# 5.18 Code Exercise 5

Max Ryoo (hr2ee)

## Part 0

Set up

In [1]:
```
data_dir = 'HW_5_DATA/'
```

In [2]:
```
count_method = 'n' # 'c' or 'n' # n = n tokens, c = distinct token (term) count
tf_method = 'sum' # sum, max, log, double_norm, raw, binary
tf_norm_k = .5 # only used for double_norm
idf_method = 'standard' # standard, max, smooth
gradient_cmap = 'YlGnBu' # YlGn, GnBu, YlGnBu; For tables; see https://matplotl
```

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

In [4]:
```
bag = CHAPS
```

In [5]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import plotly_express as px
```

In [6]:
```
sns.set()
%matplotlib inline
```

In [7]:
```
TOKEN = pd.read_csv(data_dir + 'TOKEN.csv')
VOCAB = pd.read_csv(data_dir + 'VOCAB.csv')
```

## Part 1

Write a function that returns a TFIDF matrix, with the following arguments:

1. The tokens data frame to use.
2. The OHCO level to use, e.g. which "bag" to use.
3. The type of count to use (e.g. binary counts are regular counts).
4. The type of TF to use.
5. The type of IDF to use.

In [8]:
```
def tfidf_matrix(token, ohco_level, count_method, tf_method, idf_method):
    token = token.set_index(OHCO)
    vocab = VOCAB.set_index('term_id')
```

```
    ## Filter
    token = token[~token.term_str.isna()]
    vocab = vocab[~vocab.term_str.isna()]

    ## Add term_id to TOKEN table
    token['term_id'] = token.term_str.map(vocab.reset_index().set_index('term_s

    ## Add Max POS to VOCAB (incase of missing)
    vocab['pos_max'] = token.groupby(['term_id', 'pos']).pos.count().unstack().

    ## Add Term Rank to VOCAB
    if 'term_rank' not in vocab.columns:
        vocab = vocab.sort_values('n', ascending=False).reset_index()
        vocab.index.name = 'term_rank'
        vocab = vocab.reset_index()
        vocab = vocab.set_index('term_id')
        vocab['term_rank'] = vocab['term_rank'] + 1

    ## "bag" --> ohco_level
    BOW = token.groupby(ohco_level+['term_id']).term_id.count().to_frame().rena
    BOW['c'] = BOW.n.astype('bool').astype('int')

    DTCM = BOW[count_method].unstack().fillna(0).astype('int')

    ## tf_method from params
    if tf_method == 'sum':
        TF = DTCM.T / DTCM.T.sum()

    elif tf_method == 'max':
        TF = DTCM.T / DTCM.T.max()

    elif tf_method == 'log':
        TF = np.log10(1 + DTCM.T)

    elif tf_method == 'raw':
        TF = DTCM.T

    elif tf_method == 'double_norm':
        TF = DTCM.T / DTCM.T.max()
        TF = tf_norm_k + (1 - tf_norm_k) * TF[TF > 0]

    elif tf_method == 'binary':
        TF = DTCM.T.astype('bool').astype('int')

    TF = TF.T

    ## Compute DF
    DF = DTCM[DTCM > 0].count()

    ## Compute IDF
    N = DTCM.shape[0]

    ## idf_method from params
    if idf_method == 'standard':
        IDF = np.log10(N / DF)

    elif idf_method == 'max':
        IDF = np.log10(DF.max() / DF)
```

```
    elif idf_method == 'smooth':
        IDF = np.log10((1 + N) / (1 + DF)) + 1

    ## Compute TFIDF
    TFIDF = TF * IDF

    ## Move things to their places
    vocab['df'] = DF
    vocab['idf'] = IDF

    BOW['tf'] = TF.stack()
    BOW['tfidf'] = TFIDF.stack()

    ## Apply TFIDF sum to VOCAB
    VOCAB['tfidf_sum'] = TFIDF.sum()

    return VOCAB.sort_values('tfidf_sum', ascending=False).head(20).style.backg
```

## Part 2

Use this function to get the TFIDF of the collection with books as the bag. Answer the following questions:

1. What are the top 20 words in the corpus by TFIDF sum if you use the 'n' count method (i.e. not the binary count)?

In [9]:
```
tfidf_matrix(token = TOKEN, ohco_level=BOOKS ,count_method='n', tf_method='sum'
```

Out[9]:

| | term_id | term_str | n | num | stop | p_stem | tfidf_sum |
|---|---|---|---|---|---|---|---|
| **26302** | 26302 | pierre | 1525 | 0 | 0 | pierr | 0.009838 |
| **11648** | 11648 | elinor | 623 | 0 | 0 | elinor | 0.006763 |
| **19306** | 19306 | israel | 519 | 0 | 0 | israel | 0.006504 |
| **38673** | 38673 | vernon | 104 | 0 | 0 | vernon | 0.005857 |
| **2644** | 2644 | babbalanja | 547 | 0 | 0 | babbalanja | 0.005394 |
| **22176** | 22176 | media | 497 | 0 | 0 | media | 0.004940 |
| **5540** | 5540 | catherine | 557 | 0 | 0 | catherin | 0.004324 |
| **21823** | 21823 | marianne | 499 | 0 | 0 | mariann | 0.004316 |
| **29073** | 29073 | reginald | 74 | 0 | 0 | reginald | 0.004167 |
| **11812** | 11812 | emma | 787 | 0 | 0 | emma | 0.004055 |
| **14479** | 14479 | frederica | 72 | 0 | 0 | frederica | 0.004055 |
| **8300** | 8300 | crawford | 493 | 0 | 0 | crawford | 0.004000 |
| **8901** | 8901 | darcy | 374 | 0 | 0 | darci | 0.003986 |
| **11663** | 11663 | elliot | 254 | 0 | 0 | elliot | 0.003953 |
| **13174** | 13174 | fanny | 865 | 0 | 0 | fanni | 0.003898 |
| **35769** | 35769 | tilney | 196 | 0 | 0 | tilney | 0.003290 |
| **39528** | 39528 | weston | 389 | 0 | 0 | weston | 0.003146 |
| **40380** | 40380 | yoomy | 308 | 0 | 0 | yoomi | 0.003049 |
| **39506** | 39506 | wentworth | 191 | 0 | 0 | wentworth | 0.002972 |
| **22872** | 22872 | mohi | 301 | 0 | 0 | mohi | 0.002972 |

1. How do this words compare to the top 20 words when we use chapter as the bag? Are they the same? If different, can characterize how they are different in terms of part-of-speech?

In [10]:
```
tfidf_matrix(token = TOKEN, ohco_level=CHAPS ,count_method='n', tf_method='sum'
```

Out[10]:

| | term_id | term_str | n | num | stop | p_stem | tfidf_sum |
|---|---|---|---|---|---|---|---|
| **31648** | 31648 | she | 12153 | 0 | 1 | she | 1.349173 |
| **16730** | 16730 | her | 17020 | 0 | 1 | her | 1.331294 |
| **26302** | 26302 | pierre | 1525 | 0 | 0 | pierr | 1.154340 |
| **40387** | 40387 | you | 14466 | 0 | 1 | you | 0.756695 |
| **23260** | 23260 | mr | 3420 | 0 | 0 | mr | 0.706684 |
| **17566** | 17566 | i | 27810 | 0 | 1 | i | 0.664376 |
| **39540** | 39540 | whale | 1180 | 0 | 0 | whale | 0.594694 |
| **23261** | 23261 | mrs | 2664 | 0 | 0 | mr | 0.593591 |
| **35574** | 35574 | thou | 916 | 0 | 0 | thou | 0.586061 |
| **36885** | 36885 | um | 12 | 0 | 0 | um | 0.501435 |
| **2644** | 2644 | babbalanja | 547 | 0 | 0 | babbalanja | 0.498785 |
| **22107** | 22107 | me | 7654 | 0 | 1 | me | 0.489280 |
| **35417** | 35417 | thee | 662 | 0 | 0 | thee | 0.473432 |
| **23450** | 23450 | my | 10644 | 0 | 1 | my | 0.465872 |
| **22176** | 22176 | media | 497 | 0 | 0 | media | 0.464082 |
| **19278** | 19278 | isabel | 399 | 0 | 0 | isabel | 0.462332 |
| **40399** | 40399 | your | 4111 | 0 | 1 | your | 0.461802 |
| **32157** | 32157 | sir | 1845 | 0 | 0 | sir | 0.458684 |
| **22714** | 22714 | miss | 1987 | 0 | 0 | miss | 0.457160 |
| **5293** | 5293 | captain | 1508 | 0 | 0 | captain | 0.442218 |

When using the book and chapter as the bag, we can quickly see that the top 20 worlds are completely different. There are actually no overlaps and even from the tfidf_sum we can see that the range of values for the top 20 words for each bag is completely different as well.

In [ ]: