**FAKE NEWS DETECTION USING NLP**

**Topic :** In this part you will begin building your project by loading and preprocessing the dataset. Begin building the fake news detection model by loading and preprocessing the dataset. Load the fake news dataset and preprocess the textual data.



**Introduction :**

Fake news detection is the process of identifying and verifying the accuracy of news or information that is intentionally false, misleading, or fabricated. It has become a critical concern in today’s digital age, where misinformation can spread rapidly through various media channels. Here’s an introduction to the topic.

**Definition of Fake News:** Fake news encompasses various types of misinformation, including fabricated stories, manipulated images or videos, and misleading headlines. It can be spread through websites, social media, or traditional media outlets.

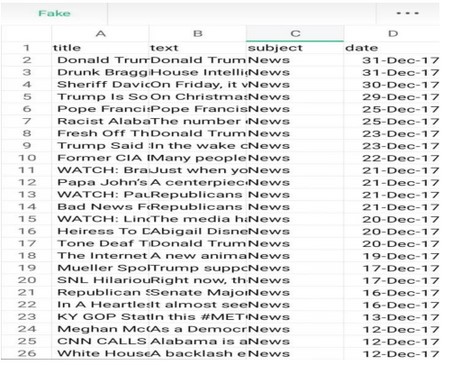
**Motivations for Fake News:** Fake news can be created for various reasons, such as political manipulation, financial gain, or simply for entertainment. It often seeks to exploit emotions, biases, or controversy to gain attention and traction.

**Impact of Fake News:** Fake news can have serious consequences, including influencing public opinion, swaying elections, causing panic, or harming individuals’ reputations. It can erode trust in journalism and democratic processes.

**Challenges in Fake News Detection**: Detecting fake news is a complex task due to its constantly evolving nature. Some challenges include the speed at which fake news spreads, the use of sophisticated techniques to make it appear legitimate, and the fine line between satire and actual misinformation.

**Given data set :**

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



23503 Rows X 5 columns ( False Dat )

21418 Rows X 5 columns ( True Data)

size of the words represents their frequency. For plotting [word cloud](https://pypi.org/project/wordcloud/) we have used word cloud python library.



# Text pre-processing

After analyzing the data, we move towards text pre-processing before building machine learning models. The text pre-processing consists of the following steps:

Step 1: Lower casing

Step 2: Stop word removal

Step 3: Special character removal

# Train Test Split

In this step, we split the data into train and test set in the ratio of 75:25 i.e., 75% of the data used in training the model and rest 25% used for testing the model. The code for splitting data is shown below.

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y,test\_size=0.25, random\_state=0)

**Necessary steps to follow :**

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

# 1.Importing Libraries and Datasets

The libraries used are :

[Pandas:](https://www.geeksforgeeks.org/python-pandas-dataframe/) For importing the dataset.

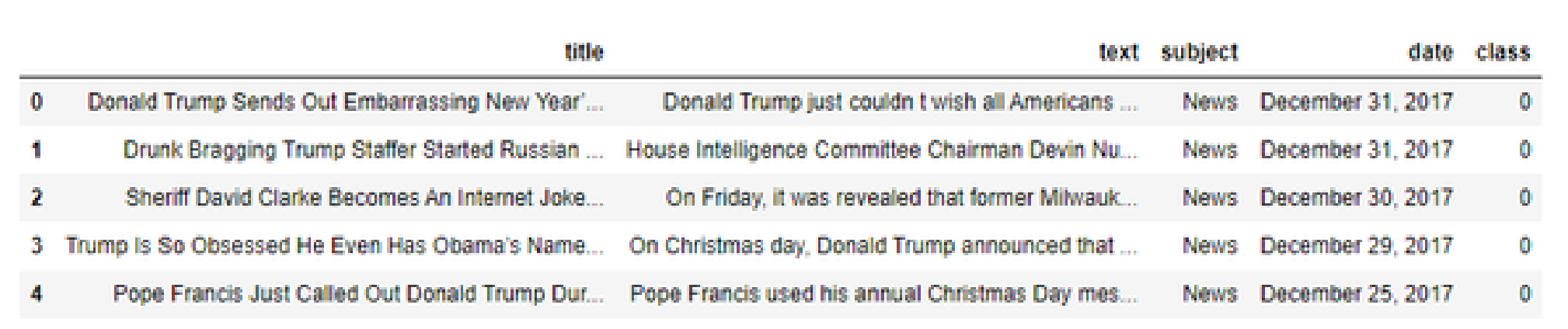
[Seaborn](https://www.geeksforgeeks.org/introduction-to-seaborn-python/)/Matplotlib: For data visualization.

**import** pandas as pd **import** seaborn as sns **import** matplotlib.pyplot as plt

Let’s import the downloaded dataset.

data = pd.read\_csv('News.csv',index\_col=0) data.head()

**OUTPUT:**



# 2.Data preprocessing

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

data.shape

OUTPUT:

(44919, 5)

As the title, subject and date column will not going to be helpful in identification of the news. So, we can drop these column.

data **=** data.drop(["title", "subject","date"], axis **=** 1)

Now, we have to check if there is any null value (we will drop those rows)

data.isnull().sum()

Output:

text 0 class 0

So there is no null value.

Now we have to shuffle the dataset to prevent the model to get bias. After that we will reset the index and then drop it. Because index column is not useful to us.

# Shuffling data **=** data.sample(frac**=**1) data.reset\_index(inplace**=**True) data.drop(["index"], axis**=**1, inplace**=**True)

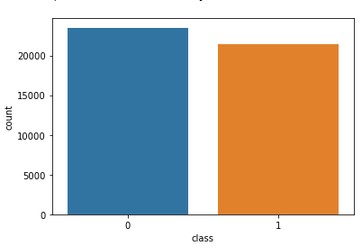
Now Let’s explore the unique values in the each category using below code.

sns.countplot(data**=**data,

x**=**'class',

order**=**data['class'].value\_counts().index)

Output:



**3.Preprocessing and analysis of News column:**

Firstly we will remove all the stopwords, punctuations and any irrelevant spaces from the text.

For that [NLTK](https://www.geeksforgeeks.org/tokenize-text-using-nltk-python/) Library is required and some of it’s module need to be downloaded.

**from** tqdm **import** tqdm **import** re **import** nltk nltk.download('punkt') nltk.download('stopwords') **from** nltk.corpus **import** stopwords **from** nltk.tokenize **import** word\_tokenize **from** nltk.stem.porter **import** PorterStemmer **from** wordcloud **import** WordCloud

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.

|  |
| --- |
| **def** preprocess\_text(text\_data): |
| preprocessed\_text  **=**    []      **for**    sentence  **in**    tqdm(text\_data):  sentence  **=**    re.sub(r'[^\w\s]', '', sentence)  preprocessed\_text.append(' '.join(token.lower()    **for**    token  **in**    str(sentence).split()    **if**    token  **not**    **in**    stopwords.words('english')))    **return**    preprocessed\_text |

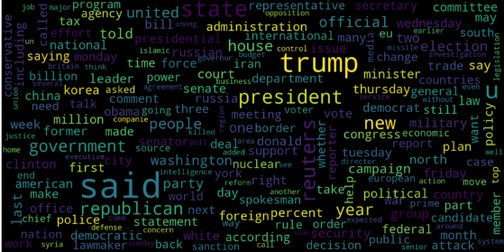
To implement the function in all the news in the text column, run the below command.

preprocessed\_review **=** preprocess\_text(data['text'].values) data['text'] **=** preprocessed\_review

Let’s visualize the WordCloud for fake and real news separately.

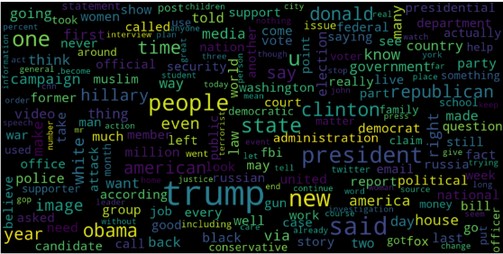
|  |
| --- |
| # Real consolidated **=** ' '.join(  word **for** word **in** data['text'][data['class'] **==** 1].astype(str)) wordCloud **=** WordCloud(width**=**1600,  height**=**800,  random\_state**=**21,  max\_font\_size**=**110, collocations**=**False) plt.figure(figsize**=**(15, 10)) plt.imshow(wordCloud.generate(consolidated), interpolation**=**'bilinear')  plt.axis('off') plt.show() |

**Output :**



|  |
| --- |
| Fake consolidated **=** ' '.join(  word **for** word **in** data['text'][data['class'] **==** 0].astype(str)) wordCloud **=** WordCloud(width**=**1600,  height**=**800,  random\_state**=**21,  max\_font\_size**=**110, collocations**=**False) plt.figure(figsize**=**(15, 10)) plt.imshow(wordCloud.generate(consolidated), interpolation**=**'bilinear')  plt.axis('off') plt.show() |

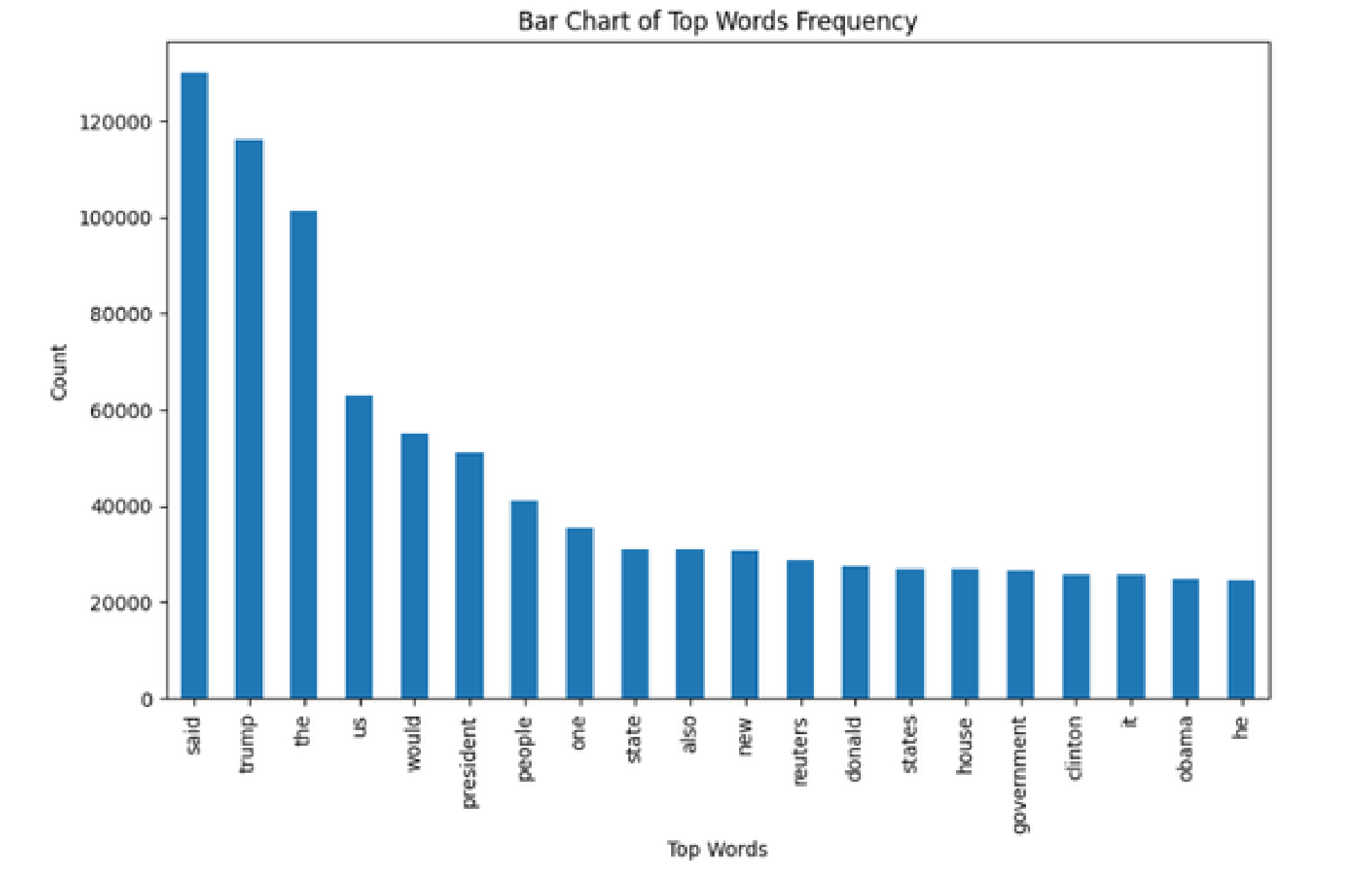
Output :



Now, Let’s plot the bargraph of the top 20 most frequent words.

|  |
| --- |
| **from** sklearn.feature\_extraction.text **import** CountVectorizer **def** get\_top\_n\_words(corpus, n**=**None):  vec **=** CountVectorizer().fit(corpus)  bag\_of\_words **=** vec.transform(corpus)  sum\_words **=** bag\_of\_words.sum(axis**=**0)  words\_freq **=** [(word, sum\_words[0, idx])  **for** word, idx **in** vec.vocabulary\_.items()]  words\_freq **=** sorted(words\_freq, key**=lambda** x: x[1],  reverse**=**True)  **return** words\_freq[:n]      common\_words **=** get\_top\_n\_words(data['text'], 20) df1 **=** pd.DataFrame(common\_words, columns**=**['Review', 'count'])  df1.groupby('Review').sum()['count'].sort\_values(ascending**=**False).plot(  kind**=**'bar',  figsize**=**(10, 6),  xlabel**=**"Top Words",  ylabel**=**"Count",  title**=**"Bar Chart of Top Words Frequency"  ) |

Output :



**4.Converting text into Vectors:**

Before converting the data into vectors, split it into train and test.

|  |
| --- |
| **from** sklearn.model\_selection **import** train\_test\_split **from** sklearn.metrics **import** accuracy\_score **from** sklearn.linear\_model **import** LogisticRegression    x\_train, x\_test, y\_train, y\_test **=** train\_test\_split(data['text'], data['class'], test\_size**=**0.25) |

Now we can convert the training data into vectors using TfidfVectorizer.

**from** sklearn.feature\_extraction.text **import** TfidfVectorizer vectorization **=** TfidfVectorizer() x\_train **=** vectorization.fit\_transform(x\_train) x\_test **=** vectorization.transform(x\_test)

**5.Model training, Evaluation, and Prediction:**

Now, the dataset is ready to train the model.

For training we will use [Logistic Regression](https://www.geeksforgeeks.org/understanding-logistic-regression/) and evaluate the prediction accuracy using accuracy\_score.

|  |
| --- |
| **from** sklearn.linear\_model **import** LogisticRegression    model **=** LogisticRegression() model.fit(x\_train, y\_train)    # testing the model print(accuracy\_score(y\_train, model.predict(x\_train))) print(accuracy\_score(y\_test, model.predict(x\_test))) |

Output :

0.993766511324171

0.9893143365983972

Let’s train with [Decision Tree](https://www.geeksforgeeks.org/decision-tree/) Classifier.

|  |
| --- |
| **from** sklearn.tree **import** DecisionTreeClassifier    model **=** DecisionTreeClassifier() model.fit(x\_train, y\_train)    # testing the model print(accuracy\_score(y\_train, model.predict(x\_train))) print(accuracy\_score(y\_test, model.predict(x\_test))) |

**Output :**

0.9999703167205913

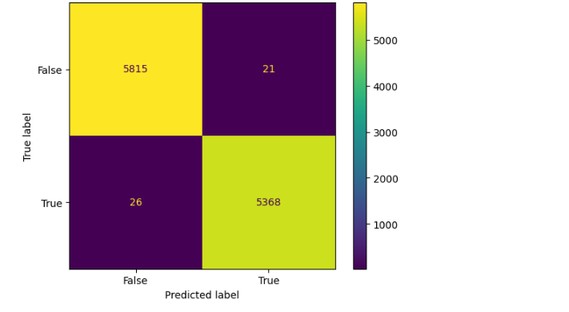
0.0.9999703167205913

0.9951914514692787

The confusion matrix for Decision Tree Classifier can be implemented with the code below.

|  |
| --- |
| # Confusion matrix of Results from Decision Tree classification **from** sklearn **import** metrics cm **=** metrics.confusion\_matrix(y\_test, model.predict(x\_test))    cm\_display **=** metrics.ConfusionMatrixDisplay(confusion\_matrix**=**cm,  display\_labels**=**[False, True])    cm\_display.plot() plt.show() |

**Output :**



PROGRAM:

FAKE NEWS DETECTION

IMPORT LIBRARIES:

Intl]:

Import numpy as np

Import pandas as pd

Import matplotlib.pyplot as plt

Import seaborn as sns

Import nltk

Import re

Import string

From sklearn.model selection import train test\_split

From sklearn.metrics import classification report

Import keras

From keras.preprocessing import text,sequence

From keras.models import Sequential

From keras.layers import Dense,Embedding,LSTM,Dropout

Import warnings

Warnings.filterwarnings('ignore')

Import os

For dirname, , filenames in os.walk('/kaggle/input'):

For filename in filenames:

Print(os.path.join(dirname, filename))

I OAD AND CHFCK DATA:

ln[2]:

Real data — pd.read csv('/kaggle/input/fake-and-real-news-datasetñrue.csv') Fake data = pd.read\_csv('/kaggle/inpuUfake-and-real-news-dataset/Fake.csv')

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Donald Trump | text  Donald  Trump just | subject | date |
|  | Sends Out | couldn t | News | December |
|  | Embarrassing New Year'...  Drunk | wish all Americans  House |  | 31, 2017 |
|  | Bragging | Intelligence |  | December |
| 1 | Trump Staffer  Started  Russian ...  Sheriff David | Committee Chairman  Devin Nu...  On Friday, it | News | 31, 2017 |
|  | Clarke | was |  | December |
| 2 | Becomes An Internet Joke...  Trump Is So  Obsessed He | revealed that former Milwauk...  On  Christmas | News | 30, 2017 |
| 3 | Even Has | day, Donald | News | December |
|  | Obama's Name...  Pope Francis | Trump announced that .  Pope  Francis used |  | 29, 2017 |
| 4 | Just Called | his annual | News | December |
|  | Out Donald  Trump Dur... | Christmas  Day mes... |  | 25, 2017 |

