FAKE NEWS PROJECT

Prepared by:

Vikash Kumar Singh

SME Name:

Gulshana Chaudhary

ACKNOWLEDGMENT

I would like to convey my heartfelt gratitude to Flip Robo Technologies for providing me with this wonderful opportunity to work on a Machine Learning project “Fake news Project” and also want to Thank my SME, **Gulshana Chaudhary** for providing the dataset and guiding me to complete this project. This project would not have been accomplished without their help and insights. I would also like to thank my academic “Data Trained Education” and their team who has helped me to learn Machine Learning.

I also references from some websites which are- https://www.youtube.com https://www.kaggle.com , https://www.github.com , https://stackoverflow.com

Working on this project was an incredible experience as I learnt more from this Project during completion.

INTRODUCTION

1. Business Problem Framing

Fake news has become one of the biggest problems of our age. It has serious impact on our online as well as offline discourse. One can even go as far as saying that, to date, fake news poses a clear and present danger to western democracy and stability of the society.

2. Conceptual Background of the Domain Problem

Nowadays fake news spreading like water and people share this information without verifying it. This is often done to further or impose certain ideas and is often achieved with political agendas. For media outlets, the ability to attract viewers to their websites is necessary to generate online advertising revenue. So, it is necessary to detect fake news.

3. Review of Literature

Fake news's simple meaning is to incorporate information that leads people to the wrong path.

4. Motivation for the Problem Undertaken

To build an application which can detect the fake and true news.

Analytical Problem Framing

1. Mathematical/ Analytical Modeling of the Problem

1) Used Panda's Library to save data into csv file
2) Cleaned Data by removing irrelevant features
3) Descriptive Statistics
4) Analyzed correlation
5) Pre-processing of text using NLP processing
6) Used Word Counts Removed Punctuation
7) Replaced extra space
8) Used Character Count Used Character
9) Added and removed stop words
10) Calculated length of sentence
11) Checked the word which are spam message
12) Checked the word which are ham message
13) Converted text into vectors using TF-IDF

2. Data Sources and their formats

The data-set is in csv format: **fake_news.csv** and **true_news.csv**.
Features of this dataset are:

- title
- text (containing news)
- subject
- date

3. Data Pre-processing:

a) Checked Top 5 Rows of both Dataset and Checked Total Numbers of Rows and Column

```
false.head()
```

]:

	title	text	subject	date
0	Donald Trump Sends Out Embarrassing New Yearâ...	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 Obamaâ□□s Na...	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.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

b) Checking the relevant info about the dataset

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44898 entries, 0 to 44897
Data columns (total 5 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   title    44898 non-null  object
 1   text     44898 non-null  object
 2   subject  44898 non-null  object
 3   date     44898 non-null  object
 4   target   44898 non-null  object
dtypes: object(5)
memory usage: 1.7+ MB
```

c) Columns name of the dataset

```
data.columns
```

[9]:
```
Index(['text', 'target', 'length', 'num_words', 'num_sent', 'clean_comments',
       'clean_length'],
      dtype='object')
```

d) Checked for Null Values

```
data.isnull().sum()
```

```
7]: title      0
    text       0
    subject    0
    date       0
    target     0
    dtype: int64
```

e) Concatenating both data frames

```
# Concatenate dataframes
data = pd.concat([false, true]).reset_index(drop = True)
data.shape
```

```
5]: (44898, 5)
```

f) Shuffling the data

```
# Shuffle the data
from sklearn.utils import shuffle
data = shuffle(data)
data = data.reset_index(drop=True)
```

g) Information about Data

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44898 entries, 0 to 44897
Data columns (total 5 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   title    44898 non-null  object
 1   text     44898 non-null  object
 2   subject  44898 non-null  object
 3   date     44898 non-null  object
 4   target   44898 non-null  object
dtypes: object(5)
memory usage: 1.7+ MB
```

h) Describing the data

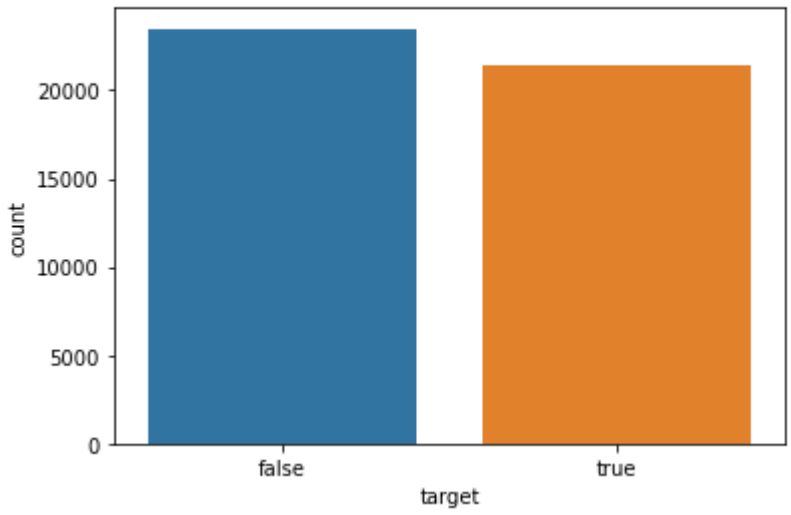
```
data.describe()
```

	target	length	num_words	num_sent	clean_length
count	44898.000000	44898.000000	44898.000000	44898.000000	44898.000000
mean	0.477015	2477.305047	448.240857	14.511270	1634.818299
std	0.499477	2176.946926	391.374585	12.376125	1330.029243
min	0.000000	1.000000	0.000000	0.000000	0.000000
25%	0.000000	1237.000000	224.000000	6.000000	804.000000
50%	0.000000	2191.500000	400.000000	12.000000	1464.000000
75%	1.000000	3113.750000	567.000000	19.000000	2092.000000
max	1.000000	51794.000000	9956.000000	311.000000	37988.000000

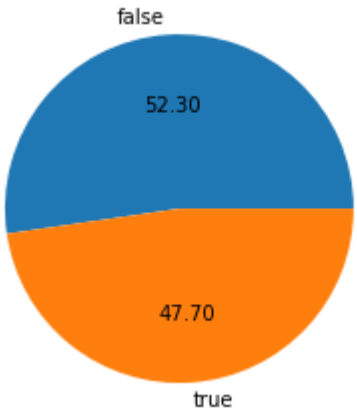
i) Data Visualization

- Count plot

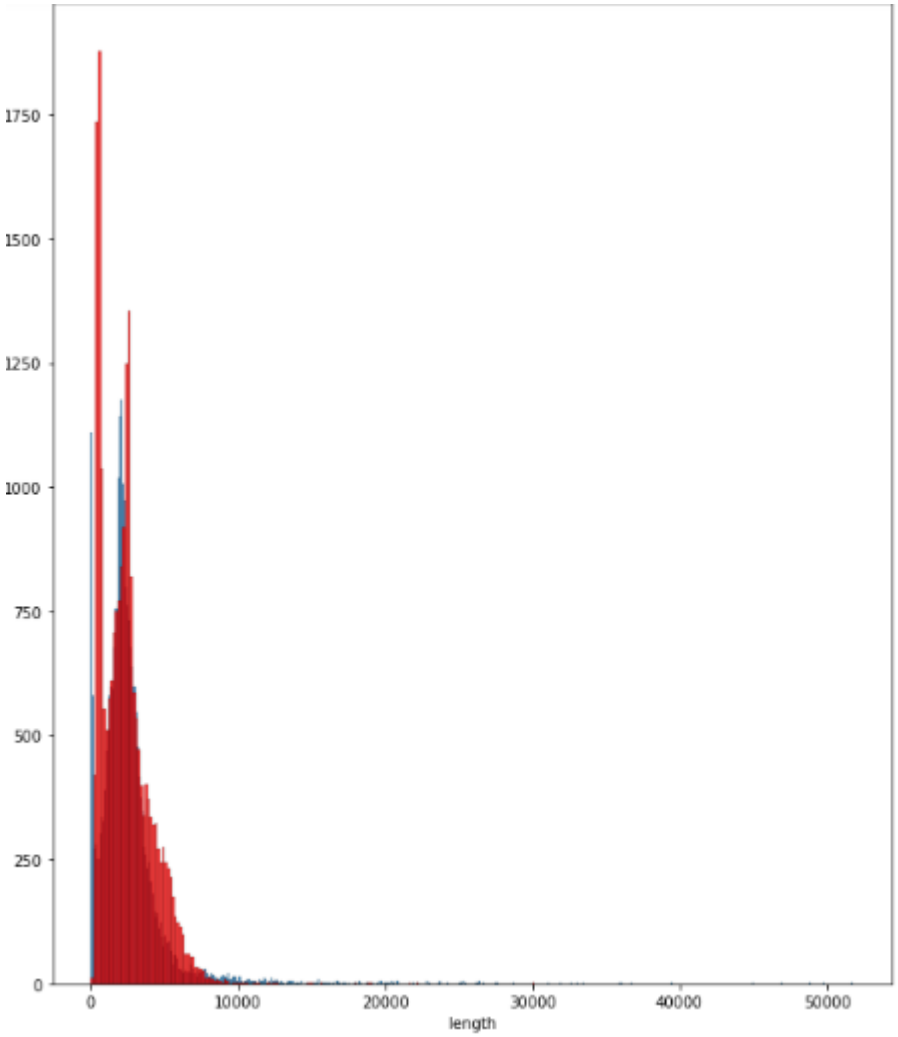
```
<AxesSubplot:xlabel='target', ylabel='count'>
```



- Pie plot



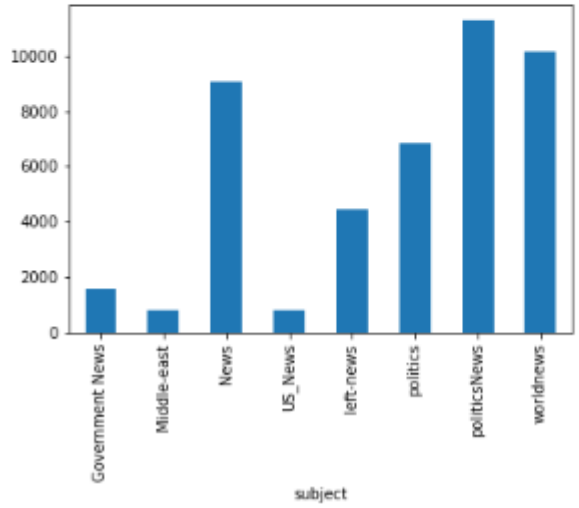
- Histplot



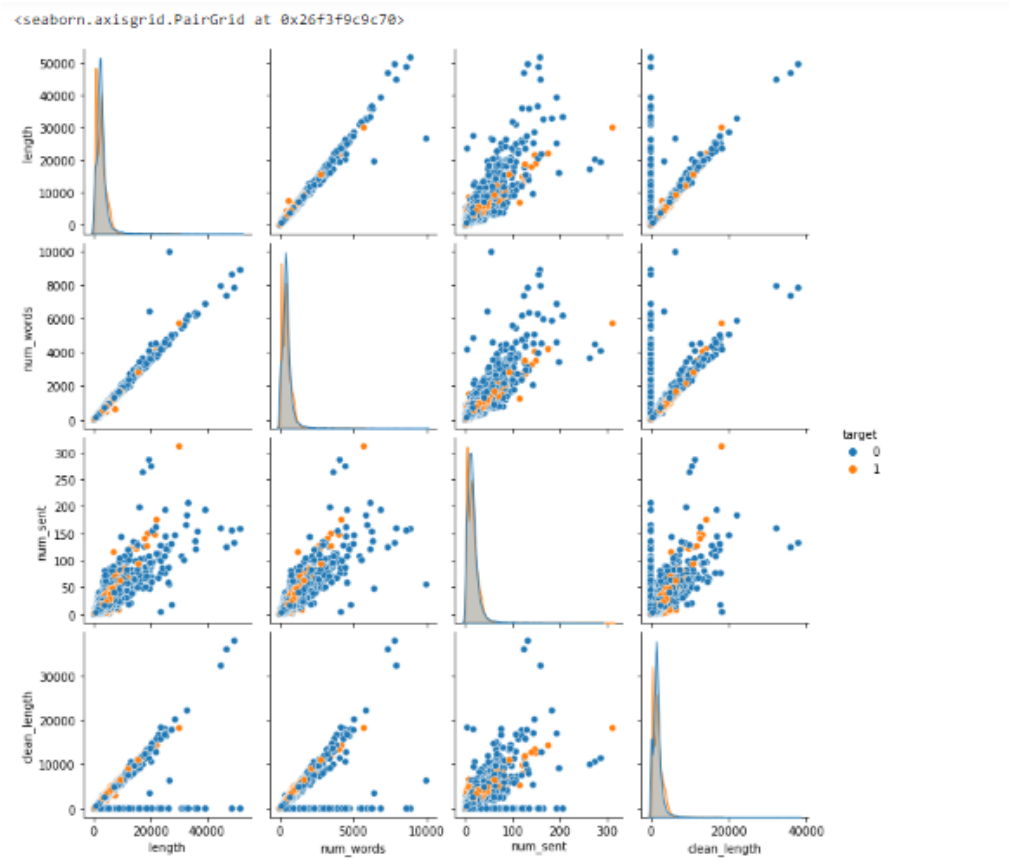
- Group plot

```
# How many articles per subject?
print(data.groupby(['subject'])['text'].count())
data.groupby(['subject'])['text'].count().plot(kind="bar")
plt.show()
```

```
subject
Government News     1570
Middle-east          778
News                9050
US_News              783
left-news           4459
politics            6841
politicsNews       11272
worldnews          10145
Name: text, dtype: int64
```



- Pair plot



4. Data Inputs- Logic- Output Relationships

i Descriptive Statistics

```
data.describe()
```

	target	length	num_words	num_sent	clean_length
count	44898.000000	44898.000000	44898.000000	44898.000000	44898.000000
mean	0.477015	2477.305047	448.240857	14.511270	1634.818299
std	0.499477	2176.946926	391.374585	12.376125	1330.029243
min	0.000000	1.000000	0.000000	0.000000	0.000000
25%	0.000000	1237.000000	224.000000	6.000000	804.000000
50%	0.000000	2191.500000	400.000000	12.000000	1464.000000
75%	1.000000	3113.750000	567.000000	19.000000	2092.000000
max	1.000000	51794.000000	9956.000000	311.000000	37988.000000

ii creating a coloumn which will contain the numbers of characters

```
data['length'] = data['text'].str.len()
data.head()
```

:

	text	target	length
0	donald trump has no real platform other than b...	0	2665
1	pamela gellar is a badass warrior who has a kn...	0	2117
2	san francisco (reuters) - democratic state sen...	1	2595
3	washington (reuters) - u.s. president donald t...	1	3061
4	chicago (reuters) - u.s. farm groups on tuesda...	1	2655

---

iii converted "target" data into binary data i.e.,fake for 0 and true for 1

```
data['target'] = data.target.map({'false':0, 'true':1})
data.head()
```

]:

	text	subject	target
0	Donald Trump has no real platform other than b...	News	0
1	Pamela Gellar is a badass warrior who has a kn...	left-news	0
2	SAN FRANCISCO (Reuters) - Democratic state sen...	politicsNews	1
3	WASHINGTON (Reuters) - U.S. President Donald T...	politicsNews	1
4	CHICAGO (Reuters) - U.S. farm groups on Tuesda...	politicsNews	1

---

iv creating a coloumn which will fetch numbers of words

```
data['num_words'] = data['text'].apply(lambda x: len(nltk.word_tokenize(x)))
data.head()
```

:

	text	target	length	num_words
0	donald trump has no real platform other than b...	0	2665	501
1	pamela gellar is a badass warrior who has a kn...	0	2117	399
2	san francisco (reuters) - democratic state sen...	1	2595	442
3	washington (reuters) - u.s. president donald t...	1	3061	517
4	chicago (reuters) - u.s. farm groups on tuesda...	1	2655	447

---

v creating a coloumn which will fetch numbers of sentences

```
data['num_sent'] = data['text'].apply(lambda x: len(nltk.sent_tokenize(x)))
data.head()
```

:

	text	target	length	num_words	num_sent
0	donald trump has no real platform other than b...	0	2665	501	16
1	pamela gellar is a badass warrior who has a kn...	0	2117	399	8
2	san francisco (reuters) - democratic state sen...	1	2595	442	13
3	washington (reuters) - u.s. president donald t...	1	3061	517	16
4	chicago (reuters) - u.s. farm groups on tuesda...	1	2655	447	13

---

vi Average length of the messages

```
comment_len = data.clean_comments.str.len()
data.clean_comments.str.len().mean()
```

6]: 1634.8182992560917

---

vii Dataset before Pre-processing

	text	target	length	num_words	num_sent
0	donald trump has no real platform other than b...	0	2665	501	16
1	pamela gellar is a badass warrior who has a kn...	0	2117	399	8
2	san francisco (reuters) - democratic state sen...	1	2595	442	13
3	washington (reuters) - u.s. president donald t...	1	3061	517	16
4	chicago (reuters) - u.s. farm groups on tuesda...	1	2655	447	13

---

viii Data Pre-processing

```python
# 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
```

```python
#Defining the stop words
stop_words = stopwords.words('english')

#Defining the lemmatizer
lemmatizer = WordNetLemmatizer()
```

```python
#Replacing '\n' in message
data['text'] = data['text'].replace('\n',' ')
```

```python
#Function Definition for using regex operations and other text preprocessing for getting cleaned texts
def clean_comments(text):

    #convert to lower case
    lowered_text = text.lower()

    #Replacing email addresses with 'emailaddress'
    text = re.sub(r'^.+@[^\.].*\.[a-z]{2,}$', 'emailaddress', lowered_text)

    #Replace URLs with 'webaddress'
    text = re.sub(r'http\S+', 'webaddress', text)

    #Removing numbers
    text = re.sub(r'[0-9]', " ", text)

    #Removing the HTML tags
    text = re.sub(r"<.*?>", " ", text)

    #Removing Punctuations
    text = re.sub(r'[^\w\s]', ' ', text)
    text = re.sub(r'\_',' ',text)

    #Removing all the non-ascii characters
    clean_words = re.sub(r'[^\x00-\x7f]',r'', text)

    #Removing the unwanted white spaces
    text = " ".join(text.split())

    #Splitting data into words
    tokenized_text = word_tokenize(text)

    #Removing remaining tokens that are not alphabetic, Removing stop words and Lemmatizing the text
    removed_stop_text = [lemmatizer.lemmatize(word) for word in tokenized_text if word not in stop_words if word.isalpha()]

    return " ".join(removed_stop_text)
```

```
data
```

