In [1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g pd.read_csv)
import nltk
```

In [9]:
```
df = pd.read_csv(r"D:\CAPSTONE PROJECT_DEPLOYMENT\11. CAPSTONE PROJECT_DEPLOYMEN
df.head()
```

Out[9]:

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

In [11]:
```
df= df.drop(["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"], axis=1)
df= df.rename(columns={'v1':'lable', 'v2':'sms'})
df.head(6)
```

Out[11]:

	lable	sms
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...
5	spam	FreeMsg Hey there darling it's been 3 week's n...

In [13]:

```
print(len(df))
```

```
5572
```

In [15]:

```
df.lable.value_counts()
```

Out[15]:

```
lable
ham     4825
spam     747
Name: count, dtype: int64
```

In [17]:

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

Out[17]:

```
403
```

In [19]:

```
df=df.drop_duplicates(keep='first')
```

In [21]:

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

Out[21]:

```
0
```

In [23]:

```
df.describe()
```

Out[23]:

	lable	sms
count	5169	5169
unique	2	5169
top	ham	Go until jurong point, crazy.. Available only ...
freq	4516	1

In [25]:

```
df.loc[:,'label'] = df.lable.map({'ham':0, 'spam':1})
print(df.shape)
df.head()
```

```
(5169, 3)
```

```
C:\Users\chitt\AppData\Local\Temp\ipykernel_12664\2601625900.py:1: SettingWithCop
yWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df.loc[:,'label'] = df.lable.map({'ham':0, 'spam':1})
```

Out[25]:

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

In [29]:

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

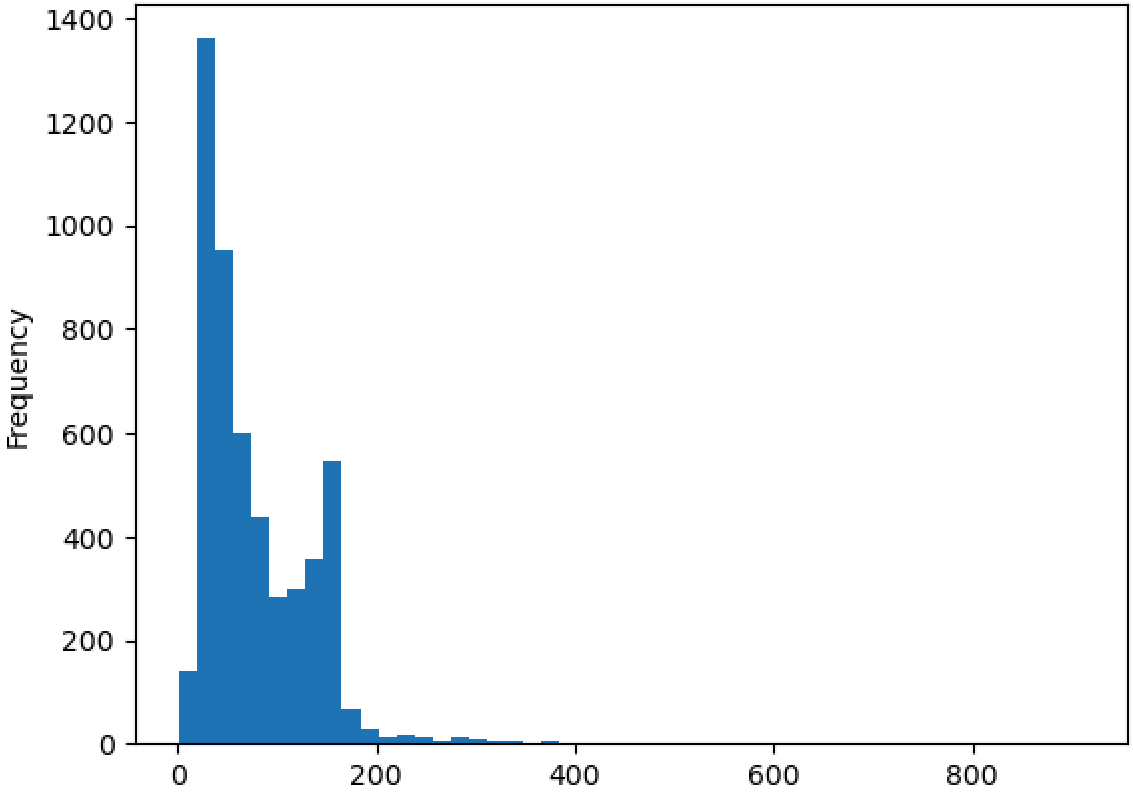
```
C:\Users\chitt\AppData\Local\Temp\ipykernel_12664\2372758055.py:1: SettingWithCop
yWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['num_characters'] = df['sms'].apply(len)
```

In [33]:

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
df['num_characters'].plot(bins=50, kind='hist')
```

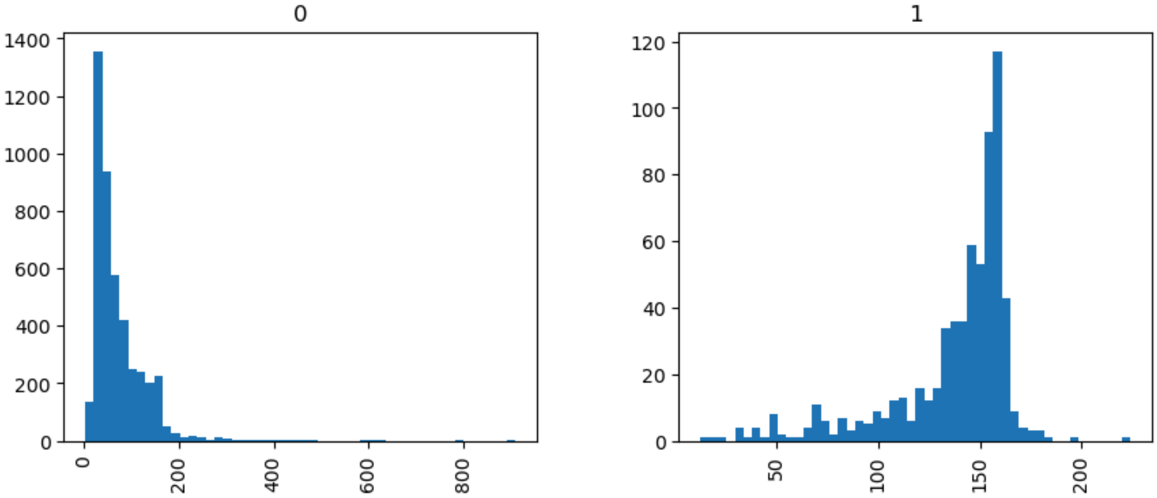
Out[33]:

```
<Axes: ylabel='Frequency'>
```

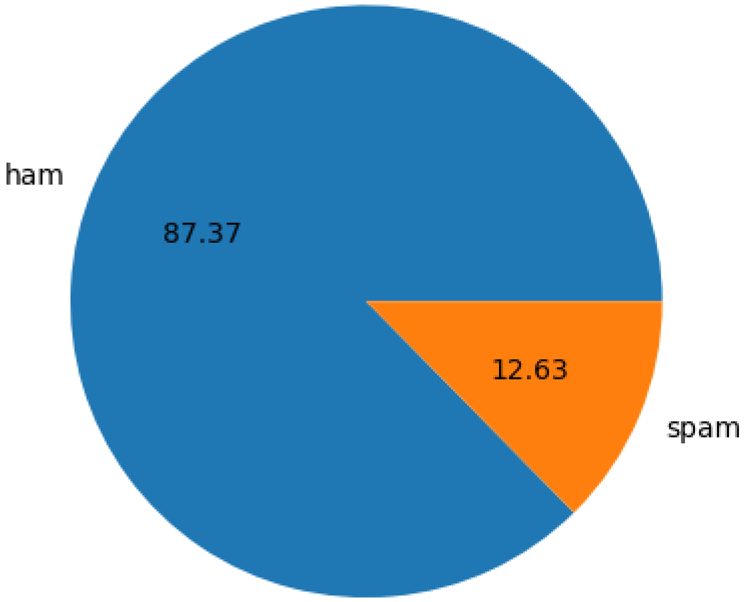


```
In [35]: df.hist(column='num_characters', by='label', bins=50, figsize=(10,4))
```

```
Out[35]: array([<Axes: title={'center': '0'}>, <Axes: title={'center': '1'}>],
               dtype=object)
```



```
In [39]: plt.pie(df['lable'].value_counts(), labels=['ham', 'spam'],autopct="%0.2f")
         plt.show()
```



In [41]:
```
# num of words
import nltk
nltk.download('punkt')
df['num_words'] = df['sms'].apply(lambda x:len(nltk.word_tokenize(x)))
```

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

In [43]:
```
df
```

Out[43]:

	lable	sms	label	num_characters	num_words
0	ham	Go until jurong point, crazy.. Available only ...	0	111	24
1	ham	Ok lar... Joking wif u oni...	0	29	8
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1	155	37
3	ham	U dun say so early hor... U c already then say...	0	49	13
4	ham	Nah I don't think he goes to usf, he lives aro...	0	61	15
...	...	...	...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...	1	161	35
5568	ham	Will Ì_ b going to esplanade fr home?	0	37	9
5569	ham	Pity, * was in mood for that. So...any other s...	0	57	15
5570	ham	The guy did some bitching but I acted like i'd...	0	125	27
5571	ham	Rofl. Its true to its name	0	26	7

5169 rows × 5 columns

In [45]:
```
df['num_sentences'] = df['sms'].apply(lambda x:len(nltk.sent_tokenize(x)))
```

In [47]:
```
df[['num_characters','num_words','num_sentences']].describe()
```

Out[47]:

	num_characters	num_words	num_sentences
count	5169.000000	5169.000000	5169.000000
mean	78.977945	18.455794	1.965564
std	58.236293	13.324758	1.448541
min	2.000000	1.000000	1.000000
25%	36.000000	9.000000	1.000000
50%	60.000000	15.000000	1.000000
75%	117.000000	26.000000	2.000000
max	910.000000	220.000000	38.000000

In [49]:
```
df[df['label'] == 0][['num_characters', 'num_words', 'num_sentences']].describe(
```

Out[49]:

	num_characters	num_words	num_sentences
count	4516.000000	4516.000000	4516.000000
mean	70.459256	17.123782	1.820195
std	56.358207	13.493970	1.383657
min	2.000000	1.000000	1.000000
25%	34.000000	8.000000	1.000000
50%	52.000000	13.000000	1.000000
75%	90.000000	22.000000	2.000000
max	910.000000	220.000000	38.000000

In [51]:

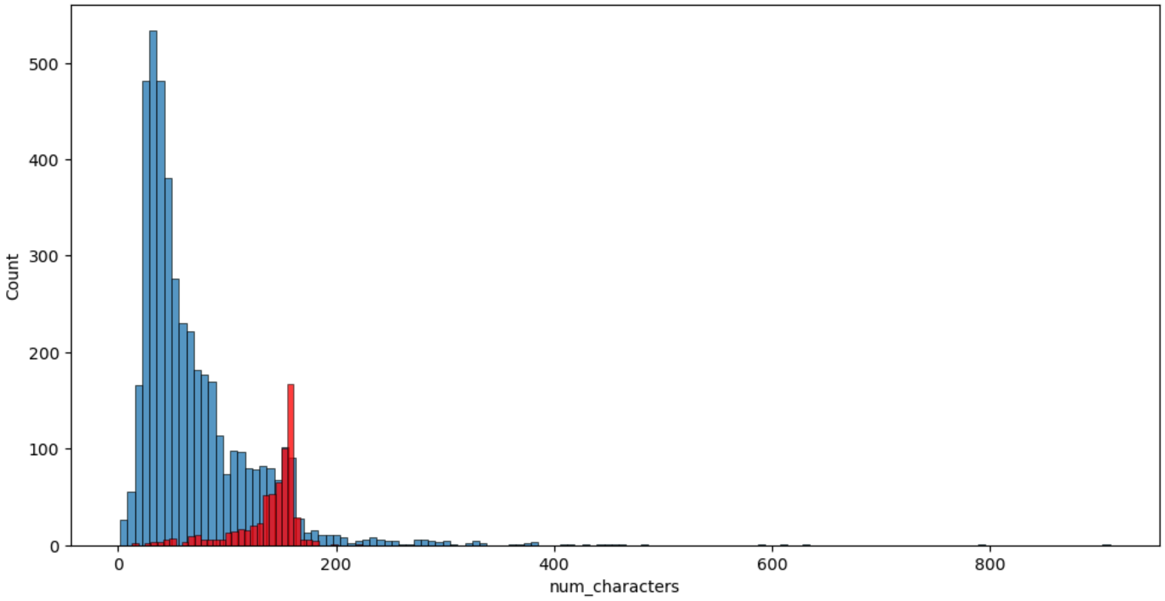
```
import seaborn as sns
```

In [57]:

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

Out[57]:

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

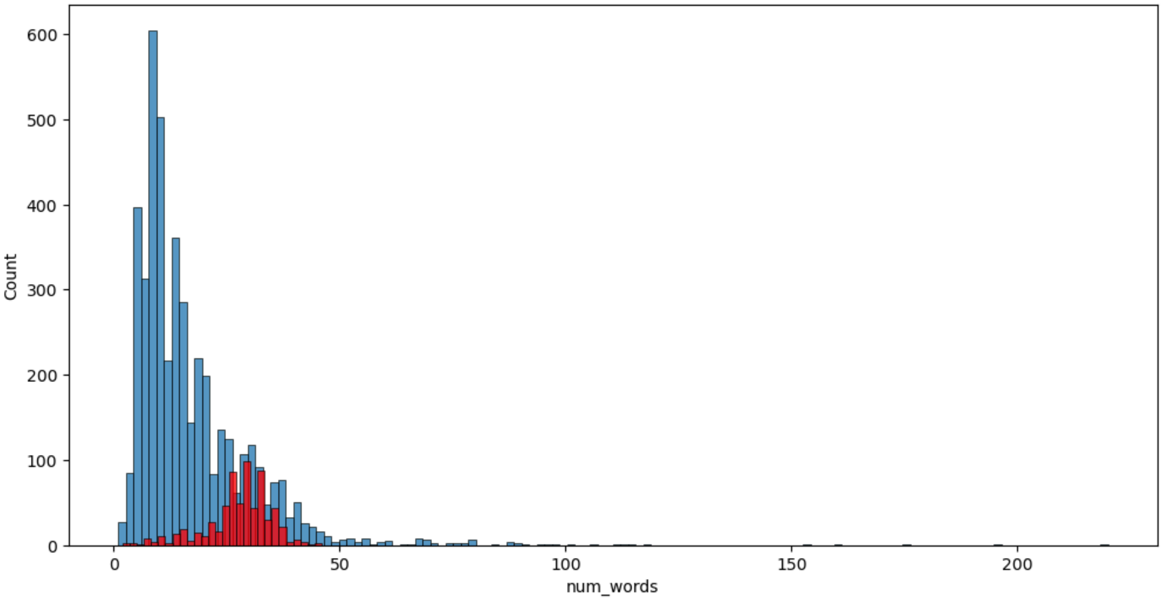


In [61]:

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

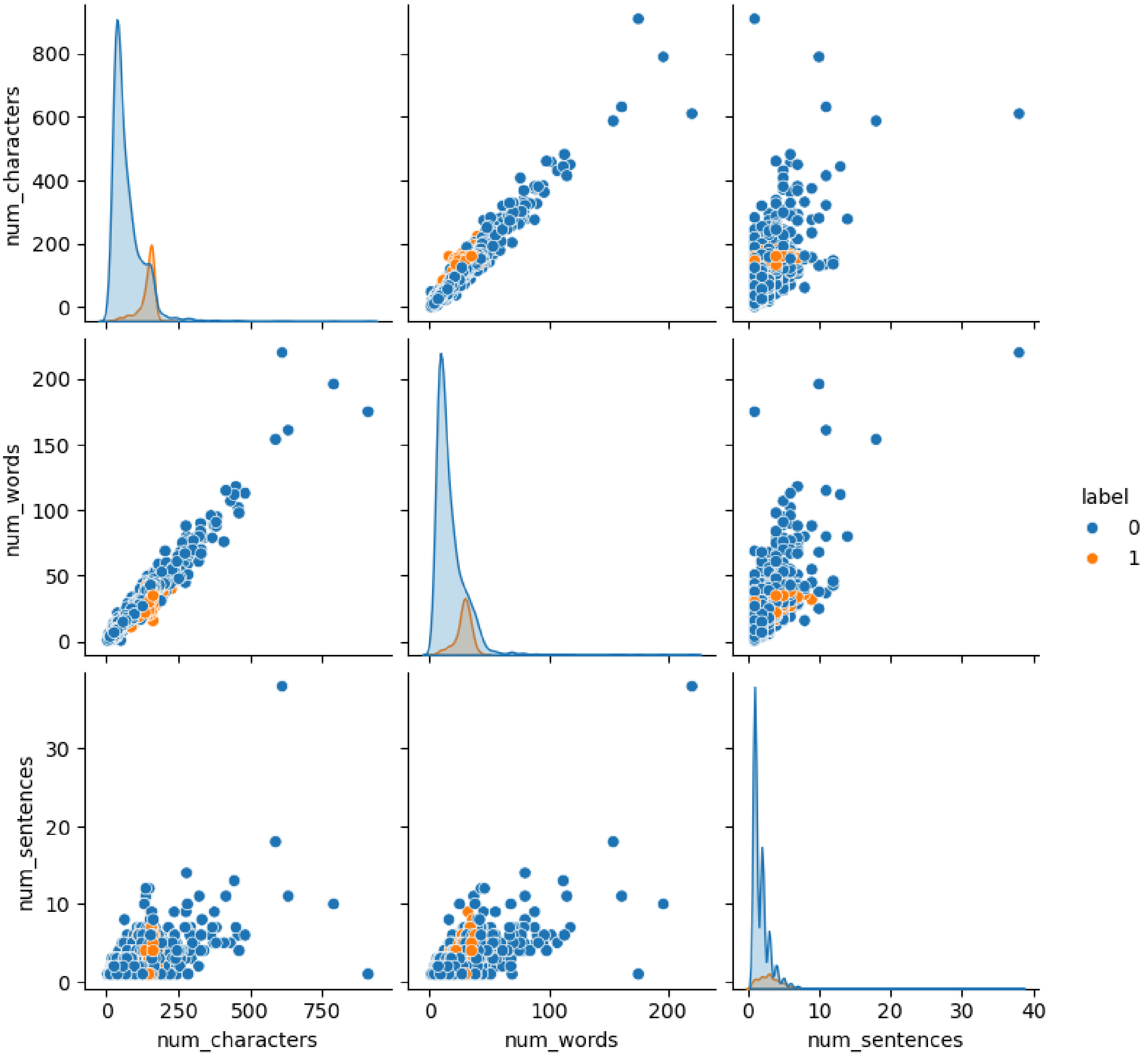
Out[61]:

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



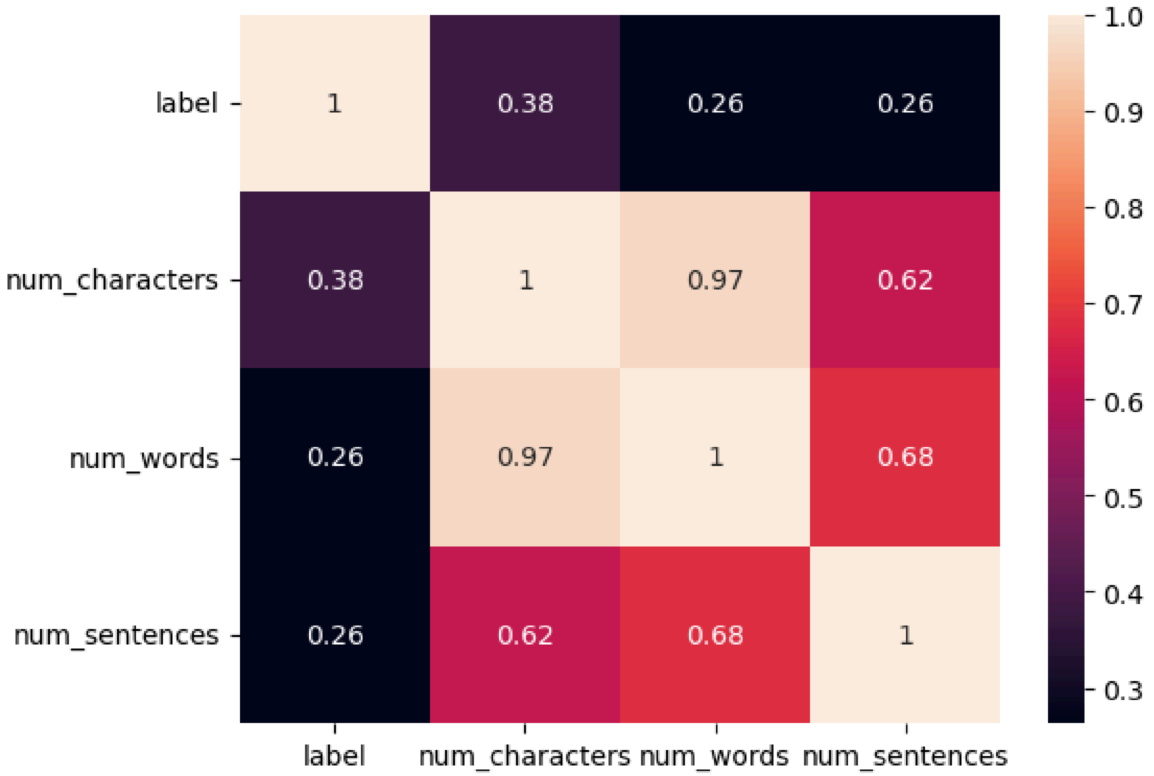
In [63]:
```
sns.pairplot(df,hue='label')
```

Out[63]:
```
<seaborn.axisgrid.PairGrid at 0x16dcdeaae70>
```



In [65]:
```
numeric_df = df.select_dtypes(include=['number'])

# Plot correlation heatmap
sns.heatmap(numeric_df.corr(), annot=True)
```

```
Out[65]: <Axes: >
```



Data Preprocessing

Lower case

Tokenization

Removing special characters

Removing stop words and punctuation

stemming

```
In [69]: df['sms'][10]
```

```
Out[69]: "I'm gonna be home soon and i don't want to talk about this stuff anymore tonig
         ht, k? I've cried enough today."
```

```
In [71]: from nltk.stem.porter import PorterStemmer
         ps = PorterStemmer()
         ps.stem('loving')
```

```
Out[71]: 'love'
```

```
In [73]: from nltk.corpus import stopwords
         import nltk
         nltk.download('stopwords')
```

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

Out[73]:

```
True
```

In [77]:

```
import nltk
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer

# Initialize Porter Stemmer
ps = PorterStemmer()

# Define the function to transform the text
def transform_text(text):
    text = text.lower()
    text = nltk.word_tokenize(text)

    y = []
    for i in text:
        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)

# Apply the text transformation
df['transformed_text'] = df['sms'].apply(transform_text)
```

In [79]:

```
df
```

Out[79]:

	lable	sms	label	num_characters	num_words	num_sentences	transformec
0	ham	Go until jurong point, crazy.. Available only ...	0	111	24	2	go jurong crazi avail b great w
1	ham	Ok lar... Joking wif u oni...	0	29	8	2	ok lar joke
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1	155	37	2	free entri comp win final tk
3	ham	U dun say so early hor... U c already then say...	0	49	13	1	u dun sa hor u c alrea
4	ham	Nah I don't think he goes to usf, he lives aro...	0	61	15	1	nah think g live a th
...	...	...	...	...	...	...	
5567	spam	This is the 2nd time we have tried 2 contact u...	1	161	35	4	2nd tim contact u p prize 2 cla
5568	ham	Will Ì_ b going to esplanade fr home?	0	37	9	1	b go espla
5569	ham	Pity, * was in mood for that. So...any other s...	0	57	15	2	piti mood su
5570	ham	The guy did some bitching but I acted like i'd...	0	125	27	1	guy bitch a interes someth els
5571	ham	Rofl. Its true to its	0	26	7	2	rofl true

