In [5]:
```python
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns
import matplotlib.pyplot as plt
```

In [6]:
```python
data = pd.read_csv('spam_ham_dataset.csv.zip')
```

In [7]:
```python
data.head()
```

Out[7]:

	Unnamed: 0	label	text	label_num
0	605	ham	Subject: enron methanol ; meter # : 988291\r\n...	0
1	2349	ham	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	3624	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	4685	spam	Subject: photoshop , windows , office . cheap ...	1
4	2030	ham	Subject: re : indian springs\r\nthis deal is t...	0

In [8]:
```python
data.isnull().sum()
```

Out[8]:
```
Unnamed: 0    0
label         0
text          0
label_num     0
dtype: int64
```

In [9]:
```python
data.shape
```

Out[9]:
```
(5171, 4)
```

In [10]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5171 entries, 0 to 5170
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Unnamed: 0  5171 non-null   int64
 1   label       5171 non-null   object
 2   text        5171 non-null   object
 3   label_num   5171 non-null   int64
dtypes: int64(2), object(2)
memory usage: 161.7+ KB
```

In [11]:
```python
data.describe()
```

Out[11]:

	Unnamed: 0	label_num
count	5171.000000	5171.000000
mean	2585.000000	0.289886
std	1492.883452	0.453753
min	0.000000	0.000000
25%	1292.500000	0.000000
50%	2585.000000	0.000000
75%	3877.500000	1.000000
max	5170.000000	1.000000

In [12]:
```python
data.dtypes
```

Out[12]:
```
Unnamed: 0     int64
label         object
text          object
label_num      int64
dtype: object
```

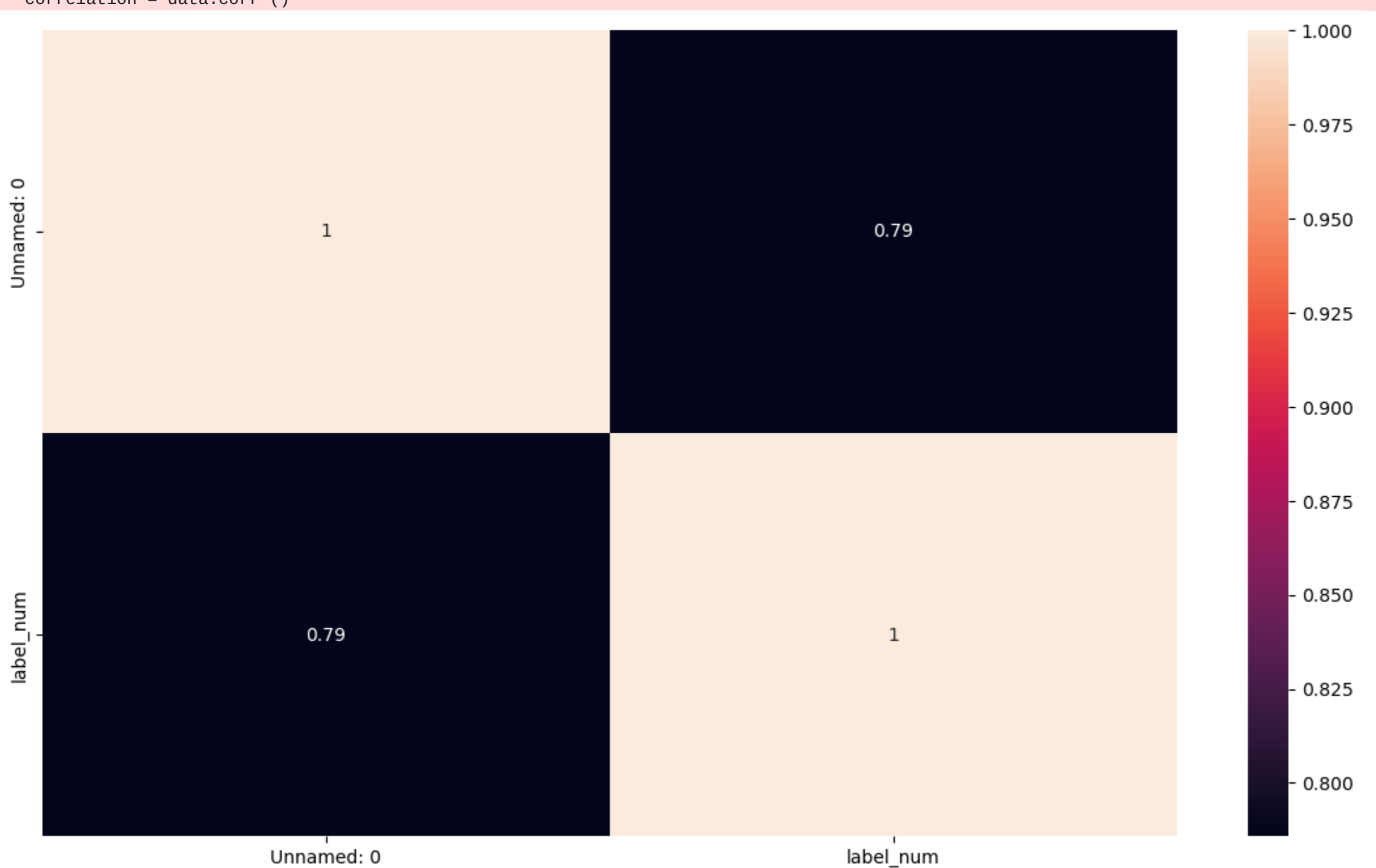
In [13]:
```python
data.tail(5)
```

Out[13]:

	Unnamed: 0	label	text	label_num
5166	1518	ham	Subject: put the 10 on the ft\r\nthe transport...	0
5167	404	ham	Subject: 3 / 4 / 2000 and following noms\r\nhp...	0
5168	2933	ham	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	1409	ham	Subject: industrial worksheets for august 2000...	0
5170	4807	spam	Subject: important online banking alert\r\ndea...	1

In [14]:
```python
import seaborn
correlation = data.corr ()
fig=plt.figure(figsize=(14,8))
seaborn.heatmap(correlation,annot=True)
plt.show()
```

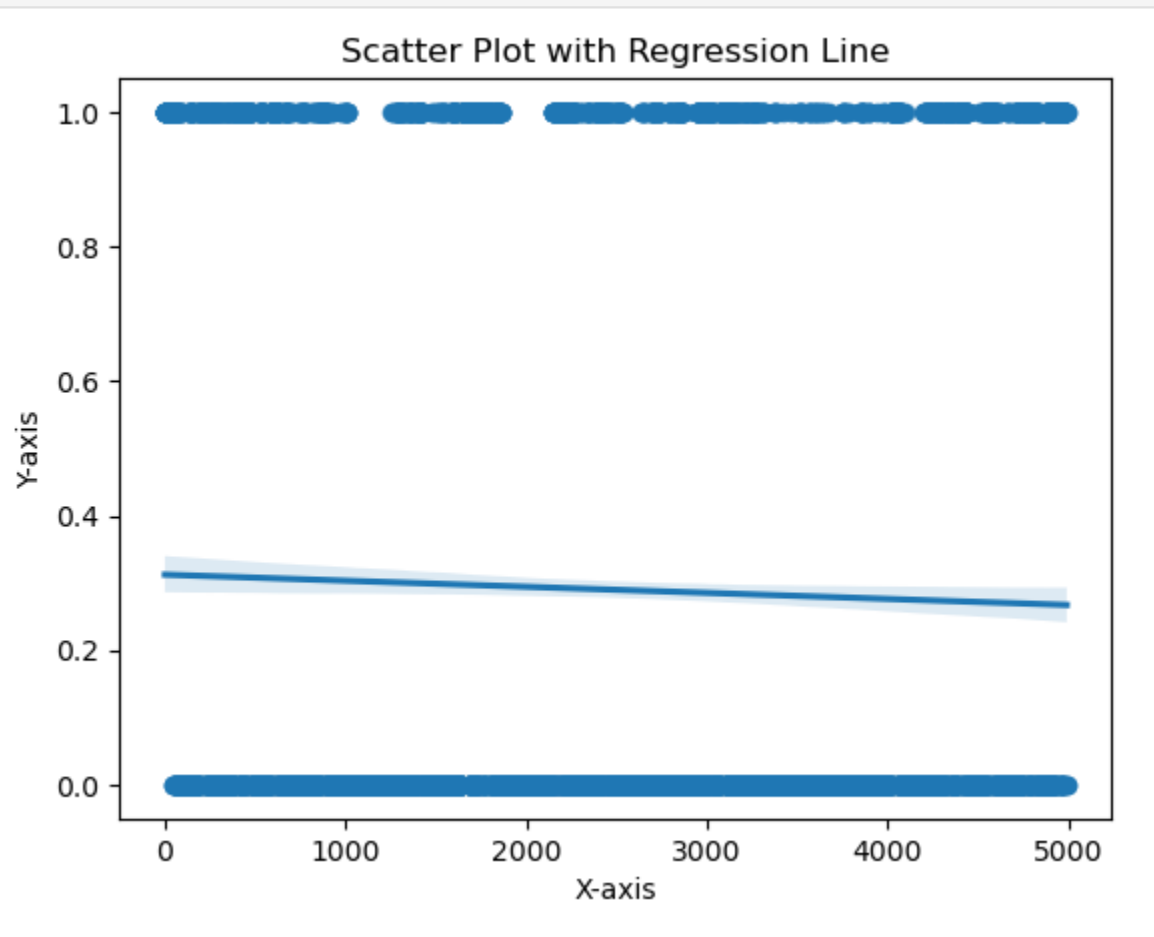
```
C:\Users\Sutharsahana\AppData\Local\Temp\ipykernel_6596\2546683889.py:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future versio
n, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  correlation = data.corr ()
```



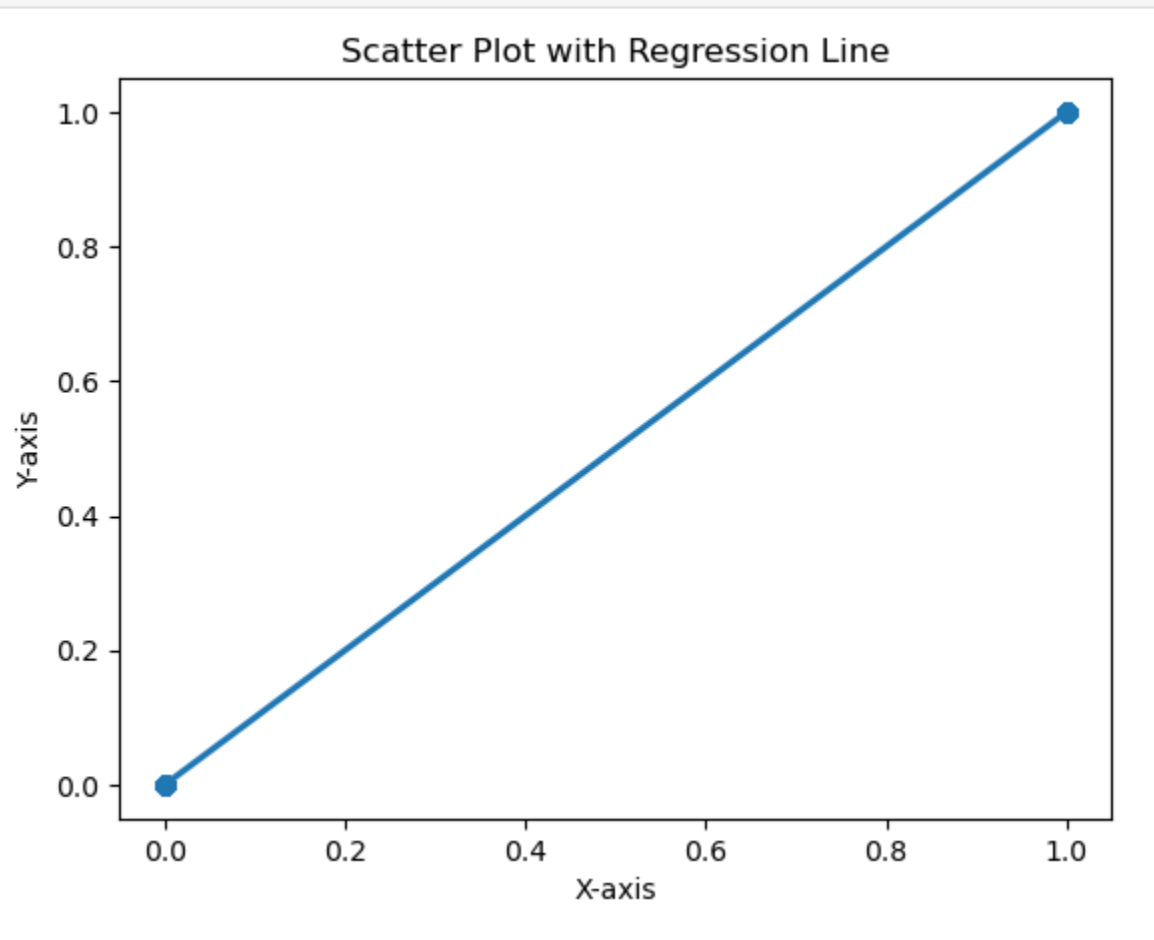
In [15]:
```python
from sklearn.preprocessing import LabelEncoder
Numerics=LabelEncoder()

data['label']=Numerics.fit_transform(data['label'])
data['text']=Numerics.fit_transform(data['text'])

print("ok")

print(data)
```

```
ok
      Unnamed: 0  label  text  label_num
0            605      0  1209          0
1           2349      0  1985          0
2           3624      0  2774          0
3           4685      1  3158          1
4           2030      0  3613          0
...          ...    ...   ...        ...
5166        1518      0  3254          0
5167         404      0   125          0
5168        2933      0   495          0
5169        1409      0  2223          0
5170        4807      1  2195          1

[5171 rows x 4 columns]
```

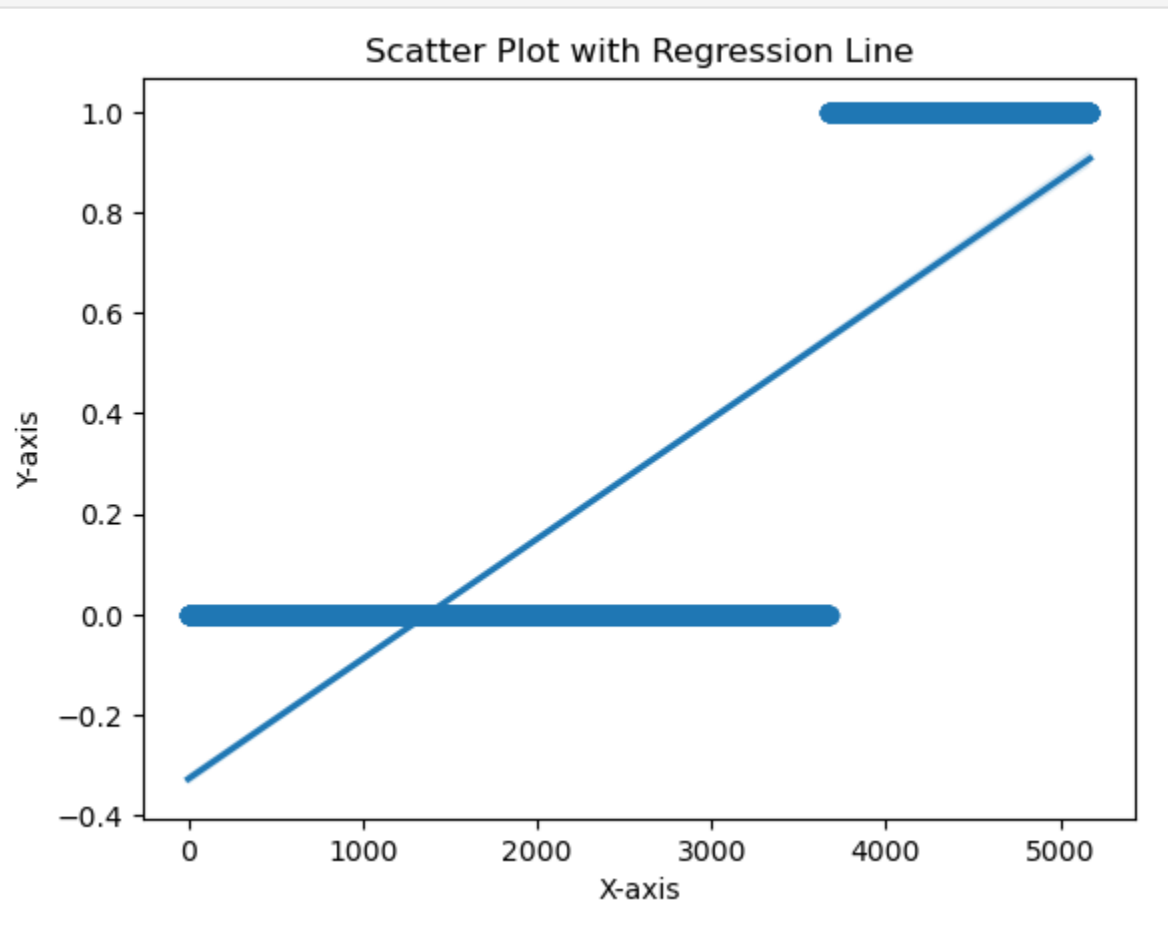
In [16]:
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Sample data
x = data['text']  # X-axis values
y = data['label']  # Y-axis values

# Create scatter plot
sns.scatterplot(x=x, y=y)

# Fit and plot regression line
sns.regplot(x=x, y=y)

# Customize the chart
plt.title("Scatter Plot with Regression Line")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")

# Display the chart
plt.show()
```



In [17]:
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Sample data
x = data['label_num']  # X-axis values
y = data['label']  # Y-axis values

# Create scatter plot
sns.scatterplot(x=x, y=y)

# Fit and plot regression line
sns.regplot(x=x, y=y)

# Customize the chart
plt.title("Scatter Plot with Regression Line")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")

# Display the chart
plt.show()
```



In [18]:
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Sample data
x = data['Unnamed: 0']  # X-axis values
y = data['label']  # Y-axis values

# Create scatter plot
sns.scatterplot(x=x, y=y)

# Fit and plot regression line
sns.regplot(x=x, y=y)

# Customize the chart
plt.title("Scatter Plot with Regression Line")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")

# Display the chart
plt.show()
```



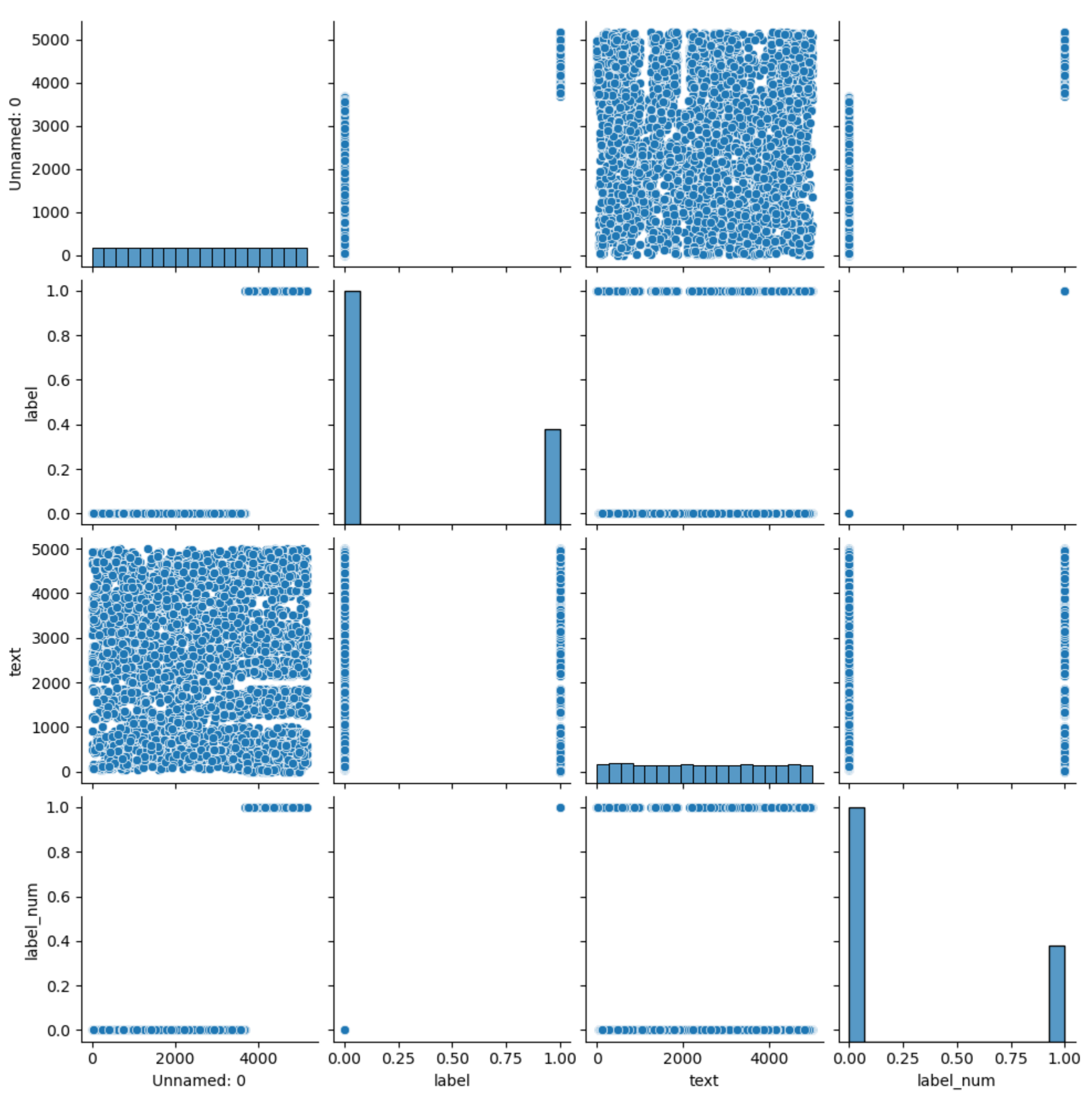
In [19]:
```python
correlation = data.corr ()
correlation.style.background_gradient (cmap = 'BrBG')
```

Out[19]:

	Unnamed: 0	label	text	label_num
Unnamed: 0	1.000000	0.785847	-0.024944	0.785847
label	0.785847	1.000000	-0.028863	1.000000
text	-0.024944	-0.028863	1.000000	-0.028863
label_num	0.785847	1.000000	-0.028863	1.000000

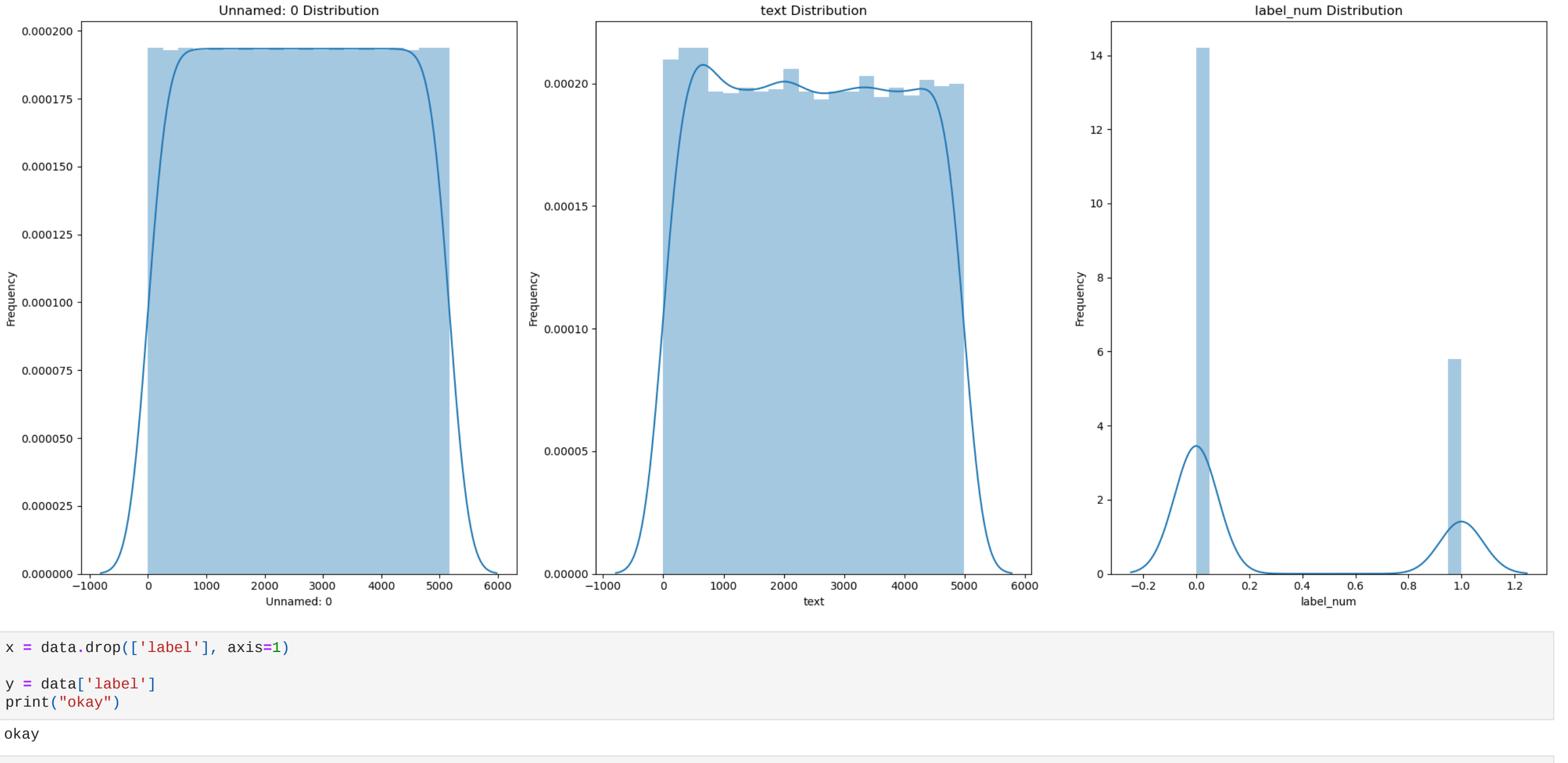
In [20]:
```python
import seaborn as sns  #seaborn
sns.pairplot(data)
```

Out[20]:
```
<seaborn.axisgrid.PairGrid at 0x1e789a85360>
```



In [21]:
```python
 #distribution plots

plt.figure(figsize=(20, 8))

for i, col in enumerate(['Unnamed: 0', 'text', 'label_num']):
    ax = plt.subplot(1, 3, i+1)
    sns.distplot(data[col], bins=20, kde=True)
    ax.set_xlabel(col)
    ax.set_ylabel('Frequency')
    ax.set_title(f'{col} Distribution')

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

```
C:\Users\Sutharsahana\AppData\Local\Temp\ipykernel_6596\2090456027.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data[col], bins=20, kde=True)
C:\Users\Sutharsahana\AppData\Local\Temp\ipykernel_6596\2090456027.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data[col], bins=20, kde=True)
C:\Users\Sutharsahana\AppData\Local\Temp\ipykernel_6596\2090456027.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data[col], bins=20, kde=True)
```



In [22]:
```python
x = data.drop(['label'], axis=1)

y = data['label']
print("okay")
```

```
okay
```

In [23]:
```python
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.33, random_state = 42)
```

In [24]:
```python
x_train.shape, x_test.shape
```

Out[24]:
```
((3464, 3), (1707, 3))
```

In [25]:
```python
x_train.dtypes
```

Out[25]:
```
Unnamed: 0    int64
text          int32
label_num     int64
dtype: object
```

In [26]:
```python
#Import Libraries file

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

from sklearn.model_selection import train_test_split #Train Test Split

from sklearn.naive_bayes import GaussianNB   # Naive Bayes Classifier

from sklearn import preprocessing            # Label Encoder

from sklearn.neighbors import KNeighborsClassifier  # KNN Classsifiers
```

In [27]:
```python
#Train Test split

x = data[["Unnamed: 0","text","label_num"]]
y = data['label']

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30, random_state=0)

x_train.shape
```

Out[27]:
```
(3619, 3)
```

In [28]:
```python
#Import Gaussian Naive Bayes model
from sklearn.naive_bayes import GaussianNB
#Create a Gaussian Classifier
gnb = GaussianNB()
#Train the model using the training sets
gnb.fit(x_train, y_train)
```

Out[28]:
```
▾ GaussianNB
GaussianNB()
```

In [29]:
```python
#Predict the response for test dataset
y_pred = gnb.predict(x_test)
```

In [30]:
```python
# Evaluating model
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 1.0
```

In [31]:
```python
# Evaluating model
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy
print("Accuracy:",metrics.classification_report(y_test, y_pred))
```

```
Accuracy:               precision    recall  f1-score   support

           0       1.00      1.00      1.00      1128
           1       1.00      1.00      1.00       424

    accuracy                           1.00      1552
   macro avg       1.00      1.00      1.00      1552
weighted avg       1.00      1.00      1.00      1552
```

In [ ]:

In [ ]:

In [ ]: