西南大学研究生课程考试

答　卷　纸

考试科目：	《线性模型》
院、所、中心：	数学与统计学院
专业或专业领域：	应用统计
研究方向：	不区分方向
级　　别：	2020
学　　年：	2020-2021 学年
学　　期：	第二学期
姓　　名：	杜兴兴
学　　号：	112020314201385
类　　别：	②全日制硕士

（①全日制博士 ②全日制硕士 ③教育硕士 ④高师硕士
⑤工程硕士 ⑥农推硕士 ⑦兽医硕士 ⑧进修)

2021 年 6 月 14 日

研 究 生 院 (筹)制

课 程 类 别		②选修课
课程考试方式		④课程论文
题号	得分	教 师 评 价
一		
二		
三		
四		
五		
六		
七		
八		
九		
十		
总分		
任课教师签名：		

备注：成绩评定以百分制或等级制评分，每份试卷均应标明课程类别(①必修课②选修课③同等学力补修课)与考核方式（①闭卷考试②口试③开卷考试④课程论文）。课程论文应给出评语。

目录

图目录

表目录

摘要

由于虚假招聘信息是某些公司或机构为了自身的利益而发布的，发布虚假招聘信息的机构等往往通过借鉴或者抄袭其他公司单位的招聘信息与虚构出过分鼓吹岗位的信息。不具备相关岗位的基本认知和没有虚假信息辨别能力的普通求职者，往往很难从众多的评论中将这些虚假信息识别处理，因此对虚假招聘信息的识别有着一定的意义。

本文所采用的虚假信息识别技术实际上是一种分类算法，将未标注的样本带入模型计算，得到样本处于真、假两个类别的概率。

主要是对在线招聘信息虚假内容进行分类预测，并对比逻辑回归与其他机器学习算法之间的差异。本文的主题思路框架如下图所示。首先根据指定数据集进行数据预处理，主要包括对数据进行编码和划分；接下来在训练集上建立逻辑回归模型和其他 8 种常见的机器学习模型；然后将建立的分类器模型应用到测试集上进行评估与对比，本文采用的评估准则为预测准确率、ROC 曲线等；然后用不同模型计算了变量重要性找出关键影响变量；

最后指出求职者对于网络招聘信息，可以根据本报告得到的“工作福利”、“公司是否有标志”两个指标变量格外关注。网络信息监管平台也可以利用本分析报告得到的模型，结果实际情况指标对在线招聘信息的虚假识别进行合理的预测，进而规范网络环境。

关键字：虚假招聘信息；逻辑回归；机器学习

Abstract

Since false recruitment information is issued by some companies or institutions for their own interests, the institutions that publish false recruitment information often use the recruitment information of other companies or units for reference and create fictitious information that excessively advocates the position. Ordinary job seekers who do not have the basic knowledge of relevant positions and the ability to identify false information often find it difficult to identify and deal with these false information from numerous comments. Therefore, the identification of false recruitment information has a certain significance.

The false information identification technology used in this paper is actually a classification algorithm, which brings the unlabeled samples into the model to calculate the probability that the samples are in two categories: true and false.

It mainly classifies and predicts the false content of online recruitment information, and compares the differences between logistic regression and other machine learning algorithms. The thematic framework of this paper is shown in the figure below. Firstly, data preprocessing is carried out according to the specified data set, which mainly includes data coding and partitioning. Secondly, logistic regression model and other 8 common machine learning models are established on the training set. Then, the established classifier model is applied to the test set for evaluation and comparison. The evaluation criteria adopted in this paper are prediction accuracy, ROC curve, etc. Then the importance of variables was calculated using different models to find out the key influencing variables.

Finally, it is pointed out that job seekers can pay special attention to the two indicator variables of "job benefits" and "whether the company has logo" according to this report. The network information supervision platform can also use the model obtained from this analysis report to reasonably predict the false identification of online recruitment information by the actual situation index, and then standardize the network environment.

Key words: False recruitment information; Logistic regression; Machine learning

第1章 绪论

1.1. 背景介绍

随着互联网技术的应用与普及，也带动了招聘网站的兴起，特别是近年来智能手机的普及使得网络招聘成为了求职者最重要的信息来源，公司也更倾向于在招聘网站上发布求职信息。在线招聘与传统的招聘方式相比，在线招聘信息有着超时空性、传播速度快且范围广、信息有行性、匿名性、非面对面性等特点[1]。因此，许多公司、机构、甚至个人在本身利益的驱动下发布了大批量的虚假评论，以此来误导潜在求职者，这种行为严重危害了人力资源市场的环境和秩序[2]。

由于虚假招聘信息是某些公司或机构为了自身的利益而发布的，因此这些发布虚假招聘信息的机构等往往通过借鉴或者抄袭其他公司单位的招聘信息与虚构出过分鼓吹岗位的信息。不具备相关岗位的基本认知和没有虚假信息辨别能力的普通求职者，往往很难从众多的评论中将这些虚假信息识别处理，因此对虚假招聘信息的识别有着一定的意义[3]。

针对在线招聘的虚假信息识别这个场景，构建真、假信息分类器是解决此需求的关键途径。真、假信息分类器属于机器学习分类算法的范畴，可以视为一个文本分类问题。一般通过构建算法模型学习已标注类别的样本数据集，得到一个分类器来解决真、假分类问题。使用训练好的分类器对未标注的新样本进行分类，输出该样本的分类结果。传统的分类方法主要基于统计学原理的机器学习方法，利用经过人工标注的数据集进行有监督学习得到分类器，从而对新数据集进行分类[4]。目前主流的分类算法有：K 近邻算法（KNN，K-Nearest Neighbor）、决策树（DT，Decision Tree）、逻辑回归（LR，Logistic Regression）、支持向量机（SVM，Support Vector Machine）、集成学习（EL，Ensemble Learning）、神经网络（NN，Neural Network）等。这些众多的分类算法模型都是对分类型的目标变量进行类别分类预测的方法，然而他们都有其独特的理论和丰富的算法逻辑，在复杂多样的商业活动中，其业务需求也是丰富多样的。

1.2. 研究现状

网络招聘中充斥着大量的虚假信息，对于这些虚假信息的识别工作很早就获得了多

方学者的关注并取得了一些成果。2012 年以来，XuQiongkai 等学者在 MyleOtt 研究的基础上，利用 MyleOtt 构建的“黄金”语料库，对特征提取进行了新的尝试。XuQiongkai 等学者从样本的句子结构出发，使用 BIPOS 和 DEP 方法提取高级语言特征，使用最大熵模型与支持向量机分类器作为分类器，LiFangtao 等学者利用网络爬虫爬取了部分招聘的网络评论数据，并通过人工标注的方式构建了样本数据库，实践了基于半监督学习方法的协同过滤算法[5]。

同时随着计算机信息技术的发展，机器学习等算法成为虚假信息识别的研究热点。分类算法可以通过算法程序自动将一群个体分成 n 类（n≥2）。从是否已知类别样本的角度，现有的分类算法可以分为两大类：有监督学习的分类、无监督学习的分类[4]。Ma 等使用深度学习模型对 虚假信息进行提取特征分析，使用了 RNN、LSTM 等循环神经网络模型对虚假信息进行特征提取[6]。Conroy 等人结合用户的网络行为数据，提出了语言提示和机器学习相结合的虚假信息自动检测方法，实验证明其的确具有良好的检测效果[7]。Liu 等人挖掘 Twitter 上的语言特征，并分别将决策树、随机森林以及支持向量机作为分类工具，采用实时性算法实现了虚假信息的实时检测[8]。

1.3. 研究内容

本文主要是对在线招聘信息虚假内容进行分类预测，并对比逻辑回归与其他机器学习算法之间的差异。本文的主题思路框架如下图所示。

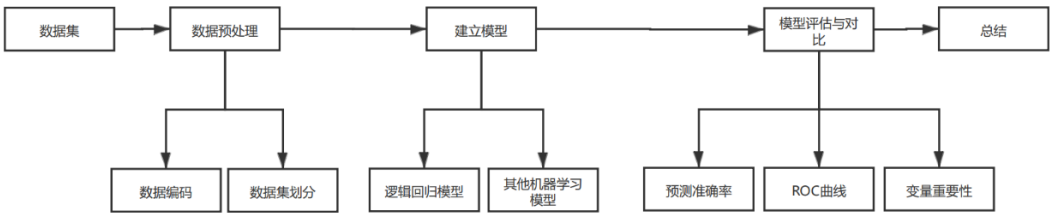

图 1　统计分析流程图

首先根据指定数据集进行数据预处理，主要包括对数据进行编码和划分；接下来在训练集上建立逻辑回归模型和其他 8 种常见的机器学习模型；然后将建立的分类器模型应用到测试集上进行评估与对比，本文采用的评估准则为预测准确率、ROC 曲线等；然后用不同模型计算了变量重要性找出关键影响变量；最后是对全文进行总结与展望。

第2章 方法与数据

2.1. 方法介绍

本文所采用的虚假信息识别技术实际上是一种分类算法，将未标注的样本带入模型计算，得到样本处于真、假两个类别的概率。从而判断该样本属于哪一类。本小节对本文应用到的逻辑回归、决策树、集成学习等机器学习分类算法进行简要介绍，除此之外，还简要介绍常用的模型评价指标和计算变量重要性的原理。

2.1.1. 逻辑回归

Logistic 回归属于概率型的非线性回归，对于二分类的 Logistic 回归，因变量y只有“是、否”两个取值，记为 1 和 0。考虑具有 n 个独立变量的向量$x = (x_1，x_2，\ldots，x_n)$，设条件概率$P(y = 1|x) = p$为根据观测量相对于某事件 x 发生的概率[9]。那么 Logistic 回归模型可以表示为：

$$P(y = 1|x) = \frac{1}{1 + e^{-g(x)}}$$

这里的$f(x) = \frac{1}{1+e^{-x}}$称为 Logistic 函数，其中$g(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$。那么在$x$的条件下$y$不发生的概率为：

$$P(y = 0|x) = 1 - P(y = 1|x) = 1 - \frac{1}{1 + e^{-g(x)}} = \frac{1}{1 + e^{g(x)}}$$

所以事件发生与不发生的概率之比为：

$$\frac{P(y = 1|x)}{P(y = 0|x)} = \frac{p}{1 - p} = e^{g(x)}$$

这个比值称为事件的优势比，简记为 odds。对 odds 取对数得到：

$$\ln\left(\frac{p}{1 - p}\right) = g(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

可以看出 Logistic 回归都是围绕 Logistic 函数来展开的。通过极大似然估计求解模型的参数。假设有m个观测样本，观测值分别为$y_1，y_2，\ldots y_n$，设$p_i = P(y = 1|x_i)$为给定条件下得到$y_i = 1$的概率，同样地，$y_i = 0$的概率为$P(y_i = 0|x_i) = 1 - p_i$，所以得到一个观测值的概率为：

$$P(y_i) = p_i^{y_i}(1 - p_i)^{1-y_i}$$

因为各个观测样本之间相互独立，那么他们的联合分布等于各边缘分布的乘积，得到似然函数为：

$$L(\beta)=\prod_{i=1}^{m}\big(p(x_i)\big)^{y_i}\big(1-p(x_i)\big)^{1-y_i}$$

取对数得：

$$lnL(\beta)=\sum_{i=1}^{m}\{y_i\ln[p(x_i)]+(1-y_i)\ln[1-p(x_i)]\}$$

对这$n+1$个β_i分布求偏导，得到$n+1$个方程：

$$\frac{\vartheta\ln\big(L(\beta_k)\big)}{\vartheta\beta_k}=\sum_{i=1}^{m}x_{ik}[y_i-p(x_i)]=0$$

这样的$n+1$个方程的解即为模型的参数估计。如同所有的回归模型一样，Logisitic回归的主要兴趣是在于参数估计值、估计标准误、t 比率以及统计显著程度。这些信息构成了所有估计广义线性模型的统计分析的核心结果。因为 Logistic 的参数较其它模型的参数更易于解释，对二元因变量，它是最广泛被应用的模型。

2.1.2. 机器学习算法

K 最近邻法是数据挖掘分类技术中最简单的方法之一。根据合适的距离函数计算测试集样本中的每一个观测数据和所有的训练集样本的距离，选择与训练集样本距离最小的 K 个样本作为测试集样本的 K 个最近邻，最后根据测试集样本的 K 个最近邻判断测试集样本的类别，通常 K 是不大于 20 的整数。依赖于训练数据集和 K 的取值，输出结果可能会有不同。

决策树的算法核心体现着特征变量与结果变量之间的某种映射关系，是一个常见的预测模型。决策树中的节点代表对象，分叉路径代表属性值，叶节点表示从根节点到该节点所经历的路径所表示的对象的值[10]。但是决策树拟合的过程容易出现过拟合现象，目前常用处理过拟合的主要手段便是通过剪枝。虽然通过决策树的剪枝可以降低模型过拟合问题，但是同时也可能会发生模型欠拟合问题。所以必须通过合适的剪枝才能得到效果最优的决策树模型。

支持向量机是一种二类分类模型，基本模型是定义在特征空间上的间隔最大的线性分类器，间隔最大使它有别与感知机，支持向量机还包括核技巧，这使它成为实质上的

非线性分类器。其学习策略是间隔最大化（可形似化为一个求解凸二次规划问题），也等价于正则化的合页损失函数的最小化问题。主要分为以下三类：线性可分支持向量机（硬间隔最大化）、线性支持向量机（软间隔最大化）、非线性支持向量机（核技巧和软间隔最大化）。

集成学习的思想是某一个弱分类器得到了错误的预测，其他的弱分类器也可以将错误纠正回来。按照弱分类器的计算方式可以分为并行（bagging）、串行（boosting）和树行（stacking）。并行算法主要包括随机森林，使用多颗决策树联合进行预测可以降低模型的方差，随机森林对训练样本和特征向量的分量都进行了随机采样[11]。串行主要包括 Boosting 算法，通常考虑的也是同质弱学习器，它以一种高度自适应的方法顺序地学习这些弱学习器（每个基础模型都依赖于前面的模型），并按照某种确定性的策略将它们组合起来。由多个弱学习器组成，预测时用每个弱学习器分别进行预测，然后投票得到结果；训练时依次训练每个弱学习器，在这里采用了与随机森林不同的策略，不是对样本进行独立的随机抽样构造训练集，而是重点关注被前面弱分类器错分的样本或是构造样本标签值。具有代表性的是 AdaBoost、GBDT、XGBoost 等。树行是当初始训练数据学习出若干个基学习器后，将这几个学习器的预测结果作为新的训练集，来学习一个新的学习器。该方法通常考虑的是异质弱学习器，并行地学习它们，并通过训练一个元模型将它们组合起来，根据不同弱模型的预测结果输出一个最终的预测结果[11]。可以说 bagging 的重点在于获得一个方差比其组成部分更小的集成模型，而 boosting 和 stacking 则将主要生成偏置比其组成部分更低的强模型（即使方差也可以被减小）。

2.1.3. 变量重要性

一些算法可以在训练时输出变量的重要性，即哪个特征对分类更有用。白箱算法主要通过得到的模型系数查看变量重要性，而黑箱算法目前常见的是通过置换法原理得到变量重要性。置换法远离是指如果某个特征很重要，那么改变样本的该特征值，该样本的预测结果就容易出现错误。也就是说，这个特征对分类结果很敏感。反之，如果一个特征对分类不重要，随便改变它对分类结果没多大影响[13]。训练某决策树时，在包外样本集中随机挑选两个样本，如果要计算某一变量的重要性，则置换这两个样本的这个特征值。对于分类问题，假设置换前样本的预测值为y^*，真实标签为 y，置换之后的预测值为y_π^*。变量重要性的计算公式：

$$v = \frac{n_{y=y^*} - n_{y=y^*_\pi}}{|oob|}$$

其中，$|oob|$为包外样本数；$n_{y=y^*}$为包外集合中在进行特征置换之前被正确分类的样本数；$n_{y=y^*_\pi}$为包外集合中特征置换之后被正确分类的样本数；二者的差反映的是置换前后的分类准确率变化值。

2.1.4. 评估方法

交叉验证法是一种重要的评估方式，先将数据集 D 划分为 K 个大小相似的互斥子集，每个子集都尽可能保持数据分布的一致性，即从 D 中通过分层采样得到；然后每次用 K-1 个子集的并集作为训练集，余下的那个子集作为测试集；得到 k 组训练和测试集，从而可进行 K 此训练和测试；最终返回的是这 K 个测试结果的均值。交叉验证法评估结果的稳定性和保正性在很大程度上取决于 K 的取值，K 最常用的取值是 10，称为 10 折交叉验证，其他常用的还有 5、20。

ROC 曲线是根据学习器的预测结果对样本进行排序，按照此顺序逐个把样本作为正例进行预测，每次计算出两个重要量的值，分别以他们为横纵轴作图，就得到 ROC 曲线。因为 FPR=1-特异度，TPR 也为灵敏度，所以 AUC 曲线可以理解为综合反应特异度和灵敏度这两个度量。分类器是为测试样本产生一个实值或概率预测，然后将这个预测值与一个分类阈值进行比较，若大于阈值则预测为正类，否则为反类。选取不同的阈值（threshold）就可以得到一组 FPR 和 TPR，即 ROC 曲线上的点。若将阈值设为 1，则为 ROC 曲线上（0，0）点，设为 0，则为 ROC 曲线上（1，1）点。ROC 曲线下面积，数值介于 0.5-1 之间，数值越大越好，AUC 是一个概率值，当前分类算法将正样本排在负样本前面的概率。

2.2. 数据说明

数据来源于国外某研究机构。总计 894 条观测样本，包含的变量如下表 1：

表 1 数据变量说明表

变量名称	英文名称	详细说明	取值范围
是否为虚假招聘信息	fraudulent	定性变量（2 水平）	1=是，0=否
公司基本信息	description	定性变量（3 水平）	消极/中立/积极

岗位要求	requirements	定性变量（4 水平）	消极/中立/积极/未知
工作福利	benefits	定性变量（4 水平）	消极/中立/积极/未知
工作部门	department	定性变量（7 水平）	其他/文员/工程/市场/销售/服务/未知
薪资范围	salary	定性变量（4 水平）	低于 5000/5000-50000/高于 50000/未知
是否接受远程办公	teleworking	定性变量（2 水平）	1=是，0=否
公司是否有标志	logo	定性变量（2 水平）	1=是，0=否
工作类型	type	定性变量（6 水平）	其他/合同制/全职/兼职/临时/未知
经验要求	experience	定性变量（8 水平）	助理/主管/入门级/执行级/实习/中高级/不适用/未知
学历要求	education	定性变量（7 水平）	其他/学士/高中或同等学历/硕士/一些高中课程/未知/不确定
公司行业类别	industry	定性变量（13 水平）	其他/会计/计算机/健康/互联网/休闲/营销/媒体/石油/房地产/服务/电信/未知
公司地点	country	定性变量（8 水平）	其他/非洲联盟/德国/英国/希腊/新西兰/美国/未知

2.3. **分析说明**

采集到的数据已使用情感分析来量化指标变量，并对所有数据进行了一定的预处理。本文通过 Python3.6 软件对数据进行进一步分析，使用到的程序包有 os、numpy、pandas、seaborn、matplotlib、sklearn、statsmodels。主要使用到的统计分析方法有 K 近邻算法、决策树算法、支持向量机算法、随机森林算法、神经网络算法、逻辑回归算法。本文通

过该数据集来研究逻辑回归和其他机器学习模型的应用区别，通过逻辑回归和其他黑箱算法的机器学习方法的不同除分类预测能力之外在其他业务需求上的应用。

首先对所有分类变量进行哑变量编码，然后对所有原始数据进行划分，数据集包含894 条观测数据，按照分层随机抽样的准则，选取 70%的样本作为训练集，剩余的 30%样本作为测试集。通过在训练集上建立分类器模型，然后在测试集上评估模型。训练集共 626 条观测样本，测试集共 268 条观测样本。

第3章　结果

3.1. 描述性分析

为了更清楚的观察各分类自变量不同水平在是否为虚假信息之间的比例，绘制了所有自变量的频率直方图，详见下图。

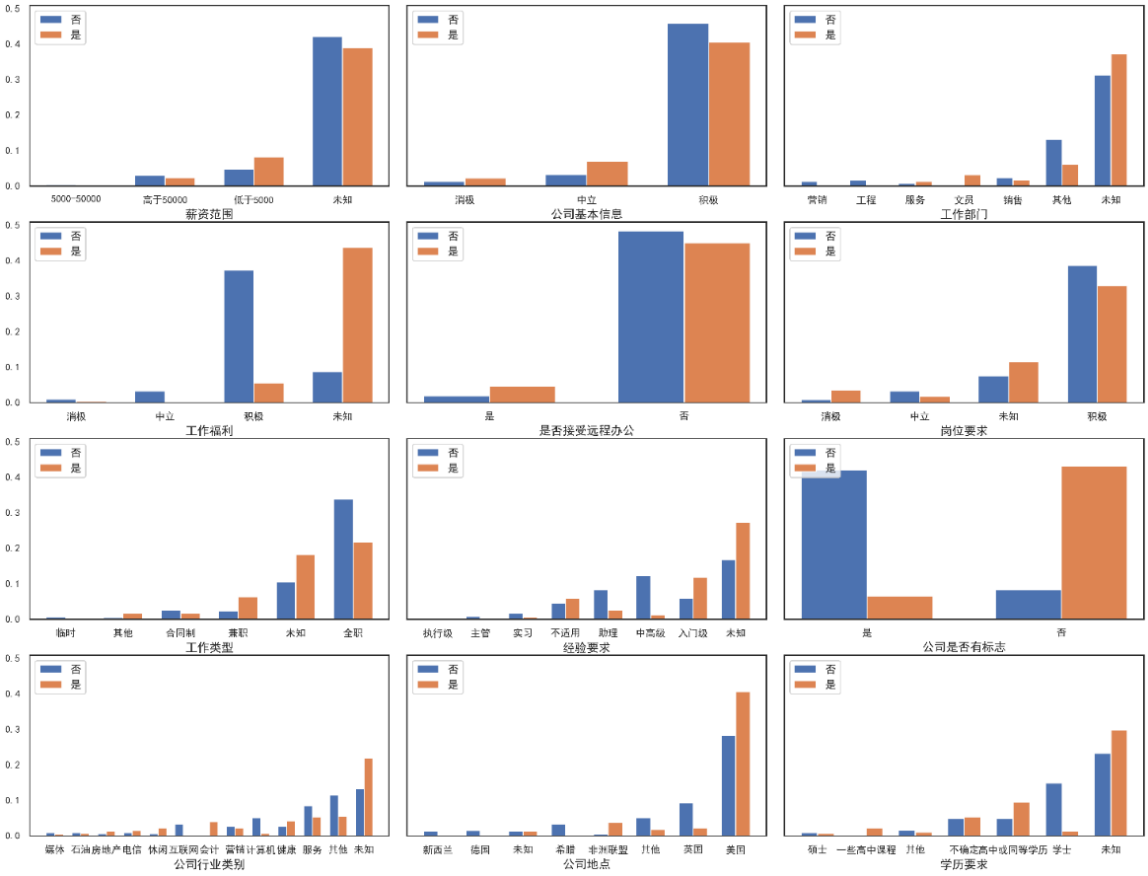

图 2　频数直方图

从图 2 中可以看出，在“薪资范围”、“工作部门”、“工作福利”、“经验要求”、“公司行业类别”、“学历要求”变量中，“未知”水平所占比例明显高于其他水平，存在“未知”水平的只有“工作类型”变量不是该水平占比最高。且在“薪资范围”、“公司基本情况”两个变量中“未知”水平不是虚假信息的比例高于是虚假信息的比例，其他的变量中“未知”水平不是虚假信息的比例低于是虚假信息的比例。

对于“薪资范围”变量，虚假信息主要分布在“低于 5000”和“未知”水平中。对于“公司基本信息”可以看出总体信息中“积极”水平占了很大比例，其中大部分不是虚假信息，但是“中立”水平中是虚假信息的比例有明显高于不是虚假信息。“工作部

门”中“其他”水平里面不是虚假信息的比例明显高于是虚假信息。“工作福利”类别中虚假信息只存在“积极”、“未知”，只有真实信息中才存在“消极”和“中立”，同时“积极”、“未知”水平中是否虚假信息的比例差距明显。从“是否接受远程办公”、“岗位要求”、“工作类型”、“公司行业类别”、“公司地点”几个类别中发现虚假信息与真实信息的差异不明显。对于“经验要求”变量中，大部分虚假信息为“未知”且比例明显高于真实信息，且虚假招聘信息主要发布在“入门级”、“不适用”、“助理”等较低的经验要求上。“公司是否有标志”变量中虚假信息几乎都没有 Logo，而真实信息中大都有 Logo。对于“学历要求”方面，除开“未知”水平外，虚假信息普遍对学历要求较低，而真实信息对学历要求较高。

通过分析，可以初步形成一定的推断，在“公司是否有标志”、“工作福利”、“学历要求”变量上，对于虚假信息的识别上具有明显的辨识度。而“是否接受远程办公”、“岗位要求”、“工作类型”、“公司行业类别”、“公司地点”几个变量对于虚假信息的辨识度较低。除此之外，各类别的“未知”水平可能会对分类识别产生一定效果。

3.2. 单因素分析

为了检验各变量对于虚假信息识别的显著性，考虑到所有自变量全为分类变量，故采用卡方检验对各自变量与因变量的频数分布表进行差异性检验，结果如下表 2。

表 2　各变量的基本统计特征结果表

Variable	Level	False: N(%)	True: N(%)	Chiq	P Value
工作部门					
	其他	117(13.09%)	54(6.04%)	77.002	<0.001(***)
	工程	14(1.57%)	1(0.11%)		
	文员	0(0.0%)	28(3.13%)		
	服务	6(0.67%)	12(1.34%)		
	未知	280(31.32%)	332(37.14%)		
	营销	12(1.34%)	2(0.22%)		
	销售	21(2.35%)	15(1.68%)		
薪资范围					
	5000-50000	4(0.45%)	1(0.11%)	11.947	0.008(**)

	低于 5000	42(4.7%)	73(8.17%)		
	未知	377(42.17%)	349(39.04%)		
	高于 50000	27(3.02%)	21(2.35%)		
公司基本信息					
	中立	29(3.24%)	62(6.94%)	16.793	<0.001(***)
	消极	12(1.34%)	20(2.24%)		
	积极	409(45.75%)	362(40.49%)		
岗位要求					
	中立	29(3.24%)	15(1.68%)	30.795	<0.001(***)
	未知	68(7.61%)	103(11.52%)		
	消极	7(0.78%)	31(3.47%)		
	积极	346(38.7%)	295(33.0%)		
工作福利					
	中立	29(3.24%)	0(0.0%)	450.911	<0.001(***)
	未知	78(8.72%)	391(43.74%)		
	消极	9(1.01%)	3(0.34%)		
	积极	334(37.36%)	50(5.59%)		
是否接受远程办公					
	否	433(48.43%)	402(44.97%)	10.801	0.001(**)
	是	17(1.9%)	42(4.7%)		
公司是否有标志					
	否	74(8.28%)	386(43.18%)	441.784	<0.001(***)
	是	376(42.06%)	58(6.49%)		
工作类型					
	临时	6(0.67%)	1(0.11%)	69.456	<0.001(***)
	全职	302(33.78%)	195(21.81%)		
	其他	4(0.45%)	14(1.57%)		
	兼职	21(2.35%)	57(6.38%)		
	合同制	23(2.57%)	14(1.57%)		

	未知	94(10.51%)	163(18.23%)		
经验要求					
	不适用	40(4.47%)	52(5.82%)	157.066	<0.001(***)
	中高级	109(12.19%)	11(1.23%)		
	主管	7(0.78%)	2(0.22%)		
	入门级	53(5.93%)	106(11.86%)		
	助理	74(8.28%)	22(2.46%)		
	实习	14(1.57%)	5(0.56%)		
	执行级	3(0.34%)	2(0.22%)		
	未知	150(16.78%)	244(27.29%)		
学历要求					
	一些高中课程	0(0.0%)	19(2.13%)	144.752	<0.001(***)
	不确定	44(4.92%)	48(5.37%)		
	其他	14(1.57%)	9(1.01%)		
	学士	133(14.88%)	11(1.23%)		
	未知	208(23.27%)	266(29.75%)		
	硕士	8(0.89%)	6(0.67%)		
	高中或同等学历	43(4.81%)	85(9.51%)		
公司行业类别					
	互联网	29(3.24%)	0(0.0%)	155.024	<0.001(***)
	休闲	5(0.56%)	20(2.24%)		
	会计	1(0.11%)	35(3.91%)		
	健康	23(2.57%)	37(4.14%)		
	其他	102(11.41%)	49(5.48%)		
	媒体	8(0.89%)	3(0.34%)		
	房地产	5(0.56%)	11(1.23%)		
	服务	76(8.5%)	48(5.37%)		
	未知	118(13.2%)	196(21.92%)		
	电信	7(0.78%)	13(1.45%)		

	石油	7(0.78%)	6(0.67%)		
	营销	23(2.57%)	20(2.24%)		
	计算机	46(5.15%)	6(0.67%)		
公司地点					
	其他	45(5.03%)	15(1.68%)	150.871	<0.001(***)
	希腊	30(3.36%)	0(0.0%)		
	德国	13(1.45%)	0(0.0%)		
	新西兰	11(1.23%)	0(0.0%)		
	未知	11(1.23%)	12(1.34%)		
	美国	253(28.3%)	363(40.6%)		
	英国	83(9.28%)	20(2.24%)		
	非洲联盟	4(0.45%)	34(3.8%)		

从检验结果可以看出，在检验水平$\alpha = 0.05$的情况下，所有自变量均通过显著性检验，表明各自变量与是否为虚假信息之间存在着较强的显著性差异。因此，可以将所有自变量纳入分类模型中。虽然所有自变量都通过显著性检验，但是可以发现“薪资范围”、“是否可以接受远程办公”两个信息指标的显著性相对于其他变量而言明显较低。推测在全模型多变量的模型中这两个变量可能对模型的结果影响较弱。从表中还可以发现，“学历要求”、“公司行业类别”、“工作福利”、“公司地点”等变量存在部分水平无样本的情况，且部分变量存在水平样本量小于 5，由于普通的卡方检验的理论条件限制，本文对这些变量采用的是 Fisher 卡方检验。

此外，由于是检验的实际频数与期望频数之间的差异，所以得到的差异性也不能完全说明自变量和因变量之间的影响关系，还需要进一步建立更加全面的模型加以分析。

3.3. Logistic 回归全模型分析

根据因变量为分类变量的特点，针对训练数据，本文首先用逻辑回归的方法对包含全部自变量的全模型进行估计，得到估计结果如表 3 所示。

表 3　逻辑回归模型参数估计结果表

Variables	Level	Estimate	Std.Err	Z-Value	p-Value	OR
Intercept		-78.547	123044.912	-0.001	0.999	

公司地点						
	其他 vs. 希腊	-33.306	43508.384	-0.001	0.999	<-999.999
	其他 vs. 德国	-21.433	63415.145	0.000	1.000	<-999.999
	其他 vs. 新西兰	-19.074	67032.756	0.000	1.000	<-999.999
	其他 vs. 未知	**2.941**	**1.321**	**2.227**	**0.026(**)**	**6.192**
	其他 vs. 美国	1.208	0.768	1.573	0.116	4.211
	其他 vs. 英国	-0.056	0.939	-0.059	0.953	0.910
	其他 vs. 非洲联盟	1.465	1.342	1.092	0.275	15.480
公司基本信息						
	中立 vs. 消极	-1.405	1.363	-1.031	0.303	1.073
	中立 vs. 积极	**-1.519**	**0.677**	**-2.243**	**0.025(**)**	**0.479**
公司是否有标志						
	否 vs. 是	**-3.149**	**0.615**	**-5.117**	**0.000(****)**	**0.027**
公司行业类别						
	互联网 vs. 休闲	28.500	33943.247	0.001	0.999	<0.001
	互联网 vs. 会计	47.586	51988.311	0.001	0.999	<0.001
	互联网 vs. 健康	24.380	33943.247	0.001	0.999	<0.001
	互联网 vs. 其他	24.321	33943.247	0.001	0.999	<0.001
	互联网 vs. 媒体	27.427	33943.247	0.001	0.999	<0.001
	互联网 vs. 房地产	23.481	33943.247	0.001	0.999	<0.001
	互联网 vs. 服务	25.549	33943.247	0.001	0.999	<0.001
	互联网 vs. 未知	24.358	33943.247	0.001	0.999	<0.001
	互联网 vs. 电信	25.024	33943.247	0.001	0.999	<0.001
	互联网 vs. 石油	25.803	33943.247	0.001	0.999	<0.001
	互联网 vs. 营销	24.520	33943.247	0.001	0.999	<0.001
	互联网 vs. 计算机	22.757	33943.247	0.001	0.999	<0.001
学历要求						
	一些高中课程 vs. 不确定	-19.922	55770.894	0.000	1.000	<-999.999
	一些高中课程 vs. 其他	-20.926	55770.894	0.000	1.000	<-999.999

	一些高中课程 vs. 学士	-23.335	55770.894	0.000	1.000	<-999.999
	一些高中课程 vs. 未知	-21.805	55770.894	0.000	1.000	<-999.999
	一些高中课程 vs. 硕士	-20.195	55770.894	0.000	1.000	<-999.999
	一些高中课程 vs. 高中或同等学历	-18.838	55770.894	0.000	1.000	<-999.999
岗位要求						
	中立 vs. 未知	-0.216	0.904	-0.239	0.811	4.165
	中立 vs. 消极	1.881	2.507	0.750	0.453	32.808
	中立 vs. 积极	0.641	0.824	0.778	0.437	2.176
工作福利						
	中立 vs. 未知	27.612	33358.602	0.001	0.999	<-999.999
	中立 vs. 消极	25.725	33358.602	0.001	0.999	<-999.999
	中立 vs. 积极	24.907	33358.602	0.001	0.999	<-999.999
工作类型						
	临时 vs. 全职	25.325	92116.120	0.000	1.000	<-999.999
	临时 vs. 其他	46.922	96885.651	0.000	1.000	<-999.999
	临时 vs. 兼职	25.436	92116.120	0.000	1.000	<-999.999
	临时 vs. 合同制	25.940	92116.120	0.000	1.000	<-999.999
	临时 vs. 未知	26.017	92116.120	0.000	1.000	<-999.999
工作部门						
	其他 vs. 工程	2.363	1.808	1.307	0.191	0.264
	其他 vs. 文员	41.850	51663.680	0.001	0.999	<-999.999
	其他 vs. 服务	**2.507**	**1.163**	**2.155**	**0.031(**)**	**6.350**
	其他 vs. 未知	-0.420	0.572	-0.734	0.463	2.857
	其他 vs. 营销	-17.674	70558.504	0.000	1.000	<-999.999
	其他 vs. 销售	**2.269**	**0.977**	**2.322**	**0.020(**)**	**1.338**
是否接受远程办公						
	否 vs. 是	0.700	1.106	0.633	0.527	2.742
经验要求						

	不适用 vs. 中高级	-1.295	0.994	-1.303	0.193	0.069
	不适用 vs. 主管	2.092	1.750	1.195	0.232	0.351
	不适用 vs. 入门级	**1.211**	**0.727**	**1.666**	**0.096(*)**	**1.553**
	不适用 vs. 助理	**1.727**	**0.873**	**1.978**	**0.048(**)**	**0.293**
	不适用 vs. 实习	2.600	5.596	0.465	0.642	0.377
	不适用 vs. 执行级	0.121	2.219	0.054	0.957	0.293
	不适用 vs. 未知	**2.437**	**0.835**	**2.917**	**0.004(***)**	**1.476**
薪资范围						0.220
	5000-50000 vs. 低于 5000	23.332	35767.927	0.001	0.999	1.293
	5000-50000 vs. 未知	23.065	35767.927	0.001	0.999	0.834
	5000-50000 vs. 高于 50000	21.958	35767.927	0.001	1.000	0.670

从表 3 中我们可以看到，模型 F 检验的 P 值非常小（P 值<0.001），表明该模型是显著的，即至少有一个招聘信息指标对该招聘信息是否为虚假信息具有一定的预测能力。进一步通过对各自变量对应的 Z 检验的 P 值的考察，可以得到以下重要结论：

“公司地点”为“其他 vs. 未知”的系数显著为正数（2.941），说明在给定其他条件相同的情况下，“公司地点”为“未知”水平比“其他”水平更容易是虚假信息，且“未知”水平中是虚假信息的可能性是“其他”水平的 6.192 倍，即该水平作为信息的危险因素（OR>1）；

“公司基本信息”为“中立 vs. 积极”的系数显著为负数（-1.519），说明在给定其他条件相同的情况下，“公司基本信息”为“积极”水平比“中立”水平更不容易是虚假信息，且“积极”水平中是虚假信息的可能性是“中立”水平的 0.479 倍，即该水平作为信息的保护因素（OR < 1）；

“公司是否有标志” 为“否 vs. 是”的系数显著为负数（-3.149），说明在给定其他条件相同的情况下，“公司是否有标志”为“是”水平比“否”水平更不容易是虚假信息，且“是”水平中是虚假信息的可能性是“否”水平的 0.027 倍，即该水平作为信息的保护因素（OR < 1）；

“工作部门”为“其他 vs. 服务”的系数显著为正数（2.507），说明在给定其他条件相同的情况下，“工作部门”为“服务”水平比“其他”水平更容易是虚假信息，且“服务”水平中是虚假信息的可能性是“其他”水平的 6.350 倍，即该水平作为信息的危险因素（OR>1）；为“其他 vs. 销售”的系数显著为正数（2.269），说明在给定其他条件

相同的情况下，“工作部门”为“销售”水平比“其他”水平更容易是虚假信息，且“销售”水平中是虚假信息的可能性是“其他”水平的 1.338 倍，即该水平作为信息的危险因素（OR>1）；

“经验要求”为“不适用 vs. 入门级”的系数显著为正数（1.211），说明在给定其他条件相同的情况下，“经验要求”为“入门级”水平比“不适用”水平更容易是虚假信息，且“入门级”水平中是虚假信息的可能性是“其他”水平的 1.553 倍，即该水平作为信息的危险因素（OR>1）；为“不适用 vs. 助理”的系数显著为正数（1.727），说明在给定其他条件相同的情况下，“经验要求”为“助理”水平比“不适用”水平更容易是虚假信息，且“助理”水平中是虚假信息的可能性是“其他”水平的 0.293 倍，即该水平作为信息的危险因素（OR<1）；为“不适用 vs. 未知”的系数显著为正数（2.437），说明在给定其他条件相同的情况下，“经验要求”为“未知”水平比“不适用”水平更容易是虚假信息，且“未知”水平中是虚假信息的可能性是“其他”水平的 1.476 倍，即该水平作为信息的危险因素（OR>1）；

3.4. 逻辑回归与其他算法的对比

本文后续建立了除逻辑回归之外的 8 中机器学习算法（KNN、DT、AdaBoost、GBDT、XGBoost、SVM、RF、MLP），由于篇幅有限，本文不再对其他 8 种模型结果进行一一介绍与介绍。建立模型的一个重要目的就是预测，因此我们对得到的 9 种不同预测模型的效果予以比较。

具体来说，本文利用单独的测试数据来对虚假信息识别进行预测，并用对应的预测结果来衡量模型的预测精度。对此类分类变量的预测，本文采用以下指标直接或间接地度量预测精度：ROC 曲线下面积、混淆矩阵、正确率，从而得到逻辑回归与其他机器学习算法在虚假信息识别问题上的差异情况。

3.4.1. 混淆矩阵

热力图体现了两个离散变量之间的组合关系，通过热力图我们可以非常直观地感受数值大小的差异状况。因此本文将 9 种模型预测结果的混淆矩阵通过热力图呈现，如图 3。

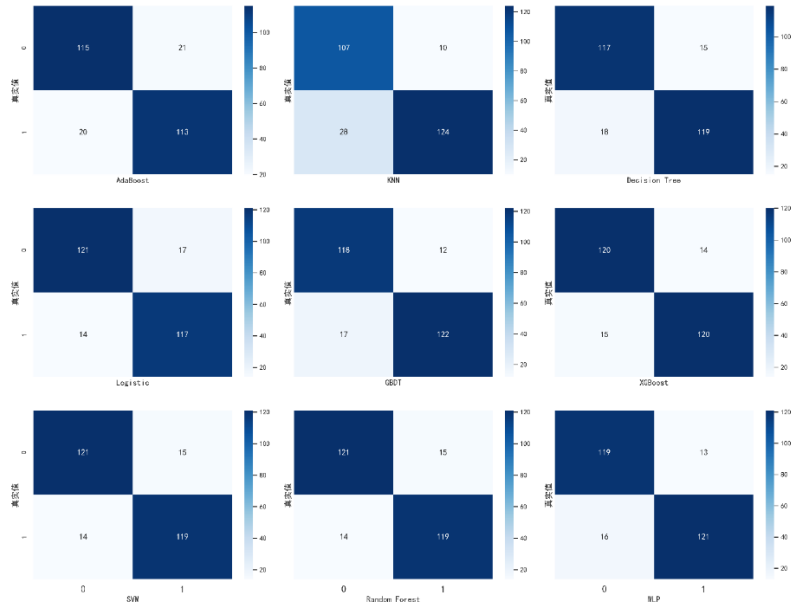

图 3　9 种机器学习模型预测混淆矩阵的热力图

从图中可以发现，对于正确预测识别真假信息的比例在 9 种模型都较大。但是 KNN 模型对于真实信息更容易识别为虚假信息，从混淆矩阵中得到其他几种模型的预测结果大体差异较小。

