The Experiment Report of Machine Learning



December 14, 2017

Grade:

 Graduate

Student ID：

201720144993

Supervisor:

Qingyao Wu

Author:

Shaojia hong

**SUBJECT:**SOFTWARE ENGINEERING

**SCHOOL:** SCHOOL OF SOFTWARE ENGINEERING

[[1]](#footnote-0)

Linear Regression, Linear Classiﬁcation and Gradient Descent

Abstract—

# INTRODUCTION

Linear regression describes the linear relationship between a predictor variable, plotted on the x-axis, and a response variable,plotted on the y-axis,the target is to learn a hypothesis or model f : X  Y

Linear Classification is a Classification that given training data (xi, yi) for i = 1 . . . n, with ∈ and  ∈ {−1, 1}, learn a classfier f(x) such that and  for a correct classification.

In order to further understand of linear regression and gradient descent.we do some experiments under small scale data set. And to realize the process of optimization and adjusting parameters.

# METHODS AND THEORY

## Dataset

We use two data sets,

The Linear Regression experiments uses [Housing](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/regression.html" \l "housing" \t "https://www.zybuluo.com/chenyaofo/note/_blank) in [LIBSVM Data](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/" \t "https://www.zybuluo.com/chenyaofo/note/_blank), including 506 samples and each sample has 13 features.And divided it into training set and verification set.  
The Linear classification experiments uses [australian](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html" \l "australian" \t "https://www.zybuluo.com/chenyaofo/note/_blank) in [LIBSVM Data](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/" \t "https://www.zybuluo.com/chenyaofo/note/_blank), including 690 samples and each sample has 14 features.

And divided it into training set and verification set.

## Experimental environment

Python3 and at least the following Python packages are included such as [sklearn](http://scikit-learn.org/stable/" \t "https://www.zybuluo.com/chenyaofo/note/_blank)，[numpy](http://www.numpy.org/" \t "https://www.zybuluo.com/chenyaofo/note/_blank)，[jupyter](http://jupyter.org/" \t "https://www.zybuluo.com/chenyaofo/note/_blank)，[matplotlib](https://matplotlib.org/" \t "https://www.zybuluo.com/chenyaofo/note/_blank).

It is recommended to install anaconda3 directly, which has built in the above Python packages.The experimental code and drawing are all done on jupyter.

## Steps

The step of 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. 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:  W= W+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. Repeated step 5 to 8 for several times, and drawing graph of  L\_train as well as  L\_validation with the number of iterations.

The step of 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:  W= W+η\*D,η 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  L\_train under the train set and  L\_validation by validating under validation set.
9. Repeated step 5 to 8 for several times, and drawing graph of  L\_train as well as  L\_validatioin with the number of iterations.

# Experiment

## Formula

### Linear regression formula

Target function is

****

Loss function is

****

### Linear Classification formula

Target function is

****

****

****

Loss function is

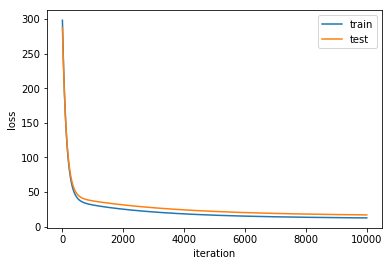
****

## Experimental results

1. Linear regression result is the following diagram and the linear regression parameter is such

Learn\_rate=0.001

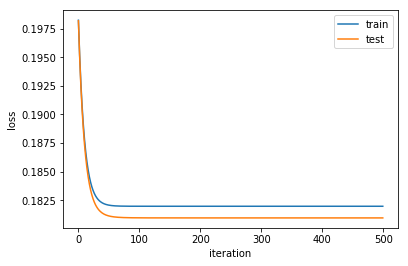
maxIteration = 10000

****

1. Linear Classification result is the following diagram and the linear regression parameter is such

iteration = 500

learning\_rate = 0.05



## code

### The code of Linear regression and gradient Descent:

from sklearn.datasets import load\_svmlight\_file

from sklearn.model\_selection import train\_test\_split

import numpy as np

from matplotlib import pyplot

train,target=load\_svmlight\_file('F:\housing\_scale')

