Topic modelling

In [6]:

```
import pandas as pd
import numpy as np
reviews_datasets = pd.read_csv("Reviews.csv")
reviews_datasets = reviews_datasets.head(20000)
reviews_datasets.dropna()
```

Out[6]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0
...	...	...	...	...	...	...
19995	19996	B002C50X1M	A1XRXZI5KOMVDD	KAF1958 "amandaf0626"	0	0
19996	19997	B002C50X1M	A7G9M0IE7LABX	Kevin	0	0
19997	19998	B002C50X1M	A38J5PRUDESMZF	ray	0	0
19998	19999	B002C50X1M	A17TPOSAG43GSM	Herrick	0	0
19999	20000	B002C50X1M	A3LWC833HQIG7J	austin_Larry	0	0

20000 rows × 10 columns

In [7]:
```
reviews_datasets['Text'][350]
```

Out[7]:
```
'These chocolate covered espresso beans are wonderful!  The chocolate is very dark and ri
ch and the "bean" inside is a very delightful blend of flavors with just enough caffine t
o really give it a zing.'
```

In [9]:
```
from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer(max_df=0.8, min_df=2, stop_words='english')
doc_term_matrix = count_vect.fit_transform(reviews_datasets['Text'].values.astype('U')
```

In [11]:
```
print(doc_term_matrix)
```

```
  (0, 1792)     1
  (0, 13973)    1
  (0, 2171)     1
  (0, 4160)     1
  (0, 5304)     1
  (0, 9997)     1
  (0, 5771)     1
  (0, 10221)    1
  (0, 9993)     2
  (0, 7640)     1
  (0, 7498)     1
  (0, 12347)    1
  (0, 9978)     1
  (0, 7992)     1
  (0, 11845)    1
  (0, 1537)     2
  (0, 7276)     1
  (0, 5120)     1
  (0, 910)      1
  (1, 9993)     2
  (1, 980)      1
  (1, 7267)     1
  (1, 7073)     2
  (1, 11136)    1
  (1, 9319)     2
  :     :
  (19999, 10161)        1
  (19999, 1296) 1
  (19999, 4741) 1
  (19999, 13468)        1
  (19999, 11691)        1
  (19999, 14361)        1
  (19999, 12890)        1
  (19999, 11135)        1
  (19999, 11877)        1
  (19999, 9775) 1
  (19999, 9628) 1
  (19999, 13008)        1
  (19999, 2538) 1
  (19999, 4430) 1
  (19999, 1435) 1
  (19999, 2542) 2
  (19999, 3210) 1
  (19999, 13942)        2
  (19999, 7720) 1
  (19999, 13081)        1
  (19999, 2025) 1
  (19999, 13954)        1
  (19999, 6223) 1
  (19999, 6870) 1
  (19999, 6587) 1
```

In [15]:
```
from sklearn.decomposition import LatentDirichletAllocation
LDA = LatentDirichletAllocation(n_components=5, random_state=42)
LDA.fit(doc_term_matrix)
```

Out[15]:
```
LatentDirichletAllocation(n_components=5, random_state=42)
```

In [16]:
```
import random
for i in range(10):
    random_id=random.randint(0,len(count_vect.get_feature_names_out()))
    print(count_vect.get_feature_names_out()[random_id])
```

```
asap
industrial
taters
loooove
scientists
unexpectedly
www
eating
drawbacks
greece
```

In [17]:
```
first_topic = LDA.components_[0]
```

In [18]:
```
top_topic_words = first_topic.argsort()[-10:]
```

In [19]:
```
for i in top_topic_words:
    print(count_vect.get_feature_names_out()[i])
```

```
water
great
just
drink
sugar
good
flavor
taste
like
tea
```

