INFO5731 Assignment 2

In this assignment, you will work on gathering text data from an open data source via web scraping or API. Following this, you will need to clean the text data and perform syntactic analysis on the data. Follow the instructions carefully and design well-structured Python programs to address each question.

Expectations:

- Use the provided *.ipynb* document to write your code & respond to the questions. Avoid generating a new file.
- Write complete answers and run all the cells before submission.
- Make sure the submission is "clean"; *i.e.*, no unnecessary code cells.
- Once finished, allow shared rights from top right corner (*see Canvas for details*).
- **Make sure to submit the cleaned data CSV in the comment section - 10 points**

Total points: 100

Deadline: Wednesday, at 11:59 PM.

Late Submission will have a penalty of 10% reduction for each day after the deadline.

Question 1 (40 points)

Write a python program to collect text data from **either of the following sources** and save the data into a **csv file:**

(1) Collect all the customer reviews of a product (you can choose any porduct) on amazon. [atleast 1000 reviews]

(2) Collect the top 1000 User Reviews of a movie recently in 2023 or 2024 (you can choose any movie) from IMDB. [If one movie doesn't have sufficient reviews, collect reviews of atleast 2 or 3 movies]

(3) Collect all the reviews of the top 1000 most popular software from G2 or Capterra.

(4) Collect the **abstracts** of the top 10000 research papers by using the query "machine learning", "data science", "artifical intelligence", or "information extraction" from Semantic Scholar.

(5) Collect all the information of the 904 narrators in the Densho Digital Repository.

```
# Your code here
'''
2) Collect the top 1000 User Reviews of a movie recently in 2023 or 2024 (you can choose
'''
#importing the required modules
import requests
from bs4 import BeautifulSoup
import pandas as pd
```

```
#creating function name with scrape_imdb_reviews
def scrape_imdb_reviews(number_of_reviews=1000):
    base_url = f'https://www.imdb.com/title/tt15398776/reviews?ref_=tt_urv'
    # creating an empty dictonary for reviews
    reviews = []
    page_num = 1
    while len(reviews) < number_of_reviews:
        url = f'{base_url}&start={page_num}'
        response = requests.get(url)
        soup = BeautifulSoup(response.content, 'html.parser')
        review_containers = soup.find_all('div', class_='text show-more__control')
        reviews.extend([container.text.strip() for container in review_containers])
        page_num += 25  # 25 reviews per the page
    return reviews[:number_of_reviews]
# Scraping top 1000 user reviews
Oppenheimer_reviews = scrape_imdb_reviews(number_of_reviews=1000)
```

```
# Creating a DataFrame
reviews_df = pd.DataFrame(Oppenheimer_reviews, columns=['reviews'])
reviews_df
```

	reviews
0	One of the most anticipated films of the year ...
1	You'll have to have your wits about you and yo...
2	I'm a big fan of Nolan's work so was really lo...
3	"Oppenheimer" is a biographical thriller film ...
4	This movie is just... wow! I don't think I hav...
...	...
995	It's isn't a masterpiece. It's a decent biopic...
996	I'm a big Nolan fan. Maybe this one just wasn'...
997	My Review - Oppenheimer\nMy Rating Ten plus 10...
998	Nolan is good at constructing complicated timi...
999	It saddens me that so many people are mistakin...

1000 rows × 1 columns

```
1 # Saving to the CSV file
2 reviews_df.to_csv('Oppenheimer_reviews.csv', index=False)
3 print(' Reviews are collected, and saved successfully ')
```

```
 Reviews are collected, and saved successfully
```

Question 2 (30 points)

Write a python program to **clean the text data** you collected in the previous question and save the clean data in a new column in the csv file. The data cleaning steps include: [Code and output is required for each part]

(1) Remove noise, such as special characters and punctuations.

(2) Remove numbers.

(3) Remove stopwords by using the stopwords list.

(4) Lowercase all texts

(5) Stemming.

(6) Lemmatization.

```
# Write code for each of the sub parts with proper comments.
#importing all the required modules
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
import string
import re

# Download NLTK resources
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

```
data_url="https://raw.githubusercontent.com/Sowmika26/INFO_5731/main/Assignment_2/Oppenh
df = pd.read_table(data_url,names=['text'])
df
```

	text
0	reviews
1	One of the most anticipated films of the year ...
2	You'll have to have your wits about you and yo...
3	I'm a big fan of Nolan's work so was really lo...
4	"Oppenheimer" is a biographical thriller film ...
...	...
996	It's isn't a masterpiece. It's a decent biopic...
997	I'm a big Nolan fan. Maybe this one just wasn'...
998	My Review - Oppenheimer\nMy Rating Ten plus 10...
999	Nolan is good at constructing complicated timi...
1000	It saddens me that so many people are mistakin...

1001 rows × 1 columns

```
#1: Remove Noise function (special characters and punctuations)
def remove_noise(text):
    clean_text = re.sub('[^a-zA-Z0-9]', ' ', text)
    return clean_text
df['clean_text'] = df['text'].apply(remove_noise)
# Displaying the Data Frame after removing noise
print("\nData Frame after removing noise:")
df
```

```
Data Frame after removing noise:
```

	text	clean_text
0	reviews	reviews
1	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...
2	You'll have to have your wits about you and yo...	You ll have to have your wits about you and yo...
3	I'm a big fan of Nolan's work so was really lo...	I m a big fan of Nolan s work so was really lo...
4	"Oppenheimer" is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...
...	...	...
996	It's isn't a masterpiece. It's a decent biopic...	It s isn t a masterpiece It s a decent biopic...
997	I'm a big Nolan fan. Maybe this one just wasn'...	I m a big Nolan fan Maybe this one just wasn ...
998	My Review - Oppenheimer\nMy Rating Ten plus 10...	My Review Oppenheimer My Rating Ten plus 10 ...
		Nolan is good at constructing complicated

```
#2:- Remove Numbers
#creating remove_number function
def remove_numbers(text):
    clean_text = re.sub(r'\d+', '', text)
    return clean_text
df['clean_text_remove_numbers'] = df['clean_text'].apply(remove_numbers)
print("\nData Frame after removing numbers:")
df
```

```
Data Frame after removing numbers:
```

	text	clean_text	clean_text_remove_numbers
0	reviews	reviews	reviews
1	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...
2	You'll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	You ll have to have your wits about you and yo...
3	I'm a big fan of Nolan's work so was really lo...	I m a big fan of Nolan s work so was really lo...	I m a big fan of Nolan s work so was really lo...
4	"Oppenheimer" is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...
...	...	...	...
996	It's isn't a masterpiece. It's a decent biopic...	It s isn t a masterpiece It s a decent biopic...	It s isn t a masterpiece It s a decent biopic...
997	I'm a big Nolan fan. Maybe this one just wasn'...	I m a big Nolan fan Maybe this one just wasn ...	I m a big Nolan fan Maybe this one just wasn ...
998	My Review - Oppenheimer\nMy Rating Ten plus 10...	My Review Oppenheimer My Rating Ten plus 10 ...	My Review Oppenheimer My Rating Ten plus I...
999	Nolan is good at constructing complicated timi...	Nolan is good at constructing complicated timi...	Nolan is good at constructing complicated timi...

```
# 3: Remove stopwords by using the stopwords List
#creating remove_stopwords function
def remove_stopwords(text):
    stop_words = set(stopwords.words('english'))
    words = nltk.word_tokenize(text)
    filter_words = [word for word in words if word.lower() not in stop_words]
    return ' '.join(filter_words)
df['clean_text_remove_stopwords'] = df['clean_text_remove_numbers'].apply(remove_stopwor
print("\nData Frame after removing stopwords without lowercase:")
df
```

```
Data Frame after removing stopwords without lowercase:
```

	text	clean_text	clean_text_remove_numbers	clean_text_remove_stopwo
0	reviews	reviews	reviews	revi
1	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One anticipated films year m people inclu
2	You'll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	wits brain fully switched watcl Oppenheim
3	I'm a big fan of Nolan's work so was really lo...	I m a big fan of Nolan s work so was really lo...	I m a big fan of Nolan s work so was really lo...	big fan Nolan work really loo forward un
4	"Oppenheimer" is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer biographical thi film writt
...	...	...	...	
996	It's isn't a masterpiece. It's a decent biopic...	It s isn t a masterpiece It s a decent biopic...	It s isn t a masterpiece It s a decent biopic...	masterpiece decent bi interesting person
997	I'm a big Nolan fan. Maybe this one just wasn'...	I m a big Nolan fan Maybe this one just wasn ...	I m a big Nolan fan Maybe this one just wasn ...	big Nolan fan Maybe one m promoted stor
998	My Review - Oppenheimer\nMy Rating Ten plus 10...	My Review Oppenheimer My Rating Ten plus 10 ...	My Review Oppenheimer My Rating Ten plus I...	Review Oppenheimer Rating plus Cinemas nc

```
1 # 4: Lowercase all texts
2 df['clean_text_lowercase'] = df['clean_text_remove_stopwords'].apply(lambda x: x.lower()
3 print("\nData Frame after converting texts to lowercase:")
4 df
```

Data Frame after converting texts to lowercase:

	text	clean_text	clean_text_remove_numbers	clean_text_remove_sto
0	reviews	reviews	reviews	
1	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One anticipated films ye people i
2	You'll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	wits brain fully switched v Oppenl
3	I'm a big fan of Nolan's work so was really lo...	I m a big fan of Nolan s work so was really lo...	I m a big fan of Nolan s work so was really lo...	big fan Nolan work really forwar
4	"Oppenheimer" is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer biographic film
...	...	...	...	
996	It's isn't a masterpiece. It's a decent biopic...	It s isn t a masterpiece It s a decent biopic...	It s isn t a masterpiece It s a decent biopic...	masterpiece dece interesting pe
997	I'm a big Nolan fan. Maybe this one just wasn'...	I m a big Nolan fan Maybe this one just wasn ...	I m a big Nolan fan Maybe this one just wasn ...	big Nolan fan Maybe on promoted
998	My Review - Oppenheimer\nMy Rating Ten plus 10...	My Review Oppenheimer My Rating Ten plus 10 ...	My Review Oppenheimer My Rating Ten plus I...	Review Oppenheimer Ra plus Cinema
999	Nolan is good at constructing complicated timi...	Nolan is good at constructing complicated timi...	Nolan is good at constructing complicated timi...	Nolan good cons complicated timi
1000	It saddens me that so many people are mistakin...	It saddens me that so many people are mistakin...	It saddens me that so many people are mistakin...	saddens many people m bigger lou

```
#5. Stemming
stemmer = PorterStemmer()
def apply_stemming(text):
    words = nltk.word_tokenize(text)
    stemmed_words = [stemmer.stem(word) for word in words]
    return ' '.join(stemmed_words)
df['clean_text_stemmed'] = df['clean_text_lowercase'].apply(apply_stemming)
print("\nData Frame after applying stemming:")
df
```

```
Data Frame after applying stemming:
```

	text	clean_text	clean_text_remove_numbers	clean_text_remove_sto
0	reviews	reviews	reviews	
1	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One anticipated films ye people i
2	You'll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	wits brain fully switched v Oppenl
3	I'm a big fan of Nolan's work so was really lo...	I m a big fan of Nolan s work so was really lo...	I m a big fan of Nolan s work so was really lo...	big fan Nolan work really forwar
4	"Oppenheimer" is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer biographic film
...	...	...	...	
996	It's isn't a masterpiece. It's a decent biopic...	It s isn t a masterpiece It s a decent biopic...	It s isn t a masterpiece It s a decent biopic...	masterpiece dece interesting pe
997	I'm a big Nolan fan. Maybe this one just wasn'...	I m a big Nolan fan Maybe this one just wasn ...	I m a big Nolan fan Maybe this one just wasn ...	big Nolan fan Maybe on promoted
998	My Review - Oppenheimer\nMy Rating Ten plus 10...	My Review Oppenheimer My Rating Ten plus 10 ...	My Review Oppenheimer My Rating Ten plus I...	Review Oppenheimer Ra plus Cinema
999	Nolan is good at constructing complicated timi...	Nolan is good at constructing complicated timi...	Nolan is good at constructing complicated timi...	Nolan good cons complicated timi
1000	It saddens me that so many people are mistakin...	It saddens me that so many people are mistakin...	It saddens me that so many people are mistakin...	saddens many people m bigger lou

```
#6. Lemmatization
lemmatizer = WordNetLemmatizer()
def apply_lemmatization(text):
    words = nltk.word_tokenize(text)
    lemmatized_words = [lemmatizer.lemmatize(word) for word in words]
    return ' '.join(lemmatized_words)
df['clean_text_lemmatized'] = df['clean_text_stemmed'].apply(apply_lemmatization)
print("\nData Frame after applying lemmatization:")
df
```

Data Frame after applying lemmatization:

	text	clean_text	clean_text_remove_numbers	clean_text_remove_sto
0	reviews	reviews	reviews	
1	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One of the most anticipated films of the year ...	One anticipated films ye people i
2	You'll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	You ll have to have your wits about you and yo...	wits brain fully switched v Oppenh
3	I'm a big fan of Nolan's work so was really lo...	I m a big fan of Nolan s work so was really lo...	I m a big fan of Nolan s work so was really lo...	big fan Nolan work really forwar
4	"Oppenheimer" is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer is a biographical thriller film ...	Oppenheimer biographic film v
...	...	...	...	
996	It's isn't a masterpiece. It's a decent biopic...	It s isn t a masterpiece It s a decent biopic...	It s isn t a masterpiece It s a decent biopic...	masterpiece dece interesting pe
997	I'm a big Nolan fan. Maybe this one just wasn'...	I m a big Nolan fan Maybe this one just wasn ...	I m a big Nolan fan Maybe this one just wasn ...	big Nolan fan Maybe on promoted
998	My Review - Oppenheimer\nMy Rating Ten plus 10...	My Review Oppenheimer My Rating Ten plus 10 ...	My Review Oppenheimer My Rating Ten plus I...	Review Oppenheimer Ra plus Cinema
999	Nolan is good at constructing complicated timi...	Nolan is good at constructing complicated timi...	Nolan is good at constructing complicated timi...	Nolan good cons complicated timi
1000	It saddens me that so many people are mistakin...	It saddens me that so many people are mistakin...	It saddens me that so many people are mistakin...	saddens many people m bigger lou

```
# Save the cleaned data to a new CSV file
df.to_csv('cleaned_data_.csv', index=False)
print("\nCleaned data saved ")
```

```
Cleaned data saved
```

Question 3 (30 points)

Write a python program to **conduct syntax and structure analysis of the clean text** you just saved above. The syntax and structure analysis includes:

(1) **Parts of Speech (POS) Tagging:** Tag Parts of Speech of each word in the text, and calculate the total number of N(oun), V(erb), Adj(ective), Adv(erb), respectively.

(2) **Constituency Parsing and Dependency Parsing:** print out the constituency parsing trees and dependency parsing trees of all the sentences. Using one sentence as an example to explain your understanding about the constituency parsing tree and dependency parsing tree.

(3) **Named Entity Recognition:** Extract all the entities such as person names, organizations, locations, product names, and date from the clean texts, calculate the count of each entity.

```
import pandas as pd
import nltk
from collections import Counter
# Download NLTK resources
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
True
```

```
# Read the data from cleaned_data_.csv
df = pd.read_csv('cleaned_data_.csv')

# Part 1: Parts of Speech (POS) Tagging
def pos_tagging(text):
    tokens = nltk.word_tokenize(text)
    pos_tags = nltk.pos_tag(tokens)
    return pos_tags
# Iterate through each row and print the POS tagging on a new line
for index, row in df.iterrows():
    pos_tags = pos_tagging(row['clean_text'])
    print(f"POS tagging for row {index + 1}:\n{pos_tags}\n")
    noun_count = verb_count = adj_count = adv_count = 0
    for _, pos in pos_tags:
        if pos.startswith('N'):
            noun_count += 1
        elif pos.startswith('V'):
            verb_count += 1
        elif pos.startswith('JJ'):
            adj_count += 1
        elif pos.startswith('RB'):
            adv_count += 1
    print(f"Total Nouns: {noun_count}")
    print(f"Total Verbs: {verb_count}")
    print(f"Total Adjectives: {adj_count}")
    print(f"Total Adverbs: {adv_count}")
```

```
Streaming output truncated to the last 5000 lines.
Total Adjectives: 9
Total Adverbs: 11
POS tagging for row 169:[('At', 'IN'), ('its', 'PRP$'), ('core', 'NN'), ('Oppenheime

Total Nouns: 125
Total Verbs: 66
Total Adjectives: 47
Total Adverbs: 22
POS tagging for row 170:[('I', 'PRP'), ('just', 'RB'), ('got', 'VBD'), ('out', 'IN')

Total Nouns: 87
Total Verbs: 58
Total Adjectives: 20
Total Adverbs: 22
POS tagging for row 171:[('Authentic', 'JJ'), ('audio', 'NN'), ('visual', 'JJ'), ('j

Total Nouns: 36
Total Verbs: 8
Total Adjectives: 14
Total Adverbs: 4
POS tagging for row 172:[('It', 'PRP'), ('s', 'VBZ'), ('isn', 'JJ'), ('t', 'IN'), ('

Total Nouns: 26
Total Verbs: 22
Total Adjectives: 15
Total Adverbs: 12
```

```
POS tagging for row 173:[('I', 'PRP'), ('m', 'VBP'), ('a', 'DT'), ('big', 'JJ'), ('N

Total Nouns: 147
Total Verbs: 103
Total Adjectives: 51
Total Adverbs: 47
POS tagging for row 174:[('My', 'PRP$'), ('Review', 'NNP'), ('Oppenheimer', 'NNP'),

Total Nouns: 220
Total Verbs: 94
Total Adjectives: 52
Total Adverbs: 26
POS tagging for row 175:[('Nolan', 'NN'), ('is', 'VBZ'), ('good', 'JJ'), ('at', 'IN'

Total Nouns: 76
Total Verbs: 32
Total Adjectives: 31
Total Adverbs: 24
POS tagging for row 176:[('It', 'PRP'), ('saddens', 'VBZ'), ('me', 'PRP'), ('that',

Total Nouns: 65
Total Verbs: 40
Total Adjectives: 35
Total Adverbs: 17
POS tagging for row 177:[('One', 'CD'), ('of', 'IN'), ('the', 'DT'), ('most', 'RBS')

Total Nouns: 68
Total Verbs: 77
Total Adjectives: 42
Total Adverbs: 32
```

```
1 # Install the required modules
2 !pip install benepar
3 !pip install tensorflow
4 !pip install tensorflow==2.8.0
```

```
# Downloading the required models
import benepar
import spacy.cli
benepar.download('benepar_en3')
spacy.cli.download("en_core_web_sm")

# Importing the libraries
import sys
import spacy
from spacy import displacy
parser = benepar.Parser("benepar_en3")
nlp = spacy.load('en_core_web_sm')
options = {'compact': True, 'font': 'Arial black', 'distance': 100}
# Parsing all the sentences in clean_text
for sentence in df['clean_text']:
    try:
        tree = parser.parse(sentence)
        print(tree)
    except:
        print("No Parse Tree")
        continue
# Printing the parse trees using spacy module
for sentence in df['clean_text']:
    doc = nlp(sentence)
    displacy.render(doc, style='dep', options=options, jupyter=True)
```

```
Collecting benepar
  Downloading benepar-0.2.0.tar.gz (33 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: nltk>=3.2 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: spacy>=2.0.9 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: torch>=1.6.0 in /usr/local/lib/python3.10/dist-packages
Collecting torch-struct>=0.5 (from benepar)
  Downloading torch_struct-0.5-py3-none-any.whl (34 kB)
Requirement already satisfied: tokenizers>=0.9.4 in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: transformers[tokenizers,torch]>=4.2.2 in /usr/local/lib/
Requirement already satisfied: protobuf in /usr/local/lib/python3.10/dist-packages (fro
Requirement already satisfied: sentencepiece>=0.1.91 in /usr/local/lib/python3.10/dist
```

```
# (3) Named Entity Recognition
import en_core_web_sm
# Loading the spaCy English model
```