CS5481: Data Engineering - Assignment1

Question 1 - Data Acquisition (20 marks)
Social media content, including blogs, articles, news, and Twitter posts, provides valuable insights for data science. Acquiring high-quality social media data is essential yet challenging. While crowdsourcing is an option, it can be costly; thus, we prefer to collect data through web scraping.

Please collect 30 pieces of social media content from designated websites, ensuring that the data meets the following criteria:

- Include articles, blogs, news, or posts along with their comments.
- Focus solely on the textual content.
- Ensure the data is formatted in a structured way (e.g., JSON or CSV).

The following social media platforms are recommended for your data collection:
https://english.news.cn
https://www.bbc.com/news
https://medium.com
https://twitter.com

Please submit both your code and the collected social media data.

Solution:
Marking Scheme:
1. 30 pieces of data. (5 marks)
2. Basic use of Python crawler. (10 marks)
3. Capture and show the scraped content. (3 marks)
4. Obtain the links on the homepage and then get the content from the links. (2 marks)

Code:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Function to scrape data from a specified URL
def scrape_data(url):
    response = requests.get(url)
    soup = BeautifulSoup(response.content, 'html.parser')

    # Example: Extracting article titles and comments
    titles = []
    comments = []

    for article in soup.find_all('article'):   # Adjust the tag based on the website structure
```

在每个 <article> 块中，查找 <h2> 标签
提取标题，存入 titles 列表。

```
        title = article.find('h2')  # Adjust based on the website structure
        if title:
            titles.append(title.get_text(strip=True))
        comment_section = article.find_all('p')  # Adjust based on the website structure
        for comment in comment_section:
            comments.append(comment.get_text(strip=True))
    return titles, comments

# List of URLs to scrape
urls = [
    "https://english.news.cn",  # Example URL
    "https://www.bbc.com/news",
    "https://medium.com",
    "https://twitter.com"
]

# Data storage
all_titles = []
all_comments = []

# Scraping each URL
for url in urls:
    titles, comments = scrape_data(url)
    all_titles.extend(titles)
    all_comments.extend(comments)

# Create a DataFrame and save to CSV
data = pd.DataFrame({
    'Title': all_titles,
    'Comment': all_comments
})

data.to_csv('social_media_data.csv', index=False)
print("Data scraped and saved to social_media_data.csv")
```

get_text(strip=True)：提取文本内容并去除多余空格。

查找 <p> 标签提取评论，存入 comments 列表。

使用 extend 方法将抓取的结果合并到全局列表中。

Question 2 - Data Preprocessing (30 marks)

Regular Expressions, abbreviated as Regex or Regexp, are a string of characters created within the framework of Regex syntax rules. You can easily manage your data with Regex, which uses commands li finding, matching, and editing. Regex is an important tool during the data preprocessing stage.

We take some exercises about regular expressions in Python

1. Write a pattern to check if a string contains only letters (both uppercase and lowercase).

-Test cases: Hello, world, 123abc

2. Write a pattern to find all words that start with a vowel.
   -Test cases: apple, banana, orange, grape
3. Write a pattern to validate an email address.
   -Test cases: test@example.com, invalid-email
4. Write a pattern to extract all digits from a string.
   -Test cases: The price is 100 dollars and 50 cents.
5. Write a pattern to match a URL.
   -Test cases: https://www.example.com, ftp://example.com
6. Write a pattern to validate a US phone number format (e.g., (123) 456-7890)
   -Test cases: (123) 456-7890, 123-456-7890
7. Write a pattern to find a string that starts and ends with the same character.
   -Test cases: radar, hello, level
8. Write a pattern to validate a complex password.
   The password must contain at least one uppercase letter, one lowercase letter, one digit, one special character, and be at least 8 characters long.
   -Test cases: Password1!, pass123, PASSWORD!, Pass!
9. Write a regex pattern to identify and extract all instances of dates in the format dd-mm-yyyy or yyyy/mm/dd from a given text.
   The pattern should handle both formats in a single regex.
   -Test cases: Important dates: 12-05-2023, 2023/06/15, and 01-01-2024.
10. Create a regex pattern that matches a valid IPv4 address.
    The address must consist of four octets separated by dots, where each octet is a number between 0 and 255.
    -Test cases: 192.168.0.1, 256.100.50.25, 172.16.254.1

Solution (3 marks each)

```
import re

1. pattern = r'^[a-zA-Z]+$'
test_cases = ['Hello', 'world', '123abc']
results = [bool(re.match(pattern, case)) for case in test_cases]
# Output: [True, True, False]

2. pattern = r'\b[aeiouAEIOU]\w*'
test_string = 'apple banana orange grape'
results = re.findall(pattern, test_string)
# Output: ['apple', 'orange']

3. pattern = r'^[\w\.-]+@[\w\.-]+\.\w+$'
test_cases = ['test@example.com', 'invalid-email']
results = [bool(re.match(pattern, case)) for case in test_cases]
# Output: [True, False]
```

`[a-zA-Z]`：匹配任意大小写英文字母。

`+`：表示前面的字符组（`[a-zA-Z]`）可以出现一次或多次。

`\b`：匹配单词边界，确保匹配的是完整单词。

`\w*`：匹配0个或多个字母、数字或下划线。

4. pattern = r'\d+'
test_string = 'The price is 100 dollars and 50 cents.'
results = re.findall(pattern, test_string)
# Output: ['100', '50']

表示匹配前面的字符或字符组（`[^\s]`）零次或多次。

5. pattern = r'^(https?|ftp)://[^\s/$.?#].[^\s]*$'
test_cases = ['https://www.example.com', 'ftp://example.com', 'invalid-url']
results = [bool(re.match(pattern, case)) for case in test_cases]
# Output: [True, True, False]

确保 URL 的域名（或路径）以一个有效字符开头，排除空白字符和特殊符号 /, $, ., ?, #。

6. pattern = r'^\(\d{3}\) \d{3}-\d{4}$'
test_cases = ['(123) 456-7890', '123-456-7890']
results = [bool(re.match(pattern, case)) for case in test_cases]
# Output: [True, False]

`(.)`：

7. pattern = r'^(.).*\1$'
test_cases = ['radar', 'hello', 'level']

捕获组，匹配任意单个字符。
圆括号 `()` 表示捕获组，捕获的字符可以在后续通过 `\1` 引用。

results = [bool(re.match(pattern, case)) for case in test_cases]
# Output: [True, False, True]

8.pattern = r'^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[!@#$%^&*])[A-Za-z\d!@#$%^&*]{8,}$'
test_cases = ['Password1!', 'pass123', 'PASSWORD!', 'Pass!']
results = [bool(re.match(pattern, case)) for case in test_cases]
# Output: [True, False, False, False]

在正则表达式中，前瞻（`(?=...)`）用于验证某些条件是否存在，但它本身不消耗字符，也就是说，前瞻只是在检查条件，验证这些字符是否在字符串中，而不实际匹配和捕获这些字符。

9.pattern = r'\b(\d{2}-\d{2}-\d{4}|\d{4}/\d{2}/\d{2})\b'
test_string = 'Important dates: 12-05-2023, 2023/06/15, and 01-01-2024.'
results = re.findall(pattern, test_string)
# Output: ['12-05-2023', '2023/06/15', '01-01-2024']

10.pattern = r'^(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)
\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$'
test_cases = ['192.168.0.1', '256.100.50.25', '172.16.254.1']
results = [bool(re.match(pattern, case)) for case in test_cases]
# Output: [True, False, True]

Question 3 - Data Processing (20 marks)
The source files of Workshop on Statistical Machine Translation (WMT) are usually xml files. Before we
train a model using these data, we should convert them from XML formt to line-based text. Please solve t
questions:
1. Please convert the data in this file 1 to the line-based text with your own Python codes.
You need to remove all punctuation and convert all text to lowercase.
You should submit

your runnable codes and output file.
2. After you obtain the line-based text file, please create a BPE vocabulary (save each BPE token line by subword-nmt 2 . You should submit your runnable codes and output file.

Solution

A1. (10 marks)

(Answer from Joecfchan3)

1. **Crawling is not necessary and codes should output the target file**
2. **Check output file**

```
import requests
from bs4 import BeautifulSoup
import string

# Read the sample-src.xml file by navigating to the file raw text URL
url = "https://raw.githubusercontent.com/wmt-conference/wmt-format-tools/main/test/sample-data/sampl
req = requests.get(url)
```

req.status_code：获取 HTTP 请求的状态码。
200 表示请求成功，服务器返回了资源内容。

```
if req.status_code == 200:
    # Parse the data in BeautifulSoup object
    bs = BeautifulSoup(req.text, features="xml")
    # Find all <doc> tags in the data
    docs = bs.find_all('doc')

    # Create file with append mode
    with open('cs5481_a1_q3_xml_text.txt', 'w') as results:
        count = 0
        # Loop through all <doc> tags to retrieve the content
        for doc in docs:
            # Find all <seg> tags in the data
            segs = doc.src.p.find_all('seg')
            for seg in segs:
                # Clean and process the text
                cleaned_text = seg.text.translate(str.maketrans('', '', string.punctuation)).lower()
                results.write(cleaned_text + '\n')
                count += 1

    print('{} rows inserted to file'.format(count))
else:
    print('Failed to retrieve the data.')
```

A2. (10 marks)

(Answer from Joecfchan3)

1. **Check shell command (python version is also ok)**
2. **Check BPE file**

1https://github.com/wmt-conference/wmt-format-tools/tree/main/test/sample-data/sample-hyp.xml
2https://github.com/rsennrich/subword-nmt.git
5# *learn bpe by 'subword-nmt learn-bpe -s {num_operations} < {train_file} > {codes_file}"*
! subword-nmt learn-bpe -s 5000 < cs5481_a1_q3_xml_text.txt > cs5481_a1_q3_xml_text_bpe.bpe

Question 4 - Data Visualization (30 marks)
Data visualization is an effective method to overall evaluate the quality of the data. Generally, the conventional visualizations include column histogram/chart, pie chart, venn diagram, scatter plot, heatma
1. For a dataset of employee records containing employee ID (Integer; 1-500), department (Categorical; H sex (Binary; Male/Female), and years of experience (Integer; 0-40),
what visualization techniques would you recommend for analyzing these attributes?
2. Create a Python script to randomly generate 500 employee records based on the above criteria and visualize the data using your recommended techniques.
3.Calculate the number of employees per department and visualize the results using a bar chart.
4. Attention[1] is a classic and popular technique in natural language processing. Given two vectors Q ∈ and K ∈ R 10×15, the attention score of Q and K are calculated as:

$$\text{Attention\_Score}(\mathbf{Q}, \mathbf{K}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}),$$

where *dk* is the hidden dimension (15 in this case).
Please randomly initialize **Q** and **K** vectors and visualize the attention score via **heatmap**.
Reference [1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Polosukhi Attention is all you need. Advances in neural information processing systems.

Solution
A1. (5 marks)
Department Distribution & Sex: Pie chart or bar chart
years of Experience: Histogram or Scatter plot.

A2. (10 marks)
```
import random
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Randomly generate 500 employee records
N = 500
employee_ids = np.arange(1, N + 1)
departments = random.choices(['HR', 'IT', 'Sales'], k=N)
sex = random.choices(['Male', 'Female'], k=N)
```

```
experience = np.random.randint(0, 41, size=N)

# Create DataFrame
employees = {
    "employee_id": employee_ids,
    "department": departments,
    "sex": sex,
    "experience": experience
}
df = pd.DataFrame(employees)

# Department distribution pie chart
department_counts = df['department'].value_counts()
department_counts.plot(kind='pie', autopct='%1.1f%%')
plt.title("Department Distribution")
plt.ylabel("")
plt.show()

# Sex distribution pie chart
sex_counts = df['sex'].value_counts()
sex_counts.plot(kind='pie', autopct='%1.1f%%', colors=['lightblue', 'lightcoral'])
plt.title("Sex Distribution")
plt.ylabel("")
plt.show()

# Work experience scatter plot
plt.scatter(df['employee_id'], df['experience'], alpha=0.5, color='lightgreen')
plt.title("Work Experience Distribution")
plt.xlabel("Employee ID")
plt.ylabel("Years of Experience")
plt.show()
```

A3.(5 marks)

```
# Calculate number of employees per department
department_counts = df['department'].value_counts()

# Bar chart for number of employees per department
department_counts.plot(kind='bar', color='orange')
plt.title("Number of Employees per Department")
plt.xlabel("Department")
plt.ylabel("Number of Employees")
plt.show()
```

A4. (10 marks)

(Answer from Wanru HUANG)

```
import math
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.special import softmax
# Initialize Q & K, calculate attention score
q = np.random.rand(10, 15)
k = np.random.rand(10, 15)
attention = softmax(np.dot(q, k.T) / math.sqrt(15), axis=1)
# row
# Draw heatmap
sns.heatmap(attention, cmap="RdYlGn", center=0, annot=True)
plt.title("Attention score", fontsize=22)
plt.show()
```