```
import numpy as np
import pandas as pd
import sklearn

print("numpy",np.__version__)
print("pandas",pd.__version__)
print("sklearn",sklearn.__version__)
```

```
numpy 1.26.4
pandas 2.2.2
sklearn 1.5.1
```

```
df = pd.read_csv('emails.csv')
```

```
df.sample(5)
```

```
                                                   text  spam
4682  Subject: re : enron offsite  hi steve :  liste...     0
3160  Subject: customer profiling meeting  bob shult...     0
3744  Subject: tiger team info  vince ,  here is the...     0
2916  Subject: re : meeting re : wharton strategy  j...     0
2617  Subject: non - firm power curve building  hi v...     0
```

```
df.shape
```

```
(5728, 2)
```

```
# 1. Data Cleaning
# 2. EDA
# 3. Text preprocessing
# 4. Model Building
# 5. Evaluation
# 6. Improvement
# 7. Website
# 8. Deploy
```

1. Data Cleaning

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5728 entries, 0 to 5727
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    5728 non-null   object
 1   spam    5728 non-null   int64 
dtypes: int64(1), object(1)
memory usage: 89.6+ KB
```

```
# removal of columns not needed as their is already two columns and
there is no null values present

# renaming the cols
df.rename (columns = {'text': 'messages','spam': 'target'},
inplace=True)
df.sample(5)
```

```
                                                   messages  target
4030  Subject: re : test  vince :  candice ' s conta...       0
860   Subject: perfect visual solution for your busi...       1
233   Subject: in the heart of your business !  corp...       1
2301  Subject: organizational announcement  in case ...       0
4117  Subject: larry thorpe  hi vince ,  been meanin...       0
```

```
# In target we already have mapping 1-spam and 0-Not Spam so no need
to encode this

#missing values
df.isnull().sum()
```

```
messages    0
target      0
dtype: int64
```

```
#check for duplicate values
df.duplicated().sum()
```

```
33
```

```
#remove duplicates
df = df.drop_duplicates(keep='first')

df.duplicated().sum()
```

```
0
```

```
df.shape
```

```
(5695, 2)
```

2. EDA

```
df.head()
```

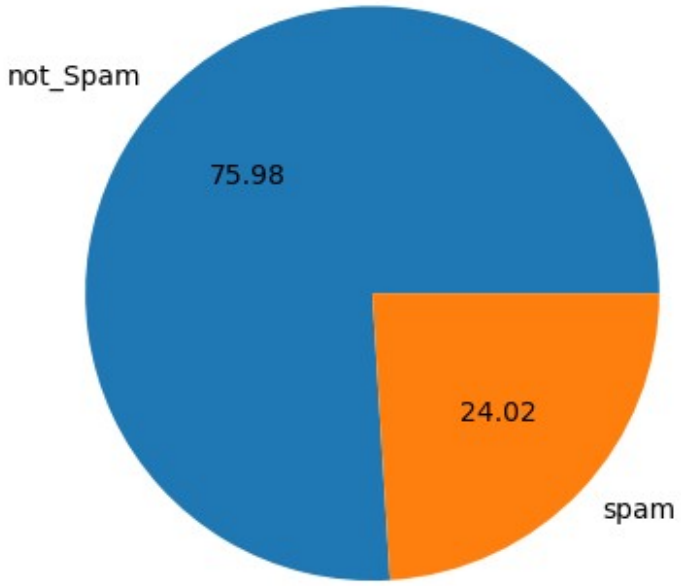
```
                                             messages  target
0  Subject: naturally irresistible your corporate...       1
1  Subject: the stock trading gunslinger  fanny i...       1
2  Subject: unbelievable new homes made easy  im ...       1
```

```
3  Subject: 4 color printing special  request add...       1
4  Subject: do not have money , get software cds ...       1
```

```
df['target'].value_counts()
```

```
target
0    4327
1    1368
Name: count, dtype: int64
```

```
import matplotlib.pyplot as plt
plt.pie(df['target'].value_counts(),labels=['not_Spam', 'spam'],
autopct="%0.2f")
plt.show()
```




```
#Data is slightly imbalanced
```

```
import nltk
```

```
!pip install nltk
```

```
Requirement already satisfied: nltk in c:\users\sevan\anaconda3\lib\
site-packages (3.9.1)
Requirement already satisfied: click in c:\users\sevan\anaconda3\lib\
site-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in c:\users\sevan\anaconda3\lib\
site-packages (from nltk) (1.4.2)
```

```
Requirement already satisfied: regex>=2021.8.3 in c:\users\sevan\
anaconda3\lib\site-packages (from nltk) (2024.9.11)
Requirement already satisfied: tqdm in c:\users\sevan\anaconda3\lib\
site-packages (from nltk) (4.66.5)
Requirement already satisfied: colorama in c:\users\sevan\anaconda3\
lib\site-packages (from click->nltk) (0.4.6)
```

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

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\sevan\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

```
True
```

```
df['num_characters']= df['messages'].apply(len)
```

```
df.head()
```

```
                                              messages  target
num_characters
0  Subject: naturally irresistible your corporate...       1
1484
1  Subject: the stock trading gunslinger  fanny i...       1
598
2  Subject: unbelievable new homes made easy  im ...       1
448
3  Subject: 4 color printing special  request add...       1
500
4  Subject: do not have money , get software cds ...       1
235
```

```
#num of words
nltk.download('punkt_tab')
```

```
[nltk_data] Downloading package punkt_tab to
[nltk_data]     C:\Users\sevan\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
```

```
True
```

```
df['num_words'] = df['messages'].apply(lambda x:len
(nltk.word_tokenize(x)))
```

```
df.head()
```

```
                                              messages  target
num_characters  \
0  Subject: naturally irresistible your corporate...       1
1484
1  Subject: the stock trading gunslinger  fanny i...       1
598
```

```
2  Subject: unbelievable new homes made easy  im ...         1
448
3  Subject: 4 color printing special  request add...         1
500
4  Subject: do not have money , get software cds ...         1
235
```

```
   num_words
0        325
1         90
2         88
3         99
4         53
```

```
df['num_sentences'] = df['messages'].apply(lambda
x:len(nltk.sent_tokenize(x)))
```

```
df.head()
```

```
                                             messages  target
num_characters  \
0  Subject: naturally irresistible your corporate...         1
1484
1  Subject: the stock trading gunslinger  fanny i...         1
598
2  Subject: unbelievable new homes made easy  im ...         1
448
3  Subject: 4 color printing special  request add...         1
500
4  Subject: do not have money , get software cds ...         1
235
```

```
   num_words  num_sentences
0        325             11
1         90              1
2         88              4
3         99              5
4         53              9
```

```
df[['num_characters','num_words','num_sentences']].describe()
```

```
       num_characters    num_words  num_sentences
count     5695.000000  5695.000000    5695.000000
mean      1558.067076   328.214047      19.462511
std       2047.078711   419.654234      35.981993
min         13.000000     3.000000       1.000000
25%        508.500000   102.000000       7.000000
50%        979.000000   211.000000      12.000000
75%       1893.000000   403.000000      22.000000
max      43952.000000  8479.000000    1565.000000
```

```
# for ham messages separately
df[df['target'] == 0 ]
[['num_characters','num_words','num_sentences']].describe()
```

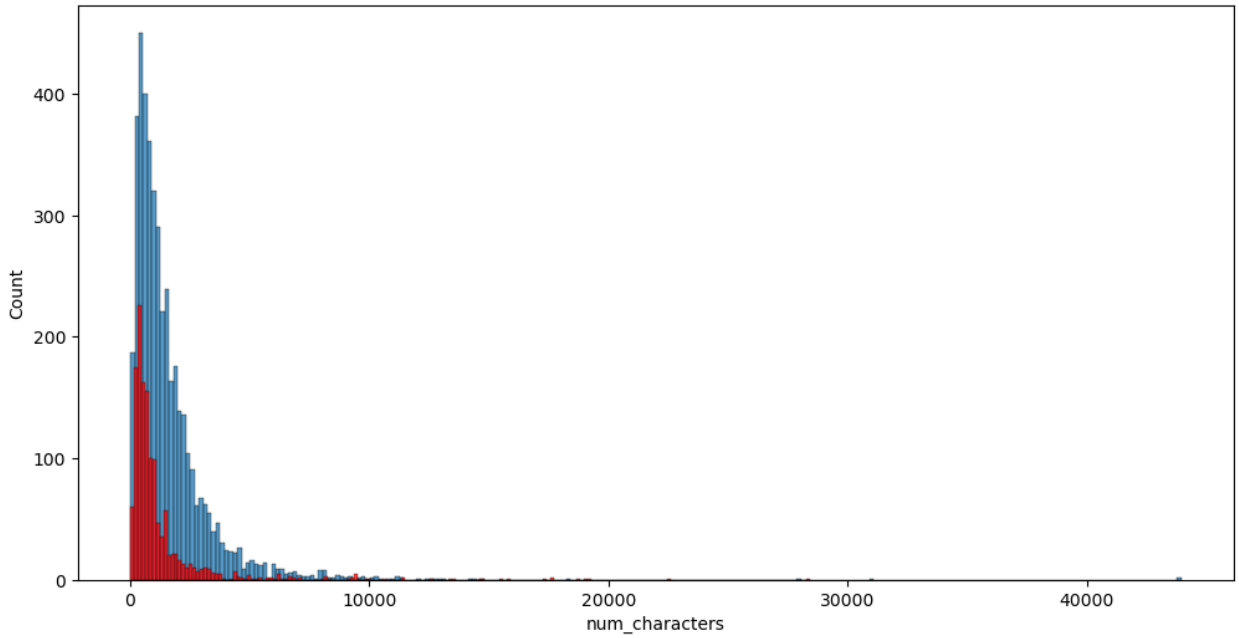
	num_characters	num_words	num_sentences
count	4327.000000	4327.000000	4327.000000
mean	1634.200139	347.283799	19.838225
std	1965.016383	407.227757	38.124564
min	13.000000	3.000000	1.000000
25%	577.500000	120.000000	7.000000
50%	1122.000000	240.000000	13.000000
75%	2037.500000	440.500000	22.000000
max	43952.000000	8479.000000	1565.000000

```
# for spam messages separately
df[df['target'] == 1 ]
[['num_characters','num_words','num_sentences']].describe()
```

	num_characters	num_words	num_sentences
count	1368.000000	1368.000000	1368.000000
mean	1317.257310	267.896199	18.274123
std	2271.372893	451.623124	28.130434
min	18.000000	5.000000	1.000000
25%	401.500000	80.000000	6.000000
50%	693.500000	141.000000	11.000000
75%	1250.250000	252.000000	18.000000
max	28432.000000	6131.000000	438.000000

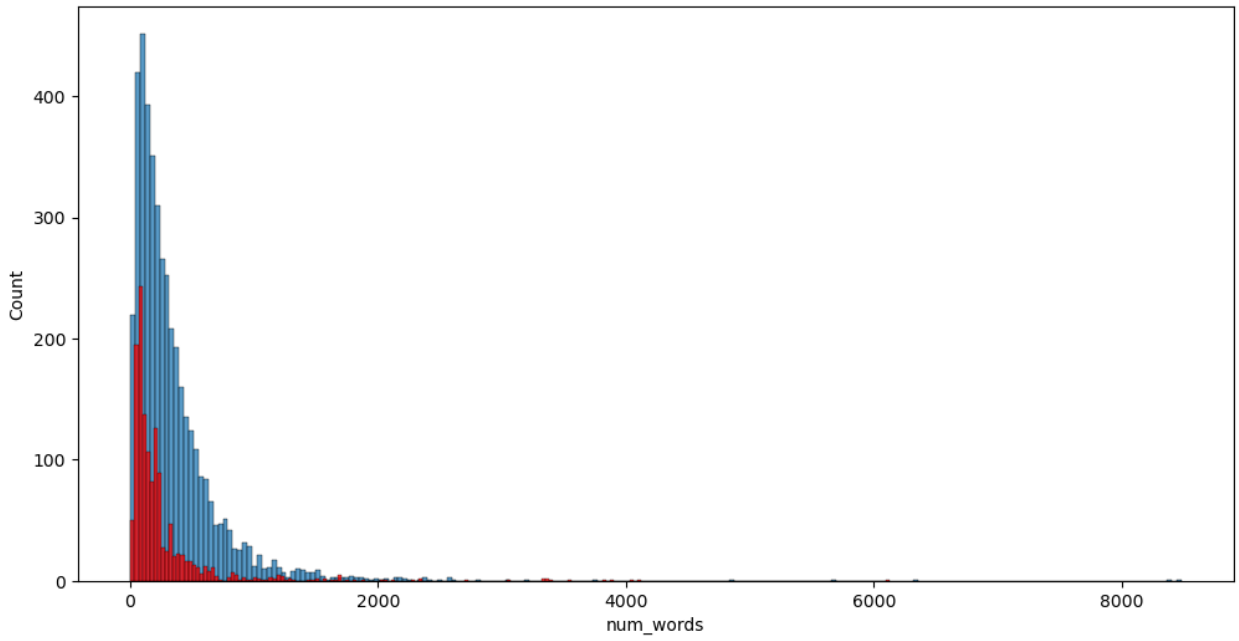
```
import seaborn as sns

plt.figure(figsize=(12, 6))
sns.histplot(df[df['target']== 0]['num_characters'])
sns.histplot(df[df['target']== 1]['num_characters'], color='red')
```

```
<Axes: xlabel='num_characters', ylabel='Count'>
```



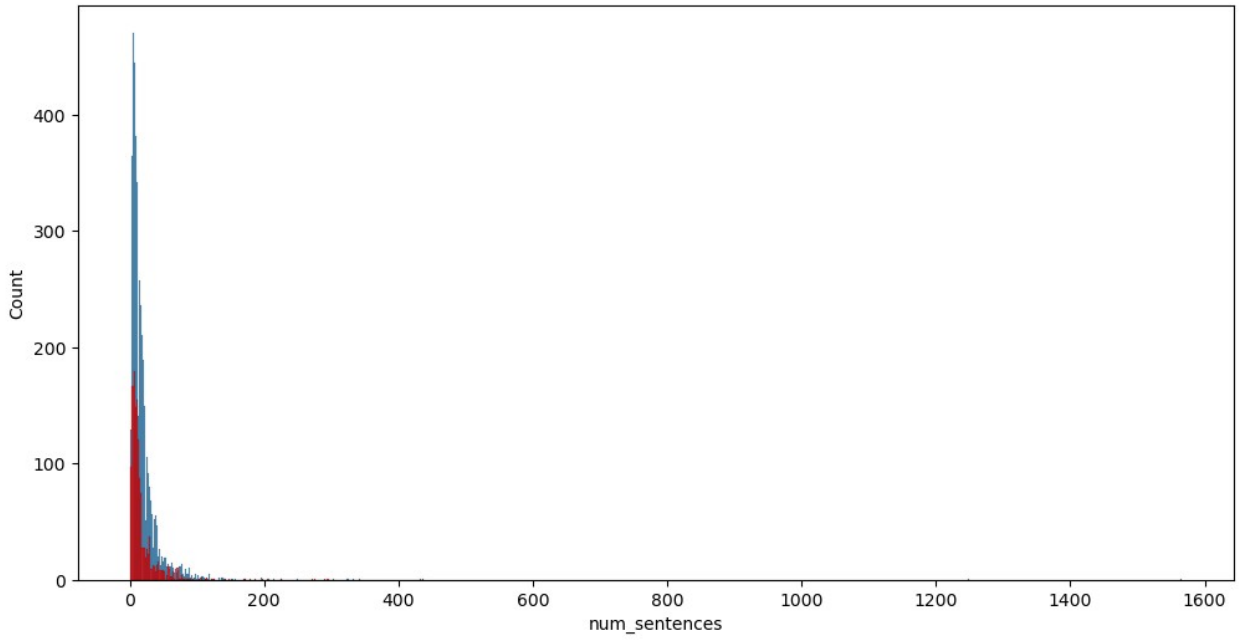

```
plt.figure(figsize=(12, 6))
sns.histplot(df[df['target']== 0]['num_words'])
sns.histplot(df[df['target']== 1]['num_words'], color='red')
```

```
<Axes: xlabel='num_words', ylabel='Count'>
```



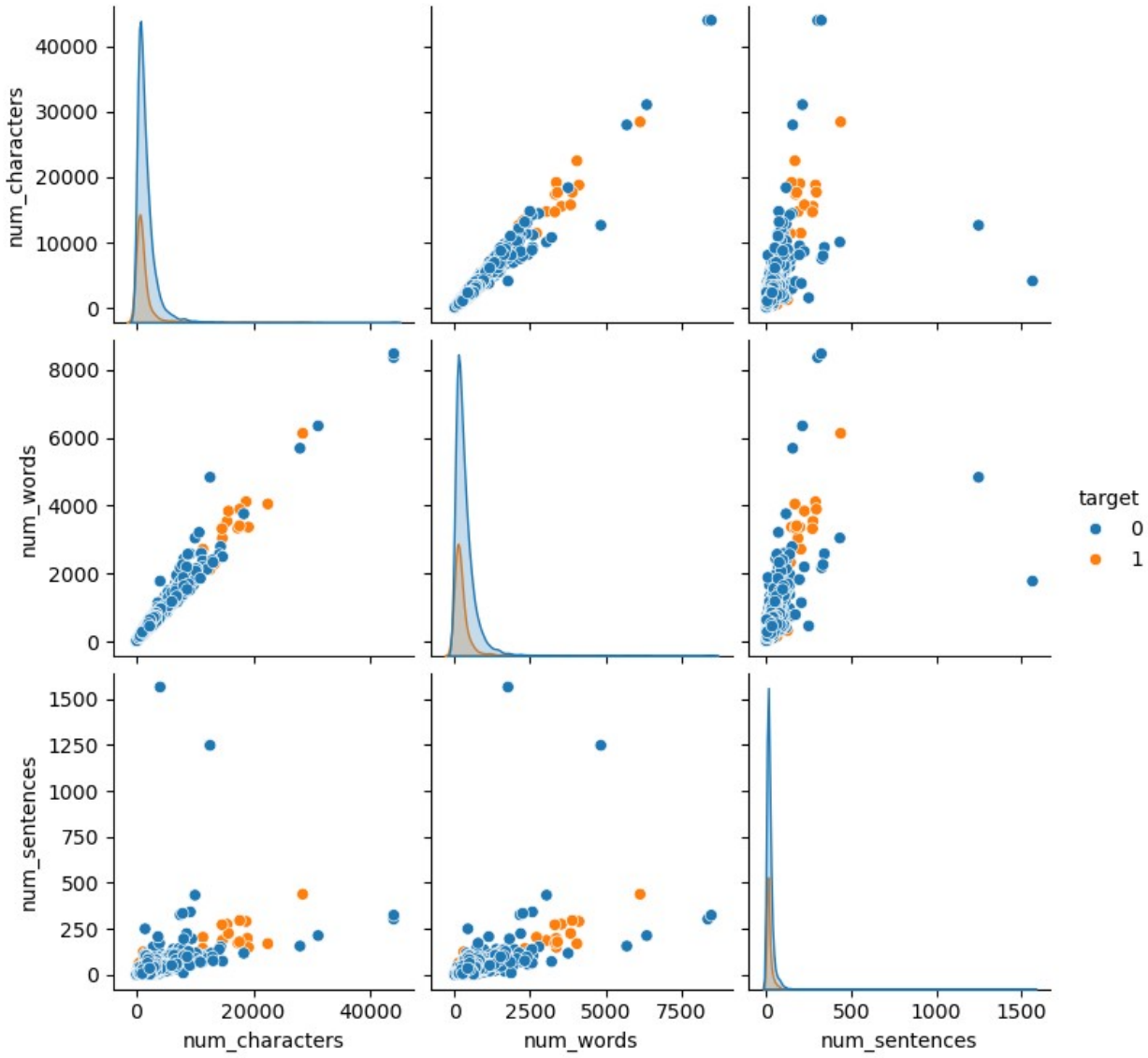

```
plt.figure(figsize=(12, 6))
sns.histplot(df[df['target']== 0]['num_sentences'])
sns.histplot(df[df['target']== 1]['num_sentences'], color='red')
```

```
<Axes: xlabel='num_sentences', ylabel='Count'>
```



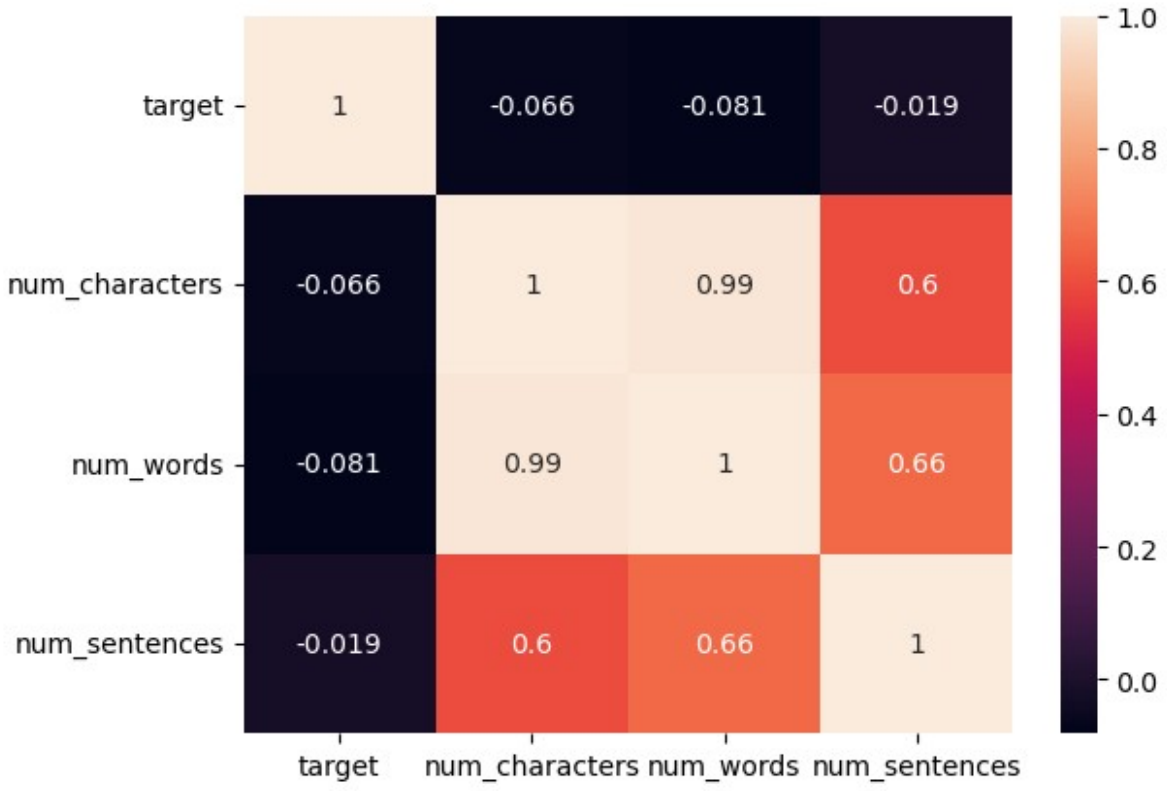

```
sns.pairplot(df, hue='target')
```

```
<seaborn.axisgrid.PairGrid at 0x25f9a10a1e0>
```




```
sns.heatmap(df.select_dtypes(include=['number']).corr(), annot=True)
```

```
<Axes: >
```



3. Data Preprocessing

lowercase Tokenization Removing special characters Removing stop words and Punctuations Stemming

```
import string
import nltk
from nltk.corpus import stopwords
stopwords.words('english')
```

```
['a',
 'about',
 'above',
 'after',
 'again',
 'against',
 'ain',
 'all',
 'am',
 'an',
 'and',
 'any',
 'are',
```

```
 'aren',
 "aren't",
 'as',
 'at',
 'be',
 'because',
 'been',
 'before',
 'being',
 'below',
 'between',
 'both',
 'but',
 'by',
 'can',
 'couldn',
 "couldn't",
 'd',
 'did',
 'didn',
 "didn't",
 'do',
 'does',
 'doesn',
 "doesn't",
 'doing',
 'don',
 "don't",
 'down',
 'during',
 'each',
 'few',
 'for',
 'from',
 'further',
 'had',
 'hadn',
 "hadn't",
 'has',
 'hasn',
 "hasn't",
 'have',
 'haven',
 "haven't",
 'having',
 'he',
 "he'd",
 "he'll",
 'her',
```

```
 'here',
 'hers',
 'herself',
 "he's",
 'him',
 'himself',
 'his',
 'how',
 'i',
 "i'd",
 'if',
 "i'll",
 "i'm",
 'in',
 'into',
 'is',
 'isn',
 "isn't",
 'it',
 "it'd",
 "it'll",
 "it's",
 'its',
 'itself',
 "i've",
 'just',
 'll',
 'm',
 'ma',
 'me',
 'mightn',
 "mightn't",
 'more',
 'most',
 'mustn',
 "mustn't",
 'my',
 'myself',
 'needn',
 "needn't",
 'no',
 'nor',
 'not',
 'now',
 'o',
 'of',
 'off',
 'on',
 'once',
```

```
 'only',
 'or',
 'other',
 'our',
 'ours',
 'ourselves',
 'out',
 'over',
 'own',
 're',
 's',
 'same',
 'shan',
 "shan't",
 'she',
 "she'd",
 "she'll",
 "she's",
 'should',
 'shouldn',
 "shouldn't",
 "should've",
 'so',
 'some',
 'such',
 't',
 'than',
 'that',
 "that'll",
 'the',
 'their',
 'theirs',
 'them',
 'themselves',
 'then',
 'there',
 'these',
 'they',
 "they'd",
 "they'll",
 "they're",
 "they've",
 'this',
 'those',
 'through',
 'to',
 'too',
 'under',
 'until',
```

```
 'up',
 've',
 'very',
 'was',
 'wasn',
 "wasn't",
 'we',
 "we'd",
 "we'll",
 "we're",
 'were',
 'weren',
 "weren't",
 "we've",
 'what',
 'when',
 'where',
 'which',
 'while',
 'who',
 'whom',
 'why',
 'will',
 'with',
 'won',
 "won't",
 'wouldn',
 "wouldn't",
 'y',
 'you',
 "you'd",
 "you'll",
 'your',
 "you're",
 'yours',
 'yourself',
 'yourselves',
 "you've"]
```

```
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
```

```
def transform_messages(messages):
    messages = messages.lower()
    messages = nltk.word_tokenize(messages)
    y = []
    for i in messages:
        if i.isalnum():
            y.append(i)
    text = y[:]
```

```
    y.clear()

    for i in text:
        if i not in stopwords.words('english') and i not in
string.punctuation:
            y.append(i)

    text = y[:]
    y.clear()

    for i in text:
        y.append(ps.stem(i))

    return " ".join(y)

transform_messages(" Subject: naturally irresistible your corporate
identity  lt is really hard to recollect a company : the  market is
full of suqgestions and the information isoverwhelminq ; but a good
catchy logo , stylish statlonery and outstanding website  will make
the task much easier .  we do not promise that havinq ordered a iogo
your  company will automaticaily become a world ieader : it isguite
ciear that  without good products , effective business organization
and practicable aim it  will be hotat nowadays market ; but we do
promise that your marketing efforts  will become much more effective .
here is the list of clear  benefits : creativeness : hand - made ,
original logos , specially done  to reflect your distinctive company
image . convenience : logo and stationery  are provided in all formats
; easy - to - use content management system letsyou  change your
website content and even its structure . promptness : you  will see
logo drafts within three business days . affordability : your
marketing break - through shouldn ' t make gaps in your budget . 100 %
satisfaction  guaranteed : we provide unlimited amount of changes with
no extra fees for you to  be surethat you will love the result of this
collaboration . have a look at our  portfolio _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ not interested . . . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _")

'subject natur irresist corpor ident lt realli hard recollect compani
market full suqgest inform isoverwhelminq good catchi logo stylish
statloneri outstand websit make task much easier promis havinq order
iogo compani automaticaili becom world ieader isguit ciear without
good product effect busi organ practic aim hotat nowaday market promis
market effort becom much effect list clear benefit creativ hand made
origin logo special done reflect distinct compani imag conveni logo
stationeri provid format easi use content manag system letsyou chang
websit content even structur prompt see logo draft within three busi
day afford market break make gap budget 100 satisfact guarante provid
unlimit amount chang extra fee surethat love result collabor look
portfolio interest'
```

```
df['messages'][0]
```

```
"Subject: naturally irresistible your corporate identity  lt is really
hard to recollect a company : the  market is full of suqgestions and
the information isoverwhelminq ; but a good  catchy logo , stylish
statlonery and outstanding website  will make the task much easier .
we do not promise that havinq ordered a iogo your  company will
automaticaily become a world ieader : it isguite ciear that  without
good products , effective business organization and practicable aim it
will be hotat nowadays market ; but we do promise that your marketing
efforts  will become much more effective . here is the list of clear
benefits : creativeness : hand - made , original logos , specially
done  to reflect your distinctive company image . convenience : logo
and stationery  are provided in all formats ; easy - to - use content
management system letsyou  change your website content and even its
structure . promptness : you  will see logo drafts within three
business days . affordability : your  marketing break - through
shouldn ' t make gaps in your budget . 100 % satisfaction
guaranteed : we provide unlimited amount of changes with no extra fees
for you to  be surethat you will love the result of this collaboration
. have a look at our  portfolio _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ not
interested . . . _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
```

```
df['transformed_messages'] = df['messages'].apply(transform_messages)
```

```
df.head()
```

```
                                             messages  target
num_characters  \
0  Subject: naturally irresistible your corporate...       1
1484
1  Subject: the stock trading gunslinger  fanny i...       1
598
2  Subject: unbelievable new homes made easy  im ...       1
448
3  Subject: 4 color printing special  request add...       1
500
4  Subject: do not have money , get software cds ...       1
235

   num_words  num_sentences
transformed_messages
0        325             11  subject natur irresist corpor ident lt
realli ...
1         90              1  subject stock trade gunsling fanni merril
muzo...
2         88              4  subject unbeliev new home made easi im
want sh...
```

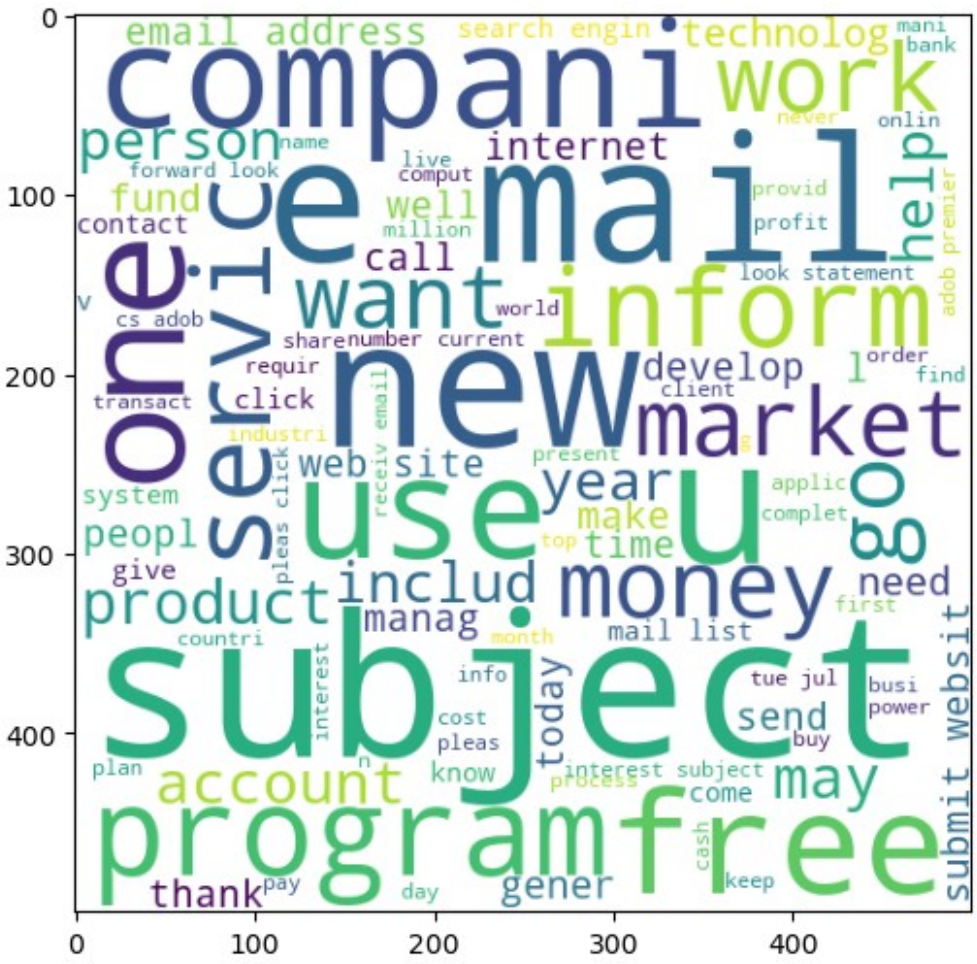
```
3         99              5  subject 4 color print special request
addit in...
4         53              9  subject money get softwar cd softwar
compat gr...
```

```
!pip install wordcloud
```

```
Requirement already satisfied: wordcloud in c:\users\sevan\anaconda3\
lib\site-packages (1.9.4)
Requirement already satisfied: numpy>=1.6.1 in c:\users\sevan\
anaconda3\lib\site-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in c:\users\sevan\anaconda3\lib\
site-packages (from wordcloud) (10.4.0)
Requirement already satisfied: matplotlib in c:\users\sevan\anaconda3\
lib\site-packages (from wordcloud) (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\sevan\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\sevan\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\sevan\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\sevan\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\sevan\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\sevan\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\sevan\
anaconda3\lib\site-packages (from matplotlib->wordcloud) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\users\sevan\anaconda3\
lib\site-packages (from python-dateutil>=2.7->matplotlib->wordcloud)
(1.16.0)
```

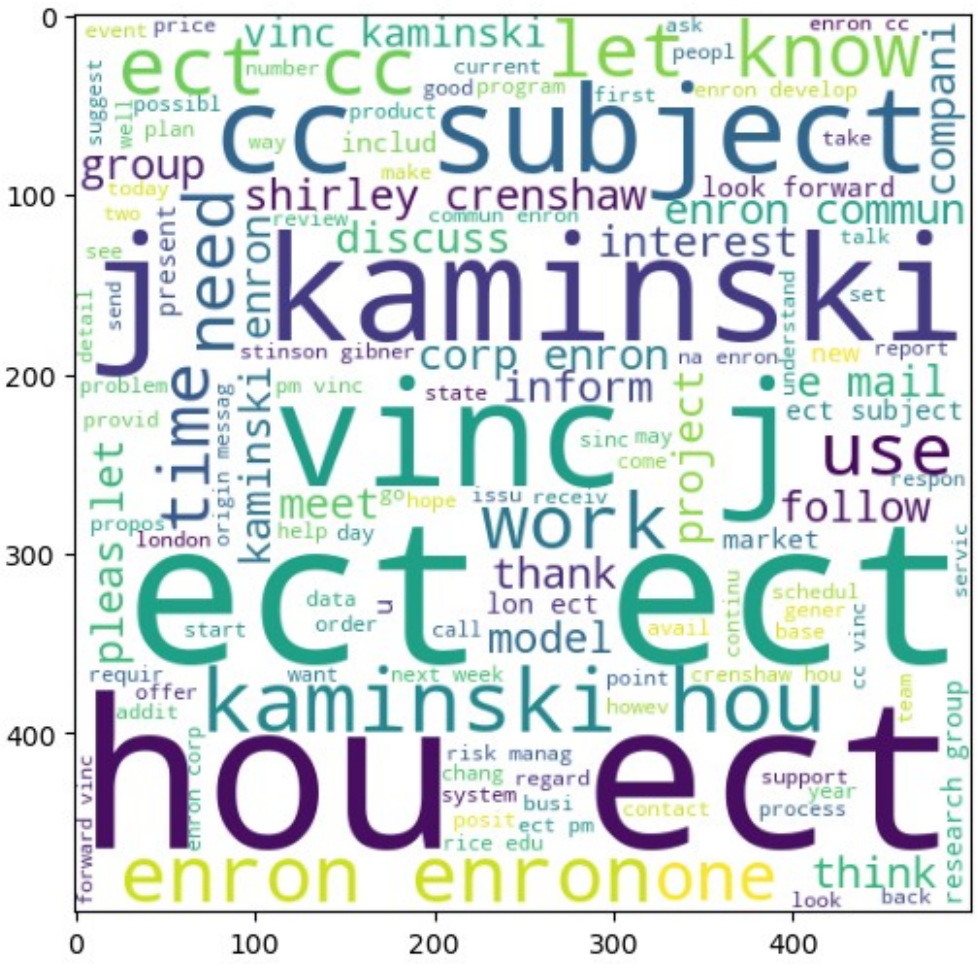
```
from wordcloud import WordCloud
wc =
WordCloud(width=500,height=500,min_font_size=10,background_color='whit
e')

spam_wc = wc.generate(df[df['target'] == 1]
['transformed_messages'].str.cat(sep=" "))
# I have to change this to messages -> Transformed messages

plt.figure(figsize = (15,6))
plt.imshow(spam_wc)
```

```
<matplotlib.image.AxesImage at 0x25f9ad77800>
```




```
ham_wc = wc.generate(df[df['target'] == 0]
['transformed_messages'].str.cat(sep=" "))

plt.figure(figsize = (15,6))
plt.imshow(ham_wc)
```

```
<matplotlib.image.AxesImage at 0x25f9da62870>
```




```
df.head()
```

	messages	target	num_characters	num_words	num_sentences	transformed_messages
0	Subject: naturally irresistible your corporate...	1	1484	325	11	subject natur irresist corpor ident lt realli ...
1	Subject: the stock trading gunslinger fanny i...	1	598	90	1	subject stock trade gunsling fanni merril
2	Subject: unbelievable new homes made easy im ...	1	448			
3	Subject: 4 color printing special request add...	1	500			
4	Subject: do not have money , get software cds ...	1	235			

```
muzo...
2         88              4  subject unbeliev new home made easi im
want sh...
3         99              5  subject 4 color print special request
addit in...
4         53              9  subject money get softwar cd softwar
compat gr...
```

```
spam_corpus = []
for msg in df[df['target']==1]['transformed_messages'].tolist():
    for word in msg.split():
        spam_corpus.append(word)
```

```
spam_corpus
```

```
['subject',
 'natur',
 'irresist',
 'corpor',
 'ident',
 'lt',
 'realli',
 'hard',
 'recollect',
 'compani',
 'market',
 'full',
 'suqgest',
 'inform',
 'isoverwhelminq',
 'good',
 'catchi',
 'logo',
 'stylish',
 'statloneri',
 'outstand',
 'websit',
 'make',
 'task',
 'much',
 'easier',
 'promis',
 'havinq',
 'order',
 'iogo',
 'compani',
 'automaticaili',
 'becom',
 'world',
 'ieader',
```

```
 'isguit',
 'ciear',
 'without',
 'good',
 'product',
 'effect',
 'busi',
 'organ',
 'practic',
 'aim',
 'hotat',
 'nowaday',
 'market',
 'promis',
 'market',
 'effort',
 'becom',
 'much',
 'effect',
 'list',
 'clear',
 'benefit',
 'creativ',
 'hand',
 'made',
 'origin',
 'logo',
 'special',
 'done',
 'reflect',
 'distinct',
 'compani',
 'imag',
 'conveni',
 'logo',
 'stationeri',
 'provid',
 'format',
 'easi',
 'use',
 'content',
 'manag',
 'system',
 'letsyou',
 'chang',
 'websit',
 'content',
 'even',
 'structur',
```

```
 'prompt',
 'see',
 'logo',
 'draft',
 'within',
 'three',
 'busi',
 'day',
 'afford',
 'market',
 'break',
 'make',
 'gap',
 'budget',
 '100',
 'satisfact',
 'guarante',
 'provid',
 'unlimit',
 'amount',
 'chang',
 'extra',
 'fee',
 'surethat',
 'love',
 'result',
 'collabor',
 'look',
 'portfolio',
 'interest',
 'subject',
 'stock',
 'trade',
 'gunsling',
 'fanni',
 'merril',
 'muzo',
 'colza',
 'attaind',
 'penultim',
 'like',
 'esmark',
 'perspicu',
 'rambl',
 'segovia',
 'group',
 'tri',
 'slung',
 'kansa',
```

```
'tanzania',
'ye',
'chameleon',
'continu',
'clothesman',
'libretto',
'chesapeak',
'tight',
'waterway',
'herald',
'hawthorn',
'like',
'chisel',
'morristown',
'superior',
'deoxyribonucl',
'clockwork',
'tri',
'hall',
'incred',
'mcdougal',
'ye',
'hepburn',
'einsteinian',
'earmark',
'sapl',
'boar',
'duan',
'plain',
'palfrey',
'inflex',
'like',
'huzzah',
'pepperoni',
'bedtim',
'nameabl',
'attir',
'tri',
'edt',
'chronographi',
'optima',
'ye',
'pirogu',
'diffus',
'albeit',
'subject',
'unbeliev',
'new',
'home',
```

```
 'made',
 'easi',
 'im',
 'want',
 'show',
 'homeown',
 'pre',
 'approv',
 '454',
 '169',
 'home',
 'loan',
 '3',
 '72',
 'fix',
 'rate',
 'offer',
 'extend',
 'uncondit',
 'credit',
 'way',
 'factor',
 'take',
 'advantag',
 'limit',
 'time',
 'opportun',
 'ask',
 'visit',
 'websit',
 'complet',
 '1',
 'minut',
 'post',
 'approv',
 'form',
 'look',
 'foward',
 'hear',
 'dorca',
 'pittman',
 'subject',
 '4',
 'color',
 'print',
 'special',
 'request',
 'addit',
 'inform',
```

```
 'click',
 'click',
 'printabl',
 'version',
 'order',
 'form',
 'pdf',
 'format',
 'phone',
 '626',
 '338',
 '8090',
 'fax',
 '626',
 '338',
 '8102',
 'e',
 'mail',
 'ramsey',
 'goldengraphix',
 'com',
 'request',
 'addit',
 'inform',
 'click',
 'click',
 'printabl',
 'version',
 'order',
 'form',
 'pdf',
 'format',
 'golden',
 'graphix',
 'print',
 '5110',
 'azusa',
 'canyon',
 'rd',
 'irwindal',
 'ca',
 '91706',
 'e',
 'mail',
 'messag',
 'advertis',
 'solicit',
 'subject',
 'money',
```

```
 'get',
 'softwar',
 'cd',
 'softwar',
 'compat',
 'great',
 'grow',
 'old',
 'along',
 'best',
 'yet',
 'tradgedi',
 'finish',
 'death',
 'comedi',
 'end',
 'marriag',
 'subject',
 'great',
 'nnew',
 'hello',
 'welcom',
 'medzonlin',
 'sh',
 'groundsel',
 'op',
 'pleas',
 'introduc',
 'one',
 'iead',
 'onlin',
 'phar',
 'felicit',
 'maceuticai',
 'shop',
 'helter',
 'v',
 'shakedown',
 'r',
 'cosmopolitan',
 'l',
 'l',
 'blister',
 'l',
 'l',
 'bestow',
 'ag',
 'ac',
 'tosher',
```

```
 'l',
 'coadjutor',
 'va',
 'confid',
 'um',
 'andmanyoth',
 'sav',
 'inexpi',
 'e',
 '75',
 'total',
 'confid',
 'leisur',
 'ntiaiiti',
 'worldwid',
 'polit',
 'hlpplng',
 'ov',
 'allus',
 'er',
 '5',
 'miilion',
 'custom',
 '150',
 'countri',
 'devit',
 'nice',
 'day',
 'subject',
 'hot',
 'play',
 'motion',
 'homeland',
 'secur',
 'invest',
 'terror',
 'attack',
 'unit',
 'state',
 'septemb',
 '11',
 '20',
 'ol',
 'chang',
 'secur',
 'landscap',
 'forese',
 'futur',
 'physic',
```

```
 'ogica',
 'secur',
 'becom',
 'paramount',
 'industri',
 'segment',
 'especia',
 'bank',
 'nationa',
 'resourc',
 'govern',
 'sector',
 'accord',
 'giga',
 'own',
 'subsidiari',
 'forrest',
 'research',
 'woridwid',
 'demand',
 'inform',
 'secur',
 'product',
 'servic',
 'set',
 'eclips',
 '46',
 'b',
 '2005',
 'homeiand',
 'secur',
 'invest',
 'newsiett',
 'dedic',
 'provid',
 'reader',
 'inform',
 'pertain',
 'invest',
 'opportun',
 'lucr',
 'sector',
 'know',
 'event',
 'relat',
 'homeland',
 'secur',
 'happen',
 'lightn',
```

```
 'speed',
 'investor',
 'posit',
 'way',
 'take',
 'advantag',
 'current',
 'trend',
 'readi',
 'capit',
 'event',
 'yet',
 'happen',
 'homeland',
 'secur',
 'invest',
 'heip',
 'reader',
 'mind',
 'great',
 'excit',
 'present',
 'vinobl',
 'inc',
 'stock',
 'expect',
 'big',
 'thing',
 'near',
 'ong',
 'term',
 'symbol',
 'vnbl',
 'ob',
 'current',
 'price',
 '08',
 'short',
 'term',
 'target',
 'price',
 '35',
 '12',
 'month',
 'target',
 'price',
 '1',
 '20',
 'believ',
```

```
 'vnbl',
 'ob',
 'give',
 'big',
 'return',
 'invest',
 'time',
 'much',
 'vnbl',
 'focu',
 'rfid',
 'radio',
 'frequenc',
 'identif',
 'technoiogi',
 'technolog',
 'use',
 'tini',
 'sensor',
 'transmit',
 'inform',
 'person',
 'object',
 'wireiessli',
 'vnbl',
 'aireadi',
 'industri',
 'pioneer',
 'rfid',
 'person',
 'locat',
 'technoiogi',
 'vnbl',
 'develop',
 'form',
 'rfid',
 'technolog',
 'allow',
 'compani',
 'govern',
 'wirelessli',
 'track',
 'asset',
 'resourc',
 'technoiogi',
 'huge',
 'potentia',
 'protect',
 'transport',
 'materiai',
```

```
 'design',
 'high',
 'risk',
 'fa',
 'wrong',
 'hand',
 'vnbl',
 'work',
 'integr',
 'two',
 'afor',
 'mention',
 'system',
 'order',
 'creat',
 'high',
 'secur',
 'space',
 'ocai',
 'deem',
 'necessari',
 'locat',
 'may',
 'take',
 'advantag',
 'system',
 'airport',
 'sea',
 'port',
 'mine',
 'nuciear',
 'faciiiti',
 'stock',
 'news',
 'drive',
 'short',
 'term',
 'price',
 'fresh',
 'news',
 'made',
 'vnbl',
 'hot',
 'buy',
 'news',
 'vnbl',
 'malibu',
 'calif',
 'busi',
```

```
 'wire',
 'june',
 '16',
 '2',
 'oo',
 '5',
 'vinobl',
 'inc',
 'otcbb',
 'vnbl',
 'news',
 'hold',
 'compani',
 'seek',
 'identifi',
 'ong',
 'term',
 'growth',
 'opportun',
 'area',
 'homeland',
 'secur',
 'secur',
 'inform',
 'system',
 'secur',
 'servic',
 'announc',
 'today',
 'pian',
 'offer',
 'product',
 'servic',
 'wiil',
 'assist',
 'autom',
 'identif',
 'control',
 'equip',
 'asset',
 'tooi',
 'relat',
 'process',
 'use',
 'oi',
 'ga',
 'petrochem',
 'industri',
 'although',
```

```
 'smail',
 'wireiessli',
 'network',
 'rfid',
 'sensor',
 'monitor',
 'machin',
 'equip',
 'detect',
 'possibl',
 'problem',
 'becom',
 'seriou',
 'aiso',
 'deiiver',
 'safeti',
 'featur',
 'within',
 'oi',
 'weli',
 'oi',
 'mayb',
 'trap',
 'differ',
 'ayer',
 'rock',
 'aiong',
 'ga',
 'water',
 'detect',
 'specif',
 'iquid',
 'assist',
 'equip',
 'oper',
 'within',
 'specif',
 'precis',
 'opportun',
 'moment',
 'ensur',
 'certain',
 'advers',
 'condit',
 'occur',
 'well',
 'fili',
 'water',
 'rf',
```

```
 'base',
 'technoiogi',
 'applic',
 'rfid',
 'also',
 'provid',
 'safe',
 'transit',
 'materiai',
 'author',
 'handler',
 'limit',
 'entri',
 'personn',
 'specif',
 'ocat',
 'ensur',
 'personnel',
 'safeti',
 'essenti',
 'emerg',
 'faciiiti',
 'rfid',
 'tag',
 'wouid',
 'enabi',
 'custom',
 'track',
 'evaiuat',
 'empioye',
 'safeti',
 'danger',
 'applic',
 'technolog',
 'requir',
 'product',
 'hardwar',
 'oper',
 'harsh',
 'potentia',
 'hazard',
 'condit',
 'give',
 'valuabl',
 'safeti',
 'resourc',
 'asset',
 'vita',
 'custom',
```

```
 'rfid',
 'aiso',
 'assist',
 'custom',
 'suppli',
 'chain',
 'track',
 'oi',
 'ga',
 'chemica',
 'product',
 'extract',
 'refin',
 'saie',
 'retai',
 'evel',
 'vinobl',
 'viewpoint',
 'previousiy',
 'state',
 'applic',
 'vaiuabl',
 'mine',
 'industri',
 'protect',
 'measur',
 'countri',
 'natura',
 'resourc',
 'commod',
 'threat',
 'preserv',
 'fuei',
 'resourc',
 'import',
 'safeti',
 'u',
 'industri',
 'economi',
 'compani',
 'believ',
 'offer',
 'servic',
 'technoiogi',
 'appiic',
 'oil',
 'ga',
 'petrochem',
 'industri',
```

```
 'wil',
 'posit',
 'vinobl',
 'rapidli',
 'expand',
 'industri',
 'whiie',
 'take',
 'advantag',
 'access',
 'increas',
 'capit',
 'gioba',
 'spend',
 'compani',
 'wi',
 'requir',
 'growth',
 'compani',
 'goal',
 'aiso',
 'provid',
 'much',
 'need',
 'servic',
 'cost',
 'manag',
 'even',
 'sma',
 'est',
 'busi',
 'afford',
 'without',
 'safeti',
 'personnel',
 'asset',
 'current',
 'state',
 'constant',
 'threat',
 'outstand',
 'news',
 'growth',
 'potenti',
 'compani',
 'except',
 'alreadi',
 'hot',
 'industri',
```

```
 'vnbl',
 'ob',
 'stand',
 'truiy',
 'innov',
 'pioneer',
 'see',
 'big',
 'thing',
 'happen',
 'stock',
 'inform',
 'within',
 'emai',
 'contain',
 'forward',
 'look',
 'statement',
 'within',
 'mean',
 'section',
 '27',
 'secur',
 'act',
 '1933',
 'section',
 '21',
 'b',
 'secur',
 'exchang',
 'act',
 '1934',
 'statement',
 'express',
 'involv',
 'discuss',
 'respect',
 'predict',
 'expect',
 'belief',
 'pian',
 'project',
 'object',
 'goal',
 'assumpt',
 'futur',
 'event',
 'perform',
 'statement',
```

```
 'historica',
 'fact',
 'may',
 'forward',
 'ook',
 'statement',
 'forward',
 'ook',
 'statement',
 'base',
 'expect',
 'estim',
 'project',
 'time',
 'statement',
 'made',
 'invoiv',
 'number',
 'risk',
 'uncertainti',
 'couid',
 'caus',
 'actua',
 'result',
 'event',
 'differ',
 'materia',
 'present',
 'anticip',
 'forward',
 'look',
 'statement',
 'action',
 'may',
 'identifi',
 'use',
 'word',
 'project',
 'forese',
 'expect',
 'wi',
 'anticip',
 'estim',
 'beiiev',
 'understand',
 'statement',
 'indic',
 'certain',
 'action',
```

```
 'may',
 'couid',
 'might',
 'occur',
 'mani',
 'micro',
 'cap',
 'stock',
 'today',
 'compani',
 'addit',
 'risk',
 'factor',
 'worth',
 'note',
 'factor',
 'inciud',
 'limit',
 'oper',
 'histori',
 'compani',
 'advanc',
 'cash',
 'reiat',
 'parti',
 'sharehold',
 'unsecur',
 'basi',
 'one',
 'vendor',
 'relat',
 'parti',
 'major',
 'stockhoid',
 'suppli',
 'nineti',
 'seven',
 'percent',
 'compani',
 'raw',
 'materiai',
 'reiianc',
 'two',
 'custom',
 'fifti',
 'percent',
 'busi',
 'numer',
 'relat',
```

```
 'parti',
 'transact',
 'need',
 'rais',
 'capit',
 'factor',
 'other',
 'fuili',
 'speil',
 'compani',
 'sec',
 'fii',
 'urg',
 'read',
 'file',
 'invest',
 'rocket',
 'stock',
 'report',
 'repres',
 'inform',
 'contain',
 'messag',
 'state',
 'ail',
 'materia',
 'fact',
 'omit',
 'materi',
 'fact',
 'necessari',
 'make',
 'statement',
 ...]
```

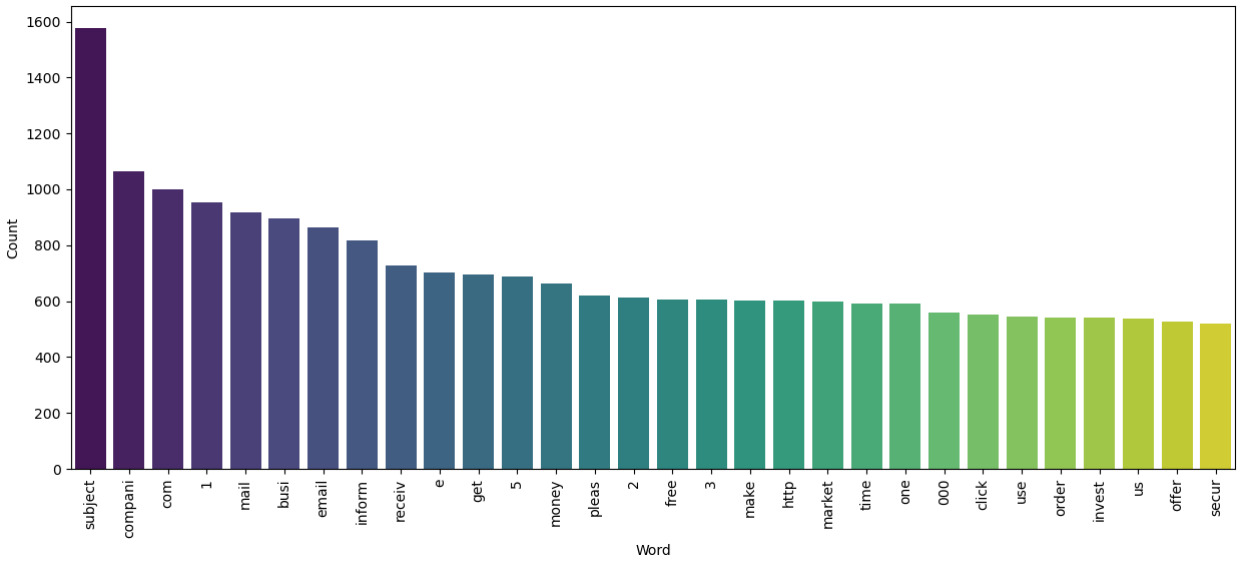
```
len(spam_corpus)
```

```
174630
```

```
# from collections import Counter
# sns.barplot(pd.DataFrame(Counter(spam_corpus).most_common(30))
[0],pd.DataFrame(Counter(spam_corpus).most_common(30))[1])
# plt.xticks(rotation='vertical')
from collections import Counter

# Convert Counter to DataFrame with named columns
df_spam = pd.DataFrame(Counter(spam_corpus).most_common(30),
columns=['Word', 'Count'])

# Create the barplot with different colors
```

```
plt.figure(figsize=(15, 6))
sns.barplot(x=df_spam['Word'], y=df_spam['Count'], palette="viridis")
# You can try "magma", "coolwarm", etc.
# Rotate x-axis labels for better readability
plt.xticks(rotation='vertical')
plt.show()
```

```
C:\Users\sevan\AppData\Local\Temp\ipykernel_2808\3513459698.py:11:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.barplot(x=df_spam['Word'], y=df_spam['Count'],
palette="viridis")  # You can try "magma", "coolwarm", etc.
```




```
ham_corpus = []
for msg in df[df['target']==0]['transformed_messages'].tolist():
    for word in msg.split():
        ham_corpus.append(word)

ham_corpus
```

```
['subject',
 'hello',
 'guy',
 'bug',
 'complet',
 'questionnair',
 'one',
 'page',
```

```
 'bio',
 'statement',
 'thought',
 'busi',
 'edu',
 'new',
 'economi',
 'record',
 'incorrect',
 'pleas',
 'ship',
 'respons',
 'want',
 'put',
 'everyth',
 'togeth',
 'next',
 'week',
 'ship',
 'back',
 'everyon',
 'questionnair',
 'attach',
 'well',
 'copi',
 'bio',
 'page',
 'michael',
 'froehl',
 'two',
 'somewhat',
 'differ',
 'approach',
 'idea',
 'latter',
 'introduc',
 'panelist',
 'give',
 'background',
 'approach',
 'issu',
 'discuss',
 'also',
 'provid',
 'copi',
 'attende',
 'use',
 'materi',
 'person',
```

```
 'introduct',
 'open',
 'panel',
 'discuss',
 'thank',
 'look',
 'forward',
 'see',
 'two',
 'week',
 'john',
 'waco',
 'background',
 'mf',
 'doc',
 'jmartinbiosketch',
 'doc',
 'questionnair',
 'doc',
 'john',
 'martin',
 'carr',
 'p',
 'collin',
 'chair',
 'financ',
 'financ',
 'depart',
 'baylor',
 'univers',
 'po',
 'box',
 '98004',
 'waco',
 'tx',
 '76798',
 '254',
 '710',
 '4473',
 'offic',
 '254',
 '710',
 '1092',
 'fax',
 'j',
 'martin',
 'baylor',
 'edu',
 'web',
```

```
 'http',
 'hsb',
 'baylor',
 'edu',
 'html',
 'martinj',
 'home',
 'html',
 'subject',
 'sacramento',
 'weather',
 'station',
 'fyi',
 'forward',
 'mike',
 'robert',
 'hou',
 'ect',
 '09',
 '20',
 '2000',
 '09',
 '06',
 'scott',
 'tholan',
 'enron',
 '09',
 '19',
 '2000',
 '07',
 '57',
 'pm',
 'mark',
 'tawney',
 'hou',
 'ect',
 'ect',
 'gari',
 'taylor',
 'hou',
 'ect',
 'ect',
 'mike',
 'robert',
 'hou',
 'ect',
 'ect',
 'cc',
 'chri',
```

```
 'clark',
 'na',
 'enron',
 'enron',
 'subject',
 'sacramento',
 'weather',
 'station',
 'hey',
 'guy',
 'talk',
 'contractor',
 'build',
 'us',
 'weather',
 'station',
 'hope',
 'quickli',
 'placement',
 'sacramento',
 'california',
 'varieti',
 'legal',
 'contractor',
 'oper',
 'reason',
 'need',
 'confirm',
 'follow',
 'requir',
 'soon',
 'possibl',
 'proceed',
 'need',
 'rainfal',
 'snowfal',
 'temperatur',
 'measur',
 'one',
 'high',
 'accuraci',
 'commerci',
 'avail',
 'weather',
 'station',
 'b',
 'need',
 'daili',
 'feed',
```

```
 'data',
 'enron',
 'weather',
 'desk',
 'mean',
 'one',
 'data',
 'dump',
 'set',
 'time',
 'per',
 'day',
 'altern',
 'need',
 'check',
 'data',
 'real',
 'time',
 'perhap',
 'vari',
 'multipl',
 'time',
 'day',
 'c',
 'instal',
 'station',
 'near',
 'sacramento',
 'california',
 'need',
 'know',
 'exactli',
 'area',
 'near',
 'sacramento',
 'suitabl',
 'site',
 'weather',
 'station',
 'name',
 'town',
 'mention',
 'mark',
 'interest',
 'time',
 'recommend',
 'weather',
 'expert',
 'accompani',
```

```
 'landman',
 'select',
 'site',
 'allow',
 'landman',
 'quickli',
 'leas',
 'instal',
 'station',
 'desir',
 'independ',
 'secur',
 'measur',
 'deter',
 'detect',
 'tamper',
 'suggest',
 'given',
 'short',
 'time',
 'fuse',
 'first',
 'instal',
 'station',
 'develop',
 'secur',
 'measur',
 'e',
 'feed',
 'data',
 'directli',
 'enron',
 'weather',
 'desk',
 'parti',
 'requir',
 'real',
 'time',
 'access',
 'data',
 'pleas',
 'forward',
 'respons',
 'directli',
 'chri',
 'clark',
 'na',
 'enron',
 'thank',
```

```
 'scott',
 'subject',
 'enron',
 'india',
 'newsdesk',
 'jan',
 '18',
 'th',
 'newsclip',
 'vinc',
 'fyi',
 'forward',
 'sandeep',
 'kohli',
 'enron',
 'develop',
 '01',
 '19',
 '2001',
 '05',
 '12',
 'pm',
 'enron',
 'india',
 'newsdesk',
 'jan',
 '18',
 'th',
 'newsclip',
 'unti',
 'editori',
 'thursday',
 'jan',
 '18',
 '2001',
 'http',
 'www',
 'economictim',
 'com',
 'today',
 '18',
 'edito',
 '2',
 'htm',
 'state',
 'give',
 'tax',
 'sop',
 'dpc',
```

```
 'buy',
 'naphtha',
 'ioc',
 'sanjay',
 'jog',
 'thursday',
 'jan',
 '18',
 'centr',
 'yet',
 'receiv',
 'propos',
 'enron',
 'thursday',
 'jan',
 '18',
 '2001',
 'http',
 'www',
 'economictim',
 'com',
 'today',
 '18',
 'infro',
 '2',
 'htm',
 'unti',
 'editori',
 'thursday',
 'jan',
 '18',
 '2001',
 'govern',
 'maharashtra',
 'want',
 'new',
 'delhi',
 'convinc',
 'power',
 'trade',
 'corpor',
 'central',
 'util',
 'buy',
 'power',
 'dabhol',
 'sell',
 'across',
 'countri',
```

```
 'would',
 'far',
 'simpler',
 'dabhol',
 'independ',
 'power',
 'produc',
 'allow',
 'sell',
 'power',
 'whoever',
 'will',
 'pay',
 'unfortun',
 'allow',
 'law',
 'forc',
 'privat',
 'sector',
 'gener',
 'sell',
 'power',
 'state',
 'util',
 'turn',
 'permit',
 'sell',
 'power',
 'across',
 'state',
 'state',
 'electr',
 'board',
 'bankrupt',
 'mseb',
 'reportedli',
 'owe',
 'central',
 'util',
 'rs',
 '5',
 '000',
 'crore',
 'bring',
 'charg',
 'mani',
 'type',
 'user',
 'power',
```

```
 'consum',
 'prevent',
 'larg',
 'scale',
 'theft',
 'electr',
 'includ',
 'wealthi',
 'maharashtra',
 'reform',
 'state',
 'electr',
 'board',
 'privatis',
 'transmiss',
 'distribut',
 'given',
 'profil',
 'buyer',
 'privat',
 'gener',
 'demand',
 'sovereign',
 'guarante',
 'help',
 'tide',
 'default',
 'risk',
 'guarante',
 'mere',
 'insul',
 'ipp',
 'risk',
 'make',
 'seb',
 'solvent',
 'yet',
 'india',
 'need',
 'power',
 'desper',
 'maharashtra',
 'india',
 'richest',
 'state',
 'experi',
 'power',
 'shortag',
 'around',
 '2',
```

```
'000',
'mw',
'sixth',
'peak',
'need',
'time',
'hunger',
'power',
'grow',
'india',
'afford',
'wait',
'pain',
'polit',
'seb',
'reform',
'work',
'govern',
'bring',
'legisl',
'allow',
'ipp',
'sell',
'power',
'directli',
'pay',
'custom',
'free',
'ipp',
'clutch',
'bankrupt',
'monopsoni',
'buyer',
'power',
'trade',
'legisl',
'unexpectedli',
'happi',
'consequ',
'govern',
'ipp',
'freed',
'oner',
'oblig',
'sell',
'power',
'singl',
'mostli',
'bankrupt',
```

```
 'buyer',
 'default',
 'risk',
 'come',
 'substanti',
 'new',
 'delhi',
 'state',
 'govern',
 'scrap',
 'guarante',
 'gave',
 'ipp',
 'past',
 'combin',
 'power',
 'trade',
 'privat',
 'invest',
 'gener',
 'transmiss',
 'distribut',
 'gradual',
 'seb',
 'reform',
 'creat',
 'commerci',
 'workabl',
 'competit',
 'power',
 'market',
 'india',
 'anyth',
 'less',
 'recip',
 'disast',
 'state',
 'give',
 'tax',
 'sop',
 'dpc',
 'buy',
 'naphtha',
 'ioc',
 'sanjay',
 'jog',
 'thursday',
 'jan',
 '18',
```

```
 '2001',
 'maharshtra',
 'govern',
 'financ',
 'depart',
 'strive',
 'reduc',
 'fiscal',
 'deficit',
 'rs',
 '9',
 '484',
 'crore',
 'rs',
 '3',
 '484',
 'crore',
 'begin',
 'april',
 'year',
 'express',
 'inabl',
 'provid',
 'sale',
 'tax',
 'waiver',
 'dabhol',
 'power',
 'compani',
 'dpc',
 'procur',
 '1',
 '2',
 'million',
 'tonn',
 'naphtha',
 'state',
 'run',
 'indian',
 'oil',
 'corpor',
 'ioc',
 'mantralaya',
 'sourc',
 'told',
 'financi',
 'express',
 'wednesday',
 'dpc',
```

```
 'would',
 'pay',
 '4',
 'per',
 'cent',
 'sale',
 'tax',
 'govern',
 'way',
 'back',
 '1995',
 'modifi',
 'sale',
 'tax',
 'rate',
 '4',
 'per',
 'cent',
 'discourag',
 'import',
 'naphtha',
 'gujarat',
 'electr',
 'compani',
 'oper',
 'maharashtra',
 'decis',
 'taken',
 'view',
 'encourag',
 'electr',
 'compani',
 'procur',
 'naphtha',
 'reduc',
 'rate',
 'within',
 'state',
 'govern',
 'sourc',
 'ad',
 'sourc',
 'said',
 'compani',
 'pay',
 'nearli',
 '15',
 '3',
 'per',
```

```
 'cent',
 'sale',
 'tax',
 'naphtha',
 'procur',
 'gujarat',
 'howev',
 'follow',
 'represent',
 'govern',
 'slash',
 'sale',
 'tax',
 'rate',
 '4',
 'per',
 'cent',
 'state',
 'financ',
 'depart',
 'opinion',
 'would',
 'present',
 'state',
 'cabinet',
 'shortli',
 'order',
 'take',
 'final',
 'decis',
 'deserv',
 'special',
 'signific',
 'especi',
 'state',
 'energi',
 'depart',
 'loss',
 'make',
 'maharashtra',
 'state',
 'electr',
 'board',
 'mseb',
 'wholeheartedli',
 'support',
 'dpc',
 'caus',
 'recommend',
```

```
 'sale',
 'tax',
 'waiver',
 'dpc',
 'ask',
 'union',
 'ministri',
 'oil',
 'petroleum',
 'procur',
 'naphtha',
 'within',
 'countri',
 'view',
 'excess',
 'avail',
 'present',
 'state',
 'govern',
 'mseb',
 'made',
 'clear',
 'would',
 'left',
 'altern',
 'pass',
 'addit',
 'burden',
 'mseb',
 'would',
 'ultim',
 'pass',
 'consum',
 'dpc',
 'also',
 'told',
 'state',
 'govern',
 'paid',
 'sale',
 'tax',
 'procur',
 'naphtha',
 'glencor',
 'calend',
 'year',
 '2000',
 'sourc',
 'state',
```

```
 'energi',
 'depart',
 'mseb',
 'stress',
 'need',
 'waiver',
 'express',
 'inabl',
 'bear',
 'addit',
 'burden',
 'suggest',
 'state',
 'reciproc',
 'offer',
 'sale',
 'tax',
 'exempt',
 'dpc',
 'ioc',
 'behest',
 'centr',
 'tri',
 'match',
 'intern',
 'land',
 'price',
 'naphtha',
 'recent',
 'sign',
 'memorandum',
 'agreement',
 'dpc',
 'state',
 'financ',
 'depart',
 'stick',
 'view',
 'may',
 'hurt',
 'state',
 'whole',
 'sourc',
 'state',
 'energi',
 'depart',
 'mseb',
 'said',
 'dpc',
```

```
 'procur',
 'naphtha',
 'rs',
 '11',
 '050',
 'per',
 'ton',
 'ioc',
 'calend',
 'year',
 '2001',
 'compar',
 'rs',
 '10',
 '050',
 'per',
 'tonn',
 'price',
 'quot',
 'glencor',
 'naphtha',
 'price',
 'compris',
 '175',
 'per',
 'tonn',
 'free',
 'board',
 'fob',
 '21',
 '8',
 'per',
 'cent',
 'custom',
 'duti',
 '5',
 '4',
 'per',
 'cent',
 'sale',
 'tax',
 '18',
 '87',
 'premium',
 'dpc',
 'senior',
 'vice',
 'presid',
 'mukesh',
```

```
 'tyagi',
 'reiter',
 'compani',
 'alreadi',
 'made',
 'appeal',
 'state',
 'govern',
 'sale',
 'tax',
 'waiver',
 'naphtha',
 'larger',
 'interest',
 'consum',
 'sale',
 'tax',
 'pass',
 'mseb',
 'bear',
 'addit',
 'burden',
 'pass',
 'consum',
 'ad',
 'centr',
 'yet',
 'receiv',
 'propos',
 'enron',
 'thursday',
 'jan',
 '18',
 '2001',
 'centr',
 'wednesday',
 'said',
 'recevi',
 'propos',
 'maharashtra',
 'govern',
 'seek',
 'help',
 'solv',
 'tangl',
 'enron',
 'promot',
 'dhabol',
 'power',
```

```
 'project',
 'relat',
 'cost',
 'surplu',
 'power',
 'ask',
 'report',
 'maharashtra',
 'govern',
 'send',
 'propos',
 'centr',
 'buy',
 'surplu',
 'power',
 'dhabol',
 'power',
 'compani',
 'power',
 'trade',
 'corpor',
 'power',
 'minist',
 'suresh',
 'prabhu',
 'said',
 'receiv',
 'propos',
 'care',
 'watch',
 'situat',
 'await',
 'concret',
 'propos',
 'interven',
 'matter',
 'parbhu',
 'said',
 'sidelin',
 'greentech',
 'environ',
 'excel',
 'award',
 'ceremoni',
 'ask',
 'whether',
 'possibl',
 'govern',
 'ask',
```

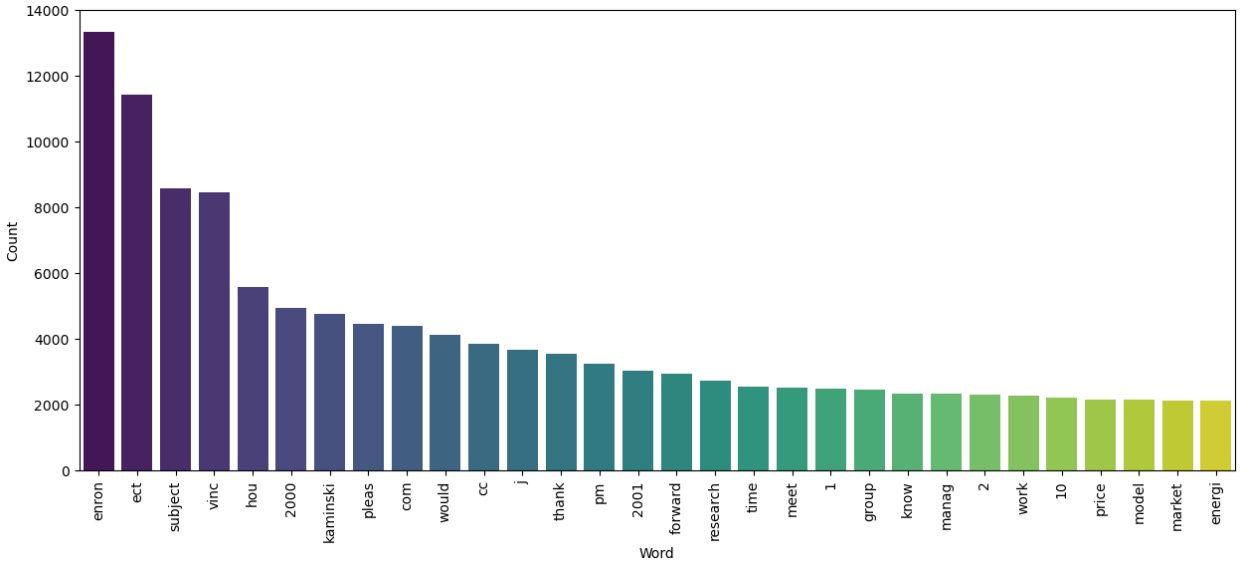
```
 'power',
 'trade',
 'corpor',
 'buy',
 'power',
 'dhabol',
 'power',
 'corpor',
 'prabhu',
 'repli',
 'ptc',
 ...]
```

```
len(ham_corpus)
```

```
702991
```

```
from collections import Counter

# Convert Counter to DataFrame with named columns
df_ham = pd.DataFrame(Counter(ham_corpus).most_common(30),
columns=['Word', 'Count'])

# Create the barplot with different colors
plt.figure(figsize=(15, 6))
sns.barplot(x=df_ham['Word'], y=df_ham['Count'], palette="viridis")  #
You can try "magma", "coolwarm", etc.
# Rotate x-axis labels for better readability
plt.xticks(rotation='vertical')
plt.show()
```

```
C:\Users\sevan\AppData\Local\Temp\ipykernel_2808\1297375888.py:8:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.barplot(x=df_ham['Word'], y=df_ham['Count'], palette="viridis")
# You can try "magma", "coolwarm", etc.
```



4. Model Buiding

```
# Text Vectorization
# 1. Using Bag of Words
df.head()
```

```
                                                messages  target
num_characters  \
0  Subject: naturally irresistible your corporate...       1
1484
1  Subject: the stock trading gunslinger  fanny i...       1
598
2  Subject: unbelievable new homes made easy  im ...       1
448
3  Subject: 4 color printing special  request add...       1
500
4  Subject: do not have money , get software cds ...       1
235

   num_words  num_sentences
transformed_messages
0        325             11  subject natur irresist corpor ident lt
realli ...
1         90              1  subject stock trade gunsling fanni merril
muzo...
2         88              4  subject unbeliev new home made easi im
want sh...
3         99              5  subject 4 color print special request
addit in...
4         53              9  subject money get softwar cd softwar
compat gr...
```

```
from sklearn.feature_extraction.text import
CountVectorizer,TfidfVectorizer
cv = CountVectorizer()
tfidf = TfidfVectorizer(max_features=3000)

# X = cv.fit_transform(df['transformed_messages']).toarray()
X = tfidf.fit_transform(df['transformed_messages']).toarray()

from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X = scaler.fit_transform(X)

# # appending the num_character col to X
# X = np.hstack((X,df['num_characters'].values.reshape(-1,1)))

X.shape
```

```
(5695, 3000)
```

```
y = df['target'].values

y
```

```
array([1, 1, 1, ..., 0, 0, 0], dtype=int64)
```

```
from sklearn.model_selection import train_test_split

X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.2,random_state=2)

from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
from sklearn.metrics import
accuracy_score,confusion_matrix,precision_score

gnb = GaussianNB()
mnb = MultinomialNB()
bnb = BernoulliNB()

gnb.fit(X_train,y_train)
y_pred1 = gnb.predict(X_test)
print(accuracy_score(y_test,y_pred1))
print(confusion_matrix(y_test,y_pred1))
print (precision_score(y_test,y_pred1))
```

```
0.9525899912203687
[[837  12]
 [ 42 248]]
0.9538461538461539
```

```
mnb.fit(X_train,y_train)
y_pred2 = mnb.predict(X_test)
print(accuracy_score(y_test,y_pred2))
```

```
print(confusion_matrix(y_test,y_pred2))
print (precision_score(y_test,y_pred2))
```

```
0.9877085162423178
[[840   9]
 [  5 285]]
0.9693877551020408
```

```
bnb.fit(X_train,y_train)
y_pred3 = mnb.predict(X_test)
print(accuracy_score(y_test,y_pred3))
print(confusion_matrix(y_test,y_pred3))
print (precision_score(y_test,y_pred3))
```

```
0.9877085162423178
[[840   9]
 [  5 285]]
0.9693877551020408
```

```
# tfidf-->MNB
```

```
!pip install xgboost
```

```
Requirement already satisfied: xgboost in c:\users\sevan\anaconda3\
lib\site-packages (3.0.0)
Requirement already satisfied: numpy in c:\users\sevan\anaconda3\lib\
site-packages (from xgboost) (1.26.4)
Requirement already satisfied: scipy in c:\users\sevan\anaconda3\lib\
site-packages (from xgboost) (1.13.1)
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
```

```
svc = SVC(kernel = 'sigmoid', gamma = 1.0)
knc = KNeighborsClassifier()
mnb = MultinomialNB()
dtc = DecisionTreeClassifier(max_depth=5)
lrc =LogisticRegression(solver = 'liblinear',penalty='l1')
rfc = RandomForestClassifier(n_estimators=50, random_state=2)
abc = AdaBoostClassifier(n_estimators=50, random_state=2)
bc = BaggingClassifier(n_estimators=50, random_state=2)
etc = ExtraTreesClassifier(n_estimators=50, random_state=2)
```

```
gbdt = GradientBoostingClassifier(n_estimators=50, random_state=2)
xgb = XGBClassifier(n_estimators=50, random_state=2)

clfs = {
    'SVC' : svc,
    'KN' : knc,
    'NB' : mnb,
    'DT' : dtc,
    'LR' : lrc,
    'RF' : rfc,
    'AdaBoost' : abc,
    'BgC' : bc,
    'ETC' : etc,
    'GBDT' : gbdt,
    'xgb' : xgb
}

def train_classifier(clf,X_train,y_train,X_test,y_test):
    clf.fit(X_train,y_train)
    y_pred = clf.predict(X_test)
    accuracy = accuracy_score(y_test,y_pred)
    precision = precision_score(y_test,y_pred)

    return accuracy, precision

train_classifier(svc,X_train,y_train,X_test,y_test)
```

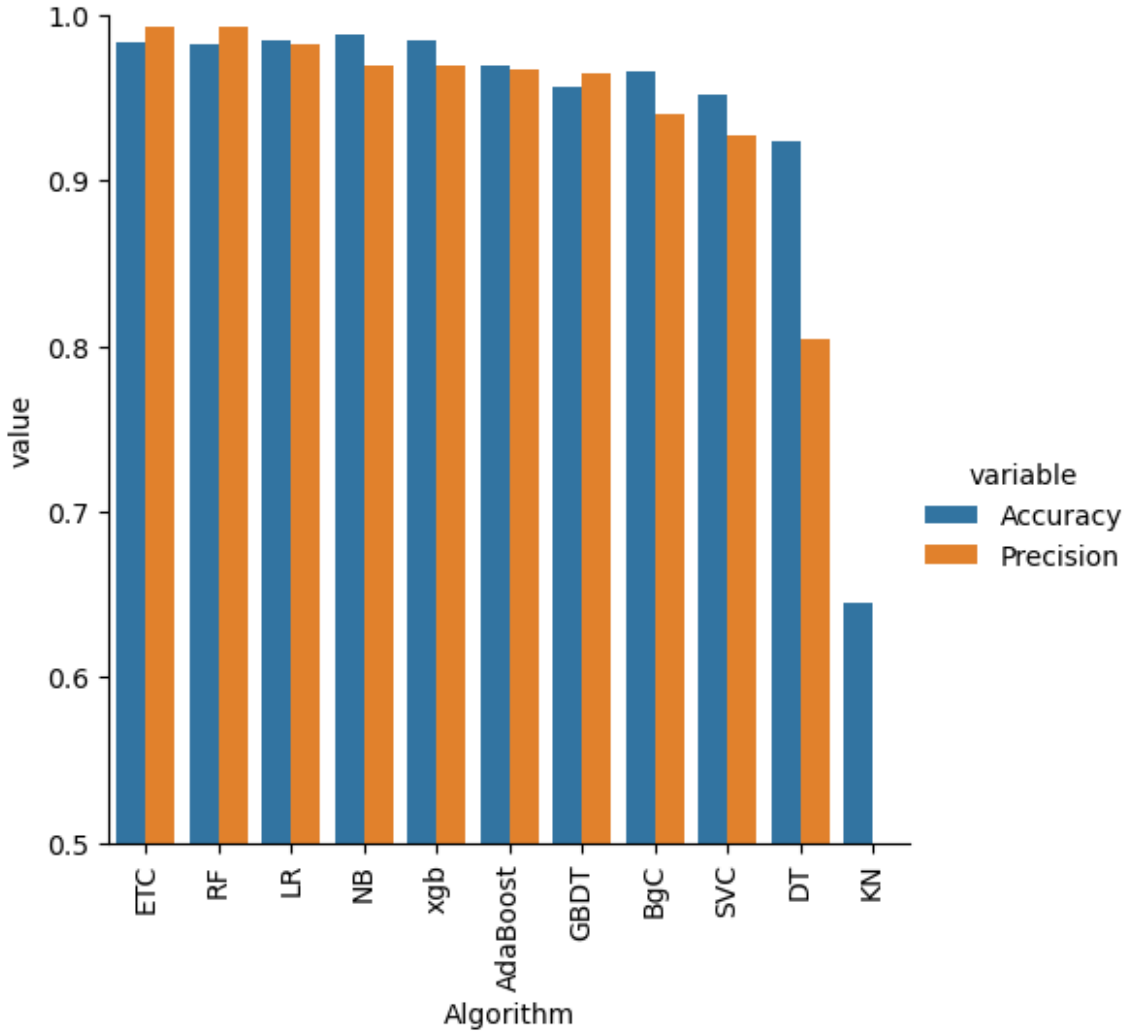
```
(0.95171202809482, 0.9272727272727272)
```

```
accuracy_scores = []
precision_scores = []

for name,clf in clfs.items():
    current_accuracy,current_precision =
train_classifier(clf,X_train,y_train,X_test,y_test)
    print("For ",name)
    print("Accuracy - ",current_accuracy)
    print("Precision - ",current_precision)
    accuracy_scores.append(current_accuracy)
    precision_scores.append(current_precision)
```

```
For  SVC
Accuracy -  0.95171202809482
Precision -  0.9272727272727272
For  KN
Accuracy -  0.6453028972783144
Precision -  0.41786743515850144
For  NB
Accuracy -  0.9877085162423178
Precision -  0.9693877551020408
For  DT
```

```
Accuracy -  0.9236172080772608
Precision -  0.8048048048048048
For  LR
Accuracy -  0.9841966637401229
Precision -  0.9822695035460993
For  RF
Accuracy -  0.9824407374890255
Precision -  0.9927007299270073

C:\Users\sevan\anaconda3\Lib\site-packages\sklearn\ensemble\
_weight_boosting.py:527: FutureWarning: The SAMME.R algorithm (the
default) is deprecated and will be removed in 1.6. Use the SAMME
algorithm to circumvent this warning.
  warnings.warn(

For  AdaBoost
Accuracy -  0.9692712906057945
Precision -  0.967032967032967
For  BgC
Accuracy -  0.9657594381035997
Precision -  0.9403508771929825
For  ETC
Accuracy -  0.9833187006145742
Precision -  0.9927272727272727
For  GBDT
Accuracy -  0.9561018437225637
Precision -  0.9651162790697675
For  xgb
Accuracy -  0.9850746268656716
Precision -  0.9690721649484536

performance_df = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy':
accuracy_scores,
'Precision':precision_scores}).sort_values('Precision',ascending=False
)

performance_df

   Algorithm  Accuracy  Precision
8        ETC  0.983319   0.992727
5         RF  0.982441   0.992701
4         LR  0.984197   0.982270
2         NB  0.987709   0.969388
10       xgb  0.985075   0.969072
6   AdaBoost  0.969271   0.967033
9       GBDT  0.956102   0.965116
7         BgC  0.965759   0.940351
0         SVC  0.951712   0.927273
3         DT  0.923617   0.804805
1         KN  0.645303   0.417867
```

```
performance_df1 = pd.melt(performance_df,id_vars = "Algorithm")

sns.catplot(x = 'Algorithm', y='value', hue = 'variable',
data=performance_df1,kind='bar',height=5)
plt.ylim(0.5,1.0)
plt.xticks(rotation = 'vertical')
plt.show()
```




```
#model improve
# 1. Change the max_features parameter of TFIDf

temp_df = pd.DataFrame({'Algorithm':clfs.keys(),
'Accuracy_max_ft_3000':accuracy_scores,
'Precision_max_ft_3000':precision_scores})

new_df = performance_df.merge(temp_df,on='Algorithm')

new_df
```

```
   Algorithm  Accuracy  Precision  Accuracy_max_ft_3000
Precision_max_ft_3000
0        ETC  0.983319   0.992727              0.983319
0.992727
1         RF  0.982441   0.992701              0.982441
0.992701
2         LR  0.984197   0.982270              0.984197
0.982270
3         NB  0.987709   0.969388              0.987709
0.969388
4        xgb  0.985075   0.969072              0.985075
0.969072
5   AdaBoost  0.969271   0.967033              0.969271
0.967033
6       GBDT  0.956102   0.965116              0.956102
0.965116
7        BgC  0.965759   0.940351              0.965759
0.940351
8        SVC  0.951712   0.927273              0.951712
0.927273
9         DT  0.923617   0.804805              0.923617
0.804805
10        KN  0.645303   0.417867              0.645303
0.417867
```

```
temp_df = pd.DataFrame({'Algorithm':clfs.keys(),
'Accuracy_scalling':accuracy_scores,'Precision_scalling':precision_sco
res})

# temp_df = pd.DataFrame({'Algorithm':clfs.keys(),
'Accurary_num_chars':accuracy_scores,'Precision_num_chars':precision_s
cores}).sort_values('Precision_num_chars',ascending=False)

new_df_scaled = new_df.merge(temp_df,on='Algorithm')

new_df_scaled
```

```
   Algorithm  Accuracy  Precision  Accuracy_max_ft_3000  \
0        ETC  0.983319   0.992727              0.983319
1         RF  0.982441   0.992701              0.982441
2         LR  0.984197   0.982270              0.984197
3         NB  0.987709   0.969388              0.987709
4        xgb  0.985075   0.969072              0.985075
5   AdaBoost  0.969271   0.967033              0.969271
6       GBDT  0.956102   0.965116              0.956102
7        BgC  0.965759   0.940351              0.965759
8        SVC  0.951712   0.927273              0.951712
9         DT  0.923617   0.804805              0.923617
10        KN  0.645303   0.417867              0.645303
```

```
    Precision_max_ft_3000  Accuracy_scalling  Precision_scalling
0                0.992727           0.983319            0.992727
1                0.992701           0.982441            0.992701
2                0.982270           0.984197            0.982270
3                0.969388           0.987709            0.969388
4                0.969072           0.985075            0.969072
5                0.967033           0.969271            0.967033
6                0.965116           0.956102            0.965116
7                0.940351           0.965759            0.940351
8                0.927273           0.951712            0.927273
9                0.804805           0.923617            0.804805
10               0.417867           0.645303            0.417867
```

```
#Voting Classifier
# svc = SVC(kernel='sigmoid', gamma=1.0,probability=True)
etc =ExtraTreesClassifier(n_estimators=50,random_state=2)
rf = RandomForestClassifier()
lr = LogisticRegression()
mnb = MultinomialNB()
from sklearn.ensemble import VotingClassifier

# voting = VotingClassifier(estimators=[('svm',svc),('nb',mnb),
('et',etc)],voting='soft')
voting = VotingClassifier(estimators=[('nb',mnb),('et',etc),('RF',rf),
('LR',lr)],voting='soft')

voting.fit(X_train,y_train)
```

```
VotingClassifier(estimators=[('nb', MultinomialNB()),
                             ('et',
                              ExtraTreesClassifier(n_estimators=50,
                                                   random_state=2)),
                             ('RF', RandomForestClassifier()),
                             ('LR', LogisticRegression())],
                 voting='soft')
```

```
y_pred = voting.predict(X_test)
print("Accuracy",accuracy_score(y_test,y_pred))
print("Precision",precision_score(y_test,y_pred))
```

```
Accuracy 0.9912203687445127
Precision 0.9929577464788732
```

```
#Apply stacking
estimators =[('LR',lr),('nb',mnb),('ETC',etc)]
final_estimator=RandomForestClassifier()

from sklearn.ensemble import StackingClassifier

clf=StackingClassifier(estimators=estimators,final_estimator=final_est
imator)
```

```
clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)
print("Accuracy",accuracy_score(y_test,y_pred))
print("Precision",precision_score(y_test,y_pred))
```

```
Accuracy 0.9920983318700615
Precision 0.9929824561403509
```

```
import pickle
pickle.dump(tfidf,open('vectorizer.pkl','wb'))
pickle.dump(mnb,open('model.pkl','wb'))
```