PHASE-4 : DEVELOPMENT Part-2

Project Title: Fake News Detection

Problem Definition:

The problem is to develop a fake news detection model using a Kaggle dataset. The goal is to distinguish between genuine and fake news articles based on their titles and text. This project involves using natural language processing (NLP) techniques to preprocess the text data, building a machine learning model for classification, and evaluating the model's performance.

Dataset Link:

https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset

Introduction:

Creating a fake news detection model involves several steps, including text preprocessing, feature extraction, model training, and evaluation.

Project Design Steps:

1. Data Source:
   Choose the fake news dataset available on Kaggle, containing articles titles and text, along with their labels (genuine or fake).This dataset should represent various topics and sources to ensure the model's generalizability.

2. Data Preprocessing:
   Clean and preprocess the textual data to prepare it for analysis. This may involve tasks such as tokenization, lowercasing, removing stop words, and stemming/lemmatization.

3. Feature Extraction:

   Convert the text data into numerical features that can be used as input for machine learning algorithms. Common techniques include:

   • TF-IDF (Term Frequency-Inverse Document Frequency): Capturing the importance of words in a document relative to a corpus.

   • Word Embeddings: Representing words as dense vectors using models like Word2Vec, GloVe, or BERT embeddings.

   • N-grams: Capturing sequences of words to capture context

4. Model Selection:

Choose an appropriate machine learning or deep learning model for fake news detection. Common choices include logistic regression, Naive Bayes, Support Vector Machines (SVM), recurrent neural networks (RNNs), or transformers like BERT.

5. Model Training:

   Train the selected model using the preprocessed data, using labeled examples to learn the patterns associated with real and fake news.The model learns to distinguish between real and fake news based on the provided features.

6. Evaluation:

Evaluate the model's performance using metrics like accuracy, precision, recall, F1-score, and ROC-AUC. It's essential to consider the class imbalance issue, as fake news might be a minority class

Installation steps and the Program:

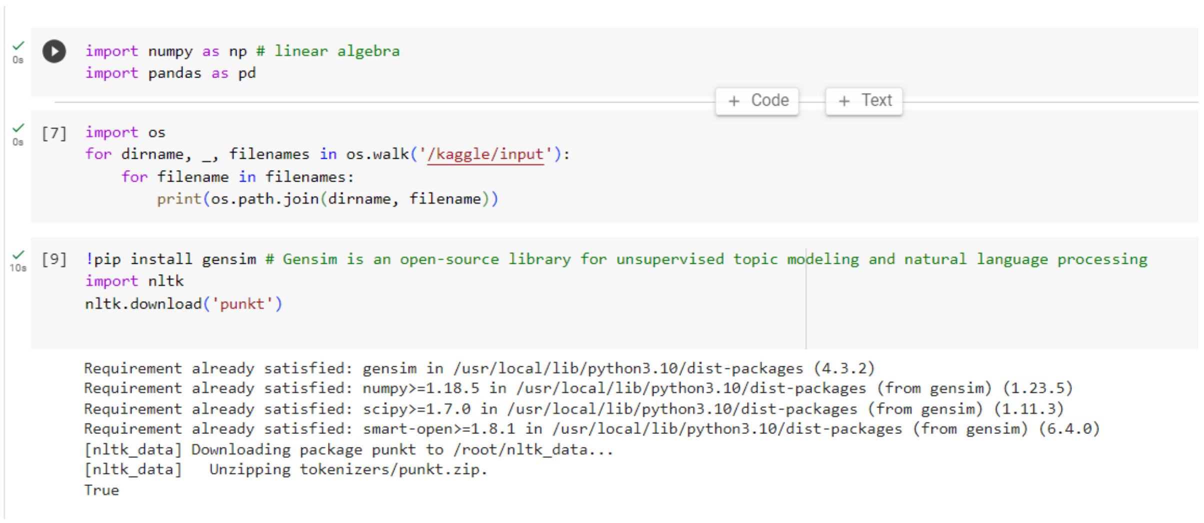
```
import numpy as np

import pandas as pd

import os

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

    for filename in filenames:

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

!pip install genism

import nltk

nltk.download('punkt')


import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from wordcloud import WordCloud, STOPWORDS

import nltk

import re

from nltk.corpus import stopwords

import seaborn as sns

import gensim
```

```
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS
import plotly.express as px
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score
from sklearn.metrics import confusion_matrix
fake_data = pd.read_csv('/kaggle/input/fake-and-real-news-dataset/Fake.csv')
print("fake_data",fake_data.shape)
true_data= pd.read_csv('/kaggle/input/fake-and-real-news-dataset/True.csv')
print("true_data",true_data.shape)
fake_data.head(5)
true_data.head(5)
true_data['target'] = 1
fake_data['target'] = 0
df = pd.concat([true_data, fake_data]).reset_index(drop = True)
df['original'] = df['title'] + ' ' + df['text']
df.head()
df.isnull().sum()
stop_words = stopwords.words('english')
stop_words.extend(['from', 'subject', 're', 'edu', 'use'])
def preprocess(text):
  result = []
  for token in gensim.utils.simple_preprocess(text):
    if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 2 and token not in
stop_words:
      result.append(token)
  return result
df.subject=df.subject.replace({'politics':'PoliticsNews','politicsNews':'PoliticsNews'})
sub_tf_df=df.groupby('target').apply(lambda x:x['title'].count()).reset_index(name='Counts')
sub_tf_df.target.replace({0:'False',1:'True'},inplace=True)
fig = px.bar(sub_tf_df, x="target", y="Counts",
            color='Counts', barmode='group',
            height=350)
fig.show()
sub_check=df.groupby('subject').apply(lambda x:x['title'].count()).reset_index(name='Counts')
fig=px.bar(sub_check,x='subject',y='Counts',color='Counts',title='Count of News Articles by Subject')
fig.show()
df['clean_title'] = df['title'].apply(preprocess)
df['clean_title'][0]
df['clean_joined_title']=df['clean_title'].apply(lambda x:" ".join(x))
plt.figure(figsize = (20,20))
wc = WordCloud(max_words = 2000 , width = 1600 , height = 800 , stopwords =
stop_words).generate(" ".join(df[df.target == 1].clean_joined_title))
plt.imshow(wc, interpolation = 'bilinear')
maxlen = -1
for doc in df.clean_joined_title:
   tokens = nltk.word_tokenize(doc)
   if(maxlen<len(tokens)):
      maxlen = len(tokens)
print("The maximum number of words in a title is =", maxlen)
fig = px.histogram(x = [len(nltk.word_tokenize(x)) for x in df.clean_joined_title], nbins = 50)
fig.show()
X_train, X_test, y_train, y_test = train_test_split(df.clean_joined_title, df.target, test_size =
0.2,random_state=2)
vec_train = CountVectorizer().fit(X_train)
X_vec_train = vec_train.transform(X_train)
X_vec_test = vec_train.transform(X_test)
model = LogisticRegression(C=2)
model.fit(X_vec_train, y_train)
predicted_value = model.predict(X_vec_test)
accuracy_value = roc_auc_score(y_test, predicted_value)
print(accuracy_value)
cm = confusion_matrix(list(y_test), predicted_value)
plt.figure(figsize = (7, 7))
```

```
sns.heatmap(cm, annot = True,fmt='g',cmap='viridis')

df['clean_text'] = df['text'].apply(preprocess)

df['clean_joined_text']=df['clean_text'].apply(lambda x:" ".join(x))

plt.figure(figsize = (20,20))

wc = WordCloud(max_words = 2000 , width = 1600 , height = 800 , stopwords =
stop_words).generate(" ".join(df[df.target == 1].clean_joined_text))

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

maxlen = -1

for doc in df.clean_joined_text:

   tokens = nltk.word_tokenize(doc)

   if(maxlen<len(tokens)):

      maxlen = len(tokens)

print("The maximum number of words in a News Content is =", maxlen)

fig = px.histogram(x = [len(nltk.word_tokenize(x)) for x in df.clean_joined_text], nbins = 50)

fig.show()

X_train, X_test, y_train, y_test = train_test_split(df.clean_joined_text, df.target, test_size =
0.2,random_state=2)

vec_train = CountVectorizer().fit(X_train)

X_vec_train = vec_train.transform(X_train)

X_vec_test = vec_train.transform(X_test)

model = LogisticRegression(C=2.5)

model.fit(X_vec_train, y_train)

predicted_value = model.predict(X_vec_test)

accuracy_value = roc_auc_score(y_test, predicted_value)

print(accuracy_value)

prediction = []

for i in range(len(predicted_value)):

   if predicted_value[i].item() > 0.5:

      prediction.append(1)

   else:

      prediction.append(0)

cm = confusion_matrix(list(y_test), prediction)

plt.figure(figsize = (6, 6))
```

sns.heatmap(cm, annot = True,fmt='g')

Import the data :

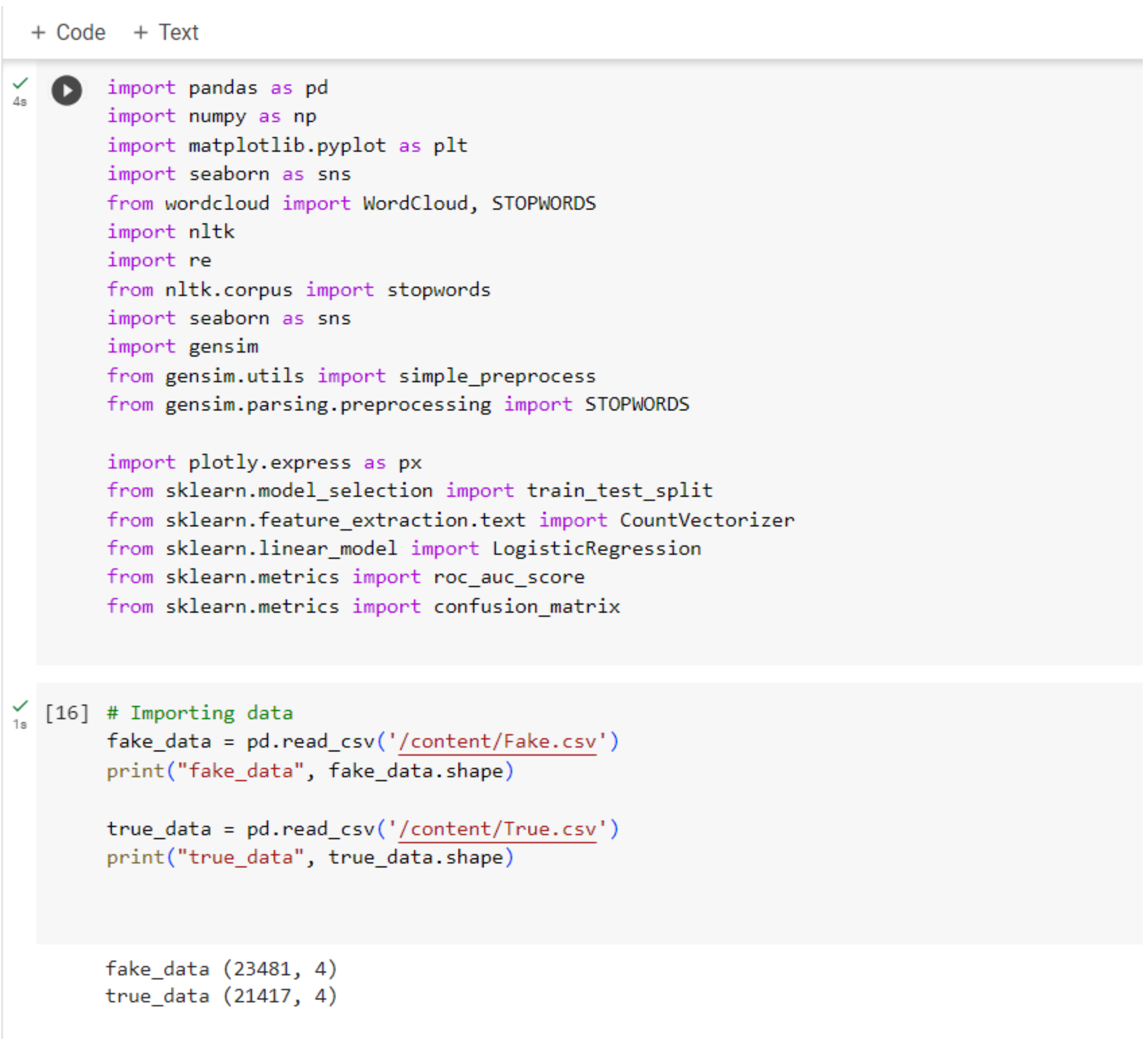


```
import numpy as np # linear algebra
import pandas as pd
```

```
[7] import os
    for dirname, _, filenames in os.walk('/kaggle/input'):
        for filename in filenames:
            print(os.path.join(dirname, filename))
```

```
[9] !pip install gensim # Gensim is an open-source library for unsupervised topic modeling and natural language processing
    import nltk
    nltk.download('punkt')
```

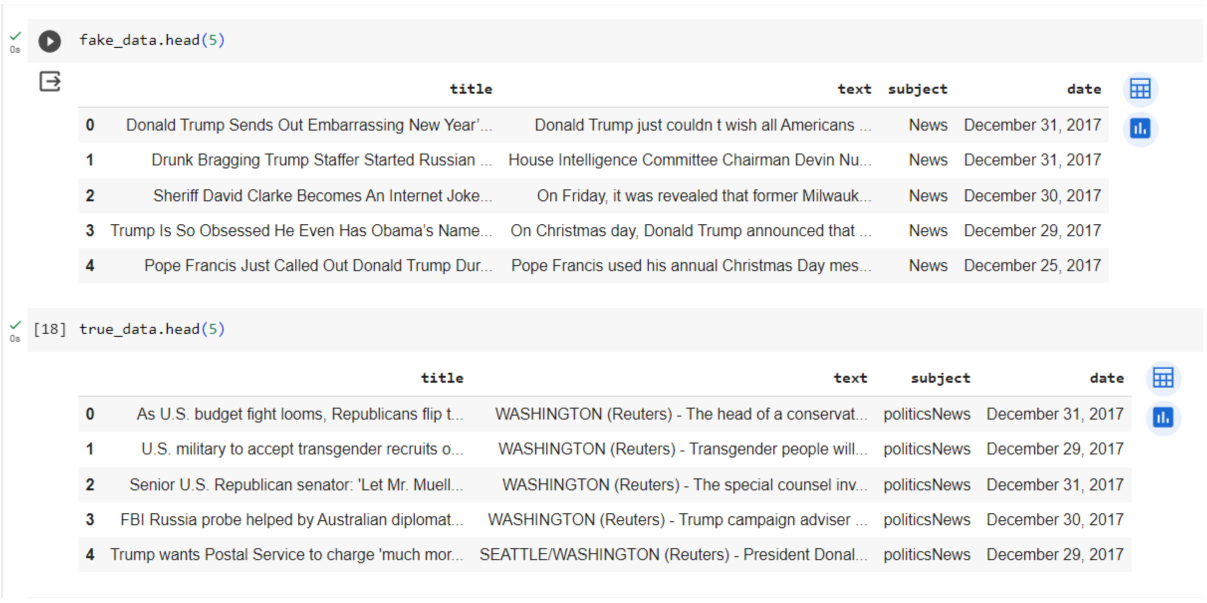
```
Requirement already satisfied: gensim in /usr/local/lib/python3.10/dist-packages (4.3.2)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.10/dist-packages (from gensim) (1.23.5)
Requirement already satisfied: scipy>=1.7.0 in /usr/local/lib/python3.10/dist-packages (from gensim) (1.11.3)
Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/python3.10/dist-packages (from gensim) (6.4.0)
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS
import nltk
import re
from nltk.corpus import stopwords
import seaborn as sns
import gensim
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS

import plotly.express as px
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score
from sklearn.metrics import confusion_matrix
```

```
[16] # Importing data
     fake_data = pd.read_csv('/content/Fake.csv')
     print("fake_data", fake_data.shape)

     true_data = pd.read_csv('/content/True.csv')
     print("true_data", true_data.shape)
```

```
fake_data (23481, 4)
true_data (21417, 4)
```

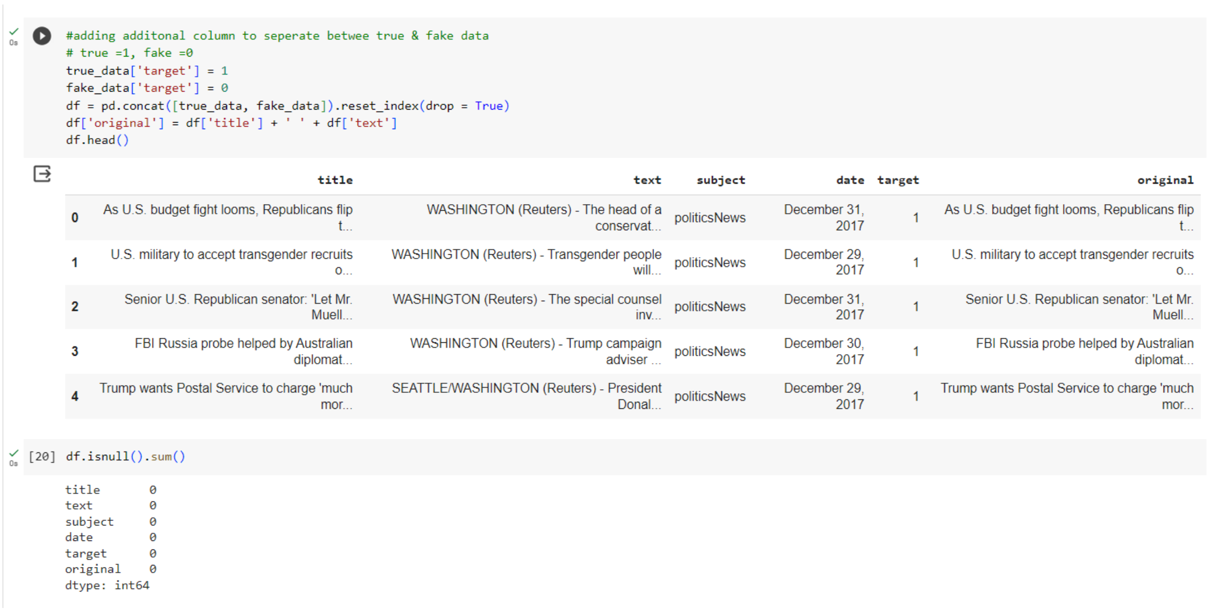


```
fake_data.head(5)
```

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

```
[18] true_data.head(5)
```

	title	text	subject	date
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017



```
#adding additonal column to seperate betwee true & fake data
# true =1, fake =0
true_data['target'] = 1
fake_data['target'] = 0
df = pd.concat([true_data, fake_data]).reset_index(drop = True)
df['original'] = df['title'] + ' ' + df['text']
df.head()
```

	title	text	subject	date	target	original
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017	1	As U.S. budget fight looms, Republicans flip t...
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017	1	U.S. military to accept transgender recruits o...
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017	1	Senior U.S. Republican senator: 'Let Mr. Muell...
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017	1	FBI Russia probe helped by Australian diplomat...
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017	1	Trump wants Postal Service to charge 'much mor...

```
[20] df.isnull().sum()
```

```
title       0
text        0
subject     0
date        0
target      0
original    0
dtype: int64
```

Data cleanup:

```python
import nltk
nltk.download('stopwords')
```

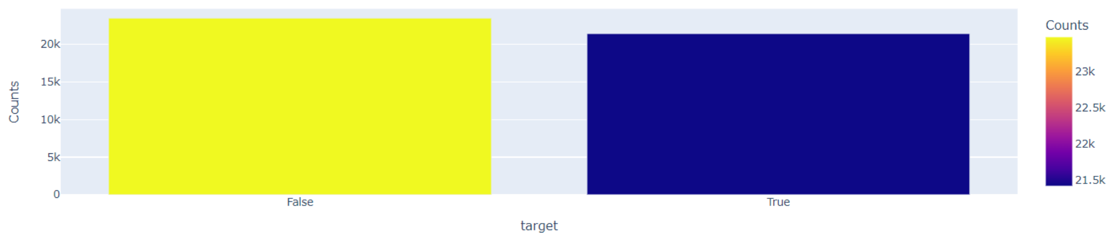
```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

[23]
```python
import nltk
nltk.download('stopwords')

from nltk.corpus import stopwords
import gensim
from gensim.utils import simple_preprocess

stop_words = stopwords.words('english')
stop_words.extend(['from', 'subject', 're', 'edu', 'use'])

def preprocess(text):
    result = []
    for token in gensim.utils.simple_preprocess(text):
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 2 and token not in stop_words:
            result.append(token)

    return result
```

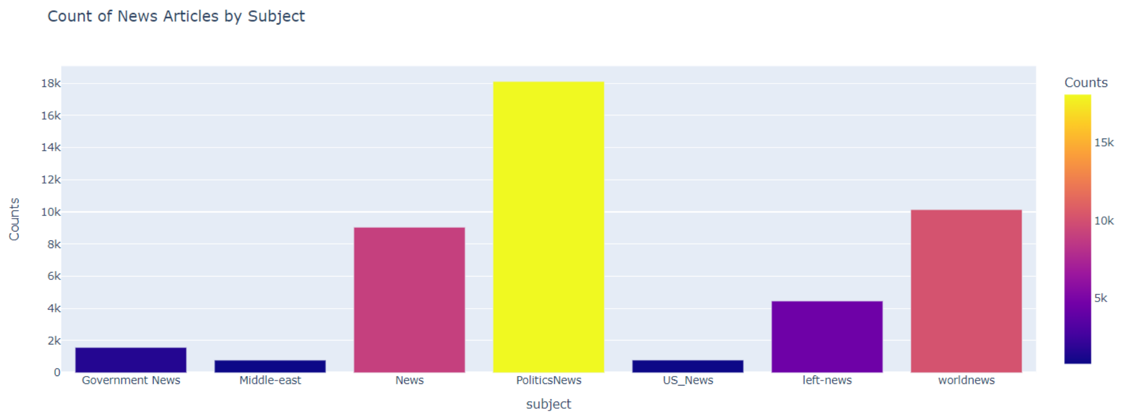
```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

[24]
```python
# Transforming the unmatching subjects to the same notation
df.subject=df.subject.replace({'politics':'PoliticsNews','politicsNews':'PoliticsNews'})
```

[25]
```python
sub_tf_df=df.groupby('target').apply(lambda x:x['title'].count()).reset_index(name='Counts')
sub_tf_df.target.replace({0:'False',1:'True'},inplace=True)
fig = px.bar(sub_tf_df, x="target", y="Counts",
             color='Counts', barmode='group',
             height=350)
fig.show()
```



```python
sub_check=df.groupby('subject').apply(lambda x:x['title'].count()).reset_index(name='Counts')
fig=px.bar(sub_check,x='subject',y='Counts',color='Counts',title='Count of News Articles by Subject')
fig.show()
```



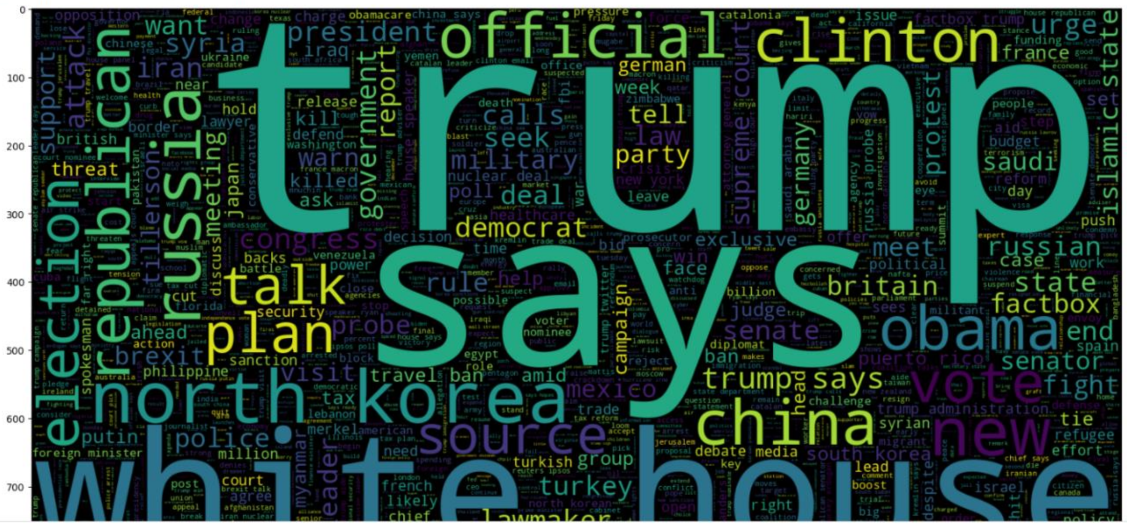
```
df['clean_title'] = df['title'].apply(preprocess)
df['clean_title'][0]
```

```
['budget', 'fight', 'looms', 'republicans', 'flip', 'fiscal', 'script']
```

```
[28] df['clean_joined_title']=df['clean_title'].apply(lambda x:" ".join(x))
```

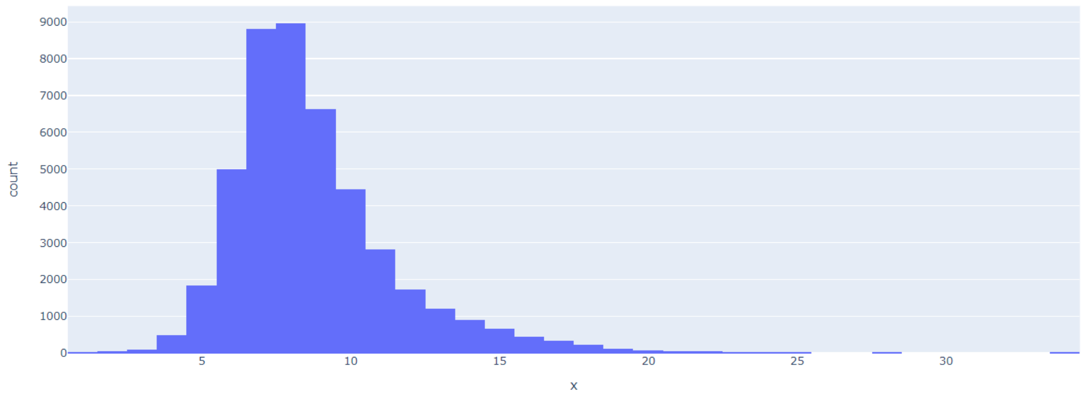
```
plt.figure(figsize = (20,20))
wc = WordCloud(max_words = 2000 , width = 1600 , height = 800 , stopwords = stop_words).generate(" ".join(df[df.target == 1].clean_joined_title))
plt.imshow(wc, interpolation = 'bilinear')
```

```
<matplotlib.image.AxesImage at 0x7adbcf47df30>
```

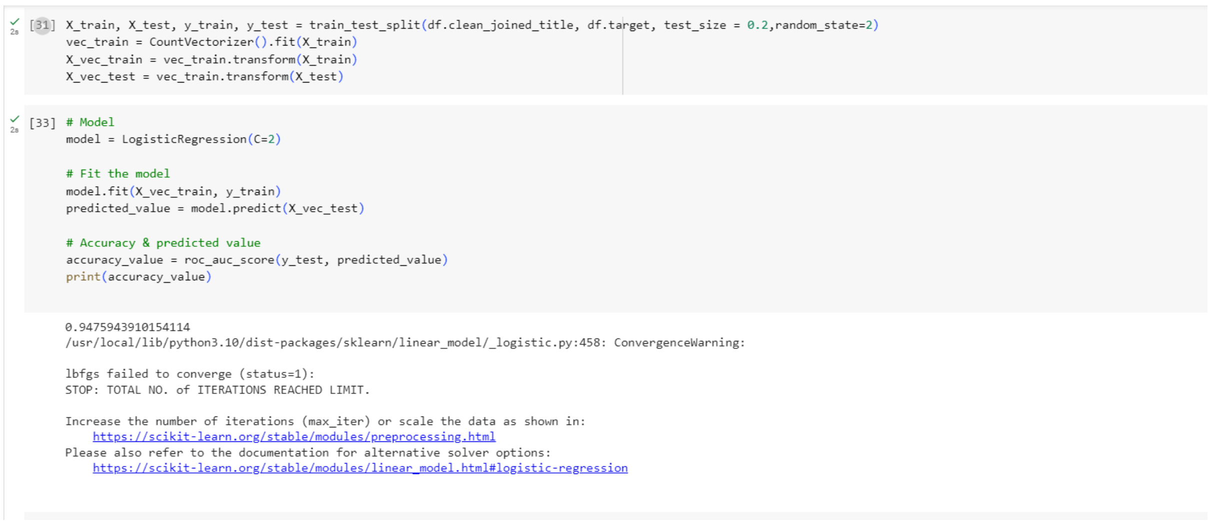


```
maxlen = -1
for doc in df.clean_joined_title:
    tokens = nltk.word_tokenize(doc)
    if(maxlen<len(tokens)):
        maxlen = len(tokens)
print("The maximum number of words in a title is =", maxlen)
fig = px.histogram(x = [len(nltk.word_tokenize(x)) for x in df.clean_joined_title], nbins = 50)
fig.show()
```

```
The maximum number of words in a title is = 34
```



Creating Prediction Model :

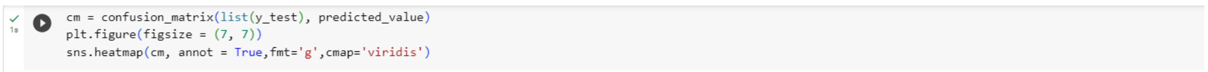


```
[31] X_train, X_test, y_train, y_test = train_test_split(df.clean_joined_title, df.target, test_size = 0.2,random_state=2)
     vec_train = CountVectorizer().fit(X_train)
     X_vec_train = vec_train.transform(X_train)
     X_vec_test = vec_train.transform(X_test)
```

```
[33] # Model
     model = LogisticRegression(C=2)

     # Fit the model
     model.fit(X_vec_train, y_train)
     predicted_value = model.predict(X_vec_test)

     # Accuracy & predicted value
     accuracy_value = roc_auc_score(y_test, predicted_value)
     print(accuracy_value)
```

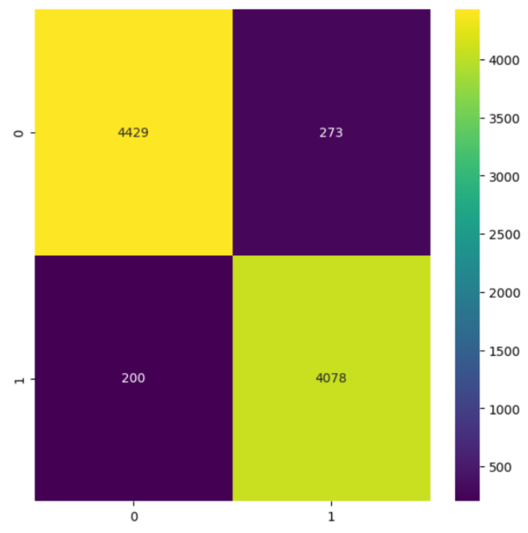
```
0.9475943910154114
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
```

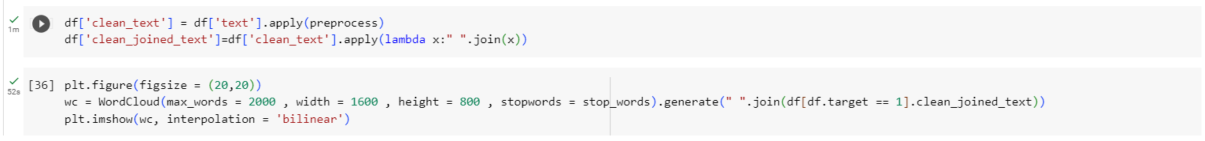
Create the confusion matrix :



```
cm = confusion_matrix(list(y_test), predicted_value)
plt.figure(figsize = (7, 7))
sns.heatmap(cm, annot = True,fmt='g',cmap='viridis')
```

```
<Axes: >
```

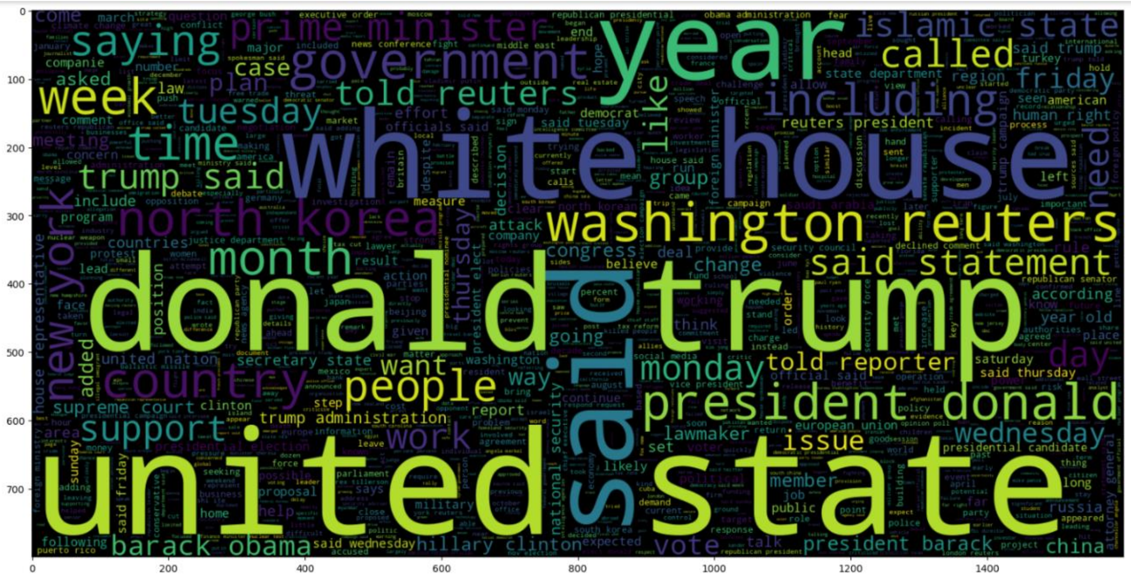




```
df['clean_text'] = df['text'].apply(preprocess)
df['clean_joined_text']=df['clean_text'].apply(lambda x:" ".join(x))
```

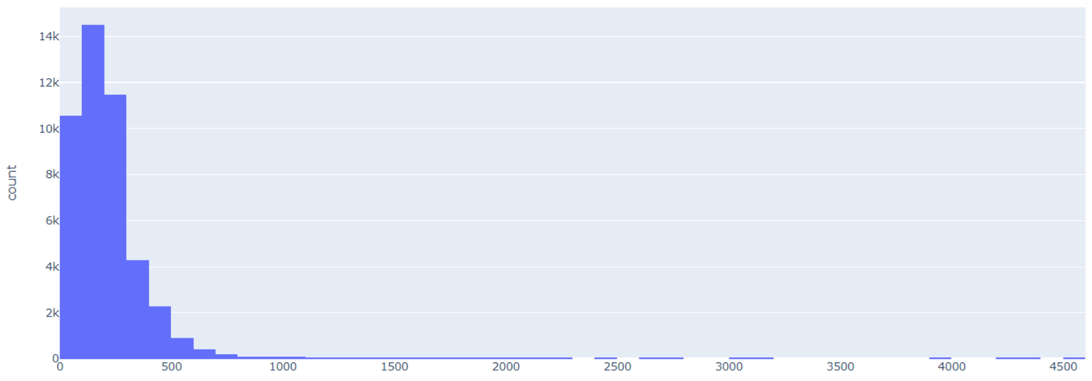
```
[36] plt.figure(figsize = (20,20))
     wc = WordCloud(max_words = 2000 , width = 1600 , height = 800 , stopwords = stop_words).generate(" ".join(df[df.target == 1].clean_joined_text))
     plt.imshow(wc, interpolation = 'bilinear')
```

Checking the content of news :



```
maxlen = -1
for doc in df.clean_joined_text:
    tokens = nltk.word_tokenize(doc)
    if(maxlen<len(tokens)):
        maxlen = len(tokens)
print("The maximum number of words in a News Content is =", maxlen)
fig = px.histogram(x = [len(nltk.word_tokenize(x)) for x in df.clean_joined_text], nbins = 50)
fig.show()
```

```
The maximum number of words in a News Content is = 4573
```

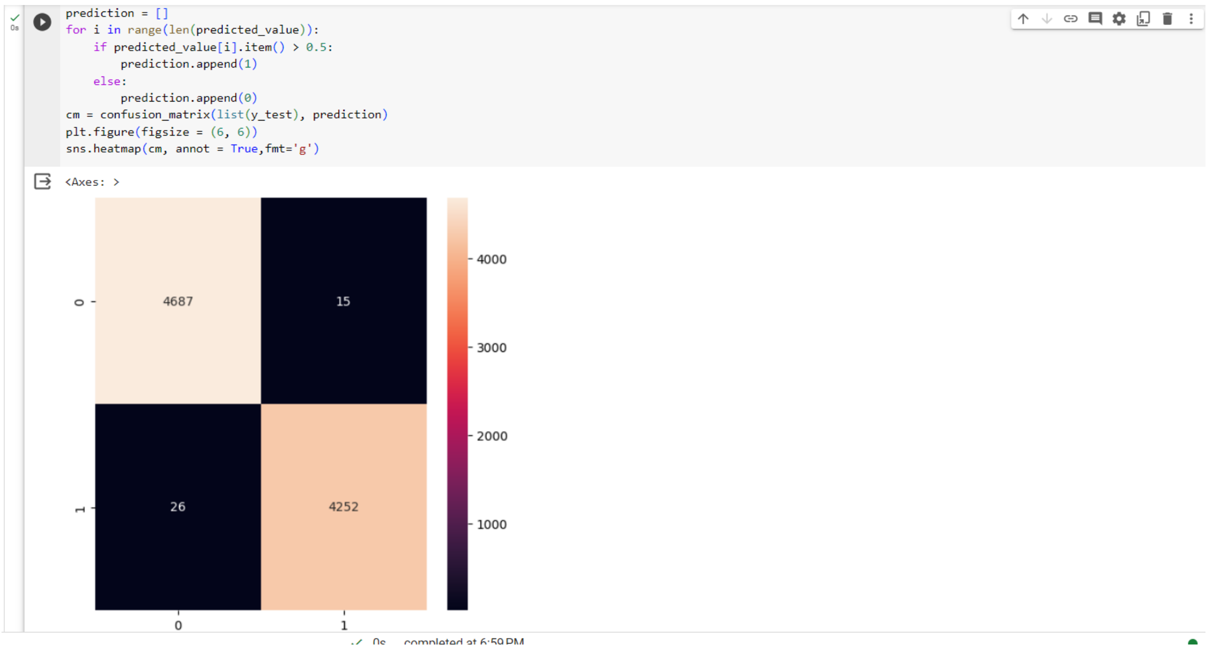


Predicting the Model :

```
X_train, X_test, y_train, y_test = train_test_split(df.clean_joined_text, df.target, test_size = 0.2,random_state=2)
vec_train = CountVectorizer().fit(X_train)
X_vec_train = vec_train.transform(X_train)
X_vec_test = vec_train.transform(X_test)
model = LogisticRegression(C=2.5)
model.fit(X_vec_train, y_train)
predicted_value = model.predict(X_vec_test)
accuracy_value = roc_auc_score(y_test, predicted_value)
print(accuracy_value)
```

```
0.9953661308915527
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
```

```
prediction = []
for i in range(len(predicted_value)):
    if predicted_value[i].item() > 0.5:
        prediction.append(1)
    else:
        prediction.append(0)
cm = confusion_matrix(list(y_test), prediction)
plt.figure(figsize = (6, 6))
sns.heatmap(cm, annot = True,fmt='g')
```



```
prediction = []
for i in range(len(predicted_value)):
    if predicted_value[i].item() > 0.5:
        prediction.append(1)
    else:
        prediction.append(0)
cm = confusion_matrix(list(y_test), prediction)
plt.figure(figsize = (6, 6))
sns.heatmap(cm, annot = True,fmt='g')
```

```
<Axes: >
```

Conclusion :

In this phase,our model's training and evaluation part is developed.