

**《机器学习》课程实验报告**

**学 院 软件学院**

**专 业 软件工程**

**组 员**  **韩松岳**

**学 号 201536471115**

**邮 箱 han14122@163.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

## 2. 实验时间：2017年 12月 2 日

## 3. 报告人:韩松岳

## 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。

## 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G。

6.取梯度G的负方向，记为D。

7.更新模型参数，W = W + rate \* D。rate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L\_train，在验证集上测试并得到Loss函数值L\_validation。

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G的负方向，记为D。

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L\_train，在验证集上测试并得到Loss函数值L\_validation。

9.重复步骤5-8若干次，画出和随迭代次数的变化图。

10.整理实验结果并完成实验报告（实验报告模板将包含在示例仓库中）

## 7. 代码内容:

线性回归

def linear\_regression (X\_train,y\_train,X\_test,y\_test,w): #L2 loss

train\_loss = np.zeros(iter)

validation\_loss = np.zeros(iter)

for i in range(iter):

#trainLoss

deviation\_train = y\_train.reshape(y\_train.shape[0],1) - np.dot(X\_train, w.reshape(dimension,1))

for j in range(deviation\_train.shape[0]):

train\_loss[i] = train\_loss[i] + deviation\_train[j] \*\* 2

train\_loss[i] = train\_loss[i] / deviation\_train.shape[0]

#validationLoss

deviation\_test = y\_test.reshape(y\_test.shape[0],1) - np.dot(X\_test, w.reshape(dimension,1))

for j in range(deviation\_test.shape[0]):

validation\_loss[i] = validation\_loss[i] + deviation\_test[j] \*\* 2

validation\_loss[i] = validation\_loss[i] / deviation\_test.shape[0]

#gradient descent

G = -np.dot(deviation\_train.reshape(1,deviation\_train.shape[0]), X\_train / X\_train.shape[0])

w = w - rate \* G

return train\_loss,validation\_loss

线性分类

def SVM (X\_train, y\_train, X\_test, y\_test, w):

train\_loss = np.zeros(iter)

validation\_loss = np.zeros(iter)

#validation\_loss = np.zeros(iter)

for i in range(iter):

hinge\_loss\_train = hinge\_loss\_f(X\_train, y\_train, w)

train\_loss[i] = np.dot(w, w.reshape(dimension, 1)) / 2 + C \* hinge\_loss\_train

hinge\_loss\_validation = hinge\_loss\_f(X\_test, y\_test, w)

validation\_loss[i] = np.dot(w, w.reshape(dimension, 1)) / 2 + C \* hinge\_loss\_validation

#gradient descent

G = derivative(X\_train, y\_train, w)

w = w - rate \* G

return train\_loss,validation\_loss

def hinge\_loss\_f (X, y, w):

loss = 0

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

judge = y[i] \* np.dot(X[i], w.reshape(dimension, 1))

if (judge >= 1):

loss = loss + 0

else:

loss = loss + 1 - judge

loss = loss / X.shape[0]

return loss

def derivative (X, y, w):

g = np.zeros(X.shape[1])

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

judge = y[i] \* np.dot(X[i], w.reshape(dimension, 1))

if (judge < 1):

g = g - y[i] \* X[i]

G = w + C \* g

return G

（针对线性回归和线性分类分别填写8-12内容）

## 选择的评估方法（留出法，交叉验证，k折交叉验证等）:

## 线性回归：留出法（训练集：验证集 = 2：1）

## 线性分类：留出法（训练集：验证集 = 3：1）

## 模型参数的初始化方法:

## 线性回归：正态分布初始化

线性分类：正态分布初始化

## 选择的loss函数及其导数:

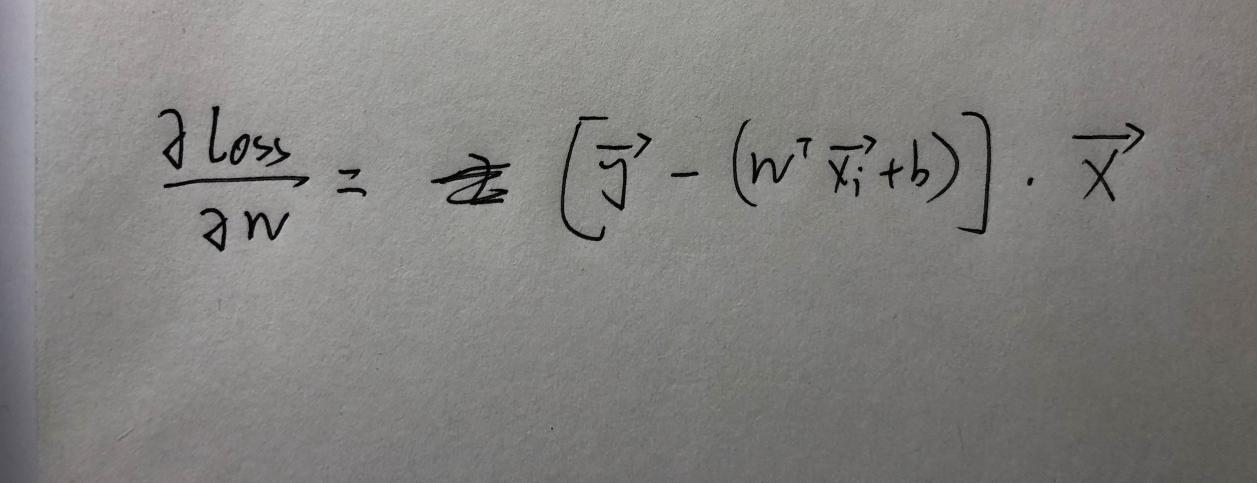
loss函数

线性回归：L2 loss

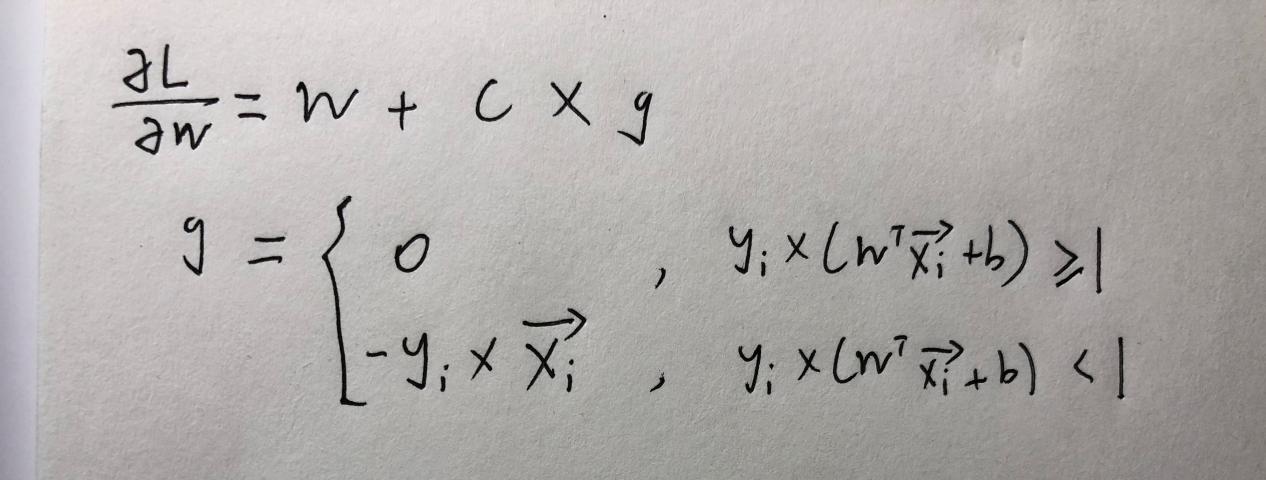
线性分类：L1 hinge loss

导数：

线性回归：



线性分类：



## 11.实验结果和曲线图:

## 超参数选择（η,epoch等）：

线性回归：步长η = 0.1，迭代次数epoch = 100

线性分类：步长η = 0.0005，迭代次数epoch = 200，C = 1

## 评估结果（根据选择的评估方法）：

线性回归：L\_train = 25.77

线性分类：L\_train = 1.04

## 预测结果（最佳结果）：

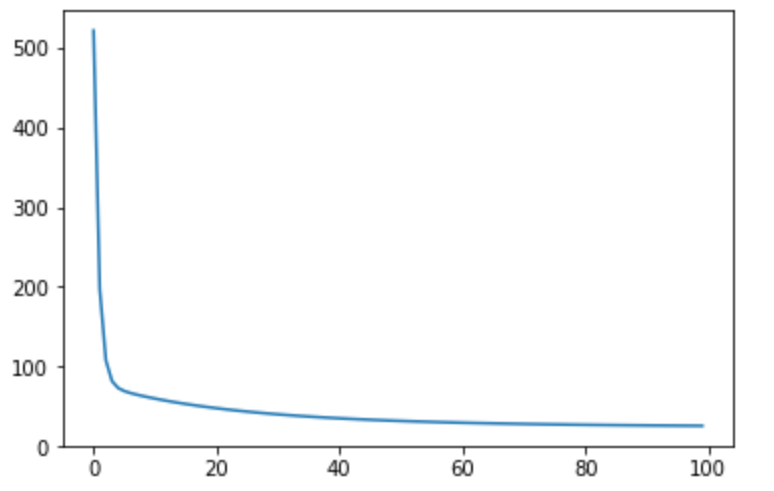
线性回归：L\_validation = 24.33

线性分类：L\_validation = 1.05

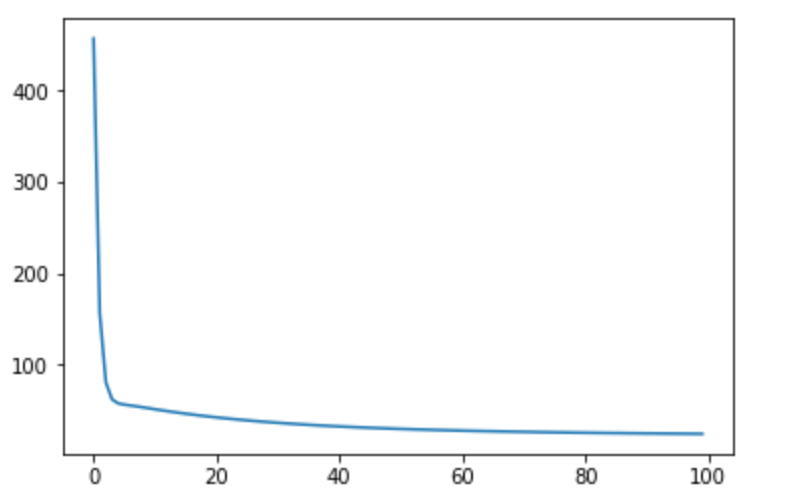
## loss曲线图：

线性回归

L\_train:

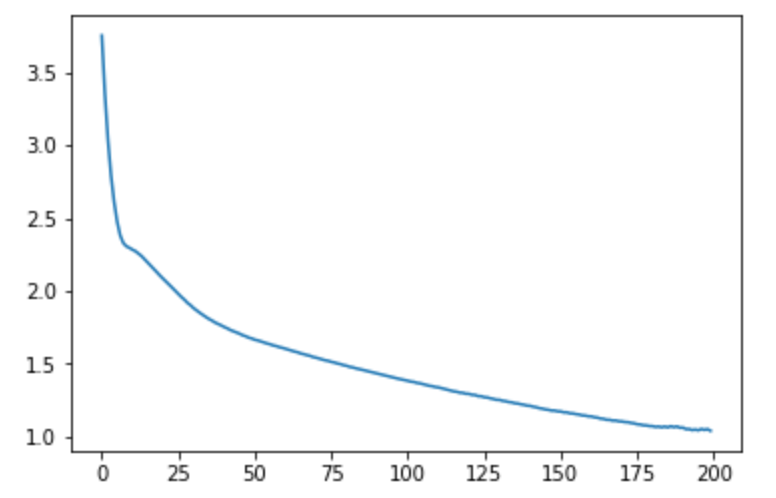


L\_validation:

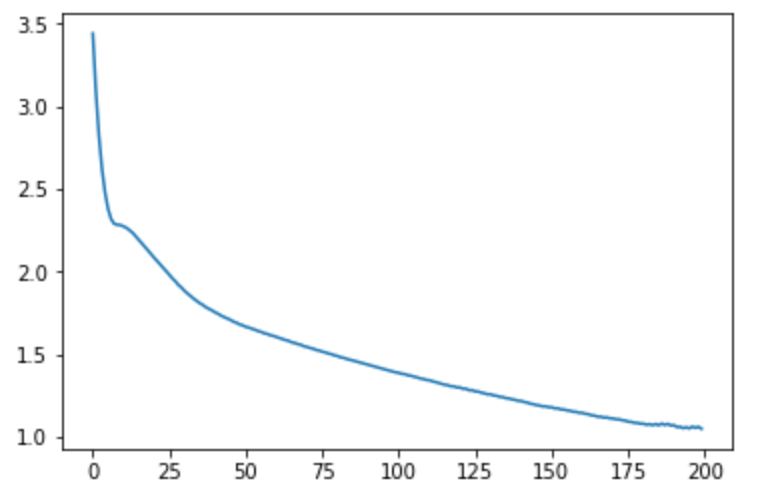


线性分类

L\_train:



L\_validation:



## 实验结果分析:

线性回归：

训练集与验证集表现相近，体现出数据分布的一致性。前5次迭代loss快速下降，随着迭代次数增长，loss下降速率快速减缓。实验中发现100次迭代以后即使减小步长η，loss也基本不再下降，维持在25以上（**使用L2 loss**）。这应该是由数据本身分布所导致的。

线性分类：

训练集与验证集表现相近，体现出数据分布的一致性。前10次迭代种loss快速下降，随着迭代次数增加，loss减小速度减缓。在调节参数适当后，线性分类loss可降低至1.0左右（L1 hinge loss），实验中发现若以0为阈值，最高约有85%的分类准确率。但在迭代200次以后，即使减小步长η也无法再提高准确率，减小loss。这应该是由数据本身分布所导致的。

## 对比线性回归和线性分类的异同点：

线性回归和线性分类均使用梯度下降算法迭代逼近最优解

线性回归使用线性回归算法，线性分类使用SVM hinge loss算法。

## 14.实验总结：

本次实验大大加深了我对于线性回归，支持向量机算法的理解。同时令我学习了python，numpy，matplotlib的使用。锻炼了我的工程能力。提高了我解决实际问题的能力，如求解梯度时如果完全按照公式计算会得到过于大的数字导致溢出，通过进行适当的等比例缩小保证正常计算。