

**The Experiment Report of**

***Machine Learning***

**College Software College**

**Subject Software Engineering**

**Members**   **黎浩坤**

**Student ID 201530611883**

**E-mail 524020165@qq.com**

**Tutor**   **谭明奎**

**Date submitted** **2017.12.8**

**1. Topic:** Linear Regression, Linear Classification and Gradient Descent

**2. Time:** 2017.12.2

**3. Reporter:**黎浩坤

**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. You are expected to download scaled edition. After downloading, you are supposed to divide it into training set, validation set.

Linear classification uses [australian](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html#australian) in [LIBSVM](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html) 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.

**6. Experimental steps:**

The experimental code and drawing are completed on jupyter.

*Linear Regression and Gradient Descent*

1. Load the experiment data.
2. Decide dataset into training set and validation set.
3. Initialize linear model parameters randomly.
4. Choose loss function and derivation.
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*.
8. Get the loss *Ltrain* under the training set and *Lvalidation* by validating under validation set.
9. Repeat step 5 to 8 for several times.
10. Drawing graph of *Ltrain* as well as *Lvalidation*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 randomly.
4. Choose loss function and derivation.
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*.
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 *Ltrain* under the training set and *Lvalidation* by validating under validation set.
9. Repeat step 5 to 8 for several times.
10. Drawing graph of *Ltrain* as well as *Lvalidation*with the number of iterations.

**7. Code:**

***Linear Regression Experiment:***

G = (1/m\_train)\*((-X\_train.T \* y\_train) + (X\_train.T \* X\_train \* w))

#Step5:Calculate gradient G toward loss function from all samples.

D = -G #Step6:Denote the opposite direction of gradient G as D.

learn\_rate = 0.05 #Learning rate

w = w + learn\_rate\*D #Step7:Update model parameters.

*Linear Classification Experiment:*

for n in range(0,m\_train):

result = 1 - y\_train[n]\*(w.T\*x\_train[n].T+b)

if(result>=0):

G\_w = G\_w + (w-C\*x\_train[n].T\*y\_train[n])

G\_b = G\_b + (-C\*y\_train[n])

elif(result<0):

G\_w = G\_w + w

G\_b = G\_b

#Step5:Calculate gradient G toward loss function from all samples.

D\_w = -G\_w

D\_b = -G\_b

#Step6:Denote the opposite direction of gradient G as D.

learn\_rate = 0.000000001 #Learning rate

w = w + learn\_rate\*D\_w

b = b + learn\_rate\*D\_b

#Step7:Update model parameters.

**8. Selection of validation:**

Linear Regression: hold-out

Linear Classification: hold-out

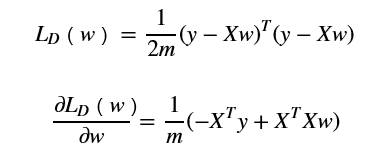
**9. The initialization method of model parameters:**

Linear Regression: randomly

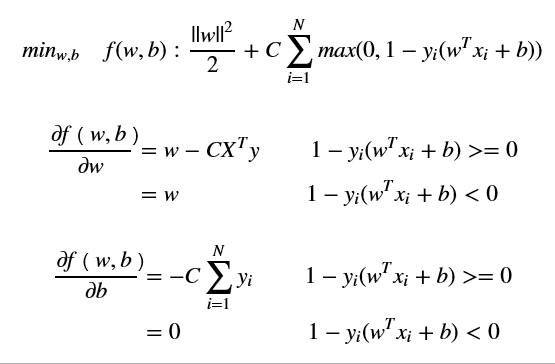
Linear Classification: randomly

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

Linear Regression:



Linear Classification:

