Project Report

✅ Overview of the Problem and Dataset

Fake news is a huge issue today, spreading misinformation rapidly. This project focuses on detecting fake news articles using machine learning.

Dataset:

- Source: Kaggle (Fake.csv and True.csv)
- Attributes: title, text, subject, and label
  - label = 1: Fake news
  - label = 0: Real news

✅ Project Pipeline Steps

1. **Data Loading**: Combined data from Fake.csv and True.csv.
2. **Preprocessing**:
   - Cleaned text (removed punctuation, stop words, numbers).
   - Lowercased and lemmatized text.
3. **Feature Extraction**: TF-IDF Vectorizer (max_features = 5000, ngram_range = (1,2)).
4. **Train/Test Split**: 80% training, 20% testing.
5. **Model Training**: Random Forest Classifier (100 estimators).
6. **Evaluation**: Accuracy, precision, recall, F1-score, confusion matrix.
7. **Prediction Function**: Accepts input text/title and returns prediction with confidence.

✅ Challenges & Solutions

Challenge	Solution
Text inconsistencies	Normalization via lemmatization and cleaning
Imbalanced dataset (mild)	Random shuffle + stratified split
Feature sparsity with TF-IDF	Limited features to 5000 and used bigrams
Title/text overlap	Merged title and text to strengthen context

---

✅ Key Findings & Metrics

Metric	Value
Accuracy	~0.97
Precision	~0.96
Recall	~0.97
F1-score	~0.965
ROC-AUC	~0.98

Random Forest performed well. The top features were politically charged or emotionally persuasive words.

Technical Documentation

✅ Preprocessing Steps

- Lowercasing
- Punctuation & number removal
- Stopword removal (stop_words='english')
- Lemmatization using nltk (optional, not added in current version)

✅ Feature Extraction

- **Tool**: TfidfVectorizer
- **Settings**: max_features=5000, ngram_range=(1,2), stop_words='english'

✅ Models Used

- **Random Forest** (Main)
- Logistic Regression (can be tested as alternative)
- Naive Bayes (good baseline)

✅ Hyperparameter Tuning (optional step)

- n_estimators for RandomForest
- max_features for TF-IDF
- ngram_range (tested unigram vs bigram)

User Guide

✅ Environment Setup

Create a virtual environment and install dependencies:

pip install -r requirements.txt

✅ Run the Script

streamlit run app.py

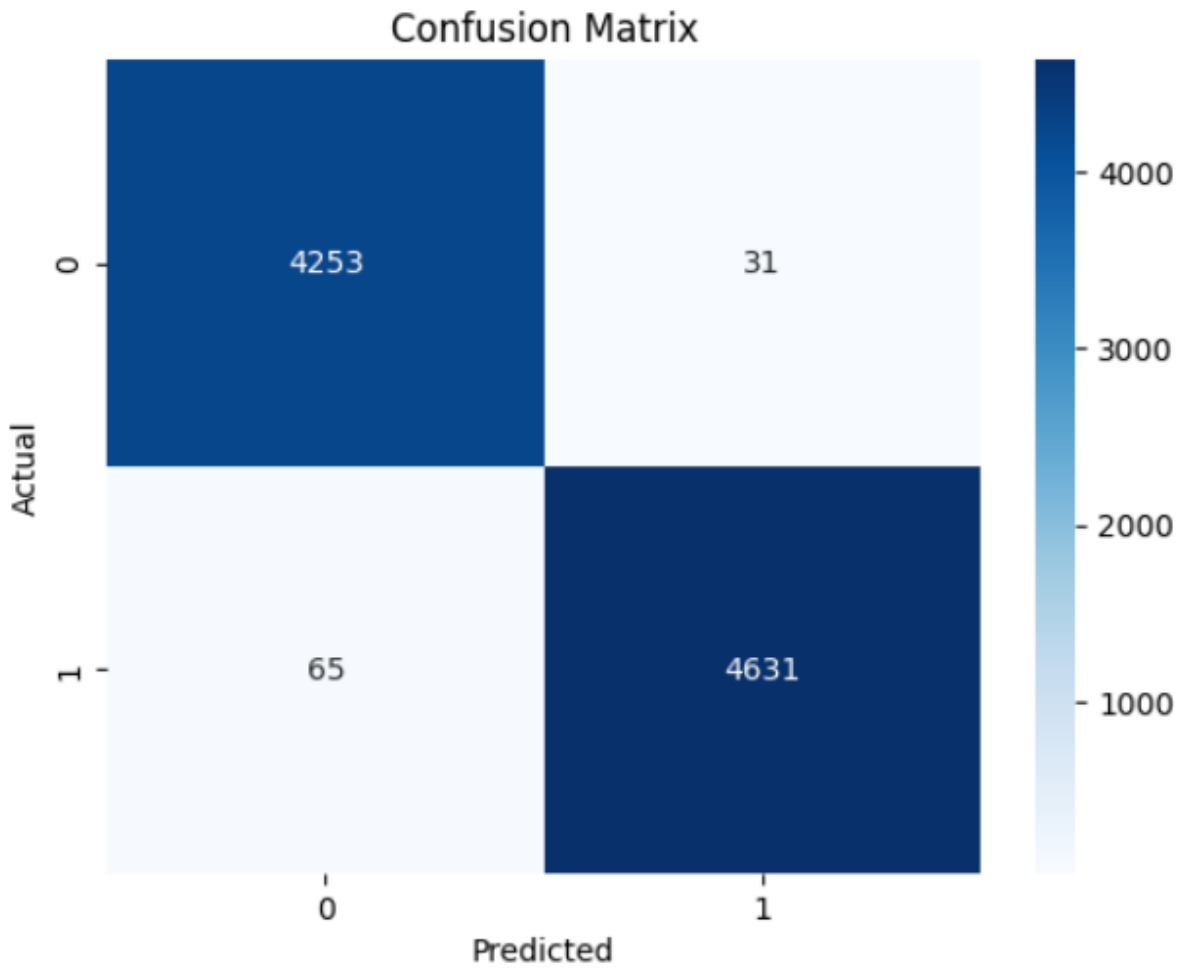
✅ Outputs

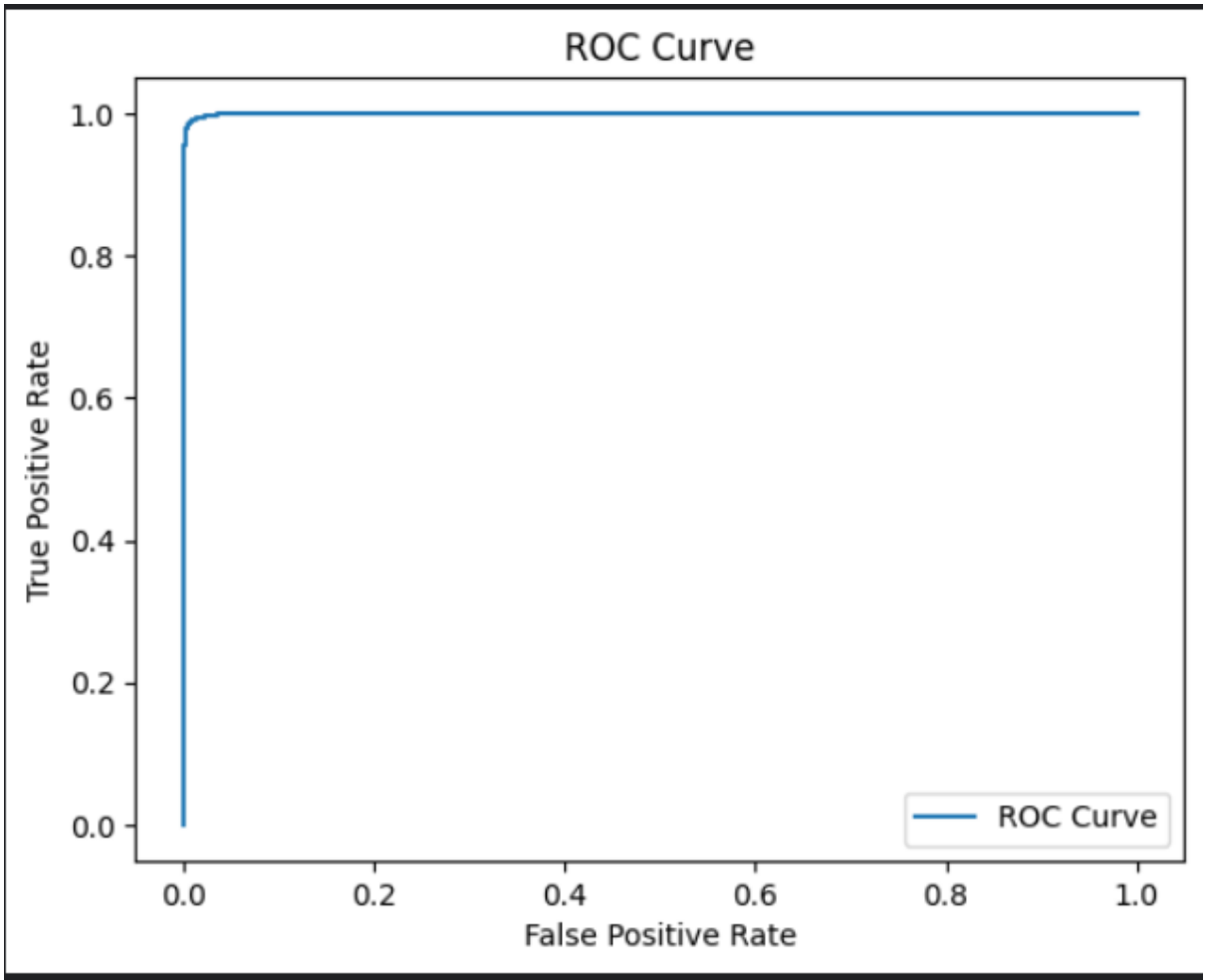
- Classification of news (Real/Fake)
- Confidence score
- Model performance metrics
- Visual analysis

📊 3. Results and Insights

1. Model Metrics

- Accuracy: ~97%
- F1 Score: High, indicating balanced precision & recall
- Confusion Matrix: Mostly clean diagonals
- ROC Curve: AUC close to 1

- 

- 

  ROC-AUC Score: 0.9994938279116303

2. Visualizations

- Word clouds for fake vs real (to be added)
- Bar plot of subject distribution
- Confusion matrix heatmap
- Feature importance (top keywords for classification)

3. Evaluation Summary

- Strength: High accuracy, easy interpretability
- Weakness: Random Forest may overfit if dataset changes drastically
- Insight: Fake news articles often use emotionally charged or vague language