In [ ]:

```python

```

In [69]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

import nltk
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
from nltk.corpus import stopwords

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

from collections import Counter

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

import string
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
```

In [2]:

```python
df = pd.read_csv('spam_classifier\\spam.csv',encoding='ISO-8859-1')
df
```

Out[2]:

	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
...	...	...	...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...	NaN	NaN	NaN
5568	ham	Will Ì_ b going to esplanade fr home?	NaN	NaN	NaN
5569	ham	Pity, * was in mood for that. So...any other s...	NaN	NaN	NaN
5570	ham	The guy did some bitching but I acted like i'd...	NaN	NaN	NaN
5571	ham	Rofl. Its true to its name	NaN	NaN	NaN

5572 rows × 5 columns

1 . Data Cleaning

In [3]:

```python
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 [4]:

```python
# droping columns
df.drop(columns = ['Unnamed: 2','Unnamed: 3','Unnamed: 4'],inplace=True )
df
```

Out[4]:

	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...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will Ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

In [5]:

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

Out[5]:

	target	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...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will Ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

In [6]:

```python
# changing ham = 0 and spam = 1
df['target'] = encoder.fit_transform(df['target'])
df.head()
```

Out[6]:

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

In [7]:
```python
#missing values or not
df.isnull().sum()
```
Out[7]:
```
target    0
text      0
dtype: int64
```

In [8]:
```python
# check for duplicate values
df.duplicated().sum()
```
Out[8]:
```
403
```

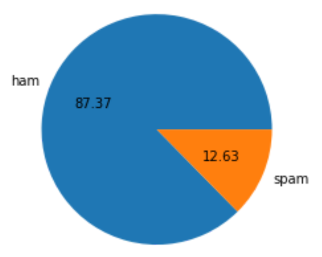
In [9]:
```python
# removing duplicates
df = df.drop_duplicates(keep='first')
df.duplicated().sum()
```
Out[9]:
```
0
```

In [10]:
```python
df.shape
```
Out[10]:
```
(5169, 2)
```

2. EDA

In [11]:
```python
df['target'].value_counts()
```
Out[11]:
```
0    4516
1     653
Name: target, dtype: int64
```

In [12]:
```python
plt.pie(df['target'].value_counts(),labels=['ham','spam'],autopct="%0.2f")
plt.show()

```



In [13]:
```python
# Data is imbalanced
```

In [14]:
```python
# num of characters
df['num_characters'] = df['text'].apply(len)
df.head()
```
```
C:\Users\Admin\AppData\Local\Temp/ipykernel_21892/4041876216.py:2: SettingWithCopyWarning:
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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy)
  df['num_characters'] = df['text'].apply(len)
```
Out[14]:

	target	text	num_characters
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 [15]:
```python
# num of words
df['num_words'] = df['text'].apply(lambda x:len(nltk.word_tokenize(x)))
df.head()
```
```
C:\Users\Admin\AppData\Local\Temp/ipykernel_21892/2894533858.py:2: SettingWithCopyWarning:
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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy)
  df['num_words'] = df['text'].apply(lambda x:len(nltk.word_tokenize(x)))
```
Out[15]:

	target	text	num_characters	num_words
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 [16]:
```python
# num of sentence
df['num_sentences'] = df['text'].apply(lambda x:len(nltk.sent_tokenize(x)))
df.head()
```
```
C:\Users\Admin\AppData\Local\Temp/ipykernel_21892/100777245.py:2: SettingWithCopyWarning:
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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy)
  df['num_sentences'] = df['text'].apply(lambda x:len(nltk.sent_tokenize(x)))
```
Out[16]:

	target	text	num_characters	num_words	num_sentences
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 [17]:
```python
df[['num_characters','num_words','num_sentences']].describe()
```
Out[17]:

