Fake_News_Prediction

November 30, 2022

```
[ ]: 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
     Dataset Link: https://www.kaggle.com/c/fake-news/data?select=train.csv
```

Importing the Dependencies

```
[ ]: import numpy as np
     import pandas as pd
     import re
     from nltk.corpus import stopwords
     from nltk.stem.porter import PorterStemmer
     from sklearn.feature_extraction.text import TfidfVectorizer
     from sklearn.model_selection import train_test_split
     from sklearn.linear_model import LogisticRegression
     from sklearn.metrics import accuracy_score
```

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

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

```
[ ]: True
```

```
[ ]: # 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', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's",
'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what',
'which', 'who', 'whom', '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', 'during', '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', "mustn'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

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

[ ]:
```
news_dataset.shape
```

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

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

[ ]:
```
   id  … label
0   0  …     1
1   1  …     0
2   2  …     1
3   3  …     1
4   4  …     1

[5 rows x 5 columns]
```

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

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

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

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

```
[ ]: 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
```

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

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

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

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

[ ]:
```
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
```

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

[ ]:
```
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
```

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

[ ]:
```
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']
```

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

```
[1 0 1 … 0 1 1]
```

[ ]:
```
Y.shape
```

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

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

X = vectorizer.transform(X)
```

[ ]:
```
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

[ ]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2,␣
↪stratify=Y, random_state=2)
```

Training the Model: Logistic Regression

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

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

[ ]:
```
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)
```

Evaluation

accuracy score

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

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

```
Accuracy score of the training data :  0.9865985576923076
```

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

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

```
Accuracy score of the test data :  0.9790865384615385
```

Making a Predictive System

```
[ ]: 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')
```

```
[0]
The news is Real
```

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

```
0
```

```
[ ]:
```