**Text Humanization:**

For the first task I had to come up with creative rule base approach to try and humanized the text output of an LLM,   
  
Main Ideas:  
- **Eliminate** immediate red flags, things like ‘as an AI’, Markdowns.  
- **Replace** text, grammar, reduce formality, slang, typos, etc.  
- **Insert** emojis, filler text.  
- **Randomness**, not apply all possible rules at once, factor in text length and rule weights.  
 (weight of one states that it should always be applied, withing the max rules count)

In applying the rules we have to keep in mind the followings: Rule order, Rule weight, Max transformation.  
  
For rule ideas and examples and implementation I leverage GPT (for ideas) and Curser with Sonnet (for code and examples), the prompts are shown in the MD files.

On top of the implementation, I created some tests for example.

\*Note I’m not yet convinced that rule base it the right approach to detect and ‘fix’ the text, this might be a simplistic solution and for a personalized version of outputs we would require advance techniques (Prompt Engineering, RAG, Fine Tuning).

**Intent Classification:**

For the Second task I had to adapt LLM output into a classification tasks, with GPT’s workflow there are some possibilities to implement this. I chose to leverage **response\_format** parameter and adapting the response to my use case by defining a **Pydantic** class and set it as the structure response.  
  
**IntentClassificaiton** format that holds in it an explanation string and intent as an **Enum**

I’ve chosen a GPT mini model for this since this kind of task is light and we can provide the results with low latency, set it to temperature to zero and seed for 42 for deterministic and reproducible results.

It’s worth mentioning that the addition of the explanation part first does affect the performance, it forces the model to ‘think’ before the classification.

By going back and forth with Curser I finalized the prompt for this use case and after an initial example attempts, I generated a dataset so we could evaluate the classification task.

In the confusion matrix the F1-score variations are roughly the same (unsurprisingly base on our generated dataset) and with an overall Accuracy of 0.88.  
Also we can see in the examples where the prediction was wrong that the text is actually something that could be intemperate in two ways (the two classes involved).

For further development we might benefit from having a secondary classification , create a better separation of classes