Project Goal

The goal of this project is to take thousands of Amazon product reviews and segment to only include reviews on Nike products. Based off this segmentation we want to be able to perform topic modeling and data clustering to gain valuable product and marketing insights.

Part 1: Importing and Extracting the Data

```
import pickle
import json
from time import sleep
import gzip
import itertools
import pandas as pd
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

We are first going to load the files in as a typical python dictionary for both files. For this first one, we will filter to only use nike related reviews.

```
##this assigns the filename we're trying to load in to a string variable
working_directory = 'drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project'
working_file = '%s/meta_Clothing_Shoes_and_Jewelry.jsonl.gz' % working_directory
loadedjson = open(working_file, 'r')
```

```
asins = []
with gzip.open("drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project/meta_Clotl
    for product in products:
        data = json.loads(product)
        categories = [c.lower() for c in
                      list(itertools.chain(*data.get("categories", [])))]
        if "nike" in categories:
            asins.append(data["asin"])
```

The correct length is 8327, to give us a sanity check.

```
len(asins)

    8327
```

```
outputfile = open('%s/allasins.txt' % working_directory, 'w')

outputfile.write(','.join(asins))
outputfile.close()
```

We will now load the "reviews" file in and combine it with our asins nike file, to only include nike reviews.

```
!wget http://128.138.93.164/reviews_Clothing_Shoes_and_Jewelry.json.gz -P /content/dri

    --2022-05-29 13:32:12--  http://128.138.93.164/reviews_Clothing_Shoes_and_Jewelr
    Connecting to 128.138.93.164:80... connected.
    HTTP request sent, awaiting response... 200 OK
    Length: 888065454 (847M) [application/octet-stream]
    Saving to: '/content/drive/MyDrive/MSDS_marketing_text_analytics/master_files/2_

    reviews_Clothing_Sh 100%[===================>] 846.92M  26.1MB/s    in 22s

    2022-05-29 13:32:35 (37.7 MB/s) - '/content/drive/MyDrive/MSDS_marketing_text_an
```

```
!gzip -d /content/drive/MyDrive/MSDS_marketing_text_analytics/master_files/2_topic_mo
```

```
##this assigns the filename we're trying to load in to a string variable
working_directory = '/content/drive/MyDrive/MSDS_marketing_text_analytics/master_file
working_file = '%s/reviews_Clothing_Shoes_and_Jewelry.json' % working_directory
loadedjson = open(working_file, 'r')
```

```
#Step 2 - Parsing the review data
#Let's load the review data into a dictionary. It's the exact same process
#as loading the review data
count = 0
allreviews = {}
for aline in loadedjson:
   count += 1
   if count % 100000 == 0:
       print(count)
   areview = eval(aline)
#    #I'm arbitrarily using the iteration count as the key for the dictionary
#    #but you don't have to do this
   allreviews[count] = areview
```

```
#how many reviews do we have?
print(len(allreviews))
#
```

```
100000
200000
300000
400000
500000
600000
700000
800000
900000
1000000
1100000
1200000
1300000
1400000
1500000
1600000
1700000
1800000
1900000
2000000
2100000
2200000
2300000
2400000
2500000
2600000
2700000
2800000
2900000
3000000
3100000
3200000
3300000
3400000
3500000
3600000
3700000
3800000
3900000
4000000
4100000
4200000
4300000
4400000
4500000
4600000
4700000
4800000
4900000
5000000
5100000
5200000
```

```
5300000
5400000
5500000
5600000
5700000
5748920
```

```
nikereviews = {}
count = 0
for areview in allreviews:
   count += 1
   if count % 100000 == 0:
       print(count/5748920)
   #setting current review as a dictionary, so we can easily reference its
   #entries
   thereview = allreviews[areview]
   theasin = thereview['asin']
   reviewerid = thereview['reviewerID']
   if theasin in asins:
       #im setting the key here as something unique. if we just did by asin
       #we'd only have one review for each asin, with the last review the only
       #one being stored
       thekey = '%s.%s' % (theasin, reviewerid)
       nikereviews[thekey] = thereview


# #that's it! all Nike reviews are stored nikereviews!
# #how many Nike reviews do we have?
# print(len(nikereviews))

#let's save our data as a JSON dictionary
#json.dump(nikereviews, open('%s/allnikereviews.json' % working_directory, 'w'))
```

```
0.017394571502125616
0.03478914300425123
0.05218371450637685
0.06957828600850247
0.08697285751062808
0.1043674290127537
0.12176200051487931
0.13915657201700493
0.15655114351913055
0.17394571502125616
0.19134028652338178
0.2087348580255074
0.226129429527633
0.24352400102975863
0.2609185725318843
0.27831314403400986
0.2957077155361355
0.3131022870382611
0.33049685854038674
0.34789143004251233
```

```
0.365286001544638
0.38268057304676356
0.4000751445488892
0.4174697160510148
0.43486428755314044
0.452258859055266
0.46965343055739167
0.48704800205951726
0.5044425735616429
0.5218371450637685
0.5392317165658941
0.5566262880680197
0.5740208595701454
0.591415431072271
0.6088100025743965
0.6262045740765222
0.6435991455786478
0.6609937170807735
0.678388288582899
0.6957828600850247
0.7131774315871503
0.730572003089276
0.7479665745914015
0.7653611460935271
0.7827557175956528
0.8001502890977784
0.8175448605999039
0.8349394321020296
0.8523340036041552
0.8697285751062809
0.8871231466084064
0.904517718110532
0.9219122896126577
0.9393068611147833
0.9567014326169089
0.9740960041190345
0.9914905756211602
```

```python
working_directory = 'drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project'
#print(len(nikereviews))


#json.dump(nikereviews, open('%s/allnikereviews.json' % working_directory, 'w'))


json_path = "%s/allnikereviews.json" % working_directory


json_file = json.load(open(json_path, 'r'))

for a_review in json_file:
  the_review = json_file[a_review]


the_review
```

```
{'asin': 'B00L5K86LO',
 'helpful': [0, 0],
 'overall': 5.0,
 'reviewText': 'Love it !',
 'reviewTime': '07 20, 2014',
 'reviewerID': 'A1KBC812A7RSY9',
 'reviewerName': 'B',
 'summary': 'So CUTE',
 'unixReviewTime': 1405814400}
```

Preprocessing the Text

```
import os
import matplotlib


try:
  import tmtoolkit
except:
  !pip install tmtoolkit
  os.kill(os.getpid(), 9)
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-whe
Collecting tmtoolkit
  Downloading tmtoolkit-0.10.0-py3-none-any.whl (7.1 MB)
     |████████████████████████████████| 7.1 MB 9.4 MB/s
Requirement already satisfied: numpy<2,>=1.19.0 in /usr/local/lib/python3.7/dist
Collecting pandas<1.2,>=1.1.0
  Downloading pandas-1.1.5-cp37-cp37m-manylinux1_x86_64.whl (9.5 MB)
     |████████████████████████████████| 9.5 MB 50.3 MB/s
Collecting globre<0.2,>=0.1.5
  Downloading globre-0.1.5.tar.gz (20 kB)
Collecting spacy<2.4,>=2.3.0
  Downloading spacy-2.3.7-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.
     |████████████████████████████████| 10.4 MB 57.1 MB/s
Collecting xlrd>=1.2.0
  Downloading xlrd-2.0.1-py2.py3-none-any.whl (96 kB)
     |████████████████████████████████| 96 kB 5.9 MB/s
Collecting scipy<1.6,>=1.5.0
  Downloading scipy-1.5.4-cp37-cp37m-manylinux1_x86_64.whl (25.9 MB)
     |████████████████████████████████| 25.9 MB 1.6 MB/s
Collecting matplotlib<3.4,>=3.3.0
  Downloading matplotlib-3.3.4-cp37-cp37m-manylinux1_x86_64.whl (11.5 MB)
     |████████████████████████████████| 11.5 MB 58.0 MB/s
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in /usr/
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dis
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dis
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-pac
```

```
     Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-package
     Requirement already satisfied: srsly<1.1.0,>=1.0.2 in /usr/local/lib/python3.7/d
     Requirement already satisfied: blis<0.8.0,>=0.4.0 in /usr/local/lib/python3.7/di
     Collecting thinc<7.5.0,>=7.4.1
       Downloading thinc-7.4.5-cp37-cp37m-manylinux2014_x86_64.whl (1.0 MB)
          |████████████████████████████████| 1.0 MB 33.5 MB/s
     Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packa
     Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.7
     Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.7/d
     Requirement already satisfied: plac<1.2.0,>=0.9.6 in /usr/local/lib/python3.7/di
     Requirement already satisfied: wasabi<1.1.0,>=0.4.0 in /usr/local/lib/python3.7/
     Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.7/d
     Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3
     Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/pytho
     Requirement already satisfied: catalogue<1.1.0,>=0.0.7 in /usr/local/lib/python3
     Requirement already satisfied: importlib-metadata>=0.20 in /usr/local/lib/python
     Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packag
     Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/di
     Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dis
     Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/l
     Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-pac
     Building wheels for collected packages: globre
       Building wheel for globre (setup.py) ... done
       Created wheel for globre: filename=globre-0.1.5-py3-none-any.whl size=19546 sh
       Stored in directory: /root/.cache/pip/wheels/40/95/37/5303ce04fce53b6e64ed74a3
     Successfully built globre
     Installing collected packages: thinc, xlrd, spacy, scipy, pandas, matplotlib, gl
       Attempting uninstall: thinc
         Found existing installation: thinc 7.4.0
```

```
import nltk
import random
import numpy as np
from tmtoolkit.corpus import Corpus
import json
import pickle
import scipy.sparse
```

```
corpus = Corpus()
for i, a_review in enumerate(json_file):
  the_review = json_file[a_review]
  corpus.add_doc(str(i), the_review['reviewText'])
```

```
print(len(corpus))
```

```
     21570
```

```
from tmtoolkit.preprocess import TMPreproc

preproc = TMPreproc(corpus, language='en')
preproc.pos_tag()
preproc.lemmatize()
preproc.tokens_to_lowercase()
preproc.remove_special_chars_in_tokens()
preproc.add_stopwords(['http', 'nt'])
```

```
/usr/local/lib/python3.7/dist-packages/spacy/util.py:275: UserWarning: [W031] Mo
  warnings.warn(warn_msg)
<TMPreproc [21570 documents / en]>
```

```
preproc.tokens['91']
```

```
['these',
 'watch',
 'have',
 'buy',
 'for',
 'its',
 'girlfriend',
 '',
 'such',
 'accessory',
 'not',
 'to',
 'find',
 'in',
 'russia',
 '',
 'alas',
 'nike',
 'do',
 'not',
 'trade',
 '',
 'when',
 'she',
 'have',
 'get',
 'and',
 'parcel',
 'of',
 'land',
 '',
 'they',
 'seem',
 'very',
 'small',
 'and',
 'on',
 'baby',
 'hand',
 '',
 'but',
 'each',
 'woman',
 'pay',
 'attention',
 'to',
 'her',
 'hand',
 'and',
 'i',
 'caughte',
 'the',
 'glance',
 'of',
 'the',
 'man',
 'on',
 'her',
```

```
preproc.vocabulary_size
```

```
19202
```

```
preproc.tokens_datatable
```

	doc	position	token	lemma	pos	whitespace
0	0	0	the	the	DET	True
1	0	1	colour	colour	NOUN	True
2	0	2	i	i	PRON	True
3	0	3	receive	receive	VERB	True
4	0	4	be	be	AUX	True
...	...	...	...	...	...	...
58	9999	58	this	this	DET	True
59	9999	59	purchase	purchase	NOUN	True
60	9999	60	be	be	AUX	True
61	9999	61	a	a+	DET	False
62	9999	62		!	PUNCT	False

1125388 rows × 6 columns

```
preproc_smaller = preproc.copy()
preproc_smaller.filter_for_pos('N', 'V', 'ADJ')
preproc_smaller.clean_tokens(remove_numbers=True, remove_shorter_than=2)
preproc_smaller.remove_common_tokens(df_threshold=0.8)
preproc_smaller.remove_uncommon_tokens(df_threshold=0.01)

print(preproc.vocabulary_size)
print(preproc_smaller.vocabulary_size)
```

```
19202
202
```

```
preproc_smaller.tokens_datatable
```

	doc	position	token	lemma	pos	whitespace
0	0	0	receive	receive	VERB	True
1	0	1	blue	blue	ADJ	True
2	0	2	show	show	VERB	True
3	0	3	change	change	VERB	True
4	0	4	get	get	VERB	True
...	...	...	...	...	...	...
5	9999	5	find	find	VERB	True
6	9999	6	good	good	ADJ	True
7	9999	7	ship	ship	VERB	True
8	9999	8	quickly	quickly	ADV	True
9	9999	9	overall	overall	ADV	False

201599 rows × 6 columns

```
print(preproc.tokens['91'])
print(preproc_smaller.tokens['91'])
print(preproc.tokens['1'])
print(preproc_smaller.tokens['1'])
print(preproc.tokens['2000'])
print(preproc_smaller.tokens['2000'])
```

```
['these', 'watch', 'have', 'buy', 'for', 'its', 'girlfriend', '', 'such', 'access
['buy', 'find', 'get', 'seem', 'small', 'pay', 'perfect', 'stylish', 'excellent'
['very', 'cute', 'and', 'be', 'really', 'practical', '', 'fit', 'well', 'on', 'sm
['cute', 'really', 'fit', 'well', 'small', 'wear', 'really', 'love']
['it', 'be', 'a', 'really', 'great', 'shoe', '', 'it', 'fit', 'right', 'and', 'be
['really', 'great', 'fit', 'right', 'really', 'comfortable', 'highly', 'recommen
```

```
doc_labels = np.array(preproc.doc_labels)
doc_labels[:10]
```

```
array(['0', '1', '10', '100', '1000', '10000', '10001', '10002', '10003',
       '10004'], dtype='<U5')
```

```
vocab_bg = np.array(preproc.vocabulary)
vocab_sm = np.array(preproc_smaller.vocabulary)
```

```
dtm_bg = preproc.dtm
dtm_sm = preproc_smaller.dtm

dtm_bg, dtm_sm
```

```
    (<21570x19202 sparse matrix of type '<class 'numpy.int32'>'
            with 743562 stored elements in Compressed Sparse Row format>,
     <21570x202 sparse matrix of type '<class 'numpy.int32'>'
            with 179630 stored elements in Compressed Sparse Row format>)
```

```
pickle.dump(doc_labels, open('%s/doc_labels.p' % working_directory, 'wb'))

scipy.sparse.save_npz('%s/small_dtm.npz' % working_directory, dtm_sm)
scipy.sparse.save_npz('%s/big_dtm.npz' % working_directory, dtm_bg)

pickle.dump(vocab_bg, open('%s/big_vocab.p' % working_directory, 'wb'))
pickle.dump(vocab_sm, open('%s/small_vocab.p' % working_directory, 'wb'))

pickle.dump(corpus, open('%s/corpus.p' % working_directory, 'wb') )
```

Filter and Preprocess again using positve and negative reviews

```
working_directory = 'drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project'
json_path = "%s/allnikereviews.json" % working_directory
```

```
from time import sleep
json_file = json.load(open(json_path, 'r'))
```

```
df = pd.read_json('drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project/allnike
```

```
df.head()
```

```
positive = df[(df['overall'] == 5)]
```

```
pos = positive.reset_index(drop=True)
```

```
pos.head(10)
```

	reviewerID	asin	reviewerName	helpful	reviewText	overall	summ
0	A3BVWMS9I8OH8U	B0000V9K32	Tatiana A. Alencar	[0, 0]	Very cute and is really practical. Fits better...	5	Cute prac
1	A3F8O512N9UNVM	B0000V9K46	D. Pando "-d"	[0, 1]	This product came promptly and as described, p...	5	Ju descr
2	A1EDPEDXSQ78G4	B0000V9KRI	SO	[0, 0]	I love this watch, i use every day, every wher...	5	Love W
3	A2RBU58FQTO2MV	B0000V9KRS	Alejandra Martinez	[0, 0]	I totally love this watch. It is much nicer an	5	Lo

```
negative = df[(df['overall'] == 1)]
```

```
pos1 = pos.T
```

```
pos1
```

	0	1	2	
reviewerID	A3BVWMS9I8OH8U	A3F8O512N9UNVM	A1EDPEDXSQ78G4	A2RBU58FQTO2N
asin	B0000V9K32	B0000V9K46	B0000V9KRI	B0000V9KF
reviewerName	Tatiana A. Alencar	D. Pando "-d"	SO	Alejandra Martin
helpful	[0, 0]	[0, 1]	[0, 0]	[0,
	Very cute and is	This product came	I love this watch, i	I totally love th
reviewTime	12 26, 2009	03 26, 2008	03 7, 2010	01 15, 20

```
neg = negative.reset_index(drop=True)
```

```
b = neg.to_dict(orient = 'index')
```

```
len(b)
```

```
1243
```

```
json_file1 = a

for a_review in json_file1:
  the_review = json_file1[a_review]
```

```
the_review
```

```
{'asin': 'B00L5K86LO',
 'helpful': [0, 0],
 'overall': 5,
 'reviewText': 'Love it !',
 'reviewTime': '07 20, 2014',
 'reviewerID': 'A1KBC812A7RSY9',
 'reviewerName': 'B',
 'summary': 'So CUTE',
 'unixReviewTime': 1405814400}
```

```
a = pos.to_dict(orient = 'index')
```

```
json_file = a

for a_review in json_file:
  the_review = json_file[a_review]
```

```
the_review
```

```
{'asin': 'B00L5K86LO',
 'helpful': [0, 0],
 'overall': 5,
 'reviewText': 'Love it !',
 'reviewTime': '07 20, 2014',
 'reviewerID': 'A1KBC812A7RSY9',
 'reviewerName': 'B',
 'summary': 'So CUTE',
 'unixReviewTime': 1405814400}
```

```
a.keys()
```

```
    dict_keys([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
```

```
corpus1 = Corpus()
for i, a_review in enumerate(a):
  the_review = a[a_review]
  corpus1.add_doc(str(i), the_review['reviewText'])
```

```
print(len(corpus1))
```

```
    13799
```

```
from tmtoolkit.preprocess import TMPreproc

preproc = TMPreproc(corpus1, language='en')
preproc.pos_tag()
preproc.lemmatize()
preproc.tokens_to_lowercase()
preproc.remove_special_chars_in_tokens()
preproc.add_stopwords(['http', 'nt'])
```

```
    /usr/local/lib/python3.7/dist-packages/spacy/util.py:275: UserWarning: [W031] Mo
      warnings.warn(warn_msg)
    <TMPreproc [13799 documents / en]>
```

```
preproc_smaller = preproc.copy()
preproc_smaller.filter_for_pos('N', 'V', 'ADJ')
preproc_smaller.clean_tokens(remove_numbers=True, remove_shorter_than=2)
preproc_smaller.remove_common_tokens(df_threshold=0.8)
preproc_smaller.remove_uncommon_tokens(df_threshold=0.01)

print(preproc.vocabulary_size)
print(preproc_smaller.vocabulary_size)
```

```
    14135
    185
```

```
vocab_bg_pos = np.array(preproc.vocabulary)
vocab_sm_pos = np.array(preproc_smaller.vocabulary)
```

```
dtm_bg_pos = preproc.dtm
dtm_sm_pos = preproc_smaller.dtm

dtm_bg_pos, dtm_sm_pos
```

```
(<13799x14135 sparse matrix of type '<class 'numpy.int32'>'
        with 443032 stored elements in Compressed Sparse Row format>,
 <13799x185 sparse matrix of type '<class 'numpy.int32'>'
        with 107796 stored elements in Compressed Sparse Row format>)
```

```
doc_labels = np.array(preproc.doc_labels)
doc_labels[:10]
```

```
array(['0', '1', '10', '100', '1000', '10000', '10001', '10002', '10003',
       '10004'], dtype='<U5')
```

```
pickle.dump(doc_labels, open('%s/doc_labels_pos.p' % working_directory, 'wb'))

scipy.sparse.save_npz('%s/small_dtm_pos.npz' % working_directory, dtm_sm_pos)
scipy.sparse.save_npz('%s/big_dtm_pos.npz' % working_directory, dtm_bg_pos)

pickle.dump(vocab_bg_pos, open('%s/big_vocab_pos.p' % working_directory, 'wb'))
pickle.dump(vocab_sm_pos, open('%s/small_vocab_pos.p' % working_directory, 'wb'))

pickle.dump(corpus1, open('%s/corpus1.p' % working_directory, 'wb') )
```

```
corpus2 = Corpus()
for i, a_review in enumerate(b):
  the_review = b[a_review]
  corpus2.add_doc(str(i), the_review['reviewText'])
```

```
print(len(corpus2))
```

```
1243
```

```
from tmtoolkit.preprocess import TMPreproc

preproc = TMPreproc(corpus2, language='en')
preproc.pos_tag()
preproc.lemmatize()
preproc.tokens_to_lowercase()
preproc.remove_special_chars_in_tokens()
preproc.add_stopwords(['http', 'nt'])
```

```
/usr/local/lib/python3.7/dist-packages/spacy/util.py:275: UserWarning: [W031] Mo
  warnings.warn(warn_msg)
<TMPreproc [1243 documents / en]>
```

```
preproc_smaller = preproc.copy()
preproc_smaller.filter_for_pos('N', 'V', 'ADJ')
preproc_smaller.clean_tokens(remove_numbers=True, remove_shorter_than=2)
preproc_smaller.remove_common_tokens(df_threshold=0.8)
preproc_smaller.remove_uncommon_tokens(df_threshold=0.01)

print(preproc.vocabulary_size)
print(preproc_smaller.vocabulary_size)
```

```
    4722
    274
```

```
vocab_bg_neg = np.array(preproc.vocabulary)
vocab_sm_neg = np.array(preproc_smaller.vocabulary)
```

```
dtm_bg_neg = preproc.dtm
dtm_sm_neg = preproc_smaller.dtm

dtm_bg_neg, dtm_sm_neg
```

```
    (<1243x4722 sparse matrix of type '<class 'numpy.int32'>'
            with 52979 stored elements in Compressed Sparse Row format>,
     <1243x274 sparse matrix of type '<class 'numpy.int32'>'
            with 12974 stored elements in Compressed Sparse Row format>)
```

```
doc_labels = np.array(preproc.doc_labels)
doc_labels[:10]
```

```
    array(['0', '1', '10', '100', '1000', '1001', '1002', '1003', '1004',
           '1005'], dtype='<U4')
```

```
pickle.dump(doc_labels, open('%s/doc_labels_neg.p' % working_directory, 'wb'))

scipy.sparse.save_npz('%s/small_dtm_neg.npz' % working_directory, dtm_sm_neg)
scipy.sparse.save_npz('%s/big_dtm_neg.npz' % working_directory, dtm_bg_neg)

pickle.dump(vocab_bg_neg, open('%s/big_vocab_neg.p' % working_directory, 'wb'))
pickle.dump(vocab_sm_neg, open('%s/small_vocab_neg.p' % working_directory, 'wb'))

pickle.dump(corpus2, open('%s/corpus2.p' % working_directory, 'wb') )
```

Importing DTM

```
try:
  from tmtoolkit.topicmod.tm_lda import compute_models_parallel
except:
  !pip install tmtoolkit['lda']
  from tmtoolkit.topicmod.tm_lda import compute_models_parallel



import pickle
import scipy.sparse
import logging
import warnings


try:
  from lda import LDA
except:
  !pip install lda


working_directory = '/content/drive/MyDrive/Marketing_Unsupervised_Learning/Final_Pro

doc_labels = pickle.load(open('%s/doc_labels.p' % working_directory, 'rb'))
dtm_sm = scipy.sparse.load_npz('%s/small_dtm.npz' % working_directory)
dtm_bg = scipy.sparse.load_npz('%s/big_dtm.npz' % working_directory)

vocab_bg = pickle.load(open('%s/big_vocab.p' % working_directory, 'rb'))
vocab_sm = pickle.load(open('%s/small_vocab.p' % working_directory, 'rb'))


doc_labels_pos = pickle.load(open('%s/doc_labels_pos.p' % working_directory, 'rb'))
dtm_sm_pos = scipy.sparse.load_npz('%s/small_dtm_pos.npz' % working_directory)
dtm_bg_pos = scipy.sparse.load_npz('%s/big_dtm_pos.npz' % working_directory)

vocab_bg_pos = pickle.load(open('%s/big_vocab_pos.p' % working_directory, 'rb'))
vocab_sm_pos = pickle.load(open('%s/small_vocab_pos.p' % working_directory, 'rb'))


doc_labels_neg = pickle.load(open('%s/doc_labels_neg.p' % working_directory, 'rb'))
dtm_sm_neg = scipy.sparse.load_npz('%s/small_dtm_neg.npz' % working_directory)
dtm_bg_neg = scipy.sparse.load_npz('%s/big_dtm_neg.npz' % working_directory)

vocab_bg_neg = pickle.load(open('%s/big_vocab_neg.p' % working_directory, 'rb'))
vocab_sm_neg = pickle.load(open('%s/small_vocab_neg.p' % working_directory, 'rb'))
```

Creating Models

```
# suppress the "INFO" messages and warnings from lda
logger = logging.getLogger('lda')
logger.addHandler(logging.NullHandler())
logger.propagate = False
warnings.filterwarnings('ignore')



# set data to use
dtms = {
    'bigger': dtm_bg,
    'smaller': dtm_sm,
    'bigger positive': dtm_bg_pos,
    'smaller positive': dtm_sm_pos,
    'bigger negative': dtm_bg_neg,
    'smaller negative': dtm_sm_neg
}

# and fixed hyperparameters
# Here, alpha represents document-topic density - with a higher alpha, documents
# are made up of more topics, and with lower alpha, documents contain fewer topics.
#Beta represents topic-word density - with a high beta, topics are made up of
#most of the words in the corpus, and with a low beta they consist of few words.
# https://www.thoughtvector.io/blog/lda-alpha-and-beta-parameters-the-intuition/
lda_params = {
    'n_topics': 16,
    'eta': .01,
    'n_iter': 500,
    'random_state': 20191122,  # to make results reproducible
    'alpha': 1/16
}

models = compute_models_parallel(dtms, constant_parameters=lda_params)


from tmtoolkit.topicmod.model_io import print_ldamodel_topic_words

model_sm = models['smaller'][0][1]
print_ldamodel_topic_words(model_sm.topic_word_, vocab_sm, top_n=5)
```

```
    topic_1
    > #1. fit (0.056601)
    > #2. order (0.052941)
    > #3. small (0.050899)
    > #4. wear (0.044685)
    > #5. great (0.039664)
    topic_2
    > #1. like (0.047710)
    > #2. little (0.042452)
    > #3. wide (0.037879)
    > #4. look (0.037574)
    > #5. fit (0.035516)
    topic_3
    > #1. small (0.074416)
    > #2. wear (0.048587)
    > #3. order (0.048508)
```

```
     > #4. get (0.036382)
     > #5. big (0.035909)
     topic_4
     > #1. buy (0.055711)
     > #2. wear (0.053542)
     > #3. last (0.034836)
     > #4. good (0.031380)
     > #5. get (0.026703)
     topic_5
     > #1. good (0.115003)
     > #2. great (0.086093)
     > #3. like (0.078888)
     > #4. look (0.076882)
     > #5. really (0.075605)
     topic_6
     > #1. recommend (0.135709)
     > #2. would (0.103716)
     > #3. great (0.064369)
     > #4. comfortable (0.059881)
     > #5. good (0.047752)
     topic_7
     > #1. back (0.033139)
     > #2. get (0.033067)
     > #3. order (0.032634)
     > #4. go (0.028085)
     > #5. send (0.028013)
     topic_8
     > #1. get (0.062702)
     > #2. buy (0.040603)
     > #3. make (0.039704)
     > #4. like (0.037532)
     > #5. look (0.031239)
     topic_9
     > #1. love (0.097920)
     > #2. wear (0.092992)
     > #3. buy (0.074923)
     > #4. comfortable (0.062056)
     > #5. great (0.048002)
     topic_10
     > #1. great (0.060172)
     > #2. well (0.058097)
```

```
model_bg = models['bigger'][0][1]
print_ldamodel_topic_words(model_bg.topic_word_, vocab_bg, top_n=5)
```

```
     topic_1
     > #1.  (0.130852)
     > #2. he (0.051235)
     > #3. my (0.047863)
     > #4. for (0.040565)
     > #5. be (0.038153)
     topic_2
     > #1.  (0.134098)
     > #2. the (0.049200)
     > #3. it (0.043192)
```

```
> #4. be (0.042951)
> #5. watch (0.031660)
topic_3
> #1.  (0.145842)
> #2. and (0.047536)
> #3. be (0.033188)
> #4. for (0.032744)
> #5. shoe (0.023944)
topic_4
> #1.  (0.108395)
> #2. i (0.055194)
> #3. a (0.050055)
> #4. size (0.049034)
> #5. be (0.040967)
topic_5
> #1.  (0.166099)
> #2. be (0.054997)
> #3. the (0.046419)
> #4. and (0.045779)
> #5. i (0.031217)
topic_6
> #1.  (0.115574)
> #2. i (0.061583)
> #3. be (0.039629)
> #4. and (0.032860)
> #5. they (0.027337)
topic_7
> #1.  (0.125347)
> #2. i (0.035601)
> #3. the (0.035135)
> #4. shoe (0.032289)
> #5. be (0.031722)
topic_8
> #1.  (0.122292)
> #2. i (0.065434)
> #3. be (0.039452)
> #4. have (0.029563)
> #5. the (0.028327)
topic_9
> #1.  (0.098228)
> #2. the (0.092874)
> #3. be (0.045929)
> #4. of (0.027653)
> #5. a (0.020667)
topic_10
> #1.  (0.115236)
> #2. i (0.045774)
> #3. be (0.041657)
```

```
model_bg_pos = models['bigger positive'][0][1]
print_ldamodel_topic_words(model_bg_pos.topic_word_, vocab_bg_pos, top_n=5)
```

```
topic_1
> #1.  (0.121707)
> #2. i (0.057859)
> #3. and (0.036639)
> #4. be (0.033370)
> #5. my (0.028136)
topic_2
> #1.  (0.138244)
> #2. i (0.064574)
> #3. be (0.042895)
> #4. have (0.038359)
> #5. shoe (0.031054)
topic_3
> #1.  (0.095429)
> #2. i (0.056065)
> #3. have (0.032518)
> #4. be (0.030335)
> #5. the (0.028690)
topic_4
> #1.  (0.144569)
> #2. y (0.032998)
> #3. muy (0.026147)
> #4. de (0.022791)
> #5. el (0.019575)
topic_5
> #1.  (0.130834)
> #2. be (0.054484)
> #3. the (0.048373)
> #4. i (0.037893)
> #5. and (0.035807)
topic_6
> #1.  (0.133094)
> #2. it (0.052334)
> #3. the (0.045478)
> #4. be (0.045401)
> #5. i (0.029993)
topic_7
> #1.  (0.151129)
> #2. the (0.061800)
> #3. be (0.059208)
> #4. and (0.038107)
> #5. they (0.029811)
topic_8
> #1.  (0.114695)
> #2. i (0.062883)
> #3. be (0.033985)
> #4. and (0.026605)
> #5. the (0.025308)
topic_9
> #1.  (0.108259)
> #2. i (0.051730)
> #3. be (0.038026)
> #4. the (0.032395)
> #5. to (0.030003)
topic_10
> #1.  (0.115921)
> #2. i (0.051303)
> #3. a (0.045695)
```

```
model_bg = models['smaller negative'][0][1]
```

```
print_ldamodel_topic_words(model_sm.topic_word_, vocab_sm_neg, top_n=5)

    topic_1
    > #1. easy (0.056601)
    > #2. keep (0.052941)
    > #3. offer (0.050899)
    > #4. red (0.044685)
    > #5. especially (0.039664)
    topic_2
    > #1. fix (0.047710)
    > #2. flat (0.042452)
    > #3. return (0.037879)
    > #4. full (0.037574)
    > #5. easy (0.035516)
    topic_3
    > #1. offer (0.074416)
    > #2. red (0.048587)
    > #3. keep (0.048508)
    > #4. either (0.036382)
    > #5. arrive (0.035909)
    topic_4
    > #1. back (0.055711)
    > #2. red (0.053542)
    > #3. finally (0.034836)
    > #4. enough (0.031380)
    > #5. either (0.026703)
    topic_5
    > #1. enough (0.115003)
    > #2. especially (0.086093)
    > #3. fix (0.078888)
    > #4. full (0.076882)
    > #5. look (0.075605)
    topic_6
    > #1. loud (0.135709)
    > #2. save (0.103716)
    > #3. especially (0.064369)
    > #4. beware (0.059881)
    > #5. enough (0.047752)
    topic_7
    > #1. appear (0.033139)
    > #2. either (0.033067)
    > #3. keep (0.032634)
    > #4. end (0.028085)
    > #5. new (0.028013)
    topic_8
    > #1. either (0.062702)
    > #2. back (0.040603)
    > #3. go (0.039704)
    > #4. fix (0.037532)
    > #5. full (0.031239)
    topic_9
    > #1. get (0.097920)
    > #2. red (0.092992)
    > #3. back (0.074923)
    > #4. beware (0.062056)
    > #5. especially (0.048002)
```

```
     topic_10
     > #1. especially (0.060172)
     > #2. remove (0.058097)
     > #3. beware (0.046512)
```

```
model_bg = models['bigger negative'][0][1]
print_ldamodel_topic_words(model_bg.topic_word_, vocab_bg_neg, top_n=5)
```

