FAKE NEWS DETECTION

Submitted by

M S AJAYNATH (Reg No.:VPAWSCS032)

ABHIJITH J (Reg No:VPAWSCS030)

SANIA VARMA (Reg No:VPAWSCS004)

VISAL KRISHNA K (Reg No:VPAWSCS012)

For the award of the Degree of

Bachelor of Science (B. Sc.) in

Computer Science

(**University of Calicut**)

under the guidance of

Dr.Shweta AND Dr. Jeevamol Joy

Assistant Professor

Department of Computer Science

Government Victoria College

Palakkad, Kerala

March 2022-2025

GOVERNMENT VICTORIA COLLEGE PALAKKAD-678001

CERTIFICATE

This is to certify that the project work entitled "FAKE NEWS DETECTION" is a bonafide record of original work done by Mr. **M S AJAYNATH** (Reg No:VPAWSCS032), Mr. **ABHIJITH J** (Reg No:VPAWSCS030), Ms. **SANIA VARMA** (Reg No:VPAWSCS004),Mr. **VISAL KRISHNA K** (Reg No:VPAWSCS012) of final year BSc. Computer science in partial requirements for the award of degree in Bachelor of Science in Computer science during the period of 2022-2025.

UNIVERSITY OF CALICUT

We also certify that the work done is original.

PROJECT GUIDE **HEAD OF THE DEPARTMENT**

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project entitled "**FAKE NEWS DETECTION**" submitted for fulfilment of the requirements for the award of the degree in **BACHELOR OF SCIENCE IN COMPUTER SCIENCE to UNIVERSITY OF CALICUT** is our original work done under the guidance of Dr. SHWETHA & Dr. JEEVAMOL JOY during our study period in **GOVERNMENT VICTORIA COLLEGE, PALAKKAD.**

Mr. M S AJAYNATH
VPAWSCS032

Mr ABHIJITH J
VPAWSCS030

Mr. VISAL KRISHNA K
VPAWSCS012

Ms. SANIA VARMA
VPAWSCS004

ACKNOWLEDGEMENT

The success of any venture depends on the blessings and prayers of all wellwishers. We wish to take this opportunity to thank everyone who had worked directly and indirectly for the successful completion of this project. Firstly we thank the God Almighty for showering his abundant blessings on us.

We express our sincere gratitude to our project guide and respected teacher Dr.Shweta KR & Dr. Jeevamol joy for her moral support that made this project a great success. A thanks is also extended to all the teaching faculties of the Department of Computer Science for their effective advises and selfless criticism. Last, but not least we thank our family and friends who paid an eminent role in the successful completion of this project.

ABSTRACT

As the name suggests, "FAKE NEWS DETECTOR" is a machine learning-based application designed to identify and classify fake news efficiently.

The Fake News Detector is a web based application meant for individuals, journalists, and organizations concerned about the authenticity of online news. With the help of this application, users can analyze news articles and determine their credibility using advanced machine learning algorithms. This solution provides a more reliable and automated alternative to manually verifying news sources.

Our key features are:

- Simple user interface.
- Detection of fake news using machine learning models.
- Utilizes Logistic Regression, Gradient Boosting, Decision Tree, and Random Forest algorithms
- Easy to use and interpret results
- Provides credibility scores for news articles

This system helps users distinguish between reliable and misleading information, promoting awareness and trust in news consumption.

OBJECTIVE

Our project aims to build a Fake News Detection system that helps users verify the credibility of news articles efficiently. The system utilizes machine learning algorithms such as Logistic Regression, Gradient Boosting, Decision Tree, and Random Forest to analyze news content and classify it as real or fake.

The main objective of this project is to combat the spread of misinformation, provide users with a reliable tool for news verification, and promote awareness about fake news. By offering accurate and fast detection, this system enhances media literacy and helps users make informed decisions about the information they consume.

1.INTRODUCTION

In today's digital era, the rapid spread of misinformation and fake news has become a critical issue, especially with the increasing reliance on social media and online news platforms. Fake news can manipulate public opinion, cause panic, and lead to misinformation-related consequences. Factors such as political propaganda, misleading advertisements, and unverified reports contribute to the growing challenge of distinguishing between real and fake news.

To address this issue, we propose a machine learning-based Fake News Detection system that efficiently analyzes and verifies the authenticity of news articles. This system uses advanced classification algorithms, including Logistic Regression, Gradient Boosting, Decision Tree, and Random Forest, to assess news credibility. The application provides users with a simple interface to input news articles or URLs, generating a credibility score based on the trained model.

While several fake news detection tools exist, many are either complex to use or lack accuracy. Our project focuses on user-friendliness, efficiency, and reliability. It helps users make informed decisions by providing clear and accurate results about news authenticity. By promoting awareness and reducing the spread of misinformation, this application aims to contribute to a more informed and responsible digital society.

2.SYSTEM STUDY

System analysis is the primary phase of the software development. It refers to the structured process for identifying and solving problems. System analysis is the process that takes place when new system is being built. It is the central interact of the system development and it includes gathering necessary data and developing a plan to the new system. System analysis not only includes the process of analysis but also that of synthesis, which is of putting parts together to creative and imaginative in producing new solutions to meet the user.

2.1 EXISTING SYSTEM

- Fake news detection is often done manually, which is time-consuming and inefficient.
- Lack of an automated system results in misinformation spreading quickly.
- Traditional methods rely on fact-checking websites, which may not cover all news sources

2.2 PROPOSED SYSTEM

- Uses machine learning models (Logistic Regression, Gradient Boosting, Decision Tree, and Random Forest) to classify news as real or fake
- Provides a user-friendly web interface using Flask for easy interaction
- Allows users to input news text and receive immediate classification results.
- Improves accuracy compared to manual fact-checking methods
- Can be expanded to support multiple languages and different sources.

3.SYSTEM SPECIFICATION

The system specification is the result of documenting the development phase activities It is completed at the conclusion of development phase. The system specification is divided into two parts; the first part is hardware specification, related to the interaction of information system with its environment. The second part is software specification which completely documents program component of the system.

3.1 HARDWARE REQUIREMENTS

Processor : Intel(R)Core(TM)

RAM : 8 GB

Hard Disk : 90GB

Speed : 1.8GHZ or higher

3.2 SOFTWARE REQUIREMENTS

Operating System : Windows 11

Coding language : Python

Framework/Libraries:Flask, Scikit-learn, pandas, Numpy,re, matplotlib, seaborn

Dataset : Kaggle (true.csv,fake.csv)

Documentation : Microsoft word

3.3 CODING LANGUAGE : PYTHON

Python is a high-level, interpreted programming language widely used for machine learning, artificial intelligence, and web development. Its simplicity and vast ecosystem of libraries make it an ideal choice for implementing the Fake News Detection System.

3.4 WHY PYTHON?

- Easy to learn: Simple syntax and readability
- Extensive Libraries: Supports ML and NLP libraries like Scikit-learn, re,

Pandas, NumPy

- Strong Community Support: Regular updates and open-source contributions

Cross-platform: Works on Windows, Linux, and MacOS

3.5 PYTHON IN FAKE NEWS DETECTION

1. **Data Handling & Processing** Libraries

   Used: pandas, numpy Purpose:

- Reads datasets (Fake.csv and True.csv) using pandas
- Assigns labels (0 for fake news, 1 for real news)
- Combines both datasets for further processing

2. **Text Cleaning Using re (Regular Expressions)**

   Library Used: re (Regular Expressions)

   Purpose: Cleans the text by removing unwanted characters like punctuation, numbers, and special symbols

   .

   Why re is Used?

   - Regular expressions are efficient for pattern matching.
   - Faster than nltk for simple text cleaning tasks.
   - Handles multiple transformations in a single step.

3. **Data Visualization**

   Libraries Used: matplotlib, seaborn.

   Purpose: Creates graphs and plots to understand the dataset, like class distribution.

4. **Splitting Data for Training and Testing**

   Library Used: sklearn.model_selection

Purpose: Divides data into training (80%) and testing (20%) sets.

5. **Machine Learning Models Libraries Used: sklearn Models Used:**
    - Logistic Regression
    - Decision Tree
    - Random Forest
    - Gradient Boosting

6. **Evaluating Performance**

    Library Used: sklearn.metrics

    Purpose: Measures accuracy, precision, recall, and F1-score.

4. DATASET IN FAKE NEWS DETECTION PROJECT

The dataset used in your fake news detection project consists of two main files: Fake.csv and True.csv, which contain labeled news articles. These datasets help train machine learning models to differentiate between real and fake news based on text content

The dataset is text-based, meaning it focuses on the content of news articles rather than images, videos, or metadata. It is widely used for natural language processing (NLP) tasks related to fake news classification

Fake.csv → Contains fake news articles, which are intentionally misleading or fabricated.

True.csv → Contains real news articles, sourced from legitimate and credible news outlets.

Both datasets are later merged into a single dataset for training and testing.

4.1 Why is this dataset important?

Fake news has become a major issue in the digital world, spreading misinformation rapidly. This dataset provides a structured way to analyze and detect fake news using machine learning algorithms.The dataset contains a large number of fake and real news articles, making it useful for training models effectively.

It is balanced, meaning there are nearly equal numbers of fake and real news articles. If the dataset were imbalanced, the model might favor the majority class, leading to inaccurate predictions.

4.2 Preprocessing The Dataset

Before training machine learning models, the dataset needs to be cleaned and preprocessed to improve accuracy. The preprocessing steps include:

- Lowercasing: Converting all text to lowercase to ensure uniformity

- Removing Numbers & Special Characters: Fake news often uses unnecessary punctuation or numbers to appear credible
- Tokenization: Splitting text into individual words for better analysis.
- Removing Stopwords: Common words like "the," "is," and "and" are removed since they do not provide useful information. • Stemming/Lemmatization: Reducing words to their root form (e.g., "running" → "run") to standardize text.

These steps help remove unnecessary noise from the text and make it easier for machine learning models to understand patterns.

4.3 Training & Testing The Dataset

The dataset is divided into training and testing sets to evaluate the model's performance: Training Set (80%): Used to teach the machine learning model how to differentiate between real and fake news

Testing Set (20%): Used to check how well the model performs on unseen data. A wellbalanced training and testing split ensures that the model does not memorize patterns but instead learns to generalize

The dataset plays a crucial role in fake news detection by providing labeled examples of real and fake news articles. Through text analysis, machine learning models can identify misleading information and help combat misinformation.

However, challenges like bias, misclassification, and evolving fake news tactics need to be addressed to maintain accuracy

```
data.head()
```

	text	class
0	Ted Cruz is fast approaching Donald Trump as t...	0
1	FLINT, Mich. (Reuters) - Democratic presidenti...	1
2	The news that the forecast for the third quart...	0
3	PARIS (Reuters) - French state-controlled util...	1
4	MEXICO CITY (Reuters) - Mexico's new finance m...	1

In this image, class 0 indicates fake news, while class 1 denotes true new

5. SYSTEM DESIGN

Design concepts provide the basic criteria for ensuring the quality of a system. In the context of Fake News Detection, design plays a crucial role in structuring how data is processed, how the model is trained, and how the system interacts with users. The design focuses on three key areas: data processing, model architecture, and user interface.

Once the system requirements are analyzed and specified, software design involves three main activities: data pre-processing, model training, and result evaluation. Each phase transforms raw data into useful insights that validate the efficiency of the Fake News Detection system

Design is the first step in moving from identifying the problem (false information spread) to providing a solution (a system that detects fake news). It acts as a bridge between the identified requirement and the final implementation of a machine learning-based system.

The detailed system design includes :

1. Module Description
2. Data Pre-processing Module
3. Feature Extraction
4. Machine Learning Module
5. Prediction Module
6. Evaluation Module
7. User Interface Module

5.1 MODULE DESCRIPTION

1.MODULE DESCRIPTION:

The Fake News Detection system consists of several essential modules, each responsible for a specific function in the pipeline. The system is designed to efficiently analyze news articles, classify them as real or fake, and provide accurate predictions based on machine learning models

2.DATA PREPROCESSING MODULE:

This module is responsible for cleaning and preparing the text data before feeding it into the machine learning models. Since raw text data often contains noise, inconsistencies, and irrelevant symbols, it is necessary to preprocess the data for better model accuracy

3.FEATURE EXTRACTION MODULE:

Once the text is cleaned, it needs to be converted into a numerical format that machine learning models can understand. This module extracts key features from the processed text using TFIDF (Term Frequency-Inverse Document Frequency)

Key Functions:

TF-IDF Transformation:

Converts text into numerical vectors based on the frequency of important words. Helps in identifying words that are crucial for distinguishing between fake and real news

Vectorization:

Converts text data into a structured format suitable for machine learning

Dimensionality Reduction (if applicable):

Reduces the number of features while retaining important information to improve model efficiency.

4.MACHINE LEARNING MODULES:

This module handles the training and implementation of machine learning models used for classification. Various models are tested to determine which one provides the best accuracy.

Implements multiple classification algorithms, including:

Logistic Regression: A statistical model that predicts fake or real news based on probability.

Decision Tree: A tree-based model that splits data into different categories.

Random Forest: An ensemble of multiple decision trees to improve accuracy. **Gradient**

Boosting: A boosting algorithm that enhances prediction performance

5. PREDICTION MODULES:

The Prediction Module plays a crucial role in real-time classification of news articles. Once the machine learning models are trained, this module allows users to input a news article and receive an immediate classification as either "Fake News" or "Real News." The system first processes the user input using the same preprocessing techniques applied during training, ensuring consistency in text transformation. The processed text is then passed through the trained model, which generates a prediction. This module is essential for real-world application, as it helps users quickly identify potentially misleading information.

6. EVALUATION MODULES:

After training the models, it is necessary to measure their performance using various evaluation metrics

Key Functions:

- Accuracy Calculation:
  - Determines how often the model correctly classifies news articles.
- Precision, Recall, and F1-Score:
  - Precision: Measures how many predicted fake news articles are truly fake.
  - Recall: Measures how many actual fake news articles are correctly identified.
  - F1-Score: Balances precision and recall to provide an overall measure of model performance.
- Confusion Matrix: Displays the number of true positives, false positives, true negatives, and false negatives This module ensures that the selected model is reliable and effective in detecting fake news

7.USER INTERFACE (UI) MODULE:

To make the system user-friendly, a web-based interface is developed using Flask. This allows users to enter news articles and receive predictions.

Frontend Design:

- Simple UI where users can input news articles.

- A button to submit the news for analysis.

Backend Connection:

- The frontend is connected to the machine learning model via Flask.
- User input is processed, and the prediction is displayed.

Real-Time Output:

- Users receive immediate feedback on whether the news is real or fake.
- This module provides an easy-to-use interface, allowing users to interact with the fake news detection system

6.SYSTEM DEVELOPMENT

CODING

The coding phase is crucial as it translates the system design into an executable program using a programming language. In this project, coding is implemented using Python along with the Flask framework for the web-based interface.

The system is developed using structured coding practices, ensuring readability, maintainability, and efficiency. The core logic of fake news detection is based on Natural Language Processing (NLP) techniques and machine learning models that classify news articles as real or fake.

To improve the quality of the source code, best coding standards are followed, such as:

- Modular programming to keep functions reusable and maintainable.
- Error handling and validation mechanisms to prevent incorrect data inputs.
- Model training and optimization to improve classification accuracy.
- Machine Learning Models Used in Fake News Detection

In our project, we implemented multiple machine learning models to classify news as real or fake. Each model has its strengths, and we compared their performance to determine the most effective one. Below is a general description of the models used:

General Description of Models

Machine learning models for fake news detection work by analyzing textual data and identifying patterns that differentiate real news from fake news. These models rely on natural language processing (NLP) techniques to extract features from the text, such as word frequency, sentiment, and readability. The models used in this project include:

1. Logistic Regression

2. Gradient Boosting

3. Decision Tree

4. Random Forest

TF-IDF (Term Frequency-Inverse Document Frequency) in Fake News Detection

TF-IDF is a statistical method used to evaluate the importance of a word in a document relative to a collection (corpus) of documents. It helps convert text into numerical form, making it suitable for machine learning models.

Understanding TF-IDF

TF-IDF consists of two main components:

- Term Frequency (TF): Measures how frequently a term appears in a document.
- Inverse Document Frequency (IDF): Measures how important a term

TF = (Number of times the term appears in a document) / (Total number of terms in the document)

IDF = log(Total number of documents / Number of documents containing the term)

TF-IDF = TF * IDF

Steps to Compute TF-IDF

Step 1: Tokenization

- Convert the document into words (tokens) by removing punctuation and splitting text.

Example:

- Original text: "Fake news spreads misinformation."
- Tokens: ["Fake", "news", "spreads", "misinformation"]

Step 2: Compute Term Frequency (TF)

Count the number of times each word appears in a document and normalize by the total number of words.

Example:

WORD	COUNT	TF
Fake	1	1/4
News	1	1/4
Spreads	1	1/4
Misinformation	1	1/4

Step 3: Compute Inverse Document Frequency (IDF)

- Count how many documents contain each word, then apply the IDF formula.

Example (Assume 1000 documents in the dataset, and "news" appears in 500 of them):

- IDF("news") = log(1000 / 500) = log(2) ≈ 0.693

Step 4: Compute TF-IDF Score

- Multiply TF and IDF for each word.

  Example for "news":

  - TF-IDF = (1/4) × 0.693 ≈ 0.173

6.1 LOGISTIC REGRESSION

- Logistic Regression is a simple yet effective classification algorithm that predicts the probability of an input belonging to a particular class (real or fake news).
- It works by applying a logistic function (sigmoid) to a linear combination of input features.

- In fake news detection, Logistic Regression analyzes word frequencies and patterns to determine the likelihood of an article being fake.

Advantages: Fast, interpretable, and works well with large datasets.

Limitations: May struggle with highly complex or nonlinear relationships.

$$P(Y = 1 \mid X) = 1 / (1 + e^{-(\beta 0 + \beta 1 X1 + \beta 2 X2 + ... + \beta n Xn)})$$

Where:

- $P(Y = 1 \mid X)$ is the probability of the positive class (news being fake).
- $\beta 0$ is the intercept. • $\beta 1, \beta 2, ..., \beta n$ are the coefficients.
- X1, X2, ..., Xn are the feature values.
- e is Euler's number (~2.718).

6.2 GRADIENT BOOSTING

- Gradient Boosting is an ensemble learning technique that builds multiple weak classifiers (decision trees) and combines them to improve accuracy.
- Each tree corrects the errors of the previous ones, gradually improving performance.
- In fake news detection, Gradient Boosting is useful for handling nonlinearity and capturing complex relationships in the text data.

Advantages: High accuracy, robust to overfitting.

Limitations: Computationally expensive and requires careful tuning of hyperparameters.

$$F_m(X) = F_{(m-1)}(X) + \gamma * h_m(X)$$

Where:

- $F_m(X)$ is the updated model.
- $F_{(m-1)}(X)$ is the previous model.
- γ is the learning rate.
- $h_m(X)$ is the new weak learner added at step m.

6.3 DECISION TREE

- Decision Trees work by splitting the dataset into branches based on feature values, forming a tree-like structure.
- Each node in the tree represents a decision based on a specific feature, leading to a final classification.
- In our project, Decision Trees help classify fake news based on word patterns and textual features.

Advantages: Easy to interpret and visualize.

Limitations: Prone to overfitting and may not generalize well.

Decision Trees use splitting criteria like Entropy or Gini Impurity.

Entropy Equation:

$$H(S) = - \Sigma (pi * \log2(pi))$$

Where:

- H(S) is the entropy of set S.
- pi is the proportion of class i in S.

Gini Impurity Equation:

$$Gini(S) = 1 - \Sigma (pi^2)$$

Where:

- pi is the probability of class i in the dataset.

6.4 RANDOM FOREST

- Random Forest is an extension of Decision Trees that builds multiple trees and combines their predictions to improve accuracy and reduce overfitting.
- Each tree is trained on a random subset of the data, making the model more robust.
- In fake news detection, Random Forest captures complex patterns in text data while minimizing errors.

Advantages: Handles large datasets well and reduces overfitting.

Limitations: Can be slower and more complex than individual Decision Trees.

Prediction Formula:

$$\hat{y} = (1/T) * \Sigma h_t(X)$$

Where:

- $\hat{y}$ is the final prediction.
- T is the number of trees in the forest.
- $h_t(X)$ is the prediction from the t-th tree.

7.PYTHON LIBRARIES

1. **NumPy: For numerical operations and array manipulation.**
2. **Pandas: For data analysis and manipulation, including data frames.**
3. **Seaborn: For data visualization and statistical graphics.**
4. **Re :provides support for working with regular expressions.**
5. **String : A collection of constants and utility functions for working with and manipulating strings, such as ASCII letters, digits, and punctuation.**
6. **Matplotlib: For creating plots and charts.**
7. **scikit-learn (sklearn): For machine learning algorithms and model evaluation metrics**

8.MODEL ANALYSIS

Performance Metrics

Use different metrics to analyze how well each model performs:

- Accuracy – Percentage of correctly classified news articles.
- Precision – Ability of the model to avoid false positives (misclassifying real news as fake).
- Recall (Sensitivity) – Ability of the model to detect fake news correctly.
- F1-score – Balance between precision and recall.
- ROC-AUC Score – Measures how well the model distinguishes between fake and real news.

```
...                  precision    recall  f1-score   support

               0       0.99      0.99      0.99      5805
               1       0.99      0.99      0.99      5415

        accuracy                           0.99     11220
       macro avg       0.99      0.99      0.99     11220
    weighted avg       0.99      0.99      0.99     11220
```

```
with open("logistic_model.pkl", "wb") as model_file:
    pickle.dump(LR, model_file)
```

```
print(classification_report(y_test,pred_dt))
```

