

**The Experiment Report of**

***Machine Learning***

**College Software College**

**Subject Software Engineering**

**Members**   **郭蕴喆**

**Student ID 201530611524**

**E-mail guoyunzhe.se@gmail.com**

**Tutor**   **Mingkui Tan**

**Date submitted** **2017.12.08**

目录

[A. Topic 4](#_Toc500461405)

[B. Time 4](#_Toc500461406)

[C. Reporter 4](#_Toc500461407)

[D. Purposes 4](#_Toc500461408)

[E. Data sets and data analysis 4](#_Toc500461409)

[1. Linear Regression 4](#_Toc500461410)

[2. Linear classification 4](#_Toc500461411)

[F. Experimental steps: 4](#_Toc500461412)

[1. Linear Regression and Gradient Descent 4](#_Toc500461413)

[2. Linear Classification and Gradient Descent 5](#_Toc500461414)

[G. Code 6](#_Toc500461415)

[1. Linear Regression and Gradient Descent 6](#_Toc500461416)

[(1) Function: Grad 6](#_Toc500461417)

[(2) Function: Draw 6](#_Toc500461418)

[(3) Function: BGD 7](#_Toc500461419)

[(4) main 7](#_Toc500461420)

[2. Linear Classification and Gradient Descent 8](#_Toc500461421)

[(1) Function: loss 8](#_Toc500461422)

[(2) Function: compare 8](#_Toc500461423)

[(3) Function: Grad 8](#_Toc500461424)

[(4) Function: corrate\_rate 9](#_Toc500461425)

[(5) Function: Draw 9](#_Toc500461426)

[(6) Function: BGD 9](#_Toc500461427)

[(7) main 10](#_Toc500461428)

[H. Selection of validation 10](#_Toc500461429)

[1. Linear Regression and Gradient Descent 11](#_Toc500461430)

[2. Linear Classification and Gradient Descent 11](#_Toc500461431)

[I. The initialization method of model parameters: 11](#_Toc500461432)

[J. The selected loss function and its derivatives: 11](#_Toc500461433)

[1. Linear Regression and Gradient Descent 11](#_Toc500461434)

[(1) Loss Function 11](#_Toc500461435)

[(2) Gradient 11](#_Toc500461436)

[2. Linear Classification and Gradient Descent 11](#_Toc500461437)

[(1) Loss Function 11](#_Toc500461438)

[(2) Gradient 11](#_Toc500461439)

[K. Experimental results and curve: 11](#_Toc500461440)

[1. Linear Regression and Gradient Descent 11](#_Toc500461441)

[(1) Hyper-parameter selection 12](#_Toc500461442)

[(2) Assessment Results (based on selected validation) 12](#_Toc500461443)

[(3) Predicted Results (Best Results) 12](#_Toc500461444)

[(4) Loss curve 12](#_Toc500461445)

[2. Linear Classification and Gradient Descent 13](#_Toc500461446)

[(1) Hyper-parameter selection 13](#_Toc500461447)

[(2) Assessment Results (based on selected validation) 13](#_Toc500461448)

[(3) Predicted Results (Best Results) 13](#_Toc500461449)

[(4) Loss curve 13](#_Toc500461450)

[(5) Accuracy 14](#_Toc500461451)

[L. Results analysis 15](#_Toc500461452)

[1. Linear Regression and Gradient Descent 15](#_Toc500461453)

[(1) 'Weights': 0.085, others the same 15](#_Toc500461454)

[(2) 'Weights': 0.000000000085, others the same 15](#_Toc500461455)

[2. Linear Classification and Gradient Descent 16](#_Toc500461456)

[(1) 'learning\_rate': 0.00085, others the same 16](#_Toc500461457)

[(2) 'threshold':1, others the same 17](#_Toc500461458)

[M. Similarities and differences 18](#_Toc500461459)

[N. Summary 18](#_Toc500461460)

# Topic

Linear Regression, Linear Classification and Gradient Descent

# Time

**2017/12/7**

# Reporter

**郭蕴喆**

# Purposes

* Further understand of linear regression and gradient descent.
* Conduct some experiments under small scale dataset.
* Realize the process of optimization and adjusting parameters.

