***Phase-3***

***Exposing the truth with advanced fake news detection powered by natural language processing***

***Student Name: S. Sembaruthi***

***Register Number: 620123106104***

***Institution: AVS Engineering College***

***Department: Electronics and Communication Engineering***

***Date of Submission: 13.05.2025***

***Github Repository Link:https://github.com/Sembaruthi-S/Nan -Muthalvan- Project-. git***

# *1.Problem Statement*

**The digital era has made the spread of misinformation easy and rapid. Fake news, especially on socialmedia, can manipulate public opinion, incite unrest, and cause harm. A reliable and automated solution isneeded to detect such news articles in real-time using modern AI techniques**

# *2.Abstract*

**This project aims to detect fake news using Natural Language Processing (NLP) and machine learning.By analyzing textual patterns in news content, the system classifies whether a news article is real or fake.The project includes preprocessing, feature extraction, model training, evaluation, and deployment of a web-based interface for user interaction.**

# *3.System Requirements*

**Hardware:**

**- RAM: 8 GB or more**

**- CPU: Intel i5/i7 or AMD equivalent**

**- GPU: Recommended for deep learning**

**Software:**

**- OS: Windows/Linux/Mac**

**- Python 3.8+**

**- Libraries: Pandas, NumPy, Matplotlib, NLTK, Scikit-learn, TensorFlow, Flask**

**- Tools: Jupyter Notebook, VS Code, Postman (for testing API**

# *4.Objectives*

# *Collect and clean fake and real news datasets.*

# *- Apply NLP techniques to preprocess and analyze the data.*

# *- Build classification models to distinguish fake from real news.*

# *- Evaluate model performance with robust metrics.*

# *- Deploy the model as a web-based prediction tool.*

# *5.Flowchart of Project Workflow*

**START**

**->Data Collection**

**-> Data Preprocessing**

**-> EDA**

**-> Feature Engineering**

**-> Model Building**

**-> Evaluation**

**->Deployment**

**->END**

# *6.Dataset Description*

***Source: Kaggle - Fake and Real News Dataset***

***- Features: title, text, subject, date, label***

***- Size: ~40,000 articles***

***- Labels: FAKE, REAL***

# *7.Data Preprocessing*

# *Convert text to lowercase*

***- Remove punctuation, numbers, stopwords***

***- Tokenization and Lemmatization (using NLTK or spaCy)***

***- Handle missing/null values***

***- Combine title + text as input features***

# *8.Exploratory Data Analysis (EDA)*

# *Class balance check (Fake vs. Real)*

***- Word cloud visualization for both classes***

***- Top word frequency count***

***- Article length distribution***

***- Sentiment analysis (optional)***

# *9.Feature Engineering*

***TF-IDF Vectorization***

***- CountVectorizer***

***- N-gram analysis (bi-gram, tri-gram)***

***- Word embeddings (optional: GloVe, Word2Vec)***

***- Dimensionality reduction (PCA/TruncatedSVD if needed)***

# *10.Model Building*

***Models used:***

***- Logistic Regression***

***- Naive Bayes***

***- Random Forest***

***- XGBoost***

***- LSTM / BERT (optional for advanced version)***

***Hyperparameter tuning via:***

***- GridSearchCV***

***- Cross-validation (k-fold)***

# *11.Model Evaluation*

**Metrics:**

**- Accuracy**

**- Precision, Recall, F1-Score**

**- Confusion Matrix**

**- ROC-AUC curve**

**Use a validation split (e.g., 80-20 or 70-30) and cross-validation.**

# *12.Deployment*

***Flask-based web app***

***- User inputs news text via a form***

***- Backend processes the input and returns a prediction***

***- Hosted on platforms like:***

***- Heroku***

***- Render***

***- Streamlit (alternative GUI)***

# *13.Source code*

**Install Required Libraries**

**pip install pandas scikit-learn nltk**

**Import Libraries**

**import pandas as pd**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.feature\_extraction.text import TfidfVectorizer**

**from sklearn.linear\_model import PassiveAggressiveClassifier**

**from sklearn.metrics import accuracy\_score, confusion\_matrix**

**import nltk**

**from nltk.corpus import stopwords**

**from nltk.stem import WordNetLemmatizer**

**import re**

**Load and Preprocess Data**

**nltk.download('stopwords')**

**nltk.download('wordnet')**

**df = pd.read\_csv('fake\_or\_real\_news.csv') # Assumes 'text' and 'label' columns**

**lemmatizer = WordNetLemmatizer()**

**stop\_words = set(stopwords.words('english'))**

**def preprocess(text):**

**text = re.sub(r'\W', ' ', text.lower())**

**words = text.split()**

**words = [lemmatizer.lemmatize(w) for w in words if w not**

**stop\_words]**

**return ' '.join(words)**

**df['text'] = df['text'].apply(preprocess)**

**Feature Extraction**

**X = df['text']**

**y = df['label']**

**tfidf = TfidfVectorizer(max\_df=0.7)**

**X\_tfidf = tfidf.fit\_transform(X)**

**Train Model**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_tfidf, y,**

**test\_size=0.2,**

**random\_state=42)**

**model = PassiveAggressiveClassifier(max\_iter=50)**

**model.fit(X\_train, y\_train)**

# *14.Future scope*

# Real-time fake news detection using Twitter/Facebook APIs

# - Transformer-based models (e.g., BERT, RoBERTa)

# - Multilingual fake news detection

# - Image/video misinformation analysis

# - Browser extension integration

# *15.Team Members and Roles*