

A FIELD PROJECT REPORT

on

**“FAKE NEWS DETECTION SYSTEM”**

**Submitted**

by

221FA04436

G. Likhitha

221FA04546

V. Gnanesh

221FA04537

M. Lasya

221FA04565

G. Manisha

**Under the guidance of**

Sajida Sultana.Sk

Assistant Professor



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH Deemed**

**to be UNIVERSITY**

**Vadlamudi, Guntur.**

**ANDHRA PRADESH, INDIA, PIN-522213.**

**CERTIFICATE**

This is to certify that the Field Project entitled “**Fake News Detection system**”. For Digital World that is being submitted by 221FA04436(G.Likhitha), 221FA04537(M.Lasya), 221FA04546(V.Gnanesh), 221FA04565(G.Manisha) for partial fulfilment of Field Project is a bonafide work carried out under the supervision of Ms. Sajida Sultana.Sk , Assistant Professor, Department of CSE.

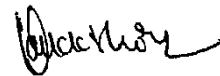
Guide name& Signature



Dr. S. V. Phani Kumar

Assistant/Associate/Professor,  
CSE

HOD,CSE



Dr.K.V. Krishna Kishore

Dean, SoCI

## DECLARATION

We hereby declare that the Field Project entitled **“Fake News Detection System For Digital World”** that is being submitted by 221FA04436(G.Likhitha), 221FA04537(M.Lasya), 221FA04546(V.Gnanesh), 221FA04565(G.Manisha) in partial fulfilment of Field Project course work. This is our original work, and this project has not formed the basis for the award of any degree. We have worked under the supervision of Ms. Sajida Sultana.Sk ., Assistant Professor, Department of CSE.

By

221FA04436(G.Likhitha),  
221FA04537(M.Lasya),  
221FA04546(V.Gnanesh),  
221FA04565(G.Manisha)

Date:

# ABSTRACT

With the exponential growth of online disinformation, comes a new requirement to build advanced systems that can detect fake news-a task that has fueled machine learning and NLP techniques used in this project. The data applied here consists of 4,009 news articles with links, headlines, body, and their corresponding binary labels whether the news is real or fake. Methods involved in this exploratory data analysis (EDA) include handling missing values, label distribution analysis, and correlation feature identification. This involved filling in missing data, standardization of features, and text data preparation through the application of the concepts of tokenization, stemming, and stop word elimination.

Some of these binary classifications were applied using machine learning models that produced good results in all possible aspects of accuracy, precision, recall, and F1-score. The **Support Vector Machine (SVM)** showed the highest accuracy at 98.12%, followed by **Random Forest** at 97.62%, and then **LightGBM** as well as **SVD** at 97.99%. Other models included **Logistic Regression** (96.37%), **XGBoost** (97.37%), **Gradient Boosting** (96.62%), **Decision Tree** (93.86%), and **Naive Bayes** (90.98%), all of which showed well-excelled results. Analysis confirms the applicability of machine learning in classifying a news piece as either of two categories: real or fake, which will aid further condensation of digital authenticity.

**Key Words:** Fake news detection, Machine learning models, Natural language processing (NLP), Support Vector Machine (SVM).

# TABLE OF CONTENTS

1. Introduction	1
1.1 What is a Fake news detection system?	2
1.2 Importance of Detection systems in digital age	3
1.3 The impact of machine learning in detection systems	3
1.4 Challenges in building detection systems	4
1.5 Applications of detection systems in various domains	5
2. Literature Survey	7
2.1 Literature Review of Existing Algorithms and Techniques	8
2.2 Motivation behind this project	10
3. Proposed System	12
3.1 Input dataset	14
3.1.1 Detailed features of dataset	14
3.2 Data Pre-processing	15
3.2.1 Explanatory Data Analysis (EDA)	16
3.2.2 Hybrid Feature Extraction	17
3.3 Model Building	18
3.4 Methodology of the system	21
3.5 Model Evaluation	24
3.6 Model Deployment	26
3.7 Performance Metrics	27
4. Implementation	29
4.1 Environment Setup	30
4.2 Sample code for data pre-processing and model implementation	31
5. Experimentation and Result Analysis	33
6. Conclusion and Future Enhancement	41
7. References	43

## LIST OF FIGURES

Figure 1.1 Flow Diagram for Fake News Detection System	
Figure 3.1 News Dataset	13
Figure 3.2 Flow Diagram for Fake News Detection System (Proposed Model)	14
Figure 3.3 SVM Architecture	15
Figure 4.1 Data Collection	34
Figure 4.2 Data Processing	33
Figure 4.3 Data Cleaning	
Figure 4.4 Support Vector Machine (SVM)	
Figure 5.1 Distribution of Classes (Fake/Real)	35
Figure 5.2 Distribution of Word Count in Cleaned Headline and Cleaned Body	36
Figure 5.3 Pair Plot of Numerical Features (Label)	37
Figure 5.4 Correlation Matrix Heatmap	38
Figure 5.5 Top 20 Most Frequent Words in Cleaned Body	
Figure 5.6 PCA of TF – IDF Features	
Figure 5.7 t- SNE of Word2Vec Features	
Figure 5.8 Correlation Heatmap of Selected Features	
Figure 5.9 Confusion Matrix of Logistic Regression	
Figure 5.10 Confusion Matrix of SVM	
Figure 5.11 Confusion Matrix of Random Forest	
Figure 5.12 Confusion Matrix of Naïve Bayes	
Figure 5.13 Confusion Matrix of Decision Tree	
Figure 5.14 Confusion Matrix of GBM	
Figure 5.15 Confusion Matrix of XG Boost	
Figure 5.16 Confusion Matrix of Light GBM	
Figure 5.17 Confusion Matrix of SVD	
Figure 5.18 Comparison of Model Accuracies	
Figure 5.19 Model Performance Comparison	
Figure 5.20 Classification metrics	

## LIST OF TABLES

Table 3.1 Performance Comparison of Classification Models
---

36
----

# **CHAPTER-1**

## **INTRODUCTION**



# 1. INTRODUCTION

The chapter aims to introduce the concept of detecting fake news systems, focusing their development and the growing importance of such systems in the digital world. Their critical role in keeping information integrity, especially on the internet, is rather interesting. In an age where misinformation just keeps spreading, the accurate detection of such content will play a significant role in protecting users against false information and maintaining their confidence and the reliability of digital media. The paper will further explain in detail how organizations, media platforms, and governments rely increasingly on fake news detection systems in an effort to curb misinformation and preserve public trust within online information ecosystems.

## 1.1 What is a Fake news detection system?

The tech solution designed to sift through digital-based news websites and social networks, meant to tag false or misleading information, is called a fake news detection system. Techniques of machine learning and natural language processing are employed by these systems to examine the content of news for distinguishing between legitimate news and fabricated stories in terms of various factors such as the textual pattern, credibility of source, and truthfulness of the facts. Most complex algorithms of Random Forest, SVM, and XGBoost are utilized to enhance detection results. In this manner, these technologies crosscheck claims and also monitor the behavior of users on various social media and in the end, the fight against misinformation makes online information trustworthy and fills digital content with confidence.

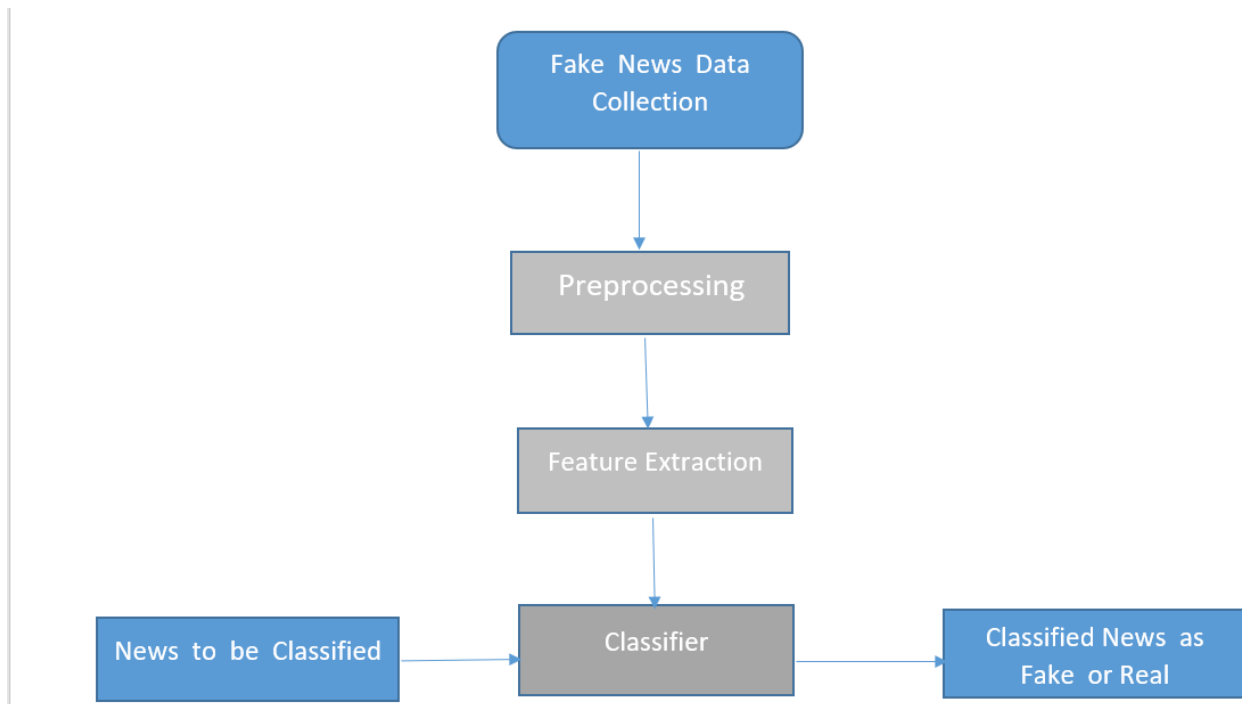


Figure 1.1 Flow Diagram for Fake News Detection System

## 1.2 Importance of Detection systems in digital age

Fake news detection systems are inevitable in safeguarding information integrity in the world of the digital age. It is such a system that will not let unsubstantiated news wander around, and therefore, for instance, it can influentially prejudice public opinion, provoke social unrest, or harm the image of an individual or an organization. The detection systems accurately identify content and filter out false information, thereby protecting users from being led astray or unduly harmed by such information, through reliable sources. These systems also make the platforms more credible, diminish the risk of disinformation campaigns, and create a more reliable and informed online environment, which is indispensable to public discourse and decision-making.

## 1.3 The impact of machine learning in detection systems

Machine learning (ML) plays a critical role in fake news detection systems by enabling them to analyze vast amounts of data and accurately distinguish between real and false information. This section will discuss:

- **Supervised learning algorithms** like Logistic Regression and Support Vector Machines (SVM), which classify news articles based on labeled data, identifying deceptive patterns and misinformation.

- **Unsupervised learning techniques** like Clustering and Anomaly Detection, which uncover hidden trends and irregularities in news content that may signal misinformation without relying on predefined labels.

#### 1.4 Challenges in building detection systems

Several challenges arise in building an efficient detection system:

- **Data Quality and Labeling:** Ensuring high-quality, accurately labeled datasets is crucial for training machine learning models. Misinformation can be subtle and context-dependent, making it difficult to create comprehensive and reliable training sets.
- **Evolving Misinformation Tactics:** The tactics used by those spreading fake news are constantly changing, which can outpace detection models. This evolution requires ongoing model updates and retraining to stay effective against new types of misinformation.
- **Complexity of Language and Context:** Natural language is nuanced, and understanding context, sarcasm, and cultural references can be challenging for detection systems. These complexities can lead to false positives or negatives, reducing the system's reliability.
- **Scalability and Real-Time Processing:** As the volume of online content grows exponentially, detection systems must efficiently process and analyze massive datasets.

#### 1.5 Applications of detection systems in various domains

Detection systems are not limited to digital platform but are also widely used in:

- **Social Media:** Identifying and flagging misleading content on platforms like Facebook and Twitter.
- **News Organizations:** Verifying the authenticity of articles to ensure credible information is published.
- **Search Engines:** Filtering out unreliable sources from search results to provide trustworthy information.
- **Public Health:** Countering misinformation during health crises to ensure the public receives accurate information.
- **Educational Platforms:** Ensuring the accuracy of educational materials by filtering out unreliable information.

# **CHAPTER-2**

## **LITERATURE SURVEY**

## 2. LITERATURE SURVEY

### 2.1 Literature review

The rise of online news and social media has increased the spread of misinformation, making fake news a significant issue globally. Fake news detection systems (FDS) leverage machine learning and natural language processing (NLP) to combat this problem by identifying false information in real-time. Models like SVM, Logistic Regression, and LightGBM, along with hybrid feature extraction methods such as TF-IDF, N-grams, and word embeddings, help improve detection accuracy. As the field advances, future systems may incorporate deep learning and multimodal approaches to enhance their ability to detect misinformation across both text and visual content.

**Huyen Trang Phan et al. [1]** In their work, the authors provide a detailed survey of graph neural network (GNN)-based methods for fake news detection. The study outlines key challenges such as handling vast volumes of data from social networks and proposes a GNN taxonomy for categorizing fake news detection approaches. By comparing models across categories, they highlight the advantages and limitations of GNNs. Future research will focus on overcoming these limitations and expanding the application of GNN-based methods for more robust fake news detection systems.

**Noureddine Seddari et al. [2]** The authors present a hybrid fake news detection system that combines linguistic and knowledge-based features to improve accuracy. The system utilizes features like word count, sentiment analysis, and source reputation to detect fake news. By using fewer features compared to other methods, it achieves 94.4% accuracy, outperforming individual approaches. Future work will focus on refining these features and testing the system across more platforms to enhance performance in various real-world scenarios.

**Rafał Kozik et al. [3]** The authors conduct a meta-analysis of state-of-the-art machine learning and AI-based fake news detection methods. They evaluate various models using precision, F1-score, and recall, highlighting the potential of deep learning techniques. By comparing models across multiple benchmark datasets, the study provides insights for improving fake news detection accuracy and scalability. Future research will focus on refining detection algorithms and exploring

new datasets for more comprehensive testing.

In their work, **Youcef Djenouri et al. [4]** present a novel system called Decomposition MapReduce Mining for Fake News Analysis (DMRM-FNA), which is designed to address the challenges of fake news detection on social networks. The system is built upon the integration of decomposition strategies and MapReduce techniques to efficiently mine relevant patterns from massive datasets. Key contributions of the study include the development of a multi-objective k-means algorithm for clustering and the application of a parallel mining approach using MapReduce to discover frequent patterns. Tests on large datasets demonstrate the framework's improved performance in terms of memory usage, accuracy, and processing time. Future work aims to further refine the model by considering other objectives like utility values of sentences and extending its application to multilingual and cross-social network environments.

**Minjung Park et al. [5]** in their work present a user-centered fake news detection model using classification algorithms and machine learning techniques. They address the limitations of existing models that primarily focus on linguistic characteristics, introducing a more comprehensive approach that considers user behavior, content, and social network features based on social capital theory. The study applies the XGBoost model for feature selection, followed by the implementation of several classification algorithms including SVM, RF, LR, CART, and NNET. Tests reveal the Random Forest model's superior accuracy, achieving a prediction rate of about 94%. The authors emphasize that future work will involve expanding the model to multilingual environments and enhancing detection capabilities by incorporating more complex features like utility values.

