Subject : AML

Subject Code : 3CS1111

Roll No : 20MCED08

Bernouli

In [1]:
```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import BernoulliNB
```

Data Import

In [2]:
```
data = pd.read_csv('news.csv')
```

In [3]:
```
data.head(1)
```

Out[3]:

	Date	Label	Top1	Top2	Top3	Top4	Top5	Top6	Top7	
0	2008-08-08	0	b"Georgia 'downs two Russian warplanes' as cou...	b'BREAKING: Musharraf to be impeached.'	b'Russia Today: Columns of troops roll into So...	b'Russian tanks are moving towards the capital...	b"Afghan children raped with 'impunity,' U.N. ...	b'150 Russian tanks have entered South Ossetia...	b"Breaking: Georgia invades South Ossetia, Rus...	cc

1 rows × 27 columns

In [4]:

```
data.tail(1)
```

Out[4]:

	Date	Label	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Tc
1988	2016-07-01	1	A 117-year-old woman in Mexico City finally re...	IMF chief backs Athens as permanent Olympic host	The president of France says if Brexit won, so...	British Man Who Must Give Police 24 Hours' Not...	100+ Nobel laureates urge Greenpeace to stop o...	Brazil: Huge spike in number of police killing...	Austria's highest court annuls presidential el...	Facebo w priva case, c track a Bel

1 rows × 27 columns

Next, let's take a look at the data with the head (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.head.html) method.

In [5]:

```
data.shape
```

Out[5]:

```
(1989, 27)
```

We've got a lot of vaiables here, but the layout is pretty straight-forward.
As a reminder, the Label variable will be a **1** if the DJIA **stayed the same or rose** on that date or **0** if the DJIA **fell** on that date.

In [6]:

```
train = data[data['Date'] < '2015-01-01']
test = data[data['Date'] > '2014-12-31']
```

Text Preprocessing

Now that our data is loaded in, we need to clean it up just a little bit to prepare it for the rest of our analysis.
To illustrate this process, look at how the example headline below changes from cell to cell.
Don't worry about the code too much here, since this example is only meant to be visual.

```
In [7]: train.tail()
```

Out[7]:

	Date	Label	Top1	Top2	Top3	Top4	Top5	
1606	2014-12-24	1	Death toll among Qatars 2022 World Cup workers...	Fishing Supertrawlers to be banned permanently...	Indian telecommunications company Airtel viola...	North Korea's Internet is down again; second b...	Jakarta to ban virginity tests for female civi...	W fact
1607	2014-12-26	1	Saudis are eagerly awaiting the approval of a ...	Due to the fall in oil prices, Saudi Arabia is...	Bill giving government the power to shutdown t...	A struggle for women's rights is brewing withi...	Putin cancels New Year's Holiday for governmen...	I si Mi it
1608	2014-12-29	0	Solar Power Storage Prices Drop 25% In Germany...	North Korea Hit Again By Internet Outage; Expe...	ARCHAEOLOGY - Massive ancient underground city...	Reopen investigation into Westminster pedophil...	Taliban declare 'defeat' of U.S., allies in Af...	F rev
1609	2014-12-30	0	China businessman jailed for 13 years for buyi...	AirAsia live: Emergency slide, plane door seen...	AirAsia plane wreckage found, bodies being rec...	Scotland confirms case of Ebola - Ebola cases ...	Pope Francis to Catholics: It's time to take a...	Oil ar bi
1610	2014-12-31	0	AirAsia flight found at the bottom of the Java...	North Korean defector details 'human experimen...	Korean Air ex-executive Cho Hyun-ah arrested -...	South Korean to drop Sony film "The Interview"...	U.S. opening of oil export widens battle: The ...	FE inve

5 rows × 27 columns

Were you able to see everything that changed?
The process involved:

- Converting the headline to lowercase letters
- Splitting the sentence into a list of words
- Transforming that list into a table of counts

Basic Model Training and Testing

In [8]:

```
trainheadlines = []
for row in range(0,len(train.index)):
    trainheadlines.append(' '.join(str(x) for x in train.iloc[row,2:27]))
```

In [9]:

```
train.shape
```

Out[9]:

```
(1611, 27)
```

In [10]:

```
basicvectorizer = CountVectorizer()
basictrain = basicvectorizer.fit_transform(trainheadlines)
print(basictrain.shape)
```

```
(1611, 31675)
```

In [11]:
```
print(basictrain)
```

```
  (0, 12120)	10
  (0, 9116)	1
  (0, 29313)	2
  (0, 24572)	5
  (0, 30656)	1
  (0, 2705)	1
  (0, 7138)	2
  (0, 18619)	1
  (0, 28628)	7
  (0, 4631)	1
  (0, 20034)	11
  (0, 30614)	5
  (0, 4542)	2
  (0, 18776)	1
  (0, 3561)	2
  (0, 14201)	1
  (0, 24571)	5
  (0, 28636)	2
  (0, 6326)	1
  (0, 29118)	2
  (0, 24352)	1
  (0, 14898)	1
  (0, 26517)	7
  (0, 20360)	7
  (0, 11399)	1
  :	:
  (1610, 17862)	2
  (1610, 15268)	1
  (1610, 19682)	1
  (1610, 5581)	2
  (1610, 16817)	1
  (1610, 21809)	1
  (1610, 2566)	1
  (1610, 7953)	1
  (1610, 2493)	1
  (1610, 12953)	1
  (1610, 25304)	1
  (1610, 25456)	1
  (1610, 19618)	1
  (1610, 4605)	1
  (1610, 1810)	1
  (1610, 19943)	1
  (1610, 17897)	1
  (1610, 6935)	1
  (1610, 24430)	1
  (1610, 5813)	1
  (1610, 22065)	1
  (1610, 12230)	1
  (1610, 26511)	1
  (1610, 8195)	1
  (1610, 6235)	1
```

Wow! Our resulting table contains counts for 31,675 different words!

Now, let's train a logistic regression model using this data.
In the cell below, we're simply naming our model, then fitting (http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRe
the model based on our X and Y values.

In [12]:
```
basicmodel = BernoulliNB()
basicmodel = basicmodel.fit(basictrain, train["Label"])
```

Our model is ready to go, so let's set up our test data.
Here, we're just going to repeat the steps we used to prep our training data, then predict (http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRe
whether the DJIA increased or decreased for each day in the test dataset.

In [13]:
```
testheadlines = []
for row in range(0,len(test.index)):
    testheadlines.append(' '.join(str(x) for x in test.iloc[row,2:27]))
basictest = basicvectorizer.transform(testheadlines)
predictions = basicmodel.predict(basictest)
```

In [14]:
```
from sklearn.metrics import classification_report,confusion_matrix,accuracy_sc
ore
matrix=confusion_matrix(test['Label'],predictions)
print(matrix)
score=accuracy_score(test['Label'],predictions)
print(score)
report=classification_report(test['Label'],predictions)
print(report)
```

```
[[ 35 151]
 [ 49 143]]
0.4708994708994709
              precision    recall  f1-score   support

           0       0.42      0.19      0.26       186
           1       0.49      0.74      0.59       192

    accuracy                           0.47       378
   macro avg       0.45      0.47      0.42       378
weighted avg       0.45      0.47      0.43       378
```

In [ ]:

In [ ]: