CMPUT 466 Coding Assignment 1

Linear Regression and Method Optimization

Eden Zhou

zhou9@ualberta.ca

PROBLEM 1

Questions & Experiment Results

1. [20% for correct implementation of linear regression] With settings *M* = 5000, *var1* = 1, *var2* = 0.3, *degree* = 45, report the weight and bias for x2y and y2x regressions. The submission should be four numbers in two rows:

*w*_x2y → 0.8257816181788808	*b*_x2y → -0.007496215212965558
*w*_y2x → 0.80390969615798	*b*_y2x → 0.008530614496293935

2. [10%] Three plots of regression models in a row, each with *var2* = 0.1, 0.3, 0.8, respectively. (Other settings remain intact: *M* = 5000, *var1* = 1, *degree* = 45).

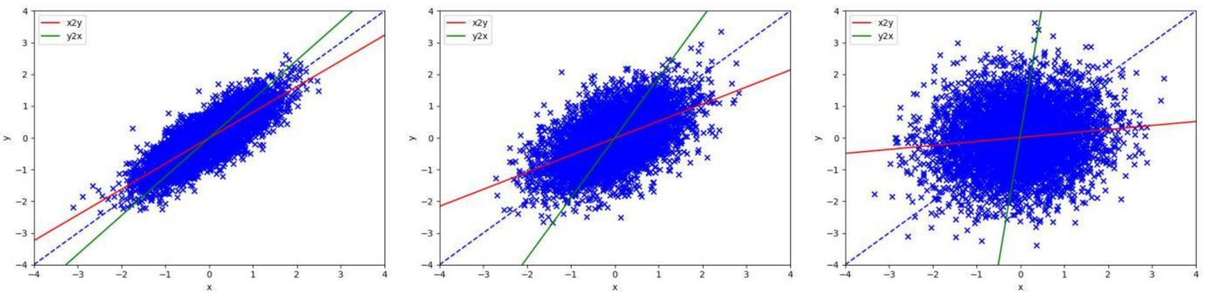

Figure 01: x2y and y2x regression models based on *M* = 5000, *var1* = 1, *degree* = 45, and *var2* = 0.1, 0.3 and 0.8 (from the left to the right).

3. [5%] A description of the phenomena found in 1) and 2).

Based on the plots shown above, we found the data distribution becomes flatter and more hanging on the central data line as *var2* decreases; As *var2* increases, both regressions deviate more from each other.

Therefore, smaller *var2* (i.e., a larger gap between *var1* and *var2*) may bring more significant relations in regression learning.

4. [15%] We now set *var2* = 0.1, but experiment with different rotation degrees. The student should design a controlled experimental protocol, plot three figures in a row with different settings, and describe the findings.

In this part, we control the variable settings of *M* (=5000), *var1* (=1), and *var2* (=0.1) to test how different choices of *degree* (=30, 90, 180, respectively) influence the learning process.

Results from three cases are plotted below:

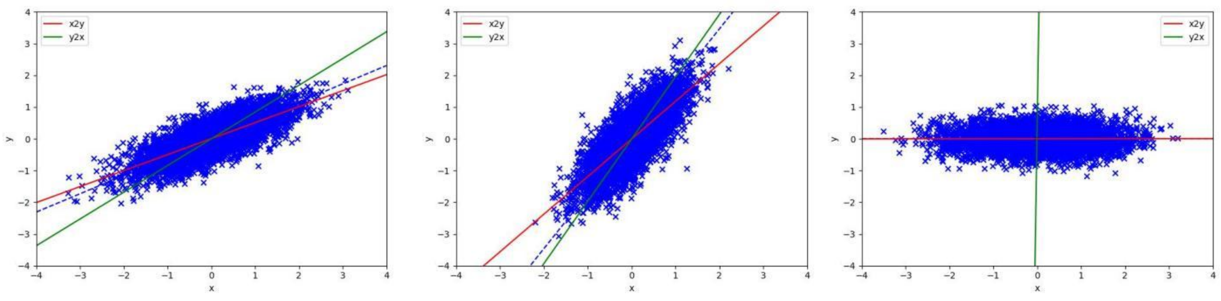

Figure 02: x2y and y2x regression models based on *M* = 5000*, var1* = 1*, var2* = 0.1, and *degree* = 30, 60 and 180 (from the left to the right).

When the rotation angle is a multiple of 90, one of the two linear regression methods approximates the dotted central data line and crosses the regression line generated by the other method. In this way, when a rotation degree is selected between multiples of 90 (e.g., a multiple of 45), both evaluations will be more acceptable and better reflect the distribution (i.e., may help avoid poor fitting conditions). For rotation degrees around 45, both regression models present similar results with positive slopes.

PROBLEM 2

Questions & Experiment Results

a. [30%] After reaching the maximum number of iterations, we pick the epoch that yields the best validation performance (the lowest risk), and test the model on the test set. Without changing default hyperparameters, we report three numbers and two plots:

Based on the iteration results from the cross-validation process, we found:

[*MaxIter*=100, *BatchSize*=10]

- The number of epoch that yields the best validation performance is 100 (=*MaxIter*);
- The validation performance (risk) in that epoch is 3.3580816862888305;
- The test performance (risk) in that epoch is 3.2370463070784026;

With plots of learning curves:

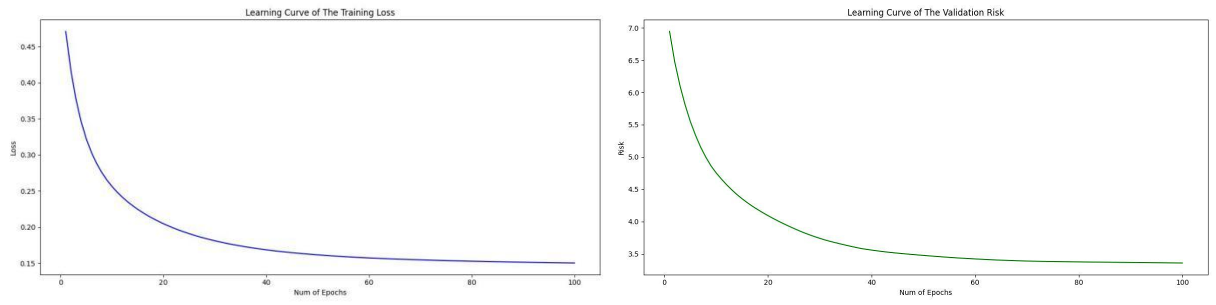

Figure 03: Learning curves of the *training loss* and the *validation risk* (from the left to the right). The number of epochs is present on the x-axis, and the corresponding *loss/risk* during the training process is shown on the y-axis.

b. [10%] We now explore non-linear features in the linear regression model. In particular, we adopt point-wise quadratic features. Report the best hyperparameter, (i.e., the one yields the best performance, and under this hyperparameter, the three numbers and two plots required in 2(a)).

Based on iteration tests, we found the best selection for the hyperparameter λ is **0.01**, with:

[*MaxIter*=100, λ=0.01, *BatchSize*=10]

- The number of epoch that yields the best validation performance is 100 (=***MaxIter***);
- The validation performance (risk) in that epoch is 3.2804253271377752;

- The test performance (risk) in that epoch is 3.1532011967821636;

With plots of learning curves:

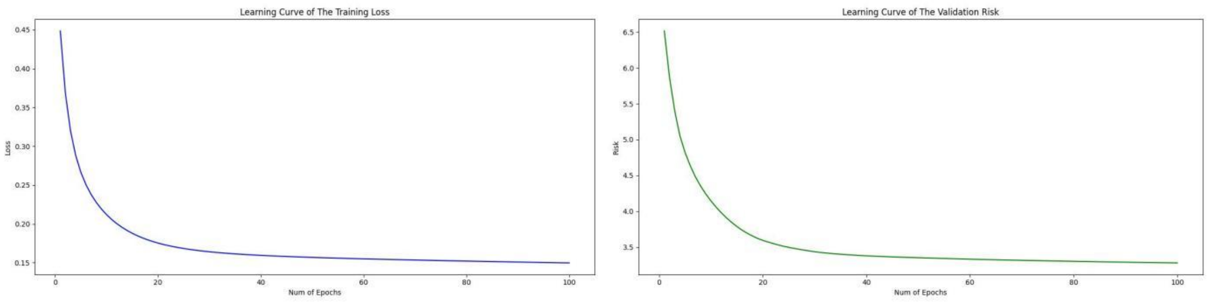

Figure 04: Learning curves of the *training loss* and *validation risk* (from the left to the right) with the known best decay term of λ=0.01. The number of epochs is present on the x-axis, and the corresponding *loss/risk* during the training process is shown on the y-axis.

c. [10%] Ask a meaningful scientific question on this task by yourself, design an experimental protocol, report experimental results, and draw a conclusion.

Question(s): "Does the growth in iteration times be proportional to the extent of error decreasing? Will a change in the number of iteration times interacts with the decay term?"

Assume the maximal iteration times is *MaxIter*=3000, with decay term of λ=0, we found:

[*MaxIter*=3000, λ=0, *BatchSize*=10]

- The number of epoch that yields the best validation performance is **2132** (<*MaxIter*);
- The validation performance (risk) in that epoch is 2.6499012905031147;
- The test performance (risk) in that epoch is 3.0608015489761082;

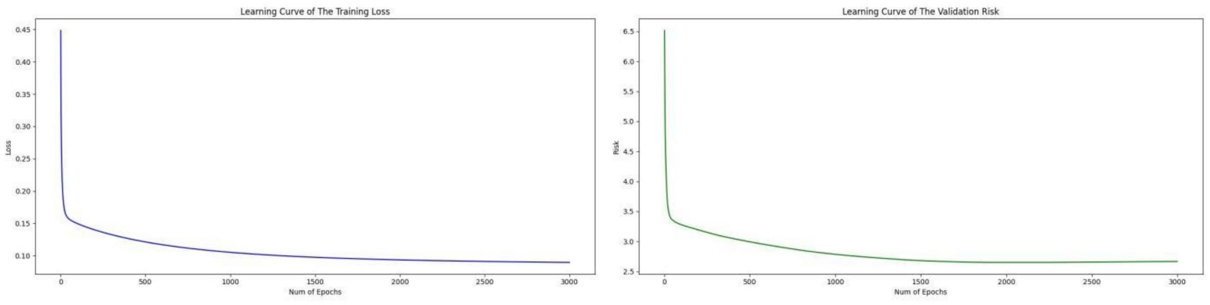

Figure 05: Learning curves of the *training loss* and the *validation risk* (from the left to the right) with *MaxIter*=3000 and no decay term applied. The number of epochs is present on the x-axis, and the corresponding *loss/risk* during the training process is shown on the y-axis.

Now we still keep the maximal iteration times to be *MaxIter*=3000, but adding the decay term of λ=0.01 (the known best decay term we found in 2(b)) and λ=1 respectively, the results tell:

[*MaxIter*=3000, λ=0.01, *BatchSize*=10]

- The number of epoch that yields the best validation performance is **3000** (=*MaxIter*);
- The validation performance (risk) in that epoch is 2.672133754945308
- The test performance (risk) in that epoch is 2.999298939376094

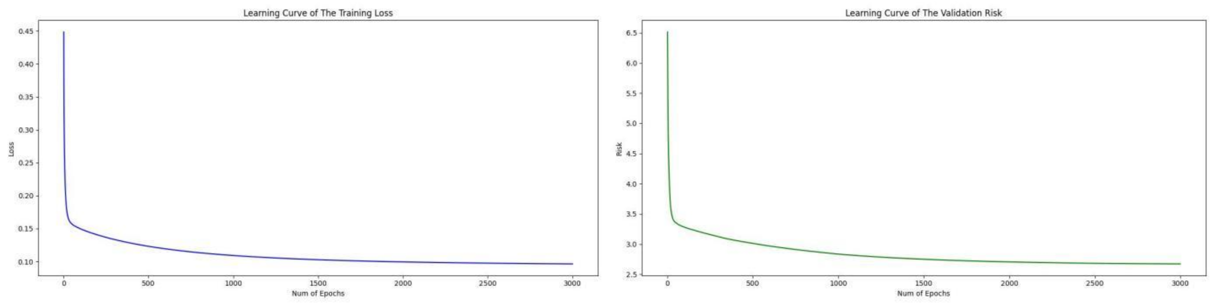

Figure 06: Learning curves of the *training loss* and the *validation risk* (from the left to the right) with *MaxIter*=3000 and a decay term of λ=0.01. The number of epochs is present on the x-axis, and the corresponding *loss/risk* during the training process is shown on the y-axis.

[*MaxIter*=3000, λ=1, *BatchSize*=10]

- The number of epoch that yields the best validation performance is **1081** (<***MaxIter***);
- The validation performance (risk) in that epoch is 3.7328505727637906;
- The test performance (risk) in that epoch is 3.1315040864706294;

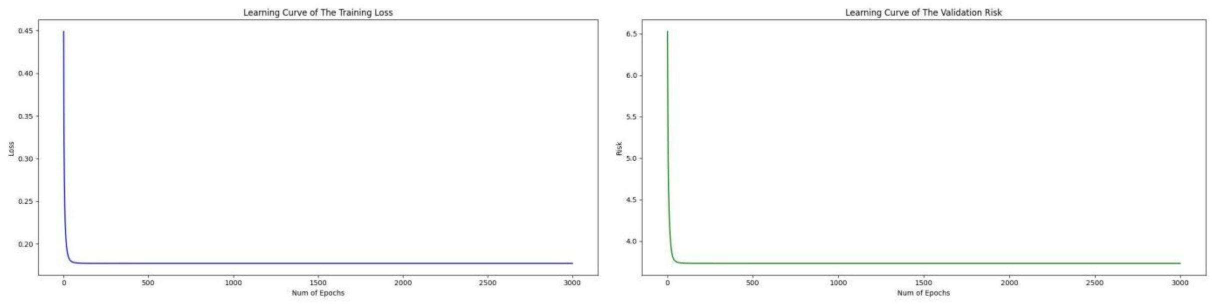

Figure 07: Learning curves of the *training loss* and the *validation risk* (from the left to the right) with *MaxIter*=3000 and a decay term of λ=1. The number of epochs is present on the x-axis, and the corresponding *loss/risk* during the training process is shown on the y-axis.

According to what the plots illustrated, we realized raising the upper bound of the maximal iteration number may improve learning performance within a specific scope (by comparing to the data acquired before). However, the growth of performance may come with an upper limit. For instance, *Figure 05* and *Figure 07* present obvious overfitting conditions: the best validation performance occurred in epoch 2132 and 1081, respectively (where the ideal *MaxIter*=3000). Even though the training loss keeps decreasing as the maximal number of epochs increases, all later epochs (after reaching the minimum) in validations perform worse. This condition may result in poor generalization of the learning results.

We can conclude the change in the upper bound of the iteration number can interact with the effects of the ℓ_2-penalize (i.e., the decay term controls). Commonly, without changing other parameter settings, the larger selection of *MaxIter* may push us to have a relatively smaller decay term to avoid over-shot issues (problem shown in *Figure 07*). However, even under a high *MaxIter* design, the decay term should not be too limited. Otherwise, the GD process may have insufficient power to go down the hill and reach the optimal point (as what happened in *Figure 05*). After a few experiments, we found an ideal scope for the decay term to balance the trade-off towards this task, the scope of (0.01, 0.1).