

**《机器学习》课程实验报告**

**学 院 软件学院**

**专 业 软件工程**

**组 员**  林俊钦

**学 号 201530612231**

**邮 箱 958008891@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，线性分类和梯度下降

## 2. 实验时间：2017年 12 月 8 日

## 3. 报告人: 林俊钦

## 4. 实验目的: 4.1进一步理解线性回归和梯度下降的原理。

## 4.2 在小规模数据集上实践。

## 4.3 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:

6.1 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。

6.2 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G。
6. 取梯度的负方向，记为D。
7. 更新模型参数，W (t)= w(t-1) +aD。a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。

## 7. 代码内容:

7.1 线性回归和梯度下降

def linear\_regression(alpha, n, X\_, y\_):

num\_col = X\_.shape[0]

num\_row = X\_.shape[1]

print(num\_col, num\_row)

w = np.random.normal(size=num\_row)

b = np.random.normal(size=1)

losses = []

for i in range(n):

grad\_w = np.random.normal(size=num\_row)

grad\_b = np.random.normal(size=1)

loss = 0

for j in range(num\_col):

y = np.dot(X\_[j][0].getA()[0], w) + b # 目标函数

loss += 0.5 \* (y-y\_[j])\*\*2 / num\_col

grad\_w += (y-y\_[j]) \* X\_[j][0].getA()[0] / num\_col

grad\_b += (y-y\_[j]) / num\_col

losses.append(loss)

w -= alpha \* grad\_w

b -= alpha \* grad\_b

print(loss)

return losses

7.2 线性分类与梯度下降

def linear\_classification(alpha, n, X\_, y\_):

num\_col = X\_.shape[0]

num\_row = X\_.shape[1]

print(num\_col, num\_row)

w = np.random.normal(size=num\_row)

b = np.random.normal(size=1)

losses = []

t = 0 # 阈值

for i in range(n):

grad\_w = np.random.normal(size=num\_row)

grad\_b = np.random.normal(size=1)

loss = 0

error = 0

C = np.power(2, 5)

for j in range(num\_col):

y = np.dot(X\_[j][0].getA()[0], w) + b # 目标函数

if y\_[j]\*y < 1 :

loss += C \* max(0, 1 - y\_[j] \* y) / num\_col

grad\_w += -C \* y\_[j] \* X\_[j][0].getA()[0] / num\_col

grad\_b += -C \* y\_[j] / num\_col

if y > t :

y = 1

else :

y = -1

if y != y\_[j]:

error += 1

losses.append(loss)

w -= alpha \* grad\_w

b -= alpha \* grad\_b

print(loss, 1-error/num\_col)

