

최종보고서(기계학습의 이해, 202403234, 양지현)

1. 프로젝트 개요

무엇을 만들었나

본 프로젝트는 자연어로 입력된 일정 문장을 기반으로

- 일정 카테고리 자동 분류(과제, 시험, 발표, 기타)
- 일정 중요도(High, Medium, Low) 예측
- 마감일·우선순위 기반 정렬
- 일정 누적 저장 및 삭제/완료 처리 기능

을 제공하는 **AI** 기반 일정 관리 웹 서비스를 개발하는 것을 목표로 한다.

어떤 문제를 해결했나

학생들은 학기 중 시험, 과제, 발표, 비교과 활동 등 다양한 일정이 동시에 발생한다.

기존 일정 앱은 일정 기록에만 초점이 맞춰져 있어 일정이 얼마나 중요한지 판단하는 기능이 부족하다.

이 프로젝트는 사용자가 문장을 입력하면 **AI**가 즉시 일정 분류 + 우선순위 분석을 수행하여 사용자가 가장 먼저 처리해야 할 일을 쉽게 확인할 수 있도록 돕는다.

2. 진행 과정

2-1. 주제 선정 및 문제 정의

- ‘일정은 입력하지만 어떤 일정이 더 중요한지 한눈에 알기 어렵다’는 문제에서 출발
- 자연어 기반 일정 문장을 **ML** 모델로 자동 분류하는 시스템이 필요하다고 판단
- 단순 기록형 일정 앱 → “의사결정 보조형 일정 관리 시스템”으로 발전 목표 설정

2-2. 데이터 수집 및 분석

(1) 직접 수집한 일정 데이터 (~50건)

- 학기 중 실제 일정(과제 제출, 시험 공부, 발표 준비, 학생회 활동 등)
- category(과제/시험/발표/기타)과 priority(High/Medium/Low)를 직접 라벨링
-

(2) Feature 구성

- text: 일정 문장
- days_left: 마감일까지 남은 일수
- contains_keyword: 핵심 키워드 포함 여부(중요도 반영)
- TF-IDF: 문장 벡터화(ngram 포함)

(3) 전처리 과정

- 날짜 정규화(YYYY-MM-DD)
- days_left 계산
- 불용어 제거 및 형태 단순화
- numeric feature 표준화 (StandardScaler)

2-3. ML학습 및 평가

- 일정 분류 모델은 Logistic Regression과 RandomForest를 비교해 본 뒤, 적은 데이터에서도 안정적인 성능을 보이는 Logistic Regression을 최종 모델로 선택하였다.
텍스트는 TF-IDF(max_features=1500, bigram 포함)로 벡터화하고, days_left 및 contains_keyword와 같은 numeric feature를 함께 결합하여 학습하였다.
- 모델 평가는 Accuracy와 Macro F1-score를 활용하여 Category(4-class) 분류와 Priority(3-class) 분류 성능을 각각 확인하였다.
- 카테고리 분류(4-class)의 경우 Test Set 기준으로 **Accuracy** 약 **0.63**, **Macro F1-score** 약 **0.62** 수준의 성능을 보였다.
텍스트 기반 특징이 영향을 크게 미치기 때문에 '과제', '시험'처럼 명확한 키워드가 존재하는 클래스에서는 높은 예측률을 보였으나 '발표'와 '기타' 클래스는 문장 표현의 다양성으로 인해 다소 혼동되는 경향이 있었다.
- 우선순위 분류(High/Medium/Low)는 상대적으로 더 좋은 결과를 보였으며, Test Set 기준 **Accuracy** 약 **0.75**, **Macro F1-score** 약 **0.74** 정도로 나타났다.
days_left와 긴급 단어 포함 여부 같은 numeric feature가 큰 영향을 미쳐 High와 Medium은 비교적 높은 정확도로 분류되었고, 데이터 수가 적은 Low 우선순위는 다소 불안정한 패턴을 보였다.
- 전체적으로 baseline 수준의 모델 구성임에도 불구하고 적절한 분류 성능을 보였다.

2-4. 서비스 개발

서비스는 **Gradio** 기반 웹 대시보드로 구현하였으며 주요 기능은 다음과 같다:

1. 일정 문장 + 마감일 입력
2. 모델이 카테고리 & 우선순위를 자동 예측
3. 일정이 실시간 테이블 형태로 누적
4. 우선순위 → 마감일 순으로 정렬된 “해야 할 일 순위표” 출력
5. 일정 삭제 기능

6. 완료 체크 기능(완료한 일정은 목록에서 제거)

사용자는 웹 UI를 통해 인터랙티브 일정 관리 서비스를 직접 이용할 수 있다.

3. 모델을 서비스로 만든 구조

학습한 모델을 실제 서비스로 연결한 방식

- 학습된 모델 파일(.pkl) 4종을 로드
 - category_model.pkl
 - priority_model.pkl
 - tfidf_vectorizer.pkl
 - feature_scaler.pkl
- 입력된 텍스트와 날짜를 실시간으로 전처리
- 같은 방식으로 feature vector 구성
- 서비스 내부에서 모델.predict() 호출하여 즉시 예측 수행
- pandas DataFrame에 일정 누적
- Gradio 테이블 컴포넌트로 시각화

시스템 구조 개요 (웹 서비스 형태)

- 사용자 입력
- ↓
- 전처리(days_left 계산, TF-IDF 변환 등)
- ↓
- ML 모델 예측(category, priority)
- ↓
- 일정 저장 및 누적 관리
- ↓
- 우선순위 기반 정렬 → 결과 UI 표시

API/CLI가 아닌 웹 기반 대화형 서비스 구조로 구현했으며
로컬 환경에서 누구나 실행할 수 있도록 단일 app.py 파일로 구성하였다.

4. 실제 사용 결과 (5회 이상 테스트)

아래는 실제 테스트 중 일부 상황 요약:

테스트 1 – “AI 윤리 과제 제출”, D-3

→ 카테고리: 과제 / 우선순위: High

→ 우선순위표 최상단 배치됨

테스트 2 – “학생회 회의 참석”, D-7

→ category: 기타, priority: Low

→ 순위표 하단에 위치

테스트 3 – “ML 기말고사 대비”, D-10

→ category: 시험 / priority: High

→ High 일정끼리 마감일 기준 정렬됨

테스트 4 – “프레젠테이션 발표 준비”, D-5

→ category: 발표 / priority: Medium

테스트 5 – 잘못된 일정 입력 후 삭제 기능 테스트

→ 정상 삭제됨

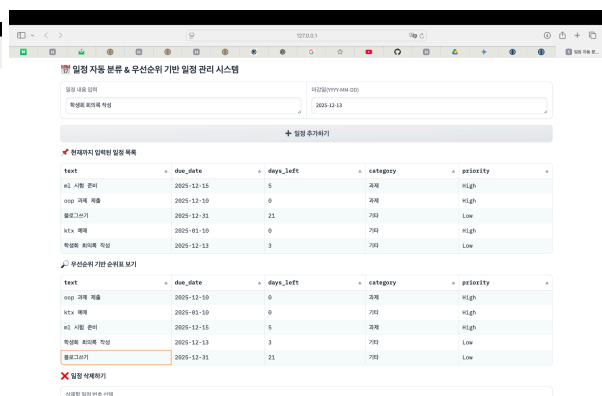
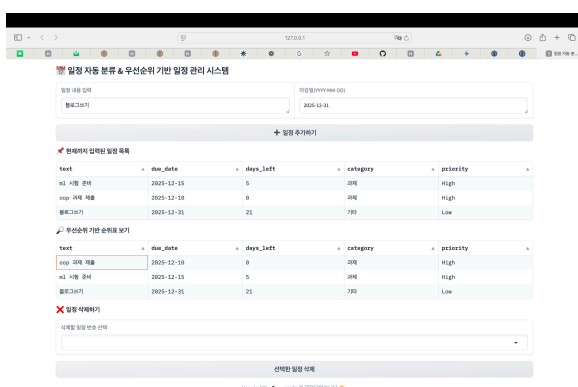
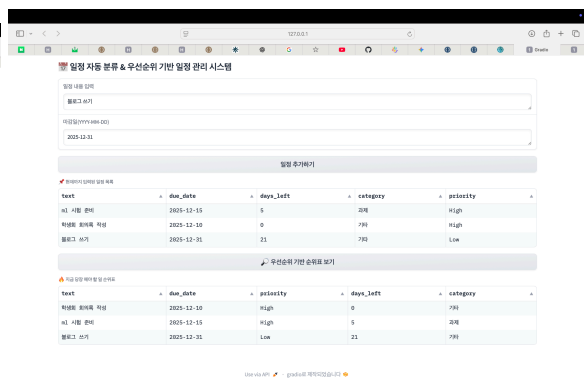
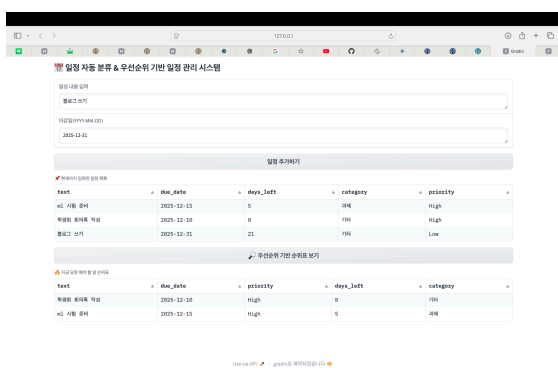
테스트 6 – 완료 체크 기능 확인

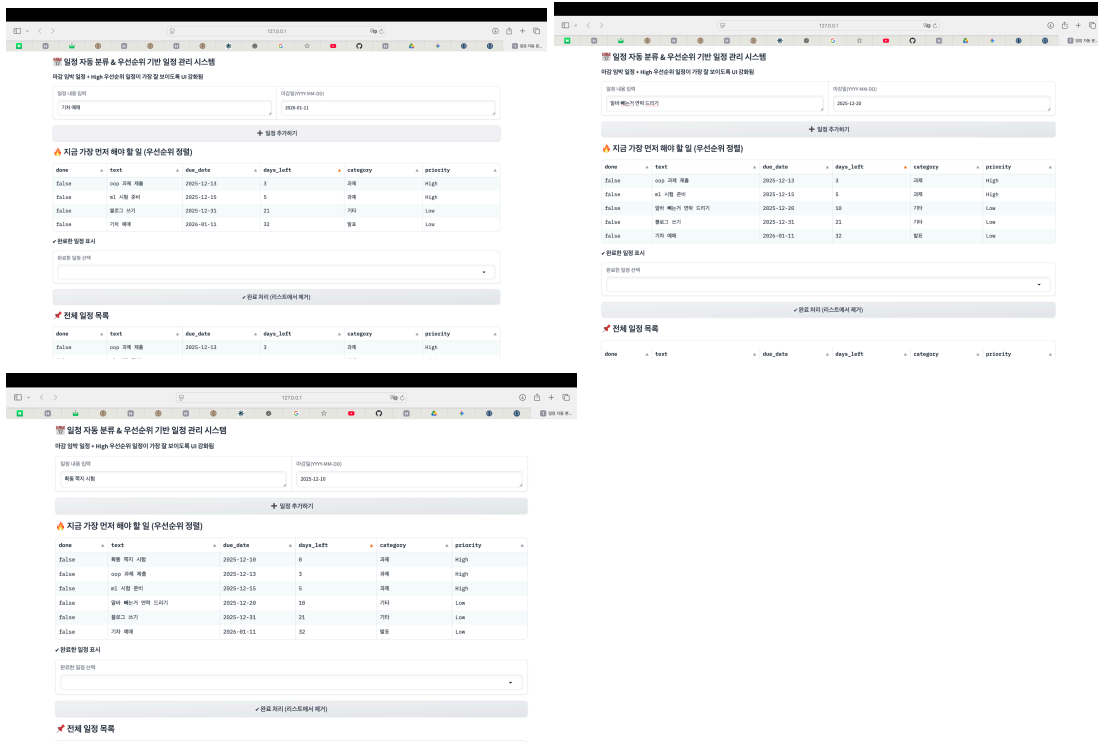
→ 완료된 일정이 테이블에서 사라짐

테스트 7 – 여러 일정 누적 테스트

→ 실시간 정렬 정상 작동

필요에 따른 UI 업데이트 과정을 거쳤다.





5. 배운 점 및 개선 방향

배운 점

- 자연어 기반 일정 분류는 단순해 보이지만 실제로 **feature** 설계가 매우 중요함
- **days_left, contains_keyword** 같은 **numeric feature**가 예측 성능에 크게 기여
- ML 모델을 “서비스 형태로 연결”하는 과정이 데이터 과제보다 더 많은 구현적 노력이 필요함

개선 방향

- 사용자 데이터 저장(CSV/DB 저장) 기능 추가
- 주간 리포트 자동 생성(High/Medium/Low 비율 시각화)
- KoBERT 기반 문장 분류 모델로 고도화
- 일정 추천 기능(“오늘은 이 3개를 먼저 끝내세요”)
- 모바일 UI 최적화
- 일정 중복 감지 기능