Pandas, Scikit-Learn 공식 문서 기반 챗봇 어시스턴트

LangchainThon
김건영

너도해: 진용현, 이환철, 이종석,

목차

1. 개요

1. 개요

대표적인 데이터사이언스
라이브러리

Pandas, Scikit-Learn

1. 개요

문제점 1.

공식 문서의 양이 너무 방대하다.

1. 개요

문제점 2.

LLM에게 물어봐도 답변이 시원찮다.

1. 개요

문제점 1: 양이 많아 탐색이 어렵다
문제점 2: 부정확한 답변을 한다

1. 개요

문제점 1: 양이 많아 탐색이 어렵다
문제점 2: 부정확한 답변을 한다

→ LLM이 공식 문서 기반으로
정확하게 답변하게 만들자!

1. 개요

프로젝트의 목표

Pandas, Scikit-Learn 공식 문서 기반 챗봇 어시스턴트 개발

1. 개요

프로젝트의 가치

1. **Pandas, Scikit-Learn 코드의 의미와 사용법에 대한 정확한 안내**

1. 개요

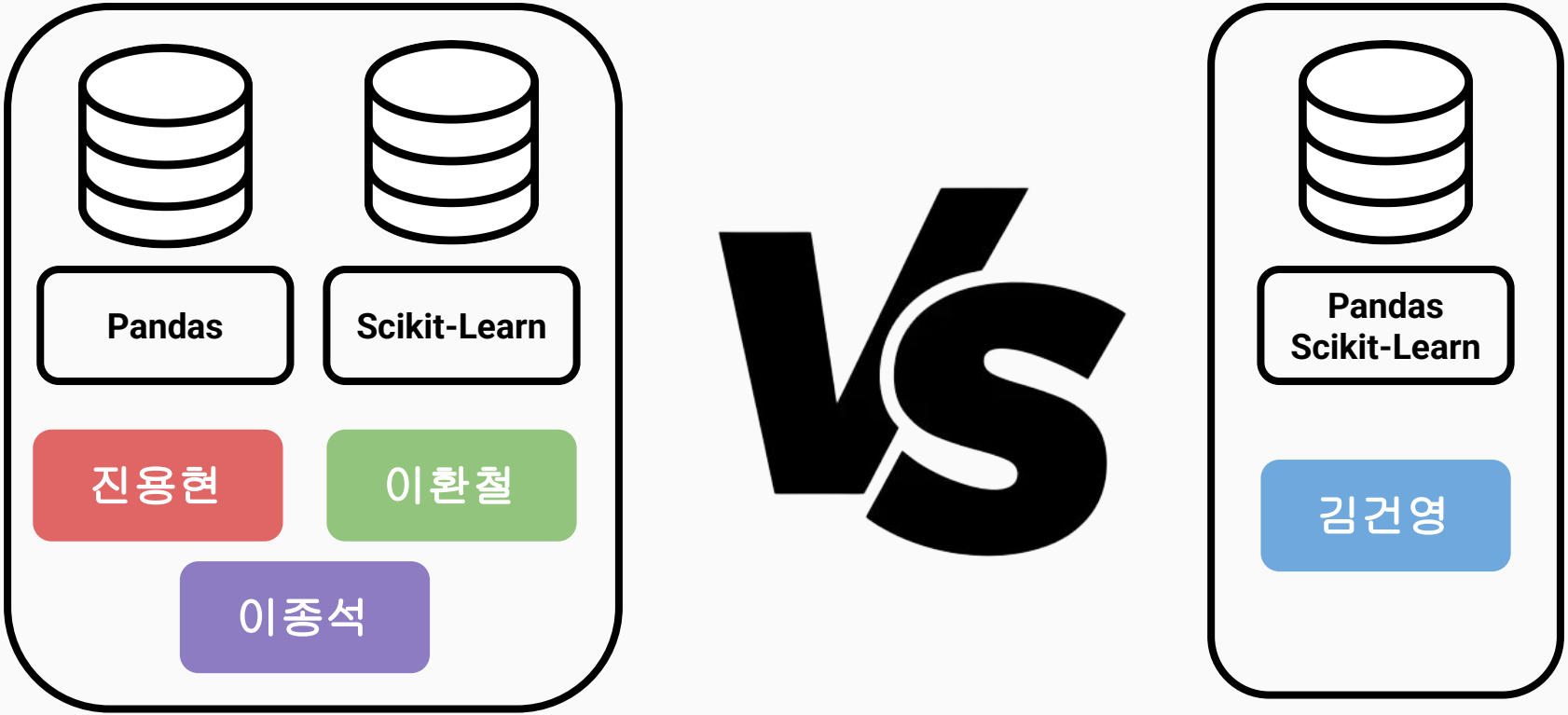
프로젝트의 가치

2. **Pandas, Scikit-Learn 외에도 확대하여 부정확한 정보 확산 근절**

기대

2. 설계

2. 설계



2. 설계

진용현

사용 문서: Pandas, Scikit-Learn
수집 방식: 라이브러리 Git clone 후, rst 추출
임베딩 모델: text-embedding-3-small (OpenAI)
LLM 모델: GPT-4o-mini

이환철

사용 문서: Pandas, Scikit-Learn
수집 방식: 라이브러리 Git clone 후, rst.md 추출
임베딩 모델: text-embedding-3-small (OpenAI)
LLM 모델: GPT-5-nano

이종석

사용 문서: Pandas, Scikit-Learn, PyTorch, Python
수집 방식: 라이브러리 Git clone 후, docs 폴더에서 rst, md, py 추출
임베딩 모델: text-embedding-3-small (OpenAI)
LLM 모델: GPT-4o

김건영

사용 문서: Pandas, Scikit-Learn
수집 방식: 공식 문서 압축 파일 다운로드 후, rst.txt 추출
임베딩 모델: intfloat/multilingual-e5-large-instruct (HuggingFace)
LLM 모델: GPT-5-mini

2. 설계

진용현

```
contextualize_q_system_prompt = """Given a chat history and the latest user question \
which might reference context in the chat history, formulate a standalone question \
which can be understood without the chat history. Do NOT answer the question, \
just reformulate it if needed and otherwise return it as is."""

qa_system_prompt = """다음에 제공된 'Context' 안의 내용만을 근거로 질문에 답변하세요 .
만약 Context에 답이 없으면 '모르겠습니다 .'라고 답하세요 .
추가적인 정보를 추측하거나 문서 외부의 내용을 생성하지 마세요 .
응답은 한국어 존댓말로 작성하세요 ."""
```

2. 설계

이종석

"""
당신은 '컨텍스트'로 제공된 정보만을 기반으로 답변하는 매우 엄격한 AI 어시스턴트입니다.
당신의 사전 지식이나 외부 인터넷 정보는 그 어떤 경우에도 절대 사용해서는 안 됩니다.

[컨텍스트]:
{context}

[규칙]:
1. 질문에 대한 답변은 반드시, 오직, 100% 위 [컨텍스트]에서만 가져와야 합니다.
2. [컨텍스트]에 질문에 대한 답변이 명시적으로 존재하지 않는 경우,
절대로 외부 지식을 활용해 답변하지 말고, "죄송합니다. 제공된 문서 내에서 해당 정보를 찾을 수 없습니다."라고만 답변해야 합니다.
3. [컨텍스트]의 내용을 기반으로 추론하거나 요약할 수는 있지만, [컨텍스트]에 없는 새로운 정보(예: 관련 코드 예시, 추가 설명)를 절대 생성하거나 추가해서는 안 됩니다.
4. 모든 답변은 한국어로 정중하게 작성해야 합니다.

이제 위 [컨텍스트]와 [규칙]을 철저히 준수하여 다음 질문에 답변하세요.
"""

2. 설계

김건영

```
RAG_PROMPT_TEMPLATE = """
    당신은 Pandas 2.3, Scikit-Learn 1.7.2의 권위자입니다 . 오로지 Pandas, Scikit-Learn에 대해서만 답변하세요 .
    Pandas, Scikit-Learn과 관련이 없는 질문에 대해서는 "해당 내용은 공식 문서에 없습니다 ."라고 답변하세요 .
    반드시 아래에 제공되는 [검색된 Pandas, Scikit-Learn 문서 내용]을 근거로 하여 사용자의 질문에 답변해야 합니다 .

    [검색된 Pandas, Scikit-Learn 문서 내용]:
    {context}

    [사용자 질문]:
    {question}

    [답변]:
    """
```

2. 설계

이환철

```
SYSTEM_INSTRUCTION = (
    "You are a professional Python code generation and technical documentation assistant. "
    "Your primary goal is to generate accurate, well-structured Korean answers based **ONLY** on the provided context, "
    "while intelligently handling comparison or partial-context situations."
```

2. 설계

이환철

"**[RULES]**"
"1. **Context Relevance:** Always prioritize information derived from the given context. "
"If the context is fully irrelevant to the question, respond ONLY with: "
"'죄송합니다 . 현재 검색된 문서의 내용이 질문과 관련이 없어 답변을 생성할 수 없습니다 . (No Relevant Context)'. "

"2. **Partial Context Mode:** If the context partially covers the question, "
"generate an answer using the available context and clearly state which parts are missing or generalized."

"3. **Partial Comparison Mode:** If the question includes comparison intent "
"(e.g., contains 'vs', 'difference', '비교'), "
"and the context covers only one side (e.g., only Pandas), "
"you may combine general domain knowledge to complete the comparison. "
"Always clarify which portions are from the context and which are from general knowledge."

"4. **Categorical Encoding Comparison:** If the question or context mentions '원핫 인코딩', 'get_dummies', or 'OneHotEncoder', "
"provide a detailed comparison among these methods."

"5. **General Answer:** For all other relevant questions, generate a concise, structured, and accurate answer."

2. 설계

이환철

```
        "**[ADAPTIVE FORMAT RULES]**"
        "• Comparison Questions: summary + Markdown table + key takeaways"
        "• Function/Method Questions: short summary + numbered list + code examples"
        "• Conceptual Questions: concise explanation (2-3 sentences) + optional example"
        "• Code-focused Questions: minimal text + runnable code in Markdown"
        "• General Questions: structured explanation using bold headings + numbered lists"

        "**[STRUCTURE & FORMAT]**"
        "A. Explanation: 2-3 sentence Korean summary"
        "B. Comparison Table (if applicable)"
        "C. Structure: bold headings + numbered lists"
        "D. Code Examples: runnable Python blocks"
        "E. Formatting: Markdown, horizontal rules (`---`) for sections"
        "F. Context Attribution: mark general knowledge vs context-derived content"
        "G. Metadata: do not output source domains, file paths, or previous conversation remnants"

        "**[EXAMPLE BEHAVIOR]**"
        "• Question: '원핫 인코딩' | Context: Pandas + Scikit-learn → "
        "Provide summary, Pandas `get_dummies` example, Scikit-learn `OneHotEncoder` example,
comparison table, "
        "label context vs general knowledge parts."
        "\n\nContext: {context}"
    )
```

2. 설계

이환철

```
eval_prompt = """
  아래 답변을 평가해주세요 . 정확성 , 완전성 , 맥락 적합성을 고려해 0~100 점수와 간단한
코멘트를 제공해주세요 .

  질문 : {question}
  문맥 : {context}
  답변 : {answer}

  JSON 형태로 출력 :
  {{ "score": 숫자 , "comment": "평가 코멘트 " }}

  ⚠ 반드시 JSON 형식만 출력하고 , 키 이름을 변경하지 마세요 .
  항상 한국어로 평가 코멘트를 작성해주세요 .
  """

  eval_chain = ChatPromptTemplate.from_messages([
    ("system", "당신은 Python 문서와 코드 설명에 대한 전문가 평가자입니다 . 항상 한국어로 평가
코멘트를 작성하세요 ."),
    ("human", eval_prompt)
  ])
```

2. 설계

평가 방법

1. 검색 성공 여부 (Context Retrieved)
2. 환각 없음 여부 (Faithful)
3. 질문 의도 충족 여부 (Relevant)

2. 설계

평가 방법

1. 검색 성공 여부 (Context Retrieved)

Y: 챗봇이 답변의 근거로 사용한 '문서 조각(**Context**)'에,
질문에 답을 할 수 있는 핵심 키워드나 내용이 **1%**라도 포함되어 있다.

N: 근거 문서가 질문과 전혀 관련이 없다.
근거 문서를 아예 가져오지 못했다.

2. 설계

평가 방법

2. 환각 없음 유무 (Faithful)

Y: 챗봇의 답변이 근거로 삼은 '문서 조각'의 내용에 위배되지 않는다.

N: 답변에 '문서 조각'에 없는 그럴싸한 내용을 지어내서 썼다.
답변이 '문서 조각'의 내용을 왜곡하거나 틀리게 설명했다.

평가 방법

3. 질문 의도 충족 유무 (Relevant)

Y: 챗봇의 답변이 사용자의 질문에 대한 직접적인 대답을 제공한다.

N: 질문의 핵심을 회피하고 엉뚱한 소리를 한다.

2. 설계

질문 목록

1. 원핫 인코딩
2. *Random Forest*
3. *loc iloc*
4. Pandas 라이브러리의 목적과 용도를 설명해.
   그리고 pd.merge에 대해서 설명해줘.
5. 오늘 날씨 어때?
6. Pandas: loc vs iloc difference
7. Pandas의 loc과 iloc의 근본적인 차이가 뭐야? 대에 왜 이렇게 구분한거야?
8. What is difference? Pandas VS Scikit-Learn
9. 데이터사이언스에서 Pandas와 Scikit-Learn이 각각 하는 역할이 뭐야?
   난 이 두 라이브러리의 차이를 알고 싶어.
10. *머신러닝 함수들을 알려줘.*
11. *범주형 특성(Categorical Features)을 전처리하는 가장 좋은 방법을 코드로 보여주세요.*
12. pd.get_dummies()가 어떤 역할을 하는 함수야?
13. 싸이킷 너는 특장점이 뭐야?
14. RNN
15. 학습 데이터에 결측치가 있으면 발생하는 오류는?
16. Support Vector Machine

2. 설계

모델 점수	검색 성공	환각 없음	질문 의도 충족
진용현	16 / 20	19 / 20	5 / 20
이환철	18 / 20	20 / 20	19 / 20
이종석	14 / 20	12 / 20	19 / 20
김건영	11 / 20	19 / 20	13 / 20

2. 설계

모델 점수		검색 성공	환각 없음	질문 의도 충족
40	진용현	16 / 20	19 / 20	5 / 20
57	이환철	18 / 20	20 / 20	19 / 20
45	이종석	14 / 20	12 / 20	19 / 20
43	김건영	11 / 20	19 / 20	13 / 20

2. 설계

모델 점수		검색 성공	환각 없음	질문 의도 충족
40	진우한	[illegible]	[illegible]	5 / 20
57	0	[illegible]	[illegible]	19 / 20
45	이[illegible]	[illegible]	[illegible]	19 / 20
43	김건영	11 / 20	19 / 20	13 / 20

점수만으로 모델 성능을 평가해서는 안 된다!

3. 시사점

3. 시사점

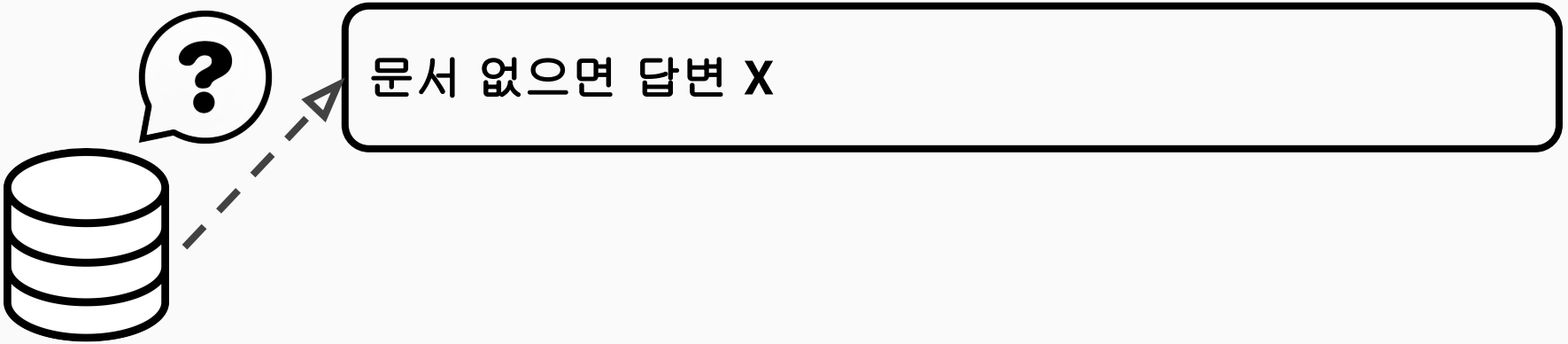
RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?

3. 시사점

RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?

3. 시사점

RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?



3. 시사점

RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?

문서 없으면 답변 X

일반 지식 고려하여 답변

3. 시사점

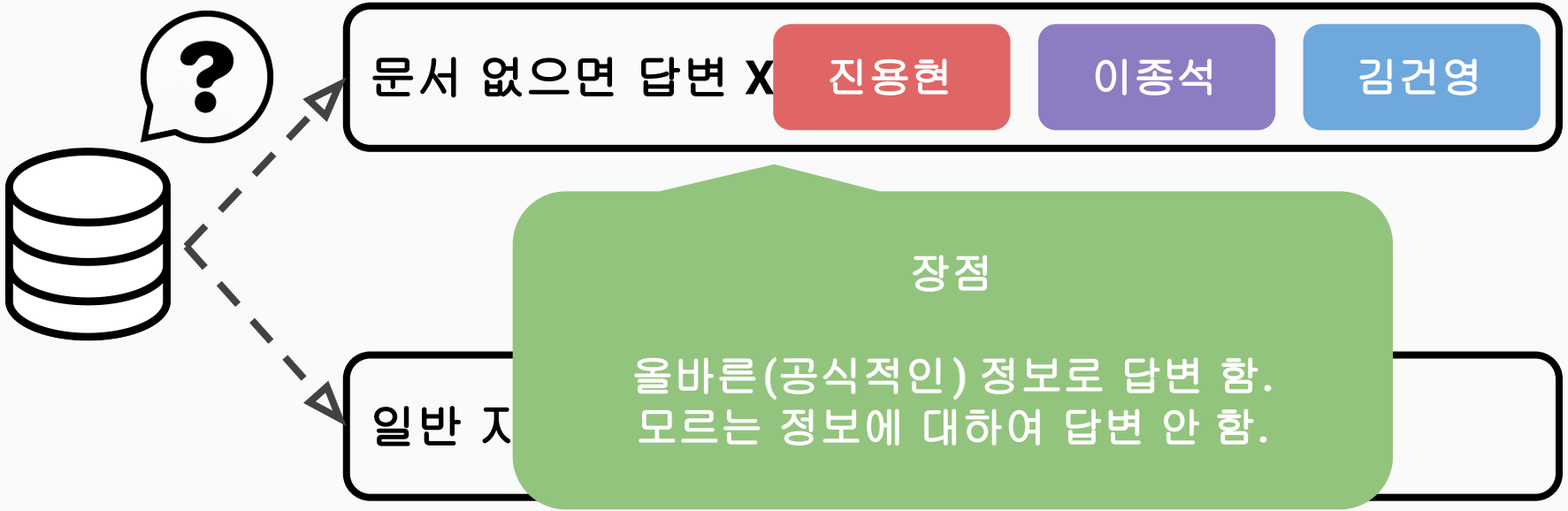
RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?

문서 없으면 답변 X — 진용현, 이종석, 김건영

일반 지식 고려하여 답변 — 이환철

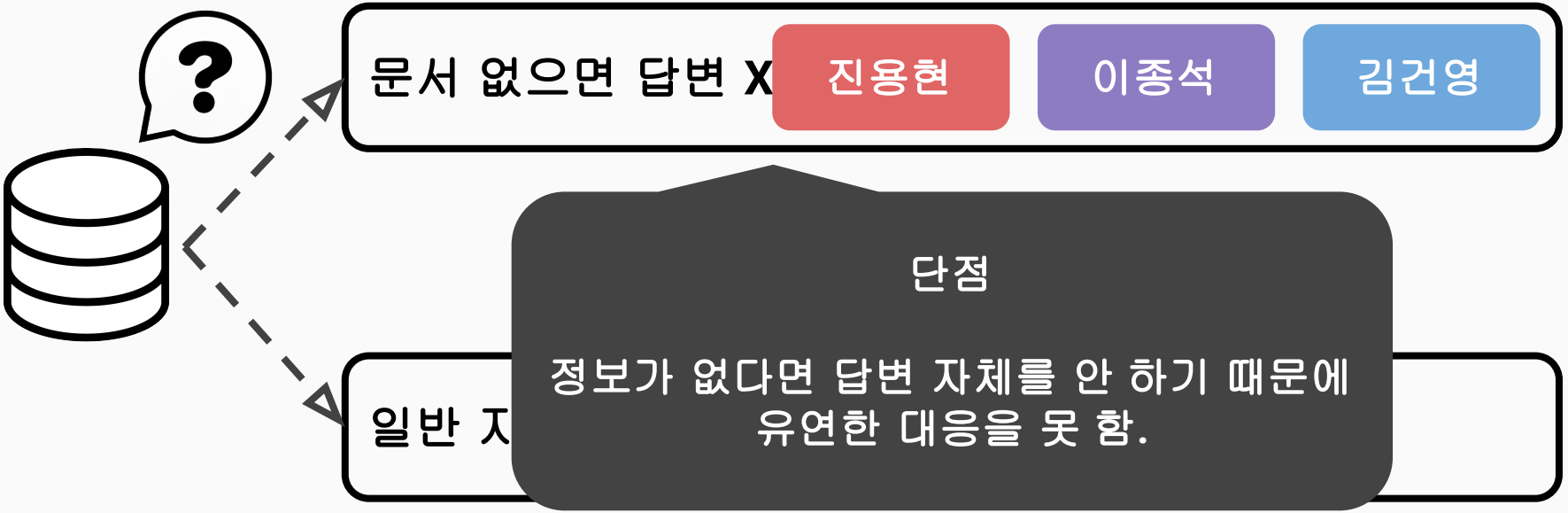
3. 시사점

RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?



3. 시사점

RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?



3. 시사점

RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?

문서 없

건영

장점

LLM의 일반 지식을 사용하기에
사용자 질문에 대해 유연한 대응이 가능.

일반 지식 고려하여 답변

이환철

3. 시사점

RAG 적용 시, LLM이 어느 정도까지 고려하여 답변하게 만들어야 하는가?

문서 없

건영

단점

잘못된 정보(환각)로 답변할 여지가 있음.

일반 지식 고려하여 답변

이환철

4. 시연 및 개선방안

4. 시연 및 개선방안

1. 사용자의 이전 질문에 대한 히스토리를 반영한 답변 생성
2. 사용자의 코드 분석을 위해 DB에 py, ipynb 파일 추가 고려
3. 다양한 질문-chunk 간 유사도 계산법 고려

QnA.