In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
from sklearn.cluster import KMeans
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score

from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.metrics import classification_report
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix
import pandas as pd
from sklearn.model_selection import train_test_split
import numpy as np
from sklearn.svm import LinearSVC
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
import pickle
```

In [2]:

```


# Read in the data
df=pd.read_excel("shuffled_gunlaw.xlsx")
df.columns=['i','date','Tweet','label']
print (f"Shape of dataframe is {df.shape}")
df.head()
X=df['Tweet']


```

```
Shape of dataframe is (13685, 4)
```

In [18]:

```
import re

import spacy

nlp = spacy.load('en_core_web_sm')

processed_tweets=[]

for tweet in range(0, len(X)):
    processed_tweet = re.sub(r'\W', ' ', str(X[tweet]))


    # Remove all the special characters

    processed_tweet = re.sub(r'http\S+', ' ', processed_tweet)

    #processed_tweet = re.sub(r'https?:\/\/+', ' ', processed_tweet)

    #processed_tweet=re.sub(r'\w+:\/{2}[\d\w-]+(\.[\d\w-]+)*(?:(?:\/[^\s/]*))*',

    processed_tweet=re.sub(r'www\S+', ' ', processed_tweet)

    processed_tweet=re.sub(r'co \S+', ' ', processed_tweet)
    # remove all single characters
    processed_tweet = re.sub(r'\s+[a-zA-Z]\s+', ' ', processed_tweet)

    # Remove single characters from the start
    processed_tweet = re.sub(r'\^[a-zA-Z]\s+', ' ', processed_tweet)

    # Substituting multiple spaces with single space
    processed_tweet= re.sub(r'\s+', ' ', processed_tweet, flags=re.I)

    # Removing prefixed 'b'
    processed_tweet = re.sub(r'^b\s+', ' ', processed_tweet)

    processed_tweet = re.sub(r'\d','',processed_tweet)

    processed_tweet= re.sub(r'\s+', ' ', processed_tweet, flags=re.I)


    # Converting to Lowercase
    processed_tweet = processed_tweet.lower()

    processed_tweets.append(processed_tweet)

print (processed_tweets)
```

```
[' sensanders bernie promoting federal tax on anyone improving and selling di
lapidated homes bernie is competing with fellow socialist maduro to see who c
an crush their populations quickest drainthedeepstate fridayfeeling election
democraticdebate guncontrol', 'rt mcjovy wonder how fetus lovers defend khidr
s cold blooded murder of some brat al kahf what would they say if khidr was a
round ', 'rt chronovarience texas mass shooting survivor lobbies congress for
```

```
less gun control notonemore enough bullym ', 'the next time you hear an elite
or wealthy democrat call for guncontrol please remind them that guncontrol wa
s founded to disarm freed slaves an excellent interview below mrcolionnoir dl
oesch stacyontheright nra', ' olofsdotterk royarahmani nzambassadorus marshab
lackburn emilyslist repspanberger rephoulahan repelaineluria yoyo_ma speakerp
elosi jeffreygoldberg speakerpelosi says trump called today about gunviolence
theatlanticfest ukraine', 'arizona state representative jen longdon is gunvio
lence survivor and real leader in the fight to end this epidemic tomorrow way
s amp means will hear her story and take action take look ', ' kamalaharris l
ot more if senatemajldr and senategop are stupid enough to pass your worthles
s new laws a ashallnotbeinfringed guncontrol aids criminals', 'ugh straight t
o the heart gopcomplicittraitors feels gopcorruption gopcomplicit gopcowards
guncontrolnow gunviolence nrabloodmoney nraisaterroristorganization moscowmit
```

In [20]:

```python
import csv
a=df['i']
d=df['date']
l=df['label']
i=0
for entry in processed_tweets:
    with open ('f_a1.csv','a', encoding="utf-8") as res:
        writer=csv.writer(res)
        s="{},{},{},{}\n".format(a[i],d[i],entry,l[i])
        res.write(s)
        print (s)
    i+=1
```

```
0,2019-09-20, sensanders bernie promoting federal tax on anyone improving and
selling dilapidated homes bernie is competing with fellow socialist maduro to
see who can crush their populations quickest drainthedeepstate fridayfeeling
election democraticdebate guncontrol,for

1,2019-09-28,rt mcjovy wonder how fetus lovers defend khidrs cold blooded mur
der of some brat al kahf what would they say if khidr was around ,against

2,2019-09-23,rt chronovarience texas mass shooting survivor lobbies congress
for less gun control notonemore enough bullym ,for

3,2019-09-27,the next time you hear an elite or wealthy democrat call for gun
control please remind them that guncontrol was founded to disarm freed slaves
an excellent interview below mrcolionnoir dloesch stacyontheright nra,for

4,2019-09-24, olofsdotterk royarahmani nzambassadorus marshablackburn emilysl
ist repspanberger rephoulahan repelaineluria yoyo_ma speakerpelosi jeffreygol
dberg speakerpelosi says trump called today about gunviolence theatlanticfest
ukraine,for
```

In [21]:

```
import csv
import pandas as pd
import spacy

nlp = spacy.load('en_core_web_sm')

df=pd.read_csv('f_a1.csv')
df.columns=['index','date','tweet','label']
#df = df.sample(frac=0.1, random_state=10)

print (df.head())

tweets=df['tweet']

import spacy

nlp = spacy.load('en_core_web_sm')
i=0
count=0

list2=[]
for tweet in tweets:
    doc = nlp(tweet)
    list1=[]
    for token in doc:
            if token.is_stop==False:
                print(token.text)
                list1.append(token.text)
    list2.append(list1)
```

```
   index        date                                              tweet  \
0      1  2019-09-28  rt mcjovy wonder how fetus lovers defend khidr...
1      2  2019-09-23  rt chronovarience texas mass shooting survivor...
2      3  2019-09-27  the next time you hear an elite or wealthy dem...
3      4  2019-09-24   olofsdotterk royarahmani nzambassadorus marsh...
4      5  2019-09-25  arizona state representative jen longdon is gu...

     label
0  against
1      for
2      for
3      for
4      for
rt
mcjovy
wonder
fetus
lovers
defend
```

In [23]:

```
a=df['index']
d=df['date']
l=df['label']
i=0
for entry in list2:
    with open ('f_a2.csv','a',encoding="utf-8") as res:
        writer=csv.writer(res)
        s="{},{},{},{}\n".format(a[i],d[i],' '.join(entry),l[i])
        res.write(s)
        print (s)
    i+=1
```

```
1,2019-09-28,rt mcjovy wonder fetus lovers defend khidrs cold blooded murder
brat al kahf khidr,against

2,2019-09-23,rt chronovarience texas mass shooting survivor lobbies congress
gun control notonemore bullym,for

3,2019-09-27,time hear elite wealthy democrat guncontrol remind guncontrol fo
unded disarm freed slaves excellent interview mrcolionnoir dloesch stacyonthe
right nra,for

4,2019-09-24,  olofsdotterk royarahmani nzambassadorus marshablackburn emilys
list repspanberger rephoulahan repelaineluria yoyo_ma speakerpelosi jeffreygo
ldberg speakerpelosi says trump called today gunviolence theatlanticfest ukra
ine,for

5,2019-09-25,arizona state representative jen longdon gunviolence survivor re
al leader fight end epidemic tomorrow ways amp means hear story action look,f
or
```

In [24]:

```
import pandas as pd
import numpy as np

# Read in the data
df = pd.read_csv('f_a2.csv')
df.columns=['index','date','Tweet','label']
print (f"Shape of dataframe is {df.shape}")
df.head()
X=df['Tweet']
Z=df['Tweet'].to_string(index=False)
print (Z)
```

```
Shape of dataframe is (13683, 4)
 rt chronovarience texas mass shooting survivor...
 time hear elite wealthy democrat guncontrol re...
   olofsdotterk royarahmani nzambassadorus mars...
 arizona state representative jen longdon gunvi...
   kamalaharris lot senatemajldr senategop stup...
 ugh straight heart gopcomplicittraitors feels ...
 democrats jumping board guncontrol surprising ...
 rt gun_control_ca doctors speak truth lines co...
  rt dgolumbia perfect libertarian internetfreedom
                     believe guys marchforourlives
 thanks comicdavesmith scotthortonshow antiwarc...
 rt perspectvz repteddeutch gop protectourdemoc...
 conservative candidate bringing american nra g...
   ayoda repdmp everytown point didn want tell ...
 know subject business making laws restrict fre...
 friendly reminder guncontrol confiscation gone...
   nickcarter support guncontrol think guys kil...
 rt forthewin poor people voting democrat years...
```

In [25]:

```
df
```

Out[25]:

	index	date	Tweet	label
0	2	2019-09-23	rt chronovarience texas mass shooting survivor...	for
1	3	2019-09-27	time hear elite wealthy democrat guncontrol re...	for
2	4	2019-09-24	olofsdotterk royarahmani nzambassadorus mars...	for
3	5	2019-09-25	arizona state representative jen longdon gunvi...	for
4	6	2019-09-20	kamalaharris lot senatemajldr senategop stup...	for
5	7	2019-09-26	ugh straight heart gopcomplicittraitors feels ...	for
6	8	2019-09-19	democrats jumping board guncontrol surprising ...	for
7	9	2019-09-27	rt gun_control_ca doctors speak truth lines co...	for
8	10	2019-09-27	rt dgolumbia perfect libertarian internetfreedom	against
9	11	2019-09-25	believe guys marchforourlives	for
10	12	2019-09-21	thanks comicdavesmith scotthortonshow antiwarc...	against
11	13	2019-09-27	rt perspectvz repteddeutch gop protectourdemoc...	for
12	14	2019-09-26	conservative candidate bringing american nra g...	for
13	15	2019-09-26	ayoda repdmp everytown point didn want tell ...	for
14	16	2019-09-24	know subject business making laws restrict fre...	for
15	17	2019-09-22	friendly reminder guncontrol confiscation gone...	for
16	18	2019-09-21	nickcarter support guncontrol think guys kil...	for
17	19	2019-09-27	rt forthewin poor people voting democrat years...	against
18	20	2019-09-25	karijoys purple doves scotland share playing...	for
19	21	2019-09-24	realdonaldtrump moscowmitch ones playing tim...	for
20	22	2019-09-26	betoorourke place firearm developed kill peo...	against
21	23	2019-09-27	ndamendment secondamendment americas freedom	against
22	24	2019-09-26	know clemetroschools students wrote produced p...	for
23	25	2019-09-20	know pediatric vaccine mmr ingredient thimeros...	against
24	26	2019-09-26	rt bremaininspain saturdaysatire thank banbury...	for
25	27	2019-09-27	asshat betoorourkes idea ndamendment actually ...	against
26	28	2019-09-22	chicago gun violence teens learning responder ...	for
27	29	2019-09-19	rt gigi thehill guncontrol ashallnotbeinfringe...	for
28	30	2019-09-20	rt rosaare bro dignity drop progun prolife bet...	against
29	31	2019-09-27	weeks ago important outside hospital castlebar...	against
...	...	...	...	...
13653	13655	2019-09-22	pulse survivor brandonwolf speaks wesh deliv...	for
13654	13656	2019-09-25	democrats destroy atomic bombs trump maga demo...	against
13655	13657	2019-09-23	rt proa_tactical tactical kinetics inch wylde ...	against

	index	date	Tweet	label
13656	13658	2019-09-26	betray ignorance dishonesty single day guns gu...	for
13657	13659	2019-09-27	rt afthealthcare compelling testimony dr aleja...	for
13658	13660	2019-09-27	having said americans stand ve said change gun...	for
13659	13661	2019-09-27	driveby outside daughters high school home get...	for
13660	13662	2019-09-20	marcgarneau m guncontrol advocate sees issue...	for
13661	13663	2019-09-26	dr john lotts testimony pennsylvania senate ju...	for
13662	13664	2019-09-27	rt barnettforaz thank support kelliwardaz kind...	against
13663	13665	2019-09-21	hey betoorourke rest people think banning ars ...	against
13664	13666	2019-09-26	rt nationalist democratic socialist party supp...	against
13665	13667	2019-09-20	planning going shooting turning gun save elses...	against
13666	13668	2019-09-22	rid homelessness good pensignal medium medium ...	against
13667	13669	2019-09-27	adefender gone traitor cliff deportthemall p...	against
13668	13670	2019-09-19	guns save lives armed citizens save lives day ...	against
13669	13671	2019-09-23	republicans wants shoot minorities downyou kno...	for
13670	13672	2019-09-20	rt cbwords anti gun twits said nt coming weapo...	for
13671	13673	2019-09-27	mentalhealthawareness nami released formal s...	for
13672	13674	2019-09-27	term libertarian misused marxists marxist left...	against
13673	13675	2019-09-19	terribly sad terribly real life major reasons ...	for
13674	13676	2019-09-20	smith_wessoninc palmettoarmory stop making ar ...	against
13675	13677	2019-09-26	trump shoots fifth ave trump supporters libera...	for
13676	13678	2019-09-19	got ta watch guncontrol	for
13677	13679	2019-09-20	bye comrade felicia aka bill de blasio miss ar...	for
13678	13680	2019-09-27	reprochoiceau abortion mothers premeditated ...	against
13679	13681	2019-09-26	bought subscriptions amee awesome output impea...	for
13680	13682	2019-09-22	rt conserv_tribune homeowner retired los angel...	for
13681	13683	2019-09-20	rt timjdillon meghan mccain stands second amen...	against
13682	13684	2019-09-19	extremeriskprotectionorders erpo aka redflag...	for

13683 rows × 4 columns

In [27]:

```
import csv
import pandas as pd
import spacy

nlp = spacy.load('en_core_web_sm')


df=pd.read_csv('f_a2.csv')
df.columns=['index','date','Tweet','label']
A=df['date']
B=df['index']
C=df['label']
tweets=df['Tweet']

import spacy

nlp = spacy.load('en_core_web_sm')
i=0
j=0
for tweet in tweets:



    count=0
    countadj=0
    countverb=0
    countadp=0
    countadv=0
    countnum=0
    countaux=0
    countconj=0
    countdet=0
    countintj=0
    countpart=0
    countpron=0
    countpropn=0
    countpropn=0
    countpunct=0
    countsconj=0
    countx=0
    doc = nlp(tweet)
    for token in doc:
            if token.pos_=='NOUN':
                count+=1
            if token.pos_=='ADJ':
                countadj+=1
            if token.pos_=='VERB':
                countverb+=1
            if token.pos_=='ADP':
                countadp+=1
            if token.pos_=='ADV':
                countadv+=1
            if token.pos_=='NUM':
                countnum+=1
            if token.pos_=='AUX':
                countaux+=1
            if token.pos_=='CONJ':
                countconj+=1
            if token.pos_=='DET':
                countdet+=1
            if token.pos_=='INTJ':
                countintj+=1
            if token.pos_=='PART':
                countpart+=1
            if token.pos_=='PRON':
                countpron+=1
            if token.pos_=='PROPN':
                countpropn+=1
            if token.pos_=='PUNCT':
                countpunct+=1
            if token.pos_=='SCONJ':
                countsconj+=1
            if token.pos_=='X':
                countx+=1

    print (f"nouns in tweet at {i} index are {count} verbs are {countverb} adject

    with open ('f_a3.csv','a',encoding="utf-8") as res:
        from textblob import TextBlob
        analysis = TextBlob(tweet)
        if C[i]=='for':
            label=1
        else:
            label=0
        s="{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{}\n".form
        res.write(s)
    i+=1
```

```
nouns in tweet at 0 index are 9 verbs are 1 adjectives are 1 adpositions are
0 adverbs are 1 numerals are 0
nouns in tweet at 1 index are 11 verbs are 3 adjectives are 4 adpositions are
0 adverbs are 0 numerals are 0
nouns in tweet at 2 index are 11 verbs are 4 adjectives are 3 adpositions are
0 adverbs are 0 numerals are 0
nouns in tweet at 3 index are 16 verbs are 2 adjectives are 2 adpositions are
0 adverbs are 0 numerals are 0
nouns in tweet at 4 index are 7 verbs are 3 adjectives are 3 adpositions are
0 adverbs are 0 numerals are 0
nouns in tweet at 5 index are 10 verbs are 2 adjectives are 2 adpositions are
0 adverbs are 0 numerals are 0
nouns in tweet at 6 index are 11 verbs are 3 adjectives are 1 adpositions are
0 adverbs are 1 numerals are 0
nouns in tweet at 7 index are 8 verbs are 1 adjectives are 2 adpositions are
0 adverbs are 0 numerals are 0
nouns in tweet at 8 index are 1 verbs are 1 adjectives are 3 adpositions are
0 adverbs are 0 numerals are 0
nouns in tweet at 9 index are 2 verbs are 1 adjectives are 0 adpositions are
```

In [29]:

```
df = pd.read_csv('f_a3.csv')
df.columns=['index','date','tweet','countnoun','countverb','countadj','countadp',
df
```

Out[29]:

	index	date	tweet	countnoun	countverb	countadj	countadp	cou
0	3	2019-09-27	time hear elite wealthy democrat guncontrol re...	11	3	4	0	
1	4	2019-09-24	olofsdotterk royarahmani nzambassadorus mars...	11	4	3	0	
2	5	2019-09-25	arizona state representative jen longdon gunvi...	16	2	2	0	
3	6	2019-09-20	kamalaharris lot senatemajldr senategop stup...	7	3	3	0	
4	7	2019-09-26	ugh straight heart gopcomplicittraitors feels ...	10	2	2	0	
5	8	2019-09-19	democrats jumping board guncontrol surprising ...	11	3	1	0	
6	9	2019-09-27	rt gun_control_ca doctors speak truth lines co	8	1	2	0	

In [30]:

```
feature_names_df = ['countnoun','countverb','countadj','countadp','countadv','cou
x_df = df[feature_names_df]
y_df = df['target']
```

In [31]:

```
from sklearn.model_selection import train_test_split


X_train, X_test, y_train, y_test = train_test_split(x_df, y_df, random_state=0)
```

In [42]:

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(solver='lbfgs', multi_class='auto',max_iter=100)
model.fit(X_train,y_train)
```

```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\sklearn
\linear_model\logistic.py:947: ConvergenceWarning: lbfgs failed to converge. In
crease the number of iterations.
  "of iterations.", ConvergenceWarning)
```

Out[42]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [43]:

```
from sklearn.metrics import roc_auc_score
from sklearn import preprocessing

def multiclass_roc_auc_score(y_test, y_pred, average="macro"):
    lb = preprocessing.LabelBinarizer()
    lb.fit(y_test)
    y_test = lb.transform(y_test)
    y_pred = lb.transform(y_pred)
    return roc_auc_score(y_test, y_pred, average=average)

# Predict the transformed test documents
predictions = model.predict((X_test))

print('AUC: ', multiclass_roc_auc_score(y_test, predictions))
```

```
AUC:  0.6015950423885251
```

In [53]:

```
from sklearn.ensemble import RandomForestClassifier

model=RandomForestClassifier(n_estimators=200,criterion='entropy')
model.fit(X_train,y_train)
```

Out[53]:

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='entropy',
                       max_depth=None, max_features='auto', max_leaf_nodes=Non
e,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=200,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

In [54]:

```
from sklearn.metrics import roc_auc_score
from sklearn import preprocessing

def multiclass_roc_auc_score(y_test, y_pred, average="macro"):
    lb = preprocessing.LabelBinarizer()
    lb.fit(y_test)
    y_test = lb.transform(y_test)
    y_pred = lb.transform(y_pred)
    return roc_auc_score(y_test, y_pred, average=average)

# Predict the transformed test documents
predictions = model.predict((X_test))

print('AUC: ', multiclass_roc_auc_score(y_test, predictions))
```

```
AUC:  0.6548388934103925
```

In [81]:

```
prediction_text="Did you know @CLEMetroSchools students wrote produced and perfor
pts=[]
pt = re.sub(r'\W', ' ', str(prediction_text))
pt = re.sub(r'http\S+', ' ', pt)
pt=re.sub(r'www\S+', ' ', pt)
pt=re.sub(r'co \S+', ' ', pt)
pt = re.sub(r'\s+[a-zA-Z]\s+', ' ', pt)
pt = re.sub(r'\^[a-zA-Z]\s+', ' ', pt)
pt= re.sub(r'\s+', ' ', pt, flags=re.I)
pt = re.sub(r'^b\s+', ' ', pt)
pt = re.sub(r'\d','',pt)
pt= re.sub(r'\s+', ' ', pt, flags=re.I)
pt = pt.lower()
pts.append(pt)
#print (pt)
nlp = spacy.load('en_core_web_sm')
doc = nlp(pt)
list3=[]
list4=[]
for token in doc:
        if token.is_stop==False:
            #print(token.text)
            list3.append(token.text)
#print (pt)
list3=' '.join(list3)
print (list3)

countnoun=0
countadj=0
countverb=0
countadp=0
countadv=0
countnum=0
countaux=0
countconj=0
countdet=0
countintj=0
countpart=0
countpron=0
countpropn=0
countpropn=0
countpunct=0
countsconj=0
countx=0
doc = nlp(list3)
for token in doc:
            if token.pos_=='NOUN':
                countnoun+=1
            if token.pos_=='ADJ':
                countadj+=1
            if token.pos_=='VERB':
                countverb+=1
            if token.pos_=='ADP':
                countadp+=1
            if token.pos_=='ADV':
                countadv+=1
            if token.pos_=='NUM':
                countnum+=1
            if token.pos_=='AUX':
                countaux+=1
            if token.pos_=='CONJ':
                countconj+=1
            if token.pos_=='DET':
                countdet+=1
            if token.pos_=='INTJ':
                countintj+=1
            if token.pos_=='PART':
                countpart+=1
            if token.pos_=='PRON':
                countpron+=1
            if token.pos_=='PROPN':
                countpropn+=1
            if token.pos_=='PUNCT':
                countpunct+=1
            if token.pos_=='SCONJ':
                countsconj+=1
            if token.pos_=='X':
                countx+=1

from textblob import TextBlob
analysis = TextBlob(list3)
s="{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{},{}".format(countnoun,countverb
print (s)
```

```
know clemetroschools students wrote produced performed play gunviolence incredi
ble accomplishments ericgordon_ceo detailing remarks thecityclub
7,6,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0.9
```

In [83]:

```
l=(model.predict([[countnoun,countverb,countadj,countadp,countadv,countnum,counta
if l==1:
    print ("For")
else:
    print ("Against")
```

```
For
```

In [84]:

```
## Count Vectorizer
```

In [3]:
```
df = pd.read_csv('f_a2.csv')
df.columns=['index','date','tweet','target']
df
```

Out[3]:

	index	date	tweet	target
0	2	2019-09-23	rt chronovarience texas mass shooting survivor...	for
1	3	2019-09-27	time hear elite wealthy democrat guncontrol re...	for
2	4	2019-09-24	olofsdotterk royarahmani nzambassadorus mars...	for
3	5	2019-09-25	arizona state representative jen longdon gunvi...	for
4	6	2019-09-20	kamalaharris lot senatemajldr senategop stup...	for
5	7	2019-09-26	ugh straight heart gopcomplicittraitors feels ...	for
6	8	2019-09-19	democrats jumping board guncontrol surprising ...	for
7	9	2019-09-27	rt gun_control_ca doctors speak truth lines co...	for
8	10	2019-09-27	rt dgolumbia perfect libertarian internetfreedom	against
9	11	2019-09-25	believe guys marchforourlives	for
10	12	2019-09-21	thanks comicdavesmith scotthortonshow antiwarc...	against
11	13	2019-09-27	rt perspectvz repteddeutch gop protectourdemoc...	for
12	14	2019-09-26	conservative candidate bringing american nra g...	for
13	15	2019-09-26	ayoda repdmp everytown point didn want tell ...	for
14	16	2019-09-24	know subject business making laws restrict fre...	for
15	17	2019-09-22	friendly reminder guncontrol confiscation gone...	for
16	18	2019-09-21	nickcarter support guncontrol think guys kil...	for
17	19	2019-09-27	rt forthewin poor people voting democrat years...	against
18	20	2019-09-25	karijoys purple doves scotland share playing...	for
19	21	2019-09-24	realdonaldtrump moscowmitch ones playing tim...	for
20	22	2019-09-26	betoorourke place firearm developed kill peo...	against
21	23	2019-09-27	ndamendment secondamendment americas freedom	against
22	24	2019-09-26	know clemetroschools students wrote produced p...	for
23	25	2019-09-20	know pediatric vaccine mmr ingredient thimeros...	against
24	26	2019-09-26	rt bremaininspain saturdaysatire thank banbury...	for
25	27	2019-09-27	asshat betoorourkes idea ndamendment actually ...	against
26	28	2019-09-22	chicago gun violence teens learning responder ...	for
27	29	2019-09-19	rt gigi thehill guncontrol ashallnotbeinfringe...	for
28	30	2019-09-20	rt rosaare bro dignity drop progun prolife bet...	against
29	31	2019-09-27	weeks ago important outside hospital castlebar...	against
...	...	...	...	...
13653	13655	2019-09-22	pulse survivor brandonwolf speaks wesh deliv...	for
13654	13656	2019-09-25	democrats destroy atomic bombs trump maga demo...	against

	index	date	tweet	target
13655	13657	2019-09-23	rt proa_tactical tactical kinetics inch wylde ...	against
13656	13658	2019-09-26	betray ignorance dishonesty single day guns gu...	for
13657	13659	2019-09-27	rt afthealthcare compelling testimony dr aleja...	for
13658	13660	2019-09-27	having said americans stand ve said change gun...	for
13659	13661	2019-09-27	driveby outside daughters high school home get...	for
13660	13662	2019-09-20	marcgarneau m guncontrol advocate sees issue...	for
13661	13663	2019-09-26	dr john lotts testimony pennsylvania senate ju...	for
13662	13664	2019-09-27	rt barnettforaz thank support kelliwardaz kind...	against
13663	13665	2019-09-21	hey betoorourke rest people think banning ars ...	against
13664	13666	2019-09-26	rt nationalist democratic socialist party supp...	against
13665	13667	2019-09-20	planning going shooting turning gun save elses...	against
13666	13668	2019-09-22	rid homelessness good pensignal medium medium ...	against
13667	13669	2019-09-27	adefender gone traitor cliff deportthemall p...	against
13668	13670	2019-09-19	guns save lives armed citizens save lives day ...	against
13669	13671	2019-09-23	republicans wants shoot minorities downyou kno...	for
13670	13672	2019-09-20	rt cbwords anti gun twits said nt coming weapo...	for
13671	13673	2019-09-27	mentalhealthawareness nami released formal s...	for
13672	13674	2019-09-27	term libertarian misused marxists marxist left...	against
13673	13675	2019-09-19	terribly sad terribly real life major reasons ...	for
13674	13676	2019-09-20	smith_wessoninc palmettoarmory stop making ar ...	against
13675	13677	2019-09-26	trump shoots fifth ave trump supporters libera...	for
13676	13678	2019-09-19	got ta watch guncontrol	for
13677	13679	2019-09-20	bye comrade felicia aka bill de blasio miss ar...	for
13678	13680	2019-09-27	reprochoiceau abortion mothers premeditated ...	against
13679	13681	2019-09-26	bought subscriptions amee awesome output impea...	for
13680	13682	2019-09-22	rt conserv_tribune homeowner retired los angel...	for
13681	13683	2019-09-20	rt timjdillon meghan mccain stands second amen...	against
13682	13684	2019-09-19	extremeriskprotectionorders erpo aka redflag...	for

13683 rows × 4 columns

```
In [4]: V=df.tweet.values.astype('U')
        cat=df.target
```

In [10]:
```
def mlpclassifier(X_train, X_test, y_train, y_test):

  from sklearn.neural_network import MLPClassifier

  mlp = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('clf', MLPClassifier(hidden_layer_sizes=(50,50,50))),
                 ])
  mlp.fit(X_train, y_train)

  y_pred = mlp.predict(X_test)



  print('accuracy %s' % accuracy_score(y_pred, y_test))
```

In [11]:
```
def train_test(X,y):

  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random


  print("Results of MLP Classifier")
  mlpclassifier(X_train, X_test, y_train, y_test)
```

In [12]:
```
train_test(V,cat)
```

```
Results of MLP Classifier
accuracy 0.9247259439707674
```

In [6]:
```
from sklearn.model_selection import train_test_split

# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(df['tweet'],
                                                    df['target'],
                                                    random_state=0)
```

In [89]:
```
print('X_train first entry:\n\n', X_train.iloc[0])
print('\n\nX_train shape: ', X_train.shape)
```

```
X_train first entry:

   truth trust adefender constitution maga usa comeandgetit rt stand secondamen
dment


X_train shape:  (10262,)
```

In [90]:
```
from sklearn.feature_extraction.text import CountVectorizer

# Fit the CountVectorizer to the training data
vect = CountVectorizer().fit(X_train)
```

In [91]:
```
print (vect.get_feature_names())
```

```
['a_j_christ', 'aa_pilot', 'aac', 'aacap', 'aacaporg', 'aacaps', 'aafp', 'aaf
pcod', 'aafpfmx', 'aapdelmonte', 'aaplog', 'aaplog_fms', 'aapublishingllc',
'aarmark', 'aaron', 'aaron_kinney', 'aaronbergcomedy', 'aaroncarter', 'aarp',
'aarpadvocates', 'aarthiswami', 'aarxn', 'aast', 'ab', 'aba', 'abandon', 'aba
ndoned', 'abapre', 'abapresident', 'abas', 'abated', 'abating', 'abbeludwig',
'abbott', 'abby', 'abbyjohnson', 'abc', 'abcaustralia', 'abcnews', 'abcthedru
m', 'abcworldnews', 'abdicated', 'abdication', 'abducted', 'abductions', 'abe
froman', 'abetting', 'abeylane', 'abide', 'abiding', 'abilities', 'ability',
'able', 'abnormal', 'abo', 'aboard', 'abolish', 'abolishabortion', 'abolishab
ortionglobally', 'abolished', 'abolishfilibuster', 'abolishing', 'abolishmen
t', 'abolishtheatf', 'abolishthefed', 'abolishtheirs', 'abolition', 'abominab
le', 'abor', 'abort', 'aborted', 'aborting', 'abortingamerica', 'abortio', 'a
bortion', 'abortionbill', 'abortioncrimeagainsthumanity', 'abortionfree', 'ab
ortionhurtswomen', 'abortionis', 'abortionisawomansright', 'abortionishealthc
are', 'abortionishealthcareke', 'abortionismurd', 'abortionismurde', 'abortio
nismurder', 'abortionisnotbirthcontrol', 'abortionisnothealtchare', 'abortion
isnothealthcare', 'abortionissin', 'abortionist', 'abortionistorture', 'abort
ionists', 'abortionnsw', 'abortionpill', 'abortionpillreversal', 'abortionref
usal', 'abortionrights', 'abortions', 'abortionthe', 'abortive', 'abortuary',
```

In [92]:
```
print (len(vect.get_feature_names()))
```

```
22119
```

In [93]:
```
# transform the documents in the training data to a document-term matrix
X_train_vectorized = vect.transform(X_train)

X_train_vectorized

print ((X_train_vectorized.shape))
```

```
(10262, 22119)
```

In [94]:
```
from sklearn.linear_model import LogisticRegression

# Train the model
model = LogisticRegression(solver='lbfgs', multi_class='auto')
model.fit(X_train_vectorized, y_train)
```

Out[94]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [95]:

```
from sklearn.metrics import roc_auc_score
from sklearn import preprocessing

def multiclass_roc_auc_score(y_test, y_pred, average="macro"):
    lb = preprocessing.LabelBinarizer()
    lb.fit(y_test)
    y_test = lb.transform(y_test)
    y_pred = lb.transform(y_pred)
    return roc_auc_score(y_test, y_pred, average=average)

# Predict the transformed test documents
predictions = model.predict(vect.transform(X_test))

print('AUC: ', multiclass_roc_auc_score(y_test, predictions))
```

```
AUC:  0.9537041802187547
```

In [96]:

```
# get the feature names as numpy array
feature_names = np.array(vect.get_feature_names())

# Sort the coefficients from the model
sorted_coef_index = model.coef_[0].argsort()

# Find the 10 smallest and 10 largest coefficients
# The 10 largest coefficients are being indexed using [:-11:-1]
# so the list returned is in order of largest to smallest
print('Smallest Coefs:\n{}\n'.format(feature_names[sorted_coef_index[:10]]))
print('Largest Coefs: \n{}'.format(feature_names[sorted_coef_index[:-11:-1]]))
```

```
Smallest Coefs:
['prolife' 'libertarian' 'adefender' 'ar' 'ndamendment' 'sharpe_way'
 'abortion' 'tenthamendment' 'heytootssweet' 'plumremson']

Largest Coefs:
['guncontrol' 'gunviolence' 'marchforourlives' 'antigun' 'ourbestbeto'
 'escapedmatrix' 'gunskillpeople' 'senatemajldr' 'starting' 'school']
```

In [97]:

```
prediction_text="Did you know @CLEMetroSchools students wrote produced and perfor
pts=[]
pt = re.sub(r'\W', ' ', str(prediction_text))
pt = re.sub(r'http\S+', ' ', pt)
pt=re.sub(r'www\S+', ' ', pt)
pt=re.sub(r'co \S+', ' ', pt)
pt = re.sub(r'\s+[a-zA-Z]\s+', ' ', pt)
pt = re.sub(r'\^[a-zA-Z]\s+', ' ', pt)
pt= re.sub(r'\s+', ' ', pt, flags=re.I)
pt = re.sub(r'^b\s+', ' ', pt)
pt = re.sub(r'\d','',pt)
pt= re.sub(r'\s+', ' ', pt, flags=re.I)
pt = pt.lower()
pts.append(pt)
#print (pt)
nlp = spacy.load('en_core_web_sm')
doc = nlp(pt)
list5=[]
for token in doc:
        if token.is_stop==False:
            #print(token.text)
            list5.append(token.text)
#print (pt)
list5=' '.join(list5)
print (list5)


# These reviews are treated the same by our current model

print(model.predict(vect.transform([list5])))
```

```
know clemetroschools students wrote produced performed play gunviolence incredi
ble accomplishments ericgordon_ceo detailing remarks thecityclub
['for']
```

In [98]:

```
### Tfidf
```

In [99]:

```
from sklearn.feature_extraction.text import TfidfVectorizer

# Fit the TfidfVectorizer to the training data specifiying a minimum document fre
vect = TfidfVectorizer(min_df=5).fit(X_train)
len(vect.get_feature_names())
```

Out[99]:

```
4410
```

In [100]:

```
X_train_vectorized = vect.transform(X_train)

from sklearn.metrics import roc_auc_score
from sklearn import preprocessing

def multiclass_roc_auc_score(y_test, y_pred, average="macro"):
    lb = preprocessing.LabelBinarizer()
    lb.fit(y_test)
    y_test = lb.transform(y_test)
    y_pred = lb.transform(y_pred)
    return roc_auc_score(y_test, y_pred, average=average)

model = LogisticRegression()
model.fit(X_train_vectorized, y_train)

predictions = model.predict(vect.transform(X_test))

print('AUC: ', multiclass_roc_auc_score(y_test, predictions))
```

```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\sklearn
\linear_model\logistic.py:432: FutureWarning: Default solver will be changed to
'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)

AUC:  0.9596089450642953
```

In [101]:

```
feature_names = np.array(vect.get_feature_names())

sorted_tfidf_index = X_train_vectorized.max(0).toarray()[0].argsort()

print('Smallest tfidf:\n{}\n'.format(feature_names[sorted_tfidf_index[:10]]))
print('Largest tfidf: \n{}'.format(feature_names[sorted_tfidf_index[:-11:-1]]))
```

```
Smallest tfidf:
['afoxauthor' 'girlpreneur' 'actress' 'splashdwcom' 'staystrong'
 'odesssastrong' 'teepublic' 'idailydesignfashion' 'idailydesignliving'
 'rewire_news']

Largest tfidf:
['ndamendment' 'rt' 'ar' 'prolife' 'marchforourlives' 'gunviolence'
 'libertarian' 'guncontrol' 'fear' 'signs']
```

In [102]:

```
sorted_coef_index = model.coef_[0].argsort()

print('Smallest Coefs:\n{}\n'.format(feature_names[sorted_coef_index[:10]]))
print('Largest Coefs: \n{}'.format(feature_names[sorted_coef_index[:-11:-1]]))
```

```
Smallest Coefs:
['prolife' 'libertarian' 'ar' 'adefender' 'ndamendment' 'abortion' 'life'
 'sharpe_way' 'progun' 'tenthamendment']

Largest Coefs:
['guncontrol' 'gunviolence' 'marchforourlives' 'ourbestbeto' 'gunsense'
 'school' 'violence' 'gun' 'antigun' 'climatechange']
```

In [103]:

```
## CountVectorizer with n-grams
```

In [104]:

```
# Fit the CountVectorizer to the training data specifiying a minimum
# document frequency of 5 and extracting 1-grams and 2-grams
vect = CountVectorizer(min_df=5, ngram_range=(1,2)).fit(X_train)

X_train_vectorized = vect.transform(X_train)

len(vect.get_feature_names())
print (vect.get_feature_names())
```

```
['aafp', 'aarp', 'aast', 'aast presidential', 'abbeludwig', 'abbyjohnson', 'a
bc', 'abc news', 'abide', 'abiding', 'abiding citizen', 'abiding citizens',
'abiding gun', 'ability', 'ability comprehend', 'able', 'able understand', 'a
bolish', 'abolishtheatf', 'abolishtheirs', 'abolishtheirs abolishtheatf', 'ab
ort', 'aborted', 'aborted babies', 'aborting', 'abortion', 'abortion attemp
t', 'abortion clinic', 'abortion demand', 'abortion industry', 'abortion murd
er', 'abortion prolife', 'abortionismurder', 'abortionismurder prolife', 'abo
rtionismurder saveourbabies', 'abortionisnothealthcare', 'abortionist', 'abor
tionists', 'abortionrights', 'abortions', 'absolute', 'absolutely', 'absolute
ly medically', 'absurd', 'abt', 'abuse', 'abused', 'academy', 'accept', 'acce
pting', 'access', 'access guns', 'accidentally', 'according', 'according ne
w', 'account', 'accountable', 'accurate', 'accusations', 'accuse', 'accused',
'acesheepdog', 'acesheepdog dgpurser', 'achievement', 'aclu', 'aclunm', 'aclu
nm nmdoh', 'acp', 'acpinternists', 'act', 'act gunviolence', 'acting', 'actio
n', 'action guncontrol', 'action gunviolence', 'action reduce', 'actions', 'a
ctive', 'active shooter', 'activeshooter', 'activeshooter backtoschool', 'act
ivism', 'activist', 'activists', 'actor', 'actors', 'actress', 'actress afoxa
uthor', 'acts', 'actual', 'actually', 'ad', 'adam', 'adam schiff', 'adamkokes
h', 'adams', 'adamschiff', 'add', 'addition', 'address', 'address gunviolenc
e', 'addressed', 'addresses', 'addressing', 'addressing gunviolence', 'adds'
```

In [105]:

```

X_train_vectorized = vect.transform(X_train)

from sklearn.metrics import roc_auc_score
from sklearn import preprocessing

def multiclass_roc_auc_score(y_test, y_pred, average="macro"):
    lb = preprocessing.LabelBinarizer()
    lb.fit(y_test)
    y_test = lb.transform(y_test)
    y_pred = lb.transform(y_pred)
    return roc_auc_score(y_test, y_pred, average=average)

model = LogisticRegression()
model.fit(X_train_vectorized, y_train)

predictions = model.predict(vect.transform(X_test))

print('AUC: ', multiclass_roc_auc_score(y_test, predictions))
```

```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\sklearn
\linear_model\logistic.py:432: FutureWarning: Default solver will be changed to
'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)

AUC:  0.9524580742997517
```

In [106]:

```
feature_names = np.array(vect.get_feature_names())

sorted_coef_index = model.coef_[0].argsort()

print('Smallest Coefs:\n{}\n'.format(feature_names[sorted_coef_index[:10]]))
print('Largest Coefs: \n{}'.format(feature_names[sorted_coef_index[:-11:-1]]))
```

```
Smallest Coefs:
['prolife' 'libertarian' 'adefender' 'ar' 'ndamendment' 'abortion'
 'sharpe_way' 'progun' 'rt sharpe_way' 'unborn']

Largest Coefs:
['guncontrol' 'gunviolence' 'marchforourlives' 'antigun' 'production ar'
 'gunskillpeople' 'senatemajldr' 'rt ourbestbeto' 'ourbestbeto' 'rt']
```

In [107]:

```
## RandomForest
```

In [108]:

```python
# Fit the CountVectorizer to the training data specifiying a minimum
# document frequency of 5 and extracting 1-grams and 2-grams
vect = CountVectorizer(min_df=5, ngram_range=(1,2)).fit(X_train)

X_train_vectorized = vect.transform(X_train)

len(vect.get_feature_names())



X_train_vectorized = vect.transform(X_train)
X_train_vectorized.todense()
from sklearn.metrics import roc_auc_score
from sklearn import preprocessing

from sklearn.ensemble import RandomForestClassifier


def multiclass_roc_auc_score(y_test, y_pred, average="macro"):
    lb = preprocessing.LabelBinarizer()
    lb.fit(y_test)
    y_test = lb.transform(y_test)
    y_pred = lb.transform(y_pred)
    return roc_auc_score(y_test, y_pred, average=average)

model = RandomForestClassifier(n_estimators=200,criterion='entropy')
model.fit(X_train_vectorized, y_train)

predictions = model.predict(vect.transform(X_test))

print('AUC: ', multiclass_roc_auc_score(y_test, predictions))
```

```
AUC:  0.9514433831501229
```

In [15]:

```
data = pd.read_csv('f_a2.csv')
data.columns=['index','date','tweet','target']
data
```

Out[15]:

	index	date	tweet	target
0	2	2019-09-23	rt chronovarience texas mass shooting survivor...	for
1	3	2019-09-27	time hear elite wealthy democrat guncontrol re...	for
2	4	2019-09-24	olofsdotterk royarahmani nzambassadorus mars...	for
3	5	2019-09-25	arizona state representative jen longdon gunvi...	for
4	6	2019-09-20	kamalaharris lot senatemajldr senategop stup...	for
5	7	2019-09-26	ugh straight heart gopcomplicittraitors feels ...	for
6	8	2019-09-19	democrats jumping board guncontrol surprising ...	for
7	9	2019-09-27	rt gun_control_ca doctors speak truth lines co...	for
8	10	2019-09-27	rt dgolumbia perfect libertarian internetfreedom	against
9	11	2019-09-25	believe guys marchforourlives	for
10	12	2019-09-21	thanks comicdavesmith scotthortonshow antiwarc...	against
11	13	2019-09-27	rt perspectvz repteddeutch gop protectourdemoc...	for
12	14	2019-09-26	conservative candidate bringing american nra g...	for
13	15	2019-09-26	ayoda repdmp everytown point didn want tell ...	for
14	16	2019-09-24	know subject business making laws restrict fre...	for
15	17	2019-09-22	friendly reminder guncontrol confiscation gone...	for
16	18	2019-09-21	nickcarter support guncontrol think guys kil...	for
17	19	2019-09-27	rt forthewin poor people voting democrat years...	against
18	20	2019-09-25	karijoys purple doves scotland share playing...	for
19	21	2019-09-24	realdonaldtrump moscowmitch ones playing tim...	for
20	22	2019-09-26	betoorourke place firearm developed kill peo...	against
21	23	2019-09-27	ndamendment secondamendment americas freedom	against
22	24	2019-09-26	know clemetroschools students wrote produced p...	for
23	25	2019-09-20	know pediatric vaccine mmr ingredient thimeros...	against
24	26	2019-09-26	rt bremaininspain saturdaysatire thank banbury...	for
25	27	2019-09-27	asshat betoorourkes idea ndamendment actually ...	against
26	28	2019-09-22	chicago gun violence teens learning responder ...	for
27	29	2019-09-19	rt gigi thehill guncontrol ashallnotbeinfringe...	for
28	30	2019-09-20	rt rosaare bro dignity drop progun prolife bet...	against
29	31	2019-09-27	weeks ago important outside hospital castlebar...	against
...	...	...	...	...
13653	13655	2019-09-22	pulse survivor brandonwolf speaks wesh deliv...	for
13654	13656	2019-09-25	democrats destroy atomic bombs trump maga demo...	against

	index	date	tweet	target
13655	13657	2019-09-23	rt proa_tactical tactical kinetics inch wylde ...	against
13656	13658	2019-09-26	betray ignorance dishonesty single day guns gu...	for
13657	13659	2019-09-27	rt afthealthcare compelling testimony dr aleja...	for
13658	13660	2019-09-27	having said americans stand ve said change gun...	for
13659	13661	2019-09-27	driveby outside daughters high school home get...	for
13660	13662	2019-09-20	marcgarneau m guncontrol advocate sees issue...	for
13661	13663	2019-09-26	dr john lotts testimony pennsylvania senate ju...	for
13662	13664	2019-09-27	rt barnettforaz thank support kelliwardaz kind...	against
13663	13665	2019-09-21	hey betoorourke rest people think banning ars ...	against
13664	13666	2019-09-26	rt nationalist democratic socialist party supp...	against
13665	13667	2019-09-20	planning going shooting turning gun save elses...	against
13666	13668	2019-09-22	rid homelessness good pensignal medium medium ...	against
13667	13669	2019-09-27	adefender gone traitor cliff deportthemall p...	against
13668	13670	2019-09-19	guns save lives armed citizens save lives day ...	against
13669	13671	2019-09-23	republicans wants shoot minorities downyou kno...	for
13670	13672	2019-09-20	rt cbwords anti gun twits said nt coming weapo...	for
13671	13673	2019-09-27	mentalhealthawareness nami released formal s...	for
13672	13674	2019-09-27	term libertarian misused marxists marxist left...	against
13673	13675	2019-09-19	terribly sad terribly real life major reasons ...	for
13674	13676	2019-09-20	smith_wessoninc palmettoarmory stop making ar ...	against
13675	13677	2019-09-26	trump shoots fifth ave trump supporters libera...	for
13676	13678	2019-09-19	got ta watch guncontrol	for
13677	13679	2019-09-20	bye comrade felicia aka bill de blasio miss ar...	for
13678	13680	2019-09-27	reprochoiceau abortion mothers premeditated ...	against
13679	13681	2019-09-26	bought subscriptions amee awesome output impea...	for
13680	13682	2019-09-22	rt conserv_tribune homeowner retired los angel...	for
13681	13683	2019-09-20	rt timjdillon meghan mccain stands second amen...	against
13682	13684	2019-09-19	extremeriskprotectionorders erpo aka redflag...	for

13683 rows × 4 columns

In [16]:

```
data_text=data[['tweet']]
data_text['index']=data_text.index
documents=data_text
```

```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy (http://pandas.pydat
a.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cop
y)
```

In [17]:

```
print (len(documents))
```

```
13683
```

In [18]:

```
print (documents[:5])
```

```
                                               tweet  index
0  rt chronovarience texas mass shooting survivor...      0
1  time hear elite wealthy democrat guncontrol re...      1
2    olofsdotterk royarahmani nzambassadorus mars...      2
3  arizona state representative jen longdon gunvi...      3
4    kamalaharris lot senatemajldr senategop stup...      4
```

In [20]:

```
import gensim
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS
from nltk.stem import WordNetLemmatizer, SnowballStemmer
from nltk.stem.porter import *
import numpy as np
np.random.seed(2018)
import nltk
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\DELL\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[20]:

```
True
```

In [23]:

```
def lemmatize_stemming(text):
    stemmer = PorterStemmer()
    return stemmer.stem(WordNetLemmatizer().lemmatize(text, pos='v'))
def preprocess(text):
    result = []
    for token in gensim.utils.simple_preprocess(text):
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3
            result.append(lemmatize_stemming(token))
    return result
```

In [24]:
```
doc_sample = documents[documents['index'] == 4310].values[0][0]
print('original document: ')
words = []
for word in doc_sample.split(' '):
    words.append(word)
print(words)
print('\n\n tokenized and lemmatized document: ')
print(preprocess(doc_sample))
```

```
original document:
['people', 'pay', 'taxes', 'like', 'envision', 'good', 'building', 'roads', 'he
lping', 'poor', 'running', 'schools', 'etc', 'small', 'percentage', 'taxes', 'a
ctually', 'useful', 'things', 'rest', 'wasted', 'quote', 'libertarian']


 tokenized and lemmatized document:
['peopl', 'tax', 'like', 'envis', 'good', 'build', 'road', 'help', 'poor', 'ru
n', 'school', 'small', 'percentag', 'tax', 'actual', 'use', 'thing', 'rest', 'w
ast', 'quot', 'libertarian']
```

In [25]:
```
processed_docs = documents['tweet'].map(preprocess)
processed_docs[:10]
```

Out[25]:
```
0    [chronovari, texa, mass, shoot, survivor, lobb...
1    [time, hear, elit, wealthi, democrat, guncontr...
2    [olofsdotterk, royarahmani, nzambassadoru, mar...
3    [arizona, state, repres, longdon, gunviol, sur...
4    [kamalaharri, senatemajldr, senategop, stupid,...
5    [straight, heart, feel, gopcorrupt, gopcomplic...
6    [democrat, jump, board, guncontrol, surpris, s...
7    [gun_control_ca, doctor, speak, truth, line, c...
8    [dgolumbia, perfect, libertarian, internetfree...
9                                         [believ, guy]
Name: tweet, dtype: object
```

In [26]:
```
dictionary = gensim.corpora.Dictionary(processed_docs)
count = 0
for k, v in dictionary.iteritems():
    print(k, v)
    count += 1
    if count > 10:
        break
```

```
0 bullym
1 chronovari
2 congress
3 control
4 lobbi
5 mass
6 notonemor
7 shoot
8 survivor
9 texa
10 democrat
```

In [27]:
```
dictionary.filter_extremes(no_below=15, no_above=0.5, keep_n=100000)
```

In [28]:
```
bow_corpus = [dictionary.doc2bow(doc) for doc in processed_docs]
bow_corpus[4310]
```

Out[28]:
```
[(69, 1),
 (101, 1),
 (131, 1),
 (170, 1),
 (249, 1),
 (268, 1),
 (279, 1),
 (291, 1),
 (339, 1),
 (384, 2),
 (512, 1),
 (600, 1),
 (694, 1),
 (1048, 1),
 (1306, 1),
 (1320, 1),
 (1418, 1),
 (1459, 1)]
```

In [29]:
```
bow_doc_4310 = bow_corpus[4310]
for i in range(len(bow_doc_4310)):
    print("Word {} (\"{}\") appears {} time.".format(bow_doc_4310[i][0],
                                               dictionary[bow_doc_4310[i][0]],
bow_doc_4310[i][1]))
```

```
Word 69 ("libertarian") appears 1 time.
Word 101 ("peopl") appears 1 time.
Word 131 ("poor") appears 1 time.
Word 170 ("actual") appears 1 time.
Word 249 ("small") appears 1 time.
Word 268 ("build") appears 1 time.
Word 279 ("thing") appears 1 time.
Word 291 ("like") appears 1 time.
Word 339 ("school") appears 1 time.
Word 384 ("tax") appears 2 time.
Word 512 ("good") appears 1 time.
Word 600 ("run") appears 1 time.
Word 694 ("help") appears 1 time.
Word 1048 ("use") appears 1 time.
Word 1306 ("quot") appears 1 time.
Word 1320 ("rest") appears 1 time.
Word 1418 ("wast") appears 1 time.
Word 1459 ("road") appears 1 time.
```

In [30]:

```
from gensim import corpora, models
tfidf = models.TfidfModel(bow_corpus)
corpus_tfidf = tfidf[bow_corpus]
from pprint import pprint
for doc in corpus_tfidf:
    pprint(doc)
    break
```

```
[(0, 0.3239431959646286),
 (1, 0.26882972251589077),
 (2, 0.4298153999078772),
 (3, 0.28896727762614743),
 (4, 0.49870097875946895),
 (5, 0.21411552326666164),
 (6, 0.3744363154050506),
 (7, 0.3461175273373146)]
```

In [31]:

```
lda_model = gensim.models.LdaMulticore(bow_corpus, num_topics=10, id2word=diction
```

In [32]:
```
for idx, topic in lda_model.print_topics(-1):
    print('Topic: {} \nWords: {}'.format(idx, topic))
```

```
Topic: 0 
Words: 0.031*"guncontrol" + 0.027*"gunviol" + 0.025*"gun" + 0.018*"shoot" + 0.0
16*"beto" + 0.016*"betoorourk" + 0.014*"guncontrolnow" + 0.013*"like" + 0.013
*"need" + 0.012*"ndamend"
Topic: 1 
Words: 0.034*"guncontrol" + 0.020*"gunviol" + 0.012*"shoot" + 0.012*"resist" +
0.011*"libertarian" + 0.009*"go" + 0.009*"american" + 0.009*"kid" + 0.008*"hand
gun" + 0.008*"impeachtrump"
Topic: 2 
Words: 0.030*"libertarian" + 0.026*"guncontrol" + 0.023*"maga" + 0.018*"democra
t" + 0.018*"news" + 0.016*"homeless" + 0.015*"trump" + 0.014*"conserv" + 0.014
*"good" + 0.014*"hous"
Topic: 3 
Words: 0.023*"gunviol" + 0.021*"guncontrol" + 0.019*"prolif" + 0.011*"beto" +
0.010*"democrat" + 0.009*"abort" + 0.009*"children" + 0.009*"bear" + 0.009*"ali
v" + 0.008*"support"
Topic: 4 
Words: 0.055*"prolif" + 0.042*"life" + 0.030*"abort" + 0.028*"thank" + 0.027*"s
tand" + 0.021*"american" + 0.019*"leadership" + 0.018*"effort" + 0.018*"need" +
0.018*"right"
Topic: 5 
Words: 0.030*"gunviol" + 0.023*"prolif" + 0.020*"libertarian" + 0.012*"news" +
0.011*"liberti" + 0.009*"pundit" + 0.009*"hear" + 0.008*"gateway" + 0.008*"mag
a" + 0.008*"support"
Topic: 6 
Words: 0.049*"guncontrol" + 0.019*"want" + 0.017*"libertarian" + 0.012*"like" +
0.011*"betoorourk" + 0.009*"gun" + 0.009*"care" + 0.009*"go" + 0.009*"gunviol"
+ 0.008*"liberti"
Topic: 7 
Words: 0.038*"colt" + 0.024*"rifl" + 0.024*"gunviol" + 0.022*"stop" + 0.021*"pr
oduct" + 0.019*"civilian" + 0.014*"market" + 0.013*"guncontrol" + 0.013*"violen
c" + 0.011*"suspend"
Topic: 8 
Words: 0.050*"trump" + 0.034*"maga" + 0.027*"right" + 0.025*"guncontrol" + 0.02
2*"realdonaldtrump" + 0.020*"prolif" + 0.013*"adefend" + 0.012*"democrat" + 0.0
10*"ndamend" + 0.009*"gun"
Topic: 9 
Words: 0.075*"guncontrol" + 0.023*"gun" + 0.012*"peopl" + 0.011*"control" + 0.0
10*"gunsens" + 0.010*"democrat" + 0.008*"check" + 0.008*"know" + 0.008*"americ
a" + 0.007*"guncontrolnow"
```

In [33]:
```
lda_model_tfidf = gensim.models.LdaMulticore(corpus_tfidf, num_topics=10, id2word
for idx, topic in lda_model_tfidf.print_topics(-1):
    print('Topic: {} Word: {}'.format(idx, topic))
```

```
Topic: 0 Word: 0.014*"libertarian" + 0.010*"gunviol" + 0.010*"conserv" + 0.010
*"democrat" + 0.010*"guncontrol" + 0.009*"meme" + 0.009*"trump" + 0.008*"maga"
+ 0.007*"great" + 0.007*"protect"
Topic: 1 Word: 0.010*"gunviol" + 0.010*"guncontrol" + 0.008*"prolif" + 0.008*"l
ibertarian" + 0.007*"peopl" + 0.007*"beto" + 0.007*"like" + 0.007*"gun" + 0.006
*"good" + 0.005*"trump"
Topic: 2 Word: 0.016*"guncontrol" + 0.010*"prolif" + 0.008*"believ" + 0.008*"de
mocrat" + 0.008*"guy" + 0.007*"maga" + 0.007*"trump" + 0.006*"republican" + 0.0
06*"need" + 0.005*"realdonaldtrump"
Topic: 3 Word: 0.009*"guncontrol" + 0.009*"gun" + 0.008*"rifl" + 0.008*"colt" +
0.007*"product" + 0.007*"gunviol" + 0.006*"prolif" + 0.006*"shoot" + 0.006*"civ
ilian" + 0.005*"trump"
Topic: 4 Word: 0.011*"guncontrol" + 0.009*"peopl" + 0.009*"shoot" + 0.009*"ndam
end" + 0.007*"adefend" + 0.007*"betoorourk" + 0.007*"bring" + 0.007*"gun" + 0.0
07*"prolif" + 0.006*"kill"
Topic: 5 Word: 0.011*"guncontrol" + 0.010*"gunviol" + 0.008*"gun" + 0.007*"libe
rtarian" + 0.006*"support" + 0.005*"colt" + 0.005*"shoot" + 0.005*"prolif" + 0.
005*"work" + 0.005*"abort"
Topic: 6 Word: 0.013*"thank" + 0.012*"life" + 0.012*"american" + 0.012*"secaza
r" + 0.012*"secpompeo" + 0.012*"effort" + 0.012*"leadership" + 0.012*"stand" +
0.011*"guncontrol" + 0.011*"prolif"
Topic: 7 Word: 0.010*"guncontrol" + 0.007*"prolif" + 0.007*"gunviol" + 0.007*"b
etoorourk" + 0.006*"peopl" + 0.006*"like" + 0.006*"weapon" + 0.006*"need" + 0.0
05*"abort" + 0.005*"plan"
Topic: 8 Word: 0.009*"guncontrol" + 0.008*"gunviol" + 0.008*"prolif" + 0.007*"l
ibertarian" + 0.007*"tlot" + 0.007*"maga" + 0.006*"tcot" + 0.006*"talk" + 0.006
*"freedom" + 0.005*"right"
Topic: 9 Word: 0.008*"guncontrol" + 0.008*"prolif" + 0.007*"think" + 0.007*"tru
mp" + 0.007*"libertarian" + 0.007*"gunviol" + 0.006*"maga" + 0.006*"gun" + 0.00
6*"say" + 0.006*"news"
```

In [34]:
```
processed_docs[4310]
```

Out[34]:
```
['peopl',
 'tax',
 'like',
 'envis',
 'good',
 'build',
 'road',
 'help',
 'poor',
 'run',
 'school',
 'small',
 'percentag',
 'tax',
 'actual',
 'use',
 'thing',
 'rest',
 'wast',
 'quot',
 'libertarian']
```

In [35]:
```
for index, score in sorted(lda_model[bow_corpus[4310]], key=lambda tup: -1*tup[1
    print("\nScore: {}\t \nTopic: {}".format(score, lda_model.print_topic(index,
```

```

Score: 0.6339181065559387
Topic: 0.031*"guncontrol" + 0.027*"gunviol" + 0.025*"gun" + 0.018*"shoot" + 0.0
16*"beto" + 0.016*"betoorourk" + 0.014*"guncontrolnow" + 0.013*"like" + 0.013
*"need" + 0.012*"ndamend"

Score: 0.3260651230812073
Topic: 0.030*"gunviol" + 0.023*"prolif" + 0.020*"libertarian" + 0.012*"news" +
0.011*"liberti" + 0.009*"pundit" + 0.009*"hear" + 0.008*"gateway" + 0.008*"mag
a" + 0.008*"support"
```

In [36]:
```
for index, score in sorted(lda_model_tfidf[bow_corpus[4310]], key=lambda tup: -1
    print("\nScore: {}\t \nTopic: {}".format(score, lda_model_tfidf.print_topic(
```

```

Score: 0.8350743055343628
Topic: 0.011*"guncontrol" + 0.009*"peopl" + 0.009*"shoot" + 0.009*"ndamend" +
0.007*"adefend" + 0.007*"betoorourk" + 0.007*"bring" + 0.007*"gun" + 0.007*"pro
lif" + 0.006*"kill"

Score: 0.12491016089916229
Topic: 0.014*"libertarian" + 0.010*"gunviol" + 0.010*"conserv" + 0.010*"democra
t" + 0.010*"guncontrol" + 0.009*"meme" + 0.009*"trump" + 0.008*"maga" + 0.007
*"great" + 0.007*"protect"
```

In [37]:
```
## Visualizations
```

In [38]:
```
df1 = pd.read_csv('f_a3.csv')
df1.columns=['index','date','tweet','countnoun','countverb','countadj','countadp'
df1
```

Out[38]:

	index	date	tweet	countnoun	countverb	countadj	countadp	cou
0	3	2019-09-27	time hear elite wealthy democrat guncontrol re...	11	3	4	0	
1	4	2019-09-24	olofsdotterk royarahmani nzambassadorus mars...	11	4	3	0	
2	5	2019-09-25	arizona state representative jen longdon gunvi...	16	2	2	0	
3	6	2019-09-20	kamalaharris lot senatemajldr senategop stup...	7	3	3	0	
4	7	2019-09-26	ugh straight heart gopcomplicittraitors feels ...	10	2	2	0	
5	8	2019-09-19	democrats jumping board guncontrol surprising ...	11	3	1	0	
6	9	2019-09-27	rt gun_control_ca doctors speak truth lines co	8	1	2	0	

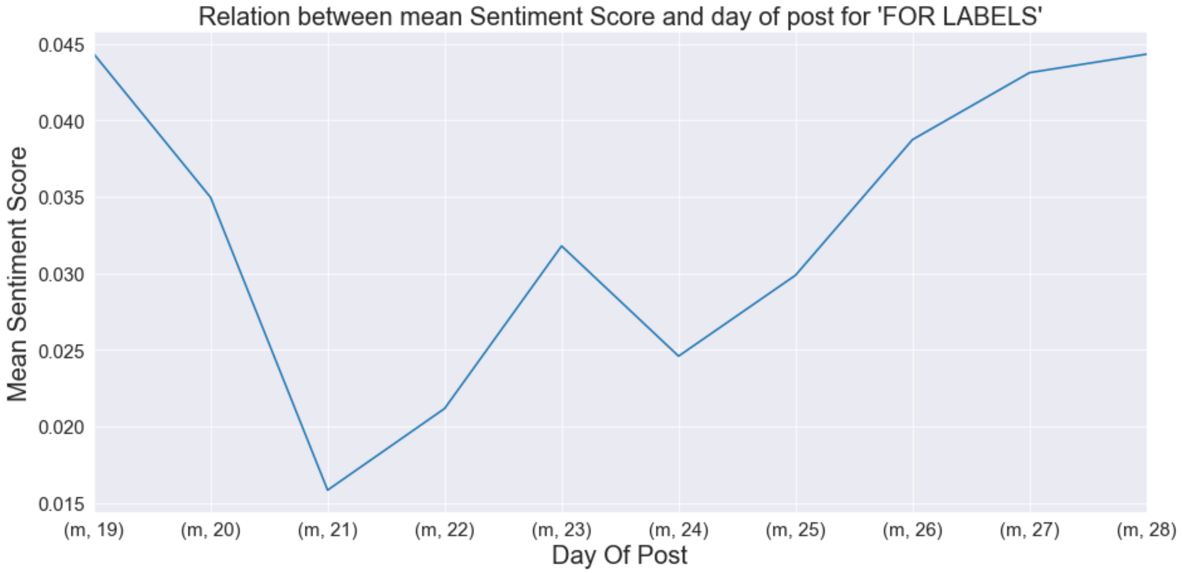
In [99]:

```
df_for=df1[df1['target']==1]
df_for['day']=df['date'].apply(lambda x :x[8:10])
fig, ax = plt.subplots(figsize=(15,7))

#temp_min = df_for.groupby(['date'])['countnoun','countverb','countadj','countadp
temp_min1 = df_for.groupby(['day'])['sentiment_score'].agg({'m': np.mean}).unstac
ax.set_xlabel('Day Of Post',fontsize=20)
ax.set_ylabel('Mean Sentiment Score',fontsize=20)
ax.set_title("Relation between mean Sentiment Score and day of post for 'FOR LABE

ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
'''temp_min2 = df_for.groupby(['day'])['countnoun'].agg({'m': np.mean}).unstack()
temp_min3 = df_for.groupby(['day'])['countverb'].agg({'m': np.mean}).unstack().pl
temp_min4 = df_for.groupby(['day'])['countadj'].agg({'m': np.mean}).unstack().plo
```

```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy (http://pandas.pydat
a.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cop
y)

c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:6: FutureWarning: using a dict on a Series for aggregation
is deprecated and will be removed in a future version. Use                 name
d aggregation instead.

    >>> grouper.agg(name_1=func_1, name_2=func_2)
```

Out[99]:

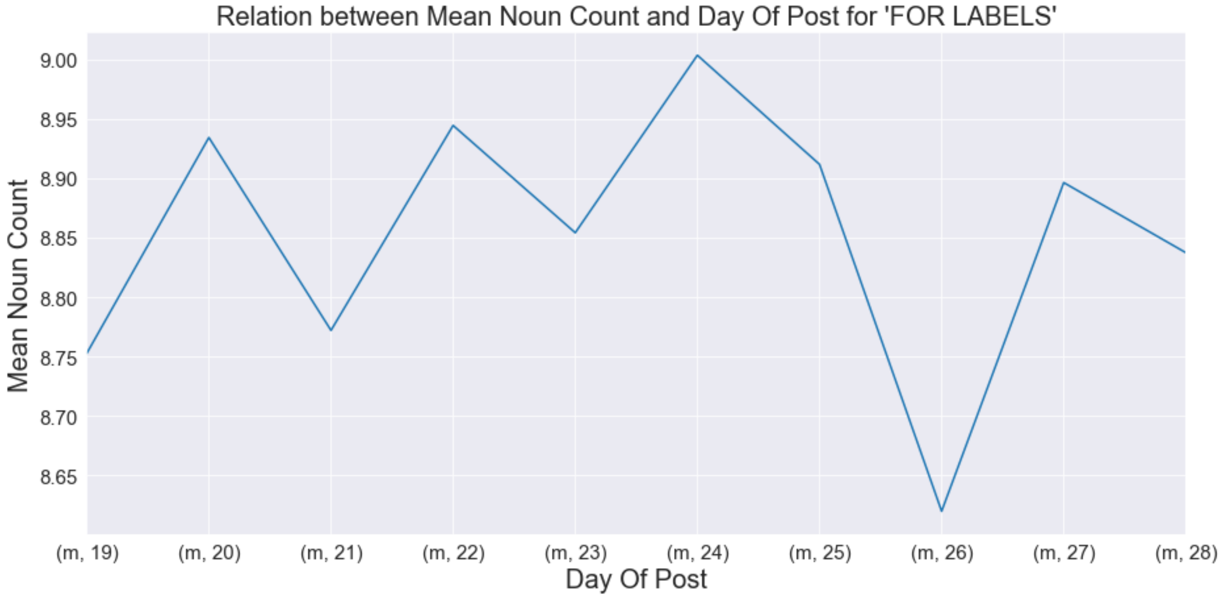
```
"temp_min2 = df_for.groupby(['day'])['countnoun'].agg({'m': np.mean}).unstack
().plot(ax=ax)\ntemp_min3 = df_for.groupby(['day'])['countverb'].agg({'m': np.m
ean}).unstack().plot(ax=ax)\ntemp_min4 = df_for.groupby(['day'])['countadj'].ag
g({'m': np.mean}).unstack().plot(ax=ax)"
```


Relation between mean Sentiment Score and day of post for 'FOR LABELS'
Mean Sentiment Score
0.045
0.040
0.035
0.030
0.025
0.020
0.015
(m, 19)
(m, 20)
(m, 21)
(m, 22)
(m, 23)
(m, 24)
(m, 25)
(m, 26)
(m, 27)
(m, 28)
Day Of Post

In [107]:

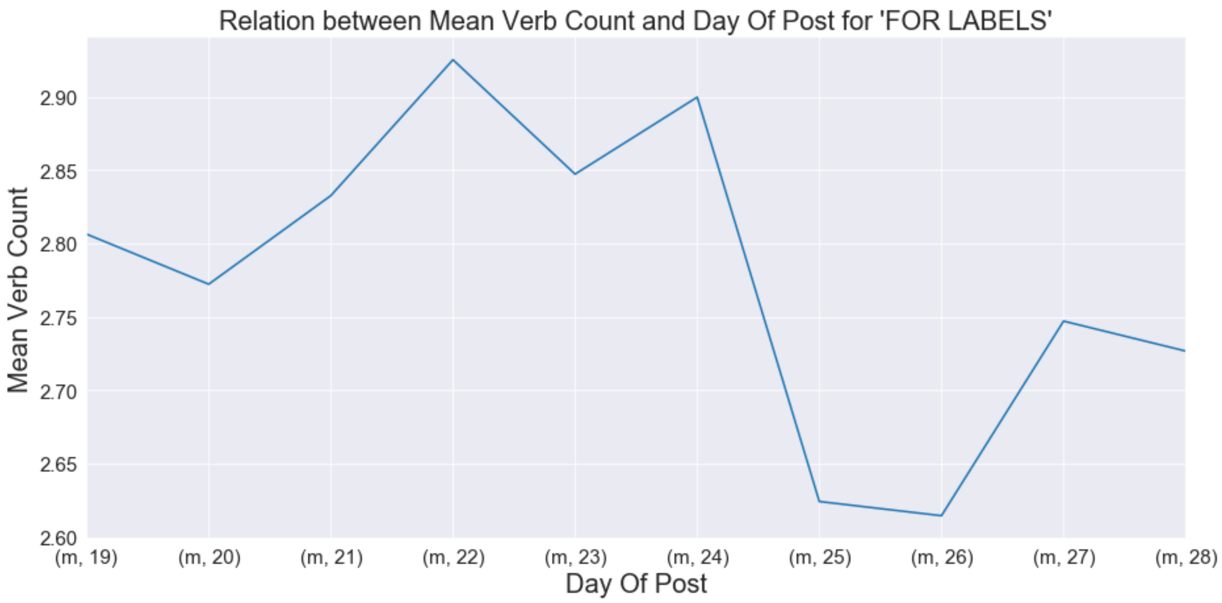
```
df_for=df1[df1['target']==1]
df_for['day']=df['date'].apply(lambda x :x[8:10])
fig, ax = plt.subplots(figsize=(15,7))

#temp_min = df_for.groupby(['date'])['countnoun','countverb','countadj','countadp
temp_min2 = df_for.groupby(['day'])['countnoun'].agg({'m': np.mean}).unstack().pl
ax.set_xlabel('Day Of Post',fontsize=20)
ax.set_ylabel('Mean Noun Count',fontsize=20)
ax.set_title("Relation between Mean Noun Count and Day Of Post for 'FOR LABELS'",
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```

```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy (http://pandas.pydat
a.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cop
y)

c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:6: FutureWarning: using a dict on a Series for aggregation
is deprecated and will be removed in a future version. Use                 name
d aggregation instead.

    >>> grouper.agg(name_1=func_1, name_2=func_2)
```



In [108]:
```
df_for=df1[df1['target']==1]
df_for['day']=df['date'].apply(lambda x :x[8:10])
fig, ax = plt.subplots(figsize=(15,7))

#temp_min = df_for.groupby(['date'])['countnoun','countverb','countadj','countadp
temp_min3 = df_for.groupby(['day'])['countverb'].agg({'m': np.mean}).unstack().pl
ax.set_xlabel('Day Of Post',fontsize=20)
ax.set_ylabel('Mean Verb Count',fontsize=20)
ax.set_title("Relation between Mean Verb Count and Day Of Post for 'FOR LABELS'",
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```

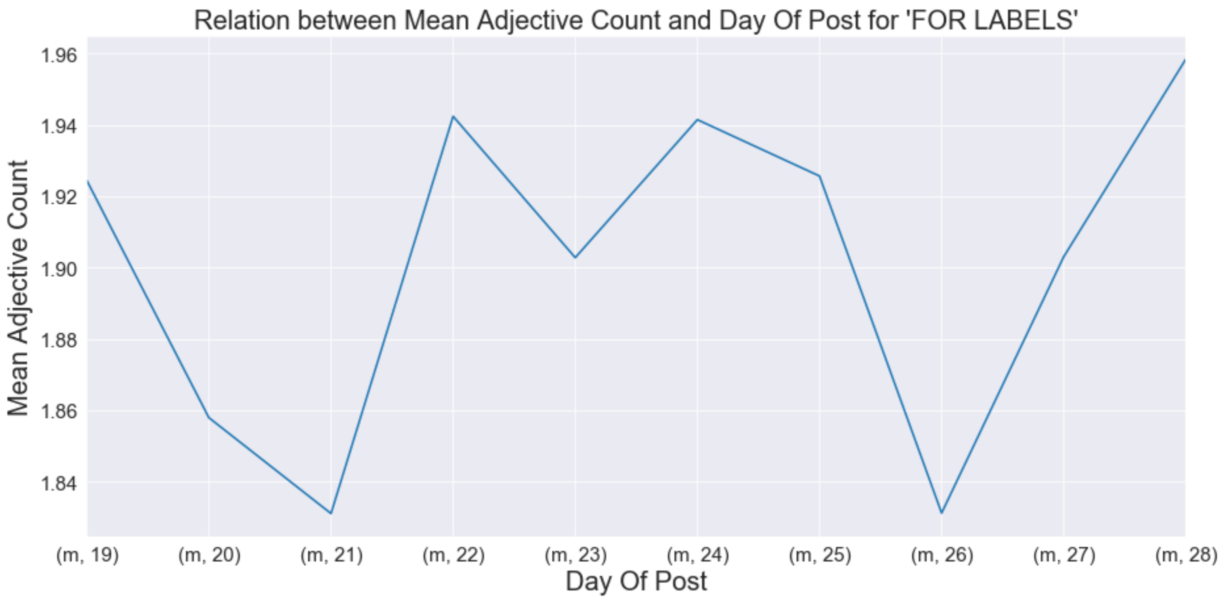
```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy (http://pandas.pydat
a.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cop
y)

c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:6: FutureWarning: using a dict on a Series for aggregation
is deprecated and will be removed in a future version. Use                 name
d aggregation instead.

    >>> grouper.agg(name_1=func_1, name_2=func_2)
```



In [109]:

```
df_for=df1[df1['target']==1]
df_for['day']=df['date'].apply(lambda x :x[8:10])
fig, ax = plt.subplots(figsize=(15,7))

#temp_min = df_for.groupby(['date'])['countnoun','countverb','countadj','countadp
temp_min3 = df_for.groupby(['day'])['countadj'].agg({'m': np.mean}).unstack().plo
ax.set_xlabel('Day Of Post',fontsize=20)
ax.set_ylabel('Mean Adjective Count',fontsize=20)
ax.set_title("Relation between Mean Adjective Count and Day Of Post for 'FOR LABE
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```

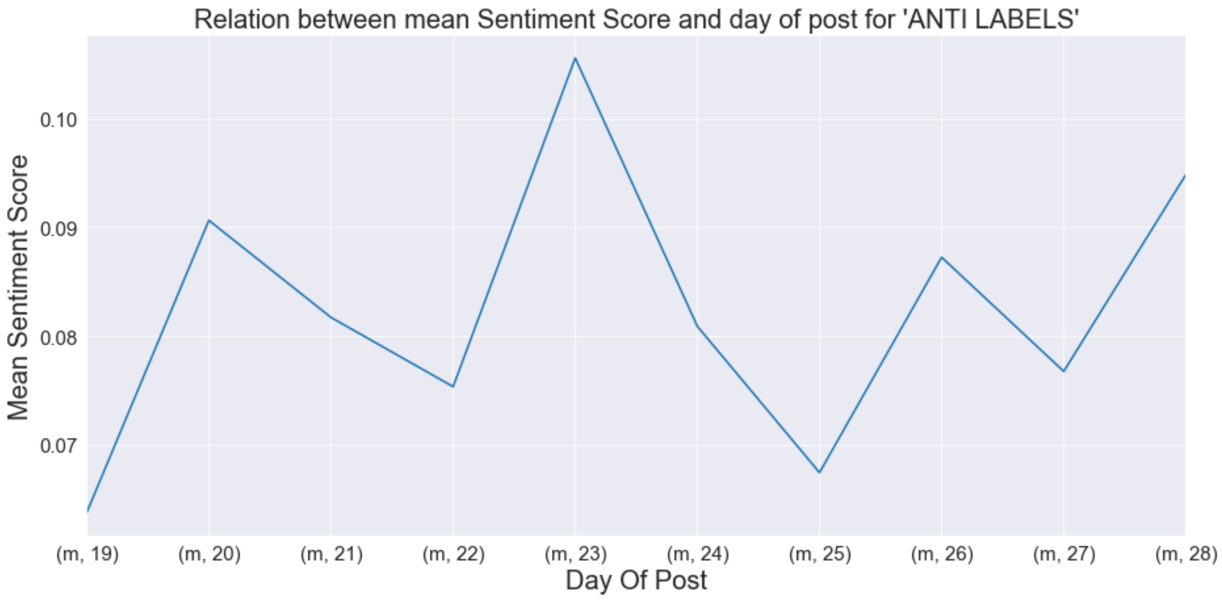
```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy (http://pandas.pydat
a.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cop
y)

c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:6: FutureWarning: using a dict on a Series for aggregation
is deprecated and will be removed in a future version. Use                 name
d aggregation instead.

    >>> grouper.agg(name_1=func_1, name_2=func_2)
```



In [110]:

```
df_against=df1[df1['target']==0]
df_against['day']=df['date'].apply(lambda x :x[8:10])
fig, ax = plt.subplots(figsize=(15,7))

#temp_min = df_for.groupby(['date'])['countnoun','countverb','countadj','countadp
temp_min1 = df_against.groupby(['day'])['sentiment_score'].agg({'m': np.mean}).ur
ax.set_xlabel('Day Of Post',fontsize=20)
ax.set_ylabel('Mean Sentiment Score',fontsize=20)
ax.set_title("Relation between mean Sentiment Score and day of post for 'ANTI LAE
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```

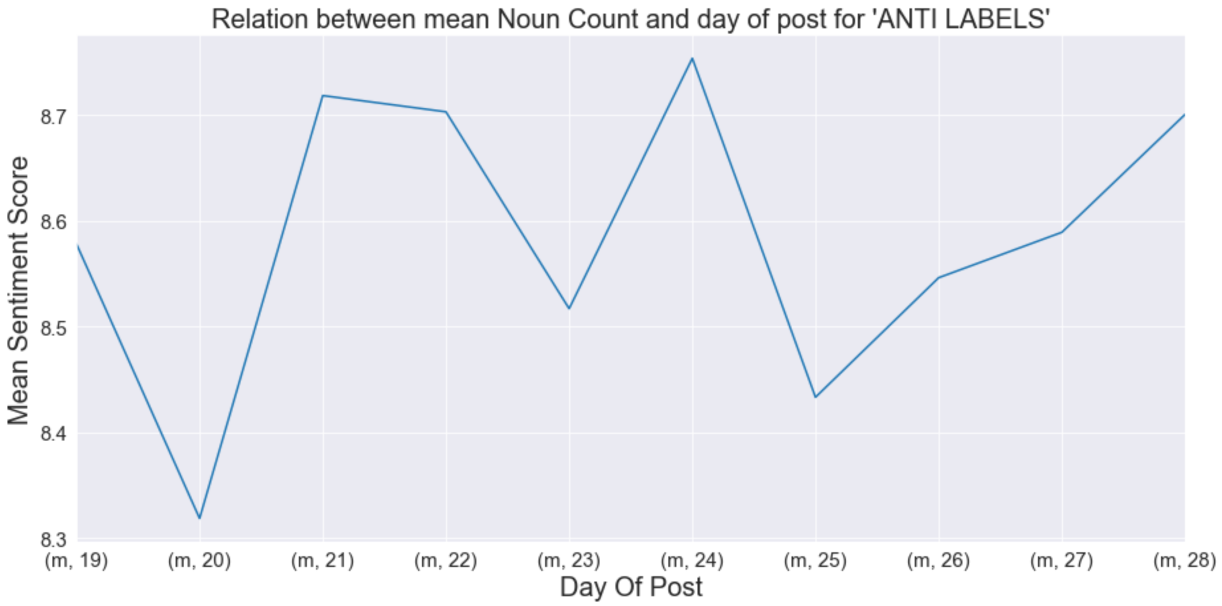
```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy (http://pandas.pydat
a.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cop
y)

c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:6: FutureWarning: using a dict on a Series for aggregation
is deprecated and will be removed in a future version. Use                 name
d aggregation instead.

    >>> grouper.agg(name_1=func_1, name_2=func_2)
```



In [111]:

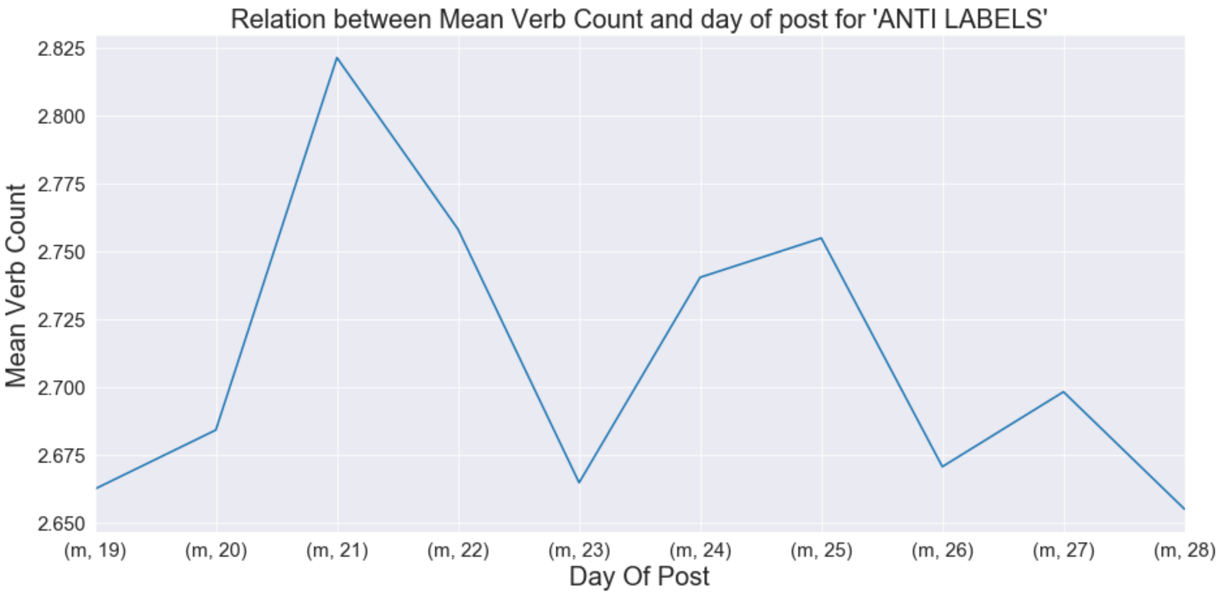
```
fig, ax = plt.subplots(figsize=(15,7))

#temp_min = df_for.groupby(['date'])['countnoun','countverb','countadj','countadp
temp_min1 = df_against.groupby(['day'])['countnoun'].agg({'m': np.mean}).unstack(
ax.set_xlabel('Day Of Post',fontsize=20)
ax.set_ylabel('Mean Sentiment Score',fontsize=20)
ax.set_title("Relation between mean Noun Count and day of post for 'ANTI LABELS''
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```

```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:4: FutureWarning: using a dict on a Series for aggregation
is deprecated and will be removed in a future version. Use                 name
d aggregation instead.

    >>> grouper.agg(name_1=func_1, name_2=func_2)

  after removing the cwd from sys.path.
```



In [112]:
```
fig, ax = plt.subplots(figsize=(15,7))

#temp_min = df_for.groupby(['date'])['countnoun','countverb','countadj','countadp
temp_min1 = df_against.groupby(['day'])['countverb'].agg({'m': np.mean}).unstack(
ax.set_xlabel('Day Of Post',fontsize=20)
ax.set_ylabel('Mean Verb Count',fontsize=20)
ax.set_title("Relation between Mean Verb Count and day of post for 'ANTI LABELS''
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```

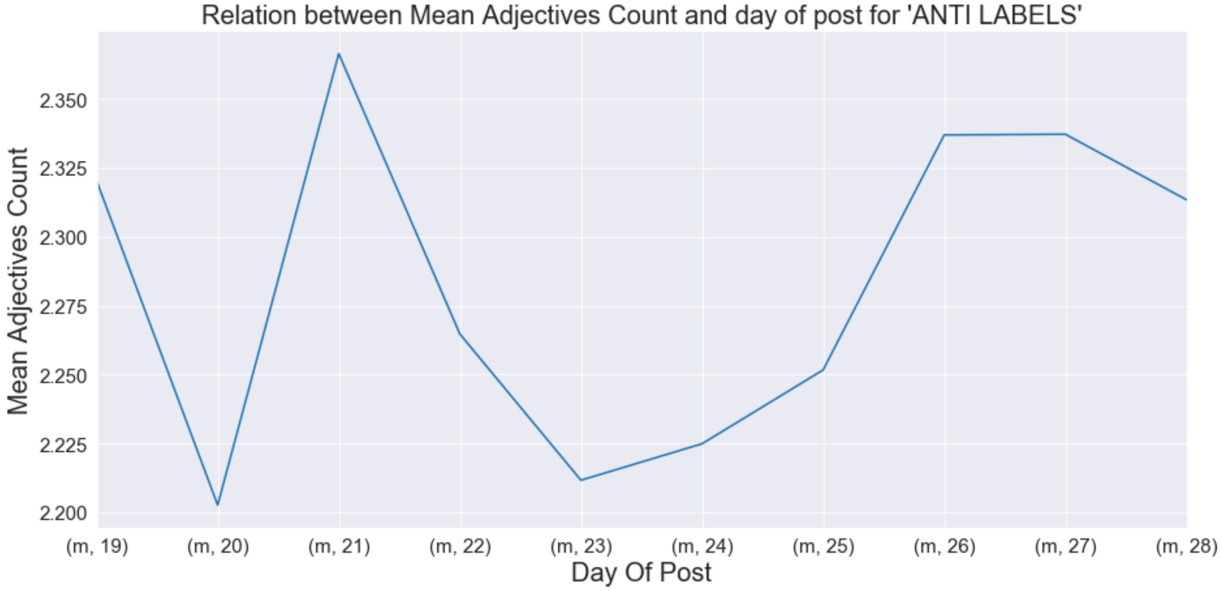
```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:4: FutureWarning: using a dict on a Series for aggregation
is deprecated and will be removed in a future version. Use                name
d aggregation instead.

    >>> grouper.agg(name_1=func_1, name_2=func_2)

  after removing the cwd from sys.path.
```



In [113]:

```
fig, ax = plt.subplots(figsize=(15,7))

#temp_min = df_for.groupby(['date'])['countnoun','countverb','countadj','countadp
temp_min1 = df_against.groupby(['day'])['countadj'].agg({'m': np.mean}).unstack(
ax.set_xlabel('Day Of Post',fontsize=20)
ax.set_ylabel('Mean Adjectives Count',fontsize=20)
ax.set_title("Relation between Mean Adjectives Count and day of post for 'ANTI LA
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```

```
c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\ipykerne
l_launcher.py:4: FutureWarning: using a dict on a Series for aggregation
is deprecated and will be removed in a future version. Use                 name
d aggregation instead.

    >>> grouper.agg(name_1=func_1, name_2=func_2)

  after removing the cwd from sys.path.
```



In [114]:
```
temp_min1 = df_for.groupby(['day'])['sentiment_score','countnoun','countverb','co
print (temp_min1)
```

```
                        day
m  sentiment_score  19      0.044357
                    20      0.034961
                    21      0.015826
                    22      0.021159
                    23      0.031787
                    24      0.024586
                    25      0.029875
                    26      0.038743
                    27      0.043119
                    28      0.044327
   countnoun        19      8.752747
                    20      8.934169
                    21      8.772093
                    22      8.944338
                    23      8.854167
                    24      9.003344
                    25      8.911641
                    26      8.620075
                    27      8.896290
                    28      8.837370
   countverb        19      2.806319
                    20      2.772205
                    21      2.832558
                    22      2.925144
                    23      2.847222
                    24      2.899666
                    25      2.624123
                    26      2.614447
                    27      2.747049
                    28      2.726644
   countadj         19      1.924451
                    20      1.857889
                    21      1.831008
                    22      1.942418
                    23      1.902778
                    24      1.941472
                    25      1.925666
                    26      1.831144
                    27      1.903035
                    28      1.958478
dtype: float64

c:\users\dell\appdata\local\programs\python\python37\lib\site-packages\pandas\c
ore\groupby\generic.py:1455: FutureWarning: using a dict with renaming is depre
cated and will be removed
in a future version.

For column-specific groupby renaming, use named aggregation

    >>> df.groupby(...).agg(name=('column', aggfunc))

  return super().aggregate(arg, *args, **kwargs)
```

In [1]:

```
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.metrics import classification_report

df = pd.read_csv('f_a2.csv')
df.columns=['index','date','tweet','target']
df
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-a260a799431c> in <module>
      4 from sklearn.metrics import classification_report
      5 
----> 6 df = pd.read_csv('f_a2.csv')
      7 df.columns=['index','date','tweet','target']
      8 df

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

In [13]:

```
flairs=['for','against']
cat = df.target

V = df.tweet

X_train, X_test, y_train, y_test = train_test_split( V, cat, test_size=0.3, rando
print("Results of Random Forest")
from sklearn.ensemble import RandomForestClassifier

ranfor = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('clf', RandomForestClassifier(n_estimators = 1000, random_stat
                  ])

ranfor.fit(X_train, y_train)
y_pred = ranfor.predict(X_test)

print('accuracy %s' % accuracy_score(y_pred, y_test))
print(classification_report(y_test, y_pred,target_names=flairs))
```

```
Results of Random Forest
accuracy 0.9527405602923265
              precision    recall  f1-score   support

         for       0.96      0.93      0.95      1935
     against       0.94      0.97      0.96      2170

    accuracy                           0.95      4105
   macro avg       0.95      0.95      0.95      4105
weighted avg       0.95      0.95      0.95      4105
```

In [13]:

```
acc_test=[]
for i in range(9):
    X_train, X_test, y_train, y_test = train_test_split( V, cat, test_size=(i+1)/

    ranfor = Pipeline([('vect', CountVectorizer()),
                      ('tfidf', TfidfTransformer()),
                      ('clf', RandomForestClassifier(n_estimators = 100, random_s
                      ])

    ranfor.fit(X_train, y_train)
    y_pred = ranfor.predict(X_test)
    acc_test.append(accuracy_score(y_pred, y_test))
    print('accuracy %s' % accuracy_score(y_pred, y_test))
print (acc_test)
```

```
accuracy 0.95836376917458
accuracy 0.9514066496163683
accuracy 0.9517661388550548
accuracy 0.9499451954694922
accuracy 0.9516223326512716
accuracy 0.9448233861144946
accuracy 0.9438354734314647
accuracy 0.9413537955604275
accuracy 0.928136419001218
[0.95836376917458, 0.9514066496163683, 0.9517661388550548, 0.9499451954694922,
0.9516223326512716, 0.9448233861144946, 0.9438354734314647, 0.9413537955604275,
0.928136419001218]
```

In [14]:

```
acc_train=[]
for i in range(9):
    X_train, X_test, y_train, y_test = train_test_split( V, cat, train_size=(i+1)

    ranfor = Pipeline([('vect', CountVectorizer()),
                      ('tfidf', TfidfTransformer()),
                      ('clf', RandomForestClassifier(n_estimators = 100, random_s
                      ])

    ranfor.fit(X_train, y_train)
    y_pred = ranfor.predict(X_test)
    acc_train.append(accuracy_score(y_pred, y_test))
    print('accuracy %s' % accuracy_score(y_pred, y_test))
print (acc_train)
```

```
accuracy 0.928136419001218
accuracy 0.9413537955604275
accuracy 0.9438354734314647
accuracy 0.9448233861144946
accuracy 0.9516223326512716
accuracy 0.9499451954694922
accuracy 0.9517661388550548
accuracy 0.9514066496163683
accuracy 0.95836376917458
[0.928136419001218, 0.9413537955604275, 0.9438354734314647, 0.9448233861144946,
0.9516223326512716, 0.9499451954694922, 0.9517661388550548, 0.9514066496163683,
0.95836376917458]
```

In [15]:

```
acc_log_test=[]
for i in range(9):
    X_train, X_test, y_train, y_test = train_test_split( V, cat, test_size=(i+1)/

    ranfor = Pipeline([('vect', CountVectorizer()),
                    ('tfidf', TfidfTransformer()),
                    ('clf', LogisticRegression(solver='lbfgs', multi_class='aut
                    ])

    ranfor.fit(X_train, y_train)
    y_pred = ranfor.predict(X_test)
    acc_log_test.append(accuracy_score(y_pred, y_test))
    print('accuracy %s' % accuracy_score(y_pred, y_test))
print (acc_log_test)
```

```
accuracy 0.9620160701241782
accuracy 0.9612714651077823
accuracy 0.9624847746650427
accuracy 0.9610887833394227
accuracy 0.9590762934814382
accuracy 0.9548112058465287
accuracy 0.9530222361415597
accuracy 0.9503060199141318
accuracy 0.9386926512383272
[0.9620160701241782, 0.9612714651077823, 0.9624847746650427, 0.961088783339422
7, 0.9590762934814382, 0.9548112058465287, 0.9530222361415597, 0.95030601991413
18, 0.9386926512383272]
```

In [16]:

```
acc_svm_test=[]
for i in range(9):
    X_train, X_test, y_train, y_test = train_test_split( V, cat, test_size=(i+1)/

    ranfor = Pipeline([('vect', CountVectorizer()),
                    ('tfidf', TfidfTransformer()),
                    ('clf', svm.SVC(kernel='linear')),
                    ])

    ranfor.fit(X_train, y_train)
    y_pred = ranfor.predict(X_test)
    acc_svm_test.append(accuracy_score(y_pred, y_test))
    print('accuracy %s' % accuracy_score(y_pred, y_test))
print (acc_svm_test)
```

```
accuracy 0.9656683710737765
accuracy 0.9663865546218487
accuracy 0.964190012180268
accuracy 0.964011691633175
accuracy 0.9614147909967846
accuracy 0.9577344701583435
accuracy 0.9539617914187285
accuracy 0.9503973691422307
accuracy 0.9404790905399919
[0.9656683710737765, 0.9663865546218487, 0.964190012180268, 0.964011691633175,
0.9614147909967846, 0.9577344701583435, 0.9539617914187285, 0.9503973691422307,
0.9404790905399919]
```

In [17]:

```
acc_log_train=[]
for i in range(9):
    X_train, X_test, y_train, y_test = train_test_split( V, cat, train_size=(i+1)

    ranfor = Pipeline([('vect', CountVectorizer()),
                       ('tfidf', TfidfTransformer()),
                       ('clf', LogisticRegression(solver='lbfgs', multi_class='aut
                       ])

    ranfor.fit(X_train, y_train)
    y_pred = ranfor.predict(X_test)
    acc_log_train.append(accuracy_score(y_pred, y_test))
    print('accuracy %s' % accuracy_score(y_pred, y_test))
print (acc_log_train)
```

```
accuracy 0.9386926512383272
accuracy 0.9503060199141318
accuracy 0.9530222361415597
accuracy 0.9548112058465287
accuracy 0.9590762934814382
accuracy 0.9610887833394227
accuracy 0.9624847746650427
accuracy 0.9612714651077823
accuracy 0.9620160701241782
[0.9386926512383272, 0.9503060199141318, 0.9530222361415597, 0.954811205846528
7, 0.9590762934814382, 0.9610887833394227, 0.9624847746650427, 0.96127146510778
23, 0.9620160701241782]
```

In [18]:
```
cat = df.target

V = df.tweet

acc_svm_train=[]
for i in range(9):
    X_train, X_test, y_train, y_test = train_test_split( V, cat, train_size=(i+1

    svtrain = Pipeline([('vect', CountVectorizer()),
                      ('tfidf', TfidfTransformer()),
                      ('clf', svm.SVC(kernel='linear')),
                      ])

    svtrain.fit(X_train, y_train)
    y_pred = svtrain.predict(X_test)
    acc_svm_train.append(accuracy_score(y_pred, y_test))
    print('accuracy %s' % accuracy_score(y_pred, y_test))
print (acc_svm_train)
```

```
accuracy 0.9404790905399919
accuracy 0.9503973691422307
accuracy 0.9539617914187285
accuracy 0.9577344701583435
accuracy 0.9614147909967846
accuracy 0.964011691633175
accuracy 0.964190012180268
accuracy 0.9663865546218487
accuracy 0.9656683710737765
[0.9404790905399919, 0.9503973691422307, 0.9539617914187285, 0.957734470158343
5, 0.9614147909967846, 0.964011691633175, 0.964190012180268, 0.966386554621848
7, 0.9656683710737765]
```

In [30]:

```
acc_k_train=[]
for i in range(9):
    X_train, X_test, y_train, y_test = train_test_split( V, cat, train_size=(i+1)

    svtrain = Pipeline([('vect', CountVectorizer()),
                      ('tfidf', TfidfTransformer()),
                      ('clf', KNeighborsClassifier(n_neighbors=3)),
                      ])

    svtrain.fit(X_train, y_train)
    y_pred = svtrain.predict(X_test)
    acc_k_train.append(accuracy_score(y_pred, y_test))
    print('accuracy %s' % accuracy_score(y_pred, y_test))
print (acc_k_train)
```

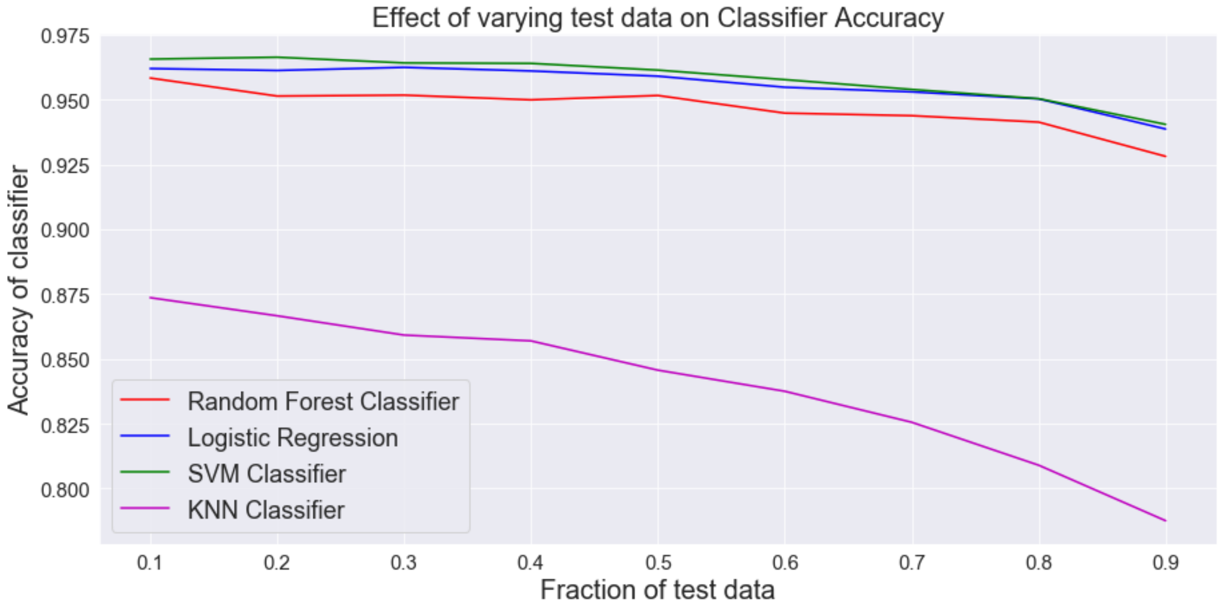
```
accuracy 0.7875761266747868
accuracy 0.8089887640449438
accuracy 0.8255559035389916
accuracy 0.8375152253349574
accuracy 0.8456591639871383
accuracy 0.8569601753744976
accuracy 0.8591961023142509
accuracy 0.866642309097552
accuracy 0.8736303871439006
[0.7875761266747868, 0.8089887640449438, 0.8255559035389916, 0.837515225334957
4, 0.8456591639871383, 0.8569601753744976, 0.8591961023142509, 0.86664230909755
2, 0.8736303871439006]
```

In [31]:

```
acc_k_test=[]
for i in range(9):
    X_train, X_test, y_train, y_test = train_test_split( V, cat, test_size=(i+1)

    ranfor = Pipeline([('vect', CountVectorizer()),
                     ('tfidf', TfidfTransformer()),
                     ('clf', KNeighborsClassifier(n_neighbors=3)),
                     ])
r
    ranfor.fit(X_train, y_train)
    y_pred = ranfor.predict(X_test)
    acc_k_test.append(accuracy_score(y_pred, y_test))
    print('accuracy %s' % accuracy_score(y_pred, y_test))
print (acc_k_test)
```

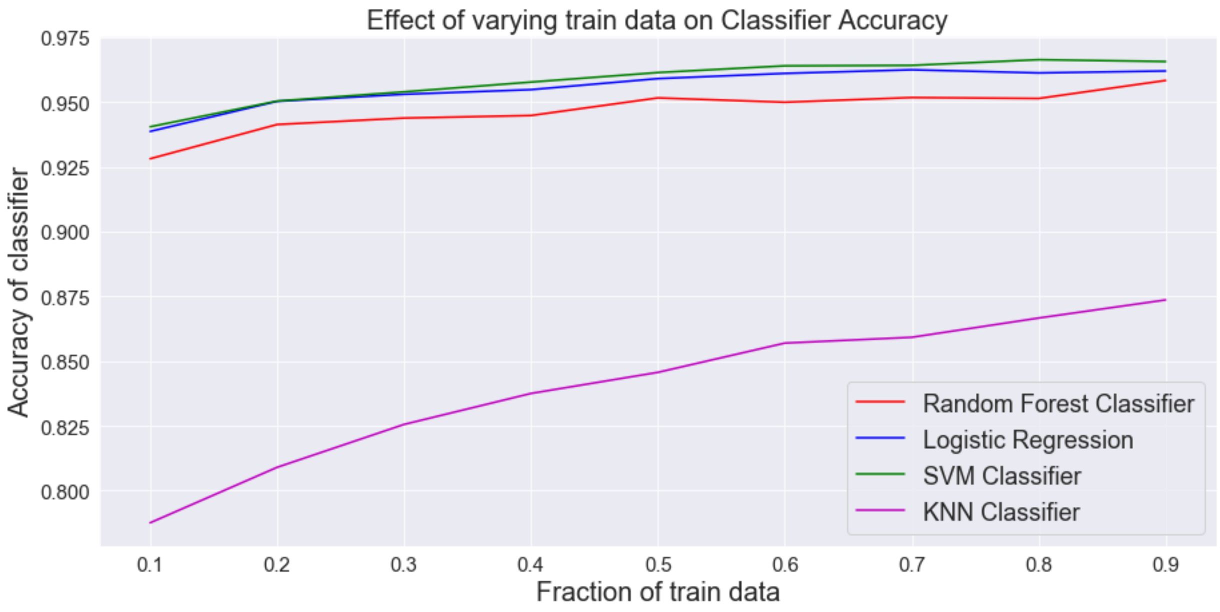
```
accuracy 0.8736303871439006
accuracy 0.866642309097552
accuracy 0.8591961023142509
accuracy 0.8569601753744976
accuracy 0.8456591639871383
accuracy 0.8375152253349574
accuracy 0.8255559035389916
accuracy 0.8089887640449438
accuracy 0.7875761266747868
[0.8736303871439006, 0.866642309097552, 0.8591961023142509, 0.8569601753744976,
0.8456591639871383, 0.8375152253349574, 0.8255559035389916, 0.8089887640449438,
0.7875761266747868]
```

In [32]:
```
import matplotlib.pyplot as plt
import numpy as np

x = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9]
y1=acc_test
y2=acc_log_test
y3=acc_svm_test
y4=acc_k_test
sns.set_style("darkgrid")

fig, ax = plt.subplots(figsize=(15,7))

plt.plot(x,y1,'r-',label='Random Forest Classifier')
plt.plot(x,y2,'b-',label='Logistic Regression')
plt.plot(x,y3,'g-',label='SVM Classifier')
plt.plot(x,y4,'m-',label='KNN Classifier')


ax.legend( prop={'size': 18})

ax.set_xlabel('Fraction of test data',fontsize=20)
ax.set_ylabel('Accuracy of classifier',fontsize=20)
ax.set_title("Effect of varying test data on Classifier Accuracy",fontsize=20)

ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```



In [33]:

```python
import matplotlib.pyplot as plt
import numpy as np

x = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9]
y1=acc_train
y2=acc_log_train
y3=acc_svm_train
y4=acc_k_train
sns.set_style("darkgrid")

fig, ax = plt.subplots(figsize=(15,7))

plt.plot(x,y1,'r-',label='Random Forest Classifier')
plt.plot(x,y2,'b-',label='Logistic Regression')
plt.plot(x,y3,'g-',label='SVM Classifier')
plt.plot(x,y4,'m-',label='KNN Classifier')

ax.legend( prop={'size': 18})

ax.set_xlabel('Fraction of train data',fontsize=20)
ax.set_ylabel('Accuracy of classifier',fontsize=20)
ax.set_title("Effect of varying train data on Classifier Accuracy",fontsize=20)
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```



In [3]:
```
df1 = pd.read_csv('f_a3.csv')
df1.columns=['index','date','tweet','countnoun','countverb','countadj','countadp'
df1
```

Out[3]:

	index	date	tweet	countnoun	countverb	countadj	countadp	cou
0	3	2019-09-27	time hear elite wealthy democrat guncontrol re...	11	3	4	0	
1	4	2019-09-24	olofsdotterk royarahmani nzambassadorus mars...	11	4	3	0	
2	5	2019-09-25	arizona state representative jen longdon gunvi...	16	2	2	0	
3	6	2019-09-20	kamalaharris lot senatemajldr senategop stup...	7	3	3	0	
4	7	2019-09-26	ugh straight heart gopcomplicittraitors feels ...	10	2	2	0	
5	8	2019-09-19	democrats jumping board guncontrol surprising ...	11	3	1	0	
6	9	2019-09-27	rt gun_control_ca doctors speak truth lines co	8	1	2	0	

In [26]:
```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.model_selection import cross_validate
clf = make_pipeline(TfidfVectorizer(), svm.SVC(kernel='linear'))

scores = cross_validate(clf, df1['tweet'], df1['target'], scoring=['accuracy'], c
print(scores)
```

```
{'fit_time': array([17.06840873, 16.07110929, 16.11045265, 16.32937288, 15.6303
6585]), 'score_time': array([3.09394646, 3.00034213, 3.21751523, 2.90204763, 2.
69931173]), 'test_accuracy': array([0.96675192, 0.96054074, 0.96967483, 0.95942
982, 0.96380256])}
```

In [27]:
```
scores
```

Out[27]:
```
{'fit_time': array([17.06840873, 16.07110929, 16.11045265, 16.32937288, 15.6303
6585]),
 'score_time': array([3.09394646, 3.00034213, 3.21751523, 2.90204763, 2.6993117
3]),
 'test_accuracy': array([0.96675192, 0.96054074, 0.96967483, 0.95942982, 0.9638
0256])}
```

In [28]:
```
scores['fit_time']
time_arr=scores['fit_time']
print (time_arr)
```

```
[17.06840873 16.07110929 16.11045265 16.32937288 15.63036585]
```

In [29]:

```
scores['score_time']
```

Out[29]:

```
array([3.09394646, 3.00034213, 3.21751523, 2.90204763, 2.69931173])
```

In [30]:

```
scores['test_accuracy']
accuracy_arr=scores['test_accuracy']
print (accuracy_arr)
```

```
[0.96675192 0.96054074 0.96967483 0.95942982 0.96380256]
```

In [ ]:

In [31]:

```
print (arr)
```

```
[0.96675192 0.96054074 0.96967483 0.95942982 0.96380256]
```

In [32]:

```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.model_selection import cross_validate
clf = make_pipeline(TfidfVectorizer(), svm.SVC(kernel='linear'))

scores10 = cross_validate(clf, df1['tweet'], df1['target'], scoring=['accuracy'],
print(scores10)
```

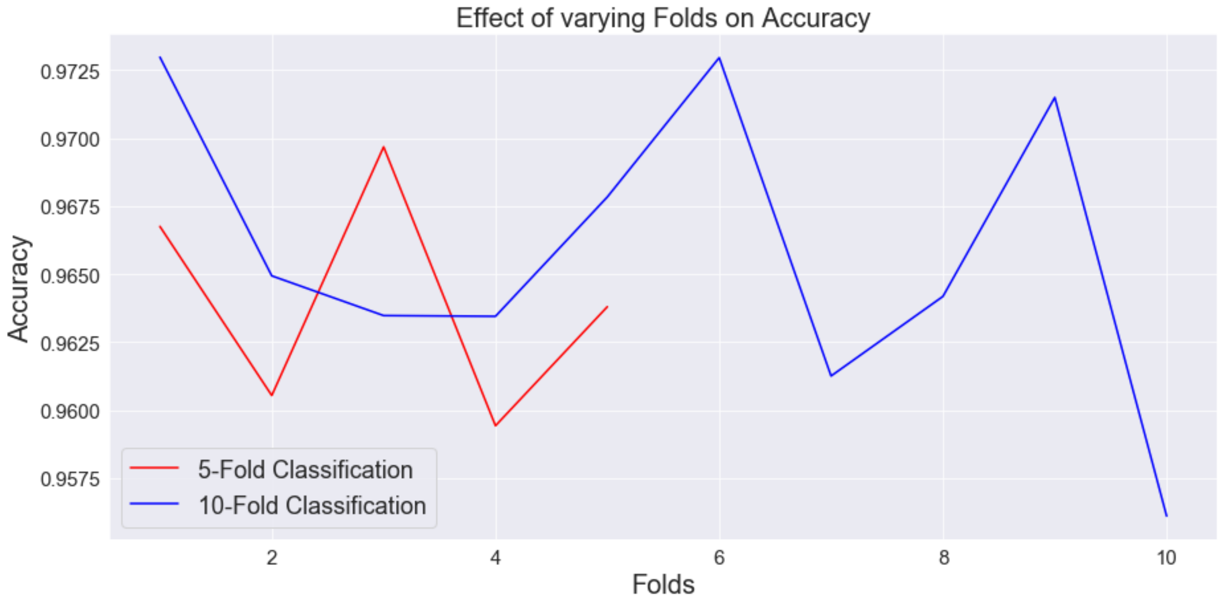
```
{'fit_time': array([20.79476452, 20.51393557, 20.65089893, 20.80029988, 20.3109
5076,
       20.65039587, 21.47918844, 21.59703183, 20.13253212, 20.59832048]), 'scor
e_time': array([1.67945838, 1.50107169, 1.75245571, 1.7122004 , 1.81509709,
       1.68341279, 1.77158451, 1.70439029, 1.38812757, 1.46474504]), 'test_accu
racy': array([0.97297297, 0.96493791, 0.96347699, 0.96345029, 0.96783626,
       0.97295322, 0.96125731, 0.96418129, 0.97149123, 0.95610827])}
```

In [33]:

```
scores10['test_accuracy']
time_arr10=scores10['test_accuracy']
print (time_arr10)
```

```
[0.97297297 0.96493791 0.96347699 0.96345029 0.96783626 0.97295322
 0.96125731 0.96418129 0.97149123 0.95610827]
```

In [34]:
```
import matplotlib.pyplot as plt
import numpy as np

x1 = [1,2,3,4,5]
x2=  [1,2,3,4,5,6,7,8,9,10]
y1=accuracy_arr
y2=time_arr10
sns.set_style("darkgrid")

fig, ax = plt.subplots(figsize=(15,7))

plt.plot(x1,y1,'r-',label='5-Fold Classification')
plt.plot(x2,y2,'b-',label='10-Fold Classification')
ax.legend( prop={'size': 18})

ax.set_xlabel('Folds',fontsize=20)
ax.set_ylabel('Accuracy',fontsize=20)
ax.set_title("Effect of varying Folds on Accuracy",fontsize=20)
ax = plt.gca()
ax.tick_params(axis = 'both', which = 'major', labelsize = 15)
```



In [ ]:
```
```