Project Report

Fake News Detection using MLOps Pipeline

Team Members:

1. **Anmol (PNR: 250250125012) (Team Lead)**
2. **Dushyant Baghel (PNR: 250250125018)**
3. **Vikrant Tyagi (PNR: 250250125070)**
4. **Aditya Vats (PNR: 250250125007)**
5. **Aman Panchal (PNR: 250250125008)**

---

1. Project Overview

This project focuses on detecting fake news using machine learning and deploying the entire system using an MLOps pipeline. It automates the data cleaning, model training, evaluation, Dockerization, deployment, and quality checks using GitHub Actions. A web interface is also created for end-users to interact with the model.

---

2. Objectives

- Build a machine learning model to classify news as fake or real.
- Implement an end-to-end MLOps pipeline using GitHub Actions.
- Ensure reproducibility, security, code quality, and containerized deployment.
- Provide a simple and interactive web application for text and CSV-based predictions.

---

3. Technologies and Tools Used

- **Programming Language**: Python
- **Modeling**: Logistic Regression, Decision Tree, LSTM (Ensemble)
- **Web Framework**: Flask
- **CI/CD**: GitHub Actions
- **Version Control**: Git, GitHub
- **Containerization**: Docker
- **Security Scanning**: Trivy
- **Code Quality**: CodeQL
- **Visualization**: Matplotlib, Seaborn, WordCloud
- **Notebook Generation**: nbformat, nbconvert

---

4. Data Pipeline

- Validates dataset structure against expected columns.
- Cleans textual content, fills missing values, removes malformed rows.
- Calculates word and character counts.
- Ensures date formatting and proper encoding.

Expected columns:

['Index', 'News Headline', 'Complete News', 'Fake News(Yes/No)', 'word_count', 'character_count', 'Publish Dates', 'Source']

---

5. Model Training Pipeline

- Uses TF-IDF vectorization and Keras tokenization.
- Trains Logistic Regression, Decision Tree, and LSTM models.
- Ensemble voting strategy used for final prediction.
- Evaluates using accuracy, precision, recall, F1-score.
- Saves models, vectorizer, and tokenizer using `joblib`.

---

6. GitHub Actions CI/CD Pipeline

Trigger: On push to `master` and on manual dispatch.

Pipeline Steps:

1. **Checkout Repository**
2. **Setup Python 3.12.7**
3. **Install dependencies** from `instructions.txt`
4. **Extract Dataset ZIP** and place in `dataset/`
5. **Clean Dataset** using `clean_dataset.py`
6. **Clone project repo** for access
7. **Train Model** using `train_model.py`
8. **Update Model File** in the app directory
9. **Create New Git Branch**, Commit and Push
10. **Create Pull Request** with GitHub CLI

11. **Build Docker Image** tagged by run number
12. **Push Docker Image to DockerHub**
13. **Run CodeQL Analysis** for code quality
14. **Run Trivy Security Scan** on Docker Image

---

7. Dockerization

Dockerfile configured to:

- Use Python 3.9-slim
- Set working directory to `/app`
- Install dependencies
- Expose port `5000`
- Run `fake_news_app/app.py`

Docker Image is built and pushed in CI pipeline:

```
docker build -t anmolmishra334/fake_news_detection:<run_number> .
docker push anmolmishra334/fake_news_detection:<run_number>
```

---

8. Web Application Features

Framework: Flask

Functionalities:

- **Text Input**: Enter a news snippet and get prediction.
- **CSV Upload**: Upload CSV file, add prediction column.

- **Analytics Notebook**:
  - Auto-generated Jupyter notebook with:
    - Fake vs Real News Count
    - Distribution of Word Counts
    - Distribution of Character Counts
    - Word Cloud of News Content
- **Downloads**: Modified CSV and notebook available for download.

---

9. Security and Quality Analysis

- **CodeQL**: Scans for code-level vulnerabilities and issues.
- **Trivy**: Docker container vulnerability scanner.

---

10. Project Folder Structure

```
.
├── .github/workflows/ci_pipeline.yml
├── dataset/
│   ├── fake_news_final.csv
│   └── new_model_folder/model.joblib
├── scripts/
│   ├── clean_dataset.py
│   └── train_model.py
├── fake_news_app/
│   ├── app.py
│   └── templates/index.html
├── uploads/
├── Dockerfile
├── instructions.txt
└── README.md
```

11. Results

- Ensemble model outperforms individual models in most metrics.
- Achieved significant automation with zero manual intervention after push.
- User interface successfully integrated with backend logic.

12. Future Enhancements

- Implement authentication and access control.
- Schedule periodic retraining with updated data.
- Deploy on cloud (AWS EC2/GCP App Engine).
- Integrate logging/monitoring (Grafana, Prometheus).
- Add feature for confidence score display.

13. Conclusion

This project demonstrates the integration of machine learning with MLOps best practices. It ensures a reproducible, scalable, and secure workflow, along with an accessible interface for non-technical users. By automating training, evaluation, and deployment, this project provides a robust blueprint for future ML-based production systems.

GitHub Repository: anmolmishra334/Fake-News-Detection-NLP

DockerHub: anmolmishra334/fake_news_detection