lable	sms	label	num_characters	num_words	num_sentences	transformec
	name					

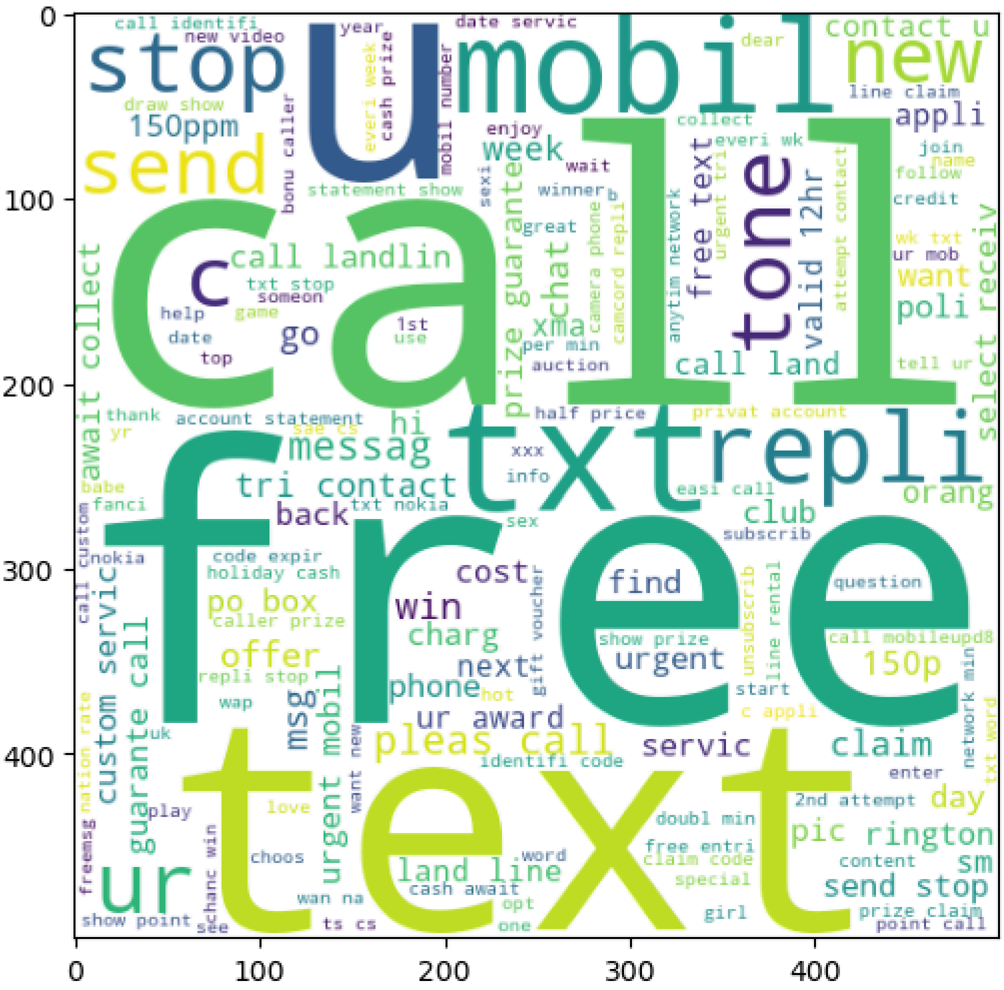
5169 rows × 7 columns

In [81]:
```
from wordcloud import WordCloud
wc = WordCloud(width=500, height=500,min_font_size=10,background_color='white')
```

In [83]:
```
spam_wc = wc.generate(df[df['label'] == 1]['transformed_text'].str.cat(sep=" "))
```

In [90]:
```
plt.figure(figsize=(15,6))
plt.imshow(spam_wc)
```

Out[90]:
```
<matplotlib.image.AxesImage at 0x16dd27406e0>
```



In [92]:
```
df.head()
```

Out[92]:

	lable	sms	label	num_characters	num_words	num_sentences	transformed_tex
0	ham	Go until jurong point, crazy.. Available only ...	0	111	24	2	go jurong poin crazi avail bugi r great world.
1	ham	Ok lar... Joking wif u oni...	0	29	8	2	ok lar joke wif u on
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1	155	37	2	free entri 2 wkl comp win fa cup final tkt 21.
3	ham	U dun say so early hor... U c already then say...	0	49	13	1	u dun say earl hor u c alreadi say
4	ham	Nah I don't think he goes to usf, he lives aro...	0	61	15	1	nah think goe us live arounc thougl

In [94]:

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

In [96]:

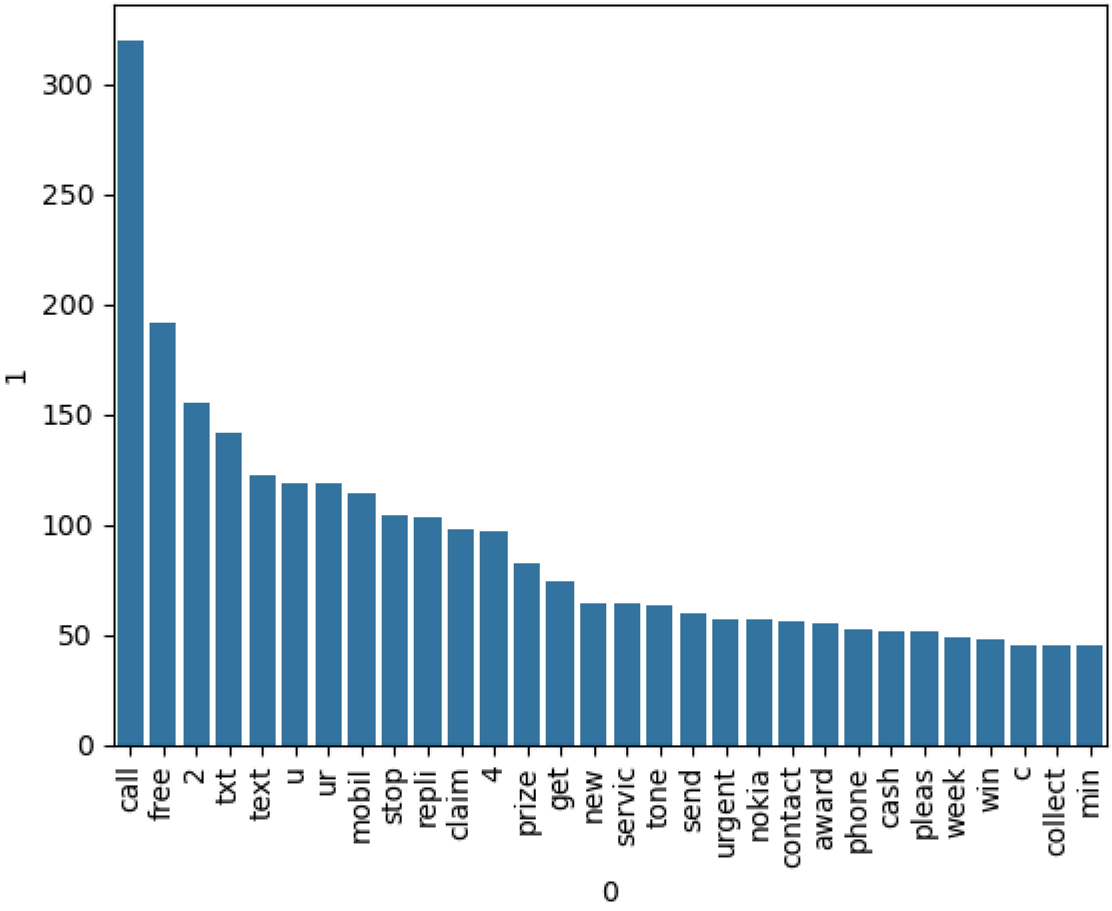
```
len(spam_corpus)
```

Out[96]:

```
9939
```

In [100…

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

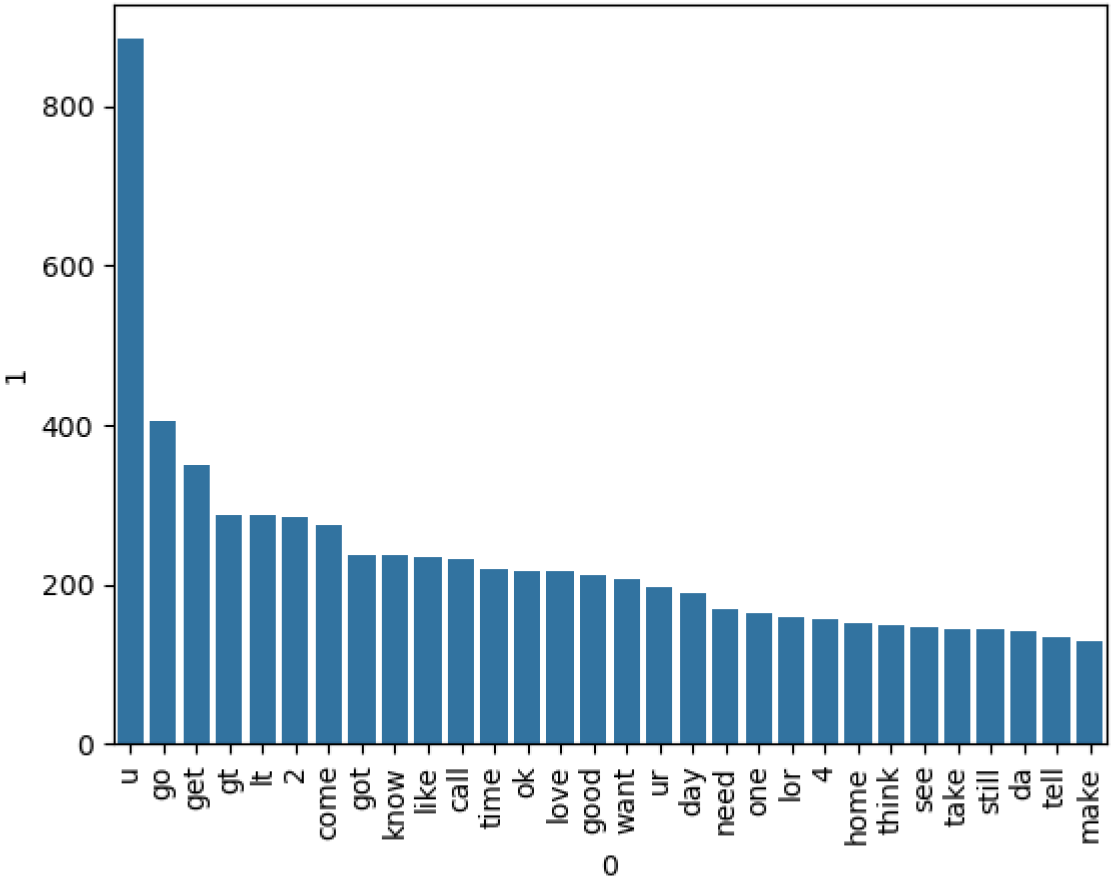


In [102…

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

In [106…

```
from collections import Counter
sns.barplot(x=pd.DataFrame(Counter(ham_corpus).most_common(30))[0], y=pd.DataFra
plt.xticks(rotation='vertical')
plt.show()
```



In [108…

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

Out[108…

	lable	sms	label	num_characters	num_words	num_sentences	transformed_tex
0	ham	Go until jurong point, crazy.. Available only ...	0	111	24	2	go jurong poin crazi avail bugi n great world..
1	ham	Ok lar... Joking wif u oni...	0	29	8	2	ok lar joke wif u on
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1	155	37	2	free entri 2 wkl comp win fa cup final tkt 21..
3	ham	U dun say so early hor... U c already then say...	0	49	13	1	u dun say earl hor u c alreadi say
4	ham	Nah I don't think he goes to usf, he lives aro...	0	61	15	1	nah think goe us live around though

Model Building

In [111…

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

In [113…

```
X = tfidf.fit_transform(df['transformed_text']).toarray()
```

In [115…

```
X.shape
```

Out[115…

```
(5169, 3000)
```

In [117…

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

In [119…

```
from sklearn.model_selection import train_test_split
```

In [121…

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

In [123…

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

In [125…

```
gnb = GaussianNB
mnb = MultinomialNB
bnb = BernoulliNB
```

In [127…

```
from sklearn.preprocessing import LabelEncoder

# Initialize LabelEncoder
label_encoder = LabelEncoder()

# fit and trensform label in y_train
y_train_encoded = label_encoder.fit_transform(y_train)

# Transform labels in y_test
y_test_encoded = label_encoder.transform(y_test)
```

In [137…

```
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score

# Initialize GaussianNB
gnb = GaussianNB()

# Fit the model
gnb.fit(X_train, y_train_encoded)

# Predict on the test set
y_pred1 = gnb.predict(X_test)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test_encoded, y_pred1))
print("Confusion Matrix:\n", confusion_matrix(y_test_encoded, y_pred1))
print("Precision:", precision_score(y_test_encoded, y_pred1))
```

```
Accuracy: 0.8694390715667312
Confusion Matrix:
 [[788 108]
 [ 27 111]]
Precision: 0.5068493150684932
```

In [141…

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score

# Initialize Multinomial Naive Bayes
mnb = MultinomialNB()

# Fit the model
mnb.fit(X_train, y_train_encoded)

# Predict on the test set
y_pred1 = mnb.predict(X_test)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test_encoded, y_pred1))
print("Confusion Matrix:\n", confusion_matrix(y_test_encoded, y_pred1))
print("Precision:", precision_score(y_test_encoded, y_pred1))
```

```
Accuracy: 0.9709864603481625
Confusion Matrix:
 [[896   0]
 [ 30 108]]
Precision: 1.0
```

In [143…

```
from sklearn.naive_bayes import BernoulliNB
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score

# Initialize Bernoulli Naive Bayes
bnb = BernoulliNB()

# Fit the model
bnb.fit(X_train, y_train_encoded)

# Predict on the test set
y_pred1 = bnb.predict(X_test)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test_encoded, y_pred1))
print("Confusion Matrix:\n", confusion_matrix(y_test_encoded, y_pred1))
print("Precision:", precision_score(y_test_encoded, y_pred1))
```

```
Accuracy: 0.9835589941972921
Confusion Matrix:
 [[895   1]
 [ 16 122]]
Precision: 0.991869918699187
```

In [147…

```
from sklearn.ensemble import ExtraTreesClassifier
etc = ExtraTreesClassifier(n_estimators=50, random_state=2)
```

In [149…

```
from sklearn.svm import SVC
svc = SVC(kernel='sigmoid', gamma=1.0, probability=True)
mnb = MultinomialNB()
etc = ExtraTreesClassifier(n_estimators=50, random_state=2)

from sklearn.ensemble import VotingClassifier
```

In [151…

```
voting = VotingClassifier(estimators=[('svm', svc), ('nb', mnb), ('et', etc)],vc
```

In [154…

```
y_pred1 = voting.predict(X_test)
print("Accuracy",accuracy_score(y_test_encoded,y_pred1))
print("Precision",precision_score(y_test_encoded,y_pred1))
```

```
Accuracy 0.9816247582205029
Precision 0.9917355371900827
```

Completed

In [ ]: