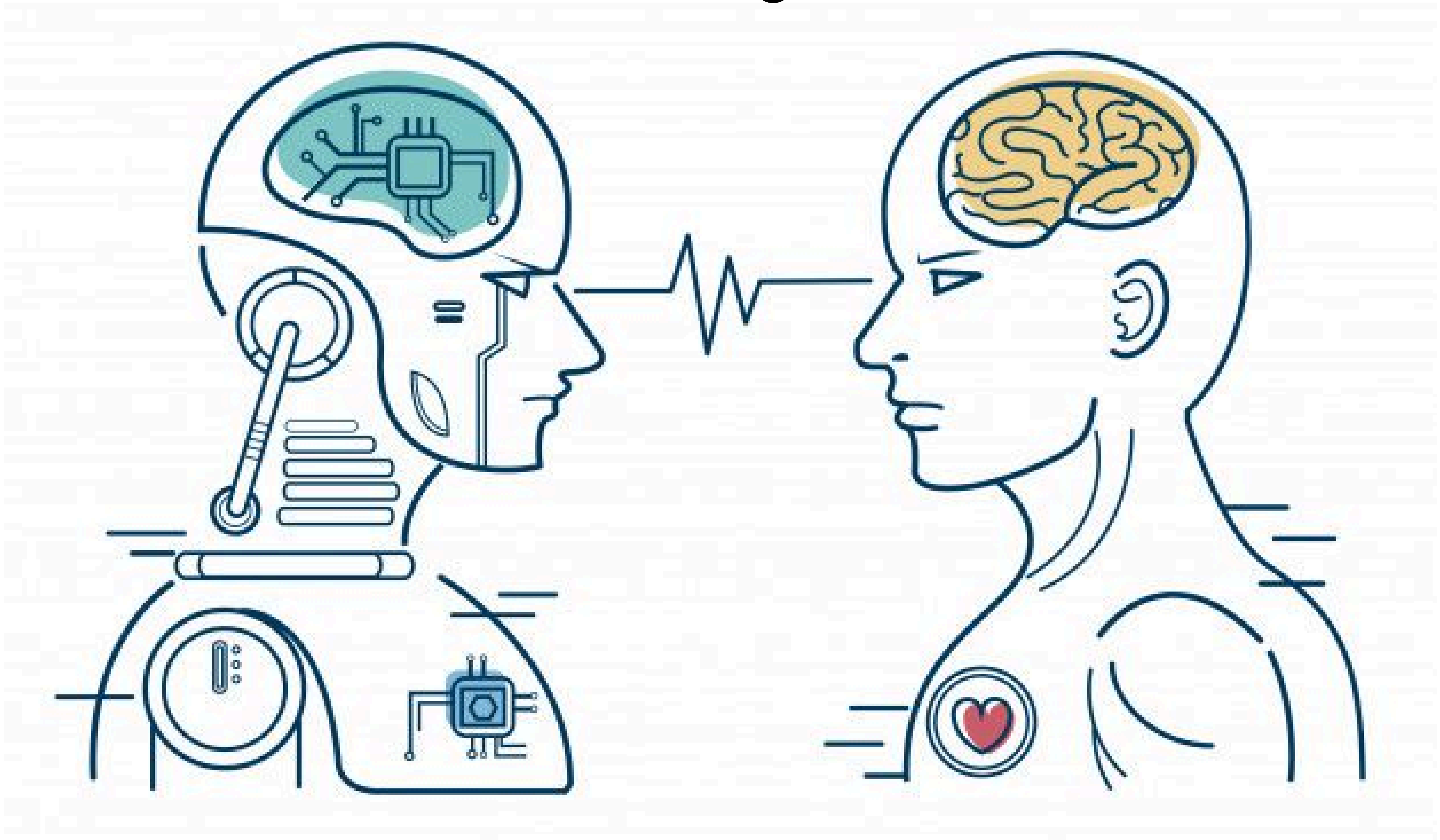
ARTIFICIAL INTELLIGENCE v/s HUMAN



TEXT CLASSIFIER

Problem Statement

With the rapid advancement of natural language generation models, particularly large language models like GPT, the distinction between human-written and AI-generated text has become increasingly indistinct.

This convergence poses significant challenges across various domains where the authenticity of authorship is critical.

There is a growing and urgent need for automated systems capable of accurately distinguishing between text authored by humans and that generated by artificial intelligence.

Such a system is essential for maintaining academic integrity, ensuring transparency in journalism, verifying originality in creative writing, and upholding trust in user-generated content on digital platforms.

MOTIVATION

1. Ensuring Academic Integrity and Authorship

As AI-generated content becomes increasingly sophisticated, there is a pressing need to safeguard academic environments from unethical practices such as automated essay writing or plagiarism via generative models.

