**模式识别作业报告**

**实现logistic回归与应用adaboost预测信用卡欺诈**

**何雨旻 Y30180644 控制1班**

经过模式识别课程的学习，在赵老师的悉心讲解下，对机器学习中一些重要的算法有了一定的了解与认识，所以本次作业希望运用课上所学的知识针对二分类问题进行编程实践。

起初想运用lintcode上的信用卡欺诈识别数据集进行对率回归，但是下采样之后的数据集如果用到我的模型上一直报错。由于能力有所欠缺，一直没有debug成功，所以找了另一个简单的数据集“疝气病症病马”进行了对率回归的分析。此外，通过阅读资料发现，元算法在分类中的效果十分地好，所以想将adaboost运用在信用卡欺诈这题上，看看能否不通过下采样得到结果。

**1、对率回归预测病马死亡率**

**1.1 疝气病症病马数据集介绍**

我所使用的数据集分别为horseColicTraing.txt(训练集)以及horseColicTest.txt（测试集），目的是通过logistic regression来判断测试集中的马的存活状态。训练集以及测试集是一个有21维特征的数据集，第22列表示了马是否已经死亡，如果死亡了为1，仍然存活为0。需要说明的是，该数据集的特点是有大约30%的数据是缺失的，我下载的数据文件是将缺失的部分补0并删除类别标签缺损的数据。

**1.2 logsitic regression 方法介绍**

1.2.1 sigmoid函数

最简单的分类方法是线性回归，即将特征与其对应的权值相乘，再与某一阈值相比较，实现分类的目的。但是如果得到的预测结果与阈值相差极小，这时则较难归类。数学上有一个函数可以将任意实数转换为[0,1]之间的函数，即sigmoid函数。



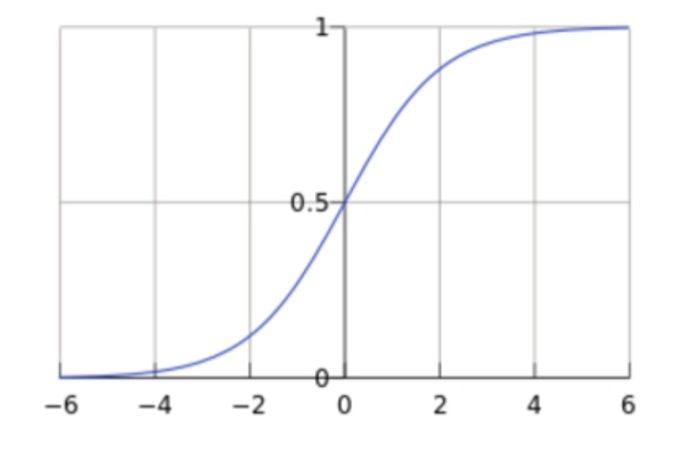


图1 sigmod函数图像

这时则可以将预测结果的取值限制在[0,1]区间上，而[0,1]区间则正是概率的区间，自然而然的可以将某一类的预测结果与概率联系起来，则预测结果可以当作某一类别的后验概率。那么，如果要实现二分类的话，则可以把0.5当作阈值区分两个类别。

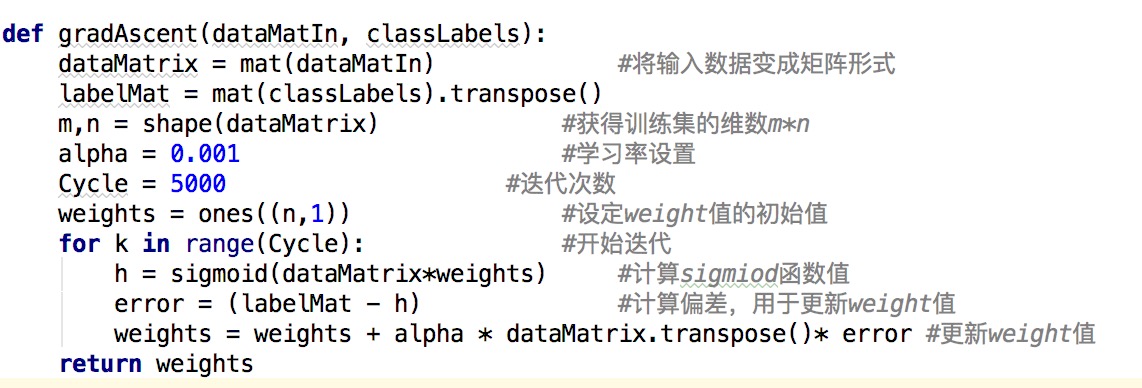
其中，sigmoid函数的输入可由下式表示，写成向量的形式为，其中向量训练集中数据，即分类器的输入数据，向量则是我们要找的最佳参数，寻找最佳参数的方法可以由梯度下降法或者梯度上升法得到。（梯度上升和梯度下降本质是一样的，就是项的正负决定的）。

1.2.2 梯度上升法

梯度上升法即每次朝着梯度的方向进行移动，则会到达某函数的最大值，,,其中，表示第j个特征的权重，为学学习率。

其中，代表sigmoid函数，所以，在使用梯度上升法更新权值时，只要根据下式进行迭代即可。

1.2.3 函数实现



.其中学习率设置为0.01，迭代次数为5000。

1.3 利用对率回归处理数据集

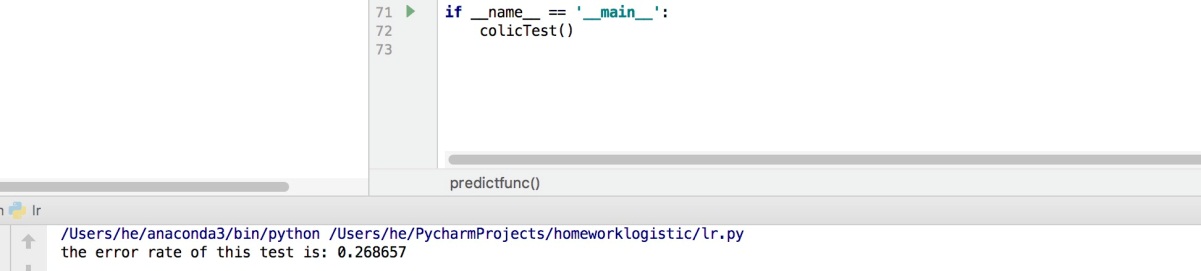
由于该数据集缺失的部分预先被处理过，所以将该数据集进行分类的代码实现相对比较简单。首先导入训练集以及测试集，利用append()函数将数据导入trainingSet和trainingLabels ,这里好像open函数导入数据之后并不是以数组形式存放的，和matlab有一些区别。之后利用前面提到的gradAscent()函数实现梯度上升方法求解weight。经过测试发现，Weight经过5000次迭代之后可以算出一个相对较优的解。然后，像一开始处理训练集那样将测试集导入，并利用predictfunc()来对得到的weight进行计算，并和测试集的结果相比较，得到测试错误率。



Predictfunc()即将所得到的利用sigmoid函数表示成[0,1]之间的数，从而进行分类。



最后得到的分类错误率是0.268657，预测结果并不是很好，我猜想有可能是因为样本有所缺失的原因造成的。



**2 利用Adaboost预测信用卡欺诈数据**

在一本书上看到说有的人认为Adaboost是最好的监督学习方法，所以我想将Adaboost运用再信用卡欺诈数据集上试试看。

**2.1 Adaboost原理介绍**

AdaBoost是基于boost方法的树集成方法，核心策略是每一轮迭代完成，更新样本的权重和基分类器权重。AdaBoost的基本训练过程如下：

首先有训练集D={}，基学习算法E，训练轮数。然后计算，初始化样本的权值分布。开始迭代，从t=0到训练轮数T，计算=E(,),基于分布，从训练集中训练出基分类器。然后计算该弱分类器的误差，如果误差大于0.5，则弱分类器不合格，终止该分类器训练。如果误差小于0.5，则计算该基分类器参与最终决策的权重。e表示该分类器的误差，最后计算新的样本权重：

最终输出的结果为

其中表示根据每个样本分类正确与否更新样本权重，即新的样本分布，分类正确的降低权重，分类错误的增加权重。但是对于正误实例降低或者增加的权重都是相同的。在信用卡欺诈数据集这种样本不均衡的情况下，基本上是非欺诈的数据，那么极有可能将非欺诈数据误判成欺诈数据，如果将此类代价高的误判赋予一个大于1的值，那么adaboost方法将更适用于不均衡数据集的判断。

所以，将

改写为

是一个代价函数，根据每次迭代每个样本的误判情况调整。具体调整方法后续会说明。

**2.2信用卡欺诈数据集介绍**

该数据集为2013年9月里某两天欧元区的信用卡交易记录。在这两天中共有284807笔交易，其中的492笔是欺诈。把欺诈交易的类(class) 认为是1，非欺诈交易的类认为是0。 这个二元问题中，class为1的样本概率只有0.172%。从图中可见在这个二元问题中，两个类所占的比重相差特别大。

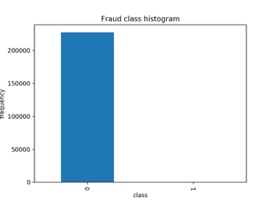


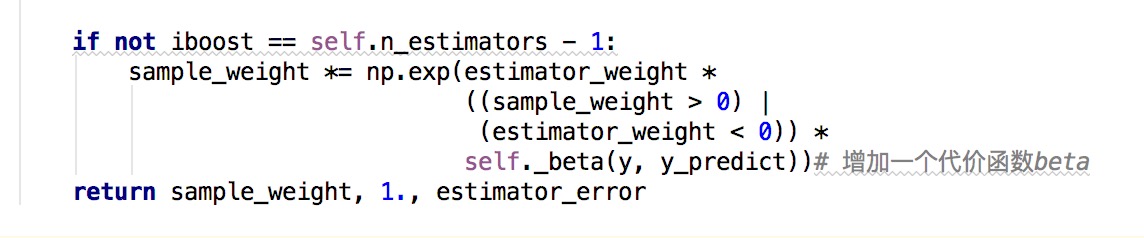
图2 信用卡欺诈数据集

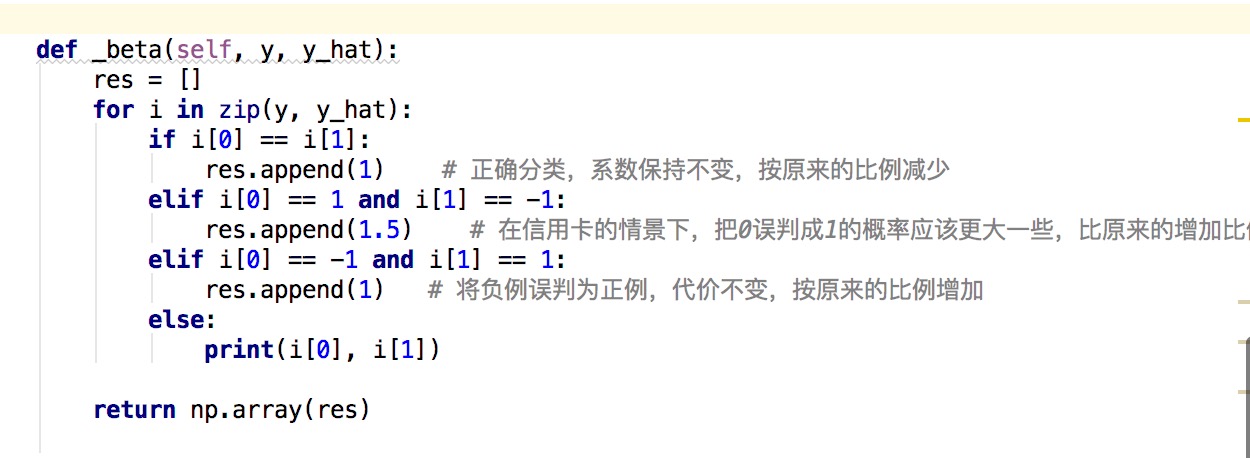
本次实验主要使用的数据文件是creditcard\_train.csv（训练集）和creditcard\_test.csv（测试集）。数据集中一共有284807条数据，284315个正例，492个负例，一共有29个特征分别为。v1-v28以及amount总数。

**2.3利用Adaboost处理信用卡欺诈数据集**

受限于编程能力而且对boost算法理解还不够深入，编程实现Adaboost算法还没有成功，所以在此仅仅改写了一下 adaboostClassifier()中实现权重更新的部分代码供训练信用卡欺诈数据集用。

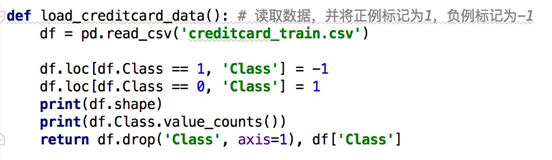
在 Adaboostclassifier 的 boost\_real方法中更新权值的公式存放在这里，所以应该在此处追加代价函数beta。





代价函数beta如上所示，正确分类，系数保持不变，按原来的比例减少，# 在信用卡的情景下，将正例误判的代价应该更大一些，比原来的增加比例要高，将负例误判为正例，代价不变，按原来的比例增加。主要作用就是将正确样本误判的那个样本的权值增加，增加分类器的精度。

在对数据集进行处理的credit test1.py中，首先定义了load\_creditcard()读取了‘creditcard\_train.csv’中的数据，并且将正例标记为1，负例标记为-1，并输出了正反例的数量。



在程序执行部分中，首先读取了数据，并且对数据进行了分割，将训练数据中的20%当做了测试集，从而可以计算取召回率、精度、F1值这类模型评估指标。此外，当模型训练好了之后，执行第28行程序载入真正的测试集，然后执行第39-46行程序即按照‘sample\_submission .csv’的格式输出真正预测数据。



修改后的Adaboost程序以及原始sklear库中Adaboost运行的结果如下：C:\Users\quyi\Documents\Tencent Files\1274013465\Image\C2C\{28C7F829-49CF-6F52-BBEE-DF5ECAFB3027}.png

C:\Users\quyi\Documents\Tencent Files\1274013465\Image\C2C\{F29544BA-AAFD-B47C-B1EE-785DFB1A1F07}.png

可以发现修改过后的Adaboost精度明显提高，但是F1值和召回率都降低了不少，增加了权值之后将可能适得其反，将许多正例误判成了反例。

**3、总结**

本次模式识别的作业我完成了利用python实现梯度上升法下的logistic regression，并将其用于疝气病病马数据集，修改了sklearn中的Adaboost方法并运用到信用卡欺诈数据集。对病马预测的错误率约为28%。对信用卡欺诈预测的召回率为50%，精度为98%，F1值为66%。

一开始，我其实想利用对率回归进行信用卡欺诈预测，将数据集行下采样后交叉预测的结果全部都为1，也一直未debug成功，所以找了一个简单一点的数据集进行处理。但还想进行信用卡欺诈预测，查阅资料后运用了Adaboost算法，虽然只是调用了sklearn的库函数，但也尽力了。

从这次作业中我发现要将赵老师上课讲的方法运用到真正的数据集中比在纸上推算法要难多了，而且不同的数据集有不同的特点，方法的选择也很重要。同时，python的编程实现也比matlab要难上不少，有许多编程上的bug都不知道怎么解决。有了这次作业的经历，感受到了机器学习以及模式识别的魅力与难度。以后一定要更加努力地学习。

程序清单：

Adacost.py-修改adaboost的程序

Creditcardtest1.py-利用修改后的adaboost验证信用卡欺诈程序

Submission1.csv- 对于信用卡欺诈输出的测试集预测结果

Lr.py-logistic回归程序