

MLOps 춘추 전국 시대 정리

변성윤

발표자 소개

쏘카 데이터 그룹

- 타다데이터팀

- MLOps TF

2018년 1월 쯤부터 머신러닝 프러덕션 공부(이 당시엔 MLOps란 단어가 거의 없던 시절)

2019년 1월에 MLOps KR 페이스북 그룹 생성



오늘 발표 소개

크게 3가지 맥락으로 설명

- MLOps란?
- MLOps Component
- MLOps 공부하고 싶다면

발표의 특징

- 머신러닝의 전체 프로세스는 알고 있다고 가정하고 발표합니다
- 다양한 MLOps 프레임워크 하나 하나 깊게 살펴보는 발표가 아닙니다.
(추후에 하나씩 정리해보도록 할게요)
- MLOps의 큰 맥락에 대해 설명하는 발표입니다!
- 발표 자료가 조금 많은데(100쪽), 자료를 추후에 공유드릴 예정이니 너무 걱정하지 않으셔도 됩니다. 살짝 말이 빠를 수 있는데, 양해 부탁드립니다

MLOps란?

여러분들이 머신러닝을 공부하셨다면

머신러닝 정말 재미있어요!

우리가 알지 못하는 것들을 예측할 수 있다는 두근거림?



(실제론 모르는데 머리가 아는척함)



머신러닝 프로세스 - Research

[문제 정의] [EDA] [Feature 생성] [모델 학습] [예측]

머신러닝 프로세스 - Research

[문제 정의] [EDA] [Feature 생성] [모델 학습] [예측]

위 프로세스는 보통 자신의 컴퓨터, 서버 등에서 실행

머신러닝 프로세스 - Research

[문제 정의] [EDA] [Feature 생성] [모델 학습] [예측]

위 프로세스는 보통 자신의 컴퓨터, 서버 등에서 실행

고정된 데이터를 사용해 학습

[문제 정의] [EDA] [Feature 생성] [모델 학습] [예측]

위 프로세스는 보통 자신의 컴퓨터, 서버 등에서 실행

고정된 데이터를 사용해 학습

이런 머신러닝 모델을 웹, 앱 서비스에 적용하기!(=Production 환경, Real World)

[문제 정의] [EDA] [Feature 생성] [모델 학습]

우선 Python 기반으로 학습 코드가 만들어졌으니,
파이썬에서 API 서버를 만들 수 있는 프레임워크 중
Flask가 가볍다고 하니 만들어보자!



```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World!'
```

[문제 정의] [EDA] [Feature 생성] [모델 학습]

서버는 어떻게 해야하지?
Docker 이미지로 만들어 인스턴스에 띄우자!
또는 AWS Lambda(서버리스) 등을 활용



Flask

+



docker

[문제 정의] [EDA] [Feature 생성] [모델 학습]

또는 꼭 API 형태로 만들지 않고
DB에 있는 데이터를 Batch(=많은 데이터) 단위로 처리하자!
1시간에 1번씩 예측해서 결과를 DB에 저장하자!

Task Management 도구인 Airflow를 사용하자!



Apache
Airflow

모델을 API 형태 또는 Batch 형태로 배포한 후, 생기는 일

모델을 API 형태 또는 Batch 형태로 배포한 후, 생기는 일

1) (운영팀 제보) 모델의 결과값이 이상한 것 같아요..!



모델을 API 형태 또는 Batch 형태로 배포한 후, 생기는 일

1) (운영팀 제보) 모델의 결과값이 이상한 것 같아요..!

(당연히) 결과값을 어딘가 저장해서 재현 가능



모델을 API 형태 또는 Batch 형태로 배포한 후, 생기는 일

1) (운영팀 제보) 모델의 결과값이 이상한 것 같아요..!

1-1) Input 데이터는 0 ~ 23까지의 데이터가 들어가야 하는데 77이란 데이터가 들어갔네..?

1-2) Float 타입으로 들어가야 하는데 Int로 들어왔네..?



모델을 API 형태 또는 Batch 형태로 배포한 후, 생기는 일

- 2) 모델 성능 어떤가요? 새롭게 배포한 모델은 어떤가요?
특정 카테고리에서 잘 맞추나요?



모델을 API 형태 또는 Batch 형태로 배포한 후, 생기는 일

- 2) 모델 성능 어떤가요? 새롭게 배포한 모델은 어떤가요?
특정 카테고리에서 잘 맞추나요?

다행히 모델 모니터링 대시보드 미리 만들어서 확인
Research 환경에서 더 좋았던 새로운 모델.
Production 환경에선 저번 모델이 더 좋다..?



모델을 **API** 형태 또는 **Batch** 형태로 배포한 후, 생기는 일

- 2) 모델 성능 어떤가요? 새롭게 배포한 모델은 어떤가요?
특정 카테고리에서 잘 맞추나요?

과거 학습한 모델 **pkl** 파일 **S3**에 저장해뒀으니
다시 배포하자



머신러닝 프로세스 - Production

(사실 제 과거 이야기)

휴 이제 모델링에 집중할까 했는데, 새벽 2시네..?

모델링에 집중하고 싶은데 할 일이 너무 많다..

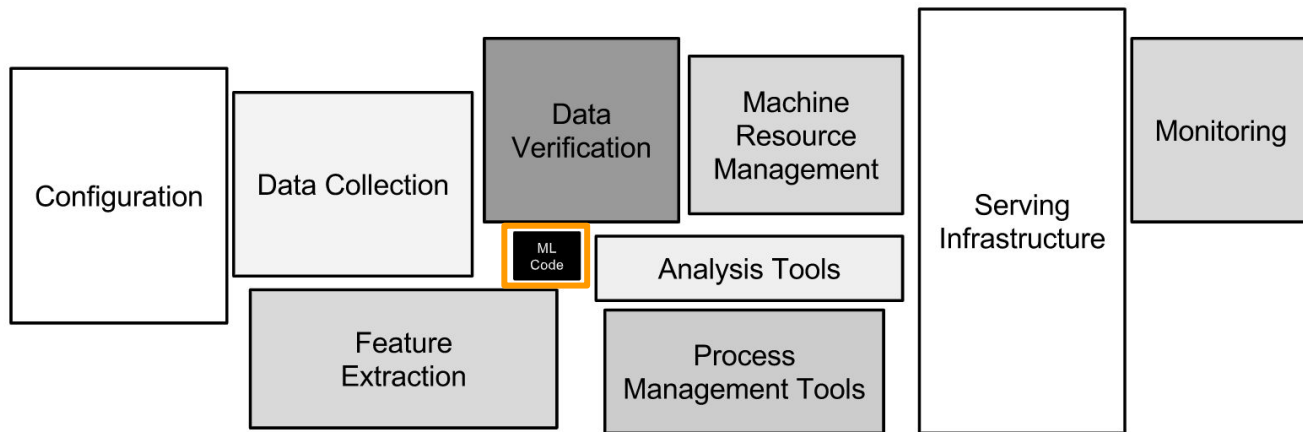


MLOps란?

정말 유명한 논문인 “Hidden Technical Debt in Machine Learning Systems”

번역 글

1줄 정리 : 머신러닝 코드는 머신러닝 시스템 중 일부에 불과하다! 다른 부분이 정말 많다



MLOps란?

MLOps = 머신러닝 엔지니어링 + 데이터 엔지니어링 + 인프라

ML + Ops

MLOps란?

MLOps = 머신러닝 엔지니어링 + 데이터 엔지니어링 + 인프라

ML + Ops

머신러닝 모델 개발(ML)과 머신러닝 모델 운영(Ops)에서 사용되는 문제, 반복을 최소화하고 비즈니스 가치를 창출하는 것이 목표

모델링에 집중할 수 있도록 관련된 인프라를 만들고, 자동으로 운영되도록 만드는 일

예 : API 형태로 서버 만들기, 실험 파라미터와 결과 저장하기, 모델 결과 자동화하기, 데이터 Validation 등

MLOps란?

최근엔 비즈니스 문제에 머신러닝/딥러닝을 적용하는 **Case**가 많아짐

Production 환경에 배포하는 과정엔 **Research**의 모델이 재현 가능해야 함

+ 현실의 **Risk** 있는 환경에서 잘 버틸 수 있어야 함

MLOps의 목표는 빠른 시간 내에 가장 적은 위험을 부담하며 아이디어 단계부터 프러덕션까지 **ML** 프로젝트를 진행할 수 있도록 기술적 마찰을 줄이는 것

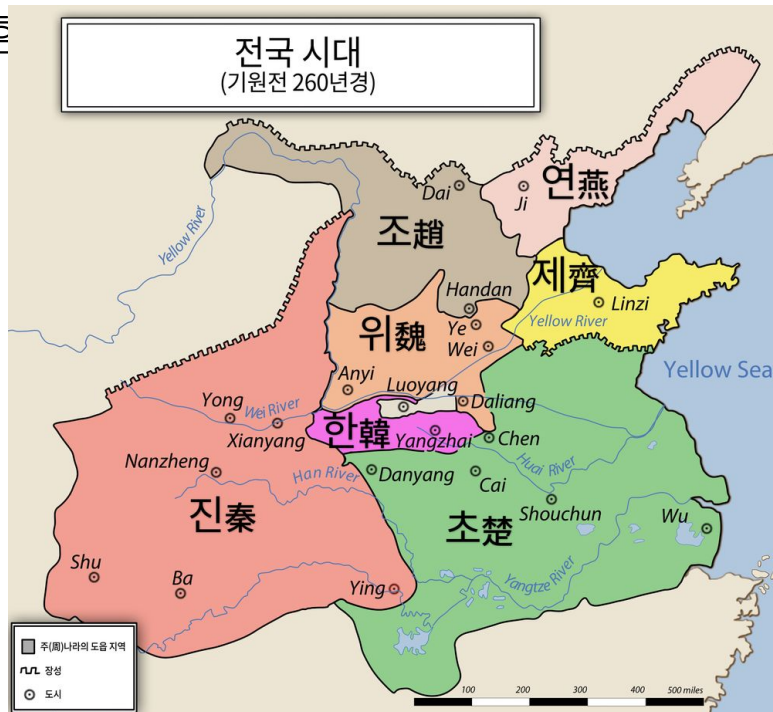
Research와 Production의 머신러닝

	Research ML	Production ML
데이터	고정 (Static)	계속 변함 (Dynamic - Shifting)
중요 요소	모델 성능 (Accuracy, RMSE 등)	모델 성능, 빠른 Inference 속도, 해석 가능성
도전 과제	더 좋은 성능을 내는 모델, 새로운 구조의 모델	안정적인 운영, 전체 시스템 구조
학습	데이터가 고정이라 모델구조, 파라미터 기반 재학습	시간의 흐름에 따라 데이터가 변경되어 재학습
목적	논문 출판	서비스에서 문제 해결
표현	Offline	Online

요즘 MLOps

춘추 전국 시대

- 정말 다양한 라이브러리들이 있고
- 이 라이브러리가 **Best다!** 라는 내용이 거의 확립되지 않음
- 클라우드에서도 **MLOps** 시장 점유를 목표로 함



MLOps Component

MLOps Component

모두 점심 식사는 하셨나요?

어떤 음식 좋아하시나요?

여러분 요리 좋아하시나요?

MLOps Component

[저는 가끔 집에서 타코를 만들어요]



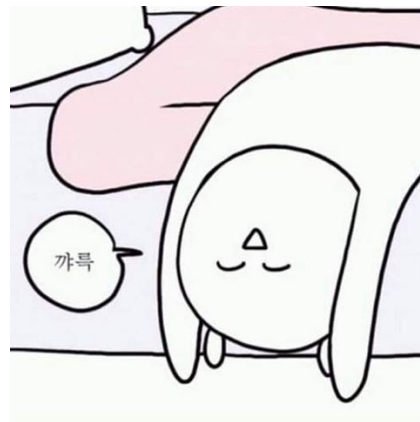
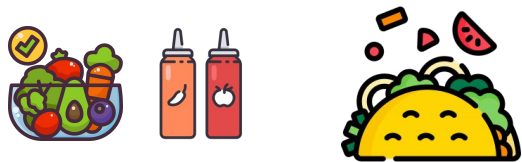
MLOps Component

[여러 야채와 소스를 가지고 타코를 만들어요]



MLOps Component

[제가 만든 타코가 너무 맛있어서, 가끔 놀라면서 고민에 빠집니다 - 장사할까?]



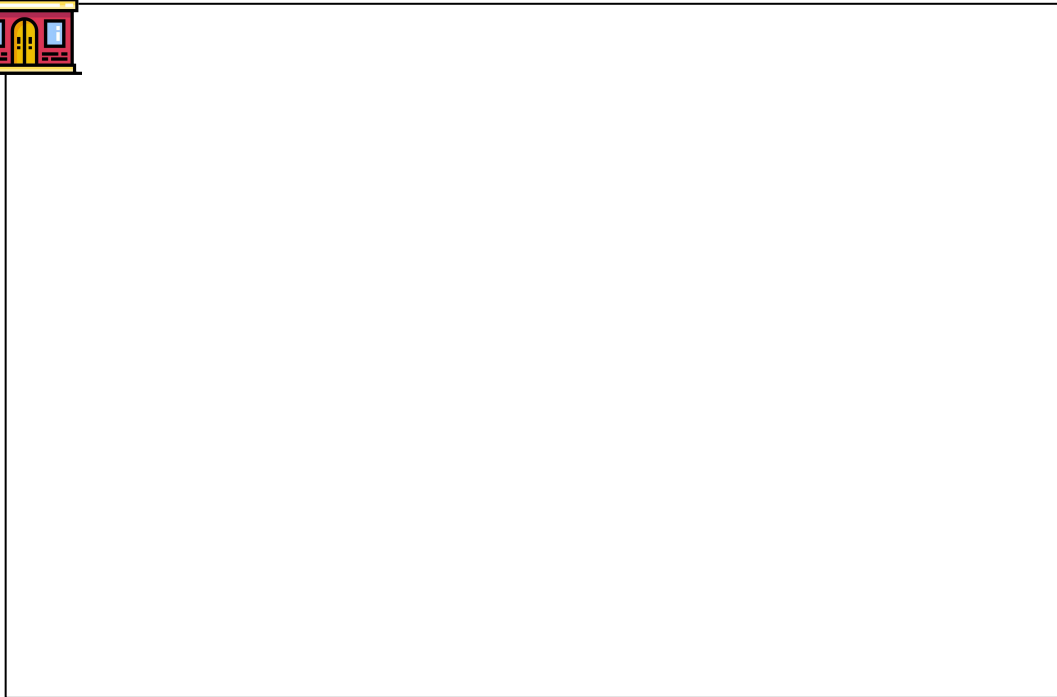
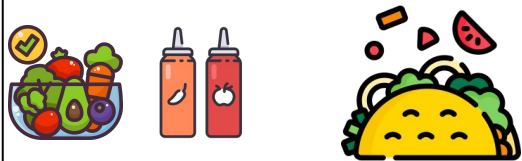
MLOps Component

[그래서 타코 레스토랑을 만든다고 가정 해봅시다]



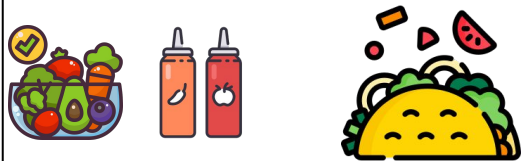
MLOps Component

[구분을 위해 집과 레스토랑의 구간을 나누겠습니다]



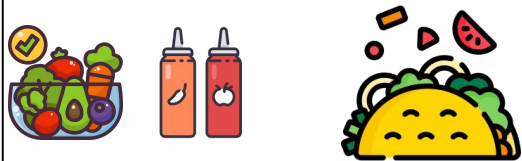
MLOps Component

[재료 납품, 업소용 소스를 사용해야 함. 비슷하지만 살짝 다른 재료!]



MLOps Component

[여기서 잠깐. 제가 무슨 이야기는 하는걸까요?]

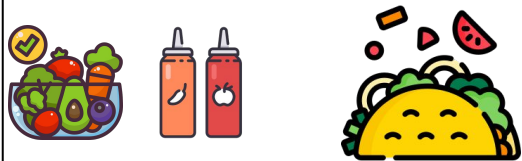


MLOps Component

[집 = Research, 레스토랑 = Production]



Research



Production



MLOps Component

[음식 = 모델]



Research



Model



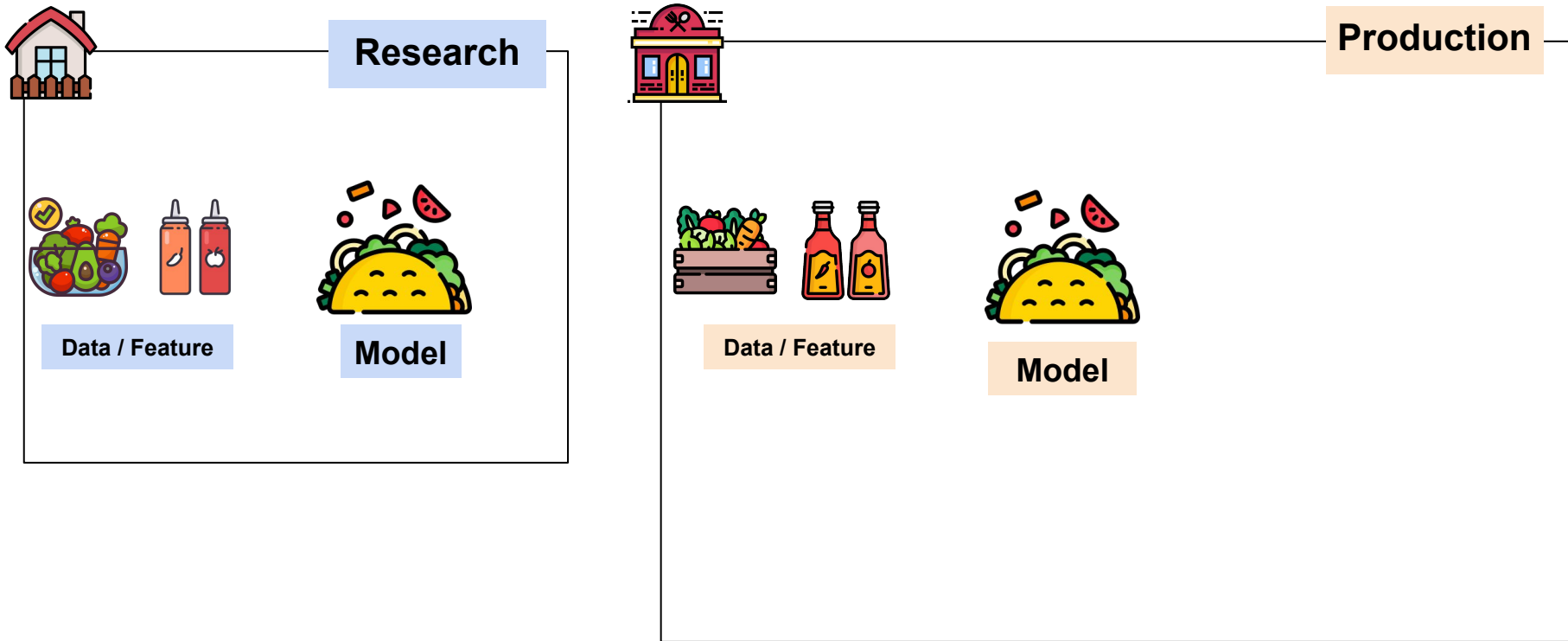
Production



Model

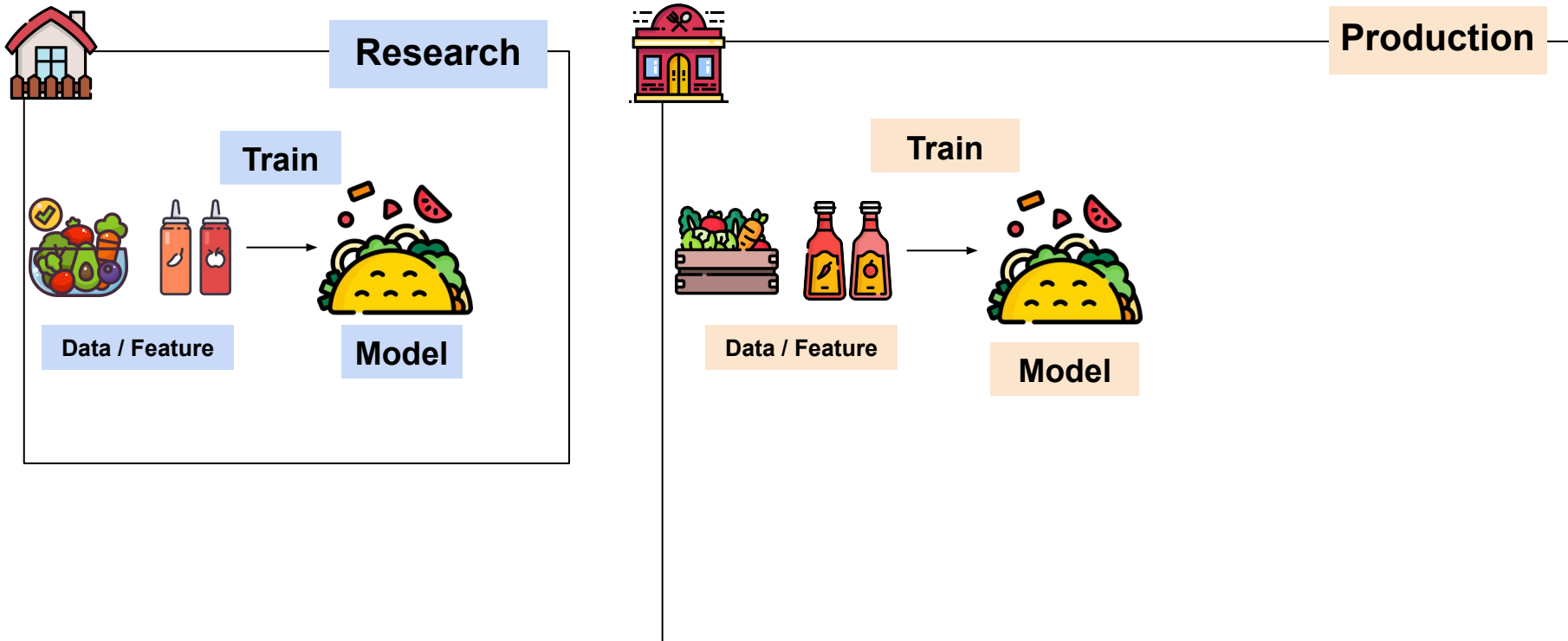
MLOps Component

[식재료, 소스, 밀가루 빵 = Data / Feature]



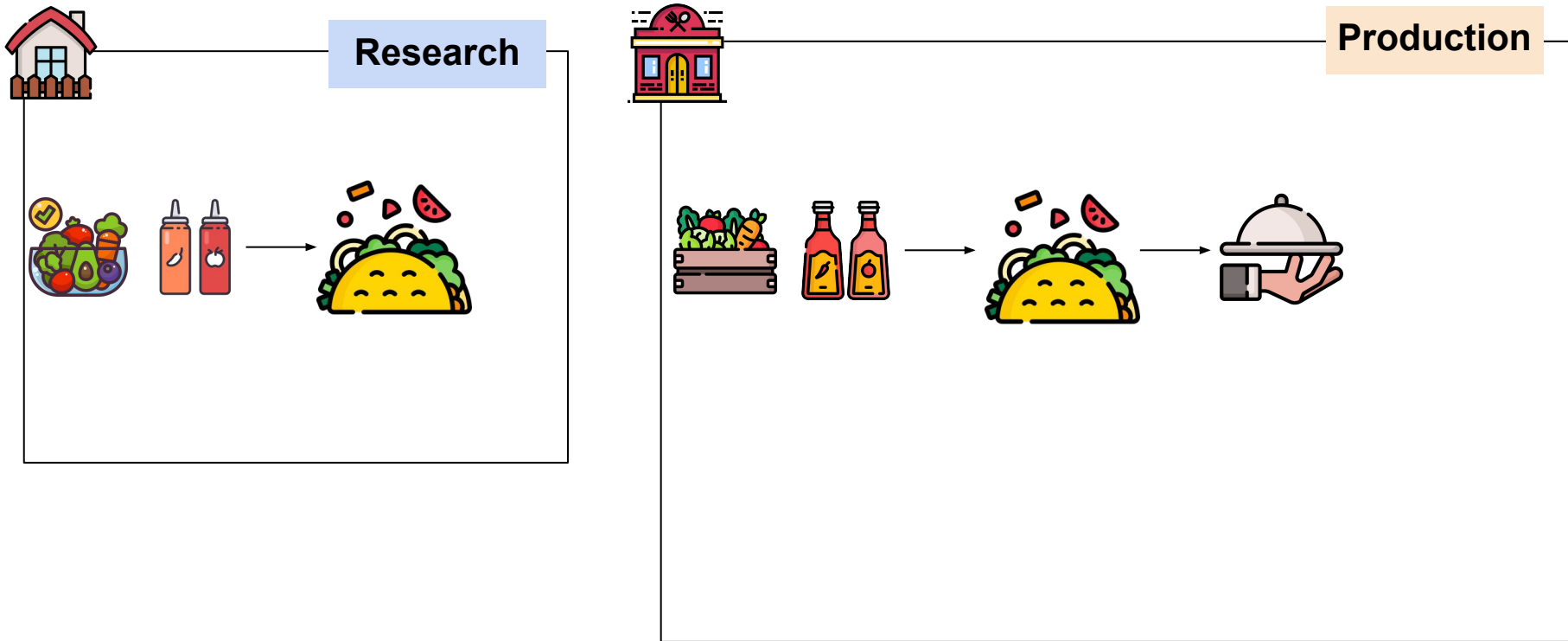
MLOps Component

[요리하는 행위 = 모델 Train]



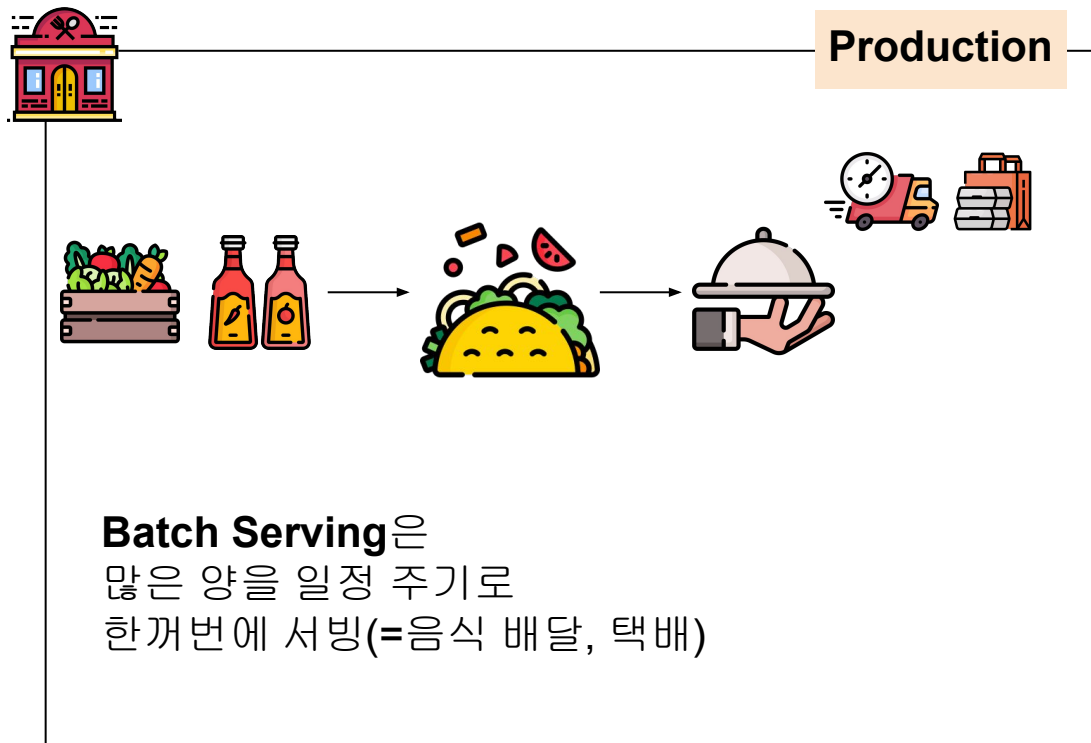
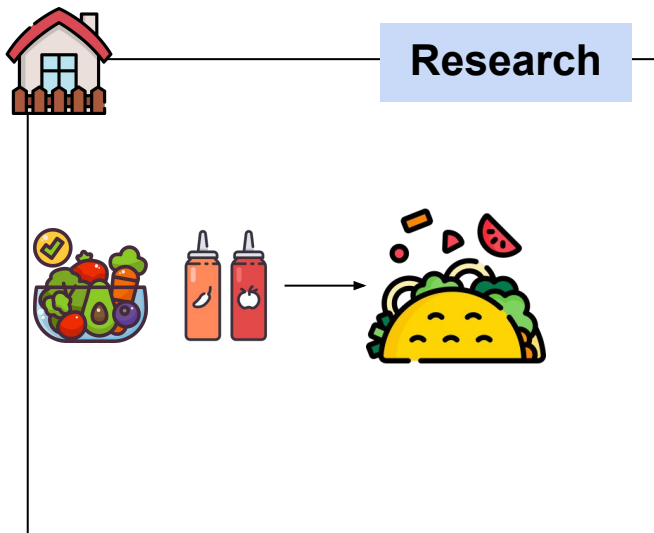
MLOps Component - Serving

[요리를 만들면 이제 손님에게 서빙해야죠!]



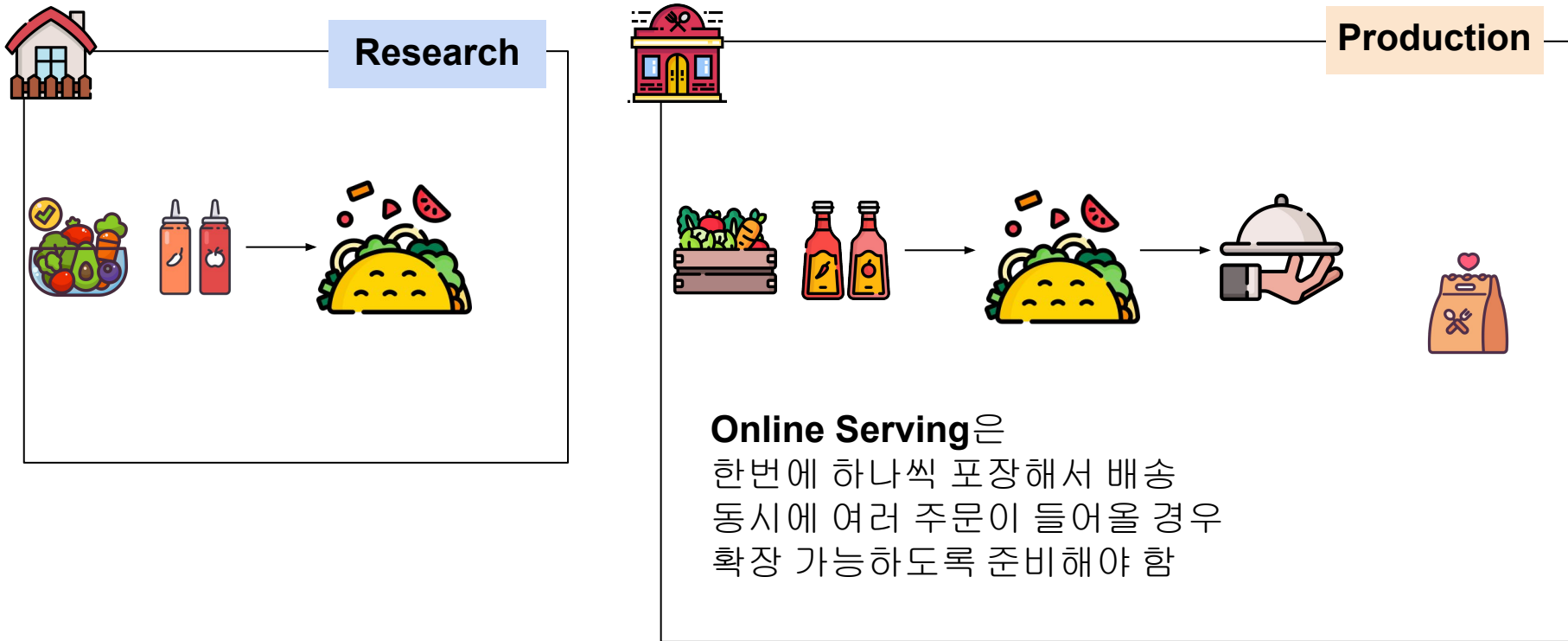
MLOps Component - Serving

[정기 배송처럼 매일 20시마다 서빙받길 원하는 경우엔 **Batch Serving**을 하게 됨]



MLOps Component - Serving

[주문하자마자 만들어서 전달하는 경우엔 **Online Serving**이라 함]



MLOps Component - Serving

Serving : Production(Real World) 환경에 사용할 수 있도록 모델을 배포

대표적인 Serving 방식

- 1) Batch 단위로 여러 데이터를 한번에 예측하는 방법
- 2) API 형태로 요청이 올 때마다 예측하는 방법

Serving 환경의 의존성 : 라이브러리, 파이썬 버전 등

MLOps Component - Serving(Batch)

Batch Serving 관련한 라이브러리는 거의 없음.
익숙한 라이브러리로 작성하고 스케줄링 실행!

Airflow, Cronjob 등으로 스케줄링 작업

학습 / 예측을 별도의 작업으로 설정

학습 : 1주일에 1번

예측 : 10분, 30분, 1시간에 1번씩

MLOps Component - Serving(Online, API 형태)

Lv 1. Flask, Fast API

Lv 2. Lv 1 + Docker

Lv 3. Lv 2 + Kubernetes

Lv 4. Serving 프레임워크 사용

처음엔 Lv 1으로 시작하다 점점 더 편하고 빠르게 모델을 Serving하기 위해 Lv 4를
향함

(모델이 많은 경우 이런 식으로 효율성 증대를 목표로 하고, 모델이 하나라면
심플하게 가는 것도 좋음)

Kubeflow, BentoML, Seldon Core, Cortex, KFServing, Tensorflow Serving, Torch
Serve 등

MLOps Component - Serving(트렌드)

처음부터 **API** 형태로 **Serving** 해야하는 것은 아님. 현재 상황에 따라 선택!

