Internship Project Report: Fake vs Real News Classifier

Name: Uribindi Dharshini
Internship Role: Artificial Intelligence Intern
Project Title: Fake vs Real News Classification using NLP

---

1. Introduction

The proliferation of fake news is one of the most pressing challenges of our digital era. With social media platforms acting as conduits for information—both accurate and misleading—the ability to automatically identify and filter out fake news has become an essential component of responsible digital communication. This project focuses on developing an intelligent system that leverages Natural Language Processing (NLP) and Machine Learning (ML) to classify news articles as either fake or real. The core aim is to combine data science principles and real-world deployment techniques to build a scalable and functional fake news classifier.

---

2. Abstract

This project involves designing and deploying a robust machine learning pipeline for classifying news content based on textual features. Starting from raw data ingestion, through preprocessing and vectorization, to model training and deployment, the project delivers a full-stack ML solution that demonstrates the practical application of NLP.
Using a dataset consisting of labeled real and fake news articles, the text data undergoes cleaning (including tokenization and stopword removal) and transformation using TF-IDF vectorization. The core model used is Logistic Regression, chosen for its balance of speed, interpretability, and high accuracy. Evaluation metrics such as F1-score and accuracy (~92%) validate the model's performance.
The trained model is then integrated into a web application developed with Streamlit. This app provides an interactive platform where users can input any news article and instantly receive predictions, confidence scores.

---

3. Tools Used

- **Programming Language:** Python 3
- **Libraries:** Scikit-learn, Pandas, NumPy, NLTK, Joblib
- **Text Processing:** NLTK for tokenization and stopword filtering
- **Vectorization:** TF-IDF using Scikit-learn
- **ML Models:** Logistic Regression, Naive Bayes
- **Web Framework:** Streamlit
- **Deployment:** Streamlit Cloud (hosted via GitHub)
- **Visualization:** Matplotlib (for local evaluation)
- **Data Source:** Kaggle - Fake and Real News Dataset

---

4. Steps Involved in Building the Project

1. **Data Collection:** Merged two structured datasets—Fake.csv and True.csv—and appended a binary label to each (1 for real, 0 for fake). Inspected and balanced the dataset for modeling.
2. **Text Preprocessing:** Used NLTK to clean the data, tokenize sentences, remove punctuation, and eliminate stopwords. This stage was crucial for transforming unstructured raw text into clean, usable tokens.
3. **Feature Engineering:** Implemented TF-IDF vectorization to extract relevant features from the text while maintaining computational efficiency. Limited to the top 5000 features for optimal performance.
4. **Model Development:** Trained and compared Logistic Regression and Naive Bayes classifiers. The Logistic model delivered higher precision, with an average accuracy score of ~92% and better generalization.
5. **Performance Evaluation:** Applied evaluation metrics including accuracy, F1-score, and confusion matrix. Crossvalidation was conducted to ensure consistency of performance across unseen data.
6. **Web App Development:** Built an interactive front-end using Streamlit. Included a clean layout, light/dark themes, keyword displays from the TF-IDF vector, and a prediction confidence bar. Focused on clarity, accessibility, and responsiveness.
7. **Deployment:** Connected the project to GitHub and deployed it to Streamlit Cloud. Prepared a requirements.txt for dependency management. Integrated nltk.download() within the script to support necessary resources like 'punkt', ‘punkt_tab’ and 'stopwords'.

---

5. Conclusion

This project is a strong example of how AI and machine learning can be applied to solve real-world challenges like fake news detection. It showcases a full-stack approach to model development—from data cleaning to deployment—and demonstrates the importance of explainable AI through keyword relevance and prediction confidence.

Not only did the project meet technical milestones such as high model accuracy and real-time inference, but it also delivered a professional-grade UI that can serve as a prototype for future product development. The knowledge gained from handling data bottlenecks, Streamlit customization, and GitHub deployment enhances confidence in handling end-to-end ML pipelines in a production-like setting.

Submitted by: Uribindi Dharshini

LinkedIn: linkedin.com/in/dharshini-u

GitHub:github.com/Dha11rshini/News_app_classifier