CSE4069- Social Media Analytics

Project Report

FAKE NEWS ANALYSIS

By

TEAM-21

B. PHANINDRA SAI-19MIA1065

O. NAGA SAI KUMAR-19MIA1071

M. JAY KUMAR PATEL-19MIA1032

T. SIVA NIKHIL-19MIA1086

Submitted to

Dr. Priyadarshini R

Assistant Professor

SCOPE, VIT Chennai

School of Computer Science and Engineering

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

APRIL 2023

ABSTRACT

Social media for news consumption is a double-edged sword. On the one hand, its low cost, easy access, and rapid dissemination of information lead people to seek out and consume news from social media.

On the other hand, it enables the wide spread of "fake news", i.e., low quality news with intentionally false information. The extensive spread of fake news has the potential for extremely negative impacts on individuals and society.

Therefore, fake news detection on social media has recently become emerging research that is attracting tremendous attention. Fake news detection on social media presents unique characteristics and challenges that make existing detection algorithms from traditional news media ineffective or not applicable.

First, fake news is intentionally written to mislead readers to believe false information, which makes it difficult and nontrivial to detect based on news content; therefore, we need to include auxiliary information, such as user social engagements on social media, to help make a determination.

Second, exploiting this auxiliary information is challenging in and of itself as users' social engagements with fake news produce data that is big, incomplete, unstructured, and noisy.

Because the issue of fake news detection on social media is both challenging and relevant, we conducted this survey to further facilitate research on the problem.

In this survey, we present a comprehensive review of detecting fake news on social media, including fake news characterizations on psychology and social theories, existing algorithms from a data mining perspective, evaluation metrics and representative datasets.

We also discuss related research areas, open problems, and future research directions for fake news detection on social media

INTRODUCTION

News medium has become a channel to pass on the information of what's happening on world to the people living. Often people perceive whatever conveyed in the news to be true.

There were circumstances where even the news channels acknowledged that their news is not true as they wrote. But some news has a significant impact not only to the people or government but also the economy.

One news can shift the curves up and down depending on the emotions of people and political situation. It is important to identify the fake news from the real true news.

The problem has been taken over and resolved with the help of Natural Language Processing tools which help us identify fake or true news based on the historical data. The news is now in safe hands!

PROBLEM STATEMENT

The authenticity of Information has become a longstanding issue affecting businesses and society, both for printed and digital media.

On social networks, the reach and effects of information spread occur at such a fast pace and so amplified that distorted, inaccurate or false information acquires a tremendous potential to cause real world impacts, within minutes, for millions of users.

Recently, several public concerns about this problem and some approaches to mitigate the problem were expressed.

The sensationalism of not-so-accurate eye catching and intriguing headlines aimed at retaining the attention of audiences to sell information has persisted all throughout the history of all kinds of information broadcast.

On social networking websites, the reach and effects of information spread are however significantly amplified and occur at such a fast pace, that distorted, inaccurate or false information acquires a tremendous potential to cause real impacts, within minutes, for millions of users.

Literature Survey:

TITLE	AUTHOR/JOURNAL NAME/YEAR	METHODS/TECHNIQUE	RESULT
Supervised Learning for Fake News Detection	Julio C. S. Reis, Andre Correia, Fabrıcio Murai	k-Nearest Neighbors (KNN), Naive Bayes (NB), Random Forests (RF), Support Vector Machine with RBF kernel (SVM), and XGBoost (XGB).	KNN AUC 0.8+-0.009 NB AUC 0.72+-0.09 RF AUC 0.85+-0.007 SVM AUC 0.79+-0.030 XGB AUC 0.86+-0.006
Fake News Detection Using Machine Learning Ensemble Methods	Iftikhar Ahmad ,Muhammad Yousaf,Suhail Yousaf	Logistic regression, Linear SVM, Multilayer perceptron, Random forest	Logistic regression (LR) 0.97 Linear SVM (LSVM) 0.98 Multilayer perceptron 0.98 Random forest (RF) 0.99
SpotFake: A Multi-modal Framework for Fake News Detection	Shivangi Singhal, Rajiv Ratn Shah, Tanmoy Chakraborty	RNN, EANN, SpotFake, EANN, MVAE	RNN 0.664, EANN 0.715, SpotFake 0.7777, EANN 0.827, MVAE 0.824
Fake News Detection using Machine Learning	Jasmine Shaikh, Rupali Patil	Naïve Bayes Classifier, Passive Aggressive Classifier, Support Vector Machine, Manisha Gahirwal, Anushaya Prabha	Naïve Bayes Classifier 84.056%, Passive Aggressive Classifier 92.9%, Support Vector Machine 95.05%, Manisha Gahirwal 87%, Anushaya Prabha 94.63%
Fake News Detection Using Machine Learning Approaches	Z Khanam, B N Alwasel, H Sirafi	XGboost, Random Forests, Naive Bayes, K-Nearest Neighbors (KNN), Decision Tree, SVM	XGBOOST is depicting the highest accuracy with more than 75%, next is SVM and Random forest with approximately 73% accuracy
Fake news detection using deep learning models: A novel approach	Sachin Kumar, Rohan Asthana, Shashwat Upadhyay	CNN, LSTM, Bidirectional LSTM, CNN+LSTM ensembled, bidirectional LSTM + LSTM ensembled model, CNN + LSTM ensembled model with attention mechanism, CNN + Bidirectional LSTM ensembled model with attention mechanism.	i. 73.29% in the CNN model; ii. 80.62% in the LSTM model; iii. 83.81% in the bidirectional LSTM model;

			iv. 86.14% in the CNN + LSTM ensembled model; v. 86.89% in the bidirectional LSTM + LSTM ensembled model; vi. 86.57% in the CNN + LSTM ensembled model with attention mechanism; vii. 88.78% in the CNN + Bidirectional LSTM ensembled model with attention mechanism.
Fake News Detection using Machine Learning Algorithms	Uma Sharma, Sidarth Saran, Shankar M. Patil	Naïve Bayes, Random Forest, Logistic Regression	Naïve Bayes Precision 0.59 Recall 0.92 F1-Score 0.72 Accuracy 0.60 Random Forest Precision 0.62 Recall 0.71 F1-Score 0.67 Accuracy 0.59 Logistic Regression Precision 0.69 Recall 0.83 F1-Score 0.75 Accuracy 0.65
Weak Supervision for Fake News Detection via Reinforcement Learning	Yaqing Wang, Weifeng Yang, Fenglong Ma	Supervised CNN, EANN, WeFEND	CNN 0.747, EANN 0.767 WeFEND 0.824
Fake News Detection Using Machine Learning and Deep Learning Algorithms	Awf Abdulrahman, Muhammet Baykara	AdaBoost, XGBoost, RNN + LSTM	AdaBoost 100%, XGBoost 91.39%, RNN + LSTM 90.12%
Fake News Detection in Social Networks Using Machine Learning and Deep Learning: Performance Evaluation	Wenlin Han and Varshil Mehta	LSTM, LSTM DROP, LSTMCNN	LSTM 82.29, LSTM DROP 73.78, LSTMCNN 80.38

Objectives of the Project

The Objectives of performing Social Network Analysis on Twitter & GitHubDatasets include:

1.Our sole objective is to classify the news from the dataset to fake or true news.

2.Extensive EDA of news.

3.Selecting and building a powerful model for classification.

Workflow of the Project

1.Introduction

2.Preprocessing and cleaning

3.Story generation and visualization from news

4.Stemming & vectoring

5.Model building: fake news classifier

6.Deep learning LSTM

7.Conclusion

Proposed Methodology

Preprocessing and Cleaning

We have to perform certain preprocessing steps before performing EDA and giving the data to the model.

Let's begin with creating the output column.

Text Processing

This is an important phase for any text analysis application.

There will be many un useful content in the news which can be an obstacle when feeding to a machine learning model.

Unless we remove them, the machine learning model doesn't work efficiently.

Let’s go step by step.

Story Generation and Visualization from news

In this section we will complete do exploratory data analysis on news such as n-gram analysis and understand which are all the words, context which are most likely found in fake news.

Stemming & Vectorizing

Stemming the reviews

Stemming is a method of deriving root word from the inflected word. Here we extract the reviews and convert the words in reviews to its root word. for example,

- Going->go
- Finally->fina

If you notice, the root words don’t need to carry a semantic meaning. There is another technique known as Lemmatization where it converts the words into root words which has a semantic meaning.

Since it takes time. I'm using stemming

Deep learning-LSTM

Here in this part, we use neural network to predict whether the given news is fake or not.

We aren't going to use normal neural network like ANN to classify but LSTM (long short-term memory) which helps in containing sequence information. Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. This is a behavior required in complex problem domains like machine translation, speech recognition, and more.

About the Dataset

train.csv: A full training dataset with the following attributes:

- **id:** unique id for a news article
- **title:** the title of a news article
- **author:** author of the news article
- **text:** the text of the article; could be incomplete
- **label:** a label that marks the article as potentially unreliable
  - 1: unreliable
  - 0: reliable

test.csv: A testing training dataset with all the same attributes at train.csv without the

label.

submit.csv: A sample submission that you can

Dataset link: https://www.kaggle.com/competitions/fake-news/data

Implementation

Import Libraries

Let's import all necessary libraries for the analysis and along with it let's bring down our dataset

In [1]:
```
#Basic libraries
import pandas as pd
import numpy as np
```

In [2]:
```
#Visualization libraries
import matplotlib.pyplot as plt
from matplotlib import rcParams
import seaborn as sns
from textblob import TextBlob
from plotly import tools
import plotly.graph_objs as go
from plotly.offline import iplot
%matplotlib inline
plt.rcParams['figure.figsize'] = [10, 5]
import cufflinks as cf
cf.go_offline()
cf.set_config_file(offline=False, world_readable=True)
```

In [3]:
```
#NLTK libraries
import nltk
import re
import string
from nltk.corpus import stopwords
from wordcloud import WordCloud,STOPWORDS
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
# Machine Learning libraries
import sklearn
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
```

In [4]:
```
#Metrics libraries
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.model_selection import cross_val_score
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

In [5]:
```
#Miscellanous libraries
from collections import Counter
```

In [6]:
```
#reading the fake and true datasets
fake_news = pd.read_csv('Fake.csv')
true_news = pd.read_csv('True.csv')

# print shape of fake dataset with rows and columns and information
print ("The shape of the  data is (row, column):"+ str(fake_news.shape))
print (fake_news.info())
print("\n ---------------------------------------- \n")

# print shape of true dataset with rows and columns and information
print ("The shape of the  data is (row, column):"+ str(true_news.shape))
print (true_news.info())
```

```
The shape of the  data is (row, column):(23481, 4)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23481 entries, 0 to 23480
Data columns (total 4 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   title    23481 non-null  object
 1   text     23481 non-null  object
 2   subject  23481 non-null  object
 3   date     23481 non-null  object
dtypes: object(4)
memory usage: 733.9+ KB
None

 ----------------------------------------

The shape of the  data is (row, column):(21417, 4)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21417 entries, 0 to 21416
Data columns (total 4 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   title    21417 non-null  object
 1   text     21417 non-null  object
 2   subject  21417 non-null  object
 3   date     21417 non-null  object
dtypes: object(4)
memory usage: 669.4+ KB
None
```

In [7]:
```
#Target variable for fake news
fake_news['output']=0

#Target variable for true news
true_news['output']=1
```

In [8]:
```
#Concatenating and dropping for fake news
fake_news['news']=fake_news['title']+fake_news['text']
fake_news=fake_news.drop(['title', 'text'], axis=1)

#Concatenating and dropping for true news
true_news['news']=true_news['title']+true_news['text']
true_news=true_news.drop(['title', 'text'], axis=1)

#Rearranging the columns
fake_news = fake_news[['subject', 'date', 'news','output']]
true_news = true_news[['subject', 'date', 'news','output']]
```

In [9]:
```
fake_news['date'].value_counts()
```

Out[9]:
```
May 10, 2017         46
May 26, 2016         44
May 6, 2016          44
May 5, 2016          44
May 11, 2016         43
                     ..
December 9, 2017      1
December 4, 2017      1
November 19, 2017     1
November 20, 2017     1
Jul 19, 2015          1
Name: date, Length: 1681, dtype: int64
```

In [10]:
```
#Removing links and the headline from the date column
fake_news=fake_news[~fake_news.date.str.contains("http")]
fake_news=fake_news[~fake_news.date.str.contains("HOST")]

