In [8]:
```
#import packages

import pandas as pd
import numpy as np
import model_evaluation_utils as meu
import matplotlib.pyplot as plt

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split

import xgboost
from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_
score, classification_report, confusion_matrix

import re
import nltk

%matplotlib inline
```

In [2]:
```
# normalize function

wpt = nltk.WordPunctTokenizer()
stop_words = nltk.corpus.stopwords.words('english')

def normalize_document(doc):
    # lower case and remove special characters\whitespaces
    #doc = re.sub(r'[^a-zA-Z\s]', '', doc, re.I)
    doc = re.sub(r'[^a-zA-Z0-9\s]', '', doc, re.I)
    doc = doc.lower()
    doc = doc.strip()
    # tokenize document
    tokens = wpt.tokenize(doc)
    # filter stopwords out of document
    filtered_tokens = [token for token in tokens if token not in stop_words]
    # re-create document from filtered tokens
    doc = ' '.join(filtered_tokens)
    doc = ''.join(i for i in doc if not i.isdigit())
    return doc

normalize_corpus = np.vectorize(normalize_document)

#Load in corpus
df = pd.read_csv('data/subset.csv')

col = ['stars_y', 'text']
df = df[col]
df = df[pd.notnull(df['text'])]

df.columns = ['stars_y', 'text']

norm_df = normalize_corpus(df['text'])
```

In [3]:
```
cv = CountVectorizer(binary=False, min_df=0.0, max_df=1.0, ngram_range=(1,2))
features = cv.fit_transform(norm_df)
labels = df.stars_y
features.shape
```

Out[3]: (706731, 9682018)

In [4]:
```
# build train and test datasets

X_train, X_test, y_train, y_test = train_test_split(features, labels, test_siz
e=0.33, random_state=42)
```

In [7]:
```
def train_model(classifier, feature_vector_train, label, feature_vector_valid
):
    # fit the training dataset on the classifier
    classifier.fit(feature_vector_train, label)

    # predict the labels on validation dataset
    predictions = classifier.predict(feature_vector_valid)

    return metrics.accuracy_score(predictions, y_test)

predictions = train_model(xgboost.XGBClassifier(), X_train.tocsc(), y_train, X
_test.tocsc())


accuracy = accuracy_score(y_test, predictions)
F1 = f1_score(y_test, predictions)
precision = precision_score(y_test, predictions)
recall = recall_score((y_test, predictions))

print ("NB:")
print ("Accuracy: ", accuracy)
print ("F1: ", F1)
print ("Precision: ", precision)
print ("Recall: ", recall)
```

```
/home/iman_lau/anaconda3/lib/python3.5/site-packages/sklearn/preprocessing/la
bel.py:151: DeprecationWarning: The truth value of an empty array is ambiguou
s. Returning False, but in future this will result in an error. Use `array.si
ze > 0` to check that an array is not empty.
  if diff:

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-7-1e3489979976> in <module>()
      8     return metrics.accuracy_score(predictions, y_test)
      9
---> 10 accuracy = train_model(xgboost.XGBClassifier(), X_train.tocsc(), y_tr
ain, X_test.tocsc())
     11 print("Xgb, Count Vectors: ", accuracy)

<ipython-input-7-1e3489979976> in train_model(classifier, feature_vector_trai
n, label, feature_vector_valid)
      6     predictions = classifier.predict(feature_vector_valid)
      7
----> 8     return metrics.accuracy_score(predictions, y_test)
      9
     10 accuracy = train_model(xgboost.XGBClassifier(), X_train.tocsc(), y_tr
ain, X_test.tocsc())

NameError: name 'metrics' is not defined
```