	text	target	length	num_words	num_sent	clean_comments
0	donald trump has no real platform other than b...	0	2665	501	16	donald trump real platform afraid afraid trump...
1	pamela gellar is a badass warrior who has a kn...	0	2117	399	8	pamela gellar badass warrior knack exposing tr...
2	san francisco (reuters) - democratic state sen...	1	2595	442	13	san francisco reuters democratic state senator...
3	washington (reuters) - u.s. president donald t...	1	3061	517	16	washington reuters u president donald trump af...
4	chicago (reuters) - u.s. farm groups on tuesda...	1	2655	447	13	chicago reuters u farm group tuesday pushed ba...
...	...	...	...	...	...	...
44893	wow! this is one of the most powerful speeches...	0	1396	270	7	wow one powerful speech ever watch hispanic am...
44894	may 7th is likely going to be a day of clashes...	0	7155	1300	40	may th likely going day clash protester coming...
44895	madrid (reuters) - dismissed catalan leader ca...	1	3196	550	18	madrid reuters dismissed catalan leader carles...
44896	well that didn t take long in the trains betwe...	0	742	127	3	well take long train leipzig chemnitz mrb sett...
44897	does anyone believe for one second that hillar...	0	3767	711	16	anyone believe one second hillary bernie give ...

```
data['clean_length'] = data['clean_comments'].map(lambda clean_comments: len(clean_comments))
data.head()
```

	text	target	length	num_words	num_sent	clean_comments	clean_length
0	donald trump has no real platform other than b...	0	2665	501	16	donald trump real platform afraid afraid trump...	1750
1	pamela gellar is a badass warrior who has a kn...	0	2117	399	8	pamela gellar badass warrior knack exposing tr...	1435
2	san francisco (reuters) - democratic state sen...	1	2595	442	13	san francisco reuters democratic state senator...	1865
3	washington (reuters) - u.s. president donald t...	1	3061	517	16	washington reuters u president donald trump af...	2172
4	chicago (reuters) - u.s. farm groups on tuesda...	1	2655	447	13	chicago reuters u farm group tuesday pushed ba...	1955

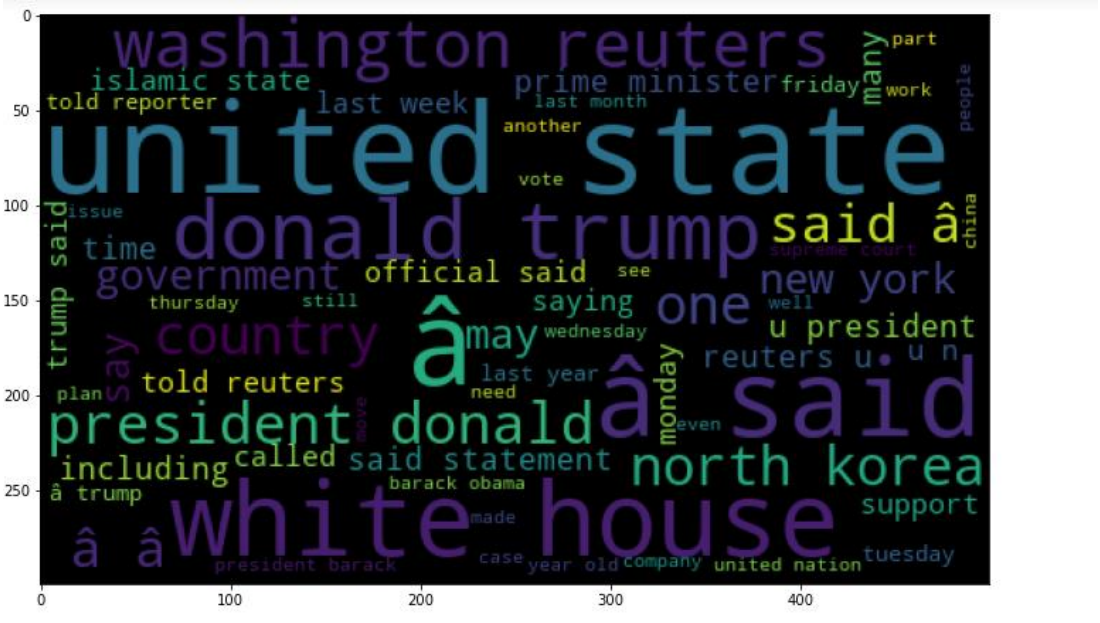
```
data.shape
```

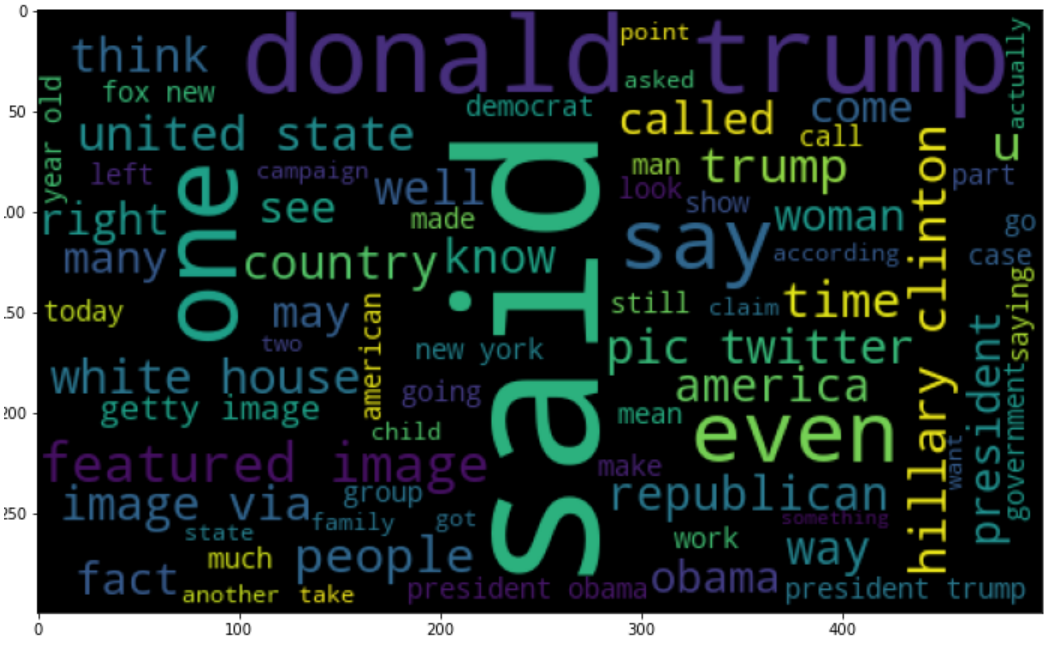
```
(44898, 7)
```

```
print("Original Length:",data.length.sum())
print("Cleaned Length:", data.clean_length.sum())
print("Total Words Removed:", (data.length.sum()) - (data.clean_length.sum()))
```

```
Original Length: 111226042
Cleaned Length: 73400072
Total Words Removed: 37825970
```

ix World cloud





x Convert text into vectors using TF-IDF

```
x = data["clean_comments"]
y = data["target"]
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
#Convert text into vectors using TF-IDF
tf_vec = TfidfVectorizer(max_features = 10000, stop_words='english')
features = tf_vec.fit_transform(data["clean_comments"])
x = features
y=data["target"]
```

5. State the set of assumptions (if any) related to theproblem under consideration

- It was observed that there are two types of news: fake and true. So, have to detect which news is fake and which is true.
- Need to add one more column which is a target column for distinguishing fake and true news by labelling 0 for fake and 1 for true news

- It was observed that title, subject and date column are irrelevant. So, we need to drop them

- It was also observed that text column containing news have stop-words, punctuation so have to replace or pre-process those values.

