Fake news detection using NLP

TEAM MEMBERS:

1.P.swathi

2.M.Nadhiya

3.V.S.varsha

Department of electronics and communication engineering – III year

Students of NPR college of engineering and technology,

natham(tk), Dindigul(dt).

Fake news has a negative impact on individuals and society, hence the detection of fake news is becoming a bigger field of interest for data scientists. Attempts to leverage artificial intelligence technologies particularly machine/deep learning techniques and natural language processing (NLP) to automatically detect fake news and prevent its viral spread have recently been actively discussed.

Fake news detection using NLP

Abstract:

The proliferation of fake news in online media poses a significant threat to information integrity and public trust. This research presents an innovative approach to tackle this issue through the utilization of Natural Language Processing (NLP) techniques. The study explores various NLP methods, including text classification, sentiment analysis, and linguistic feature extraction, to distinguish between genuine and fabricated news articles. Leveraging a diverse dataset and state-of-the-art machine learning models, this research demonstrates the effectiveness of NLP in identifying fake news with high accuracy. The findings highlight the potential of NLP as a valuable tool in combating misinformation, thereby safeguarding the credibility of information disseminated through digital platforms. This work contributes to the ongoing efforts to develop robust solutions for fake news detection and encourages further exploration in this critical domain.

Designing an innovative solution for fake news detection using Natural Language Processing (NLP) involves several steps. Below, I'll outline the complete process in detail:

1. Problem Definition and Understanding:

- Define the problem clearly: Fake news detection aims to identify and classify news articles or content as either reliable or fake.

- Understand the challenges: Recognize the complexities involved in fake news, including misinformation, disinformation, and different forms of deceptive content.

2. Data Collection:

- Gather a diverse dataset of news articles labeled as real or fake. You can use existing datasets like Snopes, PolitiFact, or create your own.

- Ensure the dataset is representative and balanced, containing a mix of both real and fake news.

3. Data Preprocessing:

- Clean and preprocess the text data:

  - Tokenization: Split text into words or subword tokens.

  - Lowercasing: Convert all text to lowercase.

  - Removing punctuation, stop words, and special characters.

- Vectorization: Convert text into numerical form using techniques like TF-IDF or word embeddings (Word2Vec, GloVe).

4. Feature Engineering:

- Extract relevant features from the text data, such as:

  - N-grams: Sequential word combinations.

  - Sentiment analysis scores.

  - Named entity recognition.

  - Meta-information like source credibility and publication date.

5. Model Selection:

- Choose appropriate NLP models for fake news detection, such as:

  - Traditional Machine Learning Models: Random Forest, Logistic Regression, Naive Bayes.

- Deep Learning Models: Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), Transformers (e.g., BERT, GPT-3).

6. Model Training:

- Split the dataset into training, validation, and testing sets.

- Train the selected model(s) using the training data.

- Hyperparameter tuning: Optimize model parameters for best performance using techniques like grid search or Bayesian optimization.

7. Model Evaluation:

- Assess the model's performance using appropriate evaluation metrics, including accuracy, precision, recall, F1-score, and AUC-ROC.

- Utilize techniques like cross-validation to ensure robustness.

8. Post-processing:

- Apply post-processing techniques to refine the model's predictions, such as thresholding or smoothing.

9. Integration and Deployment:

- Develop an easy-to-use API or user interface to make the model accessible.

- Ensure scalability, security, and reliability in the deployment infrastructure.

10. Continuous Improvement:

- Implement feedback mechanisms to continuously update and retrain the model as new data becomes available.

- Stay up-to-date with the latest advancements in NLP and fake news detection to improve the model's accuracy and effectiveness.

11. User Education:

- Educate users on how to interpret the model's results and the limitations of automated fake news detection.

12. Ethical Considerations:

- Address ethical concerns related to bias, privacy, and fairness in fake news detection.

- Be transparent about the model's decision-making process.

