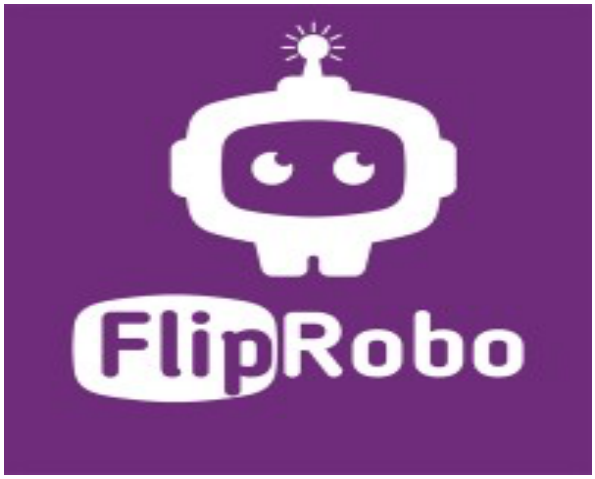
NAME OF THE PROJECT

Fake News Detection Project



Submitted by:

Pratyush Raj

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my SME Mr. Kashif from Flip Robo Technologies for letting me work on this project and providing me with all necessary information and dataset for the project. I would also like to thank my mentor of Data Trained academy for providing me sufficient knowledge so that I can complete the project. I have utilized a few external resources that helped me to complete the project. I ensured that I learn from the samples and modify things according to my project requirement. All the external resources that wereused in creating this project are listed below:

https://www.google.com/ https://www.youtube.com/
https://scikit-learn.org/stable/user_guide.html https://github.com/
https://www.analyticsvidhya.com/

INTRODUCTION

Business Problem Framing

The project is concerned with identifying a solution that could be used todetect and filter out sites containing fake news for purpose of helping users to avoid being lured by clickbait. It is imperative that such solutionsare identified as they will prove to be useful to both readers and tech companies involved in the issue.

Conceptual Background of the DomainProblem

The idea of fake news is not a novel concept. Notably, the idea has been in existence even before the emergence of the Internet as publishers used false and misleading information to further their interests. Followingthe advent of the web, more and more consumers began forsaking the traditional media channels used to disseminate information for online platforms. Not only does the latter alternative allow users to access a variety of publications in one sitting, but it is also more convenience and faster. The development, however, came with a redefined concept of fakenews as content publishers began using what has come to be commonly referred to as a clickbait

Review of Literature

If we look at some scholar work shows the issue that the fake news has been major concerned amongst scholar from various background. For instance, some authors have observed that fake news is no longer a preserve of the marketing and public relations departments. Instead, there is an increasing risk of IT security, therefore, IT department is premised on the idea that it would help avert the various risks associatedwith the problem. So, if we good deeply into it we could find that the hackers use clickbait with the help of fake news and make some professional of the organization downloads their malicious exploits in their system or leak sensitive information, albeit in an indirect manner.

The user may, for instance, be tricked into believing that they are helpingto disseminate the news further when, in the actual sense, they are providing the perpetrators with access to their emails, and we can also see that the fake news are worked extensively as they are using videos with original massage and uses their facial structure to replace the massage with false massage they want us to believe, these fake news issues is bigger day by day and we need to implement more our researchand extensive knowledge to solve the problem.

Motivation for the Problem Undertaken

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

Analytical Problem Framing

Mathematical/ Analytical Modelling of the Problem

Cleaned Data by removing irrelevant features Pre-processing of text using NLP processing Used Word Counts

Used Character Counts

Used Tf-Idf Vectorizer

Split data into train and test Built Model

Hyper parameter tunning

Data Sources and their formats

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

title

text (containing news) subject

date

Data Preprocessing Done

Data pre-processing is the process of converting raw data into a well- readable format to be used by Machine Learning model. Data pre- processing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we pre- process our data before feeding it into our model. I have used following pre- processing steps:

Importing necessary libraries and loading dataset as a data frame.

Used pandas to set display maximum columns ensuring not to find anytruncated information.

Checked some statistical information like shape, number of uniquevalues present, info, finding zero values etc.

Checked for null values and did not find any null values.

Dropped some unwanted columns as they are of no use forprediction.

Added a new label as ‘0’ for fake news dataset and ‘1’ for true newsdataset.

Merge both fake and true dataset to a single dataset as news.

Visualized each feature using seaborn and matplotlib libraries byplotting count plot. Separate feature and label data.

Data Inputs- Logic- Output Relationships

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
```

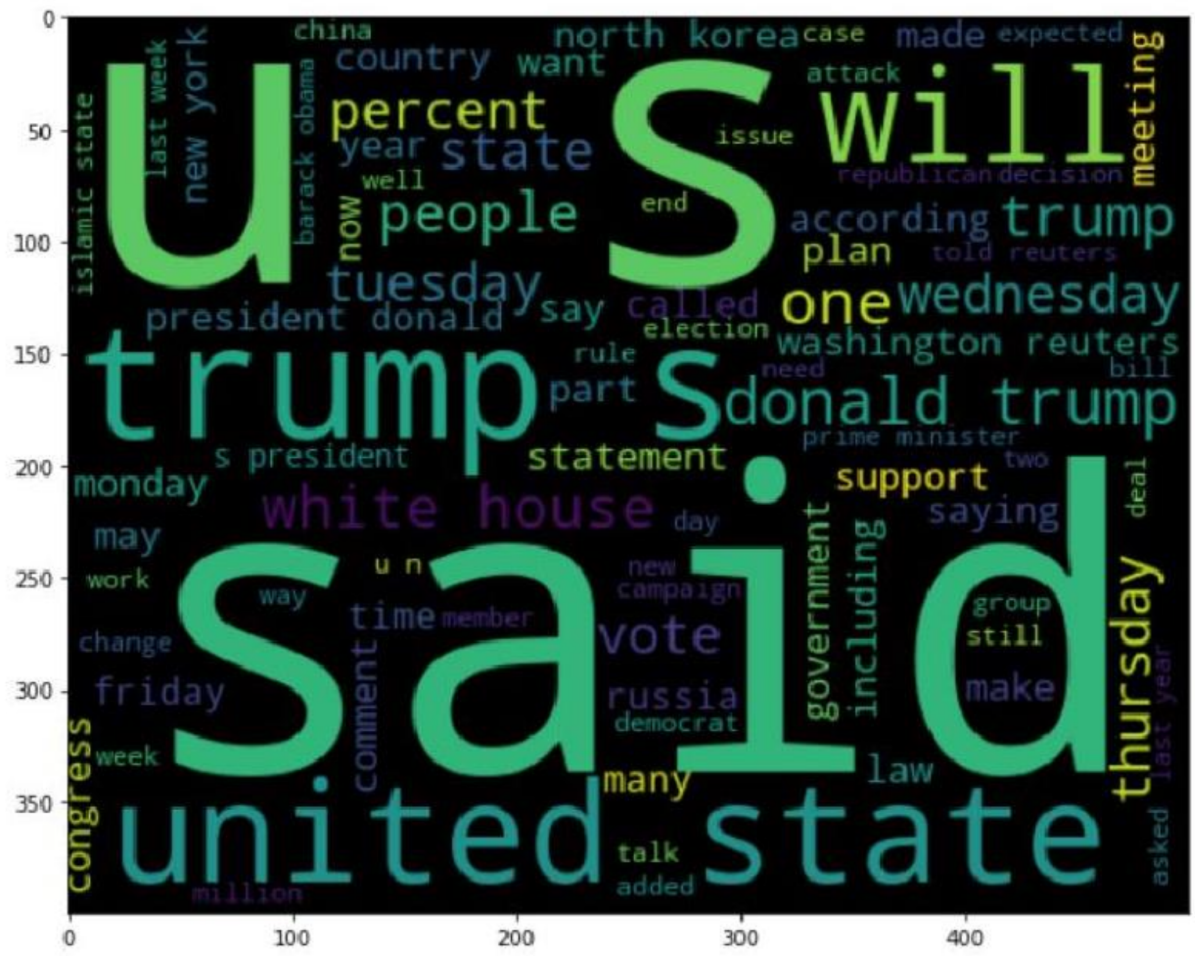
```
news['text']=news['text'].apply(wordopt)
news.head()
```

	text	label
0	donald trump just couldn t wish all americans ...	0
1	house intelligence committee chairman devin nu...	0
2	on friday it was revealed that former milwauk...	0
3	on christmas day donald trump announced that ...	0
4	pope francis used his annual christmas day mes...	0

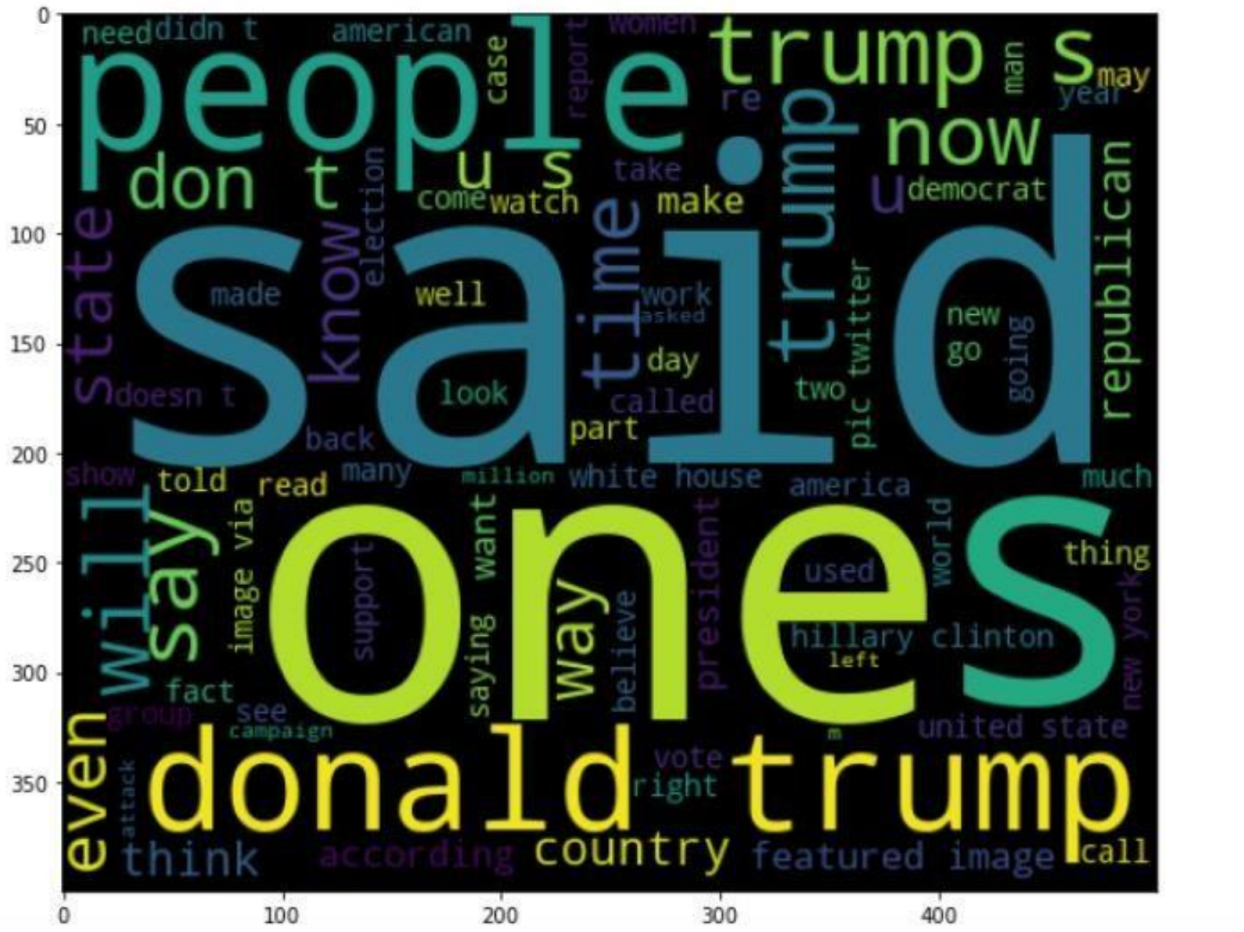
Wordcloud

```
wc = WordCloud(width = 500, height = 400, 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()
```

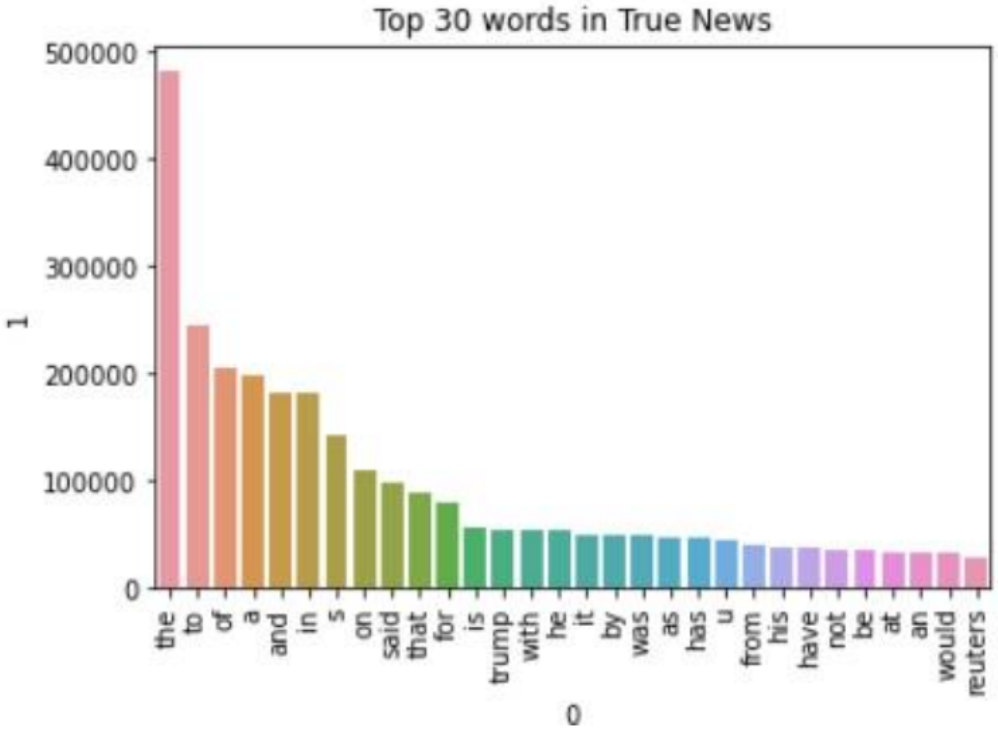



```
#Generating Word Cloud for Fake News
fake_wordcloud = wc.generate(news[news['label']==0]['text'].str.cat(sep = " "))
plt.figure(figsize=(12,8))
plt.imshow(fake_wordcloud)
plt.show()
```



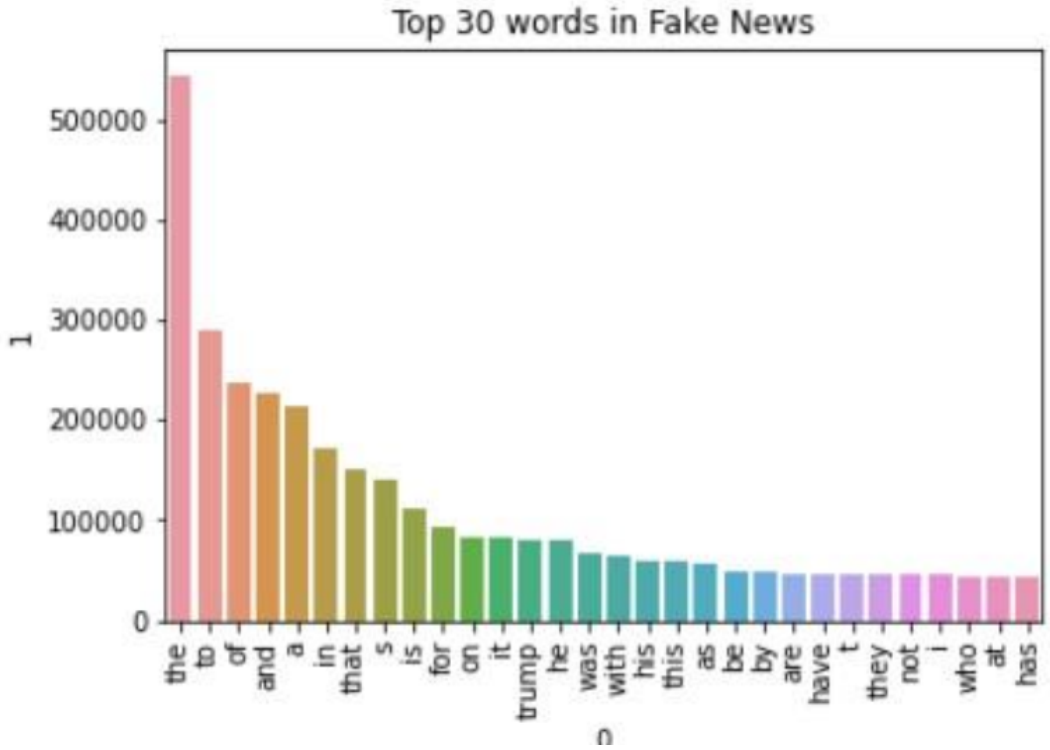

```
#Top 30 most frequently occuring words in True news
true_corpus = []
for msg in news[news['label']==1]['text'].tolist():
    for word in msg.split():
        true_corpus.append(word)
```

```
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
```




```
#Top 30 most frequently occuring words in Fake news

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
```



State the set of assumptions (if any) related tothe problem under consideration

It was observed that there are two types of news: fake andtrue. So, must 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 1for true news.

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

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

Also, must convert text (reviews) into vectors using Tf-Idfvectorization.

By looking into the Target Variable, it is assumed that it is aclassification problem.

Hardware and Software Requirements and Tools Used

Hardware Used:

Processor — core i5 and above

RAM — 8 GB or above

SSD — 250GB or above Software Used:

Programming language: Python

Distribution: Anaconda Navigator

Browser based language shell: JupyterNotebook Libraries/Packages Used: Pandas, NumPy, matplotlib, seaborn, scikit-learn and

pandas_profiling

Model/s Development and Evaluation

Identification of possible problem Solving approaches (methods)

In this project, we want to differentiate between comments and itscategories 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 whichis 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

Testing of Identified Approaches (Algorithms)

Since label is my target variable, which is categorical in nature, from this I can conclude that it is a classification type problem hence I haveused following classification algorithms

Logistic Regression

Decision Tree Classifier Gradient Boosting Classifier Random Forest Classifier Linear Support Vector Classifier Bernoulli NB

Multinomial NB

SGD Classifier

LGBM Classifier

XGB Classifier

Run and evaluate selected models

I used a total of 10 classification Models after choosing therandom state amongst 1-100 number. I have used DecisionTree Classifier to find best random state and the code is as below. The code for the models is listed below.

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 0.98913140311804 on Random_state 32
```

```
# 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)
```

```python
# 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")
```

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

```
*******************************LogisticRegression*******************************
ACCURACY SCORE PERCENTAGE: 98.913140311804
Mean Absolute Error(MAE) 0.01086859688195991
Mean Squared Error 0.01086859688195991
Root Mean Squared Error 0.1042525629515165
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      5861
           1       0.99      0.99      0.99      5364

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

CONFUSION MATRIX:
 [[5794   67]
 [  55 5309]]
```

```
*******************************DecisionTreeClassifier*******************************
ACCURACY SCORE PERCENTAGE: 99.55456570155901
Mean Absolute Error(MAE) 0.004454342984409799
Mean Squared Error 0.004454342984409799
Root Mean Squared Error 0.06674086442659999
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      1.00      1.00      5861
           1       1.00      0.99      1.00      5364

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

CONFUSION MATRIX:
 [[5842   19]
 [  31 5333]]
```

```
******************************GradientBoostingClassifier******************************
ACCURACY SCORE PERCENTAGE: 99.59910913140313
Mean Absolute Error(MAE) 0.004008908685968819
Mean Squared Error 0.004008908685968819
Root Mean Squared Error 0.06331594337896909
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       1.00      0.99      1.00      5861
           1       0.99      1.00      1.00      5364

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

CONFUSION MATRIX:
 [[5830   31]
 [  14 5350]]
```

```
*********************************RandomForestClassifier*****************
ACCURACY SCORE PERCENTAGE: 99.75055679287304
Mean Absolute Error(MAE) 0.0024944320712694877
Mean Squared Error 0.0024944320712694877
Root Mean Squared Error 0.049944289676293205
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5861
           1       1.00      1.00      1.00      5364

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

CONFUSION MATRIX:
 [[5851   10]
 [  18 5346]]
```

```
**********LinearSVC**********
ACCURACY SCORE PERCENTAGE: 99.49220489977728
Mean Absolute Error(MAE) 0.005077951002227171
Mean Squared Error 0.005077951002227171
Root Mean Squared Error 0.07125974320910208
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      1.00      1.00      5861
           1       1.00      0.99      0.99      5364

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

CONFUSION MATRIX:
 [[5839   22]
 [  35 5329]]
```

```
**********BernoulliNB**********
ACCURACY SCORE PERCENTAGE: 96.09799554565701
Mean Absolute Error(MAE) 0.039020044543429847
Mean Squared Error 0.039020044543429847
Root Mean Squared Error 0.1975349198076883
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.97      0.96      0.96      5861
           1       0.96      0.96      0.96      5364

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

CONFUSION MATRIX:
 [[5623  238]
 [ 200 5164]]
```

```
*********************************MultinomialNB*********************************
ACCURACY SCORE PERCENTAGE: 93.21158129175946
Mean Absolute Error(MAE) 0.06788418708240535
Mean Squared Error 0.06788418708240535
Root Mean Squared Error 0.2605459404450688
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.94      0.93      0.93      5861
           1       0.93      0.93      0.93      5364

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

CONFUSION MATRIX:
 [[5477  384]
 [ 378 4986]]

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

```
*********************************SGDClassifier*********************************
ACCURACY SCORE PERCENTAGE: 99.26057906458797
Mean Absolute Error(MAE) 0.0073942093541202675
Mean Squared Error 0.0073942093541202675
Root Mean Squared Error 0.08598958863792912
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      5861
           1       0.99      0.99      0.99      5364

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

CONFUSION MATRIX:
 [[5822   39]
 [  44 5320]]

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

```
*******************************LGBMClassifier*******************************
ACCURACY SCORE PERCENTAGE: 99.74164810690424
Mean Absolute Error(MAE) 0.0025835189309576837
Mean Squared Error 0.0025835189309576837
Root Mean Squared Error 0.05082832803622094
CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5861
           1       1.00      1.00      1.00      5364

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

CONFUSION MATRIX:
 [[5842   19]
 [  10 5354]]
```

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: 99.1135462604125


******************************GradientBoostingClassifier******************************
Cross validation score: 99.35854603768543


******************************RandomForestClassifier******************************
Cross validation score: 99.42536415875985


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


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


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


******************************SGDClassifier******************************
Cross validation score: 97.80391108735355


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

Hyper Parameter 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')})
```

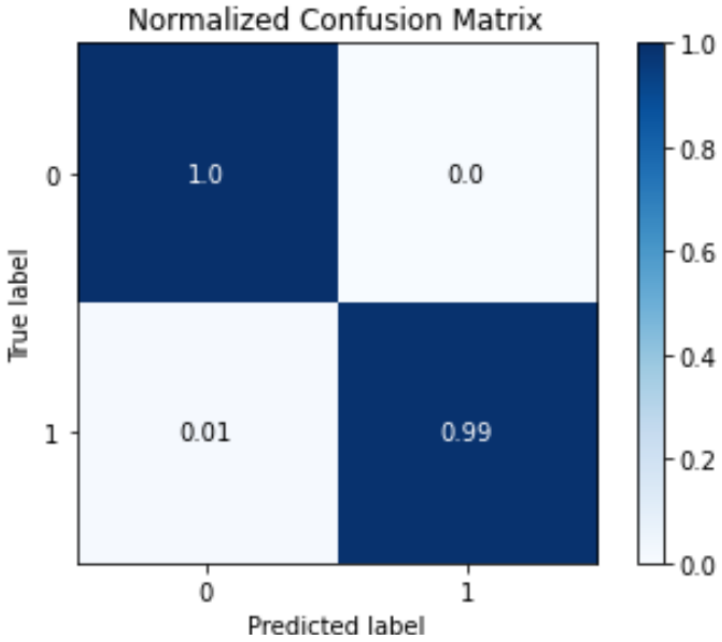
```
# Selecting the best parameters found by GridSearchCV
LSVC.best_params_
```

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

```python
# Final Model with the best chosen parameters list
best_model = LinearSVC(C= 1, loss= 'squared_hinge', penalty= 'l2')
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: 99.49220489977728

CLASSIFICATION REPORT:
              precision    recall  f1-score   support

           0       0.99      1.00      1.00      5861
           1       1.00      0.99      0.99      5364

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


CONFUSION MATRIX:
 [[5839   22]
 [  35 5329]]
```

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



Saving the Final model

```
# Saving the model using joblib library
import joblib
joblib.dump(best_model,"Fake_News_Prediction.pkl")
```

```
['Fake_News_Prediction.pkl']
```

Loading the saved model and predicting

```
# Loading the saved model
Model=joblib.load("Fake_News_Prediction.pkl")

#Prediction
prediction = Model.predict(x_test)
# Analysing Predicted vs Actual results
Fake_News_Detection = pd.DataFrame()
Fake_News_Detection['Predicted Fake News'] = prediction
Fake_News_Detection['Actual Fake News'] = y
Fake_News_Detection
```

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

11225 rows × 2 columns

Key Metrics for success in solving problemunder Consideration

The key metrics used here were Accuracy Score, Precision, Recall, F1 score, Cross Validation Score, Roc AUC Score and Confusion Matrix. We tried to find out the best parameters and to increase our scores byusing Hyperparameter Tuning and used RandomizedSearchCV method.

Accuracy score means how accurate our model is that is the degreeof closeness of the measured value to a standard or true value. It is one metric for evaluating classification models. Accuracy is the ratioof number of correct predictions into number of predictions.

Precision is the degree to which repeated measurements under thesame conditions are unchanged. It is amount of information that is

conveyed by a value. It refers to the data that is correctly classifiedby the classification algorithm.

Recall is how many of the true positives were recalled (found). Recallrefers to the percentage of data that is relevant to the class. In binary

classification problem recall is calculated as below:

Recall = Number of True Positives/ (Total number of True Positives +Total number of False Negatives)

F1 Score is used to express the performance of the machine learning model (or classifier). It gives the combined information about the precision and recall of a model. This means a high F1-score indicates ahigh value for both recall and precision.

Cross Validation Score is a technique in which we train our model usingthe subset of the data- set and then evaluate using the complementary

subset of the data-set. It is used to protect against overfitting in a predictive model, particularly in a case where the amount of data may be limited. In cross-validation, you make a fixed number of folds (or partitions) of the data, run the analysis on each fold, and then average the overall error estimate. It is used to estimate the performance of MLmodels.

Roc Auc Score: The Receiver Operator Characteristic (ROC) curve is anevaluation metric for binary classification problems. It is a probability curve that plots the TPR against FPR at various threshold values.

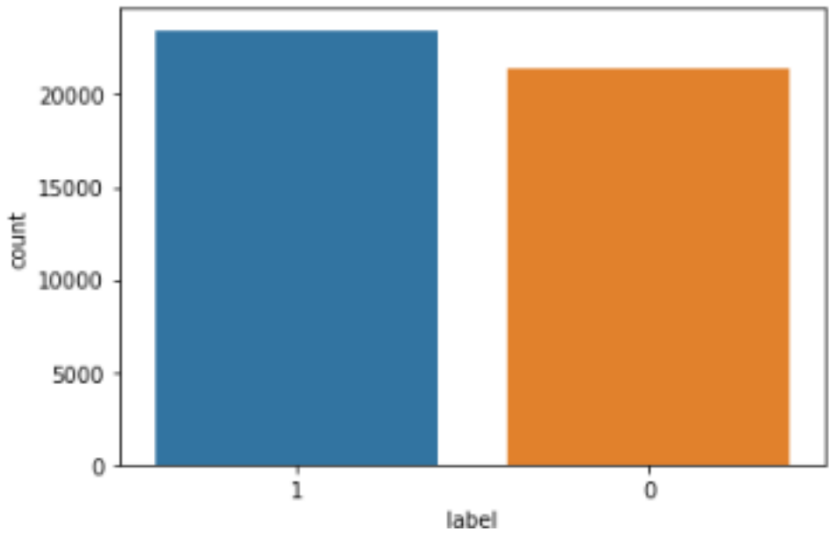
The Area Under Curve (AUC) is the measure of the ability of a classifierto distinguish between classes and is used as a summary of the ROC curve.

Confusion Matrix is one of the evaluation metrics for machine learning classification problems, where a trained model is being evaluated for accuracy and other performance measures. And this matrix is called theconfusion matrix since it results in an output that shows how the system is confused between the two classes.

Hyperparameter Tuning: There is a list of different machine learningmodels. They all are different in some way or the other, but what makes them different is nothing but input parameters for the model.These input parameters are named as Hyperparameters. These hyperparameters will define the architecture of the model, and the best part about these is that you get a choice to select these for yourmodel. You must select from a specific list of hyperparameters for a given model as it varies from model to model.Visualizations

Using Countplot

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



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 isused in this project.

Used 10 Machine Learning Algorithms for choosing onebest model which is giving best accuracy than others.

By creating/building model we get best model: LinearSVC.

CONCLUSION

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 ofimbalance. 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 testdifferent 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 ourfinal model. And finally, we got improved accuracy score for our final model.

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 can 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.

Limitations of this work and Scope for FutureWork

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