FAKE NEWS DETECTION USING NLP

TEAM MEMBERS

513121106061

513121106094

513121106097

513121106312

Phase 5 :Project Documentation & Submission

Introduction:

- Fake news detection using NLP is a dynamic field that continues to evolve with advances in machine learning and natural language processing. It plays a vital role in maintaining the credibility and reliability of information in the digital age.

- It has become humanly impossible to identify fake news on the online portals across the globe.The sheer volume and the pace at which news spreads calls the need to create a ML model to classify the fake from true news.

In[1]:

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

Import numpy as np # linear algebra
Import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

Input data files are available in the read-only "../input/" directory

For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

```
Import os

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

    For filename in filenames:

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

You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"

You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session

In[2]:

```
!pip install gensim # Gensim is an open-source library for unsupervised topic modeling and natural language processing

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

Import the data & Clean ups:

In[3]:

#importing data

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)

In[4]:

Fake_data.head(5)

Out[4]:

	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

In[5]:

True_data.head(5)

Out[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

In[6]:

#adding additonal column to separate 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()

Out[6]:

	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...

In[7]:

Df.isnull().sum()

Out[7]:

Title 0

Text 0

Subject 0

Date 0

Target 0

Original 0

Dtype: int64

In[8]:

<u>Data Clean up</u>

Create a function here that will be responsible to remove any unneccesary words (Stopwords) from the data provided

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

In[9]:

Transforming the unmatching subjects to the same notation

Df.subject=df.subject.replace({'politics':'PoliticsNews','politicsNews':'PoliticsNews'})

In[10]

Sub_tf_df=df.groupby('target').apply(lambda x:x['title'].count()).reset_index(name='Counts')

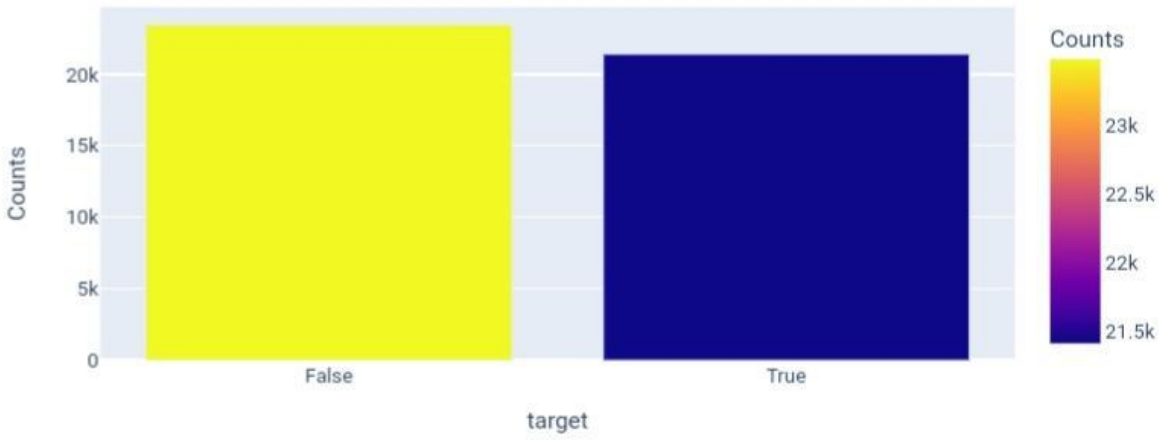
Sub_tf_df.target.replace({0:'False',1:'True'},inplace=True)

