[z5476376] Oscar Parrish

In [774…
```
# Necessary package imports

# Dataframe and Modelling
import pandas as pd
import matplotlib.pyplot as plt

# Preprocessing
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
```

In [775…
```
raw_data = pd.read_csv('dataset.tsv', sep='\t')

print(raw_data.head())
```

```
                          artist_name         track_name  release_date  \
0                             loving  the not real lake          2016
1                            incubus    into the summer          2019
2                          reignwolf           hardcore          2016
3                tedeschi trucks band             anyhow          2016
4  lukas nelson and promise of the real  if i started over          2017

   genre                                             lyrics      topic
0   rock  awake know go see time clear world mirror worl...       dark
1   rock  shouldn summer pretty build spill ready overfl...  lifestyle
2  blues  lose deep catch breath think say try break wal...    sadness
3  blues  run bitter taste take rest feel anchor soul pl...    sadness
4  blues  think think different set apart sober mind sym...       dark
```

In [776…
```
# Concat
data = pd.DataFrame({
    'content': raw_data[['artist_name', 'track_name', 'release_date', 'genre', '
    'topic': raw_data['topic']
})

print(data.head())
```

```
                                             content      topic
0  loving the not real lake 2016 rock awake know ...       dark
1  incubus into the summer 2019 rock shouldn summ...  lifestyle
2  reignwolf hardcore 2016 blues lose deep catch ...    sadness
3  tedeschi trucks band anyhow 2016 blues run bit...    sadness
4  lukas nelson and promise of the real if i star...       dark
```

In [777…
```
for line in data['content']:
    print(re.search(r'[^\w\s]', line))
    print(re.search(r'\.', line))
```

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(40, 41), match="'">
None
None
None
None
None
None
None
<re.Match object; span=(28, 29), match='('>
<re.Match object; span=(33, 34), match='.'>
None
None
None
None
None
None
None
None
<re.Match object; span=(27, 28), match='('>
<re.Match object; span=(32, 33), match='.'>
<re.Match object; span=(14, 15), match=','>
<re.Match object; span=(18, 19), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(1, 2), match='-'>
None
<re.Match object; span=(2, 3), match='.'>
<re.Match object; span=(2, 3), match='.'>
<re.Match object; span=(24, 25), match='('>
None
<re.Match object; span=(21, 22), match='&'>
None
None
None
None
None
```

```
None
None
None
None
<re.Match object; span=(8, 9), match='&'>
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(18, 19), match='('>
<re.Match object; span=(23, 24), match='.'>
None
None
None
None
None
None
<re.Match object; span=(15, 16), match='.'>
<re.Match object; span=(15, 16), match='.'>
None
None
None
None
None
None
None
None
<re.Match object; span=(25, 26), match='('>
None
<re.Match object; span=(13, 14), match="'">
None
None
None
None
None
<re.Match object; span=(23, 24), match='('>
<re.Match object; span=(28, 29), match='.'>
None
None
None
None
<re.Match object; span=(3, 4), match="'">
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
<re.Match object; span=(21, 22), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(31, 32), match=','>
None
None
None
<re.Match object; span=(15, 16), match="'">
None
None
None
<re.Match object; span=(17, 18), match='('>
None
None
None
<re.Match object; span=(1, 2), match="'">
None
None
None
None
None
<re.Match object; span=(16, 17), match='&'>
None
<re.Match object; span=(21, 22), match="'">
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(37, 38), match='('>
None
None
None
<re.Match object; span=(30, 31), match='.'>
<re.Match object; span=(30, 31), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(23, 24), match='('>
<re.Match object; span=(28, 29), match='.'>
None
None
None
None
None
None
<re.Match object; span=(19, 20), match='('>
<re.Match object; span=(26, 27), match='.'>
None
None
None
None
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(1, 2), match='.'>
None
None
```

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(27, 28), match='('>
None
<re.Match object; span=(3, 4), match="'">
None
<re.Match object; span=(18, 19), match="'">
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(11, 12), match=','>
None
None
None
None
None
<re.Match object; span=(8, 9), match='!'>
None
None
None
None
None
None
None
<re.Match object; span=(22, 23), match='('>
None
None
None
None
None
<re.Match object; span=(9, 10), match='.'>
<re.Match object; span=(9, 10), match='.'>
<re.Match object; span=(13, 14), match='.'>
<re.Match object; span=(13, 14), match='.'>
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(19, 20), match='.'>
<re.Match object; span=(19, 20), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(18, 19), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(25, 26), match="'">
None
None
None
None
None
None
None
<re.Match object; span=(6, 7), match='&'>
None
None
None
None
None
None
None
```

```
<re.Match object; span=(35, 36), match="'">
None
None
None
<re.Match object; span=(18, 19), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(15, 16), match='&'>
None
<re.Match object; span=(23, 24), match='('>
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(14, 15), match='&'>
None
None
None
<re.Match object; span=(8, 9), match='&'>
None
None
None
<re.Match object; span=(15, 16), match='&'>
None
<re.Match object; span=(5, 6), match='&'>
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
<re.Match object; span=(20, 21), match='&'>
None
None
None
None
None
None
None
None
None
<re.Match object; span=(18, 19), match='('>
<re.Match object; span=(23, 24), match='.'>
None
None
<re.Match object; span=(23, 24), match='('>
None
None
None
None
None
<re.Match object; span=(12, 13), match="'">
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(20, 21), match='('>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(15, 16), match='.'>
<re.Match object; span=(15, 16), match='.'>
None
None
```

```
None
None
<re.Match object; span=(18, 19), match='('>
None
<re.Match object; span=(18, 19), match='.'>
<re.Match object; span=(18, 19), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(22, 23), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
None
None
<re.Match object; span=(27, 28), match='('>
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(4, 5), match='•'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(16, 17), match="'">
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(16, 17), match="'">
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(21, 22), match="'">
None
None
None
```

```
None
None
<re.Match object; span=(18, 19), match='&'>
None
None
None
None
None
None
None
<re.Match object; span=(7, 8), match='&'>
None
None
None
None
None
None
None
<re.Match object; span=(30, 31), match="'">
None
None
None
None
None
None
None
<re.Match object; span=(44, 45), match='('>
None
None
None
<re.Match object; span=(19, 20), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(1, 2), match='-'>
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(42, 43), match='('>
<re.Match object; span=(47, 48), match='.'>
None
None
<re.Match object; span=(18, 19), match="'">
None
<re.Match object; span=(20, 21), match="'">
None
<re.Match object; span=(16, 17), match='('>
<re.Match object; span=(23, 24), match='.'>
None
None
None
None
None
None
<re.Match object; span=(14, 15), match="'">
None
<re.Match object; span=(21, 22), match='('>
<re.Match object; span=(26, 27), match='.'>
None
None
None
None
<re.Match object; span=(17, 18), match="'">
None
None
None
<re.Match object; span=(24, 25), match='('>
<re.Match object; span=(29, 30), match='.'>
None
None
None
None
<re.Match object; span=(22, 23), match='('>
<re.Match object; span=(27, 28), match='.'>
None
None
None
None
<re.Match object; span=(14, 15), match=','>
<re.Match object; span=(18, 19), match='.'>
None
None
<re.Match object; span=(17, 18), match="'">
None
<re.Match object; span=(33, 34), match='('>
<re.Match object; span=(38, 39), match='.'>
```

```
<re.Match object; span=(18, 19), match='('>
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(2, 3), match='.'>
<re.Match object; span=(2, 3), match='.'>
<re.Match object; span=(13, 14), match='.'>
<re.Match object; span=(13, 14), match='.'>
None
None
None
None
<re.Match object; span=(40, 41), match='('>
<re.Match object; span=(45, 46), match='.'>
<re.Match object; span=(30, 31), match='('>
None
None
None
<re.Match object; span=(2, 3), match='-'>
None
None
None
None
None
<re.Match object; span=(16, 17), match='#'>
None
None
None
None
None
<re.Match object; span=(12, 13), match="'">
None
None
None
None
None
<re.Match object; span=(15, 16), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
<re.Match object; span=(21, 22), match='/'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(8, 9), match='!'>
None
None
None
<re.Match object; span=(1, 2), match='-'>
<re.Match object; span=(32, 33), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(13, 14), match="'">
None
<re.Match object; span=(33, 34), match='('>
None
None
None
None
None
<re.Match object; span=(8, 9), match='&'>
None
<re.Match object; span=(22, 23), match='('>
None
None
None
<re.Match object; span=(15, 16), match='&'>
None
None
None
```

```
None
None
None
None
<re.Match object; span=(16, 17), match="'">
None
None
None
<re.Match object; span=(15, 16), match='&'>
None
None
None
<re.Match object; span=(11, 12), match="'">
None
None
None
None
None
None
None
None
None
<re.Match object; span=(22, 23), match='/'>
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(21, 22), match='('>
None
None
None
None
None
None
None
<re.Match object; span=(38, 39), match='('>
<re.Match object; span=(43, 44), match='.'>
<re.Match object; span=(40, 41), match='('>
<re.Match object; span=(45, 46), match='.'>
<re.Match object; span=(18, 19), match='('>
None
None
None
<re.Match object; span=(4, 5), match="'">
None
None
None
None
None
None
None
<re.Match object; span=(23, 24), match="'">
None
```

```
None
None
None
None
None
None
None
None
<re.Match object; span=(11, 12), match="'">
None
None
None
<re.Match object; span=(2, 3), match='.'>
<re.Match object; span=(2, 3), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(2, 3), match='/'>
None
None
None
None
None
None
None
<re.Match object; span=(13, 14), match='.'>
<re.Match object; span=(13, 14), match='.'>
None
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(35, 36), match='/'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(7, 8), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(20, 21), match=','>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(6, 7), match='&'>
None
```

```
None
None
<re.Match object; span=(3, 4), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(4, 5), match='•'>
None
None
None
None
None
<re.Match object; span=(17, 18), match="'">
None
None
None
None
None
<re.Match object; span=(28, 29), match=':'>
<re.Match object; span=(46, 47), match='.'>
None
None
None
None
None
None
None
None
<re.Match object; span=(16, 17), match='('>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(28, 29), match='('>
<re.Match object; span=(33, 34), match='.'>
<re.Match object; span=(21, 22), match="'">
None
```

```
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(1, 2), match='.'>
None
None
None
None
None
None
<re.Match object; span=(25, 26), match='('>
None
None
None
None
None
None
None
<re.Match object; span=(28, 29), match=','>
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(20, 21), match="'">
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(5, 6), match='!'>
None
None
None
None
None
None
None
<re.Match object; span=(21, 22), match=','>
None
<re.Match object; span=(11, 12), match="'">
None
None
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
<re.Match object; span=(4, 5), match='+'>
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(5, 6), match='!'>
None
<re.Match object; span=(24, 25), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(21, 22), match='('>
<re.Match object; span=(26, 27), match='.'>
None
None
<re.Match object; span=(12, 13), match='&'>
None
<re.Match object; span=(5, 6), match='!'>
None
None
None
None
None
None
None
```

```
<re.Match object; span=(32, 33), match=','>
None
None
None
<re.Match object; span=(1, 2), match='+'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(4, 5), match='+'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(11, 12), match=','>
<re.Match object; span=(15, 16), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(8, 9), match='!'>
None
<re.Match object; span=(25, 26), match='?'>
None
None
None
```

```
None
None
None
None
<re.Match object; span=(28, 29), match='('>
<re.Match object; span=(33, 34), match='.'>
None
None
None
None
<re.Match object; span=(19, 20), match="'">
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(29, 30), match="'">
None
None
None
None
None
None
None
None
None
<re.Match object; span=(25, 26), match='-'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
None
None
<re.Match object; span=(18, 19), match='-'>
None
None
None
None
None
<re.Match object; span=(21, 22), match="'">
None
None
None
None
None
<re.Match object; span=(18, 19), match='-'>
None
None
None
None
None
None
None
<re.Match object; span=(29, 30), match="'">
None
None
None
<re.Match object; span=(5, 6), match='!'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(21, 22), match='’'>
None
None
None
None
None
None
None
<re.Match object; span=(24, 25), match='’'>
None
None
None
<re.Match object; span=(19, 20), match='('>
<re.Match object; span=(24, 25), match='.'>
```

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(10, 11), match='.'>
<re.Match object; span=(10, 11), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(14, 15), match='-'>
None
None
None
<re.Match object; span=(6, 7), match="'">
<re.Match object; span=(17, 18), match='.'>
<re.Match object; span=(25, 26), match='('>
<re.Match object; span=(30, 31), match='.'>
None
None
None
None
None
None
<re.Match object; span=(31, 32), match='('>
<re.Match object; span=(36, 37), match='.'>
<re.Match object; span=(23, 24), match="'">
None
None
None
```

```
None
None
None
None
None
None
None
None
<re.Match object; span=(23, 24), match=','>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(1, 2), match='.'>
None
None
None
None
None
None
<re.Match object; span=(35, 36), match='&'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(22, 23), match="'">
<re.Match object; span=(35, 36), match='.'>
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(20, 21), match="'">
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
<re.Match object; span=(1, 2), match="'">
None
None
None
None
None
None
None
None
None
<re.Match object; span=(20, 21), match=',>
None
None
None
None
None
None
None
<re.Match object; span=(39, 40), match="'">
None
None
None
<re.Match object; span=(28, 29), match='('>
<re.Match object; span=(33, 34), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(15, 16), match='-'>
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(13, 14), match='.'>
<re.Match object; span=(13, 14), match='.'>
<re.Match object; span=(15, 16), match='&'>
None
<re.Match object; span=(3, 4), match='-'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(10, 11), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(34, 35), match='('>
None
None
None
<re.Match object; span=(4, 5), match='&'>
None
None
None
None
None
<re.Match object; span=(34, 35), match='('>
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(6, 7), match='['>
None
None
None
<re.Match object; span=(5, 6), match='-'>
None
None
None
None
None
None
None
<re.Match object; span=(30, 31), match="'">
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(2, 3), match='.'>
<re.Match object; span=(2, 3), match='.'>
None
None
<re.Match object; span=(28, 29), match="'">
None
<re.Match object; span=(15, 16), match='’'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(26, 27), match='('>
None
None
None
None
None
<re.Match object; span=(24, 25), match='('>
<re.Match object; span=(29, 30), match='.'>
<re.Match object; span=(15, 16), match="'">
None
<re.Match object; span=(26, 27), match='('>
None
None
None
<re.Match object; span=(5, 6), match='!'>
None
<re.Match object; span=(18, 19), match="'">
None
<re.Match object; span=(6, 7), match='&'>
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(29, 30), match='('>
<re.Match object; span=(34, 35), match='.'>
None
None
None
None
<re.Match object; span=(10, 11), match="'">
None
None
None
```

```
None
None
<re.Match object; span=(19, 20), match="'">
None
None
None
None
None
<re.Match object; span=(23, 24), match=','>
None
None
None
None
None
None
None
<re.Match object; span=(20, 21), match='.'>
<re.Match object; span=(20, 21), match='.'>
None
None
<re.Match object; span=(29, 30), match='('>
<re.Match object; span=(34, 35), match='.'>
None
None
<re.Match object; span=(19, 20), match='/'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(17, 18), match="'">
None
None
None
<re.Match object; span=(25, 26), match='('>
<re.Match object; span=(30, 31), match='.'>
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(22, 23), match="'">
None
None
None
None
None
<re.Match object; span=(3, 4), match="'">
None
None
None
None
None
<re.Match object; span=(18, 19), match='('>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
<re.Match object; span=(18, 19), match=','>
None
<re.Match object; span=(9, 10), match='%'>
None
<re.Match object; span=(18, 19), match='’'>
None
None
None
None
None
<re.Match object; span=(30, 31), match='?'>
None
None
None
<re.Match object; span=(26, 27), match='('>
None
None
None
None
None
None
None
<re.Match object; span=(21, 22), match=','>
None
None
None
<re.Match object; span=(23, 24), match='('>
None
None
None
<re.Match object; span=(3, 4), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(18, 19), match="'">
None
None
None
<re.Match object; span=(19, 20), match='('>
<re.Match object; span=(24, 25), match='.'>
None
None
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(19, 20), match="'">
None
None
None
None
None
<re.Match object; span=(16, 17), match="'">
None
None
None
<re.Match object; span=(13, 14), match='('>
<re.Match object; span=(18, 19), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(44, 45), match='?'>
None
None
None
None
None
<re.Match object; span=(13, 14), match='('>
None
None
None
None
None
<re.Match object; span=(24, 25), match='('>
<re.Match object; span=(29, 30), match='.'>
None
None
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(22, 23), match='/'>
None
None
None
None
None
None
None
<re.Match object; span=(28, 29), match='('>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(16, 17), match="'">
None
None
None
<re.Match object; span=(21, 22), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(4, 5), match='•'>
None
<re.Match object; span=(1, 2), match="'">
None
None
None
None
None
```

