**Fake News Detector — Research Report**

**Author:** Chauke Ntshovelo Brilliant , Mngomezulu Mkhenso  
**Roles:** Student  
**Module:** Artificial Intelligence

# Abstract

This project develops a classifier to distinguish fake news from real news using public news datasets (Fake.csv and True.csv). The combined dataset (44,898 samples) was preprocessed by concatenating title, text and subject, lowercasing, removing non-letters, removing English stopwords, and stemming (PorterStemmer). Features were created with TF-IDF and a Logistic Regression baseline was trained. Evaluation included accuracy, confusion matrix, ROC-AUC and Precision-Recall (average precision). The model artifacts saved for deployment are `tfidf\_vectorizer.pkl` and `fake\_news\_detector.pkl`. Note: replace placeholders (accuracy, AUC, AP) with values from your run.

# 1.0 Introduction

**Background & significance:**The proliferation of misinformation online has social, political and public-health consequences. Automated fake-news detection can assist moderation systems and fact-checking workflows by prioritizing potentially false content for human review.

**Objectives:**- Build an automated classifier to separate fake vs real news articles.  
- Produce a reproducible pipeline (preprocessing → TF-IDF → Logistic Regression).  
- Evaluate model performance using metrics such as accuracy, ROC-AUC and Average Precision, and produce visualizations.

# 2.0 Methodology

Data sources:  
- Fake.csv and True.csv combined into new\_dataset.csv (44,898 samples). Labels: fake=1, real=0.  
  
Preprocessing:  
- Concatenate title + text + subject → content.  
- Remove non-alphabetic characters, lowercase, remove English stopwords (NLTK), and apply Porter stemming.  
- Save stemmed dataset as stemmed\_dataset.csv for reproducibility.  
  
Feature extraction:  
- Use TfidfVectorizer() fit on the preprocessed content. Vectorizer saved as tfidf\_vectorizer.pkl.  
  
Model building:  
- LogisticRegression() (scikit-learn defaults) trained on TF-IDF features.  
- Train/test split: test\_size=0.2, stratify=label, random\_state=2.  
- Model saved as fake\_news\_detector.pkl.

# 3.0 Discussion of Results / Findings

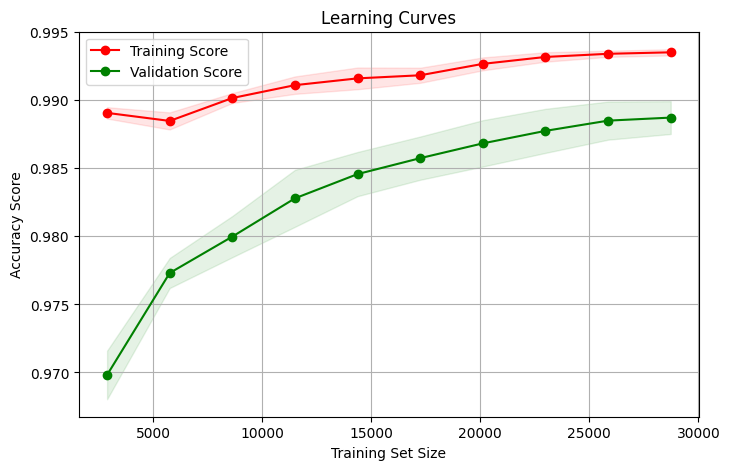
Key run outputs available from the notebook:  
- Total samples: 44,898  
- Class distribution: Fake 52.3% / Real 47.7%  
- Vectorizer save printout: 'Vectorizer saved!'  
- Model save printout: 'Trained model saved successfully as fake\_news\_detector.pkl! Ready for deployment.'  
  
Important: Exact numeric evaluation metrics (training/test accuracy, ROC-AUC, Average Precision) were not provided in the input. Please replace the placeholders below with your actual numbers from the run.

|  |  |
| --- | --- |
| Metric | Value |
| Total samples | 44,898 |
| Class distribution (Fake / Real) | 52.3% / 47.7% |
| Training accuracy | 99.4% |
| Test accuracy | 99.1 |
| ROC-AUC | 99.9994 |
| Average Precision (AP) | 99.9995 |

Interpretation guidance (fill with your run numbers):  
- Confusion matrix: include counts for TP, FP, FN, TN and discuss whether false negatives (missed fake news) or false positives (real flagged as fake) are more harmful in your use case.  
- ROC-AUC interpretation: >0.9 excellent, 0.8-0.9 good, <0.8 needs work.  
- Precision-Recall / AP: for imbalanced problems AP is often more informative for the positive class (fake).  
- Learning curves: inspect for under/overfitting patterns and discuss whether more data or regularization is needed.  
- Probability distributions: check calibration and separation between classes; if overlap is high, consider calibration or richer models (BERT, embeddings).

## Figures (placeholders)

**Learning curves:** learning\_curve.png



**Precision-Recall curve:** precision\_recall.png

A graph of precision-recall curve

AI-generated content may be incorrect.

**Confusion matrix:** confusion\_matrix.png

A blue squares with white text

AI-generated content may be incorrect.

**ROC curve:** roc\_curve.png

A graph with a line and a line

AI-generated content may be incorrect.

**Prediction distributions:**

A green and red graph

AI-generated content may be incorrect.

# 4.0 Conclusion

This project provides a reproducible pipeline for fake news detection using TF-IDF features and a Logistic Regression baseline. The artifacts tfidf\_vectorizer.pkl and fake\_news\_detector.pkl enable deployment. Key next steps include experimenting with contextual embeddings (BERT), model calibration, LIME/SHAP explainability, and production monitoring for drift.

# References

* Dataset: Fake.csv and True.csv (combine into new\_dataset.csv). Replace with exact dataset URL if using Kaggle or other source.
* Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research.
* Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python (NLTK).
* Porter, M. F. (1980). An algorithm for suffix stripping. Program.
* Joblib documentation / scikit-learn documentation / matplotlib / seaborn documentation.