Fig = px.bar(sub_tf_df, x="target", y="Counts",

Color='Counts', barmode='group',

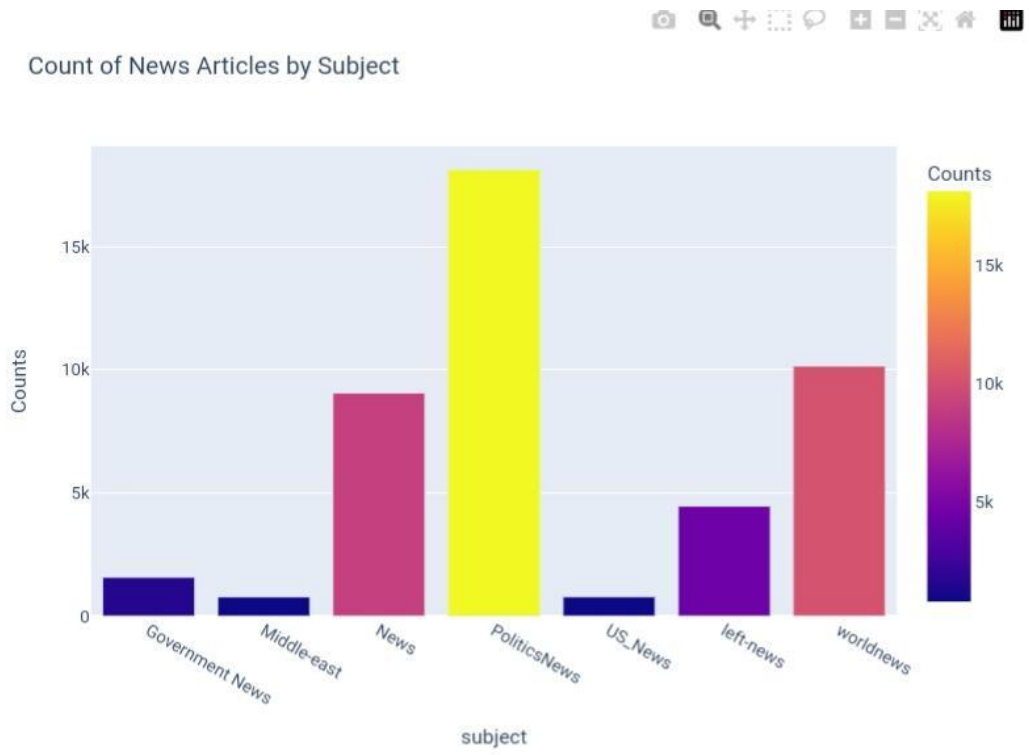
Height=350)

Fig.show()



In[11]:

```
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()
```



In[12]:

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

Df['clean_title'][0]
```

Out[12]:

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

In[13]:

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

In[14]:

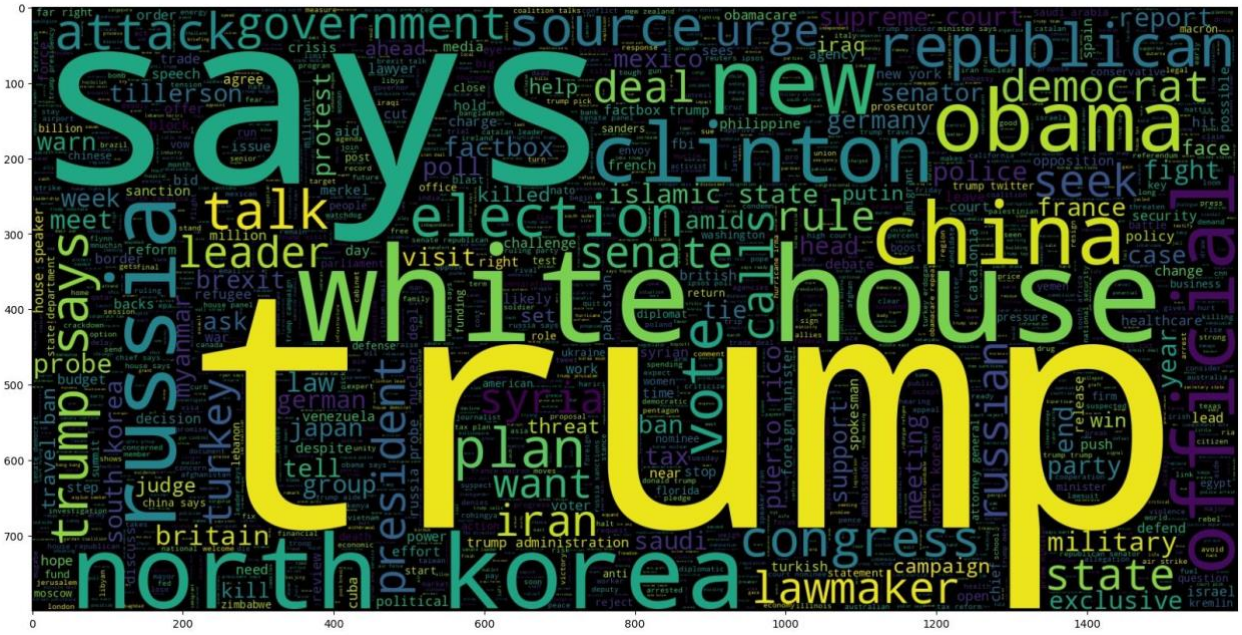
Plt.figure(figsize = (20,20))

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’)

Out[14]:

<matplotlib.image.AxesImage at 0x7cc99e7d3130>



In[15]:

Maxlen = -1

For doc in df.clean_joined_title:

  Tokens = nltk.word_tokenize(doc)

  If(maxlen<len(tokens)):

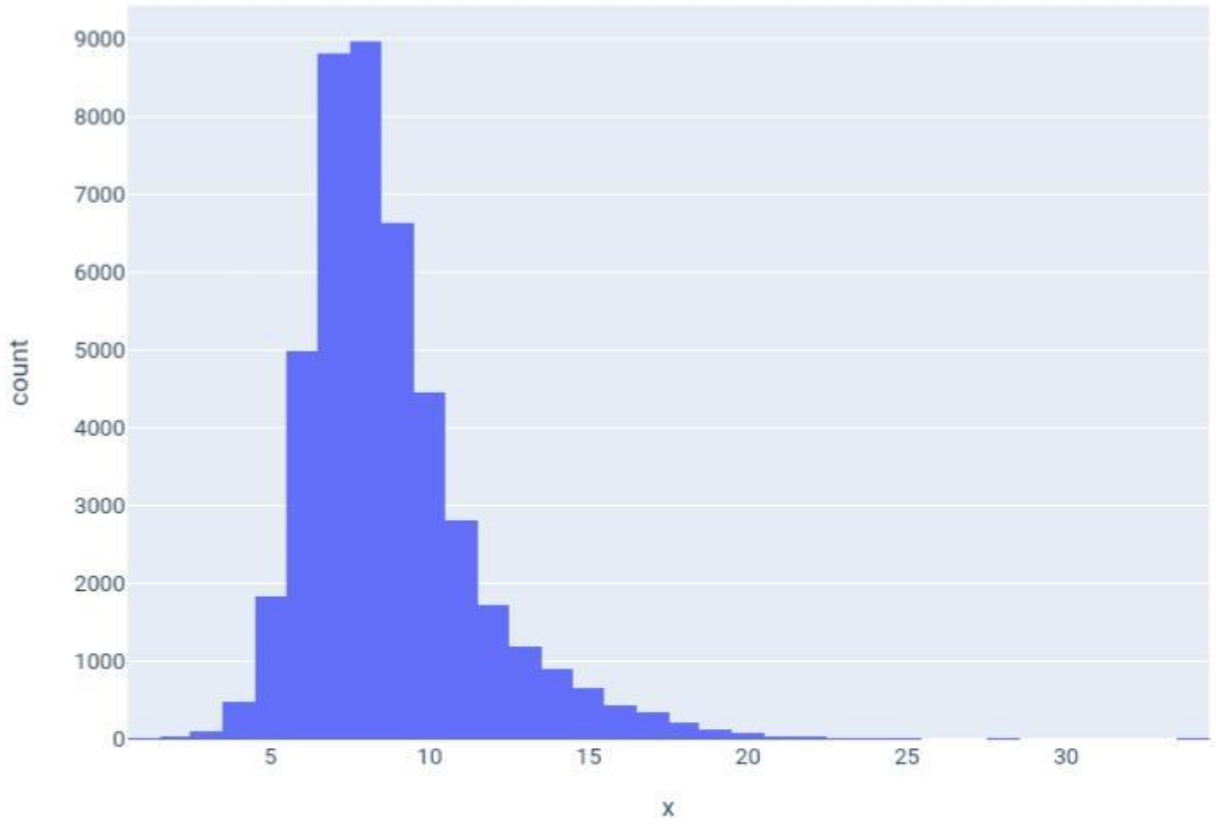
    Maxlen = len(tokens)

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:

In[16]:

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)

In[17]:

#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

/opt/conda/lib/python3.10/site-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:

In[18]:

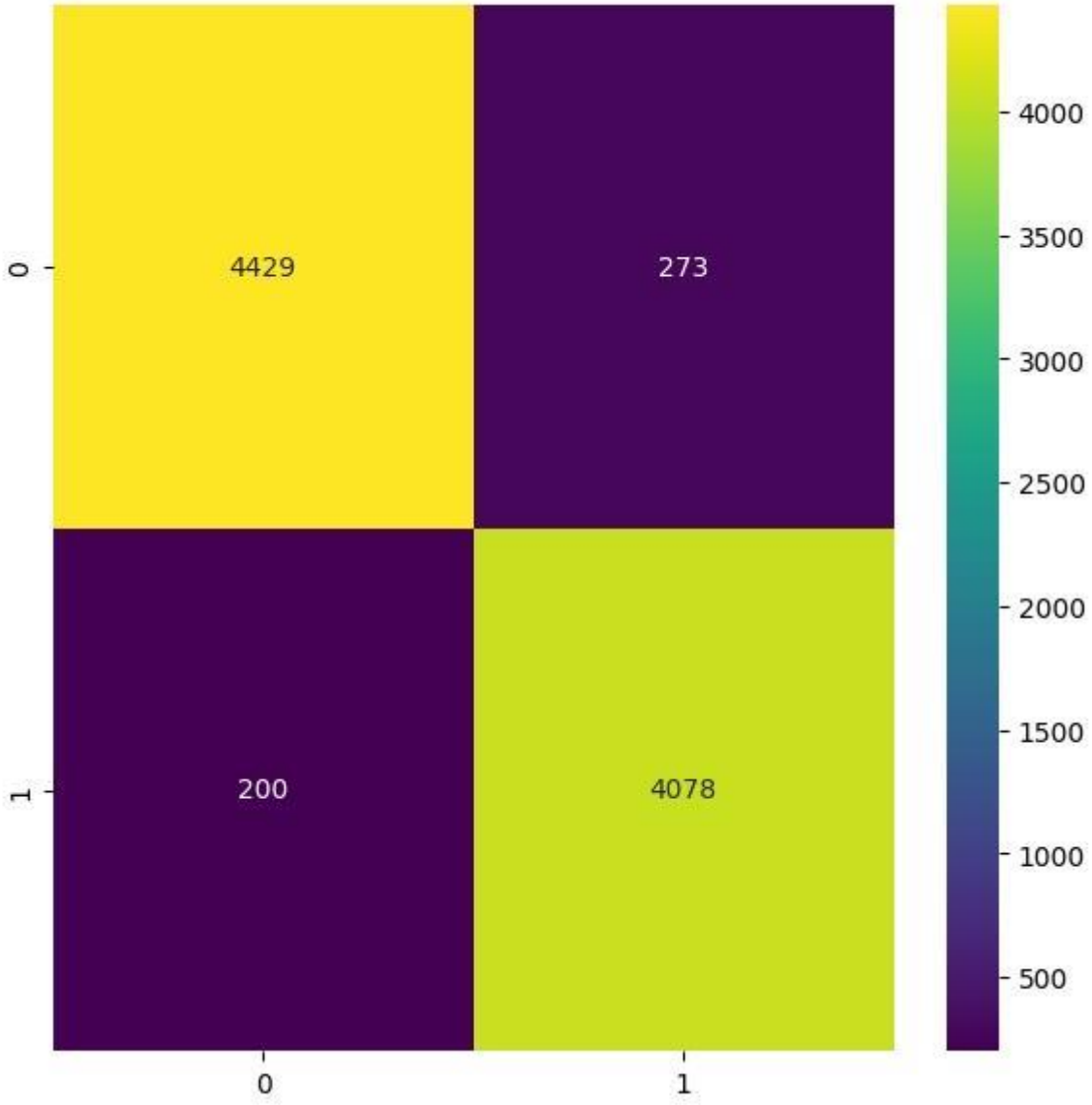
Cm = confusion_matrix(list(y_test), predicted_value)

Plt.figure(figsize = (7, 7))

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

Out[18]:

<Axes: >



4465 Fake News have been Classified as Fake

4045 Real News have been classified as Real

In[19]:

Df[‘clean_text’] = df[‘text’].apply(preprocess)

Df[‘clean_joined_text’]=df[‘clean_text’].apply(lambda x:” “.join(x))

In[20]:

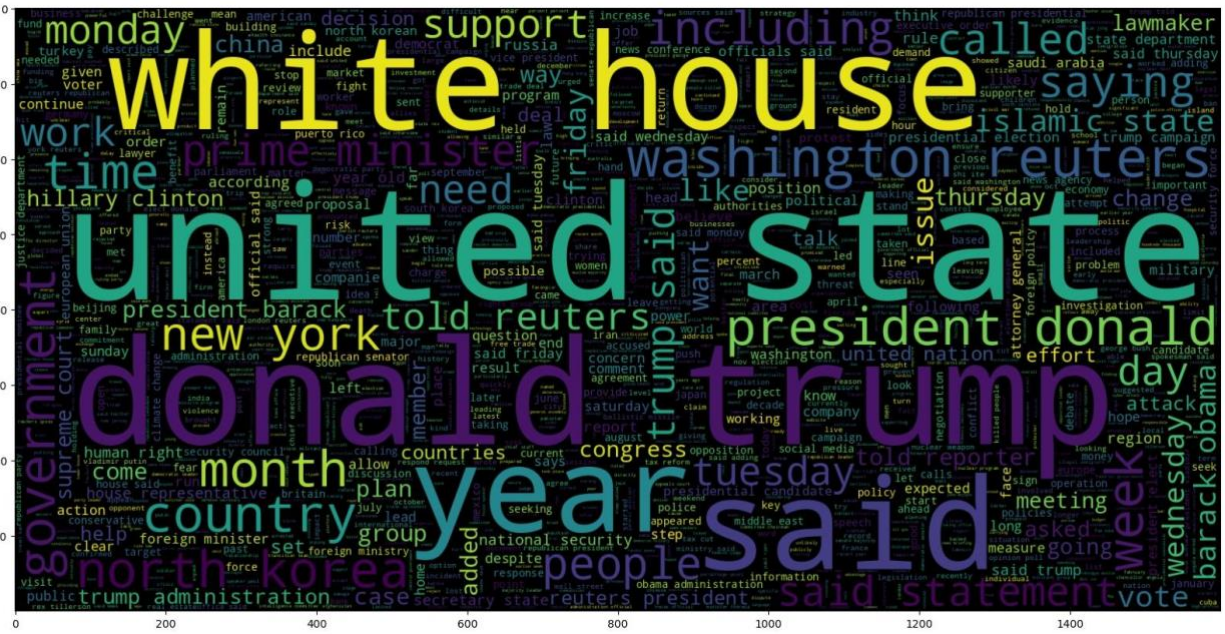
Plt.figure(figsize = (20,20))

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’)

Out[20]:

<matplotlib.image.AxesImage at 0x7cc99e7d1db0>



In[21]:

Maxlen = -1

For doc in df.clean_joined_text:

  Tokens = nltk.word_tokenize(doc)

  If(maxlen<len(tokens)):

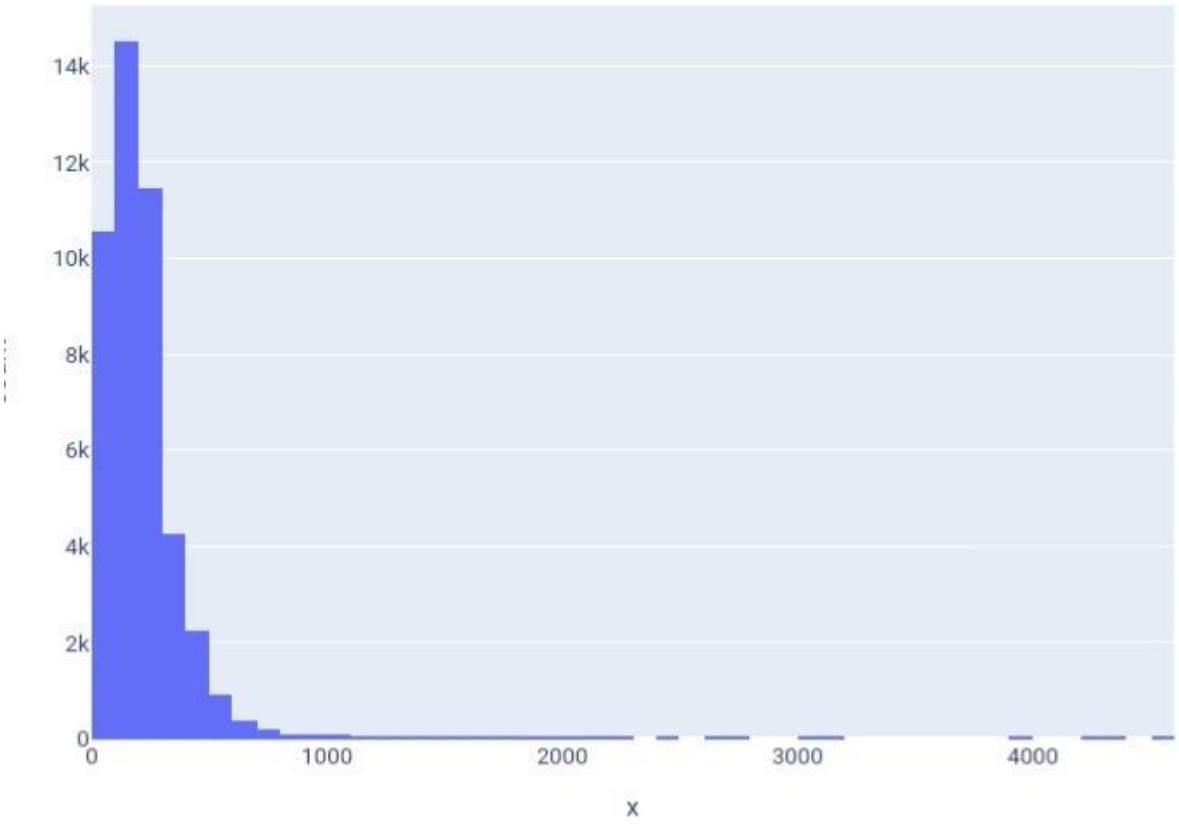
    Maxlen = len(tokens)

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:

In[22]:

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)
```

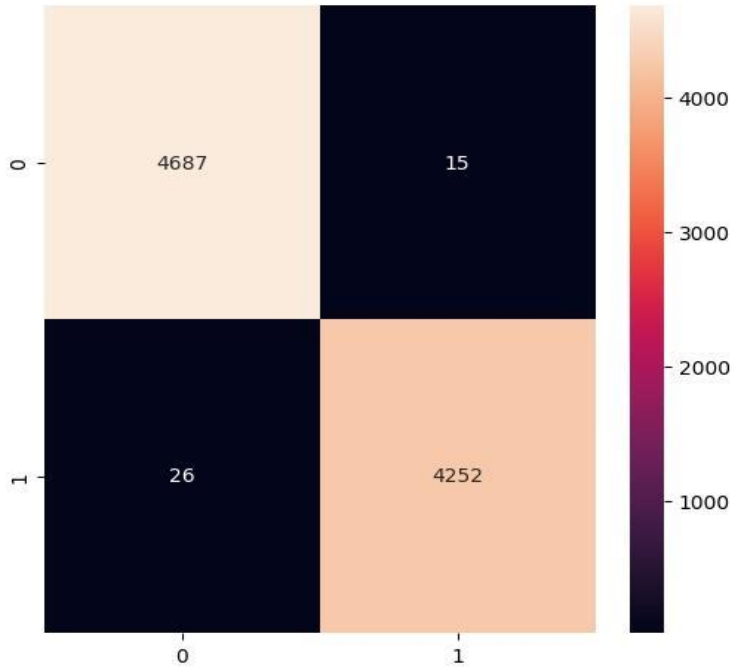
```
0.9953661308915527
/opt/conda/lib/python3.10/site-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning:

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
```

In[23]:

```
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')
```

Out[23]:

<Axes: >



Conclusion:

In conclusion, the use of Natural Language Processing (NLP) for fake news detection is a critical and evolving field in today's digital information landscape. It encompasses several key stages, including data collection, preprocessing, feature extraction, machine learning model selection, evaluation using various metrics, explainability, continuous learning, and deployment. While NLP models hold promise in identifying and mitigating the spread of misinformation, it's essential to understand that no model is foolproof. Combining NLP tools with human judgment and critical thinking remains a robust strategy in the ongoing battle against fake news. The development and refinement of NLP-based fake news

detection techniques continue to be essential in safeguarding the integrity of information in the digital age.