

**模式识别大作业**

题 目 信用卡欺诈识别

学 院 信息科学与工程

专 业 控制科学与工程

组 员 杨涛（Y30180688） 肖若诚（Y30180683）

应桃（Y30180692） 俞天钧(Y30180693)

王旭鹏（Y30180679） 乔伟涛（Y30180670）

指导教师 赵海涛

**完成日期： 2018 年 12 月14日**

**模式识别大作业——信用卡欺诈识别**

**分工：**

**1数据分析：杨涛**

**2程序编写：杨涛、肖若诚、王旭鹏**

**3报告撰写：应桃、俞天钧、乔伟涛**

**方案一（Logistics regression）**

一.题目简介

**1.1题目及数据描述**

信用卡公司需要能够识别虚假的信用卡交易，以避免用户被骗。本题就是利用欧洲某段时间的信用卡交易记录，来判断某条交易是否属于信用卡欺诈。

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227789笔交易，其中的395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.那么这个二元问题中，class为1的样本概率只有0.173%。可见在这个二元问题中，两个类所占的比重相差特别大。也就是说我们的数据集是特别不平衡的。数据集里面有从PCA转换得到的28个features。Features V1, V2, ..., V28都是经过PCA转换后获得的features。没有被PCA转换的，保留了原始数据的是"Time"和"Amount"。"Time"记录了每笔交易和第一笔交易之间的时间间隔，以秒为单位。"Amount"是交易数额。"Class"就是这笔交易的最终分类。如果为1说明这是一笔欺诈交易，为0说明这不是一笔欺诈交易。

**1.2数据特征**

本题数据是非常不平衡的，只有0.173%的交易为欺诈交易。所以如果模型什么也不做，直接把所有的交易预测为非欺诈，也会有超过90%的准确率。因此准确率在本题中不是衡量模型好坏的有效指标。而且对于银行来说，检验到欺诈交易远比检验到非欺诈交易重要，古引入召回率（后面介绍）这一概念。

二．logistics回归

**2.1 Logistics回归分析的具体思路及推导**

**2.1.1线性回归**

线性回归假设特征和结果满足线性关系。其实线性关系的表达能力非常强大，每个特征对结果的影响强弱可以有前面的参数体现，而且每个特征变量可以首先映射到一个函数，然后再参与线性计算。这样就可以表达特征与结果之间的非线性关系。

我们用*X1，X2..Xn*去描述描述对象里面的分量，我们可以做出一个估计函数：

 （其中：）

*θ*在这儿称为参数，在这的意思是调整feature中每个分量的影响力。我们程序也需要一个机制去评估我们*θ*是否比较好，所以说需要对我们做出的*h*函数进行评估，一般这个函数称为损失函数（loss function）或者错误函数(error function)，描述*h*函数不好的程度，我们引入如下函数为*J*函数：



注：这里引入1/2在之后会被削掉，是为了之后的推导过程可以更简单直观。

如何调整*θ*以使得*J(θ)*取得最小值有很多方法，其中有最梯度下降法。

**2.1.2 梯度下降**

在选定线性回归模型后，只需要确定参数*θ*，就可以将模型用来预测。然而*θ*需要在*J(θ)*最小的情况下才能确定。因此问题归结为求极小值问题，使用梯度下降法。梯度下降法最大的问题是求得有可能是全局极小值，这与初始点的选取有关。

梯度下降法是按下面的流程进行的：

1. 首先对*θ*赋值，这个值可以是随机的，也可以让*θ*是一个全零的向量。
2. 改变*θ*的值，使得*J(θ)*按梯度下降的方向进行减少。

已知：，设有：

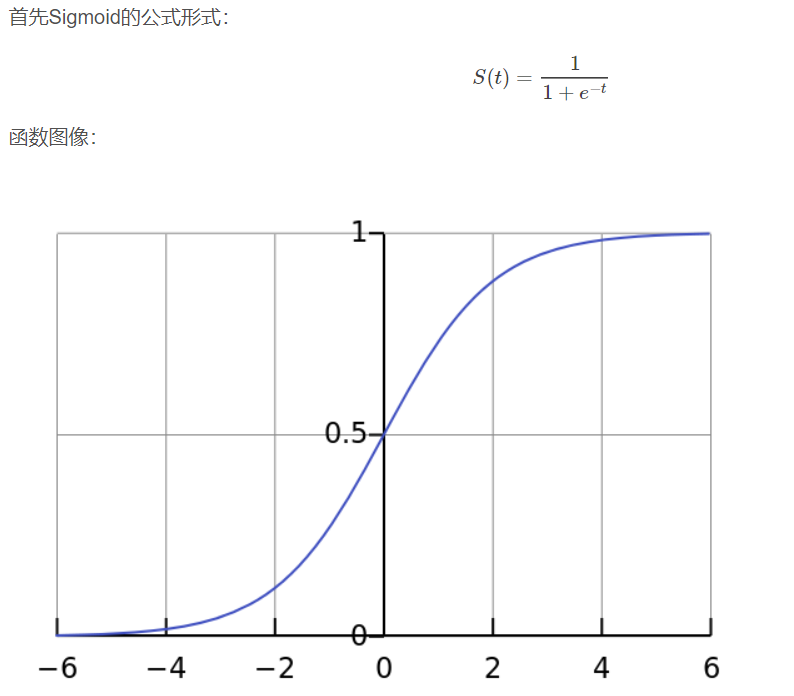


其中，梯度方向由*J(θ)*对*θ*的偏导数确定，由于求的是极小值，因此梯度方向是偏导数的反方向。

结果为：

**2.1.3 Logistics回归与Sigmoid函数**

引入Sigmoid函数的概念：



函数的基本性质：

1.定义域：(−∞,+∞)(−∞,+∞)

2.值域：(−1,1)(−1,1)

3.函数在定义域内为连续和光滑函数

4.处处可导，导数为：*s′(x)=s(x)(1−s(x))*

其中，性质4可以推导：



一般来说，回归不用在分类问题上，因为回归是连续型模型，而且受噪声影响比较大。如果非要应用进入，可以使用Logistics回归。

Logistics回归本质上是线性回归，只是在特征到结果的映射中加入了一层函数映射，即先把特征线性求和，然后使用函数 *g(z)*将最为假设函数来预测。*g(z)*可以将连续值映射到0和1上。

Logistics回归的假设函数如下，线性回归假设函数是。







Logistics回归用来分类 0/1 问题，也就是预测结果属于 0 或者 1 的二值分类问题。这里假设了二值满足伯努利分布，也就是



通过简化可以写得：

假设*m*个训练实例是独立生成的，那么我们可以写出参数的似然性：



对取对数可以得到：



最终我们的目标是最大似然估计函数，即：

**2.1.4梯度上升以及其改进**

梯度上升算法和我们平时用的梯度下降算法思想类似，梯度上升算法基于的思想是：要找到某个函数的最大值，最好的方法是沿着这个函数的梯度方向探寻！直到达到停止条件为止。梯度上升算法用来求函数的最大值，而梯度下降算法用来求函数的最小值。



