AI vs Human Text Classifier - Final Report

1. Introduction

In recent years, the distinction between human-generated and AI-generated content has become increasingly blurred. With the rise of large language models such as ChatGPT, GPT-4, and other generative AI systems, detecting the origin of text has become a valuable challenge. This project, 'AI vs Human Text Classifier', addresses that challenge through a machine learning approach implemented in a full-stack web application. It provides a seamless user experience that allows any individual to input text and receive an instant classification - indicating whether the text was written by a human or generated by an AI model.

2. Backend Development (Machine Learning and Flask)

The backend was built using Python and the Flask framework. The heart of the project is a Jupyter Notebook, which performs the following tasks:

- Loads a custom dataset comprising labeled text samples from both humans and AI.
- Cleans and preprocesses the text by removing punctuation, stopwords, and normalizing casing.
- Converts raw text into numerical features using TF-IDF vectorization.
- Trains a Logistic Regression model to distinguish between the two classes.

After training, the model and TF-IDF vectorizer are serialized using Joblib and saved as 'classifier.pkl' and 'vectorizer.pkl' respectively. A utility file (utils.py) ensures the preprocessing pipeline used during training is reused during prediction. The Flask app integrates all these components and defines a POST endpoint to process form submissions and return predictions.

3. Frontend Development (HTML, CSS, Tailwind & Bootstrap)

The frontend interface is designed to be both responsive and aesthetically appealing using modern styling frameworks. Tailwind CSS and Bootstrap are combined to create a visually rich layout. The main elements include:

- A dynamic background gradient.
- A translucent white card-like container housing the title, input form, and prediction result.
- An overlay of large, faded text ('Machine Learning Project') placed behind the form for stylistic

emphasis.

- Animated hover effects and smooth focus transitions for the text input area.
- Display of prediction result in a visually highlighted section.
- Developer credits fixed at the top right corner of the screen.

The design focuses on enhancing readability and user interaction while ensuring professional presentation.

4. Project Structure and Files

The project is structured into several key components:

- app.py: Main Flask server logic, routing, and model integration.
- model/utils.py: Text preprocessing logic including cleaning and normalization.
- model/classifier.pkl: Serialized Logistic Regression model trained in Jupyter.
- model/vectorizer.pkl: Serialized TF-IDF vectorizer.
- templates/index.html: HTML layout using Tailwind and Bootstrap.
- static/styles.css: Custom CSS animations, hover effects, and layout polish.
- AI_vs_Human_Text_Classifier_newdata.ipynb: Notebook used for data preparation, model training, evaluation, and exporting final assets.

This modular structure allows for easy maintenance and future extensibility - for example, switching models or integrating deep learning in the backend.

5. How to Deploy and Use the Application

To run the application locally, follow these steps:

1. Install required dependencies:

```
pip install flask joblib nltk scikit-learn
```

2. Ensure all project files are placed in their respective directories as described.
3. Run the Flask server:

```
python app.py
```

4. Open your browser and go to http://127.0.0.1:5000/

5. Paste any text into the input box and click 'Classify' to receive a prediction.

This simple setup makes the application lightweight and perfect for demonstrations or academic use. Future versions may integrate advanced models or connect to external APIs for real-time data classification.

6. Final Thoughts and Credits

This project successfully demonstrates how machine learning can be embedded into web interfaces to create practical, user-friendly tools. It showcases the combined power of data science and software development, all while maintaining a focus on clean design and user experience.

Developed by:
- Hassaan Bari (261934233)
- Rehman Ibrahim (261936230)

We hope this application not only meets its functional objectives but also inspires further exploration in the domain of AI interpretability and detection.