CMPSCI 389 Homework 2
Spring 2022
Assigned: Feb 15, 2022; Due: Feb 27, 2022 @ 11:59pm Eastern

Instructions: This assignment has only a programming component. You should submit two files: `main.py` and `hyper.csv` via Moodle. An auto-grader will grade your code to check your code for correct output. As such, your program must meet the requirements specified below. We will also be using cheating detection software, so, as a reminder, you are allowed to discuss the homework with other students, but you must write your code on your own.

Please get an early start. The assignment was originally going to be due on Thursday February 24, but I realized that we would not make significant grading progress until Monday morning, and so it would not be detrimental to the grading pipeline to give you until late Sunday night. Note that there are no office hours on Thursdays or Fridays and that Piazza questions will not necessarily be answered on weekends.

1 Programming (100 points)

For this assignment you will implement gradient descent on the loss function

$$l(w) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

Your implementation should include two features:

1. You should use input normalization. First, find the largest and smallest values that each features takes *on the training data*. Next, normalize all inputs to be in the range $[-1, 1]$ using the equation:

$$x_{i,j}^{\text{new}} = 2\frac{x_{i,j}^{\text{old}} - x_j^{\min}}{x_j^{\max} - x_j^{\min}} - 1,$$

   where $x_j^{\min}$ and $x_j^{\max}$ are the minimum and maximum values of the j^{th} feature within the training data.

2. A extra "offset" or "y-intercept" feature should be included. That is, each input x_i should have a 1 appended to the list of features. This offset feature does not need to be further normalized.

Your program should read three files:

1. `train.csv` contains training data, just as in the previous assignment.
2. `test.csv` contains testing data, just as in the previous assignment.
3. `hyper.csv` contains hyperparameters, α, N_1, and N_2.

After reading these three files, your program should run gradient descent on the loss function specified above using the step size α, for a total of N_2 iterations, and starting with all weights equal to zero. The points found during the first N_1 iterations should be printed to `log.csv`, with one point per line and commas separating the individual weights. The initial weight, w_0 (that is $k = 0$) should be all zero, and should be included in `log.csv`. So, for example, if $N_1 = 1$ then `log.csv` should have a single line with zeros separated by commas. If $N_1 = 2$ then `log.csv` should contain two lines, the first of which is all zeros and the second of which is w_1 (that is $k = 1$). After gradient descent has finished running (all N_2 iterations, not just N_1) your program should compute the RMSE on the testing data, and should print this RMSE to a different file, `out.csv`.

Next, search for hyperparameters that produce a final RMSE of at most 0.762 and where $N_2 \leq 1000$. Submit the file `hyper.csv` that results in this RMSE.

In summary, you should submit a file named `main.py` that produces `log.csv` (containing the first N_1 iterates of gradient descent) and `out.csv` (containing the RMSE on the test set after N_2 iterations of gradient descent) as output. Your program may be tested on other input files. You should also submit `hyper.csv`, which contains hyperparameters that result in a final RMSE of at most 0.762 and has $N_2 \leq 1000$.

To help you to debug your program, we have provided a directory named `test` that contains other example input and output files from a correct implementation. Your program should exactly match (to within errors due to floating point arithmetic) the values in `out.csv` and `log.csv`. The file `hyper.csv` that you submit should contain hyperparameters for the complete GPA data set in the parent directory, *not* hyperparameters for the simplified data in the sub-directory named `test`.