서버와 실시간 통신을 꼭 해야하는 것이 아니라면, 최초엔 **Batch Serving**을 구축하는 것도 좋음. **Batch Serving**의 결과를 **DB**에 저장하고, 서버는 그 데이터를 주기적으로 가져가는 방식으로 통신

우선 머신러닝 모델을 운영하면서 점점 **API** 형태로 변환

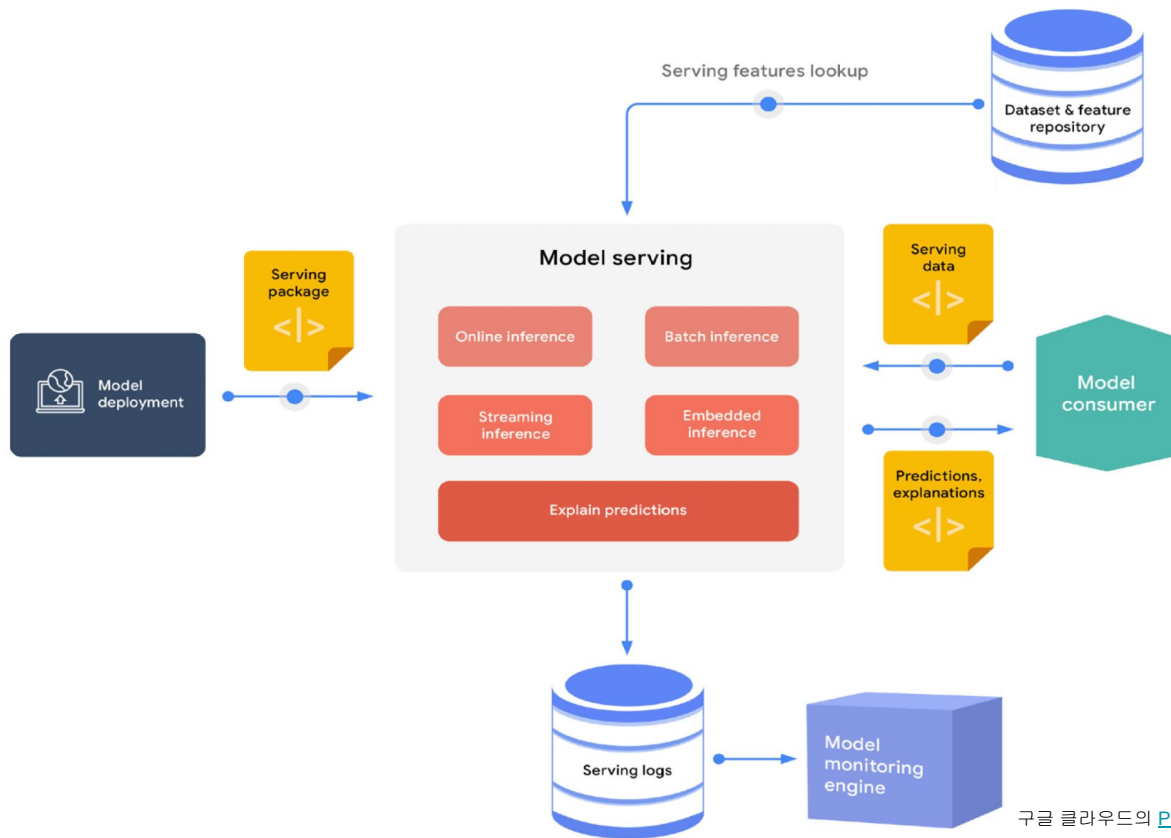
MLOps Component - Serving(트렌드)

Serving 라이브러리는 점점 간단한 코드만 작성하면,
Docker Image를 명령어 하나로 만들 수 있는 기능을 지원

예) BentoML

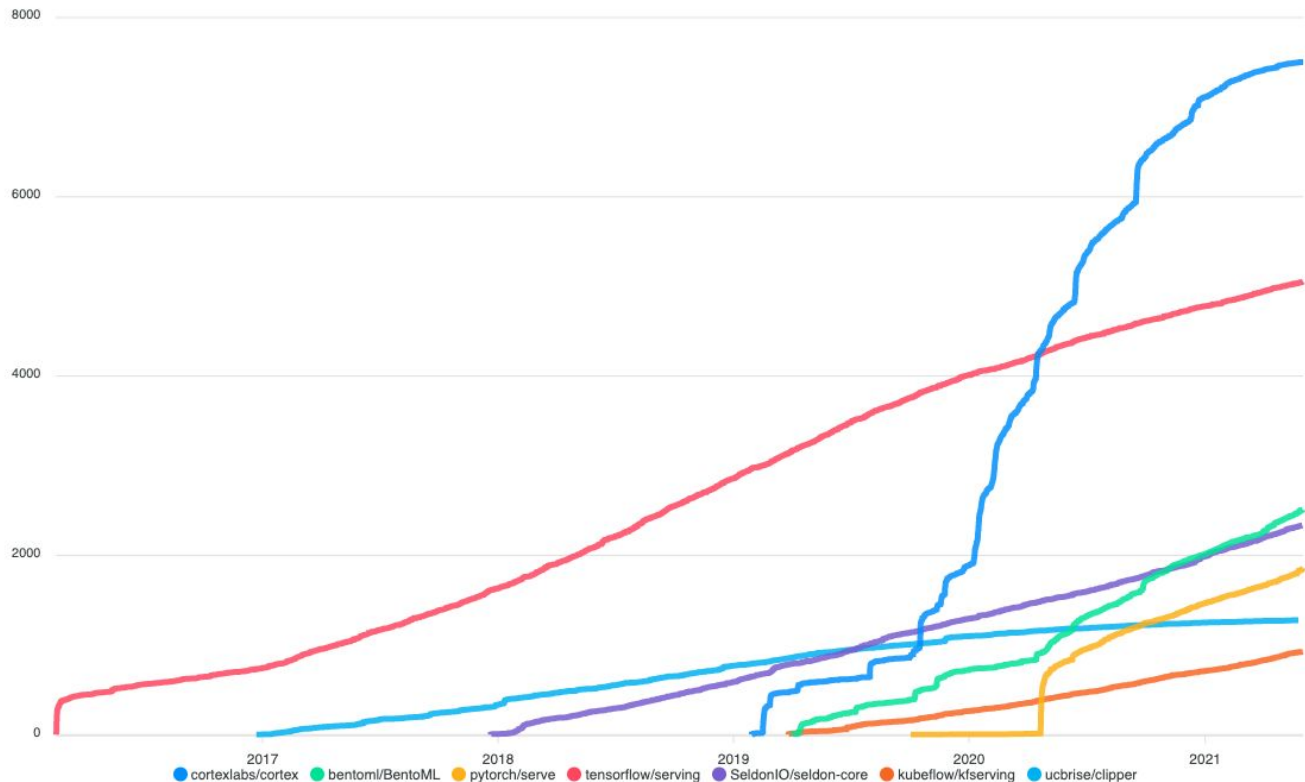


MLOps Component - Serving(트렌드)



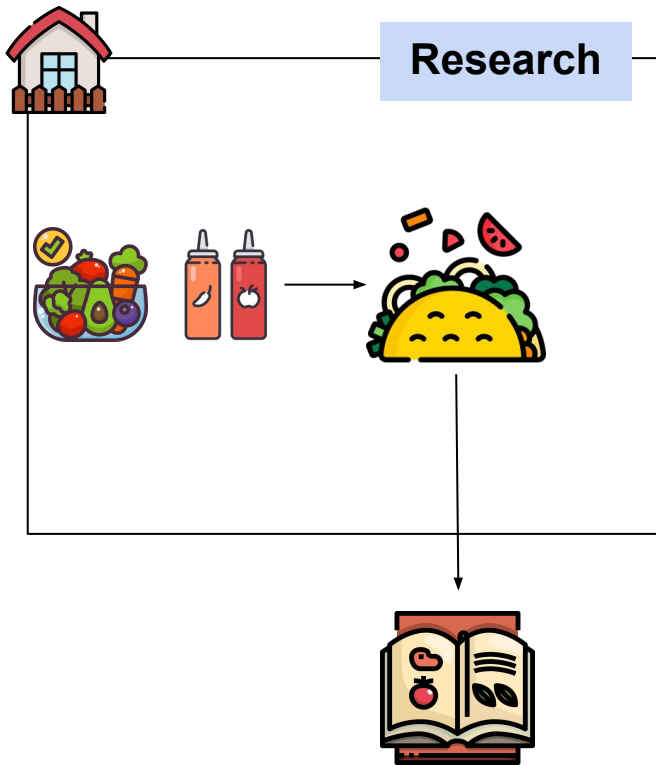
MLOps Component - Serving(대표적인 라이브러리)

Github Star History



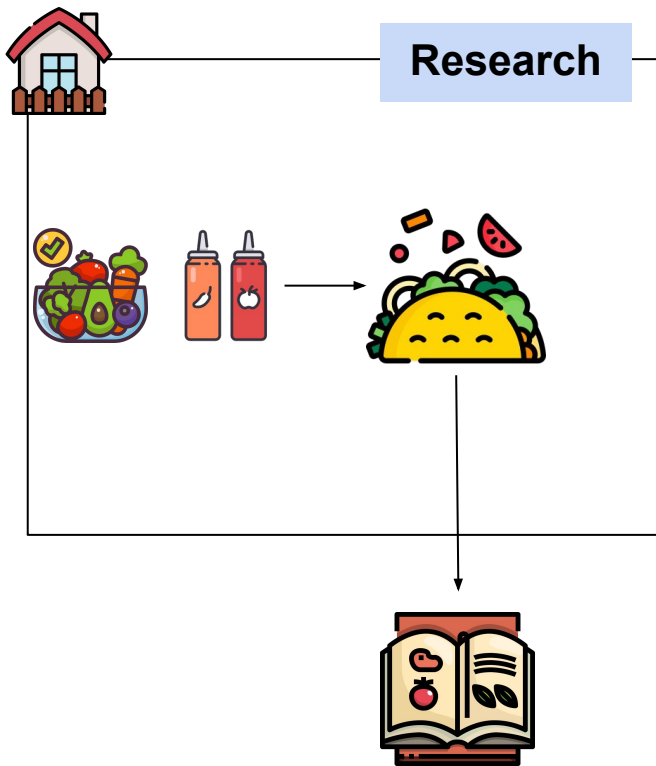
MLOps Component - Experiment, Model Management

[집에서 요리 만들 때, 레시피를 기록해야 어떤 조합이 좋은지 알 수 있음(파라미터, 모델 구조 등)]



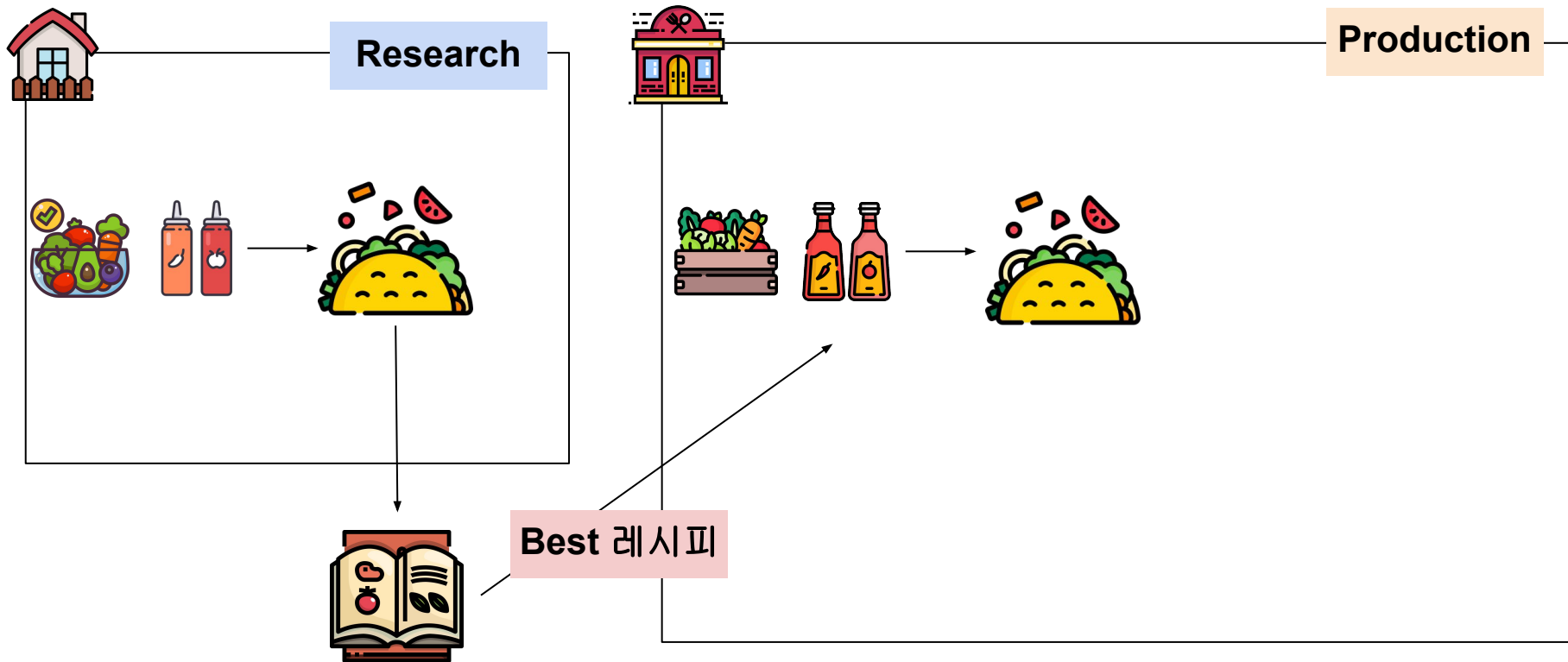
MLOps Component - Experiment, Model Management

[여러 시행착오를 겪으며 요리함(머신러닝 모델링도 많은 실험을 함!)]



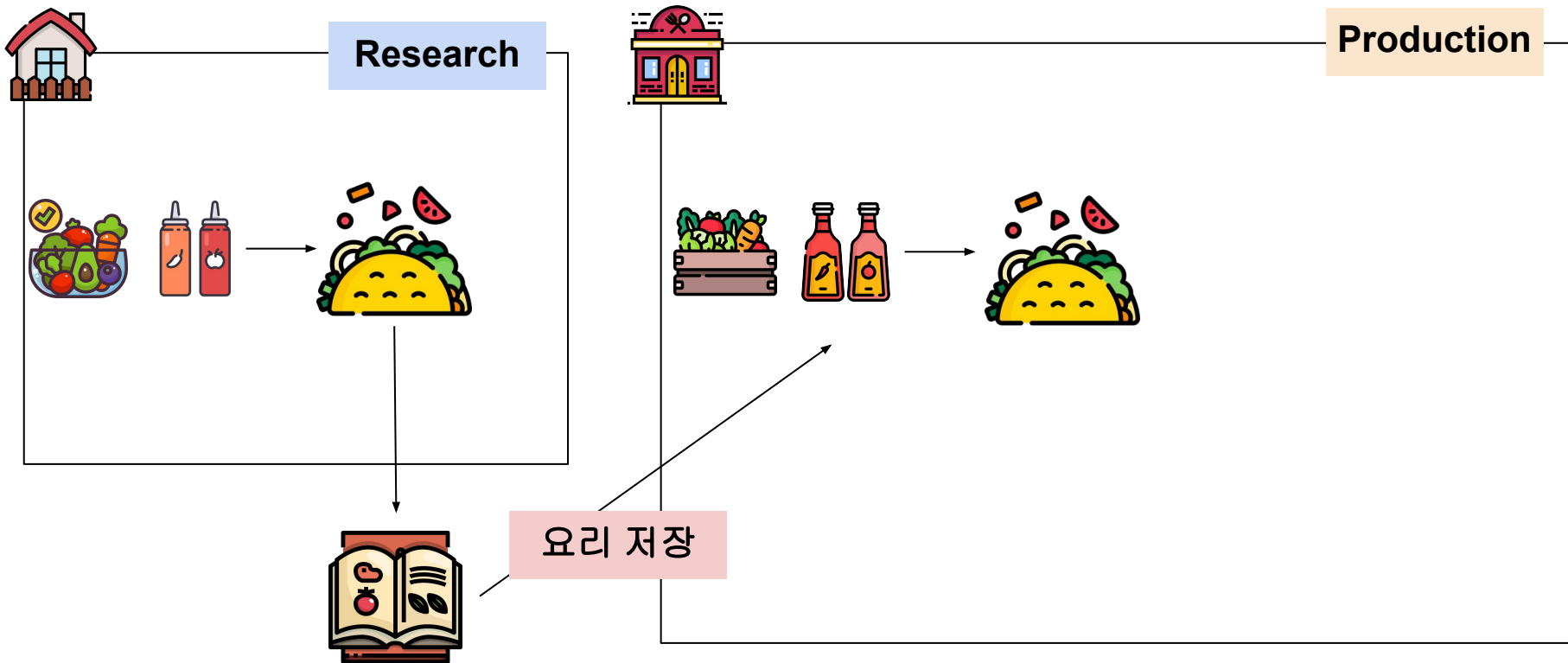
MLOps Component - Experiment, Model Management

[이 레시피에서 제일 맛있었던(성능이 좋았던) 레시피를 레스토랑에 사용하겠죠?]



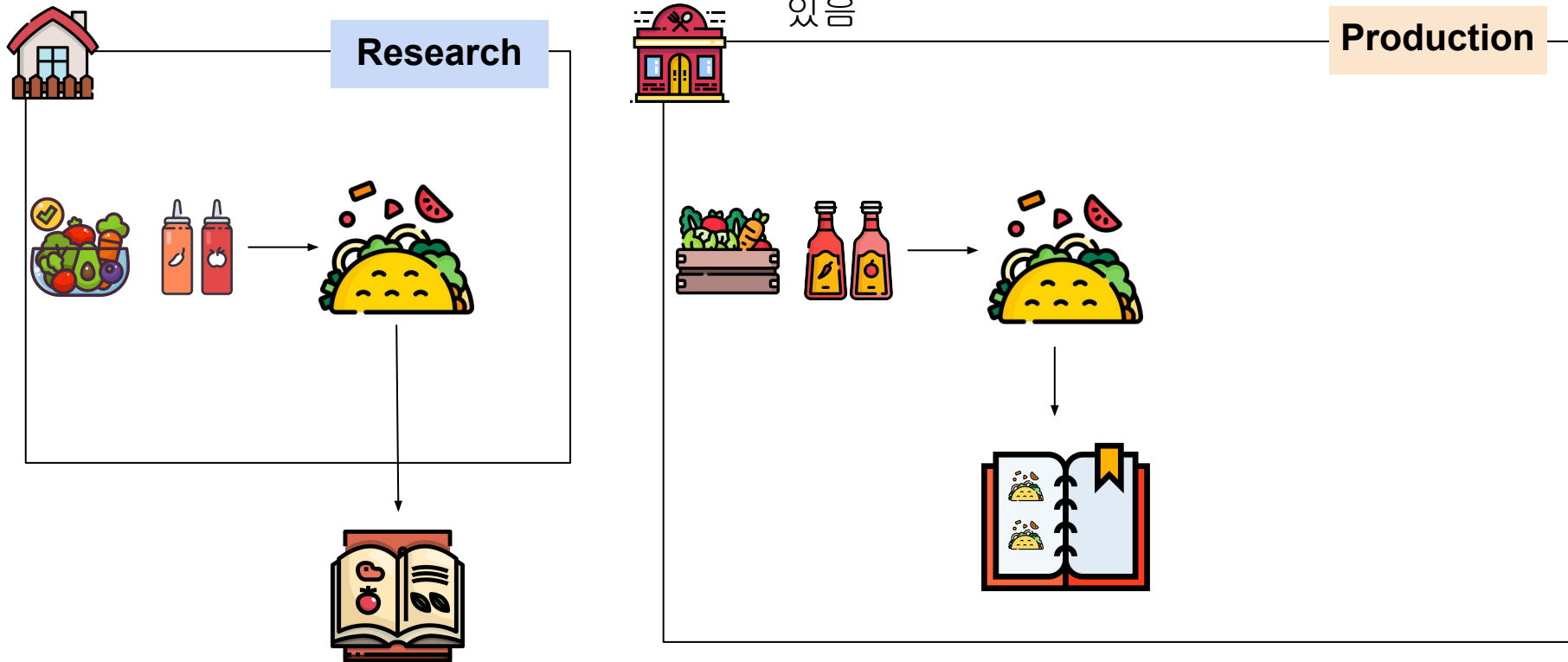
MLOps Component - Experiment, Model Management

[만들었던 요리는 하나 보관!(=모델 Artifact, 이미지 등)]



MLOps Component - Experiment, Model Management

타코(모델)는 다양한 종류가 있으므로
언제 만든 타코인지(=모델 생성일), 얼마나 맛있었는지(모델 성능), 유통기한 등을 기록해둘 수
있음



MLOps Component - Experiment, Model Management

머신러닝 모델 학습하는 과정을 매번 기록함

Artifact : 학습하는 과정에 생기는 이미지, 모델 파일 등

단순히 기록하는 것은 쉽고, 기록을 편하게 볼 수 있도록 **모니터링 대시보드**가 필요함

모든 실험을 기록하고, 그 중에 **특정 메트릭이 제일 좋은 것을** 선택하는 기능도 필요함

Weight & Bias, Neptune 등 SaaS가 꽤 많고, 오픈소스는 MLflow가 제일 강세



MLOps Component - Experiment, Model Management

Research 관점의 Model management, Production 관점의 Model management 필요

Serving하는 쪽과 연결해서 학습한 모델을 쉽게 배포할 수 있어야 함

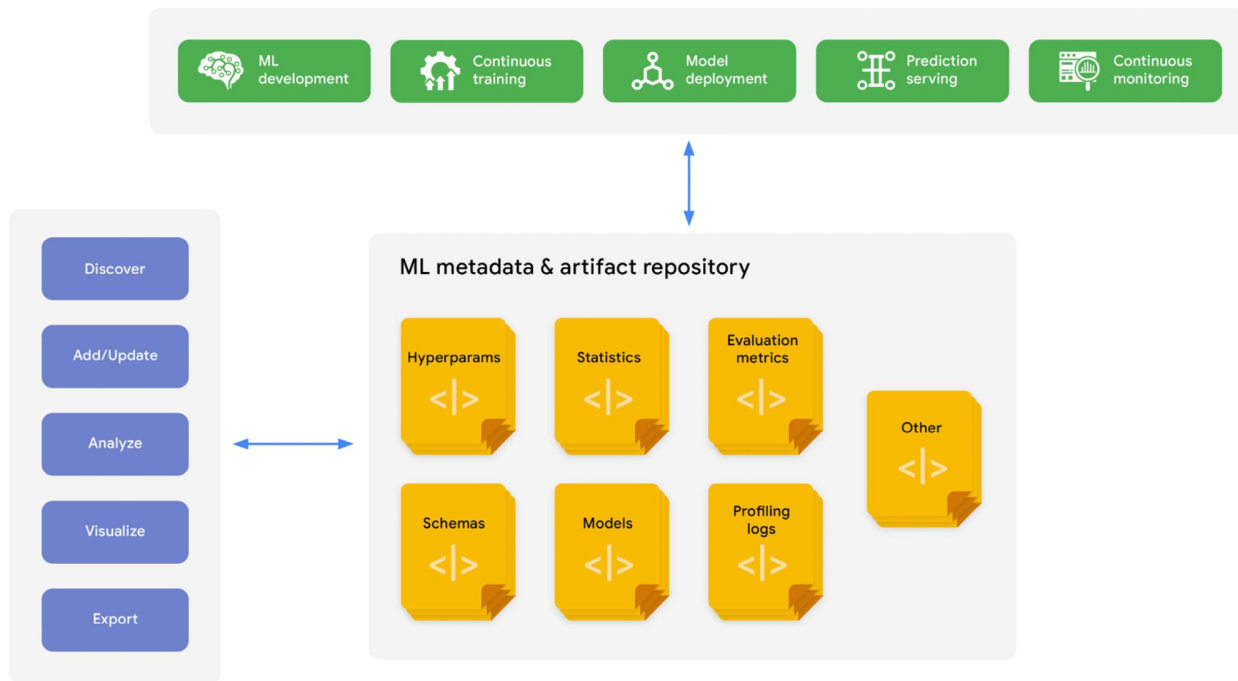
여러 모델 중 선택하는 로직의 예시

([A Machine Learning Model Management System at Uber](#) 논문)

Listing 1: Model Selection Rule Example

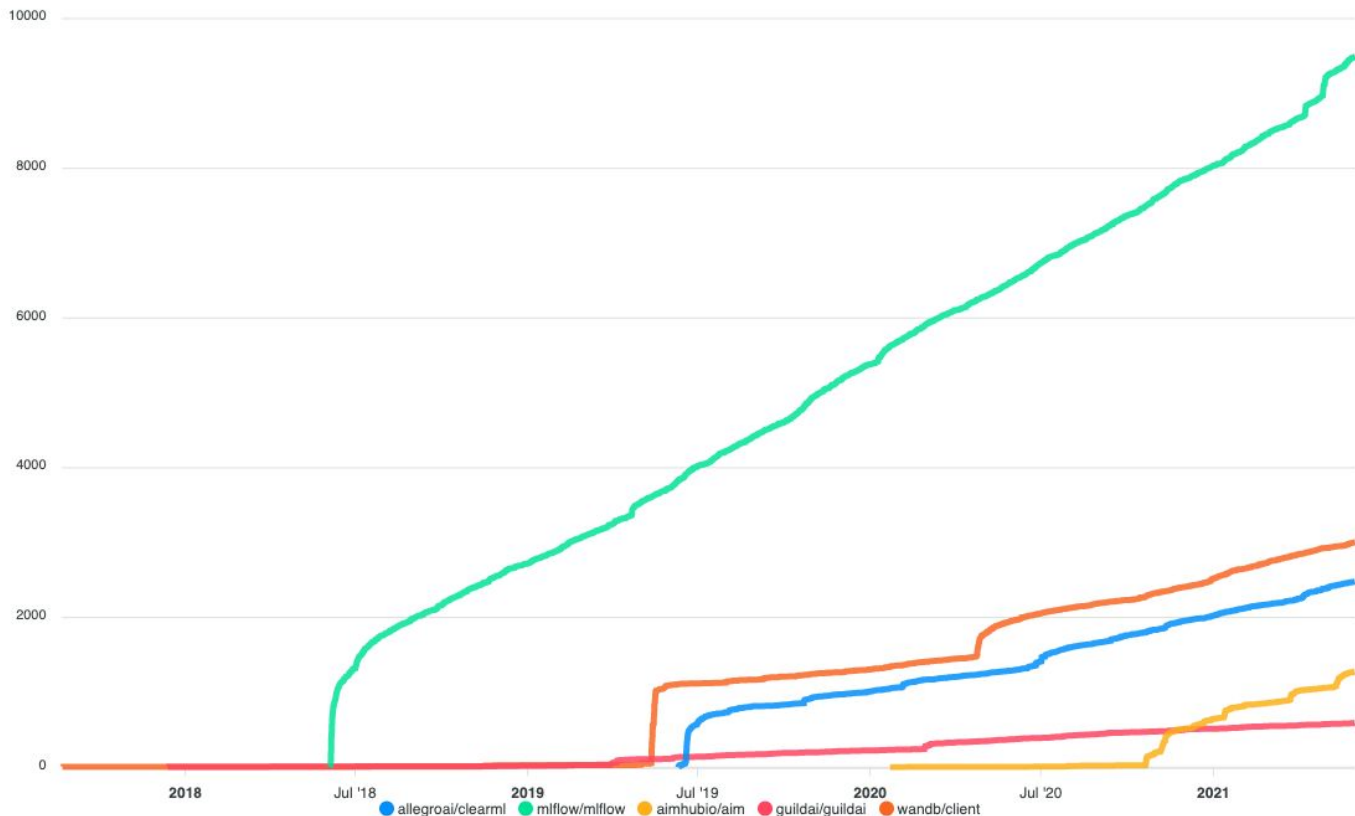
```
{
  "team": "forecasting",
  "uuid": "316b3ab4-2509-4ea7-8025-ca879dac61",
  "rule": {
    "GIVEN": modelName ==
      "linear_regression"
      AND model domain == "UberX",
    "WHEN": "metrics[\"mae\"] <= 5",
    "ENVIRONMENT": "production",
    "MODEL_SELECTION":
      "a.created_time > b.created_time"
  }
}
```

MLOps Component - Experiment, Model Management



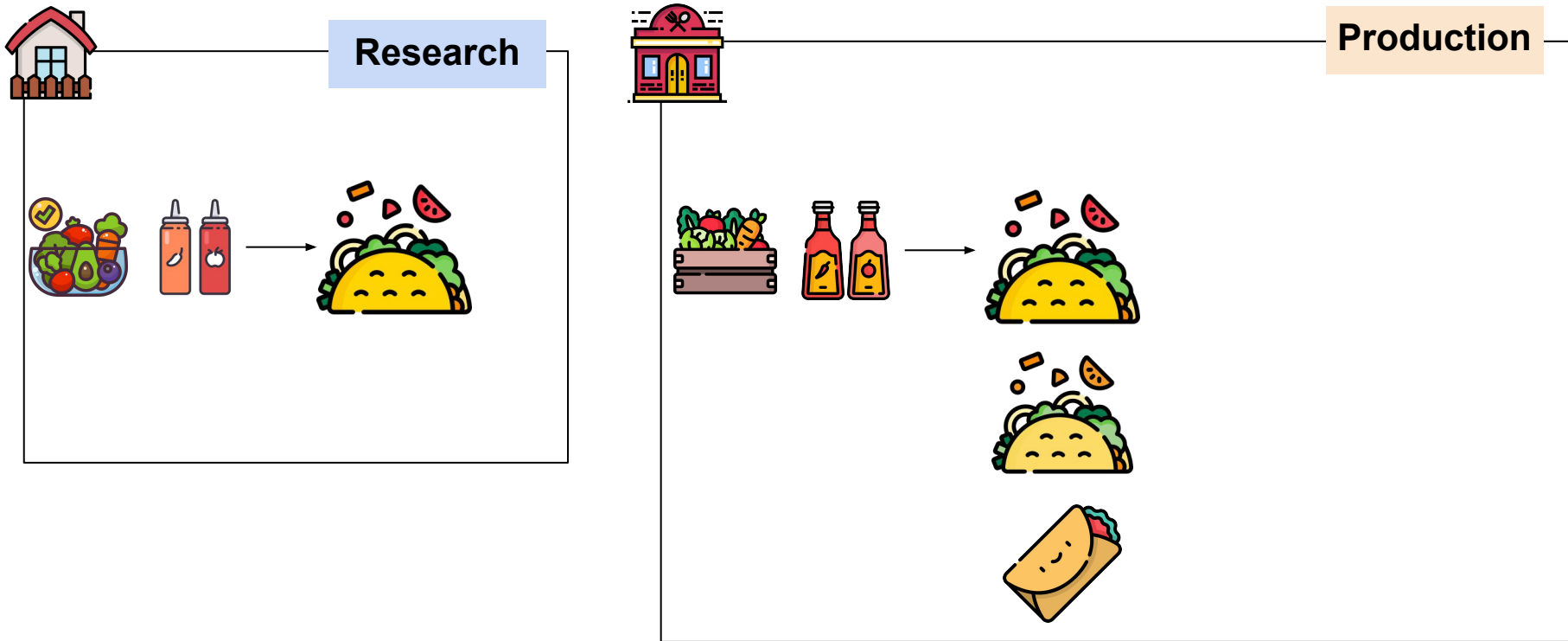
MLOps Component - Experiment, Model Management

Github Star History



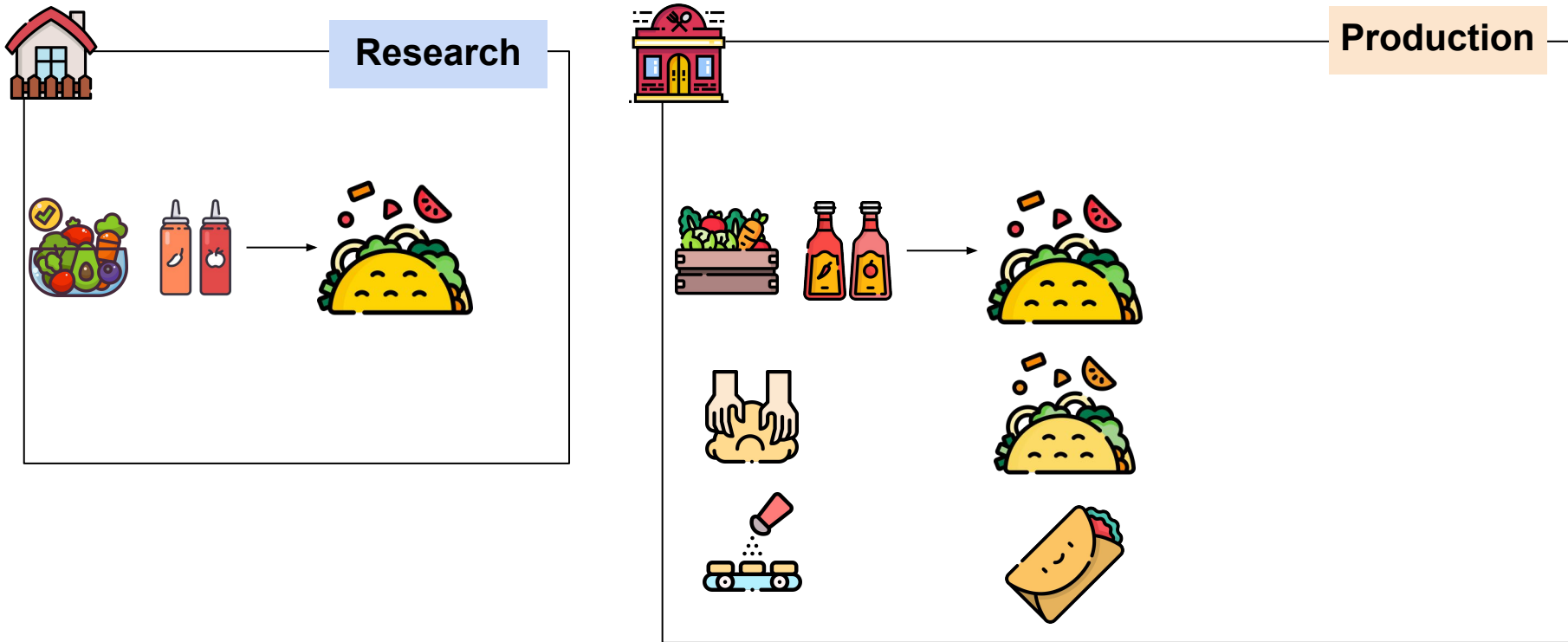
MLOps Component - Feature Store

[언제부터 닭고기 타코, 돼지고기 타코, 부리또를 만들어 판매함(=여러 모델 운영)]



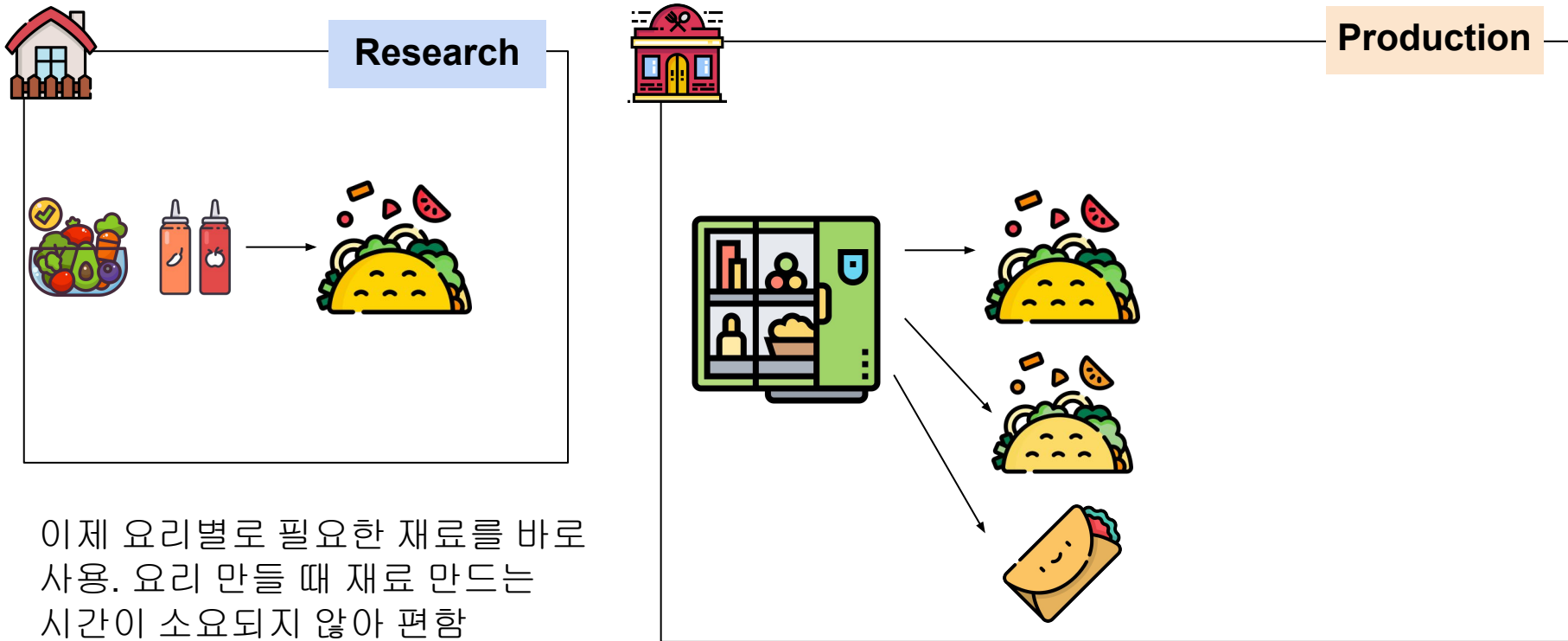
MLOps Component - Feature Store

[요리별로 사용되는 재료들이 중복됨. 반죽이나 간 등을 미리 만들면 편할듯]



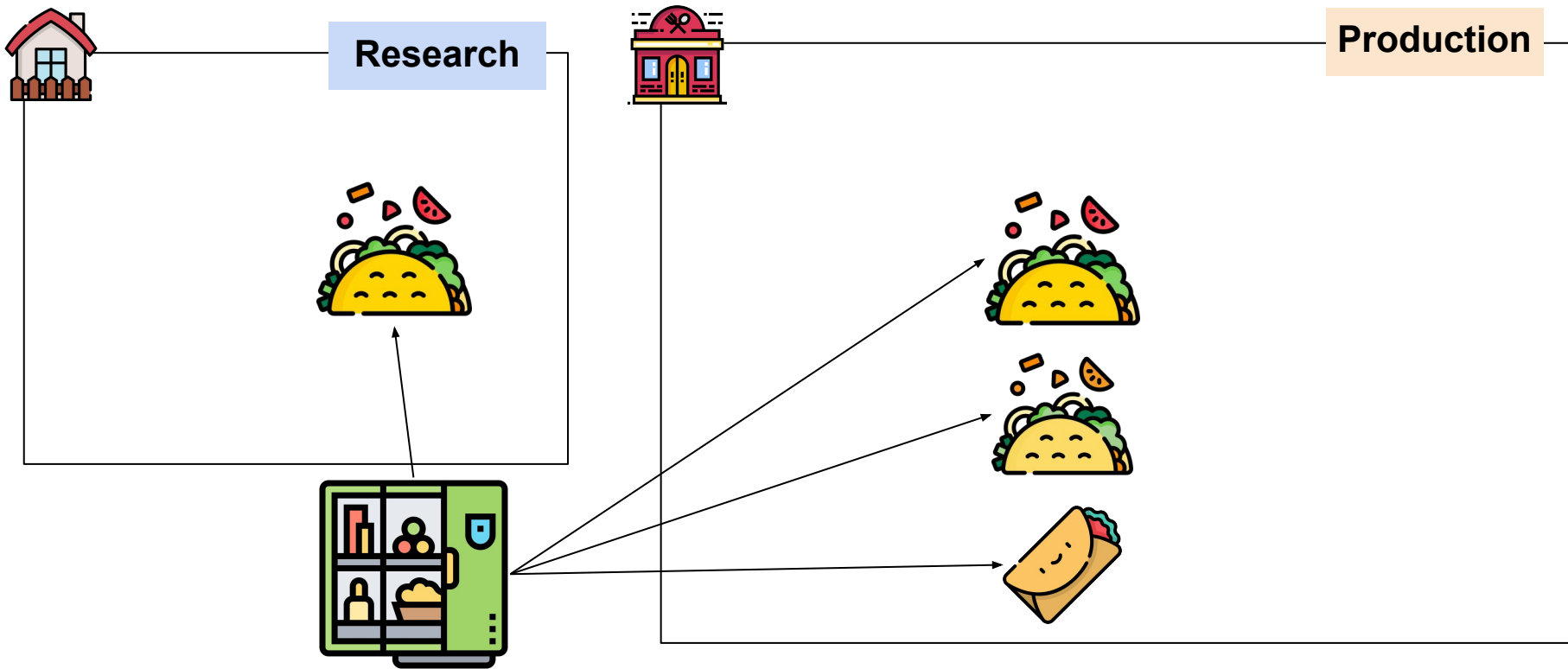
MLOps Component - Feature Store

[이런 재료를 가공해서 냉창고에 저장(=머신러닝 Feature를 집계한 Feature Store)]



MLOps Component - Feature Store

[집과 레스토랑에서 같은 재료를 사용하도록 냉장고 구축]



MLOps Component - Feature Store

모델이 많아지면 사용하는 **Feature**도 같은 경우가 존재

이런 경우 한번에 집계해서, 시간과 비용이 **Save!**

재사용을 가능하게 만듦

MLOps Component - Feature Store

최초엔 Batch 단위로(예 : 5분 단위) 쿼리를 실행해서 DB Table에 저장하고, 그 Table을 Feature Store로 사용하는 방법도 있음

그 이후엔 실시간 데이터를 가지고 Feature Store 구성

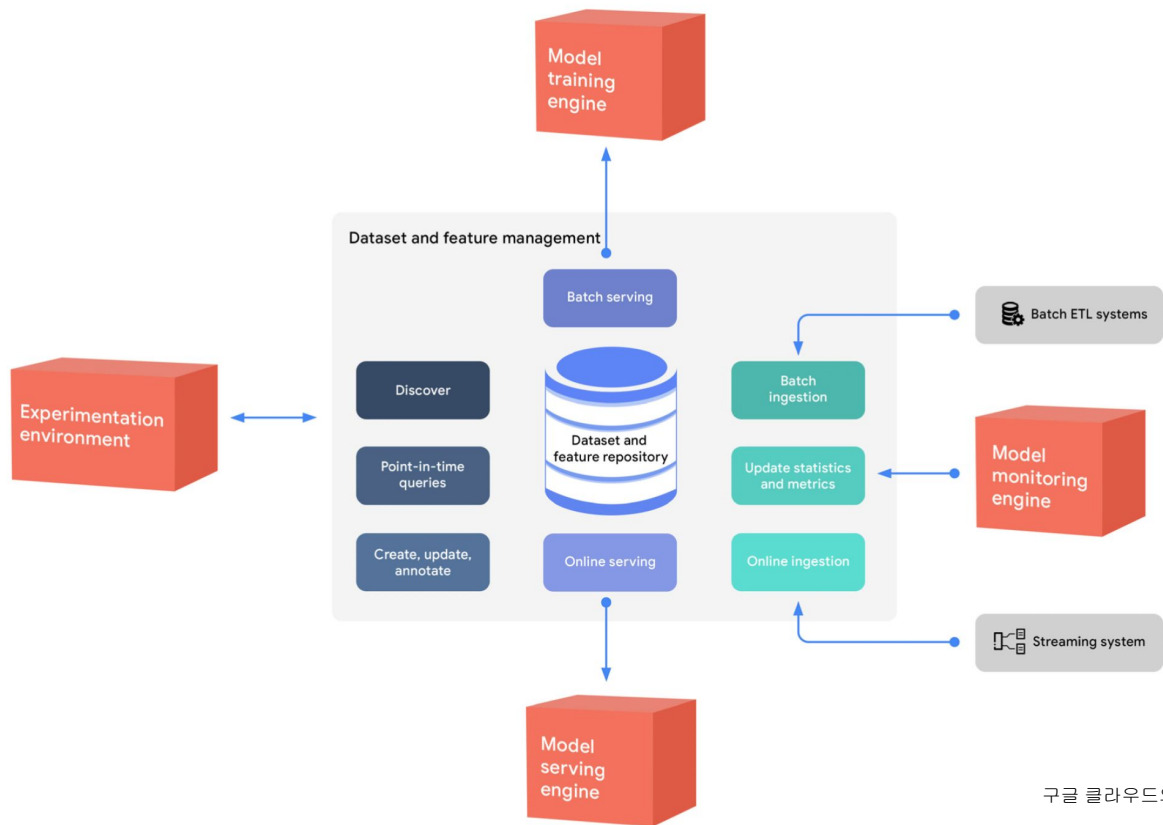
Research에서 사용하는 Feature와 Production Feature를 동일하게 사용할 수 있게 구성하는 것이 핵심!

오픈소스는 Feast, Hopsworks가 대표적이며 클라우드 서비스가 점점 생기는 중

SageMaker feature store(Amazon)

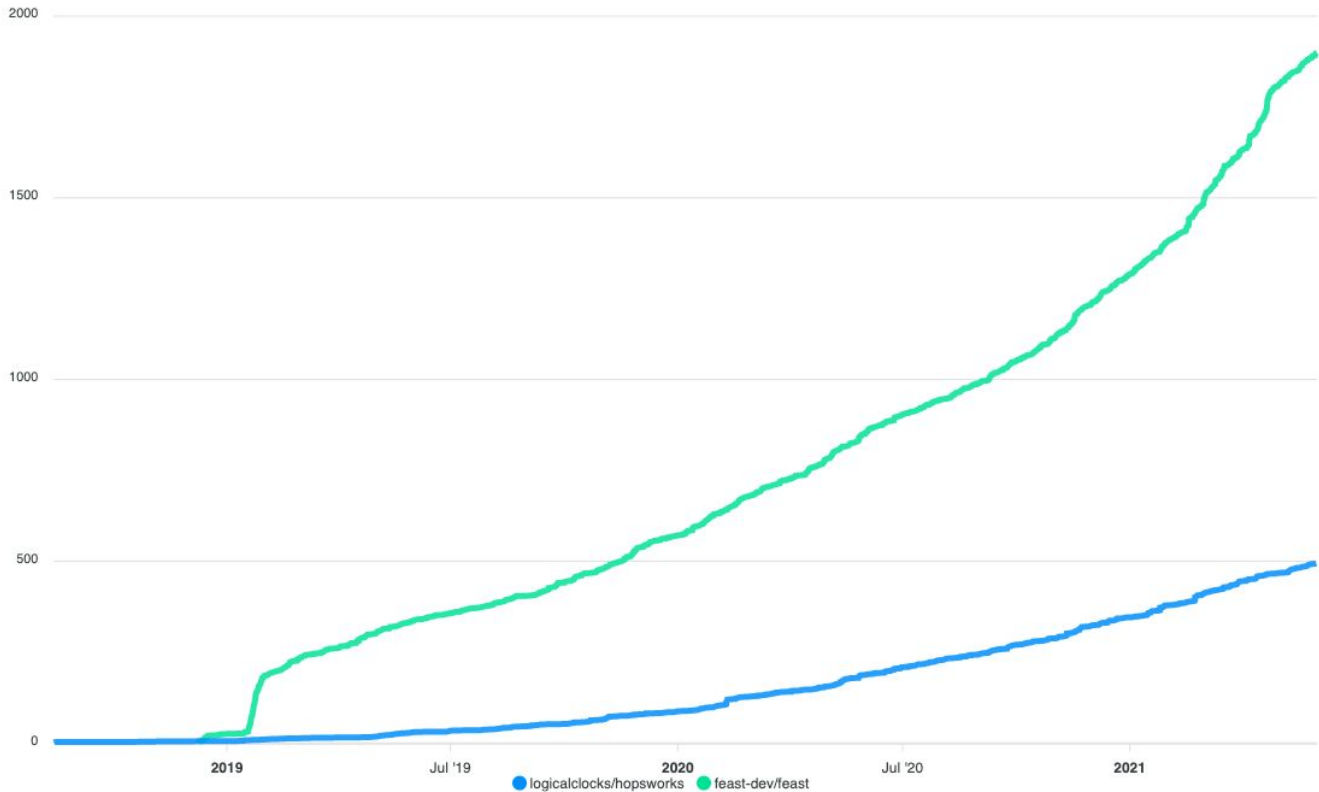
Vertex AI(Google)

MLOps Component - Feature Store



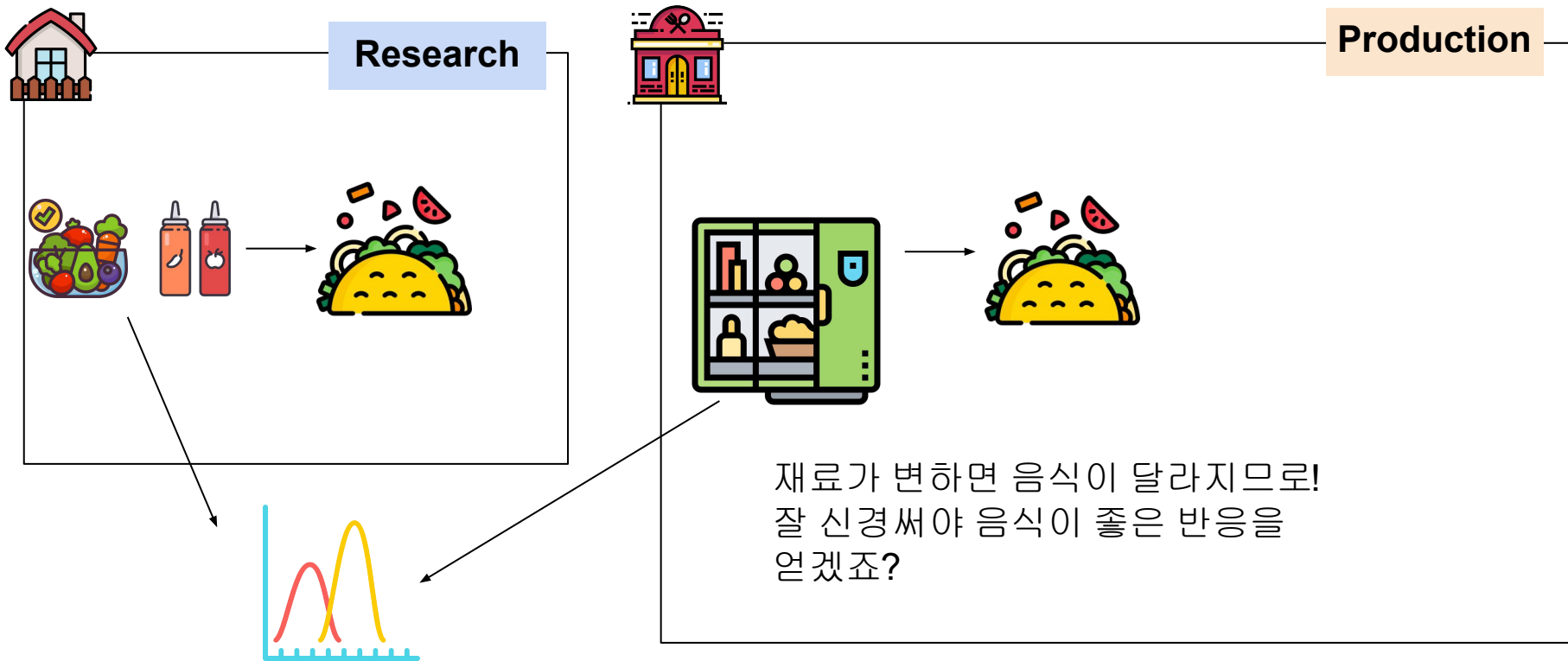
MLOps Component - Feature Store(라이브러리)

Github Star History



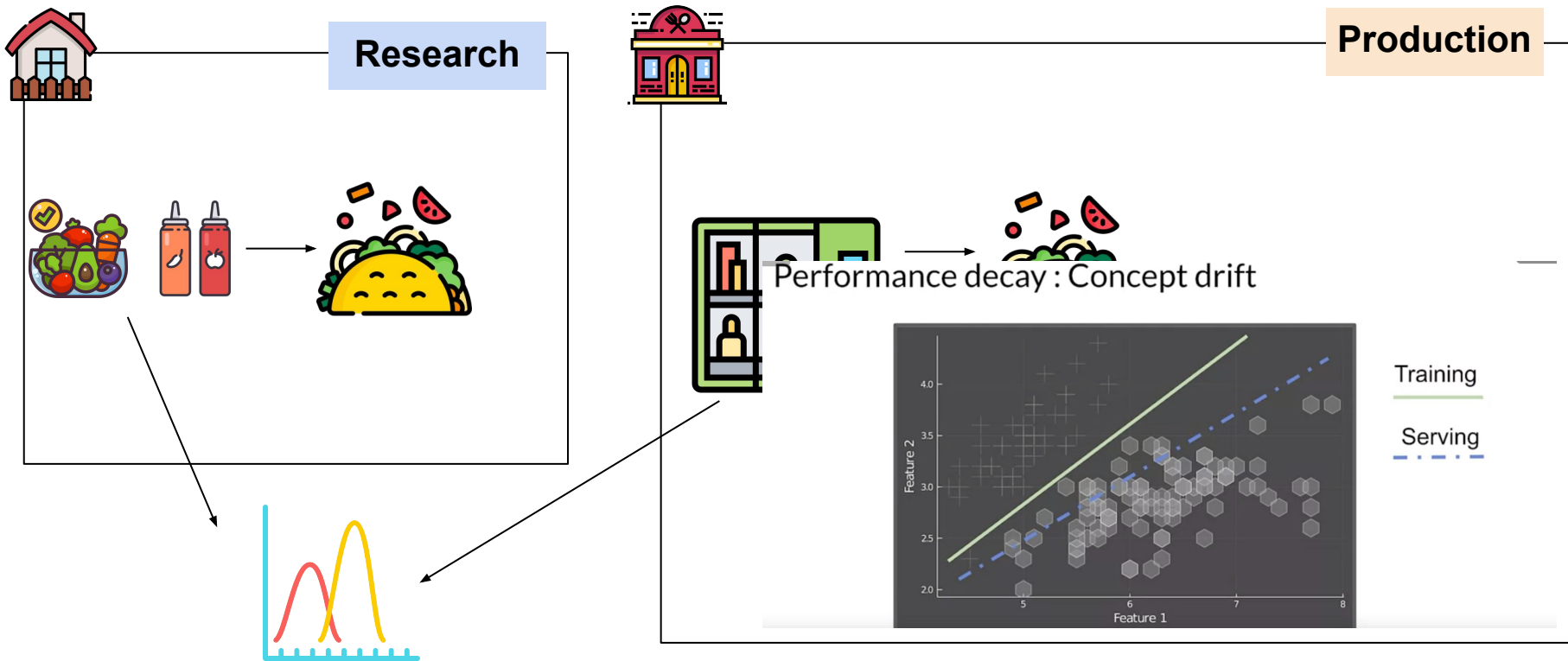
MLOps Component - Data Validation

[재료들이 집에서 사용하던 것과 비슷한지 확인할 수 있습니다(=Feature의 분포 확인)]



MLOps Component - Data Validation

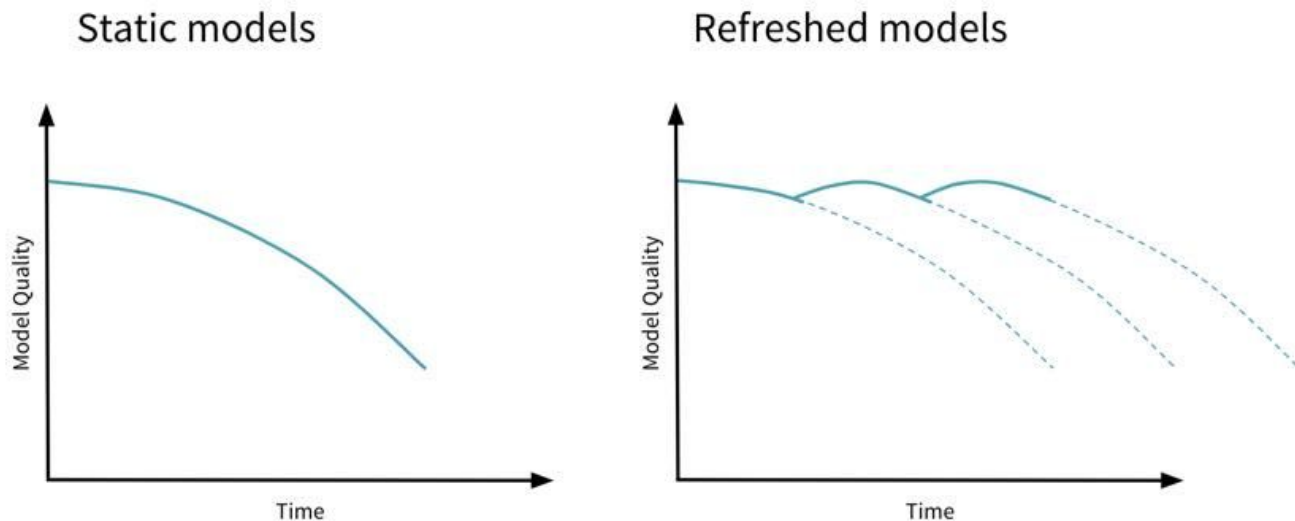
[재료들이 집에서 사용하던 것과 비슷한지 확인할 수 있습니다(=Feature의 분포 확인)]



MLOps Component - Data Validation

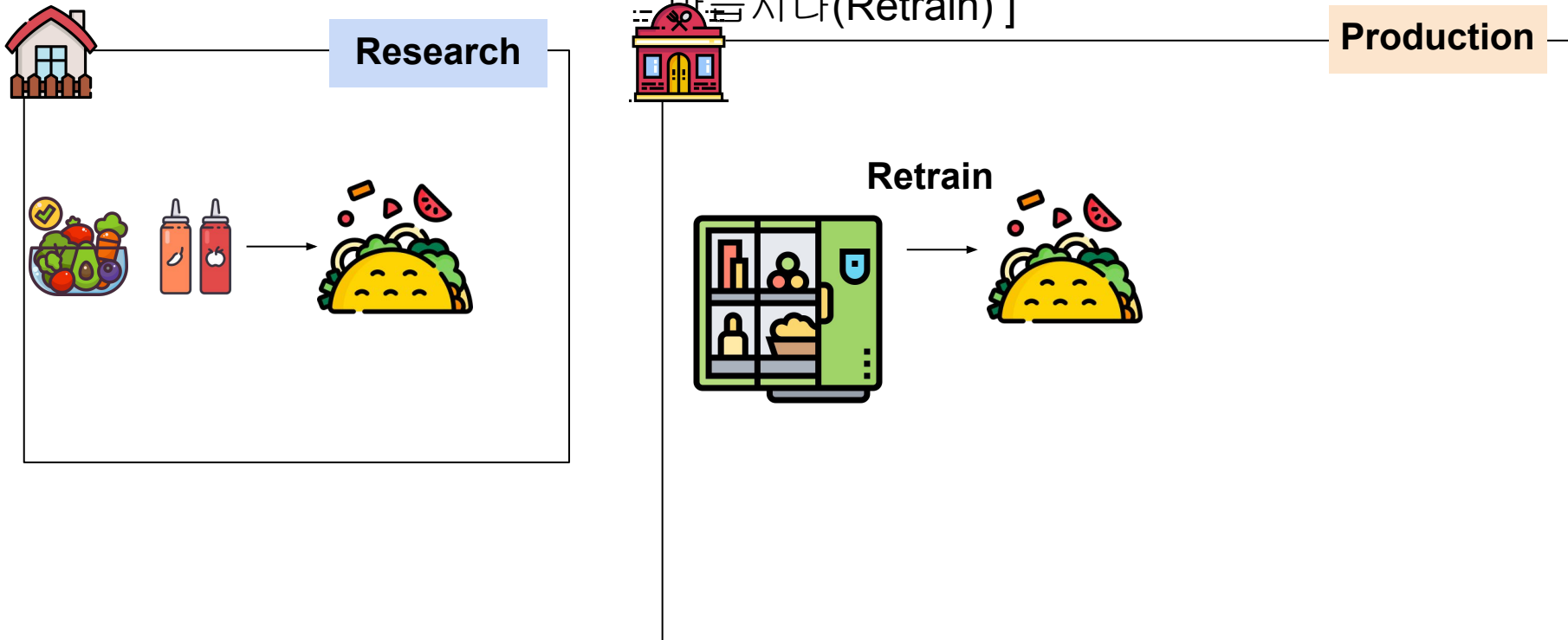
키워드 : Data Drift, Model Drift, Concept Drift

[How to Detect Model Drift in MLOps Monitoring](#)
[Productionizing Machine Learning: From Deployment to Drift Detection](#)



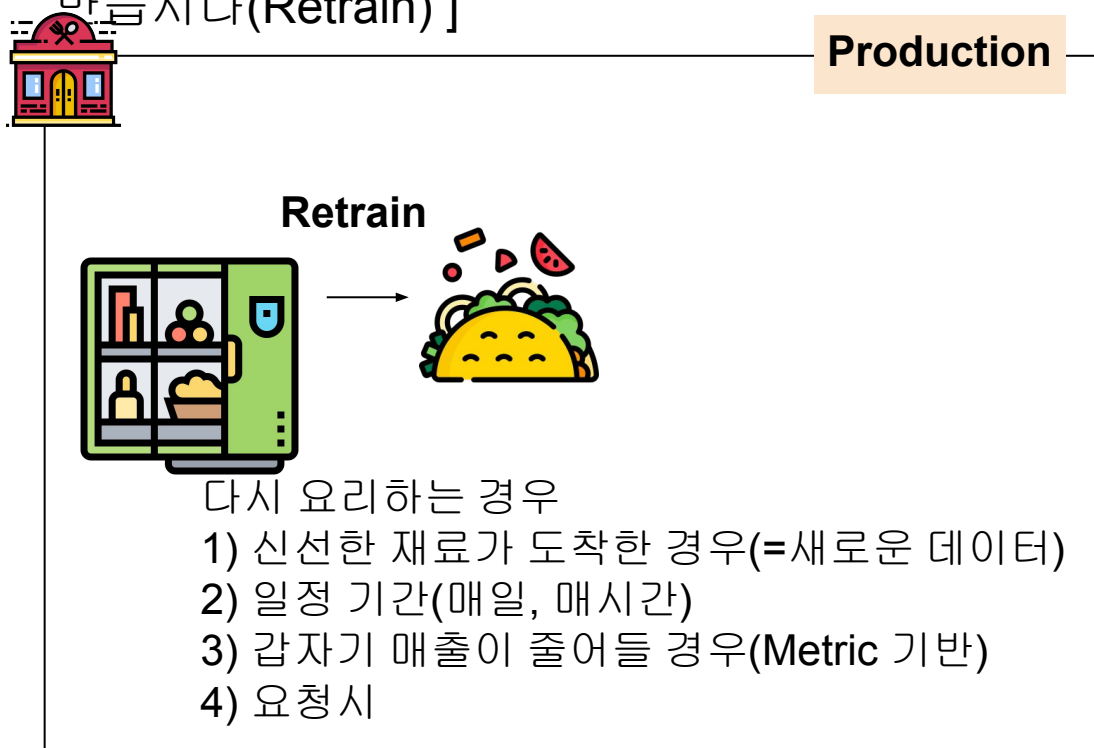
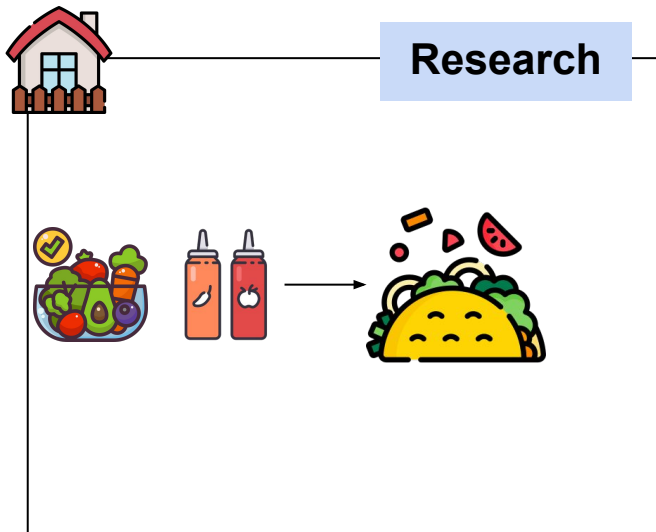
MLOps Component - Continuous Training

[만들었던 요리가 언제부터 고객분들이 좋아하지 않습니다. 신선한 재료로 다시
만들시다(Retrain)]



MLOps Component - Continuous Training

[만들었던 요리가 언제부터 고객분들이 좋아하지 않습니다. 신선한 재료로 다시 만듭시다(Retrain)]

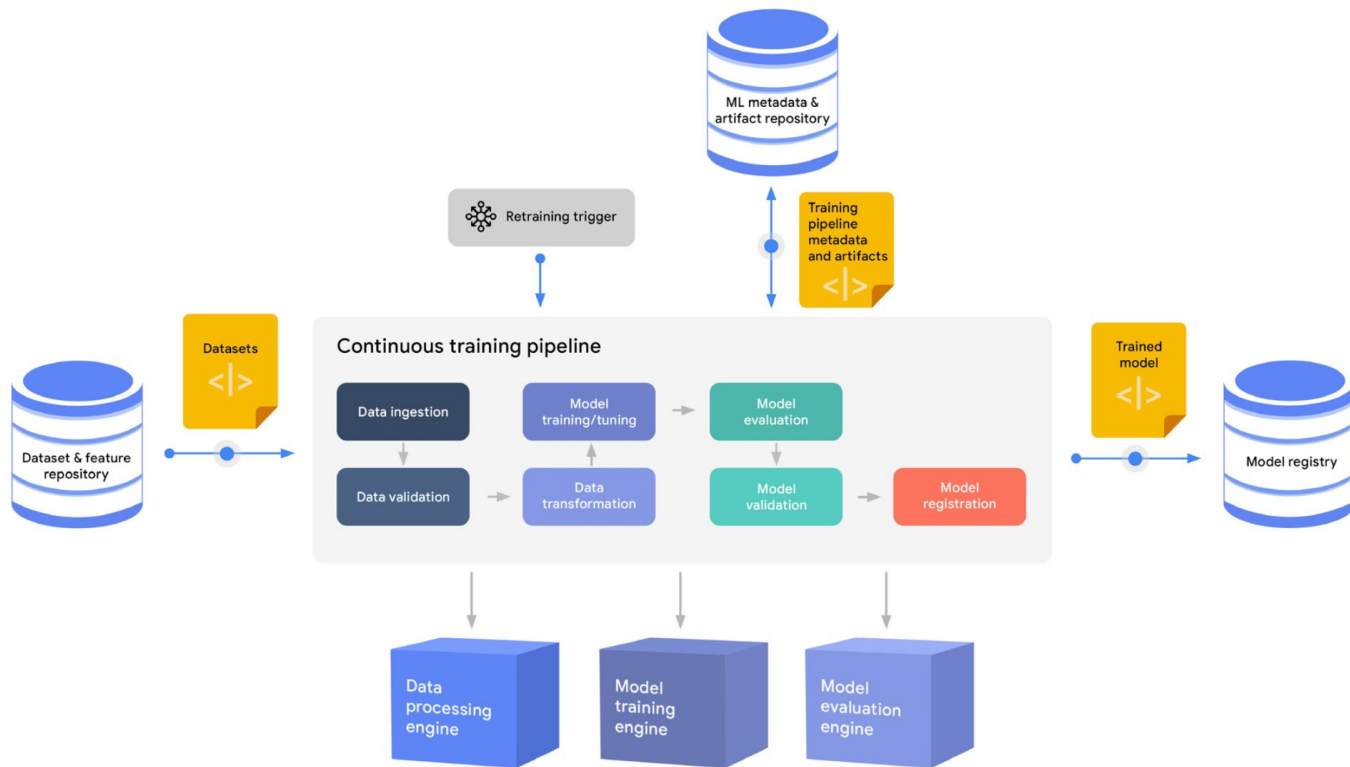


MLOps Component - Continuous Training

라이브러리가 있진 않고 CI/CD와 진행. CI, CD, CT 진행하게 됨

- 언제 다시 학습할 것인가?
 - 새로운 데이터가 들어온 경우
 - **Metric**이 떨어진 경우
 - 주기적으로
 - 코드가 변경되는 경우
 - 요청하는 경우(수동)

MLOps Component - Continuous Training



MLOps Component - Continuous Training

Deployment Action Rule 예시

([A Machine Learning Model Management System at Uber](#))

Listing 2: Action Rule Example

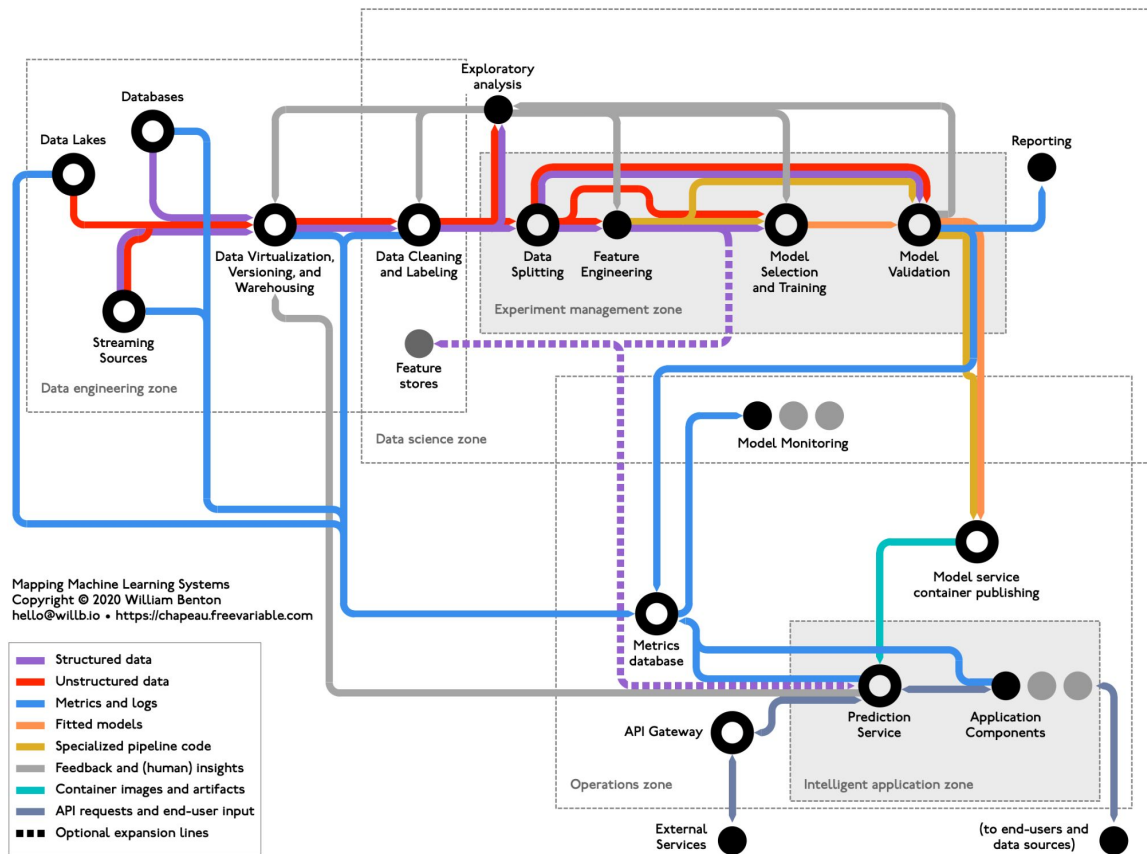
```
{
  "team": "forecasting",
  "uuid": "43057544-92b0-4421-a1b0-d7d87f77a",
  "rule": {
    "GIVEN": "model_domain == \"UberX\"
              AND model_name == \"Random Forest\",
    "WHEN": "metrics.bias <= 0.1
              AND metrics.bias > -0.1",
    "CALLBACK_ACTIONS": [
      {
        "action": "forecasting_deployment"
      }
    ]
  }
  "ENVIRONMENT": "production"
}
```

MLOps Component - 그 외

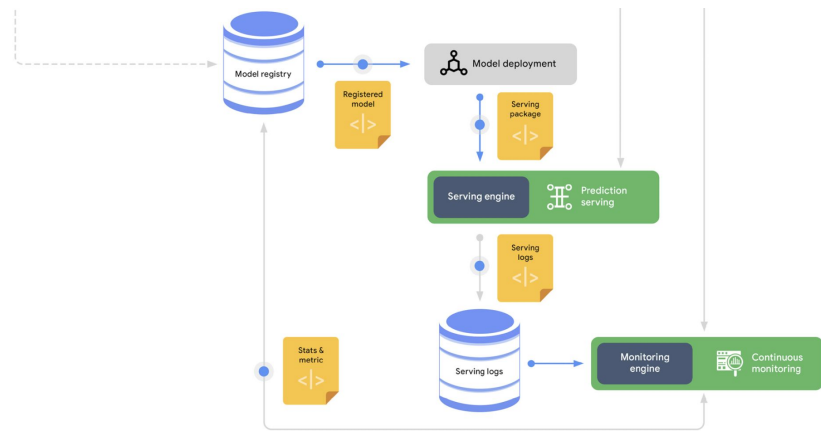
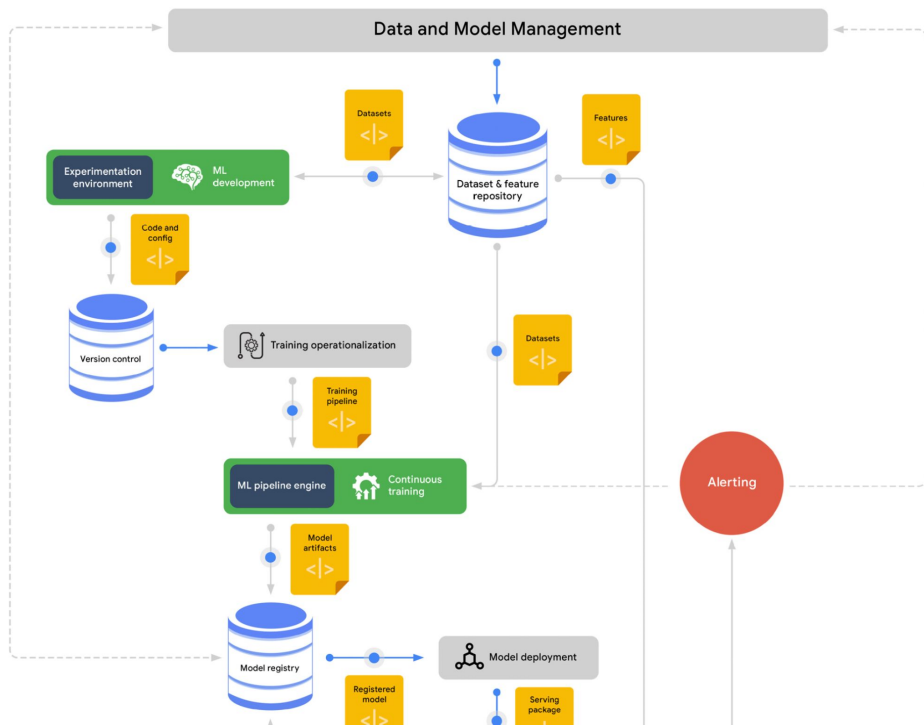
집, 레스토랑 관리(GPU Infra), 매출 기록(Monitoring),
자동화 시스템(AutoML), 음식 관리(Model Validation)



MLOps의 다양한 컴포넌트



MLOps의 다양한 컴포넌트



MLOps의 다양한 컴포넌트

Research / Production

Serving

Experiment, Model Management

Feature Store

Continuous Training

Model Analysis

Auto ML

MLOps Infra Management(GPU 등)

Monitoring

+@ 그 외에도 더 많지만 일단 이정도만..!

