```
# Imports

import os, csv, json
import pandas as pd
from time import sleep

PREPROCESSING = False                  # Set to True if preprocessing is required / False if we already have the summary file

working_directory = '/content/drive/MyDrive/'

'''

This is the code that processes the JSON file
to save time, the final result is written to a file
that is saved in a GIT repository.

If false we bypass the preprocessing and load the file from the repository

The code loads the respective JSON files and creates a pandas dataframe

Unlike the homework - it performs a join on the dataframes to create one file with both product info
and the respective review

We filter for just Nike products then just Nike shoes

We then use TextBlob to determine the sentiment around the reviews

Our final preprocessed file will consist of just Nike shoe reviews with the sentiment noted in a column

'''

if PREPROCESSING:

  # Mount google drive - this is where the JSON product and review files are stored

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

  # Load JSON Input files - only required if there is no processed data file

  review_file       = '%s/reviews_Clothing_Shoes_and_Jewelry.json' % working_directory
  product_file      = '%s/meta_Clothing_Shoes_and_Jewelry.json'    % working_directory

  # Load the json product_file into a pandas dataframe

  loadedjson = open(product_file, 'r')

  count = 0
  allproducts = {}

  for aline in loadedjson:

    aproduct = eval(aline)                      # Not true json - dict printed out in text
    allproducts[aproduct['asin']] = aproduct

    count += 1
    if count % 100000 == 0:
        print(count)

  # end for

  # load into dataframe

  product_df   = pd.DataFrame.from_dict(allproducts, orient='index')

  # See if the product has "Nike" in the category list and if so flag as a Nike product

  #print(product_df.head())

  # Function that takes a list of lists and flattens it until the list has elements and no lists
  # This is required to flatten the "categories" list so we can search just for "Nike"
```

```
from itertools import chain

def flatten(my_list):
  new_list = my_list
  while isinstance(new_list, list):
    try:
      new_list = sum(new_list, [])
    except:
      return new_list
  return my_list

# Function that flattens the "categories" list and returns true if "NIke" is there
# e.g. this is a Nike product

def Nike_Product(category_list):
  categories = flatten(category_list)
  if 'Nike' in categories:
    return True
  else:
    return False

# Apply the is "Nike_Product" function to all products to filter for just Nike entries

product_df['Nike'] = product_df['categories'].apply(Nike_Product)

# Filter product_df where "Nike" is True and put into a new dataframe called nike_products

nike_products = product_df[product_df['Nike'] == True]
nike_products.head()
print(len(nike_products))

# So we all a dataframe of all Nike products -> let's now load reviews

# Load the review_file into a pandas dataframe

loadedjson = open(review_file, 'r')

count = 0
allreviews = {}

for aline in loadedjson:

  areview = eval(aline)                    # Data is not in true json - dict printed out in text
  allreviews[count] = areview              # Use count as key

  count += 1
  if count % 100000 == 0:
      print(count)

# end for

# load all reviews into dataframe

review_df   = pd.DataFrame.from_dict(allreviews, orient='index')

# We are now going to JOIN the Nike Product dataframe with the Review dataframe (using ASIN as key) to create a MERGED dataframe

nike_df = nike_products.merge(review_df, how='left', on='asin')

# There are 23034 reviews of Nike products in multiple categories e.g. shoes / clothes / watches

print("Nike dataframe with all products: ",len(nike_df))        # There are 23034 reviews for all products

# We are going to do some modifications to the Nike dataframe (nike_df)

# Add a column called "is_shoe" to the "nike_df" dataframe if the word "Shoes" is found in the text in the column called "salesRank"

nike_df['is_shoe'] = nike_df['salesRank'].str.contains('Shoes')

# Create a new dataframe called "nike_shoes_df" that consists of records from nike_df only if 'is_shoe' is True

nike_shoes_df = nike_df[nike_df['is_shoe'] == True]

