+ Code + Text

```
import re      #regular expression
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import nltk
from nltk.stem import WordNetLemmatizer


df=pd.read_csv('/content/twitter_validation.csv')
df.columns=['id','location','target','text']
df
```

	id	location	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 I've had Madeleine McCann in my cel...
...	...	...	...	...
994	4891	GrandTheftAuto(GTA)	Irrelevant	⭐ Toronto is the arts and culture capital of ...
995	4359	CS-GO	Irrelevant	tHIS IS ACTUALLY A GOOD MOVE TOT BRING MORE VI...
996	2652	Borderlands	Positive	Today sucked so it's time to drink wine n play...
997	8069	Microsoft	Positive	Bought a fraction of Microsoft today. Small wins.
998	6960	johnson&johnson	Neutral	Johnson & Johnson to stop selling talc baby po...

999 rows × 4 columns

```
df.head()
```

	id	location	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 I've had Madeleine McCann in my cel...

```
print('lenth of data is',len(df))
```

```
lenth of data is 999
```

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

```
id          0
location    0
target      0
text        0
dtype: int64
```

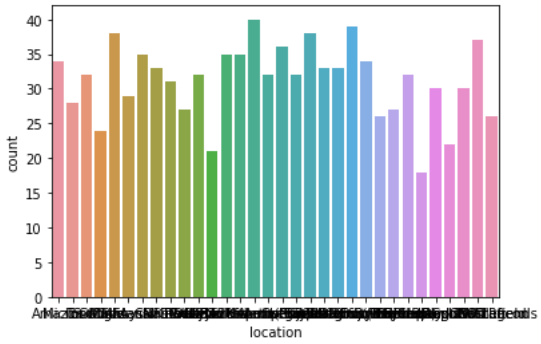
```
df['location'].value_counts()
```

```
RedDeadRedemption(RDR)               40
johnson&johnson                      39
FIFA                                 38
PlayerUnknownsBattlegrounds(PUBG)    38
LeagueOfLegends                      37
ApexLegends                          36
Nvidia                               35
TomClancysRainbowSix                 35
GrandTheftAuto(GTA)                  35
Fortnite                             34
Amazon                               34
AssassinsCreed                       33
Borderlands                          33
```

```
PlayStation5(PS5)                  33
Hearthstone                        32
Overwatch                          32
Verizon                            32
CS-GO                              32
Facebook                           32
CallOfDuty                         31
Cyberpunk2077                      30
WorldOfCraft                       30
MaddenNFL                          29
Microsoft                          28
Dota2                              27
CallOfDutyBlackopsColdWar          27
Battlefield                        26
Xbox(Xseries)                      26
Google                             24
TomClancysGhostRecon               22
NBA2K                              21
HomeDepot                          18
Name: location, dtype: int64
```

```
import seaborn as sns
sns.countplot(x='location',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7610fac9d0>
```

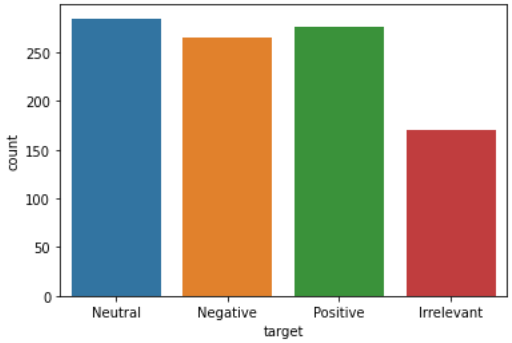



```
df['target'].value_counts()
```

```
Neutral       285
Positive      277
Negative      266
Irrelevant    171
Name: target, dtype: int64
```

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

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7610e82670>
```




```
#remove irrelevent from targrt
df.drop(df.index[(df['target']=='Irrelevant')],axis=0,inplace=True)
df
```

	id	location	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 I've had Madeleine McCann in my cel...
...	...	...	...	...
992	314	Amazon	Negative	Please explain how this is possible! How can t...
993	9701	PlayStation5(PS5)	Positive	Good on Sony. As much as I want to see the new...

```
#reset indexvalue
df.reset_index(drop=True,inplace=True)
df
```

	id	location	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 I've had Madeleine McCann in my cel...
...	...	...	...	...
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 it's time to drink wine n play...
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','location'],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 I've had Madeleine McCann in my cel...
...	...	...
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 it's time to drink wine n play...
826	Positive	Bought a fraction of Microsoft today. Small wins.
827	Neutral	Johnson & Johnson to stop selling talc baby po...

828 rows × 2 columns

```
#changing values of positive negative and neutral
df['target']=df['target'].map({'Positive':1,'Negative':-1,'Neutral':0})
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 I've had Madeleine McCann in my cel...
...	...	...
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 it's time to drink wine n play...

```
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
```

```
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 I've had Madeleine McCann in my cel...
                             ...
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 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 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 I ' ve had Madeleine McCann in my c...
                             ...
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 pl...
826    Bought a fraction of Microsoft today . Small w...
827    Johnson & Johnson to stop selling talc baby po...
Name: text, Length: 828, dtype: object
```

```
#removing special charecters example
str1='Worldof anime/?[123>:'
str2=re.sub('[^a-zA-Z0-9]'," ",str1)
str2
```

```
'Worldof anime   123  '
```

```
#removing special charecters
tweets.str.replace('[^a-zA-Z0-9]+','')
```

```
<ipython-input-20-4cef94aace1b>:2: FutureWarning: The default value of regex will change from True to False in a future version.
  tweets.str.replace('[^a-zA-Z0-9]+','')
0      BBCNewsAmazonbossJeffBezosrejectsclaimscompany...
```

```
1      MicrosoftWhydoIpayforWORDwhenitfunctionssopoor...
2      CSGOmatchmakingissofullofclosethackingitsatrul...
3      NowthePresidentisslappingAmericansinthefacetha...
4      HiEAHelpIvehadMadeleineMcCanninmycellarforthep...
                           ...
823    PleaseexplainhowthisispossibleHowcantheyletcom...
824    GoodonSonyAsmuchasIwanttoseethenewPS5whatsgoin...
825    Todaysuckedsoitstimetodrinkwinenplayborderland...
826              BoughtafractionofMicrosofttodaySmallwins
827    JohnsonJohnsontostopsellingtalcbabypowderinUSa...
Name: text, Length: 828, dtype: object
```

```
#lenth avobe 3 tokens only(for a meaning full word)
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
```

```
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))
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    plea explain how this possibl how can they let...
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: text, Length: 828, dtype: object
```

```
from nltk.corpus import stopwords
stop=stopwords.words('english')
tweets=tweets.apply(lambda x:[i for i in tk.tokenize(x) if i not in stop]).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: text, Length: 828, dtype: object
```

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

```
print(train_data)
```

```
  (0, 629)      0.24330548895023374
  (0, 416)      0.24330548895023374
  (0, 3496)     0.21930594969814493
  (0, 774)      0.1169050386344535
```

```
  (0, 951)      0.24330548895023374
  (0, 1080)     0.22926665845317631
  (0, 1973)     0.123853064057564
  (0, 232)      0.21157982374854553
  (0, 803)      0.16189143380671017
  (0, 748)      0.19992980355585638
  (0, 2767)     0.24330548895023374
  (0, 507)      0.24330548895023374
  (0, 1809)     0.22926665845317631
  (0, 571)      0.21157982374854553
  (0, 300)      0.14135718585131646
  (0, 2271)     0.3685605265618393
  (0, 476)      0.45853331690635263
  (1, 738)      0.4055823664694651
  (1, 2898)     0.4055823664694651
  (1, 2556)     0.3821800909185634
  (1, 1364)     0.4055823664694651
  (1, 3715)     0.36557591217188057
  (1, 2457)     0.3126902562590763
  (1, 3673)     0.26216072802580975
  (1, 2134)     0.24555654927912696
  :     :
  (825, 3410)   0.2560582225152134
  (825, 569)    0.22981061112100945
  (825, 944)    0.2315686698425631
  (825, 3399)   0.21750175079084832
  (825, 3244)   0.2904718522758868
  (825, 2523)   0.17148706662740873
  (826, 1338)   0.5079831062080814
  (826, 3083)   0.47867226429410115
  (826, 574)    0.4174215841659411
  (826, 3684)   0.353278941165688
  (826, 3410)   0.34523850330234374
  (826, 2134)   0.3075542453642147
  (827, 1083)   0.3243569563278974
  (827, 2815)   0.3243569563278974
  (827, 130)    0.3243569563278974
  (827, 2209)   0.3243569563278974
  (827, 653)    0.27364700363122035
  (827, 2956)   0.260368120410828
  (827, 3293)   0.260368120410828
  (827, 3209)   0.22837370245229333
  (827, 1658)   0.15768352801246954
  (827, 2575)   0.2456688731807802
  (827, 442)    0.23453726959363683
  (827, 1831)   0.38991784096820636
  (827, 774)    0.15584918644651963
```

```
train_data.shape
```

```
(828, 3824)
```

```
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,  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=0.30,random_state=42)
x_train
```

```
<579x3824 sparse matrix of type '<class 'numpy.float64'>'
        with 7378 stored elements in Compressed Sparse Row format>
```

```
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,  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,  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])
```

```
from sklearn.svm import SVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
svm_model=SVC()
nb_model=MultinomialNB()
rf_model=RandomForestClassifier()
de_model=DecisionTreeClassifier()
lstmodel=[svm_model,nb_model,rf_model,de_model]
```

```
from sklearn.metrics import confusion_matrix,classification_report
for i in lstmodel:
  print(i)
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  print("***********************************************************")
  print(classification_report(y_test,y_pred))
  print('***********************************************************')
```

```
SVC()
***********************************************************
              precision    recall  f1-score   support

          -1       0.68      0.52      0.59        79
           0       0.48      0.76      0.59        79
           1       0.72      0.51      0.59        91

    accuracy                           0.59       249
   macro avg       0.63      0.59      0.59       249
weighted avg       0.63      0.59      0.59       249

***********************************************************
MultinomialNB()
***********************************************************
              precision    recall  f1-score   support

          -1       0.62      0.56      0.59        79
           0       0.48      0.57      0.52        79
           1       0.64      0.59      0.62        91

    accuracy                           0.57       249
   macro avg       0.58      0.57      0.57       249
weighted avg       0.58      0.57      0.58       249

***********************************************************
RandomForestClassifier()
***********************************************************
              precision    recall  f1-score   support

          -1       0.47      0.72      0.57        79
           0       0.54      0.52      0.53        79
           1       0.69      0.38      0.49        91

    accuracy                           0.53       249
   macro avg       0.56      0.54      0.53       249
weighted avg       0.57      0.53      0.53       249

***********************************************************
DecisionTreeClassifier()
***********************************************************
              precision    recall  f1-score   support

          -1       0.47      0.67      0.55        79
           0       0.47      0.44      0.46        79
           1       0.57      0.40      0.47        91

    accuracy                           0.50       249
   macro avg       0.51      0.50      0.49       249
weighted avg       0.51      0.50      0.49       249

***********************************************************
```