```
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(28, 29), match='('>
<re.Match object; span=(33, 34), match='.'>
None
None
<re.Match object; span=(14, 15), match="'">
None
None
None
<re.Match object; span=(4, 5), match='•'>
None
None
None
None
None
None
None
None
None
<re.Match object; span=(17, 18), match='&'>
None
None
None
None
None
<re.Match object; span=(26, 27), match='('>
<re.Match object; span=(31, 32), match='.'>
None
None
None
None
None
None
None
None
None
<re.Match object; span=(28, 29), match='('>
None
None
None
None
None
<re.Match object; span=(6, 7), match='&'>
<re.Match object; span=(26, 27), match='.'>
None
None
None
None
None
None
None
None
None
```

```
None
None
<re.Match object; span=(17, 18), match="'">
None
<re.Match object; span=(8, 9), match='$'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(25, 26), match='('>
<re.Match object; span=(30, 31), match='.'>
<re.Match object; span=(15, 16), match='('>
<re.Match object; span=(20, 21), match='.'>
<re.Match object; span=(14, 15), match='.'>
<re.Match object; span=(14, 15), match='.'>
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(18, 19), match="'">
None
<re.Match object; span=(23, 24), match='&'>
None
None
None
None
None
None
None
<re.Match object; span=(17, 18), match='&'>
None
None
None
<re.Match object; span=(2, 3), match='-'>
<re.Match object; span=(21, 22), match='.'>
None
None
<re.Match object; span=(8, 9), match='&'>
None
```

```
None
None
<re.Match object; span=(16, 17), match='&'>
None
None
None
None
None
<re.Match object; span=(21, 22), match='('>
<re.Match object; span=(26, 27), match='.'>
None
None
None
None
<re.Match object; span=(1, 2), match='-'>
None
<re.Match object; span=(10, 11), match='('>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(26, 27), match=','>
None
None
None
<re.Match object; span=(13, 14), match="'">
None
None
None
None
None
None
None
<re.Match object; span=(21, 22), match="'">
None
<re.Match object; span=(8, 9), match='&'>
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
<re.Match object; span=(7, 8), match='&'>
None
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(23, 24), match="'">
None
None
None
None
None
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(1, 2), match='.'>
None
None
<re.Match object; span=(22, 23), match='?'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(17, 18), match="'">
None
<re.Match object; span=(5, 6), match='!'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(17, 18), match="'">
None
<re.Match object; span=(17, 18), match='?'>
None
```

```
None
None
<re.Match object; span=(23, 24), match='('>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(26, 27), match='.'>
<re.Match object; span=(26, 27), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(20, 21), match='’'>
None
<re.Match object; span=(17, 18), match='('>
<re.Match object; span=(22, 23), match='.'>
<re.Match object; span=(13, 14), match='('>
<re.Match object; span=(18, 19), match='.'>
```

```
None
None
None
None
None
None
<re.Match object; span=(24, 25), match='('>
None
None
None
None
None
None
None
<re.Match object; span=(25, 26), match='('>
<re.Match object; span=(30, 31), match='.'>
<re.Match object; span=(17, 18), match='’'>
None
None
None
<re.Match object; span=(15, 16), match="'">
None
None
None
None
None
None
None
<re.Match object; span=(8, 9), match='('>
<re.Match object; span=(13, 14), match='.'>
<re.Match object; span=(17, 18), match='’'>
None
None
None
<re.Match object; span=(21, 22), match=','>
None
None
None
<re.Match object; span=(26, 27), match='-'>
None
<re.Match object; span=(5, 6), match='!'>
None
None
None
None
None
None
None
<re.Match object; span=(8, 9), match='!'>
None
<re.Match object; span=(8, 9), match='&'>
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
<re.Match object; span=(19, 20), match='&'>
None
None
None
<re.Match object; span=(15, 16), match='&'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(10, 11), match=':'>
None
<re.Match object; span=(13, 14), match='.'>
<re.Match object; span=(13, 14), match='.'>
None
None
None
None
<re.Match object; span=(22, 23), match='('>
<re.Match object; span=(27, 28), match='.'>
None
None
None
None
<re.Match object; span=(33, 34), match='?'>
None
None
None
None
None
None
None
```

```
None
None
<re.Match object; span=(5, 6), match='&'>
None
None
None
None
None
None
None
None
None
<re.Match object; span=(27, 28), match='('>
None
None
None
None
None
<re.Match object; span=(8, 9), match='&'>
None
None
None
None
None
<re.Match object; span=(23, 24), match='.'>
<re.Match object; span=(23, 24), match='.'>
None
None
None
None
<re.Match object; span=(1, 2), match='!'>
None
<re.Match object; span=(27, 28), match='('>
<re.Match object; span=(32, 33), match='.'>
<re.Match object; span=(14, 15), match='.'>
<re.Match object; span=(14, 15), match='.'>
None
None
<re.Match object; span=(23, 24), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
<re.Match object; span=(28, 29), match=','>
None
<re.Match object; span=(24, 25), match='&'>
None
<re.Match object; span=(29, 30), match="'">
None
None
None
None
None
<re.Match object; span=(25, 26), match="'">
None
None
None
<re.Match object; span=(8, 9), match="'">
None
<re.Match object; span=(25, 26), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(5, 6), match='!'>
None
None
None
<re.Match object; span=(13, 14), match='&'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(22, 23), match='.'>
<re.Match object; span=(22, 23), match='.'>
None
None
None
None
```

```
None
None
None
None
None
None
<re.Match object; span=(1, 2), match='-'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(5, 6), match='&'>
None
None
None
None
None
<re.Match object; span=(8, 9), match='$'>
None
<re.Match object; span=(24, 25), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
<re.Match object; span=(25, 26), match='('>
None
None
None
None
None
None
None
<re.Match object; span=(16, 17), match='&'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(26, 27), match=','>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
<re.Match object; span=(8, 9), match='&'>
None
None
None
<re.Match object; span=(20, 21), match="'">
None
<re.Match object; span=(14, 15), match=','>
<re.Match object; span=(18, 19), match='.'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(20, 21), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(8, 9), match='&'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(32, 33), match="'">
None
None
None
None
None
None
None
```

```
None
None
None
None
None
None
<re.Match object; span=(15, 16), match="'">
None
<re.Match object; span=(11, 12), match=','>
<re.Match object; span=(15, 16), match='.'>
None
None
<re.Match object; span=(20, 21), match="'">
None
<re.Match object; span=(22, 23), match="'">
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(15, 16), match='&'>
None
None
None
None
None
None
None
<re.Match object; span=(30, 31), match='('>
None
None
None
<re.Match object; span=(5, 6), match='!'>
None
<re.Match object; span=(17, 18), match="'">
None
<re.Match object; span=(27, 28), match='('>
None
None
None
None
None
<re.Match object; span=(1, 2), match='!'>
None
None
None
<re.Match object; span=(20, 21), match='-'>
None
None
None
<re.Match object; span=(10, 11), match='.'>
<re.Match object; span=(10, 11), match='.'>
<re.Match object; span=(2, 3), match="'">
None
```

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(27, 28), match="'">
None
<re.Match object; span=(19, 20), match='('>
<re.Match object; span=(24, 25), match='.'>
<re.Match object; span=(18, 19), match='.'>
<re.Match object; span=(18, 19), match='.'>
<re.Match object; span=(26, 27), match='’'>
None
None
None
None
None
None
None
None
None
<re.Match object; span=(13, 14), match="'">
None
None
None
None
None
None
None
<re.Match object; span=(32, 33), match='?'>
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

