

# 信用风险数据分析

## 金融数据挖掘课设作业1

**数据集:** `credit_risk_data.csv` (共1000条记录, 15个字段)

**目标变量:** `default` (1 = 违约, 0 = 未违约)

任务	说明
任务一	读取数据并分析数据结构
任务二	识别并处理缺失值和异常值
任务三	构造具有金融含义的风险特征
任务四	对类别变量进行合理编码
任务五	分析主要变量与违约风险的关系
任务六	相似客户识别 (选取5位客户)

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.metrics.pairwise import cosine_similarity
from scipy.spatial import distance
import warnings
warnings.filterwarnings('ignore')

plt.rcParams['font.family'] = ['SimHei', 'Microsoft YaHei', 'DejaVu Sans']
plt.rcParams['axes.unicode_minus'] = False
```

### 任务一：读取数据并分析数据结构

```
In [2]: # 读取数据
data = pd.read_csv('credit_risk_data.csv')
print(f'数据集形状: {data.shape}')
print(f'共 {data.shape[0]} 条记录, {data.shape[1]} 个字段')
data.head()
```

数据集形状: (1000, 15)  
共 1000 条记录, 15 个字段

```
Out[2]:
```

	customer_id	age	annual_income	loan_amount	credit_score	debt_balance	overdu
0	1	50	221541.78	91696.60	561	40394.05	
1	2	36	94188.67	56828.91	662	63045.22	
2	3	29	168874.75	64936.91	745	74082.42	
3	4	42	102243.80	96760.73	666	86262.02	
4	5	40	45716.89	88801.82	711	51503.42	

```
In [3]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 15 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   customer_id                          1000 non-null   int64
1   age                                  1000 non-null   int64
2   annual_income                        1000 non-null   float64
3   loan_amount                          1000 non-null   float64
4   credit_score                         1000 non-null   int64
5   debt_balance                         1000 non-null   float64
6   overdue_times                        1000 non-null   int64
7   credit_card_utilization              1000 non-null   float64
8   employment_type                      1000 non-null   object
9   loan_purpose                           1000 non-null   object
10  default                              1000 non-null   int64
11  debt_to_income                       1000 non-null   float64
12  loan_to_income                       1000 non-null   float64
13  overdue_intensity                     1000 non-null   float64
14  high_utilization                     1000 non-null   int64
dtypes: float64(7), int64(6), object(2)
memory usage: 117.3+ KB
```

```
In [4]: data.describe()
```

```
Out[4]:
```

	customer_id	age	annual_income	loan_amount	credit_score	debt_
<b>count</b>	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000
<b>mean</b>	500.500000	40.947000	123585.628460	81047.616690	679.057000	49833
<b>std</b>	288.819436	11.160858	45493.742882	29653.993623	69.492395	24328
<b>min</b>	1.000000	22.000000	30000.000000	5000.000000	446.000000	0
<b>25%</b>	250.750000	31.000000	92746.730000	60371.105000	629.000000	33492
<b>50%</b>	500.500000	42.000000	122409.220000	81813.965000	677.000000	49713
<b>75%</b>	750.250000	50.000000	154682.240000	101849.050000	729.000000	67031
<b>max</b>	1000.000000	59.000000	262531.730000	184876.330000	884.000000	127316

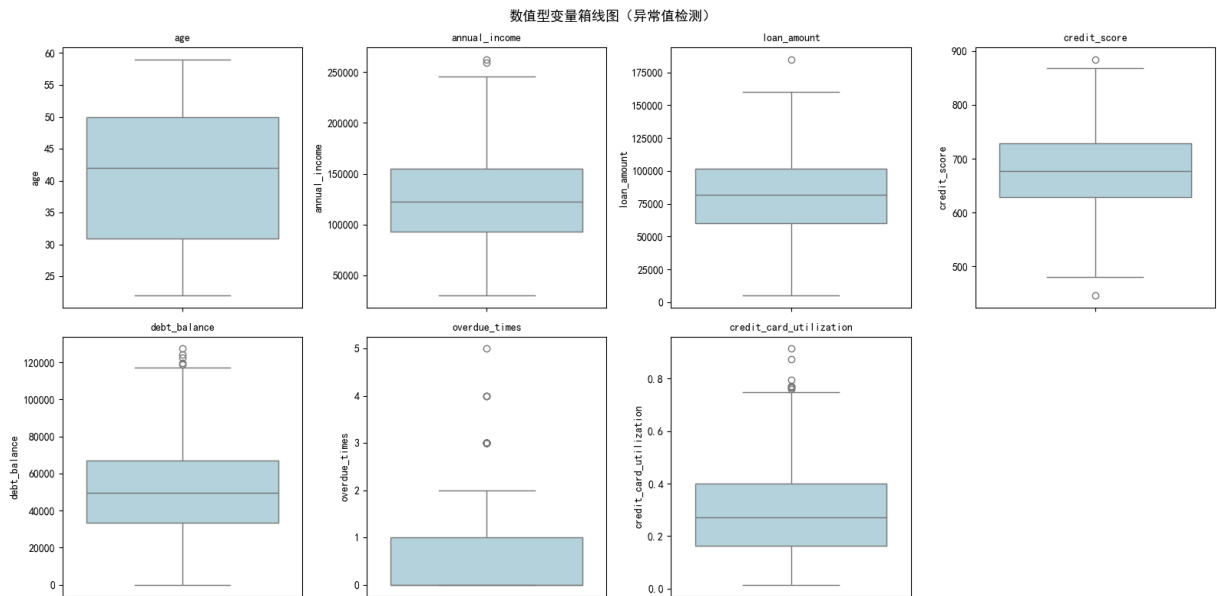
## 任务二：识别并处理缺失值和异常值

```
In [5]: # 缺失值统计
missing = data.isnull().sum()
print('各字段缺失值统计: ')
if missing.sum() == 0:
    print(' 数据集无缺失值')
else:
    print(missing[missing > 0])
```

各字段缺失值统计：  
数据集无缺失值

```
In [6]: # 数值型变量箱线图—直观判断异常值分布
numeric_cols = ['age', 'annual_income', 'loan_amount', 'credit_score',
                'debt_balance', 'overdue_times', 'credit_card_utilization']

fig, axes = plt.subplots(2, 4, figsize=(16, 8))
axes = axes.flatten()
for i, col in enumerate(numeric_cols):
    sns.boxplot(y=data[col], ax=axes[i], color='lightblue')
    axes[i].set_title(col, fontsize=10)
axes[-1].set_visible(False)
plt.suptitle('数值型变量箱线图（异常值检测）', fontsize=13)
plt.tight_layout()
plt.show()
```



```
In [7]: # IQR 截断法处理异常值
data_clean = data.copy()
print('异常值处理（IQR截断法）: ')
for col in numeric_cols:
    Q1 = data_clean[col].quantile(0.25)
    Q3 = data_clean[col].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR
```

```
n_out = ((data_clean[col] < lower) | (data_clean[col] > upper)).sum()
data_clean[col] = data_clean[col].clip(lower, upper)
print(f' {col}: 检测到异常值 {n_out} 个, 截断至 [{lower:.2f}, {upper:.2f}]')
print(f'\n处理后数据形状: {data_clean.shape}')
```

异常值处理 (IQR截断法):

```
age: 检测到异常值 0 个, 截断至 [2.50, 78.50]
annual_income: 检测到异常值 2 个, 截断至 [-156.53, 247585.50]
loan_amount: 检测到异常值 1 个, 截断至 [-1845.81, 164065.97]
credit_score: 检测到异常值 2 个, 截断至 [479.00, 879.00]
debt_balance: 检测到异常值 5 个, 截断至 [-16816.99, 117340.45]
overdue_times: 检测到异常值 28 个, 截断至 [-1.50, 2.50]
credit_card_utilization: 检测到异常值 8 个, 截断至 [-0.19, 0.75]
```

处理后数据形状: (1000, 15)

## 任务三：构造具有金融含义的风险特征

数据集已包含以下衍生特征：

字段	含义	计算方式
debt_to_income	负债收入比	debt_balance / annual_income
loan_to_income	贷款收入比	loan_amount / annual_income
overdue_intensity	逾期强度	overdue_times / 近似成年年限
high_utilization	高信用卡使用率标记	使用率 > 0.7 时取 1

在此基础上，再构造 **3 个新的风险特征**：

```
In [8]: data_feat = data_clean.copy()

# 特征1: 综合负债率—衡量客户全部债务（含本次贷款）对年收入的压力
data_feat['total_liability_ratio'] = (
    data_feat['debt_balance'] + data_feat['loan_amount']
) / data_feat['annual_income']

# 特征2: 综合信用风险指数—信用卡使用率与信用质量的复合风险（评分越低风险越大）
data_feat['credit_risk_index'] = (
    data_feat['credit_card_utilization'] * (1 - data_feat['credit_score'] / 900)
)

# 特征3: 净偿还能力—(年收入 - 现有负债余额) / 贷款金额，正值越大偿还能力越强
data_feat['net_repay_capacity'] = (
    data_feat['annual_income'] - data_feat['debt_balance']
) / data_feat['loan_amount']

print('新增特征统计描述: ')
data_feat[['total_liability_ratio', 'credit_risk_index', 'net_repay_capacity']].des
```

新增特征统计描述：

Out[8]:

	total_liability_ratio	credit_risk_index	net_repay_capacity
count	1000.000000	1000.000000	1000.000000
mean	1.278178	0.070887	1.214944
std	0.834275	0.045656	1.908211
min	0.033475	0.003229	-2.437530
25%	0.778845	0.036151	0.457851
50%	1.077302	0.063314	0.885655
75%	1.482306	0.096307	1.482959
max	7.091790	0.300902	29.872812

## 任务四：对类别变量进行合理编码

In [9]:

```
# 查看类别变量分布
print('就业类型分布: ')
print(data_feat['employment_type'].value_counts())
print()
print('贷款用途分布: ')
print(data_feat['loan_purpose'].value_counts())
```

就业类型分布:

```
employment_type
民营企业      358
国企/事业单位  242
个体工商户    174
自由职业      157
学生          69
Name: count, dtype: int64
```

贷款用途分布:

```
loan_purpose
消费      355
装修      255
经营周转  190
教育      119
医疗       81
Name: count, dtype: int64
```

In [10]:

```
# 标签编码 (Label Encoding)
data_encoded = data_feat.copy()

employment_map = {
    '国企/事业单位': 1, '民营企业': 2, '个体工商户': 3, '自由职业': 4, '学生': 5
}
loan_purpose_map = {
    '消费': 1, '装修': 2, '教育': 3, '医疗': 4, '经营周转': 5
}

data_encoded['employment_code'] = data_encoded['employment_type'].map(employment_ma
```

```
data_encoded['purpose_code'] = data_encoded['loan_purpose'].map(loan_purpose_map)

print('就业类型编码映射:', employment_map)
print('贷款用途编码映射:', loan_purpose_map)
print()
data_encoded[['employment_type', 'employment_code', 'loan_purpose', 'purpose_code']]
```

就业类型编码映射: {'国企/事业单位': 1, '民营企业': 2, '个体工商户': 3, '自由职业': 4, '学生': 5}

贷款用途编码映射: {'消费': 1, '装修': 2, '教育': 3, '医疗': 4, '经营周转': 5}

Out[10]:

	employment_type	employment_code	loan_purpose	purpose_code
0	国企/事业单位	1	装修	2
1	自由职业	4	经营周转	5
2	个体工商户	3	消费	1
3	国企/事业单位	1	装修	2
4	民营企业	2	装修	2
5	民营企业	2	消费	1
6	个体工商户	3	医疗	4
7	国企/事业单位	1	教育	3

## 任务五：分析主要变量与违约风险的关系

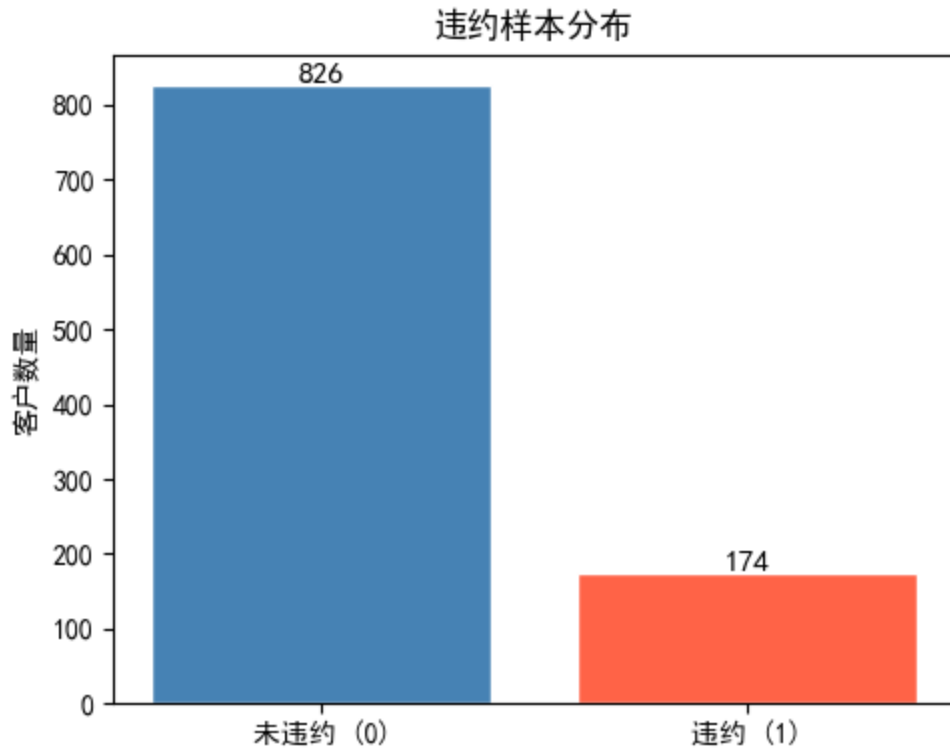
```
In [11]: # 违约样本总体分布
default_counts = data_encoded['default'].value_counts()
total = len(data_encoded)
print(f'样本总数: {total}')
print(f'未违约 (0): {default_counts[0]} ({default_counts[0]/total*100:.1f}%)')
print(f'违约 (1): {default_counts[1]} ({default_counts[1]/total*100:.1f}%)')

fig, ax = plt.subplots(figsize=(5, 4))
ax.bar(['未违约 (0)', '违约 (1)'], [default_counts[0], default_counts[1]],
       color=['steelblue', 'tomato'], edgecolor='white')
ax.set_title('违约样本分布')
ax.set_ylabel('客户数量')
for i, v in enumerate([default_counts[0], default_counts[1]]):
    ax.text(i, v + 3, str(v), ha='center', fontsize=11)
plt.tight_layout()
plt.show()
```

样本总数: 1000

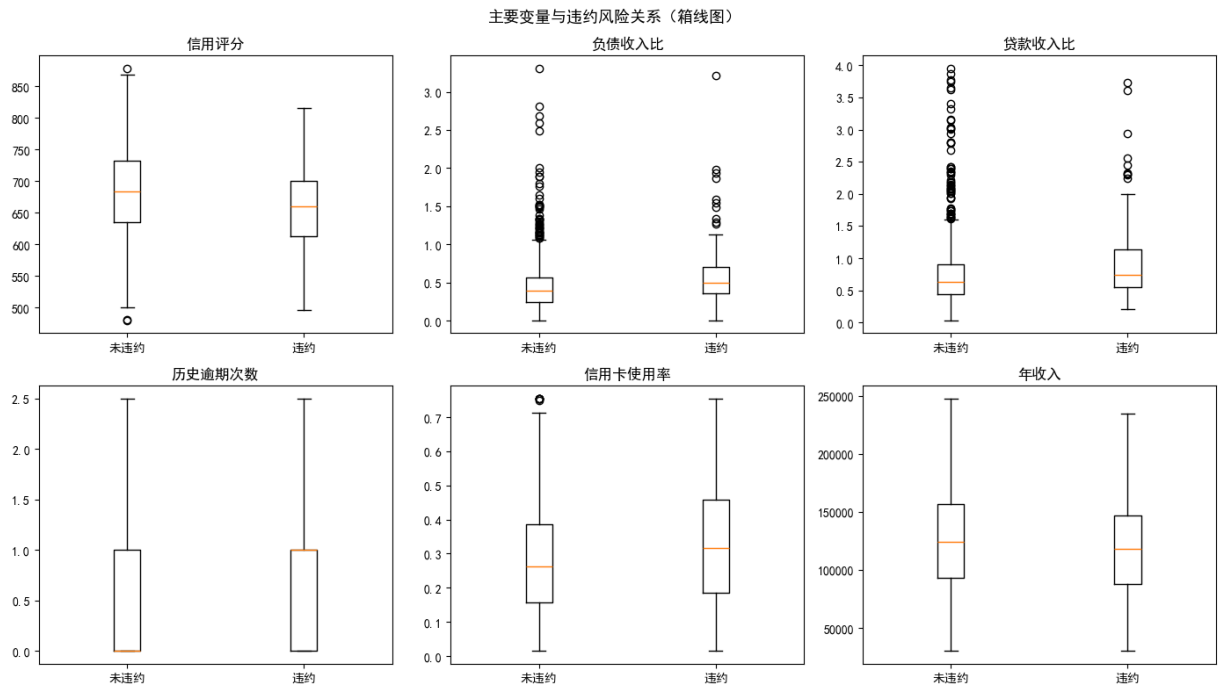
未违约 (0): 826 (82.6%)

违约 (1): 174 (17.4%)



```
In [12]: # 主要数值变量与违约状态的箱线图对比
key_vars = ['credit_score', 'debt_to_income', 'loan_to_income',
            'overdue_times', 'credit_card_utilization', 'annual_income']
key_labels = ['信用评分', '负债收入比', '贷款收入比', '历史逾期次数', '信用卡使用率', '年收入']

fig, axes = plt.subplots(2, 3, figsize=(14, 8))
axes = axes.flatten()
for i, (col, label) in enumerate(zip(key_vars, key_labels)):
    d0 = data_encoded[data_encoded['default'] == 0][col]
    d1 = data_encoded[data_encoded['default'] == 1][col]
    axes[i].boxplot([d0, d1], labels=['未违约', '违约'])
    axes[i].set_title(label)
plt.suptitle('主要变量与违约风险关系 (箱线图)', fontsize=13)
plt.tight_layout()
plt.show()
```

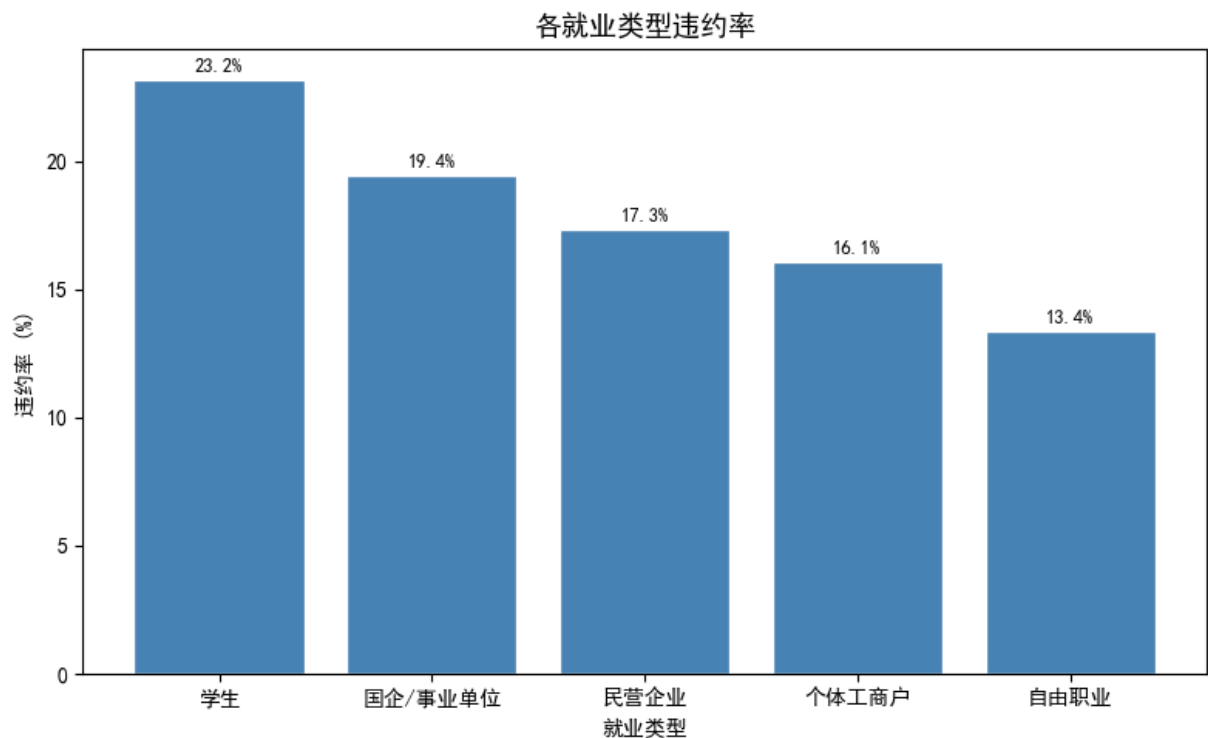


```
In [13]: # 各就业类型的违约率
emp_stats = data_encoded.groupby('employment_type')['default'].agg(['sum', 'count'])
emp_stats.columns = ['违约数', '总数']
emp_stats['违约率(%)'] = (emp_stats['违约数'] / emp_stats['总数'] * 100).round(2)
emp_stats = emp_stats.sort_values('违约率(%)', ascending=False)

fig, ax = plt.subplots(figsize=(8, 5))
bars = ax.bar(emp_stats.index, emp_stats['违约率(%)'], color='steelblue', edgecolor='black')
ax.set_title('各就业类型违约率', fontsize=13)
ax.set_ylabel('违约率 (%)')
ax.set_xlabel('就业类型')
for bar, rate in zip(bars, emp_stats['违约率(%)']):
    ax.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.3,
            f'{rate:.1f}%', ha='center', fontsize=9)
plt.tight_layout()
plt.show()

emp_stats
```





Out[13]:

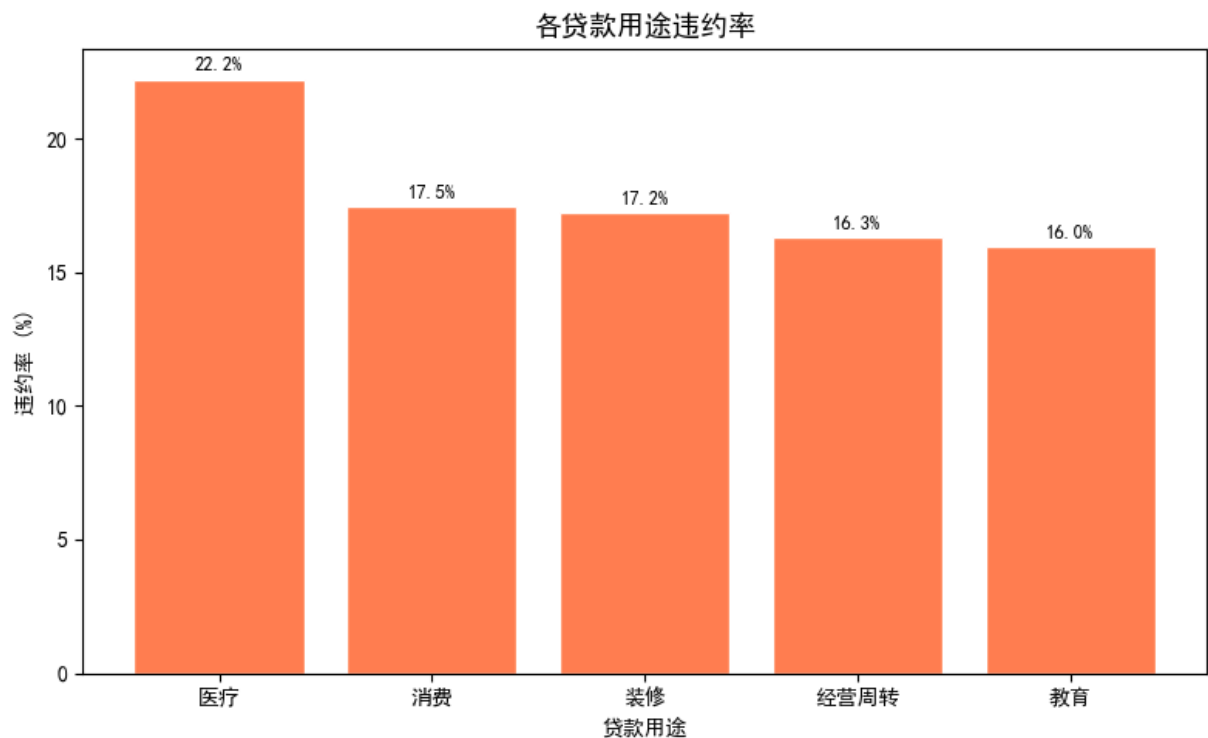
	违约数	总数	违约率(%)
学生	16	69	23.19
国企/事业单位	47	242	19.42
民营企业	62	358	17.32
个体工商户	28	174	16.09
自由职业	21	157	13.38

employment_type			
学生	16	69	23.19
国企/事业单位	47	242	19.42
民营企业	62	358	17.32
个体工商户	28	174	16.09
自由职业	21	157	13.38

```
In [14]: # 各贷款用途的违约率
purpose_stats = data_encoded.groupby('loan_purpose')['default'].agg(['sum', 'count'])
purpose_stats.columns = ['违约数', '总数']
purpose_stats['违约率(%)'] = (purpose_stats['违约数'] / purpose_stats['总数'] * 100)
purpose_stats = purpose_stats.sort_values('违约率(%)', ascending=False)

fig, ax = plt.subplots(figsize=(8, 5))
bars = ax.bar(purpose_stats.index, purpose_stats['违约率(%)'], color='coral', edgecolor='black')
ax.set_title('各贷款用途违约率', fontsize=13)
ax.set_ylabel('违约率 (%)')
ax.set_xlabel('贷款用途')
for bar, rate in zip(bars, purpose_stats['违约率(%)']):
    ax.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.3,
            f'{rate:.1f}%', ha='center', fontsize=9)
plt.tight_layout()
plt.show()

purpose_stats
```



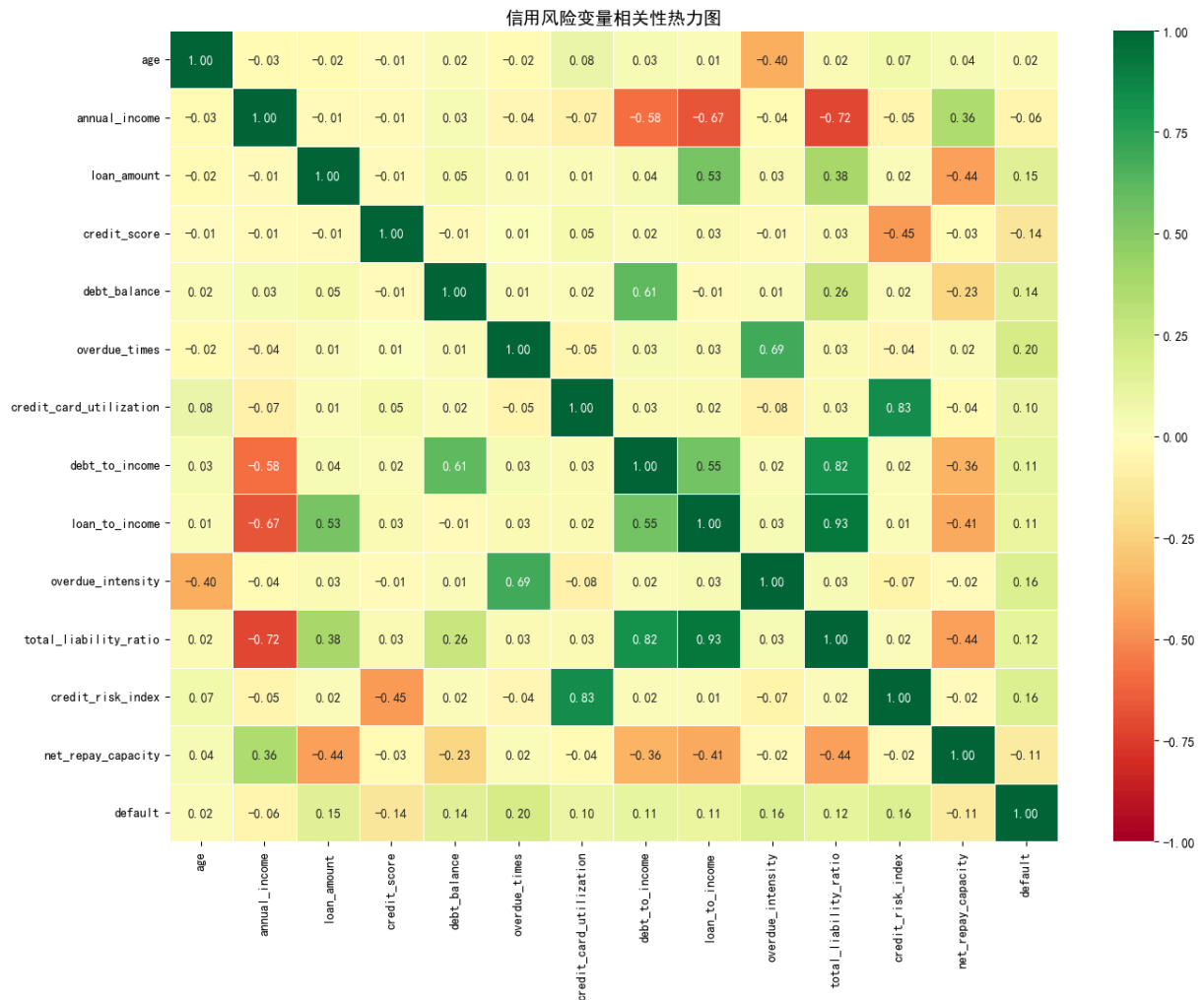
Out[14]:

	违约数	总数	违约率(%)
--	-----	----	--------

loan_purpose			
医疗	18	81	22.22
消费	62	355	17.46
装修	44	255	17.25
经营周转	31	190	16.32
教育	19	119	15.97

```
In [15]: # 数值变量相关性热力图 (含目标变量 default)
corr_cols = ['age', 'annual_income', 'loan_amount', 'credit_score',
             'debt_balance', 'overdue_times', 'credit_card_utilization',
             'debt_to_income', 'loan_to_income', 'overdue_intensity',
             'total_liability_ratio', 'credit_risk_index', 'net_repay_capacity',
             'default']
corr = data_encoded[corr_cols].corr()

plt.figure(figsize=(14, 11))
sns.heatmap(corr, annot=True, fmt='.2f', cmap='RdYlGn',
            center=0, vmin=-1, vmax=1, linewidths=0.5)
plt.title('信用风险变量相关性热力图', fontsize=14)
plt.tight_layout()
plt.show()
```



## 任务六：相似客户识别

参考 Chap2 中的相似性度量方法，选取 **5 位客户**，利用 **余弦相似度** (Cosine Similarity) 和**欧氏距离** (Euclidean Distance) 度量客户之间的相似程度，识别风险画像相近的客户群体。

计算前先对特征进行标准化 (StandardScaler)，消除量纲差异的影响。

```
In [16]: # 选取5位客户 (包含违约和未违约样本)
sample_idx = [0, 3, 11, 20, 26]

sim_features = ['age', 'annual_income', 'loan_amount', 'credit_score',
               'debt_balance', 'overdue_times', 'credit_card_utilization',
               'debt_to_income', 'loan_to_income']

print('选取的样本客户信息：')
data_encoded[['customer_id'] + sim_features + ['default']].iloc[sample_idx]
```

选取的样本客户信息：

```
Out[16]:
```

	customer_id	age	annual_income	loan_amount	credit_score	debt_balance	overc
0	1	50	221541.78	91696.60	561	40394.05	
3	4	42	102243.80	96760.73	666	86262.02	
11	12	24	119193.44	54805.24	687	48976.52	
20	21	33	108443.23	93420.78	679	0.00	
26	27	36	43117.35	65689.93	667	80492.77	

```
In [17]: # 标准化处理
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data_encoded[sim_features])
sample_scaled = data_scaled[sample_idx]

# 余弦相似度矩阵
sim_matrix = cosine_similarity(sample_scaled)
cids = [f'客户{data_encoded["customer_id"].iloc[i]}' for i in sample_idx]
sim_df = pd.DataFrame(sim_matrix, index=cids, columns=cids)

print('余弦相似度矩阵 (越接近 1 越相似): ')
sim_df.round(4)
```

余弦相似度矩阵 (越接近 1 越相似):

```
Out[17]:
```

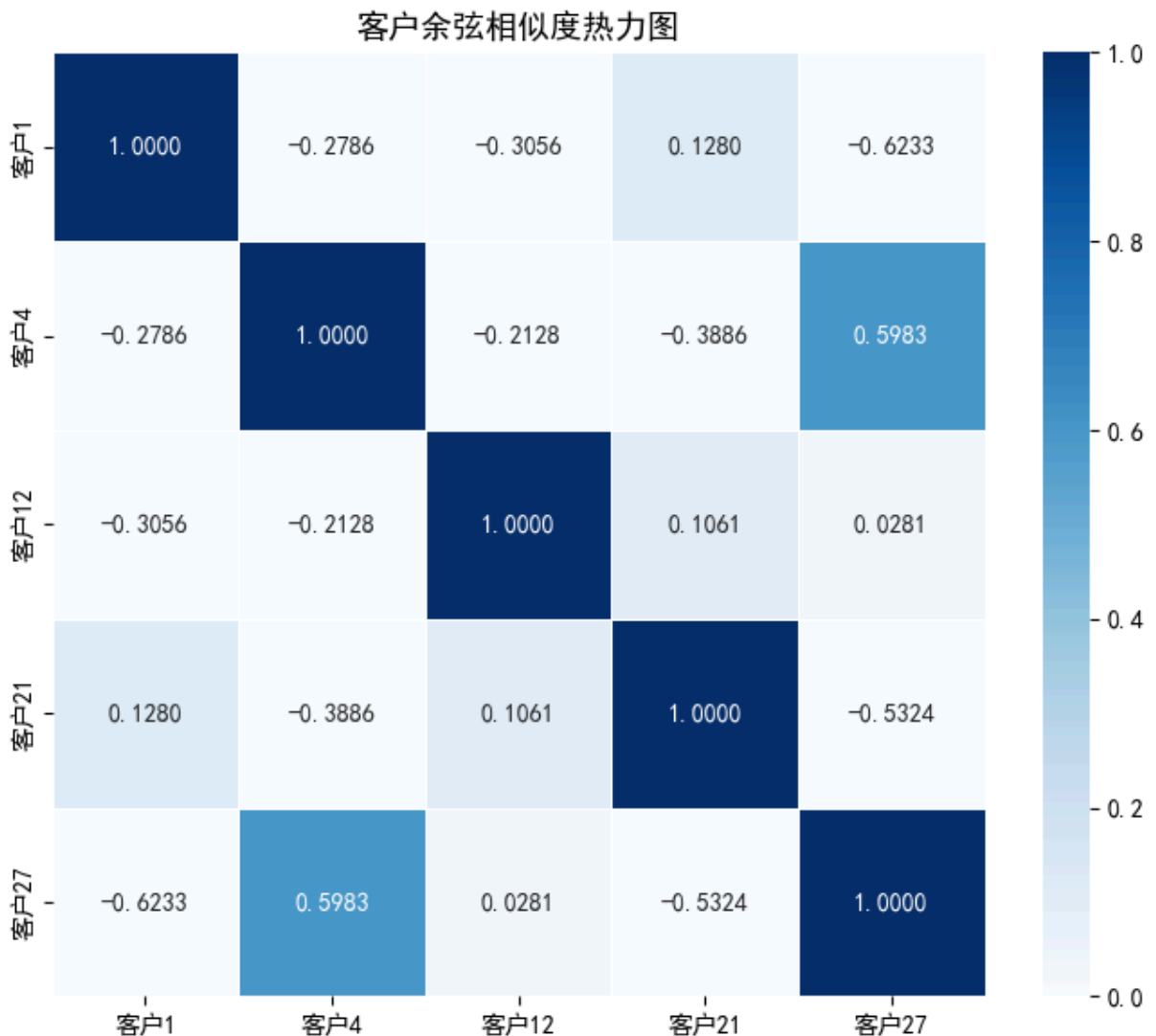
	客户1	客户4	客户12	客户21	客户27
客户1	1.0000	-0.2786	-0.3056	0.1280	-0.6233
客户4	-0.2786	1.0000	-0.2128	-0.3886	0.5983
客户12	-0.3056	-0.2128	1.0000	0.1061	0.0281
客户21	0.1280	-0.3886	0.1061	1.0000	-0.5324
客户27	-0.6233	0.5983	0.0281	-0.5324	1.0000

```
In [18]: # 欧氏距离 (两两计算)
print('客户两两欧氏距离 (越小越相似): ')
n = len(sample_idx)
for i in range(n):
    for j in range(i + 1, n):
        dist_val = distance.euclidean(sample_scaled[i], sample_scaled[j])
        ci = data_encoded['customer_id'].iloc[sample_idx[i]]
        cj = data_encoded['customer_id'].iloc[sample_idx[j]]
        print(f' 客户{ci} vs 客户{cj}: 欧氏距离 = {dist_val:.4f}')
```

客户两两欧氏距离 (越小越相似):

客户1 vs 客户4: 欧氏距离 = 4.8495  
客户1 vs 客户12: 欧氏距离 = 4.3050  
客户1 vs 客户21: 欧氏距离 = 3.9870  
客户1 vs 客户27: 欧氏距离 = 7.2211  
客户4 vs 客户12: 欧氏距离 = 3.7238  
客户4 vs 客户21: 欧氏距离 = 4.5841  
客户4 vs 客户27: 欧氏距离 = 3.7719  
客户12 vs 客户21: 欧氏距离 = 3.1860  
客户12 vs 客户27: 欧氏距离 = 5.0485  
客户21 vs 客户27: 欧氏距离 = 6.5770

```
In [19]: # 相似度热力图可视化
plt.figure(figsize=(7, 6))
sns.heatmap(sim_df, annot=True, fmt='.4f', cmap='Blues',
            vmin=0, vmax=1, linewidths=0.5)
plt.title('客户余弦相似度热力图', fontsize=13)
plt.tight_layout()
plt.show()
```



分析总结

1. **数据结构**：数据集共 1000 条记录、15 个字段，包含数值型、整数型和类别型变量，**无缺失值**。
2. **异常值处理**：采用 IQR（四分位距）截断法对 7 个数值型变量进行处理，其中 `overdue_times` 检出 28 个异常值最多，有效抑制极端值对后续分析的干扰。
3. **特征工程**：在原有 4 个衍生特征（负债收入比、贷款收入比、逾期强度、高信用卡使用率标记）基础上，新增：
  - `total_liability_ratio`：综合负债率，刻画客户全部债务压力（与 default 相关系数 +0.122）
  - `credit_risk_index`：综合信用风险指数，量化使用率与信用评分的交互风险（+0.157）
  - `net_repay_capacity`：净偿还能力，衡量扣除现有负债后的偿贷余力（-0.110）
4. **类别编码**：对就业类型（5类）和贷款用途（5类）采用有序标签编码，保留类别差异。
5. **违约风险分析**：
  - `overdue_times`（历史逾期次数）是与违约**相关最强**的变量（ $r = +0.203$ ），违约客户均值（0.91次）约为未违约客户（0.52次）的1.7倍
  - `credit_score` 与违约**负相关**（ $r = -0.140$ ），违约客户均值 657.9，未违约均值 683.5
  - `debt_to_income`、`credit_card_utilization` 均与违约**正相关**（分别为 +0.110、+0.097）
  - 就业类型违约率差异显著：学生（23.2%）> 国企/事业单位（19.4%）> 自由职业（13.4%）
  - 贷款用途违约率：医疗（22.2%）最高，教育（16.0%）最低
6. **相似客户识别**：基于标准化数值特征，通过余弦相似度和欧氏距离识别风险画像相近的客户。5位样本中客户4与客户27余弦相似度最高（0.5983），欧氏距离最近的为客户12与客户21（3.186），可为风险定价和个性化授信策略提供参考依据。