#read data

x\_train,x\_test,y\_train,y\_test= train\_test\_split(train,target,test\_size= 0.2,random\_state = 0)#切分数据集

#compuet loss

def computeCost(X, y, theta):

m = y.shape[0]

J = (np.sum((X.dot(theta) - y)\*\*2)) / (2\*m)

return J

def gradientDescent(X, y, theta, alpha, num\_iters):

m = y.shape[0]

# loss

J\_history = np.zeros((num\_iters, 1))

for iter in range(num\_iters):

J\_history[iter] = computeCost(X, y, theta)

theta = theta - (alpha/m) \* (X.T.dot(X.dot(theta) - y))

return theta,J\_history

m, n = np.shape(x\_train)

theta= np.zeros((n,1))#init parameter

alpha = 0.001#learnrate

maxIteration = 10000#Iteration number

y\_train=y\_train.reshape(m,1)

m, n = np.shape(x\_test)

y\_test=y\_test.reshape(m,1)

theta\_train,loss\_iteration\_train= gradientDescent(x\_train,y\_train, theta, alpha, maxIteration)

theta\_test,loss\_iteration\_test = gradientDescent(x\_test,y\_test, theta, alpha, maxIteration)

pyplot.plot(loss\_iteration\_train, mfc='w',label='train')

pyplot.plot(loss\_iteration\_test, mfc='w',label='test')

pyplot.legend()

pyplot.xlabel("iteration")

pyplot.ylabel("loss")

pyplot.show()

### The code of Linear Classification and gradient Descent:

from sklearn.datasets import load\_svmlight\_file

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

import numpy as np

from matplotlib import pyplot

import pandas as pd

X,y=load\_svmlight\_file(r'F:\\australian\_scale')

X\_train,X\_validation,y\_train,y\_validation=train\_test\_split(X,y,test\_size=0.25)#分割数据集

y\_train=np.reshape(y\_train,(len(y\_train),1))

y=np.mat(y)

#添加一列1在最后一列

b=np.ones((np.shape(X\_train)[0],1))

#print(X\_train.shape)

X\_train=np.column\_stack((X\_train.toarray(),b))

#print(X\_train.shape)

b1=np.ones((np.shape(X\_validation)[0],1))

X\_validation=np.column\_stack((X\_validation.toarray(),b1))

m=X\_train.shape[1]

loss\_train=[]

loss\_validation=[]

w = np.zeros((15,))

C = 0.2

iteration = 500

learning\_rate = 0.05

def get\_loss(X,w,y):

loss=0

for i in range(X.shape[0]):

#loss+=max(0,1-np.sum((y.T)\*np.dot(X,w)))

loss+=max(0,1-y[i]\*np.dot(w.T,X[i]))

loss=(C/X.shape[0])\*loss+1/2\*(w.T.dot(w))

return loss

def gradient(X,w,y):

sum = 0

for j in range(X.shape[0]):

if (1-y[j]\*np.dot(w.T,X[j]))>=0:

sum+=-y[j]\*X[j]

w = w + (C/X.shape[0])\*sum

return w

# w初始化为1 维度m\*1

loss\_train=[]

loss\_validation=[]

for i in range(0,iteration):#it次迭代

gra=gradient(X\_train,w,y\_train)

w=w-learning\_rate\*gra

loss\_train.append(get\_loss(X\_train,w,y\_train))

loss\_validation.append(get\_loss(X\_validation,w,y\_validation))

pyplot.plot(loss\_train, mfc='w',label='train')

pyplot.plot(loss\_validation, mfc='w',label='test')

pyplot.legend()

pyplot.xlabel("iteration")

pyplot.ylabel("loss")

pyplot.show()

# conclusion

## Results analysis

Linear regression:

Through the adjustment of parameters, we can get a better regression results.

Linear classification:

Through the adjustment of multiple hyper-parameters,we can get a better SVM model

## Summary

Through this experiment,I further understood the principle of linear regression ,linear classification and gradient descent. By learning the gradient descent method, we can further understand the important content of the gradient learning, and realize the process of optimizing and adjusting the parameters.

1. [↑](#footnote-ref-0)