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:
Jiaqi Zhong

Supervisor:
Qingyao Wu

Student ID：
201720144948

Grade:
Graduate

December 9, 2017

Linear Regression, Linear Classification and Gradient Descent

Abstract—This experiment of machine learning is about linear regression, linear classification and gradient descent, and the experiment is carried out on a small scale data set. The purpose of the experiment is to give us a better understanding of the principles of linear regression and gradient descent and to understand the process of optimization and tuning.

I. Introduction

There are two parts to this experiment of machine learning. The first part of the experiment was linear regression and gradient descent. In this section, I used the linear regression model to conduct experiments on the Housing Data set in LIBSVM Data. The second part was linear classification and gradient descent. In this section, I experimented on the australian Data set in LIBSVM Data with support vector machine model.

II. Methods and Theory

A. Linear regression

In statistics, linear regression is a linear approach for modeling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables) denoted X. The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called multiple linear regression.(This term is distinct from multivariate linear regression, where multiple correlated dependent variables are predicted, rather than a single scalar variable.)

In linear regression, the relationships are modeled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called linear models. Most commonly, the conditional mean of y given the value of X is assumed to be an affine function of X; less commonly, the median or some other quantile of the conditional distribution of y given X is expressed as a linear function of X. Like all forms of regression analysis, linear regression focuses on the conditional probability distribution of y given X, rather than on the joint probability distribution of y and X, which is the domain of multivariate analysis.

Linear regression was the first type of regression analysis to be studied rigorously, and to be used extensively in practical applications. This is because models which depend linearly on their unknown parameters are easier to fit than models which are non-linearly related to their parameters and because the statistical properties of the resulting estimators are easier to determine.

Multiple linear regression is a generalization of linear regression by considering more than one independent variable, and a special case of general linear models formed by restricting the number of dependent variables to one. The basic model for linear regression is

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip} + \epsilon_i.$$

To define the supervised learning problem more formally, given a training set, the aim is to learn a function h so that h(x) is a predictor for the corresponding value of y. This function h is called a hypothesis.

Here we can define a multivariate linear regression model:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \theta^T x$$

And the loss function is:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Its matrix is expressed as:

$$J(\theta) = \frac{1}{2m}(X\theta - y)^T(X\theta - y)$$

B. Gradient descent

Gradient descent is an algorithm that is used to minimize a function. Gradient descent is used not only in linear regression; it is a more general algorithm.

We will now learn how gradient descent algorithm is used to minimize some arbitrary function f and, later on, we will apply it to a cost function to determine its minimum.

First of all, it is clear that the means to repeatedly adjust θ is that the prediction J(θ) is small enough and that the prediction accuracy is high enough. In linear regression, the gradient descent is usually used to adjust θ :

$$\theta_j = \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$$

To run gradient descent on error function, we first need to compute its gradient. The gradient will act like a compass and always point us downhill. To compute it, we will need to differentiate our error function.we will need to compute a partial derivative. These derivatives work out to be:

$$\theta_j = \theta_j + \alpha\frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)}$$

The matrix (vector) of this function is expressed as follows:

$$\theta_j = \theta_j + \alpha\frac{1}{m}(y - X\theta)^T x_j$$

C. SVM

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods

such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

the hypothesis function is:

$$f(x) = w^T x + b$$

And the loss function is:

$$f(\vec{w}, b) = \left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(w \cdot x_i + b)\right)\right] + \lambda\|w\|^2$$

III. EXPERIMENT

A. Dataset

Linear Regression uses Housing in LIBSVM Data, including 506 samples and each sample has 13 features. You are expected to download scaled edition. After downloading, you are supposed to divide it into training set, validation set.

Linear classification uses australian in LIBSVM Data, including 690 samples and each sample has 14 features. You are expected to download scaled edition. After downloading, you are supposed to divide it into training set, validation set.

B. Implementation

Linear Regression and Gradient Descent:

(1)Load the experiment data. You can use load_svmlight_file function in sklearn library.

(2)Devide dataset. You should divide dataset into training set and validation set using train_test_split function. Test set is not required in this experiment.

(3)Initialize linear model parameters. I chosen to set all parameter into zero.

(4)Choose loss function and derivation: Find more detail in PPT.

(5)Calculate gradient G toward loss function from all samples.

(6)Denote the opposite direction of gradient G as D.

(7)Update model: Wt=Wt-1+ η D, η is learning rate, a hyper-parameter that we can adjust.

