MT엠티데이타	개 발 완 료 보 고 서	페이지 번호	1/26 페이지
		작 성 자	STEEL CUT
		제 출 일 자	2023.12.18

아래와 같이 개발 완료 보고서를 제출합니다.

개 발 과 제	철스크랩 가격 예측 및 철스크랩 가격 동향 분석 웹 플랫폼	
개 발 기 간	2023.10.30 ~ 2023.12.14	
개발 완료 일자	2023.12.14	
결과 발표 일자	2023.12.15	
프로젝트 팀 구성	Project Manager	이수진
	Model Engineer	박소원
	Back-end Developer	김대환
	Front-end Developer	이재필
	Research & Development Assistant	정세윤
개 발 환 경	- Python (Version: 3.9) - Node (Version: 20.9) - NPM (Version: 10.1)	
사용 라이브러리 및 프레임워크	라이브러리: - Pandas - Numpy - Selenium - BeautifulSoup4 - Scikit-Learn - XGBoost - Prophet - Darts - Autogluon - Psycopg2 프레임워크: - Keras (Python) - React (Javascript) - NodeJS (Javascript)	

목차

철 스크랩 가격 예측 웹 플랫폼

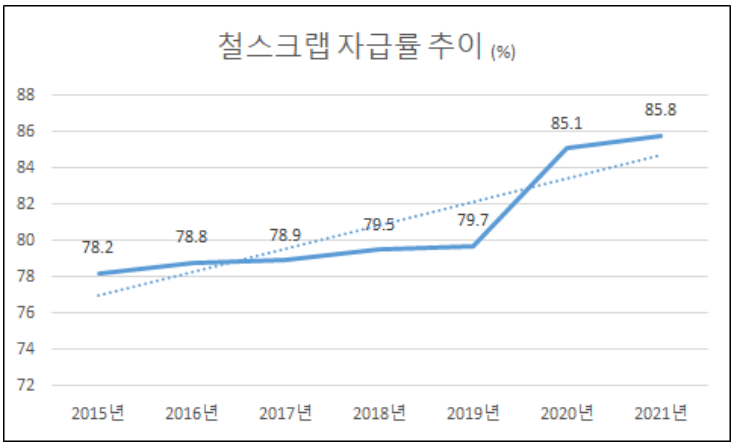
제 1절 개요

○ 1.1 기획 배경 및 동기

- 2015년 유엔 기후 변화 회의에서 채택된 '파리 협정'과, 2023년 3월 21일 대한민국 정부가 발표한 '2050 탄소중립 달성과 녹색성장 실현' 정책 등은 탄소배출 감축을 위한 글로벌 패러다임의 중요성을 강조. 이러한 맥락에서, 철강 산업은 고로를 대체하는 친환경적인 전기로의 활용 가능성을 적극적으로 모색.

- 전기로 제강에서 사용되는 철원 중 95% 이상을 차지하는 철 스크랩은, 고로에 비해 CO2 를 60% 감축하고 폐기물을 80% 줄이는 효과를 보여줌으로써 중요한 천연 자원으로 인식. 이에 따라 철 스크랩에 대한 수요는 지속적으로 증가.

- 이러한 시장 상황 속에서, 탄소 중립 실현을 위한 철강 기업들과 철강 산업 이해관계자들은 철 스크랩을 저비용으로 확보하고, 이를 통해 탄소 배출 감축을 위한 효율적인 방법을 찾아내기 위한 노력을 지속. 이는 탄소중립 목표 달성과 동시에 새로운 이익 창출의 기회를 제공하기에, 이에 대한 솔루션을 제공하고자 본 프로젝트를 진행.



철 스크랩에 대한 수요 추이 (출처: FerroTimes)

○ 1.2 개발 목표 및 방향성

- 본 프로젝트의 주요 목표는 인공지능 모델을 적용한 철 스크랩 가격 예측과 이를 활용한 웹 플랫폼 구축. 이를 통해 이용자들은 철 스크랩의 매입 시기를 적절하게 조절할 수 있게 도움을 받을 것으로 기대.

- 본 프로젝트는 철강 기업과 철강 산업 이해관계자들이 철 스크랩을 저비용으로 매입하고, 이를 통해 다양한 부가 가치를 창출할 수 있는 서비스 개발을 목표로 설정. 이는 저비용으로 철 스크랩을 확보함으로써 기업의 경쟁력을 강화하는 데 기여할 것으로 기대.

- 더불어, 철 스크랩 가격과 관련된 다양한 경제 지표와 철 스크랩의 평균 가격 등의 데이터를 지속적으로 업데이트하며, 이를 통해 철 스크랩과 전반적인 철강 산업에 대한 깊이 있는 인사이트를 제공.

- 결론적으로, 이 프로젝트를 통해 철 스크랩의 가격 예측을 통한 효율적인 매입 전략 수립을 지원하고, 철강 산업에 대한 통찰력을 제공하는 플랫폼을 개발함으로써 철강 산업의 지속 가능성에 기여.

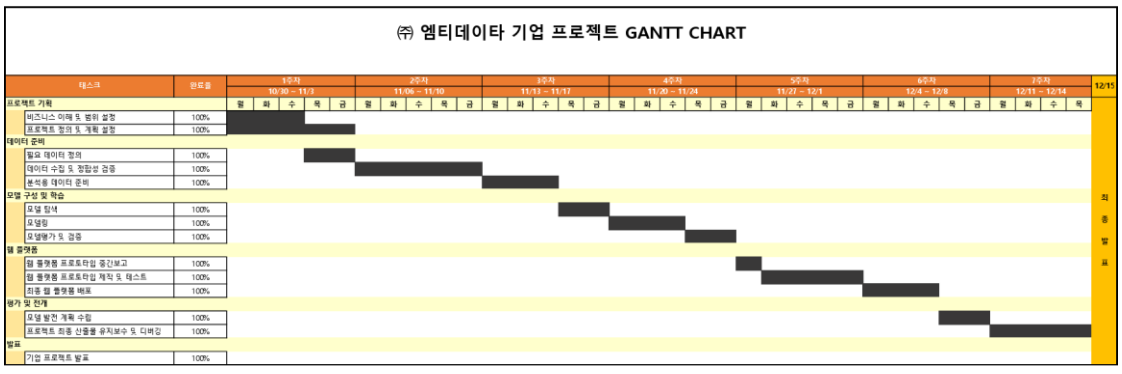
○ 1.3 유사 서비스

- 철 스크랩의 이미지 분류 서비스는 있지만, 철 스크랩 가격을 직접적으로 예측하는 서비스는 아직 없는 것으로 확인. 이는 우리 서비스의 큰 차별점이며, 이를 통해 철 스크랩 가격 예측 서비스를 선도할 수 있음을 시사.

제 2절 프로젝트 기획

○ 2.1 개발 일정 및 진행 과정

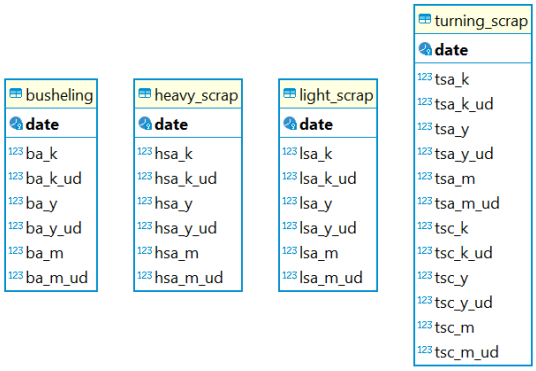
- 본 프로젝트는 2023년 10월 30일에 시작하여 2023년 12월 15일까지 약 1.5개월의 기간 동안 진행. 대부분의 태스크 과제는 2일에서 5일 이내에 완료되었으나, 데이터 수집, 데이터 정의, 모델링과 같은 프로젝트의 핵심적인 단계는 필요에 따라 반복 수행되며, 이에 최대 2주까지 소요. 특히, 모델링 단계에서는 다양한 테스트를 반복적으로 수행하였고, 이 과정은 총 7주차까지 수행. 이를 통해 모델의 성능을 계속적으로 개선하고, 최적의 결과를 도출하려고 노력.



(프로젝트 일정 관리를 위한 간트 차트)

○ 2.2 데이터베이스 구축 및 웹 플랫폼 기획 과정

- 본 프로젝트에서는 PostgreSQL 을 사용하여 데이터베이스를 구축. 엔터티 관계도(ERD)는 각 테이블을 독립적으로 설계하였으며, 철 스크랩의 품목별(생철, 중량 A, 경량 A, 선반 A, 선반 C) 현재가의 날짜를 기본 키(Primary Key)로 선정. 이를 통해 각 품목별 영남 평균가격과 등락, 경기도/인천 평균 가격과 등락, 그리고 전체 평균 가격과 등락을 효과적으로 저장.



(데이터베이스 Entity Relation Diagram)

- 또한, 본 프로젝트는 요구사항 정의서와 기능명세서를 기반으로 웹 플랫폼의 세부 기능을 사전에 기획. 이를 통해 웹 플랫폼의 전반적인 구조를 미리 파악하고, 개발 우선순위와 가능성을 명확히 확인. 이러한 사전 계획을 바탕으로 웹 플랫폼의 제작을 효율적으로 진행.

- 결론적으로, 본 프로젝트는 체계적인 데이터베이스 구축과 세부적인 웹 플랫폼 기획을 통해 철강 가격 예측을 위한 효과적인 인프라를 구축.

요구사항 정의서

0	서비스(메뉴)	기능명	기능설명	우선순위	비고
Data Analysis & Modeling		데이터 크롤링	사전 정의된 타겟변수와 피쳐변수 확보	1	
		데이터 전처리	데이터셋 선형보간, 이상치 처리	1	
		데이터셋 구축	통일된 주기를 바탕으로 최종 데이터셋 구축	1	
		AI 모델링	주단위로 철 스크랩 가격 예측값 도출	1	
Common	Route	홈 html 페이지 라우팅	홈 html 페이지 라우팅하는 홈 버튼	5	
		대시보드 html 페이지 라우팅	대시보드 html 페이지 라우팅 바로가기 버튼	5	
User	Home	금주 철 스크랩 평균 가격 예측 출력	금주 철 스크랩 평균 가격 예측 출력	2	
		생철, 중량A, 경량A, 선반 품목별 선택	각 품목의 가격 예측 값 출력하기 위한 선택 버튼	2	
		자세히 알아보기 버튼	각 품목별 통계 대시보드 출력하는 html 페이지로 라우팅	2	
	Dashboard	동향 대시보드 출력	월단위 철 스크랩 가격, 품목별 동향 등 기본 통계치 출력	3	
		데이터 테이블 출력	데이터 테이블 출력	3	
		테이블 컬럼별 정렬	컬럼별 오름차순/내림차순 정렬	4	
		품목 선택 버튼	품목별 테이블 출력	4	
Admin	Manage	최신 데이터 주기적 동기화	매주 최신 데이터 데이터베이스 갱신	2	
		AI 모델 최신 데이터 반영	최신 데이터 갱신 후 차주 가격 예측	2	

(요구사항 정의서)

㈜엠티데이타 기능명세서

구분	주 기능	상세 기능	기능 설명	가능 여부	비고
1. 홈 화면	1.1 철 스크랩 품목별 시장가	생철, 중량A, 경량A, 선반A, 선반B 시장가 게시	각 품목별 시장가와 전주 대비 등락 파악	가능	
	1.2 철 스크랩 품목별 가격 추세 대시보드	철 스크랩 전체 품목 평균 시장가 및 등락 대시보드 게시	시장가 등락 파악을 위한 대시보드	가능	
	1.3 철 스크랩 각 품목 예측 가격	생철, 중량A, 경량A, 선반A, 선반B 자주 예측 가격 게시	매입 의사결정을 위한 데이블	가능	
	1.4 탄소배출 진행상황 도넛 차트	국내 탄소배출 현재 진행상황을 보여주는 도넛 차트 게시	국내 최신 탄소배출 현황 파악을 위한 도넛 차트	가능	
	1.5 작년 철 스크랩 품목별 분기별 가격 바 차트	작년 품목별 철 스크랩 가격을 노출하는 바 차트 게시	철 스크랩 가격 트렌드 파악을 위한 바 차트	가능	
	1.6 각 나라별 탄소배출량 대표하는 코로플레스 맵	국가별 탄소배출량을 보여주는 코로플레스 맵 게시	탄소배출량 동향 파악을 위한 코로플레스 맵	가능	
2. 상세 분석 페이지	2.1 각 품목별 상세 분석 테이블 (라우팅)	2.1.1 품목별 특성 분석 테이블	생철, 중량A, 경량A, 선반A, 선반B 분석 테이블	가능	
		2.1.2 테이블 정렬 기능	컬럼 값 오름차순/내림차순 정렬	가능	
		2.1.3 테이블 필터 기능	보고 싶은 컬럼의 데이터 테이블 커스텀마이징	가능	
		2.1.4 컬럼 숨기기 기능	필요한 컬럼만 볼 수 있게 정리	가능	
		2.1.5 테이블 전체 선택 기능		가능	
		2.1.6 테이블 노출 건수 조절 기능		가능	
		2.1.7 데이터 테이블 페이지 조절 기능		가능	
	2.2 게시판 페이지 (라우팅)	2.2.1 개발자 정보 게시		가능	
		2.2.2 캘린더 일정표 게시	일정 및 휴일 입력, 날짜 조절, 페이지 조절 기능	가능	
		2.2.3 FAQ 페이지 게시	자주 묻는 질문 게시판	가능	
	2.3 동향 분석 상세 차트 페이지 (라우팅)	2.3.1 작년 품목별 철 스크랩 가격 바 차트 상세 분석 게시	마우스 호버시 추가 정보 제공 기능	가능	
		2.3.2 가격과 연관성 상위 5개 특성 상세 파이 차트 게시	마우스 호버시 추가 정보 제공 기능	가능	
		2.3.3 가격 동향 상세 분석 선 그래프 게시	마우스 호버시 추가 정보 제공 기능	가능	
		2.3.4 나라별 탄소배출량 코로플레스 맵 상세 분석 게시	마우스 호버시 추가 정보 제공 기능	가능	

(기능 명세서)

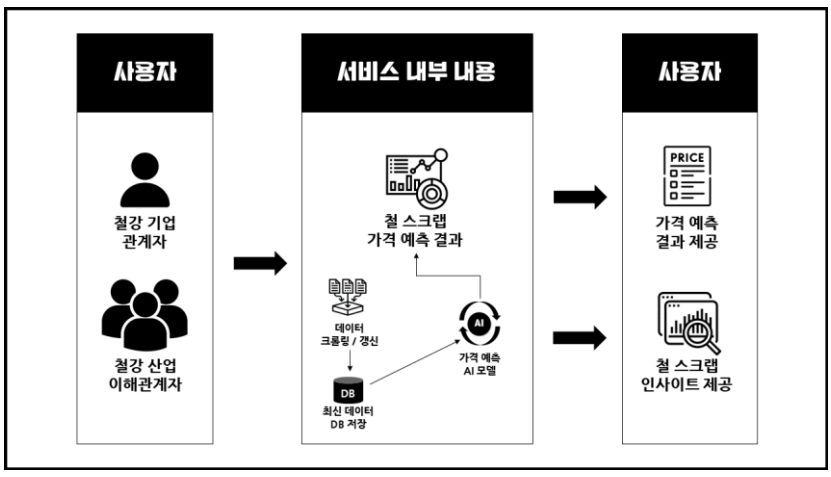
○ 2.3 시스템 아키텍쳐 설계 및 구현

- 본 프로젝트는 철강 기업 관계자와 철강 산업 이해관계자들을 주요 타겟으로 설정하고, 이들의 요구사항에 맞게 플랫폼을 개발. 또한, 데이터 수집부터 모델링까지의 과정을 효율적으로 진행하기 위한 파이프라인을 설계.

- 시간과 자원의 효율성을 고려하여, 파이프라인 실행은 공식적으로 정해진 점검 시간에 수행. 점검 시간은 매주 갱신되는 철 스크랩 가격 종속변수에 맞춰 일요일 오후 7 시로 설정. 해당 시간에 데이터를 최신화하고, 이를 데이터베이스에 저장하고, 모델링을 진행하는 파이프라인을 구성.

- 이러한 시스템 아키텍처를 통해, 본 프로젝트의 산출물은 매주 월요일부터 일요일 오후 6 시 59 분까지 타겟 이용자들에게 철 스크랩 가격 예측 결과를 제공. 더불어, 철 스크랩에 대한 다양한 인사이트를 제공함으로써 이용자들의 의사결정을 지원.

- 결론적으로, 체계적인 시스템 아키텍처 구축을 통해 철 스크랩 가격 예측 모델의 효과적인 운영이 가능.



(전반적인 시스템 아키텍쳐 시각화)

제 3절 모델링

○ 3.1 수집 데이터 정의

데이터 정의 요약 테이블

데이터 변수명	변수 종류	데이터 종류	기간 단위
철 스크랩 품목 가격: 생철	종속변수	정형 데이터	주 단위
철 스크랩 품목 가격: 중량A	종속변수	정형 데이터	주 단위
철 스크랩 품목 가격: 경량A	종속변수	정형 데이터	주 단위
철 스크랩 품목 가격: 선반A	종속변수	정형 데이터	주 단위
철 스크랩 품목 가격: 선반C	종속변수	정형 데이터	주 단위
국제 유가	독립변수	정형 데이터	일 단위
무역수지	독립변수	정형 데이터	월 단위
금 가격	독립변수	정형 데이터	일 단위
수입물가지수	독립변수	정형 데이터	주 단위
철광석 수입 금액	독립변수	정형 데이터	주 단위
철광석 수입 물량	독립변수	정형 데이터	주 단위
금속광물 수입 금액	독립변수	정형 데이터	주 단위
금속광물 수입 물량	독립변수	정형 데이터	주 단위
환율: 달러	독립변수	정형 데이터	일 단위
환율: 엔	독립변수	정형 데이터	일 단위
환율: 위안	독립변수	정형 데이터	일 단위
환율: 유로	독립변수	정형 데이터	일 단위
경기 선행지수: G20	독립변수	정형 데이터	주 단위
경기 선행지수: 한국	독립변수	정형 데이터	주 단위
경기 선행지수: 미국	독립변수	정형 데이터	주 단위
경기 선행지수: 중국	독립변수	정형 데이터	주 단위
전기요금 평균 단가	독립변수	정형 데이터	월 단위

- 본 프로젝트에서는 철 스크랩의 다섯 가지 품목인 생철, 중량 A, 경량 A, 선반 A, 그리고 선반 C 의 평균 가격을 종속 변수로 선정. 이들 평균 가격은 날짜를 기반으로 한 현재가를 의미하며, 추세성을 보이는 시계열 데이터로 정의.

- 또한, 철 스크랩 가격과 상관성을 가질 것으로 예상되는 다양한 지표들을 독립 변수로 선정.

<u>독립 변수의 목록:</u>

- 국제유가(Dubai)
- 무역수지
- 금 가격
- 수입물가지수
- 철광석 수입 금액, 철광석 수입 물량
- 금속광물 수입 금액, 금속광물 수입 물량
- 환율: 달러, 엔, 유로, 위안

- 경기 선행지수: G20, 한국, 미국, 중국
- 전기요금 월별 평균단가

- 따라서, 정의된 독립변수에 따라 철 스크랩 가격 예측에 필요한 다양한 종류의 데이터를 체계적으로 수집하고 분석. 이를 통해 철 스크랩 가격에 영향을 미치는 요인들에 대한 깊이 있는 이해를 도출.

○ 3.2 데이터 수집 – 종속변수

- 종속변수 데이터 수집을 위해 파이썬의 Selenium 라이브러리를 활용하여 동적 크롤링을 통해 데이터를 수집. 종속 변수인 철 스크랩 평균 가격은 유료 사이트인 '스틸데일리'(www.steeldaily.co.kr)에서 수집하였으며, 총 892건의 데이터를 확보. 이는 2006년 8월 4일부터 2023년 12월 5일까지의 주 단위 데이터를 포함.

```
# 창 열기
service = Service(executable_path='./data/chromedriver.exe')
# options=webdriver.ChromeOptions()
# driver=webdriver.Chrome(service=service,options=options)
driver=webdriver.Chrome()
driver.get("https://www.steeldaily.co.kr/")
driver.implicitly_wait(5) # 창 오픈까지 최대 10초를 기다려줌
time.sleep(1.5)
# ----------------------------------------------------------------------------------
# 로그인 버튼 클릭
driver.find_element(By.CSS_SELECTOR, '#userLogin > li:nth-child(2) > a').click()
time.sleep(1.5)
# ----------------------------------------------------------------------------------
# 아이디 및 비밀번호 입력 후 클릭
driver.find_element(By.CSS_SELECTOR, '#user_id').send_keys('kdt40')
time.sleep(1.5)

driver.find_element(By.CSS_SELECTOR, '#user_pw').send_keys('steelcut4')
time.sleep(1.5)

driver.find_element(By.CSS_SELECTOR, '#loginForm > button').click()
time.sleep(1.5)
# ----------------------------------------------------------------------------------
# 구독만료일 관련 안내 확인
driver.find_element(By.CLASS_NAME, 'button.nd-gray.expanded').click()
time.sleep(1.5)
# ----------------------------------------------------------------------------------
# DB 센터 클릭
secline=driver.find_elements(By.CLASS_NAME, "secline")
DB_senter=secline[7]
DB_senter.click()
time.sleep(1.5)
```

(Selenium 라이브러리 활용한 종속변수 철 스크랩 품목별 평균가격 크롤링)

- 추가적으로, 무료 사이트인 '페로타임즈'(www.ferrotimes.com)에서도 철 스크랩 평균 가격 데이터를 수집. 이 사이트에서의 데이터 수집은 Selenium 라이브러리와 OCR 라이브러리를 활용하여 이미지 텍스트 추출을 진행.

```
# 이미지 URL 추출
img_url = image.get_attribute('src')
time.sleep(2)

with urllib.request.urlopen(img_url) as url:
    image = Image.open(url)

# Tesseract 경로 설정
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

text = pytesseract.image_to_string(image, lang='kor', config='--oem 1 --psm 3')
```

(OCR 라이브러리 활용한 종속변수 철 스크랩 품목별 평균가격 크롤링)

- 그러나 '페로타임즈'에서의 데이터 활용은 세 가지 이유로 기각. 첫째, 최신 데이터의 부재로 인해 이 프로젝트의 목표인 지속적인 데이터 및 모델 최신화의 부재. 둘째, 데이터 컬럼의 불명확한 명시의 문제로 인해 객관적인 사실보다는 주관적 판단에 의존한 데이터 수집의 위험이 존재. 셋째, 이미지 인식의 문제로 인해 데이터의 누락 가능성이 존재. 이러한 이유로 인해 '페로타임즈'에서의 데이터 활용은 배제.

	구분	1개월	2개월	3개월	6개월	12개월	15개월	년도	월	일
0	2023-06-30	384.2	374.0	372.5	376.0	380.5	381.5	2023	6	30
1	2023-06-29	383.5	375.0	375.0	378.0	381.0	380.5	2023	6	29
2	2023-06-28	384.5	375.0	373.0	378.0	381.0	379.5	2023	6	28
3	2023-06-27	384.0	379.0	379.0	381.5	385.0	386.0	2023	6	27
4	2023-06-26	384.0	379.0	378.0	378.0	384.5	386.0	2023	6	26
...	...	...	...	...	...	...	...	...	...	...
401	2021-10-18	481.0	500.0	497.0	482.5	444.0	434.0	2021	10	18
402	2021-10-15	480.5	499.5	496.5	479.5	444.0	434.0	2021	10	15
403	2021-10-14	480.5	495.0	495.0	473.5	436.0	423.5	2021	10	14
404	2021-10-13	475.0	484.0	480.0	466.0	429.5	419.5	2021	10	13
405	2021-09-30	439.8	445.0	442.0	438.0	422.0	416.0	2021	9	30

406 rows × 10 columns

("1개월", "2개월"의 컬럼들이 평균가격 혹은 최소 가격을 의미하는지 등 데이터 단위에 대한 모호함이 존재)

- 따라서, 본 프로젝트는 '스틸데일리'에서 수집한 철 스크랩 평균 가격 데이터를 주로 활용. 이를 통해 철 스크랩 가격 예측 모델의 효과적인 구축을 위한 기반을 마련.

○ 3.3 데이터 수집 – 독립변수

- 독립 변수 데이터는 일, 주, 월 단위의 데이터로 수집. 데이터 수집은 소스 사이트에서 제공하는 csv, xlsx 형식의 파일 다운로드 및 크롤링을 통해 진행.

- 이러한 데이터 수집 방식은 초기 모델링을 위한 임시 방편으로, 최신 독립 변수 데이터의 수집은 상관성, 피처 중요도, 모델 정확도 등 다양한 요인을 고려하여 모델링 과정 이후에 진행할 계획. 이를 위해, 본 프로젝트는 임시 데이터셋을 구축하였고, 최종 데이터셋에 활용될 독립 변수들의 최신 데이터를 수집하기 위해 Selenium 과 BeautifulSoup4 를 활용한 자동화된 데이터 수집 파이프라인의 개발 계획을 수립.

생철(A)(Busheling) 경인	경인 등락	생철(A)(Busheling) 영남	영남 등락	생철(A)(Busheling) 평균	평균 등락	달러 환율	엔 환율	유로 환율	...	철광석 수입물량	무역수지	철광석 가격	G20	한국	미국	중국	수입물가지수	국제유가(Dubai, 최속)	금(우속)
230.0	-0	240.000000	-0	235.000000	-0	966.300000	839.020000	1235.800000	...	89.987097	427.935484	46.640121	101.084706	100.249032	101.169295	101.158790	46.916568	65.317151	624.430387
230.0	NaN	239.857143	NaN	235.000000	NaN	965.666667	840.903333	1238.076667	...	91.236129	484.580645	46.640143	101.087176	100.251398	101.170822	101.161622	46.911912	65.067589	623.433631
230.0	NaN	239.714286	NaN	235.000000	NaN	965.033333	842.786667	1240.353333	...	92.485161	541.225806	46.640162	101.089713	100.253773	101.172411	101.164571	46.905695	64.806999	622.420768
230.0	NaN	239.571429	NaN	235.000000	NaN	964.400000	844.670000	1242.630000	...	93.734194	597.870968	46.640178	101.092315	100.256156	101.174063	101.167637	46.898035	64.536350	621.394200
230.0	NaN	239.428571	NaN	235.000000	NaN	961.500000	835.290000	1232.550000	...	94.983226	654.516129	46.640192	101.094982	100.258545	101.175776	101.170820	46.889048	64.256607	620.356328
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
460.0	NaN	477.285714	NaN	469.000000	NaN	1325.733333	905.626667	1431.126667	...	123.277097	3264.258065	115.058590	100.153573	99.249974	99.415722	101.900664	166.496150	92.157582	1856.858479
460.0	NaN	477.142857	NaN	469.000000	NaN	1326.600000	905.000000	1431.930000	...	123.579677	3350.806452	115.363093	100.157349	99.252025	99.418993	101.911632	167.490793	92.333194	1854.789330
460.0	-0	477.000000	▼ -1	469.000000	-0	1322.500000	903.130000	1431.800000	...	123.882258	3437.354839	115.532735	100.161120	99.254061	99.422261	101.922618	168.512048	92.499142	1852.919561
460.0	NaN	477.000000	NaN	468.857143	NaN	1322.000000	906.690000	1438.860000	...	124.184839	3523.903226	115.616266	100.164885	99.256084	99.425529	101.933624	169.560368	92.654526	1851.261518
460.0	NaN	477.000000	NaN	468.714286	NaN	1321.400000	905.070000	1444.360000	...	124.487419	3610.451613	115.662438	100.168645	99.258093	99.428795	101.944651	170.636201	92.798446	1849.827549

columns

(모델링을 위한 임시 데이터셋 구축)

○ 3.4 데이터 전처리 과정

- 데이터 병합을 원활하게 진행하기 위해, 철 스크랩 품목의 평균 가격 날짜를 datetime 타입으로 형변환하여 데이터셋의 인덱스로 구성. 이어서, 종속변수와 각 독립변수들을 주 단위 데이터에서 일 단위 데이터로 리샘플링 진행.

- 데이터 리샘플링 과정에서 발생한 결측치는 선형보간 기법을 활용하여 대치. 이를 통해 기존 892 건의 데이터를 6237 건으로 증강.

- 따라서 이러한 체계적인 전처리 과정을 통해, 본 프로젝트는 철 스크랩 가격 예측 모델의 효과적인 구축을 위한 탄탄한 기반을 마련. 이를 통해 철 스크랩 가격에 영향을 미치는 요인들에 대한 보다 깊이 있는 이해도 도출.

```python
def pre_processing_3(df):
    # 날짜컬럼 시계열화
    df['날짜'] = pd.to_datetime(df['날짜'])

    # 날짜를 인덱스로 변환.
    df.set_index('날짜', inplace=True)

    # 날짜 인덱스를 일별로 리샘플링.
    df = df.asfreq('D')

    # 결측치를 선형 보간 (method ->  'linear', 'polynomial', 'spline' 도 가능)
    # df.interpolate(method='linear', inplace=True)
    df.interpolate(method='polynomial', order=3, inplace=True)
    df = df.reset_index()

    # 2023년 12월 5일 이전의 데이터만 선택
    # 2006년 8월 4일 이후의 데이터만 선택
    df = df[df['날짜'] <= '2023-12-05']
    df = df[df['날짜'] >= '2006-08-04']
    df = df.reset_index(drop=True)

    return df
```

(날짜 인덱스 변환, 일 단위 리샘플링, 선형보간 데이터 전처리 함수)

○ 3.5 모델링 과정

3.5.1 XGBoost, CatBoost, RandomForest Regressor 앙상블 모델

3.5.1 모델링 테스트 요약 테이블

모델명	사용 독립변수	RMSE	결정계수
3.5.1 XGBoost 1회	전체 변수 (전기 요금 제외)	162	-1.10
3.5.1 XGBoost 2회	전체 변수 (전기 요금 포함)	138	-0.40
3.5.1 XGBoost Voting 1회	전체 변수 (전기 요금 포함)	57	-1.06
3.5.1 CatBoost Voting 2회	전체 변수 (전기 요금 포함)	57	-6.53
3.5.1 RandomForest 1회	'위안 환율', '금속광물 수입금액', '철광석 수입물량'	100	0.22
3.5.1 XGBoost 3회	'위안 환율', '금속광물 수입금액', '철광석 수입물량'	90	0.35
3.5.1 RandomForest 2회	'위안 환율', '금속광물 수입금액', '무역수지'	97	0.26
3.5.1 XGBoost 4회	'위안 환율', '금속광물 수입금액', '무역수지'	110	0.05
3.5.1 RandomForest 3회	'위안 환율', '금속광물 수입금액', '금 가격'	100	0.22
3.5.1 XGBoost 5회	'위안 환율', '금속광물 수입금액', '금 가격'	97	0.26
3.5.1 RandomForest 4회	'위안 환율', '무역수지', '금 가격'	96	0.27
3.5.1 XGBoost 6회	'위안 환율', '무역수지', '금 가격'	90	0.37
3.5.1 RandomForest 5회	'금속광물 수입금액', '철광석 수입금액', '무역수지'	90	0.36
3.5.1 XGBoost 7회	'금속광물 수입금액', '철광석 수입금액', '무역수지'	90	0.36
3.5.1 RandomForest 6회	'금속광물 수입금액', '철광석 수입물량', '금 가격'	101	0.20
3.5.1 XGBoost 8회	'금속광물 수입금액', '철광석 수입물량', '금 가격'	100	0.22
3.5.1 RandomForest 7회	'금속광물 수입금액', '무역수지', '금 가격'	106	0.11
3.5.1 XGBoost 9회	'금속광물 수입금액', '무역수지', '금 가격'	90	0.36
3.5.1 RandomForest 8회	'철광석 수입금액', '무역수지', '금 가격'	96	0.29
3.5.1 XGBoost 10회	'철광석 수입금액', '무역수지', '금 가격'	95	0.30
3.5.1 RandomForest 9회	'위안 환율', '금속광물 수입금액', '철광석 수입금액', '무역수지'	113	0.005
3.5.1 XGBoost 11회	'위안 환율', '금속광물 수입금액', '철광석 수입금액', '무역수지'	112	0.14
3.5.1 RandomForest 10회	'금속광물 수입금액', '철광석 수입금액', '무역수지', '금 가격'	111	0.03
3.5.1 XGBoost 12회	'금속광물 수입금액', '철광석 수입금액', '무역수지', '금 가격'	96	0.28
3.5.1 RandomForest 11회	'위안 환율', '금속광물 수입금액', '철광석 수입금액', '철광석 수입물량', '무역수지'	111	0.04
3.5.1 XGBoost 13회	'위안 환율', '금속광물 수입금액', '철광석 수입금액', '철광석 수입물량', '무역수지'	113	0.12

- 프로젝트의 최종 과정에서는 일부 LTSF 시계열 데이터셋에서 SOTA를 달성한 최신 모델인 'DLinear' 모델을 채택. 초기 모델링 과정에서는 XGBoost Regressor, Random Forest Regressor, 그리고 LSTM의 모델을 사용하였으나, 모델의 낮은 정확도가 도출. 이는 데이터량의 부족, 불충분한 특성공학, 그리고 모델의 부적합성 등 다양한 원인이 복합적으로 작용된 것으로 판단.

- 초기 모델링 과정에서는 XGBoost Regressor 모델을 주로 사용하였으나, 이상치 제거 및 관련 컬럼의 추가 등의 작업을 진행한 뒤에도 모델의 정확도는 크게 향상되지 않았고 오히려 계속해서

음수 값의 결정계수가 도출. 이에 따라 RandomForest Regressor까지 테스트해보았으나, 결과는 마찬가지로 기대에 못 미치는 성능을 사전에 파악.

```
# 테스트 데이터 셋 점수
test_score = model.score(test_x, test_y)
print(f"Test Score: {test_score}")
```

```
Test Score: -0.04137894769597206
```

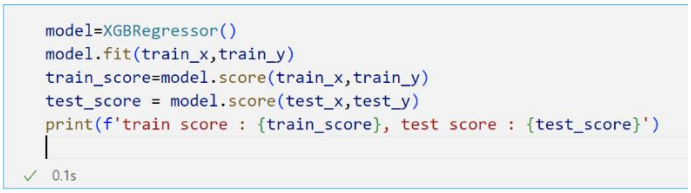
(결정계수가 계속 음수 값이 도출되면서 예측이 전혀 안되고 있는 상황이 발생)

- 이후 과정에서는 상관계수가 0.5 이상인 독립변수를 활용하여 특성 공학을 진행. 그러나 모델 낮은 정확도는 계속해서 유지됨을 확인.

```
df_corr[(df_corr['선반A (Turning Scrap A) 평균']>0.5)&(df_corr['선반A (Turning Scrap A) 평균']!=1)].index
```

```
Index(['금속광물 수입금액', '철광석 수입금액', '국제유가(Dubai, 좌축)', '금(우축)'], dtype='object')
```

(상관계수 0.5 이상인 독립변수들을 피쳐로 활용)



```
model=XGBRegressor()
model.fit(train_x,train_y)
train_score=model.score(train_x,train_y)
test_score = model.score(test_x,test_y)
print(f'train score : {train_score}, test score : {test_score}')
```

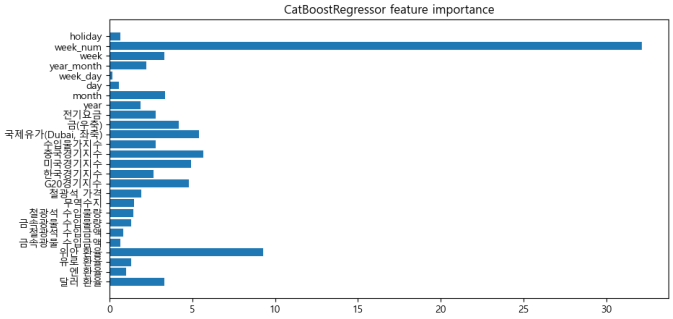
✓ 0.1s

```
train score : 0.9999786694312872, test score : -2.3860913307409723
```

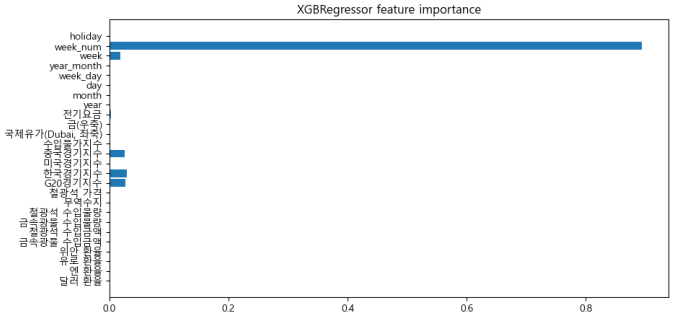
(결정계수 음수 값과 높은 RMSE 값 도출)

- 앙상블 기법의 머신러닝 모델에선 다양한 특성들을 활용하여 모델링을 추가적으로 진행. 특히, 철 스크랩은 전기로를 이용하는 원자재이므로 전기 요금을 독립변수로 입력하고 환율, 수출입 금액, 수출입 물량, 원자재 가격, 원자재 물량 등의 데이터를 추가. 또한, 년, 월, 일, 누적 주차, 요일, 그리고 년-월을 묶은 데이터를 파생변수로 생성하여 인코딩을 진행.

- 이후, CatBoost와 XGBoost의 회귀모델을 Voting 방식으로 앙상블 학습을 진행하였으나, 모델의 RMSE 값은 57점 정도로, 결정계수는 음수(-1.06, -6.53)로 도출. 하지만 피쳐 중요도를 추출하는 과정에서는 피쳐에 관한 중요한 인사이트를 확보. CatBoost 모델에서는 누적 주차를 중요한 특성으로, XGBoost 모델에서는 추세나 계절성을 반영한다는 점을 확인. 이를 통해, 시계열 학습모델에 데이터를 적용하는 것이 유용하다는 결론을 도출.



(Catboost 피쳐 중요도 결과)



(XGboost 피쳐 중요도 결과)

- 이후엔 추가적으로 여러 조합의 독립변수들을 활용한 모델 테스트를 진행. 결과적으로 3개 이상의 독립변수만을 사용했을 때 결정계수를 양수 값으로 도출. 하지만 평균적인 RMSE 값이 매우 높고 결정계수 또한 0.4 이하로 매우 낮은 점수를 확보. 또한 4개 이상의 독립변수 적용시 오히려 결정계수 점수가 현저히 떨어져, 결론적으로 해당 모델링 방식을 채택하지 않기로 결론을 도출.

3.5.2 shift 함수 사용 앙상블 모델

3.5.2 모델링 테스트 요약 테이블

모델명	사용 독립변수	RMSE	결정계수
3.5.2 RandomForest 1회	'위안 환율', '금속광물 수입금액'	90	0.34
3.5.2 RandomForest 2회	'달러 환율', '위안 환율', '금속광물 수입금액'	93	0.30
3.5.2 RandomForest 3회	'달러 환율', '위안 환율', '금속광물 수입금액', '철광석 수입금액'	101	0.18
3.5.2 XGBoost 1회	'위안 환율', '금속광물 수입금액'	91	0.32
3.5.2 XGBoost 2회	'달러 환율', '위안 환율', '금속광물 수입금액'	100	0.19
3.5.2 XGBoost 3회	'달러 환율', '위안 환율', '금속광물 수입금액', '철광석 수입금액'	107	0.08

- 피쳐 중요도에서 도출한 인사이트를 토대로 다시 XGBoost와 RandomForest를 활용한 모델링을 진행. 이 과정에서 시계열 데이터에 맞게 pandas의 shift() 함수를 사용하여 시계열성을 적용하고

파라미터 값은 63일로 설정.

```
shift_best_feature = ['달러 환율', '위안 환율', '금속광물 수입금액', '철광석 수입금액','미국']
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from itertools import combinations
print('-'*100)
filename='best_shift_rf_xgb_result'

with open(f'./1204_1208/modeling_result/{filename}.txt', 'w') as file_data:
        file_data.write('-'*100+'\n')

for cnt in range(1,len(shift_best_feature)+1):
    # 조합을 이용하여 모든 경우의 수 계산
    for i in combinations(all_df[shift_best_feature].columns,cnt):
        # 독립변수, 종속변수 분리
        feature = all_df[list(i)].shift(63).dropna()
        target = all_df['선반A (Turning Scrap A) 평균'].iloc[63:]
        # train, test 분리
        train_x=feature.iloc[:round(len(feature)*0.8),:]
        test_x=feature.iloc[round(len(feature)*0.8):,:]
        train_y=target.iloc[:round(len(feature)*0.8)]
        test_y=target.iloc[round(len(feature)*0.8):]
        # print(train_x.shape, test_x.shape, train_y.shape, test_y.shape)
        modeling_all(train_x, train_y, test_x, test_y, filename)
```

(함수 shift(63) 적용)

- Shift(63) 함수 사용 결과, 3개 이상의 독립변수 사용했을 때 나오는 '달러 환율', '위안 환율', '금속광물 수입금액' 피처가 존재. 다만 시계열성을 적용한 방식 또한 성능이 떨어져, 해당 모델링 방식을 채택하지 않기로 결정.

3.5.3 LSTM (Long Short-Term Memory) 모델

3.5.3 모델링 테스트 요약 테이블

모델명	사용 독립변수	RMSE	결정계수
3.5.3 LSTM 1회	달러 환율, 금속광물 수입금액, 국제유가, 금 가격	93.07	-0.17
3.5.3 LSTM 2회	달러 환율, 금속광물 수입금액, 철광석 가격, 금 가격	96.37	-0.25
3.5.3 LSTM 3회	달러 환율, 금속광물 수입금액, 철광석 수입금액, 금 가격	97.06	-0.27
3.5.3 LSTM 4회	위안 환율, 금속광물 수입금액, 철광석 가격, 금 가격	99.93	-0.35
3.5.3 LSTM 5회	달러 환율, 금속광물 수입금액, 금속광물 수입물량, 금 가격	101.29	-0.38
3.5.3 LSTM 6회	금속광물 수입금액, 철광석 수입물량, 금 가격	101.58	-0.39

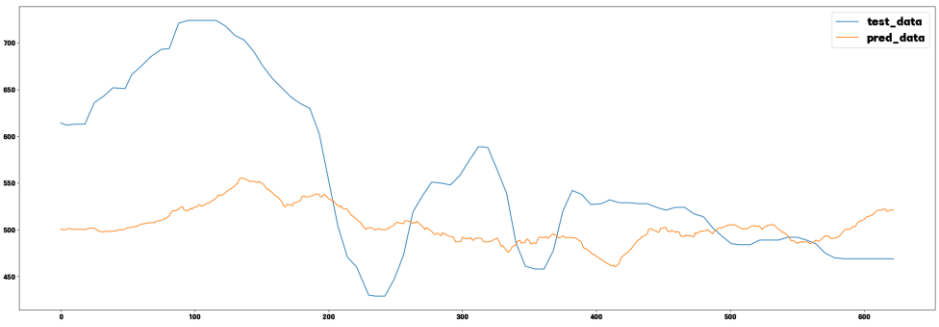
- 모델 추가 탐색 단계 전, 마지막으로 딥러닝의 LSTM 모델을 테스트. 이 과정에서는 상관성이 낮은 독립변수들을 제거하고, 상관성이 높은 독립변수들을 모델링에 활용.

- 특히, 달러 환율, 금속 광물 수입 금액, 국제유가, 금 가격 등의 독립변수를 사용하였을 때 상대적으로 가장 좋은 성능 도출. 그러나 이는 어디까지나 상대적인 것으로, 전반적인 성능은 여전히 부족하였으며, 결정 계수들 또한 음수 값을 유지.

feature	rmse	r2
('달러 환율', '금속광물 수입금액', '국제유가(Dubai, 좌축)', '금(우...	93.070935	-0.169168
('달러 환율', '금속광물 수입금액', '철광석 가격', '금(우축)')	96.365388	-0.253403
('달러 환율', '금속광물 수입금액', '철광석 수입금액', '금(우축)')	97.061768	-0.271584
('위안 환율', '금속광물 수입금액', '철광석 가격', '금(우축)')	99.931453	-0.347886
('달러 환율', '금속광물 수입금액', '금속광물 수입물량', '금(우축)')	101.285405	-0.384658
('금속광물 수입금액', '철광석 수입물량', '금(우축)')	101.581438	-0.392764

(LSTM 모델링 시 결과, 결정계수들은 음수 값 도출)

- 또한, 선별된 독립변수를 사용하고 하이퍼파라미터 튜닝을 진행한 결과, 예측 성능이 전혀 향상되지 않는 상황이 계속해서 발생. 이에 따라, LSTM 모델 역시 채택하지 않기로 결정.



(LSTM 모델링시 전혀 예측을 못하고 있는 그래프)

○ 3.6 모델 추가 탐색 - Prophet

- 모델 추가 탐색 과정에서는 시계열 데이터에 높은 성능을 보이는 Prophet 라이브러리와 Darts 라이브러리의 DLinear 모델을 대상으로 진행. 이 두 모델에 대한 테스트를 다방면으로 수행.

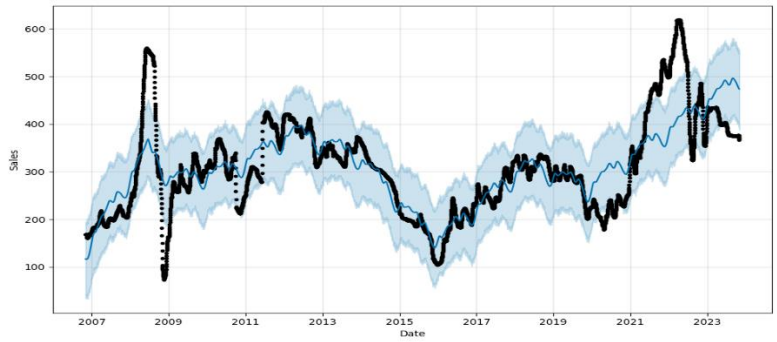
- Prophet 모델의 경우, 날짜와 가격만을 이용한 학습이 가능. 처음에는 어떠한 하이퍼파라미터 튜닝 없이 모델링을 진행하였고, 이때 RMSE 값이 61.19로 나타나, 비교적 긍정적인 전망을 파악. 하지만 시각화를 통해 확인해본 결과, 예측값(파란선)이 실제값(검은선)의 등락을 따라가지 못하는 현상이 발견하고 이는 모델 조정의 필요성을 시사.

```
from sklearn.metrics import mean_absolute_error, mean_squared_error
# mae = mean_absolute_error(test['y'], predictions)
mse = mean_squared_error(base_df.y, forecast.yhat)
rmse = mse ** 0.5
rmse
```

✓ 0.0s

```
61.19284482257656
```

(하이퍼파라미터 튜닝 없이 RMSE 61로 긍정적인 결과 도출)



(그러나 예측값(파란선)이 실제값(검은선)의 등락을 잘 따라가질 못하는 현상을 발견)

- 테스트 결과, 100일 간격으로 학습을 진행했을 때 가장 좋은 결과가 도출. 이는 RMSE 10 이하의 결과를 79%의 확률로 얻을 수 있다는 것을 의미. 하지만 Prophet 모델의 경우 날짜와 가격 변수만으로 모델링을 진행하기 때문에, 특정 경제지표가 비정상적인 변동성을 보일 경우, 모델 또한 매우 낮은 정확도를 보일 가능성이 있음을 확인. 이에 따라 다른 모델의 추가적인 탐색 진행.

```
# 100일 간격으로 일주일 이후 데이터 예측
for i in range(1,100):
    base_df = base_df.sort_values('ds')
    model = Prophet()
    model.fit(base_df[len(base_df)-90-1*7:-1*7])
    future = model.make_future_dataframe(periods = 7)
    forecast = model.predict(future)
    # figure = model.plot(forecast, xlabel = 'Date', ylabel = 'Sales')
    # figure2 = model.plot_components(forecast)
    if i == 1 :
        pred_df=forecast[['ds','yhat']].iloc[-1:]
    else:
        l_df=forecast[['ds','yhat']].iloc[-1:]
        pred_df=pd.concat([pred_df,l_df])
pred_df=pred_df[['ds','yhat']].rename({'ds':'날짜','yhat':'예측값'},axis=1)
comparison=pd.merge(pred_df, result_df, on='날짜', how="left")
comparison['오차']=comparison['선반A (Turning Scrap A) 평균']-comparison['예측값']
# print(pred_df)
print(f'간격 100일 : 오차가 10 미만인 확률은 {sum(comparison.오차<10)/len(comparison.오차)*100:2.0f}%입니다.')
```

(100일 간격으로 데이터 학습 진행)

```
14:42:28 - cmdstanpy - INFO - Chain [1] start processing
14:42:28 - cmdstanpy - INFO - Chain [1] done processing
14:42:29 - cmdstanpy - INFO - Chain [1] start processing
14:42:29 - cmdstanpy - INFO - Chain [1] done processing
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
간격 100일 : 오차가 10 미만인 확률은 79%입니다.
```

(RMSE 오차가 10미만인 확률은 79%로 도출하였으나 라이트 모델에 대한 리스크가 존재)

○ 3.7 모델 추가 탐색 및 모델 채택 – DLinear

- Prophet 모델과 함께, LTSF(Long-term Time Series Forecasting) 시계열 데이터에서 SOTA(State-Of-The-Art)를 달성한 DLinear 모델에 대한 테스트를 진행. DLinear 모델이 포함된 Darts 라이브러리는 파이썬에서 import만으로 손쉽게 사용할 수 있으며, Scikit-Learn과 같이 fit()과 predict() 함수를 이용하기 때문에 사용자 친화적 장점이 존재. 또한, 날짜와 가격 변수만을 사용하는 Prophet

모델과 달리, Autogluon의 DLinear 모델은 다양한 독립변수를 입력하여 모델링할 수 있다는 장점이 있음을 확인.

- 우선, 날짜와 가격 변수만을 사용하여 Darts 라이브러리의 DLinear 모델 테스트 진행. 이때, 예측 기간은 7일(일주일)로 선정. 15일, 30일 등의 예측 값 확인 시 오차 범위가 너무 커져, 가장 정확도가 높은 7일을 예측하는 모델을 채택.

- 학습 기간 선정 시에는 5000일 ~ 200일까지 50일 단위로 예측 모델링을 진행. 그 결과, 2850일 동안 학습했을 때가 RMSE 기준으로 가장 성능이 우수한 것으로 도출. 이 모델로 7일을 100회씩 모델링하여 총 700일을 예측한 결과, RMSE는 8.25로 도출되었고, 편차의 최솟값은 0.0023, 최대값은 44.5005, 평균 값은 5.2306으로 파악. 이를 통해 DLinear 모델이 본 프로젝트에 가장 적합한 모델임을 확인.

```
for n in tqdm(list(range(693,-1,-7))):

    # 2850일로 7일 예측하기
    train = series[-2857-n:-7-n]

    # DLinear 모델
    model = DLinearModel(input_chunk_length=128, output_chunk_length=28, batch_size=256, n_epochs=100)
    model.fit(train)

    pred = model.predict(7)
    pred = list(pred.values().flatten())

    pred_list += pred
```

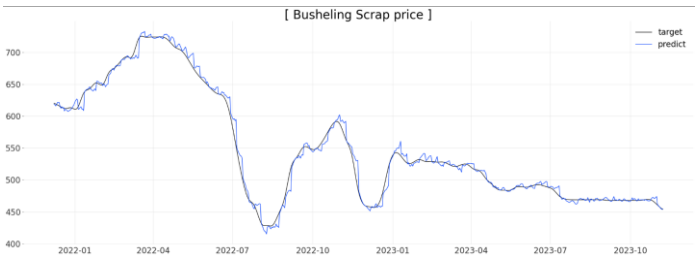
(예측 기간 7일, 일주일로 선정)

```
days_700['pred'] = pred_list
days_700['diff'] = abs(days_700['avg'] - days_700['pred'])
days_700['RMSE'] = RMSE
days_700['total_RMSE'] = mean_squared_error(days_700['avg'], days_700['pred']) ** 0.5
days_700
```

	date	avg	pred	diff	RMSE	total_RMSE
0	2021-12-08	620.335900	618.791616	1.544284	3.339297	8.25089
1	2021-12-09	619.659584	617.658511	2.001073	3.339297	8.25089
2	2021-12-10	618.968345	615.962840	3.005505	3.339297	8.25089
3	2021-12-11	618.259475	618.486311	0.226836	3.339297	8.25089
4	2021-12-12	617.530266	621.477829	3.947563	3.339297	8.25089
...	...	...	...	...	...	...
695	2023-11-03	458.143812	458.372422	0.228611	0.614709	8.25089
696	2023-11-04	456.945948	456.802174	0.143774	0.614709	8.25089
697	2023-11-05	455.837689	455.001367	0.836322	0.614709	8.25089
698	2023-11-06	454.846539	454.018552	0.827987	0.614709	8.25089
699	2023-11-07	454.000000	454.877941	0.877941	0.614709	8.25089

700 rows × 6 columns

(RMSE: 8.25089 / 편차 최솟값: 0.0023 / 편차 최대값: 44.5005 / 편차 평균값: 5.2306)



(그래프상에서의 DLinear 모델 결과: 예측 값과 실제 값의 추세)

○ 3.8 모델 발전 계획 수립

발전된 모델링 결과 요약 테이블

종속변수	7일 정확도	14일 정확도	28일 정확도	7일 RMSE	14일 RMSE	28일 RMSE
DLinear (생철)	84%	73%	62%	3.81	7.99	20.13
DLinear (중량A)	81%	77%	54%	3.69	7.78	21.80
DLinear (경량A)	80%	70%	51%	3.75	7.82	21.58
DLinear (선반A)	81%	75%	57%	3.60	7.70	20.54
DLinear (선반C)	82%	72%	57%	3.49	7.17	20.04

- 모델 발전 계획을 수립하는 과정에서, 프로젝트 팀은 Darts 라이브러리를 사용한 DLinear 모델을 선택하고, Autogluon 라이브러리를 통해 하이퍼파라미터 튜닝과 피쳐 기반 모델링을 진행. **하지만 결과적으로 독립변수를 사용하지 않은 모델과 독립변수를 사용한 모델과의 정확도 차이가 없음을 확인.** 해당 결과의 원인은 Autogluon 라이브러리의 자체 내부 기능, 즉 모델링 과정 내부에서 상관계수와 피쳐 중요도가 낮은 변수들을 자동적으로 걸러주는 기능이 있음을 확인. 이는 독립변수들이 모델의 성능 개선에 어떠한 영향을 끼치지 않고 있음을 시사.

```
Provided dataset contains following columns:
  target:           'target'
  known covariates: ['year', 'month', 'day', 'week_day', 'year_month', 'week', 'week_num', 'holiday']

AutoGluon will gauge predictive performance using evaluation metric: 'RMSE'
  This metric's sign has been flipped to adhere to being higher_is_better. The metric score can be multiplied by -1 to get the metric value.
============================================================

Starting training. Start time is 2023-12-10 14:32:54
Models that will be trained: ['DLinear']
Training timeseries model DLinear. Training for up to 72000.0s of the 72000.0s of remaining time.
  -5.0519       = Validation score (-RMSE)
  76.03   s     = Training runtime
  0.01    s     = Validation (prediction) runtime
Not fitting ensemble as only 1 model was trained.
Training complete. Models trained: ['DLinear']
Total runtime: 76.06 s
Best model: DLinear
Best model score: -5.0519

Provided dataset contains following columns:
  target:           'target'
  known covariates: ['year', 'month', 'day', 'week_day', 'year_month', 'week', 'week_num', 'holiday']
  past covariates:  ['copper', 'stannum', 'nickel', 'iron_ore', 'oil_price', 'aluminum_alloy', 'gold_price', 'euro', 'plumbum']

AutoGluon will gauge predictive performance using evaluation metric: 'RMSE'
  This metric's sign has been flipped to adhere to being higher_is_better. The metric score can be multiplied by -1 to get the metric value.
============================================================

Starting training. Start time is 2023-12-10 17:22:03
Models that will be trained: ['DLinear']
Training timeseries model DLinear. Training for up to 72000.0s of the 72000.0s of remaining time.
  -5.0519       = Validation score (-RMSE)
  54.77   s     = Training runtime
  0.01    s     = Validation (prediction) runtime
Not fitting ensemble as only 1 model was trained.
Training complete. Models trained: ['DLinear']
Total runtime: 54.80 s
Best model: DLinear
Best model score: -5.0519
```

(독립변수 여부와 상관없이 RMSE가 5.0519로 동일함)

- 결론적으로 독립변수의 유무는 모델에 어떠한 영향을 끼치지 않기 때문에 Darts 라이브러리에

없는 Autogluon의 DLinear 하이퍼파라미터 튜닝 기능을 사용하여 모델 정확도를 대폭 개선했음을 확인. Darts 라이브러리의 DLinear 모델을 활용한 품목 '생철'의 7일 예측 기간 RMSE는 8.25였고 정확도는 61%를 도출. 하지만 Autogluon을 이용해 하이퍼파라미터를 튜닝한 결과, '생철' 7일 예측 기간의 RMSE는 3.81에 정확도는 무려 84%까지 성능을 유의미하게 개선.

```
RMSE는 8.250889878입니다.
정확도는 61% 입니다.
```

(Darts를 이용한 DLinear의 '생철' RMSE 및 정확도 결과)

```
predictor.fit(
    train_set_list[j][i],
    time_limit=1200*60,     # 제한학습시간
    hyperparameters={
        "DLinear": [
            {"hidden_dimension":25, "learning_rate ":0.0004, "epochs":600, "scaling":"std",},
            ],
    },
    hyperparameter_tune_kwargs=None,
)
```

(Darts 라이브러리에 없는 Autogluon의 하이퍼파라미터 튜닝 기능)

- 독립변수를 활용한 모델링은 이전에 진행한 XGBoost 등의 모델을 통해 얻은 변수 간의 상관관계, 피쳐 중요도 등의 인사이트, 그리고 상관성이 높은 원자재들의 가격을 바탕으로 진행. 이를 통해 특성 공학을 다시 한 번 적용하여 DLinear 모델링을 수행.

- 해당 모델링 과정에서 사용된 독립변수:

Autogluon의 DLinear 모델에 쓰인 독립변수 목록 테이블

데이터 변수명	컬럼명	데이터 종류	기간 단위
구리 가격	copper	정형 데이터	일 단위
주석 가격	stannum	정형 데이터	일 단위
니켈 가격	nickel	정형 데이터	일 단위
철광석 가격	iron ore	정형 데이터	일 단위
국제 유가	oil price	정형 데이터	일 단위
알루미늄 합금 가격	aluminum alloy	정형 데이터	일 단위
금 가격	gold price	정형 데이터	일 단위
납 가격	plumbum	정형 데이터	일 단위
유로 환율	euro	정형 데이터	일 단위
년도	year	파생 변수	년 단위
월	month	파생 변수	월 단위
일	day	파생 변수	일 단위
요일	week_day	파생 변수	일 단위
해당 년도의 월	year_month	파생 변수	월 단위
해당 년도의 주	week	파생 변수	주 단위
누적주차	week_num	파생 변수	주 단위
휴일	holiday	파생 변수	일 단위

제 4절 웹 플랫폼 개발

○ 4.1 웹 UI/UX 세부 기능

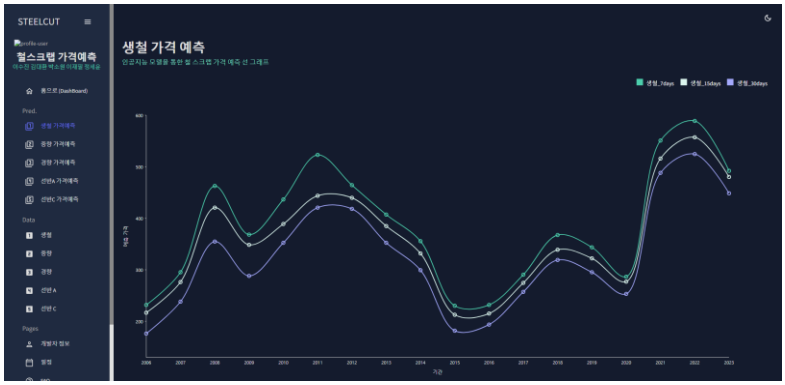
- React와 NodeJS 프레임워크를 활용하여 다양한 기능을 구현한 웹 플랫폼을 개발. 이 웹 플랫폼은 홈 화면 페이지, 철 스크랩 예측 가격 대시보드 페이지, 철 스크랩 가격 및 경제 지표 데이터 테이블 페이지, 고객센터 페이지, 그리고 세부 차트 페이지 등 총 5가지 페이지로 구성.

- 웹 플랫폼의 페이지 구성.

1. *홈 화면 페이지:*
    - 철 스크랩 각 품목별 현재 평균가격 및 전주 대비 등락
    - 철 스크랩 각 품목별 예측 가격 및 전주 대비 등락
    - 국내 2023년 기준 탄소 배출 진행 상황 도넛차트
    - 2022년 철 스크랩 품목별 평균 가격 (분기별) 바 차트
    - 나라별 탄소 배출량 코로플레스 맵
    - 모든 차트를 마우스 호버 시 세부 데이터가 출력될 수 있도록 구성

(홈 화면 페이지)

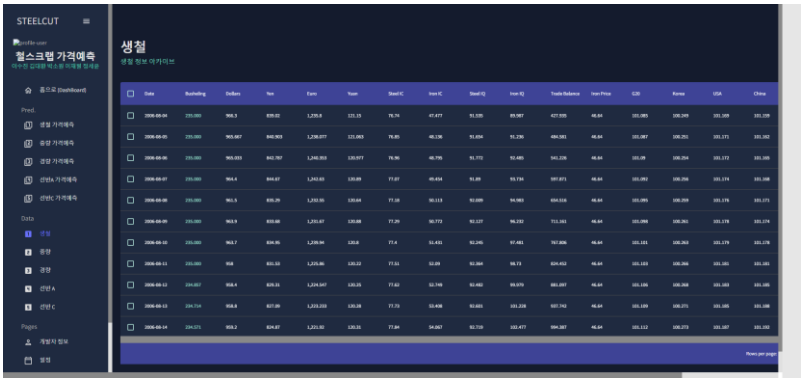
2. *철 스크랩 예측 가격 대시보드 페이지:*
    - 생철, 중량A, 경량A, 선반A, 선반C로 세부 페이지로 구성
    - 각 품목별 7일, 15일, 30일 예측 기간과 가격을 라인 그래프로 구성
    - 차트 마우스 호버 시 세부 데이터가 출력될 수 있도록 구성



(예측 가격 대시보드 페이지)

3. 철 스크랩 가격 및 경제 지표 데이터 테이블 페이지:

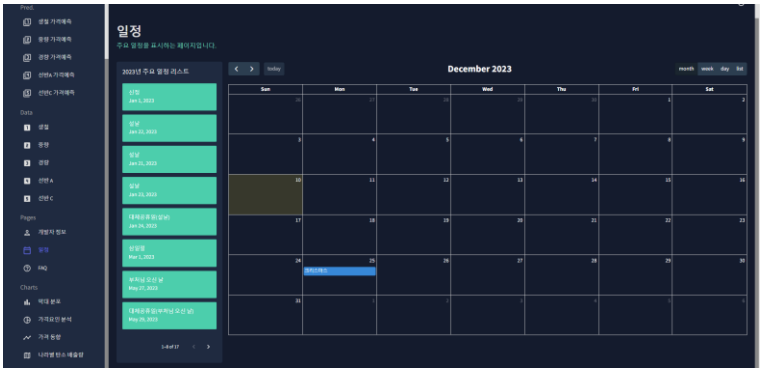
- 생철, 중량A, 경량A, 선반A, 선반C로 세부 페이지로 구성
- 각 품목 날짜 별 평균가격과 상관성과 피쳐 중요도가 높은 독립변수들 테이블 구성
- 전체선택, 필터링, 오름차순/내림차순 등의 추가 기능 구성



(철 스크랩 가격 및 경제 지표 데이터 테이블 페이지 예시)

4. 고객센터 페이지:

- 개발자 정보, 일정, FAQ(자주 묻는 질문) 세부 페이지로 구성
- 개발자 정보 공개와 함께 캘린더를 통해 일정 관리를 진행할 수 있도록 구성
- 철 스크랩에 대한 사전지식이 없는 이용자들을 위한 기본 정보 제공

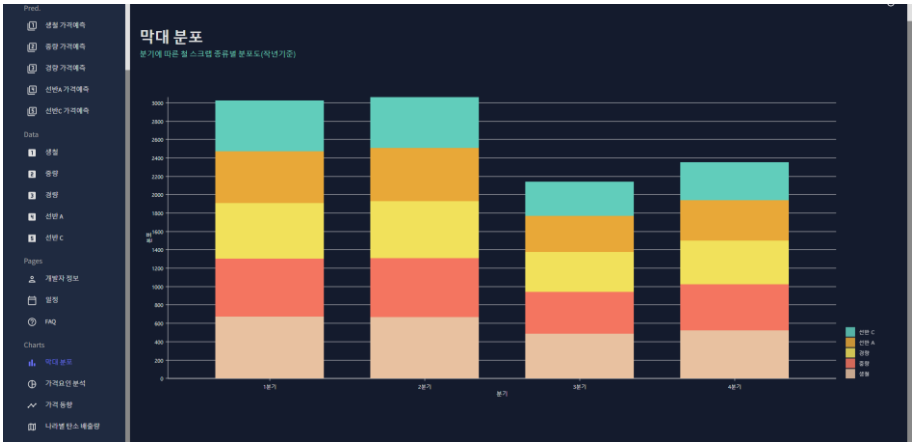


(고객센터 페이지: 캘린더 일정 관리 기능)

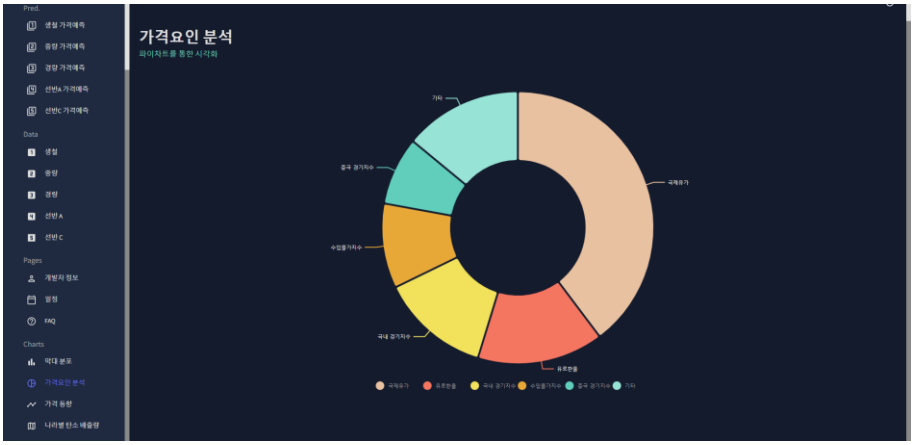
(고객센터 페이지: 철 스크랩 기본 정보 제공)

5. 세부 차트 페이지:

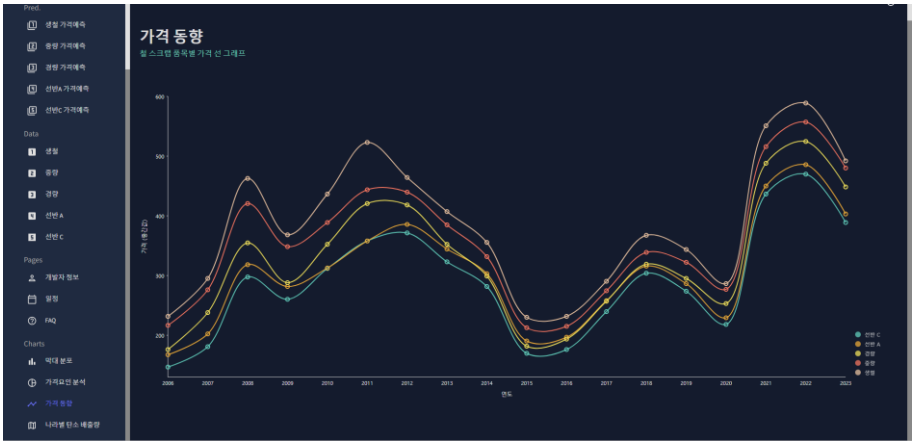
- 막대분포, 가격요인분석, 가격동향, 나라별 탄소배출량 세부 페이지로 구성
- 2022년 철 스크랩 각 품목별 가격 막대 분포로 구성
- 피쳐 중요성이 높은 독립변수들로 도넛차트 구성
- 철 스크랩 각 품목별 가격을 라인 그래프로 표현하여 가격 동향 게시
- 나라별 탄소 배출량을 코로플레스 맵으로 게시
- 모든 차트를 마우스 호버 시 세부 데이터가 출력될 수 있도록 구성



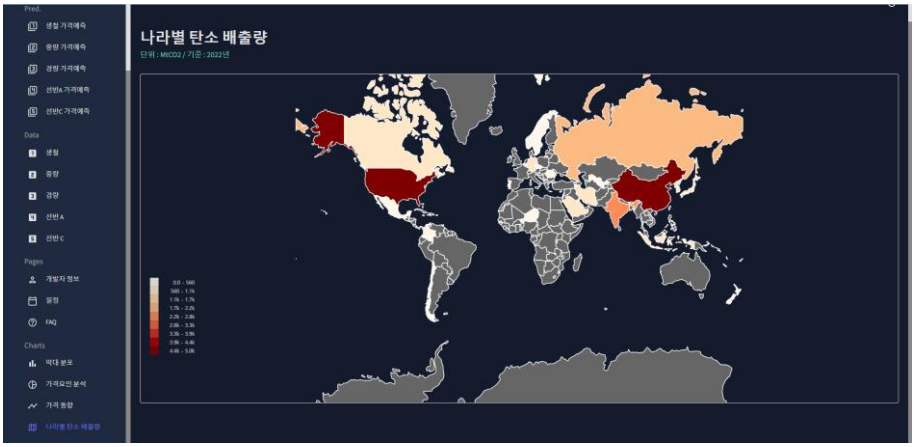
(세부 차트 페이지: 2022년 철 스크랩 각 품목 평균 가격)



(세부 차트 페이지: 가격 피쳐 중요도의 도넛차트)



(세부 차트 페이지: 2006년부터 2023년까지의 가격 동향 라인 그래프)



(세부 차트 페이지: 나라별 탄소배출량 코로플레스 맵)

○ 4.2 웹 파이프라인 구성

- 최신 데이터의 갱신 주기와 웹사이트의 이용률을 고려하여, 매주 일요일 오후 7시부터 월요일 아침 8시까지 공식 점검 시간을 설정. 이 시간 동안 최신 데이터를 갱신하고 모델 학습을 새로 진행할 계획을 구성. 이를 위해 독립변수들의 최신 데이터를 Selenium, BeautifulSoup4 등의 라이브러리를 사용하여 백그라운드에서 크롤링하고, 데이터베이스에 저장하는 과정을 구성. 또한, 데이터베이스에 저장된 데이터를 모델에 다시 학습시켜 데이터의 무결성과 모델의 정확도를 유지.

- 하지만 점검 시간을 설정하고 크롤링, DB 저장, 모델링, 일련의 과정을 자동화하기 위해 일반적으로 작업 스케줄러인 Cron을 사용하지만 **리눅스에 대한 기술 이해도 부족으로 웹 파이프라인 구성 요소를 전부 수동으로 진행.**

제 5절 결론

○ 5.1 기대 효과

- 지속 가능성을 추구하는 기업 입장에서는 저비용과 고효율 전략을 동시에 고려. 본 프로젝트는 철 스크랩 가격 예측을 통해 기업 및 이해관계자들에게 저비용으로 철 스크랩을 매입할 수 있는 기회를 제공. 이를 통해, 기업은 비용을 절감하고 매입한 철 스크랩으로부터 다양한 부가가치를 창출.

- 결과적으로, 이는 철 스크랩의 원활한 수요와 공급을 촉진하고, 궁극적으로는 탄소 중립을 달성하여 환경 보호에 크게 이바지.

○ 5.2 개선 사항

- 본 프로젝트에서는 웹 플랫폼의 최신 데이터 갱신 파이프라인 구축을 위해 Cron을 활용한 자동화 프로세스의 개발을 고려했으나, 시간적 제약으로 인해 Cron에 대한 충분한 이해도를 확보하지 못했음을 파악. 이에 따라 임시적으로 데이터를 수동으로 크롤링하고 데이터베이스에 저장하여 모델링 과정을 진행. 하지만 해당 개발 프로세스에서 서버의 안정성과 사용자 편의성을 보장하기 위해 Cron을 활용한 파이프라인 개발이 필요함을 인식. 이에 따라 인적 자원의 효율적인 배분을 위해서 Cron을 이용한 자동화 프로세스의 개발이 본 프로젝트 팀이 개선해야 할 주요 과제로 정의.

- 또한, 서버 기술에 대한 이해도 부족으로 인해 로컬 환경에서 데이터베이스 및 웹 플랫폼 구축을 실행하였으나, 이는 서버연결 및 유지보수 측면에서 불안정한 상태를 유지할 가능성이 높음을 확인. 이에 따라, 향후 서버에 대한 기술조사를 진행하여 클라이언트사 서버와 연결 프로세스를 개선하는 것을 본 프로젝트의 개선 사항으로 제시.

- 마지막으로 본 프로젝트에서는 시계열 SOTA 모델인 DLinear를 활용하였으나, 이용자의 관점에서는 품목 '생철'의 7일 예측 기간 결과의 84% 정확도는 100%의 정확도가 아니기 때문에 상대적으로 낮은 수준이 될 수 있음을 파악. 특히, 14일과 28일의 예측 결과 정확도는 각각 73%, 62%로, 더 효과적인 매입시기를 제공하기 위해 이 두가지 예측 결과의 정확도 개선이 필요함을 인지. 모델의 정확성은 서비스의 핵심 경쟁력이기에 경쟁력 유지를 위해선 지속적으로 모델의 정확도에 대한 지속적인 개선을 요구.

○ 5.3 참고 문헌

- 철스크랩 자급률 추이 : https://www.ferrotimes.com/news/articleView.html?idxno=21772

- DLinear 깃허브 가이드 : https://unit8co.github.io/darts/index.html

- Autogluon API : https://auto.gluon.ai/stable/index.html

- Darts 사용방법 : https://www.jmlr.org/papers/volume23/21-1177/21-1177.pdf

- DLinear 논문 : https://arxiv.org/pdf/2205.13504.pdf

○ 5.4 데이터 수집 사이트

- 환율(달러, 엔, 유로, 위안): https://finance.naver.com/marketindex/

- 비철금속(구리, 납, 아연, 니켈, 알루미늄, 주석): https://finance.naver.com/marketindex/

- 국제유가(Dubai): https://finance.naver.com/marketindex/materialDetail.nhn?marketindexCd=OIL_DU

- 금 가격: https://finance.naver.com/marketindex/goldDetail.nhn

- 무역수지: https://www.index.go.kr/unity/potal/main/EachDtlPageDetail.do?idx_cd=2735

- 경상수지: https://kosis.kr/statHtml/statHtml.do?orgId=301&tblId=DT_301Y013&checkFlag=N

- 전기요금: https://bigdata.kepco.co.kr/cmsmain.do?scode=S01&pcode=000166&pstate=L&redirect=Y

- 철광석가격: https://kr.investing.com/commodities/iron-ore-62-cfr-futures-historical-data

- 수입물가지수:
  https://kosis.kr/statHtml/statHtml.do?orgId=392&tblId=DT_AA12&vw_cd=MT_ZTITLE&list_id=T_21&seqNo=&lang_mode=ko&language=kor&obj_var_id=&itm_id=&conn_path=MT_ZTITLE

- 철 스크랩(수입단가, 수입량):
  https://kosis.kr/statHtml/statHtml.do?orgId=392&tblId=DT_AA12&vw_cd=MT_ZTITLE&list_id=T_21&seqNo=&lang_mode=ko&language=kor&obj_var_id=&itm_id=&conn_path=MT_ZTITLE

- 철 스크랩 가격: https://www.index.go.kr/unity/potal/main/EachDtlPageDetail.do?idx_cd=1143

- 철강 생산량:
  https://kosis.kr/statHtml/statHtml.do?orgId=363&tblId=TX_36301_A000&vw_cd=MT_ZTITLE&list_id=L_16&seqNo=&lang_mode=ko&language=kor&obj_var_id=&itm_id=&conn_path=MT_ZTITLE

- 철강 원자재 가격 동향: https://www.data.go.kr/data/3039951/fileData.do

- OECD 경기선행지수(G20, 한국, 미국, 중국): https://snapshot.bok.or.kr/dashboard/F1

- 금속광물 수입물량:
  https://kosis.kr/statHtml/statHtml.do?orgId=301&tblId=DT_403Y004&vw_cd=MT_ZTITLE&list_id=S2_301008_003_001&seqNo=&lang_mode=ko&language=kor&obj_var_id=&itm_id=&conn_path=MT_ZTITLE

- 금속광물 수입금액:
  https://kosis.kr/statHtml/statHtml.do?orgId=301&tblId=DT_403Y004&vw_cd=MT_ZTITLE&list_id=S2_301008_003_001&seqNo=&lang_mode=ko&language=kor&obj_var_id=&itm_id=&conn_path=MT_ZTITLE