[69] Python

```
                precision    recall  f1-score   support

           0       1.00      1.00      1.00      5805
           1       1.00      1.00      1.00      5415

    accuracy                           1.00     11220
   macro avg       1.00      1.00      1.00     11220
weighted avg       1.00      1.00      1.00     11220
```

```
with open("Decision_tree_model.pkl", "wb") as model_file:
    pickle.dump(DT, model_file)
```

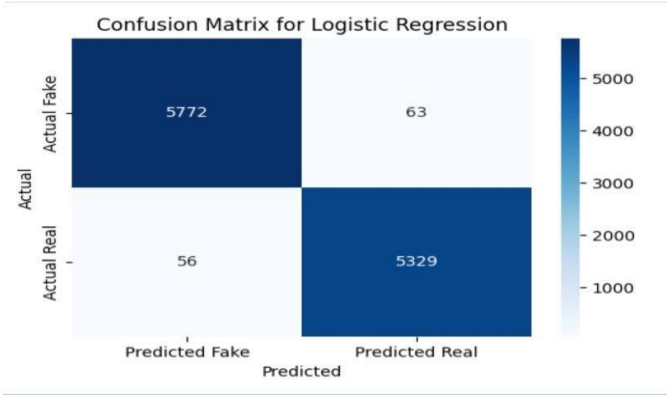
```
              precision    recall  f1-score   support

           0       1.00      0.99      1.00      5805
           1       0.99      1.00      0.99      5415

    accuracy                           0.99     11220
   macro avg       0.99      1.00      0.99     11220
weighted avg       0.99      0.99      0.99     11220
```

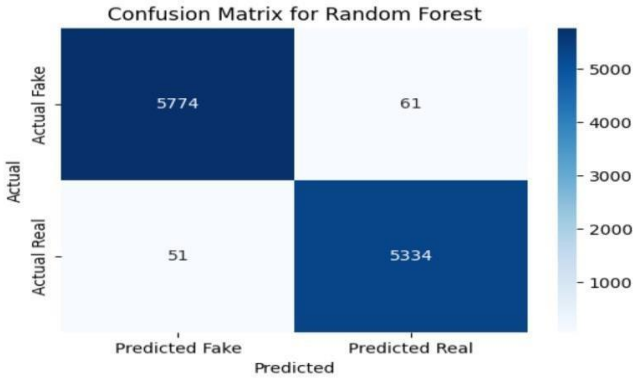
```
with open("GradientBooster_model.pkl", "wb") as model_fil
    pickle.dump(GB, model_file)
```

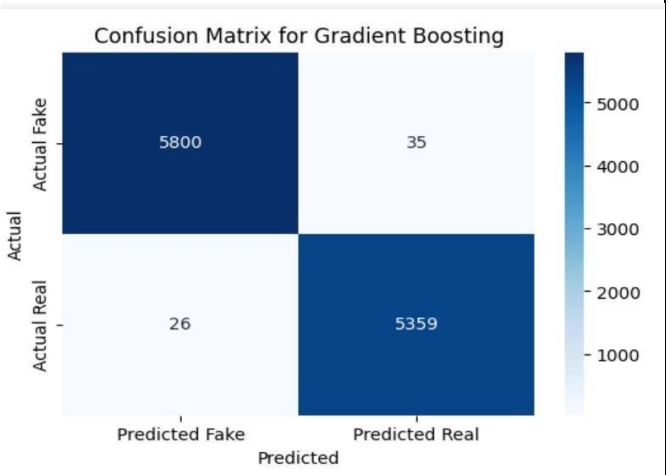
```
              precision    recall  f1-score   support

           0       0.99      0.98      0.99      5805
           1       0.98      0.99      0.98      5415

    accuracy                           0.99     11220
   macro avg       0.98      0.99      0.99     11220
weighted avg       0.99      0.99      0.99     11220
```

```
with open("RandomForest_model.pkl", "wb") as model_file:
    pickle.dump(RF, model_file)
```

9.CONFUSION MATRIX







These metrics provide insight into how well each model performed in classifying fake and real news.

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	98.87%	98.54%	99.10%	98.82%	98.88%
Decision Tree	99.63%	99.55%	99.68%	99.61%	99.63%
Random Forest	98.88%	98.74%	98.91%	98.83%	98.88%
Gradient Boosting	99.41%	99.08%	99.68%	99.38%	99.42%

10.CONCLUSION

By the end of this project, the Fake News Detection System successfully achieves its objective of identifying and categorizing news as real or fake. The system is designed with a user-friendly interface to allow users to verify the authenticity of news efficiently. Using machine learning models such as Logistic Regression, Gradient Boosting, Decision Tree, and Random Forest, the system analyzes textual data and classifies news articles with high accuracy.

The implementation of this project helps combat the spread of misinformation, allowing users to make informed decisions based on reliable news sources. As fake news continues to be a major issue, this system provides an essential tool for individuals, journalists, and organizations to validate online content.

11.FUTURE ENHANCEMENTS

To improve the effectiveness and reach of the Fake News Detection System, several enhancements can be introduced:

Integration with Social Media Platforms – The system can be expanded to analyze news shared on platforms like Facebook, Twitter, and WhatsApp to prevent the spread of misinformation in real time.

Multilingual Support – Adding support for multiple languages to detect fake news across different regions and linguistic communities.

Deep Learning Implementation – Enhancing accuracy by incorporating transformer-based models like BERT or GPT for better text analysis.

Browser Extension Development – Creating a Chrome or Firefox extension to allow users to verify news articles directly while browsing.

Fact-Checking API – Integrating with third-party fact-checking databases (e.g., PolitiFact, Snopes) to cross-verify news credibility.

12.BIBILOGRAPHY

Web Sources

1. Scikit-learn Documentation: https://scikit-learn.org/
2. TensorFlow: https://www.tensorflow.org/
3. Kaggle Fake News Dataset: https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset
4. Youtube **:** https://youtu.be/U6ieiJAhXQ4?si=D8_8kkfLmp_81YRi

Software & Tools

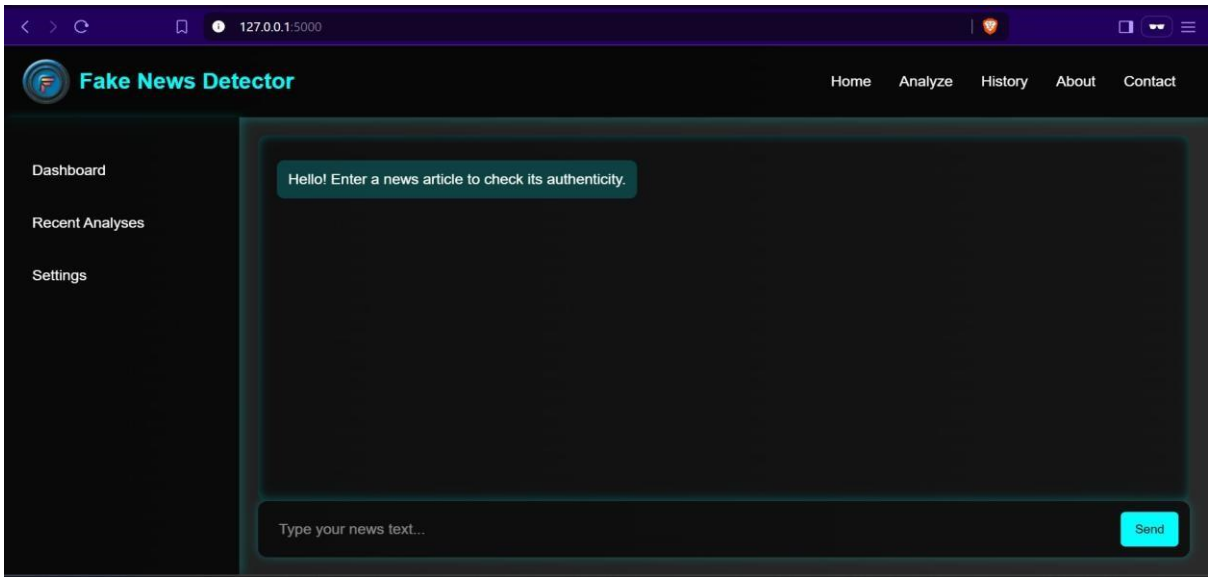
- Python (Version 3.x)
- Flask (for web application)
- Scikit-learn (for machine learning models)
- Pandas & NumPy (for data processing**)**

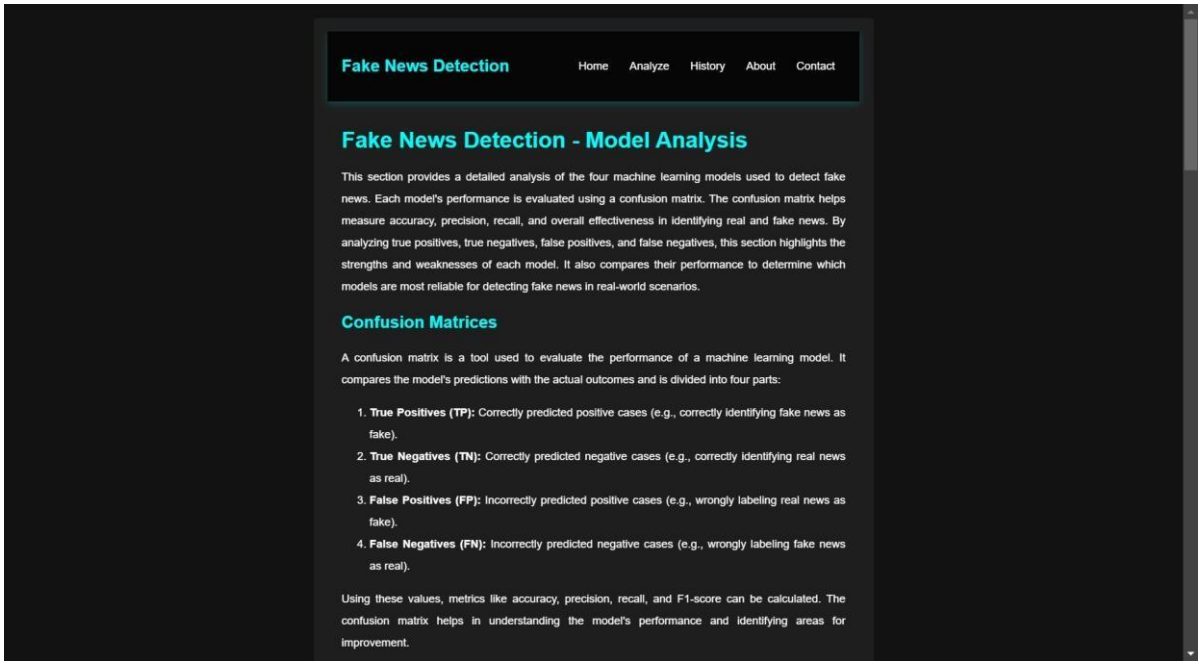
Other References

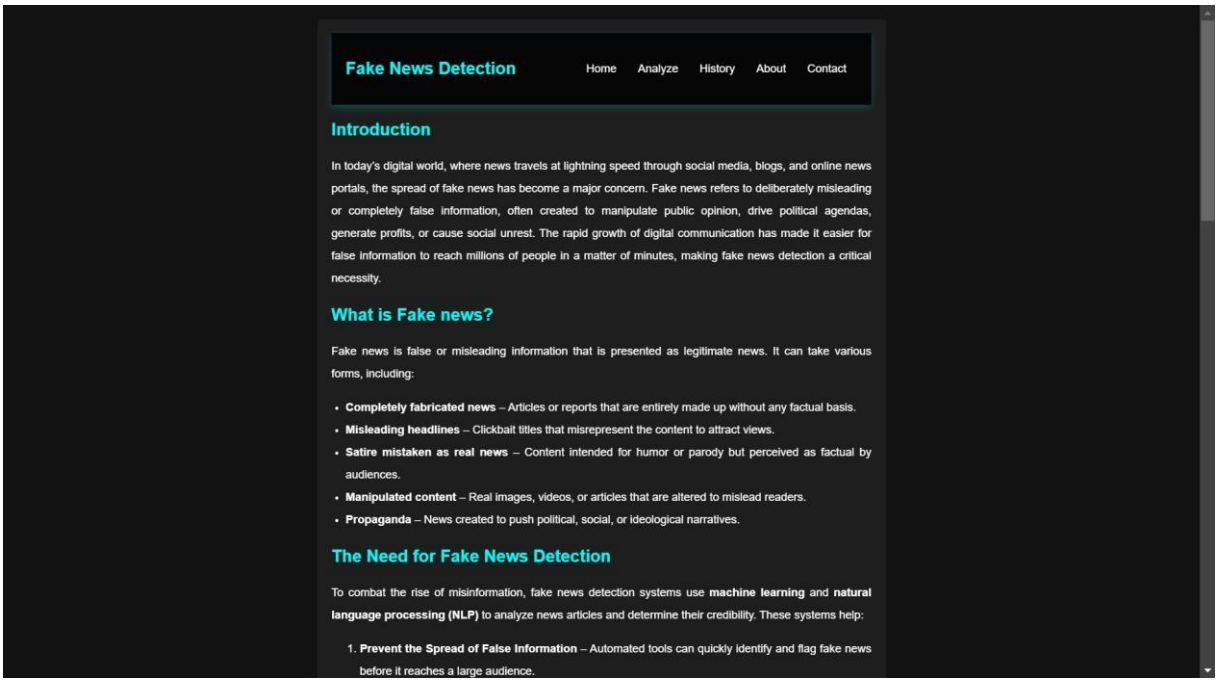
Online tutorials and courses from Coursera, Udacity, and YouTube.

