FLIP ROBO

Fake News Classification

Submitted by:

Praveen Pandey

ACKNOWLEDGMENT

I would like to express my gratitude to my guide Mohd Kashif (SME, Flip Robo) for his constant guidance, continuous encouragement and unconditional help towards the development of the project. It was he who helped me whenever I got stuck somewhere in between. The project would have not been completed without his support and confidence he showed towards me.

Lastly, I would l like to thank all those who helped me directly or indirectly toward the successful completion of the project.

INTRODUCTION

- ## Business Problem Framing

Fake news is false or misleading information presented as news. It often has the aim of damaging the reputation of a person or entity, or making money through advertising revenue. However, the term does not have a fixed definition, and has been applied more broadly to include any type of false information, including unintentional and unconscious mechanisms, and also by high-profile individuals to apply to any news unfavourable to his/her personal perspectives.

Once common in print, the prevalence of fake news has increased with the rise of social media, especially the Facebook News Feed. Political polarization, post-truth politics, confirmation bias, and social media algorithms have been implicated in the spread of fake news. It is sometimes generated and propagated by hostile foreign actors, particularly during elections. The use of anonymously-hosted fake news websites has made it difficult to prosecute sources of fake news for libel. In some definitions, fake news includes satirical articles misinterpreted as genuine, and articles that employ sensationalist or clickbait headlines that are not supported in the text.

Fake news can reduce the impact of real news by competing with it; a Buzzfeed analysis found that the top fake news stories about the 2016 U.S. presidential election received more engagement on Facebook than top stories from major media outlets. It also has the potential to undermine trust in serious media coverage. The term has at times been used to cast doubt upon legitimate news, and former U.S. president Donald Trump has been credited with popularizing the term by using it to describe any negative press coverage of himself. It has been increasingly criticized, due in part to Trump's misuse, with the British government deciding to avoid the term, as it is "poorly- defined" and "conflates a variety of false information, from genuine error through to foreign interference".

Multiple strategies for fighting fake news are currently being actively researched, and need to be tailored to individual types of fake news. Effective self-regulation and legally-enforced regulation of social media and web search engines are needed. The information space needs to be flooded with accurate news to displace fake news. Individuals need to actively confront false narratives when spotted, as well as take care when sharing information via social media. However, reason, the scientific method and critical thinking skills alone are insufficient to counter the broad scope of bad ideas. Overlooked is the power of confirmation bias, motivated reasoning and other cognitive biases that can seriously distort the many facets of immune mental health. Inoculation theory shows promise in designing techniques to make individuals resistant to the lure of fake news, in the same way that a vaccine protects against infectious diseases.

• Conceptual Background of the Domain Problem

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.

In the below blog we are going to see about how we are classifying the fake news with the genuine news; we are going to use several machine learning techniques and we will plot and analyse how to identify a news as fake. I have tried using several NLP techniques and arrived at a model that will classify news is fake or genuine.

• Review of Literature

The purpose of the literature review is to:

1. Identify the foul words or foul statements that are being used.
2. Stop the people from using these foul languages in online public forum.

To solve this problem, we are now building a model using our machine language technique that identifies all the foul language and foul words, using which the online platforms like social media principally stops these mob using the foul language in an online community or even block them or block them from using this foul language.

I have used 7 different Classification algorithms and shortlisted the best on basis on the metrics of performance and I have chosen one algorithm and build a model in that algorithm.

- Motivation for the Problem Undertaken

Fake news is a topic that has gained a lot of attention in the past few years, and for good reasons. As social media becomes widely accessible, it becomes easier to influence millions of people by spreading misinformation. As humans, we often fail to recognize if the news we read is real or fake. A study from the University of Michigan found that human participants were able to detect fake news stories only 52.29 percent of the time. But can a neural network do any better? Keep reading to find out.

The goal of this article is to answer the following questions:

- What kinds of topics or keywords appear frequently in real news versus fake news?
- How can we use a deep neural network to identify fake news stories?

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

I start analysis on this project in importing the data set and simple play around with the data and identifying the characteristics of each column.

I noticed that there are Two dataset both have four columns “title”, “text”, “subject”, “date”.

```
fake = pd.read_csv('Fake.csv')
fake.head()
```

	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

```
true = pd.read_csv('True.csv')
true.head()
```

	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

We can concatenate both dataset and create new dataset is Result.

```
Result = pd.concat([fake,true]).sample(frac=1).reset_index(drop=True)
Result.head()
```

	title	text	subject	date	Type
0	Top U.S. official visits Vietnam to assess hum...	HANOI (Reuters) - A top U.S. envoy began a two...	politicsNews	May 9, 2016	True
1	TRUMP HITS BACK After Cowgirl Congresswoman Tr...	The left is going ballistic over supposed word...	politics	Oct 18, 2017	Fack
2	FBI has sufficient resources for Russia invest...	WASHINGTON (Reuters) - The FBI's acting head s...	politicsNews	May 11, 2017	True
3	WATCH: MSNBC Cuts Mic Of GOP Senator Lindsey G...	MSNBC s Casey Hunt was interviewing war-hawk a...	politics	Jul 11, 2017	Fack
4	Democrats push for full analysis of latest Rep...	WASHINGTON (Reuters) - Democratic leaders in C...	politicsNews	September 18, 2017	True

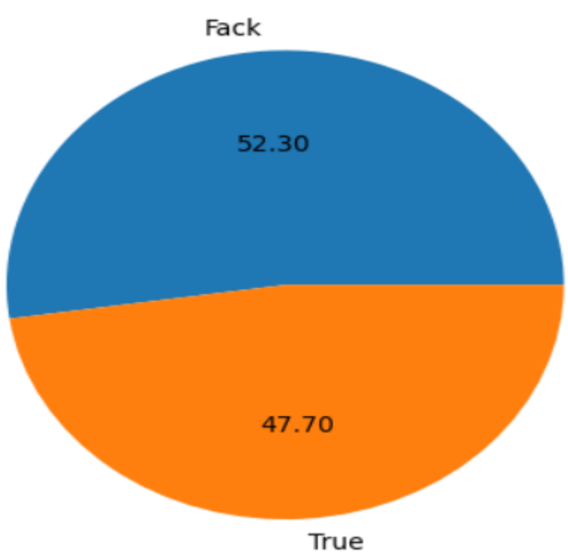
‘title’, ‘subject’ and ‘date’ which won’t help us in predicting. So, I decided to drop.

```
Result = Result.drop(['title','subject','date'],axis=1)
Result.head()
```

	text	Type
0	HANOI (Reuters) - A top U.S. envoy began a two...	1
1	The left is going ballistic over supposed word...	0
2	WASHINGTON (Reuters) - The FBI’s acting head s...	1
3	MSNBC s Casey Hunt was interviewing war-hawk a...	0
4	WASHINGTON (Reuters) - Democratic leaders in C...	1

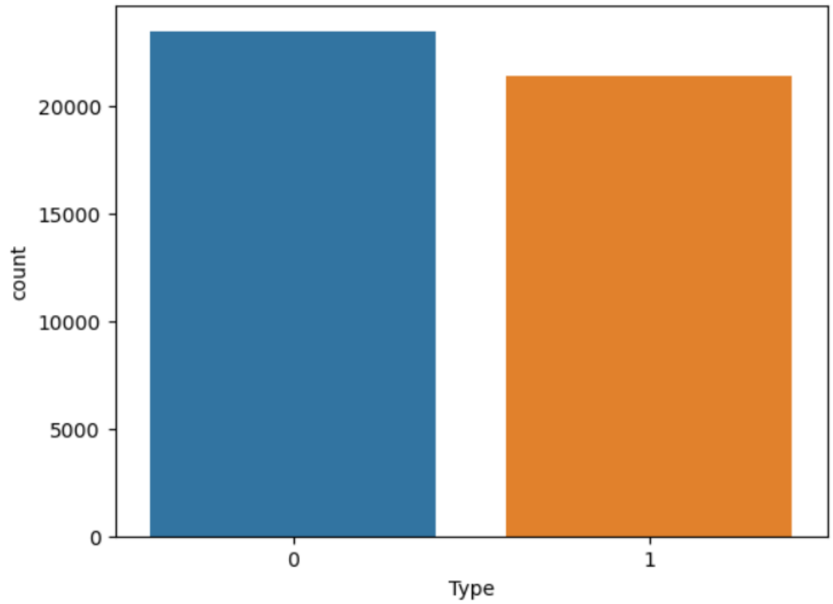
Then post this I analyzed the label column which is our target variable and I understood that label column has two variables ‘0’ and ‘1’. ‘1’denotes not a fake news and ‘0’ denotes fake news.

```
plt.pie(Result['Type'].value_counts(),labels=['Fack','True'],autopct="%0.2f")
plt.show()
Result['Type'].value_counts()
```




```
0    23481
1    21417
Name: Type, dtype: int64
```

```
sns.countplot(Result.Type);
```




```
fake_news = Result[(Result.Type ==0)]
percent=len(fake_news)/len(Result)*100
print('Percentage of Fake = ',percent)
print('Percentage of not Fake news= ', (100-percent))
```

```
Percentage of Fake =  52.29854336496058
Percentage of not Fake news=  47.70145663503942
```

On further analysis of the label data, I understood that we have a data almost 52.29% of data with fake news and 47.70 of data with not fake news. Balance data will help us in building a perfect machine learning model and we also avoid the model to overfit and underfit with the data.

- ## Data Sources and their formats

There are 6 columns in the dataset. The description of each of the column is given below:

- “title”: It is the title of the news.
- “text”: It contains the full text of the news article
- “subject”: It contain the topic is politics New and politics.
- “data”: It is a serial data
- “Type”: It tells whether the news is fake (0) or not fake (1)

- ## **Data Pre-processing**

  I started the pre-processing with cleansing the data, filtering out all the Bash data and I like to keep only the required data for our analysis.

  I started with importing the required libraries. And I have declared stop words and lemmatize to a variable.

```
#Importing Required Libraries
import nltk
import re
import string
from nltk.corpus import stopwords
from wordcloud import WordCloud
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer
```

After importing all the required libraries, I have defined stopwords and lemmatize to a variable.

```
from nltk.corpus import stopwords
def clean_text(Result, Result_column_name):

    #Converting all messages to lowercase
    Result[Result_column_name] = Result[Result_column_name].str.lower()

    #Replace email addresses with 'email'
    Result[Result_column_name] = Result[Result_column_name].str.replace(r'^.+@[^\.].*\.[a-z]{2,}$','emailaddress')

    #Replace URLs with 'webaddress'
    Result[Result_column_name] = Result[Result_column_name].str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$','weba

    #Replace money symbols with 'dollars' (£ can by typed with ALT key + 156)
    Result[Result_column_name] = Result[Result_column_name].str.replace(r'£|\$', 'dollars')

    #Replace 10 digit phone numbers (formats include paranthesis, spaces, no spaces, dashes) with 'phonenumber'
    Result[Result_column_name] = Result[Result_column_name].str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$','phonenumber

    #Replace numbers with 'numbr'
    Result[Result_column_name] = Result[Result_column_name].str.replace(r'\d+(\.\d+)?', 'numbr')

    #Remove punctuation
    Result[Result_column_name] = Result[Result_column_name].str.replace(r'[^\w\d\s]', ' ')

    #Replace whitespace between terms with a single space
    Result[Result_column_name] = Result[Result_column_name].str.replace(r'\s+', ' ')

    #Remove leading and trailing whitespace
    Result[Result_column_name] = Result[Result_column_name].str.replace(r'^\s+|\s+?$', '')

    #Remove stopwords
    stop_words = set(stopwords.words('english') + ['u', 'ü', 'â', 'ur', '4', '2', 'im', 'dont', 'doin', 'ure'])
    Result[Result_column_name] = Result[Result_column_name].apply(lambda x: ' '.join(term for term in x.split() if term not in st
```

```
clean_text(Result, 'text')
Result['text'].head()
```

Post on creating a function I have passed my data into the same to clean it.

```
clean_text(Result, 'text')
Result['text'].head()
```

