Fake News Detection: A Machine Learning Approach

Moses Daniel Kwaknat, Nentawe Gurumdimma

Department of Computer Science University of Jos, Nigeria

DOI: https://doi.org/10.51584/IJRIAS.2025.100500041

Received: 10 May 2024; Revised: 24 April 2025; Accepted: 28 April 2025; Published: 06 June 2025

ABSTRACT

Fake news has become one of the most discussed topics around the world today because of the influence it has on the decision making of individuals and even top government bodies around the world. It has become a real menace to society especially as it is easily disseminated on the internet through the use of social media. Many algorithms and models have been developed over the years to help checkmate the spread of fake news on the internet. However, there has been no real implementation of these algorithms online

for the public to use. This research built a fake news detection model using three different binary classifiers: naive Bayes, logistic regression, and random forest. The logistic regression classifier was found to be the most accurate with an accuracy of 96%. The dataset was sourced from Kaggle and comprises labeled real and fake news articles. Evaluation was extended to include precision, recall, and F1-score for better performance insight. The model was deployed as a web application.

Keywords: Dataset, machine learning, supervised learning, artificial intelligence, data science

INTRODUCTION

Fake news has become a significant global concern due to its potential to influence individual decision-making and impact government bodies worldwide (Pogue, 2017). With the rise of social media, fake news can be easily disseminated online, making it a real menace to society. Despite the development of various algorithms and models to combat fake news over the years, there has been a lack of practical implementation of these solutions for public use on the internet.

This research aimed to build a fake news detection model using three different binary classifiers: naive Bayes, logistic regression, and random forest. The logistic regression classifier model was found to be the most accurate, with a 96% accuracy rate in detecting fake news. The research further implemented this model as a web application, enabling users to detect online fake news articles.

The background section highlights the threat posed by fake news to democracy, journalism, and freedom of expression. It has undermined public trust in governments and potentially influenced significant events like the Brexit referendum and the divisive 2016 U.S. presidential elections (Pogue, 2017). Fake news has also impacted economies, causing stock market fluctuations and massive trades (Rapoza, 2017).

While fake news is not a new phenomenon, the rise of social media has played a crucial role in its surge. As of August 2017, 67% of Americans relied on social media for news (Shu et al., 2017), making it easier and cheaper to create and disseminate fake news online compared to traditional news media.

The problem statement emphasizes the growing concerns about fake news publishers posting "fake" stories and disseminating them widely using fake followers. The lack of scalable fact-checking strategies is particularly worrisome, leading to the need for fake news checkers on the internet to differentiate between genuine and fake news articles.

The research objectives include gathering a dataset, identifying existing fake news detection methods, preprocessing data to improve model accuracy, building a fake news detection model using supervised learning techniques, evaluating the models to identify the most accurate one, and implementing the model on the web.

The methodology involves conducting a literature review, procuring a reliable dataset from Kaggle, and utilizing natural language processing techniques, supervised learning, and web development tools like Flask, HTML, CSS, and Bootstrap4.

In summary, this research addresses the critical issue of fake news by developing a machine learning model for accurate detection and providing a practical web application solution for public use, contributing to the ongoing efforts to combat the spread of misinformation online.

Review of related works

The proliferation of fake news, particularly on social media, has emerged as a significant challenge with far-reaching consequences, prompting extensive research efforts to develop effective detection methods. The literature review explores various approaches that have been employed to tackle this issue.

Content-based techniques, which rely on linguistic features and writing styles, have been widely adopted. Pérez-Rosas et al. (2017) extracted linguistic cues, such as n-grams, punctuation, and psycholinguistic features, and used a linear SVM classifier for fake news detection. Conroy et al. (2015) proposed a hybrid approach combining linguistic analysis and network patterns. However, content-based methods can be language-dependent and may be susceptible to sophisticated fake news that does not exhibit obvious linguistic patterns.

Social context-based approaches leverage user profiles, social network structures, and user interactions. Shu et al. (2017) proposed a data mining framework that extracts features from news content, user characteristics, and social engagements. Ruchansky et al. (2017) developed a hybrid deep model called CSI that captures patterns from user activity, source credibility, and article content using recurrent neural networks.

Propagation-based techniques exploit the observation that fake news spreads differently from real news over social networks. Monti et al. (2019) used geometric deep learning on graph-structured data to model news propagation patterns for detection. Shu et al. (2018) studied network diffusion models to trace provenance paths and mitigate fake news spread.

Several studies have employed supervised learning models, such as logistic regression, random forests, and support vector machines, for fake news classification (Reis et al., 2019). Deep learning approaches, like convolutional neural networks and adversarial training, have also been explored (Cui et al., 2019; Monti et al., 2019). Yang et al. (2019) proposed an unsupervised generative approach that treats news credibility and user trustworthiness as latent variables.

Multimodal frameworks that combine textual content, user comments, and visual information have gained traction. Cui et al. (2019) developed SAME, a sentiment-aware multimodal embedding model that leverages user sentiments for fake news detection. Innovative solutions like blockchain technology (Paul et al., 2019) and network analysis tools (Shu et al., 2018) have also been investigated.

While significant progress has been made, the literature review highlights the need for robust, scalable, and language-independent solutions that can effectively combat the spread of fake news across different online platforms and media formats.

METHODOLOGY

Introduction

This section presents a model of a proposed solution for the research which is being carried out. It focuses on the various components needed for a fake news detection system. The data requirements as well as the architecture of the system will be discussed. This section will also present an overview of the framework used for the design of the system and the design decisions which were taken for the development of the system.

Requirements

The main focus of this system is all around natural language processing. Natural language processing focuses on the analyses of textual data. A labelled dataset that consists of thousands of written online news articles from different news urls around the world will be used as the data for this research.

Dataset

A dataset as the name implies is a set of data that consists of all information that was gathered during a survey which needs to be analyzed. The dataset being analyzed in this research consists of textual data being structured in a table in a csv format. It contains urls, headlines, bodies and labels of about four thousand and nine news articles around the world on different topics such as politics,

sports, entertainment etc. The dataset used in this research was sourced from Kaggle and is publicly available at https://www.kaggle.com/datasets/jruvika/fake-news-detection. It contains 20,800 labeled news articles in CSV format, with fields such as URLs, headlines, body text, and labels ('0' or '1', ‘0’ indicating FAKE and ‘1’ indicating REAL). The dataset includes a balanced distribution of 10,400 fake and 10,400 real news articles. While the dataset covers various topics like politics, entertainment, and sports, potential biases may arise from the data source, publisher credibility, and topic selection. This limitation is noted as part of the future improvement of the model.

Programming Language

The main programming language being considered for this work is the python programming language. There are other languages that can also be used for this29 work but the python programming language is most suitable because it is a general programming language that is increasingly becoming more popular among data scientists around the world. Also, because it has a lot of libraries that makes it very easy for data science and machine learning such as pandas, sci-kit learn etc.

Other Tools

Other tools being used for this work includes the flask micro framework which is used for the deployment of the model on the web. Also, other tools include: HTML5, CSS and Bootstrap4 for the graphical user interface.

Architecture

The system mainly consists of three parts, firstly reading the news article text files into the system, secondly the analysis of the read in news articles and thirdly making inferences from the analysis process.

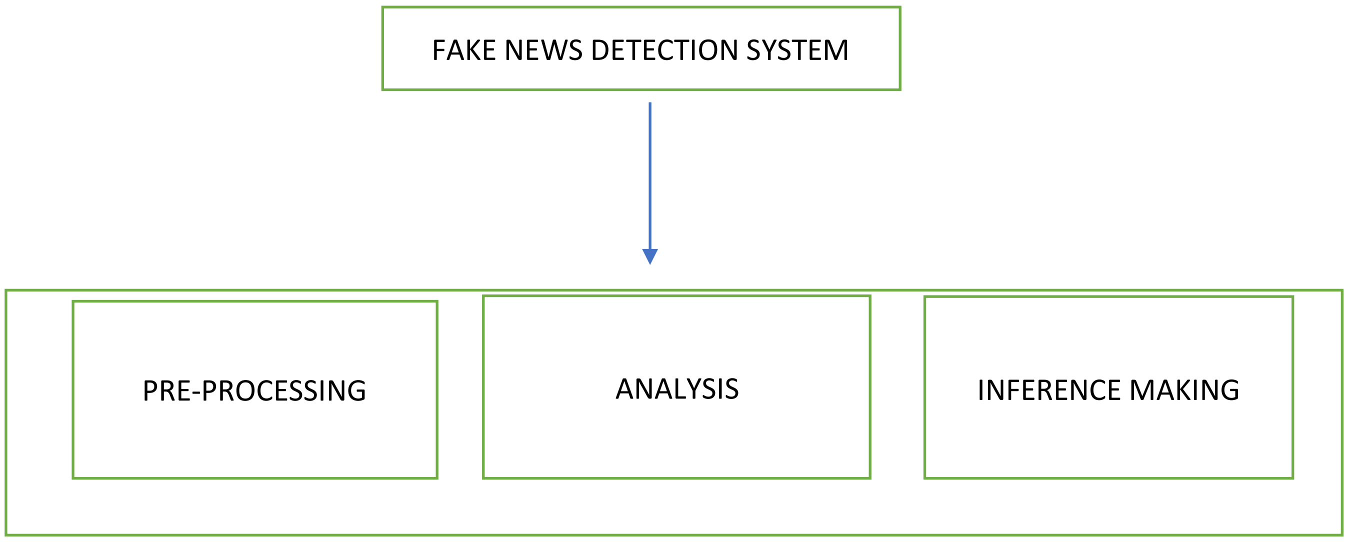


Figure 3.1 General Architecture for solution

Pre-processing

This is the first approach in the methodology, it involves collecting all the news articles which are textual. The next thing to do is data cleaning because the data contains so much noise and some words which are irrelevant in the extraction of features. Data cleaning involves removing punctuation, numbers and making all the text lowercase etc. The text in the document is then tokenized to split the sentences into single words so each word is handled independently, this is usually referred to as the bag-of-words approach. The words are reduced to their simple forms and the most important subjects from it are selected.

Analysis of Text

This involves working through a text corpus to discover certain patterns that may be hidden in the text document. A basic operation which is done in this phase is tokenization of words from the body of the news article dataset. Machine learning techniques are then carried out on the category of the predefined dataset which is binary, and is used to categorize new probabilistic observations into the said categories.

Extraction of Common Words

Relevant words were extracted from the text corpora of the fake news article

through these steps:

1. Making all the text lowercase.

2. Removing question marks, and full stops.

3. Removing all digits from the text

4. Removing text and other special characters.

After effectively cleaning the text document, the relevant words that are peculiar to both fake and real news are removed and separated. Firstly, all the English stopwords are removed. These are normal words that are used in grammar and don’t make much meaning. These words include: a, an, the, that etc. After removing all the English stopwords, the sparse words are then removed and this is followed by reducing all the words to their lowest form in a process called lemmatization. After the text document has been broken down into single words, each word is reduced to its simplest form. The NLTK WordNet Lemmatizer aids in this process. WordNet Lemmatizer is a large library which has been trained with thousands of English words. The process is done by comparing each word with words in the WordNet database and using its in-built31 logic to reveal the real form of a word. E.g., books become book. This generally increases the amount of discovery in the text. Now the common words are then extracted from the body of the news article by getting the fake words separately and also the real words from the articles that have been labelled from the dataset as both true and fake. Finally, the most common words used by both the true and fake news articles are extracted together with their frequencies through the aid of the NLTK library. A word cloud of these common words is then generated for better visualization.

Extraction of Features

The selection of features was based on empirical and literature-backed assumptions. Lexical features (e.g., word count, sentence structure), syntactic patterns (POS tagging), and news source reliability (via URLs) have been shown to influence classification in previous studies. For instance, fake news often uses exaggerated language and higher subjectivity. These features help differentiate between real and fabricated content, especially when paired with TF-IDF vectorization. Although sentiment analysis was considered, it was excluded due to inconsistent polarity trends in both fake and real news articles, as noted in the findings.

Sentiment Analysis

Sentiment analysis is the task of detecting whether a textual item (e.g., a product review, a blog post, an editorial etc.) expresses a positive or a negative opinion in general, or about a given entity, e.g., a product, a person, a political party or a policy (Nakov et al, 2019). In extracting features for the detection of online fake news, sentiment analysis was considered because there was a hypothesis that fake news normally carries negative sentiments and real news carry positive sentiments. The main feature extracted in this study is the textual feature from the news text, but only the body of the news article was extracted. The features extracted from this includes:

Language features (syntax): Sentence-level features including the bag of words approach and parts of speech tagging (POS) as features for fake news detection. The tfidf vectorizer from sci-kit learn was used in extracting this32 feature. Certain features were implemented from this set including number of words and syllables per sentence as well as tag of words categories (such as noun, verb, and adjective).

Lexical features: Typical lexical features include character and word level signals, such as the number of unique words and their frequency in the text. Linguistic features were implemented such as number of words, first person pronouns, demonstrating pronouns, verbs etc. News source features consists of the information about the publisher of the news article. We used the URL of each news article as a feature in the detection of fake news.

Cross Validation

After these features are extracted from the text document, it is passed further for cross validation using the Kfold library from sci-kit learn. The Kfold cross nvalidation is a re-sampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called K which refers to the number of groups that a given data sample is split into. The data sample in this case is split into two. The Kfold cross validation technique is used because it results in skill estimates that generally have a lower bias than other methods.

Train Test Split

This process involves splitting the extracted features into training set and test set. This process is being carries out in order to prevent over fitting and under fitting. The training set contains a known output and the model learns on this data in order to be generalized to other data later on. The test dataset is then used in order to test the model’s prediction on this subset.

Document Term Matrix (TFIDF Vectorizer)

The document term matrix is an implementation of the bag of words concept. Document term matrix is tracking the term frequency for each term by each document. We start with the bag of words representation of the documents and then for each document the number of times a term exists is tracked. The document term matrix is important because the machine learning model we are building does not understand text and so this document term matrix has to be built to convert this text into numbers and pass them into an array as vectors. The Tfidf vectorizer converts a collection of raw documents to a matrix of TF-IDF features. The tfidf vectorizer computes the word counts, idf and tf-idf values.

Fake News Detection Model

The fake news detection model was built through extracting these features mentioned in the previous section using three classifiers, these classifiers include: Naive Bayes (NB), logistic regression (LR) and Random Forest (RF) classifier. The extracted features are used to train these binary classifiers and then used for the prediction of news articles to determine whether the news article is real or fake.

Design, Implementation, Evaluation and Testing

Introduction

In the previous sections, the necessary documents needed to create the system's design and implementations were carried out. This section will present the necessary models and designs that accurately represents the system and its workings. This chapter will also go on to carry out planned and detailed evaluation testing procedures. This is usually done to ensure proper functionality and also to guarantee the efficiency of the system.

Requirements Specification

Requirements are set of documentation that describes the features and behavior of the system.

Functional Requirement Specification

The system shall operate only on unstructured textual data.

The system shall be able to extract features from the textual news articles, URLs and classify them as either true or fake.

Non Functional Requirement Specification

Efficiency: The system should manage system resources, though processing might require a lot of processing power.

Performance: This is a measure of how much fake and true news articles the system is able to detect from the text documents and its correctness.

Software Specification

The software requirements for the system include:

Python 3.5+ and jupyter notebook or any text editor

Natural language toolkit (NLTK) - An extensive python library for natural language processing patterns.

WordNet Lemmatizer - For word reduction to canonical form.

Word tokenizer - For breaking up of text into single words.

Flask - Connect user front-end and define API calls that connects to the back-end.

HTML, CSS and Bootstrap4 - For graphical user interface.

System Design

System design simply refers to the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. It simply gives the overall plan or model of a system consisting of all specifications that gives the system form and architecture (i.e. the structural implementation of the system analysis).

Architectural Design

Architectural design is the abstraction of the main components of the system and sub-systems that make up the large system.

In the figure below, textual data is parsed into a module which converts this textual data into a bag-of-words. The tokens which are gotten from this process is analyzed using trained data. The extracted word forms and patterns are detected and they are classified using the classification model. The model then infers whether the textual data contains a fake or a true news article.

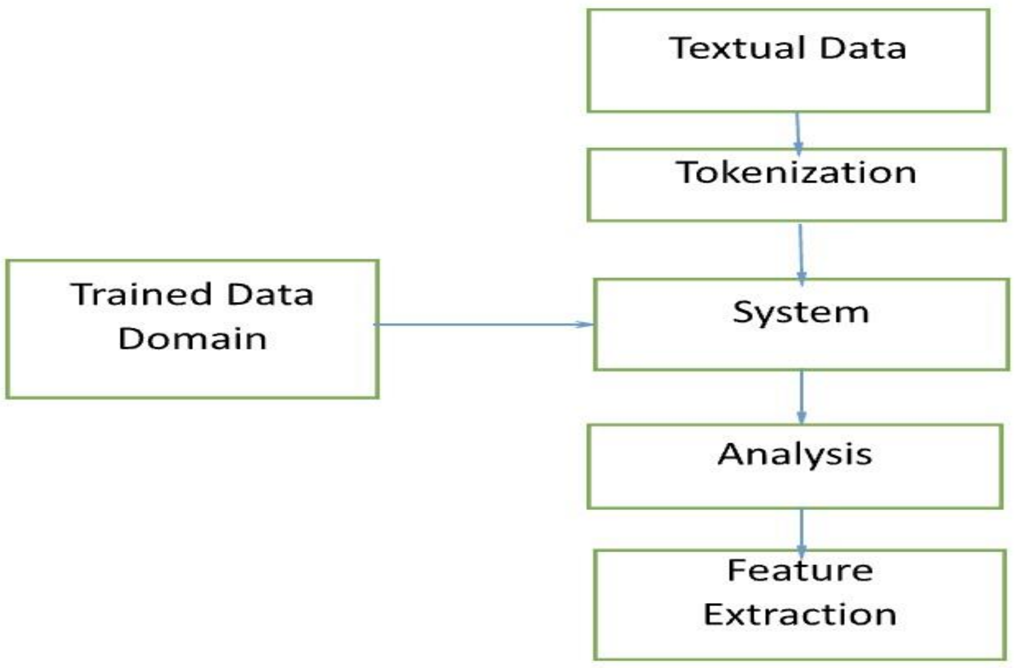


Figure 4.1 Architectural design

Activity Diagrams

This diagram shows the sequence of operations that occur in the system.

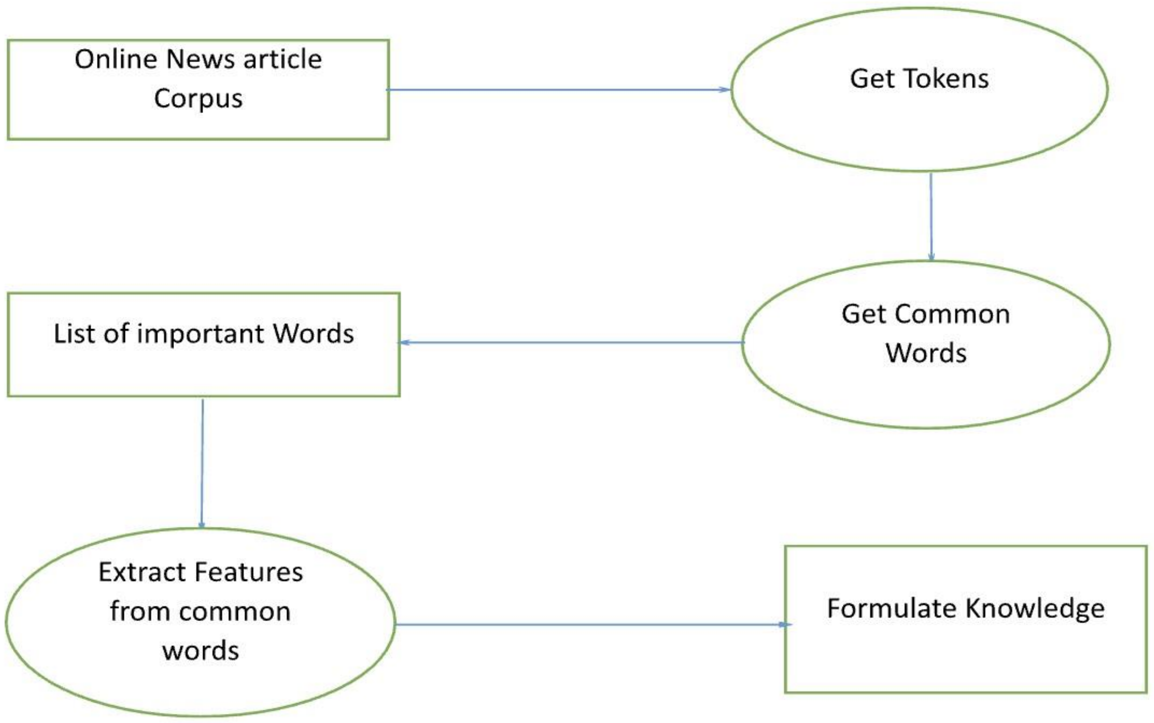


Figure 4.2 Activity Diagram

In the activity diagram above, the text documents are merged together to form a corpus, this corpus is then split using “word tokenize” from the NLTK library. The tokenized words are further reduced to their canonical forms with the aid of the same NLTK. Nouns, adjectives, and adverbs are extracted. The most occurring of these words are collected together with their different weights using the tfidf vectorizer. A document term matrix is then generated and the words are converted into numerical vectors. Machine learning techniques are then carried out on the document term matrix which contains the extracted features form the text corpus. The machine learning technique applied in this work is a binary classification technique in which different classification models were trained using the document term matrix through which knowledge is finally formulated.

Sequence Diagrams

This includes the sequence diagrams for the two major processes performed. Pre-processing and Binary classification.

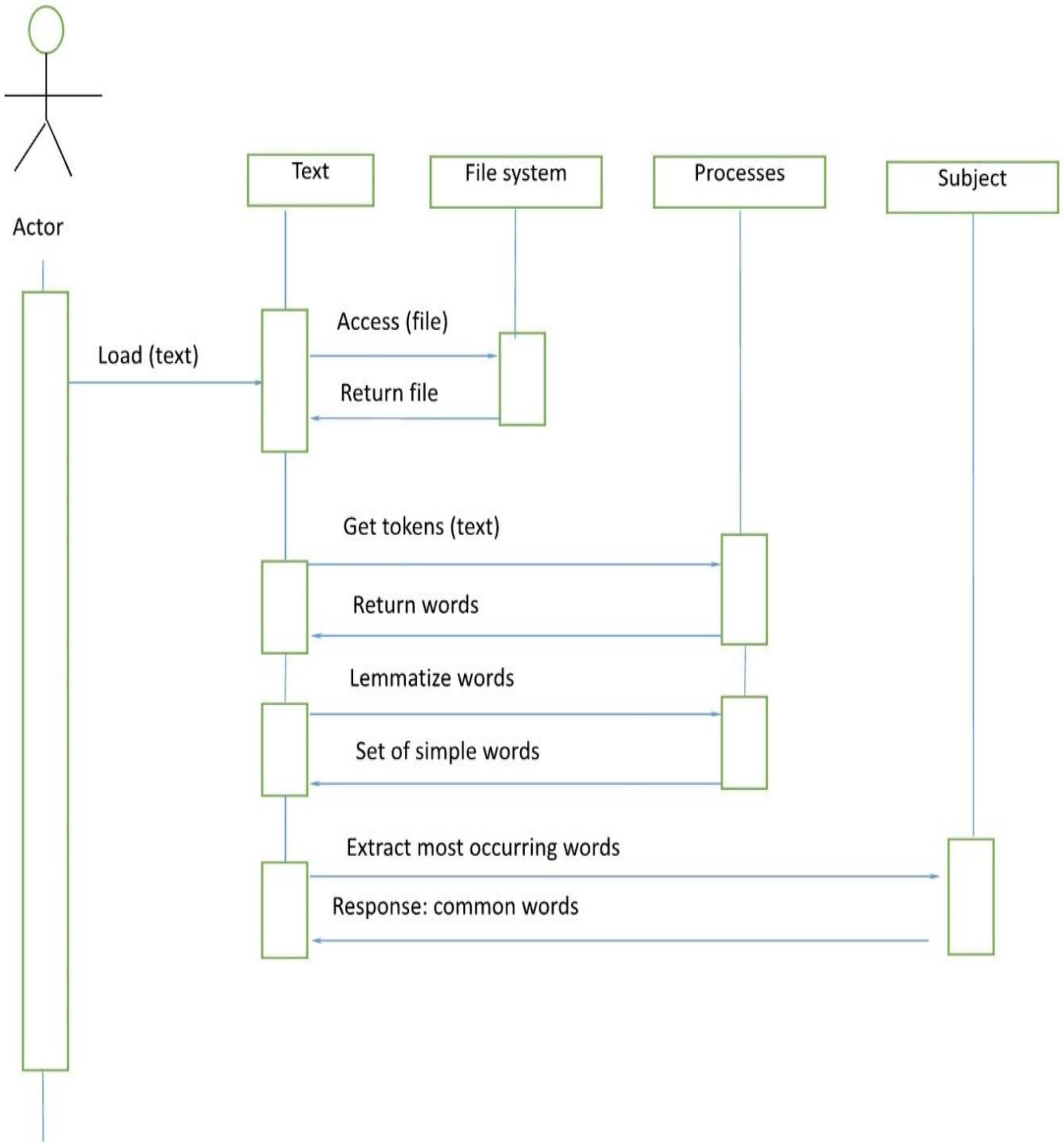


Figure 4.3 Sequence Diagram for Pre-processing

Use Case Diagram

The system performs mainly three actions; these actions perform all the necessary computations to generate hidden knowledge from the text corpus or news article. The diagram is depicted below:

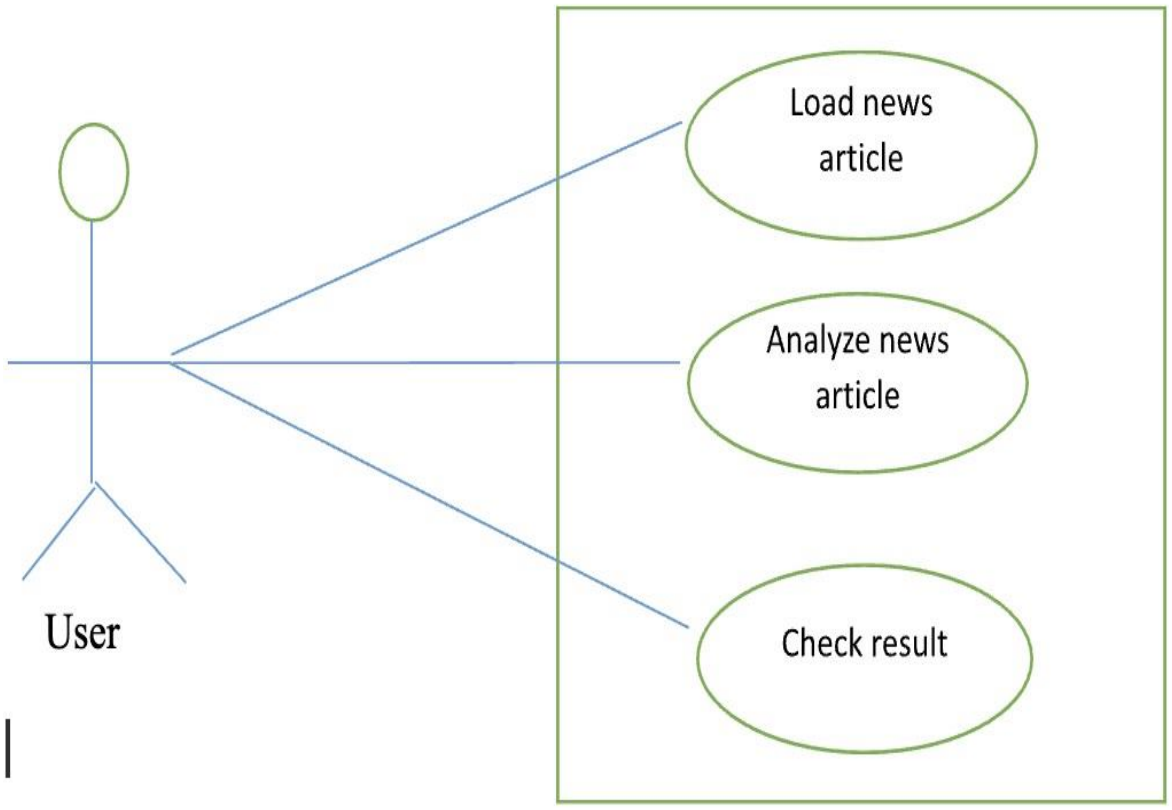


Figure 4.4 Use Case Diagram

System Implementation

System implementation is the process of defining how the information system should be built (i.e. physical system design), ensuring that the information system is operational and used, ensuring that the information system meets quality standard (i.e. quality assurance). It is the phase where programmers write the code for the modules either in a procedural-oriented design, or write the program units that implements the classes in an object-oriented design.

The modules of this system were implemented using the python programming language. Additional python libraries were also downloaded in other to carry out the project efficiently. Libraries such as pandas, numpy, and sci-kit learn, among others. The Natural language toolkit (NLTK) - an extensive and useful python library in the field of text analytics and natural language processing was also used, together with the tfidf vectorizer - an extensive library for feature extraction across various natural language processing applications. The flask web micro framework was also used to deploy the trained machine learning model into a web application.

The graphical user interface was implemented using HTML, CSS and the bootstrap4 frameworks.

