Bohao Yang
31799256
by2u20@soton.ac.uk

Reinforcement and Online Learning
Recursive Least Squares

University of Southampton

2021-02-21

1 Implement simple linear regression on a synthetic problem

I first generated covariates from a random number generator in a design matrix with 500 dimension multiply by 15. Then I implemented linear regression in closed form on these data, the graph of predicted and true targets are shown in the first plot in Fig.1.

Then I implemented linear regression using gradient descent and stochastic gradient descent. The graph of predicted and true targets and learning rate produced by these methods are shown in the first and third plots in Fig.2 and Fig.3 separately.

With stochastic gradient descent, choosing different iterations would produce difference error curves as at each iteration we only randomly pick one data point and update the corresponding weight. One solution for avoiding learning curve looks noisy and return a more consistent answer is lower the learning rate so that we would not exceed the optimal point at each updating process.

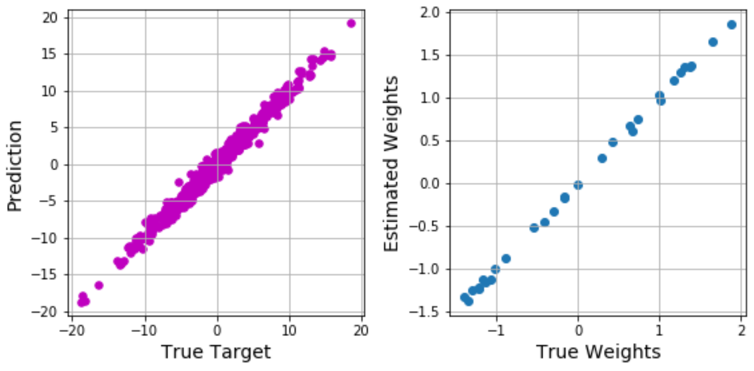


Figure 1: y and weight of closed_form linear regression

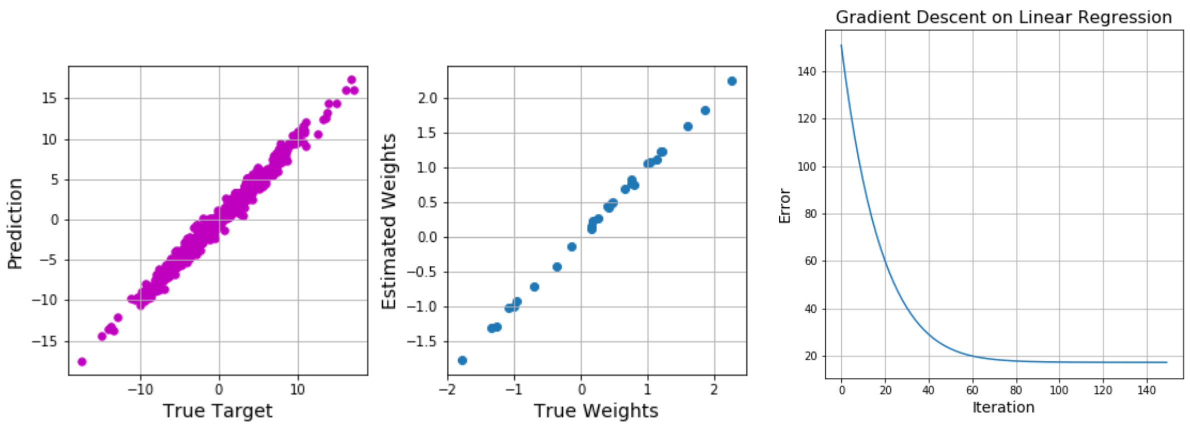


Figure 2: y, weight and learning curve of gradient descent linear regression

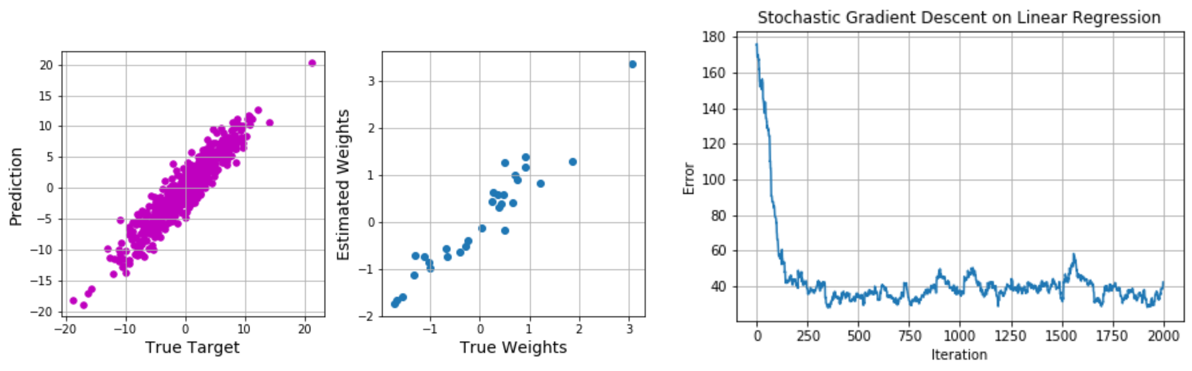


Figure 3: y, weight and learning curve of stochastic gradient descent linear regression

2 Implement my own Recursive Least Squares solution

In the second part, I implemented my own Recursive Least Squares(RLS) solution and error curve is shown in the third plot of Fig 4. The error curve would move close to zero suddenly at beginning and move around zero till the end.

As for the relationship between the dimensionality and convergence speed, I generated 5 sets of data and their dimensions are from 5 to 30 with the gap of 5 and applied RLS on these data. I found with the increase of dimension, the convergence speed would decrease. To be specific, implemented RLS on data with 5 dimension would converge faster than implementing on data with 10 or more dimensions.

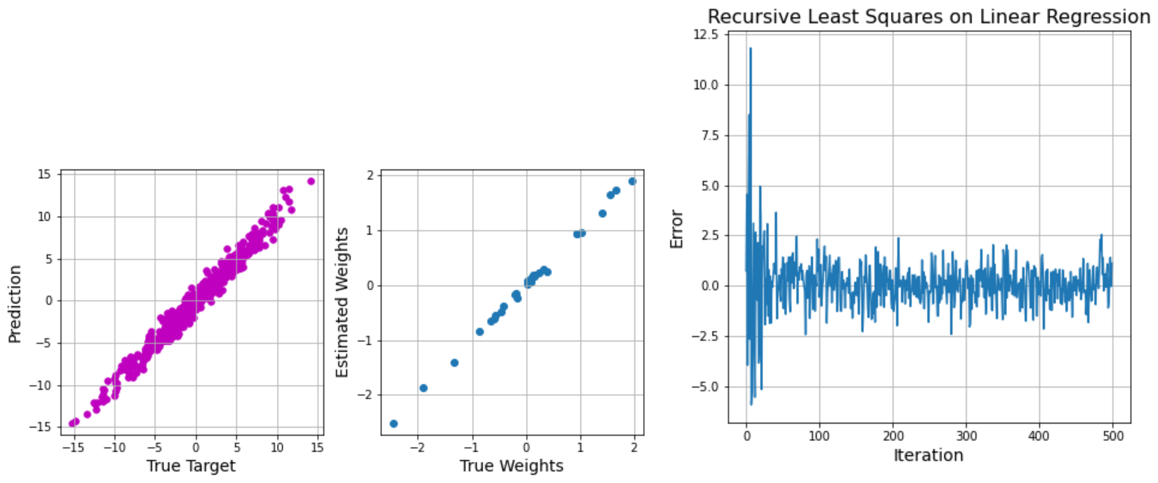


Figure 4: y, weight and learning curve of recursive least squares solution for linear regression

3 Implement my own Recursive Least Squares solution on UCI dataset

I downloaded Computer Hardware Dataset"[1] which have 209 samples with 10 dimensionality. I excluded first two and forth columns data and some outlier. Then implemented SGD with 2000 iterations and RLS on it, the error curve of RLS in the seceond plot of Fig.5 is better than that of SGD in Fig. 6 as the former one move around zero value and do not rise or drop rapidly between iterations. However the error curve of latter moves around 540 and did not get an ideal result.

[1]https://archive.ics.uci.edu/ml/datasets/Computer+Hardware

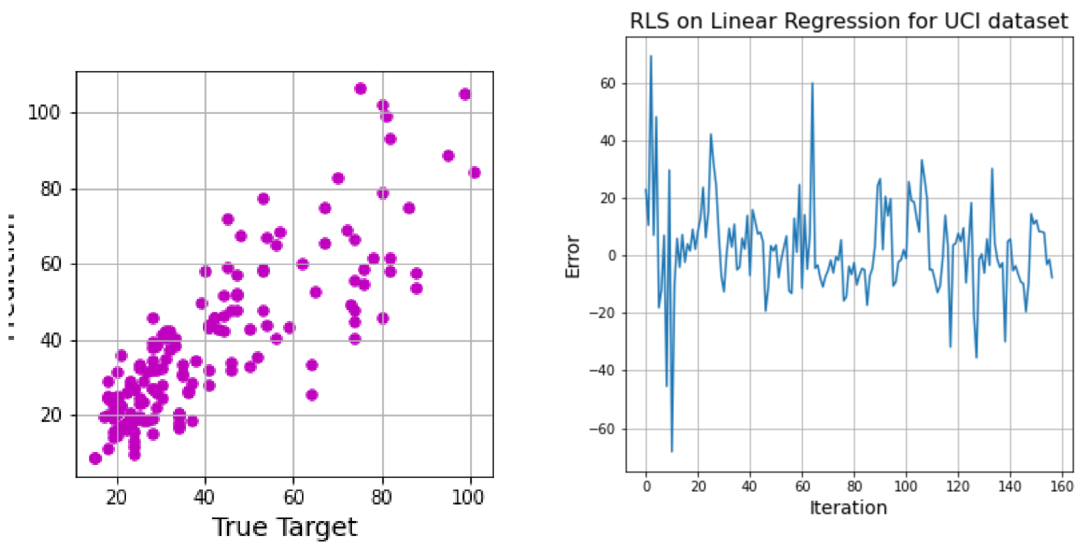


Figure 5: y and learning curve of self-implemented recursive least squares solution for linear regression on UCI dataset

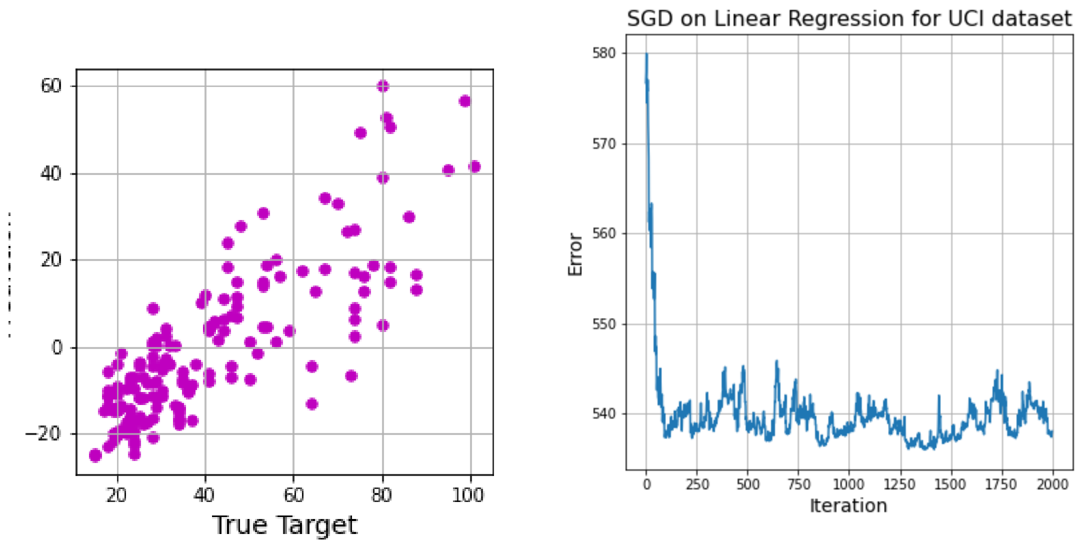
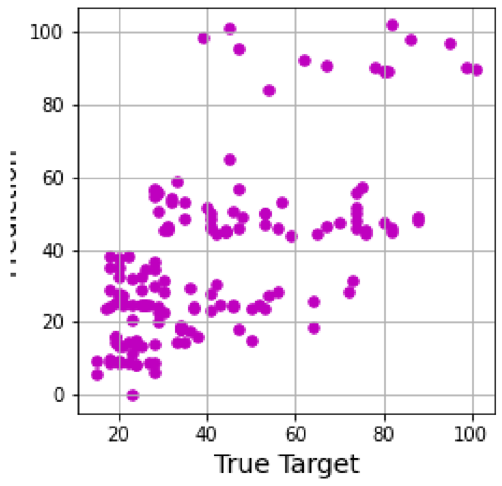


Figure 6: y and learning curve of self-implemented recursive least squares solution for linear regression on UCI dataset

4 Implement Recursive Least Squares solution from Padasip package on UCI

I implemented RLS from Padasip package on UCI dataset, the true and predicted target and error curve of can be seen in Fig 7, the error is moving from zero to 60 during the whole process, which is worse than that of RLS implemented by myself.



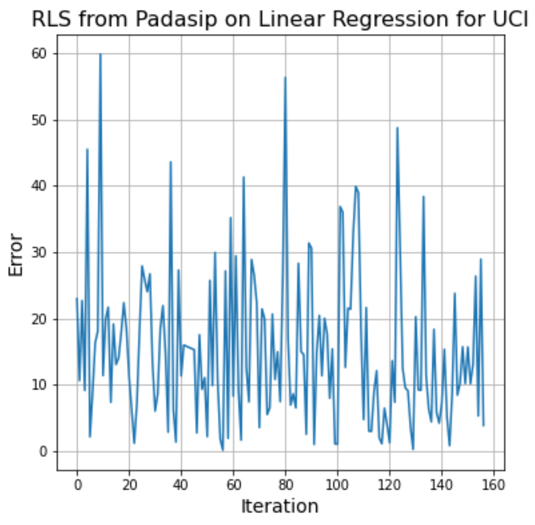


Figure 7: y and learning curve of recursive least squares solution from padasio package for linear regression on UCI dataset