

FAKE NEWS DETECTION

[+ Code](#)[+ Text](#)

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] Unzipping corpora/stopwords.zip.
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', 'yours

Data Pre-processing

```
# LOADING THE DATASET TO A PANDAS DATAFRAME
news_dataset = pd.read_csv('/content/news.csv')

<ipython-input-5-fd555db9e637>:2: DtypeWarning: Columns (24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48)
news_dataset = pd.read_csv('/content/news.csv')

news_dataset.shape

(7795, 141)

# PRINT THE FIRST 5 ROWS OF THE DATAFRAME
news_dataset.head()
```

```

    Unnamed: 0      title      text  label  Unnamed: 4  Unnamed: 5  Unnamed: 6  Unnamed: 7  Unnamed: 8  Unnamed: 9
0      You Can      Daniel
# COUNTING THE NUMBER OF MISSING VALUES IN THE DATASET
news_dataset.isnull().sum()

    Unnamed: 0      219
    title      610
    text      866
    label      1040
    Unnamed: 4      7477
    ...
    Unnamed: 136      7794
    Unnamed: 137      7794
    Unnamed: 138      7794
    Unnamed: 139      7794
    Unnamed: 140      7794
    Length: 141, dtype: int64

# REPLACING THE NULL VALUES WITH EMPTY STRING
news_dataset = news_dataset.fillna('')

    erupt in
print(news_dataset['title'])

0      You Can Smell Hillary's Fear
1      Watch The Exact Moment Paul Ryan Committed Pol...
2      Kerry to go to Paris in gesture of sympathy
3      Bernie supporters on Twitter erupt in anger ag...
4      The Battle of New York: Why This Primary Matters
    ...
7790      State Department says it can't find emails fro...
7791      The 'P' in PBS Should Stand for 'Plutocratic' ...
7792      Anti-Trump Protesters Are Tools of the Oligarc...
7793      In Ethiopia, Obama seeks progress on peace, se...
7794      Jeb Bush Is Suddenly Attacking Trump. Here's W...
Name: title, Length: 7795, dtype: object

# SEPARATING THE DATA AND LABEL
X = news_dataset.drop(columns='label', axis=1)
Y = news_dataset['label']

print(X)
print(Y)

    Unnamed: 0      title \
0      8476      You Can Smell Hillary's Fear
1      10294      Watch The Exact Moment Paul Ryan Committed Pol...
2      3608      Kerry to go to Paris in gesture of sympathy
3      10142      Bernie supporters on Twitter erupt in anger ag...
4      875      The Battle of New York: Why This Primary Matters
    ...      ...
7790      4490      State Department says it can't find emails fro...
7791      8062      The 'P' in PBS Should Stand for 'Plutocratic' ...
7792      8622      Anti-Trump Protesters Are Tools of the Oligarc...
7793      4021      In Ethiopia, Obama seeks progress on peace, se...
7794      4330      Jeb Bush Is Suddenly Attacking Trump. Here's W...

    text  Unnamed: 4  Unnamed: 5 \
0      Daniel Greenfield, a Shillman Journalism Fello...
1      Google Pinterest Digg LinkedIn Reddit Stumbleu...
2      U.S. Secretary of State John F. Kerry said Mon...
3      - Kaydee King (@KaydeeKing) November 9, 2016 T...
4      It's primary day in New York and front-runners...
    ...      ...      ...      ...
7790      The State Department told the Republican Natio...
7791      The 'P' in PBS Should Stand for 'Plutocratic' ...
7792      Anti-Trump Protesters Are Tools of the Oligar...
7793      ADDIS ABABA, Ethiopia -President Obama convene...
7794      Jeb Bush Is Suddenly Attacking Trump. Here's W...

    Unnamed: 6  Unnamed: 7  Unnamed: 8  Unnamed: 9  Unnamed: 10  ... \
0      ...
1      ...
2      ...
3      ...
4      ...
    ...      ...      ...      ...      ...      ...
7790      ...
7791      ...
7792      ...
7793      ...
7794      ...
```

```

    Unnamed: 131 Unnamed: 132 Unnamed: 133 Unnamed: 134 Unnamed: 135 \
0
1
2
3
4
...
7790
7791
7792
7793
7794

    Unnamed: 136 Unnamed: 137 Unnamed: 138 Unnamed: 139 Unnamed: 140
0
1
2
3
4

```

▼ Stemming:

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

example: actor, actress, acting -> act

```

port_stem = PorterStemmer()

def stemming(title):
    stemmed_content = re.sub('[^a-zA-Z]', ' ', title)
    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['title'] = news_dataset['title'].apply(stemming)

print(news_dataset['title'])

0          smell hillari fear
1  watch exact moment paul ryan commit polit suic...
2          kerri go pari gestur sympathi
3  berni support twitter erupt anger dnc tri warn
4          battl new york primari matter
...
7790  state depart say find email clinton specialist
7791          p pb stand plutocrat pentagon
7792  anti trump protest tool oligarchi inform
7793  ethiopia obama seek progress peac secur east a...
7794          jeb bush suddenli attack trump matter
Name: title, Length: 7795, dtype: object

# SEPARATING THE DATA AND LABEL
X = news_dataset['title'].values
Y = news_dataset['label'].values

print(X)

['smell hillari fear'
'watch exact moment paul ryan commit polit suicid trump ralli video'
'kerri go pari gestur sympathi' ...
'anti trump protest tool oligarchi inform'
'ethiopia obama seek progress peac secur east africa'
'jeb bush suddenli attack trump matter']

print(Y)

['FAKE' 'FAKE' 'REAL' ... 'FAKE' 'REAL' 'REAL']

Y.shape

(7795,)

```

```
# CONVERTING THE TEXTUAL DATA TO NUMERICAL DATA
vectorizer = TfidfVectorizer()
vectorizer.fit(X)
```

```
X = vectorizer.transform(X)
```

```
print(X)
```

```
(0, 6295)      0.7626938120805434
(0, 3151)      0.3268826814067037
(0, 2500)      0.5580733478767933
(1, 7456)      0.2829626896247514
(1, 7349)      0.22501798098545717
(1, 7061)      0.1436909337025267
(1, 6627)      0.35371370710604855
(1, 5872)      0.2990671877006965
(1, 5480)      0.2990671877006965
(1, 5175)      0.24803105695712827
(1, 4984)      0.27394480495762513
(1, 4402)      0.32880288996167356
(1, 2356)      0.4181701801321407
(1, 1360)      0.35371370710604855
(2, 6718)      0.5131942385297146
(2, 4943)      0.3854204598747483
(2, 3730)      0.4280117194264037
(2, 2882)      0.3291178772767753
(2, 2837)      0.5445825314314873
(3, 7443)      0.31941695607568427
(3, 7108)      0.37411079121086527
(3, 7030)      0.31105482541300034
(3, 6656)      0.2767436395536421
(3, 2293)      0.4452870054631296
(3, 1977)      0.35343688824367275
:              :
(7790, 1274)   0.21943768072462183
(7791, 6475)   0.38837550631498924
(7791, 5154)   0.5530598279361728
(7791, 5020)   0.4569318872066389
(7791, 4996)   0.5783619483261031
(7792, 7061)   0.18453904566694015
(7792, 6951)   0.537046590063328
(7792, 5364)   0.35021521811625833
(7792, 4769)   0.5116527902993385
(7792, 3416)   0.4030167794809183
(7792, 281)    0.3614754477985998
(7793, 6024)   0.3344008419999221
(7793, 6021)   0.3146109844084572
(7793, 5337)   0.3640211020574405
(7793, 4999)   0.3640211020574405
(7793, 4719)   0.20634173774684844
(7793, 2317)   0.42996472046873874
(7793, 2118)   0.37008467792918825
(7793, 114)    0.39970493798670664
(7794, 7061)   0.20353007667007753
(7794, 6619)   0.5769009825789405
(7794, 4203)   0.4266822088146035
(7794, 3586)   0.4100241838567959
(7794, 967)    0.38452915015504185
(7794, 427)    0.3573801312395893
```

Splitting the dataset to training & test data

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
```

Training the Model: Logistic Regression

```
model = LogisticRegression()
```

```
model.fit(X_train, Y_train)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs:
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

Increase the number of iterations (max_iter) or scale the data as shown in:

<https://scikit-learn.org/stable/modules/preprocessing.html>

Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

```
n_iter_i = _check_optimize_result(
```

```
    LogisticRegression
```

```
    LogisticRegression())
```

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 :  0.8678640153944837

# 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 :  0.7459910198845414
```

Making a Predictive System

```
X_new = X_test[3]

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

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

    ['FAKE']
    The news is Fake

print(Y_test[3])

    FAKE
```

✓ 0s completed at 9:54 PM

● ✕