南開大學

Nankai University

# COLLEGE OF SOFTWARE ENGINEERING

# Detection of Fake News Using Machine Learning and Natural Language Processing Algorithms

Prepared by:

1) Atinafu Zufan Yalemzewude ID:2120226049
2) G/ Medhin Berehane G/ Kirstos ID: 2120226043
3) Senu Yonas Demewez ID: 2120226038
4) Tadesse Tihitina Miriye ID: 2120226035

Submitted to: Dr. Ling Ma

June 2023

NKU, China

# Abstract

In recent years, the proliferation of fake news has become a serious issue, leading to widespread misinformation and confusion. Identifying fake news has become a crucial task in today's digital age, and machine learning has emerged as a powerful tool for this purpose. In this paper, we propose a machine learning-based approach for identifying fake news, which is capable of automatically analyzing news articles and determining their authenticity. Our system uses a range of features, including the content of the article, the source of the information, and the language used, to classify news articles as either real or fake. We evaluate our system on a large dataset of news articles and show that it achieves high accuracy in detecting fake news. Our system has the potential to be used in a wide range of applications, including social media platforms, news aggregation systems, and search engines, to help users identify fake news and make more informed decisions.

# Table of contents

# List of figures

# 1. Introduction

Fake news refers to deliberately fabricated and misleading information that is presented as if it were true news. It can be spread through various channels such as social media, websites, and even traditional news outlets. Fake news can have serious consequences, as it can impact public opinion and even influence political elections. There are organizations, like the House of Commons and the Crosscheck project, trying to deal with issues as confirming authors are accountable. However, their scope is so limited because they depend on human manual detection, in a globe with millions of articles either removed or being published every minute, this cannot be accountable or feasible manually. A solution could be, by the development of a system to provide a credible automated index scoring, or rating for credibility of different publishers, and news context.

In this project, a method is proposed for building a model that can determine if an article is real or fraudulent based on its words, phrases, sources, and titles. The methodology involves using supervised machine learning algorithms on a dataset that has been manually categorized and guaranteed to be annotated. Then, based on the results of the confusion matrix, feature selection techniques are used to explore and pick the best-fit features to get the highest precision. We suggest utilizing various categorization algorithms to build the model. The product will be a model that detects and categorizes bogus articles and can be utilized and connected with any system for future usage. The product model will test the hidden data, the results will be plotted, and as a result, the product will be a model that tests the unseen data.

# 2. Methodology

This section presents the methodology used for the classification. Using this model, a tool is implemented for detecting the fake news. In this method logistic regression classifier and decision tree classifier algorithms are used for model training, evaluation and prediction. We also compared the performance of these algorithms using metrics such as accuracy, precision, and recall. The algorithms used are explained below.

## 2.1 Algorithm

We used the following learning algorithms in conjunction with our proposed methodology to evaluate the performance of fake news detection classifiers.

### 2.1.1 Logistic Regression

As we are classifying text on the basis of a wide feature set, with a binary output (true/false or true news/fake news), a logistic regression (LR) model is used, since it provides the intuitive equation to classify problems into binary or multiple classes [27]. We performed hyper parameters tuning to get the best result for all individual datasets, while multiple parameters are tested before acquiring the maximum accuracies from LR model. Mathematically, the logistic regression hypothesis function can be defined as follows

$$h_{\theta}(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}.$$

Logistic regression uses a sigmoid function to transform the output to a probability value; the objective is to minimize the cost function to achieve an optimal probability. The cost function is calculated as shown in

$$\text{Cost}\left(h_\theta(x),\ y\right) = \begin{cases} \log\left(h_\theta(x)\right), & y = 1, \\ -\log\left(1 - h_\theta(x)\right), & y = 0. \end{cases}$$

### 2.1.2 Decision tree

The decision tree is an important tool that works based on flow chart like structure that is mainly used for classification problems. Each internal node of the decision tree specifies a condition or a “test” on an attribute and the branching is done on the basis of the test conditions and result. Finally the leaf node bears a class label that is obtained after computing all attributes. The distance from the root to leaf represents the classification rule. The amazing thing is that it can work with category and dependent variable. They are good in identifying the most important variables and they also depict the relation between the variables quite aptly. They are significant in creating new variables and features which is useful for data exploration and predicts the target variable quite efficiently.

## 2.2 Steps performed

- **Data preparation:**

Once we have collected the data, the next step is to prepare it for analysis. This involves cleaning and preprocessing the data, which includes tasks such as removing duplicates, handling missing values, and converting the data into a format that can be used by the machine learning algorithms.

- **Data exploration:**

After data preparation, the next step is to explore the data to gain insights into its characteristics. We can use various techniques such as visualization and statistical analysis to explore the data and identify any patterns or trends.

- **Data preparation and cleaning:**

Once we have explored the data, we may identify some data quality issues that need to be addressed. So the next step is to prepare it for analysis. This involves cleaning and preprocessing the data, which includes tasks such as removing duplicates, handling missing values and correcting errors in the data.

- **Feature engineering:**

Feature engineering involves selecting and transforming the variables (features) in the data that will be used by the machine learning algorithm. For fake news detection, common features include the length of the article, the number of unique words, and the presence of certain keywords or phrases.

- **Training model:**

After feature engineering, the next step is to train the machine learning model. We can use various algorithms such as logistic regression, decision trees, or random forests to train the model.(I have used both logistic regression and decision trees classifier as model)

- **Model building:**

Once the model is trained, the next step is to build the final model. This involves tuning the model hyper parameters and selecting the best performing model.

- **Model evaluation:**

After building the model, we need to evaluate its performance. This involves using metrics such as accuracy, precision, recall, and F1-score to assess the performance of the model.

- **Prediction:**

Once we have evaluated the model, we can use it to predict the class of new, unseen articles.

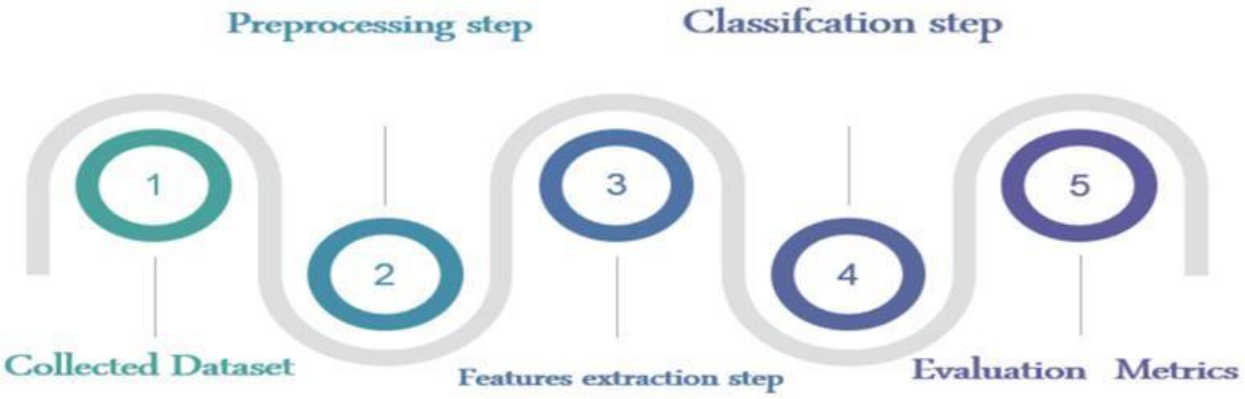

**Figure 1 Work design step of fake news detection**

# 3. Implementation

Using jupyter notebook we have implemented the system as the code shown below.

**Importing Libraries**

```
import re
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import metrics
from nltk.corpus import stopwords
from wordcloud import WordCloud
from sklearn.metrics import accuracy_score
from nltk.stem.porter import PorterStemmer
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
#from wordcloud import (WordCloud, random_color_func)
from wordcloud import (WordCloud, get_single_color_func)
from sklearn.feature_extraction.text import TfidfVectorizer
```

**Downloading stopwords**

```
import nltk
nltk.download('stopwords')
```

**Printing stop words**

```
print(stopwords.words('english'))
```

**Reading csv file**

```
news_dataset = pd.read_csv('news.csv')
print(news_dataset)
```

**Displaying key words**

```
news_dataset.keys()
```

**Displaying the dataset shape**

```
news_dataset.shape
```

**Displying the first five rows dataset**

```
news_dataset.head()
```

**Listing of the null count**

```
news_dataset.isnull().sum()
```

**Replacing the missing values with function**

```
news_dataset = news_dataset.fillna(' ')
```

**Merging 'title' and 'author' column in a new column 'body'**

```
news_dataset['body']=news_dataset['author']+' '+news_dataset['title']
```

**Creating a new Porter stemmer.**

```
port_stem =PorterStemmer()
```

### Stemming words

```
def stemming(body):
  stemmed_body =re.sub('[^a-zA-Z]',' ',body)
  stemmed_body =stemmed_body.lower()
  stemmed_body =stemmed_body.split()
  stemmed_body =[port_stem.stem(word) for word in stemmed_body if not word in
stopwords.words('english')]
  stemmed_body =' '.join(stemmed_body)
  return stemmed_body
news_dataset['body']=news_dataset['body'].apply(stemming)
```

### Printing the new dataset

```
print(news_dataset['body'])
```

### Assigning x and y values

```
X =news_dataset['body'].values
Y =news_dataset['label'].values
```

### Printing x and y

```
print(X)
print(Y)
```

### Visualizing real news

```
consolidated = ' '.join(
    word for word in news_dataset['body'][news_dataset['label'] == 0].astype(str))
wordCloud = WordCloud(width=1600,
                      height=800,
                      random_state=21,
                     max_font_size=110,
                     collocations=False)
plt.figure(figsize=(15, 10))
plt.imshow(wordCloud.generate(consolidated), interpolation='bilinear')
plt.axis('off')
plt.show()
```

**Visualizing fake news**

```
consolidated = ' '.join(
    word for word in news_dataset['body'][news_dataset['label'] == 1].astype(str))
wordCloud = WordCloud(width=1600,
                     height=800,
                     random_state=21,
                     max_font_size=110,
                     collocations=False)
plt.figure(figsize=(15, 10))
plt.imshow(wordCloud.generate(consolidated), interpolation='bilinear')
plt.axis('off')
plt.show()
```

**Converting the textual data to numerical data**

```
vectorizer =TfidfVectorizer()
vectorizer.fit(X)
X =vectorizer.transform(X)
```

**Printing X after numerical vectorization**

```
print(X)
```

**Splitting the dataset to training and test data**

```
X_train,X_test,Y_train,Y_test =train_test_split(X,Y,test_size=0.2,stratify=Y,random_state=2)
```

**Model Training**

```
model = LogisticRegression()
```

**Training the model**

```
model.fit(X_train,Y_train)
```

**Predict training data**

```
Y_train_pred = model.predict(X_train)
training_accuracy = accuracy_score(Y_train_pred,Y_train)
```

**printing accuracy score of training data**

```
print('Accuracy score of training data :',training_accuracy)
```

**Predict testing data**

```
Y_test_pred = model.predict(X_test)
test_accuracy = accuracy_score(Y_test_pred,Y_test)
```

**Printing accuracy score of test data**

```
print('Accuracy score of test data :',test_accuracy)
```

**Plot Confusion matrix of Results from LogisticRegression**

```
cm = metrics.confusion_matrix(Y_test, model.predict(X_test))
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix=cm,
                                            display_labels=[False, True])
cm_display.plot()
plt.show()
```

**Model Training : Decision Tree Classifier**

```
model = DecisionTreeClassifier()
```

**Training the model**

```
model.fit(X_train,Y_train)
```

**Predicting training data**

```
Y_train_pred=model.predict(X_train)
```

**Accuracy score of training data**

```
training_accuracy = accuracy_score(Y_train_pred,Y_train)
```

**Printing accuracy score of training data**

```
print('Accuracy score of training data :',training_accuracy)
```

**Predict testing data**

```
Y_test_pred = model.predict(X_test)
```

### Accuracy score of test data

```
test_accuracy = accuracy_score(Y_test_pred,Y_test)
```

### Printing accuracy score of test data

```
print('Accuracy score of test data :',test_accuracy)
```

### Confusion matrix of Results from Decision Tree classification

```
cm = metrics.confusion_matrix(Y_test, model.predict(X_test))
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix=cm,
                                            display_labels=[False, True])
cm_display.plot()
plt.show()
```

### Making a predictive system

```
X_new =X_test[0]
prediction = model.predict(X_new)
print(prediction)
if (prediction[0]==0):
  print("It's a real news")
else :
  print("It's a fake news")
```

### Cross verifying above prediction

```
print(Y_test[0])
if (Y_test[0]==0):
  print("It's a real news")
else :
  print("It's a fake news")
```

# 4. Discussion of experimental results

Graphical representation of the dataset is shown below.

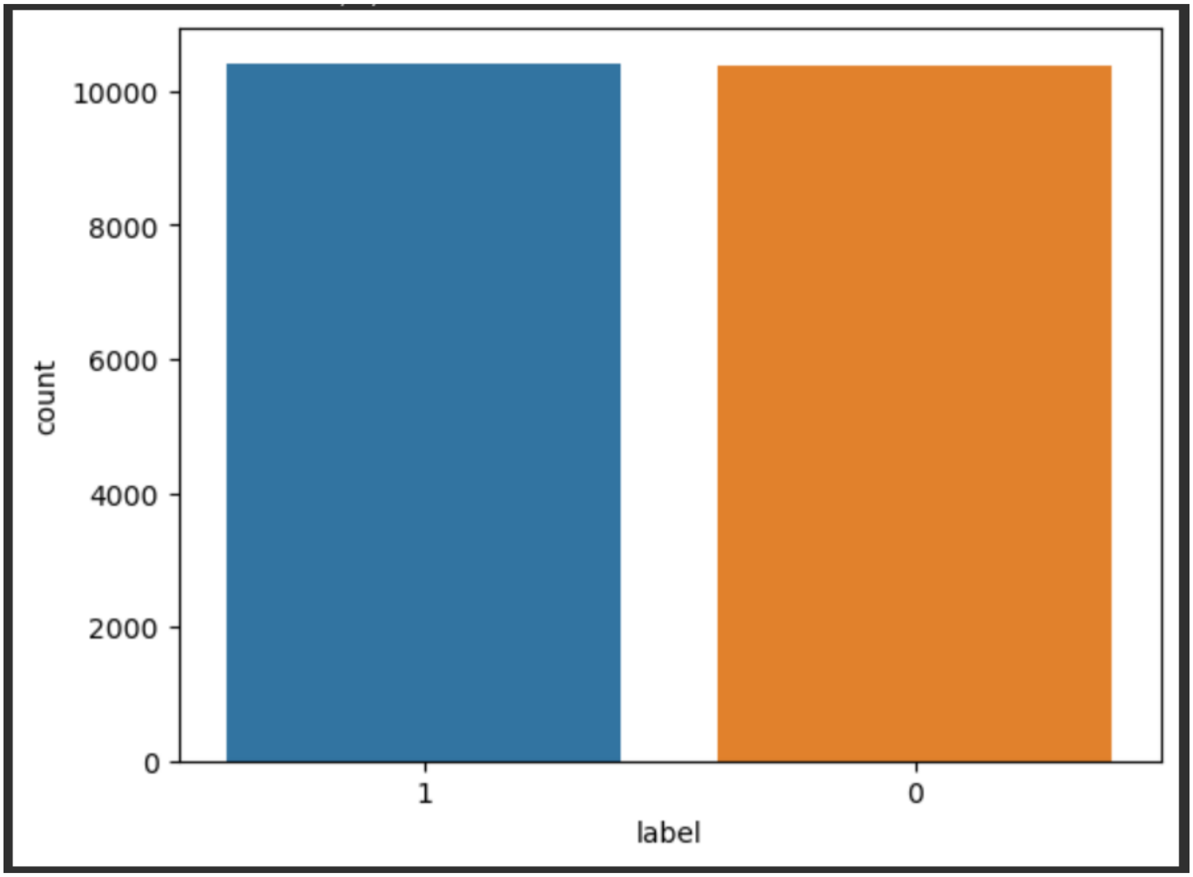

**Figure 2 Graphical representation of the dataset**

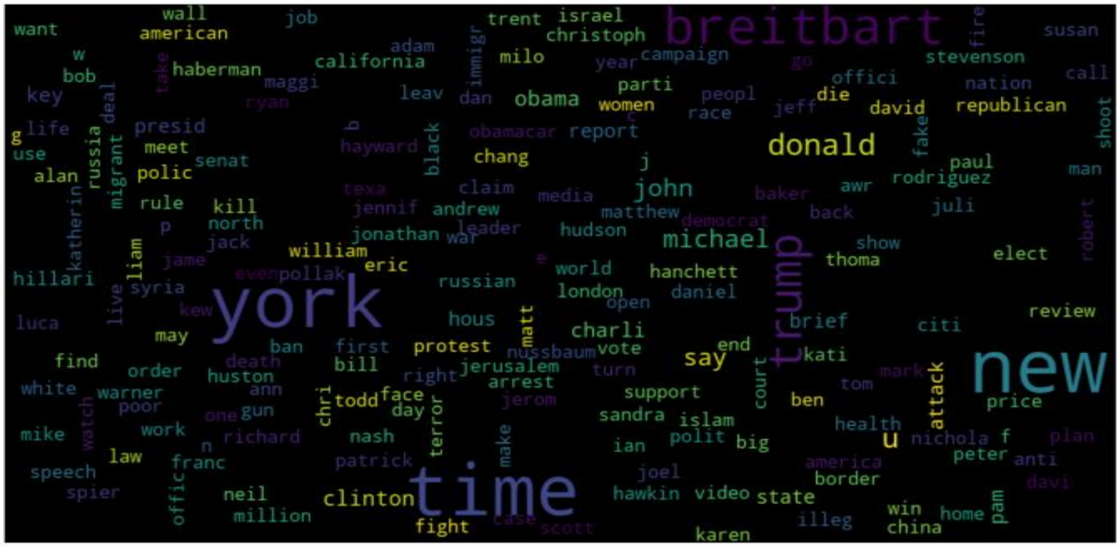

**Figure 3 Real news visualization**

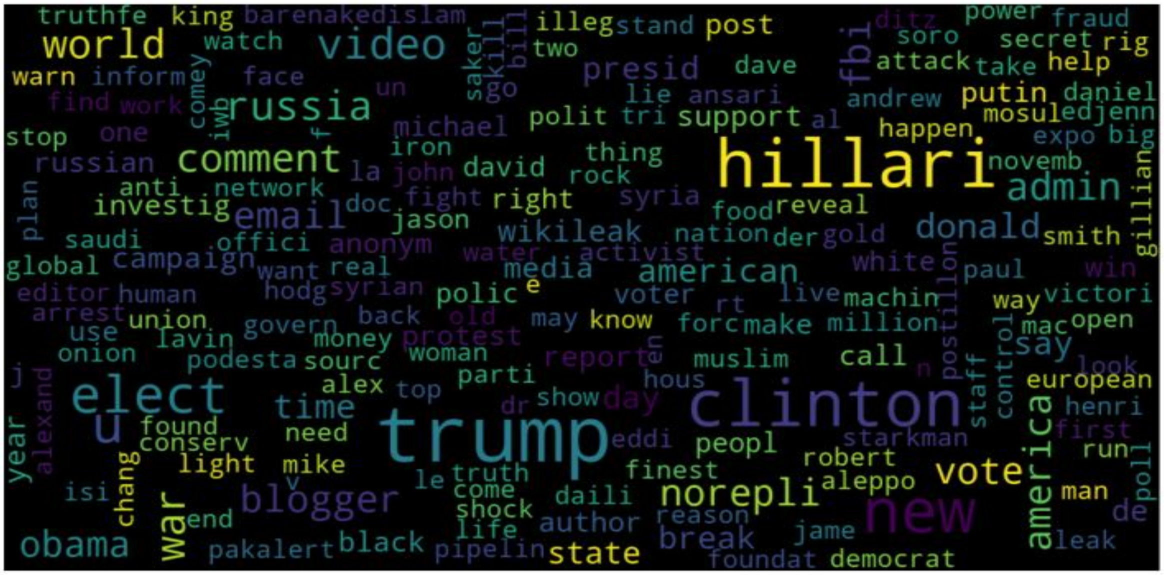

**Figure 4 Fake news visualization**

### The result of Logistic Regression

Among 2077 false news, 2004 were predicted as false (correctly classified) and 73 news predicted as true (incorrectly classified). Parallely among 2083 true news 2069 news were predicted as true (correctly classified) and 14 news predicted as false (incorrectly classified). Accuracy score of training data was 98.6% and accuracy score of test data was 97.9%.

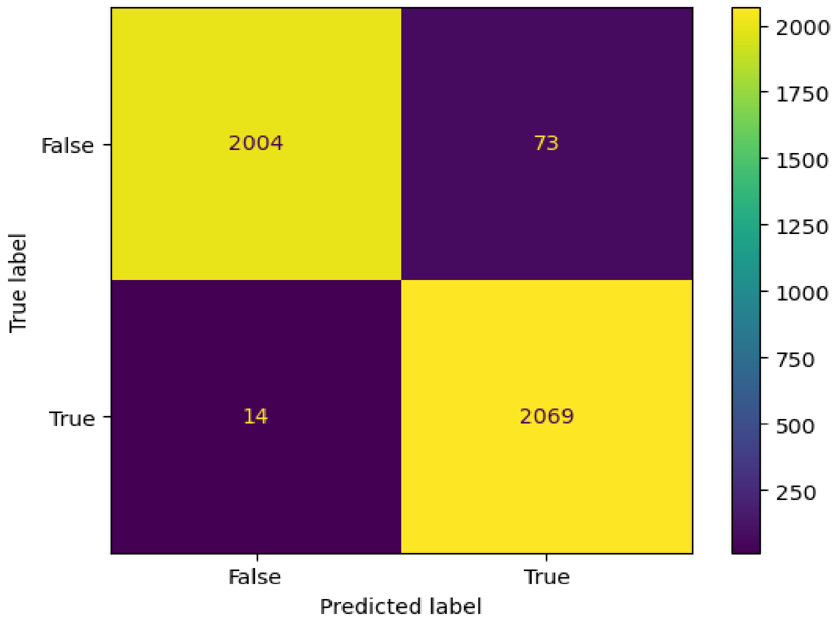

**Figure 5 Confusion matrix of Results from Logistic Regression**

**The result of** Decision Tree

Among 2077 false news, 2064 were predicted as false (correctly classified) and 13 news predicted as true (incorrectly classified). Parallely among 2083 true news 2063 news were predicted as true (correctly classified) and 20 news predicted as false (incorrectly classified). Accuracy score of training data was 1.0% and accuracy score of test data was 99%.

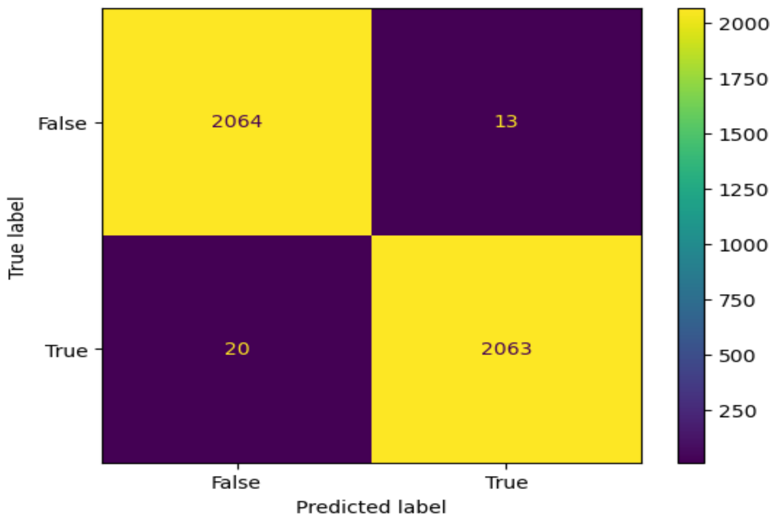

**Figure 6 Confusion matrix of Results from Decision Tree classification**

## 5. Conclusion

From the results above, we conclude that the decision tree is better than logistic regression in the classification accuracy of fake news dataset. Fake news detection has many open issues that require attention of researchers. For instance, in order to reduce the spread of fake news, identifying key elements involved in the spread of news is an important step. Machine learning techniques can be employed to identify the key sources involved in spread of fake news. Likewise, real time fake news identification in videos can be another possible future direction.

# 6. Reference

- Arvinder P.B., Sourabh C., and Mahima G. "Comparative Performance of Machine Learning Algorithms for Fake News Detection", (2019).
- J. Soll, “)e long and brutal history of fake news,” Politico Magazine, vol. 18, no. 12, 2016.
- L. Breiman, J. Friedman, R. Olshen, and C. Stone, Classification and Regression Trees, Springer, Berlin, Germany, 1984.
- N. K. Conroy, V. L. Rubin, and Y. Chen, “Automatic deception detection: methods for finding fake news,”
- Sam F., "Decision Tree Classification with Differential Privacy: A Survey", 2019.