SMS Classification

April 9, 2024

[1]:
```
##Importing Libraties
```

[2]:
```
#import libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
import warnings
warnings.filterwarnings("ignore")
```

[3]:
```
import pandas as pd

#list of possible encodings to try
encodings = ['utf-8', 'latin1','ISO-8859-1','cp1252']
file_path = "spam.csv"

#Attending to read the csv file with different encoding
for encoding in encodings:
    try:
        data = pd.read_csv(file_path, encoding=encoding)
        print(f"file successecfully read with encoding: {encoding}")
        break
    except UnicodeDecodeError:
        print(f"Falled to read with encoding:{encoding}")
        continue

if 'data' in locals():
    print("csv file has been successefully loaded.")
else:
    print("All encoding attempts failed. unable to read the csv file.")
```

```
Falled to read with encoding:utf-8
file successecfully read with encoding: latin1
csv file has been successefully loaded.
```

[4]:
```
data.sample(5)
```

[4]:
```
        v1                                                  v2 Unnamed: 2  \
1770  ham  Dont show yourself. How far. Put new pictures …         NaN
3841  ham  chile, please! It's only a  &lt;DECIMAL&gt;  h…         NaN
1028  ham  Lol you forgot it eh ? Yes, I'll bring it in babe         NaN
2910  ham                    Sorry,in meeting I'll call later         NaN
2661  ham                  Do you know when dad will be back?         NaN

     Unnamed: 3 Unnamed: 4
1770        NaN        NaN
3841        NaN        NaN
1028        NaN        NaN
2910        NaN        NaN
2661        NaN        NaN
```

1 Data Cleaning

[5]:
```
data.tail()
```

[5]:
```
        v1                                                  v2 Unnamed: 2  \
5567  spam  This is the 2nd time we have tried 2 contact u…         NaN
5568   ham              Will Ì_ b going to esplanade fr home?         NaN
5569   ham  Pity, * was in mood for that. So…any other s…         NaN
5570   ham  The guy did some bitching but I acted like i'd…         NaN
5571   ham                         Rofl. Its true to its name         NaN

     Unnamed: 3 Unnamed: 4
5567        NaN        NaN
5568        NaN        NaN
5569        NaN        NaN
5570        NaN        NaN
5571        NaN        NaN
```

[6]:
```
data.head()
```

[6]:
```
     v1                                                  v2 Unnamed: 2  \
0   ham  Go until jurong point, crazy.. Available only …         NaN
1   ham                      Ok lar… Joking wif u oni…         NaN
2  spam  Free entry in 2 a wkly comp to win FA Cup fina…         NaN
3   ham  U dun say so early hor… U c already then say…         NaN
4   ham  Nah I don't think he goes to usf, he lives aro…         NaN

  Unnamed: 3 Unnamed: 4
0        NaN        NaN
1        NaN        NaN
2        NaN        NaN
3        NaN        NaN
4        NaN        NaN
```

[7]:
```
data.shape
```

[7]:
```
(5572, 5)
```

[8]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

[9]:
```
data.duplicated().sum()
```

[9]:
```
403
```

[10]:
```
data
```

[10]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only …	NaN	NaN	NaN
1	ham	Ok lar… Joking wif u oni…	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina…	NaN	NaN	NaN
3	ham	U dun say so early hor… U c already then say…	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro…	NaN	NaN	NaN
…	…	…	…	…	…
5567	spam	This is the 2nd time we have tried 2 contact u…	NaN	NaN	NaN
5568	ham	Will Ì_ b going to esplanade fr home?	NaN	NaN	NaN
5569	ham	Pity, * was in mood for that. So…any other s…	NaN	NaN	NaN
5570	ham	The guy did some bitching but I acted like i'd…	NaN	NaN	NaN
5571	ham	Rofl. Its true to its name	NaN	NaN	NaN

```
5570        NaN        NaN
5571        NaN        NaN

[5572 rows x 5 columns]
```

[11]:
```
#drop Last 3 cols
data.drop(columns=['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],inplace=True)
```

[12]:
```
data.sample(5)
```

[12]:
```
       v1                                                  v2
4150  ham                      ÌÏ comin to fetch us oredi…
3568  ham                     She's fine. Sends her greetings
3780  ham  Dear friends, sorry for the late information. …
2763  ham  Say this slowly.? GOD,I LOVE YOU &amp;amp; I NEED …
1784  ham  Dont search love, let love find U. Thats why i…
```

[13]:
```
# renaming the cols
data.rename(columns={'v1':'label','v2':'message'},inplace=True)
data.sample(5)
```

[13]:
```
     label                                              message
224   spam  500 New Mobiles from 2004, MUST GO! Txt: NOKIA…
331    ham  Maybe i could get book out tomo then return it…
4539   ham  Urgh, coach hot, smells of chip fat! Thanks ag…
173    ham  Bloody hell, cant believe you forgot my surnam…
1425   ham          I'll be at mu in like  &lt;#&gt;  seconds
```

[14]:
```
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
```

[15]:
```
data["label"]=encoder.fit_transform(data['label'])
```

[16]:
```
data.head()
```

[16]:
```
   label                                              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…
4      0  Nah I don't think he goes to usf, he lives aro…
```

[17]:
```
data.tail()
```

[17]:
```
      label                                              message
5567      1  This is the 2nd time we have tried 2 contact u…
5568      0              Will Ì_ 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
```

[18]:
```
# Check Missing Values
data.isnull().sum()
```

[18]:
```
label      0
message    0
dtype: int64
```

[19]:
```
# check duplicate values
data.duplicated().sum()
```

[19]:
```
403
```

[20]:
```
#drop duplicates
data.drop_duplicates(inplace=True)
```

[21]:
```
data.duplicated().sum()
```

[21]:
```
0
```

[22]:
```
data.reset_index(drop=True, inplace=True)
```

[23]:
```
data.shape
```

[23]:
```
(5169, 2)
```

2 EDA

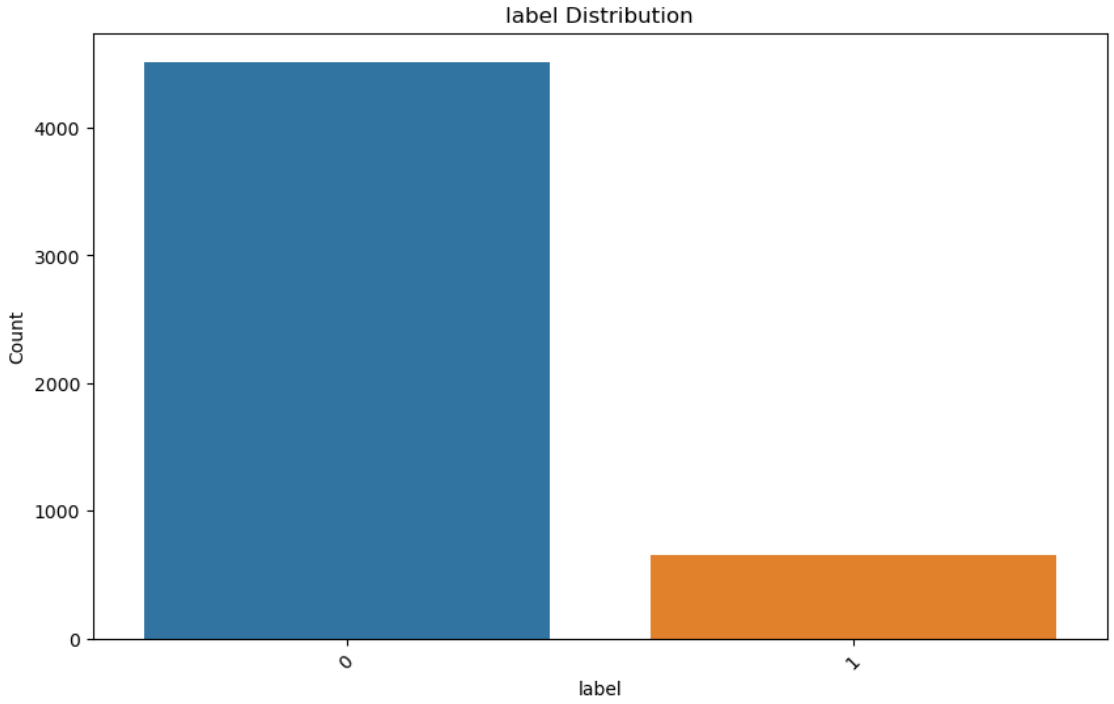
[24]:
```
data.head()
```

[24]:
```
   label                                            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…
4      0  Nah I don't think he goes to usf, he lives aro…
```

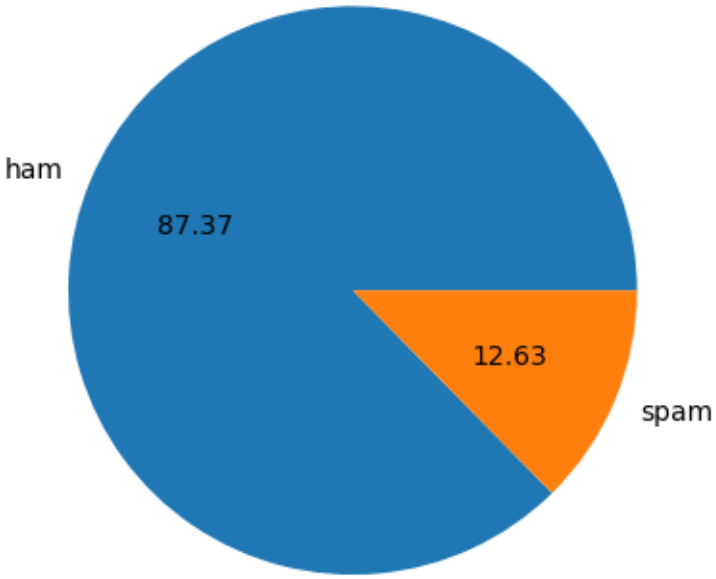
[25]:
```
data['label'].value_counts()
```

[25]:
```
label
0    4516
1     653
Name: count, dtype: int64
```

[26]:
```
# Example: Bar plot for 'label' counts
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='label',  label=['ham','spam'])
plt.title('label Distribution')
plt.xlabel('label')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



[27]:
```
import matplotlib.pyplot as plt
plt.pie(data['label'].value_counts(), labels=['ham','spam'],autopct="%0.2f")
plt.show()
```



[28]:
```
# Data is imbalanced
```

[29]:
```
import nltk
```

[30]:
```
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\mohsi\AppData\Roaming\nltk_data…
[nltk_data]   Package punkt is already up-to-date!
```

[30]: True

[31]:
```
data['num_characters'] = data['message'].apply(len)
```

[32]:
```
data.head()
```

[32]:

	label	message	num_characters
0	0	Go until jurong point, crazy.. Available only …	111
1	0	Ok lar… Joking wif u oni…	29
2	1	Free entry in 2 a wkly comp to win FA Cup fina…	155
3	0	U dun say so early hor… U c already then say…	49
4	0	Nah I don't think he goes to usf, he lives aro…	61

[33]:
```
#num of words
data['num_words']= data['message'].apply(lambda x:len(nltk.word_tokenize(x)))
```

[34]:
```
data.head()
```

[34]:
```
   label                                            message  num_characters  \
0      0  Go until jurong point, crazy.. Available only …             111
1      0                      Ok lar… Joking wif u oni…              29
2      1  Free entry in 2 a wkly comp to win FA Cup fina…             155
3      0  U dun say so early hor… U c already then say…               49
4      0  Nah I don't think he goes to usf, he lives aro…              61

   num_words
0         24
1          8
2         37
3         13
4         15
```

[35]:
```
data['num_sentences'] = data['message'].apply(lambda x:len(nltk.
↪sent_tokenize(x)))
```

[36]:
```
data
```

[36]:
```
      label                                            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…
4         0  Nah I don't think he goes to usf, he lives aro…
…     …                                                  …
5164      1  This is the 2nd time we have tried 2 contact u…
5165      0             Will Ì_ b going to esplanade fr home?
5166      0  Pity, * was in mood for that. So…any other s…
5167      0  The guy did some bitching but I acted like i'd…
5168      0                       Rofl. Its true to its name

      num_characters  num_words  num_sentences
0                111         24              2
1                 29          8              2
2                155         37              2
3                 49         13              1
4                 61         15              1
…                …          …              …
5164             161         35              4
5165              37          9              1
5166              57         15              2
```

```
5167             125          27               1
5168              26           7               2

[5169 rows x 5 columns]
```

[37]:
```
data[['num_characters','num_words','num_sentences']].describe()
```

[37]:
```
       num_characters    num_words  num_sentences
count     5169.000000  5169.000000    5169.000000
mean        78.977945    18.455794       1.965564
std         58.236293    13.324758       1.448541
min          2.000000     1.000000       1.000000
25%         36.000000     9.000000       1.000000
50%         60.000000    15.000000       1.000000
75%        117.000000    26.000000       2.000000
max        910.000000   220.000000      38.000000
```

[38]:
```
#Ham
data[data['label'] == 0][['num_characters','num_words','num_sentences']].
 ↪describe()
```

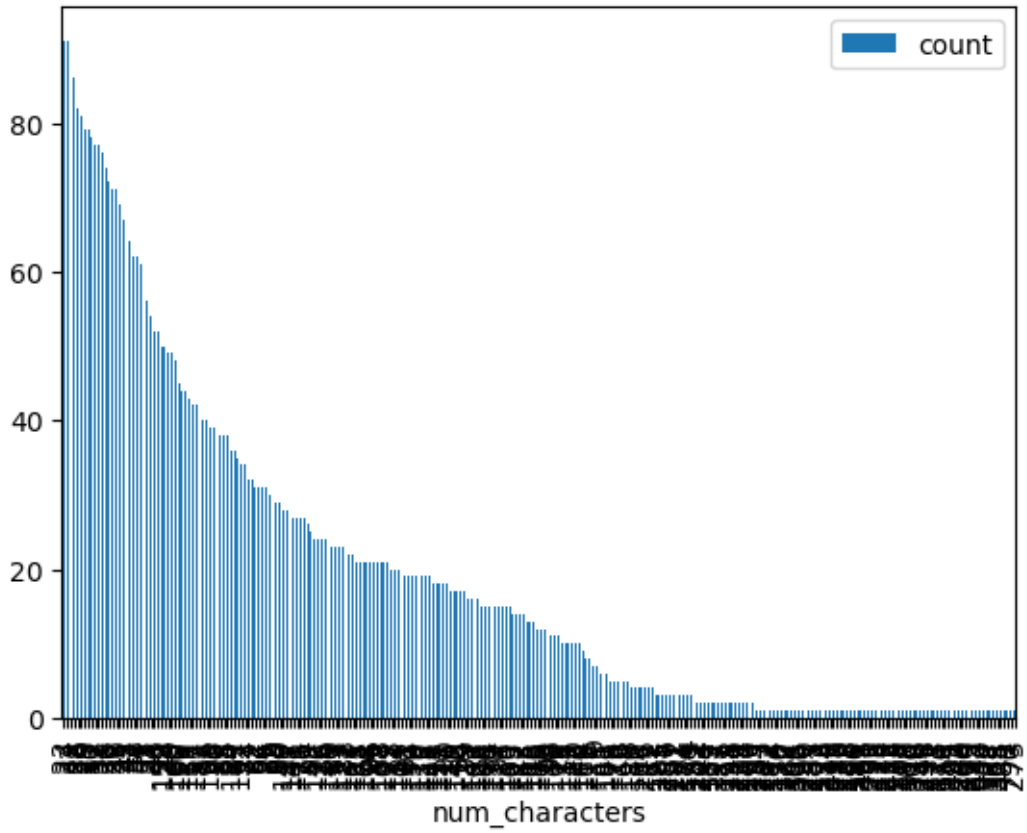
[38]:
```
       num_characters    num_words  num_sentences
count     4516.000000  4516.000000    4516.000000
mean        70.459256    17.123782       1.820195
std         56.358207    13.493970       1.383657
min          2.000000     1.000000       1.000000
25%         34.000000     8.000000       1.000000
50%         52.000000    13.000000       1.000000
75%         90.000000    22.000000       2.000000
max        910.000000   220.000000      38.000000
```

[39]:
```
#spam
data[data['label'] == 1][['num_characters','num_words','num_sentences']].
 ↪describe()
```

[39]:
```
       num_characters   num_words  num_sentences
count      653.000000  653.000000     653.000000
mean       137.891271   27.667688       2.970904
std         30.137753    7.008418       1.488425
min         13.000000    2.000000       1.000000
25%        132.000000   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
```

[40]:
```
data["num_characters"].value_counts().plot(kind='bar',legend=True)
```

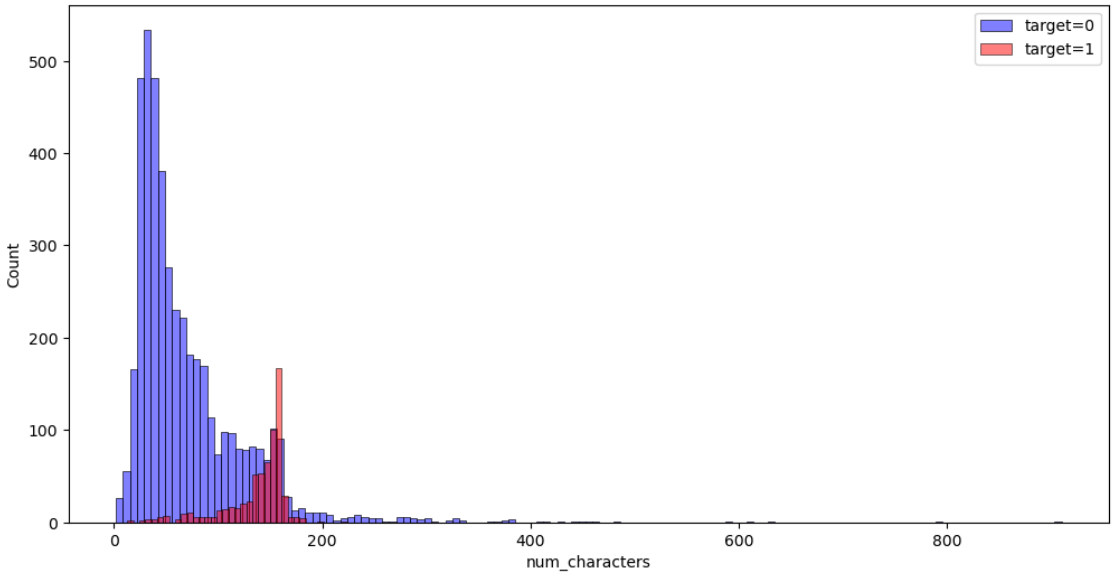
[40]:
```
<Axes: xlabel='num_characters'>
```



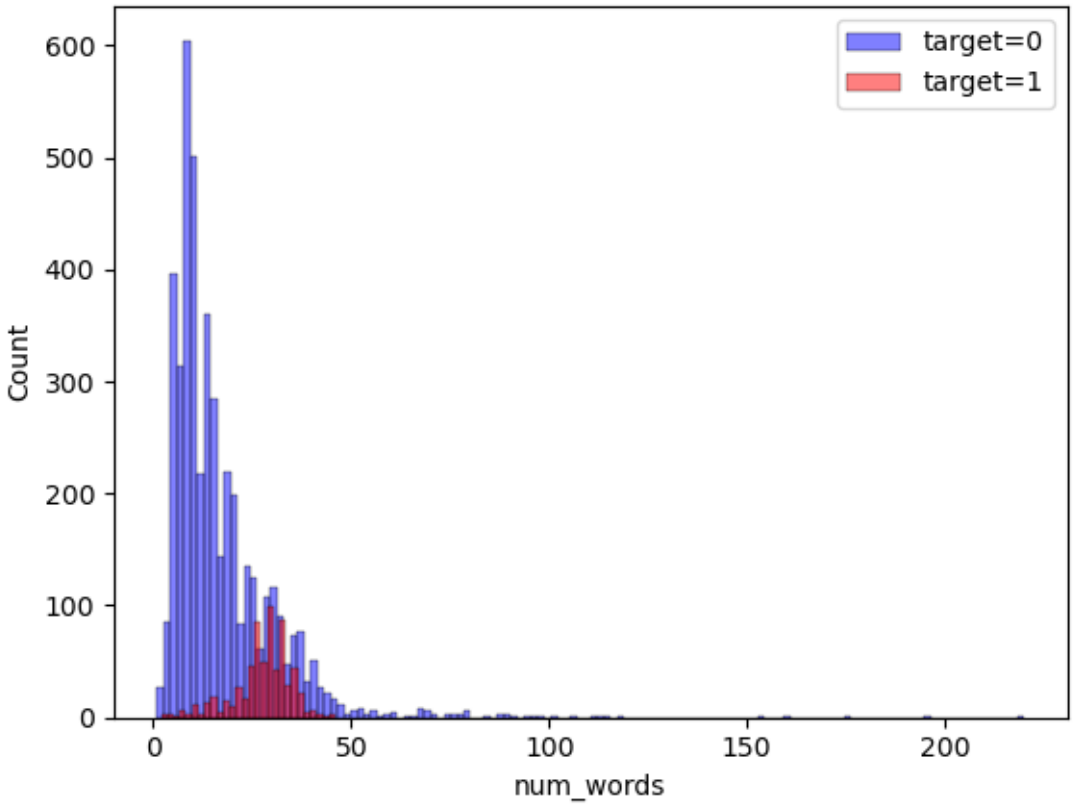
[41]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Check the column names in your DataFrame
print(data.columns)

# Set the figure size
plt.figure(figsize=(12, 6))

# Filter the DataFrame for 'target' == 0 and plot its histogram in blue
sns.histplot(data=data[data['label'] == 0], x='num_characters', color='blue',␣
 ↪alpha=0.5, label='target=0')

# Filter the DataFrame for 'target' == 1 and plot its histogram in red
sns.histplot(data=data[data['label'] == 1], x='num_characters', color='red',␣
 ↪alpha=0.5, label='target=1')

# Add legend
plt.legend()
```

```
# Show the plot
plt.show()
```

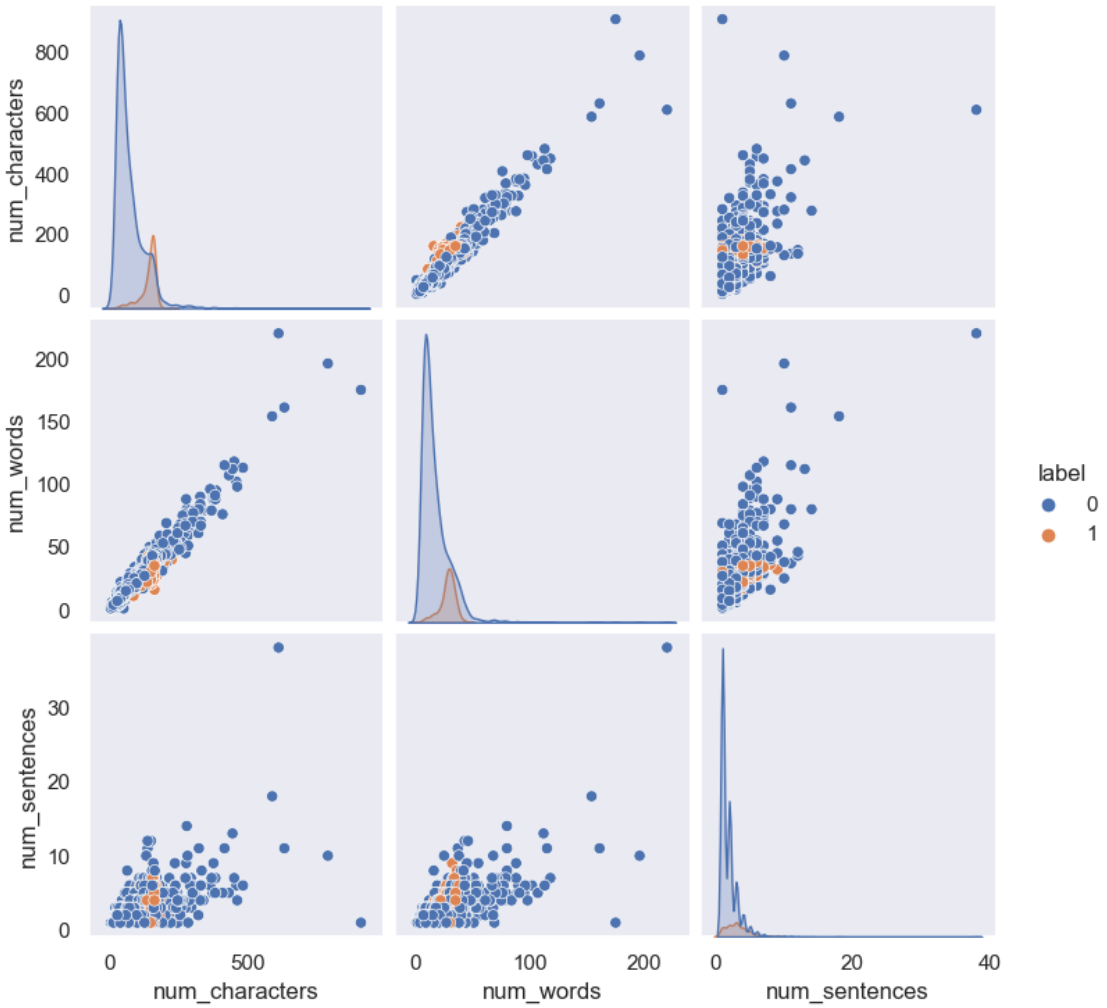
```
Index(['label', 'message', 'num_characters', 'num_words', 'num_sentences'],
dtype='object')
```



[42]:
```
# Filter the DataFrame for 'target' == 0 and plot its histogram in blue
sns.histplot(data=data[data['label'] == 0], x='num_words', color='blue',␣
 ↪alpha=0.5, label='target=0')

# Filter the DataFrame for 'target' == 1 and plot its histogram in red
sns.histplot(data=data[data['label'] == 1], x='num_words', color='red', alpha=0.
 ↪5, label='target=1')

# Add legend
plt.legend()

# Show the plot
plt.show()
```



[43]:
```
sns.set(style="dark")

# Create the pairplot
sns.pairplot(data, hue="label", diag_kind="kde")

# Show the plot
plt.show()
```

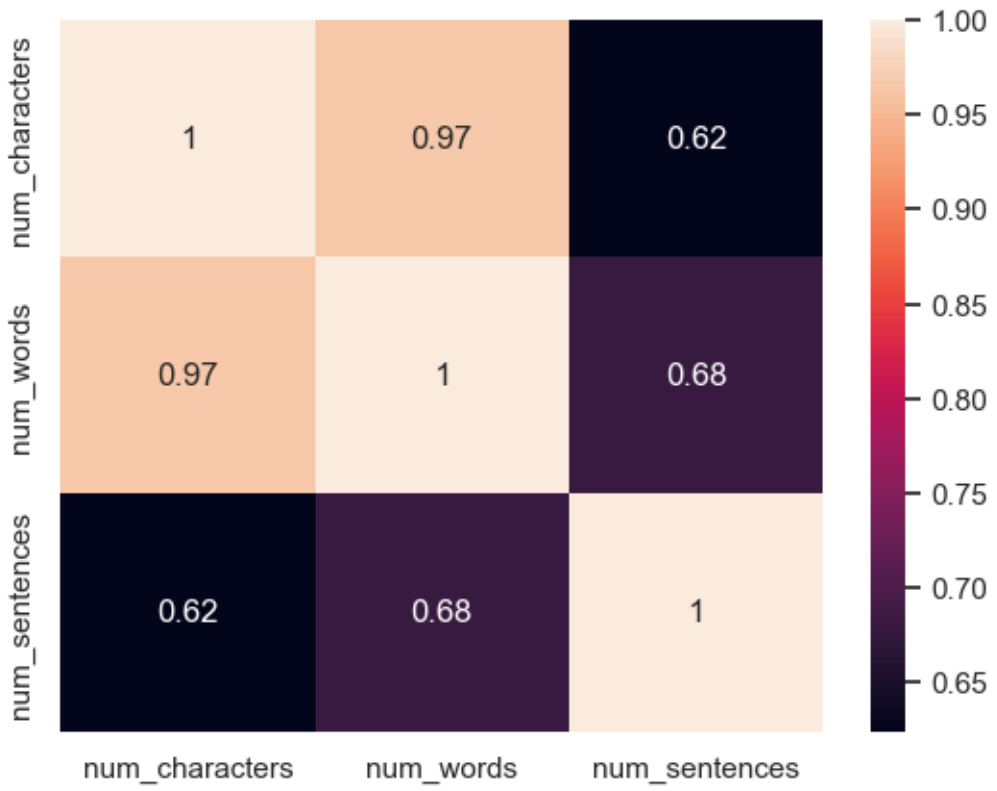


[44]:
```
print(data.dtypes)
```

```
label              int32
message           object
num_characters     int64
num_words          int64
num_sentences      int64
dtype: object
```

[45]:
```
numeric_df = data.select_dtypes(include=['int64', 'float64'])
sns.heatmap(numeric_df.corr(), annot=True)
```

[45]: <Axes: >



3 3. Data Preprocessing

Lower case

Tokenization

Removing special characters

Removing stop words and punctuation

Stemming

[46]:
```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
import string

# Initialize the Porter Stemmer
ps = PorterStemmer()

def transform_text(message):
    # Convert text to lowercase
```

```
    message = message.lower()

    # Tokenize the text
    message_tokens = word_tokenize(message)

    # Remove non-alphanumeric characters and stopwords
    filtered_tokens = [ps.stem(token) for token in message_tokens if token.
 ↪isalnum() and token not in stopwords.words('english') and token not in␣
 ↪string.punctuation]

    # Join the filtered tokens back into a string
    return " ".join(filtered_tokens)

# Example usage
message = "This is an example text, with some stopwords and punctuation. We␣
 ↪will transform it using the provided code."
transformed_text = transform_text(message)
print(transformed_text)
```

```
exampl text stopword punctuat transform use provid code
```

[47]:
```
data['message'][10]
```

[47]:
```
"I'm gonna be home soon and i don't want to talk about this stuff anymore
tonight, k? I've cried enough today."
```

[48]:
```
ps.stem('loving')
```

[48]:
```
'love'
```

[49]:
```
data['transformed_text'] = data['message'].apply(transform_text)
```

[50]:
```
data.head()
```

[50]:

	label	message	num_characters	num_words	num_sentences	transformed_text
0	0	Go until jurong point, crazy.. Available only …	111	24	2	go jurong point crazi avail bugi n great world…
1	0	Ok lar… Joking wif u oni…	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina…	155	37	2	free entri 2 wkli comp win fa cup final tkt 21…
3	0	U dun say so early hor… U c already then say…	49	13	1	u dun say earli hor u c alreadi say
4	0	Nah I don't think he goes to usf, he lives aro…	61	15	1	nah think goe usf live around though

[51]:
```
pip install wordcloud
```

```
Requirement already satisfied: wordcloud in d:\user\lib\site-packages (1.9.3)
Requirement already satisfied: numpy>=1.6.1 in d:\user\lib\site-packages (from
wordcloud) (1.24.3)
Requirement already satisfied: pillow in d:\user\lib\site-packages (from
wordcloud) (9.4.0)
Requirement already satisfied: matplotlib in d:\user\lib\site-packages (from
wordcloud) (3.7.2)
Requirement already satisfied: contourpy>=1.0.1 in d:\user\lib\site-packages
(from matplotlib->wordcloud) (1.0.5)
Requirement already satisfied: cycler>=0.10 in d:\user\lib\site-packages (from
matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in d:\user\lib\site-packages
(from matplotlib->wordcloud) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in d:\user\lib\site-packages
(from matplotlib->wordcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in d:\user\lib\site-packages
(from matplotlib->wordcloud) (23.1)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in d:\user\lib\site-
packages (from matplotlib->wordcloud) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in d:\user\lib\site-packages
(from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: six>=1.5 in d:\user\lib\site-packages (from
python-dateutil>=2.7->matplotlib->wordcloud) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

[52]:
```
from wordcloud import WordCloud
wc = WordCloud(width=500,height=500,min_font_size=10,background_color='white')
```

[53]:
```
spam_wc = wc.generate(data[data['label'] == 1]['transformed_text'].str.
↪cat(sep=" "))
```

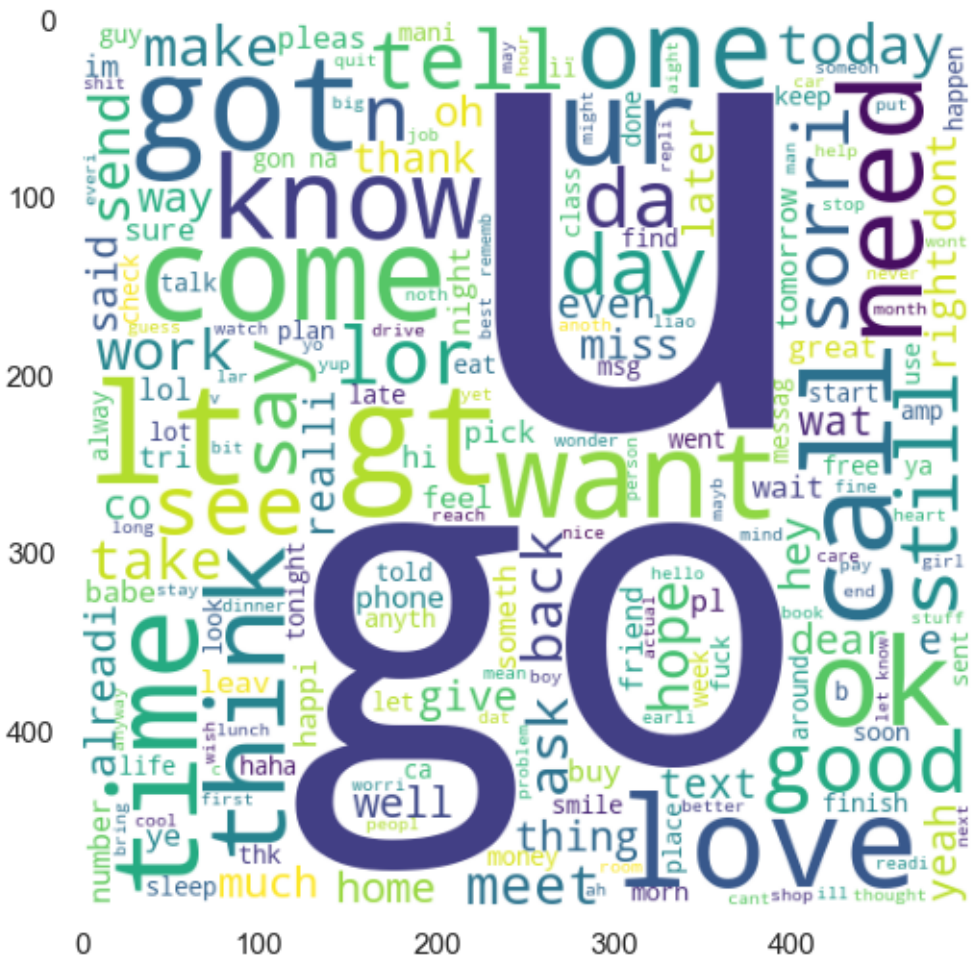
[54]:
```
plt.figure(figsize=(15,6))
plt.imshow(spam_wc)
```

[54]:
```
<matplotlib.image.AxesImage at 0x23c2b2daad0>
```

[55]:
```
ham_wc = wc.generate(data[data['label'] == 0]['transformed_text'].str.cat(sep=" "))
```

[56]:
```
plt.figure(figsize=(15,6))
plt.imshow(ham_wc)
```

[56]: <matplotlib.image.AxesImage at 0x23c2b487ed0>



[57]:
```
data.head()
```

[57]:

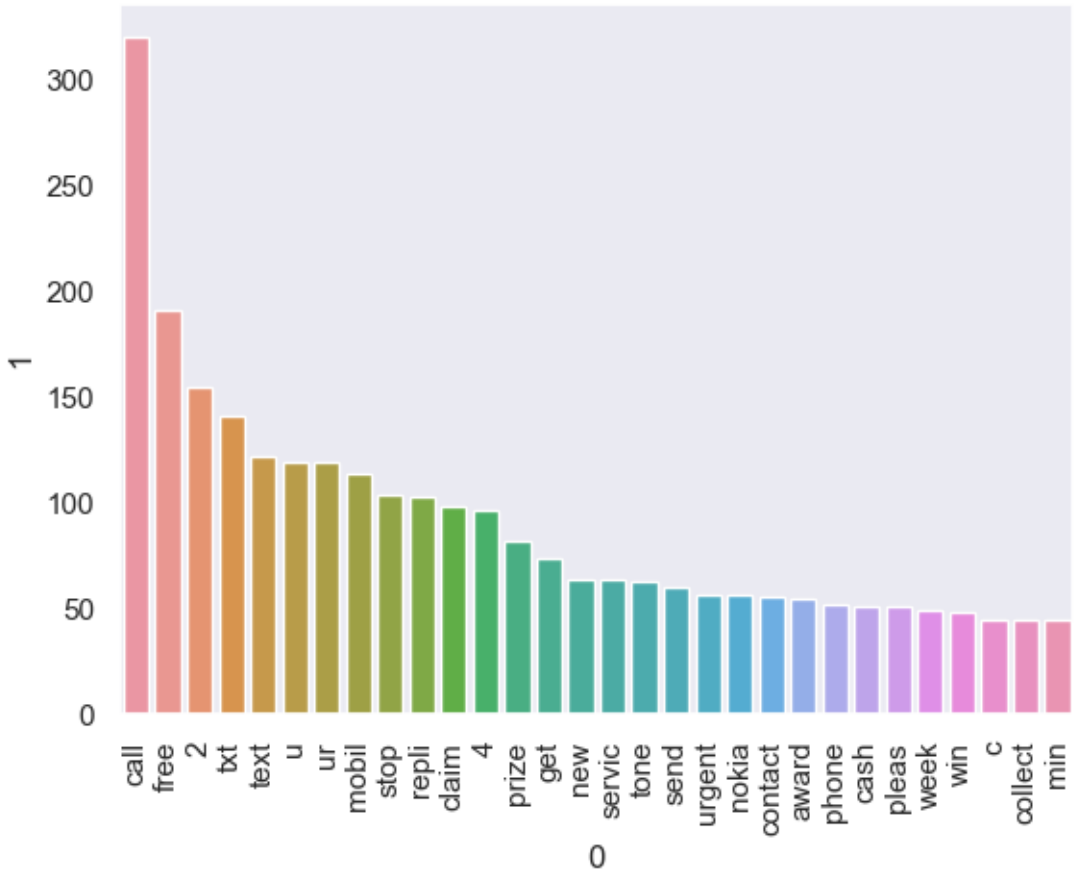
	label	message	num_characters	num_words	num_sentences	transformed_text
0	0	Go until jurong point, crazy.. Available only …	111	24	2	go jurong point crazi avail bugi n great world…
1	0	Ok lar… Joking wif u oni…	29	8	2	ok lar joke wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina…	155	37	2	free entri 2 wkli comp win fa cup final tkt 21…
3	0	U dun say so early hor… U c already then say…	49	13	1	u dun say earli hor u c alreadi say
4	0	Nah I don't think he goes to usf, he lives aro…	61	15	1	nah think goe usf live around though

[58]:
```
spam_corpus = []
for msg in data[data['label'] == 1]['transformed_text'].tolist():
    for word in msg.split():
        spam_corpus.append(word)
```

[59]:
```
len(spam_corpus)
```

[59]: 9939

[60]:
```
from collections import Counter
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

