

**《数据采集与数据集成》期末报告**

题 目：

对“大众点评”温州市吃喝住行的数据集成

学 院： 计算机与人工智能学院

班 级： 21大数据2

学 号： 21211870211

姓 名： 范逸烽

指导老师： 李洪

完成日期： 2023/12/22

**目录**

[《数据采集与数据集成》期末报告 1](#_Toc22340)

[对“大众点评”网站的数据采集与集成 1](#_Toc17087)

[目录 1](#_Toc8791)

[一、 主题及功能介绍 1](#_Toc13035)

[二、 采用的关键技术阐述 1](#_Toc29385)

[三、 实现过程 1](#_Toc8012)

[四、 程序实现的结果 6](#_Toc21148)

[五、结论与展望 9](#_Toc22164)

[六、参考资料 9](#_Toc22431)

[七、附件：原始代码 9](#_Toc14743)

# 主题及功能介绍

研究主题在于对“大众点评”网站的温州市吃喝住行数据进行数据采集和数据集成。

代码功能即为对网站数据进行爬取，再通过Kettle对数据进行清洗和集成。

# 采用的关键技术阐述

**Web Scraping:** 通过requests库发送HTTP请求，获取网页的HTML源码。

**HTML Parsing:** 使用lxml库的etree模块解析HTML源码，提取所需的信息。

**XPath:** 使用XPath表达式定位HTML元素。

**Regular Expressions:** 使用re库进行正则表达式匹配，用于检测当前页面的饭店记录数量。

**Time Management:** 使用time库的sleep函数来控制请求的频率，防止过于频繁的请求导致被服务器封禁。

**Data Storage:** 使用csv库将提取的数据存储到CSV文件中。

**Looping:** 使用Python的for循环来遍历网页和提取每个饭店的信息。

**Error Handling:** 对可能出现的异常情况（如某些信息不存在）进行了处理，避免程序因为异常而中断。

**Data Structures:** 使用Python的列表和字典来存储和组织数据。每个饭店的信息被存储在一个字典中，所有饭店的字典被存储在一个列表中。

**Kettle:**在数据集成和ETL（提取、转换、加载）过程中，我们使用了Kettle，这是一个开源的、功能强大的数据集成工具。以下是我们使用Kettle进行的一些关键操作：

转换操作：

过滤记录：我们使用条件过滤步骤来选择满足特定条件的记录。

设置值为NULL：在某些情况下，我们需要将某些字段的值设置为NULL，以表示数据缺失或未知。

替换NULL值：我们使用“如果字段值为null”步骤来替换NULL值，以便进行进一步的数据分析和处理。

合并记录：我们使用“合并记录”步骤来将来自不同源的数据合并到一起。

排序记录：我们使用“排序记录”步骤来按照特定的顺序对数据进行排序。

去除重复记录：我们使用“去除重复记录”步骤来清理数据，确保每条记录的唯一性。

剪切字符串：我们使用“剪切字符串”步骤来处理文本数据，例如去除空格或提取子字符串。

选择/改名值：我们使用“选择/改名值”步骤来选择需要的字段，并对字段进行重命名。

作业操作：

检查表是否存在：我们使用“表存在”作业项来检查数据库中是否存在特定的表。

SQL操作：我们使用“SQL脚本”作业项来执行SQL查询，以便从数据库中提取数据或对数据进行更新。

调用转换：我们使用“调用转换”作业项来执行预定义的转换，这使得我们可以在一个大的作业中重用和组合多个转换。

**数据预处理：**我们使用了Python的re模块和pandas库来进行数据预处理。具体来说，我们使用了正则表达式来提取和替换字符串中的数字，然后将这些数字转化为整数或浮点数。

**数据转换：**我们使用了pandas库的astype函数来将数据从一种类型转换为另一种类型。这是数据预处理的一个重要步骤，可以确保我们的数据符合后续分析的需求。

**数据填充：**我们使用了pandas库的replace函数和median函数来填充数据。具体来说，我们计算了某一列中非零值的中位数，然后用这个中位数来填充该列中的零值。这是一种常见的数据填充方法，可以帮助我们处理缺失值或异常值。

**列名修改：**我们使用了pandas的rename函数来修改列名，例如将“饭店标签”改为“标签类别”。

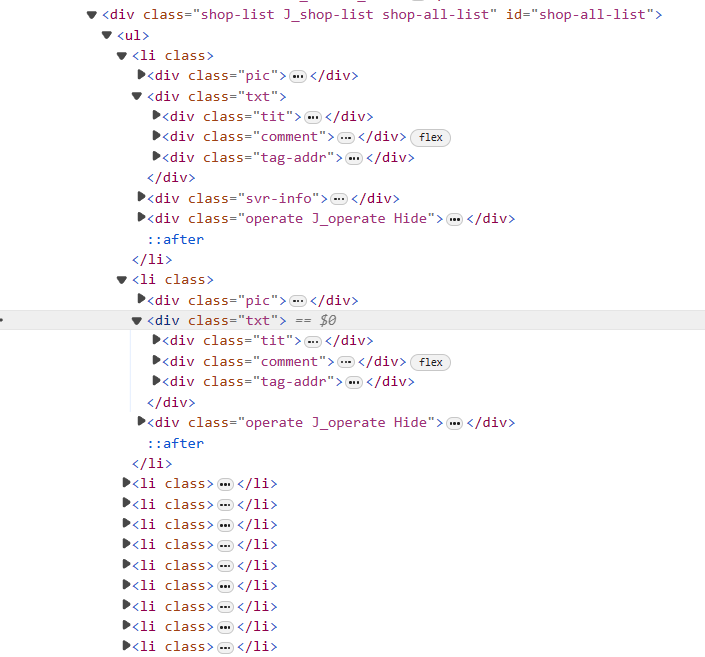
**离散化：**我们使用了pandas的cut函数和数据的分位数来将连续的数值属性（如星级评分、评论数、平均每人消费）离散化为“低”、“中”、“高”三个等级。

**数据聚合：**数据聚合是数据处理的一个重要步骤，它通过创建数据的汇总来提供有价值的信息。在我们的例子中，我们使用了pandas的groupby和agg函数来按照“标签类别”进行聚合，计算每种类型的店铺的平均星级评分、平均评论数、平均人均消费数额。

# 实现过程

**数据采集：**

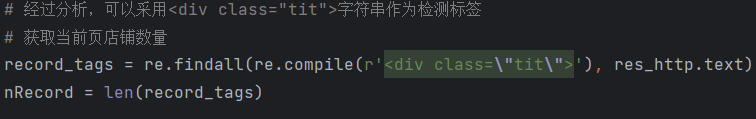
对基础网页进行分析：



发现所有的数据记录在一条li中，在li标签下的class=”tit”对应第一个出现的店名。



故可以采用该标签<div class="tit">，实现对页面数据量的获取

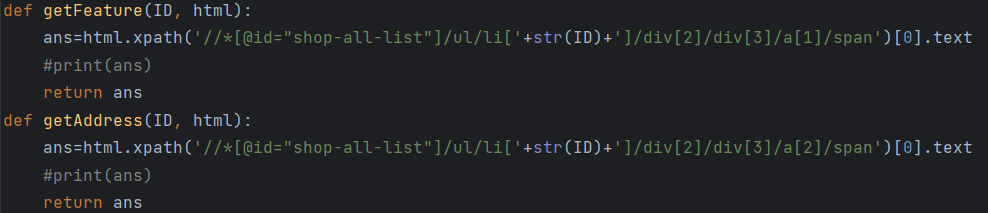


在对该目标实现的起步阶段，发现不携带登录信息的访问“大众点评”网页会默认返回登录页面，在响应头中添加headers信息，User-Agent模拟正常浏览器访问，cookie为当需要登录授权时使用，解决问题。

page\_headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0", "cookie": "navCtgScroll=0; \_lxsdk\_cuid=18c8ae19131c8-0d052398b4b53e-4c657b58-144000-18c8ae19131c8; \_lxsdk=18c8ae19131c8-0d052398b4b53e-4c657b58-144000-18c8ae19131c8; \_hc.v=a58d5b87-d9f3-d30e-e7c5-83add935d092.1703137090; WEBDFPID=0z5642946vvx5028yu437xvy6x82v73y81x1zv80592979587z251027-2018497089895-1703137087961OQKWOYOfd79fef3d01d5e9aadc18ccd4d0c95072494; fspop=test; cy=101; cye=wenzhou; \_lx\_utm=utm\_source%3Dbing%26utm\_medium%3Dorganic; Hm\_lvt\_602b80cf8079ae6591966cc70a3940e7=1703137100; dper=e5b36fcb8124ba46a46cd5ff59ba37aa8301e86af0bdf735f268324d49a3528e20448086f641a8f8591e549502b4c526e0dad78c78adc68a0d3a428cff3018f9; qruuid=14f0a4a4-c04b-4546-943d-043057f21611; ll=7fd06e815b796be3df069dec7836c3df; s\_ViewType=10; Hm\_lpvt\_602b80cf8079ae6591966cc70a3940e7=1703166930; \_lxsdk\_s=18c8c9fefe0-45f-40f-fa%7C%7C216"}

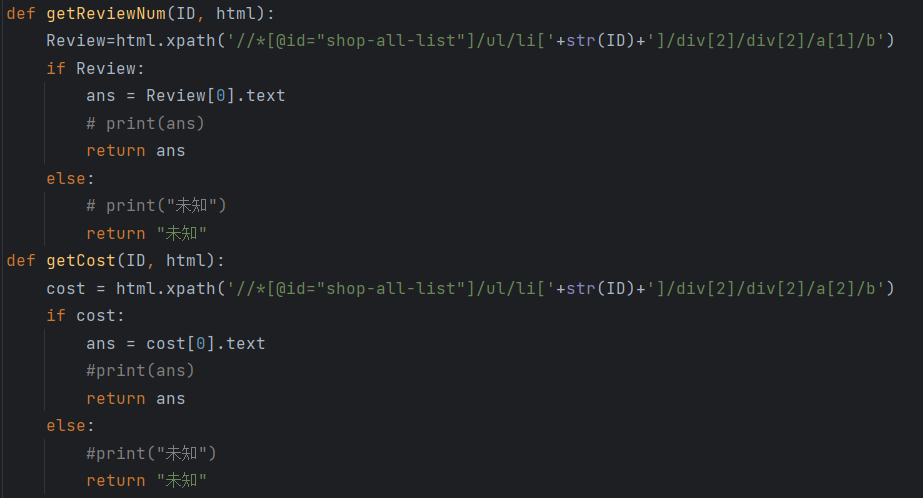
对网页进行分析确定所需数据在网页中的位置，创建方法



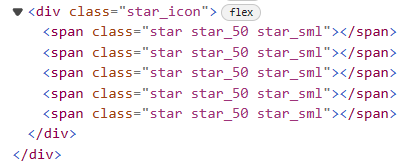


在数据采集的过程中，多次发现数据爬取程序崩溃，原因包括代码中对网页爬取的数据的不稳定性，包括部分网页数据记录不存在对应该标签的异常情况和存在标签但标签内容为空的情况，通过完善对代码中排除异常情况的操作后得到改进。





此外还发现星级评分该字段存在于span标签的class属性中，对其使用正则表达式，提取出星级评分。





存在部分网页数据记录该星级评分不存在的情况，返回none，在导出的数据集中表现为0的string字段。

为了避免存在某一页商户记录数不为15，需要检测当前页面饭店记录数量。经过分析，可以采用<div class="tit">字符串作为检测标签，获取当前页饭店数量，上面已经涉略。



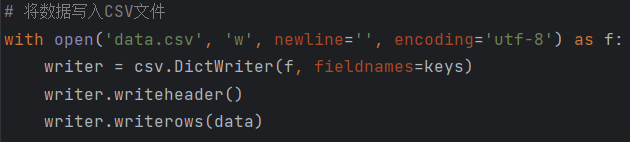
通过遍历网页，完成数据的采集。





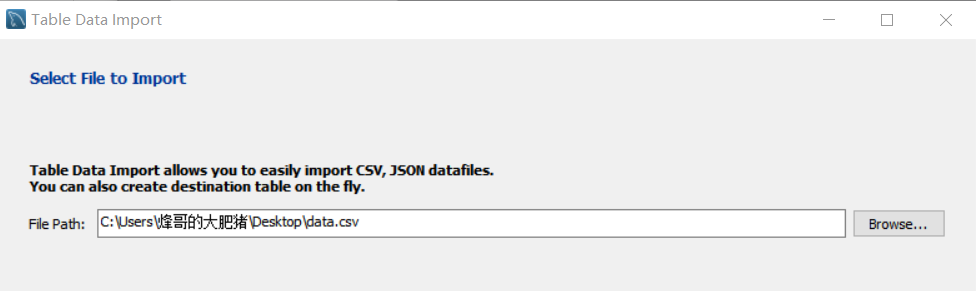


最后将数据写入data.csv文件：

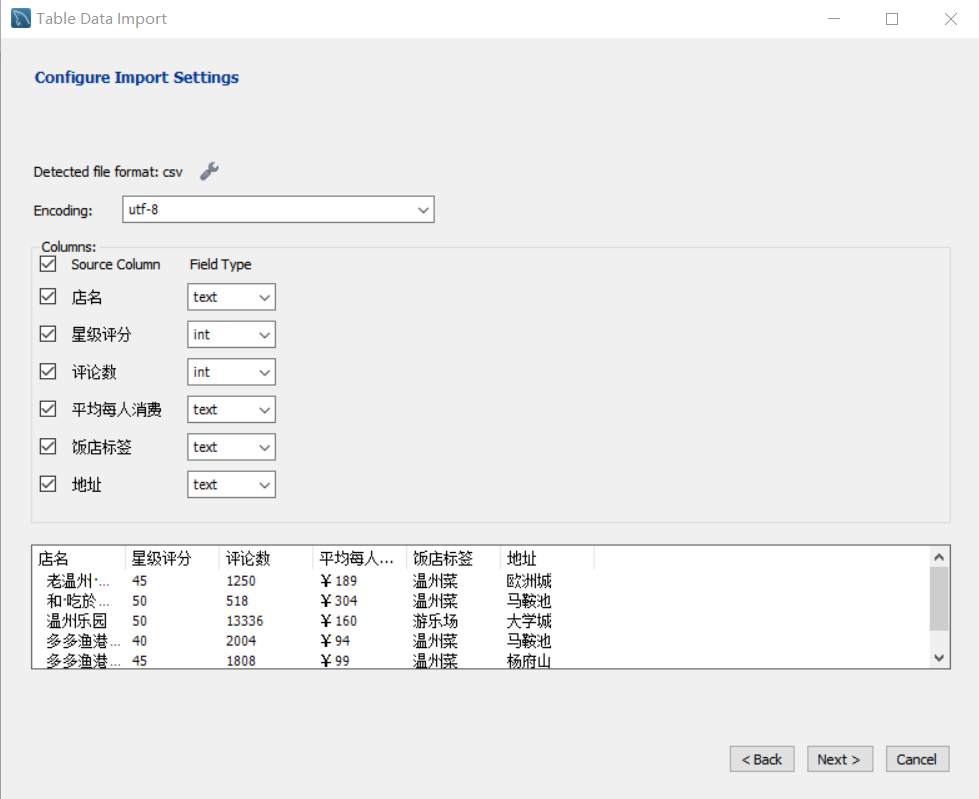


**数据集成：**

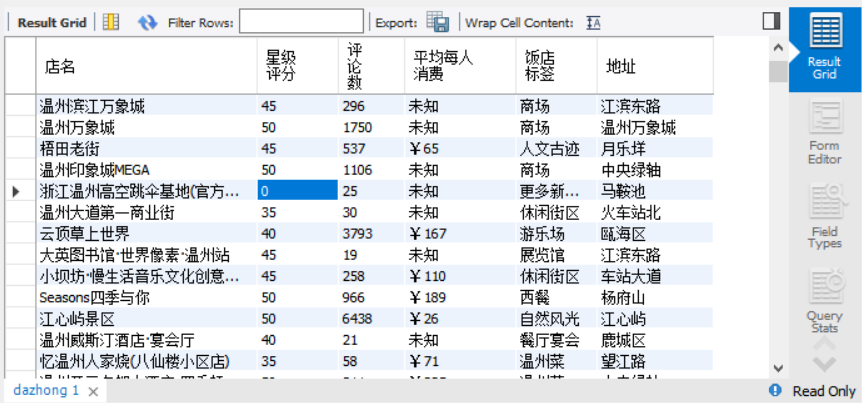
打开MySQL Workbench使用Table Data Import功能一键导入刚刚生成的data.csv.



确定字段类型和编码类型。



完成数据导入。



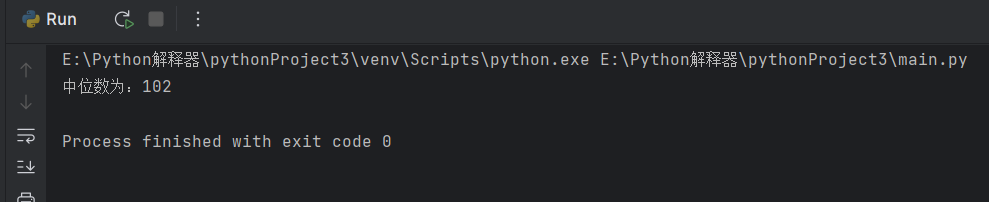
观察数据发现存在以下问题并提出解决思路：

处理缺失值：数据中，有些饭店的"平均每人消费"和"地址"是"未知"。对于处理缺失值的问题，我们可以选择填充这些缺失值，或者删除含有缺失值的行。填充缺失值的方法有很多种，比如使用平均值、中位数、众数，或者使用预测模型预测缺失值。这里的“平均每人消费”字段存在大量的“未知”数据不适合删除含有缺失值的数据，采用填充中位数的方法。但在这里对地址的填充缺失值极为困难，且缺失值较少，采用直接删除含有缺失值的数据的方法处理。

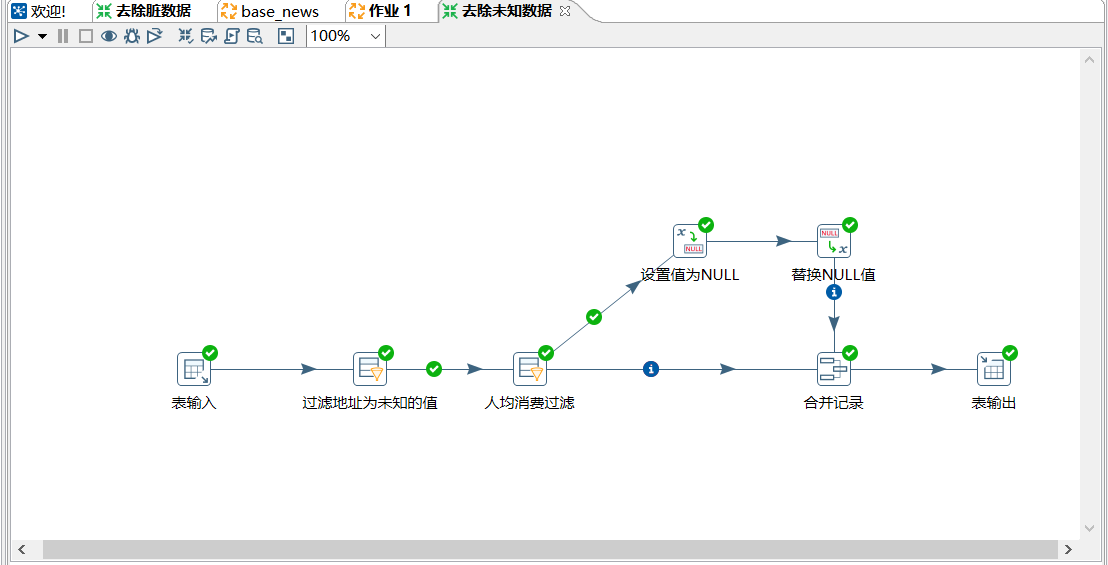
第一步：求中位数，将所爬取数据中为非“未知”的“平均每人消费”字段导出，在python中写出求取中位数的代码，

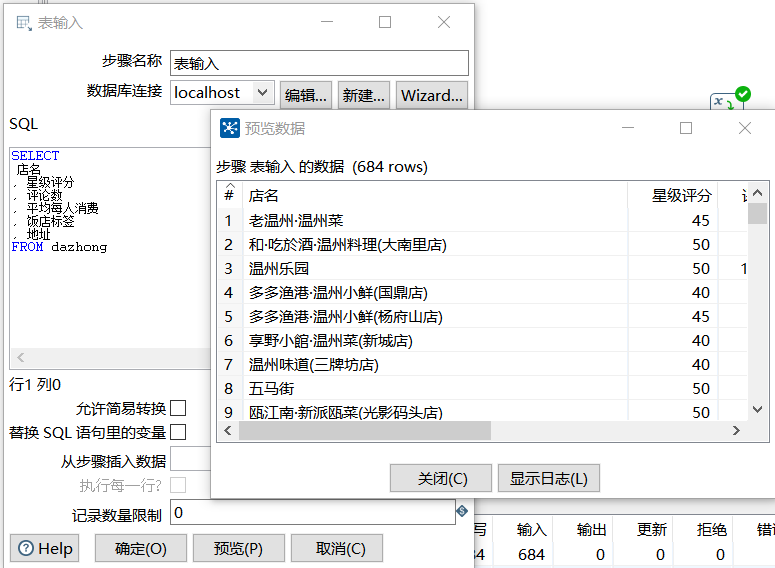


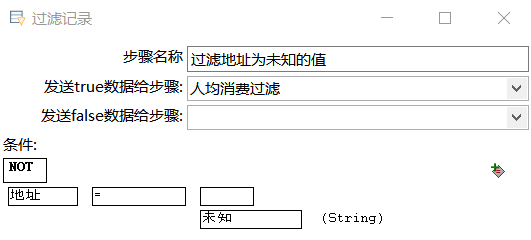
得中位数为：102

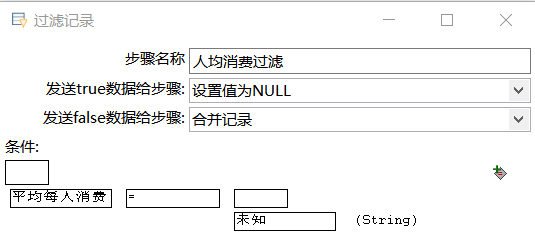


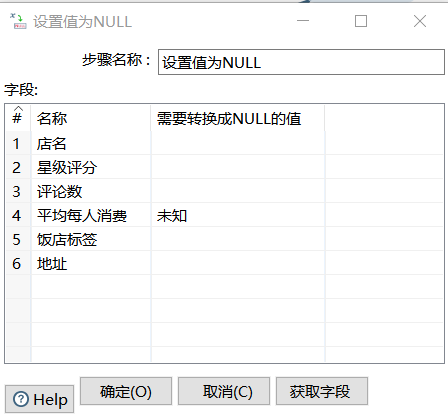
启用Kettel进行数据集成，完成如下“去除未知数据”的转换。





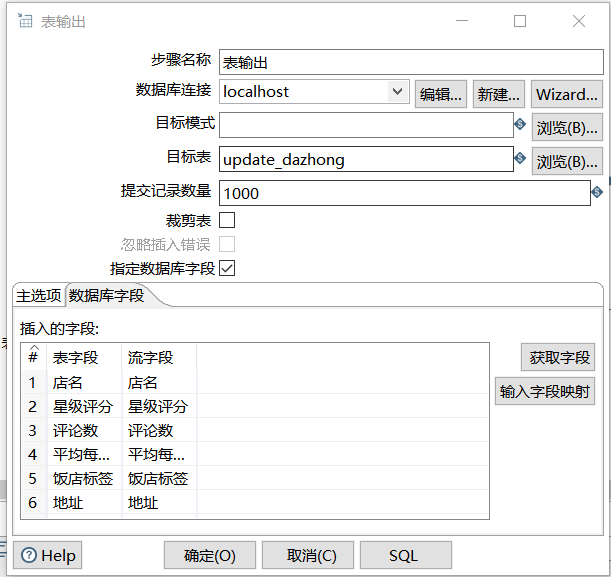




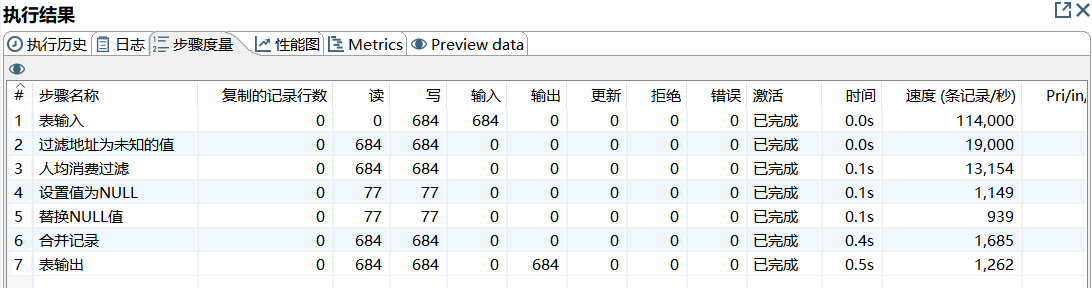


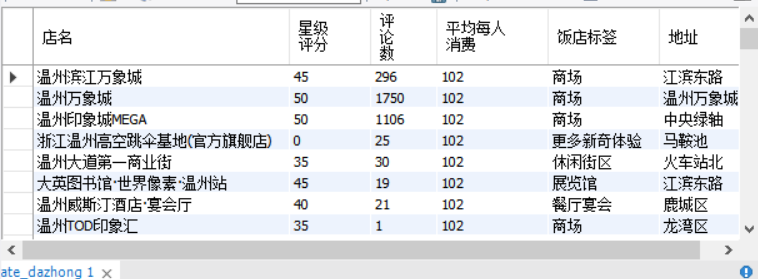






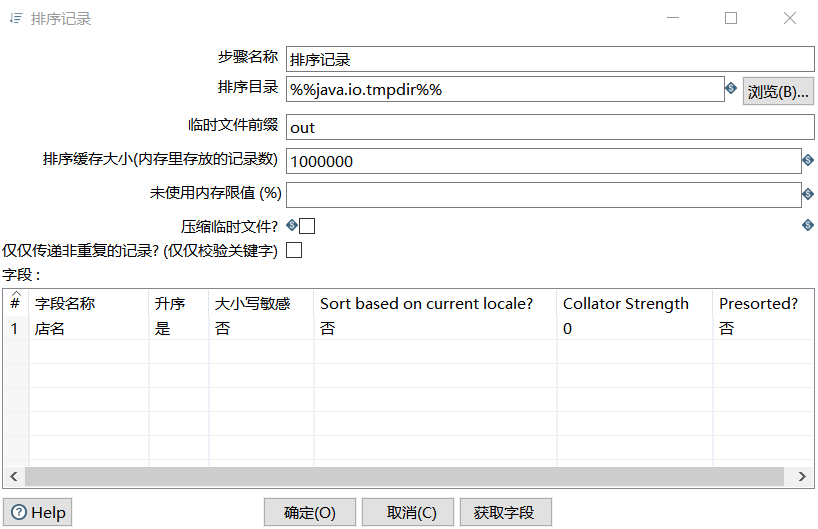
最后确认该“去除未知数据”的转换无误

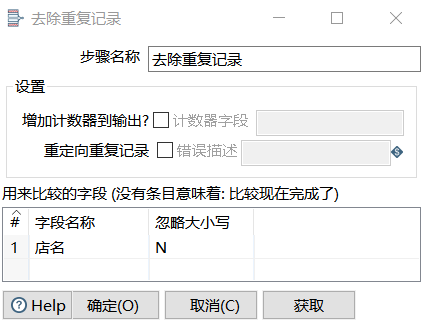




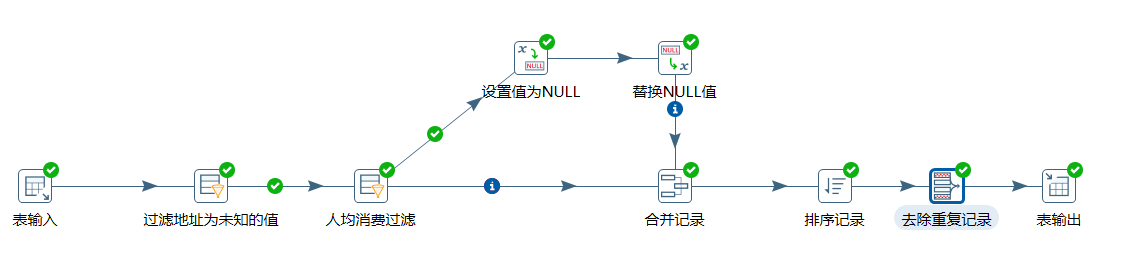
处理数据冗余：数据中存在重复的行，需要删除这些重复的行。这里还是使用kettle实现这个功能。

在原先的“去除未知数据”的转换基础上增加两个操作“排序记录”和“去除重复记录”。

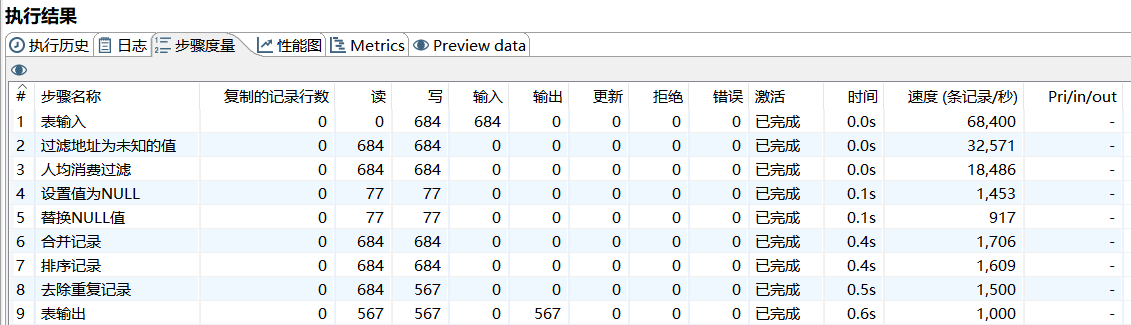




由此得到完善版“去除未知数据”的转换。

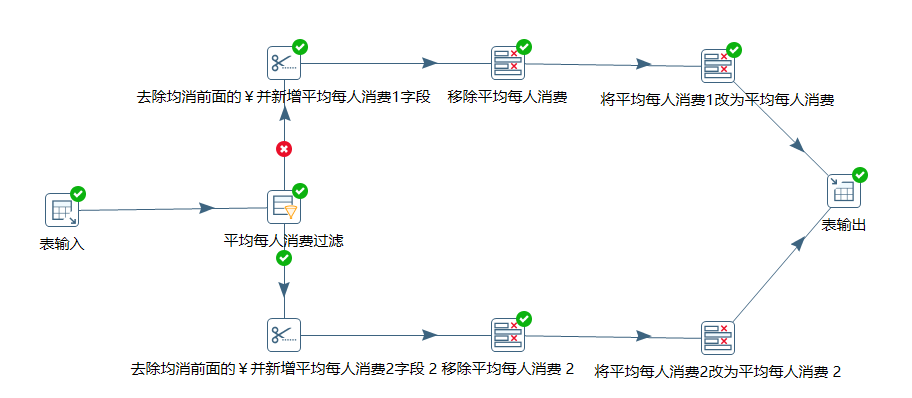


检验后成功：

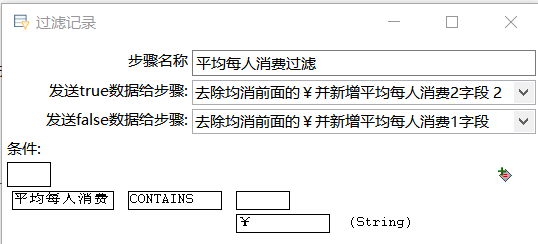


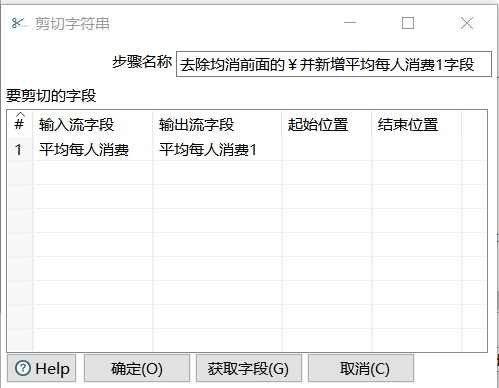
数据变换：需要将一些列的数据类型进行转换，以便进行后续的分析。比如，需要将"星级评分"的数据类型从int型转换为double型，并除以10，转化为评分为0~5的标准评分格式。需要将"平均每人消费"的数据类型从字符串转换为数值，并去掉前面的"￥"符号。

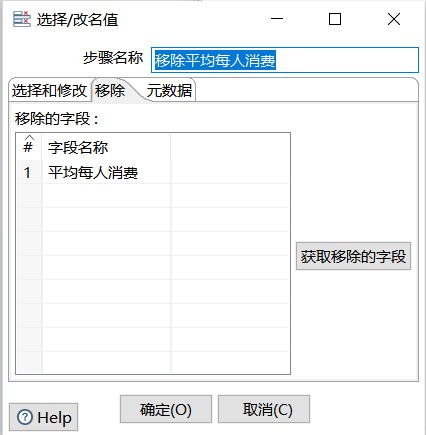
增加一个新的名为“平均每人消费数据变换”的转换，如下图所示

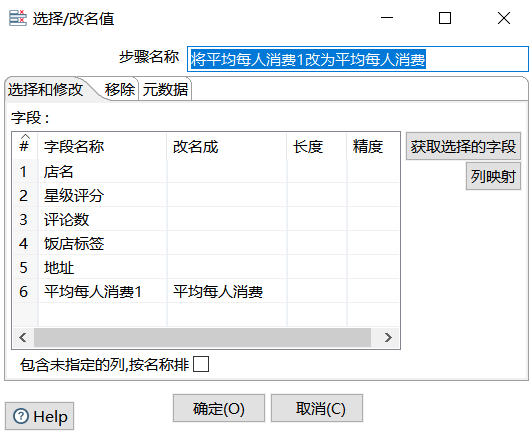


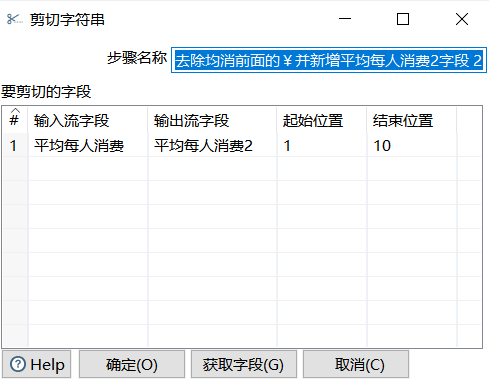


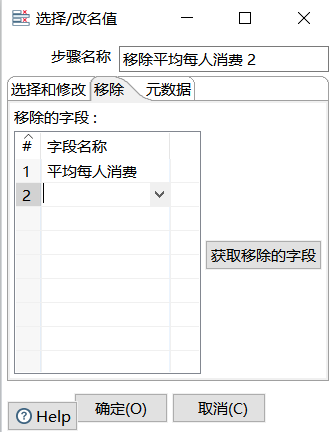


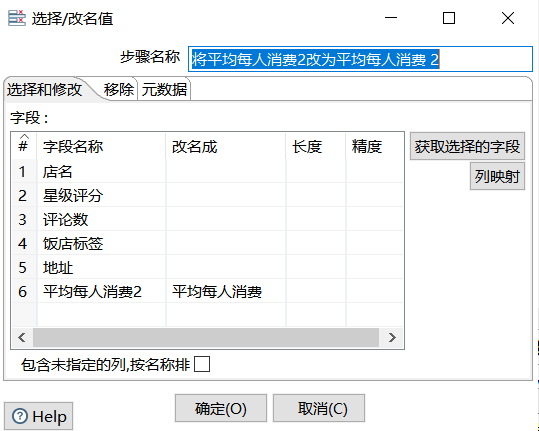


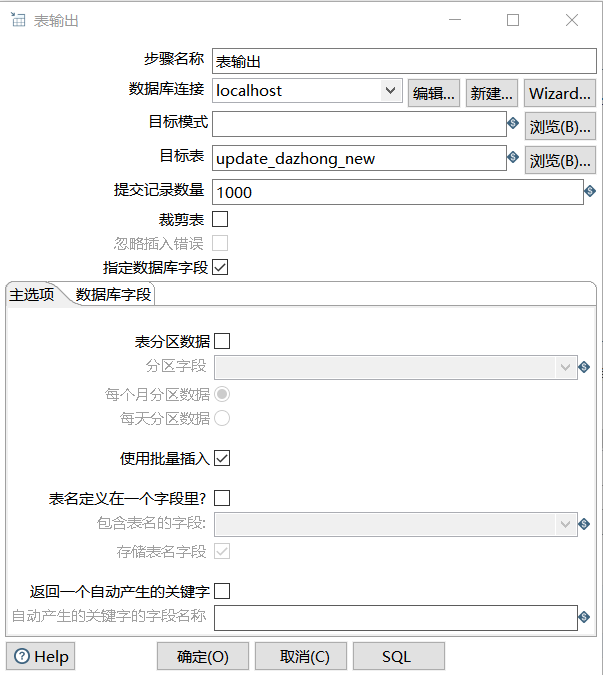




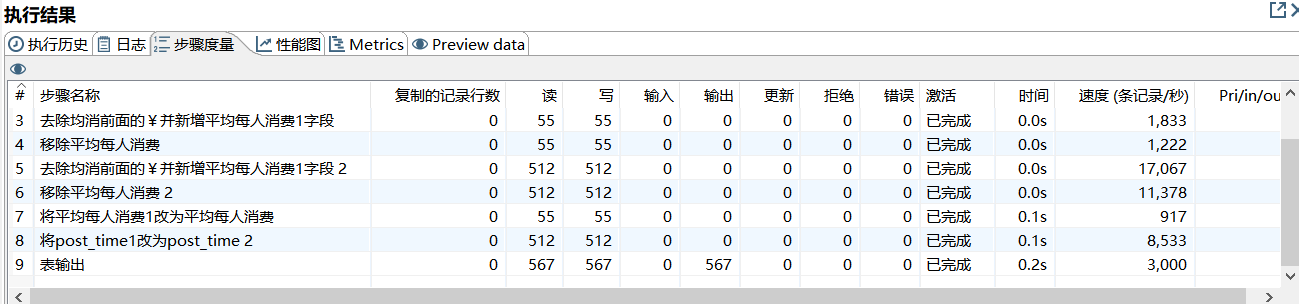




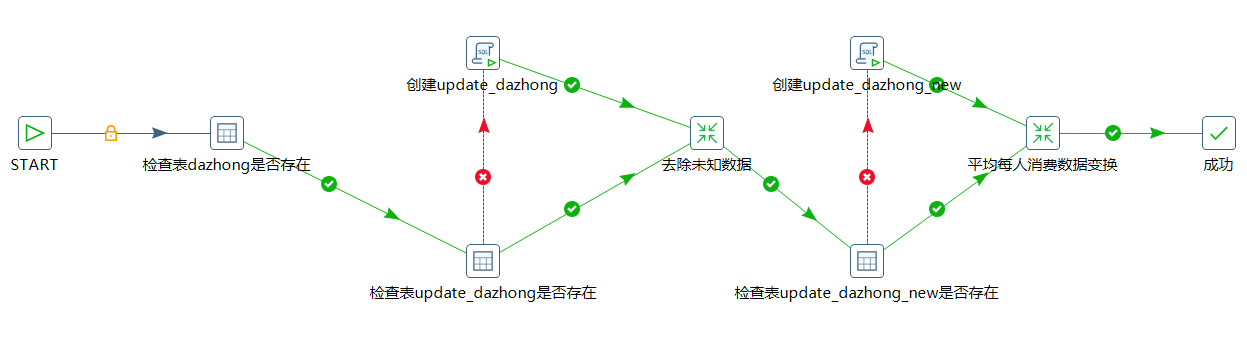




检验该转换“平均每人消费数据变换”是否成功

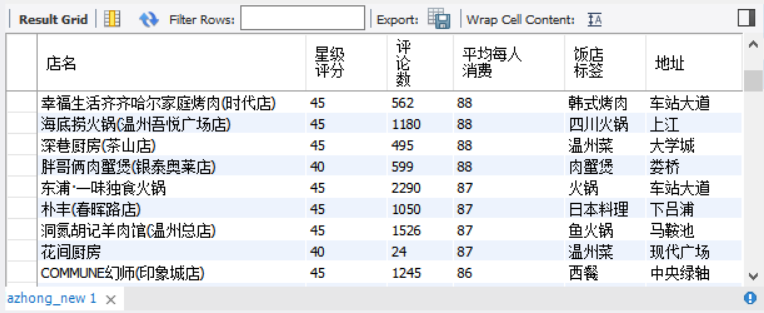


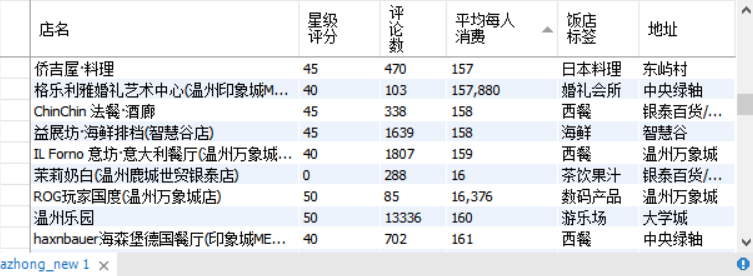
将两个名为“去除未知数据”和“平均每人消费数据变换”的转换组件载入到名为“初步数据集成”的作业中，如下图所示：



运行该作业，即可完成初步数据集成的操作。

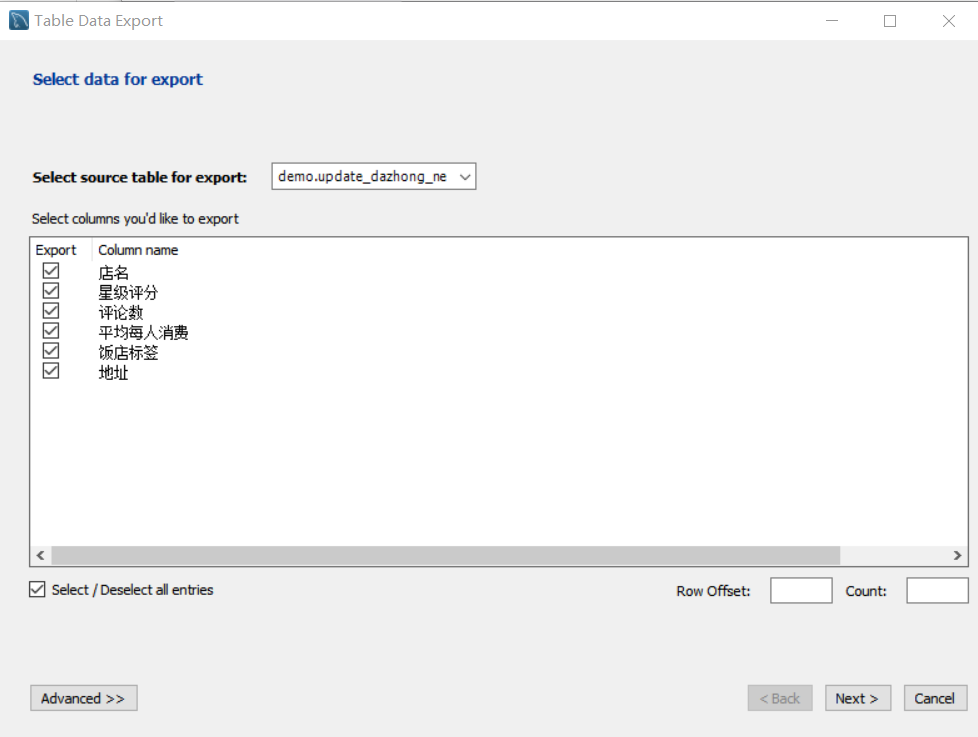
结果数据如下所示：



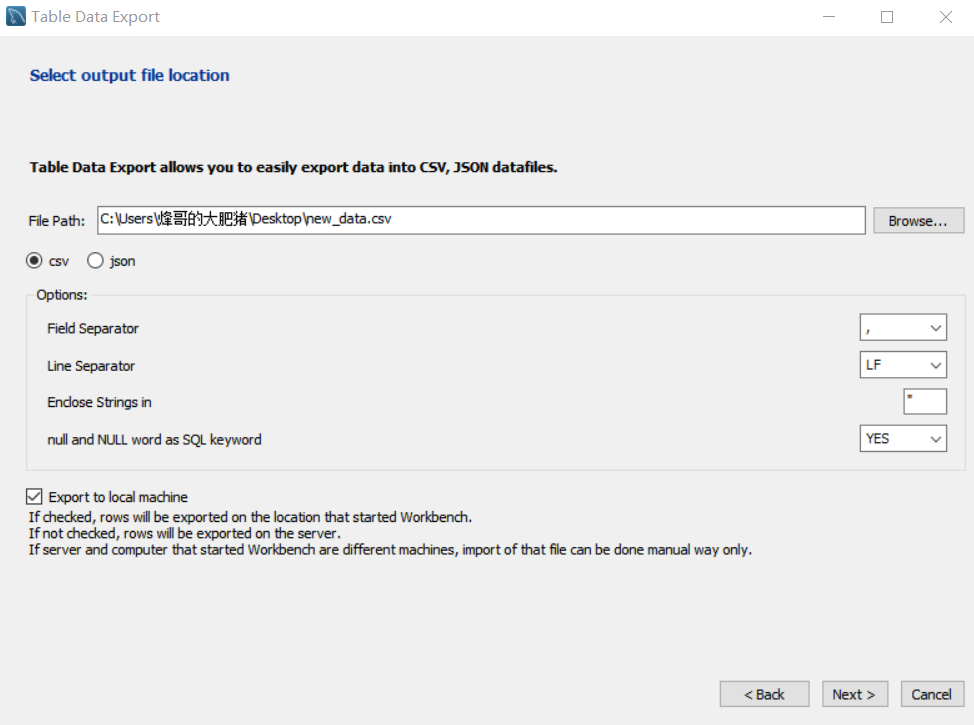
在聚类的过程中发现平均每人消费仍为text格式，且存在个别“，”现象。 

继续进行该特征的数据变换。

通过对原先存于MySQL数据库demo的表updata\_dazhong\_new进行导出。



设置分隔符和字符串包围符等如下：

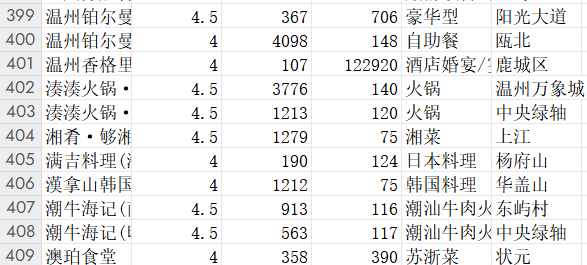


获得导出数据。

新建python项目对数据进行数据清洗和集成。其中使“平均每人消费”字段为int类型，使“星级评分”字段为double类型并且除以10保留一位小数，转化为标准的0~5评分数据。

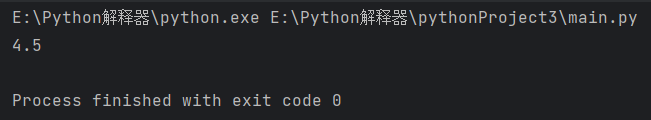


得到的改进数据集processed\_data.csv，完成目标清洗，没有错误。



再次观察发现，此前评分为0的异常数据也尚未处理，可以使用kettle，这里用python可以更加快速的处理。





将所有评分为0的异常数据的值设置为中位数4.5。

完成！

**数据归约：**

对于数值属性（如星级评分、评论数、平均每人消费），可以考虑进行离散化，比如将星级评分分为“高”、“中”、“低”三个等级，使其更加直观。



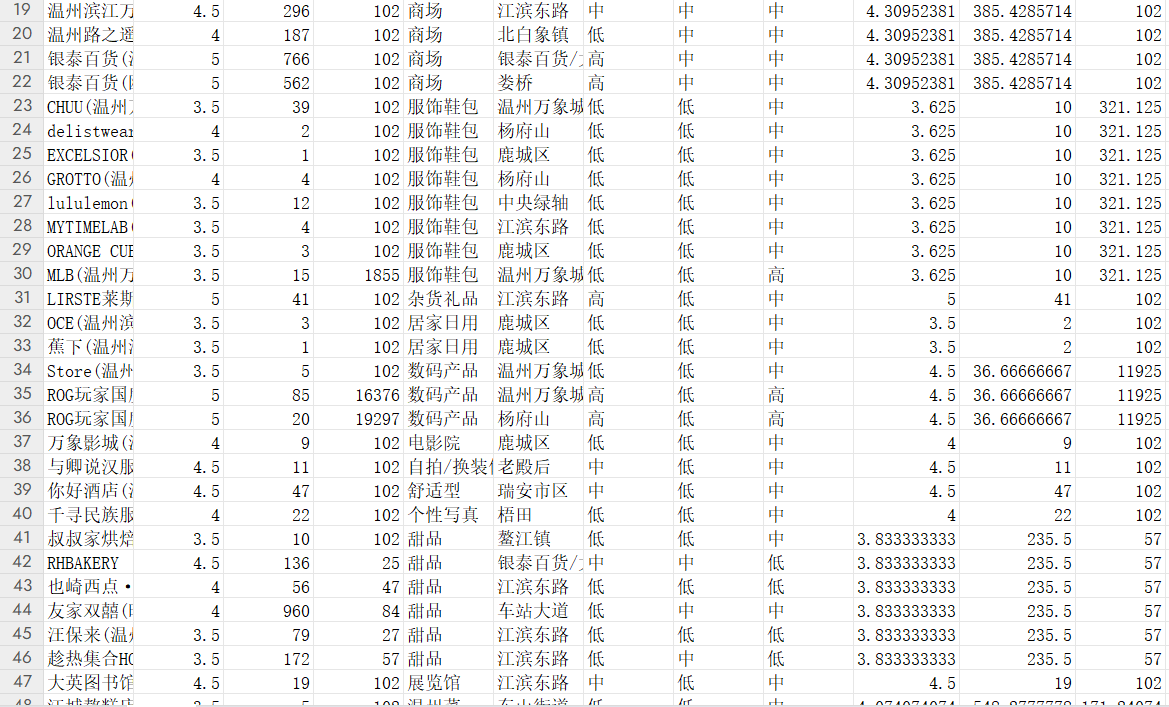
得到结果：



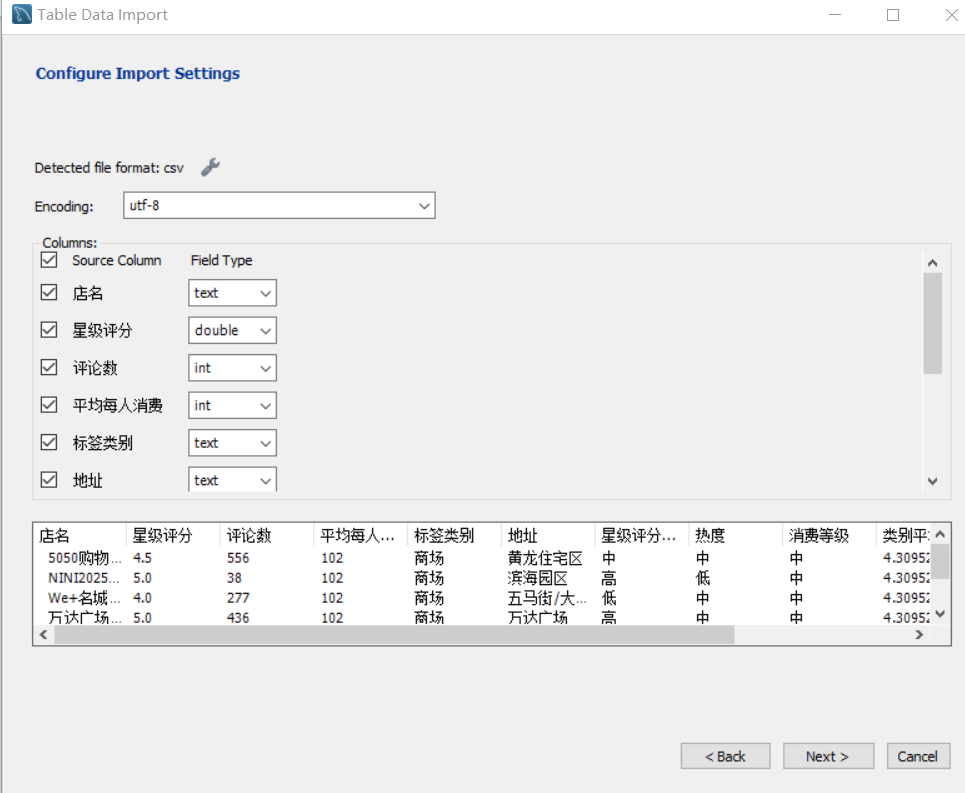
数据立方体聚合：可以根据需要将数据进行聚合，例如，按照饭店标签进行聚合，计算每种类型的店铺的平均星级评分、平均评论数、平均人均消费数额，通过比较，得出该店铺的服务质量的优劣。

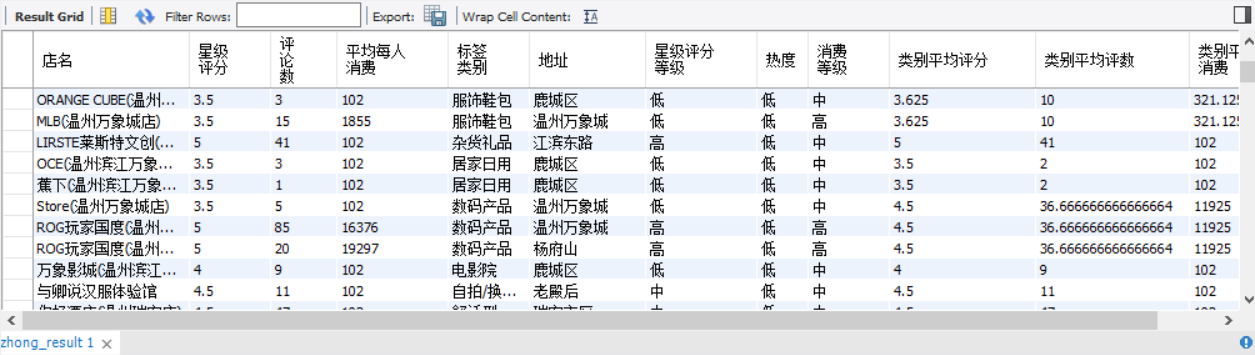


效果如下：



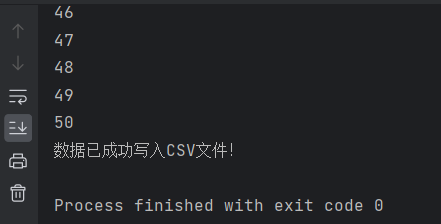
最后导入数据库完成数据采集和数据集成。

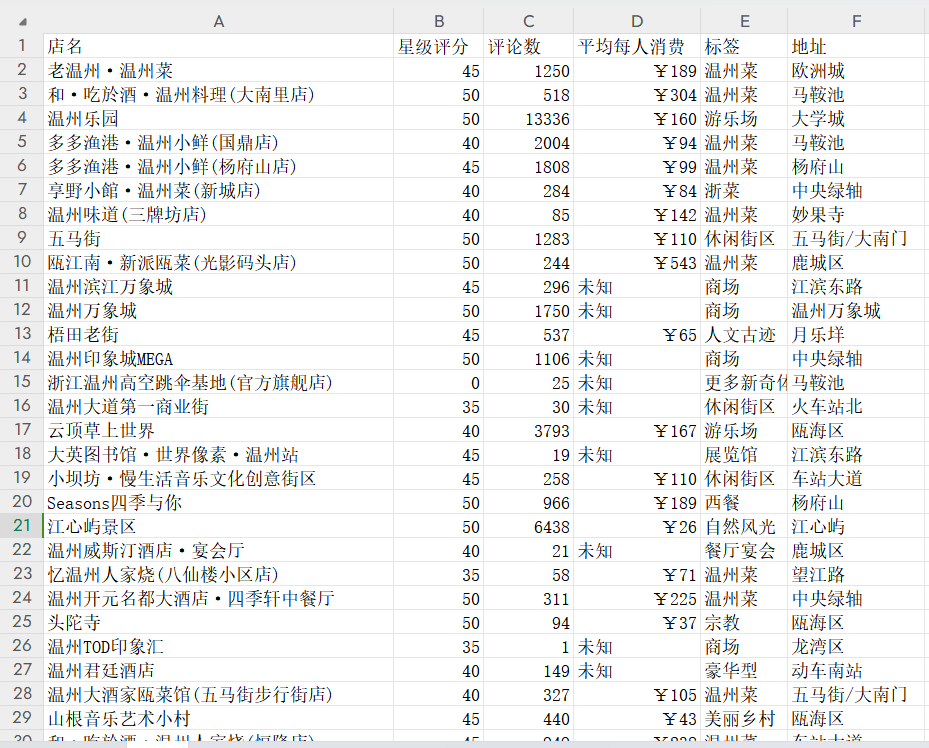




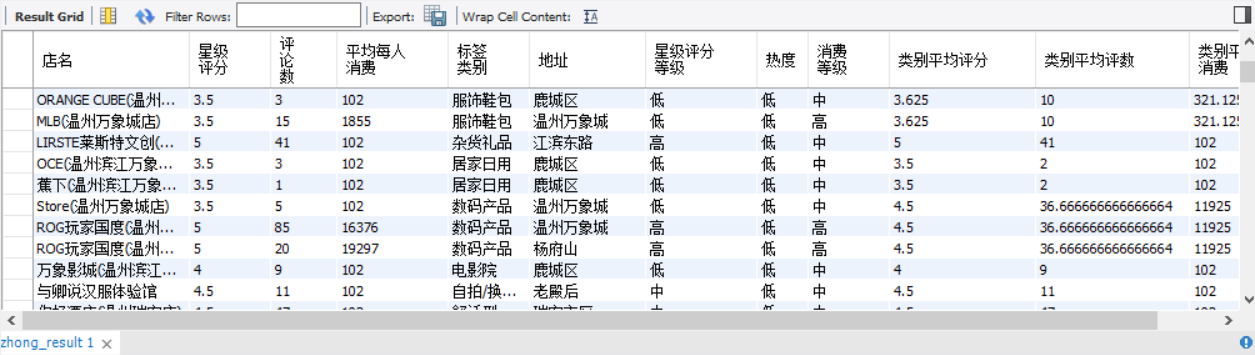
# 程序实现的结果

数据采集到的初始数据：



通过对“大众点评”数据进行爬取，得到689行数据  


数据集成后的数据：



# 结论与展望

**结论：**

数据采集和数据集成是数据分析的关键步骤。通过有效的数据采集，我们可以从各种来源获取大量的数据。然后，通过数据预处理，我们可以解决数据中的各种问题，如缺失值、数据冗余等，使数据更适合进行分析。在这个过程中，我们需要使用各种工具和技术，如XPath和正则表达式，还有kettle等工具。在这次实践中，我们成功地对一份包含多个商铺信息的数据进行了处理和分析。我们首先使用kettle以及Python的pandas库和re模块进行了数据预处理，包括读取和写入CSV文件，提取和替换字符串中的数字，将字符串转化为整数或浮点数，以及将数据从一种类型转换为另一种类型，计算了某一列中非零值的中位数，然后用这个中位数来填充该列中的零值。包括修改列名和离散化数值属性。我们还进行了数据聚合，按照特定的类别计算了平均星级评分、平均评论数和平均消费额。这些操作使我们能够从原始数据中提取有价值的信息，并以更直观的方式呈现这些信息。

**展望：**

尽管我们已经完成了一些基本的数据处理和分析任务，但仍有很多可以进一步探索的方向。例如，我们可以使用更复杂的统计方法或机器学习模型来分析数据，以发现更深层次的规律和趋势。我们还可以将我们的分析结果可视化，以便更直观地理解数据。此外，我们可以进一步优化我们的数据处理流程，例如通过使用更高效的算法或工具，或者通过自动化一些重复的步骤。随着大数据和人工智能的发展，数据采集和数据集成的重要性将越来越高。未来，我们需要开发更有效的数据采集和数据预处理方法，以处理更大规模和更复杂的数据。此外，我们还需要研究如何更好地保护数据的安全和隐私，以应对日益严重的数据泄露和滥用问题。总的来说，数据采集和数据集成将在未来的数据分析和决策制定中发挥越来越重要的作用。希望我们能够充分利用这些技术，为社会的发展做出贡献。

# 六、参考资料

部分参考Microsoft Bing AI求助

七、附件：数据采集原始代码

import requests

from lxml import etree

import re # 调用正则表达式模块用于解析

import time

import csv

# 基础URL

base\_url = 'https://www.dianping.com/search/keyword/101/0\_%E6%B8%A9%E5%B7%9E'

# 添加headers信息，User-Agent模拟正常浏览器访问，cookie为当需要登录授权时使用。

page\_headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0", "cookie": "navCtgScroll=0; \_lxsdk\_cuid=18c8ae19131c8-0d052398b4b53e-4c657b58-144000-18c8ae19131c8; \_lxsdk=18c8ae19131c8-0d052398b4b53e-4c657b58-144000-18c8ae19131c8; \_hc.v=a58d5b87-d9f3-d30e-e7c5-83add935d092.1703137090; WEBDFPID=0z5642946vvx5028yu437xvy6x82v73y81x1zv80592979587z251027-2018497089895-1703137087961OQKWOYOfd79fef3d01d5e9aadc18ccd4d0c95072494; fspop=test; cy=101; cye=wenzhou; \_lx\_utm=utm\_source%3Dbing%26utm\_medium%3Dorganic; Hm\_lvt\_602b80cf8079ae6591966cc70a3940e7=1703137100; dper=e5b36fcb8124ba46a46cd5ff59ba37aa8301e86af0bdf735f268324d49a3528e20448086f641a8f8591e549502b4c526e0dad78c78adc68a0d3a428cff3018f9; qruuid=14f0a4a4-c04b-4546-943d-043057f21611; ll=7fd06e815b796be3df069dec7836c3df; s\_ViewType=10; Hm\_lpvt\_602b80cf8079ae6591966cc70a3940e7=1703166930; \_lxsdk\_s=18c8c9fefe0-45f-40f-fa%7C%7C216"}

data = [] # 以字典列表形式存储数据，列表内每一条记录为一个字典数据结构，代表一家饭店

def getTitle(ID, html):

ans=html.xpath('//\*[@id="shop-all-list"]/ul/li['+str(ID)+']/div[2]/div[1]/a/h4')[0].text

#print(ans)

return ans

def getStar(ID, html):

star\_text = html.xpath('//\*[@id="shop-all-list"]/ul/li['+str(ID)+']/div[2]/div[2]/div/div/span/@class')

if star\_text:

star\_class = star\_text[0]

# 根据 star\_class 提取数字 45

start\_index = star\_class.find('star\_') + len('star\_')

end\_index = start\_index + 2

star\_number = star\_class[start\_index:end\_index]

#print(star\_number)

return star\_number

else:

return None

def getReviewNum(ID, html):

Review=html.xpath('//\*[@id="shop-all-list"]/ul/li['+str(ID)+']/div[2]/div[2]/a[1]/b')

if Review:

ans = Review[0].text

# print(ans)

return ans

else:

# print("未知")

return "未知"

def getCost(ID, html):

cost = html.xpath('//\*[@id="shop-all-list"]/ul/li['+str(ID)+']/div[2]/div[2]/a[2]/b')

if cost:

ans = cost[0].text

#print(ans)

return ans

else:

#print("未知")

return "未知"

def getFeature(ID, html):

ans=html.xpath('//\*[@id="shop-all-list"]/ul/li['+str(ID)+']/div[2]/div[3]/a[1]/span')[0].text

#print(ans)

return ans

def getAddress(ID, html):

ans=html.xpath('//\*[@id="shop-all-list"]/ul/li['+str(ID)+']/div[2]/div[3]/a[2]/span')[0].text

#print(ans)

return ans

keys = ['店名', '星级评分', '评论数', '平均每人消费', '饭店标签', '地址']

# 遍历p1至p50

for i in range(1, 51):

# 添加/p1等等到基础URL

page\_url = base\_url + '/p' + str(i)

# 建立http链接

res\_http = requests.get(page\_url, headers=page\_headers)

html = etree.HTML(res\_http.text)

# res\_http.text即为网页的html源码，数据类型为str。但无法解析，需要通过lxml模块的etree.HTML函数解析为xml格式，建立树状结构数据格式，并返回值赋给html对象

# 在得到经过解析的html信息后，开始循环解析获得单个饭店的信息

# 为了避免存在某一页商户记录数不为15，需要检测当前页面饭店记录数量

# 经过分析，可以采用<div class="tit">字符串作为检测标签

# 获取当前页饭店数量

record\_tags = re.findall(re.compile(r'<div class=\"tit\">'), res\_http.text)

nRecord = len(record\_tags)

for n in range(nRecord):

ID = n + 1 # 饭店记录序号

values = [] # 键值列表,每个循环均重置

values.append(getTitle(ID, html)) # 提取店名

values.append(getStar(ID, html)) # 提取星级评分

values.append(getReviewNum(ID, html)) # 提取评论数

values.append(getCost(ID, html)) # 提取平均每人消费

values.append(getFeature(ID, html)) # 提取饭店标签

values.append(getAddress(ID, html)) # 提取地址

# 建立字典并存入数据列表

data.append(dict(zip(keys, values)))

# 每访问一次页面后休眠一秒

time.sleep(1)

print(i)

# 将数据写入CSV文件

with open('data.csv', 'w', newline='', encoding='utf-8') as f:

writer = csv.DictWriter(f, fieldnames=keys)

writer.writeheader()

writer.writerows(data)

print("数据已成功写入CSV文件！")