**Eman Elsaed et al. [6]** in their paper propose a voting classifier framework for detecting fake news on social media, aiming to enhance the accuracy of detection through a combination of machine learning models. Their approach leverages preprocessing techniques, such as lemmatization and feature extraction (TF-IDF, DOC2VEC), alongside feature selection methods like chi-square and ANOVA. The system utilizes multiple classifiers, including Naïve Bayes, SVM, Logistic Regression, Random Forest, and more, to form an ensemble voting mechanism. Experimental results on three benchmark datasets—Fake-or-Real-News, Media-Eval, and ISOT—demonstrate that the model achieves up to 100% accuracy on ISOT data. Future research will focus on extending the system to handle multimedia content like images and videos.

In their work, **Almarashy et al. [7]** present a novel approach for enhancing fake news detection by combining multiple features through deep learning methods. The proposed architecture integrates convolutional neural networks (CNNs) to extract spatial features, bi-directional long short-term memory (BiLSTM) networks to capture temporal features, and Term Frequency-Inverse Document Frequency (TF-IDF) for global text features. These features are then classified using a fast learning network (FLN). This multi-feature fusion approach is tested on two publicly available datasets, ISOT and FA-KES, and results demonstrate that it significantly improves detection accuracy. Future work will focus on adopting non-English datasets and further improving precision by leveraging transfer learning techniques.

**Mannan et al. [8]** conducted a comprehensive study on fake news detection techniques, emphasizing the importance of sentiment analysis in improving detection accuracy. The paper highlights the evolution of fake news, making it necessary to integrate sentiment analysis to detect intent and emotional tone, which are often embedded in false information. The study suggests that incorporating sentiment analysis into traditional detection methods, such as machine learning algorithms, could significantly enhance the identification of emotionally charged or opinion-based fake news. Future work will focus on refining sentiment analysis models and applying them across diverse social media platforms to improve detection accuracy.

**Nawaz et al., 2024 [9]** present the FNACSPM framework, which applies sequential pattern mining (SPM) for fake news detection and classification. The study uses six publicly available datasets of both real and fake news. After preprocessing the text, SPM algorithms, such as TKS and CM-SPAM, are applied to identify frequent patterns, which are subsequently used for classification. The approach outperforms state-of-the-art methods and improves classification accuracy across multiple datasets. Future work includes expanding testing across diverse data sources to enhance generalizability.

**Xie et al., 2024 [10]** propose the **KEHGNN-FD** model for detecting fake news using a Knowledge Graph-Enhanced Heterogeneous Graph Neural Network. The approach tackles two primary challenges: linking knowledge graphs (KGs) to news content and balancing information from both KGs and news. Their model builds a heterogeneous graph based on news entities and topics, employing graph attention mechanisms to adaptively aggregate information. This fusion significantly improves fake news detection over seven baselines, as demonstrated in experiments across four datasets. Future work aims to enhance the integration of features from multiple KGs

for better accuracy.

**Ravish et al., 2022 [11]** propose a **Featured-Based Optimized MSVM Classification** model for detecting fake news by leveraging multi-layered Principal Component Analysis (PCA) and a firefly-optimized algorithm for feature extraction and selection. The system uses Multi-Support Vector Machines (MSVM) for classification, achieving improved accuracy compared to existing models on multiple datasets. Future work will focus on testing the model on larger, more diverse datasets to further enhance its performance and adaptability.

**Ahmed et al. (2024) [12]** introduce a novel approach to enhancing fake news detection models through the application of text augmentation techniques and adversarial training. The researchers demonstrate how text augmentation can improve classifier accuracy by generating new data from existing fake news datasets, showing an increase in performance by up to 11%. Additionally, adversarial attacks are used to challenge trained classifiers, with success rates reaching 90% before retraining. After adversarial retraining, the attack success rate drops to less than 28%, highlighting the resilience improvement of the model. Future work will focus on refining adversarial methods and extending the framework to handle more complex and evolving datasets.

In their work, **Choudhury et al. (2022) [13]** present a novel approach to fake news detection using a genetic algorithm (GA) integrated with machine learning classifiers. This method aims to address the challenges posed by traditional fake news detection algorithms, particularly in terms of computational complexity and optimization. The proposed solution involves utilizing genetic algorithms to enhance the performance of machine learning classifiers such as Support Vector Machines (SVM), Naive Bayes, Logistic Regression, and Random Forest. The genetic algorithm optimizes the feature selection process by simulating evolutionary behavior, thus improving accuracy in identifying fake news. Tests conducted on datasets such as LIAR, Fake Job Posting, and Kaggle Fake News datasets show promising results, with the highest accuracy achieved by SVM and Random Forest classifiers. Moving forward, the researchers aim to refine the model by increasing the population size and feature set to enhance performance across diverse datasets.

**Thakur et al.** present **MYTHYA [14]**, a real-time fake news detection system that combines a Gradient Boosted Decision Tree (GBDT) and Convolutional Neural Network (CNN). Their approach detects the stance of news articles and classifies them as real or fake, achieving an accuracy of 97.59%. The system addresses the challenge of classifying misleading information by



extracting keywords and utilizing both stance detection and news extraction modules. The proposed system demonstrates a robust performance, with future enhancements aimed at social media integration for wider application

## **2.2 Motivation**

The motivation stems from the growing need for efficient, scalable, and personalized recommendation systems in e-commerce. As online platforms continue to expand, providing relevant product recommendations becomes crucial for maintaining user engagement and driving sales.

# **CHAPTER-3**

## **PROPOSED SYSTEM**

### 3. PROPOSED SYSTEM

The proposed system for the **Fake News Detection Model** focuses on the accurate identification of false information, using combined hybrid techniques in feature extraction and a range of models in machine learning. The basis of the system is in the dataset from **Kaggle's Fake News Detection Dataset**, consisting of articles identified as being either fake or real. The hybrid feature extraction technique includes techniques like Term Frequency-Inverse Document Frequency (TF-IDF), N-grams, and Word2Vec to greatly capture the textual features of news articles for better performance of the model.

The features are passed to the training sets of machine learning algorithms for developing the models. These include Logistic Regression (LR), Random Forest (RF), Decision Trees (DT), **Support Vector Machines (SVM)**, Singular Value Decomposition (SVD), LightGBM, Gradient Boosting Machines (GBM), XGBoost, and Naive Bayes (NB). These models are applied to create classifiers that determine the news articles as fake or real. Of all these models, SVM performed well and led all other models with the highest accuracy of 98.12% in detecting fake news.

Data Preprocessing is the most significant step in this system because the missing values are replaced, and text data is converted into numeric form before training the model. The first cleaning step removes all the unnecessary characters or stop words that were attached to the data and act like noises. Then comes the hybrid technique of feature extraction, such as **TF-IDF, N-grams, and Word2Vec**, to convert the textual information into vectors that would be machine learning processable.

To test the models, the metrics like accuracy and a host of other metrics such as F1 score, precision, and recall are used. Along with it, a **confusion matrix** is also provided which in turn gives insight into how many of the predictions or outcomes are true positives, true negatives, false positives, or false negatives in a manner that fully illustrates the accuracy of the model developed. It means focusing on model optimization through hyperparameter tuning for a more efficient and accurate version of the fake news detector. This will lead to an ideal proposed system capable of delivering a robust and scalable solution to the infestation of misinformation.

### 3.1 Input dataset

The dataset used in this project, sourced from Kaggle, is a Fake News Detection Dataset consisting of several components related to news articles. These include **URLs**, which link to the original news articles; **Headlines**, representing the title or headline of the article; and the **Body**, containing the main text or content of the news. Additionally, the dataset provides a **Label**, a binary indicator where 1 represents real news and 0 denotes fake news. The textual data from the **Headlines** and **Body** are utilized for feature extraction, while the **Label** serves as the target variable, allowing machine learning models to be trained to differentiate between real and fake news.

URLs	Headline	Body	Label
<a href="http://www.bbc.com/news/world-us-canada-414191...">http://www.bbc.com/news/world-us-canada-414191...</a>	Four ways Bob Corker skewered Donald Trump	Image copyright Getty Images\nOn Sunday morning...	1
<a href="https://www.reuters.com/article/us-filmfestiva...">https://www.reuters.com/article/us-filmfestiva...</a>	Linklater's war veteran comedy speaks to modern America	LONDON (Reuters) - "Last Flag Flying", a comedy...	1
<a href="https://www.nytimes.com/2017/10/09/us/politics...">https://www.nytimes.com/2017/10/09/us/politics...</a>	Trump's Fight With Corker Jeopardizes His Legislative Push	The feud broke into public view last week when...	1

Figure 3.1 News Dataset

#### 3.1.1 Detailed Features of the Dataset

##### □ News Dataset Features:

- **URLs:** Links to the original news articles or sources.
- **Headline:** The title of the news article, summarizing the main topic or focus.
- **Body:** The full text of the news article, providing detailed information and context on the reported event or topic.
- **Label:** A classification or categorization of the article, which could represent whether the article is real or fake, relevant categories, or sentiment (depending on your task).

### 3.2 Data Pre-processing

Data pre-processing is considered the most primary activity of cleaning, transforming, and structuring raw data to prepare it for analysis and modeling so that quality and utility of the data are enhanced. It mainly includes activities such as the handling of missing values, correction of errors, feature encoding, and scaling of data to put them in their best possible form for further analysis. It ranges from various operations and transformations aimed to clean the raw data, ensuring it would be clean, structured, and amenity to subsequent analysis. This process is driven by its manifold significance in data science and analysis.

This data will be made ready using these processes: cleaning, transformation of data, feature engineering, dimensionality reduction, handling outliers, scaling, and splitting data to prepare raw data for more accurate and reliable analysis and modeling. Finally, it's about reaching greater meaningful insights, making the right kind of informed decision, and optimizing predictive models suited to a wide range of applications in data science and analysis.

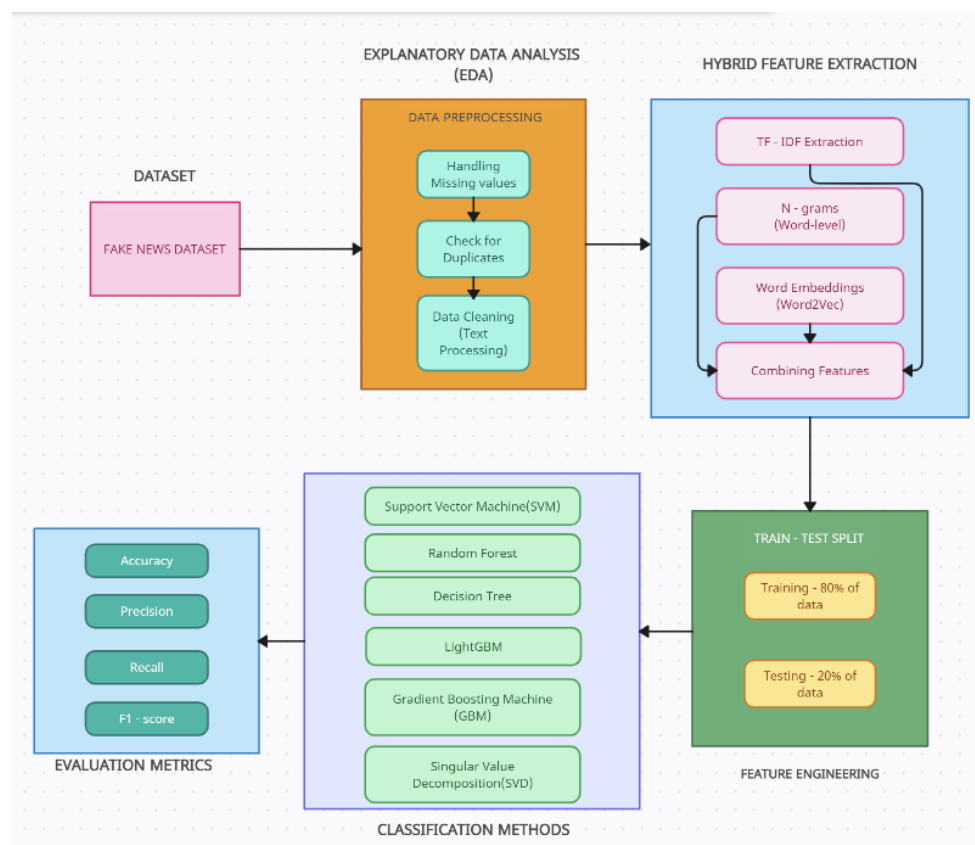


Figure 3.2 Flow Diagram for Fake News Detection System (Proposed Model)

### 3.2.1 Explanatory Data Analysis (EDA)

In the fake news detection dataset, the following steps are taken during EDA to ensure data quality and preparation for modeling:

- **Handling Missing Values:** Missing entries in key fields such as **URLs**, **Headline**, or **Body** are addressed by either imputing values (for numerical fields) or removing records with missing critical data. This ensures a clean dataset for model training.
- **Check for Duplicates:** Duplicate records are identified and removed, especially in **URLs** and **Headline** columns, to prevent bias and redundancy in the dataset.
- **Data Cleaning (Manual Text Processing):** Text fields like **Headline** and **Body** undergo cleaning, including removing special characters, handling punctuation, converting text to lowercase, and eliminating unnecessary whitespace. This step ensures that the text data is standardized and ready for feature extraction.

These EDA steps help ensure the dataset is well-prepared for further analysis and model development.

### 3.2.2 Hybrid Feature Extraction

In hybrid feature extraction for the fake news detection dataset, the following steps are implemented to capture both lexical and semantic information from the text data:

- **TF-IDF (Term Frequency-Inverse Document Frequency):** TF-IDF is applied to the **Headline** and **Body** text fields to quantify the importance of words in relation to the entire corpus. This method highlights terms that are unique to individual articles, while down-weighting common words, helping to distinguish between real and fake news based on word relevance.
- **N-grams (Unigrams, Bigrams, Trigrams):** N-grams are used to capture word sequences within the text. For instance, unigrams (single words), bigrams (two-word combinations), and trigrams (three-word combinations) are extracted from the **Headline** and **Body**. This helps the model recognize context and word patterns that are common in fake or real news, enhancing feature richness.
- **Word2Vec (Word Embeddings):** Word2Vec is employed to generate vector representations of words from the **Headline** and **Body**. These embeddings capture semantic relationships between words by mapping them into a continuous vector space. Words with

similar meanings are positioned closely in this space, allowing the model to grasp the underlying context of the articles, beyond simple word counts.

### 3.3 Model Building

The detection system employs several machine learning models to enhance accuracy in predicting user preferences. Below are the models used in this project, along with their descriptions and respective accuracies:

#### Logistic Regression

**Logistic Regression** is a statistical model that uses a logistic function to model a binary dependent variable. It predicts the probability of an instance belonging to a particular class (0 or 1).

$$P(y = 1 | X) = \frac{1}{1 + e - (\beta_0 + \beta_1 X_1 + \dots + \beta_n X_n)}$$

**How it works:** It applies a logistic function to the weighted sum of input features to generate probabilities. The output is a value between 0 and 1, which can be interpreted as a probability. If the probability exceeds a threshold (usually 0.5), the model predicts the class as 1; otherwise, it predicts 0.

#### Support Vector Machine (SVM)

**SVM** is a supervised learning algorithm used for classification and regression. It works by finding the hyperplane that best separates the data into different classes.

$$f(x) = w^T x + b$$

**How it works:** SVM creates a decision boundary (hyperplane) that separates data points from different classes. The points closest to this hyperplane are called support vectors. The goal is to maximize the margin between these support vectors and the hyperplane.

#### Random Forest (RF)

**Random Forest (RF)** Random Forest is an ensemble learning method that uses multiple decision trees to improve the predictive performance. It averages the output of various decision trees to reduce overfitting and increase accuracy.

$$y' = \frac{1}{T} \sum_{t=1}^T h_t(x)$$

**How it works:** It trains several decision trees on different subsets of the data and features. Each tree gives a prediction, and the final prediction is made by majority voting (classification) or averaging (regression). This approach reduces overfitting.

### Decision Tree (DT)

**Decision Tree** is a flowchart-like tree structure where each internal node represents a feature, each branch represents a decision rule, and each leaf node represents the outcome. It is used for classification and regression tasks.

$$Gini = 1 - \sum_{i=1}^n p_i^2$$

**How It Works:** Starting at the root node, the tree splits the data based on feature values, with each split maximizing information gain (or minimizing impurity). The process continues until each leaf node represents a final classification or prediction.

### Naive Bayes

**Naive Bayes** is a probabilistic classifier based on Bayes' Theorem, assuming independence between features.

$$P(y | X) = \frac{P(X | y)P(y)}{P(X)}$$

**How It Works:** The classifier assumes that the presence of a particular feature in a class is independent of the presence of any other feature (this is the “naive” assumption). Based on this assumption, it calculates the posterior probability for each class and classifies the instance to the class with the highest posterior.

### Gradient Boosting Machine (GBM)

**GBM** is an ensemble learning technique that builds models sequentially, with each model correcting the errors of the previous one. It uses the gradient descent algorithm to minimize the loss.

$$Fm(x) = Fm-1(x) + v \cdot hm(x)$$

**How It Works:** GBM uses decision trees as weak learners and builds them in a stage-wise manner. At each step, the model corrects the errors made by the previous learners. It minimizes a differentiable loss function using gradient descent, hence the name "gradient boosting."



## LightGBM

**LightGBM** is a gradient boosting framework based on decision trees that is designed for efficiency and speed. It uses leaf-wise growth rather than level-wise growth like traditional boosting algorithms.

$$Fm(x) = Fm - 1(x) + v \cdot hm(x)$$

It also uses histogram-based optimization for faster computation.

**How It Works:** Like GBM, LightGBM also builds models sequentially but uses a histogram-based approach for finding splits, making it faster. It also grows trees leaf-wise instead of level-wise, further improving efficiency.

## XGBoost (Extreme Gradient Boosting)

**XGBoost** is an optimized distributed gradient boosting algorithm that is efficient and highly accurate. It incorporates regularization techniques to prevent overfitting.

$$L(\theta) = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k)$$

**How It Works:** XGBoost improves upon traditional GBM by incorporating regularization to reduce overfitting. It also employs advanced optimizations like parallelization and cache-awareness, which makes it faster and more accurate.

## Singular Value Decomposition (SVD)

**SVD** is a matrix factorization technique used in dimensionality reduction and collaborative filtering. It decomposes a matrix into three other matrices.

$$A = U \Sigma V^T$$

**How It Works:** SVD decomposes a matrix into three smaller matrices that capture the essential structure of the original matrix. It helps reduce the number of features while retaining important patterns.

**Why SVM is preferred:** SVM would be used mainly due to its excellent performance with high-dimensional sparse text data, of which there is a great deal after feature extraction is done like TF-IDF or Word2Vec. It has generalization properties with low possibilities of overfitting and can use linear or non-linear decision boundaries using kernels. SVM also generalizes well on small to medium-sized datasets, so it's versatile for text classification, which comes up for tasks like fake news detection.

### 3.4 Methodology of the system

Now that we've covered the basics, let's move into the main part of our **Fake News Detection System**. In this section, we'll walk through how the system works, step by step. Just like how every part of a machine has a specific job to make it run smoothly, our system brings together important elements like data, pre-processing, building models, and evaluating the results. By combining these steps, we create a recommendation system that can understand user preferences, make accurate product suggestions, and keep improving over time, giving users a better shopping experience.

#### **SVM Architecture:**

**Support Vector Machine (SVM)** is a very powerful algorithm in supervised learning tasks, which can be used for classification and regression. SVM finds the optimal boundary-or hyperplane-between the classes while maximizing margins. SVM can use kernel functions for transformation to be efficient even in high-dimensional space, especially for separate classes such as text classification tasks like fake news detection.

#### **The Basics of SVM**

SVM aims to classify the data by finding the decision boundary that separates the points belonging to different classes with the largest possible margin. Within the context of text classification, it should distinguish between categories such as "real" and "fake" news. SVM is particularly good at dealing with sparse, high-dimensional datasets containing natural language processing.

#### **SVM Structure: Key Components**

SVM can be broken down into the following components:

1. **Input Layer (Features):** Input is typically a high-dimensional dataset like text, where each feature represents a word, term, or n-gram, obtained through techniques such as TF-IDF or Word2Vec.
2. **Kernel Trick (Feature Transformation):** The kernel function transforms the input data into a higher-dimensional space, allowing SVM to find a linear separation even if the data is non-linearly separable. Common kernels include linear, polynomial, and RBF (Radial Basis Function).
3. **Decision Boundary (Hyperplane):** SVM identifies the optimal hyperplane that separates data points of different classes. The objective is to maximize the margin—

the distance between the closest data points from both classes (support vectors) and the hyperplane.

### How SVM Architecture Works

1. **Input Layer:** Text data is processed into a numerical feature matrix (using techniques like TF-IDF or Word2Vec). Each row represents an instance (e.g., a news article), and each column represents a feature (e.g., words or phrases).
2. **Hidden Layers (Kernel Transformation):** SVM applies the kernel trick to transform the input space, allowing the algorithm to handle complex, non-linear relationships between data points. For example, the RBF kernel maps data into a higher-dimensional space, where it becomes easier to separate different classes.
3. **Output Layer (Classification):** SVM finds the optimal hyperplane in the transformed space. Data points are classified based on their position relative to this hyperplane—one side represents one class (e.g., real news), and the other side represents the opposite class (e.g., fake news). The model outputs the predicted class for each data point.

### Why SVM is Important in Text Classification (e.g., Fake News Detection)

- **Handles High-Dimensional Data:** SVM excels in text classification where feature spaces are large, such as when using TF-IDF or word embeddings, making it ideal for tasks involving high-dimensional datasets.
- **Robust to Overfitting:** SVM is less likely to overfit, even in cases where the number of features exceeds the number of samples, ensuring more generalizable models.
- **Effective with Sparse Data:** It performs well with sparse matrices common in text data, where most features (words) may not appear in every instance (document).
- **Non-Linear Class Boundaries:** Through kernel functions, SVM can handle non-linearly separable data, making it flexible for complex text classification tasks.

## Key Benefits of SVM Architecture

1. **Accuracy:** SVM often provides highly accurate classification, especially in distinguishing between similar classes, such as real vs. fake news.
2. **Scalability:** It efficiently handles large, sparse datasets, common in natural language processing tasks, by focusing on the most important support vectors.
3. **Versatility:** SVM can be used with various kernel functions (linear, polynomial, RBF) to model complex data patterns, making it versatile for different types of classification problems.

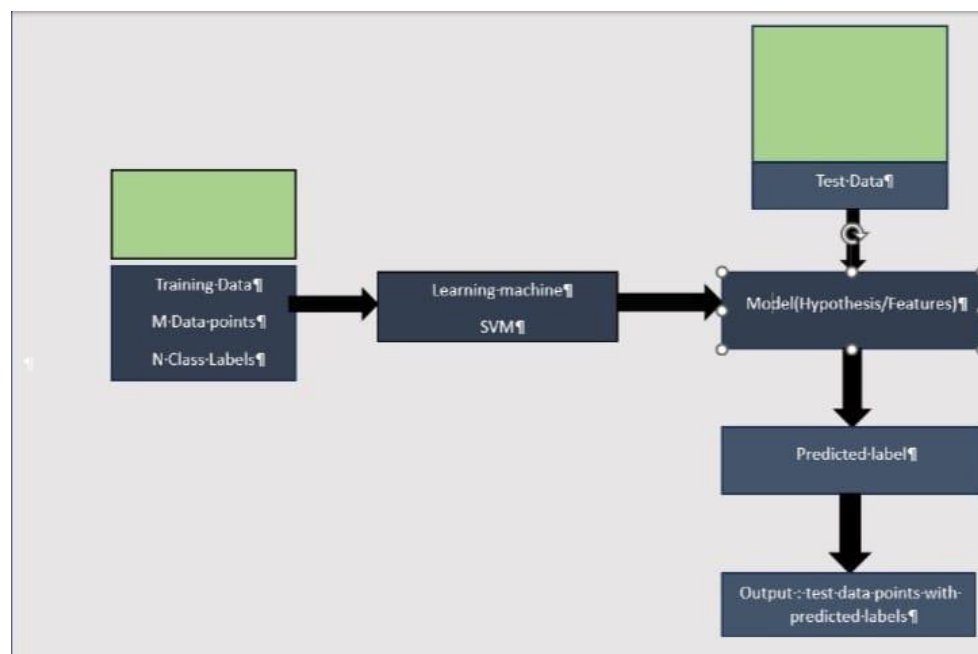


Figure 3.3 SVM Architecture

The image represents the architecture of a **Support Vector Machine (SVM)** learning model. It begins with the training data, which contains `M` data points (the number of samples) and `N` class labels, used to categorize these samples (such as fake or real in a fake news detection context). This training data is fed into the SVM learning machine, a supervised learning algorithm that finds the optimal hyperplane to separate data points into distinct classes. During training, the SVM model builds a decision boundary that best separates the class labels.

Once training is complete, the model, representing the learned decision boundary based on the features of the data, is ready to classify new, unseen data. The test data, which consists

of samples not used in the training process, is input into the model to evaluate its performance. For each test data point, the SVM model predicts a class label (such as fake or real). The final output consists of these test data points along with their predicted labels. The performance of the model is typically measured using metrics like accuracy, precision, recall, and F1-score, which indicate how well the model generalizes to new data. This SVM architecture showcases the complete process, from training to generating predictions for unseen data.

### 3.5 Model Evaluation

Model evaluation is a critical aspect of any machine learning project. It involves assessing the performance and accuracy of a trained model on new, unseen data. This step is essential for several reasons such as:

- i. **Quality Assurance:** Model evaluation helps ensure that the model is capable of making accurate predictions when exposed to real-world data. It acts as a quality control mechanism to validate the model's generalization ability.
- ii. **Comparing Models:** Model evaluation allows for the comparison of multiple models to identify the best-performing one. It helps data scientists and stakeholders make informed decisions about which model to deploy.
- iii. **Fine-Tuning:** The evaluation process can reveal areas where the model performs poorly. This information is valuable for refining the model, making it more robust, and addressing its limitations.
- iv. **Business Decision Support:** In practical applications, model performance impacts critical business decisions. A well-evaluated model provides confidence to stakeholders, leading to better decision-making.

**3.6 Model Deployment:** A thoroughly evaluated model is more likely to be deployed in production systems. It instils trust in the model's predictions, which is essential in real-world applications.

### 3.7 Performance Metrics

Several performance metrics are used to evaluate the effectiveness of the recommendation models:

- **Accuracy:** The ratio of correctly predicted instances (both true positives and true negatives) to the total number of instances. It's useful when class distribution is balanced.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Instances}$$

- **Precision:** The proportion of true positive predictions out of all positive predictions made by the model. It indicates how many of the predicted "real" or "fake" are actually correct.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

- **Recall:** The proportion of actual positives correctly identified by the model. It shows how well the model captures the positive class.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

- **F1 Score:** The harmonic mean of precision and recall, used when there is an imbalance between the two. It balances the need for both precision and recall.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

The results from the experiments show that **SVM** provides the best performance, with a **Precision** of 0.98 and an accuracy of 98.12%, outperforming the other models in both accuracy and recommendation relevance.

Model	Accuracy (%)	Precision	Recall	F1 Score
Logistic Regression	96.37	0.96	0.96	0.96
SVM	98.12	0.98	0.98	0.98
Random Forest	97.62	0.98	0.98	0.98
Decision Tree	93.86	0.94	0.94	0.94
Naïve Bayes	90.98	0.92	0.91	0.91
XG Boost	97.37	0.97	0.97	0.97
GBM	96.62	0.97	0.97	0.97
LightGBM	97.99	0.98	0.98	0.98
SVD	97.99	0.95	0.95	0.95

Table 3.1 Performance Comparison of Classification Models

# **CHAPTER-4**

## **Implementation**

## 4.Implementation

### 4.1 Environment Setup

To implement the recommendation system using SVM, you will need to set up a Python environment with the following libraries:

1. **Python:** Ensure you have Python installed (preferably version 3.6 or higher).
2. **Libraries:**
  - **NumPy:** For numerical operations.
  - **Pandas:** For data manipulation and analysis.
  - **Scikit-learn:** For building the recommendation model.
  - **Surprise:** A specialized library for building and evaluating recommendation systems.
  - **Matplotlib:** For plotting visualizations like confusion matrices.

You can install the required libraries using pip:

**pip install numpy pandas scikit-learn surprise**

**pip install numpy pandas scikit-learn matplotlib**

### 4.2 Sample Code for Data Pre-processing and Model Implementation

```
import pandas as pd

# Load the dataset
file_path = '/content/data.csv'
data = pd.read_csv(file_path)

# Show basic info and head of the dataset to understand its structure
print("Data Info:")
data.info()

print("\nFirst 5 rows:")
print(data.head())
```

Figure 4.1 Data Collection



Check for missing values

```
print("\nMissing values in each column:")
print(data.isnull().sum())

# Option 1: Drop rows with missing values
data_cleaned = data.dropna() # Drop rows with any null values
# OR Option 2: Fill missing values with a placeholder (e.g., empty string or mean for numerical data)
# data_cleaned = data.fillna('')

print("\nData shape after handling missing values:", data_cleaned.shape)
```

Figure 4.2 Data Processing

```
import pandas as pd
import string

data_cleaned = data.dropna()

# Manually defining common English stopwords
common_stopwords = set([
    'i', 'me', 'my', 'myself', 'we', 'oun', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your',
    'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it',
    "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this',
    'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had',
    'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while',
    'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above',
    'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here',
    'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such',
    'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don',
    "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn',
    "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn',
    "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn',
    "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"
])

# Function to clean text manually
def manual_clean_text(text):
    text = text.lower() # Lowercasing
    text = "".join([char for char in text if char not in string.punctuation]) # Removing punctuation
    text = " ".join([word for word in text.split() if word not in common_stopwords]) # Removing stopwords
    return text

# Apply the cleaning function to the 'Headline' and 'Body' columns
data_cleaned['cleaned_headline'] = data_cleaned['Headline'].apply(manual_clean_text)
data_cleaned['cleaned_body'] = data_cleaned['Body'].apply(manual_clean_text)

# Display the first few rows of the cleaned data
print(data_cleaned.head())

# Save cleaned data if needed
data_cleaned.to_csv('/content/cleaned_dataset.csv', index=False)
```

Figure 4.3 Data Cleaning

# SVM

```
from sklearn.svm import SVC

# Train the SVM model
model_svm = SVC(kernel='linear', C=1)
model_svm.fit(X_train_tfidf, y_train)

# Predict on the test data
y_pred_svm = model_svm.predict(X_test_tfidf)

# Accuracy
accuracy_svm = accuracy_score(y_test, y_pred_svm)
print(f"SVM Accuracy: {accuracy_svm * 100:.2f}%")

# Classification report
print(classification_report(y_test, y_pred_svm))

# Confusion matrix
cm_svm = confusion_matrix(y_test, y_pred_svm)

# Plotting confusion matrix
plt.figure(figsize=(6, 4))
sns.heatmap(cm_svm, annot=True, fmt='d', cmap='Blues', xticklabels=['Fake', 'Real'], yticklabels=['Fake', 'Real'])
plt.title('SVM Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```

Figure 4.4 Support Vector Machine (SVM)

## **CHAPTER-5**

### **Experimentation and Result Analysis**

## 5. Experimentation and Result Analysis

This section details the experimentation process undertaken to evaluate the performance of the SVM-based detection system, along with the analysis of the results obtained from the model. The evaluation metrics, data splitting strategy, and comparison with other models are also discussed.

□ **SVM Performance:** The SVM algorithm achieved the best performance in terms of accuracy, indicating its ability to effectively classify news articles as real or fake. It handled the high-dimensional feature space well, which is a common challenge in text-based classification tasks, making it a robust model for fake news detection.

□ **Impact of Hybrid Feature Extraction in Fake News Detection System:** Hybrid feature extraction, combining techniques such as TF-IDF, N-grams, and Word2Vec, significantly enhances the performance of the fake news detection system. By leveraging multiple extraction methods, the system captures both the frequency-based features and semantic relationships within the text. TF-IDF focuses on identifying important words by analyzing term frequency and inverse document frequency, N-grams capture contextual word sequences, while Word2Vec encodes the semantic meaning of words. This combination ensures a more comprehensive representation of the data, allowing machine learning models like SVM to distinguish between fake and real news more effectively. The hybrid approach improves the model's accuracy, robustness, and generalization across different datasets.

□ **Comparison with Other Models:** SVM achieved the highest accuracy (98.12%) compared to other models like Logistic Regression (96.37%), Random Forest (97.62%), and XGBoost (97.37%), outperforming them due to its robust handling of high-dimensional data. While models like LightGBM and SVD performed closely (97.99%), SVM excelled in capturing complex patterns and providing optimal classification. Naive Bayes (90.98%) and Decision Tree (93.86%) lagged behind, struggling with the complexity of textual data.

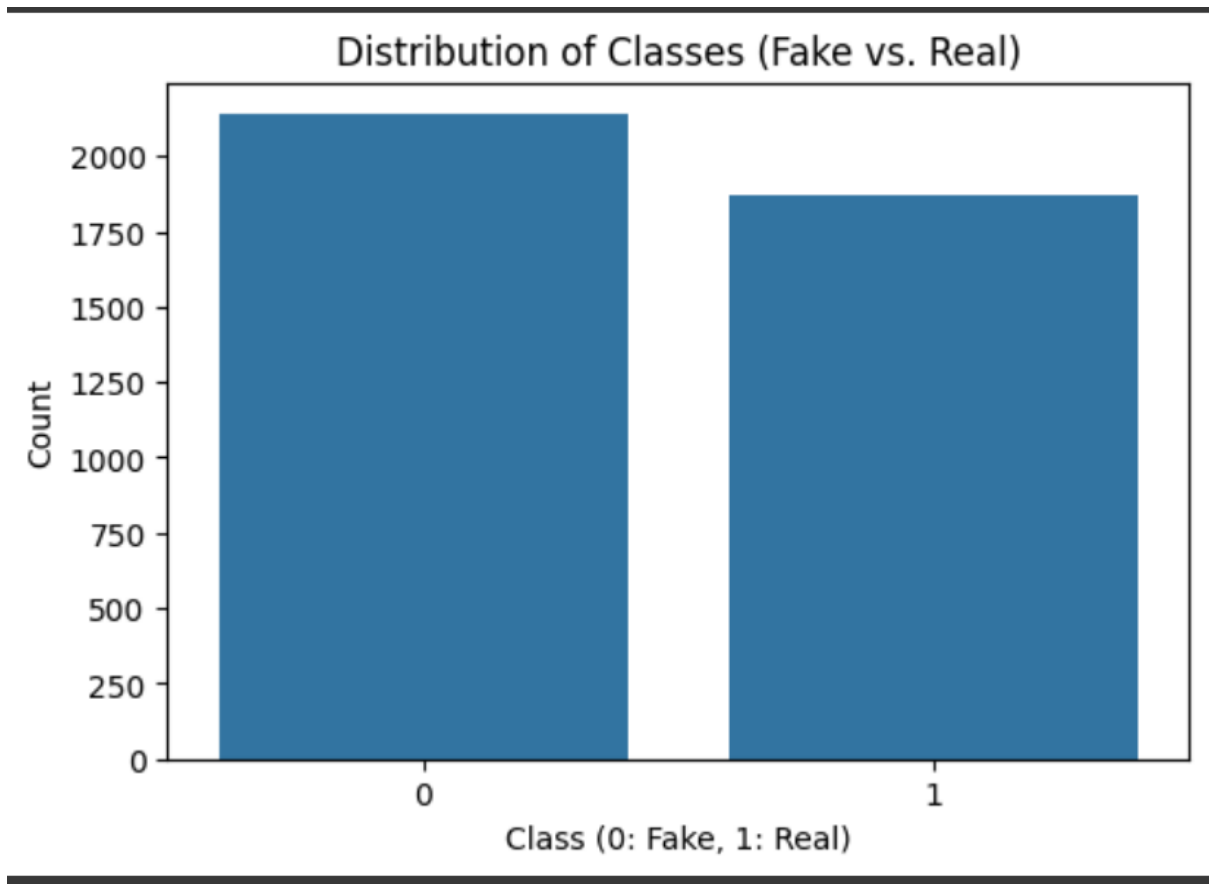


Figure 5.1 Distribution of Classes (Fake/Real)

The bar graph shows the distribution of classes in the dataset, with more instances of fake news (class 0) compared to real news (class 1).

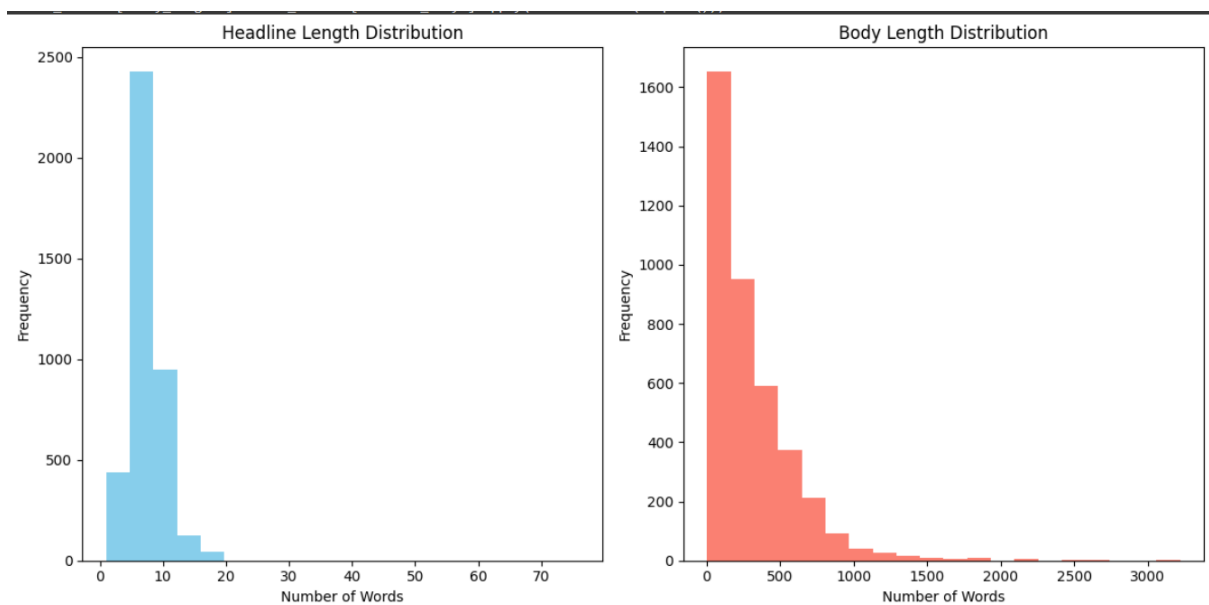
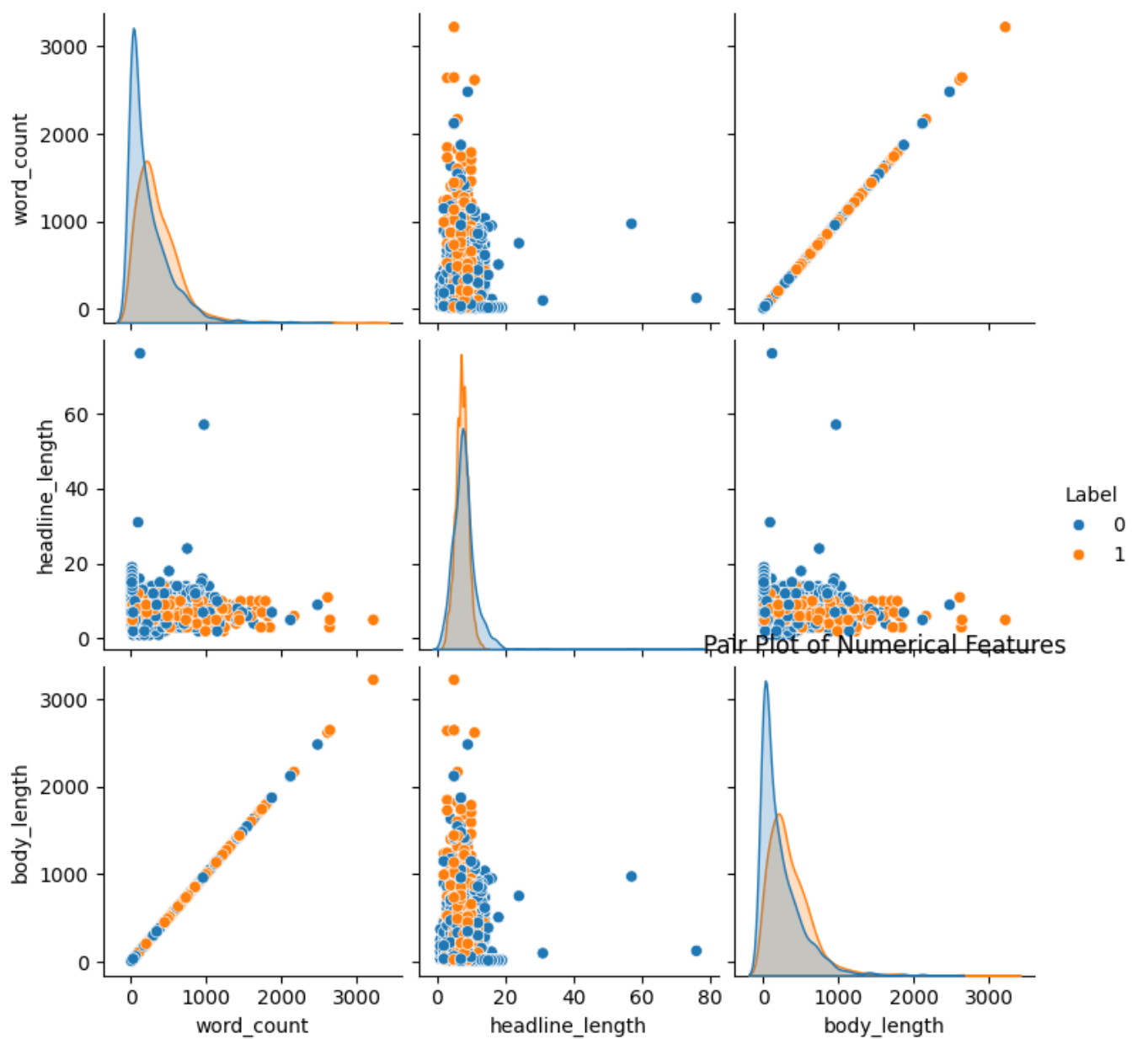


Figure 5.2 Distribution of Word Count in Cleaned Headline and Cleaned Body

The set of graphs illustrates the distributions of headline and body lengths, showing that most headlines contain fewer than 10 words, while the body texts typically have fewer than 500 words.



This scatter plot matrix shows relationships between the numerical features, such as word count, headline length, and body length. Data points are color-coded by the label (real or fake news), indicating some separation between the classes based on feature combinations.

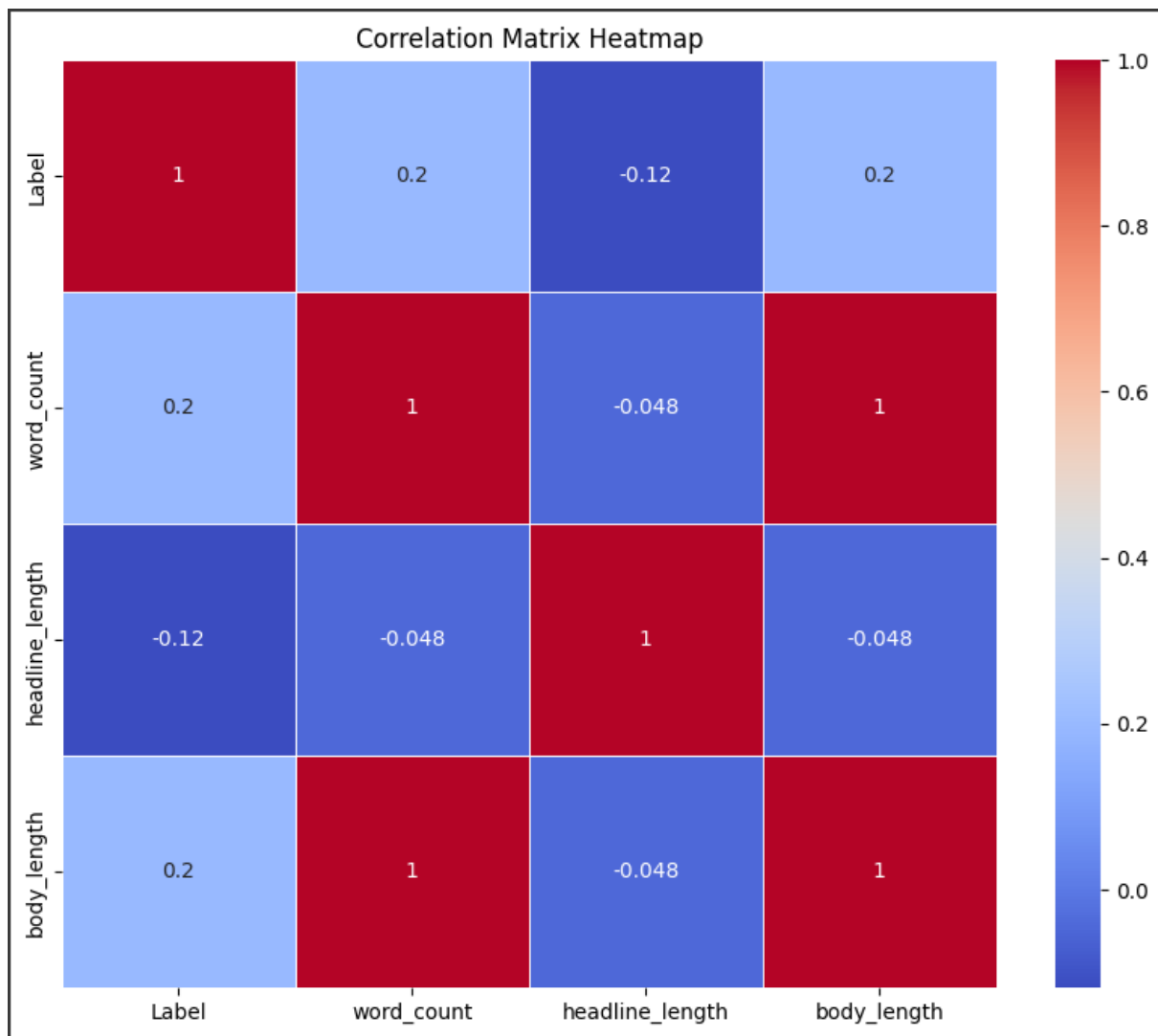


Figure 5.4 Correlation Matrix Heatmap

This heatmap displays the correlation between numerical features and the label. The highest correlation is observed between word count, body length, and the label, while headline length shows a weaker correlation.

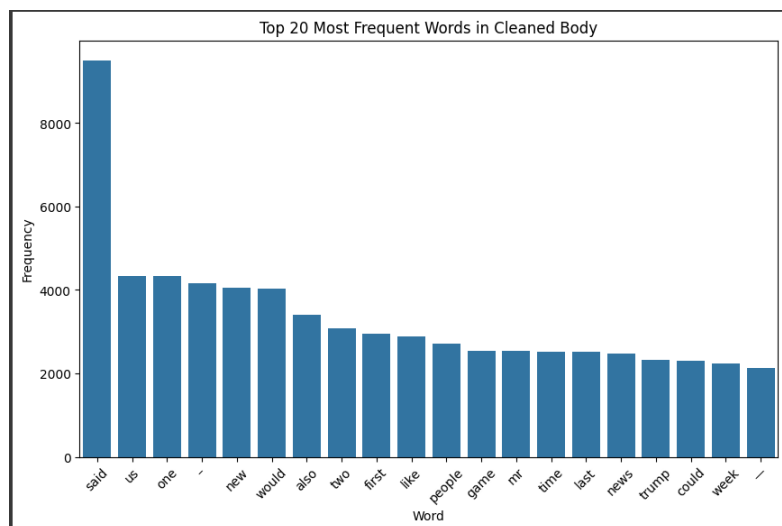


Figure 5.5 Top 20 Most Frequent Words in Cleaned Body

A bar chart highlighting the most common words in the cleaned body text of the news articles. Words like "said," "us," and "one" dominate, giving an indication of commonly used terms in the dataset after preprocessing.

