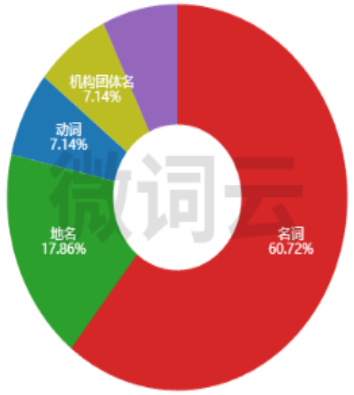
用户数据采集与关联分析

（结课作业）

信管2302 钟京序 202321054056

第一讲 课程导言与分词

字数	109个字
大小	297.00 Bytes
有效条数	1条
总词数	28个
特征词数	26个
平均句长	28.00个单词
词密度	67.86%
下载	源文本.txt 打标词表.xlsx 特征词表(词频表).xlsx 下载任务文件



单词分布情况

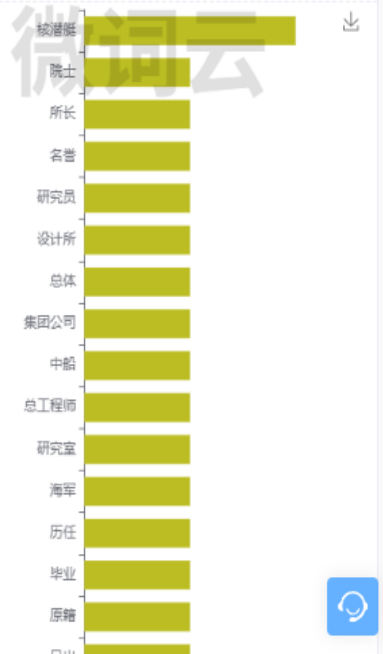
相关词云图 美化词云图

特征词表 创建词云图

单词	全部	数量	条数	tf-idf	操作
日出	名词	1	1	-0.0107510713	查看
生于	动词	1	1	-0.0107510713	查看
汕尾市	地名	1	1	-0.0107510713	查看
原籍	名词	1	1	-0.0107510713	查看
揭阳市	地名	1	1	-0.0107510713	查看
毕业	名词	1	1	-0.0107510713	查看
上海交通大学	机构团体名	1	1	-0.0107510713	查看
历任	名词	1	1	-0.0107510713	查看
北京	地名	1	1	-0.0107510713	查看
海军	名词	1	1	-0.0107510713	查看
研究室	名词	1	1	-0.0107510713	查看
总工程师	名词	1	1	-0.0107510713	查看
中船	名词	1	1	-0.0107510713	查看
重工	名动词	1	1	-0.0107510713	查看
集团公司	名词	1	1	-0.0107510713	查看
总体	名词	1	1	-0.0107510713	查看
研究	名动词	1	1	-0.0107510713	查看
设计所	名词	1	1	-0.0107510713	查看
研究员	名词	1	1	-0.0107510713	查看
名誉	名词	1	1	-0.0107510713	查看
所长	名词	1	1	-0.0107510713	查看

词性柱状图 创建词云图

名词 更改词性



```
dest_error ??
```

In [29]:
```
"Hello World.Hello'钟京序'"
```

Out[29]:
```
"Hello World.Hello'钟京序'"
```

In [5]:
```
import jieba
seg_list1 = jieba.cut("曾经有一份真诚的爱情摆在我的面前，我没有珍惜，等到失去的时候才追悔莫
print('$'.join(seg_list1))
```

曾经$有$一份$真诚$的$爱情$摆在$我$的$面前$，$我$没有$珍惜$，$等到$失去$的$时候$才$追悔莫及$，$人世间$最$痛苦$的$事情$莫过于此$。$如果$上天$能够$给$我$一个$重新$来$过$的$机会$，$我会$对$那个$女孩子$说$三个$字$：$“$我爱你$”$。$如果$非要$给$这份$爱$加上$一个$期限$，$我$希望$是$，$一万年

In [16]:
```
seg_list2 = jieba.cut("LSTM（Long Short-Term Memory）是长短期记忆网络，是一种时间递归神经网
print('@'.join(seg_list2))
```

LSTM@（@Long@ @Short@-@Term@ @Memory@）@是@长短期记忆网络@，@是@一种@时间递归神经网络@，@适合@于@处理@和@预测@时间@序列@中@间隔@和@延迟@相对@较长@的@重要@事件@。

In [17]:
```
.load_userdict("D:\数据采集与关联分析\用户数据\第1讲 感知世界：课程导言与分词\第1讲 感知世界
ist_dict = jieba.cut("LSTM（Long Short-Term Memory）是长短期记忆网络，是一种时间递归神经网络
('/'.join(seg_list_dict))
```

LSTM/（/Long/ /Short/-/Term/ /Memory/）/是/长短期记忆网络/，/是/一种/时间递归神经网络/，/适合/于/处理/和/预测/时间/序列/中/间隔/和/延迟/相对/较长/的/重要/事件/。

In [15]:
```
与分词\第1讲 感知世界：课程导言与分词\stop_words.txt", 'r', encoding='utf-8').readlines()]
人世间最痛苦的事情莫过于此。如果上天能够给我一个重新来过的机会，我会对那个女孩子说三个字：
```

曾经/有/一份/真诚/爱情/摆在/我/面前/我/没有/珍惜/等到/失去/时候/才/追悔莫及/人世间/最/痛苦/事情/莫过于此/如果/上天/能够/给/我/一个/重新/来/过/机会/我会/对/那个/女孩子/说/三个/字/：/“/我爱你/”/如果/非要/给/这份/爱/加上/一个/期限/我/希望/一万年/

In [19]:
```
import jieba
stopwords = [
    line.strip()
    for line in open(
        r"D:\数据采集与关联分析\用户数据\第1讲 感知世界：课程导言与分词\第1讲 感知世界：课
        'r',
        encoding='utf-8'
```

艇/总体/研究/设计所/研究员/、/名誉/所长/。/1994/年/当选/为/中国工程院院士/。

In [22]:
```
seg_list_huang = jieba.cut('黄旭华，1926年3月12日出生于广东省汕尾市，原籍广东省揭阳市。194
final = ''
for seg in seg_list_huang:
    if seg not in stopwords:
        final+= seg+'/'
print(final)
```

黄旭华/1926/年/3/月/12/日出/生于/广东省/汕尾市/原籍/广东省/揭阳市/1949/年/毕业/于/上海交通大学/历任/北京/海军/核潜艇/研究室/副/总工程师/中船重工集团公司/核潜艇/总体/研究/设计所/研究员/名誉/所长/1994/年/当选/为/中国工程院院士/

In [23]:
```
import jieba
from collections import Counter
# 文本
text = """
落实“企业管理年”主题，加强QEHS三体系建设，
通过自动化、数字化、智能化升级改造加快新一代信息技术与企业生产经营融合，
打造精益制造能力，提升精细化管理水平，助力公司从“制造”升级为“智造”，
从而提高经营效率和效益:通过集团一体化智数管理平台建设与运营，
丰富企业供应链管理、生产工艺控制等管理工具，不断增强生产经营过程数据获取与分析能力，
强化全过程一体化管理，提高自动化、数字化、智能化的供应链管理能力，
为体系安全稳定运行与管理水平提升保驾护航，致力打造安全智能化工厂；
打造助剂互联网技术合作和商务合作平台，构建具有国际竞争力的供应链体系。
"""
# 1. 分词处理
words = jieba.lcut(text)
# 2. 定义要统计的特殊词汇
target_words = ['数字化', '智能化', '安全']
# 统计词频
word_counts = Counter(words)
# 输出特定词汇的词频统计结果
print("特定词汇词频统计结果：")
for word in target_words:
    print(f"'{word}': {word_counts[word]}次")
# 输出所有词汇的词频（按频率降序）
print("\n所有词汇词频统计（前20个）：")
for word, count in word_counts.most_common(20):
    print(f"'{word}': {count}次")
```

特定词汇词频统计结果：
'数字化': 2次
'智能化': 3次
'安全': 2次

所有词汇词频统计（前20个）：
'，': 13次

In [19]:
```
import jieba
stopwords = [
    line.strip()
    for line in open(
        r"D:\数据采集与关联分析\用户数据\第1讲 感知世界：课程导言与分词\第1讲 感知世界：课
        'r',
        encoding='utf-8'
    ).readlines()
]
text = """黄旭华，1926年3月12日出生于广东省汕尾市，原籍广东省揭阳市。1949年毕业于上海交通大

seg_list_stopw = jieba.cut(text)
final = ''
for seg in seg_list_stopw:
    if seg not in stopwords and seg.strip() != '':
        final += seg + '/'
print(final)
```

黄旭华/1926/年/3/月/12/日出/生于/广东省/汕尾市/原籍/广东省/揭阳市/1949/年/毕业/于/上海交通大学/历任/北京/海军/核潜艇/研究室/副/总工程师/中船重工集团公司/核潜艇/总体/研究/设计所/研究员/名誉/所长/1994/年/当选/为/中国工程院院士/

In [20]:
```
import jieba
seg_list_huang = jieba.cut('黄旭华，1926年3月12日出生于广东省汕尾市，原籍广东省揭阳市。194
print('/'.join(seg_list_huang))
```

黄旭华/，/1926/年/3/月/12/日出/生于/广东省/汕尾市/，/原籍/广东省/揭阳市/。/1949/年/毕业/于/上海交通大学/。/历任/北京/海军/核潜艇/研究室/副/总工程师/、/中船重工集团公司/核潜艇/总体/研究/设计所/研究员/、/名誉/所长/。/1994/年/当选/为/中国工程院院士/。

In [21]:
```
jieba.load_userdict('D:\数据采集与关联分析\用户数据\第1讲 感知世界：课程导言与分词\第1讲 感
seg_list_huang = jieba.cut('黄旭华，1926年3月12日出生于广东省汕尾市，原籍广东省揭阳市。194
print('/'.join(seg_list_huang))
```

黄旭华/，/1926/年/3/月/12/日出/生于/广东省/汕尾市/，/原籍/广东省/揭阳市/。/1949/年/毕业/于/上海交通大学/。/历任/北京/海军/核潜艇/研究室/副/总工程师/、/中船重工集团公司/核潜艇/总体/研究/设计所/研究员/、/名誉/所长/。/1994/年/当选/为/中国工程院院士/。

In [22]:
```
seg_list_huang = jieba.cut('黄旭华，1926年3月12日出生于广东省汕尾市，原籍广东省揭阳市。194
final = ''
for seg in seg_list_huang:
    if seg not in stopwords:
        final+= seg+'/'
print(final)
```

In [28]:
```
import requests
import json

# 定义DeepSeek API的URL和headers
DEEPSEEK_API_URL = "https://api.deepseek.com/v1/chat/completions"
API_KEY = "sk-73d6a7dc401c433e944d4376b81ad588"  #直接复制过来
# 准备prompt和论文文本
paper_text = """
随着肿瘤免疫微环境（Tumor Immune Microenvironment, TIME）研究的深入，
T细胞耗竭（T cell exhaustion）被认为是限制免疫治疗效果的关键机制之一。
本研究基于免疫编辑理论，提出了一种基于单细胞RNA测序（scRNA-seq）的T细胞状态动态识别方法。
具体而言，我们使用Seurat与Monocle3等生物信息学工具对50例非小细胞肺癌患者的肿瘤样本进行细胞
结合pseudotime推断T细胞从激活到耗竭的转化过程。此外，借助CellChat软件构建细胞间通讯网络，
进一步识别可能诱导T细胞耗竭的免疫抑制信号通路，如PD-1/PD-L1和TGF-β路径。研究结果揭示了T细
"""

prompt = f"""
请从以下科技论文文本中提取包含理论、方法、工具的实体或专业术语，以json字典的格式输出：

{paper_text}
"""
# 准备请求数据
data = {
    "model": "deepseek-chat",
    "messages": [
        {"role": "user", "content": prompt}
    ],
    "temperature": 0.3
}

headers = {
    "Content-Type": "application/json",
    "Authorization": f"Bearer {API_KEY}"
}

# 发送请求
response = requests.post(DEEPSEEK_API_URL, headers=headers, data=json.dumps(data))
# 处理响应
if response.status_code == 200:
    result = response.json()
    try:
        entities = result['choices'][0]['message']['content']
        print("提取到的实体和专业术语：")
        print(entities)
    except KeyError:
        print("无法解析API响应，原始响应：")
        print(result)
else:
    print(f"请求失败，状态码：{response.status_code}")
```

阅读总结

基于关键词的学术文本聚类集成研究

《基于关键词的学术文本聚类集成研究》一文聚焦学术文本自动类别划分的现实需求，针对全球科技文献海量增长且人工分类耗时耗力、单一聚类方法性能有限的问题，结合聚类集成思想开展系统性研究。文章首先梳理了学术文本类别划分（含以期刊和文本为单位两类方法，后者逐渐成为主流但存在单一聚类局限）、聚类集成（需经基聚类生成、结果转化与合并步骤，在分类任务中应用广泛但聚类领域研究不足）及关键词抽取（分监督与无监督两类，无监督单文本抽取更适配海量学术文本场景）的相关研究现状，明确核心研究问题：聚类集成能否提升基于关键词的学术文本聚类性能，关键词抽取方法与关键词个数是否对聚类集成性能产生影响。研究中采用 ACM 计算机学科分类体系（CCS2012）下的 Rossi 数据集，该数据集含 40 个领域 3506 篇论文及 8 个测试子集，类别划分精准、数据分布均衡，为实验提供可靠基础。关键词抽取选用 TF-ISF、CSI、ECC、TextRank 四种经典无监督单文本方法，抽取 5 至 60 个不同数量的关键词；文本表示采用向量空间模型，相似度计算使用余弦相似度；聚类集成构建两种方法，分别是以 K-means 为基础、基于数据采样的 ECKM，以及以增量聚类为基础、基于算法参数调整的 ECIC，同时以 K-means 和增量聚类作为基准方法。实验以准确率、召回率和 F1 值为评估指标，通过多组对比实验得出核心结论：聚类集成方法显著提升了学术文本聚类性能，且在不同关键词抽取方法和关键词个数下更具稳定性，其中 ECKM 性能稳定性突出，ECIC 在关键词数量较少时优势明显；关键词抽取方法对聚类集成性能影响显著，TextRank 方法表现最佳，CSI 方法效果较差；随着关键词个数的增加，学术文本聚类性能整体呈上升趋势，关键词数量较多时能更全面地表征文本主旨。文章最后提出实践建议，即在学术文本类别划分中应优先采用聚类集成方法，慎重选择关键词抽取方法并尽可能抽取较多关键词，同时指出未来可在中文学术文本数据集上验证模型，或结合关键词与引文信息优化文本表示方式，进一步提升聚类性能。该研究为学术文本自动分类提供了高效可行的解决方案，对提升学术文献检索效率、促进学科结构分析具有重要参考价值。

第二讲 词频统计

CNKI数据库统计分析2014-2024年（近10年），“信息资源管理”主题变化趋势。

阶段	年度发文量	核心特征	驱动因素
2014-2017	年均约 2800 篇	平稳增长，数字资源建设为主线	大数据战略、数字图书馆推广工程
2018-2021	年均约 3400 篇	峰值出现（2020 年约 3800 篇），疫情催化应急信息服务	数据治理政策落地、突发公共卫生事件
2022-2024	年均约 3100 篇	小幅回落但质量提升，跨学科融合加速	通用人工智能、数据要素市场化改革

CNKI数据库统计分析2014-2024年（近10年），“信息资源管理”主题变化趋势

分阶段主题演化与热点迁移

1. 2014-2017：数字资源建设与传统领域深化

核心热点：数字图书馆建设、档案数字化转型、开放数据与政府信息公开、竞争情报与企业信息管理。

特征：以“资源数字化”为核心，图书馆、档案学、情报学三领域并行发展，技术应用集中于数据库建设、元数据标引等基础层面，跨学科研究占比约 25%。

典型主题词：数字档案馆、开放存取、用户信息行为、信息组织。

2. 2018-2021：数据治理与应急服务崛起

核心热点：数据治理体系构建、网络舆情监测、智慧图书馆、疫情信息服务、数字人文初步探索。

特征：从“资源管理”转向“数据治理”，政策驱动（如《数据安全法》立法讨论）与技术融合（区块链、云计算）加深，跨学科占比升至 30%+，应急信息服务在 2020 年达峰值。

典型主题词：数据治理、网络谣言舆情、智慧服务、数字人文。

3. 2022-2024：数智融合与要素价值化

核心热点：通用人工智能与信息资源管理、数据要素价值化、档案数据治理、数智素养、数字文化遗产智能计算。

特征：“数智化”全面渗透，学科边界模糊，与计算机科学、法学、经济学交叉显著，颠覆性技术（ChatGPT、大模型）推动情报分析、用户服务等方向革新，数据安全与隐私保护成为高频主题。

典型主题词：通用人工智能、数据要素、档案数据资产、数智素养、跨境数据流动。

CNKI数据库统计分析2014-2024年（近10年），“信息资源管理”主题变化趋势

核心主题热度变与归因

主题方向	2014-2017	2018-2021	2022-2024	趋势	归因
图书馆管理与服务	高	中	低	持续下降	传统服务模式向智慧服务转型，基础管理研究减少
档案管理与数字人文	中	中	高	平稳上升	数字人文兴起，档案数据要素价值凸显
数据治理与要素市场	低	中	高	快速上升	数据要素政策落地，市场化需求驱动
情报服务与智库建设	中	中	高	稳步上升	国家智库战略与 AI 赋能情报分析
信息安全与隐私保护	中	高	高	持续高热	数据安全法实施，跨境数据流动监管强化
应急与公共卫生信息	低	极高	低	脉冲式爆发后衰退	疫情催化，事件性热点退潮

ppt代码运行

```
import jieba
article = open('D:\数据采集与关联分析\用户数据\第2讲 感知世界：词频统计与分析\sanguo_10.txt','r',encoding = 'utf-8').read()
dele = {'。','！','？','的','“','”',' （',') ',' ','》','《', '}
jieba.add_word('皇叔')
words = list(jieba.cut(article))
```

```
Building prefix dict from the default dictionary ...
Loading model from cache C:\Users\zjx25\AppData\Local\Temp\jieba.cache
Loading model cost 1.502 seconds.
Prefix dict has been built successfully.
```

```
articleDict = {}  # 这是一个字典，准备词-词频的保存

articleSet = set(words)-dele

# 因为采用的是循环的方法，把所有的词都循环一遍，长度大于1，所以速度很慢！
for w in articleSet:
    if len(w)>1:
        articleDict[w] = words.count(w)

articlelist = sorted(articleDict.items(),key = lambda x:x[1], reverse = True) # 对词典中的词排序

# 输出词频的前N个
for i in range(20):
    print(articlelist[i])
```

```
('董卓', 97)
('吕布', 60)
('曹操', 59)
('袁绍', 57)
('天下', 53)
('玄德', 48)
('貂蝉', 37)
('太守', 36)
('朝廷', 32)
('孫堅', 31)
('不可', 31)
('次日', 26)
('李儒', 25)
('商議', 25)
('引兵', 25)
('天子', 24)
('左右', 23)
('玄德曰', 22)
('太師', 22)
('大喜', 21)
```

[24]:
```
f_name = open(r"D:\数据采集与关联分析\用户数据\第2讲 感知世界：词频统计与分析\name.txt",encoding = 'GB18030')
data_name = f_name.read()
data_name[:50]
print(data_name[:50])
```

諸葛亮|關羽|劉備|曹操|孫權|關羽|張飛|呂布|周瑜|趙雲|龐統|司馬懿|黃忠|馬超

[25]:
```
f_name.close()

# 将文本转化为列表
names = data_name.split('|')  # split一下names就是列表
print(names)
names
```

['諸葛亮', '關羽', '劉備', '曹操', '孫權', '關羽', '張飛', '呂布', '周瑜', '趙雲', '龐統', '司馬懿', '黃忠', '馬超']

[25]:
```
['諸葛亮',
 '關羽',
 '劉備',
 '曹操',
 '孫權',
 '關羽',
 '張飛',
 '呂布',
 '周瑜',
 '趙雲',
 '龐統',
 '司馬懿',
 '黃忠',
 '馬超']
```

[26]:
```
# 学习一种新的数据结构，字典
name_dict = {}
f_txt = open(r"D:\数据采集与关联分析\用户数据\第2讲 感知世界：词频统计与分析\sanguo.txt", encoding = 'ANSI')
data_txt = f_txt.read()
f_txt.close()
print(data_txt[:100])
```

《三国演义》（全）

（明）羅貫中著

第一回
宴桃園豪傑三結義斬黃巾英雄首立功
話說天下大勢，分久必合，合久必分：周末七國分爭，並
入于秦；及秦滅之後，楚、漢分爭，又並入於漢；漢朝自高祖
斬白

[27]:
```
f_weapon = open(r"D:\数据采集与关联分析\用户数据\第2讲 感知世界：词频统计与分析\weapon.txt", encoding = 'utf-8')
data_weapon = f_weapon.read()
print(data_weapon[:20])
```

青龍偃月刀

丈八點鋼矛

鐵脊蛇矛

词频统计

[29]:
```
import jieba
article = open("D:\数据采集与关联分析\用户数据\第2讲 感知世界：词频统计与分析\科学家博物
dele = {'。','！','？','的','“','”','（','）',' ','》','《',', '}
jieba.add_word('国立交通大学')
words = list(jieba.cut(article))
words
articleDict = {}
articleSet = set(words)-dele
articleSet
```

[29]:
```
{'\n',
 '09',
 '1020',
 '1924',
 '1949',
 '1958',
 '1978',
 '1982',
 '1985',
 '1986',
 '1989',
 '1996',
 '2',
 '2013',
 '24',
 '29',
 '648',
 '7',
 '719',
 '8',
 '、',
 '—',
 '一万年',
 '一个',
 '一九所',
 '一代',
 '一名',
 '一套',
 '一年多来',
 '一心',
 '一支',
 '一是',
 '一步',
 '一生',
 '一种',
 '一系列',
 '一部分',
 '一项',
```

[34]:
```
import jieba
article = open("D:\数据采集与关联分析\用户数据\第2讲 感知世界：词频统计与分析\科学家博物
dele = {'。','！','？','的','“','”','（','）',' ','》','《',', '}
jieba.add_word('国立交通大学')
words = list(jieba.cut(article))
words
articleDict = {}
articleSet = set(words)-dele
for w in articleSet:
    if len(w)>1:
        articleDict[w] = words.count(w)
articlelist = sorted(articleDict.items(),key = lambda x:x[1], reverse = True)
for i in range(100):
    print(articlelist[i])
```

```
('黄旭华', 53)
('核潜艇', 32)
('采集', 29)
('学术', 22)
('资料', 21)
('工作', 17)
('成长', 15)
('小组', 14)
('进行', 13)
('专业', 13)
('院士', 13)
('技术', 12)
('研制', 12)
('我国', 12)
('工程', 11)
('访谈', 10)
('第一代', 8)
('主要', 8)
('介绍', 8)
('科学', 8)
('传记', 7)
('及其', 7)
('人生', 7)
('历史', 7)
('思想', 7)
('一生', 6)
('成就', 6)
('研究', 6)
('设计', 6)
```

词频统计

```
terms = ['黄旭华','核潜艇','国立交通大学']
print(terms)
terms
terms_dict = {}
f_txt = open(r"D:\数据采集与关联分析\用户数据\第2讲 感知世界：词频统计与分析\科学家博物
data_txt = f_txt.read()
f_txt.close()
print(data_txt[:1000])
```

```
['黄旭华', '核潜艇', '国立交通大学']
```

在核潜艇领域，我国已形成一套完整的研究、设计、试验、制造、测试的核潜艇产业体系，而且装备了一支具有极高战略威慑力的、成梯次配备的、已近实现战备巡逻的核潜艇部队。回顾我国核潜艇的发展历程，人们自然会想起以黄旭华为代表的五位两弹院士及无数第一代核潜艇研制人员的带头苦干、呕心沥血、无私奉献，正是他们所铸就的国之重器使我国彻底摆脱了超级大国的核讹诈，更使我们在民族复兴的道路上迈出了坚实的一步。

而今，由黄旭华院士等人所开创的核潜艇工程以令世人震撼的力量，继续承载着捍卫“中国梦”的伟大重任。

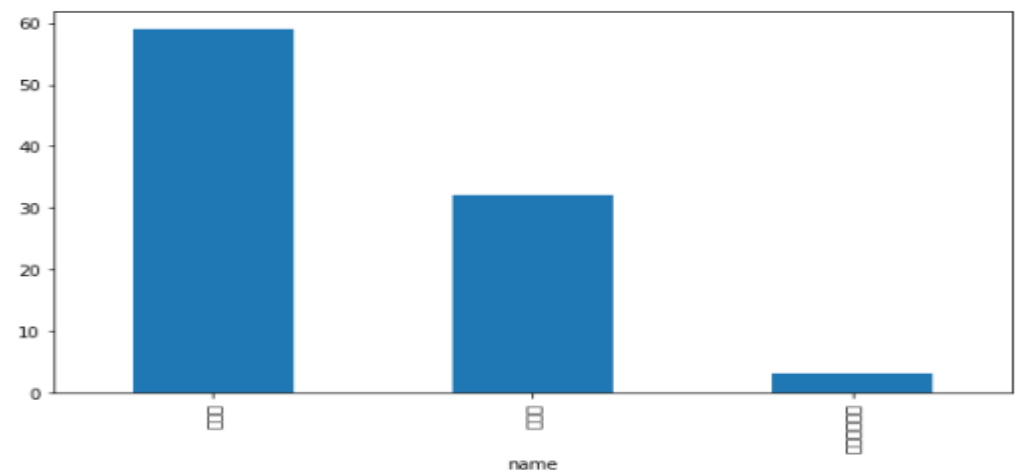
黄旭华是我国著名船舶专家、核潜艇研究设计专家、中国工程院首批院士、中国第一代核动力潜艇研制创始人之一。1924年2月24日，黄旭华出生于广东省汕尾市海丰县田墘镇，原籍广东省揭阳县。1949年，他毕业于国立交通大学造船系船舶制造专业，先后从事过民用船舶和军用舰艇的研究设计工作。1958年，黄旭华开始参与并领导我国第一代核潜艇的研究设计工作，先后出任第一代核潜艇副总设计师、第二任总设计师，历任中国船舶工业总公司及中船重工集团公司第七一九所副总工程师、副所长、所长、党委书记。黄旭华先后于1978年获全国科学大会奖、1982年获国防科工委二等奖，1986年被授予船舶工业总公司劳动模范，1989年被授予全国先进工作者，他参与完成的我国第一代核潜艇研制获1985年国家科学技术进步奖特等奖、导弹核潜艇研制获1996年国家科学技术进步奖特等奖。

黄旭华出生于以医为主、兼理农商之家，正直、勇敢、仁厚、坚韧的父母自小给予了他良好的道德与文化的熏陶。在历经了树基小学、作矶小学、聿怀中学、广益中学、桂林中学、教育部特设大学先修班的坎坷求学历程之后，他以优异的成绩进入了当时著名的国立交通大学，系统学习造船专业理论与技术，以期实现“科学强国”的报国理想。同期在地下党的培养下，历经风雨的洗礼成长为一名坚强的共产党员。

新中国成立后，经过党校系统培训学习，黄旭华在政治思想上逐步成熟。经过苏联军事舰船的转让仿制的锤炼，黄旭华在专业技术上也崭露头角。1958年，黄旭华因为政治素质过硬、专业技术精湛，成为开启“09工程的最初29位专业技术人员之一，从此将自己的一生献给了祖国的核潜艇事业。在核潜艇的研制过程中，黄旭华秉持“自力更生、艰苦奋斗、大力协同、无私奉献”的核潜艇精神，倡导以常规技术系统集成的科学理念，克服重重困

```
for term in terms:
    terms_dict[term]=data_txt.count(term)
terms_dict
def make_chinese_plot_ready():
    from matplotlib import rcParams
    rcParams['font.family'] = 'Heiti TC' # mac笔记本电脑直接替换字体
    #rcParams['font.sans-serif'] = ['FangSong'] # 或者直接使用电脑有的字体 FangSong
    rcParams['axes.unicode_minus'] = False
def draw_dict(mydict, figsize=(8, 5)):
    import pandas as pd
    import matplotlib.pyplot as plt
    make_chinese_plot_ready()
    df = pd.DataFrame(list(mydict.items()), columns=['name', 'times'])
    df.set_index('name')['times'].sort_values(ascending=False).plot(kind='bar', fi
    plt.tight_layout()
%matplotlib inline
draw_dict(terms_dict)
```

```
D:\Anaconda\lib\site-packages\matplotlib\font_manager.py:1331: UserWarning: findf
ont: Font family ['Heiti TC'] not found. Falling back to DejaVu Sans
  (prop.get_family(), self.defaultFamily[fontext]))
```



阅读总结

《Long live the scientists: Tracking the scientific fame of great minds in physics》是中国科学技术大学王国燕与复旦大学唐莉合作完成的研究成果，于 2018 年发表在信息计量学国际顶级期刊《Journal of Informetrics》上，旨在提出一种全新的科学声望（scientific fame）测度方法，突破传统学术评价体系的局限。该研究以 1.26 亿份数字资源（含 Google Books 的 3600 万本全球数字图书和 Google Scholar 的 9000 万篇学术论文）为数据基础，覆盖 57 种语言，通过分析物理学家全名在这些文献中的词频，结合大数据可视化技术，系统描绘了科学巨匠们在人类历史上的声望演变轨迹，重点探究了其学术贡献在学术圈之外的长周期社会影响力。研究指出，传统的论文发表与引用指标仅能反映学术共同体内的短期认可，而网络、社交媒体等补充数据又存在历史短暂、稳定性弱的问题，而书籍作为人类知识与文明的重要载体，长期以来在定量研究评价中未得到充分重视，因此该研究选择以图书和学术论文的词频分析为核心，为科学声望的长期测度提供了更全面、更具持续性的视角。以爱因斯坦和牛顿两位物理学巨匠为典型案例的分析显示，科学家的声望存在“群体内偏好”特征——牛顿在英国出版的书籍中被提及频次始终高于爱因斯坦，而出生于德国、后加入美国国籍的爱因斯坦则在德国和美国的书籍中更受关注；从时间维度看，学术圈自 1948 年起、社会图书自 20 世纪 70 年代后，爱因斯坦的名字提及频次逐渐超过牛顿，共词分析进一步揭示了二者的核心关联领域，爱因斯坦常与相对论（28.1%）和量子论（16.9%）一同被提及，牛顿则多与万有引力（15.2%）和微积分（7.9%）绑定出现。研究还列出了 21 世纪最具影响力的物理学家名单，前五位依次为爱因斯坦、普朗克、牛顿、帕斯卡和伽利略，证实了科学巨匠的贡献能跨越数百年甚至上千年，从具体的学术成果沉淀为科学精神的文化符号，持续激励后世学者；同时，该研究也验证了 Google Books 及其 Ngram Viewer 工具在替代计量学（altmetrics）领域的应用潜力，为更全面地评估学者及其成果的跨领域、长周期影响力提供了创新路径。

