NAAN MUDHALVAN PROJECT(IBM) IBM AI 101 ARTIFICIAL INTELLIGENCE-GROUP 1 PROJECT: TEAM-5

FAKE NEWS DETECTION USING NLP TEAM MEMBERS

1)BALA.PK (reg.no: 113321106008)

2)DARANISHWAR. GR (reg no. 113321106016)

3)DEEPAN SANKAR. J (reg no. 113321106017)

4)FERIN KINGSLY.M (reg no. 113321106026)

5)HARI RAGAAV.D(reg no. 113321106031)

About Dataset :

A dataset is a structured collection of data points or observations that are organized in a way that each data point corresponds to a distinct entity, event, or record, and each attribute or feature describes a specific characteristic of that entity. In simpler terms, a dataset is a set of information about a particular subject or a group of related subjects.

This data set consists of 40000 fake and real news. Our goal is to train our model to accurately predict whether a particular piece of news is real or fake. Fake and real news data are given in two separate data sets, with each data set consisting of approximately 20000 articles.

DATA ACQUISITION:

Collecting data from a variety of sources, including news websites, social media platforms, and other online media, as part of the data acquisition process for fake news detection using NLP.

DATA PREPROCESSING:

The process of cleaning the data by removing irrelevant information, such as stop words, lowercasing the words punctuations,and special characters.

Modeling:

1. **Data Collection**: Gather real and fake news articles.

2. **Data Preprocessing**: Clean and prepare the text data.

3. **Feature Extraction**: Convert text into numerical form (e.g., TF-IDF, embeddings).

4. **Model Selection**: Choose a suitable ML model (e.g., Naive Bayes, SVM, deep learning).

5. **Model Training:** Train the chosen model on your labeled data.

6. **Evaluation:** Assess model performance using metrics like accuracy, precision, recall.

7. **Handle Imbalance**: Address class imbalance issues in the dataset.

8. **Hyperparameter Tuning:** Optimize model settings for better performance.

9. **Ensemble Methods**: Combine models for improved results (optional).

10. **Interpretability**: Ensure the model's decisions are understandable.

11. **Continuous Monitoring**: Regularly update the model with new data.

12. **Ethical Considerations**: Be aware of biases and ethical implications.

13. **Deployment:** Implement the model in your desired application.

LOADING THE DATA SET

LOAD REQUIRED LIBRARIES:

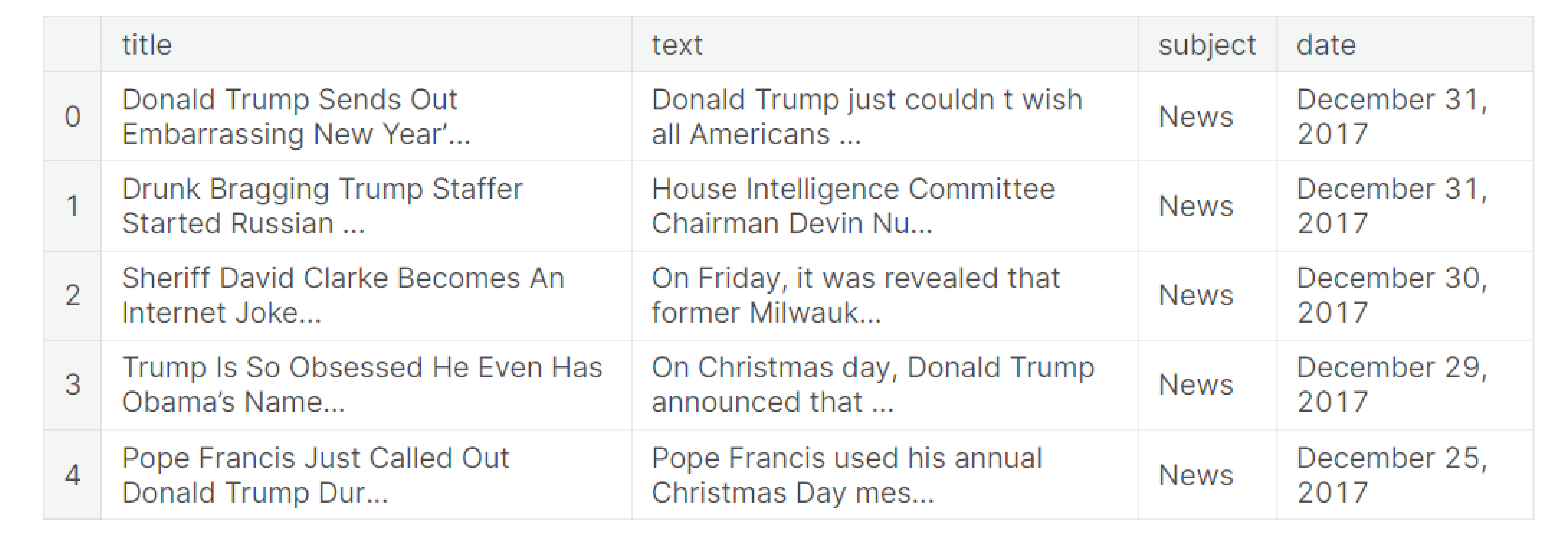
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns

from bs4 import BeautifulSoup
import re
import string
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from wordcloud import WordCloud, STOPWORDS
from nltk.tokenize import word_tokenize
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.feature_extraction.text import TfidfVectorizer
```

IMPORT THE DATA SET

Input 1:

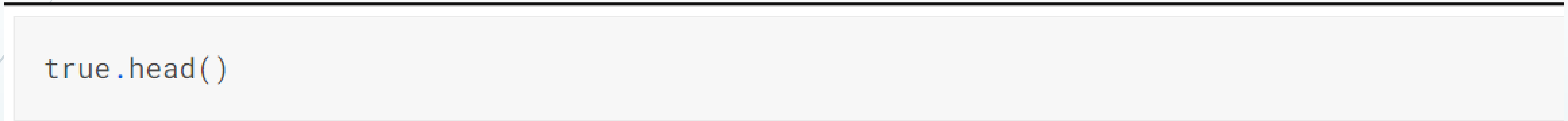
```
#import dataset
fake = pd.read_csv("../input/fake-and-real-news-dataset/Fake.csv")
true = pd.read_csv("../input/fake-and-real-news-dataset/True.csv")

#data exploration
fake.head()
```

Output 1:



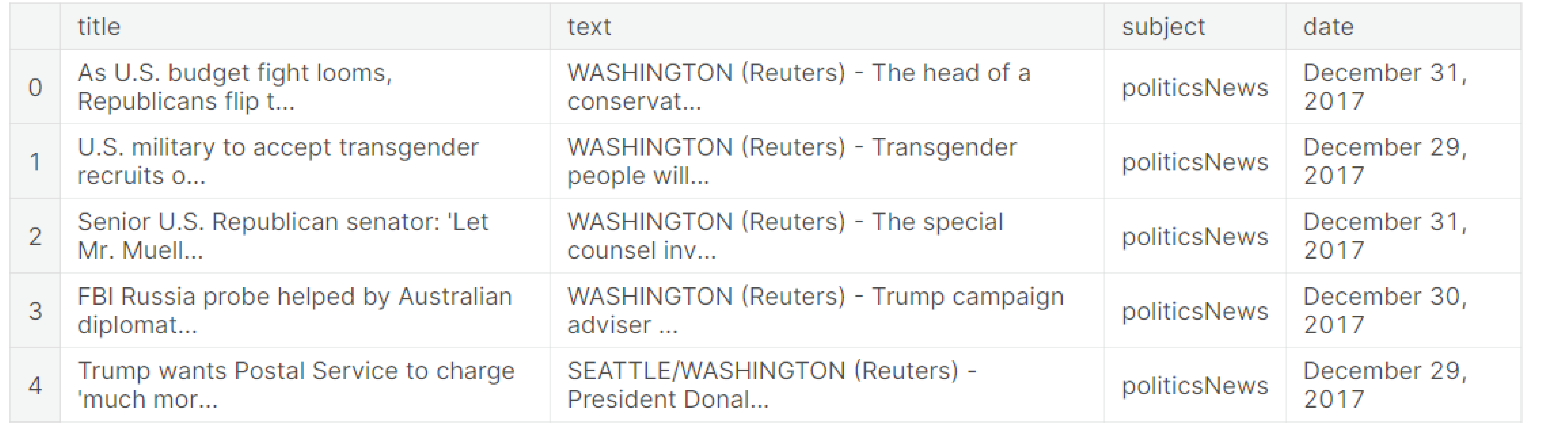
	title	text	subject	date
0	Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017
2	Sheriff David Clarke Becomes 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

INPUT 2:



```
true.head()
```

OUTPUT 2:

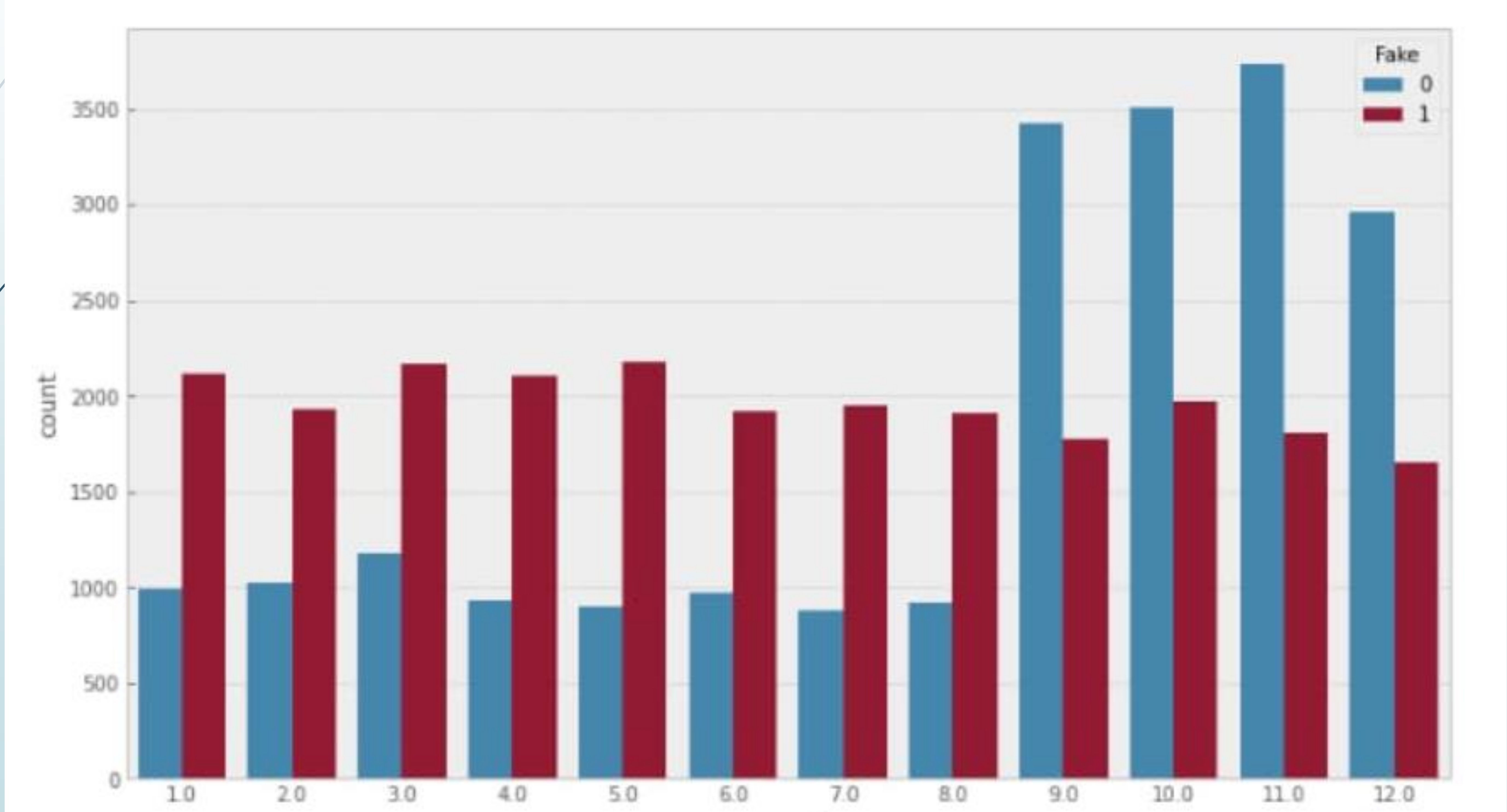


	title	text	subject	date
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017
2	Senior U.S. 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

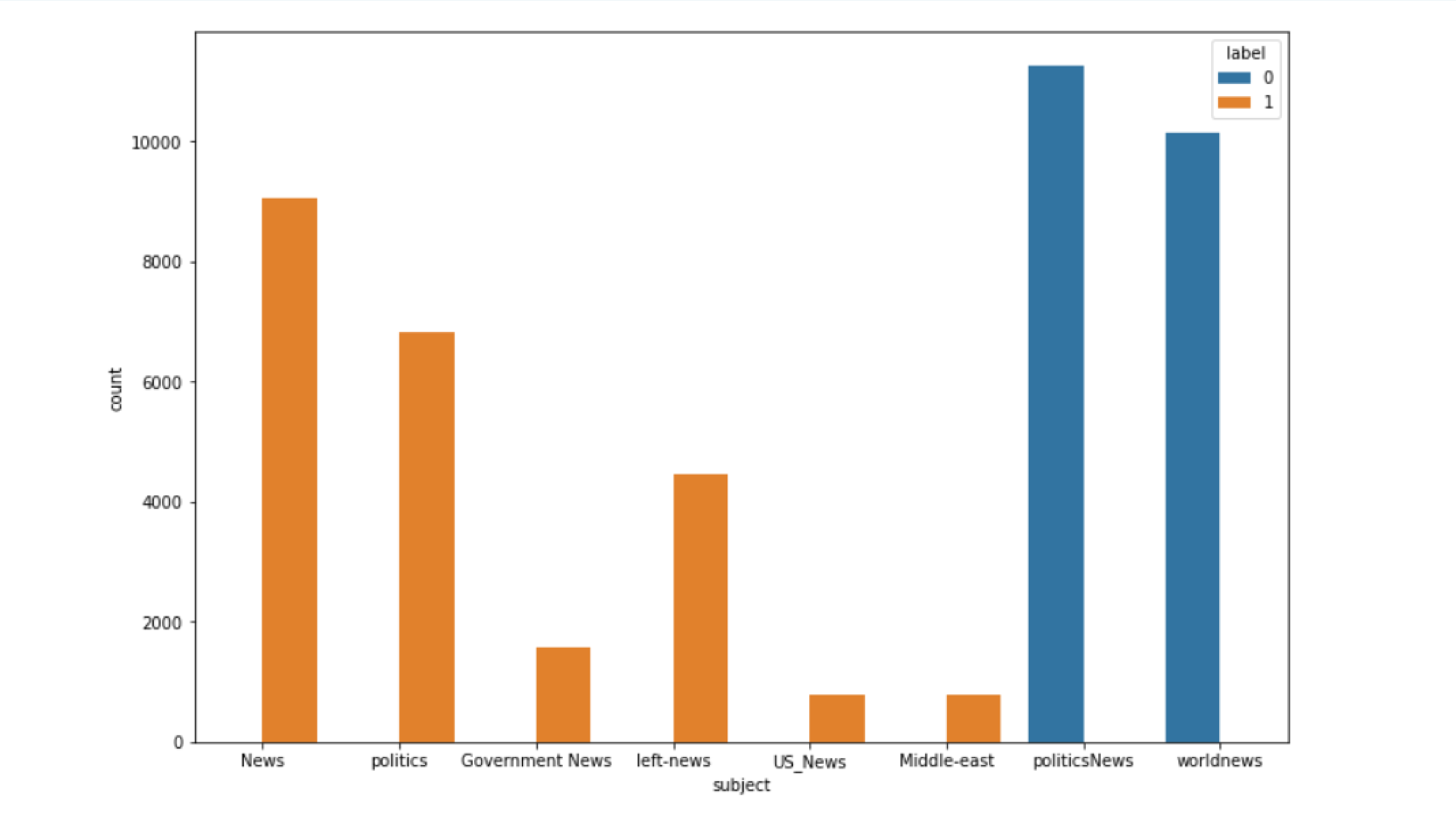
```
fake["label"] = 1
true["label"] = 0
```

DATA PREPROCESSING:

OUTPUT GRAPH:



```
plt.figure(figsize=(12,8))
sns.countplot(x = "subject", data=df, hue =
"label")
plt.show()
```



ALGORITHM FOR NLP:

- Gather diverse real and fake news articles.
- Convert text to numerical features for models.Use techniques like TF-IDF or Word Embeddings.
- Split into training and testing sets.
- Choose model: ML (e.g., SVM) or DL (e.g., LSTM). Consider NLP-specific models like BERT.
- Train chosen model on training data.Use labeled data for supervised learning.
- Assess model's performance on test data. Evaluate using metrics like accuracy, precision, recall.

- Optimize model parameters . Fine-tune for better performance.
- Address uneven real vs fake class distribution. Apply techniques like oversampling or undersampling.
- Combine multiple models for improved accuracy. Use techniques like voting or bagging.
- Implement model for real-time fake news detection.
- Continuously check model's performance. Update with new data or improved algorithms.
- Add metadata, social media data, etc. for accuracy.Enhance model with additional relevant features.

TEXT PREPROCESSING:

DATA CLEANING:

Data cleaning is a very crucial step in any machine learning model, but more so for NLP. Without the cleaning process, the dataset is often a cluster of words that the computer doesn’t understand. Here, we will go over steps done in a typical machine learning text pipeline to clean data.

```
#data cleaning
#combining the title and text columns
df['text'] = df['title'] + " " + df['text']
#deleting few columns from the data
del df['title']
del df['subject']
del df['date']
```

MISSING VALUES:

- Data cleaning is a very crucial step in any machine learning model, but more so for NLP.
- Without the cleaning process, the dataset is often a cluster of words that the computer doesn’t understand.
- Here, It will go over steps done in a typical machine learning text pipeline to clean data.

PROGRAM:

```
#data cleaning
#combining the title and text
columns df['text'] = df['title'] + " " +
df['text'] #deleting few columns
from the data del df['title']
del
df['subject']
del df['date']
df.head()
```

WORD CLOUD:

Fake News Word Cloud:

```
#word cloud for fake news
cloud = WordCloud(max_words = 500, stopwords = STOPWORDS,
background_color = "white").generate(" ".join(df[df.label == 1].text))
plt.figure(figsize=(40, 30))
plt.imshow(cloud, interpolation="bilinear")
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```

Real News Word Cloud:

```
#word cloud for real news
cloud = WordCloud(max_words = 500, stopwords = STOPWORDS,
background_color = "white").generate(" ".join(df[df.label == 0].text))
plt.figure(figsize=(40, 30))
plt.imshow(cloud, interpolation="bilinear")
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```

PROGRAM FOR TRAINING LSTM MODEL:

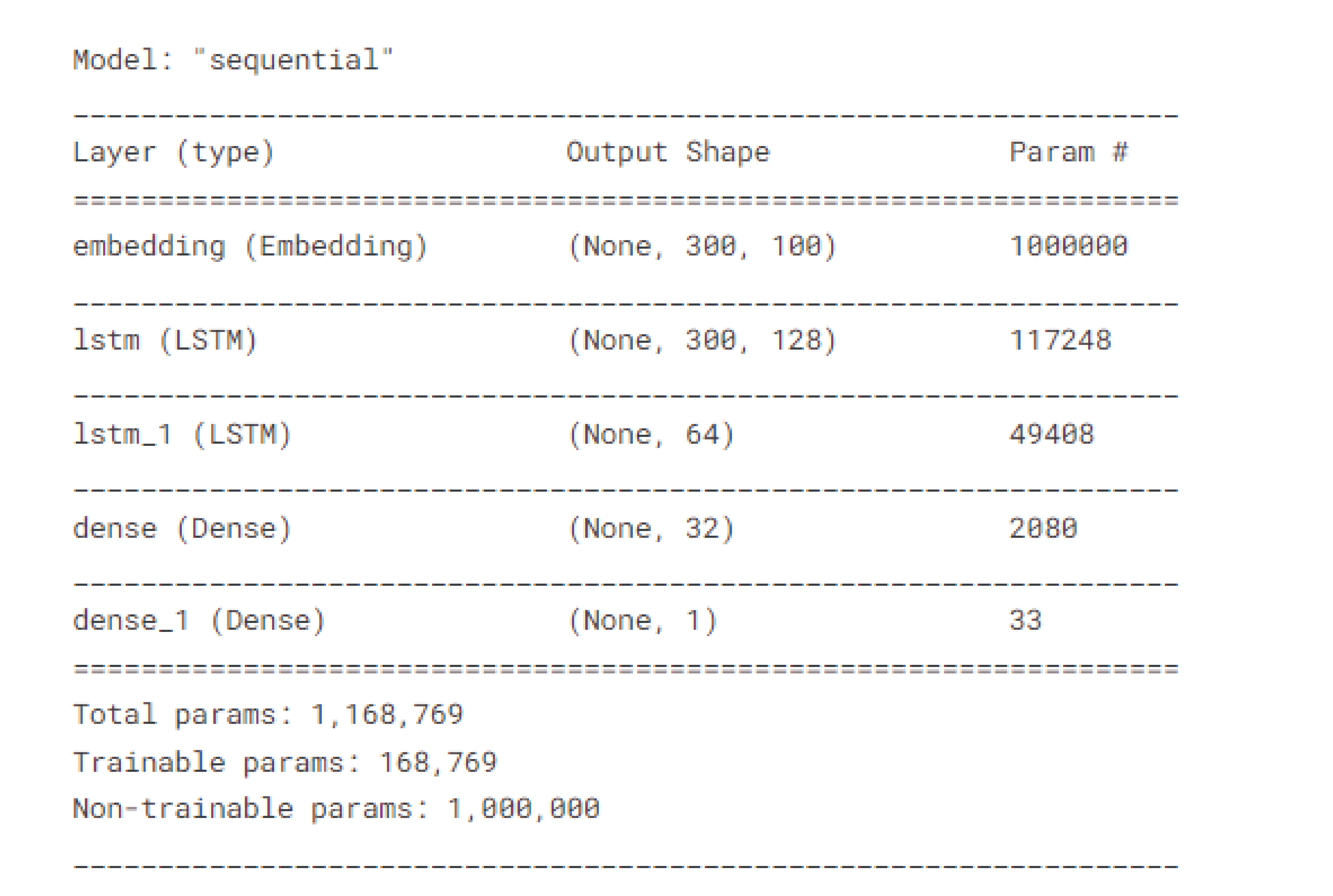
```
batch_size = 256
epochs = 10
embed_size = 100
model = Sequential()

#Non-trainable embeddidng layer
model.add(Embedding(max_features, output_dim=embed_size, input_length=maxlen, trainable=False))

#LSTM
model.add(LSTM(units=128 , return_sequences = True , recurrent_dropout = 0.25 , dropout
= 0.25)) model.add(LSTM(units=64 , recurrent_dropout = 0.1 , dropout = 0.1))
model.add(Dense(units = 32 , activation =
'relu')) model.add(Dense(1,
activation='sigmoid'))
model.compile(optimizer=keras.optimizers.Adam(lr = 0.01), loss='binary_crossentropy', metrics=['accuracy'])
history = model.fit(X_train, y_train, validation_split=0.3, epochs=10, batch_size=batch_size, shuffle=True,
verbose = 1)
```

MODEL AFTER TRAINING WITH LSTM ALGORITHM:

CODE:

```
model.summary()
```

OUTPUT:



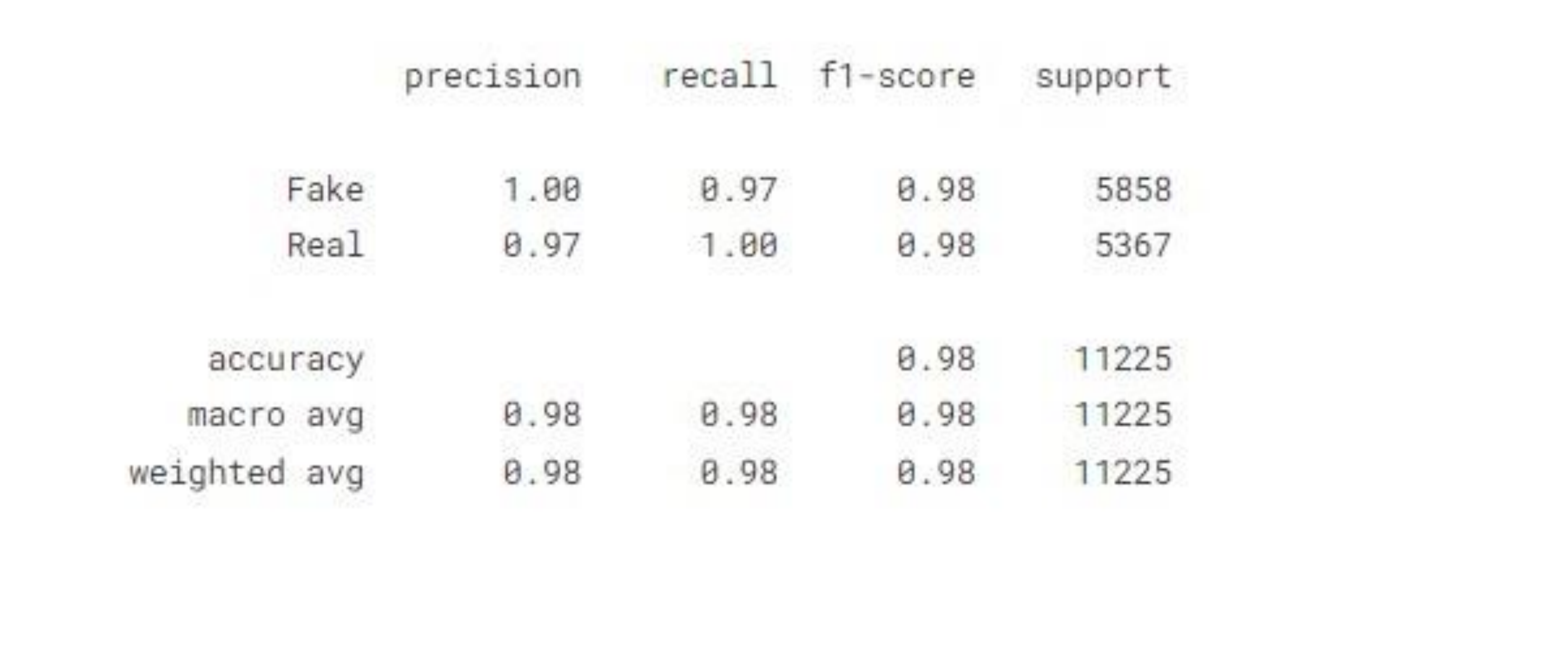
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 300, 100)          1000000
_________________________________________________________________
lstm (LSTM)                  (None, 300, 128)          117248
_________________________________________________________________
lstm_1 (LSTM)                (None, 64)                49408
_________________________________________________________________
dense (Dense)                (None, 32)                2080
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 33
=================================================================
Total params: 1,168,769
Trainable params: 168,769
Non-trainable params: 1,000,000
_________________________________________________________________
```

MODEL ANALYSIS TRAINING:

CODE:

```
pred = model.predict_classes(X_test)
print(classification_report(y_test, pred, target_names = ['Fake','Real']
```

OUTPUT:



```
              precision    recall  f1-score   support

        Fake       1.00      0.97      0.98      5858
        Real       0.97      1.00      0.98      5367

    accuracy                           0.98     11225
   macro avg       0.98      0.98      0.98     11225
weighted avg       0.98      0.98      0.98     11225
```

Feature Extraction:

1. Bag-of-Words (BoW)
2. Term Frequency-Inverse Document Frequency (TF-IDF)
3. N-grams
4. Word Embeddings (e.g., Word2Vec, GloVe)
5. Part-of-Speech (POS) Tags
6. Named Entity Recognition (NER)
7. Sentiment Analysis
8. Readability Scores (e.g., Flesch-Kincaid, Gunning Fog Index)
9. Syntactic Features (e.g., Dependency Trees)
10. Linguistic Stylistic Features (e.g., Tone, Formality)
11. Topic Modeling (e.g., Latent Dirichlet Allocation)
12. Named Entity Density

N gram analysis :

Unigram Analysis (1-gram):

In unigram analysis, each word is considered independently without any regard to its neighboring words. This is the simplest form of n gram analysis.

Bigram Analysis (2-gram):

Bigram analysis considers pairs of consecutive words. This type of analysis capture some level of context.

Trigram Analysis (3-gram):

Trigram analysis looks at sequences of three consecutive words. This provides a bit more context compared to bigrams.

Character N-grams:

Instead of words, characters can be considered as units. Character-level n-grams capture patterns of characters, which can be useful for detecting fake news with non-standard language, such as misspellings or deliberate character substitutions.

Skip-grams:

Skip-grams involve predicting the context (surrounding words) of a given word. They can be a more complex form of analysis that captures both local and global context.

THANK YOU