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

```
Mounted at /content/drive
```

Pre-processing

```
!pip install -qq -U mlxtend
```

```
|████████████████████████████████| 1.3 MB 3.9 MB/s
```

```
import pandas as pd
import numpy
from numpy.random import randint
import numpy as np
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, fpmax, fpgrowth
from mlxtend.frequent_patterns import association_rules
import nltk
from nltk.tokenize import word_tokenize
import re
from nltk.stem import PorterStemmer
from tqdm.auto import tqdm
import networkx as nx
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
from networkx.algorithms.flow import shortest_augmenting_path
nltk.download("punkt")
nltk.download("stopwords")
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
PORTER_STEMMER = PorterStemmer()
```

```
def clean_tweets(x, STOPWORDS):
    # Lowercase
    sentence = x.lower()

    # Remove all non-alphabets (punctuation, numbers, new-line characters and extra-spaces)
    sentence = re.sub('http[s]?://\S+', '', sentence)
    sentence = re.sub(r'([^a-zA-Z ]+?)', '', sentence)
    #print(sentence)
    #sentence = sentence.replace('\n', '')
    # Remove URLs
    sentence = sentence.replace("world health organization", "who")
    #print(sentence)
    # Remove double spacing
    #sentence = re.sub('\s+', ' ', sentence)
    tokenized_tweet = [word for word in word_tokenize(sentence) if word not in STOPWORDS]
    tokenized_tweet = [PORTER_STEMMER.stem(word) for word in tokenized_tweet]
    return tokenized_tweet
```

```
<>:6: DeprecationWarning: invalid escape sequence \S
<>:6: DeprecationWarning: invalid escape sequence \S
<>:6: DeprecationWarning: invalid escape sequence \S
<ipython-input-5-e9114ee358bb>:6: DeprecationWarning: invalid escape sequence \S
  sentence = re.sub('http[s]?://\S+', '', sentence)
```

```
pharma_username = ['pfizer']
#'JNJNews', 'AstraZeneca', 'GSK', 'Novartis'
public_username = ['CDCgov', 'IHSgov', 'GovCanHealth', 'GCIndigenous']
ngo_username = ['WHO']
```

```
file1 = '/content/drive/MyDrive/Project 2/data/ngo_who/'+'WHO'+'.csv'
# file2 = '/content/drive/MyDrive/Project 2/data/pharma_companies/'+'Novartis'+'.csv'
# file3 = '/content/drive/MyDrive/Project 2/data/pharma_companies/'+'JNJNews'+'.csv'
```

```
# file4 = '/content/drive/MyDrive/Project 2/data/pharma_companies/'+'GSK'+'.csv'
# file5 = '/content/drive/MyDrive/Project 2/data/pharma_companies/'+'AstraZeneca'+'.csv'
Sample_Tweets = pd.concat(
map(pd.read_csv, [file1]), ignore_index=True)
print(Sample_Tweets)
# dataFrame.to_csv('/content/drive/MyDrive/Project 2/gephi/public_health_agencies/public_merged.csv')
```

```
      24577  1212412667073302528  2020-01-01 16:38:00+00:00  1211948316570849281
      24578  1212403167931125761  2020-01-01 16:00:15+00:00  1212403167931125761
      24579  1212393982665404416  2020-01-01 15:23:45+00:00  1211948316570849281
      24580  1212346627165478914  2020-01-01 12:15:34+00:00  1211948316570849281

                                                  tweet  retweet_count  \
      0      Science has given us powerful tools to prevent...             17
      1      The #COVID19 pandemic is not over — but it can...             53
      2      The  #COVID19 Vaccine Delivery Partnership is ...             16
      3      WHO as part of the #COVID19 Vaccine Delivery P...             16
      4      Vaccine effectiveness is waning over time: thi...             25
      ...                                                  ...            ...
      24576  RT @DrTedros: As we enter 2020, we are startin...             66
      24577  Healthy resolution No. 6: Manage stress. 🧘🧘 ...            140
      24578  RT @UN_News_Centre: #HappyNewYear\n\nTo achiev...             87
      24579  Healthy resolution No. 5: Limit use of alcohol...            105
      24580  Healthy resolution No. 4: Say #NoTobacco \n🚭🚭🚭 ...            116

             like_count  reply_count  quote_count  \
      0              69           19            4
      1             119           43            4
      2              58            4            2
      3              59            6            1
      4              73           20            4
      ...           ...          ...          ...
      24576           0            0            0
      24577         328            6           11
      24578           0            0            0
      24579         285           10           16
      24580         246           10            8

                                             expanded_url language  \
      0      https://twitter.com/WHO/status/153899808600505...       en
      1                                https://bit.ly/3QZ0zsu       en
      2      https://twitter.com/WHO/status/154257791743099...       en
      3                                https://bit.ly/3neA7xu       en
      4      https://twitter.com/WHO/status/154214644940374...       en
      ...                                                  ...      ...
      24576                                              NaN       en
      24577  https://twitter.com/WHO/status/121241266707330...       en
      24578                                              NaN       en
      24579  https://twitter.com/WHO/status/121239398266540...       en
      24580  https://twitter.com/WHO/status/121234662716547...       en

             possibly_sensitive  in_reply_to_user_id username
      0                   False           14499829.0      WHO
      1                   False           14499829.0      WHO
      2                   False           14499829.0      WHO
      3                   False           14499829.0      WHO
      4                   False           14499829.0      WHO
      ...                   ...                  ...      ...
      24576               False                  NaN      WHO
      24577               False           14499829.0      WHO
      24578               False                  NaN      WHO
      24579               False           14499829.0      WHO
      24580               False           14499829.0      WHO

      [24581 rows x 13 columns]
```

```
Sample_Tweets = Sample_Tweets.sample(n=10)
```

```
Sample_Tweets.iloc[0]
```

```
      id                                              1542616744874967040
      created_at                                2022-06-30 21:11:07+00:00
      conversation_id                                 1542616744874967040
      tweet                 Children receive smaller doses of the #COVID19...
      retweet_count                                                       1
      like_count                                                          7
      reply_count                                                        15
      quote_count                                                         0
      expanded_url          https://twitter.com/CDCgov/status/154261674487...
      language                                                           en
      possibly_sensitive                                              False
      in_reply_to_user_id                                               NaN
```

```
username                                                CDCgov
Name: 0, dtype: object
```

```
stopwords_df = set(nltk.corpus.stopwords.words("english"));
# stopwords_df.add("rt")
# stopwords_df.add("amp")
# stopwords_df.add("get")
# stopwords_df.add("It")
print(stopwords_df)
```

```
{'hadn', 'have', 'did', 'or', 'a', 'very', 'what', 'to', 'does', 'other', 'wasn', "wasn't", 'how', 'your', 'his', 'its', 'tho
```

Word Cloud

```
df = None
Tweet_Texts=Sample_Tweets['tweet'].values

# Converting the text column as a single string for wordcloud
Tweets_String=str(Tweet_Texts)

# Tweet Text cleaning
import re

# Converting the whole text to lowercase
Tweet_Texts_Cleaned = Tweets_String.lower()

# Removing the twitter usernames from tweet string
Tweet_Texts_Cleaned=re.sub(r'@\w+', ' ', Tweet_Texts_Cleaned)

# Removing the URLS from the tweet string
Tweet_Texts_Cleaned=re.sub(r'http\S+', ' ', Tweet_Texts_Cleaned)


# Deleting everything which is not characters
Tweet_Texts_Cleaned = re.sub(r'[^a-z A-Z]', ' ',Tweet_Texts_Cleaned)

# Deleting any word which is less than 3-characters mostly those are stopwords
Tweet_Texts_Cleaned= re.sub(r'\b\w{1,2}\b', '', Tweet_Texts_Cleaned)

# Stripping extra spaces in the text
Tweet_Texts_Cleaned= re.sub(r' +', ' ', Tweet_Texts_Cleaned)

Tweet_Texts_Cleaned
# for username in pharma_username:

  # Sample_Tweets = pd.read_csv('/content/drive/MyDrive/Project 2/data/pharma_companies/'+username+'.csv')

  #only 10 rows
  # Sample_Tweets = Sample_Tweets[:10]

  #clean tweets
  # Extracting only the Tweet text from the data frame
  # Tweet_Texts=Sample_Tweets['tweet'].values

  # # Converting the text column as a single string for wordcloud
  # Tweets_String=str(Tweet_Texts)

  # # Tweet Text cleaning
  # import re

  # # Converting the whole text to lowercase
  # Tweet_Texts_Cleaned = Tweets_String.lower()

  # # Removing the twitter usernames from tweet string
  # Tweet_Texts_Cleaned=re.sub(r'@\w+', ' ', Tweet_Texts_Cleaned)

  # # Removing the URLS from the tweet string
  # Tweet_Texts_Cleaned=re.sub(r'http\S+', ' ', Tweet_Texts_Cleaned)


  # # Deleting everything which is not characters
  # Tweet_Texts_Cleaned = re.sub(r'[^a-z A-Z]', ' ',Tweet_Texts_Cleaned)

  # # Deleting any word which is less than 3-characters mostly those are stopwords
```

```
  # Tweet_Texts_Cleaned= re.sub(r'\b\w{1,2}\b', '', Tweet_Texts_Cleaned)

  # # Stripping extra spaces in the text
  # Tweet_Texts_Cleaned= re.sub(r' +', ' ', Tweet_Texts_Cleaned)

  # Tweet_Texts_Cleaned
```

```
' science has given powerful tools prevent detect and treat covid use them well the covid pandemic not over but can with vac
cinequity the covid vaccine delivery partnership doing this helping countries access funding facilitate delivery engaging wi
th political leaders providing technical assistance such advising targeting vaccination campaigns risk populations happynewy
ear nto achieve healthforall the will need million more nurses and midwives urging countries healthy resolution limit use al
cohol happynewyear healthy resolution say notobacco happynewyear '
```

```
print(Tweet_Texts_Cleaned)
```

```
 children receive smaller doses the covid vaccine based their age like the adult vaccine the children vaccine helps keep ther
```

```
# Plotting the wordcloud
# you can specify fonts, stopwords, background color and other options
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS

# Creating the custom stopwords
customStopwords=list(stopwords_df)

wordcloudimage = WordCloud(
                          max_words=100,
                          max_font_size=500,
                          font_step=2,
                          stopwords=customStopwords,
                          background_color='white',
                          width=1000,
                          height=720
                          ).generate(Tweet_Texts_Cleaned)

plt.figure(figsize=(15,7))
plt.axis("off")
plt.imshow(wordcloudimage)
wordcloudimage
plt.show()
```