return losses

（针对线性回归和线性分类分别填写8-12内容）

## 8. 选择的评估方法（留出法，交叉验证，k折交叉验证等）:

留去法

## 9. 模型参数的初始化方法:

随机初始化

## 10.选择的loss函数及其导数:

Loss = 1/2 ∑(h(x)-y )̂ 2 / n

Losś = ∑(h(x)-y)\*h́(x) / n

## 11.实验结果和曲线图:

## 超参数选择（η,epoch等）：

η = 0.0006

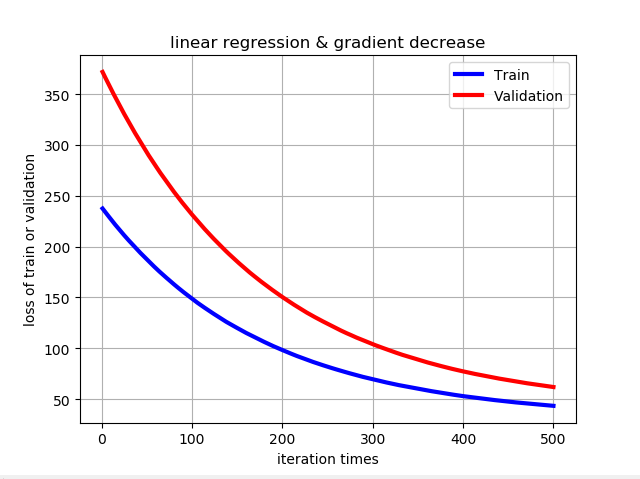
## 评估结果（根据选择的评估方法）：

在训练集和测试集上，随着训练次数的增加，loss 会减少；但测试集的loss明显高于训练集的loss,图形表现为测试集的损失函数在训练集的上方

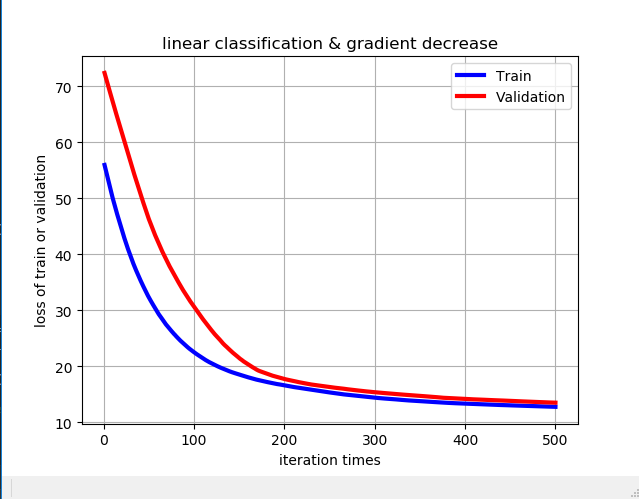
## 预测结果（最佳结果）：

训练集和测试集的loss都随训练次数的增加而降低，但二者的差距也随着训练次数的增加而减小。

## loss曲线图：



图一 线性回归和梯度下降



图二 线性回归和梯度下降

## 12.实验结果分析:

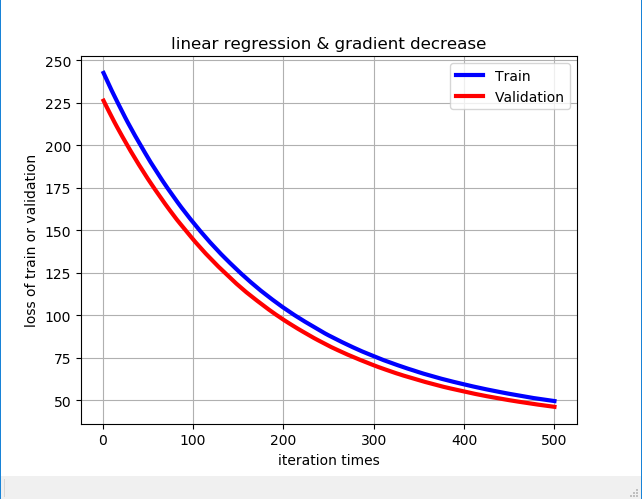
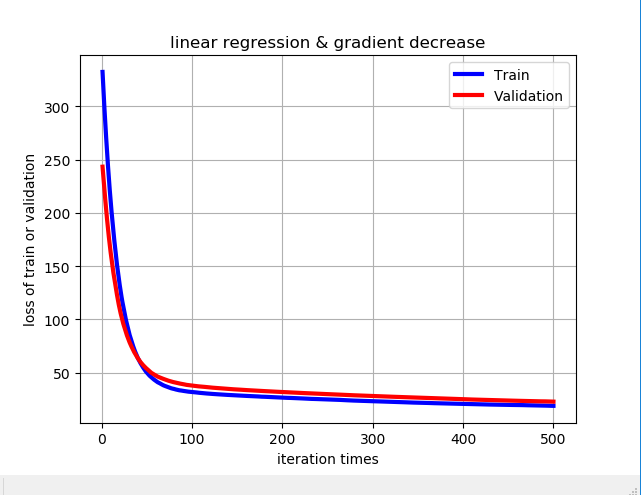
实验的结果跟预测的结果很好的符合，随着训练的次数的增加，loss值减少，由于训练集的数据大于测试集，故loss 的值较低。

发现一：

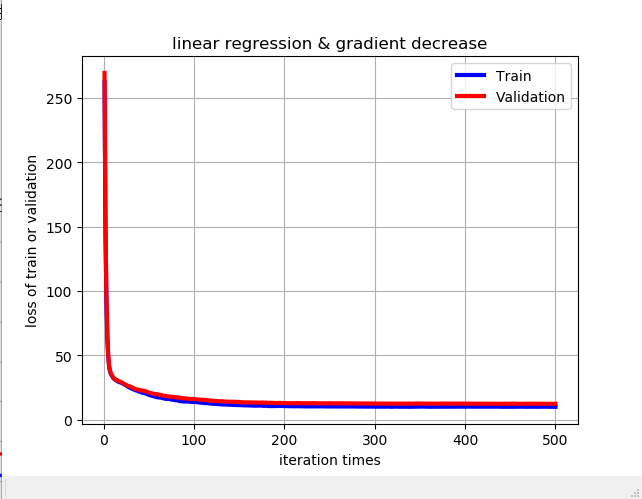
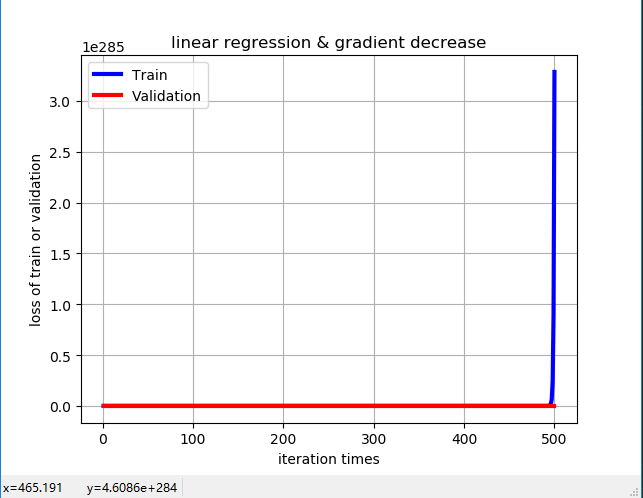
实验还发现不同的学习率也会对算法本身产生影响：

以线性回归为例：

不同的学习率对应的loss(训练集与测试集规模比 = 67:33)

图三 η = 0.0006 图四 η = 0.006

图五 η = 0.06 图六 η = 0.6

从图中的对比可以看出：梯度下降的学习率对算法本身影响很大，学习率 η 过小，则达到收敛所需的迭代次数会非常高（图一），如果学习率 η 过大，每次迭代可能不会减小代价函数，可能会越过局（图六）部最小值导致无法收敛

## 13.对比线性回归和线性分类的异同点：

相同点：两个问题本质上都是一致的，就是模型的拟合（匹配）

异同点：分类问题的y值(也称为label), 更离散化一些. 而且， 同一个y值可能对应着一大批的x, 这些x是具有一定范围的

回归问题的模型更倾向于 (很小区域内的x，或者一般是一个x) 对应着 (一个y).

14.实验总结：

通过本次实验，我更好的理解了线性回归和梯度下降的原理，进一步丰富在小规模数据集上实践的经验，也体会到在优化和调参的过程模型改进所带来的激动和喜悦，也有对莫名的结果的无助与苦思，希望通过努力，在机器学习的道路上越走越远。。。