אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev

Applied Deep Learning
O. Azencot
Spring 2023

Assignment 2: **Basic Optimization and Machine Learning**
Deadline: Apr. 16, 5 pm, 2023

In this home assignment, you will apply some of the concepts we learned in class related to optimization and machine learning. Submission is in pairs. Your submission should include two files: 1) PDF file named `report.pdf` containing the answers to the tasks below, and 2) A compressed container named `code.zip` containing all the code you used. Please submit your work via Moodle. For questions, use the Slack channel, or contact me via email.

1 Background

In the third lecture, we discussed in detail the linear least squares problem, and its solution using a simple gradient descent algorithm. In addition, we studied in the fourth lecture about basic machine learning concepts such as train and test sets and overfitting. We will briefly recall these concepts below.

Given a constant matrix $A \in \mathbb{R}^{m \times n}$ and a constraints vector $b \in \mathbb{R}^n$, the *least squares* problem deals with finding a vector $x \in \mathbb{R}^m$ such that applying A to it yields b, approximately. Formally,

$$\min_x |Ax - b|_2^2 \ , \tag{1}$$

namely, among all possible x vectors, we look for the optimal x^* that minimizes the objective $|Ax - b|_2^2$ where $|z|_2^2$ is the $\mathcal{L}_2$ (squared) norm of vectors, that is, $|z|_2^2 = \sum_{j=1}^k z_j^2$ for $z \in \mathbb{R}^k$.

In class, we learned that problem (1) can be solved using the *gradient descent* algorithm. Generally, given an initial guess of x^* denoted by x_0, the algorithm iteratively changes x_k by subtracting the gradient of the objective $|Ax - b|_2^2$ from x_k, evaluated at the point x_k, where k is the step. If we identify that the gradient norm at x_K is close to zero then we say that the algorithm converges, and we denote $x^* := x_K$. In pseudocode, the gradient descent algorithm for the problem (1) reads

$$\begin{aligned} &\text{while } 2|A^T Ax - A^T b|_2^2 > \delta \text{ do} \\ &\qquad x \to x - 2\epsilon(A^T Ax - A^T b) \end{aligned} \tag{2}$$

The parameters δ, ϵ are small nonnegative numbers chosen by the user.

Given a collection of pairs of points $\{(a^{(j)}, b^{(j)})\}_{j=1}^N$, where $a^{(j)} \in \mathbb{R}^n$ is a feature vector and $b \in \mathbb{R}$ is the output variable, we generate the *train and test sets* by randomly selecting 80% of the pairs to form the train set, and the remaining pairs form the test set. We denote by $A_{\text{tr}}, b_{\text{tr}}$ the train constraints matrix and its outputs, respectively. This matrix is defined by organizing the train

points $\{a^{(j)}\}$ in its rows, i.e., $A_{\text{tr}} = [(a^{(j)})^T]$, and similarly, $b_{\text{tr}} = (b^{(j)})$ for every j selected for the train set. We similarly denote by A_{te} and b_{te} the constraints matrix and output vector for the test set. We say that a machine learning algorithm *overfits* if its performance on the train set is good, however, it obtains poor results on the test set. For instance, on the least squares problem, the train error $|A_{\text{tr}}x^* - b_{\text{tr}}|_2^2$ for the optimal x^* is significantly smaller than the test error $|A_{\text{te}}x^* - b_{\text{te}}|_2^2$. We often view this error during optimization, i.e., we use the current point x_k of the optimization instead of the last point x^*.

2 Data

You will work with the `diabetes` dataset, which can be loaded using the `sklearn` package. This dataset includes a Diabetes listing of 442 patients with 10 feature variables and a single output variable [1]. Solving the problem directly would involve forming a 442×10 constraints matrix A and a 442 vector of outputs b. However, as you will see below, we will consider two variations of this problem that also involve train and test sets.

3 Tasks

1. Solve the linear least squares problem on the **entire** diabetes dataset using gradient descent. You are free to choose the initial point, step size ϵ, and the stopping condition δ, as you wish. Attach to your report the error plot of this procedure, namely a graph of the error $|Ax_k - b|_2^2$ where x_k is the point at step k.

2. Split the data to a train and test set with a standard 80%/20% split. Solve the least squares problem on the train set, and evaluate it on the test set. Attach to your report a graph of the train error jointly with the graph of the test error (see the definitions for the train and test errors in Sec. 1). Based on the graph, try to estimate if your algorithm underfits or overfits the data. Notice that you might need to play with the hyperparameters such as the step size and stopping condition to reach a reasonable convergence to assess fitting considerations.

3. Repeat the last task (Task 2) ten times, where at each repetition, you re-generate the train and test sets based on a different split. Keep all error graphs and compute their average. Do you observe a different fitting with the average graph? for instance, an underfitting model became overfitting or the other way around? What about the minimum of these graphs? Please discuss the behavior of the average graph and the minimum graph in your report and compare their characteristics.

4 Additional Comments

1. Do not attach code to the report. Please follow the guidelines in terms of how to attach code to your submission.

2. A significant portion of the grade is dedicated to the neatness and descriptiveness of the report. You should make all the figures to be as clear as possible.

3. At the same time, the report should not be too long. Please aim for an (at most) 8 page document.

4. For a least square example on the `diabetes` dataset, please see an example in this link.

References

[1] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.