In [20]:

```
# required libraries
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
from matplotlib.colors import LinearSegmentedColormap
import seaborn as sns

import re
import string

import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import SnowballStemmer
from nltk.corpus import wordnet
from nltk import pos_tag
from nltk.stem import WordNetLemmatizer

import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import SnowballStemmer
from nltk.corpus import wordnet
from nltk import pos_tag
from nltk.stem import WordNetLemmatizer

import spacy
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

from collections import Counter

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import cross_validate
from sklearn.model_selection import train_test_split
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_auc_score
from sklearn.utils import shuffle
from sklearn.naive_bayes import MultinomialNB

from xgboost import XGBClassifier

from transformers import pipeline

import warnings
warnings.filterwarnings('ignore')
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-20-919353ff1c25> in <module>
     27
     28 import spacy
---> 29 from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
     30
     31 from collections import Counter

ModuleNotFoundError: No module named 'wordcloud'
```

In [19]:

```
!pip install spacy
```

```
Collecting spacy
  Downloading spacy-3.4.1-cp38-cp38-win_amd64.whl (12.1 MB)
Collecting murmurhash<1.1.0,>=0.28.0
  Downloading murmurhash-1.0.8-cp38-cp38-win_amd64.whl (18 kB)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from
spacy) (2.25.1)
Collecting pathy>=0.3.5
  Downloading pathy-0.6.2-py3-none-any.whl (42 kB)
Requirement already satisfied: packaging>=20.0 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from spacy)
(21.3)
Collecting spacy-loggers<2.0.0,>=1.0.0
  Downloading spacy_loggers-1.0.3-py3-none-any.whl (9.3 kB)
Collecting thinc<8.2.0,>=8.1.0
  Downloading thinc-8.1.1-cp38-cp38-win_amd64.whl (1.3 MB)
Requirement already satisfied: setuptools in c:\users\rakesh lodem\anaconda3\lib\site-packages (from spacy) (52.0
.0.post20210125)
Requirement already satisfied: numpy>=1.15.0 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from spacy) (1
.20.1)
Collecting catalogue<2.1.0,>=2.0.6
```

```
  Downloading catalogue-2.0.8-py3-none-any.whl (17 kB)
Collecting typer<0.5.0,>=0.3.0
  Downloading typer-0.4.2-py3-none-any.whl (27 kB)
Requirement already satisfied: jinja2 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from spacy) (2.11.3)
Collecting langcodes<4.0.0,>=3.2.0
  Downloading langcodes-3.3.0-py3-none-any.whl (181 kB)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from spa
cy) (4.59.0)
Collecting preshed<3.1.0,>=3.0.2
  Downloading preshed-3.0.7-cp38-cp38-win_amd64.whl (96 kB)
Collecting spacy-legacy<3.1.0,>=3.0.9
  Downloading spacy_legacy-3.0.10-py2.py3-none-any.whl (21 kB)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<1.10.0,>=1.7.4 in c:\users\rakesh lodem\anaconda3\lib\site-
packages (from spacy) (1.9.2)
Collecting srsly<3.0.0,>=2.4.3
  Downloading srsly-2.4.4-cp38-cp38-win_amd64.whl (449 kB)
Collecting wasabi<1.1.0,>=0.9.1
  Downloading wasabi-0.10.1-py3-none-any.whl (26 kB)
Collecting cymem<2.1.0,>=2.0.2
  Downloading cymem-2.0.6-cp38-cp38-win_amd64.whl (36 kB)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\rakesh lodem\anaconda3\lib\site-packages (fro
m packaging>=20.0->spacy) (2.4.7)
Collecting smart-open<6.0.0,>=5.2.1
  Downloading smart_open-5.2.1-py3-none-any.whl (58 kB)
Requirement already satisfied: typing-extensions>=3.7.4.3 in c:\users\rakesh lodem\anaconda3\lib\site-packages (f
rom pydantic!=1.8,!=1.8.1,<1.10.0,>=1.7.4->spacy) (3.7.4.3)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from r
equests<3.0.0,>=2.13.0->spacy) (1.26.4)
Requirement already satisfied: chardet<5,>=3.0.2 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from reque
sts<3.0.0,>=2.13.0->spacy) (4.0.0)
Requirement already satisfied: idna<3,>=2.5 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from requests<3
.0.0,>=2.13.0->spacy) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from requ
ests<3.0.0,>=2.13.0->spacy) (2020.12.5)
Collecting confection<1.0.0,>=0.0.1
  Downloading confection-0.0.1-py3-none-any.whl (32 kB)
Collecting blis<0.10.0,>=0.7.8
  Downloading blis-0.9.1-cp38-cp38-win_amd64.whl (7.4 MB)
Requirement already satisfied: click<9.0.0,>=7.1.1 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from typ
er<0.5.0,>=0.3.0->spacy) (7.1.2)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from jinja2
->spacy) (1.1.1)
Installing collected packages: catalogue, srsly, murmurhash, cymem, wasabi, typer, smart-open, preshed, confectio
n, blis, thinc, spacy-loggers, spacy-legacy, pathy, langcodes, spacy
Successfully installed blis-0.9.1 catalogue-2.0.8 confection-0.0.1 cymem-2.0.6 langcodes-3.3.0 murmurhash-1.0.8 p
athy-0.6.2 preshed-3.0.7 smart-open-5.2.1 spacy-3.4.1 spacy-legacy-3.0.10 spacy-loggers-1.0.3 srsly-2.4.4 thinc-8
.1.1 typer-0.4.2 wasabi-0.10.1
```

In [21]:

```
df=pd.read_csv(r'C:\Users\RAKESH~1\AppData\Local\Temp\Rar$DIa18108.31450\1429_1.csv')
```

```
C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3165: DtypeWarning: Columns (1
,10) have mixed types.Specify dtype option on import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

In [22]:

```
df
```

Out[22]:

	id	name	asins	brand	categories	keys	manufacturer
0	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon
1	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon
2	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon

3	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon
4	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon
...	...	...	...	...	...	...	...
34655	AVpfiBlyLJeJML43-4Tp	NaN	B006GWO5WK	Amazon	Computers/Tablets & Networking,Tablet & eBook ...	newamazonkindlefirehd9wpowerfastadaptercharger...	Amazon Digital Services, Inc
34656	AVpfiBlyLJeJML43-4Tp	NaN	B006GWO5WK	Amazon	Computers/Tablets & Networking,Tablet & eBook ...	newamazonkindlefirehd9wpowerfastadaptercharger...	Amazon Digital Services, Inc
34657	AVpfiBlyLJeJML43-4Tp	NaN	B006GWO5WK	Amazon	Computers/Tablets & Networking,Tablet & eBook ...	newamazonkindlefirehd9wpowerfastadaptercharger...	Amazon Digital Services, Inc
34658	AVpfiBlyLJeJML43-4Tp	NaN	B006GWO5WK	Amazon	Computers/Tablets & Networking,Tablet & eBook ...	newamazonkindlefirehd9wpowerfastadaptercharger...	Amazon Digital Services, Inc
34659	AVpfiBlyLJeJML43-4Tp	NaN	B006GWO5WK	Amazon	Computers/Tablets & Networking,Tablet & eBook ...	newamazonkindlefirehd9wpowerfastadaptercharger...	Amazon Digital Services, Inc

34660 rows × 21 columns

In [23]:
```
df.shape
```

Out[23]:
```
(34660, 21)
```

In [24]:
```
## we have lot of unnecessary columns
```

In [25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34660 entries, 0 to 34659
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   id                    34660 non-null  object 
 1   name                  27900 non-null  object 
 2   asins                 34658 non-null  object 
 3   brand                 34660 non-null  object 
 4   categories            34660 non-null  object 
 5   keys                  34660 non-null  object 
 6   manufacturer          34660 non-null  object 
 7   reviews.date          34621 non-null  object 
 8   reviews.dateAdded     24039 non-null  object 
 9   reviews.dateSeen      34660 non-null  object 
 10  reviews.didPurchase   1 non-null      object 
 11  reviews.doRecommend   34066 non-null  object 
 12  reviews.id            1 non-null      float64
 13  reviews.numHelpful    34131 non-null  float64
 14  reviews.rating        34627 non-null  float64
 15  reviews.sourceURLs    34660 non-null  object 
 16  reviews.text          34659 non-null  object 
 17  reviews.title         34655 non-null  object 
 18  reviews.userCity      0 non-null      float64
 19  reviews.userProvince  0 non-null      float64
 20  reviews.username      34658 non-null  object 
dtypes: float64(5), object(16)
memory usage: 5.6+ MB
```

In [29]:
```
data = df[["reviews.text","reviews.rating"]].sample(20000,random_state=23)
data.head()
```

Out[29]:

	reviews.text	reviews.rating
21536	Bought as a Mother's Day Gift. This is great f...	4.0
20669	I can hold this next to my Kindle Paperwhite a...	5.0
30656	Love this device and went on to buy 2 as gifts...	5.0
25297	With some technical savvy, you can quickly hav...	5.0
9016	bought for grandkids they love them. wise choi...	5.0

In [30]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 20000 entries, 21536 to 22132
Data columns (total 2 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   reviews.text    20000 non-null  object 
 1   reviews.rating  19983 non-null  float64
dtypes: float64(1), object(1)
memory usage: 468.8+ KB
```

In [31]:
```
data.dropna(inplace=True)
```

In [32]:
```
data.isnull().sum()
```

Out[32]:
```
reviews.text      0
reviews.rating    0
dtype: int64
```

In [33]:
```
data.describe()
```

Out[33]:

	reviews.rating
count	19983.000000
mean	4.586899
std	0.735887
min	1.000000
25%	4.000000
50%	5.000000
75%	5.000000
max	5.000000

In [34]:
```
#distribution of rating
data['reviews.rating'].value_counts().sort_index(ascending=False)
```

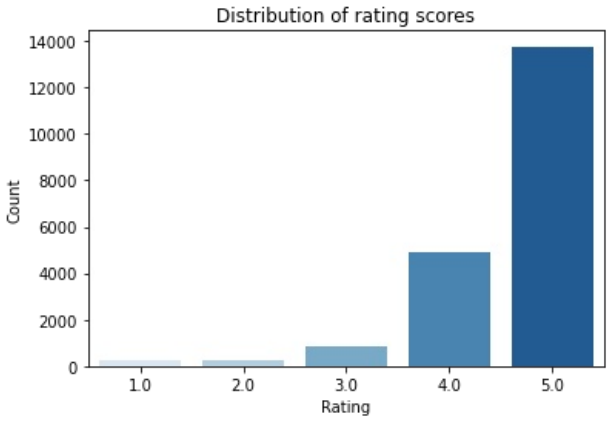
Out[34]:
```
5.0    13763
4.0     4898
3.0      849
2.0      233
1.0      240
Name: reviews.rating, dtype: int64
```

In [35]:
```
# distribution of rating
sns.countplot(data['reviews.rating'], palette='Blues')

plt.title('Distribution of rating scores')
```

```
plt.xlabel('Rating')
plt.ylabel('Count')
plt.show()
```

```
C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following va
riable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing oth
er arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [ ]:
```
##
```

In [36]:
```
data5 = data[data['reviews.rating']==5].sample(800,random_state=43)
data4 = data[data['reviews.rating']==4].sample(200,random_state=43)
data3 = data[data['reviews.rating']==3]
data2 = data[data['reviews.rating']==2]
data1 = data[data['reviews.rating']==1]

data = pd.concat([data5,data4,data3,data3,data2,data1])
```

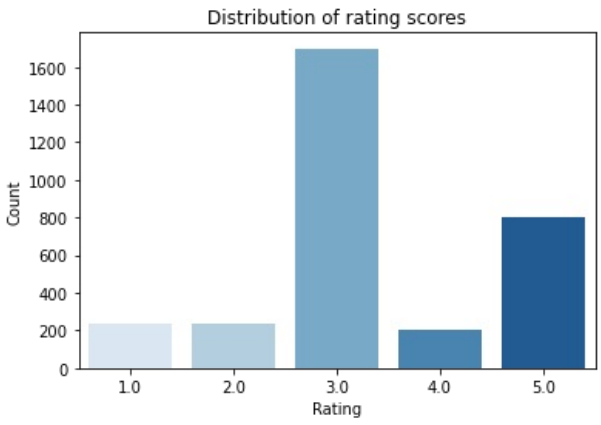
In [37]:
```
# distribution of rating
data['reviews.rating'].value_counts().sort_index(ascending=False)
```

Out[37]:
```
5.0     800
4.0     200
3.0    1698
2.0     233
1.0     240
Name: reviews.rating, dtype: int64
```

In [38]:
```
 #distribution of rating
sns.countplot(data['reviews.rating'], palette='Blues')

plt.title('Distribution of rating scores')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.show()
```

```
C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following va
riable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing oth
er arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [39]:
```
# map ratings 1, 2, 3 to 0 (NEGATIVE) and 4, 5 to 1 (POSITIVE)
sentiment_score = {1: 0,
                   2: 0,
                   3: 0,
                   4: 1,
                   5: 1}

sentiment = {0: 'NEGATIVE',
             1: 'POSITIVE'}


# mapping
data['sentiment_score'] = data['reviews.rating'].map(sentiment_score)
data['sentiment'] = data['sentiment_score'].map(sentiment)

data.head()
```

Out[39]:

	reviews.text	reviews.rating	sentiment_score	sentiment
31354	This is the best alternative to cable. Very ea...	5.0	1	POSITIVE
33776	For the price Amazon tv is the best everything...	5.0	1	POSITIVE
32587	The item works great wish the speeds were fast...	5.0	1	POSITIVE
29180	This is one of the coolest new inventions arou...	5.0	1	POSITIVE
2154	So much better than my previous kindle, I've u...	5.0	1	POSITIVE

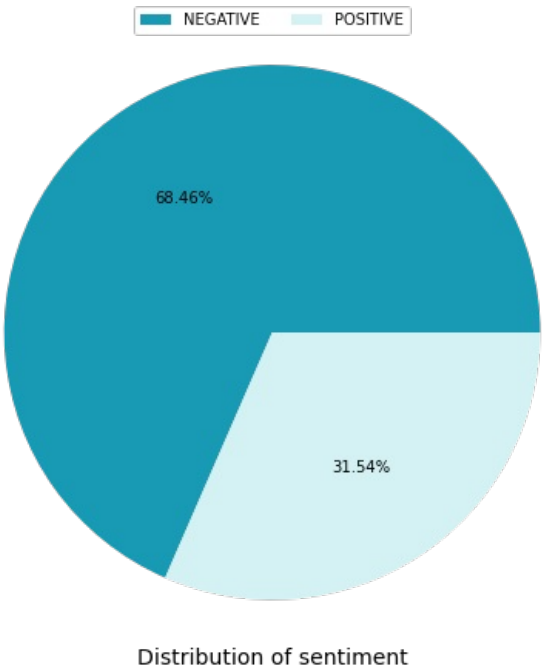
In [41]:
```
data['sentiment_score'].value_counts()
```

Out[41]:
```
0    2171
1    1000
Name: sentiment_score, dtype: int64
```

In [42]:
```
# distribution of sentiment
plt.figure(figsize = (8, 8))

labels = ['NEGATIVE','POSITIVE']
colors = ['#189AB4', '#D4F1F4']
plt.pie(data['sentiment'].value_counts(), autopct='%0.2f%%',colors=colors)

plt.title('Distribution of sentiment', size=14, y=-0.01)
plt.legend(labels, ncol=2, loc=9)
plt.show()
```



Distribution of sentiment

In [43]:
```
# get all used words
```

```
all_words = pd.Series(' '.join(data['reviews.text']).split())
```

In [ ]:

In [45]:
```
!pip install wordcloud
```

```
Collecting wordcloud
  Downloading wordcloud-1.8.2.2-cp38-cp38-win_amd64.whl (152 kB)
Requirement already satisfied: numpy>=1.6.1 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from wordcloud)
(1.20.1)
Requirement already satisfied: matplotlib in c:\users\rakesh lodem\anaconda3\lib\site-packages (from wordcloud) (
3.3.4)
Requirement already satisfied: pillow in c:\users\rakesh lodem\anaconda3\lib\site-packages (from wordcloud) (8.2.
0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from matpl
otlib->wordcloud) (1.3.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\rakesh lodem\anaconda3\lib\si
te-packages (from matplotlib->wordcloud) (2.4.7)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from ma
tplotlib->wordcloud) (2.8.1)
Requirement already satisfied: cycler>=0.10 in c:\users\rakesh lodem\anaconda3\lib\site-packages (from matplotlib
->wordcloud) (0.10.0)
Requirement already satisfied: six in c:\users\rakesh lodem\anaconda3\lib\site-packages (from cycler>=0.10->matpl
otlib->wordcloud) (1.15.0)
Installing collected packages: wordcloud
Successfully installed wordcloud-1.8.2.2
```

In [47]:
```
## text_preprocessing
```

In [48]:
```
def clean_text(text:str):
    """ Return cleaned text:
            - lowercase
            - remove whitespaces
            - remove HTML tags
            - replace digit with spaces
            - replace punctuations with spaces
            - remove extra spaces and tabs
        ------
        input: text (str)
        output: cleaned text (str)
    """
    text = str(text)

    text = text.lower()
    text = text.strip()

    text = re.sub(' \d+', ' ', text)
    text = re.compile('<.*?>').sub('', text)
    text = re.compile('[%s]' % re.escape(string.punctuation)).sub(' ', text)
    text = re.sub('\s+', ' ', text)

    text = text.strip()

    return text
```

In [49]:
```
# test
text = "   This is a message to be cleaned. It may involve some things like: <br>, ?, :, ''  26 adjacent spaces a
print(text, '\n')
clean_text(text)
```

```
   This is a message to be cleaned. It may involve some things like: <br>, ?, :, ''  26 adjacent spaces and tabs
.
```

Out[49]:
```
'this is a message to be cleaned it may involve some things like adjacent spaces and tabs'
```

In [55]:
```
## removing stopwords
```

In [56]:
```
def remove_stopwords(text:str):
    """ Remove stopwords from text:
        ------
        input: text (str)
        output: cleaned text (str)
```

```
    """
    text = str(text)
    filtered_sentence = []

    # Stop word lists can be adjusted for your problem
    stop_words = ["a", "an", "the", "this", "that", "is", "it", "to", "and"]

    # Tokenize the sentence
    words = word_tokenize(text)
    for w in words:
        if w not in stop_words:
            filtered_sentence.append(w)
    text = " ".join(filtered_sentence)

    return text
```

In [57]:

```
# test
text = "   This is a message to be cleaned. It may involve some things like: <br>, ?, :, ''  26 adjacent spaces a
print(text, '\n')
text = clean_text(text)
remove_stopwords(text)
```

```
   This is a message to be cleaned. It may involve some things like: <br>, ?, :, ''  26 adjacent spaces and tabs
.
```

Out[57]: 'message be cleaned may involve some things like adjacent spaces tabs'

In [58]:

```
## stemming
```

In [59]:

```
def stemm_text(text:str):
    """ Stemm text:
    ------
    input: text (str)
    output: Stemmed text (str)
    """
    text = str(text)
    # Initialize the stemmer
    snow = SnowballStemmer('english')

    stemmed_sentence = []
    # Tokenize the sentence
    words = word_tokenize(text)
    for w in words:
        # Stem the word/token
        stemmed_sentence.append(snow.stem(w))
    text = " ".join(stemmed_sentence)

    return text
```

In [60]:

```
# test
text = "   This is a message to be cleaned. It may involve some things like: <br>, ?, :, ''  26 adjacent spaces a
print(text, '\n')
text = clean_text(text)
text = remove_stopwords(text)
stemm_text(text)
```

```
   This is a message to be cleaned. It may involve some things like: <br>, ?, :, ''  26 adjacent spaces and tabs
.
```

Out[60]: 'messag be clean may involv some thing like adjac space tab'

In [61]:

```
##Lemmatization
```

In [63]:

```
def get_wordnet_pos(tag):
    if tag.startswith('J'):
        return wordnet.ADJ
    elif tag.startswith('V'):
        return wordnet.VERB
    elif tag.startswith('N'):
        return wordnet.NOUN
    elif tag.startswith('R'):
        return wordnet.ADV
```

```
    else:
        return wordnet.NOUN
```

In [64]:
```
import nltk
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\Rakesh Lodem\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger.zip.
```

Out[64]: True

In [65]:
```
def lemmatize(text:str):
    """ lemmatize text:
    ------
    input: text (str)
    output: lemmatized text (str)
    """
    text = str(text)

    # Initialize the lemmatizer
    wl = WordNetLemmatizer()

    lemmatized_sentence = []

    # Tokenize the sentence
    words = word_tokenize(text)
    # Get position tags
    word_pos_tags = nltk.pos_tag(words)
    # Map the position tag and lemmatize the word/token
    for idx, tag in enumerate(word_pos_tags):
        lemmatized_sentence.append(wl.lemmatize(tag[0], get_wordnet_pos(tag[1])))

    lemmatized_text = " ".join(lemmatized_sentence)

    return lemmatized_text
```

In [66]:
```
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package omw-1.4 to C:\Users\Rakesh
[nltk_data]     Lodem\AppData\Roaming\nltk_data...
```

Out[66]: True

In [67]:
```
# test
text = "   This is a message to be cleaned. It may involve some things like: <br>, ?, :, ''  26 adjacent spaces a
print(text, '\n')
text = clean_text(text)
text = remove_stopwords(text)
# text = stemm_text(text)
lemmatize(text)
```

```
   This is a message to be cleaned. It may involve some things like: <br>, ?, :, ''  26 adjacent spaces and tabs
.
```

Out[67]: 'message be clean may involve some thing like adjacent space tabs'

In [68]:
```
## applying text processing functions
```

In [69]:
```
# clean text
data['text'] = data['reviews.text'].apply(clean_text)
# remove stopwords
data['text'] = data['text'].apply(remove_stopwords)
# lemmatize
data['text'] = data['text'].apply(lemmatize)
```

In [70]:
```
# check some processed reviews
import random
```

```
i = random.choice(range(len(data)))

print(f"Original review: \n{data['reviews.text'].iloc[i]}\n")
print(f"Processed review: \n{data['text'].iloc[i]}")
```

```
Original review:
Alexa is a great device. The only drawback is not being able to simultaneously stream all devices throughout the
house. Hopefully Amazon is working on a solution

Processed review:
alexa great device only drawback not be able simultaneously stream all device throughout house hopefully amazon w
ork on solution
```

In [71]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3171 entries, 31354 to 6645
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   reviews.text     3171 non-null   object
 1   reviews.rating   3171 non-null   float64
 2   sentiment_score  3171 non-null   int64
 3   sentiment        3171 non-null   object
 4   text             3171 non-null   object
dtypes: float64(1), int64(1), object(3)
memory usage: 213.2+ KB
```

In [72]:
```
data.shape
```

Out[72]:
```
(3171, 5)
```

In [73]:
```
data.dropna(inplace=True)
```

In [74]:
```
data.isnull().sum()
```

Out[74]:
```
reviews.text       0
reviews.rating     0
sentiment_score    0
sentiment          0
text               0
dtype: int64
```

In [ ]:
```

```

In [75]:
```
## feature extraction
```

In [77]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [78]:
```
vectorizer = TfidfVectorizer(max_features=700)
vectorizer.fit(data['text'])
features = vectorizer.transform(data['text'])

features.toarray()
```

Out[78]:
```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

In [79]:
```

```

In [79]:
```
tf_idf = pd.DataFrame(features.toarray(), columns=vectorizer.get_feature_names())
# tf_idf.drop('50', axis=1, inplace=True)
tf_idf.head()
```

```
C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning: Function get_feature_names is deprecated; get_feature_names is deprecated in 1.0 and will be removed in 1.2. Please use get_feature_names_out instead.
  warnings.warn(msg, category=FutureWarning)
```

Out[79]:

	10	40	50	99	ability	able	about	absolutely	access	account	...	wouldn	write	wrong	year	yet	you	young	your	youtube	y
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.155222	0.0	0.0	0.0	0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0

5 rows × 700 columns

In [82]:
```
from sklearn.model_selection import train_test_split
```

In [89]:
```
X_train, X_test, y_train, y_test = train_test_split(tf_idf, data['sentiment_score'], test_size=0.2, random_state=

print (f'Train set shape\t:{X_train.shape}\nTest set shape\t:{X_test.shape}')
```

```
Train set shape :(2536, 700)
Test set shape  :(635, 700)
```

In [90]:
```
X_train
```

Out[90]:

	10	40	50	99	ability	able	about	absolutely	access	account	...	wouldn	write	wrong	year	yet	you	young	your
2597	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000
809	0.0	0.0	0.0	0.0	0.0	0.0	0.117085	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000
2498	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000
3014	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000
2244	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.178183	0.0	0.0	0.0	0.0	0.088565	0.0	0.131633
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3092	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000
1095	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000
1130	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000
1294	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000
860	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	...	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000

2536 rows × 700 columns

In [91]:
```
X_test
```

Out[91]:

	10	40	50	99	ability	able	about	absolutely	access	account	...	wouldn	write	wrong	year	yet	you	young	your	you
254	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	
2171	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.121229	0.0	0.000000	
969	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	
940	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.120284	0.0	0.000000	
331	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.182141	0.0	0.270714	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
789	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	
3010	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	
1080	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	
533	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	

859	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000

635 rows × 700 columns

In [104…]
```
from sklearn.metrics import classification_report,accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_auc_score
```

In [105…]
```
##MOdels
```

In [106…]
```
def modeling(Model, Xtrain = X_train, Xtest = X_test):
    """
    This function apply countVectorizer with machine learning algorithms.
    """

    # Instantiate the classifier: model
    model = Model

    # Fitting classifier to the Training set (all features)
    model.fit(Xtrain, y_train)

    global y_pred
    # Predicting the Test set results
    y_pred = model.predict(Xtest)

    # Assign f1 score to a variable
    print(classification_report(y_test, y_pred))
    print ('AUC ',roc_auc_score(y_test, y_pred))
    #cm = confusion_matrix(y_test, y_pred)
    confusion_matrix = pd.crosstab(index=y_test, columns=np.round(y_pred), rownames=['Actual'], colnames=['Predic
    plt.figure(figsize = (8,8))

    '''
    cmapGR = LinearSegmentedColormap.from_list(
        name='test',
        colors=['red','green']
    )
    '''
    sns.heatmap(confusion_matrix, annot=True,annot_kws={"fontsize":12}, fmt='.2f', cmap='Blues').set_title('Confu
```

In [107…]
```
from sklearn.naive_bayes import MultinomialNB
```

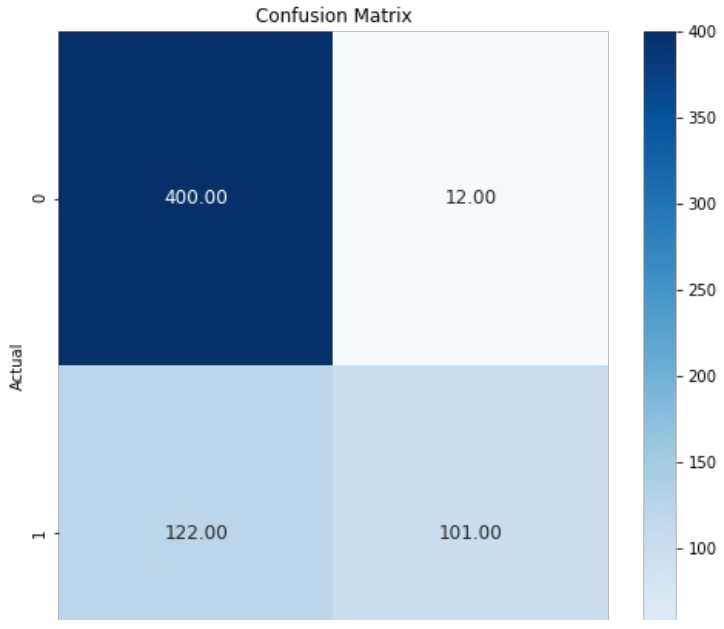
In [108…]
```
modeling(MultinomialNB())
```

```
              precision    recall  f1-score   support

           0       0.77      0.97      0.86       412
           1       0.89      0.45      0.60       223

    accuracy                           0.79       635
   macro avg       0.83      0.71      0.73       635
weighted avg       0.81      0.79      0.77       635

AUC  0.7118942923070225
```



In [109…
```
accuracy_score(y_test,y_pred)
```

Out[109…
```
0.7889763779527559
```

In [113…
```
!pip install xgboost
```

```
Collecting xgboost
  Downloading xgboost-1.6.2-py3-none-win_amd64.whl (125.4 MB)
Requirement already satisfied: scipy in c:\users\rakesh lodem\anaconda3\lib\site-packages (from xgboost) (1.6.2)
Requirement already satisfied: numpy in c:\users\rakesh lodem\anaconda3\lib\site-packages (from xgboost) (1.20.1)
Installing collected packages: xgboost
Successfully installed xgboost-1.6.2
```

In [114…
```
from xgboost import XGBClassifier
```

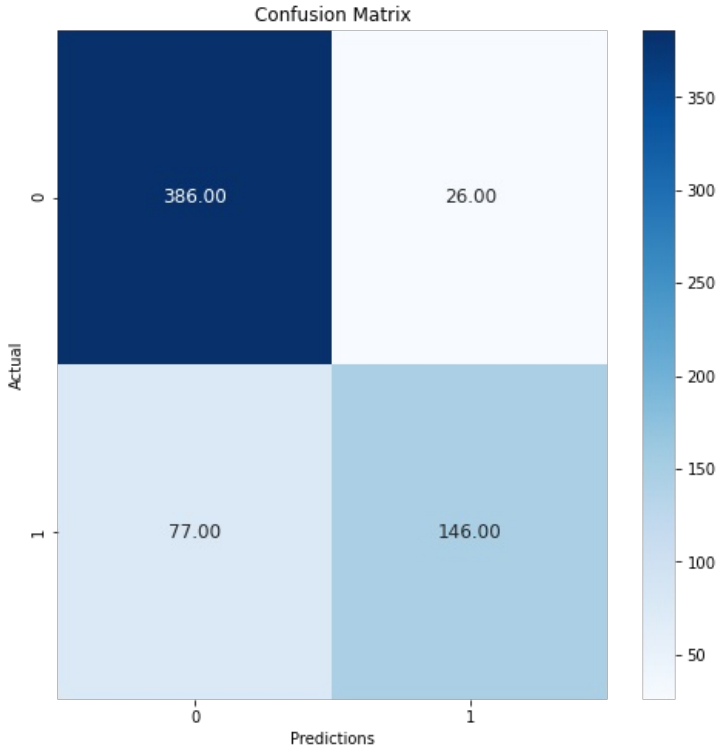
In [115…
```
modeling(XGBClassifier());
```

```
              precision    recall  f1-score   support

           0       0.83      0.94      0.88       412
           1       0.85      0.65      0.74       223

    accuracy                           0.84       635
   macro avg       0.84      0.80      0.81       635
weighted avg       0.84      0.84      0.83       635

AUC  0.7958008620314337
```



In [116…
```
accuracy_score(y_test,y_pred)
```

Out[116…
```
0.8377952755905512
```

In [118…
```
from sklearn.tree import DecisionTreeClassifier
```

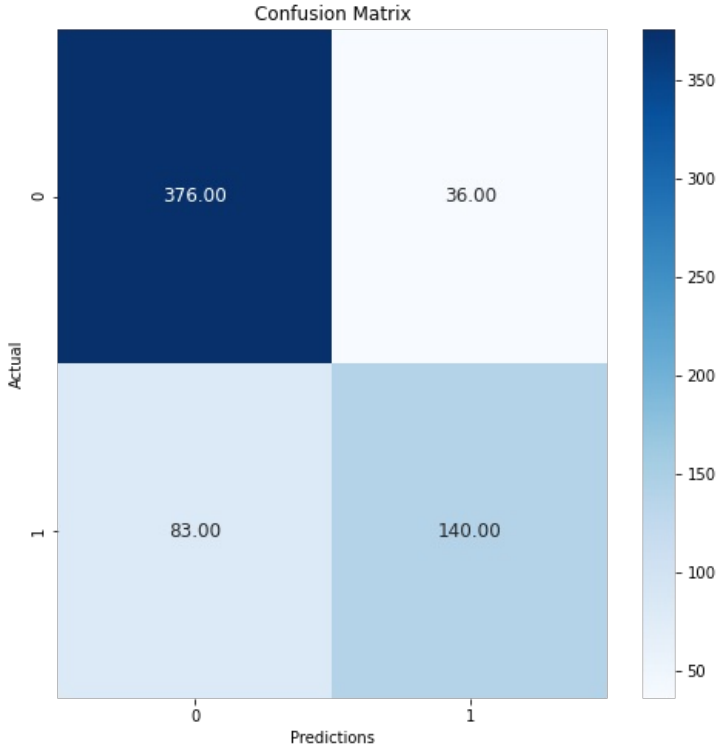
In [120…
```
modeling(DecisionTreeClassifier());
```

```
              precision    recall  f1-score   support

           0       0.82      0.91      0.86       412
           1       0.80      0.63      0.70       223

    accuracy                           0.81       635
   macro avg       0.81      0.77      0.78       635
weighted avg       0.81      0.81      0.81       635

AUC  0.7702120249031303
```



In [121…
```
accuracy_score(y_test,y_pred)
```

Out[121… 0.8125984251968504

In [129…
```
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
```

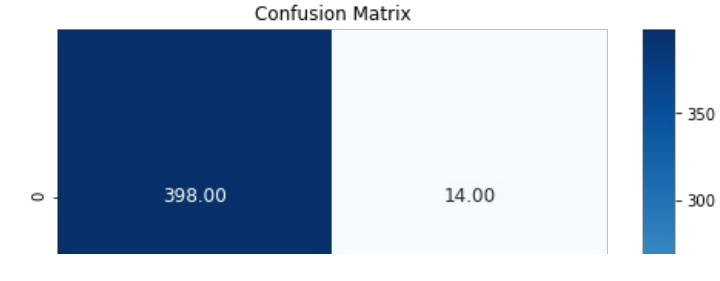
In [130…
```
modeling(RandomForestClassifier());
```

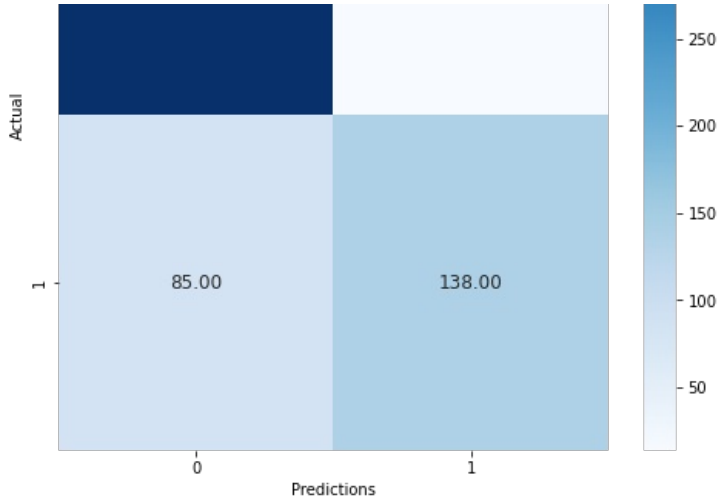
```
              precision    recall  f1-score   support

           0       0.82      0.97      0.89       412
           1       0.91      0.62      0.74       223

    accuracy                           0.84       635
   macro avg       0.87      0.79      0.81       635
weighted avg       0.85      0.84      0.84       635

AUC  0.7924267490966085
```





In [131…
```
accuracy_score(y_test,y_pred)
```

Out[131… 0.8440944881889764

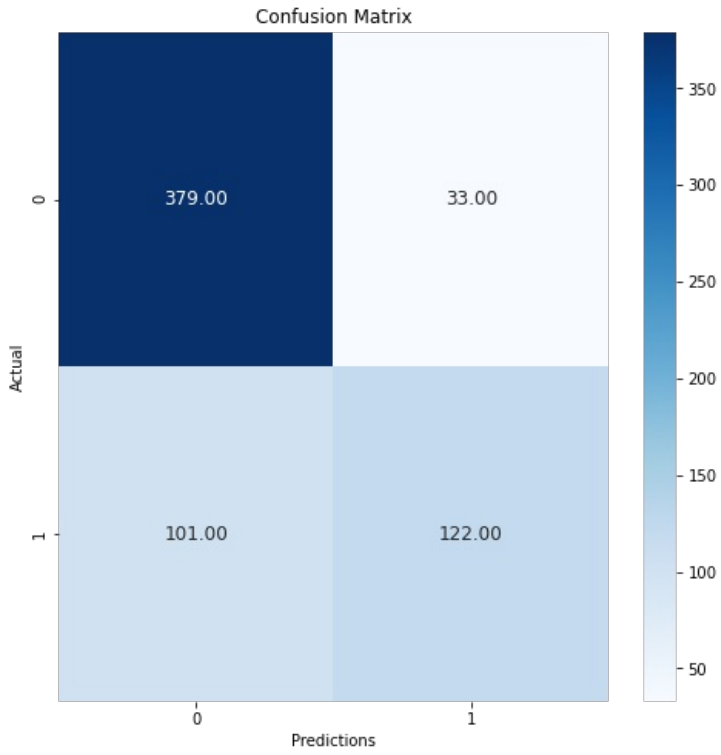
In [132…
```
modeling(GradientBoostingClassifier());
```