'''You can also execute the below code to get the result
which allows only string which has the months and rest are filtered'''
#fake_news=fake_news[fake_news.date.str.contains("Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|O
```

Out[10]:
```
'You can also execute the below code to get the result \nwhich allows only string which
has the months and rest are filtered'
```

Only fake news dataset had an issue with date column,Now let's proceed with converting the date column to datetime format

In [11]:
```
#Converting the date to datetime format
fake_news['date'] = pd.to_datetime(fake_news['date'])
true_news['date'] = pd.to_datetime(true_news['date'])
```

In [12]:
```
frames = [fake_news, true_news]
news_dataset = pd.concat(frames)
news_dataset
```

Out[12]:

	subject	date	news	output
0	News	2017-12-31	Donald Trump Sends Out Embarrassing New Year'...	0
1	News	2017-12-31	Drunk Bragging Trump Staffer Started Russian ...	0
2	News	2017-12-30	Sheriff David Clarke Becomes An Internet Joke...	0
3	News	2017-12-29	Trump Is So Obsessed He Even Has Obama's Name...	0
4	News	2017-12-25	Pope Francis Just Called Out Donald Trump Dur...	0
...	...	...	...	...
21412	worldnews	2017-08-22	'Fully committed' NATO backs new U.S. approach...	1
21413	worldnews	2017-08-22	LexisNexis withdrew two products from Chinese ...	1
21414	worldnews	2017-08-22	Minsk cultural hub becomes haven from authorit...	1
21415	worldnews	2017-08-22	Vatican upbeat on possibility of Pope Francis ...	1
21416	worldnews	2017-08-22	Indonesia to buy $1.14 billion worth of Russia...	1

44888 rows × 4 columns

In [13]:
```
#Creating a copy
clean_news=news_dataset.copy()
```

In [14]:
```
def review_cleaning(text):
    '''Make text lowercase, remove text in square brackets,remove links,remove punctuat
    and remove words containing numbers.'''
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    return text
```

In [15]:
```
clean_news['news']=clean_news['news'].apply(lambda x:review_cleaning(x))
clean_news.head()
```

Out[15]:

	subject	date	news	output
0	News	2017-12-31	donald trump sends out embarrassing new year'...	0
1	News	2017-12-31	drunk bragging trump staffer started russian ...	0
2	News	2017-12-30	sheriff david clarke becomes an internet joke...	0
3	News	2017-12-29	trump is so obsessed he even has obama's name...	0
4	News	2017-12-25	pope francis just called out donald trump dur...	0

We have removed all punctuation in our news column

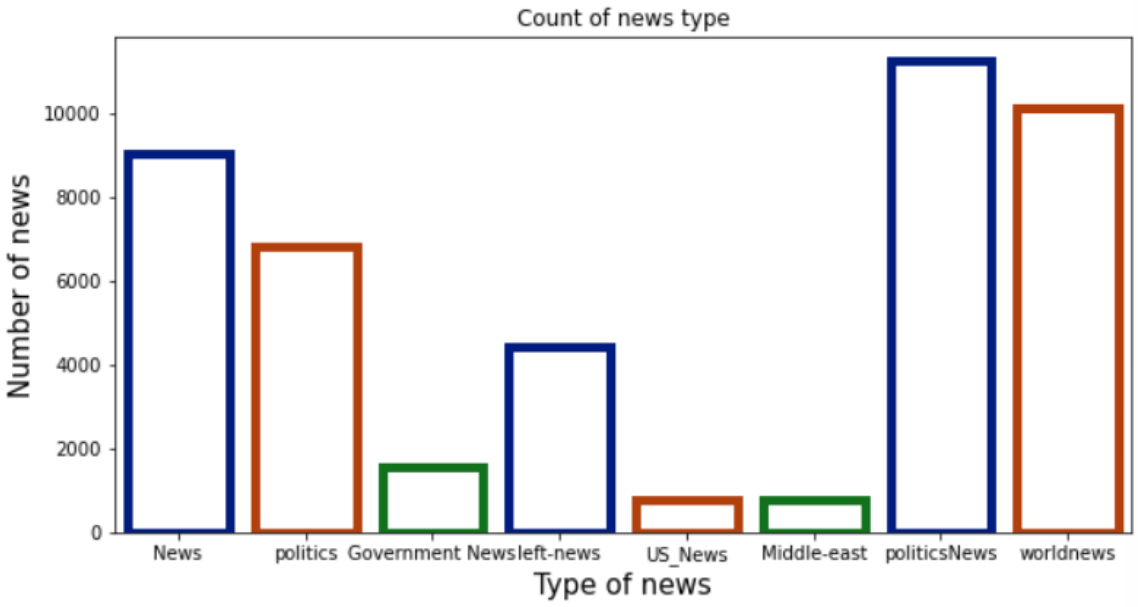
In [16]:
```
stop = stopwords.words('english')
clean_news['news'] = clean_news['news'].apply(lambda x: ' '.join([word for word in x.sp
clean_news.head()
```

Out[16]:

	subject	date	news	output
0	News	2017-12-31	donald trump sends embarrassing new year's eve...	0
1	News	2017-12-31	drunk bragging trump staffer started russian c...	0
2	News	2017-12-30	sheriff david clarke becomes internet joke thr...	0
3	News	2017-12-29	trump obsessed even obama's name coded website...	0
4	News	2017-12-25	pope francis called donald trump christmas spe...	0

We have removed all the stop words in the review column

In [17]:
```
#Plotting the frequency plot
ax = sns.countplot(x="subject", data=clean_news,
                   facecolor=(0, 0, 0, 0),
                   linewidth=5,
                   edgecolor=sns.color_palette("dark", 3))

#Setting labels and font size
ax.set(xlabel='Type of news', ylabel='Number of news',title='Count of news type')
ax.xaxis.get_label().set_fontsize(15)
ax.yaxis.get_label().set_fontsize(15)
```

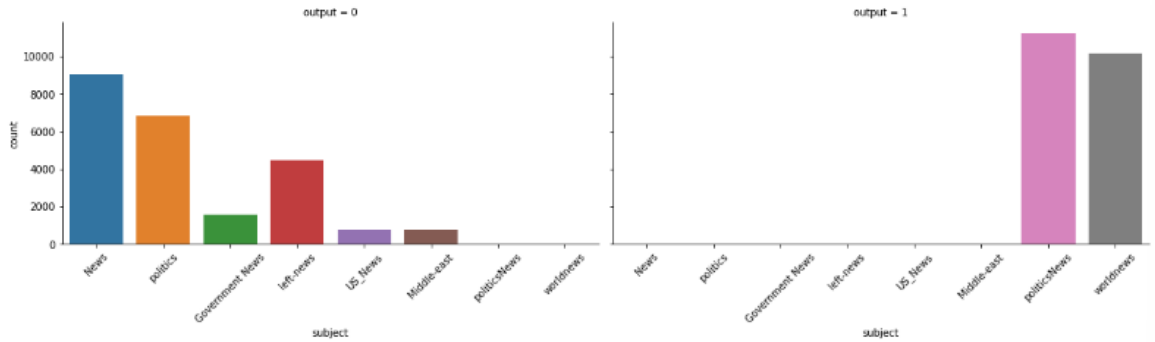


Insights:

- Our dataset has more political news than any other news followed by world news
- We have some repeated class names which expresses same meaning such as news,politics,government news etc which is similar to the alternative

In [18]:
```
g = sns.catplot(x="subject", col="output",
                data=clean_news, kind="count",
                height=4, aspect=2)

#Rotating the xlabels
g.set_xticklabels(rotation=45)
```

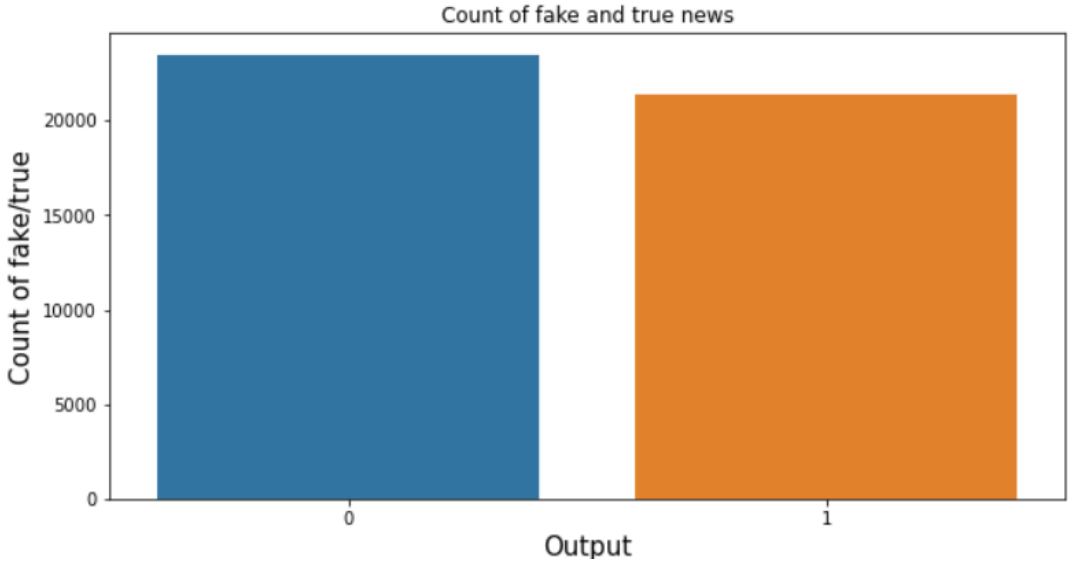
Out[18]:
```
<seaborn.axisgrid.FacetGrid at 0x7fd985c70550>
```



Insights:

- Fake news are all over the category except politics and world news
- True news are present only in politics and world news and the count is high
- THIS IS A HIGHLY BIASED DATASET and we can expect higher accuracy which doesn't signify it is a good model considering the poor quality of dataset

In [19]:
```
ax=sns.countplot(x="output", data=clean_news)

#Setting labels and font size
ax.set(xlabel='Output', ylabel='Count of fake/true',title='Count of fake and true news'
ax.xaxis.get_label().set_fontsize(15)
ax.yaxis.get_label().set_fontsize(15)
```

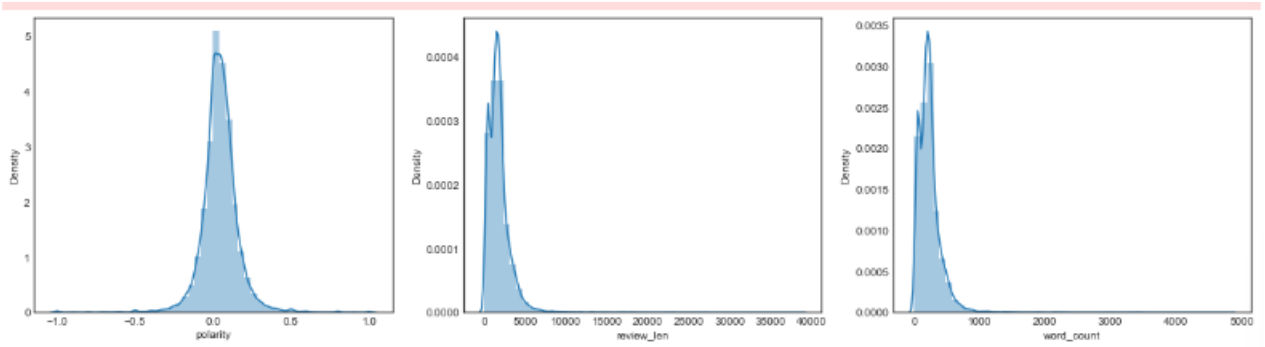


Insights:

- We have a pretty much balanced data
- But the count of fake news is higher than the true news but not on a greater extent

In [20]:
```
#Extracting the features from the news
clean_news['polarity'] = clean_news['news'].map(lambda text: TextBlob(text).sentiment.p
clean_news['review_len'] = clean_news['news'].astype(str).apply(len)
clean_news['word_count'] = clean_news['news'].apply(lambda x: len(str(x).split()))

#Plotting the distribution of the extracted feature
plt.figure(figsize = (20, 5))
plt.style.use('seaborn-white')
plt.subplot(131)
sns.distplot(clean_news['polarity'])
fig = plt.gcf()
plt.subplot(132)
sns.distplot(clean_news['review_len'])
fig = plt.gcf()
plt.subplot(133)
sns.distplot(clean_news['word_count'])
fig = plt.gcf()
```

```
/Users/phanindrasai/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:
2619: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please ada
pt your code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
```

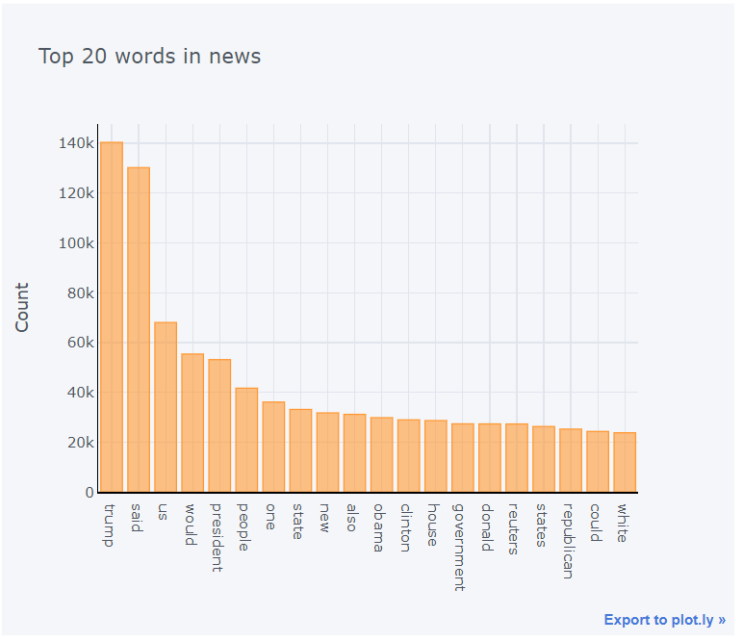


Insights:

- Most of the polarity are neutral, neither it shows some bad news nor much happy news
- The word count is between 0-1000 and the length of the news are between 0-5000 and few near 10000 words which could be an article

In [21]:

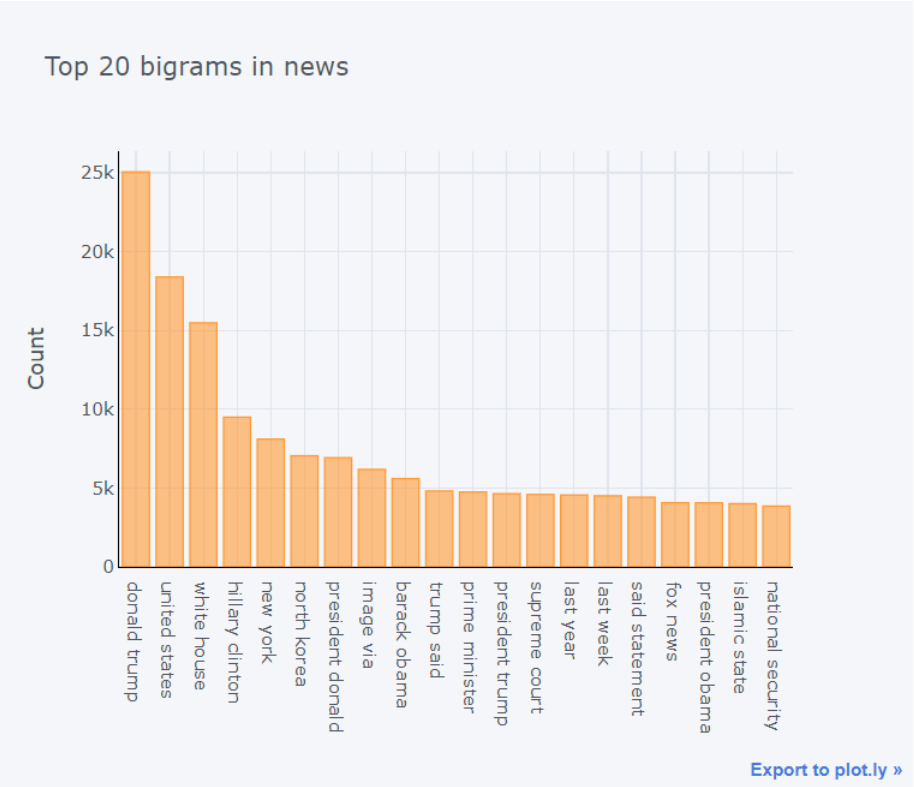
```
#Function to get top n words
def get_top_n_words(corpus, n=None):
    vec = CountVectorizer().fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
    return words_freq[:n]

#Calling function and return only top 20 words
common_words = get_top_n_words(clean_news['news'], 20)

#Printing the word and frequency
for word, freq in common_words:
    print(word, freq)

#Creating the dataframe of word and frequency
df1 = pd.DataFrame(common_words, columns = ['news' , 'count'])

#Group by words and plot the sum
df1.groupby('news').sum()['count'].sort_values(ascending=False).iplot(
    kind='bar', yTitle='Count', linecolor='black', title='Top 20 words in news')
```



Insights:

- All the top 20 news are about the US government
- Especially it's about Trump and US followed by obama
- We can understand that the news are from reuters.

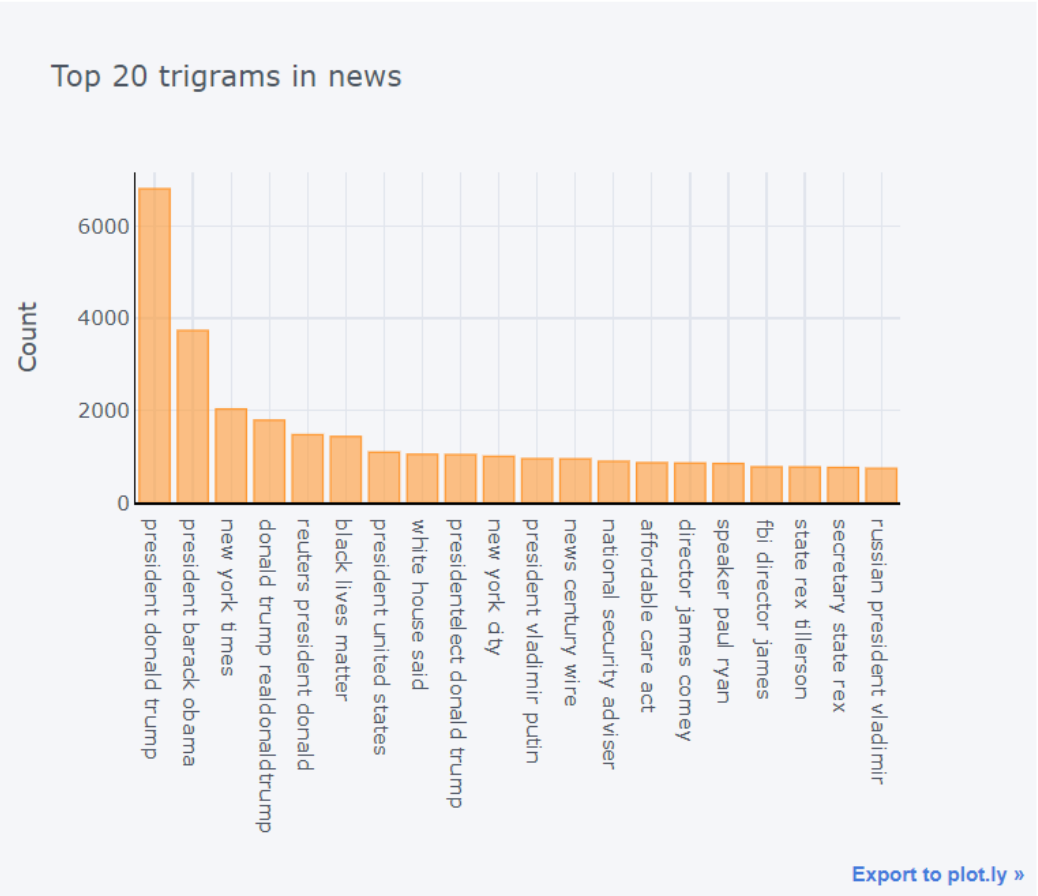
In [22]:
```
#Function to get top bigram words
def get_top_n_bigram(corpus, n=None):
    vec = CountVectorizer(ngram_range=(2, 2)).fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
    return words_freq[:n]

#Calling function and return only top 20 words
common_words = get_top_n_bigram(clean_news['news'], 20)

#Printing the word and frequency
for word, freq in common_words:
    print(word, freq)

#Creating the dataframe of word and frequency
df3 = pd.DataFrame(common_words, columns = ['news' , 'count'])

#Group by words and plot the sum
df3.groupby('news').sum()['count'].sort_values(ascending=False).iplot(
    kind='bar', yTitle='Count', linecolor='black', title='Top 20 bigrams in news')
```



Insights:

- As feared, I think the model will be biased in it's results considering the amount of trump news
- We can see the north korea news as well, I think it will be about the dispute between US and NK
- There are also few news from fox news as well

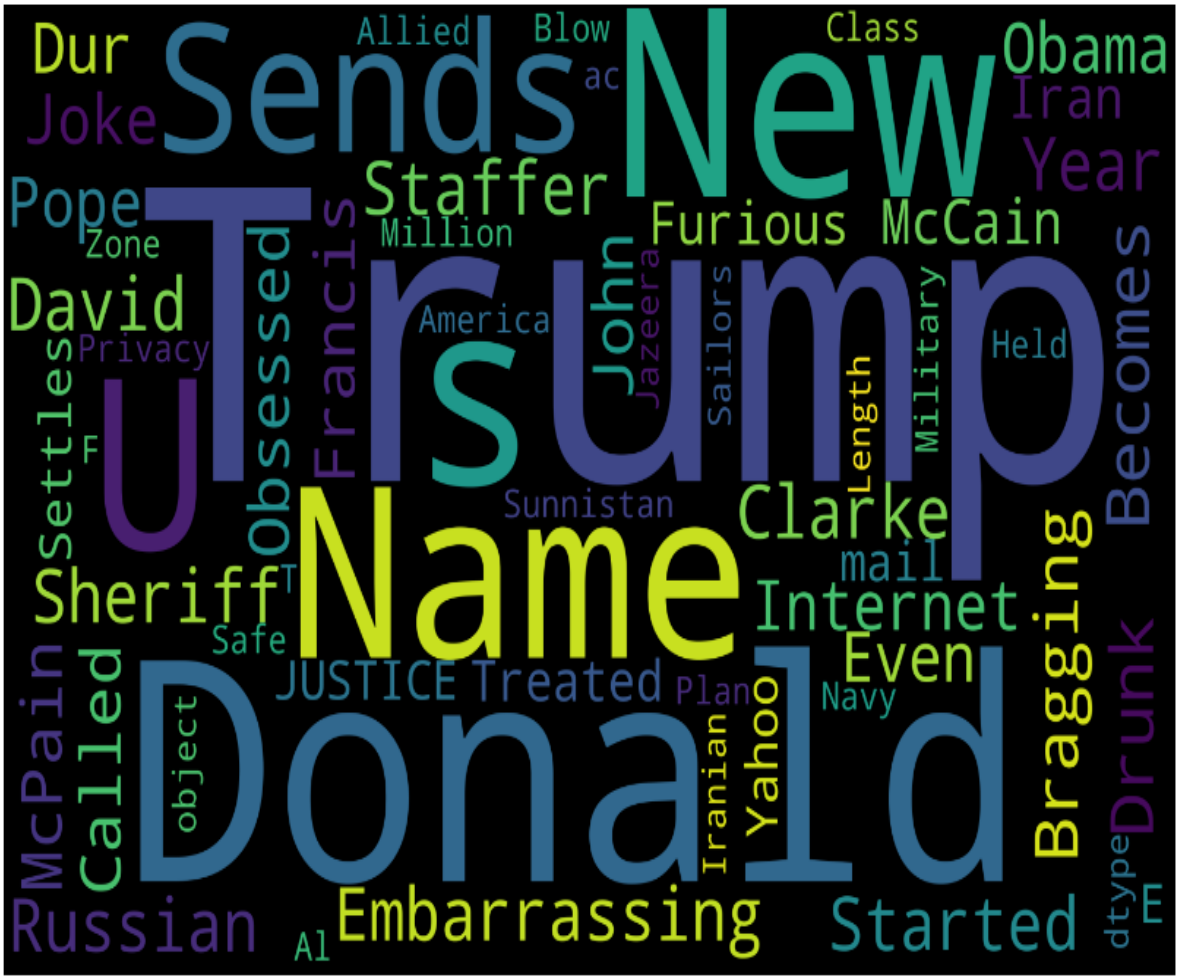
In [23]:
```
#Function to get top trigram words
def get_top_n_trigram(corpus, n=None):
    vec = CountVectorizer(ngram_range=(3, 3), stop_words='english').fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
    return words_freq[:n]

#Calling function and return only top 20 words
common_words = get_top_n_trigram(clean_news['news'], 20)

#Printing word and their respective frequencies
for word, freq in common_words:
    print(word, freq)

#Creating a dataframe with words and count
df6 = pd.DataFrame(common_words, columns = ['news' , 'count'])

#Grouping the words and plotting their frequencies
df6.groupby('news').sum()['count'].sort_values(ascending=False).iplot(
    kind='bar', yTitle='Count', linecolor='black', title='Top 20 trigrams in news')
```



Insights:

- There is an important news which ruled the US media-'Black lives matter' post the demise of Floyd. We can see that news has been covered in our data. There were lot of fake news revolved around the death.
- Rest of the news are about US politics

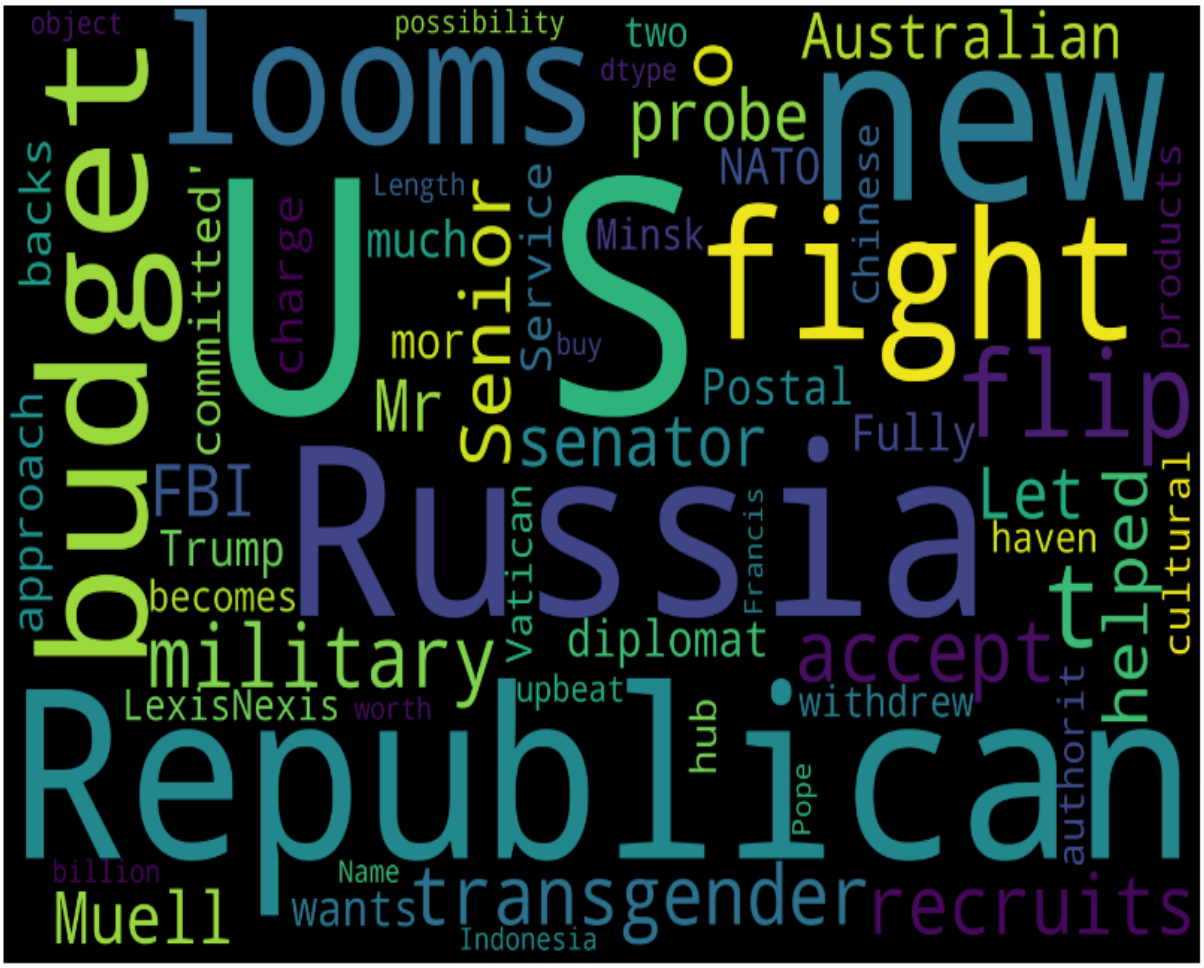
In [24]:
```
text = fake_news["news"]
wordcloud = WordCloud(
    width = 3000,
    height = 2000,
    background_color = 'black',
    stopwords = STOPWORDS).generate(str(text))
fig = plt.figure(
    figsize = (40, 30),
    facecolor = 'k',
    edgecolor = 'k')
plt.imshow(wordcloud, interpolation = 'bilinear')
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```



Insights:

- Most of the fake news revolves around Donald Trump and America
- There are also fake news about privacy, internet etc.,

In [25]:
```
text = true_news["news"]
wordcloud = WordCloud(
    width = 3000,
    height = 2000,
    background_color = 'black',
    stopwords = STOPWORDS).generate(str(text))
fig = plt.figure(
    figsize = (40, 30),
    facecolor = 'k',
    edgecolor = 'k')
plt.imshow(wordcloud, interpolation = 'bilinear')
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```

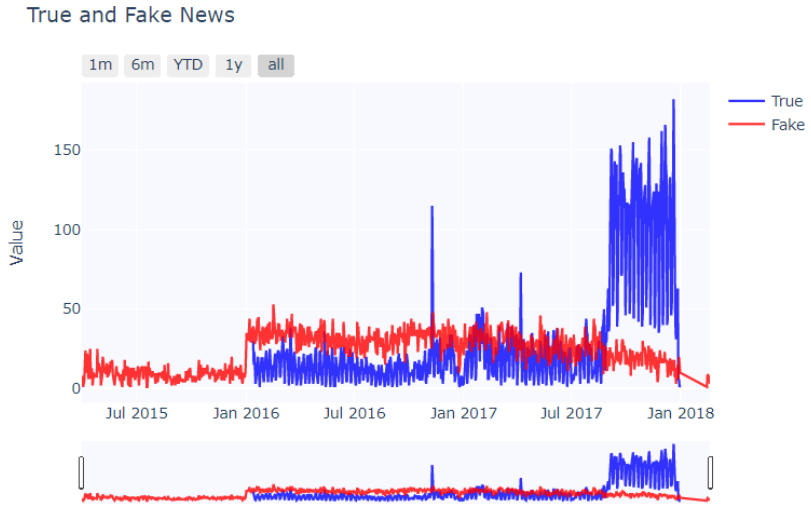


Insights:

- True news doesn't involve much trump instead on Republican Party and Russia
- There are news about Budget,military which comes under government news

TIME SERIES ANALYSIS- FAKE/TRUE news

In [26]:
```
#Creating the count of output based on date
fake=fake_news.groupby(['date'])['output'].count()
fake=pd.DataFrame(fake)

true=true_news.groupby(['date'])['output'].count()
true=pd.DataFrame(true)

#Plotting the time series graph
fig = go.Figure()
fig.add_trace(go.Scatter(
        x=true.index,
        y=true['output'],
        name='True',
    line=dict(color='blue'),
    opacity=0.8))

fig.add_trace(go.Scatter(
        x=fake.index,
        y=fake['output'],
        name='Fake',
    line=dict(color='red'),
    opacity=0.8))

fig.update_xaxes(
    rangeslider_visible=True,
    rangeselector=dict(
        buttons=list([
            dict(count=1, label="1m", step="month", stepmode="backward"),
            dict(count=6, label="6m", step="month", stepmode="backward"),
            dict(count=1, label="YTD", step="year", stepmode="todate"),
            dict(count=1, label="1y", step="year", stepmode="backward"),
            dict(step="all")
        ])
    )
)


fig.update_layout(title_text='True and Fake News',plot_bgcolor='rgb(248, 248, 255)',yaxis_title='Va

fig.show()
```



Insights:

- True news got their dominance since Aug 2017. As they are seen at a very higher rates.That is a good sign
- There are few outliers in true news where it was higher than the fake news(Nov 9, 2016 and Apr 7, 2017)
- Our dataset has more fake news than the true one as we can see that we don't have true news data for whole 2015, So the fake news classification will be pretty accurate than the true news getting classified

STEMMING & VECTORIZING

In [27]:
```
#Extracting 'reviews' for processing
news_features=clean_news.copy()
news_features=news_features[['news']].reset_index(drop=True)
news_features.head()
```

Out[27]:

	news
0	donald trump sends embarrassing new year's eve...
1	drunk bragging trump staffer started russian c...
2	sheriff david clarke becomes internet joke thr...
3	trump obsessed even obama's name coded website...
4	pope francis called donald trump christmas spe...

In [28]:
```
stop_words = set(stopwords.words("english"))
#Performing stemming on the review dataframe
ps = PorterStemmer()

#splitting and adding the stemmed words except stopwords
corpus = []
for i in range(0, len(news_features)):
    news = re.sub('[^a-zA-Z]', ' ', news_features['news'][i])
    news= news.lower()
    news = news.split()
    news = [ps.stem(word) for word in news if not word in stop_words]
    news = ' '.join(news)
    corpus.append(news)
```

In [29]:
```
corpus[1]
```

Out[29]:
```
'drunk brag trump staffer start russian collus investigationhous intellig committe chairman devin
nune go bad day assumpt like mani us christoph steeledossi prompt russia investig lash depart just
ic fbi order protect trump happen dossier start investig accord document obtain new york timesform
trump campaign advis georg papadopoulo drunk wine bar reveal knowledg russian opposit research hil
lari clintonon top papadopoulo covfef boy trump administr alleg much larger role none damn drunken
fool wine bar coffe boy help arrang new york meet trump presid abdel fattah elsisi egypt two month
elect known former aid set meet world leader trump team trump ran mere coffe boyin may papadopoulo
reveal australian diplomat alexand downer russian offici shop around possibl dirt thendemocrat pre
sidenti nomine hillari clinton exactli much mr papadopoulo said night kensington wine room austral
ian alexand downer unclear report state two month later leak democrat email began appear onlin aus
tralian offici pass inform mr papadopoulo american counterpart accord four current former american
foreign offici direct knowledg australian role papadopoulo plead guilti lie fbi cooper wit special
counsel robert mueller teamthi presid badli script realiti tv showphoto win mcnameegetti imag'
```

This is how a line looks like now, as computer cannot understand words and their sentiment we need to convert these words into 1's and 0's. To encode it we use TFIDF

TFIDF

In [30]:
```
tfidf_vectorizer = TfidfVectorizer(max_features=5000,ngram_range=(2,2))
# TF-IDF feature matrix
X= tfidf_vectorizer.fit_transform(news_features['news'])
X.shape
```

Out[30]:
```
(44888, 5000)
```

As we have considered 5000 words, we can confirm that we have 5000 columns from the shape.

In [31]:
```
#Getting the target variable
y=clean_news['output']
```

Checking for balance of data

We should be careful about when handling imbalance data. If it is imbalanced, the model will be biased towards the higher frequency class and returns max output

In [32]:
```
print(f'Original dataset shape : {Counter(y)}')
```

```
Original dataset shape : Counter({0: 23471, 1: 21417})
```

Our dataset is nearly a balanced one. So let's leave balancing it.

Train-test split(75:25)

Using train test split function we are splitting the dataset into 75:25 ratio for train and test set respectively.

In [33]:
```
## Divide the dataset into Train and Test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

MODEL: FAKE NEWS CLASSIFIER

In [34]:
```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    thresh = cm.max() / 2.
    for i in range (cm.shape[0]):
        for j in range (cm.shape[1]):
            plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
```

MODEL SELECTION

In [35]:
```
#creating the objects
logreg_cv = LogisticRegression(random_state=0)
dt_cv=DecisionTreeClassifier()
knn_cv=KNeighborsClassifier()
nb_cv=MultinomialNB(alpha=0.1)
cv_dict = {0: 'Logistic Regression', 1: 'Decision Tree',2:'KNN',3:'Naive Bayes'}
cv_models=[logreg_cv,dt_cv,knn_cv,nb_cv]

#Printing the accuracy
for i,model in enumerate(cv_models):
    print("{} Test Accuracy: {}".format(cv_dict[i],cross_val_score(model, X, y, cv=10, scoring ='ac
```

```
Logistic Regression Test Accuracy: 0.9660040199274997
Decision Tree Test Accuracy: 0.9347034035267037
KNN Test Accuracy: 0.613172841991654
Naive Bayes Test Accuracy: 0.9373328405462511
```

From the results, we can see logistic regression outdone the rest of the algorithms followed by Naive Bayes and Decision Tree.That's great. So let's go with logistic regression with hyperparameter tuning.

Logistic Regression with Hyperparameter Tuning

We use regularization parameter and penality for parameter tuning. let's see which one to plug.

In [36]:
```
param_grid = {'C': np.logspace(-4, 4, 50),
              'penalty':['l1', 'l2']}
clf = GridSearchCV(LogisticRegression(random_state=0), param_grid,cv=5, verbose=0,n_jobs=-1)
best_model = clf.fit(X_train,y_train)
print(best_model.best_estimator_)
print("The mean accuracy of the model is:",best_model.score(X_test,y_test))
```

```
/Users/phanindrasai/opt/anaconda3/lib/python3.9/site-packages/sklearn/linear_model/_logistic.py:81
4: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

```
LogisticRegression(C=24.420530945486497, random_state=0)
The mean accuracy of the model is: 0.9803065407235787
```

In [37]:
```
logreg = LogisticRegression(C=24.420530945486497, random_state=0)
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
print('Accuracy of logistic regression classifier on test set: {:.2f}'.format(logreg.score(X_test,
```

```
Accuracy of logistic regression classifier on test set: 0.98
```

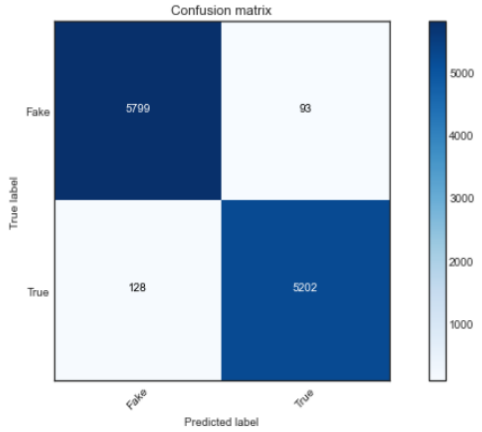
We have got 98% accuracy. As already discussed before this is a biased dataset and we can easily get such higher accuracy without any effort in processing it. But for classification problems we need to get confusion matrix and check f1 score rather than accuracy

Confusion Matrix

Let's look at the true positive and true negative classified by the model

In [38]:
```
cm = metrics.confusion_matrix(y_test, y_pred)
plot_confusion_matrix(cm, classes=['Fake','True'])
```

```
Confusion matrix, without normalization
```



Check out the diagonal elements(5799+5202), they are correctly predicted records and rest are incorrectly classified by the algorithm. Our model has done well(results are good by the data is biased :P)

Classification Report

Considering Fake news, we should seriously consider precision score (False positive). We can't afford the mistakes when the model classifies fake news as true which will lead to chaos

In [39]:
```
print("Classification Report:\n",classification_report(y_test, y_pred))
```

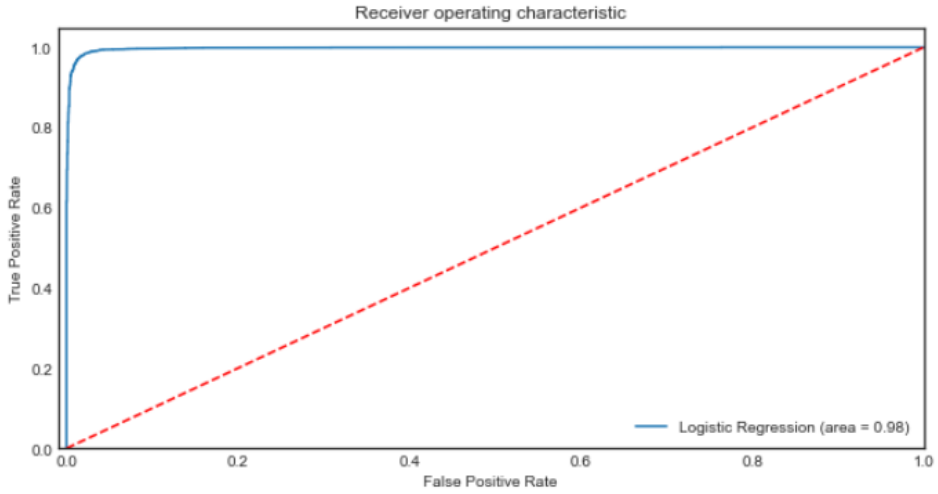
```
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.98      0.98      5892
           1       0.98      0.98      0.98      5330

    accuracy                           0.98     11222
   macro avg       0.98      0.98      0.98     11222
weighted avg       0.98      0.98      0.98     11222
```

All our scores are 98%. Certainly unreal to get such values. There are only changes in the support.

ROC-AUC Curve

This is a very important curve where we decide on which threshold to setup based upon the objective criteria

In [40]:
```
logit_roc_auc = roc_auc_score(y_test, logreg.predict(X_test))
fpr, tpr, thresholds = roc_curve(y_test, logreg.predict_proba(X_test)[:,1])
plt.figure()
plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([-0.01, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```



We should consider the AUC score here which is 98%. Very well. All metrics are performing good. The more far left the curve is better our model We can adjust our threshold based on our ROC curve to get results based on model requirements

CONCLUSION:

We have done a mainstream work on processing the data and building the model. We could have indulged in changing the n-grams while vectorizing the text data. We took 2 words and vectorized it.

Most of the fake news are surrounded among Election news and about Trump. Considering the US elections 2020. There are chances to spread fake news and application of these technology will be heavily required. Fake news is currently rooted during this pandemic situation to play politics and to scare people and force them to buy goods
Most of the news are from Reuters. We don't know whether this news media is politically influenced. So, we should always consider the source of news to find if the news is fake or true.

GIT-HUB LINK: ***https://github.com/phanindrasai27/fake-news-analysis***

References:

1. A. Douglas, "News consumption and the new electronic media," *The International Journal of Press/Politics*, vol. 11, no. 1, pp. 29–52, 2006.
   View at: Publisher Site | Google Scholar
2. J. Wong, "Almost all the traffic to fake news sites is from facebook, new data show," 2016.
   View at: Google Scholar
3. D. M. J. Lazer, M. A. Baum, Y. Benkler et al., "The science of fake news," *Science*, vol. 359, no. 6380, pp. 1094–1096, 2018.
   View at: Publisher Site | Google Scholar
4. S. A. García, G. G. García, M. S. Prieto, A. J. M. Guerrero, and C. R. Jiménez, "The impact of term fake news on the scientific community scientific performance and mapping in web of science," *Social Sciences*, vol. 9, no. 5, 2020.
   View at: Google Scholar
5. A. D. Holan, *2016 Lie of the Year: Fake News*, Politifact, Washington, DC, USA, 2016.
6. S. Kogan, T. J. Moskowitz, and M. Niessner, "Fake News: Evidence from Financial Markets," 2019, https://ssrn.com/abstract=3237763.
   View at: Google Scholar
7. A. Robb, "Anatomy of a fake news scandal," *Rolling Stone*, vol. 1301, pp. 28–33, 2017.
   View at: Google Scholar
8. J. Soll, "The long and brutal history of fake news," *Politico Magazine*, vol. 18, no. 12, 2016.
   View at: Google Scholar
9. J. Hua and R. Shaw, "Corona virus (covid-19) "infodemic" and emerging issues through a data lens: the case of China," *International Journal of Environmental Research and Public Health*, vol. 17, no. 7, p. 2309, 2020.
   View at: Publisher Site | Google Scholar
10. N. K. Conroy, V. L. Rubin, and Y. Chen, "Automatic deception detection: methods for finding fake news," *Proceedings of the Association for Information Science and Technology*, vol. 52, no. 1, pp. 1–4, 2015.
    View at: Publisher Site | Google Scholar
11. F. T. Asr and M. Taboada, "Misinfotext: a collection of news articles, with false and true labels," 2019.
    View at: Google Scholar
12. K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

View at: Publisher Site | Google Scholar

13. S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.
    View at: Publisher Site | Google Scholar
14. H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of Economic Perspectives*, vol. 31, no. 2, pp. 211–236, 2017.
    View at: Publisher Site | Google Scholar
15. V. L. Rubin, N. Conroy, Y. Chen, and S. Cornwell, "Fake news or truth? using satirical cues to detect potentially misleading news," in *Proceedings of the Second Workshop on Computational Approaches to Deception Detection*, pp. 7–17, San Diego, CA, USA, 2016.
    View at: Google Scholar
16. H. Jwa, D. Oh, K. Park, J. M. Kang, and H. Lim, "exBAKE: automatic fake news detection model based on bidirectional encoder representations from transformers (bert)," *Applied Sciences*, vol. 9, no. 19, 2019.
    View at: Publisher Site | Google Scholar
17. H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using n-gram analysis and machine learning techniques," in *Proceedings of the International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, pp. 127–138, Springer, Vancouver, Canada, 2017.
    View at: Publisher Site | Google Scholar
18. W. Y. Wang, *Liar, Liar Pants on Fire: A New Benchmark Dataset for Fake News Detection*, Association for Computational Linguistics, Stroudsburg, PA, USA, 2017.
19. B. Riedel, I. Augenstein, G. P. Spithourakis, and S. Riedel, "A simple but tough-to-beat baseline for the fake news challenge stance detection task," 2017, https://arxiv.org/abs/1707.03264.
    View at: Google Scholar
20. N. Ruchansky, S. Seo, and Y. Liu, "Csi: a hybrid deep model for fake news detection," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 797–806, Singapore, 2017.
    View at: Google Scholar