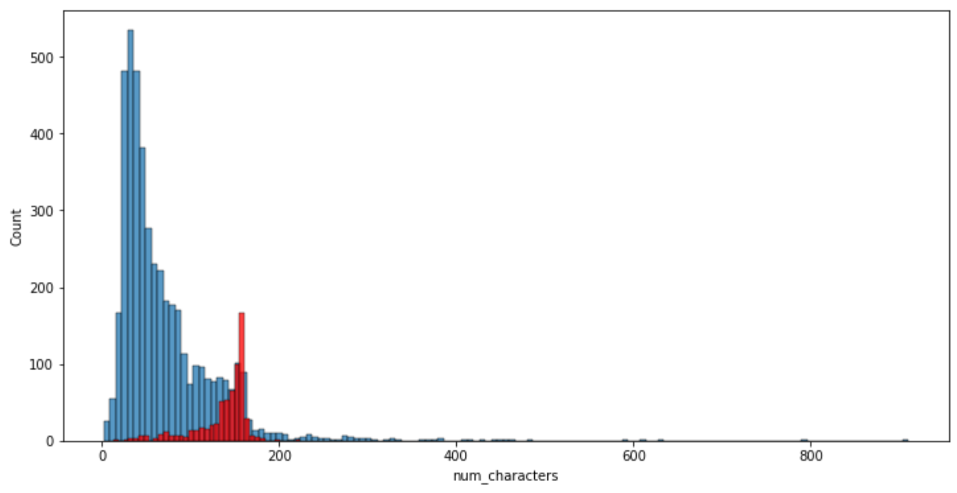
	num_characters	num_words	num_sentences
count	5169.000000	5169.000000	5169.000000
mean	78.977945	18.453279	1.947185
std	58.236293	13.324793	1.362406
min	2.000000	1.000000	1.000000
25%	36.000000	9.000000	1.000000
50%	60.000000	15.000000	1.000000
75%	117.000000	26.000000	2.000000
max	910.000000	220.000000	28.000000

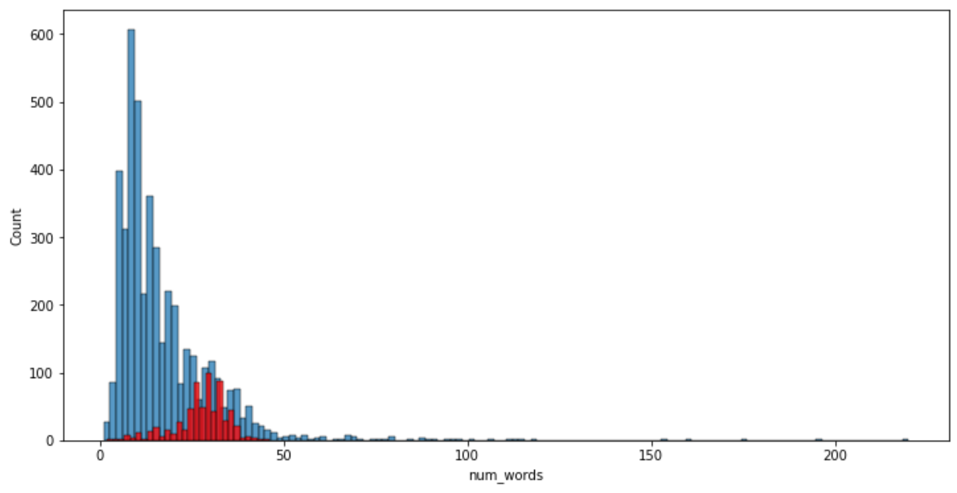
In [18]:
```python
# ham msg desc
df[df['target'] == 0][['num_characters','num_words','num_sentences']].describe()
```
Out[18]:

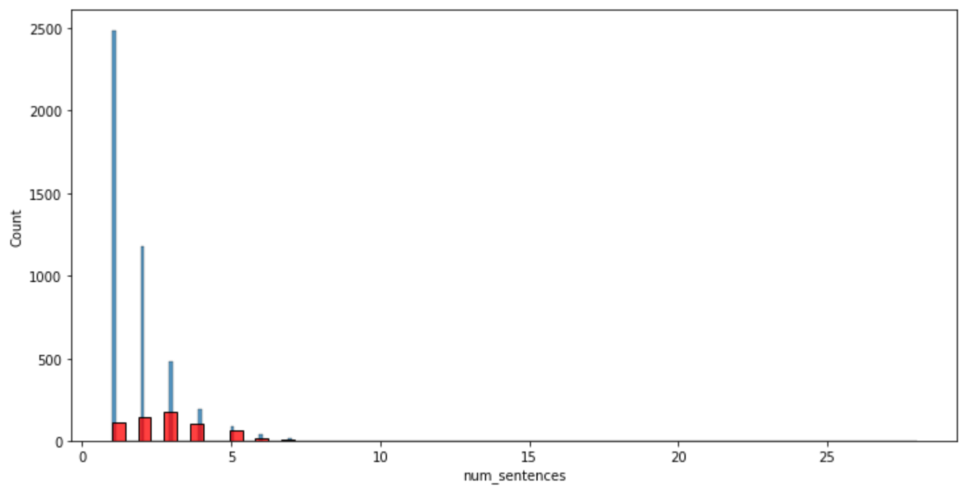
	num_characters	num_words	num_sentences
count	4516.000000	4516.000000	4516.000000
mean	70.459256	17.120903	1.799601
std	56.358207	13.493725	1.278465
min	2.000000	1.000000	1.000000
25%	34.000000	8.000000	1.000000
50%	52.000000	13.000000	1.000000
75%	90.000000	22.000000	2.000000
max	910.000000	220.000000	28.000000

In [19]:
```python
# spam msg desc
df[df['target'] == 1][['num_characters','num_words','num_sentences']].describe()
```
Out[19]:

	num_characters	num_words	num_sentences
count	653.000000	653.000000	653.000000
mean	137.891271	27.667688	2.967841
std	30.137753	7.008418	1.483201
min	13.000000	2.000000	1.000000
25%	132.000000	25.000000	2.000000
50%	149.000000	29.000000	3.000000
75%	157.000000	32.000000	4.000000
max	224.000000	46.000000	8.000000

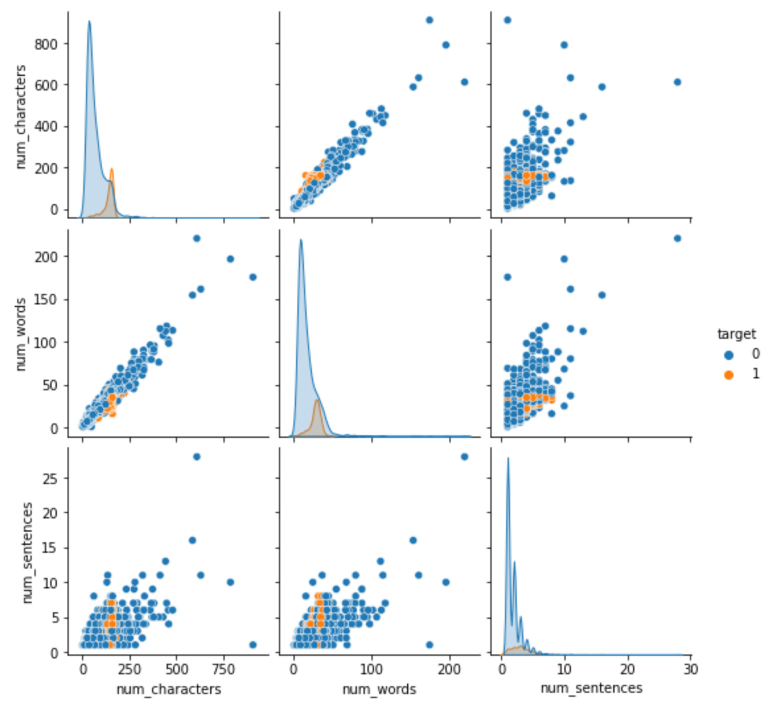
In [20]:
```python
plt.figure(figsize=(12,6))
sns.histplot(df[df['target'] == 0]['num_characters'])
sns.histplot(df[df['target'] == 1]['num_characters'],color='red')
```
Out[20]:
```
<AxesSubplot:xlabel='num_characters', ylabel='Count'>
```


In [21]:
```python
plt.figure(figsize=(12,6))
sns.histplot(df[df['target'] == 0]['num_words'])
sns.histplot(df[df['target'] == 1]['num_words'],color='red')
```
Out[21]:
```
<AxesSubplot:xlabel='num_words', ylabel='Count'>
```


In [22]:
```python
plt.figure(figsize=(12,6))
sns.histplot(df[df['target'] == 0]['num_sentences'])
sns.histplot(df[df['target'] == 1]['num_sentences'],color='red')
```
Out[22]:
```
<AxesSubplot:xlabel='num_sentences', ylabel='Count'>
```


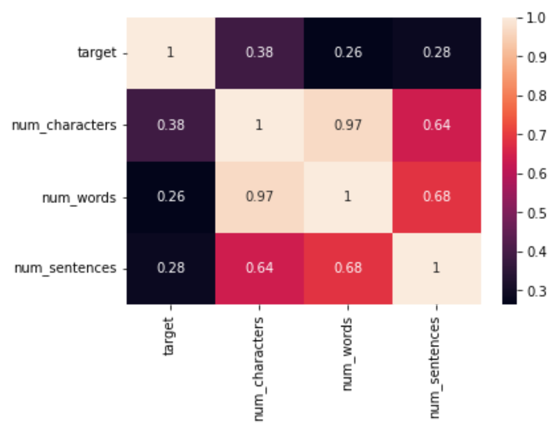
```
In [23]: sns.pairplot(df,hue='target')
```

```
Out[23]: <seaborn.axisgrid.PairGrid at 0x21cc1b136d0>
```



```
In [24]: sns.heatmap(df.corr(),annot=True)
```

```
Out[24]: <AxesSubplot:>
```



3. Data Prepocessing

Lower Case

Tokenization

Removing special characters

Removing stop words and punctuations

Steming

```
In [25]:
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)
```

```
In [26]:
df['transformed_text'] = df['text'].apply(transform_text)
df
```

```
C:\Users\Admin\AppData\Local\Temp/ipykernel_21892/1835954565.py:1: SettingWithCopyWarning:
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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df['transformed_text'] = df['text'].apply(transform_text)
```

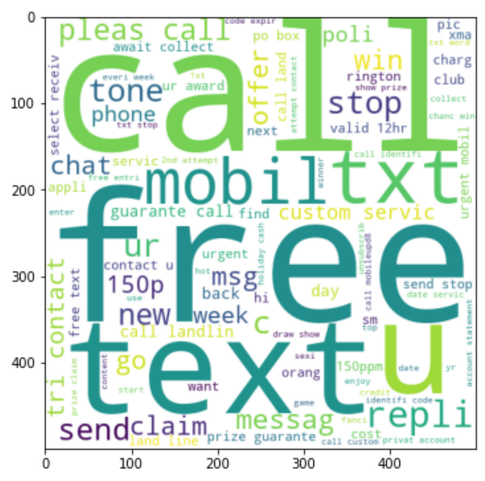
Out[26]:

	target	text	num_characters	num_words	num_sentences	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
...	...	...	...	...	...	...
5567	1	This is the 2nd time we have tried 2 contact u...	161	35	4	2nd time tri 2 contact u pound prize 2 claim e...
5568	0	Will Ì_ b going to esplanade fr home?	37	9	1	b go esplanad fr home
5569	0	Pity, * was in mood for that. So...any other s...	57	15	2	piti mood suggest
5570	0	The guy did some bitching but I acted like i'd...	125	27	1	guy bitch act like interest buy someth els nex...
5571	0	Rofl. Its true to its name	26	7	2	rofl true name

5169 rows × 6 columns

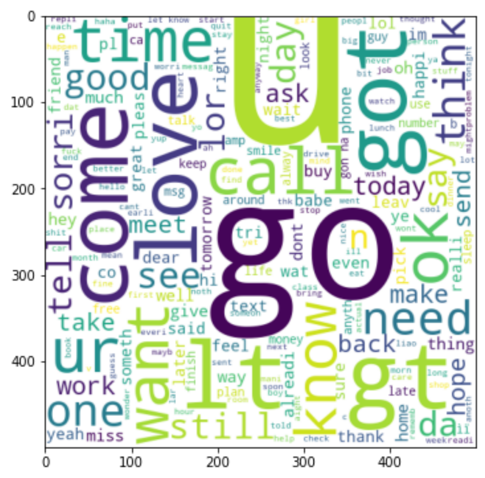
In [27]:
```python
# Spam wordcloud
spam_wc = wc.generate(df[df['target'] == 1]['transformed_text'].str.cat(sep=" "))
plt.figure(figsize=(15,6))
plt.imshow(spam_wc)
```

Out[27]:
```
<matplotlib.image.AxesImage at 0x21cc5ccf6d0>
```



In [28]:
```python
# Ham wordcloud
ham_wc = wc.generate(df[df['target'] == 0]['transformed_text'].str.cat(sep=" "))
plt.figure(figsize=(15,6))
plt.imshow(ham_wc)
```

Out[28]:
```
<matplotlib.image.AxesImage at 0x21cc5c9d870>
```



In [29]:
```python
df.head()
```

Out[29]:

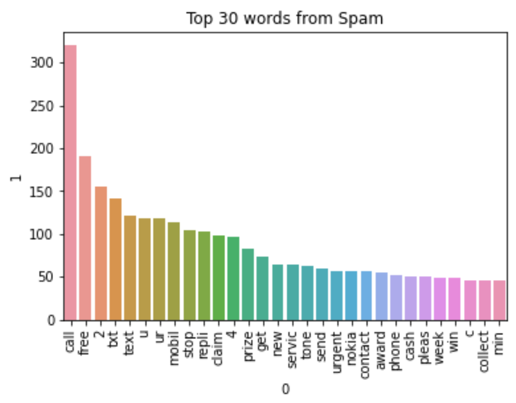
	target	text	num_characters	num_words	num_sentences	transformed_text
0	0	Go until jurong point, crazy.. Available only ...	111	24	2	go jurong point crazi avail bugi n great world...
1	0	Ok lar... Joking wif u oni...	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	155	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 [30]:
```python
spam_corpus=[]
for msg in df[df['target'] == 1]['transformed_text'].tolist():
    for word in msg.split():
        spam_corpus.append(word)
print(f"Spam corpus : {len(spam_corpus)}")

ham_corpus=[]
for msg in df[df['target'] == 0]['transformed_text'].tolist():
    for word in msg.split():
        ham_corpus.append(word)
print(f"Ham corpus : {len(ham_corpus)}")

```

```
Spam corpus : 9939
Ham corpus : 35394
```

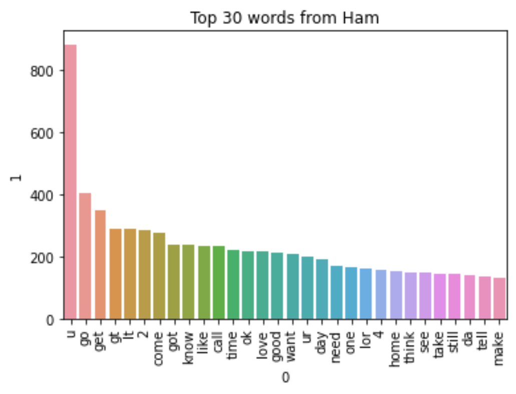
In [31]:
```python
# Spam Corpus barplot for top 30 words
sns.barplot(pd.DataFrame(Counter(spam_corpus).most_common(30))[0],pd.DataFrame(Counter(spam_corpus).most_common(30))[1]).set(title='Top 30 words from Spam')
plt.xticks(rotation="vertical")
plt.show()
```

```
C:\Python310\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and pas
sing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [32]:
```python
# Ham Corpus barplot for top 30 words
sns.barplot(pd.DataFrame(Counter(ham_corpus).most_common(30))[0],pd.DataFrame(Counter(ham_corpus).most_common(30))[1]).set(title='Top 30 words from Ham')
plt.xticks(rotation="vertical")
plt.show()
```

```
C:\Python310\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and pas
sing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



4 . Model Building

In [70]:
```python
df.head()
```

Out[70]:

	target	text	num_characters	num_words	num_sentences	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 [71]:
```python
X = tfidf.fit_transform(df['transformed_text']).toarray()
X.shape
```

Out[71]:
```
(5169, 6708)
```

In [72]:
```python
y = df['target'].values
y
```

Out[72]:
```
array([0, 0, 1, ..., 0, 0, 0])
```

In [73]:
```python
from sklearn.model_selection import train_test_split
```

In [74]:
```python
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=2)
```

In [75]:
```python
from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
from sklearn.metrics import  accuracy_score,confusion_matrix,precision_score
```

In [76]:
```python
gnb = GaussianNB()
mnb = MultinomialNB()
bnb = BernoulliNB()
```

In [77]:
```python
gnb.fit(X_train,y_train)
y_pred_1 = gnb.predict(X_test)
print(f"Accuracy_score : {accuracy_score(y_test,y_pred_1)} ")
print(f"Confusion_matrix : \n{confusion_matrix(y_test,y_pred_1)} ")
print(f"Precision_score : {precision_score(y_test,y_pred_1)} ")
```

```
Accuracy_score : 0.8762088974854932
Confusion_matrix :
[[793 103]
 [ 25 113]]
Precision_score : 0.5231481481481481
```

In [78]:
```python
mnb.fit(X_train,y_train)
y_pred_2 = mnb.predict(X_test)
print(f"Accuracy_score : {accuracy_score(y_test,y_pred_2)} ")
print(f"Confusion_matrix : \n{confusion_matrix(y_test,y_pred_2)} ")
print(f"Precision_score : {precision_score(y_test,y_pred_2)} ")
```

```
Accuracy_score : 0.9593810444874274
Confusion_matrix :
[[896   0]
 [ 42  96]]
Precision_score : 1.0
```

In [100]:
```python
rfc.fit(X_train,y_train)
y_pred_4 = rfc.predict(X_test)
print(f"Accuracy_score : {accuracy_score(y_test,y_pred_4)} ")
print(f"Confusion_matrix : \n{confusion_matrix(y_test,y_pred_4)} ")
print(f"Precision_score : {precision_score(y_test,y_pred_4)} ")
```

```
Accuracy_score : 0.9738878143133463
Confusion_matrix :
[[896   0]
 [ 27 111]]
Precision_score : 1.0
```

In [80]:
```python
# tfidf --> MNB
```

In [81]:
```python
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
```

In [82]:
```python
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)
```

In [83]:
```
clfs = {
    'SVC' : svc,
    'KN' : knc,
    'NB': mnb,
    'DT': dtc,
    'LR': lrc,
    'RF': rfc,
    'AdaBoost': abc,
    'BgC': bc,
    'ETC': etc,
    'GBDT':gbdt,
    'xgb':xgb
}
```

In [84]:
```
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
```

In [85]:
```
train_classifier(svc,X_train,y_train,X_test,y_test)
```

Out[85]:
```
(0.9729206963249516, 0.9741379310344828)
```

In [86]:
```
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)
    print("/////////////////////////////////////////")


    accuracy_scores.append(current_accuracy)
    precision_scores.append(current_precision)
```

```
For  SVC
Accuracy -  0.9729206963249516
Precision -  0.9741379310344828
/////////////////////////////////////////
For  KN
Accuracy -  0.9003868471953579
Precision -  1.0
/////////////////////////////////////////
For  NB
Accuracy -  0.9593810444874274
Precision -  1.0
/////////////////////////////////////////
For  DT
Accuracy -  0.9352030947775629
Precision -  0.8380952380952381
/////////////////////////////////////////
For  LR
Accuracy -  0.9516441005802708
Precision -  0.94
/////////////////////////////////////////
For  RF
Accuracy -  0.9738878143133463
Precision -  1.0
/////////////////////////////////////////
For  AdaBoost
Accuracy -  0.9613152804642167
Precision -  0.9454545454545454
/////////////////////////////////////////
For  BgC
Accuracy -  0.9584139264990329
Precision -  0.8625954198473282
/////////////////////////////////////////
For  ETC
Accuracy -  0.9758220502901354
Precision -  0.9829059829059829
/////////////////////////////////////////
For  GBDT
Accuracy -  0.9526112185686654
Precision -  0.9238095238095239
/////////////////////////////////////////
For  xgb
Accuracy -  0.9690522243713733
Precision -  0.9344262295081968
/////////////////////////////////////////
```

In [87]:
```
performance_df = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy':accuracy_scores,
                               'Precision':precision_scores}).sort_values('Precision',ascending=False)
performance_df
```

Out[87]:

	Algorithm	Accuracy	Precision
1	KN	0.900387	1.000000
2	NB	0.959381	1.000000
5	RF	0.973888	1.000000
8	ETC	0.975822	0.982906
0	SVC	0.972921	0.974138
6	AdaBoost	0.961315	0.945455
4	LR	0.951644	0.940000
10	xgb	0.969052	0.934426
9	GBDT	0.952611	0.923810
7	BgC	0.958414	0.862595
3	DT	0.935203	0.838095

In [89]:

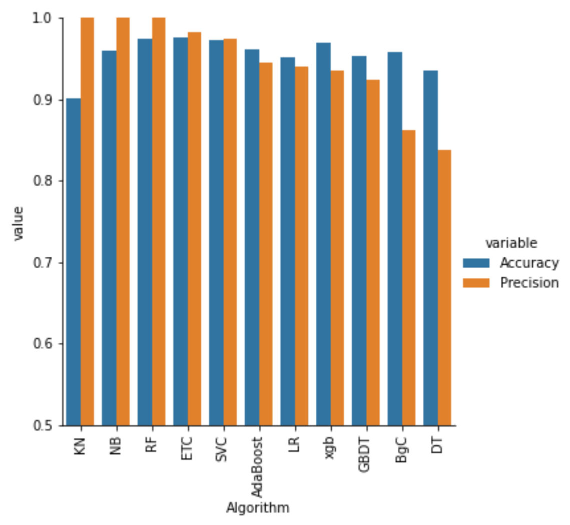
```python
performance_df1 = pd.melt(performance_df, id_vars = "Algorithm")
performance_df1

```

Out[89]:

	Algorithm	variable	value
0	KN	Accuracy	0.900387
1	NB	Accuracy	0.959381
2	RF	Accuracy	0.973888
3	ETC	Accuracy	0.975822
4	SVC	Accuracy	0.972921
5	AdaBoost	Accuracy	0.961315
6	LR	Accuracy	0.951644
7	xgb	Accuracy	0.969052
8	GBDT	Accuracy	0.952611
9	BgC	Accuracy	0.958414
10	DT	Accuracy	0.935203
11	KN	Precision	1.000000
12	NB	Precision	1.000000
13	RF	Precision	1.000000
14	ETC	Precision	0.982906
15	SVC	Precision	0.974138
16	AdaBoost	Precision	0.945455
17	LR	Precision	0.940000
18	xgb	Precision	0.934426
19	GBDT	Precision	0.923810
20	BgC	Precision	0.862595
21	DT	Precision	0.838095

In [90]:

```python
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()
# tfidf --> rfc
```



In [68]:

```python
# model improve
# 1. Change the max_features parameter of TfIdf
df
```

Out[68]:

	target	text	num_characters	num_words	num_sentences	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
...	...	...	...	...	...	...
5567	1	This is the 2nd time we have tried 2 contact u...	161	35	4	2nd time tri 2 contact u pound prize 2 claim e...
5568	0	Will Ì_ b going to esplanade fr home?	37	9	1	b go esplanad fr home
5569	0	Pity, * was in mood for that. So...any other s...	57	15	2	piti mood suggest
5570	0	The guy did some bitching but I acted like i'd...	125	27	1	guy bitch act like interest buy someth els nex...
5571	0	Rofl. Its true to its name	26	7	2	rofl true name

5169 rows × 6 columns

In [67]:

```python
temp_df = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy_max_ft_3000':accuracy_scores,'Precision_max_ft_3000':precision_scores}).sort_values('Precision_max_ft_3000',ascending=False)
temp_df
```

Out[67]:

	Algorithm	Accuracy_max_ft_3000	Precision_max_ft_3000
1	KN	0.905222	1.000000
2	NB	0.970986	1.000000
5	RF	0.974855	0.982759
0	SVC	0.975822	0.974790
8	ETC	0.974855	0.974576
4	LR	0.958414	0.970297
10	xgb	0.971954	0.943089
6	AdaBoost	0.960348	0.929204
9	GBDT	0.947776	0.920000
7	BgC	0.957447	0.867188
3	DT	0.927466	0.811881

In [66]:
```python
temp_df = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy_scaling':accuracy_scores,'Precision_scaling':precision_scores}).sort_values('Precision_scaling',ascending=False)
temp_df
```

Out[66]:

	Algorithm	Accuracy_scaling	Precision_scaling
1	KN	0.905222	1.000000
2	NB	0.970986	1.000000
5	RF	0.974855	0.982759
0	SVC	0.975822	0.974790
8	ETC	0.974855	0.974576
4	LR	0.958414	0.970297
10	xgb	0.971954	0.943089
6	AdaBoost	0.960348	0.929204
9	GBDT	0.947776	0.920000
7	BgC	0.957447	0.867188
3	DT	0.927466	0.811881

In [62]:
```python
new_df = performance_df.merge(temp_df,on='Algorithm')
new_df
```

Out[62]:

	Algorithm	Accuracy	Precision	Accuracy_scaling	Precision_scaling
0	KN	0.905222	1.000000	0.905222	1.000000
1	NB	0.970986	1.000000	0.970986	1.000000
2	RF	0.974855	0.982759	0.974855	0.982759
3	SVC	0.975822	0.974790	0.975822	0.974790
4	ETC	0.974855	0.974576	0.974855	0.974576
5	LR	0.958414	0.970297	0.958414	0.970297
6	xgb	0.971954	0.943089	0.971954	0.943089
7	AdaBoost	0.960348	0.929204	0.960348	0.929204
8	GBDT	0.947776	0.920000	0.947776	0.920000
9	BgC	0.957447	0.867188	0.957447	0.867188
10	DT	0.927466	0.811881	0.927466	0.811881

In [57]:
```python
new_df_scaled = new_df.merge(temp_df,on='Algorithm')

```

In [58]:
```python
temp_df = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy_num_chars':accuracy_scores,'Precision_num_chars':precision_scores}).sort_values('Precision_num_chars',ascending=False)

```

In [59]:
```python
new_df_scaled.merge(temp_df,on='Algorithm')

```

Out[59]:

	Algorithm	Accuracy	Precision	Accuracy_scaling_x	Precision_scaling_x	Accuracy_scaling_y	Precision_scaling_y	Accuracy_num_chars	Precision_num_chars
0	KN	0.905222	1.000000	0.905222	1.000000	0.905222	1.000000	0.905222	1.000000
1	NB	0.970986	1.000000	0.970986	1.000000	0.970986	1.000000	0.970986	1.000000
2	RF	0.974855	0.982759	0.974855	0.982759	0.974855	0.982759	0.974855	0.982759
3	SVC	0.975822	0.974790	0.975822	0.974790	0.975822	0.974790	0.975822	0.974790
4	ETC	0.974855	0.974576	0.974855	0.974576	0.974855	0.974576	0.974855	0.974576
5	LR	0.958414	0.970297	0.958414	0.970297	0.958414	0.970297	0.958414	0.970297
6	xgb	0.971954	0.943089	0.971954	0.943089	0.971954	0.943089	0.971954	0.943089
7	AdaBoost	0.960348	0.929204	0.960348	0.929204	0.960348	0.929204	0.960348	0.929204
8	GBDT	0.947776	0.920000	0.947776	0.920000	0.947776	0.920000	0.947776	0.920000
9	BgC	0.957447	0.867188	0.957447	0.867188	0.957447	0.867188	0.957447	0.867188
10	DT	0.927466	0.811881	0.927466	0.811881	0.927466	0.811881	0.927466	0.811881

In [91]:
```python
# Voting Classifier

knc = KNeighborsClassifier()
mnb = MultinomialNB()
rfc = RandomForestClassifier(n_estimators=50, random_state=2)

from sklearn.ensemble import VotingClassifier
```

In [92]:
```python
voting = VotingClassifier(estimators=[('kn', knc), ('nb', mnb), ('rf', rfc)],voting='soft')

```

In [93]:
```python
voting.fit(X_train,y_train)

```

Out[93]:
```
VotingClassifier(estimators=[('kn', KNeighborsClassifier()),
                             ('nb', MultinomialNB()),
                             ('rf',
                              RandomForestClassifier(n_estimators=50,
                                                     random_state=2))],
                 voting='soft')
```

In [94]:
```python
y_pred = voting.predict(X_test)
print("Accuracy",accuracy_score(y_test,y_pred))
print("Precision",precision_score(y_test,y_pred))

```

```
Accuracy 0.9410058027079303
Precision 1.0
```

In [95]:
```python
# Applying stacking
estimators=[('kn', knc), ('nb', mnb), ('rf', rfc)]
final_estimator=RandomForestClassifier()
```

In [96]:
```python
from sklearn.ensemble import StackingClassifier

```

In [97]:
```python
clf = StackingClassifier(estimators=estimators, final_estimator=final_estimator)

```

In [98]:
```python
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 0.971953578336557
Precision 0.8865248226950354
```

In [102]:
```python
import pickle
pickle.dump(tfidf,open('vectorizer.pkl','wb'))
pickle.dump(rfc,open('model.pkl','wb'))
```

In [ ]:
```python

```

In [ ]:
```python

```