Sadiq S.

Fake News Detection

Aim: To develop an NLP program to identify whether the news article is real or fake.

Our dataset has the following attributes:

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 the article as potentially unreliable
   - 1: unreliable
   - 0: reliable

We have ‘train.csv’ file which is our dataset.

It contains 2007 record almost 50% 1’s and 0’s.

We will check whether is there any null values. If there are any, we will fill it with empty strings.

We will need following libraries:

1. **Pandas**: Pandas is an open-source data analysis and manipulation tool which is built on top of the Python programming language.
   It is used to analyze and manipulate data in a flexible and easy way.
   Pandas provides fast, flexible, and expressive data structures designed to make working with "relational" or "labelled" data both easy and intuitive.

2. **Stopwords**: Stop words are a set of commonly used words in any language. For example, in English, “the”, “is” and “and”, would easily qualify as stop words.
   In NLP stop words are used to eliminate unimportant words, allowing applications to focus on the important words instead.

3. **PorterStemmer**: Stemming is the process of producing morphological variants of a root/base word. Stemming programs are commonly referred to as stemming algorithms or stemmers.
   A stemming algorithm reduces the words “chocolates”, “chocolatey”, and “choco” to the root word, “chocolate” and “retrieval”, “retrieved”, “retrieves” reduce to the stem “retrieve”.

4. **Regular Expressions:** A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern. RegEx can be used to check if a string contains the specified search pattern.
   Python has a built-in package called re, which can be used to work with Regular Expressions.

5. **TF IDF Vectorizer:** The TfidfVectorizer converts a collection of raw documents into a matrix of TF-IDF features. TF-IDF will transform the text into meaningful representation of integers or numbers which is used to fit machine learning algorithm for predictions.
   TF-IDF Vectorizer is a measure of originality of a word by comparing the number of times a word appears in document with the number of documents

6. **Decision Tree Classifier:** The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).

7. **Pickle:** Python's Pickle module is a popular format used to serialize and deserialize data types. This format is native to Python, meaning Pickle objects cannot be loaded using any other programming language. You can easily save different variables into a Pickle file and load them back in a different Python session, recovering your data exactly the way it was without having to edit your code.

Code:

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

In [2]: df=pd.read_csv("train.csv")

In [3]: df.head()
```

Out[3]:

	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 [4]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2007 entries, 0 to 2006
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   id      2007 non-null   int64 
 1   title   1955 non-null   object
 2   author  1798 non-null   object
 3   text    2003 non-null   object
 4   label   2007 non-null   int64 
dtypes: int64(2), object(3)
memory usage: 78.5+ KB
```

We just load the dataset. Check the top five rows and information about the dataset.

```
In [5]: df.isnull().sum()
```

```
Out[5]: id          0
        title      52
        author    209
        text        4
        label       0
        dtype: int64
```

```
In [6]: df=df.fillna('')

In [7]: df.isnull().sum()
```

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

```
In [8]: df.columns
```

```
Out[8]: Index(['id', 'title', 'author', 'text', 'label'], dtype='object')
```

```
In [9]: df=df.drop(['id', 'title', 'author'], axis=1)
```

We don't want any NULL values so we checked if there are any NULL values and we got many NULL, so we filled the null values with empty string.

After that we remove the ID, title and author column because we don't need it in our project.

```
In [10]: df.head()
```

Out[10]:

	text	label
0	House Dem Aide: We Didn't Even See Comey's Let...	1
1	Ever get the feeling your life circles the rou...	0
2	Why the Truth Might Get You Fired October 29, ...	1
3	Videos 15 Civilians Killed In Single US Airstr...	1
4	Print \nAn Iranian woman has been sentenced to...	1

```
In [11]: from nltk.corpus import stopwords

In [12]: from nltk.stem.porter import PorterStemmer

In [13]: import re

In [14]: port_stem=PorterStemmer()

In [15]: port_stem
Out[15]: <PorterStemmer>

In [16]: port_stem.stem('Hi my name    @#$%      is SADIQ #$%?')
Out[16]: 'hi my name    @#$%      is sadiq #$%?'
```

I imported the library that were needed. And created a object of PortStemmer class.

And test with a sample sentence.

```
In [17]: def stemming(content):
             con=re.sub('[^a-zA-Z]', ' ', content)
             con=con.lower()
             con=con.split()
             con=[port_stem.stem(word) for word in con if not word in stopwords.words('english')]
             con=' '.join(con)
             return con

In [18]: stemming('Hi my name    @#$%      is SADIQ #$%?')
Out[18]: 'hi name sadiq'

In [24]: df['text']= df['text'].apply(stemming)

In [25]: x=df['text']

In [26]: y=df['label']

In [27]: x.shape
Out[27]: (2007,)

In [28]: y.shape
Out[28]: (2007,)

In [29]: from sklearn.model_selection import train_test_split

In [30]: x_train , x_test , y_train, y_test = train_test_split(x, y, test_size=0.20)

In [31]: from sklearn.feature_extraction.text import TfidfVectorizer
```

I created a function name stemming which will remove all the regular expressions, convert upper case letter into lower case, remove stopwords and also removes blank space.

```
In [32]: vect=TfidfVectorizer()

In [33]: x_train=vect.fit_transform(x_train)
         x_test=vect.transform(x_test)

In [34]: x_train.shape
Out[34]: (1605, 29015)

In [35]: x_test.shape
Out[35]: (402, 29015)

In [36]: from sklearn.tree import DecisionTreeClassifier

In [37]: model=DecisionTreeClassifier()

In [38]: model.fit(x_train, y_train)
Out[38]: ▾ DecisionTreeClassifier
         DecisionTreeClassifier()

In [39]: prediction=model.predict(x_test)
```

We split the data into training and testing in 80 , 20 ratio. 80 for training and 20 for testing.

We used the decision tree as our algorithm in this project.

```
In [40]: prediction
Out[40]: array([0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0,
                0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0,
                0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0,
                1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1,
                0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0,
                1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1,
                0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0,
                0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0,
                1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1,
                1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0,
                0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1,
                1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1,
                0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1,
                1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0,
                1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
                0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0,
                0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1,
                1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
                1, 1, 0, 0, 0, 1], dtype=int64)

In [41]: model.score(x_test, y_test)
Out[41]: 0.8283582089552238

In [42]: import pickle

In [43]: pickle.dump(vect, open('vector.pkl', 'wb'))

In [44]: pickle.dump(model, open('model.pkl', 'wb'))
```

It gives us the accuracy of almost 83%.

We imported pickle library to store the data.

```
In [45]: vector_form=pickle.load(open('vector.pkl', 'rb'))

In [46]: load_model=pickle.load(open('model.pkl', 'rb'))

In [47]: def fake_news(news):
             news=stemming(news)
             input_data=[news]
             vector_form1=vector_form.transform(input_data)
             prediction = load_model.predict(vector_form1)
             return prediction
```

We created the fake news function which will detect the news is fake or not.

Output:

Unreliable in our dataset:

```
In [44]: val=fake_news(""" "House Dem Aide: We Didnâ€™t Even See Comeyâ€™s Letter Until Jason Chaffetz Tweeted It By Darrell Lucus on Octo
         With apologies to Keith Olbermann, there is no doubt who the Worst Person in The World is this weekâ€“FBI Director James Comey. E
         As we now know, Comey notified the Republican chairmen and Democratic ranking members of the House Intelligence, Judiciary, and O
         â€" Jason Chaffetz (@jasoninthehouse) October 28, 2016
         Of course, we now know that this was not the case . Comey was actually saying that it was reviewing the emails in light of â€œan
         But according to a senior House Democratic aide, misreading that letter may have been the least of Chaffetzâ€™ sins. That aide to
         So letâ€™s see if weâ€™ve got this right. The FBI director tells Chaffetz and other GOP committee chairmen about a major developm
         There has already been talk on Daily Kos that Comey himself provided advance notice of this letter to Chaffetz and other Republic
         What it does suggest, however, is that Chaffetz is acting in a way that makes Dan Burton and Darrell Issa look like models of res
         Granted, itâ€™s not likely that Chaffetz will have to answer for this. He sits in a ridiculously Republican district anchored in
         Darrell is a 30-something graduate of the University of North Carolina who considers himself a journalist of the old school. An a
         """)

In [91]: if val==[0]:
             print('reliable')
         else:
             print('unreliable')

         unreliable
```

Reliable in our dataset:

```
In [46]: val=fake_news(""" Ever get the feeling your life circles the roundabout rather than heads in a straight line toward the intended
         """)

In [47]: if val==[0]:
             print('reliable')
         else:
             print('unreliable')

         reliable

In [ ]:
```

If we want to check the live news. We will have to put the live news in the dataset first. Because if we don't put the live news into the dataset and directly feed to the program it won't work.

Because the live news didn't go through any process of stopwords removal, converting the upper-case letter into lower case, neither it goes through the process of porter Stemmer nor the decision tree classifier. So, it will not give the accurate result.

Once we put the news into the dataset. It will go through all the process and the result will be accurate.