- Also have to convert text (reviews) into vectors using Tf-Idf vectorization
- By looking into the Target Variable, it is assumed that it is a classification problem

6. Hardware and Software Requirements and Tools Used

- **Hardware tools:**

1. Windows laptop
2. i5 processor
3. 4GB ram 4. 250 GB SSD card

- **Software tools:**

1. windows 10
2. Anaconda Navigator
3. Jupyter Notebook
4. Python

- **Libraries and packages:**

1. Pandas
2. NumPy
3. SciPy
4. Seaborn
5. Mat plot
6. Sklearn

And

```
# 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
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

from sklearn.model_selection import train_test_split, GridSearchCV,cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score,classification_report
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB, MultinomialNB, BernoulliNB
import lightgbm
from sklearn.svm import LinearSVC
from sklearn.linear_model import SGDClassifier
from xgboost import XGBClassifier
import scikitplot as skplt

import nltk
import re
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
import gensim
from gensim.models import Word2Vec
from sklearn.feature_extraction.text import TfidfVectorizer
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer

import joblib
```

Model/s Development and Evaluation

1. Identification of possible problem-solving approaches(methods)

In this project, we want to differentiate between comments and its categories and for this we have used these approaches:

- Checked Total Numbers of Rows and Column
- Checked All Column Name
- Checked Data Type of All Data
- Checked for Null Values
- Checked total number of unique values
- Information about Data
- Checked Description of Data
- Dropped irrelevant Columns

- Checked all features through visualization.
- Checked correlation of features
- Converted all messages to lower case
- Removed Punctuation
- Replaced extra space
- Replaced leading and trailing white space
- Removed \n
- Added and removed stop words
- Words of Sentence
- Calculated length of sentence
- Checked the word which are Ham messages
- Checked the word which are Spam messages
- Converted text into vectors using TF-IDF

Testing of Identified Approaches (Algorithms)

1. Logistic Regression
2. Linear Support Vector Classifier
3. NB_classifier
4. Randomforest Classifier
5. Decisiontree Classifier

2. Run and evaluate selected models

Peparing the data

```
# creating new train test split using the random state.
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.30)
```

```
x.shape, y.shape
```

```
5]: ((44898, 10000), (44898,))
```

```
x_train.shape,y_train.shape, x_test.shape,y_test.shape
```

```
6]: ((31428, 10000), (31428,), (13470, 10000), (13470,))
```

1. Logistic Regression

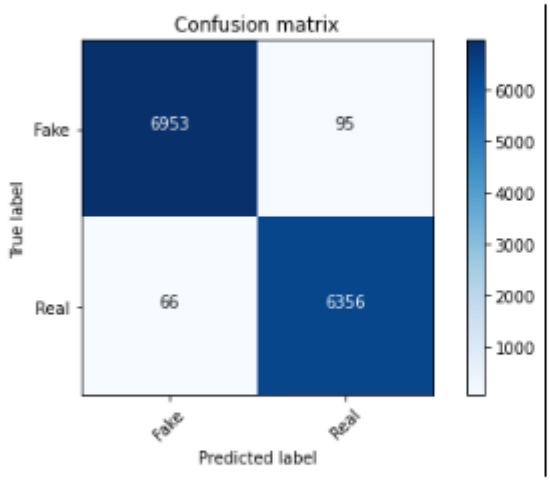
```
lr=LogisticRegression()
lr.fit(x_train,y_train)
pred_lr=lr.predict(x_test)

print("accuracy_score: ", accuracy_score(y_test, pred_lr))
print("confusion_matrix: \n", confusion_matrix(y_test, pred_lr))
print("classification_report: \n", classification_report(y_test,pred_lr))
```

```
accuracy_score:  0.9880475129918337
confusion_matrix:
 [[6953   95]
 [  66 6356]]
classification_report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      7048
           1       0.99      0.99      0.99      6422

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

Confusion Matrix for Logistic Regression

```
cm = metrics.confusion_matrix(y_test, pred_lr)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```



Cross Validation Score for Logistic Regression

```
print('CV score for Logistic Regression: ',cross_val_score(lr,x,y,cv=5).mean())
```

```
CV score for Logistic Regression:  0.9888858933651422
```

2.Naive Bayes

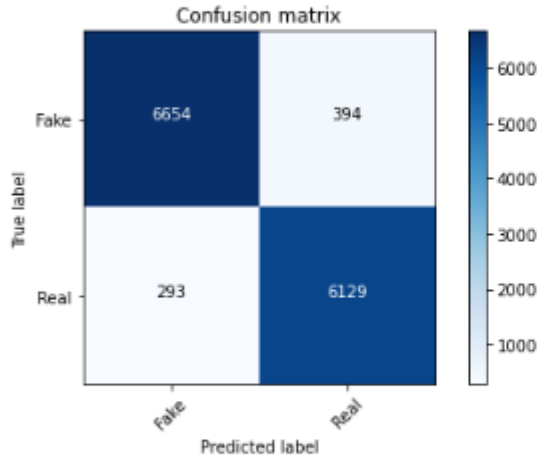
```
from sklearn.naive_bayes import MultinomialNB
NB_classifier = MultinomialNB()

NB_classifier.fit(x_train,y_train)
pred_nb=NB_classifier.predict(x_test)

print("accuracy_score: ", accuracy_score(y_test, pred_nb))
print("confusion_matrix: \n", confusion_matrix(y_test, pred_nb))
print("classification_report: \n", classification_report(y_test,pred_nb))
```

```
accuracy_score:  0.9489977728285078
confusion_matrix:
 [[6654  394]
 [ 293 6129]]
classification_report:
               precision    recall  f1-score   support

           0       0.96      0.94      0.95      7048
           1       0.94      0.95      0.95      6422

    accuracy                           0.95     13470
   macro avg       0.95      0.95      0.95     13470
weighted avg       0.95      0.95      0.95     13470
```

Confusion Matrix for Naive Bayes

```
cm = metrics.confusion_matrix(y_test, pred_nb)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```



Cross Validation Score for Naive Bayes

```
print('CV score for Naive Bayes: ',cross_val_score(NB_classifier,x,y,cv=5).mean())
```

```
CV score for Naive Bayes:  0.9507771809054087
```

3.DecisionTreeClassifier

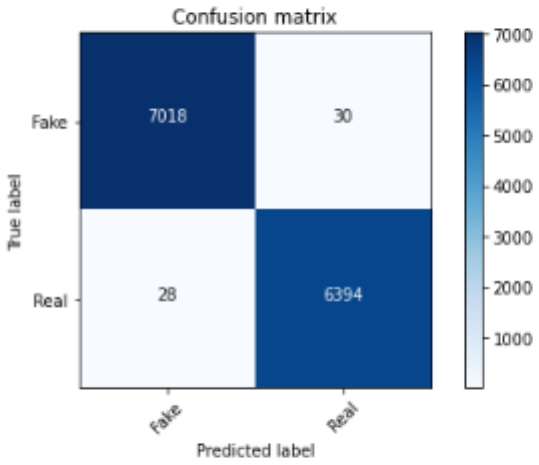
```
from sklearn.tree import DecisionTreeClassifier
dtc=DecisionTreeClassifier(criterion= 'entropy',max_depth = 20, splitter='best', random_state=42)
dtc.fit(x_train,y_train)
pred_dtc=dtc.predict(x_test)

print("accuracy_score: ", accuracy_score(y_test, pred_dtc))
print("confusion_matrix: \n", confusion_matrix(y_test, pred_dtc))
print("classification_report: \n", classification_report(y_test,pred_dtc))
```

```
accuracy_score:  0.9956941351150705
confusion_matrix:
 [[7018   30]
 [  28 6394]]
classification_report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      7048
           1       1.00      1.00      1.00      6422

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

Confusion Matrix for DecisionTreeClassifier

```
cm = metrics.confusion_matrix(y_test, pred_dtc)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```



Cross Validation Score for DecisionTreeClassifier

```
print('CV score forDecisionTreeClassifier: ',cross_val_score(dtc,x,y,cv=5).mean())
```

```
CV score forDecisionTreeClassifier:  0.9961468420127041
```

4. RandomForestClassifier

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier(n_estimators=50, criterion="entropy")
rfc.fit(x_train,y_train)
pred_rfc=dtc.predict(x_test)

print("accuracy_score: ", accuracy_score(y_test, pred_rfc))
print("confusion_matrix: \n", confusion_matrix(y_test, pred_rfc))
print("classification_report: \n", classification_report(y_test,pred_rfc))
```

```
accuracy_score:  0.9956941351150705
confusion_matrix:
 [[7018   30]
 [  28 6394]]
classification_report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00      7048
           1       1.00      1.00      1.00      6422

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

Confusion Matrix for RandomForestClassifier

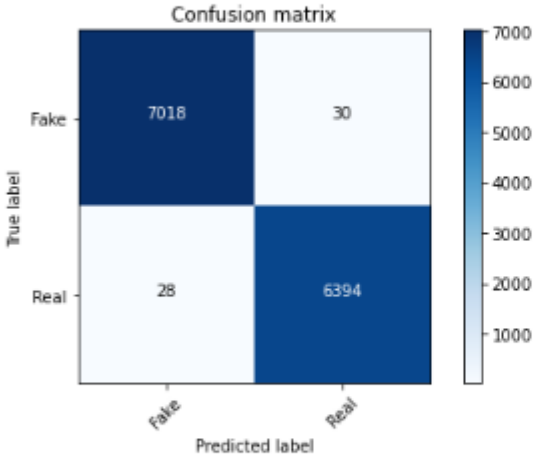
```
cm = metrics.confusion_matrix(y_test, pred_rfc)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```

Confusion Matrix for RandomForestClassifier

```
cm = metrics.confusion_matrix(y_test, pred_rfc)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```



Cross Validation Score for RandomForestClassifier

```
print('CV score for RandomForestClassifier: ',cross_val_score(rfc,x,y,cv=5).mean())
```

```
CV score for RandomForestClassifier:  0.997171350820809
```

5.LinearSVC

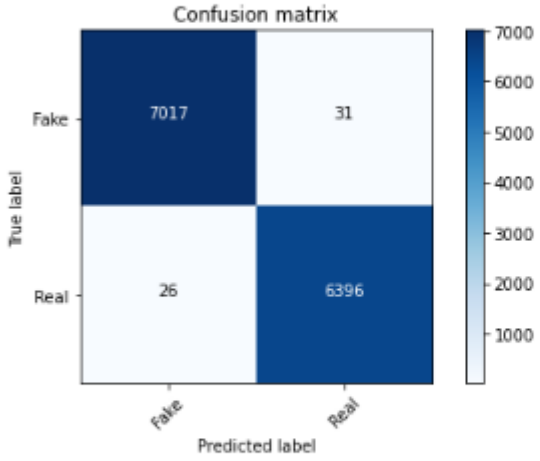
```
svc = LinearSVC()
svc.fit(x_train,y_train)
pred_svc=svc.predict(x_test)

print("accuracy_score: ", accuracy_score(y_test, pred_svc))
print("confusion_matrix: \n", confusion_matrix(y_test, pred_svc))
print("classification_report: \n", classification_report(y_test,pred_svc))
```

```
accuracy_score:  0.9957683741648107
confusion_matrix:
 [[7017   31]
 [  26 6396]]
classification_report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00      7048
           1       1.00      1.00      1.00      6422

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

Confusion Matrix for LinearSVC

```
cm = metrics.confusion_matrix(y_test, pred_svc)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

```
Confusion matrix, without normalization
```



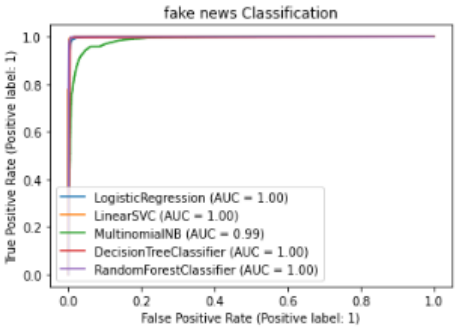
Cross Validation Score for LinearSVC

```
print('CV score for LinearSVC: ',cross_val_score(svc,x,y,cv=5).mean())
```

```
CV score for LinearSVC:  0.9955677005316289
```

ROC & AUC Curve for all model

```
#Lets plot roc curve and check auc and performance of all algorithms
from sklearn.metrics import roc_curve, auc, roc_auc_score, accuracy_score,classification_report, confusion_matrix, plot_roc_c
disp = plot_roc_curve(lr, x_test, y_test)
plot_roc_curve(svc, x_test, y_test, ax = disp.ax_)
plot_roc_curve(NB_classifier, x_test, y_test, ax = disp.ax_)
plot_roc_curve(dtc, x_test, y_test, ax = disp.ax_)
plot_roc_curve(rfc, x_test, y_test, ax = disp.ax_)
plt.title("fake news Classification")
plt.legend(prop={"size" :10} ,loc = 'lower left')
plt.show()
```



Saving the Model ¶

```
joblib.dump(rfc, "fake_news_Detection_Classifier.pkl")
```

```
16]: ['fake_news_Detection_Classifier.pkl']
```

Checking predicted and original values

```
Model = joblib.load("fake_news_Detection_Classifier.pkl")
# Predicting test data using loaded model
prediction = Model.predict(x_test)
# Analysing Predicted vs Actual results
fake_news_Detection_Classifier = pd.DataFrame()
fake_news_Detection_Classifier['Predicted fake Messages Detection'] = prediction
fake_news_Detection_Classifier['Actual fake Messages Detection'] = y
fake_news_Detection_Classifier
```

[17]:

	Predicted Spam Messages Detection	Actual Spam Messages Detection
0	1	0
1	0	0
2	0	1
3	0	1
4	0	1
...	...	...
13465	0	0
13466	0	1
13467	0	1
13468	0	0
13469	0	1

13470 rows × 2 columns

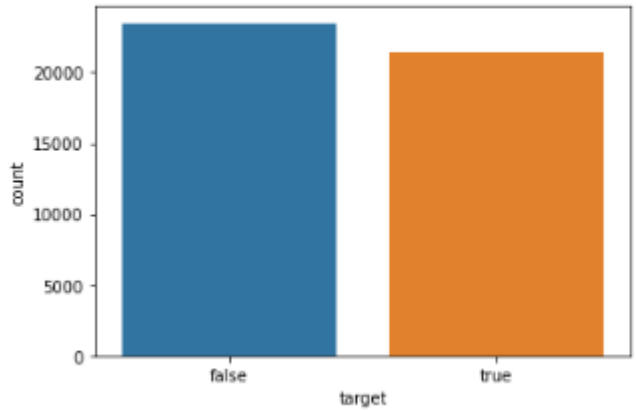
Converting the dataframe into CSV format and saving it

```
fake_news_Detection_Classifier.to_csv('fake_news_Detection_Classifier.csv', index=False)
```

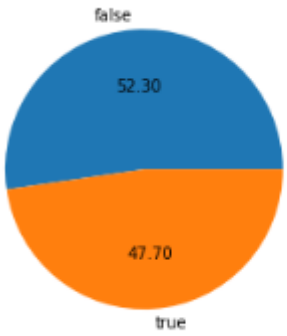
- **Key Metrics for success in solving problem under consideration**
  - Accuracy Score, Precision Score, Recall Score, F1-Score and CV score are used for success. Also, confusion matrix and AUC-ROCCurve is used for success.

- **Visualizations**
  - Count plot

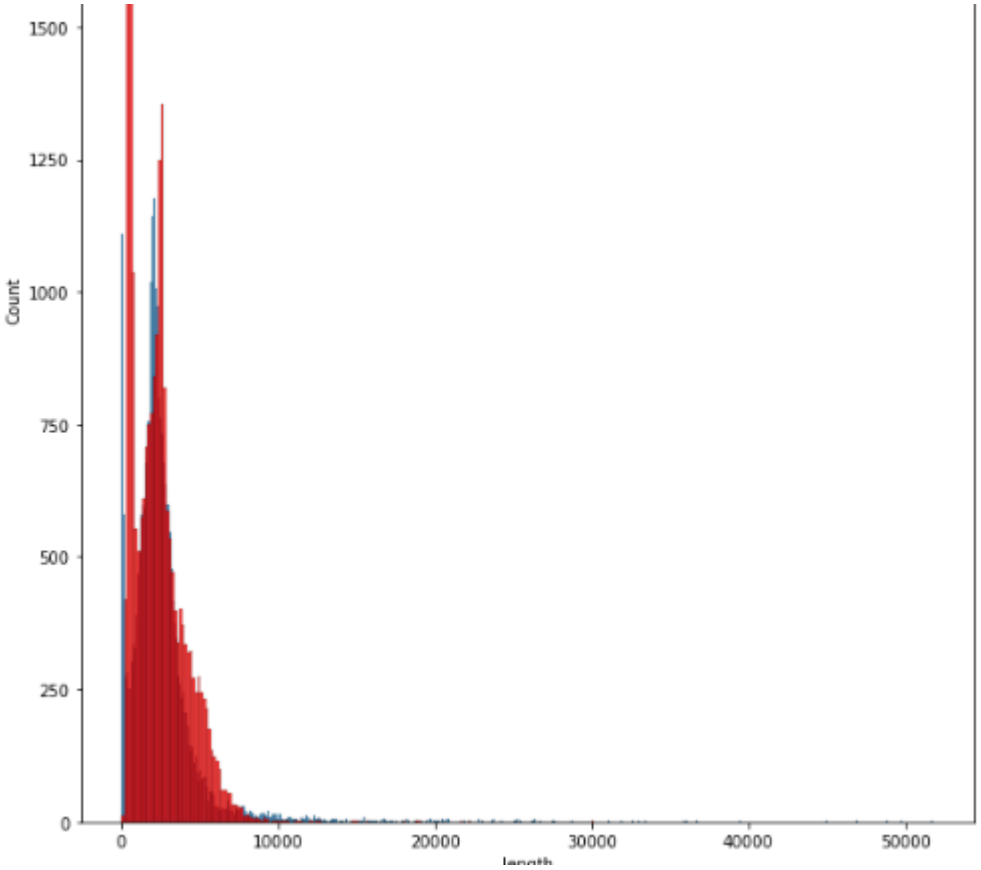
```
<AxesSubplot:xlabel='target', ylabel='count'>
```



- Pie-plot

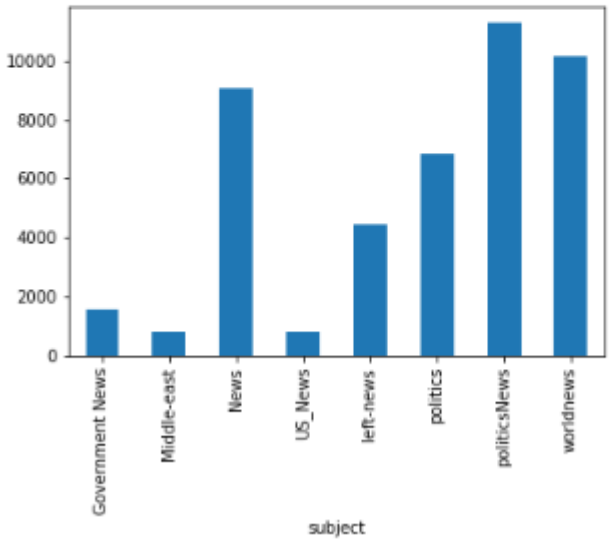


- Histplot

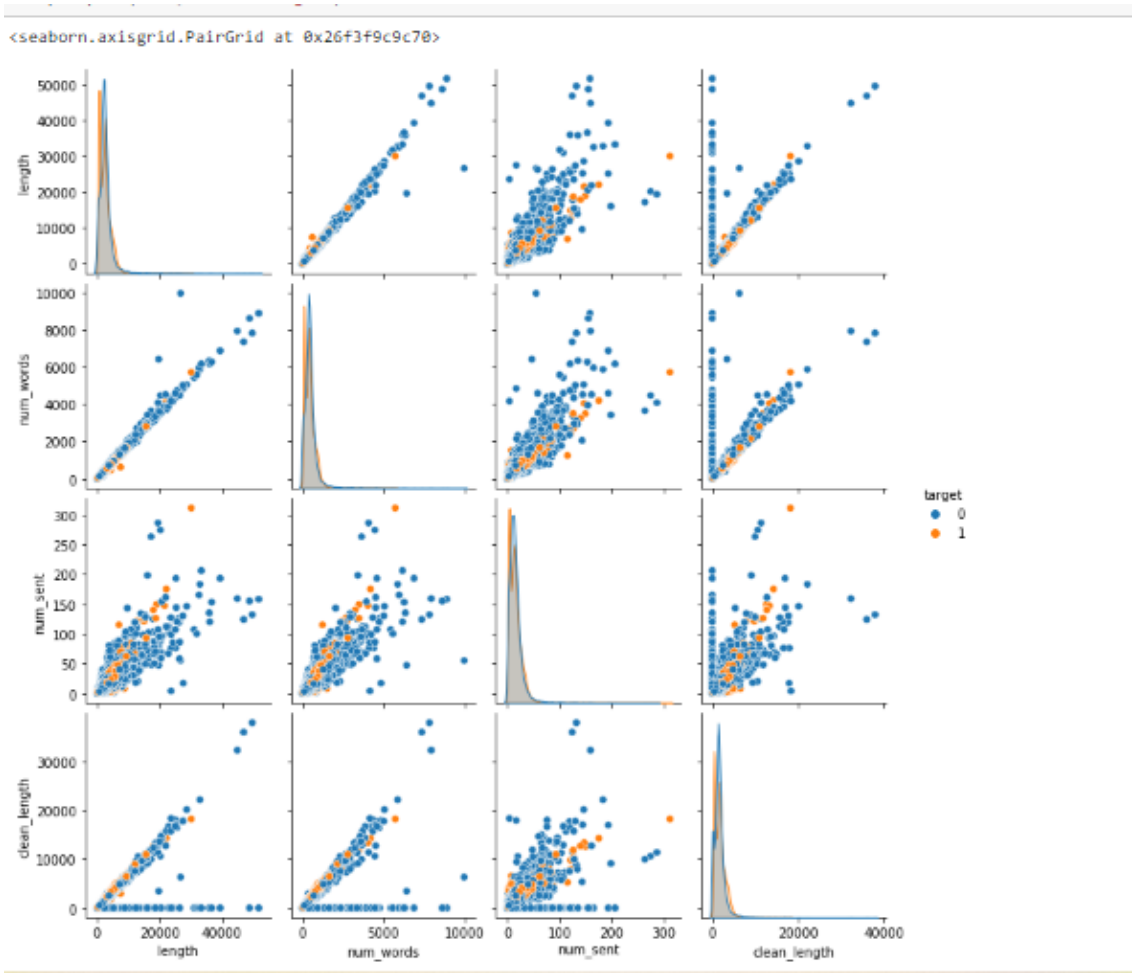


- Groupby plot

```
subject
Government News     1570
Middle-east          778
News                9050
US_News              783
left-news           4459
politics            6841
politicsNews       11272
worldnews          10145
Name: text, dtype: int64
```



- Pair plot



- **Interpretation of the Results**

  - Through Pre-processing it is interpretated that all text are converted to lower case, removed Punctation, replaced extra space, removed stop-words, Calculated length of sentence, words and characters, converted text using Counter-Vectorize.
  - Natural Language Processing and Machine Learning is used in this project
  - Used 5 Machine Learning Algorithms for choosing one best model which is giving best accuracy than others
  - By creating/building model we get best model: Linear SVC

CONCLUSION

1. Key Findings and Conclusions of the Study

In this project we have detected which news are fake news and which are true news. Then we have done different text process to eliminate problem of imbalance. By doing different EDA steps we have analyzed the text.

We have checked frequently occurring words in our data as well as rarely occurring words. After all these steps we have built function to train and test different algorithms and using various evaluation metrics we have selected Randomforest classifier for our final model.

2. Learning Outcomes of the Study in respect of Data Science

- This project has demonstrated the importance of NLP.
- Through different powerful tools of visualization, we were able to analyze and interpret the huge data and with the help of pie plot, count plot & word cloud, I am able to see ham and spam messages.
- Through data cleaning we were able to remove unnecessary columns, values, special characters, symbols, stop-words and punctuation from our dataset due to which our model would have suffered from over fitting or under fitting.

The few challenges while working on this project were: -

- To find punctuations & stop words, which took time to run using NLP.
- The data set is huge it took time to run some algorithms & to check thecross-validation score.

3. Limitations of this work and Scope for Future Work

As we know there are two types of messages to. So, it is difficult to detect with higher accuracies. Still, we can improve our accuracy by fetching more data and by doing extensive hyperparameter tuning.