

**The Experiment Report of**

***Deep Learning***

|  |  |
| --- | --- |
| **College** | **Software College** |
| **Subject** | **Software Engineering** |
| **Members** | YangyangZhao |
| **Student ID** | 201710106598 |
| **E-mail** | 365538442@qq.com |
| **Tutor** | MingkuiTan |
| **Date submitted** | 2017.12.06 |

**1. Topic:** RegressionExperiment and ClassificationExperiment

**2. Time:** 2017.12.06

**3. Reporter:**YangyangZhao

**4. Purposes:**

Further understand of linear regression and gradient descent.

Conduct some experiments under small scale dataset.

Realize the process of optimization and adjusting parameters.

**5. Data sets and data analysis:**

Linear Regression uses [Housing](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/regression.html#housing) in [LIBSVM Data](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/), including 506 samples and each sample has 13 features.

Linear classification uses [australian](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html" \l "australian" \t "_blank) in [LIBSVM Data](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/), including 690 samples and each sample has 14 features.

**6. Experimental steps:**

**Linear Regression and Gradient Descent**

1.Load the experiment data. You can use [load\_svmlight\_file](http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_svmlight_file.html" \t "_blank) function in sklearn library.

2.Devide dataset. You should divide dataset into training set and validation set using [train\_test\_split](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html" \t "_blank) function. Test set is not required in this experiment.

3. Initialize linear model parameters. You can choose to set all parameter into zero, initialize it randomly or with normal distribution.

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

8.Get the loss   under the training set and  by validating under validation set.

9.Repeate step 5 to 8 for several times, and drawing graph of  as well as  with the number of iterations.

**Linear Classification and Gradient Descent**

1.Load the experiment data.

2.Divide dataset into training set and validation set.

3.Initialize SVM model parameters. You can choose to set all parameter into zero, initialize it randomly or with normal distribution.

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

8.Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative. Get the loss  under the trainin set and  by validating under validation set.

9.Repeate step 5 to 8 for several times, and drawing graph of   as well as  with the number of iterations.

**7. Code:**

(Fill in the contents of 8-12 respectively for linear regression and linear classification)

**8. Selection of validation (hold-out, cross-validation, k-folds cross-validation, etc.):**

cross-validation

**9. The initialization method of model parameters:**

1)#Devide dataset. and train data size is 70%,the test data size is 30%

a\_train, a\_test, b\_train, b\_test = train\_test\_split(house\_scale[0], house\_scale[1], train\_size=0.7, test\_size=0.3)

2)#Initialize linear model parameters.weight matrix and bias term------w and b

w = np.random.randn(a\_train.shape[1])

b = np.random.randn()

3) # the number of iteration

iteration\_times = 200

# learning rate, gradient descent

n = 1

**10. The selected loss function and its derivatives:**

**11. Experimental results and curve:**

## Hyper-parameter selection (η, epoch, etc.):

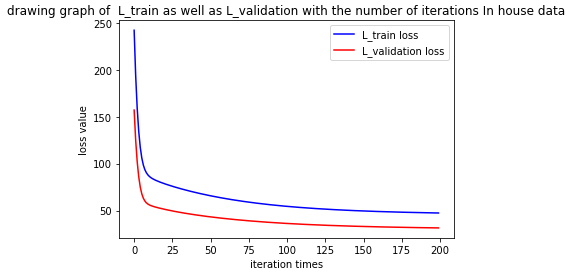
## Assessment Results (based on selected validation):

## Predicted Results (Best Results):

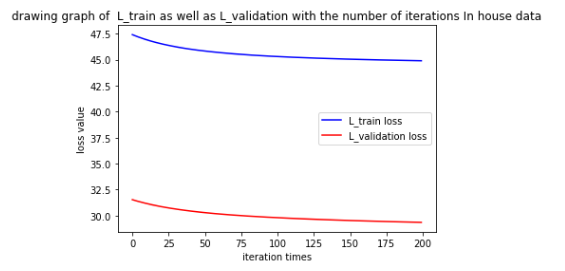
## Loss curve:

