Info

Name: Jacqui Unciano **Date:** 1/26/24 **Assignment:** HW02

Directions

Create a notebook to convert the raw text into a data frame of tokens, just as we did with Persuasion.

You may use the notebook from the lab as your guide.

Specifically, make sure your complete these tasks:

1. Remove Gutenberg's front and back matter using the lines that indicate the start and end of the project.
2. Chunk by chapter, using the pattern of locating the headers in the data frame, assigning them numbers, forward-filling those numbers, and then grouping by number (and cleaning up).
3. Split resulting data frame into paragraphs using the regex provided.
4. Split resulting data frame into sentences using the regex provided.
5. Split resulting data frame into tokens using the regex provided.

Be sure to include the OHCO of Chapters, Paragraphs, and Sentences in your data frame's index.

In [1]:
```
import pandas as pd
import configparser

config = configparser.ConfigParser()
config.read('../../../env.ini')
data_home = config['DEFAULT']['data_home']
output_dir = config['DEFAULT']['output_dir']
```

In [2]:
```
text_file = f'{data_home}/pg161.txt'
csv_file = f'{output_dir}/austen-sense-and-sensibility.csv'
```

In [3]:
```
OHCO = ['book_id','chap_num', 'para_num', 'sent_num', 'token_num'] #index names
```

In [4]:
```
LINES = pd.DataFrame(open(text_file, 'r', encoding='utf-8-sig').readlines(), column
LINES.index.name = 'line_num'
LINES.line_str = LINES.line_str.str.replace(r'\n+', ' ', regex=True).str.strip()
```

In [5]:
```
LINES.head()
```

Out[5]:

line_num	line_str
0	The Project Gutenberg EBook of Sense and Sens...
1	
2	This eBook is for the use of anyone anywhere a...
3	almost no restrictions whatsoever. You may co...
4	re-use it under the terms of the Project Guten...

In [6]:
```
title = LINES.loc[0].line_str.replace('The Project Gutenberg EBook of ', '')
print(title)
```

```
Sense and Sensibility, by Jane Austen
```

Getting just the text

In [7]:
```
clip_pats = [
    r"\*\*\*\s*START OF (?:THE|THIS) PROJECT",
    r"\*\*\*\s*END OF (?:THE|THIS) PROJECT"
]
```

In [8]:
```
# finding where the pattern is
pat_a = LINES.line_str.str.match(clip_pats[0])
pat_b = LINES.line_str.str.match(clip_pats[1])
```

In [9]:
```
# finding the line where the pattern is
line_a = LINES.loc[pat_a].index[0] + 1
line_b = LINES.loc[pat_b].index[0] - 1
```

In [10]:
```
LINES._get_value(line_a-1, 'line_str'), LINES._get_value(line_b+1, 'line_str')
```

Out[10]:
```
('*** START OF THIS PROJECT GUTENBERG EBOOK SENSE AND SENSIBILITY ***',
 '*** END OF THIS PROJECT GUTENBERG EBOOK SENSE AND SENSIBILITY ***')
```

In [11]:
```
LINES = LINES.loc[line_a : line_b]
```

Chuck, assign, forward-fill, then group by chapter

In [12]:
```
chap_pat = r"^\s*(?:chapter|letter)\s+\d+"
chap_lines = LINES.line_str.str.match(chap_pat, case=False)
# LINES.loc[chap_lines]
```

In [13]:
```
LINES.loc[chap_lines, 'chap_num'] = [i+1 for i in range(LINES.loc[chap_lines].shape
LINES.loc[chap_lines].head()
```

Out[13]:

line_num	line_str	chap_num
42	CHAPTER 1	1.0
196	CHAPTER 2	2.0
399	CHAPTER 3	3.0
562	CHAPTER 4	4.0
757	CHAPTER 5	5.0

In [14]:
```
LINES.sample(10)
```

Out[14]:

line_num	line_str	chap_num
3921	Mr. Palmer's acting so simply, with good abili...	NaN
490	approbation inferior to love."	NaN
4257	sort of a woman she is?"	NaN
507	looked forward to their marriage as rapidly ap...	NaN
1757	This suspicion was given by some words which a...	NaN
9230	When she told Marianne what she had done, howe...	NaN
2542	red, her countenance was not uncheerful.	NaN
1072	his face was not handsome, his countenance was...	NaN
6825	fall? Had I remained in England, perhaps--but...	NaN
51	life, had a constant companion and housekeeper...	NaN

In [15]:
```
LINES.chap_num = LINES.chap_num.ffill()
LINES.sample(10)
```

Out[15]:

line_num	line_str	chap_num
6231	voice at that moment! Have you forgot the las...	29.0
9815	it will be best to keep it entirely concealed ...	41.0
12337	the world;--so I was very glad to give her fiv...	49.0
3022	"I wish," said Margaret, striking out a novel ...	17.0
8155		36.0
1980	brother-in-law of Colonel Brandon, without who...	12.0
11523	governed and my temper improved. They shall n...	46.0
3341	going away without a sigh--declared his time t...	19.0
7455	she put bank-notes into Fanny's hands to the a...	33.0
3142		17.0

In [16]:
```
LINES = LINES.dropna(subset=['chap_num'])
```

In [17]:
```
LINES.chap_num.isna().sum()
```

Out[17]: 0

In [18]:
```
LINES = LINES.loc[~chap_lines]
LINES.chap_num = LINES.chap_num.astype('int')
LINES.sample(10)
```

Out[18]:

line_num	line_str	chap_num
3840	not be interested in it, even if it were true,...	20
475	much of Mrs. Dashwood's attention; for she was...	3
6430	by herself. Is there nothing one can get to c...	30
4636	praise, not merely from Lucy's assertion, but ...	23
2271	and village, and, beyond them, of those fine b...	13
1843	park; as to a stable, the merest shed would be...	12
242	I could not do less than give it; at least I t...	2
6300	"To-morrow, Marianne!"	29
4683	The young ladies went, and Lady Middleton was ...	23
2867	questions, and distinguished Elinor by no mark...	16

In [19]:
```
OHCO[1:2]
```

Out[19]:
```
['chap_num']
```

In [20]:
```
CHAPS = LINES.groupby(OHCO[1:2]).line_str.apply(lambda x: '\n'.join(x))\
                                                .to_frame('chap_str')
CHAPS.head(10)
```

Out[20]:

chap_num	chap_str
1	\n\nThe family of Dashwood had long been settl...
2	\n\nMrs. John Dashwood now installed herself m...
3	\n\nMrs. Dashwood remained at Norland several ...
4	\n\n"What a pity it is, Elinor," said Marianne...
5	\n\nNo sooner was her answer dispatched, than ...
6	\n\nThe first part of their journey was perfor...
7	\n\nBarton Park was about half a mile from the...
8	\n\nMrs. Jennings was a widow with an ample jo...
9	\n\nThe Dashwoods were now settled at Barton w...
10	\n\nMarianne's preserver, as Margaret, with mo...

In [21]:
```
CHAPS.chap_str = CHAPS.chap_str.str.strip()
CHAPS.head()
```

Out[21]:

chap_num	chap_str
1	The family of Dashwood had long been settled i...
2	Mrs. John Dashwood now installed herself mistr...
3	Mrs. Dashwood remained at Norland several mont...
4	"What a pity it is, Elinor," said Marianne, "t...
5	No sooner was her answer dispatched, than Mrs....

Split df by paragraph

In [22]:
```
para_pat = r'\n\n+'
PARAS = CHAPS['chap_str'].str.split(para_pat, expand=True).stack()\
    .to_frame('para_str').sort_index()
PARAS.index.names = OHCO[1:3]
PARAS.head()
```

Out[22]:

		para_str
chap_num	para_num	
1	0	The family of Dashwood had long been settled i...
	1	By a former marriage, Mr. Henry Dashwood had o...
	2	The old gentleman died: his will was read, and...
	3	Mr. Dashwood's disappointment was, at first, s...
	4	His son was sent for as soon as his danger was...

