

```
from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

```
# 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 5GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
#loading the data
fake=pd.read_csv("/content/drive/MyDrive/Classroom/B.Tech II year/Fake.csv")
true=pd.read_csv("/content/drive/MyDrive/Classroom/True.csv")
```

```
#Creating a category for whether fake or not
#where 1 stand for fake news and 0 stands for true news
```

```
fake["category"]=1
true["category"]=0
```

```
#joining the data the two data frame and resetting index
df=pd.concat([fake,true]).reset_index(drop=True)
df.head()
```

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

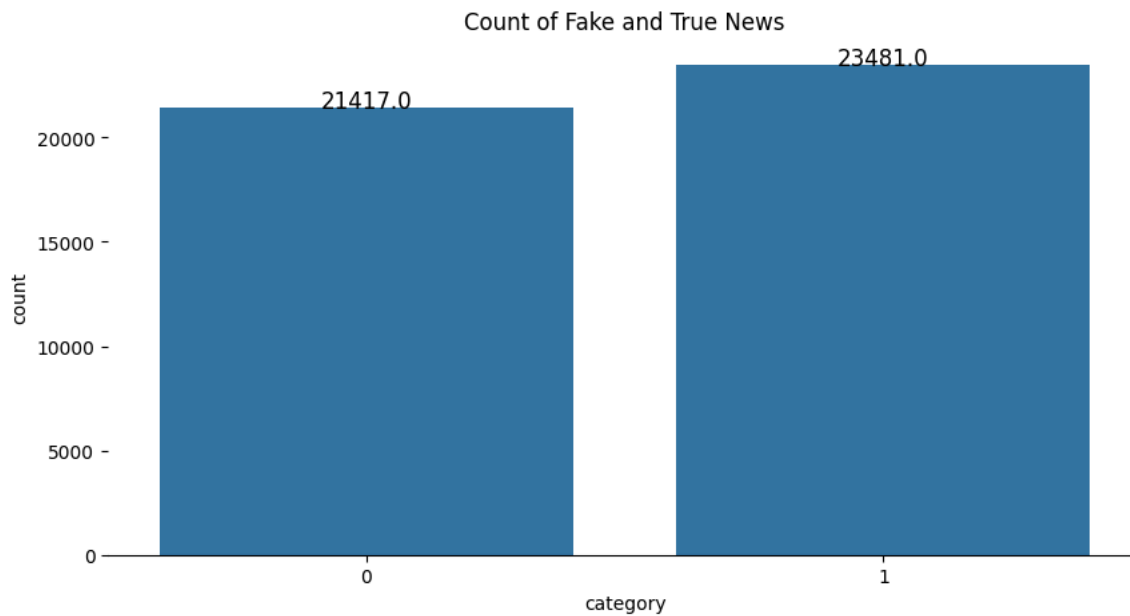
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

```
#creating a count plot for category column
fig = plt.figure(figsize=(10,5))
```

```
graph = sns.countplot(x="category", data=df)
plt.title("Count of Fake and True News")
```

```
#removing boundary
graph.spines["right"].set_visible(False)
graph.spines["top"].set_visible(False)
graph.spines["left"].set_visible(False)
```

```
#annotating bars with the counts
for p in graph.patches:
    height = p.get_height()
    graph.text(p.get_x()+p.get_width()/2., height + 0.2,height ,ha="center",fontsize=12)
```

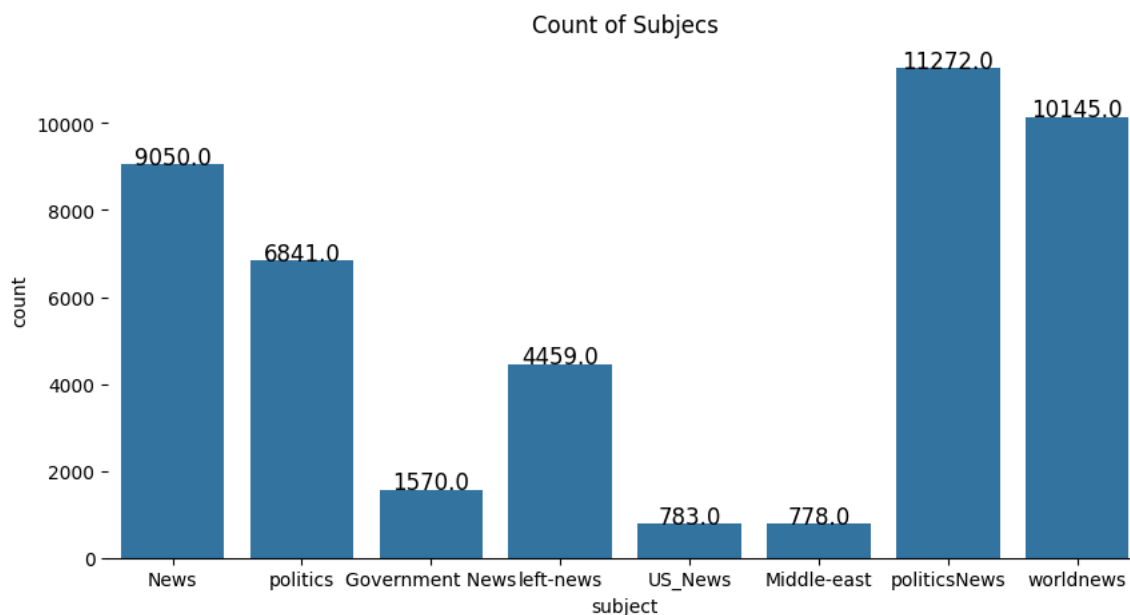


```
#creating a count plot for subject column
fig = plt.figure(figsize=(10,5))
```

```
graph = sns.countplot(x="subject", data=df)
plt.title("Count of Subjecs")
```

```
#removing boundary
graph.spines["right"].set_visible(False)
graph.spines["top"].set_visible(False)
graph.spines["left"].set_visible(False)
```

```
#annotating bars with the counts
for p in graph.patches:
    height = p.get_height()
    graph.text(p.get_x()+p.get_width()/2., height + 0.2,height ,ha="center",fontsize=12)
```



```
#checking the missing values in each columns
df.isna().sum()*100/len(df)
```

```

0
title 0.0
text 0.0
subject 0.0
date 0.0
category 0.0

```

```

#checking if there is empty string in TEXT column
blanks=[]

#index, label and review of the doc
for index, text in df["text"].items(): # it will iter through index, label and review
    if text.isspace(): # if there is a space
        blanks.append(index) #it will be noted down in empty list

```

```
len(blanks)
```

```
0
```

```
#instead of dropping these values we are going to merge title with text
```

```
df["text"] = df["title"] + df["text"]
```

```
#we only need two columns rest can be ignored
```

```
df = df[["text", "category"]]
```

```
#importing libraries for cleaning puprose
```

```

from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import spacy
import re

```

Stopwords

A stop word is a commonly used word (such as "the", "a", "an", "in") that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. We would not want these words to take up space in our database, or taking up valuable processing time. For this, we can remove them easily, by storing a list of words that you consider to stop words. NLTK (Natural Language Toolkit) in python has a list of stopwords stored in 16 different languages. You can find them in the nltk_data directory. home/pratima/nltk_data/corpora/stopwords is the directory address. (Do not forget to change your home directory name)

Lemmatisation

lemmatization looks beyond word reduction, and considers a language's full vocabulary to apply a morphological analysis to words. The lemma of 'was' is 'be' and the lemma of 'mice' is 'mouse'. Further, the lemma of 'meeting' might be 'meet' or 'meeting' depending on its use in a sentence.

```

#loading spacy library
nlp = spacy.load("en_core_web_sm")

#creating instance
lemma = WordNetLemmatizer()

#importing libraries for cleaning puprose

from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import spacy
import re
import nltk # Import nltk

# Download the stopwords resource
nltk.download('stopwords')

#loading spacy library
nlp = spacy.load("en_core_web_sm")

#creating instance

```

```

lemma=WordNetLemmatizer()

#creating list of stopwords containing stopwords from spacy and nltk

#stopwords of spacy
list1=nlp.Defaults.stop_words
print(len(list1))

#stopwords of NLTK
list2=stopwords.words('english')
print(len(list2))

#combining the stopword list
Stopwords=set((set(list1)|set(list2)))
print(len(Stopwords))

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Unzipping corpora/stopwords.zip.
326
198
401

#text cleaning function
def clean_text(text):

    """
    It takes text as an input and clean it by applying several methods

    """

    string = ""

    #lower casing
    text=text.lower()

    #simplifying text
    text=re.sub(r"i'm","i am",text)
    text=re.sub(r"he's","he is",text)
    text=re.sub(r"she's","she is",text)
    text=re.sub(r"that's","that is",text)
    text=re.sub(r"what's","what is",text)
    text=re.sub(r"where's","where is",text)
    text=re.sub(r"\ll"," will",text)
    text=re.sub(r"\ve"," have",text)
    text=re.sub(r"\re"," are",text)
    text=re.sub(r"\d"," would",text)
    text=re.sub(r"won't","will not",text)
    text=re.sub(r"can't","cannot",text)

    #removing any special character
    text=re.sub(r"[-()\#!@$%&*&{}?.,:]", " ",text)
    text=re.sub(r"s+", " ",text)
    text=re.sub('[^A-Za-z0-9]+',' ', text)

    for word in text.split():
        if word not in Stopwords:
            string+=lemma.lemmatize(word)+" "

    return string

#cleaning the whole data
df["text"]=df["text"].apply(clean_text)

```

✓ Word Cloud


A word cloud is a collection, or cluster, of words depicted in different sizes. The bigger and bolder the word appears, the more often it's mentioned within a given text and the more important it is.

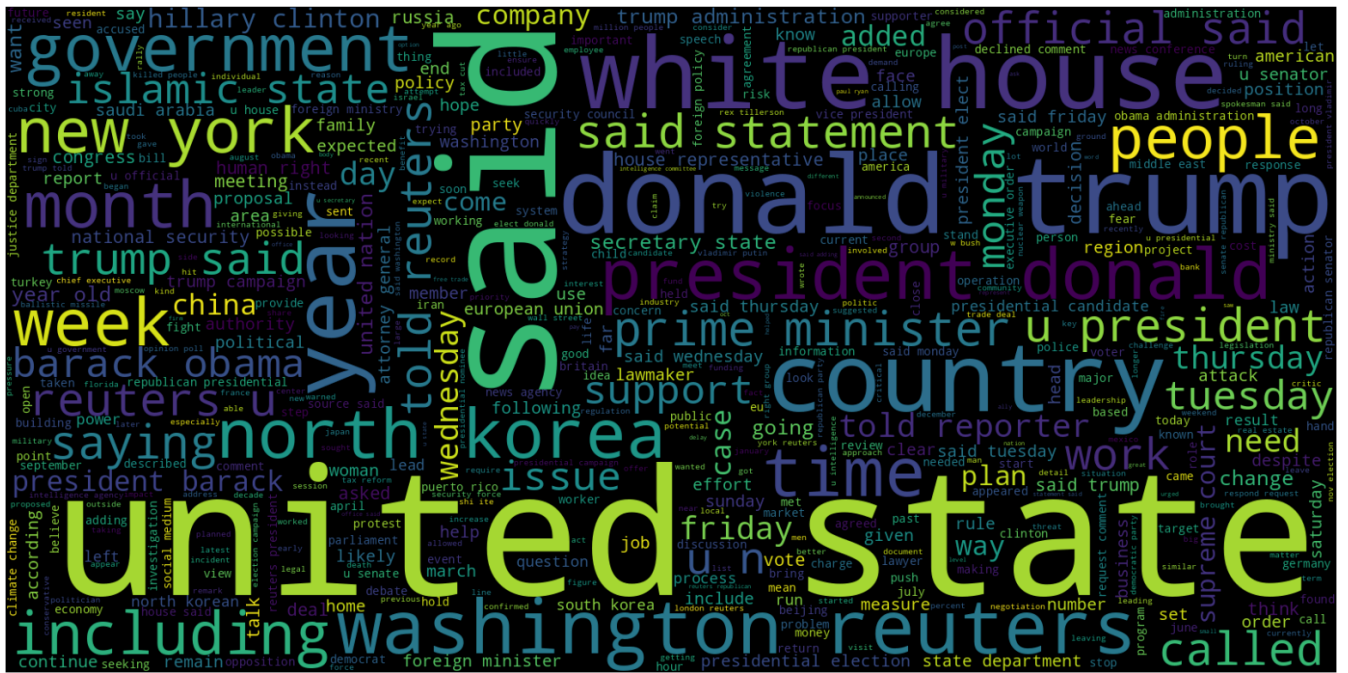
```

from wordcloud import WordCloud

#True News
plt.figure(figsize = (20,20))
Wc = WordCloud(max_words = 500 , width = 1600 , height = 800).generate(" ".join(df[df.category == 0].text))
plt.axis("off")
plt.imshow(Wc , interpolation = 'bilinear')

```

 <matplotlib.image.AxesImage at 0x7c7839edc150>



#creating more intuitive wordcloud


#pil is pillow and used for image manipulation
from PIL import Image

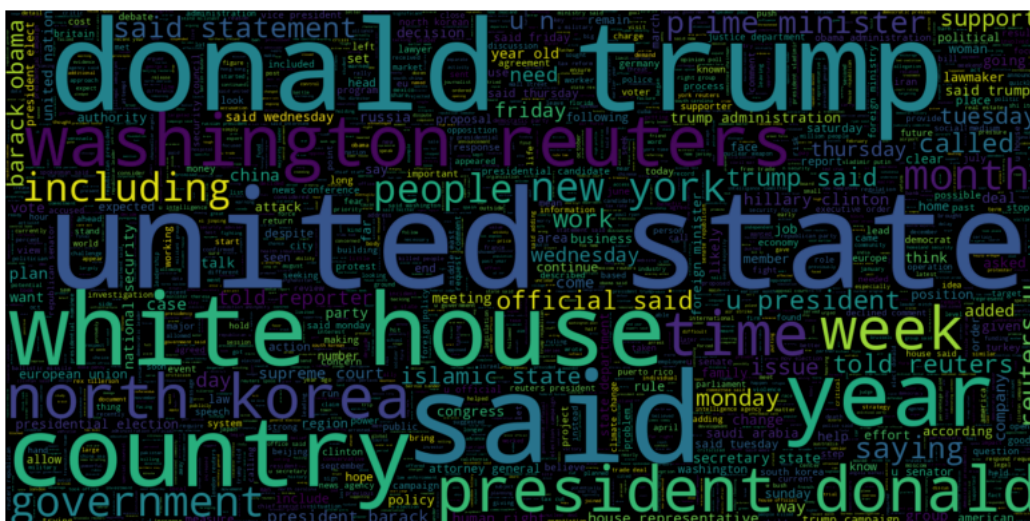
#True News

plt.figure(figsize = (10,10))

Removed the mask parameter since no mask image is available
Wc = WordCloud(max_words = 2000 , width = 1600 ,
height = 800)

Wc.generate(" ".join(df[df.category == 0].text))
plt.axis("off")
plt.imshow(Wc , interpolation = 'bilinear')

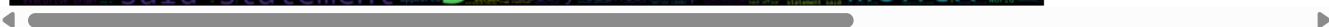
 <matplotlib.image.AxesImage at 0x7c783758d510>




```
→ <matplotlib.image.AxesImage at 0x7c783b2b0050>
```



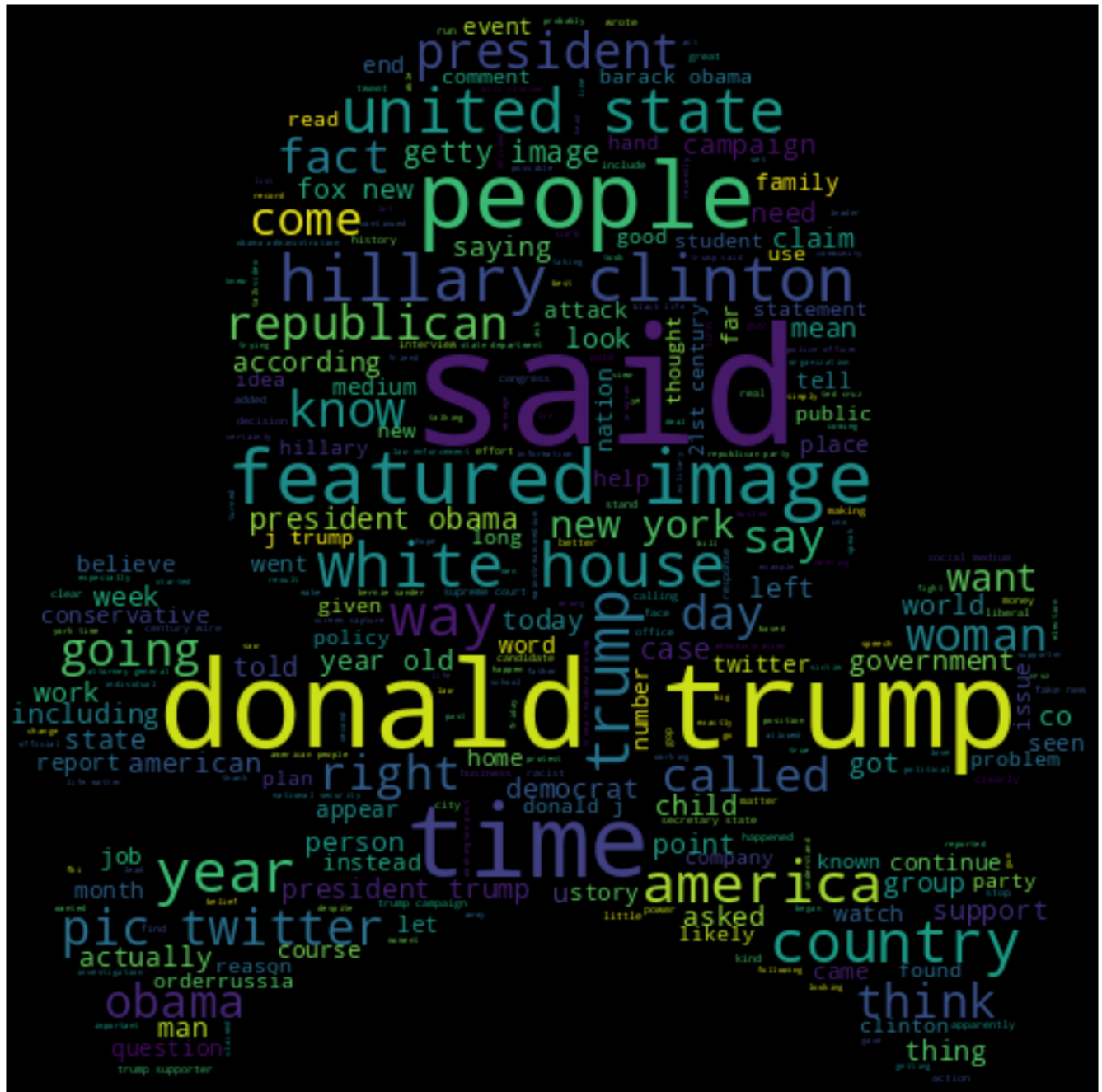
```
↵ <matplotlib.image.AxesImage at 0x7c784832ad10>
```



```
skull="/content/drive/MyDrive/Classroom/B.Tech II year/death.png"
icon=Image.open(skull)
```



```
→ <matplotlib.image.AxesImage at 0x7c783c8d4510>
```



- Feature-Extraction & Model building

```
#splitting the
from sklearn.model_selection import train_test_split

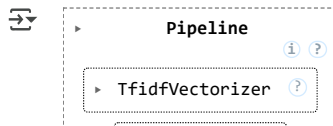
X=df["text"] #feature
y=df["category"] # target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)

#importing libraries to build a pipeline
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import LinearSVC

#this pipe line will take the text and vectorise it , and then TF-IDF, then fitting the model

text_clf=Pipeline([("tfidf",TfidfVectorizer()),("clf",LinearSVC())])
text_clf.fit(X_train,y_train)
```

```
#making prediction using the model
predictions=text_clf.predict(X_test)
```

```
from sklearn import metrics
print(metrics.classification_report(y_test,predictions))
```

```

precision    recall  f1-score   support

0           0.99      0.99      0.99         7039
1           0.99      0.99      0.99         7778

accuracy          0.99      14817
macro avg          0.99      14817
weighted avg       0.99      14817
  
```

```
#overall acuracy
print(metrics.accuracy_score(y_test,predictions))
```

```
0.992643585071202
```

```
#confusion matrix
print(metrics.confusion_matrix(y_test,predictions))
```

```
[[6993  46]
 [ 63 7715]]
```

```
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
```

```
# Get scores for ROC using the trained pipeline 'text_clf'
y_scores = text_clf.decision_function(X_test)
```

```

# Compute ROC curve and ROC area
# Note: The pos_label should be one of the actual class labels in y_test (0 or 1).
# Since 0 represents True News and 1 represents Fake News, and the goal is
# likely to evaluate the model's ability to identify Fake News,
# we'll set pos_label to 1.
fpr, tpr, thresholds = roc_curve(y_test, y_scores, pos_label=1)
roc_auc = auc(fpr, tpr)
  
```

```

# Plot ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='blue', lw=2, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='grey', lw=1, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.grid()
plt.show()
  
```

