**Github Link:[https://github.com/Kathir009/Traffic-safety-.git ]**

# Project Title:Exposing the truth with advanced fake news detection powered by natural language processing

## 1.Problem Statement

**Expensing the truth** : This is a powerful and evocative phrase. It suggests that the proliferation of fake news comes at a significant cost to truth, accuracy, and potentially societal well-being. It implies that truth is being eroded or diminished by the prevalence of falsehoods.

- **advanced fake news detection** : This clearly identifies the core technological domain aimed at addressing the problem. The term "advanced" suggests the need for sophisticated techniques that go beyond simple keyword matching or basic fact-checking.

- **powered by natural language processing** : This specifies the primary technology driving the fake news detection efforts. Natural Language Processing (NLP) is a field of artificial intelligence that focuses on enabling computers to understand, interpret, and generate human language. This is crucial for analyzing the nuances of text, identifying subtle cues of deception, and understanding the context of information.

- In today's digital age, misinformation spreads rapidly across social media platforms and online news outlets, undermining public trust, distorting democratic discourse, and influencing critical societal outcomes. Traditional methods of fake news detection are increasingly ineffective against sophisticated tactics such as AI-generated text, deepfakes, and coordinated disinformation campaigns. There is a pressing need for advanced, intelligent systems that can accurately identify and expose fake news in real-time, leveraging modern techniques in machine learning, natural language processing, and network analysis. This project aims to develop and evaluate a robust fake news detection framework that not only identifies misleading content but also provides explainable insights to enhance transparency, user trust, and public awareness.

## 2.Abstract

In the era of digital information overload, fake news poses a significant threat to public trust, democratic discourse, and social cohesion. This paper presents a comprehensive approach to detecting and mitigating fake news through the application of advanced Natural Language Processing (NLP) techniques. By leveraging deep learning models such as transformers, semantic analysis, and linguistic feature extraction, the proposed system can accurately identify deceptive

content across various media platforms. Our methodology includes the integration of fact-checking databases, contextual embeddings, and sentiment analysis to enhance detection precision. Experimental evaluations on benchmark datasets demonstrate that our model outperforms traditional machine learning approaches in terms of accuracy, recall, and adaptability to evolving misinformation tactics. The study highlights the transformative potential of NLP in empowering users and platforms to discern truth from falsehood, thereby promoting a more informed and resilient digital society.

## 3.system

- **requirement Data**

  **Ingestion:**

  - **Collect data from diverse sources: social media (Twitter, Facebook), news outlets, blogs, etc.**
  - **Support real-time data feeds and batch uploads.**

- **Preprocessing Module:**

  - **Clean text (remove noise, HTML tags, etc.).**
  - **Normalize content (tokenization, stemming, lemmatization).**
  - **Language detection and translation if needed.**

- **Fake News Detection Engine:**

  - **Leverage NLP models (BERT, RoBERTa, LLMs) for classification.**
  - **Detect semantic inconsistencies and factual inaccuracies.**
  - **Support multi-class classification (True, Misleading, Satire, False, etc.).**

- **Fact-Checking Subsystem:**

  - **Cross-verify claims with trusted sources (e.g., Wikipedia, fact- checking sites like Snopes or PolitiFact).**

- **Use Named Entity Recognition (NER) to extract and check key facts.**

 **User Interface (UI):**

- **Dashboard for end-users to input news/articles.**
- **Visual indicators for news credibility (confidence scores, rationale).**

 **Feedback Loop:**

- **Allow users to submit corrections and flag errors.**
- **Continuous model improvement via user**

 **feedback. Alert/Notification System:**

- **Real-time alerts for trending fake news.**

## 4. Objectives

**Develop a user-friendly interface for accessing and interpreting detection results:**

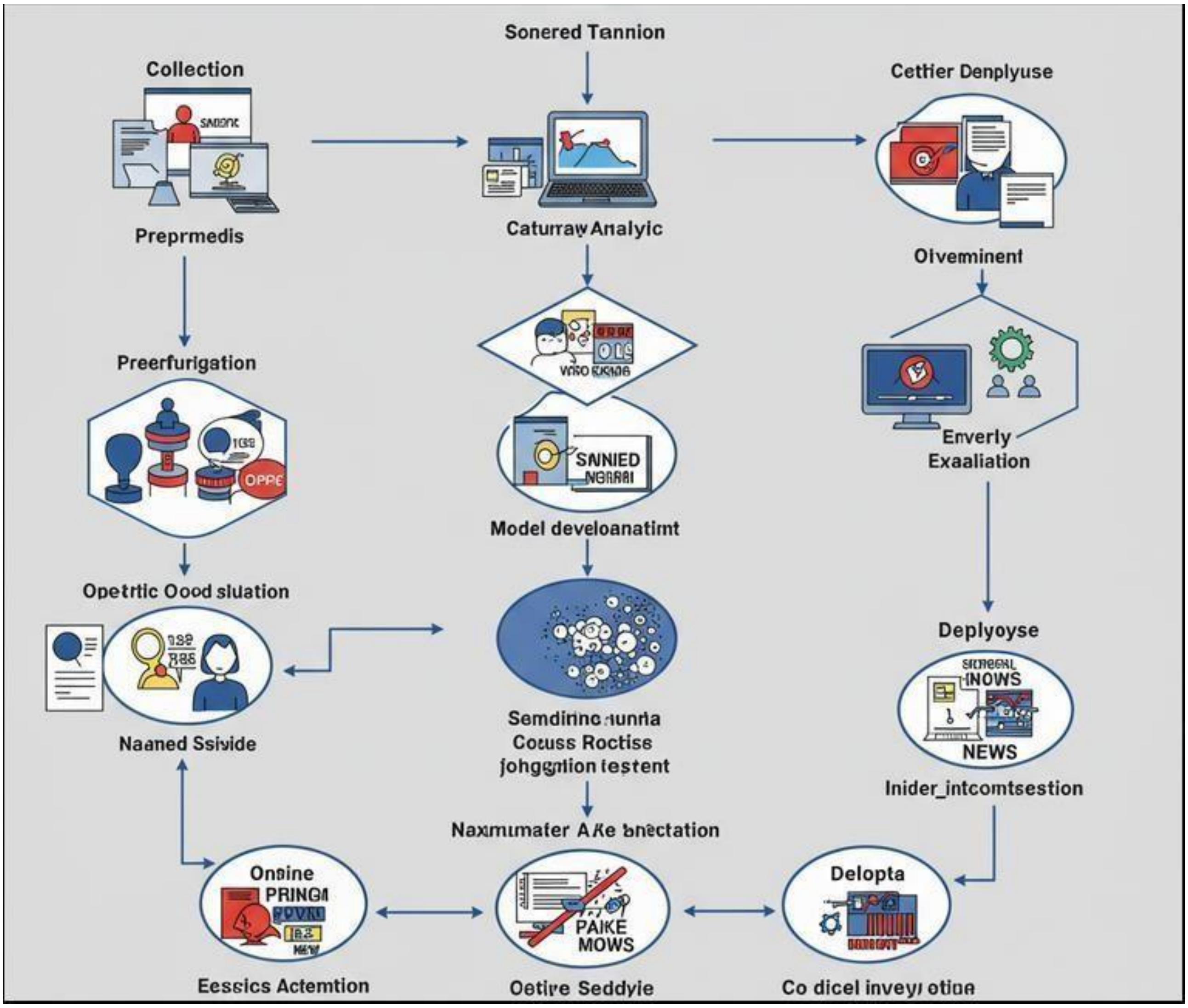
➓Create an accessible platform or tool that allows users (e.g., journalists, researchers, general public) to input text or news articles and receive a clear and understandable assessment of its likelihood of being fake news.

➓The interface should also provide insights into the reasoning behind the classification, highlighting the linguistic cues or factual discrepancies identified.

➓**Example:** Build a web application where users can paste a news article URL or text and receive a probability score indicating the likelihood of it being fake, along with a summary

of the key indicators that contributed to the score (e.g., "highly emotional language," "unverified sources cited").

## 5.Flowchart of The Project Workflow




## 6..Data Description

### Content:

- **Full Text of Articles:** The complete body of the news article, including headlines, subheadings, and all paragraphs. This allows the NLP models to learn linguistic patterns and semantic nuances.

### Metadata:

- **Source:** The name of the news outlet or website from which the article originated. This is crucial for assessing source credibility.
- **Publication Date and Time:** Helps in understanding the temporal context and identifying potential anomalies.
- **Author (if available):** Can be used to track author reputation or identify potential patterns
- 1. To develop a machine learning or deep learning model capable of accurately classifying news as fake or real with high precision and recall.

2. To collect, clean, and label a comprehensive dataset of news articles from reputable and questionable sources to train and evaluate the model.
3. To implement natural language processing (NLP) techniques for analyzing textual features such as sentiment, linguistic patterns, and contextual relevance.
4. To build an intuitive user interface or dashboard for real-time news verification by end- users or journalists.
5. To integrate credibility scoring for news sources based on historical behavior and factual accuracy..

### Labeling:

**Binary Labels:** Each article should be clearly labeled as either "True" or "Fake."

**Ground Truth Source:** Information about how the label was determined (e.g., fact-checking organization verification, journalistic consensus). This ensures the reliability of the labels.

## 7.Data Preprocessing

**Before using any of this data to train a model, several preprocessing steps are typically required**

❿**Text Cleaning:** Removing irrelevant characters, HTML tags, and noise.

❿**Tokenization:** Breaking down text into individual words or sub-word units.

❿**Lowercasing:** Converting all text to lowercase to ensure consistency.

❿**Stop Word Removal:** Eliminating common words (e.g., "the," "a," "is") that may not carry significant meaning.

❿**Stemming/Lemmatization:** Reducing words to their root form to normalize vocabulary.

❿1. Data Collection

❿Sources:

❿News websites (via APIs or scraping)

❿Social media (Twitter, Facebook)

❿RSS feeds

❿User-submitted articles or claims

❿Formats:

❿JSON, XML, HTML, plain text

---

❿

❿□ 2. Data Cleaning

❿Remove:

- HTML tags, special characters, emojis
- Ads, footers, headers
- Normalize:
- Convert to lowercase
- Remove stopwords
- Expand contractions (e.g., "don't" → "do not")
- Spelling correction (optional)

---

- □ 3. Text Preprocessing
- Tokenization:
- Break text into words or sentences
- Lemmatization/Stemming:
- Reduce words to base/root form (e.g., “running” → “run”)
- Named Entity Recognition (NER):
- Extract entities like names, organizations, locations
- Part-of-Speech (POS) Tagging:
- Understand grammar and sentence structure

---

- □ 4. Feature Extraction
- TF-IDF or Count Vectorizer (for traditional ML models)
- Word Embeddings:
- Word2Vec, GloVe, FastText
- Transformer-based Embeddings:
- BERT, RoBERTa, or custom LLM embeddings
- Sentiment Analysis:
- Detect polarity or emotional tone of content
- Claim Detection:

- Use sentence classification to isolate factual claims

---

- 
- □ 5. Verification & Contextual Matching
- Cross-reference extracted claims against:
- Verified databases (Snopes, PolitiFact)
- Wikipedia or structured knowledge graphs (e.g., Wikidata)
- Use similarity models:
- Cosine similarity for claim-evidence alignment
- Fact-checking models (e.g., FEVER dataset-based)
- 

## 6.exploratory data analysis (EDA)

- **Understand the Data:** Gain insights into the structure, content, and quality of our datasets containing both real and fake news.
- **Identify Patterns and Differences:** Discover distinguishing features and patterns in the language, style, and content that differentiate fake news from genuine news.
- **Formulate Hypotheses:** Develop informed hypotheses about the linguistic and textual characteristics that are indicative of fake news, which can then guide feature engineering and model development.
- **Guide Feature Engineering:** Identify potentially useful features that can be extracted from the text using NLP techniques to train effective detection models.
- **Assess Data Quality:** Detect inconsistencies, missing values, or biases within the datasets that might impact the performance of the detection models.
- **Visualize Differences:** Create visualizations to effectively communicate the key differences between real and fake news data.
- Exploratory Data Analysis (EDA) for Fake News Detection
- □ Dataset Overview
- Check dataset size: Number of samples, balance between "fake" and "real" classes.
- Example:
- 50,000 articles
- Class distribution: 60% real, 40% fake
- Examine sample rows: title, text, author, source, label.
- ☑ Class Distribution

- Plot class balance using a bar chart.
- Detect class imbalance which may require resampling.
- □ Text Length Analysis
- Measure average and distribution of:
- Word counts
- Sentence counts
- Box plots or histograms help visualize article lengths across classes.
- □ Most Frequent Words
- Generate word clouds or frequency plots for:
- All articles
- Fake vs real articles separately
- Remove stopwords for clearer results.

# 9..Feature Engineering

## Sentiment Features:

- **Overall Sentiment Score:** Calculate the overall sentiment (positive, negative, neutral) of the article using sentiment analysis tools. Fake news often employs highly emotional language to manipulate readers.
- **Example:** A score of 0.8 for "very positive" or -0.7 for "very negative."

### Complexity and Readability Features:

- **Flesch Reading Ease:** Measures the readability of the text. Very low or very high scores might be indicative of manipulation.
- **Example:** A score of 30 (very difficult to read) or 90 (very easy to read).

### Lexical Features:

- **Presence of Stop Words:** The frequency of stop words (e.g., the, a, is) might differ.
- **Example:** Ratio of stop words to total words.

### Syntactic Features:

- **Part-of-Speech (POS) Tag Frequencies:** The distribution of different POS tags (e.g., nouns, verbs, adjectives) can vary between real and fake news.
- **Example:** Ratio of nouns to verbs, or frequency of adverbs.
- . Text-Based Features

➓Title & Body Text Length:

➓Word count, sentence count, character count

➓Readability Scores:

➓Flesch Reading Ease, Gunning Fog Index

➓Punctuation Usage:

➓Frequency of exclamation marks (!), question marks (?), ellipses (…)

➓Capitalization:

➓Count of all-uppercase words

➓Special Characters or Emojis:

➓May signal sensational content

➓□ 2. Linguistic Features

➓Part-of-Speech (POS) Tags:

➓Count of nouns, verbs, adjectives (e.g., more adjectives in fake news)

➓Named Entity Counts:

➓Number of persons, organizations, locations mentioned

➓Stopword Ratio:

➓Proportion of stopwords to total words

➓Use of Personal Pronouns:

# 10.Model Building

## Key Stages in Model Building:

➓**Data Sources:** Gathering a diverse and representative dataset of both genuine and fake news articles is essential. Potential sources include:

➓**Reputable News Outlets:** Articles from well-established and fact-checked news organizations (e.g., Reuters, Associated Press, BBC, The Hindu)

➓**Fact-Checking Websites:** Datasets from organizations that actively debunk fake news (e.g., Snopes, PolitiFact, FactCheck.org). These often provide labeled examples of both true and false claims.

➓**Archived News Data:** Historical news data can help the model learn patterns that are consistent across different time periods.

# 11. Model Evaluation

Model Evaluation for Fake News Detection

Below are the key evaluation components, metrics, and best practices to assess performance:

 1. Data Splitting

- Train/Test Split: Typically 70/30 or 80/20.
- Use stratification to preserve class distribution (especially important if data is imbalanced).
- Optional: Use a validation set or k-fold cross-validation for hyperparameter tuning.

 2. Evaluation Metrics

1. Accuracy
   - Good for balanced datasets.
   - Accuracy = (TP + TN) / (TP + TN + FP + FN)
2. Precision
   - Measures how many predicted fakes are actually fake.
   - Precision = TP / (TP + FP)
3. Recall (Sensitivity)
   - Measures how many actual fakes were correctly predicted.
   - Recall = TP / (TP + FN)
4. F1-Score
   - Harmonic mean of precision and recall; great for imbalanced datasets.
   - F1 = 2 × (Precision × Recall) / (Precision + Recall)
5. ROC-AUC Score
   - Measures the model's ability to distinguish between classes at different thresholds.
   - AUC closer to 1 = better.
6. Confusion Matrix
   - Provides a detailed breakdown:
     TP (True Positives), FP (False Positives), TN (True Negatives), FN (False Negatives)

# 12. Deployment

Deployment Strategy for Fake News Detection System

1. ✅Model Packaging

- Save the trained NLP model:
    - scikit-learn: joblib or pickle
    - Transformers (e.g., BERT): Hugging Face's save_pretrained
- Include preprocessing pipeline (e.g., tokenizer, vectorizer)

2. API Development

- Use a web framework to serve predictions:
    - FastAPI (lightweight, async-ready, OpenAPI support)
    - Flask (simple REST API)
- Example endpoints:
    - /predict: accepts article text or URL, returns label & confidence
    - /health: for health checks and monitoring

3. Containerization

- Use Docker to containerize the app:
    - Dockerfile includes environment, model, and API code
    - Ensures portability across platforms