```
None
None
<re.Match object; span=(18, 19), match='&'>
None
None
None
None
None
<re.Match object; span=(22, 23), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(11, 12), match='’'>
None
<re.Match object; span=(18, 19), match='/'>
None
<re.Match object; span=(20, 21), match='&'>
None
<re.Match object; span=(15, 16), match='&'>
None
None
None
None
None
<re.Match object; span=(16, 17), match='-'>
None
None
None
<re.Match object; span=(3, 4), match='-'>
None
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(1, 2), match='.'>
None
None
None
None
None
None
```

```
None
None
None
None
<re.Match object; span=(29, 30), match='!'>
None
None
None
<re.Match object; span=(23, 24), match="'">
None
None
None
None
None
<re.Match object; span=(49, 50), match='('>
<re.Match object; span=(56, 57), match='.'>
<re.Match object; span=(19, 20), match='’'>
None
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(1, 2), match='.'>
<re.Match object; span=(10, 11), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(25, 26), match='('>
<re.Match object; span=(30, 31), match='.'>
None
None
None
None
<re.Match object; span=(14, 15), match="'">
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
<re.Match object; span=(18, 19), match="'">
None
None
None
```

In [778…
```
# Get class distribution
print('----INSTANCE COUNT----')
print(f'dark: {len(data[data['topic'] == 'dark'])}')
print(f'lifestyle: {len(data[data['topic'] == 'lifestyle'])}')
print(f'sadness: {len(data[data['topic'] == 'sadness'])}')
print(f'personal: {len(data[data['topic'] == 'personal'])}')
print(f'emotion: {len(data[data['topic'] == 'emotion'])}')
```

```
----INSTANCE COUNT----
dark: 490
lifestyle: 205
sadness: 376
personal: 347
emotion: 82
```

In [779…
```
# Drop duplicates and missing values
print(data.size)

data = data.drop_duplicates()
data = data.dropna()

print(data.size)
```

```
3000
2960
```

In [780…
```
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('punkt_tab')

ps = PorterStemmer()
stop_words = set(stopwords.words('english'))

# Define preprocessing function
def preprocess_text(text):
    text = text.lower()          # No change
    text = re.sub(r'[^\w\s[가-힣]]', '', text)     # Removes anything not alphan
    tokens = word_tokenize(text)
    tokens = [word for word in tokens if word not in stop_words]        # +0.2%
    tokens = [ps.stem(word) for word in tokens]       # +1% acc for stemming
    return ' '.join(tokens)

# Apply preprocessing to each document
data['content'] = data['content'].apply(preprocess_text)
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/oscarparrish/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     /Users/oscarparrish/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to
[nltk_data]     /Users/oscarparrish/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
```

In [781…
```
# Feature extraction

from sklearn.feature_extraction.text import TfidfVectorizer

# Convert text data into TF-IDF weights
```

```
vectorizer = TfidfVectorizer(max_features=400) # Empircally 400 has best results
X = vectorizer.fit_transform(data['content'])
```

In [782…
```
from sklearn.model_selection import train_test_split

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, data['topic'], test_size=
```

In [783…
```
print(X_train.shape, X_test.shape)
```

```
(1184, 400) (296, 400)
```

In [784…
```
# Get class of split
print("Training set class distribution:")
print(y_train.value_counts())

print("\nTest set class distribution:")
print(y_test.value_counts())
```

```
Training set class distribution:
topic
dark         392
sadness      289
personal     268
lifestyle    171
emotion       64
Name: count, dtype: int64

Test set class distribution:
topic
dark         95
sadness      82
personal     73
lifestyle    31
emotion      15
Name: count, dtype: int64
```

In [785…
```
from sklearn.naive_bayes import MultinomialNB
from sklearn.naive_bayes import BernoulliNB
from sklearn import svm
from sklearn.model_selection import cross_val_score, cross_validate

mnb = MultinomialNB()
# mnb.fit(X_train, y_train)

bnb = BernoulliNB()
# bnb.fit(X_train, y_train)

svecm = svm.SVC()
# svm.fit(X_train, y_train)
```

In [786…
```
from sklearn.metrics import accuracy_score, classification_report, make_scorer,

# # Predict the categories of the test set
# y_pred = mnb.predict(X_test)
# y2_pred = bnb.predict(X_test)
# y3_pred = svm.predict(X_test)

y = data['topic']
```

```
# Define multiple scoring metrics using make_scorer
scoring = {'accuracy': make_scorer(accuracy_score),
           'precision': make_scorer(precision_score, average='macro', zero_divis
           'recall': make_scorer(recall_score, average='macro'),
           'f1': make_scorer(f1_score, average='macro', zero_division=1)
           }

# scoring = make_scorer(accuracy_score)

# Defaults to 5 fold cross evaluation
mnb_cv = cross_validate(mnb, X, y, scoring=scoring)
bnb_cv = cross_validate(bnb, X, y, scoring=scoring)
svm_cv = cross_validate(svecm, X, y, scoring=scoring)
```

In [787…

```
def cv_report(cv):
    class_labels = ['dark\t', 'sadness\t', 'personal', 'lifestyle', 'emotion\t']
    print(f"Mean accuracy: {cv['test_accuracy'].mean():.2f}")
    print('\t\tPrecision\tRecall\t\tf1')
    for idx in range(5):
        print (f'{class_labels[idx]}\t{cv['test_precision'][idx]:.2f}\t\t{cv['te
```

In [788…

```
# Print accuracy and classification report
print('---- MNB ----')
cv_report(mnb_cv)

print('\n---- BNB ----')
cv_report(bnb_cv)

print('\n---- SVM ----')
cv_report(svm_cv)
```

```
---- MNB ----
Mean accuracy: 0.79
                Precision       Recall          f1
dark            0.88            0.66            0.66
sadness         0.86            0.64            0.63
personal        0.87            0.64            0.64
lifestyle       0.84            0.61            0.60
emotion         0.85            0.62            0.64

---- BNB ----
Mean accuracy: 0.65
                Precision       Recall          f1
dark            0.58            0.56            0.57
sadness         0.56            0.55            0.55
personal        0.61            0.57            0.58
lifestyle       0.60            0.57            0.58
emotion         0.52            0.51            0.51

---- SVM ----
Mean accuracy: 0.86
                Precision       Recall          f1
dark            0.92            0.85            0.88
sadness         0.89            0.84            0.86
personal        0.84            0.79            0.81
lifestyle       0.87            0.79            0.82
emotion         0.87            0.83            0.85
```

PART 1 QUESTIONS

Question 1

1. Testing has concluded that the removal of the punctuation does not positively or negatively effect the performance of the model, as under 900 characters consistent with the regex are present in the dataset. I've modified the regex to include the korean language, after discovering a korean language song in the dataset. This should ensure the song is not wiped of all lyrics by the regex.
2. In order to improve the efficacy of our validation metrics, I've implemented cross validation to be used in addition to the measured accuracy of the prediction from the fitted models. The cross validation reports a similar accuracy to that observed in the fitted models, though is consistently marginally more conservative in it's estimation.

Question 2

1. The concatenated 'content' field is passed through 5 steps of data pre-processing.
    A. All text is first converted to lowercase. This appears to have a negligible effect on model accuracy.
    B. A regex filter is then run to remove any non alphanumeric or whitespace characters. Testing indicates this has neglible effect on performance, but helps to shrink the number of tokens. After investigating the dataset file, I found a song written in Korean, so the filter has been amended to allow for Korean language words.
    C. After being tokenized, we remove then remove the stopwords from the content, following the nltk stop-words list. Testing of the models with and without the removal of stopwords demonstrate an average 0.2% increase in accuracy when stopwords are removed.
    D. The most impactful stem is the stemming of the content. All words stems are pruned by removing the suffix. This massively reduces the token count. In testing, the stemming of content improves the accuracy of all models by an average of 1%.

Question 3

1. I've used the metrics precision, recall and f1 score. Precision measures the true positive ratio of the prediction. This is especially important in this dataset, as the dataset is highly imbalanced, with only 64 'emotion' tracks. A high precision score indicates the model is not steamrolling over this class with low representation to find a greater accuracy score. In a similar vein, recall tells us the proportion of positive predictions that were correct, useful again for ensuring the minority class is not ignored in training. The f1 score gives us a harmonised view of both metrics. This is useful for providing an overall score of the model at a glance and determingin overall performance.

As demonstrated in the tables above, the MNB model consistently outperforms the BNB model in every metric, for every class. It has a specific advantage in precision. Hence the MNB model is stronger.

Question 4

1. The effect of the number of features on results was tested empirically. The data is as follows:

#Features	MNB Acc	BNB Acc	SVM Acc
400	.79	.65	.86
450	.80	.65	.85
500	.79	.66	.85
Unlimited	.63	.53	.77

I decided to use 400 as the value of N, as I chose the SVM as my method, and I aimed to maximise it's individual performance

Question 5

1. As my third machine learning method, I chose an SVM, or Support Vector Machine. SVMs use datapoints to construct an optimal hyperplane. That is, a plane of arbitrary dimensionality which tries to reasonably split data points into regions. These regions will then define our classes. If the hyperplane is well defined, new datapoints should fall in the region corresponding to their correct class. SVMs are fast and efficient methods of seperating data. An SVM works well in this scenario thanks to the low number of classes. The fewer classes there are to seperate, the broader the regions created by the SVM can be, and the smaller the chance of a mislabel is. SVMs are especially well suited to text classification, as they retain their efficacy in high dimensional space, such as TF_IDF vectors. Further, an SVM works well with small and medium datasets. As the providing dataset only includes 1500 songs, a method which is effective even with minimal data is useful. I hypothesise this method will be a noticeable improvement in all metrics over the BNB and MNB models.

My hypothesise was confirmed by the tabulated results, as seen above. The SVM consistently outperforms the MNB and BNB in recall, precision, accuracy and f1 score. Hence, I will be using this method for topic classification in the remainder of the assignment.

In [789…

```
# Split Data into weeks 1,2,3 and 4
current_data_for_part2 = data.iloc[:1000].copy()
train_data_part2 = current_data_for_part2.iloc[:750].copy()
test_data_part2 = current_data_for_part2.iloc[750:1000].copy()

# Transform training content for the classifier using the pre-fitted vectorizer
X_train_part2_transformed = vectorizer.transform(train_data_part2['content'])
y_train_part2_topics = train_data_part2['topic']
```

In [790…

```
# Fit svm to the data
svecm.fit(X_train_part2_transformed, y_train_part2_topics)

# Predict topics for the training data (crucial for defining user profiles based
train_data_part2['predicted_topic'] = svecm.predict(X_train_part2_transformed)
```

In [791…

```
# Turn tsv to dict
def package_user_taste(user):
    class_labels = ['dark', 'emotion', 'lifestyle', 'personal', 'sadness']

    out_dict = {'dark': [], 'emotion': [], 'lifestyle': [], 'personal': [], 'sad

    for label in out_dict.keys():
        if (user[user['topic'] == label].empty != True):
            out_dict[label] = user[user['topic'] == label]['keywords'].iloc[0].s

    print(out_dict)
    return out_dict
```

In [792…

```
# Load user keyword files
raw_user1 = pd.read_csv('user1.tsv', sep='\t')
raw_user2 = pd.read_csv('user2.tsv', sep='\t')

user1_taste_keywords = package_user_taste(raw_user1)
user2_taste_keywords = package_user_taste(raw_user2)

print("\nUser 1 Taste Profile Keywords from user1.tsv:")
for topic, keywords in user1_taste_keywords.items():
    print(f"  {topic}: {keywords}")

print("\nUser 2 Taste Profile Keywords from user2.tsv:")
for topic, keywords in user2_taste_keywords.items():
    print(f"  {topic}: {keywords}")

# Random common words for a hypothetical user 3
user3_taste_keywords = {
    'dark': ['night', 'shadow', 'lonely', 'lost', 'fear', 'darkness', 'empty', '
    'emotion': ['heart', 'love', 'sad', 'feel', 'dream', 'deep', 'soul', 'connec
    'lifestyle': ['money', 'party', 'street', 'life', 'city', 'gang', 'success',
    'personal': ['friend', 'home', 'family', 'myself', 'world', 'together', 'jou
    'sadness': ['tears', 'cry', 'gone', 'broken', 'alone', 'fall', 'rain', 'miss
}
print("\nUser 3 Taste Profile Keywords (Hypothetical):")
for topic, keywords in user3_taste_keywords.items():
    print(f"  {topic}: {keywords}")
```

```
{'dark': ['fire', ' enemy', ' pain', ' storm', ' fight'], 'emotion': ['love', ' m
emory', ' hug', ' kiss', ' feel'], 'lifestyle': ['party', ' city', ' night', ' li
ght', ' rhythm'], 'personal': ['dream', ' truth', ' life', ' growth', ' identit
y'], 'sadness': ['cry', ' alone', ' heartbroken', ' tears', ' regret']}
{'dark': [], 'emotion': ['romance', ' touch', ' feeling', ' kiss', ' memory'], 'l
ifestyle': [], 'personal': [], 'sadness': ['lost', ' sorrow', ' goodbye', ' tear
s', ' silence']}

User 1 Taste Profile Keywords from user1.tsv:
  dark: ['fire', ' enemy', ' pain', ' storm', ' fight']
  emotion: ['love', ' memory', ' hug', ' kiss', ' feel']
  lifestyle: ['party', ' city', ' night', ' light', ' rhythm']
  personal: ['dream', ' truth', ' life', ' growth', ' identity']
  sadness: ['cry', ' alone', ' heartbroken', ' tears', ' regret']

User 2 Taste Profile Keywords from user2.tsv:
  dark: []
  emotion: ['romance', ' touch', ' feeling', ' kiss', ' memory']
  lifestyle: []
  personal: []
  sadness: ['lost', ' sorrow', ' goodbye', ' tears', ' silence']

User 3 Taste Profile Keywords (Hypothetical):
  dark: ['night', 'shadow', 'lonely', 'lost', 'fear', 'darkness', 'empty', 'pai
n']
  emotion: ['heart', 'love', 'sad', 'feel', 'dream', 'deep', 'soul', 'connect']
  lifestyle: ['money', 'party', 'street', 'life', 'city', 'gang', 'success', 'pow
er']
  personal: ['friend', 'home', 'family', 'myself', 'world', 'together', 'journe
y', 'spirit']
  sadness: ['tears', 'cry', 'gone', 'broken', 'alone', 'fall', 'rain', 'miss']
```

In [793…

```python
# Use dict of keywords to construct list of songs user 'liked'
def get_liked_songs_content(user_keywords_by_topic, training_data_with_predictio
    class_labels = ['dark', 'emotion', 'lifestyle', 'personal', 'sadness']
    liked_songs_by_predicted_topic = {topic: [] for topic in class_labels}

    for _, song_row in training_data_with_predictions.iterrows():
        song_content = song_row['content']
        predicted_topic = song_row['predicted_topic']

        if predicted_topic in user_keywords_by_topic:
            user_keywords_for_topic = user_keywords_by_topic[predicted_topic]
            if any(keyword in song_content for keyword in user_keywords_for_topi
                liked_songs_by_predicted_topic[predicted_topic].append(song_cont

    # concat
    combined_liked_documents = {topic: ' '.join(songs_list)
                                for topic, songs_list in liked_songs_by_predicte
    return combined_liked_documents
```

In [794…

```python
user1_combined_liked_docs = get_liked_songs_content(user1_taste_keywords, train_
user2_combined_liked_docs = get_liked_songs_content(user2_taste_keywords, train_
user3_combined_liked_docs = get_liked_songs_content(user3_taste_keywords, train_
```

In [795…

```python
# Vectorise
def create_user_tfidf_profiles(combined_documents, fitted_vectorizer):
    user_tfidf_profiles = {}
    for topic, combined_doc in combined_documents.items():
```

```
        if combined_doc:
            user_tfidf_profiles[topic] = fitted_vectorizer.transform([combined_d
        else:
            user_tfidf_profiles[topic] = None # No profile if no liked songs for
    return user_tfidf_profiles
```

In [796…

```
user1_tfidf_profiles = create_user_tfidf_profiles(user1_combined_liked_docs, vec
user2_tfidf_profiles = create_user_tfidf_profiles(user2_combined_liked_docs, vec
user3_tfidf_profiles = create_user_tfidf_profiles(user3_combined_liked_docs, vec
```

In [797…

```
def get_top_n_words(tfidf_vector, feature_names, n=20):
    if tfidf_vector is None or tfidf_vector.nnz == 0: # Check if vector is empty
        return []

    tfidf_scores = tfidf_vector.sum(axis=0).A1
    top_n_indices = tfidf_scores.argsort()[-n:][::-1]
    top_words_with_scores = [(feature_names[idx], tfidf_scores[idx]) for idx in

    return top_words_with_scores
```

In [798…

```
feature_names = vectorizer.get_feature_names_out()

print("\n--- User 1 Top 20 Words per Topic Profile ---")
for topic, profile_vector in user1_tfidf_profiles.items():
    print(f"\nTopic: {topic}")
    top_words = get_top_n_words(profile_vector, feature_names)
    if top_words:
        for word, score in top_words:
            print(f"- {word} (TF-IDF: {score:.4f})")
    else:
        print("  No liked songs for this topic to form a profile.")

print("\n--- User 2 Top 20 Words per Topic Profile ---")
for topic, profile_vector in user2_tfidf_profiles.items():
    print(f"\nTopic: {topic}")
    top_words = get_top_n_words(profile_vector, feature_names)
    if top_words:
        for word, score in top_words:
            print(f"- {word} (TF-IDF: {score:.4f})")
    else:
        print("  No liked songs for this topic to form a profile.")

print("\n--- User 3 Top 20 Words per Topic Profile ---")
for topic, profile_vector in user3_tfidf_profiles.items():
    print(f"\nTopic: {topic}")
    top_words = get_top_n_words(profile_vector, feature_names)
    if top_words:
        for word, score in top_words:
            print(f"- {word} (TF-IDF: {score:.4f})")
    else:
        print("  No liked songs for this topic to form a profile.")
```

--- User 1 Top 20 Words per Topic Profile ---

Topic: dark
- fight (TF-IDF: 0.3750)
- blood (TF-IDF: 0.2158)
- black (TF-IDF: 0.2155)
- grind (TF-IDF: 0.1910)
- stand (TF-IDF: 0.1776)
- know (TF-IDF: 0.1624)
- come (TF-IDF: 0.1565)
- like (TF-IDF: 0.1468)
- kill (TF-IDF: 0.1426)
- na (TF-IDF: 0.1340)
- gon (TF-IDF: 0.1263)
- hand (TF-IDF: 0.1250)
- tell (TF-IDF: 0.1237)
- follow (TF-IDF: 0.1123)
- head (TF-IDF: 0.1101)
- build (TF-IDF: 0.1081)
- yeah (TF-IDF: 0.1026)
- death (TF-IDF: 0.0998)
- light (TF-IDF: 0.0992)
- shoot (TF-IDF: 0.0991)

Topic: emotion
- good (TF-IDF: 0.6751)
- touch (TF-IDF: 0.3852)
- feel (TF-IDF: 0.3194)
- hold (TF-IDF: 0.1884)
- morn (TF-IDF: 0.1331)
- kiss (TF-IDF: 0.1289)
- feelin (TF-IDF: 0.1280)
- know (TF-IDF: 0.1250)
- vibe (TF-IDF: 0.1217)
- miss (TF-IDF: 0.1138)
- luck (TF-IDF: 0.1113)
- go (TF-IDF: 0.1050)
- lip (TF-IDF: 0.1008)
- want (TF-IDF: 0.0991)
- love (TF-IDF: 0.0965)
- babi (TF-IDF: 0.0887)
- real (TF-IDF: 0.0788)
- na (TF-IDF: 0.0698)
- light (TF-IDF: 0.0630)
- look (TF-IDF: 0.0626)

Topic: lifestyle
- night (TF-IDF: 0.3738)
- song (TF-IDF: 0.2859)
- stranger (TF-IDF: 0.2552)
- closer (TF-IDF: 0.2500)
- sing (TF-IDF: 0.2363)
- long (TF-IDF: 0.2331)
- tire (TF-IDF: 0.2048)
- tonight (TF-IDF: 0.1907)
- come (TF-IDF: 0.1773)
- right (TF-IDF: 0.1773)
- home (TF-IDF: 0.1751)
- play (TF-IDF: 0.1685)
- time (TF-IDF: 0.1512)

- wait (TF-IDF: 0.1430)
- na (TF-IDF: 0.1307)
- wan (TF-IDF: 0.1305)
- yeah (TF-IDF: 0.1243)
- ring (TF-IDF: 0.1194)
- readi (TF-IDF: 0.1064)
- like (TF-IDF: 0.0980)

Topic: personal
- life (TF-IDF: 0.5191)
- live (TF-IDF: 0.3087)
- chang (TF-IDF: 0.1919)
- world (TF-IDF: 0.1660)
- thank (TF-IDF: 0.1519)
- na (TF-IDF: 0.1508)
- dream (TF-IDF: 0.1446)
- teach (TF-IDF: 0.1441)
- know (TF-IDF: 0.1400)
- yeah (TF-IDF: 0.1310)
- lord (TF-IDF: 0.1221)
- wan (TF-IDF: 0.1115)
- time (TF-IDF: 0.1070)
- thing (TF-IDF: 0.1062)
- like (TF-IDF: 0.1042)
- learn (TF-IDF: 0.0991)
- beat (TF-IDF: 0.0967)
- think (TF-IDF: 0.0931)
- need (TF-IDF: 0.0880)
- come (TF-IDF: 0.0874)

Topic: sadness
- think (TF-IDF: 0.5103)
- leav (TF-IDF: 0.4313)
- place (TF-IDF: 0.3314)
- blame (TF-IDF: 0.2380)
- want (TF-IDF: 0.2221)
- hold (TF-IDF: 0.1922)
- word (TF-IDF: 0.1799)
- chang (TF-IDF: 0.1651)
- caus (TF-IDF: 0.1601)
- trust (TF-IDF: 0.1487)
- space (TF-IDF: 0.1487)
- mind (TF-IDF: 0.1471)
- lord (TF-IDF: 0.1457)
- away (TF-IDF: 0.1323)
- dream (TF-IDF: 0.1029)
- tell (TF-IDF: 0.0889)
- break (TF-IDF: 0.0877)
- fail (TF-IDF: 0.0846)
- bare (TF-IDF: 0.0841)
- forgiv (TF-IDF: 0.0811)

--- User 2 Top 20 Words per Topic Profile ---

Topic: dark
  No liked songs for this topic to form a profile.

Topic: emotion
- touch (TF-IDF: 0.7126)
- good (TF-IDF: 0.3821)

- kiss (TF-IDF: 0.2386)
- morn (TF-IDF: 0.2164)
- hold (TF-IDF: 0.2145)
- luck (TF-IDF: 0.2059)
- lip (TF-IDF: 0.1526)
- knock (TF-IDF: 0.1068)
- feel (TF-IDF: 0.0856)
- wait (TF-IDF: 0.0840)
- time (TF-IDF: 0.0838)
- know (TF-IDF: 0.0816)
- feet (TF-IDF: 0.0763)
- love (TF-IDF: 0.0755)
- go (TF-IDF: 0.0664)
- bodi (TF-IDF: 0.0638)
- rememb (TF-IDF: 0.0587)
- goodby (TF-IDF: 0.0581)
- fli (TF-IDF: 0.0556)
- keep (TF-IDF: 0.0552)

Topic: lifestyle
  No liked songs for this topic to form a profile.

Topic: personal
  No liked songs for this topic to form a profile.

Topic: sadness
- open (TF-IDF: 0.4225)
- smile (TF-IDF: 0.3459)
- tear (TF-IDF: 0.3066)
- away (TF-IDF: 0.2259)
- lone (TF-IDF: 0.2028)
- come (TF-IDF: 0.1841)
- tri (TF-IDF: 0.1660)
- eye (TF-IDF: 0.1649)
- road (TF-IDF: 0.1557)
- tell (TF-IDF: 0.1520)
- babi (TF-IDF: 0.1444)
- life (TF-IDF: 0.1399)
- sight (TF-IDF: 0.1310)
- hous (TF-IDF: 0.1270)
- control (TF-IDF: 0.1260)
- promis (TF-IDF: 0.1212)
- good (TF-IDF: 0.1164)
- hold (TF-IDF: 0.1094)
- lose (TF-IDF: 0.1091)
- year (TF-IDF: 0.1067)

--- User 3 Top 20 Words per Topic Profile ---

Topic: dark
- black (TF-IDF: 0.2314)
- come (TF-IDF: 0.2154)
- fight (TF-IDF: 0.2106)
- grind (TF-IDF: 0.1972)
- fear (TF-IDF: 0.1771)
- know (TF-IDF: 0.1484)
- na (TF-IDF: 0.1416)
- blood (TF-IDF: 0.1362)
- death (TF-IDF: 0.1316)
- like (TF-IDF: 0.1304)

- hand (TF-IDF: 0.1282)
- stand (TF-IDF: 0.1252)
- rais (TF-IDF: 0.1239)
- ghost (TF-IDF: 0.1236)
- gon (TF-IDF: 0.1229)
- hear (TF-IDF: 0.1220)
- head (TF-IDF: 0.1208)
- light (TF-IDF: 0.1200)
- follow (TF-IDF: 0.1187)
- feel (TF-IDF: 0.1125)

Topic: emotion
- good (TF-IDF: 0.5899)
- touch (TF-IDF: 0.3493)
- hold (TF-IDF: 0.3391)
- feel (TF-IDF: 0.2897)
- go (TF-IDF: 0.2506)
- darl (TF-IDF: 0.1354)
- morn (TF-IDF: 0.1207)
- know (TF-IDF: 0.1183)
- heart (TF-IDF: 0.1164)
- feelin (TF-IDF: 0.1161)
- vibe (TF-IDF: 0.1104)
- lip (TF-IDF: 0.1039)
- miss (TF-IDF: 0.1032)
- want (TF-IDF: 0.1020)
- luck (TF-IDF: 0.1009)
- babi (TF-IDF: 0.0971)
- love (TF-IDF: 0.0875)
- light (TF-IDF: 0.0800)
- yeah (TF-IDF: 0.0754)
- kiss (TF-IDF: 0.0744)

Topic: lifestyle
- stranger (TF-IDF: 0.3996)
- night (TF-IDF: 0.3248)
- right (TF-IDF: 0.3092)
- sing (TF-IDF: 0.3044)
- song (TF-IDF: 0.2808)
- play (TF-IDF: 0.2319)
- come (TF-IDF: 0.2047)
- time (TF-IDF: 0.1804)
- wait (TF-IDF: 0.1560)
- want (TF-IDF: 0.1518)
- readi (TF-IDF: 0.1439)
- know (TF-IDF: 0.1389)
- make (TF-IDF: 0.1284)
- tonight (TF-IDF: 0.1154)
- home (TF-IDF: 0.1089)
- belong (TF-IDF: 0.1032)
- thought (TF-IDF: 0.1026)
- summer (TF-IDF: 0.1019)
- na (TF-IDF: 0.0942)
- life (TF-IDF: 0.0933)

Topic: personal
- world (TF-IDF: 0.4040)
- live (TF-IDF: 0.2846)
- thank (TF-IDF: 0.2710)
- life (TF-IDF: 0.2601)

