MLOps Tour @ SNU

A Practical Guide to MLOps

ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY
서 울 대 학 교 공 과 대 학

Author

박기성 @ VESSL AI

Date

04.05.2024

VESSL

Agenda

1. What is MLOps?
2. How to Start MLOps: *From Tracking to Pipeline*
   - Data Prep & Feature Engineering
   - Experiment Tracking
   - Model Registry
   - Cluster & Job Management
   - Model Serving
   - Pipeline Automation
   - Other MLOps Components
3. MLOps: The VESSL Way
4. SNU GPU First Program with VESSL AI
5. Hands-on Tutorial

VESSL

Speaker & Lecturer

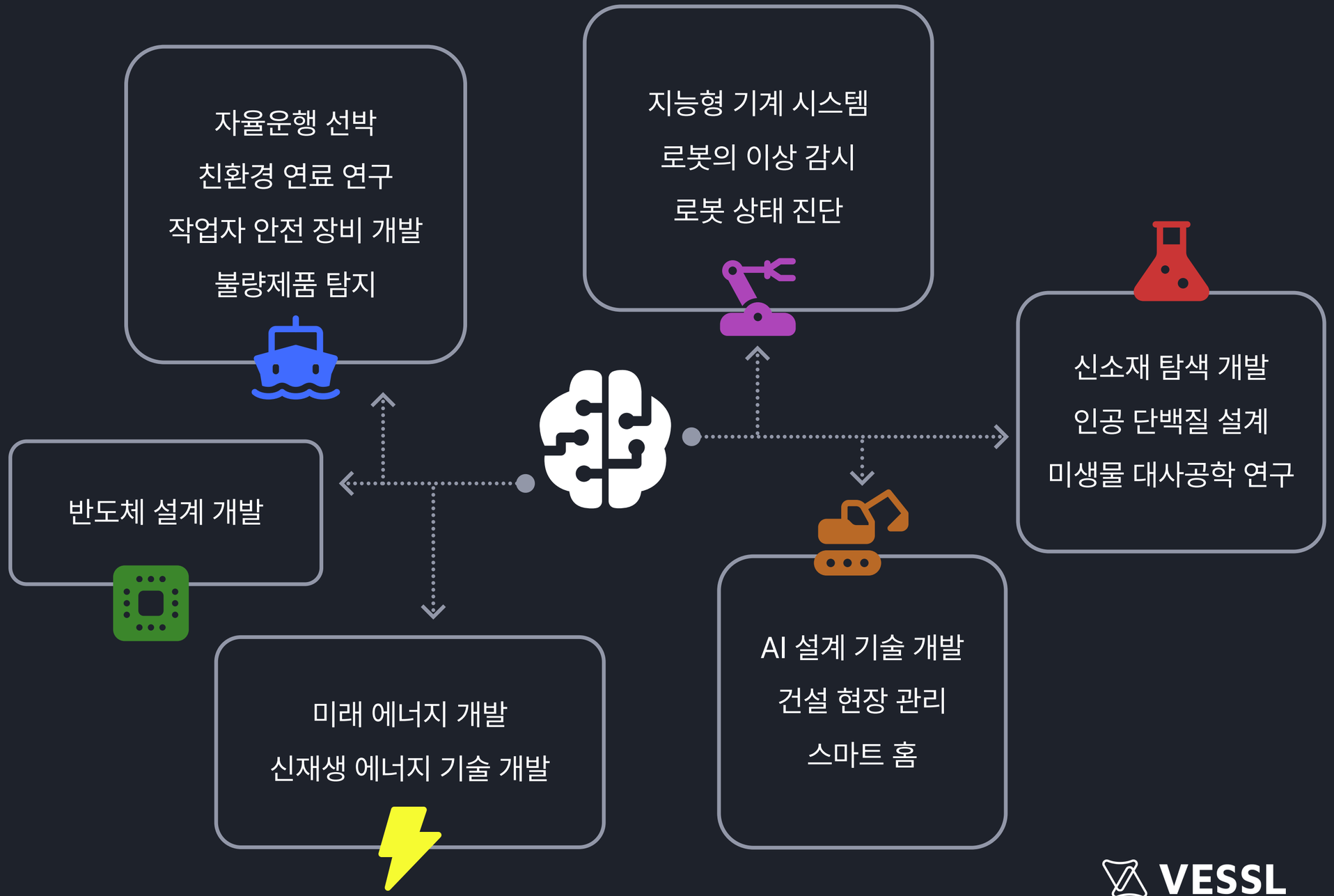
박기성 (Mika) Engineering Manager

VESSL AI

- Nexon Korea, Woowa Bros, Krafton, Sendbird Korea
- <야생의 땅: 듀랑고>의 대규모 분산 처리 게임플레이 서버 프로그래밍
- Sendbird Voice & Video Tech Lead

VESSL

AI is Everywhere



VESSL

ML in Real Industry

VESSL

ML in Real Industry

자율 주행 & 비전 인식

머신 비전 및 바코드 리딩

생성 AI 에이전트

길찾기 & 검색 및 광고 개인 추천

패션 추천, 카테고리 분류

기업별 AI 에이전트 & 챗봇

다국어 LLM 추천 및 가격 결정

금융 예측 모델

AI & ML 모델 연구

신재생 에너지 연구

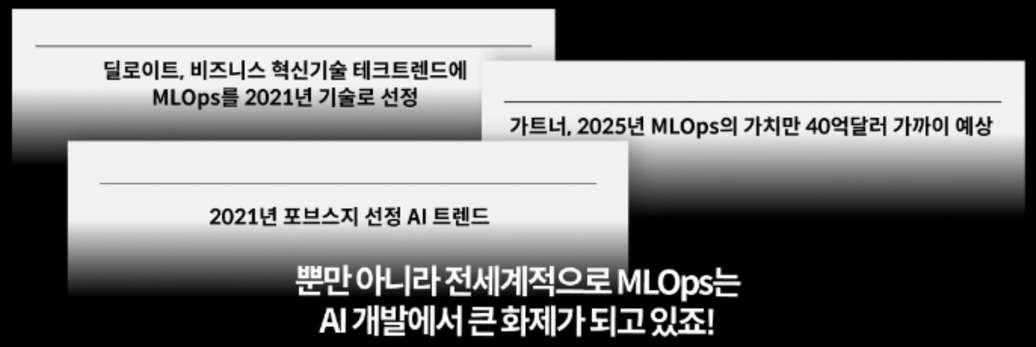
VESSL

AI 4대천왕

Andrew Ng

> By ensuring consistently high-quality data through all stages of a project, MLOps can make deployment and development of ML systems more systemic and reliable.

체계적인 데이터 관리를 위해 MLOps 가 중요하다.



We estimate that as much as 90 percent of the failures in ML development come not from developing poor models but from poor productization practices and the challenges of integrating the model with production data and business applications, which keep the model from scaling and performing as intended...

Lamarre, E., Smaje, K., & Zemmel, R. (2024). MLOps so AI can scale. McKinsey & Company.

VESSL

What is MLOps?

VESSL

What is MLOps?

MLOps = Machine Learning + DevOps

머신러닝 모델을 개발 및 배포, 운영하는 일련의 과정

Automated, Reproducible, Testable, Evolvable 한 ML Workflow를 만드는 것이 목표

VESSL

In [ ]:

```python
# Train function (with vessl logging)
from torch.autograd import Variable

def train(model, device, train_loader, optimizer
    model.train()
    loss = 0
    for batch_idx, (data, label) in enumerate(tr
        data, label = data.to(device), label.to(
        optimizer.zero_grad()
        output = model(data)
        loss = F.nll_loss(output, label)
        loss.backward()
        optimizer.step()
        if batch_idx % 128 == 0:
            print('Train Epoch: {} [{}/{} ({:.0f
                epoch + 1, batch_idx * len(data)
                100. * batch_idx / len(train_loa

    # Logging loss metrics to Vessl
    vessl.log(
        step=epoch + start_epoch + 1,
        payload={'loss': loss.item()}
    )
```

개 같은 고양이?

VESSL

트레이닝할 **데이터**는 어디서 구해서 어떻게 쌓아둬야 하지?

OO가 공유한 모델 코드는 왜 내 컴퓨터에선 **재현**이 안 되는거야?

GPU **머신** 내가 오늘 쓰기로 했는데 OO가 쓰고 있네...

코드가 돌긴 하는데 학습이 잘 되는지는 어떻게 **모니터링**해?

왜 매번 실험할때마다 결과가 달라? **데이터셋 버전**이 매번 다르네?

트레이닝이 끝나고 남는 아웃풋 중에 뭐를 **모델**로 남겨야 하지?

만든 모델이 잘 학습되었는지는 어떻게 **검증**해야 하지?

모델을 우리 **서비스에 배포**하고 싶은데 어떻게 하지?

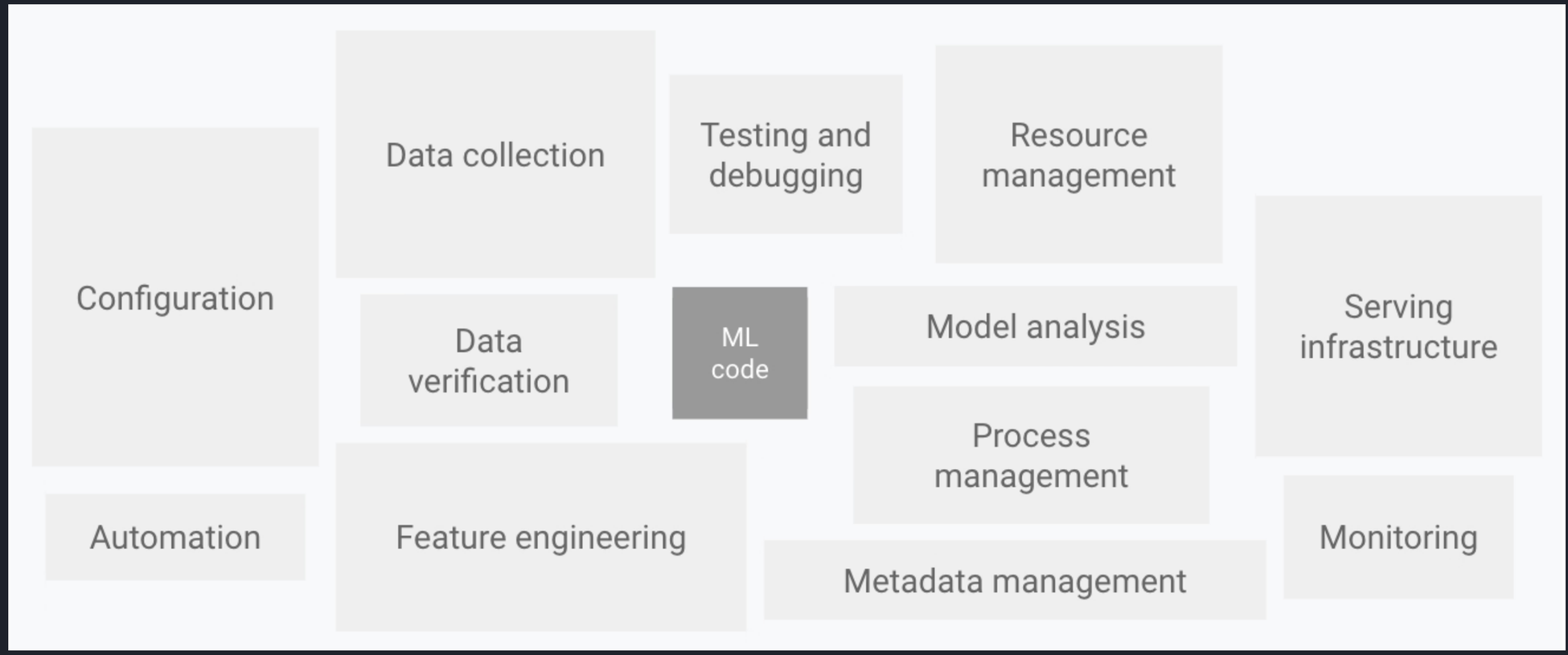
배포한 모델이 제대로 일하고 있는지 어떻게 **측정하고 모니터링**해?

이걸 매번 손으로 해야해? 코드만 푸시하면 실행되게 **자동화**할 수 없나?

...

VESSL

ML Code is a Small Part of Whole Process



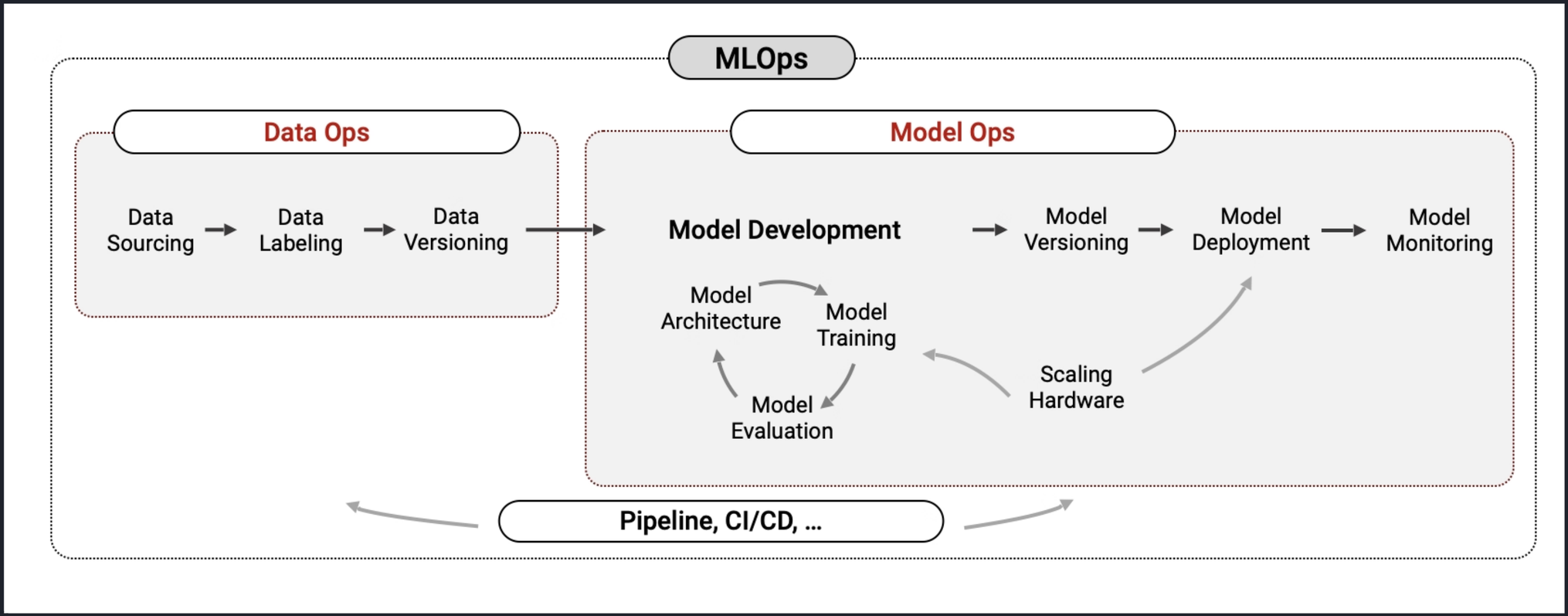
Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., & Young, M. (2015). **Hidden Technical Debt in Machine Learning Systems**. In *Proceedings of the Neural Information Processing Systems (NeurIPS)*.

Google에서 2015년에 NeurIPS에 발표, MLOps 개념의 시초가 되는 논문

VESSL

What is MLOps?

MLOps = DataOps + ModelOps



VESSL

How to Start MLOps?

VESSL

MLOps Landscape

- End-to-end Machine Learning Operations (MLOps) platforms
- Experiment tracking, model metadata storage and management
- Dataset labeling and annotation
- Data storage and versioning
- Data quality monitoring and management
- Feature stores
- Model hubs
- Model quality testing
- Workflow orchestration and pipelining tools
- Model deployment and serving
- Model observability
- Responsible AI
- Compute and infrastructure
- GPU Cloud Servers && Serverless GPUs
- Vector databases and data retrieval
- Foundation model training frameworks

...

VESSL

Core Features in MLOps

1. **DataOps - Dataset Management, Feature Store**
- 데이터를 사용하기 좋게 적절하게 라벨링/가공/피쳐 추출 및 적재
2. **Experiment Tracking**
- 머신러닝 학습/실험 결과를 재현하고 팀원들과 공유할 수 있도록 관리
3. **Cluster Management & Job Orchestration**
- 로컬 머신 대신 클러스터에서 머신러닝 실험 등의 job을 실행하고 관리할 수 있도록 지원
4. **Model Registry**
- 학습 결과물 중 의미있는 결과를 낸 것을 나중에 따로 사용할 수 있도록 버전별로 적재
5. **Model Serving**
- 만들어진 모델이 실제 일(inference)을 할 수 있도록 API/batch job 등의 형태로 배포
6. **ML Workflow Pipeline**
- 위의 과정들을 특정 주기마다, 혹은 데이터/코드/지표가 업데이트 될때마다 실행되도록 자동화

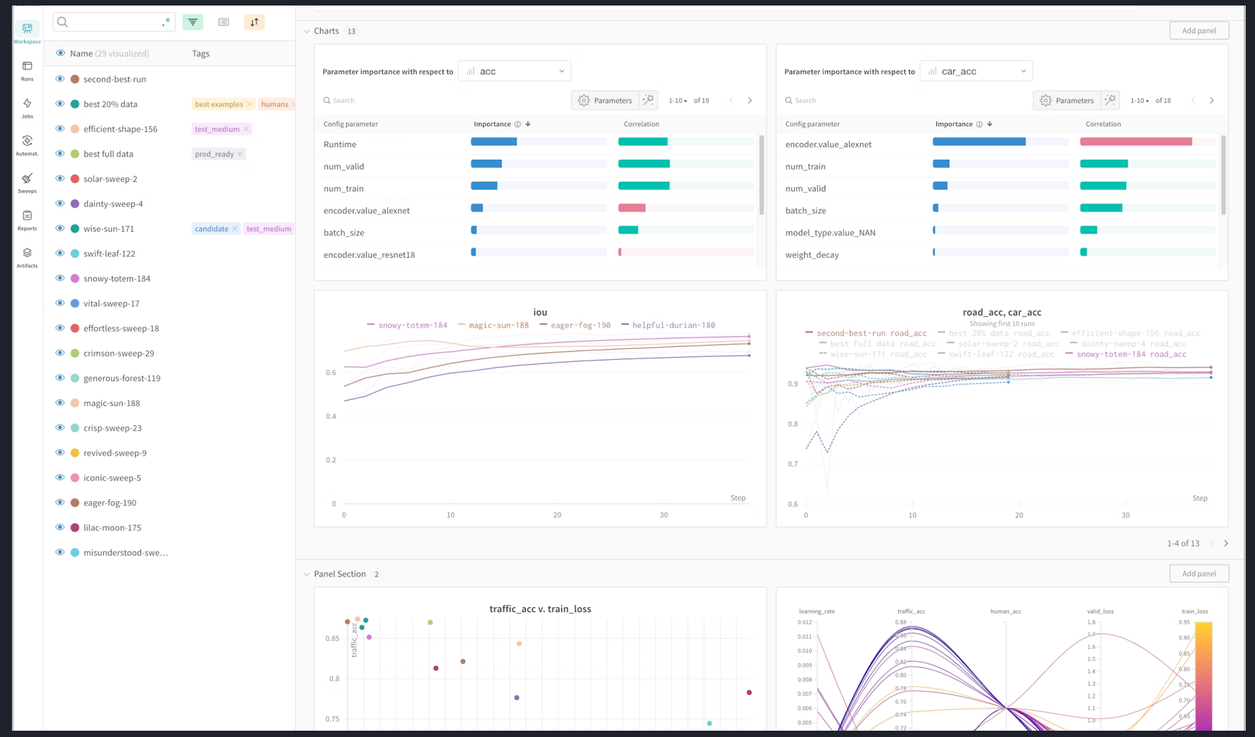
VESSL

Core Features in MLOps

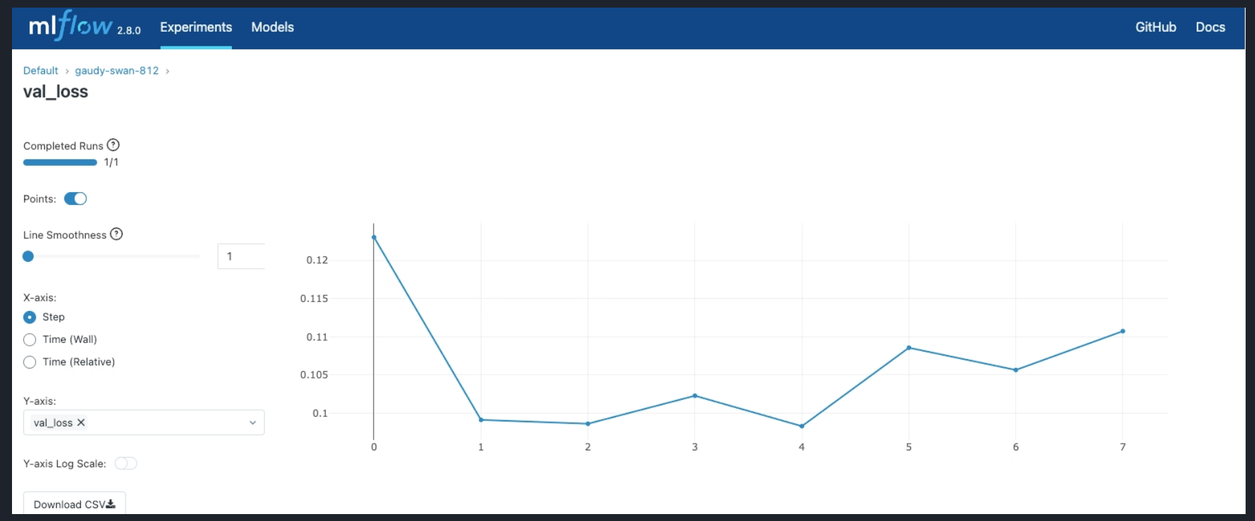
1. **DataOps - Dataset Management, Feature Store**
   - 데이터를 사용하기 좋게 적절하게 라벨링/가공/피쳐 추출 및 적재
2. **Experiment Tracking**
   - 머신러닝 학습/실험 결과를 재현하고 팀원들과 공유할 수 있도록 관리
3. **Cluster Management & Job Orchestration**
   - 로컬 머신 대신 클러스터에서 머신러닝 실험 등의 job을 실행하고 관리할 수 있도록 지원
4. **Model Registry**
   - 학습 결과물 중 의미있는 결과를 낸 것을 나중에 따로 사용할 수 있도록 버전별로 적재
5. **Model Serving**
   - 만들어진 모델이 실제 일(inference)을 할 수 있도록 API/batch job 등의 형태로 배포
6. **ML Workflow Pipeline**
   - 위의 과정들을 특정 주기마다, 혹은 데이터/코드/지표가 업데이트 될때마다 실행되도록 자동화

VESSL

- Weights & Biases Tracking Dashboard 예시



- MLFlow Tracking 예시



Experiment Tracking

- **머신러닝 실험의 모든 기록과 결과물을 저장하고 관리하는 저장소**
  - e.g) Parameter Configurations, Evaluation Metrics, Logs, Example Predictions, Performance Visualizations, ...
- 동일한 모델을 재현할 수 있도록 모든 정보를 저장
- 이후, development iteration, deployment, automation 등 에 활용

다음의 서비스들에서 사용 가능

- VESSL Experiment / Run Tracking
- tensorboard
- Weights & Biases
- MLflow

VESSL

Experiment Tracking

```
import wandb
wandb.init(project='mnist-project')
wandb.log({"loss": loss})
```

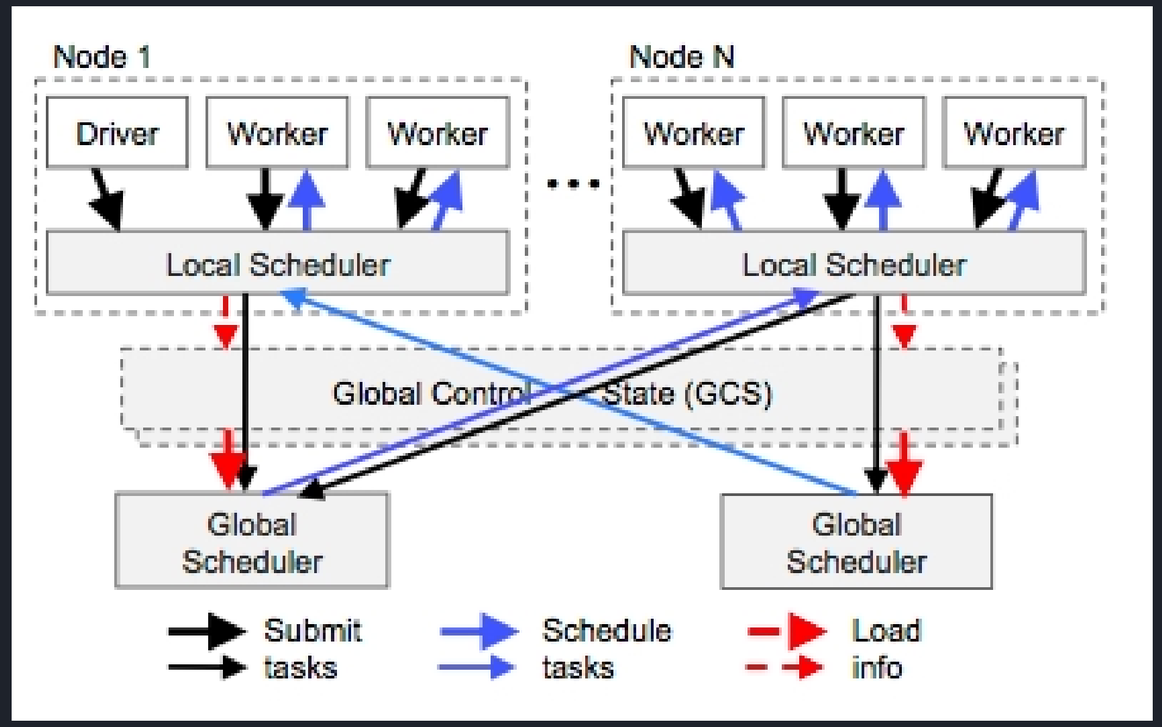
wandb 예시

Tips

- 내가 저장한 정보 만으로 **Model Reproduce**가 가능한지 확인
  - Dataset이 수시로 바뀌는 경우, Experiment가 실행될때의 Dataset 상태를 저장하지 않으면 reproduce에 실패할 수 있음

- 이후에 쉽게 찾아볼 수 있도록 **중앙화된 저장소**에 **구조화**하여 저장
  - 처음 프로젝트 세팅시, 어떤 metric들을 모니터링할것인지 미리 정하고, 해당 metric들을 logging하도록 개발 단계에서 추가
  - local에서 실험할때도, local에서 직접 tensorboard 등을 실행하여 작업하는 것 보다, 중앙화된 저장소를 이용하는 것이 좋다

Cluster Management & Job Orchestration



머신러닝 실험을 돌리기 위한 클러스터와 클러스터에서 돌아가는 Training Job들을 관리하고 운영하기 위한 기술

- Training이 진행되는 환경 (Image), 리소스, Configuration, Command 등을 명시적으로 작성해서 관리
- Container 기반 Runtime을 실행할 수 있는 환경으로 주로 구축함
- OpenAI, Google, 네이버 등의 거대 AI 조직에서는 인하우스로 운영 (ref)

다음의 서비스들에서 사용 가능

- VESSL Cluster Management
- Ray
- Kubeflow Job Scheduling
- Databricks

VESSL

KubeFlow TFJob 명세의 일부

```yaml
apiVersion: kubeflow.org/v1
kind: TFJob
metadata:
  generateName: tfjob
  namespace: your-user-namespace
spec:
  tfReplicaSpecs:
    PS:
      replicas: 1
      restartPolicy: OnFailure
      template:
        metadata:
          annotations:
            sidecar.istio.io/inject: "false"
        spec:
          containers:
            - name: tensorflow
              image: gcr.io/your-project/your-image
              command:
                - python
                - -m
                - trainer.task
                - --batch_size=32
                - --training_steps=1000
    Worker:
      replicas: 3
```

Training Job

- Training이 진행되는 환경 (Image), 리소스, Configuration, Command 등을 명시적으로 작성
- Docker 기반의 환경을 주로 사용
- 이를 기반으로 Hyperparameter Searching등 Training 작업을 scale-out 하거나, 자동화된 사용 - CI/CD, testing 등에서 활용

다음 서비스에서 사용 가능

- KubeFlow - Pipeline
- MLFlow - Projects
- VESSL - Run / Experiments

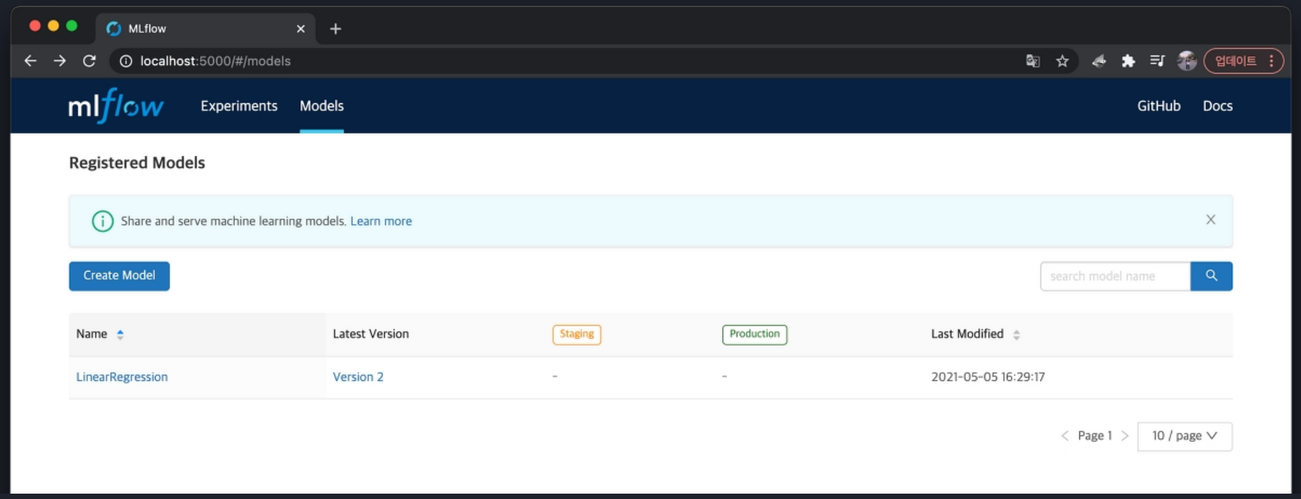
VESSL

Training Job

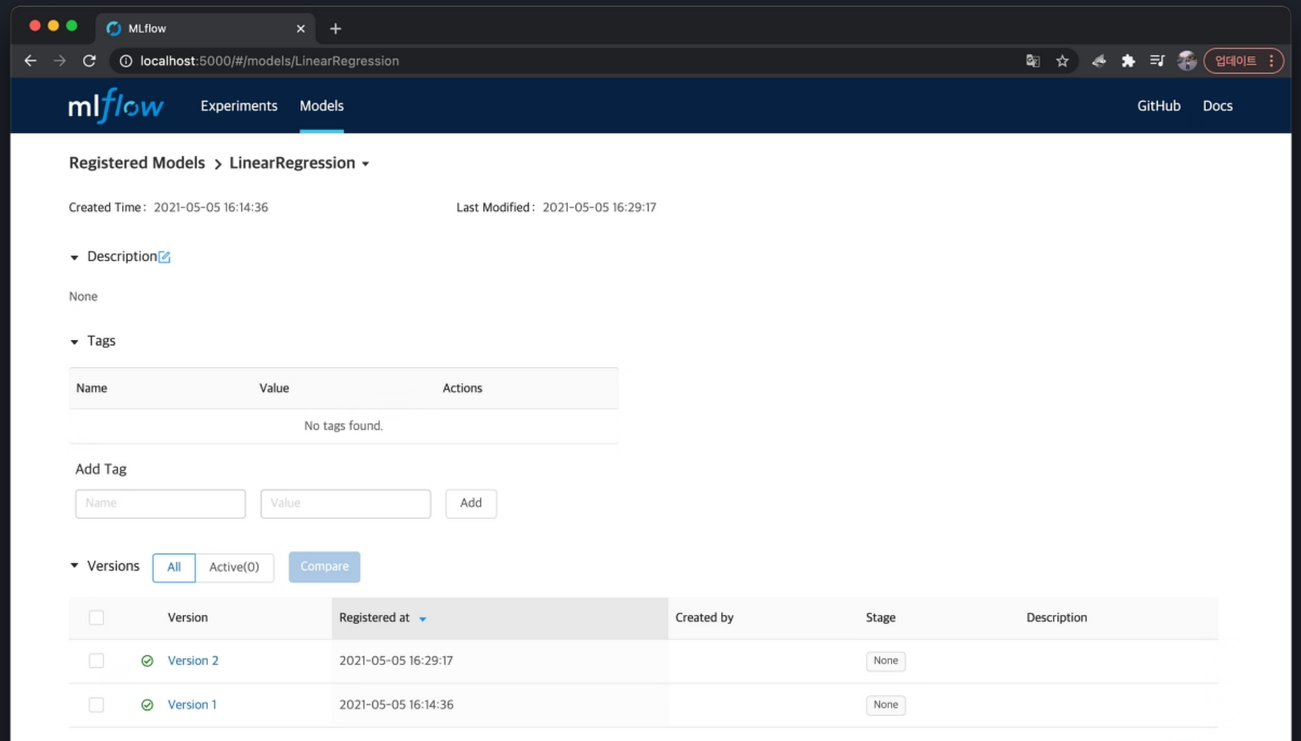
Tips

- Local에 구축한 환경을 Docker Image 의 형태 혹은 Shell Script 형태로 재구성
- Data Mount 가 필요하거나 한 경우, 다운로드, 혹은 Direct Mount 중 퍼포먼스의 이득이 큰 부분을 선택
  - 이를 위해 Cloud NFS Storage 사용, NAS 스토리지 구축 등 인프라 단 설계가 필요할 수 있음
- 스크립트를 만들어 수백개의 Training을 한번에 만들거나, 자동화된 작업에 활용 가능
  - Cluster 에 남은 자원을 모니터링 해야 하거나, 가장 효율적인 (저렴한) 인프라 설계가 필요할 수 있음
- GitHub Action/ Hook 과 연동하여 CI/CD 구축이 가능

MLFlow Model Registry 리스트



MLFlow Model Registry 상세 내용



Model Registry

- Experiment가 완료된 이후, 만들어진 모델을 과정과 결과와 함께 중앙화된 저장소에 저장
- 이후 Serving이나, 재학습, Fine-tuning등 을 활용할때 Registry에 있는 모델을 기준으로 수행
- API 연동을 통해 자동화된 Pipeline 구현

다음의 서비스들에서 사용 가능

- MLFlow - Registry
- VESSL - Model Registry
- neptune.ai - Model Registry

VESSL

Model Registry

```
mlflow.register_model(
    "runs:/d16076a3ec534311817565e6527539c0/sklearn-model",
    "sk-learn-random-forest-reg"
)
```

MLFlow 예시

Tips

- Model Registry에 저장할 파일들을 잘 정의해야 함
  - SavedModel 파일뿐 아니라, Serving 혹은 Fine-tuning때 활용 가능한 통계 혹은 config 값도 같이 저장
  - Registry에 저장된 파일로 Serving 혹은 Fine-tuning을 진행할 수 있는지 확인
- 이후에 쉽게 찾아볼 수 있도록 **중앙화된 저장소**에 **구조화**하여 저장
- Model Registry에 파일을 직접 업로드 하기 보다, Training Job (Experiment) 와 연계하여 사용
  - 저장된 Model이 어떤 과정을 통해 제작되었는지 쉽게 Tracking 하기 위해

BentoML - 1. Save Models

```python
import bentoml

from sklearn import svm
from sklearn import datasets

# Load predefined training set to build an example model
iris = datasets.load_iris()
X, y = iris.data, iris.target

# Model Training
clf = svm.SVC(gamma='scale')
clf.fit(X, y)

# Call to bentoml.<FRAMEWORK>.save(<MODEL_NAME>, model)
# In order to save to BentoML's standard format in a local model store
bentoml.sklearn.save("iris_clf", clf)

# [08:34:16 AM] INFO     Successfully saved Model(tag="iris_clf:svcryrt5xgafweb5",
#                        path="/home/user/bentoml/models/iris_clf/svcryrt5xgafweb5/")
# Tag(name='iris_clf', version='svcryrt5xgafweb5')
```

BentoML - 2. Define Services

```python
# service.py
import numpy as np
import bentoml
from bentoml.io import NumpyNdarray

# Load the runner for the latest ScikitLearn model we just saved
iris_clf_runner = bentoml.sklearn.load_runner("iris_clf:latest")

# Create the iris_classifier service with the ScikitLearn runner
# Multiple runners may be specified if needed in the runners array
# When packaged as a bento, the runners here will included
svc = bentoml.Service("iris_classifier", runners=[iris_clf_runner])

# Create API function with pre- and post- processing logic with your new "svc" annotation
@svc.api(input=NumpyNdarray(), output=NumpyNdarray())
def classify(input_series: np.ndarray) -> np.ndarray:
    # Define pre-processing logic
    result = iris_clf_runner.run(input_series)
    # Define post-processing logic
    return result
```

Model Serving

- 만들어진 모델을 API 서비스화하는 것
- 직접 Inference Code를 구현할 수도 있으나, 많이 쓰이는 ML Framework의 모델을 자동으로 API 서비스화 해주는 라이브러리들이 존재

다음 서비스에서 사용 가능

- FastAPI 등을 활용하여 직접 API 서버 만들기
- BentoML
- Seldon Core
- NVIDIA Triton Inference Server
- VESSL Serve

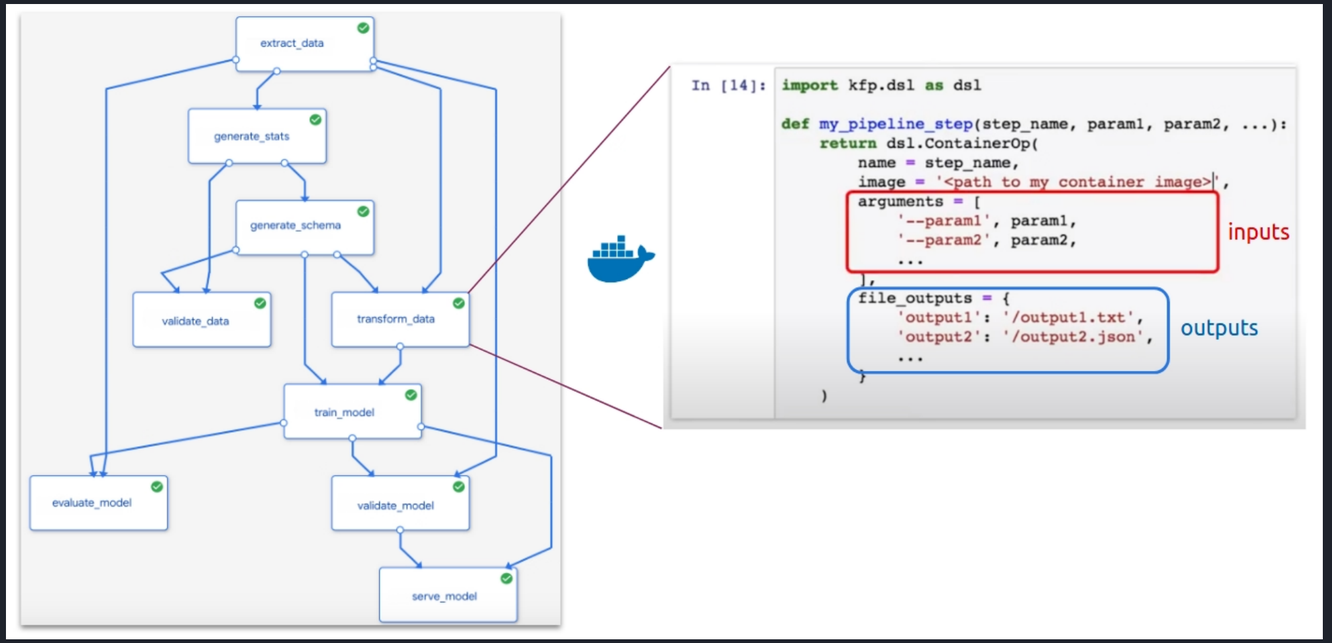
VESSL

Serving

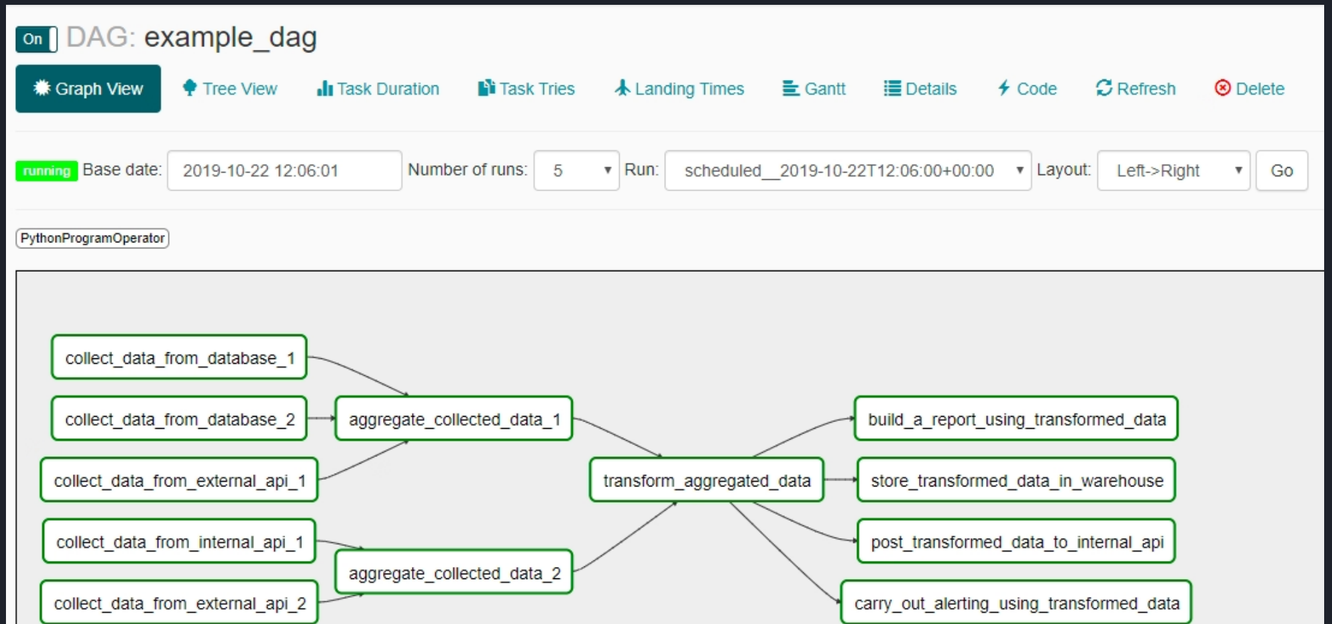
Tips

- Monitoring할 지표를 사전에 잘 정의하여 수집할 수 있도록 구현
- ML Model은 필연적으로 운영하면서 모델의 퍼포먼스가 떨어지므로,
  - ML Model의 결과로 Data의 Distribution이 바뀔수 있는 경우, 유의하여 개발이 필요
- Inference 전후로 필요한 작업 (Data preprocessing, Prediction mapping 등) 을 Serving API에서 수행할지, 혹은 Backend Code에서 수행할지 정의 필요
  - 필요에 따라 각 작업을 수행하는 Microservice가 따로 필요할 수 있음

KubeFlow Pipeline 예시



AirFlow 예시



ML Workflow Pipeline

- 위에서 다루었던 Training Job, Serving 등을 포함하여 개발 및 배포 과정을 정의하는 작업
- Pipeline을 정의하여 Schedule 혹은 Webhook 등을 이용하여 전체 ML 개발 과정을 한번에 실행 가능
  - e.g.) 매주 월요일에 자동으로 새로운 모델 Training
  - e.g.) 새로운 데이터가 업데이트 되었을때 자동으로 새로운 모델 Training 등

다음 서비스로 구축 가능

- KubeFlow Pipeline
- Airflow
- flyte
- VESSL - Pipeline

VESSL

Pipeline

Tips

- KubeFlow Pipeline의 경우 난이도가 있고, AirFlow로 먼저 간단하게 시작해 보는 것이 좋음.
  - KubeFlow - Kubernetes 위에서 ML workflow를 정의하기 위해 최근에 발표됨. 아직 안정화가 안되어 있음.
  - AirFlow - 데이터 분석 작업의 Pipeline을 만들기 위해 사용되던 Tool, ML에도 활용 가능, Python 기반으로 쉽게 시작 가능
- 인프라 설계부터 전 과정이 자동화 되어야 하기 때문에, Training, Serving, Monitoring 등 각각의 과정을 최대한 자동화

- 모든 과정을 하나의 Pipeline으로 만드려 하지 말고, 각각의 프로세스 별로 Pipeline을 만들고 lean하게 업데이트 하는 것이 좋다
  - 각 Pipeline별로 Input과 Output을 먼저 명확하게 정의, 어떤 output을 monitoring하여 pipeline을 업데이트할지 설정
  - 각 Pipeline의 목적과 실행되는 시점이 다르므로, 인프라 설계가 그에 맞춰 준비되어 있는지 확인

Other MLOps Components

MODEL MONITORING

- 머신러닝 모델이 배포되고 난 이후, 모델의 퍼포먼스나 inference 결과를 모니터링
- 저장된 결과를 바탕으로 이후 모델 개발에 반영

[Where to Start] Neptune AI, Evidently AI, VESSL Serving

DATASET VERSIONING

- Dataset이 계속 변화하는 환경일때, 파일들을 시점에 따라 그대로 복원할 수 있도록 저장
- Git-LFS 기반의 version control 시스템을 활용

[Where to Start] DVC, VESSL Dataset Versioning

HYPERPARAMETER OPTIMIZATION

- AutoML의 일종으로, 머신러닝 모델에서 필요한 하이퍼파라미터를 알고리즘이 자동으로 찾아주는 기술

[Where to Start] Weights & Biases Sweep, Katib, Ray Tune, VESSL Sweep

VESSL

Other MLOps Components

GPU CLOUD SERVERS & SERVERLESS GPU

- GPU Shortage로 인한 공급 부족을 해결하기 위한 서비스
- 빠르게 고성능 GPU를 할당받고, per-invocation / per-uptime 만큼만 과금

[Where to Start] RunPod, HuggingFace Serverless, Replicate

VECTOR DATABASES AND DATA RETRIEVAL

- 대규모 이미지, 텍스트, 센서 데이터 등을 벡터 형태로 저장하고 인덱싱 후 검색할 수 있는 서비스
- KNN, ANN(Approximate KNN) 인덱싱을 사용하여 구현

[Where to Start] Pinecone, Qdrant, Milvus

VESSL

KubeFlow

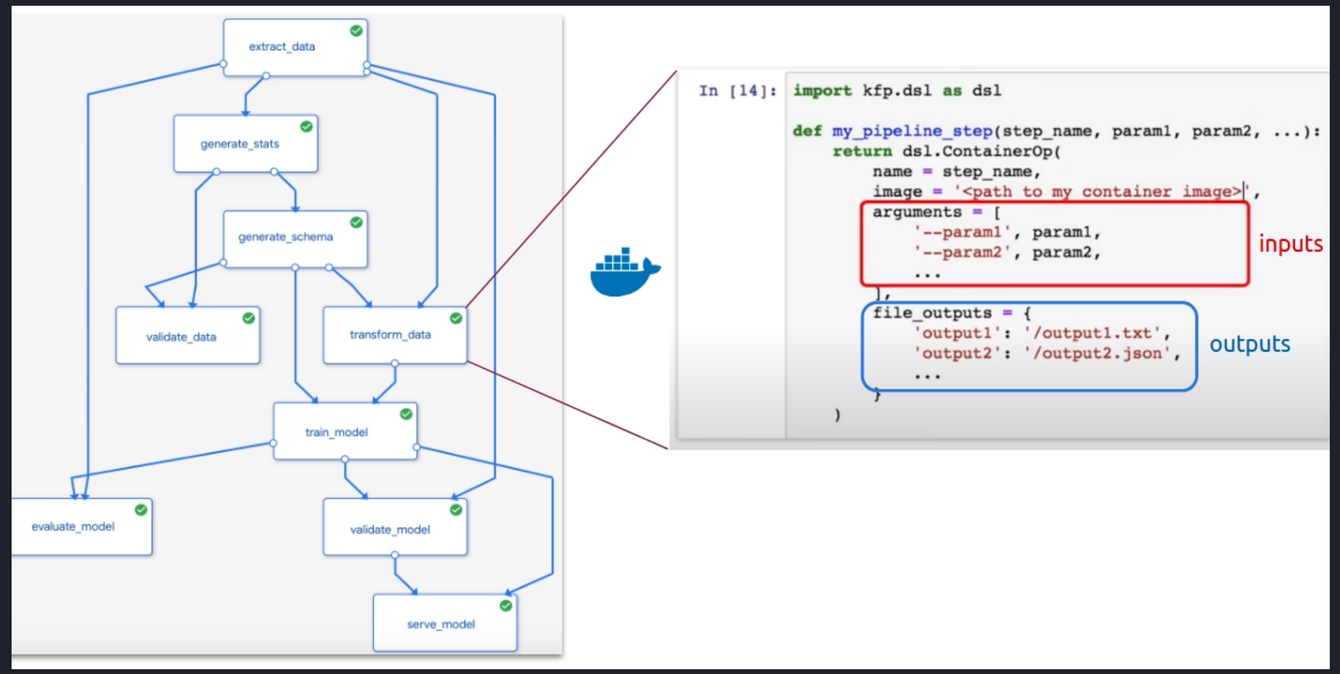
Kubernetes 위에서 ML workflow를 구축하고 배포하기위해 제공되는 플랫폼

Google에서 open-source로 발표, 이후 Vertex AI 라는 제품으로 공개

KubeFlow의 문법과 Kuberentes를 알아야 한다는 단점이 존재하나, 현재로서 많이 쓰이고 있는 MLOps 도구 중 하나

FEATURES

- Notebook Server
- **Pipeline**
- Serving
- HPO (Katib)
- **Training Job**



VESSL

MLflow

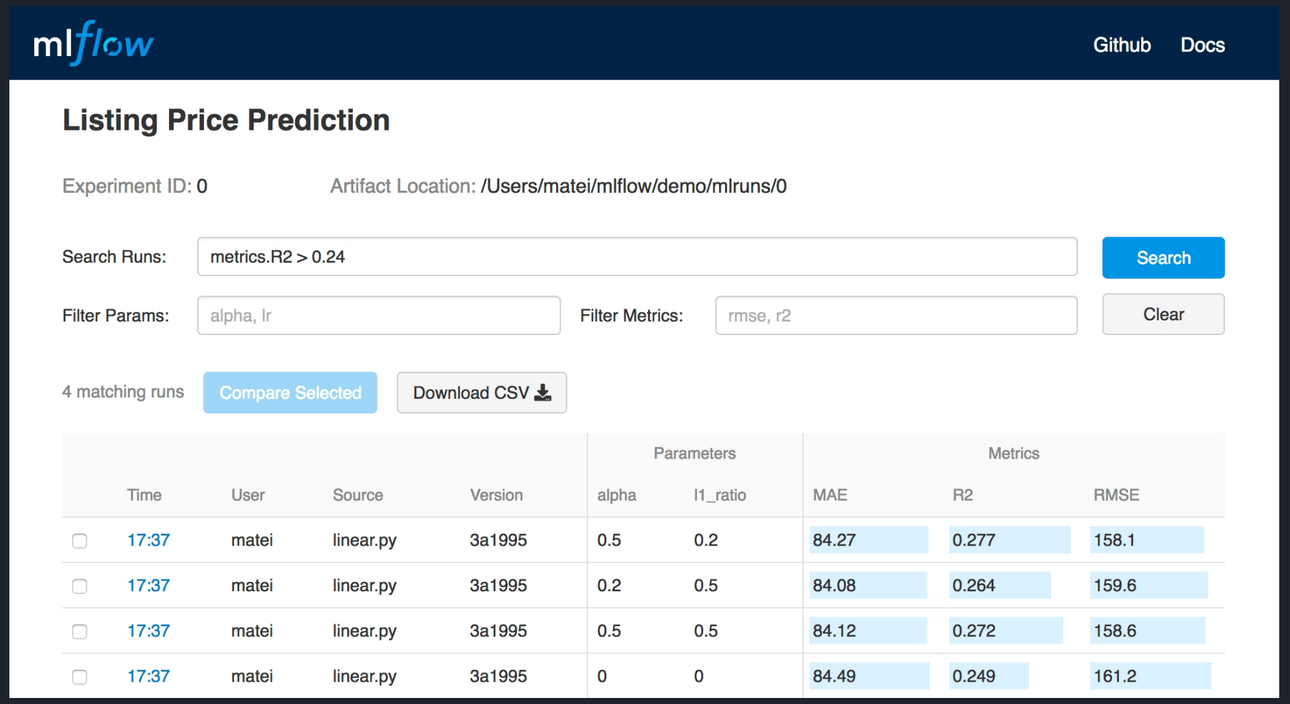
Tracking, Training Job (Project), Registry 등 MLOps의 기초적인 component를 open-source로 지원하는 제품