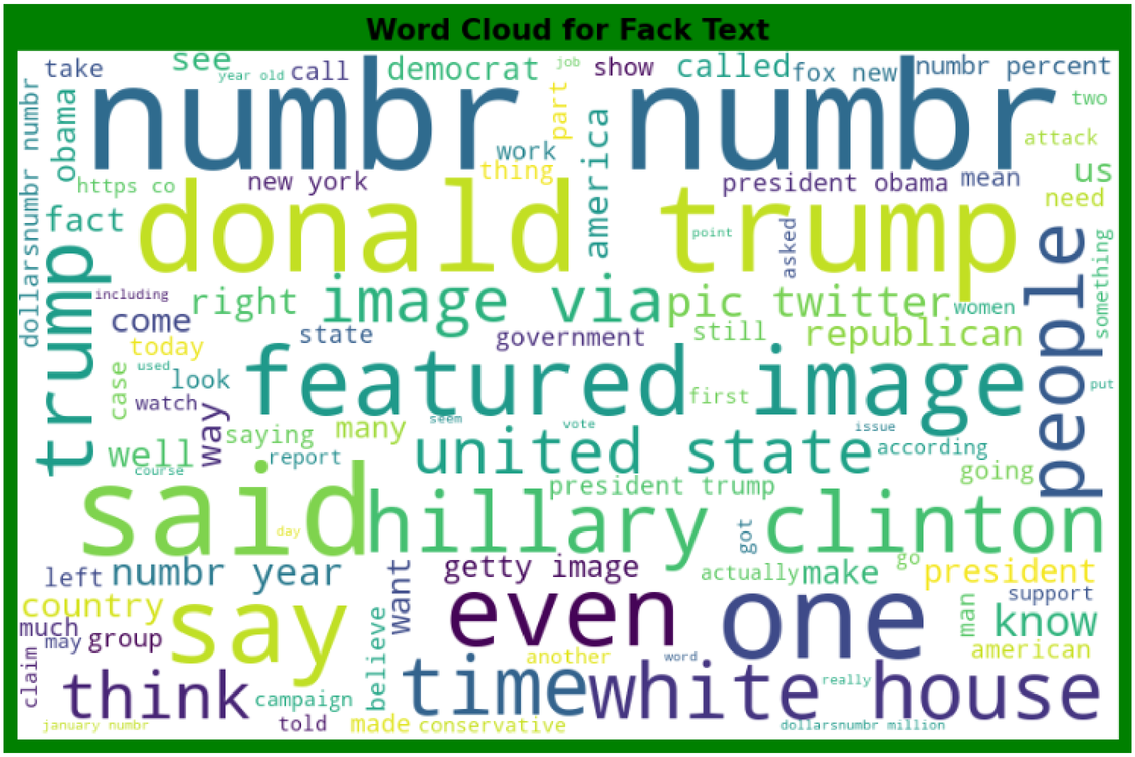
```
0    pepsi got hammered everyone lame attempt glamo...
1    manila reuters three russian warships includin...
2    rome reuters italian prime minister paolo gent...
3    canfield ohio cleveland reuters donald trump h...
4    washington reuters federal appeals court monda...
Name: text, dtype: object
```

The total amount of data that is cleansed from the original data is 44898. Now the data is cleansed and ready for training but before which I converted the data into vectors for the machine learning models to understand the data, so I imported TFIDF vectorizer and I have made the max feature as 3000.

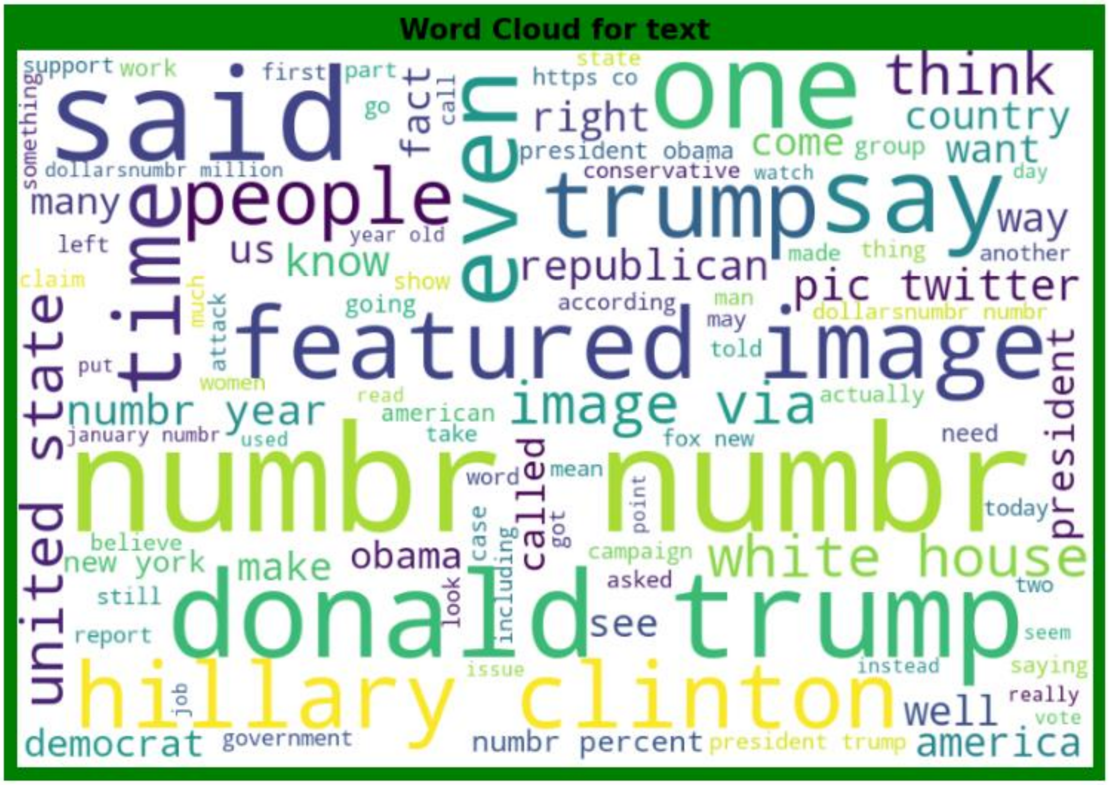
```
from sklearn.feature_extraction.text import TfidfVectorizer
tf = TfidfVectorizer(max_features=3000)
fertures = tf.fit_transform(Result['text'])
X=fertures
Y=Result[['Type']]
```

- **Data Inputs- Logic- Output Relationships**

I have analysed the input output logic with word cloud and I have word clouded the sentenced that as classified as foul language in every category.