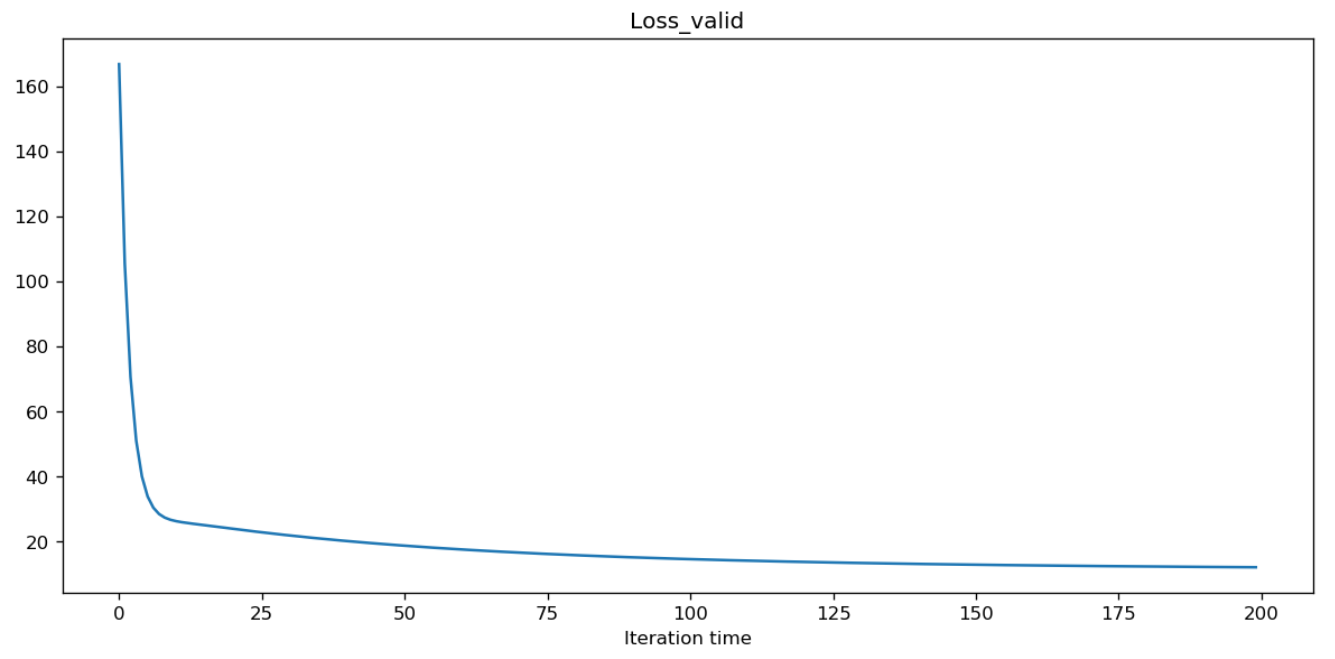


**11. Experimental results and curve:**

*Linear Regression:*

## Hyper-parameter selection: η = 0.05

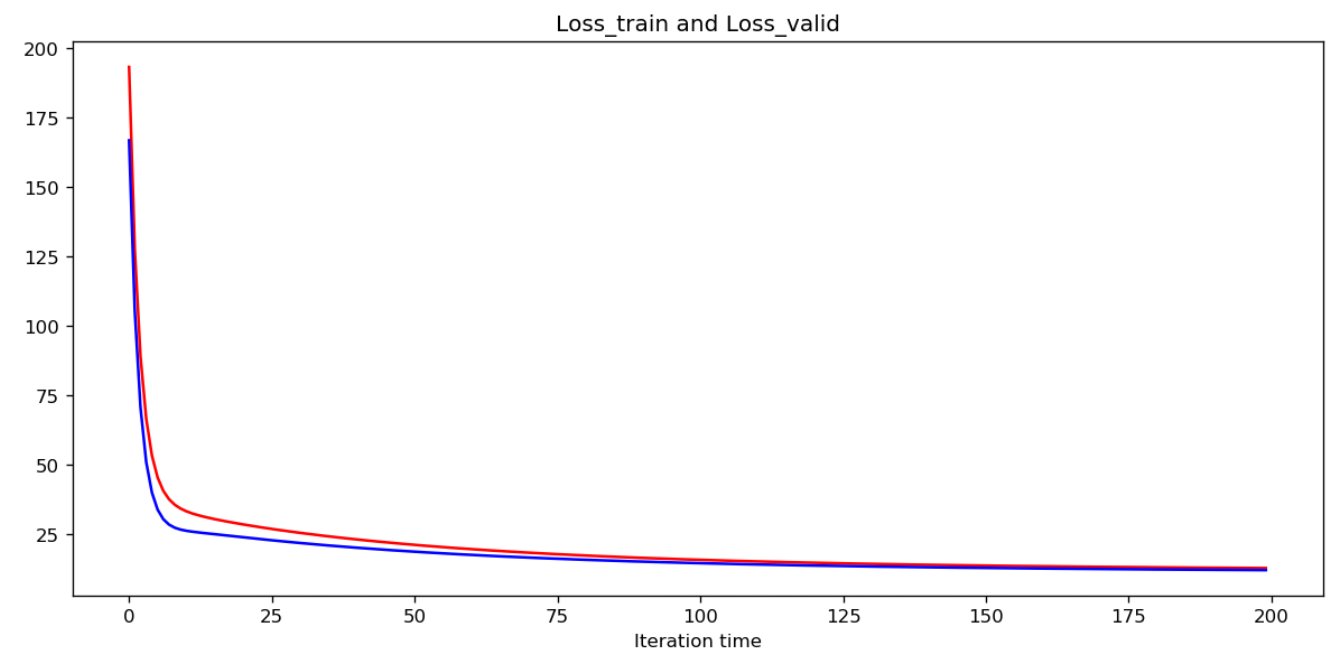
## Assessment Results:



## Predicted Results:



## Loss curve:



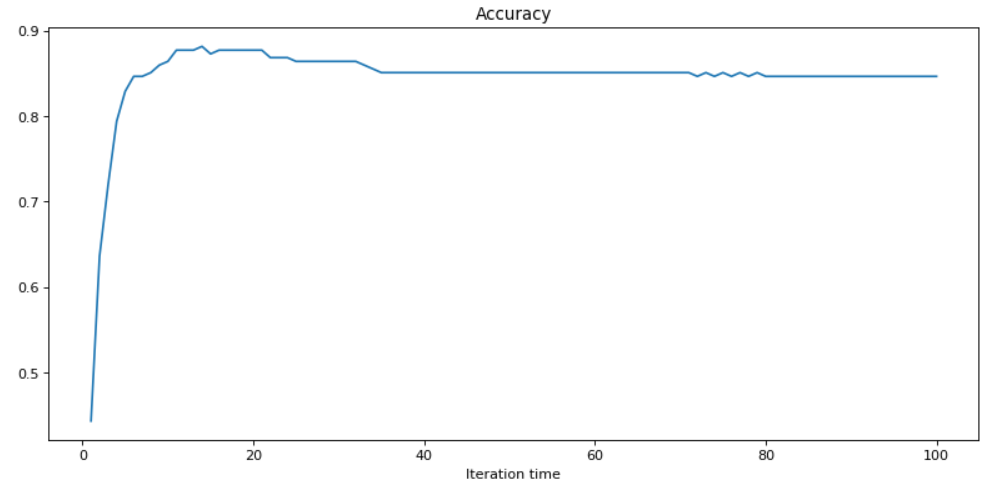
*Linear Classification:*

## Hyper-parameter selection: C =1000000

η = 0.001/C

= 0.000000001

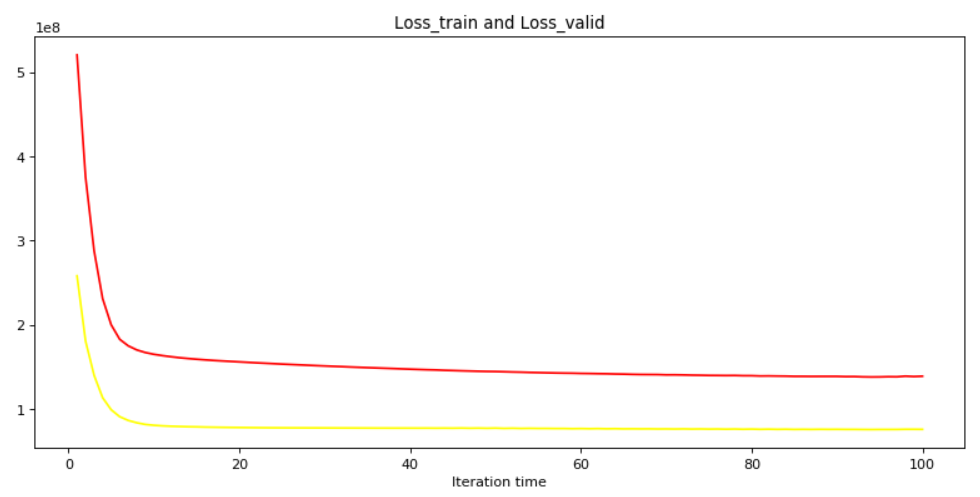
## Assessment Results:



## Predicted Results:



## Loss curve:



1. **Results analysis:**

*Linear Regression:*

Whenη = 0.05, the Linear Regression and Gradient Descent method can get a model whose loss of validation set is near 12 within 100 times iteration. And the loss of validation set decreases steadily.

*Linear Classification:*

## When C =1000000 and η = 0.001/C, the Linear Classification and Gradient Descent method can get a model whose accuracy of validation set is up to 0.88 within 15 times iteration. But the accuracy of validation keeps shaking around 0.85 later.

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

*Similarities:*

1).Both of them can predict the y according to features.

2).The structure of parameters is similar.

3).They both can use gradient descent method to optimize the parameters.

*Difference:*

1).The label of linear regression is continuous, but the label of linear classification is discrete.

2).Linear classification uses surrogate loss function to replace *l0/1(z)*function when optimize loss function. But linear regression can optimize loss function directly.

3).Linear regression can be assessed by mean square error. Linear classification can be assessed by accuracy.

**14. Summary:**

Linear Regression and Linear Classification are both method to fit the model in order to predict. They can be used in different scenes according to the requirement. We can use gradient descent to optimize the parameters of them conveniently. And we can use surrogate loss function to replace *l0/1(z)*function in linear classification problem so that we can get the derivation of loss function.