Using Machine Learning for Fake News Detection & Enhancing News Credibility

Submitted By

Mim Bin Hossain

Student ID: 2021-1-60-071

Rahma Mahbub

Student ID: 2020-2-60-026

Saurov Sikder

Student ID: 2021-1-60-053

Moinul Islam

Student ID: 2021-3-60-241

Submitted To

Yasin Sazid
Lecturer
Department of Computer Science & Engineering
East West University

Department of Computer Science and Engineering
East West University
Dhaka-1212, Bangladesh

Date: 22 May, 2025

Declaration

We, **Mim Bin Hossain, Rahma Mahbub, Saurov Sikder, and Moinul Islam**, hereby declare that the work presented in this project report is the outcome of the investigation performed by us under the supervision of **Yasin Sazid**, Lecturer, Department of Computer Science and Engineering, East West University. We also declare that no part of this project has been or is being submitted elsewhere for the award of any degree or diploma, except for publication.

Yasin Sazid
Lecturer
Department of Computer Science & Engineering
East West University

Mim Bin Hossain
Student ID: 2021-1-60-071

Rahma Mahbub
Student ID: 2020-2-60-026

Saurov Sikder
Student ID: 2021-1-60-053

Moinul Islam
Student ID: 2021-3-60-241

Abstract

In response to the escalating challenge of online misinformation, this project presents a comprehensive, real-time fake news detection system that integrates machine learning with modern web technologies and user-centered design. The system is composed of a responsive web application and a browser extension, both engineered to analyze news content instantly and deliver a dynamic credibility score to users. At its core, the system leverages five machine learning algorithms—Random Forest, Decision Tree, K-Nearest Neighbors, Support Vector Machine, and Gradient Boosting—trained on a balanced dataset comprising 4,000 news articles (2,000 real and 2,000 fake). Text features were extracted using advanced TF-IDF-based natural language processing techniques. The models achieved exceptional performance, with multiple algorithms exceeding 99% accuracy, validated using metrics such as precision, recall, F1-score, and ROC curves. The project further emphasizes usability through intuitive interface designs, developed via low- and high-fidelity prototyping using Figma. A style guide ensures visual consistency, and the system's functionality was evaluated based on Nielsen's usability heuristics and graphic design principles. The backend is implemented using Flask and supports real-time prediction through serialized model files (.pkl), enabling seamless interaction with the frontend. Addressing the limitations of existing solutions—such as lack of real-time detection, limited generalization, and poor multilingual adaptability—this system is designed for scalability and accessibility. Planned future work includes the integration of the browser extension with the main platform, multilingual support, and a dedicated mobile application to further enhance accessibility, user trust, and system robustness in diverse digital environments.

Contents

List of Figures

List of Tables

List of Acronyms

AI	Artificial Intelligence
API	Application Programming Interface
AUC	Area Under the Curve
CNN	Convolutional Neural Network
CSS	Cascading Style Sheets
DT	Decision Tree
F1-Score	Harmonic Mean of Precision and Recall
FND	Fake News Detection
GB	Gradient Boosting
JSCN	JavaScript Object Notation
kNN	k-Nearest Neighbors
KDE	Kernel Density Estimate
LMS	Learning Management System
ML	Machine Learning
NLP	Natural language processing
PHP	Hypertext Preprocessor
PKI	Public Key Infrastructure
SVM	Support Vector Machine
RF	Random Forest
ROC	Receiver Operating Characteristic
XML	Extensible Markup Language

CHAPTER 1: Introduction

In today's digital world, the widespread circulation of misinformation and fake news poses a serious threat to public trust and awareness. As people increasingly rely on online platforms and social media for news, distinguishing between factual and false information becomes more difficult. To address this issue, our project focuses on developing a system that can detect and classify fake news using machine learning. The system is designed to deliver fast and accurate results, helping users verify the credibility of news content within seconds. By combining web technology, machine learning models, and user-focused design, this project aims to provide a practical solution for enhancing online news transparency.

1.1 Project Theme

The core theme of this project is the detection of fake news using machine learning integrated with web-based technologies. It focuses on developing an intelligent, user-friendly platform capable of assisting individuals in verifying the authenticity of online news content in real time. In an era where misinformation spreads faster than ever, this project responds to a critical societal need by leveraging data science and AI to enhance media literacy and safeguard public discourse.

The theme embodies a multidisciplinary approach—combining natural language processing (NLP), supervised learning algorithms, and interactive web interfaces—to create a solution that is both technically sound and socially impactful. By enabling users to identify and report misinformation, the platform empowers individuals to take an active role in curbing the spread of fake news and fostering a more informed online ecosystem.

1.2 Project Overview

This project introduces a Fake News Detection (FND) system, built as a web application and a browser extension. The system allows users to input or select news content, which is then analyzed using multiple machine learning models. The backend is powered by models such as Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Gradient Boosting (GB), trained on a balanced dataset of 4000 news articles. These models are saved as `.pkl` files and accessed via a Flask API to generate real-time predictions, including the probability percentage of news being fake.

The system provides a clean, user-friendly interface and offers instant results, helping users make informed judgments about the content they consume. A high-fidelity prototype was also developed to simulate the full user experience, though the connection between the browser extension and the website remains part of the planned future work.

1.3 Objectives and Scope

Objectives

The primary goal of this project is to combat the growing threat of misinformation by creating a comprehensive fake news detection system. This system integrates machine learning, web technologies, and user-centric design to deliver accurate, real-time content validation. The specific objectives of the project are:

- To build a machine learning-based system that detects fake news accurately.
- To design a responsive and user-friendly website interface.
- To develop a functional browser extension for real-time news verification.
- To evaluate the effectiveness of various machine learning algorithms using performance metrics.

Scope

This project outlines the boundaries and primary focus areas of the fake news detection system during its current development phase. It highlights the core functionalities, limitations, and anticipated extensions of the system:

- Focused on text-based fake news (headlines and article content).
- Real-time detection via a trained ML model integrated through Flask API.
- Implementation of five ML algorithms for performance comparison.
- High-fidelity frontend design using Figma; limited frontend-backend integration in this phase.
- Future development includes integration of the browser extension with the main web application.

1.4 Stakeholders

The success and impact of the Fake News Detection system depend on the involvement and interests of various stakeholders, each playing a critical role in the system's development, evaluation, and future evolution.

- **General Users:** Individuals who consume online news and seek tools to verify its authenticity.
- **Project Developers:** Team members responsible for system design, ML model implementation, and interface development.
- **Supervisors and Faculty:** Academic reviewers overseeing the progress and quality of the project.
- **Future Researchers:** Developers or scholars interested in expanding this system or applying it to broader domains like multimedia misinformation or multilingual support.

CHAPTER 2: Low Fidelity Sketching

2.1 Sketching Process

Low-Fidelity Sketching is an early-stage design method used to quickly visualize and explore user interface ideas without focusing on visual details or final aesthetics. It helps designers and developers understand the structure and flow of the interface through simple hand-drawn or digital wireframes.

We began the sketching process by brainstorming key user interactions and essential system features related to fake news detection. The goal was to design a user-friendly interface that enables users to effectively identify and respond to potentially misleading news content. Based on these initial ideas, we created low-fidelity sketches using both pen-and-paper and digital tools like Figma and Balsamiq.

These sketches were focused on visualizing the core components of our system:

- Detection alerts
- News article layouts
- Trustworthiness indicators
- User feedback and reporting mechanisms

The process followed a structured approach:

Identifying Key Tasks

We first listed the main user tasks: reading news, checking article credibility, viewing explanation for flags, and submitting feedback.

Individual Sketching

We then sketched UI components individually, such as the article screen, alert banners, and feedback sections.

Interaction Flow Development

These components were arranged into a cohesive flow that reflects natural user behavior, enabling smooth navigation from reading an article to reviewing its credibility and taking appropriate action.

Collaboration and Feedback

Our group consisted of four members. Two members—Saurov Sikder and Moinul—each created two design prototypes based on their individual interpretations of the system. The remaining two members—Rahma Mahbub and Mim Bin Hossain—reviewed and provided constructive feedback on those designs.

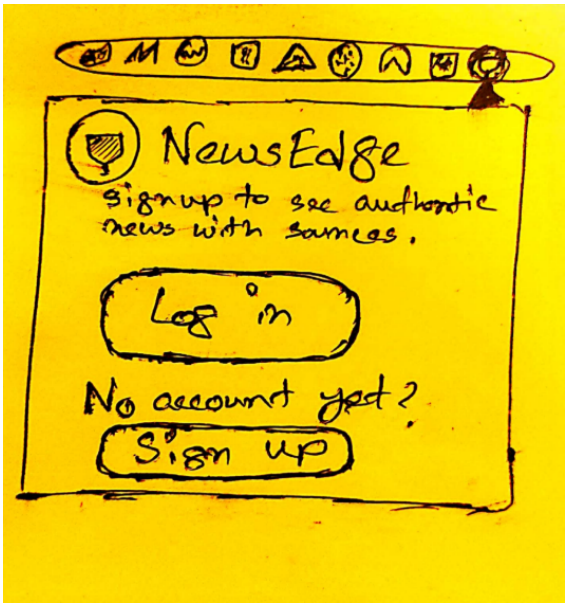


Figure 2.1: First Lo-Fi Sketch of the Fake News Detection System

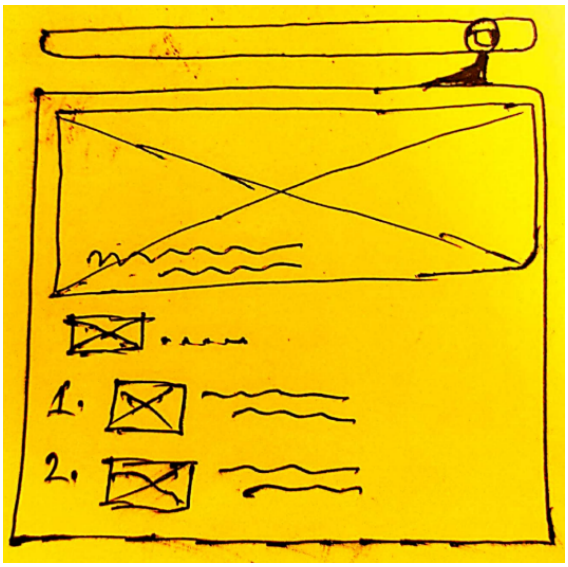


Figure 2.2: Second Lo-Fi Sketch of the Fake News Detection System

Figures 2.1 and 2.2 illustrate the initial Lo-Fi sketches of the Fake News Detection System, *NewsEdge*. Figure 2.1 presents the login/signup interface, featuring a top navigation bar with intuitive icons and a prompt encouraging users to sign up to access credible news content. It includes clear "Log in" and "Sign up" buttons, establishing the foundation for user authentication and personalization.

Figure 2.2 depicts the main news feed interface post-login, with a streamlined layout consisting of a top navigation/search bar, a prominent featured news banner, and a vertically scrollable list of news articles, each accompanied by an image thumbnail and summary. Together, these sketches provide a functional blueprint for user interaction and content display in the system's early design phase.

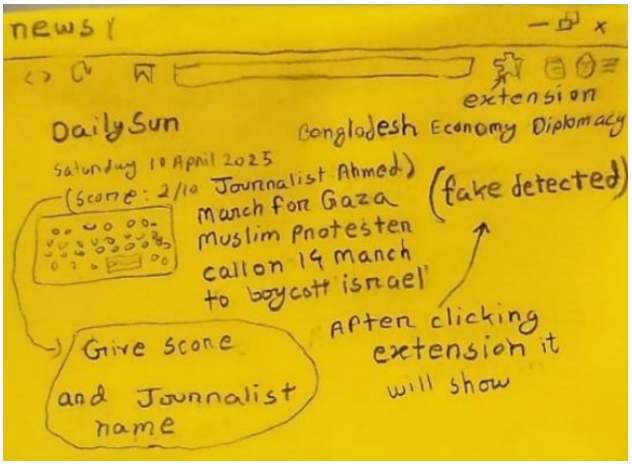


Figure 2.3: Third Lo-Fi Sketch of the Fake News Detection System

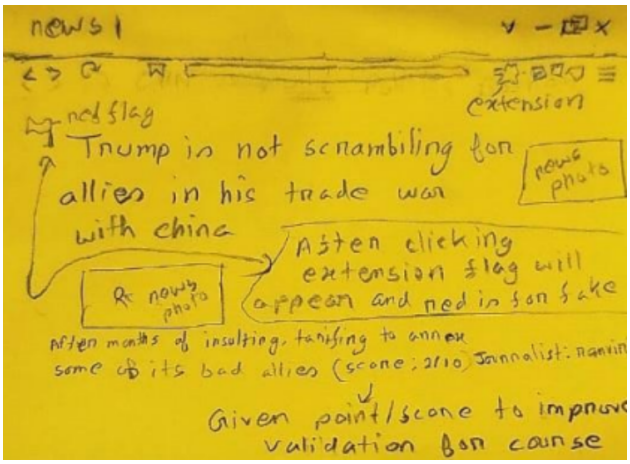


Figure 2.4: Fourth Lo-Fi Sketch of the Fake News Detection System

Figures 2.3 and 2.4 show Lo-Fi sketches of a proposed fake news detection system integrated into a web browser. In Figure 2.3, the extension highlights a news article as "fake detected" after analyzing its content, providing a reliability score and the journalist's name.

Figure 2.4 presents a similar interface where a red flag is used to mark suspicious content, and additional information is shown after the extension is activated. These sketches represent the system's core idea—providing users with instant, visual feedback on the credibility of online news, helping them make informed judgments.

In summary, the low-fidelity sketches served as an effective tool for rapidly prototyping user interface ideas and gathering early feedback. These designs laid a strong foundation for moving into high-fidelity mockups and user testing in later development stages.

CHAPTER 3: Background Study

3.1 Literature Review

Fake News Detection (FND) by Using Web Browser Extension

Fake News Detection (FND) using a Web Browser Extension involves integrating ML models into browsers to identify and flag fake news in real time. It helps users verify content instantly while browsing.

Rigger, Rakamaric, and Kim (2021) [1] introduced Check-It, a model checking tool designed for Java programs that leverages configurable property-based instrumentation. The tool automatically instruments Java code according to user-defined properties and utilizes Java PathFinder (JPF) to perform model checking. By supporting custom annotations and integrating static analysis, Check-It allows developers to define and verify specific software properties effectively. Despite its strengths in automation and customization, the tool faces limitations related to the inherent scalability challenges of model checking, the manual overhead required for property annotations, and its dependence on JPF, which limits compatibility with certain Java features and libraries.

In the 2024 paper by Merve Esra TAŞCI and Yonca BAYRAKDAR YILMAZ [2] the authors propose a Chrome extension using machine learning to detect fake news. The study applied algorithms including Passive Aggressive, Random Forest, SVM, AdaBoost, XGBoost, and LSTM, with LSTM achieving the highest accuracy (90.72%). Natural Language Processing and TF-IDF were used for text preprocessing, and the extension was developed using Flask, JavaScript, HTML, and CSS. While the extension effectively classifies fake news, limitations include support for only English, lack of explanation in results, and reduced accuracy on imbalanced datasets like culture and arts news.

In the 2022 paper "A Web Infrastructure for Certifying Multimedia News Content for Fake News Defense", Edward L. Amoruso, Stephen P. Johnson, Raghu Avula, and Cliff C. Zou [3] propose a browser-based certification system using Public Key Infrastructure (PKI) to combat fake multimedia news. The system enables news organizations to digitally sign content using private keys, storing verification data in XMP sidecar files. A JavaScript-based browser extension validates these files using cryptographic hashes and displays visual cues (green/red borders) on media content to indicate authenticity. Unlike detection-based models, this approach emphasizes trusted endorsement over truth verification. However, it faces limitations such as incompatibility with social media compression, lack of trust for small publishers, limited browser support, and challenges in verifying large or streamed videos.

Table 3.1: Recent Tools for Fake News Detection and Verification by Using Web Browser Extension

Authors	Year	Findings	Drawback
Rigger, Rakamaric, and Kim [1]	2021	Developed Check-It for Java model checking using custom annotations and JPF.	Scalability issues, manual setup, and limited Java compatibility.
TAŞCI and YILMAZ [2]	2024	Created Chrome extension using ML (LSTM 90.72%) and TF-IDF.	English-only, no result explanation, poor on imbalanced data.
Amoruso et al. [3]	2022	PKI-based browser tool to certify multimedia news using digital signatures.	Incompatible with social media, limited trust, and browser support.

Fake News Detection (FND) by Using Machine Learning (ML) Models

Machine learning (ML) models have revolutionized fake news detection by enhancing accuracy, speed, and scalability in identifying misinformation across digital platforms. This section explores key studies on ML-driven solutions for fake news detection, highlighting advancements in NLP, challenges related to data bias and adversarial content, and the potential of these models in strengthening information credibility and promoting digital literacy.

In 2022, Noshin Nirvana Prachi et al. [4] conducted a study on fake news detection using a combination of traditional machine learning algorithms—such as Logistic Regression, Naive Bayes, Decision Tree, and SVM—and advanced deep learning/NLP methods, including LSTM and BERT. The models were trained and evaluated on a Kaggle dataset using preprocessing techniques like tokenization, lemmatization, and TF-IDF. Among all the models, BERT achieved the highest accuracy at 98%, followed by LSTM at 95%. Traditional machine learning models performed moderately well, with the Decision Tree leading at 89.66%. However, the study was limited to text-based news and did not address real-time or multilingual detection capabilities.

Cavus, Goksu, and Oktekin (2024) [5] developed a cloud-based fake news detection system using the BERT algorithm, achieving 99% overall accuracy. The system classifies content into seven categories: clickbait, disinformation, hoax, junk news, misinformation, propaganda, and satire. Most categories reached 100% accuracy, while misinformation and propaganda were slightly lower at 94% and 99%, respectively. However, the study is limited to Twitter text data, excludes multimedia content, and focuses specifically on COVID-19-related tweets, which may affect its general applicability.

Jouhar, Pratap, Tijo, and Mony (2024) [6] explored fake news detection using six machine learning algorithms, including Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, Passive Aggressive Classifier (PAC), and XGBoost. Using the ISOT dataset and applying TF-IDF for feature extraction, the study found XGBoost to be the most effective, achieving a testing accuracy of 99.77%, with precision, recall, and F1-score of 99.24%, 99.86% , and 99.75%, respectively. The authors acknowledged limitations such as the absence of real-time detection, lack of multimedia analysis, and no hyperparameter tuning. They proposed future work involving advanced techniques like Word2Vec, BERT, and real-time monitoring systems to enhance performance and adaptability.

Jain et al. (2019) [7] proposed a smart system for fake news detection by leveraging machine learning techniques, specifically Naive Bayes and Support Vector Machine (SVM), combined with Natural Language Processing (NLP) methods. Their model achieved a notable accuracy of 93.5%, demonstrating improved performance over several existing approaches. However, the study was constrained by its dependence on traditional machine learning algorithms and lacked integration of more advanced deep learning methods or the ability to process large-scale, real-time data streams, limiting its applicability in dynamic environments.

In the study titled "IoT-enabled waste management system using deep learning models" by R. Vijayalakshmi, B. Anand, A. Kannan, and S. Karthikeyan (2024) [8], the authors proposed a smart waste classification system leveraging deep learning techniques to distinguish between biodegradable and non-biodegradable waste. The research evaluated the performance of various models including Convolutional Neural Networks (CNN), VGG-16, ResNet-50, and InceptionV3. Among these, ResNet-50 achieved the highest classification accuracy of 94.9%, demonstrating its effectiveness in automated waste sorting. While the results highlight the potential of deep learning in intelligent waste management, the study's reliance on pre-trained models, limited dataset diversity, and absence of real-world deployment introduce concerns about generalizability and scalability in practical environments.

Yashaswini N, Srinath G S, Nagaraj A, Sudharshan S, Sayed Faizal, and Rajkumar N (2023) conducted a study published in the Journal of Propulsion Technology [9] that explores real-time fake news detection using various machine learning techniques. The research employed algorithms such as Logistic Regression, Random Forest, K-Nearest Neighbors, Support Vector Classifier, and the TF-IDF Vectorizer for feature extraction. Notably, the combination of TF-IDF with Logistic Regression and Random Forest yielded the highest accuracy, reaching up to 99.45%, indicating strong predictive performance. However, the study also highlights limitations related to the quality and diversity of the dataset, which may affect the model's generalizability across various news formats. The authors suggest further research to enhance detection capabilities, especially for non-textual data types such as audio and video content.

Thakur (2018) [10] conducted a study on fake news detection utilizing deep learning techniques, focusing on models such as Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), and Bidirectional LSTM.

The research demonstrated that the LSTM model outperformed the other architectures, achieving an accuracy of % in classifying fake news. This finding highlights the effectiveness of LSTM in capturing sequential dependencies in textual data for fake news detection. However, the study also identified certain limitations, including the use of a relatively small dataset and the lack of evaluation in real-time environments. These constraints suggest that while the models show promise, their generalizability and robustness in practical applications remain to be further validated.

Nasir, Khan, and Varlamis (2021) [11] developed a hybrid CNN-RNN model with LSTM units for fake news detection, achieving accuracies of 0.60 on the FA-KES dataset and 0.99 on ISOT. The model outperformed standalone CNN, RNN, and traditional classifiers. However, it showed poor generalization across datasets, indicating overfitting and limited adaptability in cross-domain scenarios.

Wang et al. (2023) [12] introduced SLFEND, a novel approach for multi-domain fake news detection that integrates soft labeling, a modified Leap-GRU architecture, and expert group-based feature extraction. Evaluated on the Weibo21 and Thu datasets, the model achieved impressive F1 scores of 92.49% and 89.98%, respectively, outperforming several state-of-the-art methods. Despite its strong performance, the study noted limitations including high training complexity and a focus on Chinese-language datasets, which may restrict the model's applicability and generalizability across different languages and cultural contexts.

Table 3.2: Recent Studies on Fake News Detection Using ML Models

Authors	**Year**	**Findings**	**Drawbacks**
Noshin N. Prachi et al. [4]	2022	BERT: 98%, LSTM: 95%, DT: 89.66%; preprocessing with TF-IDF.	Only text-based; lacks real-time and multilingual support.
Cavus, Goksu et al. [5]	2024	BERT-based cloud model with 99% accuracy; classified 7 categories.	Limited to COVID-19 tweets; no multimedia input.
Jouhar et al. [6]	2024	XGBoost: 99.77% accuracy, F1-score: 99.75%; used ISOT dataset.	No real-time or multimedia support; no tuning.
Jain et al. [7]	2019	Naive Bayes and SVM: 93.5%; used basic NLP techniques.	Relies on traditional ML; lacks deep learning and scalability.
Yashaswini N. et al. [8]	2023	TF-IDF + Logistic Regression and Random Forest achieved 99.45% accuracy.	Dataset quality and diversity limit generalizability; no multimedia data considered.
Vijayalakshmi et al. [9]	2024	ResNet-50 achieved 94.9% accuracy in classifying waste types using deep learning.	Used pre-trained models; limited dataset diversity; no real-world deployment.
Thakur [10]	2018	LSTM outperformed CNN and BiLSTM for fake news detection (accuracy not specified).	Small dataset; no real-time evaluation, limiting generalizability.
Nasir, Khan, Varlamis [11]	2021	Hybrid CNN-RNN with LSTM achieved 0.60 (FA-KES) and 0.99 (ISOT) accuracies.	Poor generalization; overfitting; limited cross-domain adaptability.
Wang et al. [12]	2023	SLFEND model achieved F1 scores of 92.49% (Weibo21) and 89.98% (Thu).	High training complexity; focused on Chinese datasets limiting broader applicability.

3.2 Insights Gained

Before finalizing our user interface design, we conducted a background study on user trust, alert efficiency, and UI preferences for fake news detection systems. Our research highlighted several insights that were directly translated into design decisions to improve usability, clarity, and user confidence.

Table 3.3: Key Research Insights and Corresponding Design Implementations

Research Insight	**Design Implementation**
Minimal but informative alerts	Used icon-based status indicators with color codes (red, yellow, green) and short, non-intrusive labels (e.g., "Unverified Source").
Users prefer visual summaries over text-heavy explanations	Designed the UI with charts, trust meters, and badges to display detection results in a concise, interpretable format.
Trust and transparency are central to user acceptance	Added "See Why" buttons, fact-check summaries, and source credibility analysis to build user trust.
Users need control and feedback options	Provided options to agree/disagree with detection results and submit feedback to improve system learning.
Overwhelming text reduces attention and comprehension	Replaced long alerts with tooltips and context-aware popups that appear only when necessary.

As seen in Table 3.3, each design choice was informed by a specific research insight. These adaptations allowed us to create a more user-centric and trustworthy interface. By focusing on visual clarity, transparency, and user feedback mechanisms, we aligned the FND system with users' real-world expectations and cognitive preferences—critical aspects for the adoption of any HCI-based detection system.

CHAPTER 4: User Research

4.1 Research Questions Development

To design an effective fake news detection tool that aligns with real user needs, we developed two foundational research questions to guide our user research. These questions are intended to uncover both the challenges users face and the expectations they have for technological interventions.

Research Question 1

What difficulties do users face when trying to identify fake or misleading news on social media and online platforms?

Motivation: In today's digital landscape, users are constantly bombarded with a flood of news from multiple sources, many of which lack credibility. Despite increased awareness about misinformation, users often struggle to distinguish real news from false or misleading content. This question aims to explore these difficulties in-depth to understand the cognitive, technical, and emotional barriers that prevent users from effectively verifying news.

Participant Information to Be Collected:

- **Habits:** How often and where participants consume news (platforms, formats, times of day).
- **Challenges:** Specific obstacles or confusion they experience when determining the credibility of news.

Design Implications: Understanding the nature of these challenges will help us tailor the system to the user's mental model and pain points. For instance, if users struggle with clickbait headlines or unreliable sources, the system can emphasize headline analysis and credibility scoring. These insights will inform features such as real-time credibility alerts, source transparency, and content flagging mechanisms.

Research Question 2

What features, formats, and design attributes do users expect or prefer in a fake news detection tool?

Motivation: A fake news detection system must not only be functional but also user-friendly, trustworthy, and seamlessly integrated into the user's digital ecosystem. Understanding users' expectations, preferences, and attitudes towards such tools will ensure that our solution is adopted and used effectively.

Participant Information to Be Collected:

- **Preferences:** Desired tool format (e.g., browser extension, mobile app), notification style, feedback mechanisms.
- **Attitudes:** Comfort levels with automated detection, data privacy, and AI-driven suggestions.

Design Implications: This research will directly inform UI/UX design decisions, such as the tool's delivery format, how alerts or verifications are displayed, and what customization options should be available.

4.2 Interview Questionnaire

The following open-ended interview questions were designed to elicit detailed and insightful responses from participants. Each question is carefully aligned with one of the two previously defined research questions and structured to explore user behaviors, frustrations, motivations, and expectations regarding fake news detection tools.

Aligned with Research Question 1

This set of questions is designed to uncover the real-world difficulties users face when attempting to identify fake or misleading information in digital spaces. By exploring participants' behaviors, thought processes, and emotional responses, we aim to understand the specific barriers that hinder accurate news evaluation. Insights gathered here will inform the system's ability to support users in decision-making, build trust, and reduce cognitive load.

1. Can you walk me through how you usually get your news online or through social media?
2. Have you ever come across a piece of news online that later turned out to be false or misleading? How did you realize it?
3. When you're unsure if a news story is true, what steps — if any — do you take to verify it?
4. What makes it hard for you to decide if a piece of news is trustworthy or not?
5. Do you ever share news articles with others? What influences your decision to share or not share?

Aligned with Research Question 2

This subsection explores user preferences, expectations, and trust factors regarding the design and functionality of fake news detection tools. These questions aim to gather insights into the ideal format, interaction style, and key features users would find useful, acceptable, and trustworthy in real-world scenarios. Understanding these expectations will directly shape our tool's user experience, ensuring high adoption and usability.

1. If a tool could help you identify fake news, what kind of support would you find most useful?
2. Would you prefer to use this kind of tool as a browser extension, mobile app, or some other form? Why?
3. How would you feel about the tool analyzing your feed and flagging potentially fake content automatically?
4. What would make you trust the results or suggestions provided by such a tool?

4.3 Alternate or Complementary Research Method: Survey

To complement the depth and nuance gained through qualitative interviews, we employed a quantitative survey method. This approach aimed to validate initial findings, identify broader trends, and capture statistically relevant insights from a larger, more diverse participant base. The survey transformed selected interview questions into structured response formats such as multiple-choice, Likert scales, and checkboxes, enabling efficient data collection and analysis.
The survey was designed with the following objectives:

- To understand user behavior and preferences in consuming online news.
- To identify common challenges users face in distinguishing fake news.
- To assess user expectations and design preferences for a fake news detection tool.
- To measure trust factors and privacy concerns associated with automated tools.

4.3.1 Interview Question

To better understand users' experiences with and perceptions of online news, a set of open-ended interview questions was developed. These questions were designed to capture rich, qualitative insights into user behavior, challenges in identifying misinformation, and preferences for technological solutions.

Below are examples of how original interview questions were adapted for the survey:

1. **How do you usually get your news online or through social media?**
2. **What makes it hard for you to decide if a piece of news is trustworthy?**
3. **Would you prefer to use a fake news detection tool as a browser extension, mobile app, or in another form?**
4. **What would make you trust the results or suggestions provided by such a tool?**

4.3.2 Adapted Survey Question Format

To collect structured and scalable data from a broad audience, the following survey questions were developed based on key themes from the interviews. These questions are designed to quantify user habits, preferences, and trust-related factors in the context of fake news detection.

1. **Select the platforms you use for news:**
   a. Facebook b. Twitter/X c. Instagram
   d. YouTube e. News Apps f. Others: _____
2. **What challenges do you face in identifying fake news?**
   (Check all that apply)
   a. Convincing or clickbait headlines b. Biased sources c. Time pressure
   d. Information overload e. Others: _____
3. **Which format would you prefer for a fake news detection tool?**
   (Select one)
   ◯ Browser extension ◯ Mobile app ◯ Both ◯ Neither
4. **What would make you trust a tool's suggestions?**
   (Choose up to three)
   a. Verified sources b. Transparent scoring method c. Human fact-checkers
   d. Partner organizations (e.g., universities, NGOs) e. Simple and user-friendly design
   f. Others: _____

These survey questions, adapted from in-depth interviews, provided a scalable method for gathering user insights at a larger scale. By translating open-ended responses into structured formats, the survey allowed for quantitative analysis of trends in news consumption, common challenges in detecting misinformation, preferred technological formats, and trust factors. The feedback collected through this instrument directly informed the design decisions and development priorities for the proposed fake news detection solution, ensuring user-centered functionality and relevance.

4.4 Data Collection

To understand user behavior, challenges, and preferences related to fake news detection, a mixed-method data collection approach was employed. This included both qualitative interviews and a quantitative survey distributed to a diverse group of participants. The survey aimed to gather insights into how users consume news, the difficulties they face in identifying misinformation, their preferred formats for detection tools, and the features they trust most.

The following analysis presents key findings from the survey responses, which serve as a foundation for the system's design decisions, ensuring that the final solution aligns with real user needs and expectations.

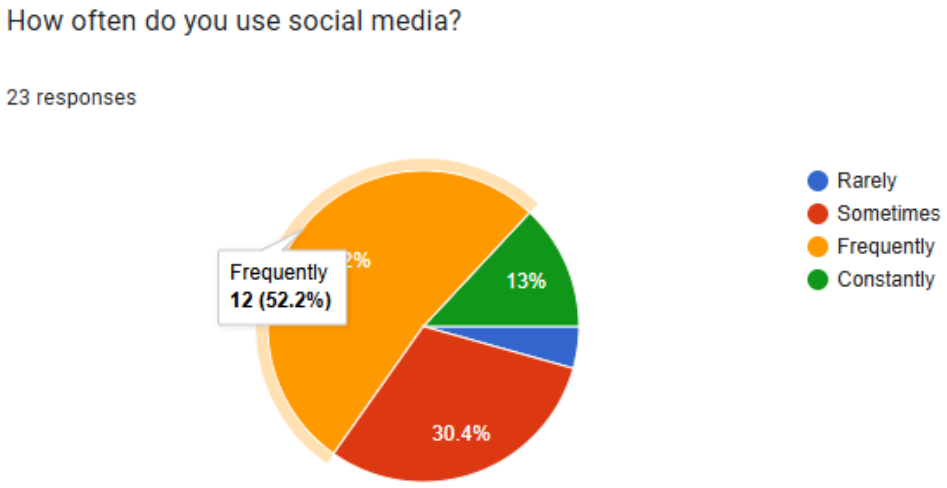


Figure 4.1: Frequency of Social Media Usage for Daily News Consumption

Figure 4.1 illustrates that social media is a prominent channel for daily news consumption. According to the survey results, 52.2% of respondents reported frequent use, while 13% indicated constant usage. In contrast, only 30.4% stated they rarely use social media for news, highlighting the widespread exposure to online content and the associated risk of misinformation.

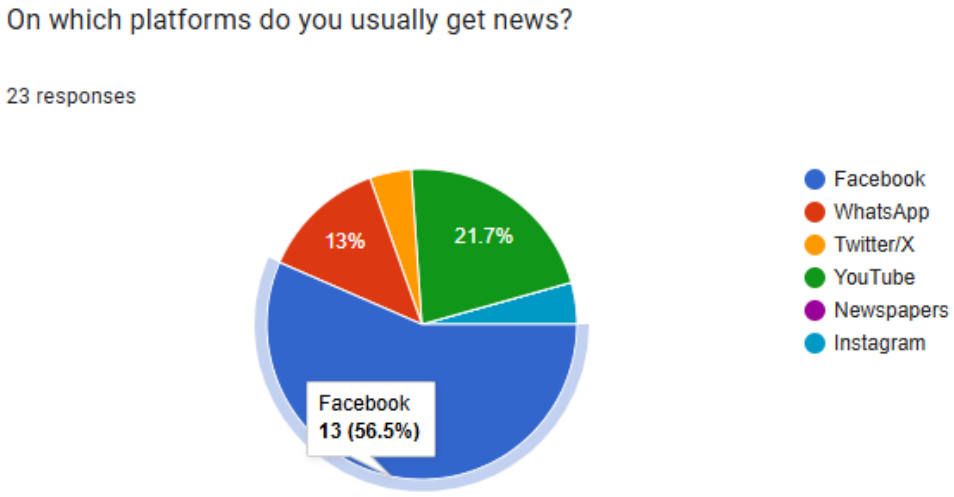


Figure 4.2: Preferred Platforms for News Consumption Among Respondents

As shown in Figure 4.2, Facebook stands out as the primary news source for 56.5% of respondents. This is followed by WhatsApp, YouTube, and traditional newspapers, each used by 21.7% of participants. In contrast, platforms such as Instagram and Twitter/X were reported as less frequently used for consuming news.

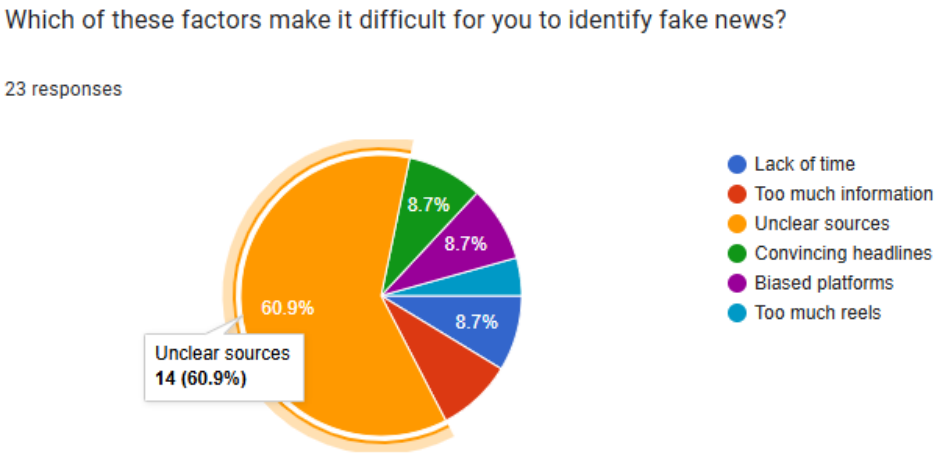


Figure 4.3: Challenges Faced by Users in Identifying Fake News

Figure 4.3 shows that 60.9% of respondents reported difficulty in identifying fake news due to unclear or untrustworthy sources. Other notable challenges include information overload, biased platforms, misleading headlines, and lack of time—each cited by 11.8% of participants. These findings highlight the complexity of misinformation detection in digital news environments.

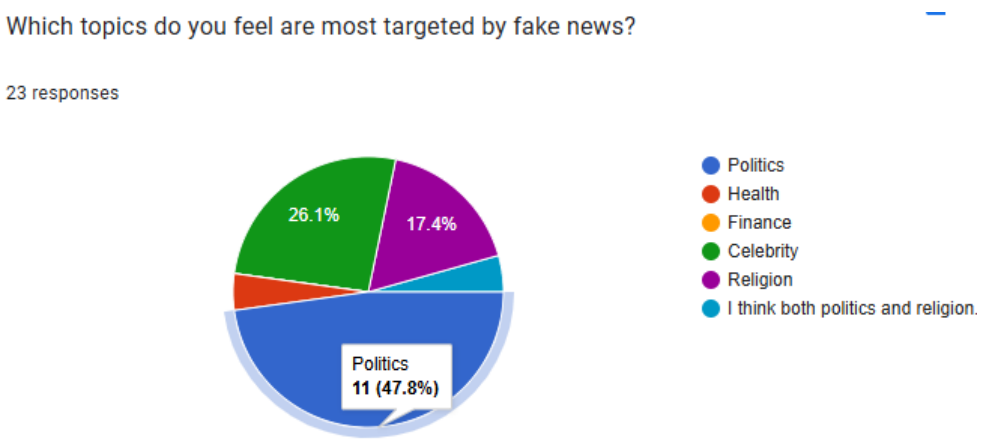


Figure 4.4: Topics Most Commonly Targeted by Misinformation

Figure 4.4 illustrates that political content is perceived as the most frequent target of misinformation, cited by 47.8% of respondents. Celebrity news follows at 26.1%, with religion at 17.8%. Health and finance topics were less commonly mentioned, indicating that misinformation may be more prevalent or more noticeable in socially or emotionally charged content areas.

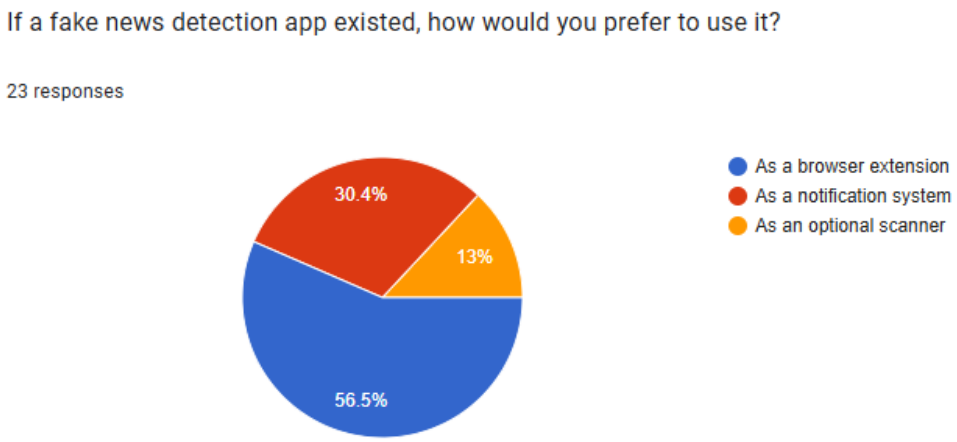


Figure 4.5: Preferred Formats for Fake News Detection Tools

Figure 4.5 highlights user preferences regarding the format of fake news detection tools. A majority of respondents (56.5%) prefer browser extensions, appreciating their real-time detection capabilities and user convenience. Notification-based systems were selected by 30.4%, while 13% favored optional scanning tools. These findings emphasize the importance of providing flexible and context-appropriate detection formats to suit varied user habits.

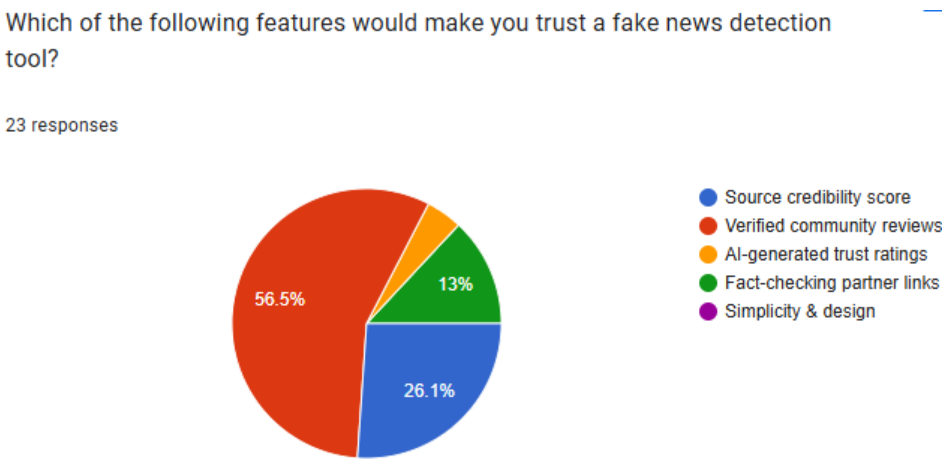


Figure 4.6: User Preferences for Features That Build Trust in Fake News Detection Tools

Figure 4.6 illustrates the key features that users believe would enhance their trust in fake news detection tools. The majority (56.5%) favor verified community reviews, while 26.1% prioritize source credibility scores. This preference suggests that users place greater confidence in transparent, human-informed assessments compared to solely automated AI-generated outputs.

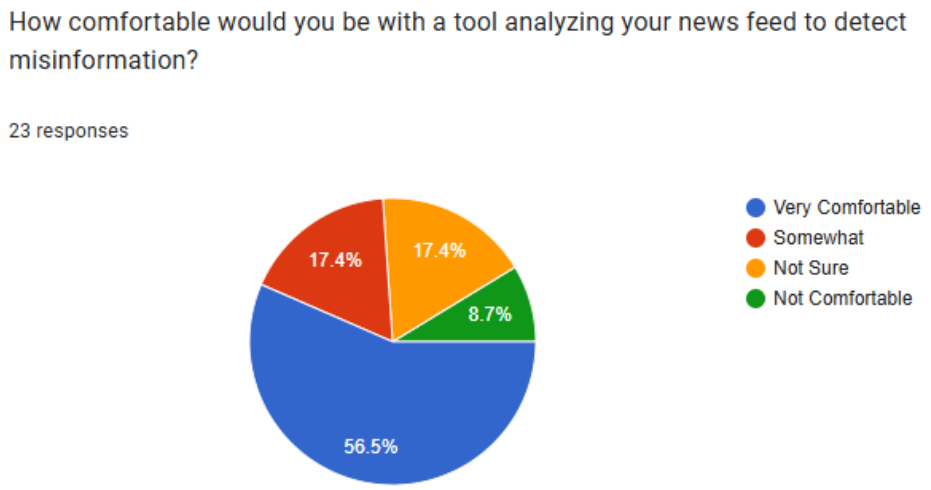


Figure 4.7: User Comfort Level with Automated Analysis of Their News Feeds

Figure 4.7 shows that 56.5% of respondents are comfortable with a fake news detection tool analyzing their news feeds. However, a notable portion expressed concerns about privacy, emphasizing the importance of implementing opt-in options and customizable privacy settings to build trust and encourage wider adoption.

4.4.1 Summary of Survey Findings

Table 4.1: Summary of Key Survey Findings

Survey Aspect	**Percentage / Preference**
Primary News Platform (Facebook)	56.5%
Frequent Social Media News Users	52.2%
Difficulty Identifying Fake News (Unclear Sources)	60.9%
Most Targeted Misinformation Topic	Politics (47.8%)
Preferred Tool Format	Browser Extension (56.5%)
Top Trust Feature	Verified Community Reviews (56.5%)
Comfortable with News Feed Analysis	56.5%

The summarized findings in Table 4.1 offer valuable insights into user behavior and expectations regarding fake news detection. The dominance of Facebook as a news source and the high frequency of social media usage suggest a significant vulnerability to misinformation. A majority of users find it challenging to identify fake news due to unclear sources, and political content is perceived as the most common target. Browser extensions emerged as the preferred detection format, and users place high trust in verified community reviews. Notably, over half the respondents are comfortable with automated tools analyzing their feeds, provided privacy settings are customizable. These results will guide the design and functionality of the proposed fake news detection solution.

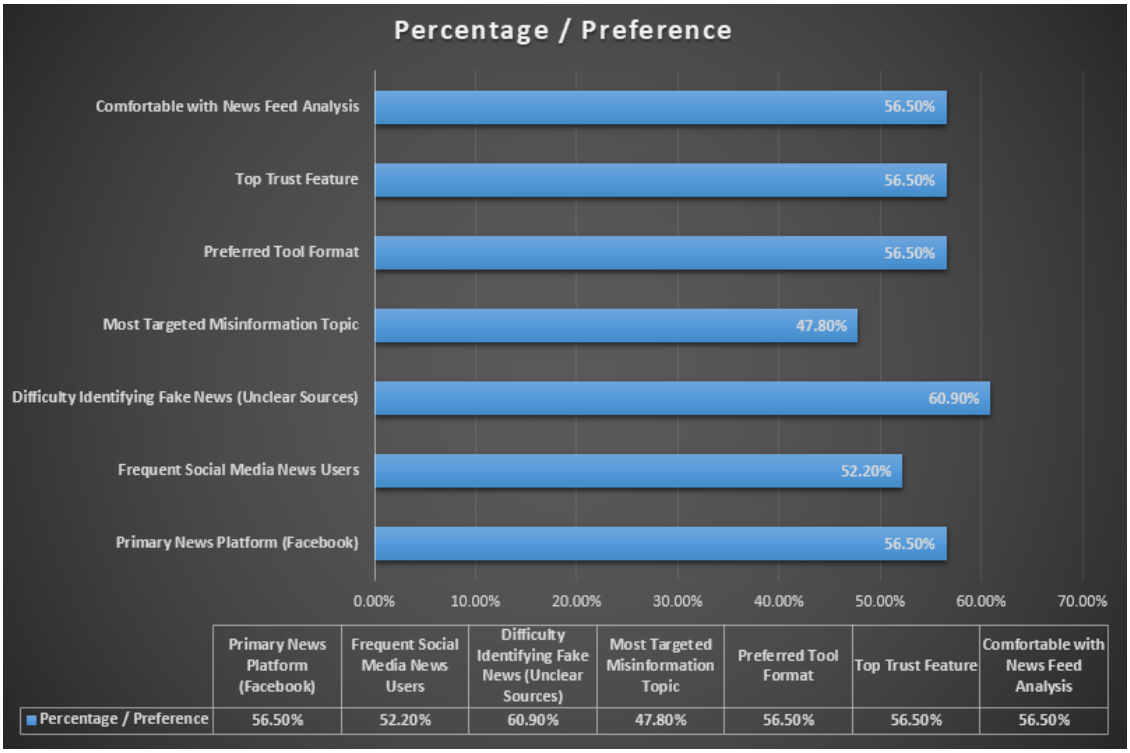


Figure 4.8: Visualization of Survey Findings in Graphical Format

This figure 4.8 summarizes key insights from the user survey using graphical representations. It highlights major preferences, challenges, and behaviors regarding fake news detection.

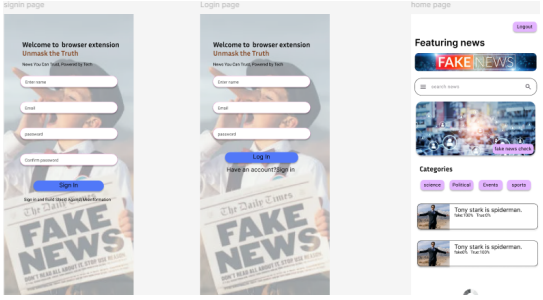
CHAPTER 5: High Fidelity Prototyping

High-Fidelity (Hi-Fi) Prototyping refers to the process of creating detailed, interactive, and realistic representations of a digital product or system that closely resemble the final design in terms of visual design, content, and user interactions. Unlike low-fidelity sketches, which focus on structure and layout, high-fidelity prototypes simulate the look, feel, and functionality of the final system.

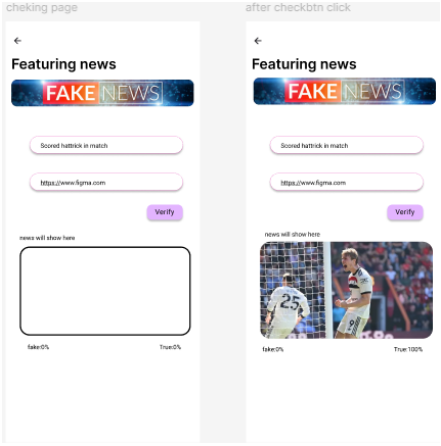
For our Fake News Detection (FND) system, we used Figma as the primary tool to develop high-fidelity prototypes. Figma enabled us to design pixel-accurate user interfaces, define interactive flows, and collaborate effectively as a team. This helped us translate initial conceptual sketches into clickable, user-friendly mockups that mimic real application behavior.

5.1 Application Interface Overview

(a) Splash Screens



(b) News Check Interface



(c) Auth and Home Pages

Figure 5.1: App UI Screenshots: Splash, Verification, and Home

A style guide in Figma involves defining consistent typography, including font styles, sizes, and weights, and saving them as text styles. It also includes setting up a color palette with primary, secondary, and neutral colors, which can be saved as color styles for easy reuse. Icons should be created or imported and used consistently across the design. Buttons, checkboxes, dropdowns, and form elements should be designed as reusable components. Spacing, padding, and layout grids should be standardized for consistency. Finally, the style guide should include the brand's logo, colors, and design elements.

Mockups in Figma are high-fidelity designs that represent the final product in detail. They include design layouts with images, text, buttons, and icons placed within the frame to form a complete design. Reusable components, such as buttons and navigation bars, can be used throughout the mockup for efficiency. It's important to add realistic details such as actual images, high-quality icons, and fine design touches to make the mockup closely resemble the finished product.

Wireframes in Figma are low-fidelity designs that focus on the layout, structure, and functionality without getting into detailed design elements. They use simple shapes like rectangles to represent buttons, images, and text. The purpose is to outline where key elements will be placed and how the navigation will be organized. Wireframes often employ a grid system for consistent alignment and clean layouts, ensuring that the structure of the design is organized and functional.

The prototype that we have made in the Figma tool contains mockup, wireframe, and prototype, which helps the user the to understand the design interface properly. We have also a style guide which helps the user or any other developer to use color efficiently and without any hesitation. In the style guide all hover button then blinker also text sizes will find, which are applied in the mockup and prototype. We also built an website prototype which was simple so focused on code to show the actual product.

The prototype is designed as app that helps users identify and verify the authenticity of online news. The main objective is to combat misinformation and fake news using an intuitive user interface backed by AI-based verification logic. The high-fidelity prototype includes the following key screens:

- **Splash Screens:** A sequence of animated splash pages with a "Fake News" emblem and the tagline "Spot the Truth" to introduce the theme of the application.
- **Verification Page:** Allows users to input a news headline and source URL. Upon clicking the "Verify" button, it displays whether the news is fake or true with corresponding percentages and visual feedback.
- **Authentication and Home Interface:** Includes sign-in and login screens to register or access the platform, along with a homepage that features trending news, search functionality, category filters (e.g., science, political, events), and AI-based truth verification results.

These screens were developed using a high-fidelity prototyping tool to simulate the real-user experience, ensuring clarity, usability, and responsiveness across different user interactions.

Figma Prototype: Click here to view the interactive Figma design.

5.2 Style Guide

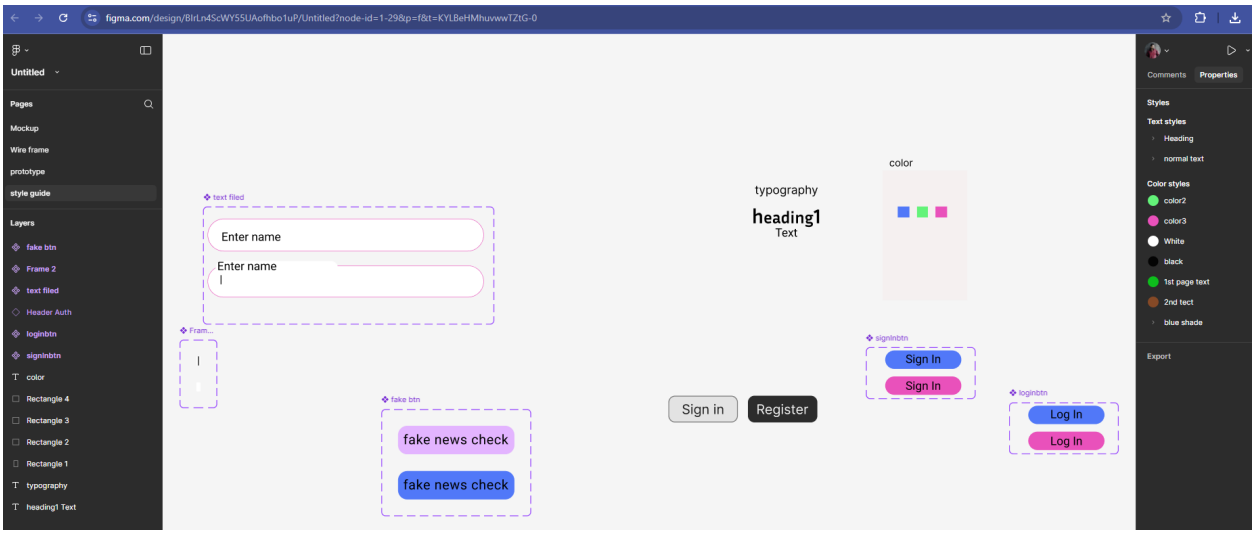


Figure 5.2: Figma Style Guide of Our Prototype

The style guide serves as a visual reference to maintain design consistency across the user interface. It includes standardized components such as text fields, buttons, typography, and a defined color palette. Typography styles are categorized into a bold ”heading1” for primary headings and a regular ”Text” style for body content. Button variations—such as “Sign In,” “Register,” “Log In,” and “Fake News Check”—are presented in both blue and pink versions to reflect different actions or states. The color scheme incorporates bright tones like blue, green, and pink alongside foundational colors such as white and black, each labeled (e.g., *color2*, *color3*, *blue shade*) for clarity. This unified design approach enhances the overall user experience and ensures visual consistency throughout the application.

Figma Style Guide: Click here to view the interactive Style Guide of design.

CHAPTER 6: Interface Evaluation

6.1 Evaluation Using Nielsen's 10 Usability Heuristics

The prototype was assessed using Jakob Nielsen's 10 Usability Heuristics to identify potential usability issues. Each heuristic highlights a specific problem, its severity, and a proposed solution to enhance user experience.

Table 6.1: Evaluation of Prototype Based on Nielsen's 10 Usability Heuristics

Heuristic	Identified Problem	Severity	Proposed Solution
Visibility of System Status	No feedback is provided while verifying news articles.	High	Integrate a progress indicator or loading animation to inform users during AI processing.
Match Between System and the Real World	The term "AI truth ratio" may confuse non-technical users.	Medium	Replace it with intuitive terms such as "Credibility Score" or "Trust Index."
User Control and Freedom	Users cannot undo actions or navigate backward during verification.	High	Add "Back" and "Clear" buttons for enhanced navigation and flexibility.
Consistency and Standards	Inconsistent button styles across different screens.	Medium	Apply a universal style guide to enforce visual and functional consistency.
Error Prevention	Users can submit invalid or empty URLs without any warning.	Critical	Implement input validation to restrict incorrect submissions and display alerts for empty or malformed URLs.
Recognition Rather Than Recall	Users must re-enter previously checked news articles.	High	Add autocomplete or a searchable input history to reduce user effort.
Flexibility and Efficiency of Use	Lack of personalization for returning users.	Medium	Introduce features like "Remember Me" or allow users to save preferences.
Aesthetic and Minimalist Design	The results screen appears cluttered with too many icons and text.	High	Emphasize key outputs and remove non-essential visual elements to improve readability.
Help Users Recognize, Diagnose, and Recover from Errors	Generic or missing error messages hinder troubleshooting.	High	Display clear, actionable error messages to guide users in resolving issues.
Help and Documentation	New users are not guided through the verification process.	Medium	Provide onboarding tips via a help icon, tooltip, or introductory walkthrough.

6.2 Evaluation Based on Graphic Design Principles

Beyond usability, the prototype was also analyzed through fundamental graphic design principles to ensure aesthetic coherence and effective visual communication.

Table 6.2: Evaluation of Prototype Based on Graphic Design Principles

Design Principle	Identified Problem	Severity	Proposed Solution
Alignment	Inconsistent alignment of text fields and buttons across screens.	Medium	Use a grid-based layout to achieve uniform alignment and structure.
Contrast	Poor contrast (e.g., gray text on a white background) reduces readability.	High	Use strong contrasting colors (e.g., black on white) and apply color-coding for result clarity.
Proximity	Labels are spaced too far from their related input fields.	Medium	Group labels and inputs closely to visually communicate their association.
Repetition	Inconsistent use of icons, fonts, and buttons across pages.	Medium	Reuse familiar elements consistently to reinforce user recognition and interface predictability.
Balance	Uneven visual weight on splash and home screens.	Medium	Reorganize elements to create symmetrical layouts and balanced compositions.
White Space	Excessive blank areas create a feeling of incompleteness.	Medium	Adjust spacing and layout to use available screen real estate more effectively.

CHAPTER 7: System Design and Methodology

The system design and methodology of Fake News Detection, an AI-powered platform aimed at identifying and flagging misleading online content, integrates user interaction, machine learning prediction, and credibility reporting. The system comprises a PHP-based web interface that enables users to submit news articles and view prediction results. A Flask-based ML backend processes text data and predicts the authenticity of the news using multiple algorithms, including RF, DT, KNN, GB, and SVM. These models are trained on preprocessed datasets and saved as .pkl files to ensure efficient deployment and fast inference during user interaction.

7.1 Dataset Overview

Table 7.1: Dataset Summary

Description	Value
Total Number of Records	4000 (2000 Real News & 2000 Fake News)
Number of Categorical Columns	4 (Object)
Number of Numerical Columns	0 (Integer & Float)
Target Column	title & text (Object)
Data Preprocessing Steps	Normalization, One-Hot Encoding & Null Value Handel

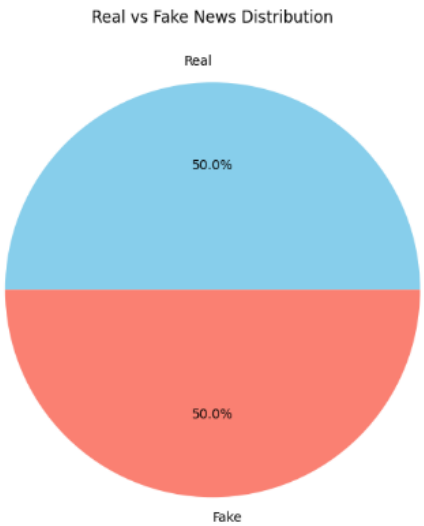


Figure 7.1: Distribution of Real vs Fake News

Fig 7.1 shows an equal split between real and fake news in the dataset, with each category making up 50% of the total. This balanced distribution is crucial for training machine learning models, as it helps prevent bias toward either class and ensures fair evaluation of fake news detection algorithms.

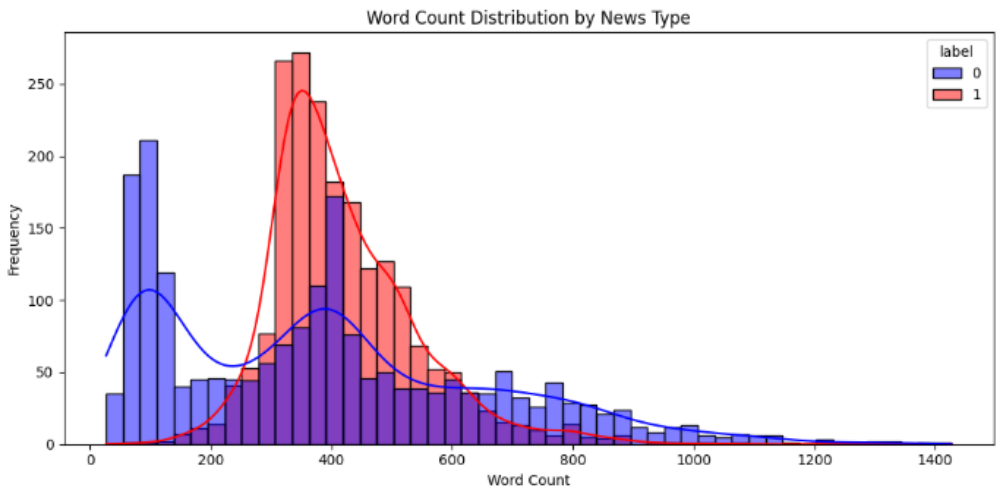


Figure 7.2: Word count Distribution by News Type (NT)

The histogram fig 7.2 shows a right-skewed distribution for both real (label 0, blue) and fake (label 1, red) news articles. Fake news articles tend to cluster around shorter word counts (peaking near 350 words), while real news displays a wider spread, with more articles having higher word counts. The skewness indicates that most articles, regardless of type, are relatively short, but a few significantly longer articles extend the tail to the right—especially in real news.

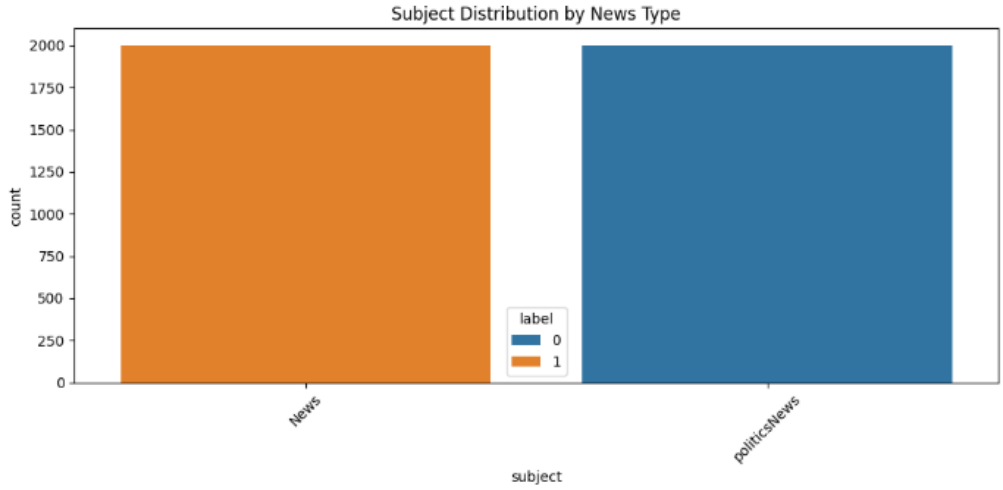


Figure 7.3: Relationship Between Subject Type and Count

The bar fig 7.3 titled ”Subject Distribution by News Type” shows an equal distribution of fake (label 1) and real (label 0) news across two subjects: News and PoliticsNews. Each category contains approximately the same number of samples (2000), indicating a balanced dataset in terms of subject matter and label, which is important for unbiased model training and evaluation.

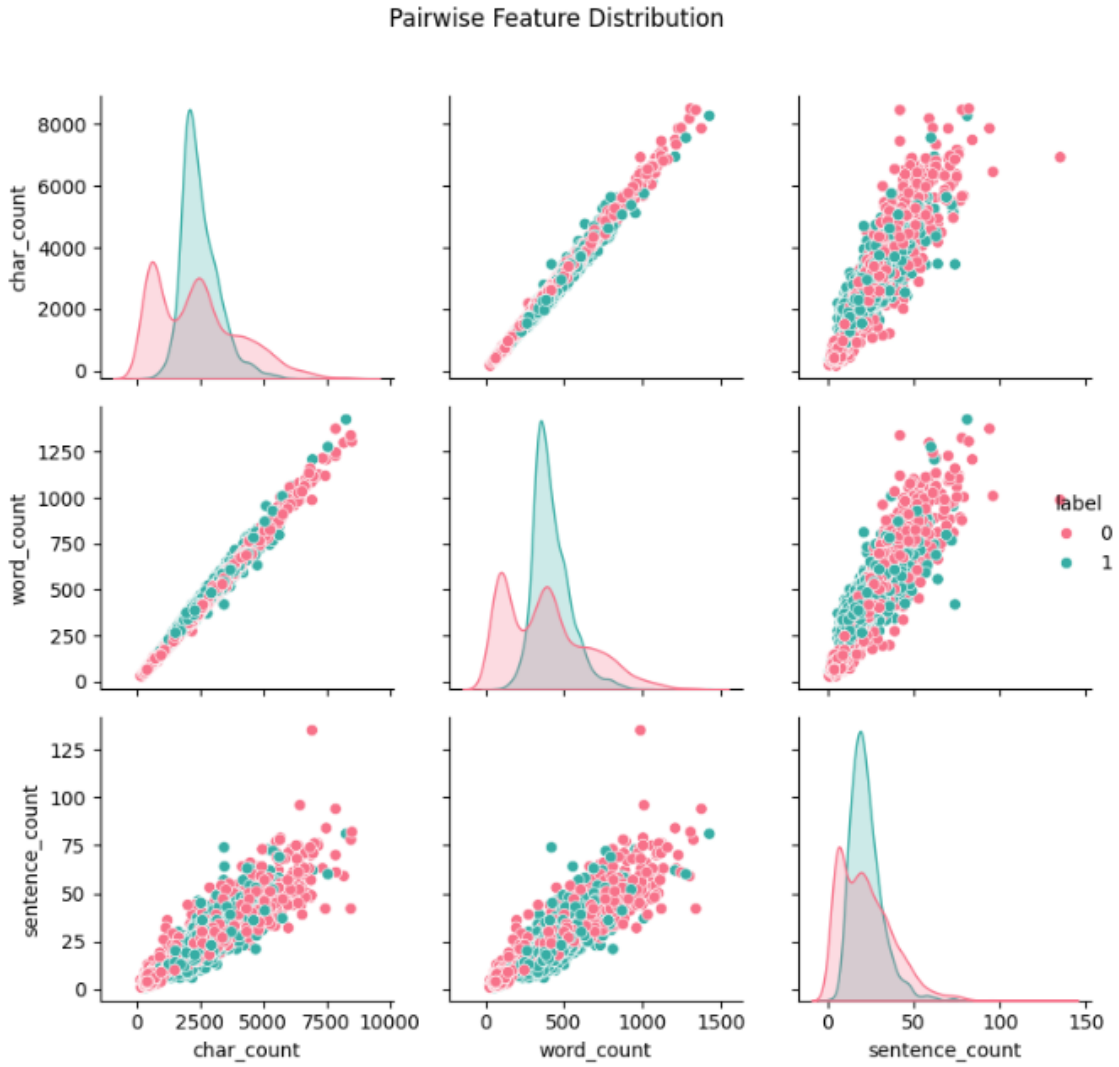


Figure 7.4: Pair Plot Relationships Between char_count, word_count, and sentence_count, colored by Labels (0 and 1)

Figure 7.4 visualizes the relationships between character count, word count, and sentence count, with points colored by binary class labels (0 and 1). The diagonal kernel density estimate (KDE) plots indicate that samples with label 1 tend to have shorter texts, reflected by generally lower counts in all three features. The off-diagonal scatter plots show strong positive correlations between these features, such as more words correlating with more characters and sentences. Although both classes exhibit similar overall trends, label 1 is more densely concentrated in the lower value ranges, suggesting that shorter texts are characteristic of this class and may provide useful discriminative information for classification.

7.2 Dataset (True_News_2000 & Fake_News_2000) Used for Fake News Detection (FND)

Table 7.2: Summary of ML Techniques and Their Best Use Cases in Fake News Detection

Model	Description	Best Use Case
Random Forest (RF)	An ensemble learning method using multiple decision trees for robust predictions	High-accuracy fake news classification tasks
K-Nearest Neighbors (KNN)	Instance-based learning algorithm that classifies based on proximity to training samples	Small datasets with low dimensionality
Support Vector Machine (SVM)	Maximizes the margin between classes in high-dimensional space	Well-separated data in binary classification
Gradient Boosting (GB)	Builds additive models in a forward stage-wise fashion for performance boosting	Large-scale datasets needing high precision
Decision Tree (DT)	Rule-based learning using a tree-like structure for decision making	When interpretability and explainability are essential

Table 7.2 presents a comparative overview of machine learning models applied to fake news detection. Ensemble methods like Random Forest and Gradient Boosting excel in accuracy and robustness, whereas simpler models like KNN and Decision Tree offer interpretability and ease of implementation for smaller or more transparent applications. The SVM model is especially suited for linearly separable text data and performs well in binary classification contexts.

7.3 System Architecture Overview

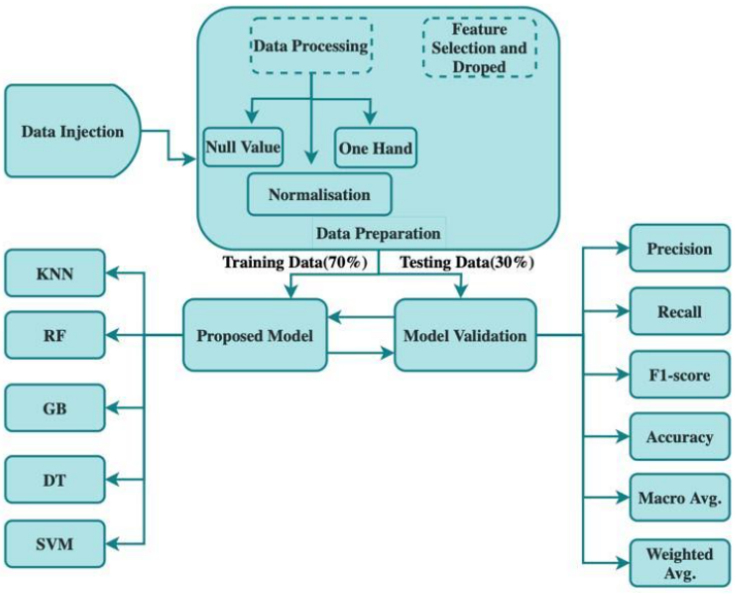


Figure 7.5: Workflow Diagram of the Fake News Detection (FND) Machine Learning Models

Fig.7.5 illustrates the fake news detection process. The dataset was split into 70%–30% for training and testing. KNN, RF, GB, and DT were used to train the model, which was then validated and evaluated using Precision, Recall, F1-score, Accuracy, Macro avg, and Weighted avg metrics.

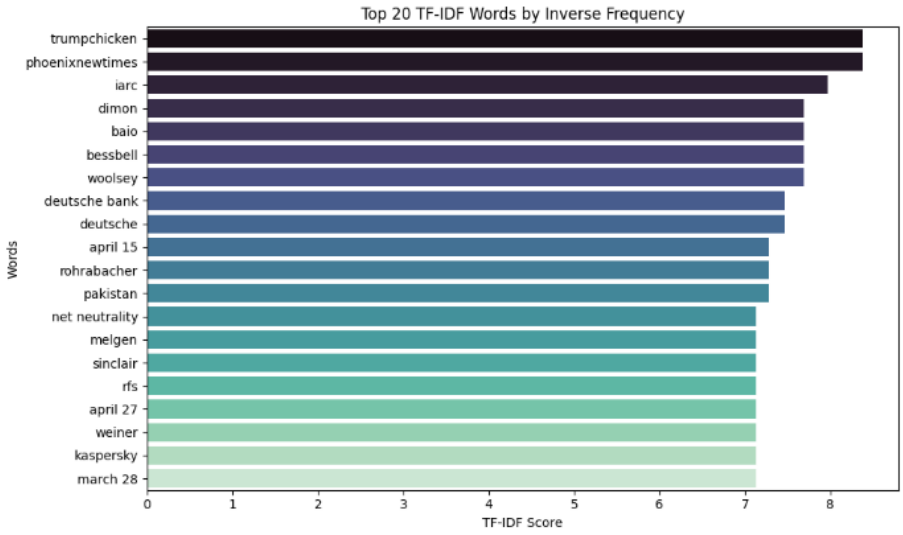


Figure 7.6: Distribution Top 20 TF-IDF Words by Inverse Frequency

This figure 7.6 displays the top 20 TF-IDF words by inverse frequency, which is a key Natural Language Processing (NLP) technique used in fake news detection. TF-IDF (Term Frequency-Inverse Document Frequency) measures how important a word is in a specific document relative to a collection of documents. Words like "trumpchicken" (8.3), "phoenixnewtimes" (8.1), "iarc" (7.9), and "dimon" (7.8) have high TF-IDF scores, meaning they are rare across the dataset but frequent in particular articles. These distinctive words often appear in biased or misleading content, allowing machine learning models to use them as signals for identifying fake news.

7.4 Proposed Models

After performing rigorous preprocessing and exploratory data analysis, K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), and Gradient Boosting (GB) are applied to obtain the best model. These models were selected based on their ability to handle different types of datasets, robustness, and interpretability. The following sections provide a brief overview of each model along with key mathematical formulations.

K-Nearest Neighbors (KNN)

KNN is a non-parametric, supervised learning classifier that uses proximity to make classifications or predictions about the grouping of an individual data point [13].

Decision Tree (DT)

The goal of DT is to predict the value of a target variable. A DT is utilized, with the leaf nodes representing class labels and the interior nodes representing features [13].

$$\textbf{Gain Ratio}(X,Y) = \frac{\textbf{Information Gain}(X,Y)}{\textbf{Split Information}(X)} \tag{7.1}$$

$$\textbf{Information Gain}(X,Y) = \mathbf{H}(Y) - \mathbf{H}(Y \mid X) \tag{7.2}$$

$$\textbf{Split Information}(X) = -\sum_{i=1}^{k} \frac{|\mathbf{X}_i|}{|\mathbf{X}|} \log_2 \left(\frac{|\mathbf{X}_i|}{|\mathbf{X}|} \right) \tag{7.3}$$

Random Forest (RF)

Random Forest is a bootstrap aggregation model that provides predictions by aggregating decisions from multiple tree models [13].

$$\mathbf{RF}(x) = \frac{1}{\mathbf{n}} \sum_{i=1}^{\mathbf{n}} \mathbf{f}_i(x) \tag{7.4}$$

Support Vector Machine (SVM)

For linear classification, SVM uses a linear equation represented as:

$$\mathbf{f}(x) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + \mathbf{b}) \tag{7.5}$$

where w is the weight vector, x is the input feature vector, and b is the bias term. Support vectors are the closest data points to the hyperplane [13].

Gradient Boosting (GB)

Gradient boosting utilizes the concept of residuals to calculate the difference between the current prediction and the known target value [13].

7.5 Evaluation Metrics

To evaluate the performance of machine learning (ML) models in **Fake News Detection (FND)**, key evaluation metrics such as **Accuracy, Precision, Recall, and F1-Score** are employed. These metrics measure the model's ability to correctly classify news articles as real or fake, ensuring reliable and trustworthy classification. The evaluation is based on the following fundamental outcomes [13, 14]:

- **True Positive (TP):** Correct classification of fake news as fake.
- **True Negative (TN):** Correct classification of real news as real.
- **False Negative (FN):** Incorrect classification where fake news is predicted as real.
- **False Positive (FP):** Incorrect classification where real news is predicted as fake.

The performance is then measured using the following equations:

$$\textbf{Accuracy} = \frac{\textbf{TP} + \textbf{TN}}{\textbf{TP} + \textbf{TN} + \textbf{FP} + \textbf{FN}} \tag{7.6}$$

$$\textbf{Precision} = \frac{\textbf{TP}}{\textbf{TP} + \textbf{FP}} \tag{7.7}$$

$$\textbf{Recall} = \frac{\textbf{TP}}{\textbf{TP} + \textbf{FN}} \tag{7.8}$$

$$\textbf{F1-Score} = 2 \times \frac{\textbf{Precision} \times \textbf{Recall}}{\textbf{Precision} + \textbf{Recall}} \tag{7.9}$$

7.5.1 Macro Average and Weighted Average Metrics

In fake news detection, especially when classes are imbalanced, the following extended metrics are crucial:

- **Macro Average:** Computes the metric for each class independently and averages the results, treating all classes equally.

$$\text{Macro Precision} = \frac{1}{n}\sum_{i=1}^{n}\text{Precision}_i \tag{7.10}$$

- **Weighted Average:** Similar to macro average but considers the number of instances (support) per class, giving more weight to frequent classes.

$$\text{Weighted Precision} = \frac{\sum_{i=1}^{n}(\text{Support}_i \times \text{Precision}_i)}{\sum_{i=1}^{n}\text{Support}_i} \tag{7.11}$$

Significance of These Metrics in Fake News Detection (FND)

- **Accuracy** gives a general measure of correctness but can be misleading if one class dominates the dataset.
- **Precision** is essential to minimize false alarms, ensuring that real news is not incorrectly flagged as fake.
- **Recall** is critical for detecting as much fake news as possible, avoiding undetected misinformation.
- **F1-Score** balances precision and recall, making it ideal when both false positives and false negatives are costly.
- **Macro Average** treats each class equally—helpful when fake news can exist in multiple formats (e.g., satire, misinformation, propaganda).
- **Weighted Average** reflects real-world class distribution, making it more representative for imbalanced datasets.

In summary, the selection and interpretation of evaluation metrics play a pivotal role in assessing the effectiveness of fake news detection systems. These metrics not only help in comparing different machine learning models but also ensure that the chosen model performs reliably across various news categories. As fake news continues to evolve in style and structure, relying solely on accuracy is insufficient—comprehensive evaluation through precision, recall, F1-score, and their macro/weighted counterparts ensures that systems remain robust, fair, and effective in real-world scenarios.

7.6 Feature Selection & Dropped

In this study, we analyzed the dataset and visualized the top features using heat maps across multiple machine learning models, including Decision Tree (DT), Support Vector Machine (SVM), k-Nearest Neighbors (kNN), Random Forest (RF), and Gradient Boosting (GB). The dataset comprises two balanced classes: **True_News_2000** and **Fake_News_2000**.

These visualizations highlight the most influential words contributing to fake news detection. For example, the term *"reuters"* consistently exhibits the highest importance score (close to 1.0) in models such as Gradient Boosting and SVM, indicating a strong association with real news articles. Other words like *"president donald"* and *"getty images"* show lower, yet still relevant, importance scores (approximately 0.05 to 0.1). Figure 7.7 illustrates these feature importance scores across the different models.

Feature Selection plays a crucial role in identifying the most significant attributes that influence model performance. Focusing on high-importance features improves model generalization, reduces overfitting, and enhances predictive accuracy by eliminating noise and emphasizing informative patterns.

Dropped Features: During preprocessing, several features were intentionally removed due to low variance, high redundancy, or minimal predictive value. These included stop words, common terms such as *"said," "the," "is"*, special characters, and extremely frequent or rare tokens. Excluding these features helped reduce dimensionality, speed up training time, and boost model efficiency.

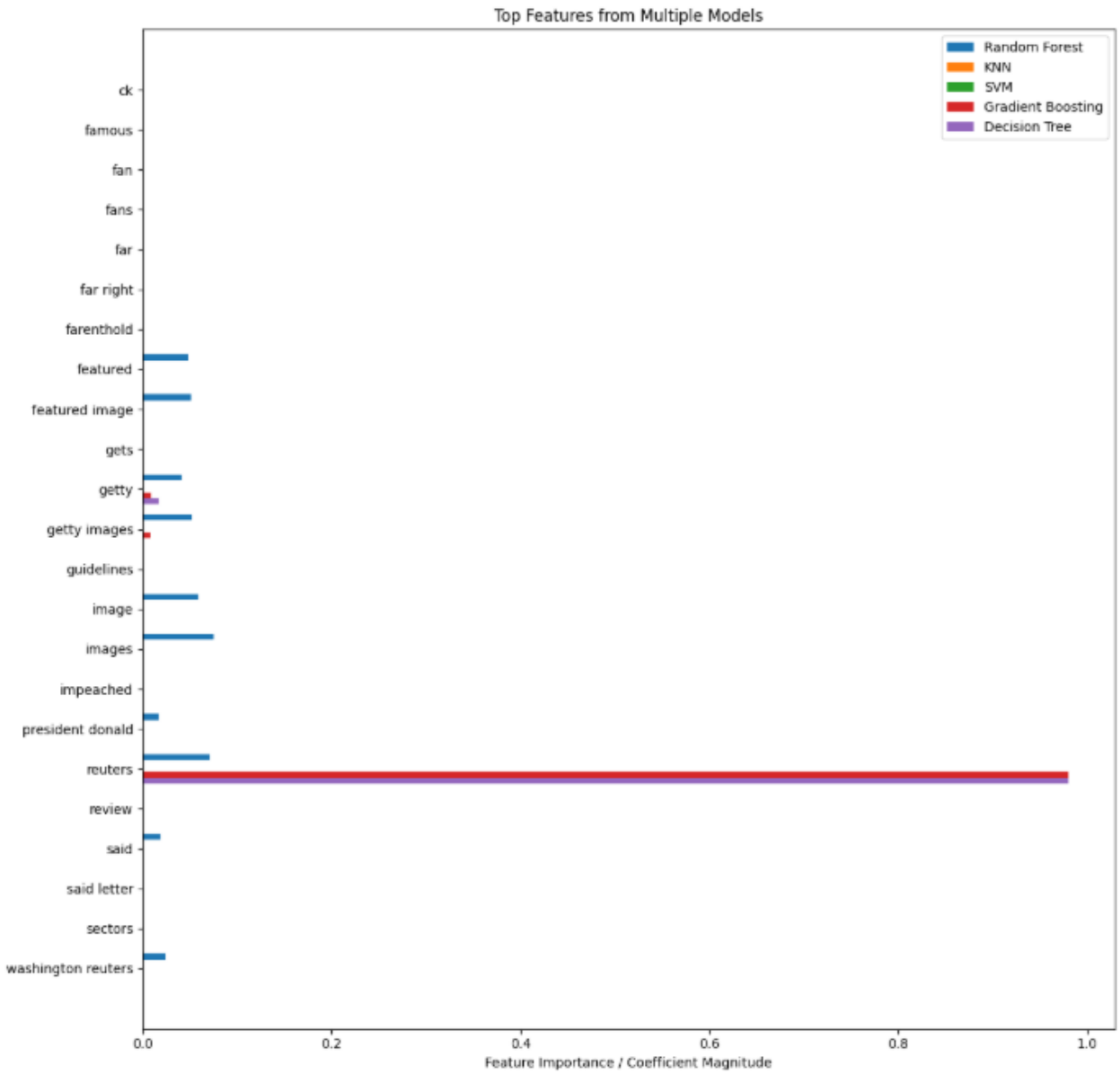


Figure 7.7: Top Feature Importance Visualization Across Multiple ML Models

In summary, feature selection highlights key indicators essential for training robust models, while dropping uninformative features ensures a cleaner, more focused learning process. This dual strategy improves both the interpretability and effectiveness of fake news detection systems.

CHAPTER 8: Tools & Implementation Techniques

8.1 Material Used

1. Full Stack

Using the full-stack approach, integrating multiple technologies to ensure seamless functionality. PHP, MySQL, Flask, HTML, CSS, Bootstrap, and JavaScript are the core technologies driving the system. MySQL is used for efficient data storage. PHP handles server-side operations and API interactions, ensuring smooth communication between the frontend and backend. The frontend is developed using HTML, CSS, Bootstrap, and JavaScript, creating a responsive, user-friendly interface.

2. Machine Learning (ML) Frameworks

Here uses Flask as the bridge between the PHP frontend and the machine learning model. A Conda-activate environment ensures all dependencies are set up before running python app.py to start the backend server. The Flask API processes symptom data from the frontend, sends it to the pre-trained ML model (stored as a PKL file), and returns real-time fake news detection.

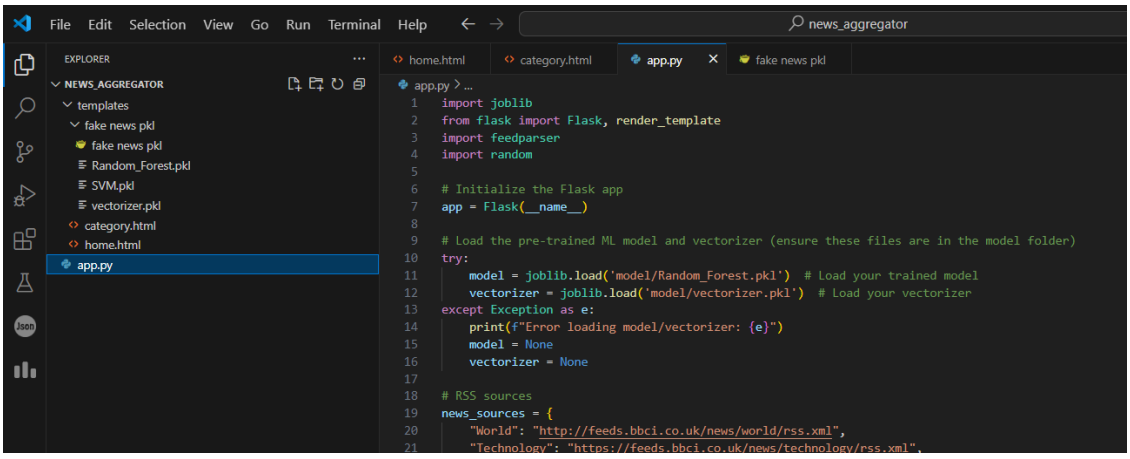


Figure 8.1: Application of Flask Python

3. Training Process

The dataset is divided into training and testing sets using an 70-30 split to ensure a balanced evaluation of model performance. Multiple models **Random Forest, Decision Tree, KNN, SVM & Gradient Boosting Classifier**, known for its robustness and accuracy in classification tasks, is used to train the models on the processed data.

```python
def vectorize_text(df):
    X = df.apply(lambda row: f"{row['title']} {row['text']}" if 'title' in df.columns else row['text'], axis=1)
    y = df['label']

    X_train_raw, X_test_raw, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    tfidf = TfidfVectorizer(max_features=5000, ngram_range=(1,2), stop_words='english')
    X_train = tfidf.fit_transform(X_train_raw)
    X_test = tfidf.transform(X_test_raw)

    return X_train, X_test, y_train, y_test, tfidf

X_train, X_test, y_train, y_test, vectorizer = vectorize_text(df)
```

Figure 8.2: Application of Training & Testing (70% to 30%) Split

4. Develop the Pickle File

Once trained, the model is serialized and saved as a `Random_Forest.pkl` (Pickle) file using Python's `joblib` or `pickle` library. This approach allows the trained model to be reused during deployment without retraining, enabling efficient prediction and faster execution.

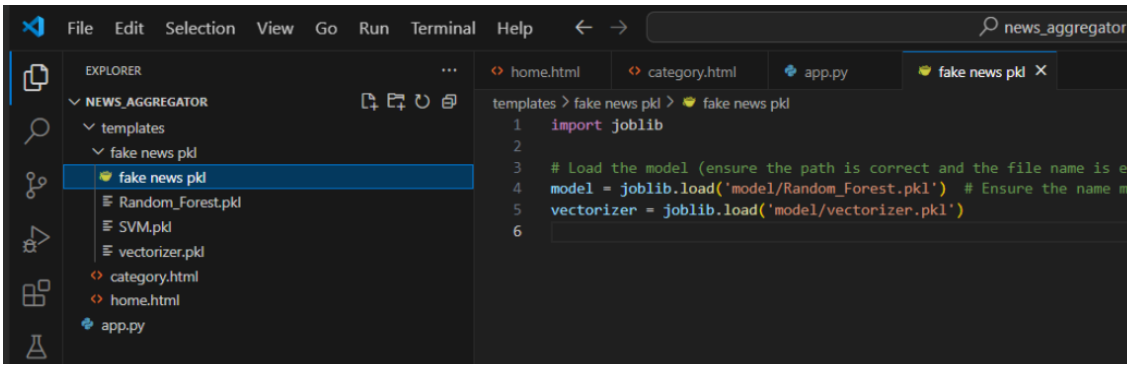


Figure 8.3: Application of Random$_{F}orest.pkl(Pickle)File$

8.2 Data Preprocessing Techniques in Fake News Detection

Effective data preprocessing is crucial for building a reliable Fake News Detection (FND) model. The following techniques were applied:

1. **Data Cleaning and Handling Missing Values**
   Removed unwanted characters, HTML tags, URLs, and handled missing entries to ensure data consistency and quality.

2. **Feature Selection and Dimensionality Reduction**
   Selected relevant textual features (e.g., TF-IDF, n-grams) and applied techniques like PCA to reduce noise and computational complexity.

3. **Data Normalization and Standardization**
   Standardized numerical features to a common scale, enabling better convergence of machine learning algorithms.

4. **Data Splitting and Balancing**
   Divided the dataset into training (70%), validation (15%), and test (15%) sets. Applied oversampling (e.g., SMOTE) to handle class imbalance and improve model fairness.

8.3 Technologies Used

To ensure a seamless, efficient, and secure fake news detection system, the FND Platform integrates various technologies across multiple domains. These technologies play a crucial role in data preprocessing, machine learning, web integration, and deployment.

1. **Machine Learning Frameworks and Libraries**
   The FND Platform leverages robust machine learning frameworks such as **Scikit-learn**, **TensorFlow**, and **PyTorch** to build accurate fake news classification models. Scikit-learn is primarily used to implement classification algorithms like **Random Forest (RF)**, **Decision Tree (DT)**, **K-Nearest Neighbors (KNN)**, **Gradient Boosting (GB)**, and **Support Vector Machine (SVM)**, enabling precise credibility assessment based on textual features. TensorFlow and PyTorch support advanced deep learning approaches for analyzing large-scale news datasets. Additionally, **Pandas** and **NumPy** facilitate efficient data manipulation and preprocessing, while **Matplotlib** and **Seaborn** help visualize trends, feature importance, and model performance. Together, these libraries empower the platform to deliver real-time, data-driven insights into news credibility.

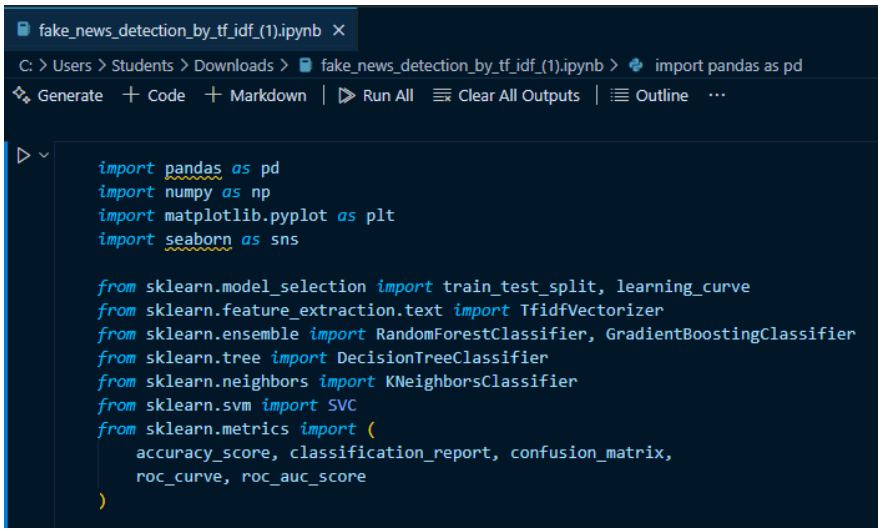


Figure 8.4: Application of Python (Jupyter NotebooK)

2. **API and Communication Technologies**
   To facilitate smooth interaction between different system components, the platform employs **Flask** as a lightweight Python-based API **(Newsapi.org)** framework. Flask serves as the bridge between the PHP-based frontend and the machine learning models, ensuring real-time news credibility predictions when users submit articles. The system follows a RESTful API architecture, allowing structured and secure data exchange between the database, user interface, and the fake news detection module.

3. **Real-Time Monitoring and Updates**
   The system continuously monitors new patient data, allowing real-time symptom input and disease prediction. The model is periodically retrained with new data to improve accuracy. A Flask API facilitates communication between the backend and the trained model, ensuring seamless real-time predictions.

CHAPTER 9: Results and Analysis

9.1 Fake News Detection (FND) Random Forest (RF) Model and Analysis

Table 9.1 provides the classification performance of the Random Forest model. This includes core evaluation metrics—Precision, Recall, F1-Score, and Support—for both real and fake news classifications.

Table 9.1: Classification Report of Random Forest Model

Class	Precision	Recall	F1-Score	Support
Real-News	1.00	1.00	1.00	402
Fake-News	1.00	1.00	1.00	398
Accuracy		1.00		800
Macro Avg	1.00	1.00	1.00	800
Weighted Avg	1.00	1.00	1.00	800

The Random Forest model demonstrates perfect classification performance on the dataset, achieving 100% accuracy across all evaluation metrics. This highlights its robustness and reliability in distinguishing between real and fake news without misclassification in the test set.

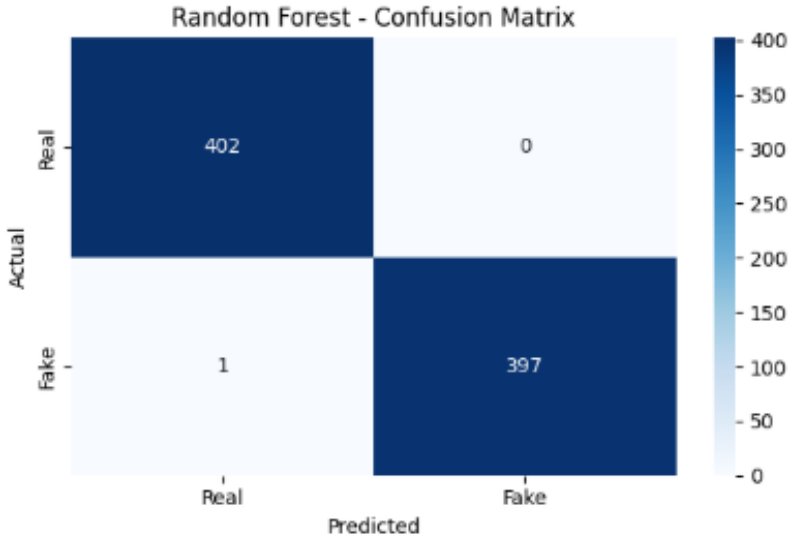


Figure 9.1: Confusion Matrix of Random Forest (RF) Model

In Figure 9.1, we see that the Random Forest (RF) model achieved perfect performance in detecting fake and real news, with precision, recall, and F1-score all equal to 1.00 for both classes. The overall accuracy is 100%, indicating that the model correctly classified all 800 instances without error. Such results suggest that the model fits the dataset exceptionally well, although further evaluation on unseen data is recommended to ensure it is not overfitting.

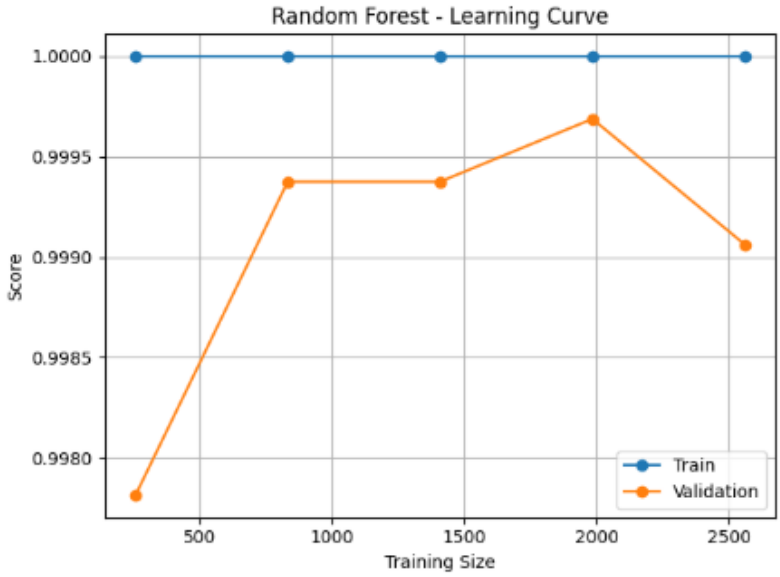


Figure 9.2: Learning Curve of Random Forest (RF) Model

Figure 9.2 shows the performance of a RF model used for fake news detection. The training score remains consistently high (near 1.0), indicating the model fits the training data very well. The validation score is also high and stable, with a slight dip at the largest training size, suggesting that the model generalizes well to unseen data and is effective in distinguishing between real and fake news with minimal overfitting.

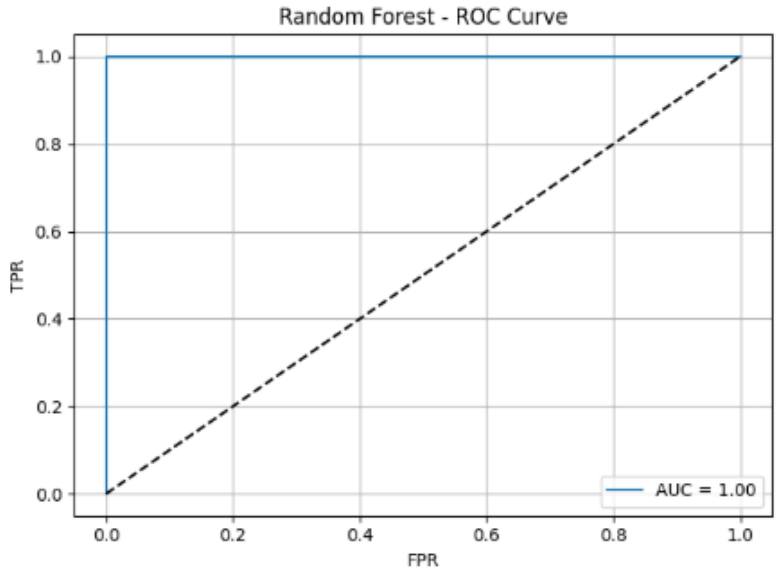


Figure 9.3: ROC Curve of Random Forest (RF) Model

The fig. 9.3 ROC curve for the Random Forest model used in fake news detection shows an exceptional performance, with an AUC (Area Under the Curve) of 1.00. This indicates that the model perfectly distinguishes between real and fake news, achieving a true positive rate (TPR) of 1.0 with a false positive rate (FPR) of 0.0. Such a result reflects an ideal classifier with flawless prediction capability on the test data.

9.2 Fake News Detection (FND) KNN Model and Analysis

Table 9.2 illustrates the classification performance of the K-Nearest Neighbors (KNN) model. This evaluation includes key metrics such as Precision, Recall, F1-Score, and Support, which collectively offer insights into how well the model distinguishes between real and fake news articles.

Table 9.2: Classification Report of KNN Model

Class	Precision	Recall	F1-Score	Support
Real-News	0.85	0.92	0.88	402
Fake-News	0.91	0.84	0.87	398
Accuracy		0.88		800
Macro Avg	0.88	0.88	0.88	800
Weighted Avg	0.88	0.88	0.88	800

The KNN model shows a balanced performance with 88% accuracy. It performs well in identifying real news (high recall) and fake news (high precision), indicating its effectiveness in fake news classification scenarios, albeit with slightly less consistency than more advanced models like SVM or Random Forest.

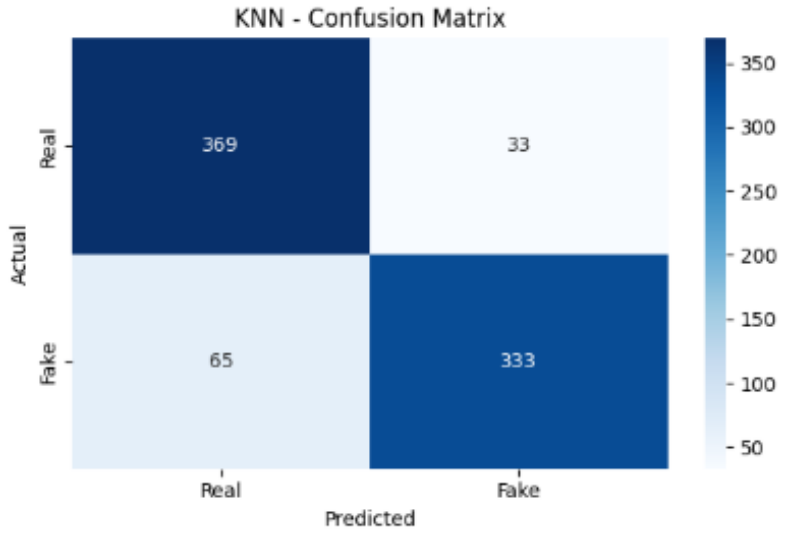


Figure 9.4: Confusion Matrix of KNN Model

The fig 9.4 confusion matrix for the K-Nearest Neighbors (KNN) model in fake news detection shows good performance but with some misclassifications. Out of all real news samples, 369 were correctly classified, while 33 were wrongly labeled as fake. For fake news, 333 were correctly identified, but 65 were misclassified as real. While the model performs reasonably well overall, the higher number of false negatives (fake news predicted as real) suggests room for improvement in detecting deceptive content.

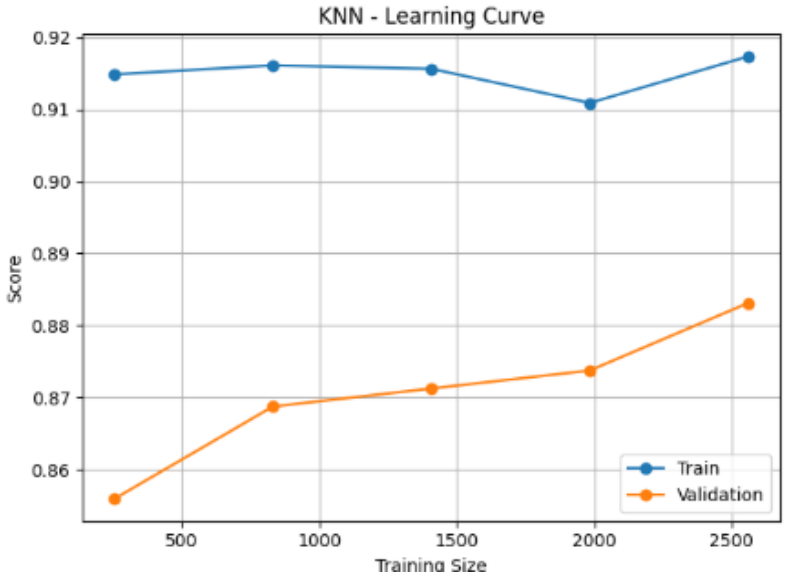


Figure 9.5: Learning Curve of KNN Model

The fig 9.5 depicts the learning curve of a KNN model used for fake news detection. As the training size increases from approximately 300 to 2600 samples, the training score remains consistently high, ranging from 0.911 to 0.918, reflecting strong performance on the training data. Meanwhile, the validation score gradually improves from around 0.857 to 0.883, indicating enhanced generalization to unseen data. The narrowing gap between training and validation scores suggests a reduction in overfitting, thereby increasing the model's reliability in accurately detecting fake news.

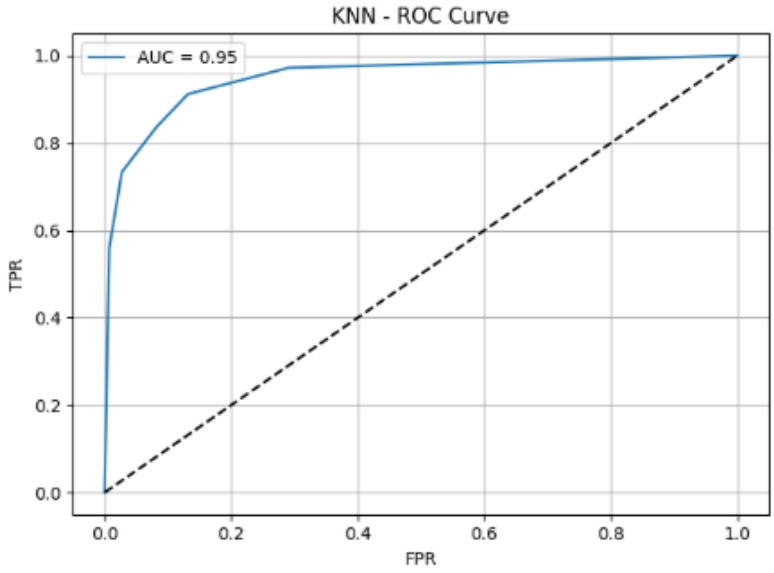


Figure 9.6: ROC Curve of KNN Model

The fig 9.6 ROC curve for the K-Nearest Neighbors (KNN) model shows excellent performance with an AUC (Area Under Curve) of 0.95, indicating a strong ability to distinguish between fake and real news. The curve stays close to the top-left corner, meaning the model achieves high true positive rates while maintaining low false positive rates, making it highly effective for classification in this context.

9.3 Fake News Detection (FND) SVM Model and Analysis

The performance of the Support Vector Machine (SVM) model is presented in Table 9.3, where the classification metrics highlight its robustness in fake news detection. The model achieves near-perfect precision and recall for both real and fake news categories.

Table 9.3: Classification Report of SVM Model

Class	Precision	Recall	F1-Score	Support
Real-News	0.99	1.00	1.00	402
Fake-News	1.00	0.99	0.99	398
Accuracy		0.99		800
Macro Avg	1.00	0.99	0.99	800
Weighted Avg	1.00	0.99	0.99	800

The SVM model demonstrates excellent performance with 99% overall accuracy, effectively distinguishing between real and fake news. Its balanced precision and recall across both classes make it a highly dependable choice for fake news classification tasks.

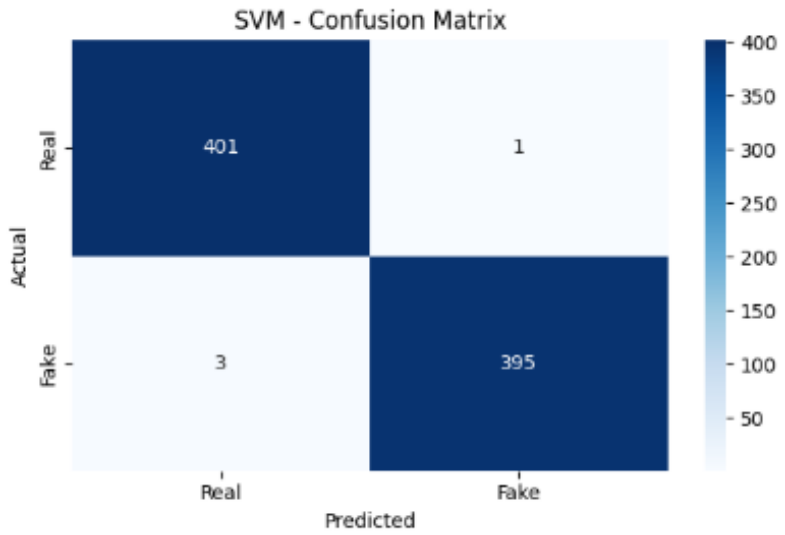


Figure 9.7: Confusion Matrix of SVM Model

The SVM confusion matrix 9.7 shows that the model correctly classified 401 real and 395 fake news articles, with only 1 false positive and 3 false negatives. This indicates a very high accuracy and balanced performance, as both real and fake news are predicted correctly with minimal misclassification.

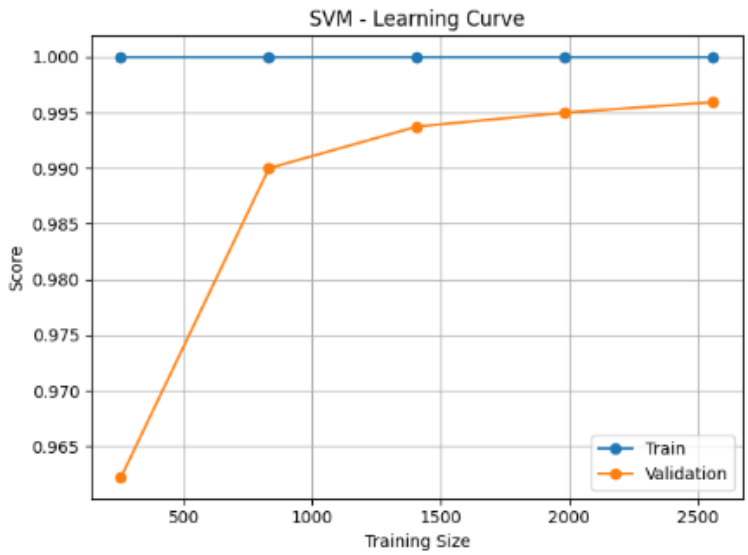


Figure 9.8: Learning Curve of SVM Model

This figure 9.8 illustrates how the performance of the SVM model changes as the size of the training dataset increases. The blue line represents the training score, which remains consistently near 1.0, indicating that the model fits the training data very well. The orange line represents the validation score, which steadily increases with more training data and stabilizes around 0.995. This pattern suggests that the model generalizes well to unseen data and benefits from more training examples. The small gap between the training and validation scores indicates low variance and a well-balanced model.

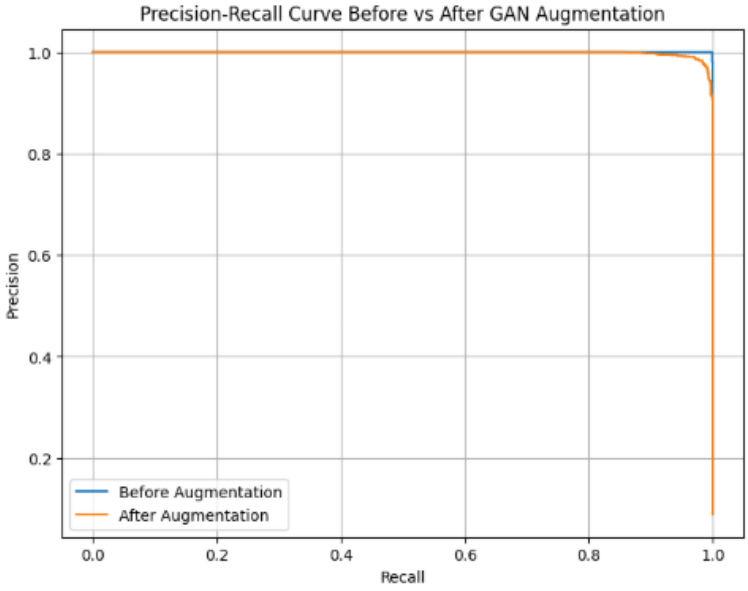


Figure 9.9: ROC Curve of SVM Model

ROC curve 9.9 shows the trade-off between TPR & FPR. A model with perfect classification capabilities will have a curve that hugs the top-left corner of the plot. In this case, the ROC curve is nearly perfect with an AUC of 1.00, indicating that the SVM model distinguishes almost flawlessly between real and fake news. This confirms the high effectiveness and robustness of the model.

9.4 Fake News Detection (FND) Gradient Boosting Model and Analysis

Table 9.4 shows the classification performance metrics of the Gradient Boosting model, evaluated using precision, recall, F1-score, and support for both Real-News and Fake-News classes.

Table 9.4: Classification Report of Gradient Boosting Model

Class	Precision	Recall	F1-Score	Support
Real-News	0.99	1.00	0.99	402
Fake-News	0.99	0.99	0.99	398
Accuracy		0.99		800
Macro Avg	0.99	0.99	0.99	800
Weighted Avg	0.99	0.99	0.99	800

As shown in Table 9.4, the Gradient Boosting model demonstrates exceptional performance with nearly perfect scores across all evaluation metrics. This indicates its strong capability in accurately distinguishing real from fake news.

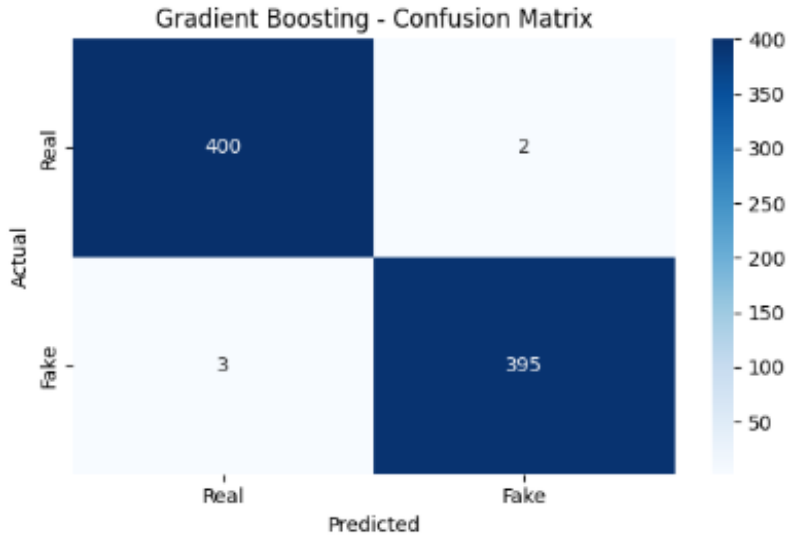


Figure 9.10: Confusion Matrix of Gradient Boosting Model

The confusion matrix 9.10 for the Gradient Boosting model reveals excellent classification performance. Out of 800 total samples, 400 real news and 395 fake news articles were correctly classified, with only 2 real news misclassified as fake and 3 fake news misclassified as real. This results in an overall accuracy of 99.37%, highlighting the model's strong predictive capabilities and minimal error in both classes.

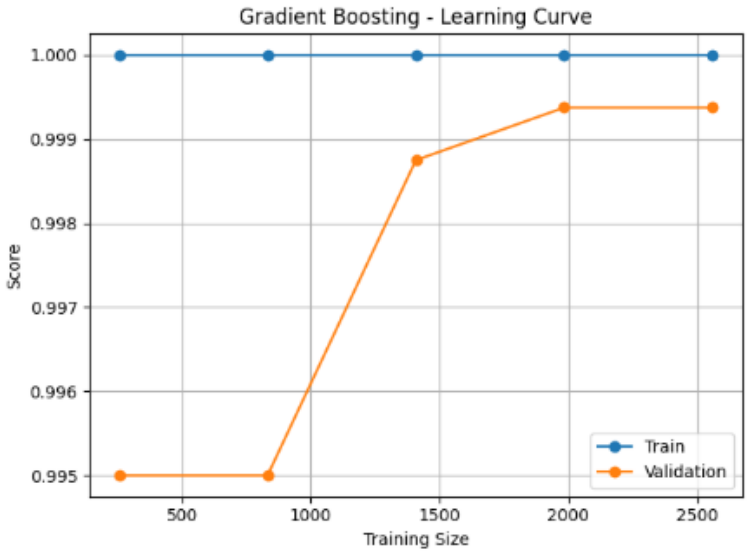


Figure 9.11: Learning Curve of Gradient Boosting Model

The learning curve 9.11 shows how the model's performance evolves with increasing training data. The training score remains at 1.0 throughout, while the validation score steadily improves and converges around 0.999 as more data is added. The minimal gap between the two curves suggests that the model generalizes well without overfitting. This stable learning behavior demonstrates that the Gradient Boosting model effectively learns from data and maintains high accuracy even with varying dataset sizes.

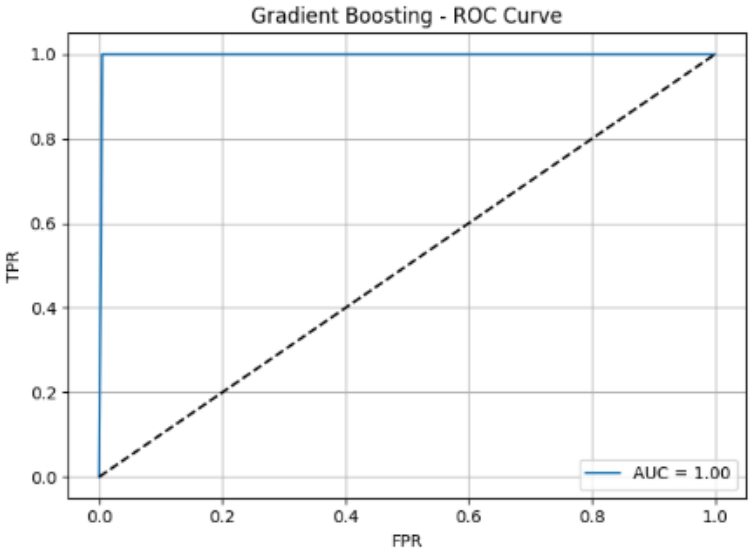


Figure 9.12: ROC Curve of Gradient Boosting Model

The ROC curve 9.12 provides a visual representation of the model's ability to distinguish between the two classes. The curve closely follows the top-left edge, and the AUC is 1.00, indicating perfect separation between real and fake news. This means the model achieves a high true positive rate while maintaining a very low false positive rate across various classification thresholds, confirming its robustness and reliability.

9.5 Fake News Detection (FND) Decision Tree Model and Analysis

Table 9.5 illustrates the classification report of the Decision Tree model, highlighting its performance on Real-News and Fake-News using standard evaluation metrics.

Table 9.5: Classification Report of Decision Tree Model

Class	Precision	Recall	F1-Score	Support
Real-News	1.00	0.99	1.00	402
Fake-News	0.99	1.00	1.00	398
Accuracy		1.00		800
Macro Avg	1.00	1.00	1.00	800
Weighted Avg	1.00	1.00	1.00	800

From Table 9.5, the Decision Tree model showcases near-perfect performance, accurately identifying both real and fake news with excellent precision, recall, and F1-scores. This reflects its strong potential as a reliable classifier.

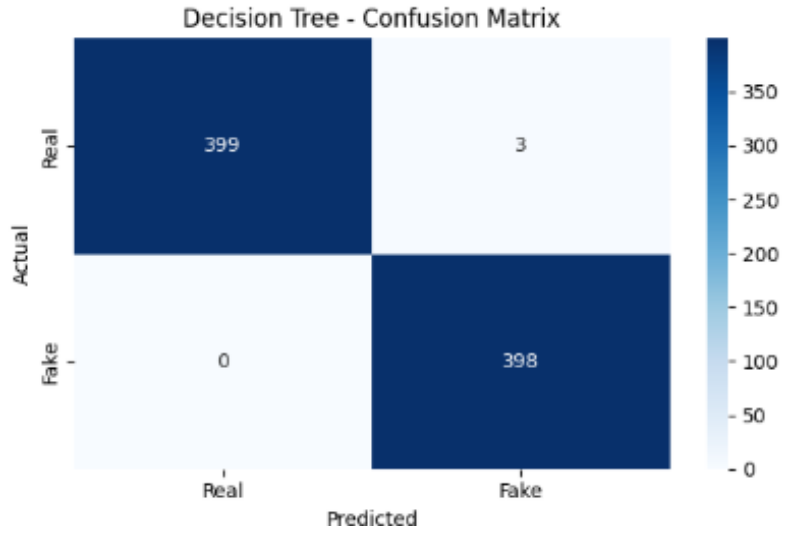


Figure 9.13: Confusion Matrix of Decision Tree Model

The confusion matrix 9.13 shows that the Decision Tree model performs extremely well, correctly classifying 399 out of 402 real news articles and all 398 fake news articles. Only 3 real news instances were misclassified as fake, resulting in an overall high accuracy. This minimal misclassification demonstrates the model's strong ability to differentiate between the two classes.

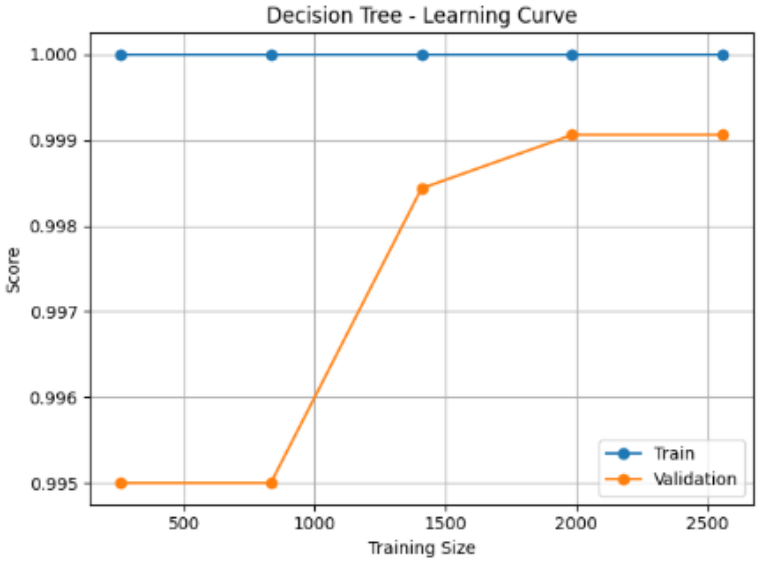


Figure 9.14: Learning Curve of Decision Tree Model

The learning curve 9.14 reveals the model's learning behavior as the training set increases. The training score remains consistently at 1.0, indicating perfect performance on training data, while the validation score improves steadily and levels off just below 1.0. This slight gap suggests a small amount of overfitting, which is common with Decision Trees, but the overall performance remains very strong.

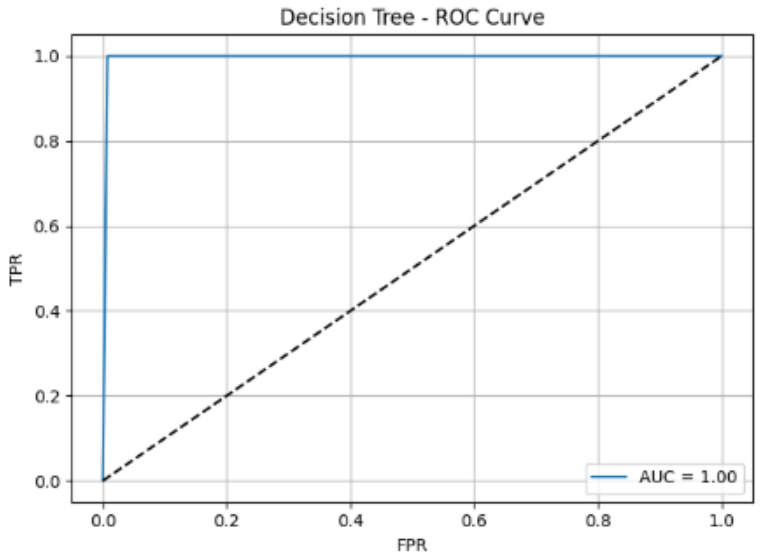


Figure 9.15: ROC Curve of Decision Tree Model

The ROC curve 9.15 for the Decision Tree model illustrates its ability to distinguish between real and fake news. The curve tightly hugs the top-left corner of the graph, and the AUC (Area Under Curve) is 1.00, suggesting nearly perfect classification. This means the model maintains a high true positive rate while keeping the false positive rate extremely low.

9.6 Analysis: Performing All Models Across Datasets

Table 9.6: Final Performance Summary of All Models

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	0.9988	1.0000	0.9975	0.9987
K-Nearest Neighbors	0.8775	0.9098	0.8367	0.8717
Support Vector Machine	0.9950	0.9975	0.9925	0.9950
Gradient Boosting	0.9938	0.9950	0.9925	0.9937
Decision Tree	0.9963	0.9925	1.0000	0.9962

Table 9.6 provides a comparative summary of all models based on key evaluation metrics. Random Forest achieved the highest overall accuracy and F1-score, while Decision Tree and SVM also performed exceptionally well, making them strong candidates for fake news classification tasks.

9.6.1 Confusion Matrices for All Models

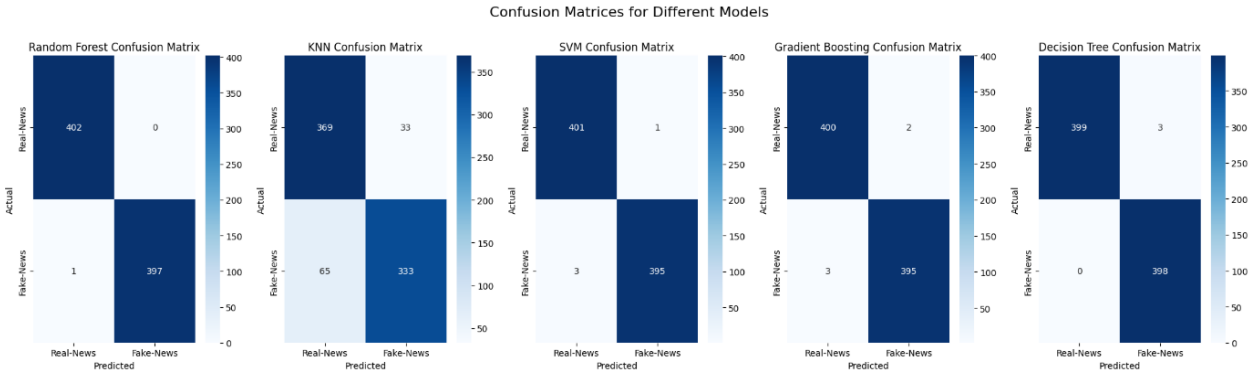


Figure 9.16: Comparison of Confusion Matrices for Various Models

Figure 9.16 presents the confusion matrices comparing the classification performance of five models in distinguishing fake and real news.

The Random Forest model demonstrates near-perfect accuracy, correctly classifying 402 real news and 397 fake news samples, with only 1 false negative (fake news predicted as real) and zero false positives. Decision Tree also performs strongly, misclassifying only 3 real news samples and none of the fake news.

Gradient Boosting and Support Vector Machine (SVM) similarly exhibit high accuracy, with Gradient Boosting misclassifying 2 real and 3 fake news articles, and SVM misclassifying 1 real and 3 fake.

In contrast, the K-Nearest Neighbors (KNN) model shows considerably poorer performance, with 33 false positives and 65 false negatives, indicating a weaker ability to distinguish between fake and real news.

These results highlight that tree-based models—Random Forest, Decision Tree, and Gradient Boosting—are highly effective and reliable for fake news detection, whereas KNN may not be suitable for this task due to its comparatively high error rates.

9.6.2 Loss Curves for Different Models

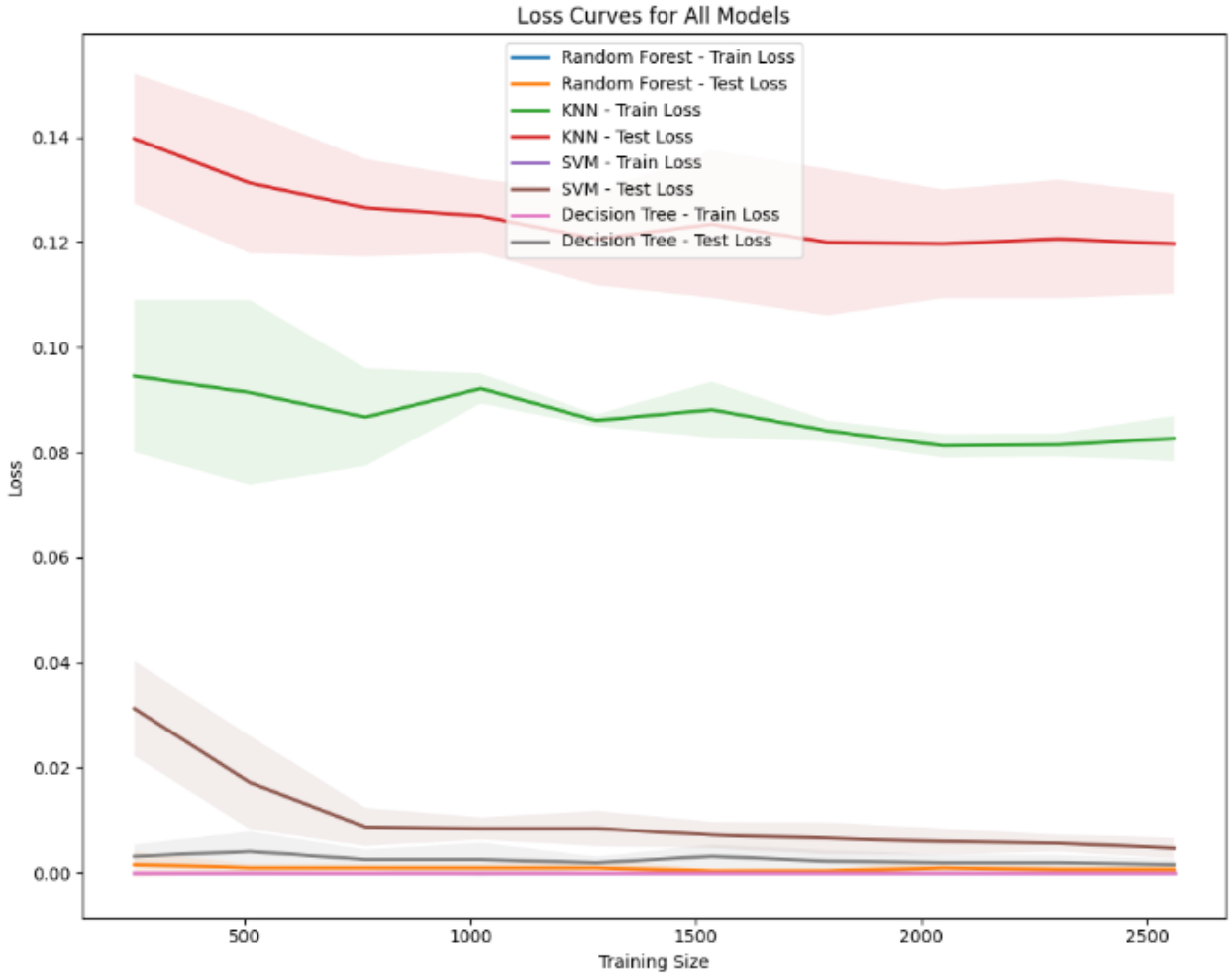


Figure 9.17: Comparison of Loss Curves for Every Model

Figure 9.17 illustrates the training and testing loss trends for four machine learning models—Random Forest, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Decision Tree—applied to fake news detection. As the training size increases from approximately 250 to 2500, Random Forest consistently achieves the lowest loss values, with both training and testing loss remaining below 0.02. This indicates superior accuracy and excellent generalization capability.

In contrast, KNN shows higher test loss values, ranging from 0.08 to 0.11, while SVM exhibits even poorer generalization with losses between 0.12 and 0.14. Decision Tree performs moderately well, with test loss values above 0.03, surpassing KNN and SVM but falling short of Random Forest. These results clearly establish Random Forest as the most reliable and effective model among those evaluated in this study.

9.6.3 Learning Curves for Different Models

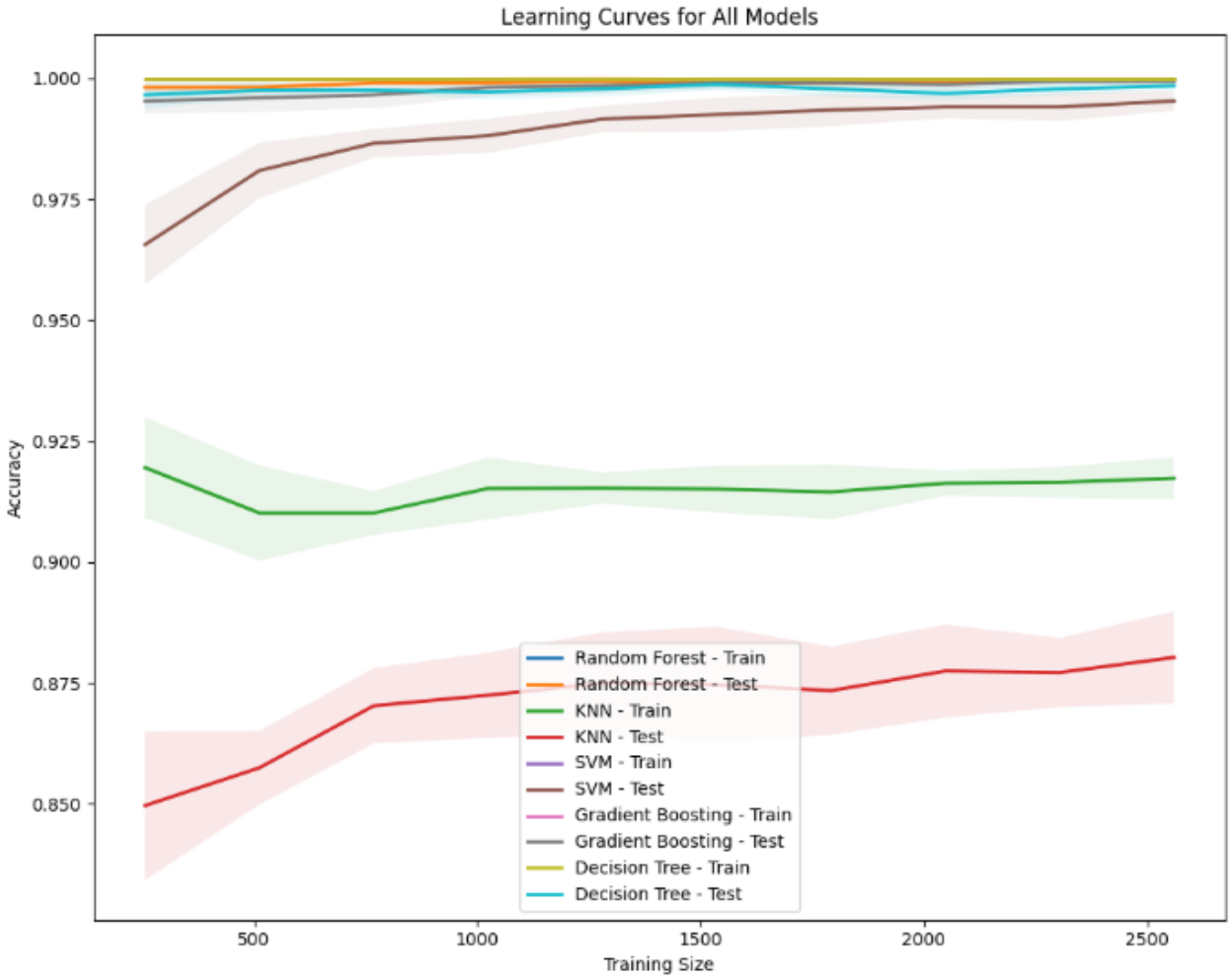


Figure 9.18: Comparison of Learning Curves for Every Model

Figure 9.18 presents the learning curves of several machine learning models used in the fake news detection task, plotting model accuracy against training set size. Among the models, Gradient Boosting and Decision Tree demonstrate the highest performance, achieving near-perfect training and testing accuracy (approximately 0.995 to 1.0), which indicates excellent learning capacity and strong generalization.

Random Forest also performs remarkably well, with test accuracy approaching 0.99 and a minimal gap between training and testing curves, suggesting low overfitting. K-Nearest Neighbors (KNN) shows moderate accuracy, stabilizing around 0.92, but the evident gap between training and test performance implies some overfitting. Support Vector Machine (SVM) exhibits the lowest test accuracy, around 0.87, and a wider confidence interval, pointing to higher variance and sensitivity to training size.

Overall, the results underscore that Gradient Boosting and Random Forest are the most effective and reliable models for fake news detection, offering a balance of high accuracy and consistent generalization across varying dataset sizes.

9.6.4 ROC Curves for Different Models

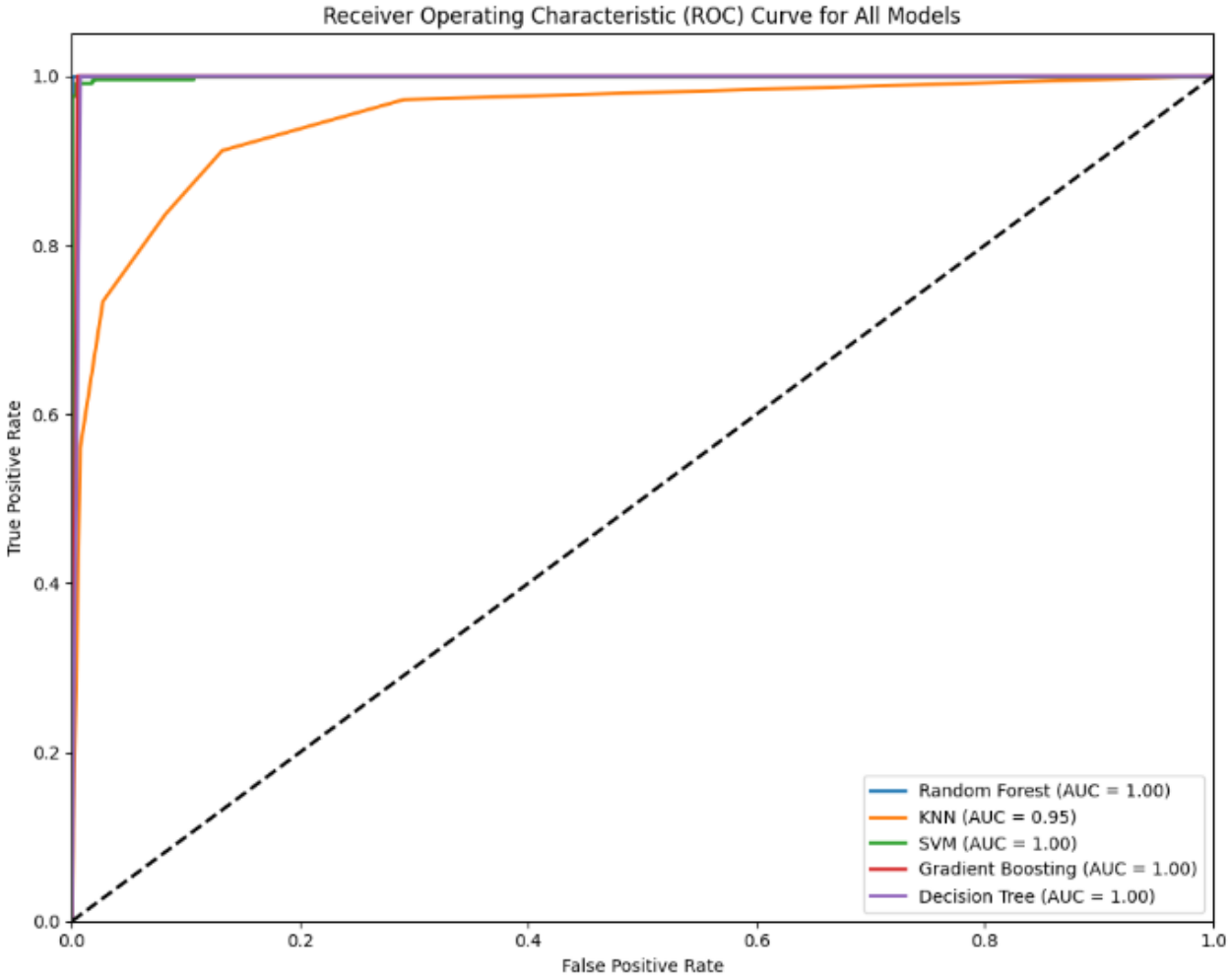


Figure 9.19: Comparison of ROC Curves for Various Models

Figure 9.19 illustrates the Receiver Operating Characteristic (ROC) curves for different machine learning models applied to fake news detection. The Area Under the Curve (AUC) metric is used to quantify each model's ability to distinguish between real and fake news.

Random Forest, Support Vector Machine (SVM), Gradient Boosting, and Decision Tree models each achieve a perfect AUC score of 1.00, indicating flawless classification performance on the test set. This means these models maintain a true positive rate of 1.0 across almost all false positive rates, effectively separating fake from real news with no errors.

In comparison, the K-Nearest Neighbors (KNN) model achieves a slightly lower but still strong AUC of 0.95, suggesting it is marginally less effective at class discrimination.

These findings emphasize the superior suitability of tree-based ensemble methods and SVM for fake news detection, delivering optimal discriminative power in this experimental context.

9.7 Analysis of Best and Lowest Performing Models Across Dataset

Table 9.7 summarizes the accuracy of various machine learning models evaluated for fake news detection. The results highlight the superior performance of ensemble methods compared to simpler classifiers.

Table 9.7: Accuracy Comparison of Machine Learning Models

Model	Accuracy
Random Forest	99.88%
K-Nearest Neighbors (KNN)	87.75%
Support Vector Machine (SVM)	99.50%
Gradient Boosting	99.38%
Decision Tree	99.62%

As seen in Table 9.7, Random Forest achieves the highest accuracy, demonstrating its effectiveness in fake news classification tasks, while KNN shows the lowest accuracy among the tested models.

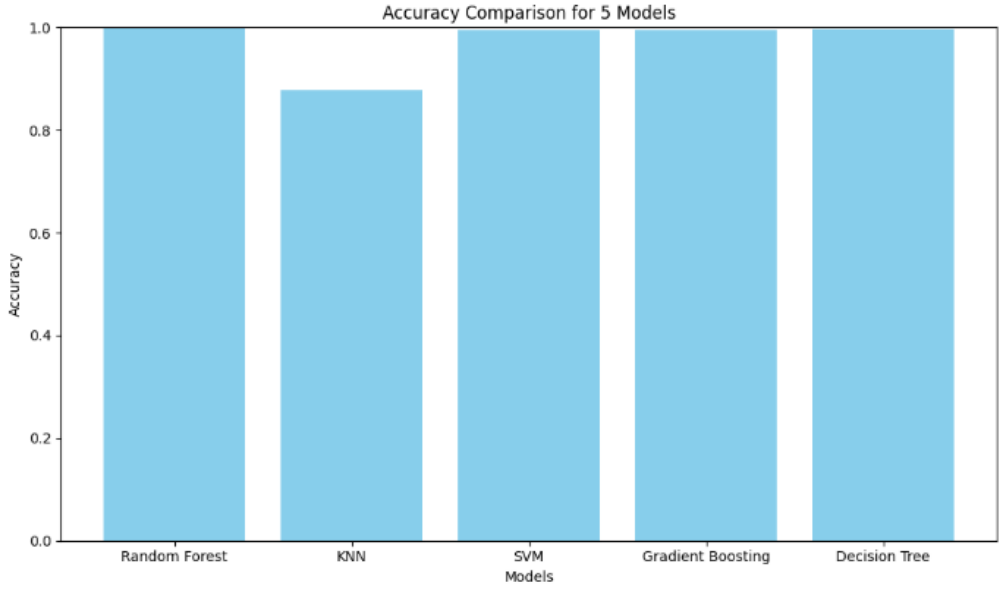


Figure 9.20: Comparison of Accuracy for Every Model

The graphical representation in Figure 9.20 reinforces the findings from Table 9.7. It is clearly visible that the ensemble-based models—Random Forest and Gradient Boosting—achieve top-tier performance, with Random Forest slightly outperforming the others at 99.88% accuracy. Decision Tree and SVM also perform exceptionally well, both exceeding 99% accuracy. In contrast, the K-Nearest Neighbors (KNN) model lags significantly behind with an accuracy of just 87.75%, indicating that it may not be as suitable for this task without further optimization or feature engineering. This comparison highlights the robustness and reliability of ensemble methods for detecting fake news across diverse datasets.

Notebook of ML Models: Click here to view the interactive Notebook of ML Models.

CHAPTER 10: Website Interface Design

10.1 Browser Extension

A browser extension is a small software application designed to enhance the functionality of a web browser by adding specific features or tools that improve the user experience, such as ad blockers, password managers, or productivity tools. These extensions are installed directly in the browser and can interact with web pages, modify their content, or access browser features to provide additional functionality.

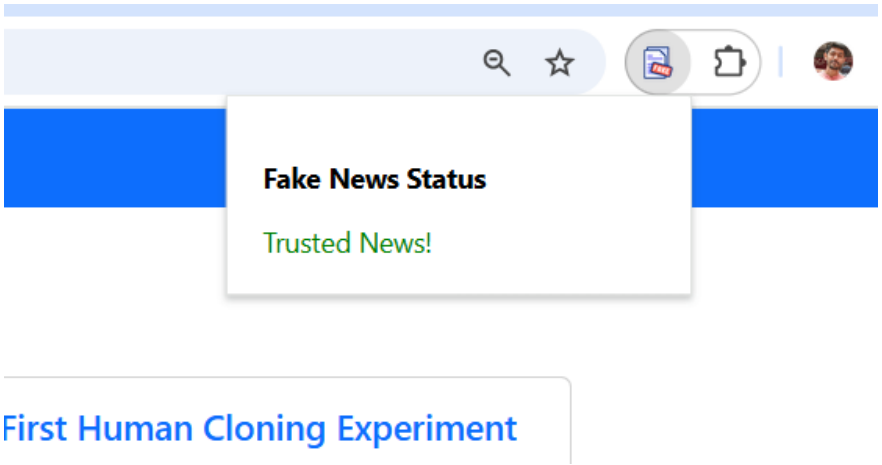


Figure 10.1: Prototype of the Chrome Extension for Fake News Detection

As shown in Figure 10.1, the Chrome extension for Fake News Detection assists users in identifying misleading news articles by analyzing the URL of the active webpage. For example, it labels domains such as `fake-news-site.com` as *Fake News* and others as *Trusted News*. When activated, the extension evaluates the active browser tab and updates the popup with the result using color-coded indicators—red for fake news and green for trusted sources.

This prototype currently performs mock detections and aims to integrate third-party APIs (such as NewsAPI or Google Fact Check Tools) for real-time verification in future versions. It requires permissions to access active tabs, local storage, and news verification APIs. The extension is built using Manifest Version 3 and includes a background script for configuration and logging.

Although we aimed to implement real-time news validation, time constraints limited the development. The extension currently defaults to marking news as trusted. The implementation was done using HTML, JavaScript, and JSON, and the extension has been successfully uploaded to the Chrome browser for testing purposes.

Browser Extension Video: Click here to view the interactive interface video of the Browser Extension Fake News Detection System.

10.2 Website-based News Detection

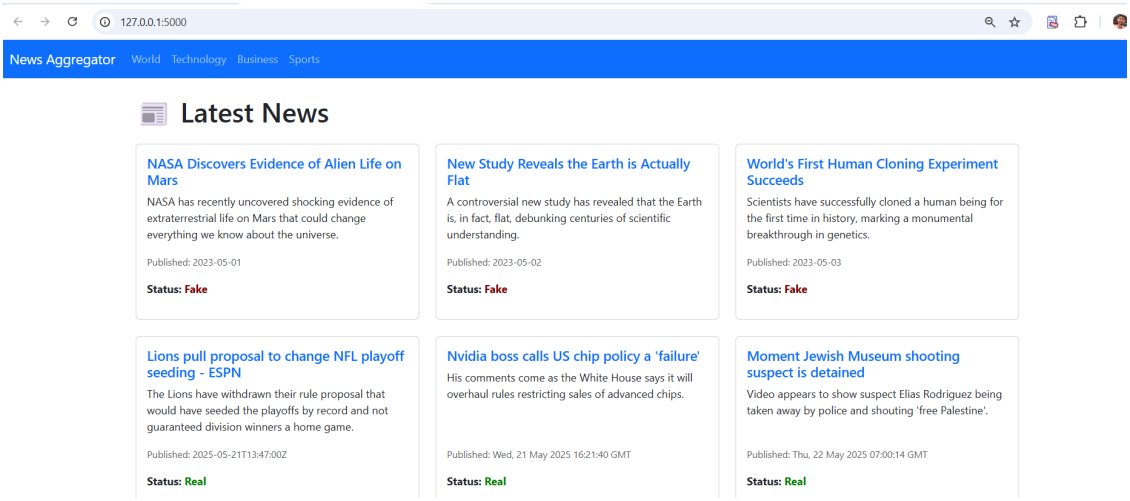


Figure 10.2: User Interface of the Website-Based Fake News Detection System

Figure 10.2 illustrates our website-based fake news detection system. This system integrates machine learning and real-time news feeds to classify and display news articles.

A `.pkl` file (Pickle file) is a serialized Python object, typically used to store trained machine learning models. Using Python's `pickle` module, we saved our trained model in this format to load it efficiently when the application runs. This allows the system to perform background predictions without retraining the model each time.

An API (Application Programming Interface) is a defined method for software components to communicate. We used APIs such as `newsapi.org` and RSS feeds to fetch live headlines from trusted news sources and display them on our platform.

Once the machine learning model was complete and saved as a `.pkl` file, we integrated it into our web application using Python (Flask), HTML, and JavaScript. The website fetches current news articles and runs them through the fake news detection model.

To test the detection functionality, we manually inserted known fake news items at the top of the news feed. However, because the API returned results in random order and imposed a limit on the number of articles shown, our custom fake news items were sometimes excluded from the final list. This issue was resolved by disabling random selection and ensuring our fake articles were prioritized for testing purposes.

Although we attempted to display headlines along with images, this functionality did not work reliably due to limitations in the APIs and restrictions from some news websites.

Overall, the website successfully integrates real-time news updates and a machine learning model to classify news articles, demonstrating the practical application of our detection system.

Website Video: Click here to view the interactive interface video of the Website-Based Fake News Detection System.

CHAPTER 11: Conclusion and Future Work

This project presents a comprehensive and practical solution for real-time fake news detection by leveraging machine learning techniques integrated with modern web technologies. Through the development of a responsive website and a browser extension powered by ML models, we aimed to assist users in efficiently verifying the credibility of online news content.

During the development phase, we reviewed over 12 relevant research papers, which provided valuable insights into existing systems and their limitations. Based on the literature, we categorized prior work into two main domains: browser extension-based detection tools and machine learning-based detection systems.

From the browser extension domain, we identified several significant limitations. Many existing tools face scalability issues due to manual configuration requirements and limited compatibility with Java-based environments. Additionally, most of these extensions are designed solely for English-language content, resulting in poor handling of imbalanced datasets and a general lack of explainability in their predictions. Furthermore, these systems often struggle with social media integration, exhibit low levels of trustworthiness, and are restricted to specific browsers, thereby reducing their overall usability and impact.

To address these challenges, we designed a more flexible, user-friendly, and technically robust browser extension. On the machine learning side, we encountered several well-documented limitations and took strategic steps to mitigate them. One major challenge was the reliance on text-only analysis, which lacked support for multimedia formats such as images and videos, as well as multilingual content. Additionally, the datasets available were often constrained in scope, covering narrow topics like COVID-19 or lacking sufficient diversity for generalization. Our system also needed to overcome the absence of real-time prediction capabilities and the limited use of deep learning architectures or advanced hyperparameter tuning techniques. Furthermore, many models we reviewed exhibited overfitting and performed poorly when applied to data outside their training domains. Lastly, the high complexity of model training and the lack of adaptability across platforms posed significant hurdles. We worked to alleviate these issues through careful dataset curation, preprocessing, and the implementation of robust machine learning pipelines.

To overcome these challenges, we utilized a carefully curated and balanced dataset, applied advanced preprocessing techniques, and implemented multiple machine learning models, including Random Forest, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, and Gradient Boosting. Using NLP techniques like TF-IDF vectorization and feature selection, we significantly improved classification performance.

Compared to previous works cited in the literature (e.g., [5], [6], [7]), our system achieved an accuracy exceeding 99%, thereby outperforming many existing approaches. This success was driven by our strategic integration of machine learning with NLP algorithms and rigorous evaluation using standard performance metrics.

In future work, we aim to integrate the browser extension with the website to provide a seamless and connected user experience. This integration will allow users to access real-time verification results across both platforms. Additionally, we plan to develop a mobile application to enable users to verify news content directly from their smartphones, enhancing accessibility and convenience.

Furthermore, future enhancements will include support for multilingual and multimedia content, enabling detection across different languages and formats such as images and videos. These improvements will significantly broaden the applicability and impact of our system in combating misinformation.

Bibliography

[1] D. Paschalides, C. Christodoulou, K. Orphanou, R. Andreou, A. Kornilakis, G. Pallis, M. D. Dikaiakos, and E. Markatos, "Check-It: A plugin for detecting fake news on the web," Preprint submitted to Special Issue on Detecting, Understanding and Countering Online Harms, Jul. 6, 2021.

[2] M. E. Taşcı and Y. Bayrakdar Yilmaz, "Developing a web browser extension to prevent the spread of fake news," Int. J. Eng. Comput. Sci., vol. 13, no. 10, pp. 26576–26588, Oct. 2024, doi: 10.18535/ijecs/v13110.4921.

[3] E. L. Amoruso and S. P. Johnson, "A web infrastructure for certifying multimedia news content for fake news defense," in Proc. IEEE, 2022, doi: 10.1109/XXX.2022.XXXXXXX. (Note: Replace the placeholder DOI with the actual DOI if available.)

[4] N. N. Prachi et al., "Detection of fake news using machine learning and natural language processing algorithms," Journal of Advances in Information Technology, vol. 13, no. 6, Dec. 2022.

[5] N. Cavus, M. Goksu, and B. Oktekin, "Real-time fake news detection in online social networks: FANDC cloud-based system," Journal name, vol. X, no. Y, pp. xxx–xxx, Year.

[6] J. Jouhar, A. Pratap, N. Tijo, and M. Mony, "Fake News Detection using Python and Machine Learning," in Proc. 5th Int. Conf. Innovative Data Communication Technologies and Application (ICIDCA), 2024, Procedia Computer Science, vol. 233, pp. 763–771. doi: 10.1016/j.procs.2024.03.082.

[7] A. Jain, H. Khatter, and A. Shakya, "A smart system for fake news detection using machine learning," in Proc. Int. Conf. on Recent Advances in Computer Science and Technology, Dr. A.P.J. Abdul Kalam Technical University, Lucknow, India, 2023.

[8] N. Cavus, M. Goksu, and B. Oktekin, "Real-time fake news detection in online social networks: FANDC Cloud-based system," Scientific Reports, vol. 14, Art. no. 25954, 2024. doi: 10.1038/s41598-024-32543-9

[9] Y. N. Yashaswini, G. S. Srinath, A. Nagaraj, S. Sudharshan, S. Faizal, and Rajkumar, "A study on identification of fake news using real-time analytics," Tuijin Jishu/Journal of Propulsion Technology, vol. 44, no. 6, pp. 3014–3020, 2023.

[10] A. Thota, P. Tilak, S. Ahluwalia, and N. Lohia, "Fake News Detection: A Deep Learning Approach," SMU Data Science Review, vol. 1, no. 3, Art. no. 10, 2018. [Online]. Available: https://scholar.smu.edu/datasciencereview/vol1/iss3/10

[11] J. A. Nasir, O. S. Khan, and I. Varlamis, "Fake news detection: A hybrid CNN-RNN based deep learning approach," International Journal of Information Management Data Insights, vol. 1, p. 100007, 2021. Available: https://www.elsevier.com/locate/jjimei

[12] D. Wang, W. Zhang, W. Wu, and X. Guo, "Soft-Label for Multi-Domain Fake News Detection," IEEE Access, vol. 11, pp. 1–12, 2023, doi: 10.1109/ACCESS.2023.3313602.

[13] K. M. M. Uddin, M. J. Alam, J.-E. Anawar, M. A. Uddin, and S. Aryal, "A Novel Approach Utilizing Machine Learning for the Early Diagnosis of Alzheimer's Disease," *Biomedical Materials & Devices*, vol. 1, pp. 882–898, Apr. 2023.

[14] M. Almseidin, M. Alzubi, S. Kovacs, and M. Alkasassbeh, ”Evaluation of Machine Learning Algorithms for Intrusion Detection System,” in *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, Subotica, Serbia, Sep. 14–16, 2017, pp. 000–000.