In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

import nltk
import re
#nltk.download ('wordnet')
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

In [2]:
```
df=pd.read_csv('D:\DP Neurotech internship\spam sms collection', sep='
\t', names=['type', 'message'])
```

In [3]:
```
df.head()
```

Out[3]:

	type	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...

In [4]:
```
df.shape
```

Out[4]:
```
(5572, 2)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
type       5572 non-null object
message    5572 non-null object
dtypes: object(2)
memory usage: 87.2+ KB
```

In [6]:
```
df.describe()
```

Out[6]:

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

In [7]:
```
df['type'] = df['type'].map({'ham': 0, 'spam': 1})
```

In [8]:
```
df
```

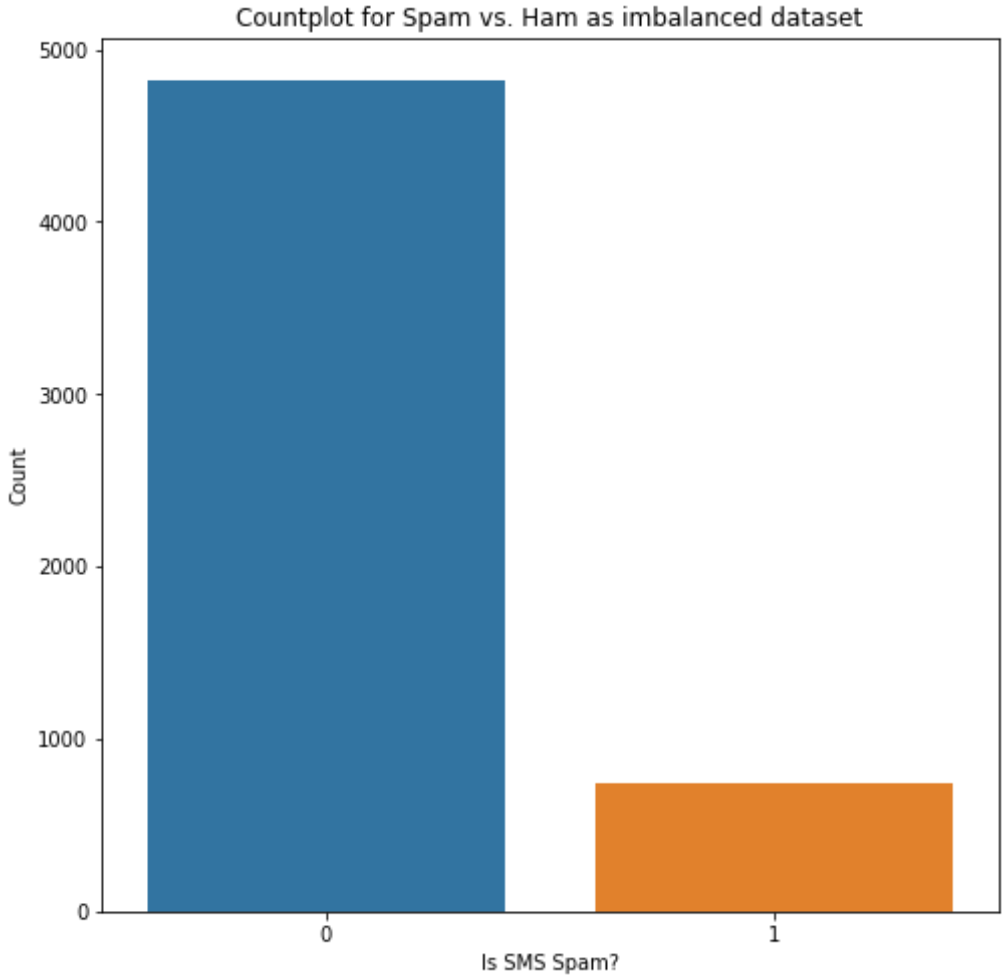
Out[8]:

	type	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...

	type	message
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

In [9]:
```
plt.figure(figsize=(8,8))
g = sns.countplot(x='type', data=df)
p = plt.title('Countplot for Spam vs. Ham as imbalanced dataset')
p = plt.xlabel('Is SMS Spam?')
p = plt.ylabel('Count')
```



In [10]:
```
print("Ham Count :",df['type'].value_counts()[0])
print("spam count :",df['type'].value_counts()[1])
```

```
Ham Count : 4825
spam count : 747
```

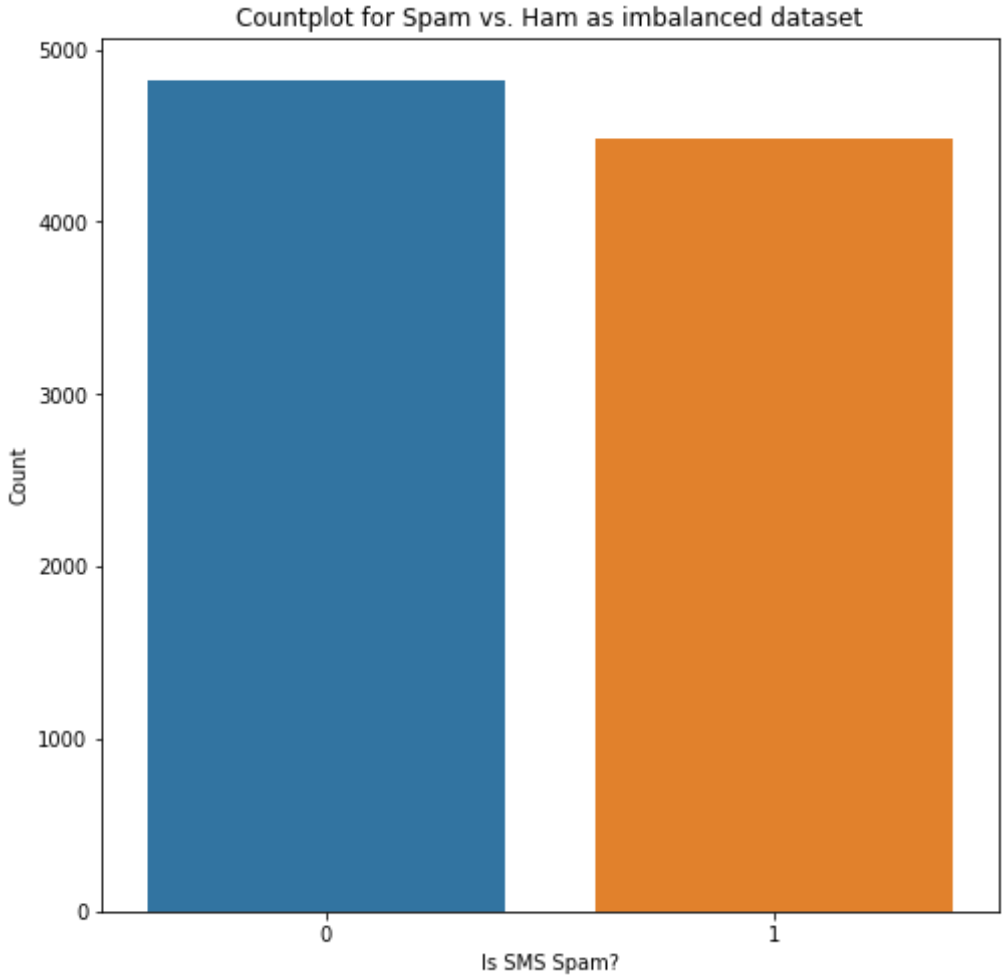
In [11]:
```
only_spam = df[df['type']==1]
count = int((df.shape[0]-only_spam.shape[0])/only_spam.shape[0])
for i in range(0, count-1):
```

```
    df = pd.concat([df,only_spam])

df.shape
```

Out[11]: (9307, 2)

In [12]:
```
plt.figure(figsize=(8,8))
g = sns.countplot(x='type', data=df)
p = plt.title('Countplot for Spam vs. Ham as imbalanced dataset')
p = plt.xlabel('Is SMS Spam?')
p = plt.ylabel('Count')
```



In [13]:
```
#creating new feature word_count
df['word_count'] = df['message'].apply(lambda x: len(x.split()))
```

In [14]:
```
df.head()
```

Out[14]:

	type	message	word_count
0	0	Go until jurong point, crazy.. Available only ...	20

	type	message	word_count
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

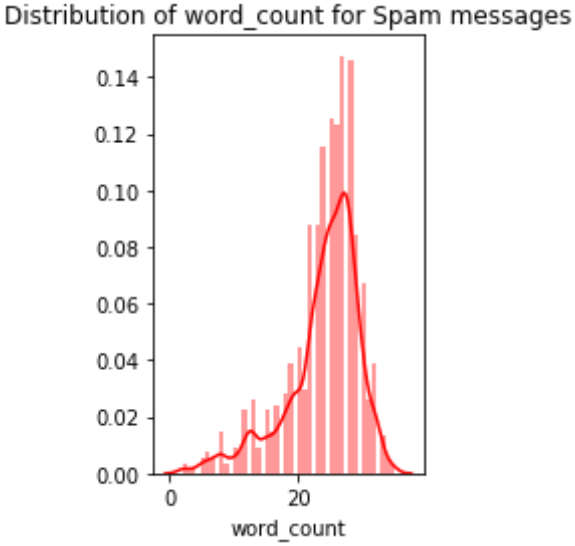
In [15]:
```
plt.figure(figsize=(12, 6))
```

Out[15]:
```
<Figure size 864x432 with 0 Axes>
```


```
<Figure size 864x432 with 0 Axes>
```


In [16]:
```
plt.subplot(1,2,2)
g = sns.distplot(a=df[df['type']==1].word_count, color='red')
p = plt.title('Distribution of word_count for Spam messages')

plt.tight_layout()
plt.show()
```



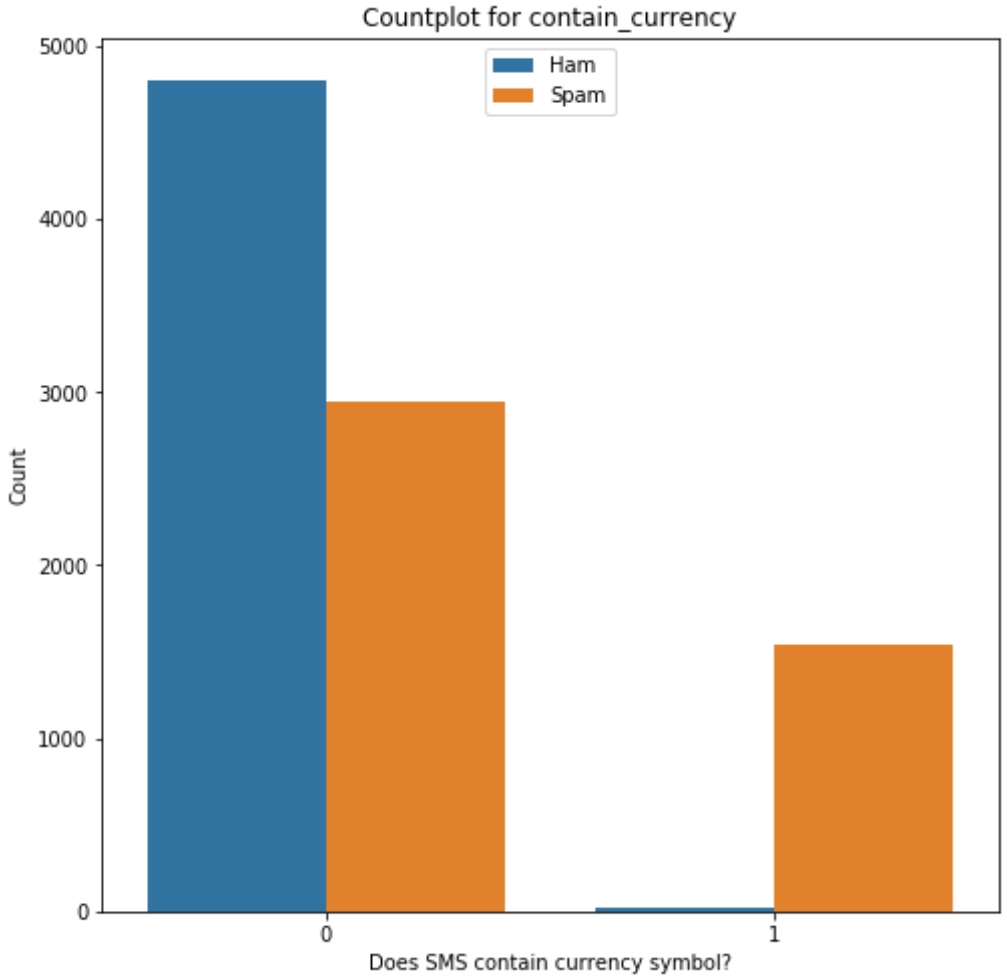
In [17]:
```
def currency(x):
  currency_symbols = ['€', '$', '¥', '₹', '£']
  for i in currency_symbols:
    if i in x:
      return 1
  return 0
df['contains_currency_symbol'] = df['message'].apply(currency)
```

In [18]:
```
df.head()
```

Out[18]:

	type	message	word_count	contains_currency_symbol
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

In [19]:
```
plt.figure(figsize=(8,8))
g = sns.countplot(x='contains_currency_symbol', data=df, hue='type')
p = plt.title('Countplot for contain_currency')
p = plt.xlabel('Does SMS contain currency symbol?')
p = plt.ylabel('Count')
p = plt.legend(labels=['Ham', 'Spam'], loc=9)
```



In [20]:
```
#cleaning the messages
corpus = []
wnl = WordNetLemmatizer()

for sms_string in list(df.message):

  #cleaning special character from the sms
  message = re.sub(pattern='[^a-zA-Z]', repl=' ', string=sms_string)
```

```
  #converting the entire sms into lower case
  message = message.lower()

  #Tokenizing the sms by words
  words = message.split()

  #Removing the stop words
  filtered_words = [word for word in words if word not in set (stopword
s.words('english'))]

  #lemmatizing the words
  lemmatized_words = [wnl.lemmatize(word) for word in filtered_words]

  #joining the lemmatized words
  message = ' '.join(lemmatized_words)

  #building a corpus of messages
  corpus.append(message)
```

In [21]:
```
corpus[0:5]
```

Out[21]:
```
['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 ent
ry question std txt rate c apply',
 'u dun say early hor u c already say',
 'nah think go usf life around though']
```

In [22]:
```
tfidf = TfidfVectorizer(max_features=500)
vectors = tfidf.fit_transform(corpus).toarray()
feature_names = tfidf.get_feature_names()
```

In [23]:
```
#Extracting independent and dependent variables from the dataset
x = pd.DataFrame(vectors, columns=feature_names)
y = df['type']
```

In [24]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3
```

```
, random_state=42)
```

In [25]:
```
mnb = MultinomialNB()
cv = cross_val_score(mnb, x, y, scoring='f1', cv=10)
print('--- Average F1-Score for MNB model: {}---'.format(round(cv.mean
(), 3)))
print('Standard Deviation: {}'.format(round(cv.std(), 3)))
```

```
--- Average F1-Score for MNB model: 0.943---
Standard Deviation: 0.004
```

In [28]:
```
mnb.fit(x_train, y_train)
y_pred = mnb.predict(x_test)

print('--- Classification report for MNB model ---')
print(classification_report(y_test, y_pred))
```

```
--- Classification report for MNB model ---
              precision    recall  f1-score   support

           0       0.95      0.94      0.95      1457
           1       0.94      0.94      0.94      1336

    accuracy                           0.94      2793
   macro avg       0.94      0.94      0.94      2793
weighted avg       0.94      0.94      0.94      2793
```