print("Nike dataframe with just shoes: ",len(nike_shoes_df))     # There are 18124 reviews now from the 23034 reviews = 78.6% ~ 80% of
```

```
  # Use textblob to score the Review Text for the Nike Shoes looking for negative sentiment

  from textblob import TextBlob

  def neg_sentiment(text):

    if type(text) != str:
        return False

    if TextBlob(text).sentiment.polarity < 0:
        return True
    else:
        return False

  nike_shoes_df["neg_sentiment"] = nike_shoes_df["reviewText"].apply(neg_sentiment)
  nike_shoes_df["neg_summary"]   = nike_shoes_df["summary"].apply(neg_sentiment)

  # nike_shoes_df.head()

  # Finished Preprocessing - save file

  working_file      = '%s/nike_reviews.csv' % working_directory
  nike_shoes_df.to_csv(working_file, index=False)

else:

  # Preprocessing done - just load finished file

  print("Preprocessing done - loading finished file")
```

```
Preprocessing done - loading finished file
```

```
BERT_TOPIC = True                   # Set to True to run BertTopic code

if BERT_TOPIC:

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

  # Test to see if we have a GPU
  # Our run requires an A100 configuration / CPU with large memory (e.g. 40GB) footprint
  # especially for the multi-model mode where memory usage can be greater > 30GB

  gpu_info = !nvidia-smi
  gpu_info = '\n'.join(gpu_info)
  if gpu_info.find('failed') >= 0:
    print('Not connected to a GPU')
  else:
    print(gpu_info)
```

```
    Mon Dec 11 18:06:13 2023
    +-----------------------------------------------------------------------------+
    | NVIDIA-SMI 525.105.17   Driver Version: 525.105.17   CUDA Version: 12.0     |
    |-------------------------------+----------------------+----------------------+
    | GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
    | Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
    |                               |                      |               MIG M. |
    |===============================+======================+======================|
    |   0  Tesla V100-SXM2...  Off  | 00000000:00:04.0 Off |                    0 |
    | N/A   38C    P0    42W / 300W |   1748MiB / 16384MiB |      0%      Default |
    |                               |                      |                  N/A |
    +-------------------------------+----------------------+----------------------+

    +-----------------------------------------------------------------------------+
    | Processes:                                                                  |
    |  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
    |        ID   ID                                                   Usage      |
    |=============================================================================|
    +-----------------------------------------------------------------------------+
```

```
# Load the preprocessed file from a Github repository

working_file = "https://huggingface.co/spaces/mswhite/Nike_Products/resolve/main/nike_reviews.csv"
nike_df      = pd.read_csv(working_file)


print(len(nike_df))              # There are 18124 reviews of Nike Shoes only
print(nike_df.head())
```

```
18124
         asin                                            related  \
0  B0002164KC                                                NaN
1  B0006NGUE6  {'also_bought': ['B000AYI8R8', 'B006LMIAJO', '...
2  B0006NGUE6  {'also_bought': ['B000AYI8R8', 'B006LMIAJO', '...
3  B0006NGUE6  {'also_bought': ['B000AYI8R8', 'B006LMIAJO', '...
4  B0006NGUE6  {'also_bought': ['B000AYI8R8', 'B006LMIAJO', '...

                                        title  price  \
0                                Nike Talaria 365    NaN
1  Nike Men's Air Rival Golf Shoes (Medium) (11 D...    NaN
2  Nike Men's Air Rival Golf Shoes (Medium) (11 D...    NaN
3  Nike Men's Air Rival Golf Shoes (Medium) (11 D...    NaN
4  Nike Men's Air Rival Golf Shoes (Medium) (11 D...    NaN

            salesRank                                              imUrl  \
0  {'Shoes': 1067625}  http://ecx.images-amazon.com/images/I/3148K7X6...
1    {'Shoes': 45137}  http://ecx.images-amazon.com/images/I/41vL%2Bn...
2    {'Shoes': 45137}  http://ecx.images-amazon.com/images/I/41vL%2Bn...
3    {'Shoes': 45137}  http://ecx.images-amazon.com/images/I/41vL%2Bn...
4    {'Shoes': 45137}  http://ecx.images-amazon.com/images/I/41vL%2Bn...