3.4.2. ROC 曲线

对于二分类预测问题，不同的阈值所对应的预测结果和精度都不一样。由于高 TPR 值总是对应着高 FPR 值，所以本文用 ROC 曲线来进行全面的度量比较 9 种模型，见图 4。

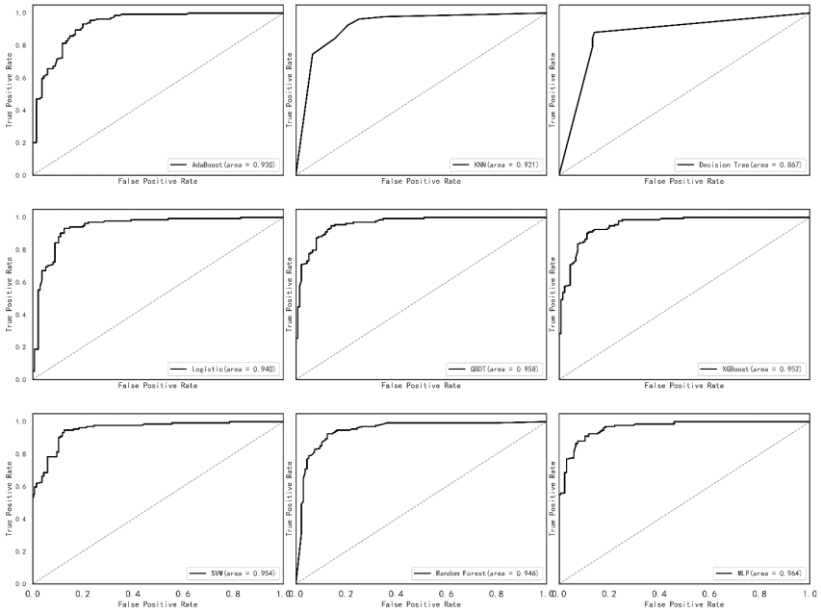

图 4　9 种机器学习算法预测的 ROC 曲线图

结果显示，ROC 曲线与对角线相比，永远是向上突起的。这表明我们的 9 种预测模型都是有效的。曲线下面积（AUC）最大的是 MLP 模型（AUC=0.964），AUC 最小的两个模型是 DT 模型（AUC=0.867）、KNN 模型（AUC=0.921），而 Logistic 模型的 AUC=0.940。由于 DT 模型和 KNN 模型的 AUC 值较低，因此可以得到 KNN 和决策树（DT）模型的预测精度较差。而其他模型的 ROC 曲线非常相似，也就是说他们的预测能力相当。

分析结果说明 Logistic 模型相对比大多数黑箱的机器学习算法而言，预测精度明显高于较为简单的 KNN 模型和 DT 模型，预测效果和较为复杂的集成学习算法和 SVM 算法接近，但是逻辑回归却有一个较为简单且能够进行变量解释的模型结果，这一点是其他法的机器学习的黑箱算法所不及的。

3.4.3. 交叉验证

为了验证上述 9 种机器学习模型的稳定性（模型在新数据集上的泛化能力）。需要确保由数据集得到的机器学习模型已经获得数据集大部分正确的信息，并且不能包含太多的噪声，也就是说，模型的偏差和方差是较小的。本文通过 10 折交叉验证来评估 9 个模型在独立数据集上的概括能力，结果如下表 4 所示。

表 4　9 种机器学习模型 10 折交叉验证的正确率

Num	AdaBoost	KNN	DT	Logistic	GBDT	XGBoost	SVM	RF	MLP
1	0.855	0.866	0.892	0.885	0.892	0.892	0.900	0.903	0.907
2	0.892	0.900	0.896	0.911	0.933	0.903	0.911	0.907	0.941
3	0.848	0.866	0.888	0.870	0.896	0.896	0.888	0.907	0.874
4	0.833	0.844	0.870	0.829	0.866	0.881	0.859	0.888	0.877
5	0.862	0.877	0.862	0.892	0.881	0.896	0.903	0.885	0.911
6	0.885	0.896	0.881	0.896	0.914	0.922	0.911	0.937	0.937
7	0.844	0.877	0.874	0.870	0.877	0.892	0.888	0.911	0.877
8	0.888	0.914	0.892	0.918	0.933	0.918	0.941	0.926	0.944
9	0.851	0.866	0.896	0.874	0.903	0.892	0.892	0.900	0.900
10	0.885	0.881	0.874	0.900	0.892	0.900	0.907	0.903	0.907

箱线图利用数据的五个统计量可以初略的识别数据是否具有对称性，分布的离散程度等信息。因此为了便于观察，对交叉验证得到的正确率绘制箱线图，结果如图 5 所示。

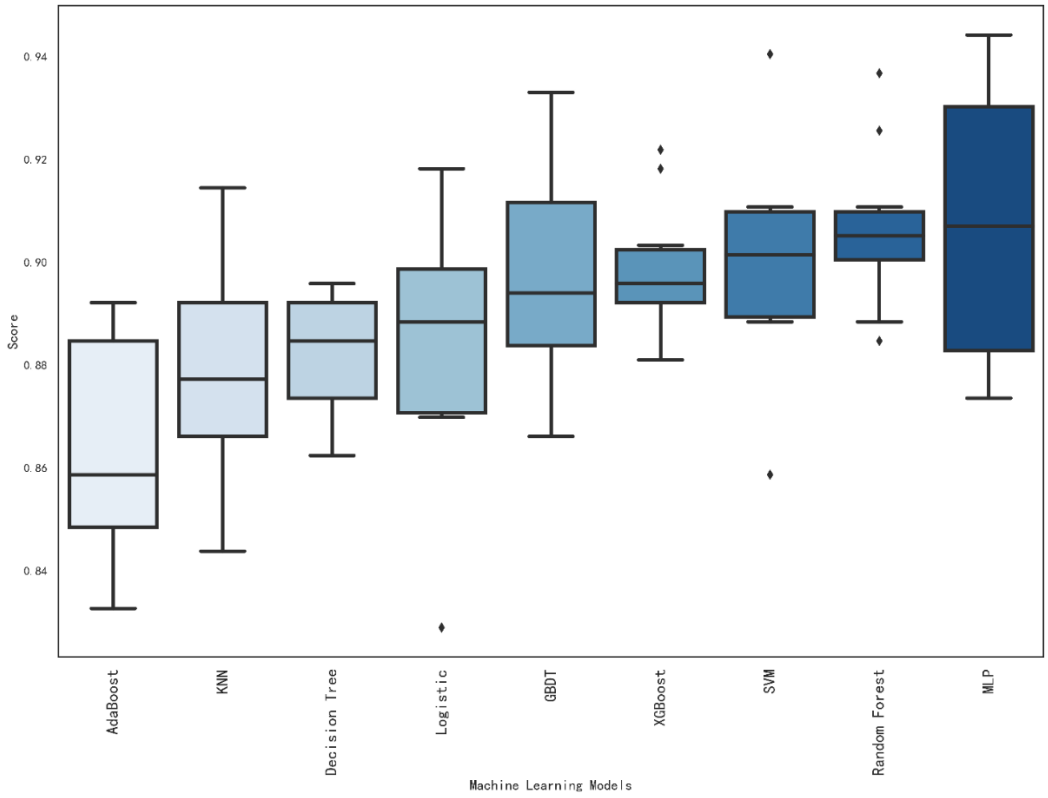

图 5 9 种机器学习模型 10 折交叉验证正确率的箱线图

从上图可以发现，交叉验证结果中平均（中位数）预测正确率低于 0.90 的模型为：AdaBoost、KNN、决策树，平均（中位数）预测正确率最高的两个模型是：MLP 和随机森林，逻辑回归的平均（中位数）预测正确率低于 GBDT，高于决策树，但是除 AdaBoost 和 KNN 外的其他模型的平均（中位数）预测正确率之间差异不大。逻辑回归、XGBoost、SVM、随机森林在 10 折交叉验证的结果中存在离群值。从离散情况来看，AdaBoost、GBDT、MLP 模型的离散程度都较大，XGBoost 和随机森林的离散程度较小，说明这几个模型的预测效果并不稳定。从模型的预测精度、离群点、稳定性角度考虑，较为理想的模型是决策树、随机森林、逻辑回归、XGBoost 模型。

3.5. **变量重要性**

根据分析由上文得到的理想模型为决策树、随机森林、逻辑回归 XGBoost 模型。通过这 4 个模型计算变量的重要性（逻辑回归模型通过系数值衡量），进一步便于观察不同模型之间得到的变量重要性的差异，绘制直方图见图 6 所示。

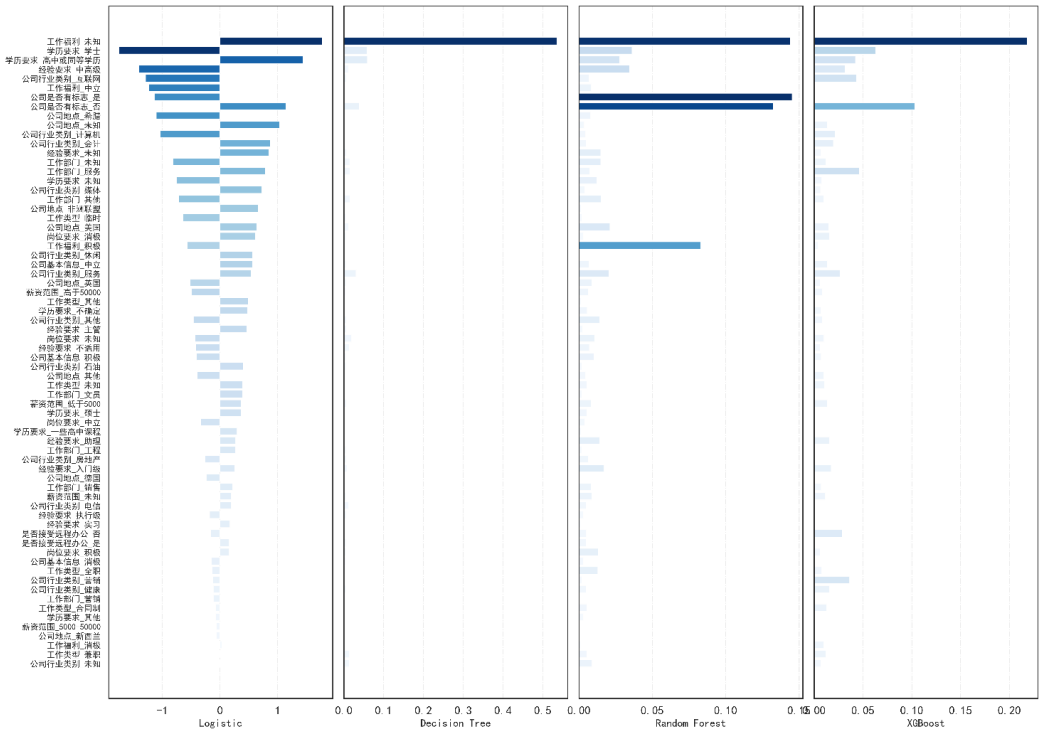

图 6 不同机器学习模型计算的变量重要性分布图

从图中可以发现，逻辑回归模型的系数值分布较为均衡，走势平缓，将“工作福利”中的“未知”水平、“学历要求”、等变量系数值较大。决策树模型中对于变量重要性的计算结果偏斜分布较为极端，“工作福利”中的“未知”水平明显高于其他所有变量，且其他变量的重要性程度几乎为 0。随机森林模型中关于变量的重要性结果分布也比较偏斜，“工作福利”中的“未知”和“积极”水平、“公司是否有标志”明显高于其他所有变量，其他变量重要性较低但不为 0。XGBoost 模型把“工作福利”中的“未知”水平和“公司是否有标志”中的“否”水平作为明显的重要变量，其他变量重要性较低但也不为 0。观察不同模型计算的变量重要性可以得到，“工作福利”中的“未知”水平在 4 个模型中均是最重要的变量，“公司是否有标志”在随机森林和 XGBoost 中也作为了较重要的变量。

