基于层次分析法的中小微型企业的信贷决策

摘要

本文以中小型企业的信贷决策问题为研究目的，首先，依据题目确定研究问题的自变量有：价税合计、供求关系以及公司的信誉等级。

针对模型一，已知信誉等级的123家企业做量化分析，做出信贷决策，在解决方案中，按照三个步骤操作：首先，根据企业的实力和信誉，利用主成分分析方法，构造信贷风险评估函数；然后，依据企业的实例和供求能力，利用层次分析法，构造提供贷款与否的判断函数；最后，若提供贷款，依据信誉和信贷风险评估值，再次利用层次分析法，决定利率优惠的政策，分配并给予相应的金额和利率。

针对模型二，在没有信誉等级的情况下，考虑机器学习中决策树模型，以信贷风险为信息熵，构造贷款决策树，并进行灵敏度分析，若提供贷款，则进一步考虑做出贷款分配方案。

针对模型三，我们搜集在新冠疫情的影响下不同行业的进项税、销项税等数据变化，利用短期时间序列分析ARMA模型，首先对数据进行探索分析，而后建造模型，对某企业的日价税合计做了预测；属于多元线性回归模型，随机变量价税估计的取值不仅与前p期的序列值有关还与前q期的随机扰动有关。

关键词：量化分析 信用转移矩阵 层次分析法 主成分分析 决策树 ARMA模型

一、问题重述

问题背景：

统计数据显示，当前我国中小微企业有近 4000 万家，占我国企业总数的99%左右，占我国 GDP 总量的 60%。但一方面，由于小微企业能否按照合同要求归还贷款主要依赖于自身的经营状况，并且小微企业的风险抵抗能力较弱，导致商业银行信贷风险增加；另一方面，随着国内金融市场的逐步完善以及「互联网+ 商业银行」时代的来临，利率市场化进程也在稳步推进，商业银行所面临市场环境的不确定性增加。不管是历史传统上，还是中小银行的金融产品开发和提供能力上，中小银行和当地中小微企业都是最搭的。他们彼此之间不仅互相熟悉，而且互相成就，彼此之间是鱼和水之间的关系。当中小微型企业做强了，中小银行的效益自然会上涨。因此有针对性的破解中小微企业融资难问题，是一个极为重要的问题。

让中小企业享受政策红利，对中小银行的风险还是比较大的，因为具有比较高的挑战性。同时，监管单位必须要在科学把握中小银行风险特征的前提下，才能扬长避短，实现扶持中小微企业的目的。与此同时，如果遇到突发情况，中小微型企业的应对能力不足够强，也会进一步导致商业银行的信贷风险增加。因此，有必要分析银行与中小微型企业之间的系统风险模型，虽然这种风险是普通企业暴雷所不可避免的，但是模型的预测分析有助于提前减少损失。

经分析，需要解决的问题有：

问题一：根据附件1和3中的123家企业的信誉等级和进销项税发票等数据，协助银行对各个企业做出贷款金额和利率决策；

问题二：根据附件2对302家位置信誉等级的企业单位，依据所给出的行业信息依据进销项税发票数据，协助银行对各个企业做出贷款金额和利率决策；

问题三：考虑面临突发情况，比如2020年新冠疫情，不同领域的企业会受到哪些影响，结合企业的收益情况以及抗风险能力的评估，协助银行对各个企业进行新一轮的贷款金额和利率决策。

二、问题分析

针对问题一，本文建立企业的实力、信誉、供给关系与贷款金额和利率的递归关系模型，利用层次析方法，确定各个自变量的在层次内的权重，构造以贷款金额为因变量的多元函数，同时结合附件3中所给信誉等级流失率，给出相应的利率调整区间；

针对问题二，本文采取聚类分析的方式，将未知信誉等级的企业经过机器学习中典型的决策树模型，判定其所属类别，从而借用模型一的贷款金额多元函数，同样结合银行贷款利率与客户流失率的关系，做出相应的利率决策；

针对问题三，我们搜集在新冠疫情这一突发情况下，不同行业的企业遭受的进项税、销项税数据信息，利用短时期的时间序列分析 ARMA 模型，依据随机变量价税估计的取值不仅与前 p 期的序列值有关还与前 q 期的随机扰动相关性，对日价税估计，即此时企业的唯一甄选指标----企业实力进行评估，从而做出信贷决策。

三、模型假设与约定

根据题目中条件作出假设

A.2017年下半年到2019年下半年，没有突发情况的发生；

B.题目限定贷款期限为一年，故而不必考虑贷款期限与中小微型企业信贷风险评估值之间的影响关系；

根据题目中要求作出假设

A.对于附件1和附件2中的作废发票,采取去除处理,对于负数发票作为参考数据加入分析评估过程,与有效发票的价税合计进行对冲平衡；

B.在模型一和模型二中上下游企业之间的链式关系在没有突发事件发生的情况下不予考虑；

约定：

1.信用贷款指的是债务人不需要提供任何抵押品、不需要存在第三方担保，仅凭借信誉就可以取得贷款，并且以借款方的信誉程度作为还款保证，不考虑此间贷款所需的其他信息；

2.在本文处理的问题中，中小微型企业信用贷款是指没有担保，仅依据企业自身的信誉作为银行对企业贷款金额的衡量指标；

3.信用贷款风险指的是在银行贷款中借款者违约的风险，不考虑投资债券；

4.信用贷款风险不以人的意志为转移，不考虑贷款风险随着信贷金额的扩张收缩存在的周期性周而复始的恶性循环；一般认为这种风险，对于银行来说是可控的；

四、符号说明及名词定义

数据量	符号
Strength	企业实力
Input_VAT	进项税合计
Output_VAT	销项税合计
θ_1	进项发票频数标准差
θ_2	销项发票频数标准差
Credit	企业信誉
Evaluate	信贷风险
difference_value	进销税差

表 1

五、模型建立

5.1 模型一

5.1.1 数据预处理

利用Excel筛选出无效发票并剔除,再用RStudio读入附件1转成csv格式的数据,筛选出2017-2019年的数据,按月统计每个公司的销项发票数量与进项发票数量,并为每个公司计算出销项进项发票数量标准差。再按月统计每个公司销项进项价税合计之差，并为每个公司计算出月价税合计之差的标准差。对于信誉等级A、B、C、D，令A:100，B:80，C:60，D:40,将等级转化为分数,如果该公司有违约,那么将等级分数-40分。

因为销项进项发票数量标准差月价税合计差的标准差、信誉等级分数的单位不一致,所以利用 Z-score 公式进行标准化。

5.1.2 自变量量化分析

设定自变量企业实力，供求关系，信誉，对这 3 个变量依次分析：

5.1.2.1 企业实力

企业实力，一般由财力、生产能力、技术水平、管理水平、销售能力等决定。依据附件已有数据，则以财力作为衡量指标。根据发票的金额大小依据发票的频率来判断，其中每一家公司都有销项税和进项税，因此进销税差 difference_value 有如下计算公式：

$$\text{difference}\_value = \sum_{1}^{30} Output\_VAT - \sum_{1}^{30} Input\_VAT$$

即对于每家企业，遍历当月所有日期的价税合计金额并作差。例如合计公司 E1 的 2017.7-2019.12 的价税合计表，以半年为单位长度，绘制出如下折线图：

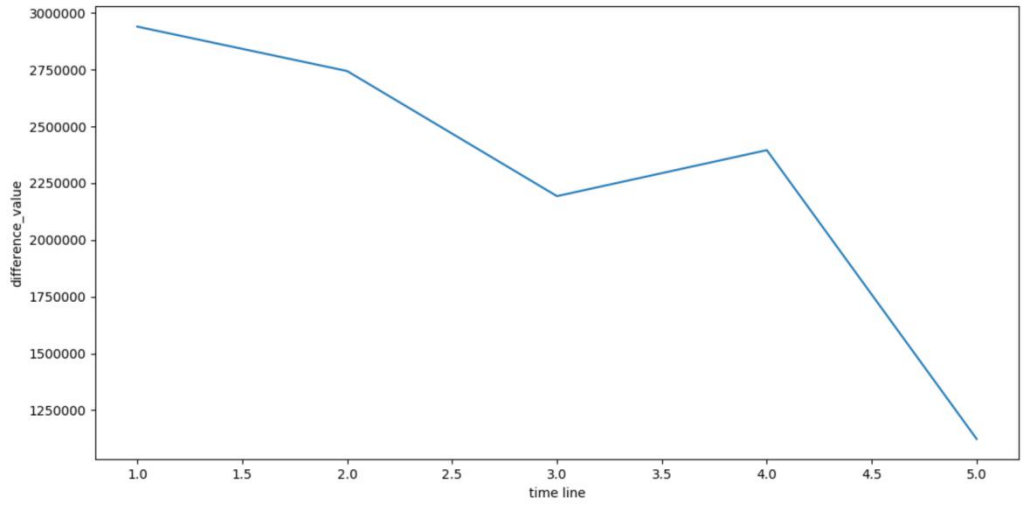


图 1

其中时间线最左侧是 2017 年 7 月，最右侧是到 2019 年 12 月。图形的曲线

变化显示了公司 E1 随着时间变化价税合计的变化情况。(代码见 test_data.py)

根据机器学习中的监督学习算法，我们考虑将附件一 123 组数据进行分组，训练集和测试集，第一组占据百分之七十属于训练集 train_data，第二组百分之三十作为测试集 test_data。对第一组的每一个企业做如 E1 的相同处理。

根据各个公司自身的月度进销税差 difference_value 随时间的变化曲线可知其为离散型数据，符合 Logistics 模型使用条件。

Logistics 模型求解的本质上是通过构建似然函数，让似然函数对各个参数的偏导等于 0，得到使似然函数值最大时各个参数的值。

将极大似然法应用于企业的实力评估模型中，对应的物理意义是具体的参数未知的作为随机样本的公司满足某种概率分布，通过已知的数据进行若干次试验，观察并利用结果推出参数的估计值。已知某个参数能使这个样本的价税合计出现的概率最大，我们就不会再去选择其他小概率的样本，所以干脆就把这个参数作为估计的真实值。

在本文的论述方案中，企业实力评估问题，响应变量不一定就是0和1两种情况，响应变量是一端区间内变化的值，故而引入了多分类Logistic回归模型．记Y是一个响应变量价税估计有C个取值，从0取到(C-1)，并且Y=0是一个参照组，协变量X时间序列 $X=(x_1,x_2,.....x_p)$ ，那么可以得出y的条件概率:

$$P(y=k|x)=\frac{e^{g_k(x)}}{1+\Sigma_{j=1}^{C-1}e^{g_j(x)}}$$

其中 k=0,1,2, ,c-1.由此可以得出相应的 Logistic 回归模型

$$g_k(x)=ln[\frac{P(y=k|x)}{P(y=0|x)}]$$

$$=\beta_{k0}+\beta_{k1}x_1+...+\beta_{kp}x_p$$

最小二乘估计在这里我们对 y 的每一个取值都进行 n 次独立观测,为了方便,我们把所有 y 的观测均写到一起,可以得到如下的矩阵方程

$$\begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1,c-1} \\ y_{21} & y_{22} & \cdots & y_{2,c-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{n,c-1} \end{pmatrix}=\begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}\begin{pmatrix} \beta_{10} & \beta_{20} & \cdots & \beta_{c-1,0} \\ \beta_{11} & \beta_{21} & \cdots & \beta_{c-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{1p} & \beta_{2p} & \cdots & \beta_{c-1,p} \end{pmatrix}$$

$$记Y=\begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1,c-1} \\ y_{21} & y_{22} & \cdots & y_{2,c-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{n,c-1} \end{pmatrix}$$

$$X=\begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}$$

$$B=\begin{pmatrix} \beta_{10} & \beta_{20} & \cdots & \beta_{c-1,0} \\ \beta_{11} & \beta_{21} & \cdots & \beta_{c-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{1p} & \beta_{2p} & \cdots & \beta_{c-1,p} \end{pmatrix}$$

令 B=($\beta_1,\beta_2..\beta_{c-1}$), z 则有 Y=XB 成立.

为了运算的方便,把 Y 和 B 按列拉直,故可以得到:

$$\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_{c-1} \end{pmatrix}=\begin{pmatrix} X & 0 & \cdots & 0 \\ 0 & X & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & X \end{pmatrix}\begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{c-1} \end{pmatrix}$$

令$Y'=(Y'_1,Y'_2..Y'_{c-1}),\beta'=(\beta'_1,\beta'_2..\beta'_{c-1})$，于是可以得到$\beta$的最小二乘估计:

$$\beta=\left|X'X\right|^{-1}X'Y$$

为了构造似然函数,我们使用二进制编码 0 和 1 来表示观测组的值,这些编码的引入仅仅是为了解释似然函数,我们规定,如果 y=0 那么 $y_0=1,y_1=0,y_2=0,\ldots,y_{c-1}=0$,如果 y=1 那么$y_0=1,y_1=1,y_2=0,\ldots,y_{c-1}=0$,依次 1 下去有,如果 y=c-1 那么$y_0=0,y_1=0,y_2=0,\ldots,y_{c-1}=1$.可以得出无论 y 取什么值,总有 $\sum_{j=0}^{c-1}y_j=1$ 成立,在这里我们考虑 n 个独立的观测样本记作(x_i,j_i) = 1,2,.,. n 利用上面的规定,得出如下的似然函数:

$$l(\beta)=\prod_{i=1}^{n}[\pi_0(x_i)^{y_{0i}}\pi_1(x_i)^{y_{1i}}\pi_2(x_i)^{y_{2i}}\cdots\pi_{c-1}(x_i)^{y_{c-i}}]$$

$$= \prod_{i=1}^{n} \{ \prod_{j=0}^{c-1} [\pi_j(x_i)^{y_{ji}}] \}$$

其中$\pi_j(x_i) = P(y = j|x_i)$，对上式两端取对数,并且借助于对$\sum_{j=0}^{c-1} y_j = 1$每个 i 均成立,可以得到如下的对数似然函数

$$\begin{aligned} L(\beta) &= \ln\left(\prod_{j=0}^{c-1}[\pi_j(\mathbf{x}_1)^{y_{j1}}]\right) + \ln\left(\prod_{j=0}^{c-1}[\pi_j(\mathbf{x}_2)^{y_{j2}}]\right) + \cdots + \ln\left(\prod_{j=0}^{c-1}[\pi_j(\mathbf{x}_n)^{y_{jn}}]\right) \\ &= \sum_{i=1}^{n} y_{0i} \ln[\pi_0(\mathbf{x}_i)] + \sum_{i=1}^{n} y_{1i} \ln[\pi_1(\mathbf{x}_i)] + \sum_{i=1}^{n} y_{2i} \ln[\pi_2(\mathbf{x}_i)] \\ &+ \cdots + \sum_{i=1}^{n} y_{c-1i} \ln[\pi_{c-1}(\mathbf{x}_i)] \\ &= \sum_{j=0}^{c-1}\sum_{i=1}^{n} y_{ji} \ln[\pi_j(\mathbf{x}_i)] \end{aligned}$$

由于$\ln[\pi_j(\mathbf{x_i})] = \ln[P(y = j|\mathbf{x_i})] = \ln\frac{e^{g_j(\mathbf{x}_1)}}{1+\sum_{j=1}^{c-1} e^{g_j(\mathbf{x_i})}}$,于是

$$\begin{aligned} L(\beta) &= \sum_{j=1}^{c-1}\sum_{i=1}^{n} y_{ji} g_j(\mathbf{x_i}) - \sum_{i=1}^{n} \ln\left[1 + \sum_{j=1}^{c-1} e^{g_j(\mathbf{x}_1)}\right] \\ &= \sum_{i=1}^{n}\left[\sum_{j=1}^{c-1} y_{ji} g_j(\mathbf{x}_i) - \ln\left(1 + \sum_{j=1}^{c-1} e^{g_j(\mathbf{x}_1)}\right)\right] \end{aligned}$$

对$L(\beta)$关于(c-1) x (p+1)个未知参数β_{jk}求一阶偏导,似然方程就可以得到.为了书写方便,我们令$\pi_{ji} = \pi_k x_i$似然方程的一般形式如下

$$\frac{\partial L(B)}{\partial \beta_{jk}} = \sum_{i=1}^{n} x_{k_i}(y_{ji} - \pi_{ji})$$

而 j=1,2,...,c-1.且 k=0,1,2,...,p.并且$\pi_{ji} = \pi_j(x_i)$对每一个 i 均成立.再对上式关于$\beta_{jk'}, \beta_{jk'}$求偏导,那么 L(3)求二阶偏导后的形式如下:

$$\frac{\partial^2 L(\beta)}{\partial \beta_{jk} \partial \beta_{jk'}} = -\sum_{i\pm 1}^{n} x_{k'i} x_{ki} \pi_{ji} \cdot (1 - \pi_{ji})^1$$

且

$$\frac{\partial^2 L(\beta)}{\partial \beta_{jk} \partial \beta_{j'k'}} = \sum_{i=i}^{n} x_{k'i} x_{k_i} \pi_{ji} \pi_{j'i}$$

而 j,j',= 1,2,...,c-1.且 k' = 0,1, 2, ...p. 那么 Fisher 信息阵$I(\widehat{\beta})$,是. (c-1)(p+1) x(c-1)(p+ 1)的矩阵,矩阵中的元素是等式(2.5)和(2.6) [上面两个式子] 关于$\widehat{\beta}$的相反数,极大似然估计的协方差矩阵的估计是信息阵的逆.

$$Var(\widehat{\beta}) = I(\widehat{\beta})^{-1}$$

则 Fisher 信息阵可以表示成如下形式:

$$I(\widehat{\beta}) = \begin{pmatrix} \hat{I}(\widehat{\beta})_{11} & \hat{I}(\widehat{\beta})_{12} \ldots \hat{I}(\widehat{\beta})_{1,c-1} \\ \hat{I}(\widehat{\beta})_{21} & \hat{I}(\widehat{\beta})_{22} \ldots \hat{I}(\widehat{\beta})_{2,c-1} \\ & \ldots \\ \hat{I}(\widehat{\beta})_{c-1,1} & \hat{I}(\widehat{\beta})_{c-2,2} \ldots \hat{I}(\widehat{\beta})_{c-1,c-1} \end{pmatrix}$$

其中

$$\hat{I}(\widehat{\beta})_{jj} = (X' V_{jj} X)$$

并且

$$\hat{I}(\widehat{\beta})_{jj'} = \hat{I}(\widehat{\beta})_{j'j} = -(X' V_{j'+j} X)$$

剔除第 k 个观测之后进行极大似然估计，考虑对数似然函数

$$L(\beta) = \sum_{i=1, i\neq k}^{n} \left[\sum_{j=1}^{c-1} y_{ji}\, g_j(\mathbf{x}_i) - \ln\left(1 + \sum_{j=1}^{c-1} e^{g_j(\mathbf{x}_1)} \right) \right]$$

