Assignment-1 (CS-729)

October 17, 2014

Note: You may use *any* programming language of your choice and any software/code available from internet or otherwise. However, I recommend using `cvx` available at `http://cvxr.com/cvx/`. Do not copy code from other teams. Do not ask them how they are doing this assignment. In case you have a doubt approach me directly. It is easy to identify defaulters during the demo.

Let $\mathcal{X} \subset \mathbb{R}^n$, loss be square-loss and

$$\mathcal{F}_W = \left\{ f \mid \exists\ w \in \mathcal{H}_k,\ \|w\|_{\mathcal{H}_k} \le W \ \ni f(x) = \langle w, \phi_k(x) \rangle_{\mathcal{H}_k} \right\},$$

where $k(x,z) = e^{-\frac{1}{2}\|x-z\|^2}$ is the (normalized) Gaussian kernel, and $\phi_k, \mathcal{H}_k$ are the kernel induced map and Hilbert space. Assume that the dataset is normalized[1]. From our lectures we have the following different algorithms for simultaneously selecting the right $\hat{W}_m$ from the set $\{10^{-3}, 5 * 10^{-3}, 10^{-2}, 5 * 10^{-2}, 10^{-1}, 5 * 10^{-1}, 1, 5, 10, 50, 100, 500, 1000\}$ and the right $\hat{w}_m \ \ni \|w\| \le \hat{W}_m$:

1. Gauranteed/Structural Risk Minimization (SRM) + ERM:

    (a) For each W solve the corresponding ERM and obtain the minimum empirical risk. Also, compute the empirical maximum dis-

[1]Normalize data in each input feature to zero mean and unit variance. After this only, split into training, validation, test etc. (if required).

crepency[2]. Then find the $\hat{W}_m$ that minimizes the sum of these two terms. Report $\hat{w}_m$ as the empirical risk minimizer corresponding to $\hat{W}_m$.

(b) For each W solve the corresponding ERM and obtain the minimum empirical risk. Also, compute the empirical Rademacher average[3]. Then find the $\hat{W}_m$ that minimizes the sum of the first term and twice the second. Report $\hat{w}_m$ as the empirical risk minimizer corresponding to $\hat{W}_m$.

(c) For each W solve the corresponding ERM and obtain the minimum empirical risk. Then find the $\hat{W}_m$ that minimizes the sum of this and $2\frac{WR}{\sqrt{m}}$. Report $\hat{w}_m$ as the empirical risk minimizer corresponding to $\hat{W}_m$.

(d) Solve[4]:

$$(\hat{W}_m, \hat{w}_m) \equiv \text{argmin}_{W \geq 0, \|w\| \leq W} \hat{R}_m[f] + 2\frac{WR}{\sqrt{m}}.$$

2. ERM + ERM:

(a) Split given training dataset into two equal parts: training and validation. For each W, find empirical risk minimizer using the new training data alone. With each such minimizer, compute error on validation set, pick the best and report the corresponding W as $\hat{W}_m$ and corresponding ERMizer as $\hat{w}_m$.

(b) Repeat above 100 times with various random splits. Pick $\hat{W}_m, \hat{w}_m$ by looking at sum of all validation errors.

(c) Pick $\hat{W}_m$ that minimizes 5-fold cross validation error. For definition of k-fold cross validation error, please refer to the internet.

(d) Repeat above 100 times with various random splits. Pick $\hat{W}_m, \hat{w}_m$ by looking at sum of all 5-fold cross-validation errors.

[2] Note that, for every W, ERM is a convex program and hence can be solved using cvx. However, computing the empirical maximum discrepency is NOT convex. But, fortunately, it can be solved efficiently. Please read section 3.5.2 in [1]. The idea is to write down the SDP relaxation, which is exact in this special case, and solve it using cvx.

[3] For this you will need to solve a maximization problem 2^m times, one for each pattern of σs. Each of this max. can be again solved using the SDP relaxation, which in this case, is solving a generalized eigen value problem.

[4] This is a convex program and can be solved by cvx.

Your goal is to compare all these 8 methods in terms of computational effort and accuracy (as measured on an independent test set) for various values of m. More specifically, do the following in case of your dataset:

Split (randomly) your dataset into two equal parts: training and test. Now, consider increasingly bigger subsets of the training set $m = 5, 6, 7, 8, 9, 10, 11, 12, 13, \ldots$ as the actual training set for each of the 8 methods. With each m and each of the 8 methods above, note the cpu-time for entire process of choosing $\hat{W}_m, \hat{w}_m$ and also note the accuracy on test set. Now plot a cpu-time vs. test set accuracy plot for each of the method. You may skip plotting an m value and method combination that takes "too long". You should automate your code so that I can run it on any dataset including the one assigned to you.

References

[1] A. Nemirovski. Lectures on Modern Convex Optimization. `http://www.isye.gatech.edu/faculty-staff/profile.php?entry=an63`, 2005.