University of Birmingham Lab Report

Pages 1–3
2020

University Admission Prediction Using Linear Regression

Letian Li / 2214560

LM Mathematical Foundations of Artificial Intelligence (AI) and Machine Learning (ML)

School of Computer Science / College of Engineering and Physical Sciences

11.12.2020

Keywords: Linear Regression, Standard Gradient Descent, Stochastic Gradient Descent, Mini-batch Gradient Descent

1 CONCEPT DESCRIPTION

1.1 Linear Regression Model

Linear Regression Model is a model to approximate an underlying linear function from a set of noisy data. For example, having N training samples, we then have N independent variables x and N dependent variables y. We estimate the dependent variables by multiplying the independent variables by the weights and plus the intercept. Namely, we estimate dependent variables y by linear combinations of independent variables x.

1.2 Linear Least Squares

Linear Least Squares (LLS) is a method to help us to approximate the linear regression model by minimizing the sum of the squares of the differences between the observed dependent variable y and the predicted value $\hat{y}$.

1.3 Gradient Descent

Gradient Descent is a first-order iterative optimization algorithm using to find a minimum value of cost function (aka "loss funcion"). In other words, it is used to find the optimal parameters which make the predicted $\hat{y}$ close to the observed dependent variable y as much as possible. The main idea is to take repeated steps in the opposite direction of the gradient of the cost function at the current point. Because the negative gradient is the direction of steepest descent. Generally, stepping in the direction along with the gradient will result in a maximum value of the cost function. To minimizing the cost function, we update the parameters with the opposite direction of gradient. This iterative procedure is called gradient Descent.
Generally, there are three types of gradient descent, Standard Gradient Descent, Stochastic Gradient Descent, and Mini-batch Gradient Descent. A simple way to distinguish these three gradient descent is to look at how many samples are used in one batch.

1.3.1 Standard Gradient Descent uses all of the training samples for every iteration. In LLS problems, which are convex, using standard gradient descent can always find a stationary point, which happened to be global minimum. This means we can always find the closed-form solution of this kind of optimization problem.
However, the drawbacks of standard gradient descent are also obvious. Because of using all samples in every iteration, the computer needs to load the whole dataset into memory and compute the gradient. Thus, it is not feasible for large dataset for a complex problem. In addition, it is not easy to be parallelised. And the worst thing is that using standard gradient descent make the model easy to be trapped by local minimum.

1.3.2 Stochastic Gradient Descent is just a way to solve above problem faced by standard gradient descent. It use only one data point at each iteration to calculate the cost gradient.In fact, the true gradient can be approximated by a gradient at a single example. The only thing need to do is random shuffling at each iteration and sampling cost functions of all samples without replacement. By doing so, it enables gradient descent to jump to new and potentially better local minimum. But this also make the stochastic Gradient Descent unstable convergence to the exact minimum and keep overshooting.

1.3.3 Mini-batch Gradient Descent is a trade-off method between Standard Gradient Descent and Stochastic Gradient Descent. It iterates over mini-batches in each epoch, so that the cost function, and therefore gradient, is averaged over N_{mb} samples. And N_{mb} is called the mini-batch size. Similar to stochastic gradient descent, mini-batch gradient descent algorithm also need to randomly shuffle the samples before they are chose to do the iteration.

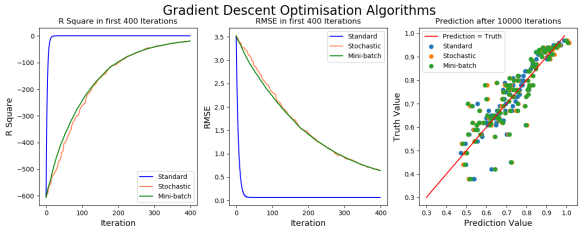

Fig. 1. Gradient Descent Optimisation Algorithms

2 PREDICTION PERFORMANCE

In the implementation of three gradient descent into Linear Regression Model, R Square and Root Mean Square Error (RMSE) are used as metrics for predictive performance. After several tests and tuning the hyperparameters, here are the best performance (shown as Figure 2) that this experiment can achieve (note that the result of the corresponding code is random for each time because the dataset is randomly divided into training set and validation set):

2.1 Standard Gradient Descent

R Square: 0.8231
Root Mean Square Error(RMSE): 0.0574
(Obtain Theta: [0.7233 0.0175 0.0188 0.0063 0.0027 0.0189 0.0687 0.0121])

2.2 Stochastic Gradient Descent

R Square: 0.8097
Root Mean Square Error(RMSE): 0.0595
(Obtain Theta: [0.7266 0.0277 0.0149 -0.0013 -0.0040 -0.0001 0.0632 0.0184])

2.3 Mini-batch Gradient Descent

R Square: 0.8141
Root Mean Square Error(RMSE): 0.0588
(Obtain Theta: [0.7151 0.0207 0.0190 0.0126 0.0013 0.0231 0.0703 0.0138])

2.4 Closed-Form Solution

Theta: [0.7233 0.0175 0.0188 0.0063 0.0027 0.0189 0.0687 0.0121]

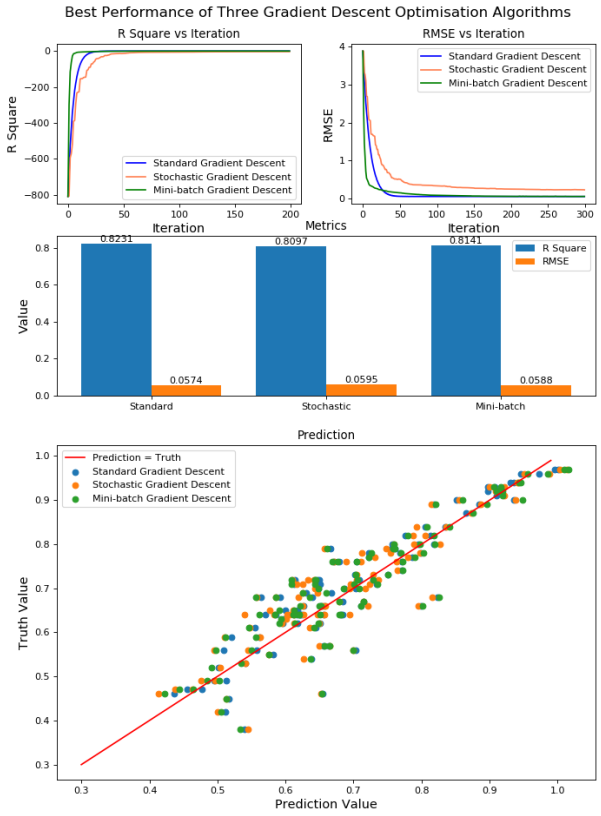

Fig. 2. Best Performance of Three Gradient Descent Optimisation Algorithms

3 ANALYSIS AND DISCUSSION

3.1 Hyperparameters

3.1.1 Learning Rate

Learning Rate is generally the most important hyperparameter in gradient descent that controls how much to update the model parameters of weights in each iteration. In order to explore the influence of learning rate on gradient descent, this experiment conducted the same several tests on the three gradient descent models. By using different learning rates, the change of model loss value is observed. Figure 3 shows the changes of metrics in three gradient descent models with the learning rate of 10^{-3} ,10^{-4} ,10^{-5} ,10^{-6} , and 10^{-7} . It can be seen that, no matter what kind of gradient descent algorithm is used, the greater the learning rate is, the faster metrics move. In the experiment, a different situation which is not shown in the figure also occurred. When the learning rate is too high, the loss of the model cannot decline and converge. On the contrary, this situation will result in exponential explosion of loss. Therefore, it can be concluded that: the smaller the learning rate is, the slower the loss will decline, which means that it will take longer to reach the local minimum. However, when the learning rate is too large, loss cannot converge and fail to find the optimal solution of the model.

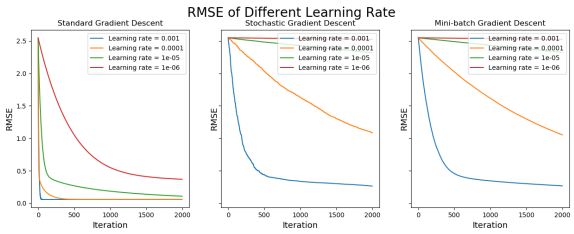

Fig. 3. RMSE of Different Learning Rate

3.1.2 Iteration

One iteration means one time for update the parameters. A similar concept is epoch, which is the value of how many times the dataset has been used to the model for searching optima. Namely, one epoch means the entire dataset is passed through the model for only once. So one epoch can have many iterations, and they are both used to represent the learning time of the model. In this experiment, the effect of the iteration on the models is illustrated by taking R square and Root Mean Square Error as the measurement index. Figure 4 shows the prediction performance of three gradient descent models with the same learning rate in each iteration. We can see that with the increase of the iteration, the three models were optimized. Among them, the standard gradient descent is optimized the fastest, while the other two gradient descent models are much slower. In addition, it can be clearly observed that the process of stochastic gradient descent is quite oscillating in each iteration.

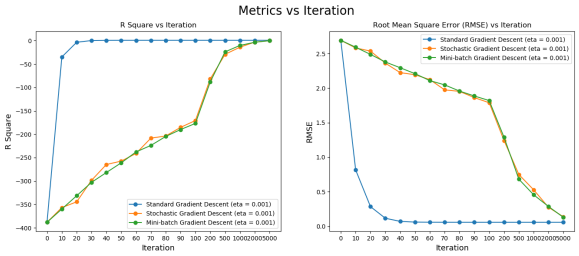

Fig. 4. Metrics vs Iteration

3.1.3 Batch Size

Batch Size is the number of training examples present in a single batch. It is used to divide dataset into number of sets for updating parameters (eg. weights and intercept) in one iteration. Batch method was only implemented in Mini-batch Gradient Descent , but for the convenience of comparison, Standard Gradient Descent and Stochastic Gradient Descent with the same parameters were added in the figure 5. When comparing batch size and iteration, it can be seen that no matter which batch size is, the RMSE value decrease at a similar speed. The standard gradient

descent is the fastest because it uses the entire dataset in one iteration. Compared to the figure of the epoch, it can be seen that the small batch size decreased faster because they iterated more times in one epoch. It is worth noting that although the batch size of 1 is not drawn in the figure, it can be inferred from the trend that when the batch size is 1, it happens to be the Stochastic Gradient Descent. This is also consistent with what we know about this two algorithms.

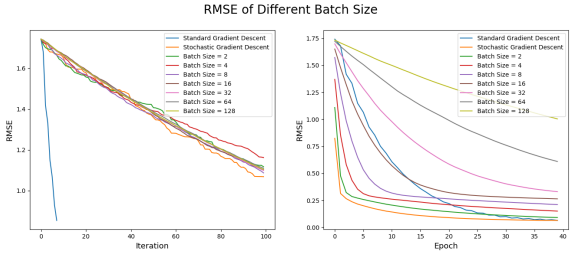

Fig. 5. RMSE of Different Batch Size

3.2 Compare to Closed-Form Solution

Because this problem is a convex LLS problem, it always have the Closed-Form Solution. After the closed solution is obtained, the weight parameters and model performance trained by the three models can be better compared. It can be found that the standard gradient descent can always reach the optimal solution under enough iterations. Stochastic gradient descent, although the decline is large, but it is easy to lock down to a certain extent and no longer decline, and violent repeated fluctuations. The performance and degradation rate of Mini Batch are in between the first two. Although it is difficult to achieve the optimal solution, it still can be reduced to an acceptable level, which gives a good performance.

3.3 Normalization and Learning Rate

A highlight finding in this experiment is the role of normalization. Prior to normalization, all three gradient descent exhibited extremely slow rates of descent that were almost impossible to reach the local minimum. As far as I am concerned, the reasons are as follows: because there is no normalization, the parameters of weights and intercept are too sensitive to the training data, so the learning rate must be adjusted smaller to make its aggregation, otherwise, the loss will increase in reverse. However, if the learning rate is set too low, the gradient descent will be too slow. This problem is solved when data is normalized.

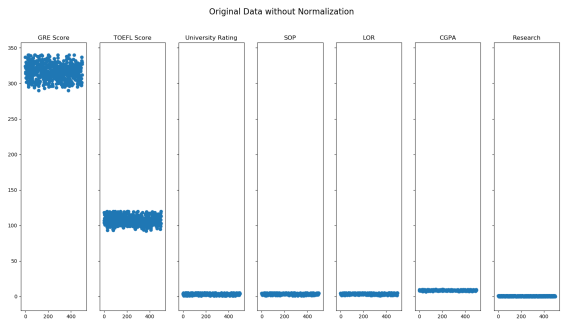

Fig. 6. Original Data without Normalization

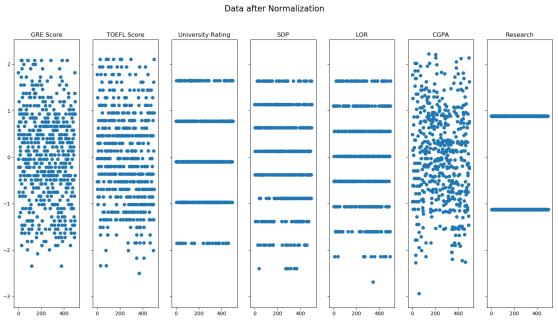

Fig. 7. Data after Normalization

3.4 Conclusion

Through the experiment of three kinds of gradient descent algorithms, it can be realized that different optimization methods, hyperparameter settings and normalization processing all have great influence on the linear regression model. It must be admitted that in uncomplicated linear problems, Standard Gradient Descent is an efficient and accurate way to find the optimal solution. However, due to its demand for storage space, it becomes impractical in the face of problems with huge number of parameters. In this case, Mini-batch Gradient Descent is an excellent choice. Its performance is close to the Standard Gradient Descent, and its consumption of resources is relatively controllable. Regarding hyperparameters, it can be seen in this experiment that the learning rate has the most critical impact on the speed of reaching the local minimum. It is also worth mentioning the importance of normalization, which should be the first to be thought of every time a machine learning task is established.