```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

```
import textblob
import requests
import bs4
```

```
url= requests.get("https://en.wikipedia.org/wiki/Economic_impact_of_the_COVID-19_pandemic")
```

```
from bs4 import BeautifulSoup
```

```
v1 = BeautifulSoup(url.content,'html')
```

```
v1 = v1.getText(strip = True)
```

```
v1
```

```
'Economic impact of the COVID-19 pandemic - WikipediaJump to contentMain menuMain menumove to sidebarhideNavigationMain pageContentsCurren
t eventsRandom articleAbout WikipediaContact usDonateContributeHelpLearn to editCommunity portalRecent changesUpload fileLanguagesLanguage
links are at the top of the page across from the title.SearchSearchCreate accountLog inPersonal toolsCreate accountLog inPages for logged
out editorslearn moreContributionsTalkContentsmove to sidebarhide(Top)1Background2Overall economic contractionToggle Overall economic cont
raction subsection2.1Economic recovery programmes3Population growth4Agriculture and foodToggle Agriculture and food subsection4.12022 food
crisis5Financial markets6ManufacturingToggle Manufacturing subsection6.12021 Car production crisis7The arts, entertainment and sportToggle
The arts, entertainment and sport subsection7.1Cinema7.2Sport7.3Television7.4Video games8Medicine9Publishing10RetailToggle Retail subsecti
```

```
import re
```

```
v1= re.sub(r'\[\d+\]', "",v1)
```

```
v1=re.sub(r'\[\w+\]',"",v1)
```

```
v1=re.sub('[,]+',"",v1)
```

```
v1=re.sub('[0-9]+',"",v1)
```

```
v1=re.sub('[?]+',"",v1)
```

```
v1=re.sub('[()]+',"",v1)
```

```
v1=re.sub('[|]+',"",v1)
```

```
v1=re.sub('[:]+',"",v1)
```

```
v1=re.sub('[/]+',"",v1)
```

```
v1=re.sub('[;]+',"",v1)
```

```
v1
```

```
'Economic impact of the COVID- pandemic - WikipediaJump to contentMain menuMain menumove to sidebarhideNavigationMain pageContentsCurrent
eventsRandom articleAbout WikipediaContact usDonateContributeHelpLearn to editCommunity portalRecent changesUpload fileLanguagesLanguage l
inks are at the top of the page across from the title.SearchSearchCreate accountLog inPersonal toolsCreate accountLog inPages for logged o
ut editorslearn moreContributionsTalkContentsmove to sidebarhideTopBackgroundOverall economic contractionToggle Overall economic contracti
on subsection.Economic recovery programmesPopulation growthAgriculture and foodToggle Agriculture and food subsection. food crisisFinancia
l marketsManufacturingToggle Manufacturing subsection. Car production crisisThe arts entertainment and sportToggle The arts entertainment
and sport subsection.Cinema.Sport.Television.Video gamesMedicinePublishingRetailToggle Retail subsection.Business closuresE-commerceRestau
```

```
import nltk
```

```
from nltk.tokenize import sent_tokenize, word_tokenize
```

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

```
sentence = sent_tokenize(v1)
```

```
sentence = sent_tokenize(v1)
```

```
len(sentence)
```

```
1325
```

```
word = word_tokenize(v1)
```

```
len(word)
```

```
30334
```

```
sentence = v1
sentence = sentence.lower()
```

```
from nltk.tokenize import word_tokenize
v11 = word_tokenize(sentence)
```

```
print(v11)
```

```
['economic', 'impact', 'of', 'the', 'covid-', 'pandemic', '-', 'wikipediajump', 'to', 'contentmain', 'menumain', 'menumove', 'to', 'sidebar
```

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

```
import nltk
from nltk.corpus import stopwords
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

```
# Load stop words from a file
stop_words = set(open("/content/StopWords_DatesandNumbers1.txt").read().split())
```

```
filtered_tokens = [token for token in v11 if token.lower() not in stop_words]
```

```
print(filtered_tokens)
```

```
['economic', 'impact', 'of', 'the', 'covid-', 'pandemic', '-', 'wikipediajump', 'to', 'contentmain', 'menumain', 'menumove', 'to', 'sidebar
```

```
clean_text_1 = filtered_tokens
```

```
print(clean_text_1)
```

```
['economic', 'impact', 'of', 'the', 'covid-', 'pandemic', '-', 'wikipediajump', 'to', 'contentmain', 'menumain', 'menumove', 'to', 'sidebar
```

```
# Load stop words from a file
stop_words = set(open("/content/StopWords_Generic1.txt").read().split())
```

```
clean_text_1 = [token for token in filtered_tokens if token.lower() not in stop_words]
```

```
clean_text_2 = clean_text_1
```

```
print(clean_text_2)
```

```
['economic', 'impact', 'covid-', 'pandemic', '-', 'wikipediajump', 'contentmain', 'menumain', 'menumove', 'sidebarhidenavigationmain', 'pag
```

```
# Load stop words from a file
stop_words = set(open("/content/StopWords_Geographic1.txt").read().split())
```

```
clean_text_2 = [token for token in clean_text_1 if token.lower() not in stop_words]
```

```
clean_text_3=clean_text_2
```

```
print(clean_text_3)
```

```
['economic', 'impact', 'covid-', 'pandemic', '-', 'wikipediajump', 'contentmain', 'menumain', 'menumove', 'sidebarhidenavigationmain', 'pag
```

```
stop_words = set(open("/content/StopWords_Currencies11.csv").read().split())
```

```
clean_text_3 = [token for token in clean_text_2 if token.lower() not in stop_words]
```

```
clean_text_4=clean_text_3
```

```
print(clean_text_4)
```

```
['economic', 'impact', 'covid-', 'pandemic', '-', 'wikipediajump', 'contentmain', 'menumain', 'menumove', 'sidebarhidenavigationmain', 'pag
```

```
# Load stop words from a file
stop_words = set(open("/content/StopWords_Auditor1.txt").read().split())
```

```
clean_text_4 = [token for token in clean_text_3 if token.lower() not in stop_words]
```

```
clean_text_5=clean_text_4
```

```
print(clean_text_5)
```

```
['economic', 'impact', 'covid-', 'pandemic', '-', 'wikipediajump', 'contentmain', 'menumain', 'menumove', 'sidebarhidenavigationmain', 'pag
```

```
# Load stop words from a file
stop_words = set(open("/content/StopWords_GenericLong1.txt").read().split())
```

```
clean_text_5 = [token for token in clean_text_4 if token.lower() not in stop_words]
```

```
clean_text_6=clean_text_5
```

```
print(clean_text_6)
```

```
['economic', 'impact', 'covid-', 'pandemic', '-', 'wikipediajump', 'contentmain', 'menumain', 'menumove', 'sidebarhidenavigationmain', 'pag
```

```
stop_words = set(open("/content/StopWords_Names1.txt").read().split())
```

```
clean_text_6 = [token for token in clean_text_5 if token.lower() not in stop_words]
```

```
clean_text_7=clean_text_6
```

```
print(clean_text_7)
```

```
['economic', 'impact', 'covid-', 'pandemic', '-', 'wikipediajump', 'contentmain', 'menumain', 'menumove', 'sidebarhidenavigationmain', 'pag
```

```
from nltk import sentiment
from textblob import TextBlob
from nltk.sentiment import SentimentIntensityAnalyzer
```

```
with open('/content/positive-words.txt', 'r') as f:
    positive_words = f.read().splitlines()
```

```
with open('/content/negative_words1.csv', 'r') as f:
    negative_words = f.read().splitlines()
```

```
positive_dict = {}
negative_dict = {}
```

```
for token in clean_text_7:
    if token in positive_words:
        if token in positive_dict:
            positive_dict[token] += 1
        else:
            positive_dict[token] = 1
    elif token in negative_words:
        if token in negative_dict:
            negative_dict[token] += -1
        else:
            negative_dict[token] = -1
```

```
# Print the positive and negative words dictionaries
print("Positive words:", positive_dict)
print("Negative words:", negative_dict)
```

```
Positive words: {'top': 2, 'recovery': 34, 'led': 11, 'significant': 15, 'helped': 2, 'protection': 5, 'relaxed': 1, 'exceed': 1, 'great':
Negative words: {'warp': -2, 'recession': -21, 'crash': -4, 'crisis': -61, 'instability': -2, 'outbreak': -42, 'disruption': -9, 'vulnerabl
```

```
positive_score = sum([1 for word in clean_text_7  if word.lower() in positive_words])
negative_score = sum([1 for word in clean_text_7   if word.lower() in negative_words])
```

```
print(positive_score)
print(negative_score)
```

```
279
785
```

```
# Calculate polarity and subjectivity scores
polarity_score = (positive_score - negative_score) /((positive_score + negative_score)+ 0.000001)
```

```
print(polarity_score)
```

```
-0.4755639093274775
```

```
subjectivity_score = (positive_score - negative_score) / len(clean_text_7)+ 0.000001
```

```
print(subjectivity_score)
```

```
-0.027723508246123497
```

```
sentence = nltk.sent_tokenize(v1)
```

```
num_sentence = len(sentence)
```

```
print(num_sentence)
```

```
1325
```

```
word = nltk.word_tokenize(v1)
```

```
num_word = len(word)
```

```
print(num_word)
```

```
30334
```

```
average_Sentence_length = num_word / num_sentence
```

```
print(average_Sentence_length)
```

```
22.893584905660376
```

```
import nltk
from nltk.corpus import cmudict
nltk.download('cmudict')
```

```
[nltk_data] Downloading package cmudict to /root/nltk_data...
[nltk_data]   Package cmudict is already up-to-date!
True
```

```
def count_complex_words(clean_text_7):
    d = cmudict.dict()
    words = clean_text_7
    num_complex = 0
    for word in clean_text_7:
        num_syllables = 3
        if word.lower() in d:
            num_syllables = max([len(list(y for y in x if y[-1].isdigit()))] for x in d[word.lower()])
        if num_syllables == 3 :
            num_complex += 1
    return num_complex
```

```
text1 = clean_text_7
complex_words =count_complex_words (text1)
print(complex_words)
```

```
8280
```

```
num_word_1= len(clean_text_7)
```

```
print(num_word_1)
```

```
18251
```

```
Percentage_Complex_words = complex_words /num_word
```

```
print(Percentage_Complex_words)
```

```
0.27296103382343245
```

```
Fog_Index = 0.4 *(average_Sentence_length + Percentage_Complex_words)
```

```
print(Fog_Index)
```

```
9.266618375793524
```

```
Average_Number_Words_Per_Sentence = num_word + num_sentence
```

```
print(Average_Number_Words_Per_Sentence)
```

```
31659
```

```
def num_syllables(word):
    d = cmudict.dict()
    if word.lower() in d:
        return max([len(list(y for y in x if y[-1].isdigit())) for x in d[word.lower()]])
    return 0
```

```
import re
```

```
def count_personal_pronouns(v1):
    pattern_1 = r'\b(I|we|my|ours|us)\b'
    word_9 = re.findall(pattern_1, v1, flags=re.IGNORECASE)
    word_9 = [match for match in word_9 if match.lower() != "us"]
    return len(word_9)



text2= v1
count = count_personal_pronouns(text2)
print(count)
```

```
17
```

```
def avg_word_len(clean_text_7):
    words = clean_text_7
    total_chars = sum(len(word) for word in clean_text_7)
    num_words = len(words)
    avg_len = total_chars / num_words
    return avg_len



text2 = clean_text_7
avg_word_len = avg_word_len(text2)
print(avg_word_len)
```

```
7.6703742260698045
```

Colab paid products - Cancel contracts here

✓ 0s completed at 11:39 PM