对自变量β二次求导，有

$$-\frac{\partial^2 L(\beta)}{\partial \beta \partial \beta'} = \mathbf{X}' V \mathbf{X} - \mathbf{X}_k \mathbf{v}_k \mathbf{X}_k' = Z'Z - \mathbf{z}_k \mathbf{z}_k'$$

以初始值$\beta^{0(k)}$开始，首先近似到$\beta^{\wedge 1(k)}$，β的估计值建立在剔除第 k 个观测值的基础上，可以得到如下的表达式：

$$\beta^{\wedge 1(k)} = \beta^{0(k)} + (Z'Z - z_k z_{k'})^{-1}(\mathrm{X}'S - X_k s_k)$$

使用同样的方法得到初始值

$$\beta^{0(k)} = (X'X - X_k X'_k)^{-1}(X'Y - X_k Y_k)$$ 。

整理之后得到如下表达式：

$$\beta^{0(k)} = \beta^0 - (1 - h_{kk})^{-1}(X'X)^{-1} X_k e_k$$

而$e_k = Y_k - X'_k\beta^0$，且$h_{kk} = X'_k(X'X)^{-1}X_k$

在上述假设下，可以得到如下结论：

$$\widehat{\beta}^{1(k)} = \widehat{\beta}^1 - \left(1 - h^*_{kk}\right)^{-1}\left(Z'Z\right)^{-1}\mathbf{z_k}e^*_k - (1 - h_{kk})^{-1}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{x_k}e_k$$

$$e^*_k = \left[\frac{s_k}{\sqrt{v_{kk}}} - \mathbf{z}'_{\mathbf{k}}\left(Z'Z\right)^{-1}Z'V^{-1/2}S\right]_{\beta=\beta^0} \text{ 且 } h^*_{kk} = \mathbf{z}'_{\mathbf{k}}\left(Z'Z\right)^{-1}\mathbf{z_k}$$

由此得出参数β的极大似然值，然后依据价税合计与时间序列和β的函数得出 Y 的估计值。

将得到的β最大值带入到第二组验证数据，依据 X 矩阵和 B 矩阵求解 Y 矩阵，得到对应公司的实力极大似然估计值 Estimate，与实际价税估计值 Actual 做比较，发现模拟效果较好。

5.1.2.2 供求关系

供求关系，所谓的供求关系稳定主要指的是企业开票频率以及金额波动是否异常，从这两个方面分析企业稳定性。在处理过程中，统计各个企业的月度进项发票频数以及销项发票频数，并计算两者的标准差，使用 Z-score 标准化后得到供求稳定性偏差系数分别为θ_1, θ_2。

5.1.2.3 企业信誉

企业信誉，在附件信息中定义有信誉等级 A B C D，对于信誉等级 A、B、C、D，令 A:100，B:80，C:60，D:40,将等级转化为分数,如果该公司有违约,那么将等级分数-40 分。

5.1.3 模型一的建立

5.1.3.1 信贷风险评估

由数据量化处理结果得{E1,E2,.....E123}的实力值（Strength）为一维数组 Y，以及信誉（Credit）一维数组 X，绘制散点图如下：

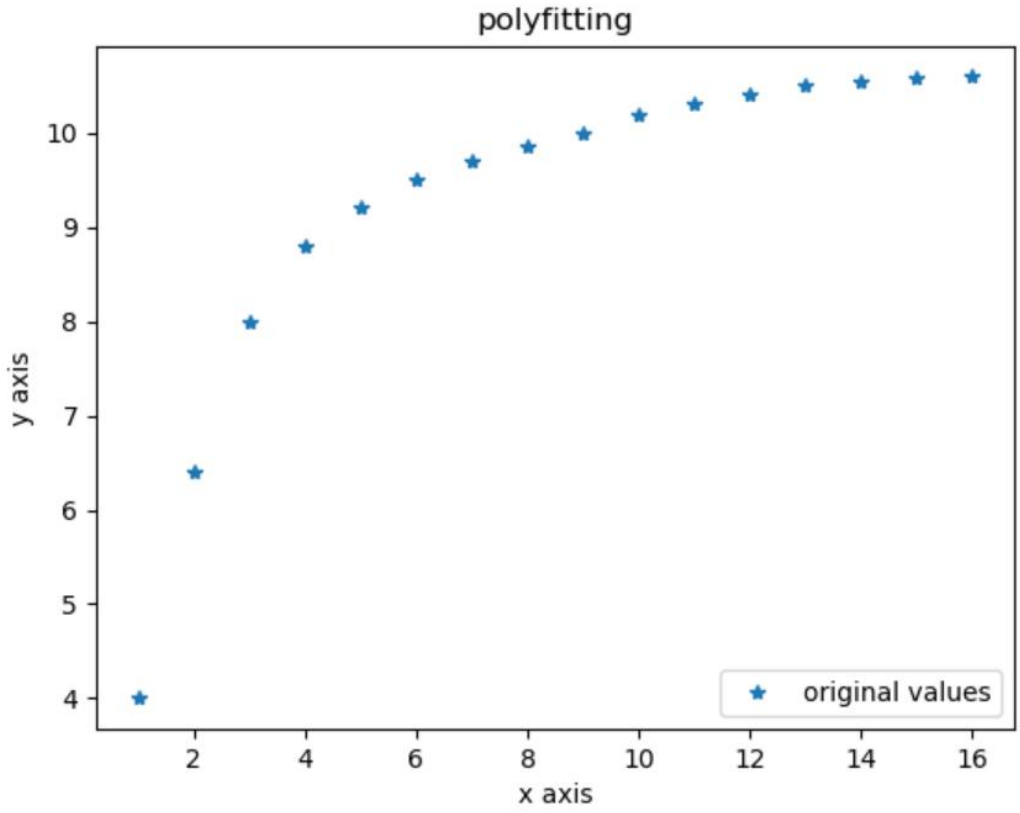


图 2

其中横坐标走势表示企业信誉，纵坐标走势表示企业信实力，分析数据及图表发现，信贷风险评估 Evaluate 这一变量里，企业信誉和实力占据不同的权重，故而我们使用主成分分析 PCA 方法，求解二者的近似权重系数，得出关系函数如下：

$$Evaluate = 0.7 * Strength + 0.3 * Credit$$ （程序见 strength_metrix.py）

5.1.3.2 利用实力和供求关系判断是否贷款

在对供求关系进行量化分析的过程中，针对每一个企业的进项发票频率和销项发票频率分别以月为单位，进行统计分析，例如 E1 公司的进项发票数量月变化趋势图如下左图，销项发票数量月变化趋势图如下右图所示：

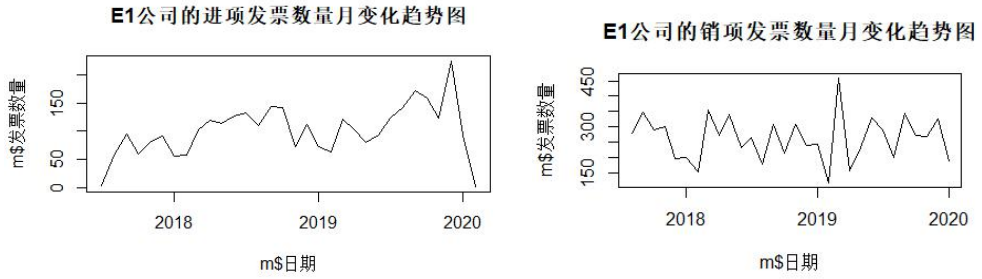


图 3

对于 E1 公司的进项发票频度分布，求出方差θ_1，销项发票频度分布，求出方差θ_2，由于方差的大小与稳定性呈现反比关系，故而取$\frac{1}{\theta_1}$和$\frac{1}{\theta_2}$作为是否贷款的影响自变量，实现正相关变化；同时结合企业的实力 strength，依据三者的数据变化，利用层次分析法确立系统中各个因素之间的关系，构造判断矩阵，由判断矩阵计算比较元素之间的相对权重，并进行一致性检验。

得到自变量之间的隐含关系如图所示：

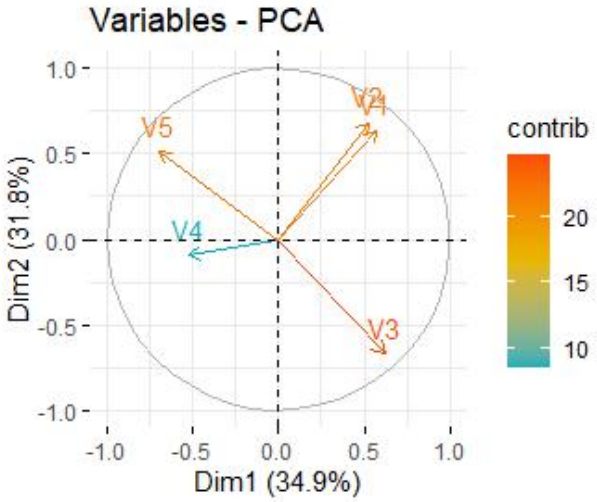


图 4

均分占比数值化结果如下：

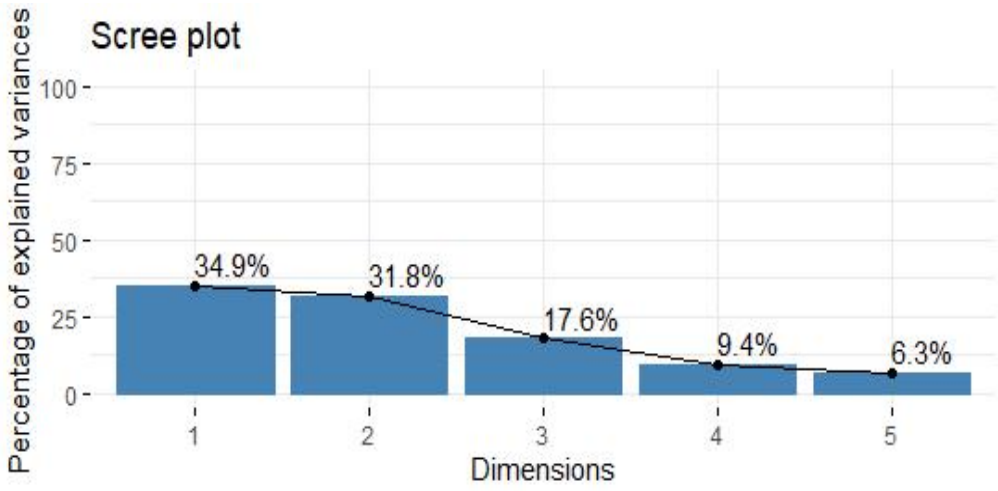


图 5

其中 Dim1 指的是进项发票频率占比，Dim2 指的是销项发票频率占比，Dim4 指的是企业的实力占比，按照分比关系,得到是否提供贷款判断函数为：

$$\mathrm{L} = 0.45 * \frac{1}{\theta_1} + 0.41 * \frac{1}{\theta_2} + 0.14 * strength$$

分析数据得出 L 的变化曲线，并对函数进行二次求导，得出 L 的分界线 Limitation 为 67.78，即说明当企业的 L 值处于 Limitation 之下，可能由于信誉过低、偿还能力不足等等问题，从而取消银行贷款资格；反之，L 值高于 Limitation 的公司，则首先取得贷款的资格。

5.1.3.3 利用信誉和信贷风险评估明确贷款金额以及年利率

类比上面的处理过程,对企业荣誉 Credit 和信贷风险 Evaluate 做相同的处理，具体见程序模型一数据处理代码.Rmd,求解得到贷款金额与二者的关系函数近似为：

$$\mathrm{Loan\_}amount = 0.682 * Evaluate + 0.318 * Credit$$

即在贷款金额的确定过程中，企业的信誉占 31.8%的影响比例，银行的对该企业的风险评估占约 68.3%的影响比例。

最终求得 123 家中可贷款小微型企业的 Loan_amount,按照比例对将贷款总额进行分割，逐次借贷给企业。

各个企业的贷款金额为：

$$Fund\_E_i = TotalFund \times \frac{Loan_{amount_i}}{\sum_{j=1}^{n} loan_{amount_i}}$$

$$n = 123, i = 1,2,\ldots 123$$

同时，为了确定贷款的年利率，结合附件三银行贷款年利率与客户流失率关系的统计数据，绘制年利率与客户流失率的关系如下：

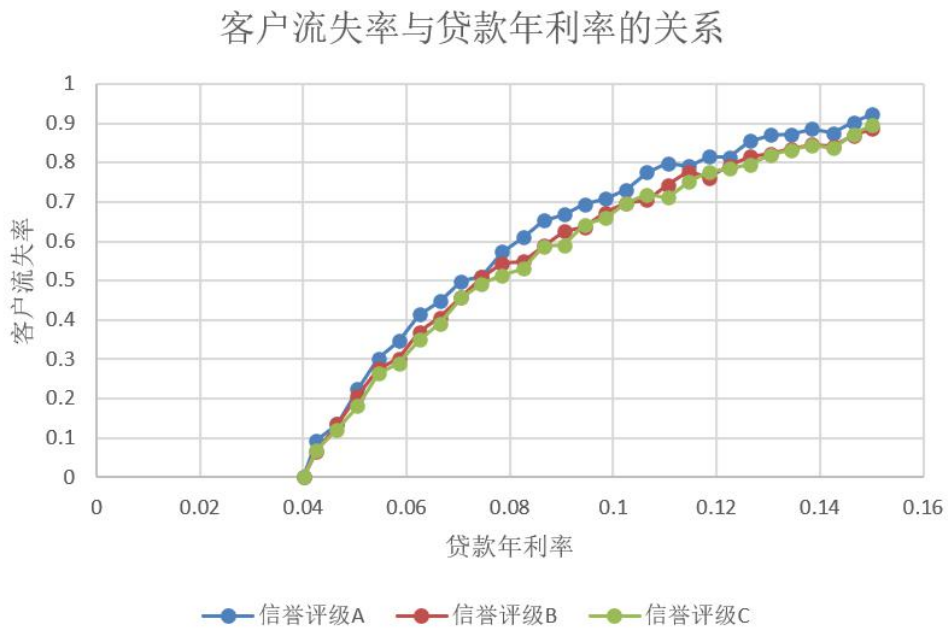


图 6

其中信誉评级 A、B、C 对应的多项式函数拟合曲线为：

$$y_A = -76.41x^2 + 21.984x - 0.6971$$

$$y_B = -67.933x^2 + 20.207x - 0.6504$$

$$y_C = -63.942x^2 + 19.569x - 0.6393$$

观察图像并结合函数可得，信誉等级为 A 的企业确定其利率区间为[0.04，0.0705],信誉等级为 B 的企业确定其利率区间为[0.0625,0.0865],信誉等级为 C 的企业确定其利率区间为[0.0785,0.1025]，信誉等级 D 认定为没有贷款资格的企业，不予考虑。

5.2 模型二

5.2.1 用于信誉等级划分的机器学习算法对比

由于附件 2 中的 302 家企业没有信誉信息，而模型一中的 123 家公司有注明响应的信誉等级，根据这些公司的进项交易票据、销项交易票据以及上下游企业等因素，结合机器学习中四种分类算法，综合对比得到性能预测准确度结果如下：

决策树	0.725274
逻辑回归	0.516483
K 近邻	0.486842
朴素贝叶斯	0.241758

表 2

因此，决定采取决策树算法对 302 企业的信誉进行分类分析，将他们的信誉等级归为 A B C D 中一个，以信贷风险评估值为信息熵筛选标准。

即决策树的流程处理框架如下：

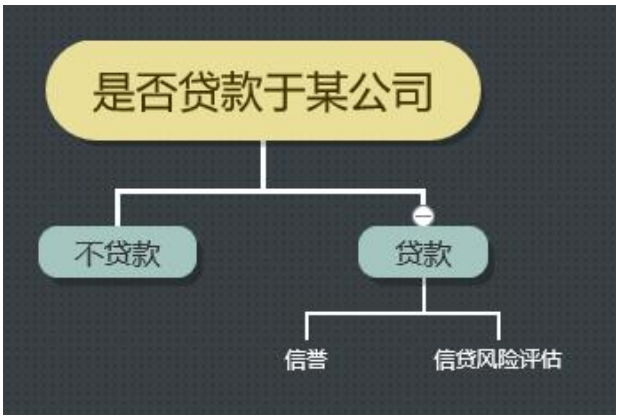


图 7

5.2.2 决策树模型预测信誉等级

开始，所有企业作为一个记录结点；遍历企业的价税合计、供求关系特征，以信贷风险为分裂特征点；分裂为两个节点，不贷款和贷款；对分裂后的结点继续执行 2-3 遍，直到足够“纯”为止；具体程序见附录 HD_DecisionTree.py

这里的特征结点的信息增益越大，表明特征对样本的熵的减少能力更强，这个特征使得数据由不确定性到确定性的能力越强。

分类预测结果如下图显示：

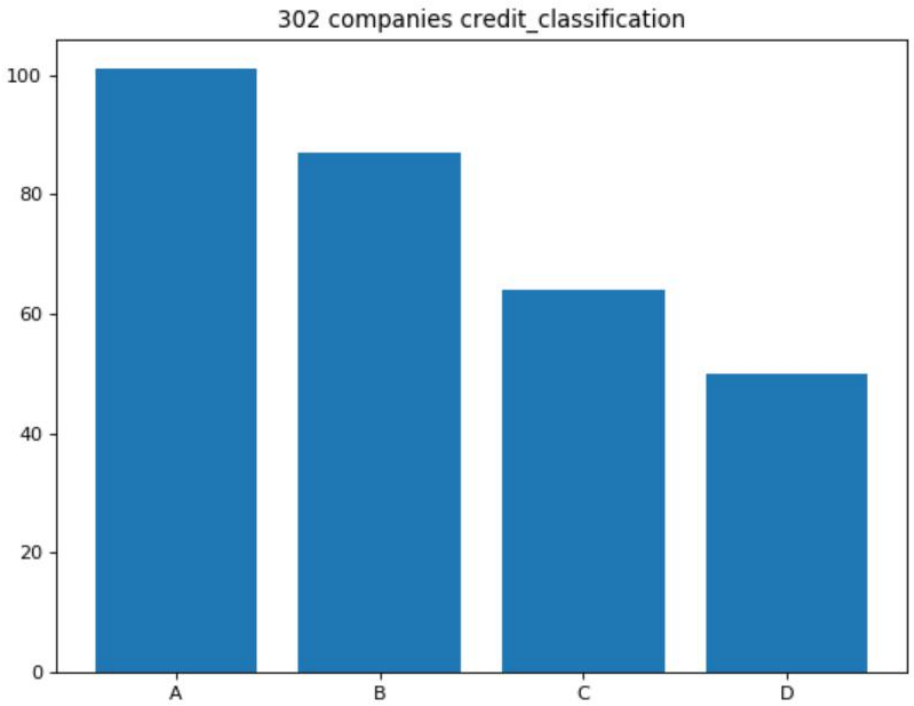


图 8

其中信誉等级 A、B、C、D 对应的企业数量依次为 101,87,64,50。

5.2.3 贷款决策

此时 302 家中小微型企业都具有了各自的信誉等级，分析过程与模型一相同，借用风险评估函数进行风险预测，根据 $\mathrm{L}=0.45*\frac{1}{\theta_1}+0.41*\frac{1}{\theta_2}+0.14*strength$ 函数，确立其 Limitation2；然后在确立贷款的基础上，套用 Loan_amount 函数，得出公司的贷款值，根据贷款值之间的比例关系，对一亿元进行按比分配贷款金额；

贷款金额为：

$$Fund\_E_i = TotalFund \times \frac{Loan_{amount_i}}{\sum_{j=1}^{n} loan_{amount_i}}$$

$$n = 302, i = 1,2,...302$$

对应贷款年利率与上述模型一处理过程相同，此处不再赘述。

5.3 模型三

5.3.1 突发因素影响企业的现实背景

企业的生产经营和经济效益可能会受到一些突发因素影响，比如新冠疫情、洪涝灾害、地质灾害等因素，并且突发因素往往对不同行业的企业会产生不同的影响。

比如 2020 年年初的突发事件新冠疫情对于中小微型不同领域的企业产生了不同程度的影响。查阅统计数据得出如下结论：

1. 2020 年第一季度约 20%的企业被客户取消订单较多；酒店、餐饮、旅游行业、中介服务、建筑建材和房地产行业营收大幅下降比例较高；

2. 企业普遍面临比较大的租金、工资、税费等综合成本压力：小型企业面临的综合成本更大，民营企业面临的综合成本压力显著高于国有企业和外资企业；

3. 企业面临原料价格普遍上涨，统计数据显示面临原料价格大幅上涨的企业占比 10.69%，小幅上涨的占据 55.2%，合计越达到 66%；

4. 一旦企业供应发生中断，有 32.6%的企业面临难以找到替代的服务商，说明大部分企业供应链缺乏韧性；存在四分之一的企业可能因原料库比较少而无法保重生产的稳定性；

企业受突发因素影响，可能会出现上下游之间的供应不足、原料生产不足、销量变化等情况，可推断出在本文模型选取的自变量中，进项税、销项税将受到不同程度的影响，供求关系也会发生变化。

我们调查 2020 年不同企业销项税、进项税第一季度 1-3 月份数据，突发情况下，不同公司的长期的供求能力和荣誉不再因变量主导地位，因此，在本文解决方案中金保留价税合计作为企业实力的衡量指标，选择使用短期时间序列分析 ARMA 模型。

5.3.2 ARMA 分析模型

本文目的是预测突发情况下银行信贷的分配情况，故可采用时间血泪分析法来分析价税合计的历史数据，并预测未来几天内或者几周内的变动情况，建模的总体流程如下图所示：

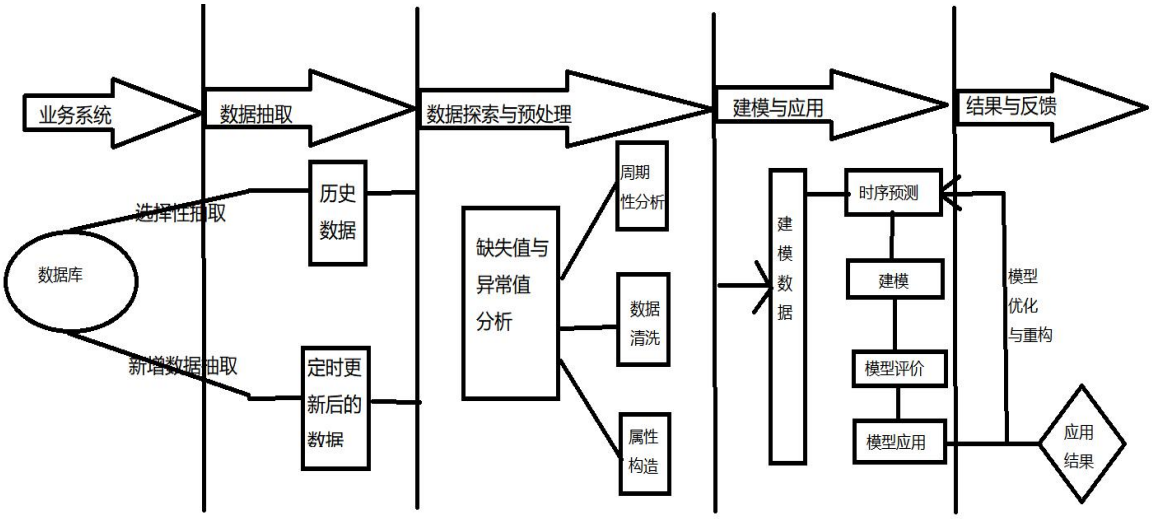


图 9

1. 分析方法与过程

本文的目的是预测任意一家公司遇到突发因素影响后的企业信贷风险量化指标，因此对于每家公司不仅需要考虑未受到突发因素影响的历史数据，也要新增近期受到影响后的短期新数据，并且所增数据需属于企业所在的行业及上下游行业，以此达到更为全面的信贷风险考量。

2. 数据抽取

从外部数据库抽取了时间段为 2020-01-24~2020-03-13 的每类行业的不同公司的进销项发票信息数据。（business.xlsx+ 公司分类.xlsx）

3. 数据探索与分析

A. 原始数据转换为时序数据

原始数据是每发生一次进项发票和销项发票行为就上传一次数据，为了生成每天的价税合计，需要将数据按日做 groupby，并求和，算出具体某一天的价税合计，同时删除不必要的属性，代码见 prepare.py 的 date_week_day 函数

B. 缺失值处理

从外部获取的数据可能出现发票信息缺失的情况，则将缺失值用均值代替，如果是日期缺失，则删除该记录。观察可知空缺值零散在各个月，对实际预测影响可忽略不计。prepare.py 的 get_datelist 函数。

4. 统计外部价税合计信息

首先，以每家公司每月发票销进项发票价税合计金额之差为纵坐标，以月份为横坐标单位，绘制条形图（具体程序参见 month_data 函数：

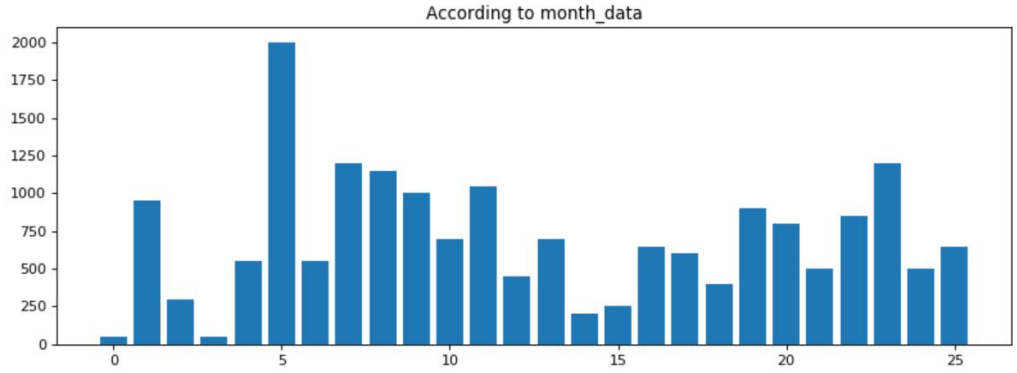
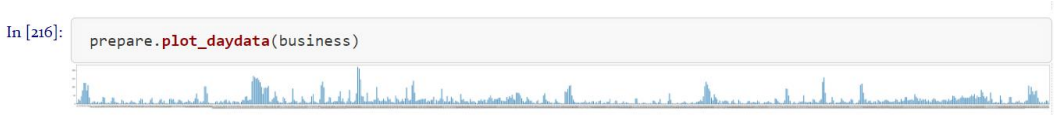


图 10

首先，以每家公司每月发票销进项发票价税合计金额之差为纵坐标，以月份为横坐标单位，绘制如下条形图，程序 prepare.py 的 plot_daydata 函数



观察发现，价税合计的进销项之差具有周期性，对于企业来说，还应该有一定的上升或者下降趋势，本文推断其趋势如下图：代码见模型一二的代码.Rmd

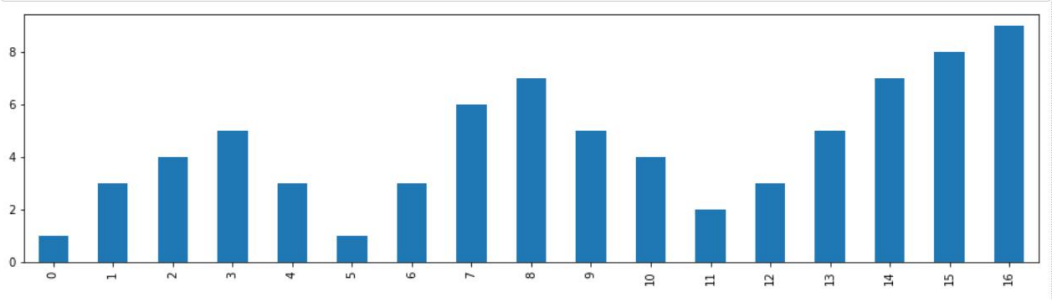


图 11

即这个时序序列估计是个平稳时序即序列在某一常数附近浮动且范围有限。

5. 模型构建

本小题价税合计的预测模型的建模历程：观测值序列->平稳性检测,若为 N，则进行差分运算，知道到达平稳为止；若为 Y，则继续下一步模型识别，经过白噪声检验和参数估计进行预测，得到预测结果后，做误差分析，对比实际值，若误差过大，则重新估计，反之，输出结果；

