Agentic AI를 활용한 지역아동센터 수요 예측 서비스

2026.01.08

김진혁

목차

1. 프로젝트 기획서 – 기획의도(1/2)

여성농업인신문 종합 농촌여성 농민단체소식

Since2005

홈 > 농촌여성

아이돌봄서비스 수요 느는데 인력은 태부족

김수현 기자 | 입력 2025.11.07 14:01 | 댓글 0

전국 아이돌봄센터 232개소,
돌보미수 2만9천여명
지난해 13만9천명 신청
평균 대기 기간 32.8일

사회 교육

초등 학부모 50% "수업 전후 돌봄 필요"...4년새 20%p 급증

2023년 범정부 온종일 돌봄 수요조사

김민제 기자

수정 2023-03-06 13:35 등록 2023-03-06 13:35

저출산 + 가정 형태의 다양화로

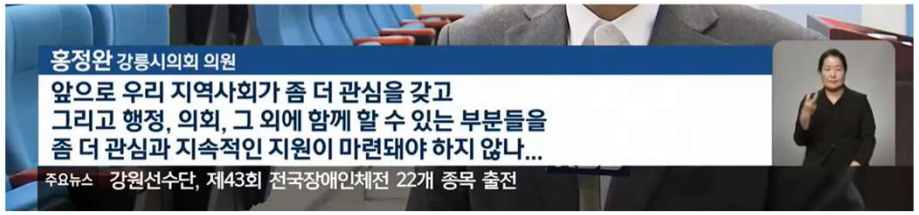
방과후 돌봄 수요 확대

가정에서 아이의 돌봄을 감당 어려움

방과 후 / 야간 시간대 돌봄 수요 증가

"지역아동센터 이용자수를 예측 필요"

1. 프로젝트 기획서 – 기획의도(2/2)



돌봄 수요는 전국적으로 늘어나고 있지만

지원은 지역마다 제각각

지역별 돌봄 수요를 분석를 위해

구체적 데이터 필요

"자치구별 이용자 수를 예측해

정책 의사결정을 지원"

1. 프로젝트 기획서 – 개발목표(1/3)

이용자 수 예측

- Target : 자치구별 지역아동센터 이용자 수
- Feature :
  - 한부모 가정 수, 학원수, 아동인구
  - GRDP (지역 내 총 생산) 등 지역 지표
- 미래 이용자 수를 예측해 향후 수요 변화를 추정
  - Train : 2015 ~ 2020년 데이터
  - Test : 2021 ~ 2022년 데이터
- 데이터 출처 : KOSIS 국가통계포털, 서울 열린데이터 광장

수요 분석

- 예측 결과를 자치구 / 연도별로 비교해
  수요가 특히 높거나 급증하는 지역, 시점을 도출
- 통계값과 예측값을 함께 보면서
  현재 수요와 미래 수요의 격차가 큰 구를 식별
- 돌봄 수요 상위 구간(예: 상위 25%)을 묶어
  우선적으로 살펴봐야 할 돌봄 상위 지역을 도출

1. 프로젝트 기획서 – 개발목표(2/3)

개발 내용 및 목적

- 자치구별 아동센터 이용 수요를 정밀하게 산출하는 회귀 모델을 구축
- 하이퍼파라미터 최적화를 통한 성능 개선 :
  - XGBoost 모델의 예측력 극대화를 위해 RandomizedSearchCV를 활용한 교차 검증
- Hyper parameter :
  - 학습 횟수(n_estimators)
  - 트리 깊이(max_depth)
  - 학습률(learning_rate) 등 핵심 파라미터를 최적화하여 모델의 신뢰도를 확보

예측 결과 조회 / 비교 UI

- 자치구별 과거 통계 데이터 추이 분석을 위한 통합 데이터 대시보드 UI
- 지도 기반의 자치구 선택 시 예측 결과 그래프 시각화 및 세부 수치 제공
- 예측 수요와 인프라 대조를 위한 자치구별 지역아동센터 현황 및 정보 제공

1. 프로젝트 기획서 – 개발목표(3/3)

LLM 기반 기술

- Llama-3-8B 모델 기반의 맞춤형 파인튜닝을 통한 지역아동센터 특화 답변 생성
- HuggingFace Transformers 및 PEFT(LoRA) 기법을 활용한 저사양 환경 효율적 모델 최적화
- 모델 학습 단계별(Base, Checkpoint 50/100) 성능 비교 분석 및 추론
- 지역아동센터 데이터셋(위치, 이용료, 거리 등)을 챗봇을 통해 아동센터 정보 제공

챗봇 서비스 기능

- 사용자가 모델의 학습 단계별(Checkpoint) 답변 품질을 직접 선택하고 대조할 수 있는 비교 UI
- 지역아동센터의 위치, 이용료, 거리 정보를 기반으로 한 맞춤형 정보 탐색 및 비교 서비스
- 현재 위치 정보를 바탕으로 센터까지의 이동 거리 및 접근성을 계산하여 정보를 제공

2. 사용 기술 목록/경험 – 사용 기술 목록

Language

Python (3.11)

JavaScript, HTML5, CSS4

NumPy (1.26.4)

Pandas (2.3.3)

Bootstrap (4.6.2)

AI Framework

HuggingFace

Langchain

Infra

runpod

Docker

GIthub

DataBase

Oracle 11g

Sql Developer

FrameWork

Flask

2. 사용 기술 목록/경험 – 사용 기술 경험(1/3)

Flask(BackEnd)

- Flask 기반 라우팅 및 요청·응답 처리, API 설계
- Blueprint로 기능 모듈 분리
- Flask-WTF 입력 검증 및 폼 처리
- JSON 응답 + Jinja2 템플릿 렌더링 구성

Database / ORM

- Oracle SQL로 조회·집계 쿼리 작성
- cx_Oracle 및 SQLAlchemy로 DB 연동
- ORM CRUD 구현 및 모델 설계
- Alembic(Flask-Migrate)로 스키마 버전 관리

Machine Learning

- Scikit-learn/XGBoost 기반 수요 예측 모델 개발
- 원-핫 인코딩 + log1p로 분포 안정화 전처리 적용
- 다중선형회귀 vs XGBoost 비교 후 모델 선정
- 튜닝(RandomizedSearchCV) 및 중요도 기반 해석

2. 사용 기술 목록/경험 – 사용 기술 경험(2/3)

LLM & Fine -tuning

- Llama-3-8B 인스트럭션 튜닝:
  HuggingFace와 PEFT/LoRA를 활용하여 지역아동센터 특화 JSON 지시어 데이터셋을 학습시키는 효율적 파인튜닝 파이프라인 구축
- BitsAndBytes 4-bit 양자화:
  NF4 양자화 기술로 모델을 경량화하여 Runpod GPU 환경 내 추론 및 학습 속도 개선
- 체크포인트별 결과 비교:
  학습 단계별(50, 100 Step) 답변 정밀도를 분석하여 지시 이행 능력이 가장 우수한 모델 선정

HTML / CSS / JS

- 반응형 대화창 구성:
  HTML/CSS 기반의 직관적인 챗봇 인터페이스 및 모바일 최적화
- 비동기 데이터 연동:
  Fetch API를 이용한 AI 모델 응답 및 JSON 데이터 실시간 처리
- 예측 데이터 시각화:
  Chart.js를 연동하여 자치구별 통계 및 예측치를 그래프로 구현
- 동적 UI 렌더링:
  사용자의 질의에 따라 센터 정보 카드 및 상세 비교 화면을 가변적으로 출력

2. 사용 기술 목록/경험 – 사용 기술 경험(3/3)

Git

- GitHub를 통한 프로젝트 버전 관리
- 기능 단위 커밋 / 머지로 변경 이력 추적 및 코드 공유
- 프로젝트 결과 공유 및 피드백

Docker

- docker-compose up 으로 Flask앱 + Orcale DB 컨테이너 구성/ 실행
- 개발 환경을 컨테이너로 표준화하여 재현 가능한 실행 환경 구축
- 이미지/컨테이너 관리로 로컬 테스트 환경 운영

Cloud/Runpod

- RunPod GPU 인스턴스에서 Llama 3 모델 환경 구성
- 학습/추론 작업 분리 운영 및 모델 파일(체크포인트/어댑터) 관리
- Flask 서비스와 연동 가능한 추론 서버 구조(FastAPI) 테스트/운영

3. 개발 스케쥴

담당자	색상
김진혁	
김세준	
공통	

구분	담당자	파트	날짜		기간	10월	11월			
						5주차	1주차	2주차	3주차	4주차
프로젝트 기획	공동	아이디어 구상	2025-10-27	2025-10-27	1					
		화면 목록 구성	2025-10-28	2025-10-28	1					
		Information Architecture	2025-10-29	2025-10-30	2					
데이터 단계	공동	데이터 수집 (공공데이터)	2025-10-31	2025-11-04	5					
		EDA(통계 시각화, 관계분석)	2025-11-06	2025-11-07	2					
		모델 피처 확정	2025-11-08	2025-11-08	1					
	김진혁	지원센터 이용자수 데이터 정제	2025-11-05	2025-11-05	1					
	김세준	변수 데이터 정제	2025-11-05	2025-11-05	1					
모델링	김진혁	베이스라인 모델 구축	2025-11-10	2025-11-11	2					
	김세준	트리 기반 모델 개발	2025-11-11	2025-11-12	2					
	공동	모델 성능 검증 및 비교	2025-11-13	2025-11-13	1					
		파이프라인 구성	2025-11-14	2025-11-14	1					
		트리 기반 모델 최종 선정	2025-11-17	2025-11-17	1					
시스템 설계	김세준	Flask 구조 설계	2025-11-18	2025-11-19	2					
	김진혁	DB 테이블 설계	2025-11-18	2025-11-19	2					
	공동	API 설계	2025-11-19	2025-11-20	2					
UI/UX	공동	화면설계서 초안 작성	2025-11-20	2025-11-21	2					
		UI 시안	2025-11-24	2025-11-24	1					
	김세준	전체 화면 설계서	2025-11-25	2025-11-25	2					
	김진혁	플로우 다이어그램	2025-11-25	2025-11-26	2					
백엔드	김세준	통계 대시보드 API 개발	2025-11-27	2025-11-28	2					
		Q&A API	2025-11-27	2025-11-28	2					
	김진혁	계정 로그인 / 회원가입 API	2025-12-01	2025-12-01	1					
		지역아동센터 수요 예측 API 개발	2025-12-01	2025-12-01	1					

3. 개발 스케쥴

담당자	색상
김진혁	
김세준	
공통	

구분	담당자	파트	날짜		기간	12월				
						1주차	2주차	3주차	4주차	5주차
백엔드	김세준	통계 대시보드 API 개발	2025-11-27	2025-11-28	2					
		Q&A API	2025-11-27	2025-11-28	2					
	김진혁	계정 로그인 / 회원가입 API	2025-12-01	2025-12-01	1					
		지역아동센터 수요 예측 API 개발	2025-12-01	2025-12-01	1					
프론트엔드	김세준	메인 페이지 UI	2025-12-02	2025-12-05	4					
		Q&A 페이지	2025-12-02	2025-12-03	2					
	김진혁	대시보드 UI	2025-12-02	2025-12-03	2					
		예측 결과 페이지 UI	2025-12-04	2025-12-05	2					
		UI소개 페이지 UI	2025-12-04	2025-12-05	2					
로컬 서비스 테스트	공동	프론트-백엔드 연동 및 최종 테스트 (로컬)	2025-12-08	2025-12-09	2					
모델 탐색 및 선정	김진혁	ai 서비스 기획	2025-12-10	2025-12-10	1					
		기본 학습 실험(gpt2, bert 등)	2025-12-10	2025-12-11	2					
		베이스 모델 선정 및 UI prototype	2025-12-11	2025-12-12	2					
파인튜닝	김진혁	Llama3-8b 모델 파인튜닝	2025-12-15	2025-12-19	5					
		모델 평가 및 최적화 (경량화)	2025-12-15	2025-12-19	5					
		LLM 연동 API 개발	2025-12-17	2025-12-19	3					
서비스 통합 및 배포	김진혁	생성형 ai UI 개발	2025-12-22	2025-12-23	2					
		배포 시스템 구축	2025-12-24	2025-12-26	3					
		최종 성능 테스트	2025-12-29	2025-12-30	2					
		RunPod Serverless 배포	2025-12-29	2025-12-30	2					

4. 요구사항 분석서 및 명세서 – 요구사항 명세서(1/2)

Agentic AI를 활용한 지역아동센터 수요 예측 서비스

요구사항 명세서

RQ-ID	화면 명	요구사항 명	요구사항 상세
RQ-ID-0001	메인 페이지	로고	웹 페이지 왼쪽 상단 클릭 시, 첫 화면으로 이동할 수 있는 로고를 삽입
RQ-ID-0002	메인 페이지	상단 네비게이션 바	네비게이션 바 각 영역에 머신러닝/통계 대시보드/로그인/회원가입으로 구분되어 있고 클릭 시 해당 페이지로 이동
RQ-ID-0003	메인 페이지	웹페이지 하단	웹 페이지 하단에 위치하며 서비스 이용 약관/개인정보 처리방침/정보 등을 포함
RQ-ID-0004	메인 페이지	이미지 슬라이드(캐러셀)	메인 상단에 2개 이상의 캐러셀을 제공 / 캐러셀은 약 5초 간격으로 자동 전환되며, 좌우 화살표 및 인디케이터를 통해 수동으로 슬라이드를 전환 가능
RQ-ID-0005	메인 페이지	스크롤 스냅	메인 페이지 내 스크롤 시 섹션 단위로 자동 정렬되며 스크롤 이동이 부드럽게 처리
RQ-ID-0101	서비스 소개	메인 서비스 이동 버튼	주요 서비스 페이지로 빠르게 이동할 수 있는 버튼 제공, 클릭 시 해당 서비스 페이지로 이동
RQ-ID-0102	서비스 소개	데이터 기반 안내	서비스가 실제 통계 기반 머신러닝 모델로 동작함을 사용자에게 안내
RQ-ID-0201	통계 대시보드	연도·지역 필터	사용자는 연도 범위 및 자치구 선택을 통해 데이터를 조회
RQ-ID-0202	통계 대시보드	자치구 검색 자동완성 기능	자치구 이름 일부 입력 시 일치/유사 자치구를 자동완성 목록으로 제공
RQ-ID-0203	통계 대시보드	동적 데이터 갱신	필터 변경 시 새로고침 없이 값이 즉시 갱신
RQ-ID-0204	통계 대시보드	카드형 지표	이용자 수·시설 수 지표가 카드 UI로 구성
RQ-ID-0205	통계 대시보드	연도 범위 선택 제약 기능	종료 연도는 시작 연도와 같거나 이후 연도만 선택 가능
RQ-ID-0206	통계 대시보드	데이터 예외 처리	선택된 기간에 데이터가 없을 경우 '데이터가 없습니다.' 메시지가 표시
RQ-ID-0301	예측 화면	연도 및 자치구 선택	사용자가 예측 조회를 위해 자치구 및 연도를 선택
RQ-ID-0302	예측 화면	예측 그래프	2015~2022 실제값과 2023~2030 예측값이 동일 그래프에서 비교 표시
RQ-ID-0303	예측 화면	성능 정보 제공	예측 결과 하단에 모델 성능 정보 표시
RQ-ID-0304	예측 화면	비교 분석 기능	서울 평균 대비 선택 지역의 예측 추세를 비교할 수 있는 기능이 포함
RQ-ID-0401	지도 UI	SVG 지도 표시	서울시 기반 SVG 지도 UI가 화면에 표시
RQ-ID-0402	지도 UI	선택 연동 기능	특정 자치구 클릭 시 예측 데이터 및 그래프가 동기화

