**华中科技大学**

**计算机科学与技术学院**

**《机器学习》结课报告**



专 业： 计算机科学与技术

班 级： cs2003

学 号： U202015360

姓 名： 胡沁心

成 绩：

指导教师： 张腾

**完成日期： 2022年 7 月 6 日**

目录

[1. 实验要求 2](#_Toc108028801)

[2. 算法设计与实现 2](#_Toc108028802)

[2.1 数据预处理 2](#_Toc108028803)

[2.2 对数几率回归实现 2](#_Toc108028804)

[2.3 决策树桩实现 3](#_Toc108028805)

[2.4 Adaboost算法实现 5](#_Toc108028806)

[3. 实验环境与平台 6](#_Toc108028807)

[4. 结果与分析 6](#_Toc108028808)

[5. 个人体会 8](#_Toc108028809)

[6. 问题回答 9](#_Toc108028810)

Adaboost算法实现

# 实验要求

分别实现以对数几率回归和决策树桩为基分类器的AdaBoost 算法。读取 data.csv 及 targets.csv 两个文件，并输出在不同数目基分类器条件下的 10 折交叉验证的预测结果至 experiments/base#\_fold#.csv。基分类器数目取 1 5 10 100 这四种数值。

# 2. 算法设计与实现

## 2.1 数据预处理

1. 将csv文件的内容以header=None的形式读到矩阵中。
2. 将data和targets的矩阵大小，以及十折交叉验证的矩阵的大小记录到self中，以便之后引用。data大小为3680\*57，targets大小为3680\*1;十折验证的data大小为3312\*57，targets大小为368\*1。令m=3680，n=57，x=3312，y=368。
3. 如果以对数几率回归作为基分类器，需要将数据进行归一化处理。归一化公式为x=(x-min)/(max-min)
4. 十次十折交叉验证的要求，将数据按顺序分为9和1，以9为训练样本，1为预测样本。

## 2.2 对数几率回归实现

介绍：

对数几率回归，简称对率回归，又称逻辑回归，是使用Sigmoid函数作为联系函数时的广义线性模型。可使用梯度下降法、牛顿法等求损失函数最小时的参数值。梯度下降是迭代法的一种,可以用于求解最小二乘问题。在求解机器学习算法的模型参数时，梯度下降是最常采用的方法之一。在求解损失函数的最小值时，可以通过梯度下降法来一步步的迭代求解，得到最小化的损失函数和模型参数值。

一、训练函数logical\_regression的说明

1)参数：data(训练样本)

label (训练标签)

D(权重)

alpha(步长)5.8

epochs(迭代次数)15000

epsilon(容错率)1e-5

2)算法：

1. 初始化梯度矩阵w全部为1，矩阵大小为x\*1
2. 预测函数h初始化
3. 开始迭代：
   1. 计算损失函数loss，将adaboost中的权重D乘到损失函数中，得到基于之前的模型训练结果的更精确的数据。
   2. 梯度下降计算
   3. 按新的梯度重新计算h
   4. 若损失函数的模小于容错率，则结束迭代
4. 返回梯度矩阵

3)伪代码：

w <- 1 # 初始化梯度

h = sigmoid(data \* w) # 预测函数

for i = 0 to epochs-1

loss = (h - label) \* D # 损失函数

w -= alpha \* (data.T \* loss) # 梯度下降

h = sigmoid(data \* w) # 计算新的预测函数

if alpha \* sum(abs(loss)) < epsilon # 损失函数的模小于容错值，结束

break

return w

二、错误率计算函数cal\_err0

1)参数：r(预测结果)

label(标签)

d(权重)

2)算法：

1. 初始化err=0
2. 以j作为下标，遍历比较预测结果和标签，若不相等，err加上当前样本的权重，即err+=d[j]。
3. 返回err

三、超参数的设置

先将迭代次数设置为15000(为了使运行时间不要太长)，令alpha为2^n，alpha越大，梯度下降越快。从n=-3开始尝试，试到n>=3时正确率开始降低，因此当epochos=15000时，alpha值应在4-8之间。

在这个范围内，发现当基分类器个数为100，alpha为5.8-5.9时正确率最高。因此alpha取5.8或其整数分之一倍，设置alpha为5.8，迭代次数为15000(也可以设置alpha为2.9，epochos为30000或以上，或alpha更小epochos更大，正确率有提高但运行时间长一些)。

## 2.3 决策树桩实现

介绍：

决策树桩（decision stump），也称单层决策树，它是一种简单的决策树，通过定的阈值进行分类。从实际意义上来看，决策树桩根据一个属性的单个判断(但是实际上待判断的物体具有多个属性)就确定最终的分类结果。这种特性比较适合做集成学习中的弱学习器，因为其至少比随机的效果好一些，又计算较为容易。

一、训练函数dicision\_stump

1)参数：data(训练样本)

label(训练标签)

D(权重)

2)算法：

每个分类器由4个值构成：feature (特征序号)

split (分界值)

symbol (划分依据，1表示大于，-1表示小于)

err(错误率)

1. 给每一个特征(feature)都去除重复特征值，然后升序排列。这样是为了得到每个feature的特征值的升序排列。
2. 取每两个相邻特征值的平均值作为分界值(split)。如果大于split分类为1，则symbol取1，否则取-1。用这个分类器对data进行分类，计算错误率(err)。
3. 对每一个feature遍历特征值，取err最小的分类器，作为当前feature的分类器。用变量cnt计数，如果在后5个分类器之内出现了更小的err，将cnt清零：如果该err比总错误率小，更新树桩。若后5个分类器都没有出现更小的err，就结束当前feature的特征值的遍历，开始遍历下一个feature的特征值（这样是为了缩短运行时间，验证后发现对正确率影响不大）。
4. 返回树桩

3)伪代码：

stump = {} # 决策树桩

minErr = 1 # 总最小错误率

x = data.T

for feature = 0 to y-1

c = list(set(x[feature])) # 去掉重复特征值

c.sort(reverse=False) # 升序

m = c.shape # 不重复的特征值的个数

cnt = 0

min\_err = 1 # 当前feature的最小错误率

for j = 0 to m-2

split = (c[j] + c[j + 1]) / 2 # 分界值

err, symbol = cal\_err # 计算当前错误率

if err < min\_err # 当前feature错误率最小的分类器

cnt = 0

min\_err = err

if err < minErr # 总错误率最小的分类器

minErr = err

stump <- 当前feature的分类器

else

cnt += 1

if cnt == 5 #当前feature中，5个之内没有出现更小的错误率

break

return stump

二、错误率计算函数cal\_err1

1)参数：r(预测结果)

label(标签)

d(权重)

2)实现过程：

1. 初始化err=0
2. 假设：当特征值大于split时预测结果为1(即symbol=1)。以j作为下标，遍历比较预测结果和标签，若不相等，err加上当前样本的权重，即err+=d[j]。
3. 如果err>0.5，说明当symbol为-1时错误率更小，将symbol设置为-1，err=1-err。
4. 返回err和symbol。

## 2.4 Adaboost算法实现

介绍：

adaboost算法的工作机制是先在初始训练集上训练出一个基学习器，再根据基学习器的表现对训练样本的权重分布进行调整，使得先前基学习器预测错的样本在后续受到更多关注，然后基于调整后的权重分布训练下一个基学习器；如此重复进行，直至基学习器数目达到事先指定的值T，最终将这T 个基学习器进行加权线性组合。

一、首先说明一下pre函数：

1)功能：分类器将当前这组样本的预测结果为1时返回1，为0时返回-1。该函数在adaboost和predict函数中会用到。

2)参数：h(分类器)

data(训练样本)

3)实现过程：

如果base==1(决策树桩)：如果symbol==1，当分类器中指定的特征值>=split时返回1，否则返回-1。如果symbol==-1，则相反。

如果base==0(对数几率回归)：sigmoid(data \* h['w'])>=0.5时返回1，否则返回-1。

二、训练函数adaboost的实现过程

1)参数：T(基分类器个数)

data(训练样本)

label(训练标签)

2)算法：

1. 初始化权重矩阵都为1/x，矩阵大小为x\*1，基分类器为h
2. 开始生成基分类器。如果base=0，选对数几率回归，base=1选决策树桩
3. 用基分类器得到预测结果
4. 计算错误率。错误率的计算方法在上面两个基分类器的部分已经分别说明。如果错误率大于0.5，就结束循环。如果选用决策树桩，将这个值改为0.45，可以使运行时间减少且对正确率影响不大。
5. 计算权重系数
6. 根据预测结果更新权重
7. 归一化权重
8. 在h中添加该基分类器
9. 返回h

3)伪代码：

Dt = 1 / x # 初始化权重

h = [] # 基分类器

for t = 0 to T-1

r = [] # 预测结果

if base == 1:

classifier = decision\_stump(data, label, Dt) # 决策树桩

for j = 0 to x

r[j] = (pre(classifier, data[j]) + 1) / 2

err = classifier['err'] # 决策树桩的错误率在生成时已计算好

else:

classifier = logistic\_regression(data, label, Dt) # 逻辑回归

r = sigmoid(data \* classifier['w']) >= 0.5

err = cal\_err0(r, label, Dt) # 计算错误率

if err >= 0.5

break

alpha = log((1 - err) / err) / 2 # 计算权重系数

for j = 0 to x-1

if(label[j] == r[j])

Dt[j] \*= exp(-alpha) # 根据预测结果更新权重

else

Dt[j] \*= exp(alpha)

for j = 0 to x-1

Dt[j] /= sum(Dt) # 归一化权重

classifier['alpha'] = alpha

h[t] = classifier

return h

三、adaboost生成的分类器的预测过程

1)参数：x\_file(需要预测的样本文件)

2)实现过程：

1. 如果基分类器是对数几率回归，需要对数据进行归一化处理。
2. 对于每一组样本，将每一个基分类器的权重系数与分类结果(1则为1，0则为-1)相乘，作为一次分类依据，加到参考变量r中。
3. 如果r>=0预测结果为1，否则为0。
4. 返回预测结果

3)伪代码：

if base == 0

data = normalization(data) #选对数几率回归，需要归一化数据

for i = 0 to m-1

r = 0 # 参考变量

for j in classifier

r += j['alpha'] \* pre(j, data[i]) # 权重系数与分类结果的乘积

result[i] = (r >= 0) # 预测结果

return result

# 3. 实验环境与平台

操作系统：Windows 11 家庭中文版

Python版本：2.7

IEDA：PyCharm Community Edition 2022.1.2

处理器：Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz 2.30 GHz

# 4. 结果与分析

表1 对数几率回归 十折交叉验证及预测结果

|  |  |  |
| --- | --- | --- |
| 基分类器个数 | 十折交叉验证的平均正确率 | 最终预测结果的正确率 |
| 1 | 0.714 | 0.855 |
| 5 | 0.711 | 0.855 |
| 10 | 0.750 | 0.876 |
| 100 | 0.806 | 0.874 |

alpha偏高偏低都会导致结果正确率下降。若将alpha除以整数倍，并加倍迭代次数，可以提高结果的正确率，但运行时间会变长。(比如设置alpha为2.9，epochos为30000或以上，或alpha更小epochos更大)。在错误率小于容错率之前，迭代次数越高正确率越高，但运行时间变长。

表2 决策树桩 十折交叉验证及预测结果

|  |  |  |
| --- | --- | --- |
| 基分类器个数 | 十折交叉验证的平均正确率 | 最终预测结果的正确率 |
| 1 | 0.681 | 0.792 |
| 5 | 0.846 | 0.885 |
| 10 | 0.879 | 0.905 |
| 100 | 0.893 | 0.930 |

生成决策树桩的操作是将特征值升序排列，取两个特征值的平均数作为分界值。大于或小于该分界值的分类情况按照计算的错误率来确定，这两个错误率的和为1，所以选小于错误率小于0.5的分类依据，可以提高结果的正确率。

基分类器数目越大，十折交叉验证和最终预测结果的平均正确率基本就越高，但并不一定。因为十折交叉的正确率仅仅是针对十分之一的样本的正确率，选取十折交叉验证中正确率最高的模型，用该模型去分类所有样本，可能会有较大误差。

以下是不同基分类器个数下预测结果的可视化图，红色为1，蓝色为0(如果看的是纸质版，y=0以上是1，以下是0)。

对数几率回归：

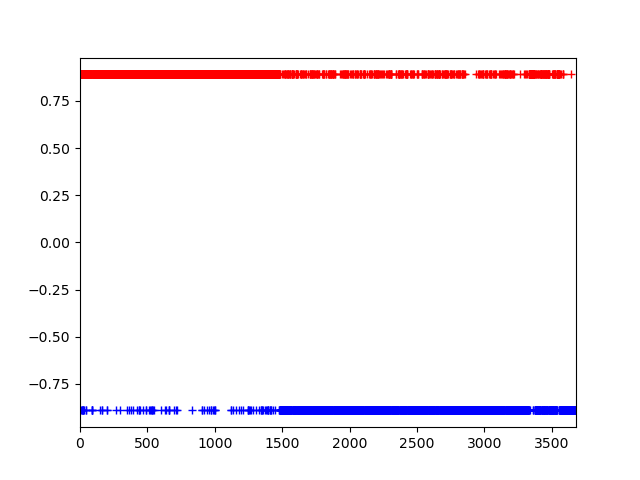


图1 基分类器1

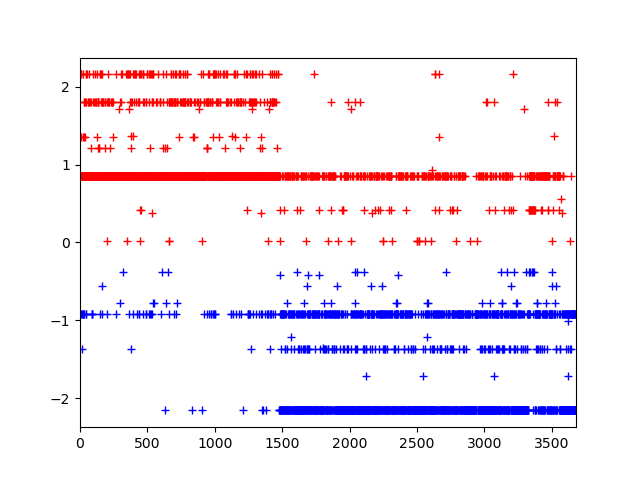


图2 基分类器5

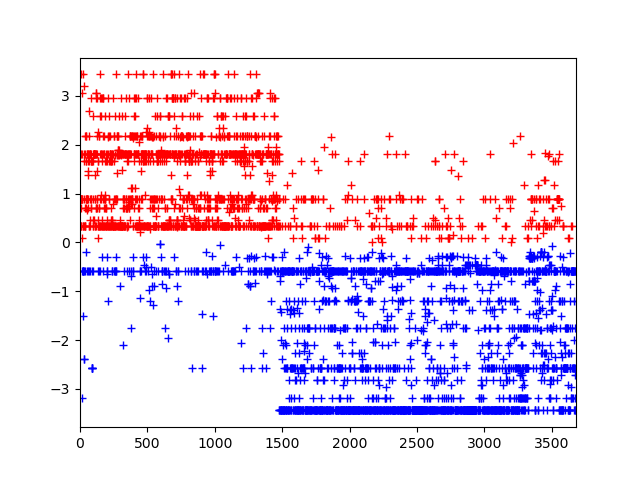
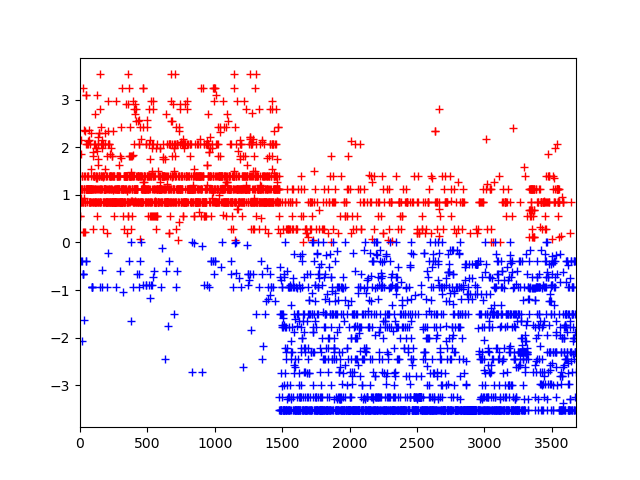


图3 基分类器10

图4 基分类器100

决策树桩：

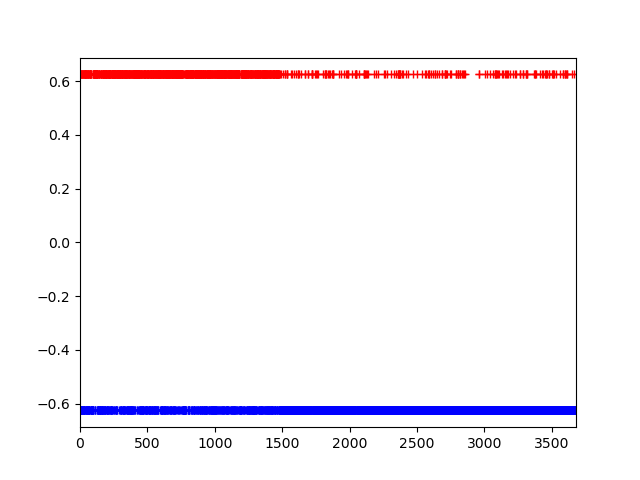


图5 基分类器1

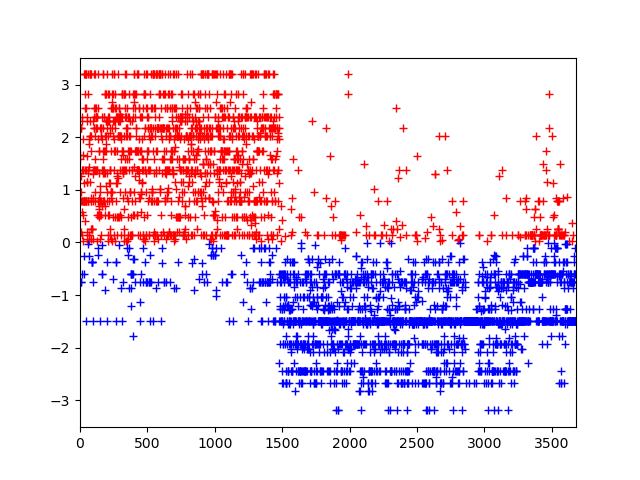


图7 基分类器10

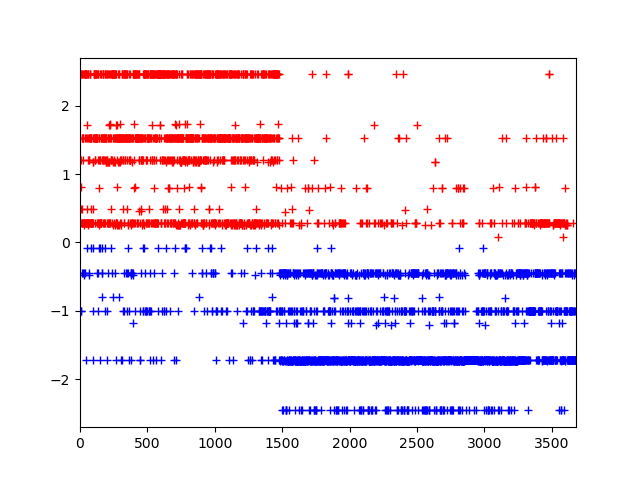


图6 基分类器5

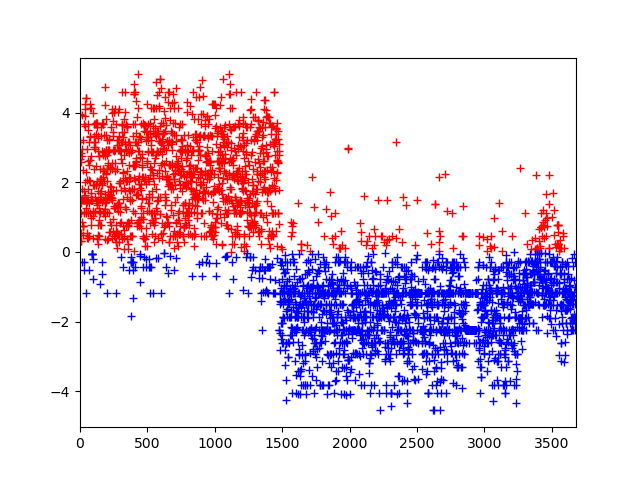


图8 基分类器100

# 5. 个人体会

1. csv文件的读写。需要引入pandas包，我了解了DataFrame、read\_csv，to\_csv等函数的用法和参数列表。
2. 矩阵乘法。两个矩阵大小分别为a\*b和c\*d，要相乘必须满足b=c。
3. 对数几率回归中，alpha值的调整。用2的次幂一个个去尝试，试出范围后在细调，且运行时间不能太长。
4. 决策树桩的时间优化问题。决策树桩的生成过程中，对于每一个分界值都需要预测所有训练样本的结果，计算错误率，以得到错误率最小的树桩，因此运行时间会非常长。通过在运行过程中观察输出每次的错误率，发现总体趋势是先下降后上升，因此错误率的总体趋势开始上升时就可以结束该特征值的分界值遍历。同时，改动了adaboost的结束循环的条件，将最大错误率从0.5改到了0.45，就可以少跑几个错误率比较高的基分类器。
5. 十折交叉验证的手动实现。在数组中取某一部分数据，取数的范围需要储存在数组中，不能储存在列表中。
6. Pyplot函数的运用。因为要将Adaboost分类器的预测结果可视化，需要输出图像，我对plot函数的运用和参数列表，怎样画出标记和线条进行了学习。

对数几率回归做基分类器也可以用贝叶斯线性回归，但普通的梯度下降已经能够满足正确率的要求，而且运行时间更短，所以还是选择了普通的梯度下降。

这次大作业使我对决策树桩、对数几率回归有了更深刻的了解，也让我对模型训练的优化过程有了体会。学习了Adaboost的原理和运用，能够用adaboost的方法强化以上两个弱学习器来得到正确率更高的训练模型。同时也了解了贝叶斯线性回归的代码实现方法，以及十折交叉验证的作用和手动实现的方法。但根据代码的运行结果来看，时间上的优化还有待加强。总的来说收获很大。

建议：希望能统一python解释器的版本，2.7版本下evaluate函数中正确率的计算是整数除以整数，运行结果一直是0，需要在第一行加”from \_\_future\_\_ import division”才能得到浮点数，3.10版本会给”np.int”报异常，最终选的是2.7版本。

# 6. 问题回答

(1) 你对AdaBoost 算法有何新的认识？

Adaboost的工作机制是先在初始训练集上训练出一个基学习器，再根据基学习器的表现对训练样本的权重分布进行调整，使得先前基学习器预测错的样本在后续受到更多关注，然后基于调整后的权重分布训练下一个基学习器；如此重复进行，直至基学习器数目达到事先指定的值T，最终将这T 个基学习器进行加权线性组合。

在基分类器个数为100的运行过程中输出基分类器的个数，可以发现基本不会跑到100个分类器，大多在20到40个就结束了，因此当基分类器的指定个数较多时，结束循环的条件决定了基分类器的个数和预测结果的精度，而不是个数越多精度越高。

(2) 关于基分类器类型、超参数设置对最终模型性能的影响，你有何发现？

对于这次大作业提供的数据，是以决策树桩作为基分类器更好。

超参数设置时，迭代次数不能过高，容错率不能太小，否则运行时间过长。步长不能过小，否则基分类器不能在迭代次数内达到较高的精度。