About the Dataset:

1. id: unique id for a news article
2. title: the title of a news article
3. author: author of the news article
4. text: the text of the article; could be incomplete
5. label: a label that marks whether the news article is real or fake:

```
1: Fake news
0: real News
```

Importing the Dependencies

In [1]:
```
import pandas as pd # for importing the dataset and insert item into dataframe
import re           # re stands for regular expression and it is used to search text in a paragraph
from nltk.corpus import stopwords  # there are many words in many languages which are not much important so stopwords
# gives words in a particular language which are not useful and can be removed from the text for further analysis

from nltk.stem.porter import PorterStemmer  # used to get the stem word of different forms of word
from sklearn.feature_extraction.text import TfidfVectorizer  # tf stands for term frequency and idf stands for inverse document

# frequency. basically this module helps to convert textual data to featured data in numerical form beacuse computer can't
# understand textual data we need to convert it to first numerical data and then apply some machine learning algorithm in it.
# tf basically selects those words which have many occurances in the text and predict that this word might be much important
# and assign some numerical value to it.
# while at the same time it may happen the words which are coming very frequently doesn't have any exact meaning in itself this
# is taken care by idf (inverse document frequency)


from sklearn.model_selection import train_test_split  # for splitting the dataset into train and testing
from sklearn.linear_model import LogisticRegression   # for prediction
from sklearn.metrics import accuracy_score            # to calculate the accuracy score
```

In [2]:
```
import nltk
nltk.download('stopwords')
```

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

Out[2]:
```
True
```

In [3]:
```
# printing the stopwords in English
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'hi
s', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'who
m', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing',
'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'duri
ng', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there',
'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than',
'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn',
"couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "must
n't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

Data Pre-processing

In [4]:
```
# loading the dataset to a pandas DataFrame
news_dataset = pd.read_csv('train.csv')
```

In [5]:
```
news_dataset.shape
```

Out[5]:
```
(20800, 5)
```

In [6]:
```
# print the first 5 rows of the dataframe
news_dataset.head()
```

Out[6]:

	id	title	author	text	label
0	0	House Dem Aide: We Didn’t Even See Comey’s Let...	Darrell Lucus	House Dem Aide: We Didn’t Even See Comey’s Let...	1
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Airstr...	1
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1

In [7]:
```
# counting the number of missing values in the dataset
news_dataset.isnull().sum()
```

Out[7]:
```
id          0
title     558
author   1957
text       39
label       0
dtype: int64
```

In [8]:
```
# replacing the null values with empty string
news_dataset = news_dataset.fillna('')
```

In [9]:
```
# merging the author name and news title
news_dataset['content'] = news_dataset['author']+' '+news_dataset['title']
```

In [10]:
```
print(news_dataset['content'])
```

```
0        Darrell Lucus House Dem Aide: We Didn’t Even S...
1        Daniel J. Flynn FLYNN: Hillary Clinton, Big Wo...
2        Consortiumnews.com Why the Truth Might Get You...
3        Jessica Purkiss 15 Civilians Killed In Single ...
4        Howard Portnoy Iranian woman jailed for fictio...
                              ...
20795    Jerome Hudson Rapper T.I.: Trump a ’Poster Chi...
20796    Benjamin Hoffman N.F.L. Playoffs: Schedule, Ma...
20797    Michael J. de la Merced and Rachel Abrams Macy...
20798    Alex Ansary NATO, Russia To Hold Parallel Exer...
20799              David Swanson What Keeps the F-35 Alive
Name: content, Length: 20800, dtype: object
```

In [11]:
```
# separating the data & label
X = news_dataset.drop(columns='label', axis=1)
Y = news_dataset['label']
```

In [12]:
```
print(X)
print(Y)
```

```
          id                                              title  \
0          0  House Dem Aide: We Didn’t Even See Comey’s Let...
1          1  FLYNN: Hillary Clinton, Big Woman on Campus - ...
2          2                  Why the Truth Might Get You Fired
3          3  15 Civilians Killed In Single US Airstrike Hav...
4          4  Iranian woman jailed for fictional unpublished...
...      ...                                                ...
20795  20795  Rapper T.I.: Trump a ’Poster Child For White S...
20796  20796  N.F.L. Playoffs: Schedule, Matchups and Odds -...
20797  20797  Macy’s Is Said to Receive Takeover Approach by...
20798  20798  NATO, Russia To Hold Parallel Exercises In Bal...
20799  20799                          What Keeps the F-35 Alive

                                     author  \
0                                Darrell Lucus
1                               Daniel J. Flynn
2                             Consortiumnews.com
3                                Jessica Purkiss
4                                 Howard Portnoy
...                                          ...
20795                             Jerome Hudson
20796                           Benjamin Hoffman
20797  Michael J. de la Merced and Rachel Abrams
20798                               Alex Ansary
20799                             David Swanson

                                                  text  \
0      House Dem Aide: We Didn’t Even See Comey’s Let...
1      Ever get the feeling your life circles the rou...
2      Why the Truth Might Get You Fired October 29, ...
3      Videos 15 Civilians Killed In Single US Airstr...
4      Print \nAn Iranian woman has been sentenced to...
...                                                  ...
20795  Rapper T. I. unloaded on black celebrities who...
20796  When the Green Bay Packers lost to the Washing...
20797  The Macy’s of today grew from the union of sev...
20798  NATO, Russia To Hold Parallel Exercises In Bal...
20799    David Swanson is an author, activist, journa...

                                                 content
0      Darrell Lucus House Dem Aide: We Didn’t Even S...
1      Daniel J. Flynn FLYNN: Hillary Clinton, Big Wo...
2      Consortiumnews.com Why the Truth Might Get You...
3      Jessica Purkiss 15 Civilians Killed In Single ...
4      Howard Portnoy Iranian woman jailed for fictio...
...                                                  ...
20795  Jerome Hudson Rapper T.I.: Trump a ’Poster Chi...
20796  Benjamin Hoffman N.F.L. Playoffs: Schedule, Ma...
20797  Michael J. de la Merced and Rachel Abrams Macy...
20798  Alex Ansary NATO, Russia To Hold Parallel Exer...
20799            David Swanson What Keeps the F-35 Alive

[20800 rows x 5 columns]
0        1
1        0
2        1
```

```
3        1
4        1
        ..
20795    0
20796    0
20797    0
20798    1
20799    1
Name: label, Length: 20800, dtype: int64
```

Stemming:

Stemming is the process of reducing a word to its Root word

example: actor, actress, acting --> act

In [13]:
```
port_stem = PorterStemmer()
```

In [14]:
```
def stemming(content):
    stemmed_content = re.sub('[^a-zA-Z]',' ',content)
    stemmed_content = stemmed_content.lower()
    stemmed_content = stemmed_content.split()
    stemmed_content = [port_stem.stem(word) for word in stemmed_content if not word in stopwords.words('english')]
    stemmed_content = ' '.join(stemmed_content)
    return stemmed_content
```

In [15]:
```
news_dataset['content'] = news_dataset['content'].apply(stemming)
```

In [16]:
```
print(news_dataset['content'])
```

```
0        darrel lucu hous dem aid even see comey letter...
1        daniel j flynn flynn hillari clinton big woman...
2                   consortiumnew com truth might get fire
3        jessica purkiss civilian kill singl us airstri...
4        howard portnoy iranian woman jail fiction unpu...
                               ...
20795    jerom hudson rapper trump poster child white s...
20796    benjamin hoffman n f l playoff schedul matchup...
20797    michael j de la merc rachel abram maci said re...
20798    alex ansari nato russia hold parallel exercis ...
20799                            david swanson keep f aliv
Name: content, Length: 20800, dtype: object
```

In [17]:
```
#separating the data and label
X = news_dataset['content'].values
Y = news_dataset['label'].values
```

In [18]:
```
print(X)
```

```
['darrel lucu hous dem aid even see comey letter jason chaffetz tweet'
 'daniel j flynn flynn hillari clinton big woman campu breitbart'
 'consortiumnew com truth might get fire' ...
 'michael j de la merc rachel abram maci said receiv takeov approach hudson bay new york time'
 'alex ansari nato russia hold parallel exercis balkan'
 'david swanson keep f aliv']
```

In [19]:
```
print(Y)
```

```
[1 0 1 ... 0 1 1]
```

In [20]:
```
Y.shape
```

Out[20]:
```
(20800,)
```

In [21]:
```
# converting the textual data to numerical data
vectorizer = TfidfVectorizer()
vectorizer.fit(X)

X = vectorizer.transform(X)
```

In [22]:
```
print(X)
```

```
  (0, 15686)    0.28485063562728646
  (0, 13473)    0.2565896679337957
  (0, 8909)     0.3635963806326075
  (0, 8630)     0.29212514087043684
  (0, 7692)     0.24785219520671603
  (0, 7005)     0.21874169089359144
  (0, 4973)     0.233316966909351
  (0, 3792)     0.2705332480845492
  (0, 3600)     0.3598939188262559
  (0, 2959)     0.2468450128533713
  (0, 2483)     0.3676519686797209
  (0, 267)      0.27010124977708766
  (1, 16799)    0.30071745655510157
  (1, 6816)     0.1904660198296849
  (1, 5503)     0.7143299355715573
  (1, 3568)     0.26373768806048464
  (1, 2813)     0.19094574062359204
  (1, 2223)     0.3827320386859759
  (1, 1894)     0.15521974226349364
  (1, 1497)     0.2939891562094648
  (2, 15611)    0.41544962664721613
  (2, 9620)     0.49351492943649944
  (2, 5968)     0.3474613386728292
  (2, 5389)     0.3866530551182615
  (2, 3103)     0.46097489583229645
  :     :
  (20797, 13122)        0.2482526352197606
  (20797, 12344)        0.27263457663336677
  (20797, 12138)        0.24778257724396507
  (20797, 10306)        0.08038079000566466
  (20797, 9588) 0.174553480255222
  (20797, 9518) 0.2954204003420313
  (20797, 8988) 0.36160868928090795
  (20797, 8364) 0.22322585870464118
  (20797, 7042) 0.21799048897828688
  (20797, 3643) 0.21155500613623743
  (20797, 1287) 0.33538056804139865
  (20797, 699)  0.30685846079762347
  (20797, 43)   0.29710241860700626
  (20798, 13046)        0.22363267488270608
  (20798, 11052)        0.4460515589182236
  (20798, 10177)        0.3192496370187028
  (20798, 6889) 0.32496285694299426
  (20798, 5032) 0.4083701450239529
  (20798, 1125) 0.4460515589182236
  (20798, 588)  0.3112141524638974
  (20798, 350)  0.28446937819072576
  (20799, 14852)        0.5677577267055112
  (20799, 8036) 0.45983893273780013
  (20799, 3623) 0.37927626273066584
  (20799, 377)  0.5677577267055112
```

Splitting the dataset to training & test data

In [23]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, stratify=Y, random_state=0)
```

Training the Model: Logistic Regression

In [24]:
```
model = LogisticRegression()
```

In [25]:
```
model.fit(X_train, Y_train)
```

Out[25]:
```
LogisticRegression()
```

Evaluation

accuracy score

In [26]:
```
# accuracy score on the training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

In [27]:
```
print('Accuracy score of the training data : ', training_data_accuracy)
```

```
Accuracy score of the training data :  0.9866586538461538
```

In [28]:
```
# accuracy score on the test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

In [29]:
```
print('Accuracy score of the test data : ', test_data_accuracy)
```

```
Accuracy score of the test data :  0.9774038461538461
```

Making a Predictive System

In [30]:
```
X_new = X_test[3]

prediction = model.predict(X_new)
print(prediction)

if (prediction[0]==0):
  print('The news is Real')
else:
  print('The news is Fake')
```

```
[1]
The news is Fake
```

In [31]:
```
print(Y_test[3])
```

```
1
```

In [36]:
```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=3)
neigh.fit(X_train, Y_train)
X_test_prediction = neigh.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
test_data_accuracy
```

Out[36]:
```
0.5377403846153846
```

In [40]:
```
from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train, Y_train)
X_test_prediction = clf.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
test_data_accuracy
```

Out[40]:
```
0.9913461538461539
```

In [41]:
```
from sklearn import svm
clf = svm.SVC()
clf.fit(X_train, Y_train)
X_test_prediction = clf.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
test_data_accuracy
```

Out[41]:
```
0.9889423076923077
```

In [52]:
```
# import numpy
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
Y_pred = gnb.fit(X_train, Y_train).predict(X_test)
test_data_accuracy = accuracy_score(Y_pred, Y_test)
test_data_accuracy
```

Out[52]:
```
0.8
```

In [ ]:
```

```