In [1]:
```
import pandas as pd
import numpy as np
import warnings
import re
warnings.filterwarnings('ignore')
dataset=pd.read_csv("https://github.com/XiaoyuLiu198/IMDB-Classify/blob/master/Tweets.csv")
dataset.head()
```

Out[1]:

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline	ai
0	570306133677760513	neutral	1.0000	NaN	NaN	Virgin America	
1	570301130888122368	positive	0.3486	NaN	0.0000	Virgin America	
2	570301083672813571	neutral	0.6837	NaN	NaN	Virgin America	
3	570301031407624196	negative	1.0000	Bad Flight	0.7033	Virgin America	
4	570300817074462722	negative	1.0000	Can't Tell	1.0000	Virgin America	

In [2]:
```
#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['airline_sentiment'].iloc[i]=dataset['airline_sentiment'].iloc[i]
data_senti.head()
```

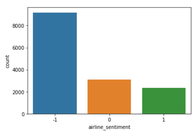
Out[2]:

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline	ai
0	570306133677760513	neutral	1.0000	NaN	NaN	Virgin America	
1	570301130888122368	positive	0.3486	NaN	0.0000	Virgin America	
2	570301083672813571	neutral	0.6837	NaN	NaN	Virgin America	
3	570301031407624196	negative	1.0000	Bad Flight	0.7033	Virgin America	
4	570300817074462722	negative	1.0000	Can't Tell	1.0000	Virgin America	

In [3]:
```
data_senti=data_senti.replace("negative",-1)
data_senti=data_senti.replace("positive",1)
data_senti=data_senti.replace("neutral",0)
```

In [4]:
```
#cheeck if data is balanced

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

Out[4]: <matplotlib.axes._subplots.AxesSubplot at 0x1c69242cfc8>



In [5]:
```
data=data_senti.copy(deep=True)
```

In [6]:
```
data=data[['text','airline_sentiment']]
data.head()
```

Out[6]:

	text	airline_sentiment
0	VirginAmerica What dhepburn said	0
1	VirginAmerica plus you've added commercials t...	1
2	VirginAmerica I didn t today Must mean I n...	0
3	VirginAmerica it s really aggressive to blast...	-1
4	VirginAmerica and it s a really big bad thing...	-1

In [7]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data['text'], data['airline_sentiment'],
test_size=0.2, random_state=0)
```

test_word_sent=teststet.copy(deep=True) for i,comments in enumerate(testset['text']): test_word_sent['text'].iloc[i]=re.sub("[^a-zA-Z]"," ",comments).split() test_word_sent['sentiment'].iloc[i]=teststet['sentiment'].iloc[i] test_word_senti.head()

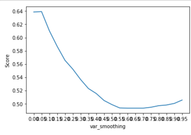
In [8]:
```
import nltk
#nltk.download('stopwords')
```

In [9]:
```
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_strings=[] 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 [10]:
```
##etd-idf transformation
tf_x_train=X_train.copy()
tf_x_test=X_test.copy()
from sklearn.feature_extraction.text import TfidfVectorizer
tfidfconverter = TfidfVectorizer(sublinear_tf=True,max_features=2000, min_df=5, max_df=0.7)
tf_x_train =tfidfconverter.fit_transform(tf_x_train)#train the vectorizer, build the vocabul
ary
tf_x_test=tfidfconverter.transform(tf_x_test)
```

In [11]:
```
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import confusion_matrix
from sklearn import preprocessing
from sklearn.metrics import precision_recall_curve
from sklearn.naive_bayes import GaussianNB
lists=np.arange(0,1,0.05)
scoreList = []
accuracies = {}
for i in lists:
    clf = GaussianNB(var_smoothing=i)
    clf.fit(tf_x_train.toarray(),y_train)
    scoreList.append(clf.score(tf_x_test.toarray(),y_test))

plt.plot(np.arange(0,1,0.05), scoreList)
plt.xticks(np.arange(0,1,0.05))
plt.xlabel("var_smoothing")
plt.ylabel("Score")
plt.show()
```



In [12]:
```
from sklearn.metrics import classification_report
clf = GaussianNB(var_smoothing=0.01)
clf.fit(tf_x_train.toarray(),y_train)
predictions = clf.predict(tf_x_test.toarray())
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

          -1       0.88      0.64      0.74      1870
           0       0.44      0.54      0.49       614
           1       0.40      0.76      0.53       444

    accuracy                           0.63      2928
   macro avg       0.58      0.64      0.58      2928
weighted avg       0.72      0.63      0.65      2928
```

In [13]:
```
tf_x_train[1].toarray()
```

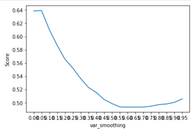
Out[13]:
```
array([[0., 0., 0., ..., 0., 0., 0.]])
```

In [15]:
```
from sklearn.naive_bayes import MultinomialNB
model=MultinomialNB(alpha=1.0, fit_prior=True, class_prior=None)
model.fit(tf_x_train,y_train)
predictions = model.predict(tf_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 [16]:
```
#conda install -c glemaitre imbalanced-learn
from imblearn.over_sampling import RandomOverSampler
from imblearn.under_sampling import RandomUnderSampler
#import RandomUnderSampler from imblearn
undersample = RandomUnderSampler(sampling_strategy='majority')
X_under, y_under = undersample.fit_resample(data, data['airline_sentiment'])
```

In [17]:
```
X_train, X_test, y_train, y_test = train_test_split(data['text'], data['airline_sentiment'],
test_size=0.2, random_state=0)
##etd-idf transformation
tf_x_train=X_train.copy()
tf_x_test=X_test.copy()
tfidfconverter = TfidfVectorizer(sublinear_tf=True,max_features=2000, min_df=5, max_df=0.7)
tf_x_train =tfidfconverter.fit_transform(tf_x_train)#train the vectorizer, build the vocabul
ary
tf_x_test=tfidfconverter.transform(tf_x_test)
lists=np.arange(0,1,0.05)
scoreList = []
accuracies = {}
for i in lists:
    clf = GaussianNB(var_smoothing=i)
    clf.fit(tf_x_train.toarray(),y_train)
    scoreList.append(clf.score(tf_x_test.toarray(),y_test))

plt.plot(np.arange(0,1,0.05), scoreList)
plt.xticks(np.arange(0,1,0.05))
plt.xlabel("var_smoothing")
plt.ylabel("Score")
plt.show()
clf = GaussianNB(var_smoothing=0.01)
clf.fit(tf_x_train.toarray(),y_train)
predictions = clf.predict(tf_x_test.toarray())
print(classification_report(y_test,predictions))
model=MultinomialNB(alpha=1.0, fit_prior=True, class_prior=None)
model.fit(tf_x_train,y_train)
predictions = model.predict(tf_x_test)
print(classification_report(y_test,predictions))
```




```
              precision    recall  f1-score   support

          -1       0.88      0.64      0.74      1870
           0       0.44      0.54      0.49       614
           1       0.40      0.76      0.53       444

    accuracy                           0.63      2928
   macro avg       0.58      0.64      0.58      2928
weighted avg       0.72      0.63      0.65      2928

              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 [ ]:
```
```