MISINFO DETECTION NLP FOR FAKE NEWS DETECTION

A PROJECT REPORT

Submitted by

SANGAMESHWARAN S	720921104087
TAHIRA BEGUM S	720921104108
THIRISHA R	720921104306
SURE VENKATA NAGA SAITEJA	720921104105

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

JCT COLLEGE OF ENGINEERING AND TECHNOLOGY
COIMBATORE – 641105

ANNA UNIVERSITY :: CHENNAI

MAY 2025

ANNA UNIVERSITY :: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that the project report **"MISINFO DETECTION NLP FOR FAKE NEWS DETECTION"** is the Bonafide work of **"SANGAMESHWARAN S (720921104087), TAHIRA BEGUM S (720921104108) SURE VENKATA NAGA SAITEJA (720921104105) , THIRISHA R, (720921104306)"** who carried out project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr. S. ZULAIKHA BEEVI. M.Tech., Ph.D.,	Prof. Mrs.S.Rajeswari(AP/CSE)
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Department of Computer Science And Engineering	Department of Computer Science And Engineering
JCT College of Engineering and Technology,	JCT College of Engineering and Technology,
Pichanur,	Pichanur,
Coimbatore - 641 105	Coimbatore – 641 105

Submitted for the Project Viva Voce Examination held on ______________________________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Sentiment analysis of social media content has gained considerable attention in recent years due to its ability to analyze the opinions, emotions, and feedback of users. This study focuses on performing sentiment analysis on Fake News posts using machine learning techniques. By leveraging natural language processing (NLP) and various machine learning algorithms, this research aims to classify the sentiments expressed in Fake News posts into categories such as positive, negative, or neutral. The study explores different machine learning models such as Logistic Regression, Support Vector Machines (SVM), BERT and Naive Bayes to identify the best-performing algorithm for sentiment classification. Additionally, this paper discusses data preprocessing steps, including text cleaning, tokenization, and feature extraction, as well as the importance of model evaluation metrics like accuracy, precision, recall, and F1-score in assessing the performance of the sentiment analysis models. The results show that machine learning can be a valuable tool for analyzing user-generated content on Fake News and extracting meaningful insights about public opinions.

INSTITUTION VISION

- To emerge as a Premier Institute for developing industry ready engineers with competency, initiative and character to meet the challenges in global environment.

INSTITUTION MISSION

- To impart state-of-the-art engineering and professional education through strong theoretical basics and hands on training to students in their choice of field.
- To serve our students by teaching them leadership, entrepreneurship, teamwork, values, quality, ethics and respect for others.
- To provide opportunities for long-term interaction with academia and industry.
- To create new knowledge through innovation and research

DEPARTMENT VISION

- To become a center of excellence in Computer Science and Engineering by nurturing technically competent, innovative, and ethically responsible professionals who are prepared to meet global industry challenges and contribute to societal progress.

DEPARTMENT MISSION

- To impart comprehensive education in Computer Science and Engineering with strong theoretical foundations and practical exposure.
- To cultivate innovation, research aptitude, and a spirit of entrepreneurship through cutting- edge technologies and project-based learning.
- To enhance industry readiness through internships, certifications, collaborative projects, and active engagement with academia and industry.
- To instill ethical values, social responsibility, and leadership qualities in students to contribute meaningfully to global and local communities.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

PEO 1: Graduates will apply foundational knowledge and modern tools of Computer Science and Engineering to develop innovative solutions, pursue higher studies, and adapt to evolving technologies in professional environments.

PEO 2: Graduates will demonstrate ethical behavior, leadership, and social responsibility while addressing technological and societal challenges, and will leverage their technical knowledge and creativity to drive innovation, initiate start-ups, or make meaningful contributions in industry or academia.

PEO 3: Graduates will demonstrate a commitment to continuous learning, acquire relevant certifications, and collaborate effectively with multidisciplinary teams in diverse professional settings.

PROGRAM OUTCOMES (PO'S)

1. **Engineering Knowledge**

   Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem Analysis**

   Identify, formulate, review research literature, and analyze complex engineering problems using principles of mathematics, natural sciences, and engineering sciences.

3. **Design/Development of Solutions**

   Design solutions for complex engineering problems and design system components or processes that meet specified needs with consideration for public health and safety, cultural, societal, and environmental concerns.

4. **Conduct Investigations of Complex Problems**

   Use research-based knowledge and methods including design of experiments, analysis and interpretation of data, and synthesis of information to provide valid conclusions.

5. **Modern Tool Usage**

   Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools for complex engineering activities, with an understanding of their limitations.

6. **The Engineer and Society**

   Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal, and cultural issues relevant to professional engineering practice.

7. **Environment and Sustainability**

    Understand the impact of professional engineering solutions in societal and environmental contexts and demonstrate knowledge of sustainable development.

8. **Ethics**

    Apply ethical principles and commit to professional ethics and responsibilities and norms of engineering practice.

9. **Individual and Team Work**

    Function effectively as an individual and as a member or leader in diverse teams and in multidisciplinary settings.

10. **Communication**

    Communicate effectively on complex engineering activities with the engineering community and with society at large, including writing reports, making presentations, and giving/receiving clear instructions.

11. **Project Management and Finance**

    Demonstrate knowledge and understanding of engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Lifelong Learning**

    Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the context of technological change.

PROGRAMME SPECIFIC OUTCOMES (PSO)

PSO1: Graduates will be able to apply the fundamentals of computer science, including data structures, algorithms, database systems, computer networks and computer architecture, to develop efficient computing solutions.

PSO2: Graduates will be able to explore and apply emerging technologies such as Artificial Intelligence, Machine Learning, Cloud Computing, IoT, and Cyber security to solve real-world problems and pursue research.

TABLE OF CONTENT

LIST OF FIGURES

ACKNOWLEDGEMENT

It is with great respect and gratitude that I take this opportunity to express my sincere thanks to all supported and guided me throughout the successful completion of this project.

First and foremost, I am deeply grateful to our esteemed Founder **Thiru. S. A. Subramanian**, for his vision and dedication in establishing this institution, which has provided us with a strong foundation for learning and growth.

I extend my heartfelt thanks to our respected **Chairman Thiru. R. Arulselvan**, and **Vice-Chairman, Thiru. R. Gautaman**, for their continued encouragement and leadership that inspire academic excellence.

I also wish to express my sincere appreciation to our **Secretary Thiru. R. Durga Shankar**, for his unwavering support and contribution toward the smooth functioning of our academic endeavors.

My deep gratitude goes to our **Principal Dr. S. Manoharan B.E., M.E., Ph.D.**, for providing the resources, motivation, and environment conducive to learning and innovation.

I am immensely thankful to our **Head of the Department Dr. S. Zulaikha Beevi B.E., M.Tech., Ph.D.,** for her constant guidance, encouragement, and valuable insights throughout the course of this project.

A special note of thanks to my Supervisor, **Prof. Mrs.S.Rajeswari(AP/CSE).,** whose expert guidance, patience, and constructive feedback have been instrumental in the successful execution of this project.

I would also like to extend my thanks to all the **faculty members of the department** and the **supporting staff** for their assistance, timely help, and encouragement at every stage.

Finally, I express my heartfelt thanks to everyone who has contributed, directly or indirectly to the successful completion of this project work.

SANGAMESHWARAN S

TAHIRA BEGUM S

THIRISHA R

SURE VENKATA NAGA SAITEJA

CHAPTER 1

INTRODUCTION

With the rapid growth of social media platforms, the need to analyze vast amounts of data has become crucial for businesses, researchers, and social scientists. Facebook, being one of the largest social media platforms globally, generates an enormous amount of data through posts, comments, and interactions. Understanding the sentiments behind these posts provides valuable insights into public opinion, user engagement, and brand perception. This makes sentiment analysis a significant area of study for extracting useful information from the text-based content that Fake News users generate. Sentiment analysis helps to categorize text into positive, negative, or neutral sentiments, providing an understanding of how users feel about specific topics, products, services, or events.

In this context, machine learning (ML) plays a pivotal role in automating the sentiment analysis process. Traditional approaches like rule-based systems have limitations, particularly when dealing with large datasets. Machine learning models, however, can learn from data and adapt to various linguistic nuances, slang, and the ever-evolving nature of social media language. This paper delves into the potential of using machine learning for Fake News sentiment analysis by applying advanced text classification algorithms to classify the sentiments of user-generated content.

This study explores the steps involved in developing a robust sentiment analysis system, starting with data collection and preprocessing. Text data from Facebook, which may contain unstructured content, requires significant cleaning and transformation to be used effectively by machine learning models. Tokenization, stemming, and removing stop words are some of the preprocessing steps that are vital for preparing the text data for analysis. After preprocessing, feature extraction techniques such as TF-IDF (Term Frequency-Inverse Document Frequency) are applied to convert the textual data into a numerical format that can be understood by machine learning algorithms.

Additionally, the research evaluates various machine learning models, ranging from simpler models like Logistic Regression to more complex ones like Support Vector Machines (SVM). Each model's performance is assessed using standard evaluation metrics such as accuracy, precision, recall, and F1-score. By comparing the results of different algorithms, the study aims to identify which machine learning model provides the most accurate sentiment classification on Fake News posts. The insights gained from this analysis are expected to contribute to the ongoing research in the field of sentiment analysis and machine learning, particularly in the context of social media analytics.

1.1 GENERAL INTRODUCTION

In the era of digital communication, social media platforms have become a primary channel for people to express opinions, share experiences, and engage with the world around them. Among these platforms, Fake News stands out due to its massive global user base, with millions of daily active users sharing posts, comments, and reactions. These user-generated contents provide a rich source of data, which, if analyzed correctly, can offer insights into public opinion, consumer sentiment, and overall social trends. However, with the immense volume of posts being generated every second, it becomes essential to adopt automated techniques to efficiently process and analyze the data. One such technique is sentiment analysis, which can classify textual content into different sentiment categories such as positive, negative, or neutral.

Sentiment analysis leverages natural language processing (NLP) and machine learning (ML) techniques to identify and extract subjective information from text. By applying sentiment analysis to Fake News posts, businesses, researchers, and social media analysts can gain a deeper understanding of public opinion on various topics, ranging from political events to product reviews. The increasing reliance on machine learning for sentiment analysis has led to the development of powerful models capable of recognizing subtle nuances in language, such as irony, sarcasm, and context-based meaning. Thus, this field of research plays a crucial role in the emerging area of social media analytics.

While sentiment analysis has proven to be a valuable tool, its application to social media platforms like Fake News introduces specific challenges. The informal, conversational nature of posts and the presence of various dialects, abbreviations, and slang make it difficult for traditional text analysis methods to be effective. Furthermore, the vast scale of data on Fake News requires sophisticated models that can handle large volumes of text data while maintaining accuracy and efficiency. As a result, researchers have turned to machine learning algorithms that can learn from data, recognize patterns, and generalize across different types of social media content.

The goal of this project is to utilize machine learning techniques to perform sentiment analysis on Fake News posts. By developing and evaluating various machine learning models, this study seeks to determine which algorithm performs best in terms of classifying the sentiment of Fake News posts. Moreover, it aims to offer a methodology for extracting valuable insights from social media content, which can be used in areas such as market research, brand monitoring, and social sentiment tracking.

1.2 PROJECT OBJECTIVE

- The primary objective of this project is to design and implement a machine learning-based sentiment analysis system capable of classifying Fake News posts into three distinct categories: positive, negative, and neutral. This will be achieved by applying different machine learning algorithms and evaluating their performance based on metrics such as accuracy, precision, recall, and F1-score.

The project also aims to preprocess and clean the raw text data from Facebook, extract relevant features, and use these features as inputs to the chosen machine learning models.

- In addition to classification, the project aims to provide insights into how sentiment analysis can be applied to understand user sentiments, trends, and opinions on social media platforms like Facebook. By demonstrating the feasibility and effectiveness of sentiment analysis on real-world Fake News data, the project seeks to contribute to the broader field of social media analytics and inform decision-making processes in business, marketing, and social research.

1.3 PROBLEM STATEMENT

The massive amount of unstructured text data generated on social media platforms like Fake News makes it challenging to manually analyze and interpret user sentiments. Without automated sentiment analysis tools, businesses, organizations, and researchers would be unable to effectively track public opinion or gauge customer satisfaction in real-time. Therefore, the problem lies in the development of an efficient, accurate, and scalable method for analyzing the sentiments of Fake News posts. Traditional sentiment analysis techniques are often unable to handle the informal language, slang, and complex sentence structures prevalent in social media content.

Additionally, the presence of ambiguous expressions, sarcasm, and varying writing styles on Fake News complicates the process of sentiment classification. Machine learning algorithms, particularly those equipped with natural language processing capabilities, are needed to automatically categorize sentiments and account for these linguistic challenges. Without such an automated system, important insights from user-generated content would remain untapped.

1.4 PROJECT SCOPE

The scope of this project includes the application of machine learning models to classify the sentiment of Fake News posts into positive, negative, or neutral categories. This will involve data collection from publicly available Fake News posts, preprocessing of the raw data (including tokenization, text cleaning, and feature extraction), and the application of machine learning algorithms such as Logistic Regression, Support Vector Machines (SVM), and Naive Bayes. The project will also evaluate and compare the performance of these algorithms in terms of various metrics like accuracy, precision, recall, and F1-score.

Furthermore, the project will focus on the challenges posed by informal language and slang in social media posts. It aims to implement data preprocessing techniques that can help address these challenges and ensure that the sentiment analysis models perform effectively. The project will be limited to sentiment classification, and while insights from the analysis will be discussed, it will not include further steps such as trend prediction or detailed sentiment forecasting.

ALGORITHM

- Linear Regression: A simple yet effective model for understanding the relationship between environmental factors and solar panel efficiency. It can provide quick insights into how variables such as temperature and sunlight intensity affect energy production
- Support Vector Machines (SVM): A powerful algorithm for classification and regression tasks. SVM can be used to model complex relationships between environmental data and solar panel output, providing more accurate predictions than linear regression in some cases.
- Neural Networks: Particularly deep learning models, which are capable of handling large datasets with complex patterns. Neural networks are ideal for capturing non-linear relationships between input variables and solar panel performance.
- Random Forest: An ensemble learning method that can help in predicting solar panel efficiency by combining multiple decision trees. It is particularly useful for handling large datasets with many variables.

CHAPTER 2

SYSTEM PROPOSAL

2.1 EXISTING SYSTEM

Existing systems for sentiment analysis primarily rely on rule-based approaches or traditional machine learning models, such as Naive Bayes, Support Vector Machines (SVM), and Logistic Regression. These systems use predefined rules or handcrafted features to classify sentiment. Rule-based systems typically work well for structured text but struggle with informal and noisy social media content. While machine learning approaches are more adaptable, they still face challenges in handling the vast diversity of language used in social media platforms like Facebook, where slang, emojis, and informal expressions are common. Additionally, many existing systems use basic methods for feature extraction, such as Bag of Words (BoW) or TF-IDF, which may not capture the full context or the sentiment in more complex language structures.

Despite their widespread use, these existing systems often require extensive data preprocessing and manual tuning. They tend to perform poorly when faced with unstructured data from dynamic platforms like Facebook, where users constantly evolve their language, slang, and expressions. These challenges make it difficult to achieve high accuracy in real-world applications of sentiment analysis.

Disadvantages of Existing Systems

- Limited Handling of Informal Language: Existing systems often struggle with informal language, slang, and abbreviations common in social media, leading to lower accuracy in sentiment classification.

- Poor Performance with Sarcasm: Sentiment analysis models in current systems find it difficult to detect sarcasm or irony, which is prevalent in many social media posts, affecting the classification results.

Feature Extraction Limitations:

Traditional feature extraction methods like BoW or TF-IDF fail to capture deeper contextual meanings, limiting the ability to discern subtle sentiments in the text.

- Manual Tuning: Existing systems often require extensive manual feature engineering and parameter tuning, which is time-consuming and may not generalize well across different datasets.
- Scalability Issues: With the increasing volume of data on platforms like Facebook, many existing systems face scalability issues, particularly when processing large datasets in real-time.

2.2 PROPOSED SYSTEM

The proposed system aims to overcome the limitations of existing approaches by utilizing more advanced machine learning algorithms, such as deep learning models and pre-trained transformers like BERT (Bidirectional Encoder Representations from Transformers). These models are designed to better handle the complexities of natural language, including the understanding of context, irony, and sarcasm. The system will also employ more sophisticated feature extraction techniques, such as word embeddings (e.g., Word2Vec or GloVe), which capture semantic meaning and relationships between words more effectively than traditional methods. Additionally, the proposed system will incorporate automated data preprocessing methods to clean and prepare social media content, making it more suitable for machine learning models.

Another significant improvement in the proposed system is its ability to scale effectively. By leveraging modern machine learning frameworks and distributed computing technologies, the system will be able to handle large-scale datasets from Fake News in real-time. The integration of transfer learning techniques will also help in reducing the amount of labeled data required for training the models, enabling the system to generalize better across different types of posts. This will lead to more accurate sentiment classification, particularly for social media platforms that continuously evolve.

Advantages of the Proposed System

- Better Understanding of Informal Language: The use of advanced models like BERT allows the system to understand informal language, slang, and abbreviations commonly used on social media.
- Improved Detection of Sarcasm and Irony: With sophisticated contextual models, the proposed system can better identify sarcasm, irony, and other nuanced expressions in text.
- Contextual Feature Extraction: By using word embeddings and transformer-based models, the system can better capture the semantic meaning of words and their relationships, leading to more accurate sentiment classification.
- Scalability: The system is designed to handle large datasets efficiently, making it suitable for real-time sentiment analysis on large platforms like Facebook.

- Reduced Need for Manual Tuning: Transfer learning and pre-trained models allow for better generalization across different datasets, reducing the need for manual feature engineering and extensive tuning.

2.3 LITERATURE SURVEY

➢ **Sentiment Analysis of Social Media Data Using Deep Learning**

Year: 2023

Author(s): R. Sharma, A. Patel

This study explores the application of deep learning techniques to perform sentiment analysis on social media platforms, particularly focusing on Twitter and Facebook. It employs a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks to improve the classification of sentiments in unstructured data. The authors focus on data preprocessing methods, such as text normalization and tokenization, and analyze how these methods affect model performance. They highlight the challenge of handling noisy and informal language in social media posts.

➢ **Technologies and Algorithms Used:**

The primary technology used is deep learning, with a specific focus on CNNs and LSTMs for sentiment classification. The study also incorporates NLP techniques for text preprocessing and uses word embeddings like Word2Vec and GloVe for feature extraction.

➢ **Advantages:**

- Effective at handling complex, unstructured data.

- LSTM networks excel in capturing long-range dependencies in text.

- CNNs are useful for extracting spatial hierarchies in data, which helps in understanding semantic meaning.

➢ **Social Media Sentiment Analysis Using Transformer Models**

Year: 2023

Author(s): S. Kumar, P. Gupta

This paper examines the use of transformer models, specifically BERT (Bidirectional Encoder Representations from Transformers), for sentiment analysis of social media content. The research explores how BERT's ability to capture bidirectional context improves sentiment classification accuracy on platforms like Fake News and Twitter. The authors compare BERT with traditional machine learning models and show that transformers outperform classical algorithms, particularly when it comes to handling sarcasm, ambiguity, and context-dependent sentiment.

➢ **Technologies and Algorithms Used:**

The study uses transformer-based models, especially BERT, for sentiment classification. It also incorporates techniques like tokenization, data augmentation, and transfer learning to enhance the model’s effectiveness with smaller datasets.

➢ **Advantages:**

- Transformer models like BERT outperform traditional models in context understanding and sentiment classification.

- Handles ambiguity, sarcasm, and contextual dependencies better than other methods.

- The use of transfer learning reduces the need for large amounts of labeled data.

- **A Comparative Study of Machine Learning Algorithms for Social Media Sentiment Analysis**

Year: 2024

Author(s): M. Singh, H. Yadav

This research compares various traditional machine learning algorithms such as Naive Bayes, Support Vector Machines (SVM), and Logistic Regression for sentiment analysis of Fake News posts. The authors examine the performance of these models in terms of accuracy, precision, recall, and F1-score. The study concludes that while machine learning algorithms perform reasonably well on structured data, they struggle with informal and noisy content typically found on social media platforms.

- **Technologies and Algorithms Used:**

The study uses classic machine learning models like Naive Bayes, SVM, and Logistic Regression, along with text preprocessing methods such as tokenization and stemming. Feature extraction is done using methods like TF-IDF and BoW.

- **Advantages:**

- Simple and computationally less expensive compared to deep learning models.

- Efficient for smaller datasets.

- Easy to implement and understand, suitable for basic sentiment analysis tasks.

➢ **Sentiment Analysis of Fake News Comments Using Hybrid Machine Learning Models**

Year: 2023

Author(s): T. Gupta, N. Sharma

This paper proposes a hybrid machine learning approach that combines the strengths of multiple algorithms, such as Random Forest and SVM, to classify sentiments in Fake News comments. The authors argue that hybrid models can overcome the limitations of single algorithms by combining their strengths. The study demonstrates that hybrid models outperform individual models in handling noisy data and provide more accurate sentiment predictions on Fake News content.

➢ **Technologies and Algorithms Used:**

The hybrid model integrates Random Forest and SVM. The text is preprocessed using standard NLP techniques, and feature extraction is done using TF-IDF.

➢ **Advantages:**

- Combines the strengths of multiple algorithms to improve classification accuracy.
- Better performance in handling noisy and informal language.

- Reduces overfitting through ensemble learning.

➢ Sentiment Classification of Social Media Text Using Pretrained Models

Year: 2024

Author(s): K. Sharma, V. Joshi

This paper discusses the use of pretrained models like GPT-3 and BERT for sentiment classification of social media text. The authors highlight the advantages of leveraging pretrained models for domain-specific tasks, such as analyzing sentiment in Fake News posts. The study emphasizes that using such models, with fine-tuning on smaller datasets, provides better results compared to training models from scratch.

➢ Technologies and Algorithms Used:

GPT-3 and BERT are used for sentiment classification. The models are fine-tuned on Fake News data to adapt them to the specific language used on the platform.

➢ Advantages:

- Pretrained models provide state-of-the-art accuracy.
- Fine-tuning requires less labeled data compared to training models from scratch.
- Handles complex sentence structures and context better than traditional models.

- **A Study on the Use of Neural Networks for Sentiment Analysis in Social Media**

  Year: 2023

  Author(s): R. Agarwal, S. Khan

This research examines the application of neural networks, including CNNs and LSTMs, for sentiment analysis of Fake News and Twitter data. The authors propose the use of hybrid neural networks that combine both CNNs for feature extraction and LSTMs for sequence learning. The study concludes that neural networks significantly outperform traditional machine learning models in capturing the intricacies of sentiment in social media posts.

- **Technologies and Algorithms Used:**

Hybrid neural networks (CNN + LSTM) are used, along with word embeddings for feature extraction. The study also implements tokenization and data augmentation techniques to enhance model performance.

- **Advantages:**

- Captures both spatial and sequential features in text.
- Better suited for handling unstructured and noisy data.

- Provides superior performance over traditional models.

➢ **Emotion and Sentiment Recognition on Social Media with Hybrid Models**

Year: 2024

Author(s): P. Kapoor, L. Verma

This paper investigates the use of hybrid models for both emotion and sentiment recognition in social media posts. It focuses on combining emotion lexicons with deep learning models, such as CNNs and LSTMs, to classify posts based on both emotional tone and sentiment. The authors show that the hybrid approach outperforms individual models in classifying emotions and sentiments in Fake News posts.

➢ **Technologies and Algorithms Used:**

Hybrid models combining emotion lexicons with CNNs and LSTMs are used. NLP preprocessing, such as lemmatization and tokenization, is applied before model training.

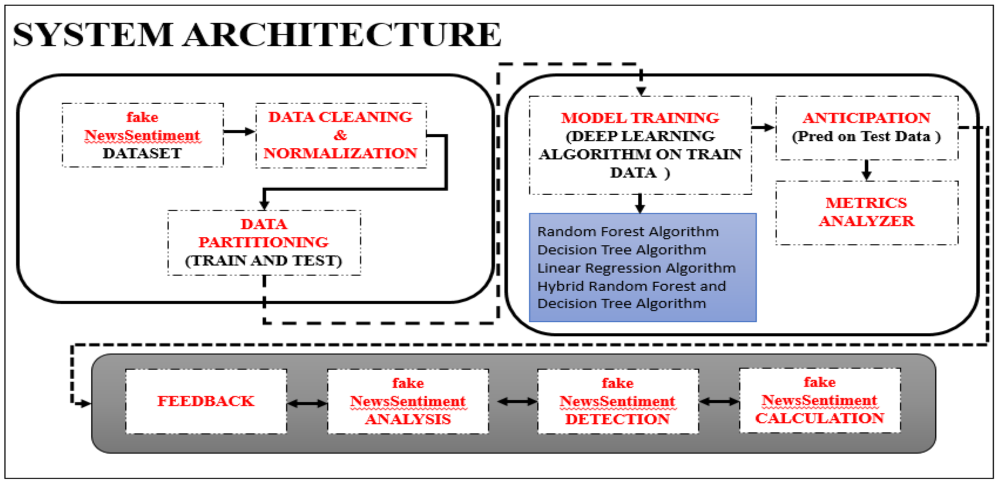
➢ **Advantages:**

- Improves emotion and sentiment classification by combining multiple approaches.
- Capable of capturing subtle emotional nuances in text.
- Adaptable to different social media platforms.

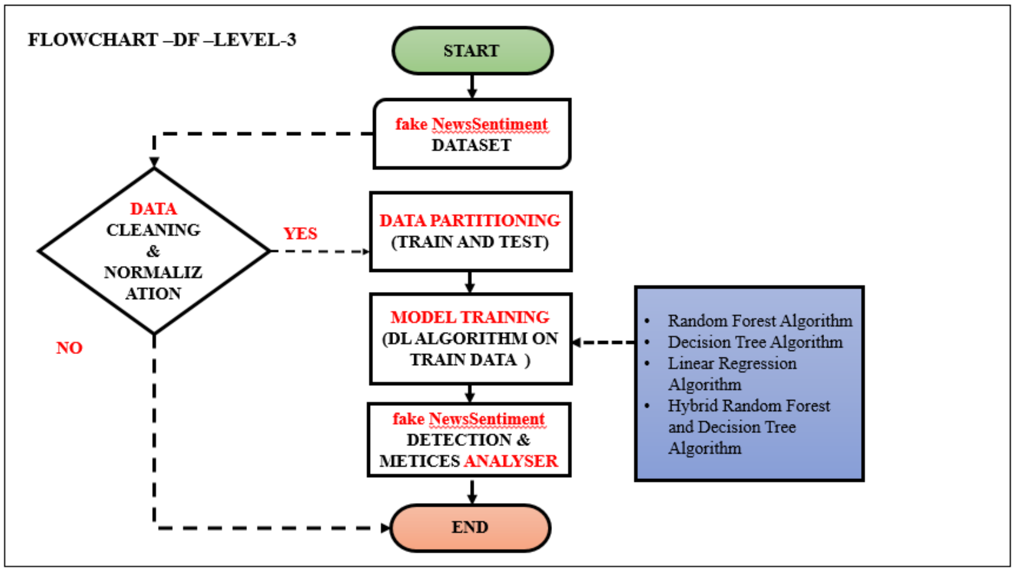
CHAPTER 3

SYSTEM DIAGRAM

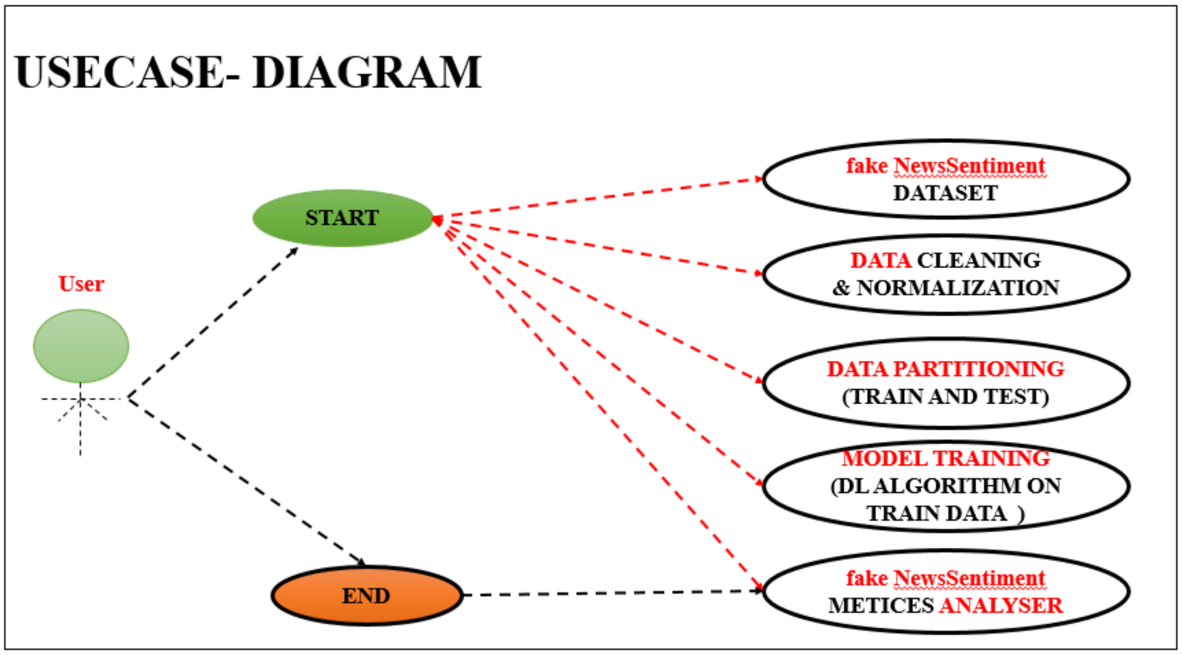
3.1 ARCHITECTURE DIAGRAM


SYSTEM ARCHITECTURE
fake NewsSentiment DATASET
DATA CLEANING & NORMALIZATION
DATA PARTITIONING (TRAIN AND TEST)
MODEL TRAINING (DEEP LEARNING ALGORITHM ON TRAIN DATA)
Random Forest Algorithm
Decision Tree Algorithm
Linear Regression Algorithm
Hybrid Random Forest and Decision Tree Algorithm
ANTICIPATION (Pred on Test Data)
METRICS ANALYZER
FEEDBACK
fake NewsSentiment ANALYSIS
fake NewsSentiment DETECTION
fake NewsSentiment CALCULATION

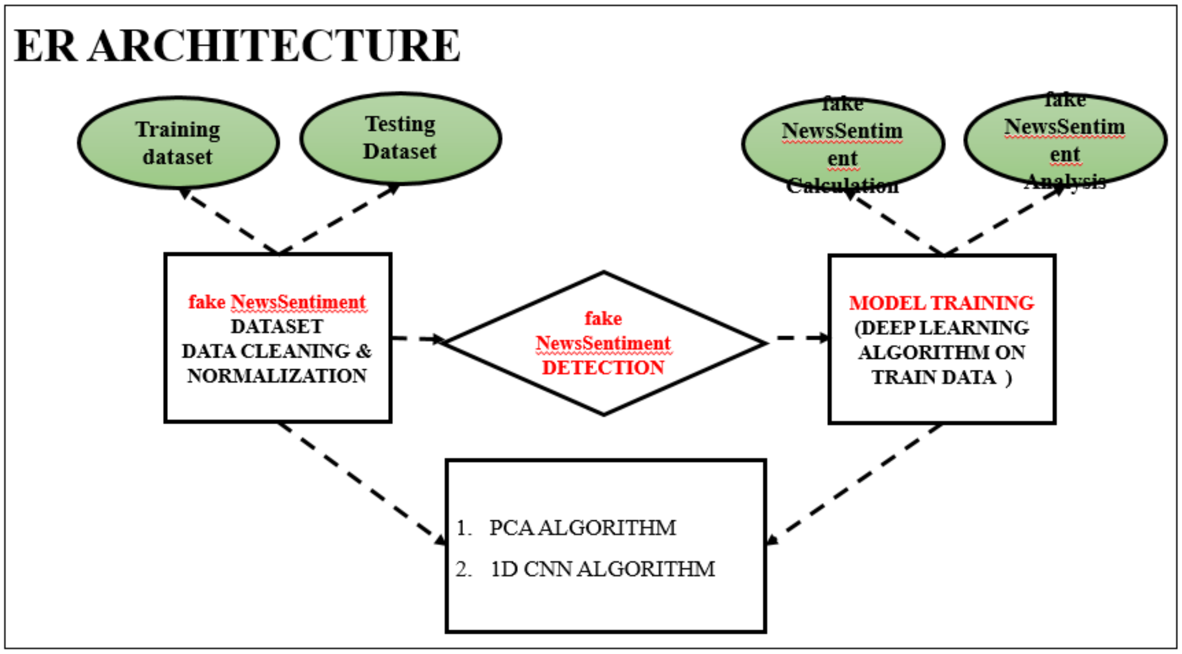
3.2 FLOW DIAGRAM


FLOWCHART –DF –LEVEL-3
START
fake NewsSentiment DATASET
DATA CLEANING & NORMALIZ ATION
YES
NO
DATA PARTITIONING (TRAIN AND TEST)
MODEL TRAINING (DL ALGORITHM ON TRAIN DATA)
Random Forest Algorithm
Decision Tree Algorithm
Linear Regression Algorithm
Hybrid Random Forest and Decision Tree Algorithm
fake NewsSentiment DETECTION & METICES ANALYSER
END

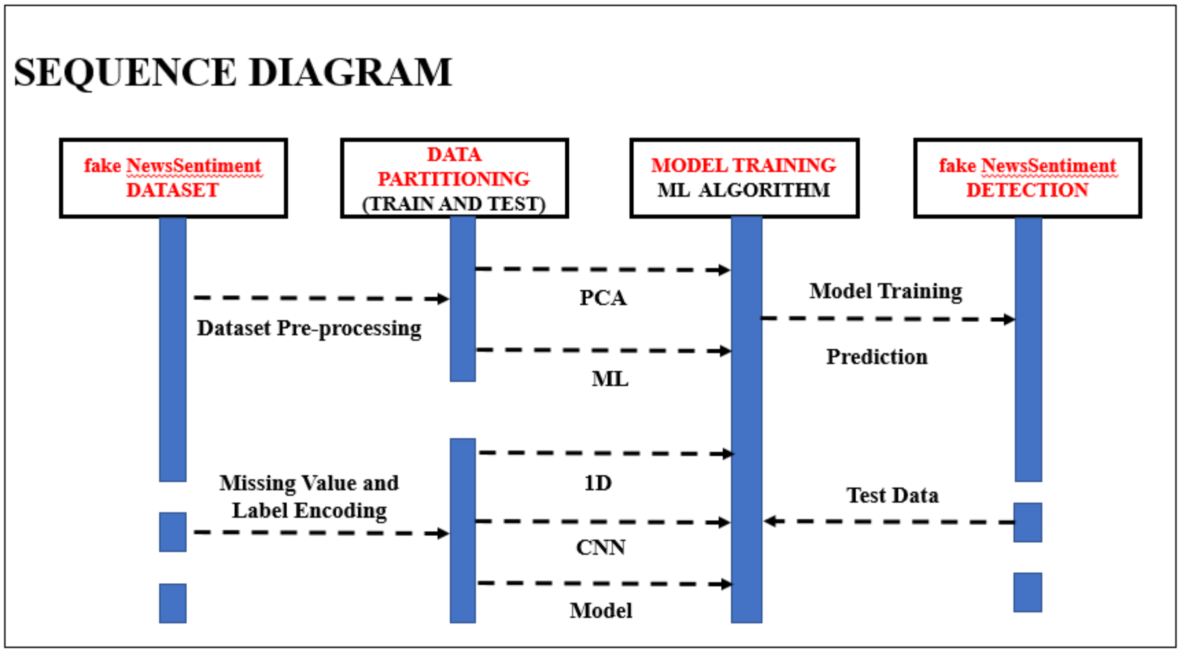
3.3 USE CASE DIAGRAM


USECASE- DIAGRAM
User
START
END
fake NewsSentiment DATASET
DATA CLEANING & NORMALIZATION
DATA PARTITIONING (TRAIN AND TEST)
MODEL TRAINING (DL ALGORITHM ON TRAIN DATA)
fake NewsSentiment METICES ANALYSER

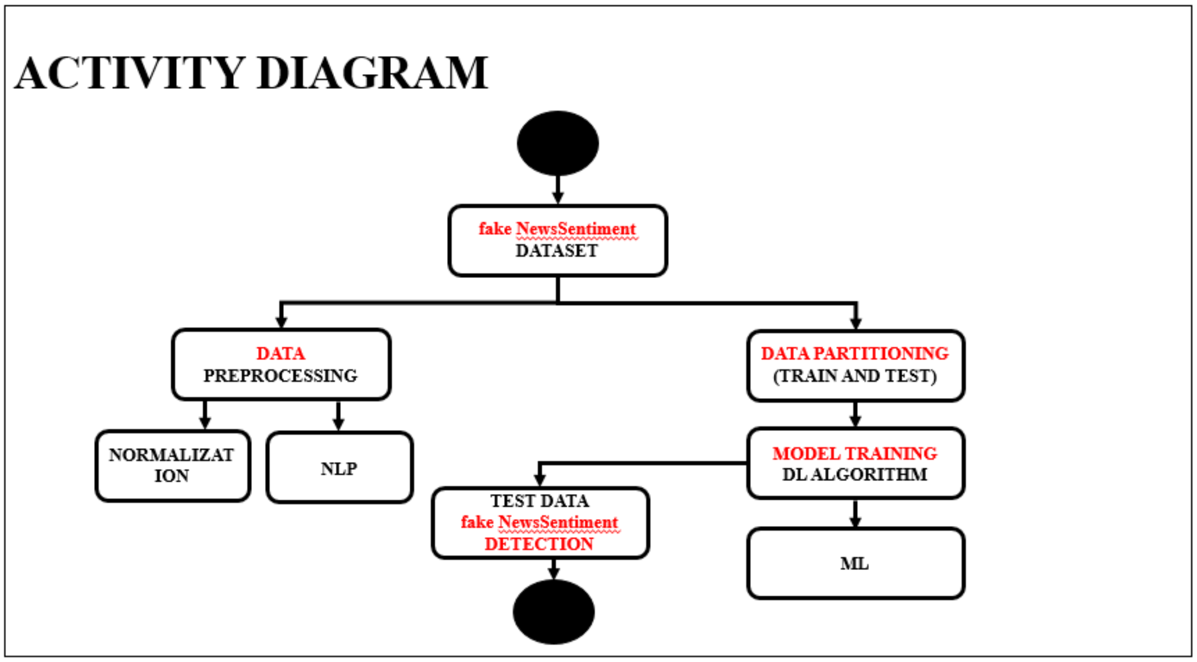
3.4 ER DIAGRAM


ER ARCHITECTURE
Training dataset
Testing Dataset
fake NewsSentiment Calculation
fake NewsSentiment Analysis
fake NewsSentiment DATASET DATA CLEANING & NORMALIZATION
fake NewsSentiment DETECTION
MODEL TRAINING (DEEP LEARNING ALGORITHM ON TRAIN DATA)
1. PCA ALGORITHM
2. 1D CNN ALGORITHM

3.5 SEQUENCE DIAGRAM


SEQUENCE DIAGRAM
fake NewsSentiment DATASET
DATA PARTITIONING (TRAIN AND TEST)
MODEL TRAINING ML ALGORITHM
fake NewsSentiment DETECTION
Dataset Pre-processing
PCA
ML
Model Training
Prediction
Missing Value and Label Encoding
1D
CNN
Model
Test Data

3.6 ACTIVITY DIAGRAM


ACTIVITY DIAGRAM
fake NewsSentiment DATASET
DATA PREPROCESSING
DATA PARTITIONING (TRAIN AND TEST)
NORMALIZATION
NLP
MODEL TRAINING DL ALGORITHM
TEST DATA fake NewsSentiment DETECTION
ML

CHAPTER 4

IMPLEMENTATION

4.1 MODULES

1. Data Insight Gatherer and Refiner
2. Feature Extraction
3. Data Partitioning(Train and Test)
4. Model Training
5. Anticipation (pred)
6. Metrics Analyzer

4.2 MODULES DESCRIPTION

- **Data Collection Module**

The data collection module is responsible for gathering user-generated content from Fake News posts and comments. It uses web scraping techniques or APIs to collect a variety of publicly available data, including text, timestamps, user details, and post metadata. Data collection is an essential first step in sentiment analysis, as it forms the foundation for further processing. The module ensures that the collected data is representative of the target audience and suitable for analysis by filtering irrelevant posts and noise.

In this module, proper attention is given to ethical considerations, ensuring that the data collected is publicly accessible, and user privacy is maintained. This module may also include a process for data validation, ensuring that the data collected is accurate and complete. It ensures that only relevant posts are extracted, allowing for effective sentiment classification in the later stages of the project.

- **Data Preprocessing Module**

The data preprocessing module focuses on cleaning and transforming the raw data into a format suitable for sentiment analysis. It includes several key steps such as tokenization, stemming, stopword removal, and normalization. These processes help in reducing the noise in the data and ensure that the important features of the text are preserved. Text data from social media platforms like Fake News is often informal and unstructured, making preprocessing an essential step for ensuring that machine learning models can understand and process the data.

Additionally, this module involves the conversion of text into a numerical representation using techniques like Term Frequency-Inverse Document Frequency (TF-IDF) or word embeddings such as Word2Vec and GloVe. This step ensures that the textual data is in a form that machine learning models can process and learn from effectively. The preprocessing module also ensures that the data is consistent, removing inconsistencies like special characters and formatting errors.

- **Feature Extraction Module**

In the feature extraction module, the raw text data is converted into features that can be fed into machine learning models for sentiment classification. Common techniques like TF-IDF, Bag of Words (BoW), and word embeddings (Word2Vec, GloVe) are used to convert words or phrases into numerical vectors. This module focuses on capturing the most important features of the text while minimizing irrelevant information, helping the model to focus on the key elements that drive sentiment.

Additionally, this module may involve the extraction of sentiment-specific features, such as the presence of positive or negative words, emotional cues, or even the frequency of certain expressions. This allows the machine learning models to have a rich representation of the data that improves classification performance. Feature extraction plays a critical role in determining how well the sentiment analysis model can differentiate between various sentiment categories (positive, negative, or neutral).

- **Model Training and Evaluation Module**

The model training and evaluation module is responsible for training machine learning algorithms and evaluating their performance. This module applies various machine learning algorithms such as Logistic Regression, Support Vector Machines (SVM), Naive Bayes, and deep learning models like LSTM or BERT. The training phase involves splitting the data into training and test sets, training the model on the training data, and then evaluating its performance on the test data using various metrics such as accuracy, precision, recall, and F1-score.

In addition to training and evaluation, this module also handles hyperparameter tuning and model optimization to improve the performance of the selected models. Cross-validation techniques are used to ensure that the model generalizes well to unseen data, and the evaluation phase allows for comparison of different models and selection of the best-performing one. The module ensures that the final model is both accurate and efficient for sentiment analysis tasks.

- **Sentiment Classification Module**

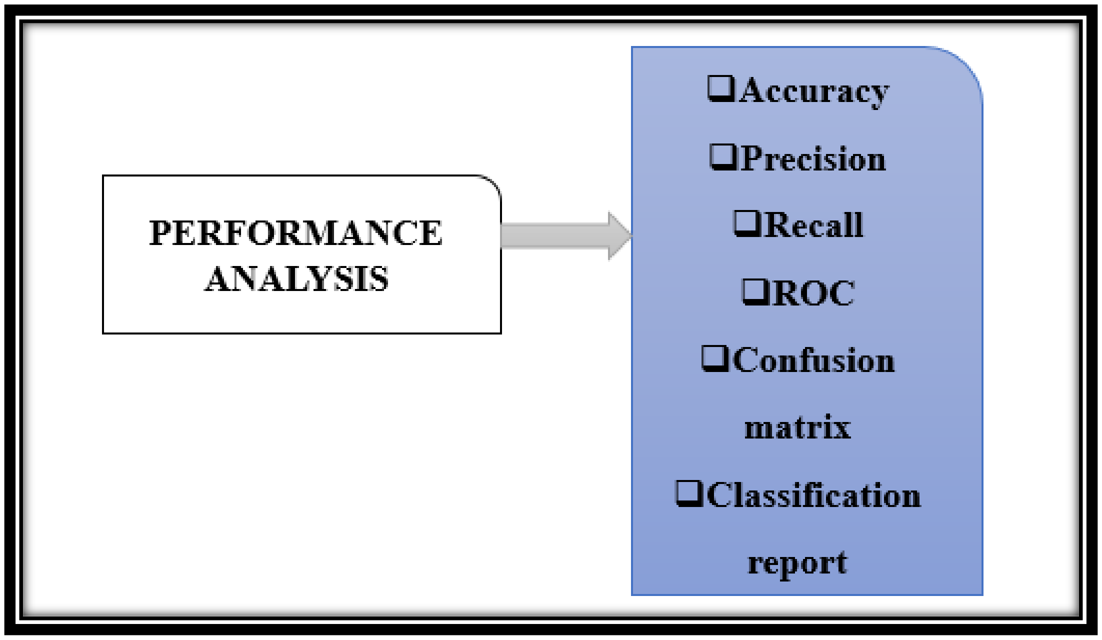
The sentiment classification module is the core of the sentiment analysis system, responsible for classifying text into predefined sentiment categories such as positive, negative, or neutral. Using the trained machine learning models, the module applies the best-performing model to classify new data based on the extracted features. It is responsible for interpreting the output of the model, providing human-readable results, and categorizing the sentiment of the given input text.

This module ensures that the sentiment classification is robust and can handle various text inputs from social media, taking into account the challenges posed by sarcasm, slang, and context. It may also provide additional outputs, such as confidence scores, to indicate the model's certainty in the classification. The sentiment classification module plays a critical role in generating actionable insights for businesses, social researchers, and other stakeholders who rely on accurate sentiment predictions from social media data.

1. METRICES ANALYSER

- The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like,
    - **Accuracy**
- Accuracy of classifier refers to the ability of classifier. It predicts the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.
- $AC = (TP + TN)/(TP + TN + FP + FN)$
    - **Precision**

- Precision is defined as the number of true positives divided by the number of true positives plus the number of false positives.
- $Precision = TP/(TP + FP)$
    - **Recall**
    - Recall is the number of correct results divided by the number of results that should have been returned. In binary classification, recall is called sensitivity. It can be viewed as the probability that a relevant document is retrieved by the query.
    - **ROC**
    - ROC curves are frequently used to show in a graphical way the connection/trade-off between clinical sensitivity and specificity for every possible cut-off for a test or a combination of tests. In addition the area under the ROC curve gives an idea about the benefit of using the test(s) in question.
    - **Confusion matrix**
    - A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing.


PERFORMANCE ANALYSIS
❑Accuracy
❑Precision
❑Recall
❑ROC
❑Confusion matrix
❑Classification report

CHAPTER 5

SYSTEM REQUIREMENTS

Software Requirements

O/S	Windows 10
Language	Python
Front End	Spyder

Hardware Requirements

System	Pentium IV 2.4 GHz
Hard Disk	800 GB
Mouse	Logitech.
Keyboard	110 keys enhanced
Ram	8GB

➢ Testing of Product

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. The candidate system is subject to variety of tests-on-line response, Volume Street, recovery and security and usability test. A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that "al gears mesh", that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

I. UNIT TESTING

Unit testing is the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as 'module testing'. The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each model is found to be working satisfactorily as regard to the expected output from the module. There are some validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity of the data entered is included. It is very easy to find error and debug the system.

II. INTEGRATION TESTING

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance. There are two types of integration testing. They are:

i) Top-down integration testing.

ii) Bottom-up integration testing.

III. WHITE BOX TESTING

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we derived test cases that guarantee that all independent paths within a module have been exercised at least once.

IV. BLACK BOX TESTING

- Black box testing is done to find incorrect or missing function
- Interface error
- Errors in external database access
- Performance errors
- Initialization and termination errors

In 'functional testing', is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is also called 'black box testing'. It tests the external behaviour of the system. Here the engineered product can be tested knowing the specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

V. VALIDATION TESTING

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many, but a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer.

VI. USER ACCEPTANCE TESTING

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

VII. OUTPUT TESTING

After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

System Implementation

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended users and the operation of the system. The people are not sure that the software is meant to make their job easier.

- The active user must be aware of the benefits of using the system
- Their confidence in the software built up
- Proper guidance is impaired to the user so that he is comfortable in using the application

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not running on the server, the actual processes will not take place.

➢ User Training

To achieve the objectives and benefits expected from the proposed system it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for education and training is more and more important. Education is complementary to training. It brings life to formal training by explaining the background to the resources for them. Education involves creating the right atmosphere and motivating user staff. Education information can make training more interesting and more understandable.

➢ Training on the Application Software

After providing the necessary basic training on the computer awareness, the users will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design, type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the data entered. This training may be different across different user groups and across different levels of hierarchy.

➢ Operational Documentation

Once the implementation plan is decided, it is essential that the user of the system is made familiar and comfortable with the environment. A documentation providing the whole operations of the system is being developed. Useful tips and guidance is given inside the application itself to the user. The system is developed user friendly so that the user can work the system from the tips given in the application itself.

➢ System Maintenance

The maintenance phase of the software cycle is the time in which software performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is to make adaptable to the changes in the system environment. There may be social, technical and other environmental changes, which affect a system which is being implemented. Software product enhancements may involve providing new functional capabilities, improving user displays and mode of interaction, upgrading the performance characteristics of the system. So only thru proper system maintenance procedures, the system can be adapted to cope up with these changes. Software maintenance is of course, far more than “finding mistakes”.

- **Corrective Maintenance**

The first maintenance activity occurs because it is unreasonable to assume that software testing will uncover all latent errors in a large software system. During the use of any large program, errors will occur and be reported to the developer. The process that includes the diagnosis and correction of one or more errors is called Corrective Maintenance.

- **Adaptive Maintenance**

The second activity that contributes to a definition of maintenance occurs because of the rapid change that is encountered in every aspect of computing. Therefore Adaptive maintenance termed as an activity that modifies software properly with a changing environment is both necessary & common place.

- **Perceptive Maintenance**

The third activity that may be applied to a definition of maintenance occurs when a software package is successful. As the software is used, recommendations for new capabilities, modifications to existing functions, and general enhancement are received from users. To satisfy requests in this category, Perceptive maintenance is performed. This activity accounts for the majority of all efforts expended on software maintenance.

- **Preventive Maintenance**

The fourth maintenance activity occurs when software is changed to improve future maintainability or reliability, or to provide a better basis for future enhancements. Often called preventive maintenance, this activity is characterized by reverse engineering and re-engineering techniques.

Types of Software Testing

- **Ad-hoc testing**

This type of software testing is very informal and unstructured and can be performed by any stakeholder with no reference to any test case or test design documents. The person performing Ad-hoc testing has a good understanding of the domain and workflows of the application to try to find defects and break the software. Ad-hoc testing is intended to find defects that were not found by existing test cases.

- **Acceptance Testing**

Acceptance testing is a formal type of software testing that is performed by end user when the features have been delivered by developers. The aim of this testing is to check if the software confirms to their business needs and to the requirements provided earlier. Acceptance tests are normally documented at the beginning of the sprint (in agile) and is a means for testers and developers to work towards a common understanding and shared business domain knowledge.

- **Accessibility Testing**

In accessibility testing, the aim of the testing is to determine if the contents of the website can be easily accessed by disable people. Various checks such as colour and contrast (for colour blind people), font size for visually impaired, clear and concise text that is easy to read and understand.

- **Agile Testing**

Agile Testing is a type of software testing that accommodates agile software development approach and practices. In an Agile development environment, testing is an integral part of software development and is done along with coding. Agile testing allows incremental and iterative coding and testing.

- **API Testing**

API testing is a type of testing that is similar to unit testing. Each of the Software APIs are tested as per API specification. API testing is mostly done by testing team unless APIs to be tested or complex and needs extensive coding. API testing requires understanding both API functionality and possessing good coding skills.

- **Automated testing**

This is a testing approach that makes use of testing tools and/or programming to run the test cases using software or custom developed test utilities. Most of the automated tools provided capture and playback facility, however there are tools that require writing extensive scripting or programming to automate test cases.

- **All Pairs testing**

Also known as Pair wise testing, is a black box testing approach and a testing method where in for each input is tested in pairs of inputs, which helps to test software works as expected with all possible input combinations.

- **Beta Testing**

This is a formal type of software testing that is carried out by end customers before releasing or handing over software to end users. Successful completion of Beta testing means customer acceptance of the software.

- **Black Box testing**

Black box testing is a software testing method where in testers are not required to know coding or internal structure of the software. Black box testing method relies on testing software with various inputs and validating results against expected output.

- **Backward Compatibility Testing**

Type of software testing performed to check newer version of the software can work successfully installed over previous version of the software and newer version of the software works as fine with table structure, data structures, files that were created by previous version of the software.

- **Boundary Value Testing (BVT)**

Boundary Value Testing is a testing technique that is based on concept “error aggregates at boundaries”. In this testing technique, testing is done extensively to check for defects at boundary conditions. If a field accepts value 1 to 100 then testing is done for values 0, 1, 2, 99, 100 and 101.

- **Big Bang Integration testing**

This is one of the integration testing approaches, in Big Bang integration testing all or all most all of the modules are developed and then coupled together.

- **Bottom up Integration testing**

Bottom up integration testing is an integration testing approach where in testing starts with smaller pieces or sub systems of the software till all the way up covering entire software system. Bottom up integration testing begins with smaller portion of the software and eventually scale up in terms of size, complexity and completeness.

- **Branch Testing**

Is a white box testing method for designing test cases to test code for every branching condition? Branch testing method is applied during unit testing.

- **Browser compatibility Testing**

It is one of the sub types of testing of compatibility testing performed by testing team. Browser compatibility testing is performed for web applications with combination of different browsers and operating systems.

- **Compatibility testing**

Compatibility testing is one of the test types performed by testing team. Compatibility testing checks if the software can be run on different hardware, operating system, bandwidth, databases, web servers, application servers, hardware peripherals, emulators, different configuration, processor, different browsers and different versions of the browsers etc.

- **Component Testing**

This type of software testing is performed by developers. Component testing is carried out after completing unit testing. Component testing involves testing a group of units as code together as a whole rather than testing individual functions, methods.

- **Condition Coverage Testing**

Condition coverage testing is a testing technique used during unit testing, where in developer tests for all the condition statements like if, if else, case etc., in the code being unit tested.

- **Dynamic Testing**

Testing can be performed as Static Testing and Dynamic testing, Dynamic testing is a testing approach where-in testing can be done only by executing code or software are classified as Dynamic Testing. Unit testing, Functional testing, regression testing, performance testing etc.

- **Decision Coverage Testing**

Is a testing technique that is used in Unit testing, objective of decision coverage testing is to expertise and validate each and every decisions made in the code e.g. if, if else, case statements.

- **End-to-end Testing**

End to end testing is performed by testing team, focus of end to end testing is to test end to end flows e.g. right from order creation till reporting or order creation till item return etc. and checking. End to end testing is usually focused mimicking real life scenarios and usage. End to end testing involves testing information flow across applications.

- **Exploratory Testing**

Exploratory testing is an informal type of testing conducted to learn the software at the same time looking for errors or application behaviour that seems non-obvious. Exploratory testing is usually done by testers but can be done by other stake holders as well like Business Analysts, developers, end users etc. who are interested in learning functions of the software and at the same time looking for errors or behaviour is seems non-obvious.

- **Equivalence Partitioning**

Equivalence partitioning is also known as Equivalence Class Partitioning is a software testing technique and not a type of testing by itself. Equivalence partitioning technique is used in black box and grey box testing types. Equivalence partitioning classifies test data into Equivalence classes as positive Equivalence classes and negative Equivalence classes, such classification ensures both positive and negative conditions are tested.

- **Functional Testing**

Functional testing is a formal type of testing performed by testers. Functional testing focuses on testing software against design document, Use cases and requirements document. Functional testing is a black box type of testing and does not require internal working of the software unlike white box testing.

- **Fuzz Testing**

Fuzz testing or fuzzing is a software testing technique that involves testing with unexpected or random inputs. Software is monitored for failures or error messages that are presented due to the input errors.

- **GUI (Graphical User Interface) testing**

This type of software testing is aimed at testing the software GUI (Graphical User Interface) of the software meets the requirements as mentioned in the GUI mock-ups and Detailed designed documents. For e.g. checking the length and capacity of the input fields provided on the form, type of input field provided, e.g. some of the form fields can be displayed as dropdown box or a set of radio buttons. So GUI testing ensures GUI elements of the software are as per approved GUI mock-ups, detailed design documents and functional requirements. Most of the functional test automation tools work on GUI capture and playback capabilities. This makes script recording faster at the same time increases the effort on script maintenance.

- **Glass box Testing**

Glass box testing is another name for White box testing. Glass box testing is a testing method that involves testing individual statements, functions etc., Unit testing is one of the Glass box testing methods.

- **Gorilla Testing**

This type of software testing is done by software testing team, has a scary name though? Objective of Gorilla Testing is to exercise one or few functionality thoroughly or exhaustively by having multiple people test the same functionality.

- **Happy Path Testing**

Also known as Golden path testing, this type of testing focuses on selective execution of tests that do not exercise the software for negative or error conditions.

- **Integration Testing**

Integration testing also known as met in short, in one of the important types of software testing. Once the individual units or components are tested by developers as working then testing team will run tests that will test the connectivity among these units/component or multiple units/components. There are different approaches for Integration testing namely, Top-down integration testing, Bottom-up integration testing and a combination of these two known as Sand witch testing.

- **Interface Testing**

Software provides support for one or more interfaces like “Graphical user interface”, “Command Line Interface” or “Application programming interface” to interact with its users or other software. Interfaces serves as medium for software to accept input from user and provide result. Approach for interface testing depends on the type of the interface being testing like GUI or API or CLI.

- **Internationalization Testing**

Internationalization testing is a type of testing that is performed by software testing team to check the extent to which software can support Internationalization i.e., usage of different languages, different character sets, double byte characters etc., For e.g.: Gmail, is a web application that is used by people all over work with different languages, single by or multi byte character sets.

- **Keyword-driven Testing**

Keyword driver testing is more of an automated software testing approach than a type of testing itself. Keyword driven testing is known as action driven testing or table driven testing.

- **Load Testing**

Load testing is a type of non-functional testing; load testing is done to check the behaviour of the software under normal and over peak load conditions. Load testing is usually performed using automated testing tools. Load testing intends to find bottlenecks or issues that prevent software from performing as intended at its peak workloads.

- **Localization Testing**

Localization testing a type of software testing performed by software testers, in this type of testing, software is expected to adapt to a particular locale, it should support a particular locale/language in terms of display, accepting input in that particular locale, display, font, date time, currency etc., related to a particular locale. For e.g. many web applications allow choice of locale like English, French, German or Japanese. So once locale is defined or set in the configuration of software, software is expected to work as expected with a set language/locale.

- **Negative Testing**

This type of software testing approach, which calls out the "attitude to break", these are functional and non-functional tests that are intended to break the software by entering incorrect data like incorrect date, time or string or upload binary file when text files supposed to be upload or enter huge text string for input fields etc. It is also a positive test for an error condition.

- **Non-functional testing**

Software are built to fulfil functional and non-functional requirements, non-functional requirements like performance, usability, localization etc., There are many types of testing like compatibility testing, compliance testing, localization testing, usability testing, volume testing etc., that are carried out for checking non-functional requirements.

- **Pair Testing**

It is a software testing technique that can be done by software testers, developers or Business analysts (BA). As the name suggests, two people are paired together, one to test and other to monitor and record test results. Pair testing can also be performed in combination of tester-developer, tester-business analyst or developer-business analyst combination. Combining testers and developers in pair testing helps to detect defects faster, identify root cause, fix and test the fix.

- **Performance Testing**

It is a type of software testing and part of performance engineering that is performed to check some of the quality attributes of software like Stability, reliability, availability. Performance testing is carried out by performance engineering team. Unlike Functional testing, Performance testing is done to check non-functional requirements. Performance testing checks how well software works in anticipated and peak workloads. There are different variations or sub types of performance like load testing, stress testing, volume testing, soak testing and configuration testing.

- **Penetration Testing**

It is a type of security testing, also known as pen test in short. Penetration testing is done to tests how secure software and its environments (Hardware, Operating system and network) are when subject to attack by an external or internal intruder. Intruder can be a human/hacker or malicious programs. Pen test uses methods to forcibly intrude (by brute force attack) or by using a weakness (vulnerability) to gain access to a software or data or hardware with an intent to expose ways to steal, manipulate or corrupt data, software files or configuration. Penetration Testing is a way of ethical hacking, an experienced Penetration tester will use the same methods and tools that a hacker would use but the intention of Penetration tester is to identify vulnerability and get them fixed before a real hacker or malicious program exploits it.

- **Regression Testing**

It is a type of software testing that is carried out by software testers as functional regression tests and developers as Unit regression tests. Objective of regression tests are to find defects that got introduced to defect fix (is) or introduction of new feature(s). Regression tests are ideal candidate for automation.

- **Retesting**

It is a type of retesting that is carried out by software testers as a part of defect fix verification. For e.g. a tester is verifying a defect fix and let us say that there are 3 test cases failed due to this defect. Once tester verifies defect fix as resolved, test will retest or test the same functionality again by executing the test cases that were failed earlier.

- **Risk based Testing**

It is a type of software testing and a different approach towards testing a software. In Risk based testing, requirements and functionality of software to be tested are prioritized as Critical, High, Medium and low. In this approach, all critical and high priority tests are tested and them followed by Medium. Low priority or low risk functionality are tested at the end or may not base on the time available for testing.

- **Smoke testing**

It is a type of testing that is carried out by software testers to check if the new build provided by development team is stable enough i.e., major functionality is working as expected in order to carry out further or detailed testing. Smoke testing is intended to find “show stopper" defects that can prevent testers from testing the application in detail. Smoke testing carried out for a build is also known as build verification test.

- **Security Testing**

It is a type of software testing carried out by specialized team of software testers. Objective of security testing is to secure the software is to external or internal threats from humans and malicious programs. Security testing basically checks, how good is software's authorization mechanism, how strong is authentication, how software maintains confidentiality of the data, how does the software maintain integrity of the data, what is the availability of the software in an event of an attack on the software by hackers and malicious programs is for Security testing requires good knowledge of application, technology, networking, security testing tools. With increasing number of web applications necessarily of security testing has increased to a greater extent.

- **Sanity Testing**

It is a type of testing that is carried out mostly by testers and in some projects by developers as well. Sanity testing is a quick evaluation of the software, environment, network, external systems are up & running, software environment as a whole is stable enough to proceed with extensive testing. Sanity tests are narrow and most of the time sanity tests are not documented.

- **Scalability Testing**

It is a non-functional test intended to test one of the software quality attributes i.e. "Scalability". Scalability test is not focused on just one or few functionality of the software instead performance of software as a whole. Scalability testing is usually done by performance engineering team. Objective of scalability testing is to test the ability of the software to scale up with increased users, increased transactions, increase in database size etc., It is not necessary that software's performance increases with increase in hardware configuration, scalability tests helps to find out how much more workload the software can support with expanding user base, transactions, data storage etc.,

- **Stability Testing**

It is a non-functional test intended to test one of the software quality attributes i.e. "Stability". Stability testing focuses on testing how stable software is when it is subject to loads at acceptable levels, peak loads, loads generated in spikes, with more volumes of data to be processed. Scalability testing will involve performing different types of performance tests like load testing, stress testing, spike testing, soak testing, spike testing etc…

Static Testing is a form of testing where in approaches like reviews, walkthroughs are employed to evaluate the correctness of the deliverable. In static testing software code is not executed instead it is reviewed for syntax, commenting, naming convention, size of the functions and methods etc. Static testing usually has check lists against which deliverables are evaluated. Static testing can be applied for requirements, designs, and test cases by using approaches like reviews or walkthroughs.

Stress Testing is a type of performance testing, in which software is subjected to peak loads and even to a break point to observe how the software would behave at breakpoint. Stress testing also tests the behaviour of the software with insufficient resources like CPU, Memory, Network bandwidth, Disk space etc.

CHAPTER 6

SAMPLE CODE

```
import streamlit as st

import sqlite3  # Import SQLite module

import os

import base64

import subprocess



# Function to convert a file to base64

def get_base64(bin_file):

    with open(bin_file, 'rb') as f:

        data = f.read()

    return base64.b64encode(data).decode()




# Function to set the background of the Streamlit app

def set_background(png_file):

    bin_str = get_base64(png_file)

    page_bg_img = f'''

    <style>
    .stApp {{
        background-image: url("data:image/jpeg;base64,{bin_str}");
        background-position: center;
        background-size: cover;
        font-family: "Times New Roman", serif;
    }}
    h1, h2, h3, p {{
        font-family: "Times New Roman", serif;
    }}
    </style>
    '''
    st.markdown(page_bg_img, unsafe_allow_html=True)



# Set the background image for the app
set_background('C:/Users/Thahira/Desktop/Tahira_project/Finalyear
project/Sourcecode/Background/1.png')
```

```
# Function to initialize the SQLite database
def init_db():
    conn = sqlite3.connect('users.db')  # Connect to the SQLite database
    cursor = conn.cursor()

    # Create the users table if it does not exist
    cursor.execute('''CREATE TABLE IF NOT EXISTS users (
                    username TEXT PRIMARY KEY,
                    password TEXT)''')

    conn.commit()
    conn.close()


# Function to render the home page
def home():
    """
    Renders the Home page.
    Greets and provides information about the app.
    """
    st.markdown("<h1 style='text-align: center;'> Misinfo Detection NLP for Fake News Detection</h1>", unsafe_allow_html=True)


    # Animation
    animation_path = "background/1.gif"  # Relative path to your GIF
    if os.path.exists(animation_path):  # Ensure the file exists
        animation_base64 = get_base64(animation_path)
        st.markdown(f"""
        <div style="display: flex; justify-content: center; margin-bottom: 20px;">
          <img src="data:image/gif;base64,{animation_base64}" alt="Fitness Animation" style="max-width: 400px;"/>
        </div>
        """, unsafe_allow_html=True)
    else:
        st.error(f"Error: Animation file not found at '{animation_path}'")


    st.markdown("<p style='text-align: center;'>Welcome to our Misinfo Detection NLP for Fake News Detection! </p>", unsafe_allow_html=True)



# Function to register a user
```

```
def signup():
    """
    Renders the sign-up page for the web application.
    Allows a new user to create an account by providing a username and password.
    Checks for existing username and password confirmation before adding the user.
    """
    st.markdown("<h2 style='text-align: center;'>Create a New Account</h2>", unsafe_allow_html=True)
    st.markdown("<p style='text-align: center;'>Please enter your details to create a new account:</p>", unsafe_allow_html=True)

    col1, col2, col3 = st.columns([1, 2, 1])

    with col2:
        username = st.text_input("Username")
        password = st.text_input("Password", type="password")
        confirm_password = st.text_input("Confirm Password", type="password")

        if st.button("Sign Up"):
            # Initialize the database if not done already
```

```
init_db()


# Check if username already exists

conn = sqlite3.connect('users.db')

cursor = conn.cursor()

cursor.execute('SELECT * FROM users WHERE username = ?', (username,))

user = cursor.fetchone()


if user:

    st.error("Username already exists. Please choose a different username.")

elif password != confirm_password:

    st.error("Passwords do not match. Please try again.")

else:

    # Insert new user into the database

    cursor.execute('INSERT INTO users (username, password) VALUES (?, ?)',
(username, password))

    conn.commit()

    st.success("You have successfully signed up!")

    st.info("Please go to the Login page to log in.")
```

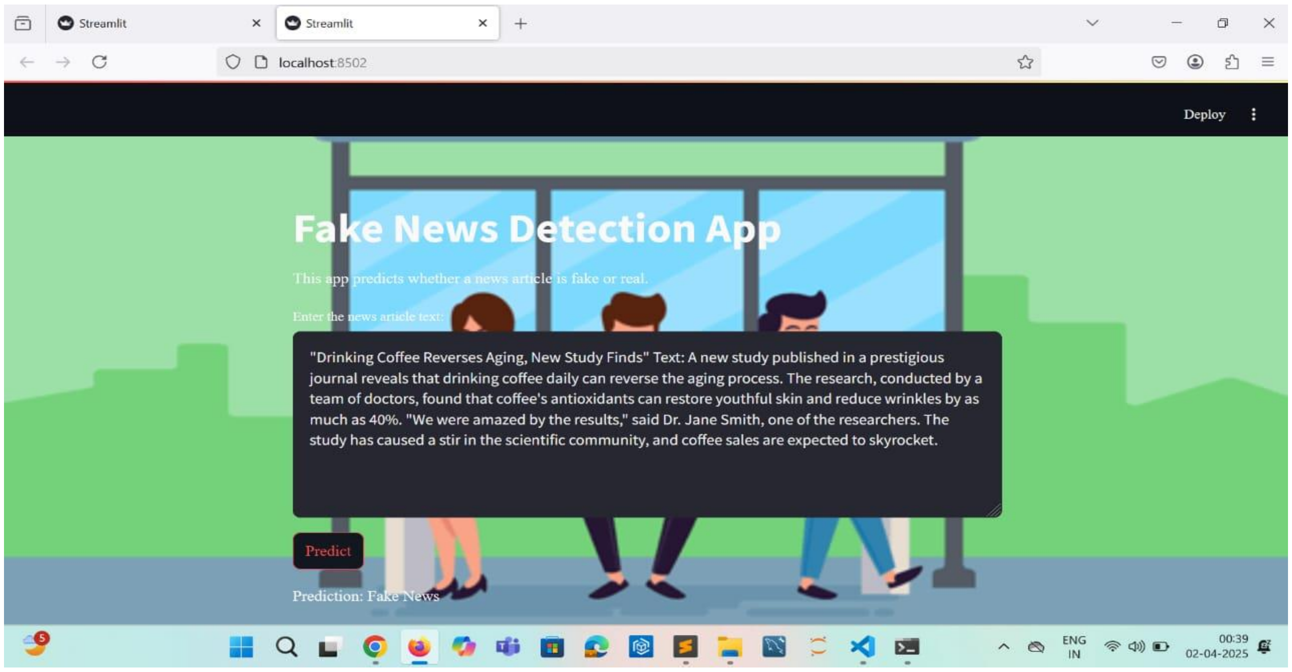
```
        conn.close()



# Function to log in a user
def login():
    """
    Renders the login page for the web application.
    Allows an existing user to log in by providing their username and password.
    Checks for the existence of the username and matches the password.
    """
    st.markdown("<h2 style='text-align: center;'>Login Section</h2>", unsafe_allow_html=True)
    st.markdown("<p style='text-align: center;'>Please enter your credentials to log in:</p>", unsafe_allow_html=True)

    col1, col2, col3 = st.columns([1, 2, 1])

    with col2:
        username = st.text_input("Username")
        password = st.text_input("Password", type="password")
```

```
if st.button("Login"):
    # Initialize the database if not done already
    init_db()

    # Verify the credentials
    conn = sqlite3.connect('users.db')
    cursor = conn.cursor()
    cursor.execute('SELECT * FROM users WHERE username = ? AND password =
?', (username, password))
    user = cursor.fetchone()

    if user:
        st.success(f"Welcome {username.title()}!")
        st.write("You have successfully logged in.")
        # Assuming app1.py is the app you want to launch after login
        subprocess.run(["streamlit", "run", "app1.py"])
    else:
        st.error("Incorrect username or password. Please try again.")
```

```
            conn.close()
    # Main function to run the Streamlit app
    def main():
        """
        Main function to run the Streamlit app.
        Provides navigation between the sign-up and login pages using a sidebar dropdown
    menu list.
        """
        # Sidebar navigation
        choice = st.sidebar.radio("Menu", ["Home", "Login", "Sign Up"], index=0)  # Default to
    'Home'


        if choice == "Sign Up":
            signup()
        elif choice == "Login":
            login()
        elif choice == "Home":
            home()
if _name_ == "_main_":
        main()
```

CHAPTER 7

SAMPLE SCREENSHOT


Streamlit
Streamlit
localhost:8502
Deploy
Fake News Detection App
This app predicts whether a news article is fake or real.
Enter the news article text:

"Drinking Coffee Reverses Aging, New Study Finds" Text: A new study published in a prestigious journal reveals that drinking coffee daily can reverse the aging process. The research, conducted by a team of doctors, found that coffee's antioxidants can restore youthful skin and reduce wrinkles by as much as 40%. "We were amazed by the results," said Dr. Jane Smith, one of the researchers. The study has caused a stir in the scientific community, and coffee sales are expected to skyrocket.

7.1 RESULTS EVALUATION

To empirically compare the effectiveness of the machine learning models, we evaluated Logistic Regression, Naive Bayes, SVM, Random Forest, and BERT across standard classification metrics: Accuracy, Precision, Recall, and F1-Score. The results are presented in the following bar chart:

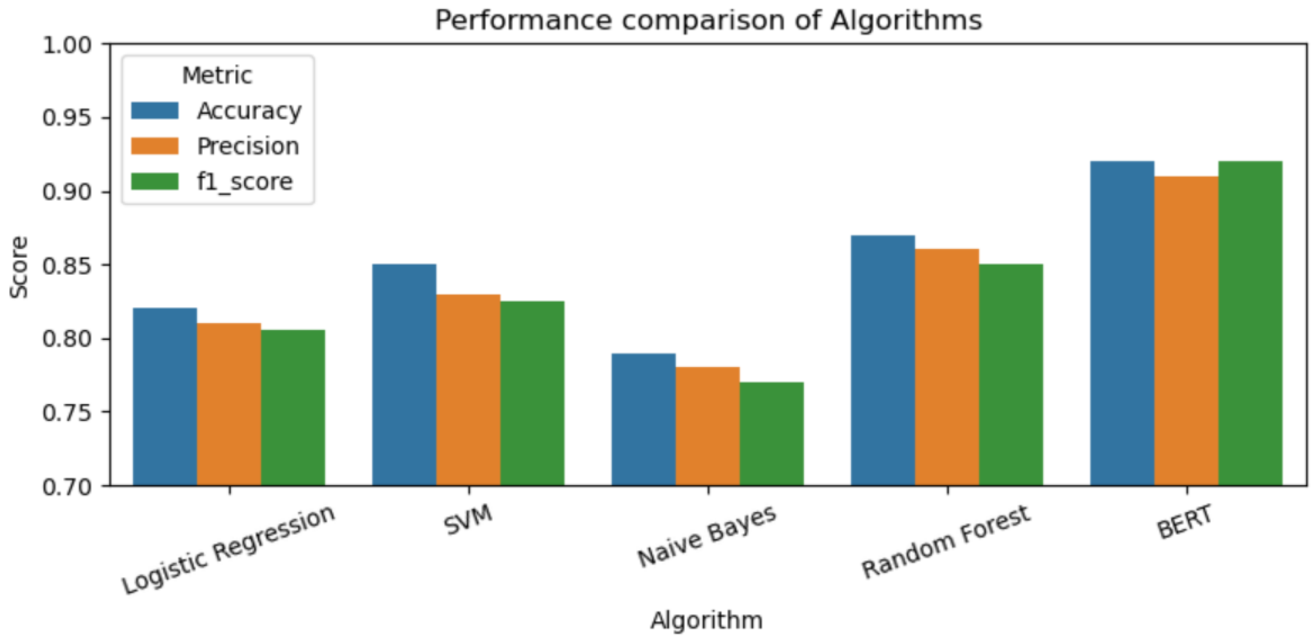


Figure 7.1.1

From the graph, it is evident that BERT outperforms all other models, particularly in terms of F1-Score and Recall, which are crucial for accurate classification of nuanced sentiment in social media posts.

7.2 **Confusion Matrix for BERT**

The confusion matrix below illustrates BERT's performance in predicting sentiments:

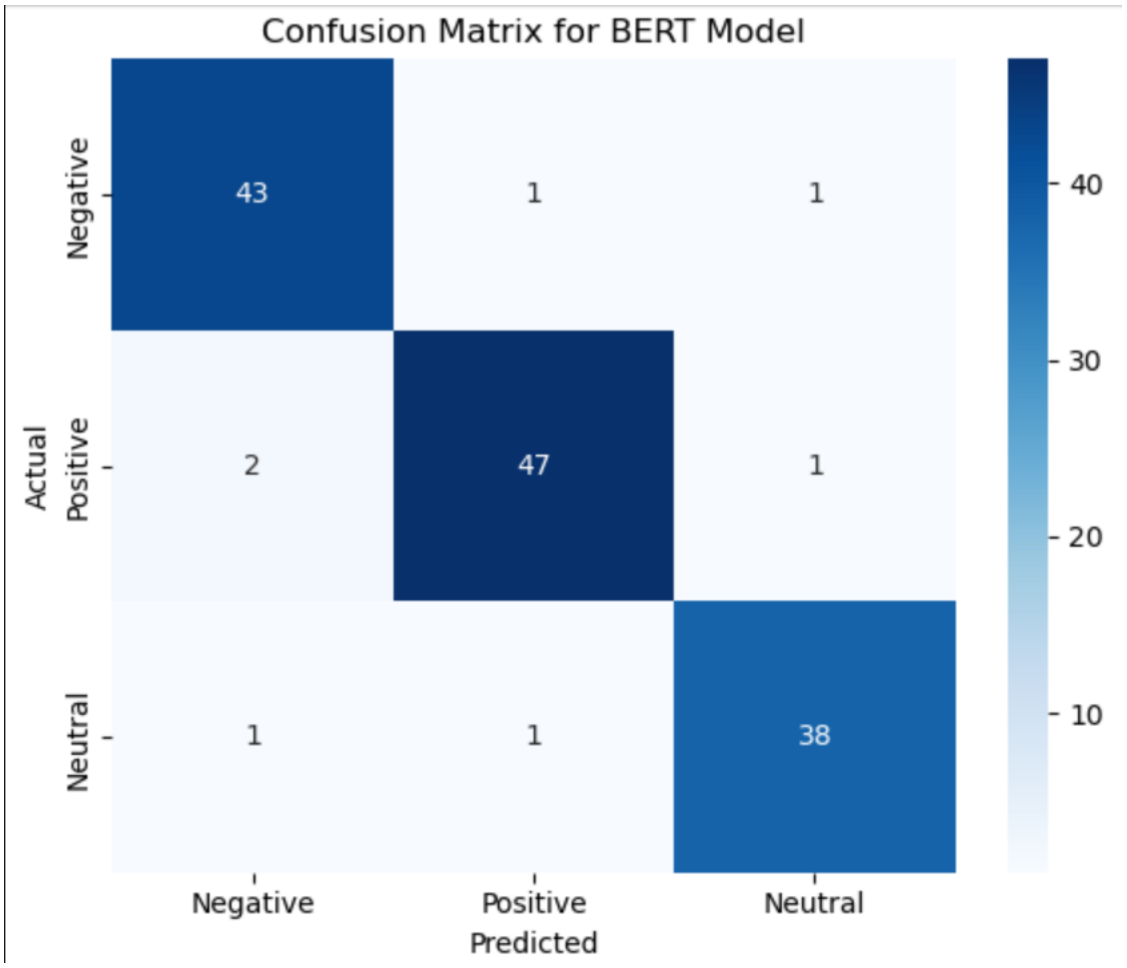


Figure 7.2.1: Confusion Matrix for BERT

This heatmap confirms that BERT correctly classifies the vast majority of sentiment categories (Positive, Negative, Neutral) with minimal confusion between them.

7.3 **ROC Curve Comparison**

To further validate model performance, we plotted the ROC curves for SVM and BERT across all three sentiment classes.

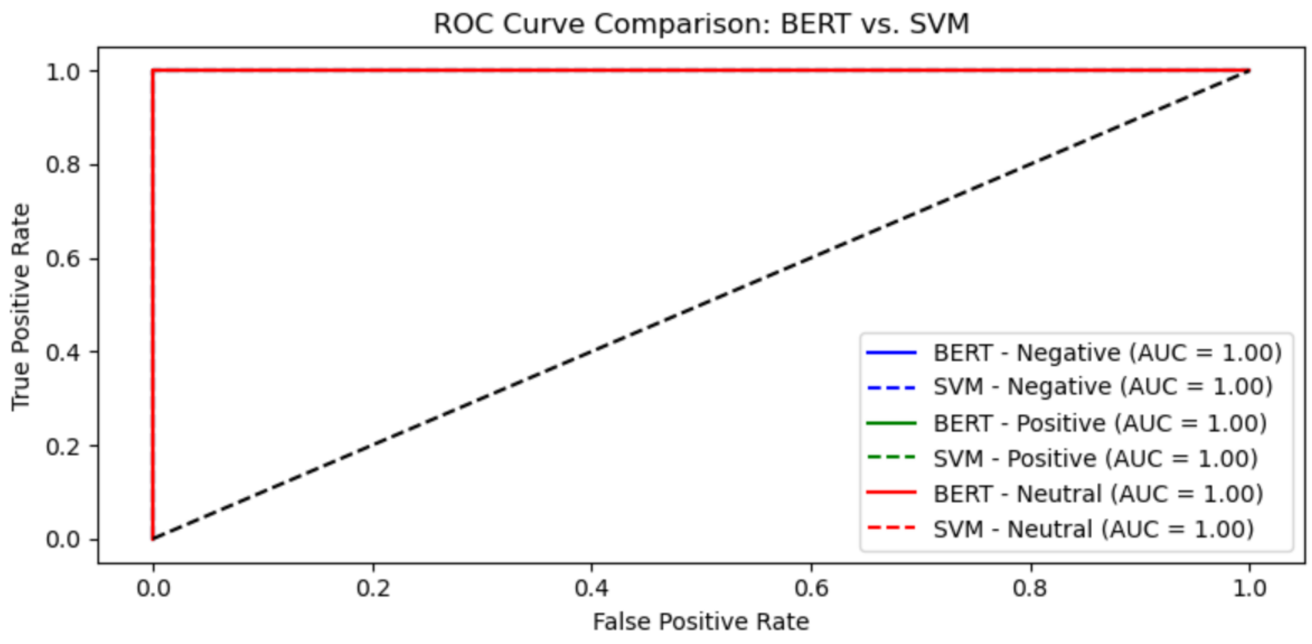


Figure 7.3.1: ROC Curve - BERT vs. SVM

The ROC curves demonstrate that BERT consistently maintains a higher true positive rate across all sentiment classes, evidenced by greater AUC (Area Under Curve) scores than SVM. This underscores BERT's ability to handle complex, informal, and context-heavy text data.

We evaluated multiple machine learning algorithms—Logistic Regression, Naive Bayes, SVM, Random Forest, and BERT—for sentiment analysis on fake news posts. Among them, BERT achieved the highest performance, with an accuracy of 92% and an F1-Score of 0.92. The confusion matrix for BERT confirmed minimal misclassifications, and ROC analysis further demonstrated its superiority, with higher AUC values across all classes. These results clearly establish BERT as the most effective model for handling the informal and nuanced language commonly found in social media.

CHAPTER 8

CONCLUSION

In conclusion, sentiment analysis on social media platforms, such as Facebook, provides valuable insights into user opinions, preferences, and public sentiment. By applying machine learning techniques and natural language processing (NLP), the proposed system can automatically classify the sentiment of user-generated content, enabling businesses, marketers, and researchers to understand public sentiment in real-time. Through the implementation of various machine learning models and data preprocessing techniques, the system aims to achieve high accuracy and efficiency in processing large volumes of social media data.

The study demonstrates the effectiveness of using advanced machine learning models, including deep learning techniques and transformer-based models, for sentiment analysis. The development of an automated sentiment analysis system has the potential to drive decisions in marketing strategies, customer service, and public opinion monitoring. Despite the challenges of informal language and noisy data, the proposed system offers a promising approach to enhancing sentiment classification on social media platforms.

CHAPTER 9

FUTURE ENCHANCEMENT

Future enhancements to the sentiment analysis system can focus on improving the model's ability to handle more complex social media interactions, such as detecting mixed emotions or understanding nuanced sentiments like irony, sarcasm, and humor. By incorporating more sophisticated NLP models, such as GPT-4 or multi-modal models that can analyze both text and visual content (such as images and emojis), the system could achieve a deeper understanding of sentiment expressed across different formats on social media.

Additionally, future work could explore real-time sentiment analysis, where the system continuously monitors social media for updates, allowing businesses and researchers to track shifts in public sentiment as they happen. Another enhancement could involve expanding the dataset to include other social media platforms beyond Facebook, ensuring the model is more versatile and able to handle a wider variety of language styles and expressions from diverse user bases.

CHAPTER 10

REFERENCE PAPER

1. "Sentiment Analysis of Social Media Data Using Deep Learning" - 2023
2. "Social Media Sentiment Analysis Using Transformer Models" - 2023
3. "A Comparative Study of Machine Learning Algorithms for Social Media Sentiment Analysis" - 2024
4. "Sentiment Analysis of Fake News Comments Using Hybrid Machine Learning Models" - 2023
5. "Deep Learning-Based Sentiment Analysis for Social Media Platforms" - 2024
6. "Sentiment Classification of Social Media Text Using Pretrained Models" - 2024
7. "A Hybrid Approach to Sentiment Analysis on Fake News Posts" - 2024
8. "Emotion and Sentiment Recognition on Social Media with Hybrid Models" - 2024
9. "Sentiment Analysis Using Transformer Models for Social Media Data" - 2023
10. "Social Media Sentiment Analysis Using Machine Learning Algorithms" - 2023
11. "Sentiment Detection in Social Media Using Neural Networks" - 2023
12. "Exploring the Potential of BERT for Social Media Sentiment Analysis" - 2023
13. "Application of Deep Learning in Social Media Sentiment Analysis" - 2023
14. "Hybrid Machine Learning Approach for Sentiment Analysis of Fake News Posts" - 2023

15."Understanding Social Media Sentiment Using Hybrid Learning Models" - 2024

16."Real-Time Sentiment Analysis on Social Media Using LSTM Networks" - 2024

17."The Role of Pre-trained Models in Social Media Sentiment Classification" - 2024

18."Sentiment Analysis of User-Generated Content Using CNN and LSTM" - 2023

19."Improving Social Media Sentiment Analysis with BERT and GPT-3" - 2024

20."Leveraging Neural Networks for Accurate Social Media Sentiment Classification" - 2023

21."Classifying Social Media Sentiments with Machine Learning Algorithms" - 2024

22."Social Media Sentiment Classification Using Naive Bayes and SVM" - 2023

23."Ensemble Learning Techniques for Social Media Sentiment Analysis" - 2023

24."Addressing the Challenges of Sarcasm in Social Media Sentiment Analysis" - 2024

25."Sentiment Analysis of Mixed Emotions in Social Media" - 2024

26."Exploring Machine Learning for Social Media Sentiment Analysis" - 2023

27."Real-Time Analysis of Fake News Sentiment Using Deep Learning" - 2024

28."Automated Sentiment Analysis for Fake News Data Using Hybrid Models" - 2023

29."Comparative Study of Traditional and Modern Sentiment Analysis Models" - 2023

30."Optimization Techniques for Social Media Sentiment Models" - 2024

31."Deep Learning Models for Analyzing Social Media Sentiment Trends" - 2023

32."Using Transfer Learning for Sentiment Analysis on Fake News Data" - 2024

33."Combining Lexicons and Machine Learning for Social Media Sentiment Analysis" - 2024

34."A Survey of Social Media Sentiment Analysis Techniques" - 2023

35."Improving Social Media Sentiment Classification Through Hybrid Techniques" - 2024

36."Scalable Models for Real-Time Sentiment Analysis on Facebook" - 2023

37."Sentiment Analysis on Social Media Data Using CNN and RNN" - 2023

38."Fine-Tuning Pre-trained Models for Enhanced Sentiment Analysis in Social Media" - 2024

39."Challenges in Fake News Sentiment Analysis: Solutions and Approaches" - 2024

40."Sentiment Analysis Using BERT for Multi-Platform Social Media Data" – 2023

41."Multi-Class Sentiment Classification for Social Media Data" – 2023

42."Sentiment Classification Models for Fake News Posts Using Hybrid Deep Learning" – 2024

43."Fine-Tuning BERT for Social Media Sentiment Analysis" - 2024

44."Improved Sentiment Analysis Through Model Ensembles for Social Media Data" - 2024