```
- lord (TF-IDF: 0.1638)
- na (TF-IDF: 0.1577)
- chang (TF-IDF: 0.1473)
- know (TF-IDF: 0.1457)
- need (TF-IDF: 0.1378)
- want (TF-IDF: 0.1284)
- yeah (TF-IDF: 0.1282)
- give (TF-IDF: 0.1279)
- good (TF-IDF: 0.1167)
- simpl (TF-IDF: 0.1153)
- wan (TF-IDF: 0.1095)
- day (TF-IDF: 0.1038)
- thing (TF-IDF: 0.0974)
- year (TF-IDF: 0.0958)
- dream (TF-IDF: 0.0954)
- time (TF-IDF: 0.0950)

Topic: sadness
- fall (TF-IDF: 0.6140)
- break (TF-IDF: 0.2492)
- heart (TF-IDF: 0.2473)
- away (TF-IDF: 0.1740)
- like (TF-IDF: 0.1447)
- feel (TF-IDF: 0.1430)
- na (TF-IDF: 0.1326)
- babi (TF-IDF: 0.1309)
- know (TF-IDF: 0.1304)
- leav (TF-IDF: 0.1241)
- walk (TF-IDF: 0.1195)
- wan (TF-IDF: 0.1188)
- come (TF-IDF: 0.1126)
- yeah (TF-IDF: 0.1110)
- think (TF-IDF: 0.1066)
- hurt (TF-IDF: 0.1039)
- start (TF-IDF: 0.1032)
- wall (TF-IDF: 0.1025)
- time (TF-IDF: 0.0894)
- caus (TF-IDF: 0.0886)
```

PART 2 QUESTIONS

1. The top 20 words for each user per topic are listed above. They appear to be reasonably congruent with each user's given keywords. For instance, the top appearing words in 'dark' songs for user 1, generated from their keywords [fire, enemy, pain, storm, fight]; are words such as fight, blood, black, stand and kill. These words seem congruous with the user's interests. The same is true for user 2 and user 3.

2. For the matching algorithm, I found using an item-ased similiratiy approach maximising cosine similarity to be the most effective approach. Cosine similarity determines similiraity by comparing angles between vectors, making it suited to high dimensional data such as TF_IDF vectors. The other best approach, Jaccard similarity focuses on the overlap between sets. Hence, it is useful for binary classification, but struggles with the multiple classes, which is required in this problem.

In [799…

```
from sklearn.metrics.pairwise import cosine_similarity

X_test_part2_transformed = vectorizer.transform(test_data_part2['content'])
test_data_part2['predicted_topic'] = svecm.predict(X_test_part2_transformed)

original_test_songs_info = raw_data.loc[test_data_part2.index][['artist_name', '
test_data_part2 = test_data_part2.merge(original_test_songs_info, left_index=Tru

N_recommendations = 10
```

In [800…

```
def recommend_songs_for_user(user_tfidf_profiles, test_songs_df, X_test_transfor
    recommendations = []

    for i, song_row in test_songs_df.iterrows():
        song_content_vector = X_test_transformed[test_songs_df.index.get_loc(i)]
        predicted_topic = song_row['predicted_topic']
        artist_name = song_row['artist_name']
        track_name = song_row['track_name']
        user_profile_vector = user_tfidf_profiles.get(predicted_topic)

        if user_profile_vector is not None and user_profile_vector.nnz > 0:
            similarity = cosine_similarity(song_content_vector, user_profile_vec
            recommendations.append({
                'artist_name': artist_name,
                'track_name': track_name,
                'predicted_topic': predicted_topic,
                'similarity': similarity
            })
    recommendations.sort(key=lambda x: x['similarity'], reverse=True)
    return recommendations[:N]
```

In [801…

```
print("\n--- Recommendations for User 1 (Top 10) ---")
user1_recommendations = recommend_songs_for_user(user1_tfidf_profiles, test_data
if user1_recommendations:
    for rec in user1_recommendations:
        print(f"- '{rec['track_name']}' by {rec['artist_name']} (Topic: {rec['pr
else:
    print(" No recommendations found for User 1 based on available profiles and

print("\n--- Recommendations for User 2 (Top 10) ---")
user2_recommendations = recommend_songs_for_user(user2_tfidf_profiles, test_data
if user2_recommendations:
    for rec in user2_recommendations:
        print(f"- '{rec['track_name']}' by {rec['artist_name']} (Topic: {rec['pr
else:
    print(" No recommendations found for User 2 based on available profiles and

print("\n--- Recommendations for User 3 (Top 10) ---")
user3_recommendations = recommend_songs_for_user(user3_tfidf_profiles, test_data
if user3_recommendations:
    for rec in user3_recommendations:
        print(f"- '{rec['track_name']}' by {rec['artist_name']} (Topic: {rec['pr
else:
    print(" No recommendations found for User 3 based on available profiles and
```

```
--- Recommendations for User 1 (Top 10) ---
- 'once in a while' by timeflies (Topic: emotion, Similarity: 0.6831)
- 'got it good' by justin moore (Topic: emotion, Similarity: 0.6759)
- 'horsefly' by dirty heads (Topic: emotion, Similarity: 0.5572)
- 'life changes' by thomas rhett (Topic: personal, Similarity: 0.5021)
- 'sit awhile' by the band steele (Topic: personal, Similarity: 0.4584)
- 'alta' by ty segall (Topic: personal, Similarity: 0.4368)
- 'boy in the bubble' by alec benjamin (Topic: dark, Similarity: 0.4233)
- 'you're the best thing yet' by anita baker (Topic: personal, Similarity: 0.402
9)
- 'living it up' by damian marley (Topic: personal, Similarity: 0.4018)
- 'wash away' by iya terra (Topic: personal, Similarity: 0.4017)

--- Recommendations for User 2 (Top 10) ---
- 'got it good' by justin moore (Topic: emotion, Similarity: 0.4263)
- 'once in a while' by timeflies (Topic: emotion, Similarity: 0.3466)
- 'hey yo' by 311 (Topic: sadness, Similarity: 0.3368)
- 'balcony' by lester nowhere (Topic: sadness, Similarity: 0.3316)
- 'will you be mine' by anita baker (Topic: sadness, Similarity: 0.3051)
- 'smile' by tesseract (Topic: sadness, Similarity: 0.3026)
- 'doors closing' by moonchild (Topic: sadness, Similarity: 0.2641)
- 'horsefly' by dirty heads (Topic: emotion, Similarity: 0.2618)
- 'cranes in the sky' by solange (Topic: sadness, Similarity: 0.2612)
- 'light years' by the national (Topic: sadness, Similarity: 0.2550)

--- Recommendations for User 3 (Top 10) ---
- 'once in a while' by timeflies (Topic: emotion, Similarity: 0.6212)
- 'love falls' by hellyeah (Topic: sadness, Similarity: 0.6206)
- 'got it good' by justin moore (Topic: emotion, Similarity: 0.6150)
- 'the fighter' by keith urban (Topic: sadness, Similarity: 0.5621)
- 'no one in the world' by anita baker (Topic: personal, Similarity: 0.5319)
- 'truly madly deeply' by yoke lore (Topic: sadness, Similarity: 0.5123)
- 'horsefly' by dirty heads (Topic: emotion, Similarity: 0.5081)
- 'crossfire / so into you' by nai palm (Topic: sadness, Similarity: 0.4912)
- 'we find love' by daniel caesar (Topic: sadness, Similarity: 0.4881)
- 'fallin (feat. 6lack)' by bazzi (Topic: sadness, Similarity: 0.4794)
```

In [ ]:

```python
def get_liked_songs_by_fixed_indices(liked_song_indices, training_data_with_pred
    class_labels = ['dark', 'emotion', 'lifestyle', 'personal', 'sadness']
    liked_songs_by_predicted_topic = {topic: [] for topic in class_labels}
    selected_liked_songs = training_data_with_predictions.loc[liked_song_indices

    for _, song_row in selected_liked_songs.iterrows():
        song_content = song_row['content']
        predicted_topic = song_row['predicted_topic']

        if predicted_topic in class_labels:
            liked_songs_by_predicted_topic[predicted_topic].append(song_content)
        else:
            print(f"Warning: Song at index {song_row.name} has an unexpected pre

    combined_liked_documents = {topic: ' '.join(songs_list)
                                for topic, songs_list in liked_songs_by_predicte
    return combined_liked_documents

# Define the specific indices for the liked songs provided by my user
fixed_liked_indices = [19, 26, 125, 158, 214, 239, 278, 280, 380, 405, 562, 563,
```

In [804…

```python
from pprint import pprint
```

```
N_user_study = 30

user_study_combined_liked_docs = get_liked_songs_by_fixed_indices(fixed_liked_in
user_study_tfidf_profiles = create_user_tfidf_profiles(user_study_combined_liked

if user_study_tfidf_profiles:
    week4_recommendations = recommend_songs_for_user(user_study_tfidf_profiles,

    if week4_recommendations:
        print(f"Top {N_user_study} Recommended Songs for Week 4:")
        for i, rec in enumerate(week4_recommendations):
            print(f"{i+1}. **'{rec['track_name']}' by {rec['artist_name']}** (To
    else:
        print("**No recommendations generated for Week 4. This could be due to a
```

```
Top 30 Recommended Songs for Week 4:
1. **'woman, amen' by dierks bentley** (Topic: personal, Similarity: 0.4244)
2. **'mississippi kisses' by leon bridges** (Topic: sadness, Similarity: 0.4123)
3. **'(your love keeps lifting me) higher and higher' by elvin bishop** (Topic: l
ifestyle, Similarity: 0.3588)
4. **'mad world' by jordan rakei** (Topic: personal, Similarity: 0.2617)
5. **'skin the rabbit' by dispatch** (Topic: dark, Similarity: 0.2615)
6. **'first time again' by jason aldean** (Topic: lifestyle, Similarity: 0.2539)
7. **'holler' by band of rascals** (Topic: personal, Similarity: 0.2462)
8. **'baroness' by wolfmother** (Topic: lifestyle, Similarity: 0.2460)
9. **'chocolatize' by brant bjork** (Topic: sadness, Similarity: 0.2416)
10. **'no one in the world' by anita baker** (Topic: personal, Similarity: 0.240
9)
11. **'terraformer' by thank you scientist** (Topic: sadness, Similarity: 0.2160)
12. **'it's only right' by wallows** (Topic: lifestyle, Similarity: 0.2100)
13. **'oh, what a world' by kacey musgraves** (Topic: personal, Similarity: 0.197
5)
14. **'one call away' by charlie puth** (Topic: sadness, Similarity: 0.1931)
15. **'life changes' by thomas rhett** (Topic: personal, Similarity: 0.1904)
16. **'wash away' by iya terra** (Topic: personal, Similarity: 0.1902)
17. **'bury my bones' by whiskey myers** (Topic: dark, Similarity: 0.1841)
18. **'heroin' by badflower** (Topic: dark, Similarity: 0.1836)
19. **'sit awhile' by the band steele** (Topic: personal, Similarity: 0.1817)
20. **'live without limit' by jahmiel** (Topic: personal, Similarity: 0.1779)
21. **'donner bell' by deca** (Topic: dark, Similarity: 0.1595)
22. **'7-t's' by marcus miller** (Topic: lifestyle, Similarity: 0.1574)
23. **'live well' by palace** (Topic: personal, Similarity: 0.1495)
24. **'boy in the bubble' by alec benjamin** (Topic: dark, Similarity: 0.1459)
25. **'whole wide world' by cage the elephant** (Topic: personal, Similarity: 0.1
448)
26. **'the mystic' by adam jensen** (Topic: sadness, Similarity: 0.1436)
27. **'lay' by the blue stones** (Topic: sadness, Similarity: 0.1384)
28. **'changing' by john mayer** (Topic: personal, Similarity: 0.1289)
29. **'guess it’s all over' by mndsgn** (Topic: dark, Similarity: 0.1278)
30. **'us against the world' by killswitch engage** (Topic: personal, Similarity:
0.1233)
```

PART 3 QUESTIONS

Above is the predicted output for week 4 according to my users selected songs. The User was provided 30 songs from weeks 1,2 and 3 respectively, and selected the following songs:

Week 1

- post malone spoil my night (feat. swae lee) 2018
- imagine dragons mouth of the river 2017 rock
- derek b bullet from a gun 2016 hip hop boot
- buczer hip hop prod. crackhouse 2017 hip hop
- imagine dragons not today 2016 rock
- imagine dragons bullet in a gun 2018 rock

Week 2

- yungblud kill somebody 2018 rock
- half•alive the fall 2017 rock
- the weeknd ordinary life 2016 pop
- t-rock what's stopping you (feat. unexpected) 2016 hip hop

Week 3

- panic! at the disco golden days 2016
- avril lavigne head above water 2019 pop
- ed sheeran what do i know? 2017 pop
- twenty one pilots neon gravestones

The final recommended songs achieved a notably lower cosine similarity score than the hypothetical user's recommendations from part 2. I believe this i due to the significantly smaller 'liked songs' list. Selecting songs a user 'likes' based on keywords cannot provide an accurate picture of that person's preferences, and is more likely to lead the algorithm astray.

Of the songs recommended, my user liked:

- 'one call away' by charlie puth
- 'the mystic' by adam jensen
- 'heroin' by badflower

Giving the recommendation an efficacy of 10%.

My user reported that my recommendations offered him a significant number of jazz and blues songs. This is contrary to his tastes, as he selected exclusively songs marked 'rock', 'pop', and 'hip-hop'. This could be the result of a class imbalance in genres, or error in the code.