```
              precision    recall  f1-score   support

           0       0.79      0.92      0.85       412
           1       0.79      0.55      0.65       223

    accuracy                           0.79       635
   macro avg       0.79      0.73      0.75       635
weighted avg       0.79      0.79      0.78       635

AUC  0.7334940572075406
```



In [133…
```
accuracy_score(y_pred,y_test)
```

Out[133… 0.7889763779527559

In [134…]
```
## RandomForestClassifier is working better
```

In [135…]
```
params={'n_estimators':[100, 300, 500, 700],
        'min_samples_split':[1,2,3,4],
        'min_samples_leaf':[1,2,3,4],
        'max_depth':[None,1,2,3,4,5,6,7,8,9,10,15,20,25,30,35,40]}
```

In [136…]
```
from sklearn.model_selection import RandomizedSearchCV
```

In [137…]
```
g=RandomizedSearchCV(RandomForestClassifier(),params,cv=10)
```

In [139…]
```
g.fit(X_train,y_train)
```

```
C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py:378: FitFailedWarning:
40 fits failed out of a total of 100.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
40 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 686, in _
fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\ensemble\_forest.py", line 476, in fit
    trees = Parallel(
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\joblib\parallel.py", line 1043, in __call__
    if self.dispatch_one_batch(iterator):
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\joblib\parallel.py", line 861, in dispatch_one_batch
    self._dispatch(tasks)
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\joblib\parallel.py", line 779, in _dispatch
    job = self._backend.apply_async(batch, callback=cb)
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\joblib\_parallel_backends.py", line 208, in apply_async
    result = ImmediateResult(func)
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\joblib\_parallel_backends.py", line 572, in __init__
    self.results = batch()
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\joblib\parallel.py", line 262, in __call__
    return [func(*args, **kwargs)
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\joblib\parallel.py", line 262, in <listcomp>
    return [func(*args, **kwargs)
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\utils\fixes.py", line 117, in __call__
    return self.function(*args, **kwargs)
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\ensemble\_forest.py", line 189, in _parallel_bu
ild_trees
    tree.fit(X, y, sample_weight=curr_sample_weight, check_input=False)
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\tree\_classes.py", line 969, in fit
    super().fit(
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\tree\_classes.py", line 265, in fit
    check_scalar(
  File "C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\utils\validation.py", line 1480, in check_scala
r
    raise ValueError(
ValueError: min_samples_split == 1, must be >= 2.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\Rakesh Lodem\anaconda3\lib\site-packages\sklearn\model_selection\_search.py:953: UserWarning: One or mor
e of the test scores are non-finite: [0.69361364        nan 0.69361364 0.80166817 0.72517351        nan
        nan        nan 0.73936697 0.6940089 ]
  warnings.warn(
```

Out[139…]

In [140…]
```
print(g.best_estimator_)
print(g.best_params_)
print(g.best_score_)
```

```
RandomForestClassifier(max_depth=25, min_samples_leaf=4, min_samples_split=3,
                       n_estimators=500)
```

```
{'n_estimators': 500, 'min_samples_split': 3, 'min_samples_leaf': 4, 'max_depth': 25}
0.8016681709252745
```

In [141…
```
m=RandomForestClassifier(max_depth=25, min_samples_leaf=4, min_samples_split=3,n_estimators=500)
m.fit(X_train,y_train)
p=m.predict(X_test)
```

In [144…
```
print('Accuracy',np.round(accuracy_score(p,y_test),4))
print('------------------------------------------------------------')
print('Confusion Matrix')
print(confusion_matrix(p,y_test))
print('------------------------------------------------------------')
print('Classification Report')
print(classification_report(p,y_test))
```

```
Accuracy 0.7843
------------------------------------------------------------
Confusion Matrix
[[393 118]
 [ 19 105]]
------------------------------------------------------------
Classification Report
              precision    recall  f1-score   support

           0       0.95      0.77      0.85       511
           1       0.47      0.85      0.61       124

    accuracy                           0.78       635
   macro avg       0.71      0.81      0.73       635
weighted avg       0.86      0.78      0.80       635
```

In [145…
```
accuracy_score(y_test,y_pred)
```

Out[145… 0.7889763779527559

In [147…
```
## saving the model
```

In [148…
```
import pickle
```

In [150…
```
filename='ratings prediction.pkl'
```

In [151…
```
pickle.dump(m,open(filename,'wb'))
```

In [152…
```
## prediction
```

In [153…
```
import numpy as np
```

In [154…
```
y_test=np.array(y_test)
```

In [155…
```
y_test
```

Out[155…
```
array([1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0,
       1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
       1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0,
       0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1,
       0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1,
```

```
       1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,
       0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
       0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0],
      dtype=int64)
```

In [160…

```
pred=np.array(m.predict(X_test))
pred
```

Out[160…

```
array([0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
       0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
      dtype=int64)
```

In [164…

```
df_com=pd.DataFrame({'predicted':pred,'actual':y_test},index=range(len(y_test)))
df_com
```

Out[164…

	predicted	actual
0	0	1
1	0	0
2	1	1
3	0	1
4	1	1
...	...	...
630	1	1
631	0	0
632	0	0
633	0	1
634	0	1

635 rows × 2 columns

In [ ]:

In [ ]: