In [274]:
```
import pandas as pd
import numpy as np
import warnings
import re
warnings.filterwarnings('ignore')
trainset=pd.read_csv('D:\PyCharm Community Edition 2020.1.3\datasets\\train.csv')
testset=pd.read_csv('D:\PyCharm Community Edition 2020.1.3\datasets\\test.csv')
dataset=pd.concat([trainset,testset],ignore_index=True)
dataset
```

Out[274]:

	text	sentiment
0	Now, I won't deny that when I purchased this o...	neg
1	The saddest thing about this "tribute" is that...	neg
2	Last night I decided to watch the prequel or s...	neg
3	I have to admit that i liked the first half of...	neg
4	I was not impressed about this film especially...	neg
...	...	...
49995	For one thing, he produced this movie. It has ...	neg
49996	The title comes from an alteration an adolesce...	pos
49997	Christopher Nolan's first film is a 'no budget...	pos
49998	The story is shortly about the faith-lacking b...	neg
49999	I found parts of this movie rather slow, espec...	pos

50000 rows × 2 columns

In [275]:
```
#split and extract words
data_senti=dataset.copy(deep=True)
for i,comments in enumerate(dataset['text']):
    data_senti['text'].iloc[i]=re.sub("[^a-zA-Z]"," ",comments)
    data_senti['sentiment'].iloc[i]=dataset['sentiment'].iloc[i]
data_senti.head()
```

Out[275]:

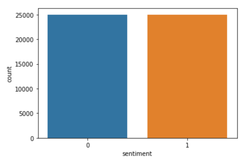
	text	sentiment
0	Now I won t deny that when I purchased this o...	neg
1	The saddest thing about this tribute is that...	neg
2	Last night I decided to watch the prequel or s...	neg
3	I have to admit that i liked the first half of...	neg
4	I was not impressed about this film especially...	neg

In [276]:
```
data_senti=data_senti.replace("neg",0)
data_senti=data_senti.replace("pos",1)
```

In [277]:
```
##check if data is balanced

import seaborn as sns
sns.countplot(x='sentiment', data=data_senti)
```

Out[277]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x27dbd7d6c88>
```



In [278]:
```
data=data_senti.copy(deep=True)
```

In [300]:
```
X_train=data.loc[:35000,'text']
y_train=data.loc[:35000,'sentiment']
X_test=data.loc[35000:,'text']
y_test=data.loc[35000:,'sentiment']
```

In [301]:
```
x_word2_train=X_train.copy()
x_word2_test=X_test.copy()
```

In [311]:
```
x_word2_test
```

Out[311]:
```
35000    Dennis Patrick plays a man who accidentally ki...
35001    This   hour    minute inside joke is best unde...
35002    The novel WEAPON which serves as the basis for...
35003    Is it possible for a movie to get any worse th...
35004    For the very reason that I love movies such as...
                               ...
49995    For one thing  he produced this movie  It has ...
49996    The title comes from an alteration an adolesce...
49997    Christopher Nolan s first film is a  no budget...
49998    The story is shortly about the faith lacking b...
49999    I found parts of this movie rather slow  espec...
Name: text, Length: 15000, dtype: object
```

In [5]:
```
import nltk
#nltk.download('stopwords')
```

In [6]:
```
from nltk.corpus import stopwords
en_stops = set(stopwords.words('english'))
```

def delet_stw(sets,n_ds): for i,strings in enumerate(sets['text']): n_string=[] for words in strings: if words not in en_stops: n_string.append(words) n_ds[i]=n_string return n_ds x_word2_train=delet_stw(x_word2_train,x_word2_train) x_word2_test=delet_stw(x_word2_test,x_word2_test)

In [255]:
```
#import nltk
#nltk.download('punkt')
from nltk.tokenize import sent_tokenize, word_tokenize
import warnings

warnings.filterwarnings(action = 'ignore')

import gensim
from gensim.models import Word2Vec
sets=pd.concat([trainset,testset],axis=0,ignore_index=True)
# Replaces escape character with space
f0 = data['text'].replace(","," ")
#f1 = f.replace("<br/>", " ")
data = []

# iterate through each sentence in the file
for m in range(len(f0)-1):
    f2=str(f0[m+1])
    for i in sent_tokenize(f2):
        temp = []
        # tokenize the sentence into words
        for j in word_tokenize(i):
            temp.append(j.lower())
        data.append(temp)
```

In [303]:
```
model2 = gensim.models.Word2Vec(data, min_count = 2, size = 20,
                                window = 5, sg = 1)
```

In [13]:
```
def get_set(dataset,targetset):
    f1 = dataset['text'].replace(","," ")
#f1 = f.replace("<br/>", " ")
    for m in range(len(f1)-1):
        f2=str(f1[m+1])
        vector = []
        for i in sent_tokenize(f2):
            for j in word_tokenize(i):
                try:
                    vector_i=model2.wv[j.lower()]
                    vector.append(vector_i)
                except:
                    continue
        vec=np.mean([vector[token] for token in range(len(vector))], axis=0).tolist()
        targetset.iloc[m,0]=vec
        targetset.iloc[m,1]=dataset.iloc[m,1]
        targetset=targetset.iloc[0:-1,]
    return targetset
```

In [304]:
```
f1=[]
transed_test=[]
transed_train=[]
sel_y_train=[]
sel_y_test=[]
f1 = x_word2_train.replace(","," ")
for m in range(len(f1)-1):
    try:
        f2=str(f1[m+1])
    except:
        continue
    vector = []
    for j in word_tokenize(f2):
        try:
            vector_i=model2.wv[j.lower()]
            vector.append(vector_i)
        except:
            continue
    vec=np.mean([vector[token] for token in range(len(vector))], axis=0).tolist()
    transed_train.append(vec)
    sel_y_train.append(y_train[m])
```

In [312]:
```
f1 = x_word2_test.replace(","," ")
for m in range(35000,len(f1)-1+35000):
    try:
        f2=str(f1[m+1])
        vector = []
        for j in word_tokenize(f2):
            try:
                vector_i=model2.wv[j.lower()]
                vector.append(vector_i)
            except:
                continue
    except:
        continue
    vec=np.mean([vector[token] for token in range(len(vector))], axis=0).tolist()
    transed_test.append(vec)
    sel_y_test.append(y_test[m])
#train=get_set(trainset,train)
#test=get_set(testset,test)
len(transed_test)
#len(sel_y_test)
```

Out[312]:
```
14999
```

In [353]:
```
x_train = pd.Series(transed_train)
x_test = pd.Series(transed_test)
```

In [357]:
```
x_train=x_train.values.reshape(1,-1)
x_test=x_test.values.reshape(1,-1)
```

In [216]:
```
from sklearn.naive_bayes import GaussianNB
model=GaussianNB(var_smoothing=0.01)
model.fit(x_train,y_train)
predictions = model.predict(x_test)
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

          -1       0.74      0.98      0.85      1870
           0       0.76      0.31      0.44       614
           1       0.87      0.38      0.53       444

    accuracy                           0.75      2928
   macro avg       0.79      0.56      0.61      2928
weighted avg       0.76      0.75      0.71      2928
```

In [ ]:
```

```