数据分析笔试题目

在消费信贷业务中，越来越多有借贷需求的个人借款者通过线上渠道获取贷款。有线上消费借贷业务的公司或者机构，通过获取到的个人借款者授权的数据（如手机数据、征信数据、第三方风控数据等），构建智能风控模型，制定信用风险政策，对个人借款者进行信用评估，根据个人借款者的逾期风险水平高低，并结合业务目标，决定是否为个人借款者发放贷款。

个人借款者逾期风险的高低，受到各方面因素影响。由于数据可获得性的限制，信用风险决策往往在有限的条件下进行。本次笔试 Excel 数据文件中，是一张表名为 dataset 的数据表及对应的数据字典。本数据集为一家线上消费信贷公司，在个人借款者首次提交借款申请时，获取到的个人基本信息、手机 APP 信息、征信报告信息，以及个人借款者在该公司的借款申请信息。

请利用本数据集，完成以下问题：

1. 请对数据集进行数据清洗，说明数据集的异常情况，统计最终数据集的记录数。

由于该项数据分析的最终目的是结合业务目标，决定是否为个人借款者发放贷款，所以数据清洗的目的主要有以下几点：

（1）处理缺失值和重复值（2）确保数据类型是适合用来后面的数据分析的

（3）识别和处理无效值或异常值（4）对数据的格式进行标准化

该数据集的异常情况有以下几点：

（1）数据集中缺失值太多，disbr_dt 和 is_loan 两列都有 28049 行空值。

（2）数据集中有几列代表的是同样的意思，导致数据集冗余

（3）数据集中变量类型过多，有些变量对于实际的分析目的来说没有太大的价值。

（4）数据集中含有-99 的行太多，这对于实际的分析目的有很大的影响。

通过数据的清洗，最终能用于数据分析的有价值数据只有 5830 行。

2. 请完成以下统计：

* 注：请将按要求将结果列在每道题之后，数据结果在 Excel 文件中展示，每道题 1 个 Excel Sheet，Excel 文件命名为 result2。

2.1 用一段 SQL 统计不同婚姻状态下，申请贷款的用户数量、通过审批的用户数量、放款的用户数量，以及通过审批的数量占申请贷款数量的比例、放款数量占通过审批数量的比例。

作答：SQL 如下：

```
select
marriage_v,
count(customer_id) as total_applicants,
sum(is_pass) as passed_applicants,
sum(is_pass)/count(customer_id) as passed_rate
from dataset
group by marriage_v;

select
marriage_v,
count(customer_id) as total_applicants,
sum(is_loan) as loaned_applicants,
sum(is_loan)/count(customer_id) as loaned_rate
from dataset
group by marriage_v;
```

2.2 请用一段 SQL 查询出每个职业类型下，用户年龄从高到低前 5 名的 customer_id，及该 id 对应的年龄、月收入。

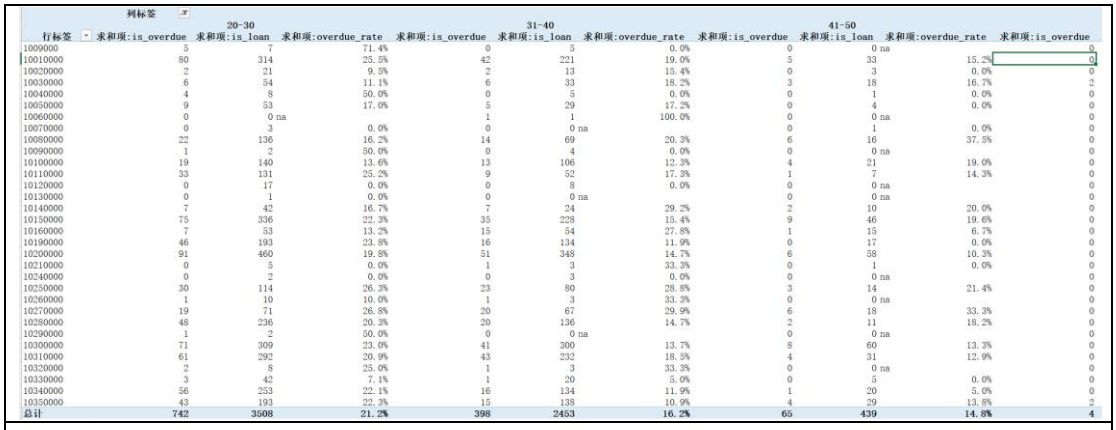
作答：SQL 如下：

```
select
occupation_v,customer_id as users,age,mon_salary_v
from
(select occupation_v,customer_id,age,mon_salary_v,
```

```
row_number() over (partition by occupation_v order by age DESC) as num from dataset
where age between 18 and 65)
as ranked
where
num <= 5
order by
occupation_v, age DESC;
```

2.3 请用 Excel 透视表，按周统计不同年龄段的用户逾期的比例（百分比保留一位小数）。

作答：Excel 透视表截图如下：



列标签	20-30			31-40			41-50			
行标签	求和项:is_overdue	求和项:is_loan	求和项:overdue_rate	求和项:is_overdue	求和项:is_loan	求和项:overdue_rate	求和项:is_overdue	求和项:is_loan	求和项:overdue_rate	求和项:is_overdue
1009000	5	7	71.4%	0	5	0.0%	0	0 na		0
10010000	80	314	25.5%	42	221	19.0%	5	33	15.2%	0
10020000	2	21	9.5%	2	13	15.4%	0	3	0.0%	0
10030000	6	54	11.1%	6	33	18.2%	3	18	16.7%	2
10040000	4	8	50.0%	0	5	0.0%	0	1	0.0%	0
10050000	9	53	17.0%	5	29	17.2%	0	4	0.0%	0
10060000	0	0 na		1	1	100.0%	0	0 na		0
10070000	0	3	0.0%	0	0 na		0	1	0.0%	0
10080000	22	136	16.2%	14	69	20.3%	6	16	37.5%	0
10090000	1	2	50.0%	0	4	0.0%	0	0 na		0
10100000	19	140	13.6%	13	106	12.3%	4	21	19.0%	0
10110000	33	131	25.2%	9	52	17.3%	1	7	14.3%	0
10120000	0	17	0.0%	0	8	0.0%	0	0 na		0
10130000	0	1	0.0%	0	0 na		0	0 na		0
10140000	7	42	16.7%	7	24	29.2%	2	10	20.0%	0
10150000	75	336	22.3%	35	228	15.4%	9	46	19.6%	0
10160000	7	53	13.2%	15	54	27.8%	1	15	6.7%	0
10190000	46	193	23.8%	16	134	11.9%	0	17	0.0%	0
10200000	91	460	19.8%	51	348	14.7%	6	58	10.3%	0
10210000	0	5	0.0%	1	3	33.3%	0	1	0.0%	0
10240000	0	2	0.0%	0	3	0.0%	0	0 na		0
10250000	30	114	26.3%	23	80	28.8%	3	14	21.4%	0
10260000	1	10	10.0%	1	3	33.3%	0	0 na		0
10270000	19	71	26.8%	20	67	29.9%	6	18	33.3%	0
10280000	48	236	20.3%	20	136	14.7%	2	11	18.2%	0
10290000	1	2	50.0%	0	0 na		0	0 na		0
10300000	71	309	23.0%	41	300	13.7%	8	60	13.3%	0
10310000	61	292	20.9%	43	232	18.5%	4	31	12.9%	0
10320000	2	8	25.0%	1	3	33.3%	0	0 na		0
10330000	3	42	7.1%	1	20	5.0%	0	5	0.0%	0
10340000	56	253	22.1%	16	134	11.9%	1	20	5.0%	0
10350000	43	193	22.3%	15	138	10.9%	4	29	13.8%	2
总计	742	3508	21.2%	398	2453	16.2%	65	439	14.8%	4

图 2-1 不同州不同年龄段用户逾期比例

由于图片大小问题，word 中只展示了 20-50 岁年龄区间的数据，完整数据呈现于 excel 中。

3. 请利用 Excel/SQL/Python 等数据分析软件，任选主题，对该数据集进行分析，提交一份数据分析报告。

作答：

* 注：请将分析过程中产生的代码附上。最终的答卷和代码一起打包，并以自己的名字_学校_年级命名，回复邮件。

数据分析软件选择：Python 的 jupyter notebook

数据分析目的及主题：通过对数据的清洗，筛选出有价值数据，然后通过对数据的描述性统计分析发现数据之间的关系，最后从 K 近邻模型、GBDT 模

型、XGBboost 模型中选择准确率最高，最能用来对该数据集进行分类的机器学习模型，期望基于该模型建立一个简易的公司的智能风控模型。

3.1 数据分析准备工作

要想进行数据分析，首先需要知道该数据集的大小，变量以及该数据集有哪些列。部分代码如下：

```
%cd D:\桌面文件\实习\量子数能-数据分析实习笔试
import pandas as pd
import numpy as np
df = pd.read_excel('量子数能-数据分析实习_dataset.xlsx',engine="openpyxl")
df.columns          # 了解该数据集有哪些列
df.shape            # 观察该数据集的大小
df.sample(10)       # 初步了解一下数据
df.isnull().sum()   # 查看缺失值情况
```

3.2 数据清洗和预处理

3.2.1 初步删除缺失值和对最终分析结果影响不大的列

前面通过观察该数据集的缺失值发现，大部分 is_loan 列的缺失值都是没有通过公司的借贷申请的，由于我们的目的是想通过变量来预测借贷者是否会逾期，所以从一开始就没有通过公司的借贷申请的数据将会被删除，而这正好使 is_loan 列的缺失值。同时根据我们的分析目的，我们也可以初步删除那些对是否逾期有影响的列。部分代码如下：

```
loaned=df
loaned.dropna(subset=['is_overdue','age','occupation_v','cd_score'],inplace=True)
                    # 初步删除缺失值
loaned.drop(
columns=['apply_dt','disbr_dt','live_county','work_county','system_version',
'cont1_rel_v','cont2_rel_v','ap_al','ap_pdal','ap_longal','ap_dd_1lo','ap_dd_lalo','ap_dd_1lo_1ap'],inplace=True) # 删除对于分析结果影响不大的列
loaned['ap_loanal']=loaned['ap_loanal'].fillna(loaned['ap_pd30d']+loaned['ap_long90d'])
                    # 根据不同列之间的关系来填补部分的缺失值
loaned.drop_duplicates(inplace=True)
                    # 删除重复值
```

3.2.2 对数据进行格式化

由于该数据集中的工资列是一个工资的区间，这样类型的值不利于后续的数据分析，所以将每一个借贷人的工资转化为他所处的工资区间的中值。同时对于是否逾期这一列，原数据集是使用 1 来代表逾期，0 来代表未逾期，这样在后续的分析中不太直观，所以这里将其转化为逾期或未逾期的字样来表示。部分代码如下：

```
loaned['mon_salary_v'].unique()
                    # 查看工资有哪些区间
salary_mapping = {
    '50,000-100,000': 75000,
    '20,000-30,000': 25000,
    '10,000-15,000': 12500,
    '30,000-50,000': 40000,
    '15,000-20,000': 17500,
    '100,000-200,000': 150000,
    '200,000-500,000': 350000,
    '>=500,000': 500000,
    '<10,000': 10000}
loaned['mon_salary_v']=loaned['mon_salary_v'].map(salary_mapping)
                    # 将区间的中值映射到数据集上进行替换
is_overdue_mapping = {
    1:'逾期',
    0:'未逾期'}
loaned['is_overdue']=loaned['is_overdue'].map(is_overdue_mapping)
                    # 与工资的映射同理
```

3.2.3 进一步简化数据

在观察了描述性分析之后发现有几点问题：

（1）age 列有异常数据

（2）ap_pd24h;ap_pd3d;ap_30d 同时出现会造成数据分析冗余（ap_long3d 等同理）只需要保留 ap_30d，

（3）有几列还存在-99 的数据还没有去除

所以进一步简化数据。部分代码如下：

```
loaned.drop(columns=['customer_id','is_pass','is_loan','cd_enq_tot30d','cd_enq_tot60d','cd_enq_org30d','cd_enq_org60d','cd_max_ovdudys'],inplace=True)
loaned.drop(columns=['ap_pd24h','ap_pd3d','ap_long3d','ap_long30d'],inplace=True)
                    # 进一步删除与分析目标无关的列
loaned=loaned[(loaned['age'] >=18) & (loaned['age']<=65)]
                    # 只选取年龄在 18-65 区间的数据
loaned=loaned[~(loaned['cd_score'] == -99)]
loaned=loaned[~(loaned['ap_pd30d'] == -99)]
                    # 删除那些含有-99 的行
```

3.3 数据的探索性分析

做数据的探索性分析目的是了解数据背后的特征，如数据的集中趋势、离散趋势、数据形状和变量间的关系等。在进行数据分析时，需要知道每个变量的基本统计值，如均值、中位数、众数等，只有了解了所需处理的数据特征，才能做到“心中有数”。

部分代码如下：

```
loaned.describe()     # 对于数据总体的一个探索性分析
loaned.describe(include=['object'])
                      # 对于 object 类型的探索性分析
```

3.3.1 对于连续性变量的分析

```
import matplotlib.pyplot as plt
plt.style.use('ggplot')
fig, axes = plt.subplots(3, 1)
loaned.age[loaned.is_overdue == '逾期'].plot(kind = 'kde', label = 'overdued', ax = axes[0], legend = True, linestyle = '-')
loaned.age[loaned.is_overdue == '未逾期'].plot(kind = 'kde', label = 'dont overdued', ax = axes[0], legend = True, linestyle = '--')
                      # 绘制是否会逾期的年龄核密度图

loaned.mon_salary_v[loaned.is_overdue == '逾期'].plot(kind = 'kde', label = 'overdued', ax = axes[1], legend = True, linestyle = '-')
loaned.mon_salary_v[loaned.is_overdue == '未逾期'].plot(kind = 'kde', label = 'dont overdued', ax = axes[1], legend = True, linestyle = '--')
                      # 绘制是否会逾期的月收入核密度图

loaned.cd_score[loaned.is_overdue == '逾期'].plot(kind = 'kde', label = 'overdued', ax = axes[2], legend = True, linestyle = '-')
loaned.cd_score[loaned.is_overdue == '未逾期'].plot(kind = 'kde', label = 'dont overdued', ax = axes[2], legend = True, linestyle = '--')
                      # 绘制是否会逾期的征信核密度图
plt.show()
```

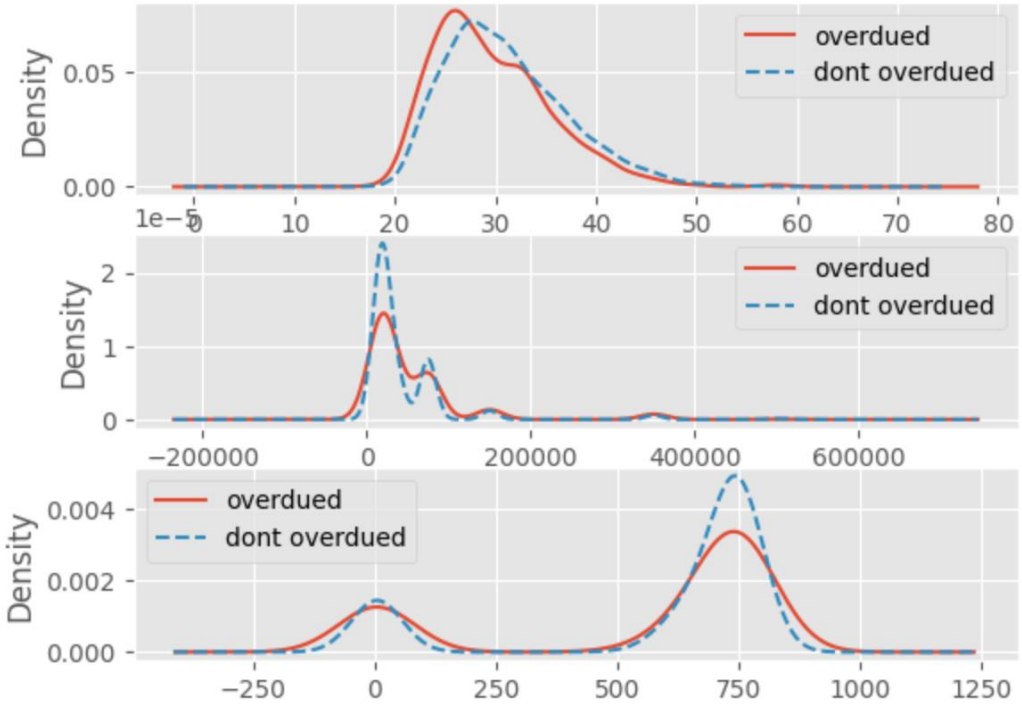


图 3-1 是否会逾期的年龄、月收入和征信核密度图

3.3.2 对于离散型变量的分析

```
occupation = pd.DataFrame(
loaned.groupby(by ['occupation_v','is_overdue']).aggregate(np.size).loc[:,'age'])
                        # 不同职业的人的逾期情况
occupation = occupation.reset_index()
                        # 重设索引
```

	occupation_v	is_overdue	age
0	Armed forces occupations	未逾期	31
1	Armed forces occupations	逾期	8
2	Clerical support workers	未逾期	286
3	Clerical support workers	逾期	72
4	Craft and related trades workers	未逾期	85
5	Craft and related trades workers	逾期	21
6	Elementary occupations	未逾期	47
7	Elementary occupations	逾期	5
8	Managers	未逾期	1039
9	Managers	逾期	214
10	Plant and machine operators, and assemblers	未逾期	146
11	Plant and machine operators, and assemblers	逾期	31
12	Professional	未逾期	1194
13	Professional	逾期	248
14	Service and sales workers	未逾期	1556
15	Service and sales workers	逾期	357
16	Skilled agricultural, forestry and fishery wor...	未逾期	45
17	Skilled agricultural, forestry and fishery wor...	逾期	15
18	Technicians and associate prefessionals	未逾期	359
19	Technicians and associate prefessionals	逾期	71

图 3-2 不同职业的逾期情况

不同职业的逾期情况可视化柱状图

```
import seaborn as sns
plt.figure(figsize=(9,5))
sns.barplot(y="occupation_v", x="counts", hue = 'is_overdue', data=occupation)
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.show()
```

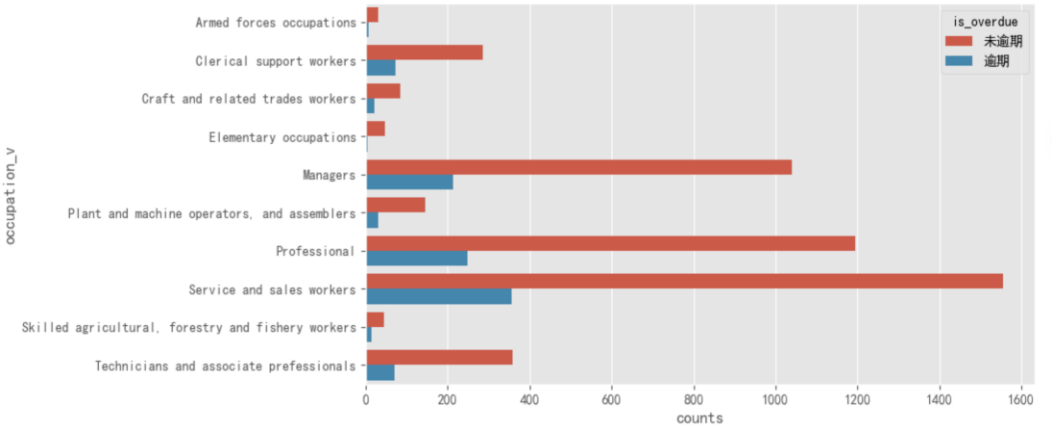


图 3-3 不同职业的逾期情况图

3.4 数据建模

前面对数据也进行了清理，同时也观察了数据的基本状态，现在需要开始选择对数据进行建模，最终建立我们的分类模型了。

3.4.1 对离散型变量进行编码

```
for feature in loaned.columns:
    if loaned[feature].dtype == 'object':
        loaned[feature] = pd.Categorical(loaned[feature]).codes
loaned.head()
```

3.4.2 拆分数据集

将数据拆分为训练集和测试集，通过训练集中的数据来训练模型，通过数据集中的数据来测试模型。

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(loaned.loc[:,'age':'cd_tot_bal'],

loaned['is_overdue'], train_size = 0.75, random_state = 1234)
print('训练数据集共有%d 条观测' %X_train.shape[0])
print('测试数据集共有%d 条观测' %X_test.shape[0])
```

3.4.3 模型构建

在机器学习的模型中有 K 近邻模型、GBDT 模型、XGBoost 模型，这些都可以用到本次的分类中，接下来使用我们刚才处理好的数据，分别使用到这三个模型中，看看哪一个模型的准确率最好，然后选择一个最好的来进一步优化该模型的参数。

（1）默认的 K 近邻模型

```
from sklearn.neighbors import KNeighborsClassifier
kn = KNeighborsClassifier()
kn.fit(X_train, y_train)
print(kn)                              # 构建 k 近邻模型
kn_pred = kn.predict(X_test)
print(pd.crosstab(kn_pred, y_test))# 对模型进行拟合
print('模型在训练集上的准确率%f' %kn.score(X_train,y_train))
print('模型在测试集上的准确率%f' %kn.score(X_test,y_test))
                                       # 查看模型的准确率
from sklearn import metrics
fpr, tpr, _ = metrics.roc_curve(y_test, kn.predict_proba(X_test)[:,1])
plt.plot(fpr, tpr, linestyle = 'solid', color = 'yellow')
plt.stackplot(fpr, tpr, color = 'steelblue')
plt.plot([0,1],[0,1], linestyle = 'dashed', color = 'black')
plt.text(0.6,0.4,'AUC=%.3f' % metrics.auc(fpr,tpr))
plt.show()                             # 使用 ROC 曲线对模型进行评估，其中 AUC 的值越大代表模型越好
```

k 近邻模型的评估结果如下：

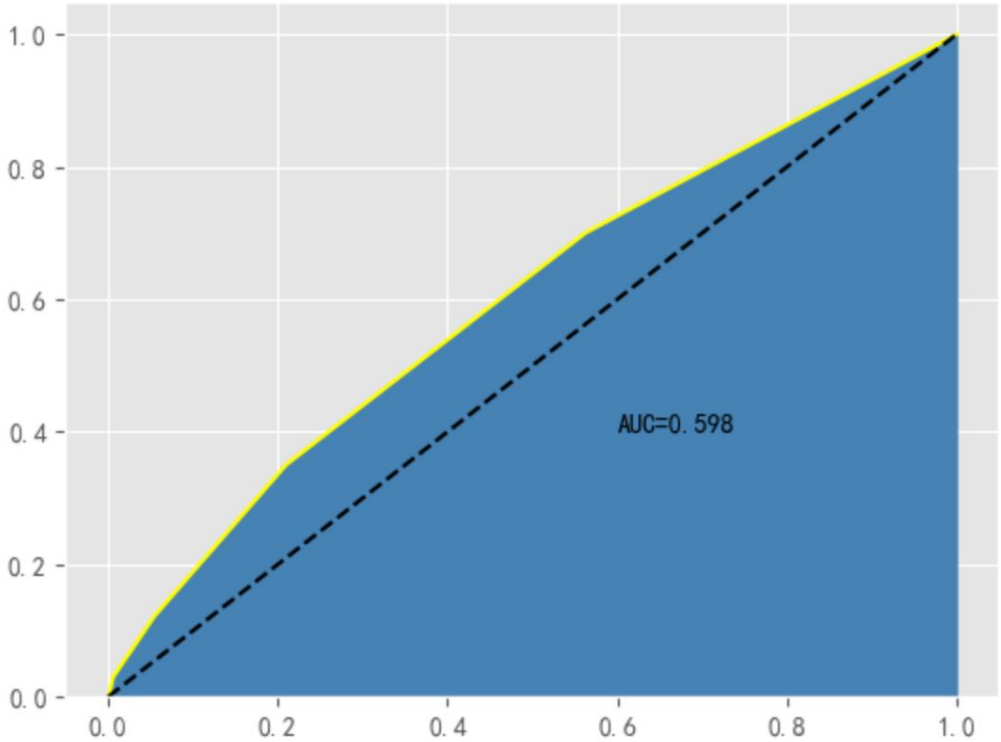


图 3-4 k 近邻模型的评估结果

（2）默认的 GBDT 模型

```
from sklearn.ensemble import GradientBoostingClassifier
gbdt = GradientBoostingClassifier()
gbdt.fit(X_train, y_train)
print(gbdt)                              # GBDT 模型的构建

gbdt_pred = gbdt.predict(X_test)
print(pd.crosstab(gbdt_pred, y_test))

print('模型在训练集上的准确率%f' %gbdt.score(X_train,y_train))
print('模型在测试集上的准确率%f' %gbdt.score(X_test,y_test))
                                         #  查看模型准确率
fpr, tpr, _ = metrics.roc_curve(y_test, gbdt.predict_proba(X_test)[:,1])
plt.plot(fpr, tpr, linestyle = 'solid', color = 'red')
plt.stackplot(fpr, tpr, color = 'steelblue')
plt.plot([0,1],[0,1], linestyle = 'dashed', color = 'black')
plt.text(0.6,0.4,'AUC=%.3f' % metrics.auc(fpr,tpr), fontdict = dict(size = 18))
plt.show()                               # ROC 曲线的绘制
```

GBDT 模型的评估结果如下：

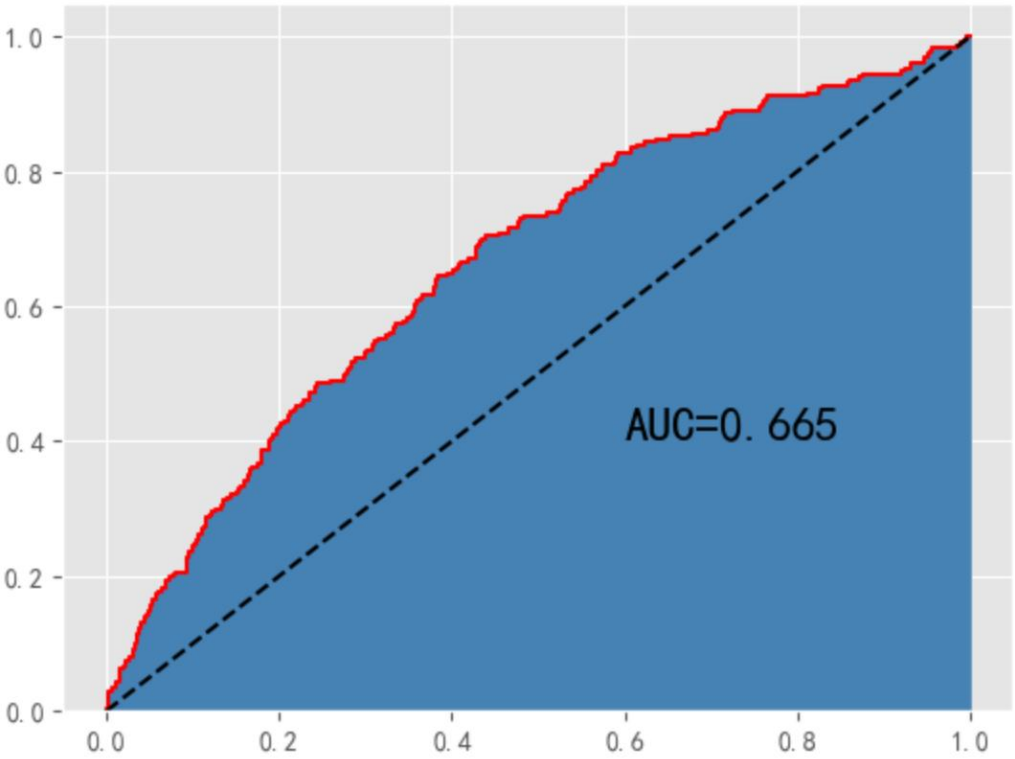


图 3-5 GBDT 模型的评估结果

（3）默认的 XGBboost 分类模型

```
from xgboost import XGBClassifier
from sklearn import metrics
xgboost = XGBClassifier(objective='reg:logistic',n_estimators=1000)
xgboost.fit(X_train, y_train)
xgboost_pred = xgboost.predict(X_test)
```

```
print(pd.crosstab(xgboost_pred, y_test))
print('模型在训练集上的准确率%f' % xgboost.score(X_train, y_train))
print('模型在测试集上的准确率%f' % xgboost.score(X_test, y_test))

fpr, tpr, _ = metrics.roc_curve(y_test, xgboost.predict_proba(X_test)[:, 1])
plt.plot(fpr, tpr, linestyle='solid', color='red')
plt.stackplot(fpr, tpr, color='steelblue')
plt.plot([0, 1], [0, 1], linestyle='dashed', color='black')
plt.text(0.6, 0.4, 'AUC=%.3f' % metrics.auc(fpr, tpr), fontdict=dict(size=18))
plt.show()                                  # 绘制 ROC 曲线
```

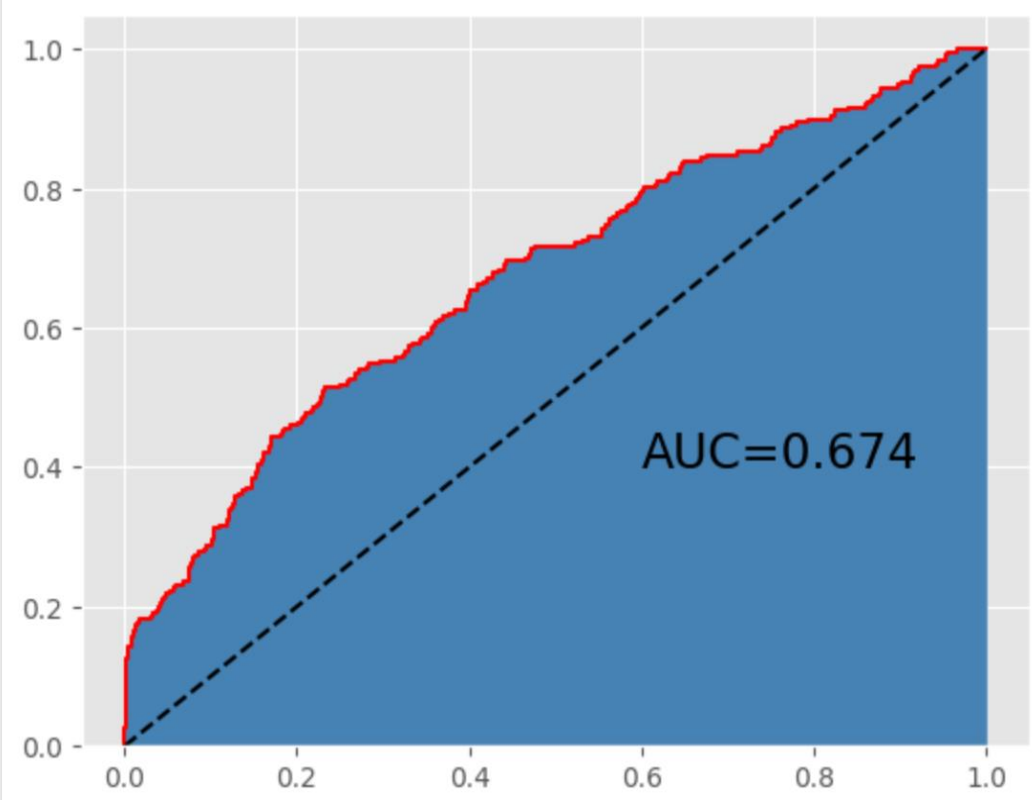


图 3-6 XGBoost 模型的评估结果

所以，XGBoost 模型的评估结果最好，所以继续使用该模型，对该模型的参数进行调整，进一步优化该模型。

3.5 XGBoost 模型的网格搜索

模型的网格搜索是一种用于系统地遍历多种参数组合，通过交叉验证来确定最佳参数设置的方法。它通常用于机器学习中的超参数优化，即模型训练过程中无法从数据中学习到的参数，需要手动设置的参数。模型的网格搜索主要作用和优势是：

（1）参数调优：通过网格搜索可以尝试不同的参数组合，找到使模型性能最优的参数。

（2）避免过拟合：通过交叉验证，网格搜索可以在多个子数据集上评估模型，减少因特定数据分割而引入的偏差或方差，有助于更准确地评估模型在新数据上的泛化能力。

（3）提高模型效率：虽然网格搜索会耗费一定的计算资源，但是它能够系统地搜索参数空间，有效地找到最优参数组合，从而提高模型的训练效率和预测准确率。

代码如下：

```
import pandas as pd
import xgboost as xgb
from sklearn.model_selection import GridSearchCV
params = {'max_depth': [3, 6, 10],
                'learning_rate': [0.01, 0.05, 0.1],
              'n_estimators': [100, 500, 1000],
              'colsample_bytree': [0.3, 0.7]
             }
xgbr = xgb.XGBClassifier(seed=123456)
grid_xgboost = GridSearchCV(estimator=xgbr,
                            param_grid=params,
                            scoring='accuracy',
                            verbose=1)
grid_xgboost.fit(X_train, y_train)
print("Best parameters:", grid_xgboost.best_params_)

调整参数后继续评估调整后的模型
xgboost.set_params(
    objective='reg:logistic',
    n_estimators=1000,
    colsample_bytree=0.3,
    learning_rate= 0.01,
    max_depth=10)
xgboost.fit(X_train, y_train)
fpr, tpr, _ = metrics.roc_curve(y_test, xgboost.predict_proba(X_test)[:, 1])
plt.plot(fpr, tpr, linestyle='solid', color='red')
plt.stackplot(fpr, tpr, color='steelblue')
plt.plot([0, 1], [0, 1], linestyle='dashed', color='black')
plt.text(0.6, 0.4, 'AUC=%.3f' % metrics.auc(fpr, tpr), fontdict=dict(size=18))
plt.show()
```

一般来说，AUC 值在 0.7 以上的模型就有一定的实用性，AUC 值在 0.8 以上的模型表现良好，AUC 值在 0.9 以上的模型表现非常好。 调整参数后的模型的 AUC=0.702>0.7 表明调整后的模型有一定的实用性

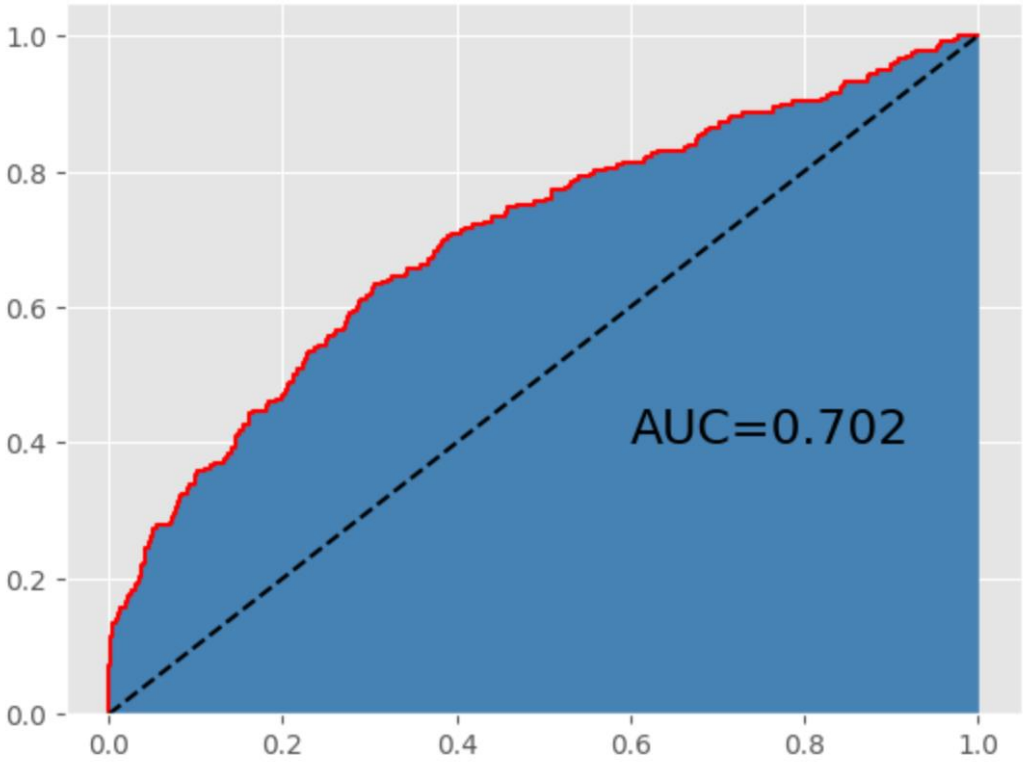


图 3-7 调整后的 XGBoost 模型评估结果

3.6 模型的保存与后续调用

将该调整好的模型保存之后，后续有新的数据录入之后，主要对新数据做好对于的数据处理工作之后，就可以用来预测贷款者是否会逾期，以此来确定是否要通过该人的贷款。

代码如下:

```
xgboost.save_model('is_loan_model.json') #  模型的保存
model = XGBClassifier()
model.load_model('is_loan_model.json')    #  再一次模型的调用
predictions = model.predict(new_data)      # new_data  就是后面需要进行分类预测的新数据
```