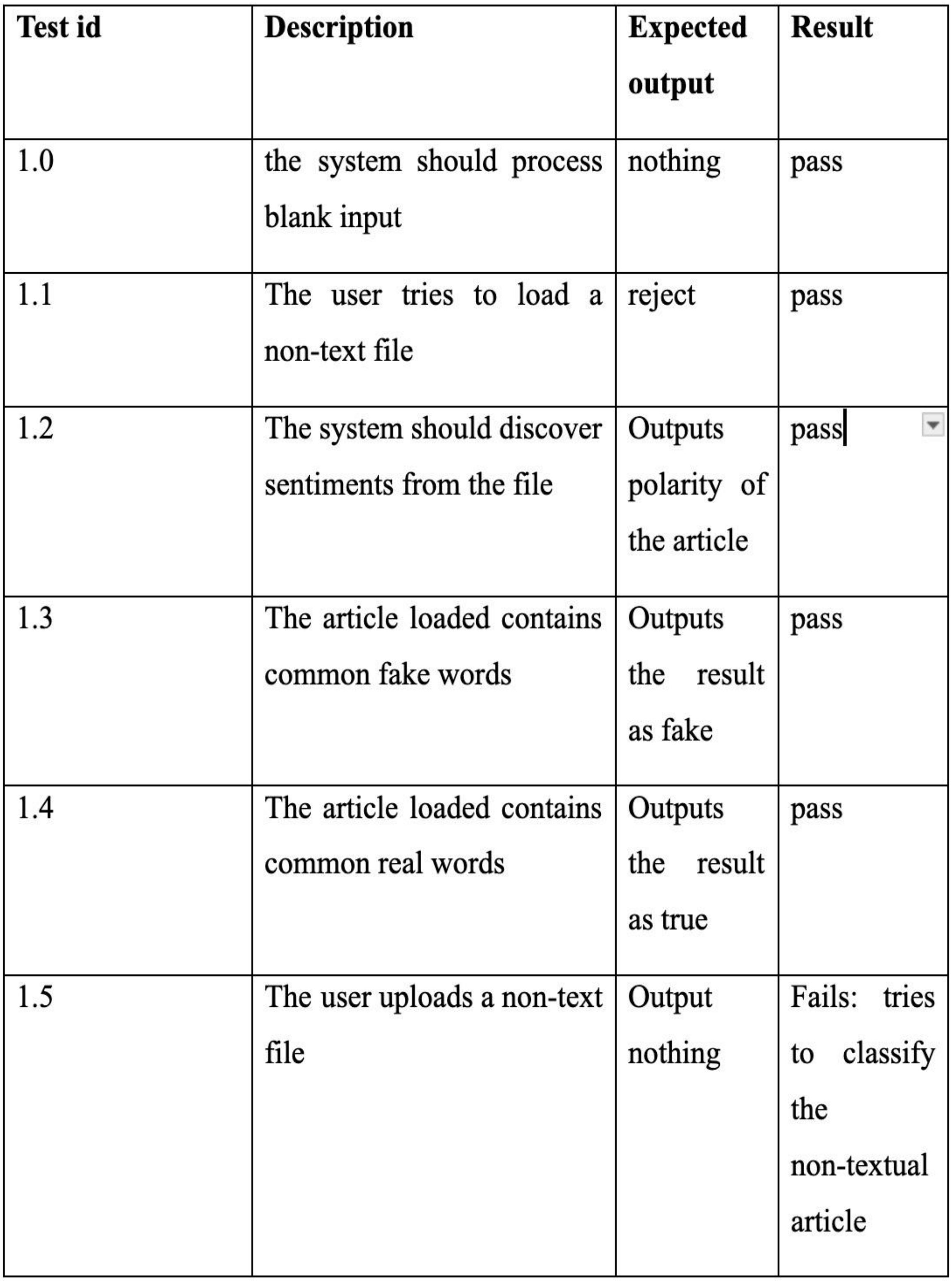
System Testing

System testing simply involves subjecting a system to function within different condition in order to discover how efficient the system is and to observe its behavior. This enables the creation of a system which maintains the same functionality all the time.

Test Cases and Results

The table depicted below, provides information about the test cases with their pass and failures:

Table 4.1 Test plan



Test id	Description	Expected output	Result
1.0	the system should process blank input	nothing	pass
1.1	The user tries to load a non-text file	reject	pass
1.2	The system should discover sentiments from the file	Outputs polarity of the article	pass
1.3	The article loaded contains common fake words	Outputs the result as fake	pass
1.4	The article loaded contains common real words	Outputs the result as true	pass
1.5	The user uploads a non-text file	Output nothing	Fails: tries to classify the non-textual article

FINDINGS

Following the methodology from chapter 3, when sentiment analysis was carried out on the online news article dataset, the result obtained from the polarities showed that some real news articles had negative polarities (sentiments) and also other fake news articles had negative sentiments. This contradicts the hypothesis that fake news articles normally carry negative sentiments and real news articles carry positive sentiments. Also it was discovered that the logistic regression classifier model had a very high accuracy which is prove that it performs better with lots of data. On the account of this finding, sentiment analysis wasn't used as a feature for the fake news detection model.

Classification Results

Table 4.2 classification results for different classification models

In addition to accuracy, we evaluated classifier performance using precision, recall, F1-score, and ROC-AUC.

Model	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Naïve Bayes	0.91	0.90	0.92	0.91	0.94
Logistic Regression	0.96	0.95	0.97	0.96	0.98
Random Forest	0.93	0.92	0.93	0.92	0.96

Evaluating the Model

The performance of this model is going to be calculated using the confusion matrix and the F1 score.

Confusion Matrix

A confusion matrix also known as an error matrix, is a performance measure for assessing the classification models such as the ones used in the building of the fake news detection model. The number of correct and incorrect predictions are summarized with count values and broken down by each class. It gives us an insight not only into the errors being made by a classifier but more importantly the types of errors that are being made. The confusion matrix has some terms that will help us understand this measure of performance:

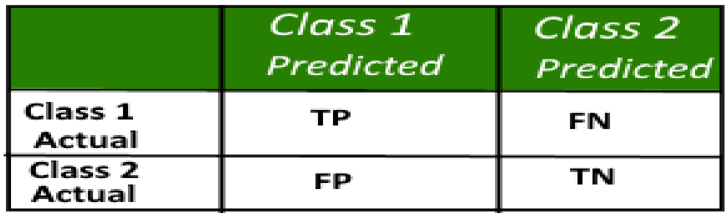
True Positive (TP): Outcome where the model correctly predicts the positive class.

True Negative (TN): Outcome where the model correctly predicts the negative class.

False Positive (FP, Type 1 error): Outcome where the model incorrectly predicts the positive class.

False Negative (FN, Type 2 error): Outcome where the model incorrectly predicts the negative class.

A confusion matrix is of the form:



	Class 1 Predicted	Class 2 Predicted
Class 1 Actual	TP	FN
Class 2 Actual	FP	TN

Figure 4.5 Confusion Matrix

Class 1: Positive

Class 2: Negative

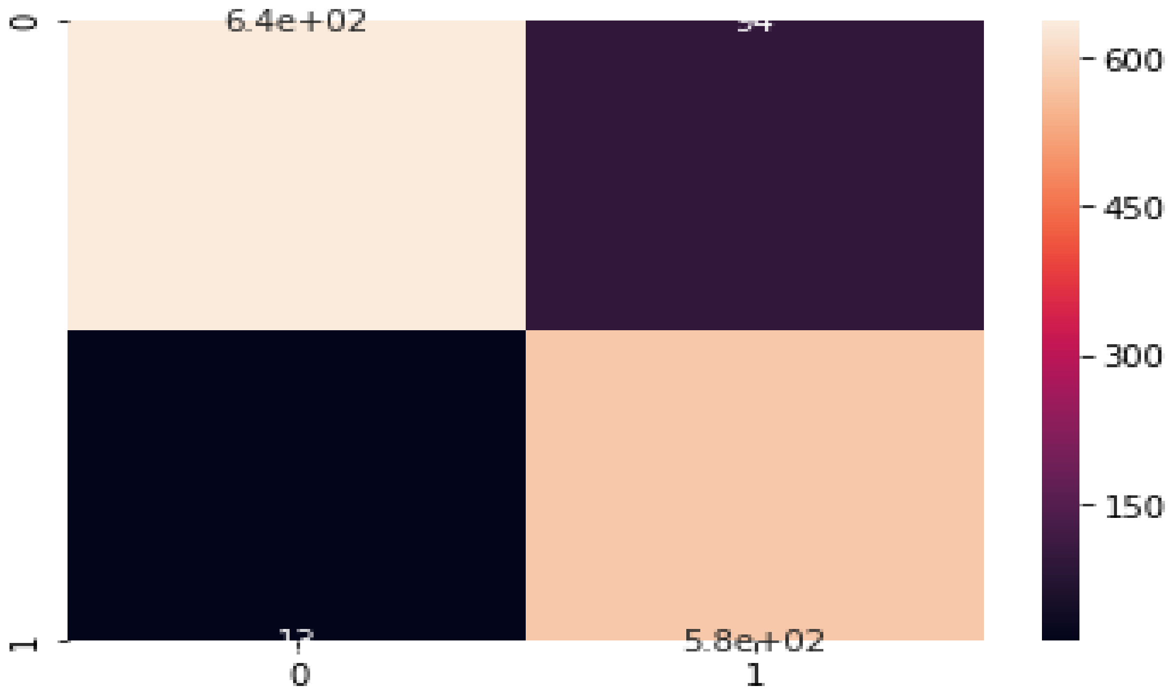


Figure 4.6 Confusion Matrix for Naive Bayes Classifier Model.

