安徽大学人工智能学院《知识图谱》实验报告

学号 WA2214014 姓名 杨跃浙 时间 05.19

【实验名称】 知识图谱第一次实验

【实验内容】

题目 1：获取结构化数据
任务： 编写一个 Python 脚本，从一个包含产品信息的 CSV 文件中读取数据，并计算每个类别的平均价格。
提示： 使用 pandas 库。
示例 CSV 文件内容：

test.csv
category,product_name,price
Electronics,Smartphone,699
Electronics,Laptop,999
Clothing,T - shirt,19
Clothing,Jeans,49
Home,Blender,59
Home,Coffee Maker,89

题目 2：获取非结构化数据
任务： 编写一个 Python 脚本，从一个包含英文文本的文件中，提取所有的句子，并统计每个句子的单词数。
提示： 使用 nltk 库。
示例文本文件内容：
test.txt
Natural Language Processing (NLP) is a fascinating field of study. It involves the interaction between computers and humans through language. NLP is used in various applications such as chatbots, translation, and sentiment analysis.

题目 3：获取半结构化数据
任务： 编写一个 Python 脚本，从一个包含产品信息的 JSON 文件中，提取所有产品的名称和价格，并计算总价格。
提示： 使用 json 库。
示例 JSON 文件内容：

test.json
```
{
    "products": [
```

```
        {"name": "Smartphone", "price": 699},
        {"name": "Laptop", "price": 999},
        {"name": "T - shirt", "price": 19},
        {"name": "Jeans", "price": 49},
        {"name": "Blender", "price": 59},
        {"name": "Coffee Maker", "price": 89}
    ]
}
```

【实验代码】

题目 1：获取结构化数据

```
import pandas as pd

# 读取 CSV 文件
df = pd.read_csv('./KG-Class/Project1/test.csv')

# 按类别分组并计算平均价格
avg_prices = df.groupby('category')['price'].mean()

# 打印结果
print("每个类别的平均价格：")
print(avg_prices)
```

题目 2：获取非结构化数据

```
import nltk
nltk.data.path.append('/home/yyz/KG-Class/Project1/nltk_data')
# nltk.download('punkt_tab')
from nltk.tokenize import sent_tokenize, word_tokenize

# 读取文本文件
with open('./KG-Class/Project1/test.txt', 'r') as file:
text = file.read()

# 分句
sentences = sent_tokenize(text)
```

```
# 统计每句的单词数
print("每个句子的单词数（含符号）：")
for i, sentence in enumerate(sentences, 1):
word_count = len(word_tokenize(sentence))
print(f"句子 {i}: {word_count} 个单词")
```

题目 3：获取半结构化数据

```
import json

# 读取 JSON 文件
with open('./KG-Class/Project1/test.json', 'r') as file:
data = json.load(file)

# 提取名称和价格，计算总价
total_price = 0
print("产品名称与价格：")
for product in data['products']:
name = product['name']
price = product['price']
total_price += price
print(f"{name}: ${price}")

# 打印总价格
print(f"\n总价格：${total_price}")
```

【实验结果】

题目 1：获取结构化数据

```
(yyzttt) yyz@4028Dog:~$ /usr/loc
每个类别的平均价格：
category
Clothing        34.0
Electronics    849.0
Home            74.0
Name: price, dtype: float64
```

题目 2：获取非结构化数据

题目 3：获取半结构化数据

【实验总结】

本次实验我在 Mac 系统下通过 VSCode 远程连接 Linux 服务器完成，使用的是之前已配置好的 yyzttt 环境，Torch 版本为 1.9.1，CUDA 版本为 11.7，没有重新配置新环境。实验里的三个题目分别围绕结构化、非结构化和半结构化数据的处理展开，我通过运用不同的 Python 库实现了数据提取与计算任务。

在处理题目 1 的结构化数据时，我借助 pandas 库读取 CSV 文件，然后按类别分组计算平均价格。整个过程进行得很顺利，pandas 在数据处理与分析方面的高效性体现得很明显，它内置的分组聚合功能不用我写复杂的循环逻辑，就能快速对结构化数据进行统计操作，轻

松实现了预期目标。

题目 2 是对非结构化文本数据提取句子并统计单词数，我用了 nltk 库的 sent_tokenize 和 word_tokenize 函数。不过在这个过程中碰到了问题：服务器没办法直接通过 nltk.download('punkt_tab')下载分词模型。于是我从官网手动下载了 punkt.zip 压缩包，把它上传到服务器，同时将 nltk 库降级到稳定的 3.8.1 版本，这才解决了依赖问题，成功完成句子分割与单词计数。这次经历让我意识到，处理非结构化数据时，库版本兼容性和资源获取方式非常重要。

处理题目 3 的半结构化 JSON 数据时，我利用 json 库读取文件，提取产品名称与价格，再通过遍历列表计算总价格。这个过程比较顺畅，json 库的 load 函数能直接把 JSON 格式数据转换成 Python 字典结构，方便我进行后续的数据遍历和计算，清晰展现了半结构化数据处理的流程和关键操作。

总的来说，这次实验让我巩固了对不同类型数据处理方法的理解，在解决实际问题的过程中也积累了不少经验，像手动配置 nltk 资源和处理库版本冲突等。这些实践经历提升了我在复杂环境下进行数据处理的能力，也为我后续知识图谱构建中涉及的数据获取与预处理环节奠定了基础。未来我还需要进一步熟悉各类库的高级功能，这样才能应对更复杂的数据处理需求。

punkt.zip 下载链接：

https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/package

s/tokenizers/punkt.zip

代码开源在：

https://github.com/Bean-Young/Misc-Projects