FAKE NEWS PROJECT

Report by:

Abhishek Ranjan

SME Name:

Mohd. Kashif

ACKNOWLEDGMENT

I would like to convey my immense gratitude to Flip Robo Technologies for providing me this wonderful opportunity to work on a Machine Learning project “Fake News Project” and also want to thank my SME “Mohd. Kashif” for providing the dataset and directions to complete this project. This would not have been accomplished without his 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.

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 Modelling of the Problem

1) Cleaned Data by removing irrelevant features
2) Pre-processing of text using NLP processing
3) Used Word Counts
4) Used Character Counts
5) Used TF-IDF Vectorizer
6) Split data into train and test
7) Built Model
8) Hyper parameter tunning

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

1.

```
fake_news.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

2.

```
true_news.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

b) Checked Total Numbers of Rows and Column

```
fake_news.shape
```

```
(21417, 4)
```

```
true_news.shape
```

```
(23481, 4)
```

c) Checked All Column Name

```
fake_news.columns
```

```
Index(['title', 'text', 'subject', 'date'], dtype='object')
```

```
true_news.columns
```

```
Index(['title', 'text', 'subject', 'date'], dtype='object')
```

d) Checked Data Type of All Data

```
fake_news.dtypes
```

```
title      object
text       object
subject    object
date       object
```

```
true_news.dtypes
```

```
title      object
text       object
subject    object
date       object
```

e) Checked for Null Values

```
fake_news.isnull().sum()
```

```
title      0
text       0
subject    0
date       0
```

```
true_news.isnull().sum()
```

```
title      0
text       0
subject    0
date       0
```

f) Creating one column 'label' in both dataset

```
fake_news['label']= 0
```

```
#checking dataset again after adding one more column
fake_news.head()
```

	title	text	subject	date	label
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017	0
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017	0
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017	0
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017	0
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017	0

```
true_news['label']= 1
```

```
#checking dataset again after adding one more column
true_news.head()
```

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

g) Describing Data

```
fake_news.describe()
```

	title	text	subject	date
count	21417	21417	21417	21417
unique	20826	21192	2	716
top	Factbox: Trump fills top jobs for his administ...	(Reuters) - Highlights for U.S. President Dona...	politicsNews	December 20, 2017
freq	14	8	11272	182

```
true_news.describe()
```

	title	text	subject	date
count	23481	23481	23481	23481
unique	17903	17455	6	1681
top	MEDIA IGNORES Time That Bill Clinton FIRED His...		News	May 10, 2017
freq	6	626	9050	46

h) Combining/Merging both dataset

```
news= pd.concat([fake_news, true_news], ignore_index=True)
```

```
#checking the dataset after combining
news
```

	title	text	subject	date	label
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017	0
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017	0
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017	0
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017	0
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017	0

i) Data cleaning

- Dropped Column " title, date, subject " as these columns are irrelevant for prediction.

```
news.drop(['title', 'date', 'subject'], axis=1, inplace=True)
```

j) Data Information

```
news.info()
```

```
RangeIndex: 44898 entries, 0 to 44897
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    44898 non-null  object
 1   label   44898 non-null  int64 
dtypes: int64(1), object(1)
memory usage: 701.7+ KB
```

4. Data Inputs- Logic- Output Relationships

I. Text Pre-Processing

Creating a function to process the texts

```
def wordopt(text):
    text = text.lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub("\\W"," ",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
```

```
news['text']=news['text'].apply(wordopt)
```

```
news.head()
```

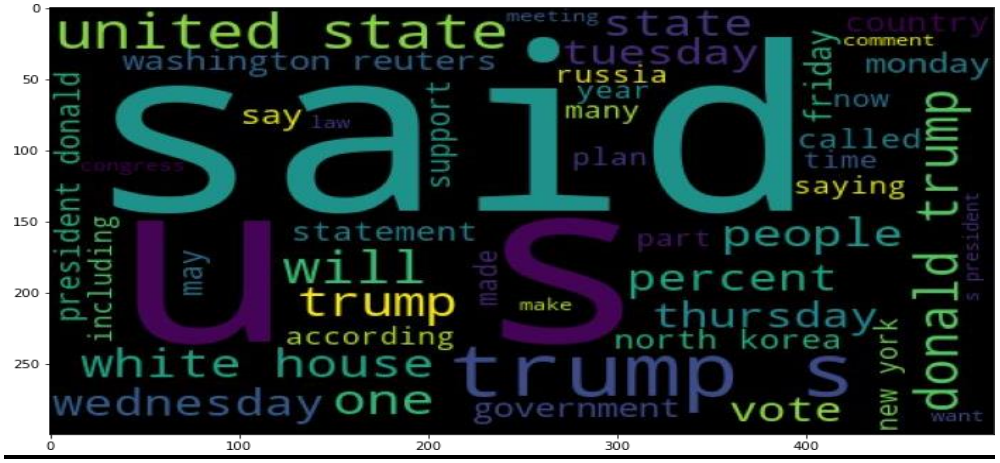
	text	label
0	washington reuters the head of a conservat...	0
1	washington reuters transgender people will...	0
2	washington reuters the special counsel inv...	0
3	washington reuters trump campaign adviser ...	0
4	seattle washington reuters president donal...	0

Wordcloud

```
wc = WordCloud(width = 500, height = 300, min_font_size= 10, background_color= 'black')
```

```
#Generating Word Cloud for True News
true_wordcloud = wc.generate(news[news['label']==1]['text'].str.cat(sep = " "))
plt.figure(figsize=(12,8))
plt.imshow(true_wordcloud)
plt.show()
```

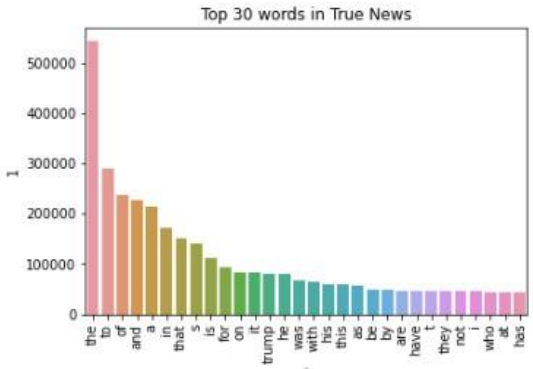
```
#Generating Word Cloud for Fake News
fake_wordcloud = wc.generate(news[news['label']==0]['text'].str.cat(sep = " "))
plt.figure(figsize=(12,8))
plt.imshow(fake_wordcloud)
plt.show()
```



Top 30 most frequently occuring words

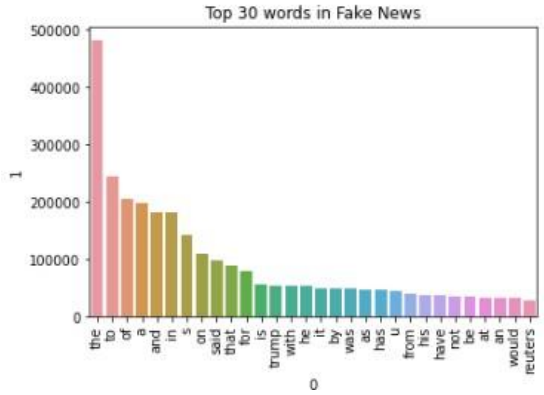
```
true_corpus = []
for msg in news[news['label']==1]['text'].tolist():
    for word in msg.split():
        true_corpus.append(word)
```

```
sns.barplot(pd.DataFrame(Counter(true_corpus).most_common(30))[0] , pd.DataFrame(Counter(true_corpus).most_common(30))[1])
plt.title("Top 30 words in True News")
plt.xticks(rotation = "vertical")
plt.show
```



```
fake_corpus = []
for msg in news[news['label']==0]['text'].tolist():
    for word in msg.split():
        fake_corpus.append(word)
```

```
sns.barplot(pd.DataFrame(Counter(fake_corpus).most_common(30))[0] , pd.DataFrame(Counter(fake_corpus).most_common(30))[1])
plt.title("Top 30 words in Fake News")
plt.xticks(rotation = "vertical")
plt.show
```