推导过程如下：

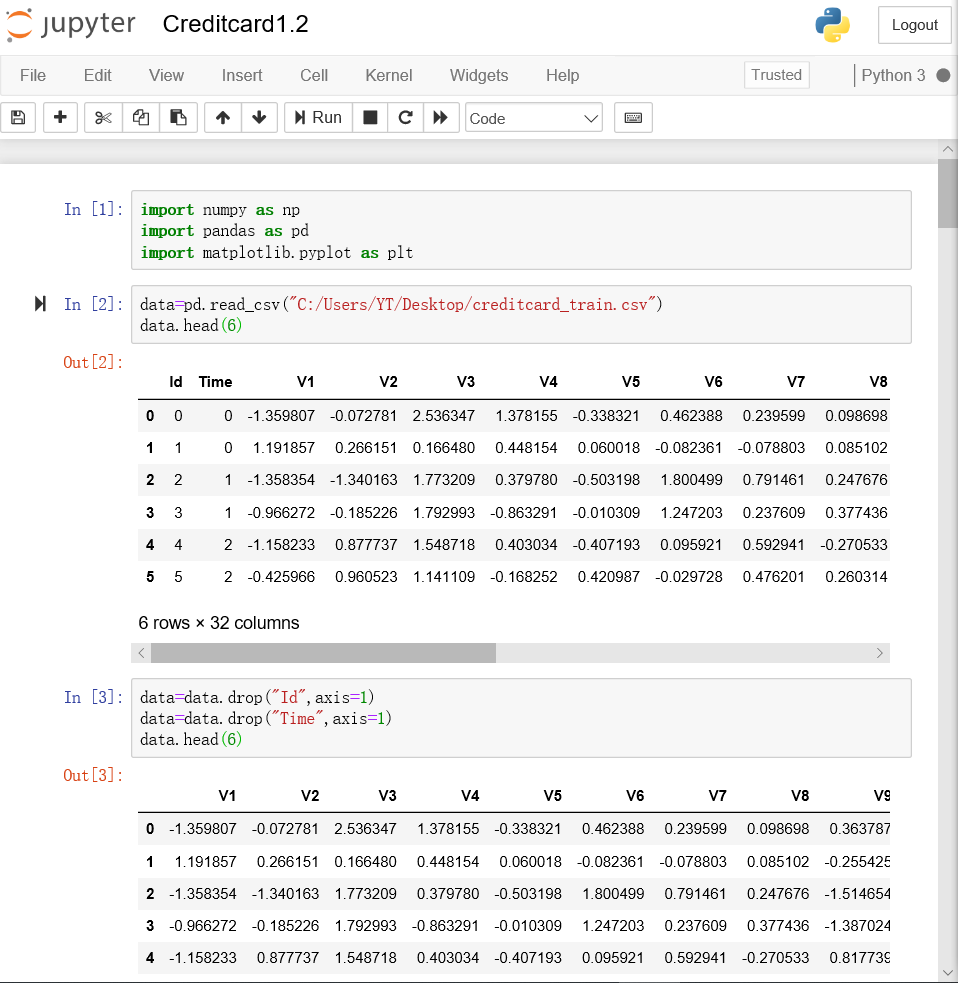


最终可以得到结论为：

三 数据分析

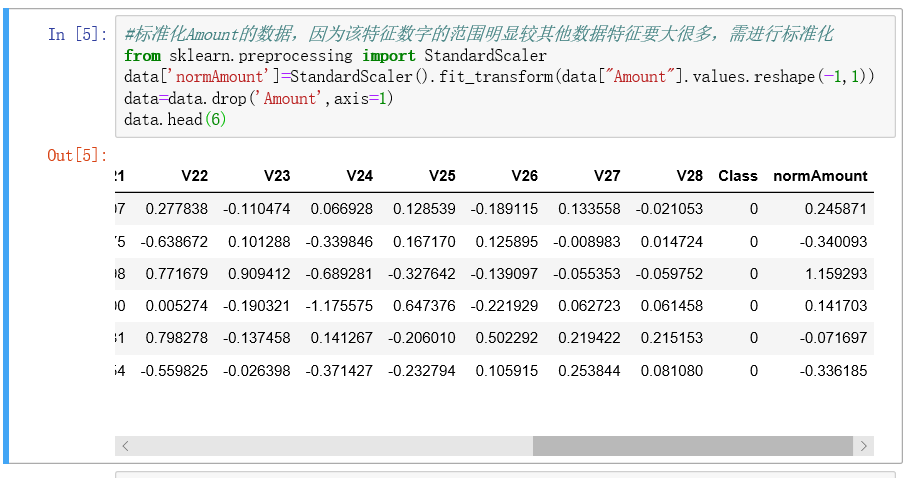
**3.1 数据读入与分析**

读入数据并将不需要的特征属性去除

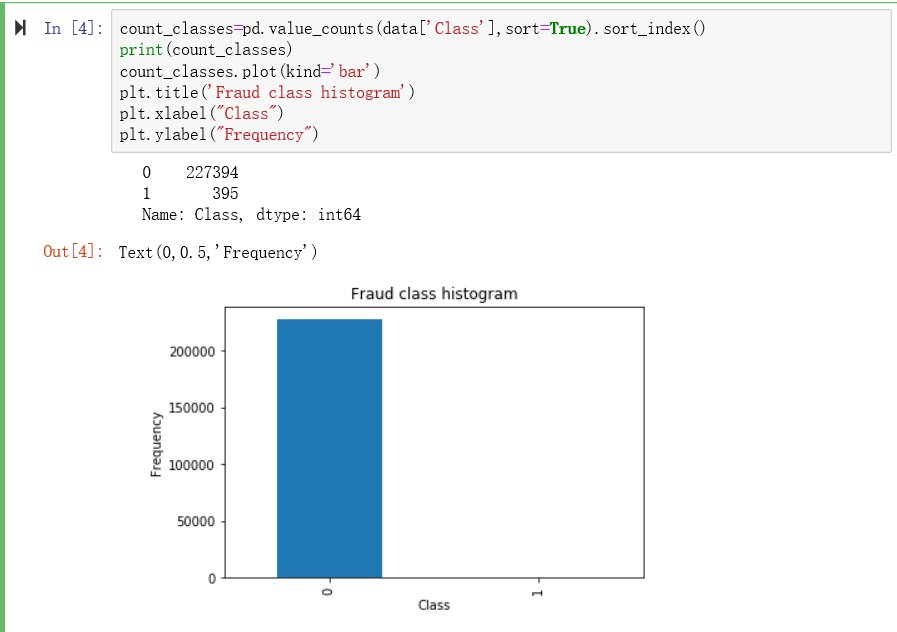


由于数据集给出时已经经过PCA处理，此处我们只需对数据做一些基本分析与处理。再去除不用的特征属性Time与Id后我们观察剩下的特征属性发现，有一个属性的取值相对于其他属性取值较大（Amount），其他特征属性经处理后取值基本都坐落在-1到1之间，而Amount的取值接近三位数，若不对其进行相应的处理会使得在模型建立时，会使得模型潜意识将这一属性赋予看的很重要，使其权重很大。故需对这一属性进行标准化。

标准化程序与结果：



统计分析样本分布情况：



读入数据后我们利用简单的柱形图直观的观察一下数据集的分布。可以发现存在极度的分布不均匀，有问题样本仅仅只有395个，相比于无问题样本的个数而言太少。对于这类样本分布极度不均匀的数据集不可直接拿来训练模型参数，同时，精度这一判断正确率的手段在此处也不在适用。假若我么所建立的模型就是简单地将所有样本都判定为无问题样本，那我们的准确率（即精度）也将达到90%以上。但是，问题在于我们所建模型目的在于将问题样本找出来，然而此处却未找到一个有问题样本，那我们的模型也就没有实际的用处。故在判断我们模型的好坏时我们引入召回率（即正确找到有问题样本的概率。例如，某班级工50人，其中女生有20人，男生30。目标为找到所有女生。TP表示将女生判定为女生的人数，FP表示将男生判定为女生的人数，FN表示将女生判定为男生的人数，TN表示将男生判定为男生的人数。而召回率Recall=TP/TP+FN）.

在面对这样分布极度不均匀的数据集时，我们有两种方法处理梳理数据以使得参数训练样本集分布均衡。

1：下采样法：

下采样法是从较多的正常样本集中随机抽取与相对较少的有问题的样本相同的个数与其重新构成参数训练样本集。

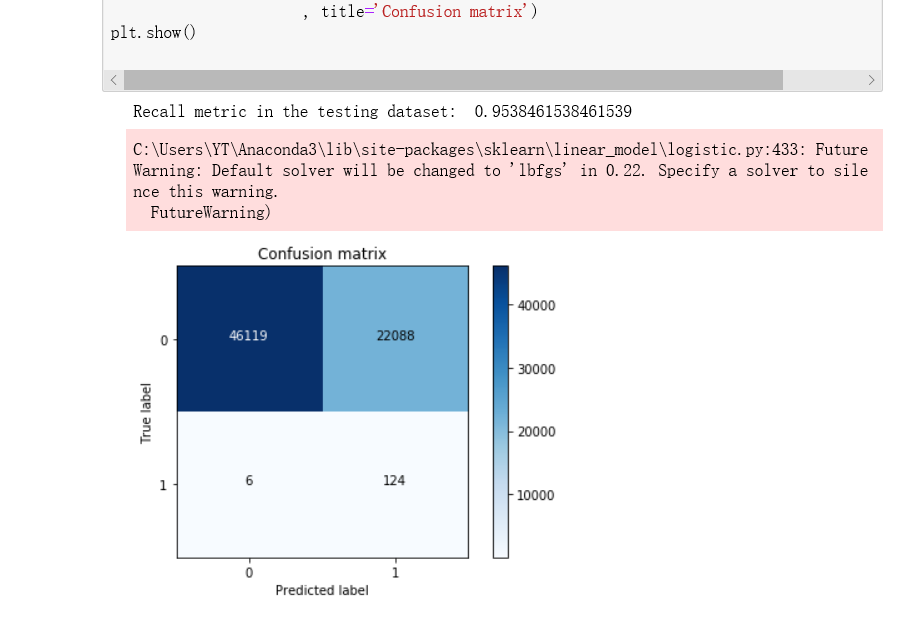
下采样程序：



通过下采样的数据分析我们发现，下采样虽然能够在平衡样本数据的同时极大的减少运算量，但我们在交叉验证时利用混淆矩阵显示结果时可以很清晰的发现其误杀率太大，也即为了找出有问题的样本，错将太多的无问题样本也判定成为有问题样本。这样虽然使得召回率提升了，但精度却大幅度下降了。假设该模型用于检测癌症患者，为了找出个别的癌症患者而错将很多无癌症的人判定为癌症患者而通知他们病情，这必将引起恐慌。故下采样法存在着一定能够的缺陷。

下采样模型的交叉验证结果：





从结果可以看出，该模型的召回率已经达到95%以上了，但再看混淆矩阵时可以发现下采样时的模型为了找出124个有问题样本时错将20874个无问题样本判定成为有问题样本，真可谓是宁可错杀一千也不可错放一个啊。但相比于不对样本做任何处理直接来训练来说其效果还是有很明显的提升。直接利用原始数据进行训练时，模型的召回率基本只有50%-60%左右（如下图）。



2：上采样法（过采样法）：

考虑到下采样所带的问题，我们考虑另一种均衡数据方法——上采样。上采样与下采样相反，通过随机生成有问题样本使其数目与无问题样本达到均衡。此处需要引入一个随机生成样本的算法——SMOTE算法。

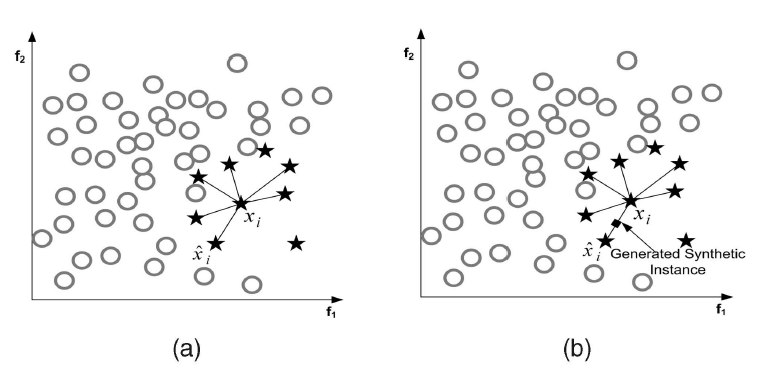
SMOTE（Synthetic Minority Oversampling Technique），合成少数类过采样技术．它是基于随机过采样算法的一种改进方案，由于随机过采样采取简单复制样本的策略来增加少数类样本，这样容易产生模型过拟合的问题，即使得模型学习到的信息过于特别(Specific)而不够泛化(General)，SMOTE算法的基本思想是对少数类样本进行分析并根据少数类样本人工合成新样本添加到数据集中，具体如下图所示，算法流程如下。

(1)对于少数类中每一个样本x，以欧氏距离为标准计算它到少数类样本集中所有样本的距离，得到其k近邻。

(2)根据样本不平衡比例设置一个采样比例以确定采样倍率N，对于每一个少数类样本x，从其k近邻中随机选择若干个样本，假设选择的近邻为xn。

(3)对于每一个随机选出的近邻xn，分别与原样本按照如下的公式构建新的样本。

IMG_256

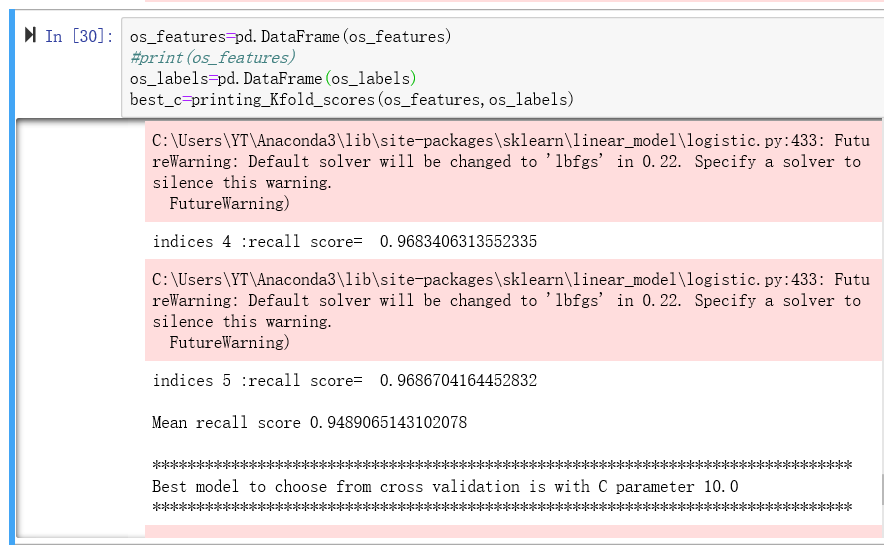


程序实现如下：

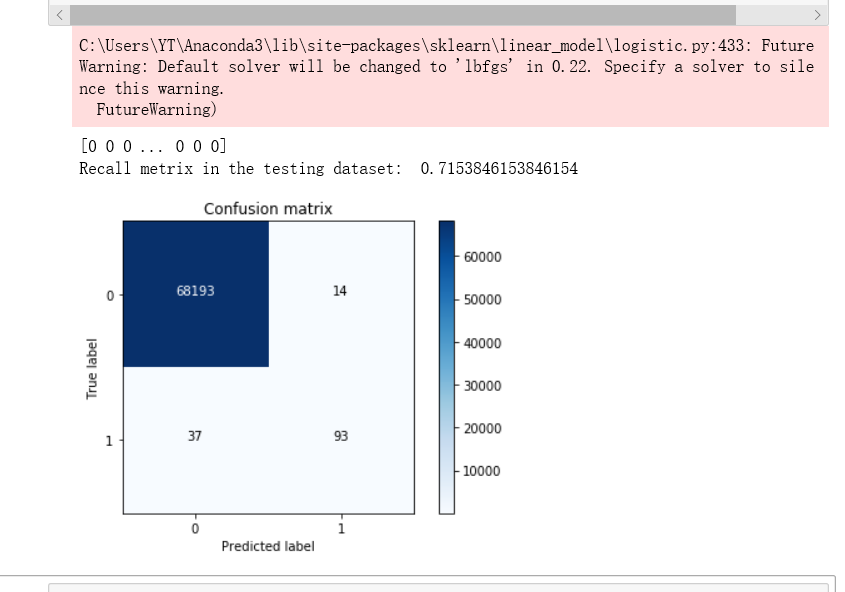


可以看到经过SMOTE算法随机生成样本后，样本类型为1的目前已有181926个。与下采样法一样，上采样法也存在缺陷。上采样一方面带来了巨大的计算量，另一方面由于随机生成的样本毕竟不是原始样本，故其只能用于模型的参数训练，确定模型的最佳参数，不能用作模型的训练，否则会造成极大的过拟合。

过采样参数选择程序：



混淆矩阵：



比较下采样与上采样两种平衡数据方法的交叉验证结果可以看出，相比较于下采样而言，上采样虽然虽然增加了计算量，而且召回率相比较下采样也有下降，但是从两者的混淆矩阵可以看出，上采样极大的减少了误杀率。总体来说上采样的效果更好。故此次我们建立模型时采用上采样方法。

3.2 交叉验证

交叉验证是在机器学习建立模型和验证模型参数时常用的办法。交叉验证，顾名思义，就是重复的使用数据，把得到的样本数据进行切分，组合为不同的训练集和测试集，用训练集来训练模型，用测试集来评估模型预测的好坏。在此基础上可以得到多组不同的训练集和测试集，某次训练集中的某样本在下次可能成为测试集中的样本，即所谓“交叉”。

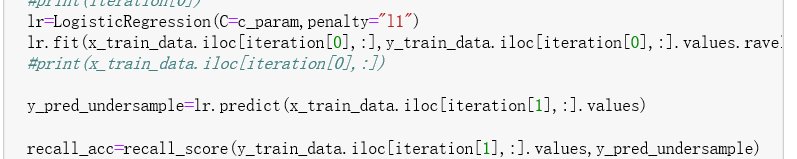
我们随机的将样本数据分为两部分（比如：70%的训练集，30%的测试集），然后用训练集来训练模型，在测试集上验证模型及参数。接着，我们再把样本打乱，重新选择训练集和测试集，继续训练数据和检验模型。最后我们选择损失函数评估最优的模型和参数。

样本切分程序：



其中test\_size=0.3表示训练集占7成，测试集占3成。

交叉验证程序：



利用训练集建立模型，再利用测试集检验模型。

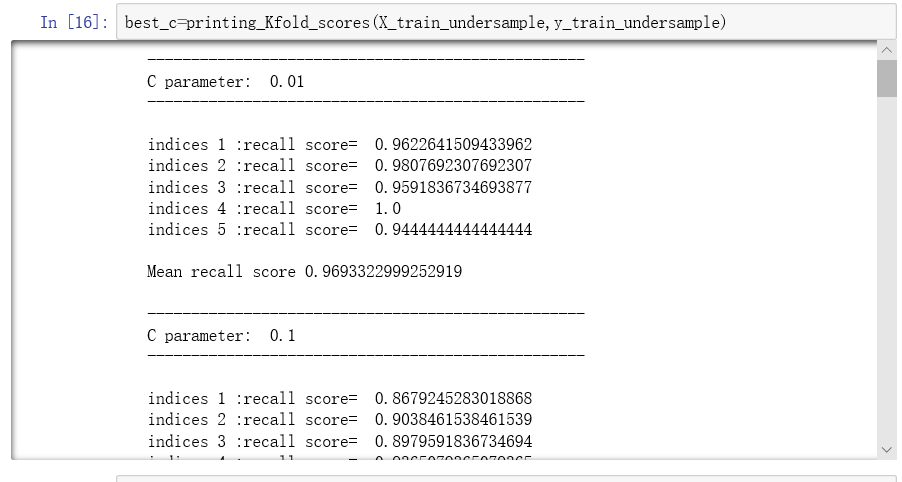
**3.3正则化惩罚**

过度拟合的问题通常发生在变量（特征）过多的时候。这种情况下训练出的方程总是能很好的拟合训练数据，也就是说，我们的代价函数可能非常接近于0或者就为0。但是，这样的曲线千方百计的去拟合训练数据，这样会导致它无法泛化到新的数据样本中。