ARM (Itemsets)

```
len(pd.read_csv(file1))
```

```
24581
```

```
from google.colab import output
output.enable_custom_widget_manager()
```

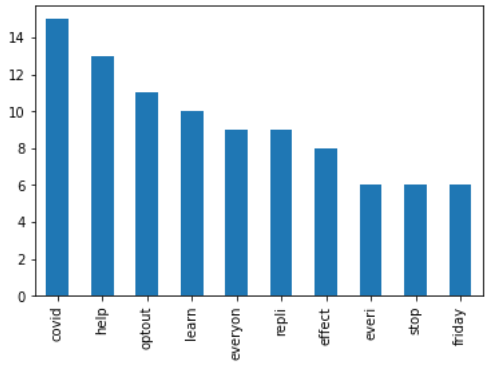
```
df = Sample_Tweets

#clean tweets
tqdm.pandas()
df["tokenized_tweet"] = df["tweet"].progress_apply(lambda x : clean_tweets(x, stopwords_df))
tokenized_tweets = df["tokenized_tweet"].values.tolist()
te = TransactionEncoder()
te_ary = te.fit(tokenized_tweets).transform(tokenized_tweets)
df = pd.DataFrame(te_ary, columns=te.columns_)
```

```
100%                                                    24581/24581 [00:20<00:00, 713.54it/s]
```

```
#Itemsets


min_sup=0.5
frequent_itemsets = apriori(df, min_support=min_sup, use_colnames=True)
print("len of frequent itemsets: ", len(frequent_itemsets.index))

while(len(frequent_itemsets.index)<100):
  min_sup = min_sup/2
  frequent_itemsets = apriori(df, min_support=min_sup, use_colnames=True)
  print("len of frequent itemsets: ", len(frequent_itemsets.index))

itemsets_df = frequent_itemsets[:100]
print(itemsets_df)
itemsets_df.to_csv("itemsets.csv");


# for username in pharma_username:

  # df = pd.read_csv('/content/drive/MyDrive/Project 2/data/pharma_companies/'+username+'.csv')

  #only 10 rows
  # df = df[:10]
  # df = Sample_Tweets

  # #clean tweets
  # tqdm.pandas()
  # df["tokenized_tweet"] = df["tweet"].progress_apply(lambda x : clean_tweets(x, stopwords_df))
  # tokenized_tweets = df["tokenized_tweet"].values.tolist()
  # te = TransactionEncoder()
  # te_ary = te.fit(tokenized_tweets).transform(tokenized_tweets)
  # df = pd.DataFrame(te_ary, columns=te.columns_)

  # #Itemsets


  # min_sup=0.5
  # frequent_itemsets = apriori(df, min_support=min_sup, use_colnames=True)
  # print("len of frequent itemsets: ", len(frequent_itemsets.index))

  # while(len(frequent_itemsets.index)<100):
  #   min_sup = min_sup/2
  #   frequent_itemsets = apriori(df, min_support=min_sup, use_colnames=True)
  #   print("len of frequent itemsets: ", len(frequent_itemsets.index))

  # itemsets_df = frequent_itemsets[:100]
  # print(itemsets_df)
  # itemsets_df.to_csv("itemsets.csv");
```

```
allItems = []
for i in itemsets_df['itemsets']:
    for j in i:
        allItems.append(j)

bar_df = pd.DataFrame(allItems, columns=['Items'])
print(bar_df['Items'].value_counts().head(10))
bar_df['Items'].value_counts().head(10).plot.bar()
```

```
covid      15
help       13
optout     11
learn      10
everyon     9
repli       9
effect      8
everi       6
stop        6
friday      6
Name: Items, dtype: int64
<matplotlib.axes._subplots.AxesSubplot at 0x7f72da24e590>
```



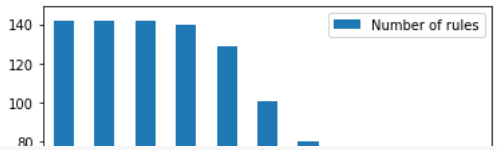

```
df
```

	a	aa	aackeri	aacvpr	aadncinfopubsaandccanadaca	aadubyk	aafccanada	aagotii	aaimcloud	aaip	...	zuzubear	zx
0	False	False	False	False	False	False	False	False	False	False	...	False	
1	False	False	False	False	False	False	False	False	False	False	...	False	
2	False	False	False	False	False	False	False	False	False	False	...	False	
3	False	False	False	False	False	False	False	False	False	False	...	False	
4	False	False	False	False	False	False	False	False	False	False	...	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	
65558	False	False	False	False	False	False	False	False	False	False	...	False	
65559	False	False	False	False	False	False	False	False	False	False	...	False	
65560	False	False	False	False	False	False	False	False	False	False	...	False	
65561	False	False	False	False	False	False	False	False	False	False	...	False	
65562	False	False	False	False	False	False	False	False	False	False	...	False	

65563 rows × 29903 columns

```
frequent_itemsets2 = apriori(df, min_support=0.125, use_colnames=True)
frequent_itemsets2

confidence = [0, .1, .2, .3, .4, .5, .6, .7, .8, .9, 1]
number_of_rules = []

for i in confidence:
    a = association_rules(frequent_itemsets2, metric="confidence", min_threshold=i)
    number_of_rules.append(len(a))

data = {'Confidence': confidence, 'Number of rules': number_of_rules}
df2 = pd.DataFrame(data=data)
print(df2)
plot = df2.plot.bar(x='Confidence', y='Number of rules')
```

```
    Confidence  Number of rules
0          0.0              142
1          0.1              142
2          0.2              142
3          0.3              140
4          0.4              129
5          0.5              101
6          0.6               80
7          0.7               69
8          0.8               55
9          0.9               46
10         1.0                7
```




```
association_rules(frequent_itemsets2, metric="confidence", min_threshold=0.5)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(learn)	(covid)	0.226805	0.423059	0.194988	0.859718	2.032147	0.099036	4.112712
1	(lt)	(covid)	0.169852	0.423059	0.169654	0.998833	2.360979	0.097796	494.216762
2	(repli)	(covid)	0.341443	0.423059	0.193249	0.565979	1.337825	0.048799	1.329293
3	(covid)	(vaccin)	0.423059	0.304135	0.275735	0.651765	2.143012	0.147068	1.998262
4	(vaccin)	(covid)	0.304135	0.423059	0.275735	0.906620	2.143012	0.147068	6.178415
...	...	...	...	...	...	...	...	...	...
96	(covid, repli)	(stop, vaccin)	0.193249	0.175770	0.174519	0.903078	5.137844	0.140552	8.504068
97	(covid, vaccin)	(stop, repli)	0.275735	0.224929	0.174519	0.632924	2.813887	0.112498	2.111474
98	(repli, vaccin)	(stop, covid)	0.178470	0.178805	0.174519	0.977865	5.468888	0.142608	37.099619
99	(repli)	(stop, covid, vaccin)	0.341443	0.175709	0.174519	0.511123	2.908920	0.114525	1.686091
100	(vaccin)	(stop, covid, repli)	0.304135	0.174519	0.174519	0.573821	3.288014	0.121442	1.936937

101 rows × 9 columns

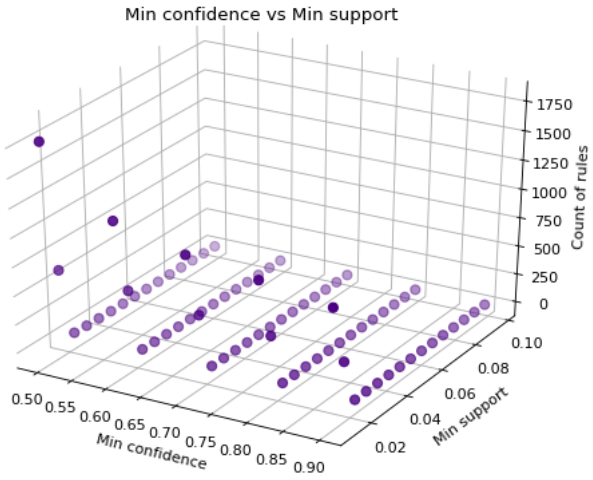
ARM (Rules)

```
matrix_df = pd.DataFrame(columns=['Threshold Support', 'Threshold Confidence', 'Count of rules'])
for min_support_initialize in np.arange(0.01, 0.1, 0.00625): #0.125, 0.5, 0.0625
  for min_threshold_initialize in np.arange(0.5, 1, 0.1):
    frequent_itemsets_temp = apriori(df, min_support=min_support_initialize, use_colnames=True)
    if(frequent_itemsets_temp.empty):
      continue
    rules = association_rules(frequent_itemsets_temp, metric="confidence", min_threshold=min_threshold_initialize)
  # rules = rules.sort_values(by='confidence', ascending =False)
  # print(rules)
    matrix_df.loc[len(matrix_df.index)] = [min_support_initialize, min_threshold_initialize, len(rules.index)]

print(matrix_df)
```

```
    Threshold Support  Threshold Confidence  Count of rules
0              0.0100                   0.5          1783.0
1              0.0100                   0.6          1252.0
2              0.0100                   0.7          1099.0
3              0.0100                   0.8          1022.0
4              0.0100                   0.9           930.0
..                ...                   ...             ...
70             0.0975                   0.5             1.0
71             0.0975                   0.6             0.0
72             0.0975                   0.7             0.0
73             0.0975                   0.8             0.0
74             0.0975                   0.9             0.0

[75 rows x 3 columns]
```

```
matrix_df.to_csv('Support vs Confidence for WHO.csv')
```

```
from IPython.core.pylabtools import figsize
# importing mplot3d toolkits, numpy and matplotlib
from mpl_toolkits import mplot3d
import numpy as np
import matplotlib.pyplot as plt
import itertools
fig = plt.figure(figsize=(8, 6), dpi=80)

# syntax for 3-D projection
ax = plt.axes(projection ='3d', )

# defining all 3 axes
z = matrix_df['Count of rules']
x = matrix_df['Threshold Confidence']
y = matrix_df['Threshold Support']

# plotting

ax.scatter(x, y, z, color='indigo', s=40)
ax.set_title('Min confidence vs Min support')
ax.set_xlabel('Min confidence')
ax.set_ylabel('Min support')

# First remove fill
ax.xaxis.pane.fill = False
ax.yaxis.pane.fill = False
ax.zaxis.pane.fill = False

# Now set color to white (or whatever is "invisible")
ax.xaxis.pane.set_edgecolor('w')
ax.yaxis.pane.set_edgecolor('w')
ax.zaxis.pane.set_edgecolor('w')