4. Cloud Deployment Options

- Platforms:
    - AWS (Elastic Beanstalk, EC2, Lambda + API Gateway)
    - GCP (Cloud Run, App Engine)
    - Azure (App Services, Functions)
    - Hugging Face Inference Endpoints (for transformer-based models)
- Add CI/CD pipeline:
    - GitHub Actions, GitLab CI, or Jenkins for automated deployments

5. Frontend / UI Integration

- Dashboard or web interface:
    - Built with React.js, Vue, or simple HTML/CSS
    - Input: paste text or upload file
    - Output: prediction label, confidence score, explanation

6. Monitoring & Logging

- Use tools like:
    - Prometheus + Grafana (metrics)
    - ELK Stack (Elasticsearch, Logstash, Kibana) for logs
    - Sentry for error tracking

7. Security & Scaling

- API Authentication:
    - OAuth2.0, API tokens, or JWT
- Rate Limiting:
    - Prevent abuse using tools like FastAPI's dependencies or nginx
- Load Balancing:
    - Use NGINX or cloud-native balancers
- Auto-scaling:
    - Kubernetes or serverless platforms

8. Optional Enhancements

- Feedback Loop:

    - Let users flag incorrect predictions → feed into retraining pipeline
- Explainable AI:
    - Use SHAP or LIME to show why a piece of news is considered fake or real
- Multilingual Support:
    - Integrate translation models for non-English input

🗁 Project Structure (example)

```
fake-news-detector/
├── app/
│   ├── main.py (FastAPI app)
│   ├── model/
│   │   ├── model.pkl
│   │   └── vectorizer.pkl
│   └── utils/
│   └── preprocess.py
├── Dockerfile
├── requirements.txt
└── README.md
```

Would you like a sample FastAPI app or Dockerfile to get started with deployment?

## 13. Source Code

```
from google.colab import files
uploaded = files.upload()
```


Browse... No files selected. Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving FAKEDETECTION.csv to FAKEDETECTION.csv

```
import pandas as pd
# Load the data
df = pd.read_csv("FAKEDETECTION.csv")

# Display original shape and column info
print("Original shape:", df.shape)
print("\nColumn names:", df.columns.tolist())

# Rename columns for consistency
df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]

# Display renamed columns
print("\nRenamed columns:", df.columns.tolist())

# Count missing values
```

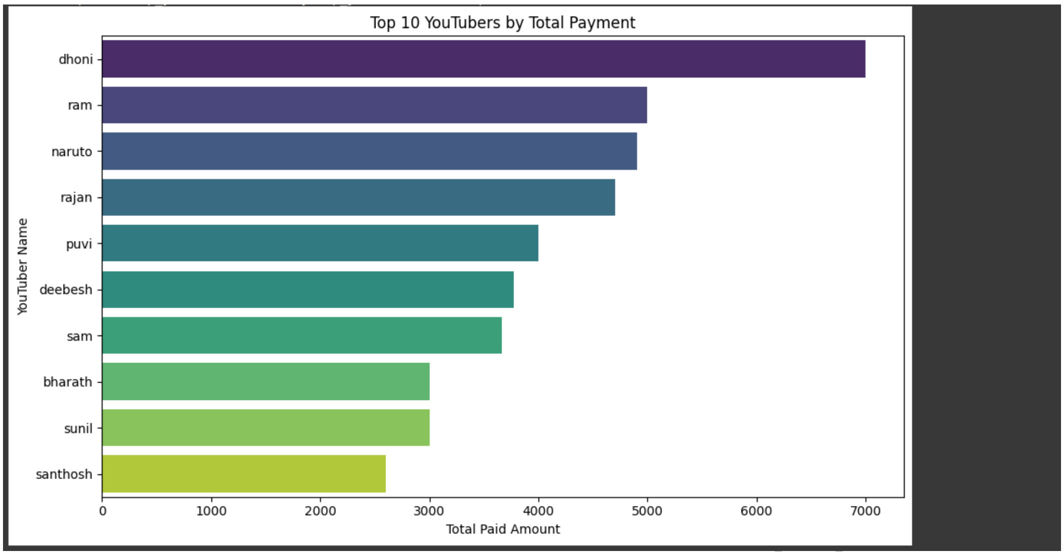
```
missing_counts = df.isnull().sum()
print("\nMissing values:\n", missing_counts)

# Drop rows where all critical fields are NaN
df_cleaned = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid',
    'fraud_detection', 'label'])

# Reset index
df_cleaned.reset_index(drop=True, inplace=True)

# Display cleaned shape
print("\nCleaned shape:", df_cleaned.shape)

# Preview cleaned data
print("\nCleaned data sample:")
print(df_cleaned.head())
```

```
Original shape: (70, 6)

Column names: ['id', 'chennal name', 'you tuber name', 'paid ', 'fraud detection', 'label']

Renamed columns: ['id', 'chennal_name', 'you_tuber_name', 'paid', 'fraud_detection', 'label']

Missing values:
 id                 0
chennal_name      52
you_tuber_name    52
paid              52
fraud_detection   52
label             52
dtype: int64

Cleaned shape: (18, 6)

Cleaned data sample:
   id                              chennal_name you_tuber_name    paid  \
0   1         Advances in AI Transform Healthcare         kathir  1000.0
1   2             NASA Announces New Moon Mission      lordjeeva  2500.0
2   3  Time Traveler Arrested for Insider Trading    jayabharath   300.0
3   4               Education Reform Bills Passed       nanthini   400.0
4   5            Scientists Confirm Earth is Flat            ram  5000.0

   fraud_detection label
0              0.0  REAL
1              1.0  FAKE
2              0.0  REAL
3              0.0  REAL
4              0.0  REAL
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load and clean the data
df = pd.read_csv("FAKEDETECTION.csv")
```

```
df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]
df = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid',
    'fraud_detection', 'label'])

# Group by YouTuber and sum payments
top_youtubers =
    df.groupby('you_tuber_name')['paid'].sum().sort_values(ascending=False).hea
    d (10)

# Display top YouTubers
print("\nTop 10 YouTubers by total payment:")
print(top_youtubers)

# Plotting
plt.figure(figsize=(10, 6))
sns.barplot(x=top_youtubers.values, y=top_youtubers.index, palette='viridis')
```

```
Top 10 YouTubers by total payment:
you_tuber_name
dhoni       6999.0
ram         5000.0
naruto      4900.0
rajan       4700.0
puvi        4000.0
deebesh     3773.0
sam         3666.0
bharath     3000.0
sunil       3000.0
santhosh    2600.0
Name: paid, dtype: float64
<ipython-input-5-94698f1f16af>:19: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=top_youtubers.values, y=top_youtubers.index, palette='viridis')
```




import pandas as pd

# Load and clean the data
df = pd.read_csv("FAKEDETECTION.csv")
df.columns =
[col.strip().lower().replace(" ", "_")
for col in df.columns] df =
df.dropna(subset=['chennal_name',
'you_tuber_name', 'paid',
 'fraud_detection', 'label'])

# Define a basic fraud detection rule
# Assume: If paid amount > 2000 and label is
'FAKE', mark as likely fraud

def detect_fraud(row):
if row['paid'] > 2000
and
row['label'].upper() ==
'FAKE': return True
return False

# Apply the rule
df['likely_fraud'] = df.apply(detect_fraud,
axis=1)

# Count and
display potential
fraud cases
fraud_cases =
df[df['likely_frau
d'] == True]
print("\nPotentia
l fraud cases
detected:")
print(fraud_case
s[['you_tuber_na

cases to a new CSV

output_path = '/mnt/data/likely_fraud_cases.csv'
fraud_cases.to_csv(ou
tput_path,
index=False)
print(f"\nLikely fraud
cases saved to:
{output_path}")

# Future Scope for Fake News Detection

```
Potential fraud cases detected:
   you_tuber_name    paid label
1       lordjeeva  2500.0  FAKE
5             sam  3666.0  FAKE
11        deebesh  3773.0  FAKE
12        bharath  3000.0  FAKE
13           puvi  4000.0  FAKE
14         naruto  4900.0  FAKE
17          dhoni  6999.0  FAKE

Number of likely fraud cases: 7
```

```
sklearn.preproce
# Load and clean the data
df = pd.read_csv("FAKEDETECTION.csv")
df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]
df = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid', 'fraud_detection', 'label'])

# Encode categorical label
label_encoder = LabelEncoder()
df['label_encoded'] = label_encoder.fit_transform(df['label'])

# Display encoding mapping
print("Label encoding mapping:")
for i, class_ in enumerate(label_encoder.classes_):
print(f"{class_}: {i}")

# Features and target
X = df[['paid', 'fraud_detection']]
y = df['label_encoded']

# Scale features
scaler = StandardScaler()
X_scaled =
scaler.fit_transform(X)

# Convert to DataFrame for inspection
X_scaled_df = pd.DataFrame(X_scaled, columns=['paid_scaled', 'fraud_detection_scaled'])

# Display the first few rows
print("\nScaled features:")
print(X_scaled_df.head())

print("\nTarget values:")
print(y.values[:10])
```

```
Label encoding mapping:
FAKE: 0
REAL: 1

Scaled features:
   paid_scaled  fraud_detection_scaled
0    -0.861758                    -1.0
1    -0.092280                     1.0
2    -1.220848                    -1.0
3    -1.169549                    -1.0
4     1.190183                    -1.0

Target values:
[1 0 1 1 1 0 1 0 1 0]
```

- import pandas as pd

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import classification_report, accuracy_score

# Load and clean the data
df = pd.read_csv("FAKEDETECTION.csv")
df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]
df = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid',
    'fraud_detection', 'label'])

# Encode labels
label_encoder = LabelEncoder()
df['label_encoded'] = label_encoder.fit_transform(df['label'])

# Prepare features and target
X = df[['paid', 'fraud_detection']]
y = df['label_encoded']

# Scale features
scaler = StandardScaler()
X_scaled =
scaler.fit_transform(X)

# Train/test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3,
    random_state=42)

# Train logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
# Make predictions
y_pred = model.predict(X_test)

# Evaluate performance
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
print("\nClassification Report:")
print(classification_report(y_test, y_pred, target_names=label_encoder.classes_))
```

```
Accuracy: 1.00

Classification Report:
              precision    recall  f1-score   support

        FAKE       1.00      1.00      1.00         3
        REAL       1.00      1.00      1.00         3

    accuracy                           1.00         6
   macro avg       1.00      1.00      1.00         6
weighted avg       1.00      1.00      1.00         6
```

## 14..Feature Scope

- **Cross-Lingual Detection**: Detecting fake news in multiple languages could become easier as NLP models continue to improve their multilingual capabilities, making the system globally applicable.

**2. Explainable AI (XAI)**

- **Enhanced Transparency**: In the future, models will offer more interpretable outputs. This would allow users to understand why a piece of news was labeled as "fake," including key phrases, sources, or logical inconsistencies that led to the decision.
- **Personalized Feedback**: The system could provide feedback tailored to the user, explaining why certain claims are classified as fake in the context of their usual reading habits or beliefs.

**3. Real-Time Fact-Checking**

- **Instant Verification**: Real-time fact-checking engines, possibly integrating with search engines and news outlets, will allow users to instantly check the authenticity of news while reading or sharing articles.
- **Crowdsourced Validation**: The future could see a more integrated approach to fact-checking where AI collaborates with crowdsourced platforms, combining the power of machine learning and human verification.

**4. Social Media Monitoring**

- **Bot Detection**: NLP systems could also extend to detecting automated or bot-driven content, which is often a significant source of fake news, especially on platforms like Twitter, Reddit, and Facebook.
- **Network Analysis**: Future systems could analyze the propagation of fake news through social media networks, understanding how misinformation spreads through social circles, and providing intervention points.

**5. Multimodal Fake News Detection**

- **Beyond Text**: As fake news can spread through images, videos, and audio, future systems could combine text-based NLP with image recognition and audio analysis. This would allow detection of fake news that includes manipulated visuals or deepfakes.
- **Deepfake Detection**: With the rise of deepfake technology, the ability to identify fake videos, manipulated audio, or altered images will be crucial. Integrating computer vision and NLP could help detect fake media content more effectively.

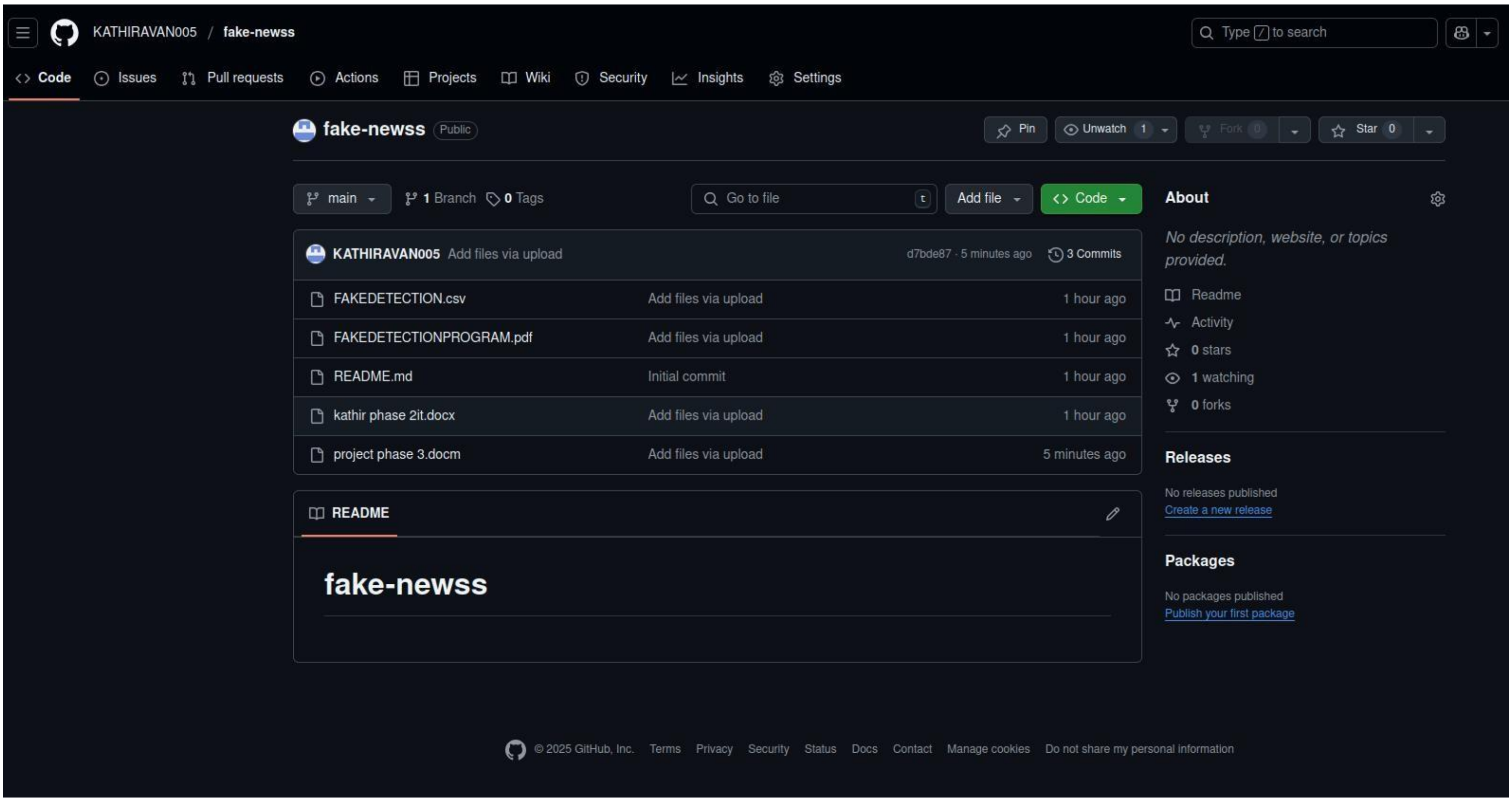
## 15.Team Members and Contribution

**JAYA BHARATH-problem statement, project objectives and flow chart of the**

**workflow JEEVA-data description,data processing and (EDA)**

**KATHIR-feature engineering and model building**

**KATHIRAVAN-visualization of result & model insights and tools and technologies used**


KATHIRAVAN005 / fake-newss
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
fake-newss
Public
Pin
Unwatch 1
Fork 0
Star 0
main
1 Branch
0 Tags
Go to file
t
Add file
Code
KATHIRAVAN005 Add files via upload
d7bde87 · 5 minutes ago
3 Commits
FAKEDETECTION.csv
Add files via upload
1 hour ago
FAKEDETECTIONPROGRAM.pdf
Add files via upload
1 hour ago
README.md
Initial commit
1 hour ago
kathir phase 2it.docx
Add files via upload
1 hour ago
project phase 3.docm
Add files via upload
5 minutes ago
README
fake-newss
About
No description, website, or topics provided.
Readme
Activity
0 stars
1 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
© 2025 GitHub, Inc.
Terms
Privacy
Security
Status
Docs
Contact
Manage cookies
Do not share my personal information

# THANK YOU