

**《机器学习》课程实验报告**

**学 院 软件学院**

**专 业 软件工程**

**组 员**  **林艾琳**

**学 号 201530612200**

**邮 箱 743852586@qq.com**

**指导教师**  **吴庆耀**

**提交日期** **2017年12月8日**

## 1. 实验题目: [线性回归、线性分类与梯度下降](https://www.zybuluo.com/chenyaofo/note/949882" \t "https://www.zybuluo.com/chenyaofo/note/_blank)

## 2. 实验时间：2017年12月2日

## 3. 报告人：林艾琳

## 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。

## 数据集以及数据分析：

## **数据集**

## 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。

## 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。

## **数据分析**

## 使用了python包：sklearn，numpy，jupyter，matplotlib等工具，以及线性回归、线性分类、梯度下降等方法。

## 6. 实验步骤:

## **线性回归和梯度下降**

## 读取实验数据，使用sklearn库的[load\_svmlight\_file](http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_svmlight_file.html" \t "https://www.zybuluo.com/chenyaofo/note/_blank)函数读取数据。

## 将数据集切分为训练集和验证集，本次实验不切分测试集。使用[train\_test\_split](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html" \t "https://www.zybuluo.com/chenyaofo/note/_blank)函数切分数据集。

## 线性模型参数初始化，可以考虑全零初始化，随机初始化或者正态分布初始化。

## 选择Loss函数及对其求导，过程详见课件ppt。

## 求得所有样本对Loss函数的梯度。

## 取梯度的负方向，记为。

## 更新模型参数，。为学习率，是人为调整的超参数。

## 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。

## 重复步骤5-8若干次，画出和随迭代次数的变化图。

## **线性分类和梯度下降**

## 读取实验数据，使用sklearn库的[load\_svmlight\_file](http://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_svmlight_file.html" \t "https://www.zybuluo.com/chenyaofo/note/_blank)函数读取数据。

## 将数据集切分为训练集和验证集，本次实验不切分测试集。使用[train\_test\_split](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html" \t "https://www.zybuluo.com/chenyaofo/note/_blank)函数切分数据集。

## 支持向量机模型参数初始化，可以考虑全零初始化，随机初始化或者正态分布初始化。

## 选择Loss函数及对其求导，过程详见课件ppt。

## 求得所有样本对Loss函数的梯度。

## 取梯度的负方向，记为。

## 更新模型参数，。为学习率，是人为调整的超参数。

## 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。

## 重复步骤5-8若干次，画出和随迭代次数的变化图。

## 7. 代码内容:

## 线性回归

# -\*- coding: UTF-8 -\*-

from sklearn.datasets import load\_svmlight\_file

from numpy import \*

import numpy as np

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

from math import ceil

import matplotlib.pyplot as plt

def **get\_data**(s):

data = load\_svmlight\_file(s)

return data[0], data[1]

X, Y = get\_data(*"data1.txt"*)

data\_num=506

t\_size=0.33

validation\_num=int(ceil(data\_num\*t\_size))

train\_num=data\_num-validation\_num

X\_train, X\_validation, Y\_train, Y\_validation = train\_test\_split(X, Y,test\_size=t\_size, random\_state=50)

Xt=mat(X\_train.toarray())

Xv=mat(X\_validation.toarray())

#<class 'numpy.matrixlib.defmatrix.matrix'>

Xt = np.column\_stack((Xt,[np.float64(1) for x in range(train\_num)]))

Xv = np.column\_stack((Xv,[np.float64(1) for x in range(validation\_num)]))

Yt=mat(Y\_train).T

Yv=mat(Y\_validation).T

w=mat([np.float(10)for i in range(14)]).T

learningRate=0.0001#学习率

times=10

lossT=[]

lossV=[]

for i in range(0,times):

LT=np.float(((Xt\*w-Yt).T)\*(Xt\*w-Yt))

LV=np.float(((Xv\*w-Yv).T)\*(Xv\*w-Yv))

w+=-2\*learningRate\*(Xt.T)\*(Xt\*w-Yt)

lossT.append(LT/train\_num)

lossV.append(LV/validation\_num)

x=np.arange(0,times)

plt.plot(x,lossT,label=*"Train"*)

plt.plot(x,lossV,label=*"Validation"*)

plt.title(*u"Loss"*)

plt.legend()#Places a legend on the axes

plt.xlabel(*u"Time"*)

plt.ylabel(*u"Loss"*)

plt.show()

## 线性分类

# -\*- coding: UTF-8 -\*-

from sklearn.datasets import load\_svmlight\_file

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

from numpy import \*

import numpy as np

import matplotlib.pyplot as plt

from math import ceil

def **sigmoid**(X):

return 1.0/(1+exp(-X))

def **get\_data**(s):

data = load\_svmlight\_file(s)

return data[0], data[1]

X, Y = get\_data(*"data2.txt"*)

data\_num,att\_num = shape(X)

t\_size=0.33

validation\_num=int(ceil(data\_num\*t\_size))

train\_num=data\_num-validation\_num

X\_train, X\_validation, Y\_train, Y\_validation = train\_test\_split(X, Y, test\_size=t\_size, random\_state=50)

Xt=mat(X\_train.toarray())

Xv=mat(X\_validation.toarray())

Yt=mat(Y\_train).T

Yv=mat(Y\_validation).T

times=10

lossT=[]

lossV=[]

learningRate = 0.001

w= ones((att\_num,1))

for k in range(times):

hT = sigmoid(Xt\*w)

tempT = Yt - hT

lossT.append(np.float64((tempT.T)\*tempT)/train\_num)

hV=sigmoid(Xv\*w)

tempV = (Yv - hV)

lossV.append(np.float64((tempV.T) \* tempV)/validation\_num)

w += learningRate \* Xt.T\* tempT

x=np.arange(0,times)

plt.plot(x,lossT,label=*"Train"*)

plt.plot(x,lossV,label=*"Validation"*)

plt.title(*u"Loss"*)

plt.legend()#Places a legend on the axes

plt.xlabel(*u"Time"*)

plt.ylabel(*u"Loss"*)

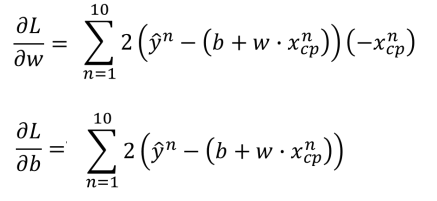
plt.show()

## 选择的评估方法：留出法

## 9. 模型参数的初始化方法：全零初始化

## 10.选择的loss函数及其导数：

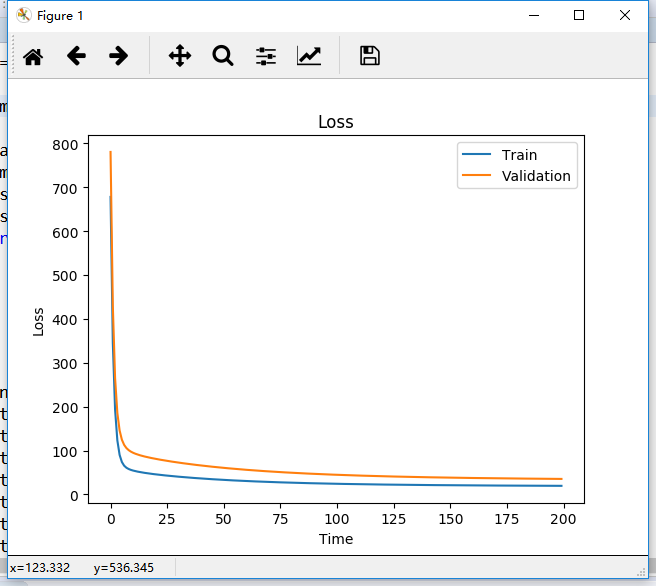
## 



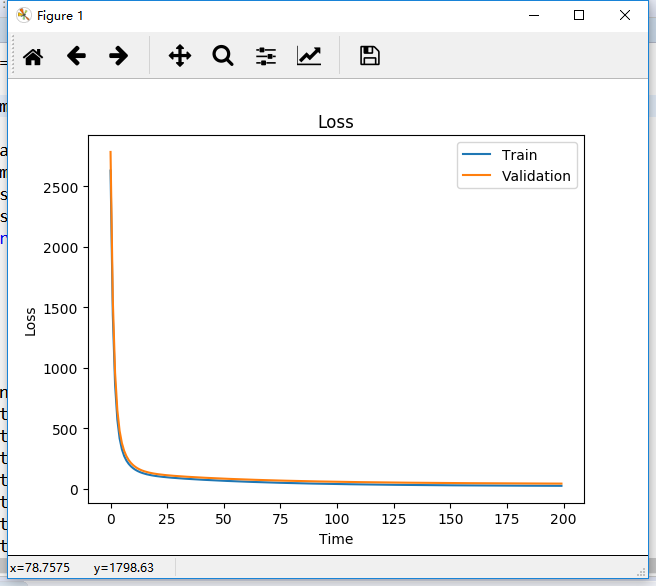
## 11.实验结果和曲线图：

## **超参数选择（η,epoch等）：**η=0.0001

## **评估结果（根据选择的评估方法）：**

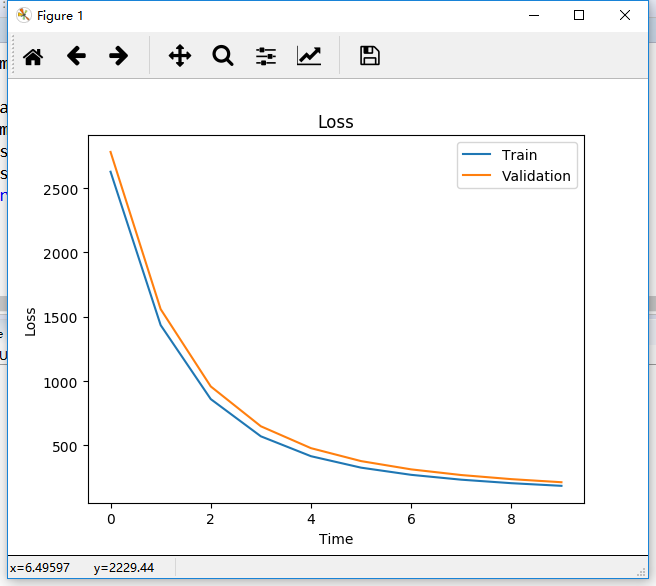


## **预测结果（最佳结果）：**

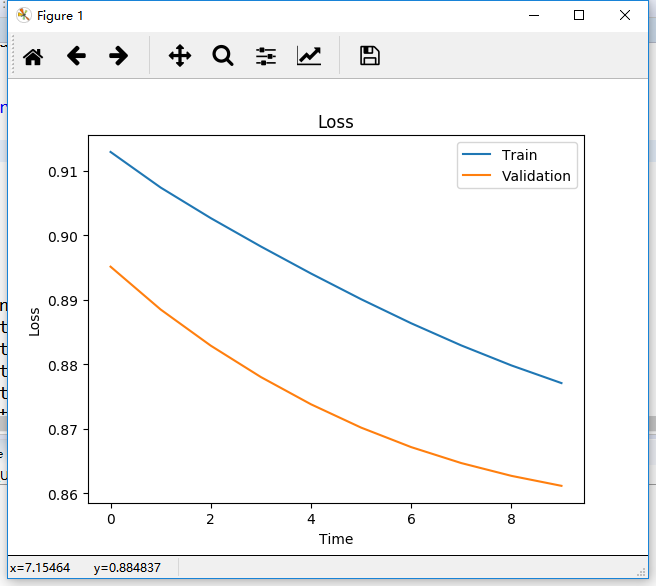


## **loss曲线图：**

线性回归：



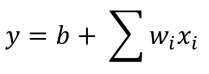
线性分类：



## 实验结果分析:

## 通过梯度下降算法，wT自发进行调整，使得loss下降，函数对数据的拟合程度提升。

## 13.对比线性回归和线性分类的异同点：

同：

## 异：线性回归y的取值范围是R，线性分类y的取值范围是{+1,-1}

## 14.实验总结：

## 这次实验，我对线性回归、线性分类与梯度下降有了初步了解