Databricks에서 제공, 더 많은 기능은 Databricks Platform 안에서 제공하고 있음

Open-source로 빠르게 시작해볼 수 있는 것이 장점, 제한적인 기능으로 더 많은 기능을 사용하려면 Databricks를 사용해야 함

FEATURES

- MLflow Tracking
- **MLflow Project**
- MLflow Models (Serving)
- **MLflow Registry**



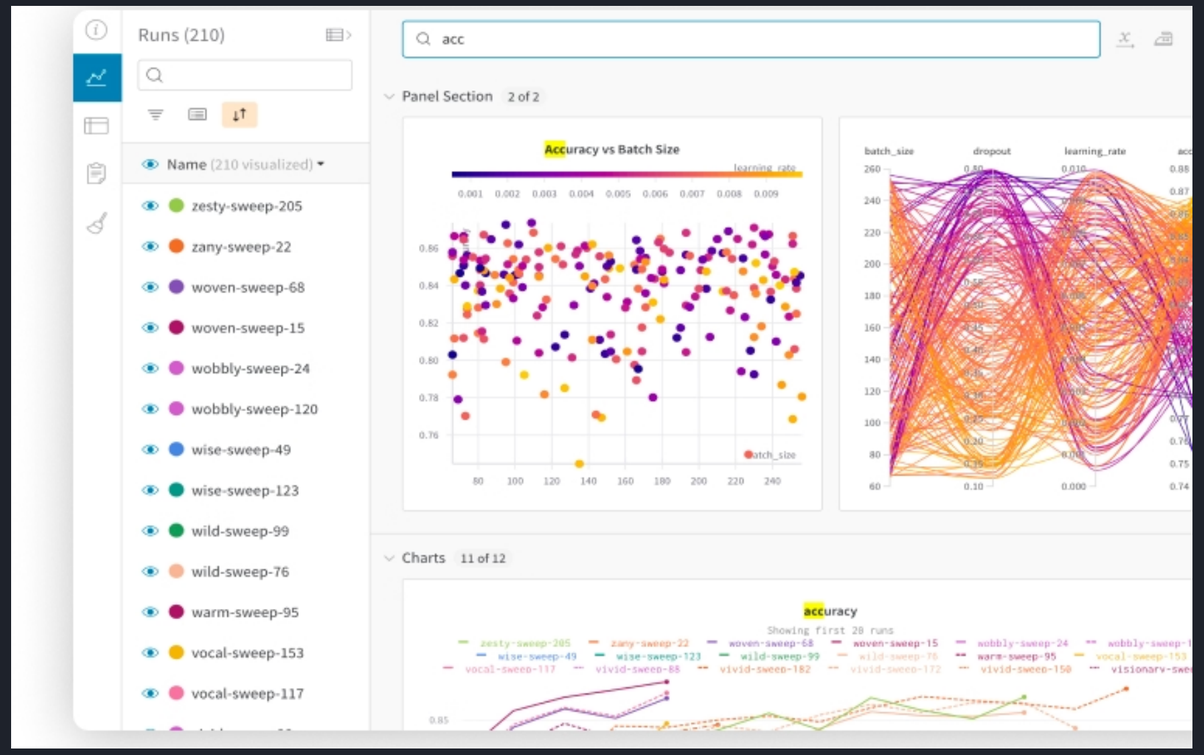
VESSL

Weights & Biases

강력한 Tracking Dashboard 기능 제공, Sweep, Artifacts, Reports 등의 기능을 추가 지원

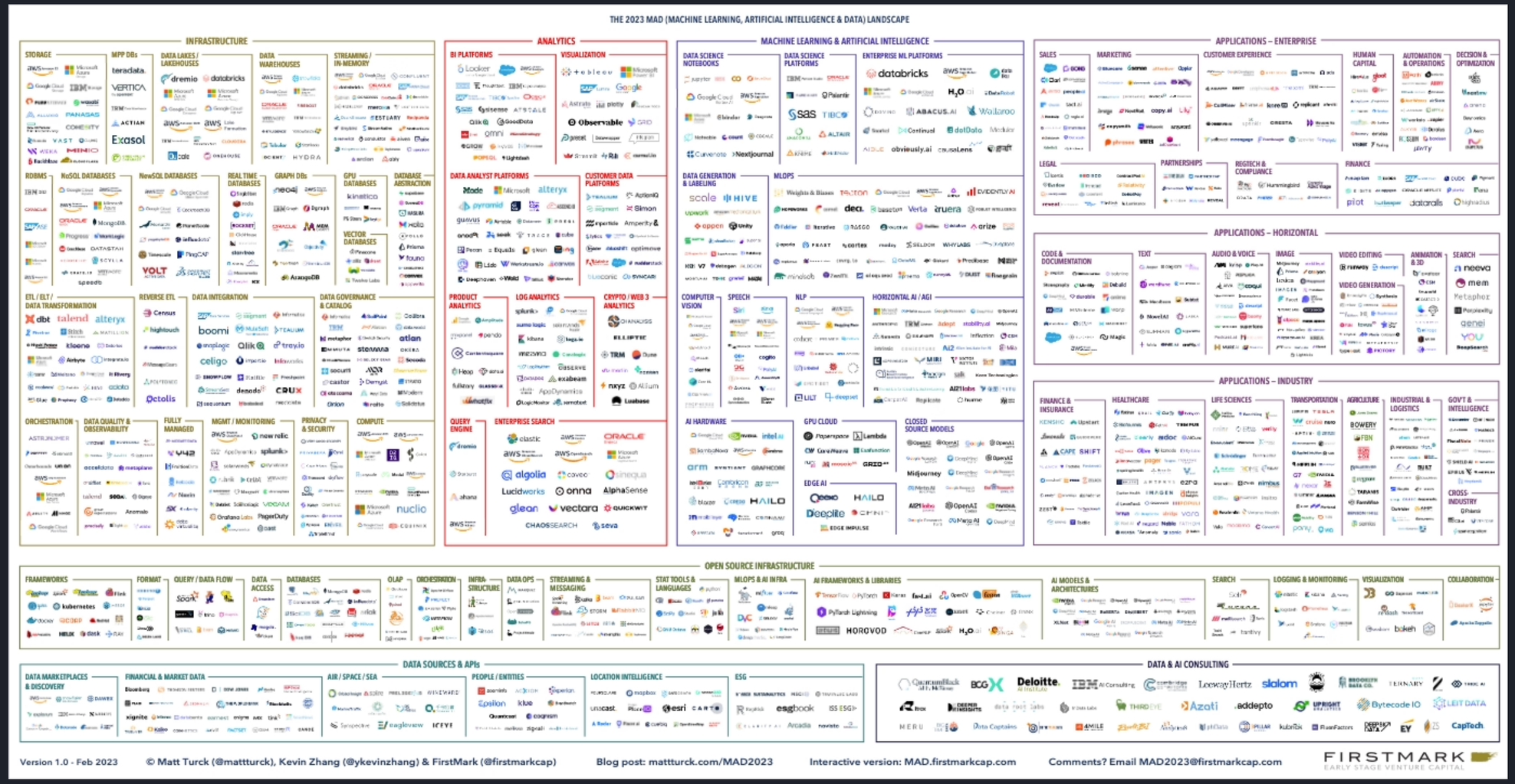
FEATURES

- **Experiment Tracking**
- Hyperparameter Tuning
- Data + Model Versioning
- **Collaborative Reports**



VESSL

그 외에도 많은 제품들이 있습니다...



https://mad.firstmark.com/

VESSL

MLOps: The VESSL Way

VESSL

What is VESSL?

The MLOps Platform - at any scale, on any cloud

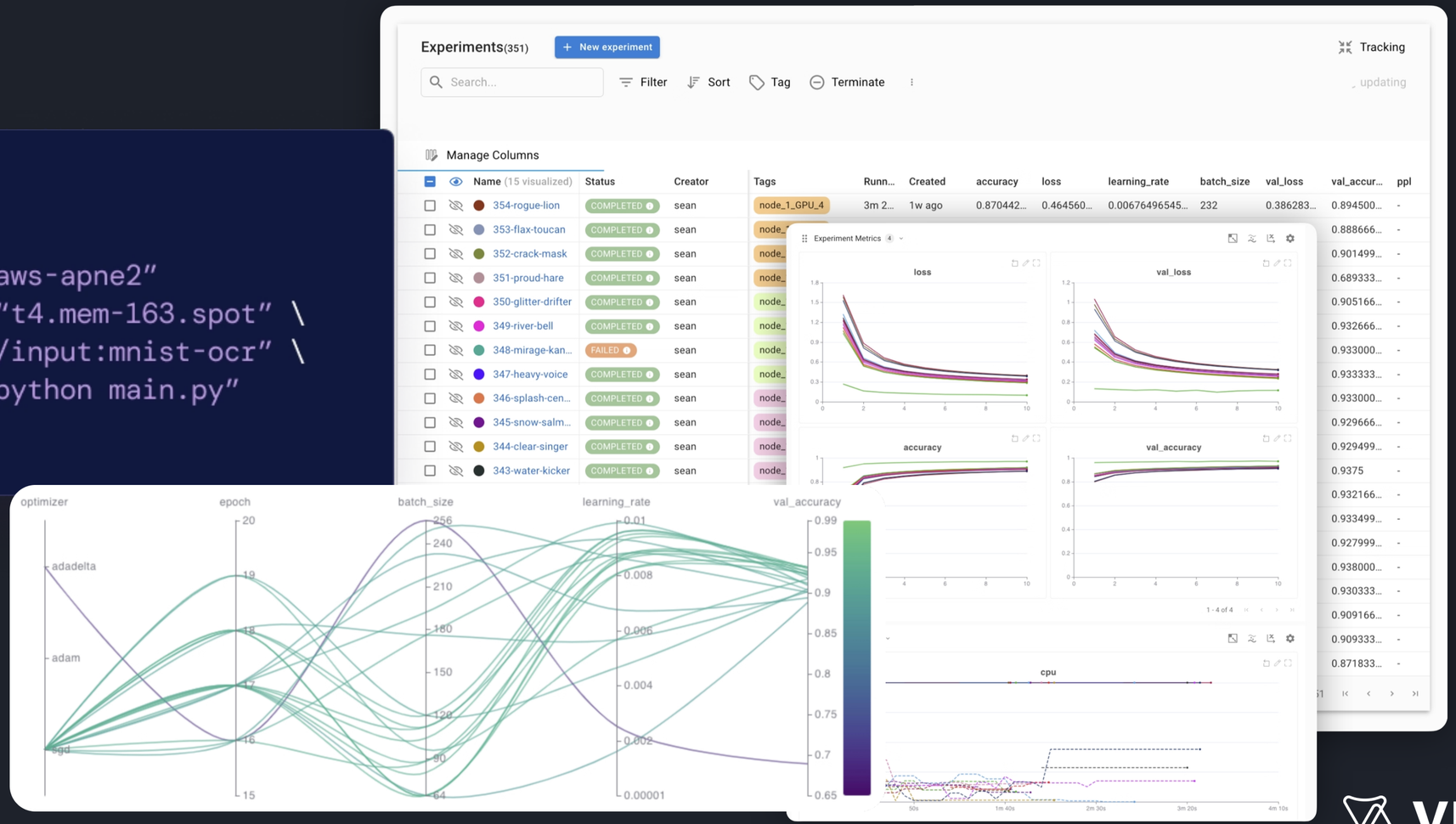
Build, train, deploy, automate on one platform.

