**TITLE : FAKE NEWS DETECTION USING NLP**

**introduction :**

**Fake news detection using Natural Language Processing (NLP) involves leveraging computational linguistics and machine learning techniques to identify misinformation or fake news articles. Here's a general approach to fake news detection using NLP**

**Detecting fake news using Natural Language Processing (NLP) techniques is a significant application in the field of machine learning.**

**Problem statement :**

**In today's digital age, the proliferation of misinformation and fake news poses a significant threat to society, democracy, and public discourse. With the rise of social media platforms and online news sources, distinguishing between genuine news and fabricated content has become increasingly challenging. This challenge has given rise to a critical need for reliable and efficient methods to detect fake news.**

**Design thinking :**

**Empathize:**

**Understand the challenges faced by users in identifying fake news.**

**Conduct user interviews and surveys to grasp the concerns and needs of the audience.**

**Explore existing solutions and their limitations to learn from past attempts at fake news detection.**

**Define:**

**Clearly define the problem by synthesizing the insights gathered during the empathize phase.**

**Establish specific goals, such as high accuracy, real-time processing, and adaptability to evolving misinformation tactics.**

**Identify key metrics for evaluating the effectiveness of the fake news detection system.**

**Ideate:**

**Brainstorm potential NLP techniques and algorithms suitable for fake news detection.**

**Encourage creativity and diverse thinking among team members to generate innovative ideas.**

**Explore different feature extraction methods, such as word embeddings, TF-IDF, and linguistic features, to capture nuanced language patterns.**

**Prototype:**

**Develop a prototype of the NLP-based fake news detection system using a representative dataset.**

**Implement various NLP algorithms, such as recurrent neural networks (RNNs), long short-term memory networks (LSTMs), or transformer-based models like BERT, to evaluate their performance.**

**Experiment with different pre-processing techniques, including text cleaning, tokenization, and stemming, to enhance the model's input data quality.**

**Test:**

**Evaluate the prototype using diverse and real-world datasets, ensuring it covers various topics, sources, and writing styles.**

**Fine-tune the model based on feedback and performance metrics, iterating on the design to enhance accuracy and efficiency.**

**Conduct A/B testing to compare the new system's performance against existing solutions or baseline models.**

**Implement:**

**Develop a user-friendly interface for the fake news detection system, allowing users to easily input text or URLs for analysis.**

**Integrate the NLP model into the interface, ensuring seamless communication between the user input and the backend processing.**

**Implement a real-time processing mechanism, enabling the system to handle incoming data streams efficiently.**

**Iterate:**

**Gather user feedback and monitor the system's performance in real-world scenarios.**

**Iterate on the design based on user input, emerging trends in misinformation, and advancements in NLP techniques.**

**Continuously update the system to adapt to evolving language use and new deceptive tactics employed by misinformation creators.**

**Deploy:**

**Deploy the finalized fake news detection system on reliable and scalable infrastructure to handle user demand.**

**Implement security measures to protect user data and ensure the system's integrity against potential attacks.**

**Provide regular updates and maintenance to address any issues and improve the system's overall performance.**

**Objective:**

**The primary objective of fake news detection using Natural Language Processing (NLP) is to identify and mitigate the spread of misinformation and disinformation within digital content. Here are the key objectives of fake news detection using NLP techniques.**

* **Promoting Information Accuracy**
* **Preserving Credibility**
* **Enhancing Media Literacy**
* **Supporting Fact-Checking Efforts**
* **Curbing Misinformation Spread**
* **Protecting Democratic Processes**
* **Improving Online Discourse**
* **Advancing Research and Development:**

**NLP :**

**Natural Language Processing (NLP) is a branch of artificial intelligence (AI) that focuses on enabling computers to understand, interpret, and generate human language in a way that is both valuable and meaningful. NLP combines computational linguistics, computer science, and cognitive psychology to bridge the gap between human communication and computer understanding.**

* **Tokenization: Breaking down text into smaller units like words or sentences.**
* **Part-of-Speech Tagging: Identifying the grammatical parts of words (nouns, verbs, etc.).**
* **Question Answering: Understanding and generating human-like responses to questions.**
* **Chatbots: Creating computer programs that can conduct a conversation with human users.**
* **Text Summarization: Generating concise summaries from large pieces of text.**

**1. Data Collection:**

* **Gather a diverse and well-labeled dataset containing both real and fake news articles**

**2. Data Preprocessing:**

**Text Cleaning:**

* **Remove special characters, numbers, and punctuation.**
* **Handle capitalization and convert all text to lowercase.**
* **Eliminate URLs and irrelevant white spaces.**

**Tokenization:**

* **Split the text into words or tokens.**

**Removing Stopwords:**

* **Remove common words (e.g.,**

**"and," "the," "is") that don't**

**provide significant meaning.**

**Stemming/Lemmatization:**

* **Reduce words to their root form to normalize variations.**

**3. Feature Extraction:**

**TF-IDF (Term Frequency-Inverse Document Frequency):**

* **Convert text data into numerical vectors, reflecting word importance in documents.**

**Word Embeddings:**

* **Utilize pre-trained word embeddings (Word2Vec, GloVe) to capture semantic relationships.**

**4. Model Selection:**

**Choose an appropriate machine learning or deep learning model for text classification. Common choices include:**

* **Multinomial Naive Bayes: Simple and effective for text classification tasks.**
* **Random Forest: Ensemble learning method capable of handling high-dimensional data.**
* **LSTM (Long Short-Term Memory) or BERT (Bidirectional Encoder Representations from Transformers): Deep learning models capable of capturing complex linguistic patterns.**

**5. Training the Model:**

* **Split the preprocessed dataset into training and testing sets.**
* **Train the selected model on the training data.**

**6. Model Evaluation:**

* **Evaluate the model's performance using metrics such as accuracy, precision, recall, and F1-score on the test data.**
* **Fine-tune hyperparameters and 9adjust the model based on evaluation results.**

**7. Post-processing:**

* **Set a threshold for classifying news articles as fake or real based on model output probabilities.**
* **Implement additional logic to handle ambiguous cases or improve the model's confidence levels.**

**8. Deployment:**

* **Deploy the trained model as part of a web application, API, or any platform where users can input news articles for classification.**

**9. Continuous Monitoring and Updating:**

* **Monitor the model's performance in real-world scenarios.**
* **Regularly update the model with new data and retrain it to ensure it remains effective against evolving fake news strategies.**

**Fake news classification model :**

**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))**

**/kaggle/input/fake-and-real-news-dataset/True.csv**

**/kaggle/input/fake-and-real-news-dataset/Fake.csv**

**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:**

**#import data**

**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]:**

| **S NO** | **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 30, 2017** |
| **2** | **Sheriff David Clarke Becomes An Internet Joke…** | **On Friday, it was revealed that former Milwauk..** | **News** | **december 31, 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]:**

| **S NO** | **TITLE** | **TEXT** | **SUBJECT** | **DATE** |
| --- | --- | --- | --- | --- |
| **0** | **As U.S. budget fight looms, Republicans flip t…** | **WASHINGTON (Reuters) - The head of a conservat…** | **Politics news** | **December 31, 2017** |
| **1** | **U.S. military to accept transgender recruits o…** | **WASHINGTON (Reuters) - Transgender people will…** | **Politics news** | **December 30, 2017** |
| **2** | **Senior U.S. Republican senator: 'Let Mr. Muell…** | **WASHINGTON i(Reuters) - The special counsel inv…** | **Politics news** | **December 31, 2017** |
| **3** | **FBI Russia probe helped by Australian diplomat…** | **WASHINGTON (Reuters) - Trump campaign adviser …** | **Politics news** | **December 27, 2017** |
| **4** | **Trump wants Postal Service to charge 'much mor…** | **SEATTLE/WASHINGTON (Reuters) - President Donal…** | **Politics news** | **December 21, 2017** |

**IN[6]:**

**#adding additional column to separate between 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]:**

| **S NO** | **TITLE** | **TEXT** | **SUBJECT** | **DATE** | **TARGET** | **ORIGINAL** |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | **As U.S. budget fight looms, Republicans flip t…** | **WASHINGTON (Reuters) - The head of a conservat…** | **Politics news** | **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….** | **Politics news** | **December 30, 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…** | **Politics news** | **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 …** | **Politics news** | **December 25, 2017** | **1** | **Trump wants Postal Service to charge 'much mor…** |
| **4** | **Trump wants Postal Service to charge 'much mor…** | **SEATTLE/WASHINGTON (Reuters) - President Donal…** | **Politics news** | **December 29, 2017** | **1** | **FBI Russia probe helped by Australian diplomat…** |

**IN[7]:**

**df.isnull().sum()**

**OUT[7]:**

**title 0**

**text 0**

**subject 0**

**date 0**

**target 0**

**original 0**

**dtype: int64**

**Data Clean up:**

**create a function here that will be responsible to remove any unnecessary words (Stopwords) from the data provided**

**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 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')**

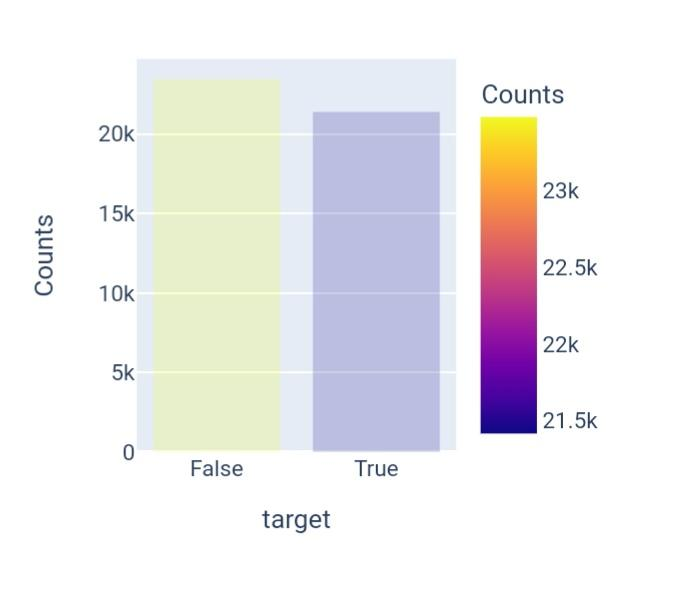
**sub\_tf\_df.target.replace({0:'False',1:'True'},inplace=True)**

