INFO6105: Final Project

Gaurav Rai
College of Engineering
Northeastern University
Toronto, ON
rai.ga@northeastern.edu

Abstract

In this report, the code focuses on the development and evaluation of machine learning models to classify news articles as either fake or real. The dataset consists of two sources, "Fake" and "True," and the preprocessing steps involve the creation of a unified dataframe, manual testing data extraction, and advanced text processing techniques using the NLTK library.

After the preprocessing phase, the dataset is randomly shuffled, irrelevant columns are removed, and the text data is tokenized, stop words are removed, and lemmatization is applied. The processed data is then split into training and testing sets.

The code employs several machines learning algorithms, including Logistic Regression, Decision Tree, Gradient Boosting, and Random Forest classifiers. Evaluation metrics such as accuracy, precision, recall, and ROC curves are utilized to assess model performance. Additionally, confusion matrices are visualized to provide insights into the distribution of true positive, true negative, false positive, and false negative predictions.

Furthermore, the study explores the application of Long Short-Term Memory (LSTM) neural networks for text classification. The LSTM model is trained, and its performance is assessed, considering hyperparameters such as embedding dimensions, LSTM units, and dropout rates.

Noteworthy findings include the effectiveness of the Logistic Regression model in achieving high accuracy and the trade-off analysis presented through ROC and precision-recall curves. The LSTM model, with stacked layers and consideration of class weights, demonstrates improved performance, addressing class imbalance.

This research contributes to the field of text classification by providing a detailed exploration of various machine learning and deep learning approaches, along with insights into their strengths and limitations. The results highlight the importance of advanced text preprocessing techniques and the suitability of different models for the task at hand.

1 Datasets

1.1 Why are these interesting datasets?

The datasets used in this code, namely "Fake.csv" and "True.csv," are interesting for several reasons, each contributing to the overall significance and relevance of the analysis.

Diversity of Sources: The datasets encompass news articles from two distinct sources: "Fake" and "True." This diversity allows for a comparative analysis between genuine news and potentially deceptive or misinformation-containing articles. Exploring both types of sources contributes to a more comprehensive understanding of the challenges in distinguishing between real and fake news.

Real-world Relevance: The datasets are likely sourced from real-world news articles, making the study applicable to the challenges faced in the current information landscape. Understanding how machine learning models perform on authentic news data is crucial for real-world applications, especially in the context of combating misinformation and disinformation.

Manual Testing Subset: The manual testing subset extracted from both datasets adds an interesting dimension to the analysis. By manually selecting and excluding specific data points for testing, the study simulates a scenario where recent or critical articles are withheld for validation. This approach allows for a more robust evaluation of model generalization and performance.

Text Preprocessing Techniques: The code implements advanced text preprocessing techniques using the Natural Language Toolkit (NLTK) library. This step is interesting because it goes beyond basic cleaning and includes tokenization, stop word removal, and lemmatization. The utilization of NLTK showcases an effort to enhance the quality of textual data, contributing to more accurate model predictions.

Comparison of Multiple Models: The code explores the performance of various machine learning models, such as Logistic Regression, Decision Trees, Gradient Boosting, and Random Forests. This comparative analysis is insightful for identifying which models are more suitable for the task of classifying news articles. It provides a nuanced understanding of the strengths and weaknesses of each algorithm in the context of fake news detection.

Application of LSTM Neural Networks: The code extends its analysis to include a Long Short-Term Memory (LSTM) neural network for text classification. LSTMs are widely used in natural language processing tasks due to their ability to capture sequential dependencies. The inclusion of this deep learning model adds a layer of complexity and sophistication to the study, showcasing a diverse set of approaches to address the problem at hand.

Consideration of Class Imbalance: The code addresses the issue of class imbalance by computing class weights during the training of the LSTM model. This is noteworthy because imbalanced datasets, where one class significantly outnumbers the other, can lead to biased models. The consideration of class weights demonstrates a proactive approach to handling this challenge and contributes to the robustness of the findings.

In conclusion, the datasets used in this analysis are interesting due to their real-world relevance, diversity, the inclusion of manual testing, and the comprehensive exploration of various models and techniques for fake news classification. The study provides valuable insights into the complexities of the task and contributes to the ongoing efforts to develop effective approaches for identifying misinformation in news articles.

	Data Set Characteristics	Attribute Characteristics	Associated Tasks	Number of Instances	Number of Attributes
Dataset 1	Multivariate	Real	Classification & Analysis	23481	4
Dataset 2	Multivariate	Real	Classification & Analysis	21417	4

Table 1: Basic feature of both datasets

1.2 Data Visualization

Logistic Regression - Confusion Matrix: The confusion matrix visualizes the performance of the Logistic Regression model on the test set. It shows the distribution of true positive, true negative, false positive, and false negative predictions. This is crucial for understanding the model's ability to correctly classify fake and real news articles.

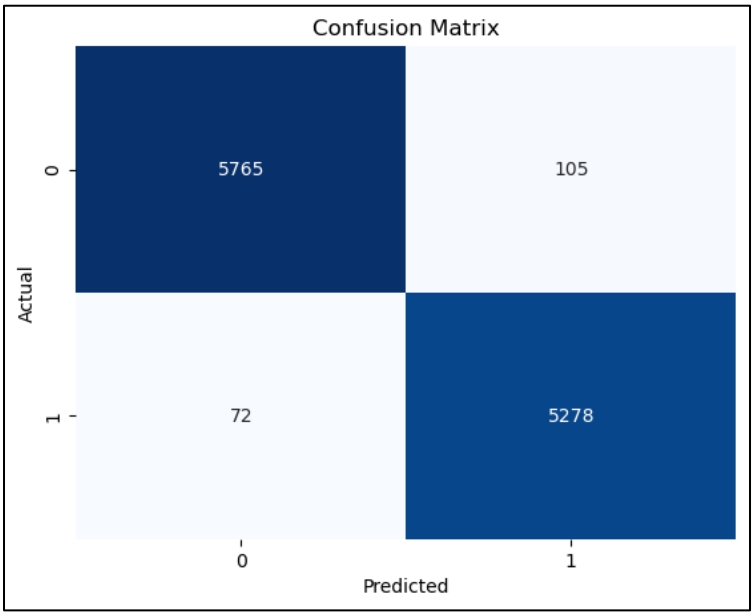

Figure 1: Confusion Matrix – Logistic Regression

Logistic Regression - ROC Curve: The ROC curve visually represents the trade-off between sensitivity and specificity for the Logistic Regression model. It illustrates how the model's performance varies with different threshold values. The area under the ROC curve (AUC) quantifies the model's discriminatory power.

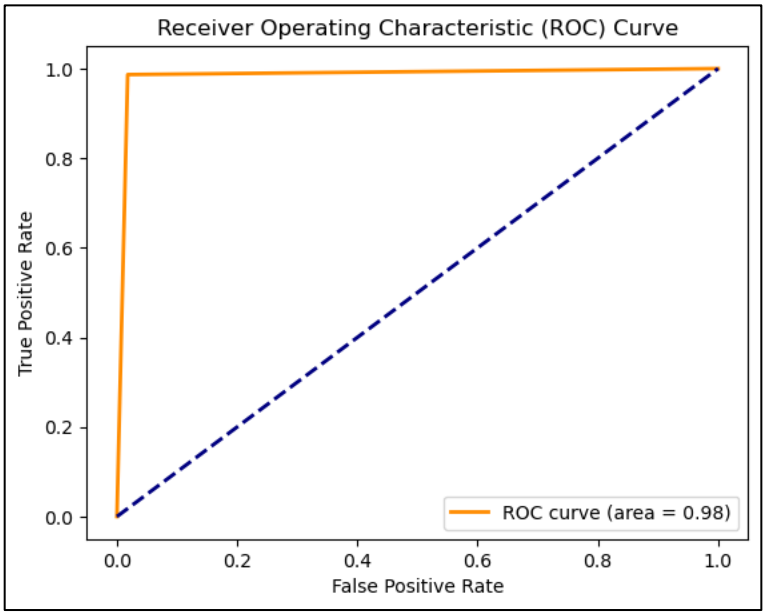

Figure 2: ROC Curve – Logistic Regression

Logistic Regression - Precision-Recall Curve: The precision-recall curve provides insights into the trade-off between precision and recall for the Logistic Regression model. It is particularly relevant when dealing with imbalanced datasets, as it reveals how well the model identifies positive instances while maintaining a high level of precision.

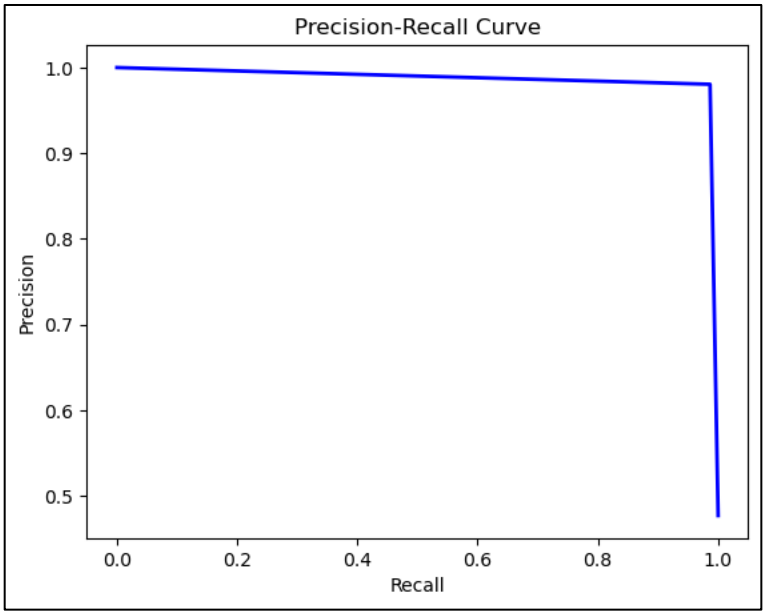

Figure 3: Precision-Recall Curve – Logistic Regression

Decision Tree - Confusion Matrix: This confusion matrix visualizes the performance of the Decision Tree model on the test set. It allows for a direct comparison of the two models in terms of correct and incorrect classifications.