(8)Get the loss L*train* under the training set and L*validation* by validating under validation set.

(9)Repeate step 5 to 8 for several times, and drawing graph of L*train* as well as L*validation* with the number of iterations.

The experimental results of different learning rates are shown in Fig. 1, Fig. 2, Fig. 3 and Fig. 4.

From this figures, we can see that the algorithm works better when the learning rate is 0.4.

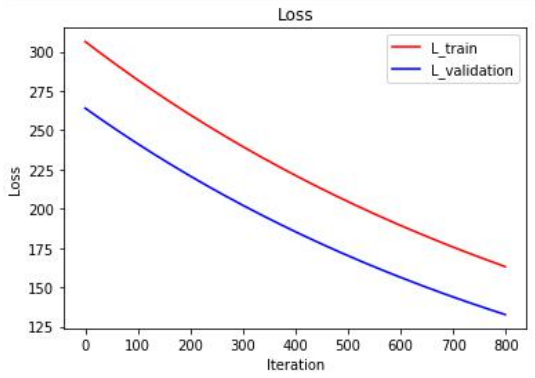

Fig. 1. the result of learning rate = 0.0001

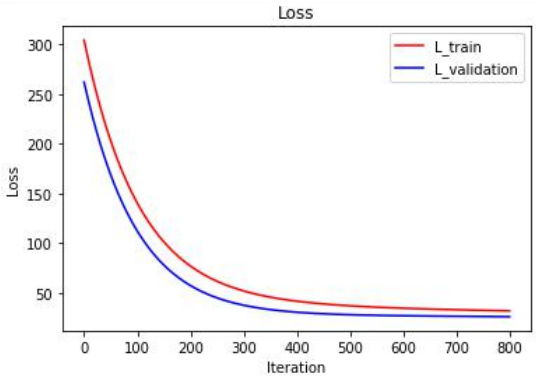

Fig. 2. the result of learning rate = 0.001

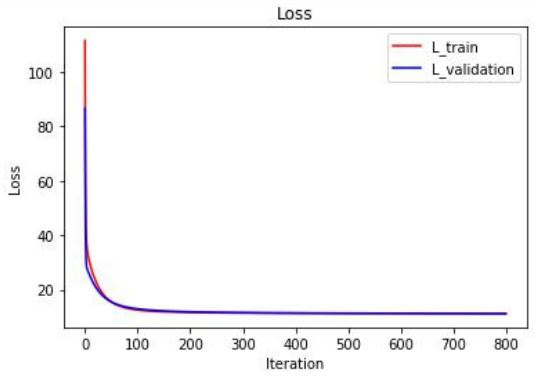

Fig. 3. the result of learning rate = 0.1

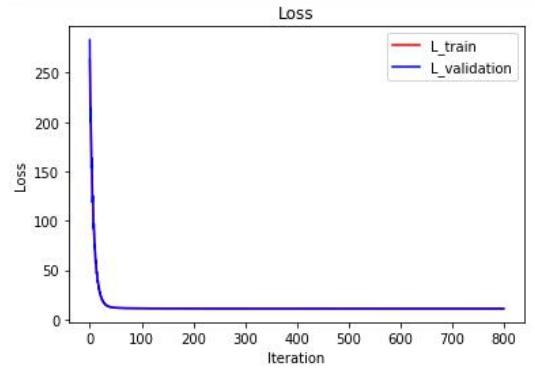

Fig. 4. the result of learning rate = 0.4

Linear Classification and Gradient Descent:

(1)Load the experiment data.

(2)Divide dataset into training set and validation set.

(3)Initialize linear model parameters. I chosen to set all

parameter into zero.

(4)Choose loss function and derivation: Find more detail in PPT.

(5)Calculate gradient G toward loss function from all samples.

(6)Denote the opposite direction of gradient G as D.

(7)Update model: Wt=Wt-1+ η D, η is learning rate, a hyper-parameter that we can adjust.

(8)I selected the appropriate threshold is 0, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative. Get the loss L_{train} under the training set and $L_{validation}$ by validating under validation set.

(9)Repeate step 5 to 8 for several times, and drawing graph of L_{train} as well as $L_{validation}$ with the number of iterations.

The experimental results of different learning rates are shown in Fig. 5, Fig. 6 and Fig. 7. From this figures, we can see that the algorithm works better when the learning rate is 0.1.

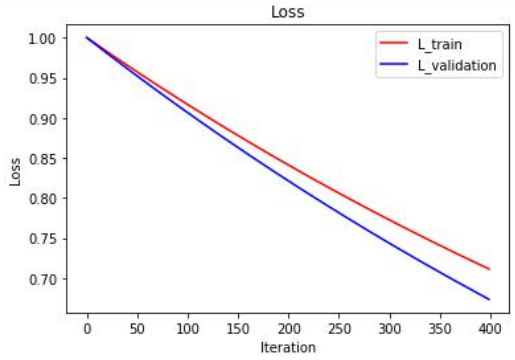

Fig. 5(a). the loss of learning rate = 0.001

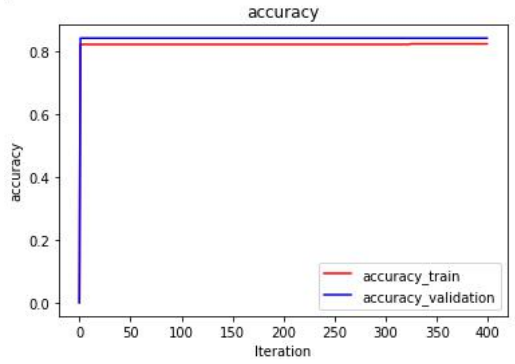

Fig. 5(b). the accuracy of learning rate = 0.001

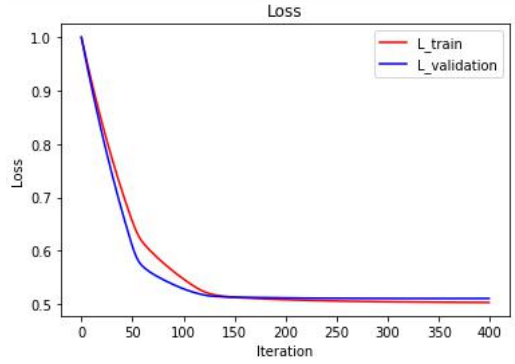

Fig. 6(a). the loss of learning rate = 0.01

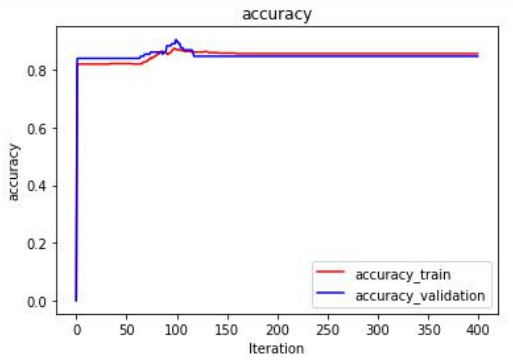

Fig. 6(b). the accuracy of learning rate = 0.01

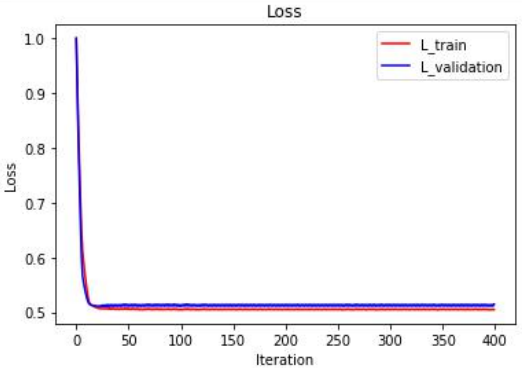

Fig. 7(a). the loss of learning rate = 0.1

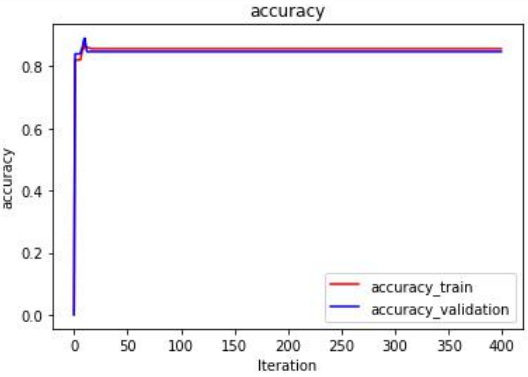

Fig. 7(b). the accuracy of learning rate = 0.1

IV. CONCLUSION

In this experiment, a total of two small experiments are included, one is linear regression and gradient descent; the other is linear classification and gradient descent. The former uses a linear regression model; the latter uses a support vector machine model. After this experiment, I have further understood of linear regression and gradient descent; conducted some experiments under small scale dataset; and realize the process of optimization and adjusting parameters.