using the following syntax.

`[w, fval] = l2svm(spam_label, spam_inst, eta).`

You need to add the path of `MOSEK` toolbox to top of the Matlab file `l2svm.m`, see comments in the file. In each round t, the input `spam_label` is the vector of $t-1$ labels, `spam_inst` is the matrix (of size $(t-1) \times 57$) of the corresponding feature vectors, `eta` is the learning rate η. The output `w` is the optimal $w^\star$, and `fval` is the optimal objective value.

To compute the regret, we need to calculate the cumulative loss of the best classifier in hindsight $\min_{w \in W} \sum_{t=1}^{T} f_t(w)$, where $W = \{w : \|w\|_2 \leq B\}$. This can be obtain by solving an optimization problem

$$\begin{array}{ll} \underset{w}{\text{minimize}} & \sum_{i=1}^{T} \max\{0, 1 - y_i w^T x_i\} \\ \text{subject to} & \|w\|_2 \leq B \end{array}$$

where $w \in \mathbb{R}^n$. Again, you are provided a Matlab function to solve this optimization problem. You can imply call the Matlab function `l2svmball` provided in this homework package using the following syntax.

`[w, fval] = l2svmball(spam_label, spam_inst, B).`

You need to add the path of `MOSEK` toolbox on top of `l2svmball.m` as well.

Implement online SVM with `FTRL` on the provided dateset. Choose $B = 2$, and again use the learning rate that minimizes the regret bound. Report the learning rate, the cumulative loss $\sum_{t=1}^{T} f_t(w_t)$, the cumulative loss of the best predictor in hindsight $\min_{w \in W} \sum_{t=1}^{T} f_t(w)$, and the regret.

Note that, although `MOSEK` is highly efficient, the runtime on the given dataset could be as long as 15-30 minutes depending on your computer configuration. So it might be a good idea to test your code on a small subset of data first.

(c) We apply the algorithm `OGD` for online SVM. In each round t, the prediction w_t is made by the update

$$w_t = w_{t-1} - \eta z_{t-1},$$

where $z_{t-1} \in \partial f_{t-1}(w_{t-1})$, η is the learning rate. If 0 is in the subgradient set, always choose it (i.e. no need to do update on w).

Implement online SVM with `OGD` on the provided dateset. First, use the same learning rate used for `FTRL`; report the learning rate and the cumulative loss $\sum_{t=1}^{T} f_t(w_t)$. Using the the cumulative loss of the best predictor in hindsight calculated in the previous part, compute the regret.

(d) Compare the runtime of `FTRL` and `OGD`. What can you conclude?