# Data sets and data analysis

## Linear Regression

Linear Regression uses ***housing*** in LIBSVM Data, including 506 samples and each sample has 13 features.

## Linear classification

Linear classification uses ***australian*** in LIBSVM Data, including 690 samples and each sample has 14 features.

# Experimental steps:

## Linear Regression and Gradient Descent

1. Load the experiment data. Use ***load\_svmlight\_file***function in ***sklearn*** library.
2. Divide dataset.
3. Initialize linear model parameters. Set all parameter into zero, initialize it ***randomly***.
4. Choose loss function and derivation.
5. Calculate gradienttoward loss function from all samples.
6. Denote the opposite direction of gradient as.
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. Repeated step 5 to 8 for several times, and drawing graph of as well as with the number of iterations.

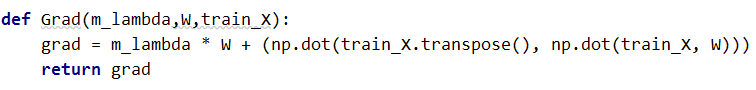
## Linear Classification and Gradient Descent

1. Load the experiment data.
2. Divide dataset into training set (70%) and validation set (30%).
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
5. Calculate gradient 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 ***training set*** and by validating under ***validation set***.
9. Repeated step 5 to 8 for several times, and drawing graph of as well as with the number of iterations.

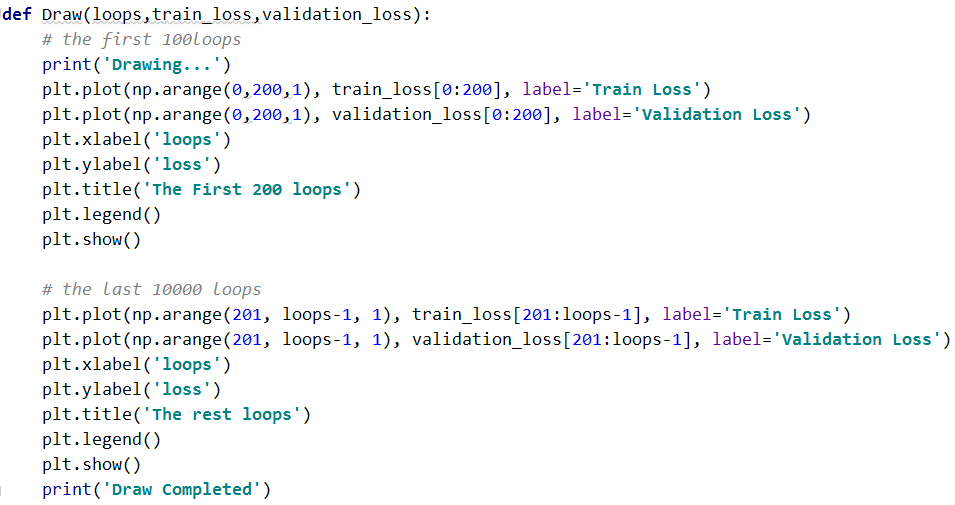
# Code

## Linear Regression and Gradient Descent

### Function: Grad



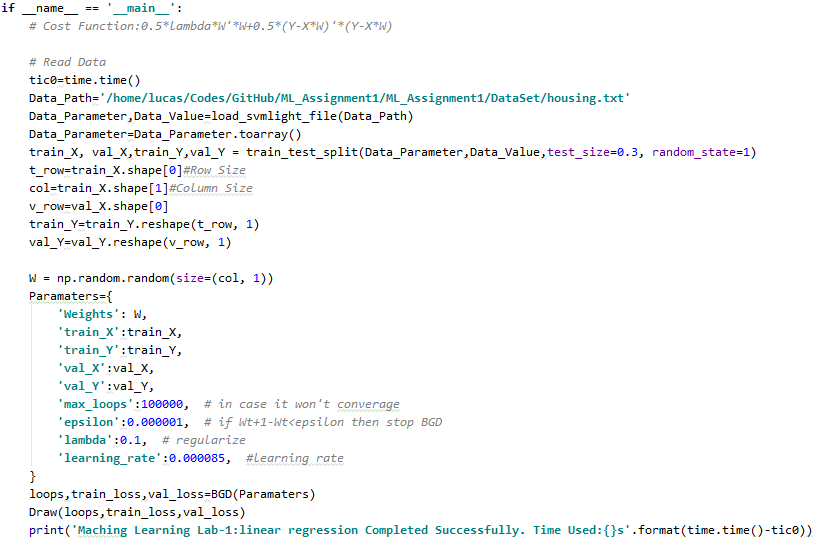
### Function: Draw



### Function: BGD

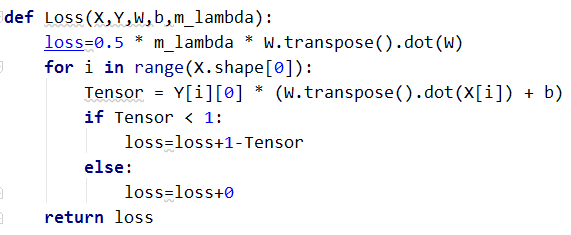


### main

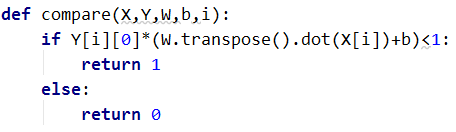


## Linear Classification and Gradient Descent

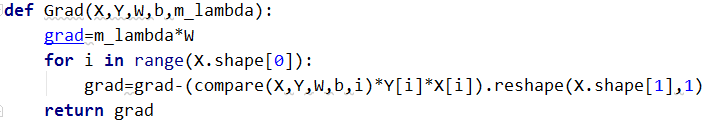
### Function: loss



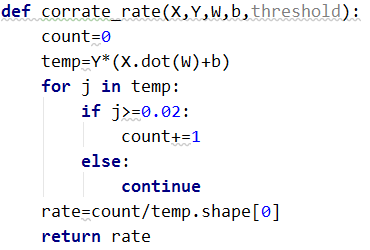
### Function: compare



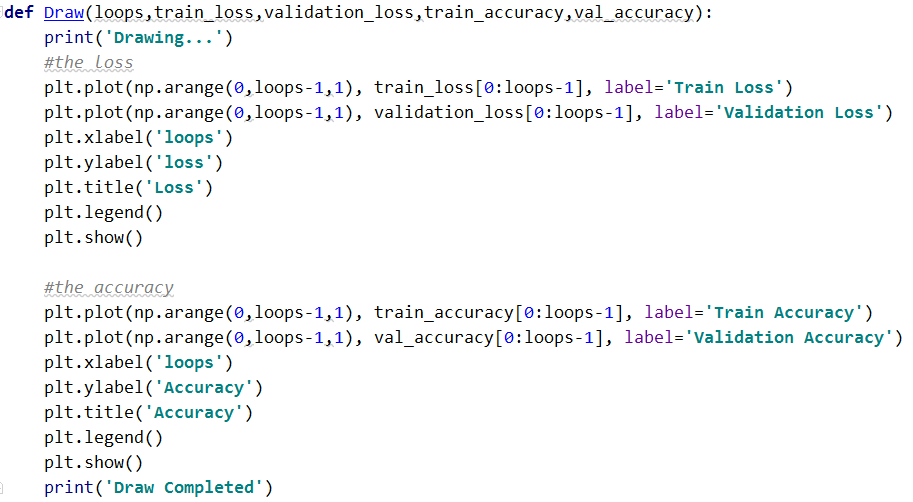
### Function: Grad



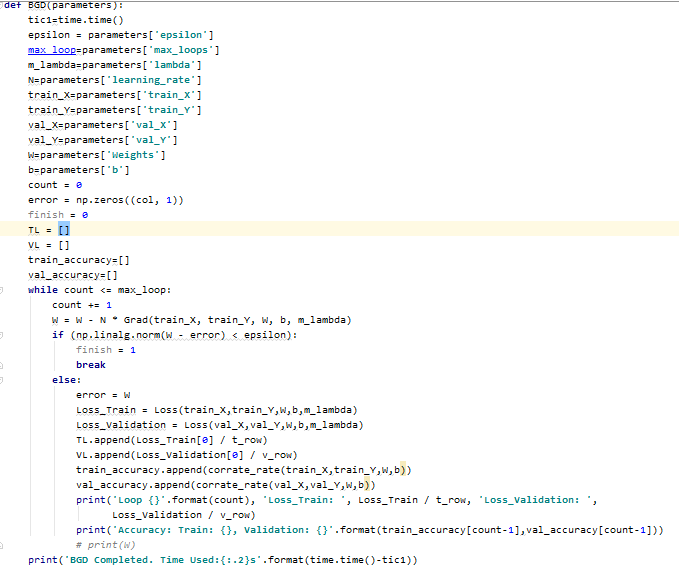
### Function: corrate\_rate



### Function: Draw



### Function: BGD



### main



# **Selection of validation**

## Linear Regression and Gradient Descent

I use Cross-Validation as the method to validate the result, during the whole experiment, the size of my validation set is 30%.

## Linear Classification and Gradient Descent

I use Cross-Validation as the method to validate the result, during the whole experiment, the size of my validation set is 25%.

# The initialization method of model parameters:

In two experiments, we all initialize the parameters in the same way:

W: Randomly initialize it in range of 0 and 1

the others: initialize according to experience

# The selected loss function and its derivatives:

## Linear Regression and Gradient Descent

### Loss Function

### Gradient

## Linear Classification and Gradient Descent

### Loss Function

### Gradient

# Experimental results and curve:

## Linear Regression and Gradient Descent

### Hyper-parameter selection

### Assessment Results (based on selected validation)

Train loss is about 0.1 bigger than validation loss

### Predicted Results (Best Results)

Loss\_Train: [13.11968115] Loss\_Validation: [ 10.89455921]

W:

[[ 0.62322054]

[ 0.6750166 ]

[ 0.19095844]

[ 0.19668393]

[ 0.94118078]

[ 0.89430195]

[ 0.29710537]

[ 0.49181249]

[ 0.46435008]

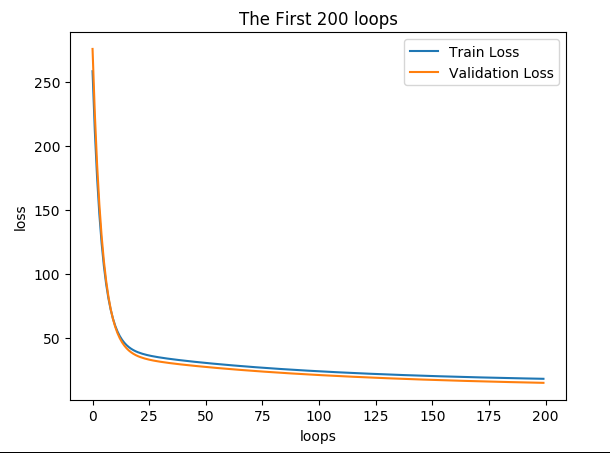
[ 0.46661833]

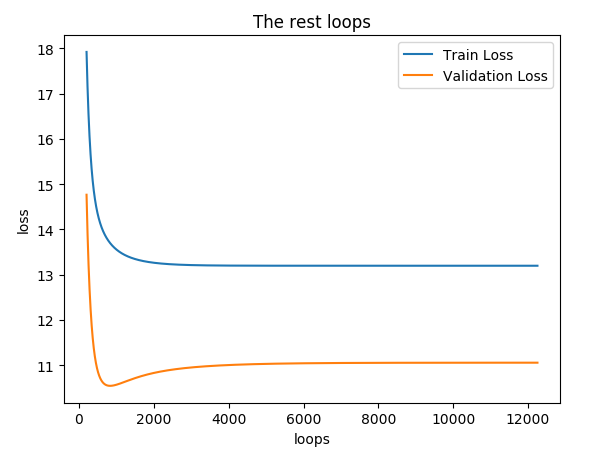
[ 0.63671179]

[ 0.43016872]

[ 0.54114606]]

### Loss curve





## Linear Classification and Gradient Descent

### Hyper-parameter selection

### Assessment Results (based on selected validation)

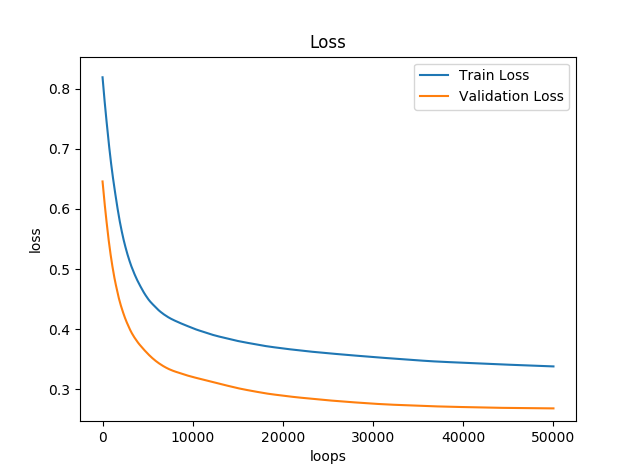
Train loss is about 0.15 bigger than validation loss, the accuracy in validation set will better than that in train set.

### Predicted Results (Best Results)

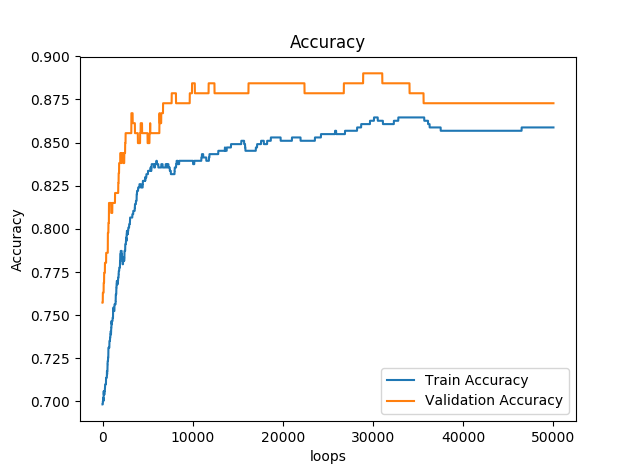
Loss\_Train: [0.34371135] Loss\_Validation: [0.2769692]

Accuracy: Train: 0.8588007736943907, Validation: 0.8728323699421965

### Loss curve



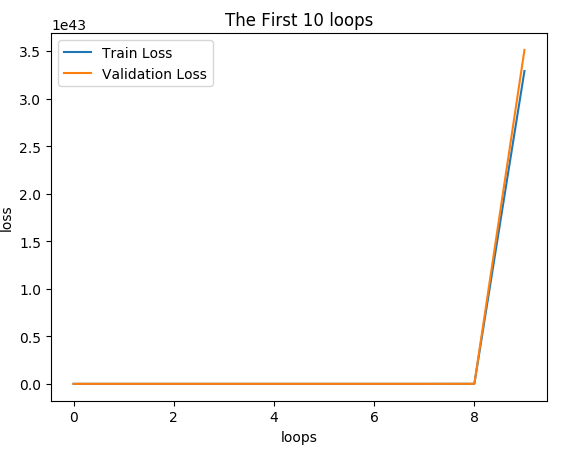
### Accuracy



# Results analysis

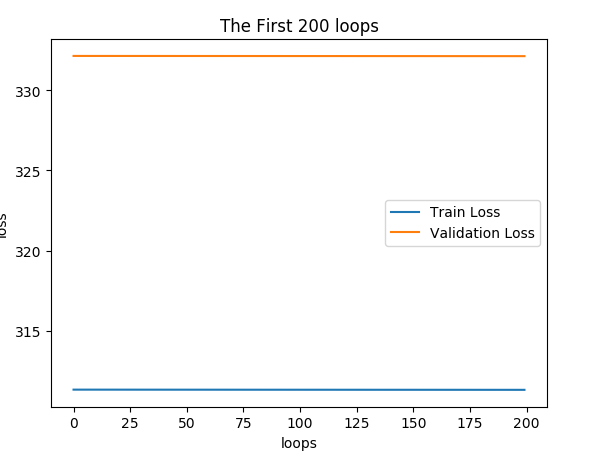
## Linear Regression and Gradient Descent

### 'Weights': 0.085, others the same



The curve won’t converge, because the learning rate is too big, it will always miss the local/global minimum and go away from it.

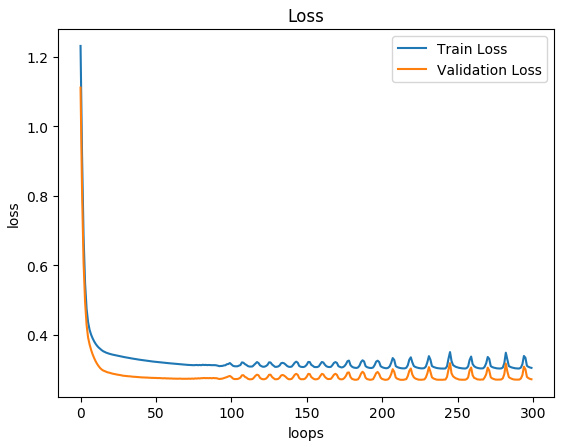
### 'Weights': 0.000000000085, others the same



The curve won’t converge either, because W is too small that it will take a very long time period.

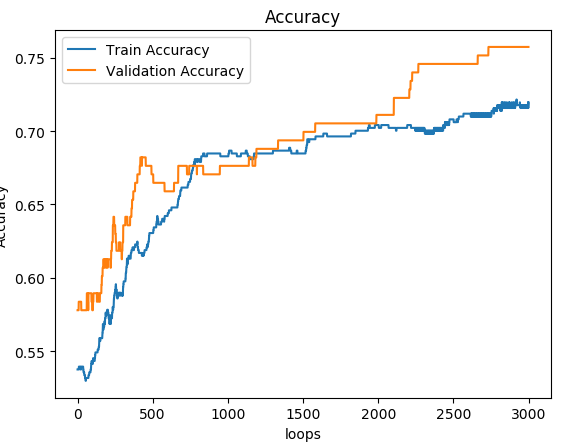
## Linear Classification and Gradient Descent

### 'learning\_rate': 0.00085, others the same



We can see clearly that at the last, our curve welter, I think the reason is the learning rate is a little big so it will go from a point close to the minimum to the symmetry point and then go back, it’s rather interesting.

### 'threshold':1, others the same



We can find the accuracy is much less than the curve of 'threshold':0, I think this is due to there are little outliers in the dataset.

# Similarities and differences

I think both problem are trying to find a loss function and using some method to make it smaller and smaller. In this way, we can use gradient decent in the two problem. However, linear regression focus on finding a function to solve the problem with successive values while classification focus on problem with discrete value and make classification. The loss they used are also different.

# Summary

I do believe we can overcome all the difficulties on our way learning machine learning, we should learn more from teachers, papers, books, and even from the web. We should also practice more, we can join kaggle or other competitions in the AI field. I think with the help of computer science, we will have a brighter future and I hope I can be one of the scientists and make something for the world.