Project Report: Fake News Detection System

Using Support Vector Machine and Custom TF-IDF Implementation

Nohail Faraz (2024519), Abdul Rehman (2024026), Haseeb Syed (2024220)

1. Abstract

The proliferation of fake news on social media and digital platforms poses a significant threat to public discourse. This project implements a machine learning-based solution to automate the detection of fake news. Using a dataset of labeled news articles, the system employs Natural Language Processing (NLP) techniques for text cleaning and feature extraction. Notably, a custom Term Frequency-Inverse Document Frequency (TF-IDF) algorithm was implemented from scratch to convert text into numerical vectors. A Linear Support Vector Machine (SVM) classifier was trained to distinguish between "Real" and "Fake" news, achieving high accuracy and providing a robust tool for content verification.

2. Introduction

2.1 Problem Statement

In the digital age, information spreads rapidly without verification. Misinformation and disinformation (fake news) can manipulate public opinion, incite violence, and erode trust in media. Manual fact-checking is labor-intensive and cannot keep pace with the volume of content generated daily. There is an urgent need for automated systems to flag potentially deceptive content.

2.2 Solution Overview

This project develops a Fake News Detector that classifies news articles into two categories: - **0 (Fake):** Deceptive or fabricated news. - **1 (Real):** Authentic, verified news.

The solution utilizes statistical text analysis and supervised machine learning to identify linguistic patterns associated with fake vs. real news.

2.3 Use Cases

- **Social Media Filtering:** Automatically flagging suspicious posts for review.
- **News Aggregators:** Ensuring only verified sources are displayed to users.
- **Journalism Tools:** Assisting reporters in preliminary fact-checking.
- **Education:** Helping users understand the linguistic differences between credible and non-credible sources.

3. System Architecture & Methodology

The project follows a standard Machine Learning pipeline: 1. **Data Acquisition:** Downloading the dataset programmatically. 2. **Exploratory Data Analysis (EDA):** Understanding data distribution and trends. 3. **Preprocessing:** Cleaning raw text to remove noise. 4. **Feature Engineering:** Converting text to numerical vectors (Custom TF-IDF). 5. **Model Training:** Training a Linear SVM. 6. **Evaluation:** Assessing performance using metrics and confusion matrices.

3.1 Libraries and Tools

Library	Purpose
Pandas	Data manipulation, dataframe management, and time-series analysis.
NumPy	Numerical computations and matrix operations for the custom TF-IDF.
Matplotlib	Data visualization (bar charts, line graphs, confusion matrices).
Scikit-Learn	Model building (LinearSVC), splitting data, and evaluation metrics.
Opendatasets	Downloading the dataset directly from Kaggle.
Collections	Used Counter to analyze word frequency distributions.

4. Data Analysis and Preprocessing

4.1 Dataset Description

- **Source:** Kaggle (Mahdi Mashayekhi - Fake News Detection Dataset) • **Key Features:**
- `text` : The main body of the news article.
- `label` : The target variable (`fake` or `real`).
- `date` : Publication date (used for time-series analysis).
- `category` : The genre of the news (e.g., politics, sports).

4.2 Exploratory Data Analysis (EDA)

- **Temporal Trends:** Articles grouped by date to visualize news volume over time.
- **Label Distribution:** Ratio of Fake vs. Real news to check for class imbalance.
- **Word Counts:** Histograms comparing length of fake vs. real articles.
- **Top Words:** Most frequent words visualized; real news has formal terminology, fake news shows different linguistic markers.

4.3 Text Preprocessing

Custom `clean_text` function: 1. Lowercasing. 2. Filtering non-alphabetic characters. 3. Stopword removal. 4. Removing words with fewer than 3 characters.

5. Implementation Details

5.1 Feature Engineering: Custom TF-IDF

- **Vocabulary Building:** Unique index assigned to every word.
- **Term Frequency (TF):** $$TF(t, d) = \frac{\text{count of word } t \text{ in doc } d}{\text{total words in doc } d}$$
- **Inverse Document Frequency (IDF):** $$IDF(t) = \log(\frac{N + 1}{DF(t) + 1}) + 1$$
- **Vectorization:** $$TF\text{-}IDF = TF \times IDF$$ **5.2 Model**

Selection: Support Vector Machine (SVM)

- **Model:** LinearSVC.
- **Reasoning:** Efficient for high-dimensional text data, computationally efficient, less prone to overfitting on sparse datasets.

5.3 Training

- **Split:** 80% training, 20% testing.
- **Training:** Build vocabulary, compute TF-IDF, train SVM.
- **Testing:** Evaluate model on unseen articles.

6. Results and Evaluation

6.1 Performance Metrics

- **Accuracy:** Percentage of correctly classified articles.
- **Classification Report:** Precision, Recall, F1-Score for both classes.

6.2 Confusion Matrix

- Normalized confusion matrix plotted.
- High diagonal values indicate correct predictions.
- Low off-diagonal values indicate minimal misclassification.

6.3 Observations

- Custom TF-IDF captured importance of specific words.
- Linear SVM effectively separated the two classes.
- Fake news differs in sentence length and vocabulary from real news, learned by the model.

7. Conclusion and Future Scope

7.1 Conclusion

The project demonstrates effective machine learning application in combating misinformation. Custom TFIDF highlights NLP foundations. Linear SVM provides robust text classification.

7.2 Limitations

- Context is ignored (Bag-of-Words).
- Model trained on historical data; may not detect emerging fake news without retraining.

7.3 Future Scope

- Incorporate N-Grams (Bigrams/Trigrams).
- Use Deep Learning (LSTM, BERT) for context.
- Real-time deployment via web API (Flask/Streamlit).