华南理工大学

South China University of Technology

The Experiment Report of Machine Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Haishan Ao

Supervisor:
Qingyao Wu

Student ID：
201720145051

Grade:
Graduate

December 13, 2017

Linear Regression, Linear Classification and Gradient Descent

Abstract—This experiment implements linear regression and linear classification by using gradient descent.

I. Introduction

THIS experiment implements linear regression and linear classification by using gradient descent. The experiment helps us further understand of linear regression and gradient descent. At this time, we conduct some experiments under small scale dataset. What's more, we realize the process of optimization and adjusting parameters.

II. Methods and Theory

A. Linear Regression

The linear regression problem can be simply described as follows. Given *m* training samples $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ and the corresponding target $y^{(1)}, y^{(2)}, \dots, y^{(m)}$, the regression equation $h_W(x)$ can be obtained, which make it possible to predict the target *y* corresponding to a new sample *x* accordingly. The regression equation is commonly defined as:

$$h_w\left(x^{(i)}\right) = w_0 + w_1 x_1 + \cdots + w_n x_n = \sum_{i=0}^{n} w_n x_n$$

where $x_0 = 1$; x_j is the *j-th* feature of sample *x*, j = 1,2, … , n; *w* is the regression parameter. $x^{(i)}$ is the *i-th* sample(i = 1,2, … , m), which is a column vector contains *j* features, i.e. $x^{(i)} = \left[x_1^{(i)}, x_2^{(i)}, \dots, x_m^{(i)}\right]^T$.

We can also use vector to describe $h_W(x)$ as:

$$h_w\left(x^{(i)}\right) = w^T x_j = W^T X$$

where $X = [1, x_1, \dots, x_n]^T$;W= $[w_0, w_1, \dots, w_n]^T$.

Then we have the loss function:

$$J(w) = \frac{1}{2m}\sum_{i=0}^{m}\left(h_w\left(x^{(i)}\right) - y^{(i)}\right)^2$$

To minimize the loss, we use gradient descent method. First, compute gradient of loss with respect to parameters $\frac{\partial J(w)}{\partial w}$. The derivation of the loss function is:

$$\frac{\partial J(w)}{\partial w} = \frac{1}{m}\sum_{i=0}^{m}\left(h_w\left(x^{(i)}\right) - y^{(i)}\right)x^{(i)}$$

Then update parameters:

$$w' \leftarrow w - \eta\frac{\partial J(w)}{\partial w}$$

where η is learning rate, a hyper-parameter that we can adjust.

B. Linear Classification

We use support vector machine (SVM) to implement linear classification. We choose normalization such that $w^T x_+ + b = +1$ and $w^T x_- + b = -1$ for the positive and negative support vectors respectively. Then the margin is given by

$$\frac{w}{\| w \|}\cdot(x_+ - x_-) = \frac{w^T(x_+ - x_-)}{\| w \|} = \frac{2}{\| w \|}$$

Learning the SVM can be formulated as an optimization:

$$\min_{w,b} L: \frac{\lambda \| w \|^2}{2} + \frac{C}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(w^T x_i + b)\right)$$

where λ is the regularization parameter.

Then the derivation of the loss function is:

$$\nabla_w L(w,b) = \begin{cases} \lambda w \ \ if\ 1 - y_i(w^T x_i + b) < 0 \\ \lambda w + \dfrac{C}{n}\displaystyle\sum_{i=1}^{n}(-y_i x_i) \ \ otherwise \end{cases}$$

$$\nabla_b L(w,b) = \begin{cases} 0 \quad if\ 1 - y_i(w^T x_i + b) < 0 \\ \dfrac{C}{n}\displaystyle\sum_{i=1}^{n}(-y_i) \quad otherwise \end{cases}$$

Update the parameters:

$$w' \leftarrow w - \eta\nabla_w L(w,b)$$
$$b' \leftarrow b - \eta\nabla_b L(w,b)$$

where η is learning rate, a hyper-parameter that we can adjust.

III. Experiment

A. Dataset

Linear regression uses Housing in LIBSVM Data, including 506 samples and each sample has 13 features. We divide it into training set and validation set(7:3).

Linear classification uses australian in LIBSVM Data, including 690 samples and each sample has 14 features. We divide it into training set and validation set(7:3).

B. Implementation

The experiment process as follows:

Step1: Load the experiment data and divide the dataset into training set and validation set.

Step2: Initialize linear model parameters. Set all parameter into zero, initialize it randomly or with normal distribution.

Step3: Define the loss function of the linear regression to be Least squared loss, and the loss function of the linear classification to be Hinge loss.

Step4: Compute the gradient of the loss function with respect to the weight W and bias b.

Step5: Update the parameters W and b.

Step6: Repeat above steps for several times until convergence.

We chose 5 different learning rates [0.003,0.05,0.1,0.2,0.4] to optimize the linear regression algorithm, the results are as the following：

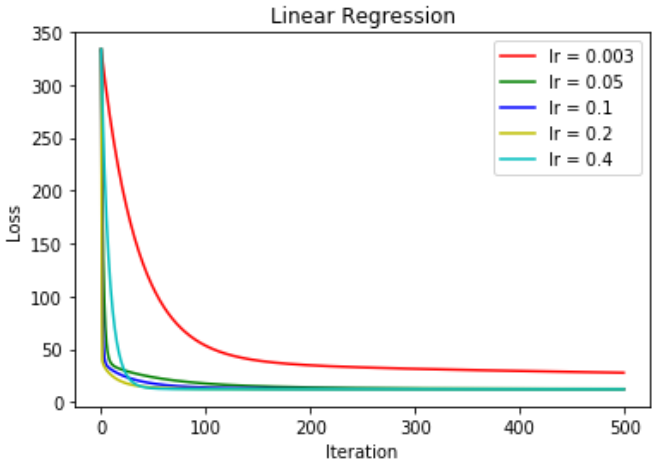

Fig. 1. Different learning rates has impacts on convergence

If the learning rate is too small, it will be too slow to converge. If the learning rate is too large, the convergence will be oscillatory and may even diverge.

In linear regression, we train the dataset with 500 epochs with learning rate η = 0.2, and we get the loss L_{train} under the training set and $L_{\text{validation}}$ by validating under validation set, is shown as Fig. 2.

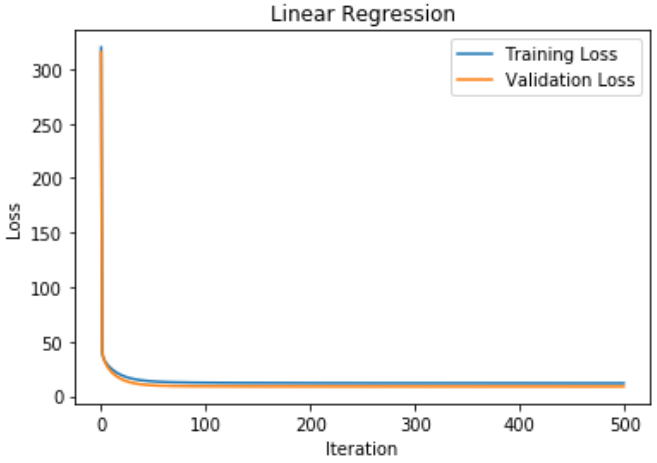

Fig.2. L_{train} and $L_{\text{validation}}$ of linear regression

We chose 5 different learning rates [0.01,0.05,0.1,0.2,0.3] to optimize the linear classification algorithm, the results are shown as Fig.3.

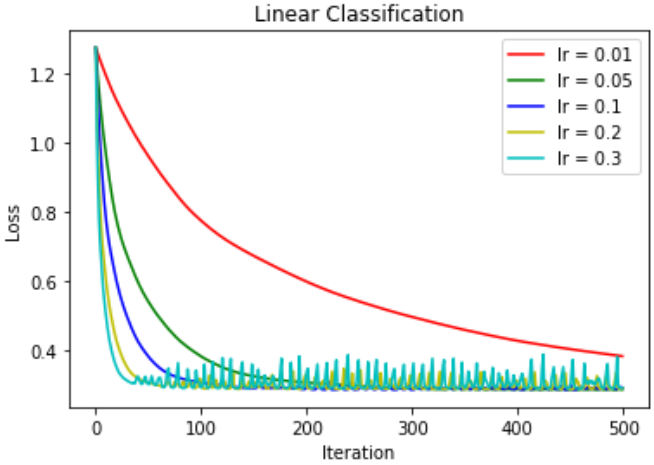

Fig. 3. Different learning rates has impacts on convergence

If the learning rate is too small, the update of the parameter W will be small. The decline of the curve is slow and the number of the iteration to reach convergence will be large. If the learning rate is too large, the loss curve will be oscillating.

We chose 5 different regularization parameter [0.01, 0.05, 0.1, 0.25, 0.5] to optimize the linear classification algorithm, the results are shown as Fig.4.

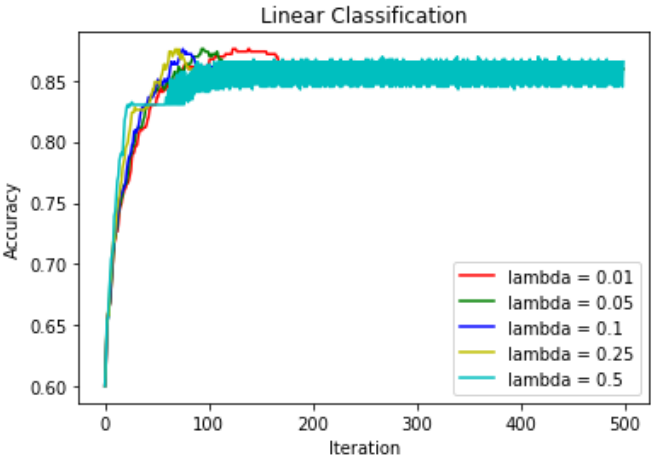

Fig. 4. The impact of different λ on accuracy in linear classification

If the λ is too small, the proportion of the regularization term will be small and the model may be easy to fall into over-fitting. If the λ is too large, the proportion of the error term will be small and the model may be under-fitting.

In linear classification, we train the dataset with 500 epochs with learning rate η = 0.1, regularization parameter $\lambda = 0.1$, threshold, and we get the loss L_{train} under the training set and $L_{\text{validation}}$ by validating under validation set, and accuracy is shown as Fig. 5.

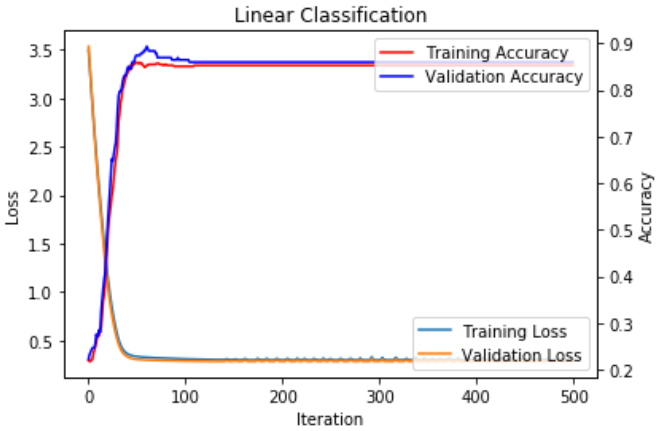

Fig.5. L_{train} , $L_{\text{validation}}$ and accuracy of linear regression

IV. Conclusion

Gradient decent is a valid method to optimize both regression problem and classification problem. At the beginning of optimization, the loss decrease quickly and the accuracy of model increase sharply. With epoch goes by, the model verges to be optimized. The greater the learning rate is, the quicker the loss decreases. But large learning rate may lead to vibrating. Different metrics are used to evaluate different model. In this experiment, we compare the validation loss to evaluate the linear regression. For linear classification, we can also evaluate the model by calculating the accuracy.