  brand                                      categories  description  Nike  \
0   NaN  [['Clothing, Shoes & Jewelry', 'N', 'Nike'], [...          NaN  True
1   NaN  [['Clothing, Shoes & Jewelry', 'N', 'Nike'], [...          NaN  True
2   NaN  [['Clothing, Shoes & Jewelry', 'N', 'Nike'], [...          NaN  True
3   NaN  [['Clothing, Shoes & Jewelry', 'N', 'Nike'], [...          NaN  True
4   NaN  [['Clothing, Shoes & Jewelry', 'N', 'Nike'], [...          NaN  True

   ...               reviewerName helpful  \
0  ...  Amgad Okasha "&#34;D&#34;"  [0, 0]
1  ...       1/3 of The O Boyz "M"  [0, 0]
2  ...      A. G. Paterson "jimmy"  [3, 3]
3  ...             Amazon Customer  [0, 0]
4  ...             Amazon Customer  [3, 3]

                                        reviewText overall          summary  \
0  By far, the best pair of shoes I've ever owned...     5.0    Unmatchable.
1  I received these shoes last week. Aside from t...     5.0  Worth the Price
2  a good looking shoe but not to be recommended ...     3.0       golf shoes
3  Very nice and comfortable golf shoes!! This wa...     5.0    Nice shoes!!
4  I bought this shoe for my husband who is norma...     4.0       Great shoe

  unixReviewTime   reviewTime is_shoe  neg_sentiment  neg_summary
0   1.244851e+09  06 13, 2009    True          False        False
1   1.351469e+09  10 29, 2012    True           True        False
2   1.337299e+09  05 18, 2012    True          False        False
3   1.354752e+09   12 6, 2012    True          False        False
4   1.357344e+09   01 5, 2013    True          False        False

[5 rows x 21 columns]
```

```
# Create a dataframe called select_reviews_df where the overall column is rated 3 or less and negative sentiment is True

nike_df["review"]    = nike_df["reviewText"]+" "+nike_df["summary"]         # combine reviewText and summary to form review

select_reviews_df    = nike_df[nike_df['overall'] <= 3]
select_reviews_df    = select_reviews_df[select_reviews_df['neg_sentiment'] == True]
select_reviews_df.head()

allreviewtext = []

for review in select_reviews_df['review']:
  if type(review) == str:
    allreviewtext.append(review)

print("Number of reviews: ",len(allreviewtext))
```

```
Number of reviews:  933
```

```
from sentence_transformers import SentenceTransformer

# Pre-calculate embeddings

embedding_model = SentenceTransformer("all-MiniLM-L6-v2")
embeddings      = embedding_model.encode(allreviewtext, show_progress_bar=True)
```

```
Batches: 100%                                                30/30 [00:00<00:00, 84.05it/s]
```

```
from umap import UMAP

umap_model = UMAP(n_neighbors=15, n_components=5, min_dist=0.0, metric='cosine', random_state=42)


from hdbscan import HDBSCAN

hdbscan_model = HDBSCAN(min_cluster_size=20, metric='euclidean', cluster_selection_method='eom', prediction_data=True)


from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction import text

#vectorizer_model = CountVectorizer(stop_words="english", min_df=2, ngram_range=(1, 2))

#print(type(text.ENGLISH_STOP_WORDS))
#print(len(text.ENGLISH_STOP_WORDS))

my_stop_words    = list(text.ENGLISH_STOP_WORDS.union({"Nike","shoe","shoes"}))                                  # Modify standard stop
#my_stop_words    = list(text.ENGLISH_STOP_WORDS)                                                              # Standard stop words includes

#print(type(my_stop_words))
#print(len(my_stop_words))
#print(my_stop_words)

if "Nike" in my_stop_words:
    print("Nike IS in my_stop_words")
else:
    print("Nike IS NOT in my_stop_words")

vectorizer_model = CountVectorizer(stop_words=my_stop_words, min_df=2, ngram_range=(1, 2))               # Use custom stop word list
```

```
Nike IS in my_stop_words
```

```
from bertopic.representation import KeyBERTInspired, MaximalMarginalRelevance, PartOfSpeech

# KeyBERT
keybert_model = KeyBERTInspired()

# Part-of-Speech
pos_model = PartOfSpeech("en_core_web_sm")

# MMR
mmr_model = MaximalMarginalRelevance(diversity=0.3)
```

```
# All representation models
representation_model = {
    "KeyBERT": keybert_model,
    "MMR": mmr_model,
    "POS": pos_model
}


from bertopic import BERTopic

topic_model = BERTopic(

  # Pipeline models
  embedding_model=embedding_model,
  umap_model=umap_model,
  hdbscan_model=hdbscan_model,
  vectorizer_model=vectorizer_model,
  representation_model=representation_model,

  # Hyperparameters
  top_n_words=10,
  verbose=True
)

topics, probs = topic_model.fit_transform(allreviewtext, embeddings)
```

```
2023-12-11 18:07:48,579 - BERTopic - Dimensionality - Fitting the dimensionality reduction algorithm
2023-12-11 18:07:51,864 - BERTopic - Dimensionality - Completed ✓
2023-12-11 18:07:51,866 - BERTopic - Cluster - Start clustering the reduced embeddings
2023-12-11 18:07:51,896 - BERTopic - Cluster - Completed ✓
2023-12-11 18:07:51,900 - BERTopic - Representation - Extracting topics from clusters using representation models.