In [20]:
```
for i,topic in enumerate(LDA.components_):
    print(f"Top 10 words for topic #{i}:")
    print([count_vect.get_feature_names_out()[i] for i in topic.argsort()[-10:]])
    print('\n')
```

```
Top 10 words for topic #0:
['water', 'great', 'just', 'drink', 'sugar', 'good', 'flavor', 'taste', 'like', 'tea']


Top 10 words for topic #1:
['br', 'chips', 'love', 'flavor', 'chocolate', 'just', 'great', 'taste', 'good', 'like']


Top 10 words for topic #2:
['just', 'drink', 'orange', 'sugar', 'soda', 'water', 'like', 'juice', 'product', 'br']


Top 10 words for topic #3:
['gluten', 'eat', 'free', 'product', 'like', 'dogs', 'treats', 'dog', 'br', 'food']


Top 10 words for topic #4:
['cups', 'price', 'great', 'like', 'amazon', 'good', 'br', 'product', 'cup', 'coffee']
```

In [21]:
```
topic_values = LDA.transform(doc_term_matrix)
topic_values.shape
```

Out[21]:
```
(20000, 5)
```

In [22]:
```
reviews_datasets['Topic'] = topic_values.argmax(axis=1)
```

In [23]:
```
reviews_datasets.head(3)
```

Out[23]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1:
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1:
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1:

In [24]:

```
import pandas as pd
import numpy as np
reviews_datasets = pd.read_csv("Reviews.csv")
reviews_datasets = reviews_datasets.head(20000)
reviews_datasets.dropna()
```

Out[24]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0
...	...	...	...	...	...	...
19995	19996	B002C50X1M	A1XRXZI5KOMVDD	KAF1958 "amandaf0626"	0	0
19996	19997	B002C50X1M	A7G9M0IE7LABX	Kevin	0	0
19997	19998	B002C50X1M	A38J5PRUDESMZF	ray	0	0
19998	19999	B002C50X1M	A17TPOSAG43GSM	Herrick	0	0
19999	20000	B002C50X1M	A3LWC833HQIG7J	austin_Larry	0	0

20000 rows × 10 columns

In [32]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
Tfidf_vect = TfidfVectorizer(max_df=0.8, min_df=2, stop_words='english')
doc_term_matrix = Tfidf_vect.fit_transform(reviews_datasets['Text'].values.astype('U'))
```

In [34]:
```
from sklearn.decomposition import NMF
nmf = NMF(n_components=5, random_state=42)
nmf.fit(doc_term_matrix)
```

```
C:\Users\nihar\Anaconda3\lib\site-packages\sklearn\decomposition\_nmf.py:289: FutureWarni
ng: The 'init' value, when 'init=None' and n_components is less than n_samples and n_feat
ures, will be changed from 'nndsvd' to 'nndsvda' in 1.1 (renaming of 0.26).
  warnings.warn(
```

Out[34]:
```
NMF(n_components=5, random_state=42)
```

In [35]:
```
first_topic = nmf.components_[0]
top_topic = first_topic.argsort()[-10:]
```

In [36]:
```
for i in top_topic_words:
    print(tfidf_vect.get_feature_names_out()[i])
```

```
water
great
just
drink
sugar
good
flavor
taste
like
tea
```

In [40]:
```
for i, topic in enumerate(nmf.components_):
print(f'Top 10 words for topic #{i}:')
print([tfidf_vect.get_feature_names_out()[i] for i in topic.argsort()[-10:]])
print('\n')
```

```
Top 10 words for topic #0:
['really', 'chocolate', 'love', 'flavor', 'just', 'product', 'taste', 'great', 'good', 'l
ike']


Top 10 words for topic #1:
['like', 'keurig', 'roast', 'flavor', 'blend', 'bold', 'strong', 'cups', 'cup', 'coffee']


Top 10 words for topic #2:
['com', 'amazon', 'orange', 'switch', 'water', 'drink', 'soda', 'sugar', 'juice', 'br']


Top 10 words for topic #3:
['bags', 'flavor', 'drink', 'iced', 'earl', 'loose', 'grey', 'teas', 'green', 'tea']


Top 10 words for topic #4:
['old', 'love', 'cat', 'eat', 'treat', 'loves', 'dogs', 'food', 'treats', 'dog']
```

In [42]:
```
topic_values = nmf.transform(doc_term_matrix)
reviews_datasets['Topic'] = topic_values.argmax(axis=1)
reviews_datasets.head()
```

Out[42]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1

In [ ]:
```
```