基于 Lasso 回归模型与 BP 神经网络模型的信用评分体系建立与等级划分

摘　要

随着大数据技术的不断发展，**信用风险评价**在金融行业和个体借贷过程中起到了愈发重要的作用。本文主要对信用风险评分体系和信用等级的划分展开了研究，综合建立了 Lasso 回归模型、二分类概率单位回归模型和 BP 神经网络模型，并结合中华人民共和国金融行业标准 (JR/T0030.2-2006)《信贷市场和银行间债券市场信用评级规范》进行了问题的求解。

对于问题一，主要解决了对于德国信用数据集的指标筛选以及信用风险评价问题。本文首先进行了 **KMO 检验和 Bartlett 球形度检验**，得到了低于0.6的KMO 值，且考虑到数据并不存在缺失值与异常值，因此，我们不考虑采用主成分分析法，而是建立了**Lasso 回归模型**和**二分类概率单位回归模型**，得到了11 条**标准化系数**较大的指标，剔除了13条标准化系数为 0 的指标。我们根据11条标准化系数较大的指标建立起了信用风险评价模型。

对于问题二， 主要解决了对于德国信用数据集中的个人信用评分问题。本文首先根据问题一的结果剔除了冗余的指标，并对剩余的指标进行了**归一化处理**。然后我们建立了 **BP 神经网络模型，**利用 SPSS 软件进行分析后，我们得到了结果的**预测概率**，根据预测概率我们使用百分制，对个体信用进行评分，例如第一条样本预测不违约结果概率为 0.4044，则该个体的信用得分为 40.4 分。

对于问题三， 主要解决了自建模型和其他分类模型的**性能比较**问题。本文首先分别对德国和澳大利亚的信用数据集进行了 BP 神经网络模型的建立，再利用 SPSS 软件和 MATLAB 代码将自建模型和随机森林、决策树、支持向量机以及 KNN 的结果进行了比较分析。以澳大利亚信用数据集为例，**自建 BP 神经网络模型**的准确率达到了0.899，**决策树模型**准确率为 0.83，**KNN 模型**准确率为0.853，**支持向量机模型**准确率为0.851，**随机森林模型**准确率为0.841，综合其它指标可以得到自建BP神经网络模型性能优于其它分类模型。

对于问题四，主要解决了德国信用数据集中信用等级划分的问题。本文根据中华人民共和国金融行业标准 (JR/T0030.2-2006)《**信贷市场和银行间债券市场信用评级规范**》对数据进行了计算，得到了各个等级对应的 WOE_e 值，依据“**信用等级越高，信用风险越低**”的划分标准，将信用等级划分为九种，分别为 AAA 、AA 、A 、BBB 、BB 、B、CCC 、CC 以及 C 等级。例如，AA 等级的得分区间为 [905.73-987.59], 对应的人数为 26 人。

本文的最后对所建立的模型进行了讨论和分析，对模型的优缺点进行了综合的评价。

关键词：Lasso 回归模型、二分类概率单位回归模型、BP 神经网络模型、《信贷市场和银行间债券市场信用评级规范》

一、 问题重述

1.1 问题背景

信用风险识别是金融行业以及个体借贷过程中至关重要的一个部分。通过分析借款方的个人和财务信息，评价其是否能够和愿意偿还贷款，进而降低贷款机构风险暴露的可能性。

随着大数据技术的快速发展，信用风险识别获得了更多数据来源和分析手段，但是在信用风险识别的研究中也存在着较多的问题，例如：在对于指标的筛选中，对于影响程度最大的指标选定以及克服多重共线性和过度拟合等问题是几大难点；在实际情况下，往往会出现违约评价少、非违约评价多的问题，进而导致对违约样本识别不足的问题。同时，平衡模型的预测准确性和可解释性也是一大难点；在信用等级划分中，在确保等级划分的鲁棒性和普适性的情况下，对于选择恰当的阈值、聚类模型以及非线性规划模型将信用得分映射到信用等级，求解信用等级划分结果又是一大难点。

1.2 问题要求

题目给出了UCI公开的德国信用数据集以及澳大利亚信用数据集，其中包含了个体的个人以及财务等信息指标，现要求我们通过建立数学模型解决如下问题：

问题一：选择合适的模型，减轻高维数据对于信用风险评价带来的评价指标反应信息冗余的问题，提升信用风险识别准确性和可解释性。

问题二：选择合适的信用评分模型，克服传统线性加权方法无法准确刻画指标与风险之间非线性关系的问题以及减少样本分布不均匀带来的模型识别过度和不足的问题。

问题三：利用附件1和附件2中的数据，自建信用评分模型，并将所建模型和决策树、K 最近邻、随机森林以及支持向量机等多种现有分类模型进行对比分析，进而判断 该信用评分模型的合理性和准确性，并填写相关表格。

问题四：以“信用等级越高、信用风险越低”为信用等级划分标准，构建非线性规划模型，在德国信用数据集上划分个体信用等级。

二、 问题分析

2.1 问题一的分析

对于问题一，题目中附件1给出了德国信用数据集的相关信息，我们注意到有 X1-X24 共 24 个指标，由于高维数据易带来评价指标反应信息冗余的情况，我们考虑对这 24 个指标进行降维。首先，我们先进行 KMO 检验和 Bartlett 球形度检验，通常认为在 KMO 值大于 0.6 时适宜使用主成分分析法。接着，我们采用 Lasso 回归模型，对24个指标进行筛选，得到不同的权重系数，从而剔除去相关性较弱的一些指标，达到信用风险评价的目标。

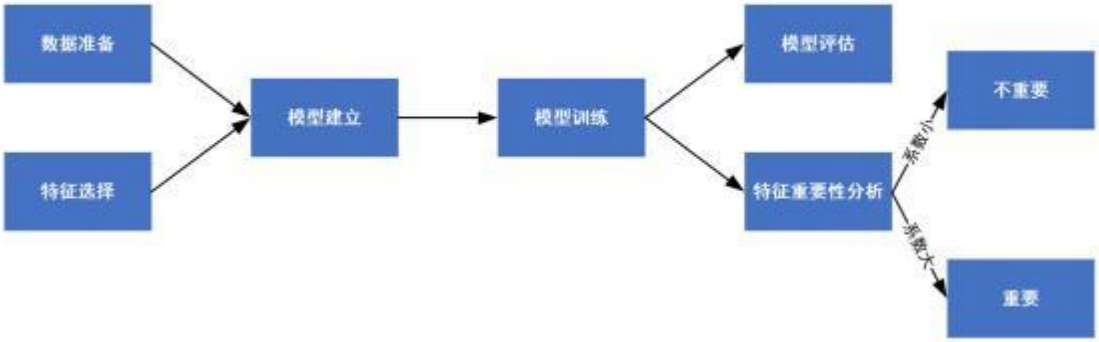

图 2-1: Lasso 回归模型求解流程图

2.2 问题二的分析

对于问题二，我们注意到本题要求我们建立一个个体信用评分模型来计算德国信用数据集中的个人信用得分。因此，我们首先应进行数据的预处理，去除部分冗余指标后再对剩余指标进行归一化处理。接着我们使用 BP 神经网络模型，通过多个神经元和层次结构以及反向传播算法得到信用评分模型，对数据进行加权处理后得到较为简明易懂的分数值，达到信用评分的目的。

2.3 问题三的分析

对于问题三，我们需要将自行建立的信用评分模型和给出的随机森林（RF）、决策树（DT）、K 最近邻（KNN）以及支持向量机（SVM）等分类模型进行比较。首先我们先利用 SPSS 软件对自建模型进行分析，求出 Accuracy、AUC、Type1-error 和 Type2-error 等参数，再调整为其他模型进行分析，通过各项参数的对比进行模型性能的对比。

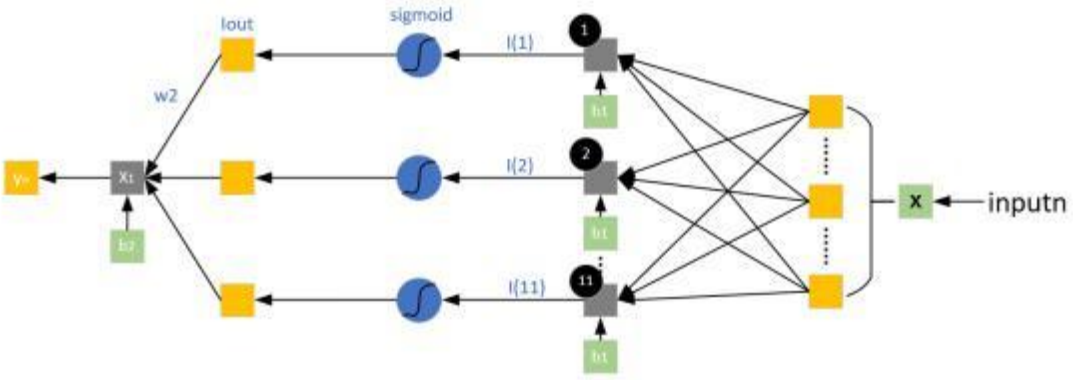

图 2-2: BP 神经网络原理结构图

2.4 问题四的分析

对于问题四，基于中华人民共和国金融行业标准 (JR/T0030.2-2006) 《信贷市场和银行间债券市场信用评级规范》，将德国信用数据集划分为 9 个等级包括 AAA、AA、A、BBB、BB、B、CCC、CC 以及 C 级，进行多目标非线性规划对信用等级进行划分。DIS 为违约与非违约客户平均信用得分的离散距离和，各信用等级对应的证据权重 (weight of evidence,WOE) 反映违约与非违约用户的个数关系，以最大化 $WOE_9 - WOE_1$ 以及DIS 距离为目标函数进行双目标函数非线性规划，得到不同等级信用得分的划分区间，达到“信用等级越高、信用风险越低”的划分目的。

三、 模型假设

1. 模型假设一：假设信用风险识别指标均为极大型指标；
2. 模型假设二：观察到 X1-X24 的指标类型可能有两类，因此我们假设 X16-X24 的 指标和 X1-X15 的指标类型一致。

四、 名词解释与符号说明

4.1 名词解释与说明

1. **鲁棒性：**模型或算法对异常值、噪声或不完全符合假设的数据的稳健性和可靠性。
2. **阈值：**指一个界限值或临界点，用于决定模型预测结果的分类或决策。
3. **信用风险评价：**信用风险评价是评估个人、公司或其他实体未来违约可能性的过程。
4. **交叉验证：**一种评估机器学习模型泛化能力的统计技术，综合评估模型在不同数 据子集上的表现。
5. **噪声：**数据集中的不希望的随机或非随机波动。

4.2 主要符号与说明

序号	符号	符号说明
1	$x_{i,j}$	X_i 的各项指标
2	Y_i	第 i 人违约与否
3	n	x_i 指标总个数
4	α_i	拟合系数
5	ϵ_i	无法观测且满足一定限制的扰动项
6	F_j	隐含层输出量
7	O_k	输出层输出变量
8	G	bp 神经网络激励系数
9	E	单次 bp 神经网络的训练误差
10	LOSS(ii)	第 ii 次训练的训练集的误差和
11	inputn	用于训练 bp 神经网络的输入训练集
12	PD_j	第 j 人的违约概率
13	WOE_e	e 信用等级对应的证据权重值

五、 模型的建立与求解

5.1 问题一模型的建立和求解

5.1.1 数据预处理

首先，对附件一的数据进行分析处理。附件一给出的是UCI公开的德国信用数据集，其中编号 X1-X24 表示个体的个人及财务等信息指标，间接反应个人信用风险。通过对数据的观察，我们发现数据没有缺失值和明显的异常值，可以认为所有数据是可分析的。

但是高维数据往往会为信用风险评价带来评价指标反应信息冗余等问题，我们发现许多指标之间具有很强的相关性，我们可以采用基于特征重要性的方法，如 Lasso 回归与二分类概率单位回归，对原始数据进行处理，把不重要的特征的系数进行压缩，从而实现特征选择。

5.1.2 Lasso 回归模型的建立与求解

对于这种高维多指标评价问题，一般会出现两个主要挑战：一是特征之间存在多重共线性，可能会使模型参数不稳定；二是有一些指标实际上对结果的贡献率不大，而使用传统的方法可能会出现过拟合的现象，干扰到模型的预测结果与模型本身的性能。我们要考虑是否可用主成分分析法，首先我们进行 KMO 检验和 Bartlett 的检验，检验结果如下：

表 5-1: KMO 检验和 Bartlett 的检验

KOM 值		0.516
Bartlett 球形度检验	近似卡方	4884.588
	df	276
	P	0.000***

对于 KMO 值：0.8 以上非常合适做主成分分析，0.7-0.8 之间为一般适合，0.6-0.7 之间不太适合，0.5-0.6 之间表示差，0.5以下表示极不适合，根据表格 KMO 检验的结果显 示，KMO 的值为 0.516，主成分分析程度为不适合。因此我们建立以下模型：

Lasso 回归（Least Absolute Selection and Shrinkage Operator）是一种替代最小二乘法的压缩估计方法。在 Lasso 回归中，模型的目标函数被修改为最小化残差平方和加上一个与模型参数绝对值相关的正则化项，这个正则化项通常是一个常数乘以所有模型参数绝对值的总和。假设数据

$$Xi(i \in \{1, 2, \ldots, 24\}), Yi$$

分别是第 i 个个体的个人及财务等信息指标及其违约与否情况。考虑线性回归模型：

$$Y_i = \alpha_i + \sum_{j=1}^{\infty} \beta_j x_{ij} + \varepsilon_i, \quad \varepsilon_i \sim N\left(0, \sigma^2\right)$$

在通常的回归结构中, 通过修改目标函数来引入正则化项, 假设观测值彼此独立, 或者 Yi 在观测值给定的情况下独立, 即 Yi 关于 Xi 条件独立, 同时假设 xij 是标准化的, 也就是

$$\frac{1}{n}\sum_j x_{ij} = 0, \frac{1}{n}\sum_j x_{ij}^2 = 1 \tag{1}$$

Lasso 回归的目标函数为：

$$\begin{aligned}(\hat{\alpha}, \hat{\beta}) = \arg\min_{\beta} &\left\{ \sum_i \left(Y_i - \alpha_i - \sum_j \beta_j x_{ij} \right)^2 + \lambda \sum_j |\beta_j| \right\} \\ \text{s. t. } &\sum_j |\beta_j| \le t\end{aligned} \tag{2}$$

上式中，β_j 是第 j 个特征的系数，λ 是正则化参数，代表控制惩罚的强度。t 是一个正的调和参数, 用于控制正则化的强度。是当 t 增加时, 正则化项的权重增加, 使得回归系数总体变小, 若令

$$t^0 = \sum_j |\beta_j|, t \le t^0 \tag{3}$$

就会使一些回归系数缩小并趋于 0, 但是任一系数都不为 0, 回归系数的减少可以简化模型并减少过拟合，并且最终保留了所有的变量。

首先通过可视化的方式给出交叉验证选择 λ 值的情况：

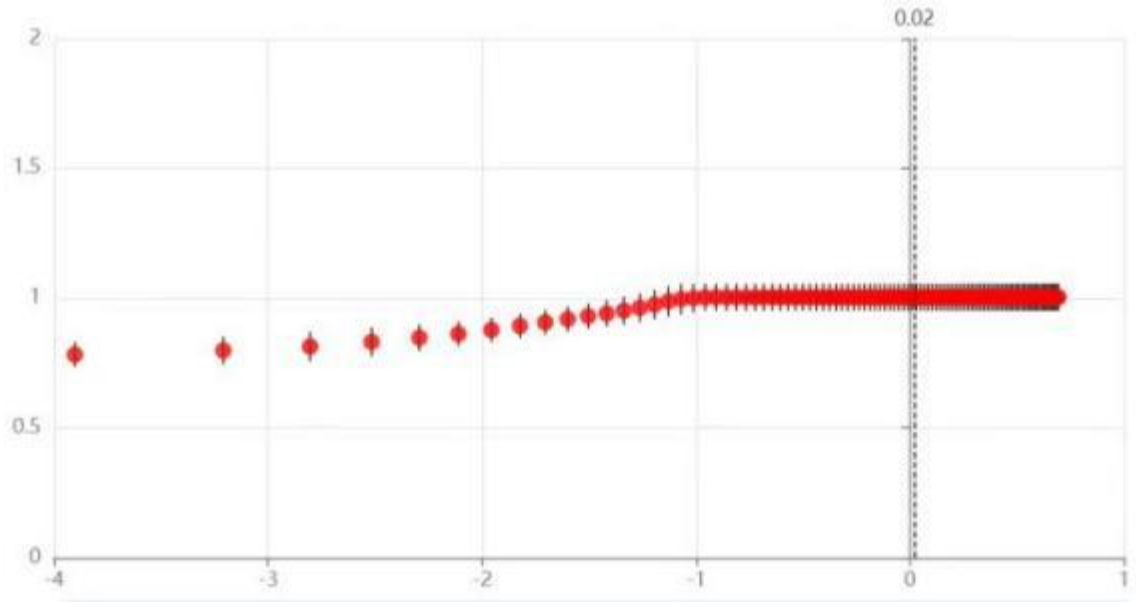

图 5-3: Lasso 回归交叉验证图

由于 Lasso 回归无显式解，在此采用最小角估计法进行近似求解，然后利用 SPSS 软件画出 λ 与模型回归系数图以确定主要特征，来达到降维的目的：

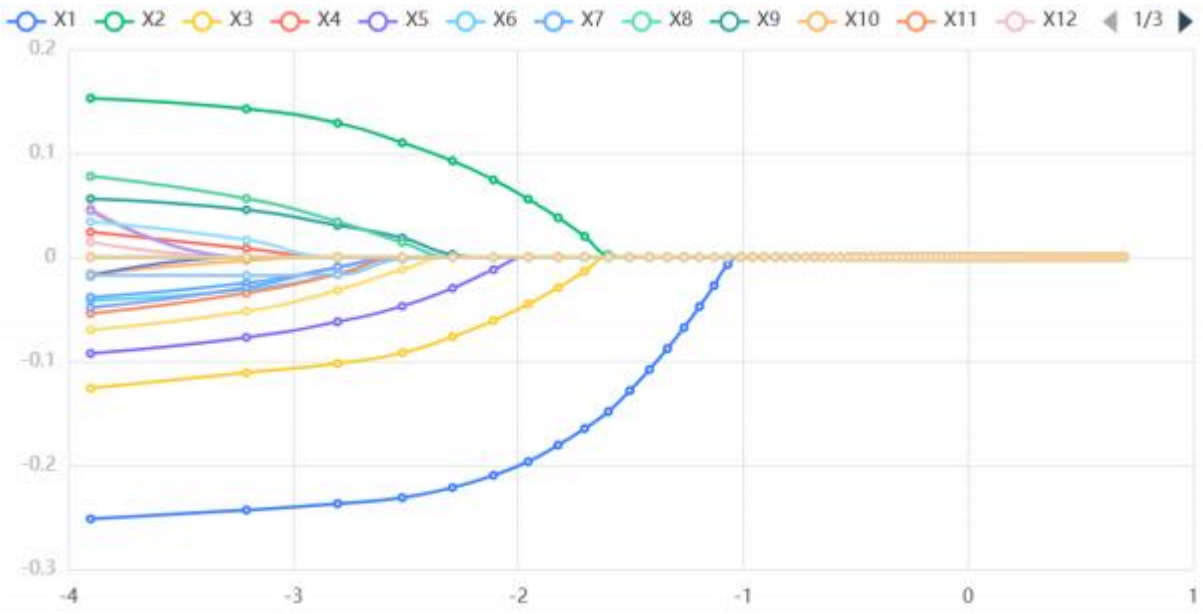

图 5-4: λ 与模型回归系数图

上图展示了随着 λ 的对数值变化，模型系数变化的情况。其中 λ 为 Lasso 回归中的惩罚系数，它控制着正则化的程度。随着横坐标参数的增大，有些指标的回归系数会逐渐向零收缩，其中指标越不重要，自变量系数会被越早压缩为零，从而实现特征筛选和模型精简的目的。所以，我们可以通过观察曲线与横轴的交点位置来判断哪些指标的系数最后可能不为零，即哪些指标可以被选中。

下面给出模型系数情况：

表 5-2: 模型系数表

变量名	标准化系数	非标准化系数	变量名	标准化系数	非标准化系数
截距	0.659	0.654	X13	0	0
X1	-0.093	-0.087	X14	0	0
X2	0.006	0.005	X15	0	-0.029
X3	-0.046	-0.044	X16	0.002	0.04
X4	0.001	0	X17	0	-0.052
X5	-0.027	-0.019	X18	0	0
X6	-0.01	-0.007	X19	0	0
X7	-0.005	-0.008	X20	0	0
X8	0	0	X21	0	-0.019
X9	0.015	0.014	X22	0	0
X10	-0.002	0	X23	0	0
X11	-0.008	-0.012	X24	0	0
X12	0	0			

在 Lasso 回归模型的系数表中，我们可以观察到一系列自变量（X1至 X24）与因变量之间的线性关系。 Lasso 回归作为一种回归分析方法，旨在通过施加 L1 正则化（即绝对值的和最小化）来减少模型的复杂度，并通过系数压缩来避免过拟合问题。

从标准化系数来看，截距项为0.659，表明在自变量全为 0 时，因变量的预测值为 0.659 个标准单位。然而，标准化系数更侧重于展示自变量与因变量之间的相对重要性，且已消除量纲影响。

对于自变量 X1 至 X24，大部分变量的标准化系数较小（绝对值接近 0），表明这些变量对因变量的直接影响较弱。具体来说， X1、X3、X5、X6、X7、X11 和 X15 的标准化系数均为负值，意味着这些变量的增加可能与因变量的减少有关；而 X2 、X9 和 X16 的标准化系数为正值，暗示这些变量的增加可能与因变量的增加相关。然而，由于这些系数的绝对值均较小，这些关系的实际影响可能并不显著。

值得注意的是，X4、X8、X10、X12、X13、X14、X17、X18、X19、X20、X22、X23 和 X24 的标准化系数均为 0，表明这些变量在 Lasso 回归模型中被完全压缩， 即它们对因变量的影响在模型中被视为 0。这可能是由于这些变量与因变量之间不存在线性关系，或者在数据集中这些变量的变异度较小。

此外，非标准化系数提供了变量在原始尺度上的影响大小。然而，由于各变量的量纲可能不同，非标准化系数的直接比较可能不太准确。在此表中，一些非标准化系数（如 X15 和 X17）与标准化系数存在差异，这可能是由于数据标准化过程中均值和标准差的影响。

利用 Lasso 回归模型求解后，我们共得到了 11 条标准化系数较大的指标， 而其它的 13 条数据标准化系数均为 0，因此我们选取标准化系数相对较大的 11 条指标来建立信用风险评价模型。

下面给出问题一的模型结果图：

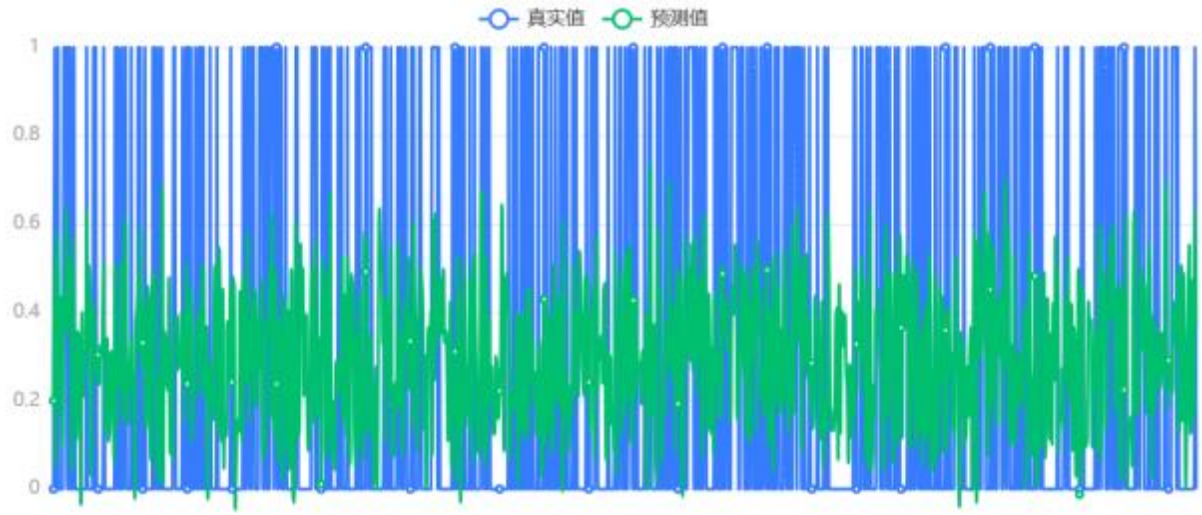

图 5-5: 问题一模型结果图

5.2 问题二模型的建立和求解

5.2.1 数据预处理

对于问题二，我们首先根据问题一的结果去除了冗余的指标，并且对剩余指标进行归一化处理（下表展示归一化结果），从而使信用评分体系准确化。

0	X2	X3	X4
0	0.029411765	1	0.054945055
0.333333333	0.647058824	0.5	0.318681319
1	0.117647059	1	0.104395604
0	0.558823529	0.5	0.423076923
0	0.294117647	0.75	0.258241758
1	0.470588235	0.5	0.489010989
1	0.294117647	0.5	0.142857143
0.333333333	0.470588235	0.5	0.368131868
1	0.117647059	0.5	0.159340659
0.333333333	0.382352941	1	0.274725275

表 5-3: 数据归一化处理 (部分)

5.2.2 BP 神经网络模型的建立与求解

- BP 神经网络是一种多层的前馈神经网络，其主要的特点是：信号是前向传播的，而误差是反向传播的。对于如下的只含一个隐含层的神经网络模型：

BP 神经网络的过程主要分为两个阶段，第一阶段是信号的前向传播，算出预测误 差 E，第二阶段是误差的反向传播，FI(j) 是隐含层j 激活函数的导数， 用于计算出 w_j 的调整值 dw_j 。系统的学习速率设为 β, 使用以下公式更新 **w**：

$$\mathbf{w} = \mathbf{w} + \mathbf{dw} + \beta * \mathbf{dw}^T \tag{4}$$

BP 神经网络的学习规则是使用最速下降法，以一个三层 BP 神经网络举例：

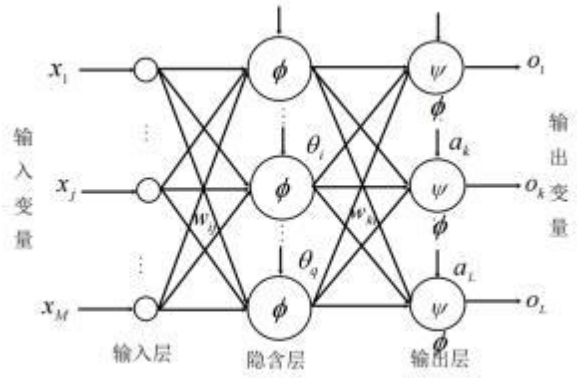

图 3.6 BP 网络结构

图 5-6: BP 神经网络原理图

隐含层的输出量设为 F_j，输出层的输出变量设为 O_k，系统的激励函数设为 G，则其三个层之间有如下数学关系：

$$\begin{cases} F_j = G\left(\sum_{i=1}^{m} \omega_{ij} x_i + a_j\right) \\ O_k = \sum_{j=1}^{l} sigmiod(F_j)\omega_{jk} + b_k \end{cases} \tag{5}$$

其中，

$$sigmoid(F_j) = c\frac{1}{1+e^{-F_j}} \tag{6}$$

系统期望的输出量设为 T_k，则系统的误差 E 可由实际输出值和期望目标值的方差表示，具体关系表达式如下：

$$E = \frac{1}{2}\sum_{k=1}^{n}(T_k - O_k)^2 \tag{7}$$

并令，利用梯度下降原理，则系统权值和偏置的更新公式如下：

$$\begin{cases} \omega_{ij} = \omega_{ij} + \beta F_j\left(1-F_j\right) x_i \sum_{k=1}^{n} \omega_{jk} e_k \\ \omega_{jk} = \omega_{jk} + \beta F_j e_k \end{cases} \tag{8}$$

$$\begin{cases} a_j = a_j + \beta F_j\left(1-F_j\right) x_i \sum_{k=1}^{n} \omega_{jk} e_k \\ b_k = b_k + \beta e_k \end{cases} \tag{9}$$

在 BP 神经网络分类的混淆矩阵热力图测试结果中，呈现了 BP 神经网络模型的分类表现。首先，观察矩阵结构，我们可以看到三个类别的预测结果分布，其中主对角线元素表示模型正确分类的情况，副对角线元素则体现了分类错误的情况。第一行第一列的元素“1.0”表示模型正确地将所有真实标签为第一类（假设为“A”类）的样本识别为该类，即真正例（True Positive, TP）为 1，假负例（False Negative, FN）为 0。这反映了模型在“A”类上的高识别准确率。

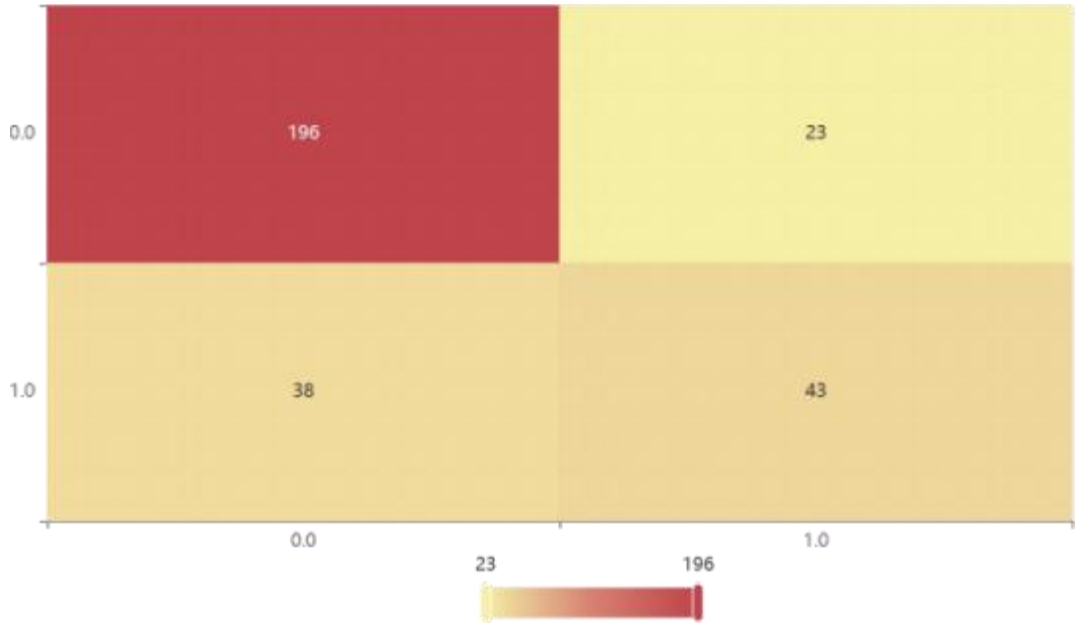

图 5-7: 热力图

第二行揭示了模型在第二类（假设为“B”类）上的分类表现。其中，第二行第二列的元素“196”表示模型正确地将 196 个真实标签为“B”类的样本识别为该类， 即 TP 为 196。然而，第二行第一列的元素“23”表明有 23 个真实标签为“A”类的样本被错误地分类为“B”类，即假正例（False Positive, FP）为 23。这显示了模型在区分“A”类和“B”类时存在一定的混淆。

第三行数据反映了模型在第三类（假设为“C”类）上的分类性能。第三行第三列的元素“43”表示模型正确地将 43 个真实标签为“C”类的样本识别为该类， 即 TP 为 43。同时，第三行第二列的元素“38”表明有 38 个真实标签为“B”类的样本被错误地分类为“C”类， 即 FP 为 38。这表明模型在“B”类和“C”类之间的区分能力相对一般，存在一些误判。我们可以进一步计算出模型的性能指标，如准确率、精确率、召回率和 F1 分数等。准确率是所有样本中正确分类的比例，精确率是预测为正样本的实例中真正为正样本的比例，召回率是实际为正样本的实例中被预测为正样本的比例， 而 F1 分数是精确率和召回率的调和平均值。

我们可以看出该 BP 神经网络分类模型在大部分样本上能够做出准确的分类，但在“A”类和“B”类之间存在一定的混淆。为了提高模型的分类性能，可以考虑进一步调整模型参数、优化特征选择或采用更复杂的网络结构等方法。同时，对于不平衡数据集的情况，还需要考虑采取相应策略来平衡各类别之间的样本数量，以提高模型在少数类样本上的分类性能。

在评估 BP 神经网络分类模型的性能时，关键指标包括准确率、召回率、精确率和

表 5-4: BP 神经网络（德国）

	准确率	召回率	精确率	F1
训练集	0.77	0.77	0.76	0.759
交叉验证集	0.753	0.753	0.742	0.737
测试集	0.797	0.797	0.787	0.79

F1 分数。这些指标提供了模型在不同数据集上表现的综合评价。首先， 观察训练集的结果，模型达到了 0.77 的准确率，同时召回率和精确率也接近该值， 显示出模型在训练数据上具有较好的分类性能。然而，高训练集性能并不一定代表模型在未见过的数据上也能有同样好的表现，因此还需要关注交叉验证集和测试集的结果在交叉验证集上，模型的准确率为 0.753，略低于训练集，但整体仍表现出较好的分类能力。召回率与准确率一致，表明模型在识别正样本方面较为稳定。然而，精确率略低于召回率，这可能意味着模型在预测为正样本的实例中，有一部分实际上是负样本， 即存在一定的误报。F1 分数综合了召回率和精确率的信息，为 0.737，进一步证实了模型在交叉验证集上的性能。在测试集上，模型表现最佳，准确率达到 0.797，高于训练集和交叉验证集。这一结果表明模型在未见过的数据上也具有较好的泛化能力。同时，召回率、精确率和 F1 分数也均有所提升，特别是 F1 分数达到 0.79，说明模型在精确性和召回性之间取得了较好的平衡。

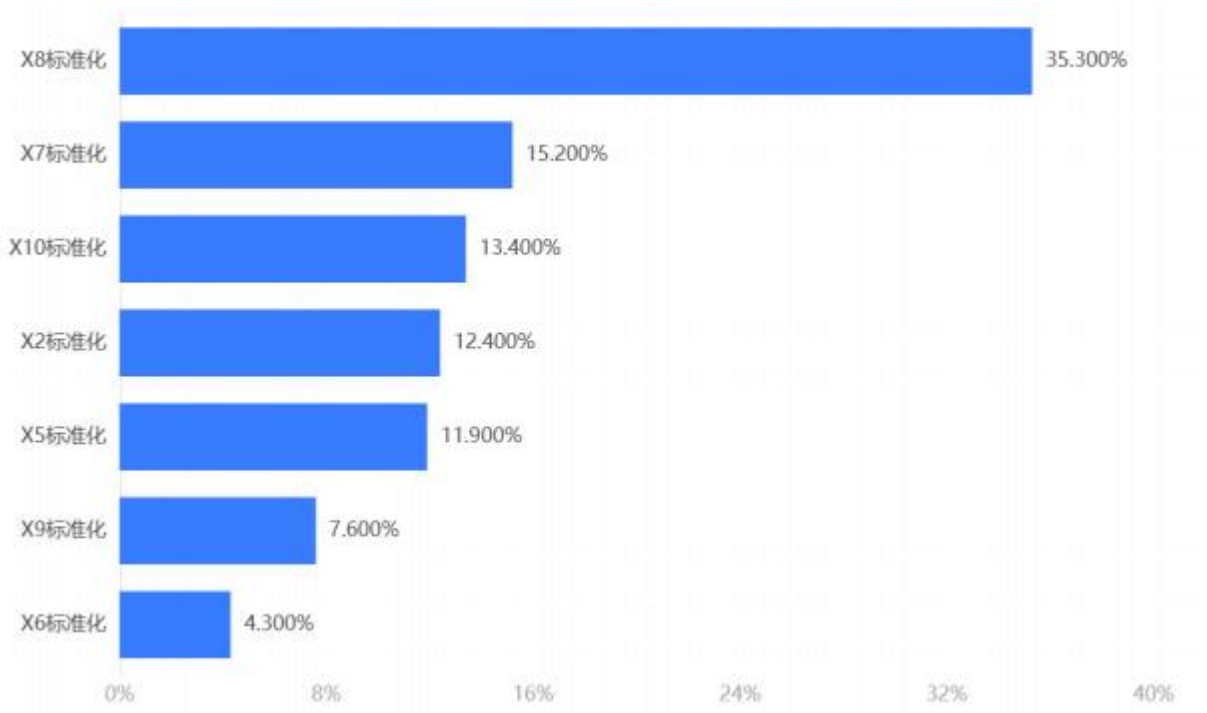

图 5-8: BP 神经网络示意图

本模型采用德国信用评估样本的 70% 作为训练集，30% 作为预测集，进行 10 次训练， LOSS(ii) 定义为第 ii 次训练的训练集的误差和。下面是对于各个主要的标准化后变量对于违约与否的预测占比比重以及预测的信用得分：

预测结果 Y	原始 Y	预测结果概率 _1	预测结果概率 _0	信用得分
1	0	0.59557245	0.40442755	40.4
1	0	0.761093687	0.238906313	23.9
0	0	0.039909007	0.960090993	96
0	0	0.035786542	0.964213458	96.4
0	0	0.197439209	0.802560791	80.3
0	0	0.14283909	0.85716091	85.7
1	0	0.582540408	0.417459592	41.7
0	0	0.059662346	0.940337654	94
0	0	0.466747079	0.533252921	53.3
1	1	0.55523787	0.44476213	44.5
1	0	0.575047076	0.424952924	42.5
1	1	0.812330157	0.187669843	18.8

表 5-5: 信用得分 (部分)

5.3 问题三模型的建立和分析

5.3.1 模型的建立和求解

• BP 神经网络（澳大利亚）：

在 BP 神经网络分类的混淆矩阵热力图[1]测试结果中，所展示的数据揭示了模型在不同类 别间的分类表现。首先，观察矩阵结构，我们可以看到三个类别的预测结果分布，其中对角线元素表示模型正确分类的情况，非对角线元素则揭示了分类错误的情况。

第一行第一列的元素“1.0”表示模型正确地将所有真实标签为第一类（假设为“A”类）的样本识别为该类， 即真正例（True Positive, TP）为 1，假负例（False Negative, FN）为 0。这反映了模型在“A”类上的高识别准确率。

第二行揭示了模型在第二类（假设为“B”类）上的分类表现。其中，第二行第二列的元素“108”表示模型正确地将 108 个真实标签为“B”类的样本识别为该类， 即 TP 为 108。然而， 第二行第一列的元素“16”表明有 16 个真实标签为“A”类的样本被错误地分类为“B”类，即假正例（False Positive, FP）为 16。这显示了模型在区分“A”类和“B”类时存在一定的混淆。

第三行数据反映了模型在第三类（假设为“C”类）上的分类性能。第三行第三列的元素“79”表示模型正确地将 79 个真实标签为“C”类的样本识别为该类， 即 TP 为 79。同时， 第三行第二列的元素“5”表明有 5 个真实标签为“B”类的样本被错误地分类为“C”类， 即 FP 为 5。这表明模型在“B”类和“C”类之间的区分能力相对较好，但仍存在少量误判。

在分析了澳大利亚的信用数据集之后我们可以进一步得到模型的性能指标，我们不难发现，在模型的训练集上，模型的准确率、召回率等指标达到了 0.85 上下，可见模型

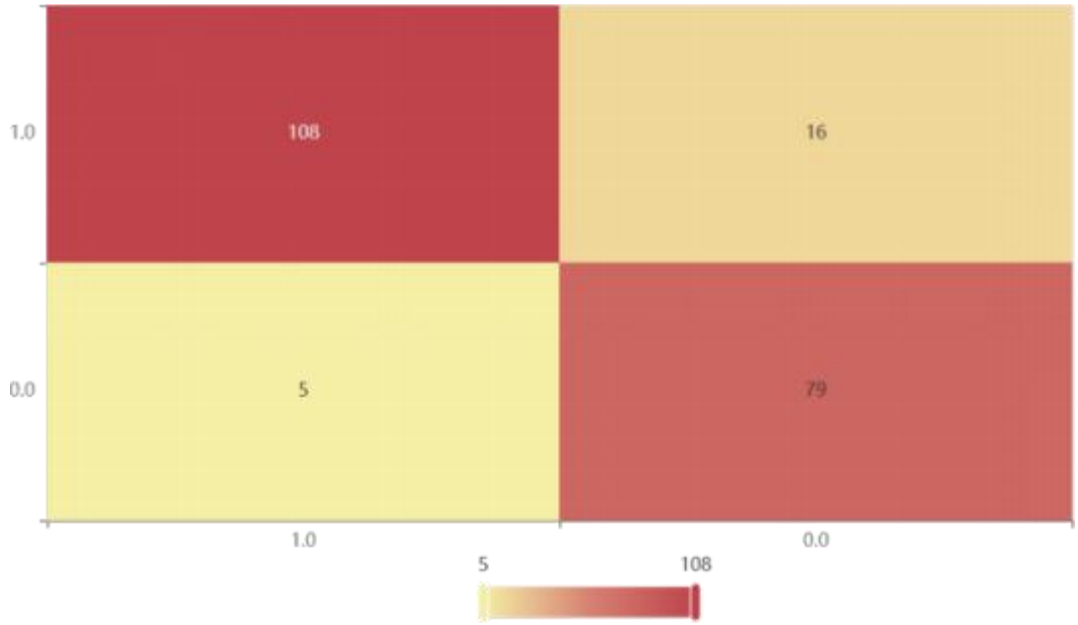

图 5-9: 热力图（澳大利亚）

对于训练集的数据分类效果相对较好。从测试集来看，该模型的性能出现了提升，其各项参数指标达到了 0.9 上下，相比其他模型的性能有着较为优异的表现，因此 BP 神经网络模型的性能更为优越。

表 5-6: BP 神经网络（澳大利亚）

	准确率	召回率	精确率	F1
训练集	0.846	0.846	0.848	0.847
交叉验证集	0.844	0.844	0.849	0.845
测试集	0.899	0.899	0.906	0.9

• 决策树（澳大利亚）：

在利用决策树模型对于澳大利亚数据集进行分析时，我们发现该模型在训练集上的准确率、召回率、精确率和 F1 值均达到了 1 的水平，说明在训练过程中，该模型有着极佳的拟合能力。但是，同其他类模型一样，该模型在测试集上的性能评估参数出现了下降，其三个参数位于 0.83 上下，F1 参数更是下降到了 0.83 之下，说明了该模型的性能在测试集的性能出现了明显下滑，因此，决策树模型的性能差于 BP 神经网络模型。

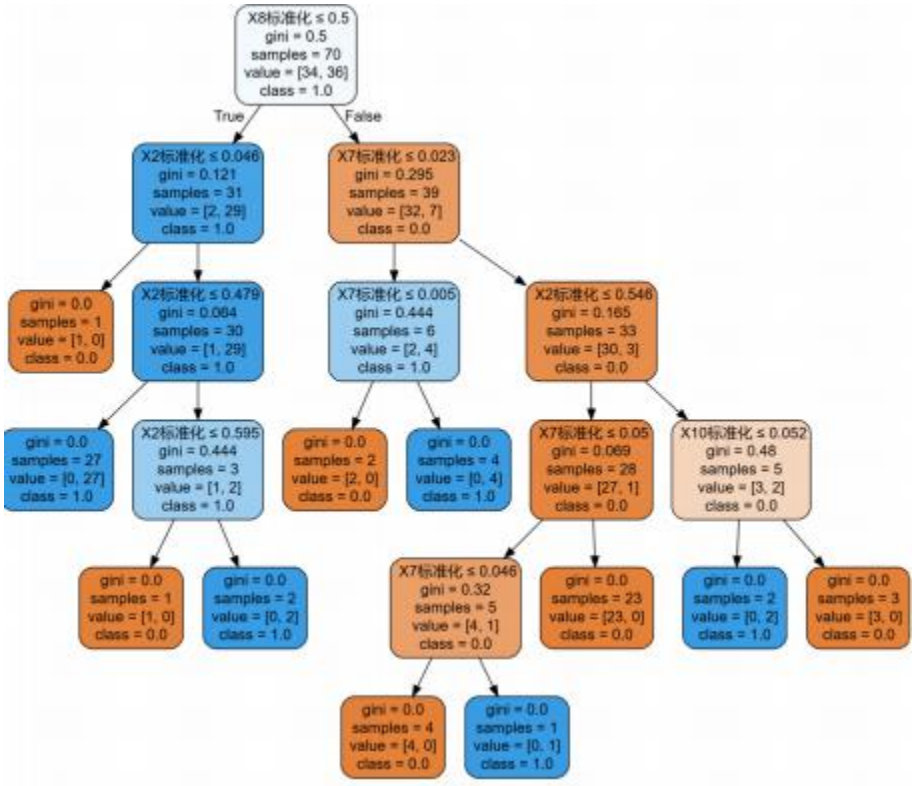

图 5-10: 决策树（澳大利亚）

表 5-7: 决策树（澳大利亚）

	准确率	召回率	精确率	F1
训练集	1	1	1	1
测试集	0.83	0.833	0.836	0.824

- KNN（澳大利亚）：

表 5-8: KNN（澳大利亚）

	准确率	召回率	精确率	F1
训练集	0.9	0.9	0.9	0.9
测试集	0.856	0.856	0.856	0.855

在分类任务中，给定一个新样本 **x**，通过以下步骤来预测其类别：

计算新样本 **x** 与训练集中所有样本之间的距离。选择距离最近的 K 个样本。这 K 个最近邻样本中， 出现次数最多的类别即为新样本的预测类别。假设第 k 个最近邻样本的类别为 y_k ，则新样本 **x** 的预测类别$\hat{y}$为：

$$\hat{y} = \arg\max_{c \in C} \sum_{k=1}^{K} I(y_k = c)$$

其中，C 是所有可能类别的集合， I(·) 是指示函数，当括号内的条件为真时取值为 1，否 则为 0。

在使用 K 近邻（KNN）模型[2]对于澳大利亚的信用数据集分析后，我们发现 K 近邻 （KNN）分类模型在训练集和测试集的效果有所不同。模型在训练数据上能够很好地拟 合数据，模型的准确率、召回率、精确率和 F1 得分均高达 0.9，并且对于正负样本的识 别能力较强。然而，当模型于测试集上应用时，性能出现了一定程度的下降，准确率、 召回率、精确率和 F1 得分均降至 0.856 左右，其中 F1 得分为 0.855。但是，我们注意到 KNN 模型的性能受 K 值选择的影响较大。且该模型易受到噪声的影响以及容易忽略数 据的局部结构。由此看来，KNN 模型的准确率低于 BP 神经网络模型的准确率。

- 支持向量机 SVM（澳大利亚）：

表 5-9: 支持向量机（澳大利亚）

	准确率	召回率	精确率	F1
训练集	0.857	0.857	0.857	0.867
测试集	0.851	0.851	0.866	0.851

在使用支持向量机（SVM）模型对于澳大利亚的信用数据集分析后，我们从模型评估结果中发现， 该模型虽然在训练集和测试集上均取得了相近的性能指标且显示出了模 型的稳定性和泛化能力。但是由于模型训练的复杂度随着数据量的增大而增大，因此在 增大数据量之后，模型的性能出现了下降。在数据分析过程中，由于数据指标较为高维， 支持向量机模型的求解速度明显慢于其他几类模型。从图表来看，支持向量机模型在训 练集和测试集上均达到了 0.85 以上的准确率，但该模型的准确率仍然低于 BP 神经网络 所得到的准确率。

- 随机森林（澳大利亚）：

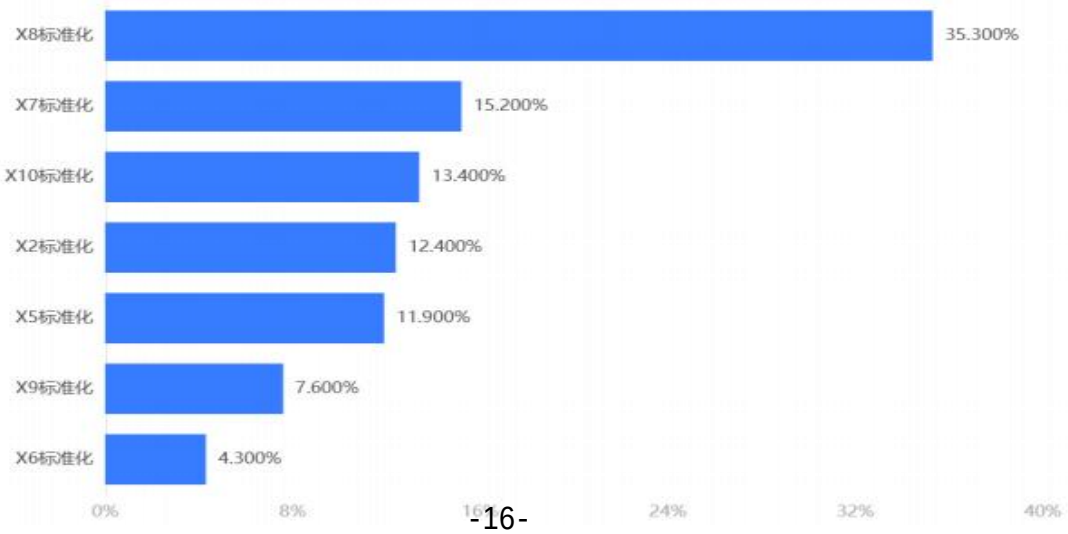

图 5-11: 特征重要性

表 5-10: 随机森林（澳大利亚）

	准确率	召回率	精确率	F1
训练集	0.961	0.961	0.961	0.961
测试集	0.841	0.841	0.841	0.841

在使用随机森林模型[3]对于澳大利亚的信用数据集分析后，我们观察到训练集和测试 集的性能指标有所差异，这体现了模型在未见数据上的泛化能力。在训练集上，准确率、 召回率、精确率和 F1 值均高达 0.961，表明模型在拟合训练数据时表现优异，能够准确 地将样本分类到其对应的类别。但是该模型在测试集上的这些指标均下降至 0.841，说 明了模型在面对新数据时性能有所下降。

对于这种性能下降（或称为过拟合）的情况，我们考虑是由于模型训练过程较为复 杂，导致它过于关注训练数据中的一些特定细节，而忽略了数据的一般规律。我么不难 发现虽然随机森林分类模型在训练集上表现出色，但在测试集上存在明显的性能下降情 况，泛化能力受到限制，分类的准确度和可信度难以得到保障，模型的性能也落后于 BP 神经网络模型。

表 5-11: 澳大利亚信用数据集分类方法对比结果

	Accuracy	AUC	Type1-error	Type2-error
你们的模型	0.899	0.926	0.067	0.033
DT	0.83	0.8095	0.120	0.091
KNN	0.856	0.8805	0.082	0.063
RF	0.851	0.9252	0.077	0.072
SVM	0.841	0.8756	0.029	0.120

- BP 神经网络（德国）：

在利用 BP 神经网络模型对德国信用数据集进行分析后，我们注意到模型在训练集上的准确率、召回率、精确率和 F1 等指标均达到了 0.76 到 0.77，可见该模型在训练数据拟合时具有相对较好的性能，再看测试集的相关数据，我们发现测试集上的模型评估参数得到了一定的提升，准确率、召回率等参数均达到了0.79-0.80的水平，很好地避免了过拟合的问题，因此，BP 神经网络模型的性能优于其他分类模型。

表 5-12: BP 神经网络（德国）

	准确率	召回率	精确率	F1
训练集	0.77	0.77	0.76	0.759
交叉验证集	0.753	0.753	0.742	0.737
测试集	0.797	0.797	0.787	0.79

- 决策树（德国）：

在使用决策树模型对于德国的信用数据集分析后，我们注意到该模型在训练集上的准确率、召回率、精确率以及F1的值均达到了0.87上下的水平，说明该决策树模型对于训 练过程中的数据拟合效果较好，但是其在测试集上的准确率等指标均有大幅度下降，保持在0.7上下的大小，说明该模型在测试集的性能有了明显下降，因此该模型的性能差于 BP 神经网络模型。

表 5-13: 决策树（德国）

	准确率	召回率	精确率	F1
训练集	0.873	0.873	0.871	0.869
测试集	0.703	0.703	0.69	0.694

- KNN（德国）：

在使用 K 近邻（KNN）模型对于德国的信用数据集分析后，我们发现 K 近邻分析模型在 训练集和测试集的效果同样存在差异。在训练集中，KNN 模型的准确率、召回率、精确

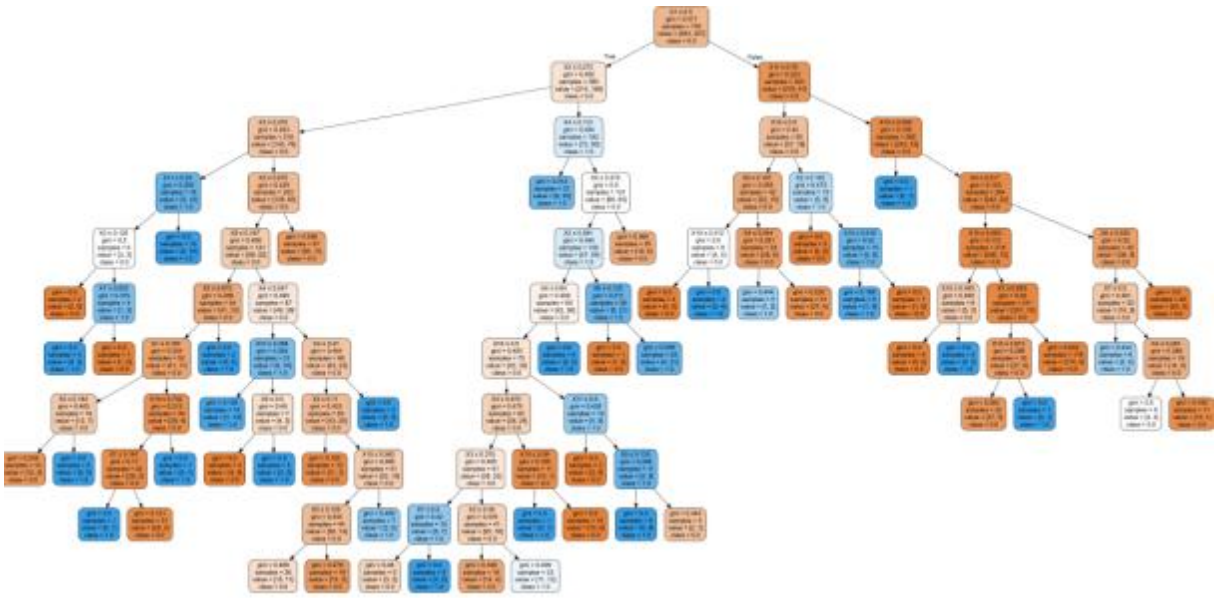

图 5-12: 决策树（德国）

率和 F1 得分均达到了 0.8，能够相对较好地拟合数据。但在测试集中，我们注意到 KNN 模型的各项参数均未达到 0.8，因此，KNN 模型在信用评分的准确性上低于 BP 神经网络模型。

表 5-14: KNN（德国）

	准确率	召回率	精确率	F1
训练集	0.811	0.811	0.805	0.801
测试集	0.733	0.733	0.718	0.72

- 支持向量机（德国）：

在使用支持向量机（SVM）模型对于德国的信用数据集分析后，我们从模型评估结果看出该模型在训练集和测试集上的性能也存在一定的不同，但该模型不同于其他几类模型的是，其准确率、召回率等评估参数在测试集上均高于训练集。我们对于其在测试集上的表现可以看出，其准确率、召回率、精确率和 F1 值浮动在 0.75 至 0.78 范围内，这一大小可以反映出该模型对于数据的拟合效果较好，但仍然低于 BP 神经网络模型。

表 5-15: 支持向量机（德国）

	准确率	召回率	精确率	F1
训练集	0.751	0.751	0.734	0.729
测试集	0.773	0.773	0.764	0.758

- 随机森林（德国）：

在使用随机森林模型对于德国的信用数据集进行分析后，我们观察到该模型在训练集和测试集的性能表现存在着显著的区别。在训练集上，该模型的准确率、召回率、精确率和 F1 值均高于 0.9，说明该模型在进行拟合训练的时候表现优异，但是在测试集上这些指标均出现了明显的下降，各项指标均在 0.8 以下，说明该模型在对未知数据进行拟合时性能有所下降。虽然随机森林在训练时性能优异，但最终得到的测试数据可信度和准确度较低，模型的性能同样低于 BP 神经网络模型。

表 5-16: 随机森林（德国）

	准确率	召回率	精确率	F1
训练集	0.909	0.909	0.916	0.904
测试集	0.77	0.77	0.76	0.754

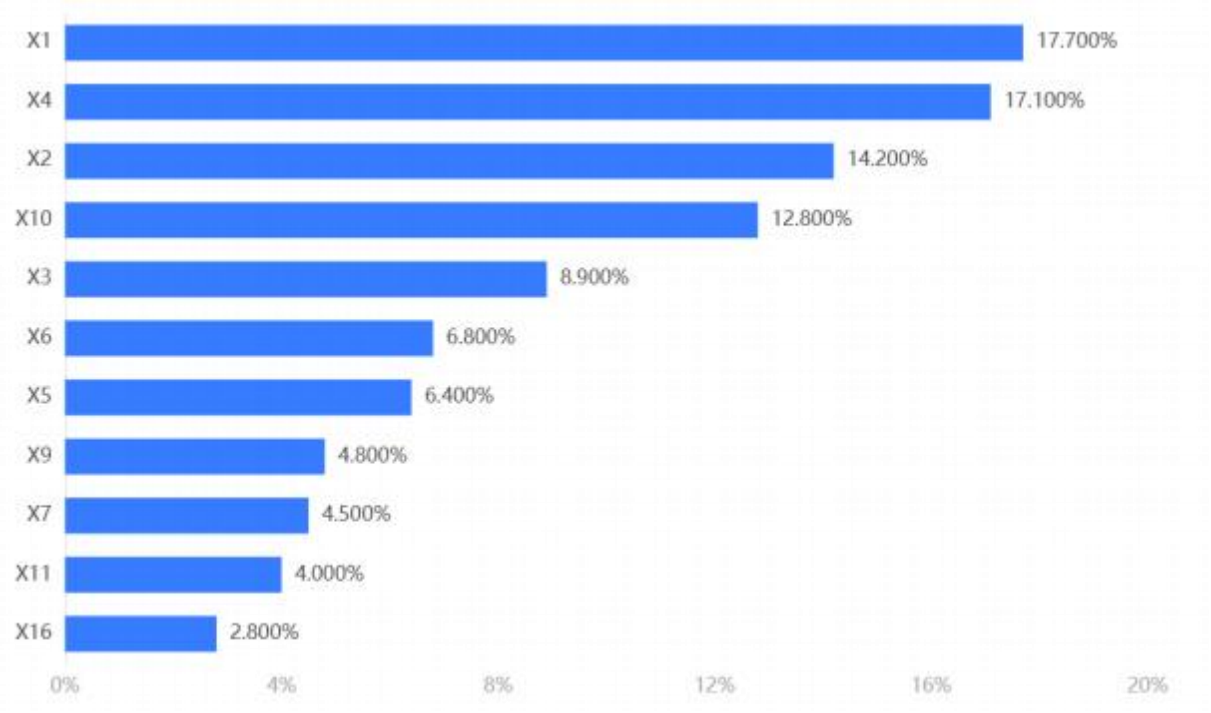

图 5-13: 随机森林（德国）

表 5-17: 德国信用数据集分类方法对比结果

	Accuracy	AUC	Type1-error	Type2-error
你们的模型	0.797	0.802	0.143	0.733
DT	0.703	0.6463	0.177	0.120
KNN	0.733	0.7239	0.177	0.090
RF	0.773	0.7735	0.173	0.060
SVM	0.77	0.6546	0.167	0.060

5.4 问题四模型的建立和分析

5.4.1 模型的建立与求解

信用评级方程是信用评级模型的核心，而信用评级方程的权重系数直接影响着个人信用等级的划分结果。构建非线性规划模型的核心思想在于通过最大化违约与非违约客户间信用得分的离散度差异，逆向求解出各评价指标的最优权重组合。

根据中华人民共和国金融行业标准《信贷市场与银行间债券市场信用评级规范第 2 部分: 信用评级业务规范》(JR/T 0030. 2-2006), 其将企业信用等级划分为 9 个等级包括 AAA、AA、A、BBB、BB、B、CCC、CC 以及 C 级。[4] 本文参照此标准将个人信用等级划分为 9 个等级。

参照现有研究将个人 j 的违约概率 PD_j 转化为信用得分 [5]，转换公式如下：

$$s_j = 650 + 50 * \log_2 [(1 - PD_j)/PD_j] \tag{10}$$

设 DIS 为违约与非违约客户平均信用得分的距离，n_0 用来表示非违约客户的户数，$\mathbf{W} = (w_1, w_1, w_1, \ldots, w_n)$ 为第 j 个指标的权重，$P_i^0(\mathbf{W})$ 为权重向量 **W** 条件下第 i 个非违约客户的信用得分;$P_i^1(\mathbf{W})$ 为权重向量 **W** 条件下第 i 个违约客户的信用得分. $\varphi(P_i^0(\mathbf{W}))$ 为 n_0 户非违约客户信用得分的标准差，类似有 $(P_i^1(\mathbf{W}))$ 为n_0 户违约客户信用得分的标准差。

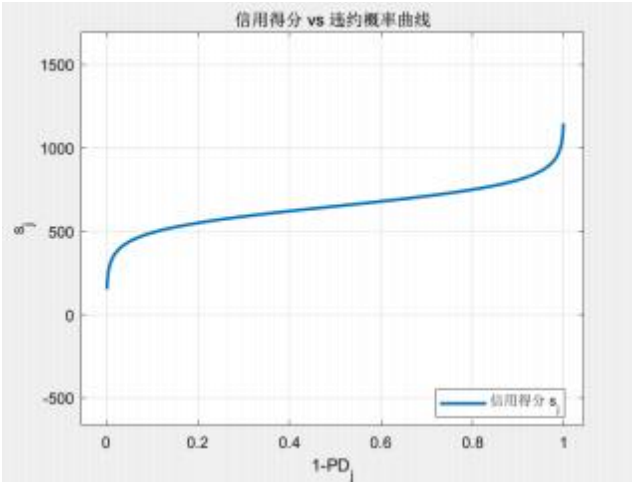

图 5-14: 信用率与信用

其中：

$$DIS = \frac{\sum_{i=1}^{n_0} \frac{P_i^0(\mathbf{W})}{n_0} - \sum_{i=1}^{n_1} \frac{P_i^1(\mathbf{W})}{n_0}}{\sqrt{\varphi(P_i^0(\mathbf{W}))\varphi(P_i^1(\mathbf{W}))}} \tag{11}$$

$$\varphi(P_i^0(\mathbf{W})) = \sqrt{\frac{1}{n_0}\sum_{1}^{n_0}(P_i^0(\mathbf{W}) - \sum_{1}^{n_0} P_i^0(\mathbf{W})/n_0)^2} \tag{12}$$

$$\varphi(P_i^1(\mathbf{W})) = \sqrt{\frac{1}{n_1}\sum_{1}^{n_1}(P_i^1(\mathbf{W}) - \sum_{1}^{n_1} P_i^1(\mathbf{W})/n_1)^2} \tag{13}$$

式 (11) 的分子越大，分母越小，区分程度 D 越大，违约客户与非违约客户的信用得分交叠越少，信用得分越能够区分违约与非违约客户。通过各信用等级对应的证据权重值 WOE 反映等级内违约与非违约客户的个数关系。将所有客户按信用得分由高到低排列后, 划分为多个等级。设 e 为信用等级, 则各信用等级对应的 WOE_e 值为 [6]：

$$WOE_e = \ln(\frac{M_e^1 + 1}{n_1} / \frac{M_e^0 + 1}{n_0}) \tag{14}$$

其中，M_e^1 为等级 e 内违约人数，M_e^0 为等级 e 内非违约人数, 为了保证第e个等级与第 e − 1 个等级之间差异最大化，设 WOE 与 DIS 的影响因子相等，则目标函数可以化简为：

$$obj \max \sum_{e=1}^{n} (WOE_e - WOE_{e-1}) + DIS = \max(WOE_9 - WOE_1) + DIS \tag{15}$$

约束条件为：

$$\text{s.t.} \sum_{j=1}^{n} w_j \tag{16}$$

$$w_j > 0, j = 1, 2, 3, \ldots, n$$

$$WOE_e - WOE_{e-1} > 0, e = 2, \ldots, n \tag{17}$$

类似于《基于违约鉴别能力最大的信用等级划分方法》的信用等级划分 [4]，按照“信用等级越高、损失率越低，信用风险越低”的划分标准，得到划分结果如下表：

信用等级	得分区间	损失率 (%)	人数
AAA	[987.59,-]	0.01	0
AA	[905.73,987.59]	0.05	26
A	[887.60,905.73]	0.16	19
BBB	[822.21,887.60]	0.23	140
BB	[804.00,822.21]	0.80	59
B	[780.69,804.00]	1.23	77
CCC	[699.46,780.69]	1.35	289
CC	[331.99,699.46]	10.10	390
C	[0 ,331.99]	66.90	0

表 5-18: 信用等级划分结果

六、 模型的评价

6.1 模型的优点

1. 多模型综合应用：论文中综合使用了 Lasso 回归模型、BP 神经网络模型等，在信用风险评价中克服了多重共线性、过度拟合等问题，并且突破了传统线性预测的壁垒，增强了模型预测的准确性和可解释性；

2. 模型具有较强的自适应性和鲁棒性，可以通过训练对数据的规律和本质进行自动学习，不需要过多的人工干预来设定复杂的规则或特征，并且对数据中的噪声和不理想值具有一定的容忍能力，能给出相对较好的结果；

3. 很好的泛化能力：模型可以自行识别冗余特征值并进行剔除，降低了计算复杂度和数据存储需求，以达到降维的目的，从而选出信用评价的关键指标，提高了对数据的预测能力。

6.2 模型的缺点

1. 容易陷入局部最优解：在对复杂的信用风险数据进行训练时，由于初始权重的设置等原因，BP 神经网络可能会收敛到局部最小点，而不是全局最优解，不同初始权值会导致不同收敛结果，可能会使模型性能不佳。

2. 参数选择影响：例如 Lasso 回归模型的正则化参数 λ 需要通过交叉验证等方式谨慎选择，BP 神经网络的训练比例与训练速率也需要慎重考虑，选择不当可能会影响模型性能。

3. 模型实际上可能会剔除一些对结果有贡献的特征指标，影响到最终对信用评估的准确性。

参考文献

[1] 李航. 统计学习方法. 北京：清华大学出版社， 2012：第七章，pp.95-135

[2] Scientific Platform Serving for Statistics Professional 2021. SPSSPRO. (Version 1.0.11)[Online Application Software]. Retrieved from https://www.spsspro.com.

[3] 周志华. 机器学习 [M]. 清华大学出版社, 2016.

[4] 梅子行, 毛鑫宇. 智能风控———Python 金融风险管理与评分卡建模 [M]. 北京: 机械工业出版社,2020

[5] 中华人民共和国金融行业标准 (JR/T0030.2-2006) 《信贷市场和银行间债券市场信用评级规范》

[6] Abdou H. A. Genetic Programming for Credit Scoring: The Case of Egyptian Public Sector Banks[J]. Expert Systems with Appli cations, 2009,36(9):11402-11417

附 录

python 程序一

```
inporт nunpy as np
inporт pandas as pd
fron sklearn.tree inporт DecisionTreeClassifier
fron sklearn.feature_selection inporт RFE,RFECV
fron natplotlib inporт pyplot as plt

file_path = IE:\\fujian\\nyfujian1.csvI
df = pd.read_csv(file_path)
nun_cols = len(df.columns)
new_col_nanes = [fIX{i+1}I for i in ran9e(nun_cols)]

df.columns = new_col_nanes
X = df.drop(IX25I,axis=1)

Y = df[IX25I]
print(X.head())
nodel = DecisionTreeClassifier(random_state=42)

rfe = RFE(estinator=nodel, n_features_to_select=11)
X_rfe = rfe.fit_transforn(X, Y)
print(rfe.rankin9_)  #结果是[ 1  1  4  1  6  1  5  3  2  1  7
   9 17 10 18 13 11 15 14 19 16 20 12  8]
rankin9 = rfe.rankin9_
featureˍrankin9 = pd.DataFrane({IFeatureI: X.columns, IRankin9I
   : rankin9})
featureˍrankin9 = featureˍrankin9.sort_values(by=IRankin9I)
plt.fi9ure(fi9size=(12, 6))
plt.barh(featureˍrankin9[IFeatureI], featureˍrankin9[IRankin9I
   ], color=Iskyblue I)
plt.xlabel(IRankin9I)
plt.ylabel(IFeaturesI)
plt.title(IFeature Rankin9 usin9 RFEI)
plt.9ca().inverт_yaxis()  # 倒置 y 轴以使排名从上到下
```

```
plt.show()

'''
print('特征数量: %.3f' %rfe.n_features_)
mean_test_scores = rfe.cv_results_['mean_test_score']
print('auc scores:', mean_test_scores)
plt.xlabel("Number of features selected")
plt.ylabel("auc score")
plt.plot(range(1, len(mean_test_scores) + 1),mean_test_scores)
plt.show()
'''
```

./code/np3.py

matlab 程序二 BP 神经网络

```
%%BP神经网络matlab代码
clc
clear
filename = 'german3.xlsx';
tempp = readtable(filename);
k=rand(1,1000);
[m,n]=sort(k);

%输入输出数据
input=tempp(:,1:11);
output1 =tempp(:,12);
input = table2array(input);
output1 = table2array(output1);
%把输出从1维变成2维，违约转换
output=zeros(1000,2);
for i=1:1000
    switch output1(i,1)
        case 0
            output(i,:)=[1 0];  %[1,0] 没有违约
        case 1
            output(i,:)=[0 1];
    end
end
```

```
%随机提取700个样本为训练样本，  300个样本为预测样本
input_train=input(n(1:700),:)';
output_train=output(n(1:700),:)';
input_test=input(n(701:1000),:)';
output_test=output(n(701:1000),:)';
disp(sum(output_test,2))  %实际上没有违约的
%输入数据归一化
[inputn,inputps]=mapminmax(input_train);

%% 网络结构
innum=11;
midnum=25;
outnum=2;
%权值初始化
w1=rands(midnum,innum);
b1=rands(midnum,1);
w2=rands(midnum,outnum);
b2=rands(outnum,1);

w2_1=w2;w2_2=w2_1;
w1_1=w1;w1_2=w1_1;
b1_1=b1;b1_2=b1_1;
b2_1=b2;b2_2=b2_1;

%学习率
xite=0.1;
alfa=0.01;
loopNumber=10;
I=zeros(1,midnum);
Iout=zeros(1,midnum);
FI=zeros(1,midnum);
dw1=zeros(innum,midnum);
db1=zeros(1,midnum);

%% 训练
LOSS = zeros(1,loopNumber);
```

```
for  ii=1:loopNumber
    LOSS(ii)=0;
    for  i=1:1:700
       %% 网络预测
        x=inputn(:,i);
        % 隐含层输出
        for  j=1:1:midnum
            I(j)=x'*w1(j,:)'+b1(j);    %
            Iout(j)=1/(1+exp(-I(j)));  % sigmoid 函数归一化
        end
        % 输出层输出
        yn=w2'*Iout'+b2;
       %% 权值阀值修正
        %计算误差
        e=output_train(:,i)-yn;
        LOSS(ii)=LOSS(ii)+sum(abs(e));  %Loss 函数

        %计算权值变化率
        dw2=e*Iout;
        db2=e';

        for  j=1:1:midnum
            S=1/(1+exp(-I(j)));
            FI(j)=S*(1-S);
        end

        for  k=1:1:innum
            for  j=1:1:midnum
                dw1(k,j)=FI(j)*x(k)*  sum(e'  .*  w2(j,:));
                db1(j)=FI(j)*sum(e'.*  w2(j,:));
            end
        end

        w1=w1_1+xite*dw1';
        b1=b1_1+xite*db1';
        w2=w2_1+xite*dw2';
        b2=b2_1+xite*db2';
```

```
98
99          w1_2=w1_1;w1_1=w1;
100         w2_2=w2_1;w2_1=w2;
101         b1_2=b1_1;b1_1=b1;
102         b2_2=b2_1;b2_1=b2;
103     end
104 end
105 %%
106 inputn_test=napninnax(Iapply I,input_test,inputps);
107 fore=zeros(2,300);
108 for ii=1:1
109     for i=1:300                                          %1500
110         %隐含层输出
111         for j=1:1:nidnun
112             I(j)=inputn_test(:,i)I*w1(j,:)I+b1(j);
113             Iout(j)=1/(1+exp(-I(j)));
114         end
115         fore(:,i)=w2 I*Iout I+b2;
116     end
117 end
118 %disp(fore)
119
120 %根据网络输出找出数据属于哪类
121 output_fore=zeros(1,300);
122 for i=1:300
123     output_fore(i)=find(fore(:,i)==nax(fore(:,i)));
124 end
125 %BP网络预测误差
126 output1(n(701:1000)) = output1(n(701:1000))+1;
127 error=output_fore-output1(n(701:1000))I;
128
129 fi9ure(1)
130 plot(output_fore,IrI)
131 hold on
132 plot(output1(n(701:1000))I,Ib I)
133 le9end(I预测类别I,I实际类别I)
134 %画出误差图
```

```
figure(2)
plot(error)
title('BP网络分类误差','fontsize',12)
xlabel('检验变量序号','fontsize',12)
ylabel('分类误差','fontsize',12)
k=zeros(1,2);
%找出判断错误的分类 k
for i=1:300
    if error(i)~=0
        [b,c]=max(output_test(:,i));
        switch c
            case 1
                k(1)=k(1)+1;
            case 2
                k(2)=k(2)+1;
        end
    end
end
kk=zeros(1,2);
for i=1:300
    [b,c]=max(output_test(:,i));
    switch c
        case 1
            kk(1)=kk(1)+1;
        case 2
            kk(2)=kk(2)+1;
    end
end

right=(kk-k)./kk;
disp('正确率')
disp(right);
```

./code/bp 神经网络.m

matlab 程序三

```
%% KNN
filename = 'german3.xlsx';
```

```
% 或者选择德国数据文件 'german3.xlsx'

tempp = readtable(filename);

labels = tempp{:, end};
features = tempp{:, 1:end-1};
labels = categorical(labels);

cv = cvpartition(size(tempp, 1), 'HoldOut', 0.3);
Idx = training(cv);
tIdx = test(cv);

tFeatures = features(Idx, :);
tLabels = labels(Idx, :);
testFeatures = features(tIdx, :);
testLabels = labels(tIdx, :);

model = fitcknn(tFeatures, tLabels, 'NumNeighbors', 5);

pre = predict(model, testFeatures);
pre = categorical(pre);

acc = sum(pre == testLabels) / length(testLabels);
disp(['German KNN 的accuracy: ', num2str(acc)]);

Mat = confusionmat(testLabels, pre);
disp('混淆矩阵:');
disp(Mat);

mdlPosterior = fitcknn(tFeatures, tLabels, 'NumNeighbors', 5, '
    Standardize', 1);

[~, scoresTest] = predict(mdlPosterior, testFeatures);

% 提取类别
cats = categories(labels);

```

```
if size(scoresTest, 2) == 2
    pClass = cats{2};
    [X, Y, T, AUC] = perfcurve(testLabels, scoresTest(:, 2),
        pClass);

    figure;
    plot(X, Y);
    xlabel('False Positive Rate');
    ylabel('True Positive Rate');
    title('German KNN 的 ROC');

    disp('German KNN 的 auc:');
    disp(AUC);
end



%% 使用决策树算法
filename = 'Australian_standard.xlsx';
% 或者选择德国数据文件 'german3.xlsx'

tenpp = readtable(filename);

labels = tempp{:, end};
features = tempp{:, 1:end-1};
labels = categorical(labels);

cv = cvpartition(size(tempp, 1), 'HoldOut', 0.3);
Idx = training(cv);
tIdx = test(cv);

tFeatures = features(Idx, :);
tLabels = labels(Idx, :);
testFeatures = features(tIdx, :);
testLabels = labels(tIdx, :);

% 使用决策树算法
```

```
model = fitctree(tFeatures, tLabels);

pre = predict(model, testFeatures);
pre = categorical(pre);

acc = sum(pre == testLabels) / length(testLabels);
disp(['Australian_standard Decision Tree accuracy: ', num2str(
    acc)]);

Mat = confusionmat(testLabels, pre);
disp('混淆矩阵:');
disp(Mat);

% 使用决策树算法并标准化
mdlPosterior = fitctree(tFeatures, tLabels, 'PredictorSelection
    ', 'curvature', 'Surrogate', 'on');
[~, scoresTest] = predict(mdlPosterior, testFeatures);

% 提取类别
cats = categories(labels);

if size(scoresTest, 2) == 2
    pClass = cats{2};
    [X, Y, T, AUC] = perfcurve(testLabels, scoresTest(:, 2),
        pClass);

    figure;
    plot(X, Y);
    xlabel('False Positive Rate');
    ylabel('True Positive Rate');
    title('Australian_standard Tree 的 ROC');

    disp('Australian_standard Tree 的 auc:');
    disp(AUC);
end
```

```
109
110 %% 随机森林
111 filename = 'Australian_standard.xlsx'; % 第三问澳大利亚就用
    Australian_standard.xlsx
112 % 德国 german3.xlsx
113 tempp = readtable(filename);
114 labels = tempp{:, end}
115 features = tempp{:, 1:end-1}
116 labels = categorical(labels);
117
118 cv = cvpartition(size(tempp, 1), 'HoldOut', 0.3);
119 Idx = training(cv);
120 tIdx = test(cv);
121 tFeatures = features(Idx, :);
122 tLabels = labels(Idx, :);
123 testFeatures = features(tIdx, :);
124 testLabels = labels(tIdx, :);
125 model = TreeBagger(100, tFeatures, tLabels, 'Method', '
    classification');
126
127 pre= predict(model, testFeatures);%Accuracy= （TP+TN+FP+FN）/（
    TP+TN）
128 acc = sum(pre == testLabels) / length(testLabels);
129 disp(['accuracy: ', num2str(acc)]);
130 pre = categorical(pre);
131 Mat = confusionmat(testLabels, pre);
132 disp('混淆矩阵：');
133 disp(Mat);
134
135 [~, scores] = predict(model, testFeatures);
136 cats = categories(labels)
137 if size(scores, 2) == 2
138       pClass = cats{2};
139       [X, Y, T, AUC] = perfcurve(testLabels, scores(:, 2),
          pClass);
140       % ROC
141       figure;
```

```
        plot(X, Y);
        xlabel('FP Rate');
        ylabel('T P Rate');
        title('ROC of Australian_standard forest');
        disp("Australian_standard forest 的 auc：")
        disp(AUC)
end


%% SVM算法
filename = 'Australian_standard.xlsx';  % 选择德国数据文件

tempp = readtable(filename);

labels = tempp{:, end};
features = tempp{:, 1:end-1};
labels = categorical(labels);

% 创建训练和测试分区
cv = cvpartition(size(tempp, 1), 'HoldOut', 0.3);
Idx = training(cv);
tIdx = test(cv);

tFeatures = features(Idx, :);
tLabels = labels(Idx, :);
testFeatures = features(tIdx, :);
testLabels = labels(tIdx, :);

model = fitcsvm(tFeatures, tLabels, 'Standardize', true, '
    KernelFunction', 'RBF');

% 使用交叉验证以得到概率估计
mdlPosterior = fitPosterior(model);

% 在测试集上进行预测
[pre, scores] = predict(mdlPosterior, testFeatures);

```

```
pre = categorical(pre);

acc = sum(pre == testLabels) / length(testLabels);
disp(['Australian SVM accuracy: ', num2str(acc)]);

Mat = confusionmat(testLabels, pre);
disp('混淆矩阵:');
disp(Mat);

% 提取类别
cats = categories(labels);

if size(scores, 2) == 2
    pClass = cats{2};
    [X, Y, T, AUC] = perfcurve(testLabels, scores(:, 2), pClass
        );

    % 绘制 ROC 曲线
    figure;
    plot(X, Y);
    xlabel('False Positive Rate');
    ylabel('True Positive Rate');
    title('Australian SVM ROC');

    disp('Australian SVM auc: ');
    disp(AUC);
End
```

./code/问题三代码.m