EE3-23: Assignment 3

Spring, 2017

Instructions:

Solve all problems.

You need to submit a single zip file on Blackboard, named `assignment3_<name>.zip` where `<name>` is your name. The file should contain a report in PDF (write your name on your solution) named `assignment3_<username>_<name>.pdf` where `<username>` is your blackboard user name, and your code and other relevant files. Your code must include a main file named `assignment3_<username>` with a possible extension (such as `.m` or `.py`) or a Makefile which produces the main file on any standard linux system: the main file should be either an executable or a script file, which generates all the data and plots required in your solution.

Justify your answers. You are allowed to consult with others (mention their names at the beginning of your solution), but you need to work out and write up your solution alone. The use of any written sources (e.g., books or the web) is permitted.

Include the following pledge into your answer:

I, <YOUR NAME>, pledge that this assignment is completely my own work, and that I did not take, borrow or steal work from any other person, and that I did not allow any other person to use, have, borrow or steal portions of my work. I understand that if I violate this honesty pledge, I am subject to disciplinary action pursuant to the appropriate sections of Imperial College London.

If you don't include your pledge, you will get 0%.

Deadline: March 9, 20:00, followed by a 5-minute grace period, after which no submissions are accepted.

Problem 1 (Approximate ERM)**.** Suppose you need to solve a binary classification problem with the binary error (i.e., the error is zero if the prediction is correct and 1 otherwise). Assume you are given data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ drawn i.i.d. from the true data distribution. For any $\varepsilon \geq 0$, give a bound on the test error of any hypothesis h_ε that is ε-optimal on the training set $\mathcal{D}$, that is, it satisfies $\widehat{R}_n(h_\varepsilon) \leq \varepsilon + \min_{h \in \mathcal{H}} \widehat{R}_n(h)$;[1] more precisely, give a high-probability upper bound on $R(h_\varepsilon) - \min_{h \in \mathcal{H}} R(h)$. **[4 points]**

Problem 2 (Optimal predictor for the absolute error)**.** Given data points $y_1, \ldots, y_n \in \mathbb{R}$, find a value m minimizing the average absolute error $\frac{1}{n} \sum_{i=1}^{n} |y - y_i|$. **[4 points]**

Remark: More generally, one can show that for any random variable pair $(X, Y) \in \mathcal{X} \times \mathbb{R}$, and loss function $\ell(y', y) = |y - y'|$, the expected error $\mathbb{E}\left[\ell(f(X), Y) | X\right]$ is minimized for any $f : \mathcal{X} \to \mathbb{R}$ such that $f(X)$ is a median of the conditional distribution Y given X, where for a real random variable Z, any real number m satisfying $\mathbb{P}\left(Z \geq m\right) \geq 1/2$ and $\mathbb{P}\left(Z \leq m\right) \geq 1/2$ is a median.

[1] In this problem—to simplify the treatment—it is assumed that all minima exist. However, this is not necessary, and each minimum can be replaced with the corresponding infimum.

Problem 3 (Movie recommendation)**.** Consider the movie recommendation problem studied in class: in this problem the goal is to predict the ratings users assign to movies in a minimum mean-squared error sense. In the provided dataset (`movie-data.zip`), viewer ratings corresponding to 9066 movies and 671 users are provided together with a feature vector for each movie (whose components are indicators of different genres).[2] The data is given in `csv` format with a header row describing the meaning of each column:[3]

- `movie-titles.csv`: title and id of the movies in the dataset (only for information purposes);
- `movie-features.csv`: binary (0-1-valued) genre features for the movies, each row contains variables `movieId,feature1,...,feature18`;
- `ratings-train.csv`: training set $\mathcal{D}$, containing `(userId,movieId,rating)` triplets in each row;
- `ratings-test.csv`: test set $\mathcal{T}$, containing `(userId,movieId,rating)` triplets in each row.

Here `userId` $\in \{1, 2, \ldots, 671\}$, `movieId` $\in \{1, 2, \ldots, 9066\}$, and `rating` $\in \{0.5, 1, 1.5, \ldots, 5\}$.

The task in this problem is to develop and test different predictors for the movie ratings whose performance is measured by the squared error. Training should only be performed on the training set, final test results should be measured on the test set, and no parameter of the training method (including which algorithm to use) should be selected based on the test set. For each sub-question, explain clearly and concisely what you do and why (including formulas with derivations if necessary), and present the answers in the most meaningful way. Your solution to the problem must not be longer than 4 pages with a minimum font size of 10pt (it can be shorter).

(a) Constant base predictors: The simplest predictor for this problem is to provide a rating prediction for any movie (independent of the user) or any user (independent of the movie). Find such predictors and report their performance. Which one would you prefer, constant ratings based on the movies or the users? **[2 points]**

(b) Linear regression baseline: Based on the features provided for the movies, estimate the rating given by each user using linear regression, possibly with some added regularization. That is, if $\hat{v}_j \in \mathbb{R}^d$ is the (possibly normalized) feature vector for movie j, find weights $u_i \in \mathbb{R}^d$ for user i with the aim of minimizing the error $(r_{ij} - u_i^\top \hat{v}_j)^2$ over the test set $\mathcal{T}$. **[6 points]**

(c) Linear regression with transformed features: Suggest some way of transforming the features in a non-linear manner and repeat the above with the transformed features. **[6 points]**

(d) Collaborative filtering: Learn simultaneously the weights for the users and the features for the movies by devising predictions of the form $u_i^\top v_j$ where $u_i, v_j \in \mathbb{R}^K$ for some $K \geq 1$ and v_j is the K-dimensional (learned) feature vector for movie j. **[10 points]**

[2] The data is extracted from a dataset provided by GroupLens about ratings collected by MovieLens `https://grouplens.org/datasets/movielens/`. For licensing and other information, see `http://files.grouplens.org/datasets/movielens/ml-latest-small-README.html`.

[3] Recommendation: use built-in csv-reader functions to read the files.

In all the above questions, select the parameters (such as K, coefficients for penalties, step size for gradient descent, etc.) in a disciplined way (e.g., using cross-validation) and do not forget to report how these are done. If you are using gradient descent, do not forget to derive the update rule and give the exact algorithm you are using. Analyze the results and report your findings (including a comparison of the solutions you get at different stages of this problem). You can use any available standard library functions in your code (e.g., for ridge regression).