

**《机器学习》课程实验报告**

**学 院 软件学院**

**专 业 软件工程**

**组 员**  **肖亮昱**

**学 号 201530613146**

**邮 箱 624198747@qq.com**

**指导教师**  **吴庆耀**

**提交日期** **2017年 12月7日**

## 1. 实验题目: 线性回归、线性分类与梯度下降

## 2. 实验时间：2017年 12 月 2 日

## 3. 报告人:肖亮昱

## 4. 实验目的:

进一步理解线性回归和梯度下降的原理。

在小规模数据集上实践。

体会优化和调参的过程。

## 5. 数据集以及数据分析：

线性回归使用的是LIBSVM Data中的Housing数据，包含506个样本，每个样本有13个属性。请自行下载scaled版本，并将其切分为训练集，验证集。

线性分类使用的是LIBSVM Data中的australian数据，包含690个样本，每个样本有14 个属性。请自行下载scaled版本，并将其切分为训练集，验证集。

以上所有属性均加1，如线性回归样本属性为13个，实际过程中引入偏置，增加模型的表达能力

## 6. 实验步骤:

### *线性回归和梯度下降*

1、读取实验数据，使用sklearn库的load\_svmlight\_file函数读取数据。

2、将数据集切分为训练集和验证集，使用train\_test\_split函数切分数据集。

3、线性模型参数初始化，使用随机初始话

4、选择Loss函数及对其求导

5、求得所有样本对Loss函数的梯度。

6、取梯度的负方向，更新模型参数

7、在训练集上测试并得到Loss，在验证集上测试并得到Loss。

8、重复步骤5-7若干次，画出和随迭代次数的变化图。

### 线性分类和梯度下降

1、读取实验数据，使用sklearn库的load\_svmlight\_file函数读取数据。

2、将数据集切分为训练集和验证集，使用train\_test\_split函数切分数据集。

3、支持向量机模型参数初始化，使用随机初始化。

4、选择Loss函数及对其求导。

5、求得所有样本对Loss函数的梯度。

6、取梯度的负方向，更新模型参数。

7、选择合适的阈值，将计算结果大于阈值的标记为正类，反之为负类。在训练集上测试并得到Loss函数值，在验证集上测试并得到Loss函数值。

8、重复步骤5-7若干次，画出和随迭代次数的变化图。

## 7. 代码内容:

**线性回归**

#!/usr/bin/python

# -\*- coding: utf-8 -\*-

"""

-------------------------------------------------

File Name： LabTest

Description :

Author : Nathan

date： 2017/12/1

-------------------------------------------------

Change Activity:

2017/12/1:

-------------------------------------------------

"""

from random import random

\_\_author\_\_ = 'Nathan'

from sklearn.datasets import load\_svmlight\_file

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

import numpy as np

import matplotlib.pyplot as plt

def compute\_error(W,X\_test,y\_test):

# L=0

# for i in range(y\_test.shape[0]):

# L+=0.5\*((y\_test[i]-np.dot(W.transpose(),X\_test[i]))\*\*2)

# return L

N=X\_test.shape[0]

loss=np.matmul(y\_test.T,y\_test)-2\*np.matmul(np.matmul(W.T,X\_test.T),y\_test)+np.matmul(np.matmul(np.matmul(W.T,X\_test.T),X\_test),W)

return np.linalg.det(loss)/N

def compute\_gradient(W,X\_train,Y\_train):

L=np.dot(-X\_train.transpose(),Y\_train)+np.dot(np.dot(X\_train.transpose(),X\_train),W)

return L

def optimizer(X\_train,y\_trian,W\_start,learning\_rate):

W=W\_start

W=W-learning\_rate\*compute\_gradient(W,X\_train,y\_trian)

return W

def shuffle(X,y):

rng\_state = np.random.get\_state()

np.random.shuffle(X)

np.random.set\_state(rng\_state)

np.random.shuffle(y)

return X,y

def getdata():

X, y = load\_svmlight\_file(r'C:\Users\jy\Desktop\Libsvmdata\housing\_scale.txt')

