SMS Spam Detection

```
#library and packages
import numpy as np
import pandas as pd

#import dataset
dataset=pd.read_csv("SMSSpamCollection",sep='\t',names=['labels','message'])
```

```
dataset
```

	labels	message
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 ü 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

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   labels   5572 non-null   object
 1   message  5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
```

```
dataset.describe() #to get count of labels
```

	labels	message
count	5572	5572
unique	2	5169
top	ham	Sorry, I'll call later
freq	4825	30

```
dataset['labels'] = dataset['labels'].map({'ham': 0, 'spam': 1}) # mapping
with 0 and 1
```

```
dataset
```

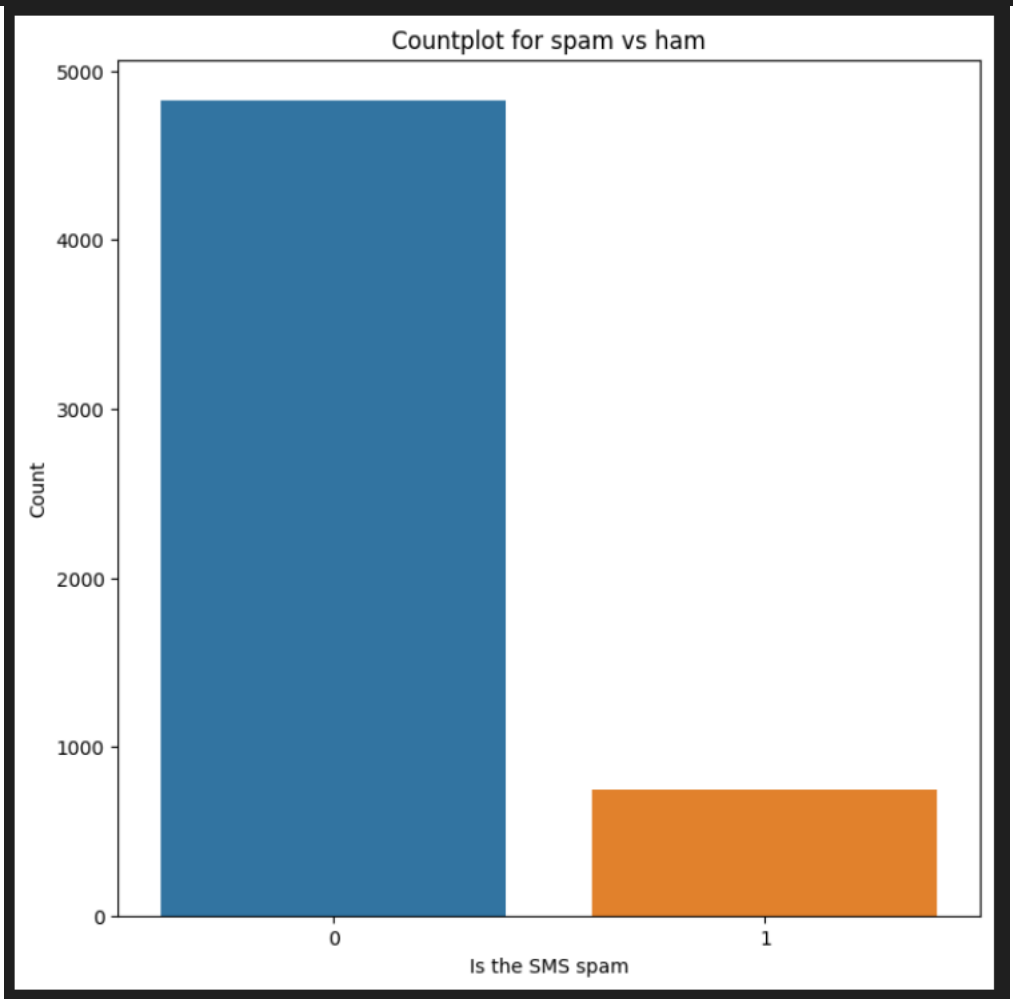
	labels	message
0	0	Go until jurong point, crazy.. Available only ...
1	0	Ok lar... Joking wif u oni...
2	1	Free entry in 2 a wkly comp to win FA Cup fina...
3	0	U dun say so early hor... U c already then say...
4	0	Nah I don't think he goes to usf, he lives aro...
...	...	...
5567	1	This is the 2nd time we have tried 2 contact u...
5568	0	Will ü b going to esplanade fr home?
5569	0	Pity, * was in mood for that. So...any other s...
5570	0	The guy did some bitching but I acted like i'd...
5571	0	Rofl. Its true to its name

5572 rows × 2 columns

```
#visulation
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

```
# count plot for spam vs ham to check imbalance


plt.figure(figsize=(8,8))
g = sns.countplot(x="labels", data=dataset)
```

```
plt.title('Countplot for spam vs ham')
plt.xlabel('Is the SMS spam')
plt.ylabel('Count')
```




```
#so we can see the data is imbalance
#handling imbalance dataset with oversampling
only_spam=dataset[dataset["labels"]==1]
```

```
only_spam
```

	labels	message
2	1	Free entry in 2 a wkly comp to win FA Cup fina...
5	1	FreeMsg Hey there darling it's been 3 week's n...
8	1	WINNER!! As a valued network customer you have...
9	1	Had your mobile 11 months or more? U R entitle...
11	1	SIX chances to win CASH! From 100 to 20,000 po...
...	...	...
5537	1	Want explicit SEX in 30 secs? Ring 02073162414...
5540	1	ASKED 3MOBILE IF 0870 CHATLINES INCLU IN FREE ...
5547	1	Had your contract mobile 11 Mnths? Latest Moto...
5566	1	REMINDER FROM O2: To get 2.50 pounds free call...
5567	1	This is the 2nd time we have tried 2 contact u...

747 rows × 2 columns

```
len(dataset)-len(only_spam)
```

```
#so we need to replicate spam 6-7 times to make dataset imbaalnce
count=int((dataset.shape[0]-only_spam.shape[0])/only_spam.shape[0])
```

```
count
```

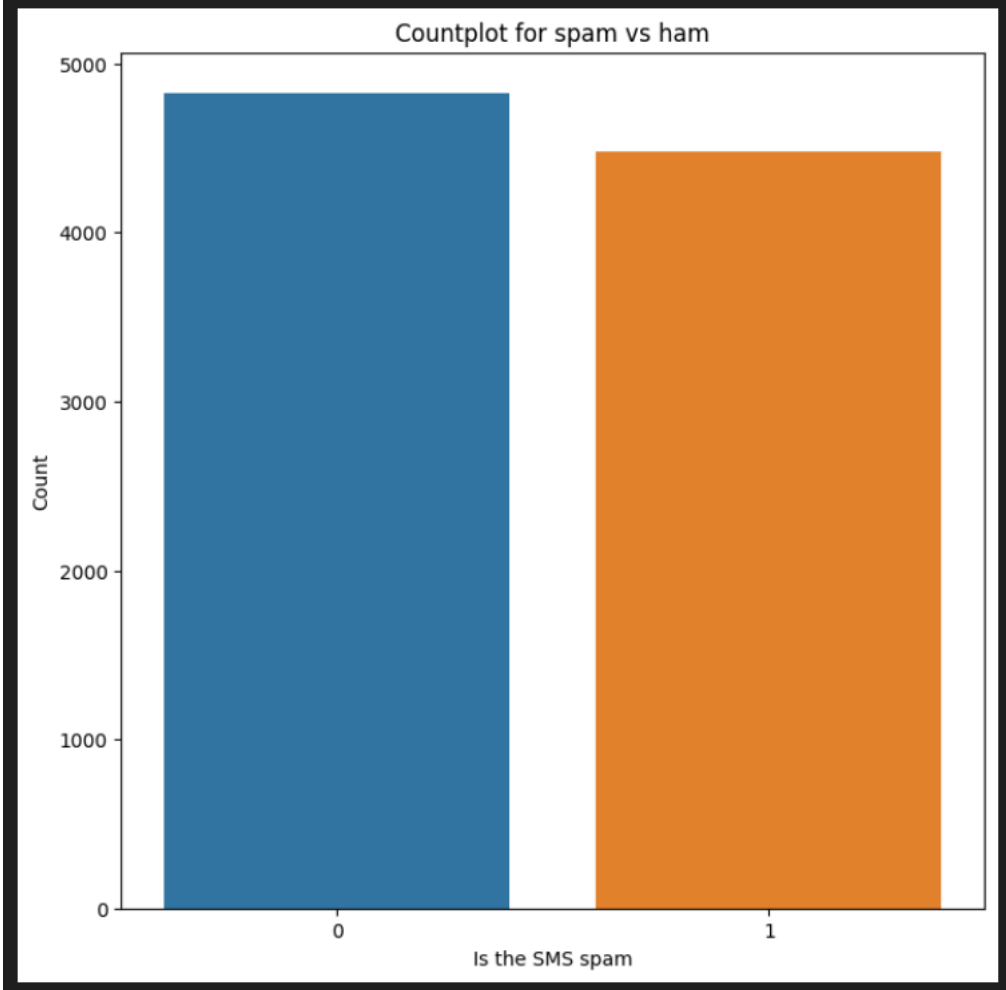
```
6
```

```
for i in range(0,count-1):
    dataset=pd.concat([dataset,only_spam])
dataset.shape
```

```
(9307, 2)
```

```
# count plot for spam vs ham to check imbalance


plt.figure(figsize=(8,8))
g = sns.countplot(x="labels", data=dataset)
plt.title('Countplot for spam vs ham')
plt.xlabel('Is the SMS spam')
plt.ylabel('Count')
```

```
Text(0, 0.5, 'Count')
```



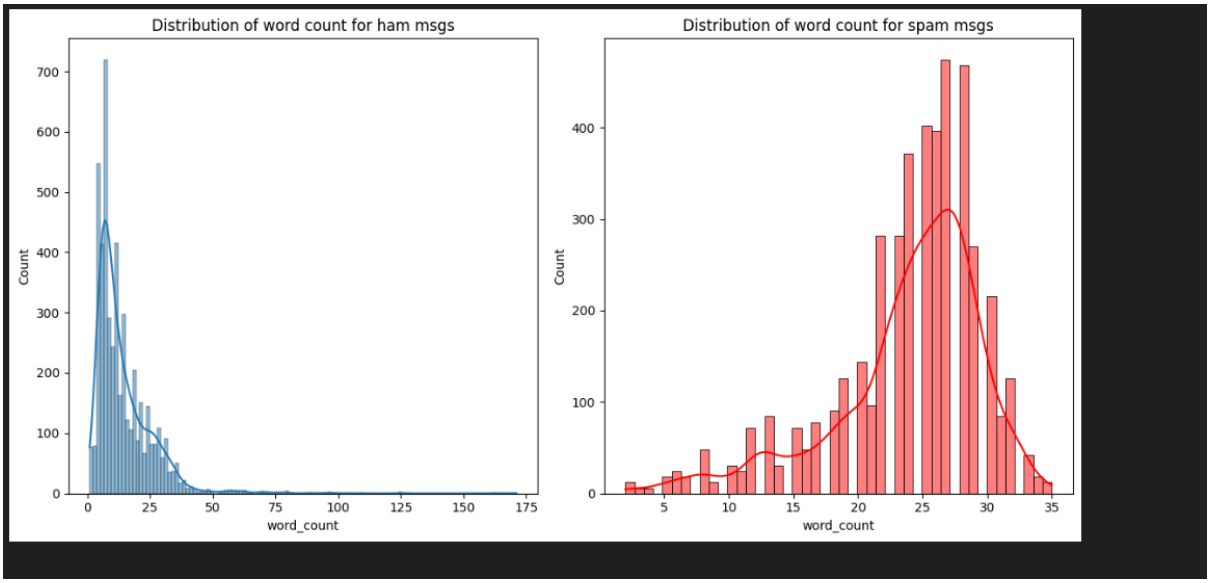

```
# creating new features
dataset['word_count']=dataset['message'].apply(lambda x: len(x.split()))
```

```
dataset
```

	labels	message	word_count
0	0	Go until jurong point, crazy.. Available only ...	20
1	0	Ok lar... Joking wif u oni...	6
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	28
3	0	U dun say so early hor... U c already then say...	11
4	0	Nah I don't think he goes to usf, he lives aro...	13
...	...	...	...
5537	1	Want explicit SEX in 30 secs? Ring 02073162414...	16
5540	1	ASKED 3MOBILE IF 0870 CHATLINES INCLU IN FREE ...	33
5547	1	Had your contract mobile 11 Mnths? Latest Moto...	28
5566	1	REMINDER FROM O2: To get 2.50 pounds free call...	28
5567	1	This is the 2nd time we have tried 2 contact u...	30

9307 rows × 3 columns

```
plt.figure(figsize=(12,6))

#(1,1)
plt.subplot(1,2,1)
g=sns.histplot(dataset[dataset["labels"]==0].word_count,kde=True)
p=plt.title('Distribution of word count for ham msgs')

#(1,2)
plt.subplot(1,2,2)
g=sns.histplot(dataset[dataset["labels"]==1].word_count,color="red",kde=True)
p=plt.title('Distribution of word count for spam msgs')


plt.tight_layout() ## padding in between
plt.show()
```




```
#createing new feature of containing_currency_symbol
def currency_present(data):
    currency_symbols=['¢','$','€','£','₹']
    for i in currency_symbols:
        if i in data:
            return 1
    return 0
```

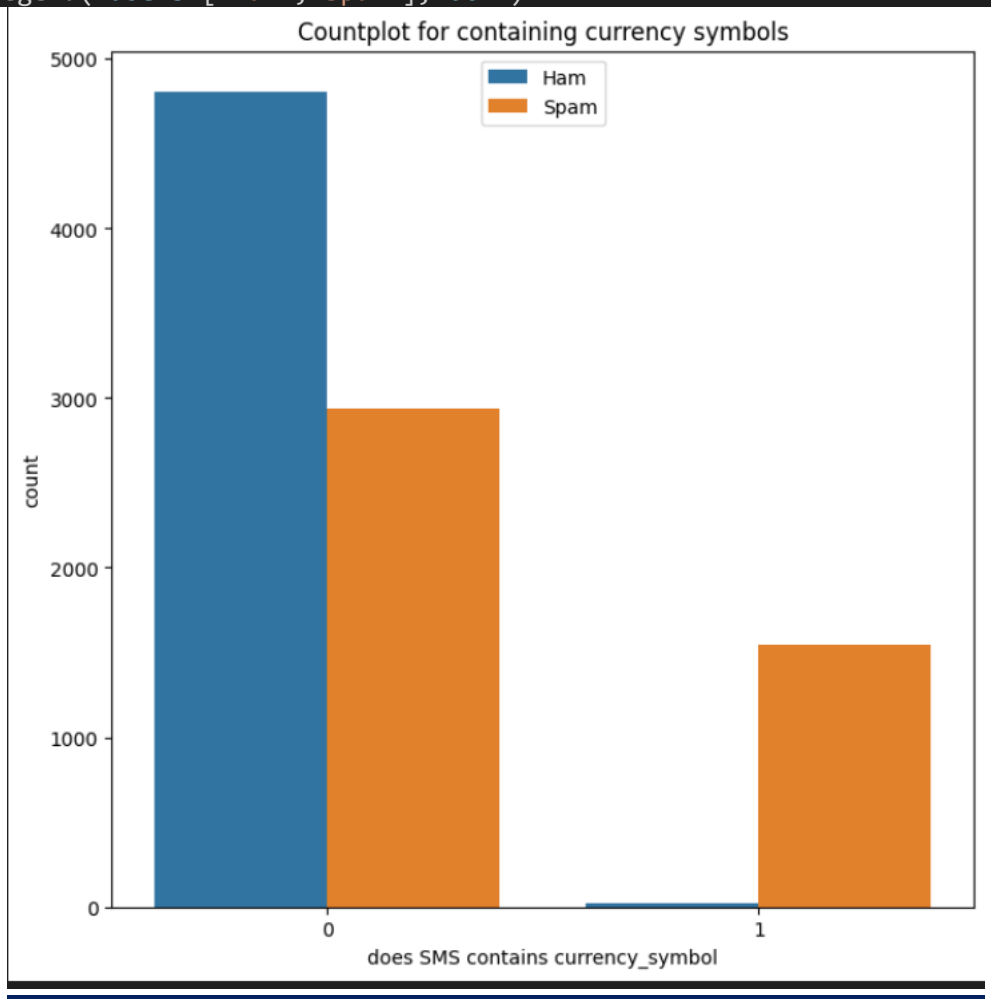
```
dataset["contains_currency_symbols"]=dataset["message"].apply(currency_present
)
```

```
dataset
```

