

**The Experiment Report of**

***Machine Learning***

**College Software College**

**Subject Software Engineering**

**Members**  JunPeng Su

**Student ID 201530612743**

**E-mail sujunpeng.chn.@gmail.com**

**Tutor**  MingKui Tan

**Date submitted** **2017. 12 . 7**

**1. Topic:**

**Linear Regression, Linear Classification and Gradient Descent**

**2. Time:** 2017.12.2-2017.12.7

**3. Reporter:** JunPeng Su

**4. Purposes:**

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

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

1. Linear Regression uses Housing in LIBSVM Data, including 506 samples and each sample has 13 features.
2. Linear classification uses australian in LIBSVM Data, including 690 samples and each sample has 14 features.

**6. Experimental steps:**

Experiment includes two parts: 1) Linear Regression Model , 2) Linear Classification Model. Therefore, the experimental steps are also separated into two parts.

Linear Regression Model

1. Load experiment data and divide the dataset into train set and validation set.
2. Choose mean squared error as the loss function of the linear regression model.
3. Set the hyper-parameters of the linear model: learning rate and iterations number *epochs.*
4. Initialize the linear regression model parameters by setting all parameters into 0.
5. Calculate gradient G towards the loss function from all samples.
6. Denote the opposite direction of gradient as D.
7. Update the linear regression model by .
8. Calculate the train loss under train set and the validation loss under validation set.
9. Repeat step 5) to step 8) *epochs* times.
10. Draw graph of the train loss and the validation loss with the number of iterations

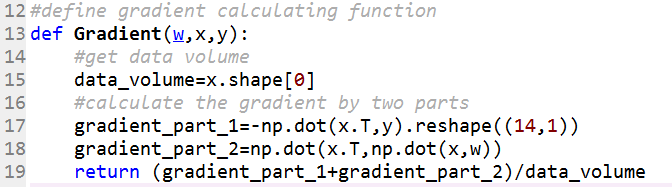
P.S. To calculate the gradient more efficiently, one column of all ones is added to feature matrix. And then the linear model is changed into .

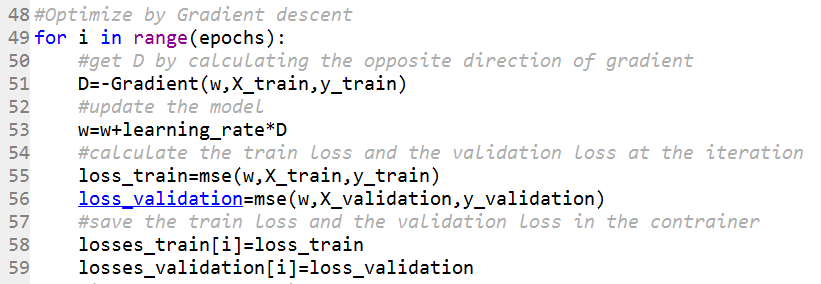
Linear Classification model

* 1. Load experiment data and divide dataset into train set and validation set.
  2. Choose loss function of the SVM model.
  3. Set hyper-parameters of the SVM model.
  4. Initialize SVM model parameters by setting all parameters into zeros.
  5. Calculate the gradient towards loss function from all samples.
  6. Denote the opposite direction of gradient as D.
  7. Update the SVM model by .
  8. Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative. Calculate the train loss under train set and the validation loss under validation set.
  9. Repeat step 5) to step 8) *epochs* times.
  10. Draw graph of the train loss and the validation loss with the number of iterations

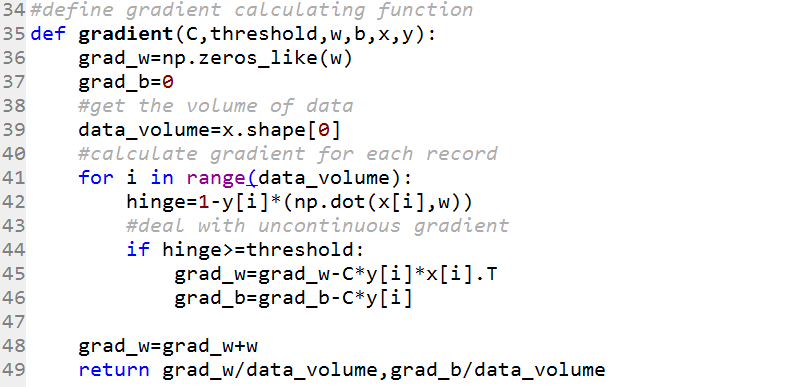
**7. Code:**

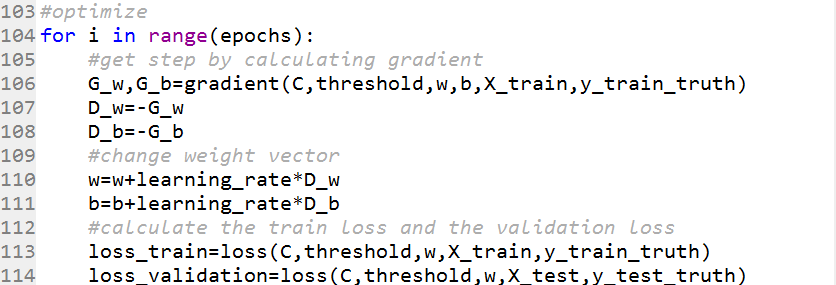
Linear regression model:





Linear classification model:





P.S. There are just important part of experiment codes. The whole codes are in ClassificationExperiment.ipynb and RegressionExperiment.ipynb.

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

hold-out validation

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

Setting all parameters into 0.

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

Linear Regression Model

loss function ( Mean squared error ):

derivatives:

Linear Classification Model:

loss function:

derivatives:

here,

**11. Experimental results and curve:**

Linear Regression Model

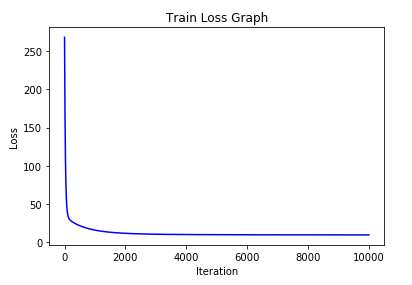
Try severalηthat include 0.00015, 0.0015, 0.0008, 0.002.

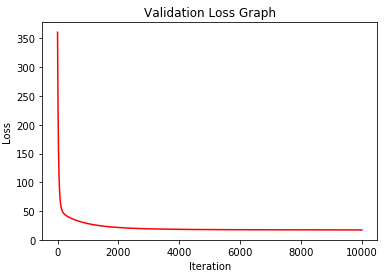
Try several epochs that include 1000, 5000, 10000.

|  |  |  |
| --- | --- | --- |
| η | epochs | Final Validation Loss |
| 0.00015 | 1000 | 142.495576657 |
| 0.00015 | 5000 | 47.3124665093 |
| 0.00015 | 10000 | 39.5746561725 |
| 0.0015 | 5000 | 22.1735036528 |
| 0.0008 | 5000 | 28.3924757137 |
| 0.0015 | 10000 | 18.6522737125 |
| 0.002 | 10000 | 18.0515503825 |

The best result is obtained by setting η into 0.002 and epochs into 10000.

## Loss curve:





Linear Classification Model

Try severalηthat include 0.00015, 0.015, 0.025, 0.02

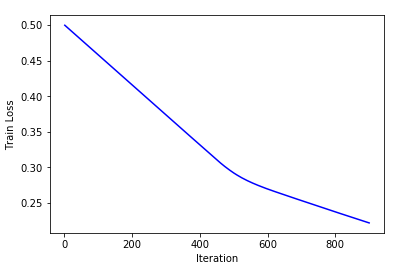
Try several epochs that include 500,900,1000, 2000

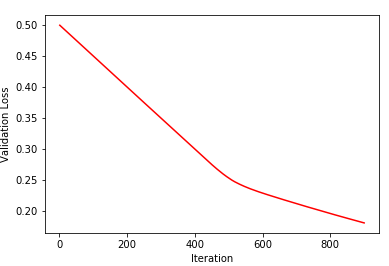
Try several thresholds that include 0, -0.1, 0.1 ,0.5

|  |  |  |  |
| --- | --- | --- | --- |
| η | epochs | thresholds | Accuracy |
| 0.00015 | 500 | 0 | 0.86231 |
| 0.00015 | 1000 | 0 | 0.86232 |
| 0.0015 | 1000 | 0 | 0.88405 |
| 0.0025 | 1000 | 0 | 0.88406 |
| 0.002 | 1000 | 0 | 0.89130 |
| 0.002 | 2000 | 0 | 0.88406 |
| 0.002 | 900 | 0 | 0.90580 |
| 0.002 | 900 | 0.5 | 0.86231 |
| 0.002 | 900 | 0.1 | 0.90580 |
| 0.002 | 900 | -0.1 | 0.89855 |

The best result is obtained by setting η into 0.02, epochs into 900 and threshold into 0. It obtains the highest accuracy which is 90.6%

## Loss curve:





**12. Results analysis:**

Linear Regression Model:

The loss curve indicates that the train loss and the validation loss decrease fast at first and stay in a stationary level. It means that there is no over-fitting and the model performs well.

Linear Classification Model:

As the train loss decreases, the model performs better and get a higher accuracy. When the loss is low enough, more epochs will not be able to improve the accuracy. Therefore, large epochs may not be always beneficial for better result but waste time and resources. It is important to make a balance.

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

Similarities: The main idea of both models is trying to use linear form to reveal the relationship between the features and the labels.

Differences: The value of labels in linear regression model is continuous while the value of labels in linear classification model is discrete.

**14. Summary:**

During the experiment, I build linear regression model and linear classification model from scratch. It further my understanding of both models. And the process of selecting better hyper-parameters make me realize the impact of hyper-parameters to the model performance.