Dataiku 실전 아카데미 & AI 해커톤

2일차

P&D Solution

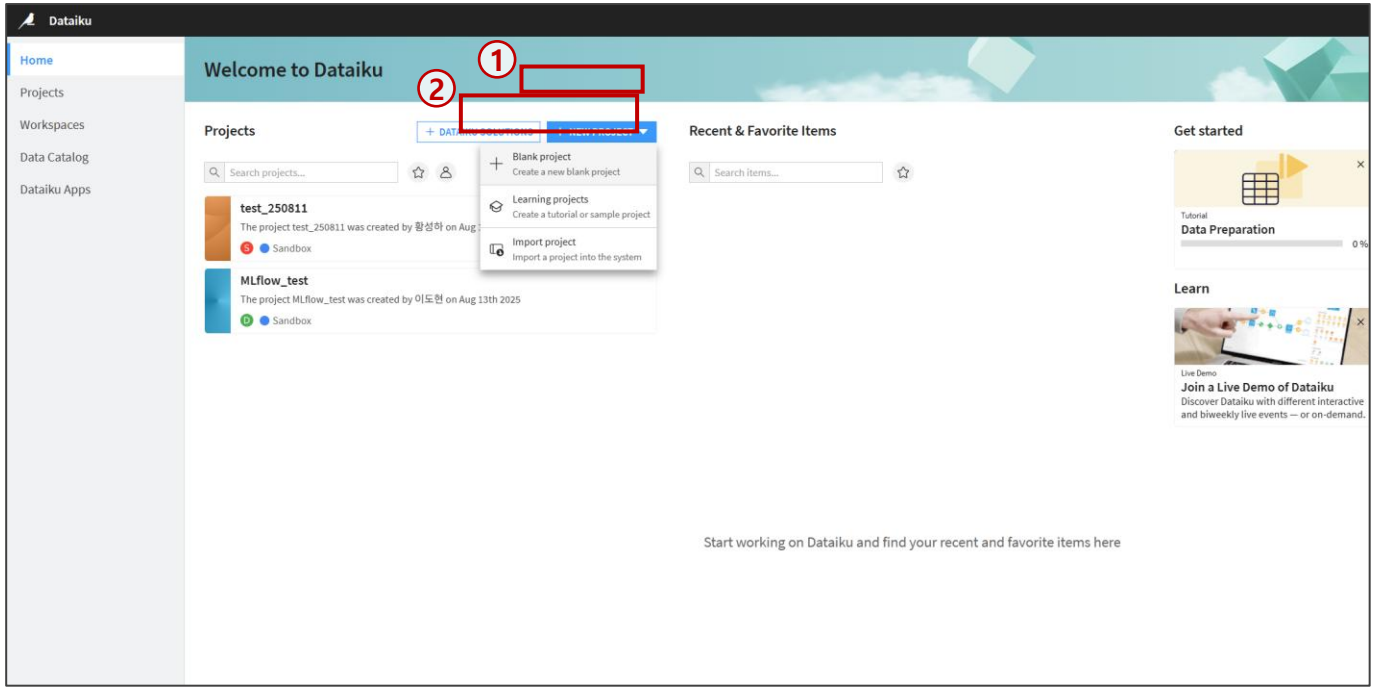
Dataiku 기본 세팅

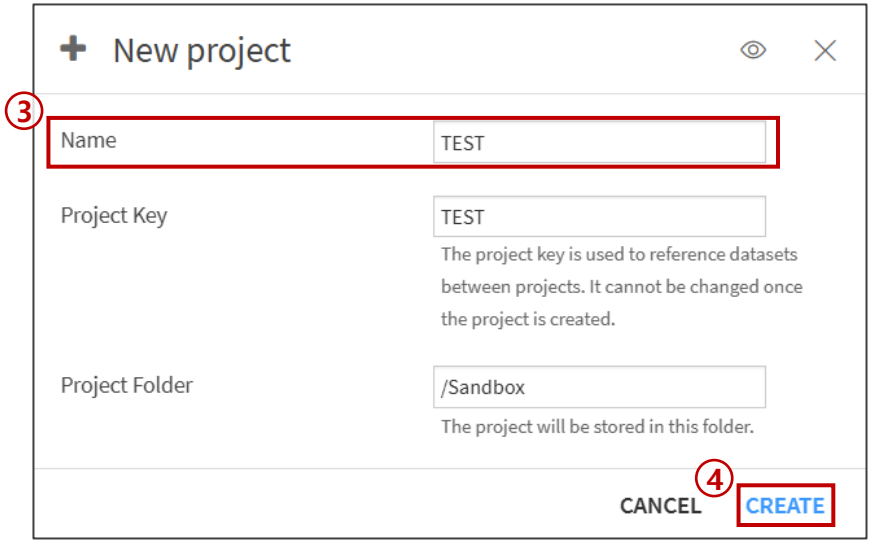
P&D
Solution

프로젝트 생성

- 프로젝트 생성 과정 : NEW PROJECT > Blank project > Name(프로젝트명) > CREATE

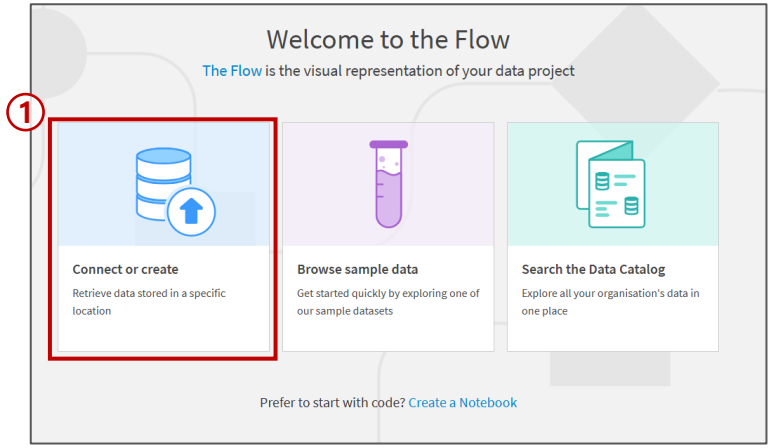
  *본 실습은 DSS 14.3.0 버전에서 진행

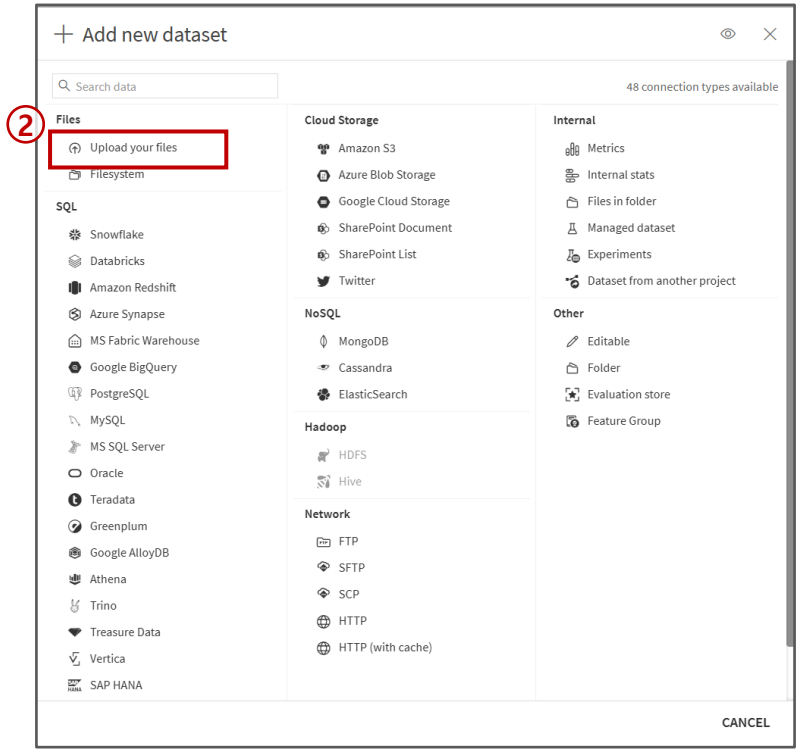


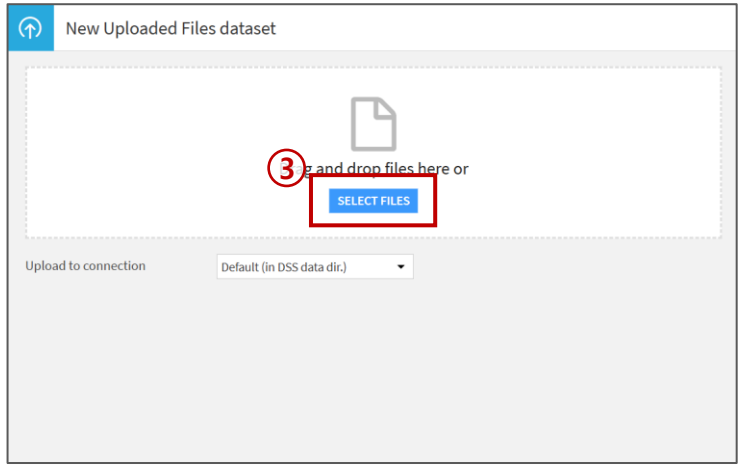


데이터 불러오기

- 로컬 파일 및 다양한 DB 형태의 데이터 연결 가능
- 데이터 불러오기 과정: Connect or create > Upload your files > SELECT FILES > Schema > INFER TYPES FROM DATA > CREATE

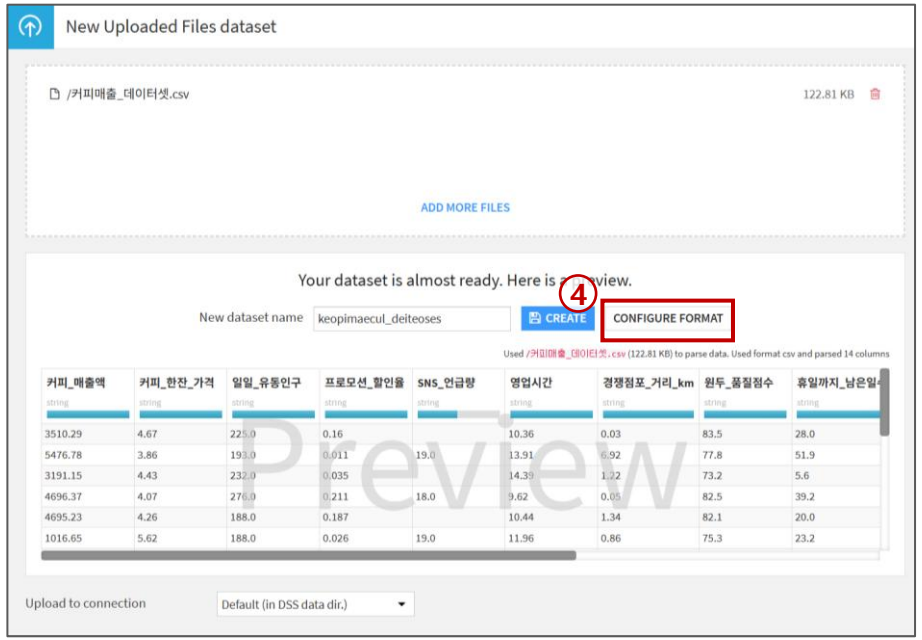


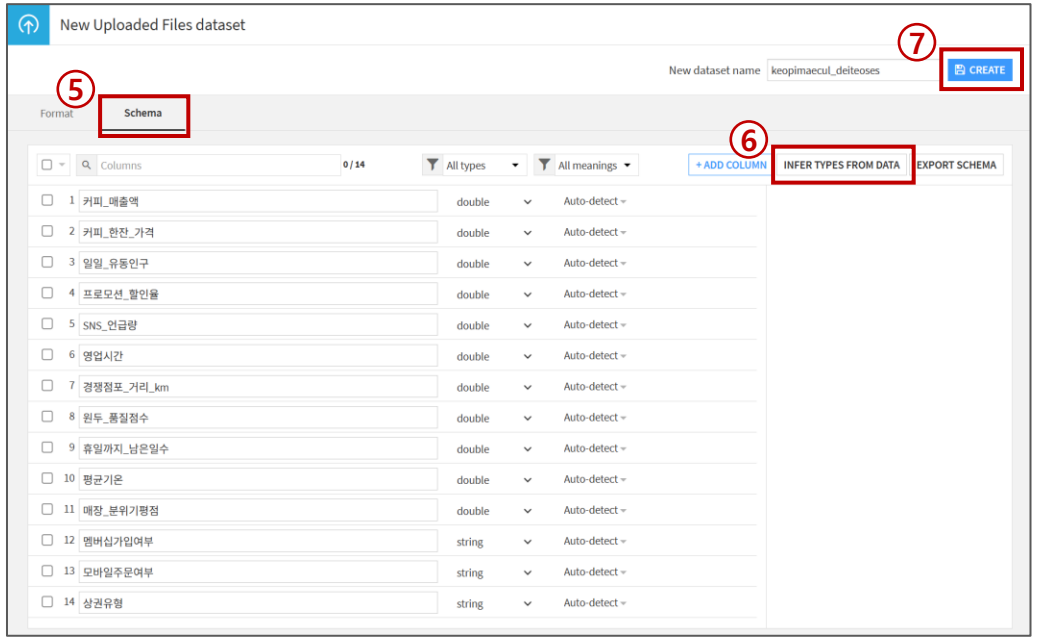




데이터 불러오기

- 로컬 파일 및 다양한 DB 형태의 데이터 연결 가능
- 데이터 불러오기 과정: Connect or create > Upload your files > SELECT FILES > Schema > INFER TYPES FROM DATA > CREATE





Dataiku Hands-on 1

P&D
Solution

Hands-on 1 - What is the purpose?

- Hands-on 1 목표 : **Dataiku 기본기능 익히기**

Hands-on 1에서는 커피 판매 데이터를 활용해 Dataiku의 핵심 기능을 직접 체험합니다.

모델링 이전 단계인 **데이터 탐색, 전처리, 시각화, 통계 분석, 차트 활용, 다변량 분석** 등을

순차적으로 실습하며, Dataiku의 전체 분석 흐름과 인터페이스를 익히는 것이 목표입니다.

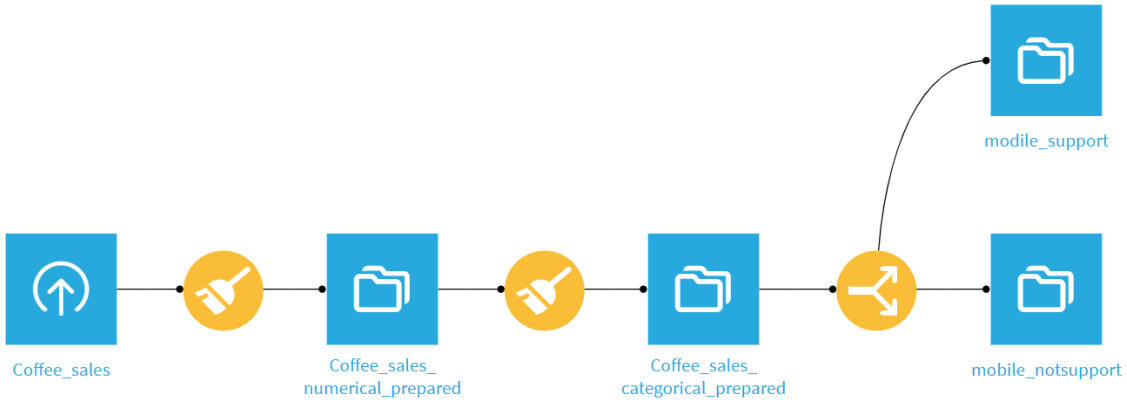
Hands-on 1 - 데이터셋 소개

데이터셋1. 커피 매출액(Coffee Sales.csv)

rows: 1,500 columns: 14

컬럼명	컬럼 유형	설명	범위
커피 매출액	연속형 (float)	반응변수 하루 커피 매출액(원)	0.000 ~ 9614.990
커피 한 잔 가격		커피 1잔 평균 판매가격	2.560 ~ 6.460
일일 유동인구		매장 인근 하루 유동인구(명)	38.000 ~ 1043.000
프로모션 할인율		당일 적용된 프로모션 할인율	0.001 ~ 0.285
SNS 언급량		해당 매장/브랜드의 일일 SNS 언급량	8.000 ~ 38.000
영업 시간		당일 영업시간(시간).	8.000 ~ 15.980
경쟁점포 거리 (km)		가장 가까운 경쟁 매장까지의 거리(km)	0.000 ~ 12.140
원두 품질 점수		원두 품질 평가지표(50~95점)	50.000 ~ 95.000
휴일까지 남은 일수		다음 공휴일까지 남은 일수(일)	0.100 ~ 59.900
평균 기온`		해당 일의 평균 기온(℃)	-4.000 ~ 38.000
매장 분위기 평점		매장 분위기/서비스 평점(1~5점)	1.100 ~ 5.000
멤버십 가입 여부	범주형	고객의 멤버십 가입 여부.	회원(783) | 비회원(717)
모바일 주문 여부		모바일 주문(테이크아웃/픽업) 지원 여부.	지원(864) | 미지원(636)
상권 유형		매장이 위치한 상권 유형.	Business(720) | Residential(718) | Tourist(62)

Hands-on 1 - 시나리오



Step1

데이터 적재 및 **EDA**

Dataiku에 CSV 데이터를 적재 시키고, **Analyze, Charts, Statistics** 기능을 통해 데이터를 살펴보는 **EDA** 과정을 진행합니다.

Step2

전처리

EDA 결과를 기반으로 데이터 전처리를 진행합니다. Visual recipes 중 **Prepare recipe**을 활용하며, 본 과정에서는 연속형 변수 전처리와 범주형 변수 전처리를 구별하여 진행합니다.

Step3

데이터 분할

Visual recipes 중 **Split recipe**를 활용하여, 데이터셋을 분할합니다.
본 과정에서는 모바일 주문 여부 컬럼을 기준으로 두 개의 데이터셋으로 분할합니다.

Dataiku Hands-on 2

P&D
Solution

Hands-on 2 - What is the purpose?

- Hands-on 2 목표 : Dataiku에서 **데이터 분석 전 과정 진행하기**

Hands-on 2에서는 제조 데이터를 활용해 Dataiku에서 데이터 분석의 전 과정을 수행합니다.

Hands-on 1에서 다뤘던 **EDA**와 **전처리 포함, AutoML 기능**으로 모델을 학습하고 해석 및 배포까지

진행하며 Dataiku를 통한 데이터 분석 전체 흐름을 익히는 것이 목표입니다.

Hands-on 2 - 데이터셋 소개

데이터셋 2_1. 제조 데이터1 (manufacturing1.csv)

rows: 6,736 columns: 4

컬럼명	컬럼 유형	설명	범위
ID	범주형	설비 ID	T0001 ~ T6736
sub_ID		설비의 추가 ID	AA001 ~ AW136
Line_id		라인 ID	A ~ F
PassOrFail	수치형	정비 이상(1), 정상(0)	정상(6,697), 이상(39)

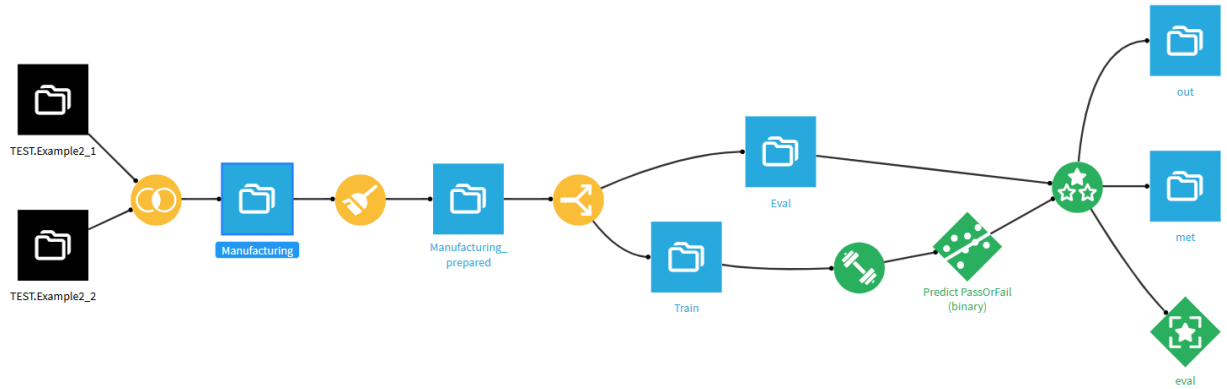
Hands-on 2 - 데이터셋 소개

데이터셋 2 _2. 제조 데이터1 (manufacturing2.csv)

rows: 6,736 columns: 26

컬럼명	컬럼 유형	설명	범위
ID	범주형	설비 ID	T0001 ~ T6736
Sub_ID		설비의 추가 ID	AA001 ~ AW136
고압사출/주입/융용/생산/금형고정시간	연속형 (float)	생산관련 시간데이터	고압사출 (9.36~13.39)
스크류/계량완료/금형이동 위치		위치데이터	스크류위치(653.39~655)
용융최대/사출최대/사출평균속도		사출 공정 전체에서의 평균 속도	용융최대속도(38.5~64.8)
최대/보압/최대압력2/평균압력		전체 사이클에서 평균적으로 가해진 압력	최대압력(140.7 ~ 169.10)
배럴온도1~6		사출기 배럴(원료가 녹는 통) 구간별 온도(히터존 6개)	배럴온도1(244.7~277.9)
호퍼온도		원료(펠릿)를 담는 호퍼의 온도	호퍼온도(38.5 ~ 70.6)
금형온도1, 2		금형의 양쪽(좌/우 또는 전/후)의 온도	금형온도(19.1~25.3)

Hands-on 2 - 시나리오



Step1

데이터 적재 및 **EDA**

Dataiku에 CSV 데이터를 적재 시키고, **EDA** 과정을 진행합니다

Step2

전처리

EDA 결과를 기반으로 데이터 전처리를 진행합니다. Visual recipes 중 **Join recipe**와 **Prepare recipe**을 활용합니다.

Step3

데이터 분할

Visual recipes 중 **Split recipe**를 활용하여, 데이터셋을 분할합니다.
본 과정에서는 랜덤으로 **Train/Evaluation** 데이터로 분할합니다

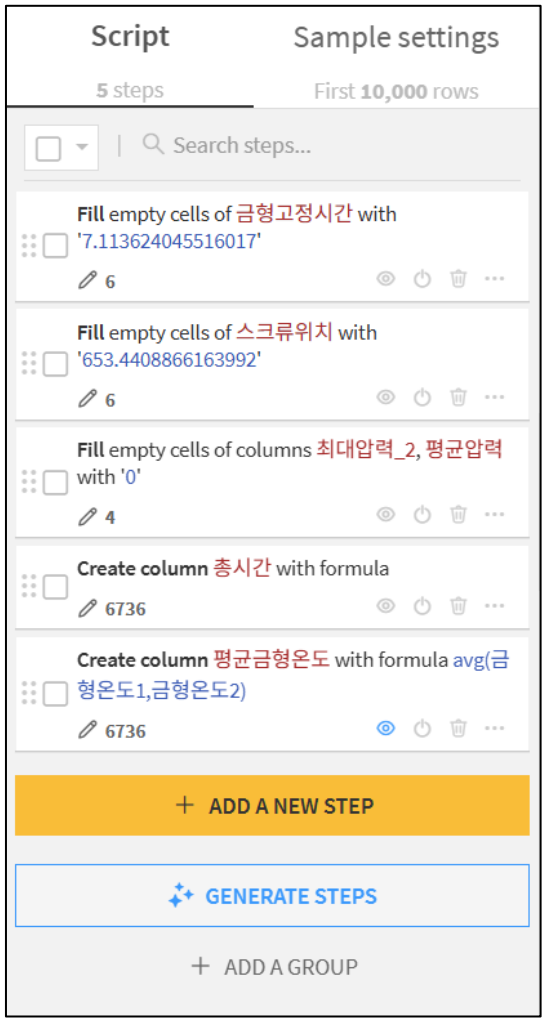
Step4

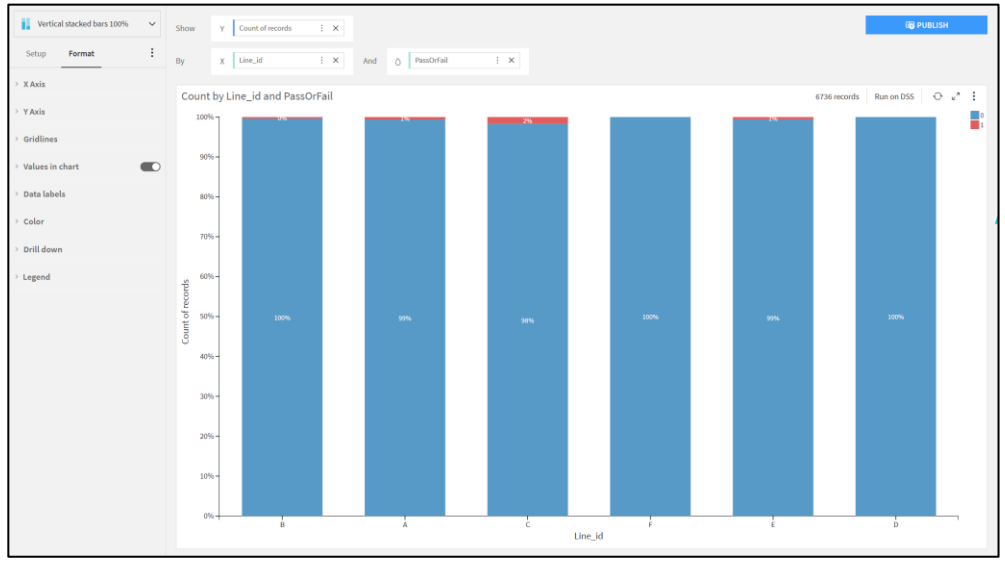
모델 생성 및 배포, 평가

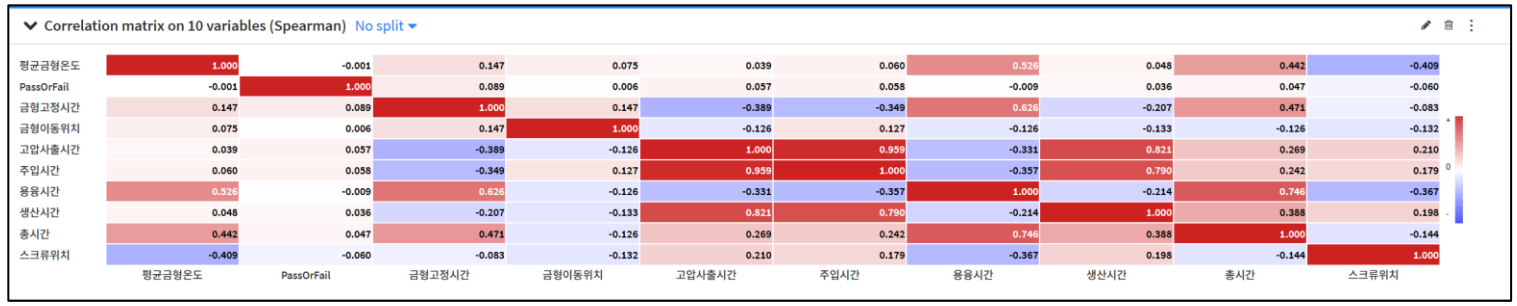
Train 데이터로 모델을 생성하고, 정확도가 높은 모델을 배포하며,
Evaluation 데이터를 활용하여, 모델이 잘 생성되었는지를 평가합니다