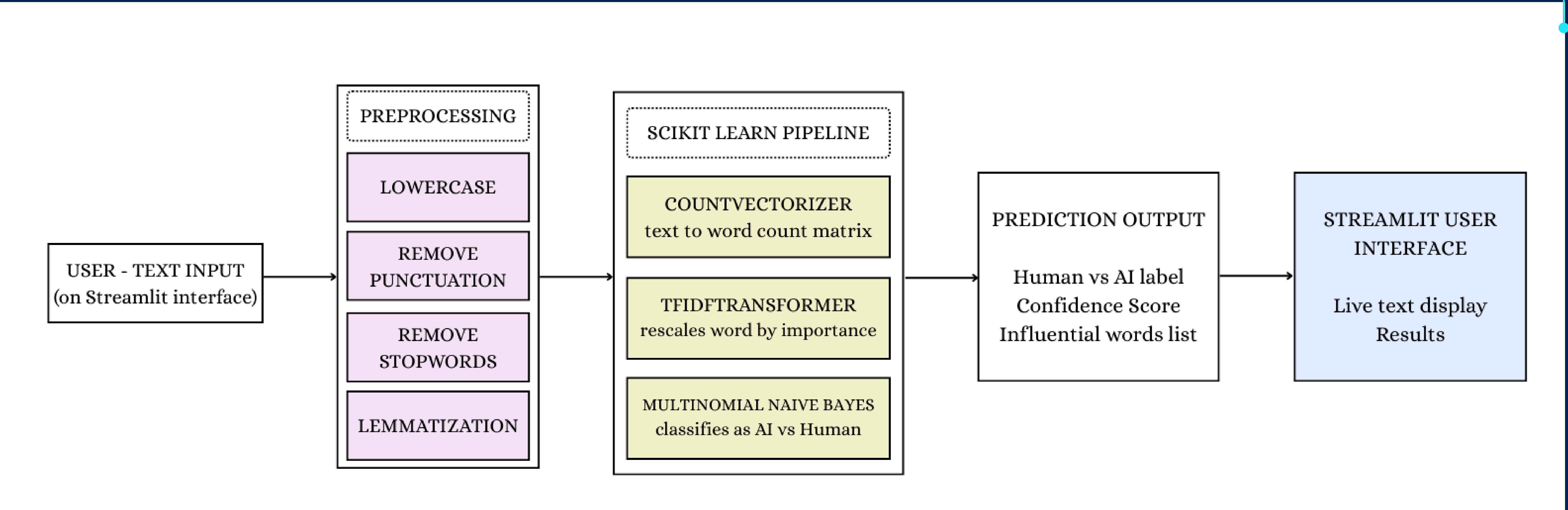
2. Enhancing Trust in Content Authenticity

Distinguishing between human and AI-generated text supports media outlets, publishers, and digital platforms in combating misinformation, maintaining credibility, and ensuring transparency in authorship.

3. Advancing Linguistic Understanding of Human vs AI Text

Exploring the subtle differences in language usage between AI and human authors contributes to the broader field of computational linguistics.

ARCHITECTURE DIAGRAM



IMPLEMENTATION DETAILS

```
[12] df.describe()
```

	generated
count	487235.000000
mean	0.372383
std	0.483440
min	0.000000
25%	0.000000
50%	0.000000
75%	1.000000
max	1.000000

Stop Words Removal

```
import nltk
from nltk.corpus import stopwords

# Download 'punkt' and 'stopwords' resources if not already downloaded
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('punkt_tab') # Download the missing punkt_tab data package

def remove_stopwords(text):
    stop_words = set(stopwords.words('english'))
    words = nltk.word_tokenize(text) # This now uses the downloaded punkt resource
    filtered_words = [word for word in words if word.lower() not in stop_words]
    filtered_words= ' '.join(filtered_words)
    return filtered_words

df['text']=df['text'].apply(remove_stopwords)
```

```
Total Texts: 487235
Human Written Texts: 305797
AI Generated Texts: 181438
```

IMPLEMENTATION DETAILS

Spell Check

```
[ ] import nltk
    from nltk.corpus import words

    nltk.download('words')
    english_words = set(words.words())


    def is_spelled_correctly(word):
        return word in english_words
```

```
[nltk_data] Downloading package words to /root/nltk_data...
[nltk_data]   Unzipping corpora/words.zip.
```

```
pipeline.fit(X_train, y_train)
```

Pipeline

CountVectorizer

TfidfTransformer

MultinomialNB

```
[34] import joblib
     import re
     import nltk
     from nltk.corpus import stopwords
     from nltk.stem import WordNetLemmatizer

     # Download required NLTK data
     nltk.download('stopwords')
     nltk.download('wordnet')

     # Load the trained classifier
     pipeline = joblib.load('text_classifier_pipeline.pkl')

     # Preprocessing function
     def preprocess(text):
         text = text.lower()
         text = re.sub(r'[^\w\s]', '', text)
         stop_words = set(stopwords.words('english'))
         lemmatizer = WordNetLemmatizer()
         words = text.split()
         words = [lemmatizer.lemmatize(w) for w in words if w not in stop_words]
         return " ".join(words)

     # === Interactive Input ===
     user_input = input("🔤 Enter the text to classify:\n")

     preprocessed = preprocess(user_input)
     prediction = pipeline.predict([preprocessed])[0]

     # Output result
     if prediction == 0:
         print("\n🧑 This text is classified as: Human-written")
     else:
         print("\n🤖 This text is classified as: AI-generated")
```

IMPLEMENTATION DETAILS

```python
text = input("Enter a sentence: ")
explain_prediction(pipeline, text)
```

```
Enter a sentence: After a long day at college, i am just so exhausted and tired out of my mind. I can't even take a nap because i have stuff, important stuff to do.

🔍 Prediction: 🧑 Human-written
✅ Confidence: 89.82%
💡 Top influential words:
  - college (importance: -5.99)
  - even (importance: -6.4)
  - day (importance: -6.59)
  - take (importance: -6.6)
  - cant (importance: -7.22)
```

```python
text = input("Enter a sentence: ")
explain_prediction(pipeline, text)
```

```
Enter a sentence: Technology has rapidly transformed the way we communicate, learn, and work. With the rise of artificial intelligence, businesses are now able to automate t

🔍 Prediction: 🤖 AI-generated
✅ Confidence: 99.85%
💡 Top influential words:
  - learn (importance: -6.75)
  - technology (importance: -6.77)
  - education (importance: -6.78)
  - work (importance: -6.78)
  - way (importance: -7.0)
```

RESULTS

```
[41] from sklearn.metrics import classification_report
     print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

         0.0       0.94      0.99      0.96     91597
         1.0       0.98      0.89      0.94     54574

    accuracy                           0.95    146171
   macro avg       0.96      0.94      0.95    146171
weighted avg       0.96      0.95      0.95    146171
```

STREAMLIT APP

🧠 Human vs AI Text Classifier with Confidence & Explanation

📝 Enter your text below:

After a long day at college, i am just so exhausted and tired out of my mind. I can't even take a nap because i have stuff, important stuff to do.

🔍 Classify

🧑 This text is classified as **Human-written**

🔎 Confidence: `89.82%`

💬 Top Influential Words:

- `college` (importance: -5.99)
- `even` (importance: -6.4)
- `day` (importance: -6.59)
- `take` (importance: -6.6)
- `cant` (importance: -7.22)

🧠 Human vs AI Text Classifier with Confidence & Explanation

📝 Enter your text below:

Technology has rapidly transformed the way we communicate, learn, and work. With the rise of artificial intelligence, businesses are now able to automate tasks, analyze vast datasets, and deliver personalized experiences to customers. AI tools are revolutionizing industries such as healthcare, finance, and education by improving efficiency and accuracy.

🔍 Classify

🤖 This text is classified as **AI-generated**

🔎 Confidence: `99.85%`

💬 Top Influential Words:

- `learn` (importance: -6.75)
- `technology` (importance: -6.77)
- `education` (importance: -6.78)
- `work` (importance: -6.78)
- `way` (importance: -7.0)

Scope and Limitations

1. Focus on Short to Medium-Length English Texts

The current system is optimized for analyzing short to moderately long passages written in English, such as essays, paragraphs, and opinion-based writing. It may not be well-suited for analyzing highly technical documents, multi-language content, or full-length books/articles without further customization or preprocessing adjustments.

2. Utilization of Classical Machine Learning Techniques

This project leverages classical machine learning models—specifically a Multinomial Naive Bayes classifier—within a Scikit-learn pipeline these models are lightweight, interpretable, and effective for text classification tasks but may not capture deeper contextual semantics.

3. Reduced Performance on Heavily Paraphrased or Noisy Inputs

The classifier's accuracy may decline when presented with content that is deliberately paraphrased, syntactically unconventional, or embedded with excessive noise

FUTURE WORK

1. Integration of Transformer-Based Language Models

To enhance contextual understanding and improve classification accuracy, future iterations of the system can incorporate state-of-the-art transformer-based models such as BERT, RoBERTa, or DistilBERT.

2. Support for Batch Classification via File Uploads

To accommodate large-scale use cases, the system could be extended to accept bulk inputs through .txt, .csv, or .json file uploads.

3. Automated Generation of PDF Reports

For documentation and audit purposes, the system could include functionality to generate detailed PDF reports for each classification result. These reports would summarize the input, predicted label, confidence score, and influential words, providing a standardized format for record-keeping, review, or academic assessment.