Fake news on different platforms is spreading widely and is a matter of serious concern, as it causes social wars and permanent breakage of the bonds established among people. A lot of research is already going on focused on the classification of fake news.

.

**Steps to be followed**

1. Importing Libraries and Datasets.
2. Data Preprocessing.
3. Preprocessing and analysis of News column.
4. Converting text into Vectors.
5. Model training, Evaluation, and Prediction.

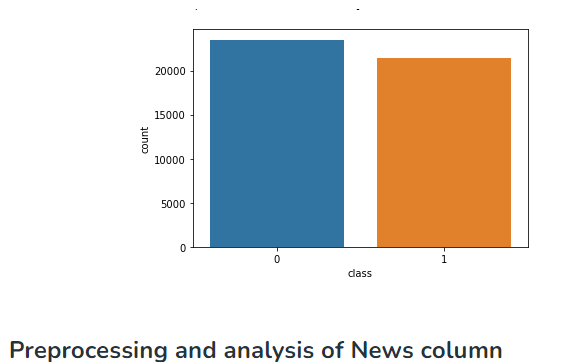
**Importing Libraries and Datasets**

The libraries used are :

Pandas: For importing dataset

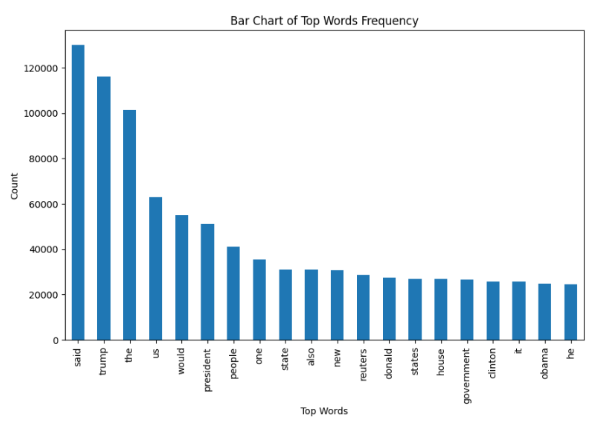
Seaborn/MatplotLib: For data visualization.

## Data preprocessing

The shape of the dataset can be found by the below code.

**Preprocessing and analysis of News column**

* Firstly we will remove all the stopwords, punctuations and any irrelevant spaces from the text. For that NLTK Library is required and some of it’s module need to be downloaded. So, for that run the below code.
* Once we have all the required modules, we can create a function name preprocess text. This function will preprocess all the data given as input.
* To implement the function in all the news in the text column, run the below command. Let’s visualize the WordCloud for fake and real news separately.



## Converting text into Vectors

Before converting the data into vectors, split it into train and test. Now we can convert the training data into vectors using TfidfVectorizer.

## Model training, Evaluation, and Prediction

Now, the dataset is ready to train the model.

For training we will use Logistic Regression and evaluate the prediction accuracy using accuracy\_score.

**CODE:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import matplotlib

import seaborn as sns

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

*#default theme*

plt.style.use('ggplot')

sns.color\_palette("tab10")

sns.set(context='notebook', style='darkgrid', font='sans-serif', font\_scale=1, rc=None)

matplotlib.rcParams['figure.figsize'] =[20,8]

matplotlib.rcParams.update({'font.size': 15})

matplotlib.rcParams['font.family'] = 'sans-serif'

*Read the data*

df=pd.read\_csv('../input/textdb3/fake\_or\_real\_news.csv')

*#Get shape and head*

print(df.shape)

df.head()

*#DataFlair - Get the labels*

labels=df.label

labels.head()

target=df.label.value\_counts()

target

sns.countplot(df.label)

plt.title('the number of news fake/real);

*#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)

*#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)

*#DataFlair - Initialize a PassiveAggressiveClassifier*

pac=PassiveAggressiveClassifier(max\_iter=50)

pac.fit(tfidf\_train,y\_train)

*#DataFlair - Predict on the test set and calculate accuracy*

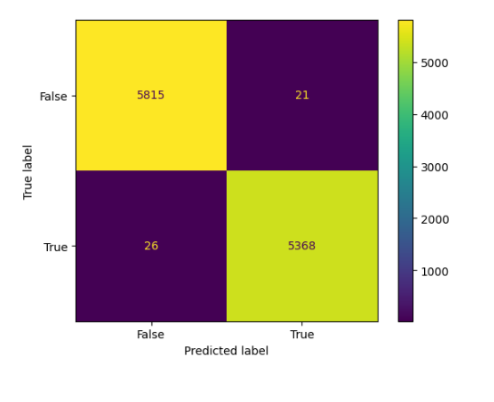
y\_pred=pac.predict(tfidf\_test)

score=accuracy\_score(y\_test,y\_pred)

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

*#DataFlair - Build confusion matrix*

confusion\_matrix(y\_test,y\_pred, labels=['FAKE','REAL'])

**OUTPUT:**