综合分析可以得出“工作福利”、“公司是否有标志”变量对于虚假信息的识别效果中具有明显作用。

第4章 总结

随着互联网的发展，招聘网站的虚假信息识别的研究目前已逐渐备许多研究者关注，将虚假招聘信息识别问题抽象分析，其识别过程实际上是利用算法将待识别的信息进行真假分类，进行达到识别的目的。总结本文分析内容，对于网络招聘信息的识别结论如下：

1. 逻辑回归模型与其他机器学习模型的预测效果各有优劣。逻辑回归模型在虚假招聘信息识别的预测精度上和其他较为复杂机器学习模型差异相差不大，但明显优于KNN、决策树等逻辑较为简单的算法模型，且逻辑回归模型在模型的可解释性方面却是明显强于其他 8 种机器学习模型。

2. 网络在线招聘信息指标对于该信息是否为虚假信息的识别有一定的预测能力。具体而言，“工作福利”、“公司是否有标志”两个指标变量，尤其是“工作福利”中的“未知”水平，对于识别虚假信息非常重要。

求职者对于网络招聘信息，可以根据本报告得到的“工作福利”、“公司是否有标志”两个指标变量格外关注。网络信息监管平台也可以利用本分析报告得到的模型，结果实际情况指标对在线招聘信息的虚假识别进行合理的预测，进而规范网络环境。虚假信息识别问题是一个较为复杂的热点问题，在数据采集、数据清洗、识别算法等许多方面都存在新的思虑可以尝试，结合本文的实际，可以进一步考虑对自变量的筛选问题和融合多种模型进行识别问题进行优化。

参考文献

[1] 刘文燕 and 马金雪, 编造、故意传播虚假信息罪适用问题及对策研究. 哈尔滨学院学报, 2021. **42**(05): p. 48-50.

[2] 梦非 and 朱庆华, 社交网络信息传播中意见偏差的国外研究进展. 情报理论与实践: p. 1-11.

[3] 王帆, 社交网络信息可信度实时评估的研究与应用. 2017, 电子科技大学.

[4] 滕洁琪, 基于机器学习的新浪微博机器用户识别研究. 2016, 北京理工大学.

[5] 韩昱, 轻量级分布式虚假信息爬虫的设计与实现. 2019, 辽宁大学.

[6] Liu, X., et al., Real-time Rumor Debunking on Twitter. Conference on Information and Knowledge Management, 2015.

[7] Elsa, H., Transforming practices of diplomacy: the European External Action Service and digital disinformation. International Affairs, 2021. **97**(3).

[8] Michael, Y., S. Walter, and W. Tim, Meme warfare: AI countermeasures to disinformation should focus on popular, not perfect, fakes. Bulletin of the Atomic Scientists, 2021. **77**(3).

[9] John, M., Alan Miller: How the News Literacy Project teaches schoolchildren (and adults) to dismiss and debunk internet disinformation. Bulletin of the Atomic Scientists, 2021. **77**(3).

[10]刘力银, 基于逻辑回归的推荐技术研究及应用. 2013, 电子科技大学.

[11]徐继伟 and 杨云, 集成学习方法:研究综述. 云南大学学报(自然科学版), 2018. **40**(06): p. 1082-1092.

[12]李诒靖, et al., 一种基于Boosting的集成学习算法在不均衡数据中的分类. 系统工程理论与实践, 2016. **36**(01): p. 189-199.

[13]张晓凤, 侯艳, and 李康, 基于AUC统计量的随机森林变量重要性评分的研究. 中国卫生统计, 2016. **33**(03): p. 537-540+542.

附录：Python 代码

```
#!/usr/bin/env python
# coding: utf-8

# #### 包导入、系统配置
import os
import re

import pandas as  pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm

import statsmodels.api as sm
import statsmodels.formula.api as smf
from scipy.stats import chi2_contingency

from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,AdaBoostClassifier,Gradien
tBoostingClassifier
from sklearn.neural_network import MLPClassifier
from xgboost import XGBClassifier

from sklearn.model_selection import ShuffleSplit,cross_val_score,train_test_sp
lit
from sklearn import metrics
import seaborn as sns
import warnings



warnings.filterwarnings("ignore")

inputpath = "../data/"
outputpath = "../output/"


figsize = (20,15)
plt.rcParams['font.sans-serif'] = ['SimHei']  # 用来正常显示中文标签
```

```
plt.rcParams['axes.unicode_minus'] = False  # 用来正常显示负号
plt.rcParams["savefig.dpi"] = 500
plt.rcParams["figure.figsize"] = (20,15)
plt.rcParams["figure.subplot.hspace"] = 0.2
plt.rcParams["figure.subplot.wspace"] = 0.05
plt.rcParams["lines.linewidth"] = 3
plt.rcParams["xtick.labelsize"] = 14
plt.rcParams["ytick.labelsize"] = 10
plt.rcParams["legend.loc"] = "upper left"
# plt.style.use("seaborn-poster")

def num2color(values, cmap="RdBu"):
    """将数值映射为颜色"""
    norm = mpl.colors.Normalize(vmin=np.min(values), vmax=np.max(values))
    cmap = mpl.cm.get_cmap(cmap)
    return [cmap(norm(val)) for val in values]


colormap1 = ["saddlebrown","dodgerblue"]
colormap2 = "Blues"
# plt.rcdefaults()   # 恢复默认配置


inputdata = pd.read_csv(os.path.join(inputpath,"真假招聘信息.csv"))

temp = pd.DataFrame(index=[list(inputdata.columns) * 14])
val = np.random.randint(0,10,14)
temp.loc[inputdata.columns[0],"x"] = val
temp.loc[inputdata.columns[0],:]


# 1.数据格式预处理：字符修正、设置 id 变量
inputdata1 = inputdata.copy()

codingmap = {"unknown":"未知","other":"其他",
             0:"否",1:"是",
             "positive":"积极","negative":"消极", "neutral":"中立",
             "less-5000":"低于 5000", "5000-50000":"5000-50000", "more-50000":"
高于 50000",
             "clerical":"文员", "engineering":"工程","marketing":"市场
", "sales":"销售","service":"服务",
             "contract":"合同制", "full-time":"全职", "part-time":"兼职
","temporary":"临时",
```

```
            "associate":"助理", "director":"主管", "entry level":"入门级
","executive":"执行级","internship":"实习","mid-senior level":"中高级
", "not applicable":"不适用",
            "bachelor's degree":"学士","high school or equivalent":"高中或同等
学历","master's degree":"硕士", "some high school coursework":"一些高中课程
", "unspecified":"不确定",
            "accounting":"会计","computer":"计算机",  "health":"健康
", "internet":"互联网", "leisure":"休闲", "marketing":"营销", "media":"媒体
", "oil":"石油","real":"房地产",  "services":"服务",  "telecommunications":"电信
",
            "au":"非洲联盟", "de":"德国", "gb":"英国", "gr":"希腊", "nz":"新西
兰", "us":"美国",
             }
columnmap = {"fraudulent":"是否为虚假招聘信息",
             "description":"公司基本信息",
             "requirements":"岗位要求",
             "benefits":"工作福利",
             "department":"工作部门",
             "salary":"薪资范围",
             "telecommuting":"是否接受远程办公",
             "logo":"公司是否有标志",
             "type":"工作类型",
             "experience":"经验要求",
             "education":"学历要求",
             "industry":"公司行业类别",
             "country":"公司地点",
             "index":"id"
            }

for x in inputdata.columns:
    if x not in ["telecommuting", "logo", "fraudulent"]:
        inputdata1[x] = inputdata1.loc[:,x].str.lower()
        inputdata1[x] = inputdata1[x].map(lambda  x : x.replace("0other","othe
r"))
        inputdata1[x] = inputdata1[x].map(codingmap)
    if x in ["telecommuting", "logo","fraudulent"]:
        inputdata1[x] = inputdata1[x].map(codingmap)
inputdata2 = inputdata1.reset_index().rename(columns=columnmap)

Adata = inputdata2.copy()


def descri(data,index,columns="是否为虚假招聘信息"):
    """
    描述性统计结果
```

```
    单变量卡方检验
    """
    tempdata1 = pd.crosstab(index=data[index],columns=data[columns])
    tempdata1.loc[:,"Variable"]  = index
    # 计算频数、频率
    tempdata1["prob_True"] = np.round(tempdata1["否
"] / data.shape[0] * 100,2).astype(str) + "%"
    tempdata1["prob_False"] = round(tempdata1["是
"] / data.shape[0] * 100,2).astype(str) + "%"
    tempdata1["False: Overall Sample N(%)"] = tempdata1["否
"].astype(str) +"(" + tempdata1["prob_True"] + ")"
    tempdata1["True: Overall Sample N(%)"] = tempdata1["是
"].astype(str) +"(" + tempdata1["prob_False"] + ")"
    # 卡方检验
    chiq_test = chi2_contingency(tempdata1.loc[:,["否","是"]])
    tempdata1["Chiq"] = "{:.3f}".format(chiq_test[0])
    # 显著性检验
    if chiq_test[1] < 0.001:
        tempdata1["P Value"] = "<0.001(***)"
    elif chiq_test[1] <0.01:
        tempdata1["P Value"] = "{:.3f}(**)".format(chiq_test[1])
    elif chiq_test[1] <0.05:
        tempdata1["P Value"] = "{:.3f}(*)".format(chiq_test[1])

    tempdata2 = tempdata1.reset_index().rename(columns={index:"Level"})

    return tempdata2

# 对所有变量进行计算（描述性统计）
flag = 0
for col in Adata.columns:

    if col not in ["是否为虚假招聘信息","id"]:

        tempdata3 = descri(data=Adata, index=col)

        if flag==0:
            tempdata4 = tempdata3
        else:
            tempdata4 = pd.concat([tempdata4,tempdata3])

        flag += 1
```

```
outputdata1 = tempdata4[["Variable", "Level", "False:
Overall Sample N(%)", "True: Overall Sample N(%)", "Chiq", "P Value"]].copy()
outputdata1.to_csv(os.path.join(outputpath,"outputdata1_描述性统计结
果.csv"), encoding="utf_8_sig")


# 可视化描述
fig, ax = plt.subplots(4,3,sharey=True)

for i in range(len(tempdata4["Variable"].unique())):
    # 绘图数据整理
    tempdata5 = tempdata4.loc[tempdata4["Variable"]==tempdata4["Variable"].uni
que()[i],
                              ["Level","否","是
"]].set_index("Level") / Adata.shape[0]
    tempdata5["marge_sum"] = tempdata5.sum(axis=1)
    tempdata5.sort_values("marge_sum", inplace =True)
    width=0.34
    x = np.arange(len(tempdata5.index))

    # 子图绘制
    rects1 = ax[i//3,i%3-1].bar(x-width/2, tempdata5["否"], width, label='否')
    rects2 = ax[i//3,i%3-1].bar(x+width/2, tempdata5["是"], width, label='是')

    ax[i//3,i%3-1].set_xlabel(tempdata4["Variable"].unique()[i])
    ax[i//3,i%3-1].set_xticks(x)
    ax[i//3,i%3-1].set_xticklabels(tempdata5.index,rotation=0,fontsize=10)
    ax[i//3,i%3-1].legend(loc="upper left")
# plt.subplots_adjust(wspace=0.05)
plt.savefig(os.path.join(outputpath,"outputplot1_频率直方
图.png"),bbox_inches = "tight")


# #### 数据集划分
# 逻辑回归：数据集整理
AdataTemp = Adata.copy()
AdataTemp["是否为虚假招聘信息"] = AdataTemp["是否为虚假招聘信息"].map({"是":1,"否
":0})
AdataTemp.to_csv(os.path.join(outputpath,"outputdata_逻辑回归分析数
据.csv"),encoding="utf_8_sig",index=False)

# 机器学习：哑变量编码
X_data = pd.get_dummies( Adata.drop(["id","是否为虚假招聘信息"],axis=1))
Y_data = Adata["是否为虚假招聘信息"].map({"是":1,"否":0})
```

```
# 逻辑回归参数检验：数据集划分
tdata = AdataTemp.iloc[AdataTemp.loc[AdataTemp["是否为虚假招聘信息
"]==1,"id"].index,:]
fdata = AdataTemp.iloc[AdataTemp.loc[AdataTemp["是否为虚假招聘信息
"]==0,"id"].index,:]

tdata2_train = tdata.sample(int(len(tdata) * 0.7),random_state=123)
tdata2_test = pd.concat([tdata,tdata2_train]).drop_duplicates(keep=False)

fdata2_train = fdata.sample(int(len(fdata) * 0.7),random_state=123)
fdata2_test = pd.concat([fdata,fdata2_train]).drop_duplicates(keep=False)

train_Adata = pd.concat([tdata2_train,fdata2_train])
test_Adata = pd.concat([tdata2_test,fdata2_test])

# 机器学习：数据集划分
X_train = X_data.iloc[train_Adata.index,:]
X_test = X_data.iloc[test_Adata.index,:]
Y_train = Y_data[train_Adata.index]
Y_test = Y_data[test_Adata.index]


Adata2 = train_Adata.copy()


# 模型字典
models = {
        "AdaBoost":AdaBoostClassifier(),
        "KNN":KNeighborsClassifier(),
        "Decision Tree":DecisionTreeClassifier(),
        "Logistic":LogisticRegression(),
        "GBDT":GradientBoostingClassifier(),
        "XGBoost":XGBClassifier(),
        "SVM":SVC(probability=True),
        "Random Forest":RandomForestClassifier(),
        "MLP":MLPClassifier()
        }


# #### 逻辑回归参数估计
# Logistic 模型拟合
model = sm.GLM.from_formula("是否为虚假招聘信
息 ~ {}".format("+".join(list(Adata2.columns[1:-1]))),
                            family = sm.families.Binomial(),
                            data=Adata2)
```

```
result = model.fit()

# 结果整理
res_logist = np.round(pd.concat([pd.DataFrame(result.params),
                                 result.bse,
                                 result.conf_int(),
                                 result.tvalues,
                                 result.pvalues],axis=1),3).reset_index()

res_logist.columns = ["Var.Level","Estimate","Std.Err","95%Lower","95%Upper","
Z-Value","p-Value"]
res_logist["Variables"] = res_logist["Var.Level"].str.replace("]","").str.repl
ace("[","").str.split("T.").str[0]
res_logist["Level"] = res_logist["Var.Level"].str.replace("]","").str.replace(
"[","").str.split("T.").str[1]

# 对所有变量进行计算 Odds/LogOdds
def deal_odds(data, index, column="是否为虚假招聘信息"):

    c = pd.crosstab(data[index], data[column])
    c = c.apply(lambda x: x/x.sum(), axis=1)
    c["Odds"] = c.loc[:, 1] / c.loc[:, 0]
    c["LogOdds"] = np.log(c["Odds"])
    c["Variables"] = index

    return c.reset_index().rename(columns={index:"Level"})

flag = 0
for col in Adata2.columns:

    if col not in ["是否为虚假招聘信息","id"]:

        tempodds1 = deal_odds(Adata2,col)

        if flag==0:
            tempodds2 = tempodds1
        else:
            tempodds2 = pd.concat([tempodds2,tempodds1])

        flag += 1

res_odds = np.round(tempodds2,3)

# 模型估计结果和 Odds 结果合并
```

```
res_logist_merge = res_logist.merge(res_odds.loc[:,["Variables","Level","Odds"
,"LogOdds"]], how="outer",on=["Variables","Level"])

res_logist_merge["Odds"] = res_logist_merge["Odds"].replace({np.inf:">999.999"
, -np.inf:"<-999.999",0:"<0.001"})
res_logist_merge["LogOdds"] = res_logist_merge["LogOdds"].replace({np.inf:">99
9.999", -np.inf:"<-999.999",0:"<0.001"})

res_logist_merge.sort_values(["Variables","Level"],inplace=True)
res_logist_out = res_logist_merge[["Var.Level","Variables","Level","Estimate",
"Std.Err","95%Lower","95%Upper",
                                   "Z-Value","p-Value","Odds","LogOdds"]]
res_logist_out.rename(columns={
    "Var.Level":"变量-水平",
    "Variables":"变量名",
    "Level":"水平",
    "Estimate":"回归系数",
    "Std.Err":"标准误",
    "95%Lower":"0.95 置信区间下限",
    "95%Upper":"0.95 置信区间上限",
    "Z-Value":"Z-值",
    "p-Value":"P-值",
    "Odds":"几率",
    "LogOdds":"对数几率"
},inplace=True)
res_logist_out.to_csv(os.path.join(outputpath,"outputdata_逻辑回归参数估计结
果.csv"),encoding="utf_8_sig",index=False)


# #### 逻辑回归：结果评价
logit_clf = models["Logistic"].fit(X_train,Y_train)

## 在训练集和测试集上分布利用训练好的模型进行预测
train_predict = logit_clf.predict(X_train)
test_predict = logit_clf.predict(X_test)



## 利用 accuracy（准确度）【预测正确的样本数目占总预测样本数目的比例】评估模型效果
print('The accuracy of the Logistic Regression on train_predict is:',metrics.a
ccuracy_score(Y_train,train_predict))
print('The accuracy of the Logistic Regression on test_predict is:',metrics.ac
curacy_score(Y_test,test_predict))

# 利用热力图对于结果进行可视化
```

```
confusion_matrix_result = metrics.confusion_matrix(test_predict,Y_test)
plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix_result, annot=True, cmap=colormap2,fmt='.50g')
plt.xlabel('预测值')
plt.ylabel('真实值')
plt.savefig(os.path.join(outputpath,"outputplot_逻辑回归混淆矩阵
图.png"),dpi=500,bbox_inches = "tight")
plt.show()


def acu_curve(y,prob):
    fpr,tpr,threshold = metrics.roc_curve(y,prob) ###计算真正率和假正率
    roc_auc = metrics.auc(fpr,tpr) ###计算 auc 的值

    plt.figure()
    lw = 3
    plt.figure(figsize=(10,8))
    plt.plot(fpr, tpr, color='cornflowerblue',
             lw=lw, label='ROC curve (AUC = %0.3f)' % roc_auc) ###假正率为横坐
标，真正率为纵坐标做曲线
    plt.plot([0, 1], [0, 1], color='k', lw=1, linestyle='--',alpha=0.5)
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.legend(loc="lower right")


###通过 decision_function()计算得到的 y_score 的值，用在 roc_curve()函数中
y_score = logit_clf.fit(X_train, Y_train).decision_function(X_test)
acu_curve(Y_test,y_score)
plt.savefig(os.path.join(outputpath,"outputplot_逻辑回归 ROC 曲线
图.png"),dpi=500,bbox_inches = "tight")
plt.show()


# ##### 变量重要性
# 变量重要性排序
coef_lr = pd.DataFrame({'var' : X_test.columns,
                        'coef' : logit_clf.coef_.flatten()
                        })
coef_lr_sort0 = coef_lr.sort_values(["coef"],ascending=True)
index_sort =  np.abs(coef_lr['coef']).sort_values().index
coef_lr_sort = coef_lr.loc[index_sort,:]
```

```
# 水平柱形图绘图

fig,ax=plt.subplots(figsize=(20,15))
x, y = coef_lr_sort['var'], coef_lr_sort['coef']
map_vir = cm.get_cmap(name=colormap2)
norm_y = plt.Normalize(min(coef_lr_sort0["coef"]),max(coef_lr_sort0["coef"]))(
coef_lr_sort0["coef"])
color = map_vir(norm_y)
rects = plt.barh(x, y, color=color)
plt.grid(linestyle="-.", axis='x', alpha=0.4)
plt.yticks(fontsize=10)
plt.xticks(fontsize=13)
plt.savefig(os.path.join(outputpath,"outputplot_逻辑回归变量重要性柱状图
2.png"),dpi=500,bbox_inches = "tight")
plt.show()


# #### 机器学习：模型交叉验证对比
# ##### 混淆矩阵可视化
fig, ax = plt.subplots(3,3,figsize=(20,15),sharey=True,sharex=True)

flag = 0
for name, model in models.items():

    clf = models[name].fit(X_train,Y_train)
    train_predict = clf.predict(X_train)
    test_predict = clf.predict(X_test)

    # 利用热力图对于结果进行可视化


    confusion_matrix_result = metrics.confusion_matrix(test_predict,Y_test)
    sns.heatmap(confusion_matrix_result, annot=True, cmap=colormap2,fmt='.50g'
,ax=ax[flag//3,flag%3])

    ax[flag//3,flag%3].set_xlabel(name)
    ax[flag//3,flag%3].set_ylabel("真实值")
    flag +=1
# plt.subplots_adjust(wspace=0.1,hspace=0.1)
plt.savefig(os.path.join(outputpath,"outputplot_所有模型混淆矩阵
图.png"),dpi=500,bbox_inches = "tight")
plt.show()


# ##### ROC 曲线可视化
```

```
fig, ax = plt.subplots(3,3,figsize=(20,15),sharey=True,sharex=True)

flag = 0
for name, model in models.items():
#     if name == "KNN" or name == "Decision Tree" or name == "XGBoost":
#         continue
    clf = models[name].fit(X_train,Y_train)
    y_score = clf.predict_proba(X_test)

    fpr,tpr,threshold = metrics.roc_curve(Y_test,y_score[:,1],pos_label=1)
    roc_auc = metrics.auc(fpr,tpr)

    # 绘制 ROC 曲线
    ax[flag//3,flag%3].plot(fpr, tpr,lw=2,color="black", label='{}(area = {:.3
f})'.format(name, roc_auc))

    ax[flag//3,flag%3].set_xlabel(name)


    ax[flag//3,flag%3].plot([0, 1], [0, 1], color='k', lw=1, linestyle='--
',alpha=0.5)
    ax[flag//3,flag%3].set_xlim([0.0, 1.0])
    ax[flag//3,flag%3].set_ylim([0.0, 1.05])
    ax[flag//3,flag%3].set_xlabel('False Positive Rate')
    ax[flag//3,flag%3].set_ylabel('True Positive Rate')
    ax[flag//3,flag%3].legend(loc="lower right")
    flag +=1


# plt.subplots_adjust(wspace=0.1,hspace=0.1)
plt.savefig(os.path.join(outputpath,"outputplot_所有模型 ROC 曲线图
1.png"),dpi=500,bbox_inches = "tight")
plt.show()


plt.figure(figsize=(15,10))
for name, model in models.items():
#     if name == "KNN" or name == "Decision Tree" or name == "XGBoost":
#         continue
    clf = models[name].fit(X_train,Y_train)
    y_score = clf.predict_proba(X_test)

    fpr,tpr,threshold = metrics.roc_curve(Y_test,y_score[:,1],pos_label=1)
    roc_auc = metrics.auc(fpr,tpr)
```

```
    # 绘制 ROC 曲线
    plt.plot(fpr, tpr,lw=2, label='{}(AUC = {:.3f})'.format(name, roc_auc))

plt.plot([0, 1], [0, 1], color="k", lw=1, linestyle='--',alpha=0.5)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc="lower right")
# plt.subplots_adjust(wspace=0.1,hspace=0.1)
plt.savefig(os.path.join(outputpath,"outputplot_所有模型 ROC 曲线图
2.png"),dpi=500,bbox_inches = "tight")
plt.show()


# ##### 交叉验证
# 建立交叉模型
scores_dict = {}
for name,model in models.items():

    cv = ShuffleSplit(n_splits=10, test_size=.3, random_state=0)
    scores_CV = cross_val_score(model, X_data,Y_data, cv=cv)
    print(name,":",scores_CV)
    scores_dict[name] = scores_CV  # 保存交叉验证评分结果
df_scoresCV = pd.DataFrame(scores_dict).round(3)
df_scoresCV.to_csv(os.path.join(outputpath,"outputdata2_交叉验证得分结
果.csv"),index=True,header=True)


# 可视化绘制箱线图
score_plotdata = pd.DataFrame(scores_dict).melt()

plt.figure(figsize=(15,10))
# sns.set(style="white")
color_p = sns.color_palette(colormap2,9)
sns.boxplot(x="variable", y="value",data=score_plotdata,palette=color_p)
plt.xticks(rotation=90)
plt.xlabel("Machine Learning Models")
plt.ylabel("Score")

plt.savefig(os.path.join(outputpath,"outputplot2_交叉验证得分结果箱线
图.png"),dpi=500, bbox_inches = "tight")
plt.show()
```

```
# #### 变量重要性
fig, ax = plt.subplots(1,4,figsize=(20,15),sharey=True)

flag = 0

models2 = {
        "Logistic":LogisticRegression(),
        "Decision Tree":DecisionTreeClassifier(),
        "Random Forest":RandomForestClassifier(),
        "XGBoost":XGBClassifier(),
        }

for name, model in models2.items():
    if name == "Logistic" or name == "Decision Tree" or name == "Random Forest
" or name == "XGBoost" :

        clf = models2[name].fit(X_train,Y_train)

        if name == "Logistic":
            importance = clf.coef_
        else:
            importance = clf.feature_importances_
        # 变量重要性排序
        coef = pd.DataFrame({'var' : X_test.columns,
                             'coef' : importance.flatten()
                             })

        coef_sort0 = coef["coef"].sort_values(ascending=False)
        if flag == 0:
            index_sort =  np.abs(coef['coef']).sort_values().index
        coef_sort = coef.loc[index_sort,:]


        # 水平柱形图绘图
        x, y = coef_sort['var'], coef_sort['coef']


        map_vir = cm.get_cmap(name=colormap2)
        if flag == 0:
            norm_y = plt.Normalize(min(abs(index_sort)),max(abs(index_sort)))(
abs(index_sort))
        norm_y = plt.Normalize(min(abs(y)),max(abs(y)))(abs(y))
        color = map_vir(norm_y)

        rects = ax[flag%4].barh(x, y, color=color)
```

```
        ax[flag%4].grid(linestyle="-.", axis='x', alpha=0.4)
        ax[flag%4].set_xlabel(name,fontsize=13)
        flag += 1
plt.subplots_adjust(wspace=0.05,hspace=0.1)
plt.savefig(os.path.join(outputpath,"outputplot 机器学模型重要性柱状
图.png"),dpi=500,bbox_inches = "tight")
plt.show()


dt_clf = DecisionTreeClassifier().fit(X_train,Y_train)

# 变量重要性排序
coef_dt = pd.DataFrame({'var' : X_test.columns,
                        'coef' : dt_clf.feature_importances_.flatten()
                        })
coef_dt_sort0 = coef_dt.sort_values(["coef"],ascending=True)
index_sort =  np.abs(coef_dt['coef']).sort_values().index
coef_dt_sort = coef_dt.loc[index_sort,:]

# 水平柱形图绘图
fig,ax=plt.subplots(figsize=(20,15))
x, y = coef_dt_sort['var'], coef_dt_sort['coef']
map_vir = cm.get_cmap(name=colormap2)
color = map_vir(coef_dt_sort0["coef"])
rects = plt.barh(x, y, color=color)
plt.grid(linestyle="-.", axis='x', alpha=0.4)
plt.yticks(fontsize=10)
plt.xticks(fontsize=13)
plt.savefig(os.path.join(outputpath,"outputplot 决策树变量重要性柱状图
2.png"),dpi=500,bbox_inches = "tight")
plt.show()


rf_clf = RandomForestClassifier().fit(X_train,Y_train)
# importance = rf_clf.feature_importances_
# # for i,v in enumerate(importance):
# #     print('Feature: %0d, Score: %.5f' % (i,v))
# # plot feature importance
# plt.bar([x for x in range(len(importance))], importance)
# plt.show()


# 变量重要性排序
coef_rf = pd.DataFrame({'var' : X_test.columns,
                        'coef' : rf_clf.feature_importances_.flatten()
```

```
                        })
coef_rf_sort0 = coef_rf.sort_values(["coef"],ascending=True)
index_sort =  np.abs(coef_rf['coef']).sort_values().index
coef_rf_sort = coef_rf.loc[index_sort,:]

# 水平柱形图绘图

fig,ax=plt.subplots(figsize=(20,15))
x, y = coef_rf_sort['var'], coef_rf_sort['coef']
map_vir = cm.get_cmap(name=colormap2)
color = map_vir(coef_rf_sort0["coef"])
rects = plt.barh(x, y, color=color)
plt.grid(linestyle="-.", axis='x', alpha=0.4)
plt.yticks(fontsize=10)
plt.xticks(fontsize=13)
plt.savefig(os.path.join(outputpath,"outputplot 随机森林变量重要性柱状图
2.png"),dpi=500,bbox_inches = "tight")
plt.show()


xgb_clf = XGBClassifier().fit(X_train,Y_train)
# importance = xgb_clf.feature_importances_
# # for i,v in enumerate(importance):
# #     print('Feature: %0d, Score: %.5f' % (i,v))
# # plot feature importance
# plt.bar([x for x in range(len(importance))], importance)
# plt.show()


# 变量重要性排序
coef_xgb = pd.DataFrame({'var' : X_test.columns,
                        'coef' : xgb_clf.feature_importances_.flatten()
                        })
coef_xgb_sort0 = coef_xgb.sort_values(["coef"],ascending=True)
index_sort =  np.abs(coef_xgb['coef']).sort_values().index
coef_xgb_sort = coef_xgb.loc[index_sort,:]

# 水平柱形图绘图

fig,ax=plt.subplots(figsize=(20,15))
x, y = coef_xgb_sort['var'], coef_xgb_sort['coef']
map_vir = cm.get_cmap(name=colormap2)
color = map_vir(coef_xgb_sort0["coef"])
rects = plt.barh(x, y, color=color)
plt.grid(linestyle="-.", axis='x', alpha=0.4)
```

```
plt.yticks(fontsize=10)
plt.xticks(fontsize=13)
plt.savefig(os.path.join(outputpath,"outputplotXGBoost 变量重要性柱状图
2.png"),dpi=500,bbox_inches = "tight")
plt.show()
```