# **LLM - Detect AI Generated Text**

# 

1. Problem :

* Understanding the approach
* Evaluation metrics
* Data set skim (LLM-generated essays and student-written ones)

1. Result:

* Model that can determine whether an essay was written by a middle or high school student or if it was generated by a large language model (LLM)

1. Data Exploration
2. Feature Engineering:

* Features from the text data
* Distinguish the features of the easy written by the students and LLM generated ones

1. Model Development
2. Training and evaluation
3. Testing
4. To verify the submission file format

Main aim of the project is to check where the model's efficiency in correctly identifying the origin of essays and how well it generalizes to unseen data.

* **{test|train}\_essays.csv**
  + id - A unique identifier for each essay.
  + prompt\_id - Identifies the prompt the essay was written in response to.
  + text - The essay text itself.
  + generated - Whether the essay was written by a student (0) or generated by an LLM (1). This field is the target and is not present in test\_essays.csv.
* **train\_prompts.csv** - Essays were written in response to information in these fields.
  + prompt\_id - A unique identifier for each prompt.
  + prompt\_name - The title of the prompt.
  + instructions - The instructions given to students.
  + source\_text - The text of the article(s) the essays were written in response to, in Markdown format. Significant paragraphs are enumerated by a numeral preceding the paragraph on the same line, as in 0 Paragraph one.\n\n1 Paragraph two.. Essays sometimes refer to a paragraph by its numeral. Each article is preceded with its title in a heading, like # Title. When an author is indicated, their name will be given in the title after by. Not all articles have authors indicated. An article may have subheadings indicated like ## Subheading.
* **sample\_submission.csv** - A submission file in the correct format. See the [**Evaluation**](https://www.kaggle.com/competitions/llm-detect-ai-generated-text/overview/evaluation) page for details.

# **LLM - Detect AI-Generated Text using Transformer:**

* Test Essays (test\_essays.csv):
  + Contains columns id, prompt\_id, and text.
  + The text column likely contains essay texts.
  + Sample Data:
    - ID: 0000aaaa, Prompt ID: 2, Text: "Aaa bbb ccc."
    - ID: 1111bbbb, Prompt ID: 3, Text: "Bbb ccc ddd."
* Train Essays (train\_essays.csv):
  + Contains columns id, prompt\_id, text, and generated.
  + The text column contains essay texts, similar to the test essays.
  + The generated column indicates whether the essay was generated (0 or 1).
  + Sample Data:
    - ID: 0059830c, Prompt ID: 0, Text: "Cars. Cars have been around...", Generated: 0
    - ID: 005db917, Prompt ID: 0, Text: "Transportation is a large necessity...", Generated: 0
* Train Prompts (train\_prompts.csv):
  + Contains columns prompt\_id, prompt\_name, instructions, and source\_text.
  + This file provides the prompts for the essays.
  + Sample Data:
    - Prompt ID: 0, Prompt Name: "Car-free cities", Instructions: "Write an explanatory essay to inform fellow citizens...", Source Text: "In German Suburb, Life Goes On Without Cars..."
    - Prompt ID: 1, Prompt Name: "Does the Electoral College work?", Instructions: "Write a letter to your state senator in which...", Source Text: "What Is the Electoral College? by the Office..."

For a thorough EDA, we can explore various aspects such as:

* Distribution of essays across different prompts.
* Length of essays in words or characters.
* Analysis of the generated flag in training essays.
* Text analysis, like common words or themes in essays.
* Any correlations between essay characteristics and prompts.

feature engineering techniques:

* Text Length Metrics:
  + Word Count: Number of words in each essay.
  + Character Count: Number of characters in each essay.
  + Sentence Count: Number of sentences in each essay.
  + Average Word Length: Average length of words in each essay.
* Text Complexity Metrics:
  + Lexical Diversity: Ratio of unique words to total words.
  + Readability Scores: Such as Flesch Reading Ease or Flesch-Kincaid Grade Level.
* N-grams and Bag of Words:
  + Create features based on the frequency of individual words (unigrams) or sequences of words (bigrams, trigrams, etc.).
* TF-IDF (Term Frequency-Inverse Document Frequency):
  + Reflects how important a word is to a document in a collection or corpus.
* Sentiment Analysis:
  + Analyze the sentiment of the essays (positive, negative, neutral).
* POS Tagging (Part of Speech):
  + Analyze the distribution of different parts of speech (nouns, verbs, adjectives, etc.).
* Prompt Features:
  + If your prompts have categorizable features (like subject area), these can be encoded and used as features.
* Embeddings:
  + Use pre-trained word embeddings (like Word2Vec or GloVe) or sentence embeddings (like BERT) to capture semantic information.
* Syntactic Features:
  + Parse tree depth, syntactic complexity, etc.
* Custom Features Based on Domain Knowledge:
  + Any specific features that might be relevant based on the nature of the essays or prompts.

Reference: