```
import numpy as np
import pandas as pd            # 0 rap  1 pop
import seaborn as sns
import nltk
import matplotlib.pyplot as plt
import re
df =pd.read_csv('/content/train.csv',encoding ='ISO-8859-1')     #nlp
df
```

	lyric	class
0	Can't drink without thinkin' about you	1
1	Now Lil Pump flyin' private jet (Yuh)	0
2	No, matter fact, you ain't help me when I had ...	0
3	And you could find me, I ain't hidin'	0
4	From the way you talk to the way you move	1
...	...	...
51049	I told her pour me some more, then she went ri...	0
51050	Hit the ground and crawl to the dresser	0
51051	Just keep breathin' and breathin' and breathin...	1
51052	Down go the system, long live the king (King)	0
51053	If your mother knew all the things we do (From...	1

51054 rows × 2 columns

```
df.head()
```

	lyric	class
0	Can't drink without thinkin' about you	1
1	Now Lil Pump flyin' private jet (Yuh)	0
2	No, matter fact, you ain't help me when I had ...	0
3	And you could find me, I ain't hidin'	0
4	From the way you talk to the way you move	1

```
df.tail()
```

	lyric	class
51049	I told her pour me some more, then she went ri...	0
51050	Hit the ground and crawl to the dresser	0
51051	Just keep breathin' and breathin' and breathin...	1
51052	Down go the system, long live the king (King)	0
51053	If your mother knew all the things we do (From...	1

```
df.columns
```

```
Index(['lyric', 'class'], dtype='object')
```

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

```
lyric    0
class    0
dtype: int64
```

```
df.dtypes
```

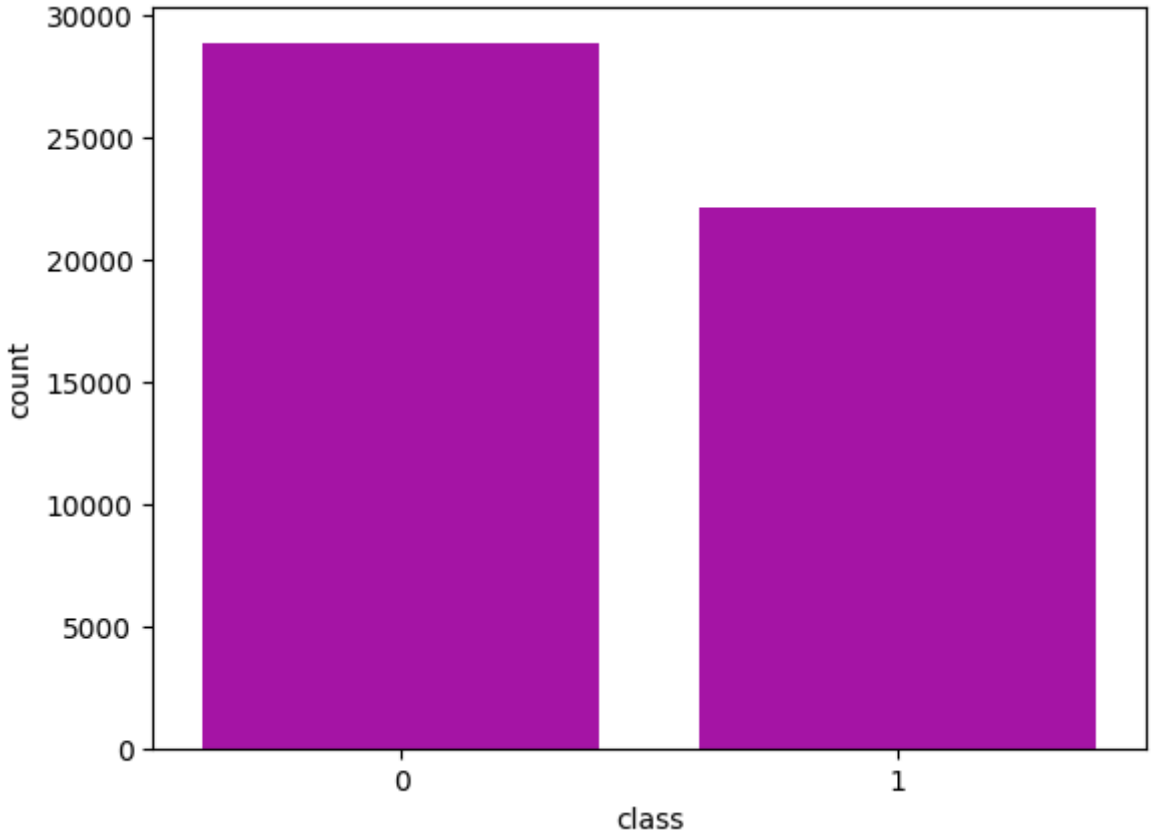
```
lyric    object
class     int64
dtype: object
```

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

```
class
0    28885
1    22169
Name: count, dtype: int64
```

```
sns.countplot(x='class',data=df,color='m')
```

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




```
df.dtypes
```

```
lyric    object
class     int64
dtype: object
```

```
df.shape
```

```
(51054, 2)
```

```
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
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] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
True
```

```
df['lyric']
```

```
0                   Can't drink without thinkin' about you
1                    Now Lil Pump flyin' private jet (Yuh)
2        No, matter fact, you ain't help me when I had ...
3                    And you could find me, I ain't hidin'
4                From the way you talk to the way you move
                                  ...
51049    I told her pour me some more, then she went ri...
51050              Hit the ground and crawl to the dresser
51051    Just keep breathin' and breathin' and breathin...
51052        Down go the system, long live the king (King)
51053    If your mother knew all the things we do (From...
Name: lyric, Length: 51054, dtype: object
```

```
#tokenization
```

```
from nltk.tokenize import TweetTokenizer
tk = TweetTokenizer()
df['lyric'] =df['lyric'].apply(lambda x:tk.tokenize(x)).apply(lambda x:" ".join(x))   #va
df['lyric']
```

```
0                   Can't drink without thinkin ' about you
1                 Now Lil Pump flyin ' private jet ( Yuh )
2        No , matter fact , you ain't help me when I ha...
3                   And you could find me , I ain't hidin '
4                From the way you talk to the way you move
                                  ...
51049    I told her pour me some more , then she went r...
51050              Hit the ground and crawl to the dresser
51051    Just keep breathin ' and breathin ' and breath...
51052     Down go the system , long live the king ( King )
51053    If your mother knew all the things we do ( Fro...
Name: lyric, Length: 51054, dtype: object
```

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

```
0                   Can't drink without thinkin ' about you
1                 Now Lil Pump flyin ' private jet ( Yuh )
2        No , matter fact , you ain't help me when I ha...
3                   And you could find me , I ain't hidin '
4                From the way you talk to the way you move
                                  ...
51049    I told her pour me some more , then she went r...
51050              Hit the ground and crawl to the dresser
51051    Just keep breathin ' and breathin ' and breath...
51052     Down go the system , long live the king ( King )
51053    If your mother knew all the things we do ( Fro...
Name: lyric, Length: 51054, dtype: object
```

```
#minimum 3 letters meaningful word

#character which are less than 2 will remove


from nltk.tokenize import word_tokenize
df['lyric'] = df['lyric'].apply(lambda x:' '.join([w for w in word_tokenize(x) if len(w)>
df['lyric']
```

```
0                          drink without thinkin about
1                                   Pump flyin private
2                          matter fact help when money
3                                     could find hidin
4                                       From talk move
                               ...
51049    told pour some more then went right blow blow
51050                               ground crawl dresser
51051    Just keep breathin breathin breathin breathin
51052                  Down system long live king King
51053                     your mother knew things From
Name: lyric, Length: 51054, dtype: object
```

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

```
0                          drink without thinkin about
1                                    pump flyin privat
2                          matter fact help when money
3                                     could find hidin
4                                       from talk move
                               ...
51049    told pour some more then went right blow blow
51050                               ground crawl dresser
51051    just keep breathin breathin breathin breathin
51052                  down system long live king king
51053                     your mother knew thing from
Name: lyric, Length: 51054, dtype: object
```

```
#stop words remove

from nltk.corpus import stopwords
sw = stopwords.words('english')
df['lyric'] = df['lyric'].apply(lambda x:[i for i in tk.tokenize(x) if i not in sw]).appl
df['lyric']
```

```
0                           drink without thinkin
1                               pump flyin privat
2                          matter fact help money
3                                could find hidin
4                                      talk move
                               ...
51049             told pour went right blow blow
51050                        ground crawl dresser
```

```
51051    keep breathin breathin breathin breathin
51052                  system long live king king
51053                           mother knew thing
Name: lyric, Length: 51054, dtype: object
```

```
x=np.array(df['lyric'])
y=np.array(df['class'])
```

```
#veectoraization TF-IDF

from sklearn.feature_extraction.text import TfidfVectorizer
vec =TfidfVectorizer()
x = vec.fit_transform(x)
print(x)
```

```
  (0, 9189)     0.5913773036008608
  (0, 10128)    0.5600421173466104
  (0, 2763)     0.5801945463236456
  (1, 7048)     0.5513421467505191
  (1, 3467)     0.5709184753812636
  (1, 7139)     0.6083370214649627
  (2, 5924)     0.40443323669464504
  (2, 4216)     0.5067740224693644
  (2, 3189)     0.5493407888304445
  (2, 5638)     0.5271039223284705
  (3, 4247)     0.7622091066747508
  (3, 3355)     0.49946420035078565
  (3, 2066)     0.41179216878181224
  (4, 5998)     0.7272207241739417
  (4, 9025)     0.6864036846724585
  (5, 4288)     0.5259522053218819
  (5, 4102)     0.6376666164613831
  (5, 8445)     0.5628104156532112
  (6, 7590)     0.7212704144588531
  (6, 5950)     0.5076857021408858
  (6, 5027)     0.4711944578070398
  (7, 5028)     1.0
  (8, 9520)     0.5562106525014121
  (8, 7134)     0.4260353276621745
  (8, 3148)     0.7135289830327406
  :     :
  (51045, 9944) 0.30923431664385453
  (51045, 5273) 0.24063326695256906
  (51046, 5771) 0.6951880029912184
  (51046, 8162) 0.7188279630739761
  (51047, 5265) 0.6324487217797286
  (51047, 2866) 0.7746022297406505
  (51048, 5373) 0.6126887355524886
  (51048, 1878) 0.7903243089561985
  (51049, 10018)        0.349961314475995
  (51049, 7540) 0.2816518909759345
  (51049, 6961) 0.39650127892138387
  (51049, 987)  0.7373707597718934
  (51049, 9327) 0.3118820115477033
  (51050, 2755) 0.6347035520700485
  (51050, 2115) 0.5754274874517341
```

```
(51050, 3969) 0.5157854279394172
(51051, 1151) 0.9896790129494607
(51051, 4914) 0.14330195856087008
(51052, 8996) 0.46742745598674895
(51052, 4994) 0.7666990158587504
(51052, 5312) 0.3043971346491529
(51052, 5361) 0.3178467820953434
(51053, 5973) 0.6566727408062916
(51053, 9187) 0.4707879487155193
(51053, 5019) 0.5891855555138692
```

```
from sklearn.cluster import KMeans
```

```
wcss = []
for i in range(1,11):
  km = KMeans(n_clusters=i)
  km.fit_predict(x)
  wcss.append(km.inertia_)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
```

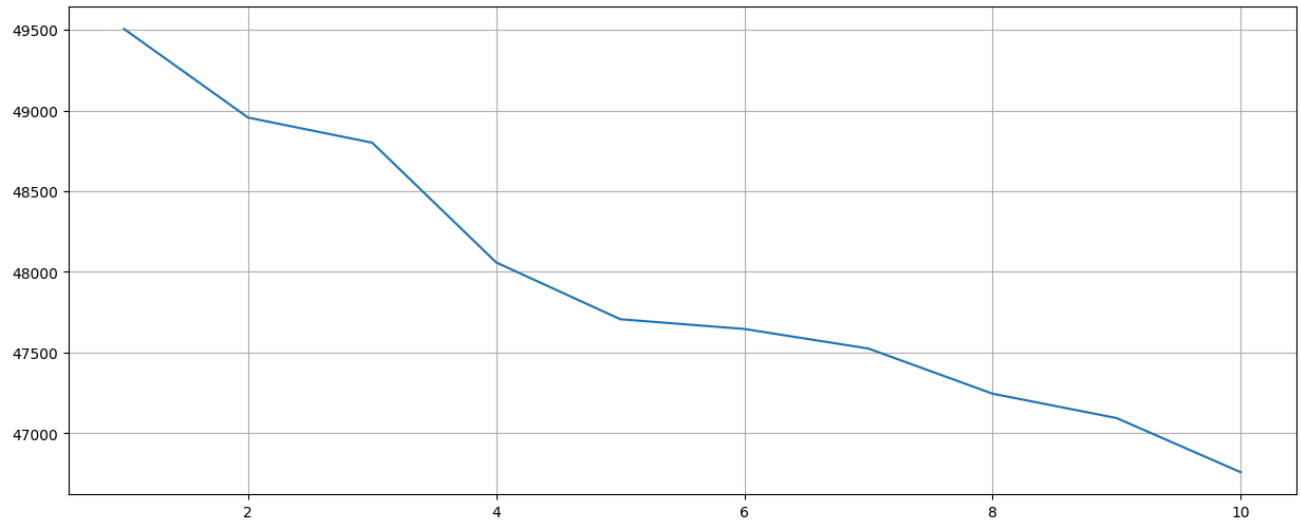
```
print(km.cluster_centers_)
```

```
[[3.18542250e-04 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 ...
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [1.02629821e-04 2.55725326e-05 1.10473246e-05 ... 1.16497784e-05
  1.28178835e-05 1.51774645e-05]]
```

```
wcss
```

```
[49506.00181138105,
 48956.529048506185,
 48801.044496685936,
 48058.35253245229,
 47706.595592824924,
 47646.17335414111,
 47524.778588488756,
 47245.00471219566,
 47093.86825448595,
 46757.98770357904]
```

```
plt.figure(figsize=(15,6))
plt.plot(range(1,11),wcss)
plt.grid()
plt.xticks(rotation = 0)
plt.show()
```




```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size =0.33,random_state=42)
xtrain
```

```
<34206x10319 sparse matrix of type '<class 'numpy.float64'>'
        with 99975 stored elements in Compressed Sparse Row format>
```

```
xtest
```

```
<16848x10319 sparse matrix of type '<class 'numpy.float64'>'
        with 49112 stored elements in Compressed Sparse Row format>
```

```
ytrain
```

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

```
ytest
```

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

```
#model create

from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import BernoulliNB
```

```
for i in lst:
  print("model name is",i)
  i.fit(xtrain,ytrain)
  ypred=i.predict(xtest)
  print("predicted value is",ypred)
  print('accuracy_score is',accuracy_score(ytest,ypred))
  print('classification report is',classification_report(ytest,ypred))
```

```
    model name is KNeighborsClassifier(n_neighbors=7)
    predicted value is [0 1 1 ... 1 1 0]
    accuracy_score is 0.6254154795821463
    classification report is               precision    recall  f1-score   support

               0       0.80      0.45      0.58      9533
               1       0.54      0.85      0.66      7315

        accuracy                           0.63     16848
       macro avg       0.67      0.65      0.62     16848
    weighted avg       0.69      0.63      0.61     16848

    model name is BernoulliNB()
    predicted value is [0 1 0 ... 1 1 0]
    accuracy_score is 0.7399097815764483
    classification report is               precision    recall  f1-score   support

               0       0.81      0.71      0.75      9533
               1       0.67      0.78      0.72      7315

        accuracy                           0.74     16848
       macro avg       0.74      0.75      0.74     16848
    weighted avg       0.75      0.74      0.74     16848

    model name is SVC()
    predicted value is [0 1 0 ... 1 1 0]
    accuracy_score is 0.8087013295346629
    classification report is               precision    recall  f1-score   support

               0       0.82      0.85      0.83      9533
               1       0.79      0.76      0.77      7315

        accuracy                           0.81     16848
       macro avg       0.81      0.80      0.80     16848
    weighted avg       0.81      0.81      0.81     16848
```