- The main pipeline of the project is implemented in main.ipynb
- We mainly focused for weeks on 20 question game, as it seemed the most challenging part of the project. We have implemented different strategies in different notebooks as described below:
    1. **Entropy Minimization Policy based on LLM logits** in *"LLM_Entropy_based_Method.ipynb":* In this file we look at the problem of 20 questions game from a game theoretic approach and find optimal question and guesses that reduces the uncertainty the most at each step.
    At first we also included some rule-based questions as TAs mentioned Kaggle 20q competition and many of the winners have used this method. Later, TAs mentioned that only llm logits must be used. So we ignored them at the evaluation phase at the end and in evaluation we just use entropy method based on llm answer logits to semantic questions (the rule-based code is still there but not used at the end).
    Hugging Face resources:
        - *LLM logits for semantic questions*
        - *Full data: 3 subsets: semantic_table, rule-based table, nouns table*
        - *Optimal Policy tree: optimal semantic question and guesses (up to depth 15)*

    we got 92% accuracy with just optimal semantic questions calculated from llm logits to (semantic question , word) pairs. For evaluation you can only run *"LLM_Entropy_quick_run.ipynb"* and it will fetch required data from HF.

    2. **Supervised Finetuning on Public Gameplays:** in *"SFT_on_Publice_Gameplays.ipynb"*: In this file we supervisely finetune a mistral7b with a very small QLORA adaptor due to the resource limits. We combined two datasets available in Internet which have 20q gameplays. They don't make guess at each step, so the model will become expert more on making questions. Due to resource limits, we couldn't even finetune 1 epoch (I guess we did 10k steps) but the accuracy reached about 40%. (It's more for demonstration and I guess this limited SFT on Kaggle should not make major difference). We also tried so long to perform RLHF finetuning after SFT, but even after extensive memory optimizations we couldn't fit all RLHF models to Kaggle memory.
    The last section in the notebook is for evaluation. You can just install required packages at first and start from this section. The model will be loaded from last checkpoint on Hugging Face.

3. **Supervised Finetuning on Optimal Policy tree:** in *"SFT_on_Optimal_Policy_tree.ipynb":* In this notebook we extracted samples from optimal policy tree in first section up to depth 11 (for each episode with horizon=k, the best_guess at the depth=k in that path was considered correct guess). Then we performed SFT on mistral7b using these samples.
   At first, it seemed like the ultimate solution, because the gameplays was ideal and optimal, and they also include guesses, but unfortunately we didn't get enough accuracy and we didn't have time and resource to investigate this method more.
4. **Pure LLM with Reasoning and Corrective Prompting:** in *"pure-llm-with-reasoning-and-corrective-prompting.ipynb":* We saw some works that claimed that if we explicitly want the model to write its reasoning step-by-step and then make the final answer, the performance can improve significantly. We did this with prompt engineering in system and user prompt and the performance really improved with respect to pure llm. We also regenerate question with corrected prompt if question is duplicate or not in correct format.
5. **Pure LLM:** in *"main.ipynb":* This is just the pure LLM gameplay with appropriate user and system prompts which has accuracy about 35%. The test script is written inside the notebook.