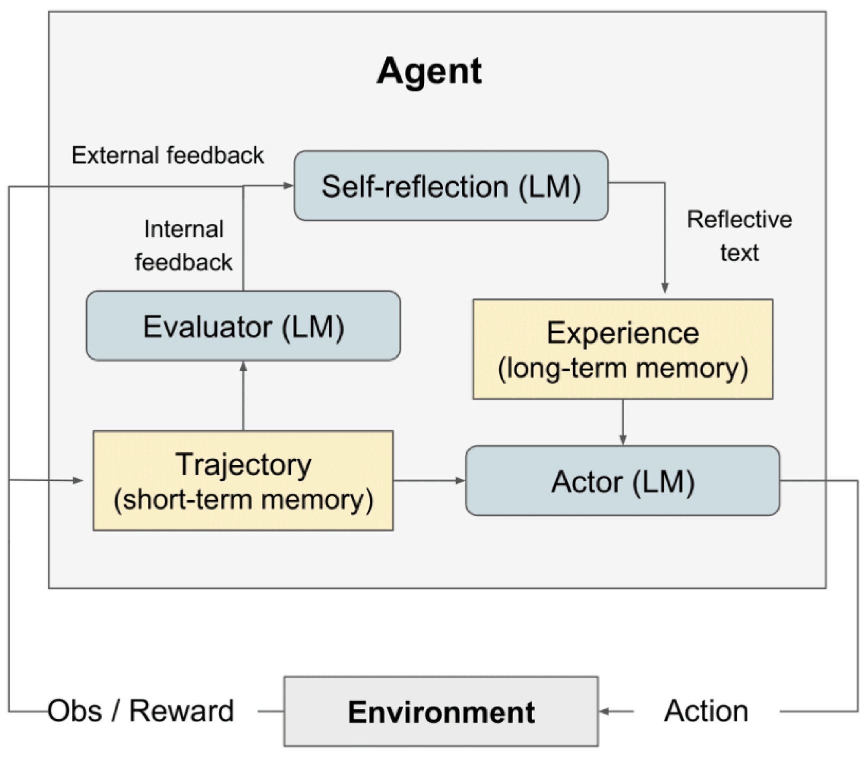
Using LLM to Generate Code for Classification and Regression Tasks

Shiqi Zhang , Shengyao Chen , Pengrui Lu

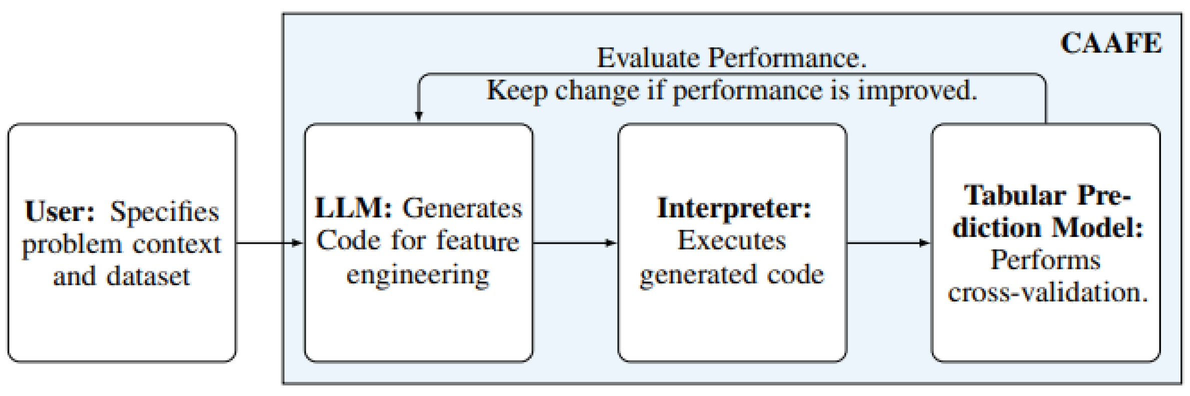
SHANGHAI JIAO TONG UNIVERSITY

Background

Reflexion



CAAFE



Terminologies

Reflexion

Iteratively analyze and refine LLM outputs through self-evaluation.
generator: LLM that generates code and self-Reflection.

executor: Executes the generated code and provides feedback.

CAAFE

Provides context-sensitive feedback for automated feature engineering. Only used in classification tasks.

Supervised Fine-Tuning

Trains LLMs on specific tasks using labeled datasets.

LoRA

Efficient fine-tuning with low-rank matrices for large models.

