

**课程设计报告**

( 2024--2025年度 第一学期 )

课程名称: python高级应用

院 系: 软件学院

专业班级: 2022级软件工程(X)班

学 号:

学生姓名:

**基于天猫订单数据可视化分析**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **一** | **二** | **三** | **四** | **五** | **总分** |
|  |  |  |  |  |  |

**一、****选题的背景介绍（10分）（标题的字体四号黑体，不用缩进）**

为什么要选择该主题？要达到的数据分析目标是什么？从社会、经济、技术、

数据来源等方面进行描述。（不少于200字）

1. **社会背景**：
   * 电子商务的快速发展使得平台订单数据具有重要的研究价值。
   * 天猫作为中国领先的电商平台，其数据能够反映消费者行为、市场趋势等内容。
   * 对电商订单数据的分析可以帮助优化消费者体验，促进社会经济发展。
2. **经济意义**：
   * 分析天猫订单数据能够为商家提供运营指导，例如优化库存管理、促销策略等。
   * 通过挖掘消费数据，发现潜在的市场增长点和问题，提高整体经济效益。
3. **技术角度**：
   * 数据分析与可视化是现代企业分析的重要手段，可以将复杂的电商数据转化为清晰的商业洞察。
   * 通过技术手段处理和展示订单数据，提升对商业模式的理解能力。
4. **数据价值**：
   * 天猫平台积累了海量的用户订单数据，通过数据挖掘，可以深入了解消费者行为偏好。
   * 这些数据对于研究消费趋势、地域分布、时间规律等方面具有很高的价值。

分析目标：

研究天猫平台的订单数据，包括销量、品类和用户购买行为等内容。

为商家和平台提供决策依据，支持其优化运营和提升用户体验。

**二、大数据分析设计方案（15 分）**

1.本数据集的数据内容与数据特征分析

**数据内容**：

* 数据集来源于天猫平台，包含订单交易相关的信息，主要字段有：
  1. **订单编号**：唯一标识每个订单。
  2. **支付金额**：订单的支付金额。
  3. **下单时间**：用户创建订单的时间。
  4. **成交时间**：订单完成交易的时间。
  5. **用户行为**：可能涉及订单支付、取消、退款等行为信息。
  6. **地域信息**：订单发生的地区。

**数据特征分析**：

* **时间特征**：
  + 数据中包含订单的时间维度（如日期、小时等），可分析订单的时间分布规律。
  + 时间字段可帮助揭示不同时间段的消费高峰、季节性趋势等。
* **金额特征**：
  + 支付金额是订单的核心特征，用于分析成交额、单均消费额等。
  + 结合地域、时间特征，进一步分析不同区域的消费能力或销售热点。
* **用户行为特征**：
  + 用户从下单到支付、从支付到完成的行为数据反映用户转化效率。
  + 转化率可以通过销售漏斗分析，发现订单创建到成交中的问题。
* **地域特征**：
  + 地区字段反映了不同地域用户的购买行为和消费能力。
  + 可挖掘高消费地区或市场潜力较大的区域。
* **缺失值与异常值**：
  + 数据集中可能存在缺失值（如未完成的订单没有成交时间）。
  + 异常值（如负值或极大的支付金额）需要识别和处理。

2.数据分析的课程设计方案概述

1. **数据预处理**：
   * 导入数据集并检查字段类型和数据完整性。
   * 处理缺失值和异常值，例如对空值进行填补或剔除。
   * 转换时间字段为标准的时间格式，并提取年月日、小时等特征。
   * 数据分类和排序（如按时间、金额、地域等分组），为后续分析打好基础。
2. **分析维度**：
   * **时间维度分析**：
     + 分析订单量和成交金额随时间（按月、按日、按小时）的变化趋势。
     + 绘制成交金额与时间的折线图，识别高峰期与低谷期。
   * **地域维度分析**：
     + 分析不同地区的订单量和成交金额，绘制区域热力图。
     + 找出消费能力最强和最弱的地区。
   * **用户行为分析**：
     + 研究用户订单支付、取消、退款的行为模式。
     + 计算订单支付到成交的转化率，定位关键问题。
   * **销售漏斗分析**：
     + 绘制漏斗图，展示订单创建到全额成交的转化过程，分析各环节的用户流失情况。
3. **可视化设计**：
   * 利用Python中的Matplotlib、Seaborn等库生成可视化图表，包括折线图、柱状图、热力图、漏斗图等。
   * 用图表展示数据分布和趋势，直观传递分析结果。

**课程设计实现工具**：

* **数据处理**：Pandas、Numpy。
* **可视化工具**：Matplotlib、Seaborn。

**三、数据分析过程与步骤（55 分）**

1.数据源描述 （5分）

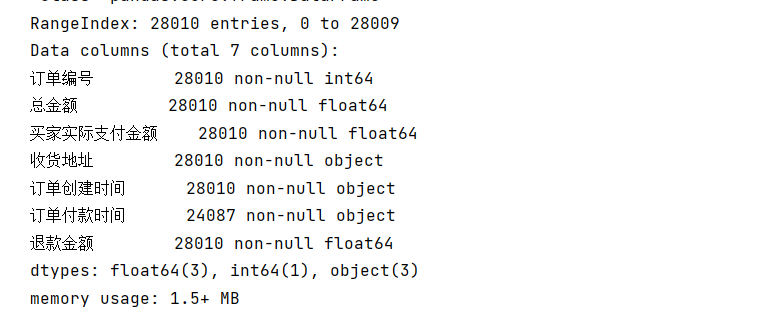
和鲸社区获取的数据集

2.数据清洗 （15分）

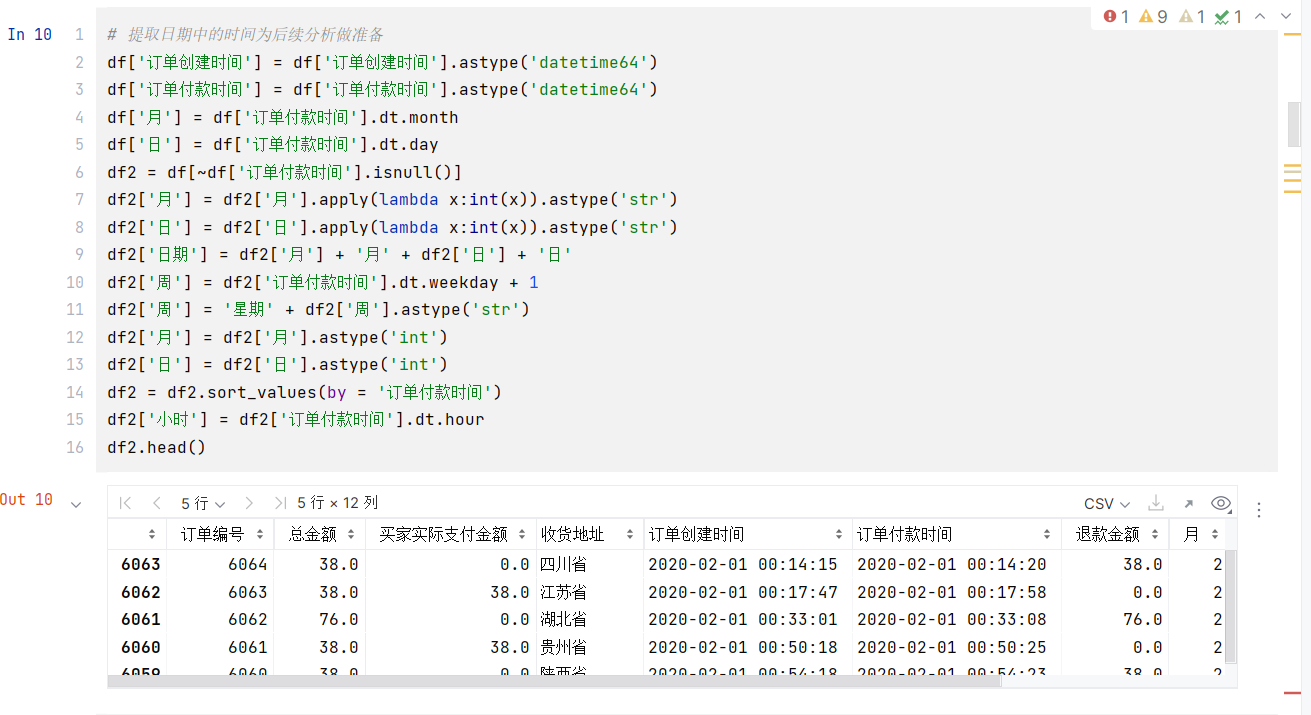
1、数据导入和清洗



数据集基本描述

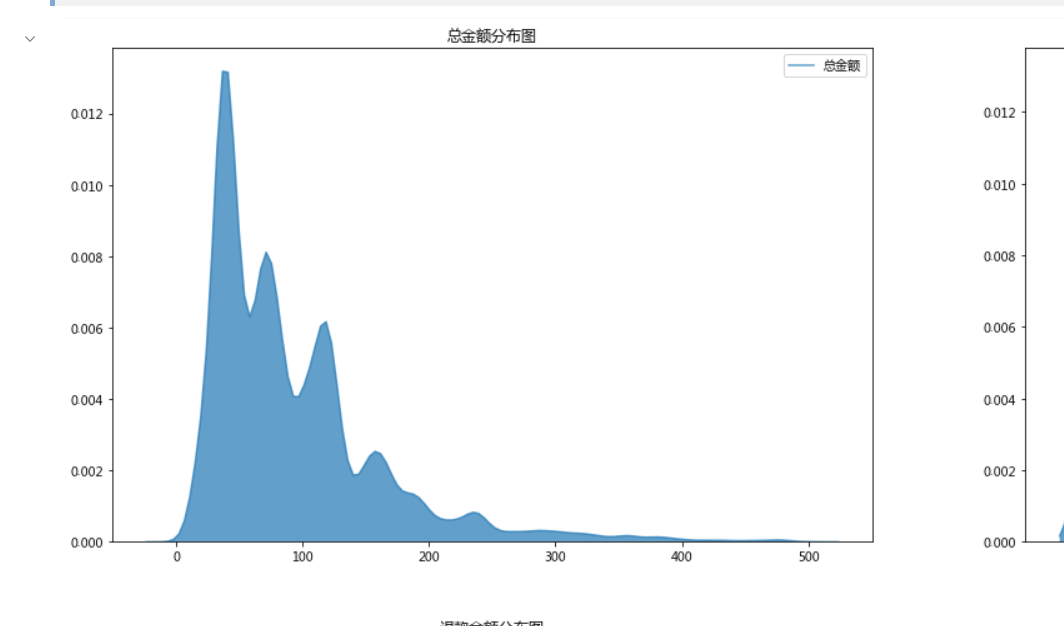


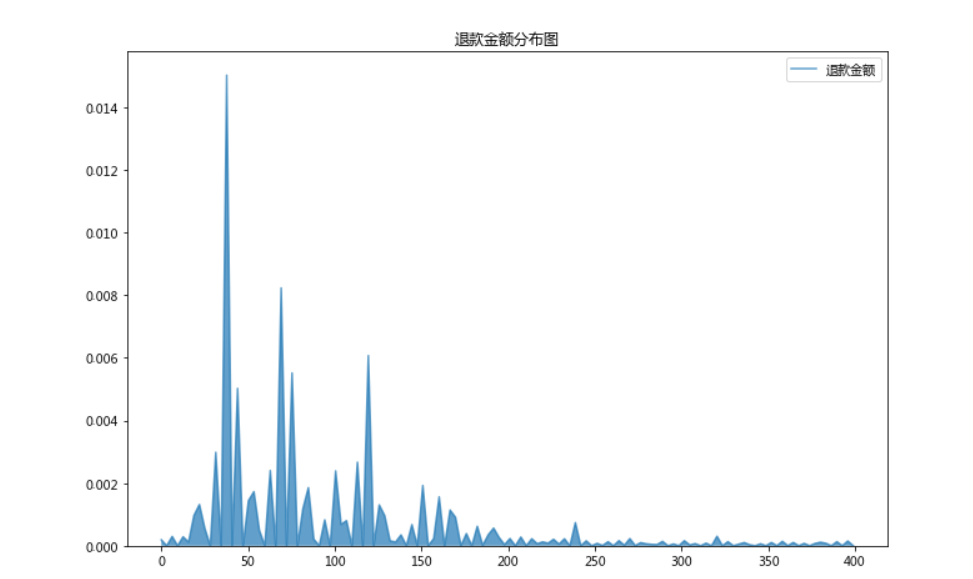
2、按时间段提取数据，为后面分析做准备



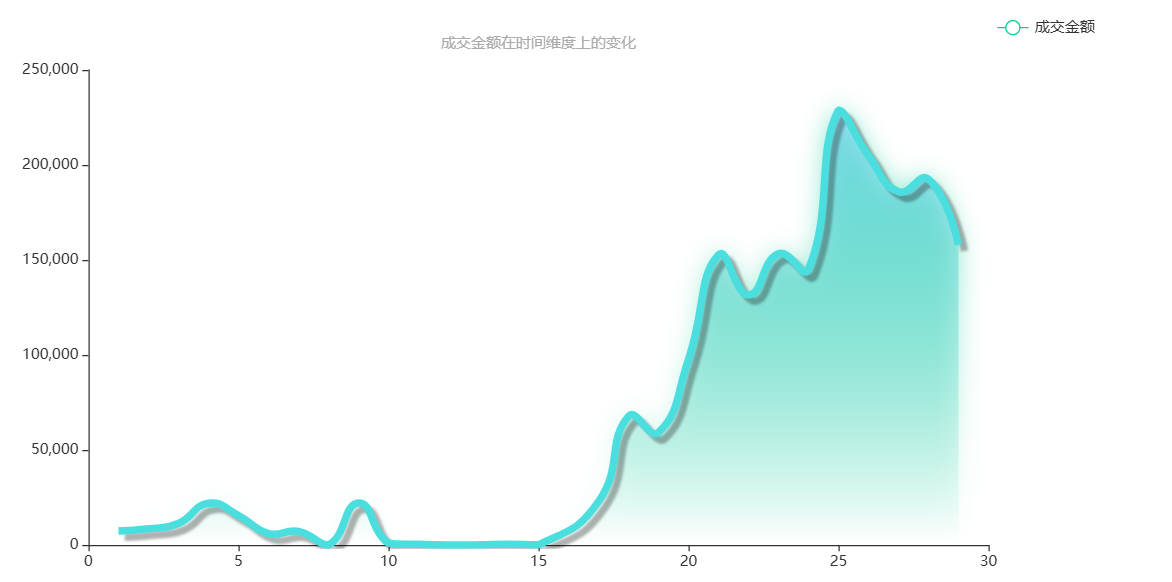
3.数据分析及可视化（35分）

数据分布情况



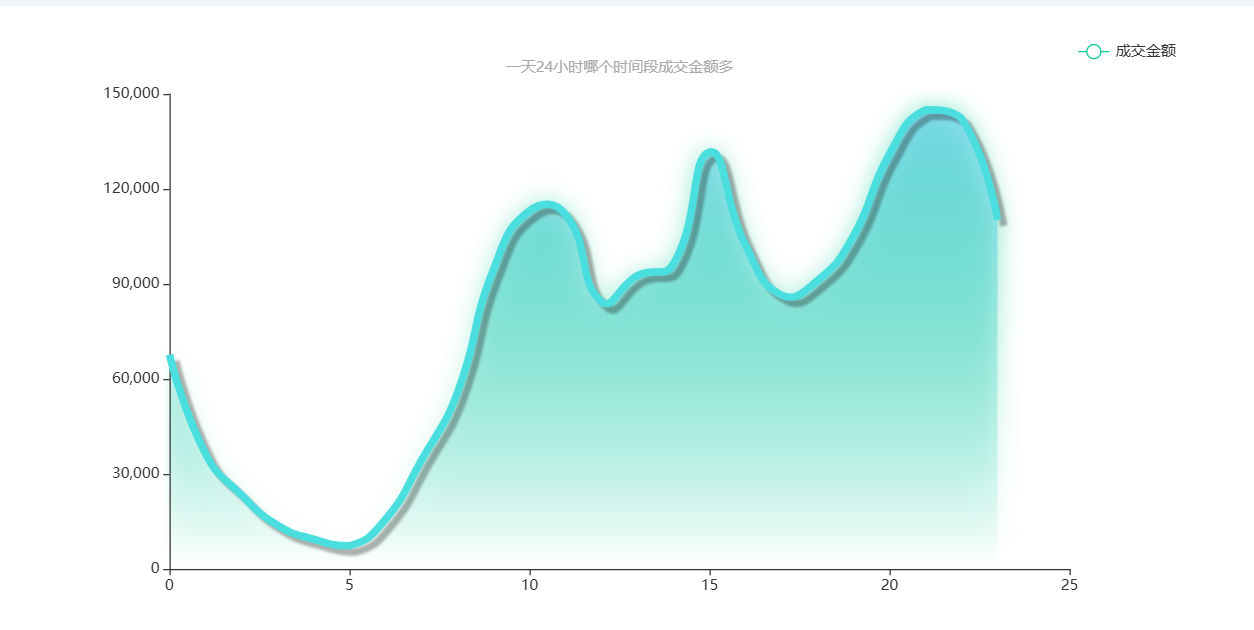


2、成交金额在时间维度上的变化



绘制成交金额在时间维度上的变化图，可以看到在2020年2月份的天猫成交金额在15日以前都是呈现一个较低的状态，而从15日开始，出现一个稳步上升的趋势，并且在22日和25日的时候分别达到了高峰。2022年的1月25日是春节，因此怀疑导致成交金额如此低下的原因，可能是受到春节影响，大部分消费者和商家都在停业过春节，而随着春节的结束，大部分人都开始逐渐进行了复工，因此从15日开始，成交金额开始逐步恢复正常，并且稳步提升。

1. 每天各时段成交金额变化



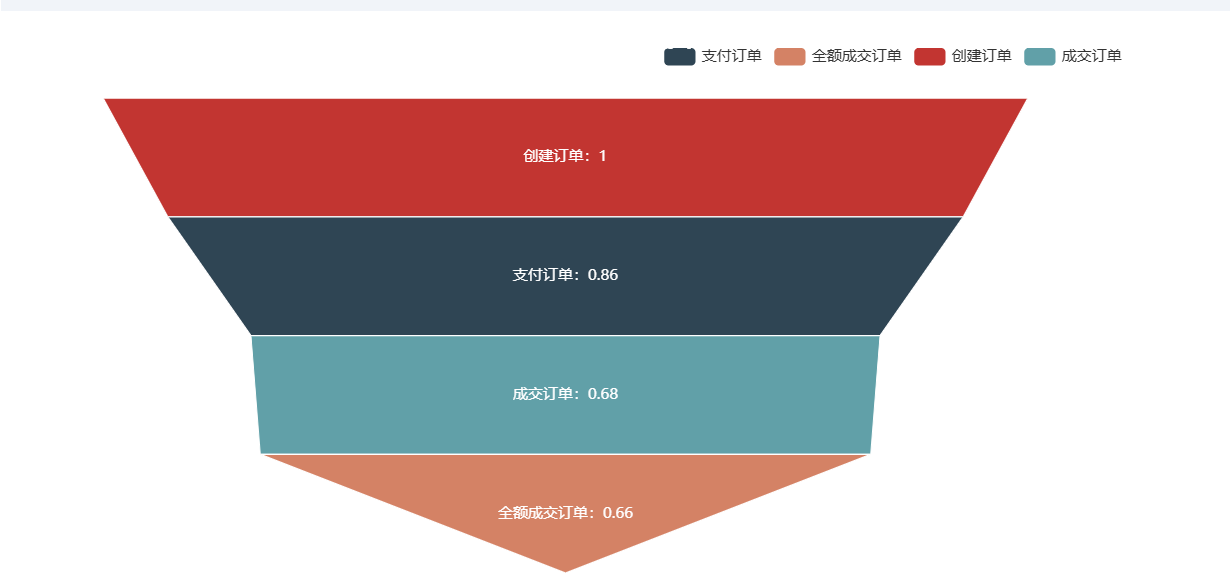
绘制每天各时段的成交金额变化图，可以发现，每天的早上10点，下午的15点，晚上的21-23点的时间段，是成交金额的高峰期。尤其是晚上这个时间段，成交金额相对来说是最高的。这可能因为人在晚上是比较容易冲动消费的，也可能因为大家再进行了一整天的工作后，只有在晚上吃完饭，忙完所有工作以后，才有时间来天猫逛一逛，进行一些网购等。而10点和15点通常也都是大家吃完早餐和午餐以后，进行了一段时间的工作以后，稍微休息一下，进行一些网购行为（猜的奥~）。

1. 成交金额在地区维度上的分布



绘制成交金额在全国各地的分布，我们可以发现江浙沪一带、以及北京、广东、四川等地区的成交金额是比较高的，而新疆西藏内蒙古等其他地区的成交金额普遍较低。北上广深等一线城市本身就是人口较多、经济较发达的城市，所以成交金额自然也会较高。

1. 天猫2月销售订单转化率斗图



1. 通过销售额漏斗图可以发现，成交订单到全额成交订单的转化率还是蛮高的，只下降了2个百分点，这说明通常来说只要订单成交了，那么基本上就是进行的全额成交，只会有小部分消费者有部分退款行为。
2. 创建订单到支付订单的转化率有86%，这说明有14%的用户在创建了订单以后，没有选择支付，而是取消了该笔订单的支付。应当重点追溯一下导致这个情况的原因是什么。可以查询创建订单以后的用户行为都有哪些，创建了订单以后，用户是立即取消了订单支付？还是用户离开了支付界面进行了其他行为？用户离开支付界面，都进行了哪些行为，是继续浏览同类商品？还是退出app忘记支付？是因为看到了其他同类更好的商品决定不购买该商品了？还是只是点错了，不小心进到了支付界面？还是想要购买的商品规格不符合自己的预期？提升创建订单到支付订单的转化率是较为关键的。
3. 支付订单到成交订单的转化率同样比较低，转换率下降了18%。这就说明，有18% 的用户在支付了订单金额以后，选择了全额退款。这个问题也不容小觑。应当查明导致该问题的原因。是购买的产品规格不符合预期需要重新下单？还是收到了商品以后发现有质量问题，需要全额退款？还是因为物流速度太慢，导致的全额退款行为？在整个销售漏斗中，这部分的转化率是最低的，因此应当结合其他数据，着重查找该环节的原因，从而提升销售额。

**四、总结（10分）**

通过数据的多维分析，我们发现天猫平台的消费行为主要受到以下因素影响：

1. 时间因素：节假日和时间高峰对消费行为具有显著影响。
2. 地域因素：经济发达地区贡献了主要销售额，但三四线城市具有增长潜力。
3. 用户行为因素：支付和退款环节是销售漏斗的关键节点，需要进一步优化。

通过时间、地域、用户行为等多个维度的分析，明确了市场趋势、消费习惯及问题环节。为商家优化库存、设计促销策略、提高订单转化率提供了数据支持。为电商平台在精准营销、提升用户体验、改进销售流程等方面提供了科学依据。

**五、****附录（10分）**

**（源代码中文字体宋体，西文字体Times New Roman，小四号，行距固定值18磅）**

import pandas as pd  
import numpy as np  
from pyecharts.charts import Pie,Bar,Line,Map,Map3D,Funnel  
from pyecharts import options as opts  
import matplotlib.pyplot as plt  
import warnings  
import seaborn as sns  
from pyecharts.commons.utils import JsCode  
from pyecharts.globals import ThemeType, ChartType  
import textwrap  
  
*# 字体设置*plt.rcParams['font.sans-serif']=['Microsoft YaHei']   
plt.rcParams['axes.unicode\_minus']=False  
plt.rc('font',family = 'Microsoft YaHei',size = '15')  
warnings.filterwarnings("ignore")  
  
%matplotlib inline

df = pd.read\_csv('tmall\_order\_report.csv')  
df.head(10) *#测试导入结果，显示前10行数据*

*# 去除字段名中的空格*new\_columns = [col.strip() for col in df.columns]  
df.columns = new\_columns  
*# 显示 dataframe 信息*df.info()

*# 数据基本描述*print('数据的时间区间为',df['订单创建时间'].min(),'到',df['订单创建时间'].max())  
print('收货地址总计有：',df['收货地址'].nunique(),'个')  
df.describe()

*# 确保 '订单创建时间' 和 '订单付款时间' 列以正确的 datetime64[ns] 格式进行转换*df['订单创建时间'] = pd.to\_datetime(df['订单创建时间'])  
df['订单付款时间'] = pd.to\_datetime(df['订单付款时间'])  
  
*# 提取月份和日期*df['月'] = df['订单付款时间'].dt.month  
df['日'] = df['订单付款时间'].dt.day  
  
*# 过滤掉 '订单付款时间' 为空的行*df2 = df.dropna(subset=['订单付款时间'])  
  
*# 将月份和日期转换为字符串并添加中文字符*df2['日期'] = df2['月'].astype(str) + '月' + df2['日'].astype(str) + '日'  
  
*# 获取星期几，并将其转换为中文表示*df2['周'] = '星期' + (df2['订单付款时间'].dt.weekday + 1).astype(str)  
  
*# 确保 '月' 和 '日' 是整数类型*df2['月'] = df2['月'].astype(int)  
df2['日'] = df2['日'].astype(int)  
  
*# 按 '订单付款时间' 排序*df2 = df2.sort\_values(by='订单付款时间')  
  
*# 提取小时*df2['小时'] = df2['订单付款时间'].dt.hour  
  
*# 查看前几行*df2.head()

*# 查看收货地址信息*df2.收货地址.unique()

df2['收货地址'] = df2.收货地址.apply(lambda x:x.strip('省|自治区'))  
df2['收货地址'] = df2.收货地址.replace(['新疆维吾尔','广西壮族','宁夏回族'],['新疆','广西','宁夏'])  
df2.head()  
df2.收货地址.unique()

*# 查看缺失数据*df[df['订单付款时间'].isnull()].head()

*# 查看是否有重复值*df[df['退款金额'] > df['总金额']]  
print('重复值数量为：',df.duplicated().sum())

def kde\_plot\_array(df):  
 *"""  
 绘制概率密度图矩阵函数  
 df:要绘制图像的dataframe  
 绘制各个字段的概率密度分布，最终返回图像的show()  
 """* plt.figure(figsize = (24,16)) *# 图一figure* for num,columns in zip(range(len(df.columns)),df.columns):  
 plt.subplot(round(len(df.columns)/2,0),2,num+1)  
 *# sns.set(font = 'SimHei',font\_scale = 1.6)* index = columns  
 sns.kdeplot(df[columns],  
 shade = True,label = index,alpha = 0.7)  
 plt.legend()  
 plt.title('{}分布图'.format(index))  
 return plt.show()  
  
*# 过滤极端数据*df.describe()  
df[df.总金额 > 5000]  
plot\_df = df[(df.总金额 < 500)&(df.退款金额 < 400)][['总金额','买家实际支付金额','退款金额']]  
kde\_plot\_array(plot\_df)

df[df.总金额 > 3000]

change = df2[['买家实际支付金额','日']].groupby('日').sum().round(2).reset\_index().sort\_values(by = '日')

def echarts\_line(x,y,title = '主标题',subtitle = '副标题',label = '图例'):  
 *"""  
 x: 函数传入x轴标签数据  
 y：函数传入y轴数据  
 title：主标题  
 subtitle：副标题  
 label：图例  
 """* line = Line(  
 init\_opts=opts.InitOpts(  
 bg\_color='#080b30', *# 设置背景颜色* theme='dark' *# 设置主题  
 # width='980px', # 设置图的宽度  
 # height='800px' # 设置图的高度* )  
 )  
 line.add\_xaxis(x)  
 line.add\_yaxis(  
 label,  
 y,  
 is\_symbol\_show=False, *# 是否显示数据标签点* is\_smooth=True, *# 设置曲线平滑* label\_opts=opts.LabelOpts(  
 is\_show=False, *# 是否显示数据* ),  
 itemstyle\_opts=opts.ItemStyleOpts(color='#00ca95'), *# 设置系列颜色  
 # 线条粗细阴影设置* linestyle\_opts={  
 "normal": {  
 "color": "#4ADEDE", *#线条颜色* "shadowColor": 'rgba(0, 0, 0, .3)', *#阴影颜色和不透明度* "shadowBlur": 2, *#阴影虚化大小* "shadowOffsetY": 5, *#阴影y偏移量* "shadowOffsetX": 5, *#阴影x偏移量* "width": 6 *# 线条粗细* },  
 },  
 *# 阴影设置* areastyle\_opts={  
 "normal": {  
 "color": JsCode("""new echarts.graphic.LinearGradient(0, 0, 0, 1, [{  
 offset: 0,  
 color: '#7BD5F5'  
 },  
 {  
 offset: 1,  
 color: 'rgba(0,202,149, 0)'  
 }  
 ], false)"""), *#设置底色色块渐变* "shadowColor": 'rgba(0,202,149, 0.9)', *#设置底色阴影* "shadowBlur": 20 *#设置底色阴影大小* }  
 },  
 )  
 line.set\_global\_opts(  
 *# 标题设置* title\_opts=opts.TitleOpts(  
 title=title, *# 主标题* subtitle=subtitle, *# 副标题* pos\_left='center', *# 标题展示位置* title\_textstyle\_opts=dict(color='#fff') *# 设置标题字体颜色* ),  
 *# 图例设置* legend\_opts=opts.LegendOpts(  
 is\_show=True, *# 是否显示图例* pos\_left='right', *# 图例显示位置* pos\_top='3%', *#图例距离顶部的距离* orient='horizontal' *# 图例水平布局* ),  
 )  
 return line.render\_notebook()

echarts\_line(change['日'],change['买家实际支付金额'],title = '成交金额变化图',subtitle = "成交金额在时间维度上的变化",  
 label = '成交金额')

week\_change = df2[['周','买家实际支付金额']].groupby('周').sum().round(2).reset\_index()

def echarts\_bar(x,y,title = '主标题',subtitle = '副标题',label = '图例'):  
 *"""  
 x: 函数传入x轴标签数据  
 y：函数传入y轴数据  
 title：主标题  
 subtitle：副标题  
 label：图例  
 """* bar = Bar(  
 init\_opts=opts.InitOpts(  
 bg\_color='#080b30', *# 设置背景颜色* theme='dark' *# 设置主题  
 # width='980px', # 设置图的宽度  
 # height='800px' # 设置图的高度* )  
 )  
 bar.add\_xaxis(x)  
 bar.add\_yaxis(label,y,  
 label\_opts=opts.LabelOpts(is\_show=True) *# 是否显示数据* ,category\_gap="50%" *# 柱子宽度设置* )   
 bar.set\_series\_opts( *# 自定义图表样式* label\_opts=opts.LabelOpts(is\_show=False), *# 是否显示数据标签* markpoint\_opts=opts.MarkPointOpts(  
 data=[  
 opts.MarkPointItem(type\_="min", name="最小值"), *# 显示最小值标签* opts.MarkPointItem(type\_="max", name="最大值"), *# 显示最大值标签* opts.MarkPointItem(type\_="average", name="平均值") *# 显示均值标签* ]  
 ),  
 itemstyle\_opts={   
 "normal": {  
 "color": JsCode(  
 """new echarts.graphic.LinearGradient(0, 0, 0, 1, [{  
 offset: 0,color: 'rgba(0, 244, 255, 1)'}  
 ,{offset: 1,color: 'rgba(0, 77, 167, 1)'}], false)  
 """  
 ), *# 调整柱子颜色渐变* "barBorderRadius": [100, 100, 100, 100], *# 调整柱子圆角弧度* "shadowColor": "rgb(0, 160, 221)", *# 调整阴影颜色* }  
 }  
 )  
 bar.set\_global\_opts(  
 *# 标题设置* title\_opts=opts.TitleOpts(  
 title=title, *# 主标题* subtitle=subtitle, *# 副标题* pos\_left='center', *# 标题展示位置* title\_textstyle\_opts=dict(color='#fff') *# 设置标题字体颜色* ),  
 *# 图例设置* legend\_opts=opts.LegendOpts(  
 is\_show=True, *# 是否显示图例* pos\_left='right', *# 图例显示位置* pos\_top='3%', *#图例距离顶部的距离* orient='horizontal' *# 图例水平布局* ),  
 )  
 return bar.render\_notebook()

echarts\_bar(week\_change['周'].tolist(),week\_change['买家实际支付金额'].tolist(),title = '订单成交金额平均每周对比',  
 subtitle = '每周对比图',label = '成交金额')

hour\_change = df2[['小时','买家实际支付金额']].groupby('小时').sum().round(2).reset\_index()

echarts\_line(hour\_change['小时'],hour\_change['买家实际支付金额'],title = '每天各时段成交金额变化图'  
 ,subtitle = '一天24小时哪个时间段成交金额多',label = '成交金额')  
*#%% md*### b. 成交金额在地区维度上的分布  
*#%%*change\_map = df2[['收货地址','买家实际支付金额']].groupby('收货地址').sum().round(2).reset\_index().sort\_values(by = '买家实际支付金额',  
 ascending = False)  
*#%%*def echarts\_map(province,data,title = '主标题',subtitle = '副标题',label = '图例'):  
 *"""  
 province：传入省份List  
 data：传入各省对应的数据List  
 title：主标题  
 subtitle：副标题  
 label：图例  
 """* map\_ = Map(  
 init\_opts=opts.InitOpts(  
 bg\_color='#080b30', *# 设置背景颜色* theme='dark' *# 设置主题  
 # width='980px', # 设置图的宽度  
 # height='800px' # 设置图的高度* )  
 )  
 map\_.add(label,[list(i) for i in zip(province,data)])  
 map\_.set\_global\_opts(  
 *# 标题设置* title\_opts=opts.TitleOpts(  
 title=title, *# 主标题* subtitle=subtitle, *# 副标题* pos\_left='center', *# 标题展示位置* title\_textstyle\_opts=dict(color='#fff') *# 设置标题字体颜色* ),  
 *# 图例设置* legend\_opts=opts.LegendOpts(  
 is\_show=True, *# 是否显示图例* pos\_left='right', *# 图例显示位置* pos\_top='3%', *#图例距离顶部的距离* orient='horizontal' *# 图例水平布局* ),  
 visualmap\_opts = opts.VisualMapOpts(max\_ = int(data.max()),is\_piecewise = False)  
 )  
 return map\_.render\_notebook()  
  
*#%%*echarts\_map(change\_map['收货地址'],change\_map['买家实际支付金额'],title = '成交金额分布图'  
 ,subtitle = '成交金额在全国各地分布情况',label = '成交金额')  
*#%%*change\_map = df2[['收货地址','买家实际支付金额']].groupby('收货地址').sum().round(2).reset\_index().sort\_values(by = '买家实际支付金额',  
 ascending = False)  
*#%% md*## 2. 退款金额  
*#%% md*### a. 退款金额在时间维度上的变化  
*#%%*back\_money = df2[['日','退款金额']].groupby('日').sum().round(2).reset\_index()  
*#%%*echarts\_bar(back\_money['日'].tolist(),back\_money['退款金额'].tolist(),title = '退款金额变化图'  
 ,subtitle = '退款金额日变化图',label = '退款金额')  
*#%%*hour\_back\_money = df2[['小时','退款金额']].groupby('小时').sum().round(2).reset\_index()  
*#%%*echarts\_line(hour\_back\_money['小时'].tolist(),hour\_back\_money['退款金额'].tolist(),title = '退款金额变化图'  
 ,subtitle = '退款金额每日小时变化图',label = '退款金额')  
*#%% md*### b. 退款金额在地区维度上的分布  
*#%%*local\_back\_money = df2[['收货地址','退款金额']].groupby('收货地址').sum().round(2).reset\_index()  
*#%%*echarts\_map(local\_back\_money['收货地址'],local\_back\_money['退款金额'],title = '退款金额分布图'  
 ,subtitle = '退款金额在全国各地分布情况',label = '退款金额')  
*#%%*map3d\_with\_bar3d(local\_back\_money['收货地址'],local\_back\_money['退款金额'],title = '退款金额分布图'  
 ,subtitle = '退款金额在全国各地分布情况',label = '退款金额')