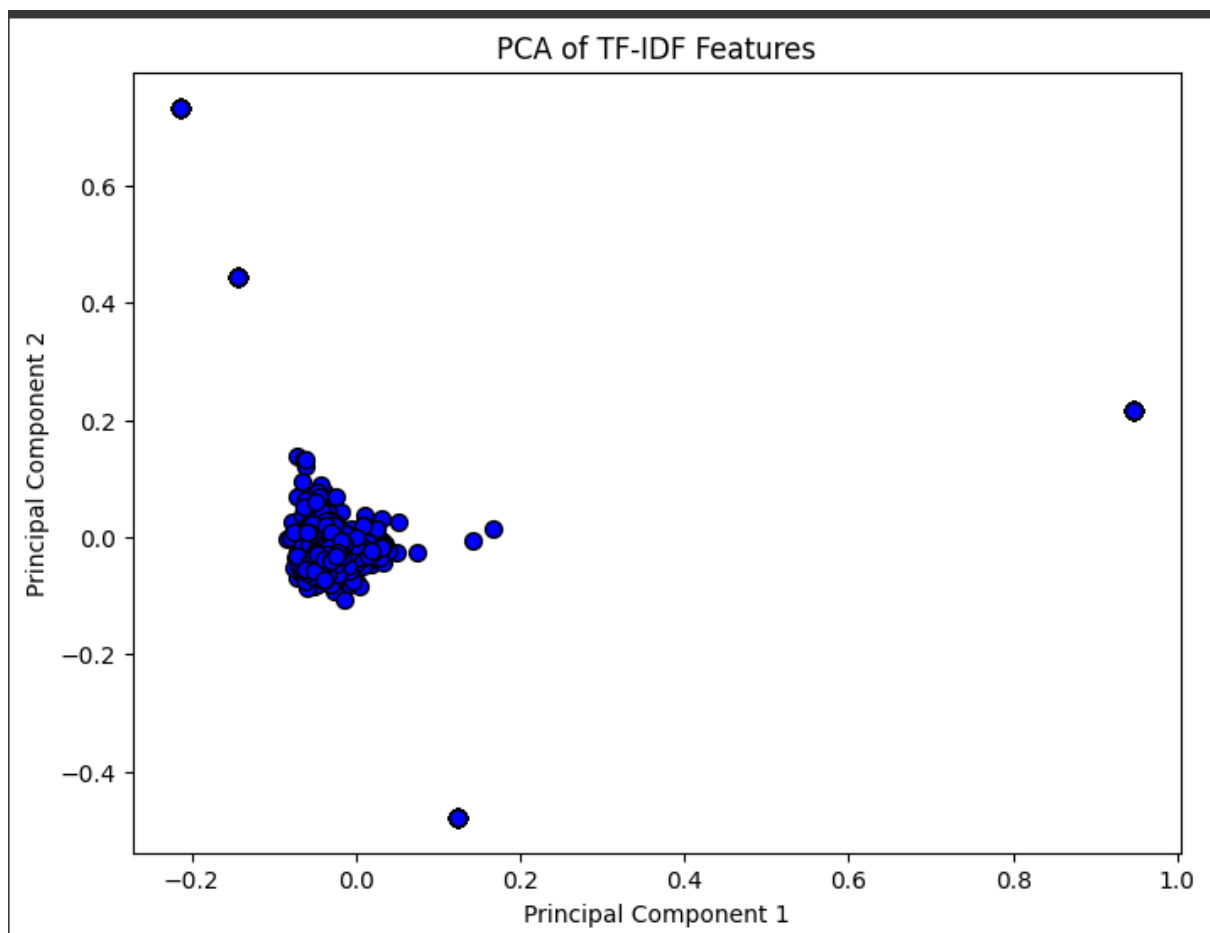


Figure 5.6 PCA of TF – IDF Features



This is a Principal Component Analysis (PCA) plot visualizing TF-IDF features. Most data points are clustered near the origin, suggesting the primary variance in the dataset is concentrated there, while a few points are spread out. PCA reduces high-dimensional TF-IDF features into two components for easier visualization.

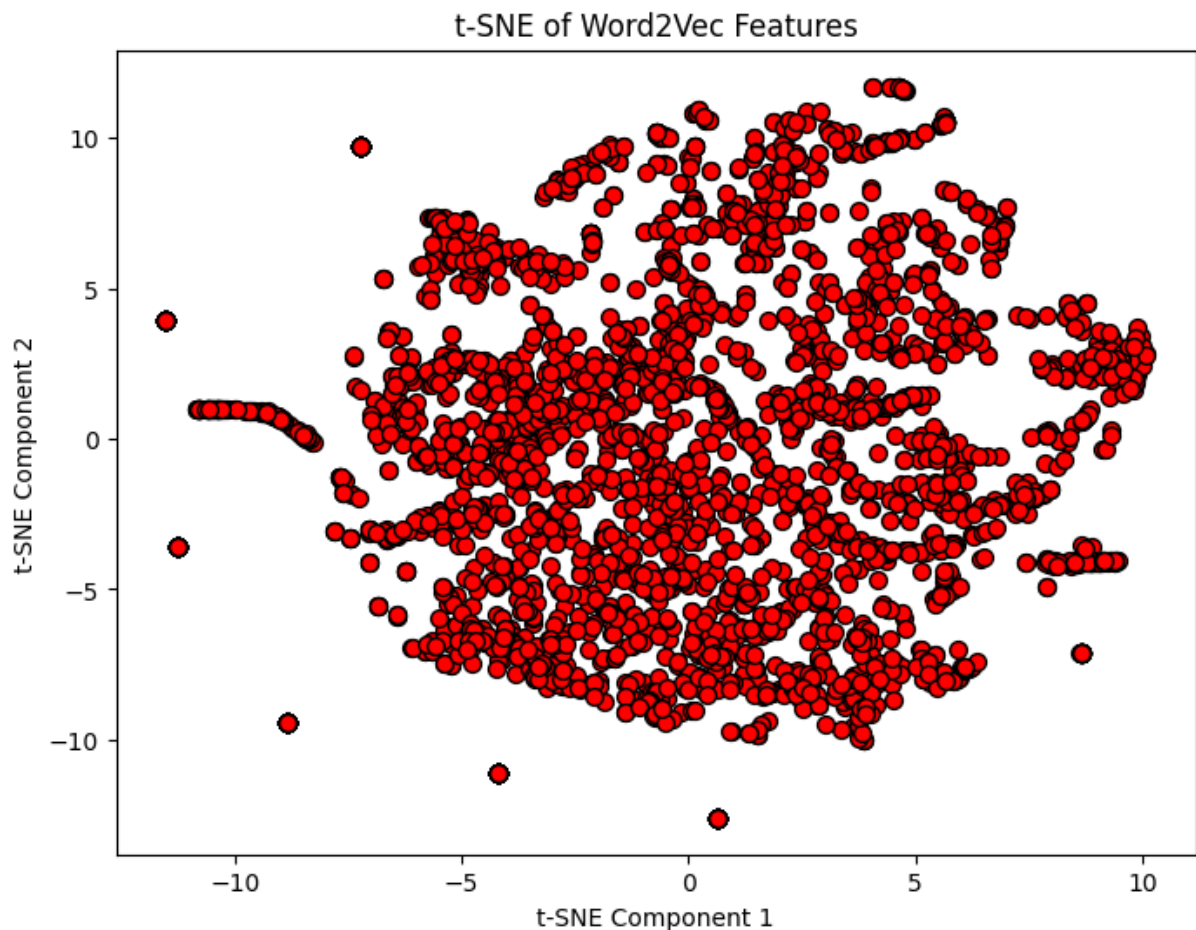


Figure 5.7 t- SNE of Word2Vec Features

This t-SNE plot shows a two-dimensional representation of Word2Vec features. Unlike the PCA, the points are more scattered, indicating complex relationships in the word embeddings. t-SNE (t-distributed Stochastic Neighbor Embedding) is often used for high-dimensional data visualization, capturing non-linear relationships between words.

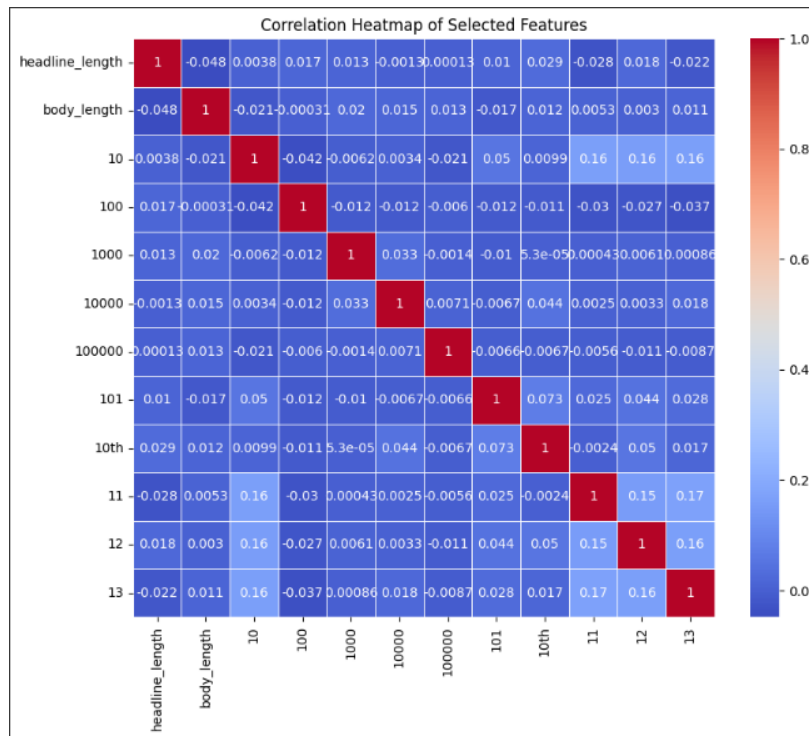


Figure 5.8 Correlation Heatmap of Selected Features

This heatmap shows the correlation matrix between different features. Strong positive correlations (in red) are found along the diagonal (as expected), while the rest show mostly weak correlations (closer to blue). The heatmap helps identify the relationships and dependencies between features such as "headline\_length" and "body\_length".

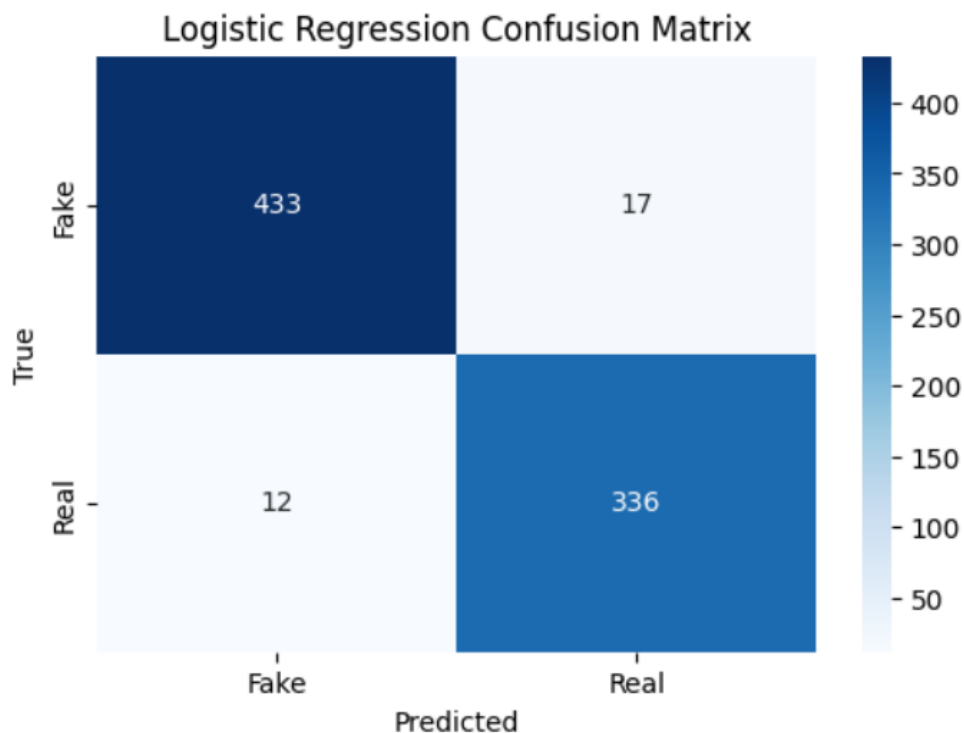


Figure 5.9 Confusion Matrix of Logistic Regression

This confusion matrix represents the performance of a Logistic Regression model in predicting two classes: Fake and Real. The matrix shows that the model correctly identified 433 instances of Fake and 336 instances of Real. However, it made some misclassifications, with 17 Real instances being wrongly classified as Fake (false negatives), and 12 Fake instances being classified as Real (false positives). Overall, while the model demonstrates good performance, it has slightly more errors compared to the SVM model (previous confusion matrix), particularly in predicting Real cases as Fake. The darker color shades in the matrix highlight areas where the model performed best, indicating higher numbers of correct classifications.

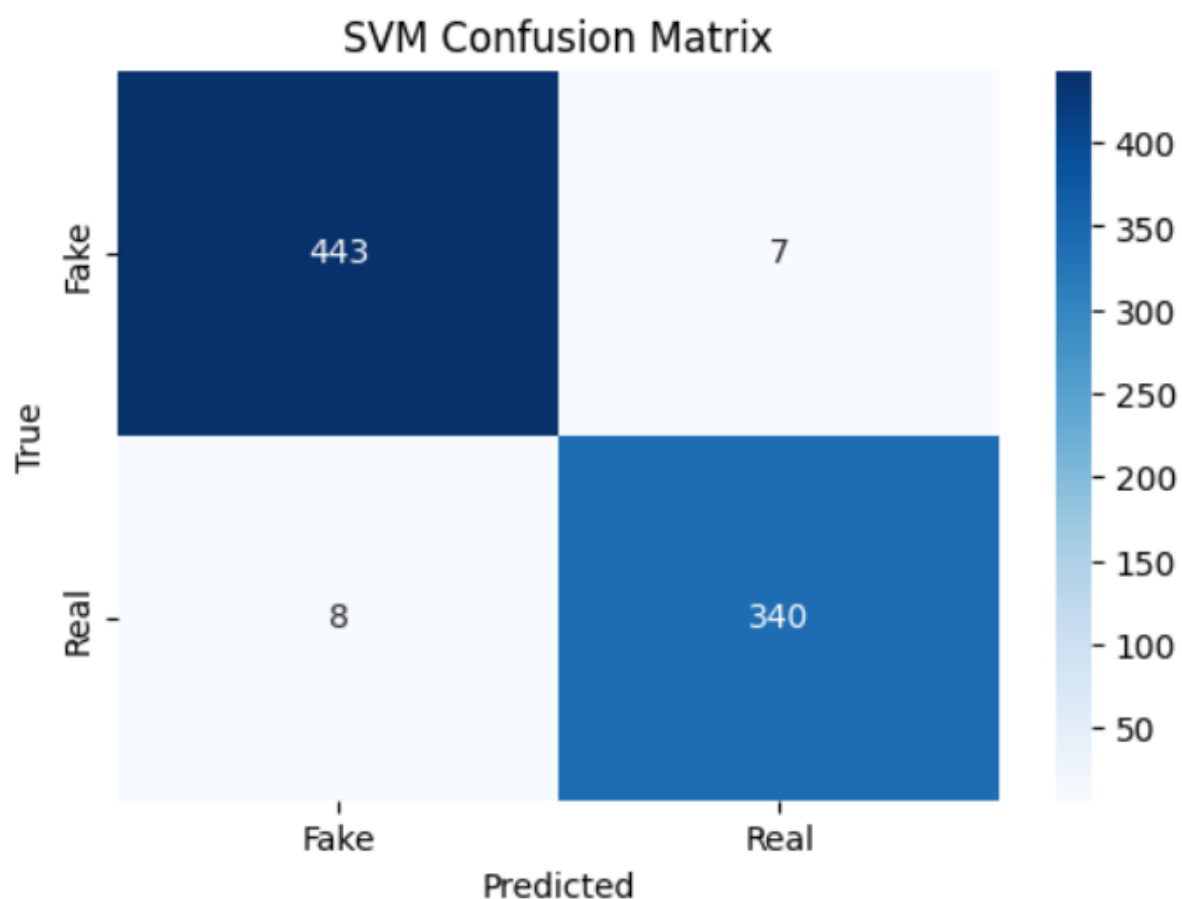


Figure 5.10 Confusion Matrix of SVM

This confusion matrix represents the performance of a Support Vector Machine (SVM) model in classifying instances as either Fake or Real. The matrix shows that the model correctly predicted 443 Fake instances and 340 Real instances, demonstrating a high level of accuracy. However, there were a few misclassifications: 7 Real instances were incorrectly predicted as Fake (false negatives), and 8 Fake instances were classified as Real (false positives). Despite these minor errors, the overall performance of the SVM is strong, with the color intensity in the matrix visually highlighting the higher number of correct predictions. The darker shades reflect the areas where the model's accuracy was highest, indicating that it is effective in distinguishing between the two classes.

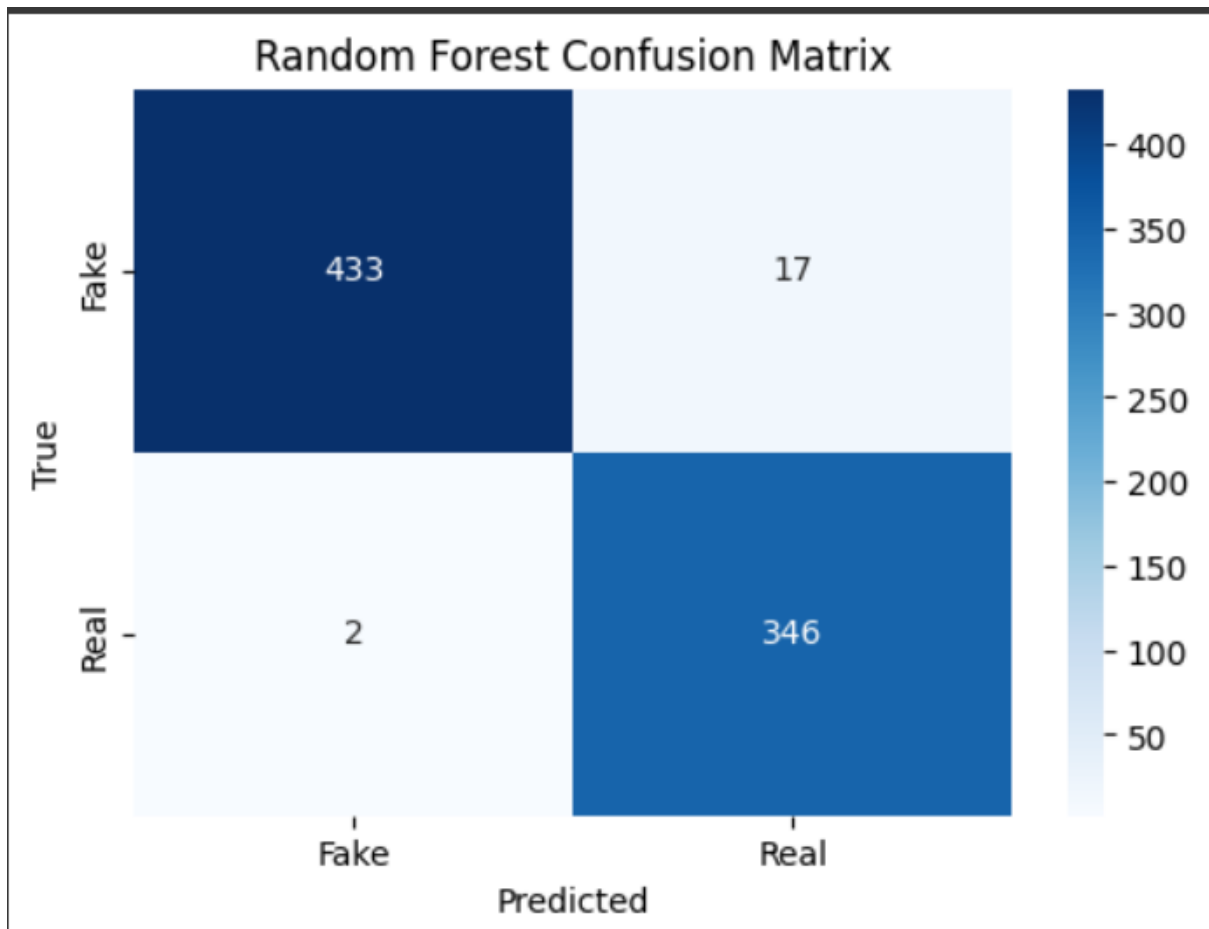


Figure 5.11 Confusion Matrix of Random Forest

This confusion matrix represents the performance of a Random Forest classification model in predicting Fake and Real instances. The model correctly identified 433 Fake instances and 346 Real instances, showing strong overall accuracy. However, it made some errors, misclassifying 17 Real instances as Fake (false negatives) and only 2 Fake instances as Real (false positives). Despite these few misclassifications, the model performs well, especially in identifying Real instances with a very low false positive rate. The color intensity in the matrix, with darker shades representing higher accuracy, highlights the model's effectiveness, particularly in correctly classifying both Fake and Real cases.

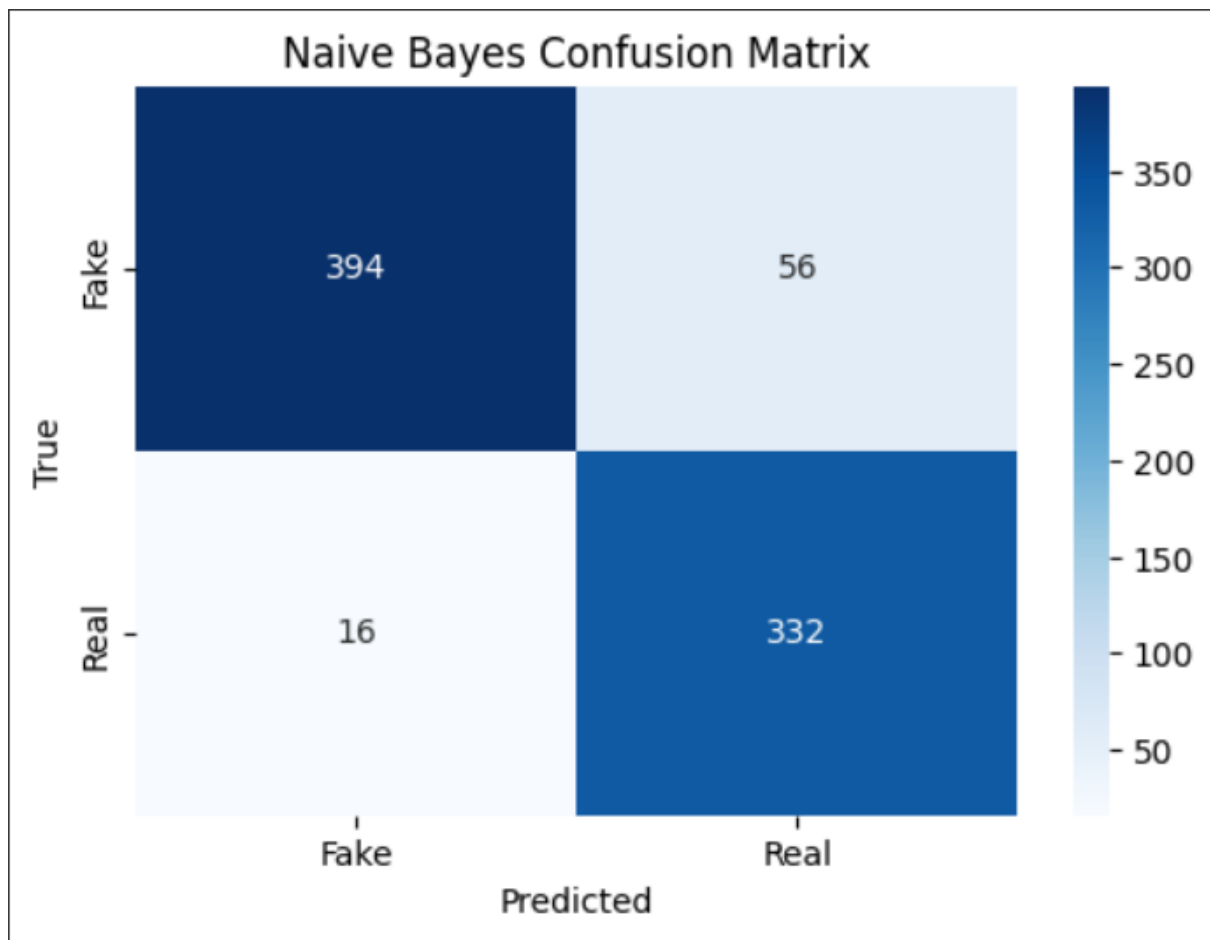


Figure 5.12 Confusion Matrix of Naïve Bayes

This confusion matrix represents the performance of a Naïve Bayes model in predicting Fake and Real instances. The model correctly identified 394 Fake and 332 Real instances. However, it misclassified 16 Real instances as Fake (false negatives) and 56 Fake instances as Real (false positives). The model shows good performance but with a higher false positive rate compared to the others. The color intensity in the matrix, with darker shades representing higher accuracy, highlights the model's effectiveness, particularly in correctly classifying both Fake and Real cases.