**1.house data**

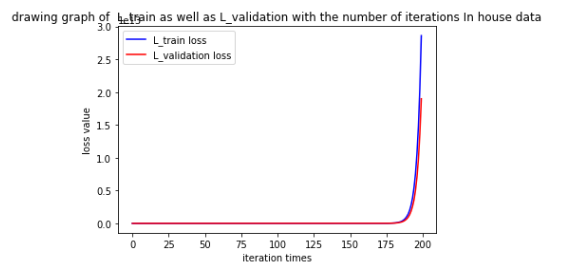
n = 0.05



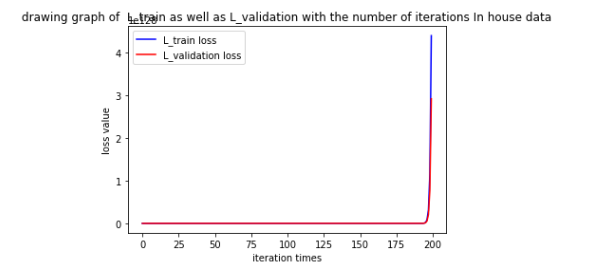
n = 0.1



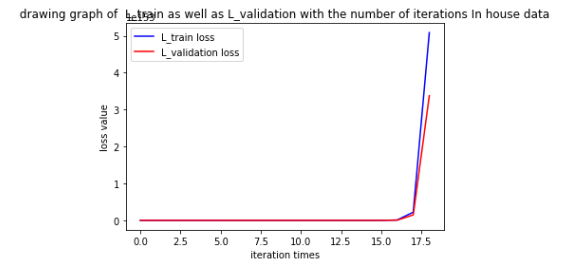
n = 0.5



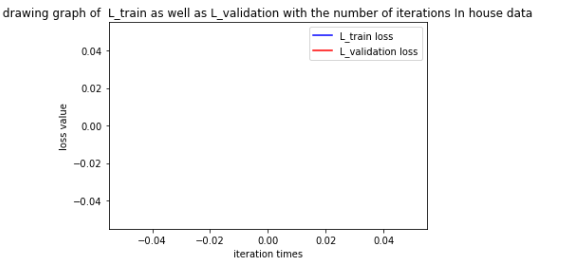
n = 1



n = 5

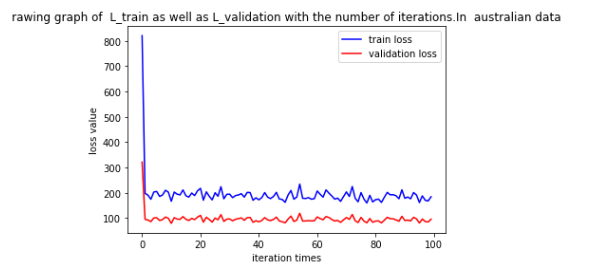


n = 10

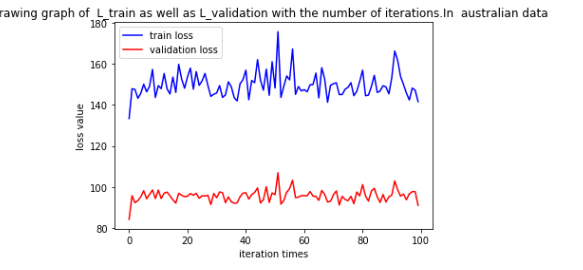


**2.australian data**

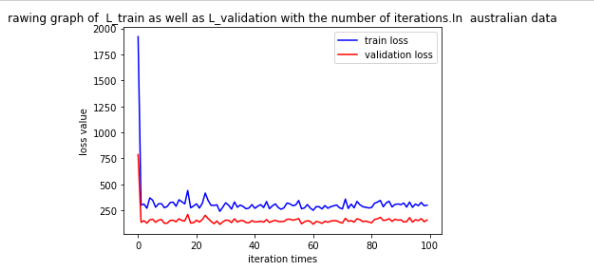
lamb= 0.05



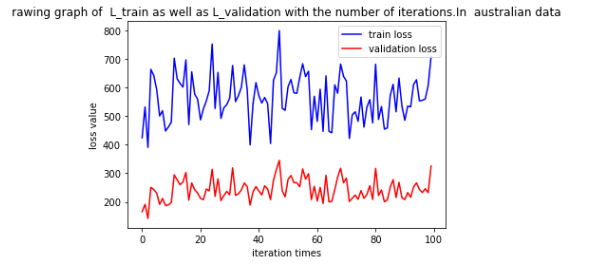
lamb = 0.1



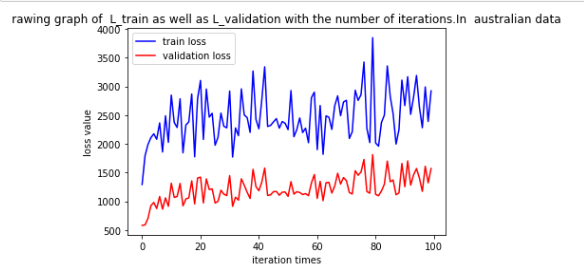
lamb = 0.5



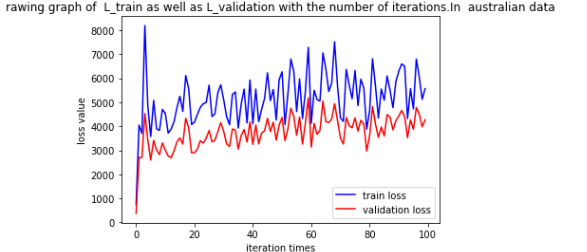
lamb = 1



lamb = 5



lamb = 10

****

**12. Results analysis:**

Linear regression: the loss function of the training set and test set decreases with the increase of iteration times in n less than a certain range.When n is greater than a certain range, the loss function of the training set and test set increases with the number of iterations.

Linear classification: at the time of lamb=0.05 and 0.5, the loss function can be reduced and then fluctuated within a certain range as the number of iterations reaches a certain value.Other cases fail to see the pattern.

**13. Similarities and differences between linear regression and linear classification:**

the two problems are essentially consistent, which is the fitting of the model.But the y value of the classification problem is a little bit more discrete, and the same y value might correspond to a bunch of x's, which have a certain range.

So the classification problem is more of a certain region of x that corresponds to a y. And the regression model is more likely to be an x in a very small area, or an x that corresponds to a y

**14. Summary:**

There are several steps to both types of problems,

How to select a reasonable model (linear, or non-linear e.g. step function, gaussian.

Create a "nice" error function you can evaluate the fitting degree, or the function of the function

The best model parameters are obtained by using all possible techniques e.g. Derivative descent method, solution of extremum equation method

In general, the two problems are essentially consistent, which is the fitting of the model.But the y value of the classification problem is a little bit more discrete, and the same y value might correspond to a bunch of x's, which have a certain range.

So the classification problem is more of a certain region of x that corresponds to a y. And the regression model is more likely to be an x in a very small area, or an x that corresponds to a y