CWordTM Usage on BBC News

This Jupyter notebook demonstrates how to use the package "CWordTM" on the BBC News:

1. Meta Information Features
2. Utility Features
3. Text Visualization - Word Cloud
4. Text Summarization
5. Topic Modeling - LDA and BERTopic

1. Meta Information Features

In [1]:
```
import cwordtm
from cwordtm import *
```

In [2]:
```
# Show execution time
df = util.load_text("BBC/BBC News Train.csv", timing=True)
```

```
Loading file 'BBC/BBC News Train.csv' ...
Finished 'load_text' in 0.0480 secs
```

In [3]:
```
# Execute and show code
df = util.load_text("BBC/BBC News Train.csv", code=1)
```

```
Loading file 'BBC/BBC News Train.csv' ...

def load_text(filepath, nr=0, info=False):
    """Loads and returns the text from the prescribed file path ('filepath').

    :param filepath: The prescribed filepath from which the text is loaded,
        default to None
    :type filepath: str
    :param nr: The number of rows of text to be loaded; 0 represents all rows,
        default to 0
    :type nr: int, optional
    :param info: The flag whether the dataset information is shown,
        default to False
    :type info: bool, optional
    :return: The collection of text with the prescribed number of rows loaded
    :rtype: pandas.DataFrame
    """

    print("Loading file '%s' ..." %filepath)
    df = pd.read_csv(filepath)
    if nr > 0:
       print("Initial Records:")
       print(df.head(int(nr)))
    if info:
        print("\nDataset Information:")
        df.info()
    return df
```

In [4]:
```
# Show code without execution
df = util.load_text("BBC/BBC News Train.csv", code=2)
```

```
def load_text(filepath, nr=0, info=False):
    """Loads and returns the text from the prescribed file path ('filepath').

    :param filepath: The prescribed filepath from which the text is loaded,
        default to None
    :type filepath: str
    :param nr: The number of rows of text to be loaded; 0 represents all rows,
        default to 0
    :type nr: int, optional
    :param info: The flag whether the dataset information is shown,
        default to False
    :type info: bool, optional
    :return: The collection of text with the prescribed number of rows loaded
    :rtype: pandas.DataFrame
    """

    print("Loading file '%s' ..." %filepath)
    df = pd.read_csv(filepath)
    if nr > 0:
       print("Initial Records:")
       print(df.head(int(nr)))
    if info:
        print("\nDataset Information:")
        df.info()
    return df
```

In [5]:
```
# Add timing and code reveal features to some other function
from importlib_resources import files
files = meta.addin(files)
files(code=2)
```

```
@package_to_anchor
def files(anchor: Optional[Anchor] = None) -> Traversable:
    """
    Get a Traversable resource for an anchor.
    """
    return from_package(resolve(anchor))
```

2. Utility Features

Load BBC News

In [6]:
```
bbc_file = "BBC/BBC News Train.csv"
df = util.load_text(bbc_file, info=True)
```

```
Loading file 'BBC/BBC News Train.csv' ...

Dataset Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1490 entries, 0 to 1489
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   ArticleId  1490 non-null   int64
 1   Text       1490 non-null   object
 2   Category   1490 non-null   object
dtypes: int64(1), object(2)
memory usage: 35.0+ KB
```

Preprocessing Text

In [7]:
```
text_list = util.get_text_list(df.iloc[:500], text_col='Text')
text = util.preprocess_text(text_list)
```

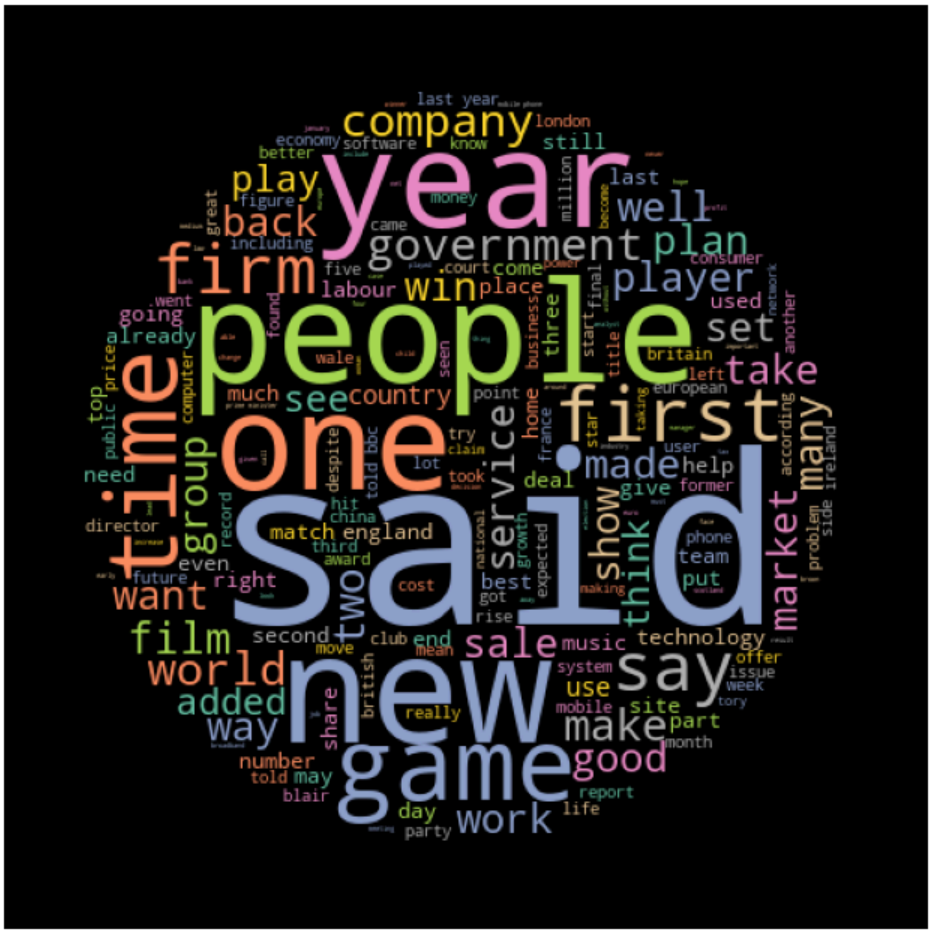
3. Text Visualization - Word Cloud

In [8]:
```
# White background with no image mask
viz.show_wordcloud(text)
```

```
C:\Dev\Anaconda3\envs\aiml\lib\site-packages\wordcloud\wordcloud.py:106: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed two minor releases later. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get_cmap(obj)`` instead.
  self.colormap = plt.cm.get_cmap(colormap)
```

In [9]:
```
# Black background with the prescribed image as the mask
viz.show_wordcloud(text, bg='black', image='images/disc.png')
```

```
C:\Dev\Anaconda3\envs\aiml\lib\site-packages\wordcloud\wordcloud.py:106: MatplotlibDeprecationWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be removed two minor releases later. Use ``matplotlib.colormaps[name]`` or ``matplotlib.colormaps.get_cmap(obj)`` instead.
  self.colormap = plt.cm.get_cmap(colormap)
```



4. Text Summarization

In [10]:
```
news = df.iloc[:5]['Text']  # "df" stores previously loaded text
ta.summary_en(news, sent_len=5)
```

Out[10]:
```
['but ms cooper  who now runs her own consulting business  told a jury in new york on wednesday that external auditors arthur andersen had approved worldcom s accounting in early 2001 and 2002. she said andersen had given a  green light  to the procedures and practices used by worldcom.',
 'cynthia cooper  worldcom s ex-head of internal accounting  alerted directors to irregular accounting practices at the us telecoms giant in 2002. her warnings led to the collapse of the firm following the discovery of an $11bn (£5.7bn) accounting fraud.',
 'prosecution lawyers have argued that mr ebbers orchestrated a series of accounting tricks at worldcom  ordering employees to hide expenses and inflate revenues to meet wall street earnings estimates.',
 'the university of california said the trial in the case is scheduled to begin in october 2006. it joined the lawsuit in december 2001alleging  massive insider trading  and fraud  claiming it had lost $145m on its investments in the company.',
 'the bbc s david willey in rome says one reason for that result is the changeover from the lira to the euro in 2001 which is widely viewed as the biggest reason why their wages and salaries are worth less than they used to be.']
```

5. Topic Modeling

LDA Model

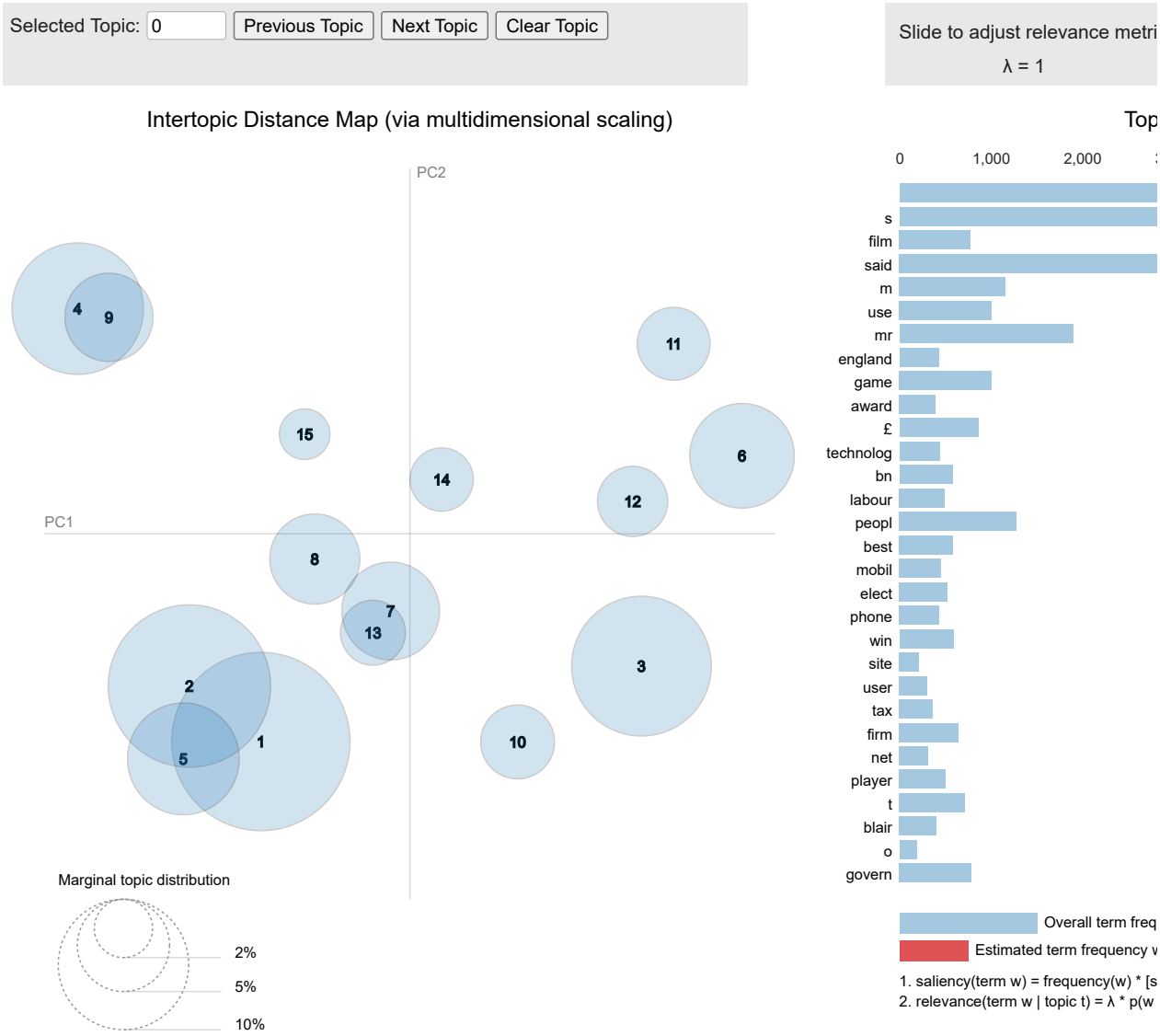
In [11]:
```
doc_file = "BBC/BBC News Train.csv"
lda = tm.lda_process(doc_file, source=1, text_col='Text', eval=True)
```

```
Corpus loaded!
Text preprocessed!
Text trained!
If no visualization is shown,
  you may execute the following commands to show the visualization:
    > import pyLDAvis
    > pyLDAvis.display(lda.vis_data)
Visualization prepared!

Topics from LDA Model:
[(0,
  '0.019*"s" + 0.015*"said" + 0.014*"" + 0.008*"tax" + 0.007*"mr" + 0.005*"t" '
  '+ 0.005*"peopl" + 0.005*"labour" + 0.004*"new" + 0.004*"say"'),
 (1,
  '0.023*"" + 0.015*"said" + 0.013*"use" + 0.011*"net" + 0.011*"peopl" + '
  '0.009*"virus" + 0.009*"e-mail" + 0.008*"s" + 0.007*"broadband" + '
  '0.007*"servic"'),
 (2,
  '0.024*"" + 0.019*"s" + 0.009*"said" + 0.008*"roddick" + 0.006*"nadal" + '
  '0.006*"year" + 0.005*"£" + 0.005*"t" + 0.005*"set" + 0.004*"point"'),
 (3,
  '0.017*"" + 0.015*"said" + 0.013*"use" + 0.012*"peopl" + 0.011*"technolog" + '
  '0.010*"mobil" + 0.009*"s" + 0.008*"music" + 0.008*"phone" + 0.008*"digit"'),
 (4,
  '0.023*"said" + 0.019*"" + 0.019*"s" + 0.018*"mr" + 0.008*"elect" + '
  '0.008*"say" + 0.008*"govern" + 0.007*"parti" + 0.007*"labour" + '
  '0.007*"minist"'),
 (5,
  '0.022*"s" + 0.017*"" + 0.016*"england" + 0.012*"game" + 0.009*"play" + '
  '0.009*"said" + 0.007*"wale" + 0.007*"player" + 0.007*"win" + 0.007*"coach"'),
 (6,
  '0.025*"" + 0.020*"s" + 0.019*"film" + 0.016*"m" + 0.011*"said" + 0.009*"£" '
  '+ 0.007*"year" + 0.006*"director" + 0.006*"actor" + 0.005*"festiv"'),
 (7,
  '0.028*"" + 0.018*"s" + 0.005*"said" + 0.004*"use" + 0.004*"goal" + '
  '0.004*"unit" + 0.004*"new" + 0.004*"camera" + 0.003*"v" + 0.003*"citi"'),
 (8,
  '0.029*"" + 0.026*"s" + 0.013*"film" + 0.012*"best" + 0.010*"award" + '
  '0.009*"year" + 0.009*"m" + 0.008*"said" + 0.007*"star" + 0.006*"won"'),
 (9,
  '0.024*"s" + 0.017*"" + 0.013*"o" + 0.007*"said" + 0.006*"win" + '
  '0.006*"ireland" + 0.006*"m" + 0.006*"tri" + 0.005*"liverpool" + '
  '0.005*"minut"'),
 (10,
  '0.023*"said" + 0.021*"s" + 0.016*"" + 0.009*"compani" + 0.009*"firm" + '
  '0.009*"bn" + 0.008*"bank" + 0.006*"govern" + 0.005*"m" + 0.005*"year"'),
 (11,
  '0.023*"s" + 0.011*"" + 0.011*"said" + 0.007*"t" + 0.007*"game" + '
  '0.006*"player" + 0.006*"arsenal" + 0.005*"play" + 0.005*"unit" + '
  '0.005*"ve"'),
 (12,
  '0.016*"hunt" + 0.016*"said" + 0.013*"" + 0.012*"site" + 0.008*"s" + '
  '0.006*"law" + 0.005*"use" + 0.004*"polic" + 0.004*"mr" + 0.004*"peopl"'),
 (13,
  '0.052*"" + 0.015*"said" + 0.015*"s" + 0.013*"year" + 0.007*"game" + '
  '0.006*"market" + 0.006*"bn" + 0.006*"£" + 0.005*"sale" + 0.005*"new"'),
 (14,
  '0.025*"s" + 0.010*"said" + 0.010*"mr" + 0.009*"" + 0.006*"t" + 0.005*"say" '
  '+ 0.005*"offer" + 0.004*"club" + 0.004*"£" + 0.004*"new"')]

Model Evaluation Scores:
  Coherence: 0.39745546640733204
  Perplexity: -7.840092035719051
  Topic diversity: 0.0007996998649594917
  Topic size distribution: 0.0016755096341803965
```

In [12]:

```python
# LDA Model Visualization
import pyLDAvis
pyLDAvis.display(lda.vis_data)
```

Out[12]:



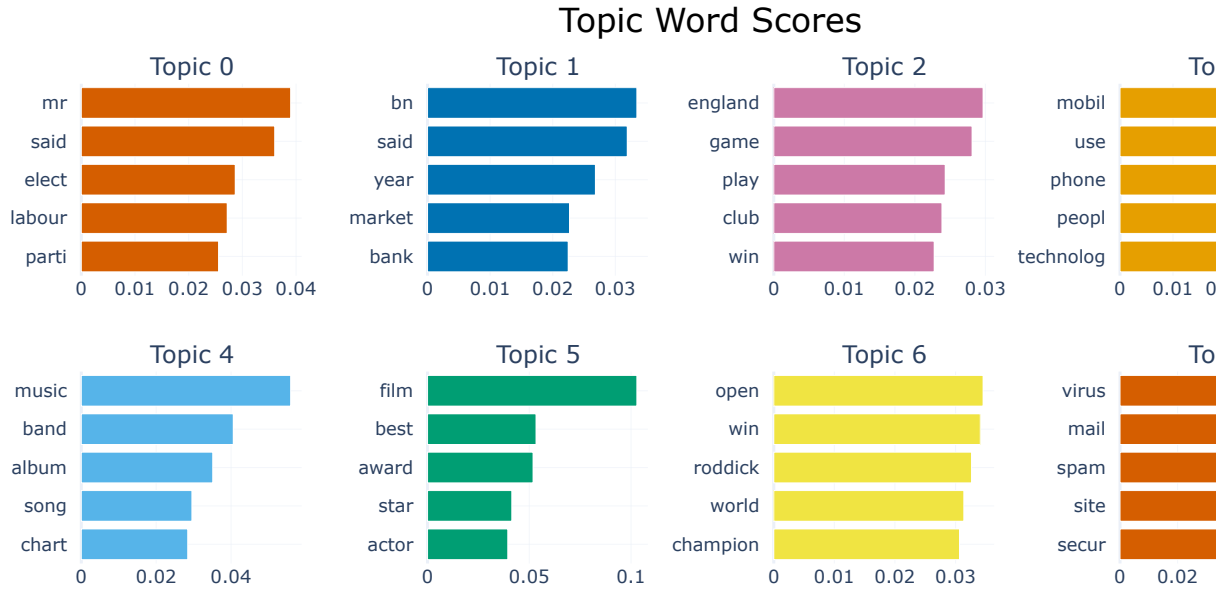
BERTopic Model

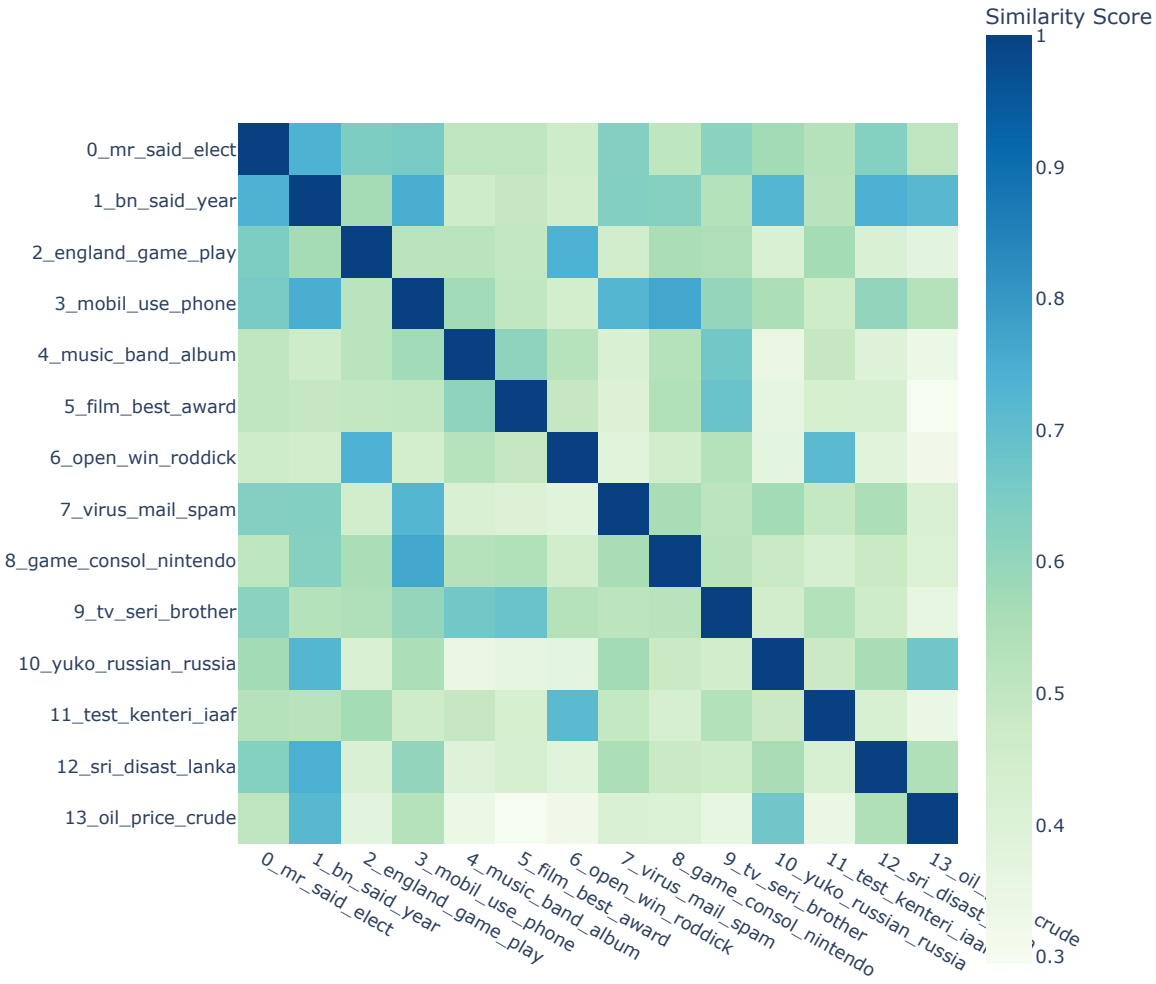
In [13]:

```
btm = tm.btm_process(doc_file, source=1, text_col='Text', eval=True)
```

```
Corpus loaded!
Text preprocessed!
Text trained!

Topics from BERTopic Model:
Topic 0: mr | said | elect | labour | parti | blair | govern | say | minist | tori
Topic 1: bn | said | year | market | bank | growth | economi | firm | rate | sale
Topic 2: england | game | play | club | win | player | ireland | wale | half | team
Topic 3: mobil | use | phone | peopl | technolog | said | servic | digit | gadget | music
Topic 4: music | band | album | song | chart | record | singl | singer | year | perform
Topic 5: film | best | award | star | actor | oscar | nomin | director | actress | won
Topic 6: open | win | roddick | world | champion | year | olymp | seed | match | final
Topic 7: virus | mail | spam | site | secur | user | program | attack | use | softwar
Topic 8: game | consol | nintendo | gamer | xbox | high | soni | dvd | titl | definit
Topic 9: tv | seri | brother | channel | said | evict | televis | celebr | mtv | big
Topic 10: yuko | russian | russia | gazprom | tax | oil | khodorkovski | compani | auction | court
Topic 11: test | kenteri | iaaf | cont | greek | olymp | drug | thanou | athlet | ban
Topic 12: sri | disast | lanka | indonesia | countri | econom | tsunami | aid | damag | thailand
Topic 13: oil | price | crude | barrel | gas | suppli | opec | oecd | land | product

Model Evaluation Scores:
  Coherence: 0.6339203070708426

BERTopic Model Visualization:
```


Intertopic Distance Map
D2
D1
Topic 0
Topic 0
Topic 2
Topic 4
Topic 6
Topic 8
Topic 10
Topic 12


Topic Word Scores
Topic 0
mr
said
elect
labour
parti
0 0.01 0.02 0.03 0.04
Topic 1
bn
said
year
market
bank
0 0.01 0.02 0.03
Topic 2
england
game
play
club
win
0 0.01 0.02 0.03
To
mobil
use
phone
peopl
technolog
0 0.01 0
Topic 4
music
band
album
song
chart
0 0.02 0.04
Topic 5
film
best
award
star
actor
0 0.05 0.1
Topic 6
open
win
roddick
world
champion
0 0.01 0.02 0.03
To
virus
mail
spam
site
secur
0 0.02




```
If no visualization is shown,
  you may execute the following commands one-by-one:
    btm.model.visualize_topics()
    btm.model.visualize_barchart()
    btm.model.visualize_heatmap()
```