**fig = px.bar(sub\_tf\_df, x="target", y="Counts",**

**color='Counts', barmode='group',**

**height=350)**

**fig.show()**

****

**● The data looks balanced and no issues on building the model**

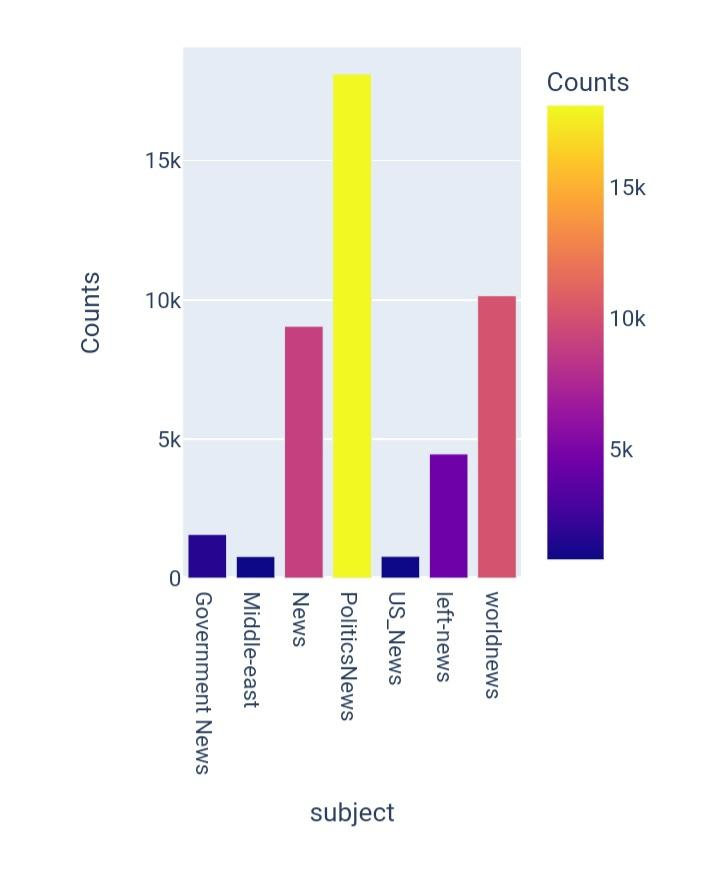
**IN[11]:**

**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()**

**Count of news articles by subjects:**

****

**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]:**

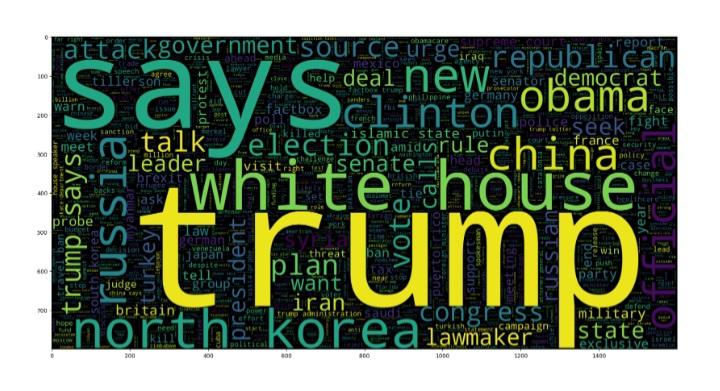
**plt.figure(figsize = (20,20))**

**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)):**

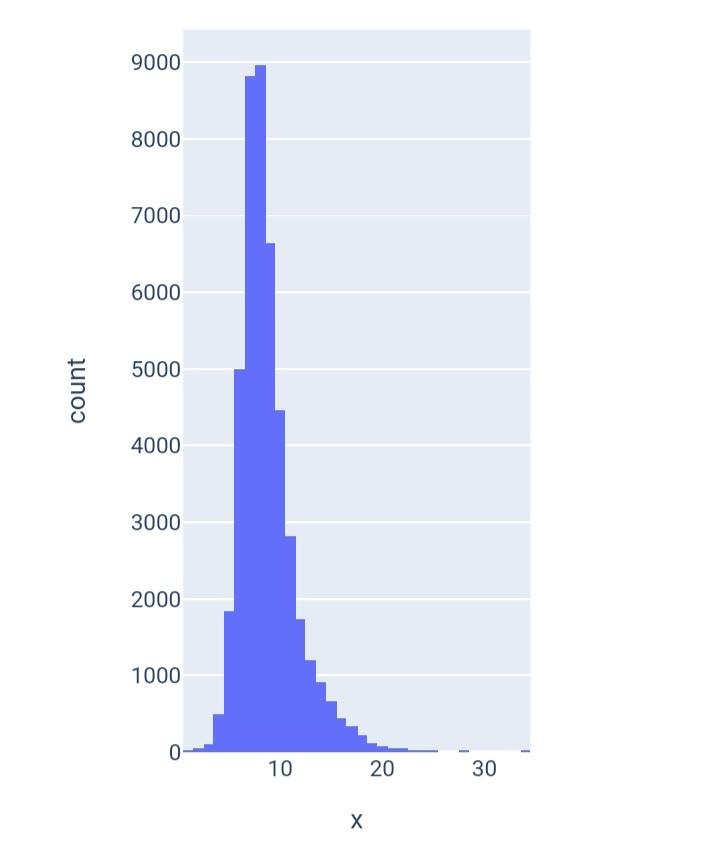
**maxlen = len(tokens)**

**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**

**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]:**

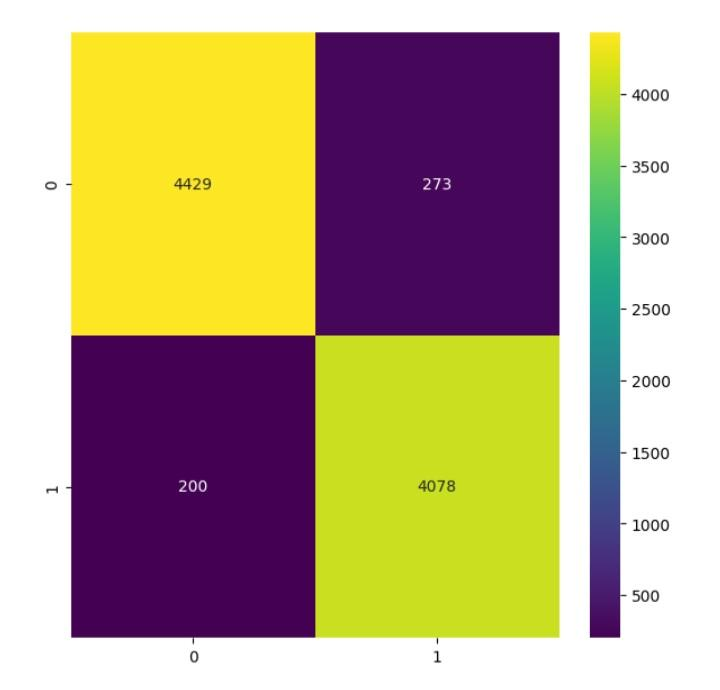
**cm = confusion\_matrix(list(y\_test), predicted\_value)**

**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**

**Checking the content of news:**

**IN[19]:**

**df['clean\_text'] = df['text'].apply(preprocess)**

**df['clean\_joined\_text']=df['clean\_text'].apply(lambda x:" ".join(x))**

**IN[20]:**

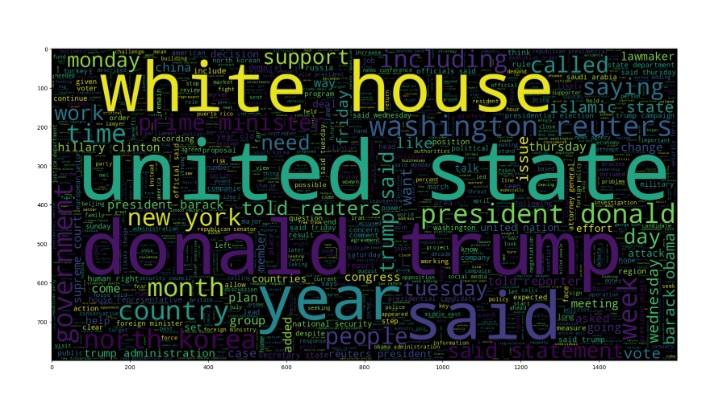
**plt.figure(figsize = (20,20))**

**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)):**

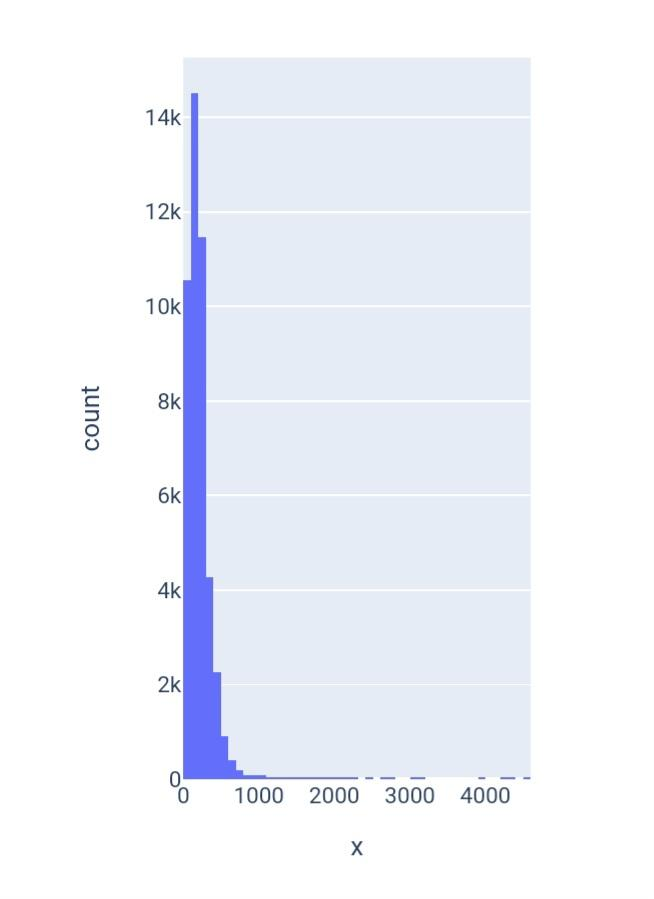
**maxlen = len(tokens)**

**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**

**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:**

**IN[23]:**

**prediction = []**

**for i in range(len(predicted\_value)):**

**if predicted\_value[i].item() > 0.5:**

**prediction.append(1)**

**else:**

**prediction.append(0)**

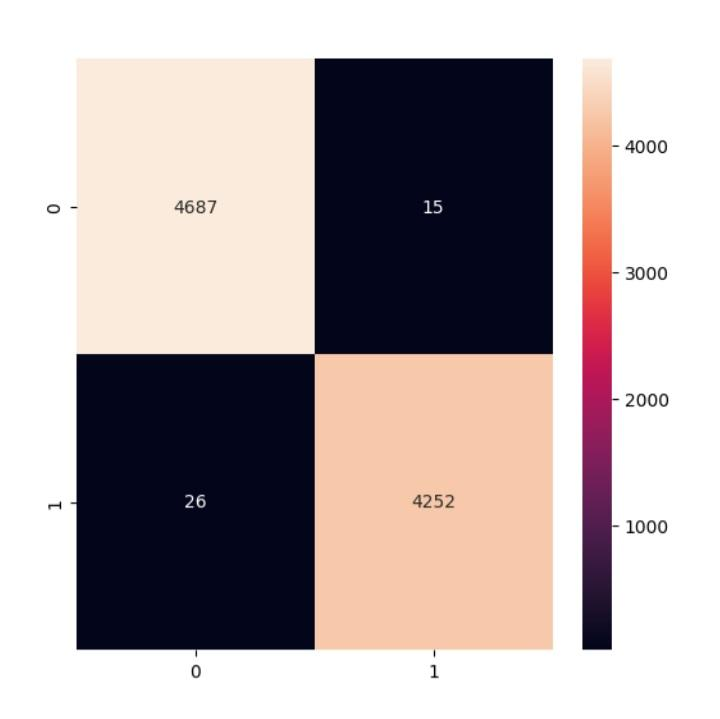
**cm = confusion\_matrix(list(y\_test), prediction)**

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

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

**OUT[23]:**

**<Axes: >**

****

**CONCLUSION :**

**employing Natural Language Processing (NLP) techniques for fake news detection is a promising and vital area of research. By leveraging advanced algorithms and machine learning models, NLP can effectively analyze textual content to identify misleading or fabricated information. Researchers and practitioners in the field continue to refine and develop innovative methods to enhance the accuracy and efficiency of fake news detection systems. As technology evolves, the application of NLP in this context will play a crucial role in combating the spread of misinformation, ensuring a more reliable and trustworthy information environment for all.**