```
     topic_1
     > #1. i (0.073670)
     > #2.  (0.071039)
     > #3. to (0.046483)
     > #4. be (0.045781)
     > #5. it (0.041923)
     topic_2
     > #1.  (0.147261)
     > #2. the (0.096572)
     > #3. be (0.070927)
     > #4. shoe (0.060508)
     > #5. not (0.031257)
     topic_3
     > #1.  (0.143518)
     > #2. the (0.067671)
     > #3. watch (0.060455)
     > #4. it (0.056446)
     > #5. i (0.041694)
     topic_4
     > #1.  (0.107365)
     > #2. to (0.055744)
     > #3. the (0.047504)
     > #4. of (0.032720)
     > #5. a (0.028842)
     topic_5
     > #1.  (0.093927)
     > #2. i (0.067125)
     > #3. be (0.041864)
     > #4. to (0.039967)
     > #5. the (0.034037)
     topic_6
     > #1.  (0.169295)
     > #2. be (0.069224)
     > #3. they (0.046652)
     > #4. not (0.043454)
     > #5. these (0.039692)
     topic_7
     > #1.  (0.077981)
     > #2. the (0.071812)
     > #3. of (0.040019)
     > #4. i (0.030371)
     > #5. and (0.029105)
     topic_8
     > #1.  (0.210918)
     > #2. the (0.057262)
     > #3. shoe (0.043867)
     > #4. of (0.033098)
     > #5. for (0.032047)
```

```
topic_9
> #1.  (0.117408)
> #2. the (0.067261)
> #3. be (0.050893)
> #4. shoe (0.028572)
> #5. and (0.027233)
topic_10
> #1.  (0.102362)
> #2. i (0.066253)
> #3. shoe (0.033912)
```

Evaluation

```
import os
import matplotlib

if matplotlib.__version__ != "3.1.3":
    !pip uninstall -y matplotlib
    !pip install matplotlib==3.1.3
    os.kill(os.getpid(), 9)
```

```
Found existing installation: matplotlib 3.1.3
Uninstalling matplotlib-3.1.3:
  Successfully uninstalled matplotlib-3.1.3
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-whe
Collecting matplotlib==3.1.3
  Using cached matplotlib-3.1.3-cp37-cp37m-manylinux1_x86_64.whl (13.1 MB)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/l
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dis
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dis
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-package
Installing collected packages: matplotlib
ERROR: pip's dependency resolver does not currently take into account all the pa
tmtoolkit 0.10.0 requires matplotlib<3.4,>=3.3.0, but you have matplotlib 3.1.3 
albumentations 0.1.12 requires imgaug<0.2.7,>=0.2.5, but you have imgaug 0.2.9 w
Successfully installed matplotlib-3.1.3
```

```
doc_labels = pickle.load(open('%s/doc_labels.p' % working_directory, 'rb'))

vocab_sm = scipy.sparse.load_npz('%s/small_dtm.npz' % working_directory)
vocab_bg = scipy.sparse.load_npz('%s/big_dtm.npz' % working_directory)

vocab_bg = pickle.load(open('%s/big_vocab.p' % working_directory, 'rb'))
vocab_sm = pickle.load(open('%s/small_vocab.p' % working_directory, 'rb'))
```

```
dtm_sm = scipy.sparse.load_npz('%s/small_dtm.npz' % working_directory)
dtm_bg = scipy.sparse.load_npz('%s/big_dtm.npz' % working_directory)
```

```
const_params = {'n_iter': 500,'eta': 0.1, 'random_state': 20191122  }
ks = list(range(1, 50, 5))
print(ks)
varying_params = [dict(n_topics=k, alpha=1/k) for k in ks]
print(varying_params)
```

```
[1, 6, 11, 16, 21, 26, 31, 36, 41, 46]
[{'n_topics': 1, 'alpha': 1.0}, {'n_topics': 6, 'alpha': 0.16666666666666666}, {
```

```
from tmtoolkit.topicmod import tm_lda
eval_results = tm_lda.evaluate_topic_models(dtm_sm_neg,
    varying_params,
    const_params)
```

```
from tmtoolkit.topicmod.evaluate import results_by_parameter
from tmtoolkit.topicmod.visualize import plot_eval_results

results_by_n_topics = results_by_parameter(eval_results, 'n_topics')
print(results_by_n_topics)
```

```
[(1, {'cao_juan_2009': nan, 'arun_2010': 153.98953901259955, 'coherence_mimno_20
```

```
plot_eval_results(results_by_n_topics)
```

```
(<Figure size 576x432 with 3 Axes>,
 array([<matplotlib.axes._subplots.AxesSubplot object at 0x7fc07e7a5910>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc07d35e950>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc07d318d10>]
```

Lower Beta

```
const_params = {'n_iter': 500,'eta': 0.01, 'random_state': 20191122}
ks = list(range(1, 50, 5))
print(ks)
varying_params = [dict(n_topics=k, alpha=1/k) for k in ks]
print(varying_params)
```

```
[1, 6, 11, 16, 21, 26, 31, 36, 41, 46]
[{'n_topics': 1, 'alpha': 1.0}, {'n_topics': 6, 'alpha': 0.16666666666666666}, {
```

```
from tmtoolkit.topicmod import tm_lda
eval_results = tm_lda.evaluate_topic_models(dtm_sm_neg,
    varying_params,
    const_params)
```

```
from tmtoolkit.topicmod.evaluate import results_by_parameter
from tmtoolkit.topicmod.visualize import plot_eval_results

results_by_n_topics = results_by_parameter(eval_results, 'n_topics')
print(results_by_n_topics)
```

```
[(1, {'cao_juan_2009': nan, 'arun_2010': 153.96324418591306, 'coherence_mimno_20
```

```
plot_eval_results(results_by_n_topics)
```

```
(<Figure size 576x432 with 3 Axes>,
 array([<matplotlib.axes._subplots.AxesSubplot object at 0x7fc07d2239d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc07cd88d90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc07d223a50>],
       dtype=object))
```

Eval Higher Beta

```
const_params = {'n_iter': 500,'eta': 0.5, 'random_state': 20191122}
ks = list(range(1, 50, 5))
print(ks)
varying_params = [dict(n_topics=k, alpha=1/k) for k in ks]
print(varying_params)
```

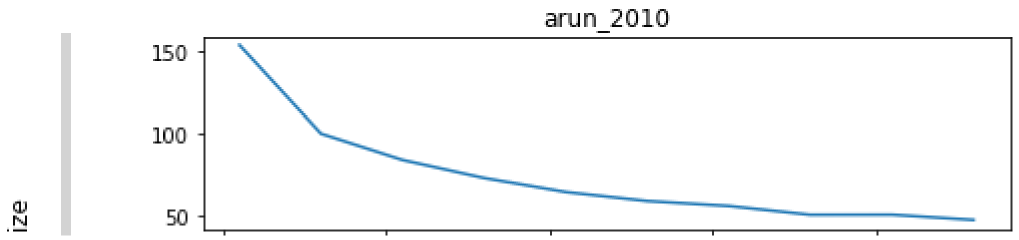
```
[1, 6, 11, 16, 21, 26, 31, 36, 41, 46]
[{'n_topics': 1, 'alpha': 1.0}, {'n_topics': 6, 'alpha': 0.16666666666666666}, {
```

```
from tmtoolkit.topicmod import tm_lda
eval_results = tm_lda.evaluate_topic_models(dtm_sm_neg,
    varying_params,
    const_params)


from tmtoolkit.topicmod.evaluate import results_by_parameter
from tmtoolkit.topicmod.visualize import plot_eval_results

results_by_n_topics = results_by_parameter(eval_results, 'n_topics')
print(results_by_n_topics)
```

```
[(1, {'cao_juan_2009': nan, 'arun_2010': 154.10543277374092, 'coherence_mimno_20
```

```
plot_eval_results(results_by_n_topics)
```

```
(<Figure size 576x432 with 3 Axes>,
 array([<matplotlib.axes._subplots.AxesSubplot object at 0x7fc07cc7ea10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc07cc9f490>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc07cc536d0>],
       dtype=object))
```



Eval Lower Alpha

```
const_params = {'n_iter': 500,'eta': 0.1, 'random_state': 20191122}
ks = list(range(1, 50, 5))
print(ks)
varying_params = [dict(n_topics=k, alpha=1/(10*k)) for k in ks]
print(varying_params)
```

```
[1, 6, 11, 16, 21, 26, 31, 36, 41, 46]
[{'n_topics': 1, 'alpha': 0.1}, {'n_topics': 6, 'alpha': 0.016666666666666666},
```

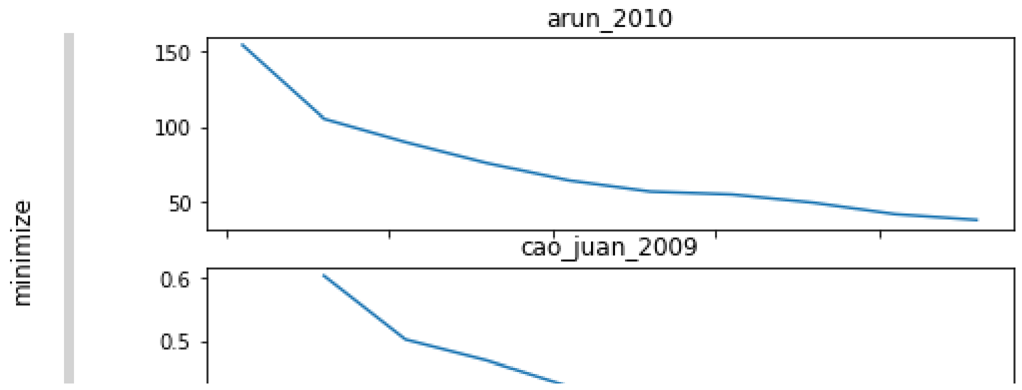
```
from tmtoolkit.topicmod import tm_lda
eval_results = tm_lda.evaluate_topic_models(dtm_sm_neg,
    varying_params,
    const_params)
```

```
from tmtoolkit.topicmod.evaluate import results_by_parameter
from tmtoolkit.topicmod.visualize import plot_eval_results

results_by_n_topics = results_by_parameter(eval_results, 'n_topics')
print(results_by_n_topics)
```

```
[(1, {'cao_juan_2009': nan, 'arun_2010': 153.98953901259955, 'coherence_mimno_20
```

```
plot_eval_results(results_by_n_topics)
```

```
(<Figure size 576x432 with 3 Axes>,
 array([<matplotlib.axes._subplots.AxesSubplot object at 0x7fc07cb83ad0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc07cb3d110>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fc0806ffd50>],
       dtype=object))
```



Classification

```
try:
  import pyLDAvis
except:
  !pip install pyLDAvis==2.1.2
  import pyLDAvis
try:
  import tmtoolkit
except:
  !pip install tmtoolkit
  import tmtoolkit

import nltk
import random
import numpy as np
from tmtoolkit.corpus import Corpus
import json

try:
  from lda import LDA
except:
  !pip install lda
  from lda import LDA

import logging
import warnings
from tmtoolkit.topicmod.tm_lda import compute_models_parallel

import pickle
import scipy.sparse
```

```
random.seed(20191120)   # to make the sampling reproducible
np.set_printoptions(precision=5)
```

```
    Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-whe
    Collecting pyLDAvis==2.1.2
      Downloading pyLDAvis-2.1.2.tar.gz (1.6 MB)
         |████████████████████████████████| 1.6 MB 10.6 MB/s
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
    Mounted at /content/drive
    Requirement already satisfied: jinja2>=2.7.2 in /usr/local/lib/python3.7/dist-pa
```

```
working_directory = 'drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project'
json_path = "%s/allnikereviews.json" % working_directory
```

```
    Collecting funcy
```

```
doc_labels_neg = pickle.load(open('%s/doc_labels_neg.p' % working_directory, 'rb'))
dtm_sm_neg = scipy.sparse.load_npz('%s/small_dtm_neg.npz' % working_directory)
dtm_bg_neg = scipy.sparse.load_npz('%s/big_dtm_neg.npz' % working_directory)

vocab_bg_neg = pickle.load(open('%s/big_vocab_neg.p' % working_directory, 'rb'))
vocab_sm_neg = pickle.load(open('%s/small_vocab_neg.p' % working_directory, 'rb'))
```

```
    Requirement already satisfied: attrs>=17.4.0 in /usr/local/lib/python3.7/dist-pa
```

```
# suppress the "INFO" messages and warnings from lda
logger = logging.getLogger('lda')
logger.addHandler(logging.NullHandler())
logger.propagate = False
warnings.filterwarnings('ignore')
```

```
      Created wheel for pyLDAvis: filename=pyLDAvis-2.1.2-py2.py3-none-any.whl size=
```

Creating Models

```
    Successfully installed funcy-1.17 pyLDAvis-2.1.2
```

```
# set data to use
dtms = {
    'smaller negative': dtm_sm_neg
}

# and fixed hyperparameters
lda_params = {
    'n_topics': 21,
    'eta': .5,
    'alpha': 1/21,
    'n_iter': 1000,
    'random_state': 20191122  # to make results reproducible
}

models = compute_models_parallel(dtms, constant_parameters=lda_params)
```

```
    /usr/local/lib/python3.7/dist-packages/scipy/sparse/sputils.py:17: DeprecationWa
```

```
from tmtoolkit.topicmod.model_io import print_ldamodel_topic_words
model_sm = models['smaller negative'][0][1]
```

```
print_ldamodel_topic_words(model_sm.topic_word_, vocab_sm_neg, top_n=3)
```

```
topic_1
> #1. would (0.055123)
> #2. buy (0.036965)
> #3. get (0.030480)
topic_2
> #1. make (0.053404)
> #2. want (0.041667)
> #3. get (0.032277)
topic_3
> #1. get (0.057337)
> #2. back (0.033909)
> #3. know (0.031443)
topic_4
> #1. small (0.065682)
> #2. wear (0.045316)
> #3. order (0.037169)
topic_5
> #1. buy (0.046206)
> #2. get (0.042315)
> #3. go (0.038424)
topic_6
> #1. back (0.063035)
> #2. send (0.053244)
> #3. purchase (0.034884)
topic_7
> #1. run (0.047030)
> #2. buy (0.038119)
> #3. bad (0.038119)
topic_8
> #1. wide (0.049258)
> #2. well (0.042510)
> #3. narrow (0.039811)
topic_9
> #1. buy (0.093249)
> #2. last (0.054348)
> #3. one (0.046339)
topic_10
> #1. fall (0.057814)
> #2. apart (0.055273)
> #3. buy (0.041296)
topic_11
> #1. order (0.070302)
> #2. send (0.059790)
> #3. receive (0.050591)
topic_12
> #1. like (0.066376)
> #2. feel (0.058624)
> #3. wear (0.051841)
topic_13
> #1. use (0.042169)
> #2. good (0.032798)
> #3. get (0.027443)
topic_14
> #1. look (0.052727)
> #2. fake (0.050303)
> #3. real (0.049091)
topic_15
> #1. take (0.037906)

Collecting thinc<7.5.0,>=7.4.1
```

Topic Names and Classification

```
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packag
```

```
from tmtoolkit.bow.bow_stats import doc_lengths
from tmtoolkit.topicmod.model_stats import generate_topic_labels_from_top_words

doc_lengths_sm = doc_lengths(dtm_sm_neg)
topic_labels = generate_topic_labels_from_top_words(
    model_sm.topic_word_,
    model_sm.doc_topic_,
    doc_lengths_sm,
    vocab_sm_neg,
    lambda_=0.7
)
```

```
      Successfully uninstalled thinc-7.4.0
```

```
topic_labels
```

```
array(['1_would_replace', '2_make_want', '3_get_know', '4_small_12',
       '5_go_buy', '6_back_send', '7_run_bad', '8_wide_narrow',
       '9_buy_last', '10_fall_apart', '11_order_black', '12_feel_like',
       '13_use_good', '14_real_fake', '15_loud_take', '16_look_like',
       '17_make_wear', '18_break_buy', '19_return_make',
       '20_order_return', '21_back_get'], dtype='<U15')
    Found existing installation: scipy 1.4.1
```

```
from tmtoolkit.topicmod.model_io import ldamodel_top_doc_topics
doc_topic = model_sm.doc_topic_
documentclassifications = ldamodel_top_doc_topics(doc_topic, doc_labels_neg, top_n=2,
```

```
    Uninstalling pandas-1.3.5:
```

```
import pandas as pd
```

```
    Found existing installation: matplotlib 3.2.2
```

```
#documentclassifications.head()
```

```
ERROR: pip's dependency resolver does not currently take into account all the pa
```

```
corpus = pickle.load(open('%s/corpus2.p' % working_directory, 'rb'))
```

```
/usr/local/lib/python3.7/dist-packages/nltk/decorators.py:70: DeprecationWarning
```

```
documentclassifications['text'] = np.nan
for index, arow in documentclassifications.iterrows():
  documentclassifications['text'][index] = corpus[index]
```

```
    |████████████████████████████████| 351 kB 8.5 MB/s
```

```
documentclassifications.loc['1']
```

```
rank_1                               20_order_return (0.5655)
```

```
rank_2                              19_return_make (0.1905)
text      I'm on my 4th watch... I keep returning it due...
Name: 1, dtype: object
```

```
documentclassifications.loc['1']['text']
```

```
'I'm on my 4th watch... I keep returning it due to poor design.  The band keeps
coming apart in the same spot!  Nike hasn't been helpful when I've been in conta
ct with them.  Now, I'm on my 4th watch and something funking is going on with t
he face of this watch and I've brought it in to a jeweler to have the battery ch
```

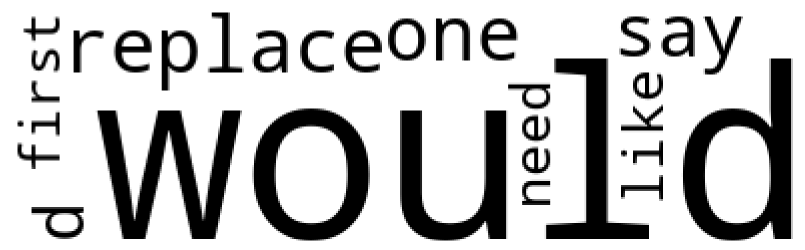
```
documentclassifications.to_excel('%s/topics.documentclassification.xlsx' % working_di
```

```
from tmtoolkit.topicmod.visualize import parameters_for_ldavis

ldavis_params = parameters_for_ldavis(model_sm.topic_word_,
                                      model_sm.doc_topic_,
                                      dtm_sm_neg,
                                      vocab_sm_neg)
```

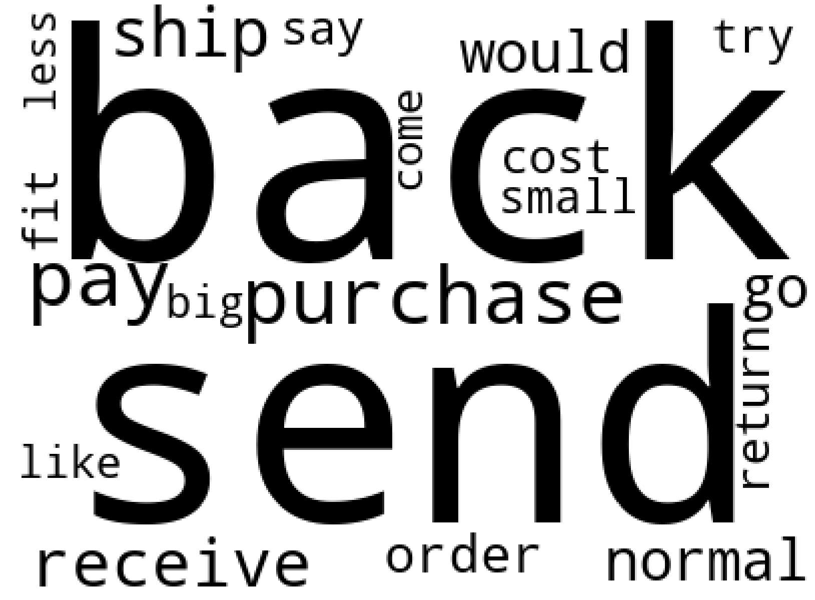
```
from tmtoolkit.topicmod.visualize import generate_wordclouds_for_topic_words

# some options for wordcloud output
img_w = 400   # image width
img_h = 300   # image height

topic_clouds = generate_wordclouds_for_topic_words(
    model_sm.topic_word_, vocab_sm_neg,
    top_n=20, topic_labels=topic_labels,
    width=img_w, height=img_h
)

# show all generated word clouds
topic_clouds.keys()
```

```
dict_keys(['1_would_replace', '2_make_want', '3_get_know', '4_small_12', '5_go_b
```

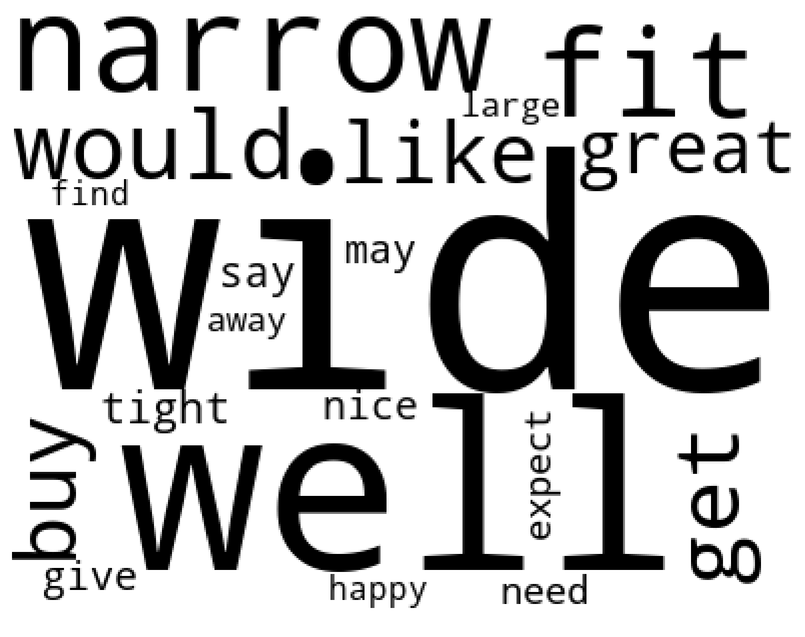
```
topic_clouds['1_would_replace']
```



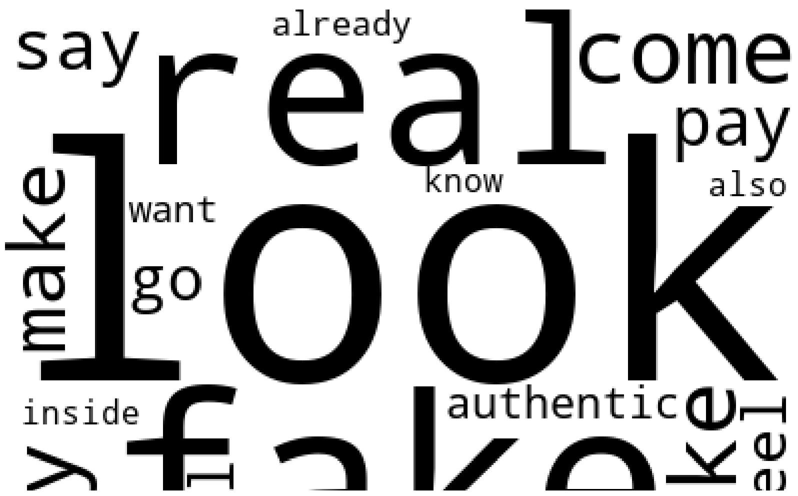
```
topic_clouds['6_back_send']
```



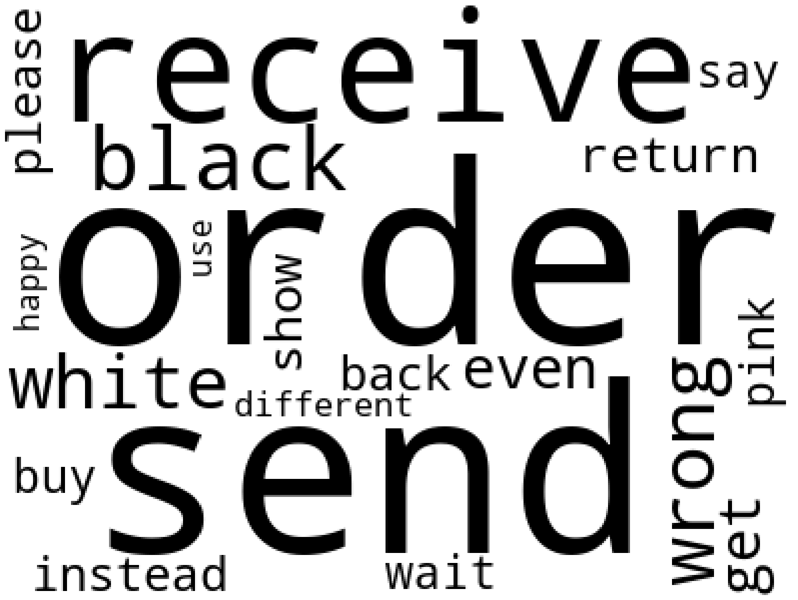
```
topic_clouds['8_wide_narrow']
```



```
topic_clouds['14_real_fake']
```



```
topic_clouds['11_order_black']
```



Topic Modeling with BERTopic

```
import json
import pandas as pd
import os
try:
  from bertopic import BERTopic
except:
  !pip install bertopic[all]
  os.kill(os.getpid(), 9)
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
    Mounted at /content/drive
```

```
working_directory = 'drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project'
json_path = "%s/allnikereviews.json" % working_directory
json_file = json.load(open(json_path, 'r'))
```

```
df = pd.read_json('drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project/allnik
```

```
negative = df[(df['overall'] == 1)]
```

```
neg = negative.reset_index(drop=True)
```

```
b = neg.to_dict(orient = 'index')
```

```
allreviewtext = []
for areview in b:
  allreviewtext.append(b[areview]['reviewText'])
```

```
len(allreviewtext)
```

```
    1243
```

```
topic_model = BERTopic(language="english", calculate_probabilities=True, verbose=True
topics, probs = topic_model.fit_transform(allreviewtext)
```

```
Downloading: 100%        1.18k/1.18k [00:00<00:00, 9.07kB/s]
Downloading: 100%        190/190 [00:00<00:00, 2.54kB/s]
Downloading: 100%        10.2k/10.2k [00:00<00:00, 107kB/s]
Downloading: 100%        612/612 [00:00<00:00, 7.81kB/s]
Downloading: 100%        116/116 [00:00<00:00, 1.27kB/s]

Downloading: 100%        349/349 [00:00<00:00, 3.64kB/s]
```

Extracting Topics

```
freq = topic_model.get_topic_info(); freq.head(5)
```

	Topic	Count	Name
0	0	1104	0_the_and_to_of
1	1	139	1_watch_the_it_to

```
topic_model.get_topic(0)  # Select the most frequent topic
```

```
[('the', 0.1226312069283534),
 ('and', 0.0820134147955848),
 ('to', 0.07451743057510261),
 ('of', 0.05946631460170428),
 ('they', 0.05295465223975309),
 ('shoes', 0.05285020395469197),
 ('for', 0.05112947203775947),
 ('not', 0.04812367581928727),
 ('is', 0.047972893832918535),
 ('my', 0.04787900067991311)]
```

Visualization

```
topic_model.visualize_hierarchy(top_n_topics=50)
```

Hierarchical Clustering

```
watch the it-
```

Trying Again

```
working_directory = 'drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project'
json_path = "%s/allnikereviews.json" % working_directory
json_file = json.load(open(json_path, 'r'))
```

```
allreviewtext = []
for areview in json_file:
  allreviewtext.append(json_file[areview]['reviewText'])
```

```
topic_model = BERTopic(language="english", calculate_probabilities=True, verbose=True)
topics, probs = topic_model.fit_transform(allreviewtext)
```

```
Batches: 100%                                     675/675 [12:50<00:00, 7.35it/s]
2022-06-03 18:43:17,739 - BERTopic - Transformed documents to Embeddings
2022-06-03 18:44:11,311 - BERTopic - Reduced dimensionality
2022-06-03 18:44:52,235 - BERTopic - Clustered reduced embeddings
```

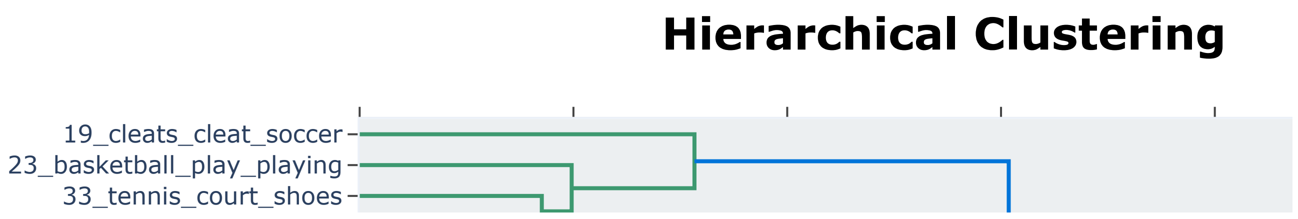
```
freq = topic_model.get_topic_info(); freq.head(5)
```

	Topic	Count	Name
0	-1	7876	-1_shoes_shoe_them_and
1	0	1142	0_watch_it_band_wrist
2	1	1055	1_nike_nikes_as_of
3	2	645	2_socks_sock_are_they
4	3	512	3_running_run_shoes_runner

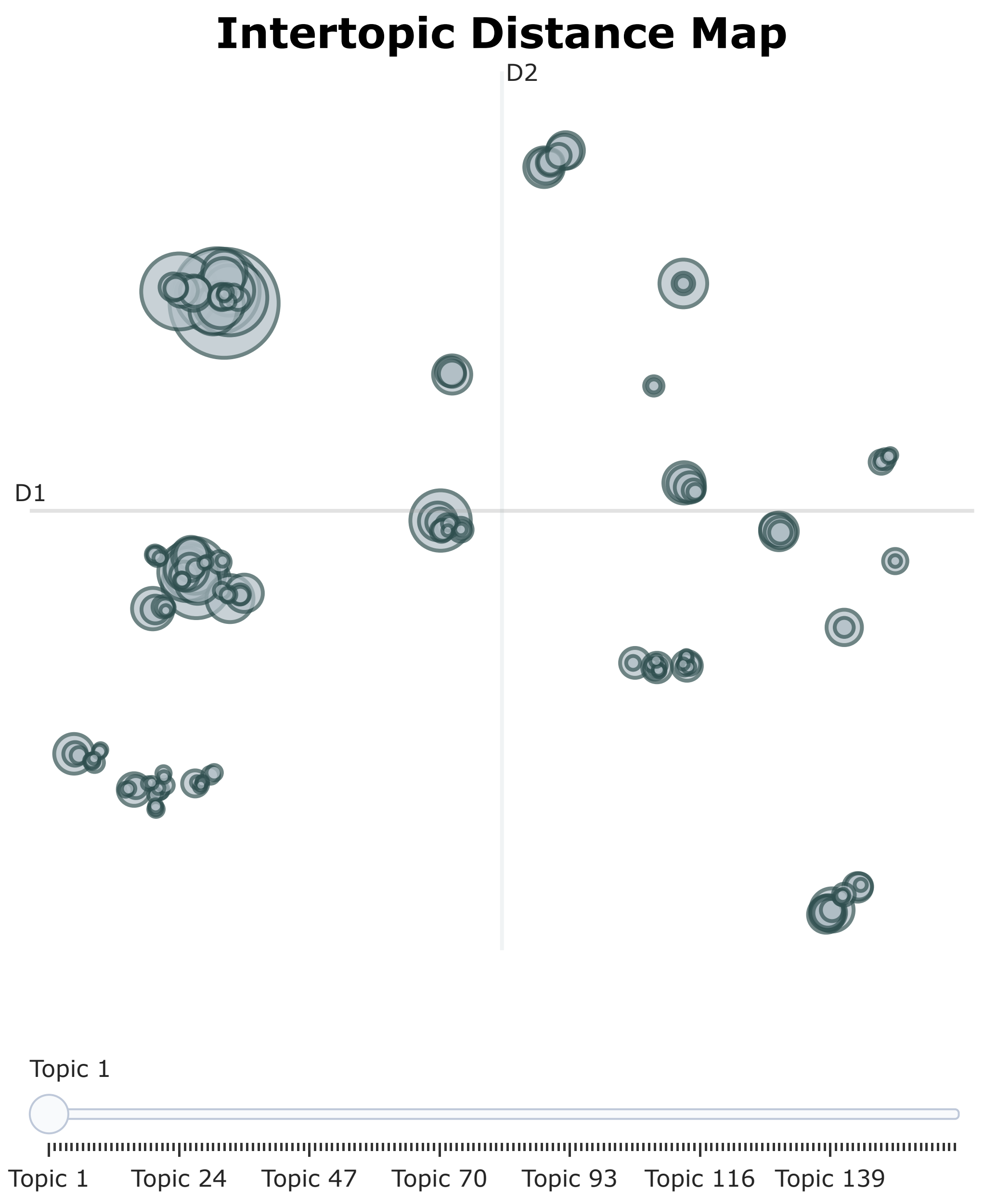
```
topic_model.get_topic(0)  # Select the most frequent topic
```

```
[('watch', 0.035799559412178036),
 ('it', 0.012771515914620515),
 ('band', 0.011974676486615387),
 ('wrist', 0.01050496037770599),
 ('watches', 0.009048698806764675),
 ('battery', 0.00901343802823276),
 ('is', 0.008540975328548405),
 ('this', 0.008299059492684821),
 ('the', 0.007595207988257942),
 ('its', 0.007227063955332146)]
```

```
topic_model.visualize_hierarchy(top_n_topics=50)
```



```
topic_model.visualize_topics()
```



```
working_directory = 'drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project'
```

```
from google.colab import drive
```

```
drive.mount('/content/drive')

     Mounted at /content/drive


import json
import gzip
path = working_directory

def parse(path):
  g = gzip.open(path, 'r')
  for l in g:
    yield json.dumps(eval(l))

f = open("output.strict", 'w')
for l in parse("drive/MyDrive/Marketing_Unsupervised_Learning/Final_Project/meta_Clotl
  f.write(l + '\n')


f

     <_io.TextIOWrapper name='output.strict' mode='w' encoding='UTF-8'>
```