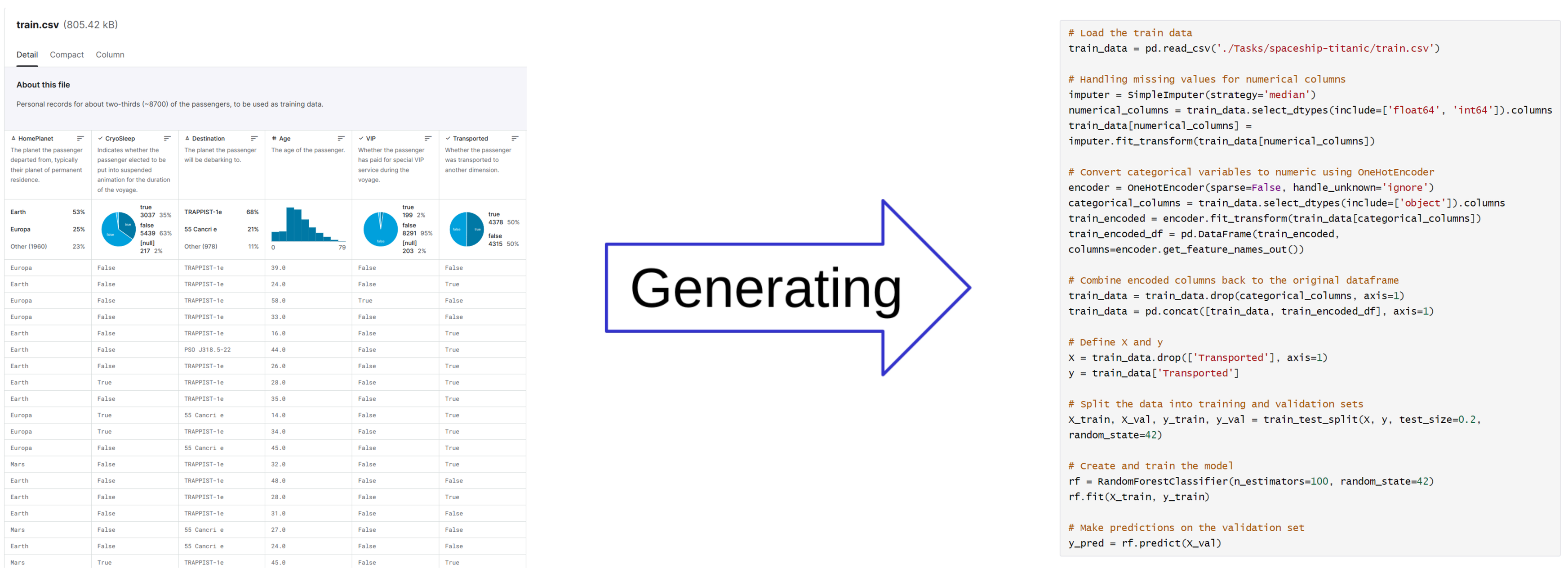
LLaMA-Factory

Framework for efficient fine-tuning and deployment of LLMs.

Task

Task Description

Generate code for machine learning tasks, focusing on classification and regression.



Challenges

Code Correctness : LLMs always generate wrong code.

Insufficient GPUs : Deploying or finetuning a large model requires too much computing power.

Limited Data : Few datasets are available for fine-tuning code generation, especially for classification/regression tasks.

Our Approach

Apply Reflexion

1. **Initial Generation**: Model generates code based on the input task.
2. **Evaluation**: The executor evaluates the generated code and provides feedback(error, performance, etc.).
3. **Self-Reflection Generation**: The generator model generates self-reflection words based on the code generated and the feedback.
4. **Code Refinement**: The generator model refines the code based on the self-reflection words.
5. **Iterative Process**: These steps are repeated until the generated code meets the

desired quality standards or a preset number of iterations is reached.

Apply CAAFE

prompt engineering: Designing prompts to guide the generator model to generate code using CAAFE.

Supervised Fine-Tuning

LoRA: Low-Rank Adaptation (LoRA) fine-tunes large language models efficiently by introducing low-rank matrices, reducing parameters and computational overhead while maintaining or enhancing performance.

Experiments

Setup

Data Preparation

1. **Data Collection**: Datasets were collected from Kaggle, including both simple and complex feature datasets.
2. **Data Preprocessing**: Steps included extracting task descriptions, target labels, and several example rows from the datasets.

Datasets

- **Tasks**: House Price Dataset, Spaceship Titanic, Mobile Price Classification, etc.
- **Fine-tuning**: AlpacaCode

Compared Models

We compared the following models in our experiments:

- **Code Llama 7B**: A model designed for code generation tasks.
- **Qwen 0.5-1.5B Chat**: A chat-oriented model with varying parameter sizes.
- **Qwen 7B Chat**: A larger chat-oriented model.
- **Llama3 8B Instruct**: An instruction-tuned version of the Llama3 model.
- **Llama-3-8b-Instruct-bnb-4bit**: A quantized version of the Llama3 8B Instruct model for efficient fine-tuning.

Results

Reflexion

model	Dataset	None	Using Reflexion
Llama3 8B Instruct	House Price Dataset	error or 0.19(rank 3800+)	0.15(rank 2840)
Qwen 7B Chat	Spaceship Titanic	cannot correctly generate code	0.787(rank 1736)

Table 1: performances of Reflexion

CAAFE

model	Dataset	None	Using CAAFE
Llama3 8B Instruct	Spaceship Titanic	0.75(rank 2150+)	0.79(rank 1120)

Table 2: performances of CAAFE