	labels	message	word_count	contains_currency_symbols
0	0	Go until jurong point, crazy.. Available only ...	20	0
1	0	Ok lar... Joking wif u oni...	6	0
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	28	0
3	0	U dun say so early hor... U c already then say...	11	0
4	0	Nah I don't think he goes to usf, he lives aro...	13	0
...	...	...	...	...
5537	1	Want explicit SEX in 30 secs? Ring 02073162414...	16	0
5540	1	ASKED 3MOBILE IF 0870 CHATLINES INCLU IN FREE ...	33	1
5547	1	Had your contract mobile 11 Mnths? Latest Moto...	28	0
5566	1	REMINDER FROM O2: To get 2.50 pounds free call...	28	0
5567	1	This is the 2nd time we have tried 2 contact u...	30	1

9307 rows × 4 columns

```
#countplot for contains_currency_symbol
plt.figure(figsize=(8,8))
g=sns.countplot(x='contains_currency_symbols',data=dataset,hue="labels")
p=plt.title('Countplot for containing currency symbols')
p=plt.xlabel('does SMS contains currency_symbol')
p=plt.ylabel('count')
p=plt.legend(labels=["Ham","Spam"],loc=9)
```




```
#creating new feature of containing number
#ord function to convert into ascii and then again to normal
def number(data):
    for i in data:
        if ord(i) >= 48 and ord(i) <= 57:
            return 1
    return 0
```

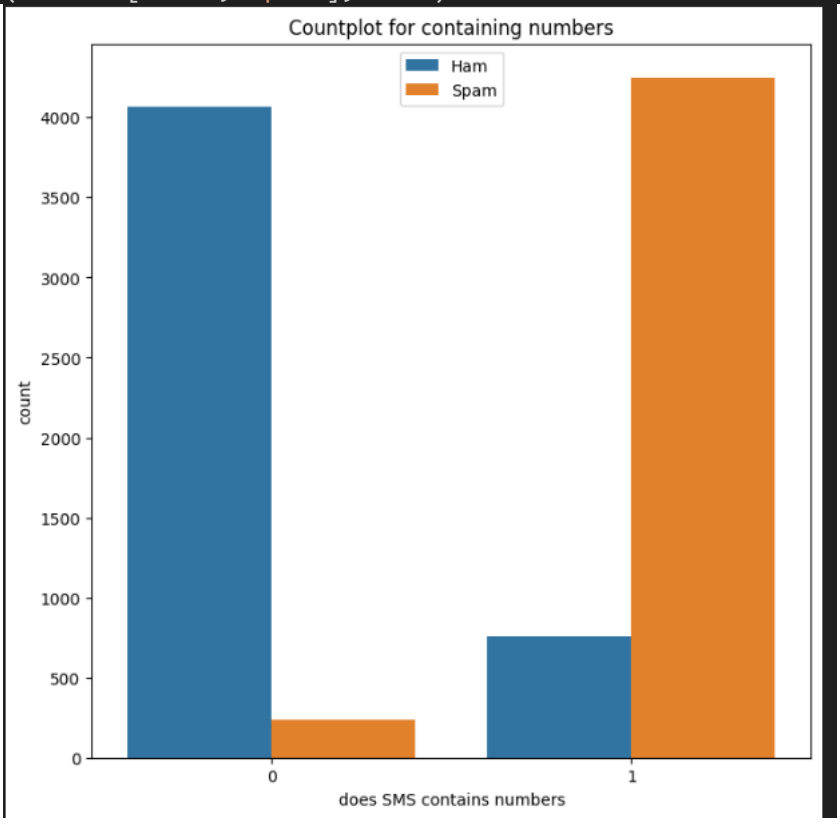
```
dataset["contains_number"]=dataset['message'].apply(number)
```

```
dataset
```

	labels	message	word_count	contains_currency_symbols	contains_number
0	0	Go until jurong point, crazy.. Available only ...	20	0	0
1	0	Ok lar... Joking wif u oni...	6	0	0
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	28	0	1
3	0	U dun say so early hor... U c already then say...	11	0	0
4	0	Nah I don't think he goes to usf, he lives aro...	13	0	0
...	...	...	...	...	...
5537	1	Want explicit SEX in 30 secs? Ring 02073162414...	16	0	1
5540	1	ASKED 3MOBILE IF 0870 CHATLINES INCLU IN FREE ...	33	1	1
5547	1	Had your contract mobile 11 Mnths? Latest Moto...	28	0	1
5566	1	REMINDER FROM O2: To get 2.50 pounds free call...	28	0	1
5567	1	This is the 2nd time we have tried 2 contact u...	30	1	1

9307 rows × 5 columns

```
#countplot for how many msgs have numbers
plt.figure(figsize=(8,8))
g=sns.countplot(x='contains_number',data=dataset,hue="labels")
p=plt.title('Countplot for containing numbers')
p=plt.xlabel('does SMS contains numbers')
p=plt.ylabel('count')
p=plt.legend(labels=["Ham","Spam"],loc=9)
```




```
#data cleaning because unstructured data
import nltk
import re
nltk.download('stopwords')
nltk.download('wordnet')
```

```
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
```

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

```
corpus = []
wnl = WordNetLemmatizer()

for sms in list(dataset.message):
    message = re.sub(pattern='[^a-zA-Z]', repl=' ', string=sms)  # filtering
out special characters and numbers
    message = message.lower()
    words = message.split()  # tokenizing (added parentheses after split)
    filtered_words = [word for word in words if word not in
set(stopwords.words('english'))]
    lemm_words = [wnl.lemmatize(word) for word in filtered_words]
    message = ' '.join(lemm_words)

    corpus.append(message)
```

```
corpus
```

```
['go jurong point crazy available bugis n great world la e buffet cine got amore wat',
 'ok lar joking wif u oni',
 'free entry wkly comp win fa cup final tkts st may text fa receive entry question std txt rate c apply',
 'u dun say early hor u c already say',
 'nah think go usf life around though',
 'freemsg hey darling week word back like fun still tb ok xxx std chgs send rcv',
 'even brother like speak treat like aid patent',
 'per request melle melle oru minnaminunginte nurungu vettam set callertune caller press copy friend callertune',
 'winner valued network customer selected receivea prize reward claim call claim code kl valid hour',
 'mobile month u r entitled update latest colour mobile camera free call mobile update co free',
 'gonna home soon want talk stuff anymore tonight k cried enough today',
 'six chance win cash pound txt csh send cost p day day tsandcs apply reply hl info',
 'urgent week free membership prize jackpot txt word claim c www dbuk net lccltd pobox ldnw rw',
 'searching right word thank breather promise wont take help granted fulfil promise wonderful blessing time',
 'date sunday',
 'xxxmobilemovieclub use credit click wap link next txt message click http wap xxxmobilemovieclub com n qjkgighjjgcbl',
 'oh k watching',
 'eh u remember spell name yes v naughty make v wet',
 'fine way u feel way gota b',
 'england v macedonia dont miss goal team news txt ur national team eg england try wale scotland txt poboxox w wq',
 'seriously spell name',
 'going try month ha ha joking',
 'pay first lar da stock comin',
 'aft finish lunch go str lor ard smth lor u finish ur lunch already',
 'ffffffffff alright way meet sooner',
...
 'change e one next escalator',
 'yetunde class run water make ok pls',
 'lot happened feel quiet beth aunt charlie working lot helen mo',
 'wait bus stop aft ur lect lar dun c go get car come back n pick',
 ...]
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```
#creating the bag of words model
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer(max_features=500)
vectors = tfidf.fit_transform(corpus).toarray()
feature_names = tfidf.get_feature_names()
```

```
#seprating dependent and independent
x=pd.DataFrame(vectors,columns=feature_names)
y=dataset['labels']
```

```
#cross validation report
#creating cnfusion matrix
from sklearn.model_selection import cross_val_score,train_test_split
from sklearn.metrics import classification_report,confusion_matrix
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,
random_state=42)
```

```
x_test
```

