Build a spam filter using Python and the multinomial Naive Bayes algorithm.

Check Spam or Ham? Email Classifier Using Python using MultinomialNB.

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

```
df= pd.read_csv("/content/spam.csv")
df.head()
```

	Category	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...

```
df.shape
```

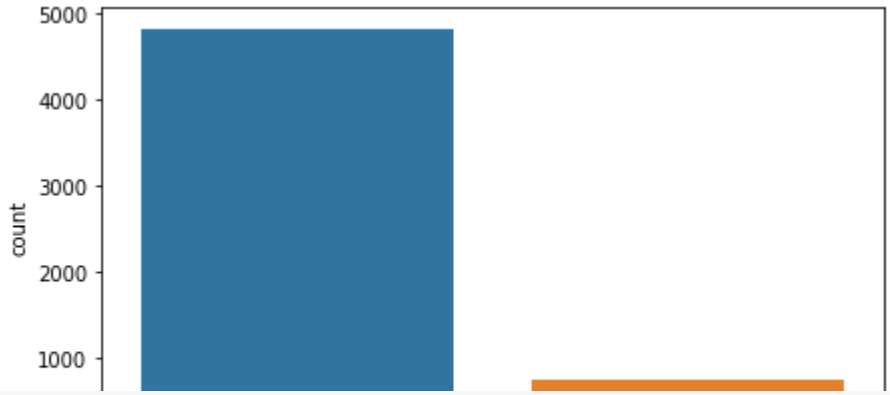
```
(5572, 2)
```

```
df.info()
```

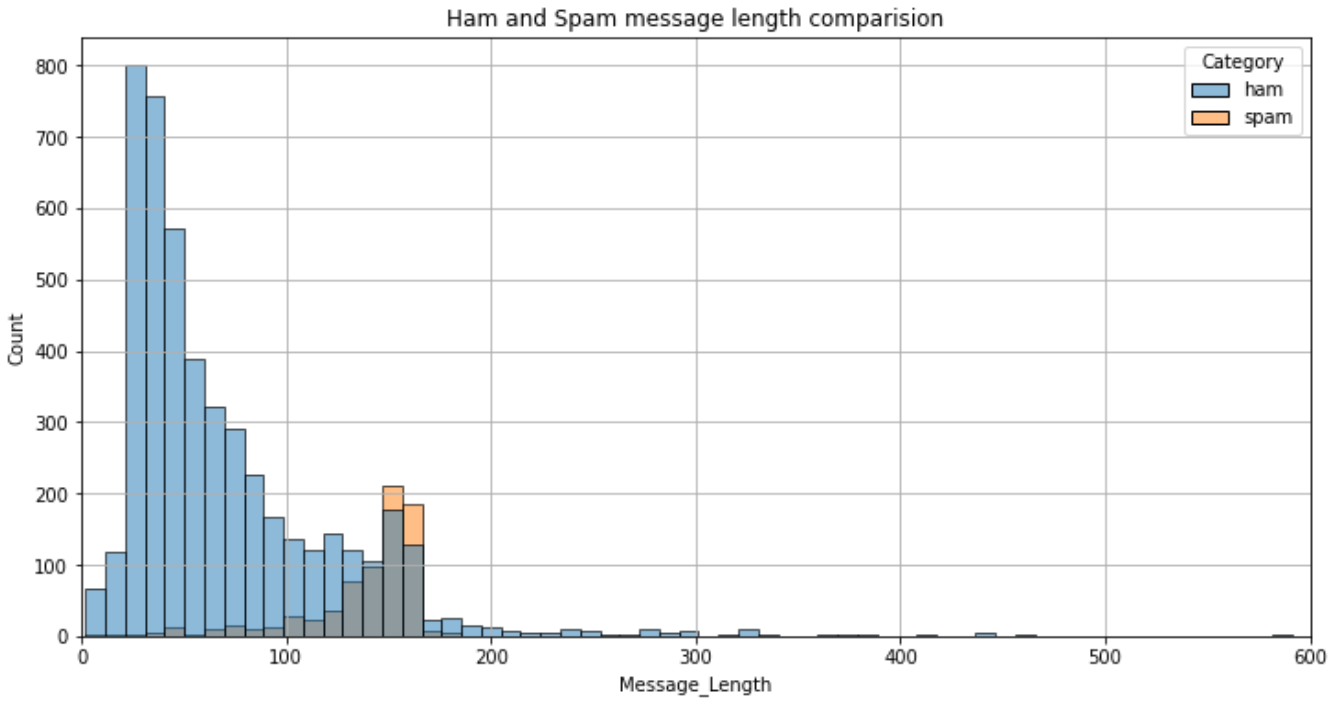
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Category  5572 non-null   object
 1   Message   5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB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

```
plt.figure(figsize=(7,4))
sns.countplot(df.Category)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
```




```
plt.figure(figsize=(12,6))
df['Message_Length']= df['Message'].apply(len)
sns.histplot(x=df['Message_Length'],hue=df['Category'])
plt.xlim((0,600))
plt.title('Ham and Spam message length comparision')
plt.grid()
plt.show()
```




```
from sklearn.preprocessing import LabelEncoder
le= LabelEncoder()
```

```
df.Category=le.fit_transform(df.Category)
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
textFeatures= df['Message'].copy()
```

```
vect= TfidfVectorizer('english')
x= vect.fit_transform(textFeatures)
y=df['Category']
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size= 0.2, random_state= 5)
```

```
from sklearn.naive_bayes import MultinomialNB
mnb= MultinomialNB().fit(x_train, y_train)
print(mnb.predict(x_train))
print(y_train)
```

```
[0 0 1 ... 0 0 0]
1658    0
1509    0
3266    1
5199    0
3217    1
       ..
3046    0
1725    0
4079    0
2254    0
2915    1
Name: Category, Length: 4457, dtype: int64
```

```
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
pred= mnb.predict(x_train)
print(classification_report(y_train,pred))
```

```
                  precision    recall  f1-score   support

               0       0.96      1.00      0.98      3855
               1       1.00      0.76      0.87       602

        accuracy                           0.97      4457
       macro avg       0.98      0.88      0.92      4457
    weighted avg       0.97      0.97      0.97      4457
```

```
print("Confusion Matrix ::> \n", confusion_matrix(y_train,pred))
print("Accuracy Score ::> ", accuracy_score(y_train,pred))
```

```
Confusion Matrix ::>
 [[3855    0]
 [ 142  460]]
Accuracy Score ::>  0.9681400044873233
```

✓ 0s completed at 14:29