X = np.c\_[np.ones(len(X.toarray())), X.toarray()]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

y\_train = y\_train.reshape([len(y\_train), 1])

y\_test = y\_test.reshape([len(y\_test), 1])

X\_train, y\_train = shuffle(X\_train, y\_train)

X\_test,y\_test=shuffle(X\_test,y\_test)

return X\_train, X\_test, y\_train, y\_test

def linearmodel():

X\_train, X\_test, y\_train, y\_test = getdata()

W = np.random.rand(14,1)

learning\_rate = 0.0001

iter\_num=50

test\_error=[]

train\_error=[]

for i in range(iter\_num):

W=optimizer(X\_train,y\_train,W,learning\_rate)

test\_error.append(compute\_error(W,X\_test,y\_test))

train\_error.append(compute\_error(W,X\_train,y\_train))

plt.figure('learning\_rate='+str(learning\_rate))

plt.xlabel('Iteration')

plt.ylabel('Loss')

plt.plot(range(iter\_num),test\_error,label="validation error")

plt.plot(range(iter\_num), train\_error, label="train error")

plt.legend()

plt.show()

linearmodel()

**线性分类**：

#!/usr/bin/python

# -\*- coding: utf-8 -\*-

"""

-------------------------------------------------

File Name： LinearClassification

Description :

Author : Nathan

date： 2017/12/2

-------------------------------------------------

Change Activity:

2017/12/2:

-------------------------------------------------

"""

\_\_author\_\_ = 'Nathan'

from sklearn.datasets import load\_svmlight\_file

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

import numpy as np

import matplotlib.pyplot as plt

plt.rcParams['font.sans-serif']=['SimHei'] #用来正常显示中文标签

plt.rcParams['axes.unicode\_minus']=False #用来正常显示负号

parm={"C":0.9}

def compute\_error(W,X\_test,y\_test):

L=0

N=y\_test.shape[0]

for i in range(N):

L+=max(0,1-y\_test[i]\*np.dot(W.transpose(),X\_test[i]))

return (0.5\*np.dot(W.T,W)+L\*parm.get("C"))/N

def compute\_gradient(W,X\_train, y\_train):

L\_dW=np.repeat(0.,X\_train.shape[1])

N = y\_train.shape[0]

for i in range(N):

if y\_train[i]\*(np.dot(W.transpose(),X\_train[i]))<1:

L\_dW+=-y\_train[i]\*X\_train[i]

else:

pass

return parm.get("C")\*L\_dW+W

def SVM\_train(X\_train, y\_train,W,learning\_rate):

W-=learning\_rate\*compute\_gradient(W,X\_train, y\_train)

return W

def shuffle(X,y):

rng\_state = np.random.get\_state()

np.random.shuffle(X)

np.random.set\_state(rng\_state)

np.random.shuffle(y)

return X,y

def getdata():

X, y = load\_svmlight\_file(r'C:\Users\jy\Desktop\Libsvmdata\australian\_scale.txt')

X = np.c\_[np.ones(len(X.toarray())), X.toarray()]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

X\_train, y\_train = shuffle(X\_train, y\_train)

X\_test, y\_test = shuffle(X\_test, y\_test)

return X\_train, X\_test, y\_train, y\_test

def SVMModel():

X\_train, X\_test, y\_train, y\_test=getdata()

learning\_rate = 0.0002

iter\_num = 1000

train\_error=[]

test\_error=[]

W = np.random.rand(X\_train.shape[1])

for i in range(iter\_num):

W=SVM\_train(X\_train, y\_train,W,learning\_rate)

train\_error.append(compute\_error(W,X\_train,y\_train))

test\_error.append(compute\_error(W,X\_test,y\_test))

y\_predict=np.dot(X\_test,W)

y\_predict = y\_predict \* y\_test

for i in range(y\_predict.shape[0]):

if y\_predict[i]>=0:

y\_predict[i]=1

else:

y\_predict[i]=0

print("Acurracy:")

print(sum(y\_predict)/y\_predict.shape[0])

plt.xlabel('Iteration')

plt.ylabel('Loss')

plt.plot(range(iter\_num), train\_error,label="train error")

plt.plot(range(iter\_num), test\_error, label="validation error")

plt.legend()

plt.show()

SVMModel()

（针对线性回归和线性分类分别填写8-12内容）

## 线性回归：

### 8. 选择的评估方法（留出法，交叉验证，k折交叉验证等）:

采用留出法来切分数据集：调用train\_test\_split函数切分数据集, test\_size=0.33,选取0.33的数据作为验证集，随机选择。

### 9. 模型参数的初始化方法:

采用随机方式初始话权重W

### 10.选择的loss函数及其导数:

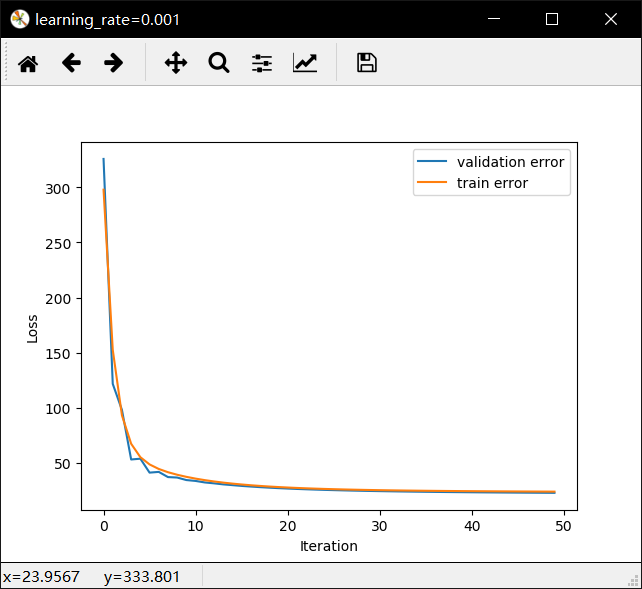
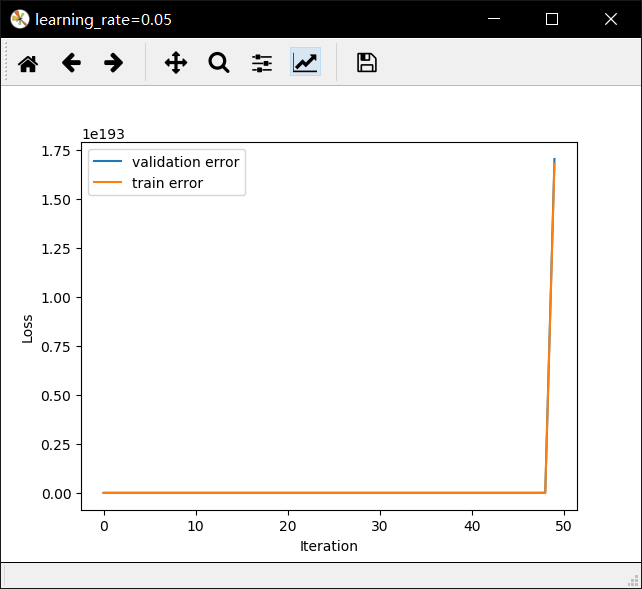
### 11.实验结果和曲线图:

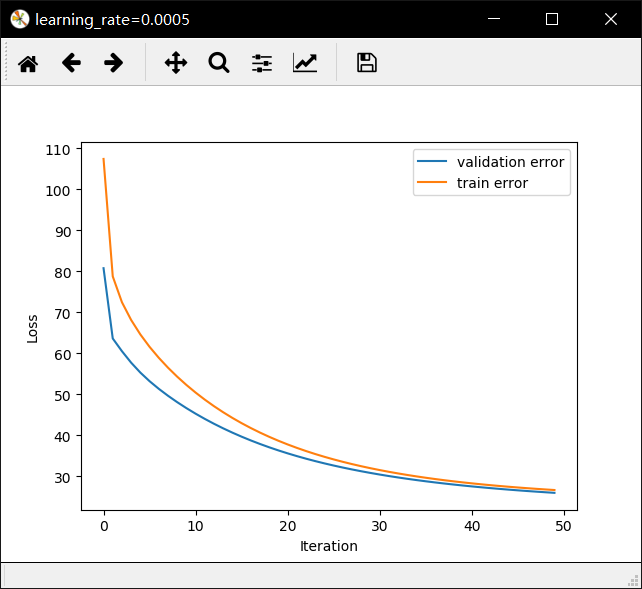
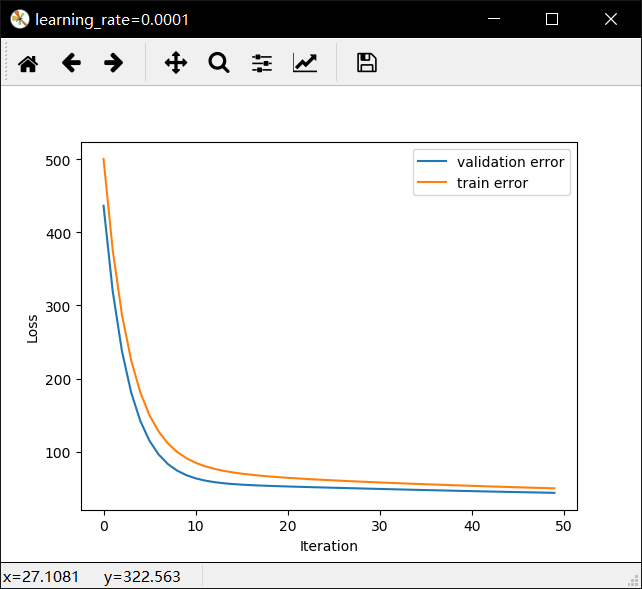
#### 超参数选择（η,epoch等）：

η = 0.001,0.05,0.0005,0.0001

iter\_num=50

#### 评估结果（根据选择的评估方法）：

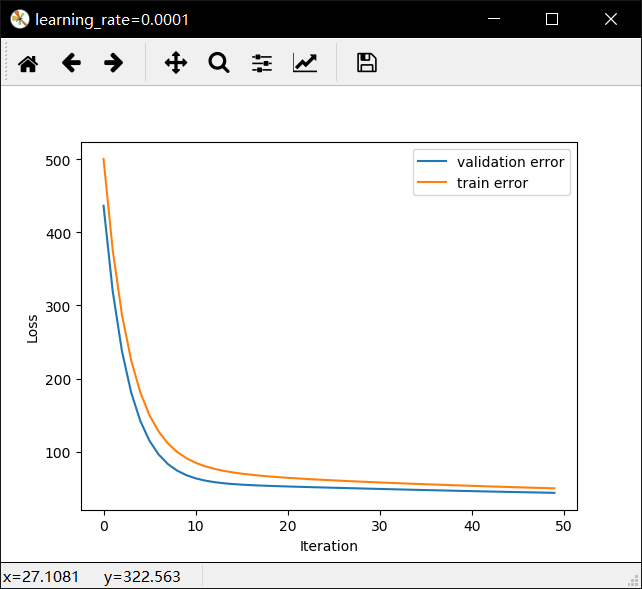
 

#### 预测结果（最佳结果）：

η = 0.0001

iter\_num=50

#### loss曲线图：



#### 12.实验结果分析:

本次实验以线性模型为基础，用包含506个样本的Housing数据进行训练和测试，用梯度下降的方法优化更新权重，在0到10次的迭代过程中，loss下降明显，在10到50次的迭代过程中，loss的下降明显趋于缓和，此时梯度更新对loss的影响较小，同时对learning rate 的调整可以看出，learning rate的设置不宜过大，否则可能会导致梯度下降找不到最优点