	ac	access	account	address	admirer	age	already	also	always	amp	...	xxx	ya	yeah	year	yes	yesterday	yet	yo	yr	yup
1155	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.72412	0.0	0.0	0.0	0.0
1790	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.557154	0.0	0.0	0.00000	0.0	0.0	0.0	0.0
3003	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.00000	0.0	0.0	0.0	0.0
6489	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.00000	0.0	0.0	0.0	0.0
592	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.00000	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4147	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.00000	0.0	0.0	0.0	0.0
274	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.00000	0.0	0.0	0.0	0.0
1345	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.00000	0.0	0.0	0.0	0.0
8891	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.00000	0.0	0.0	0.0	0.0
4031	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.000000	0.0	0.0	0.00000	0.0	0.0	0.0	0.0

1862 rows × 500 columns

```
#trining using naive bayes
from sklearn.naive_bayes import MultinomialNB
mnb=MultinomialNB()
cv=cross_val_score(mnb,x,y,scoring= 'f1',cv=10)
print(round(cv.mean(),3))
print(round(cv.std(),3))
```

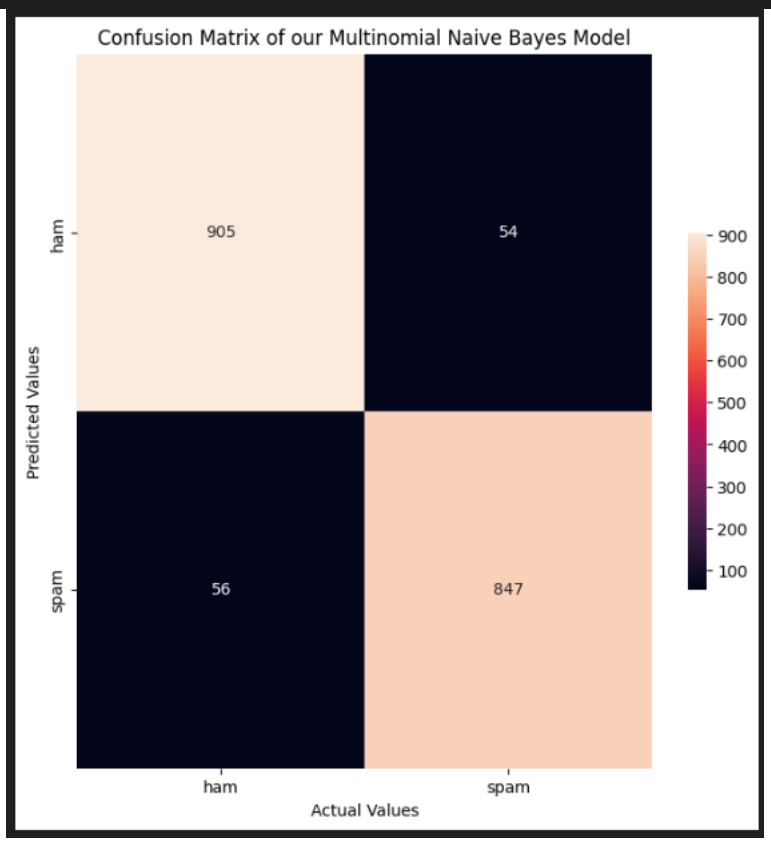
```
0.943
0.004
```

```
mnb.fit(x_train,y_train)
y_pred=mnb.predict(x_test)
```

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.94      0.94      0.94       959
           1       0.94      0.94      0.94       903

    accuracy                           0.94      1862
   macro avg       0.94      0.94      0.94      1862
weighted avg       0.94      0.94      0.94      1862
```

```
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 8))
axis_labels = ['ham', 'spam']
g = sns.heatmap(data=cm, xticklabels=axis_labels, yticklabels=axis_labels,
annot=True, fmt='g', cbar_kws={"shrink": 0.5})
plt.title("Confusion Matrix of our Multinomial Naive Bayes Model")
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.show()
```



```
#using decision tree
from sklearn.tree import DecisionTreeClassifier
dt=DecisionTreeClassifier()
cv1=cross_val_score(dt,x,y,scoring='f1',cv=10)
print(round(cv1.mean(),3))
print(round(cv1.std(),3))
```

```
0.98
0.004
```

```
dt.fit(x_train,y_train)
y_pred1=dt.predict(x_test)
```

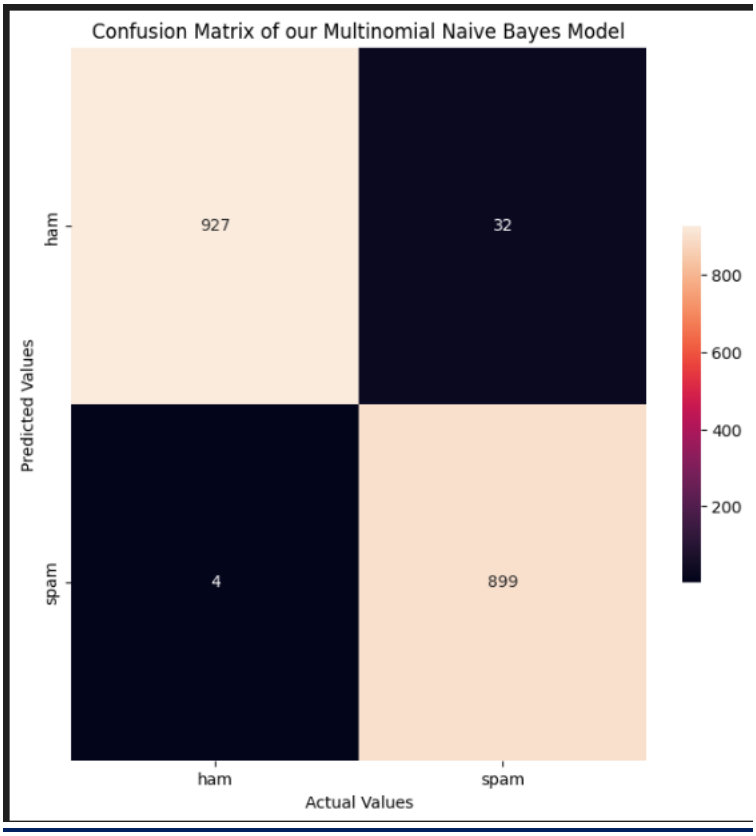
```
print(classification_report(y_test,y_pred1))
```

```
              precision    recall  f1-score   support

           0       1.00      0.97      0.98       959
           1       0.97      1.00      0.98       903

    accuracy                           0.98      1862
   macro avg       0.98      0.98      0.98      1862
weighted avg       0.98      0.98      0.98      1862
```

```
cm = confusion_matrix(y_test, y_pred1)
cm
```

```
array([[927,  32],
       [  4, 899]], dtype=int64)
```

```
plt.figure(figsize=(8, 8))
axis_labels = ['ham', 'spam']
g = sns.heatmap(data=cm, xticklabels=axis_labels, yticklabels=axis_labels,
annot=True, fmt='g', cbar_kws={"shrink": 0.5})
plt.title("Confusion Matrix of our Multinomial Naive Bayes Model")
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.show()
```




```
def predict_spam(sms):
    message = re.sub(pattern='[^a-zA-Z]', repl=' ', string=sms)  # filtering
out special characters and numbers
    message = message.lower()
    words = message.split()  # tokenizing (added parentheses after split)
    filtered_words = [word for word in words if word not in
set(stopwords.words('english'))]
    lemm_words = [wnl.lemmatize(word) for word in filtered_words]
    message = ' '.join(lemm_words)
    temp=tfidf.transform([message]).toarray()
    return dt.predict(temp)
```

```
#prediction 1
sample_message=""

if predict_spam(sample_message):
    print('this is a spam message')
else:
    print('this is not a spam message')
```