Logistic&KNN 学习笔记
——基于线性、Logistic 模型和 KNN 的个人信贷违约案例分析

黄馨莹 32920172201174
黄子睿 32920172201175

2020 年 12 月 12 日

目录

1 引言

1.1 研究背景及意义

近年个人信贷消费需求不断上升，而银行对个人信用的管理水平仍然相对落后不成熟。其中一个主要因素是完善的个人信用评估方法的缺失。个人信用评估是建立在综合考虑个人经济、社会、家庭等情况基础上，评判其对各种经济承诺的履约能力与信誉度。其实质是一个分类问题，即根据个人各方面信息综合考置将其划归为不同群类，如“予以贷款”或“拒绝贷款”，更细一点可划归到不同贷款额度对象。国内众多商业银行在个人信贷评估中仍然采取根据客户填报信息来进行经验判断，这必然导致评估结果的主观性，同时也影响决策效率，一定程度上增加了人工成本，从长远看来会引起银行不良贷款率偏高。充分有效地利用客户数据，通过科学合理的个人信用评估模型来得出客观、一致的评估结果，将使银行效率得到很大提升，缩短审核时间，提升客户体验，增强行业竞争力。

1.2 文章框架

本报告首先对 Logistic 模型和 KNN 的理论进行简单概述，其次介绍所选数据集并对数据进行预处理和数据集划分，使用线性回归、Logistic 模型和 KNN 三种方法对训练集进行实证分析，计算训练集和测试集的混淆矩阵，并画出 ROC 曲线，计算 AUC。最后对三种方法进行比较分析。

2 理论概述

2.1 Logistic 模型

一般线性模型用于分析服从正态分布的连续型被解释变量与各类解释变量之间的数量变化关系。但实际应用中还有相当多的问题研究解释变量如何对一个类别型变量产生影响，被解释变量可以是二分类变量，也可以是多分类变量。本文的研究主题，是否授予贷款作为被解释变量，申请者的收入、婚姻情况等个人信息作为解释变量。采用一般线性回归模型对上述问题进行研究时，会出现被解释变量的取值分布不再满足一般线性回归模型对其的分布要求。在一般的线性回归模型中解释变量的分布与取值均没有限制，因此被解释变量作为解释变量的各种线性组合其取值连续并且无范围限制。

此外，误差项也不再满足高斯-马尔科夫假定。所以，当被解释变量不是服从正态分布的连续型数值变量时，不能直接采用一般线性回归模型。为应对这一问题，对被解释变量进行一些变换处理，最终形成了 Logistic 回归分析的理论。

Ligistic 回归模型可以为银行等金融机构预测客户的信用好坏，降低信用风险。Logistic 模型的一般形式如下：

$$p_i = \frac{1}{1+e^{-(a+bx_i)}} = \frac{e^{a+bx_i}}{1+e^{a+bx_i}} \tag{1}$$

对 p_i 进对数变换可以转换成线性形式如下：

$$ln(\frac{p_i}{1-p_i}) = a + bx_i \tag{2}$$

当自变量有 m 个时，

$$p_i = \frac{e^{\beta_0+\beta_1 x_1+\beta_2 x_2+...+\beta_m x_m}}{1+e^{\beta_0+\beta_1 x_1+\beta_2 x_2+...+\beta_m x_m}} \tag{3}$$

经过对数变换后的多元 Logistic 回归模型为：

$$ln(\frac{p_i}{1-p_i}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_m x_m = \beta_0 + \sum_{j=1}^{m} \beta_j x_{ij} \tag{4}$$

其中 $p_i = P(y_i = 1|X_{1i}, X_{2i}, ..., X_{mi})$ 表示给定自变量 $X_{1i}, X_{2i}, ..., X_{mi}$ 时事件发生 $y_i = 1$ 的条件概率。

2.2 KNN

K 近邻 KNN（k-Nearest Neighbor）算法，也叫 K 最近邻算法，1968 年由 Cover 和 Hart 提出，是机器学习算法中比较成熟的算法之一。K 近邻算法使用的模型实际上对应于对特征空间的划分。KNN 算法不仅可以用于分类，还可以用于回归。

KNN 就是给定一个训练数据集，对新的输入实例，在训练数据集中找到与该实例最邻近的 K 个实例（K 个邻居），这 K 个实例的多数属于某个类，就把该输入实例分类到这个类中。如果一个样本在特征空间中的 k 个最相似（即特征空间中最邻近）的样本中的大多数属于某一个类别，则该样本也属于这个类别。K 近邻算法使用的模型实际上对应于对特征空间的划分。

KNN 没有显示的训练过程，在测试时，计算测试样本和所有训练样本的距离，根据最近的 K 个训练样本的类别，通过多数投票的方式进行预测。具体算法描述如下：

输入：训练数据集 $T = \{(x1, y1), (x2, y2), ..., (xn, yn)\}$，其中 x$i \in Rn, yi \in \{c1, c2, ..., cK\}$ 和测试数据 x

输出：实例 x 所属的类别

1) 根据给定的距离度量，在训练集 T 中找到与 x 距离最近的 k 个样本，涵盖这 k 个点的 x 的邻域记作 Nk(x)。

2）在 Nk(x) 中根据分类规则（如多数表决）确定 x 的类别 y：

$$y = argmax_j \sum_{x_i \in N_k(x)} I\{y_i = c_i\}, i = 1, 2, ..., n; j = 1, 2, ..., K \tag{5}$$

在选择 K 时，我们面临 bias 和 variance 的权衡问题。当 K=1 时，KNN 的训练错误率为 0，偏差低方差高。随着 K 的增长，该方法变得不那么灵活，会产生一个更接近线性的决策边界，具有更低的方差和更高的偏差。

3 模型实证分析

3.1 数据来源及指标介绍

本文采用的数据是国际上应用广泛的信贷数据集 German Credit Dataset，来源于 UCI®，是德国某银行的个人信贷数据。该数据集包括用户的各类个人信用指标以及银行对信贷用户的划分，共有两类：0 为差客户，1 为好客户。好客户就是信用水平较高的客户，差客户则是银行认为其违约风险较高而拒绝提供贷款的客户。

表 1: 数据指标分类

个人指标	性别，年龄，婚姻状况，现居住状况，工作状况， 应抚养人数，是否外籍员工，有无电话，房屋状况
经济指标	储蓄存款账户状况、现工作就业时间，财产状况， 分期付款额占月收入百分比的比例，在本银行已有存款数目
信用指标	经常账户状况，账户持续时间（贷款时间），贷款历史状况， 贷款用途，贷款数额，其他债务或保证金，其他分期付款计划

3.2 数据预处理

该数据集事先已经过初步处理，数据较为干净，没有缺失值，对于该数据的预处理工作主要为连续属性的离散化。经离散化后的数据如下：

表 2: 离散化后的数据

属性编号	属性含义	属性值
V1	经常账户状况	A11：账户余额小于 0 马克 A12：账户余额在 0 马克和 200 马克之间 A13:：大于 200 马克；A14：无经常账户余额
V2	账户持续时间（月）	连续型变量
V3	贷款历史状况	A30：无学分/所有信用证均已如期偿还 A31：我行所有信用证均已如期还清 A32：到期偿还的现有信用证；A33：过去延迟还款 A34：关键客户/其他现有信贷（不在本行）
V4	贷款用途	A40：汽车（新）；A41：汽车（二手）；A42：家具/设备 A43：广播/电视；A44：家用电器；A45：修理；A46：教育 A47：假期；A48：再培训;A49：商务；A410：其他
V3	贷款金额	连续型变量
V6	储蓄账户存款状况	A61：账户余额小于 100 马克 A62: ：账户余额在 100 马克和 500 马克之间 A63：账户余额在 500 马克和 1000 马克之间 A64：账户余额大于 1000 马克；A65：未知/无储蓄账户
V7	现工作就业时间	A71：失业；A72：小于 1 年
V8	分期付款率占可支配收入的百分比	-
V9	个人状况和性别	A91：男性：离婚/分居；A92：女性：离婚/分居/已婚 A93：男：单身；A94：男性：已婚/丧偶；A95：女性：单身
V10	其他债务或担保人	A101：无；A102：共同申请人；A103：担保人
V11	现居住状况	-
V12	财产状况	A121：房地产；A124：未知/无财产 A122：无房产不动产，有社保储蓄协议或养老保险 A123：无房产不动产，无社保协议，有汽车
V13	年龄	连续型变量
V14	其他分期付款计划	A141：银行；A142：商店；A143：无
V15	住房状况	A151：租金；A152：自有；A153：免费
V16	本行现有存款数量	-
V17	工作状况	A171：失业/非熟练-非居民；A172：非熟练-常驻 A173：熟练员工/官员；A174：管理层/个体经营者/高级雇员
V18	应抚养人数	1；2
V19	电话	A191：无；A192:：有，以客户的名义注册
V20	外籍工人	A201：是；A202：否
V21	客户性质	2：差客户；1：好客户

3.3 数据集划分

总数据集包括 700 个好客户和 300 个差客户，通过不放回的随机抽样，从 1000 个客户中抽取 700 个作为训练集，剩余 300 个客户作为测试集。

3.4 模型建立

3.4.1 线性回归模型

我们首先建立简单线性模型如下：

$$V_{21} = \beta_0 + \beta_1 V_1 + \beta_2 V_2 + ... + \beta_{19} V_{19} + \beta_{20} V_{20} \tag{6}$$

回归结果如下：

表 3: 线性模型回归结果

解释变量	估计值	标准误	p 值
(Intercept)	1.133e+00	1.757e-01	2.13e-10 ***
V1	-9.818e-02	1.301e-02	1.46e-13 ***
V2	5.483e-03	1.840e-03	0.002991 **
V3	-6.141e-02	1.688e-02	0.000296 ***
V4	-1.478e-02	6.073e-03	0.015163 *
V5	1.613e-06	8.708e-06	0.853133
V6	-2.725e-02	1.029e-02	0.008244 **
V7	-1.869e-02	1.404e-02	0.183736
V8	4.085e-02	1.573e-02	0.009623 **
V9	-5.889e-02	2.222e-02	0.008231 **
V10	-5.674e-02	3.225e-02	0.078906
V11	3.825e-03	1.521e-02	0.801551
V12	4.422e-02	1.715e-02	0.010134 *
V13	-1.291e-03	1.539e-03	0.401572
V14	-3.567e-02	2.207e-02	0.106475
V15	-5.714e-02	3.374e-02	0.090837
V16	4.004e-02	3.106e-02	0.197759
V17	-2.143e-02	2.719e-02	0.430892
V18	3.965e-02	4.490e-02	0.377485
V19	-4.815e-02	3.519e-02	0.171681
V20	-1.261e-01	8.354e-02	0.131715

模型的 R 方为 0.2184，调整后 R 方为 0.1954，可见直接建立线性回归模型则解释变量对被解释变量的解释程度不够高。训练集及测试集的混淆矩阵如下：

表 4: 线性回归模型训练集混淆矩阵

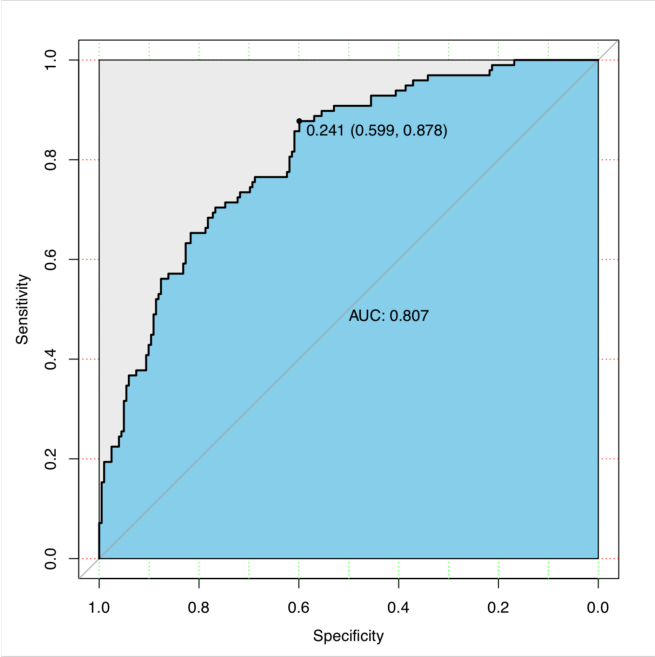
		true default	
		0	1
predicted default	No	0.90763052	0.54455446
	Yes	0.09236948	0.45544554

表 5: 线性回归模型测试集混淆矩阵

		true default	
		0	1
predicted default	No	0.93564356	0.63265306
	Yes	0.06435644	0.36734694

该模型训练集的 FNR 为 0.54455446，FPR 为 0.09236948，测试集的 FNR 为 0.63265306，FPR 为 0.06435644。画出该线性模型的 ROC 曲线如下图，AUC 为 0.807。

图 1: 线性回归模型 ROC 曲线



3.4.2 Logistic 模型

该信用数据集包含 20 个描述客户信息的特征变量 $V_i = (V_{1i}, V_{2i}, ..., V_{20i}), (i = 1, 2, ..., 1000))$, 另外包含一个描述客户是否违约 (即是好客户还是差客户) 的类别变量（即 V_{21})。当客户是差客户时 $V_{21} = 2$，客户是好客时 $V_{21} = 1$, 则客户是好客户发生的概率为 $p_i = P(V_{21i} = 1|V_i)$。于是建立 Logistic 回归模型:

$$y_i = ln(\frac{p_i}{1-p_i}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_m x_m \tag{7}$$

回归结果如下:

表 6: 线性模型回归结果

解释变量	估计值	标准误	p 值
(Intercept)	4.411e+00	1.184e+00	0.000195 ***
V1	-5.852e-01	8.521e-02	6.55e-12 ***
V2	3.026e-02	1.095e-02	0.005716 **
V3	-3.622e-01	1.039e-01	0.000494 ***
V4	-1.009e-01	3.940e-02	0.010427 *
V5	2.427e-05	5.261e-05	0.644594
V6	-1.876e-01	6.901e-02	0.006573 **
V7	-1.140e-01	8.500e-02	0.180023
V8	2.628e-01	9.937e-02	0.008182 **
V9	-3.496e-01	1.373e-01	0.010912 *
V10	-3.628e-01	2.133e-01	0.088961
V11	2.743e-03	9.290e-02	0.976445
V12	2.529e-01	1.084e-01	0.019687 *
V13	-9.389e-03	9.501e-03	0.323061
V14	-2.324e-01	1.297e-01	0.073180
V15	-3.295e-01	2.028e-01	0.104126
V16	2.401e-01	1.946e-01	0.217246
V17	-1.227e-01	1.629e-01	0.451113
V18	2.092e-01	2.826e-01	0.459167
V19	-2.884e-01	2.204e-01	0.190765
V20	-1.069e+00	6.758e-01	0.113800

在该案例中，FNR 为把差客户误分为好客户的概率，FPR 为把好客户误分为差客户的概率。对于银行来说，误把钱借给没有偿还能力的顾客所遭受的损失比误拒借钱给好客户所受损失要大，即 FNR 更重要。在二值分类中，如果使用 0.5 作为 threshold，则 $p(y=1|x) > 0.5$ 时 $\hat{y} = 1$。我们可以通过更改 $\hat{y} = 1$ 的 threshold 来最小化我们希望的错误类型，在本例中即为

最小化 FNR。

首先我们令 threshold=0.5：

表 7: Logistic 模型训练集混淆矩阵（threshold=0.5）

		true default	
		0	1
predicted default	No	0.8995984	0.5099010
	Yes	0.1004016	0.4900990

表 8: Logistic 模型测试集混淆矩阵（threshold=0.5）

		true default	
		0	1
predicted default	No	0.96039604	0.75510204
	Yes	0.03960396	0.24489796

我们尝试降低 threshold 为 0.1，新的混淆矩阵如下：

表 9: Logistic 模型训练集混淆矩阵（threshold=0.1）

		true default	
		0	1
predicted default	No	0.30321285	0.05445545
	Yes	0.69678715	0.94554455

表 10: Logistic 模型测试集混淆矩阵（threshold=0.1）

		true default	
		0	1
predicted default	No	0.93069307	0.64285714
	Yes	0.06930693	0.35714286

通过与上表对比，降低 threshold 的 FNR 也随之降低，说明对 FNR 的惩罚增大，同时 FPR 有所增大，但是增幅很小，因此该处理是有效的。继续降低 threshold 为 0.01，新的混淆矩阵如下：

表 11: Logistic 模型训练集混淆矩阵（threshold=0.01）

		true default	
		0	1
predicted default	No	0.00401606	0.00000000
	Yes	0.99598397	1.00000000

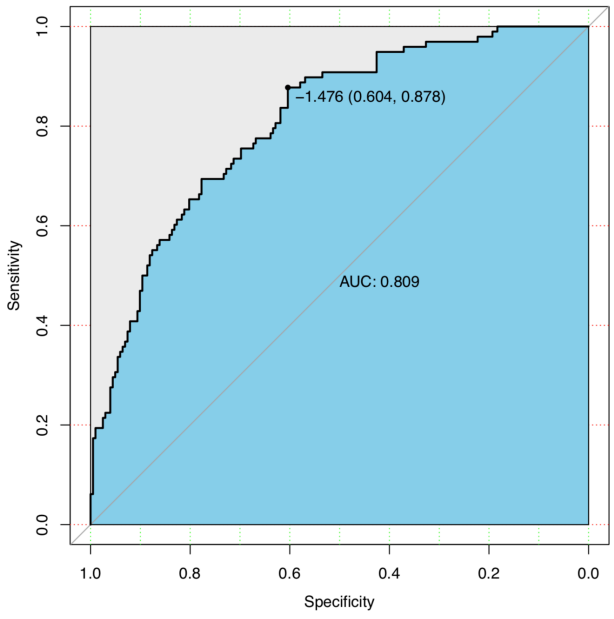
表 12: Logistic 模型测试集混淆矩阵（threshold=0.01）

		true default	
		0	1
predicted default	No	0.92079208	0.61224490
	Yes	0.07920792	0.38775510

此时惩罚力度非常大，使得训练集的 FNR 为 0，但 FPR 过大。但是测试集的表现仍较好。

画出该 logistic 模型的 ROC 曲线如下图，AUC 为 0.809。

图 2: Logistic 模型 ROC 曲线



3.4.3 KNN 模型

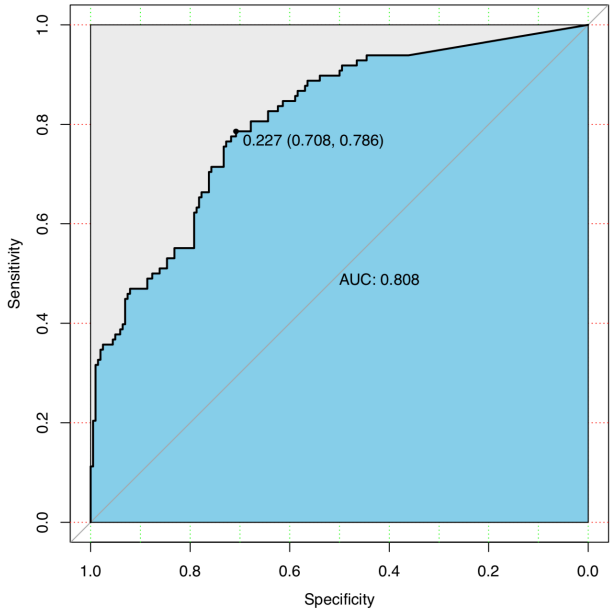
KNN 没有显示的训练过程，拟合得到的混淆矩阵如下：

表 13: KNN 混淆矩阵

		true default	
		0	1
predicted default	No	0.92079208	0.61224490
	Yes	0.07920792	0.38775510

画出 ROC 曲线如下，AUC 为 0.808。

图 3: KNN 模型 ROC 曲线



4 模型比较与总结

常见的参数模型——线性模型、Logistic 模型、Probit 模型均可用于二分类问题。而在二分类问题中，Probit 与 Logistic 相同，因此无需比较。而二分类变量不符合线性模型的假设，因此参数模型中应用 Logistic。在非参

数模型中，我们选择了 KNN 来与其进行比较。KNN 是一种非参数（无模型）的方法，一般来说这些方法在各种情况下都能很好地进行预测，因为它们不做任何实际的假设。缺点是它们本质上是一个黑盒子，缺乏可解释性。它们的计算成本也更高，因为它们通常需要存储整个数据，并在对一个新点进行预测时使用它们。而参数化方法则用一组固定的参数对数据进行总结，这些参数足以进行预测。此外，KNN 还存在一个维数问题：给定 N，当 p 即输入空间的维数逐渐变大时，数据变得相对稀疏。在高维中，K 个最近点表示的邻域可能不是局部的。上述三个模型相比，Logistic 模型的 AUC 最大，对于数据的分类效果最好。

5 附录：代码

```
data0=read.csv("F:/R/Gernum.csv")
dim(data0)
View(data0)
data0=data0[,-22]
names(data0)[1]="V1"
data0[data0$V21==1,]$V21=0
#data0$V21
data0[data0$V21==2,]$V21=1
#data0$V21
#library(infotheo)
#library(dplyr)
set.seed(1234)
c=sample(1:nrow(data0),nrow(data0)*0.3)
test=data0[c,]
View(test)
train=data0[-c,]
dim(data0[-c,])
View(train)
rownames(train)=NULL
rownames(test)=NULL
#train0$V1
```

```
#View(test)
##linear
linfit=lm(train$V21~.,data=train)
summary(linfit)
#accuracy
cutoff=.5
lin_yred=as.factor(linfit$fit>cutoff)
levels(lin_yred)=c("No","Yes")
t_lin1=table(lin_yred,train$V21,dnn=c("predicted default","true default"))
prop.table(t_lin1,2)
lin.pred=predict(linfit,newdata=test)
lin_pred=as.factor(lin.pred>cutoff)
levels(lin_pred)=c("No","Yes")
t_lin2=table(lin_pred,test$V21,dnn=c("predicted default","true default"))
prop.table(t_lin2,2)
#acc_lintrain=(t_lin1[1,1]+t_lin1[2,2])/sum(t_lin1)
#acc_lintest=(t_lin2[1,1]+t_lin2[2,2])/sum(t_lin2)
#print(c(c("Accuracy of tree train: ", acc_lintrain),c("Accuracy of tree test:", acc_lintest)))
#error=(t_lin2[1,2]+t_lin2[2,1])/sum(t_lin2)
#error
#ROC
library("pROC")
roclin=roc(as.ordered(test$V21) ,as.ordered(lin.pred))
plot(roclin,print.auc=T, auc.polygon=T, grid=c(0.1, 0.2), grid.col=c("green","red"), max.auc.polygon=T, auc.polygon.col="skyblue",print.thres=T)
##logistic
logfit=glm(train$V21~.,family=binomial(link = "logit"),data=train)
summary(logfit)
###cost matrix
cutoff=.5
logit.p=logfit$fit
```

```
logit.y=as.factor(logit.p>cutoff)
levels(logit.y)=c("No","Yes")
t=table(logit.y,train$V21,dnn=c("predicted default","true default"))
t
prop.table(t,2)
log.pred=predict(logfit,newdata=test)
log_pred=as.factor(log.pred>cutoff)
levels(log_pred)=c("No","Yes")
t_log2=table(log_pred,test$V21,dnn=c("predicted default","true default"))
prop.table(t_log2,2)
#ROC
roclog=roc(as.ordered(test$V21) ,as.ordered(log.pred))
plot(roclog,print.auc=T, auc.polygon=T, grid=c(0.1, 0.2), grid.col=c("green","red"),
max.auc.polygon=T, auc.polygon.col="skyblue",print.thres=T)
#change cutoff
cutoff=.1
logit.p=logfit$fit
logit.y=as.factor(logit.p>cutoff)
levels(logit.y)=c("No","Yes")
t=table(logit.y,train$V21,dnn=c("predicted default","true default"))
t
prop.table(t,2)
log.pred=predict(logfit,newdata=test)
log_pred=as.factor(log.pred>cutoff)
levels(log_pred)=c("No","Yes")
t_log2=table(log_pred,test$V21,dnn=c("predicted default","true default"))
prop.table(t_log2,2)
#change cutoff
cutoff=.01
logit.p=logfit$fit
logit.y=as.factor(logit.p>cutoff)
levels(logit.y)=c("No","Yes")
t=table(logit.y,train$V21,dnn=c("predicted default","true default"))
```

```
t
prop.table(t,2)
log.pred=predict(logfit,newdata=test)
log_pred=as.factor(log.pred>cutoff)
levels(log_pred)=c("No","Yes")
t_log2=table(log_pred,test$V21,dnn=c("predicted default","true default"))
prop.table(t_log2,2)
#knn
library("kknn")
knnm=kknn(train$V21~.,train=train,test=test,k=7,distance=2, kernel="gaussian")
# 自变量中含有类别变量用 kknn
#kfit=fitted(knnm)
summary(knnm)
k.pred=predict(knnm,newdata=test)
k_pred=as.factor(k.pred>0.5)
levels(k_pred)=c("No","Yes")
t_k2=table(k_pred,test$V21,dnn=c("predicted default","true default"))
prop.table(t_k2,2)
#CTt=table(test$V21,knnm$fitted.values)
#CTt
#acc_ktest=(CTt[1,1]+CTt[2,2])/sum(CTt)
#acc_ktest
#ROC
rock=roc(as.ordered(test$V21) ,as.ordered(knnm$fitted.values))
plot(rock,print.auc=T, auc.polygon=T, grid=c(0.1, 0.2), grid.col=c("green","red"),
max.auc.polygon=T, auc.polygon.col="skyblue",print.thres=T)
```

参考文献

[1] German Credit Dataset[EB/OL].
https://archive.ics.uci.edu/ml/datasets/Stalog+(German+Gredit+Data)

[1] 李泉. 随机森林在个人信用评估中的应用研究 [D]. 江西财经大学,2016.

[2] 陶艳丽. 随机森林改进模型对个人信贷违约预测的研究 [D]. 河北经贸大学,2020.