## 线性分类：

### 8. 选择的评估方法（留出法，交叉验证，k折交叉验证等）:

采用留出法来切分数据集：调用train\_test\_split函数切分数据集, test\_size=0.33,选取0.33的数据作为验证集，随机选择。

### 9. 模型参数的初始化方法:

采用随机方式初始话权重W

### 10.选择的loss函数及其导数:

Loss函数为margin loss和hinge loss的组合

其导数为：

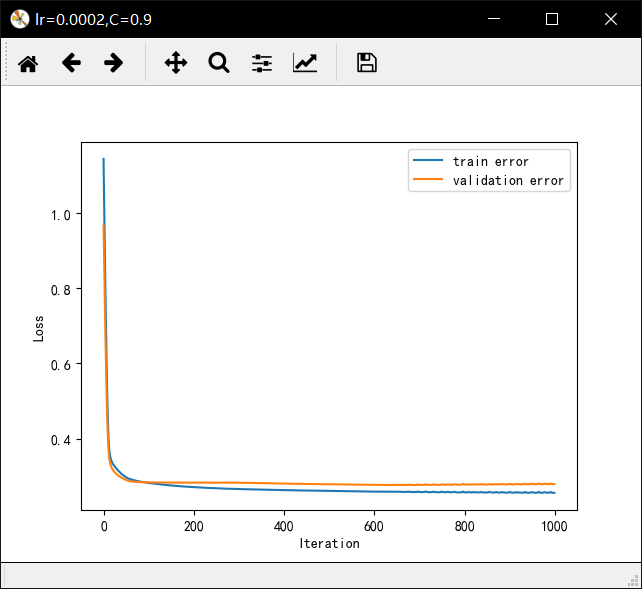
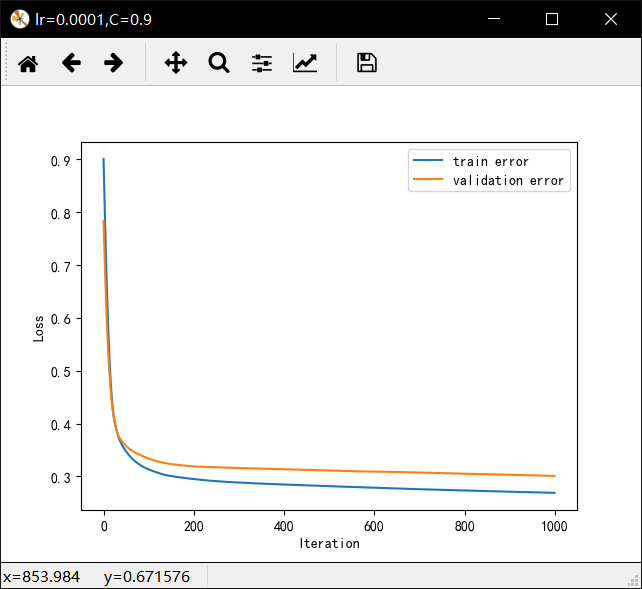
### 11.实验结果和曲线图:

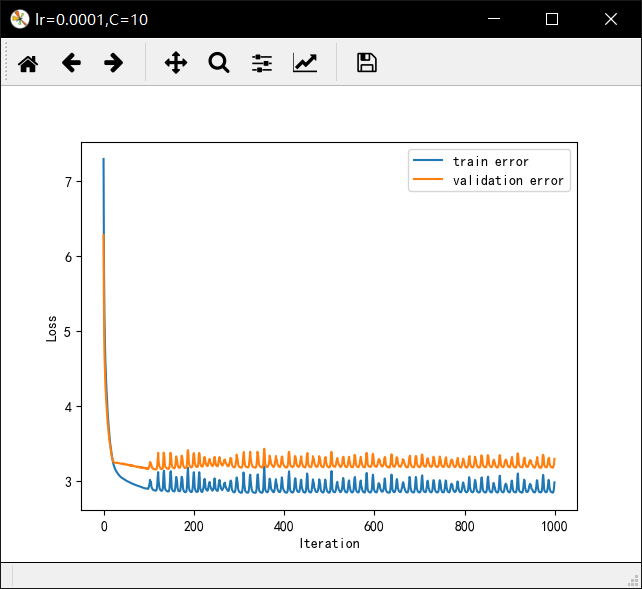
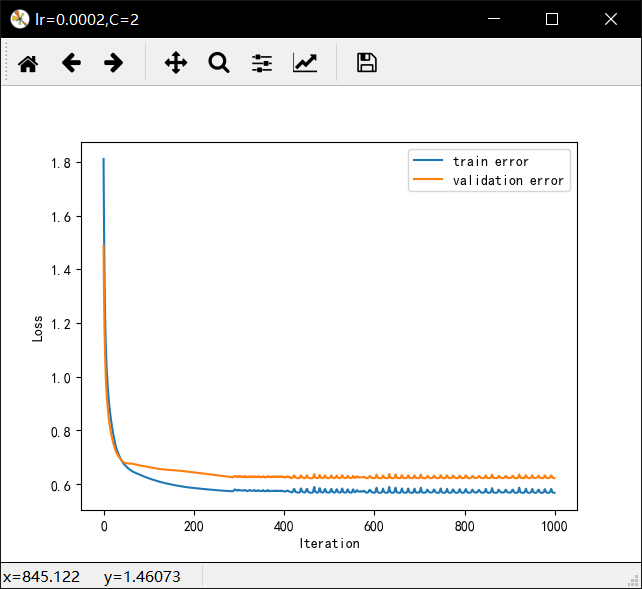
超参数选择（η,epoch等）