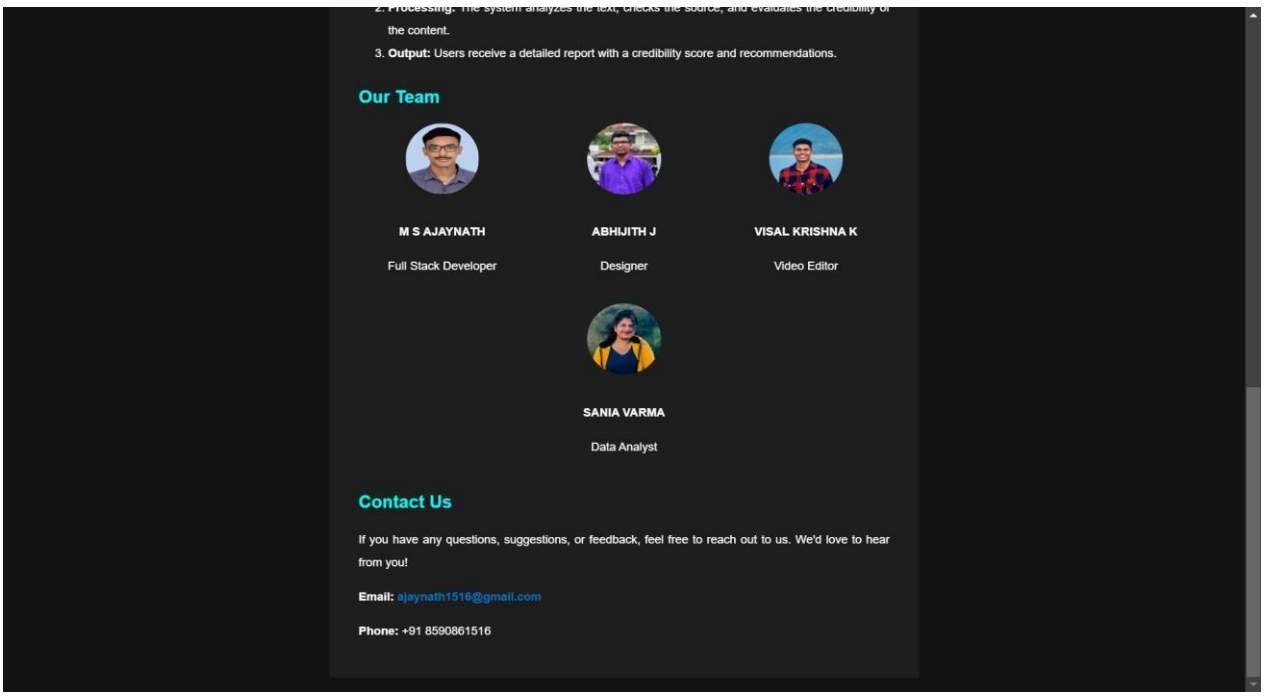
13.APPENDICES

13. 1 SCREENSHOTS OF OUR INTERFACE






Fake News Detection
Home
Analyze
History
About
Contact
Introduction
In today's digital world, where news travels at lightning speed through social media, blogs, and online news portals, the spread of fake news has become a major concern. Fake news refers to deliberately misleading or completely false information, often created to manipulate public opinion, drive political agendas, generate profits, or cause social unrest. The rapid growth of digital communication has made it easier for false information to reach millions of people in a matter of minutes, making fake news detection a critical necessity.
What is Fake news?
Fake news is false or misleading information that is presented as legitimate news. It can take various forms, including:
Completely fabricated news – Articles or reports that are entirely made up without any factual basis.
Misleading headlines – Clickbait titles that misrepresent the content to attract views.
Satire mistaken as real news – Content intended for humor or parody but perceived as factual by audiences.
Manipulated content – Real images, videos, or articles that are altered to mislead readers.
Propaganda – News created to push political, social, or ideological narratives.
The Need for Fake News Detection
To combat the rise of misinformation, fake news detection systems use machine learning and natural language processing (NLP) to analyze news articles and determine their credibility. These systems help:
1. Prevent the Spread of False Information – Automated tools can quickly identify and flag fake news before it reaches a large audience.


the content.
3. Output: Users receive a detailed report with a credibility score and recommendations.
Our Team
M S AJAYNATH
Full Stack Developer
ABHIJITH J
Designer
VISAL KRISHNA K
Video Editor
SANIA VARMA
Data Analyst
Contact Us
If you have any questions, suggestions, or feedback, feel free to reach out to us. We'd love to hear from you!
Email: ajaynath1516@gmail.com
Phone: +91 8590861516

13. 2 SAMPLE CODE

MAIN.PY

```
import pickle import
pandas as pd import
re
import string

# Load the saved Logistic Regression model with
open("logistic_model.pkl", "rb") as model_file:
    LR = pickle.load(model_file)

# Load the saved Decision tree model with
open("Decision_tree_model.pkl", "rb") as model_file:
    DT = pickle.load(model_file)

# Load the saved Gradient booster model with
open("GradientBooster_model.pkl", "rb") as model_file:
    GB = pickle.load(model_file)

# Load the saved Random forest model with
open("RandomForest_model.pkl", "rb") as model_file:
    RF = pickle.load(model_file)


# Load the saved TF-IDF vectorizer with
open("tfidf_vectorizer.pkl", "rb") as vectorizer_file:
    loaded_vectorizer = pickle.load(vectorizer_file)


print("Model and vectorizer loaded successfully!")


def wordopt(text):
    text = text.lower()  # Convert text to lowercase

    text = re.sub(r"□    □.*?      ", "", text)  # Remove text inside square
brackets     text = re.sub(r"[^\w\s]", "", text)  # Remove punctuation but
keep spaces     text = re.sub(r'https?://\S+|www\.\S+', "", text)  # Remove
URLs     text = re.sub(r'<.*?>+', "", text)  # Remove HTML tags
    text = re.sub(r"[%s]" % re.escape(string.punctuation), "", text)  # Remove punctuation
    text = re.sub(r'\n', "", text)  # Remove newlines
```

```
    text = re.sub(r'\w*\d\w*', "", text)  # Remove words containing numbers     return text


def     output_lable(n):
if n==0:
        return    "Fake    News"
elif n==1:
        return "Not A Fake News"

def  manual_testing(news):        testing_news  =
{"text":[news]}                    new_def_test     =
pd.DataFrame(testing_news)
    new_def_test['text']          =          new_def_test["text"].apply(wordopt)
new_x_test = new_def_test["text"]
    new_xv_test = loaded_vectorizer.transform(new_x_test)
pred_LR = LR.predict(new_xv_test)     pred_DT =
DT.predict(new_xv_test)     pred_GB =
GB.predict(new_xv_test)
    pred_RF = RF.predict(new_xv_test)

     return "\n\n<b>LR Predicition:</b> {} \n\n<b>DT Prediction:</b> {} \n\n<b>GB
Prediction:</b> {} \n\n<b>RF Prediction:</b> {}".format(output_lable(pred_LR[0]),
                                                        output_lable(pred_DT[0]),
                                                        output_lable(pred_GB[0]),
                                                        output_lable(pred_RF[0]))
#news = str(input("Enter the news : "))
#manual_testing(news)
```

APP.PY

```
from main import manual_testing
from flask import Flask, render_template, request, jsonify

app = Flask(__name__)

@app.route('/') def index():     return
render_template('index.html')
@app.route('/process',
methods=['POST']) def process():
```

```
data = request.json.get('user_input')  #
Get data sent from JavaScript
response_text = manual_testing(data)
# Process the input
    return jsonify(response=response_text)  # Send response back to frontend

@app.route('/about') def about():
return render_template('about.html')

@app.route('/Analyze') def Analyze():
return render_template('Analyze.html')

@app.route('/contact') def contact():
return render_template('contact.html')

if          __name__          ==          '__main__':
app.run(debug=True)
```