The confusion matrix of the naive Bayes classifier model shows that the model predicts more than half of the positive and negative classes correctly. The time in which it gets the classes wrong is very minute compared to its accuracy, this simply means that the model is performing well.

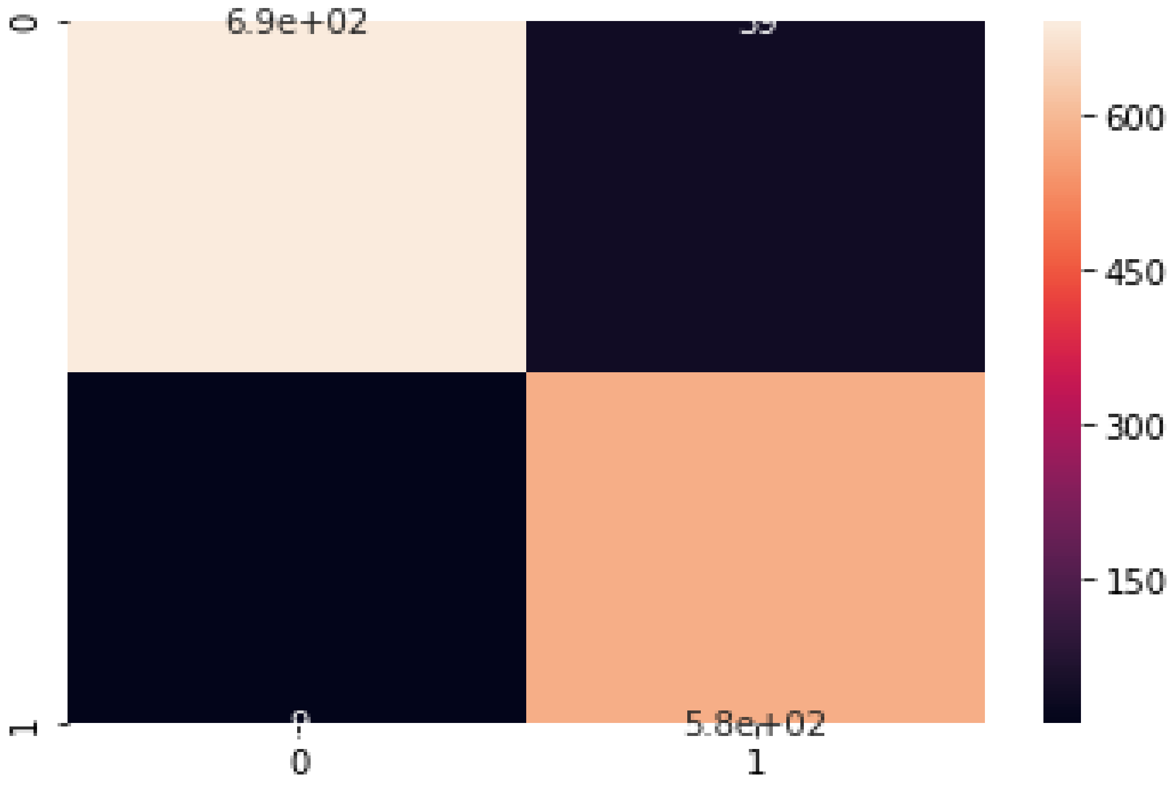


Figure 4.7 Confusion Matrix for Logistic Regression Classifier Model.

The confusion matrix of the logistic regression classifier model shows that the model predicts more than half of the positive and negative classes correctly. This model is very accurate compared to the naive Bayes classifier and the random forest classifiers and hence also has a higher f1 score.

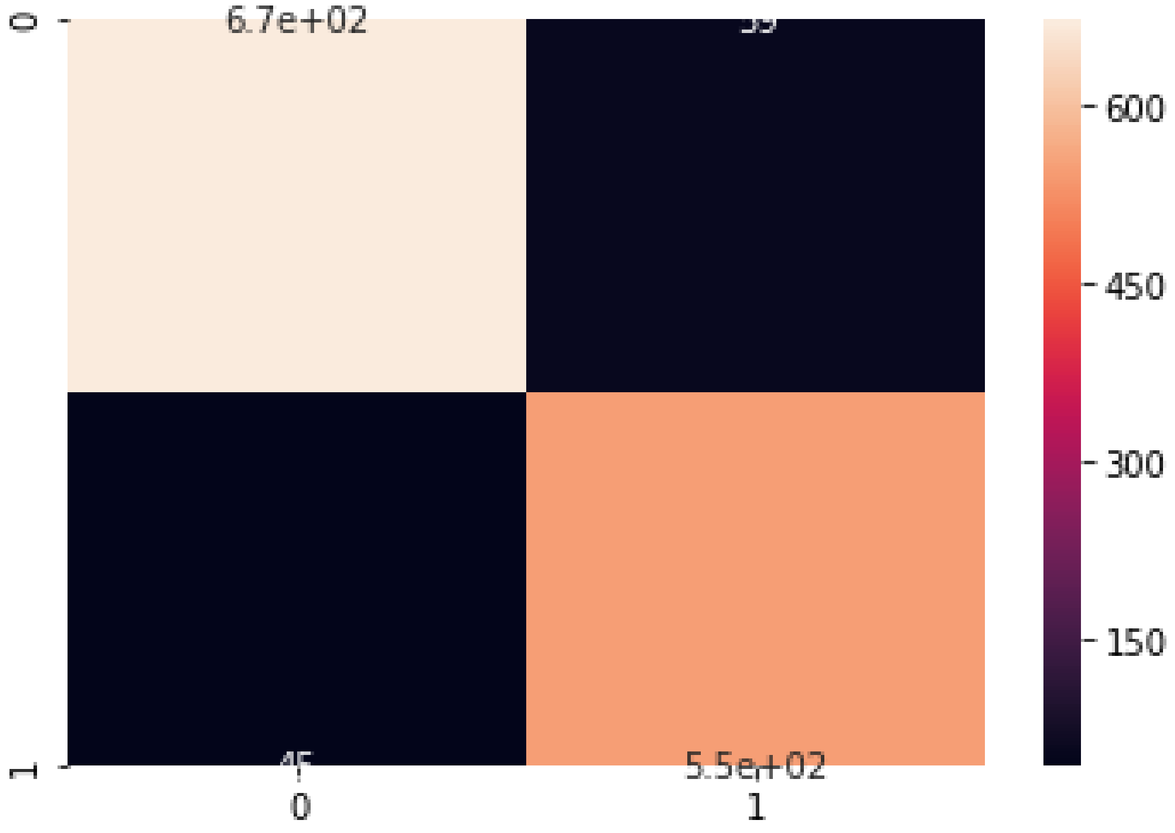


Figure 4.8 Confusion Matrix for Random Forest Classifier Model.

F1 Score

The F1 score is a measure of a test's accuracy - it is the harmonic mean of precision and recall. It can have a maximum score of 1 (perfect precision and recall) and a minimum of 0. Overall it is a measure of the preciseness and robustness of the model.

The F1 score is calculated as follows:

F1 = 2 *(precision * recall/precision + recall)

The result for the f1 score of the model classifiers is tabulated below:

Classifier	**F1 Score**
Naive Bayes	0.919
Logistic Regression	0.963
Random Forest	0.921

Model Interpretability

Understanding which features influence classification decisions is critical for trust and accountability. For this study: Feature importance was analyzed using the `coef_` attribute in logistic regression and `feature_importances_` in random forest. Top words contributing to fake classification included: 'shocking', 'claims', 'reportedly'. Real news was more associated with neutral or journalistic language like 'stated', 'confirmed', and 'update'. For deeper interpretability in future work, model-agnostic tools like SHAP and LIME will be explored.

Summary, conclusion and Future Work

Summary

At the inception of this research, several assumptions were made and also investigations were carried out to obtain more knowledge for this research work. The various results which were obtained from the various investigations provided the various requirements for which to continue with this work. The work carried out in this research aimed at developing a model that will help detect fake news articles online. Using regular expressions, the text document was able to be pre-processed and the unique features that are peculiar to real and fake news were able to be extracted using the various sklearn libraries as explained in section 3 of this study. In the end different models were developed using different binary classifiers, the models were evaluated and the most accurate model was obtained, tested and implemented. This section goes on to conclude this research work by summarizing the work done through achievements obtained and the future work which must be done to ensure the continuity of the research project.

CONCLUSION

Fact checking is a damage control strategy that is both essential and not scalable (Reis et al, 2019). It might be hard to take out the human component any time soon, especially if it involves any sensitive subjects such as politics. Automatic fake news detection such as this can be used by fact checkers as an auxiliary tool for identifying content that is most likely to be fake. The results obtained from this research shows that the predictive performance of our classifier has a high degree of detecting fake news. Our model can detect more than 90% of fake news in our dataset, which is very sufficient in helping fact checkers.

Future Work

In addition to existing directions, future work will focus on: Multilingual detection (extending the model to handle multiple languages), Multimodal detection (including images and videos), Bias mitigation (developing methods to reduce training bias), and Model interpretability tools (like LIME/SHAP) to better understand model behavior and support human fact-checkers.

REFERENCES

1. Conroy, N. J., Rubin, V. L., & Chen, Y. (2015). Automatic deception detection: Methods for finding fake news. Proceedings of the Association for Information Science and Technology, 52(1), 1-4.
2. Cui, L., & Lee, S. W. D. (2019). SAME: Sentiment-Aware Multi-Modal Embedding for Detecting Fake News.
3. Gurav, S., Sase, S., Shinde, S., Wabale, P., & Hirve, S. (2019). Survey on Automated System for Fake News Detection using NLP & Machine Learning Approach.
4. Monti, F., Frasca, F., Eynard, D., Mannion, D., & Bronstein, M. M. (2019). Fake News Detection on Social Media using Geometric Deep Learning. arXiv preprint arXiv:1902.06673.
5. Nakov, P., Ritter, A., Rosenthal, S., Sebastiani, F., & Stoyanov, V. (2019). SemEval- 2016 task 4: Sentiment analysis in Twitter. arXiv preprint arXiv:1912.01973.
6. Paul, S., Joy, J. I., Sarker, S., Ahmed, S., & Das, A. K. (2019, June). Fake News Detection in Social Media using Blockchain. In 2019 7th International Conference on Smart Computing & Communications (ICSCC) (pp. 1-5). IEEE.

7. Pérez-Rosas, V., Kleinberg, B., Lefevre, A., & Mihalcea, R. (2017). Automatic detection of fake news. arXiv preprint arXiv:1708.07104.
8. Reis, J. C., Correia, A., Murai, F., Veloso, A., Benevenuto, F., & Cambria, E. (2019). Supervised Learning for Fake News Detection. IEEE Intelligent Systems, 34(2), 76-81.
9. Rubin, V. L., Chen, Y., & Conroy, N. J. (2015, November). Deception detection for news: three types of fakes. In Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community (p. 83). American Society for Information Science.
10. Ruchansky, N., Seo, S., & Liu, Y. (2017, November). Csi: A hybrid deep model for fake news detection. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (pp. 797-806). ACM.
11. Shu, K., Bernard, H. R., & Liu, H. (2019). Studying fake news via network analysis: detection and mitigation. In Emerging Research Challenges and Opportunities in Computational Social Network Analysis and Mining (pp. 43-65). Springer, Cham.
12. Shu, K., Mahudeswaran, D., & Liu, H. (2019). FakeNewsTracker: a tool for fake news collection, detection, and visualization. Computational and Mathematical Organization Theory, 25(1), 60-71.
13. Shu, K., Sliva, A., Wang, S., Tang, J., & Liu, H. (2017). Fake news detection on social media: A data mining perspective. ACM SIGKDD Explorations Newsletter, 19(1), 22-36.
14. Yang, S., Shu, K., Wang, S., Gu, R., Wu, F., & Liu, H. (2019). Unsupervised fake news detection on social media: A generative approach. In Proceedings of 33rd AAAI Conference on Artificial Intelligence.
15. Zhou, X., & Zafarani, R. (2018). Fake news: A survey of research, detection methods, and opportunities. arXiv preprint arXiv:1812.00315.