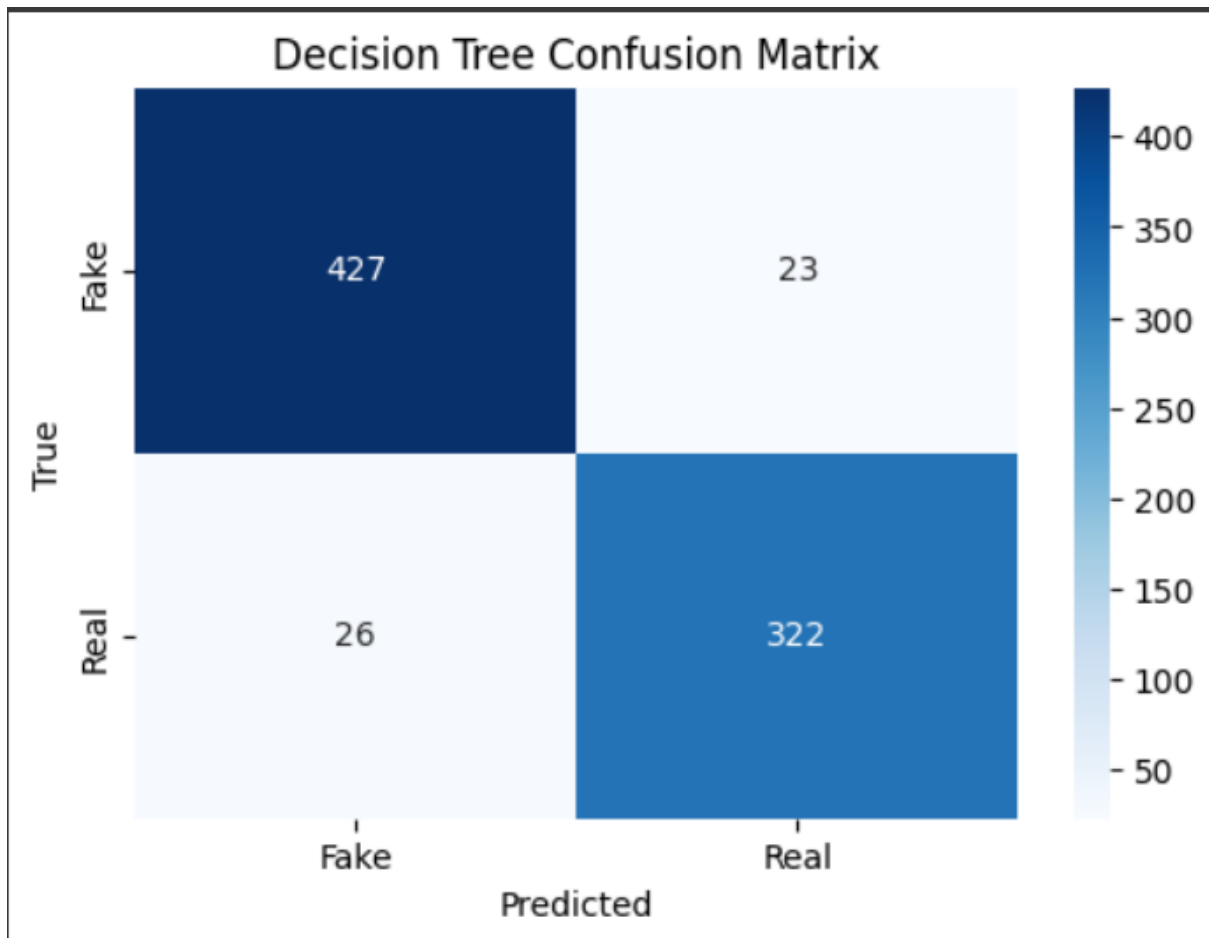


Figure 5.13 Confusion Matrix of Decision Tree

This confusion matrix represents the performance of a Decision Tree model in predicting Fake and Real instances. The model correctly identified 427 Fake and 322 Real instances. It misclassified 26 Real instances as Fake and 23 Fake instances as Real. This model shows balanced accuracy, with slightly fewer false positives than the Naive Bayes model. The color intensity in the matrix, with darker shades representing higher accuracy, highlights the model's effectiveness, particularly in correctly classifying both Fake and Real cases.

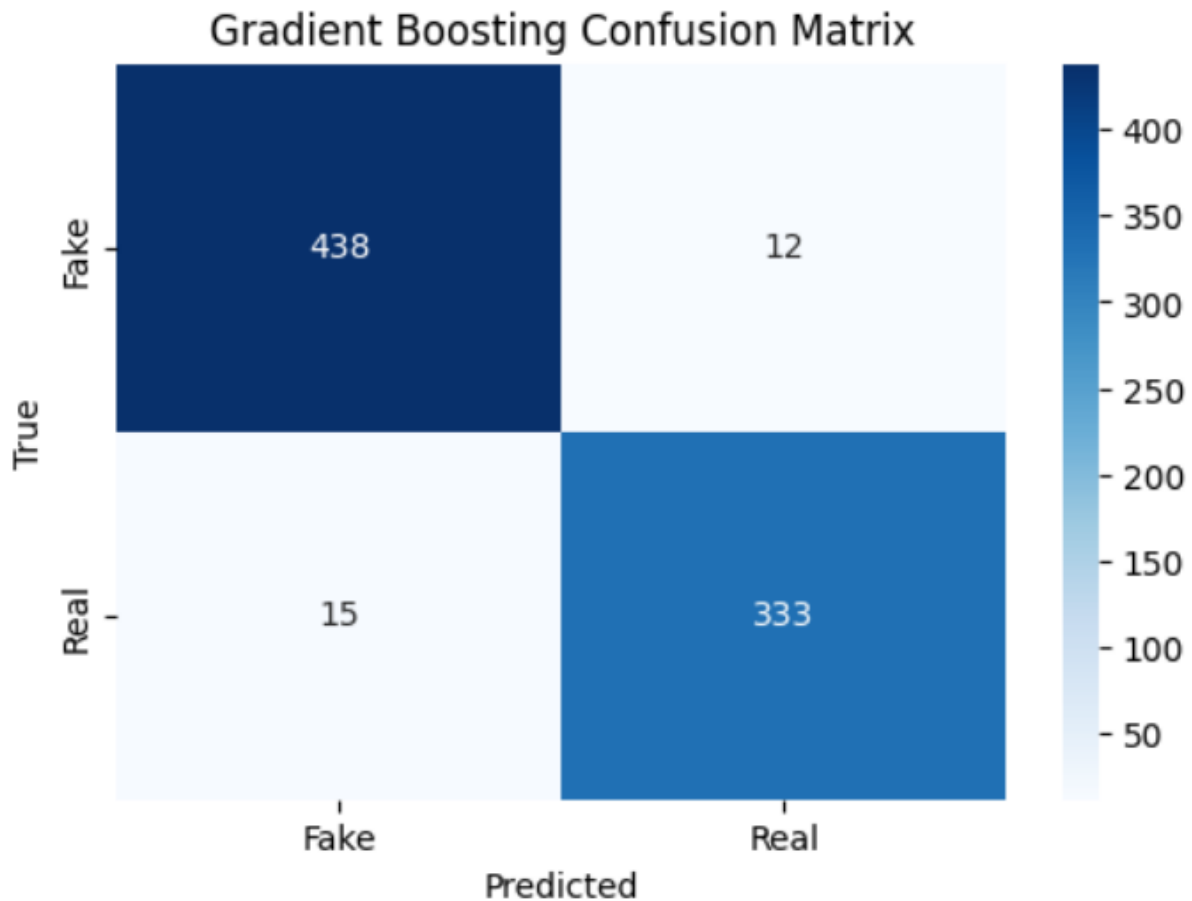


Figure 5.14 Confusion Matrix of GBM

This confusion matrix represents the performance of a Gradient Boosting model in predicting Fake and Real instances. The model had the strongest performance, correctly identifying 438 Fake and 333 Real instances. It made the fewest errors, misclassifying 15 Real instances as Fake and only 12 Fake instances as Real. The darker shades in the matrix highlight its high accuracy, particularly in correctly classifying Fake instances with a very low false positive rate. The color intensity in the matrix, with darker shades representing higher accuracy, highlights the model's effectiveness, particularly in correctly classifying both Fake and Real cases.

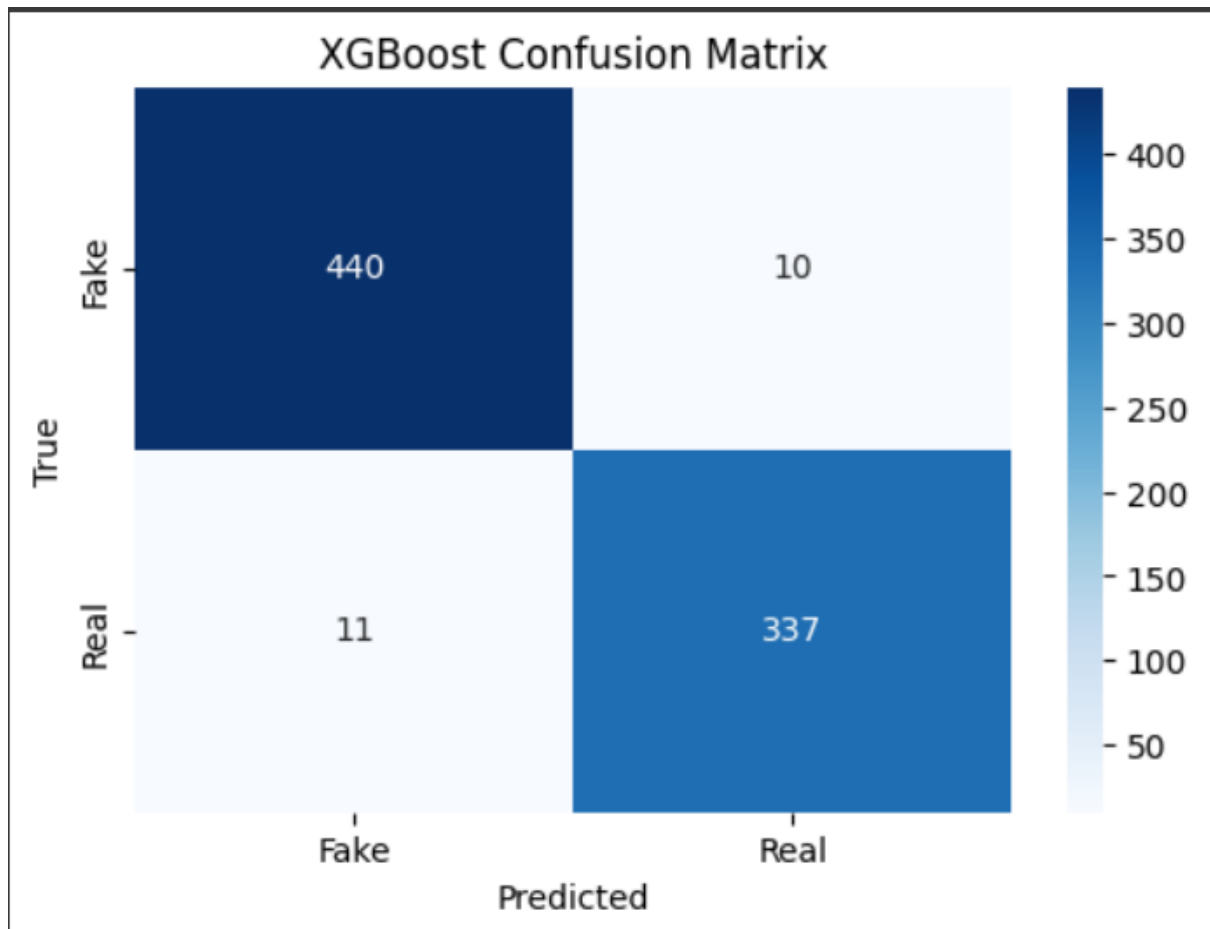


Figure 5.15 Confusion Matrix of XG Boost

The image represents the confusion matrix for the **XGBoost** model. The model correctly classified 440 fake instances and 337 real instances. However, it misclassified 10 real instances as fake and 11 fake instances as real. This indicates that XGBoost performs well, with only a small number of misclassifications, and a strong ability to distinguish between fake and real instances, with comparable performance in both categories.



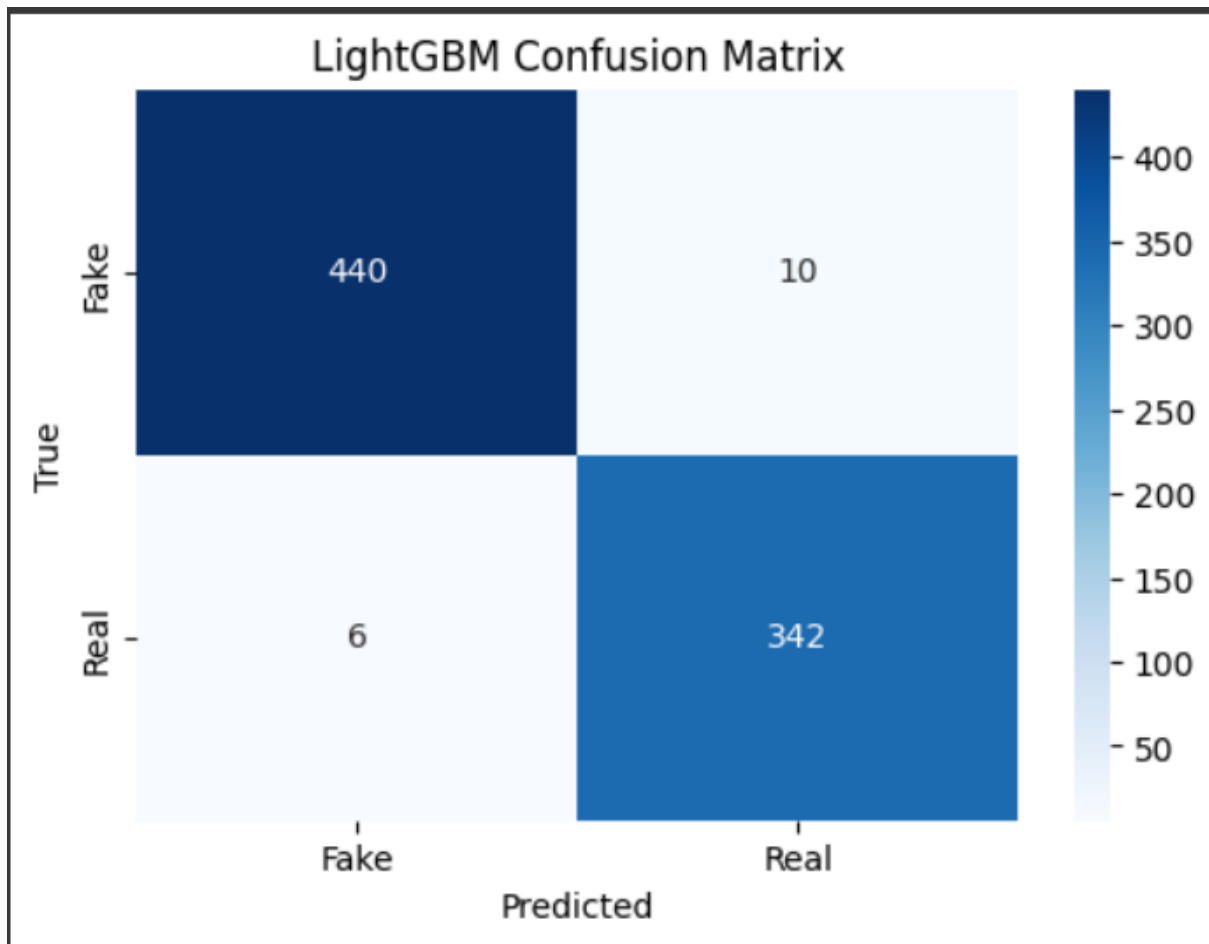


Figure 5.16 Confusion Matrix of Light GBM

The image shows the confusion matrix for the **LightGBM** model. Similar to XGBoost, it also correctly classified 440 fake instances, but it slightly outperforms XGBoost in terms of real instance predictions by correctly identifying 342 real instances. The model misclassified 10 real instances as fake (just like XGBoost) but performed better with fewer false real predictions, with only 6 fake instances being incorrectly classified as real. Overall, LightGBM demonstrates excellent performance with even fewer errors in detecting real instances.

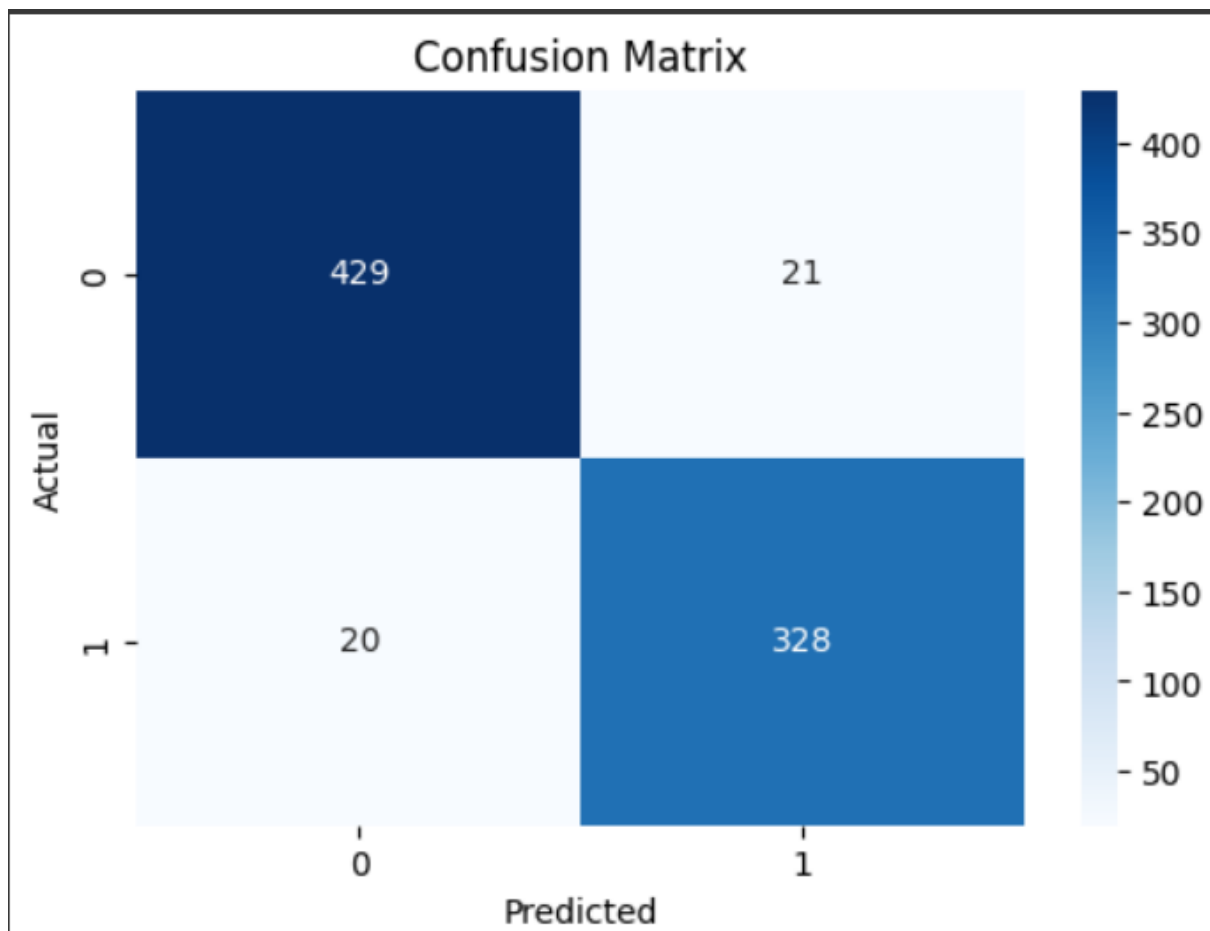


Figure 5.17 Confusion Matrix of SVD

The image displays the confusion matrix for the **SVD** model. While the model performs well, it shows a slightly lower accuracy compared to XGBoost and LightGBM. It correctly classified 429 fake instances and 328 real instances but made more misclassifications overall. It wrongly predicted 21 real instances as fake and 20 fake instances as real. This suggests that the SVD model, while still reliable, has a slightly higher error rate in distinguishing between fake and real instances compared to the XGBoost and LightGBM models.

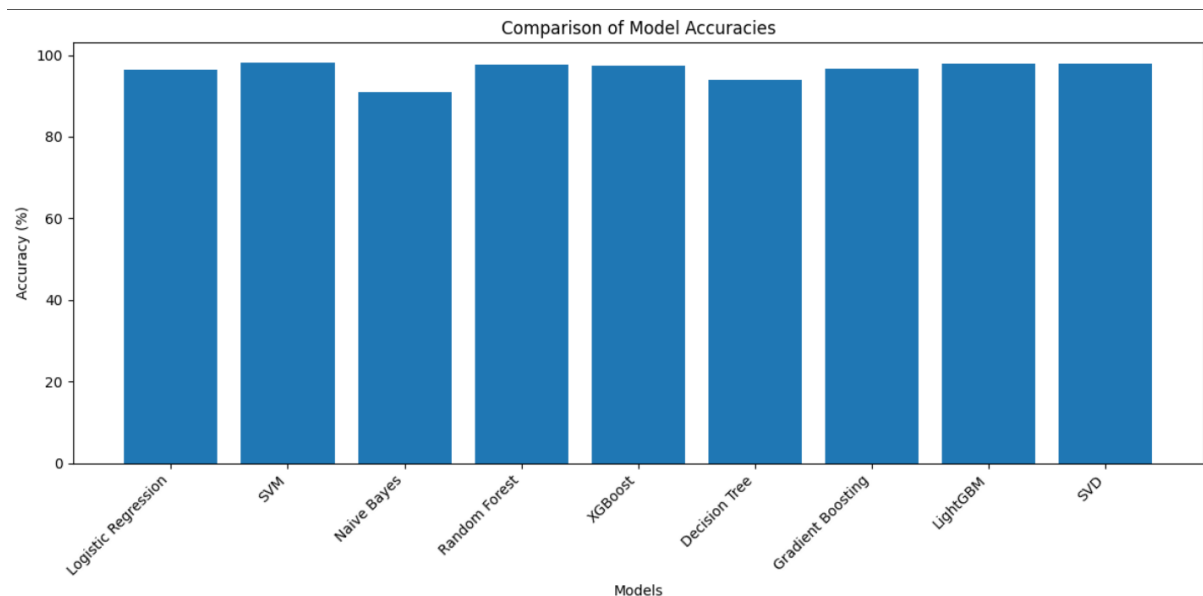


Figure 5.18 Comparison of Model Accuracies

The graph, titled "**Comparison of Model Accuracies**," presents the accuracy of various machine learning models. The models compared include Logistic Regression, SVM, Naive Bayes, Random Forest, XGBoost, Decision Tree, Gradient Boosting, LightGBM, and SVD. From the visual representation, it is evident that most models achieve a high level of accuracy, with values ranging from around 90% to 100%. Models such as SVM, XGBoost, and LightGBM show particularly strong accuracy, whereas Naive Bayes slightly trails behind the other models. Overall, the graph highlights that these models perform very well in terms of accuracy, making them competitive choices depending on the application.

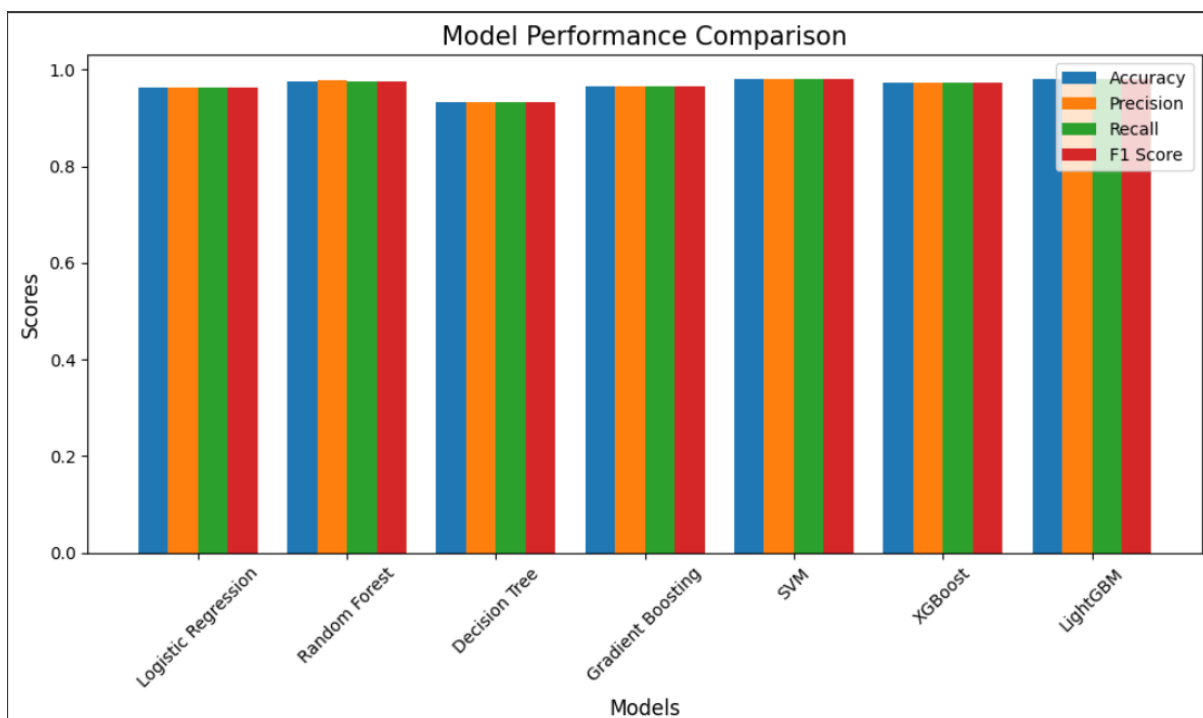


Figure 5.19 Model Performance Comparison

The second graph, titled "**Model Performance Comparison**," delves deeper into multiple performance metrics: Accuracy, Precision, Recall, and F1 Score. The same set of models is evaluated here, excluding Naive Bayes and SVD. The metrics are represented in bars of different colors: blue for Accuracy, orange for Precision, green for Recall, and red for F1 Score. Across the board, the models, including Logistic Regression, Random Forest, Decision Tree, Gradient Boosting, SVM, XGBoost, and LightGBM, show consistently high performance for all metrics. The scores hover around 0.9 to 1, indicating that the models are not only accurate but also demonstrate strong precision, recall, and balanced F1 scores, making them reliable across multiple evaluation criteria.

	precision	recall	f1-score	support
0	0.98	0.98	0.98	450
1	0.98	0.98	0.98	348
accuracy			0.98	798
macro avg	0.98	0.98	0.98	798
weighted avg	0.98	0.98	0.98	798

Figure 5.20 Classification metrics

The table shows the classification metrics for a binary classification model. Precision measures how many of the positive predictions were actually correct, recall measures how many of the actual positive cases were correctly predicted, and f1-score is the harmonic mean of precision and recall. The accuracy is the overall proportion of correct predictions. The macro average and weighted average consider the performance across both classes, with the weighted average taking into account the class imbalance.

## **CHAPTER-6**

### **Conclusion and Future Enhancement**

## 6.1 Conclusion

The Fake News Detection System (FDS) implemented in this project leverages machine learning techniques and advanced text feature extraction methods to distinguish between real and fake news. By combining hybrid feature extraction approaches, including TF-IDF, N-grams, and word embeddings like Word2Vec, we aimed to capture the nuanced linguistic patterns present in fake news articles. The integration of multiple machine learning models such as SVM, Logistic Regression, Random Forest, XGBoost, LightGBM, and others provided a comparative performance analysis, revealing SVM as the best-performing model with 98.12% accuracy.

The application of SVD (Singular Value Decomposition) for dimensionality reduction further refined the feature space, allowing us to train the models more efficiently while retaining key information from the text. This approach minimized overfitting and improved computational speed without significant loss in accuracy. Throughout this process, we visualized feature relationships, class distributions, and the importance of various features in making predictions, providing us with deeper insights into the nature of fake news.

Our results show that sophisticated models such as SVM and LightGBM are highly effective at identifying fake news when paired with hybrid feature extraction techniques. The confusion matrix, classification reports, and accuracy scores demonstrate the robustness of these models in identifying subtle patterns between real and fake articles. This system is a promising step towards combating misinformation on digital platforms, ensuring that false information is flagged before it reaches larger audiences. However, the variability of fake news formats and evolving nature of disinformation tactics pose continuous challenges, requiring future improvements in model adaptability.

## 6.2 Future Enhancement

While the current Fake News Detection System achieves commendable results, several future enhancements can elevate its performance and versatility. Firstly, the incorporation of deep learning models such as Bidirectional Encoder Representations from Transformers (BERT) or Long Short-Term Memory (LSTM) networks can significantly improve the system's ability to understand context and semantics in textual data. Unlike traditional machine

learning models, these neural networks can capture the deeper structure of language and are more resilient to the evolving nature of disinformation tactics.

Incorporating real-time data streams from social media platforms or news websites is another potential enhancement. This will enable the system to continuously learn and adapt to new types of fake news, rather than relying on a static, pre-collected dataset. To ensure scalability, cloud-based implementations using platforms like AWS or Google Cloud could help process large-scale data with faster computing power, allowing the system to handle global-scale information streams in real-time.

Another area of improvement is multilingual fake news detection. Fake news is a global problem, and current models might be limited to specific languages (e.g., English). Extending the system to support multiple languages using translation models and language-specific feature extraction techniques would make it more universally applicable. Additionally, expanding feature engineering techniques beyond text (e.g., incorporating metadata, user engagement, and network structure analysis) could provide more comprehensive signals to detect disinformation.

Finally, integrating explainable AI (XAI) techniques can improve trust and transparency in fake news detection. Allowing users to understand why the system flagged an article as fake will increase confidence in its reliability. Moreover, addressing ethical concerns related to freedom of speech and potential biases in model predictions will be critical to ensuring responsible use of this technology in diverse socio-political contexts. These enhancements will create a more adaptive, scalable, and trustworthy fake news detection system capable of handling the complex and dynamic nature of global information ecosystems.

## **CHAPTER-7**

### **References**



## 7. References

- [1] Phan, H.T., Nguyen, N.T., & Hwang, D. (2023). Fake News Detection: A Survey of Graph Neural Network Methods. *Applied Soft Computing*, 139, 110235.
- [2] Seddari, N., Derhab, A., Belaoued, M., Halboob, W., Al-Muhtadi, J., & Bouras, A. (2022). A Hybrid Linguistic and Knowledge-Based Analysis Approach for Fake News Detection on Social Media. *IEEE Access*, 10, 62097-62110.
- [3] Kozik, R., Pawlicka, A., Pawlicki, M., Choraś, M., Mazurczyk, W., & Cabaj, K. (2024). A Meta-Analysis of State-of-the-Art Automated Fake News Detection Methods. *IEEE Transactions on Computational Social Systems*, 11(4), 5219-5230.
- [4] Djenouri, Y., Belhadi, A., Srivastava, G., & Lin, J. C.-W. (2023). Advanced Pattern-Mining System for Fake News Analysis. *IEEE Transactions on Computational Social Systems*, 10(6), 2949-2959.
- [5] Park, M., & Chai, S. (2023). Constructing a User-Centered Fake News Detection Model by Using Classification Algorithms in Machine Learning Techniques. *IEEE Access*, 11, 71517-71527.
- [6] Elsaeed, E., Ouda, O., Elmogy, M. M., & Atwan, A. (2021). Detecting Fake News in Social Media Using Voting Classifier. *IEEE Access*, 9, 161909-161923.
- [7] Almarashy, A. H. J., Feizi-Derakhshi, M.-R., & Salehpour, P. (2023). Enhancing Fake News Detection by Multi-Feature Classification. *IEEE Access*, 11, 139601-139613.
- [8] Mannan, I., & Nova, S. N. (2023). An Empirical Study on Theories of Sentiment Analysis in Relation to Fake News Detection. *IEEE International Conference on Information and Communication Technology for Sustainable Development (ICT4SD)*, 79-83.
- [9] Nawaz, M. Z., Nawaz, M. S., Fournier-Viger, P., & He, Y. (2024). Analysis and Classification of Fake News Using Sequential Pattern Mining. *Big Data Mining and*

*Analytics*, 7(3), 942-963.

[10] Xie, B., Ma, X., Wu, J., Yang, J., & Fan, H. (2024). Knowledge Graph Enhanced Heterogeneous Graph Neural Network for Fake News Detection. *IEEE Transactions on Consumer Electronics*, 70(1), 2826-2838.

[11] Ravish, Katarya, R., Dahiya, D., & Checker, S. (2022). Fake News Detection System Using Featured-Based Optimized MSVM Classification. *IEEE Access*, 10, 113184-113197.

[12] Ahmed, H., Traore, I., Saad, S., & Mamun, M. (2024). Effect of text augmentation and adversarial training on fake news detection. *IEEE Transactions on Computational Social Systems*, 11(4), 4775-4787.

[13] Choudhury, D., & Acharjee, T. (2022). A novel approach to fake news detection in social networks using genetic algorithm applying machine learning classifiers. *Multimedia Tools and Applications*, 82, 9029–9045.

[14] Choudhury, D., & Acharjee, T. (2022). A novel approach to fake news detection in social networks using genetic algorithm applying machine learning classifiers. *Multimedia Tools and Applications*, 82, 9029–9045.

[15] Thakur, A., Shinde, S., Patil, T., Gaud, B., & Babanne, V. (2020). **MYTHYA: Fake News Detector, Real Time News Extractor and Classifier**. Proceedings of the Fourth International Conference on Trends in Electronics and Informatics (ICOEI), IEEE. doi:10.1109/ICOEI.2020.9237078.