

**《机器学习》课程实验报告**

**学 院 软件学院**

**专 业 软件工程**

**组 员**  **文永亮**

**学 号 201530613023**

**邮 箱 [471814775@qq.com](mailto:471814775@qq.com)**

**指导教师**  **吴庆耀**

**提交日期** **2017年 12 月 02 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 02 日

## 3. 报告人:文永亮

## 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 "/home/todd/Documents\\x/_blank)中的[Housing](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/regression.html" \l "housing" \t "/home/todd/Documents\\x/_blank)数据，包含506个样本，每个样本有13个属性。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 "/home/todd/Documents\\x/_blank)中的[australian](https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html" \l "australian" \t "/home/todd/Documents\\x/_blank)数据，包含690个样本，每个样本有14 个属性。将其切分为训练集，验证集。

## 实验步骤:

（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 "/home/todd/Documents\\x/_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 "/home/todd/Documents\\x/_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。

（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 "/home/todd/Documents\\x/_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 "/home/todd/Documents\\x/_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。

## 代码内容:

1. **线性回归和梯度下降**

线性回归梯度下降关键代码：

|  |
| --- |
| for i in range(n\_estimator):  tloss=loss\_fuc(x, y, w)  print ("[The %d iteration in train]: %f" % (i,tloss))  vloss=loss\_fuc(xvali,yvali,w)  print("[The %d iteration in validation]: %f" % (i,vloss))  loss\_train.append(tloss)  loss\_vali.append(vloss)  pred=np.dot(x,w)  w\_gradient = np.dot(x.T, pred - y)/xlen  #print (np.shape(w\_gradient))  w=w-lr\*w\_gradient |

1. **线性分类和梯度下降**

线性分类梯度下降关键代码：

|  |
| --- |
| n\_sample,n\_feature=np.shape(x)  n\_vali\_sample,n\_vali\_feature=np.shape(xvali)  loss\_train=[]  loss\_vali=[]  for i in range(n\_estimator):  # Hinge loss:  tloss = 0.0  vloss=0.0  grad\_w = 0.0  grad\_b = 0.0  for \_x, \_y in zip(x, y):  tmp = 1 - (\_y \* np.dot(\_x, w) + b)  if tmp >= 0:  tloss += tmp  grad\_w += ((-\_y) \* \_x).T  grad\_b += (-\_y)  else:  tloss += 0.0  grad\_w += 0.0  grad\_b += 0.0  for \_vx,\_vy in zip(xvali,yvali):  vtmp = 1 - (\_vy \* np.dot(\_vx, w) + b)  if vtmp >= 0:  vloss += vtmp  else:  vloss += 0.0  norm=np.linalg.norm(w)\*\*2  tloss+=norm/2.0  vloss+=norm/2.0  tloss=tloss.mean()/n\_sample  vloss=vloss.mean()/n\_vali\_sample  grad\_w=(w+C\*grad\_w)  grad\_b=grad\_b\*C  print ("[The %d iteration in train]: %f" % (i,tloss))  print("[The %d iteration in validation]: %f" % (i,vloss))  loss\_train.append(tloss)  loss\_vali.append(vloss)  #print ("pred:",np.shape(pred))  # delta=pred - y  #print ("delta:",np.shape(delta))  #print (np.shape(w\_gradient))  w=w-lr\*grad\_w/n\_sample  b=b-lr\*grad\_b/n\_sample |

## 选择的评估方法（留出法，交叉验证，k折交叉验证等）:

线性回归和分类都选择留出法，这次使用[train\_test\_split](http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html" \t "/home/todd/Documents\\x/_blank)函数切分数据集，0.25比例为验证集，0.75比例为训练集。

|  |
| --- |
| X\_train,X\_vali,y\_train,y\_vali=sklearn.model\_selection.train\_test\_split(X, y, random\_state=1000) |

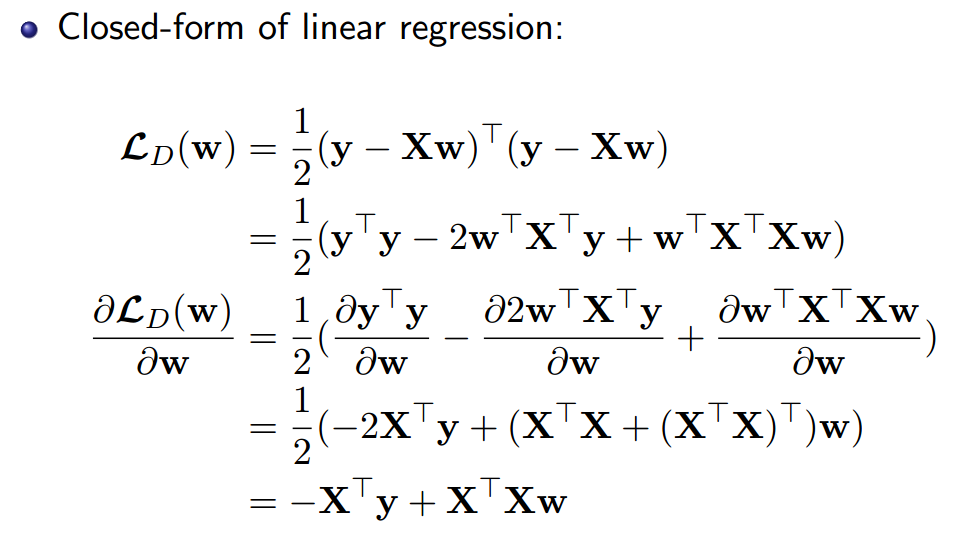
## 模型参数的初始化方法:

线性回归和分类都使用了全0初始化。

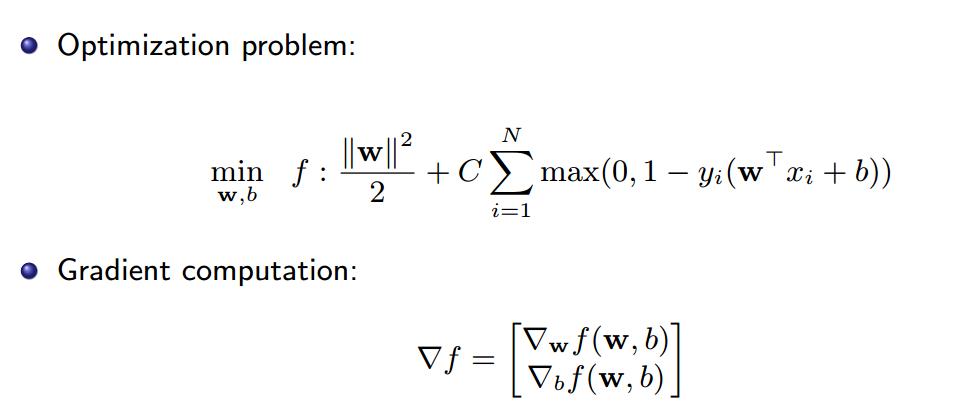
|  |
| --- |
| w0=np.zeros(shape=(n\_feature,1)) |

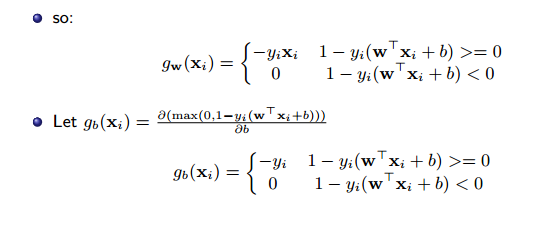
## 选择的loss函数及其导数:

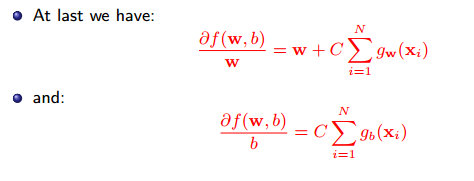
线性回归：



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：

|  |
| --- |
| lr=0.001#学习率  w0=np.zeros(shape=(n\_feature+1,1)))#w的全零初始化  n\_estimator=10000#迭代次数 |

线性分类：

|  |
| --- |
| lr=0.1#学习率  w0=np.zeros(shape=(n\_feature,1))#w的全零初始化  b0=0.0#b的初始化  C=0.5#松弛系数  n\_estimator=400#迭代次数 |

## 评估结果（根据选择的评估方法）：

线性回归：

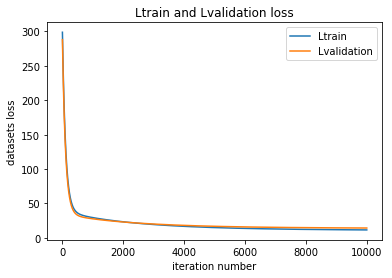
|  |
| --- |
| [The 9999 iteration in train]: 11.625240  [The 9999 iteration in validation]: 14.379286 |

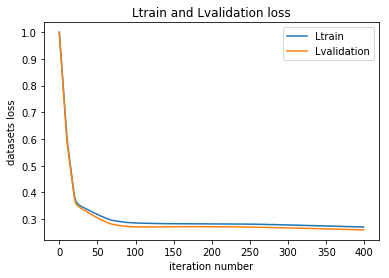
线性分类：

|  |
| --- |
| [The 399 iteration in train]: 0.270163  [The 399 iteration in validation]: 0.259784 |

## 预测结果（最佳结果）：

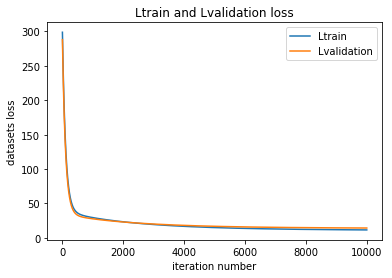
线性回归：

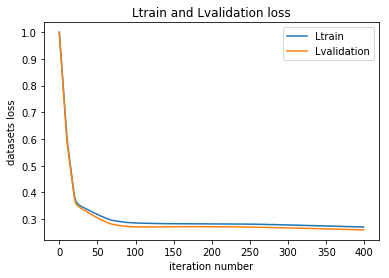
线性分类：



## loss曲线图：

线性回归：

线性分类：



## 实验结果分析:

实验结果误差不算大，但是验证集上的损失值比训练集上的小，可能存在一定的问题。

## 13.对比线性回归和线性分类的异同点：

机器学习中最基本的两类问题，线性回归和分类。 能让大家有个清晰的思想，对于这两类问题都有以下几个步骤,

1. 如何选取一个 合理的模型(线性的，or 非线性的(e.g. 阶跃函数， 高斯函数)).
2. 制造一个"美好"的 误差函数 (可以评估拟合程度，而且还是convex函数)
3. 采取一切可能的技术(e.g. 导数下降法，解极值方程法) 求出最好的模型参数

谈谈回归和分类的区别：

总的来说两个问题本质上都是一致的，就是模型的拟合（匹配）。 但是分类问题的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。

所以分类问题更多的是 (一定区域的一些x) 对应 着 (一个y).   而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：

学习了如何实现线性回归和线性分类的代码，以及矩阵之间的变换，关于Hinge loss以及正则化有一定的理解，体会到验证集和训练集的差异。