13. Monitoring and Feedback Loop:

- Implement a monitoring system to track the model's performance in real-time.

- Encourage user feedback to continuously improve the system.

14. Collaboration:

- Collaborate with fact-checking organizations and experts to enhance the model's accuracy and credibility.

15. Legal and Regulatory Compliance:

- Ensure compliance with data protection laws and regulations, such as GDPR or HIPAA.

Flow chart:

The system involves a lot of steps, data preprocessing, and machine learning models. Here's a high-level flowchart:

Start

|

|-- Data Collection

| |

| |-- Collect news articles and their labels (fake or real)

| |

| |-- Preprocess Data

| | |

| | |-- Tokenization

| | |

| | |-- Stopword Removal

| | |

| | |-- Lemmatization/Stemming

| |

| |-- Feature Extraction

| | |

| | |-- TF-IDF Vectorization

| | |

| | |-- Word Embeddings (e.g., Word2Vec, GloVe)

| |

|-- Build a Fake News Detection Model

| |

| |-- Split Data into Training and Testing Sets

| |

| |-- Train a Machine Learning Model (e.g., Logistic Regression, Random Forest, LSTM)

| |

|-- Model Evaluation

| |

| |-- Evaluate the model using metrics like accuracy, precision, recall, F1-score

| |

|-- Model Fine-tuning (Optional)

| |

| |-- Hyperparameter tuning or selecting different models

| |

|-- Deploy Model

| |

| |-- Deploy the model as a web application or API

| |

|-- Real-time Prediction

| |

| |-- User submits a news article for analysis

| |

| |-- Preprocess the user's input

| |

| |-- Use the trained model to classify the news as fake or real

| |

| |-- Display the result to the user

| |

|-- End

This flowchart outlines the major steps involved in building a fake news detection system using NLP and machine learning.

Code:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read
_csv)
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
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

```
fake_data = pd.read_csv('/kaggle/input/fake-and-real-news-datase
t/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()
```

Data Clean up

```
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 a
nd len(token) > 2 and token not in stop_words:
            result.append(token)
    return result
```

```
df.subject=df.subject.replace({'politics':'PoliticsNews','politi
csNews':'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)
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')
df['clean_title'] = df['title'].apply(preprocess)
df['clean_title'][0]


df['clean_joined_title']=df['clean_title'].apply(lambda x:" ".jo
in(x))
plt.figure(figsize = (20,20))
wc = WordCloud(max_words = 2000 , width = 1600 , height = 800 ,
stopwords = stop_words).generate(" ".join(df[df.target == 1].cle
an_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.c
lean_joined_title], nbins = 50)
```

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')
```

Checking the content of news

```
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].cle
an_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 =", maxl
en)
fig = px.histogram(x = [len(nltk.word_tokenize(x)) for x in df.c
lean_joined_text], nbins = 50)
```

Predicting the Model