# Assuming spam_corpus is a list of strings
word_counts = Counter(spam_corpus)

# Convert the Counter object to a DataFrame and select the 30 most common␣
 ↪elements
most_common_df = pd.DataFrame(word_counts.most_common(30))

# Plotting
sns.barplot(x=most_common_df[0], y=most_common_df[1])
plt.xticks(rotation='vertical')
plt.show()
```



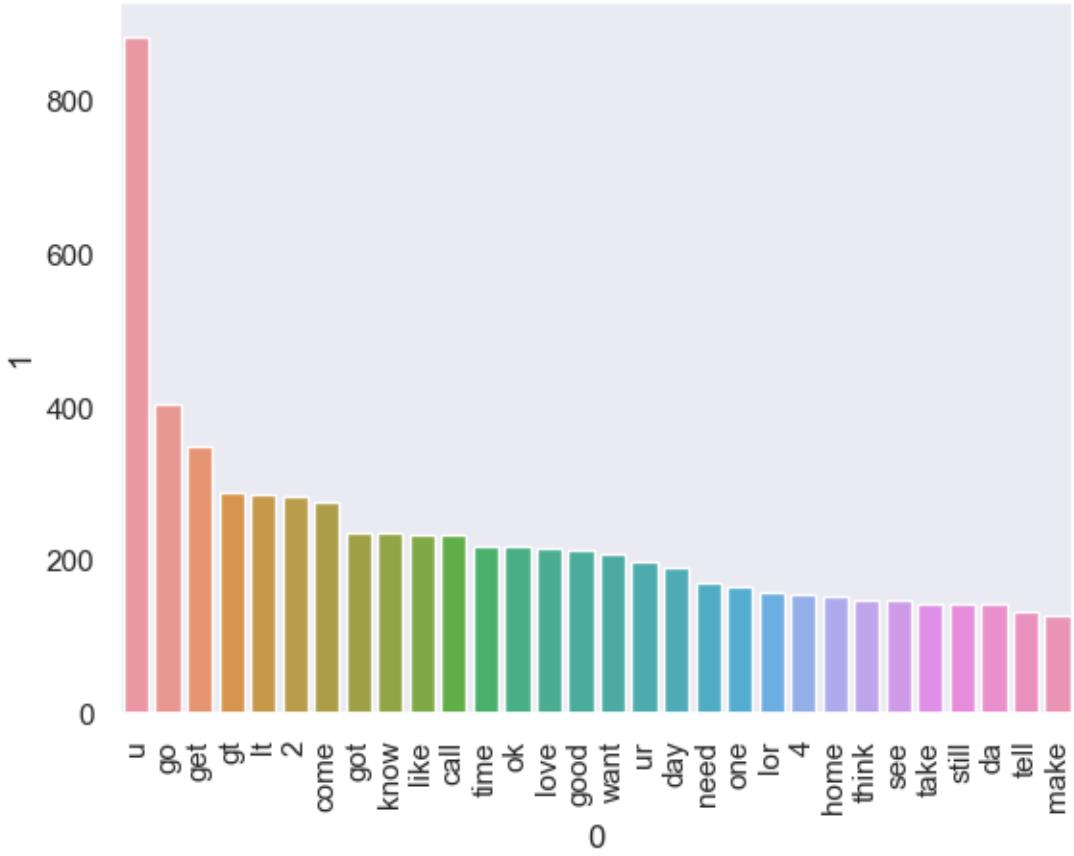
[61]:
```
ham_corpus = []
for msg in data[data['label'] == 0]['transformed_text'].tolist():
    for word in msg.split():
        ham_corpus.append(word)
```

[62]:
```
len(ham_corpus)
```

[62]: 35404

[63]:
```
from collections import Counter
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

# Assuming ham_corpus is a list of strings
word_counts = Counter(ham_corpus)

# Convert the Counter object to a DataFrame and select the 30 most common␣
 ↪elements
```

```
most_common_df = pd.DataFrame(word_counts.most_common(30))

# Plotting
sns.barplot(x=most_common_df[0], y=most_common_df[1])
plt.xticks(rotation='vertical')
plt.show()
```



4 Model Building

[64]:
```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

# Using CountVectorizer
cv = CountVectorizer(max_features=3000)  # Set the maximum number of features
X = cv.fit_transform(data['transformed_text']).toarray()  # Fit the vectorizer␣
 ↪and transform the text data

# Using TfidfVectorizer
tfidf = TfidfVectorizer(max_features=3000)  # Set the maximum number of features
```

```
X_tfidf = tfidf.fit_transform(data['transformed_text']).toarray() # Fit the␣
↪vectorizer and transform the text data
```

[65]:
```
X.shape
```

[65]: (5169, 3000)

[66]:
```
y=data['label'].values
```

[67]:
```
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=2)
```

[68]:
```
from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
from sklearn.metrics import accuracy_score,confusion_matrix,precision_score
```

[69]:
```
# Gaussian Naive Bayes
gnb = GaussianNB()

# Multinomial Naive Bayes
mnb = MultinomialNB()

# Bernoulli Naive Bayes
bnb = BernoulliNB()
```

[70]:
```
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score

# Instantiate the Gaussian Naive Bayes classifier
gnb = GaussianNB()

# Fit the classifier to the training data and make predictions on the test data
y_pred1 = gnb.fit(X_train, y_train).predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred1)
print("Accuracy:", accuracy)

# Calculate confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred1)
print("Confusion Matrix:")
print(conf_matrix)

# Calculate precision score
precision = precision_score(y_test, y_pred1)
print("Precision Score:", precision)
```

```
Accuracy: 0.874274661508704
```

```
Confusion Matrix:
[[787 109]
 [ 21 117]]
Precision Score: 0.5176991150442478
```

[71]:
```
mnb.fit(X_train,y_train)
y_pred2 = mnb.predict(X_test)
print(accuracy_score(y_test,y_pred2))
print(confusion_matrix(y_test,y_pred2))
print(precision_score(y_test,y_pred2))
```

```
0.971953578336557
[[880  16]
 [ 13 125]]
0.8865248226950354
```

[72]:
```
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score

# Fit the Bernoulli Naive Bayes classifier to the training data and make␣
 ↪predictions on the test data
y_pred3 = bnb.fit(X_train, y_train).predict(X_test)

# Calculate and print accuracy
print("Accuracy:", accuracy_score(y_test, y_pred3))

# Calculate and print confusion matrix
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred3))

# Calculate and print precision score
print("Precision Score:", precision_score(y_test, y_pred3))
```

```
Accuracy: 0.9835589941972921
Confusion Matrix:
[[895   1]
 [ 16 122]]
Precision Score: 0.991869918699187
```

[73]:
```
# tfidf --> MNB
```

[74]:
```
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier,␣
 ↪BaggingClassifier, ExtraTreesClassifier, GradientBoostingClassifier
from xgboost import XGBClassifier
```

```
svc = SVC(kernel='sigmoid', gamma=1.0)
knc = KNeighborsClassifier()
mnb = MultinomialNB()
dtc = DecisionTreeClassifier(max_depth=5)
lrc = LogisticRegression(solver='liblinear', penalty='l1')
rfc = RandomForestClassifier(n_estimators=50, random_state=2)
abc = AdaBoostClassifier(n_estimators=50, random_state=2)
bc = BaggingClassifier(n_estimators=50, random_state=2)
etc = ExtraTreesClassifier(n_estimators=50, random_state=2)
gbdt = GradientBoostingClassifier(n_estimators=50,random_state=2)
xgb = XGBClassifier(n_estimators=50,random_state=2)
```

[75]:
```
clfs = {
    'SVC' : svc,
    'KN' : knc,
    'NB': mnb,
    'DT': dtc,
    'LR': lrc,
    'RF': rfc,
    'AdaBoost': abc,
    'BgC': bc,
    'ETC': etc,
    'GBDT':gbdt,
    'xgb':xgb
}
```

[76]:
```
def train_classifier(clf,X_train,y_train,X_test,y_test):
    clf.fit(X_train,y_train)
    y_pred = clf.predict(X_test)
    accuracy = accuracy_score(y_test,y_pred)
    precision = precision_score(y_test,y_pred)

    return accuracy,precision
```

[77]:
```
train_classifier(svc,X_train,y_train,X_test,y_test)
```

[77]:
```
(0.9284332688588007, 0.753968253968254)
```

[78]:
```
accuracy_scores = []
precision_scores = []

for name,clf in clfs.items():

    current_accuracy,current_precision = train_classifier(clf,␣
 ↪X_train,y_train,X_test,y_test)
```

```
    print("For ",name)
    print("Accuracy - ",current_accuracy)
    print("Precision - ",current_precision)

    accuracy_scores.append(current_accuracy)
    precision_scores.append(current_precision)
```

```
For  SVC
Accuracy -  0.9284332688588007
Precision -  0.753968253968254
For  KN
Accuracy -  0.9110251450676983
Precision -  1.0
For  NB
Accuracy -  0.971953578336557
Precision -  0.8865248226950354
For  DT
Accuracy -  0.9245647969052224
Precision -  0.9166666666666666
For  LR
Accuracy -  0.9709864603481625
Precision -  0.9736842105263158
For  RF
Accuracy -  0.9738878143133463
Precision -  0.9826086956521739
For  AdaBoost
Accuracy -  0.9632495164410058
Precision -  0.9385964912280702
For  BgC
Accuracy -  0.9613152804642167
Precision -  0.9016393442622951
For  ETC
Accuracy -  0.9777562862669246
Precision -  0.9831932773109243
For  GBDT
Accuracy -  0.9448742746615088
Precision -  0.945054945054945
For  xgb
Accuracy -  0.9738878143133463
Precision -  0.9663865546218487
```

[79]:
```
performance_data = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy':
↪accuracy_scores,'Precision':precision_scores}).
↪sort_values('Precision',ascending=False)
```

[80]:
```
performance_data
```

[80]:

	Algorithm	Accuracy	Precision
1	KN	0.911025	1.000000
8	ETC	0.977756	0.983193
5	RF	0.973888	0.982609
4	LR	0.970986	0.973684
10	xgb	0.973888	0.966387
9	GBDT	0.944874	0.945055
6	AdaBoost	0.963250	0.938596
3	DT	0.924565	0.916667
7	BgC	0.961315	0.901639
2	NB	0.971954	0.886525
0	SVC	0.928433	0.753968

[82]:
```
performance_data1 = pd.melt(performance_data, id_vars = "Algorithm")
```

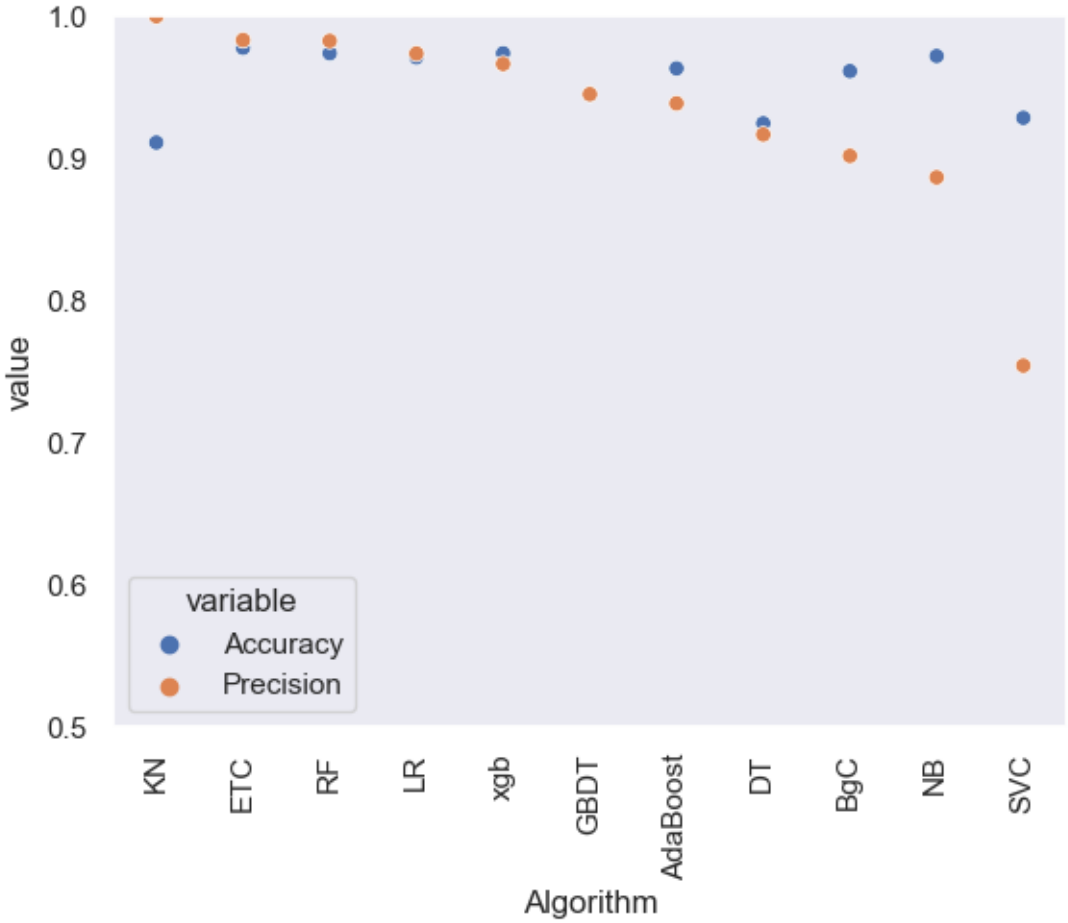
[83]:
```
performance_data1
```

[83]:

	Algorithm	variable	value
0	KN	Accuracy	0.911025
1	ETC	Accuracy	0.977756
2	RF	Accuracy	0.973888
3	LR	Accuracy	0.970986
4	xgb	Accuracy	0.973888
5	GBDT	Accuracy	0.944874
6	AdaBoost	Accuracy	0.963250
7	DT	Accuracy	0.924565
8	BgC	Accuracy	0.961315
9	NB	Accuracy	0.971954
10	SVC	Accuracy	0.928433
11	KN	Precision	1.000000
12	ETC	Precision	0.983193
13	RF	Precision	0.982609
14	LR	Precision	0.973684
15	xgb	Precision	0.966387
16	GBDT	Precision	0.945055
17	AdaBoost	Precision	0.938596
18	DT	Precision	0.916667
19	BgC	Precision	0.901639
20	NB	Precision	0.886525
21	SVC	Precision	0.753968

[85]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming performance_df1 is a DataFrame containing performance metrics
# and 'Algorithm' as one of the columns
```

```
# Plotting scatter plot
sns.scatterplot(x='Algorithm', y='value', hue='variable',␣
 ↪data=performance_data1)

# Adjusting y-axis limit
plt.ylim(0.5, 1.0)

# Rotating x-axis labels for better readability
plt.xticks(rotation='vertical')

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



[86]:
```
# model improve
# 1. Change the max_features parameter of TfIdf
```

[87]:
```
temp_data = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy_max_ft_3000':
 ↪accuracy_scores,'Precision_max_ft_3000':precision_scores}).
 ↪sort_values('Precision_max_ft_3000',ascending=False)
```

[88]:
```
temp_data = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy_scaling':
 ↪accuracy_scores,'Precision_scaling':precision_scores}).
 ↪sort_values('Precision_scaling',ascending=False)
```

[90]:
```
new_data = performance_data.merge(temp_data,on='Algorithm')
```

[91]:
```
new_data_scaled = new_data.merge(temp_data,on='Algorithm')
```

[92]:
```
temp_data = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy_num_chars':
 ↪accuracy_scores,'Precision_num_chars':precision_scores}).
 ↪sort_values('Precision_num_chars',ascending=False)
```

[93]:
```
new_data_scaled.merge(temp_data,on='Algorithm')
```

[93]:

	Algorithm	Accuracy	Precision	Accuracy_scaling_x	Precision_scaling_x	\
0	KN	0.911025	1.000000	0.911025	1.000000	
1	ETC	0.977756	0.983193	0.977756	0.983193	
2	RF	0.973888	0.982609	0.973888	0.982609	
3	LR	0.970986	0.973684	0.970986	0.973684	
4	xgb	0.973888	0.966387	0.973888	0.966387	
5	GBDT	0.944874	0.945055	0.944874	0.945055	
6	AdaBoost	0.963250	0.938596	0.963250	0.938596	
7	DT	0.924565	0.916667	0.924565	0.916667	
8	BgC	0.961315	0.901639	0.961315	0.901639	
9	NB	0.971954	0.886525	0.971954	0.886525	
10	SVC	0.928433	0.753968	0.928433	0.753968	

	Accuracy_scaling_y	Precision_scaling_y	Accuracy_num_chars	\
0	0.911025	1.000000	0.911025	
1	0.977756	0.983193	0.977756	
2	0.973888	0.982609	0.973888	
3	0.970986	0.973684	0.970986	
4	0.973888	0.966387	0.973888	
5	0.944874	0.945055	0.944874	
6	0.963250	0.938596	0.963250	
7	0.924565	0.916667	0.924565	
8	0.961315	0.901639	0.961315	
9	0.971954	0.886525	0.971954	
10	0.928433	0.753968	0.928433	

	Precision_num_chars
0	1.000000
1	0.983193

```
2           0.982609
3           0.973684
4           0.966387
5           0.945055
6           0.938596
7           0.916667
8           0.901639
9           0.886525
10          0.753968
```

[94]:
```
# Voting Classifier
svc = SVC(kernel='sigmoid', gamma=1.0,probability=True)
mnb = MultinomialNB()
etc = ExtraTreesClassifier(n_estimators=50, random_state=2)

from sklearn.ensemble import VotingClassifier
```

[95]:
```
voting = VotingClassifier(estimators=[('svm', svc), ('nb', mnb), ('et',␣
 ↪etc)],voting='soft').
```

[96]:
```
voting.fit(X_train,y_train)
```

[96]:
```
VotingClassifier(estimators=[('svm',
                              SVC(gamma=1.0, kernel='sigmoid',
                                  probability=True)),
                             ('nb', MultinomialNB()),
                             ('et',
                              ExtraTreesClassifier(n_estimators=50,
                                                   random_state=2))],
                 voting='soft')
```

[97]:
```
y_pred = voting.predict(X_test)
print("Accuracy",accuracy_score(y_test,y_pred))
print("Precision",precision_score(y_test,y_pred))
```

```
Accuracy 0.9796905222437138
Precision 0.968
```

[98]:
```
# Applying stacking
estimators=[('svm', svc), ('nb', mnb), ('et', etc)]
final_estimator=RandomForestClassifier()
```

[99]:
```
from sklearn.ensemble import StackingClassifier
clf = StackingClassifier(estimators=estimators, final_estimator=final_estimator)
```

[100]:
```
clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)
print("Accuracy",accuracy_score(y_test,y_pred))
```

```
print("Precision",precision_score(y_test,y_pred))
```

```
Accuracy 0.9835589941972921
Precision 0.9689922480620154
```

[101]:
```
import pickle
pickle.dump(tfidf,open('vectorizer.pkl','wb'))
pickle.dump(mnb,open('model.pkl','wb'))
```

[ ]:
```

```