Final Project Step 2

```
Course:   DS 5001
Module:   Final
Date:     8 May 2022
Author:   Thomas McIntyre gem5cm@virginia.edu
Purpose:  This notebook will utlize SkLearn and the csvs created in step
1 to create the following csvs (BOW, DOC, DTM, VOCAB, TFIDF)
```

In [1]:
```
data_home = "data"
local_lib = "code"
OHCO = ['book_id', 'chap_num', 'para_num', 'sent_num', 'token_num']
SENTS = OHCO[:4]
PARAS = OHCO[:3]
CHAPS = OHCO[:2]
BOOKS = OHCO[:1]
```

In [2]:
```
ngram_range = (1,4)
n_terms = 5000
```

In [3]:
```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer, TfidfTran
import nltk

nltk_resources = [
    'tokenizers/punkt',
    'taggers/averaged_perceptron_tagger',
    'corpora/stopwords',
    'help/tagsets'
]

for rsc in nltk_resources:
    try:
        nltk.data.find(rsc)
    except IndexError:
        nltk.download(rsc)
```

In [4]:
```
LIB = pd.read_csv(f"{data_home}/LIB.csv").set_index(OHCO[:1])
LIB.head()
```

Out[4]:

book_id	book_title	book_file	chap_regex	book_length	n_cha
1	A Game of Thrones, by RR Martin	corpus/MARTIN_A_GAME_OF_THRONES-pg1.txt	[A-Z]+[A-Z]+[A-Z]+	294315	
2	A Clash of Kings, by RR Martin	corpus/MARTIN_A_CLASH_OF_KINGS-pg2.txt	[A-Z]+[A-Z]+[A-Z]+	324029	1

book_id	book_title	book_file	chap_regex	book_length	n_cha
3	A Storm of Swords, by RR Martin	corpus/MARTIN_A_STORM_OF_SWORDS-pg3.txt	[A-Z]+[A-Z]+[A-Z]+	417469	
4	The Fellowship of the Ring, by JRR Tolkien	corpus/TOLKIEN_THE_FELLOWSHIP_OF_THE_RING-pg4.txt	_Chapter	180888	
16	Peter Pan, by James M. Barrie	corpus/BARRIE_PETER_PAN-pg16.txt	((Chapter)\s+\D+)	47631	

In [5]:
```
CORPUS = pd.read_csv(f"{data_home}/CORPUS.csv").set_index(OHCO)
```

In [6]:
```
exec(open("code/methods.py").read())
DOC = gather_docs(CORPUS, 2)
DOC['n_tokens'] = DOC.doc_str.apply(lambda x: len(x.split()))
DOC.head()
```

Out[6]:

book_id	chap_num	doc_str	n_tokens
1	1	we should start back gared urged as the woods ...	3860
	2	the morning had dawned clear and cold with a c...	3037
	3	catelyn had never liked this godswood she had ...	2074
	4	her brother held the gown up for her inspectio...	4161
	5	the visitors poured through the castle gates i...	3801

In [7]:
```
count_engine = CountVectorizer(
    stop_words='english',
    ngram_range=ngram_range,
    max_features=n_terms)

X = count_engine.fit_transform(DOC.doc_str)

DTM = pd.DataFrame(X.toarray(),
                   columns=count_engine.get_feature_names(),
                   index=DOC.index)
DTM
```

Out[7]:

book_id	chap_num	abandoned	able	abode	abroad	abruptly	absence	absent	absolutely	absurd

book_id	chap_num	abandoned	able	abode	abroad	abruptly	absence	absent	absolutely	absurd
1	1	2	0	0	0	0	0	0	0	0
	2	0	0	0	0	0	0	0	0	0
	3	0	0	0	0	0	0	0	0	0
	4	0	1	0	0	0	0	0	0	1
	5	0	1	0	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...
26654	13	0	0	0	0	0	0	0	0	0
	14	0	1	0	0	0	0	1	0	0
	15	0	0	0	1	1	0	0	0	0
	16	0	2	0	0	0	0	0	0	0
	17	0	0	0	0	0	0	0	0	0

533 rows × 5000 columns

In [8]:
```
tfidf_engine = TfidfTransformer(norm='l2', use_idf=True)
X1 = tfidf_engine.fit_transform(DTM)
TFIDF = pd.DataFrame(X1.toarray(), columns=DTM.columns, index=DTM.index)
TFIDF
```

Out[8]:

book_id	chap_num	abandoned	able	abode	abroad	abruptly	absence	absent	absolutely	a
1	1	0.034811	0.000000	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.
	2	0.000000	0.000000	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.
	3	0.000000	0.000000	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.
	4	0.000000	0.006485	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.
	5	0.000000	0.008425	0.0	0.000000	0.000000	0.0	0.015641	0.0	0.
...	...	...	...	...	...	...	...	...	...	
26654	13	0.000000	0.000000	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.
	14	0.000000	0.013908	0.0	0.000000	0.000000	0.0	0.025818	0.0	0.
	15	0.000000	0.000000	0.0	0.014096	0.013007	0.0	0.000000	0.0	0.
	16	0.000000	0.020908	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.
	17	0.000000	0.000000	0.0	0.000000	0.000000	0.0	0.000000	0.0	0.

533 rows × 5000 columns

In [9]:
```
## Create Vocab
VOCAB = DTM.sum().to_frame('n')

## Add Stats
VOCAB['tfidf_mean'] = TFIDF.mean()
VOCAB['df'] = DTM[DTM > 0].count()
VOCAB['dfidf'] = VOCAB.df * np.log2(len(TFIDF)/VOCAB.df)

## Add part of speech, stop word, and other basic stats
VOCAB.index.name = 'term_str'
VOCAB['p'] = VOCAB.n / VOCAB.n.sum()
VOCAB['i'] = -np.log2(VOCAB.p)
VOCAB['max_pos'] =CORPUS[['term_str','pos']].value_counts().unstack(fill_value=0).idxma
TPM = CORPUS[['term_str','pos']].value_counts().unstack()
VOCAB['n_pos'] = TPM.count(1)
sw = pd.DataFrame(nltk.corpus.stopwords.words('english'), columns=['term_str'])
sw = sw.reset_index().set_index('term_str')
sw.columns = ['dummy']
sw.dummy = 1
VOCAB['stop'] = VOCAB.index.map(sw.dummy)
VOCAB['stop'] = VOCAB['stop'].fillna(0).astype('int')
```

In [10]:
```
VOCAB
```

Out[10]:

term_str	n	tfidf_mean	df	dfidf	p	i	max_pos	n_pos	stop
abandoned	81	0.001990	63	194.084843	0.000094	13.381541	VBN	7.0	0
able	294	0.004507	203	282.709129	0.000340	11.521719	JJ	4.0	0
abode	45	0.001393	39	147.130991	0.000052	14.229538	NN	4.0	0
abroad	49	0.001363	40	149.442545	0.000057	14.106681	RB	9.0	0
abruptly	54	0.001403	53	176.493777	0.000062	13.966504	RB	7.0	0
...	...	...	...	...	...	...	...	...	...
youth	200	0.003464	103	244.263593	0.000231	12.077535	NN	7.0	0
youthful	43	0.001027	27	116.183814	0.000050	14.295127	JJ	3.0	0
youve	223	0.004084	144	271.881608	0.000258	11.920491	NN	16.0	0
yunkai	40	0.000941	6	38.838175	0.000046	14.399463	NNP	4.0	0
æneas	55	0.001268	11	61.584161	0.000064	13.940032	NNP	7.0	0

5000 rows × 9 columns

In [11]:
```
BOW = DTM[DTM > 0].stack().to_frame('n').join(TFIDF[TFIDF > 0].stack().to_frame('tfidf'
BOW
```

Out[11]:

book_id	chap_num		n	tfidf
1	**1**	**abandoned**	2.0	0.034811

			n	tfidf
book_id	**chap_num**			
		accustomed	1.0	0.017493
		admitted	1.0	0.014089
		aemon	2.0	0.038254
		afraid	2.0	0.019660
...	**...**	**...**	...	...
26654	**17**	**years**	2.0	0.009484
		yes	15.0	0.065695
		yesterday	1.0	0.009803
		young	1.0	0.004212
		youre	1.0	0.006011

389575 rows × 2 columns

In [12]:
```
VOCAB.to_csv("data/VOCAB.csv")
CORPUS.to_csv("data/CORPUS.csv")
LIB.to_csv("data/LIB.csv")
DOC.to_csv("data/DOC.csv")
DTM.to_csv("data/DTM.csv")
TFIDF.to_csv("data/TFIDF.csv")
BOW.to_csv("data/BOW.csv")
```

In [ ]:
```

```