```
X_train, X_test, y_train, y_test = train_test_split(df.clean_joi
ned_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')
```

Code explanation:

This code is a Python script for performing text classification and analysis on a dataset of news articlesto determine whether they are real or fake news . Let'sbreak down the code step by step:

Importing Libraries:

- The code starts by importing necessary Python libraries, including NumPy, pandas, scikit-learn (for machine learning), Matplotlib (for plotting), and otherrelated modules.

Data Loading:

- It loads two CSV files using `pd.read_csv` to read thereal news dataset (`True.csv`) and fake news dataset (`Fake.csv`) from a directory, presumably from a Kaggledataset.

Data Exploration:

- It explores the datasets by printing the first 50 rows ofthe real news dataset and checking the count of news articles in each subject category for both datasets.
- It also checks for missing values in the datasets and prints the shape (number of rows and columns) of eachdataset.

Label Assignment:

- It assigns labels to the news articles in the real andfake datasets. Real news is assigned a label of 1, and fake news is assigned a label of 0.

Data Concatenation:

- It concatenates the real and fake news datasets into asingle DataFrame called `data`, which combines both datasets. This merged dataset will be used for further analysis.

Data Splitting:

- It splits the data into training and testing sets. Xcontains the "text" column (the content of the newsarticles), and y contains the corresponding labels.

Text Vectorization:

- It uses the `CountVectorizer` to convert the text datainto numerical vectors. Both the training and testing data are transformed separately. `X_train_vectors` and `X_test_vectors` store the vectorized training and testingdata.

Multinomial Naive Bayes Classification:

- It uses a Multinomial Naive Bayes classifier to train onthe vectorized

training data (`X_train_vectors`) and theircorresponding labels (`y_train`).

- It then makes predictions on the testing data and calculates the accuracy of the model using scikit-learn's `accuracy_score`.

Predicting New Texts:

- It uses the trained Naive Bayes classifier to predict thelabels for two new text samples.
- The new texts are vectorized using the same `CountVectorizer`, and their predictions are printed.

Alternative Text Classification:

- The code performs an alternative text classification task. It combines the "title" and "text" columns from thereal and fake news datasets and creates a new DataFrame.

It uses TF-IDF vectorization (`TfidfVectorizer`) totransform the combined text data into numerical vectors.

- Another Multinomial Naive Bayes classifier is trainedon this vectorized data.

Prediction for a Single Title:

- There's a function `predict_label` that takes an input title, combines it with an empty input text, vectorizes theinput data, and predicts its label (fake or true).
- It demonstrates the function by using it to predict thelabel for a given input title.

Processed code with output:

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

In [1]:

```
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))
```

In [2]:

```
import nltk
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]:

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

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]:

```
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
```

Data Clean up

In [8]:

```
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 a
nd len(token) > 2 and token not in stop_words:
            result.append(token)

    return result
```

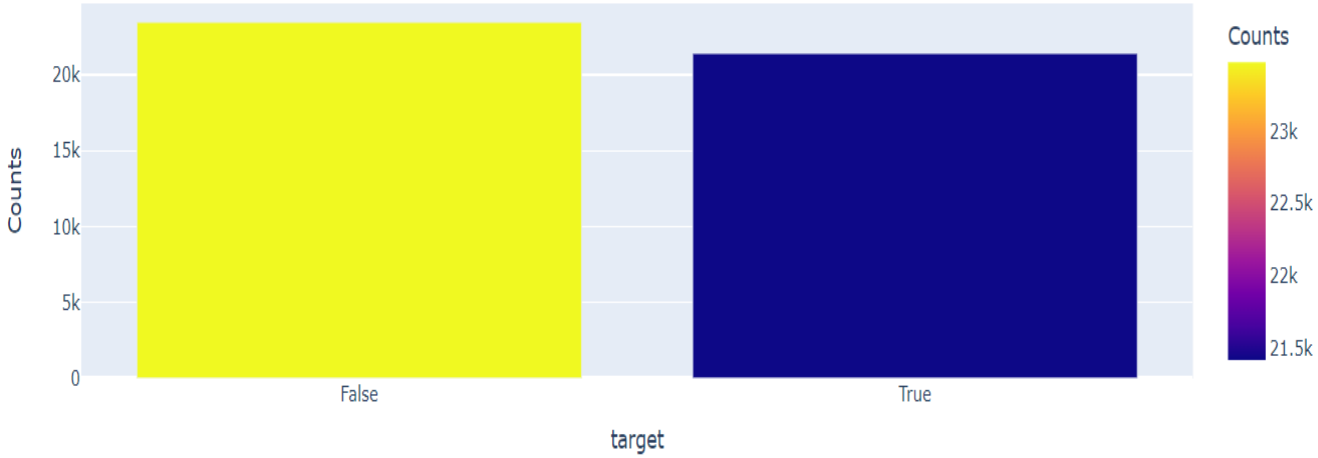
In [9]:

```
df.subject=df.subject.replace({'politics':'PoliticsNews','politi
csNews':'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)
```

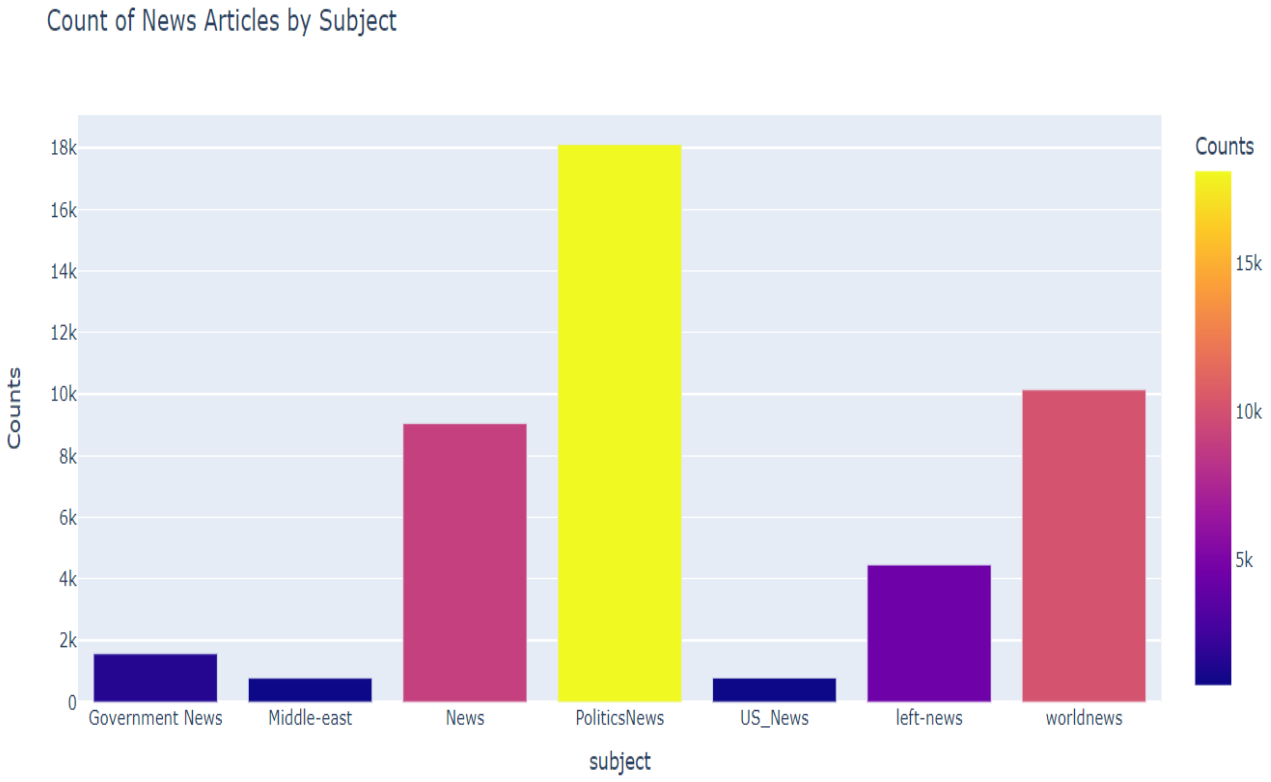
```
fig.show()
```



The data looks balanced and no issues on building the model

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', 's
cript']
```

In [13]:

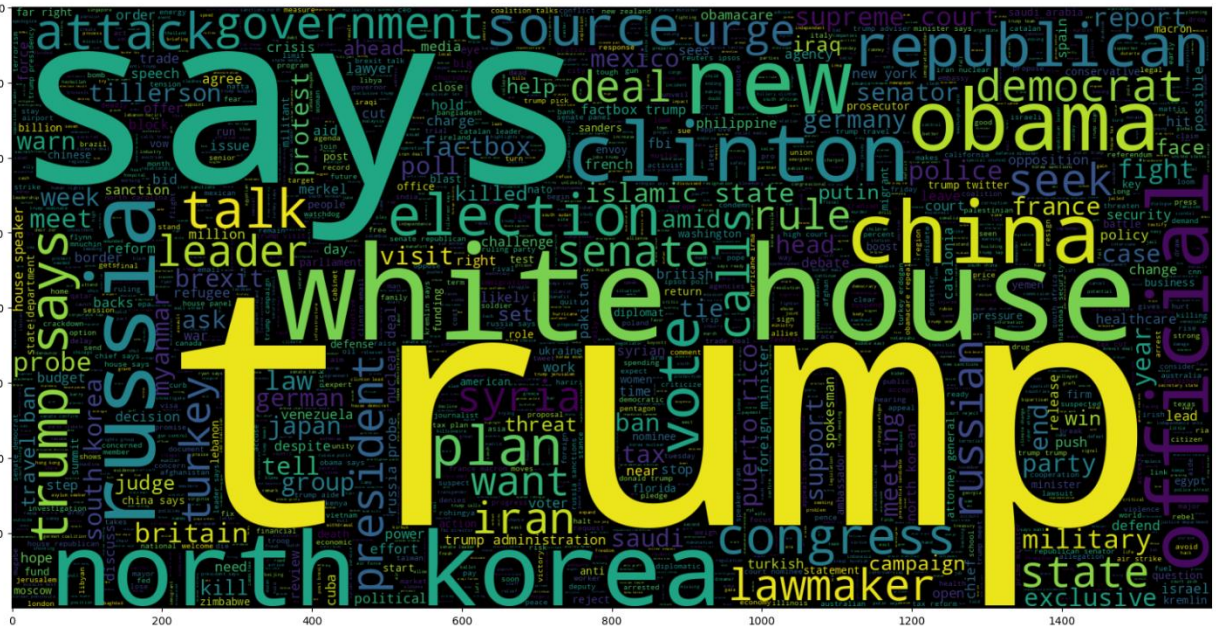
```
df['clean_joined_title']=df['clean_title'].apply(lambda x:" ".jo
in(x))
```

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].cle
an_joined_title))
plt.imshow(wc, interpolation = 'bilinear')
```

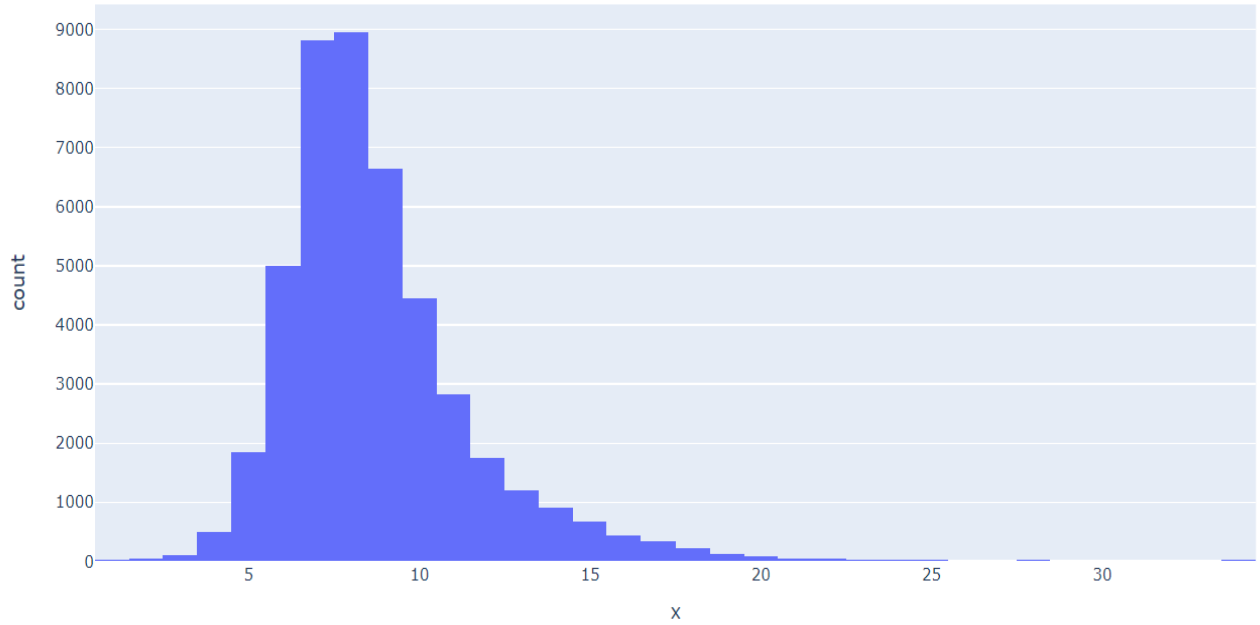
Out[14]:

```
<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.c
lean_joined_title], nbins = 50)
fig.show()
```




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

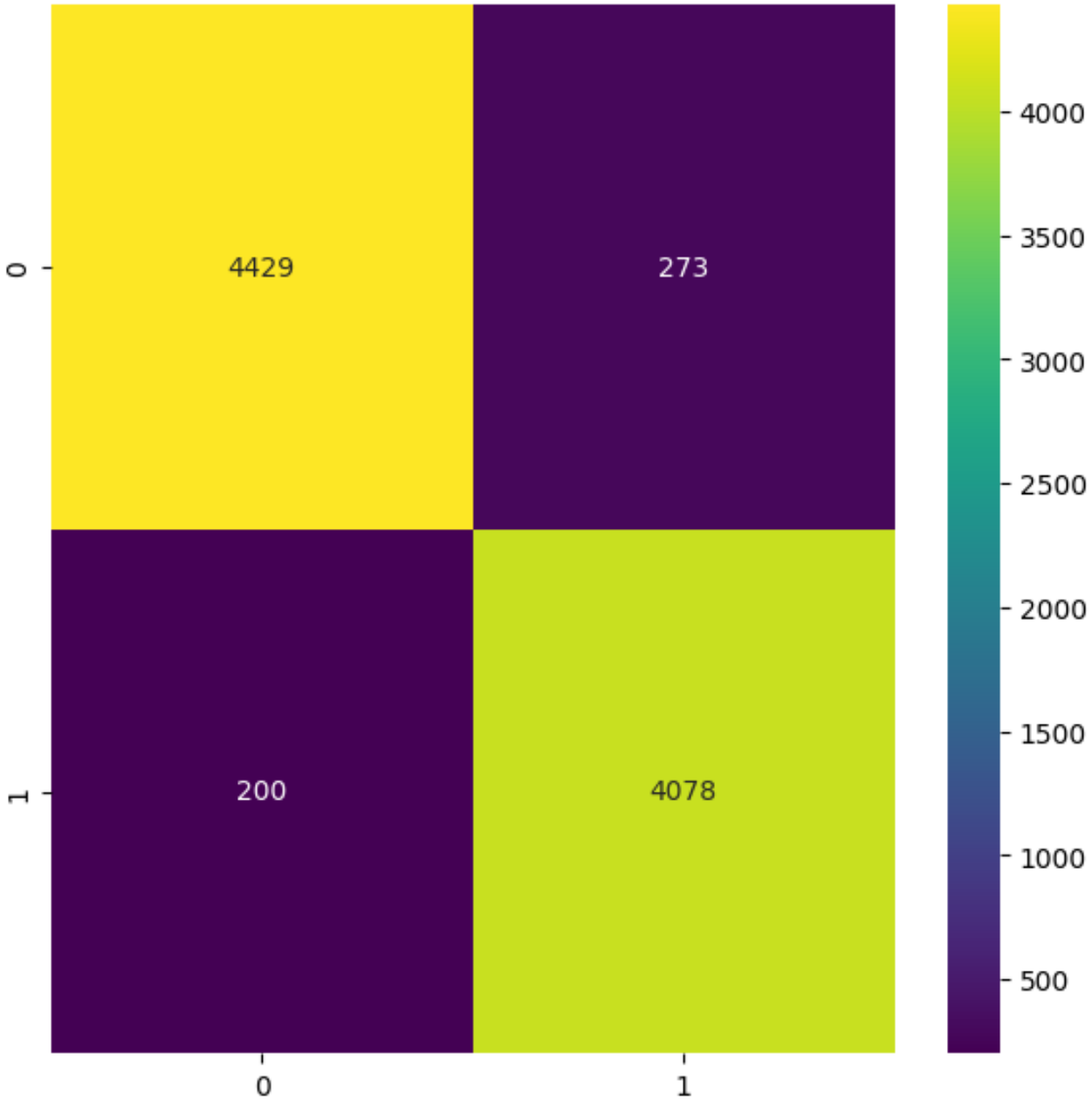
Create the confusion matrix

In [16]:

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

Out[16]:

```
<Axes: >
```



- 4465 Fake News have been Classified as Fake
- 4045 Real News have been classified as Real

Checking the content of news

In [17]:

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

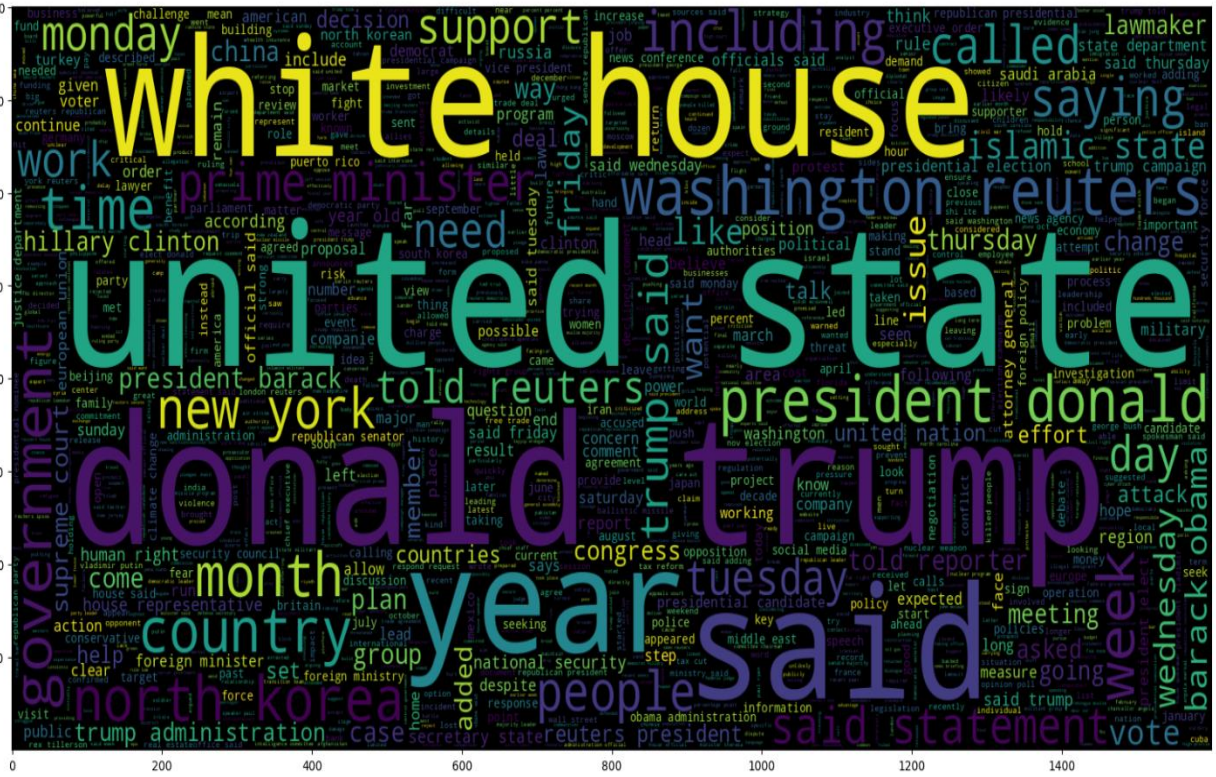
In [18]:

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

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

Out[18]:

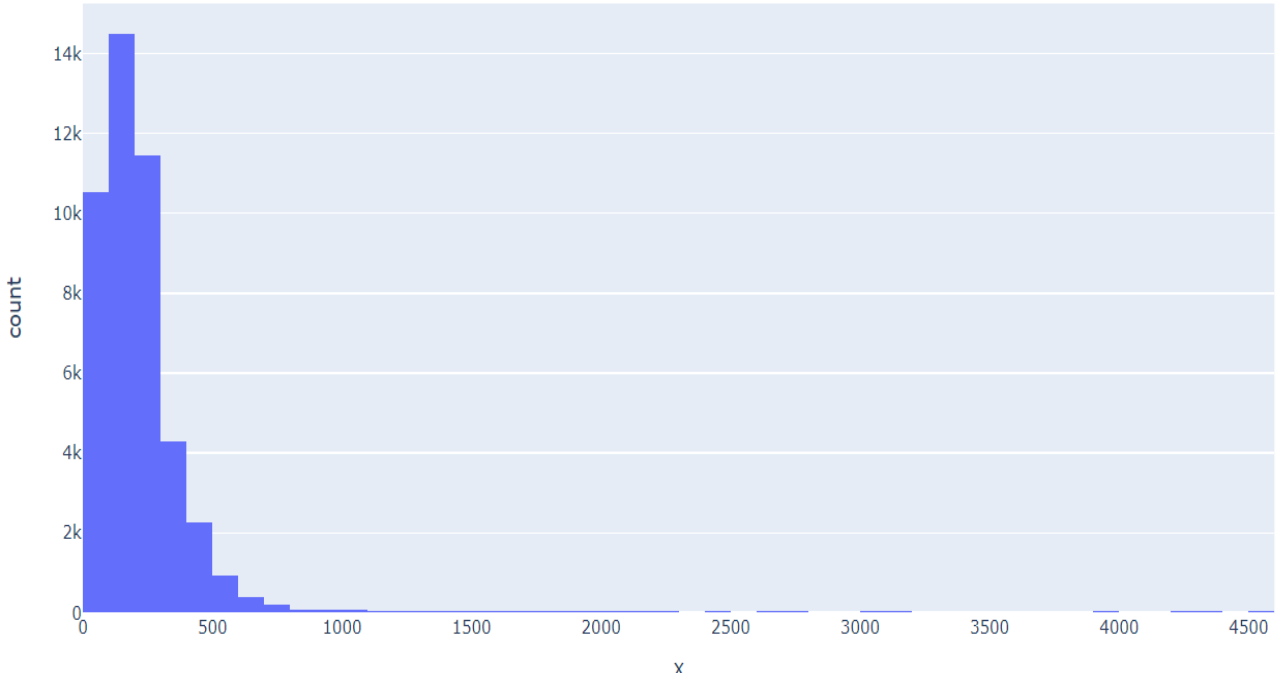
```
<matplotlib.image.AxesImage at 0x7cc99e7d1db0>
```



In [19]:

```
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 =", maxl
en)

fig = px.histogram(x = [len(nltk.word_tokenize(x)) for x in df.c
lean_joined_text], nbins = 50)
fig.show()
```




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

Predicting the Model

In [20]:

```
X_train, X_test, y_train, y_test = train_test_split(df.clean_joi
ned_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

In [21]:

```
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[21]:

```
<Axes: >
```