Double-click (or enter) to edit

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

```
df=pd.read_csv('/content/twitter_validation.csv',header=None,encoding="ISO-8859-1")
df.columns=['ID','SOC_MEDIA','STATUS','REVIEW']
df
```

	ID	SOC_MEDIA	STATUS	REVIEW
0	3364	Facebook	Irrelevant	I mentioned on Facebook that I was struggling ...
1	352	Amazon	Neutral	BBC News - Amazon boss Jeff Bezos rejects clai...
2	8312	Microsoft	Negative	@Microsoft Why do I pay for WORD when it funct...
3	4371	CS-GO	Negative	CSGO matchmaking is so full of closet hacking,...
4	4433	Google	Neutral	Now the President is slapping Americans in the...
...	...	...	...	...
995	4891	GrandTheftAuto(GTA)	Irrelevant	â□ï¸□ Toronto is the arts and culture capital...
996	4359	CS-GO	Irrelevant	tHIS IS ACTUALLY A GOOD MOVE TOT BRING MORE VI...
997	2652	Borderlands	Positive	Today sucked so itâ□□s time to drink wine n pl...
998	8069	Microsoft	Positive	Bought a fraction of Microsoft today. Small wins.
999	6960	johnson&johnson	Neutral	Johnson & Johnson to stop selling talc baby po...

```
df.head()
```

	ID	SOC_MEDIA	STATUS	REVIEW
0	3364	Facebook	Irrelevant	I mentioned on Facebook that I was struggling ...
1	352	Amazon	Neutral	BBC News - Amazon boss Jeff Bezos rejects clai...
2	8312	Microsoft	Negative	@Microsoft Why do I pay for WORD when it funct...
3	4371	CS-GO	Negative	CSGO matchmaking is so full of closet hacking,...
4	4433	Google	Neutral	Now the President is slapping Americans in the...

```
df.tail()
```

	ID	SOC_MEDIA	STATUS	REVIEW
995	4891	GrandTheftAuto(GTA)	Irrelevant	â□ï¸□ Toronto is the arts and culture capital...
996	4359	CS-GO	Irrelevant	tHIS IS ACTUALLY A GOOD MOVE TOT BRING MORE VI...
997	2652	Borderlands	Positive	Today sucked so itâ□□s time to drink wine n pl...
998	8069	Microsoft	Positive	Bought a fraction of Microsoft today. Small wins.
999	6960	johnson&johnson	Neutral	Johnson & Johnson to stop selling talc baby po...

```
df.shape
```

```
(1000, 4)
```

```
df.isna().sum()
```

```
ID           0
SOC_MEDIA    0
STATUS       0
REVIEW       0
dtype: int64
```

```
df.dtypes
```

```
ID            int64
SOC_MEDIA    object
STATUS       object
REVIEW       object
dtype: object
```

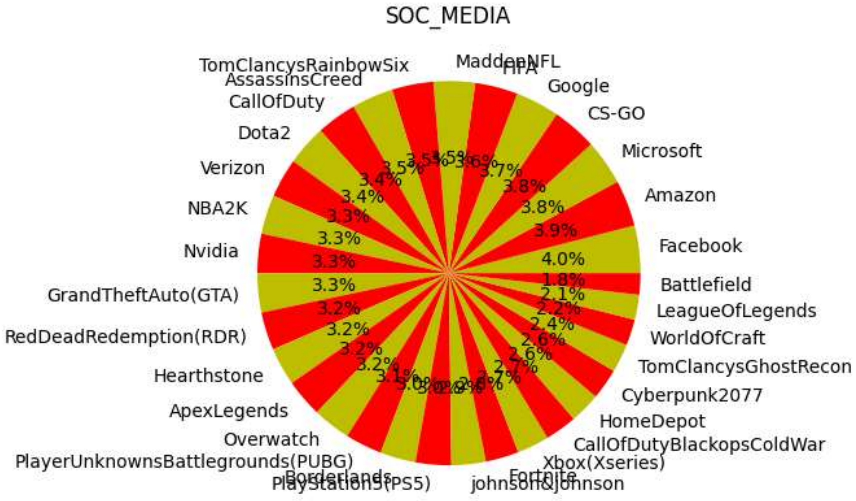
```
a=df['SOC_MEDIA'].value_counts()
a
```

```
SOC_MEDIA
RedDeadRedemption(RDR)               40
johnson&johnson                      39
FIFA                                 38
PlayerUnknownsBattlegrounds(PUBG)    38
LeagueOfLegends                      37
ApexLegends                          36
TomClancysRainbowSix                 35
Nvidia                               35
GrandTheftAuto(GTA)                  35
Amazon                               34
Fortnite                             34
Facebook                             33
PlayStation5(PS5)                    33
AssassinsCreed                       33
Borderlands                          33
Overwatch                            32
Hearthstone                          32
Verizon                              32
CS-GO                                32
CallOfDuty                           31
Cyberpunk2077                        30
WorldOfCraft                         30
MaddenNFL                            29
Microsoft                            28
Dota2                                27
CallOfDutyBlackopsColdWar            27
Xbox(Xseries)                        26
Battlefield                          26
Google                               24
TomClancysGhostRecon                 22
NBA2K                                21
HomeDepot                            18
Name: count, dtype: int64
```

```
a1=df['SOC_MEDIA'].unique()
```

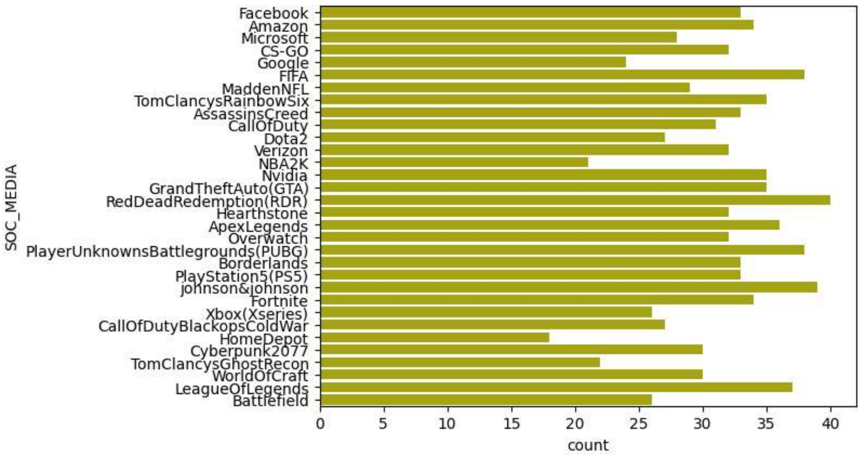
```
plt.pie(a,labels=a1,autopct='%.1f%%',colors=['y','r'])
plt.title("SOC_MEDIA")
```

```
Text(0.5, 1.0, 'SOC_MEDIA')
```



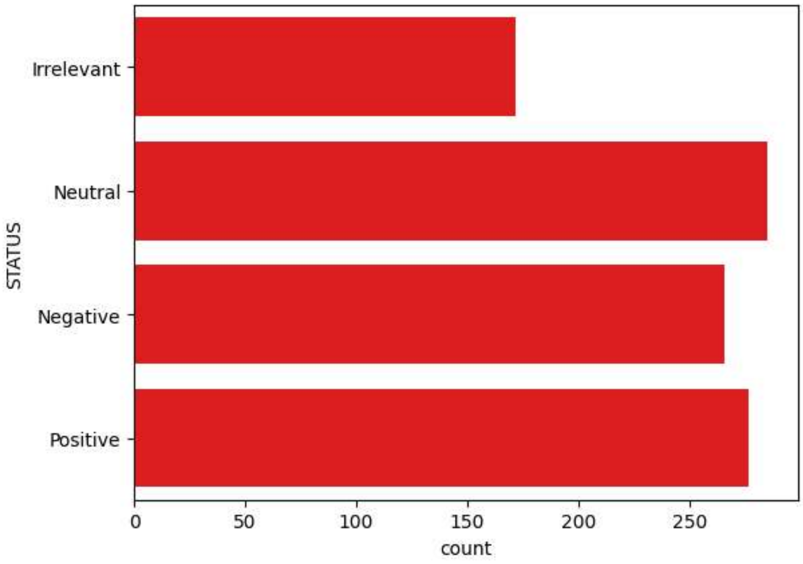
```
sns.countplot(y='SOC_MEDIA',data=df,color='y')
```

```
<Axes: xlabel='count', ylabel='SOC_MEDIA'>
```



```
sns.countplot(y='STATUS',data=df,color='r')
```

```
<Axes: xlabel='count', ylabel='STATUS'>
```



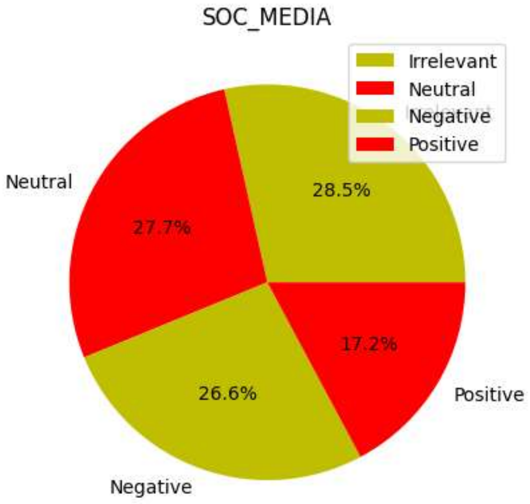
```
b=df['STATUS'].value_counts()
b
```

```
STATUS
Neutral        285
Positive       277
Negative       266
Irrelevant     172
Name: count, dtype: int64
```

```
b1=df['STATUS'].unique()
```

```
plt.pie(b,labels=b1,autopct='%.1f%%',colors=['y','r'])
plt.legend()
plt.title("SOC_MEDIA")
```

```
Text(0.5, 1.0, 'SOC_MEDIA')
```



```
# drop irrelevant
df.drop(df.index[(df['STATUS']=='Irrelevant')],axis=0,inplace=True)
df.shape
```

```
(828, 4)
```

```
# to correct the index
df.reset_index(drop=True,inplace=True)
df
```

	ID	SOC_MEDIA	STATUS	REVIEW
0	352	Amazon	Neutral	BBC News - Amazon boss Jeff Bezos rejects clai...
1	8312	Microsoft	Negative	@Microsoft Why do I pay for WORD when it funct...
2	4371	CS-GO	Negative	CSGO matchmaking is so full of closet hacking,...
3	4433	Google	Neutral	Now the President is slapping Americans in the...
4	6273	FIFA	Negative	Hi @EAHelp lâ□□ve had Madeleine McCann in my c...
...	...	...	...	...
823	314	Amazon	Negative	Please explain how this is possible! How can t...
824	9701	PlayStation5(PS5)	Positive	Good on Sony. As much as I want to see the new...
825	2652	Borderlands	Positive	Today sucked so itâ□□s time to drink wine n pl...
826	8069	Microsoft	Positive	Bought a fraction of Microsoft today. Small wins.
827	6960	johnson&johnson	Neutral	Johnson & Johnson to stop selling talc baby po...

828 rows × 4 columns

```
df.drop(['ID','SOC_MEDIA'],axis=1,inplace=True)
df
```

	STATUS	REVIEW
0	Neutral	BBC News - Amazon boss Jeff Bezos rejects clai...
1	Negative	@Microsoft Why do I pay for WORD when it funct...
2	Negative	CSGO matchmaking is so full of closet hacking,...
3	Neutral	Now the President is slapping Americans in the...
4	Negative	Hi @EAHelp Iâ□□ve had Madeleine McCann in my c...
...	...	...
823	Negative	Please explain how this is possible! How can t...
824	Positive	Good on Sony. As much as I want to see the new...
825	Positive	Today sucked so itâ□□s time to drink wine n pl...
826	Positive	Bought a fraction of Microsoft today. Small wins.
827	Neutral	Johnson & Johnson to stop selling talc baby po...

828 rows × 2 columns

```
df['STATUS'] = df['STATUS'].map({'Positive': 1,'Negative': -1,'Neutral': 0})
df
```

	STATUS	REVIEW
0	0	BBC News - Amazon boss Jeff Bezos rejects clai...
1	-1	@Microsoft Why do I pay for WORD when it funct...
2	-1	CSGO matchmaking is so full of closet hacking,...
3	0	Now the President is slapping Americans in the...
4	-1	Hi @EAHelp Iâ□□ve had Madeleine McCann in my c...
...	...	...
823	-1	Please explain how this is possible! How can t...
824	1	Good on Sony. As much as I want to see the new...
825	1	Today sucked so itâ□□s time to drink wine n pl...
826	1	Bought a fraction of Microsoft today. Small wins.
827	0	Johnson & Johnson to stop selling talc baby po...

828 rows × 2 columns

```
nltk.download('wordnet')
nltk.download('stopwords')
nltk.download('punkt')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

```
# assign all text in the dataframe in a variable
tweets=df['REVIEW']
tweets
```

```
0      BBC News - Amazon boss Jeff Bezos rejects clai...
1      @Microsoft Why do I pay for WORD when it funct...
2      CSGO matchmaking is so full of closet hacking,...
3      Now the President is slapping Americans in the...
4      Hi @EAHelp Iâ□□ve had Madeleine McCann in my c...
                             ...
823    Please explain how this is possible! How can t...
824    Good on Sony. As much as I want to see the new...
825    Today sucked so itâ□□s time to drink wine n pl...
826    Bought a fraction of Microsoft today. Small wins.
827    Johnson & Johnson to stop selling talc baby po...
Name: REVIEW, Length: 828, dtype: object
```

```
from nltk.tokenize import word_tokenize
from nltk.tokenize import TweetTokenizer
tk=TweetTokenizer()
tweets=tweets.apply(lambda x:tk.tokenize(x)).apply(lambda x: ' '.join(x)) # second lambda is to join the tokens

tweets
```

```
0      BBC News - Amazon boss Jeff Bezos rejects clai...
1      @Microsoft Why do I pay for WORD when it funct...
2      CSGO matchmaking is so full of closet hacking ...
3      Now the President is slapping Americans in the...
4      Hi @EAHelp Iâ ⍰ ⍰ ve had Madeleine McCann in m...
                              ...
823    Please explain how this is possible ! How can ...
824    Good on Sony . As much as I want to see the ne...
825    Today sucked so itâ ⍰ ⍰ s time to drink wine n...
826    Bought a fraction of Microsoft today . Small w...
827    Johnson & Johnson to stop selling talc baby po...
Name: REVIEW, Length: 828, dtype: object
```

```
# remove special characters
import re
tweets=tweets.str.replace('[^a-zA-Z0-9]',' ',regex=True)
tweets
```

```
0      BBC News   Amazon boss Jeff Bezos rejects clai...
1       Microsoft Why do I pay for WORD when it funct...
2      CSGO matchmaking is so full of closet hacking ...
3      Now the President is slapping Americans in the...
4      Hi  EAHelp I      ve had Madeleine McCann in m...
                              ...
823    Please explain how this is possible   How can ...
824    Good on Sony   As much as I want to see the ne...
825    Today sucked so it      s time to drink wine n...
826    Bought a fraction of Microsoft today   Small w...
827    Johnson   Johnson to stop selling talc baby po...
Name: REVIEW, Length: 828, dtype: object
```

```
# remove the words having less than 3 characters
from nltk.tokenize import word_tokenize
tweets=tweets.apply(lambda x:' '.join((w for w in tk.tokenize(x) if len(w)>=3)))
tweets
```

```
0      BBC News Amazon boss Jeff Bezos rejects claims...
1      Microsoft Why pay for WORD when functions poor...
2      CSGO matchmaking full closet hacking truly awf...
3      Now the President slapping Americans the face ...
4      EAHelp had Madeleine McCann cellar for the pas...
                              ...
823    Please explain how this possible How can they ...
824    Good Sony much want see the new PS5 what going...
825    Today sucked time drink wine play borderlands ...
826           Bought fraction Microsoft today Small wins
827    Johnson Johnson stop selling talc baby powder ...
Name: REVIEW, Length: 828, dtype: object
```

```
from nltk.stem import SnowballStemmer
stm=SnowballStemmer('english')
tweets = tweets.apply(lambda x: [stm.stem(i.lower()) for i in tk.tokenize(x)]).apply(lambda x: ' '.join(x))
# to remove the tail and convert it into lowercase
tweets
```

```
0      bbc news amazon boss jeff bezo reject claim co...
1      microsoft whi pay for word when function poor ...
2           csgo matchmak full closet hack truli aw game
3      now the presid slap american the face that rea...
4      eahelp had madelein mccann cellar for the past...
                              ...
823    pleas explain how this possibl how can they le...
824    good soni much want see the new ps5 what go ri...
825    today suck time drink wine play borderland unt...
826            bought fraction microsoft today small win
827    johnson johnson stop sell talc babi powder and...
Name: REVIEW, Length: 828, dtype: object
```

```
# remove stop words
from nltk.corpus import stopwords
data=stopwords.words('english')
tweets = tweets.apply(lambda x: [stm.stem(i.lower()) for i in tk.tokenize(x) if i.lower() not in data]).apply(lambda x: ' '.join(x
tweets
```

```
0      bbc news amazon boss jeff bezo reject claim co...
1      microsoft whi pay word function poor samsungus...
2           csgo matchmak full closet hack truli aw game
3      presid slap american face realli commit unlaw ...
4      eahelp madelein mccann cellar past year littl ...
                             ...
823    plea explain possibl let compani overcharg sca...
824    good soni much want see new ps5 go right much ...
825    today suck time drink wine play borderland sun...
826            bought fraction microsoft today small win
827    johnson johnson stop sell talc babi powder can...
Name: REVIEW, Length: 828, dtype: object
```

```
# vectorization
# we use the method TFIDF method
from sklearn.feature_extraction.text import TfidfVectorizer
vec=TfidfVectorizer()
data=vec.fit_transform(tweets)
data
```

```
<828x3759 sparse matrix of type '<class 'numpy.float64'>'
        with 10459 stored elements in Compressed Sparse Row format>
```

```
print(data)
```

```
  (0, 668)      0.2608257828483461
  (0, 981)      0.2608257828483461
  (0, 1107)     0.23509805002803952
  (0, 1974)     0.13277165480466424
  (0, 286)      0.22681557001542715
  (0, 838)      0.17354914655342313
  (0, 785)      0.21432663830218204
  (0, 2737)     0.2608257828483461
  (0, 545)      0.2608257828483461
  (0, 1811)     0.24577602391989378
  (0, 610)      0.22681557001542715
  (0, 353)      0.1515362387424402
  (0, 2264)     0.38864111655856126
  (0, 515)      0.49155204783978756
  (1, 775)      0.4055823664694651
  (1, 2867)     0.4055823664694651
  (1, 2534)     0.3821800909185634
  (1, 1382)     0.4055823664694651
  (1, 3656)     0.36557591217188057
  (1, 2438)     0.3126902562590763
  (1, 3615)     0.26216072802580975
  (1, 2132)     0.24555654927912696
  (2, 1404)     0.1689251539717079
  (2, 463)      0.36574263611909275
  (2, 3408)     0.36574263611909275
  :     :
  (825, 3369)   0.3395996844494919
  (825, 3359)   0.2560582225152134
  (825, 608)    0.22981061112100945
  (825, 974)    0.2315686698425631
  (825, 3349)   0.21750175079084832
  (825, 3202)   0.2904718522758868
  (825, 2503)   0.17148706662740873
  (826, 1358)   0.5079831062080814
  (826, 3046)   0.47867226429410115
  (826, 613)    0.4174215841659411
  (826, 3626)   0.353278941165688
  (826, 3359)   0.34523850330234374
  (826, 2132)   0.3075542453642147
  (827, 182)    0.3283693467320579
  (827, 1109)   0.3283693467320579
  (827, 2784)   0.3283693467320579
  (827, 134)    0.3283693467320579
  (827, 689)    0.2770320970909926
  (827, 2922)   0.2635889502019104
  (827, 3250)   0.2635889502019104
  (827, 3169)   0.2311987519368367
  (827, 1665)   0.15963411936668057
  (827, 2552)   0.24870786898500463
  (827, 483)    0.23743856420618148
  (827, 1832)   0.3947412386878786
```

```
data.shape
```

```
(828, 3759)
```

```
y=df['STATUS'].values
y
```

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

```python
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(data,y,test_size=0.30,random_state=42)
x_train
```

```
<579x3759 sparse matrix of type '<class 'numpy.float64'>'
	with 7220 stored elements in Compressed Sparse Row format>
```

```python
x_test
```

```
<249x3759 sparse matrix of type '<class 'numpy.float64'>'
	with 3239 stored elements in Compressed Sparse Row format>
```

```python
y_train
```

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

```
y_test
```

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

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import BernoulliNB
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
knn=KNeighborsClassifier(n_neighbors=7)
nb=BernoulliNB()
sv=SVC()
dc=DecisionTreeClassifier(criterion='entropy')
rf=RandomForestClassifier()
lst=[knn,nb,sv,dc,rf]
for i in lst:
  print("Model started")
  print(i)
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  print("confusion matrix is....")
  print(confusion_matrix(y_test,y_pred))
  print("accuracy_score is......")
  print(accuracy_score(y_test,y_pred))
  print("CLASSIFICATION REPORT....")
  print(classification_report(y_test,y_pred))
  print("\n\n")
```

```
[[56 13 10]
 [34 30 15]
 [45 15 31]]
accuracy_score is......
0.46987951807228917
CLASSIFICATION REPORT....
              precision    recall  f1-score   support

          -1       0.41      0.71      0.52        79
           0       0.52      0.38      0.44        79
           1       0.55      0.34      0.42        91

    accuracy                           0.47       249
   macro avg       0.50      0.48      0.46       249
weighted avg       0.50      0.47      0.46       249




Model started
BernoulliNB()
confusion matrix is....
[[53  7 19]
```