NLP Twitter Sentiment Analysis

```
import numpy as np
import pandas as pd
import nltk
import seaborn as sns
import re
```

```
df=pd.read_csv('/content/twitter_validation.csv',header=None,encoding='ISO-8859-1')
df
```

	0	1	2	3
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

Next steps: View recommended plots

```
df.columns=['id','Social Media','target','text']
df
```

	id	Social Media	target	text
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

Next steps: View recommended plots

```
df.head()
```

	id	Social Media	target	text
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...

Next steps: View recommended plots

```
df.tail()
```

	id	Social Media	target	text
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.

```
df.dtypes
```

```
id              int64
Social Media    object
target          object
text            object
dtype: object
```

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

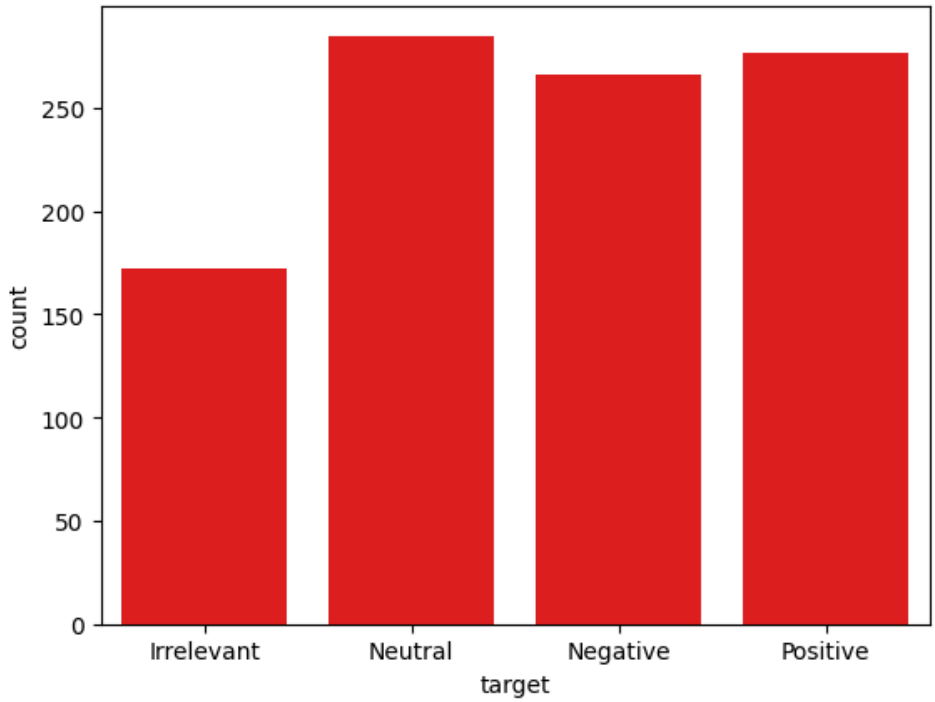
```
id              0
Social Media    0
target          0
text            0
dtype: int64
```

```
df['target'].value_counts()
```

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

```
sns.countplot(x='target',data=df,color='red')
```

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




```
df['Social Media'].value_counts
```

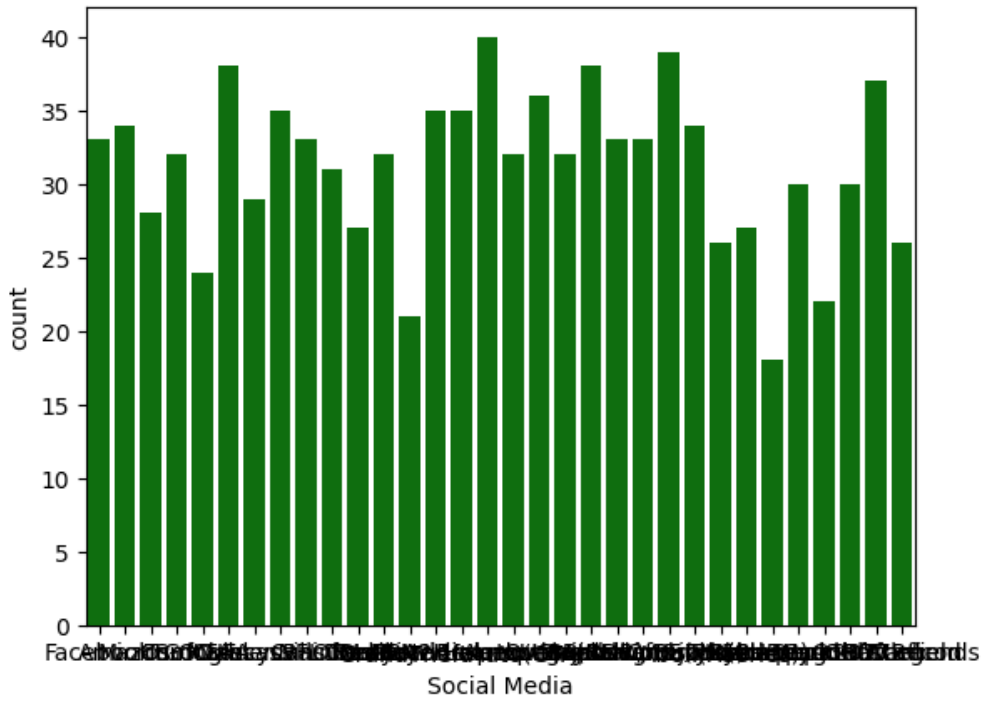
pandas.core.base.IndexOpsMixin.value_counts
```
def value_counts(normalize: bool=False, sort: bool=True, ascending: bool=False,
bins=None, dropna: bool=True) -> Series
```

/usr/local/lib/python3.10/dist-packages/pandas/core/base.py
Return a Series containing counts of unique values.

The resulting object will be in descending order so that the
first element is the most frequently-occurring element.
Excludes NA values by default.

```
sns.countplot(x='Social Media',data=df,color='green')
```

```
<Axes: xlabel='Social Media', ylabel='count'>
```



```
df['target'].unique()
```

```
array(['Irrelevant', 'Neutral', 'Negative', 'Positive'], dtype=object)
```

```
df.drop(df.index[(df['target']=='Irrelevant')],axis=0,inplace=True)
df
```

	id	Social Media	target	text
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...
5	6273	FIFA	Negative	Hi @EAHelp Iâ□□ve had Madeleine McCann in my c...
...	...	...	...	...
993	314	Amazon	Negative	Please explain how this is possible! How can t...
994	9701	PlayStation5(PS5)	Positive	Good on Sony. As much as I want to see the new...
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.

Next steps: View recommended plots

```
df.reset_index(drop=True,inplace=True)
df
```

	id	Social Media	target	text
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 Iâ□□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.

Next steps: View recommended plots

```
df.drop(['id','Social Media'],axis=1,inplace=True)
df
```

	target	text
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

Next steps: View recommended plots

```
#positive    :1  ,negative      :-1  ,  neutral   :0
df['target']=df['target'].map({'Positive':1,'Neutral':0,'Negative':-1})
```

```
df
```

	target	text
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

Next steps: View recommended plots

```
df.dtypes
```

```
target     int64
text      object
dtype: object
```

```
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('word-net')
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Error loading word-net: Package 'word-net' not found in
[nltk_data]     index
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
True
```

```
tweets=df.text
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: text, Length: 828, dtype: object
```

TOKENIZATION

```
from nltk import TweetTokenizer
tk=TweetTokenizer()
tweets=tweets.apply(lambda x:tk.tokenize(x)).apply(lambda x:" ".join(x))
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: text, Length: 828, dtype: object
```

```
tweets=tweets.str.replace('[^a-zA-Z0-9]+',' ')
tweets
```

```
<ipython-input-75-243a49c37bfd>:1: FutureWarning: The default value of regex will change from True to F
  tweets=tweets.str.replace('[^a-zA-Z0-9]+',' ')
0      BBC News Amazon boss Jeff Bezos rejects claims...
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 my cell...
                             ...
```

```
823    Please explain how this is possible How can th...
824    Good on Sony As much as I want to see the new ...
825    Today sucked so it s time to drink wine n play...
826     Bought a fraction of Microsoft today Small wins
827    Johnson Johnson to stop selling talc baby powd...
Name: text, Length: 828, dtype: object
```

```
#minimum 3 character
#remove characters lesss than three
from nltk.tokenize import word_tokenize
tweets=tweets.apply(lambda x:' '.join([w for w in word_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: text, Length: 828, dtype: object
```

STEMMING

```
from nltk.stem import SnowballStemmer
stemmer=SnowballStemmer('english')
tweets=tweets.apply(lambda x:[stemmer.stem(i.lower()) for i in tk.tokenize(x)]).apply(lambda x:' '.join(x))
```

STOP WORD REMOVING

```
from nltk.corpus import stopwords
sw=stopwords.words('english')
tweets=tweets.apply(lambda x:[i for i in tk.tokenize(x) if i not in sw]).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    pleas explain possibl let compani overcharg sc...
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: text, Length: 828, dtype: object
```

VECTORIZATION

```
#vectorization
#convert into numerical
from sklearn.feature_extraction.text import TfidfVectorizer
vec=TfidfVectorizer()
train_data=vec.fit_transform(tweets)
train_data
```

```
<828x3783 sparse matrix of type '<class 'numpy.float64'>'
        with 10505 stored elements in Compressed Sparse Row format>
```

```
print(train_data)
```

```
  (0, 691)      0.2608257828483461
  (0, 1004)     0.2608257828483461
  (0, 1130)     0.23509805002803952
  (0, 1996)     0.13277165480466424
  (0, 309)      0.22681557001542715
  (0, 860)      0.17354914655342313
  (0, 807)      0.21432663830218204
  (0, 2761)     0.2608257828483461
  (0, 568)      0.2608257828483461
  (0, 1833)     0.24577602391989378
  (0, 633)      0.22681557001542715
  (0, 376)      0.1515362387424402
  (0, 2287)     0.38864111655856126
  (0, 538)      0.49155204783978756
  (1, 797)      0.4055823664694651
  (1, 2891)     0.4055823664694651
  (1, 2558)     0.3821800909185634
  (1, 1405)     0.4055823664694651
  (1, 3679)     0.36557591217188057
  (1, 2462)     0.3126902562590763
  (1, 3639)     0.26216072802580975
  (1, 2155)     0.24555654927912696
  (2, 1427)     0.1689251539717079
  (2, 486)      0.36574263611909275
  (2, 3432)     0.36574263611909275
  :     :
  (825, 3393)   0.3395996844494919
  (825, 3383)   0.2560582225152134
  (825, 631)    0.22981061112100945
  (825, 997)    0.2315686698425631
  (825, 3373)   0.21750175079084832
  (825, 3226)   0.2904718522758868
  (825, 2527)   0.17148706662740873
  (826, 1381)   0.5079831062080814
  (826, 3070)   0.47867226429410115
  (826, 636)    0.4174215841659411
  (826, 3650)   0.353278941165688
  (826, 3383)   0.34523850330234374
  (826, 2155)   0.3075542453642147
  (827, 195)    0.3283693467320579
  (827, 1132)   0.3283693467320579
  (827, 2808)   0.3283693467320579
  (827, 143)    0.3283693467320579
  (827, 712)    0.2770320970909926
  (827, 2946)   0.2635889502019104
  (827, 3274)   0.2635889502019104
  (827, 3193)   0.2311987519368367
  (827, 1686)   0.15963411936668057
  (827, 2576)   0.24870786898500463
  (827, 506)    0.23743856420618148
  (827, 1854)   0.3947412386878786
```

```
y=df['target'].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])
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(train_data,y,test_size=.30,random_state=42)
```

```
from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=7)
knn.fit(x_train,y_train)
y_pred=knn.predict(x_test)
y_pred
```

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

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix,accuracy_score
from sklearn.metrics import classification_report
k_model=KNeighborsClassifier(n_neighbors=7)
n_model=MultinomialNB()
s_model=SVC()
lst_model=[k_model,n_model,s_model]



for i in lst_model:
  print('Model is',i)
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  print('*'*50)
  print("")
  print(confusion_matrix(y_test,y_pred))
  print('Accuracy score is',accuracy_score(y_test,y_pred))
  print("")
  print('-------Classification Report-----------')
  print(classification_report(y_test,y_pred))
```

```
Model is KNeighborsClassifier(n_neighbors=7)
**************************************************

[[57 13  9]
 [34 29 16]
 [46 14 31]]
Accuracy score is 0.46987951807228917

-------Classification Report-----------
              precision    recall  f1-score   support

          -1       0.42      0.72      0.53        79
           0       0.52      0.37      0.43        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 is MultinomialNB()
**************************************************

[[43 27  9]
 [14 43 22]
 [15 22 54]]
Accuracy score is 0.5622489959839357

-------Classification Report-----------
              precision    recall  f1-score   support

          -1       0.60      0.54      0.57        79
           0       0.47      0.54      0.50        79
           1       0.64      0.59      0.61        91

    accuracy                           0.56       249
   macro avg       0.57      0.56      0.56       249
weighted avg       0.57      0.56      0.56       249

Model is SVC()
**************************************************

[[41 34  4]
 [ 9 60 10]
 [11 36 44]]
Accuracy score is 0.5823293172690763
```

-------Classification Report-----------

	precision	recall	f1-score	support
-1	0.67	0.52	0.59	79
0	0.46	0.76	0.57	79
1	0.76	0.48	0.59	91
accuracy			0.58	249
macro avg	0.63	0.59	0.58	249
weighted avg	0.64	0.58	0.58	249