WhyFi : 금융 용어 알리미

Euron 7기 Research팀 - 이서영, 김경민, 장서연, 김나경

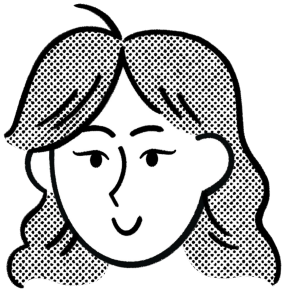
github: https://github.com/Xeoyeon/whyfi.git

EWHA EURON

목차

EWHA EURON

Our Team



이서영
통계학과 20

김경민
통계학과 21

장서연
컴퓨터공학과 21

김나경
컴퓨터공학과 22

01 주제

📌 어려운 금융 용어... 쉽게 설명해 줄 수 있는 서비스가 없을까?

📌 실시간으로 바뀌는 주식과 경제 흐름... 빠르게 파악할 수 있는 방법이 없을까?

-> RAG를 활용하자

기존 LLM	RAG
최신 정보 부족	실시간 검색을 통해 최신 데이터 제공 가능 (WhyFi : 최신 뉴스와 키워드 함께 제공)
금융 도메인에 특화되지 않음	금융 교육 자료 활용
잘못된 정보를 제공할 가능성이 있음 (hallucination)	신뢰할 수 있는 출처를 기반으로 응답 생성

01 주제

RAG(Retrieval-Augmented Generation) 기반
금융 용어 설명 서비스 : WhyFi

02 기술

📌RAG(Retrieval-Augmented Generation)

- 검색으로 입력 데이터를 증강 → 답변 생성
- LLM(대규모 언어 모델)이 신뢰할 수 있고 최신화된 외부 지식을 활용하도록 하는 프로세스

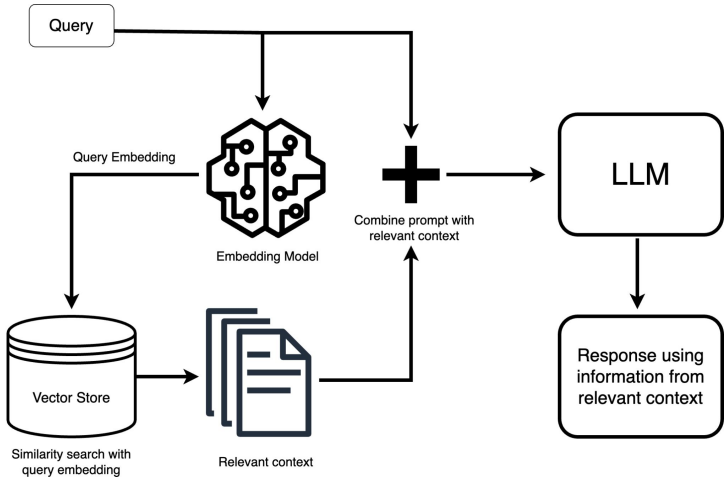
  🔍 외부 지식: 학습 시 사용되지 않은 데이터

<과정>

1. 외부 지식(책, 백서 등) → 임베딩 모델로 벡터화 → 벡터 DB에 저장
2. 사용자 질문 벡터화

   → 벡터 DB에 저장된 외부 지식 벡터와 유사도 검색(코사인 유사도)
3. 사용자 질문과 벡터 데이터베이스에서 불러온

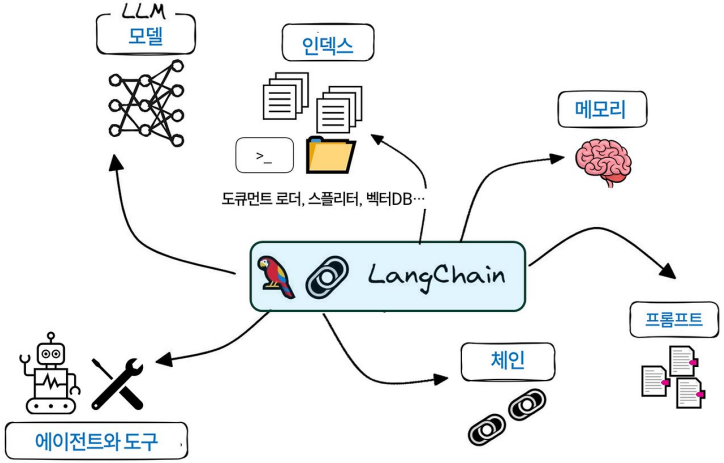
   외부 지식(context)을 프롬프트에 넣어 LLM이 답변 생성



02 기술

📌Langchain

- LLM을 효과적으로 활용하기 위한 프레임워크
  - 입력: 문서 전처리, 벡터 DB, 프롬프트, 메모리 등
  - 모델: 다양한 모델을 동일한 프로세스로 사용할 수 있도록 함
  - 출력: 출력 파서, 모니터링(langsmith)
  - 연결: chain, agent
- 기본 chain 문법
  - 입력 형식(prompt), llm, 출력 형식(output_parser)을 연결
  - 이전 단계의 출력값이 다음 단계의 입력으로 사용됨




```
prompt | llm | output_parser
```

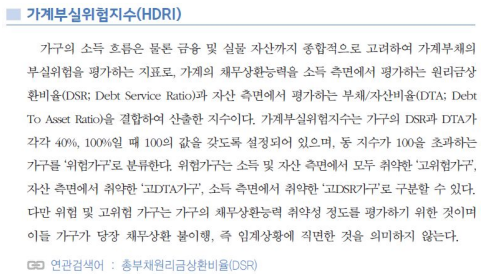
03 구현 - 데이터 준비

📌사용 데이터

- 한국은행 <경제금융용어 700선> pdf : [금융 용어 / 개념 설명 / 연관 단어]의 청크로 이루어진 자료
- 국세청 <2024 한권으로 OK 주식과 세금> pdf

📌pymupdf

- pdf를 LLM에 넣을 수 있는 markdown 형식으로 변환하기 위해 전처리 진행
- Tesseract, google vision, pymupdf, LlamaParse의 4가지 방법 중 가장 정확한 파싱을 수행한 pymupdf로 결정



가계부실위험지수(HDRI)

가구의 소득 흐름은 물론 금융 및 실물 자산까지 종합적으로 고려하여 가계부채의 부실위험을 평가하는 지표로, 가계의 채무상환능력을 소득 측면에서 평가하는 원리금상환비율(DSR; Debt Service Ratio)과 자산 측면에서 평가하는 부채/자산비율(DTA; Debt To Asset Ratio)을 결합하여 산출한 지수이다. 가계부실위험지수는 가구의 DSR과 DTA가 각각 40%, 100%일 때 100의 값을 갖도록 설정되어 있으며, 동 지수가 100을 초과하는 가구를 '위험가구'로 분류한다. 위험가구는 소득 및 자산 측면에서 모두 취약한 '고위험가구', 자산 측면에서 취약한 '고DTA가구', 소득 측면에서 취약한 '고DSR가구'로 구분할 수 있다. 다만 위험 및 고위험 가구는 가구의 채무상환능력 취약성 정도를 평가하기 위한 것이며 이들 가구가 당장 채무상환 불이행, 즉 임계상황에 직면한 것을 의미하지 않는다.

연관검색어 : 총부채원리금상환비율(DSR)

```
## 가계부실위험지수(HDRI)

가구의 소득 흐름은 물론 금융 및 실물 자산까지 종합적으로 고려하여 가계부채의 부실위험을 평가하는 지표로,
가계의 채무상환능력을 소득 측면에서 평가하는 원리금상환비율(DSR; Debt Service Ratio)과 자산 측면에서 평가하는 부채/자산비율(DTA; Debt To Asset Ratio)
가계부실위험지수는 가구의 DSR과 DTA가 각각 40%, 100%일 때 100의 값을 갖도록 설정되어 있으며, 동 지수가 100을 초과하는 가구를 '위험가구'로 분류한다.
위험가구는 소득 및 자산 측면에서 모두 취약한 '고위험가구', 자산 측면에서 취약한 '고DTA가구', 소득 측면에서 취약한 '고DSR가구'로 구분할 수 있다.
다만 위험 및 고위험 가구는 가구의 채무상환능력 취약성 정도를 평가하기 위한 것이며 이들 가구가 당장 채무상환 불이행, 즉 임계상황에 직면한 것을 의미하지 않

연관검색어 : 총부채원리금상환비율(DSR)
```

```python
from langchain.document_loaders import PyMuPDFLoader

loader = PyMuPDFLoader("./[2024 한권으로 OK 주식과 세금].pdf")
docs = loader.load()
preprocess(docs=docs, collection_name="stock_book")
```

→ langchain의 PyMuPDFLoader 활용

EWHA EURON

03 구현 - 데이터 준비

📌데이터 정제 - 경제용어700선

가계수지	가정에서 일정 기간의 수입(명목소득)과 지출을 비교해서 남았는지 모자랐는지를 표시한 것을 가계수지(household's total income and expenditure)라 한다. 가계수지가 ----- 흑자를 냈다면 그 가정은 벌어들인 수입 일부만을 사용했다는 것을 의미하며, 적자를 냈다면 수입 외에 빚을 추가로 얻어 사용한 것이라고 보아야 한다. 우리나라는 통계청에 서 가계의 수입과 지출을 조사하여 국민의 소득수준 및 생활실태를 파악하기 위해 표본으로 선정된 가계에 가계부를 나누어 주고 한 달간의 소득과 지출을 기록하도록 한 다음 이를 토대로 가계수지 통계를 작성하여 발표하고 있다. 가계부의 소득항목에는 근로소득·사업소득·재산소득·이전소득 항목이 있고, 비용항목에는 식료품비·주거비· 수도광열비·보건의료비·교육비 항목이 있다. 연관검색어 : 경상수지, 재정수지

가계수지	가정에서 일정 기간의 수입(명목소득)과 지출을 비교해서 남았는지 모자랐는지를 표시한 것을 가계수지(household's total income and expenditure)라 한다. 가계수지가 흑자를 냈다면 그 가정은 벌어들인 수입 일부만을 사용했다는 것을 의미하며, 적자를 냈다면 수입 외에 빚을 추가로 얻어 사용한 것이라고 보아야 한다. 우리나라는 통계청에 서 가계의 수입과 지출을 조사하여 국민의 소득수준 및 생활실태를 파악하기 위해 표본으로 선정된 가계에 가계부를 나누어 주고 한 달간의 소득과 지출을 기록하도록 한 다음 이를 토대로 가계수지 통계를 작성하여 발표하고 있다. 가계부의 소득항목에는 근로소득·사업소득·재산소득·이전소득 항목이 있고, 비용항목에는 식료품비·주거비· 수도광열비·보건의료비·교육비 항목이 있다. 연관검색어 : 경상수지, 재정수지

```python
# clean text
def clean_text(text):
    text = re.sub(r"\n+", " ", text)  # 연속된 줄바꿈은 공백으로 대체
    text = re.sub(r"-{2,}"," ", text)  # ----- 제거
    text = re.sub(r"\s{2,}", " ", text)  # 중복된 공백 제거
    text = text.strip()  # 양쪽 공백 제거
    return text

df['Content'] = df['Content'].apply(clean_text)
df.to_csv("cleaned_word_dict.csv", index=False, encoding="utf-8-sig")
```

- 불필요한 줄바꿈과 특수문자 제거 → cleaned_dict.csv 생성
- CSVLoader를 통해 데이터 로드

```python
from langchain_community.document_loaders.csv_loader import CSVLoader

loader = CSVLoader(file_path="./cleaned_word_dict.csv")
docs = loader.load()
preprocess(docs=docs, collection_name="words700")
```

EWHA EURON

03 구현 - 임베딩

📌문서 단위를 기계가 이해할 수 있는 수치적 형태로 변환하는 과정

1) 사용자 질문의 의도와 문서의 의미적 맥락을 이해하고 검색 향상을 위해 문서 임베딩 모델 사용

- Multi-Functionality, Multi-Linguality, Multi-Granularity 세 측면에서 모두 뛰어난 성능을 보인다고 알려진 BGE-M3 사용
- 그 중 한국어로 파인튜닝한 BGE-m3-ko로 결정

시장조사기관 IDC는 AI 소프트웨어 시장이 2022년 640억 달러에서 2027년 2,510억 달러로 연평균 성장률 31.4%를 기록하며 급성장할 것으로 예상 1

AI 소프트웨어 시장은 AI 플랫폼, AI 애플리케이션, AI 시스템 인프라 소프트웨어(SIS), AI 애플리케이션 개발·배포(AI AD&D) 소프트웨어를 포괄

협업, 콘텐츠 관리, 전사적 자원관리(ERM), 공급망 관리, 생산 및 운영, 엔지니어링, 고객관계관리(CRM)를 포함하는 AI 애플리케이션은 AI 소프트웨어의 최대 시장으로 2023년 전체 매출의 약 3분의 1을 차지하며 2027년까지 21.1%의 연평균 성장률을 기록할 전망 2

AI 비서를 포함한 AI 모델과 애플리케이션의 개발을 뒷받침하는 AI 플랫폼은 두 번째로 시장 규모가 큰 분야로, 2027년까지 35.8%의 연평균 성장률이 예상됨

분석, 비즈니스 인텔리전스, 데이터 관리와 통합을 포함하는 AI SIS는 기존 소프트웨어 시스템과 통합되어 방대한 데이터를 활용한 의사결정과 운영 최적화를 지원하며, 현재 매출 규모는 비교적 작지만 5년간 연평균 성장률은 32.6%로 시장 전체를 웃돌 전망 3

애플리케이션 개발, 소프트웨어 품질과 수명주기 관리 소프트웨어, 애플리케이션 플랫폼을 포함하는 AI AD&D는 향후 5년간 카테고리 중 가장 높은 38.7%의 연평균 성장률이 예상됨

- 1번 단락: [0.1, 0.5, 0.9, ... , 0.1, 0.2]
- 2번 단락: [0.7, 0.1, 0.3, ... , 0.5, 0.6]
- 3번 단락: [0.9, 0.4, 0.5, ... , 0.4, 0.3]

질문: "시장조사기관 IDC 가 예측한 AI 소프트웨어 시장의 연평균 성장률은 어떻게 되나요?"

- [0.1, 0.5, 0.9, ..., 0.2, 0.4]

유사도 계산 예시

- **1번: 80%** -> 선택!
- 2번: 30%
- 3번: 25%

03 구현 - 벡터 스토어 저장

📌생성된 임베딩 벡터들을 효율적으로 저장하고 관리하는 과정

2) 벡터 스토어 생성

- 대표적인 chromaDB와 FAISS 모두 사용해본 결과, 성능과 속도 면에서 큰 차이 없음을 확인
  → 따라서 모든 팀원이 수행해본 Chroma 사용

```
from langchain_chroma import Chroma
from langchain_huggingface.embeddings import HuggingFaceEmbeddings
```

```
class ChromaDB:
    def __init__(self, collection_name):
        self.embedding_model = HuggingFaceEmbeddings(model_name="BAAI/bge-m3")
        self.vectorstore = Chroma(
            collection_name=collection_name, persist_directory="./chroma_index", embedding_function=self.embedding_model
        )
```

```
db = ChromaDB("words700")
# db = ChromaDB("stock_book")
```

EWHA EURON

03 구현 - 검색기

📌사용자의 질문과 관련 문서를 검색하여 알맞은 답변을 생성하는 과정

3) Dense Retriever
- 키워드가 완벽히 일치하지 않더라도 의미적으로 가장 관련성이 높은 문서를 검색 (k : 반환할 문서 수)

4) 프롬프트 설계
- 서비스 목적에 맞게 용어 정의는 풀어서 쉬운 단어로 제공하도록 설계
- 활용 예시와 관련 단어를 추가하여 이해를 도움
- 관련 단어는 모델이 검색 과정에서 거치는 단어들이 반영되도록

5) LLM 튜닝 및 검색 체인 생성
- 안정된 성능과 무료 LLM인 Gemini 1.5 사용 → 자연스러운 답변 생성
- 체인을 사용하여 질문에 대한 답변을 출력하도록 설계

? 신용등급

정의 :

신용등급은 마치 사람의 성적표 같은 거야.
은행이나 카드사 같은 곳에서 당신이 돈을 잘 갚을 사람인지, 아니면 돈을 빌려주면 떼일 가능성이 높은 사람
점수가 높으면 믿을 만한 사람으로 인정받아서 돈을 빌리기 쉽고, 이자도 낮게 적용받을 수 있어.
반대로 점수가 낮으면 돈을 빌리기 어렵거나 이자가 높아질 수 있지.

```
class RAGAgent:
    def __init__(self, prompt_template):
③       self.word_retriever = word_collection.as_retriever(search_kwargs={"k": 3})
        self.book_retriever = book_collection.as_retriever(search_kwargs={"k": 2})

④   prompt = PromptTemplate(
        input_variables=["word_context", "book_context", "term"],
        template=prompt_template
    )

⑤   llm = ChatGoogleGenerativeAI(model="gemini-1.5-flash")

    # 병렬 검색을 위한 retriever 설정
    retriever_chain = {
        "word_context": self.word_retriever | self.format_retriever_output,
        "book_context": self.book_retriever | self.format_retriever_output,
    }

    self.chain = (
        {"term": RunnablePassthrough()}
        | retriever_chain
        | prompt
        | llm
        | StrOutputParser()
    )
```

03 구현 - 추가 기능

https://eiec.kdi.re.kr/bigdata/issueTrend.do

📌서비스의 사용자 경험을 향상시키기 위해 최신 뉴스 및 금융 키워드 순위 추가

1) 네이버 뉴스 API를 활용하여 검색 단어와 관련된 최신 뉴스 정보 제공하며 서비스 신뢰성 및 편의성 높임
2) KDI 경제교육·정보센터에서 제공하는 Top 10 경제 키워드 트렌드(검색일 기준 이전 달)를 정기적으로 크롤링하여 json 파일 생성 후 사용. 버튼 구성을 통해 클릭 시 해당 키워드로 바로 검색 가능

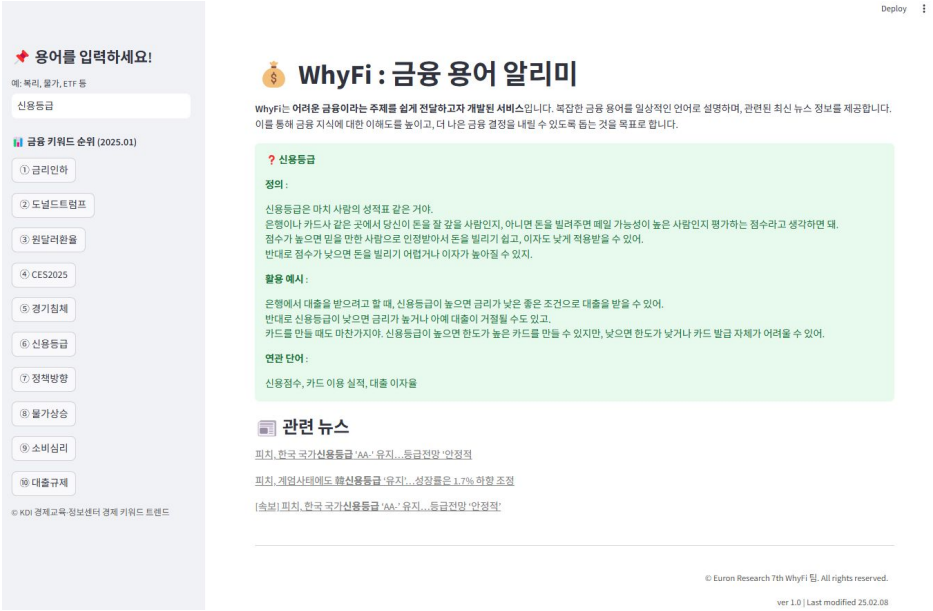
```
if response.status_code == 200:
    news_items = response.json().get("items", [])
    if not news_items:
        return "관련 뉴스를 찾을 수 없습니다."
    random_news = random.sample(news_items, min(3, len(news_items)))
    return [{"title": item['title'], "link": item['link']} for item in random_news]

return f"API 요청 실패: {response.status_code}"
```

04 최종 결과물

📌Streamlit

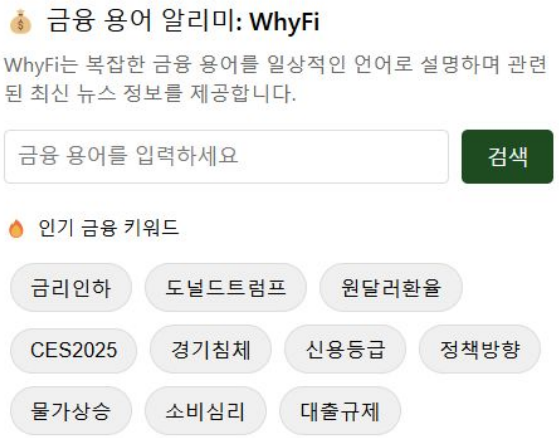
- 프로젝트 기획 의도에 맞게 Streamlit을 활용하여 유저 친화적인 UI 구현 및 로컬 배포
- QR 코드를 통해 **금융 용어 알리미** 서비스를 체험해보실 수 있습니다!
  (많이 사용해주세요❤)



04 최종 결과물

📌Chrome-Extension

- 웹스토어 등록을 위해서는 개발자 등록 비용과 심사 통과 과정을 거쳐야 함 → 배포 어려움
- chrome://extensions/ 에서 개발자 모드 실행 후 chrome-extension 폴더 로드하여 실행가능



05 회고 및 마무리

계속 구상하고 있던 아이디어를 짧은 시간에 완성도 높은 결과물로 만들어낼 수 있었던 너무 좋은 경험이었습니다! 모두들 부족하거나 개선할 부분을 계속 생각하고 실행하는 모습이 멋졌어요:)

프로젝트로 RAG를 구현해보는 것뿐만 아니라 실제로 활용할 수 있게 streamlit과 chrome-extension을 사용해서 눈에 보이는 결과물을 만들어낸 것이 좋았습니다! 좋은 팀원분들한테 많이 배울 수 있었던 프로젝트였습니다.

논문 세미나에서 배운 LLM과 RAG를 직접 활용해볼 수 있어서 유익했습니다. Langchain, Chrome-extension 제작 등 정말 많은 것을 배우고 경험할 수 있었던 프로젝트였습니다!

Langchain의 다양한 기능을 실험해볼 수 있는 기회였습니다! 확장해보고 싶은 기능 아이디어가 많은데 다 구현해보지 못한 점은 아쉽네요ㅎㅎ 정말 순조롭게 진행된 프로젝트여서 즐겁게 참여했습니다 :D