

**《机器学习》课程实验报告**

**学 院 软件学院**

**专 业 软件工程**

**组 员**  **温志全**

**学 号 201530612996**

**邮 箱 957918462@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

## 2. 实验时间：2017年 12 月 2 日

## 3. 报告人: 温志全

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。**

## 数据集以及数据分析：

**线性回归使用的是[LIBSVM Data](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/" \t "_blank)中的[Housing](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/regression.html" \l "housing" \t "_blank)数据，包含506个样本，每个样本有13个属性。并将其切分为训练集，验证集。**

**线性分类使用的是[LIBSVM Data](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/" \t "_blank)中的[australian](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html" \l "australian" \t "_blank)数据，包含690个样本，每个样本有14 个属性。并将其切分为训练集，验证集。**

## 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](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" \t "_blank)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" \t "_blank)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。**

***线性回归***

## 7. 代码内容:

（针对线性回归和线性分类分别填写8-12内容）

**import** numpy **as** np  
**import** matplotlib.pyplot **as** plt  
**from** sklearn.datasets **import** load\_svmlight\_file  
**from** sklearn.model\_selection **import** train\_test\_split  
  
data = load\_svmlight\_file(**'./housing\_scale'**)  
another2 = np.ones(shape=[506,1])  
x\_train,x\_validate,y\_train,y\_validate = train\_test\_split(data[0],data[1],test\_size=0.33)  
x\_train = x\_train.todense()  
x\_validate = x\_validate.todense()  
y\_train = np.mat(y\_train)  
y\_validate = np.mat(y\_validate)  
another = np.ones(shape=[339,1])  
another1 = np.ones(shape=[167,1])  
x\_train = np.hstack((x\_train,another))  
x\_validate = np.hstack((x\_validate,another1))  
init\_w = np.zeros(shape=[14,1])  
  
train\_list=[]  
test\_list=[]  
  
**def** train(num\_iter):  
 learning\_rate = 0.0001  
 w = init\_w  
 gradient = -2 \* np.dot(x\_train.T, (np.dot(x\_train, init\_w)) - data[1].T)  
 **for** i **in** range(num\_iter):  
 w = w + learning\_rate \* gradient  
 loss = 0.5 \* 1/339 \* np.dot((y\_train.T - np.dot(x\_train, w)).T, y\_train.T - np.dot(x\_train, w))  
 loss1 = 0.5 \*1/167 \* np.dot((y\_validate.T - np.dot(x\_validate, w)).T, y\_validate.T - np.dot(x\_validate, w))  
 gradient = -2 \* np.dot(x\_train.T, (np.dot(x\_train, w)) - data[1].T)  
 z = loss.tolist()  
 q = loss1.tolist()[0][0]  
 train\_list.append(z[0][0])  
 test\_list.append(q)  
  
 plt.plot(np.arange(0,num\_iter),train\_list,label=**u'train'**)  
 plt.plot(np.arange(0,num\_iter),test\_list,label=**u'validate'**)  
 plt.legend()  
 plt.show()  
  
train(1000)

## 8. 选择的评估方法（留出法，交叉验证，k折交叉验证等）:交叉验证

## 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:

**导数 ： 2X^T（Xw - y）**

**Loss：L(w) = 1/2（y-Xw）^T（y-Xw）**

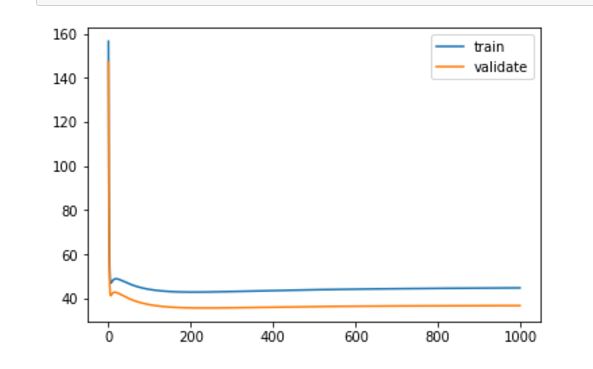
## 11.实验结果和曲线图:

## 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：



## 12.实验结果分析:

由图可知，经过训练，随着梯度的下降，train loss 与 test loss 在初始是下降的很快，接近垂直，但到达一定值后就不怎么下降了，并出现接近平行的情况，可见，loss已经降到相对最低点了，达到局部最优了