A. 平稳性预测

为了确定原始数据序列中是否存在随机趋势或未知的确定趋势，需要对数据进行平稳性检测，在检测之前，为避免极大值的干扰，将对价税估计做对数处理。具体见 prepare.pystationary_test 函数

执行结果为：

In [224]:
```
##stationarity test,测试样本7
prepare.stationarity_test(business_copy,7)
```

```
原始序列经过0阶差分后归于平稳，p值为0.00469951067314
```

这说明数据是平稳的时序序列，与我们的猜测几乎相同。

B. 白噪声检验

为了验证序列中是否是白噪声，如果是白噪声，即此序列都是随机扰动，则无法选择此模型进行判断和预测。具体程序见 prepare.py 文件中 whitenoise_test 函数

检验结果：

In [225]:
```
##whitenoise_test
prepare.whitenoise_test(business_copy,7)
```

```
原始序列为非白噪声序列，对应的p值为：7.81474158571e-60
一阶差分序列为非白噪声序列，对应的p值为：0.000493556298573
```

检验结果显示原始序列是非白噪声，故而可以使用 ARMA 模型。

C. 模型识别

有上面的平稳性监测和白噪声检验，确定原始序列是平稳非白噪声序列。下一步要做的就是计算平稳非白噪声序列的自相关系数 ACF 和偏自相关系数 PACF。

首先，画出原始序列的自相关图 ACF：

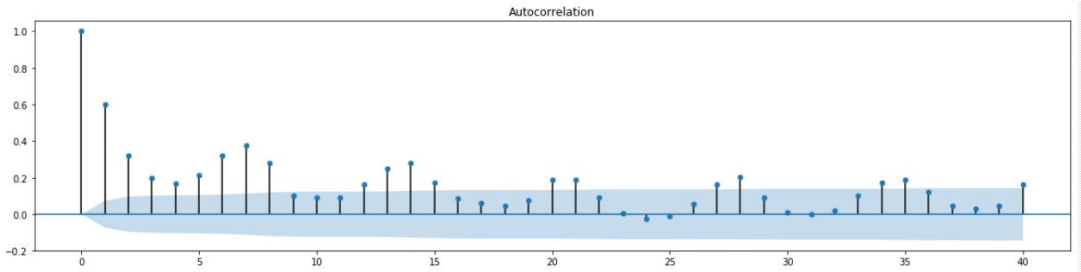


图 12

其次，画出原始序列的偏相关图 PACF：

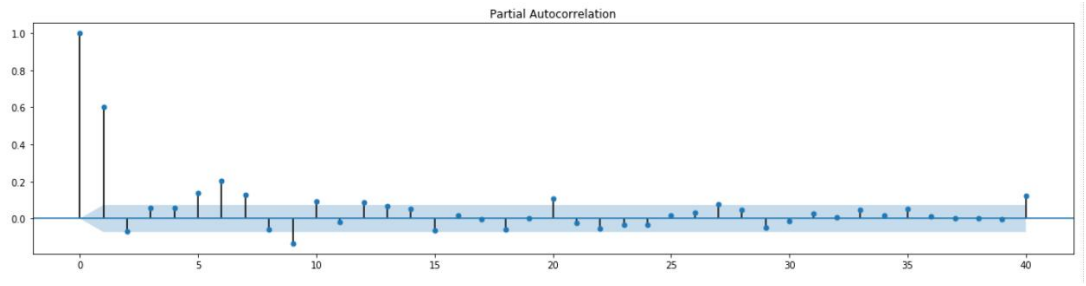


图 13

很明显自相关图显示出自相关图与偏自相关图均存在一定的拖尾性。本文这里使用模型 ARMA，并选取 p=1,q=1。

6. 拟合及预测

本文使用 fitting 模型并预测未来 21 天的数据，同时将预测数据与真实数据进行对比，误差用 RMSE 表示。 程序见 prepare.py 文件中 plot_results 和 arma_predict 函数

对比图形如下图：

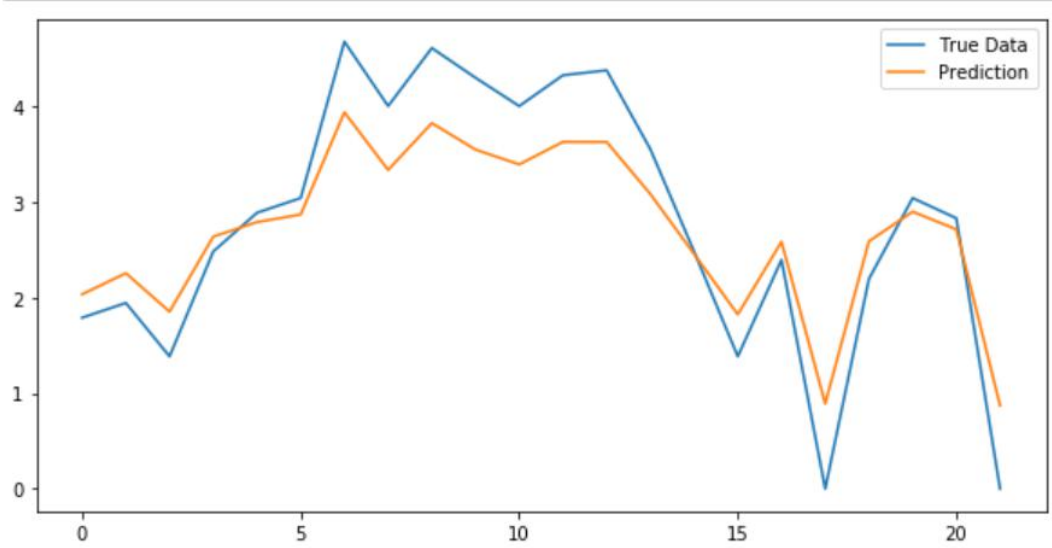


图 14

7. 误差分析

这里计算结果的均方误差为：

```
In [293]: RMSE

0.53236883500699539
```

图 15

由数学知识可得，随机短期时间序列函数：

$$Y_n = a_1 Y_{n-1} + a_2 Y_{n-2} + \ldots + a_p Y_{n-p} + \epsilon_n - c_1 \epsilon_{n-1} - c_2 \epsilon_{n-2} - \ldots - c_q \epsilon_{n-q}$$

其中，a_i 指的是最小二乘法参数估计所得常数项，c_i 指的是参数估计中二次项系数，ε 指的是参数估计所得误差项，Y_i 指的是回归模型因变量价税合计，即每一天的价税合计都与前期的价税合计信息相关。

根据 Y 函数，求解得出对企业的贷款额度，依据金额比例，对总资金进行比例分配，得到每家企业的贷款金额为：

$$Fund\_E_i = TotalFund \times \frac{Loan_{amount_i}}{\sum_{j=1}^{n} loan_{amount_i}}$$

$$n = \text{企业数量}, i = 1,2,\ldots n$$

对应贷款年利率与上述模型一处理过程相同，此处不再赘述。

5.3.3 总结

本小题主要采用了 ARMA 模型对某企业的日价税合计做了预测，其中详细描述了数据挖掘中建模的一般性流程，并结合具体实际系统数据做了分析，误差分析显示模型性能较好。该模型及分析问题也可以应用搭配销售数据预测，景区共享交通共用等实际场景中。

六、模型检验

模型一二主要建立了风险评估体系和决策方案，在理论很好的评估了中小微型企业信用贷款的风险、可以贷款额度的范围，反映了企业贷款风险与贷款额度和贷款利率之间的关系。本节选取企业代表的案例对此进行实例验证和分析。

6.1 实例企业的安全贷款额度测算

本文决策模型建立后，需要运用到实际生产环境中进行有效性检测，现选择了商业银行的一家贷款小微企业进行案例测试，为不泄露企业信息，用 X 企业代替。X 企业是一家批发型企业，根据企业的实际情况，按照文本的风险评估体系以及策略方案，针对思想指标进行风险评估及安全额度和利率的测算。

根据测算，X 企业的总体风险评估得分为 85.63，贷款利率为 0.465。

企业属于批发行业，年销项税和为 1500 万元，进项税总和为 350 万元，因此，根据企业实力和信誉计算风险值，作为核定项，公司可获得的贷款额度为 1500*46.5%-350=347.5 万元。因此该客户的安全贷款额度为 347.5 万元，或小于等于安全贷款额度，风险是比较可控的。

6.2 实例企业信用贷款额度和信贷风险的相关性分析

根据 A 企业的实际情况，应用加入额度控制的风险评估体系，逆向测算在不同的贷款金额下，产生的风险。得到如下图像关系：

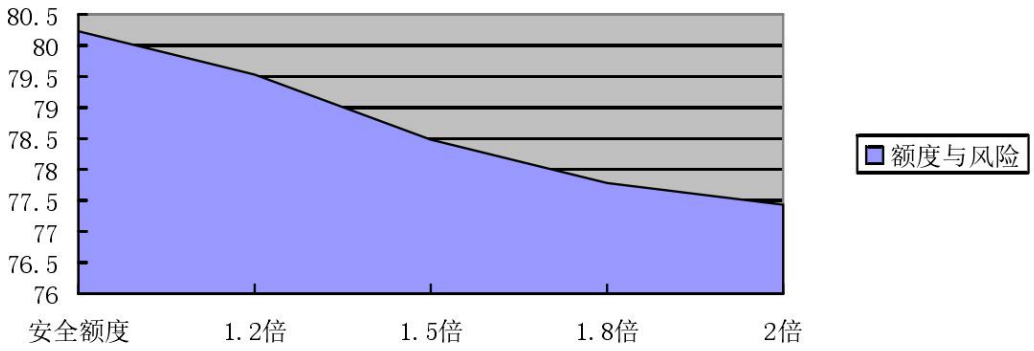


图 16

通过上图可以看到，随着贷款额度的增长，企业对应的信用风险评估值反之会不断下降，对于批发类企业，表格数据显示风险评估值等于 78 分时，该风险估项下的对应的贷款额度为最大可贷款金额，若超出该金额，则认为风险不再可控。

七、模型评价

特点：

在我们的模型中，主要特点是多次使用简约易懂的层次分析法，为了确定自变量在结果中的权重值，本文依据 AHP 建立层次结构模型，而后构造判断矩阵，最后进行层次单排序，从而确立与因变量之间的关系函数。在确立函数模型之后，抽取一定的外部数据集进行测试和一致性检验以及灵敏度分析。

在模型的检验模型中，本文分别讨论了模型一的灵敏度，合理地给出每个决策方案的每个标准的权数，利用权数求出各方案的优劣次序，检测其有效性；在第二个决策树分类模型中，对比其他机器学习分类算法，研究其稳定性；在第三小题的 ARMA 模型中，在建造模型后，通过预测和实际的对比，分析了误差影响。

最后，本文通过 AHP 函数关系以及信誉等级流失率，探讨了贷款额度、贷款利率的决策。

优点：

设计基于层次分析法对中小微型企业的信用贷款风险评估方法，结合商业银行对企业贷款的相关因素，甄取关键性指标，做出对企业的贷款金额以及利率的决策方案；

在体系方案的指标选择和设计，不仅采用附件所给信息的财务数据中的实力、信誉、价税估计的指标进行信贷风险评估，同时还通过大数据平台，搜集外部数据，作为测试集，包括有税务、法院、工商等中小微型企业的信息，设计了更适宜企业的风险评估模型和决策模型；

探讨了信用贷款额度与信用贷款利率这两个关键信贷指标与风险评估之间的关系，估算了突发事件下，短期信用贷款安全贷款额度，分析了彼此内部的影响关系，得到贷款额度、贷款利率与风险评估之间的量化关系；

缺点：

因银行支持中小微型企业的年限较短，流失的可能性较大，在自变量中没有加入这一因素，只是在最终贷款分配方案中结合了这一数据信息；

搜集的外部信息由于各家银行对于中小微企业违约的情况统计信息不完善，无法获取大量准确的历史数据，在层次分析法中，无法通过系数对相关矩阵进行系数调节；

没有考虑到上下游企业之前的产业链关系，比如疫情发生时，房地产行业急需原材料以搭建方舱医院，但可能出现原材料供给不足，无替代商等问题。

改进方法：

A.在更加友好的大数据平台上，搜集更加基层的数据信息作为支撑；

B.在简约方法的基础之上，稍微进一步甄选更多几个的自变量，可以更好的体现信贷风险与企业资产的内在关系；

C.依据行业名进行筛选，判断同一领域内的企业在突发情况下面临的问题和挑战，从而做出对某一行业的信贷决策方案；

八、模型推广

本论文建立的信贷决策模型能够通过对中小微型企业的信用贷款的风险评估研究，结合企业信誉做出贷款金额和利率的分配方案，能够在一定程度上解决目前银行对中小微型企业，由于风险高，风险可控手段有限而惜贷的社会现状。在银行贷款时，确定风险可控的基础之上，响应国家政策，扶持中小微企业的发展，促进经济的发展、民生的改善、社会的进步。

九、参考文献

[1] 党馨璐，基于层次分析法的小微企业信用贷款风险评估研究，2016，山东财经大学,37-44

[2] 葛美玲,多分类 logistic 回归及统计推断，2010，北京工业大学，10-18

[3] 任声策，调查报告：新冠肺炎疫情对我国企业复工和经营的影响,http://www.eeo.com.cn/2020/0221/376796.shtml.2020-02-21

十、附录

"模型一数据处理代码"

"2020/9/11"

```
    #这段代码用于计算销项发票金额与进项发票金额之差，每月趋势变化图
#读取数据文档
dat1<-read.csv("C:/Users//Desktop/11.csv",header = FALSE)
dat2<-read.csv("C:/Users//Desktop/22.csv",header = FALSE)
#简单清洗数据,分别获取处理后的进销项价税合计月总和
st<-"2020-01-01"#考虑到 2020 年数据量较少,且可能受到疫情影响,故删去
st<-lubridate::ymd(st)#转换为 Date 型数据
w1<-which(dat1$V3>=st)#找出 2020 以后的数据行数
w2<-which(dat2$V3>=st)#找出 2020 以后的数据行数
dat1<-dat1[-w1,]#删除指定行
dat2<-dat2[-w1,]#删除指定行
s1<-dat1[1:7]
s2<-dat2[1:7]
s1[1,1]<-"E124"
dat1<-s1
s2[1,1]<-"E124"
dat2<-s1
x1<-s1$V1
x1<-str_replace(x1,"E","")#去掉公司代号前的 E
s1$V1<-x1
x2<-s2$V1
x2<-str_replace(x2,"E","")#去掉公司代号前的 E
s2$V1<-x2
mon1<-lubridate::ymd(s1$V3)#把字符型日期转换为日期型
mon1<-as.Date(mon1)
s1$V3<-mon1
te1<-split(s1,s1$V1)#按照公司序号切割分组
mon2<-lubridate::ymd(s2$V3)#把字符型日期转换为日期型
mon2<-as.Date(mon2)
s2$V3<-mon2
te2<-split(s2,s2$V1)#按照公司序号切割分组

#计算每月价税合计之和的销项进项之差
ln<-list()#公司名称
ls<-list()#对应的销项进项之差

for(i  in  1:length(te1))
{
  ori1<-te1[[i]]
```

```
  #计算进项月总和
  date1<-ori1$V3
  tt1<-floor_date(date1,"month")#把所有日期归成当月 1 日
  tt1<-as.data.frame(tt1)
  sa1<-cbind(tt1,ori1$V7)#添加一列对应对的价税合计
  names(sa1)<-c("tt1","num1")
  m1<-aggregate(num1~tt1,data=sa1,FUN=sum)
  #并乘上-1,作为相反数
  m1$num1<-(-1)*m1$num1

  for(j  in  1:length(te2)){
    #计算销项月总和
    ori2<-te2[[j]]
    if(ori2$V1!=ori1$V1){
      next
    }

    date2<-ori2$V3
    tt2<-floor_date(date2,"month")#把所有日期归成当月 1 日
    tt2<-as.data.frame(tt2)
    sa2<-cbind(tt2,ori2$V7)#添加一列对应对的价税合计
    names(sa2)<-c("tt1","num1")
    m2<-aggregate(num1~tt1,data=sa2,FUN=sum)
  }


  #按月相加,即获得销项进项之差
  m<-rbind(m1,m2)
  m<-aggregate(num1~tt1,data=m,FUN=sum)

  #绘制图片
  #  names(m)<-c("日期","销项进项价税合计之差")
  #  plot(x=m$"日期",y=m$"销项进项价税合计之差",type='l',main  ="E1 公司销项进项价税
合计之差月变化趋势图")

  #求均值
  len<-length(m$num1)
  a<-as.numeric(a)
  a<-sum(a)
  eve<-a/len
  ln[[i]]<-(te1[[i]]$V1)[1]
  ls[[i]]<-eve
}
  ls<-data.frame(ls)
```

```
ln<-data.frame(ln)
ls<-t(ls)
ln<-t(ln)
l<-cbind(ls,ln)#合并公司编号和对应的标准差为一个数据
write.table(l,"C:/Users/  /Desktop/anss.csv",row.names=FALSE,col.names=TRUE,sep=",")


    #这段 R 代码用于绘制进项与销项发票的月数量变化曲线图
#library(lubridate)
#dat1<-read.csv("1.csv",header  =  FALSE)
#dat2<-read.csv("2.csv",header  =  FALSE)
#读取附件 1 的进项发票与销项发票信息的 sheet
#dat1 代表进项,dat2 代销项
#dat2<-dat2[-1,]
st<-"2020-01-01"#考虑到 2020 年数据量较少,且可能受到疫情影响,故删去
st<-lubridate::ymd(st)#转换为 Date 型数据
w<-which(dat1$V3>=st)#找出 2020 以后的数据行数
dat1<-dat1[-w,]#删除指定行
s1<-dat1[1:7]
dat1<-s1
s1[1,1]<-"E124"
x<-s1$V1
x<-str_replace(x,"E","")#去掉公司代号前的 E
s1$V1<-x
mon<-lubridate::ymd(s1$V3)#把字符型日期转换为日期型
mon<-as.Date(mon)
s1$V3<-mon
te<-split(s1,s1$V1)#按照公司序号切割分组
date<-te$"1"
date<-date$V3
tt<-floor_date(date,"month")#把所有日期归成当月 1 日
num<-rep(1,times=length(tt))#添加一列数字 1
tt<-as.data.frame(tt)
tt<-cbind(tt,num)#将日期和这列数字 1 合并成一个数据框
m<-aggregate(num~date(tt$tt),FUN=sum)#计算出每月有效的进项发票数量
names(m)<-c("日期","发票数量")
plot(x=m$"日期",y=m$"发票数量",type='l',main  =  "E1 公司的项发票数量月变化趋势图")


    #这段代码用于计算出 123 家公司的以月销量为最小计数点的总体月销量标准差
#若标准差为 NA，说明该公司只有一条记录，不足以用来计算标准差
#library(stringr)  加载字符串处理包
#library(lubridate)
dat2<-read.csv("C:/Users/  /Desktop/01.csv",header  =  FALSE)
```

```
st<-"2020-01-01"#考虑到 2020 年数据量较少,且可能受到疫情影响,故删去
st<-lubridate::ymd(st)#转换为 Date 型数据
#dat2<-dat2[-1,]#仅 dat2 需要此行
w<-which(dat2$V3>=st)#找出 2020 以后的数据行数
dat2<-dat2[-w,]#删除指定行
s1<-dat2[1:7]
dat2<-s1
s1[1,1]<-"E124"
x<-s1$V1
x<-str_replace(x,"E","")#去掉公司代号前的 E
s1$V1<-x
mon<-lubridate::ymd(s1$V3)#把字符型日期转换为日期型
mon<-as.Date(mon)
s1$V3<-mon
te<-split(s1,s1$V1)#按照公司序号切割分组
date<-te$"1"
date<-date$V3
tt<-floor_date(date,"month")#把所有日期归成当月 1 日
num<-rep(1,times=length(tt))#添加一列数字 1
tt<-as.data.frame(tt)
tt<-cbind(tt,num)#将日期和这列数字 1 合并成一个数据框
m<-aggregate(num~date(tt$tt),FUN=sum)#计算出每月有效的进项发票数量
names(m)<-c("日期","发票数量")

lst<-list()#创建一个列表来保存每家公司的月发票数量标准差
ls<-list()
for (i in 1:123){
da<-te[[i]]
da<-da$V3
tt<-floor_date(da,"month")#把所有日期归成当月 1 日
num<-rep(1,times=length(tt))#添加一列数字 1
tt<-as.data.frame(tt)
tt<-cbind(tt,num)#将日期和这列数字 1 合并成一个数据框
m<-aggregate(num~date(tt$tt),FUN=sum)#计算出每月有效的进项发票数量
names(m)<-c("日期","发票数量")
ans<-sd(m$发票数量)#计算标准差
lst[[i]]<-ans
ls[[i]]<-(te[[i]]$V1)[1]#获取当前公司编号,存储当前标准差所对应的公司编号
}
ls<-data.frame(ls)
lst<-data.frame(lst)
ls<-t(ls)
lst<-t(lst)
l<-cbind(ls,lst)#合并公司编号和对应的标准差为一个数据框
```

```
write.table(l,"C:/Users/  /Desktop/ans1.csv",row.names=FALSE,col.names=TRUE,sep=",")
      #这段代码用于进行主成分分析

#  library("FactoMineR")

#  library("factoextra")

#  library(reshape2)

#  library(ggplot2)

#  dat2<-read.csv("C:/Users/  /Desktop/gg.csv",header  =  FALSE)

#  dat2<-dat2[-1,]

dat2<-as.numeric(dat2)

dat<-PCA(dat2)

dat2$V4<-dat2$V5

dat2$V6<-dat2$V6

dat2$V5<-dat2$V6

dat2[1:5]

a<-PCA(dat2)

fviz_eig(a)

fviz_pca_var(a,  col.var  =  "blue")

var<-get_pca_var(a)

eig<-get_eigenvalue(a)

corrplot(var$cos2,  is.corr=FALSE)

fviz_pca_var(a,  col.var  =  "contrib",  gradient.cols  =  c("#00AFBB",  "#E7B800",  "#FC4E07
"))

write.table(eig,"C:/Users/  /Desktop/PCA.csv",row.names=FALSE,col.names=TRUE,sep=",")
```