Hands-on 2 - 시나리오

- 전처리 진행 & 불량비율이 가장 높은 라인 찾기 & 평균금형온도와 상관성 가장 높은 변수



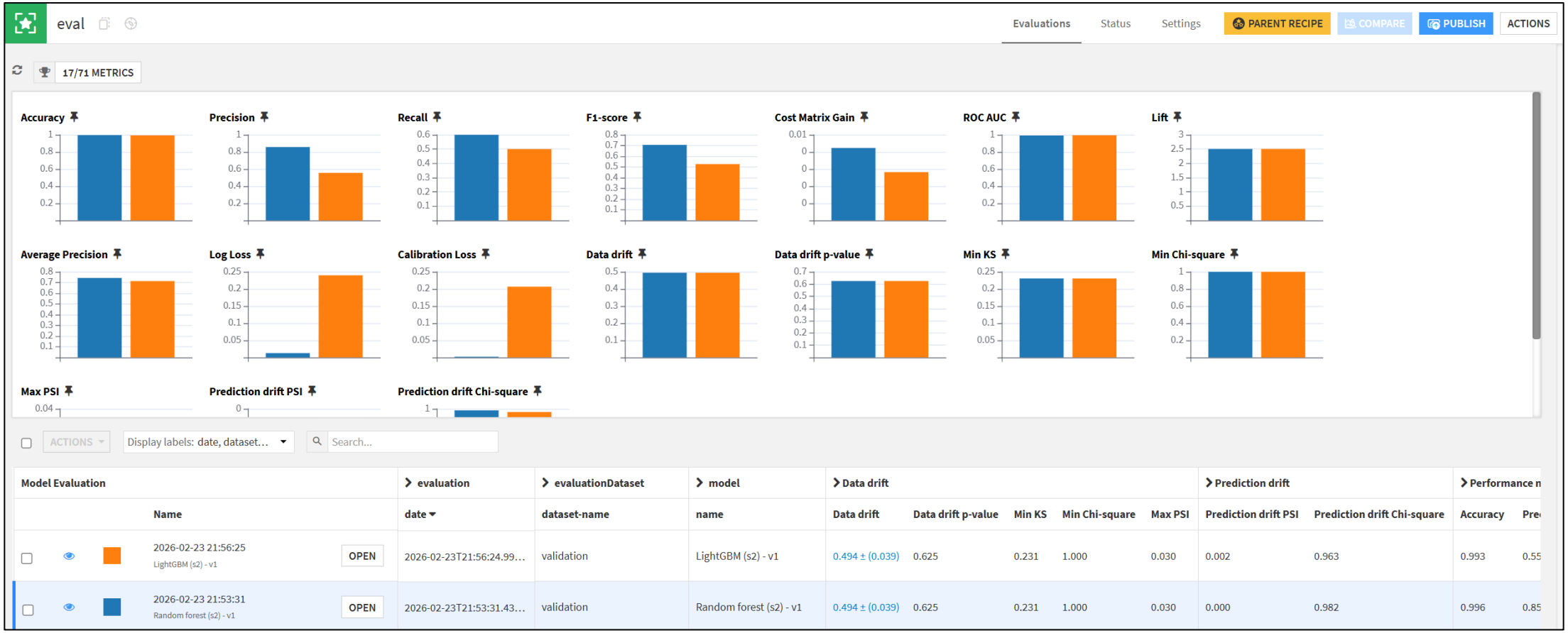




	평균금형온도	PassOrFail	금형고정시간	금형이동위치	고압사출시간	주입시간	용융시간	생산시간	총시간	스크류위치
평균금형온도	1.000	-0.001	0.147	0.075	0.039	0.060	0.526	0.048	0.442	-0.409
PassOrFail	-0.001	1.000	0.089	0.006	0.057	0.058	-0.009	0.036	0.047	-0.060
금형고정시간	0.147	0.089	1.000	0.147	-0.389	-0.349	0.626	-0.207	0.471	-0.083
금형이동위치	0.075	0.006	0.147	1.000	-0.126	0.127	-0.126	-0.133	-0.126	-0.132
고압사출시간	0.039	0.057	-0.389	-0.126	1.000	0.959	-0.331	0.821	0.269	0.210
주입시간	0.060	0.058	-0.349	0.127	0.959	1.000	-0.357	0.790	0.242	0.179
용융시간	0.526	-0.009	0.626	-0.126	-0.331	-0.357	1.000	-0.214	0.746	-0.367
생산시간	0.048	0.036	-0.207	-0.133	0.821	0.790	-0.214	1.000	0.388	0.198
총시간	0.442	0.047	0.471	-0.126	0.269	0.242	0.746	0.388	1.000	-0.144
스크류위치	-0.409	-0.060	-0.083	-0.132	0.210	0.179	-0.367	0.198	-0.144	1.000

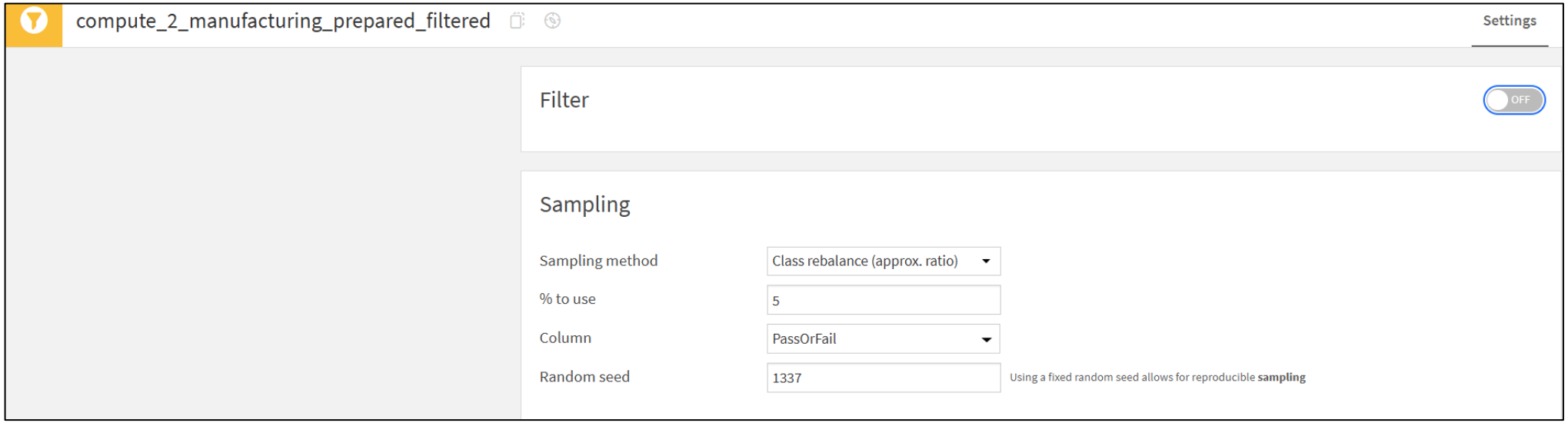
Hands-on 2 - 시나리오

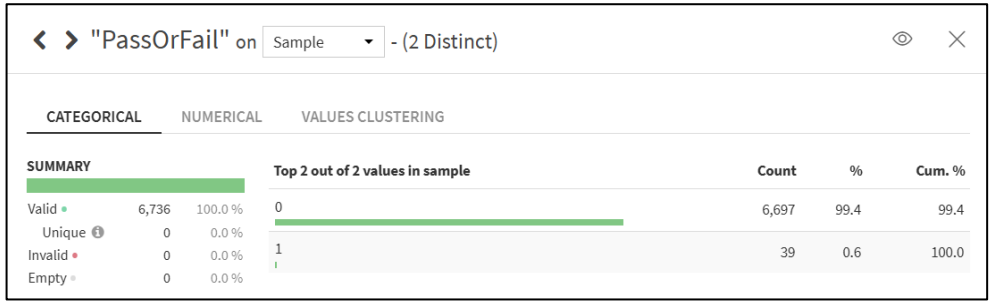
모델 선택 후 모델 변경하여 재평가 하기

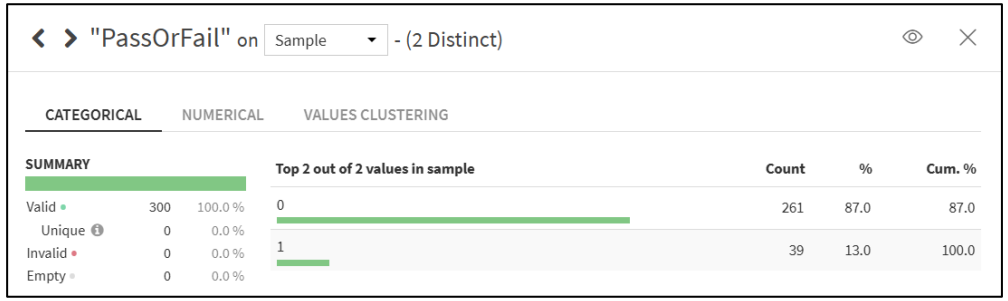


Hands-on 2 - 시나리오

- Class rebalancing을 통해 데이터 변경(정상/불량 비율 차이가 많이 날 때)







Dataiku 실습 1

(중고차 가격 예측 모델 생성)

P&D
Solution

실습 1 - What is the purpose?

실습 1 목표 : 중고차 가격 예측 모델 생성하기

1.데이터 업로드 및 EDA

- 두 개의 CSV 파일 Dataiku에 업로드
  - car_data_1.csv
  - car_data_2.csv
- Tip: 업로드 시 데이터 타입 세팅

2. 데이터 전처리

- Join recipe
  - car_data_1와 car_data_2 데이터를 조인
  - 기준 데이터: car_data_1
  - Join 방식: left join
  - key: 차량ID

- Prepare recipe
  - 제조사 및 차종 컬럼 전처리
    - "/" 기준으로 세 컬럼으로 분할 (split)
    - 분할된 세 컬럼명을 각각 국가, 제조사, 차종로 컬럼명 변경 (rename)
    - 제조사가 bmw인 행만 필터링(filter on the value)
    - 제조사 및 차종, 제조사 컬럼 제거(delete)
  - 불필요한 컬럼 제거 및 결측치 대체
    - 결측치 비율이 30% 이상인 컬럼을 제거
    - 결측치 비율이 30% 이하인 컬럼은 결측치 대체 (결측 대체 방법은 자유롭게 선정)
  - 새로운 파생 변수
    - Formula 기능을 통해 1개 이상 생성
    - Example. 정비_다수 = if(val("정비횟수")>3, 1, 0)

실습 1 - What is the purpose?

실습 1 목표 : 중고차 가격 예측 모델 생성하기

- Prepare recipe
  - 제조사 및 차종 컬럼 전처리
    - "/" 기준으로 세 컬럼으로 분할 (split)
    - 분할된 세 컬럼명을 각각 국가, 제조사, 차종로 컬럼명 변경 (rename)
    - 제조사가 bmw인 행만 필터링(filter on the value)
    - 제조사 및 차종, 제조사 컬럼 제거(delete)
  - 불필요한 컬럼 제거 및 결측치 대체
    - 결측치 비율이 30% 이상인 컬럼을 제거
    - 결측치 비율이 30% 이하인 컬럼은 결측치 대체 (결측 대체 방법은 자유롭게 선정)
  - 새로운 파생 변수
    - Formula 기능을 통해 1개 이상 생성
    - Example. 정비_다수 = if(val("정비횟수")>3, 1, 0)

•Split recipe

- 연도가 2024년인 데이터와 2025년인 데이터로 분할

3. 모델링

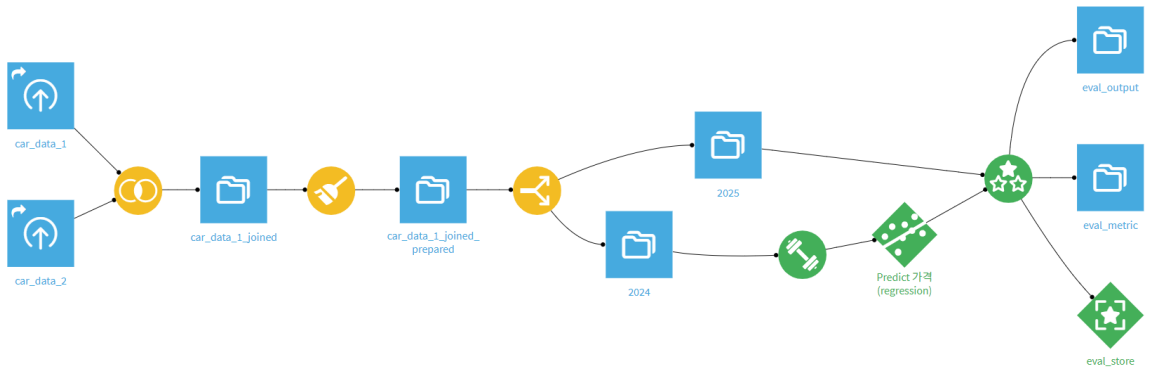
• 모델 구축

- 연도가 2024인 데이터셋과 AutoML기능을 통해 가격을 예측하는 중고차 가격 예측 모델을 생성
- 평가 지표는 MAPE(Mean Absolute Percentage Error)로 변경
- 모델 알고리즘을 자유롭게 선정하여, 최적의 모델을 생성 및 배포

• 모델 평가

- 연도가 2025인 데이터셋과 배포된 모델로 Evaluate recipe 진행

실습 1 - 시나리오



Step1

데이터 적재 및 **EDA**

Dataiku에 CSV 데이터를 적재 시키고, **EDA** 과정을 진행합니다

Step2

전처리

EDA 결과를 기반으로 데이터 전처리를 진행합니다. Visual recipes 중 **Join recipe**와 **Prepare recipe**을 활용합니다.

Step3

데이터 분할

Visual recipes 중 **Split recipe**를 활용하여, 데이터셋을 분할합니다.
본 과정에서는 2024년도와 2025년도 데이터로 분할합니다.

Step4

모델 생성 및 배포, 평가

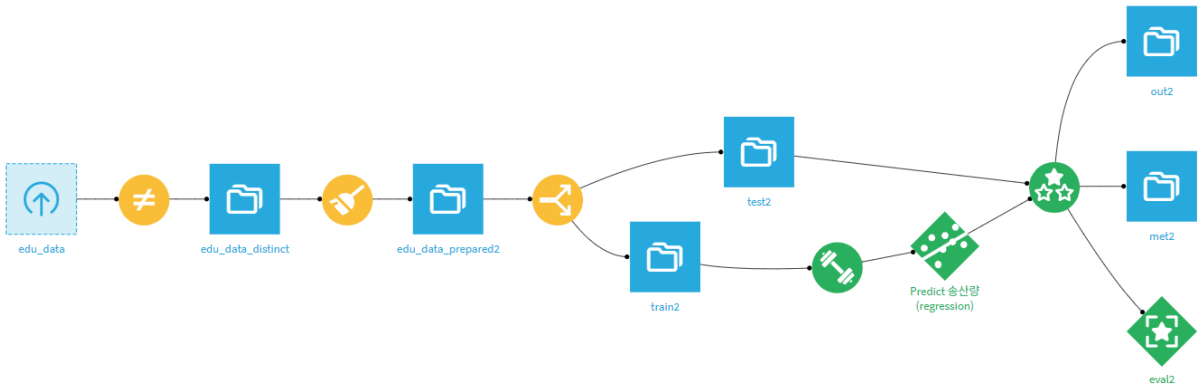
2024년 데이터로 모델을 생성하고, 정확도가 높은 모델을 배포하며,
Evaluation 데이터를 활용하여, 모델이 잘 생성되었는지를 평가합니다

Dataiku 실습 2

(송산량 예측 모델 생성)

P&D Solution

실습 2 - 시나리오



Step1

데이터 적재 및 **EDA**

Dataiku에 CSV 데이터를 적재 시키고, **EDA** 과정을 진행합니다

Step2

Distinct & Prepare Recipe

Distinct Recipe로 중복 행을 제거하고 Prepare Recipe로 데이터 전처리를 진행합니다.

Step3

모델 생성 및 배포

Train 데이터로 모델을 생성하고, 정확도가 높은 모델을 배포합니다.

Step4

모델 예측

Test 데이터를 활용하여, 모델 활용하여 예측합니다

실습 2 - What is the purpose?

실습 2 목표 : 송산량 예측 모델 생성

1.데이터 업로드 및 EDA

- 한 개의 CSV 파일 Dataiku에 업로드
    - 3제강 전로 데이터(Dataiku 교육용).csv

    * Tip: 업로드 시 데이터 타입 세팅

- 전처리

1. 행별로 중복 제거 (재료번호 중복 있음) : Distinct Recipe 활용

   (12,786 rows * 61 columns) -> (12,530 rows * 62 columns)

2. 각 행 별로 어떤 컬럼이든 null값이 있으면 그 행 삭제 (-1,606)

3. "상취패턴" N30, N31만 사용 (-11)

4. "총송산량" 12000 이상 18000미만 값 사용 (-13)

5. "DY측정시점" 11000이상 13000미만 (-530)

6. "종점산소" 300이상 700미만 (-52)

7. "DY냉각재" 0이상 10000미만 (-1)

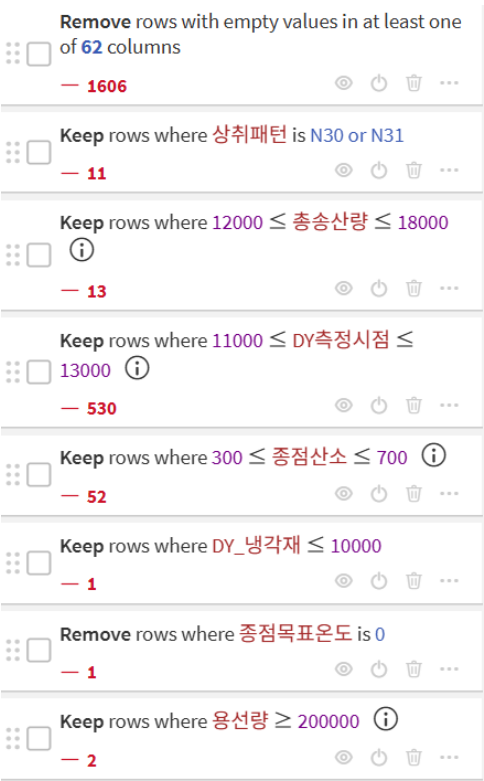
8. "종점목표온도" 0 제외 (-1)

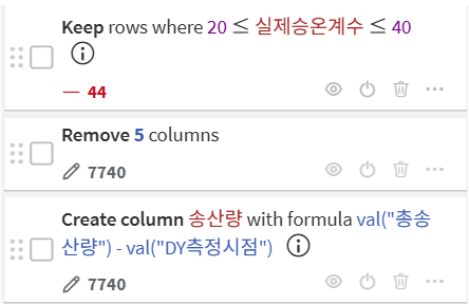
9. "용선량" 200000 미만 제외 (-2)

10 "실제승온계수" 20이상 40미만의 값 사용 (-44)

11. DY실적송산량, 최초종점온도, 최초종점산소, D1_DY_온도, D1_DY_C는 중복으로 삭제

12. 총송산량 – DY측정시점을 통해 "송산량" 컬럼 생성





실습 2 - What is the purpose?

실습 2 목표 : 송산량 예측 모델 생성

2. Split Recipe 활용

80 % Train data, 20% Validation data 생성

3. 모델 생성

Feature Handling

예측변수(승온량) – Y

주요인자

- 전로번호, 전로수명
- 대기시간, 장입온도, 실적배제각도
- DY측정C
- 종점목표온도 ~ Dy열원

4. Algorithms

- Random Forest
- Ridge/Lasso Regression
- LightGBM/ XGBoost

Dataiku 실습 3

(이직 대상자 예측 모델 생성)

P&D
Solution

실습 2 - What is the purpose?

실습 2 목표 : 이직 대상자 예측 모델 생성 및 대쉬보드 생성

1.데이터 업로드 및 EDA

- 두 개의 CSV 파일 Dataiku에 업로드
    - Human_train.csv
    - Human_test.csv
- Tip: 업로드 시 데이터 타입 세팅
- 시각화

- Chart : 회사 부서별 인원 현황표(바차트)

- Statistics – Multivariate Analysis, Correlation Matrix

  -> 직무만족도에 가장 상관성을 갖는 변수는?

통계분석

남녀간의 직무만족도에 차이가 있는지 T-test 로 검정결과 만들기

2. 모델링

- 모델 구축
    - Human_Train 데이터로 AutoML기능을 통해 이직 여부를 예측하는 이직 예측 모델을 생성
    - 평가 지표는 ROC AUC로 생성
    - 모델 알고리즘을 자유롭게 선정하여, 최적의 모델을 생성 및 배포
- 모델 예측
    - Human_Test 데이터로 Score 기능을 활용하여, 예측 실행

3. 대쉬보드 생성

- Pivot Table

  - Row : 이름

  - Column : 없음

  - Value : 부서(Max), 초과근무여부(MAX), 성별(MAX), 결혼여부(MAX)로 생성

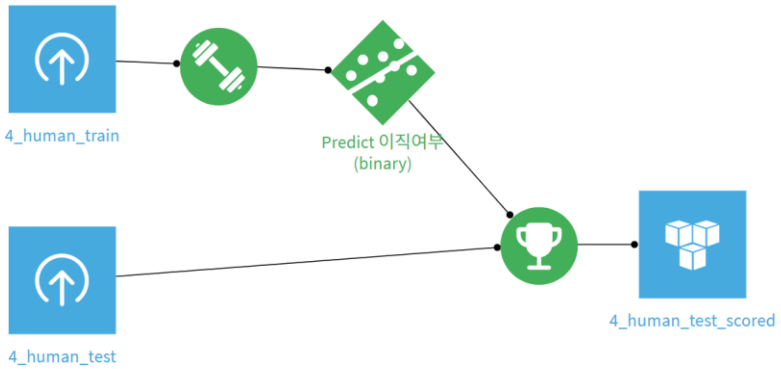
Tip : Filters에 Prediction value를 넣어서 Y만 나오게 선택

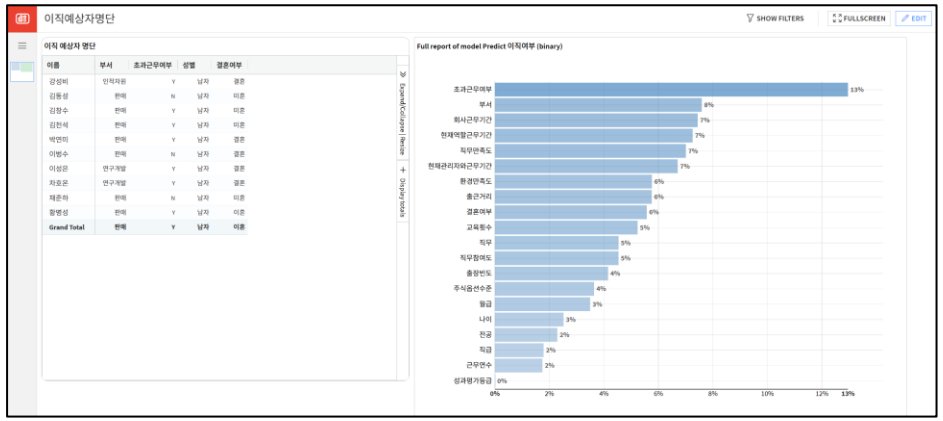
- 생성모델 Feature Importance Chart를 대쉬보드 화

  Feature Importance 메뉴에서 Publish 선택

P&D Solution

실습 2 - 시나리오





Step1

데이터 적재 및 **EDA**

Dataiku에 CSV 데이터를 적재 시키고, **EDA** 과정을 진행합니다

Step2

모델 생성 및 배포

Train 데이터로 모델을 생성하고, 정확도가 높은 모델을 배포합니다

.

Step3

모델 예측

Test 데이터를 활용하여, 모델 활용하여 예측합니다

Step4

모델 예측 결과 **Dashboard**화

Pivot table 활용하여 이직 예측 명단을 생성하고, 생성모델의 주요변수를 확인하는 Feature Importance Table을 생성합니다.

Dataiku 기능 소개

P&D
Solution

Dataiku 기능 소개

Dataiku의 다양한 기능

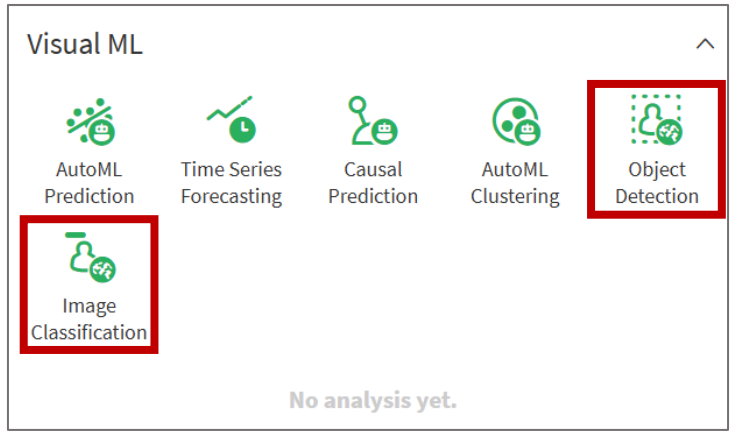
- Dataiku는 다양한 기능을 갖추고 있어 폭넓은 분석을 수행할 수 있으며, **유연하게 커스터마이징 가능**
- 정기적이고 잦은 업데이트로 신기술에 대한 신속한 반영과 사용자 편의성을 지속적으로 향상시키고 있음
  - **Major** update: 대략 연 1회주기로 수행(x.0.0), 대규모 신규 기능과 UI/플랫폼 변화 등 대규모 업데이트 수행
  - **Minor** update: 대략 3개월 주기로 수행(x.y.0), 신규 기능 + 기능 개선 등 비교적 가벼운 업데이트 수행
  - **Patch** update: 필요 시 수행(x.y.z), 버그/성능/보안 수정 등 수행

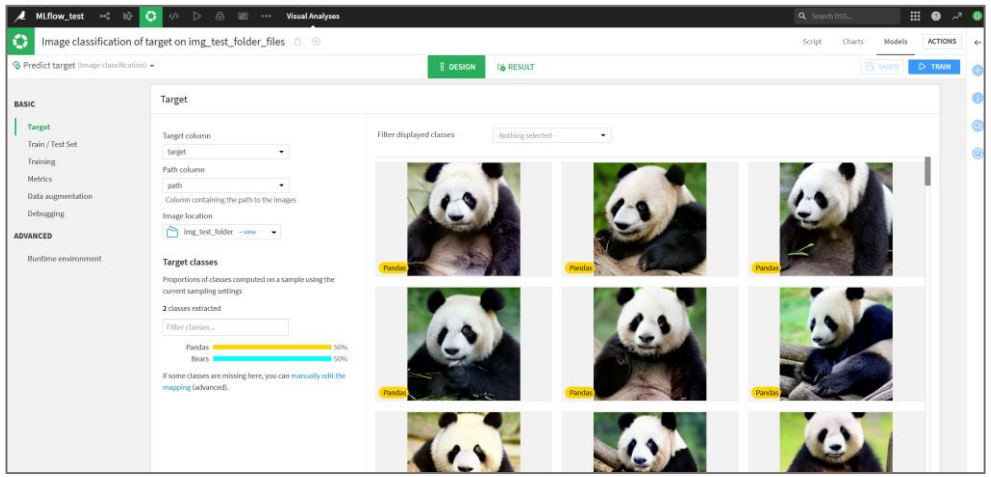
- 오늘 소개할 기능
  - 이미지 분석
  - Code recipes
  - Plugins
  - Saved Models
  - LLM

Dataiku 기능 – 이미지 분석

이미지 분석

- Dataiku는 AutoML과 마찬가지로 **No-Code 기반의 이미지 분석을 지원**합니다.
- 지원하는 이미지 분석
  - ✓ **이미지 분류(Image Classification)**
  - ✓ **객체 탐지(Object Detection)**
- GPU 연동 시 **GPU 자원 활용 가능**

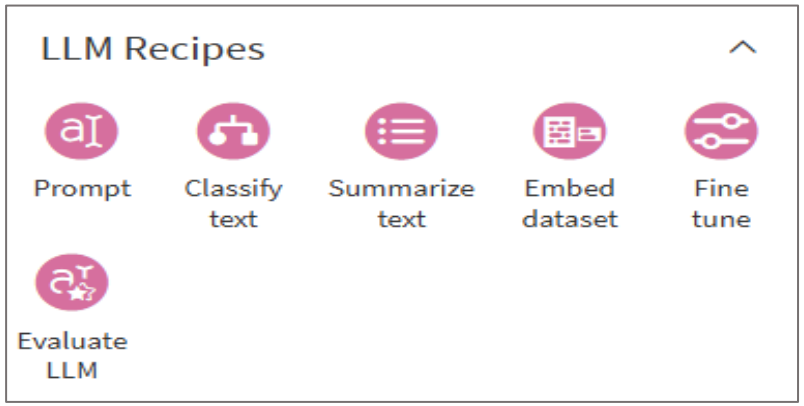


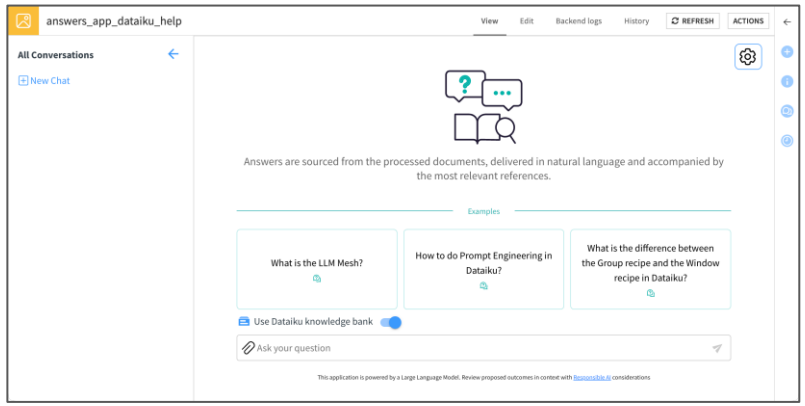


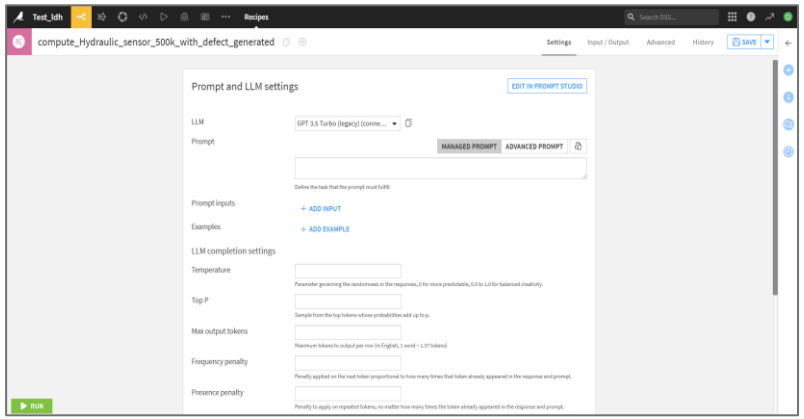
Dataiku 기능 – 이미지 분석

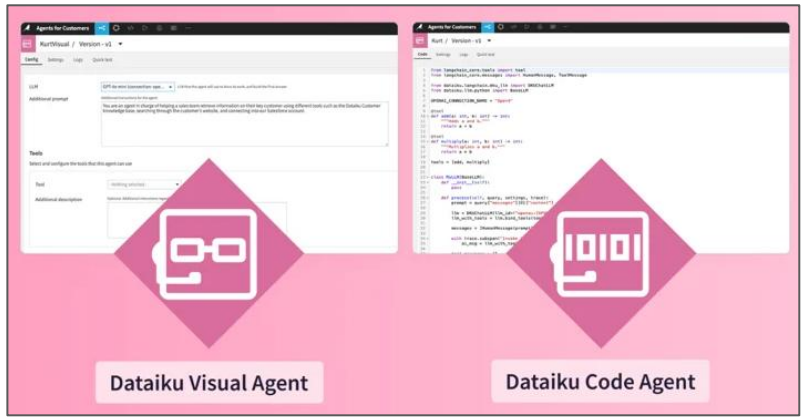
LLM

- Dataiku는 AutoML과 마찬가지로 **No-Code 기반의 LLM 분석과 Code 기반의 LLM을 모두 지원**
- **간단한 텍스트 분석부터 AI Agent까지 모두 지원**









Dataiku 기능 – Code recipes

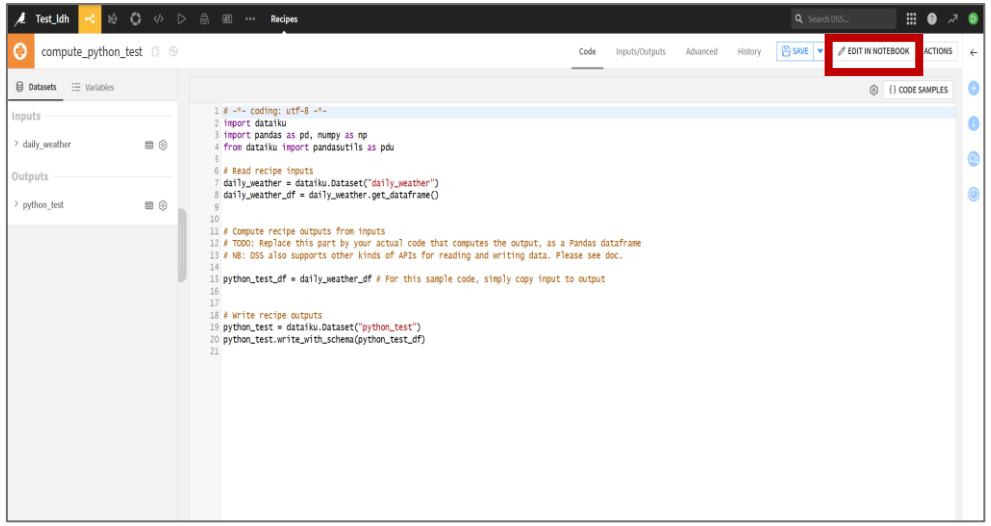
Code recipes

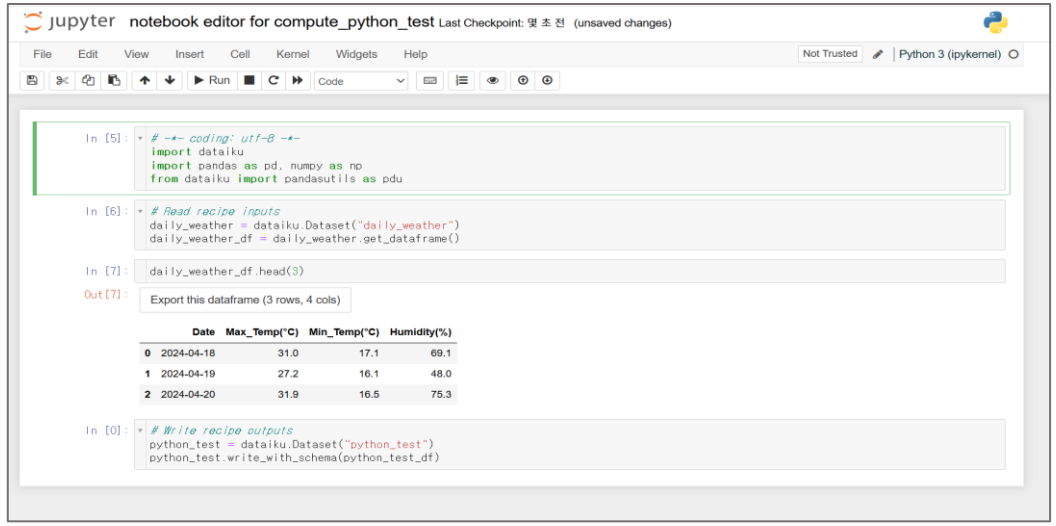
- Dataiku는 **코드 개발자를 위한** Code recipes를 지원
- Code recipes 종류: Python, R, SQL, Shell, Hive, Impala, Spark SQL, Spark Scala, PySpark, Spark R
- **개발자 친화적 환경 제공**
  - ✓ Code Samples 기능을 통한 **코드 재활용 및 공유**
  - ✓ Python / R recipe의 경우**, Jupyter Notebook과 연동** 가능
  - ✓ 플러그인 설치를 통해 로컬 환경의 **Visual Studio Code/Pycharm/Git과도 연동** 가능

Dataiku 기능 – Code recipes

Code recipes

- Dataiku는 **코드 개발자를 위한** Code recipes를 지원
- Code recipes 종류: Python, R, SQL, Shell, Hive, Impala, Spark SQL, Spark Scala, PySpark, Spark R
- **개발자 친화적 환경 제공**
  - ✓ Code Samples 기능을 통한 **코드 재활용 및 공유**
  - ✓ Python / R recipe의 경우**, Jupyter Notebook과 연동** 가능
  - ✓ 플러그인 설치를 통해 로컬 환경의 **Visual Studio Code/Pycharm/Git과도 연동** 가능

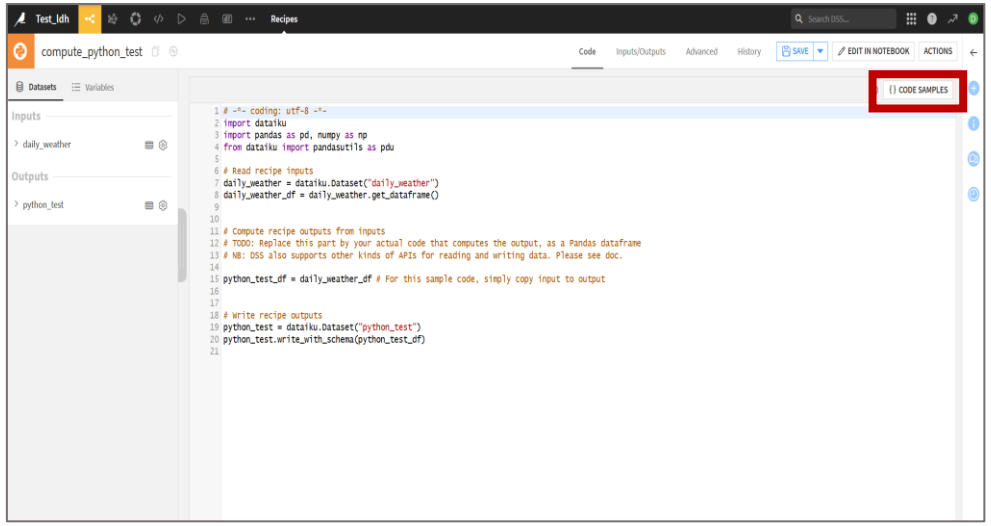


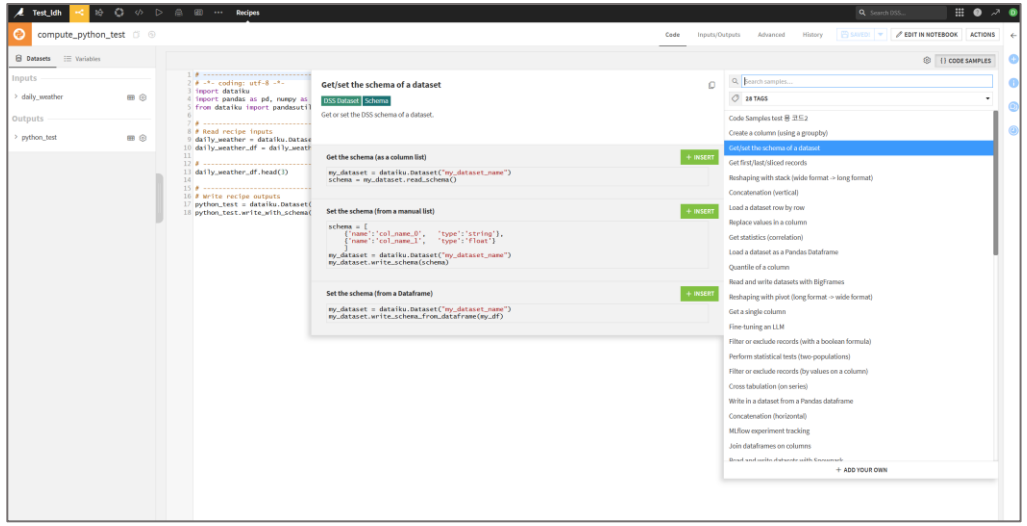


Dataiku 기능 – Code recipes

Code recipes

- Dataiku는 **코드 개발자를 위한** Code recipes를 지원
- Code recipes 종류: Python, R, SQL, Shell, Hive, Impala, Spark SQL, Spark Scala, PySpark, Spark R
- **개발자 친화적 환경 제공**
  - ✓ Code Samples 기능을 통한 **코드 재활용 및 공유**
  - ✓ Python / R recipe의 경우**, Jupyter Notebook과 연동** 가능
  - ✓ 플러그인 설치를 통해 로컬 환경의 **Visual Studio Code/Pycharm/Git과도 연동** 가능





compute_tx_fraudulent_rnd_assigned.py

DATAIKU DSS

PROJECTS

- Credit Card Fraud for Administrat...
  - Recipes
    - compute_features_drift_1
    - compute_logs_api_cleaned
    - compute_tx_fraudulent_rnd_a...
    - compute_tx_top_features_wit...
    - compute_tx_valid_last_weeks...
  - Webapps
  - Wikis
    - Appendix
    - Deliverables
    - Project Overview
    - Requirements
  - Libraries
    - R
    - external-libraries.json
    - python

PLUGINS

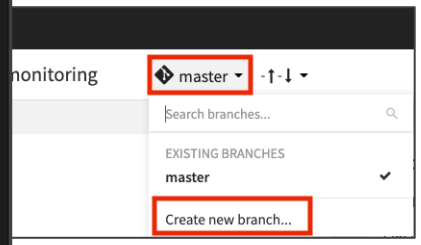
compute_tx_fraudulent_rnd_assigned.py 4

n Support > Code > User > globalStorage > dataiku.dataiku-dss > SOL_CC_FRAUD > RECIPES > compute_tx_fraudulent_rnd_assigned.py

```python
import dataiku
import pandas as pd, numpy as np
from dataiku import pandasutils as pdu
import json

# Read recipe inputs
tx_fraudulent_distinct = dataiku.Dataset("tx_fraudulent_distinct")
df = tx_fraudulent_distinct.get_dataframe()

# Get the list of users
users = json.loads(dataiku.get_custom_variables()['USERS'])

# Randomly assign alerts to users
df['user'] = np.random.choice(users, df.shape[0])
df.sort_values(by='user',inplace=True)

# Write recipe outputs
tx_fraudulent_rnd_assigned = dataiku.Dataset
("tx_fraudulent_rnd_assigned")
tx_fraudulent_rnd_assigned.write_with_schema(df)
```

⊗ 0 ⚠ 4 ⓘ 7 ▷ Run compute_tx_fraudulent_rnd_assigned Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.10.8 64-bit 7 Spell

ark R



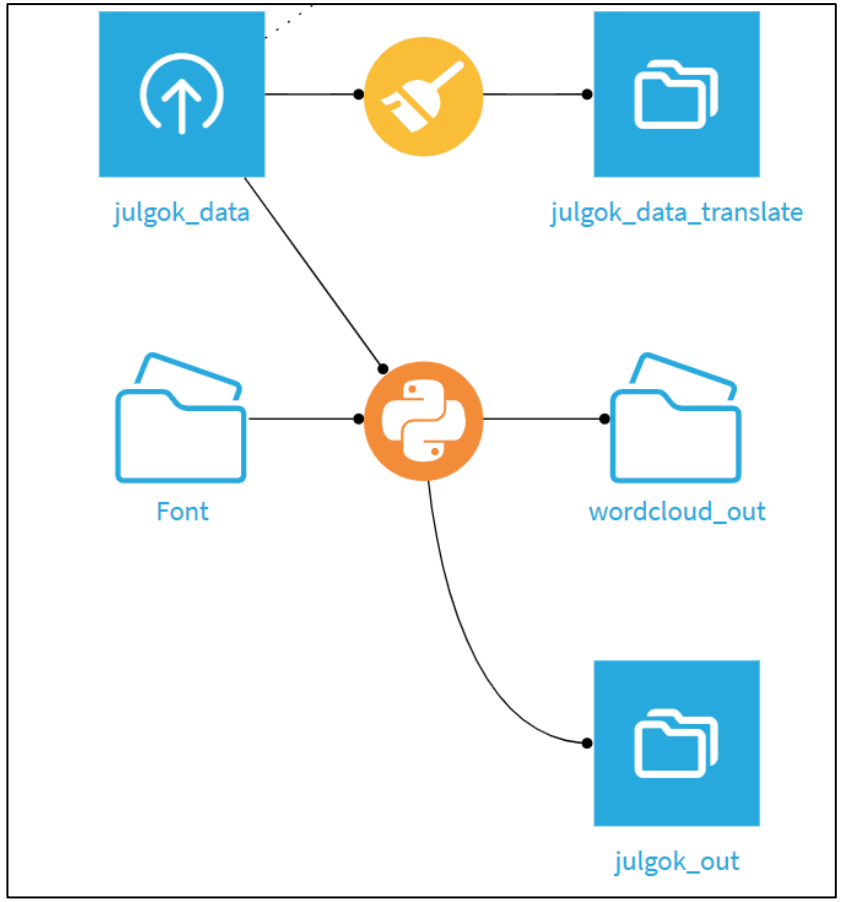
Dataiku & VS Code

Dataiku & Pycharm

Dataiku & Git

Dataiku 기능 – Code recipes

Code recipes



참고사항

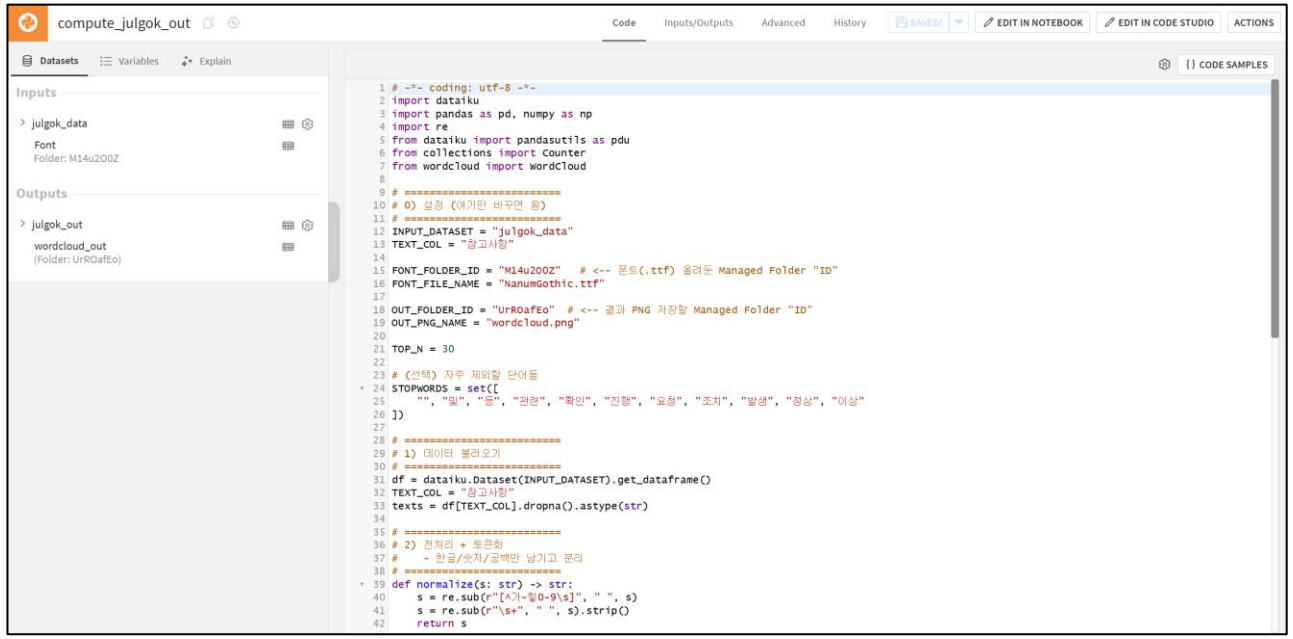
string

Natural lang.

예방정비 네스트 ,클립,(1차) 락커(1차) 전체분해 셋팅 및 구리스도포
각도 불량 정비/좌우각도편차불량/서보 예압 스프링 조정및 절단부 좌우조정 정상작동
날받이부 정비/ 안착트러블 정비 /에어노즐 교체 및 위치 조정으로 정상작동
날받이부 예방정비/ 미그넷네스트 정비 /마그넷네스트 개선품 수정가공품 교체및 세팅 정상
날받이부 예방정비/ 미그넷네스트 정비 /마그넷네스트 개선품 수정가공품 교체및 세팅 정상
날받이부 예방정비/ 미그넷네스트 정비 /마그넷네스트 개선품 수정가공품 교체및 세팅 정상
날받이부 정비/ 안착트러블 정비 /날받이슈트 위치 조정으로 정상작동
자동보정절곡기 캘리브레이션 측정
X값이상으로 비젼점검 비젼카메라 비젼 재셋팅
공급부 정비 / 락커 잠금밀대 정비 / E링관련 핀빠짐으로 E링 교체 정상작동
네스트 청소 에어 예방 정비 / 네스트 에어 상시설정으로 2차휠 안착시 이탈 / 에어노즐 교체및 타임 설정 정상작동
네스트 청소 에어 예방 정비 / 네스트 에어 상시설정으로 2차휠 안착시 이탈 / 에어노즐 교체및 타임 설정 정상작동
네스트 청소 에어 예방 정비 / 네스트 에어 상시설정으로 2차휠 안착시 이탈 / 에어노즐 교체및 타임 설정 정상작동
네스트 청소 에어 예방 정비 / 네스트 에어 상시설정으로 2차휠 안착시 이탈 / 에어노즐 교체및 타임 설정 정상작동
네스트 청소 에어 예방 정비 / 네스트 에어 상시설정으로 2차휠 안착시 이탈 / 에어노즐 교체및 타임 설정 정상작동
네스트 청소 에어 예방 정비 / 네스트 에어 상시설정으로 2차휠 안착시 이탈 / 에어노즐 교체및 타임 설정 정상작동

Dataiku 기능 – Code recipes

Code recipes




```python
# -*- coding: utf-8 -*-
import dataiku
import pandas as pd, numpy as np
import re
from dataiku import pandasutils as pdu
from collections import Counter
from wordcloud import WordCloud

# ========================
# 0) 설정 (여기만 바꾸면 됨)
# ========================
INPUT_DATASET = "julgok_data"
TEXT_COL = "참고사항"

FONT_FOLDER_ID = "M14u2O0Z"   # <-- 폰트(.ttf) 올려둔 Managed Folder "ID"
FONT_FILE_NAME = "NanumGothic.ttf"

OUT_FOLDER_ID = "UrROafEo"  # <-- 결과 PNG 저장할 Managed Folder "ID"
OUT_PNG_NAME = "wordcloud.png"

TOP_N = 30

# (선택) 자주 제외할 단어들
STOPWORDS = set([
    "", "및", "등", "관련", "확인", "진행", "요청", "조치", "발생", "정상", "이상"
])

# ========================
# 1) 데이터 불러오기
# ========================
df = dataiku.Dataset(INPUT_DATASET).get_dataframe()
TEXT_COL = "참고사항"
texts = df[TEXT_COL].dropna().astype(str)

# ========================
# 2) 전처리 + 토큰화
#    - 한글/숫자/공백만 남기고 분리
# ========================
def normalize(s: str) -> str:
    s = re.sub(r"[^가-힣0-9\s]", " ", s)
    s = re.sub(r"\s+", " ", s).strip()
    return s
```

Dataiku 기능 – Code recipes

Code recipes

wordcloud

- General
- Packages to install
- Currently installed packages
- Permissions
- Containerized execution
- Resources
- Logs
- Usages

SAVE | SAVE AND UPDATE

☑ Update all packages

Updates packages in the virtual environment so that the installed packages match the list and versions of the packages specified by the user.

☑ Run resources init script

☐ Rebuild env

Clears the code environment and rebuilds it from scratch

Packages to install

Install core packages ☑ Install mandatory set of packages (you won't be able to use Dataiku APIs without this)

Install Jupyter support ☑ Install support for Jupyter notebooks

The following packages were previously "Requested" and are currently installed, but have been removed from the requested list. These packages may become unavailable. Check whether this is desired.
regex

Requested packages (Pip)

Add here the packages that you want to be available in the environment. **Do not remove packages once they are installed.** Only packages mentioned here are guaranteed to be installed.

ADD SETS OF PACKAGES

```
1 wordcloud
```

Syntax help

For information: base packages (Pip)

These are the packages required for the selected core features. These packages cannot be removed nor reversioned

```
pandas>=2.3,<2.4
numpy<2.4
python-dateutil >=2.8.2,<3
requests<3

decorator==5.1.1
ipykernel==6.23.3
ipython>=8.12,<8.13
ipython_genutils==0.2.0
jupyter_client==6.1.12
jupyter_core==4.12.0
pexpect==4.8.0
pickleshare==0.7.5
ptyprocess==0.7.0
pyzmq<26
simplegeneric==0.8.1
tornado>=6.3,<6.6
traitlets==5.9.0
```

Currently installed packages

Installed packages (Pip)

These are the packages that are **currently** installed in the code env. Note that only the list in "Packages to install" is authoritative as to which packages will be installed.
Packages that are not in "Packages to install" may not be installed when rebuilding this env or in containers.
This list only reflects the "local" code env, not what is available in containers.

```
asttokens==3.0.1
backcall==0.2.0
certifi==2025.11.12
charset-normalizer==3.4.4
comm==0.2.3
contourpy==1.3.0
cycler==0.12.1
debugpy==1.8.19
decorator==5.1.1
executing==2.2.1
fonttools==4.60.2
idna==3.11
importlib_resources==6.5.2
ipykernel==6.23.3
ipython==8.12.3
ipython-genutils==0.2.0
jedi==0.19.2
jupyter-client==6.1.12
jupyter_core==4.12.0
kiwisolver==1.4.7
matplotlib==3.9.4
matplotlib-inline==0.2.1
nest-asyncio==1.6.0
numpy==2.0.2
packaging==25.0
pandas==2.3.3
parso==0.8.5
pexpect==4.8.0
pickleshare==0.7.5
pillow==11.3.0
prompt_toolkit==3.0.52
psutil==7.1.3
ptyprocess==0.7.0
pure_eval==0.2.3
Pygments==2.19.2
pyparsing==3.2.5
python-dateutil==2.9.0.post0
pytz==2025.2
pyzmq==25.1.2
regex==2026.1.15
requests==2.32.5
simplegeneric==0.8.1
six==1.17.0
stack-data==0.6.3
tornado==6.5.4
traitlets==5.9.0
typing_extensions==4.15.0
tzdata==2025.3
urllib3==2.6.2
wcwidth==0.2.14
wordcloud==1.9.4
zipp==3.23.0
```

```
wcwidth==0.2.14
wordcloud==1.9.4
zipp==3.23.0
```

Dataiku 기능 – Plugins

플러그인(Plugins)

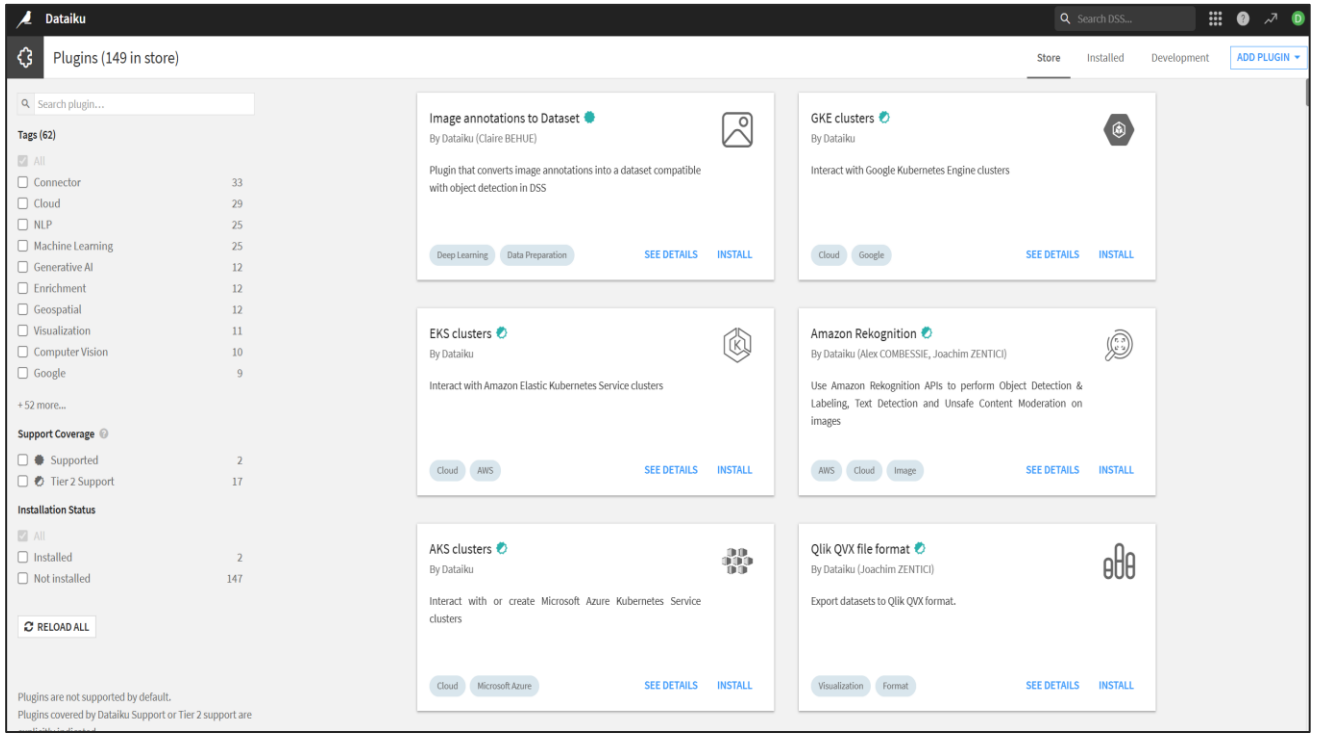
- **플러그인?**

  Dataiku에서 기본 제공 기능 외에, 사용자가 필요에 따라 **새로운 기능을 확장할 수 있도록 만든 모듈로** 제공되는 플러그인을 **설치**하거나 **직접 생성** 가능

- **플러그인으로 할 수 있는 일**
  - 레시피(Recipes) : 커스텀 데이터 처리, 알고리즘 실행
  - 데이터 커넥터(Connectors) : 새로운 데이터 소스/타겟 연결
  - 매크로(Macros) : 프로젝트/데이터셋 단위의 자동화 스크립트
  - 시각화 & Webapps : 맞춤형 차트/웹앱 제공

Dataiku 기능 – Plugins

플러그인(Plugins)



Dataiku 기능 – Plugins

플러그인(Plugins)

Dataiku DSS

Plugins / Feature reduction master Summary

ADD

- feature-reduction
  - custom-recipes
    - partial-least-square
      - recipe.json 0.8 KB
      - recipe.py 1.49 KB
    - stat-selection
      - recipe.json 2.46 KB
      - recipe.py 6.6 KB
  - python-lib
  - plugin.json 0.89 KB

recipe.py recipe.json

feature-reduction/custom-recipes/stat-selection/recipe.py

```python
# -*- coding: utf-8 -*-
import warnings
warnings.filterwarnings('ignore')

import dataiku
import pandas as pd
import numpy as np
import statsmodels.api as sm
import statsmodels.formula.api as smf
from dataiku.customrecipe import get_input_names_for_role, get_output_names_for_role, get_recipe_config

# --- Load Configs ---
config = get_recipe_config()
method = config.get("method", "stepwise").lower()
target_col = config.get("target")
significance_level = float(config.get("significance_level", 0.05))
fixed_column = config.get("fixed_column", [])
deleted_column = config.get("deleted_column", [])

# --- Load Datasets ---
input_dataset_name = get_input_names_for_role("input_dataset")[0]
ref_dataset_name = get_input_names_for_role("reference_dataset")[0]
output_dataset_name = get_output_names_for_role("output_dataset")[0]
output_summary_name = get_output_names_for_role("output_summary")[0]

df_input = dataiku.Dataset(input_dataset_name).get_dataframe()
tag_info = dataiku.Dataset(ref_dataset_name).get_dataframe()
```

Dataiku DSS

Plugins / Feature reduction master Summary

ADD

- feature-reduction
  - custom-recipes
    - partial-least-square
      - recipe.json 0.8 KB
      - recipe.py 1.49 KB
    - stat-selection
      - recipe.json 2.46 KB
      - recipe.py 6.6 KB
  - python-lib
  - plugin.json 0.89 KB

recipe.py recipe.json

feature-reduction/custom-recipes/partial-least-square/recipe.json

```json
{
  "kind": "PYTHON",

  "meta": {
    "label": "PLS Dimensionality Reduction",
    "description": "Reduce dimensions using Partial Least Squares"
  },

  "inputRoles": [
    {
      "name": "input_dataset",
      "label": "Input dataset",
      "arity": "ONE_DATASET"
    }
  ],

  "outputRoles": [
    {
      "name": "output_dataset",
      "label": "Output dataset",
      "arity": "ONE_DATASET"
    }
  ],

  "params": [
    {
      "name": "target",
      "label": "Target column name",
      "type": "STRING",
      "mandatory": true
```

Statistical Variable Selection Ver1.0

OLS와 KFlod 기반 변수 선택법 (TAG 데이터만 출력 됩니다)

타겟 변수	target	구하고자 하는 변수를 선택해주세요 (이외의 QM_MD/TD 포함된 컬럼은 자동 제외)
변수 선택 방법	Stepwise Selection	변수 선택 방법을 지정해주세요 (기본값 = Stepwise)
K-Fold 교차검증 수	5	변수 선택에 사용할 K-Fold 수 (1을 입력하면 KFold 사용 안 함)
유의수준 (alpha)	0.05	변수 선택에 사용되는 기준 p-value 값을 정해주세요 (기본값 = 0.05)
필수 태그	+ 반드시 포함할 변수를 지정해주세요	
제외 태그	+ 반드시 제외할 변수를 지정해주세요	

Dataiku 기능 – Saved Models

Saved Models

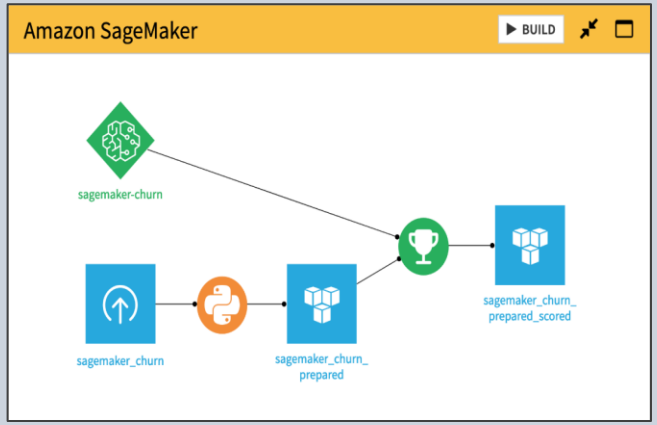
- **Dataiku가 아닌 외부에서 배포된 모델을 Dataiku 내에서 사용 가능**(DSS 14 이상 제공)

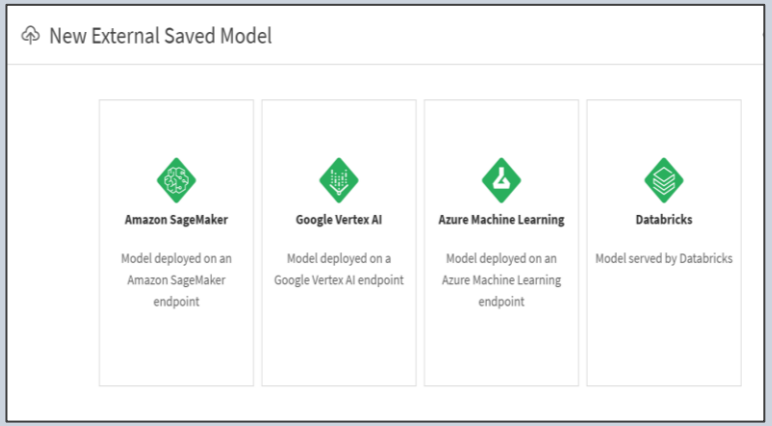
- 연동 가능한 모델
  - **Amazon SageMaker, Google Vertex AI, Azure Machine Learning, Databricks**
  - **로컬 환경에서 만든 모델**(ex. pkl, pt)도 **MLflow** 라이브러리를 통해 활용 가능

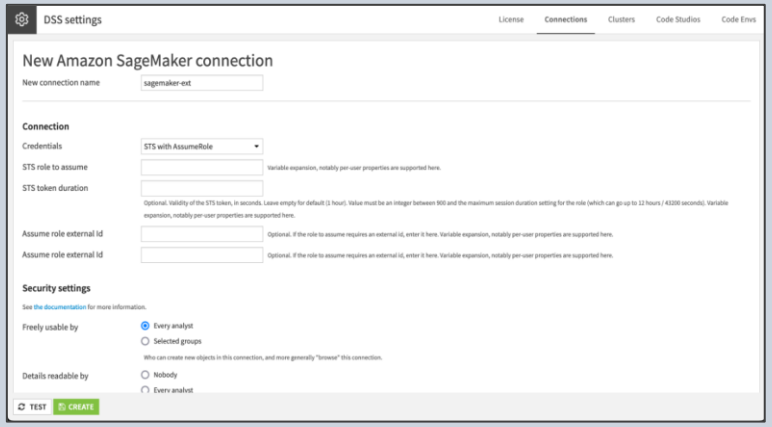
- **Dataiku의 모델 관리 기능을 사용 가능**
  - 모델 버전 관리
  - Scoring recipe를 통한 예측 수행
  - Model Comparison을 통한 여러 모델(또는 버전) 비교 수행
  - 다른 데이터셋에 대한 성능 분석 및 모델 평가
  - Analyzing Drift를 통한 모델 변화 모니터링

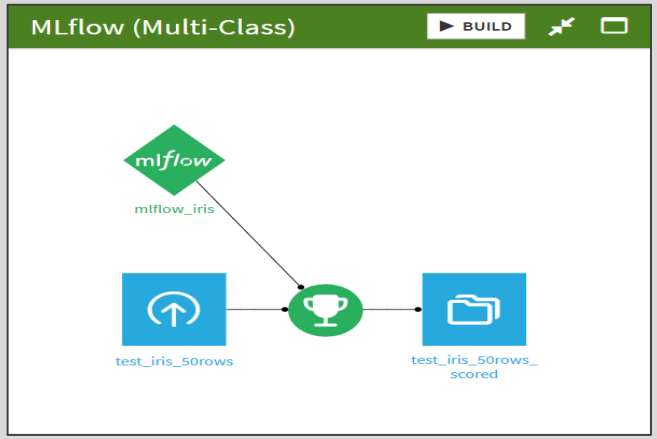
Dataiku 기능 – Saved Models

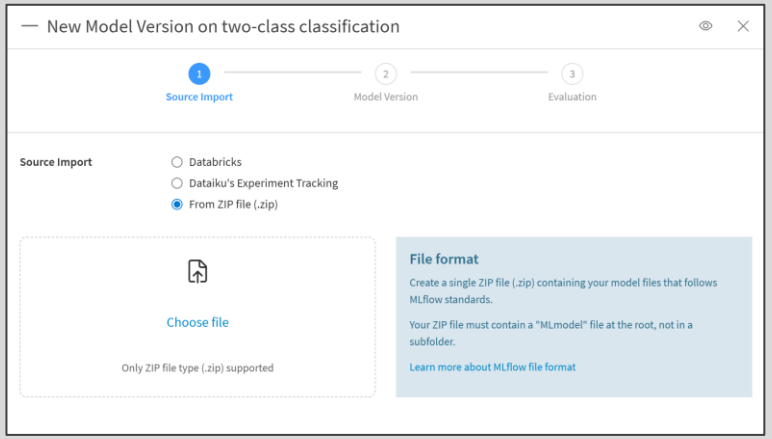
Saved Models

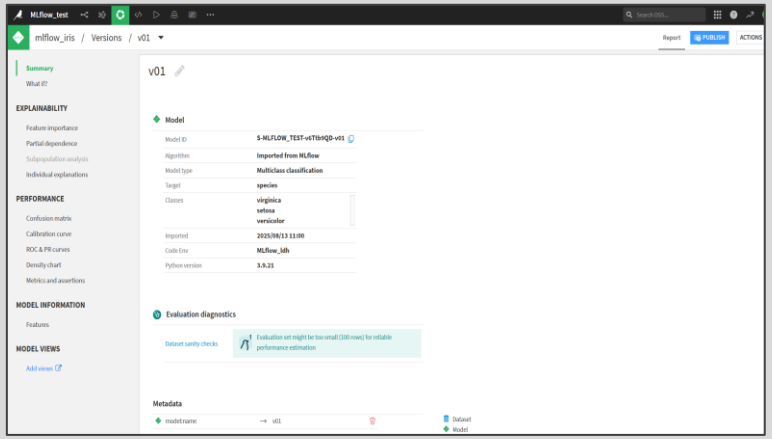






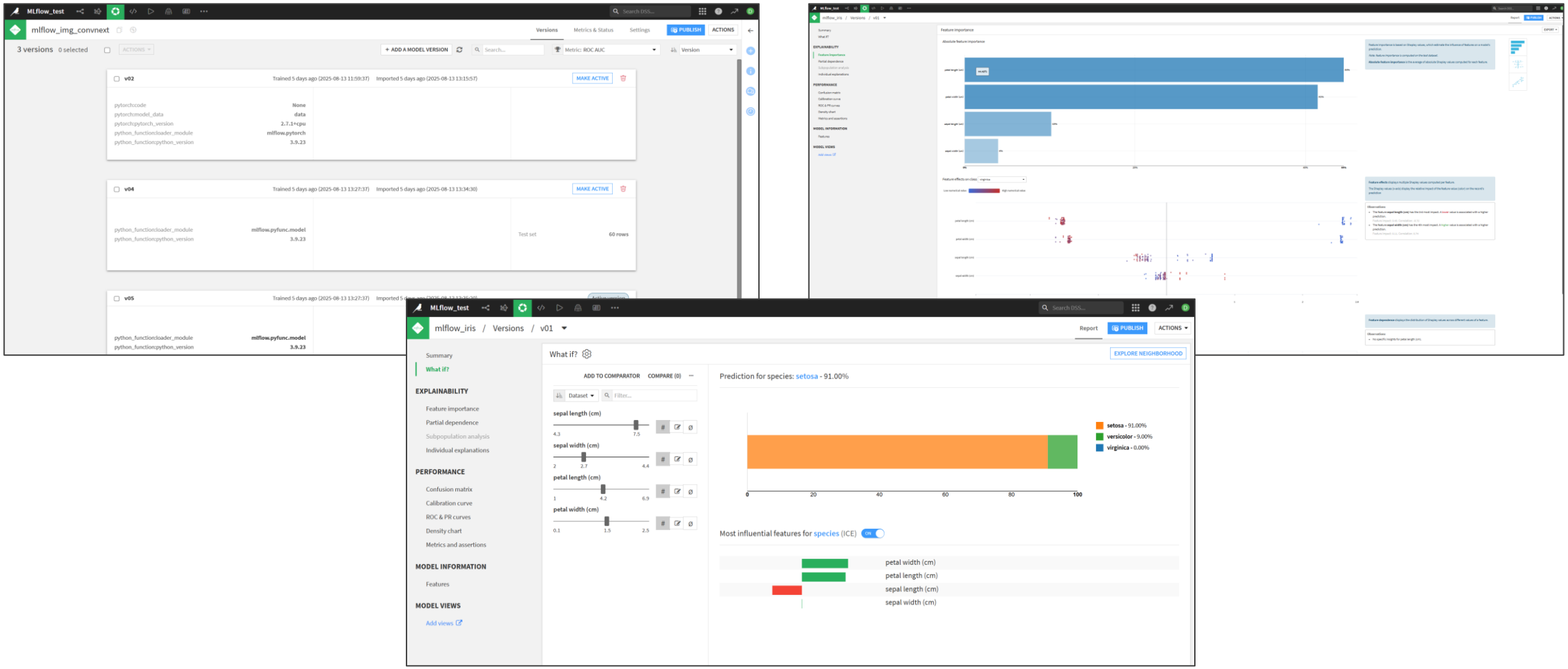






Dataiku 기능 – Saved Models

Saved Models

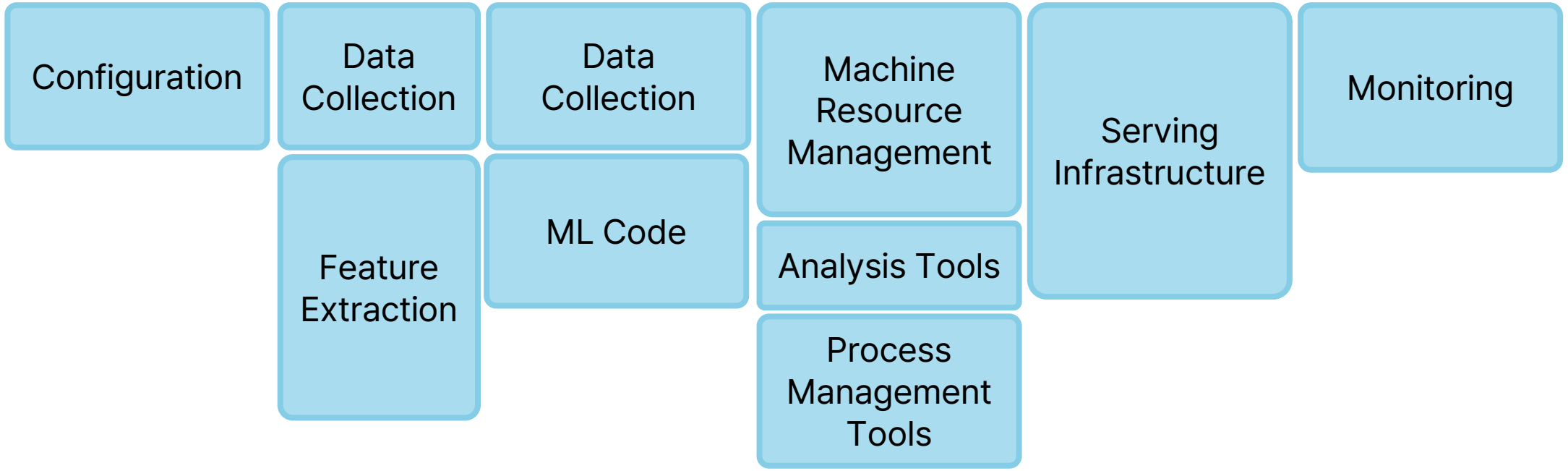


MLOps 개요 및 필요성

P&D
Solution

MLOps의 필요성

- 많은 프로젝트에서 AI가 다양한 분야에서 발생한 문제를 풀 수 있을 것 같다는 가능성만 확인함



대부분의 프로젝트가 POC(Proof of Concept) 단계에서 머물러 있는 이유는
머신러닝과 딥러닝 개발에 사용된 코드만으로는 실제 서비스를 운영할 수 없기 때문임

MLOps의 필요성

MLOps의 필요성

- 많은 프로젝트에서 AI 가 다양한 분야에서 발생한 문제를 풀 수 있을 것 같다는 가능성만 확인함

- 서비스 초기에는 팀 또는 회사 규모가 작아 구분이 필요 없지만, 서비스 규모가 커질수록 개발 팀과 운영 팀이 분리되면서 물리적인 한계가 발생

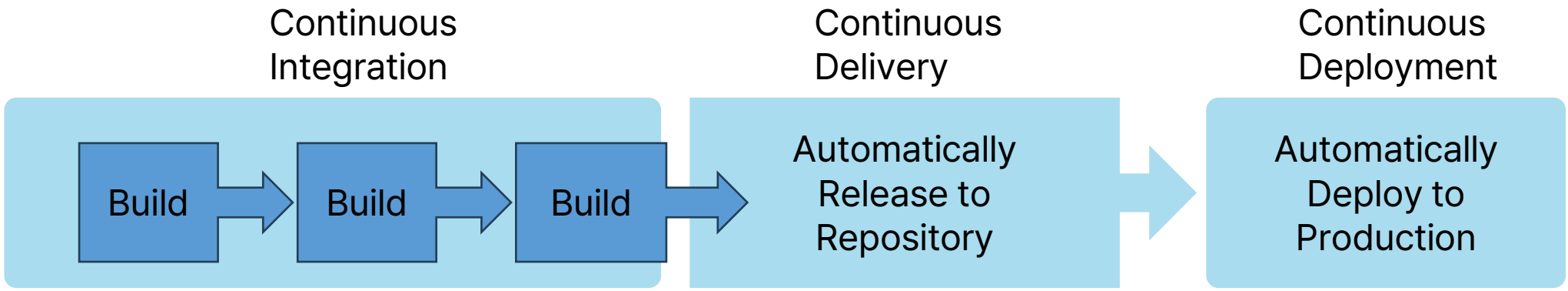
- 이는 사일로(Silo) 현상이 발생하는 원인이 될 수 있음
  *사일로 현상(Silo effect): 조직 내 부서 간 장벽이나 부서 이기주의로 인해 서로 협력하지 않는 현상

- 이러한 사일로 문제를 해결하기 위해 DevOps가 등장함

MLOps의 필요성

MLOps의 필요성

많은 프로젝트에서 AI가 다양한 분야에서 발생한 문제를 풀 수 있을 것 같다는 가능성만 확인함



DevOps는 CI(Continuous Integration)와 CD(Continuous Delivery) 파이프라인을 통해 서로 협력할 수 있음

- CI(Continuous Integration) : 지속적인 통합(예 : 모델 build, test 등에 사용된 코드 변경 사항 저장)
- CD(Continuous Delivery) : 지속적인 제공/배포(예 : 앱 개발 시간 및 릴리스 수 단축)

MLOps의 필요성

- MLOps(Machine Learning Operations)란?

MLOps는 DevOps라는 개발 방법론에서 파생된 단어

MLOps는 머신러닝과 딥러닝의 워크플로우(Workflow) 및 배포를 자동화하고 단순화하는 일련의 과정을 의미함

MLOps는 ML(머신러닝)과 Ops(운영)의 합성어로,
AI모델의 개발부터 배포, 모니터링, 재학습까지의 전과정을 자동하고 관리하는 체계를 의미

MLOps의 필요성

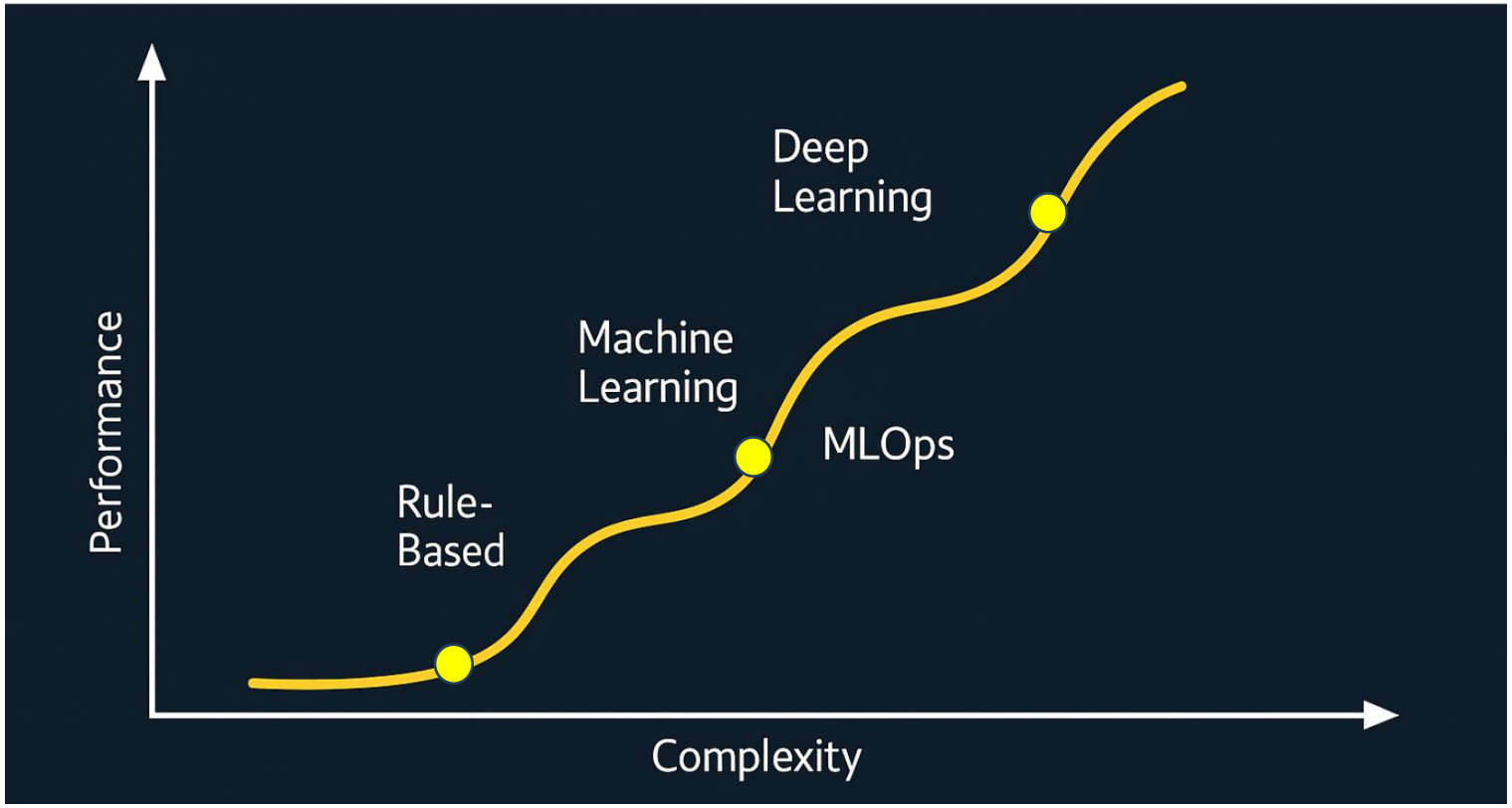
MLOps의 필요성

Rule Based	간단한 규칙 기반으로 아이템 추천
	특별한 이유가 없는 이상 모델 수정 필요 없음
Machine Learning	서비스 규모가 커지면서 데이터가 증가함
	데이터 기반의 머신러닝 모델 생성 후 아이템 추천
Deep Learning	개인화 추천 요구 증가 및 성능 개선이 필요함
	딥러닝 모델을 사용하여 아이템 추천

대규모 데이터 처리 및 최적의 모델을 구축하고 데이터에서
유의미한 인사이트를 도출하기 위해서는 MLOps 가 효과적임

MLOps의 필요성

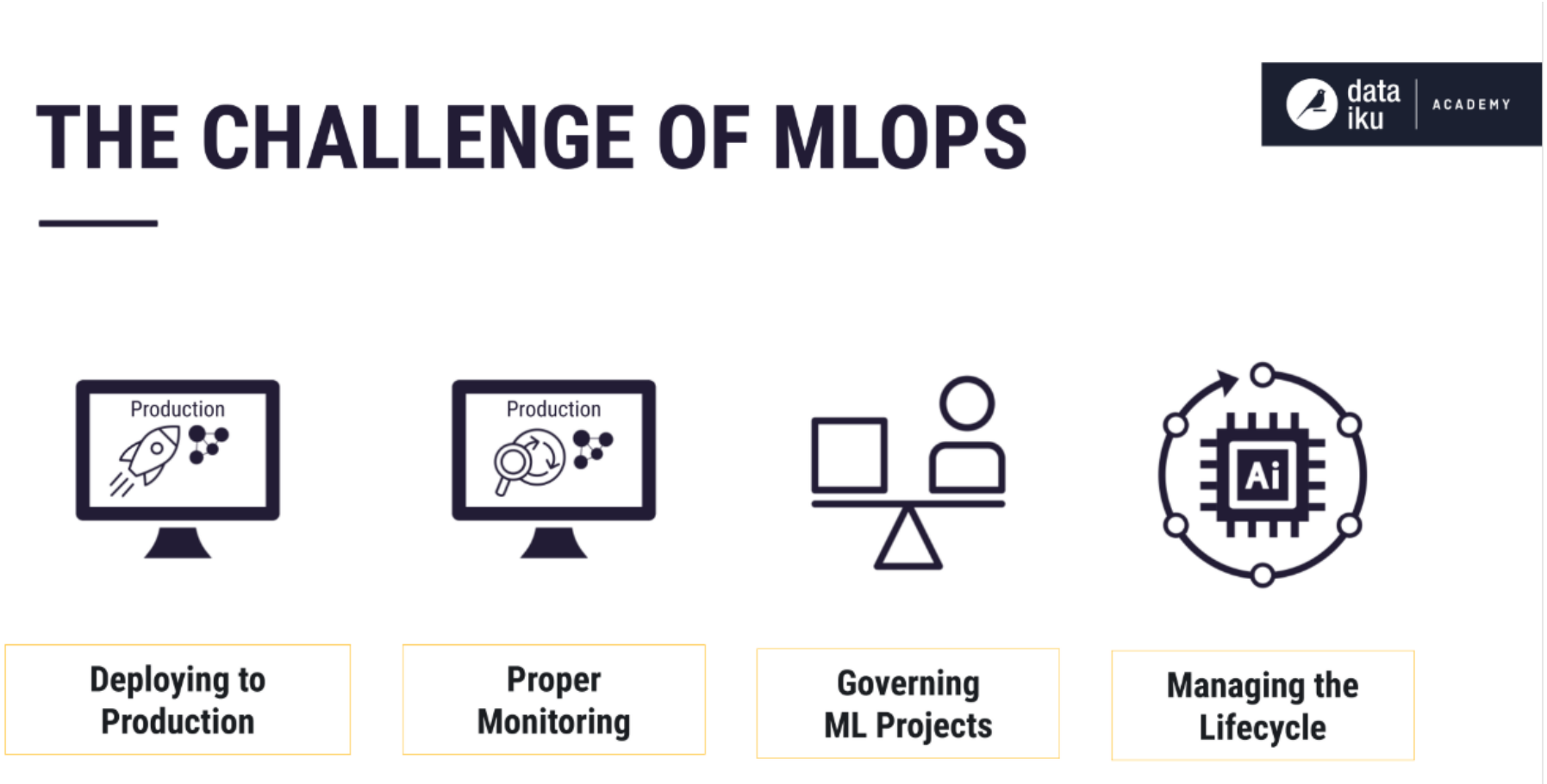
MLOps의 필요성



1. 모델의 복잡도가 증가할수록 모델의 성능도 같이 증가(상승관계)
2. 개발한 모델이 정상적으로 배포될 수 있는 지 CI/CD를 하는 것이 MLOps의 목표

MLOps 과제

- ML 모델을 관리하기 어려운 이유



MLOps 기본 원칙

- 배포, 안전하고 견고한 환경, 관리, 업데이트 가능

모델 개발이 MLOps 에 미치는 영향

배포 후 MLOps에 영향을 미칠 수 있는 모델 개발의 핵심 구성 요소

1. 사업 목표와의 일치 : 비즈니스 목표는 모델 개발의 방향성을 제시해야 함

- 우리가 해결하고자 하는 비즈니스 문제는 무엇인가? 이 모델은 어떻게 그 문제를 해결할 것인가?
- 어떤 수준의 모델 성과가 기업에서 수용 가능한가?
- 해당 모델이 사업에 부정적인 영향을 미치는가, 긍정적인 영향을 미치는가?
- 시간 경과에 따른 모델 모니터링을 통해 모델 성능 저하를 감지하는 방법은 무엇입니까?
- 실제 운영 환경에서 머신 러닝 모델의 성능과 유지관리를 담당하는 사람은 누구인가요?

2. 탐색적 데이터 분석(EDA)

- 데이터 품질이 충분한가요? 예를 들어, 데이터가 완전하고 적절한 형식으로 제공되나요?
- 데이터를 합법적으로 사용할 수 있나요?
- 이해관계자가 데이터를 이해하고 이를 통해 분석을 도출할 수 있을까요?
- 생산에 들어가면 실시간으로 데이터를 사용할 수 있나요?
- 데이터가 변경됨에 따라 모델이 예상대로 동작하는지 확인하기 위해 모델 성능을 지속적으로 모니터링하고 평가하려면 어떻게 해야 할까요?

모델 개발이 MLOps 에 미치는 영향

배포 후 MLOps에 영향을 미칠 수 있는 모델 개발의 핵심 구성 요소

3. 기능 엔지니어링 및 선택

- EDA와 비즈니스 전문성에 따라 기존 데이터에서 어떤 기능을 생성할 수 있습니까?
- 해당 기능이 비즈니스 목표에서 발견된 문제를 해결하는 데 도움이 됩니까?
- 기능에 대해 어느 정도의 설명이 필요할까요?
- 해당 기능이 관련성이 없고 데이터 품질이 좋지 않습니까?
- 시간이 지나도 이러한 특징을 어떻게 유지할 수 있을까?

4. 재현성

- 디자인 환경이 잘 문서화되어 있나요?
- 동일한 모델 결과를 실제 생산에서 재현할 수 있나요?
- 관련된 모든 자산과 매개변수에 대한 버전 제어가 있습니까?

모델 개발이 MLOps 에 미치는 영향

배포 후 MLOps에 영향을 미칠 수 있는 모델 개발의 핵심 구성 요소

5. 책임 있는 AI

- 검증된 비즈니스 목표에 맞춰 모델이 동작하는지 어떻게 확인할 수 있나요?
- 고객, 직원, 사용자 및 시민의 데이터 개인 정보를 보호하기 위해 모델 학습 과정에서 어떤 보호된 특성(민족, 성별, 나이, 종교 등)을 생략할 수 있습니까?
- 특정 그룹에 대한 모델 편향과 불공정성을 어떻게 설명하고 완화할 수 있을까?
- 원래 의도된 용도를 넘어 얼마 동안 합법적으로 데이터를 보관할 수 있습니까?
- 우리가 데이터를 수집하고 저장하는 수단이 GDPR과 같은 규제 기준과 우리 회사의 표준에 부합합니까?
- 시간이 지나도 책임감 있는 AI를 보장하려면 어떻게 해야 할까?

6. 훈련, 평가 및 드리프트

- 모델을 개발하고 선택할 때 어떤 성과 지표를 측정합니까?
- 이 모델은 얼마나 설명이 가능합니까?
- 모델을 배포하는 것은 얼마나 쉬운가요?
- 이 모델은 서로 다른 그룹의 구성원을 어떻게 대하는가?(즉, 이 모델은 공정한가?)
- 머신 러닝 모델은 하루, 한 달, 일 년 후에도 여전히 비즈니스 목표에 부합할까요?
- 모델을 업데이트하기 위해 데이터 드리프트를 추적할 수 있나요?

MLOps in Dataiku

Node의 역할

Design Node

데이터분석가, 사이언티스트

- 데이터 준비(Prepare)
- 시각화 및 분석
- 모델링(AutoML)
- 플로우 구성
- 플러그인 개발

Automation Node

운영자, 배치 작업 관리자

- Flow 실행(Build, Scenario 등)
- Scenario 및 Trigger 관리
- 데이터 정기 갱신(배치 기반)
- 알람 설정, 에러 핸들링

API Node

백엔드 개발자, API 사용자

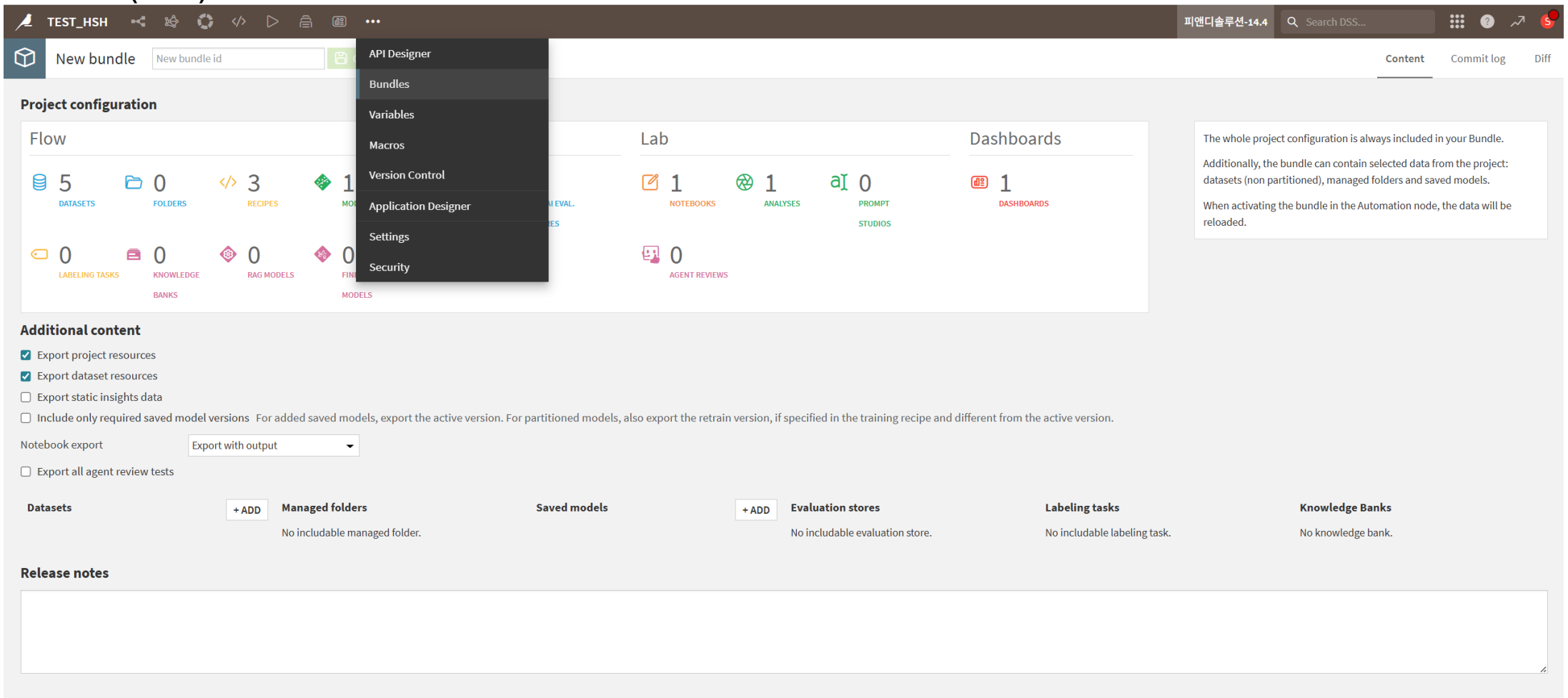
- 학습된 모델 API endpoint 로 배포
- API 키 기반 인증 및 보안
- 예측 요청/ 응답처리 (JSON기반)
- 로드밸런싱, 모니터링 지원

Deployer Node

Govern Node

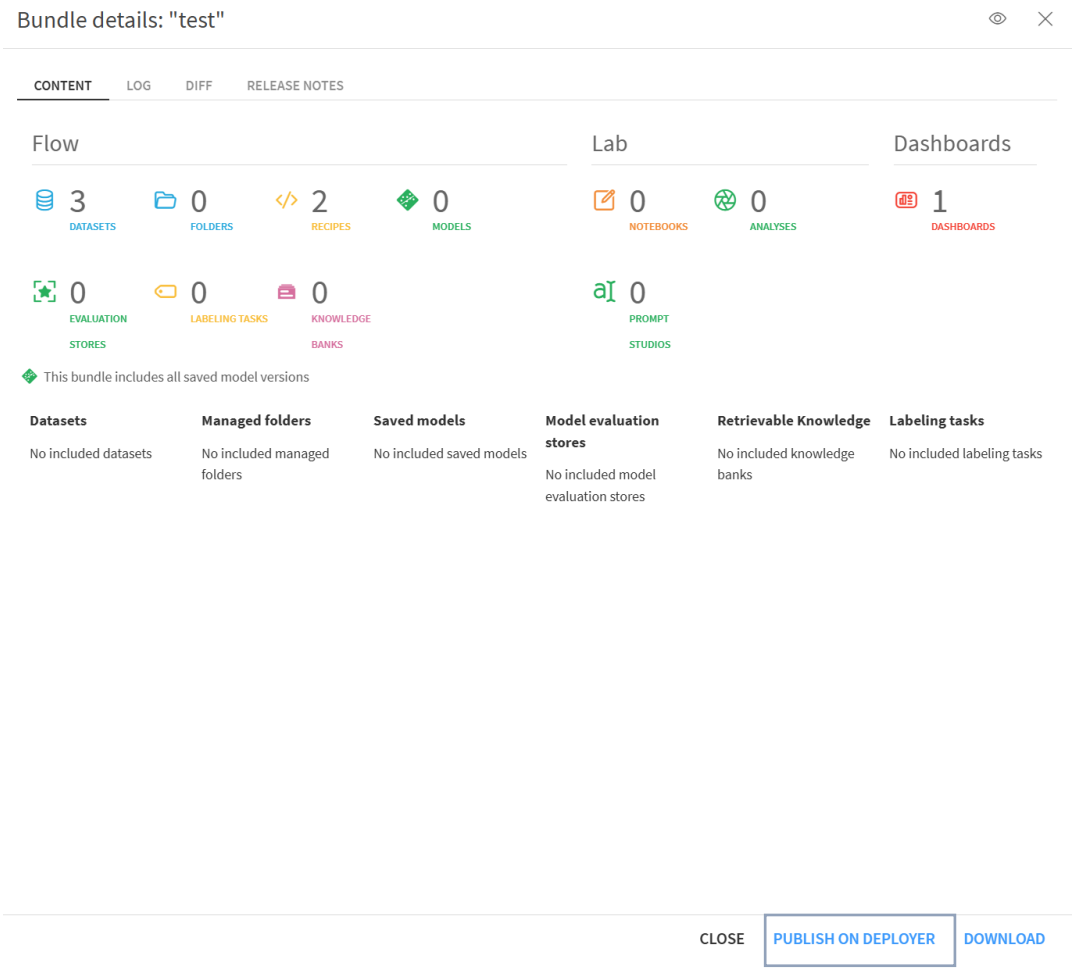
MLOps in Dataiku

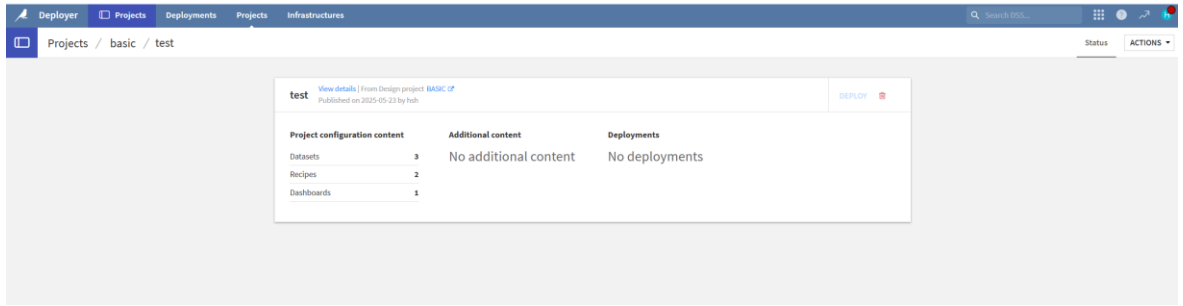
Bundles(번들)



MLOps in Dataiku

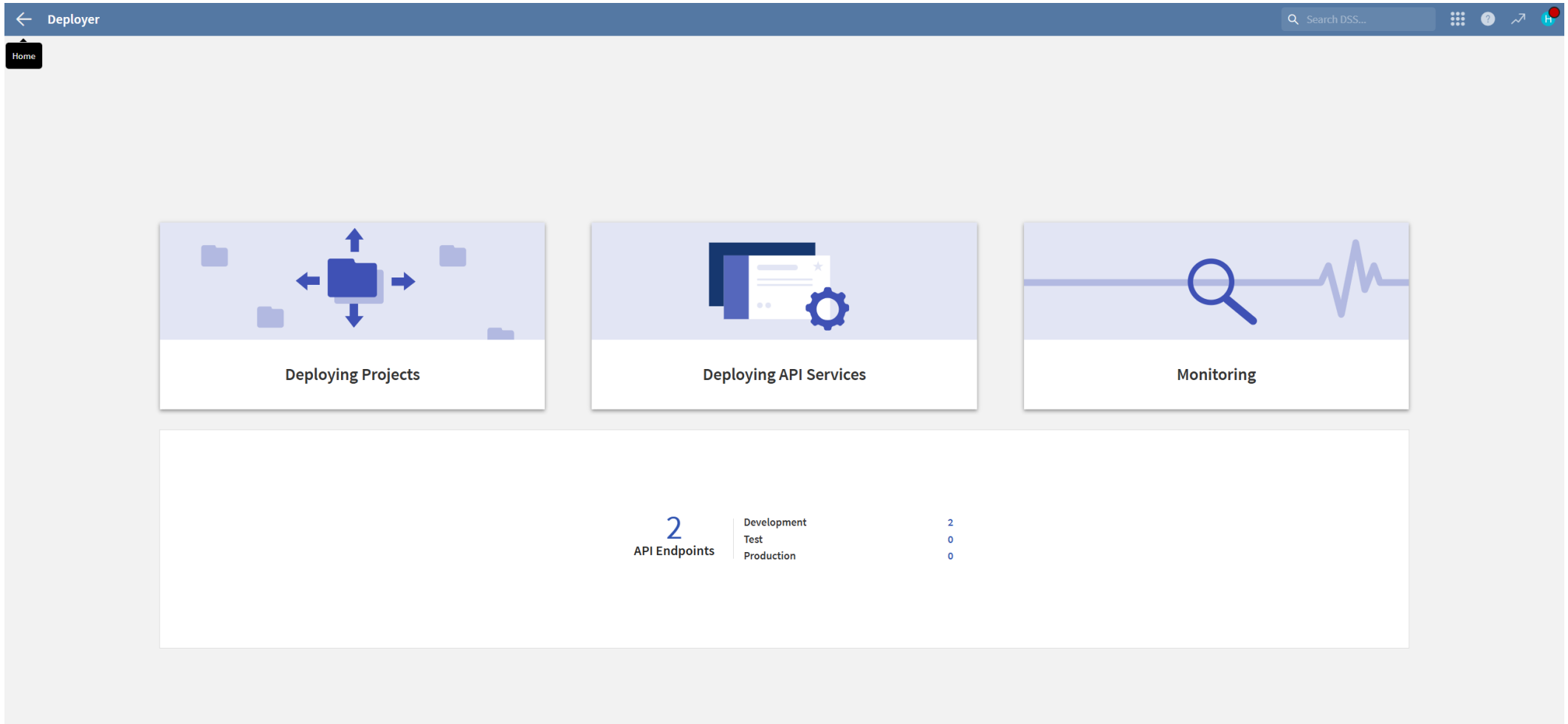
Bundles(번들)





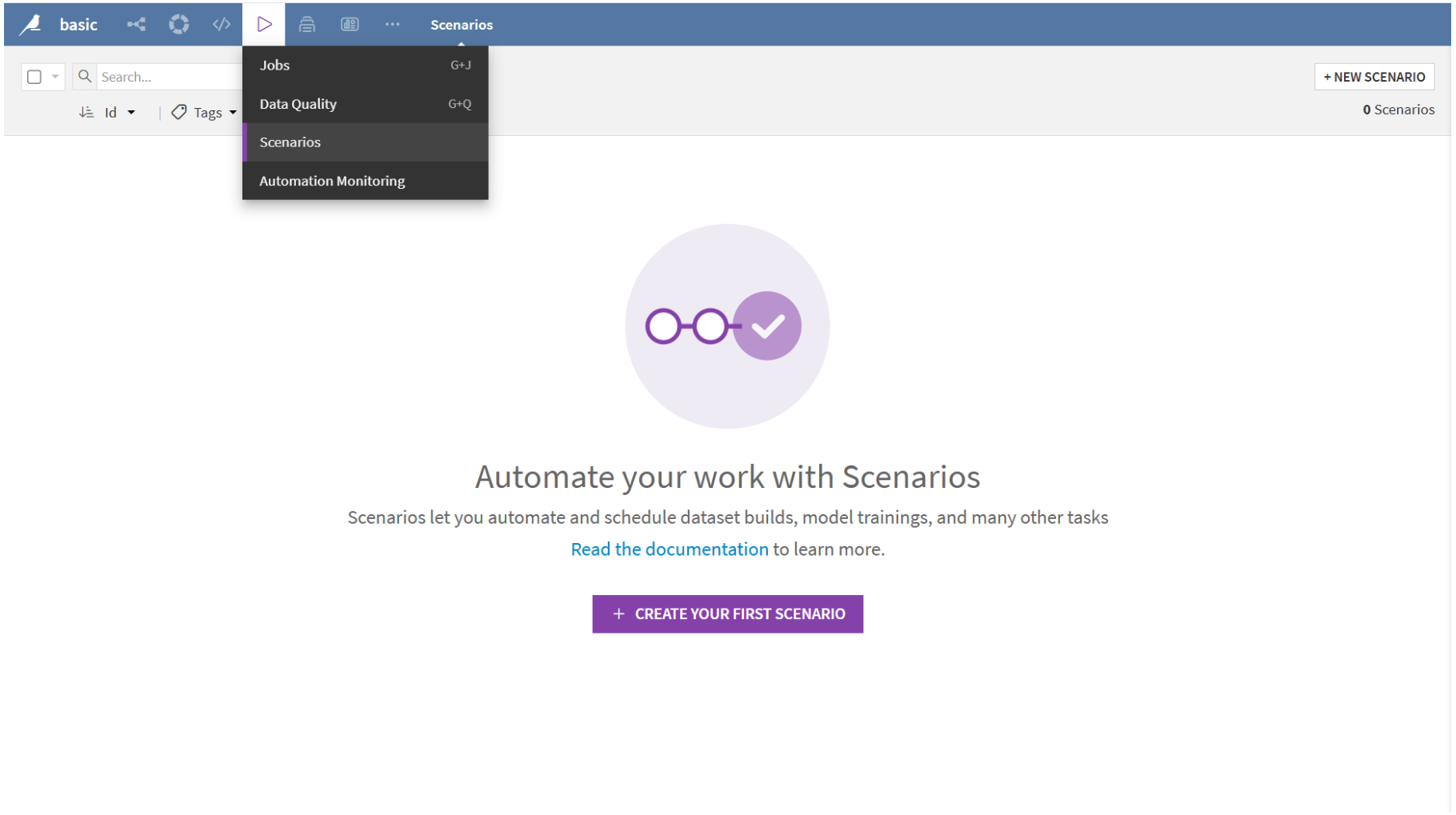
MLOps in Dataiku

- Deployer : 모델 및 프로젝트 배포 관리 도구



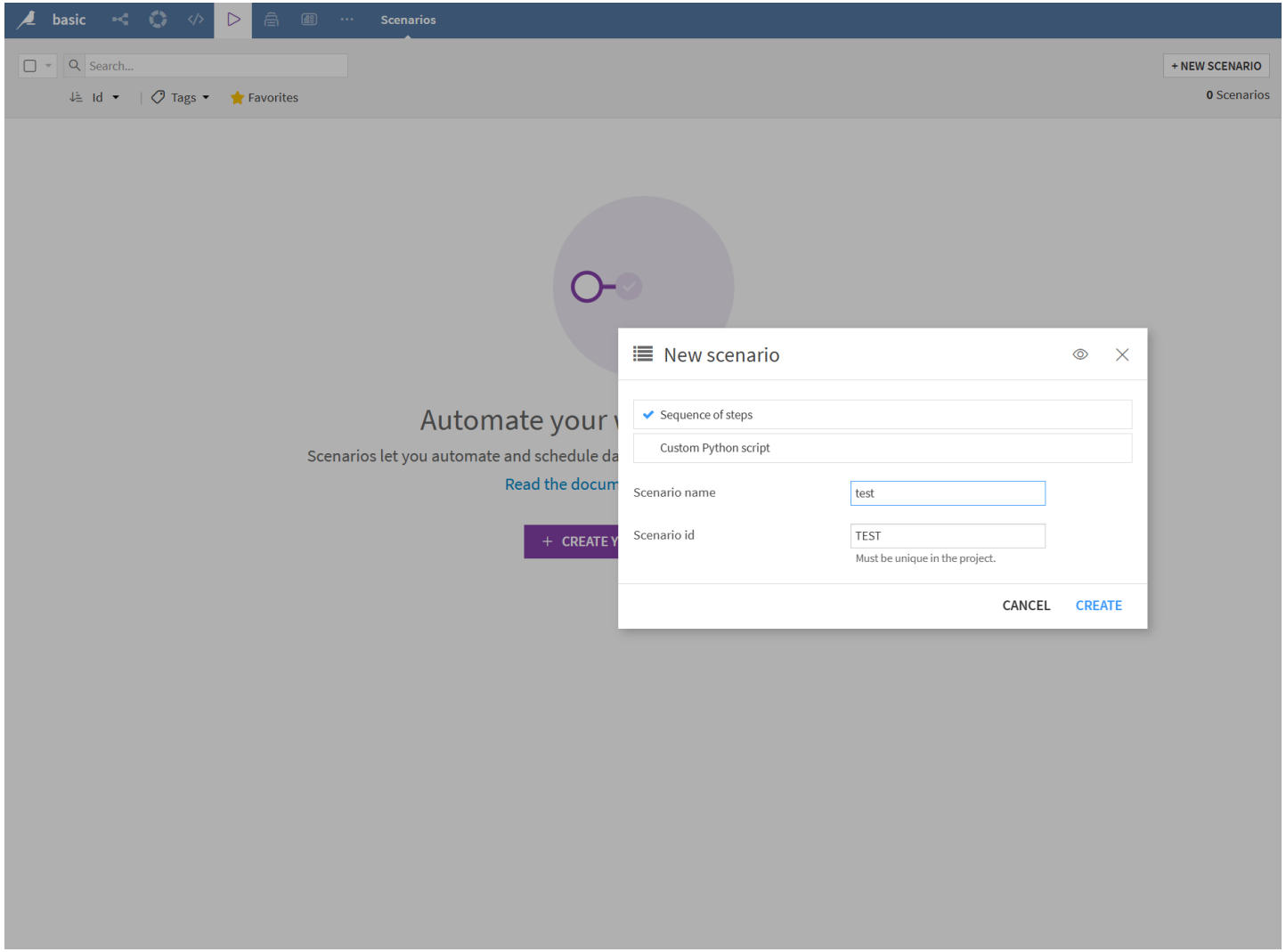
MLOps in Dataiku

Scenario



MLOps in Dataiku

Scenario



시나리오 예시

1. 정기적 데이터 처리

- 매일 새벽에 데이터베이스에서 데이터를 읽어와 정제 후 업데이트

2. 모델 재학습 자동화

- 특정 주기에 모델을 재학습하고 성능 비교 후 최신 모델을 운영에 배포

3. 에러 감지 및 알림

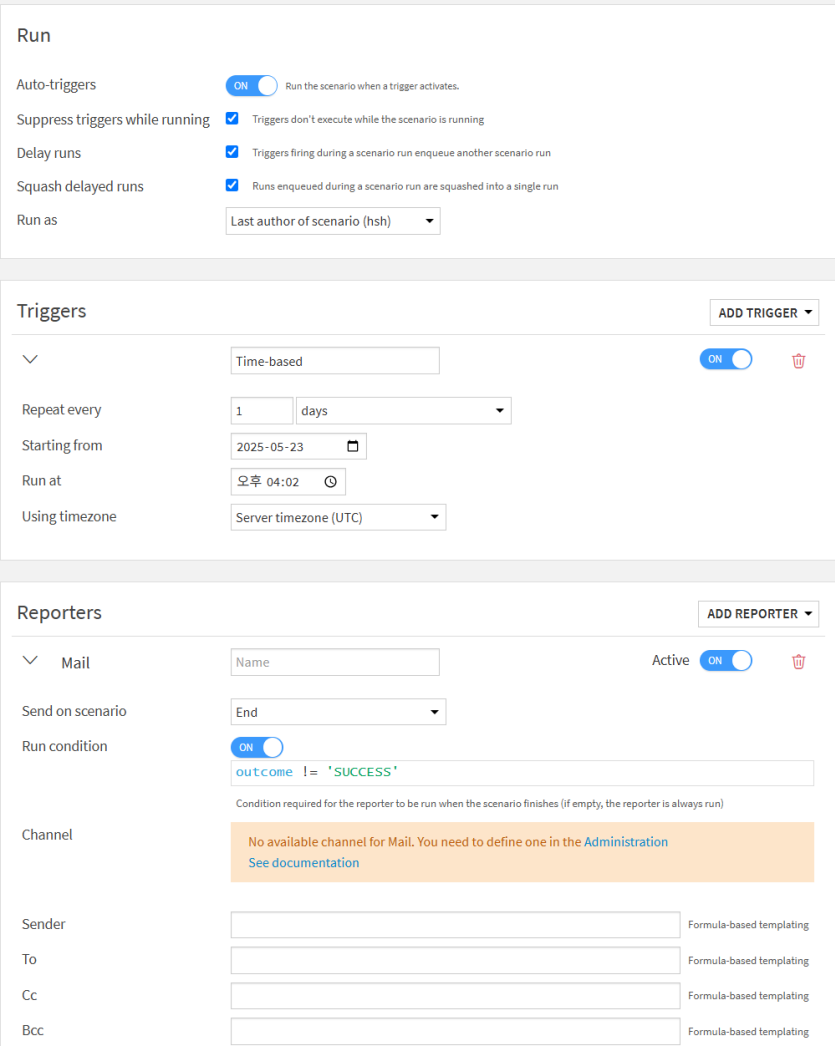
- 데이터 누락, 처리 실패 시 자동으로 슬랙/이메일 알림 전송

4. 리포트 자동 발송

- KPI 대시보드를 PDF로 생성해 매주 팀에 이메일로 발송

MLOps in Dataiku

Scenario



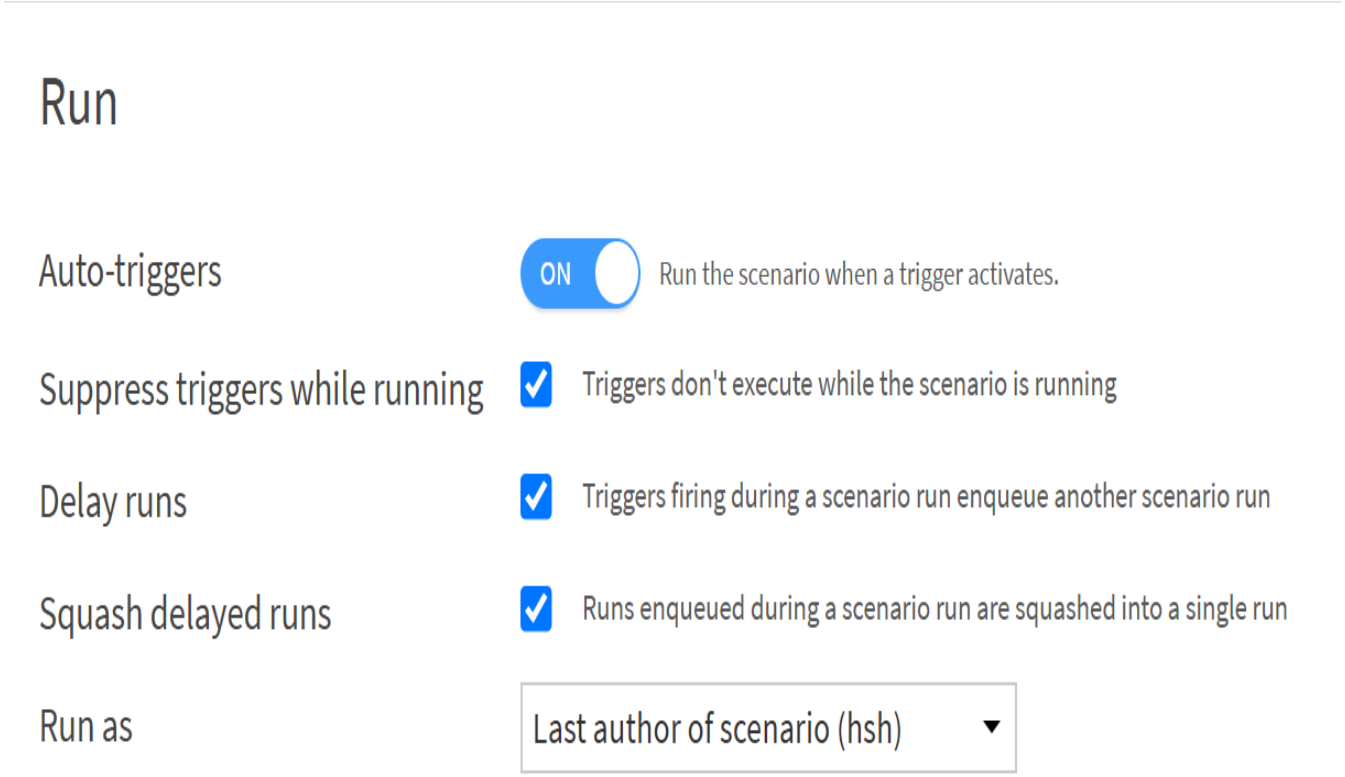
Run : 트리거 중복 방지, 사용자 권한 지정 등의 실행 설정

Trigger : 시나리오를 자동 실행시키는 조건.

예: 매일 6시, 데이터셋 변경 시, 다른 시나리오 완료 후 등

Reporters : 시나리오 실행 결과를 이메일, 슬랙 등으로 알림 설정 가능

Scenario - RUN



Auto-triggers : ON이면, 트리거 조건이 충족될 때 시나리오가 자동 실행됩니다.(예: 일정 시간, 데이터 변경 등)

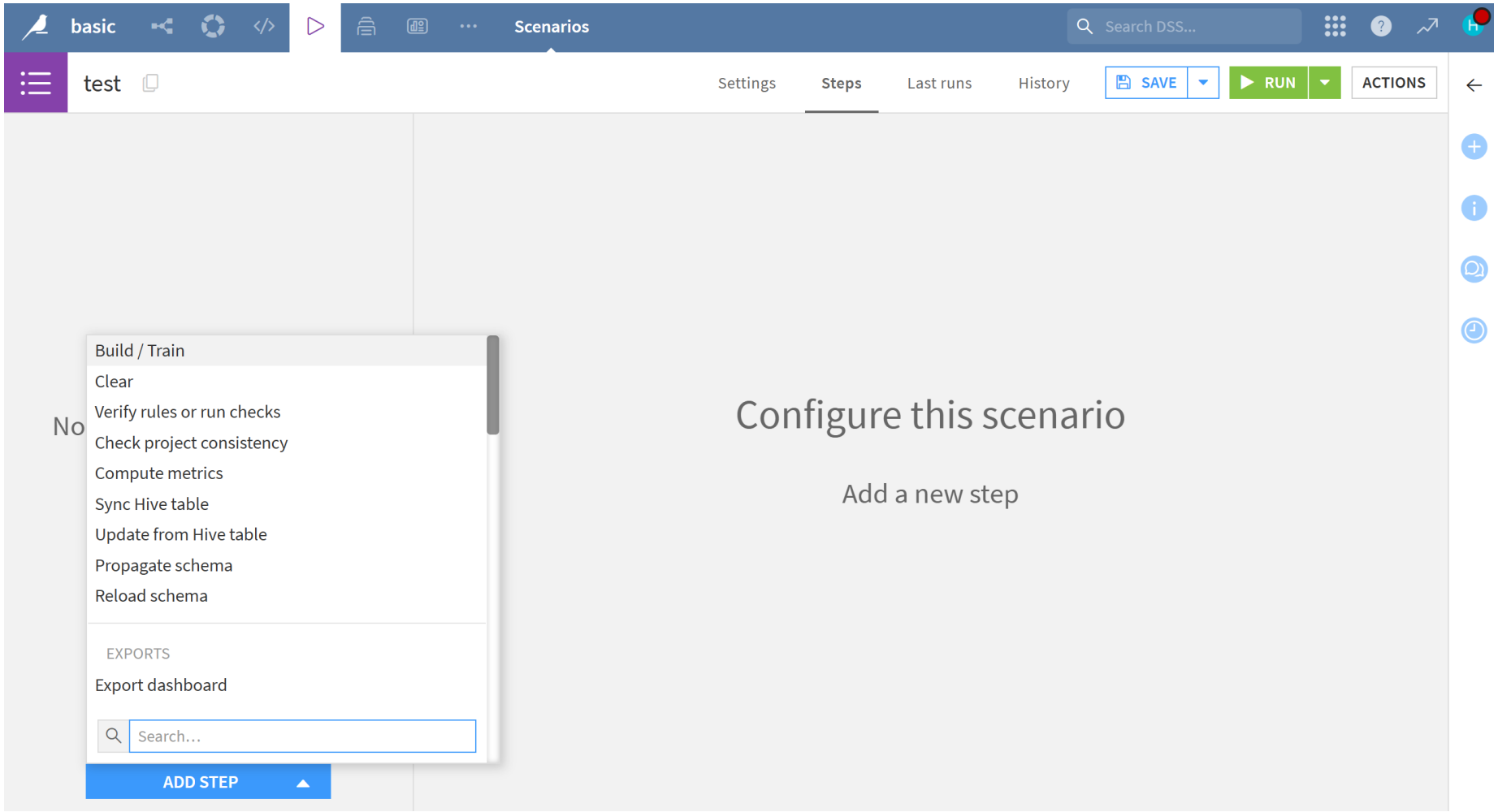
Suppress triggers while running : 시나리오 실행 중일 때 새로운 트리거가 발생해도 무시, 중복실행 방지

Delay runs : 시나리오 실행 중에 트리거가 발생하면 나중에 실행할 수 있도록 예약 (enqueue)

Squash delayed runs : 대기 중인 여러 실행 요청이 있을 경우 한번만 실행이 되도록 병합

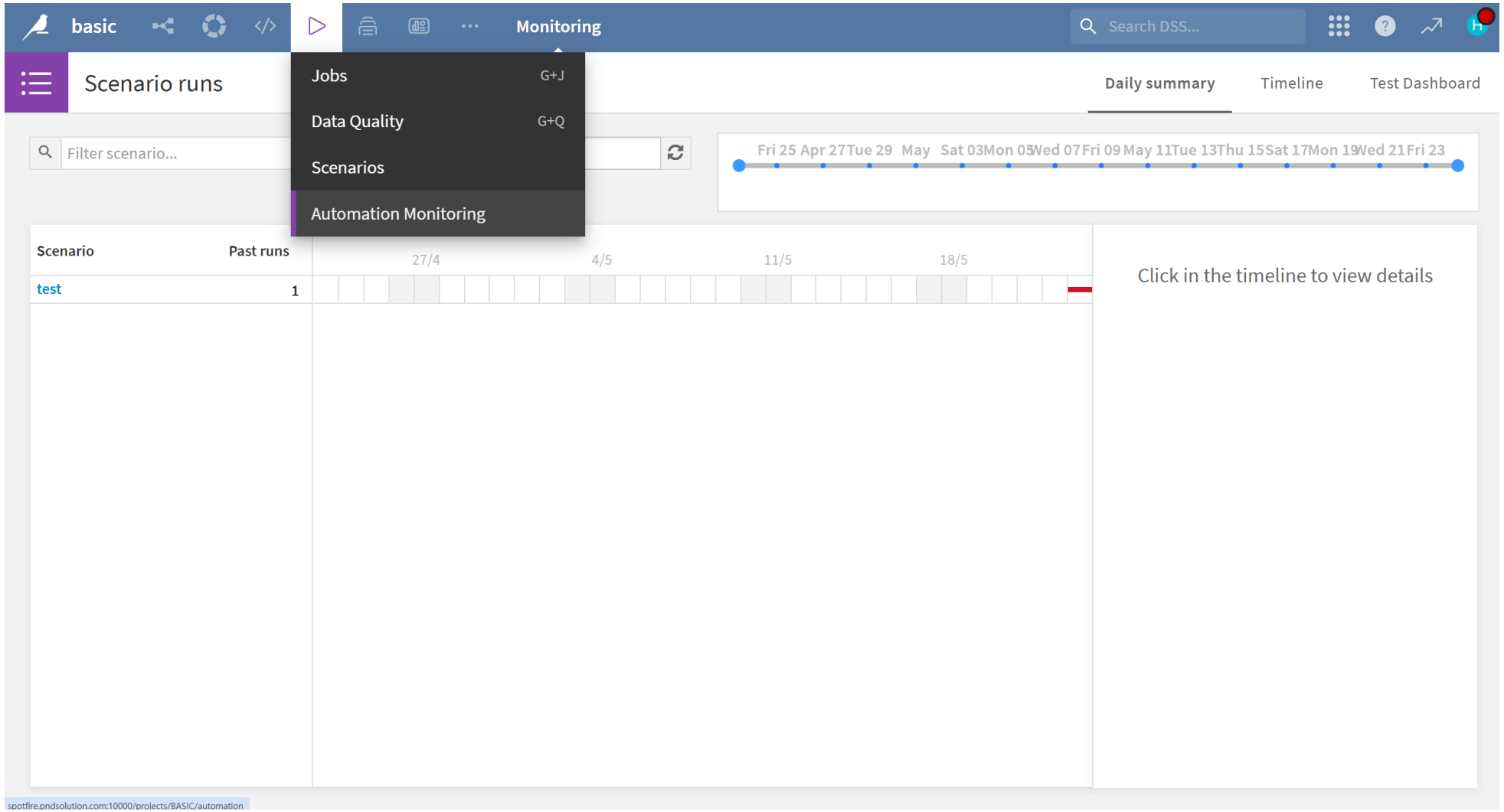
MLOps in Dataiku

Scenario – Steps



MLOps in Dataiku

Automation Monitoring



P&D Solution

Dataiku 활용 사례

P&D
Solution

Dataiku 활용 사례

ML 모델링	절곡기 데이터로 Dataiku를 활용하여 정비 시점을 사전에 정확도 높게 예측하고자 함
AI agents	인사 데이터로 Dataiku를 활용하여 AI agents를 구축하여 챗봇을 만들어 활용
이미지분석	이미지 데이터로 Dataiku를 활용하여 불량원인을 찾는 모델을 만들어 활용

P&D Solution

1. 분석과제 개요

절곡기 00호기 측정 데이터(각도, 거리, 모터 위치 정보)를 활용하여 정비 시점 예측 PoC를 통해 AI 솔루션(Dataiku) 도입 타당성을 평가하고자 함

목표	절곡기 00호기 데이터로 Dataiku를 활용하여 정비 시점을 사전에 정확도 높게 예측하고자 함
예측 및 평가 방법	(1) 각도차 기준(2.5 이상)에 따른 취출 여부를 예측 타겟으로, (2) 각도, 거리, 모터 위치 정보를 활용하여 예측 모델을 생성하고, (3) 특정월 데이터에 적용, 예측한 다음, 실제 정비 이력과 비교함
예측 및 평가 결과	(1) 2가지 모델링 방안을 수립하였고, 방안별 최고 정확도 모델 기준으로 98% 이상의 정확도를 확보함 (2) 모델을 7월 & 11월 데이터에 적용하여 취출 여부를 예측하였고, 7월 & 11월 정비 이력과 비교한 결과, 정비 시점 예측이 가능함

P&D Solution

2. 데이터 설명

2025년 1월 ~ 11월 절곡기 00호기 측정 데이터와 정비 이력 데이터를 활용하였음

측정 데이터	정비 이력 데이터
예측 모델을 생성하기 위해 사용된 타겟 변수 및 설명 변수는 다음과 같음 **(1) 타겟 변수 :** 취출 여부(각도차 이상) **(2) 설명 변수** 1) 각도 : 좌 12개, 우 12개 2) 거리 : 좌 12개, 우 12개 3) 모터 위치 정보 : 1개 **=> 거리 변수는 최종 제외함**	**(1) 모델 생성 후, 예측 결과와 비교하기 위해 사용되는 실제 정비 이력임** **(2) 각도 관련된 정비 이력만 정비 시점 비교 대상으로 선정함**

(1) 타겟 변수 : 예측을 해야 하는 대상(목표) 변수 (Y변수)
(2) 설명 변수 : 타켓 변수를 예측하기 위해 사용되는 변수 (X변수)
(3) 거리 변수를 제외한 이유는 거리차 기준(0.1 이상)으로 취출되는 데이터 비율이 적어 예측 모델 생성에 중요한 변수가 아님

P&D Solution

3. 분석 프로세스

절곡기 00호기 정비 시점 예측을 위한 분석 프로세스는 다음과 같음

기초 분석(데이터 현황 파악) → **예측 모델링** → **예측 결과 vs. 정비 이력 비교**

(1) 각도 데이터 현황 분석
- 각도 좌/우의 평균, 분산

(2) 각도 좌/우 상관분석
- 좌 12개 데이터 간 상관관계
- 우 12개 데이터 간 상관관계
- 좌 vs. 우 데이터 간 상관관계

(3) 각도 vs. 모터 위치 상관분석

2가지 모델링 방안을 적용함
(1) 각도로만 취출 여부 예측
(2) 각도 및 모터 위치로 취출 여부 예측

2가지 모델링 방안 중 정확도가 높은 모델을 선정하여 미래 시점(7월, 11월)의 정비 이력과 비교함

P&D Solution

좌/우 각도, 각도 vs. 각도, 각도 vs. 모터 위치 등에 대한 기초 분석 결과는 다음과 같음

분석 항목	분석 결과
각도(좌)	평균 최대 : 12번 각도, 평균 최소 : 7번 각도, 분산 최대 : 11번 각도
각도(우)	평균 최대 : 1번 각도, 평균 최소 : 3번 각도, 분산 최대 : 12번 각도
각도 간 상관관계	(1) 좌 12개 각도 간 상관관계 높지 않음 (2) 우 12개 각도 간 상관관계 높음 (3) 좌 vs. 우 간 상관관계 높지 않음
각도 vs. 모터 위치 간 상관관계	모터 위치 값이 클수록 각도(우) 값이 작아짐

각도(좌), 각도(우)의 최대 평균, 최소 평균, 최대 분산을 가지고 있는 각도가 취출 여부 예측에 영향을 줄 가능성이 높으며,
특히 각도(좌)보다는 상관관계가 높은 각도(우)들이 영향을 많이 줄 것으로 예상됨

2가지 모델링 방안을 적용하여 1월 ~ 6월 데이터로 정확도가 높은 모델을 최종 선정하고, 7월과 11월 데이터를 최종 모델에 대입하여 정비 시점을 예측함

모델링 방법

[모델 생성]

1월 데이터

2월 데이터

…

6월 데이터

1월 ~ 6월 데이터에 2가지 모델링 방안을 적용함

[모델 검증(정비 이력 비교)]

7월 데이터

11월 데이터

최종 모델을 7월, 11월 데이터에 대입하여 정비 시점을 예측함

[모델링 방안]

(1) 취출 여부(0, 1) = 각도(좌1) + … + 각도(우12)

(2) 취출 여부(0, 1) = 각도(좌1) + … + 각도(우12) + 모터 위치 정보

2가지 모델링 방안에 각각 적용되는 예측 모델 기법은 여러 개이며, 이 중 정확도가 높은 기법을 2가지 모델링 방안의 최종 모델로 선정함

정비 시점 예측 기준

정비 시점은 누출률, False Alarm Rate를 고려하여 품질 비용이 최소가 되는 기준을 수립해야 함

날짜	시간	취출 여부 예측	누적 취출 비율	정비 시점
12/01	00:00:00	0	0%	
	…	…	…	
	…	1	35%	
	…	…	…	
	…	**1**	**80%**	**Alarm**
	…	…	…	
	23:59:59	1	100%	

예를 들어, 일별 기준으로 취출로 예측되는 결과를 누적하여 특정 비율 초과 시 정비 시점에 대한 Alarm을 발송함

(2) 예측 모델 결과

2가지 모델링 방안 결과 최고 정확도는 거의 유사하지만, 취출 여부에 영향을 미치는 주요 인자의 경우, 기초 분석에서 예상한 3번과 12번 우측 각도가 도출된 모델링 방안1의 LightGBM을 최종 모델로 선정함

모델링 방안1(각도로 각도차 이상의 취출 여부 예측)

예측 모델 기법	정확도	주요인자1	주요인자2
Random Forest	0.890	3번 우측 각도	12번 우측 각도
Logistic Regression	0.623	3번 우측 각도	3번 좌측 각도
LightGBM	**0.985**	3번 우측 각도	12번 우측 각도
XGBoost	0.975	3번 우측 각도	12번 우측 각도
Decision Tree	0.667	3번 우측 각도	7번 우측 각도

5개 예측 모델 기법을 적용하여 모델을 생성하였고, 최고 98.5%의 정확도를 보이며, 취출 여부에 영향을 미치는 주요 인자는 3번 우측 각도, 12번 우측 각도 순서임 (7월 데이터 기준으로 실제 취출 여부와 예측 취출 여부 비교 결과 6.4% 차이 발생함)

모델링 방안2(각도/모터 위치로 각도차 이상의 취출 여부 예측)

예측 모델 기법	정확도	주요인자1	주요인자2
Random Forest	0.896	3번 우측 각도	6번 우측 각도
Logistic Regression	0.630	3번 우측 각도	12번 좌측 각도
LightGBM	**0.984**	6번 우측 각도	3번 우측 각도
XGBoost	0.974	6번 우측 각도	3번 우측 각도
Decision Tree	0.678	6번 우측 각도	3번 우측 각도

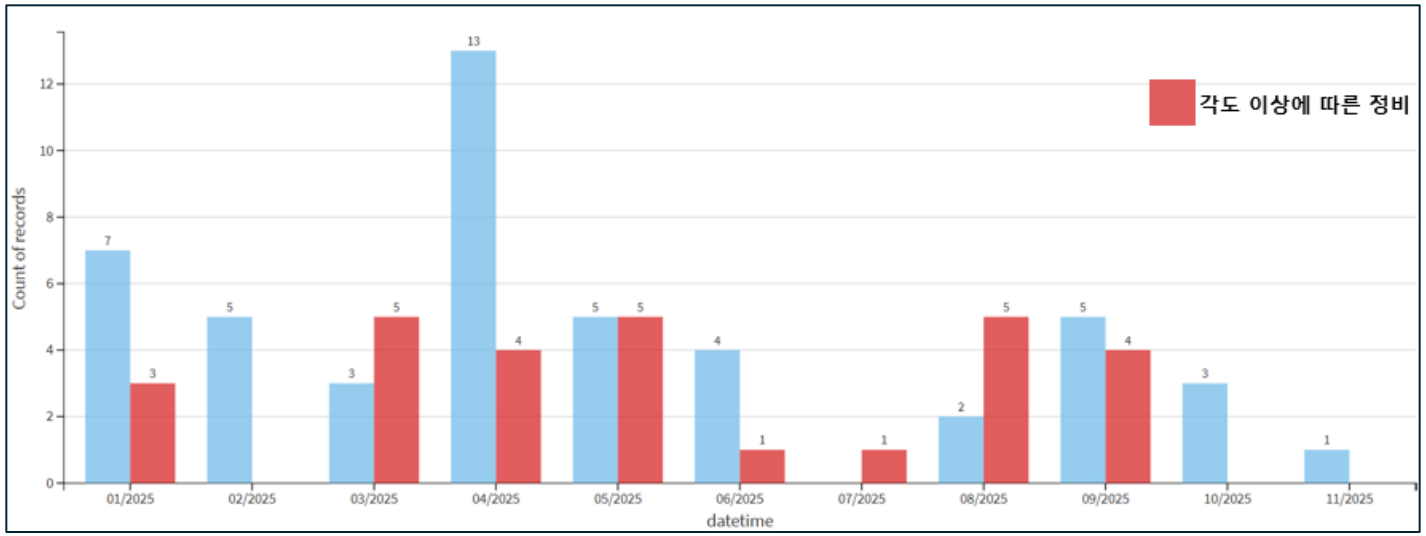
5개 예측 모델 기법을 적용하여 모델을 생성하였고, 최고 98.4%의 정확도를 보이며, 취출 여부에 영향을 미치는 주요 인자는 6번 우측 각도, 3번 우측 각도 순서임

P&D Solution

6. 예측 결과 vs. 정비 이력 비교

각도 관련 정비 이력의 경우, 7월에는 7월 17일 1건, 11월에는 정비 이력이 없었는데, 예측 결과와 동일함

[월별 실제 정비 이력 분포]



[누적 취출 비율별 정비 시점 예측 일자(7월 데이터)]

일별 누적 취출 비율	정비 시점 예측 일자 (모델링 방안1 적용)	실제 정비 일자
80% 이상	**7/17**	**7/17**
70% 이상	**7/17**, 7/18	
50% 이상	7/2, 7/4, **7/17** ~ 7/20, 7/24	

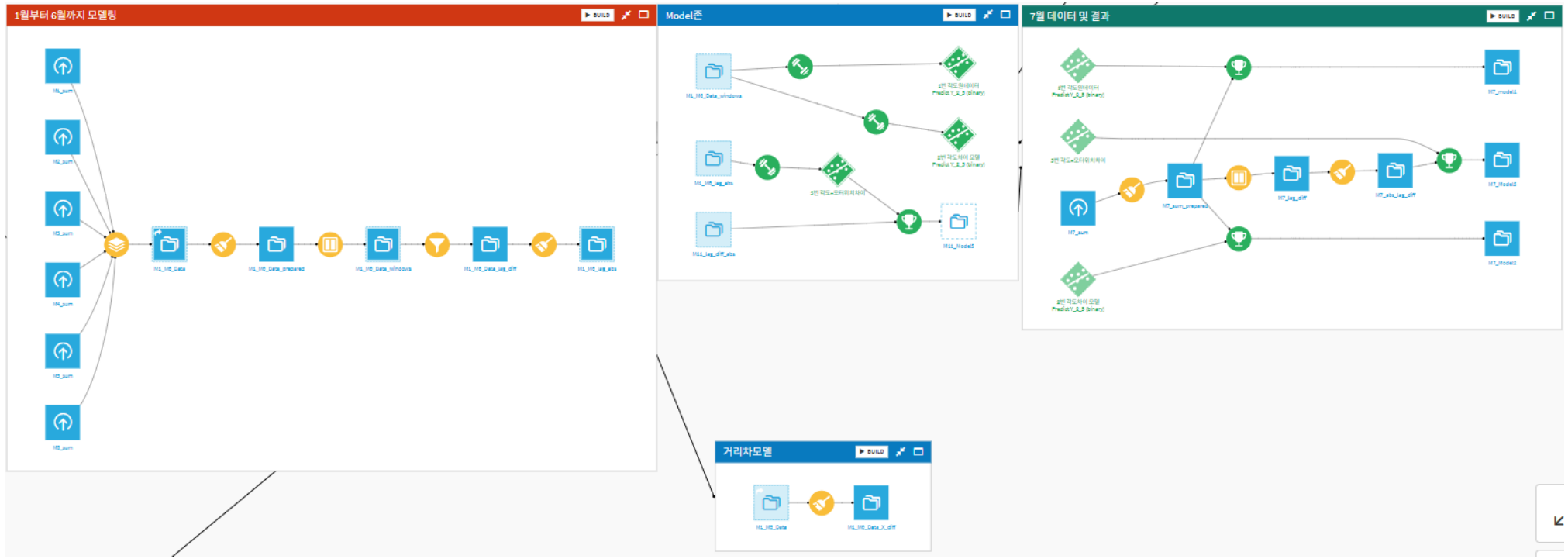
[누적 취출 비율별 정비 시점 예측 일자 11월 데이터)]

일별 누적 취출 비율	정비 시점 예측 일자 (모델링 방안1 적용)	실제 정비 일자
30% 이상	**없음**	**정비 없음**
20% 이상	**없음**	
10% 이상	11/5, 11/7, 11/17	

7. Dataiku 화면

Dataiku는 간단한 클릭만으로 데이터 전처리 및 모델링이 가능하며, 해당 과정을 Flow 형태로 생성하여 공유 및 재활용할 수 있음

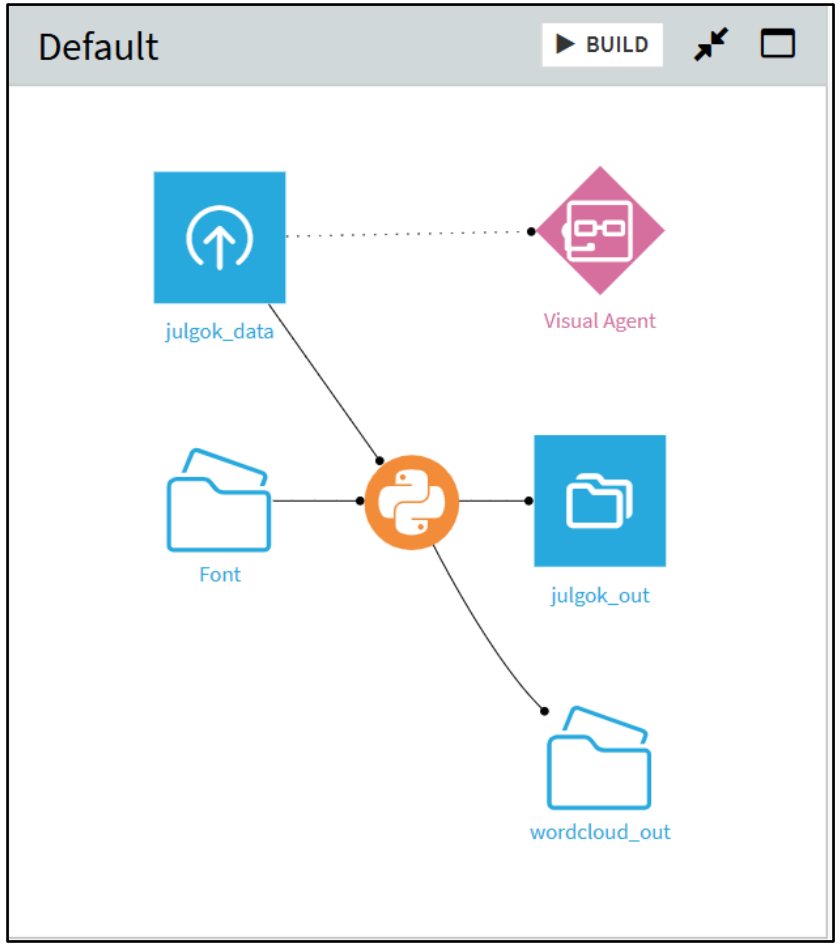
[Dataiku PoC Flow 화면]



파란색 아이콘 : 데이터, 노란색 아이콘 : 데이터 전처리, 초록색 아이콘 : 모델링, 주황색 아이콘 : 코드 작업

8. 정비 이력 정보를 활용한 챗봇

실제 정비 이력 정보로 간단한 챗봇 기능을 구현하였고, 정비 이력을 포함하여 인사 규정, 설비 도면 정보 등을 활용한 AI 기능의 범위를 모델링 뿐만 아니라 챗봇까지 확대, 적용할 수 있음



9. 제언

절곡기 이상 감지는 기존 각도차(또는 거리차) 기준으로 하되, 정비 시점 예측은 각도 데이터만으로 가능하기 때문에 현재로서는 추가 데이터 확보 등은 불필요하며, AI 솔루션을 통한 정비 시점 예측 및 정비 이력 챗봇 구현 등 다양한 기능을 활용하여 공정 운영, 품질 관리, 업무 효율 개선에 기여할 수 있을 것임

1. 데이터 수집 환경 개선

정비 시점 예측 모델 뿐만 아니라 다양한 모델을 개발하고, 시스템에 의한 모델 업데이트 등의 자동화된 운영 관리를 위해서는 기존 파일 형태의 수집에서 시스템에 의한 자동 연계가 필요함

2. 정비 시점 예측 기준 수립

이상 제품의 누출, 정상 제품의 불필요한 Alarm 등 품질 비용을 최소화할 수 있도록 설비 특성, 월별 불량 발생 패턴 등을 고려하여 정비 시점 예측 기준을 수립해야 함

3. AI 추진에 대한 전사 이해 및 공감대 형성

AI 솔루션 도입을 통해 비즈니스 성과를 달성하기 위해서는 AI에 대한 정확한 이해와 도입 목적을 명확히 하고, 전사적인 공감대 형성이 필요하며, 이를 지원할 수 있는 솔루션이 필수적임

P&D Solution

인사규정자료 AI Agents 구축

Agent 1

인사 규정자료

AI Agent **구축**

+

Agent 2

- 직원 정보를 활용한 **이직 예측** Agent 구축

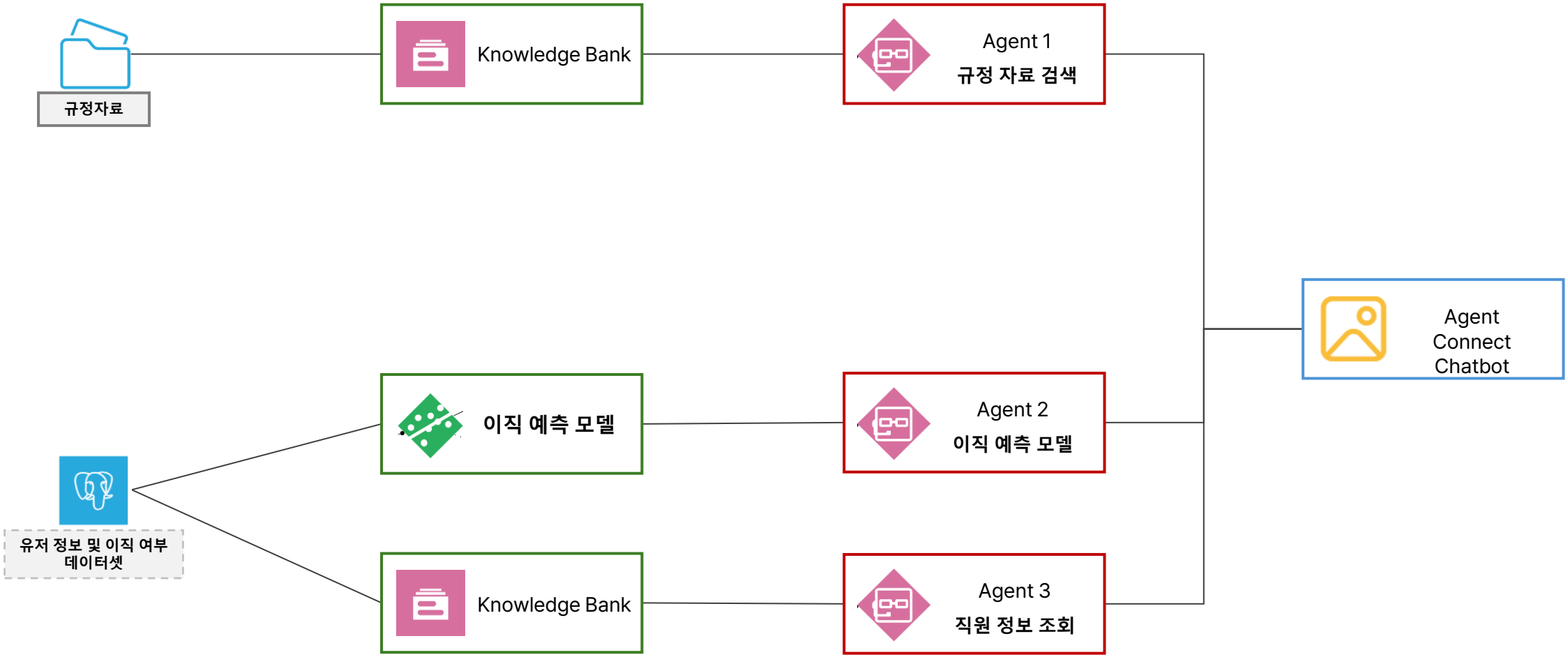
Agent 3

- **직원 정보 조회** Agent 구축

추가 기능

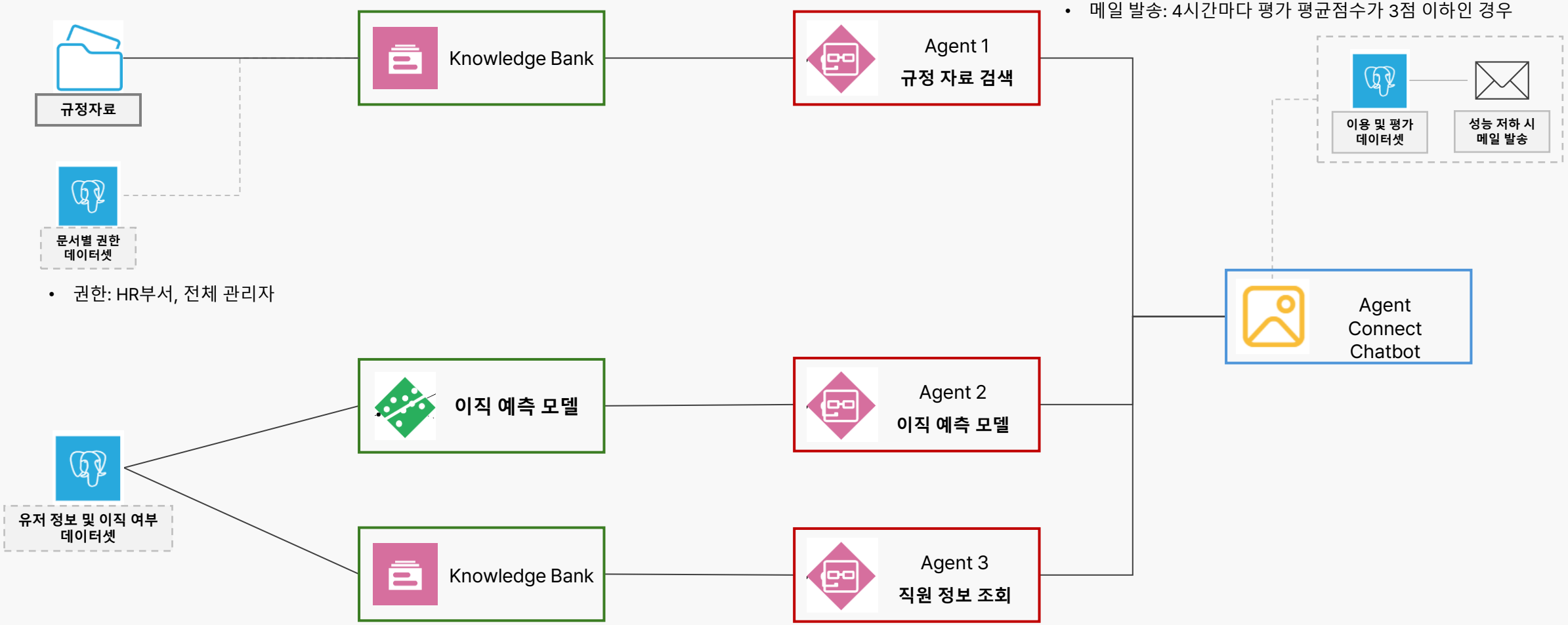
- LLM Chatbot 이용 기록 DB 구축
- Chatbot 평가에 따른 메일 발송
- 권한 별 답변 내용 제한

인사규정자료 AI Agents 구축



인사규정자료 AI Agents 구축

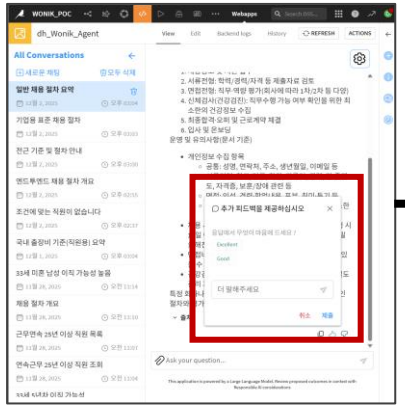
- 평가: Excellent(5점), good(4점), Not Bad(3점), Bad(2점)
- 메일 발송: 4시간마다 평가 평균점수가 3점 이하인 경우



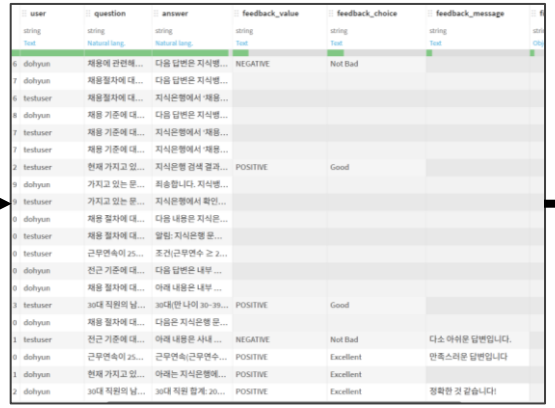
- 권한: HR부서, 전체 관리자

인사규정자료 AI Agents 구축

사용자 피드백



챗봇 이용 내역 저장



※ 사용자, 질문 시간, 답변, 피드백 등

피드백 점수 평균 계산

	feedback_score	feedback_value	feedback_choice
	double Decimal	string Text	string Text
(직원용)" ...	5.0	POSITIVE	Excellent
해본 결과...	4.0	POSITIVE	Good
못해, 일...	3.0	NEGATIVE	Not Bad
파일명 기...	4.0	POSITIVE	Good
이며, 성...	4.0	POSITIVE	Good
에 따른 ...	3.0	NEGATIVE	Not Bad
다음과 같...	5.0	POSITIVE	Excellent
제목(파...	5.0	POSITIVE	Excellent
) ¶- 여자:...	5.0	POSITIVE	Excellent

Excellent(5점), good(4점), Not Bad(3점), Bad(2점)

※ 피드백 종류 및 점수는 변경 가능

평균 3점 이하

평균 3점 초과

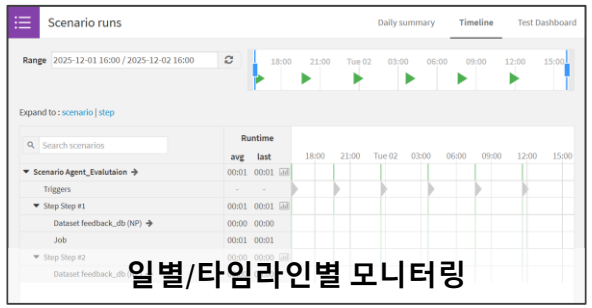
이메일 발송

PASS

스케줄링 기능

※ 4시간 간격 **자동** 수행

No Code 스케줄링 설정



일별/타임라인별 모니터링

P&D Solution

인사규정자료 AI Agents 구축

관리자

D 채용 절차에 대해서 알려줘

아래 내용은 지식은행 문서(채용 프로세스 및 운영 기준, 개인정보 처리 기준(내·외부), 면접비 지급 기준)를 바탕으로 정리했습니다.
표준 채용 절차(요약)

- 공고 및 지원 접수: 채용 공고 게재, 지원서/이력서 등 접수
- 서류전형: 자격·경력·학력 등 기본 요건 검토
- 면접전형: 직무·역량·조직적합도 평가(필요 시 과제/테스트 병행)
- 최종선정 및 처우협의: 합격 결정, 처우·입사일 협의
- 사전검증: 직무수행 가능 여부 판단에 필요한 최소한의 건강정보 등 확인
- 입사 및 온보딩: 계약/입사서류, 계정·장비·오리엔테이션 등

개인정보 및 서류 처리(문서 근거 요약)

- 채용 단계에서 수집 가능한 항목: 성명, 연락처, 주소, 생년월일, 사진, 이메일 등 공통 정보, 학력/전공/성적/경력/자격, 연구실적, 보훈·장애 관련 사항, 면접 관련 응답, 직무수행 가능 여부 판단에 필요한 최소한의 건강정보 등
- 미채용자의 서류 반환: 채용여부 확정 후 미채용 구직자가 요청하면 14일 이내 반환(다만 홈페이지·이메일로 제출되어 반환이 어려운 서류는 제외). 불필요해진 정보는 지체 없이 파기

D 전근 종류에 대해서 알려줘

아래 내용은 지식은행 문서(시행기준-전근기준)를 바탕으로 정리했습니다.
전근의 종류

- 파견
  - 정의: 타 지역(사업장)에 단신으로 일정 기간 근무 후 원소속으로 복귀
  - 기간: 3~12개월(3개월 미만은 출장 처리)
- 이동
  - 정의: 인사발령에 따라 타 지역(사업장)으로 완전 이전하여 계속 근무(근무지 변경)

운영·유의사항(요약)

- 전근은 입사 후 재직 중 인사발령 사안이며, 발령 전 대상자와 업무상 필요성·생활상 불이익 등을 신의칙에 따라 사전 협의
- 신규 입사자는 전근 해당 없음
- 출퇴근 가능 권역(본가/자택 인근 등)은 전근 지원 기준 미적용
- 임원은 숙소 외 정착지원금 미적용
- 서울권 이동 발령자는 숙소/기숙사 미지원(판교 포함)

필요하시면 파견/이동 각각의 절차(발령-온보딩-복귀), 비용/수당 기준, 서식(전근 동의서, 안내문)까지 맞춤으로 정리해 드리겠습니다. 어느 케이스(파견/이동)와 대상 직군을 검토 중이신가요?

HR 부서

T 채용 절차에 대해서 알려줘

지식은행에서 '채용 절차' 관련 문서를 찾지 못해, 아래 내용은 일반 HR 실무 기준에 따른 안내입니다.
표준 채용 절차(End-to-End)

- 인력계획/승인: 헤드카운트·예산 승인, 직무기술서(JD), 급여밴드 확정.
- 공고/소싱: 공고문(직무, 고용형태, 근무지, 급여범위, 전형절차, 제출서류, 마감일) 작성·게시, 채널 믹스(잡보드, 링크드인, 리퍼럴, 학회/커뮤니티).
- 서류전형: 필수·우대 요건 기준표, 블라인드 항목 관리, ATS 세팅, 개인정보 수집·보관 동의.
- 평가/과제: 직무과제, 코딩/케이스/역량검사 등 필요시 실시(직무연관성·공정성 검토).
- 면접: 구조화 면접 질문지·평가루브릭, 패널 구성, 편향 방지·동일 질문 원칙, 합격기준 사전합의.
- 최종검증: 학력·자격·경력 확인, 평판조회(사전 동의 필수), 필요한 경우 백그라운드 체크 범위 제한.
- 오퍼/협상: 처우패키지(연봉, 직급, 인센티브, 스톡옵션, 입사일), 오퍼레터·유효기간 안내.
- 최종승인·계약: 근로계약서 필수기재사항(임금, 소정근로시간, 업무·근무장소, 휴일·휴가, 수습조건 등), 전자서명.
- 프리보딩: 4대보험·신규 입사자 정보 수집, 장비/계정·좌석·교육 일정 준비, 환영 커뮤니케이션.
- 온보딩/수습: 30-60-90 계획, 멘토 지정, 정기 체크인, 수습평가 기준·서면 피드백.

대한민국 법규 유의사항(핵심)

T 전근 종류에 대해서 알려줘

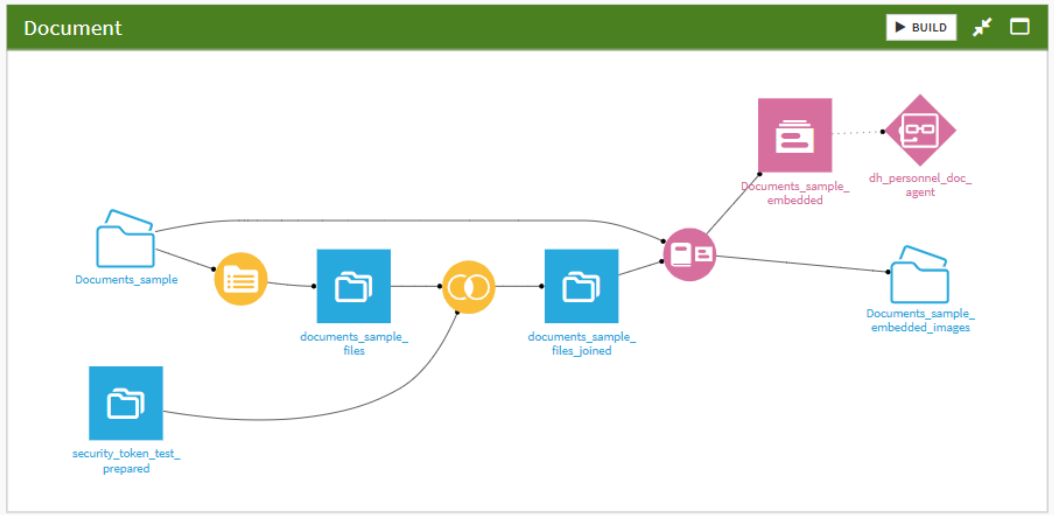
지식은행 문서(시행기준-전근기준) 기준 요약
전근 종류

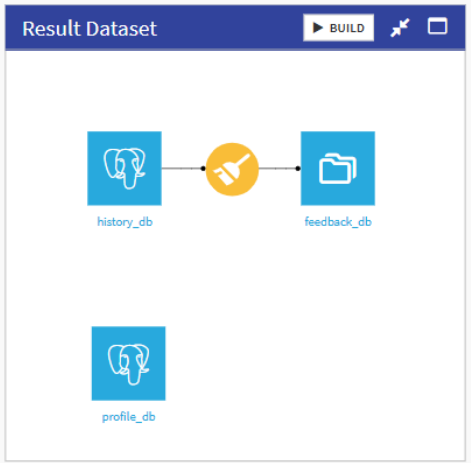
- 파견: 타 지역(사업장)에 일정 기간 단신 파견 후 복귀하는 형태
  - 기간: 3~12개월(3개월 미만은 출장 처리)
- 이동: 인사발령으로 발령지로 완전히 이동하여 계속 근무하는 형태
  - 기간: 상시(지속 근무)

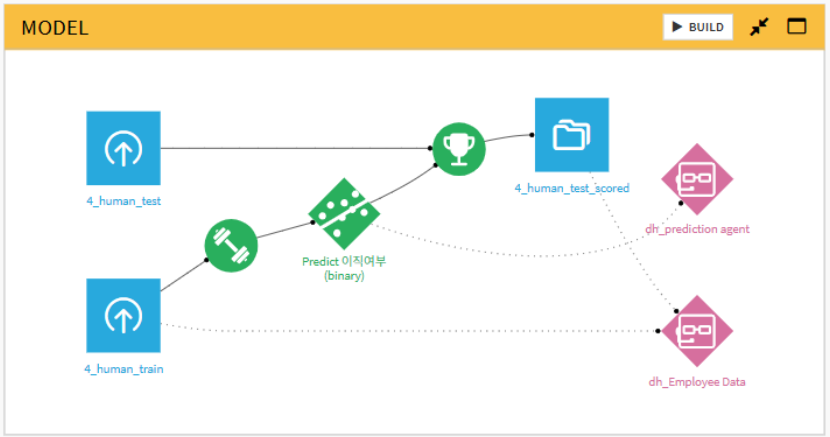
관련 지원 제도(요약)

- 파견: 숙소(원룸) 또는 숙소보조금 중 선택, 기혼자가 가족과 떨어져 근무 시 귀향지원금 월 30만원(최대 1년)
- 이동
  - 단신 부임: 숙소 지원(원룸/보조금, 최대 3년), 정착비(기혼 200만원·미혼 100만원, 1년 이내 퇴사 시 반납)
  - 동반 부임: 가족 동반 이사 시 중개료/이사비 각 150만원 한도 실비 정산, 6개월 임시숙소 또는

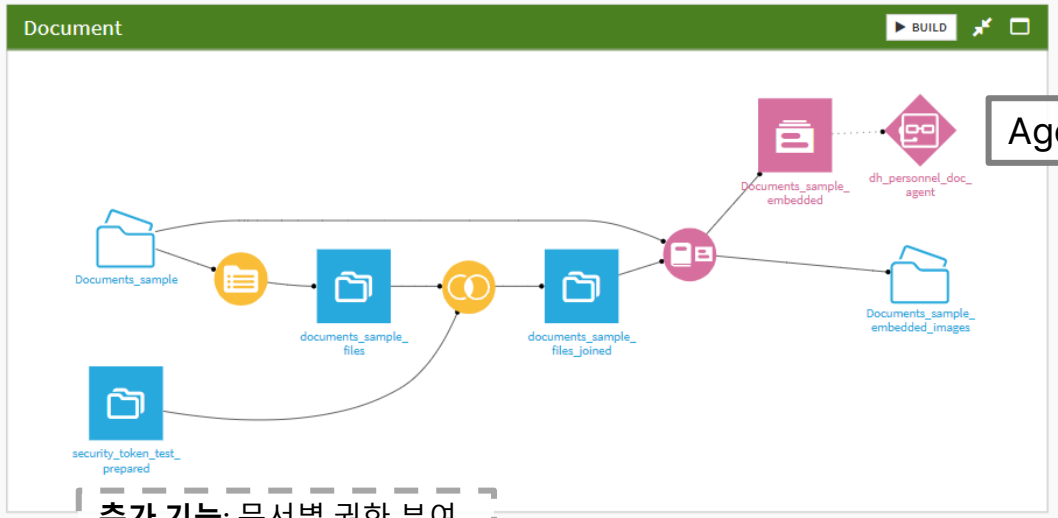
인사규정자료 AI Agents 구축





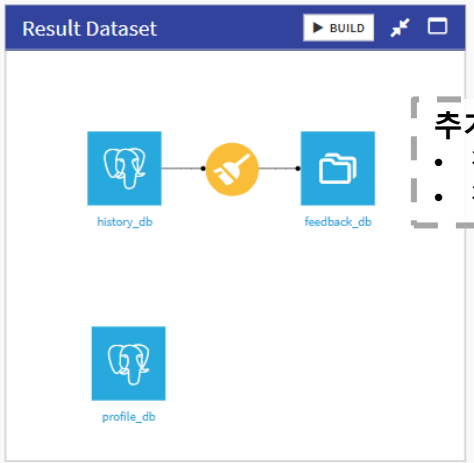


인사규정자료 AI Agents 구축

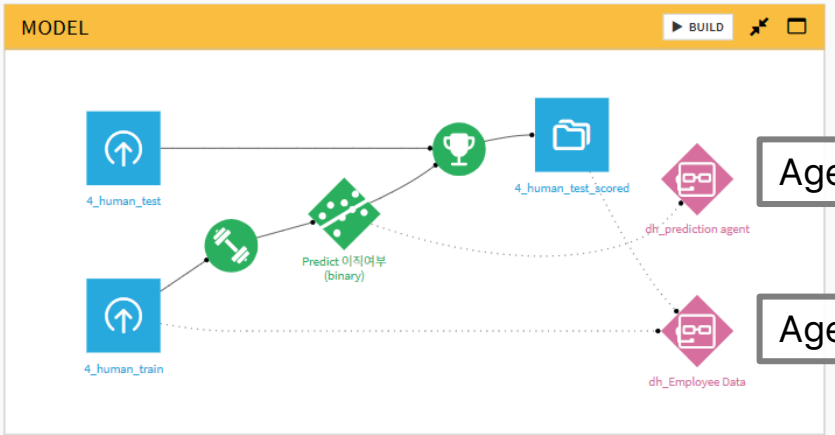


Agent 1 : 인사 규정자료 검색 에이전트

추가 기능: 문서별 권한 부여



추가 기능
- 챗봇 이용 내역 DB 구축
- 챗봇 평가 및 이메일 발송



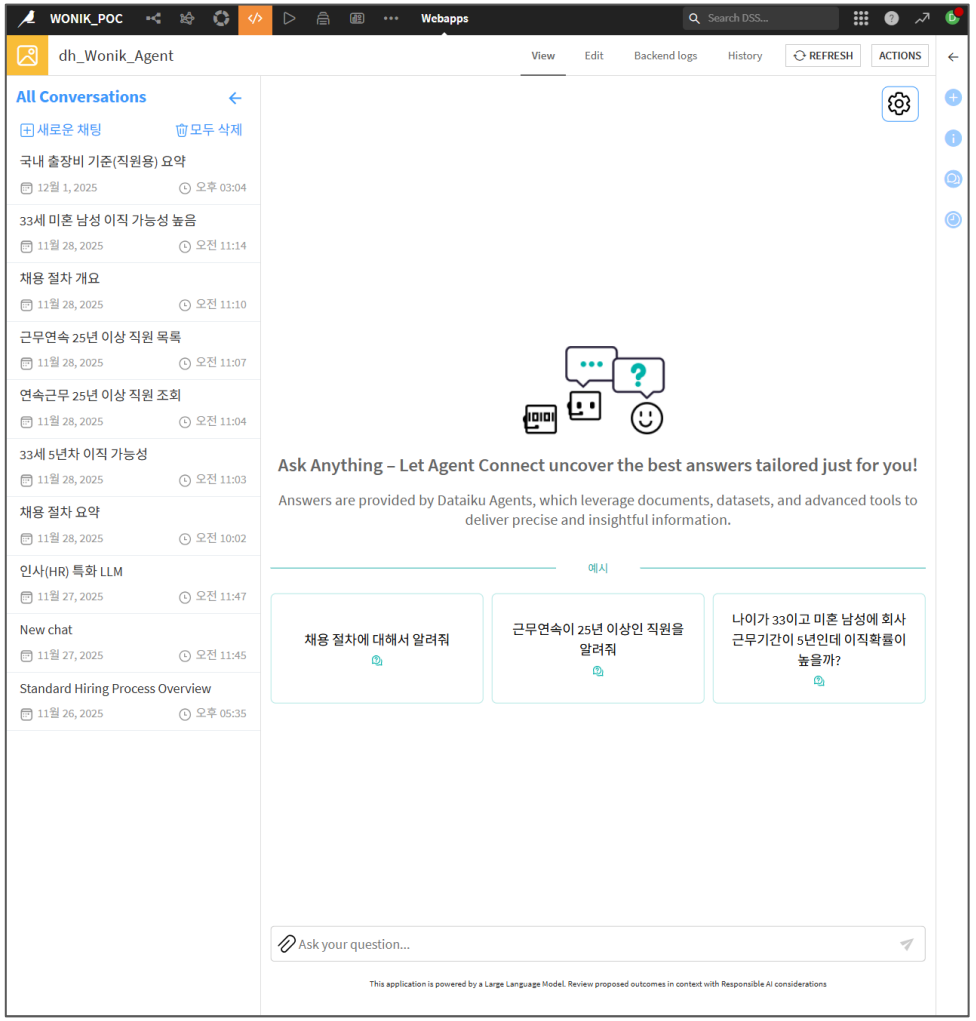
Agent 2 : 이직 예측 에이전트

Agent 3 : 직원 조회 에이전트

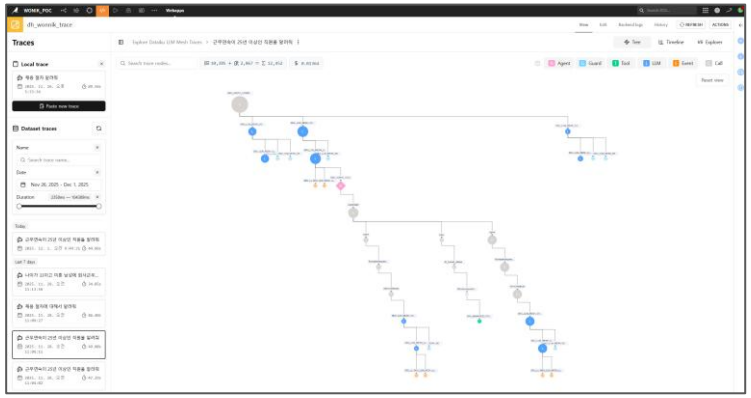
P&D Solution

인사규정자료 AI Agents 구축

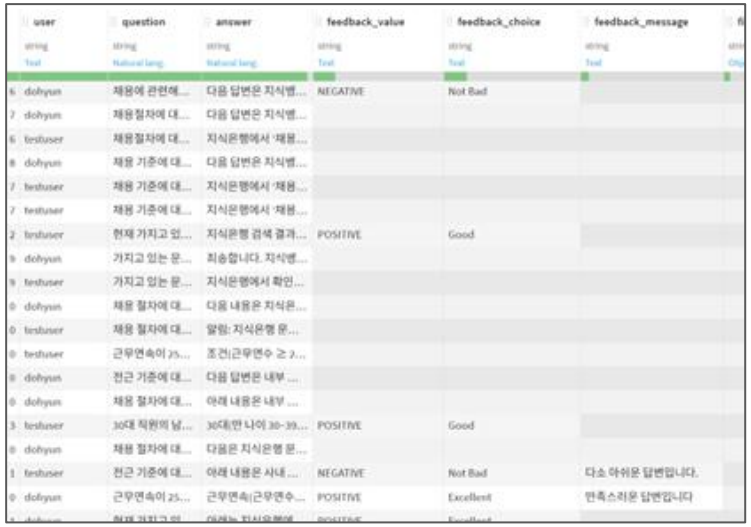
챗봇 UI



Trace 화면



이용 내역 데이터셋



user	question	answer	feedback_value	feedback_choice	feedback_message
dohyun	채용에 관련해...	다음 답변은 지식뱅...	NEGATIVE	Not Bad	
dohyun	채용절차에 대...	다음 답변은 지식뱅...			
testuser	채용절차에 대...	지식은행에서 '채용...			
dohyun	채용 기준에 대...	다음 답변은 지식뱅...			
testuser	채용 기준에 대...	지식은행에서 '채용...			
testuser	채용 기준에 대...	지식은행에서 '채용...			
testuser	현재 가지고 있...	지식은행 검색 결과...	POSITIVE	Good	
dohyun	가지고 있는 문...	죄송합니다. 지식뱅...			
testuser	가지고 있는 문...	지식은행에서 확인...			
dohyun	채용 절차에 대...	다음 내용은 지식은...			
testuser	채용 절차에 대...	알림: 지식은행 문...			
testuser	근무연속이 25...	조건(근무연수 ≥ 2...			
dohyun	전근 기준에 대...	다음 답변은 내부 ...			
dohyun	채용 절차에 대...	아래 내용은 내부 ...			
testuser	30대 직원의 남...	30대(만 나이 30-39...	POSITIVE	Good	
dohyun	채용 절차에 대...	다음은 지식은행 문...			
testuser	전근 기준에 대...	아래 내용은 사내 ...	NEGATIVE	Not Bad	다소 아쉬운 답변입니다.
dohyun	근무연속이 25...	근무연속(근무연수...	POSITIVE	Excellent	만족스러운 답변입니다

P&D Solution

이미지 데이터 분석 – Object Detection

01 제품 생산 공정 내 불량 판정 프로세스

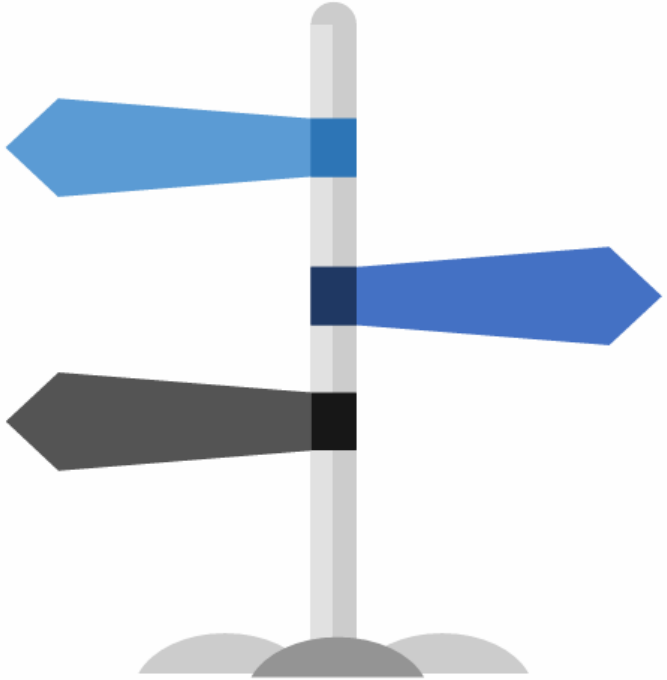
- 현 프로세스는 **1차 시스템, 2차 검수단계**로 구성, 두 과정을 통해 불량제품 식별
- 1차 시스템은 YOLO 모델 기반의 자동화된 불량 판정 수행
- 2차 검수는 인력을 활용한 추가 확인 과정

02 현행 시스템 문제점 및 개선사항

- **불량 판정 정확도 문제**
  - 현재 YOLO 모델의 불량 판정 정확도가 낮아 불량 미 판정 사례 발생
  - 불량 제품 발생 시, 업체 과실로 인한 손실 초래

  → **AI 모델의 성능 개선** 및 품질 안정성 확보
- **모델 운영 및 관리 효율성 부족**
  - 이미지 추가 및 품목 변경 시마다 코드 수정 필요
  - 재학습 및 업데이트에 불편성을 느낌

  → 데이터 변경 및 모델 업데이트가 용이한 **플랫폼 기반 모델 관리로 운영 효율성 향상**

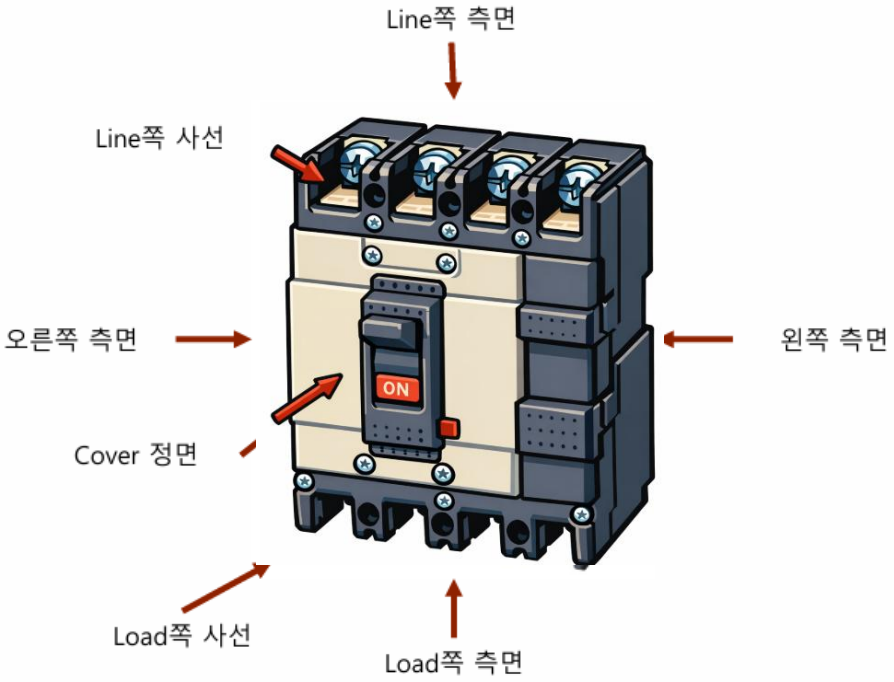


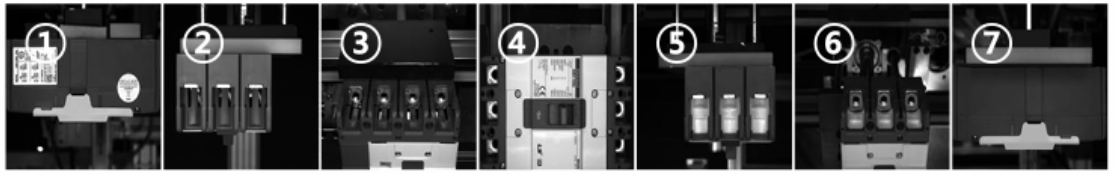
이미지 데이터 분석 – Object Detection

제품 이미지 및 검사 항목

- 공정 과정에서 촬영된 이미지를 활용한 제품의 불량 검출 파악
- 총 7 방향에서 촬영된 이미지 데이터가 존재하며, 각각의 검사 항목이 존재

위치	검사 항목
① 오른쪽 측면	• 명판 형식명, 정격 검사 • 실드 라벨 유무 혼입, 훼손 • Case, Main Cover 파손 및 미성형
② Load쪽 측면	• Case, Main Cover 파손 및 미성형 • Load Barrier 파손 및 미성형
③ Load쪽 사선	• Tap 유무
④ Cover 정면	• Main Cover 정격 검사 • Aux Cover 형식명 검사 • Aux Cover, Main Cover 파손 및 미성형
⑤ Line쪽 사선	• Tap 유무
⑥ Line쪽 측면	• Case, Main Cover 파손 및 미성형 • Arc Barrier 유무 • Base 파손 및 미성형
⑦ 왼쪽 측면	• Case, Main Cover 파손 및 미성형 • Aux Cover 파손 및 미성형 • Barrier 노출 검사(2pole)





이미지 데이터 분석 – Object Detection

제품 이미지 별 데이터 구성

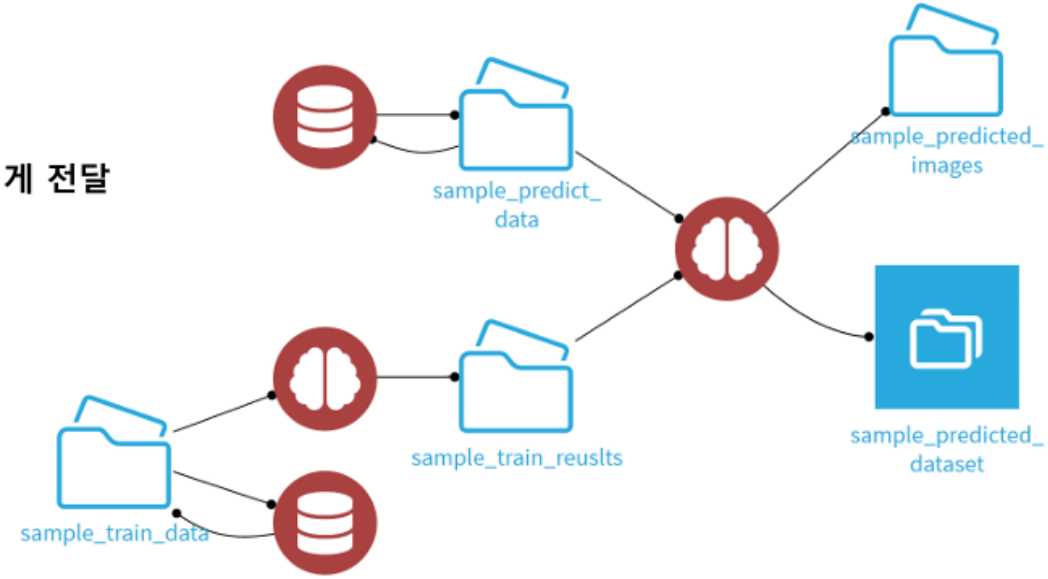
위치	양품	아크 베리어	옥스 커버	베리어	케이스	탭	실드 라벨	전체
LEFT	10	-	2	-	9	-	-	**21**
LINE	15	113	-	-	22	-	-	**150**
LINE_TAP	15	111	2	-	1	23	-	**149**
LOAD	15	-	-	138	11	-	-	**164**
LOAD_TAP	40	-	-	102	6	3	-	**153**
RIGHT	20	-	16	-	28	-	69	**133**
TOP	25	-	1	-	1	-	-	**27**

- **LEFT와 TOP**의 경우 **매우 적은 데이터** 수를 보유
- **LINE, LINE_TAP, LOAD, LOAD_TAP**의 경우 충분한 데이터가 있으나, 결함 유형 간 **데이터 비율이 극단적으로 불균형**
- **RIGHT**의 경우 **가장 균형적인 데이터** 분포를 보임

이미지 데이터 분석 – Object Detection

전체 프로세스

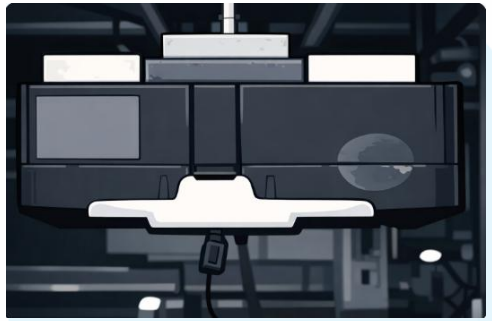
- **모델 성능 향상**을 위한 **이미지 전처리** 수행
- 결함 예측을 위한 딥러닝 기법 활용
  - Image Detection 기법인 **YOLO**
  - 이미지에서 텍스트를 읽는 **Easyocr**
- 예측 결과를 **이미지**와 **데이터프레임**으로 시각화하여 **쉽게 전달**
- **Plugins**을 이용한 **모듈 간편화**



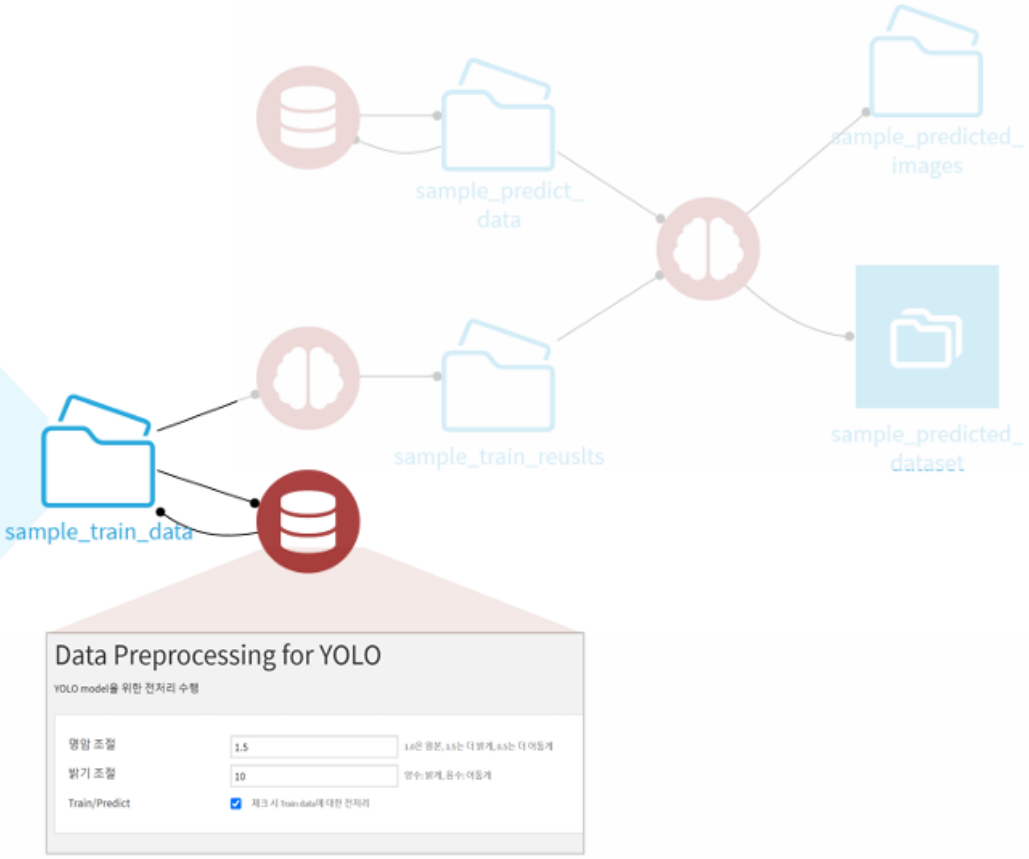
이미지 데이터 분석 – Object Detection

데이터 전처리

- Input: **이미지 데이터**와 **라벨링 데이터** (=결함 위치 정보)
- **전처리** 코드
  - **밝기** 및 **대비** 조절
  - **Yolo에 필요한 데이터 생성**



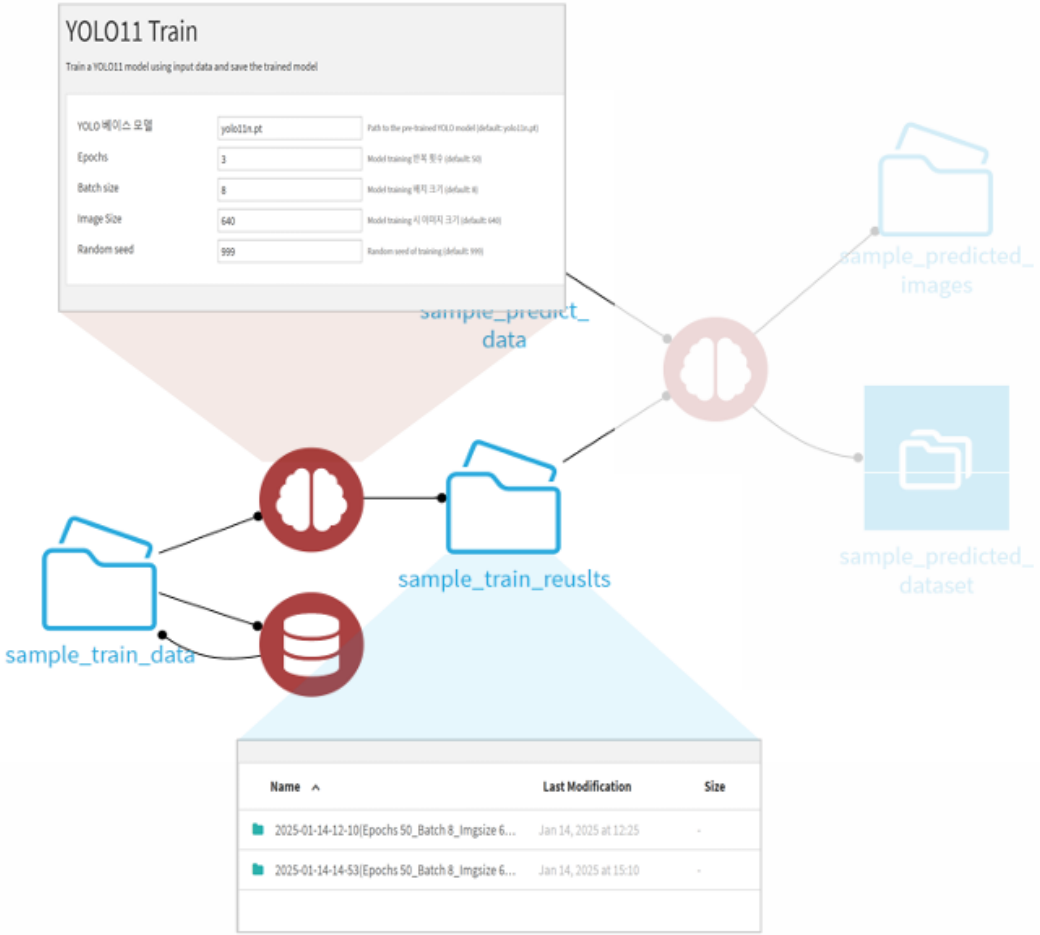
0 0.779003 0.528809 0.185458 0.205078



이미지 데이터 분석 – Object Detection

모델 구축

- **Yolo v11 nano 기반 사전모델** 학습 사용
- 데이터를 이용한 **Fine-tunning** 진행

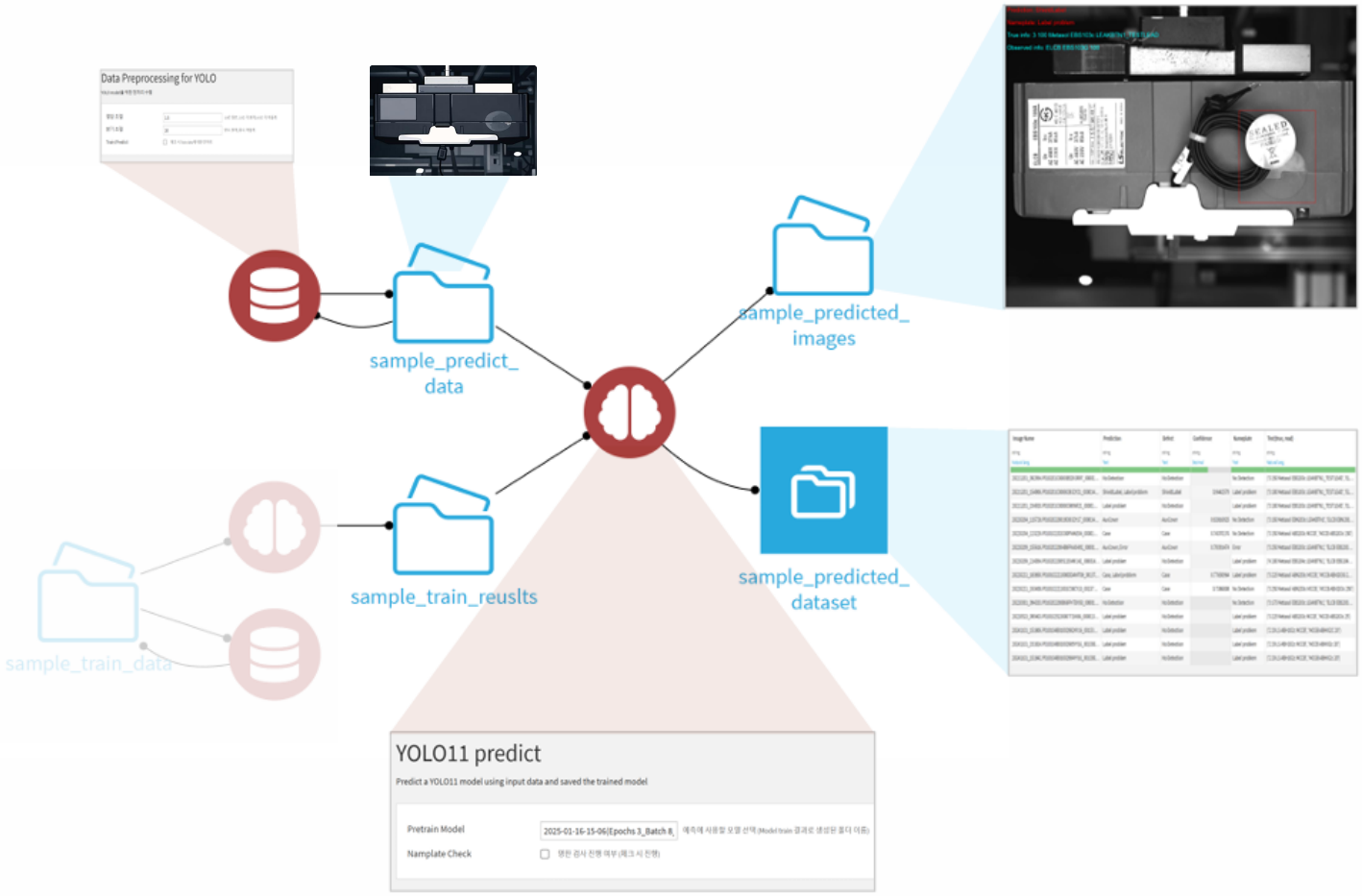


P&D Solution

이미지 데이터 분석 – Object Detection

예측

- 예측 이미지에 대한 전처리 수행 (밝기, 대비 조절)
- Fine-Tunning된 모델을 통한 예측 수행
  - 필요 시 명판 분석도 수행
- 예측 결과 출력
  - **이미지: 이미지 위에 예측 결과 및 위치 시각화**
  - **CSV: 예측 결과를 데이터 프레임 형태 출력**



이미지 데이터 분석 – Object Detection

제품 이미지 별 데이터 및 성능

위치	전체	Precision	Recall	mAP50	mAP50-95	소요시간(CPU)
LEFT	21	0.017	1.000	0.663	0.282	2분 23초
LINE	150	0.975	0.662	0.913	0.554	19분 40초
LINE_TAP	149	0.991	0.992	**0.995**	**0.601**	18분 14초
LOAD	164	0.957	0.500	0.698	0.326	20분 08초
LOAD_TAP	153	0.846	0.459	0.496	0.266	18분 18초
RIGHT	133	0.748	0.771	**0.823**	**0.409**	16분 37초
TOP	27	0.000	0.000	0.000	0.000	2분 58초

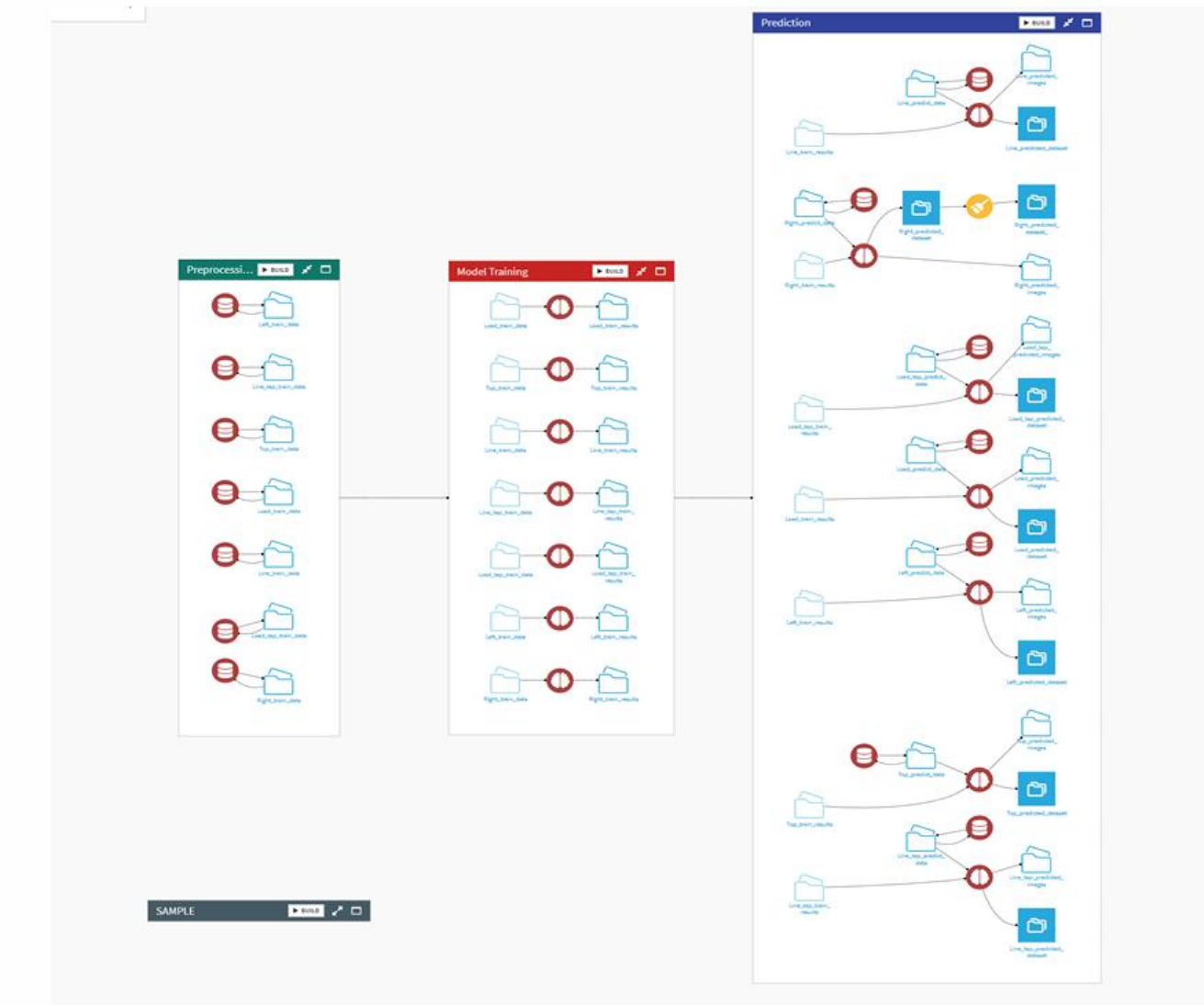
모델 결과

- **데이터의 수가 충분**하고 결함 간 데이터 **분포가 균형**적인 경우 **모델의 성능이 우수**하게 나타남 → **RIGHT, LINE_TAP**
- **데이터 수가 적은** 경우 **모델 성능이 낮게** 나타남을 확인 → LEFT, TOP
- 데이터 수가 충분하더라도 결함 유형 간 **데이터 비율이 극단적으로 불균형** 할 경우 모델의 **성능이 다소 낮게 나타남**을 확인 → LINE, LINE_TAP, LOAD, LOAD_TAP

향후 개선 방향

- **데이터를 확충**하고, 특히 데이터 수가 **적은 결함 유형에 대한 보완**이 필요
- 동시에 결함 부위 **라벨링 작업의 정확도**를 높인다면, 더 높은 성능의 모델 구축이 가능할 것으로 기대됨

이미지 데이터 분석



감사합니다 :)

Dataiku Certification 소개

Core Designer

Your first steps with Dataiku

5 Courses

Advanced Designer

Become an expert in Dataiku's visual tools for building data pipelines

7 Courses

ML Practitioner

Hone your skills on the visual tools in Dataiku for building machine learning models

10 Courses

Generative AI Practitioner

Learn how to leverage the LLM visual recipes in Dataiku

4 Courses

Developer

Discover Dataiku's code integrations and expand your horizons beyond the visual tools

7 Courses

MLOps Practitioner

Learn how to deploy and monitor projects and API services in production environments

4 Courses

AI Governance Specialist

Learn how the Dataiku Govern node supports your governance mission.

5 Courses

Dataiku Certification 소개

설문조사

- 설문 부탁드립니다

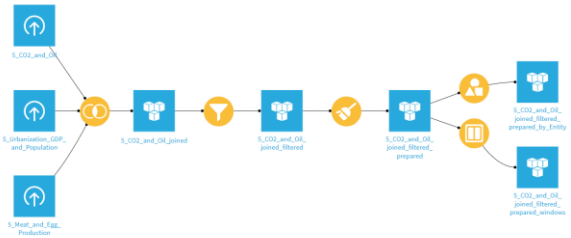
P&D Solution

Certification(Core Designer)

https://github.com/RaOOns/dataiku_handson/

1. 프로젝트 생성 및 데이터 불러오기

1. Dataiku 홈 화면에서 **+ New Project** 클릭
2. **Blank project** 선택
3. **Dataset Import** (Tip. Import 시 Schema 체크하기)
   - `CO2_and_Oil`
   - `Meat_and_Egg_Production`
   - `Urbanization_GDP_and_Population`



2. 데이터 결합 (Join Recipe)

- **세 개의 Dataset을 하나의 Dataset으로 Join**
  - 기준파일: `CO2_and_Oil`
  - Key값: `Entity` , `Code` , `Year`

3. 데이터 필터링 (Sample/Filter Recipe)

- `Year` **컬럼이 2008년부터 2012년까지의 데이터만 포함되도록 필터링**

4. 데이터 준비 (Prepare Recipe)

1. **파생변수 생성**

   아래 세 개의 컬럼(국가 총계)을 **1인당(per-capita)** 기준으로 변환

   - **방법**: 각 컬럼을 **Formula**기능을 활용해 `해당 컬럼 값 / Population` 으로 계산
   - Tip. 컬럼명은 자유롭게 설정
   - **대상 컬럼**
     - `Oil production (Etemad & Luciana) (terawatt-hours)`
     - `meat_prod_tonnes`
     - `Food Balance Sheets: Eggs - Production (FAO (2017)) (tonnes)`
2. **컬럼 삭제(4개)**
   - `Population`
   - `Oil production (Etemad & Luciana) (terawatt-hours)`
   - `meat_prod_tonnes`
   - `Food Balance Sheets: Eggs - Production (FAO (2017)) (tonnes)`
3. **행 삭제(filtering)**
   - `Entity` 컬럼 값이 **world**인 **행 제거**
4. **결측치 대체**
   - 4-1에서 계산한**per-capita 컬럼(3개)**에 대한 **결측치 대체 (대체값 = 0)**

5. 데이터 그룹별 집계 (Group Recipe)

1. **[4. 데이터 준비] 단계에서 생성된 Dataset 사용**
2. `Group By` 값 `Entity`
3. `Code` , `Year` 를 **제외한 나머지 컬럼들**에 대해서 `Avg` 값 집계

6. 데이터 랭크 (Window Recipe)

1. 다시 **[4. 데이터 준비] 단계에서 생성된 Dataset 사용**
2. (좌측 옵션) **Windows definions** 설정
   - `PARTITONING COLUMNS` 값 `YEAR`
   - `ORDER COLUMNS` 값 `Per capita $CO_2$ emissions (tonnes per capita)` 컬럼
3. (좌측 옵션) **Aggregations** 설정
   - `Compute` 값 `Rank` 만 설정
4. (좌측 옵션) **Post-filter** 설정
   - `rank <= 25`

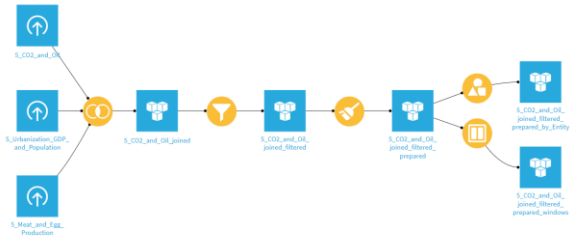
P&D Solution

Certification(Core Designer)

https://github.com/RaOOns/dataiku_handson/

1. 프로젝트 생성 및 데이터 불러오기

1. Dataiku 홈 화면에서 **+ New Project** 클릭
2. **Blank project** 선택
3. **Dataset Import** (Tip. Import 시 Schema 체크하기)
   - `CO2_and_Oil`
   - `Meat_and_Egg_Production`
   - `Urbanization_GDP_and_Population`



2. 데이터 결합 (Join Recipe)

- **세 개의 Dataset을 하나의 Dataset으로 Join**
  - 기준파일: `CO2_and_Oil`
  - Key값: `Entity`, `Code`, `Year`

3. 데이터 필터링 (Sample/Filter Recipe)

- `Year` **컬럼이 2008년부터 2012년까지의 데이터만 포함되도록 필터링**

4. 데이터 준비 (Prepare Recipe)

1. **파생변수 생성**

   아래 세 개의 컬럼(국가 총계)을 **1인당(per-capita)** 기준으로 변환

   - **방법**: 각 컬럼을 **Formula**기능을 활용해 `해당 컬럼 값 / Population` 으로 계산
   - Tip. 컬럼명은 자유롭게 설정
   - **대상 컬럼**
     - `Oil production (Etemad & Luciana) (terawatt-hours)`
     - `meat_prod_tonnes`
     - `Food Balance Sheets: Eggs - Production (FAO (2017)) (tonnes)`
2. **컬럼 삭제(4개)**
   - `Population`
   - `Oil production (Etemad & Luciana) (terawatt-hours)`
   - `meat_prod_tonnes`
   - `Food Balance Sheets: Eggs - Production (FAO (2017)) (tonnes)`
3. **행 삭제(filtering)**
   - `Entity` 컬럼 값이 **world**인 **행 제거**
4. **결측치 대체**
   - 4-1에서 계산한**per-capita 컬럼(3개)**에 대한 **결측치 대체 (대체값 = 0)**

5. 데이터 그룹별 집계 (Group Recipe)

1. **[4. 데이터 준비] 단계에서 생성된 Dataset 사용**
2. `Group By` 값 `Entity`
3. `Code`, `Year` 를 **제외한 나머지 컬럼들**에 대해서 `Avg` 값 집계

6. 데이터 랭크 (Window Recipe)

1. 다시 **[4. 데이터 준비] 단계에서 생성된 Dataset 사용**
2. (좌측 옵션) **Windows definions** 설정
   - `PARTITONING COLUMNS` 값 `YEAR`
   - `ORDER COLUMNS` 값 `Per capita $CO_2$ emissions (tonnes per capita)` 컬럼
3. (좌측 옵션) **Aggregations** 설정
   - `Compute` 값 `Rank` 만 설정
4. (좌측 옵션) **Post-filter** 설정
   - `rank <= 25`

P&D Solution