***线性分类***

## 7. 代码内容:

（针对线性回归和线性分类分别填写8-12内容）

**import** numpy **as** np  
**from** sklearn.datasets **import** load\_svmlight\_file  
**from** sklearn.model\_selection **import** train\_test\_split  
**import** matplotlib.pylab **as** plt  
  
data = load\_svmlight\_file(**'./australian\_scale'**)  
x\_train,x\_validate,y\_train,y\_validate = train\_test\_split(data[0],data[1],test\_size=0.33)  
x\_train = x\_train.todense()  
x\_validate = x\_validate.todense()  
init\_w = np.zeros(shape=[15,1])*#将b塞进w*another = np.ones(shape=[462,1])  
another1 = np.ones(shape=[228,1])  
x\_train = np.hstack((x\_train,another))  
x\_validate = np.hstack((x\_validate,another1))*#分别将训练集与验证集加多一列，全为1*c = 0.4  
learning\_rate = 0.0001  
  
w = init\_w  
sum = np.zeros(shape=[1,15])  
round = 500  
loss\_train = []  
loss\_test = []  
  
**for** i **in** range(round):  
 sum = 0  
 **for** j **in** range(x\_train.shape[0]):  
 **if** 1 - (y\_train[j] \* (np.dot(w.T, x\_train[j].T))).tolist()[0][0] >= 0:  
 g\_w = -np.dot(np.mat(y\_train[j]),np.mat(x\_train[j]))  
 sum += g\_w  
  
 w = w - (learning\_rate \* (w + (c \* sum).T))  
  
 *#Hinge LOss 中的||w||平方* total\_w = np.dot(w.T,w)  
  
  
 total\_ao\_train=0  
 total\_ao\_test=0  
 *#Hinge Loss 中的另一部分* **for** z **in** range(x\_train.shape[0]):  
 total\_ao\_train += max(0,1 - (y\_train[z] \* (np.dot(w.T, x\_train[z].T))).tolist()[0][0])  
  
 **for** n **in** range(x\_validate.shape[0]):  
 total\_ao\_test += max(0,1 - (y\_validate[n] \* (np.dot(w.T, x\_validate[n].T))).tolist()[0][0])  
  
 hinge\_loss\_train = 0.5 \*1/462 \* total\_w[0] + c \* total\_ao\_train  
 hinge\_loss\_test = 0.5 \*1/228\* total\_w[0] + c \* total\_ao\_test  
  
 loss\_train.append(hinge\_loss\_train.tolist()[0][0])  
 loss\_test.append(hinge\_loss\_test.tolist()[0][0])  
print(loss\_train)  
print(loss\_test)  
  
plt.plot(np.arange(0,round),loss\_train,label=**u'train'**)  
plt.plot(np.arange(0,round),loss\_test,label=**u'validate'**)  
plt.legend()  
plt.show()

## 8. 选择的评估方法（留出法，交叉验证，k折交叉验证等）:交叉验证

## 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:

导数：w + C \* 累加（gw（i））

Loss ： hinge loss ：1/2w^2 + C \* 累加（max（0,1-y(i)(w^Tx(i) + b））

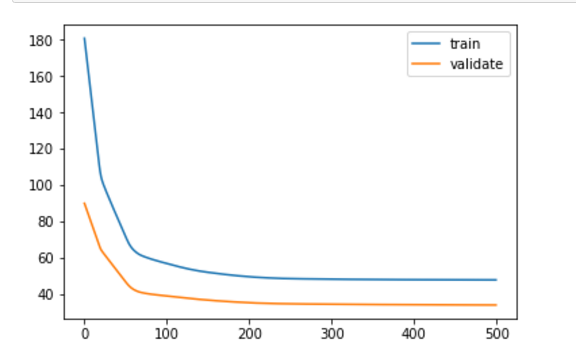
## 11.实验结果和曲线图:

## 超参数选择（η,epoch等）：η= 0.0001 ,C = 0.4

## 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：



## 12.实验结果分析:

由图可知，经过训练，随着梯度的下降，train loss 与 test loss 在初始是下降的很快，但到达一定值后就不怎么下降了，并出现接近平行的情况，可见，loss已经降到相对最低点了，达到局部最优了

## 13.对比线性回归和线性分类的异同点：

同：使用线性模型加上梯度下降的方法进行优化模型，使模型达到局部最优化

异：loss函数表达方式不一样，线性分类用了hinge loss ，而线性回归用了square loss，同时，梯度的表达方式也不一样，线性分类梯度求导方法通过判断是否分类成功有不同的表示，而线性回归梯度相对比较容易表示

## 14.实验总结：

通过这次实验让我深刻理解了通过梯度下降的方式进行线性回归，线性分类含义，让我们不局限于书本知识，让我们实践，更容易理解这些抽象的概念。