HD_DecisionTree.py

```
    from HD_Probe import data, plot_cnf_matirx
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report
from sklearn import metrics
import pydotplus
from sklearn import tree
from sklearn.metrics import confusion_matrix

x = data.iloc[:, :13]
y = data.iloc[:, 13]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=7)

# 基于信息熵的决策树
dt1 = DecisionTreeClassifier(criterion='entropy', random_state=0)
print(dt1.fit(x_train, y_train))
y_pred = dt1.predict(x_test)
score = dt1.score(x_test, y_test)
print(score)

# 分类报告
print(classification_report(y_test, y_pred))

# 绘制 AUC 曲线
y_score = dt1.predict_proba(x_test)[:, 1]
fpr, tpr, threshold = metrics.roc_curve(y_test, y_score)
roc_auc = metrics.auc(fpr, tpr)
plt.stackplot(fpr, tpr, color='steelblue', alpha=0.5, edgecolor='black')  # alpha 调节颜色深浅
plt.plot(fpr, tpr, color='black', lw=1)  # lw：调节线条粗细
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
plt.text(0.5, 0.3, 'ROC curve (area = %0.2f)' % roc_auc)
plt.xlabel('1-specificity')
plt.ylabel('sensitivity')
plt.show()

# 决策树可视化
dot_data = tree.export_graphviz(dt1, out_file=None)
graph = pydotplus.graph_from_dot_data(dot_data)
graph.write_png('heart_disease1.png')  # 输出 PNG 格式
```

```
# graph.write_pdf('heart_disease1.pdf')  # 输出 PDF 格式

# 混淆矩阵
cnf_matrix = confusion_matrix(y_test, y_pred)
print(cnf_matrix)
# 此处调用前面的绘制函数
plot_cnf_matirx(cnf_matrix, 'Confusion matrix -- DecisionTree')

# 基于基尼系数的决策树
dt2 = DecisionTreeClassifier(criterion='gini', random_state=0)
print(dt2.fit(x_train, y_train))
y_pred = dt2.predict(x_test)
score = dt2.score(x_test, y_test)
print(score)

# 分类报告
print(classification_report(y_test, y_pred))

# 绘制 AUC 曲线
y_score = dt2.predict_proba(x_test)[:, 1]
fpr, tpr, threshold = metrics.roc_curve(y_test, y_score)
roc_auc = metrics.auc(fpr, tpr)
plt.stackplot(fpr, tpr, color='steelblue', alpha=0.5, edgecolor='black')  # alpha 调节颜色深浅
plt.plot(fpr, tpr, color='black', lw=1)
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
plt.text(0.5, 0.3, 'ROC curve (area = %0.2f)' % roc_auc)
plt.xlabel('1-specificity')
plt.ylabel('sensitivity')
plt.show()

# 可视化
dot_data = tree.export_graphviz(dt2, out_file=None)
graph = pydotplus.graph_from_dot_data(dot_data)
graph.write_png('heart_disease2.png')
# graph.write_pdf('heart_disease2.pdf')

# 混淆矩阵
cnf_matrix = confusion_matrix(y_test, y_pred)
print(cnf_matrix)
# 此处调用前面的绘制函数
plot_cnf_matirx(cnf_matrix, 'Confusion matrix -- DecisionTree')

# 预剪枝
dt3 = DecisionTreeClassifier(criterion='gini', max_depth=6, random_state=0)  # 基于基尼指数的决策树，
```

```
设置最大层数为 6（测试为最佳层数）
print(dt3.fit(x_train, y_train))
y_pred = dt3.predict(x_test)
score = dt3.score(x_test, y_test)
print(score)

# 分类报告
print(classification_report(y_test, y_pred))

# 绘制 AUC 曲线
y_score = dt3.predict_proba(x_test)[:, 1]
fpr, tpr, threshold = metrics.roc_curve(y_test, y_score)
roc_auc = metrics.auc(fpr, tpr)
plt.stackplot(fpr, tpr, color='steelblue', alpha=0.5, edgecolor='black')  # alpha 调节颜色深浅
plt.plot(fpr, tpr, color='black', lw=1)
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
plt.text(0.5, 0.3, 'ROC curve (area = %0.2f)' % roc_auc)
plt.xlabel('1-specificity')
plt.ylabel('sensitivity')
plt.show()

# 可视化
dot_data = tree.export_graphviz(dt2, out_file=None)
graph = pydotplus.graph_from_dot_data(dot_data)
graph.write_png('heart_disease3.png')
# graph.write_pdf('heart_disease3.pdf')

# 混淆矩阵
cnf_matrix = confusion_matrix(y_test, y_pred)
print(cnf_matrix)
# 此处调用前面的绘制函数
plot_cnf_matirx(cnf_matrix, 'Confusion matrix -- DecisionTree')

# 三种方式对比
y_score1 = dt1.predict_proba(x_test)[:, 1]
y_score2 = dt2.predict_proba(x_test)[:, 1]
y_score3 = dt3.predict_proba(x_test)[:, 1]
fpr1, tpr1, threshold = metrics.roc_curve(y_test, y_score1)
fpr2, tpr2, threshold = metrics.roc_curve(y_test, y_score2)
fpr3, tpr3, threshold = metrics.roc_curve(y_test, y_score3)
roc_auc1 = metrics.auc(fpr1, tpr1)
roc_auc2 = metrics.auc(fpr2, tpr2)
roc_auc3 = metrics.auc(fpr3, tpr3)
plt.plot(fpr1, tpr1, color='black', lw=1)
```

```
plt.plot(fpr2, tpr2, color='blue', lw=1)
plt.plot(fpr3, tpr3, color='green', lw=1)
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
# plt.text(0.5,0.3,'ROC curve (area = %0.2f)'%roc_auc)
plt.legend(labels=['信息熵', '基尼指数', '剪枝基尼指数'])
plt.xlabel('1-specificity')
plt.ylabel('sensitivity')
plt.show()
```

其他代码见支撑材料