学生校园消费行为分析

目录

[1.数据导入与预处理 1](#_Toc190014420)

[1.1加载数据，理解字段含义 1](#_Toc190014421)

[1.2建立关联 4](#_Toc190014422)

[2.食堂就餐行为分析 4](#_Toc190014423)

[2.1绘制各食堂就餐人次的占比饼图，分析高峰期 4](#_Toc190014424)

[2.2绘制工作日和非工作日的就餐时间变化曲线图 6](#_Toc190014425)

[3.学生消费行为分析 12](#_Toc190014426)

[3.1根据整体校园消费数据，计算月人均刷卡频次和人均消费额 12](#_Toc190014427)

[3.2以人均消费额为分界线，将学生群体划分为两个群体，通过对比本月这两个群体在不同消费地点所消费消费额的饼图，分析群体消费特点。 15](#_Toc190014428)

[3.3对低消费学生群体的行为进行分析，探讨是否存在特征，作为助学金评定的参考 19](#_Toc190014429)

1.数据导入与预处理

1.1加载数据，理解字段含义

查阅附录一：数据说明，理解字段含义后，导入文件。

data1.csv、data2.csv对应学生ID表、消费记录表，其中都有CardNo字段，代表每个学生唯一的校园卡号，在后面的处理中可以利用这个特性连接两份表格。

import pandas as pd

data1=pd.read\_csv(r'C:\Users\DELL\Desktop\数字图书馆关键技术期末作业-20250108\学生校园消费行为分析\data1.csv', sep=',', encoding='gbk')

data2=pd.read\_csv(r'C:\Users\DELL\Desktop\数字图书馆关键技术期末作业-20250108\学生校园消费行为分析\data2.csv', sep=',', encoding='gbk')

data3=pd.read\_csv(r'C:\Users\DELL\Desktop\数字图书馆关键技术期末作业-20250108\学生校园消费行为分析\data3.csv', sep=',', encoding='gbk')

定义一个函数，用于处理缺失值

*# 定义一个函数来检测和处理缺失值*

def detect\_and\_handle\_missing\_values(df):

*# 打印缺失值情况*

    print("缺失值检测结果：")

    print(df.isna().sum())

*# 使用线性插值法填充缺失值*

    df.interpolate(method='linear', inplace=True)

    return df

定义一个函数，用于处理异常值

*# 定义一个函数来检测和处理异常值*

def detect\_and\_handle\_outliers(df):

*# 选择数值类型的列*

    numeric\_cols = df.select\_dtypes(include=[np.number]).columns

*# 计算IQR*

    Q1 = df[numeric\_cols].quantile(0.25)

    Q3 = df[numeric\_cols].quantile(0.75)

    IQR = Q3 - Q1

*# 定义异常值的范围*

    lower\_bound = Q1 - 1.5 \* IQR

    upper\_bound = Q3 + 1.5 \* IQR

*# 检测异常值*

    outliers = (df[numeric\_cols] < lower\_bound) | (df[numeric\_cols] > upper\_bound)

*# 打印异常值*

    print("异常值检测结果：")

    print(outliers.sum())

*# 处理异常值：用中位数替换*

    df[numeric\_cols] = df[numeric\_cols].where(~outliers, df[numeric\_cols].median(), axis=1)

    return df

对数据进行处理，检测缺失值与异常值的情况并输出检测结果。

*# 对 data1、data2 和 data3 进行缺失值和异常值的检测和处理*

data1 = detect\_and\_handle\_missing\_values(data1)

data1 = detect\_and\_handle\_outliers(data1)

data2 = detect\_and\_handle\_missing\_values(data2)

data2 = detect\_and\_handle\_outliers(data2)

data3 = detect\_and\_handle\_missing\_values(data3)

data3 = detect\_and\_handle\_outliers(data3)

data1缺失值检测结果：

Index 0

CardNo 0

Sex 0

Major 0

AccessCardNo 0

dtype: int64

异常值检测结果：

Index 0

CardNo 2

AccessCardNo 697

dtype: int64

data2

Index 0

CardNo 0

PeoNo 0

Date 0

Money 0

FundMoney 0

Surplus 0

CardCount 0

Type 0

TermNo 0

TermSerNo 512106

conOperNo 519116

OperNo 0

Dept 0

dtype: int64

异常值检测结果：

Index 828

CardNo 10768

PeoNo 105352

Money 5197

FundMoney 18382

Surplus 23385

CardCount 16007

TermNo 82

conOperNo 28193

OperNo 0

dtype: int64

data3

缺失值检测结果：

Index 0

AccessCardNo 0

Date 0

Address 0

Access 0

Describe 0

dtype: int64

异常值检测结果：

Index 0

AccessCardNo 0

Access 1407

dtype: int64

处理缺失值、异常值，并保存为“task1\_1\_X.csv”（如果包含多张数据表，X可从1开始往后编号）

*# 将处理后的数据保存到指定文件名*

datasets = [data1, data2, data3]

for i, df in enumerate(datasets, start=1):

    df.to\_csv(f'C:/Users/DELL/Desktop/数字图书馆关键技术期末作业-20250108/学生校园消费行为分析/task\_1\_1\_{i}.csv', index=False, encoding='utf-8-sig')

    print(f"处理完成，结果已保存为 task\_1\_1\_{i}.csv")

处理完成，结果已保存为 task\_1\_1\_1.csv

处理完成，结果已保存为 task\_1\_1\_2.csv

处理完成，结果已保存为 task\_1\_1\_3.csv

1.2建立关联

利用文件间的共同字段，将数据两两关联。将data1.csv中的学生个人信息与data2.csv中的消费记录建立关联将data1.csv中的学生个人信息与data3.csv中的门禁进出记录建立关联。

*# 将 data1 与 data2 进行关联*

merged\_data1\_2 = pd.merge(data1, data2, on='CardNo', how='inner')

*# 将 data1 与 data3 进行关联*

merged\_data1\_3 = pd.merge(data1, data3, on='AccessCardNo', how='inner')

处理结果分别保存为“task1\_2\_1.csv”、“task1\_2\_2.csv”

*# 保存结果*

merged\_data1\_2.to\_csv('C:/Users/DELL/Desktop/数字图书馆关键技术期末作业-20250108/学生校园消费行为分析/task1\_2\_1.csv', index=False, encoding='utf-8-sig')

merged\_data1\_3.to\_csv('C:/Users/DELL/Desktop/数字图书馆关键技术期末作业-20250108/学生校园消费行为分析/task1\_2\_2.csv', index=False, encoding='utf-8-sig')

print("处理完成，结果已保存为 task1\_2\_1.csv 和 task1\_2\_2.csv")

处理完成，结果已保存为 task1\_2\_1.csv 和 task1\_2\_2.csv

2.食堂就餐行为分析

2.1绘制各食堂就餐人次的占比饼图，分析高峰期

数据处理，时间间隔近的视为一次（15分钟内）。

*# 将日期列转换为 datetime 类型*

df['Date'] = pd.to\_datetime(df['Date'], format='%Y/%m/%d %H:%M')

*# 按照15分钟进行分组，并只保留每个时间段的第一条记录*

df['TimeBucket'] = df['Date'].dt.floor('15T')  *# 每15分钟一个时间段*

df = df.groupby('TimeBucket').first().reset\_index()

*# 统计每天的消费次数*

daily\_counts = df.groupby('Date').size().reset\_index(name='Count')

print(daily\_counts)

*# 统计每个食堂的消费次数*

canteen1 = df['Dept'].apply(str).str.contains('第一食堂').sum()

canteen2 = df['Dept'].apply(str).str.contains('第二食堂').sum()

canteen3 = df['Dept'].apply(str).str.contains('第三食堂').sum()

canteen4 = df['Dept'].apply(str).str.contains('第四食堂').sum()

canteen5 = df['Dept'].apply(str).str.contains('第五食堂').sum()