MLOps 공부하고 싶다면

공부 방법

관심 있는 분들의 상황

- 1) 회사에서 머신러닝 인프라를 구축해야 하는 경우
- 2) MLOps에 대해 공부하고 싶은 경우(학생 포함)

공부 방법

관심 있는 분들의 상황

- 1) 회사에서 머신러닝 인프라를 구축해야 하는 경우
- 2) MLOps에 대해 공부하고 싶은 경우(학생 포함)

모두 “머신러닝”의 기초는 알고 있어야 함

머신러닝에 대해 모르는 상황에서 MLOps를 구축하려면 괴리감이 생김(혹은 팀으로 구성)

기본적으로 “CI/CD”, “클라우드”, “쿠버네티스”를 잘 알고 있으면 좋음

두 경우 모두 MLOps 기본 강의를 먼저 수강하는 것을 추천!

MLOps 교육 Content - 강의

- 1) 앤드류 응님의 [Machine Learning Engineering for Porudction\(MLOps\) 특화 과정](#)
- 2) [Full Stack Deep Learning](#)
- 3) [\[애저든보잡\] MLOps 101](#)
- 4) 송호연님의 [머신러닝 엔지니어 실무\(인프런\)](#)

Spring 2021 Online Course

검색 > Data Science > Machine Learning

제공자: DeepLearning.AI

Machine Learning Engineering for Production (MLOps) 특화 과정

Become a Machine Learning expert. Productionize your machine learning knowledge and expand your production engineering capabilities.

★★★★★ 4.8 199개의 평가

Andrew Ng ±3명의 강사 [최고의 강사](#)

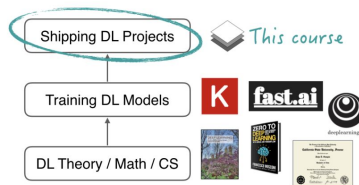
무료로 등록
년 6월 4일 일 개시 예정

재정 지원 가능

6,712명이 이미 등록했습니다.

About this course

There are many great courses to learn how to train deep neural networks. However, training the model is just one part of shipping a deep learning project. This course teaches full-stack production deep learning:



- Formulating the problem and estimating project cost
- Finding, cleaning, labeling, and augmenting data
- Picking the right framework and compute infrastructure
- Troubleshooting training and ensuring reproducibility
- Deploying the model at scale

MLOps 교육 Content - 강의

- 1) 앤드류 응님의 [Machine Learning Engineering for Porudction\(MLOps\)](#) 특화 과정
- 4개 강의로 구성. 4개 중 아직 2개 강의만 열림(21년 6월 초 기준)
- 기초적인 내용을 잘 다룸. 기본서 느낌, TFX 기반 강의

MLOps 교육 Content - 강의

- 1) 앤드류 응님의 [Machine Learning Engineering for Production\(MLOps\)](#) 특화 과정

- 4개 강의로 구성. 4개 중 아직 2개 강의만 열림(21년 6월 초 기준)
- 기초적인 내용을 잘 다룸. 기본서 느낌, TFX 기반 강의

- 2) [Full Stack Deep Learning](#)

- 13주차로 구성
- 엄청 실용적인 관점에서 문제 정의, 베이스라인 정하기, MLOps 각종 도구, 트러블 슈팅, AI Ethics, Test, Deploy에 대해 다룸
- 현존하는 MLOps 관련 강의 중 제일 추천

MLOps 교육 Content - 강의

- 1) 앤드류 응님의 [Machine Learning Engineering for Production\(MLOps\) 특화 과정](#)
 - 4개 강의로 구성. 4개 중 아직 2개 강의만 열림(21년 6월 초 기준)
 - 기초적인 내용을 잘 다룸. 기본서 느낌, TFX 기반 강의
- 2) [Full Stack Deep Learning](#)
 - 13주차로 구성
 - 엄청 실용적인 관점에서 문제 정의, 베이스라인 정하기, MLOps 각종 도구, 트러블 슈팅, AI Ethics, Test, Deploy에 대해 다룸
 - 현존하는 MLOps 관련 강의 중 제일 추천
- 3) [\[애저든보잡\] MLOps 101](#)
 - 한국어 강의
 - 분량은 1시간 이내로, 핵심적인 부분을 잘 알려줌 + Azure 기반 MLOps

MLOps 교육 Content - 강의

- 1) 앤드류 응님의 [Machine Learning Engineering for Production\(MLOps\) 특화 과정](#)
 - 4개 강의로 구성. 4개 중 아직 2개 강의만 열림(21년 6월 초 기준)
 - 기초적인 내용을 잘 다룸. 기본서 느낌, TFX 기반 강의
- 2) [Full Stack Deep Learning](#)
 - 13주차로 구성
 - 엄청 실용적인 관점에서 문제 정의, 베이스라인 정하기, MLOps 각종 도구, 트러블 슈팅, AI Ethics, Test, Deploy에 대해 다룸
 - 현존하는 MLOps 관련 강의 중 제일 추천
- 3) [\[애저든보잡\] MLOps 101](#)
 - 한국어 강의
 - 분량은 1시간 이내로, 핵심적인 부분을 잘 알려줌 + Azure 기반 MLOps
- 4) 송호연님의 [머신러닝 엔지니어 실무\(인프런\)](#)
 - 한국어 강의
 - 여러 오픈소스를 사용하는 방법

MLOps 교육 Content - 다양한 라이브러리, 클라우드 경험해보기

강의를 토대로 큰 그림을 이해한 후

각자의 문제 상황에 맞는 시스템 구축해보기

<https://github.com/EthicalML/awesome-production-machine-learning>

<https://github.com/visenger/awesome-mlops>

여기 나오는 각종 **Component**의 라이브러리를 사용해보고, 비교해보기

클라우드의 **MLOps** 서비스도 체험해보고, 사용성 확인 추천
(직접 시스템 구축보다 클라우드 서비스 사용이 쉬우니 이런 방식으로 사용되는구나!
파악해보기)

Azure MLOps 쪽이 굉장히 좋았음

MLOps 교육 Content - 다양한 라이브러리, 클라우드 경험해보기

<https://github.com/EthicalML/awesome-production-machine-learning>

Star : 9K

여기 있는 라이브러리로 MLOps 대부분 파악 가능

Awesome production machine learning

This repository contains a curated list of awesome open source libraries that will help you deploy, monitor, version, scale, and secure your production machine learning.

Quick links to sections in this page

 Explaining predictions & models	 Privacy preserving ML	 Model & data versioning
 Model Training Orchestration	 Model Serving and Monitoring	 Neural Architecture Search
 Reproducible Notebooks	 Visualisation frameworks	 Industry-strength NLP
 Data pipelines & ETL	 Data Labelling	 Metadata Management
 Functions as a service	 Computation distribution	 Model serialisation
 Optimized computation frameworks	 Data Stream Processing	 Outlier and Anomaly Detection
 Feature engineering	 Feature Stores	 Adversarial Robustness
 Commercial Platforms	 Data Storage Layer	

머신러닝 시스템 디자인 패턴

머신러닝 시스템 디자인 패턴을 보고 어떤 패턴들이 있는지 학습하기

https://mercari.github.io/ml-system-design-pattern/README_ko.html

Serving patterns

서빙 패턴은 실제 운영 환경에서 머신러닝 모델을 이용할 수 있도록 만드는

- Web single pattern
- Synchronous pattern
- Asynchronous pattern
- Batch pattern
- Prep-pred pattern
- Microservice vertical pattern
- Microservice horizontal pattern
- Prediction cache pattern
- Data cache pattern
- Prediction circuit break pattern
- Multiple stage prediction pattern
- Edge prediction pattern: To do
- Antipatterns
 - Online bigsize pattern
 - All-in-one pattern

QA patterns

예측 서버와 모델 평가하기 위한 패턴들입니다.

- Shadow AB-testing pattern
- Online AB-testing pattern
- Loading test pattern
- Antipatterns
 - Offline-only pattern

Training patterns

학습 파이프라인을 구성하기 위한 패턴들입니다.

- Batch training pattern
- Pipeline training pattern
- Parameter and architecture search pattern
- Antipatterns
 - Only-me pattern
 - Training code in serving pattern
 - Too many pipes pattern

Operation patterns

ML 학습 시스템의 설정과 로깅, 모니터링, 알람 시스템을 위한 관리 및 운영 패턴들입니다.

- Model-in-image pattern
- Model-load pattern
- Data model versioning pattern
- Prediction log pattern
- Prediction monitoring pattern
- Parameter-based serving pattern
- Condition-based-serving pattern
- Antipatterns
 - No logging pattern
 - Nobody knows pattern

Lifecycle patterns

실제 운영을 위한 ML 시스템 전체를 구성하기 위해 여러 패턴들을 조합한 패턴들입니다.

- Train-then-serve pattern
- Training-to-serving pattern

글로벌 회사들의 MLOps Platform

논문, 다양한 기업들의 MLOps Use Case 찾아보기

Google의 [TFX](#) (오픈소스)

Netflix의 [Metaflow](#) (오픈소스)

Uber의 [Michelangelo](#) (오픈소스 X)

Airbnb의 [Bighead](#) (오픈소스 X)

Lyft의 [Flyte](#) (오픈소스)

Doordash의 [ML Platform](#)

Facebook의 [FBLearner](#)

AWS의 SageMaker

GCP의 Vertex AI

Azure의 Machine Learning

논문

Uber MLOps 논문들

구글 클라우드의 [Practitioners Guide to MLOps](#)

Superb AI의 [실리콘밸리의 ML옵스](#)

추천 사이트

<https://ml-ops.org/> : 정말 다양한 MLOps에 대해 잘 정리된 사이트

<https://ml-ops.org/content/references.html> : 자료 정말 잘 정리되어 있음

Motivation for MLOps



You will learn for what to use Machine Learning, about various scenarios of change that need to be managed and the iterative nature of ML-based software development. Finally, we provide the MLOps definition and show the evolution of MLOps.

[Read more](#)

Designing ML-powered Software



This part is devoted to one of the most important phase in any software project — understanding the business problem and requirements. As these equally apply to ML-based software you need to make sure to have a good understanding before setting out designing things.

[Read more](#)

End-to-End ML Workflow Lifecycle



In this section, we provide a high-level overview of a typical workflow for machine learning-based software development.

[Read more](#)

Three Levels of ML-based Software



You will learn about three core elements of ML-based software — Data, ML models, and Code. In particular, we will talk about

- Data Engineering Pipelines
- ML Pipelines and ML workflows.
- Model Serving Patterns and Deployment Strategies

[Read more](#)

MLOps Principles



In this part, we describe principles and established practices to test, deploy, manage, and monitor ML models in production.

[Read more](#)

State of MLOps (Tools & Frameworks)



This part presents an overview of software tools and frameworks that manage ML artifacts, and cover the whole machine learning cycle.

[Read more](#)

MLOps Books

- 1 ["Machine Learning Engineering" by Andriy Burkov, 2020](#)
- 2 ["ML Ops: Operationalizing Data Science" by David Sweeney, Steven Hillion, Dan Rope, Dev Kannabiran, Thomas Hill, Michael O'Connell](#)
- 3 ["Building Machine Learning Powered Applications" by Emmanuel Ameisen](#)
- 4 ["Building Machine Learning Pipelines" by Hannes Hapke, Catherine Nelson, 2020, O'Reilly](#)
- 5 ["Managing Data Science" by Kirill Dubovikov](#)
- 6 ["Accelerated DevOps with AI, ML & RPA: Non-Programmer's Guide to AIOPS & MLOPS" by Stephen Fleming](#)
- 7 ["Evaluating Machine Learning Models" by Alice Zheng](#)
- 8 [Agile AI. 2020. By Carlo Appugliese, Paco Nathan, William S. Roberts, O'Reilly Media, Inc.](#)
- 9 ["Machine Learning Logistics". 2017. By T. Dunning et al. O'Reilly Media Inc.](#)
- 10 ["Machine Learning Design Patterns" by Valliappa Lakshmanan, Sara Robinson, Michael Munn, O'Reilly 2020](#)
- 11 ["Serving Machine Learning Models: A Guide to Architecture, Stream Processing Engines, and Frameworks" by Boris Lublinsky, O'Reilly Media, Inc. 2017](#)
- 12 ["Kubeflow for Machine Learning" by Holden Karau, Trevor Grant, Ilan Filonenko, Richard Liu, Boris Lublinsky](#)

궁금한 것이 있으시면 커뮤니티에서 이야기해요!

카카오톡 오픈 채팅방

MLOps KR 페이스북 커뮤니티에서 공식적으로 운영하는 것은 아니지만

카카오톡 오픈 채팅방에 MLOps라는 방이 있어요 :)

페이스북 MLOps KR

커뮤니티에 스터디 글 올리셔서 스터디 꾸려보시면 어떨까요?

채용

아직 채용이 엄청 많은 공고가 보이진 않고, 머신러닝 엔지니어/데이터 엔지니어가
진행하는 경우도 다수 존재

향후 2-3년 내로 더 많이 생길 것으로 예상

MLOps 엔지니어, 머신러닝 인프라, 머신러닝 플랫폼 개발자 등으로 채용 공고가 보임

The screenshot shows a job search interface with filters for '태그 막 맞는 기업찾기', '지역 한국', '경력 전체', and '최신순'. It displays five job listings:

- MLOps 엔지니어** (네오사피언스(Neosapience))
서울 · 한국
채용보상금 1,000,000원
- MLOps Engineer** (엑스브레인(XBRAIN))
서울 · 한국
채용보상금 1,000,000원
- MLOps Engineer** (마키나락스)
[응답률 매우 높음]
서울 · 한국
채용보상금 1,000,000원
- DataOps(MLOps) 엔지니어** (브이엔티지)
서울 · 한국
채용보상금 1,000,000원
- MLOps Engineer** (카카오모빌리티(KakaoMobility))
경기 · 한국
채용보상금 1,000,000원
- DevOps/MLOps엔지니어** (에이아이트릭스(AITRICS))
[응답률 매우 높음]
서울 · 한국
채용보상금 1,000,000원

마치며

MLOps는 ML + DevOps

프러덕션 환경으로 가기 위한 여러 시스템

절대 진리가 있는 것은 아니고, 현재 요구 상황에 따라 다르게 정리해보기!

참고 자료로 하나씩 같이 공부해요 :)

궁금하신 내용은

email : snugyun01@gmail.com

instagram : @data.scientist

연락주세요 :)

그러니 모두에게 이렇게 말하고 싶다

많이 힘들었죠?

이제까지 정말
수고하셨습니다

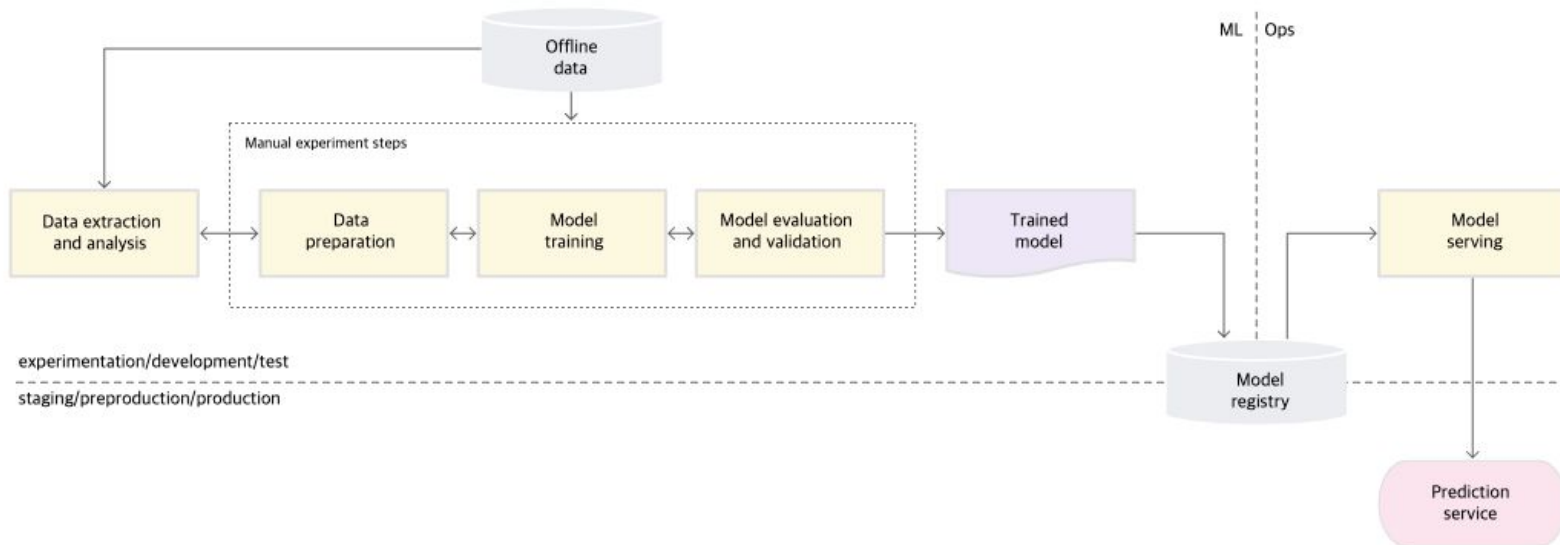


포기않고 달려온 여러분은 멋진 사람입니다

발표에서 다루진 않았지만, 참고 자료

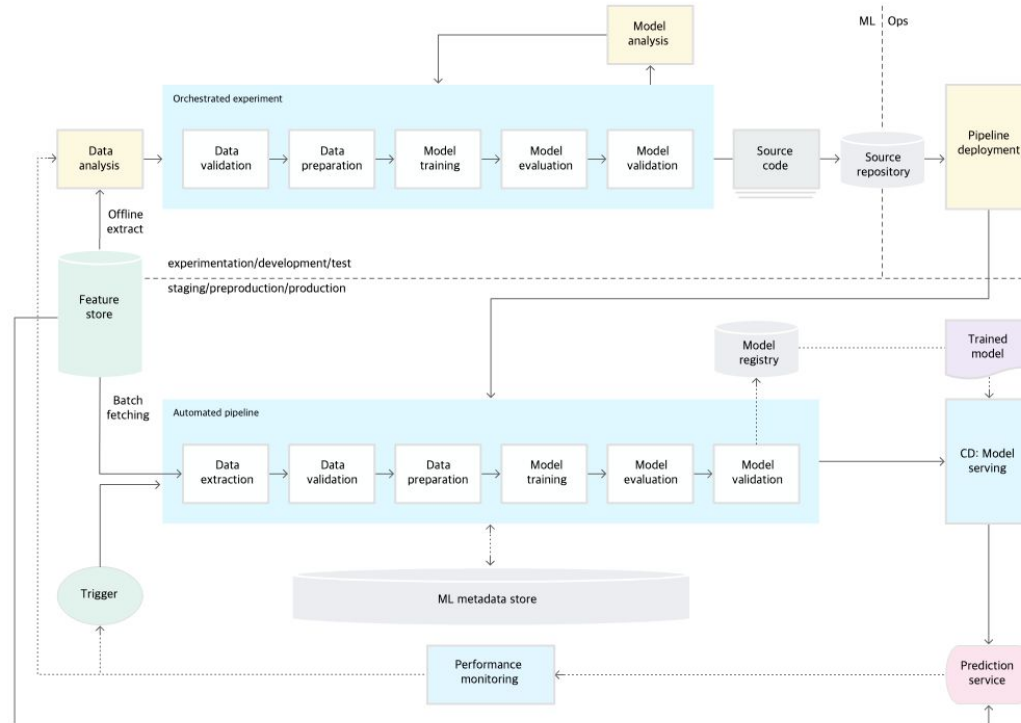
MLOps Level 0 : Manual Process

MLOps: Continuous delivery and automation pipelines in machine learning



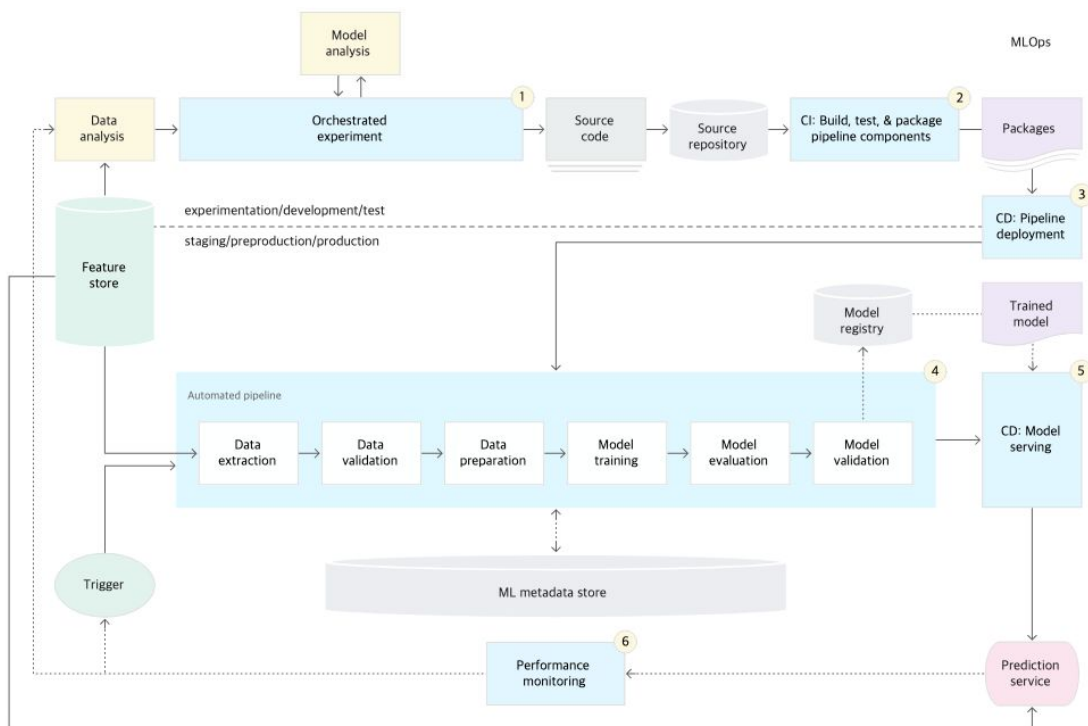
MLOps Level 1 : ML pipeline automation

MLOps: Continuous delivery and automation pipelines in machine learning



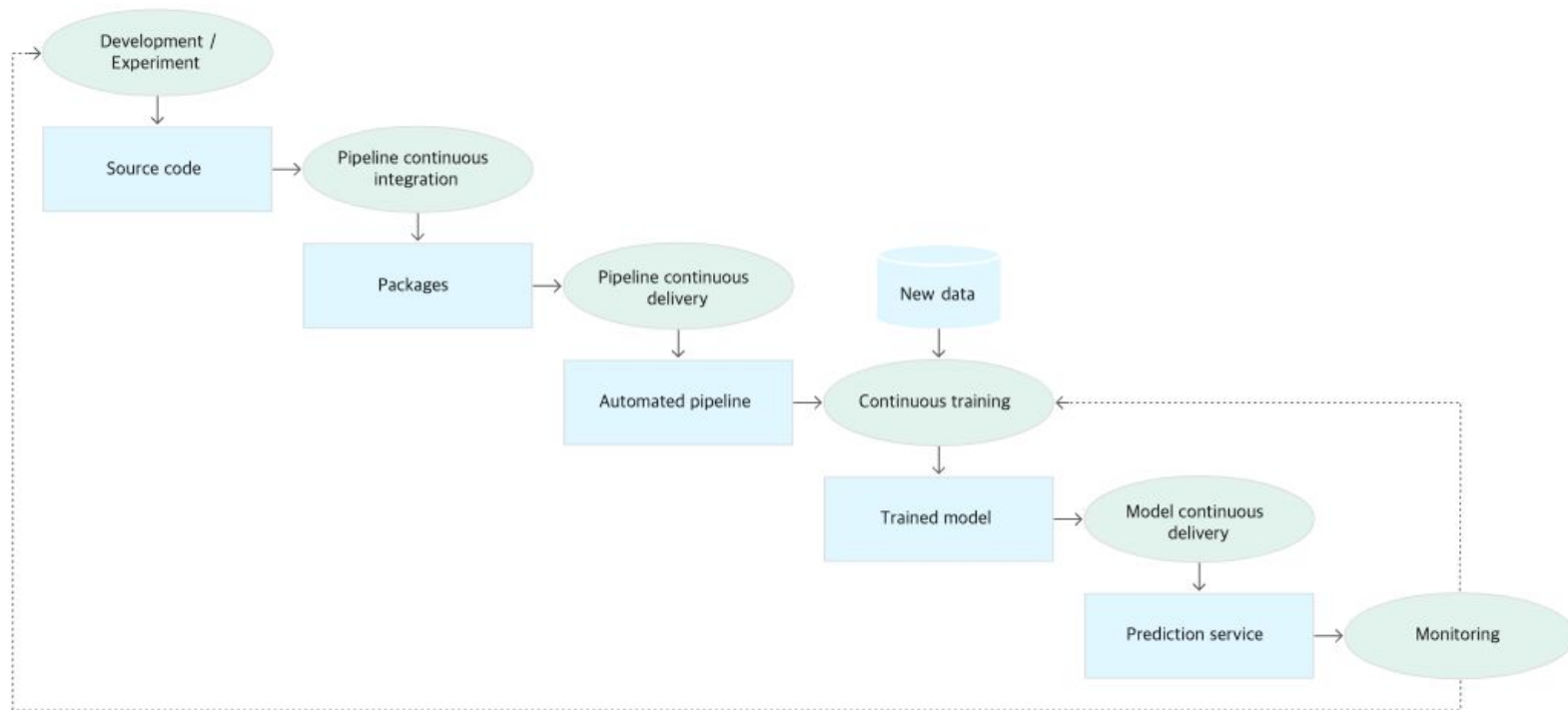
MLOps Level 2 : CI/CD pipeline automation

MLOps: Continuous delivery and automation pipelines in machine learning



MLOps Level 2 : CI/CD pipeline automation

MLOps: Continuous delivery and automation pipelines in machine learning



MLOps Component - Model Analysis

TFX - TFMA

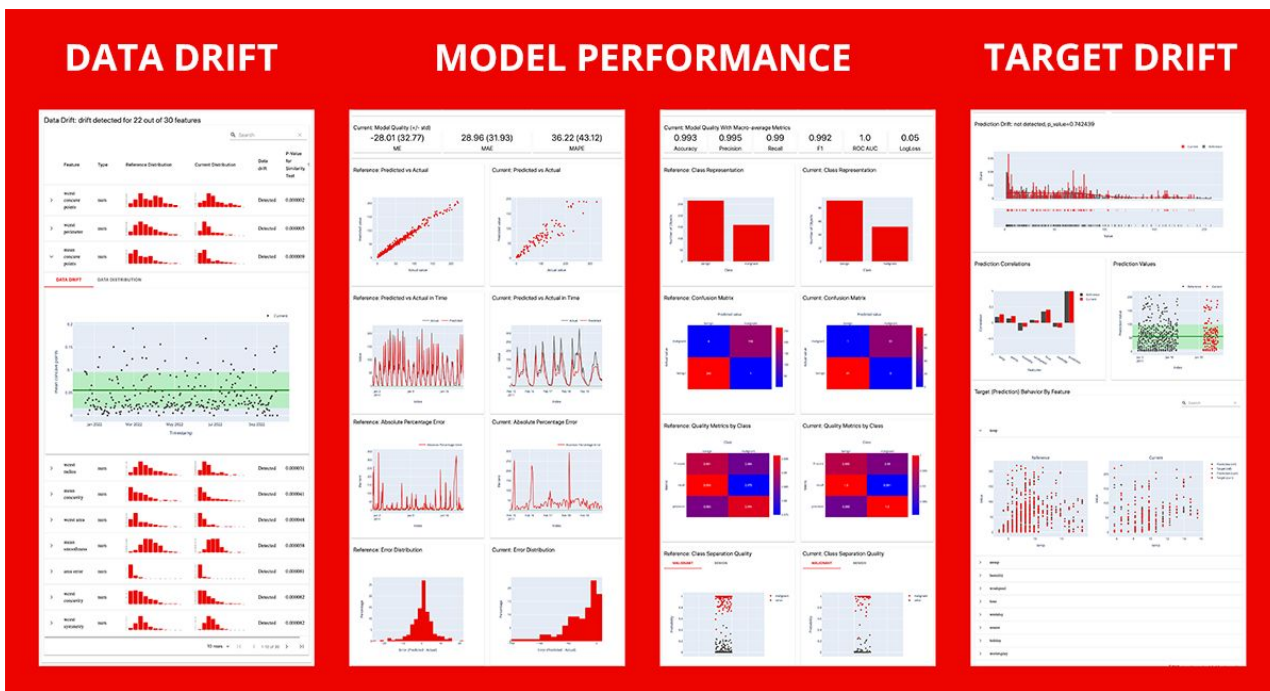
<https://github.com/tensorflow/model-analysis>



MLOps Component - Model Analysis

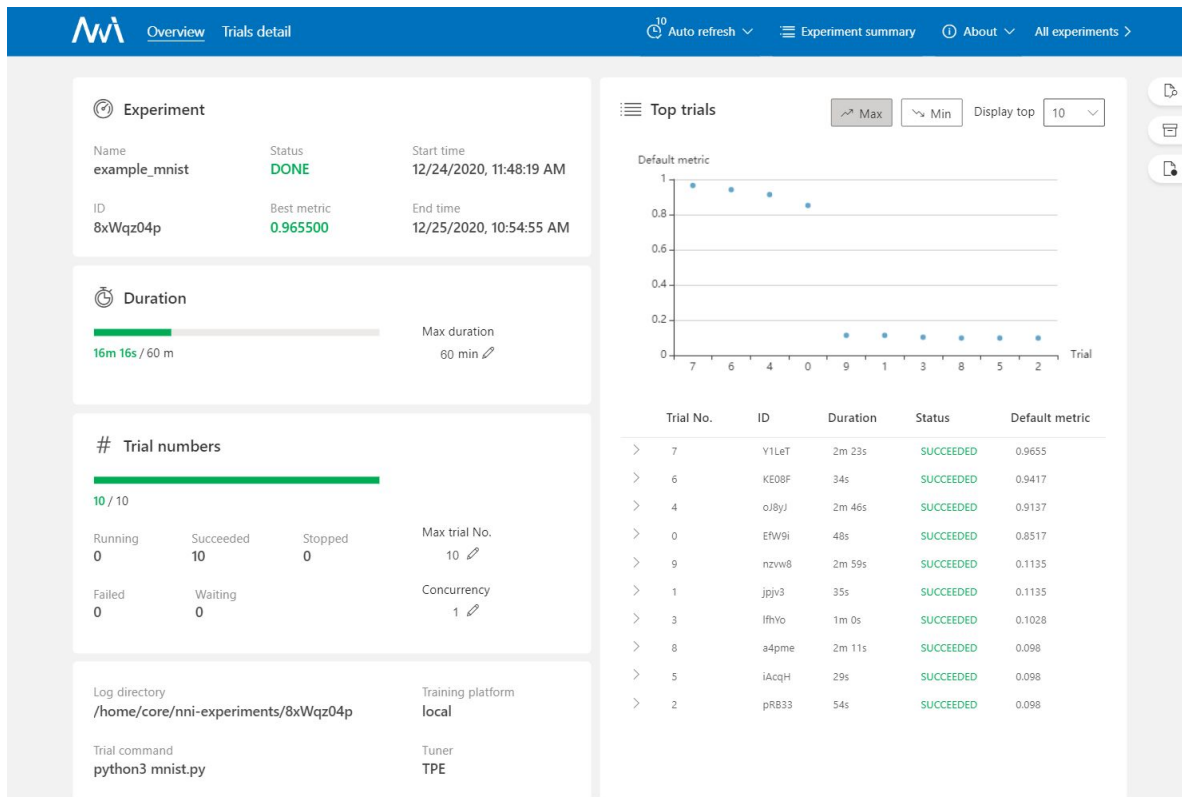
Evidently

<https://github.com/evidentlyai/evidently>



MLOps Component - AutoML

Microsoft NNI



Machine Learning Tools

Machine Learning tools & platforms landscape - v.1.0 January 2021

Presented by  MLReef

DATA MANAGEMENT

Data Exploration & Management

COHESITY, rubrik, allegroai, ALLUXIO, Amundsen, druid, hudi, MLReef, databricks, ALGORITHMIA, SPARK, APARANT, ATSCALE, BRIM, CAZENA, CLOUDBERA, kaggle, datagrok, ClearOS, DATERA, dremio, elasticsearch, erwin, Excelero, FLOU, ELIANT, HANDBOOK, HYCU, Imple, komprise, MANTA, MONDRIAN, ORACLE, PARQUET, PLOMO, presto, tony, VSEARCH, VEXATO, VIKAS

Data Labelling

appen, DataRobot, IMERIT, prodigy, Labelbox, SUPERVISELY, Playment, scole, snoriel, HIVE

Data Streaming

ALLUXIO, hudi, kafka, CONFLUENT, VALHAI, GoogleCloud, strim

Data Version Control

databricks, Data Hub, DVC, FLOU, MLReef, pachyderm, WATSON, allegroai

Data Generation

scole, scrapinghub, DATPROF

Data Privacy

airlock, Celantur, mostly, Pyth, TUNULT

Data Quality Checks

arize, gret_expectations, Navegro

MODELLING

Notebook / Code Management

colab, databricks, FLOU, kaggle, MLReef,Weights & Biases, attyrex

Data Processing & Visualization

attlyrex, colab, DASK, databricks, dotData, Flyte, gluon, iguazio, incho, mflow, MLReef, MODIN, Navegro, Pachyderm, presto, Prometheus, S3AS, snoriel, SOLFlow, Stream, tuzi, SVBOX, VALHAI, allegroai,Weights & Biases

Feature Engineering

delData, FEAST, Featuretools, Pachyderm, TACTON, S3AS, tuzi, iguazio

Model Training

attlyrex, iguazio, colab, databricks, dotData, FLOU, Flyte, kaggle, MLReef, NAVEG, mflow, Percept Labs, snoriel, tuzi, VALHAI, S3AS, AnyScale, Pachyderm, GEM

Experiment Tracking

iguazio, allegroai, comet, DataRobot, datmo, DODING, FLOU, mflow, MLReef, polyaxon, S3AS, SPELL, VALHAI, GoogleCloud

Model (Hyperparameter) Optimization

attlyrex, Angel, comet, DataRobot, polyaxon, SIGOPT, SPELL, TUNO, DPTOR

Auto ML

DataRobot, Determined AI, dotData, GoogleCloud, KUB, TransmogriAI, iguazio

Model Management

attlyrex, allegroai, databricks, dotData, FLOU, LUON, MLReef, mflow, Percept Labs, polyaxon, S3AS, VALHAI, VERT

Model Evaluation

arize, fiddler, InterpretML, Model Explainability, fiddler, InterpretML, Percept Labs, Shop

Frameworks & major libraries

Chainer, Keras, PyTorch, spaCy, TensorFlow, XGBoost, ONNX, CNTK, theano, matplotlib, julia, torch

CONTINUOUS DEPLOYMENT

Data Flow Management

ALLUXIO, SPARK, HADOOP, kafka, Data Hub, dotData, HYCU, PERFECT

Feature Transformation

FEAST, Featuretools, TACTON, iguazio, Data Hub

Monitoring

ALGORITHMIA, arize, DataRobot, DODING, iguazio, Lossy, snoriel, Unravel, VALHAI, VERT, attyrex

Model Compliance & Audit

ALGORITHMIA, S3AS, KUB

Model Deployment & Serving

AIBLE, ALGORITHMIA, allegroai, cortex, datahub, datmo, DODING, dotData, FLOU, FRITZ, iguazio, Kubeflow, mflow, NAVEG, PERCEPT, S3AS, SELLDO, SPELL, SHARPE, VALHAI, VERT, KUB, GoogleCloud, MLReef, attyrex

Model Validation

arize, datahub, fiddler, LUCID, MLReef, S3AS, snoriel

Model Compatibility

IMMOR, ONNX, gladML

COMPUTING MANAGEMENT

Computing & Data Infrastructure

ORACLE, CLOUDBERA, HADOOP, KUB, LINODE, MANTA, PLOMO, VEXATO

Environment Management

CONDA, databricks, datmo, MAHOUT, MLReef, allegroai

Resource Allocation

ALGORITHMIA, GoogleCloud, MLReef, databricks, Data Hub, Determined AI, FLOU, polyaxon, SPELL, VALHAI, allegroai

SCALING

argo, datahub, datmo, DODING, SELLDO, TUNO, VALHAI

SECURITY & PRIVACY

ALGORITHMIA, DODING, mflow, Pyth