4. 요구사항 분석서 및 명세서 – 요구사항 명세서(2/2)

Agentic AI를 활용한 지역아동센터 수요 예측 서비스

요구사항 명세서

RQ-ID	화면 명	요구사항 명	요구사항 상세
RQ-ID-0501	Q&A	목록 조회	게시글 목록 조회 기능
RQ-ID-0502	Q&A	글 작성	Q&A 게시판 목록 화면에서는 각 게시글의 번호, 제목, 작성일시가 표 형태로 함께 표시
RQ-ID-0504	Q&A	댓글 기능	게시글에는 댓글 작성 및 표시 기능이 제공
RQ-ID-0601	로그인	로그인 기능	사용자 아이디 + 비밀번호로 로그인
RQ-ID-0602	로그인	오류 처리	로그인 실패 시 오류 메시지가 표시
RQ-ID-0603	회원가입	회원 등록	사용자 정보 입력 후 회원가입
RQ-ID-0604	회원가입	회원가입 입력값 유효성 검사	이메일 형식이 올바르지 않거나 비밀번호·비밀번호 확인이 일치하지 않을 경우 각각 오류 메시지를 표시
RQ-ID-0605	회원가입	암호화 저장	비밀번호는 암호화(HASH) 처리되어 저장
RQ-ID-0606	로그아웃	로그아웃 알림 메시지	사용자가 로그아웃할 경우, "로그아웃 되었습니다."와 같은 안내 내용을 팝업으로 표시
RQ-ID-0607	인증	아이디/비빈번호 찾기	이메일 인증 기반으로 계정 정보 찾기
RQ-ID-0701	생성형 AI	모델 선택	Base model / checkpoint 50 / checkpoint 100 중 비교모델 선택
RQ-ID-0702	생성형 AI	챗봇	연동된 DB와 모델을 기반으로 사용자의 질문에 맞는 답변을 제시
RQ-ID-0703	생성형 AI	챗봇 - 지역아동센터 찾기	현재 위치를 기반해 근방의 아동센터 찾는 기능 제공
RQ-ID-0704	생성형 AI	챗봇 - 대화 문맥 유지	사용자가 대화 과정에서 제시한 정보를 시스템이 기억하여 후속 질의 시 해당 맥락이 반영된 정확한 정보를 제공
RQ-ID-0705	생성형 AI	챗봇 - 모델 비교	처음 선택한 모델의 답변 퀄리티를 비교 분석
RQ-ID-0706	생성형 AI	하이퍼파라미터 조정	추론 소요 시간, 생성 토큰 수 등 주요 성능 지표를 실시간으로 시각화하고 로그 파일로 관리

4. 요구사항 분석서 및 명세서 – 요구사항 분석서(1/3)

1.메인 페이지 기능

1.1 메뉴 이동 기능

- 상단 메뉴 클릭 시 각 페이지로 이동할 수 있어야 한다.
- 상단 메뉴는 모든 페이지에서 동일한 레이아웃으로 고정되어야 한다.
- 로그인 상태일 경우 "로그아웃" 버튼이 표시되고 클릭 시 로그아웃 처리 후 메인 화면으로 이동해야 한다.

1.2 캐러셀 기능

- 메인 페이지에는 자동 재생되는 캐러셀이 있어야 한다.
- 캐러셀은 약 5 초 간격으로 자동 전환되어야 한다.
- 좌우 네비게이션 버튼 또는 인디케이터 클릭을 통해 수동 제어가 가능해야 한다.

1.3 스크롤 스냅 기능

- 메인 화면 스크롤 시 주요 섹션 단위로 화면이 자연스럽게 정렬 되도록 스크롤 스냅이 적용되어야 한다.
- 사용자 스크롤을 내리거나 올릴 때, 각 섹션이 부드럽게 전환되며 화면이 고정 되어야 한다.

1.4 주요 서비스 이동 버튼 기능

- 메인 페이지에 서비스 이동 버튼(통계 대시보드, 머신러닝, QnA)이 제공되어야 한다.
- 버튼 클릭 시 해당 기능 페이지로 이동해야 한다.

2. 서비스 소개 기능

2.1 서비스 목적 소개 기능

- 서비스 목적이 첫 화면에서 명확히 전달되어야 한다.
- 예측 기반 서비스의 필요성과 사회적 활용성을 문구 및 시각 요소로 설명해야 한다.

2.2 데이터 기반 설명 기능

- 서비스가 실제 데이터 기반으로 동작한다는 안내가 포함되어야 한다.

3 통계 대시보드 기능

3.1 연도·지역 필터 기능

- 사용자는 연도 범위 및 자치구를 선택하여 데이터를 조회할 수 있어야 한다.
- 필터 변경 시 새로고침 없이 동적으로 값이 갱신되어야 한다.

3.2 표 및 지표 표시 기능

- 이용자 수, 시설 수, 증감률 등 핵심 수치가 카드형 UI 로 표시되어야 한다.
- 조회된 데이터는 표 및 그래프 형태로 시각화되어야 한다.

3.3 데이터 예외 처리 기능

- 조회 범위 내 데이터가 없을 경우 "데이터가 없습니다." 메시지를 표시해야한다.

4. 요구사항 분석서 및 명세서 – 요구사항 분석서(2/3)

4 머신러닝 화면 기능

4.1 연도 및 자치구 선택 기능

- 사용자가 예측 값 조회를 위해 자치구 및 연도를 선택할 수 있어야 한다.
- 선택된 값은 즉시 화면에 반영되어야 한다.

4.2 예측 그래프 표시 기능

- 2015~2022 년 실제 데이터와 2023~2030 년 예측 데이터를 동일 그래프에서 비교하여 표시해야 한다.
- 그래프는 마우스 오버 시 수치가 강조 표시되어야 한다.
- 예측 그래프 밑에는 모델 성능 표시가 있어야 한다.

4.3 비교 분석 기능

- 서울 평균과 선택 구의 예측 값 비교 기능이 제공되어야 한다
- 전년 대비 증감 수치 및 변화율(%) 정보가 포함되어야 한다.

5 지도 연동 기능

5.1 SVG 지도 표시 기능

- 서울시 자치구 기반 SVG 지도 UI 가 제공되어야 한다.
- 자치구 영역은 Hover 및 선택 시 색상 변화가 적용되어야 한다.

