In [201]:
```
import nltk
nltk.download('punkt')
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
from nltk import stem
stemmer = stem.PorterStemmer()
from nltk import word_tokenize
nltk.download('stopwords')
from nltk.corpus import stopwords
stops = set(stopwords.words('english'))
import string
punct = list(string.punctuation)
from collections import Counter
import requests
import spacy
import pandas as pd
import seaborn as sns
sns.set()
import matplotlib.pyplot as plt
!pip install PRAW
import numpy as np
import praw
import datetime
!pip install requests
!pip install lyricsgenius
import json
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import LatentDirichletAllocation
```

```
lanphere/anaconda3/lib/python3.11/site-packages (from prawcore<3,>=
2.1->PRAW) (2.31.0)
Requirement already satisfied: six in /Users/blaiselanphere/anaconda
3/lib/python3.11/site-packages (from websocket-client>=0.54.0->PRAW)
(1.16.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /Users/bl
aiselanphere/anaconda3/lib/python3.11/site-packages (from requests<
3.0,>=2.6.0->prawcore<3,>=2.1->PRAW) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /Users/blaiselanpher
e/anaconda3/lib/python3.11/site-packages (from requests<3.0,>=2.6.0-
>prawcore<3,>=2.1->PRAW) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /Users/blaisela
nphere/anaconda3/lib/python3.11/site-packages (from requests<3.0,>=
2.6.0->prawcore<3,>=2.1->PRAW) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in /Users/blaisela
nphere/anaconda3/lib/python3.11/site-packages (from requests<3.0,>=
2.6.0->prawcore<3,>=2.1->PRAW) (2023.7.22)
Requirement already satisfied: requests in /Users/blaiselanphere/ana
conda3/lib/python3.11/site-packages (2.31.0)
```

In [142]:
```
# importing the VAD model from excel
vad = pd.read_excel('vad_model.xlsx', index_col = 0)  #VAD norms
sm = pd.read_excel('sensorimotor.xlsx', index_col = 0) #Sensorimotor r
sm = sm[['auditory', 'gustatory', 'haptic', 'interoceptive', 'olfactor
       'visual', 'foot_leg', 'hand_arm', 'head', 'mouth', 'torso']]
```

Importing the data from Exa for each year 2016-2023 for articles that have Trump in the title

In [79]:
```
from exa_py import Exa
exa = Exa("93003122-a32c-4db9-b474-fcc8644bac91")

trump_2016 = exa.search_and_contents(
  "Trump",
  num_results=100,
  start_published_date="2016-01-01",
  end_published_date="2016-12-31",
  use_autoprompt=True,
  text={"include_html_tags": False},
)
```

In [80]:
```
results = trump_2016.results
trump_2016 = [result.text for result in results]
```

In [81]:
```
trump_2016 = ''.join(trump_2016)
trump_2016
```

Out[81]:
```
'  Randal Pinkett's first day in the Trump Organization was one he w
ould never forget. Summoned to the offices in Trump Tower, the billi
onaire's garish midtown skyscraper, Pinkett entered the room as Trum
p thumbed through a stack of the day's newspapers and magazines. It
was 2005, and having just won season four of The Apprentice, the onl
y African American to do so in the show's history, Pinkett expected
Trump's attention. But as the two spoke about his hard-won contract
with the company, it was clear Trump really only cared about one thi
ng: himself. He broke off from the conversation intermittently, pull
ing a paper from the pile, carefully scanning each page with a yello
w Post-It note stuck to it and disregarding the rest — an aide had a
lready combed through the publications to mark out every article tha
t mentioned the boss. This was his morning routine. "I think that ju
st speaks volumes," Pinkett said in an interview. "Donald loves Dona
ld. "His identity is wrapped around being a winner. If you challenge
him, or if he's put into a losing position, now you begin to take Do
nald out of his comfort zone." In interviews with 12 former employee
s of Donald Trump, the frontrunner for the Republican presidential n
omination and now one of the most controversial figures in modern Am
erican politics, none disagreed with Pinkett's frank assessment of h
```

In [82]:
```
from exa_py import Exa
exa = Exa("93003122-a32c-4db9-b474-fcc8644bac91")

trump_2017 = exa.search_and_contents(
  "Trump",
  num_results=100,
  start_published_date="2017-01-01",
  end_published_date="2017-12-31",
  use_autoprompt=True,
  text={"include_html_tags": False},
)
```

In [83]:
```
results = trump_2017.results
trump_2017 = [result.text for result in results]
```

In [84]:
```
trump_2017 = ''.join(trump_2017)
trump_2017
```

```
restaurant consultant whose poor judgment cost a business valuable t
ime and money. Two people familiar with the discussion said the Situ
ation Room meeting, in which the president’s advisers anticipated he
would sign off on a new Afghanistan strategy, was so unproductive th
at the advisers decided to continue the discussion at the Pentagon t
he next day in a smaller setting where the president could perhaps b
e more focused. “It wasn’t just the number of people. It was the ide
a of focus,” according to one person familiar with the discussion. T
he thinking was: “Maybe we need to slow down a little and explain th
e whole world” from a big-picture perspective, this person said. The
Pentagon meeting was also attended by Vice President Mike Pence; Tre
asury Secretary Steven Mnuchin; Gen. Joseph Dunford, the chairman of
the Joint Chiefs of Staff; Gen. Paul Selva, the vice chairman; Defen
se Secretary James Mattis; Deputy Defense Secretary Patrick Shanaha
n; Stephen Bannon, then Trump’s chief strategist; Jared Kushner, the
president\'s son-in-law and senior adviser; and Reince Preibus, then
chief of staff. Sean Spicer, then the White House spokesman, and Kei
th Schiller, who was director of Oval Office operations at the time,
also accompanied Trump to the Pentagon that day.  Related: Donald Tr
ump Has History of Mixed Messages on Nuclear Weapons   Asked for a r
```

In [85]:
```
from exa_py import Exa
exa = Exa("93003122-a32c-4db9-b474-fcc8644bac91")

trump_2018 = exa.search_and_contents(
  "Trump",
  num_results=100,
  start_published_date="2018-01-01",
  end_published_date="2018-12-31",
  use_autoprompt=True,
  text={"include_html_tags": False},
)
```

In [86]:
```
results = trump_2018.results
trump_2018 = [result.text for result in results]
```

In [87]:
```
trump_2018 = ''.join(trump_2018)
trump_2018
```

Out[87]:
```
'   President Trump boasted in a fundraising speech Wednesday that h
e made up\xa0information in a meeting with the leader of a top U.S.
ally, saying he insisted to Canadian Prime Minister Justin Trudeau t
hat the United States runs a trade deficit with its neighbor to the
north without knowing whether that was true.  “Trudeau came to see m
e. He’s a good guy, Justin. He said, ‘No, no, we have no trade defic
it with you, we have none. Donald, please,’ ” Trump said, mimicking
Trudeau, according to audio of the private event in\xa0Missouri obta
ined by The Washington Post. “Nice guy, good-looking guy, comes in —
‘Donald, we have no trade deficit.’ He’s very proud because everybod
y else, you know, we’re getting killed. “... So, he’s proud. I said,
‘Wrong, Justin, you do.’ I didn’t even know.\xa0... I had no idea. I
just said, ‘You’re wrong.’ You know why? Because we’re so stupid. …
And I thought they were smart. I said, ‘You’re wrong, Justin.’ He sa
id, ‘Nope, we have no trade deficit.’ I said, ‘Well, in that case, I
feel differently,’ I said, ‘but I don’t believe it.’ I sent one of o
ur guys out, his guy, my guy, they went out, I said, ‘Check, because
I can’t believe it.’ ‘Well, sir, you’re actually right. We have no d
eficit, but that doesn’t include energy and timber. … And when you d
```

In [88]:
```
from exa_py import Exa
exa = Exa("93003122-a32c-4db9-b474-fcc8644bac91")

trump_2019 = exa.search_and_contents(
  "Trump",
  num_results=100,
  start_published_date="2019-01-01",
  end_published_date="2019-12-31",
  use_autoprompt=True,
  text={"include_html_tags": False},
)
```

In [89]:
```
results = trump_2019.results
trump_2019 = [result.text for result in results]
```

In [90]:
```
trump_2019 = ''.join(trump_2019)
trump_2019
```

Out[90]:
```
'  \nThe New York Times: Digital and Home Delivery Subscriptions\n
\n \n \n \n \n Offer for a New York Times News subscription; current
subscribers not eligible. Subscription excludes print edition. Subsc
ription also excludes digital access to New York Times Games, Cookin
g, Wirecutter or The Athletic. Your payment method will automaticall
y be charged in advance the introductory rate of $4 every 4 weeks fo
r 1 year, and after 1 year the standard rate of $17 every 4 weeks. Y
our subscription will continue until you cancel. Cancellation takes
effect at the end of your current billing period. Taxes may apply. O
ffer terms are subject to change.\n \n \n \n \n plus-icon\n \n \n \n
\n \n check\n \n \n \n \n  Subscribe to The Times to read (and prin
t) as many articles as you'd like. nytimes.com/subscription      Wa
shington — President Donald Trump has put on some pounds and is now
officially considered obese. The White House on Thursday released re
sults of his most recent physical, revealing that his Body Mass Inde
x is now 30.4. That's based on the fact that he's now carrying 243 p
ounds on his 6-foot, 3-inch frame. That's up from 236 pounds in Sept
ember 2016 before he became president. An index rating of 30 is the
level at which doctors consider someone obese under this commonly us
```

In [91]:
```
from exa_py import Exa
exa = Exa("93003122-a32c-4db9-b474-fcc8644bac91")

trump_2020 = exa.search_and_contents(
  "Trump",
  num_results=100,
  start_published_date="2020-01-01",
  end_published_date="2020-12-31",
  use_autoprompt=True,
  text={"include_html_tags": False},
)
```

In [92]:
```
results = trump_2020.results
trump_2020 = [result.text for result in results]
```

In [93]:
```
trump_2020 = ''.join(trump_2020)
trump_2020
```

Out[93]:
```
'   During weeks of briefings and discussions over the escalating co
ronavirus, President Trump has repeatedly fixated on one thing above
all: the numbers. He has aggressively quizzed aides about infection
statistics — asking how many cases are in each state, and how the qu
antity compares with other countries. He has clung to the rosiest pr
ojections, repeating only the figures that support his belief that t
he coronavirus is not morphing into a global catastrophe. And he has
intensely followed the plummeting stock market, which plunged more t
han 1,600 points Wednesday.  Trump’s obsession with numbers — both p
ublicly and privately — has dominated and shaped the administratio
n’s response to the coronavirus, as advisers and public health exper
ts try to placate a leader who largely views the global pandemic thr
ough the political lens of how the statistics reflect on his preside
ncy and hopes for reelection. This account is based on interviews wi
th more than a dozen administration officials and other observers, m
any of whom spoke on the condition of anonymity to discuss internal
deliberations. He wants market numbers up, and he wants case numbers
down. Trump is a man who has measured much of his life in numbers —
first wealth, then crowd size and votes, and now unemployment and ec
```

In [168]:
```
from exa_py import Exa
exa = Exa("93003122-a32c-4db9-b474-fcc8644bac91")

trump_2021 = exa.search_and_contents(
  "Trump",
  num_results=100,
  start_published_date="2021-01-01",
  end_published_date="2021-12-31",
  use_autoprompt=True,
  text={"include_html_tags": False},
)
```

In [169]:
```
results = trump_2021.results
trump_2021 = [result.text for result in results]
```

In [170]:
```
trump_2021 = ''.join(trump_2021)
trump_2021
```

Out[170]:
```
' \n Donald Trump golfing. (Photo via Twitter)\n    According to a r
eport from Politico, Donald Trump\'s grand ambitions for his post-pr
esidency are floundering during his "exile" to his Mar-a-Lago resor
t. The former president was expected to play kingmaker in the Republ
ican Party as he made plans for his political comeback, but aides cl
ose to the president and former White House officials are now saying
his post-presidency is directionless and that his "barebones" staff
is trying to keep him relevant.  According to Gabby Orr and Meredith
McGraw, "His vow to target disloyal Republicans with personally-recr
uited primary challengers has taken a backseat to conventional endor
sements of senators who refused to indulge his quest to overturn the
2020 election. And though he was supposed to build a massive politic
al apparatus to keep his MAGA movement afloat, it\'s unclear to Repu
blicans what his PAC is actually doing, beyond entangling itself in
disputes with Republican icons and the party\'s fundraising arms." O
ne person close to Trump lamented, "There is no apparatus, no struct
ure and part of that is due to a lack of political understanding on
Trump\'s behalf."   GOP strategist Matt Gorman agreed, adding, "It
\'s like political phantom limbs. He doesn\'t have the same politica
```

In [171]:
```
from exa_py import Exa
exa = Exa("93003122-a32c-4db9-b474-fcc8644bac91")

trump_2022 = exa.search_and_contents(
  "Trump",
  num_results=100,
  start_published_date="2022-01-01",
  end_published_date="2022-12-31",
  use_autoprompt=True,
  text={"include_html_tags": False},
)
```

In [172]:
```
results = trump_2022.results
trump_2022 = [result.text for result in results]
```

In [173]:
```
trump_2022 = ''.join(trump_2022)
```

In [119]:
```
from exa_py import Exa
exa = Exa("93003122-a32c-4db9-b474-fcc8644bac91")

trump_2023 = exa.search_and_contents(
  "Trump",
  num_results=100,
  start_published_date="2023-01-01",
  end_published_date="2023-12-31",
  use_autoprompt=True,
  text={"include_html_tags": False},
)
```

In [120]:
```
results = trump_2023.results
trump_2023 = [result.text for result in results]
```

In [122]:
```
trump_2023 = ''.join(trump_2023)
```

Tokenizing, lemmatizing and cleaning the data

I have taken each of the strings i have that contain 100 webpages that mention trump for each year since 2016, and tokenized them, removed punctuation and stopwords, then lemmatized and removed anything that is not alpha

In [180]:
```
tokens_2016 = word_tokenize(trump_2016)
tokens_2016 = [token for token in tokens_2016 if token not in string.p

stop_words = set(stopwords.words('english'))
tokens_2016 = [token for token in tokens_2016 if token.lower() not in
tokens_2016 = [word.lower() for word in tokens_2016]

lemmas_2016 = [lemmatizer.lemmatize(i) for i in tokens_2016]
filtered_2016 = [token for token in lemmas_2016 if token.isalpha()]
```

In [182]:
```
tokens_2017 = word_tokenize(trump_2017)
tokens_2017 = [token for token in tokens_2017 if token not in string.p

tokens_2017 = [token for token in tokens_2017 if token.lower() not in
tokens_2017 = [word.lower() for word in tokens_2017]

lemmas_2017 = [lemmatizer.lemmatize(i) for i in tokens_2017]
filtered_2017 = [token for token in lemmas_2017 if token.isalpha()]
```

In [183]:
```
tokens_2018 = word_tokenize(trump_2018)
tokens_2018 = [token for token in tokens_2018 if token not in string.p

tokens_2018 = [token for token in tokens_2018 if token.lower() not in
tokens_2018 = [word.lower() for word in tokens_2018]

lemmas_2018 = [lemmatizer.lemmatize(i) for i in tokens_2018]
filtered_2018 = [token for token in lemmas_2018 if token.isalpha()]
```

In [184]:
```
tokens_2019 = word_tokenize(trump_2019)
tokens_2019 = [token for token in tokens_2019 if token not in string.p

tokens_2019 = [token for token in tokens_2019 if token.lower() not in
tokens_2019 = [word.lower() for word in tokens_2019]

lemmas_2019 = [lemmatizer.lemmatize(i) for i in tokens_2019]
filtered_2019 = [token for token in lemmas_2019 if token.isalpha()]
```

In [185]:
```
tokens_2020 = word_tokenize(trump_2020)
tokens_2020 = [token for token in tokens_2020 if token not in string.p

tokens_2020 = [token for token in tokens_2020 if token.lower() not in
tokens_2020 = [word.lower() for word in tokens_2020]

lemmas_2020 = [lemmatizer.lemmatize(i) for i in tokens_2020]
filtered_2020 = [token for token in lemmas_2020 if token.isalpha()]
```

In [186]:
```
tokens_2021 = word_tokenize(trump_2021)
tokens_2021 = [token for token in tokens_2021 if token not in string.p

tokens_2021 = [token for token in tokens_2021 if token.lower() not in
tokens_2021 = [word.lower() for word in tokens_2021]

lemmas_2021 = [lemmatizer.lemmatize(i) for i in tokens_2021]
filtered_2021 = [token for token in lemmas_2021 if token.isalpha()]
```

In [187]:
```
tokens_2022 = word_tokenize(trump_2022)
tokens_2022 = [token for token in tokens_2022 if token not in string.p

tokens_2022 = [token for token in tokens_2021 if token.lower() not in
tokens_2022 = [word.lower() for word in tokens_2022]

lemmas_2022 = [lemmatizer.lemmatize(i) for i in tokens_2022]
filtered_2022 = [token for token in lemmas_2022 if token.isalpha()]
```

In [188]:
```
tokens_2023 = word_tokenize(trump_2023)
tokens_2023 = [token for token in tokens_2023 if token not in string.p

tokens_2023 = [token for token in tokens_2023 if token.lower() not in
tokens_2023 = [word.lower() for word in tokens_2023]

lemmas_2023 = [lemmatizer.lemmatize(i) for i in tokens_2023]
filtered_2023 = [token for token in lemmas_2023 if token.isalpha()]
```

Word Frequency

In [189]:
```
freq_2016 = Counter(filtered_2016)
freq_2016 = freq_2016.most_common(20)
freq_2016
```

Out[189]:
```
[('trump', 1914),
 ('said', 592),
 ('donald', 323),
 ('would', 247),
 ('one', 216),
 ('people', 196),
 ('time', 194),
 ('say', 193),
 ('like', 192),
 ('new', 174),
 ('year', 163),
 ('campaign', 147),
 ('told', 145),
 ('get', 144),
 ('republican', 137),
 ('presidential', 132),
 ('woman', 125),
 ('know', 123),
 ('could', 118),
 ('company', 115)]
```

In [190]:
```
freq_2017 = Counter(filtered_2017)
freq_2017 = freq_2017.most_common(20)
freq_2017
```

Out[190]:
```
[('trump', 1481),
 ('said', 546),
 ('president', 421),
 ('would', 226),
 ('time', 224),
 ('one', 220),
 ('donald', 189),
 ('like', 185),
 ('new', 184),
 ('year', 178),
 ('tax', 172),
 ('say', 169),
 ('white', 164),
 ('people', 158),
 ('also', 146),
 ('official', 138),
 ('told', 125),
 ('house', 121),
 ('could', 119),
 ('know', 117)]
```

In [191]:
```
freq_2018 = Counter(filtered_2018)
freq_2018 = freq_2018.most_common(20)
freq_2018
```

Out[191]:
```
[('trump', 1559),
 ('said', 991),
 ('president', 530),
 ('trade', 414),
 ('donald', 233),
 ('time', 219),
 ('state', 216),
 ('know', 208),
 ('would', 204),
 ('canada', 197),
 ('one', 192),
 ('told', 186),
 ('guy', 178),
 ('trudeau', 177),
 ('deficit', 171),
 ('like', 169),
 ('people', 160),
 ('united', 151),
 ('also', 146),
 ('even', 138)]
```

In [192]:
```
freq_2019 = Counter(filtered_2019)
freq_2019 = freq_2019.most_common(20)
freq_2019
```

Out[192]:
```
[('trump', 1062),
 ('president', 482),
 ('said', 426),
 ('white', 197),
 ('new', 183),
 ('house', 180),
 ('court', 176),
 ('would', 170),
 ('state', 156),
 ('people', 141),
 ('year', 140),
 ('donald', 130),
 ('last', 126),
 ('mr', 110),
 ('campaign', 98),
 ('also', 92),
 ('support', 91),
 ('zervos', 90),
 ('case', 85),
 ('former', 82)]
```

In [193]:
```
freq_2020 = Counter(filtered_2020)
freq_2020 = freq_2020.most_common(20)
freq_2020
```

Out[193]:
```
[('trump', 1576),
 ('said', 633),
 ('president', 540),
 ('coronavirus', 420),
 ('would', 297),
 ('donald', 271),
 ('one', 241),
 ('people', 239),
 ('house', 216),
 ('told', 214),
 ('new', 212),
 ('time', 211),
 ('white', 210),
 ('year', 197),
 ('say', 194),
 ('health', 188),
 ('also', 180),
 ('official', 177),
 ('administration', 171),
 ('u', 168)]
```

In [194]:
```
freq_2021 = Counter(filtered_2021)
freq_2021 = freq_2021.most_common(20)
freq_2021
```

Out[194]:
```
[('trump', 1353),
 ('said', 516),
 ('president', 457),
 ('former', 275),
 ('republican', 242),
 ('would', 227),
 ('state', 221),
 ('vaccine', 179),
 ('call', 174),
 ('donald', 171),
 ('one', 165),
 ('get', 159),
 ('office', 158),
 ('news', 154),
 ('time', 153),
 ('election', 147),
 ('people', 144),
 ('house', 142),
 ('told', 133),
 ('biden', 130)]
```

In [195]:
```
freq_2022 = Counter(filtered_2022)
freq_2022 = freq_2022.most_common(20)
freq_2022
```

Out[195]:
```
[('trump', 1353),
 ('said', 516),
 ('president', 457),
 ('former', 275),
 ('republican', 242),
 ('would', 227),
 ('state', 221),
 ('vaccine', 179),
 ('call', 174),
 ('donald', 171),
 ('one', 165),
 ('get', 159),
 ('office', 158),
 ('news', 154),
 ('time', 153),
 ('election', 147),
 ('people', 144),
 ('house', 142),
 ('told', 133),
 ('biden', 130)]
```

In [196]:
```
freq_2023 = Counter(filtered_2023)
freq_2023 = freq_2023.most_common(20)
freq_2023
```

Out[196]:
```
[('trump', 2564),
 ('president', 492),
 ('republican', 482),
 ('would', 480),
 ('one', 473),
 ('new', 400),
 ('said', 371),
 ('donald', 369),
 ('people', 346),
 ('american', 339),
 ('time', 311),
 ('like', 306),
 ('political', 296),
 ('even', 292),
 ('former', 292),
 ('party', 276),
 ('state', 270),
 ('say', 260),
 ('year', 255),
 ('u', 232)]
```

In [197]:
```
type(filtered_2016)
```

Out[197]:
```
list
```

Here I have found the word freqencies within the lists each year, Interesting insights can be pulled out, with new words introdiced each year depending on the subject of more webites/articles then.

In his first year of presidency, 2017, the introduction of tax was added to the most common words.

2018 saw Trudeau added because of the altercation between the two of them at the G7 summit, and repercussions of that.

In 2019 there was the case of Summer Zeveros who accused Trump of sexual misconduct and so the words Zerveros and court were brought in.

In 2020, the introduction of coronavirus and virus is seen, whcih reflects the state of the world at the time.

2021 sees vaccine and Fauci, who was Trump's advisor during the pandemic.

2023 sees the introduction of former and a significant jump in th frequency of the word

In [141]:
```
filtered_lists = filtered_2016 + filtered_2017 + filtered_2018 + filte

nt occurrences of each word across all lists
_counts = Counter(all_filtered_lists)

 the 20 most common words
30_words = word_counts.most_common(30)

nt the top 20 most common words
word, count in top_30_words:
print(word, count)
```

VAD modelling

In [143]:
```
words = []
emo = []

for i in filtered_2016:
    if i in vad.index:
        emo.append(vad.loc[i])
        words.append(i)
    else:
        pass
```

In [146]:
```
vad_2016 = pd.DataFrame(emo, index = words)
average_vad_2016 = pd.DataFrame(emo).mean()
average_vad_2016
```

Out[146]:
```
valence      0.591861
arousal      0.518860
dominance    0.578840
dtype: float64
```

In [147]:
```
words = []
emo = []

for i in filtered_2017:
    if i in vad.index:
        emo.append(vad.loc[i])
        words.append(i)
    else:
        pass
```

In [148]:
```
vad_2017 = pd.DataFrame(emo, index = words)
average_vad_2017 = pd.DataFrame(emo).mean()
average_vad_2017
```

Out[148]:
```
valence      0.582069
arousal      0.524651
dominance    0.566854
dtype: float64
```

In [149]:
```
words = []
emo = []

for i in filtered_2018:
    if i in vad.index:
        emo.append(vad.loc[i])
        words.append(i)
    else:
        pass
```

In [150]:
```
vad_2018 = pd.DataFrame(emo, index = words)
average_vad_2018 = pd.DataFrame(emo).mean()
average_vad_2018
```

Out[150]:
```
valence      0.586854
arousal      0.519408
dominance    0.574211
dtype: float64
```

In [152]:
```
words = []
emo = []

for i in filtered_2019:
    if i in vad.index:
        emo.append(vad.loc[i])
        words.append(i)
    else:
        pass
```

In [155]:
```
vad_2019 = pd.DataFrame(emo, index = words)
average_vad_2019 = pd.DataFrame(emo).mean()
average_vad_2019
```

Out[155]:
```
valence      0.558836
arousal      0.534122
dominance    0.554222
dtype: float64
```

In [156]:
```
words = []
emo = []

for i in filtered_2020:
    if i in vad.index:
        emo.append(vad.loc[i])
        words.append(i)
    else:
        pass
```

In [157]:
```
vad_2020 = pd.DataFrame(emo, index = words)
average_vad_2020 = pd.DataFrame(emo).mean()
average_vad_2020
```

Out[157]:
```
valence      0.577908
arousal      0.522620
dominance    0.561058
dtype: float64
```

In [159]:
```
words = []
emo = []

for i in filtered_2021:
    if i in vad.index:
        emo.append(vad.loc[i])
        words.append(i)
    else:
        pass
```

In [160]:
```
vad_2021 = pd.DataFrame(emo, index = words)
average_vad_2021 = pd.DataFrame(emo).mean()
average_vad_2021
```

Out[160]:
```
valence      0.589420
arousal      0.523905
dominance    0.572191
dtype: float64
```

In [162]:
```
words = []
emo = []

for i in filtered_2022:
    if i in vad.index:
        emo.append(vad.loc[i])
        words.append(i)
    else:
        pass
```

In [163]:
```
vad_2022 = pd.DataFrame(emo, index = words)
average_vad_2022 = pd.DataFrame(emo).mean()
average_vad_2022
```

Out[163]:
```
valence      0.589420
arousal      0.523905
dominance    0.572191
dtype: float64
```

In [164]:
```
words = []
emo = []

for i in filtered_2023:
    if i in vad.index:
        emo.append(vad.loc[i])
        words.append(i)
    else:
        pass
```

In [165]:
```
vad_2023 = pd.DataFrame(emo, index = words)
average_vad_2023 = pd.DataFrame(emo).mean()
average_vad_2023
```

Out[165]:
```
valence      0.580365
arousal      0.526838
dominance    0.570649
dtype: float64
```

There are no significant trends of the VAD analysis.

Valence scores range from 0.558 (2020) to 0.592 (2019) over the years, which we could assume that the lowest score from 2020 which would indicate a less positive (but still positive) news articles, websites etc, that were published during the Coronavirus.

The arousal scores range from 0.519 to 0.524 which represents the level of excitment associated with the language in that subject, the earlier years, 2016 etc, have a less positive value than those in later years.

Dominance scores range from 0.554 to 0.578, with a notable jump from 0.574 in 2019 to 0.554 in 2020, indicating a change in controlled language used in articles from these years.

TF-IDF

In [200]:
```
all_filtered_lists = filtered_2016 + filtered_2017 + filtered_2018 + f

# Convert the list of tokenized words into strings
all_texts = [''.join(tokens) for tokens in all_filtered_lists]

# Create a TF-IDF vectorizer
tfidf_vectorizer = TfidfVectorizer()

# Fit the vectorizer to the data and transform the data
tfidf_matrix = tfidf_vectorizer.fit_transform(all_texts)

# Print the shape of the TF-IDF matrix
print("Shape of TF-IDF matrix:", tfidf_matrix.shape)
```

```
Shape of TF-IDF matrix: (362487, 21337)
```

This indicates that the TF-IDF matrix contains 362,,487 documents (rows) and 21,337 unique words (columns)

The values in the matrix represent the TF-IDF score for each word in each document. TF-IDF

Topic Modelling

In [205]:
```
lda_model = LatentDirichletAllocation(n_components=8, random_state=42)

# Fit the LDA model to the TF-IDF matrix
lda_model.fit(tfidf_matrix)

# Print the topics
print("Topics:")
feature_names = tfidf_vectorizer.get_feature_names_out()

for idx, topic in enumerate(lda_model.components_):
    print("Topic %d:" % (idx))
    print(" ".join([feature_names[i] for i in topic.argsort()[:-11:-1]]))
    print()
```

```
Topics:
Topic 0:
said president people way coronavirus america cnn court work biden

Topic 1:
state republican going back trade united great show asked candidate

Topic 2:
time year former also american even many election want good

Topic 3:
would know last day political think week made thing may

Topic 4:
new say house get two right mr story take presidential

Topic 5:
like campaign office york country call national see go million

Topic 6:
donald one told white could make party case washington much

Topic 7:
trump news first official according never still really number lot
```

Topic modelling can help discover underlying topics in the corpus by analyzing the TF-IDF matrix i completed before

Interpreting these results:

Topic 0: This topic looks at doscussions about the president and coronavirus. It would involve discussions about current events and political figures including Biden who was elected in 2020/2021

Topic 1: This topic may involve discussions about states, republicans and shows, it may relate to political campaigns and elections so most likely would have articles from 2016/2017 or 2020/21

Topic 2: This topic seems to involve discussions about historical events, past administeations and election cycles

Topic 3: This may incolve discussions about political perspectives and current affairs

Topic 4: This would involve statements, stories and actions related to new developments, houses and presidential matters that would relate to news coverage and actiosn taken by political figures

Topic 5: This would involve discussions around campaigns and countries with relationto international relations and political campaigns

Topic 6: This involves discussions about Trump, the White House, parties and cases

Topic 7: This topic involves more numeric in it regarding Trump.

Reflection on the process:

I spent much time researching different API's. Although for my first assignment i used the one from Genius.com, the limit that allowed me to take lyrics from the Genius website was very low, and meant it was taking a lot of time to actually get the data in the first place. After I attended the AI Sprint and we were introduced to Exa, I wanted to try and use their API. The free version had limits, however i got in contact with the founder who gave me a code to get the unlimited version for free. Exa does have a Beta version where you can specify the type ofwebite you want it to collect data from, however when i tried to do this just for news articles, it was erroring.

I decided I wanted to look at the perception of Trump in the media over the years from when he was president to the current day, so i chose 2016-2023, and used the Exa API to collect 100 websites that had Trump in them from each of the years.

After collecting the data, I tokenized it, lemmatized it, removed stop words and punctuation, made it all lowercase, and removed all non-alpha characters.

I then found the most common words for each year and analysed which words were added and removed each year depending on world events, and what was going on with Trump. This included the coronavirus, Biden entering the presidential race, Fauci etc.

I then looked at VAD modelling, to see if there were changes over the years and evaluated this. Although there were no significant trends in the VAD modelling, there was some change within each, which can also be attributed to different events, and therefore the context of the websites and articles that were published during that year.

I then completed TF-IDF (topic frequency – inverse document frequency) which concatenated all my lists, and then made them into a matrix. From this I could have done multiple things such as clustering, topic modelling or classification. I chose to do topic modelling to see what topics were evident, for 8 topics. This gave some interesting insight, however, I initially thought there would be clustering for each year, however as elections are a repetitive cycle and certain things are discussed and talked about more frequently during specific times, it would topic model these together rather than the years. For example, in his first year of presidency there will be similarities in topics, election year as well will have similarities. There are some cases where there will be topics such as coronavirus which happened mostly all in 2020 (for the most part).

I went into this assignment, and topic, thinking that public opinion would have changed dramatically, as although he was strongly polarizing when he was initially elected against Hillary Clinton in 2016, he still won the vote and therefore was popular amongst a vast group of people. With all the trials and new information coming out about him over the past few years, I would have believed that some of the VAD scores would have changed, as well as more dramatic addition and subtraction in the common words. There has been a great increase in the frequency of using the word trump. In 2016 Trump was used 1896 times, compared to 2023 where it was 2547, and a low was in 2019 where it was only used just over 1000 times.

Overall, this was an interesting topic to choose, however i think greater knowledge of NLP would have aided in creating a more in depth analysis of public opinion of Trump from news articles.

[This contains 604 words]

In [ ]: