LLM Text Detector

Project Report

Name-Paaban Panda

Branch-EE

Enrolment Number-22115110

Dataset – Dataset downloaded from Kaggle and used to train the model. It was in csv form.

Then data is analysed in an jupyter notebook. Studied various aspects of data like distribution of classes, plotted graph of various aspects of dataset.

Data Preprocessing- In a .py file, made various functions which are called in main.py. Functions including loading data, preprocessing data, splitting data and processing the text data.

```python
# Function to load data from an Excel file
def load_data(file_path):
    # Read the Excel file into a pandas DataFrame
    df = pd.read_excel(file_path)
    return df
# Function to preprocess data
def preprocess_data(df):
    # Use LabelEncoder to encode the 'generated' column (0: human-written, 1: AI-generated)
    label_encoder = LabelEncoder()
    df['generated'] = label_encoder.fit_transform(df['generated'])
    return df
# Function to split the data into training and testing sets
def split_data(df):
    # Split the data into features (X) and target variable (y)
    X_train, X_test, y_train, y_test = train_test_split(df['text'], df['generated'], test_size=0.2, random_state=42)
    return X_train, X_test, y_train, y_test
# Function to preprocess text data using TF-IDF vectorization
def preprocess_text_data(X_train, X_test):
    # Fill NaN values in text data with an empty string
    X_train = X_train.fillna('')
    X_test = X_test.fillna('')
    # Initialize a TF-IDF vectorizer with a maximum of 1000 features and stop words in English
    tfidf_vectorizer = TfidfVectorizer(max_features=1000, stop_words="english")
    # Transform the training and testing text data using TF-IDF vectorization
    X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
    X_test_tfidf = tfidf_vectorizer.transform(X_test)
    return X_train_tfidf, X_test_tfidf, tfidf_vectorizer
```

Training Model- In a .py file made 3 functions which are again called in main.py, these functions are intended to train the model. Functions include training the model, evaluating the model and to save the model and vectorizer.

```python
def train_model(X_train, y_train):
    model = RandomForestClassifier(n_estimators=100, random_state=42)
    model.fit(X_train, y_train)
    return model

def evaluate_model(model, X_test, y_test):
    predictions = model.predict(X_test)
    accuracy = accuracy_score(y_test, predictions)
    print(f"Accuracy: {100*accuracy}%")

def save_model(model, vectorizer, model_path, vectorizer_path):
    # Save the trained model
    joblib.dump(model, model_path)

    # Save the TF-IDF vectorizer
    joblib.dump(vectorizer, vectorizer_path)
```

The trained model and the trained vectorizer are stored in form of pkl file inside the model directory.

In the main.py I called all the functions that were intended to preprocess and train the model. On running the main.py the data is pre-processed and model is trained.

Predicting the output- A predict.py file is made where two functions are created and called for loading model and to predict output.

```python
def load_model_and_vectorizer(model_path, vectorizer_path):
    # Load the trained model
    model = joblib.load(model_path)

    # Load the TF-IDF vectorizer
    vectorizer = joblib.load(vectorizer_path)

    return model, vectorizer

def predict_text(model, vectorizer, text):
    # Vectorize the input text using the TF-IDF vectorizer
    text_tfidf = vectorizer.transform([text])

    # Make a prediction using the trained model
    prediction = model.predict(text_tfidf)

    return prediction
```

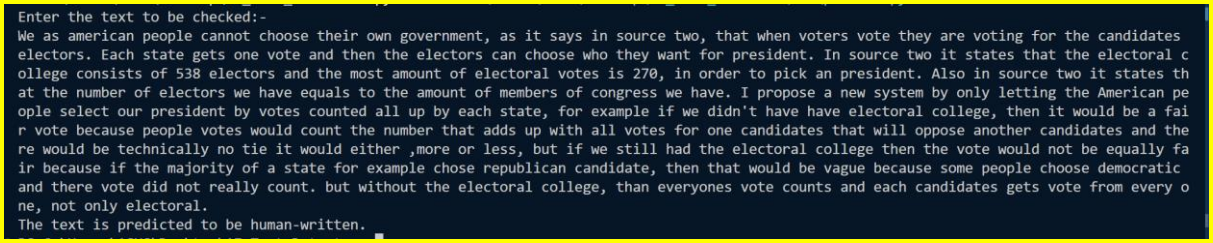
Based on user's input the model will predict if the text is generated or not by AI.

```python
if prediction[0] == 0:
    print(f"The text is predicted to be human-written.")
elif prediction[0] == 1:
    print(f"The text is predicted to be AI-generated.")
else:
    print(f"Invalid prediction result.")
```

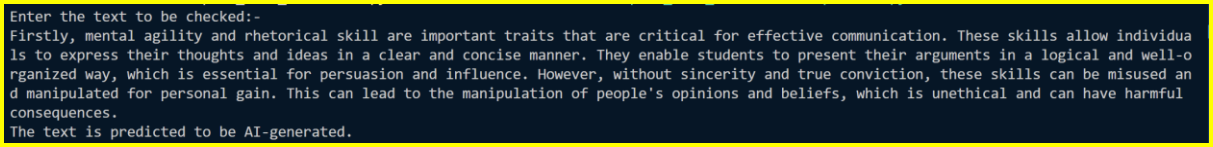
Prediction Result-

On running the predict.py , it will ask to enter a paragraph.

Here is demo of two outputs.



Here the model predicted that the text is written by a human.



Here the model predicted that the text is generated by AI.