Convert text to vectors using TfidfVectorizer

```
vectorizer = TfidfVectorizer(max_features=3000)
message_mat = vectorizer.fit_transform(news['text'])
message_mat
```

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 used:**
  - **Processor**: 11th Gen Intel(R) Core (TM) i3-1125G4 @2.00GHz 2.00 GHz
  - **System Type**: 64-bit OS

- **Software used:**
  - **Anaconda** for 64-bit OS
  - **Jupyter** notebook

- **Tools, Libraries and Packages used:**

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

```
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
```

```
from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
```

```
from collections import Counter
import scikitplot as skplt

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
- Description of Data
- Dropped irrelevant Columns
- Added one target column to distinguish which new is fake and which is true
- Replaced special characters and irrelevant data
- Checked all features through visualization.
- Converted all messages to lower case
- Removed special characters
- Removed punctuations
- Removed Stop-Words
- Used Word Counts
- Used Character Counts
- Checked loud word using Word-Cloud
- Converted text into vectors using TF-IDF Vectorization

2. Testing of Identified Approaches (Algorithms)

1. Logistic Regression
2. Decision Tree Classifier
3. Gradient Boosting Classifier
4. Random Forest Classifier
5. Linear Support Vector Classifier
6. Bernoulli NB
7. Multinomial NB
8. SGD Classifier
9. LGBM Classifier

10. XGB Classifier

3. Run and evaluate selected models

Creating Model

Finding the best random state

```
maxAccu = 0
maxRs = 0
for i in range(1,100):
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=i)
    LR = LogisticRegression()
    LR.fit(x_train,y_train)
    pred_LR = LR.predict(x_test)
    acc = accuracy_score(y_test,pred_LR)
    if acc>maxAccu:
        maxAccu=acc
        maxRs = i
print(f'Best Accuracy is {maxAccu} on Random_state {maxRs}')
```

```
Best Accuracy is 0.9892204899777283 on Random_state 34
```

Splitting into train and test

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=maxRs)
```

```
# Lets check the shapes of traning and test data
print("x_train", x_train.shape)
print("x_test", x_test.shape)
print("y_train", y_train.shape)
print("y_test", y_test.shape)
```

```
x_train (33673, 3000)
x_test (11225, 3000)
y_train (33673,)
y_test (11225,)
```

```
# Defining the Classification Machine Learning Algorithms
lr = LogisticRegression(solver='lbfgs')
dtc = DecisionTreeClassifier()
gbc = GradientBoostingClassifier(random_state=0)
rfc = RandomForestClassifier(random_state=0)
svc = LinearSVC()
bnb = BernoulliNB()
mnb = MultinomialNB()
sgd = SGDClassifier()
lgb = LGBMClassifier()
xgb = XGBClassifier(verbosity=0)
```

```
# Creating a function to train and test the model with evaluation metrics
def BuiltModel(model):
    print('*'*30+model.__class__.__name__+'*'*30)
    model.fit(x_train, y_train)
    y_pred = model.predict(x_train)
    # Prediction
    pred = model.predict(x_test)

    # Accuracy Score
    accuracy = accuracy_score(y_test, pred)*100
    print(f"ACCURACY SCORE PERCENTAGE:", accuracy)

    # Mean Absolute Error(MAE)
    print('Mean Absolute Error(MAE)',mean_absolute_error(y_test,pred))

    # Mean Squared Error(MSE)
    print('Mean Squared Error',mean_squared_error(y_test,pred))

    # Root Mean Squared Error (RMSE)
    print('Root Mean Squared Error',np.sqrt(mean_squared_error(y_test,pred)))

    # Classification report
    print(f"CLASSIFICATION REPORT: \n {classification_report(y_test, pred)}")

    # Confusion matrix and
    print(f"CONFUSION MATRIX: \n {confusion_matrix(y_test, pred)}\n")

    print("-"*120)
    print("\n")
```

Training and testing of all the classification algorithms

```
for model in [lr,dtc,gbc,rfc,svc,bnb,mnb,sgd,lgb]:
    BuiltModel(model)
```

```
**********DecisionTreeClassifier**********
ACCURACY SCORE PERCENTAGE: 99.52783964365256
Mean Absolute Error(MAE) 0.004721603563474387
Mean Squared Error 0.004721603563474387
Root Mean Squared Error 0.06871392554260299
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       1.00      0.99      1.00      5412
           1       0.99      1.00      1.00      5813

    accuracy                           1.00     11225
   macro avg       1.00      1.00      1.00     11225
weighted avg       1.00      1.00      1.00     11225

CONFUSION MATRIX:
 [[5379   33]
 [  20 5793]]

------------------------------------------------------------
**********LogisticRegression**********
ACCURACY SCORE PERCENTAGE: 98.92204899777283
Mean Absolute Error(MAE) 0.010779510022271715
Mean Squared Error 0.010779510022271715
Root Mean Squared Error 0.10382441920026192
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      5412
           1       0.99      0.99      0.99      5813

    accuracy                           0.99     11225
   macro avg       0.99      0.99      0.99     11225
weighted avg       0.99      0.99      0.99     11225

CONFUSION MATRIX:
 [[5358   54]
 [  67 5746]]

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

**********GradientBoostingClassifier**********
**********RandomForestClassifier**********
ACCURACY SCORE PERCENTAGE: 99.82182628062361
Mean Absolute Error(MAE) 0.0017817371937639199
Mean Squared Error 0.0017817371937639199
Root Mean Squared Error 0.04221062891931272
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5412
           1       1.00      1.00      1.00      5813

    accuracy                           1.00     11225
   macro avg       1.00      1.00      1.00     11225
weighted avg       1.00      1.00      1.00     11225

CONFUSION MATRIX:
 [[5402   10]
 [  10 5803]]

------------------------------------------------------------
```

```
*****************************LinearSVC*********************************
ACCURACY SCORE PERCENTAGE: 99.456570155902
Mean Absolute Error(MAE) 0.005434298440979955
Mean Squared Error 0.005434298440979955
Root Mean Squared Error 0.07371769421909474
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      5412
           1       0.99      0.99      0.99      5813

    accuracy                           0.99     11225
   macro avg       0.99      0.99      0.99     11225
weighted avg       0.99      0.99      0.99     11225

CONFUSION MATRIX:
 [[5382   30]
 [  31 5782]]

------------------------------------------------------------------------
```

```
*****************************BernoulliNB*********************************
ACCURACY SCORE PERCENTAGE: 95.94654788418708
Mean Absolute Error(MAE) 0.04053452115812917
Mean Squared Error 0.04053452115812917
Root Mean Squared Error 0.20133186821298105
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.96      0.96      0.96      5412
           1       0.96      0.96      0.96      5813

    accuracy                           0.96     11225
   macro avg       0.96      0.96      0.96     11225
weighted avg       0.96      0.96      0.96     11225

CONFUSION MATRIX:
 [[5198  214]
 [ 241 5572]]

------------------------------------------------------------------------
```

```
*****************************LGBMClassifier*********************************
ACCURACY SCORE PERCENTAGE: 99.77728285077951
Mean Absolute Error(MAE) 0.0022271714922048997
Mean Squared Error 0.0022271714922048997
Root Mean Squared Error 0.04719291781830087
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5412
           1       1.00      1.00      1.00      5813

    accuracy                           1.00     11225
   macro avg       1.00      1.00      1.00     11225
weighted avg       1.00      1.00      1.00     11225

CONFUSION MATRIX:
 [[5404    8]
 [  17 5796]]

------------------------------------------------------------------------
```

```
*****************************MultinomialNB*****************************
ACCURACY SCORE PERCENTAGE: 93.27394209354121
Mean Absolute Error(MAE) 0.06726057906458797
Mean Squared Error 0.06726057906458797
Root Mean Squared Error 0.2593464460226667
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.93      0.93      0.93      5412
           1       0.93      0.94      0.94      5813

    accuracy                           0.93     11225
   macro avg       0.93      0.93      0.93     11225
weighted avg       0.93      0.93      0.93     11225

CONFUSION MATRIX:
 [[5007  405]
 [ 350 5463]]

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


*****************************SGDClassifier*****************************
ACCURACY SCORE PERCENTAGE: 99.23385300668151
Mean Absolute Error(MAE) 0.0076614699331848555
Mean Squared Error 0.0076614699331848555
Root Mean Squared Error 0.08752982310724074
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      5412
           1       0.99      0.99      0.99      5813

    accuracy                           0.99     11225
   macro avg       0.99      0.99      0.99     11225
weighted avg       0.99      0.99      0.99     11225

CONFUSION MATRIX:
 [[5375   37]
 [  49 5764]]

--------------------------------------------------------------------------------
```

Cross validation score for best score models

```
def cross_val(model):
    print('*'*30+model.__class__.__name__+'*'*30)
    scores = cross_val_score(model,x,y, cv = 3).mean()*100
    print("Cross validation score:", scores)
    print("\n")

for model in [lr,dtc,gbc,rfc,svc,bnb,mnb,sgd,lgb]:
    cross_val(model)
```

```
**********LogisticRegression**********
Cross validation score: 96.95754822041071

**********DecisionTreeClassifier**********
Cross validation score: 99.17590984008197

**********GradientBoostingClassifier**********
Cross validation score: 99.34295514276806

**********RandomForestClassifier**********
Cross validation score: 99.4565459485946

**********LinearSVC**********
Cross validation score: 98.48768319301527

**********BernoulliNB**********
Cross validation score: 90.40714508441356

**********MultinomialNB**********
Cross validation score: 88.3669651209408

**********SGDClassifier**********
Cross validation score: 97.69032028152701

**********LGBMClassifier**********
Cross validation score: 99.55900040090873
```

HyperParameter Tuning

Linear SVC with GridSearchCV

```
# Lets select the different parameters for tuning our best model (Linear SVC)
grid_params =  {'C':(0.001, 0.01, 0.1, 1, 10),
                'penalty':('l1','l2'),
               'loss':('hinge','squared_hinge')}

# Train the model with given parameters using GridSearchCV
LSVC =  GridSearchCV(svc, grid_params, cv=3)
LSVC.fit(x_train, y_train)
```

```
GridSearchCV(cv=3, estimator=LinearSVC(),
             param_grid={'C': (0.001, 0.01, 0.1, 1, 10),
                         'loss': ('hinge', 'squared_hinge'),
                         'penalty': ('l1', 'l2')})
```

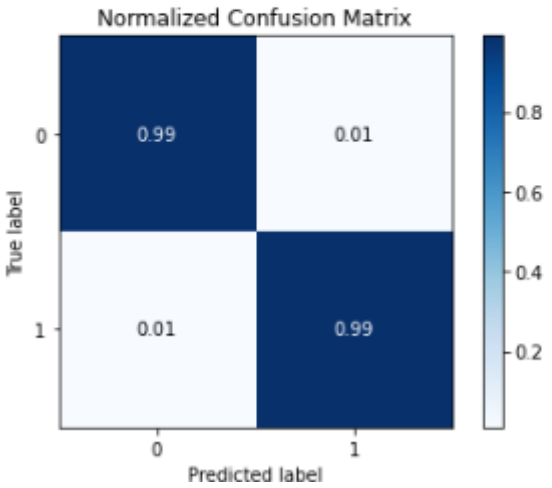
```
# Selecting the best parameters found by GridSearchCV
LSVC.best_params_
```

```
{'C': 1, 'loss': 'squared_hinge', 'penalty': 'l2'}
```

```
# Final Model with the best chosen parameters list
best_model = LinearSVC(C= 1, loss= 'squared_hinge', penalty= '12')
best_model.fit(x_train,y_train) # fitting data to the best model
pred = best_model.predict(x_test)
accuracy = accuracy_score(y_test, pred)*100
# Printing the accuracy score
print("ACCURACY SCORE:", accuracy)
# Printing the classification report
print(f"\nCLASSIFICATION REPORT: \n {classification_report(y_test, pred)}")
# Printing the Confusion matrix
print(f"\nCONFUSION MATRIX: \n {confusion_matrix(y_test, pred)}")
```

```
ACCURACY SCORE: 99.456570155902

CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      5412
           1       0.99      0.99      0.99      5813

    accuracy                           0.99     11225
   macro avg       0.99      0.99      0.99     11225
weighted avg       0.99      0.99      0.99     11225


CONFUSION MATRIX:
 [[5382   30]
 [  31 5782]]
```

```
# Creating a normalized confusion matrix here
skplt.metrics.plot_confusion_matrix(y_test, pred, normalize=True)
```



- **Saving The Predictive Model**

```
joblib.dump(best_model, "Fake_News_Detection_Project.pkl")
```

Loading the final model

```
Model = joblib.load("Fake_News_Detection_Project.pkl")
```

- **Comparing Actual and Predicted**

```
# Predicting test data using loaded model
prediction = Model.predict(x_test)
# Analysing Predicted vs Actual results
Fake_News_Detection_Project = pd.DataFrame()
Fake_News_Detection_Project['Predicted Fake News'] = prediction
Fake_News_Detection_Project['Actual Fake News'] = y
Fake_News_Detection_Project
```

	Predicted Fake News	Actual Fake News
0	1	0
1	1	0
2	0	0
3	0	0
4	1	0
...	...	...
11220	0	0
11221	1	0
11222	0	0
11223	0	0
11224	1	0

11225 rows × 2 columns

- **Saving the model in CSV format**

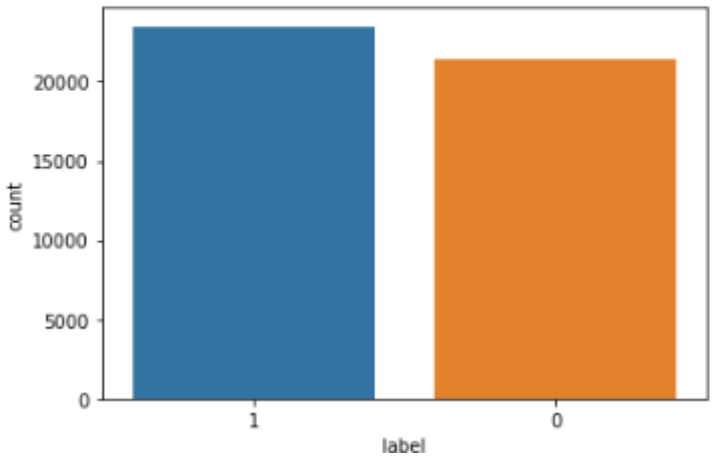
```
# Converting the dataframe into CSV format and saving it
Fake_News_Detection_Project.to_csv('Fake_News_Detection_Project.csv', index=False)
```

4. Key Metrics for success in solving problem underconsideration
    - Accuracy Score, Precision Score, Recall Score, F1-Score and CVscore are used for success. Also, confusion matrix is used for success.

5. Visualization

Using Countplot

```
sns.countplot(data=news, x='label', order=news['label'].value_counts().index)
```



6. Interpretation of the Results

- Through Pre-processing it is interpretated that all text is converted to lower case, removed Punctation, replaced extra space, removed stop-words, Calculated length of sentence, words and characters, converted text using TF-IDF Vectorization.
- Natural Language Processing and Machine Learning is used in this project.
- Used 10 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 differentEDA 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 Linear-SVC for our final model.

Finally, by doing hyperparameter tuning we got optimum parameters for our final model. And finally, we got improved accuracy score for our final model.

2. Learning Outcomes of the Study in respect of DataScience

- 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 count plot & word cloud, I am able to see the distribution of fake and true news.
- Through data cleaning we were able to remove unnecessary columns, values, stop-words and punctuation from our dataset due to which our model would have suffered from overfitting or underfitting.

The few challenges while working on this project were: -

- Using NLP to find punctuations & stop words, it took time in giving theresult.
- The data set took time to run some algorithms & to check the cross-validation score.

3. Limitations of this work and Scope for FutureWork

As we know there are two types of news to detect. 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.