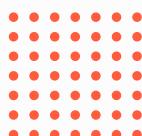
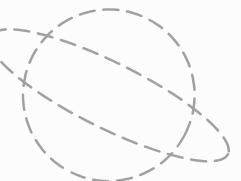




今天你吃了吗？

定福庄周边美食推荐系统与可视化



目 录

CONTENTS

01. 工作概述
WORK OVERVIEW

02. 数据获取
DATA CRAWLING

03. 算法设计
ALGORITHM DESIGN

04. 前端展示
FRONT-END DEMO



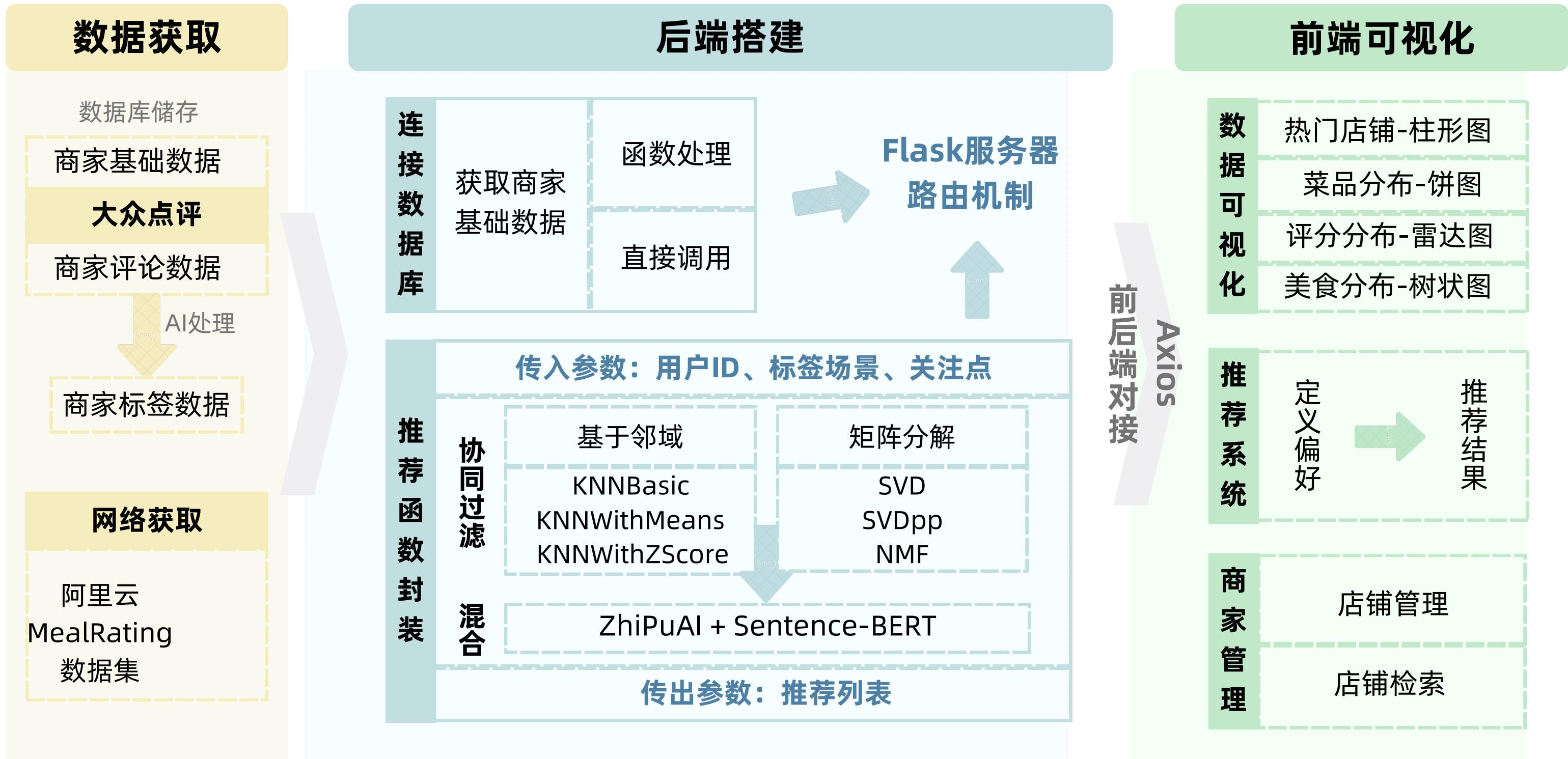
WORK OVERVIEW

工作概述

PART ONE

OVERVIEW

定福庄周边美食推荐与可视化





DATA CRAWLING

数据获取

数据获取与存储

PART TWO

数据获取

Filter 750 rows...						
商家ID	商家链接	标签	位置	评论数量	人均消费	评分
G6M6693H...	https://www.d...	老北京火锅	高碑店	8159	108	4.6
G8t0h13QU...	https://www.d...	小吃	传媒大学	1683	33	4.2
GalUhNn3E...	https://www.d...	烧烤串串	双桥	1346	90	4.1
I16fzkaLDT...	https://www.d...	咖啡厅	高碑店	838	49	4.2
I9V0ZQHVQ...	https://www.d...	海鲜火锅	双桥	2372	118	4.8
I6bBWh7JGt...	https://www.d...	潮汕牛肉...	高碑店	1602	119	4.8
k8exKy5h5...	https://www.d...	重庆火锅	定福庄	1804	83	4.8
HaYDUGdCf...	https://www.d...	重庆火锅	高碑店	7829	134	4.8
H2k8xu85q...	https://www.d...	融合烤肉	双桥	4108	116	4.6
G4errGiPQ...	https://www.d...	粉面馆	高碑店	908	32	4.3
kauqnKQQs...	https://www.d...	家常菜	传媒大学	967	66	4.6
H6tON4EX5...	https://www.d...	京菜	双桥	2684	120	4.4
I3S71o6YTr...	https://www.d...	京菜	高碑店	1478	160	4.6
H1MpED3O...	https://www.d...	江西菜	传媒大学	2186	95	4.6
H8LPKAkbUL...	https://www.d...	面包烘焙	高碑店	49	68	4.2
I63UUVTig6...	https://www.d...	融合烤肉	高碑店	1506	89	4.5
F5Gm3rFB3...	https://www.d...	川菜馆	双桥	2357	86	4.8
H4lxMZQ3d...	https://www.d...	咖啡厅	高碑店	157	47	4.4
G3EwciCCV...	https://www.d...	日本料理	定福庄	2390	125	4.8
G1mVTkHw...	https://www.d...	日本料理	双桥	402	150	4.6
G1sgFEN1j...	https://www.d...	新疆菜	传媒大学	3573	78	4.3
jLK5BQE4H...	https://www.d...	烤鸭	传媒大学	1838	89	4.7
k1VjM3Beg...	https://www.d...	四川火锅	定福庄	3282	121	4.7
k5Z1oHfuJv...	https://www.d...	烤鱼	双桥	3327	106	4.6

店铺ID	评论时间	评论内容
ArjNNxh92oE	2024-11-21 23:13	这家店在1919创意园里的麻辣烫，和杨国福那种做法一样，但是我感到口味：味道很好！自助小料齐全堪比火锅啦！菜品也算丰富的，看着新餐厅的口味正宗地道。环境卫生干净
ArjNNxh92oE	2024-11-20 18:11	ArjNNxh92oE 2024-12-05 19:59 鸡架很正宗，酥脆可口，店里很干净，菜品也很丰富，宝藏小店可以尝试一下
ArjNNxh92oE	2024-12-04 18:13	ArjNNxh92oE 2024-11-29 18:59 味道不错，卖相也挺好！很多材料看起来挺新鲜。在园区里，比较好找
ArjNNxh92oE	2024-11-23 19:54	ArjNNxh92oE 2024-11-18 19:05 味道很不错，品种很丰富，给力，下次继续来，鸡架很香
ArjNNxh92oE	2024-11-18 19:05	环境干净整洁，菜品好吃实惠，非常好吃的一家麻辣烫，菜品的口味超级好，店里很干净，价格实惠，食材新鲜，麻辣烫麻味拌种类丰富
ArjNNxh92oE	2024-11-16 19:10	ArjNNxh92oE 2024-11-11 15:56 经常吃，好吃味道很好价格实惠菜品非常干净好评
ArjNNxh92oE	2024-11-10 16:26	ArjNNxh92oE 2024-11-08 18:54 店内很干净，食材新鲜种类也很多，味道那叫一个地道会常来出差北京住在附近，饿了就随便吃一口，菜品不错的。主要是干净，
ArjNNxh92oE	2024-11-08 18:54	文创园里的宝藏小吃店串串种类很多价格实惠服务周到环境也超级舒服，出差北京住在附近，饿了就随便吃一口，菜品不错的。主要是干净，
ArjNNxh92oE	2024-11-02 20:55	ArjNNxh92oE 2024-10-09 12:41 救命啊好咸好咸好咸，他们家放盐好像不要钱，咸得我舌头都掉了
ArjNNxh92oE	2024-09-28 15:57	无意中发现的这家宝藏小店，在2049文创园里，小店很文艺范，环境
ArjNNxh92oE	2024-11-29 18:56	[薄荷]环境：在半地下，还是比较安静的一个麻辣烫，中午人多，晚上一家做麻辣烫和串串的店子点了他家的原汤，但是我觉得原汤味道有点
ArjNNxh92oE	2024-09-18 12:26	园区里的麻辣烫，店面不大，但是很干净，老板是东北的。也是老乡，
ArjNNxh92oE	2024-07-21 23:58	在双桥附近的园区里，中午有很多人来吃饭。按重量称了算钱。我在双桥附近的园区里，中午很多人来吃饭。按重量称了算钱。我
ArjNNxh92oE	2024-04-26 16:11	在北京吃到的最正宗的东北麻辣烫，就在双桥这边的商业园区内，还
ArjNNxh92oE	2024-04-25 18:13	一个很辣的店铺，总体都很差劲，他家豆皮很贵都别吃啊远离远
ArjNNxh92oE	2024-09-17 13:10	「甘肃麻辣烫」惊喜的小馆子，干净，卫生还有各种麻辣烫、麻辣拌
ArjNNxh92oE	2024-09-09 15:33	园区里的麻辣烫还是可以的，店不大很干净，晚上去吃的，不错的[薄
ArjNNxh92oE	2024-06-22 17:56	开在2049文创园里面的甘肃天水麻辣烫，正宗好吃
ArjNNxh92oE	2024-06-03 19:02	超有食欲的甘肃麻辣烫，他家麻辣烫味道也是特别香，食材种类真的
ArjNNxh92oE	2024-03-31 13:13	在2049里边，为数不多的小饭馆，所以工作日人很多。整体种类还挺多
ArjNNxh92oE	2024-05-14 16:16	园区里的店，很干净，种类很多，价格也比较实惠，点的是甘肃麻辣
ArjNNxh92oE	2024-05-09 16:03	麻辣烫超好吃，汤鲜味美！就在2049园区里很不错的一家店
ArjNNxh92oE	2024-05-07 19:54	五一活动在园区里转转，饭点附近对付一口。价格算比较实惠的，25块
ArjNNxh92oE	2024-05-05 20:05	麻辣烫味道真不错，环境很好，推荐推荐
ArjNNxh92oE	2024-03-17 20:41	店内有100多种菜品，看着很新鲜，室内装修温馨，明档操作，干净卫
ArjNNxh92oE	2024-03-15 20:51	新开的店，整体很干净，环境卫生不错一家人经营，有麻辣烫，有炸
ArjNNxh92oE	2024-02-07 18:42	环境非常好，很好吃，推荐，
ArjNNxh92oE	2024-12-04 21:16	完全不会踩雷的一家店，大家放心大胆来，跟描述的基本没差
ArjNNxh92oE	2024-04-22 17:22	这个麻辣烫来很多次了每次也都点很多味道太好了老板也很热情
ArjNNxh92oE	2024-04-22 17:22	同事推荐好几次了今天终于吃上了自己调味可真不错种类多，环境干
ArjNNxh92oE	2024-03-15 20:29	老板人品真的是很差劲啊，头天打电话想大量预订，都说好了老板也打
ro580aWbFyA	2024-12-11 11:37	超市出来电梯口就是，买了栗子和糖葫芦，基本没有坏果，而且主要是
ro580aWbFyA	2024-12-08 17:33	栗子非常好吃，没有坏的，里面是金黄的，最主要的是好剥皮，不上手
ro580aWbFyA	2024-12-04 13:43	手

ReviewID	Review	ReviewTime	ReviewTitle	ReviewContent	MealRating	ReviewType	ReviewScore
A2WOH395IH	5 风味独特，真的不错！	1685479456	B0040HNZTW	凉拌血旺	其他	17	
A32KH50VN0I	3 有特色，也比较卫生	1685479508	B006Z48Tzs	手撕茄子	素菜	12	
A1YQ4Z5U9N	5 家常美味，推荐！	1685479676	B00CDBTQcw	素鱼焗苦瓜	素菜	10	
A3E5V5TSTAY	4 好吃	1685481656	B00751YQ4	栗子焖鸡	鸡肉	39	
A1V50CTTDJ7	5 不得不赞	1685482409	B00COOLT6S	广式肠粉	其他	10	
A3GD8QRSS0	5 每周必点的一道菜	1685482456	B006WC63X8	小炒黄瓜	素菜	10	
AK6G2ZYJVQ4	5 同事们都很喜欢	1685482498	B009JKR6U	黑椒芦笋牛扒	牛肉	24	
A2FPYIU94BN	5 性价比高	1685484440	B0087YNHNI	韩式拌饭	主食	15	
A2WOH395IH	5 有惊喜	1685485465	B002A493NY	葱油莴笋	素菜	16	
AM0LNVIXZI	3 味道一般	1685486400	B002KGEOFM	蒸鸡蛋羹	蛋类	10	
A1VOJC5KLSC	5 太美味了，强烈推荐！	1685486400	B0045TGRSG	干锅香鸡	鸡肉	27	
A2FPYIU94BN	3 味道很正	1685486400	B0073J5X58	芦笋炒鱼片	鱼	19	
AMXDV160YD	5 太美味了，强烈推荐！	1685486400	B006Z48Tzs	手撕茄子	素菜	12	
A33T3FTA07C	4 非常非常好吃	1685486400	B0071E6J30	炝炒土豆丝	素菜	13	
A37FUJC2L7D	5 太美味了，强烈推荐！	1685486400	B00FWLG2MM	蓝莓酸奶司康	其他	16	
A1GH5XCC5D	5 太美味了，强烈推荐！	1685486400	B007JF83YY	蜜汁叉烧	猪肉	18	
A3GUNEINK3	5 太美味了，强烈推荐！	1685486400	B007JF83YY	蜜汁叉烧	猪肉	18	
A79TS2VPN9D	5 太美味了，强烈推荐！	1685486400	B007JF83YY	蜜汁叉烧	猪肉	18	
ALGXUXW3P7JI	4 非常非常好吃	1685486400	B007VQLRGW	木瓜豌豆鸡丁	鸡肉	18	
A1V50CTTDJ7	5 太美味了，强烈推荐！	1685486400	B008QTTGGG	傣味鱼皮	鱼	11	
A1V50CTTDJ7	5 太美味了，强烈推荐！	1685486400	B009FZFONO	高汤	汤品	15	
A21ZL2UCY9S	5 太美味了，强烈推荐！	1685486400	B000OJFV3S	橙汁鸡球	鸡肉	36	
AM0LNVIXZI	4 非常非常好吃	1685486400	B001F6ZIXC	蘑菇酸丸子	其他	14	
A1VX5R7X7X	4 非常非常好吃	1685486400	B0025ZBG4K	炝拌土豆丝	素菜	15	
A3HGRD34E	4 非常非常好吃	1685486400	B002Y0TGOU	毛豆肉丁	猪肉	25	
ABLBRUT8DAl	5 太美味了，强烈推荐！	1685486400	B003VDX6MW	黄金芝士虾球	素菜	29	
A3HU3OP8Y9Y	4 非常非常好吃	1685486400	B000ESNKDHA	家常鱼香肉丝	鱼	18	
AAD3KLMJH6	5 太美味了，强烈推荐！	1685486400	B000ETKG8AI	牛排	牛肉	29	
A1V50CTTDJ7	5 太美味了，强烈推荐！	1685486400	B008BQG3RE	土豆焖鸡肉	鸡肉	19	
A1ALCXU70A	4 非常非常好吃	1685486400	B000GLYM3QS	蒜蓉干锅虾	海鲜	23	
A2XIK5MBJ6Q	5 太美味了，强烈推荐！	1685486400	B00F40652U	芝士蟹	海鲜	28	
A3TR9E9GZOC	5 太美味了，强烈推荐！	1685486400	B00F49E806	五花肉烧茄子	猪肉	16	
A1RPTVW5VEI	5 太美味了，强烈推荐！	1685486400	B00H8C57GA	瑞士排骨	猪肉	29	
A2WOH395IH	5 太美味了，强烈推荐！	1685486400	B0030URVUO	干煸有机菜花	素菜	16	
A1Y3NTL7XXII	5 太美味了，强烈推荐！	1685486400	B00BLTE				

店铺获取

```
# '标签' + '位置'
def get_tag_loc(res_text):

    tag_temp = re.findall('<span class="tag">.+</span></a>\n            <em class="sep">',res_text)
    tag_temp = [v.replace('</a>\n            <em class="sep">','') for v in tag_temp]
    loc_temp = re.findall('</em>\n            .+(<span class="tag">.+</span>)',res_text)

    tag = [[v.replace('<svgmtsi class="tagName">','') for v in n] for n in [re.findall('>(.+?)<',
        loc = [[v.replace('<svgmtsi class="tagName">','') for v in n] for n in [re.findall('>(.+?)<',
            tag = [v[0] for v in tag]
            loc = [v[0] for v in loc]

            return tag, loc

# 设置链接
url = 'https://www.dianping.com/search/keyword/2/0_%E4%BC%A0%E5%AA%92%E5%A4%A7%E5%AD%90'

# 设置 headers
# User-Agent 与 Cookie 替换为对应内容
headers = {
    'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.130 Safari/537.36',
    'Cookie': '_lx_utm=utm_source%3DBaidu%26utm_medium%3Dorganic; _lxsd़_cuid=193921'
}

# 进行访问
res = requests.get(url, headers=headers)

# 当访问报错时停留 5s 继续访问
while res.status_code != 200:
    print('正在尝试访问，请稍等', end='\r')
    print('请等一下', end='\r')
    time.sleep(5)
    res = requests.get(url, headers=headers)
```

```
# 爬取评分信息
# 获取店铺名称
title = get_name_url(res_text[i])[0]
# 获取店铺 shopid
shopid = [v.split('/')[-1] for v in get_name_url(res_text[i])[1]]
for v in shopid:
    # 设置链接
    url = 'https://www.dianping.com/ajax/json/shopDynamic/reviewAndStar'
    # 设置 params
    # 打开 F12 找到 'https://www.dianping.com/ajax/json/shopDynamic/reviewAndStar' 对应
    # token 与 uuid 替换为对应内容
    params = {
        'shopId': v,
        'cityId': '1',
        'mainCategoryId': '34245',
        '_token': '71zf92j5s5ipkhamod13',
        'uuid': '72ac3770-1829-4496-97ca-248dd73d83ed',
        'platform': '1',
        'partner': '150',
        'optimusCode': '10',
        'originUrl': 'https://www.dianping.com/shop/' + v
    }
    # 设置 headers
    # User-Agent 与 Cookie 替换为对应内容
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.130 Safari/537.36',
        'Cookie': 's_ViewType=10; _lxsd़_cuid=190affabb7cc8-0c5f98b9c0905e-26001f51-1bc4-0000-0000-000000000000'
    }
    # 进行访问
    res = requests.get(url, headers=headers, params=params)
```

评论获取

```
def get_comment(contents, re_, comment_list):
    comments = re_.finditer(contents)
    for comment_ in comments:
        raw_data = comment_.group("comment").strip()
        processed_data = re.sub(r"\xA;|<img.*?/>", "", raw_data)
        comment_list.append(processed_data)

    return comment_list

def get_time(contents, re_, time_list):
    times = re_.finditer(contents)
    for time_ in times:
        raw_data = time_.group("time").strip()
        processed_data = re.sub(".*?更新于|' '", "", raw_data, flags=re.S)
        time_list.append(processed_data)

    return time_list

def load_comments(contents, re_1, re_2, re_3, re_4, csv_writer, id):
    # 获取评论内容的部分
    comment_list = []
    time_list = []
    # 获取 hide 的评论
    comment_list = get_comment(contents, re_1, comment_list)
    time_list = get_time(contents, re_2, time_list)
    # 获取未 hide 的评论
    comment_list = get_comment(contents, re_3, comment_list)
    time_list = get_time(contents, re_4, time_list)
    print(time_list)
    for comment_, time_ in zip(comment_list, time_list):
        csv_writer.writerow([id, time_, comment_])

    time.sleep(5)

    return True
```



```
import sqlite3
conn = sqlite3.connect('sql/商家信息.db')
cursor = conn.cursor()
query = "SELECT 商家ID FROM shops LIMIT 5 OFFSET 213"
cursor.execute(query)
shopid = [row[0] for row in cursor.fetchall()]
cursor.close()
conn.close()

for code in shopid:
    # 定义参数
    user_agent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.6514.100 Safari/537.36'
    cookie = 's_ViewType=10; _lxsd_k_cuid=190affabb7cc8-0c5f98b9c0905e-26001f51-1bcab9-190affabb7cc8; _lxsd_k=190affabb7cc8-0c5f98b9c0905e-26001f51-1bcab9-190affabb7cc8'
    min_num = 0
    max_num = 12
    headers = update_headers(user_agent, cookie)

    # 正则表达式
    re_1_comment = re.compile(r'<div class="review-words Hide">(?P<comment>.*)<div class="less-words">', re.S)
    re_1_time = re.compile(r'<div class="review-words Hide">.*)<span class="time">(?P<time>.*)</span>', re.S)
    re_2_comment = re.compile(r'<div class="review-words">(?P<comment>.*)</div>', re.S)
    re_2_time = re.compile(r'<div class="review-words">.*)<span class="time">(?P<time>.*)</span>', re.S)

    for i in tqdm(range(min_num, max_num)):
        resp = requests.get(f"https://www.dianping.com/shop/{code}/review_all/p{i}", headers=headers)
        resp.encoding = "utf-8"
        contents = resp.text
        load_comments(contents,
                      re_1=re_1_comment,
                      re_2=re_1_time,
                      re_3=re_2_comment,
                      re_4=re_2_time,
                      csv_writer=csv_writer,
                      id=code)
```

PART TWO

ALGORITHM DESIGN

算法设计

推荐算法

PART THREE

推荐算法

设计思路

基于surprise的协同过滤

1. 阿里云MealRating数据集：训练集 - 测试集。协同过滤推荐算法 → 推荐菜品MealList。
2. 提取shop菜品：利用智谱清言API提取shop评论中的菜品。

基于内容推荐

1. **菜品相似度分数**：计算店铺描述与为用户推荐的菜品描述的相似度，保留每道菜品下相似度得分靠前的20个店铺
2. **场景相似度分数**：如店铺场景标签包含用户输入场景，加1分
3. **关注点相似度分数**：如店铺在用户关注点的评分表现 >4 ，加1分
4. **总体评分**：如店铺总分 >4.5 ，加1分；大于4.3，加0.5分
5. **最终推荐分加和**：每一步相似度得分加权得最后店铺推荐分，分别取每道菜品下推荐分最高的1个店铺，得到最终推荐的5个店铺

基于已有数据集，无法直接推荐店铺（没有user对应的shop评分），只能根据阿里云数据集推荐菜品，然后映射到shop，并根据shop标签微调权重。

无法根据店铺已有菜品数据直接计算该店铺与为用户推荐菜的相似度，故调用大模型生成用户推荐菜描述、基于店铺菜品生成店铺描述后，再计算二者相似度。

为避免最后推荐店铺结果仅相似于一道推荐菜品，将推荐菜品分开放取相似店铺。

PART THREE

协同过滤

```
file_path = 'MealRating.csv'
df = pd.read_csv(file_path)
df['Rating'] = pd.to_numeric(df['Rating'], errors='coerce')
data = df[['UserID', 'MealID', 'Rating']].dropna()
reader = Reader(rating_scale=(1, 5))
dataset = Dataset.load_from_df(data, reader)
trainset, testset = train_test_split(dataset, test_size=0.2, random_state=42)

algo = KNNBasic()
algo.fit(trainset)

test_predictions = algo.test(testset)
print(f"Model RMSE: {accuracy.rmse(test_predictions)})")
```

RMSE: 1.0945
Model RMSE: 1.0944843063010792



```
def evaluate_algorithms(data):
    algorithms = {
        'KNNBasic': KNNBasic(),
        'KNNWithMeans': KNNWithMeans(),
        'KNNWithZScore': KNNWithZScore(),
        'SVD': SVD(),
        'SVDpp': SVDpp(),
        'NMF': NMF()
    }

    results = {}
    for name, algo in algorithms.items():
        print(f"\nEvaluating {name}...")
        cv_results = cross_validate(algo, data,
                                     measures=['RMSE', 'MAE'],
                                     cv=5, verbose=False)
        results[name] = {
            'RMSE': cv_results['test_rmse'].mean(),
            'MAE': cv_results['test_mae'].mean()
        }
    return results
```

背景

- 在推荐系统中，RMSE在0.8-1.2之间通常被认为是可接受的
- NETFLIX PRIZE竞赛的获奖方案RMSE为0.8567
- 餐饮推荐系统中，由于个人口味差异大，RMSE通常稍高



	RMSE	MAE
KNN Basic	1.1039	0.8024
KNN With Means	1.0628	0.7467
KNN With Z Score	1.0675	0.7489
SVD	0.9753	0.7439
SVD pp	0.9699	0.7293
NMF	1.1074	0.8467

PART THREE

协同过滤

```
def optimize_svd_params(data):
    param_grid = {
        'n_epochs': [20, 30, 40],
        'lr_all': [0.005, 0.01],
        'reg_all': [0.02, 0.1, 0.4]
    }

    gs = GridSearchCV(SVD, param_grid, measures=['rmse', 'mae'], cv=5)
    gs.fit(data)

    return gs.best_score['rmse'], gs.best_params['rmse']
```

Best SVD parameters: {'n_epochs': 40, 'lr_all': 0.005, 'reg_all': 0.1}
Best RMSE: 0.9599



```
def train_final_model(data, best_params):
    trainset, testset = train_test_split(data, test_size=0.2, random_state=42)

    algo = SVD(
        n_epochs=best_params['n_epochs'],
        lr_all=best_params['lr_all'],
        reg_all=best_params['reg_all']
    )

    algo.fit(trainset)
    predictions = algo.test(testset)
    return algo, accuracy.rmse(predictions), accuracy.mae(predictions)
```

RMSE: 0.9652
MAE: 0.7272
Final Model RMSE: 0.9652
Final Model MAE: 0.7272

优化

- 定义SVD算法的参数网格，包括：迭代次数（20, 30, 40），学习率（0.005, 0.01），正则化参数（0.02, 0.1, 0.4）。
- 使用GRIDSEARCHCV进行网格搜索，通过5折交叉验证找到最佳参数组合。



训练

- 将数据集划分为训练集（80%）和测试集（20%）。
- 使用优化后的参数初始化SVD算法。
- 在训练集上训练模型，并在测试集上进行预测。
- 返回训练好的模型、测试集上的RMSE和MAE。

PART THREE

协同过滤

```
def get_top_n_recommendations(algo, user_id, n=5):
    user_ratings = df[df['UserID'] == user_id]['MealID'].unique()
    all_items = df['MealID'].unique()
    items_to_predict = np.setdiff1d(all_items, user_ratings)

    predictions = []
    for item_id in items_to_predict:
        pred = algo.predict(user_id, item_id)
        predictions.append((item_id, pred.est))

    predictions.sort(key=lambda x: x[1], reverse=True)

    top_n = predictions[:n]

    recommended_meals = []
    for meal_id, pred_rating in top_n:
        meal_info = df[df['MealID'] == meal_id].iloc[0]
        recommended_meals.append({
            'MealID': meal_id,
            'MealName': meal_info['mealName'],
            'MealType': meal_info['mealType'],
            'MealPrice': meal_info['mealPrice'],
            'PredictedRating': round(pred_rating, 2)
        })

    return recommended_meals
```

Generating recommendations for user: A2W0H395IHGS0T

Top 5 Recommended Meals:

1. 烤排骨 (猪肉) - ¥29
| Predicted Rating: 5
2. 菠萝培根卷 (其他) - ¥15
| Predicted Rating: 5
3. 湖南菜紫油姜 (其他) - ¥12
| Predicted Rating: 5
4. 酥皮香肠卷 (饼) - ¥8
| Predicted Rating: 5
5. 傣家柠檬鱼 (鱼) - ¥26
| Predicted Rating: 5

Dataset Statistics:

Total number of users: 5130
Total number of meals: 1685
Total number of ratings: 38384
Rating sparsity: 99.56%

分析

- 评分稀疏度为99.56%（用户-菜品评分矩阵中未被评分的比例）。
- 基于矩阵分解的方法（如SVD、SVDPP）能够有效地捕捉潜在特征，所以在高稀疏度环境下表现较好。
- 结合内容推荐（利用菜品的属性信息）和协同过滤，可以在高稀疏度下提升推荐效果。

基于内容推荐

```
prompt = f"请对以下菜品生成一条描述: {comment}。
```

要求: 含原料、炒制方法、口感、菜系等, 不需要复述菜品名称, 以“一道.....”开始描述。

至少要求50个字, 客观描述。

示例输入: “小鸡炖蘑菇”

示例输出: 一道经典的东北菜, 以其鲜美的味道和丰富的营养而广受欢迎。这道菜的主要食材包括鸡肉、干蘑菇和粉条, 烹饪方法主要是炖煮。鸡肉的鲜嫩与蘑菇的香味相结合, 使得这道菜口感丰富, 味道浓郁。”

```
prompt = f"请对以下店铺生成一条描述: {comment}。
```

要求: 强调店铺的菜系, 以及两道招牌菜的介绍。招牌菜介绍中包括菜系、原料、制作方法、口感, 每道菜描述至少40字。

示例输入: “烤羊腿老虎菜, 骨肉相连, 小红茶, 手工猪肉大葱水饺, 烤牛蹄筋, 烤肥肠, 烤蚕蛹, 炒方便面, 烤牛板

示例输出: 该店铺菜系为烤羊腿, 招牌菜1是手工猪肉大葱水饺, 是东北菜系, 原料为猪肉馅、大葱、饺子皮,

```
try:
```

```
    response = client.chat.completions.create(  
        model="glm-4-flash",  
        messages=[  
            {"role": "system", "content": "你是一个专门生成店铺描述的助手。"},  
            {"role": "user", "content": prompt}  
        ],  
        temperature=0.1,  
        top_p=0.7,  
        max_tokens=400  
    )  
    feature_text = response.choices[0].message.content.strip()  
    return {"description": feature_text}
```

处理进度: 713/715

mealall: 比萨奶油蘑菇汤, 传统披萨, 小龙虾, 海盐巧克力蛋糕, 培根披萨, 奥尔良鸡肉披披萨, 棒约翰, 川香鸡块, 朗姆栗子蛋糕, 星星薯饼, 提拉米苏, 羊肚菌菌菇鸡肉披萨, 芝士酱莲披萨

店铺描述: 该店铺菜系为意式披萨与中式烧烤, 招牌菜1是比萨奶油蘑菇汤, 是意式西餐, 2是奥尔良鸡肉披萨, 是意式披萨, 选用奥尔良腌制的鸡肉, 搭配新鲜蔬菜和芝士, 制作时

处理进度: 1673/1680

菜品名称: 芝麻泡菜

菜品描述: 一道融合了四川风味的传统小吃, 以鲜嫩的豆腐、爽口的泡菜和香脆的芝麻为原料, 麻的香浓, 是川菜中的佳肴。

处理进度: 1674/1680

菜品名称: 芝士土豆泥

菜品描述: 一道融合了芝士香浓与土豆软糯的经典西式菜品。选用新鲜土豆, 经过精心炒制, 是西餐中的经典佳肴。

菜品与店铺描述生成

- 调用智谱清言大模型生成菜品描述、基于店铺菜品生成店铺描述
- 为后续计算店铺与推荐菜品的相似度提供计算依据

基于内容推荐

```
# 第一步：算菜品相似度
def calc_food_similarity(shop_description, user_description):
    if pd.isna(shop_description): # 检查是否为NaN
        return 0.0
    shop_embedding = model.encode(shop_description, convert_to_tensor=True)
    user_embedding = model.encode(user_description, convert_to_tensor=True)
    cosine_scores = util.pytorch_cos_sim(shop_embedding, user_embedding)
    return float(cosine_scores[0]) * 10 # 转换为10分制

# 第二步：算场景相似度
def calc_scene_similarity(scene_tags, user_scene):
    if pd.notna(scene_tags) and user_scene in scene_tags:
        return 1
    else:
        return 0

# 第三步：算关注点相似度
def calc_focus_similarity(focus_score, focus_point):
    if pd.notna(focus_score) and focus_score > 4:
        return 1
    else:
        return 0

# 第四步：算总体得分
def calc_overall_score(total_score):
    if pd.isna(total_score):
        return 0
    elif total_score > 4.5:
        return 1
    elif total_score > 4.3:
        return 0.5
    else:
        return 0
```

相似度计算函数

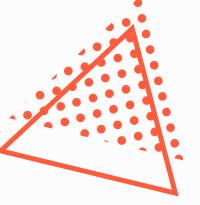
- 在计算菜品相似度环节，调用SENTENCE-BERT模型，获取两个描述文本的相似度得分，并转化为10分制

```
# 计算每个描述与所有店铺的【菜品相似度得分】，并保留前20个店铺
top_shops_per_description = {}
for description in user_food_descriptions:
    df_shop_info['菜品相似度得分'] = df_shop_info['店铺描述'].apply(lambda x: calc_food_similarity(x, description))
    top_20_shops = df_shop_info.nlargest(20, '菜品相似度得分').copy()
    top_shops_per_description[description] = top_20_shops

# 对每个描述中的20个店铺进行后续计算，并选出推荐分最高的店铺
recommended_shops = []
for description, top_20_shops in top_shops_per_description.items():
    # 对这20个店铺进行第二三四五步的其他相似度计算
    top_20_shops['场景相似度得分'] = top_20_shops.apply(lambda row: calc_scene_similarity(str(row['场景标签']), u
    top_20_shops['关注点相似度得分'] = top_20_shops.apply(lambda row: calc_focus_similarity(row[user_focus_point])
    top_20_shops['总体评分'] = top_20_shops.apply(lambda row: calc_overall_score(row['总分']), axis=1)

    # 计算综合推荐分
    top_20_shops['推荐分'] = (
        top_20_shops['菜品相似度得分'] * 0.4 +
        top_20_shops['场景相似度得分'] * 0.4 +
        top_20_shops['关注点相似度得分'] * 0.5 +
        top_20_shops['总体评分'] * 0.1
    )

    # 从这20个店铺中选出推荐分最高的一个店铺
    best_shop = top_20_shops.nlargest(1, '推荐分')
    recommended_shops.append(best_shop[['店铺名称', '菜品相似度得分', '场景相似度得分', '关注点相似度得分', '总体评分',
```



FRONT-END DEMO

前后端搭建

flask + vue + echarts

PART FOUR



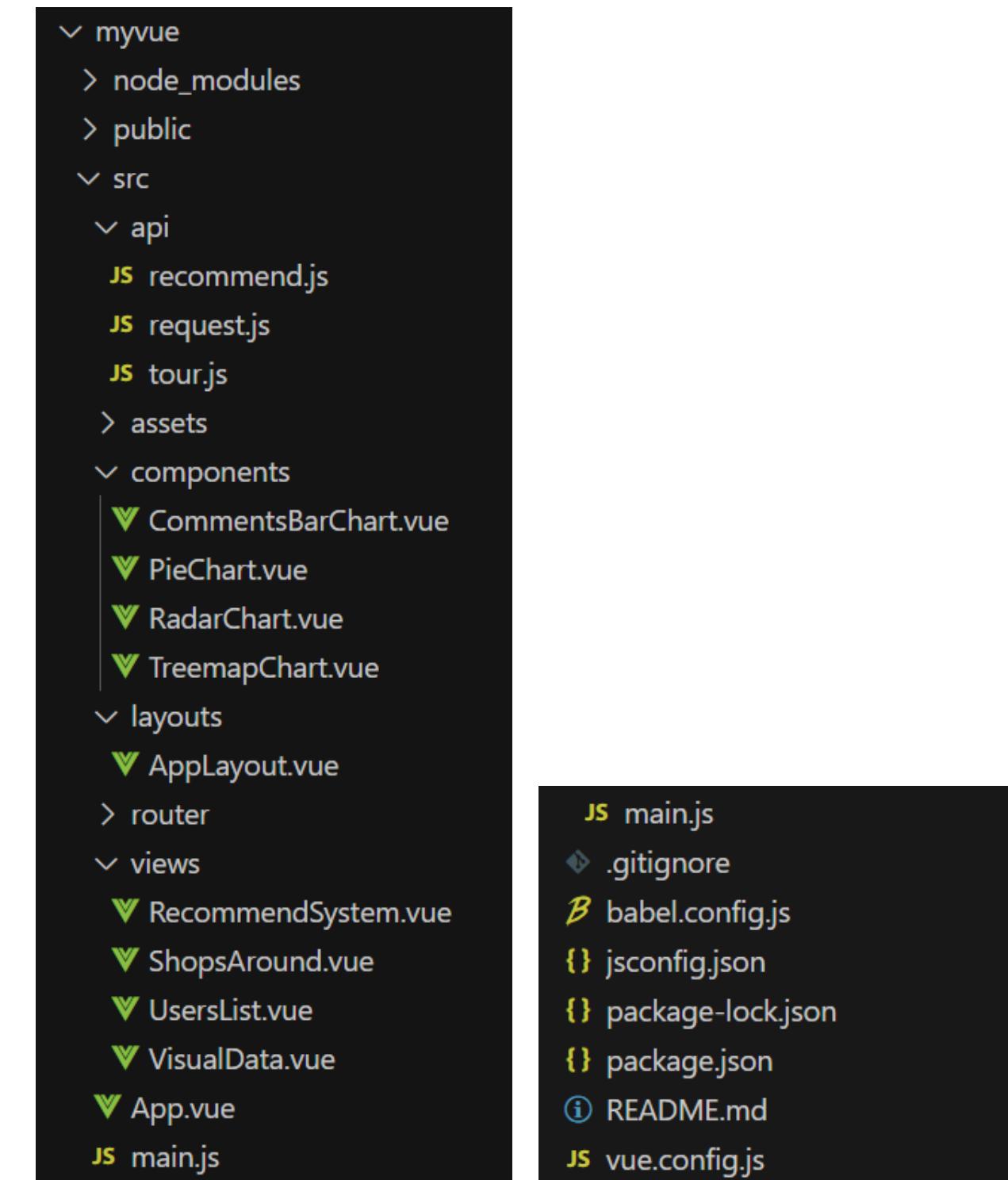
前端可视化

前端框架搭建与页面编写

- 采用VUEJS作为前端开发框架构建用户界面，页面的HEADER部分通过的路由机制为不同的页面分配相应的路径，开发了商家信息可视化页面、美食推荐系统页面和周边商家管理与搜索页面，通过前端的导航设计使用户能够便捷地在不同页面间切换。

```
<template>
  <el-container class="container" >
    <el-aside class="aside" width="200px">
      <el-menu :default-active="activeIndex"
        background-color="#545c64"
        text-color="#000"
        active-text-color="#ffd04b"
        class="el-menu-vertical">
        <el-menu-item index="/VisualData" @click="navigateTo('/VisualData')">
          <i class="el-icon-s-marketing"></i>
          数据可视化</el-menu-item>
        <el-menu-item index="/RecommendSystem" @click="navigateTo('/RecommendSystem')">
          <i class="el-icon-star-off"></i>
          推荐系统</el-menu-item>
        <el-menu-item index="/ShopsAround" @click="navigateTo('/ShopsAround')">
          <i class="el-icon-star-on"></i>
          周边店铺</el-menu-item>
        <!-- 其他菜单项 -->
      </el-menu>
    </el-aside>
    <el-container>
```

AppLayout.vue



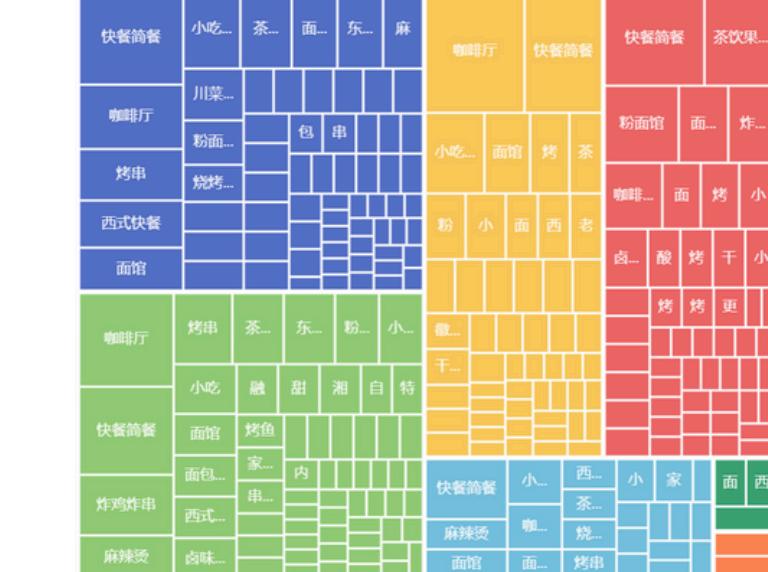
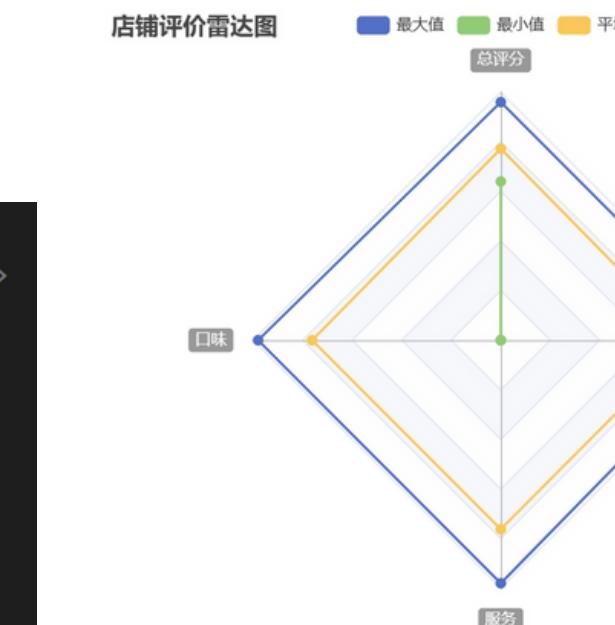
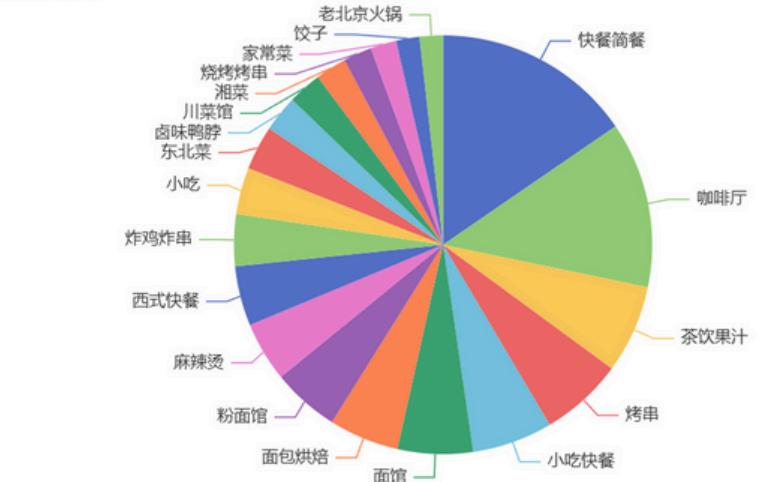
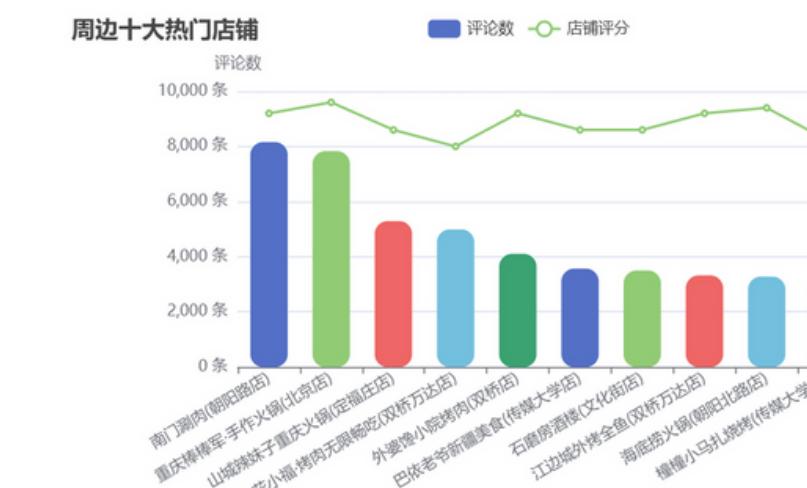
PART FOUR

前端可视化

Echarts可视化图表

- 引入ECHARTS库构建一个可视化数据页面的布局
 - 根据评论数选取热度前十的商家，通过柱状图和折线图组合的形式对比热门商家的评分情况
 - 根据商家菜品类型数量分布情况制作饼图
 - 对总评分、环境评分、口味评分和服务评分四个维度绘制展现最大值、最小值和平均值的雷达图
 - 使用树状图展示不同商圈的菜品类型数量分布

```
<el-row :gutter="15">
  <!-- Top left chart (Bar Chart) -->
  <el-col :span="12">
    <div class="chart" style="height: 400px;">
      <CommentsBarChart/>
    </div>
  </el-col>
  <!-- Top right chart (Pie Chart) -->
  <el-col :span="12">
    <div class="chart" style="height: 350px;">
      <PieChart/>
    </div>
  </el-col>
</el-row>
</el-row>
<el-row :gutter="20" style="margin-top: 20px;">
  <el-col :span="12">
    <div class="chart" style="height: 400px;">
      <RadarChart/>
    </div>
  </el-col>
  <el-col :span="12">
    <div class="chart" style="height: 300px;">
      <TreemapChart/>
    </div>
  </el-col>
</el-row>
```



PART FOUR

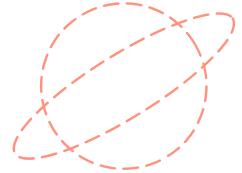
前端可视化

周边商家展示与搜索

- VUE 的数据绑定、组件通信和事件处理功能，结合 ELEMENT UI 的组件，提供了一个基本的周边店铺管理页面框架，包含了店铺信息的展示、搜索、添加、编辑和删除操作，以及分页功能

```
<template>
  <div class="tours-container">
    <el-card class="box-card">
      <div slot="header" class="header">
        <span class="header-title">周边店铺管理</span>
        <div class="header-controls">
          <el-input v-model="searchTitle" placeholder="输入标题进行搜索" class="search-input"></el-input>
          <el-button type="primary" @click="fetchData">搜索</el-button>
          <el-button type="success" @click="handleAddTour">添加店铺</el-button>
        </div>
      </div>
      <div>
        <el-table :data="shops" style="width: 100%">
          <el-table-column prop="商家名称" label="商家名称" width="180"></el-table-column>
          <el-table-column prop="商家ID" label="商家ID" width="150"></el-table-column>
          <el-table-column prop="商家链接" label="商家链接" width="150"></el-table-column>
          <el-table-column prop="标签" label="标签" width="150"></el-table-column>
          <el-table-column prop="位置" label="位置" width="150"></el-table-column>
          <el-table-column prop="评论数量" label="评论数量" width="120"></el-table-column>
          <el-table-column prop="人均消费" label="人均消费" width="120"></el-table-column>
          <el-table-column prop="评分" label="评分" width="120"></el-table-column>
          <el-table-column prop="口味" label="口味" width="120"></el-table-column>
          <el-table-column prop="环境" label="环境" width="120"></el-table-column>
          <el-table-column prop="服务" label="服务" width="120"></el-table-column>
          <el-table-column label="操作" min-width="180">
            <template slot-scope="scope">
              <el-button @click="handleEditTour(scope.row)" type="text" size="small">编辑</el-button>
              <el-button @click="handleDeleteTour(scope.row)" type="text" size="small">删除</el-button>
            </template>
          </el-table-column>
        </el-table>
      </div>
    </el-card>
  </div>

```



```
</el-card>
<el-dialog title="编辑店铺" :visible.sync="dialogVisible">
  <el-form :model="form">
    <el-form-item label="商家名称" :label-width="formLabelWidth">
      <el-input v-model="form.name"></el-input>
    </el-form-item>
    <el-form-item label="商家ID" :label-width="formLabelWidth">
      <el-input v-model="form.alias"></el-input>
    </el-form-item>
    <el-form-item label="商家链接" :label-width="formLabelWidth">
      <el-input v-model="form.alias"></el-input>
    </el-form-item>
    <el-form-item label="标签" :label-width="formLabelWidth">
      <el-input v-model="form.alias"></el-input>
    </el-form-item>
    <el-form-item label="位置" :label-width="formLabelWidth">
      <el-input v-model="form.alias"></el-input>
    </el-form-item>
    <el-form-item label="评论数量" :label-width="formLabelWidth">
      <el-input v-model="form.reviewCount" type="number"></el-input>
    </el-form-item>
    <el-form-item label="评分" :label-width="formLabelWidth">
      <el-input v-model="form.rating" type="number"></el-input>
    </el-form-item>
    <el-form-item label="口味" :label-width="formLabelWidth">
      <el-input v-model="form.rating" type="number"></el-input>
    </el-form-item>
  </el-form>
</el-dialog>
```

PART FOUR

后端搭建与传输

推荐函数封装

- 定义名为 GET_FINAL_RECOMMENDATIONS 的函数，接收从前端传入的三个参数：USERID、USER_MEAL_SCENE 和 USER_FOCUS_POINT，分别代表用户ID、用户的用餐场景标签和用户的关注点，通过协同过滤和基于内容的推荐算法，返回个性化的店铺推荐列表。

```
def get_final_recommendations(UserID, user_meal_scene, user_focus_point):  
    # 加载并准备数据  
    file_path = 'myflask\\app\\推荐算法\\MealRating.csv'  
    df = pd.read_csv(file_path)  
    df['Rating'] = pd.to_numeric(df['Rating'], errors='coerce')  
    data = df[['UserID', 'MealID', 'Rating']].dropna()  
  
    # 创建 Surprise 读取器和数据集  
    reader = Reader(rating_scale=(1, 5))  
    dataset = Dataset.load_from_df(data,  
        # 合并推荐结果并去重，选取最终推荐的5个店铺  
        final_recommendations = pd.concat(recommended_shops).drop_duplicates(subset=[  
            result1 = final_recommendations[['店铺名称', '菜品相似度得分', '场景相似度得分']]  
            result2 = pd.DataFrame(result1)  
            result_list = result2.to_dict(orient='records')  
            print(result_list)  
  
    # 评估不同算法  
    def evaluate_algorithms(data):  
        algorithms = {  
            'KNNBasic': KNNBasic(),  
            'KNNWithMeans': KNNWithMeans(),  
            'KNNWithZScore': KNNWithZScore(),  
            'SVD': SVD(),  
            'SVDpp': SVDpp(),  
            'NMF': NMF()  
        }  
        results = {}  
        for name, algo in algorithms.items():  
            results[name] = algo.fit(data)
```

```
<template>  
  <div class="recommendation-container">  
    <div class="form-container">  
      <h2>推荐系统输入</h2>  
      <el-form :model="form" label-width="100px">  
        <el-form-item label="用户ID">  
          <el-input v-model="form.UserID" placeholder="请输入用户ID"></el-input>  
        </el-form-item>  
        <el-form-item label="场景标签">  
          <el-select v-model="form.user_meal_scene" placeholder="请选择场景标签">  
            <el-option label="朋友聚会" value="朋友聚会"></el-option>  
            <el-option label="家庭聚餐" value="家庭聚餐"></el-option>  
            <el-option label="情侣约会" value="情侣约会"></el-option>  
            <el-option label="商务宴请" value="商务宴请"></el-option>  
            <el-option label="公司团建" value="公司团建"></el-option>  
            <el-option label="节日聚餐" value="节日聚餐"></el-option>  
            <el-option label="学生聚餐" value="学生聚餐"></el-option>  
            <el-option label="主题派对" value="主题派对"></el-option>  
            <el-option label="亲子溜娃" value="亲子溜娃"></el-option>  
            <el-option label="单人餐" value="单人餐"></el-option>  
            <el-option label="单身餐" value="单身餐"></el-option>  
            <el-option label="外卖" value="外卖"></el-option>  
            <el-option label="工作日" value="工作日"></el-option>  
          </el-select>  
        </el-form-item>  
        <el-form-item label="关注点">  
          <el-select v-model="form.user_focus_point" placeholder="请选择关注点">  
            <el-option label="口味" value="口味"></el-option>  
            <el-option label="环境" value="环境"></el-option>  
            <el-option label="服务" value="服务"></el-option>  
          </el-select>  
        </el-form-item>  
      </el-form>  
    </div>  
  </div>
```

前端页面设计

PART FOUR

后端搭建与传输

后端搭建

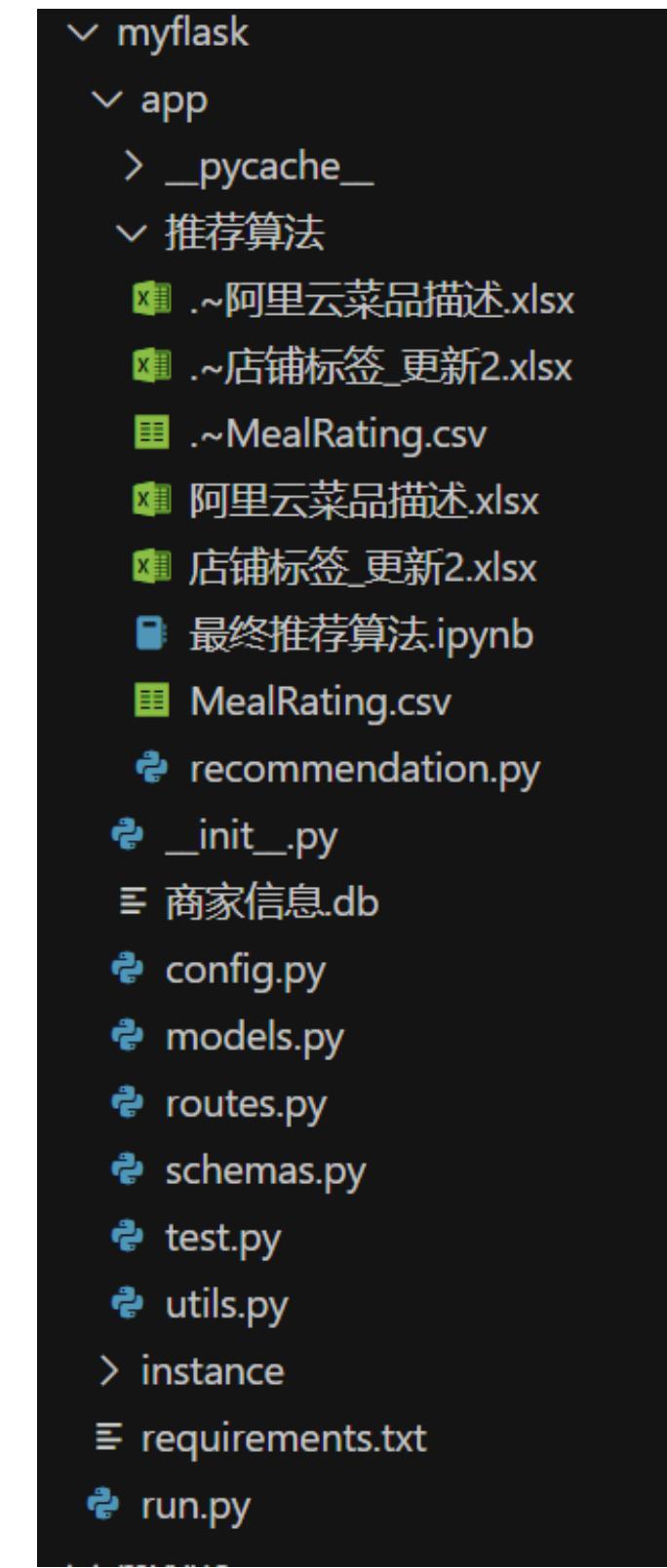
- 搭建FLASK 服务器处理前端发送的各种请求，根据请求的类型和参数执行相应的业务逻辑。主要通过路由机制，将不同的 URL 请求映射到相应的处理函数上，确保系统的不同功能模块可以正确响应前端的需求。

```
# 评论TOP10店铺
@main.route('/commentsRank', methods=['GET'])
def get_top_shops():
    try:
        top_shops = Shop.query.order_by(Shop.评论数量.desc()).limit(10).all()
        result = shops_schema.dump(top_shops)
        return make_response(data=result)
    except Exception as e:
        return make_response(code=1, message=str(e))

# 商家信息
@main.route('/numberRank', methods=['GET'])
def get_top_Rank():
    try:
        ret = db.session.query(Shop.标签.label('name'),
                               db.func.count(Shop.ROWID).label('value')).group_by(Shop.标签).order_by(db.desc('value')).limit(20).all()
        result = chart_schema.dump(ret)
        return make_response(data=result)
    except Exception as e:
        return make_response(code=1, message=str(e))

# 推荐算法
@main.route('/scoreradar', methods=['GET'])
def shop_radar_chart():
```

routes.py



PART FOUR

后端搭建与传输

前端跨域传输

- 通过AXIOS库前端可以向后端发送异步请求，后端的 FLASK 服务器会根据前端的请求，调用相应的处理函数，并将处理结果以 JSON 格式返回给前端。前端接收到数据后根据数据的类型和用途进行相应的处理。

```
import axios from 'axios'

const service = axios.create({
  baseURL: '/api',
  timeout: 20000,
  headers: {
    'Content-Type': 'application/json'
  }
}

export default service
```



```
import request from '@/api/request'

// 评论前十的景点
export function getCommentsRank() {
  return request({
    url: '/commentsRank',
    method: 'get'
  })
}

// 按照数量排名
export function getNumberRank() {
  return request({
    url: '/numberRank',
    method: 'get'
  })
}

// 按照数量排名
export function getScoresRadar() {
  return request({
    url: '/scoreradar',
    method: 'get'
  })
}
```

后端搭建与传输

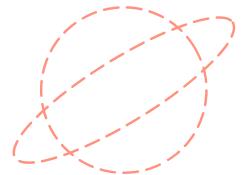
传输呈现

```
{  
    "平均值": {  
        "口味": 3.807466666666649,  
        "总评分": 3.861705006765884,  
        "服务": 3.8075999999999866,  
        "环境": 3.79706666666665  
    },  
    "最大值": {  
        "口味": 4.9,  
        "总评分": 4.8,  
        "服务": 4.9,  
        "环境": 4.8  
    },  
    "最小值": {  
        "口味": 0,  
        "总评分": 3.2,  
        "服务": 0,  
        "环境": 0  
    }  
}
```

```
{  
    "code": 0,  
    "data": [  
        {"ROWID": "1",  
        "人均消费": 108,  
        "位置": "高碑店",  
        "口味": 4.7,  
        "商家ID": "G6M6693HWSWDU9j4",  
        "商家名称": "南门涮肉(朝阳路店)",  
        "商家链接": "https://www.dianping.com/shop/G6M6693HWSWDU9j4",  
        "服务": 4.4,  
        "标签": "老北京火锅",  
        "环境": 4.4,  
        "评分": 4.6,  
        "评论数量": 8159  
},  
        {"ROWID": "8",  
        "人均消费": 134,  
        "位置": "高碑店",  
        "口味": 4.8,  
        "商家ID": "HaYDUGdCf0edDZXh",  
        "商家名称": "重庆棒棒军·手作火锅(北京店)",  
        "商家链接": "https://www.dianping.com/shop/HaYDUGdCf0edDZXh",  
        "服务": 4.8,  
        "标签": "重庆火锅",  
        "环境": 4.8,  
        "评分": 4.8,  
        "评论数量": 7829  
},  
        {"ROWID": "93",  
        "人均消费": 112,  
        "位置": "定福庄",  
        "口味": 4.3,  
        "商家ID": "1lqxQloYBE2zyohw",  
        "商家名称": "山城辣妹子重庆火锅(定福庄店)",  
        "商家链接": "https://www.dianping.com/shop/1lqxQloYBE2zyohw",  
        "服务": 4.2,  
        "标签": "重庆火锅",  
        "环境": 4.3,  
        "评分": 4.3,  
        "评论数量": 5294  
},  
        {"ROWID": "68",  
        "人均消费": 78,  
        "位置": "双桥",  
        "口味": 4,  
        "商家ID": "15NK4UTZliGdb5Ke",  
        "商家名称": "花小福·烤肉无限畅吃(双桥万达店)",  
        "商家链接": "https://www.dianping.com/shop/15NK4UTZliGdb5Ke",  
        "服务": 4,  
        "标签": "烤肉自助",  
        "环境": 4,  
        "评分": 4,  
        "评论数量": 4993  
},  
    ]  
}
```

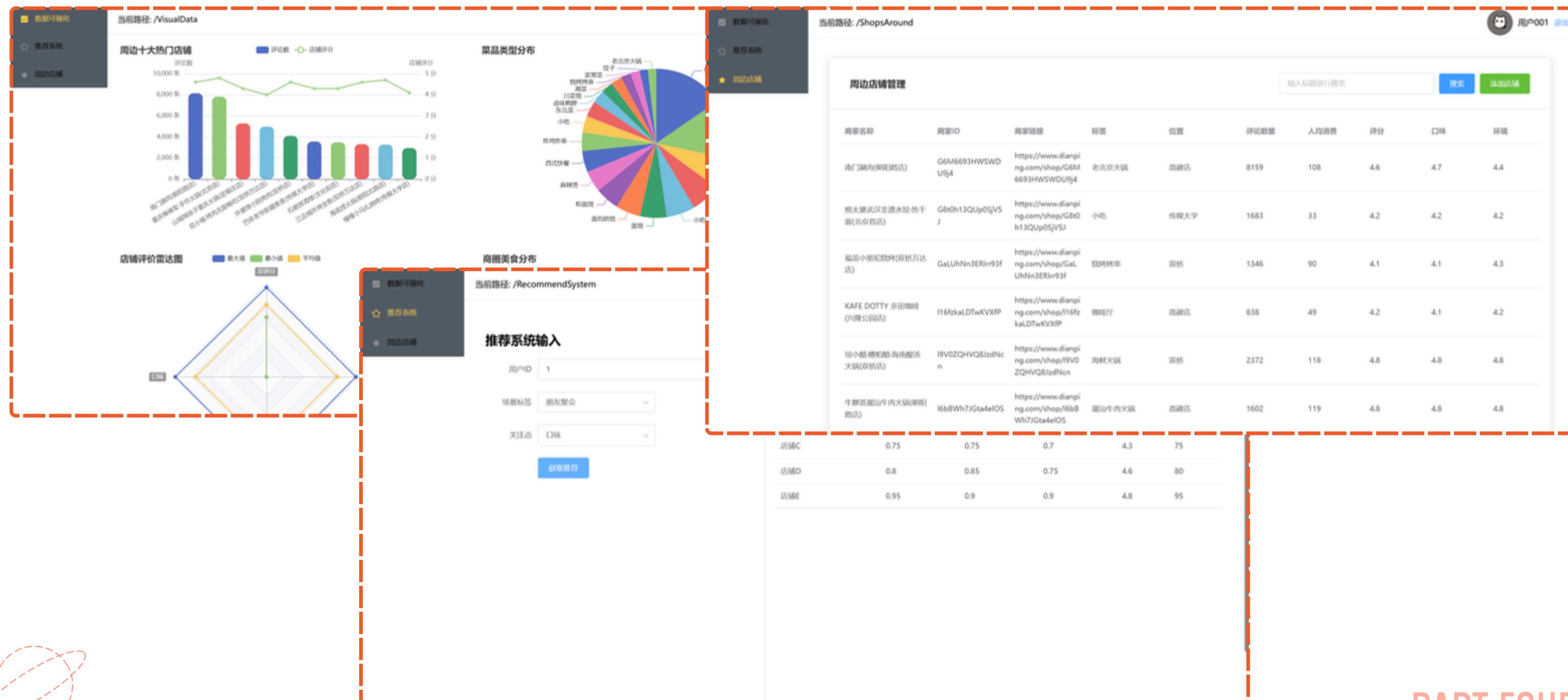
```
{  
    "code": 0,  
    "data": [  
        {"name": "快餐简餐",  
        "value": 77  
},  
        {"name": "咖啡厅",  
        "value": 64  
},  
        {"name": "茶饮果汁",  
        "value": 34  
},  
        {"name": "烤串",  
        "value": 32  
},  
        {"name": "小吃快餐",  
        "value": 31  
},  
        {"name": "面馆",  
        "value": 29  
},  
        {"name": "面包烘焙",  
        "value": 27  
},  
        {"name": "粉面馆",  
        "value": 26  
},  
        {"name": "麻辣烫",  
        "value": 23  
},  
        {"name": "西式快餐",  
        "value": 23  
},  
        {"name": "炸鸡炸串",  
        "value": 20  
},  
    ]  
}
```

可视化展示



PART FOUR

前端系统展示



推荐系统补充展示

```
test_user = df['UserID'].iloc[0]
```

```
user_meal_scene = "情侣约会" # 例如: 家庭聚餐、情侣约会等  
user_focus_point = "环境" # 关注点(口味/服务/环境)
```

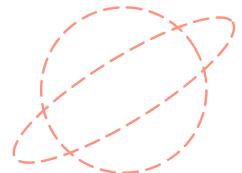
最终推荐度最高的5个店铺及其各项得分:						
	店铺名称	菜品相似度得分	场景相似度得分	关注点相似度得分	总体评分	推荐分
471	树小面(万达广场店)	9.211121	1	1 0.0	4.584448	
18	草原黑森林碳烤羊腿·定福庄扛把子(朝阳北路店)	8.374645	1	1 1.0	4.349858	
277	大书房咖啡(中传国交店)	9.319821	0	1 0.0	4.227928	
530	晓燃地摊烤肉(北花园店)	8.013055	1	1 1.0	4.205222	

```
user_meal_scene = "公司团建" # 例如: 家庭聚餐、情侣约会等  
user_focus_point = "环境" # 关注点(口味/服务/环境)
```

最终推荐度最高的5个店铺及其各项得分:						
	店铺名称	菜品相似度得分	场景相似度得分	关注点相似度得分	总体评分	推荐分
233	肯悦咖啡(远通桥店)	8.720967	1	1 0.0	4.388387	
471	树小面(万达广场店)	9.211121	0	1 0.0	4.184448	
530	晓燃地摊烤肉(北花园店)	8.669795	0	1 1.0	4.067918	
203	Ditto儿咖啡	8.568885	0	1 0.5	3.977554	
42	云滇阁	7.464296	1	1 0.0	3.885718	

```
user_meal_scene = "情侣约会" # 例如: 家庭聚餐、情侣约会等  
user_focus_point = "环境" # 关注点(口味/服务/环境)
```

最终推荐度最高的5个店铺及其各项得分:						
	店铺名称	菜品相似度得分	场景相似度得分	关注点相似度得分	总体评分	推荐分
471	树小面(万达广场店)	9.211121	1	1 0.0	4.584448	
273	WOOD ONE TAIL FISH木尾鱼咖啡(高碑店店)	8.980811	1	1 0.5	4.542324	
530	晓燃地摊烤肉(北花园店)	8.669795	1	1 1.0	4.467918	
203	Ditto儿咖啡	8.568885	1	1 0.5	4.377554	
667	泉州面线糊(财满街店)	8.106827	1	1 0.0	4.142731	



感谢您的观看

