**FAKE NEWS DETECTION USING NLP IN ARTIFICIAL INTELLIGENCE**

**TEAM MEMBER**

**820421205045:NILOFERNISHA N**

**Phase-3 SUBMISSION DOCUMENT**

**Project: Fake News Detection**

**Phase 3: *Development Part 1***

**TOPIC:**Begin building the fake news detection model by loading and preprocessing the dataset. Load the fake news dataset and preprocess the textual data.

****

**Fake News Detection**

**ABSTRACT**

• The proliferation of fake news in the digital age has become a significant challenge in maintaining the integrity of information dissemination.

• To address this issue, the development of a fake news detection model is essential.

• This abstract outlines the initial steps involved in building such a model, focusing on the critical processes of loading and preprocessing the dataset.

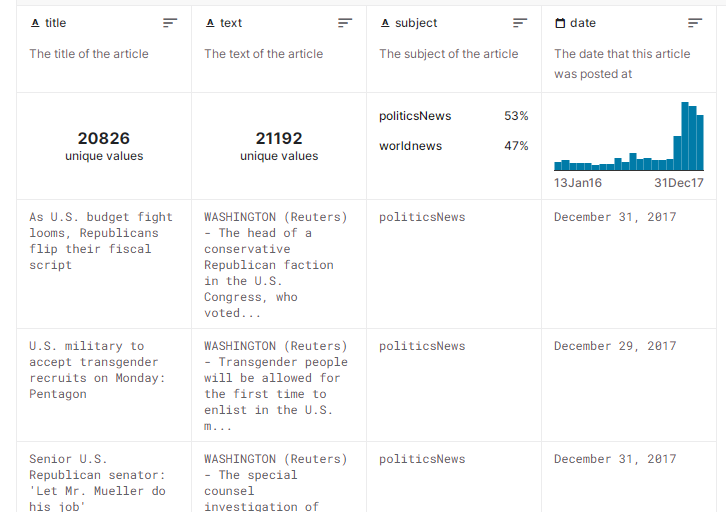
**INTRODUCTION**

Building a fake news detection model involves several steps, with one of the initial steps being loading and preprocessing the dataset. In this example, I'll provide a general outline of how to load and preprocess a fake news dataset using Python. For this purpose, we'll use Python and popular libraries like Pandas and NLTK. Please note that you'll need to have your dataset in a format suitable for this process.

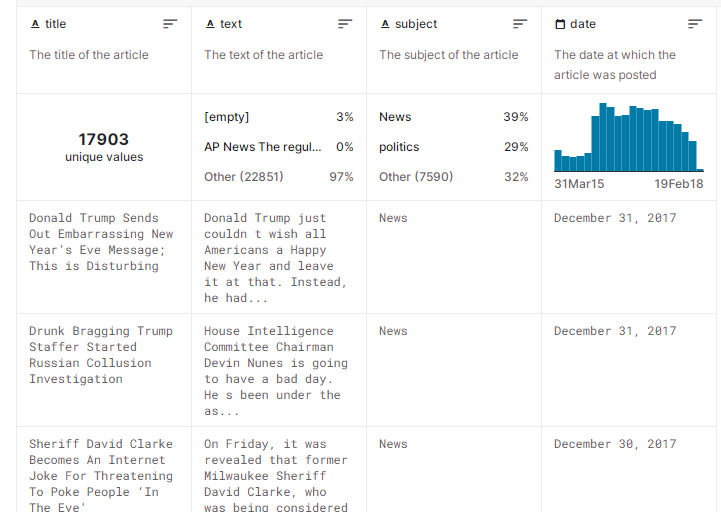
**GIVEN DATASET**

**Dataset Link:**[**https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset**](https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset)

**real.csv**

****

**fake.csv**

****

**DATA DESCRIPTION:**

Title: the title of a news article

Text: the text of the article; could be incomplete

Subject: display the field of the news

Date: publish date

**NECESSARY STEPS TO FOLLOW**

**IMPORTING LIBRARIES**

Before begin working with the dataset,import essential Python libraries,such as pandas for data manipulation,nltkor spaCy for text preprocessing, and sklearn for machine learning.

**PROGRAM**

importnumpy as np

import pandas as Pd

importmatplotlib.pyplotasplt

importseabornassns

fromsklearn.feature\_extraction.text import TfidfVectorizer

fromsklearn.model\_selection import train\_test\_split

fromsklearn.metrics import accuracy\_score

fromsklearn.metrics import classification\_report

fromsklearn.metrics

importconfusion\_matrix,ConfusionMatrixDisplay

import re

import string

**SPLIT THE DATA**

Make separate training and testing sets from your dataset. This aids in the performance evaluation of your model afterwards.

**PROGRAM**

fromsklearn.model\_selection import train\_test\_split

**FEATURE SCALING**

Apply feature scaling to normalize your data, ensuring that all features have similar scales. Standardization (scaling to mean=0 andstd=1) is a common choice.

**PROGRAM**

fromsklearn.feature\_extraction.text import TfidfVectorizer

**LOADING THE DATASET**

Using a library like pandas to load the chosen dataset into a DataFrame . This allows us to easily manipulate and analyze the data.

**PROGRAM**

true\_data=pd.read\_csv("C:\\Users\\gokul\\OneDrive\\Desktop\\machine\_learning\\projects\\True.csv")

fake\_data=pd.read\_csv("C:\\Users\\gokul\\OneDrive\\Desktop\\machine\_learning\\projects\\Fake.csv")

**DATA PREPROCESSING**

Preprocessing steps are essential in fake news detection using Natural Language Processing (NLP). These steps help clean and prepare the text data for analysis, making it easier for machine learning models to identify patterns and features that distinguish fake news from real news.

**ADDING TARGET ATTRIBUTE TO DATASET**

**PROGRAM**

true\_data['class']=1

fake\_data['class']=0

**CONCATENATION OF TRUE AND FAKE DATASET**

**PROGRAM**

data=pd.concat([true\_data,fake\_data],axis=0)

**DATA EXPLORATION**

checking for missing values, exploring the data's statistics, andvisualizing it to identify patterns.

**PROGRAM**

**Removing unwanted columns**

data.drop(['title','subject','date'],axis=1,inplace=True)

Removing NULL values:

data.isnull().sum()

**RANDOM SHUFFLING THE DATAFRAME**

To make sure that the data order does not induce biases throughout the model training phase, shuffle the dataset at random.

**PROGRAM**

data=data.sample(frac=1)

**Tokenization and Text Preprocessing**

Tokenize the text data and apply common preprocessing steps (e.g., lowercasing, stop word removal, stemming) to prepare the text for analysis. Tokenization involves breaking the text into individual words or tokens, which is crucial for NLP analysis.

**PROGRAM**

defwordopt(text):

text = text.lower()

text = re.sub('\[.\*?\]', '', text)

text = re.sub("\\W"," ",text)

text = re.sub('https?://\S+|www\.\S+', '', text)

text = re.sub('<.\*?>+', '', text)

text = re.sub('[%s]' % re.escape(string.punctuation), '', text)

text = re.sub('\n', '', text)

text = re.sub('\w\*\d\w\*', '', text)

return text

data['text']=data['text'].apply(wordopt)

**SPLITTING OF DATA**

x=data['text']

y=data['class']

TEXT TO VECTOR:

tfv=TfidfVectorizer()

x=tfv.fit\_transform(x)

SPLITTING DATA TO TRAIN AND TEST DATA

X\_train,X\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.20)

**PROGRAM**

importnumpy as np

import pandas as pd

importmatplotlib.pyplot as plt

importseaborn as sns

fromsklearn.model\_selection import train\_test\_split

fromsklearn.feature\_extraction.text import TfidfVectorizer

fromsklearn.metrics import accuracy\_score

fromsklearn.metrics import classification\_report

fromsklearn.metrics import confusion\_matrix,ConfusionMatrixDisplay

import re

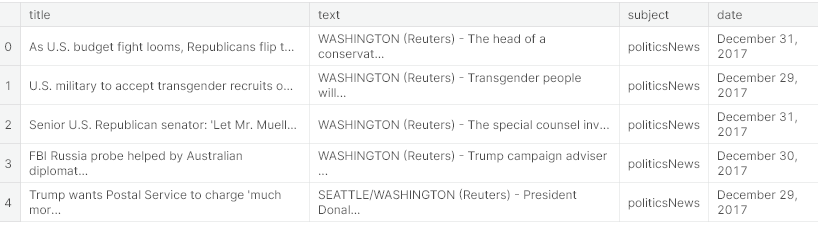
import string

true\_data=pd.read\_csv("C:\\Users\\gokul\\OneDrive\\Desktop\\machine\_learning\\projects\\True.csv") #reading true news dataset

fake\_data=pd.read\_csv("C:\\Users\\gokul\\OneDrive\\Desktop\\machine\_learning\\projects\\Fake.csv") #reading fake news dataset

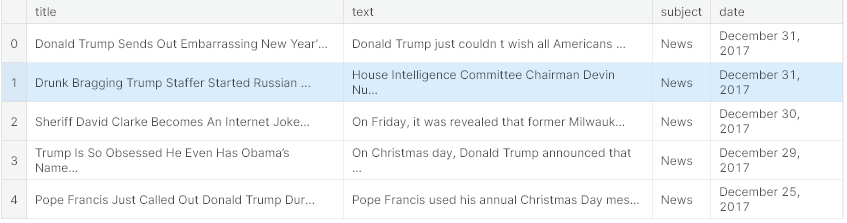
print(true\_data.head())

**OUTPUT**



print(fake\_data.head())

**OUTPUT**

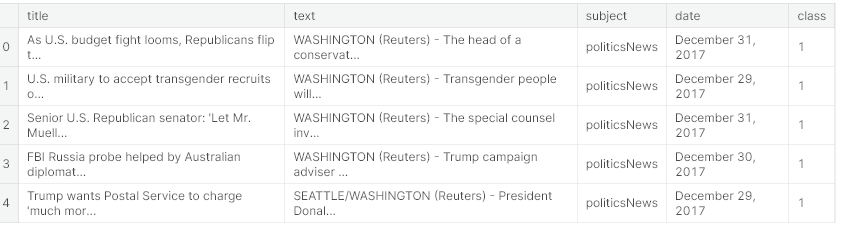
****

true\_data['class']=1

fake\_data['class']=0

print(true\_data.head())

**OUTPUT**



print(true\_data.shape , fake\_data.shape)

**OUTPUT**

((21417, 5), (23481, 5))

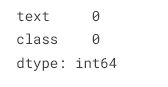
**CONCATINATING FAKE AND REAL DATASET**

data=pd.concat([true\_data,fake\_data],axis=0)

data.drop(['title','subject','date'],axis=1,inplace=True)

print(data.isnull().sum())

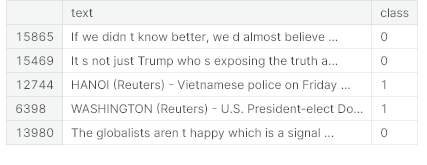
**OUTPUT**



data=data.sample(frac=1)

Print(data.head())

**OUTPUT**



defwordopt(text):

text=text.lower()

text=re.sub('\[.\*?\]', '', text)

text=re.sub("**\\**W"," ",text)

text=re.sub('https?://\S+|www\.\S+', '', text)

text=re.sub('<.\*?>+', '', text)

text=re.sub('[**%s**]'%re.escape(string.punctuation), '', text)

text=re.sub('**\n**', '', text)

text=re.sub('\w\*\d\w\*', '', text)

returntext

data['text']=data['text'].apply(wordopt)

x=data['text']

y=data['class']

tfv=TfidfVectorizer()

x=tfv.fit\_transform(x)

X\_train,X\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.20)

fromsklearn.linear\_model import LogisticRegression

lr\_model=LogisticRegression()

lr\_model.fit(X\_train,y\_train)

**OUTPUT**

LogisticRegression()

y\_pred\_lr=lr\_model.predict(X\_test)

y\_pred\_lr

**OUTPUT**

array([1, 1, 0, ..., 1, 0, 0], dtype=int64)

accuracy\_score(y\_pred\_lr,y\_test)

**OUTPUT**

0.9870824053452116

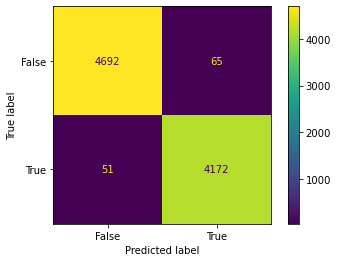
cm = confusion\_matrix(y\_test, y\_pred\_lr)

cm\_display = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=[False, True])

cm\_display.plot()

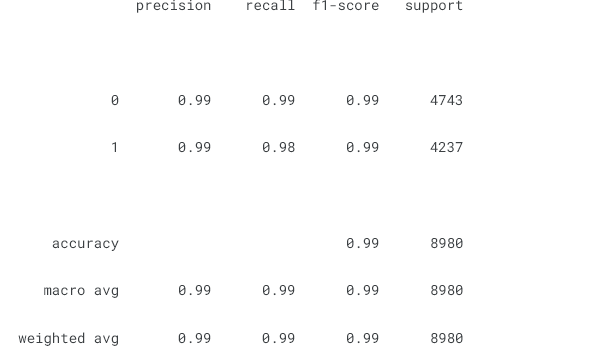
plt.show()

**OUTPUT**



print(classification\_report(y\_pred\_lr,y\_test))

**OUTPUT**



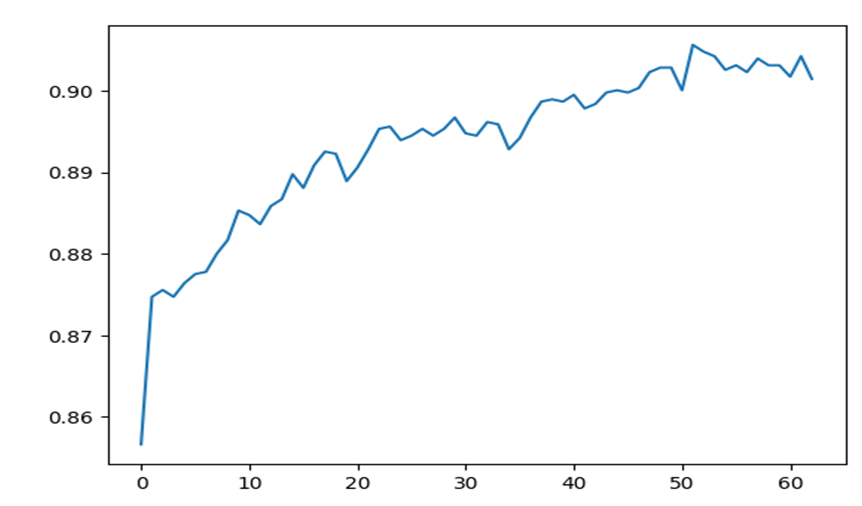
**DATA VISUALIZATION**

import matplotlib.pyplot as plt

plt.plot(clf.validation\_scores\_)

**OUTPUT**

[<matplotlib.lines.Line2D at 0x7aa26f0901f0>]

****

**CONCLUSION:**

Fake news detection using NLP is a pivotal application of technology in safeguarding the reliability of information in the digital era. By leveraging natural language processing and machine learning, we can develop effective tools to distinguish between genuine and deceptive content.