In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import nltk
```

In [2]:

```
#if Utf-8 encoding error comes then use encoding='ISO-8859-1' and then warning appears w
df=pd.read_csv(r'spam.csv',encoding='ISO-8859-1')
df.sample(5)
```

Out[2]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
2716	ham	House-Maid is the murderer, coz the man was mu...	NaN	NaN	NaN
3354	ham	Minimum walk is 3miles a day.	NaN	NaN	NaN
2858	ham	Do you know why god created gap between your f...	NaN	NaN	NaN
2993	ham	No idea, I guess we'll work that out an hour a...	NaN	NaN	NaN
1121	spam	Do you want 750 anytime any network mins 150 t...	NaN	NaN	NaN

In [3]:

```
df.shape
```

Out[3]:

```
(5572, 5)
```

In [4]:

```
#data cleaning
#EDA
#Text preprocessing
#Model building
#Evaluation
#improvements
```

In [5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

In [6]:

```
df.isnull().sum()
```

Out[6]:

```
v1               0
v2               0
Unnamed: 2    5522
Unnamed: 3    5560
Unnamed: 4    5566
dtype: int64
```

In [7]:

```
df.isnull().value_counts()
```

Out[7]:

```
v1     v2     Unnamed: 2  Unnamed: 3  Unnamed: 4
False  False  True        True        True          5522
              False       True        True            38
                          False       False            6
                                      True             6
dtype: int64
```

In [8]:

```
#inplace=True changes will reflect to actual data
df.drop(['Unnamed: 2','Unnamed: 3','Unnamed: 4'],axis=1,inplace=True)
```

In [9]:

```
df.head()
```

Out[9]:

	v1	v2
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

In [10]:

```
df.rename({'v1':'main','v2':'text'},axis=1,inplace=True)

#df.rename(columns={'v1':'main','v2':'text'},inplace=True)
```

In [11]:

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

Out[11]:

```
ham     4825
spam     747
Name: main, dtype: int64
```

In [12]:

```
df.head()
```

Out[12]:

	main	text
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

In [13]:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df['main']=le.fit_transform(df['main'])
#ham=0
#spam=1
```

In [14]:

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

Out[14]:

```
403
```

In [15]:

```
#keep='first means keep first occurence value and remove its corresponding duplicate val
df=df.drop_duplicates(keep='first')
```

In [16]:

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

Out[16]:

	main	text
86	0	For real when u getting on yo? I only need 2 m...
63	0	Sorry my roommates took forever, it ok if I co...
4405	1	As one of our registered subscribers u can ent...
3185	0	Happy birthday to you....dear.with lots of lov...
4273	0	Ball is moving a lot.will spin in last :)so ve...

In [17]:

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

Out[17]:

```
0
```

In [18]:

```
df.shape
```

Out[18]:

```
(5169, 2)
```

In [19]:

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

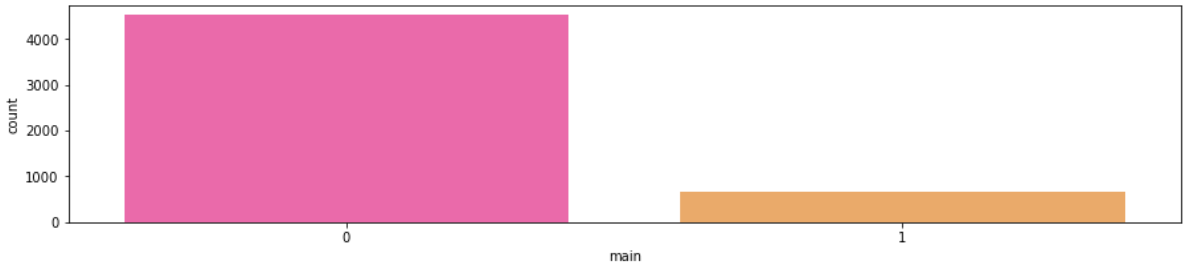
Out[19]:

```
0    4516
1     653
Name: main, dtype: int64
```

In [20]:

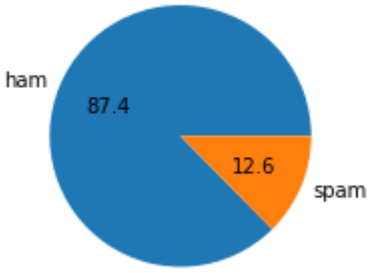
```
plt.rcParams['figure.figsize']=  (15,3)
sns.countplot(df['main'],palette='spring')
plt.show()
```

```
C:\Users\lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Fut
ureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```



In [21]:

```
#autopct='%0.1f' for getting % distribution of ham and spam 0.1f means needs only one d
plt.pie(df['main'].value_counts(),labels=['ham','spam'] ,autopct='%0.1f')
plt.show()
```



In [22]:

```
#NO. of character
#adding extra column to our dataframe which counts the number of character in the senter
df['num_char']=df['text'].apply(len)
df.head()
```

Out[22]:

	main	text	num_char
0	0	Go until jurong point, crazy.. Available only ...	111
1	0	Ok lar... Joking wif u oni...	29
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155
3	0	U dun say so early hor... U c already then say...	49
4	0	Nah I don't think he goes to usf, he lives aro...	61

In [23]:

```
from nltk.tokenize import word_tokenize
```

In [24]:

```
#no. of words
#make tokens of the sentence
df['text'].apply(lambda c: nltk.word_tokenize(c))
```

Out[24]:

```
0       [Go, until, jurong, point, ,, crazy, .., Avail...
1                 [Ok, lar, ..., Joking, wif, u, oni, ...]
2       [Free, entry, in, 2, a, wkly, comp, to, win, F...
3       [U, dun, say, so, early, hor, ..., U, c, alrea...
4       [Nah, I, do, n't, think, he, goes, to, usf, ,,...
                              ...
5567    [This, is, the, 2nd, time, we, have, tried, 2,...
5568     [Will, Ì_, b, going, to, esplanade, fr, home, ?]
5569    [Pity, ,, *, was, in, mood, for, that, ., So, ...
5570    [The, guy, did, some, bitching, but, I, acted,...
5571                  [Rofl, ., Its, true, to, its, name]
Name: text, Length: 5169, dtype: object
```

In [25]:

```
#word length and stored in new column
df['num_word']=df['text'].apply(lambda c: len(nltk.word_tokenize(c)))
```

In [26]:

```
df.head()
```

Out[26]:

	main	text	num_char	num_word
0	0	Go until jurong point, crazy.. Available only ...	111	24
1	0	Ok lar... Joking wif u oni...	29	8
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37
3	0	U dun say so early hor... U c already then say...	49	13
4	0	Nah I don't think he goes to usf, he lives aro...	61	15

In [27]:

```
#sentence count using sent_tokenize
df['num_sentence']=df['text'].apply(lambda c: len(nltk.sent_tokenize(c)))
```

In [28]:

```
df.head()
```

Out[28]:

	main	text	num_char	num_word	num_sentence
0	0	Go until jurong point, crazy.. Available only ...	111	24	2
1	0	Ok lar... Joking wif u oni...	29	8	2
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2
3	0	U dun say so early hor... U c already then say...	49	13	1
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1

In [29]:

```
df[['num_word','num_char','num_sentence']].describe()
```

Out[29]:

	num_word	num_char	num_sentence
count	5169.000000	5169.000000	5169.000000
mean	18.455407	78.977945	1.961308
std	13.322448	58.236293	1.432583
min	1.000000	2.000000	1.000000
25%	9.000000	36.000000	1.000000
50%	15.000000	60.000000	1.000000
75%	26.000000	117.000000	2.000000
max	220.000000	910.000000	38.000000

In [30]:

```
#max 220 words are used 910 char use (alphabets) and 38 sentences
```

In [31]:

```
#ham
df[df['main']==0][['num_word','num_char','num_sentence']].describe()
#mean of ham
# mean  17.123339   70.459256   1.815545
```

Out[31]:

	num_word	num_char	num_sentence
count	4516.000000	4516.000000	4516.000000
mean	17.123339	70.459256	1.815545
std	13.491315	56.358207	1.364098
min	1.000000	2.000000	1.000000
25%	8.000000	34.000000	1.000000
50%	13.000000	52.000000	1.000000
75%	22.000000	90.000000	2.000000
max	220.000000	910.000000	38.000000

In [32]:

```
#spam
df[df['main']==1][['num_word','num_char','num_sentence']].describe()
# mean  27.667688   137.891271  2.969372
```

Out[32]:

	num_word	num_char	num_sentence
count	653.000000	653.000000	653.000000
mean	27.667688	137.891271	2.969372
std	7.008418	30.137753	1.488910
min	2.000000	13.000000	1.000000
25%	25.000000	132.000000	2.000000
50%	29.000000	149.000000	3.000000
75%	32.000000	157.000000	4.000000
max	46.000000	224.000000	9.000000

In [33]:

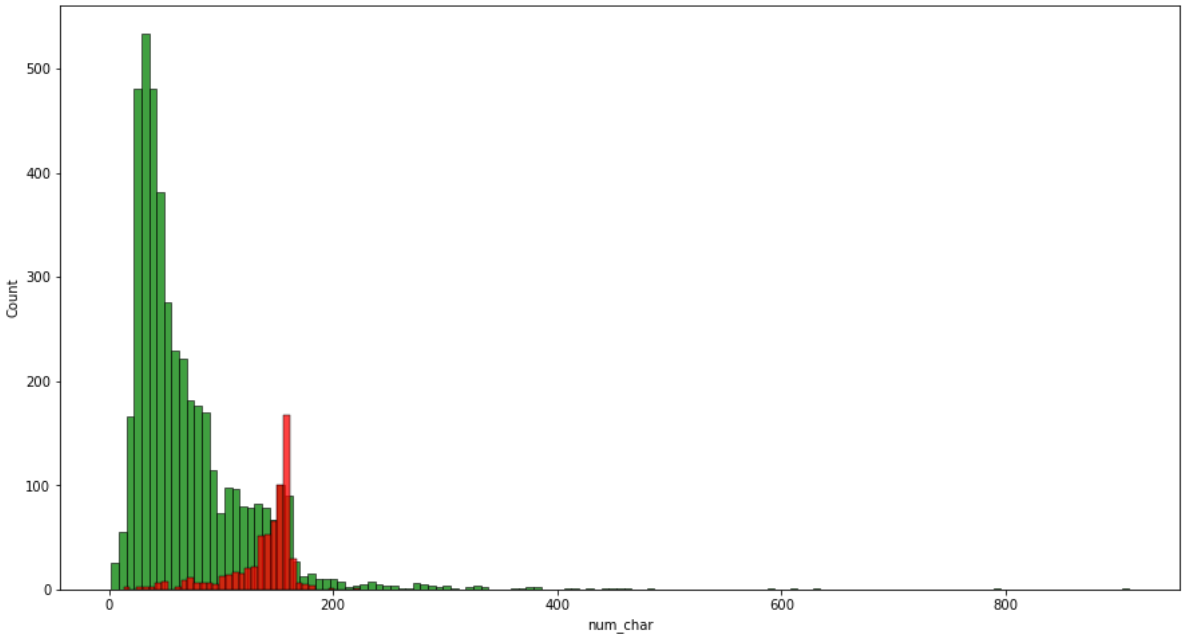
```
#Length of spam > ham as observed from it mean value
```

In [34]:

```
plt.rcParams['figure.figsize']=(15,8)
sns.histplot(df[df['main']==0]['num_char'],color='green')
sns.histplot(df[df['main']==1]['num_char'],color='red')
```

Out[34]:

```
<AxesSubplot:xlabel='num_char', ylabel='Count'>
```

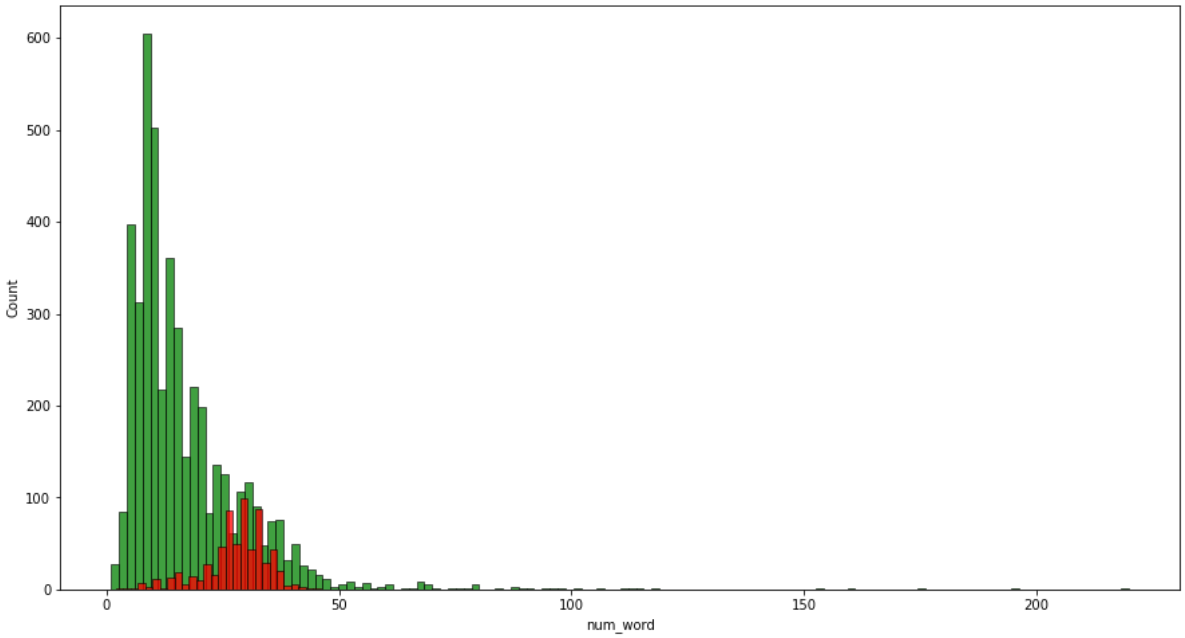


In [35]:

```
plt.rcParams['figure.figsize']=(15,8)
sns.histplot(df[df['main']==0]['num_word'],color='green')
sns.histplot(df[df['main']==1]['num_word'],color='red')
```

Out[35]:

```
<AxesSubplot:xlabel='num_word', ylabel='Count'>
```

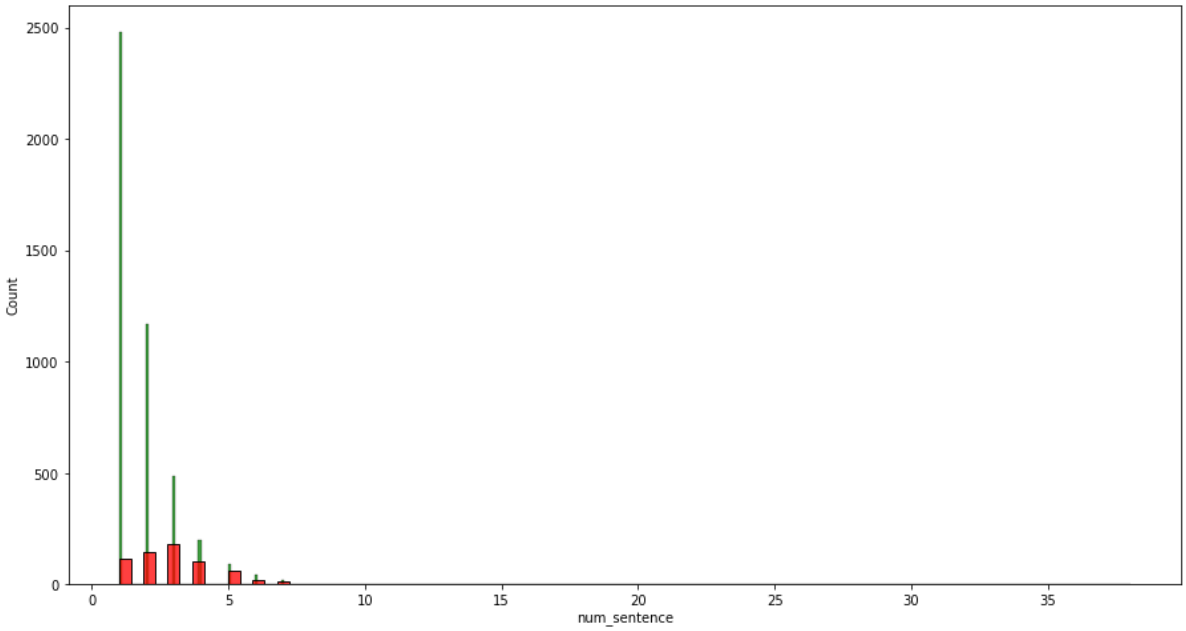


In [36]:

```
plt.rcParams['figure.figsize']=(15,8)
sns.histplot(df[df['main']==0]['num_sentence'],color='green')
sns.histplot(df[df['main']==1]['num_sentence'],color='red')
```

Out[36]:

```
<AxesSubplot:xlabel='num_sentence', ylabel='Count'>
```

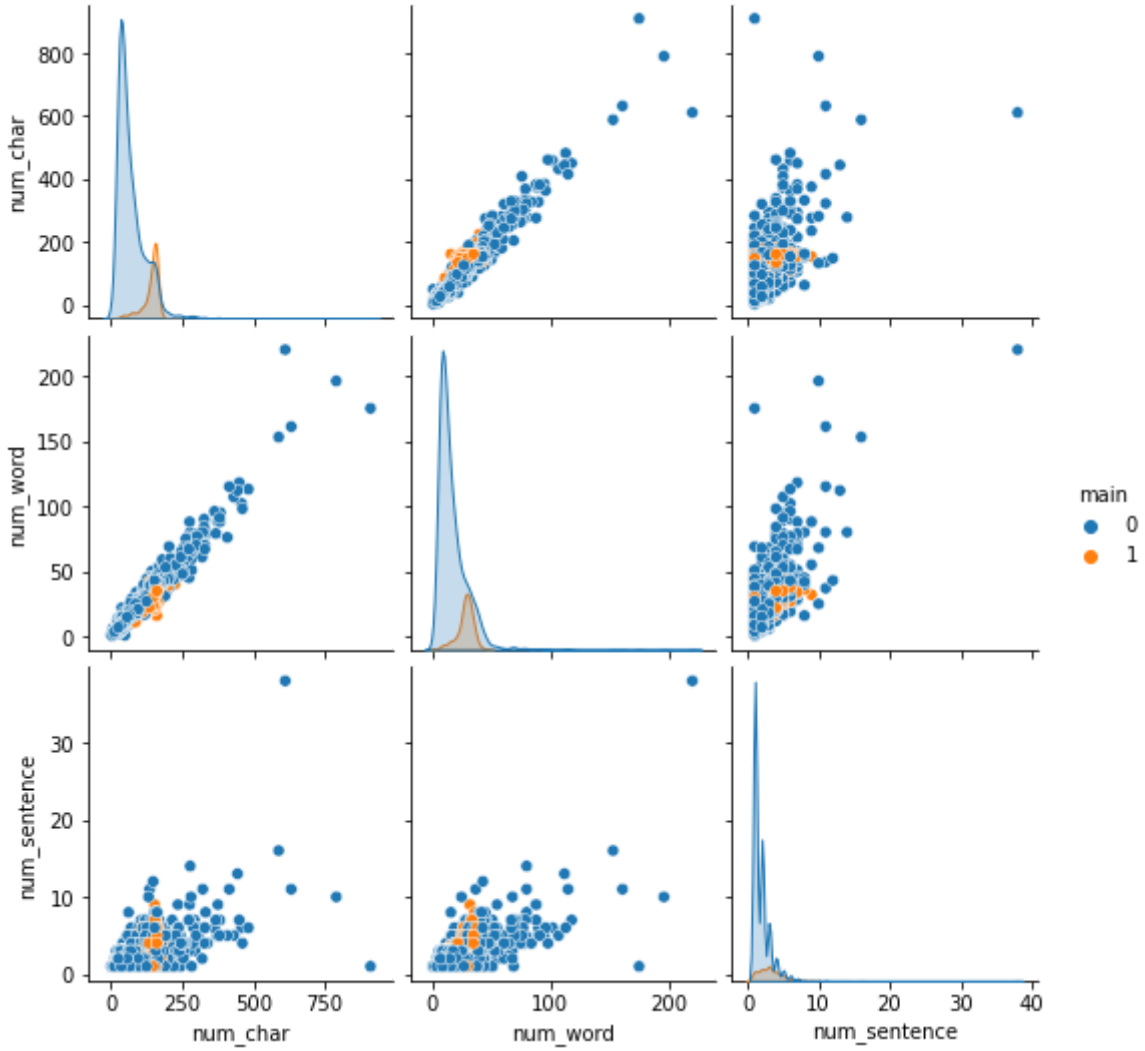


In [37]:

```
sns.pairplot(df,hue='main')
#outliers present in the data
```

Out[37]:

```
<seaborn.axisgrid.PairGrid at 0x1f479b62070>
```



In [38]:

```
df.corr()
```

Out[38]:

	main	num_char	num_word	num_sentence
main	1.000000	0.384717	0.262969	0.267602
num_char	0.384717	1.000000	0.965784	0.626118
num_word	0.262969	0.965784	1.000000	0.680882
num_sentence	0.267602	0.626118	0.680882	1.000000

In [39]:

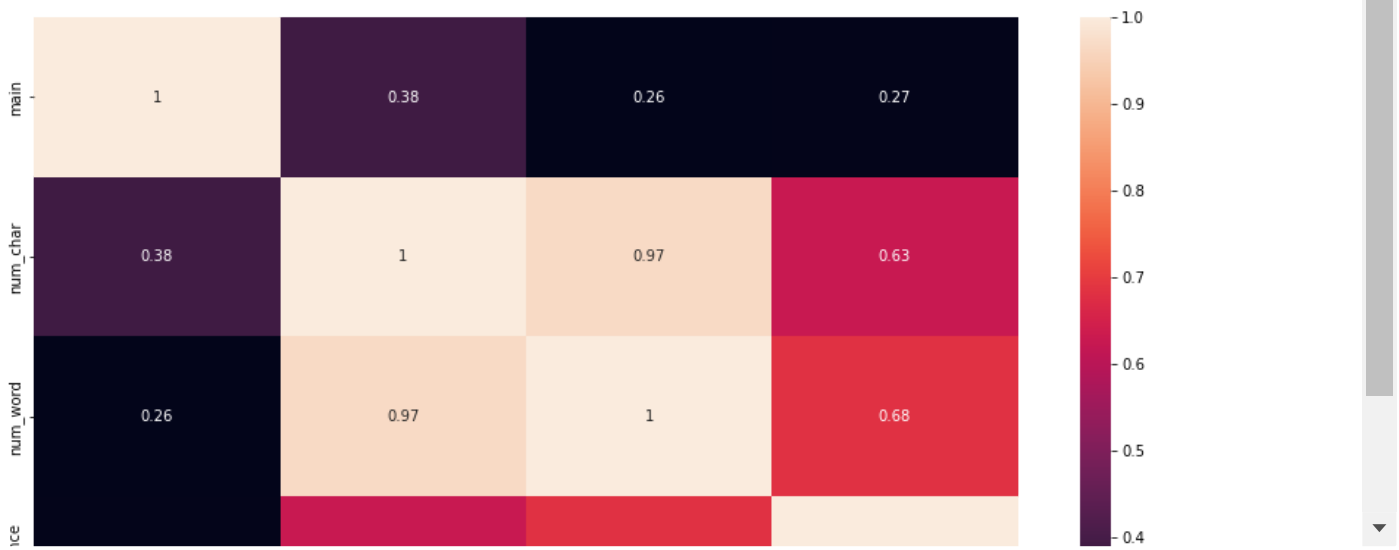
```
#annot =True can show the correlation value in each cell

plt.rcParams['figure.figsize']=(15,8)
sns.heatmap(df.corr(),annot=True)

#  main and num_char has strong corr=0.38
```

Out[39]:

```
<AxesSubplot:>
```



In [ ]:

Lower

Tokens

Removing special characters

Removing stop words and puctuation

Stemming (convert to root words)

In [40]:

```
#for punctuation
import string

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

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

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

In [41]:

```
def call(lem):

    #convert into lower
    lem=lem.lower()
    #convert into tokens
    lem=nltk.word_tokenize(lem)

    #make a list y
    #this function append alphanumeric value from the text or list y
    y=[]
    for i in lem:
        if i.isalnum():
            y.append(i)

    #list cloning
    lem=y[:]
    #
    y.clear()

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

    lem=y[:]
    y.clear()

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

    #in last we are joining the string " " means have gap between tokens
    return " ".join(y)
```

In [42]:

```
#in stemming e in last of words sometimes drop

lem=('''Subscribe my Youtube channel ok@ broTher.!, and let's dance together in the ever

call(lem)
```

Out[42]:

```
'subscrib youtub channel ok brother let danc togeth even parti'
```

In [43]:

```
#cloning list example

def clone(lst):
    lst_copy=lst[:]
    return lst

lst=('1,2,3,4,5')
lst_copy=clone(lst)
print(lst_copy)
```

```
1,2,3,4,5
```

In [44]:

```
#applying our call function
df['transformed_text']=df['text'].apply(call)
```

In [45]:

```
df.head()
```

Out[45]:

	main	text	num_char	num_word	num_sentence	transformed_text
0	0	Go until jurong point, crazy.. Available only ...	111	24	2	go jurong point crazi avail bugi n great world...
1	0	Ok lar... Joking wif u oni...	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	37	2	free entri 2 wkli comp win fa cup final tkt 21...
3	0	U dun say so early hor... U c already then say...	49	13	1	u dun say earli hor u c alreadi say
4	0	Nah I don't think he goes to usf, he lives aro...	61	15	1	nah think goe usf live around though

In [46]:

```
#pip install wordcloud
#        or
#conda install -c conda-forge wordcloud


#What is wordcloud in NLP?

#Image result for wordcloud python
#It is a visualization technique for text data wherein each word is picturized
#with its importance in the context or its frequency
```

In [47]:

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

In [48]:

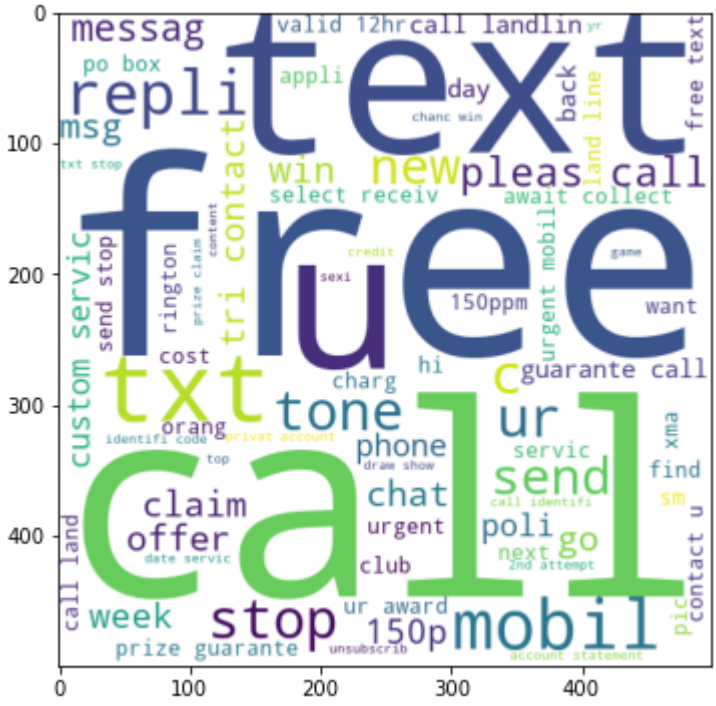
```
#for spam concatenating string in a single image
spam_wc=wc.generate(df[df['main']==1]['transformed_text'].str.cat(sep=" "))
```

In [49]:

```
plt.figure(figsize=(8,6))
plt.imshow(spam_wc)
```

Out[49]:

```
<matplotlib.image.AxesImage at 0x1f47c45fa00>
```



In [50]:

```
ham_wc=wc.generate(df[df['main']==0]['transformed_text'].str.cat(sep=" "))
plt.figure(figsize=(8,6))
plt.imshow(ham_wc)
```

Out[50]:

```
<matplotlib.image.AxesImage at 0x1f47dc37f10>
```

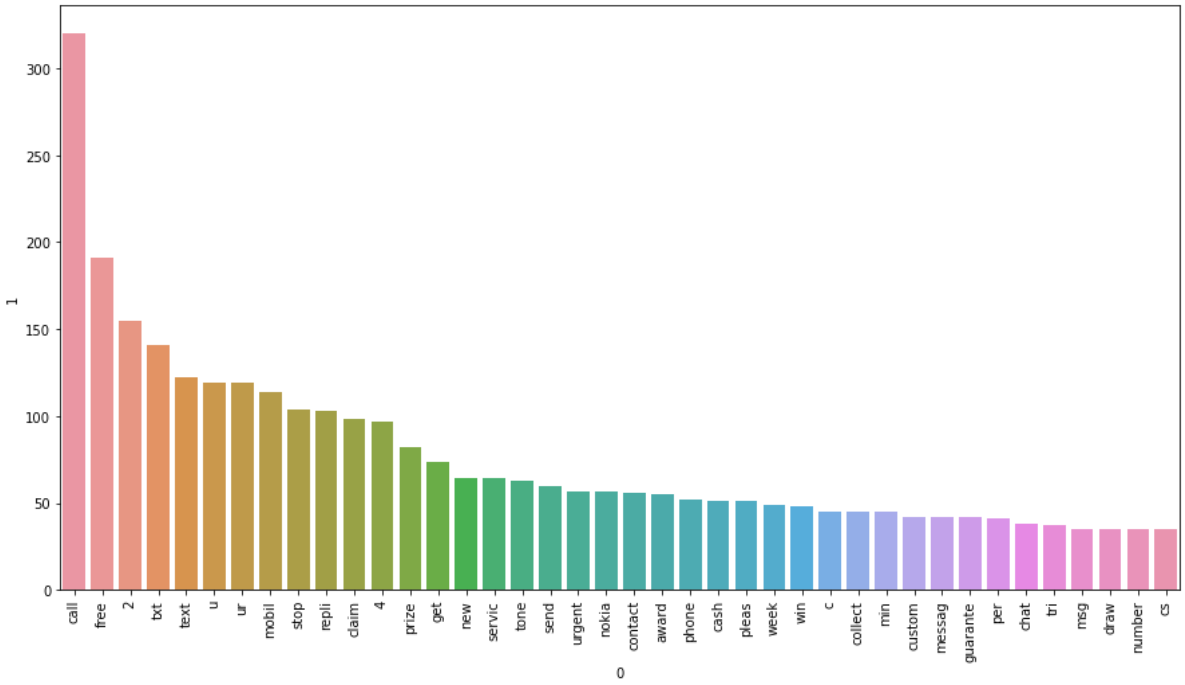
In [51]:

```
spam_y=[]
for msg in df[df['main']==1]['transformed_text'].tolist():
    for word in msg.split():
        spam_y.append(word)
```

In [52]:

```
#Counter for finding freq. of words
#top 40 most common words comes in spam
from collections import Counter
sns.barplot(pd.DataFrame(Counter(spam_y).most_common(40))[0], pd.DataFrame(Counter(spam_
plt.xticks(rotation='vertical')
plt.show()
```

```
C:\Users\lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Fut
ureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing ot
her arguments without an explicit keyword will result in an error or misin
terpretation.
  warnings.warn(
```



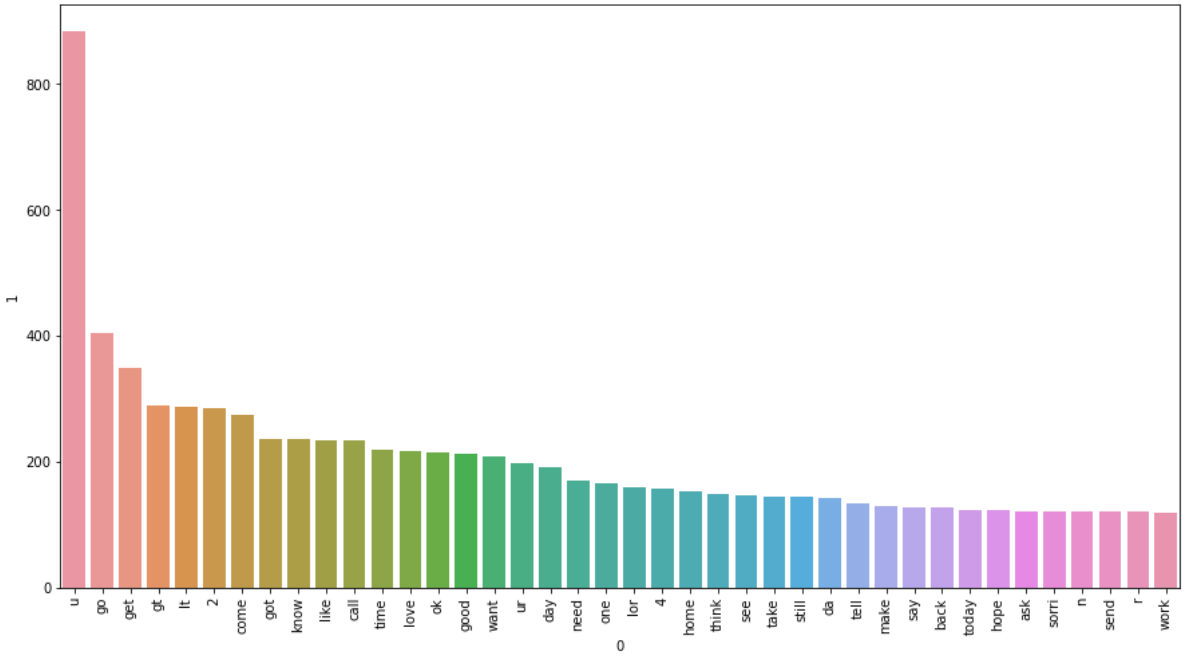
In [53]:

```
ham_y=[]
for msg in df[df['main']==0]['transformed_text'].tolist():
    for word in msg.split():
        ham_y.append(word)
```

In [54]:

```
from collections import Counter
sns.barplot(pd.DataFrame(Counter(ham_y).most_common(40))[0], pd.DataFrame(Counter(ham_y)
plt.xticks(rotation='vertical')
plt.show()
```

```
C:\Users\lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Fut
ureWarning: Pass the following variables as keyword args: x, y. From versi
on 0.12, the only valid positional argument will be `data`, and passing ot
her arguments without an explicit keyword will result in an error or misin
terpretation.
  warnings.warn(
```



Model Building

In [55]:

```
#for textual data naive baye's is best
```

In [56]:

```
from sklearn.feature_extraction.text import CountVectorizer ,TfidfVectorizer
```

In [57]:

```
cv=CountVectorizer()
tf= TfidfVectorizer(max_features=3000)
```

In [58]:

```
#sparse to dense array
X=tf.fit_transform(df['transformed_text']).toarray()
X
```

Out[58]:

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

In [59]:

```
X.shape
```

Out[59]:

```
(5169, 3000)
```

In [60]:

```
Y=df['main'].values
Y
```

Out[60]:

```
array([0, 0, 1, ..., 0, 0, 0])
```

In [61]:

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,confusion_matrix,precision_score
```

In [62]:

```
X_train,X_test,Y_train,Y_test= train_test_split(X,Y, test_size=0.2,random_state=2)
```

In [63]:

```
from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
gnb=GaussianNB()
mnb=MultinomialNB()
bnb=BernoulliNB()
```

In [64]:

```
gnb.fit(X_train,Y_train)
Y_predict1=gnb.predict(X_test)
print(accuracy_score(Y_test,Y_predict1))
print(confusion_matrix(Y_test,Y_predict1))
print(precision_score(Y_test,Y_predict1))
```

```
0.8694390715667312
[[788 108]
 [ 27 111]]
0.5068493150684932
```

In [65]:

```
mnb.fit(X_train,Y_train)
Y_predict2=mnb.predict(X_test)
print(accuracy_score(Y_test,Y_predict2))
print(confusion_matrix(Y_test,Y_predict2))
print(precision_score(Y_test,Y_predict2))

#in case of tfidf mnb is best imbalanced data is there so  precision score is best
# no false positive (a12==0)
```

```
0.9709864603481625
[[896   0]
 [ 30 108]]
1.0
```

In [66]:

```
bnb.fit(X_train,Y_train)
Y_predict3=bnb.predict(X_test)
print(accuracy_score(Y_test,Y_predict3))
print(confusion_matrix(Y_test,Y_predict3))
print(precision_score(Y_test,Y_predict3))
```

```
0.9835589941972921
[[895   1]
 [ 16 122]]
0.991869918699187
```