



Применение LLM-агентов для AutoML

Чумаков Станислав

@lemoncolgate

05.09.2024

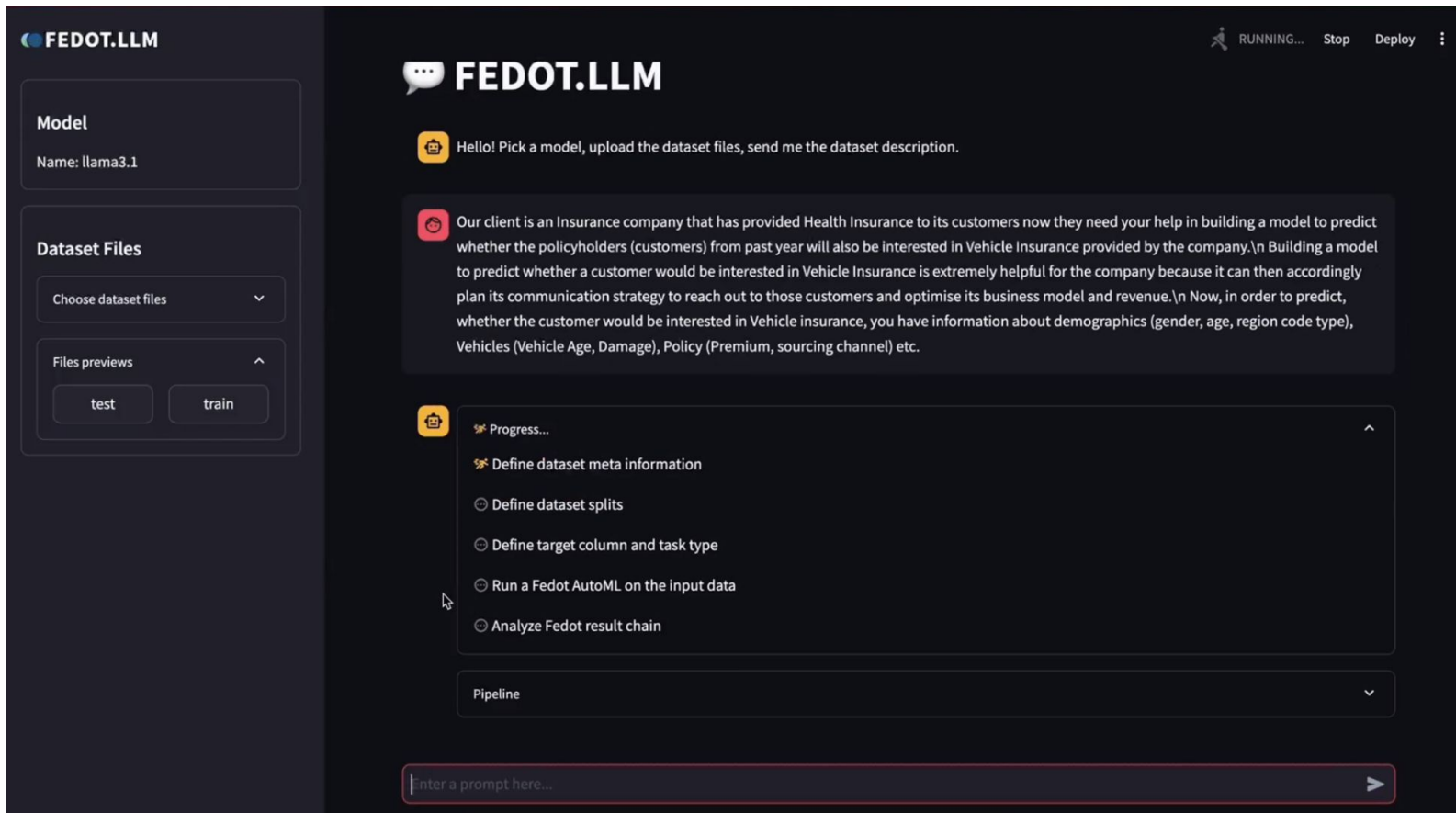
Так как конфигурирование, оперирование и интерпретация AutoML систем (как FEDOT) требует обширных знаний, опыт применения агентных моделей может упростить многие аспекты системы

Направления

1. Предобработка входящих данных
2. Запуск AutoML-фреймворка
3. Постобработка результатов работы фреймворка
4. Интеграция знаний эксперта

Применение LLM-агентов в AutoML

- Реализация агентов - универсальный интерфейс для обращения к локальным (ollama) или request-based моделям на основе BaseChatModel из LangChain, используется LLaMa 3.1 8b
- На данный момент все рассчитано на английский
- На данный момент отсутствует планировщик - присутствуют обязательные этапы уточнения данных, запуска фреймворка, объяснения результатов



FEDOT.LLM

RUNNING...

Stop

Deploy

company because it can then accordingly plan its communication strategy to reach out to those customers and optimise its business model and revenue.\n Now, in order to predict, whether the customer would be interested in Vehicle insurance, you have information about demographics (gender, age, region code type), Vehicles (Vehicle Age, Damage), Policy (Premium, sourcing channel) etc.

Progress...

✓ Define dataset meta information

{'name': 'Insurance Customer Segmenta...

✓ Define dataset splits

{'test': 'train', 'train': 'train'}

✓ Define target column and task type

{'task_type': 'classification', 'tang...

✓ Run a Fedot AutoML on the input data

{'predictions': array([[0]], [0...

✂ Analyze Fedot result chain

Pipeline

Define dataset meta information

→

Define dataset splits

→

Define target column and task type


→

Run a Fedot AutoML on the input data

→

Analyze Fedot result chain

Enter a prompt here...


FEDOT.LLM
>
Deploy
⋮

Define dataset meta information

Define dataset splits

Define target column and task type

Run a Fedot AutoML on the input data

Analyze Fedot result chain

Here is the pipeline of the model I built:

Model Pipeline

The pipeline consists of:

- A Random Forest (RF) stage with 12 jobs processed in parallel (`n_jobs=12`)
- A scaling stage (no specific details provided)

Model Metrics

Metric	Value
roc_auc	0.844
accuracy	0.871

These metrics indicate that the model has a good ability to distinguish between classes, with an Area Under the Receiver Operating Characteristic Curve (AUC) of 0.844. The accuracy of the model is also high, at 0.871, indicating that it correctly predicts the class label for most instances in the test dataset.

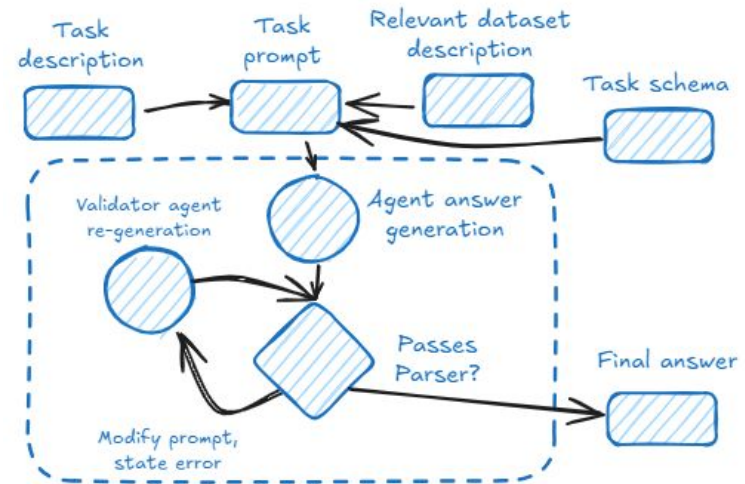
Enter a prompt here...
>

Предобработка входящих данных iТМО

Изначально в фреймворк поступают сырые файлы с данными и общее описание задачи

Путем приведения данных к текстовому описанию и итеративному промптированию через LangChain устанавливаются:

- Роли различных сплитов
- Информация о проблеме: Цель и тип задачи
- Определение описаний и выделение категориальных признаков



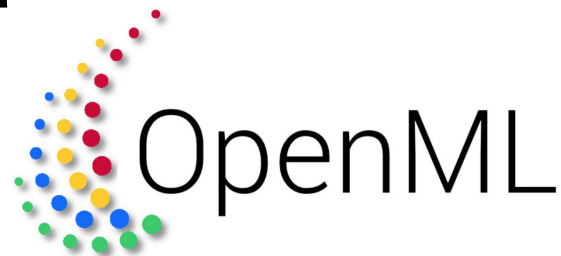
Проверка результатов, при ошибке - промпт модифицируется с сообщением об ошибке и передается на вторую модель

Предобработка входящих данных ITMO

**Бенчмаркинг - тестирование последовательности
уточнения вплоть до желаемого этапа**

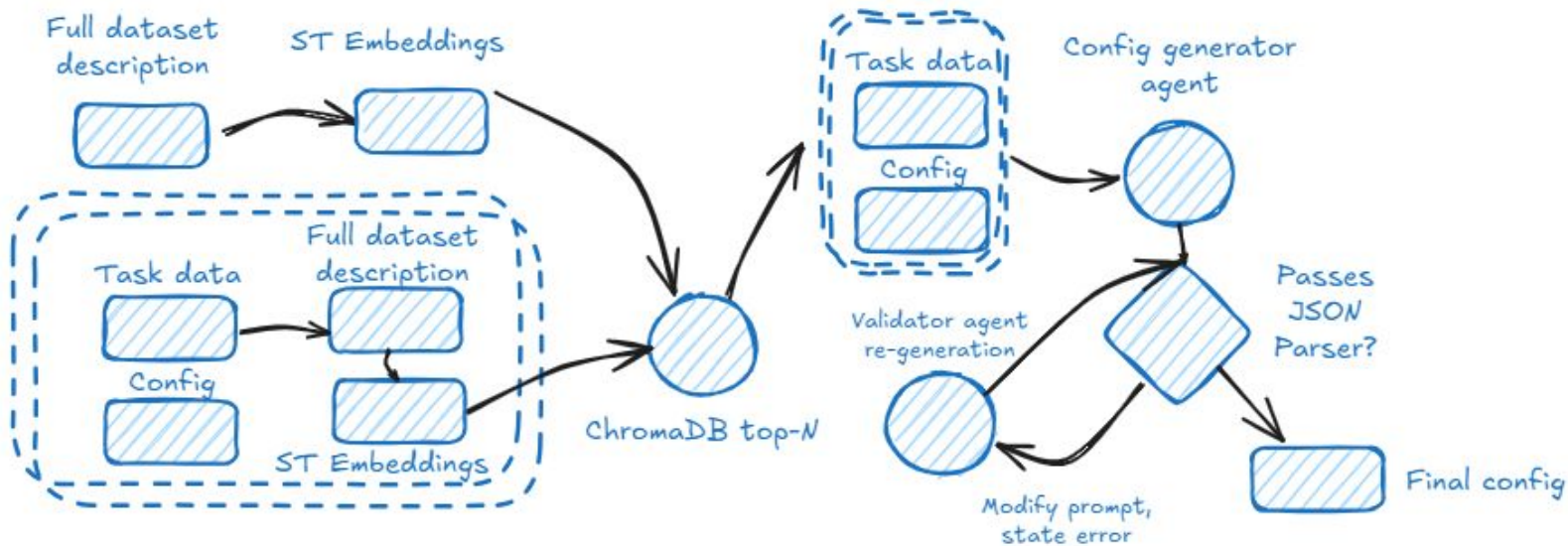
- Выделение категориальных признаков
- Информация о проблеме: тип задачи
- Информация о проблеме: Target
- ...

Размеченные данные с
репозиториях + дизайн описаний
(простые, сложные)



Запуск AutoML-фреймворка

Используя наиболее полное описание датасета, производится поиск с SentenceTransformers по базе с кейсами применения AutoML



FEDOT запускается с учетом полученных ранее метаданных и конфигурации

Компоненты:

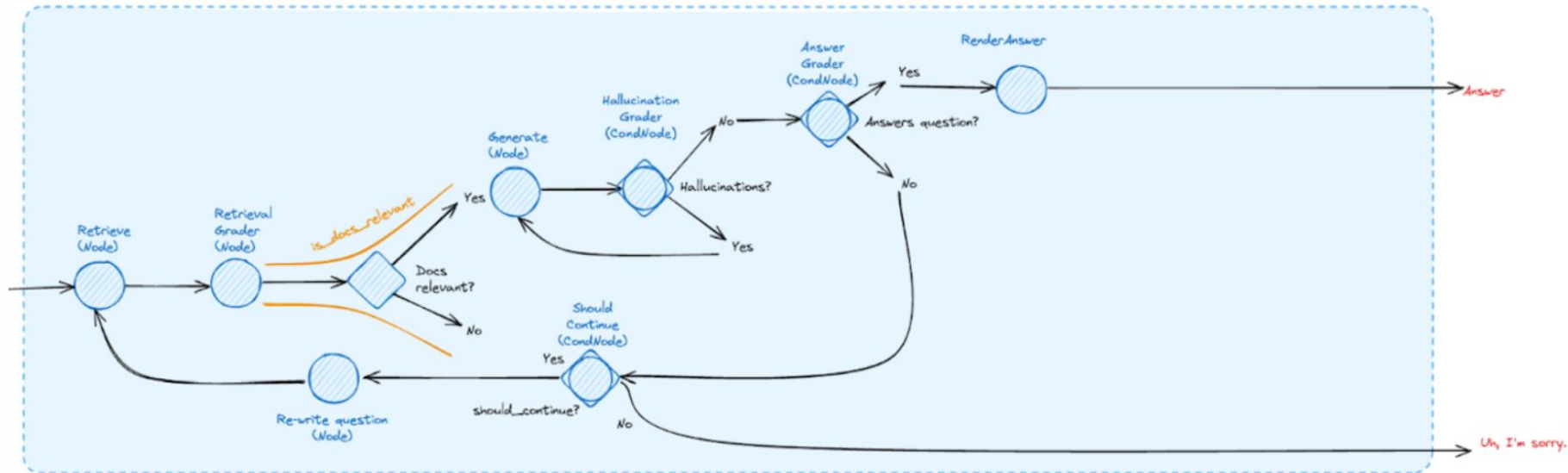
- Анализ пайплайна, предложенного AutoML (агент)
- Предложение визуализации результатов
- Генерация текстового описания получившегося решения и хода работы AutoML простым языком (агент)

Помимо аналитических навыков для качественного анализа требуется понимание принципов работы AutoML - инъекция экспертных знаний

Self-RAG

Высококачественное извлечение знаний через ChromaDB + LangGraph из документации FEDOT

RAG + self-reflection



Плюсы:

- Уменьшение контекста за счет выбора только документов, которые реально релевантны для ответа на вопрос.
- Проверка на галлюцинации ответа от LLM - вся информация в ответе базируется на информации из предоставленных документов.
- Проверка, отвечает ли наш ответ на поставленный вопрос.
- Повторная попытка с измененным вопросом, которая должна вытащить смежные документы из векторной бд.

Минусы:

- Увеличивается количество запросов, увеличивается затрачиваемое время.



Self-RAG - пример вывода

```
answer = app.invoke({"question": "Fedot automl regression example?"})
```

```
display(Markdown(answer["answer"]))
```

For a regression example using FEDOT, you can refer to the following resources:

1. **Regression using API:** [API Regression Example](#)
2. **Regression with tuning:** [Regression with Tuning Example](#)

Here is a basic code snippet demonstrating how to set up a regression model in FEDOT:

```
model = Fedot(problem='regression', timeout=5, preset='best_quality', n_jobs=-1)
model.fit(features=x_train, target=y_train)
prediction = model.predict(features=x_test)
metrics = model.get_metrics(target=y_test)
```

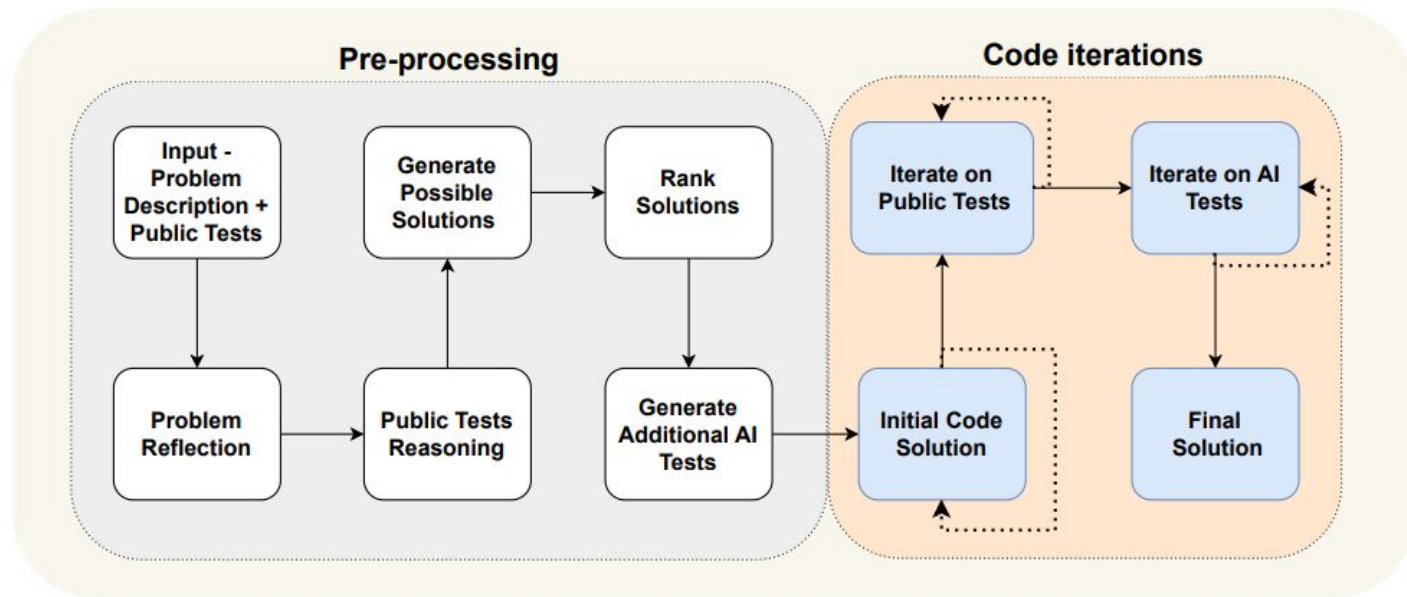
This code initializes a regression model, fits it to the training data, makes predictions on the test data, and retrieves the performance metrics of the model [2].

После “официальной” части - живой чат по задаче

- Структурирование ответов через Markdown
- Отображение актуального процесса, каков текущий этап (astream_events в LangChain)
- Вся информация должна быть приведена к виду, понятному не-экспертному виду
- Проверка информации через саморефлексию

Перспектива - автокодинг для анализа

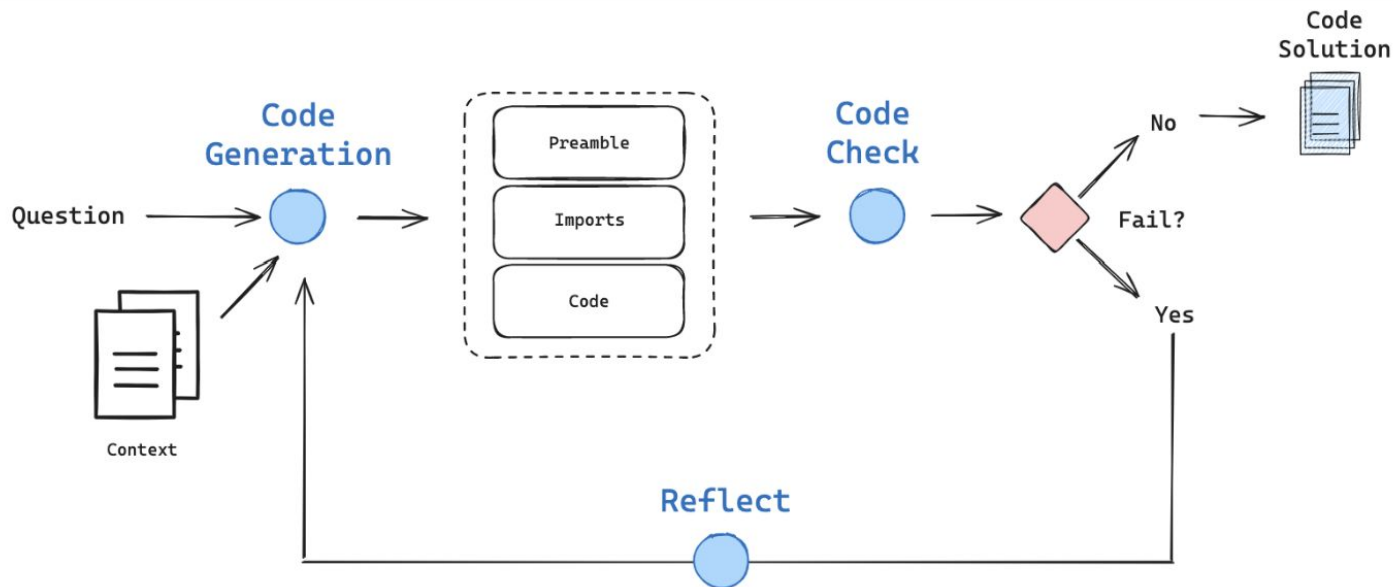
Вариант реализации агента, способного к написанию кода с помощью matplotlib и pandas - (AlphaCodium)



Ridnik, Tal, Dedy Kredo, and Itamar Friedman. "Code generation with alphacodium: From prompt engineering to flow engineering." *arXiv preprint arXiv:2401.08500* (2024).

Перспектива - автокодинг для анализа

Более простой вариант через LangGraph





ІІТМО

Спасибо за внимание!