Detecting Fake News with Python and Machine Learning

Do you trust all the news you hear from social media?

All news are not real, right?

How will you detect fake news?

The answer is Python. By practicing this advanced python project of detecting fake news, you will easily make a difference between real and fake news.

Before moving ahead in this machine learning project, get aware of the terms related to it like fake news, tfidfvectorizer, PassiveAggressive Classifier.

Also, I like to add that DataFlair has published a *series of machine learning Projects* **where you will get interesting and open-source advanced ml projects. Do check, and then share your experience through comments. Here is the list of top Python projects:**

1. **Fake News Detection Python Project**
2. **Parkinson's Disease Detection Python Project**
3. **Color Detection Python Project**
4. **Speech Emotion Recognition Python Project**
5. **Breast Cancer Classification Python Project**
6. **Age and Gender Detection Python Project**

7. Handwritten Digit Recognition Python Project
8. Chatbot Python Project
9. Driver Drowsiness Detection Python Project
10. Traffic Signs Recognition Python Project
11. Image Caption Generator Python Project

What is Fake News?

A type of yellow journalism, fake news encapsulates pieces of news that may be hoaxes and is generally spread through social media and other online media. This is often done to further or impose certain ideas and is often achieved with political agendas. Such news items may contain false and/or exaggerated claims, and may end up being viralized by algorithms, and users may end up in a filter bubble.

What is a TfidfVectorizer?

TF (Term Frequency): The number of times a word appears in a document is its Term Frequency. A higher value means a term appears more often than others, and so, the document is a good match when the term is part of the search terms.

IDF (Inverse Document Frequency): Words that occur many times a document, but also occur many times in many others, may be irrelevant. IDF is a measure of how significant a term is in the entire corpus.

The TfidfVectorizer converts a collection of raw documents into a matrix of TF-IDF features.

What is a PassiveAggressiveClassifier?

Passive Aggressive algorithms are online learning algorithms. Such an algorithm remains passive for a correct classification outcome, and turns aggressive in the event of a miscalculation, updating and adjusting. Unlike most other algorithms, it does not converge. Its purpose is to make updates that correct the loss, causing very little change in the norm of the weight vector.

Detecting Fake News with Python

To build a model to accurately classify a piece of news as REAL or FAKE.

About Detecting Fake News with Python

This advanced python project of detecting fake news deals with fake and real news. Using sklearn, we build a TfidfVectorizer on our dataset. Then, we initialize a PassiveAggressive Classifier and fit the model. In the end, the accuracy score and the confusion matrix tell us how well our model fares.

The fake news Dataset

The dataset we'll use for this python project- we'll call it news.csv. This dataset has a shape of 7796×4. The first column identifies the news, the second and third are the title and text, and the fourth column has labels denoting whether the news is REAL or FAKE. The dataset takes up 29.2MB of space and you can *download it here*.

Project Prerequisites

You'll need to install the following libraries with pip:

```
pip install numpy pandas sklearn
```

You'll need to install Jupyter Lab to run your code. Get to your command prompt and run the following command:

```
C:\Users\DataFlair>jupyter lab
```

You'll see a new browser window open up; create a new console and use it to run your code. To run multiple lines of code at once, press Shift+Enter.

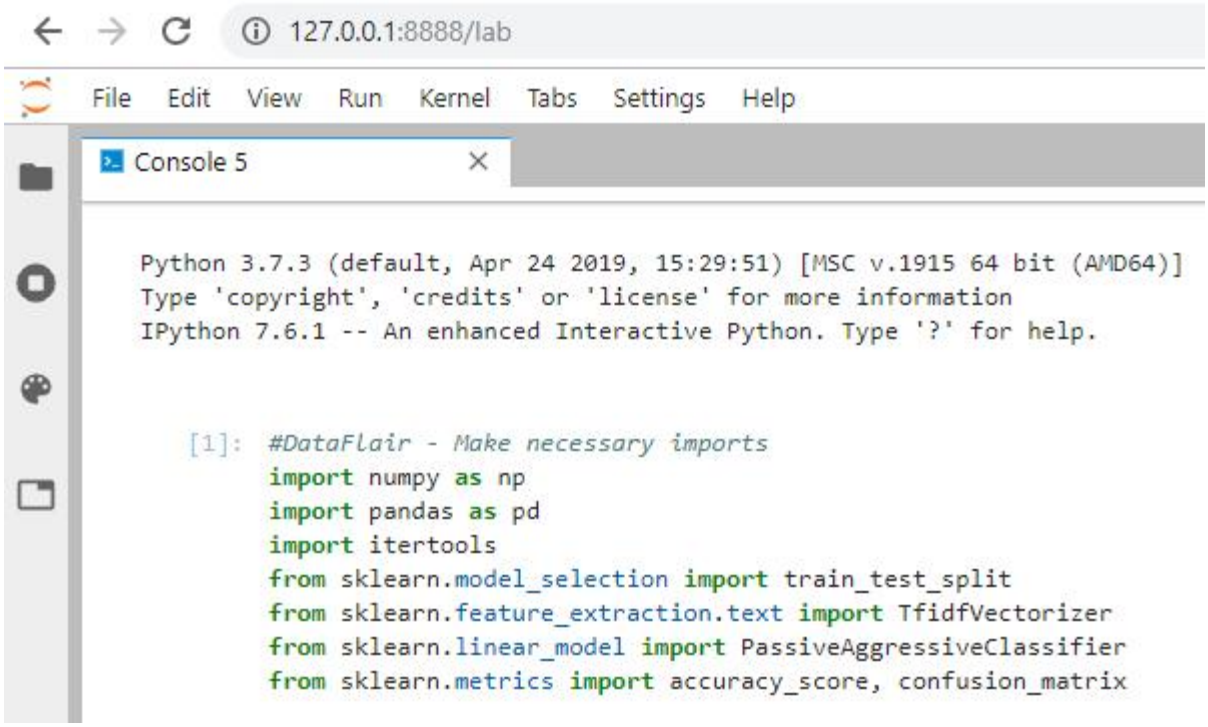
Steps for detecting fake news with Python

Follow the below steps for detecting fake news and complete your first advanced Python Project –

1. **Make necessary imports:**

```
import numpy as np

import pandas as pd

import itertools

from sklearn.model_selection import train_test_split

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.linear_model import PassiveAggressiveClassifier

from sklearn.metrics import accuracy_score, confusion_matrix
```

Screenshot:



2. Now, let's read the data into a DataFrame, and get the shape of the data and the first 5 records.

```
#Read the data

df=pd.read_csv('D:\\DataFlair\\news.csv')

#Get shape and head

df.shape

df.head()
```

Output Screenshot:

```
[2]: #Read the data
     df=pd.read_csv('D:\\DataFlair\\news.csv')

     #Get shape and head
     df.shape
     df.head()
```

[2]:

	Unnamed: 0	title	text	label
0	8476	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	875	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL

3. And get the labels from the DataFrame.

```
#DataFlair - Get the labels

labels=df.label

labels.head()
```

Output Screenshot:

```
[3]: #DataFlair - Get the labels
     labels=df.label
     labels.head()

[3]: 0    FAKE
     1    FAKE
     2    REAL
     3    FAKE
     4    REAL
     Name: label, dtype: object
```

4. Split the dataset into training and testing sets.

```
#DataFlair - Split the dataset

x_train,x_test,y_train,y_test=train_test_split(df['text'], labels,
test_size=0.2, random_state=7)
```

Screenshot:

```
[4]: #DataFlair - Split the dataset
     x_train,x_test,y_train,y_test=train_test_split(df['text'], labels, test_size=0.2, random_state=7)
```

5. Let's initialize a TfidfVectorizer **with stop words from the English language and a maximum document frequency of 0.7 (terms with a higher document frequency will be discarded). Stop words are the most common words in a language that are to be filtered out before processing the natural language data. And a TfidfVectorizer turns a collection of raw documents into a matrix of TF-IDF features.**

Now, fit and transform the vectorizer on the train set, and transform the vectorizer on the test set.

```
#DataFlair - Initialize a TfidfVectorizer

tfidf_vectorizer=TfidfVectorizer(stop_words='english', max_df=0.7)

#DataFlair - Fit and transform train set, transform test set

tfidf_train=tfidf_vectorizer.fit_transform(x_train)

tfidf_test=tfidf_vectorizer.transform(x_test)
```

Screenshot:

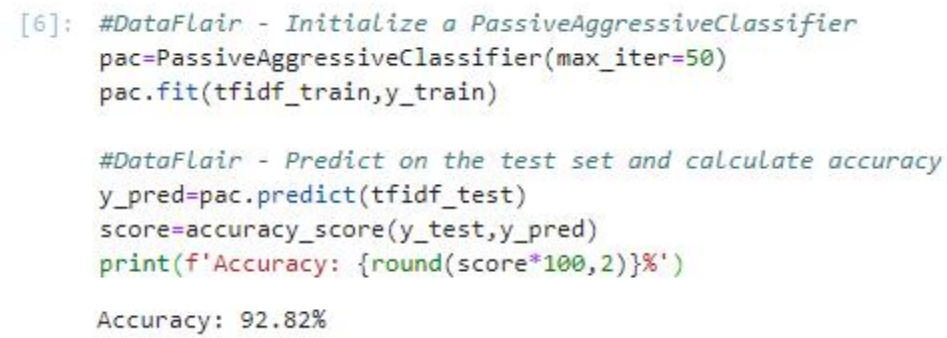
```
[5]: #DataFlair - Initialize a TfidfVectorizer
     tfidf_vectorizer=TfidfVectorizer(stop_words='english', max_df=0.7)

     #DataFlair - Fit and transform train set, transform test set
     tfidf_train=tfidf_vectorizer.fit_transform(x_train)
     tfidf_test=tfidf_vectorizer.transform(x_test)
```

6. Next, we'll initialize a PassiveAggressiveClassifier. This is. We'll fit this on tfidf_train and y_train.

Then, we'll predict on the test set **from the TfidfVectorizer and calculate the accuracy with accuracy_score() from sklearn.metrics.**

```
#DataFlair - Initialize a PassiveAggressiveClassifier

pac=PassiveAggressiveClassifier(max_iter=50)

pac.fit(tfidf_train,y_train)

#DataFlair - Predict on the test set and calculate accuracy

y_pred=pac.predict(tfidf_test)

score=accuracy_score(y_test,y_pred)

print(f'Accuracy: {round(score*100,2)}%')
```

Output Screenshot:



7. We got an accuracy of 92.82% with this model. Finally, let's print out a confusion matrix to gain insight into the number of false and true negatives and positives.

```
#DataFlair - Build confusion matrix

confusion_matrix(y_test,y_pred, labels=['FAKE','REAL'])
```

Output Screenshot:

```
[7]: #DataFlair - Build confusion matrix
     confusion_matrix(y_test,y_pred, labels=['FAKE','REAL'])

[7]: array([[589,  49],
            [ 42, 587]], dtype=int64)
```

```
[ ]:
```

So with this model, we have 589 true positives, 587 true negatives, 42 false positives, and 49 false negatives.

Summary

Today, we learned to detect fake news with Python. We took a political dataset, implemented a TfidfVectorizer, initialized a PassiveAggressiveClassifier, and fit our model. We ended up obtaining an accuracy of 92.82% in magnitude.

Hope you enjoyed the fake news detection python project. Keep visiting DataFlair for more interesting python, data science, and machine learning projects.

You give me 15 seconds I promise you best tutorials