Topic Modeling

In [1]:
```
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[1]:

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

20000 rows × 10 columns

In [2]:
```
reviews_datasets['Text'][350]
```

Out[2]:
```
'These chocolate covered espresso beans are wonderful!  The chocolate is very dark an
d rich and the "bean" inside is a very delightful blend of flavors with just enough c
affine to really give it a zing.'
```

In [3]:
```
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 [4]:
```
CountVectorizer?
```

In [5]:
```
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 [11]:
```
from sklearn.decomposition import LatentDirichletAllocation
LDA = LatentDirichletAllocation(n_components=5, random_state=42)
LDA.fit(doc_term_matrix)
```

Out[11]:
```
LatentDirichletAllocation(n_components=5, random_state=42)
```

In [14]:
```
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])
```

```
charts
goodies
trusted
cultural
prevents
wallop
repurchasing
tobacco
closed
kneading
```

In [15]:
```
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', 'te
a']


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


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


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', 'coffe
e']
```

In [22]:
```
topic_values = LDA.transform(doc_term_matrix)
topic_values.shape
```

Out[22]:
```
(20000, 5)
```

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

In [29]:
```
reviews_datasets.head(3)
```

Out[29]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Sco
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	

In [38]:
```
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[38]:

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

20000 rows × 10 columns

In [39]:
```
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 [40]:
```
from sklearn.decomposition import NMF
nmf = NMF(n_components=8, random_state=42)
nmf.fit(doc_term_matrix)
```

Out[40]:
```
NMF(n_components=8, random_state=42)
```

In [41]:
```
first_topic = nmf.components_[0]
top_topic_words = first_topic.argsort()[-10:]
```

In [42]:
```
for i in top_topic_words:
    print(Tfidf_vect.get_feature_names_out()[i])
```

```
gluten
just
buy
free
love
amazon
price
good
product
great
```

In [45]:
```
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()[-15:]])
  print('\n')
```

```
Top 10 words for topic #0:
['peanut', 'butter', 'store', 'use', 'time', 'gluten', 'just', 'buy', 'free', 'love',
'amazon', 'price', 'good', 'product', 'great']


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


Top 10 words for topic #2:
['know', 'ingredients', 'review', 'bag', 'pack', 'water', 'product', 'organic', 'hre
f', 'gp', 'www', 'http', 'com', 'amazon', 'br']


Top 10 words for topic #3:
['taste', 'leaves', 'like', 'drink', 'black', 'love', 'bags', 'flavor', 'iced', 'ear
l', 'loose', 'grey', 'teas', 'green', 'tea']


Top 10 words for topic #4:
['ingredients', 'healthy', 'love', 'organic', 'newman', 'old', 'like', 'cat', 'eat',
'loves', 'treat', 'dogs', 'food', 'treats', 'dog']


Top 10 words for topic #5:
['natural', 'tangerine', 'kiwi', 'carbonated', 'fruit', 'flavor', 'switch', 'sweet',
'orange', 'taste', 'soda', 'like', 'sugar', 'drink', 'juice']


Top 10 words for topic #6:
['keurig', 'cups', 'mix', 'flavor', 'taste', 'tried', 'like', 'cookies', 'best', 'dar
k', 'cup', 'milk', 'cocoa', 'hot', 'chocolate']


Top 10 words for topic #7:
['spicy', 'salty', 'love', 'bags', 'flavors', 'snack', 'chip', 'vinegar', 'like', 'ke
ttle', 'potato', 'bag', 'flavor', 'salt', 'chips']
```

In [46]:
```
topic_values = nmf.transform(doc_term_matrix)
reviews_datasets['Topic'] =topic_values.argmax(axis=1)
reviews_datasets.head(3)
```

Out[46]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Sco
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	

In [ ]: