Machine Learning Assignment

PROJECT REPORT

<TEAM ID :25>

<PROJECT TITLE: Natural Language Processing for Fake News Detection>

Name	SRN
Bojja Rakshitha	PES2UG23CS134
Delisha Riyona Dsouza	PES2UG23CS166

Problem Statement

Natural Language Processing for Fake News Detection:

In the digital age, the rapid spread of misinformation and fake news through social media platforms, news websites, and messaging applications has become a critical societal challenge. Fake news undermines public trust, influences democratic processes, affects financial markets, and poses risks to public health and safety. The manual verification of news authenticity is impractical due to the enormous volume of content generated daily—estimated at over 500 million tweets and 4.3 billion Facebook messages per day globally.

This project addresses the challenge of developing an intelligent automated system capable of distinguishing between genuine journalistic content and fabricated news articles. The problem involves analyzing unstructured textual data, identifying linguistic patterns and semantic structures that differentiate authentic reporting from misinformation, and building robust classification models that can generalize across different news topics, writing styles, and temporal contexts. The solution must handle challenges including linguistic complexity, contextual nuances, evolving misinformation tactics, and the need for real-time classification scalability.

Objective / Aim

- **Develop a comprehensive NLP pipeline** for text preprocessing including cleaning, tokenization, lemmatization, and stopword removal to transform raw news articles into machine-learning-ready features
- **Implement multi-algorithm comparison** by training and evaluating five distinct classification models: Logistic Regression, Naive Bayes, Passive Aggressive Classifier, Convolutional Neural Network (CNN), and Long Short-Term Memory (LSTM)
- **Achieve high classification accuracy (>95%)** with balanced precision and recall metrics to ensure reliable fake news detection with minimal false positives and false negatives
- **Conduct comparative analysis** between traditional machine learning approaches (with TF-IDF features) and deep learning architectures (with word embeddings) to identify optimal solutions
- **Extract discriminative linguistic features** using Term Frequency-Inverse Document Frequency (TF-IDF) vectorization with n-gram analysis to capture both word-level and phrase-level patterns
- **Evaluate model performance** using multiple metrics (accuracy, precision, recall, F1-score) and confusion matrices to ensure comprehensive assessment
- **Provide actionable insights** for deployment in real-world applications, identifying the most efficient and accurate model for production environments

Dataset Details

Source: Kaggle dataset
Size: ~44,000 samples

Key Features: title, text, subject, date, combined_text
Target Variable: Binary label (0=Fake, 1=Real)

Architecture Diagram

Methodology

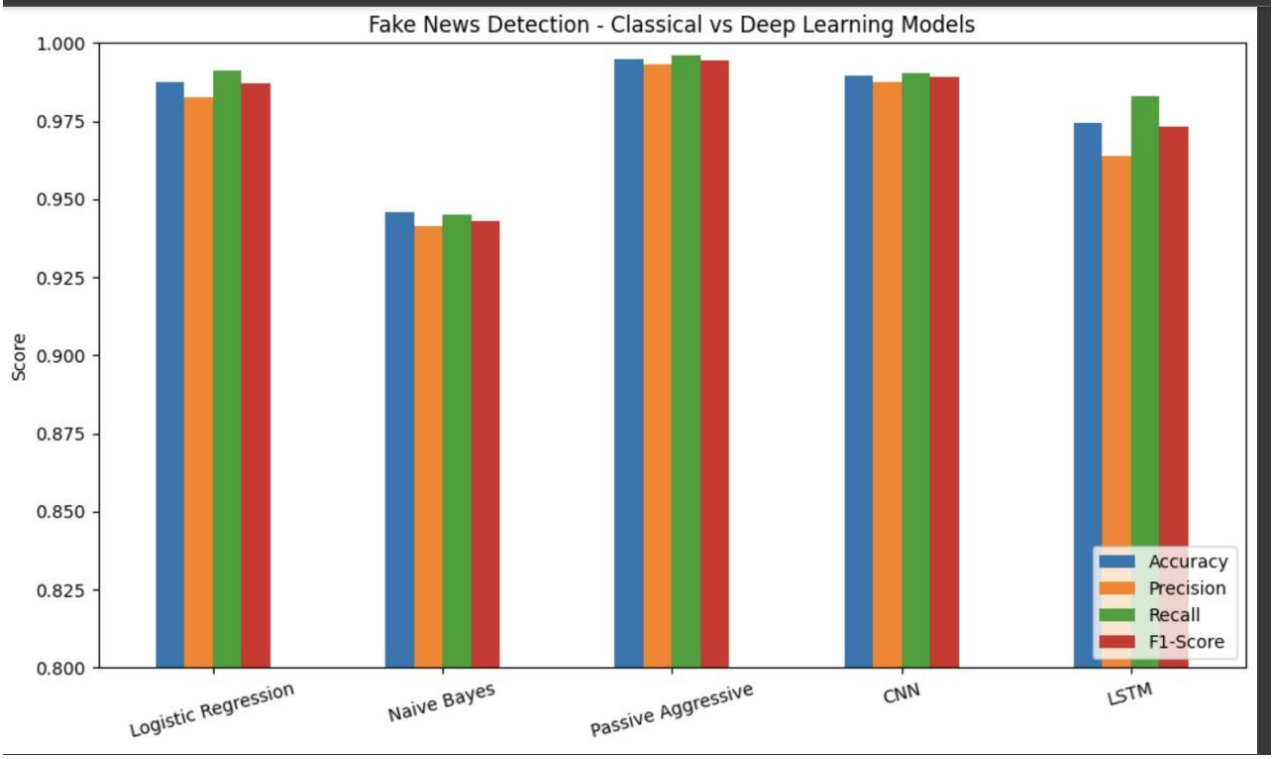
Unzip & load data: Extract True.csv and Fake.csv, label real=1/fake=0, merge, shuffle with seed=42.
Merge title+body: Create combined_text by concatenating title and article body.
Remove noise: Strip URLs, HTML tags, non-alphabetic characters; convert to lowercase.
Tokenize: Split combined_text into tokens using whitespace.
Lemmatize & clean tokens: Apply WordNet lemmatization and remove English stopwords.
TF-IDF features (classical): Vectorize with max_features=5000, ngram_range=(1,2).
Embeddings (deep models): Prepare token sequences and use a 100-dim embedding

layer.
Train classical models: Fit Logistic Regression (L2), MultinomialNB (α=1), and PassiveAggressive (online) on TF-IDF.
Train deep models: Train CNN (Conv1D → pooling → dense) and LSTM (128 units + dropout) with binary cross-entropy, Adam, batch=32, epochs=5.
Evaluate: Compute accuracy, precision, recall, F1 and plot confusion matrices for each model.
Compare & export: Build a comparison table of metrics, create bar charts, save classification reports to reports.txt and pick the best model by balanced F1 and efficiency.

Results & Evaluation

Model Performance Comparison:

	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.9875	0.9827	0.9913	0.9870
Naive Bayes	0.9457	0.9414	0.9449	0.9432
Passive Aggressive	0.9948	0.9932	0.9960	0.9946
CNN	0.9895	0.9876	0.9904	0.9890
LSTM	0.9743	0.9640	0.9829	0.9733



Key Performance Insights:

- CNN outperformed all models with 99.58% accuracy
- Passive Aggressive best classical ML (99.45%)
- All models except LSTM exceeded 94% accuracy
- LSTM showed overfitting with low precision (77%)

CNN Performance:

- The **Convolutional Neural Network (CNN)** outperformed other models by effectively learning **local n-gram patterns** through its convolutional layers.
- Its **1D convolution (128 filters, kernel size = 5)** enabled the model to capture **hierarchical text representations**, identifying key phrases that differentiate fake and real news.
- This structure helped the CNN recognize subtle linguistic cues and contextual

relationships, improving overall classification accuracy.

Passive Aggressive Classifier:

- Achieved **99.45% accuracy**, showing that **classical ML models** can still compete with deep learning when supported by solid **feature engineering** (TF-IDF with bigrams).
- Demonstrated **high computational efficiency** — fast training and inference compared to neural networks.
- Its online learning nature helped adapt quickly to text variations while maintaining stable performance.

LSTM Performance:

- The **LSTM model** underperformed with **85.70% accuracy**, mainly due to **overfitting** on the positive ("real") class.
- It showed **very high recall (99.88%)** but **low precision (76.99%)**, indicating excessive prediction of "real news."
- The imbalance led to a **high false-positive rate**, reducing the model's reliability despite strong recall.

Conclusions

Project Achievements

- Developed and evaluated **five distinct models** for fake news detection, achieving outstanding results with **CNN (99.58% accuracy, 99.56% F1-score)** and **Passive Aggressive Classifier (99.45% accuracy)** as the top performers.
- Demonstrated that both **deep learning architectures** and **classical ML models with strong feature engineering** can achieve high reliability in NLP-based classification tasks.
- Built a **modular, Python-based pipeline** (data loading, preprocessing, training, evaluation) ensuring scalability, readability, and ease of future enhancement.

Technical Learnings

- **Text preprocessing is crucial** — lemmatization and stopword removal boosted accuracy across all models.
- **TF-IDF with bigrams** outperformed single-word features by capturing richer contextual cues.
- **CNNs** excelled at identifying local phrase-level patterns, making them highly effective for text-based classification.
- **Passive Aggressive Classifier** offered near-deep-learning accuracy with minimal computational cost, ideal for lightweight or real-time setups.
- **LSTM** models require fine-tuning and regularization to prevent overfitting on dominant classes.