# ax.grid(False)

ax.set_zlabel('Count of rules')
plt.show()
```




```
# 0              0.1250                    0.5           101.0

# 4              0.01                   0.8         0
frequent_itemsets_temp = apriori(df, min_support=0.01, use_colnames=True)

rules = association_rules(frequent_itemsets_temp, metric="confidence", min_threshold=0.8)
rules[rules['lift']>=1]
print(rules)
```

```
                        antecedents  \
    0                     (variant)
    1                      (worker)
    2                     (pahowho)
    3                     (pahowho)
    4                     (pahowho)
    ...                         ...
```

```
1017  (whosearo, pahowho, drtedro)
1018            (whowpro, drtedro)
1019           (whoeurop, drtedro)
1020            (pahowho, drtedro)
1021           (whosearo, drtedro)

                                        consequents  antecedent support  \
0                                            (covid)            0.013303
1                                           (health)            0.023311
2                                          (whoafro)            0.024368
3                                          (whoemro)            0.024368
4                                         (whoeurop)            0.024368
...                                              ...                 ...
1017           (whoemro, whoafro, whoeurop, whowpro)            0.010333
1018  (whoemro, whoeurop, whoafro, pahowho, whosearo)            0.011147
1019   (whoemro, whowpro, whoafro, pahowho, whosearo)            0.010862
1020  (whoemro, whoeurop, whowpro, whoafro, whosearo)            0.010903
1021   (whoemro, whoeurop, whowpro, whoafro, pahowho)            0.010496

      consequent support   support  confidence       lift  leverage  \
0               0.405232  0.011554    0.868502   2.143222  0.006163
1               0.227656  0.019731    0.846422   3.717996  0.014424
2               0.037915  0.019812    0.813022  21.443011  0.018888
3               0.030186  0.019649    0.806344  26.712587  0.018914
4               0.026769  0.019812    0.813022  30.372168  0.019160
...                  ...       ...         ...        ...       ...
1017            0.019527  0.010089    0.976378  50.000722  0.009887
1018            0.019568  0.010089    0.905109  46.254670  0.009871
1019            0.019527  0.010089    0.928839  47.566230  0.009877
1020            0.019527  0.010089    0.925373  47.388744  0.009876
1021            0.019527  0.010089    0.961240  49.225517  0.009884

      conviction
0       4.523006
1       5.029016
2       5.145434
3       5.007919
4       5.205050
...          ...
1017   41.506679
1018   10.332245
1019   13.778222
1020   13.138334
1021   25.296196

[1022 rows x 9 columns]
```

```
type(rules)
```

```
pandas.core.frame.DataFrame
```

```
df1 = (rules.explode('antecedents')
         .reset_index(drop=True)
         .explode('consequents')
         .reset_index(drop=True))
df1.to_csv('WHO rules.csv')
```

Bokeh plot

[ ] ↳ *10 cells hidden*

Followers (Maybe not)

[ ] ↳ *6 cells hidden*

Metrics

Recommender systems can also profit from the messages shared on social media

The higher the popularity of a tweet, the more likely it is to contain highly supported association rule.

```
Sample_Tweets['Tweet_rank'] = Sample_Tweets['like_count']+Sample_Tweets['quote_count']+Sample_Tweets['reply_count']+Sample_Tweets[
```

```
Sample_Tweets = Sample_Tweets.sort_values(by=['Tweet_rank'], ascending=False)
```

```
Sample_Tweets["tokenized_tweet"] = Sample_Tweets["tweet"].progress_apply(lambda x : clean_tweets(x, stopwords_df))
```

```
100%                                    24581/24581 [00:16<00:00, 1747.44it/s]
```

```
Sample_Tweets
```

	id	created_at	conversation_id	tweet	retweet_count	like_count	reply_count	quote_
24303	1217043229427761152	2020-01-14 11:18:12+00:00	1217043229427761152	Preliminary investigations conducted by the Ch...	23372	29369	15873	
21955	1237777021742338049	2020-03-11 16:26:53+00:00	1237774421307228160	🚨 BREAKING 🚨 \n\n"We have therefore made the as...	51313	52822	2011	
21233	1243972193169616898	2020-03-28 18:44:17+00:00	1243972193169616898	FACT: #COVID19 is NOT airborne. \n\nThe #coron...	39560	44307	2809	
21970	1237721991471382528	2020-03-11 12:48:13+00:00	1237721991471382528	These are 7 simple steps to protect yourself a...	31615	37607	604	
14887	1313841832598687749	2020-10-07 14:01:17+00:00	1313841832598687749	We are thrilled to have @SuperM joining our Bi...	18175	45970	913	
...	...	...	...	...	...	...	...	
1086	1522099819740909570	2022-05-05 06:24:10+00:00	1522099819740909570	https://t.co/BmNCAD0jmf	0	0	0	
11649	1356994405111128072	2021-02-03 15:54:12+00:00	1356994405111128072	https://t.co/wSojwxGRa5	0	0	0	
19048	1262368554042654720	2020-05-18 13:04:51+00:00	1262368554042654720	https://t.co/FUUYdc2Iv0	0	0	0	
19046	1262369680397918208	2020-05-18 13:09:20+00:00	1262369680397918208	https://t.co/qKkZVnASif	0	0	0	
18943	1262742422582226948	2020-05-19 13:50:28+00:00	1262742422582226948	https://t.co/C7q61yGb8r	0	0	0	

24581 rows × 15 columns

```
Sample_Tweets["tokenized_tweet"]
```

```
24303    [preliminari, investig, conduct, chines, autho...
21955    [break, therefor, made, assess, covid, charact...
21233    [fact, covid, airborn, coronaviru, mainli, tra...
21970     [simpl, step, protect, other, covid, coronaviru]
14887    [thrill, superm, join, big, event, mental, hea...
                               ...
1086                                                    []
11649                                                   []
19048                                                   []
19046                                                   []
18943                                                   []
Name: tokenized_tweet, Length: 24581, dtype: object
```

```
type(Sample_Tweets['tokenized_tweet'])
```

```
pandas.core.series.Series
```

```
sorted_df = Sample_Tweets['tokenized_tweet']
```

```
sorted_df.to_csv('sorted.csv')
```

```
Sample_Tweets.iloc[0]['tweet']
```

```
'Preliminary investigations conducted by the Chinese authorities have found no clear evidence of human-to-human transmission
of the novel #coronavirus (2019-nCoV) identified in #Wuhan, #China🇨🇳. https://t.co/Fnl5P877VG'
```

```
rules['rank'] = rules['antecedent support']+rules['confidence']+rules['consequent support']+rules['leverage']+rules['lift']+rules[
```

```
sorted_rules = rules.sort_values(by=['rank'], ascending=False)
```

```
sorted_rules
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	rank
285	(whoafro, whowpro, pahowho)	(whosearo)	0.019609	0.023555	0.019609	1.000000	42.454231	0.019147	inf	inf
691	(whoemro, whoeurop, whowpro)	(whosearo, pahowho)	0.019527	0.019812	0.019527	1.000000	50.474333	0.019140	inf	inf
820	(whoeurop, whowpro, whoafro, pahowho, drtedro)	(whosearo)	0.010089	0.023555	0.010089	1.000000	42.454231	0.009851	inf	inf
365	(whoemro, whoafro, whowpro)	(whosearo)	0.019527	0.023555	0.019527	1.000000	42.454231	0.019067	inf	inf
363	(whoemro, whosearo, whowpro)	(whoafro)	0.019527	0.037915	0.019527	1.000000	26.374464	0.018787	inf	inf
...	...	...	...	...	...	...	...	...	...	...
127	(media, brief, drtedro)	(covid)	0.012855	0.405232	0.010821	0.841772	2.077261	0.005612	3.758936	7.112490
19	(pandem, vaccin)	(covid)	0.012001	0.405232	0.010008	0.833898	2.057831	0.005144	3.580748	6.904862
13	(media, brief)	(covid)	0.015337	0.405232	0.012774	0.832891	2.055346	0.006559	3.559169	6.887308
16	(coronaviru, countri)	(covid)	0.012083	0.405232	0.010008	0.828283	2.043974	0.005112	3.463651	6.768341

```
type(sorted_rules['antecedents'])
```

```
pandas.core.series.Series
```

Accuracy

Compare sorted_rules['antecedents']+sorted_rules['consequents]' with Sample_tweets['tokenized_tweet'] using iloc

```
type(list(sorted_rules.iloc[0]['antecedents']))
```

```
list
```

```
count=0
l1 = list(sorted_rules.iloc[0]['antecedents'])
for value in l1:
  if value in sorted_df.iloc[0]:
    count=count+1
l2 = list(sorted_rules.iloc[0]['consequents'])
if(count==len(l1)):
  for value in l2:
    if value in sorted_df.iloc[0]:
      count=count+1
if(count==len(l1)+len(l2)):
  print("exists")
else:
  print("does not exist")
```

```
exists
```

```
len(sorted_rules.index)
```

```
1022
```

```
len(sorted_df.index)
```

```
24581
```

```
sorted_df
```

```
24303    [preliminari, investig, conduct, chines, autho...
21955    [break, therefor, made, assess, covid, charact...
21233    [fact, covid, airborn, coronaviru, mainli, tra...
21970     [simpl, step, protect, other, covid, coronaviru]
14887    [thrill, superm, join, big, event, mental, hea...
                               ...
1086                                                    []
11649                                                   []
19048                                                   []
19046                                                   []
18943                                                   []
Name: tokenized_tweet, Length: 24581, dtype: object
```

```
counter=0

save_df = pd.DataFrame(columns=['Tweet ID (iloc)', 'tweet', 'tokenized', 'Rule No.', 'rule', 'Username'])
# matrix_df.loc[len(matrix_df.index)] = [min_support_initialize, min_threshold_initialize, len(rules.index)]
for i in range (0, 2458, 1): #data
  for j in range(0, 1022, 1): #rules
    count=0
    l1 = list(sorted_rules.iloc[j]['antecedents'])
    for value in l1:
      if value in sorted_df.iloc[i]:
        count=count+1
    l2 = list(sorted_rules.iloc[j]['consequents'])
    if(count==len(l1)):
      for value in l2:
        if value in sorted_df.iloc[i]:
          count=count+1
    if(count==len(l1)+len(l2)):
      save_df.loc[len(save_df.index)] = [i, Sample_Tweets.iloc[i]['tweet'], Sample_Tweets.iloc[i]["tokenized_tweet"], j, list(sort
      # print("Tweet ", i, " contains rule number ", j)
      counter=counter+1
# for i in range(0, 5, 1):
#   for j in range(0, 75798, 1):
#     count=0
#     l1 = list(sorted_rules.iloc[j]['antecedents'])
#     for value in l1:
#       if value in sorted_df.iloc[i]:
#         count=count+1
#     l2 = list(sorted_rules.iloc[j]['consequents'])
#     if(count==len(l1)):
#       for value in l2:
#         if value in sorted_df.iloc[i]:
#           count=count+1
#     if(count==len(l1)+len(l2)):
#       print(j, " exists at ", i)
#       counter=counter+1
#     else:
#       print("does not exist")
```

```
save_df
```

	Tweet ID (iloc)	tweet	tokenized	Rule No.	rule	Username
0	15	RT @DrTedros: Thank you, @BTS_twt, for includi...	[rt, drtedro, thank, btstwt, includ, sign, lan...	1010	[thank, rt, drtedro]	WHO

```
counter
```

```
46841
```

```
save_df.to_csv('final WHO.csv')
```

```
counter_0to20=0
counter_21to40=0
counter_41to60=0
counter_61to80=0
counter_81to100=0
counter_remaining=0
x=0
i=0
for index, row in save_df.iterrows():
  if(row['Tweet ID (iloc)'])<=500:
    counter_0to20=counter_0to20+1
  elif(row['Tweet ID (iloc)'])<=1000:
    counter_21to40=counter_21to40+1
  elif(row['Tweet ID (iloc)'])<=1500:
    counter_41to60=counter_41to60+1
  elif(row['Tweet ID (iloc)'])<=2000:
    counter_61to80=counter_61to80+1
  else:
    x=x+1
```

```
print(counter_0to20, " ", counter_21to40, " ", counter_41to60, " ", counter_61to80, " ", x)
```

```
6846   14851   11967   5480   7697
```

```
import matplotlib.pyplot as plt
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
lim = ['500', '1000', '1500', '2000', '2500']
val = [counter_0to20,counter_21to40,counter_41to60,counter_61to80,x]
ax.bar(lim,val, color = 'orange')
plt.show()
```