Computing for Data Sciences

PGDBA, First Year, Indian Statistical Institute

Assignment 1

Posted on 9 August 2015 | Seek clarification by 14 August 2015 | Submit by 23 August 2015

Your solution set should be in a compressed folder `GroupXX_assign1_solution.zip`, where `XX` is your group number as submitted to me at the beginning of the course (i.e., `XX` = 01 to 13). The solution set should be emailed to `sg.sourav@gmail.com`, with a copy to all your group members, before the midnight of the submission deadline. Late submissions will not be considered.

The solution set should contain a *single* PDF file for the theoretical parts of all the solutions, and individual source files for each `Python` and/or `R` code that you write to solve the problems. Specific submission policies for the solutions is provided with the respective problem statements.

Problem 1 [20 points]

Problem statement: Write a simple Python program, without exploiting the power of special functions or libraries, that takes as input two positive integers – value x (max 10 digits), precision p (max 20 digits) – and outputs the value of $\sqrt{x}$, correct upto p digits after the decimal point.

Submission policy: Submit your proposed algorithm, a discussion on the runtime complexity of the algorithm, and a proof of correctness (may be informal) of the algorithm as the theoretical part of your solution. Submit a single source file for the code, named `sqrt_assign1_solution.py`.

Problem 2 [40 points]

Background: Given an m-sample training set $\{x_1^{(i)}, x_2^{(i)}, \ldots, x_k^{(i)}, y^{(i)}\}$ for $i = 1, 2, \ldots, m$, one may assume an initial linear relation between the features $\{x_1, x_2, \ldots, x_k\}$ and y as follows:

$$y \;\approx\; h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_k x_k,$$

and try to learn the optimal values of the relational parameters $\theta = \{\theta_0, \theta_1, \ldots, \theta_k\}$ using an *ordinary least squares* model for linear regression, which minimizes the cost function

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 = \frac{1}{2}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \cdots + \theta_k x_k^{(i)} - y^{(i)}\right)^2.$$

Batch Gradient Descent is a well-known representative for a broad class of iterative algorithms for minimizing $J(\theta)$, based on the least mean squares (LMS) update rule, also known as the Widrow-Hoff learning rule, with the parameter α controlling the learning rate, as follows.

$$\theta_j \;\leftarrow\; \theta_j - \alpha\frac{\partial J}{\partial \theta_j} = \theta_j - \alpha\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x_j^{(i)} \qquad \text{for each } j = 0, 1, \ldots, k.$$

Problem statement: Write the algorithm for *batch gradient descent*, as discussed above, which takes into consideration *all* m training samples $\{x_1^{(i)}, x_2^{(i)}, \ldots, x_k^{(i)}, y^{(i)}\}$ at every iteration of the update rule for *each* parameter θ_j (denoted by the summation). Discuss the runtime of the algorithm, and the effect of the *learning rate* α, if any, on the runtime and convergence of the algorithm. Discuss the choice of the initial guess for the parameters $\theta = \{\theta_0, \theta_1, \ldots, \theta_k\}$, and the effect of this initial guess, if any, on the runtime and convergence of the algorithm.

Submission policy: Submit the complete write-up as a part of the PDF file.

Problem 3 [40 points]

Problem statement: Notice that in the batch gradient descent algorithm, each update of θ requires you to scan through the entire set of m training samples, making it slow for large m. Modify the algorithm for gradient descent so that one may start updating θ_j (for every j) by scanning a single training sample at a time, and continue this way for the complete training set. Write the modified algorithm for gradient descent, and discuss its runtime and convergence.

Submission policy: Submit the complete write-up as a part of the PDF file.

Problem 4 [50 points]

Problem statement: Implement both the above algorithms (as in Problems 2 and 3) as two individual functions in a single R file, where each function takes as input a training data set, an initial guess for parameters θ (as applicable), and a measure of precision for convergence of the algorithm (as applicable), and outputs the optimal values of the parameters $\theta = \{\theta_0, \theta_1, \ldots, \theta_k\}$.

Submission policy: Submit a single file `grad_assign1_solution.R` containing both functions.

Problem 5 [50 points]

Background: Write the cost function of gradient descent in vectorial format to obtain:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 = \frac{1}{2}(X\theta - y)^T(X\theta - y),$$

where the feature set X is an $m \times k$ matrix built of rows $(1, x_1^{(i)}, x_2^{(i)}, \ldots, x_k^{(i)})$, parameter vector θ is a $k \times 1$ vector $(\theta_0, \theta_1, \ldots, \theta_k)^T$, and the actual values y is an $m \times 1$ vector $(y^{(1)}, y^{(2)}, \ldots, y^{(m)})^T$.

Problem statement: Minimize the cost function $J(\theta)$ using the notion of matrix calculus, by setting the 'derivative' of $J(\theta)$ to zero, and obtain the optimal analytic solution for θ that minimizes $J(\theta)$. The equation set representing such an analytic solution is called the *normal equations* for θ. Implement linear regression in R using the analytic model for θ you just computed. Your function should take as input a training data set, and output the optimal value of the parameter vector $\theta = (\theta_0, \theta_1, \ldots, \theta_k)^T$, computed using the *normal equations* you derived.

Submission policy: Write the theoretical part of the solution as a part of the PDF file, and submit a single code file `neqn_assign1_solution.R` containing the regression function.

Problem 6 [50 points]

Background: Suppose that we redefine the cost function of gradient descent as follows:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m} w^{(i)} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 = \frac{1}{2}\sum_{i=1}^{m} w^{(i)} \left(\theta_0 + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \cdots + \theta_k x_k^{(i)} - y^{(i)}\right)^2,$$

to introduce a local weight factor $w^{(i)}$ associated with each training sample $\{x^{(i)}, y^{(i)}\}$, where $w^{(i)}(x) = \exp(-(x^{(i)} - x)^T(x^{(i)} - x)/2\tau^2)$ is calculated individually at each query vector x.

Problem statement: Represent the aforesaid definition of $J(\theta)$ in vectorial format, and derive an analytic solution for θ that minimizes $J(\theta)$, using the notion of *normal equations*. You must justify the derivation of normal equations for θ in this weighted scenario through matrix calculus, as well as present an argument to justify this idea using the notion of fundamental subspaces.

Implement this locally weighted linear regression in R using the analytic model for θ you just computed. Your function should take as input a training data set, a value of τ to compute the weights (as applicable), and output the optimal value of the parameter vector $\theta = (\theta_0, \theta_1, \dots, \theta_k)^T$, computed using the *weighted normal equations* you derived.

Submission policy: Write the theoretical part of the solution as a part of the PDF file, and submit a single code file `wght_assign1_solution.R` containing the regression function.

Problem 7 [50 points]

Background: In the regular R installation, the library `MASS` contains a dataset `Boston`, depicting 'Housing Values in Suburbs of Boston', containing 506 data samples with 14 features/columns. You may access the dataset using the commands `require(MASS)` followed by `data(Boston)`.

Problem statement: Create a basic single-feature training set from the `Boston` dataset, using `Boston$rm` as $x^{(i)}$ and `Boston$medv` as $y^{(i)}$, and apply each of the four regression functions – as in `grad_assign1_solution.R`, `neqn_assign1_solution.R` and `wght_assign1_solution.R` – to compute the linear regression line in each case. Write an R code to use the previous functions you wrote, and output the optimal values of θ in each case. Plot the training set $(x^{(i)}, y^{(i)})$ superimposed with each regression line. In case of the locally weighted linear regression, use $\tau = 0.1, 0.5, 1.0$ to obtain three different solutions, and plot all three lines on the same figure.

Submission policy: Provide the optimal values of θ and the plots as a part of the PDF file, and submit a single code file `regr_assign1_solution.R` containing the complete R code.

Properly acknowledge every source of information that you referred to, including discussions with your friends outside the group. Verbatim copy from any source is strongly discouraged, and plagiarism will be heavily penalized. It is strongly encouraged to include in the write-ups your own interpretation and analysis of every problem, and to write the codes completely on your own.