Convex Optimization

Project 1

Spring 1401
Due date: 30th of Ordibehesht

Predicting complete rankings. A (complete) ranking of K items consists of an ordering of the items from rank 1 to rank K. For example, these could be K candidates, ranked from 1 (best) to K (worst), or the order in which K horses cross the finish line in a race. We represent a ranking of K items as a vector $\pi \in \mathbf{R}^K$, with π_i the rank of item i. In the vector π, the numbers $1, \ldots, K$ each appear exactly once (so it can also be considered a permutation), so there are K ! different rankings. We will let $\mathcal{P} \subset \mathbf{R}^K$ denote the set of all $K!$ rankings.

For example with $K = 3, (2,3,1)$ and $(1,3,2)$ are two of the six possible rankings. In the first ranking, item 1 has rank 2, whereas in the second ranking, item 1 has rank 1 . Both rankings agree that item 2 has rank 3 .

There are many ways to assign a distance between two rankings π and σ, but we will use a simple one, $(1/2)\|\pi - \sigma\|_1$. This distance is zero if and only if $\pi = \sigma$, and one if and only if π and σ assign the same rank to all items except two, whose ranks are off by one. The maximum possible distance is $K^2/4$ for K even and $\left(K^2-1\right)/4$ for K odd, achieved by, e.g., $\pi = (1,2,\ldots,K)$ and $\sigma = (K, K-1, \ldots, 1)$. The average distance between two randomly chosen rankings is $\left(K^2-1\right)/6$. (These observations are not relevant for this problem, but only meant to give you an idea of the range and scale of the distance between rankings.)

We wish to build a predictor of an outcome which is a ranking, based on a vector of features. We denote the predictor as $P : \mathbf{R}^d \to \mathcal{P}$, where $P(x)$ is the ranking we predict when the feature vector is $x \in \mathbf{R}^d$. We will judge a predictor by the average distance between the true ranking and the predicted one, on a test set of data $(x_i^{\text{test}}, \pi_i^{\text{test}}), i = 1, \ldots, N^{\text{test}}$ (that presumably was not used to develop or fit the predictor):

$$\frac{1}{2N^{\text{test}}} \sum_{i=1}^{N^{\text{test}}} \left\|\pi_i^{\text{test}} - P\left(x_i^{\text{test}}\right)\right\|_1 .$$

We refer to this quantity as the average test error of the predictor. (The smaller this is, the better the predictor performs on the test data set.) We will consider a simple predictor of the form $P(x) = \Pi(\theta x)$, where $\theta \in \mathbf{R}^{K \times d}$ is the predictor coefficient matrix, and $\Pi : \mathbf{R}^K \to \mathcal{P}$ is Euclidean projection onto $\mathcal{P}$. (We will describe this projection in more detail below, but for now we note that if there are multiple rankings that are closest to θx, we arbitrarily choose one.)

We choose the predictor parameter matrix θ to minimize

$$\frac{1}{2N} \sum_{i=1}^{N} \|\pi_i - \theta x_i\|_1 ,$$

where $(x_i, \pi_i), i = 1, \ldots, N$, is some given training data. (Note that this objective would become the average distance between the true and predicted rankings if we replace θx_i with $\Pi(\theta x_i)$, but then the objective is no longer convex.)

Projection onto rankings. You can use the following, without deriving or justifying it. The projection $\pi = \Pi(y)$ is the vector of rank orders of the entries of y in nondecreasing order. For example with $y = (1.1, -0.3, 0.5, 0.4)$, we have $\Pi(y) = (4,1,3,2)$, since the first entry of y is the largest (i.e., has rank 4), the second entry of y is the smallest (i.e., has rank 1), and so on. So we can compute $\Pi(y)$ by sorting the entries of y (breaking any ties arbitrarily), keeping track of the sort ordering.

Explain how to fit the predictor using the training data with convex optimization.

The data file ranking_est_data.* contains functions that generate synthetic training and test data, as well as a function that implements Π. The data are in the matrices X_train, pi_train, X_test, pi_test, and the projection Π is given in Pi(). Fit the predictor using the training data, and give the average distance between the true and predicted ranking on both the training and test data sets.

Good Luck!