**Lr=0.0002,0.0001,0.0001**

**C=0.9,0.9,10**

#### 评估结果（根据选择的评估方法）：

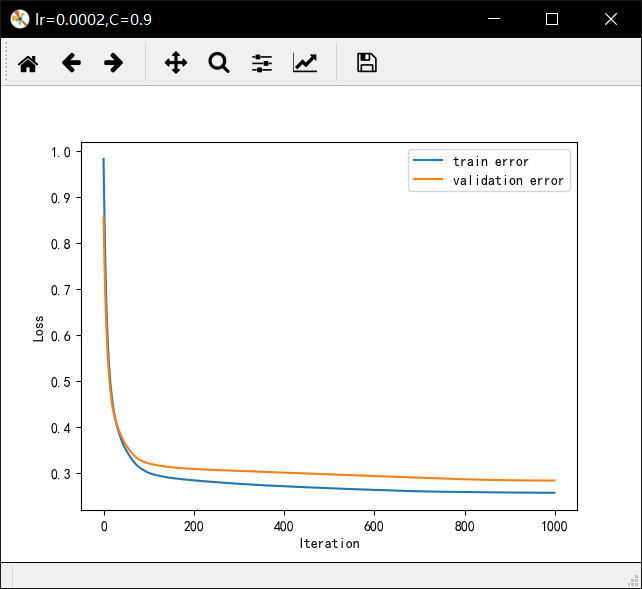
 

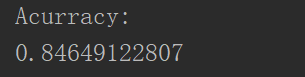
 

#### 预测结果（最佳结果）：

Lr=0.0002,C=0.9

#### loss曲线图：





### 12.实验结果分析:

本次实验以SVM模型为基础，用包含690个样本的[australian](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html#australian)数据进行训练和测试，用梯度下降的方法优化更新权重，在0到100次的迭代过程中，loss下降明显，在100到1000次的迭代过程中，loss的下降明显趋于缓和，可能此时梯度下降的方法到达了一个最优点，同时，测试数据在迭代过程中保持了和训练数据相似的下降曲线，说明模型和参数选择合理，泛化能力较强，learning rate 和参数C的选择我们可以看出，递归下降的收敛速度跟这二者有着很大关系。

## 13.对比线性回归和线性分类的异同点：

线性回归和线性分类本质上都是模型的拟合，都是在最小化loss的过程中调整参数来拟合数据。他们的不同在于输出变量的类型，线性回归输出的是定量分析，也就是预测一个实际值，线性分类的输出是定性分析，也就是做分类判断。在最小化loss函数的过程中，他们的思路是相通相近的，都可以用递归下降的方式来优化权重

## 14.实验总结：

通过此次实验，进一步理解了线性回归，线性分类和梯度下降的原理。在小规模数据集上实践体会优化和调参的过程。同时以代码的形式实现了线性回归，线性分类，理论与实践结合，更加深刻地掌握了机器学习的基本原理和方法