InL51: real\_data['target'] = 1 fake\_data['target'] = O

ln[6]:

 real data.tail()

|  |  |  |  |
| --- | --- | --- | --- |
|  | As U.S. | text | subject |
|  | budget | WASHINGTON |  |
| O | fight looms, | (Reuters) - The head of | politicsNews |
|  | Republicans flip t... | a conservat... |  |
|  | U.S. military | WASHINGTON |  |
| 1 | to accept | (Reuters) - Transgender | politicsNews |
|  | transgender recruits o...  Senior U.S. | people will... |  |
|  | Republican | WASHINGTON |  |
| 2 | senator: | (Reuters) - The special | politicsNews |
|  | 'Let Mr. Muell...  FBI Russia | counsel inv... |  |
|  | probe | WASHINGTON |  |
| 3 | helped by | (Reuters) - Trump | politicsNews |
|  | Australian diplomat...  Trump wants | campaign adviser ... |  |
|  | Postal | SEATTLE/WASHINGTON |  |
| 4 | Service to | (Reuters) - President | politicsNews |
|  | charge  'much | Donal... |  |

mor...

ln[3]: real\_data.head

text subject date

'Fully BRUSSELS committed' (Reuters) -

August

NATO NATO

21412 worldnews 22, backs new allies on

2017 U.S. Tuesday approach... we...

LexisNexis

# LONDON

withdrew

(Reuters) - August

two

1. LexisNexis, worldnews 22,

products

a provider 2017 from of l... Chinese

Minsk

MINSK

cultural

(Reuters) -

hub August

In the

1. becomes worldnews 22,

shadow of

haven 2017 disused from

Sov... authorities

Vatican MOSCOW upbeat on (Reuters) - August

1. possibility Vatican worldnews 22, of Pope Secretary 2017

Francis ... of State ...

Indonesia

JAKARTA to buy

$1.14 (Reuters) - August

1. Indonesia worldnews 22,

billion

will buy 1 1 2017 worth of Sukh... Russia...

Data = pd.concat([real data, fake\_data], ignore\_index=True, sort=False) Data.head()

|  |  |  |  |
| --- | --- | --- | --- |
|  | As U.S. | text | subject |
|  | budget | WASHINGTON |  |
| 0 | fight looms, | (Reuters) - The head of | politicsNews |
|  | Republicans flip t... | a conservat... |  |
|  | U.S. military to accept | WASHINGTON |  |
|  | transgender | (Reuters) - Transgender | politicsNews |
|  | recruits o...  Senior U.S. | people will... |  |
|  | Republican | WASHINGTON |  |
| 2 | senator: | (Reuters) - The special | politicsNews |
|  | 'Let Mr. Muell...  FBI Russia | counsel inv... |  |
|  | probe | WASHINGTON |  |
| 3 | helped by | (Reuters) - Trump | politicsNews |
|  | Australian diplomat...  Trump wants | campaign adviser ... |  |
|  | Postal | SEATTLE/WASHINGTON |  |
| 4 | Service to | (Reuters) - President | politicsNews |
|  | charge  'much | Donal... |  |

mor...

InL8J-: data.isnull().sum()

Title O

Text O

Subject 0

Date O

Target 0

Dtype: int64

VISUALIZATION

Count of Fake and Real Data

print(data["target"].value\_counts()) fig, ax = plt.subplots(1,2, figsize=(19, 5))

Gl = sns.countplot(data.target,ax=ax[0],palette="pastel");

Gl.set\_title("Count of real and fake data")

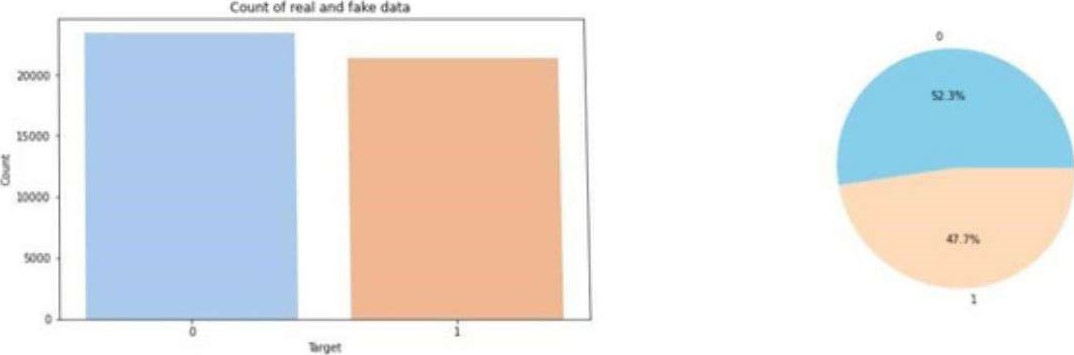
Gl.set\_ylabel("Count")

Gl.set\_xlabel("Target")

plt. pie(d values , explode: [0 , O] , labels=data.target. inde x, autopct='%l. SkyBlue' , 'PeachPuff']) fig .show()

0 1 21417

Name: target, dtype: int64





print(data.subject.value counts()) plt.figure(figsize=(10, 5))

ax = sns.countplot(x="subject", hue='target', data=data, palette="pastel") plt.title("Distribution of The

Subject According to Real and Fake Data")

politicsNews 11272 worldnews 10145 News 9050

Politics 6841 Left-news 4459

Government News 1570

US News 783

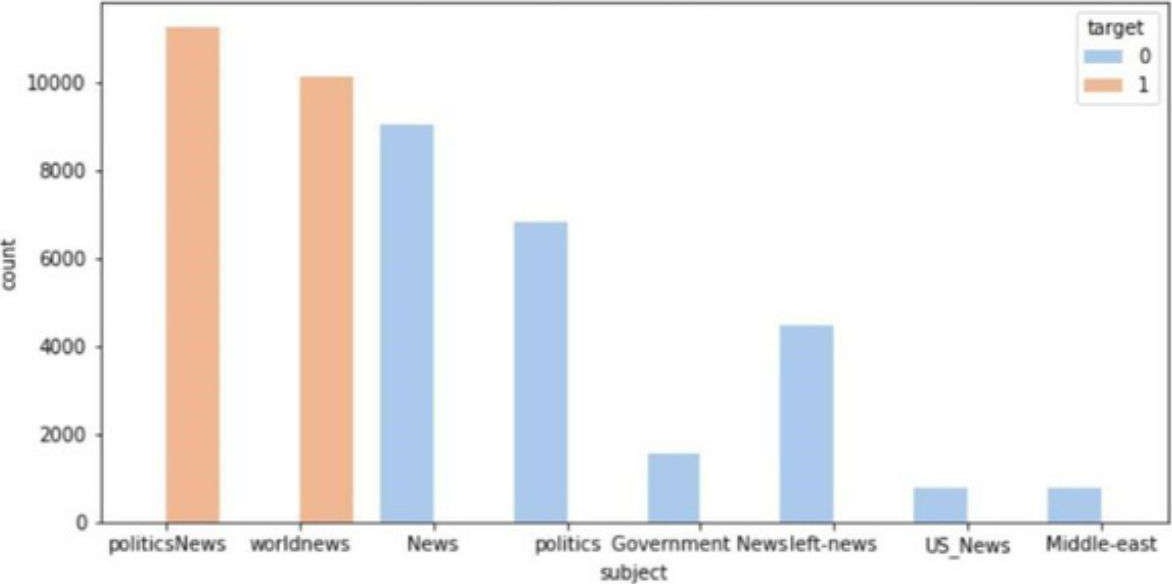
Middle-east 778

Name: subject, dtype: int64



Text(0.5, 1.0, 'Distribution of The Subject According to Real and Fake Data')

Dtstnbution of The Subject According to Real and Fake Data



DATA CLEANING





data['text']= data['subject'] + " " + data['title'] + + data['text'] del data['title'] del data['subject'] del data['date'] data.

text target

politicsNews As U.S. budget fight looms, Repub...

1 politicsNews U.S. military to accept transgend...

politicsNews Senior U.S. Republican senator:

2

politicsNews FBI Russia probe helped by 3

Austra...

4 politicsNews Trump wants Postal Service to cha...

Int[12]:

from wordcloud import WordCloud,STOPWORDS

plt.figure(figsize = (15, 15))

Wc = WordCloud(max\_words = 500 , width = 1000 , height = 500 , stopwords =

STOPWORDS).generate(" ".join(data[data.target == l].text))

Plt.imshow(wc , interpolation = 'bilinear')

|  |
| --- |
| outli |
|  |

<matplotlib.image.Axeslmage at Ox7f6934fd2750>

whiteOYhousesay 1 obamang

¥onep preres s; dent bar1 don ack t- doha I d country r:tuesqay - including new york support

Ounited state

governmentnorth korea t hur sday said

ime prime minister-

# do na

## -3 official said islamic è2FèuterŠÈstate people

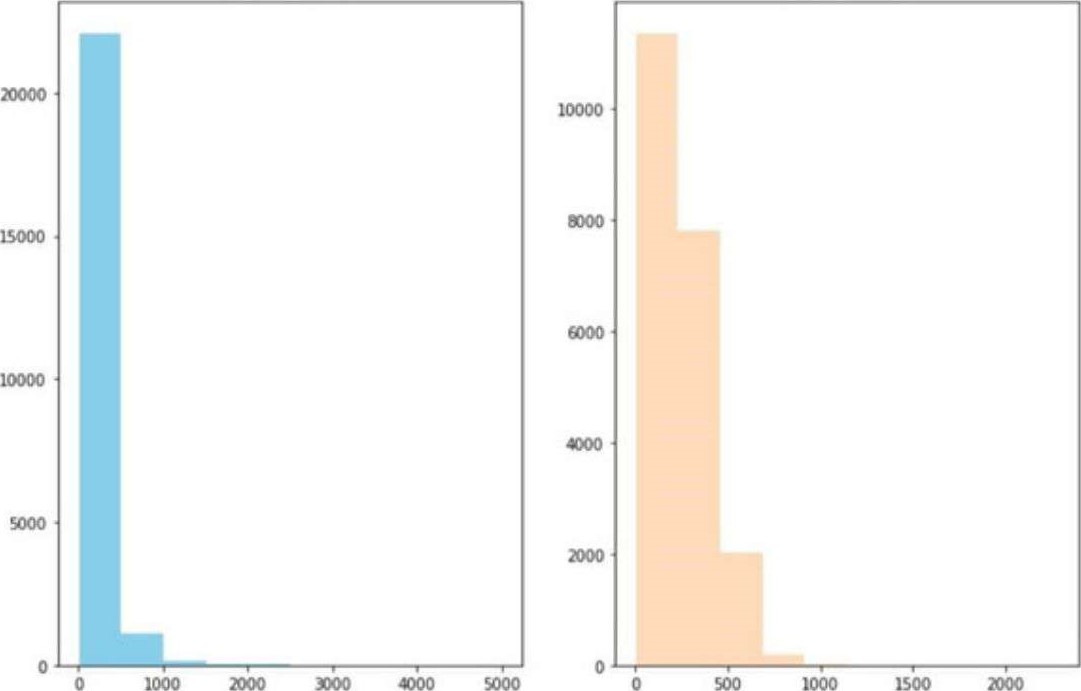
Number of words in each text

fig,(ax1,ax2)=plt.subplots(1,2,figsize=(12,8)) text len=data[data['target']==0]

['text'].str.split().map(lambda x: len(x)) axl.hist(text\_len,color='SkyBlue') axl.set\_title('Fake news text')  x: len(x)) ax2.hist(text len,color='PeachPuff') ax2.set\_title('Real news text') fig.suptitle(Words in texts') plt.show()

Words in texts

Fake news text Real news text



The number of words seems to be a bit different. 500 words are most common in real news category while around 250 words are most common in fake news category.

N-Gram Analysis

### Int[14]•

Texts = '.join(data['text']



String = texts.split("



def draw n\_gram(string,i):

N\_gram = (pd.Series(nltk.ngrams(string, i)).value\_counts())[:15]

N\_gram\_df = n\_gram\_df.reset\_index()

N\_gram\_df = n\_gram\_df.rename(columns={"index": "word", 0: "count")

Print(n\_gram\_df.head())

Plt.figure(figsize = (16, 9))

Return sns.barplot(x='count' ,y='word', data=n\_gram\_df)

Unigram-Analysis Intl-17--]:

Draw\_n\_gram(string ) word count

1. (trump,) 149603
2. (said,) 133030
3. (u,) 78516
4. (state,) 62726 4 (president,) 58790



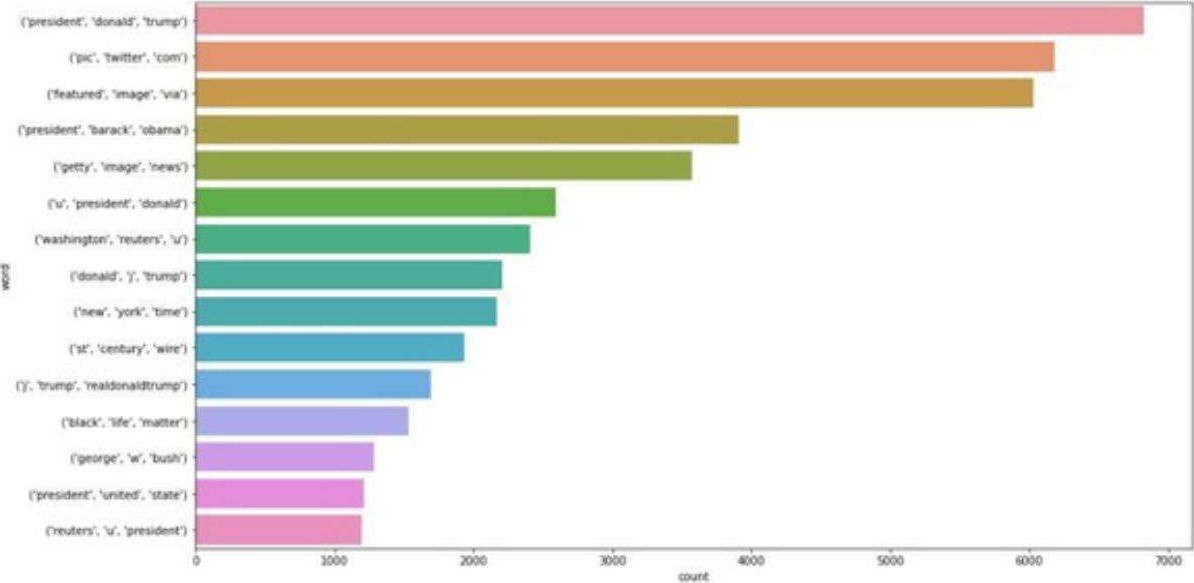
<AxesSubplot:xlabel='count', ylabel='word '>

<AxesSubplot:xlabel='count', ylabel='word'>

Trigram-Analysis -Intl-191:



<AxesSubplot:xlabel='count', ylabel='word'>



Train Test Split

Int[20]:

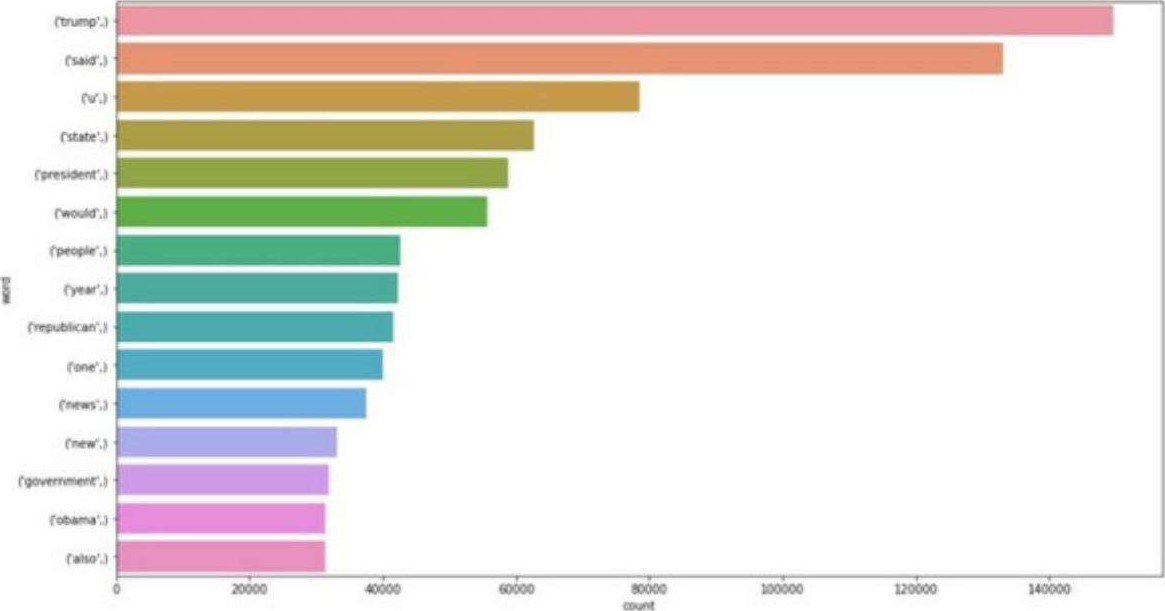
X train, X test, y\_train, y test = train test\_split(data['text'], data['target'], random\_state=0)



Tokenizing Text -> Repsesenting each word by a number

Mapping of orginal word to number is preserved in word index property of tokenizer





Bigram-Analysis IntL18J-•

Draw\_n\_gram(string,2)

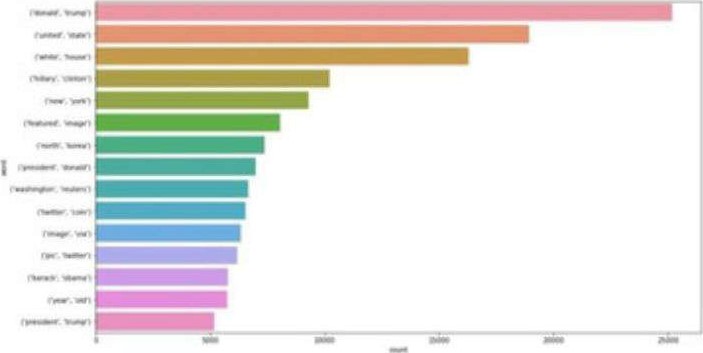
(donald, trump) 252Û3

2 . (united, state) 18943

3 . white, house) 16296

1. . hillary, clinton) 18217
2. . (new, york) 9395

Out[18]:



Conclusion:

Decision Tree Classifier and Logistic regression are performing well.

**Project Conclusion:**

In conclusion, fake news detection using Natural Language Processing (NLP) is a vital and evolving field in the fight against misinformation. NLP techniques have shown promise in identifying and flagging potentially deceptive content by analyzing linguistic patterns, sources, and context. However, it is essential to acknowledge that no single method is foolproof, and ongoing research and development are necessary to stay ahead of increasingly sophisticated fake news tactics. Collaborative efforts between researchers, technology companies, and fact-checkers are crucial in building more robust and accurate fake news detection systems to promote trustworthy information in the digital age.