

**《机器学习》课程实验报告**

**学 院 软件学院**

**专 业 软件工程**

**组 员**   **洪梓杰**

**学 号 201530611593**

**邮 箱 464538338@qq.com**

**指导教师**  **吴庆耀**

**提交日期** **2017年 12 月 7 日**

## 1. 实验题目:

## 2. 实验时间：2017年 12 月 2 日

## 3. 报告人:洪梓杰

## 4. 实验目的:

1.进一步理解线性回归和梯度下降的原理。

2.在小规模数据集上实践。

3.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。

## 6. 实验步骤:

本次实验代码及画图均在jupyter上完成。

*线性回归和梯度下降*

1. 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 "_blank)函数读取数据。
2. 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)函数切分数据集。
3. 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。
4. 选择Loss函数及对其求导，过程详见课件ppt。
5. 求得**所有样本**对Loss函数的梯度。
6. 取梯度的负方向，记为。
7. 更新模型参数，。为学习率，是人为调整的超参数。
8. 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。
9. 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*

1. 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 "_blank)函数读取数据。
2. 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)函数切分数据集。
3. 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。
4. 选择Loss函数及对其求导，过程详见课件ppt。
5. 求得**所有样本**对Loss函数的梯度。
6. 取梯度的负方向，记为。
7. 更新模型参数，。为学习率，是人为调整的超参数。
8. 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。
9. 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:

（针对线性回归和线性分类分别填写8-12内容）

本次定义的alpha即为学习率，其中只给出梯度下降的代码：

**（1）线性回归**

#测试过正则化

grad = (y-label) \*X /m #+delta \* sum(norm(weights[1:]))/m

#更新梯度

weights = weights - (alpha \* grad)

y = X \* weights # shape(339,0)

y\_test = X\_test \* weights

#计算损失

loss\_train= loss(y,label,weights)

**（2）线性分类 （GD for SVM）**

#SGD

index = random.randint(0,m)

Wx = X[index,:] \* weights

#print( label[:,label])

if(1 - Wx[0,0] \* label[:,index] >= 0):

grad = 1/delta \* linalg.norm(weights) - X[index,:].T \* label[:,index]

else:

grad = 1/delta \* linalg.norm(weights)

weights = weights - alpha \* grad

` y = X\*(weights)

#计算损失

loss\_train= loss(y,label,weights)

## 8. 选择的评估方法（留出法，交叉验证，k折交叉验证等）:

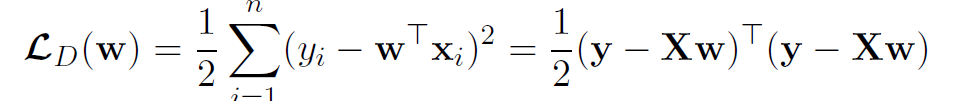
留出法

## 9. 模型参数的初始化方法:

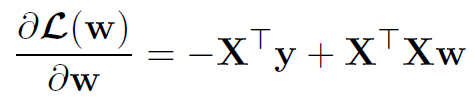
学习率为预先给定，权重（weights）初始化为0的矩阵

## 10.选择的loss函数及其导数:

（1）线性回归  
 loss函数：

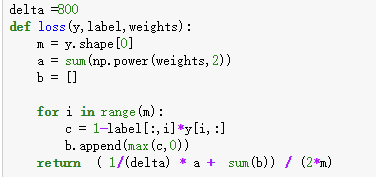


导数：

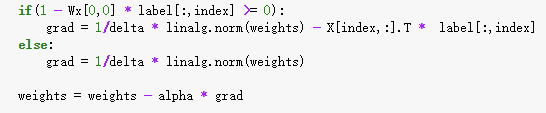


（2）线性分类（SVM）：

Loss函数：



导数(SGD)：

****

## 11.实验结果和曲线图:

## 超参数选择（η,epoch等）：

Eta = 0..01

C = 1000

## 评估结果（根据选择的评估方法）：

使用留出法，分别对train和test的样本集画出其loss的下降曲线，并对其进行分析。

## 预测结果（最佳结果）：

（1）linear regression:

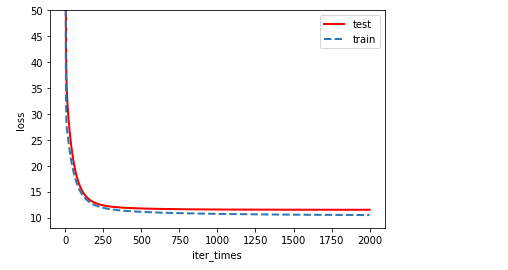
Loss = 10.364229694102526

(2) linear classification:

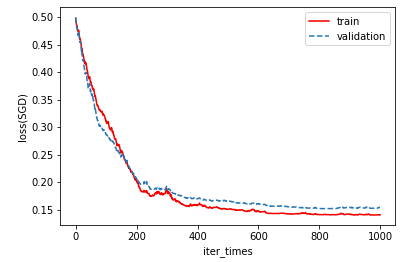
Loss= 0.14069433729818692

## loss曲线图：

***（1）linear regression:***



***(2)linear classification:***

******

## 12.实验结果分析:

从实验结果来看，训练集所对应loss函数普遍比测试集的loss函数要小，说明存在着一定的过拟合的情况。此外linear regression中的train和test的loss降不到接近于0的值，说明函数的复杂度还不够，可以考虑多加些X的不同次项，或者添加核。

## 13.对比线性回归和线性分类的异同点：

**相同点：**

在本次实验中

0.都是有监督学习（**Supervised learning**）

1.都需要对Loss function 进行处理，使其最小化，

2.均用到了梯度下降的做法（SVM可以考虑用SMO法更好训练和实现）

3.都使用了直线来进行拟合或者分类

**不同点**

1.线性回归作用于预测问题，而线性分类作用于分类问题

2.loss函数不同，导数不同

3.线性分类需要设置阈值，而线性回归在本次不需要

## 14.实验总结：

在进一步理解了线性回归和梯度下降的原理，体验了调参和训练过程，对机器学习的实践操作有了感性的认识和理解。