Stock Sentiment Analysis using News Headlines

In [1]:
```
import pandas as pd
```

In [2]:
```
path = 'D:/Projects/Stock-Sentiment-Analysis/'
```

In [5]:
```
df = pd.read_csv(path+'Stock_News_Dataset.csv',encoding='ISO-8859-1')
```

In [8]:
```
df.head()
```

Out[8]:

	Date	Label	Top1	Top2	Top3	Top4	Top5	Top6	Top7	
0	2000-01-03	0	A 'hindrance to operations': extracts from the...	Scorecard	Hughes' instant hit buoys Blues	Jack gets his skates on at ice-cold Alex	Chaos as Maracana builds up for United	Depleted Leicester prevail as Elliott spoils E...	Hungry Spurs sense rich pickings	G so
1	2000-01-04	0	Scorecard	The best lake scene	Leader: German sleaze inquiry	Cheerio, boyo	The main recommendations	Has Cubie killed fees?	Has Cubie killed fees?	Ha
2	2000-01-05	0	Coventry caught on counter by Flo	United's rivals on the road to Rio	Thatcher issues defence before trial by video	Police help Smith lay down the law at Everton	Tale of Trautmann bears two more retellings	England on the rack	Pakistan retaliate with call for video of Walsh	C co h mc
3	2000-01-06	1	Pilgrim knows how to progress	Thatcher facing ban	McIlroy calls for Irish fighting spirit	Leicester bin stadium blueprint	United braced for Mexican wave	Auntie back in fashion, even if the dress look...	Shoaib appeal goes to the top	I 'sha I bl
4	2000-01-07	1	Hitches and Horlocks	Beckham off but United survive	Breast cancer screening	Alan Parker	Guardian readers: are you all whingers?	Hollywood Beyond	Ashes and diamonds	W forr n

5 rows × 27 columns

In [11]:
```
train = df[df['Date']<'20150101']
test = df[df['Date']>'20141231']
```

In [24]:
```
#Removing punctuations
def punct_rem(dataset):
    df = dataset.copy()
    data = df.iloc[:,2:]
    data.replace('[^a-zA-Z]',' ',regex=True, inplace=True)
    return data

#Rename columns for ease of access
def col_rename(df):
    l1 = [str(i) for i in range(len(df.columns))]
    df.columns = l1
    return df

#converting headlines to lower cases
def lower_case(df):
    for i in df.columns:
        df[i] = df[i].str.lower()
    return df

#joining all the headlines of particular row into single healines
def join_headlines(df):
    headlines = []
    for row in range(len(df)):
        headlines.append(' '.join(str(x) for x in df.iloc[row,:]))
    return headlines
```

In [23]:
```
data = punct_rem(train)
data = col_rename(data)
data = lower_case(data)
data.head()
```

Out[23]:

0	1	2	3	4	5	6	7

	0	1	2	3	4	5	6	7	
0	a hindrance to operations extracts from the...	scorecard	hughes instant hit buoys blues	jack gets his skates on at ice cold alex	chaos as maracana builds up for united	depleted leicester prevail as elliott spoils e...	hungry spurs sense rich pickings	gunners so wide of an easy target	der raise glass strupa deb doub
1	scorecard	the best lake scene	leader german sleaze inquiry	cheerio boyo	the main recommendations	has cubie killed fees	has cubie killed fees	has cubie killed fees	hopki furious foste lack hanniba
2	coventry caught on counter by flo	united s rivals on the road to rio	thatcher issues defence before trial by video	police help smith lay down the law at everton	tale of trautmann bears two more retellings	england on the rack	pakistan retaliate with call for video of walsh	cullinan continues his cape monopoly	mcgra puts inc out th mise
3	pilgrim knows how to progress	thatcher facing ban	mcilroy calls for irish fighting spirit	leicester bin stadium blueprint	united braced for mexican wave	auntie back in fashion even if the dress look...	shoaib appeal goes to the top	hussain hurt by shambles but lays blame on e...	englanc decade disaste
4	hitches and horlocks	beckham off but united survive	breast cancer screening	alan parker	guardian readers are you all whingers	hollywood beyond	ashes and diamonds	whingers a formidable minority	al park part tv

5 rows × 25 columns

In [25]:
```
headline = join_headlines(data)
```

In [29]:
```
headline[4]
```

Out[29]:
```
'hitches and horlocks beckham off but united survive breast cancer screening alan parker
guardian readers  are you all whingers  hollywood beyond ashes and diamonds whingers   a
formidable minority alan parker   part two thuggery  toxins and ties met faces fresh att
ack on race crime everton fans top racist  league of shame  our breasts  ourselves russi
a s new boss has an extremely strange history always and forever most everywhere   udis
most wanted   chloe lunettes return of the cane  completely off the agenda  from sleepy
hollow to greeneland blunkett outlines vision for over   s embattled dobson attacks  pla
```

```
y now  pay later  livingstone doom and the dome what is the north south divide  aitken r
eleased from jail gone aloft'
```

In [34]:
```python
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.ensemble import RandomForestClassifier as RFC
```

In [33]:
```python
#implement Bag of Words
cv = CountVectorizer(ngram_range=(2,2))
trainset = cv.fit_transform(headline)
```

In [37]:
```python
#implement RFC
rfc = RFC(n_estimators=200, criterion='entropy')
rfc.fit(trainset, train['Label'])
```

Out[37]:
```
RandomForestClassifier(criterion='entropy', n_estimators=200)
```

In [39]:
```python
#predict for testset
data = punct_rem(test)
data = col_rename(data)
data = lower_case(data)
headline = join_headlines(data)
```

In [41]:
```python
testset = cv.transform(headline)
pred = rfc.predict(testset)
```

In [42]:
```python
pred
```

Out[42]:
```
array([1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0,
       1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1,
       1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1,
       0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1,
       1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0,
       1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1,
       0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1,
       1, 1, 1, 1], dtype=int64)
```

In [43]:
```
#Import libraries to check accuracy
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score
```

In [44]:
```
matrix = confusion_matrix(test['Label'],pred)
print(matrix)
```

```
[[139  47]
 [  7 185]]
```

In [45]:
```
score = accuracy_score(test['Label'],pred)
print(score)
```

```
0.8571428571428571
```

In [46]:
```
report = classification_report(test['Label'],pred)
print(report)
```

```
              precision    recall  f1-score   support

           0       0.95      0.75      0.84       186
           1       0.80      0.96      0.87       192

    accuracy                           0.86       378
   macro avg       0.87      0.86      0.85       378
weighted avg       0.87      0.86      0.86       378
```

In [ ]:
```

```