```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.naive_bayes import GaussianNB,MultinomialNB, BernoulliNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import BaggingClassifier, RandomForestClassifier, ExtraTreesClassifier
from sklearn.ensemble import AdaBoostClassifier, GradientBoostingClassifier
from xgboost import XGBClassifier, XGBRFClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,precision_score
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from wordcloud import WordCloud
from collections import  Counter
import nltk
import pickle
import string
```

```
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
df = pd.read_csv("/content/spam.csv", encoding="latin-1")
df.sample(1)
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
468	ham	Yup, no need. I'll jus wait 4 e rain 2	NaN	NaN	NaN

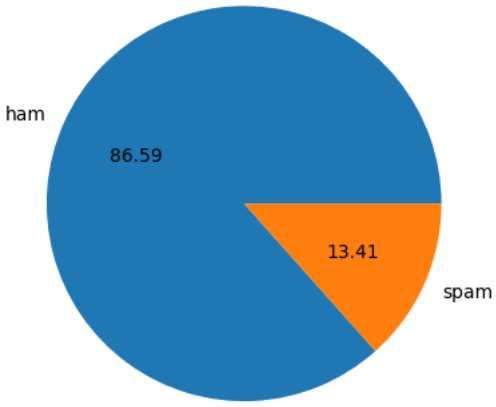
```
df.rename(columns={"v1": "output","v2": "input"}, inplace = True)
df.sample(1)
```

	output	input	Unnamed: 2	Unnamed: 3	Unnamed: 4
2427	ham	Do you think i can move <#> in	NaN	NaN	NaN

```
le = LabelEncoder()
df["output"] = le.fit_transform(df["output"])
df.sample(1)
```

	output	input	Unnamed: 2	Unnamed: 3	Unnamed: 4
2324	0	Ok lor.	NaN	NaN	NaN

```
plt.pie(df["output"].value_counts(),autopct = "%.2f", labels=['ham','spam'])
plt.show()
```



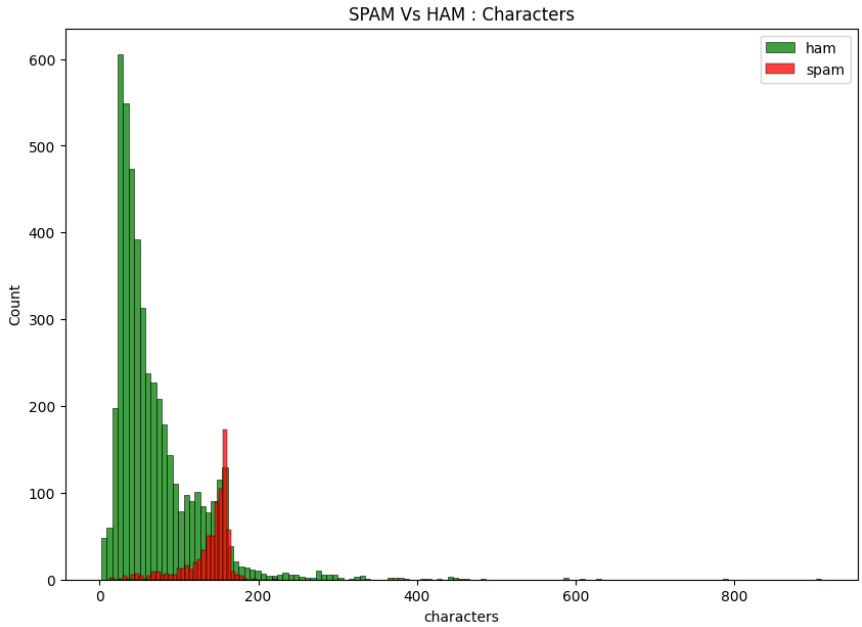

```
df["characters"] = df["input"].apply(len)
df["word"] = df["input"].apply(lambda x:len( nltk.word_tokenize(x)))
df["sentence"] = df["input"].apply(lambda x:len(nltk.sent_tokenize(x)))
df.head(1)
```

output	input	Unnamed: 2	Unnamed: 3	Unnamed: 4	characters	word	sentence
	Go until						

```
df[["characters","word", "sentence"]].describe()
```

	characters	word	sentence
count	5572.000000	5572.000000	5572.000000
mean	80.118808	18.699390	1.996411
std	59.690841	13.741932	1.520159
min	2.000000	1.000000	1.000000
25%	36.000000	9.000000	1.000000
50%	61.000000	15.000000	1.500000
75%	121.000000	27.000000	2.000000
max	910.000000	220.000000	38.000000

```
plt.figure(figsize=(10,7))
sns.histplot(df[df["output"]==0]["characters"],label= "ham",color="green")
sns.histplot(df[df["output"]==1]["characters"],label= "spam",color = "red")
plt.title("SPAM Vs HAM : Characters")
plt.legend()
plt.show()
```



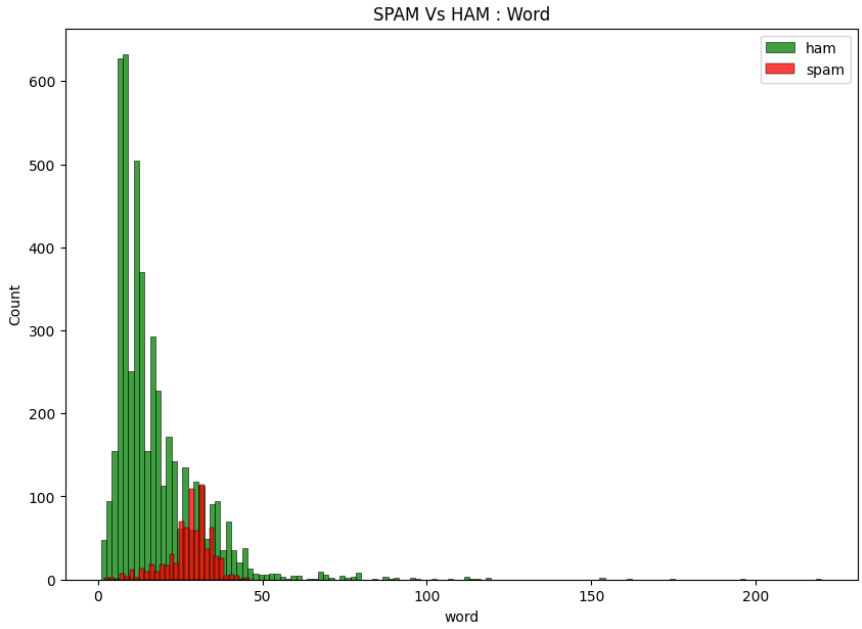

```
df[df["output"]==0][["characters","word", "sentence"]].describe()
```

	characters	word	sentence
count	4825.000000	4825.000000	4825.000000
mean	71.023627	17.276269	1.837720
std	58.016023	13.988585	1.454388
min	2.000000	1.000000	1.000000
25%	33.000000	8.000000	1.000000
50%	52.000000	13.000000	1.000000
75%	92.000000	22.000000	2.000000
max	910.000000	220.000000	38.000000

```
df[df["output"] ==1][["characters","word", "sentence"]].describe()
```

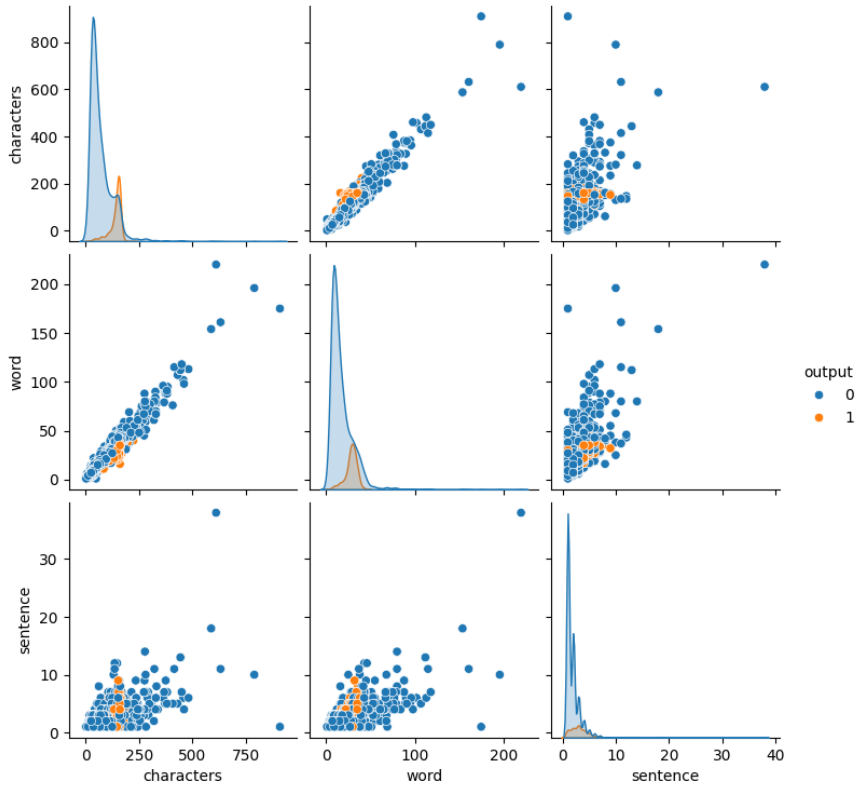
	characters	word	sentence
count	747.000000	747.000000	747.000000
mean	138.866131	27.891566	3.021419
std	29.183082	6.867007	1.537580
min	13.000000	2.000000	1.000000
25%	132.500000	25.000000	2.000000
50%	149.000000	29.000000	3.000000
75%	157.000000	32.000000	4.000000
max	224.000000	46.000000	9.000000

```
plt.figure(figsize=(10,7))
sns.histplot(df[df["output"]==0]["word"],label= "ham",color="green")
sns.histplot(df[df["output"]==1]["word"],label= "spam",color = "red")
plt.title("SPAM Vs HAM : Word")
plt.legend()
plt.show()
```




```
sns.pairplot(df,hue="output")
```

```
<seaborn.axisgrid.PairGrid at 0x7b30ab800700>
```




```
df.corr()
```

	output	characters	word	sentence
output	1.000000	0.387285	0.263221	0.265332
characters	0.387285	1.000000	0.966310	0.631881
word	0.263221	0.966310	1.000000	0.685165
sentence	0.265332	0.631881	0.685165	1.000000

```
sns.heatmap(df.corr(),annot=True,cmap="viridis")
```

```
<Axes: >
```