设置时间段，过滤数据。

*# 添加时间段列*

def get\_time\_period(hour):

    if 6 <= hour < 10:

        return '早'

    elif 10 <= hour < 17:

        return '中'

    else:

        return '晚'

df['时间段'] = df['Date'].dt.hour.apply(get\_time\_period)

*# 过滤只包含 canteen1 到 canteen5 的记录*

canteens = ['第一食堂', '第二食堂', '第三食堂', '第四食堂', '第五食堂']

df\_filtered = df[df['Dept'].isin(canteens)]

*# 分组统计每个时间段内各个食堂的就餐人数*

time\_period\_canteen\_counts = df\_filtered.groupby(['时间段', 'Dept']).size().unstack(fill\_value=0)

*# 打印统计结果*

print(time\_period\_canteen\_counts)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 时间段 | 第一食堂 | 第三食堂 | 第二食堂 | 第五食堂 | 第四食堂 |
| 中 | 28066 | 32181 | 55280 | 53589 | 35191 |
| 早 | 15572 | 595 | 50482 | 33896 | 135 |
| 晚 | 18452 | 19327 | 49111 | 30130 | 25515 |

确定各食堂的就餐高峰期

*# 确定高峰期*

peak\_periods = time\_period\_canteen\_counts.idxmax(axis=0)

print("每个食堂的高峰期：")

for canteen, peak\_period in peak\_periods.items():

    print(f"{canteen}: {peak\_period}")

每个食堂的高峰期：

第一食堂: 中

第三食堂: 中

第二食堂: 中

第五食堂: 中

第四食堂: 中

可以看出，五个食堂在一天中的高峰就餐时间段都是午餐时段（10:00—17:00）。

绘制饼图，分析早中晚餐的高峰期是哪个时间段。将早中晚的就餐占比分别绘制三个饼图，并放在同一张图片内。

*# 设置字体*

font\_path = r"C:\Users\DELL\Desktop\数字图书馆关键技术期末作业-20250108\OPPOSans-R.ttf"  *# 请根据实际情况调整路径*

try:

    font = FontProperties(fname=font\_path)

except Exception as e:

    print(f"加载字体失败: {e}")

    font = None

*# 设置全局字体配置*

plt.rcParams['font.sans-serif'] = ['OPPOSans'] if font is None else [font.get\_name()]

plt.rcParams['axes.unicode\_minus'] = False  *# 解决负号显示问题*

*# 提高分辨率*

plt.rcParams['figure.dpi'] = 150  *# 设置图像分辨率为150 DPI*

*# 绘制三个饼图，每个时间段一个饼图*

time\_periods = ['早', '中', '晚']

fig, axes = plt.subplots(1, 3, figsize=(18, 6))

for i, time\_period in enumerate(time\_periods):

    ax = axes[i]

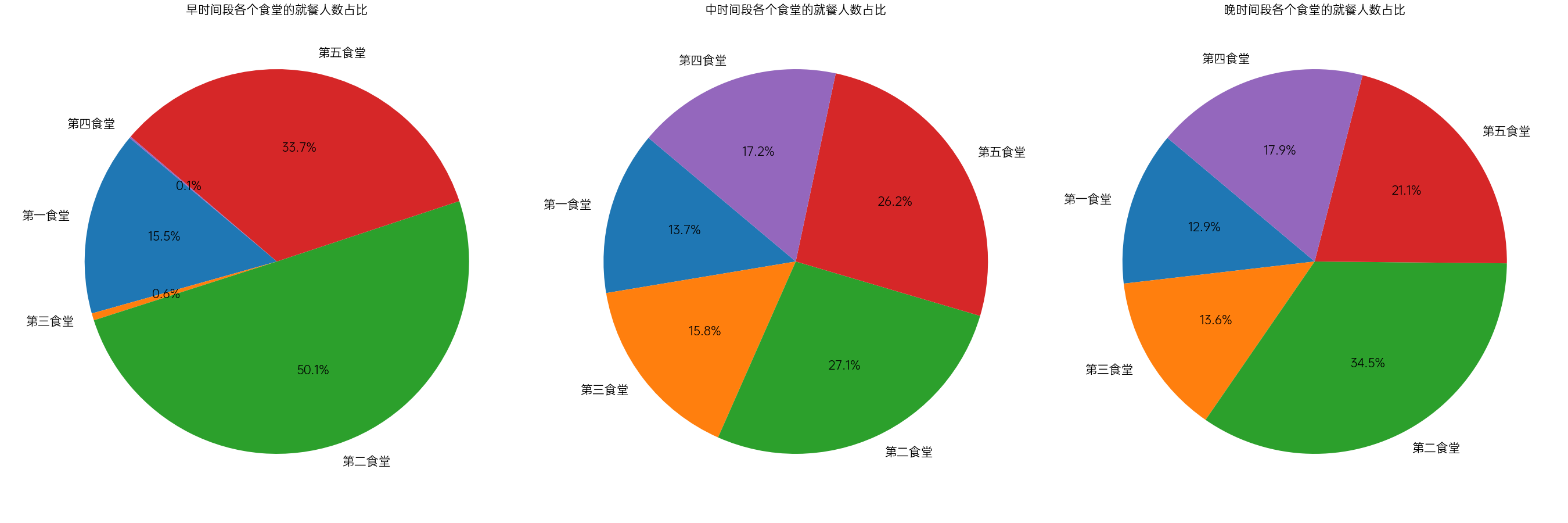
    canteen\_counts = time\_period\_canteen\_counts.loc[time\_period]

    ax.pie(canteen\_counts, labels=canteen\_counts.index, autopct='%1.1f%%', startangle=140, textprops={'fontproperties': font})

    ax.set\_title(f'{time\_period}时间段各个食堂的就餐人数占比', fontproperties=font)

plt.tight\_layout()

plt.show()



可以看出，早餐时间段去第二食堂、第五食堂吃饭的人最多，其中去第二食堂的人数占全部就餐者的50.1%，去第五食堂吃饭的人占33.7%；午餐时间段去各个食堂吃饭的人比较平均，其中第二食堂、第五食堂分别占27.1%、26.2%；晚餐时间段去第二食堂吃饭的人最多，达到了34.5%，其次是去第五食堂吃饭的人，达到了21.7%。

2.2绘制工作日和非工作日的就餐时间变化曲线图

转换时间类型

*#将 Date 列转换为 datetime 类型*

df['Date'] = pd.to\_datetime(df['Date'], format='%Y/%m/%d %H:%M')

*# 显示数据类型*

print(df.dtypes)

*# 检查并过滤掉无效的日期值*

invalid\_dates = df[df['Date'].isna()]

print("无效日期的数量:", invalid\_dates.shape[0])

print("无效日期的具体值:")

print(invalid\_dates['Date'].head())

*# 查看无效日期的具体值*

print("无效日期的详细信息:")

print(invalid\_dates.head(10))

[519367 rows x 15 columns]

Index int64

CardNo int64

PeoNo int64

Date datetime64[ns]

Money float64

FundMoney float64

Surplus float64

CardCount int64

Type object

TermNo int64

TermSerNo object

conOperNo float64

OperNo int64

Dept object

是否工作日 bool

dtype: object

无效日期的数量: 0

无效日期的具体值:

Series([], Name: Date, dtype: datetime64[ns])

无效日期的详细信息:

Empty DataFrame

Columns: [Index, CardNo, PeoNo, Date, Money, FundMoney, Surplus, CardCount, Type, TermNo, TermSerNo, conOperNo, OperNo, Dept, 是否工作日]

Index: []

*# 过滤掉无效的日期值*

df = df.dropna(subset=['Date'])

*# 再次检查数据类型，确保Date列已经转换为datetime类型*

if df['Date'].dtype == 'datetime64[ns]':

    print("Date列已经成功转换为datetime类型")

else:

    print("Date列转换失败，仍然为object类型")

Date列已经成功转换为datetime类型

判断工作日与非工作日

*# 添加一列标识工作日和非工作日*

df['是否工作日'] = df['Date'].dt.weekday < 5  *# 0-4为工作日，5-6为周末*

*# 打印结果*

print(df)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Index | CardNo | PeoNo | ... | OperNo | Dept | 是否工作日 |
| 0 | 117342773 | 181316 | 20181316 | ... | 235 | 第一食堂 | False |
| 1 | 117344766 | 181316 | 20181316 | ... | 27 | 第二食堂 | False |
| 2 | 117346258 | 181316 | 20181316 | ... | 27 | 第二食堂 | True |
| 3 | 117308066 | 181317 | 20181317 | ... | 133 | 好利来食品店 | False |
| 4 | 117309001 | 181317 | 20181317 | ... | 48 | 好利来食品店 | True |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 519362 | 117159040 | 182706 | 20182706 | ... | 204 | 第五食堂 | True |
| 519363 | 117159346 | 182706 | 20182706 | ... | 175 | 第五食堂 | True |
| 519364 | 117161216 | 182706 | 20182706 | ... | 166 | 第五食堂 | True |
| 519365 | 117165544 | 182706 | 20182706 | ... | 220 | 第五食堂 | True |
| 519366 | 117139394 | 182707 | 20182707 | ... | 15 | 第三食堂 | True |

[519367 rows x 15 columns]

分别统计工作日和非工作日的每小时就餐人数。

*# 按小时统计就餐人数*

df['小时'] = df['Date'].dt.hour

*# 分组统计工作日和非工作日的每小时就餐人数*

workday\_counts = df[df['是否工作日']].groupby('小时').size()

non\_workday\_counts = df[~df['是否工作日']].groupby('小时').size()

绘制工作日曲线图

*# 设置字体*

font\_path = r"C:\Users\DELL\Desktop\数字图书馆关键技术期末作业-20250108\OPPOSans-R.ttf"  *# 请根据实际情况调整路径*

try:

    font = FontProperties(fname=font\_path)

except Exception as e:

    print(f"加载字体失败: {e}")

    font = None

*# 设置全局字体配置*

plt.rcParams['font.sans-serif'] = ['OPPOSans'] if font is None else [font.get\_name()]

plt.rcParams['axes.unicode\_minus'] = False  *# 解决负号显示问题*

*# 设置中文字体*

font = FontProperties(fname='C:/Windows/Fonts/simhei.ttf')  *# 确保路径正确*

*# 绘制工作日的食堂就餐时间曲线图*

plt.figure(figsize=(10, 6))

plt.plot(workday\_counts.index, workday\_counts.values, label='工作日', color='blue')

plt.xlabel('小时', fontproperties=font)

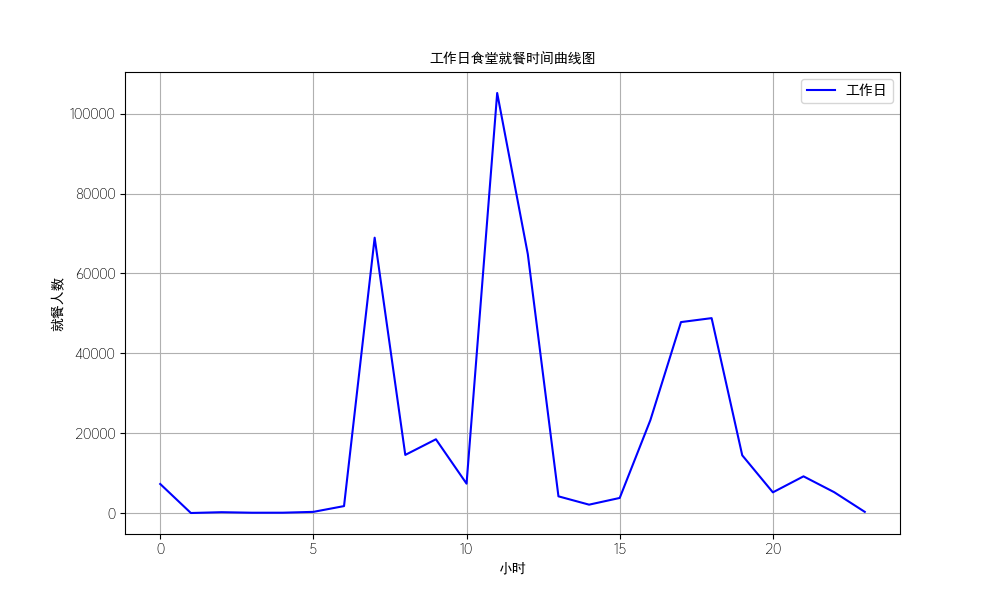
plt.ylabel('就餐人数', fontproperties=font)

plt.title('工作日食堂就餐时间曲线图', fontproperties=font)

plt.legend(prop=font)

plt.grid(True)

plt.show()



绘制非工作日曲线图

*# 绘制非工作日的食堂就餐时间曲线图*

plt.figure(figsize=(10, 6))

plt.plot(non\_workday\_counts.index, non\_workday\_counts.values, label='非工作日', color='red')

plt.xlabel('小时', fontproperties=font)

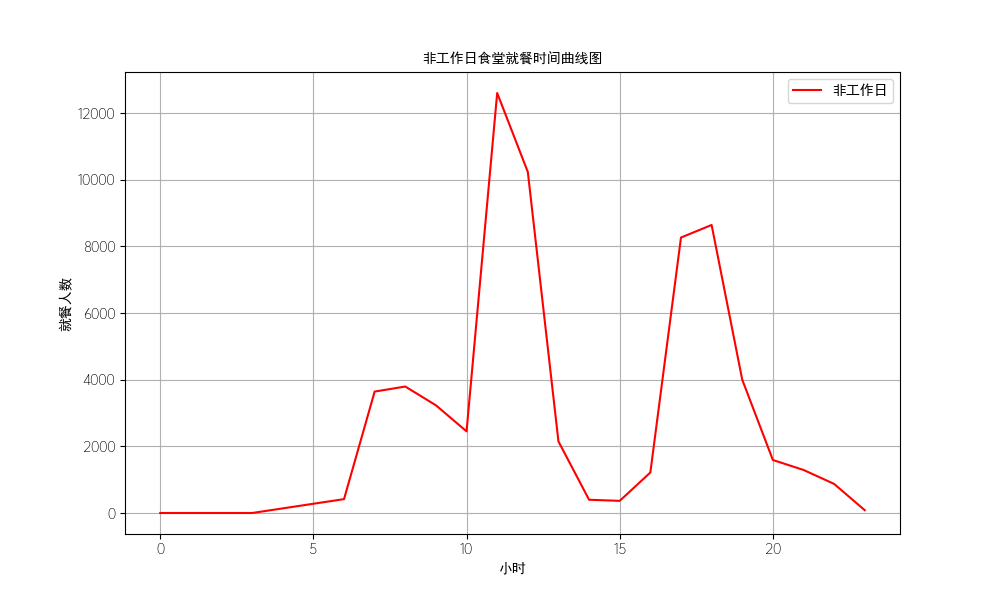
plt.ylabel('就餐人数', fontproperties=font)

plt.title('非工作日食堂就餐时间曲线图', fontproperties=font)

plt.legend(prop=font)

plt.grid(True)

plt.show()



分析就餐峰值：工作日就餐峰值分别在7:00、12:00和17:00左右出现，三个峰值较为平均，中＞早＞晚；非工作日峰值仅在12:00和17:00较为明显，早餐峰值较小，且中＞晚＞＞早。

使用函数定义早中晚的时间段

*# 定义时间段*

def get\_time\_period(hour):

    if 6 <= hour < 10:

        return '早'

    elif 10 <= hour < 17:

        return '中'

    else:

        return '晚'

*# 添加时间段列*

df['时间段'] = df['Date'].dt.hour.apply(get\_time\_period)

将餐厅的数据筛选出来，并按照时间分类

*# 过滤只包含 canteen1 到 canteen5 的记录*

canteens = ['第一食堂', '第二食堂', '第三食堂', '第四食堂', '第五食堂']

df\_filtered = df[df['Dept'].isin(canteens)]

*# 分组统计每个时间段内各个食堂的就餐人数*

time\_period\_canteen\_counts = df\_filtered.groupby(['时间段', 'Dept']).size().unstack(fill\_value=0)

*# 打印统计结果*

print(time\_period\_canteen\_counts)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dept | 第一食堂 | 第三食堂 | 第二食堂 | 第五食堂 | 第四食堂 |
| 时间段 |  |  |  |  |  |
| 中 | 28066 | 32181 | 55280 | 53589 | 35191 |
| 早 | 15572 | 595 | 50482 | 33896 | 135 |
| 晚 | 18452 | 19327 | 49111 | 30130 | 25515 |

*# 绘制柱状图*

time\_period\_canteen\_counts.plot(kind='bar', figsize=(12, 8))

plt.xlabel('时间段', fontproperties=font)

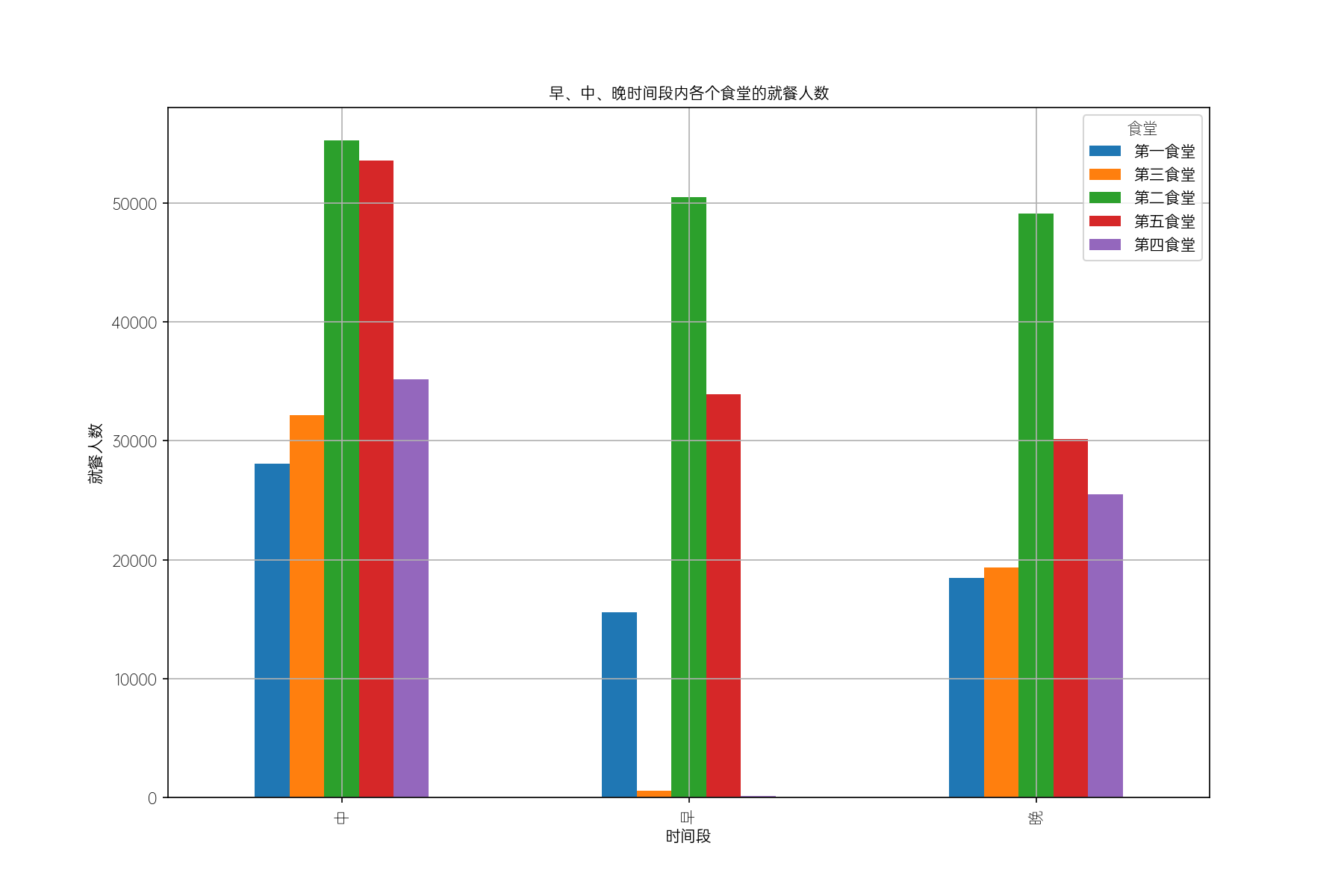
plt.ylabel('就餐人数', fontproperties=font)

plt.title('早、中、晚时间段内各个食堂的就餐人数', fontproperties=font)

plt.legend(title='食堂', prop=font)

plt.grid(True)

plt.show()



早中晚喜欢去的食堂是哪一个：整体来看，学生早餐喜欢去第二食堂和第五食堂；午餐较为平均，第二食堂人数最多，其次为第五食堂、第四食堂、第三食堂；晚餐更偏好第二食堂，其次为第五食堂、第四食堂。

作为经理根据结果提出运营建议：①进行分时段供餐，早餐：提高第二/五食堂出餐效率，其他食堂增设便携早餐窗口；午餐：第二食堂增设分流引导，提高出餐效率，第三/四食堂推出特色套餐形成差异化，吸引学生就餐；晚餐：第四食堂延长供餐时段，第五食堂开发夜宵品类，错峰竞争。②优化空间资源，第二食堂增设打包窗口，第三食堂推出活动吸引学生闲置时段前来自习/活动，提升利用率。

3.学生消费行为分析

3.1根据整体校园消费数据，计算月人均刷卡频次和人均消费额

使用前面已经做好的融合数据，导入

import pandas as pd

import matplotlib.pyplot as plt

from matplotlib.font\_manager import FontProperties

*# 读取CSV文件并显示前几行和列名*

file\_path = r'C:\Users\DELL\Desktop\数字图书馆关键技术期末作业-20250108\学生校园消费行为分析\task1\_2\_1.csv'

*# 尝试使用utf-8编码读取文件*

try:

    df = pd.read\_csv(file\_path, sep=',', encoding='utf-8')

except UnicodeDecodeError:

    print("使用utf-8编码读取文件失败，尝试使用gbk编码")

    df = pd.read\_csv(file\_path, sep=',', encoding='gbk', encoding\_errors='replace')

*# 显示前几行数据*

print(df.head())

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Index\_x | CardNo | Sex | Major | ... | TermSerNo | conOperNo | OperNo | Dept |
| 0 | 1 | 180001 | 男 | 18国际金融 | ... | NaN | 221.085671 | 249 | 第四食堂 |
| 1 | 1 | 180001 | 男 | 18国际金融 | ... | NaN | 221.097654 | 236 | 第一食堂 |
| 2 | 1 | 180001 | 男 | 18国际金融 | ... | NaN | 181.206452 | 18 | 第四食堂 |
| 3 | 1 | 180001 | 男 | 18国际金融 | ... | NaN | 103.935416 | 2 | 第四食堂 |
| 4 | 1 | 180001 | 男 | 18国际金融 | ... | NaN | 103.864524 | 236 | 第一食堂 |

[5 rows x 18 columns]

计算人均消费频次和人均消费额

*# 计算人均刷卡频次(总刷卡次数/学生总人数)*

cost\_count = df['Date'].count()

student\_count = df['CardNo'].nunique()  *# 使用nunique()而不是value\_counts().count()以避免NaN问题*

average\_cost\_count = int(round(cost\_count / student\_count))

print("人均刷卡频次:", average\_cost\_count)

*# 计算人均消费额(总消费金额/学生总人数)*

cost\_sum = df['Money'].sum()

average\_cost\_money = int(round(cost\_sum / student\_count))

print("人均消费额:", average\_cost\_money)

人均刷卡频次: 77

人均消费额: 278

将数据按照专业和性别进行分类。

*# 设置字体*

font\_path = r"C:\Users\DELL\Desktop\数字图书馆关键技术期末作业-20250108\OPPOSans-R.ttf"

try:

    font = FontProperties(fname=font\_path)

except Exception as e:

    print(f"加载字体失败: {e}")

    font = None

*# 设置全局字体配置*

plt.rcParams['font.sans-serif'] = ['OPPOSans'] if font is None else [font.get\_name()]

plt.rcParams['axes.unicode\_minus'] = False  *# 解决负号显示问题*

*# 设置中文字体*

font = FontProperties(fname='C:/Windows/Fonts/simhei.ttf')

*# 提高分辨率*

plt.rcParams['figure.dpi'] = 150

*# 计算不同专业、不同性别人均消费*

data\_gb1 = df['Money'].groupby(df['Major']).mean().reset\_index()

data\_gb2 = df['Money'].groupby([df['Sex'], df['Major']]).mean().reset\_index()

data\_boy = data\_gb2[data\_gb2['Sex'] == '男']

data\_girl = data\_gb2[data\_gb2['Sex'] == '女']

选择3个专业，分析不同专业间不同性别学生群体的消费特点并绘图。此处选取了18国际金融、18计算机应用、18建筑工程三个专业。

*# 分别选取了18国际金融、18计算机应用、18建筑工程三个专业。*

major = ['18国际金融','18计算机应用','18建筑工程']

data\_gb3 = data\_gb1.loc[data\_gb1['Major'].isin(major)]

data\_boy1 = data\_boy.loc[data\_boy['Major'].isin(major)]

data\_girl1 = data\_girl.loc[data\_girl['Major'].isin(major)]

*# 绘制三个专业、不同性别人均消费柱状图*

plt.rcParams['font.sans-serif'] = 'SimHei'  *# 设置中文显示*

p = plt.figure(figsize = (12,6))  *# 将画布设定为正方形，则绘制的饼图是正圆*

ax1 = p.add\_subplot(1,3,1)

plt.bar(data\_gb3['Major'], data\_gb3['Money'])

plt.title('不同专业人均消费柱状图', size=20)  *# 绘制标题*

plt.xticks(rotation=90, size=14)

ax2 = p.add\_subplot(1,3,2)

plt.bar(data\_boy1['Major'], data\_boy1['Money'])

plt.title('男生人均消费柱状图', size=20)  *# 绘制标题*

plt.xticks(rotation=90, size=14)

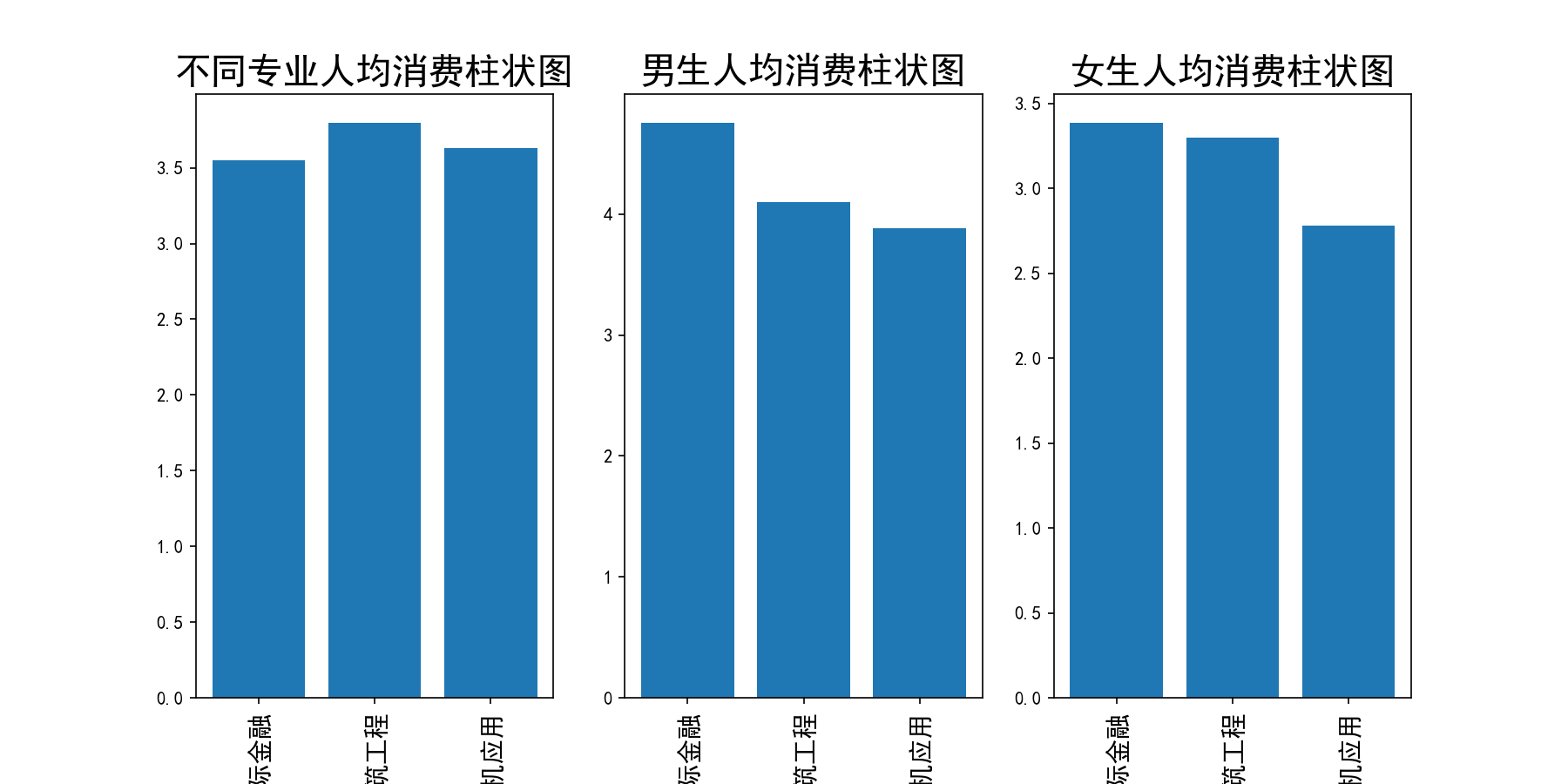
ax3 = p.add\_subplot(1,3,3)

plt.bar(data\_girl1['Major'], data\_girl1['Money'])

plt.title('女生人均消费柱状图', size=20)  *# 绘制标题*

plt.xticks(rotation=90, size=14)

plt.show()



结论：18建筑工程专业的学生整体消费水平高于18计算机应用机应用专业，计算机应用专业的消费基本与18国际金融的消费持平。在同级男生中，国际金融的消费高于建筑工程专业，建筑工程专业高于计算机应用专业。专业间同级女生消费趋势与同级男生类似，但女生消费整体水平低于男生，同时组间差距较小。

3.2以人均消费额为分界线，将学生群体划分为两个群体，通过对比本月这两个群体在不同消费地点所消费消费额的饼图，分析群体消费特点。

计算人均消费额

*# 计算每个学生的总消费金额*

student\_total\_cost = df.groupby('CardNo')['Money'].sum()

*# 计算每个学生的平均消费金额*

student\_average\_cost = student\_total\_cost / df.groupby('CardNo')['Date'].count()

*# 打印前几行的平均消费金额*

print(student\_average\_cost.head())

*# 打印平均消费金额的描述性统计信息*

print(student\_average\_cost.describe())

*# 定义人均消费额的界限*

threshold = student\_average\_cost.median()  *# 使用中位数作为阈值*

*# 打印阈值*

print("调整后的阈值:", threshold)

CardNo

180001 4.617143

180002 2.476596

180004 5.580000

180005 5.141667

180006 3.726087

dtype: float64

count 3268.000000

mean 3.807654

std 1.237508

min 0.200000

25% 2.906785

50% 3.677930

75% 4.631485

max 10.500000

dtype: float64

调整后的阈值: 3.6779299847792997

划分出两个消费群体，并统计

*# 划分两个群体*

high\_spending = df[df['CardNo'].isin(student\_average\_cost[student\_average\_cost > threshold].index)]

low\_spending = df[df['CardNo'].isin(student\_average\_cost[student\_average\_cost <= threshold].index)]

*# 打印高消费群体和低消费群体的样本数量*

print("高消费群体样本数量:", len(high\_spending['CardNo'].unique()))

print("低消费群体样本数量:", len(low\_spending['CardNo'].unique()))

*# 打印高消费群体和低消费群体的平均消费金额*

print("高消费群体平均消费金额:")

print(student\_average\_cost[student\_average\_cost > threshold].describe())

print("低消费群体平均消费金额:")

print(student\_average\_cost[student\_average\_cost <= threshold].describe())

*# 统计高消费群体在不同消费地点的消费次数*

high\_spending\_counts = high\_spending['Dept'].value\_counts()

*# 统计低消费群体在不同消费地点的消费次数*

low\_spending\_counts = low\_spending['Dept'].value\_counts()

*# 检查数据是否为空*

if high\_spending\_counts.empty:

    print("高消费群体数据为空")

else:

    print("高消费群体消费地点分布:")

    print(high\_spending\_counts)

if low\_spending\_counts.empty:

    print("低消费群体数据为空")

else:

    print("低消费群体消费地点分布:")

    print(low\_spending\_counts)

高消费群体样本数量: 1634

低消费群体样本数量: 1634

高消费群体平均消费金额:

count 1634.000000

mean 4.802271

std 0.868907

min 3.678082

25% 4.147516

50% 4.631935

75% 5.269250

max 10.500000

dtype: float64

低消费群体平均消费金额:

count 1634.000000

mean 2.813036

std 0.573630

min 0.200000

25% 2.439830

50% 2.906721

75% 3.265249

max 3.677778

dtype: float64

高消费群体消费地点分布:

Dept

第二食堂 24739

第五食堂 22943

第四食堂 21243

第三食堂 13871

第一食堂 10974

好利来食品店 8114

财务处 4585

红太阳超市 3431

教师食堂 1343

水电缴费处 577

医务室 192

第二图书馆 61

第一图书馆 55

基础课部 50

人文社科 41

自然科学书库 35

第七教学楼 35

第五教学楼 26

第六教学楼 22

飞凤轩宿管办 22

第四教学楼 16

财务部 14

宿管办 13

第一教学楼 12

第三教学楼 5

机电系 2

第二教学楼 2

外语系 1

Name: count, dtype: int64

低消费群体消费地点分布:

Dept

第二食堂 51469

第五食堂 34784

第一食堂 20106

第三食堂 9600

第四食堂 9516

好利来食品店 5578

财务处 3821

红太阳超市 3044

水电缴费处 943

医务室 183

第二图书馆 84

第一图书馆 71

自然科学书库 52

飞凤轩宿管办 35

财务部 32

第四教学楼 27

人文社科 23

第七教学楼 22

基础课部 21

第五教学楼 21

宿管办 21

第六教学楼 20

第三教学楼 10

第一教学楼 5

第二教学楼 4

机电系 1

青鸾苑宿管办 1

工商系部 1

Name: count, dtype: int64

分析群体消费特点，绘制饼图

*# 设置字体*

font\_path = r"C:\Users\DELL\Desktop\数字图书馆关键技术期末作业-20250108\OPPOSans-R.ttf"

try:

    font = FontProperties(fname=font\_path)

except Exception as e:

    print(f"加载字体失败: {e}")

    font = None

*# 设置全局字体配置*

plt.rcParams['font.sans-serif'] = ['OPPOSans'] if font is None else [font.get\_name()]

plt.rcParams['axes.unicode\_minus'] = False  *# 解决负号显示问题*

*# 设置中文字体*

font = FontProperties(fname='C:/Windows/Fonts/simhei.ttf')

*# 提高分辨率*

plt.rcParams['figure.dpi'] = 150

*# 创建画布*

fig, axes = plt.subplots(1, 2, figsize=(20, 10))

*# 绘制高消费群体的消费地点分布（饼图）*

if not high\_spending\_counts.empty:

    high\_spending\_counts.plot(kind='pie', ax=axes[0], autopct='%1.1f%%', startangle=90, colors=['skyblue', 'lightblue', 'lightgreen', 'lightcoral', 'lightsalmon'])

    axes[0].set\_title("高消费群体消费地点分布", fontproperties=font, fontsize=20)

    axes[0].set\_ylabel("")  *# 去掉默认的y轴标签*

else:

    axes[0].text(0.5, 0.5, "高消费群体数据为空", fontsize=20, ha='center', va='center')

*# 绘制低消费群体的消费地点分布（饼图）*

if not low\_spending\_counts.empty:

    low\_spending\_counts.plot(kind='pie', ax=axes[1], autopct='%1.1f%%', startangle=90, colors=['salmon', 'lightcoral', 'lightsalmon', 'lightpink', 'lightgrey'])

    axes[1].set\_title("低消费群体消费地点分布", fontproperties=font, fontsize=20)

    axes[1].set\_ylabel("")  *# 去掉默认的y轴标签*

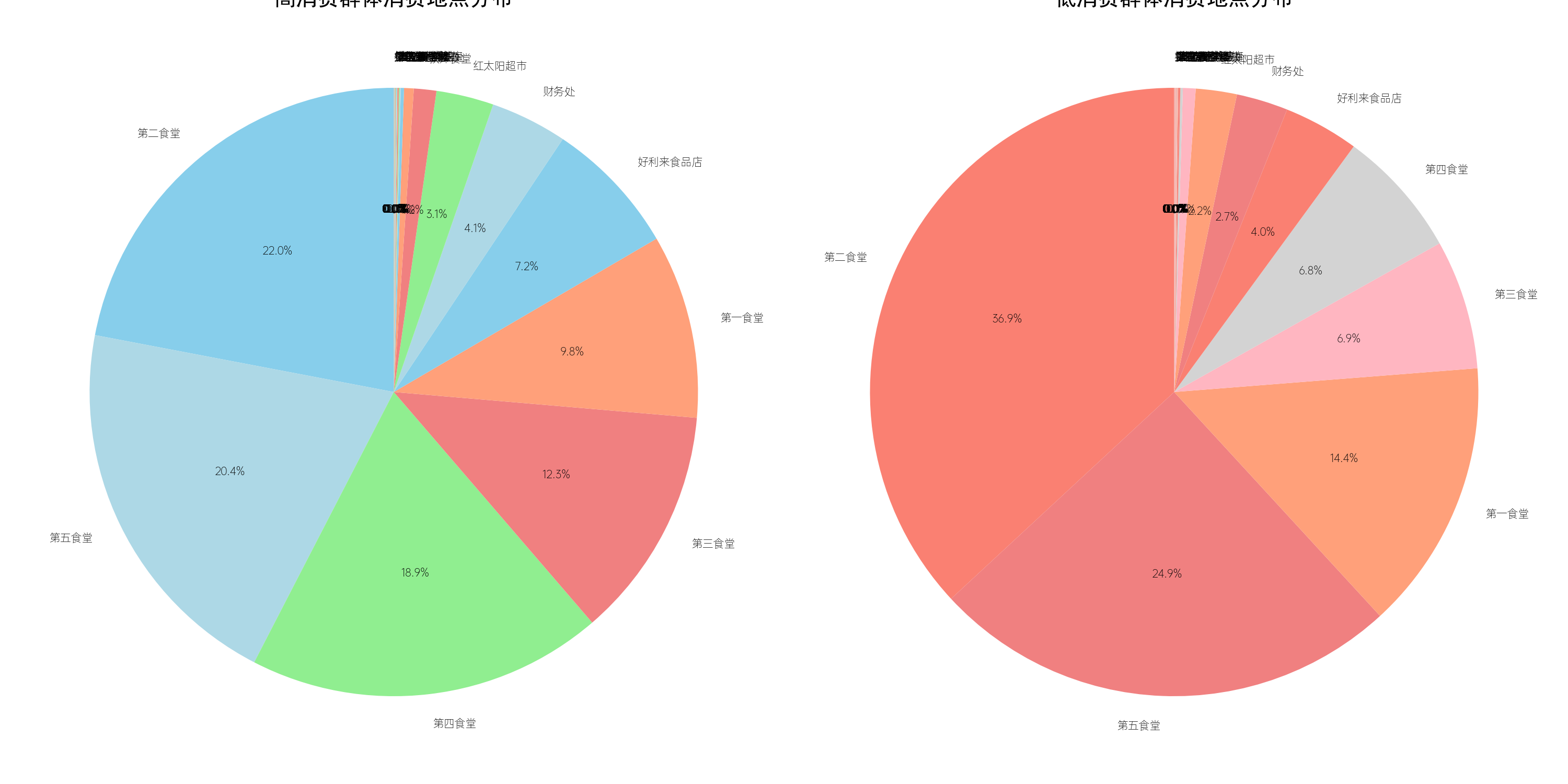
else:

    axes[1].text(0.5, 0.5, "低消费群体数据为空", fontsize=20, ha='center', va='center')

*# 显示图像*

plt.tight\_layout()

plt.show()



高消费群体的消费对象丰富，各食堂占比较为平均，同时超市、食品店消费也具有一定比例。低消费学生群体的消费主要集中在各个食堂，其中又以第二食堂与第五食堂为主，食堂外的消费较少。

3.3对低消费学生群体的行为进行分析，探讨是否存在特征，作为助学金评定的参考

*# 打印低消费群体的平均消费金额描述性统计信息*

print("低消费群体平均消费金额:")

print(student\_average\_cost[student\_average\_cost <= threshold].describe())

低消费群体平均消费金额:

count 1634.000000

mean 2.813036

std 0.573630

min 0.200000

25% 2.439830

50% 2.906721

75% 3.265249

max 3.677778

dtype: float64

low\_spending['Date'] = pd.to\_datetime(low\_spending['Date'])

*# 分析低消费群体的消费时间段分布*

low\_spending['hour'] = low\_spending['Date'].dt.hour

low\_spending\_hourly = low\_spending.groupby('hour')['Money'].sum().reset\_index()

plt.figure(figsize=(10, 6))

sns.barplot(x='hour', y='Money', data=low\_spending\_hourly)

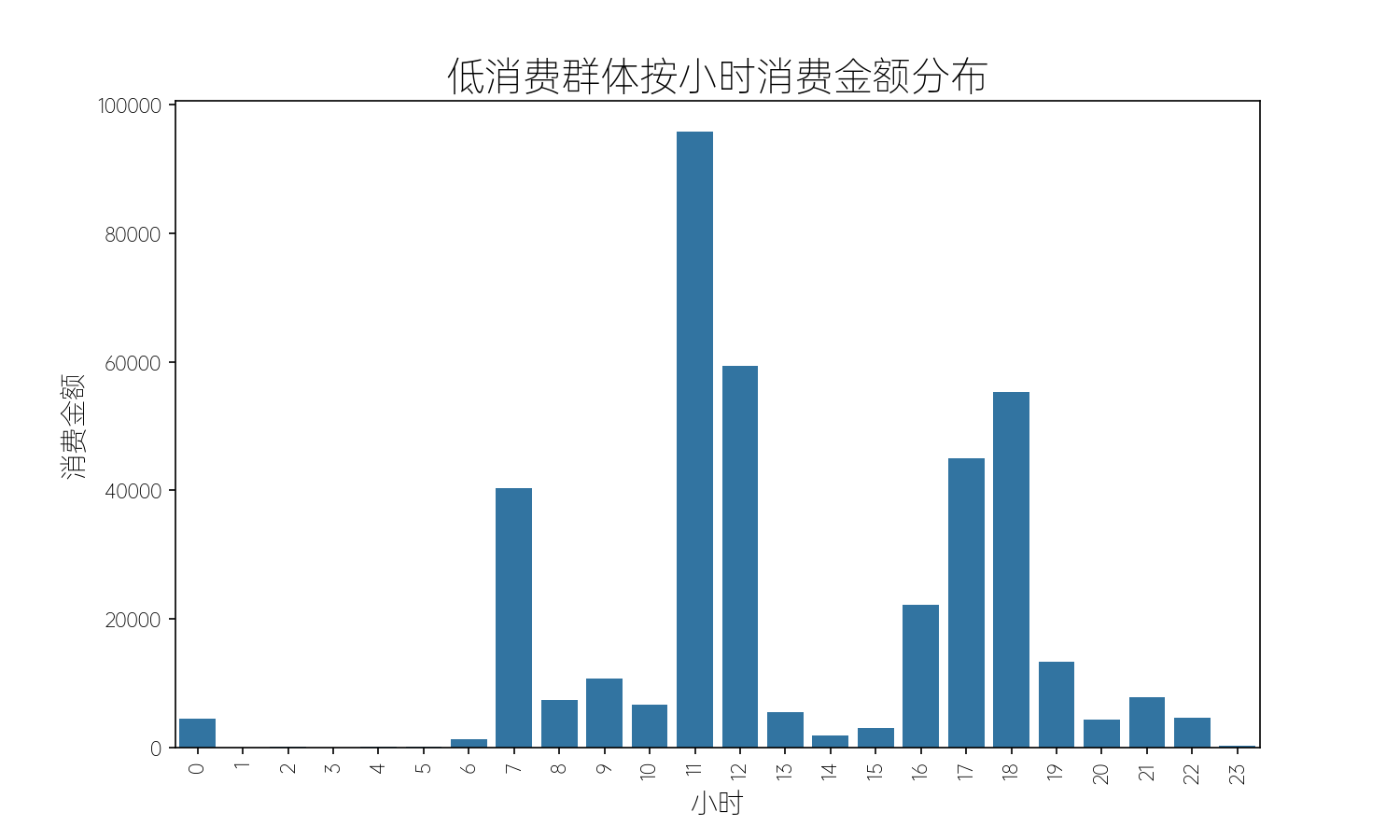
plt.title('低消费群体按小时消费金额分布', fontsize=20)

plt.xlabel('小时', fontsize=14)

plt.ylabel('消费金额', fontsize=14)

plt.xticks(rotation=90)

plt.show()



可以看出，消费时间段集中在早上7点、中午11-12点、傍晚16-18点。

*# 分析低消费群体的月就餐次数*

low\_spending\_monthly = low\_spending.groupby('CardNo')['Date'].count().reset\_index()

low\_spending\_monthly.columns = ['CardNo', '月就餐次数']

plt.figure(figsize=(10, 6))

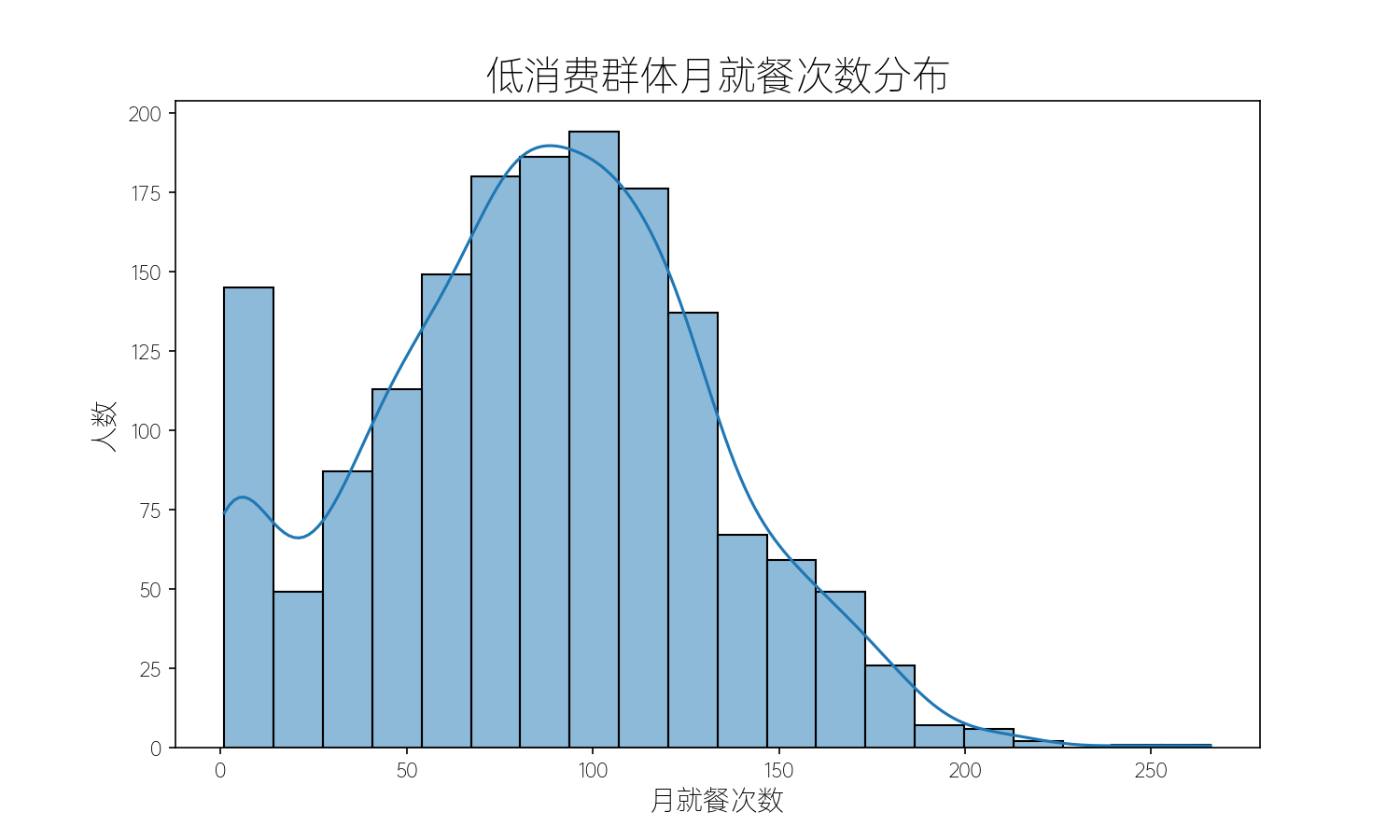
sns.histplot(data=low\_spending\_monthly, x='月就餐次数', bins=20, kde=True)

plt.title('低消费群体月就餐次数分布', fontsize=20)

plt.xlabel('月就餐次数', fontsize=14)

plt.ylabel('人数', fontsize=14)

plt.show()



可以看出，低消费群体月就餐次数峰值为100次左右。

*# 分析低消费群体的性别和专业差异*

low\_spending\_gender\_major = low\_spending.groupby(['Sex', 'Major'])['Money'].mean().reset\_index()

plt.figure(figsize=(12, 8))

sns.barplot(data=low\_spending\_gender\_major, x='Major', y='Money', hue='Sex')

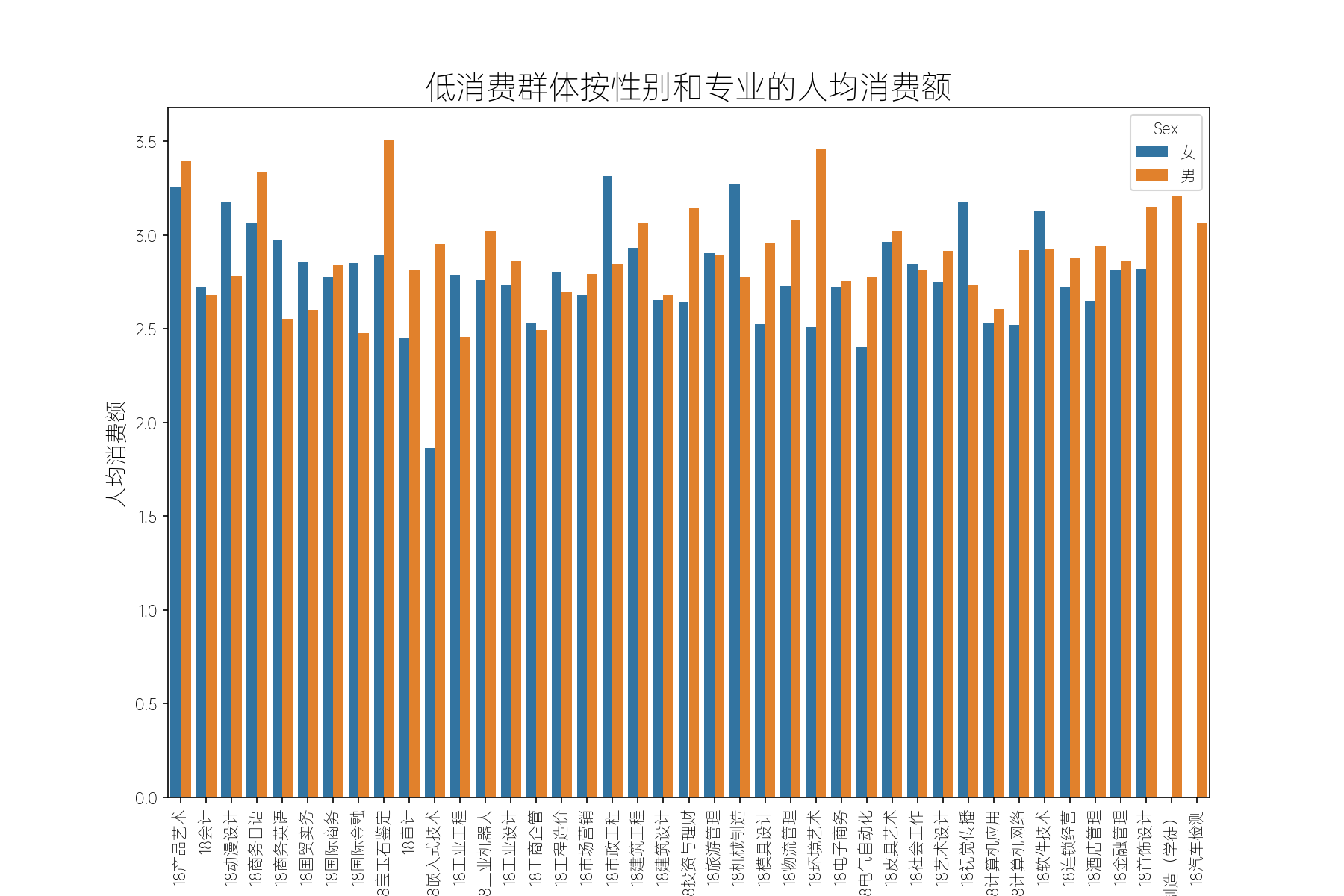
plt.title('低消费群体按性别和专业的人均消费额', fontsize=20)

plt.xlabel('专业', fontsize=14)

plt.ylabel('人均消费额', fontsize=14)

plt.xticks(rotation=90)

plt.show()



低消费学生群体的消费总额低于人均线，消费结构单一，主要集中在饮食领域。中位数（2.91）接近均值（2.81）：群体消费水平集中在中等偏低区间，无明显两极分化平均消费金额较低，且消费金额的波动较小。消费地点较为集中，可能更多地集中在价格较低的消费场所。消费时间段集中在早上7点、中午11-12点、傍晚16-18点，显示出他们在某些时间段（如午餐或晚餐）有较高的消费频率。性别和专业差异对消费存在一定影响，部分专业和性别的学生更可能属于低消费群体。