对于这种问题的解决方法，我们选用正则化。正则化中我们将保留所有的特征变量，但是会减小特征变量的数量级（参数数值的大小θ(j)）。当我们有很多特征变量时，其中每一个变量都能对预测产生一点影响。我们可以有很多特征变量，其中每一个变量都是有用的，因此我们不希望把它们删掉，这就导致了正则化概念的发生。

如果我们用一个更高次的多项式去拟合，最终我们可能会得到一个曲线，它能很好地拟合训练集，但却并不是一个好的结果，因为它过度拟合了数据，因此，我们要加上惩罚项，我们设惩罚项系数为C=0.01，0.1，1，10，100。依次进行运算。

程序运行结果如下（程序见附录）



3.4 模型评估

**模型评估**

一般classification问题可能考虑准确率就够了。但是本题由于超过90%的分类都为非欺诈，即使直接“预测”成非欺诈也会有90%以上的正确率。因此precision在本题并不是最重要的指标。应当注意到，银行更关注的是欺诈交易有没有被都筛选出来，也是recall这个指标所表示的含义，本题用来衡量模型好坏的指标是F1值。F1值有既考虑precision，又考虑recall的优点。

通过表2，我们可以认识到TP,FP,FN,TN这4种不同的分类情况，召回率(Recall)=系统检索到的相关文件/系统所有相关的文件总数，即Recall=TP/(TP+FN)。利用召回率，我们可以检验我们的模型是否更加合理。

最终的公式为：

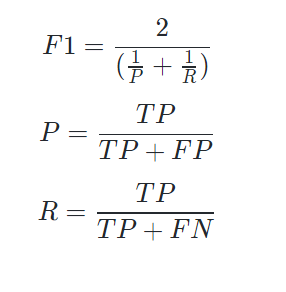
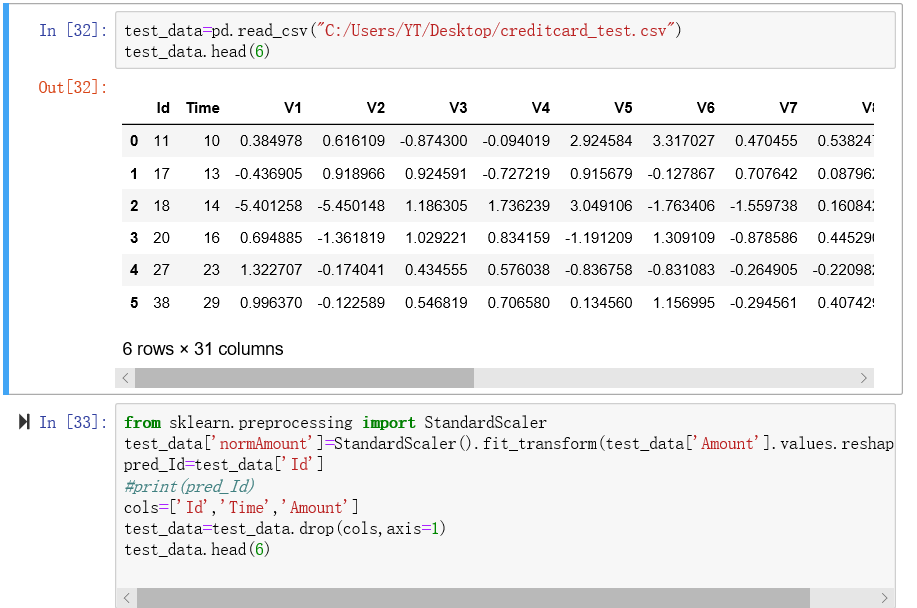


表2 4 种不同的分类情况

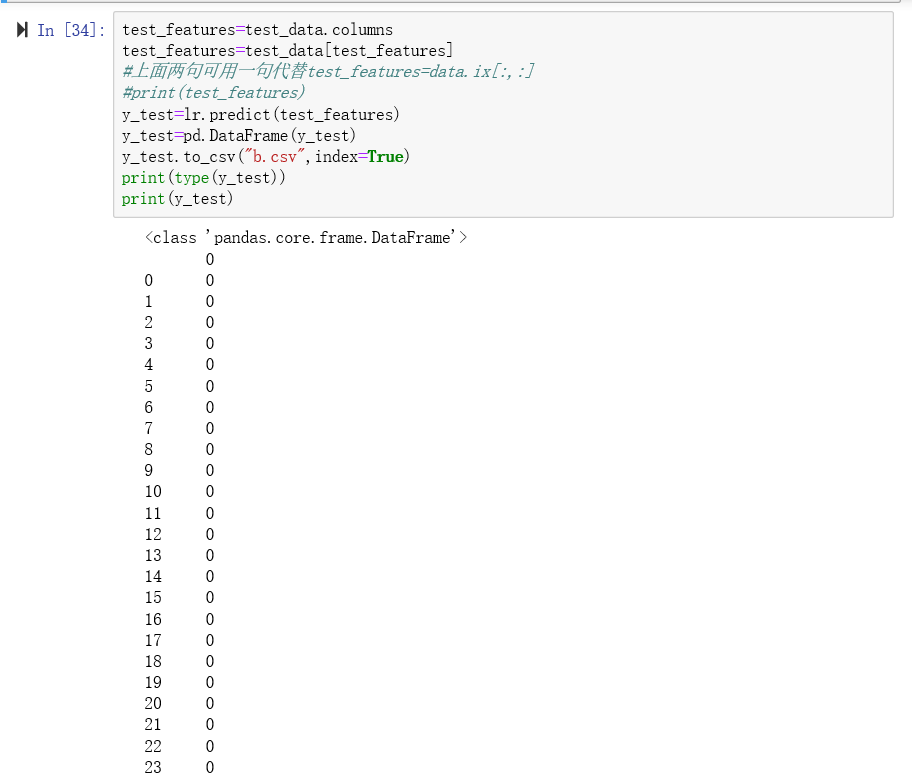
|  |  |  |
| --- | --- | --- |
|  | **相关(Relevant),正类** | **无关(NonRelevant),负类** |
| **被检索到(Retrieved)** | true positives(TP 正类判定为正类, | false positives(FP 负类判定为正类,"存伪", |
| **未被检索到(Not Retrieved)** | false negatives(FN 正类判定为负类,"去真", | true negatives(TN 负类判定为负类, |

四 模型训练及预测

**4.1：预测数据的读入与预处理**



预测结果输出：



注意在训练模型时我们用的是原始分割样本，不是上采样样本。

4.2 提交结果



方案二（朴素贝叶斯）

**一、信用卡欺诈识别简介**

在本题中，我们使用欧洲某段时间的信用卡交易记录，来判别是否为欺诈。数据中提炼了经过PCA变换的28个特征数据、交易时间间隔、交易金额和是否为信用卡欺诈的标签。针对这些数字特征确定出合适的模型，并用于判定是否为信用卡欺诈。

**二、整体解决方案**

将最终信用卡欺诈与否抽象为一个0-1分类问题，在仔细思考与对原始数据研究的基础上，我首先确定了一个大的求解的框架，就是把这个问题看成一个分类问题，利用数据，找出判别信用卡欺诈的因素，并且对这些“因素”与“是否为欺诈”这两者关系进行建模。

建立模型后，在训练数据上反复调试模型中的部分参数，使得分类器对于训练数据达到较好的分类效果后，将该模型用于测试数据的预测，并分析其预测效果。

**2.1 数据预处理**

由于在数据中共有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. 那么这个二元问题中，class为1的样本概率只有0.172%。可见在这个二元问题中，两个类所占的比重相差特别大。也就是说我们的数据集是特别不平衡的。失衡数据很可能导致train出来的模型是无效模型，因此需要处理这种失衡的数据。在这次的想法中，通过对欺诈数据进行上采样，即将欺诈数据扩充为原数据的200倍，增大欺诈分类占比。其代码如下：

clc

clearvars -except data Id indices S2 SM testV V V1 k %保留部分变量

m=length(SM);

k=30;

indices=crossvalind('Kfold',m,k);%随机分K折

data=cell(1,k);%分k个单元

for i=1:k

data{1,i}=[];

end

for i=1:m %遍历indices，分配数据

data{1,indices(i)}=[data{1,indices(i)};SM(i,:)];

end

%% k个子集中找出0-1变化位置

a=0;b=0;

count=zeros(k,1);

%把每折数据中的两类分开 data里的每折数据是label先为0后为1的排列顺序，因此找出由0变1的位置即可（count）

for i=1:k

n=length(data{1,i});

for j=1:n

if (data{1,i}(j,30)==0)

count(i,1)=count(i,1)+1;

else

break

end

end

end

**2.2 算法原理及程序实现**

**2.2.1 朴素贝叶斯模型**

本次训练数据采用的是朴素贝叶斯方法。即，已知条件概率为：



可以得出贝叶斯分类器的公式为：



其中即判定是否为信用卡欺诈。

由于用条件概率进行分类，因此是基于某些事情(w)发生的情况下，然后判断这件事情属于哪一类。在本题的信用卡欺诈识别中：基于给定的30组数据，判别这条信息是否属于欺诈，w就是这些关键数据组成的向量。设,即欺诈标注为1，正常交易标注为0。如此一来。就是已知w，求p(1|w)和p(0|w)，根据贝叶斯分类器的公式，如果求得p(1|w)>p(0|w),那么就认为w属于欺诈。

**2.2.2 根据题目对贝叶斯分类器进行处理**

同时，为了提高训练结果的准确性，对训练集数据进行交叉验证设计，将修正后的数据均匀分布在30个组内，依次以每一组作为测试样本进行模型调试，获取最优效果。其代码如下：

首先由于不影响比较关系，所以没有必要参与计算。其次是先验概率，比较容易求得。比如数据库中共有10条数据，如果有4条是诈骗，那么就是p(1)=4/10,p(0)=6/10。

比较难处理的是p(w|1)和p(w|0)，这里假设每个数据出现是独立的，不会相互影响，即，且由于28个数据特征和交易时间间隔是连续函数，可视作服从正态分布。这样在求联合概率密度过程中，我们就可以将先求每个特征的条件概率再相乘即可。

但是针对交易金额为离散变量，需要作离散贝叶斯处理。将label为1和0中的离散交易金额数据进行条件概率计算，在最终测试集中，通过比较其条件概率进行类别判定。不可避免的，存在从未出现过的金额数据，处于计算方便，将其忽略。

同时，为了提高训练结果的准确性，对训练集数据进行交叉验证设计，将修正后的数据均匀分布在30个组内，依次以每一组作为测试样本进行模型调试，获取最优效果，即存在30次训练模型。其代码如下：

%% 朴素贝叶斯&交叉验证

for z=1:30

clearvars -except data Id indices S2 SM testV V V1 count z aaa k

%将训练数据与测试数据分开 29份做训练 1份做测试

traindata\_0=zeros(1,29);

traindata\_1=zeros(1,29);

for j=1:k

n=length(data{1,j});

if(j~=z)

traindata\_0=[traindata\_0;data{1,j}(1:count(j,1),1:29)];%j!=z时为训练集

traindata\_1=[traindata\_1;data{1,j}((count(j,1)+1):n,1:29)];

else

testdata\_0=[data{1,z}(1:count(j,1),1:29)];%j=z时为训练集

testdata\_1=[data{1,z}(count(j,1)+1:n,1:29)];

end

end

traindata\_0(1,:)=[];%清除第一行

traindata\_1(1,:)=[];

% ---------------------朴素贝叶斯-------------------------

%分别计算两个类别训练集28个特征的均值和方差 假设28个特征均服从正态分布且互相独立

for i=1:28

[mu1(i,1),sigma1(i,1)]=normfit(traindata\_0(:,i));

[mu2(i,1),sigma2(i,1)]=normfit(traindata\_1(:,i));

end

p1=length(traindata\_0)/(length(traindata\_0)+length(traindata\_1));%求先验概率

%计算后验概率 post为后验概率 prodt为似然函数

testdata=[testdata\_0;testdata\_1];

m=length(data{1,z});

answer=zeros(m,1);

for i=1:m

for j=1:28

prodt1(i,j)=normpdf(testdata(i,j),mu1(j,1),sigma1(j,1));%post1代表0类，post2代表1类

post1(i,1)=p1\*prod(prodt1(i,:));

prodt2(i,j)=normpdf(testdata(i,j),mu2(j,1),sigma2(j,1));

post2(i,1)=(1-p1)\*prod(prodt2(i,:));

end

%交易金额amount条件概率计算

c(i,1)=(length(find(traindata\_0(:,29)==data{1,z}(i,29))))/length(traindata\_0);

d(i,1)=(length(find(traindata\_1(:,29)==data{1,z}(i,29))))/length(traindata\_1);

if (c(i,)~=0)%若该交易金额在训练集中未曾出现则跳过

post1(i,1)=post1(i,1)\*c(i,1);

post2(i,1)=post2(i,1)\*d(i,1);

end

if (post1(i,1)>=post2(i,1)) %归一化因子相同，比较似然函数大小

answer(i,1)=0;

else

answer(i,1)=1;

end

end

%F1值计算

true\_answer=[zeros(count(z),1);ones(m-count(z),1)];

TP=sum(answer(count(z)+1:m,1).\*true\_answer(count(z)+1:m,1));

TFN=sum(xor(true\_answer,answer));

F1(z,1)=2\*TP/(2\*TP+TFN);

end

[F1max,z]=max(F1);%取F1最大时的z对应的训练集为测试数据的训练模型

%% ----------------用最佳训练模型测试数据集------------------------------

m=length(testV);

% testV\_slash=testV(:,[4 5 6 7 9 11 12 14 16 18]);

testV\_slash=testV;%28个特征可以筛除一些特征

traindata\_0\_slash=traindata\_0;

traindata\_1\_slash=traindata\_1;

[p,q]=size(testV\_slash);

for i=1:q

[mu1(i,1),sigma1(i,1)]=normfit(traindata\_0\_slash(:,i));

[mu2(i,1),sigma2(i,1)]=normfit(traindata\_1\_slash(:,i));

end

p1=length(traindata\_0)/(length(traindata\_0)+length(traindata\_1));

for i=1:m

for j=1:q

prodt1(i,j)=normpdf(testV\_slash(i,j),mu1(j,1),sigma1(j,1));%post1代表0类，post2代表1类

prodt2(i,j)=normpdf(testV\_slash(i,j),mu2(j,1),sigma2(j,1));

end

post1(i,1)=p1\*prod(prodt1(i,:));

post2(i,1)=(1-p1)\*prod(prodt2(i,:));

%交易金额amount条件概率计算

c(i,1)=(length(find(traindata\_0(:,29)==testV(i,29))))/length(traindata\_0);

d(i,1)=(length(find(traindata\_1(:,29)==testV(i,29))))/length(traindata\_1);

post1(i,1)=post1(i,1)\*c(i,1);

post2(i,1)=post2(i,1)\*d(i,1);

if (post1(i,1)>=post2(i,1))

answer(i,1)=0;

else

answer(i,1)=1;

end

end

TPN=sum(answer(:,1));%计算预测为1的总个数

在本文中，我们求的并不是正真意义上的p(1|w)和p(0|w)。但是其原理还属于朴素贝叶斯分类器。

**2.2.3 具体步骤分析：**

1.收集数据：读取给定的数据文件

2.准备数据：基于上采样方法对数据进行处理，扩充数据容量

3.分析数据：将数据进行随机分组，为交叉验证做准备

4.训练算法：建立贝叶斯模型

5.使用算法：使用建立的模型对测试集数据进行计算

6.得出结果：将测试得到的结果输出为Submission.csv文件，并计算准确率，代码如下：

%% 导出csv文件

columns={'Id','Class'};

data=table(Id,answer,'VariableNames', columns);

writetable(data, 'submission.csv')

上传至网站，结果如下