MLOps 전 과정을 하나의 툴 안에서, 가장 쉽게 제공하는 것이 목표

VESSL

VESSL - Experiment Tracking

```
$ vessl run
    --cluster "aws-apne2"
    --resource "t4.mem-163.spot" \
    --dataset "/input:mnist-ocr" \
    --command "python main.py"
```



VESSL

VESSL - Cluster Management & Job Orchestration

Resource Specs

NAME	PROCESSOR	GPU TYPE	RESOURCE	PRICE	SPOT	EXPERIMI
gpu-1	GPU	NVIDIA-GeFor...	16CPU/60Gi/1GPU	0.1	false	enabled
gpu-2	GPU	NVIDIA-GeFor...	32CPU/120Gi/2GPU	0.1	false	enabled
gpu-4	GPU	NV				
gpu-8	GPU	NV				
no-gpu	CPU	Em				

Nodes 31

A node of a cluster is worker machine that handles a machine learning workload.

NAME	STATUS	CPU			
s00	Unavailable	5			
s01	Ready	18			
s02	Ready	9.(			
s03	Ready	8.2 / 128	84.2GiB / 503GiB	8 / 8	3 Workspaces 2 Experiments 0
s04	Ready	15.88 / 128	245GiB / 503GiB	6 / 8	1 Workspace 1 Experiment 0
s05	Ready	0.37 / 128	191GiB / 503GiB	6 / 8	3 Workspaces 0

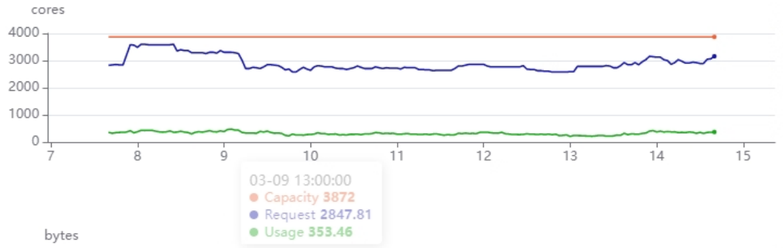
Resource statistics

1h | 3h | 12h | 1d | 3d | 1w | Refresh

CPU

Usage 370 / 3872

Requested 3158.81 / 3872



MEMORY

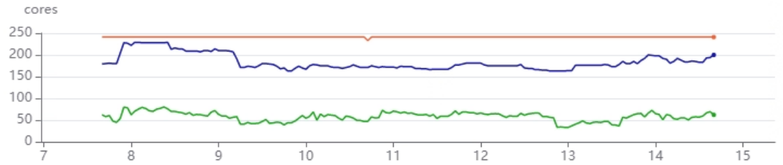
Usage 3.90TiB / 15.0TiB

Requested 11.6TiB / 15.0TiB

GPU

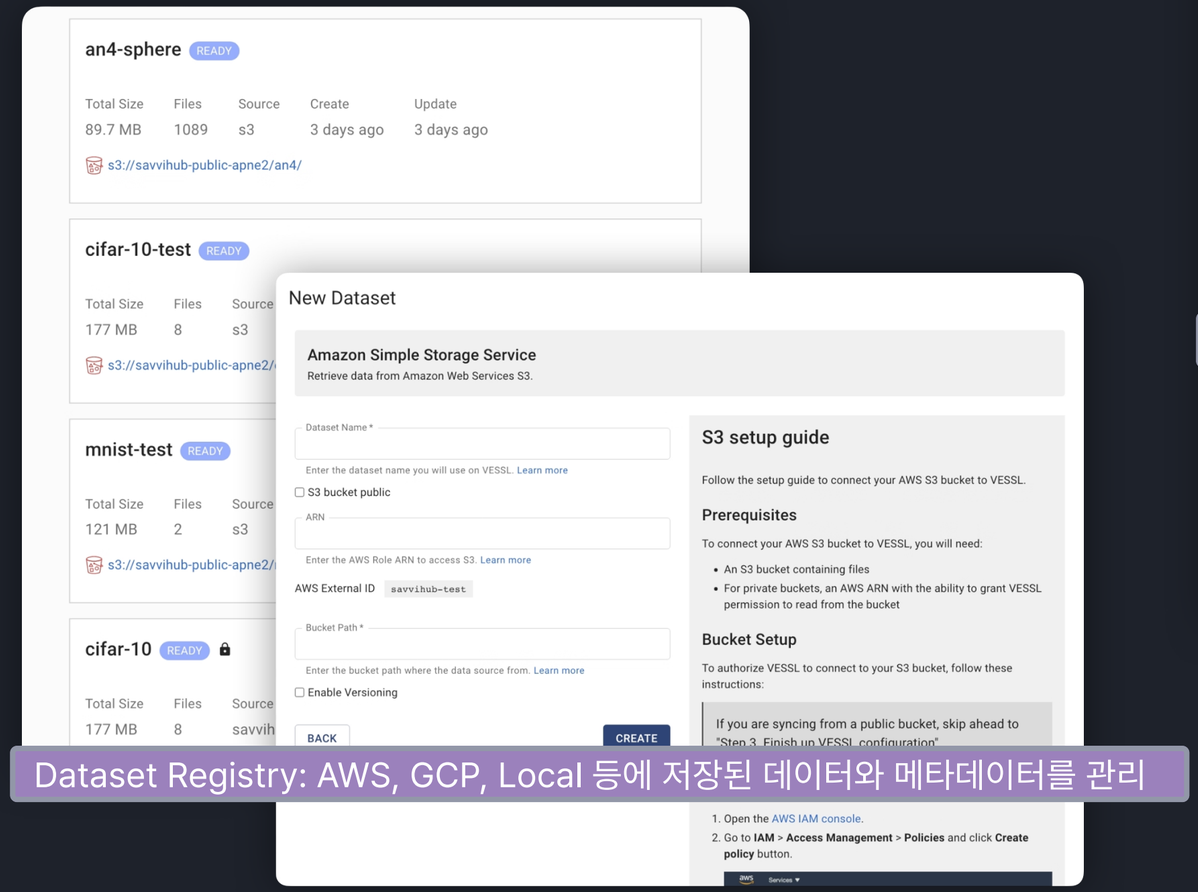
Usage 62.01 / 241

Requested 200 / 241

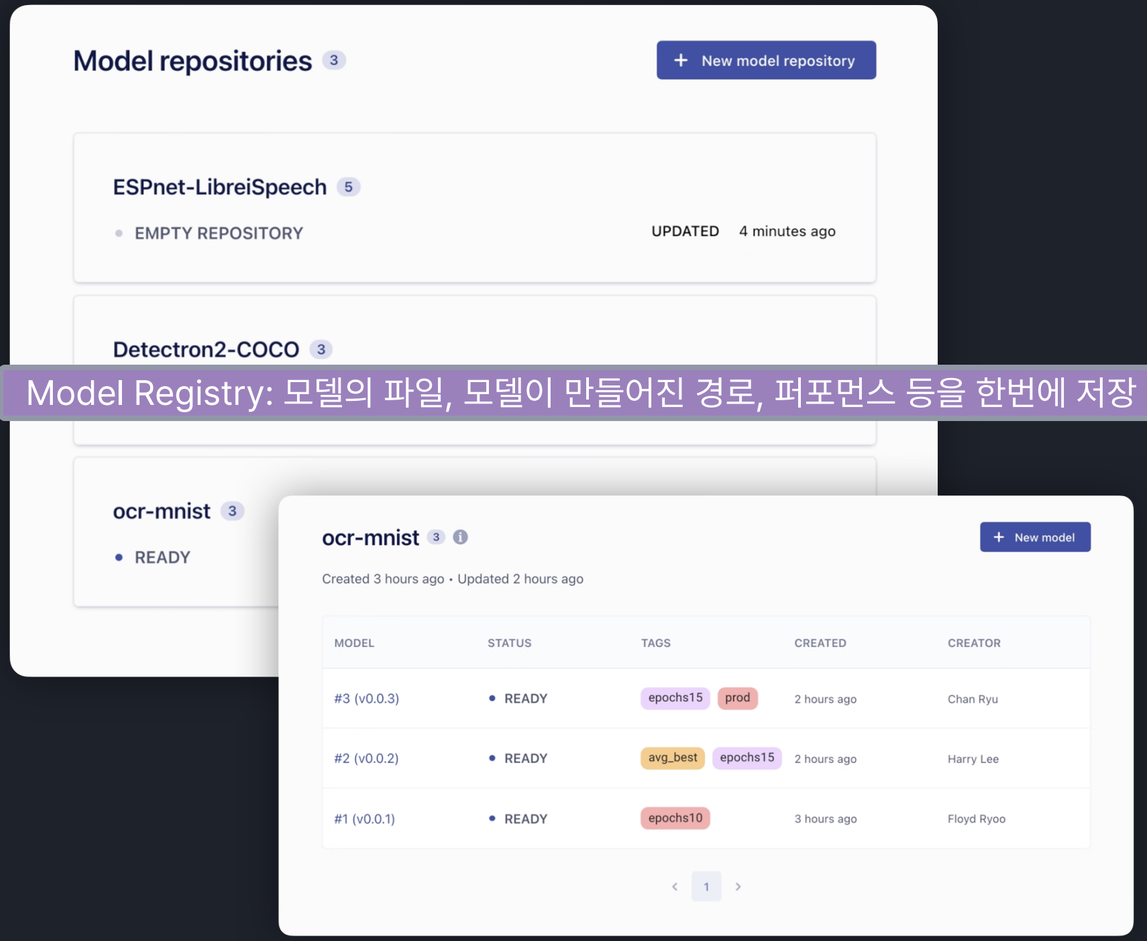


VESSL

VESSL - Dataset & Model Registry



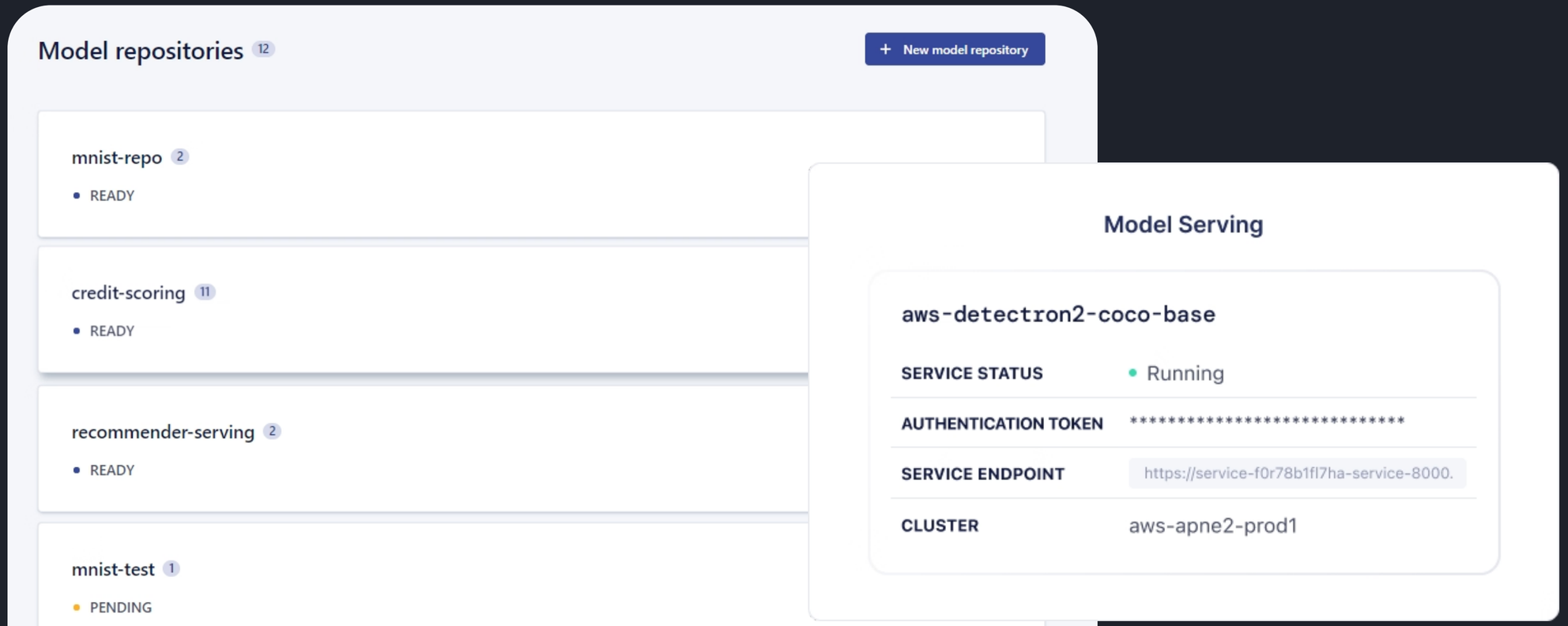
Dataset Registry: AWS, GCP, Local 등에 저장된 데이터와 메타데이터를 관리



Model Registry: 모델의 파일, 모델이 만들어진 경로, 퍼포먼스 등을 한번에 저장

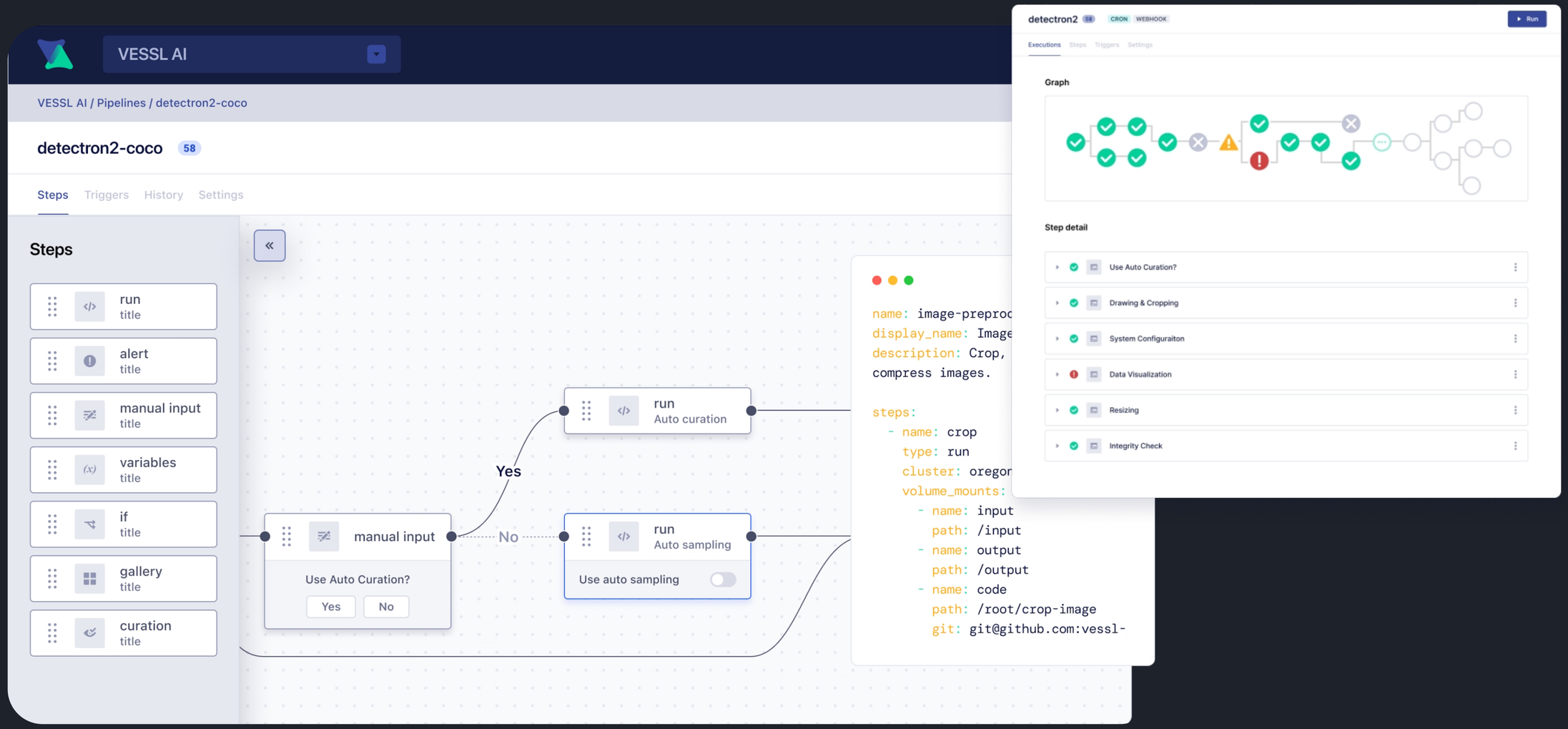
VESSL

VESSL - Model Serving



VESSL

VESSL - Pipeline



VESSL

SNU GPU First Program with VESSL AI

VESSL

SNU 공과대학 GPU First 소개

https://eng.snu.ac.kr/service/gpu/index.html

GPU First

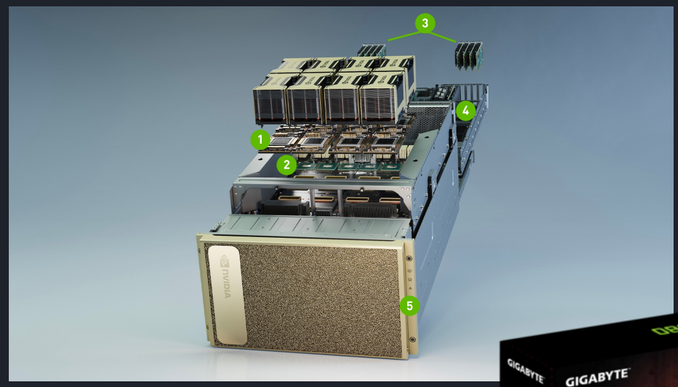
서울대학교 공과대학은 학내 고성능 컴퓨터 수요가 급증하고 외부 클라우드 서비스 이용률이 높아지면서 더 나은 조건으로 클라우드 서비스를 이용할 수 있도록 시중 요금의 약 10% (DGX A100 GPU 8개 사용기준 시간당 4,000원) 또는 무료(GTX 1080 사용기준) 프로그램을 제공하고 있습니다. 많은 관심과 적극적인 서비스 이용 부탁드립니다.

ENGINEERING 서울대학교공과대학 | VESSL AI

VESSL

SNU 공과대학 GPU 클러스터 소개

- https://vessl.ai/snu-eng-dgx
  - NVIDIA-A100-SXM4-40GB
  - 8 GPU devices, 256 CPU cores, ~1TB Memory (per-node)
  - 고성능이 필요한 큰 모델의 학습 용도
- https://vessl.ai/snu-eng-gtx1080
  - NVIDIA-GeForce-GTX-1080
  - 1 GPU device, 12 CPU cores, 15GB RAM (per-node)
  - 교육, 실습, 모델 코드의 간단한 검증을 위한 interactive coding(Notebook) 용도



VESSL

SNU 공과대학 GPU First 소개

DGX - A100

- 제품 : 8-GPU NVIDIA A100 40GB, 총 32 GPU카드, 13,824 Tensor Cores
- 성능 : 5 PFLOPS(FP32), 20 PFLOPS(AI), 40 POPS INT8
- 수량 : 4 노드
- 사용대상 : 서울대학교 공과대학 전 구성원
- AI/빅데이터 연구용, AI 교육용(대형강좌 등) 공동 활용
- 외부 클라우드 대비 **10%** 요금 적용 (DGX A100 GPU 8개 사용기준 시간당 4,000원)
- MLOps 플랫폼 VESSL 사용

GTX - 1080

- 제품 : DELL XPS 8930 SE
- 성능 : 인텔 i7-8700k, 16GB DDR4 RAM, GTX1080(8GB), 512GB M.2 SSD
- 수량 : 102대 (한송엽 명예교수님 20대, 이병호 교수님 12대, 김정식 대덕전자 회장님 27대 기증포함)
- 사용대상 : 서울대학교 공과대학 전 구성원
- 수업활용 : IoT 인공지능 빅데이터 개론 및 실습(2018. 2학기부터 시행)
- 행사활용 : SNU FastMRI Challenge (2021. 여름학기부터 시행)
- 연구활용 : 연구자 대여 (2018. 겨울학기부터 시행)
- 수업, 행사, 연구 활용에 한해 **무료**
- MLOps 플랫폼 VESSL 사용

VESSL

🙌 Hands-on VESSL AI Tutorial 🙌

VESSL

Hands-on Tutorial

- VESSL 회원 가입, organization 접속
- Workspace 생성 / Jupyter Notebook 접속
- Project 관리, Experiment 생성
- Experiment tracking, Sweep
- CLI / SDK
- VESSL Hub / VESSL Run 소개
- Q&A

VESSL

For more information...

- Official Docs
  - https://docs.vessl.ai/
- SNU User Manual
  - https://vesslai.notion.site/abd6a6cb25764b28b619e28236bd22eb
- Hubspot
  - From vessl.ai homepage
- Email
  - support@vessl.ai

VESSL