Fake and Real News Detection Project

Objective:

To build a machine learning model that can classify whether a given news article is fake or real based on its textual content.

Applications:

- Combats misinformation on social media
- Helps journalism and fact-checking
- Adds credibility to content recommendation engines

Technologies and Libraries Used:

- Python
- Pandas, NumPy – data manipulation
- Scikit-learn – ML models
- NLTK – Natural Language Processing
- TfidfVectorizer – text vectorization
- Logistic Regression – classification model

Dataset:

- Use the Fake and Real News Dataset
- Contains two CSV files: Fake.csv and True.csv
- Common columns: title, text, subject, date
- You can get the files on Kaggle

Step 1: Data Preprocessing

```
import pandas as pd


# Load the data

fake = pd.read_csv("Fake.csv")

true = pd.read_csv("True.csv")
```

```
# Add labels
fake["label"] = 0
true["label"] = 1

# Combine datasets
data = pd.concat([fake, true], axis=0)
data = data[["text", "label"]]
```

Clean and Normalize Text

```
import re
import string

def clean_text(text):
    text = text.lower()
    text = re.sub(f'[{string.punctuation}]', '', text)
    text = re.sub(r'\d+', '', text)
    return text

data["text"] = data["text"].apply(clean_text)
```

Step 2: Feature Extraction with TF-IDF

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

X = data["text"]
y = data["label"]

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Vectorize text
```

```
vectorizer = TfidfVectorizer(stop_words="english", max_df=0.7)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
```

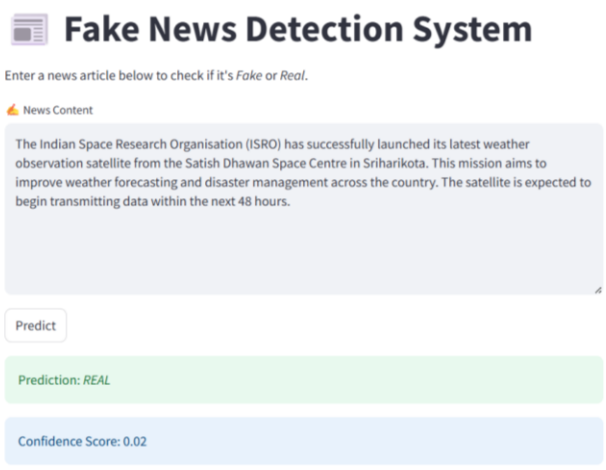
Step 3: Train Classifier (Logistic Regression)

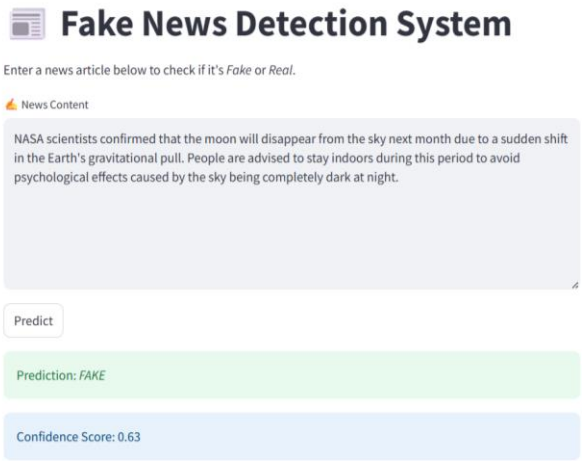
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Train model
model = LogisticRegression()
model.fit(X_train_tfidf, y_train)

# Predict
y_pred = model.predict(X_test_tfidf)

# Evaluation
print("Accuracy:", accuracy_score(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Step 4: Test with Custom Input

```
def predict_news(news):
    news = clean_text(news)
    vector = vectorizer.transform([news])
    prediction = model.predict(vector)
    return "Real News" if prediction[0] == 1 else "Fake News"

# Example
print(predict_news("Biden declares national emergency due to cyber threat"))
```

Output Sample:





Limitations:

- Model only detects based on textual patterns, not facts.
- Can be misled by well-written fake news or satirical content.
- Doesn't verify the credibility of sources.

Future Improvements:

- Use deep learning models like LSTM or BERT
- Include image-based and source-based verification
- Add a browser plugin or web app interface