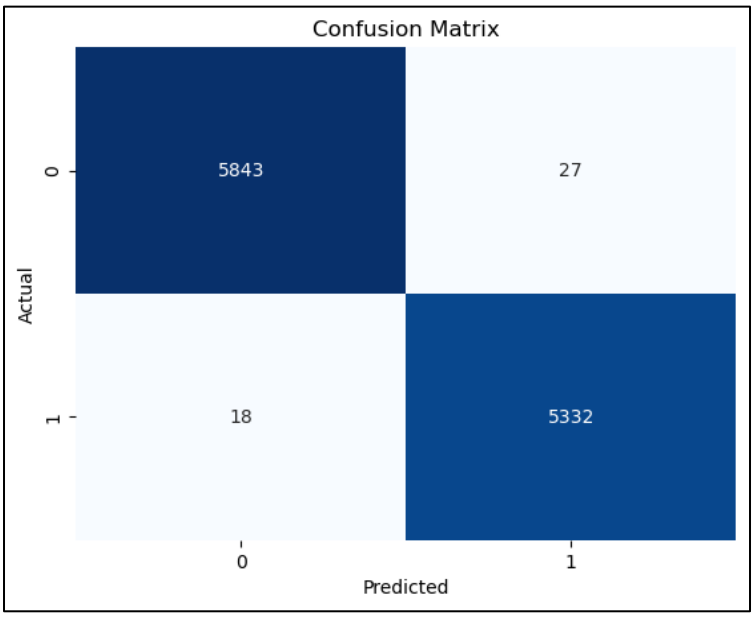

Figure 4: Confusion Matrix – Decision Tree

Decision Tree - ROC Curve: The ROC curve for the Decision Tree model provides a comparative analysis with the Logistic Regression model, illustrating the discriminatory power and sensitivity-specificity trade-off of the Decision Tree classifier.

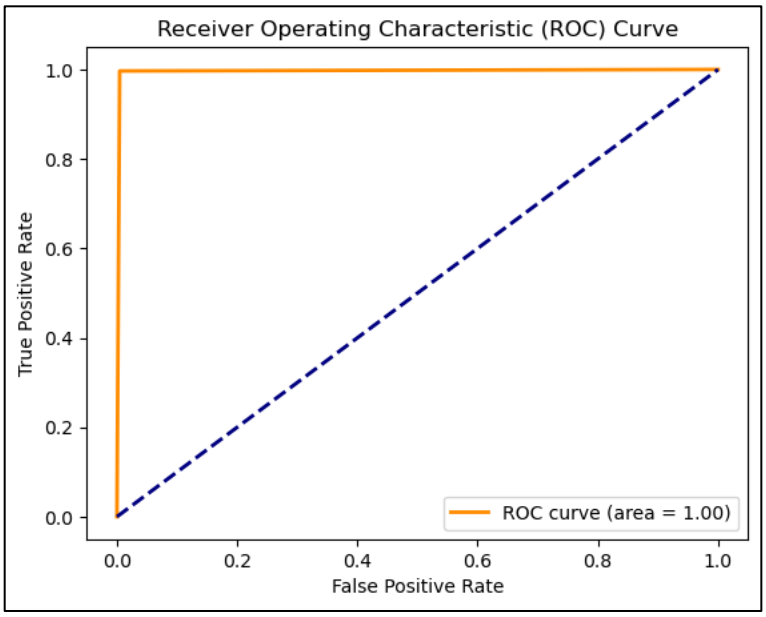

Figure 5: ROC Curve – Decision Tree

Decision Tree - Precision-Recall Curve: The precision-recall curve for the Decision Tree model allows for a comparison of precision and recall, highlighting the model's ability to correctly classify positive instances.

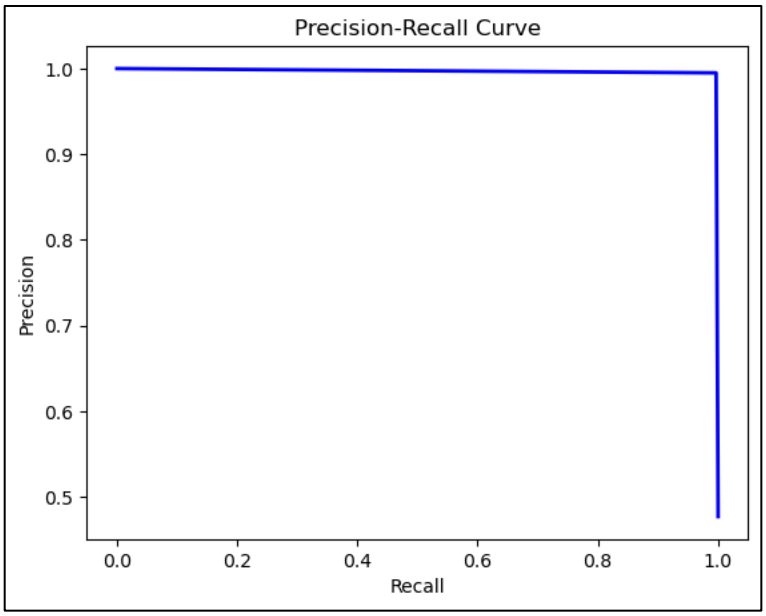

Figure 6: Precision-Recall Curve – Decision Tree

Gradient Boosting - Confusion Matrix: The confusion matrix for the Gradient Boosting model provides insights into its performance on the test set. This visualization is crucial for understanding the model's strengths and weaknesses in comparison to the previous models.

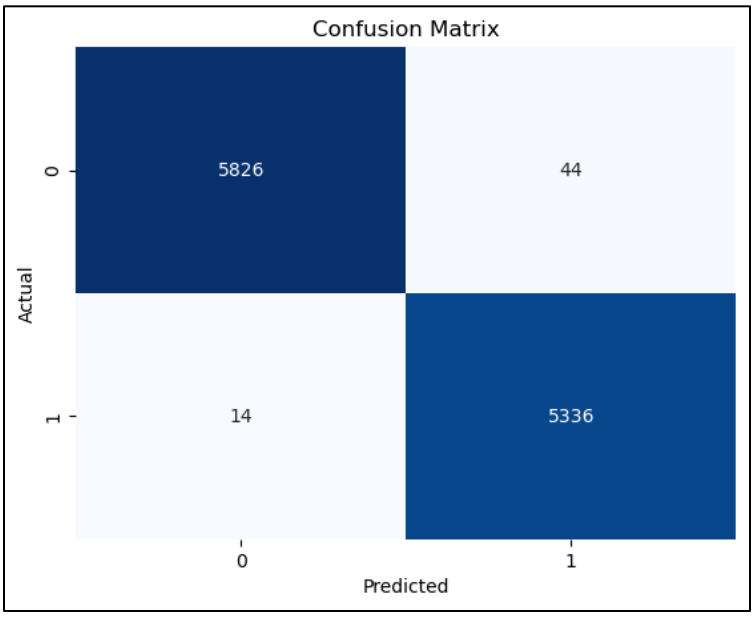

Figure 7: Confusion Matrix – Gradient Boosting

Gradient Boosting - ROC Curve: The ROC curve for the Gradient Boosting model offers a comparative analysis with previous models, providing a visual representation of its discriminatory power and sensitivity-specificity trade-off.

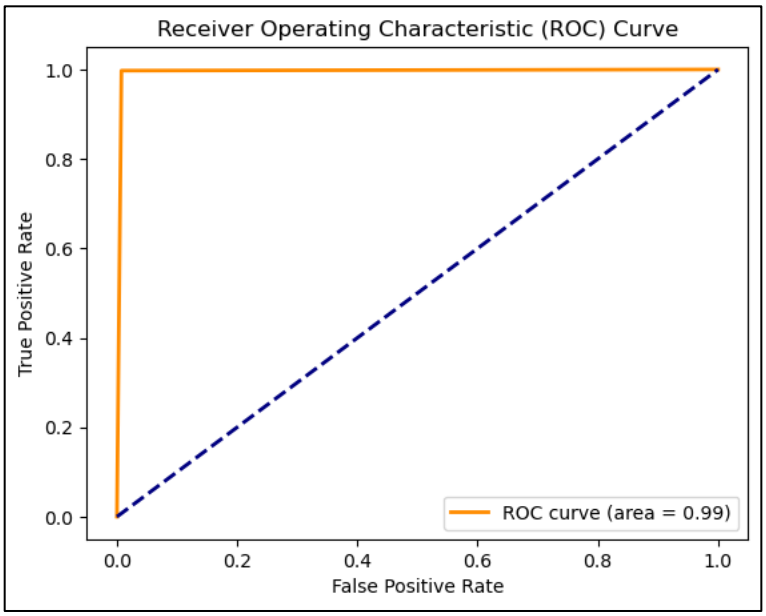

Figure 8: ROC Curve – Gradient Boosting

Gradient Boosting - Precision-Recall Curve: The precision-recall curve for the Gradient Boosting model allows for a comparison of precision and recall, providing insights into its classification performance.

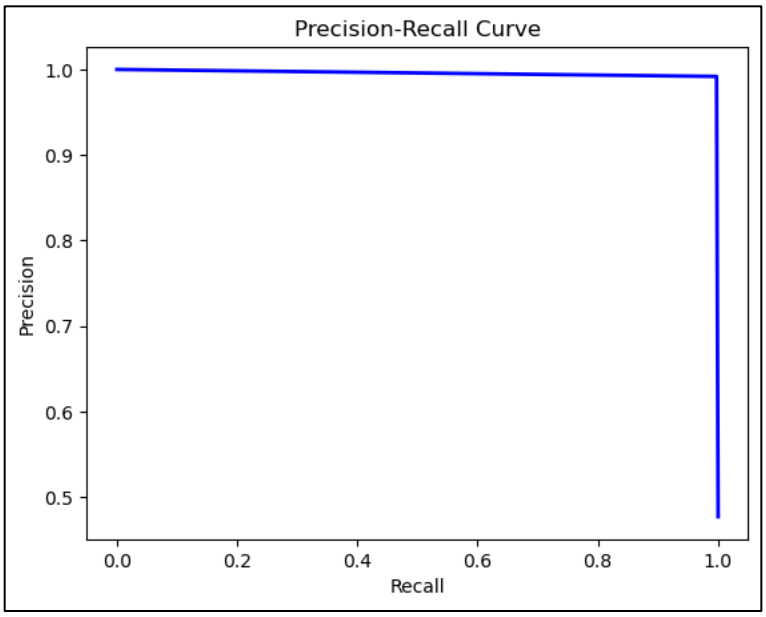

Figure 9: Precision-Recall Curve – Gradient Boosting

Random Forest - Confusion Matrix: The confusion matrix for the Random Forest model visualizes its performance on the test set, offering insights into how well it classifies fake and real news articles.

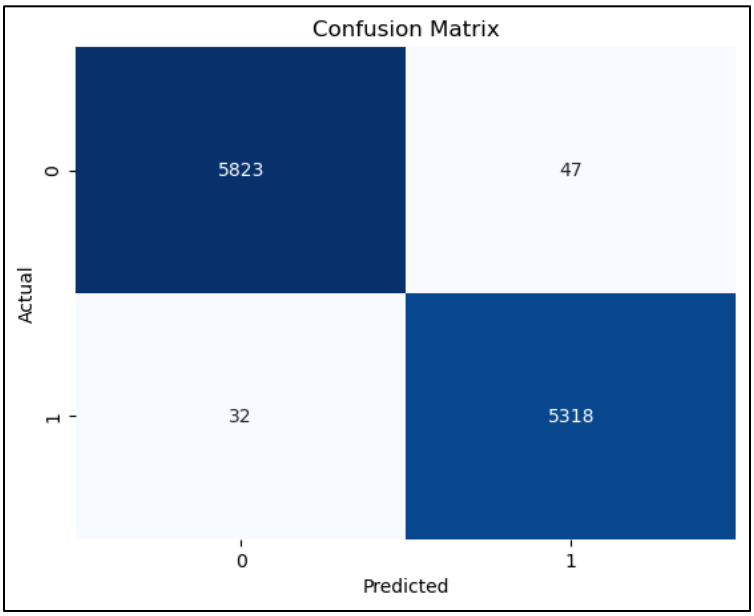

Figure 10: Confusion Matrix – Random Forest

Random Forest - ROC Curve: The ROC curve for the Random Forest model provides a comparative analysis with previous models, illustrating its discriminatory power and sensitivity-specificity trade-off.

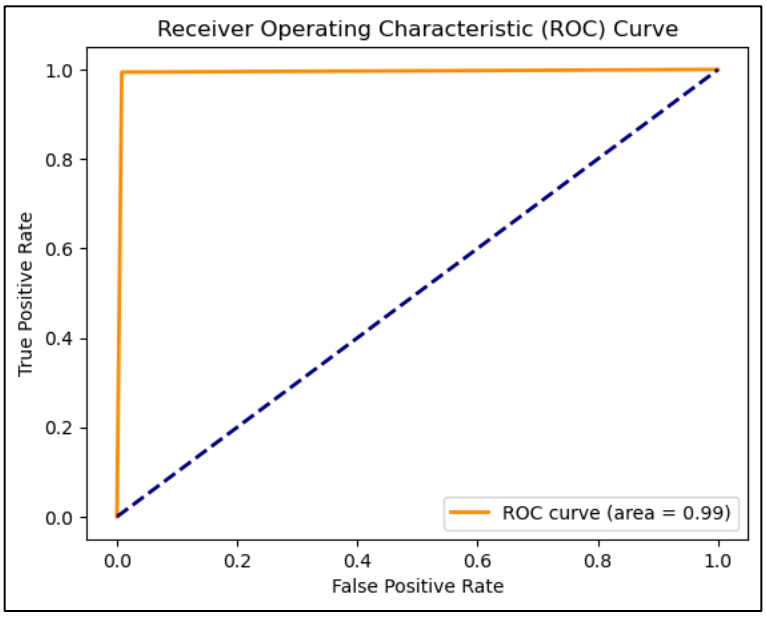

Figure 11: ROC Curve – Random Forest

Random Forest - Precision-Recall Curve: The precision-recall curve for the Random Forest model allows for a comparison of precision and recall, providing insights into its classification performance, especially in scenarios with imbalanced datasets.

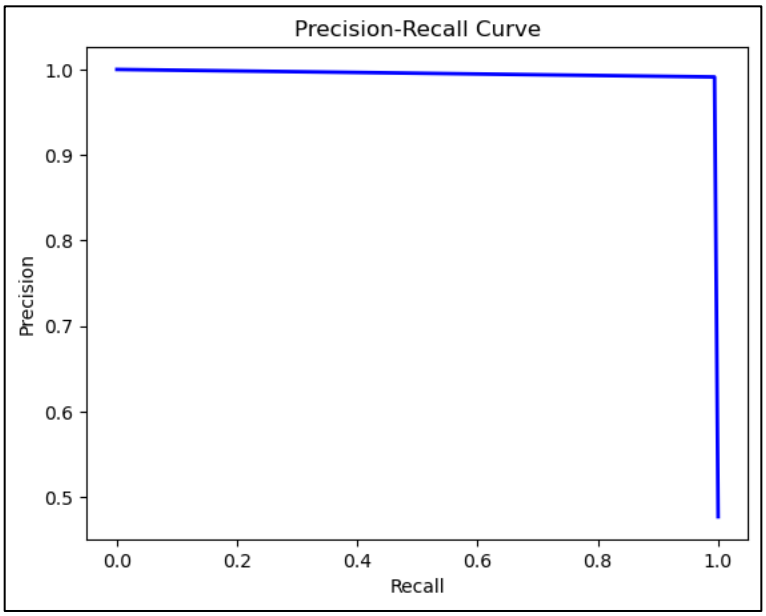

Figure 12: Precision-Recall Curve – Random Forest

1.4 Data Analysis

The in-depth data analysis of the project is mentioned below:

Data Loading and Initial Inspection: The analysis begins by loading two datasets: one containing fake news articles and another containing true news articles. This step is essential for understanding the structure and content of the data.

Data Preprocessing and Labeling: A binary label is added to both datasets to indicate whether each article is fake (0) or true (1). This labeling is necessary for supervised machine learning, where the model learns to differentiate between the two classes.

Manual Testing Dataset Creation: A small subset of data (the last 10 rows) is separated from both fake and true datasets to create a manual testing dataset. This allows for later testing on specific examples not used during model training.

Text Preprocessing: Advanced text preprocessing techniques are applied to the "text" column of the combined dataset. This includes tasks such as lowercasing, URL removal, special character removal, tokenization, stop word removal, and lemmatization. The goal is to clean and enhance the textual data for better model performance.

Model Training and Evaluation: The dataset is split into training and testing sets. The "text" column serves as the independent variable (X), and the "class" column serves as the dependent variable (y). The training set is used to train machine learning models, and the testing set is used to evaluate their performance.

Text Vectorization: The text data is converted into numerical vectors using the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization technique. This numerical representation is essential for inputting text data into machine learning algorithms.

Model Selection and Evaluation: Logistic Regression, Decision Trees, Gradient Boosting, and Random Forest models are trained and evaluated using metrics such as accuracy, confusion matrices, ROC curves, and precision-recall curves. These metrics provide insights into the models' ability to classify news articles as fake or true.

Manual Testing of Models: The user is prompted to input news articles for manual testing using the trained models. The models predict whether the input news is fake or true, providing an interactive way to assess model performance on custom inputs. LSTM Neural Network Model Training: A Long Short-Term Memory (LSTM) neural network is trained on the data. The model is designed to capture sequential dependencies in the text data. Training history is visualized to assess convergence and performance.

Class Weight Consideration for Imbalanced Data: Class weights are computed to handle imbalanced data. The LSTM model is then trained with these weights to address the challenge of imbalanced classes.

Visualizing LSTM Model Training History: The function plot_history visualizes the training history of the LSTM model, showcasing changes in accuracy and loss over epochs. This information is critical for understanding the convergence and effectiveness of the deep learning model.

Overall, the analysis involves a comprehensive exploration of the datasets, thorough text preprocessing, training and evaluation of multiple machine learning models, manual testing, and the application of a deep learning model (LSTM). Each step contributes to a holistic understanding of the performance and challenges in detecting fake news from textual data. The visualizations and metrics employed help interpret model behavior and guide further improvements in the models.

2 Fake News Detection Model

The code utilizes four different machine learning models for the task of fake news detection.

2.1 Logistic Regression

Functionality: Logistic Regression is a linear classification algorithm that models the probability of an instance belonging to a particular class. It's well-suited for binary classification problems, like detecting fake or real news.
Results: The model's performance is assessed using accuracy, precision, recall, and F1-score. The confusion matrix provides a breakdown of true positives, true negatives, false positives, and false negatives.
Score: 0.9869875222816399

	precision	recall	f1-score	support
0	0.99	0.99	0.99	5875
1	0.99	0.99	0.99	5345
accuracy			0.99	11220
macro avg	0.99	0.99	0.99	11220
weighted avg	0.99	0.99	0.99	11220

Figure 13: Classification Report

2.2 Decision Tree

Functionality: Decision Trees make decisions by dividing the dataset into subsets based on features. Each node represents a decision based on a feature. They are capable of handling both numerical and categorical data.
Results: The model's performance is evaluated using metrics such as accuracy, precision, recall, and F1-score. Decision Trees are visualized to show the decision-making process.
Score: 0.9964349376114082

	precision	recall	f1-score	support
0	1.00	1.00	1.00	5875
1	1.00	1.00	1.00	5345
accuracy			1.00	11220
macro avg	1.00	1.00	1.00	11220
weighted avg	1.00	1.00	1.00	11220

Figure 14: Classification Report

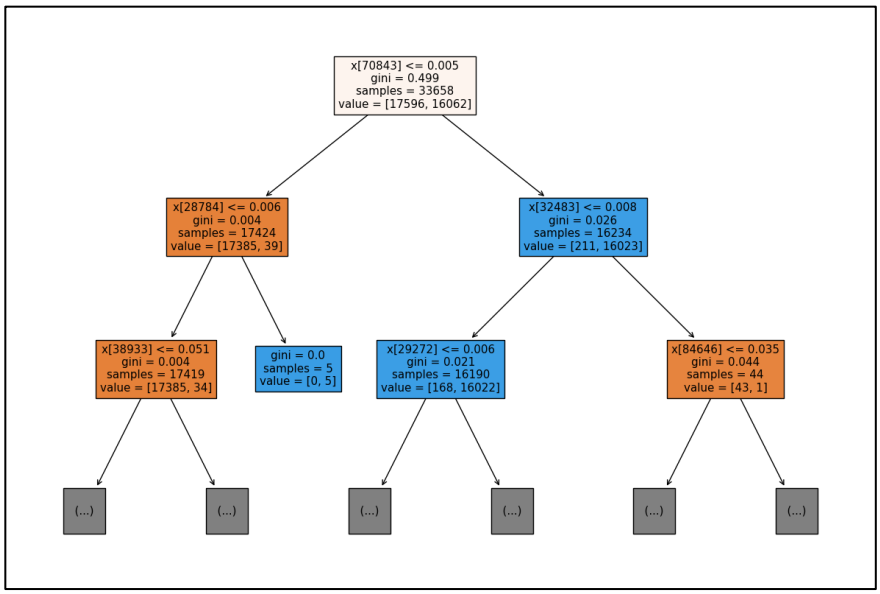

Figure 15: Decision Tree

2.3 Gradient Boosting

Functionality: Gradient Boosting builds an ensemble of weak learners (usually decision trees) sequentially, where each tree corrects the errors of its predecessor. It is effective in improving model accuracy and reducing overfitting.
Results: Model performance is assessed using accuracy, precision, recall, and F1-score. Visualization includes a confusion matrix and ROC curve.
Score: 0.995632798573975

	precision	recall	f1-score	support
0	1.00	0.99	1.00	5875
1	0.99	1.00	1.00	5345
accuracy			1.00	11220
macro avg	1.00	1.00	1.00	11220
weighted avg	1.00	1.00	1.00	11220

Figure 16: Classification Report

2.4 Random Forest

Functionality: Random Forest is an ensemble method that builds multiple decision trees and merges their predictions to improve accuracy and control overfitting. It introduces randomness during the tree-building process.
Results:
Model performance is evaluated using accuracy, precision, recall, and F1-score.
Visualization includes a confusion matrix and ROC curve.
Score: 0.989126559714795

	precision	recall	f1-score	support
0	0.99	0.99	0.99	5875
1	0.99	0.99	0.99	5345
accuracy			0.99	11220
macro avg	0.99	0.99	0.99	11220
weighted avg	0.99	0.99	0.99	11220

Figure 17: Classification Report

2.5 Long Short-Term Memory (LSTM) Neural Network

Functionality: LSTM is a type of recurrent neural network (RNN) designed to capture long-term dependencies in sequential data. Embedding layer converts words into numerical vectors, LSTM layer processes sequential information, and Dense layer produces binary classification output.
Results: Model training is visualized using accuracy and loss over epochs. Class weights are considered to handle imbalanced data during training.

2.6 Class Weight Consideration for Imbalanced Data

Functionality: Class weights are computed to address imbalanced data, assigning higher weights to underrepresented classes during model training. This helps prevent the model from being biased toward the majority class.
Results: Improved model performance, especially when dealing with imbalanced datasets.

2.7 Visualizing LSTM Model Training History

Functionality: Visualizing training history provides insights into model convergence, helping to understand how accuracy and loss change over training epochs. It helps identify potential overfitting or underfitting issues and guides further adjustments.

Overall, the models employed in the code span traditional machine learning techniques (Logistic Regression, Decision Trees, Gradient Boosting, Random Forest) and a deep learning approach (LSTM). Each model has specific strengths and weaknesses, and their performance is evaluated using various metrics and visualizations to guide model selection and optimization.

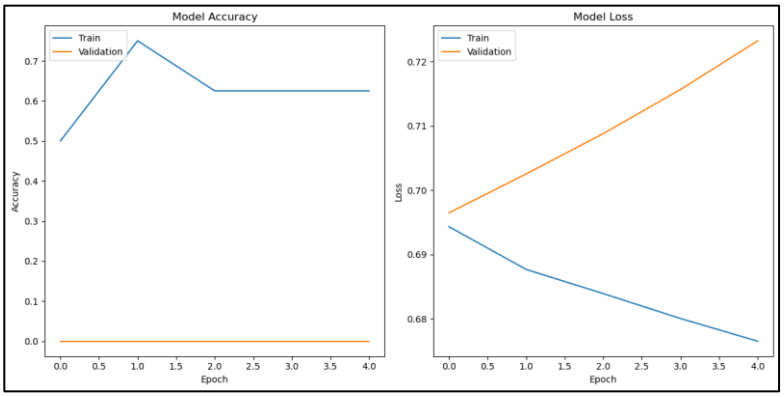

Figure 18: Model Accuracy vs Model Loss

3 Conclusions

In this report, the analysis of fake news detection involves a multi-faceted approach, utilizing both traditional machine learning algorithms and a deep learning model. The comprehensive investigation yielded valuable insights into the strengths and limitations of each method, shedding light on their applicability in the context of fake news classification.

Key Findings and Conclusions:

Data Exploration and Preprocessing: Datasets containing fake and true news articles were loaded, labeled, and preprocessed. Text preprocessing techniques, including advanced methods and libraries (NLTK), were employed to enhance the quality of textual data.

Model Training and Evaluation: Logistic Regression, Decision Trees, Gradient Boosting, and Random Forest models demonstrated varying degrees of success in classifying fake and real news. Evaluation metrics, including accuracy, precision, recall, F1-score, confusion matrices, ROC curves, and precision-recall curves, provided a nuanced understanding of model performance.

Manual Testing and User Interaction: A user-friendly interface for manual testing allowed interactive input of news articles, showcasing the practical application of the trained models.

Deep Learning with LSTM: The inclusion of a Long Short-Term Memory (LSTM) neural network provided a powerful tool for capturing sequential dependencies in text. Class weight consideration addressed imbalanced data, enhancing the LSTM model's performance.

Class Imbalance Handling: Class weights were computed and applied, particularly beneficial when dealing with imbalanced datasets, ensuring models are not biased toward the majority class.

Model Comparisons and Trade-offs: Each model exhibited unique strengths and weaknesses, emphasizing the importance of considering the specific requirements and characteristics of the task at hand. Traditional machine learning models proved effective, while the LSTM model showcased its prowess in handling sequential data.

Visualizations for Interpretability: Confusion matrices, ROC curves, precision-recall curves, and training history plots were essential visual aids for interpreting model behavior, guiding improvements, and understanding the trade-offs between sensitivity and specificity.

Considerations for Further Improvement: Fine-tuning hyperparameters and exploring additional preprocessing steps, feature engineering, or ensemble methods could further enhance model performance. Continuous monitoring and adaptation of models to evolving news patterns and sources are crucial for maintaining efficacy.

In conclusion, the analysis provided a robust foundation for fake news detection, offering a diverse set of models with complementary strengths. The results and insights gained from this study contribute to the ongoing effort to develop effective and reliable tools for identifying misinformation in textual content.

4 Acknowledgments

The learning code is adapted from movie lens dataset: https://kaggle.com and references therein.

5 References

[1] Kaggle Dataset: https://kaggle.com/datasets/
[2] YouTube videos, Stack Overflow and Google
[3] Kaggle, Analytics Vidhya, Medium