We can see the foul words that are mostly used in Fake News classified sentences we are seeing top 100 words the words which are bigger in size are mostly used.



We can see the foul words that are mostly used in News classified sentences we are seeing top 100 words the words which are bigger in size are mostly used.

- **Hardware and Software Requirements and Tools Used**

1. Python 3.10
2. NumPy.
3. Pandas.
4. Matplotlib.
5. Seaborn. 6. Data science.
6. SciPy
7. Sklearn.
8. Anaconda Environment, Jupyter Notebook.

Model/s Development and Evaluation

- ## Testing of Identified Approaches (Algorithms)

I have started the training in selecting the best random state parameter for the model as follows.

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
maxAccu = 0
maxRs = 0
for i in range(1,200):
    X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=.30,random_state=i)
    RF = RandomForestClassifier()
    RF.fit(X_train,Y_train)
    predRF = RF.predict(X_test)
    acc = accuracy_score(Y_test,predRF)
    if acc>maxAccu:
        maxAccu=acc
        maxRs = i
print(f'Best Accuracy is {maxAccu} on Random_state {maxRs}')
```

```
Best Accuracy is 0.988394584139265 on Random_state 195
```

After selecting the best random state parameter, I have spitted the data into test and train with test size as 30 %. Again, I have imported the required libraries to import my ML algorithms.

- ## Run and evaluate selected models

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import r2_score,confusion_matrix,classification_report,mean_absolute_error,mean_squared_error

LOR = LogisticRegression()
LOR.fit(X_train,Y_train)

# Prediction
predLOR = LOR.predict(X_test)
print('R2 Score :',r2_score(Y_test,predLOR))

# Mean Absolute Error(MAE)
print('Mean Absolute Error(MAE)',mean_absolute_error(Y_test,predLOR))

# Mean Squared Error(MSE)
print('Mean Squared Error',mean_squared_error(Y_test,predLOR))

# Root Mean Squared Error (RMSE)
print('Root Mean Squared Error',np.sqrt(mean_squared_error(Y_test,predLOR)))

print("------------------------------------------------------")
# Accuracy Score
print(accuracy_score(Y_test, predLOR))
print("------------------------------------------------------")
# Confusion Matrix
print(confusion_matrix(Y_test, predLOR))
print("------------------------------------------------------")
# Classification Report
print(classification_report(Y_test,predLOR))
```

Output: -

```
R2 Score : 0.9475834417164274
Mean Absolute Error(MAE) 0.013066072754268746
Mean Squared Error 0.013066072754268746
Root Mean Squared Error 0.11430692347477797
-----------------------------------------------------
0.9869339272457313
-----------------------------------------------------
[[6999   99]
 [  77 6295]]
-----------------------------------------------------
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      7098
           1       0.98      0.99      0.99      6372

    accuracy                           0.99     13470
   macro avg       0.99      0.99      0.99     13470
weighted avg       0.99      0.99      0.99     13470
```

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier

# Checking accuracy for Random Forest Classifier
RFC = RandomForestClassifier()
RFC.fit(X_train,Y_train)

# [Prediction]
predRFC = RFC.predict(X_test)
print('R2 Score:',r2_score(Y_test,predRFC))

# Mean Absolute Error(MAE)
print('Mean Absolute Error',mean_absolute_error(Y_test,predRFC))

# Mean Squared Error(MSE)
print('Mean Squared Error',mean_squared_error(Y_test,predRFC))

# Root Mean Squared Error (RMSE)
print('Root Mean Squared Error',np.sqrt(mean_squared_error(Y_test,predRFC)))
print("-----------------------------------------------------")
# Accuracy Score
print('Accuracy Score: ',accuracy_score(Y_test, predRFC))
print("-----------------------------------------------------")
# Confusion Matrix
print('Confusion Matrix:\n',confusion_matrix(Y_test, predRFC))
print("-----------------------------------------------------")
# Classification Report
print(classification_report(Y_test,predRFC))
```

Output: -