2023-12-11 18:07:53,845 - BERTopic - Representation - Completed ✓
```

```
#topic_model.get_topic_info()
freq = topic_model.get_topic_info(); freq.head(20)
```

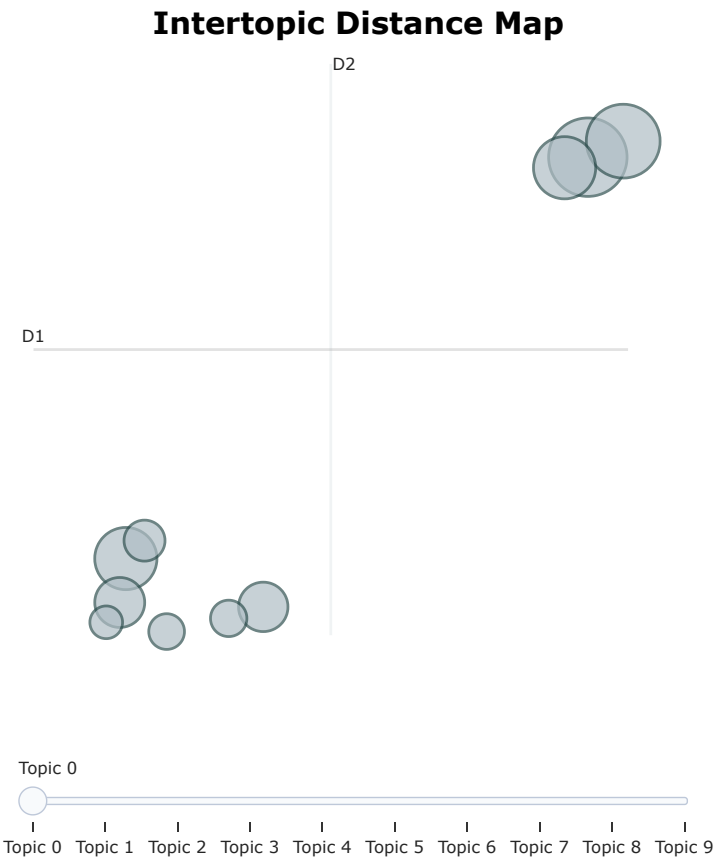
	Topic	Count	Name	Representation	KeyBERT	MMR	POS	Representative_Docs
0	-1	351	-1_nike_like_size_just	[nike, like, size, just, pair, buy, bad, disap...	[nike, sneakers, soles, sole, toes, socks, hee...	[nike, like, size, just, pair, buy, bad, disap...	[nike, size, pair, bad, disappointed, feet, re...	[I've been running for 35 years, almost all wi...
1	0	121	0_size_small_11_wear	[size, small, 11, wear, 10, nike, runs, run, o...	[size 10, size 11, wear size, sizing, size sma...	[size, small, 11, wear, 10, nike, runs, run, o...	[size, small, wear, nike, tight, fit, order, b...	[small not my size my size 11.5 and my order ...
2	1	107	1_size_small_return_ordered	[size, small, return, ordered, order, big, tim...	[sizing, wrong size, sizes, size way, return s...	[size, small, return, ordered, order, big, tim...	[size, small, big, time, sizes, wrong, fit, di...	[SUPER BUMMED, bought these for my son for bal...
					[narrow foot, wide	[narrow,		

```
topic_model.get_topic(1, full=True)
```

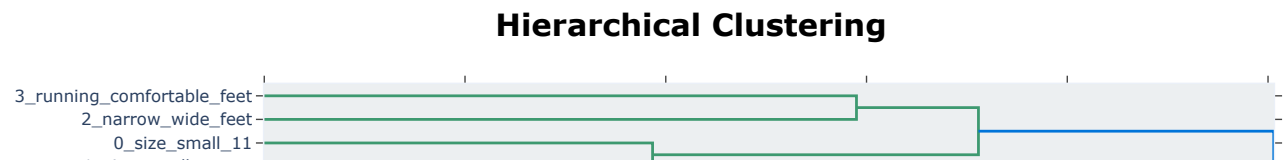
```
{'Main': [('size', 0.0701593070434795),
  ('small', 0.06792945845886877),
  ('return', 0.04256348652616789),
  ('ordered', 0.033917195109861895),
  ('order', 0.02946125301016122),
  ('big', 0.025897987839980863),
  ('time', 0.025506860735961104),
  ('sizes', 0.024474321456080354),
  ('wrong', 0.02366846862141052),
  ('way small', 0.022604517269229126)],
 'KeyBERT': [('sizing', 0.5389854),
  ('wrong size', 0.53335226),
  ('sizes', 0.5299534),
  ('size way', 0.47340178),
  ('return small', 0.45249236),
  ('size', 0.4452215),
```

```
        ('sized', 0.43202212),
        ('small tight', 0.41954643),
        ('small ordered', 0.40102607),
        ('fit', 0.38127196)],
       'MMR': [('size', 0.0701593070434795),
        ('small', 0.06792945845886877),
        ('return', 0.042563486526167 89),
        ('ordered', 0.0339171951098618 95),
        ('order', 0.02946125301016122),
        ('big', 0.025897987839980863),
        ('time', 0.025506860735961104),
        ('sizes', 0.024474321456080354),
        ('wrong', 0.02366846862141052),
        ('way small', 0.022604517269229126)],
       'POS': [('size', 0.0701593070434795),
        ('small', 0.06792945845886877),
        ('big', 0.025897987839980863),
        ('time', 0.025506860735961104),
        ('sizes', 0.024474321456080354),
        ('wrong', 0.02366846862141052),
        ('fit', 0.020707057659431528),
        ('disappointed', 0.019320761076871282),
        ('tight', 0.018371670493900014),
        ('wrong size', 0.0180836138153833)]}
```

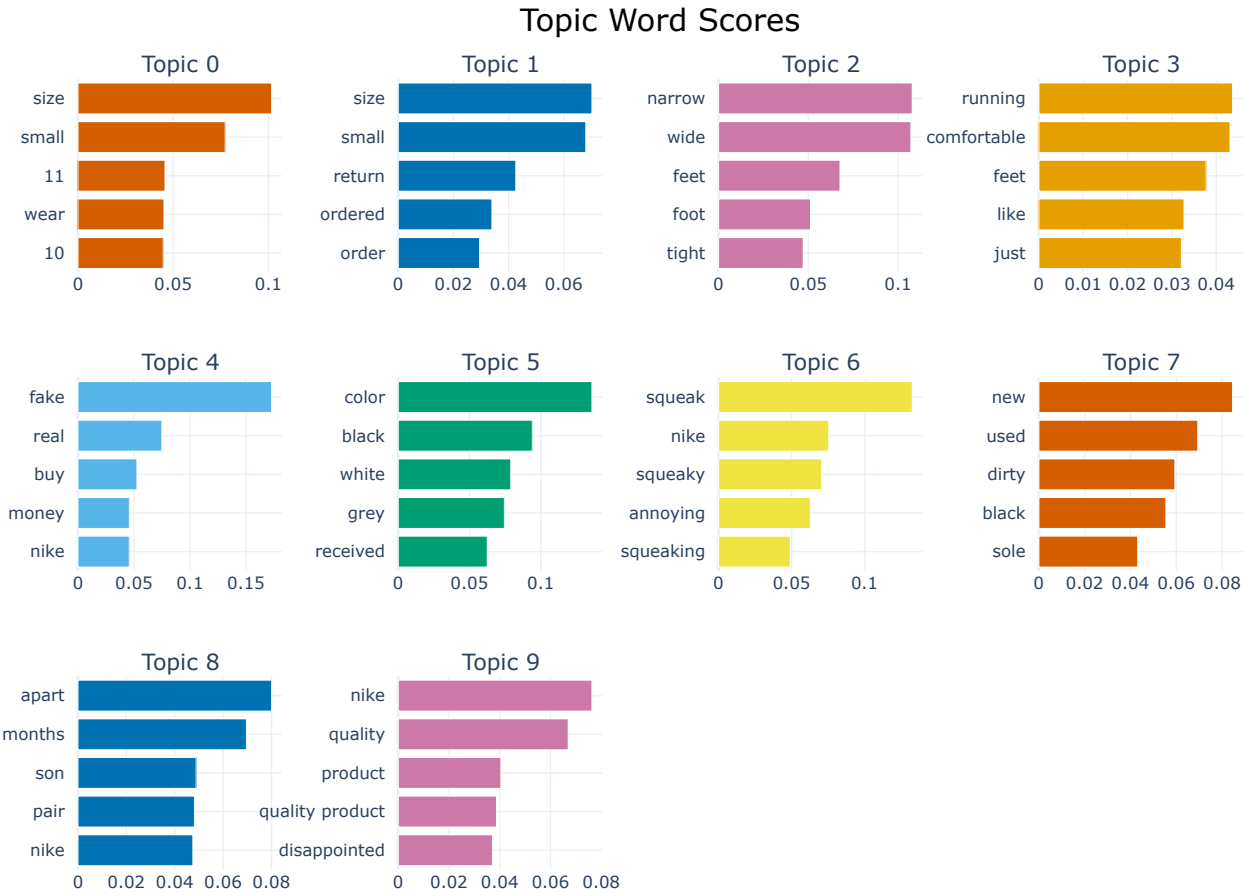
```
topic_model.visualize_topics()
```



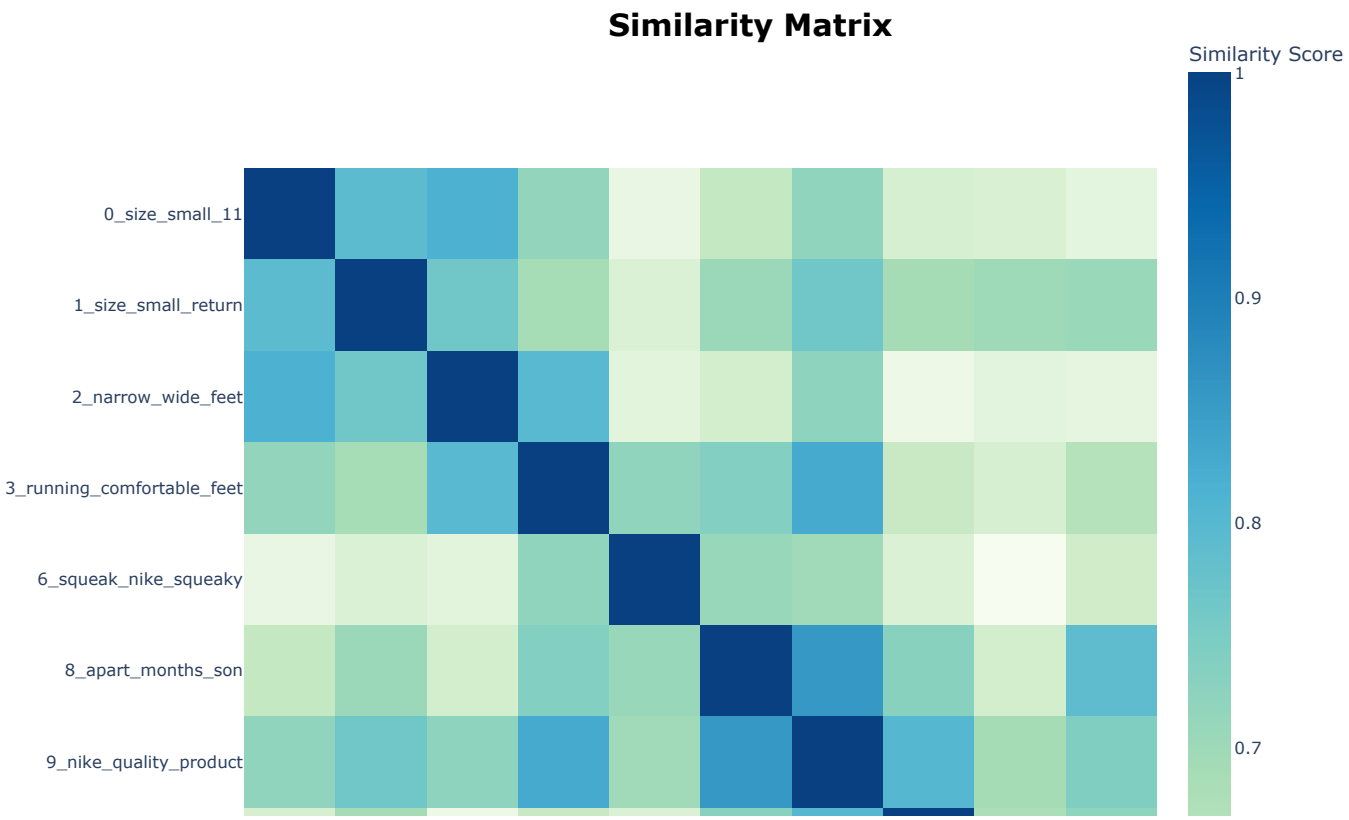

```
topic_model.visualize_hierarchy(top_n_topics=50)
```



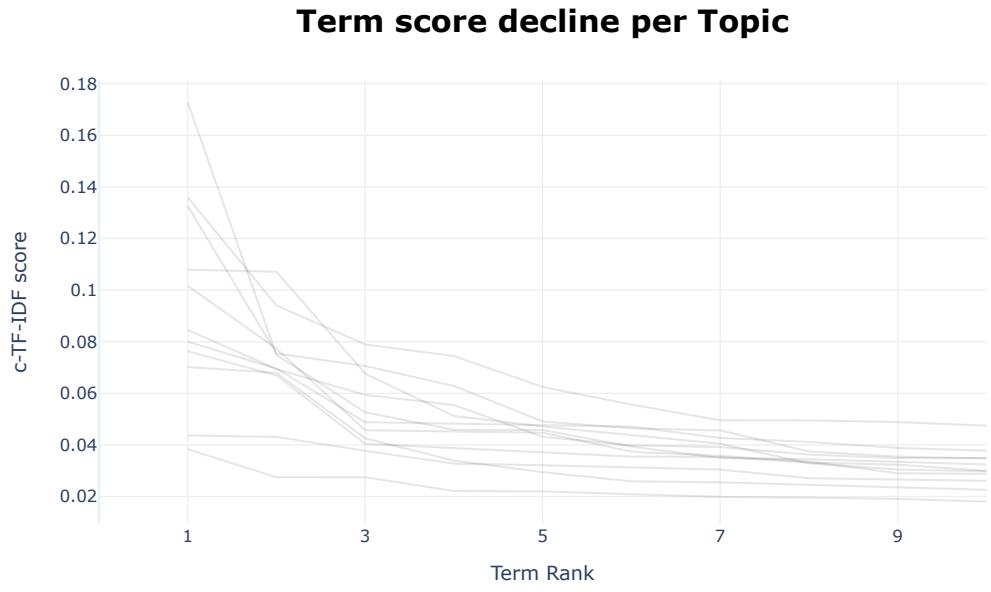

```
topic_model.visualize_barchart(top_n_topics=12)
```




```
topic_model.visualize_heatmap(n_clusters=3, width=1000, height=1000)
```



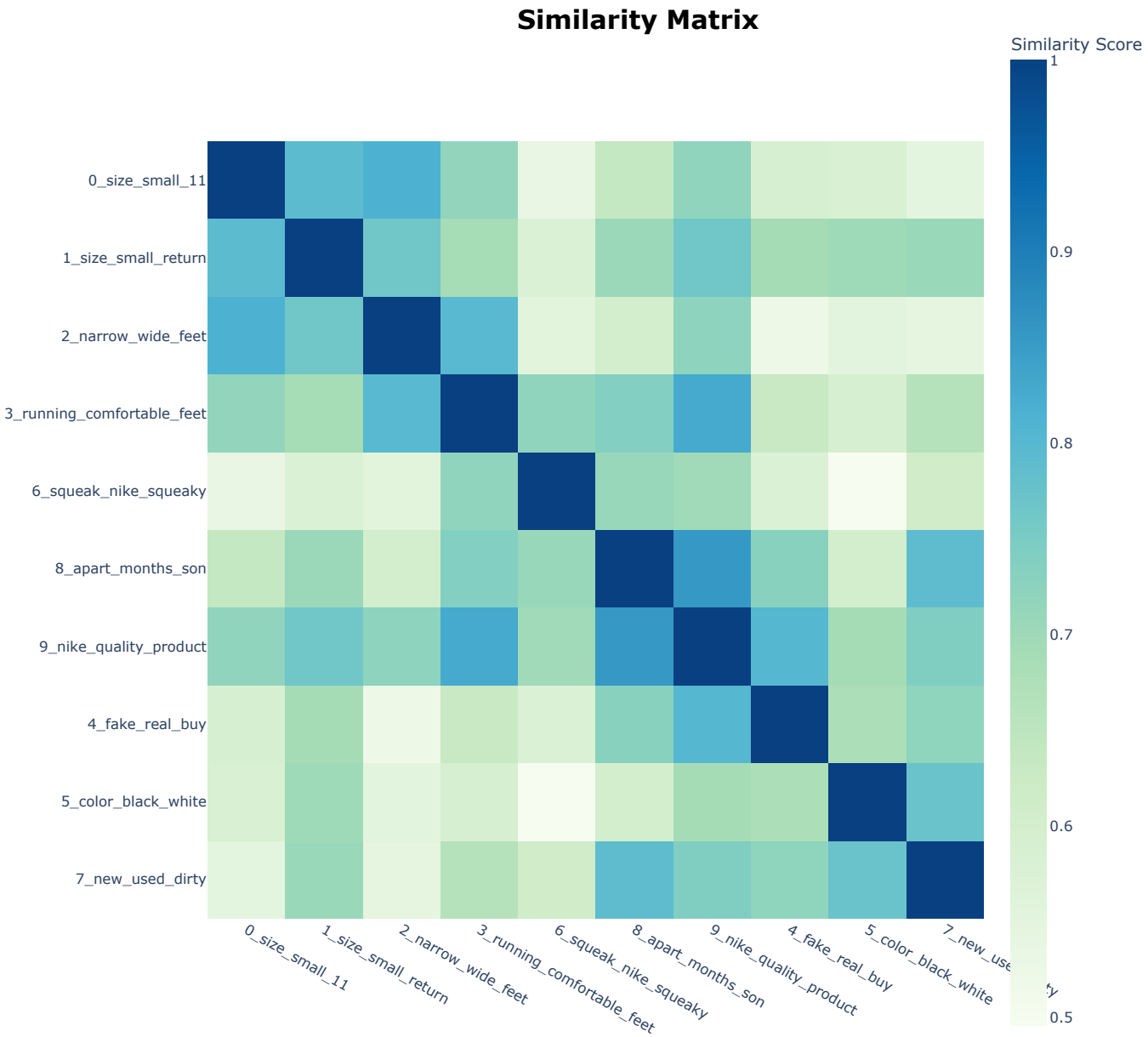

```
topic_model.visualize_term_rank()
```




```
results = topic_model.reduce_topics(allreviewtext, nr_topics="auto")
```

```
     2023-12-11 18:09:20,693 - BERTopic - Topic reduction - Reducing number of topics
     2023-12-11 18:09:22,399 - BERTopic - Topic reduction - Reduced number of topics from 11 to 11
```

```
topic_model.visualize_heatmap(n_clusters=3, width=1000, height=1000)
```


Similarity Matrix
Similarity Score
1
0.9
0.8
0.7
0.6
0.5
0_size_small_11
1_size_small_return
2_narrow_wide_feet
3_running_comfortable_feet
6_squeak_nike_squeaky
8_apart_months_son
9_nike_quality_product
4_fake_real_buy
5_color_black_white
7_new_used_dirty