MAil Spam And Ham Classification

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

IMPORT DATASET

```
spam_df = pd.read_csv("/content/emails.csv")
```

```
spam_df
```

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	military	allowing	ff
0	Email 1	0	0	1	0	0	0	2	0	0	...	0	0	0	0	0	0	0	0
1	Email 2	8	13	24	6	6	2	102	1	27	...	0	0	0	0	0	0	0	1
2	Email 3	0	0	1	0	0	0	8	0	0	...	0	0	0	0	0	0	0	0
3	Email 4	0	5	22	0	5	1	51	2	10	...	0	0	0	0	0	0	0	0
4	Email 5	7	6	17	1	5	2	57	0	9	...	0	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5167	Email 5168	2	2	2	3	0	0	32	0	0	...	0	0	0	0	0	0	0	0
5168	Email 5169	35	27	11	2	6	5	151	4	3	...	0	0	0	0	0	0	0	1
5169	Email 5170	0	0	1	1	0	0	11	0	0	...	0	0	0	0	0	0	0	0
5170	Email 5171	2	7	1	0	2	1	28	2	0	...	0	0	0	0	0	0	0	1
5171	Email 5172	22	24	5	1	6	5	148	8	2	...	0	0	0	0	0	0	0	0

5172 rows × 3002 columns

```
spam_df.head(10)
```

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	military	allowing	ff	dry
0	Email 1	0	0	1	0	0	0	2	0	0	...	0	0	0	0	0	0	0	0	0
1	Email 2	8	13	24	6	6	2	102	1	27	...	0	0	0	0	0	0	0	1	0
2	Email 3	0	0	1	0	0	0	8	0	0	...	0	0	0	0	0	0	0	0	0
3	Email 4	0	5	22	0	5	1	51	2	10	...	0	0	0	0	0	0	0	0	0
4	Email 5	7	6	17	1	5	2	57	0	9	...	0	0	0	0	0	0	0	1	0
5	Email 6	4	5	1	4	2	3	45	1	0	...	0	0	0	0	0	0	0	0	0
6	Email 7	5	3	1	3	2	1	37	0	0	...	0	0	0	0	0	0	0	0	0
7	Email 8	0	2	2	3	1	2	21	6	0	...	0	0	0	0	0	0	0	1	0
8	Email 9	2	2	3	0	0	1	18	0	0	...	0	0	0	0	0	0	0	0	0
9	Email 10	4	4	35	0	1	0	49	1	16	...	0	0	0	0	0	0	0	0	0

10 rows × 3002 columns

```
spam_df.tail(10)
```

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	military	allowing	ff
5162	Email 5163	2	3	1	2	1	2	32	0	0	...	0	0	0	0	0	0	0	0
5163	Email 5164	0	0	1	0	0	0	1	0	0	...	0	0	0	0	0	0	0	0
5164	Email 5165	21	18	3	1	6	4	106	1	2	...	0	0	0	0	0	0	0	0
5165	Email 5166	1	0	1	0	3	1	12	1	0	...	0	0	0	1	0	0	0	0
5166	Email 5167	1	0	1	1	0	0	4	0	0	...	0	0	0	0	0	0	0	0
5167	Email 5168	2	2	2	3	0	0	32	0	0	...	0	0	0	0	0	0	0	0
5168	Email 5169	35	27	11	2	6	5	151	4	3	...	0	0	0	0	0	0	0	1
5169	Email 5170	0	0	1	1	0	0	11	0	0	...	0	0	0	0	0	0	0	0
5170	Email 5171	2	7	1	0	2	1	28	2	0	...	0	0	0	0	0	0	0	1
5171	Email 5172	22	24	5	1	6	5	148	8	2	...	0	0	0	0	0	0	0	0

10 rows × 3002 columns

```
spam_df.describe()
```

	the	to	ect	and	for	of	a	you	hou	
count	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	5172.00
mean	6.640565	6.188128	5.143852	3.075599	3.124710	2.627030	55.517401	2.466551	2.024362	10.60
std	11.745009	9.534576	14.101142	6.045970	4.680522	6.229845	87.574172	4.314444	6.967878	19.28
min	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00
25%	0.000000	1.000000	1.000000	0.000000	1.000000	0.000000	12.000000	0.000000	0.000000	1.00
50%	3.000000	3.000000	1.000000	1.000000	2.000000	1.000000	28.000000	1.000000	0.000000	5.00
75%	8.000000	7.000000	4.000000	3.000000	4.000000	2.000000	62.250000	3.000000	1.000000	12.00
max	210.000000	132.000000	344.000000	89.000000	47.000000	77.000000	1898.000000	70.000000	167.000000	223.00

8 rows × 3001 columns

```
spam_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5172 entries, 0 to 5171
Columns: 3002 entries, Email No. to Prediction
dtypes: int64(3001), object(1)
memory usage: 118.5+ MB
```

Visualize dataset

```
ham = spam_df[ spam_df['spam'] == 0]
```

```
ham
```

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	military	allowing	ff
0	Email 1	0	0	1	0	0	0	2	0	0	...	0	0	0	0	0	0	0	0
1	Email 2	8	13	24	6	6	2	102	1	27	...	0	0	0	0	0	0	0	1
2	Email 3	0	0	1	0	0	0	8	0	0	...	0	0	0	0	0	0	0	0
3	Email 4	0	5	22	0	5	1	51	2	10	...	0	0	0	0	0	0	0	0
4	Email 5	7	6	17	1	5	2	57	0	9	...	0	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5167	Email 5168	2	2	2	3	0	0	32	0	0	...	0	0	0	0	0	0	0	0
5168	Email 5169	35	27	11	2	6	5	151	4	3	...	0	0	0	0	0	0	0	1
5169	Email 5170	0	0	1	1	0	0	11	0	0	...	0	0	0	0	0	0	0	0
5170	Email 5171	2	7	1	0	2	1	28	2	0	...	0	0	0	0	0	0	0	1
5171	Email 5172	22	24	5	1	6	5	148	8	2	...	0	0	0	0	0	0	0	0

5104 rows × 3002 columns

```
spam = spam_df[spam_df['spam'] == 1 ]
```

```
spam
```

	Email No.	the	to	ect	and	for	of	a	you	hou	...	connevey	jay	valued	lay	infrastructure	military	allowing	
149	Email 150	5	11	1	2	2	5	49	11	2	...	0	0	0	0	0	0	0	
391	Email 392	8	14	3	7	4	6	291	6	0	...	0	0	0	0	0	0	0	1
528	Email 529	2	4	2	0	1	1	23	2	0	...	0	0	0	0	0	0	0	
706	Email 707	1	6	2	0	1	0	41	1	0	...	0	0	0	0	0	0	0	
708	Email 709	1	3	2	0	1	1	37	1	0	...	0	0	0	0	0	0	0	
746	Email 747	10	15	6	12	7	4	140	7	1	...	0	0	0	2	0	0	0	
809	Email 810	10	17	6	6	14	4	120	4	3	...	0	0	0	0	0	0	0	
1084	Email 1085	11	14	7	6	11	4	141	7	2	...	0	0	0	0	0	0	0	
1189	Email 1190	24	24	4	11	11	23	222	17	6	...	0	0	0	0	0	0	0	
1294	Email 1295	8	8	5	4	4	3	219	8	1	...	0	0	0	0	0	0	0	
1346	Email 1347	16	17	8	8	15	8	160	12	2	...	0	0	0	0	0	0	0	
1408	Email 1409	4	8	1	1	2	3	60	5	2	...	0	0	0	0	0	0	0	
1458	Email 1459	12	18	8	11	11	7	159	14	2	...	0	0	0	6	0	0	0	
1498	Email 1499	10	16	7	9	10	5	122	8	2	...	0	0	0	2	0	0	0	
1558	Email 1559	14	30	15	14	21	11	260	12	7	...	0	0	0	2	0	0	0	
1653	Email 1654	12	18	8	11	12	5	146	9	2	...	0	0	0	0	0	0	0	
1747	Email 1748	18	16	6	12	10	6	153	8	1	...	0	0	0	1	0	0	0	
1798	Email 1799	24	20	8	11	10	10	164	8	3	...	0	0	0	0	0	0	1	
1854	Email 1855	10	14	7	8	11	6	115	7	1	...	0	0	0	0	0	0	0	
1980	Email 1981	12	26	13	20	32	12	271	7	4	...	0	0	0	0	0	0	0	
1995	Email 1996	11	14	6	10	9	5	111	7	1	...	0	0	0	1	0	0	0	
2025	Email 2026	14	15	6	11	11	5	127	9	1	...	0	0	0	0	0	0	0	
2114	Email 2115	1	2	1	1	0	0	8	0	0	...	0	0	0	0	0	0	0	
2184	Email 2185	11	16	6	12	9	5	118	7	1	...	0	0	0	0	0	0	0	
2210	Email 2211	4	4	4	0	2	4	78	14	0	...	0	0	0	0	0	0	0	
2217	Email 2218	1	1	1	1	0	0	8	0	0	...	0	0	0	0	0	0	0	
2409	Email 2410	10	13	6	6	9	6	99	7	1	...	0	0	0	0	0	0	0	
2539	Email 2540	14	24	5	11	12	7	146	9	2	...	0	0	0	1	0	0	1	
	Email																		

```
print(" Spam percentage =", (len(spam)/len(spam_df))*100,'%')
```

```
Spam percentage = 1.1020881670533642 %
```

imbalance

```
spam_df.columns
```

```
Index(['Email No.', 'the', 'to', 'ect', 'and', 'for', 'of', 'a', 'you', 'hou',
       ...
       'connevey', 'jay', 'valued', 'lay', 'infrastructure', 'military',
       'allowing', 'ff', 'dry', 'Prediction'],
      dtype='object', length=3002)
```

```
spam_df['Prediction'].value_counts()
```

	count
Prediction	
0	3672
1	1500

dtype: int64

```
spam = spam_df[spam_df['Prediction'] == 1] # this is the ROws
spam_percentage = (len(spam)/len(spam_df)) * 100
print("Your spam percentage =", spam_percentage, "%")
```

```
Your spam percentage = 29.00232018561485 %
```

```
#ham percentage
ham = spam_df[spam_df['Prediction'] == 0 ] # this is column
ham_percentage = (len(ham)/len(spam_df)) * 100
print("Your Ham percentage =", ham_percentage, "%")
```

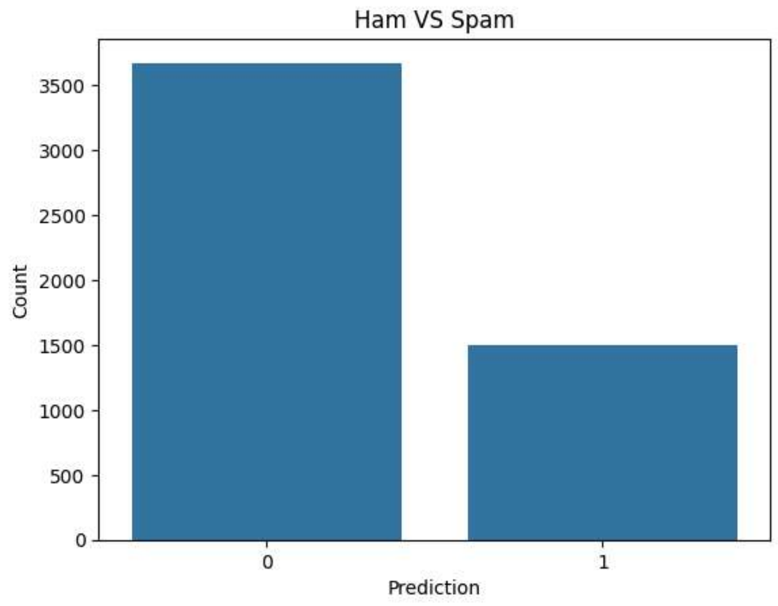
```
Your Ham percentage = 70.99767981438515 %
```

Spam and Ham pervcentage View

```
# sns.seaborn(spam_df['spam'])
sns.countplot(x='Prediction', data =  spam_df)
plt.xticks([0,1]), ['Ham', 'Spam']
plt.title("Ham VS Spam")
plt.ylabel("Count")
plt.show()
```

2746	Email 2747	10	8	7	6	3	2	91	18	2	...	0	0	0	0	0	0	0
3030	Email 3031	41	33	12	17	10	12	296	7	1	...	0	0	0	0	0	0	0
3312	Email 3313	12	20	6	6	13	5	160	7	2	...	0	0	0	1	0	0	0
3405	Email 3406	10	12	6	6	9	4	94	7	1	...	0	0	0	0	0	0	0
3418	Email 3419	10	13	6	6	10	4	94	7	2	...	0	0	0	0	0	0	0
3467	Email 3468	10	12	5	6	9	4	98	7	1	...	0	0	0	0	0	0	0
[illegible]	Email 3644	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
3844	Email 3845	11	17	6	8	10	4	113	8	2	...	0	0	0	0	0	0	0
[illegible]	Email 4120	[illegible]	15	6	11	13	5	148	8	1	...	0	0	0	0	0	0	0
4283	Email 4284	10	12	5	6	9	5	135	7	2	...	0	0	0	0	0	0	0
[illegible]	Email [illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
4611	Email 4612	17	43	14	15	13	11	222	9	3	...	0	0	0	0	0	0	0
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
4786	Email 4787	0	5	3	1	2	0	62	0	0	...	0	0	0	0	0	0	0
4881	Email 4882	11	16	6	10	13	7	129	8	2	...	0	0	0	2	0	0	0
4963	Email 4964	13	17	5	9	11	5	117	11	1	...	0	0	0	0	0	0	0
4978	Email 4979	36	46	7	41	26	22	345	19	1	...	0	0	0	1	0	0	0
4983	Email 4984	10	17	6	11	11	8	133	11	1	...	0	0	0	1	0	0	0
5059	Email 5060	11	11	5	6	10	5	112	7	1	...	0	0	0	0	0	0	0
5106	Email 5107	33	34	8	23	20	10	267	13	1	...	0	0	0	3	0	0	0

57 rows × 3002 columns



Count vecrorize

```
from sklearn.feature_extraction.text import CountVectorizer
sample_data = ['this is the first document ', " this is the second document ", "this is the third document "]
sample_vectorizer = CountVectorizer()
```

```
X = sample_vectorizer.fit_transform(sample_data)
```

```
print(sample_vectorizer.get_feature_names_out())
```

```
['document' 'first' 'is' 'second' 'the' 'third' 'this']
```

```
print(X.toarray())
```

```
[[1 1 1 0 1 0 1]
 [1 0 1 1 1 0 1]
 [1 0 1 0 1 1 1]]
```

COUNT VECTORIZATION

```
# view Text colum
spam_df.columns
```

```
Index(['Email No.', 'the', 'to', 'ect', 'and', 'for', 'of', 'a', 'you', 'hou',
       ...
       'connevey', 'jay', 'valued', 'lay', 'infrastructure', 'military',
       'allowing', 'ff', 'dry', 'Prediction'],
      dtype='object', length=3002)
```

```
from sklearn.feature_extraction.text import CountVectorizer
```

This is just for see my emil text but i already download a vectorize CSV file thats why my email have no text

```
# word_cols = spam_df.columns[1:-1]  ## exclude 'Email No.' and 'Prediction'

# spam_df['full_text'] = spam_df[word_cols].astype(str).agg(' '.join, axis=1)

# # See first few rows
# print(spam_df[['Email No.', 'full_text', 'Prediction']].head())
```

TRAIN THE MODEL

```
label = spam_df['spam'].values
```

```
label
```

```
array([0, 0, 0, ..., 0, 0, 0])
```

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix
```

```
X = spam_df.iloc[:, 1:-1]      # all word columns
y = spam_df['Prediction'].values
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

```
NB_classifier = MultinomialNB()
NB_classifier.fit(X_train, y_train)
```

```
▾ MultinomialNB
MultinomialNB()
```

```
y_pred = NB_classifier.predict(X_test)
print("Accuracy =", accuracy_score(y_test, y_pred))
print("Confusion Matrix and YN", confusion_matrix(y_test, y_pred))
```

```
Accuracy = 0.9545893719806763
Confusion Matrix and YN [[704  35]
 [ 12 284]]
```

```
from sklearn.metrics import accuracy_score
print(" my Model Accuracy:", accuracy_score(y_test, y_pred))
```

```
 my Model Accuracy: 0.9545893719806763
```

EVALUATING THE MODEL

```
from sklearn.metrics import classification_report, confusion_matrix
```

```
Y_predict_train = NB_classifier.predict(X_train)
```

```
Y_predict_train
```

```
array([1, 0, 1, ..., 0, 1, 1])
```

```
cm = confusion_matrix(y_train, Y_predict_train)
sns.heatmap(cm, annot= True)
```