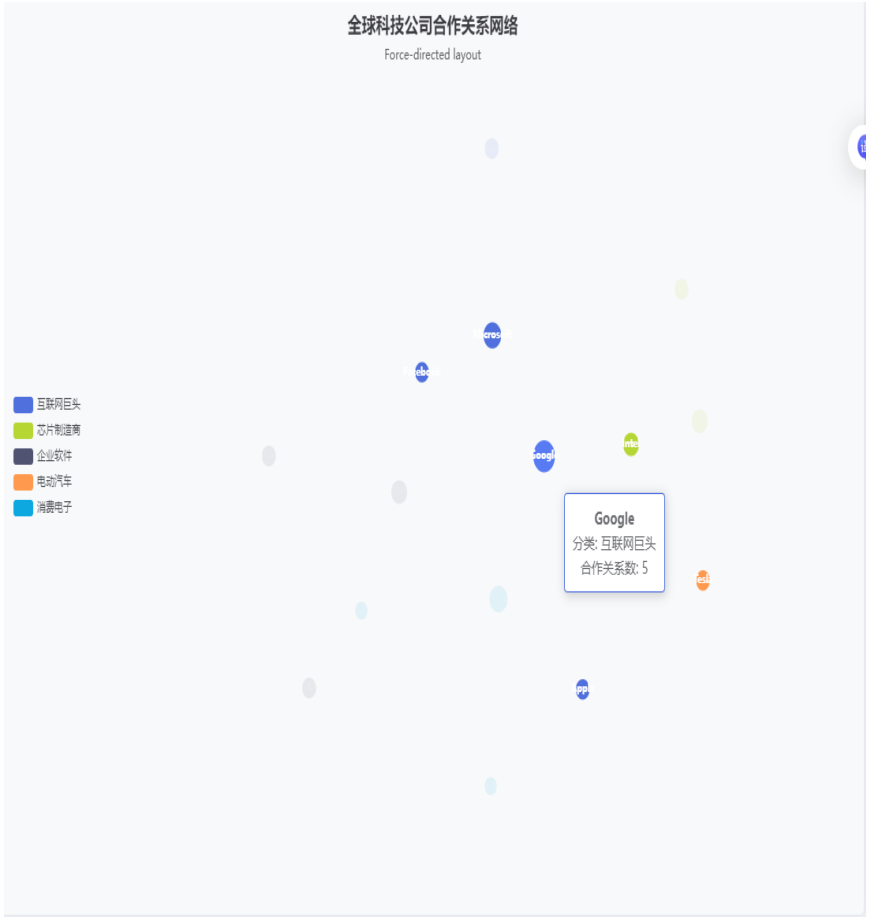
第三讲 词云与可视化

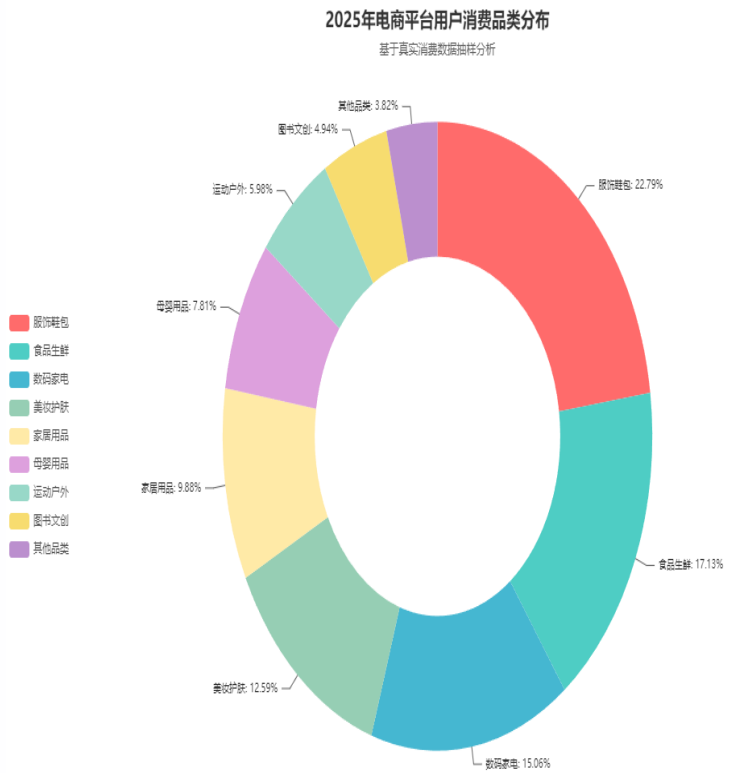
词云图

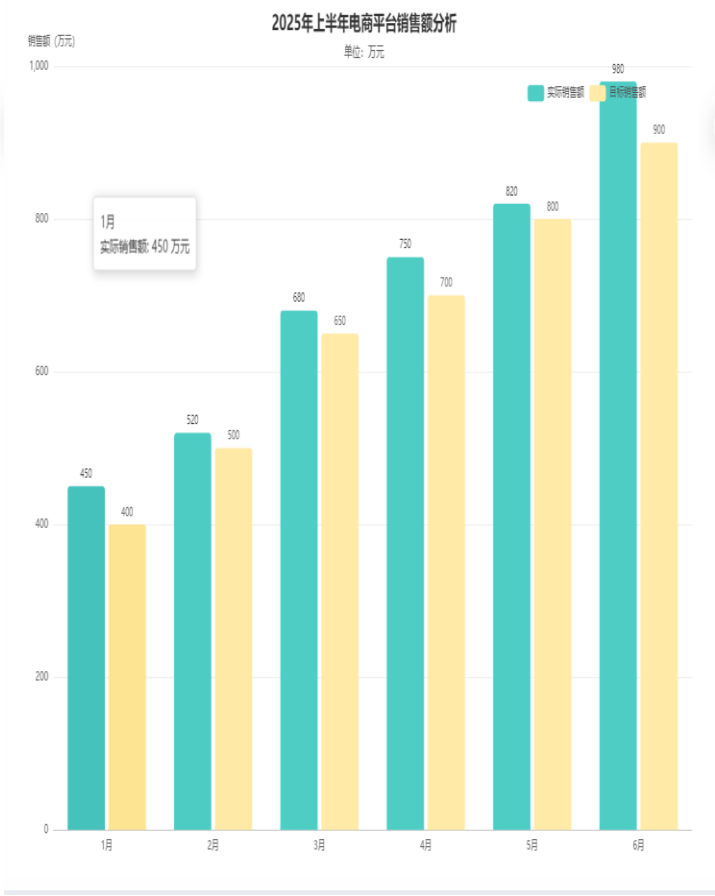
Filter（关键词）	Size（权重）	Color	Angle	Font
草莓	15	Default	Default	Default
没有坏果	14	Default	Default	Default
香甜	12	Default	Default	Default
饱满	11	Default	Default	Default
分量足	8	Default	Default	Default
包装严实	7	Default	Default	Default
个头均匀	7	Default	Default	Default
新鲜	7	Default	Default	Default
汁水多	6	Default	Default	Default
口感好	6	Default	Default	Default
物流快	5	Default	Default	Default
色泽鲜艳	5	Default	Default	Default
无损伤	5	Default	Default	Default
酸甜适中	4	Default	Default	Default
推荐购买	4	Default	Default	Default
果肉细腻	4	Default	Default	Default
性价比高	3	Default	Default	Default
味美	3	Default	Default	Default
品质好	3	Default	Default	Default

这张草莓商品评价的词云图，以“草莓”为核心主题，通过“没有坏果”“新鲜”“无损伤”凸显商品品质的稳定性，“香甜”“汁水多”“口感好”“酸甜适中”等词汇集中体现了口味层面的正面反馈，“饱满”“个头均匀”“色泽鲜艳”则反映了外观与规格的优质表现，而“包装严实”“物流快”补充了服务环节的良好体验，整体词云以高频正面词汇为主，直观展现出该草莓在 2025 年 12 月期间收获的好评维度，清晰传递出“品质佳、口味好、服务优”的推荐印象。

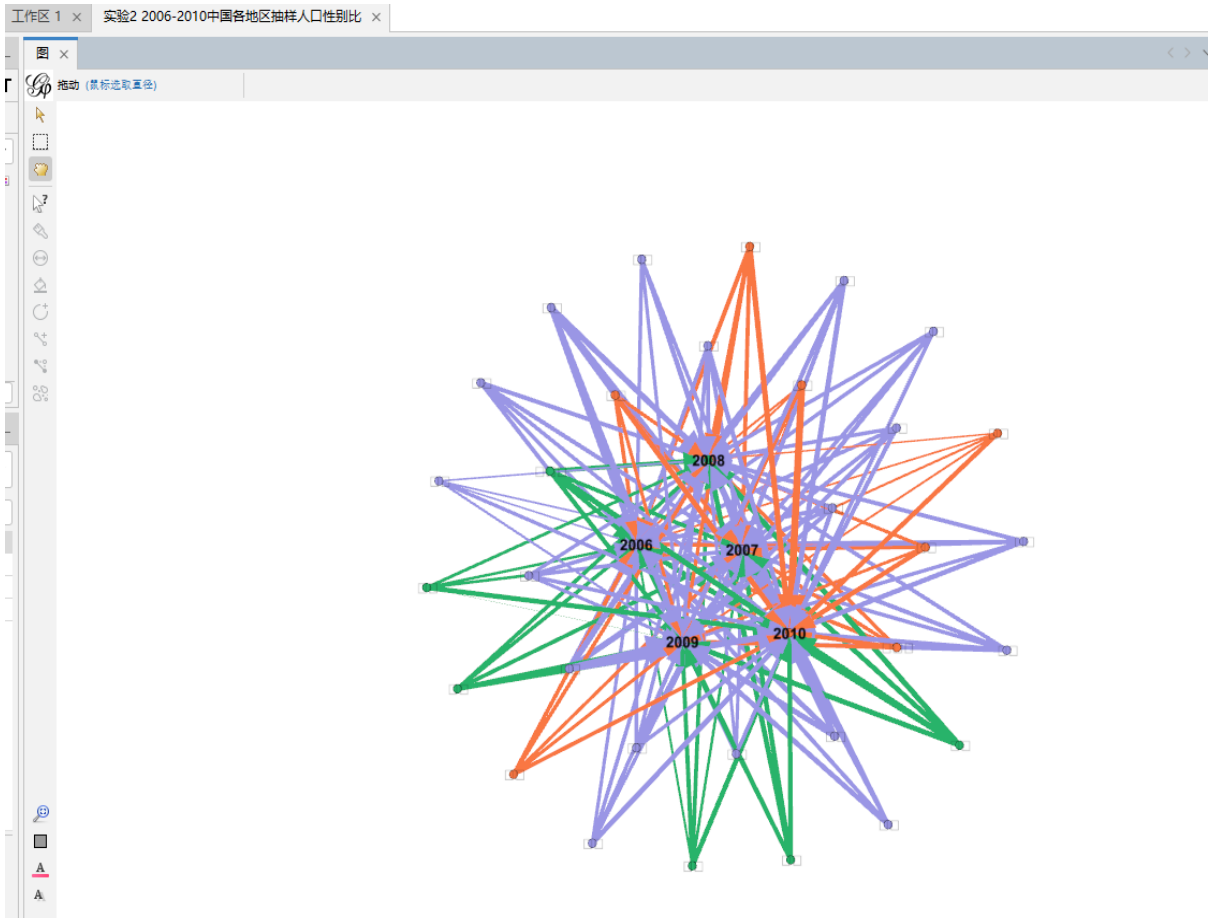
Echarts图表


全球科技公司合作关系网络
Force-directed layout
互联网巨头
芯片制造商
企业软件
电动汽车
消费电子
Google
分类: 互联网巨头
合作关系数: 5


2025年电商平台用户消费品类分布
基于真实消费数据抽样分析
服饰鞋包: 22.79%
食品生鲜: 17.13%
数码家电: 15.06%
美妆护肤: 12.59%
家居用品: 9.88%
母婴用品: 7.81%
运动户外: 5.98%
图书文创: 4.94%
其他品类: 3.82%
服饰鞋包
食品生鲜
数码家电
美妆护肤
家居用品
母婴用品
运动户外
图书文创
其他品类


2025年上半年电商平台销售额分析
单位: 万元
销售额 (万元)
实际销售额
目标销售额
1月
实际销售额: 450 万元
1,000
800
600
400
200
0
450
400
520
500
680
650
750
700
820
800
980
900
1月
2月
3月
4月
5月
6月

Gehpi图（以2006~2010中国各地区抽样性别比为例）



科学家文本的词云图

```
In [9]:
f_cn = open(r"D:\数据采集与关联分析\用户数据\第3讲 感知世界：词云与可视化\第3讲 感知世界：词云与可视化\
text_cn = f_cn.read()
f_cn.close()
text_cn[:50]
import jieba
text_cn_word = "/".join(jieba.cut(text_cn))
print(text_cn_word[:100])
print(text_cn_word)
from wordcloud import WordCloud
wordcloud = WordCloud().generate(text_cn_word)
from wordcloud import WordCloud
wordcloud_cn = WordCloud(font_path="simsun.ttf").generate(text_cn_word)
%pylab inline
import matplotlib.pyplot as plt
plt.imshow(wordcloud_cn, interpolation='bilinear')
plt.axis('off')
plt.show()
```

```
/屠/呦/呦/，/1930/年/12/月/30/日出/生于/浙江/宁波/，/著名/药学/家/，/中国/首位/诺贝尔/生理学/或/
医学
 /屠/呦/呦/：/青蒿素/的/发现者/与/疟疾/的/征服者/
/
/屠/呦/呦/，/1930/年/12/月/30/日出/生于/浙江/宁波/，/著名/药学/家/，/中国/首位/诺贝尔/生理学/或/
医学奖/获得者/，/"/共和国/勋章/"/获得者/，/现任/中国/中医/科学院/青蒿素/研究/中心/主任/、/终身/研
究员/兼/首席/研究员/。/
/
/ /成长/历程/
/
/屠/呦/呦/的/名字/取自/《/诗经/》/中/"/呦/呦/鹿鸣/"/，/她/的/人生/也/如同/这/诗句/一般/，/在/医
学/科研/领域/奏响/了/动人/乐章/。/1951/年/，/她/考入/北京/医学院/(/现/北京大学医学部/)/药学系/，/
1955/年/毕业/后/被/分配/至/卫生部/中医/研究院/(/现/中国/中医/科学院/)/工作/。/1959/年/，/她/参
加/"/全国/第三期/西医/离职/学习/中医/班/"/，/系统/学习/中医药/知识/，/为/日后/中西医/结合/研究/奠
定/了/坚实基础/。/
/
/ /青蒿素/的/发现/：/对抗/疟疾/的/里程碑/
/
/20/世纪/60/年代/，/全球/疟疾/疫情/严重/，/疟原虫/对/传统/奎宁/类药物/产生/抗性/，/100/多个/国家/
```

第四讲 情感分析

情感分析

123 情感分析

请输入一段中文文本:

“这是一部男人必看的电影。”人人都这么说。但单纯从性别区分，就会让这电影变狭隘。《肖申克的救赎》突破了男人电影的局限，通篇几乎充满令人难以置信的温馨基调，而电影里最伟大的主题是“希望”。当我们无奈地遇到了如同肖申克一般囚禁了心灵自由的那种囹圄，我们是无奈的老布鲁克，灰心的瑞德，还是智慧的安迪？运用智慧，信任希望，并且勇敢面对恐惧心理，去打败它？ 经典的电影之所以经典，因为他们都在做同一件事——让你从不同的角度来欣赏希望的美好。

215/1000

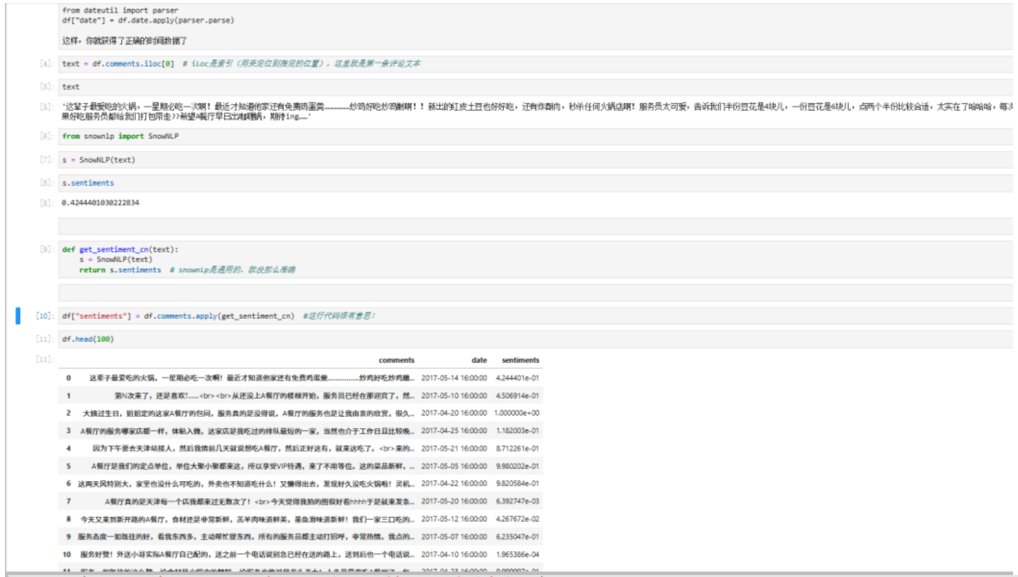
情感分析

情感极性

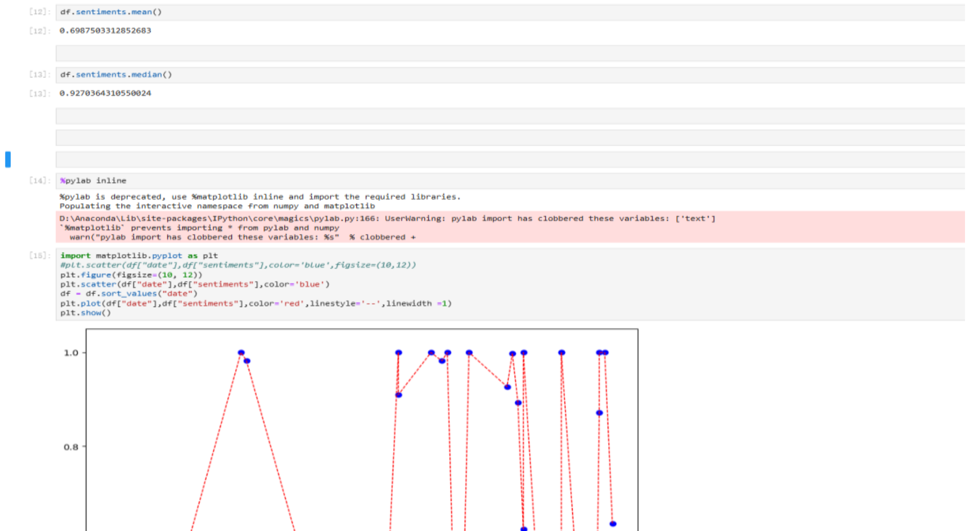
ppt代码运行

[1]: import pandas as pd # pandas用来处理表格数据的工具包
[2]: df = pd.read_excel("restaurant-comments.xlsx")
[3]: df.head()
comments
date
0 这辈子最爱吃的火锅，一星期必吃一次啊！最近才知道他家还有免费鸡蛋羹............炒鸡好吃炒鸡嫩... 2017-05-14 16:00:00
1 第N次来了，还是喜欢?......

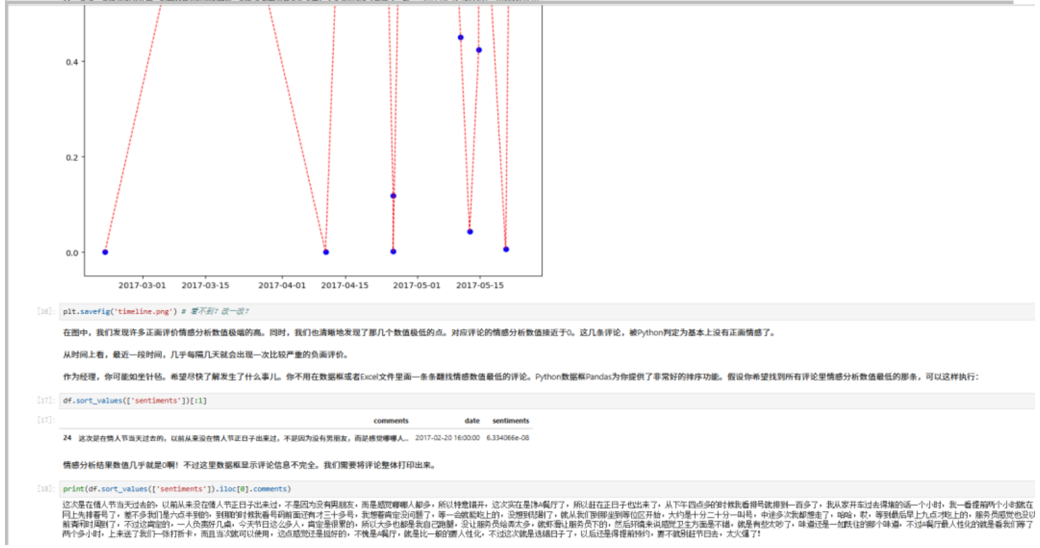
从还没上A餐厅的楼梯开始，服务员已经在那迎宾了，然... 2017-05-10 16:00:00
2 大姨过生日，姐姐定的这家A餐厅的包间，服务真的是没得说，A餐厅的服务也是让我由衷的欣赏，很久... 2017-04-20 16:00:00
3 A餐厅的服务哪家店都一样，体贴入微。这家店是我吃过的排队最短的一家，当然也介于工作日且比较晚... 2017-04-25 16:00:00
4 因为下午要去天津站接人，然后我借前几天就说想吃A餐厅，然后正好这有，就来这吃了。
来的... 2017-05-21 16:00:00



from dateutil import parser
df["date"] = df.date.apply(parser.parse)
text = df.comments.iloc[0]
from snownlp import SnowNLP
s = SnowNLP(text)
s.sentiments
0.4244401030222834
def get_sentiment_cn(text):
s = SnowNLP(text)
return s.sentiments
df["sentiments"] = df.comments.apply(get_sentiment_cn)
df.head(100)
comments
date
sentiments



df.sentiments.mean()
0.6987503312852683
df.sentiments.median()
0.9270364310550024
%pylab inline
import matplotlib.pyplot as plt
plt.figure(figsize=(10, 12))
plt.scatter(df["date"],df["sentiments"],color='blue')
df = df.sort_values("date")
plt.show()
1.0
0.8



0.4
0.2
0.0
2017-03-01
2017-03-15
2017-04-01
2017-04-15
2017-05-01
2017-05-15
plt.savefig('timeline.png')
df.sort_values(['sentiments'])[:1]
comments
date
sentiments
print(df.sort_values(['sentiments']).iloc[0].comments)

ppt代码运行

```
[1]: text = "I am happy today. I feel sad today."

[2]: from textblob import TextBlob
     blob = TextBlob(text)

[3]: blob

[3]: TextBlob("I am happy today. I feel sad today.")

[4]: # 原封不动的打印出来了？
     # 实际上已经把文本分成了句子了，看一看
     blob.sentences

[4]: [Sentence("I am happy today."), Sentence("I feel sad today.")]

[5]: blob.sentences[0].sentiment

[5]: Sentiment(polarity=0.8, subjectivity=1.0)

[6]: # 上面的结果什么意思呢？
     # 情感极性0.8，主观性1.0。说明一下，情感极性的变化范围是[-1, 1]，-1代表完全负面，1代表完全正面。
     # 我表达的是我很高兴，那么这个结果是对的

[7]: blob.sentences[1].sentiment

[7]: Sentiment(polarity=-0.5, subjectivity=1.0)

[8]: # 整段文本的情感呢？
     blob.sentiment

[8]: Sentiment(polarity=0.15000000000000002, subjectivity=1.0)

[9]: # 你可能会觉得没有道理，怎么一句"高兴"，一句"沮丧"，合并起来最后会得到正向结果呢？
     # 首先不同极性的词，在数值上是有区别的。我们应该可以找到比"沮丧"更为负面的词汇。而且这也符合逻辑，谁会这么"天上一脚，地下一脚"矛盾地描述自己此时的心情呢？

[21]: # 例如
      text_taobao_1 = "显示效果：挺帮的 运行速度：目前来说很满意 拍照效果：拍照效果挺帮的 电池续航：一天一冲 总结：目前没啥毛病，用了一段时间"

[22]: taobao_1 = SnowNLP(text_taobao_1)

[23]: taobao_1.sentiments

[23]: 0.999947261146611

[24]: text_taobao_2 = "总结：这是我买过最不满意的一款手机！两千多元的手机这样，真的很不值！"

[25]: taobao_2 = SnowNLP(text_taobao_2)

[26]: for sentence in taobao_2.sentences:
          print(sentence)

      总结：这是我买过最不满意的一款手机
      两千多元的手机这样
      真的很不值

[27]: taobao_2.sentiments

[27]: 0.889005139666256

[28]: # 以上的结果看上去是有问题的，分析的不准确，

[29]: text_taobao_3 = "显示效果：像素不行 运行速度：微信聊时发短不了语音，调重新开机后才能发，才买半个月的手机就这样，客服态度也很差！ 拍照效果：拍照不清晰！ 电池续航：手机不零电，充满电后用半天就没电了，一天得充两次电！！ 总结：这是我买过最不满意的一款手机"

[30]: taobao_3 = SnowNLP(text_taobao_3)

[31]: taobao_3.sentiments

[31]: 5.7076094222563434e-05
```

```
[10]: text_cn = u"我今天很快乐。我今天很愤怒。"

[11]: #注意在引号前面我们加了一个字母u，它很重要，因为它提示Python，“这一段我们输入的文本编码格式是Unicode，别搞错了哦”。至于文本编码格式的细节，有机会我们再详细聊。

[12]: from snownlp import SnowNLP

[13]: senti_cn = SnowNLP(text_cn)

[14]: # 看看snownlp包的分句能力
      for sentence in senti_cn.sentences:
          print(sentence)

      我今天很快乐
      我今天很愤怒

[15]: senti_cn_1 = SnowNLP(senti_cn.sentences[0])

[16]: # 一个细节上的问题，英文是x.sentiment，中文是x.sentiments，多了一个s
      # 另外，在句法上和英文的也略有不同，比如直接用语句：senti_cn.sentences[0].sentiments是会报错的
      senti_cn_1.sentiments

[16]: 0.971889316039116

[17]: senti_cn_2 = SnowNLP(senti_cn.sentences[1])

[18]: senti_cn_2.sentiments

[18]: 0.07763913772213482
```

这里你肯定发现了问题——“愤怒”这个词表达了如此强烈的负面情感，为何得分依然是正的？

这是因为SnowNLP和textblob的计分方法不同。SnowNLP的情感分析取值，表达的是“这句话代表正面情感的概率”。也就是说，对“我今天很愤怒”一句，SnowNLP认为，它表达正面情感的概率很低很低。

这样解释就是ok了

```
[19]: senti_cn.sentiments

[19]: 0.7237619924203508

[20]: # 整个句子，貌似就有问题了
```

```
[1]: import requests
     import json

     # DeepSeek API 端点
     url = "https://api.deepseek.com/v1/chat/completions"

     # 替换为您的 DeepSeek API 密钥
     API_KEY = "sk-78e664727760463394135e9246255cfa" # 直接复制过来

     # 请求头，包含 API 密钥和内容类型
     headers = {
         "Authorization": f"Bearer {API_KEY}",
         "Content-Type": "application/json"
     }

[2]: # 患者描述文本
     text = (
         "我今年58岁，退休后感到生活失去了重心，开始出现失眠、头痛和疲乏无力的症状。"
         "后来，皮肤变得异常敏感，连衣服的触碰都像针扎一样疼痛。"
         "我常常感到心慌、胸闷，背部沉重得像压了一块石头。"
         "对光线和声音变得极度敏感，电话铃声都会让我惊恐。"
         "多次到医院检查，结果都显示没有器质性疾病。"
         "我变得不愿出门，不想与人交流，整天把自己关在屋里，拉紧窗帘，感觉生活毫无意义。"
     )

     # 构建提示词，要求模型提取细粒度情感实体
     prompt = (
         "请从以下患者描述中提取出具体的身体部位、症状以及对应的情感状态，"
         "并以 JSON 格式返回，格式如下："
         "{'实体': [{'部位': '...', '症状': '...', '情感': '...'}, ...]}\n\n"
         f"患者描述：{text}"
     )

[3]: # 请求体，包含模型参数和提示词
     data = {
         "model": "deepseek-chat",
         "messages": [
             {"role": "user", "content": prompt}
         ],
         "temperature": 0.3
     }

     try:
         # 发送 POST 请求
         response = requests.post(url, headers=headers, data=json.dumps(data))

         # 检查响应状态码
         if response.status_code == 200:
             # 解析 JSON 响应
             result = response.json()
             # 提取模型生成的内容
             generated_text = result['choices'][0]['message']['content']
             print("细粒度情感实体抽取结果：")
             print(generated_text)
         else:
```

ppt代码运行

```
[3]: # 请求体，包含模型参数和提示词
data = {
    "model": "deepseek-chat",
    "messages": [
        {"role": "user", "content": prompt}
    ],
    "temperature": 0.3
}

try:
    # 发送 POST 请求
    response = requests.post(url, headers=headers, data=json.dumps(data))

    # 检查响应状态码
    if response.status_code == 200:
        # 解析 JSON 响应
        result = response.json()
        # 提取模型生成的内容
        generated_text = result['choices'][0]['message']['content']
        print("细粒度情感实体抽取结果:")
        print(generated_text)
    else:
        # 处理错误响应
        print(f"请求失败，状态码: {response.status_code}")
        print(f"错误信息: {response.text}")

except requests.exceptions.RequestException as e:
    # 处理网络请求异常
    print(f"网络请求失败: {e}")
except json.JSONDecodeError as e:
    # 处理 JSON 解析异常
    print(f"JSON 解析失败: {e}")
except Exception as e:
    # 处理其他异常
    print(f"发生未知错误: {e}")
```

```
        {
          "部位": "耳朵",
          "症状": "对声音极度敏感，电话铃声引发惊恐",
          "情感": "惊恐"
        },
        {
          "部位": "全身",
          "症状": "不愿出门，不想与人交流，整天关在屋里拉紧窗帘",
          "情感": "生活毫无意义"
        }
    ]
}
```

```
直接用如下格式输出：
情感词：XXX
情感值：X.X

句子：{sent}"""

    data = {
        "model": "deepseek-chat",
        "messages": [{"role": "user", "content": prompt}],
        "temperature": 0.3
    }

    try:
        response = requests.post(API_URL, headers=headers, json=data)
        time.sleep(0.1)  # 控制请求频率

        if response.status_code == 200:
            content = response.json()['choices'][0]['message']['content']

            # 宽容匹配格式，如：情感词：悲伤，情感值：-0.6
            match = re.search(r"情感词[：:]?\s*([^\n\r：:\s]+)\s*[\n\r]+情感值[：:]?\s*(-?\d+(?:\.\d+)?)", content)
            if match:
                word = match.group(1)
                score = float(match.group(2))
                emotion_results.append((num, sent, word, score))
            else:
                print(f"[⚠ 无法匹配] 第{num}句: {sent}")
                print("模型输出:", content)
                emotion_results.append((num, sent, "无", 0.0))
        else:
            print(f"[✗ API错误 {response.status_code}]")
            print(response.text)
            emotion_results.append((num, sent, "无", 0.0))

    except Exception as e:
        print(f"[异常] 第{num}句处理失败: {e}")
        emotion_results.append((num, sent, "无", 0.0))
```

```
[5]: for item in emotion_results:
    print(f"句子{item[0]}: 情感词: {item[2]}, 情感值: {item[3]}")
```

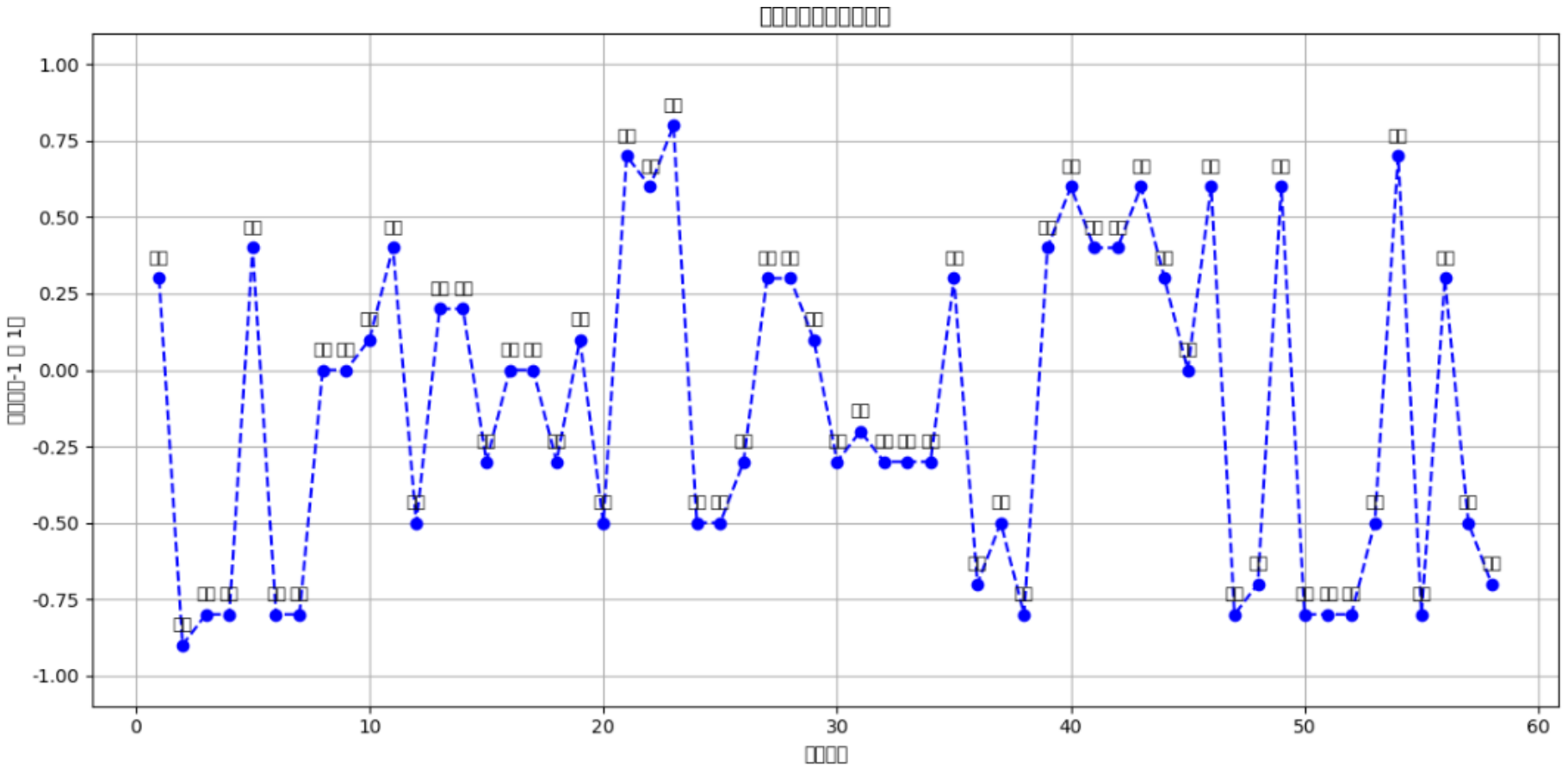
```
句子1: 情感词: 怀念, 情感值: 0.3
句子2: 情感词: 悲伤, 情感值: -0.9
句子3: 情感词: 悲伤, 情感值: -0.5
句子4: 情感词: 悲伤, 情感值: -0.8
句子5: 情感词: 安慰, 情感值: 0.3
句子6: 情感词: 悲伤, 情感值: -0.8
句子7: 情感词: 惨澹, 情感值: -0.8
句子8: 情感词: 平静, 情感值: 0.0
句子9: 情感词: 平静, 情感值: 0.0
句子10: 情感词: 平静, 情感值: 0.0
句子11: 情感词: 仔细, 情感值: 0.4
句子12: 情感词: 嫌隙, 情感值: -0.3
```

```
import requests
import time
import re

API_KEY = "sk-78e664727760463394135e9246255cfa" #替换成你自己的密钥
API_URL = "https://api.deepseek.com/v1/chat/completions"

headers = {
    "Content-Type": "application/json",
    "Authorization": f"Bearer {API_KEY}"
}

emotion_results = []

for num, sent in numbered_sentences:
    prompt = f"""你是一个情感分析助手。请对下面这句话做如下分析：

1. 提取或生成一个最能代表情绪的词语（只能一个）。
2. 对整句话的情绪进行打分，范围是 -1（非常负面）到 1（非常正面），中性为 0。

直接用如下格式输出：
情感词：XXX
情感值：X.X

句子：{sent}"""

    data = {
        "model": "deepseek-chat",
        "messages": [{"role": "user", "content": prompt}],
        "temperature": 0.3
    }

    try:
        response = requests.post(API_URL, headers=headers, json=data)
        time.sleep(0.1)  # 控制请求频率

        if response.status_code == 200:
            content = response.json()['choices'][0]['message']['content']

            # 宽容匹配格式，如：情感词：悲伤，情感值：-0.6
            match = re.search(r"情感词[：:]?\s*([^\n\r：:\s]+)\s*[\n\r]+情感值[：:]?\s*(-?\d+(?:\.\d+)?)", content)
            if match:
                word = match.group(1)
                score = float(match.group(2))
                emotion_results.append((num, sent, word, score))
            else:
                print(f"[⚠ 无法匹配] 第{num}句: {sent}")
                print("模型输出:", content)
                emotion_results.append((num, sent, "无", 0.0))
        else:
            print(f"[✗ API错误 {response.status_code}]")
            print(response.text)
            emotion_results.append((num, sent, "无", 0.0))
```

```
import matplotlib.pyplot as plt
from matplotlib import rcParams

# 设置中文字体（适配不同系统）
#plt.rcParams['font.family'] = 'SimHei'  # 黑体，适用于 Windows
plt.rcParams['font.family'] = 'Heiti TC' #mac对应的黑体
plt.rcParams['axes.unicode_minus'] = False  # 解决负号 '-' 显示为方块的问题

# 如果你使用的是 macOS，可换为：
# plt.rcParams['font.family'] = 'Heiti TC'

# 如果是 Linux（如 Ubuntu）且安装了中文字体：
# plt.rcParams['font.family'] = 'Noto Sans CJK SC'

# 画图
x = [item[0] for item in emotion_results]
y = [item[3] for item in emotion_results]
labels = [item[2] for item in emotion_results]

plt.figure(figsize=(12, 6))
plt.plot(x, y, linestyle='--', marker='o', color='blue')

# 添加每个点的中文情感词标签
for i in range(len(x)):
    plt.text(x[i], y[i] + 0.05, labels[i], ha='center', fontsize=9)

plt.title("散文情感词时间序列图")
plt.xlabel("句子编号")
plt.ylabel("情感值（-1 到 1）")
plt.ylim(-1.1, 1.1)
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
findfont: Font family 'Heiti TC' not found.
findfont: Font family 'Heiti TC' not found.
findfont: Font family 'Heiti TC' not found.
findfont: Font family 'Heiti TC' not found.
findfont: Font family 'Heiti TC' not found.
findfont: Font family 'Heiti TC' not found.
findfont: Font family 'Heiti TC' not found.
findfont: Font family 'Heiti TC' not found.
findfont: Font family 'Heiti TC' not found.
C:\Users\19798\AppData\Local\Temp\ipykernel_47060\1526999595.py:32: UserWarning: Glyph 21477 (\N{CJK UNIFIED IDEOGRAPH-53E5}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
C:\Users\19798\AppData\Local\Temp\ipykernel_47060\1526999595.py:32: UserWarning: Glyph 23376 (\N{CJK UNIFIED IDEOGRAPH-5B50}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
C:\Users\19798\AppData\Local\Temp\ipykernel_47060\1526999595.py:32: UserWarning: Glyph 32534 (\N{CJK UNIFIED IDEOGRAPH-7F16}) missing from font(s) DejaVu Sans.
  plt.tight_layout()
C:\Users\19798\AppData\Local\Temp\ipykernel_47060\1526999595.py:32: UserWarning: Glyph 21495 (\N{CJK UNIFIED IDEOGRAPH-53F7}) missing from font(s) DejaVu Sans.
```

ppt代码运行



代码分析

这些代码核心实现了中文文本情感分析、数据可视化与大模型信息抽取的 NLP 基础流程：在餐厅评论情感分析模块，通过 pandas 读取 Excel 评论数据并借助 dateutil 解析日期格式，利用 SnowNLP 计算单条及批量评论的情感得分（得分越接近 1 正面情感概率越高），定义 get_sentiment_cn 函数结合 apply 为数据新增 sentiments 列，还计算出情感得分均值 0.698、中位数 0.927，通过 matplotlib 绘制 “日期 - 情感得分” 散点 + 折线图展现情感随时间波动，并利用 sort_values 筛选出得分最低的评论定位负面反馈

第六讲 知识图谱理念

阿里商品大脑案例分析

2025年，阿里商品大脑的知识图谱生态在淘系电商的“场景化导购”业务中实现了深度落地，有效解决了传统导购“需求匹配不精准、用户决策成本高”的问题。该案例以用户场景需求为核心，通过知识图谱的多维度关联，构建“场景-品类-商品-属性”的全链路导购路径，具体实现流程如下：

第一步，需求场景化识别。当用户在淘系搜索“户外烧烤”时，知识图谱首先通过场景图谱将query映射为“户外烧烤场景”，并挖掘出该场景下的核心需求：烧烤工具（烤架、炭火）、烧烤食材（肉串、蔬菜）、辅助用品（一次性餐具、野餐垫）、安全用品（灭火器、防烫手套）等细分需求维度。同时，结合用户画像（如是否有儿童、购买力水平），进一步细化需求，例如若用户画像包含“儿童”标签，则优先关联“儿童安全烤架”“无添加食材”等细分属性。

第二步，品类与商品关联。基于场景与品类的关联关系，知识图谱从商品图谱中筛选出符合需求的核心品类，并通过CPV属性匹配，筛选出高适配度商品。例如，针对“户外烧烤”场景，关联“便携式烧烤架”品类，并筛选出“折叠设计”“耐高温材质”“适合3-5人使用”等属性匹配的商品；针对“烧烤食材”品类，筛选出“冷链运输”“已腌制”“新鲜日期”等属性的商品。

第三步，场景化内容生成与推荐。结合多模态知识，为用户生成场景化导购页面：以视频形式展示烧烤架的使用场景，以图片形式呈现食材的新鲜度，以文本形式提供烧烤攻略（如腌制方法、烧烤时间）。同时，通过知识推理挖掘关联场景，为用户推荐“户外露营”“野餐”等相关场景的商品，实现“一站式购物”体验。

该案例的落地效果显著：在“户外烧烤”“备孕”等核心场景中，用户搜索转化率提升35%，人均浏览商品数量增加2.3件，用户停留时长提升40%；商家端的商品曝光精准度提升50%，无效点击成本降低28%。这一实践充分验证了知识图谱在打通人-货-场链路、提升电商交易效率中的核心价值。

阿里商品大脑案例分析

一、阿里商品大脑知识图谱生态构建的背景动因

随着阿里业务边界的持续拓展，从传统淘系电商到闲鱼、新零售等多元场景，数据互联与深度认知的需求日益迫切。这一需求的背后，是当前电商数据应用面临的四大核心痛点，也是其知识图谱生态构建的核心动因。

其一，非结构化数据占比高且噪声密集。当前阿里体系内的核心数据多为用户查询词（query）、商品标题、用户评论、购物攻略等非结构化文本，这些数据受用户表达习惯与商家营销诉求影响，存在语法不规范、关键词堆砌、虚假宣传等问题，给用户需求的精准识别带来极大挑战。例如，商家为提升曝光率，可能在商品标题中混入与核心品类无关的热门词汇，导致传统检索算法难以区分真实需求与干扰信息。

其二，多模态、多源数据融合难度大。随着短视频、直播电商的兴起，商品信息已不再局限于文本，图片、视频等多模态数据成为核心内容载体。同时，不同业务线（如淘系、闲鱼、盒马）的数据分散存储，形成“数据孤岛”，如何实现跨模态、跨业务的数据关联与整合，成为支撑“全域购物场景”的关键障碍。

其三，类目体系碎片化，缺乏统一标准。不同业务线因场景属性差异，需维护独立的CPV（类目-属性-属性值）体系。例如，闲鱼的“包配饰”因二手交易高频需求需精细划分，而淘系中“鞋包配饰”仅为二级类目下的细分品类，这种碎片化的类目体系导致跨业务检索、商品关联推荐需重复开发，极大提升了运营成本。

其四，缺乏对用户需求的深度认知。传统电商算法多聚焦于商品本身的属性匹配，而难以挖掘用户需求背后的深层关联。例如，用户搜索“叶酸”时，算法需识别其“备孕”的核心需求，并关联推荐孕期相关商品；用户频繁浏览烧烤调料与工具时，需精准洞察其“户外烧烤”的场景需求。这种需求的深度认知，正是提升用户购物体验、实现“主动导购”的核心前提。

在此背景下，阿里商品大脑启动电商认知图谱的生态构建，旨在建立一套全局统一的知识表示与查询框架，通过结构化处理复杂数据、融合分散资源、深度解读用户需求，实现人-货-场的精准联动，为全业务线提供标准化的知识服务支撑。

阿里商品大脑案例分析

二、核心架构与最新进展

2025年，阿里商品大脑电商认知图谱形成“四层架构（用户、场景、品类、商品图谱）+全链路运营”体系，通过异构图实现多维度知识关联，全链路运营保障知识准确与时效。最新进展核心体现在三方面：

（一）全局统一Schema体系完善落地

为解决数据分散与类目碎片化问题，阿里构建电商专属全局Schema体系，明确实体类型、属性及60余种核心关系定义。2025年新增多语言实体映射模块，融合外网知识与行业标准，支撑跨境电商。

同时建立“算法+人工”审核机制，保障知识质量。截至2025年8月，该体系覆盖全业务线95%以上核心实体，融合68.9万+知识对，实现跨场景数据互联标准化。

（二）多模态抽取与场景图谱精细化升级

2025年升级多模态知识抽取框架，实现文本、图片、视频协同抽取：文本层面完成核心类目CPV挖掘，query全量识别率提升至60%；图像层面商品属性识别准确率达89%；视频层面挖掘商品使用场景。

场景图谱从通用升级为精细化，抽象10万+场景概念，建立多维度关联，实现需求分层覆盖，其关联强度计算能力为精准推荐提供核心支撑。

（三）知识服务平台生态化输出与动态迭代

2025年构建标准化知识服务平台，向全集团输出基础检索、智能推荐、数据治理三大核心服务，降低业务线研发与治理成本。

平台建立动态迭代机制，通过实时数据监控触发更新，定期审核优化，可快速响应露营经济等热点需求，保障知识时效性。

阿里商品大脑案例分析

三、典型应用案例：淘系电商“场景化导购”的落地实践

2025年，阿里商品大脑知识图谱生态深度落地淘系电商“场景化导购”业务，精准解决传统导购需求匹配不准、用户决策成本高的痛点。该案例以用户场景需求为核心，构建“场景-品类-商品-属性”全链路导购路径，核心流程如下：其一，需求场景化识别。用户搜索“户外烧烤”时，知识图谱将查询词映射为对应场景，挖掘出烧烤工具、食材、辅助用品等细分需求，结合用户画像（如是否有儿童）进一步细化属性匹配；其二，品类与商品关联。依托场景与品类的关联关系筛选核心品类，通过CPV属性匹配筛选高适配商品，如为户外烧烤场景匹配折叠、耐高温的便携式烤架及冷链运输的腌制食材；其三，场景化内容生成与推荐。融合多模态知识生成导购页面，同步挖掘关联场景推荐相关商品，实现一站式购物。案例落地成效显著：核心场景用户搜索转化率提升35%，人均浏览商品数增加2.3件，停留时长提升40%；商家端商品曝光精准度提升50%，无效点击成本降低28%，充分验证了知识图谱打通人-货-场链路的核心价值。

四、阿里商品大脑知识图谱生态构建的评价与启示

阿里商品大脑的电商认知图谱生态，是垂直领域知识图谱构建的典型范本，其成功实践既体现了显著的优势与价值，也暴露出行业共性的挑战，为后续企业的知识图谱建设提供了重要启示。一是精准把握垂直领域痛点，实现知识体系的定制化构建。阿里未采用通用知识图谱的构建思路，而是针对电商场景的非结构化数据噪声、类目碎片化、需求深层挖掘等核心痛点，构建了以“场景认知”为核心的电商专属知识体系。这种定制化思路确保了知识图谱与业务的高度适配，使得知识服务能直接解决业务痛点，提升商业价值。例如，场景图谱的构建直接回应了用户需求深层关联的痛点，成为精准推荐的核心支撑。

二是构建“技术+运营”的双轮驱动模式，保障知识质量与效率。阿里通过算法实现知识的自动抽取、融合与迭代，提升构建效率；通过专业团队的人工审核，保障知识的准确性与权威性。这种双轮驱动模式平衡了效率与质量的矛盾，解决了纯算法构建知识图谱易出现的错误关联、噪声知识等问题，为知识图谱的商业化应用奠定了基础。

阿里商品大脑案例分析

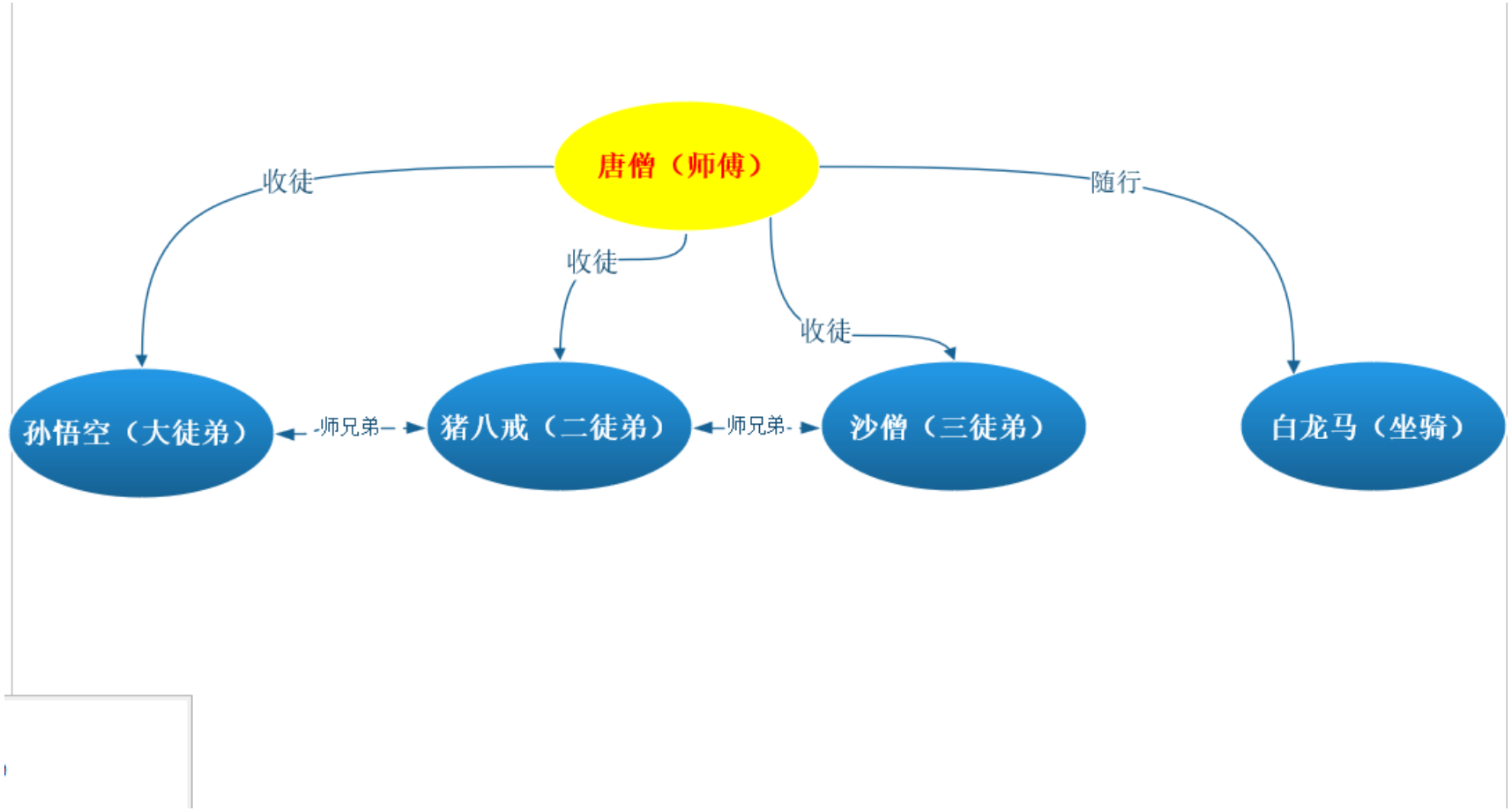
三是实现知识能力的生态化输出，提升全链路协同效率。阿里将知识图谱能力封装为标准化服务平台，向全集团各业务线输出，打破了“数据孤岛”与“能力孤岛”，实现了知识资源的共享与复用。这种生态化输出模式不仅降低了各业务线的技术研发成本，还确保了全集团数据与知识的一致性，提升了跨业务协同效率，为全域电商战略的推进提供了核心支撑。

五、总结

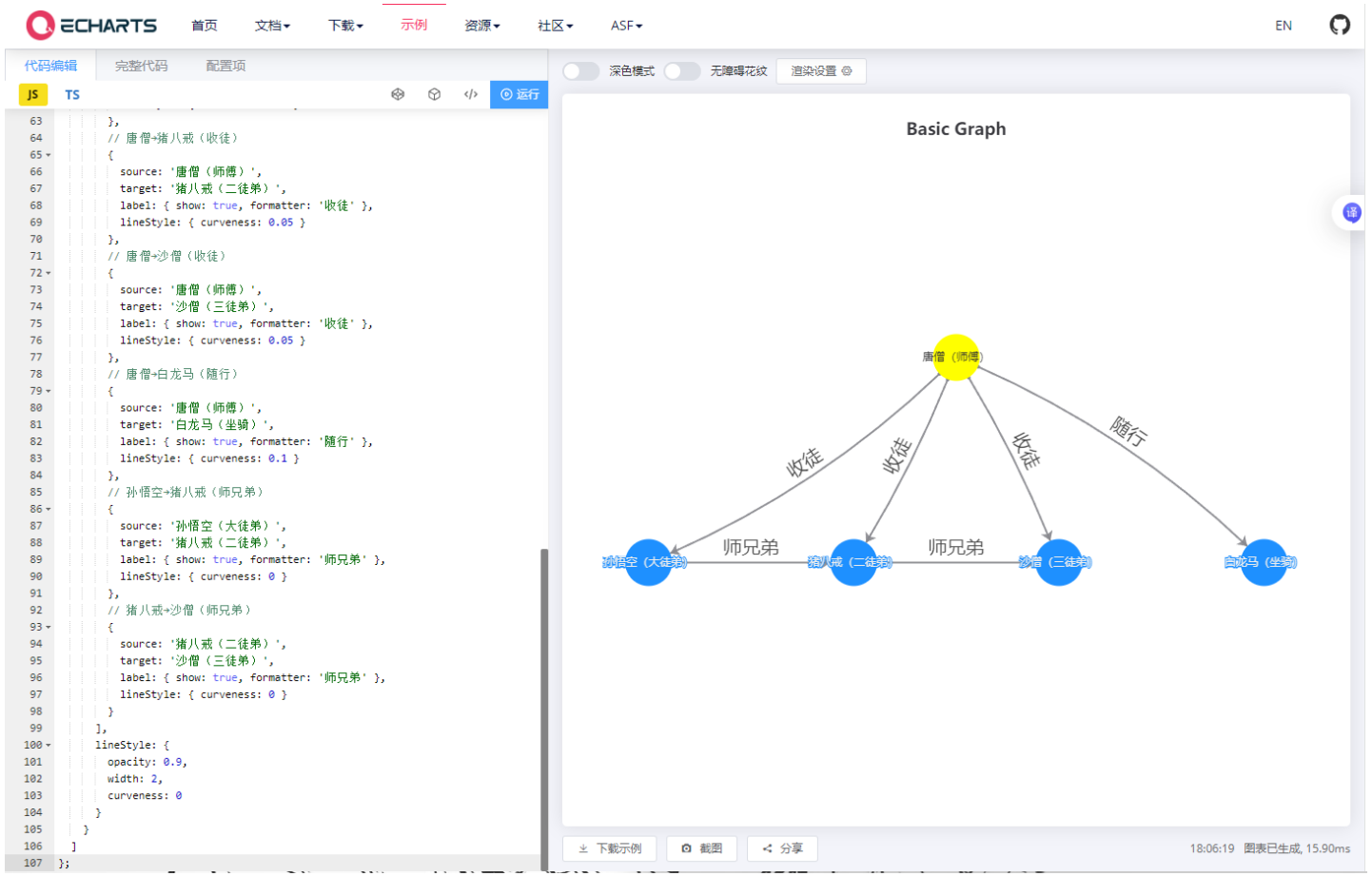
2025年阿里商品大脑的电商认知图谱生态，通过全局统一Schema体系、多模态知识抽取、生态化知识服务平台的构建，成功解决了电商场景下的数据互联、需求认知、跨业务协同等核心痛点，在场景化导购等业务中实现了商业价值的显著提升。其“定制化体系+双轮驱动+生态输出”的构建模式，为垂直领域知识图谱的建设提供了可借鉴的范本。同时，其面临的多模态融合深度不足、动态更新滞后、合规压力等挑战，也是全行业需要共同攻克的课题。未来，随着人工智能技术的持续演进与行业标准的不断完善，阿里商品大脑的知识图谱生态有望实现更深度的多模态融合、更快速的动态迭代与更广泛的生态协同，为电商行业的智能化升级注入更强动力。

知识图谱（基于OpenKG）

白板建模

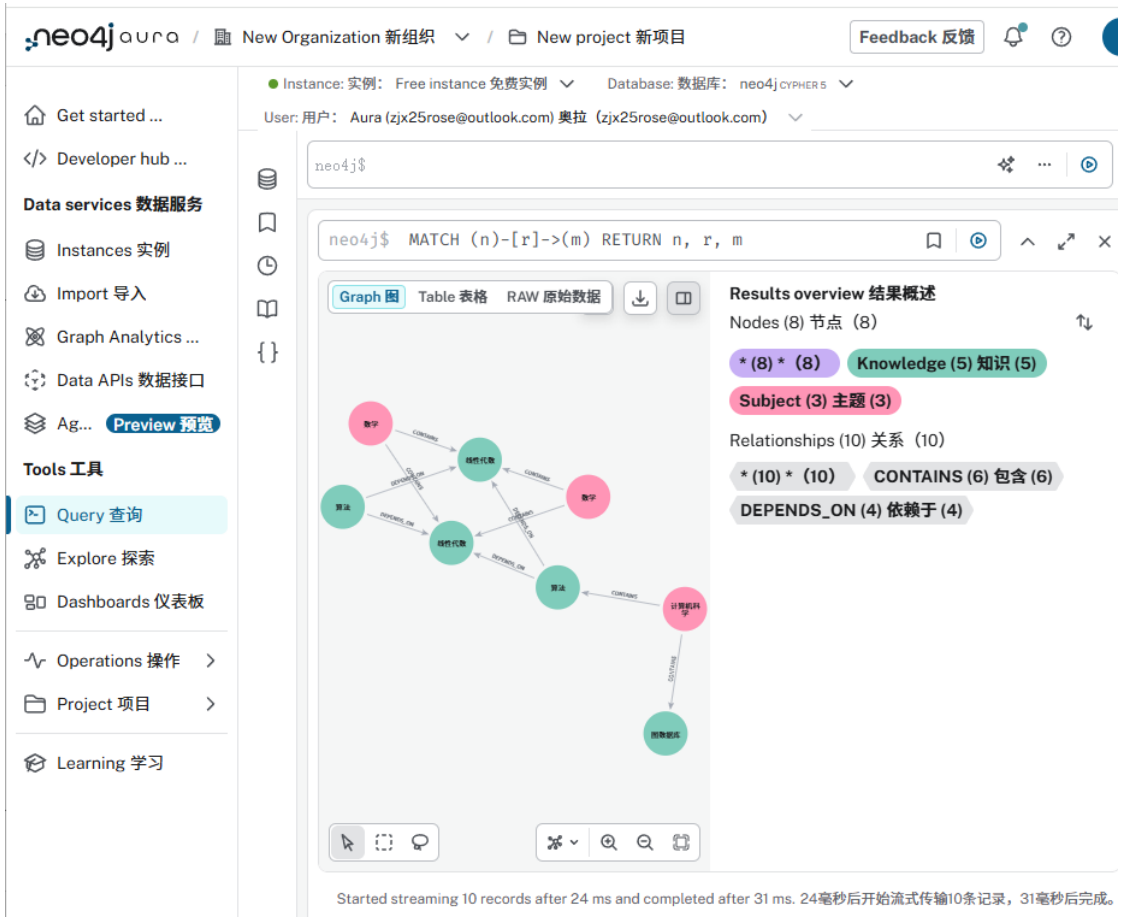


echarts




```
    },
    // 唐僧→猪八戒（收徒）
    {
      source: '唐僧（师傅）',
      target: '猪八戒（二徒弟）',
      label: { show: true, formatter: '收徒' },
      lineStyle: { curveness: 0.05 }
    },
    // 唐僧→沙僧（收徒）
    {
      source: '唐僧（师傅）',
      target: '沙僧（三徒弟）',
      label: { show: true, formatter: '收徒' },
      lineStyle: { curveness: 0.05 }
    },
    // 唐僧→白龙马（随行）
    {
      source: '唐僧（师傅）',
      target: '白龙马（坐骑）',
      label: { show: true, formatter: '随行' },
      lineStyle: { curveness: 0.1 }
    },
    // 孙悟空→猪八戒（师兄弟）
    {
      source: '孙悟空（大徒弟）',
      target: '猪八戒（二徒弟）',
      label: { show: true, formatter: '师兄弟' },
      lineStyle: { curveness: 0 }
    },
    // 猪八戒→沙僧（师兄弟）
    {
      source: '猪八戒（二徒弟）',
      target: '沙僧（三徒弟）',
      label: { show: true, formatter: '师兄弟' },
      lineStyle: { curveness: 0 }
    }
  ],
  lineStyle: {
    opacity: 0.9,
    width: 2,
    curveness: 0
  }
 }
]
};
```

Neo4j


neo4j aura / New Organization 新组织 / New project 新项目
Feedback 反馈
Instance: 实例： Free instance 免费实例
Database: 数据库： neo4j CYPHER 5
User: 用户： Aura (zjx25rose@outlook.com) 奥拉（zjx25rose@outlook.com）
Get started ...
Developer hub ...
Data services 数据服务
Instances 实例
Import 导入
Graph Analytics ...
Data APIs 数据接口
Ag... Preview 预览
Tools 工具
Query 查询
Explore 探索
Dashboards 仪表板
Operations 操作
Project 项目
Learning 学习
neo4j$
neo4j$ MATCH (n)-[r]->(m) RETURN n, r, m
Graph 图
Table 表格
RAW 原始数据
Results overview 结果概述
Nodes (8) 节点（8）
* (8) * (8)
Knowledge (5) 知识 (5)
Subject (3) 主题 (3)
Relationships (10) 关系（10）
* (10) * (10)
CONTAINS (6) 包含 (6)
DEPENDS_ON (4) 依赖于 (4)
Started streaming 10 records after 24 ms and completed after 31 ms. 24毫秒后开始流式传输10条记录，31毫秒后完成。