```
R2 Score: 0.9919588234451338
Mean Absolute Error 0.0020044543429844097
Mean Squared Error 0.0020044543429844097
Root Mean Squared Error 0.04477113292049253
-----------------------------------------------------
Accuracy Score:  0.9979955456570155
-----------------------------------------------------
Confusion Matrix:
 [[7082   16]
 [  11 6361]]
-----------------------------------------------------
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      7098
           1       1.00      1.00      1.00      6372

    accuracy                           1.00     13470
   macro avg       1.00      1.00      1.00     13470
weighted avg       1.00      1.00      1.00     13470
```

MultinomialNB Classifier

```python
from sklearn.naive_bayes import MultinomialNB

# Checking accuracy for MultinomialNB Classifier
MNB = MultinomialNB()
MNB.fit(X_train,Y_train)

# [Prediction]
predMNB = MNB.predict(X_test)
print('R2 Score:',r2_score(Y_test,predMNB))

# Mean Absolute Error(MAE)
print('Mean Absolute Error',mean_absolute_error(Y_test,predMNB))

# Mean Squared Error(MSE)
print('Mean Squared Error',mean_squared_error(Y_test,predMNB))

# Root Mean Squared Error (RMSE)
print('Root Mean Squared Error',np.sqrt(mean_squared_error(Y_test,predMNB)))
print("-----------------------------------------------------")
# Accuracy Score
print('Accuracy Score: ',accuracy_score(Y_test, predMNB))
print("-----------------------------------------------------")
# Confusion Matrix
print('Confusion Matrix:\n',confusion_matrix(Y_test, predMNB))
print("-----------------------------------------------------")
# Classification Report
print(classification_report(Y_test,predMNB))
```

Output: -

```
R2 Score: 0.7298760320272706
Mean Absolute Error 0.06733481811432813
Mean Squared Error 0.06733481811432813
Root Mean Squared Error 0.25948953372791
-----------------------------------------------------
Accuracy Score:  0.9326651818856718
-----------------------------------------------------
Confusion Matrix:
 [[6673  425]
 [ 482 5890]]
-----------------------------------------------------
              precision    recall  f1-score   support

           0       0.93      0.94      0.94      7098
           1       0.93      0.92      0.93      6372

    accuracy                           0.93     13470
   macro avg       0.93      0.93      0.93     13470
weighted avg       0.93      0.93      0.93     13470
```

BernoulliNB

```python
from sklearn.naive_bayes import BernoulliNB

# Checking accuracy for BernoulliNB Classifier
BNB = BernoulliNB()
BNB.fit(X_train,Y_train)

# [Prediction]
predBNB = BNB.predict(X_test)
print('R2 Score:',r2_score(Y_test,predBNB))

# Mean Absolute Error(MAE)
print('Mean Absolute Error',mean_absolute_error(Y_test,predBNB))

# Mean Squared Error(MSE)
print('Mean Squared Error',mean_squared_error(Y_test,predBNB))

# Root Mean Squared Error (RMSE)
print('Root Mean Squared Error',np.sqrt(mean_squared_error(Y_test,predBNB)))
print("-----------------------------------------------------")
# Accuracy Score
print('Accuracy Score: ',accuracy_score(Y_test, predBNB))
print("-----------------------------------------------------")
# Confusion Matrix
print('Confusion Matrix:\n',confusion_matrix(Y_test, predBNB))
print("-----------------------------------------------------")
# Classification Report
print(classification_report(Y_test,predBNB))
```

Output: -

```
R2 Score: 0.8528762511813359
Mean Absolute Error 0.036674090571640686
Mean Squared Error 0.036674090571640686
Root Mean Squared Error 0.19150480560978278
-----------------------------------------------------
Accuracy Score:  0.9633259094283593
-----------------------------------------------------
Confusion Matrix:
 [[6850  248]
 [ 246 6126]]
-----------------------------------------------------
              precision    recall  f1-score   support

           0       0.97      0.97      0.97      7098
           1       0.96      0.96      0.96      6372

    accuracy                           0.96     13470
   macro avg       0.96      0.96      0.96     13470
weighted avg       0.96      0.96      0.96     13470
```

Extra Trees Classifier

```python
from sklearn.ensemble import ExtraTreesClassifier

# Checking accuracy for Extra Trees Classifier
ETC = ExtraTreesClassifier()
ETC.fit(X_train,Y_train)

# [Prediction]
predETC = ETC.predict(X_test)
print('R2 Score:',r2_score(Y_test,predETC))

# Mean Absolute Error(MAE)
print('Mean Absolute Error',mean_absolute_error(Y_test,predETC))

# Mean Squared Error(MSE)
print('Mean Squared Error',mean_squared_error(Y_test,predETC))

# Root Mean Squared Error (RMSE)
print('Root Mean Squared Error',np.sqrt(mean_squared_error(Y_test,predETC)))
print("-----------------------------------------------------")
# Accuracy Score
print('Accuracy Score: ',accuracy_score(Y_test, predETC))
print("-----------------------------------------------------")
# Confusion Matrix
print('Confusion Matrix:\n',confusion_matrix(Y_test, predETC))
print("-----------------------------------------------------")
# Classification Report
print(classification_report(Y_test,predETC))
```

Output: -

```
R2 Score: 0.9669418297188832
Mean Absolute Error 0.00824053452115813
Mean Squared Error 0.00824053452115813
Root Mean Squared Error 0.09077738992259102
-----------------------------------------------------
Accuracy Score:  0.9917594654788419
-----------------------------------------------------
Confusion Matrix:
 [[7013   85]
 [  26 6346]]
-----------------------------------------------------
              precision    recall  f1-score   support

           0       1.00      0.99      0.99      7098
           1       0.99      1.00      0.99      6372

    accuracy                           0.99     13470
   macro avg       0.99      0.99      0.99     13470
weighted avg       0.99      0.99      0.99     13470
```

AdaBoostClassifier

```
from sklearn.ensemble import AdaBoostClassifier
ADA = AdaBoostClassifier()
ADA.fit(X_train,Y_train)

# [Prediction]
predADA = ADA.predict(X_test)
print('R2 Score:',r2_score(Y_test,predADA))

# Mean Absolute Error(MAE)
print('Mean Absolute Error',mean_absolute_error(Y_test,predADA))

# Mean Squared Error(MSE)
print('Mean Squared Error',mean_squared_error(Y_test,predADA))

# Root Mean Squared Error (RMSE)
print('Root Mean Squared Error',np.sqrt(mean_squared_error(Y_test,predADA)))
print("-----------------------------------------------------")
# Accuracy Score
print('Accuracy Score: ',accuracy_score(Y_test, predADA))
print("-----------------------------------------------------")
# Confusion Matrix
print('Confusion Matrix:\n',confusion_matrix(Y_test, predADA))
print("-----------------------------------------------------")
# Classification Report
print(classification_report(Y_test,predADA))
```

Output: -

```
R2 Score: 0.9818328974130799
Mean Absolute Error 0.004528582034149963
Mean Squared Error 0.004528582034149963
Root Mean Squared Error 0.06729474001844396
-----------------------------------------------------
Accuracy Score:  0.99547141796585
-----------------------------------------------------
Confusion Matrix:
 [[7066   32]
 [  29 6343]]
-----------------------------------------------------
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      7098
           1       0.99      1.00      1.00      6372

    accuracy                           1.00     13470
   macro avg       1.00      1.00      1.00     13470
weighted avg       1.00      1.00      1.00     13470
```

Support Vector Machine Classifier

```
from sklearn.svm import SVC

# Checking accuracy for Support Vector Machine Classifier
svc = SVC()
svc.fit(X_train,Y_train)

# [Prediction]
predsvc = svc.predict(X_test)
print('R2 Score:',r2_score(Y_test,predsvc))

# Mean Absolute Error(MAE)
print('Mean Absolute Error',mean_absolute_error(Y_test,predsvc))

# Mean Squared Error(MSE)
print('Mean Squared Error',mean_squared_error(Y_test,predsvc))

# Root Mean Squared Error (RMSE)
print('Root Mean Squared Error',np.sqrt(mean_squared_error(Y_test,predsvc)))
print("-----------------------------------------------------")
# Accuracy Score
print('Accuracy Score: ',accuracy_score(Y_test, predsvc))
print("-----------------------------------------------------")
# Confusion Matrix
print('Confusion Matrix:\n',confusion_matrix(Y_test, predsvc))
print("-----------------------------------------------------")
# Classification Report
print(classification_report(Y_test,predsvc))
```

Output: -

```
R2 Score: 0.9696222219038386
Mean Absolute Error 0.007572383073496595
Mean Squared Error 0.007572383073496595
Root Mean Squared Error 0.0870194407790389
-----------------------------------------------------
Accuracy Score:  0.9924276169265034
-----------------------------------------------------
Confusion Matrix:
 [[7041   57]
 [  45 6327]]
-----------------------------------------------------
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      7098
           1       0.99      0.99      0.99      6372

    accuracy                           0.99     13470
   macro avg       0.99      0.99      0.99     13470
weighted avg       0.99      0.99      0.99     13470
```

Cross Validation Score

```python
from sklearn.model_selection import cross_val_score

#cv score for Logistic Regression
print('Logistic Regression',cross_val_score(LOR,X,Y,cv=5).mean())

# cv score for Random Forest Classifier
print('Random Forest Classifier',cross_val_score(RFC,X,Y,cv=5).mean())

# cv score for KNeighbors Classifier
print('BernoulliNB Classifier:',cross_val_score(BNB,X,Y,cv=5).mean())

# cv score for Support Vector  Classifier
print('Support Vector  Classifier',cross_val_score(svc,X,Y,cv=5).mean())

# cv score for Extra Trees Classifier
print('Extra Trees Classifier:',cross_val_score(ETC,X,Y,cv=5).mean())

# cv score for Naive Bias Classifier
print('MultinomialNB Classifier:',cross_val_score(MNB,X,Y,cv=5).mean())

# cv score for AdaBoosting Classifier
print('AdaBoosting Classifier:',cross_val_score(ADA,X,Y,cv=5).mean())
```

```
Logistic Regression 0.9881509218118694
Random Forest Classifier 0.9980623012716382
BernoulliNB Classifier: 0.962225519530724
Support Vector  Classifier 0.9938972822257132
Extra Trees Classifier: 0.9922936865058312
MultinomialNB Classifier: 0.9325360833283105
AdaBoosting Classifier: 0.9953672947840928
```

Hyperparameter Tuning

```
parameters = {'criterion' : ['gini','entropy'],
              'random_state' : [10, 50, 1000],
              'max_depth' : [0, 10, 20],
              'n_jobs' : [-2, -1, 1],
              'n_estimators' : [50,100, 200, 300]}
```

```
from sklearn.model_selection import GridSearchCV
GCV=GridSearchCV(RandomForestClassifier(),parameters,cv=3)
GCV.fit(X_train,Y_train)
```

```
GridSearchCV(cv=3, estimator=RandomForestClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [0, 10, 20],
                         'n_estimators': [50, 100, 200, 300],
                         'n_jobs': [-2, -1, 1],
                         'random_state': [10, 50, 1000]})
```

```
GCV.best_params_
```

```
{'criterion': 'entropy',
 'max_depth': 20,
 'n_estimators': 100,
 'n_jobs': -2,
 'random_state': 1000}
```

```
Fack_news= RandomForestClassifier(criterion='entropy', max_depth=20, n_estimators=100, n_jobs=-2, random_state=1000)
Fack_news.fit(X_train, Y_train)
pred = Fack_news.predict(X_test)
acc=accuracy_score(Y_test,pred)
print('After HyperParameter tuning we have received an accuracy score of',acc*100)
```

```
After HyperParameter tuning we have received an accuracy score of 99.74016332590942
```

CONCLUSION

- ## Key Findings and Conclusions of the Study

The finding of the study is that when the news's are being published on a bogus name, the author names not available that news are end up being Fake, and also, we can understand this fake news's are desperately being spread among the public to create a fake image of an individual, or to get profit out of it or to destroy the good deeds of the target person.

- ## Learning Outcomes of the Study in respect of Data Science

The universe of "fake news" is much larger than simply false news stories. Some stories may have a nugget of truth, but lack any contextualizing details. They may not include any verifiable facts or sources. Some stories may include basic verifiable facts, but are written using language that is deliberately inflammatory, leaves out pertinent details or only presents one viewpoint. "Fake news" exists within a larger ecosystem of mis- and disinformation. Misinformation is false or inaccurate information that is mistakenly or inadvertently created or spread; the intent is not to deceive. Disinformation is false information that is deliberately created and spread "in order to influence public opinion or obscure the truth". As per our evaluation, we found that lesser number of Authors or bogus names or authors unknown have released fake news. We trained 18000 observations for five context categories using a Random Forest algorithm for context detection. Then, the system classifies the fake news in one of the trained contexts in the text conversation. In our testbed, we observed 52.30% of records have fake news but if we search for the authors names in fake news only 47.70% of the authors spread almost all the fake news. Hence, our proposed approach can identify the Fake news and the authors who spread fake news, as discussed usually on a no source news or on a bogus name these fake news's are spread.

- ## Limitations of this work and Scope for Future Work

The limitation of the study is that this data was taken in a shorter time frame on a current trend which might help us in a prediction for a shorted period of time. So, if the prediction of fake news was done with very old data with our model there are chances that the prediction won't be accurate. Same applies for not immediate future data. So, in such case if we have analysis the trend of the news, and if we split the news category as politics, sports arts, general, local, international then we might get some accurate prediction.