In [23]:
```
PARAS['para_str'] = PARAS['para_str'].str.replace(r'\n', ' ', regex=True)
PARAS['para_str'] = PARAS['para_str'].str.strip()
PARAS = PARAS[~PARAS['para_str'].str.match(r'^\s*$')] # Remove empty paragraphs
```

In [24]:
```
PARAS.sample(10)
```

Out[24]:

		para_str
chap_num	para_num	
43	7	On the morning of the third day however, the g...
25	13	"If Elinor is frightened away by her dislike o...
40	3	"You judged from your knowledge of the Colonel...
34	39	"Dear, dear Elinor, don't mind them. Don't le...
8	11	"But he talked of flannel waistcoats," said Ma...
29	10	"Indeed, Ma'am," said Elinor, very seriously, ...
31	21	"You have probably entirely forgotten a conver...
37	17	"Four months!--Have you known of this four mon...
40	30	"Yes," continued Elinor, gathering more resolu...
49	10	Edward could only attempt an explanation by su...

Split df by sentence

In [25]:
```
# sent_pat = r'[.?!;:"]+', but excluding quotes
sent_pat = r'[.?!;:]+' # some mistakes: abreviations ex: Mr. and Mrs.
SENTS = PARAS['para_str'].str.split(sent_pat, expand=True).stack()\
    .to_frame('sent_str')
SENTS.index.names = OHCO[1:4]
SENTS = SENTS[~SENTS['sent_str'].str.match(r'^\s*$')] # Remove empty paragraphs
SENTS.sent_str = SENTS.sent_str.str.strip() # CRUCIAL TO REMOVE BLANK TOKENS
```

In [26]:
```
SENTS.head()
```

Out[26]:

			sent_str
chap_num	**para_num**	**sent_num**	
1	**0**	**0**	The family of Dashwood had long been settled i...
		1	Their estate was large, and their residence wa...
		2	The late owner of this estate was a single man...
		3	But her death, which happened ten years before...
		4	for to supply her loss, he invited and receive...

Split df by token

In [27]:
```
token_pat = r"[\s',-]+"
TOKENS = SENTS['sent_str'].str.split(token_pat, expand=True).stack()\
    .to_frame('token_str')
TOKENS.index.names = OHCO[1:5]
TOKENS['term_str'] = TOKENS.token_str.replace(r'[\W_]+', '', regex=True).str.lower(
```

In [28]:
```
TOKENS.sample(10)
```

Out[28]:

				token_str	term_str
chap_num	**para_num**	**sent_num**	**token_num**		
18	**29**	**2**	**6**	I	i
25	**15**	**4**	**18**	sources	sources
5	**5**	**9**	**18**	drift	drift
19	**38**	**1**	**16**	the	the
41	**13**	**1**	**1**	he	he
49	**8**	**3**	**5**	finished	finished
33	**51**	**4**	**6**	fashion	fashion
28	**13**	**2**	**6**	explanations	explanations
30	**36**	**2**	**4**	John	john
12	**6**	**3**	**24**	to	to

Once you have done this, combine both Persuasion and Sense and Sensibility into a single data frame with an appropriately modified OHCO list. In other words, make sure your index includes a new index level for the book. Use the attached CSV (austen-persuasion.csv Download austen-persuasion.csv) to get the Persuasion data and then import it into your notebook as a data frame.

In [29]:
```
csv_file = f"{data_home}/austen-persuasion.csv"
df = pd.read_csv(csv_file, index_col=OHCO[1:5])
```

In [30]:
```
df.head()
```

Out[30]:

				token_str	term_str
chap_num	**para_num**	**sent_num**	**token_num**		
1	**0**	**0**	**0**	Sir	sir
			1	Walter	walter
			2	Elliot	elliot
			3	of	of
			4	Kellynch	kellynch

In [31]:
```
TOKENS.head()
```

Out[31]:

				token_str	term_str
chap_num	**para_num**	**sent_num**	**token_num**		
1	**0**	**0**	**0**	The	the
			1	family	family
			2	of	of
			3	Dashwood	dashwood
			4	had	had

In [32]:
```
comb_df = pd.concat([TOKENS, df], keys=["Sense and Sensibility","Persuasion"])
comb_df.index.names = OHCO[:]
comb_df.head()
```

Out[32]:

					token_str	term_str
book_id	**chap_num**	**para_num**	**sent_num**	**token_num**		
Sense and Sensibility	**1**	**0**	**0**	**0**	The	the
				1	family	family
				2	of	of
				3	Dashwood	dashwood
				4	had	had

In [33]:
```
comb_df.tail()
```

Out[33]:

book_id	chap_num	para_num	sent_num	token_num	token_str	term_str
Persuasion	**24**	**13**	**0**	**6**	of	of
				7	Persuasion	persuasion
				8	by	by
				9	Jane	jane
				10	Austen	austen

From the combined data frame, extract a vocabulary, i.e. a data frame with term string as index, along with term frequency and term length as features.

In [34]:
```
# my_text = TextImporter(src_file=csv_file, ohco_pats=[('book',r"(\w+\s*)+",'m'),('
```

In [35]:
```
VOCAB = comb_df.term_str.value_counts().to_frame('n')
VOCAB.index.name = 'term_str'
VOCAB['term_len'] = VOCAB.index.str.len()
VOCAB['term_freq'] = VOCAB.n/VOCAB.n.sum()
VOCAB.head()
```

Out[35]:

term_str	n	term_len	term_freq
the	7436	3	0.035921
to	6924	2	0.033448
and	6290	3	0.030385
of	6145	2	0.029685
her	3747	3	0.018101

After you have done all this, answer the following questions by extracting features from the corpus.

1. How many raw tokens are in the combined data frame?
2. How many distinct terms are there in the combined data frame (i.e. how big is the vocabulary)?
3. How many more terms does the vocabulary of Sense and Sensibility have than that of Persuasion?
4. What is the average number of tokens, rounded to an integer, per chapter in the corpus?
5. What is the average number of tokens, rounded to an integer, per paragraph in the corpus?

Question 1

In [36]:
```
## the number of raw tokens in combined df is 207,007 tokens
sum(VOCAB.n)
```

Out[36]: 207007

Question 2

In [37]:
```
## the number of distinct tokens in combined df is 8237 tokens
len(VOCAB)
```

Out[37]: 8237

Question 3

In [38]:
```
VOCAB_gb = comb_df.groupby(OHCO[0]).term_str.value_counts().to_frame('n')
VOCAB_gb.index.names = ['book_id','term_str']
VOCAB_gb.sample(10)
```

Out[38]:

		n
book_id	**term_str**	
Persuasion	**exertion**	8
Sense and Sensibility	**posture**	3
	unavoidable	3
Persuasion	**suit**	9
	songs	2
Sense and Sensibility	**resolving**	3
Persuasion	**stagnation**	1
Sense and Sensibility	**families**	3
Persuasion	**fearing**	4
	converse	2

In [39]:
```
VOCAB_gb = VOCAB_gb.groupby('book_id').n.count().to_frame()
VOCAB_gb
```

Out[39]:

book_id	n
Persuasion	5759
Sense and Sensibility	6278

In [40]:
```
## SnS has 519 more terms than Per.
VOCAB_gb.n[1] - VOCAB_gb.n[0]
```

Out[40]: 519

Question 4

In [41]:
```
q4 = comb_df.groupby(OHCO[:2]).token_str.count().to_frame('token_count')
q4.sample(10)
```

Out[41]:

book_id	chap_num	token_count
Sense and Sensibility	**16**	2034
	3	1565
	43	3464
	35	2436
Persuasion	**22**	5999
	19	2430
Sense and Sensibility	**49**	4338
	20	2557
Persuasion	**12**	5645
Sense and Sensibility	**45**	2181

In [42]:
```
len(q4.index)
```

Out[42]: 74

In [43]:
```
## the average number of tokens per chapter is 2808 in the corpus
sum(q4.token_count)/len(q4.index)
```

Out[43]: 2807.945945945946

Question 5

In [44]:
```
q5 = comb_df.groupby(OHCO[:3]).token_str.count().to_frame('token_count')
q5.sample(10)
```

Out[44]:

			token_count
book_id	**chap_num**	**para_num**	
Sense and Sensibility	**44**	**34**	44
	12	**12**	99
	44	**17**	38
	49	**8**	116
Persuasion	**7**	**35**	98
Sense and Sensibility	**3**	**8**	16
Persuasion	**15**	**7**	95
	22	**37**	108
	21	**40**	4
Sense and Sensibility	**19**	**39**	145

In [45]:
```
## the average number of tokens per paragraph is 74 in the corpus
sum(q5.token_count)/len(q5.index)
```

Out[45]:
```
73.68368794326241
```