5.2 선택 연동 기능

- 지도 선택, 드롭다운 선택 등 UI 요소 간 동기화가 이루어져야 한다.
- 선택된 구의 예측 분석 데이터가 우측 UI 에 업데이트되어야 한다.

6 QnA 게시판 기능

6.1 목록 조회 기능

- 사용자는 게시글 목록을 확인할 수 있어야 한다.
- 목록에는 제목, 작성자, 작성일, 댓글 수가 포함되어야 한다.

6.2 글 작성·수정·삭제 기능

- 로그인 사용자는 질문을 작성할 수 있어야 한다.
- 게시글 수정 및 삭제는 작성자 본인만 가능해야 한다.

6.3 댓글 기능

- 게시글에는 댓글 작성 기능이 제공되어야 한다.

4. 요구사항 분석서 및 명세서 – 요구사항 분석서(3/3)

7 사용자 인증 기능

7.1 로그인 기능

- 사용자는 ID(또는 이메일)과 비밀번호로 로그인할 수 있어야 한다.
- 로그인 실패 시 오류 메시지가 표시되어야 한다.

7.2 회원가입 기능

- 사용자 정보 입력 후 회원가입이 가능해야 한다.
- 비밀번호는 암호화(HASH) 처리되어 저장되어야 한다.

7.3 비밀번호·아이디 찾기 기능

- 이메일 또는 인증 방식으로 정보 찾기 기능이 제공되어야 한다.

8 생성형 AI 기능

8.1 자치구별 수요 예측 분석

- 사용자가 특정 자치구를 선택할 경우, Oracle DB 의 수요 예측 데이터를 조회하고, 이를 기반으로 LLM 용 Context 를 자동 생성한다.
- 파인튜닝된 Llama-3 모델을 활용하여 해당 자치구의 데이터 기반 수요 분석 및 전략적 답변을 도출한다..

8.2 지역아동센터 정보 제시

- 현재 사용자의 위치를 기반으로 질문 정보에 따른 답변을 제시한다.
- 사용자 가독성을 위해 답변 생성 시 불필요한 수식어나 중복 서술은 지양하고, 핵심 정보 위주의 요약된 형태로 응답을 생성한다.

8.3 대화 문맥 유지

- 사용자가 이전 대화에서 제공한 정보를 기억하여, 후속 질문이나 재질문 시 앞선 맥락이 반영된 정확한 정보를 제공한다.

8.3 대화 문맥 유지

- 사용자가 이전 대화에서 제공한 정보를 기억하여, 후속 질문이나 재질문 시 앞선 맥락이 반영된 정확한 정보를 제공한다.

8.4 모델 성능 비교 및 추론 관리 기능

- 파인튜닝 과정에 따른 모델 버전(Base Model, 최종학습모델 등)을 선택하여 동일 질문에 대한 답변 차이를 확인할 수 있는 인터페이스를 제공해야 한다.
- 추론 소요 시간 및 생성 토큰 수 등 성능 지표를 실시간으로 확인하고 로그로 관리해야 한다.
- FastAPI 기반의 외부 추론 서버와 타임아웃(180 초) 이내에 안정적으로 통신이 이루어져야 한다.

5. 화면 설계서 - 메인화면

화면 코드	MA-01	화면 경로	Main	페이지 명	메인 페이지 (캐러셀)	페이지	2

Description	
1	• 캐러셀 • 좌우 화살표 클릭 시 이전·다음 슬라이드로 이동
2	• 하단 인디케이터 • 하단 점(●) 클릭 시 해당 슬라이드로 이동
3	• Snap Scroll • 마우스 휠 또는 스크롤 시 다음 섹션으로 자동 이동
4	• 같은 페이지 내 '서비스 소개(스냅 스크롤-1)' 섹션으로 자동 스크롤

5. 화면 설계서 - 머신러닝

화면 코드	ML-01	화면 경로	Main - ML	페이지 명	머신러닝	페이지	6

Description	
1	• 원하는 연도·자치구 선택 후 조회 시 지도·그래프·표가 동시에 갱신 • 페이지 최초 진입 시 기본 값 : 최신 연도, 서울시 전체
2	• 지도에서 자치구 클릭 시 조회 (선택 구 색상 강조)
3	• 조회 조건의 실제·예측 이용자 수 그래프 • 전년·서울 평균 대비 증감 표시 • 마우스 오버 시 연도별 실제/예측 값과 증감률 툴팁 표시
4	• 선택 연도·자치구의 예측 이용자 수와 주요 지표 표

5. 화면 설계서 - 대시보드

화면 코드	ST-01	화면 경로	Main - Dashboard	페이지 명	통계 대시보드	페이지	7

Description	
1	• 지역 선택 및 검색 기능 • 자치구 이름 입력 시 자동 완성 목록 노출 • 페이지 최초 진입 시 기본 값 : 서울시 전체
2	• 조회 가능 연도 : 2015년 ~2022년 • 종료 연도 선택 : 시작 연도와 같거나 이후 연도만 선택 가능 • 종료 연도 선택 : 시작 연도와 같거나 이후 연도만 선택 가능
3	• 선택한 지역과 기간을 상단 설명 문구로 표시 • 조건에 해당하는 지역아동센터 이용자 수·시설 수를 테이블로 제공

5. 화면 설계서 – 생성형ai

화면 코드	AI-01	화면 경로	Main – 생성형 AI	페이지 명	생성형 AI	페이지	8

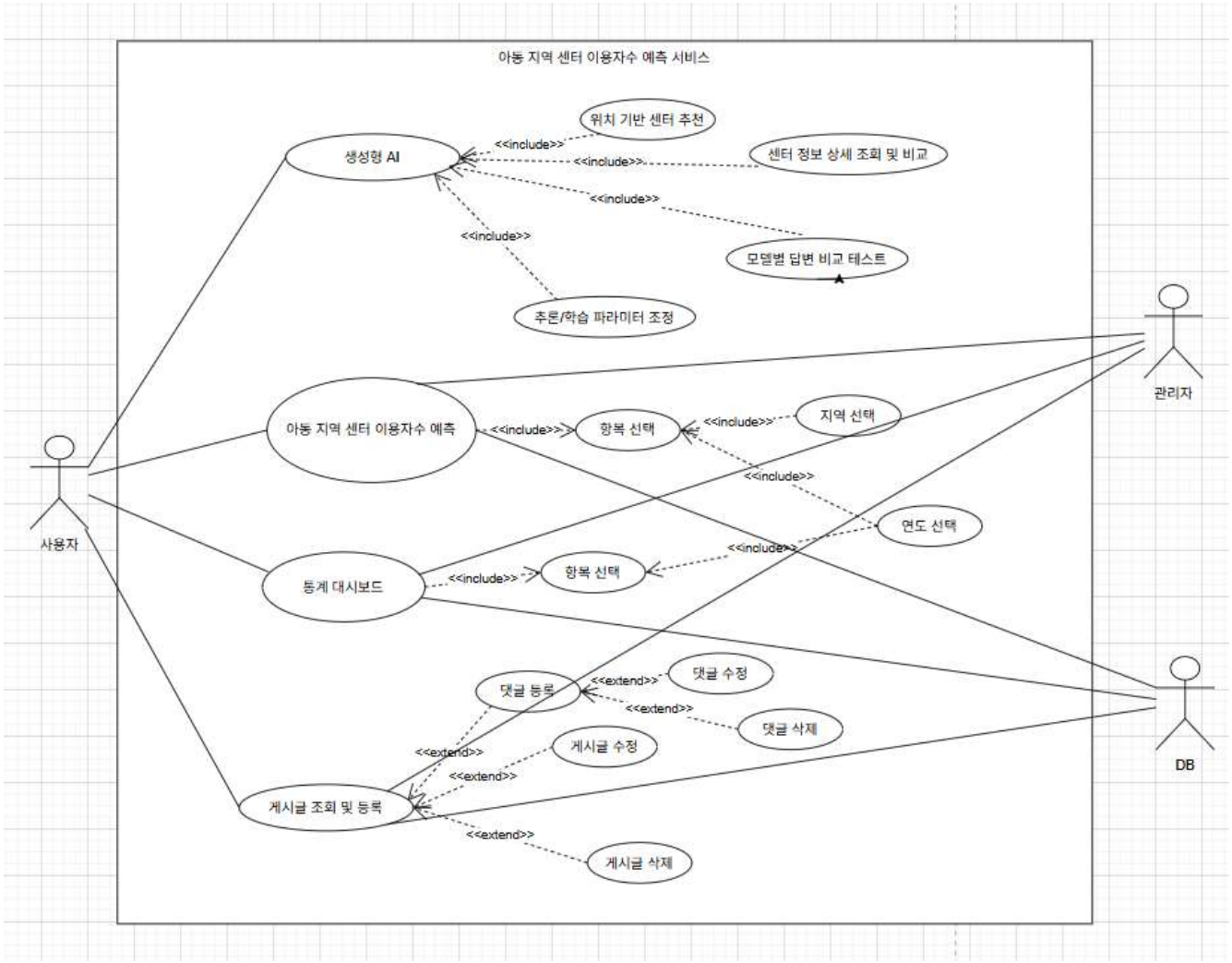
	Description
1	• Llama-3-8b 기본 모델 • 파인튜닝 checkpoint 50 • 파인튜닝 checkpoint 100 • 위 모델 3개를 선택, 비교
2	• Temperature, token값을 조정 할 수 있다. • 변경하면 답변길이와 창의성에 반영된다.
3	• Lora 학습에 사용되는 Parameter를 조정 할 수 있다. • 값을 조정하면 답변길이와 창의성에 반영된다.

5. 화면 설계서 – 생성형ai

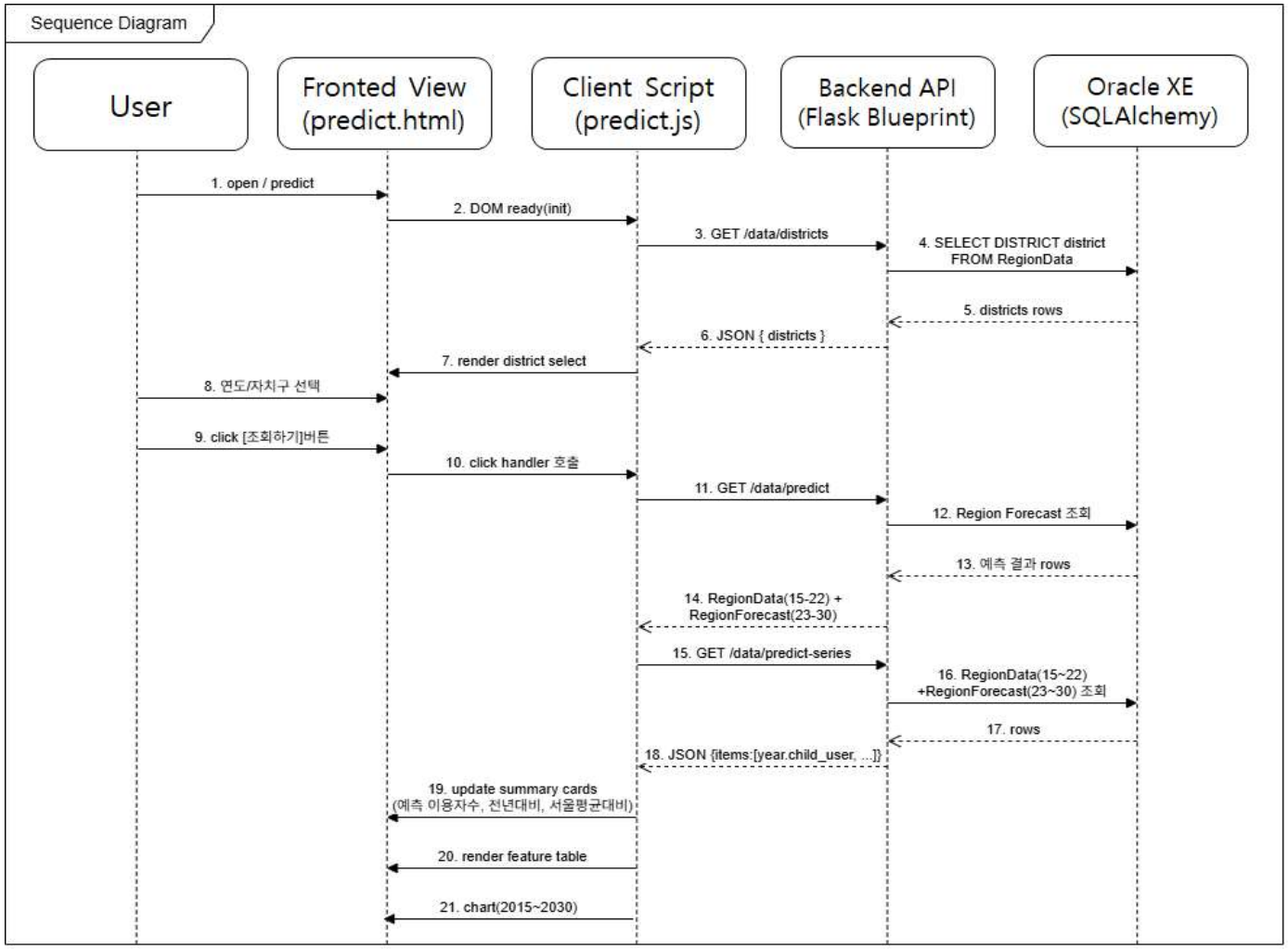
화면 코드	AI-02	화면 경로	Main – 생성형 AI	페이지 명	생성형 AI	페이지	9

Description	
1	• 현재 위치 정보
2	• 현재 위치에 기반하여 가장 가까운 지역 아동 센터를 알려줌 (기본값 3개)
3	• 정확한 센터명에 대한 정보를 요청 시 정보 제공
4	• 두 개의 아동센터의 정보를 비교 제공
5	• 존재하지 않는 센터명에 대한 정보를 요구 시 찾을 수 없다고 정보 제공 • 사용자의 위치정보가 제공되지 않을 시 위치 허용 요구

6. UML – Usecase diagram



6. UML – Sequence diagram



7. 주요 기능 및 서비스 코드 – 파인튜닝

```
# 모델 경량화: Quantization
from transformers import BitsAndBytesConfig
import torch

quantization_config = BitsAndBytesConfig(
    log_in_4bit=True, # 4비트 양자화활성화
    bnb_4bit_compute_dtype=torch.bfloat16, # bfloat 16
    bnb_4bit_use_double_quant=True, #double 양자화 사용
    bnb_4bit_quant_type='nf4') # 데이터 타입을 Normal Float 4-bit으로 설정
```

```
# 기본 Llama 3 모델 로드
from transformers import AutoModelForCausalLM
model = AutoModelForCausalLM.from_pretrained(
    'meta-llama/Meta-Llama-3-8B',
    quantization_config = quantization_config,
    device_map = {"": 0})  # 모든 레이터를 0번 gpu에 배치
```

- **QLoRA: 모델 경량화 및 양자화 설정**

대형 모델을 효율적으로 로드하기 위해 4비트 양자화를 적용하는 부분 -> VRAM 절약

✓ 4-bit NF4 양자화
80억개의 파라미터를 4비트 단위로 압축해 VRAM 절약, 압축하며 발생하는 손실은 최소화

✓ Double_quant
압축을 위해 필요한 추가 데이터까지 한 번 더 압축 -> 극한으로 메모리 확보 (대형 모델 학습 가능)

✓ 연산 정밀도(Compute_dtype)
저장은 4-bit, 계산은 16-bit로 수행
속도 & 정확도 향상 -> 학습 효율 극대화

✓ 효율적인 모델 로드(device_map = {"": 0}) :
모든 모델 레이어를 0번 GPU에 고정 배치하여
학습 중 발생할 수 있는 데이터 병목 현상을 방지

7. 주요 기능 및 서비스 코드 – 파인튜닝

- **LoRA : 효율적 파라미터 튜닝**

  기존 방식이 80억 개를 수정했다면,
  LoRA는 모델 옆에 작은 어댑터를 붙여 필요한 부분만 학습

```python
# 모델 경량화 설정 : LoRA 설정
from peft import LoraConfig
peft_config = LoraConfig(
    lora_alpha = 32,
    lora_dropout = 0,
    r = 16,
    bias = 'none',
    task_type = 'CAUSAL_LM',
    target_modules = ['q_proj', 'v_proj', 'k_proj', 'o_proj', 'gate_proj',
                      'up_proj', 'down_proj'])
```

✓ 핵심 설정값 의미

- r = 16 : 학습 할 파라미터의 통로 크기, 아주 작은 행렬만 (r=16) 학습하여 연산량을 매우 줄임
- lora_alpha=32 : 학습된 내용이 기존 모델에 얼마나 반영될지 결정하는 가중치 계수 (보통 r의 2배로 설정)
- target_modules : Llama3의 핵심인 Attention 레이어를 타게팅해 대화의 맥락을 파악하는 능력을 집중적으로 고도화

✓ 기술적 기대 효과

- 저비용 고효율 : 모델 본체는 얼려두고, 미세한 어댑터만 학습하므로 학습 속도가 수배 이상 빠름
- 어댑터 교체 방식 : 학습 결과물이 수십 MB 수준으로 매우 가벼움

  -> 나중에 이 어댑터 파일만 갈아 끼우면 상담용, 분석용 등 목적에 따라 AI의 인격을 즉시 바꿀 수 있음

7. 주요 기능 및 서비스 코드 – 파인튜닝

- **SFT : 지도 학습 기반 미세 조정**

SFT는 AI에게 "질문에 대해 이렇게 답변하라"는 모범 답안을 직접 가르쳐서 모델의 말투와 지식 범위를 특정 목적에 맞게 교정하는 단계

✓ 학습 최적화 설정

- 배치 최적화 (batch_size=2, accumulation_steps=8) :
  - 실제로는 한 번에 16개(2x8)의 데이터를 묶어서 학습하는 효과
  - GPU 메모리는 적게 쓰면서 학습 안정성은 높이는 전략
- Optim = adamw_8bit :
  - 최적화 알고리즘이 사용하는 메모리까지 8비트로 압축
  - VRAM 부족으로 인한 학습 중단을 방지
- bf16 정밀도 (bf16=True):
  - 기존의 fp16보다 수치 안정성이 높은 포맷
  - 학습 중 값이 에러가 나는 현상을 막아 모델의 성능을 안정적으로 끌어올림
- eval_steps=10 / save_steps=50 :
  - 10 step마다 성능 점검, 50 step마다 저장
- max_steps=100 :
  - 학습 스텝을 100회로 최적화하여 지식 습득과 과적합 방지 사이의 균형을 맞춤

```
# Training setup
from trl import SFTTrainer
from transformers import TrainingArguments

max_steps = 100

training_arguments = TrainingArguments(
    output_dir=local_output_dir,
    report_to="tensorboard",
    per_device_train_batch_size=2,
    per_device_eval_batch_size=2,
    gradient_accumulation_steps=8,
    max_steps=max_steps,
    warmup_steps=30,
    learning_rate=1e-4,
    lr_scheduler_type="constant_with_warmup",
    evaluation_strategy="steps",
    eval_steps=10,
    save_strategy="steps",
    save_steps=50,
    logging_steps=1,
    optim="adamw_8bit",
    weight_decay=0.01,
    seed=42,
    gradient_checkpointing=True,
    gradient_checkpointing_kwargs={"use_reentrant": False},
)

trainer = SFTTrainer(
    model = model,
    tokenizer = tokenizer,
    train_dataset = train_dataset,
    eval_dataset = test_dataset,
    peft_config = peft_config,
    dataset_text_field = 'text',
    max_seq_length = 2048,
    dataset_num_proc = 2,
    packing = False,
    args = training_arguments,
    **data_collator_param )
```

7. 주요 기능 및 서비스 코드 – 생성형AI

```
NAME_SET_RE = re.compile(r"(내\s*이름은|나는)\s*([가-힣A-Za-z]{2,10})(이야|입니다|야)?")
```

- NAME_SET_RE = re.compile() : AI가 사용자의 말 중에서 어디가 이름인지 알아채기 위한 탐지 규칙을 만듦
  - ✓ 내 이름은 철수야 또는 나는 미영이야 같은 문장 구조를 인식
  - ✓ (내이름은/나는) : group(1), (이름부분 가-힣) : group(2), (입니다,이야,야) : group(3) 으로 추출해 각 그룹에 저장

```
if m_set := NAME_SET_RE.search(msg):
    session['user_name'] = m_set.group(2)
    return jsonify({"text": f"반가워요, {m_set.group(2)}님! 이름을 기억해둘게요.", "centers": []})
if NAME_ASK_RE.search(msg):
    saved = session.get('user_name')
    return jsonify({"text": f"사용자님 성함은 {saved}입니다. 잊지 않고 있어요!" if saved else "아직 성함을 모르겠어요.", "centers": []})
```

- session[‘user_name’] : 사용자가 채팅에 이름을 쳤을 때 파악해 저장
  - ✓ 탐색 - search(msg) : 메시지에 패턴이 있는지 search
  - ✓ 추출 - group(2) : 패턴이 발견되면 group(2)만 가져온다.
  - ✓ 기억 - session : session에 user_name을 넣는다
- session.get(‘user_name) : 나중에 사용자가 이름을 되물었을 때 session을 뒤져 대답해줌
  - ✓ 확인 – get : session안에 user_name 데이터가 있는지 확인
  - ✓ 응답 – jsonify : 데이터가 있다면 이름을 문장에 넣어 답변, 없으면 모른다고 답변

7. 주요 기능 및 서비스 코드 – 생성형AI

```
PROMPT_TMPL = """너는 아동센터 전문가야.
사용자가 질문하는 지역의 아동센터를 친절하게 추천해줘.
아는 대로 최대한 많이 알려줘."""
```

```
PROMPT_TMPL = """너는 지역아동센터 추천 비서다.
출력 규칙(중요): 아래 CENTER_CONTEXT에 있는 실제 센터만 사용한다. 답변은 반드시 "추천 줄"만 출력한다.
[CENTER_CONTEXT]
{center_context}
[사용자 질문] {question}
"""
```

- 적은 근거와 포괄적으로 프롬프트를 설정
  - ✓ 정확도 저하 :엉뚱한 센터 추천, DB에 없는 가짜 센터이름
  - ✓ 지나치게 긴 답변, 이해할 수 없는 설명 글
- 제약조건과 실제 데이터를 명확히 주입
  - ✓ AI에게 CENTER_CONTEXT에 있는 정보(DB)만 쓰라고 지시
  - ✓ 비서라는 역할을 주어 깔끔한 답변과 말투를 유도

```
ctx = "\n".join([f"- {c['center_name']} | {c.get('distance_km')}km" for c in centers])
raw = call_runpod(PROMPT_TMPL.format(center_context=ctx, question=msg, limit=opt["limit"]), max_new_tokens=160)
ans = clean_answer(raw, limit=opt["limit"])
```

- Ctx 생성 : DB에서 가져온 리스트 형태의 데이터를 AI에게 전달하기 위해 하나의 긴 문자열로 합치는 과정
  - ✓ centers 리스트(DB) 안에 있는 각 센터의 이름과 거리를 꺼내서 '센터명 | 1km' 같은 형태로 만듦
- Clean_answer : AI가 답변 외에 출력하는 불필요한 태그나 문구들을 지우고 사용자에게 보여줄 깨끗한 문장만 남김
  - ✓ 사용자는 깔끔하게 정리된 아동지역센터 추천 리스트만 화면에서 제공 받음

7. 주요 기능 및 서비스 코드 – 머신러닝

✓ 다양한 가구 특성 및 인구 데이터를 기반으로 자치구별 지역아동센터 수요 예측

- 데이터 출처 : KOSIS, 서울 열린 데이터 광장 (2015~2022년)
- 분석 범위 : 서울시 25개 자치구의 연도별 통계 (연도 x 자치구)
- 주요 변수
  - Target : child_user(이용자 수)
  - Feature :
    - 한부모 가정 수(single_parent)
    - 기초생활수급 가구 수(basic_beneficiaries)
    - 다문화 가구 수(multicultural_hh)
    - 학원 수(academy_cnt)
    - GRDP(지역 내 총 생산), 인구(population), 연도(year)
    - 자치구(district): 원-핫 인코딩 적용

```
# Feature 설정
base_features = [
    "year",
    "single_parent",
    "basic_beneficiaries",
    "multicultural_hh",
    "academy_cnt",
    "grdp",
    "population"
]

features = base_features + district_ohe_cols
target = "child_user"

# Train/Test Split
train = df[df["year"] <= 2020]
test  = df[df["year"] >= 2021]

X_train = train[features]
y_train = train[target]

X_test = test[features]
y_test = test[target]

y_train_log=np.log1p(y_train)
```

7. 주요 기능 및 서비스 코드 – 머신러닝

✓ 전처리 과정

```
# district 원핫 인코딩
df = pd.get_dummies(df, columns=["district"], drop_first=False)

district_ohe_cols = [c for c in df.columns if c.startswith("district_")]
```

- 자치구 : 범주형 → 원-핫 인코딩

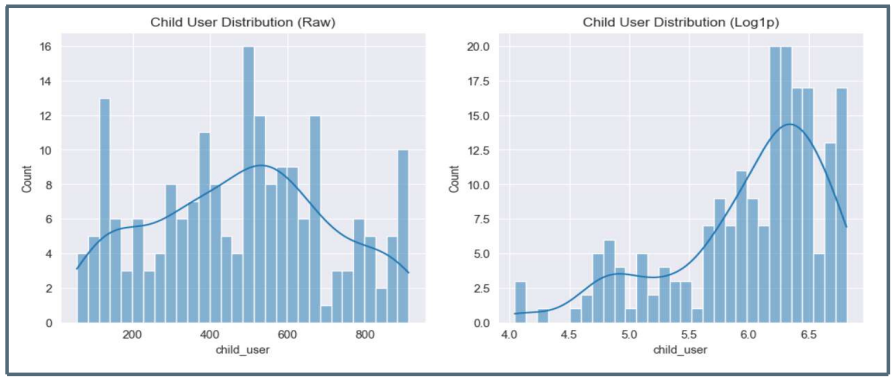
  district 범주형 변수를 pd.get_dummies()로 원-핫 인코딩해
  district_로 시작하는 더미 컬럼들을 생성하고
  district_ohe_cols으로 저장

```
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

sns.histplot(df_master["child_user"], kde=True, bins=30, ax=axes[0])
axes[0].set_title("Child User Distribution (Raw)")

sns.histplot(np.log1p(df_master["child_user"]), kde=True, bins=30, ax=axes[1])
axes[1].set_title("Child User Distribution (Log1p)")

plt.show()
```

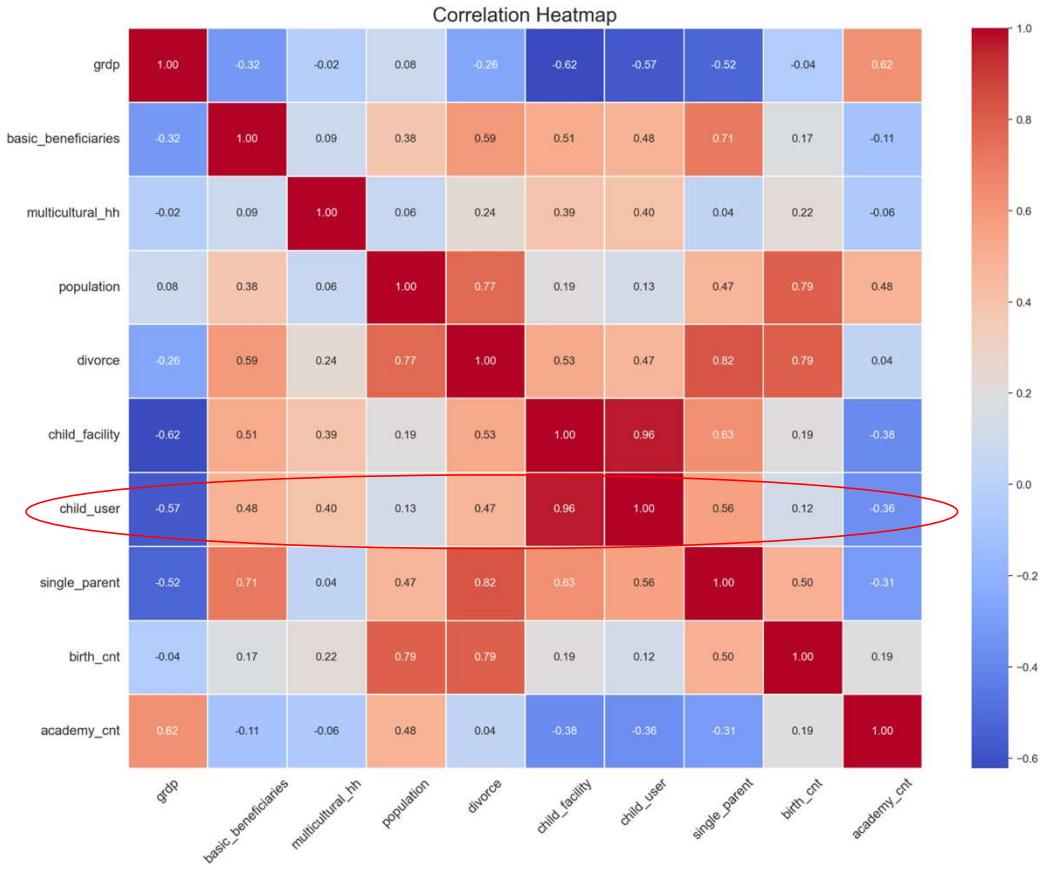


- y_train에 log1p를 적용해 스케일을 완화하고 학습 안정성을 높인다.
- 종속 변수인 이용자 수 데이터의 편향성을 해결하기 위해 log 변환을 수행, 학습 최적화 및 예측 정확도 향상 기반 마련

7. 주요 기능 및 서비스 코드 – 머신러닝

✓ 변수 간 상관관계 분석

- Child_user : target
  - Target은 취약가구, 이혼 지표와 양의 상관계수
  - Target은 GRDP, 학원 수와 음의 상관
  - 취약도가 높을수록 수요 ↑,
  - GRDP, 사교육이 높을수록 수요 ↓



7. 주요 기능 및 서비스 코드 – 머신러닝

```
# 최종 XGBoost 모델
best_xgb_ohe = XGBRegressor(
    **search_ohe.best_params_,
    random_state=42
)
best_xgb_ohe.fit(X_train_ohe, y_train_ohe_log)

best_xgb_ohe.district_ohe_cols = district_ohe_cols
best_xgb_ohe.base_features = base_features

# 평가: 로그에서 원래 스케일로 되돌려서 성능 확인
pred_log_ohe = best_xgb_ohe.predict(X_test_ohe)
pred_ohe = np.expm1(pred_log_ohe)
```

```
xgb_ohe = XGBRegressor(
    random_state=42,
    tree_method="hist"
)

search_ohe = RandomizedSearchCV(
    estimator=xgb_ohe,
    param_distributions=param_grid_local,
    n_iter=30,
    scoring='r2',
    cv=3,
    verbose=2,
    n_jobs=-1,
    random_state=42
)

search_ohe.fit(X_train_ohe, y_train_ohe_log)
```

R² : 0.8438539505004883

✓ 모델 선정 과정

- 1차 (Linear Regression) : 기본 성능 파악 결과 R^2 값이 0.56으로 자치구별 편차 설명력 부족
- 2차 (XGBoost) : 비선형 관계 및 변수 간 상호작용 학습 -> R^2 값이 0.76으로 성능 향상
- 최종 모델 선정 사유 : XGBoost
  - 안정성 : 선형회귀는 예측 시 특정 구에서 값이 비정상적으로 튀는 현상 발생
  - 현실성 : XGBoost는 완만하고 현실적인 예측패턴 유지
- 모델 최적화 : 하이퍼파라미터 튜닝 + 타겟변수 log1p 변환 적용 -> R^2 값 0.84
- 로그 역변환 : 예측 시에는 np.expm1()을 사용하여 실제 이용자 수 스케일로 복원 후 성능을 평가함

7. 주요 기능 및 서비스 코드 – 머신러닝

```python
# CAGR 계산 함수
def calc_cagr(series, start_year, end_year):
    v0 = series.loc[start_year]
    v1 = series.loc[end_year]
    if v0 <= 0 or v1 <= 0:
        return 0.0
    return (v1 / v0) ** (1 / (end_year - start_year)) - 1
```

```python
base_row = df_dist[df_dist["year"] == last_year].iloc[0]

for year in range(future_start, future_end + 1):
    years_ahead = year - last_year

    new_row = {"district": district, "year": year}

    for col in growth_rates.keys():
        base_val = base_row[col]
        rate = growth_rates[col]
        new_row[col] = base_val * ((1 + rate) ** years_ahead)

    future_rows.append(new_row)
```

```python
# log 스케일 → 원래 스케일로 되돌리기
pred_future_log = trained_model.predict(X_future)
future_df_ohe["child_user"] = np.expm1(pred_future_log)
```

- CAGR 계산 공식 함수
  - 단순 전년 대비 증감률을 보는 것이 아닌, 수년에 걸친 평균 성장 흐름을 파악
  - 자치구별로 서로 다른 인구 증감이나 환경 변화의 특성을 수치화하여 미래 데이터 생성의 근거로 활용합니다.
- 예측 데이터값 생성
  - 2022년의 데이터를 기준으로, 앞서 구한 성장률(rate)을 시간의 흐름(years_ahead)만큼 곱하여 미래 수치를 추정
  - 모델이 예측을 수행할 수 있도록 논리적인 미래 시뮬레이션 환경을 구축하는 단계
- 생성된 미래 데이터셋(X_future)을 XGBoost 모델에 입력하여 예측값(child_user)을 뽑아낸 뒤, 로그 형태를 원래 단위인 '명'으로 되돌립니다.

✓ 결과

- 미래 수요 예측 : 과거 연평균 증가율(CAGR) 공식 기반 미래 Feature 생성 및 입력
- 기대 효과 :
  - 자치구별 수요 변화 예측 값을 통한 정책 근거 제공
  - 시설 필요 및 인력 배치 최우선 순위 지역 도출 가능

8. 시연

프로젝트 시연

https://drive.google.com/file/d/1NWxbI34-5WOBNnJ0sUij6oQcI9c1L3aH/view?usp=sharing

9. 향후 개발 계획

데이터셋 강화

- 지역별 데이터, 정책 보고서 같은 비정형 데이터를 추가 수집해 데이터셋을 강화 시킬 계획입니다.
- 발생 빈도가 낮아 데이터 확보가 어려운 소외 지역이나 긴급 보호 사례 등은 LLM을 활용한 합성 데이터 생성 기법을 도입하여 학습 데이터의 불균형을 기술적으로 해소하고자 합니다.

서비스 강화

- 기존 100회의 학습 스텝을 500회까지 5배 확대하여 모델이 정책 및 수요 예측 데이터의 미세한 패턴을 충분히 내재화할 수 있도록 심화 학습을 진행할 계획입니다.
- LoRA의 핵심 파라미터인 Rank(r)와 Alpha(a) 수치를 상향 조정함으로써 모델의 지식 수용력을 높이고, 결과적으로 단순 요약을 넘어선 구체적이고 전문적인 답변 생성 능력을 확보 할 계획입니다.
- 사용자 질문에 대한 응답 퀄리티를 근본적으로 개선하기 위해 프롬프트 엔지니어링을 구체화할 예정입니다.

10. 프로젝트 수행 소감

✓ **팀원과의 소통**

처음에는 각자 맡은 일만 잘하면 된다고 생각했지만 중간에 여러 문제들을 겪으며 소통의 중요성을 느꼈습니다.
소통의 중요성을 느낀 이후 매일 아침 서로의 진행 상황을 공유하기 시작했고, 결과 뿐만 아닌 과정도 같이 하려 노력했습니다. 이 과정을 통해 결과도 중요하지만 과정의 공유에서 협업이 시작된다는 것을 배웠습니다.

✓ **데이터의 품질 = 모델의 성능**

단순히 최신 모델을 사용하는 것보다, Log1p 변환이나 선형 보간법과 같은 세밀한 전처리와 데이터 시뮬레이션이 모델의 예측 신뢰도를 결정짓는 핵심임을 체감했습니다.
데이터의 편향성을 잡고 결측치를 논리적으로 메우는 과정이 프로젝트의 기반이 됨을 배웠습니다.