



MLOps Coursera - 1

202132033 염지현



2022. 01. 05.

good
slide

CONTENTS

MLOps[1]

Introduction to Machine Learning in Production

- I Overview of the ML Lifecycle and Deployment
- II Select and Train a Model
- III Data Definition and Baseline
- IV Scoping

CONTENTS

MLOps[2] Machine Learning Data Lifecycle in Production

I

Collecting, Labeling and Validating Data

II

Feature Engineering, Transformation and Selection

MLOps[1]

Introduction to Machine Learning in Production

I

Overview of the ML Lifecycle and Deployment

1. MLOps
2. ML Project Lifecycle
3. 개념 드리프트 모니터링

I 개념 및 사용하는 이유

MLOps(Machine Learning Operation)

1) 개념

- : 좋은 모델을 개발한 후에 실제 모델을 Production에 적용한 기술
- : Production Machine Learning System을 구축 및 배포하는데 필요한 기술

2) 사용하는 이유

- : Machine Learning Model이 아무리 훌륭해도 Production에 적용하지 못하면 최대 가치 창출이 어려움

MLOps

개발자의 역할

예: 컴퓨터비전 기반 스마트폰 제조 라인에서 나오는 스마트폰 결함 유무 예측

시스템 흐름 순서

검사 소프트웨어에서 스마트폰이 제조 라인에서 나올 때 사진 찍는 카메라 제어

검사 소프트웨어에서 촬영한 사진 예측 서버에 전달하기 위한 API 호출

예측 서버에서 API 호출 수락 및 이미지 수신

예측 서버에서 스마트폰 결함 여부 예측 및 결정

예측 서버에서 예측 반환

검사 소프트웨어 제조 라인에서 폐기/생산 여부 결정

개발자의 역할

예: 컴퓨터비전 기반 스마트폰 제조 라인에서 나오는 스마트폰 결함 유무 예측

시스템 흐름 순서

검사 소프트웨어에서 스마트폰이 제조 라인에서 나올 때 사진 찍는 카메라 제어

검사 소프트웨어에서 촬영한 사진 예측 서버에 전달하기 위한 API 호출

예측 서버에서 API 호출 수락 및 이미지 수신

예측 서버에서 스마트폰 결함 여부 예측 및 결정

예측 서버에서 예측 반환

검사 소프트웨어 제조 라인에서 폐기/생산 여부 결정

1. Machine Learning Model을 예측 서버에 올리기
2. API 인터페이스 설정하기
3. 소프트웨어 작성하기

MLOps

I 개발자의 역할

예: 컴퓨터비전 기반 스마트폰 제조 라인에서 나오는 스마트폰 결함 유무 예측

시스템 흐름 순서

검사 소프트웨어에서 스마트폰이 제조 라인에서 나올 때 사진 찍는 카메라 제어

검사 소프트웨어에서 촬영한 사진 예측 서버에 전달하기 위한 API 호출

예측 서버에서 API 호출 수락 및 이미지 수신

예측 서버에서 스마트폰 결함 여부 예측 및 결정

예측 서버에서 예측 반환

검사 소프트웨어 제조 라인에서 폐기/생산 여부 결정

1. Machine Learning Model을 예측 서버에 올리기
2. API 인터페이스 설정하기
3. 소프트웨어 작성하기

개발자의 역할

예: 컴퓨터비전 기반 스마트폰 제조 라인에서 나오는 스마트폰 결함 유무 예측

시스템 흐름 순서

검사 소프트웨어에서 스마트폰이 제조 라인에서 나올 때 사진 찍는 카메라 제어

검사 소프트웨어에서 촬영한 사진 예측 서버에 전달하기 위한 API 호출

예측 서버에서 API 호출 수락 및 이미지 수신

예측 서버에서 스마트폰 결함 여부 예측 및 결정

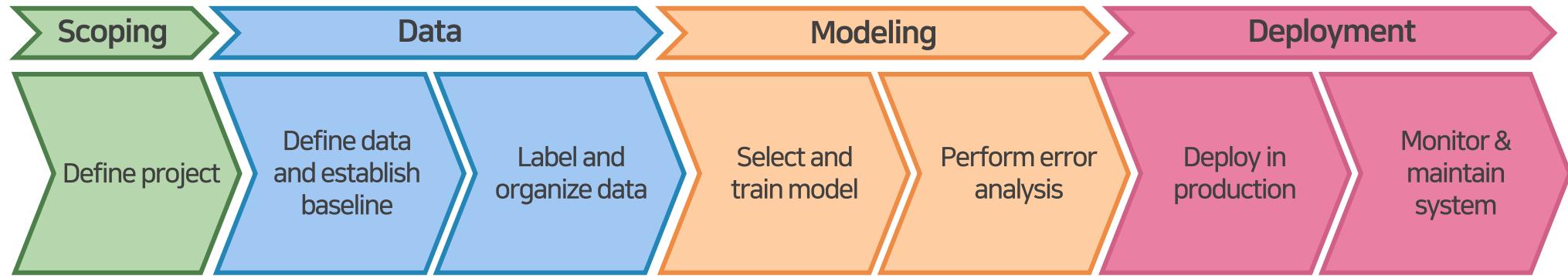
예측 서버에서 예측 반환

검사 소프트웨어 제조 라인에서 폐기/생산 여부 결정

1. Machine Learning Model을 예측 서버에 올리기
2. API 인터페이스 설정하기
3. [소프트웨어 작성하기](#)

ML Project Lifecycle

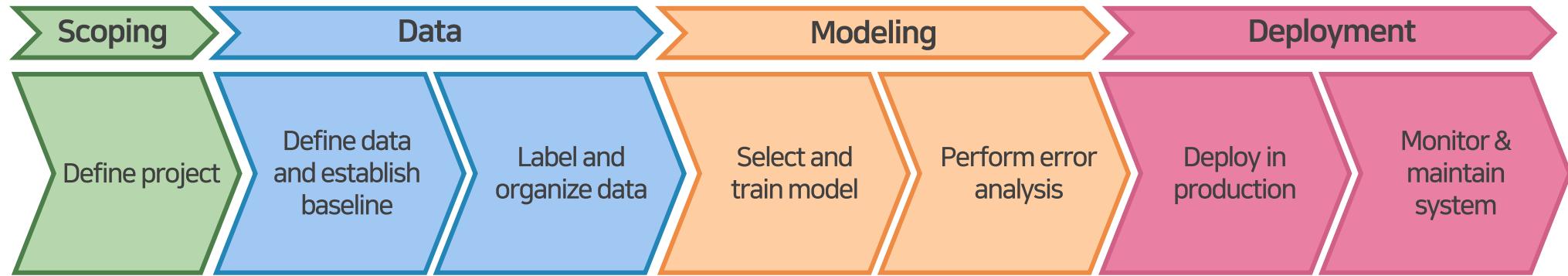
프로세스



ML Project Lifecycle

Scoping

예: 음성 인식 시스템



Scoping 단계: 프로젝트 정의 단계

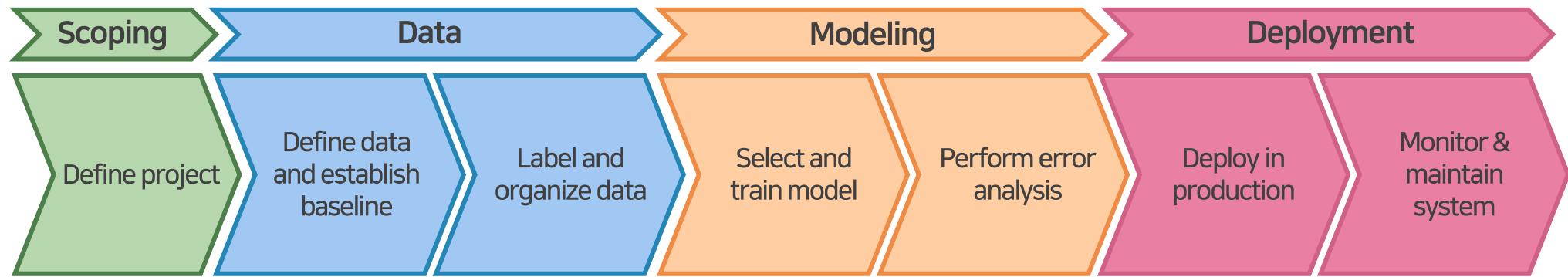
예) 음성 인식 작업에 대한 결정

- 비즈니스 가치
- 음성인식 시스템 대기 시간, 처리량, 초당 처리하는 쿼리 수, 필요한 리소스 추정
- 예산과 일정 계산

ML Project Lifecycle

Data

예: 음성 인식 시스템



Data 단계: Data 정의 및 기준선 설정, Data labeling 및 정리

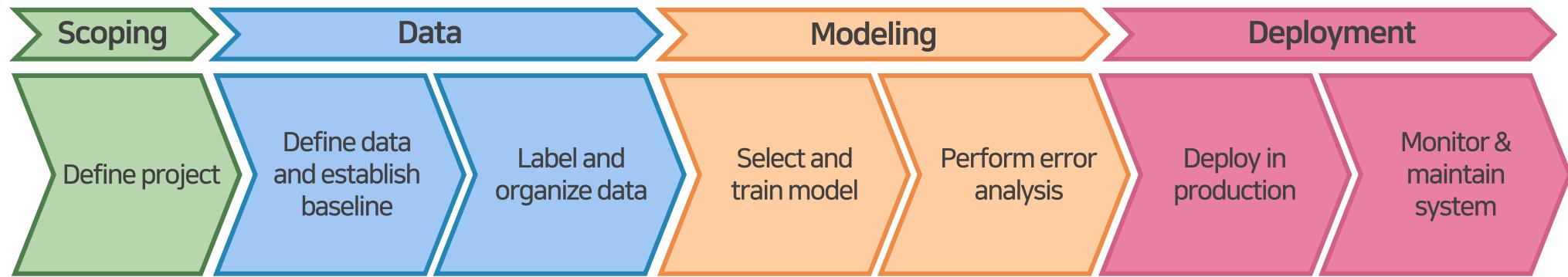
예) 음성 인식 작업 관련 데이터 정의

- 데이터 정의: label의 일관성 고려
 - “어? 안녕.”, “어. 안녕.”, “어! 안녕” → 일관되지 않은 레이블링
- 고품질 데이터를 확보하기 위한 체계적인 프레임워크 수행
 - 오디오에서 어느 부분을 clip?
 - 시작 지점, 끝 지점 정의

ML Project Lifecycle

Modeling

예: 음성 인식 시스템



Modeling 단계: 모델 선택, 훈련, 오류 분석

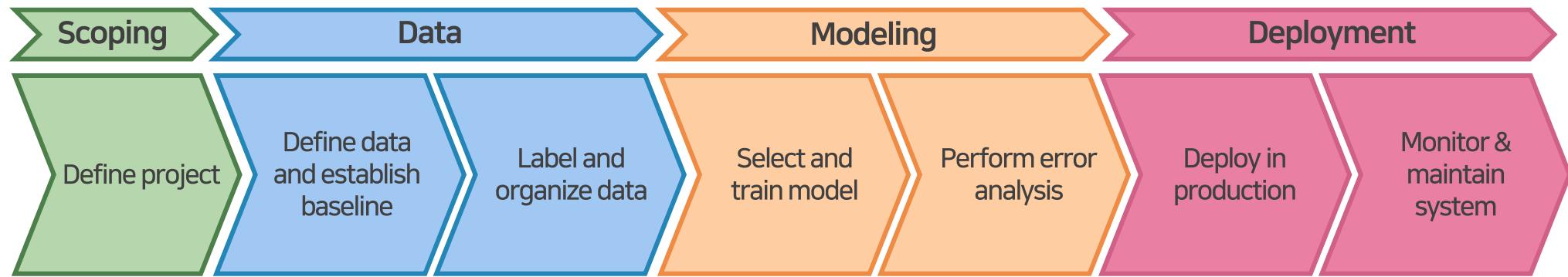
예) 음성 인식 기계 학습 모델

- 연구 목적: Data 고정 → code 수정, 하이퍼파라미터 수정
- 비즈니스 가치 창출 목적: Code 고정 → 데이터 수정, 하이퍼파라미터 수정

ML Project Lifecycle

Deployment

예: 음성 인식 시스템



Deployment 단계: 배포

예) 음성 인식 기계 학습 모델 배포

- 일반적인 배포 패턴
 - 학습 알고리즘 + VAD(음성 활동 감지 모듈) 작업
 - 스마트폰에서 사람이 말하는 오디오만 선택
 - 오디오 클립을 예측 서버에 전송
 - 스크립트를 반환하여 사용자가 시스템에서 말한 내용 출력
- Concept/data drift: “대박”이라는 유행어 데이터 학습했으나 시간이 지나면서 “킹받네”를 인식하지 못함 → 음성 인식 성능 저하
→ 데이터 보완 필요

ML Project Lifecycle

배포가 어려운 이유

ML 통계적 문제

- Concept drift
 - 입력 데이터의 분포는 변하지 않았으나 시간이 지남에 따라 **예측 결과가 달라짐**
 - 예: $x = \text{주택 크기}, y = \text{주택 가격}$ 일 때, 시간이 지남에 따라 주택 가격이 더 비싸질 수 있음
- Data drift
 - 시간이 지남에 따라 **입력 데이터의 분포가 변함**
 - 예: 주택의 크기가 커지거나, 더 작아지는 경우 시간이 지남에 따라 주택에 대한 정보가 변할 수 있음

SW 엔진 고려사항

- Realtime or Batch?
 - 예: 음성 인식은 실시간 예측이 필요하지만 병원의 경우 밤마다 환자의 상태를 일괄적으로 기록
- Cloud vs. Edge/Browser?
 - 예: 자동차, 공장과 같이 인터넷 연결이 끊겨도 계속 수행해야 될 경우 edge device에서 수행
- Computer resource
 - 메모리 리소스 양을 파악하여 압축 정도 확인
- Latency, throughput
 - 대기 시간과 처리량 고려
- Logging
 - 분석 및 검토를 위해 가능한 많은 데이터 로깅
 - 향후 학습 알고리즘 재교육을 위해 사용
- 보안 및 개인 정보 보호
 - 민감도에 따라 규정 요구사항을 기반으로 적절한 수준의 보안 및 개인정보 보호 설계

Monitoring

I 개념 및 추적 방법

Monitoring

1) 개념

: 대시보드를 사용하여 시간 경과에 따라 어떻게 작동하는지 추적하는 방법

*대시보드: 시간이 지남에 따라 변화하는 과정을 그래프로 표현한 보드

: 시간이 지남에 따라 큰 변화가 있을 경우, 무언가 잘못되었다는 표시

: 모니터링 대상을 결정하려고 할 때 팀원들끼리 잘못될 수 있는 모든 상황에 대해 브레인스토밍 하는 것이 좋음

: 오류 분석 수행, 더 많은 데이터 수집을 얻기 위해 [이전 단계로 돌아가도록 문제를 찾아주는 것은](#) 오직 시스템 모니터링을 통해서만 가능

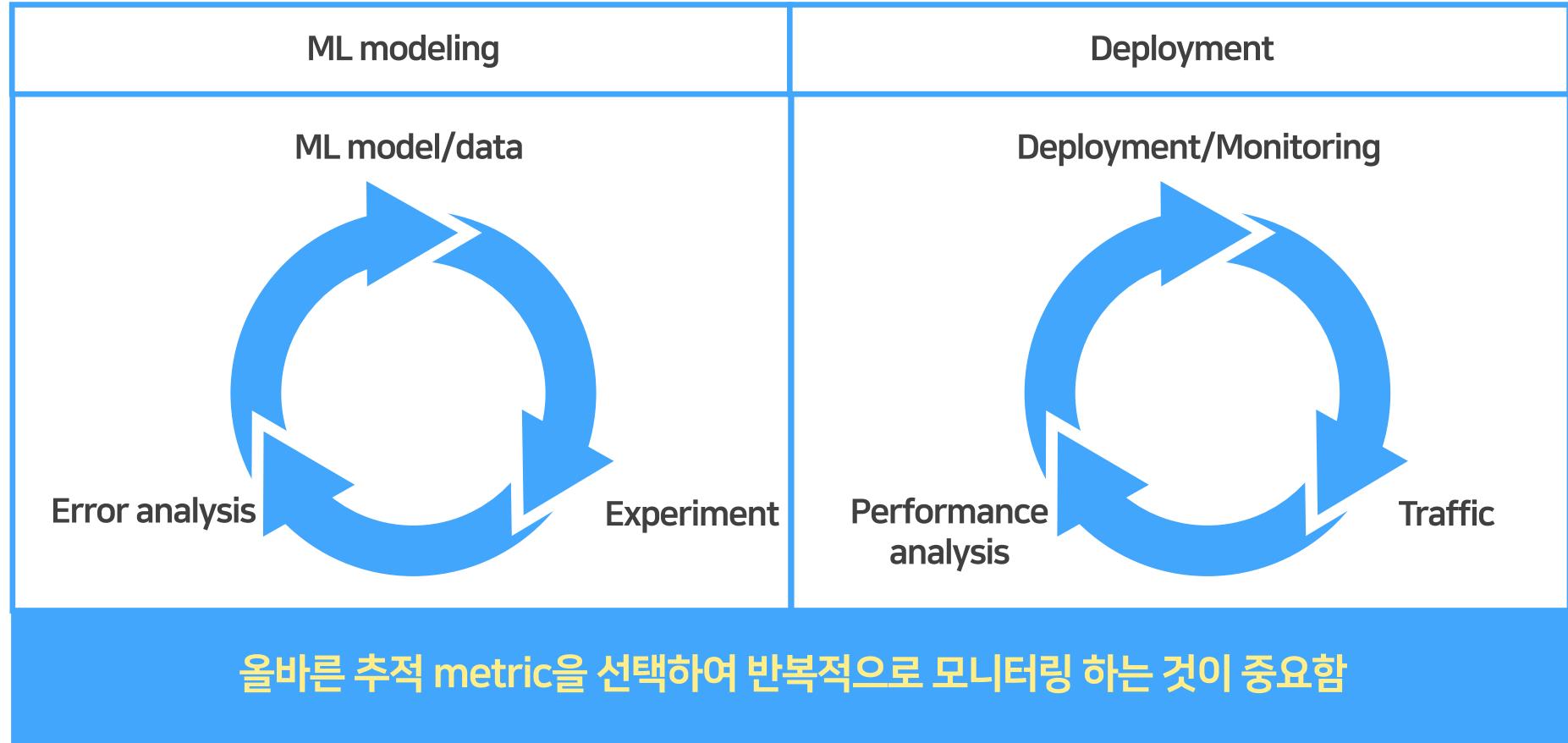


2) 추적 방법

- Software metric: 메모리, 지연 시간, 처리량 확인 → SW가 제대로 실행되는지 확인 가능
- Input metric: Avg input length, Avg input volume, Num missing value → 입력 분포 변화 확인
- Output metric: # times return null, # times user redo search, CTR → 출력 분포 변화 확인
- 입력 및 출력 메트릭은 애플리케이션에 따라 다르기 때문에 도구는 애플리케이션에 맞추어 선택

Monitoring

I ML 모델링과 배포 비교



MLOps[1]

Introduction to Machine Learning in Production

II

Select and Train a Model

1. 모델링 주요 과제
2. 모델링 파이프라인
3. Data-centric AI development

모델링 주요 과제

모델링 주요 과제

1

하위 집합 데이터 세트 성능

- 모든 데이터 그룹에 있어서 성능이 동일한가?
- 편향으로 인해 차별이 발생되지 않는가?
- 예: 대출 승인 예측 프로그램, 가게 추천 시스템 등

2

희귀 클래스

- 희귀 클래스에게 있어서 정확도의 오류가 없는가?
- 예: 인구의 99%가 질병이 없고 1%만 질병이 있다면 입력 데이터를 기반으로 예측할 때 "질병이 없음" 만 출력해도 99% 정확도 달성

3

테스트 세트에 대한 성능

- 개발자는 테스트 세트에서 요구한 정확도를 달성하였으나 실제 애플리케이션에서는 정확도를 달성하지 못함
- 테스트 세트에서 잘 수행되는 정도에 그치지 않고 프로덕션 배포에서도 동일한 성능을 보이는지 확인 필요

모델링 파이프라인

II Establish a baseline

Unstructured data

Image



Audio



Text

사랑하는 사람에게,
이 풍화체를 네가 소중한 이를 알고 싶었다.
하지만 너는 풍화체에 만족하지 않고, 너의 마음을 알고 싶어졌다.
내가 사랑하는 이 풍화체를 알았다면,
이 성장이 너에게 허락되는 풍화를 되게 좋겠다.
- 너의 향기를 바라는 엄마, 아빠,

Structured data

번호	이름	국어	수학	영어	화학
1	바다야크	80	90	70	
2	사과	100	80	80	
3	배	70	60	80	
4	포도	70	100	80	
5	バナナ	80	90	100	
6	수박	80	50	70	
~~~		~~~	~~~	~~~	~~~

직원 평가표

소속	이름	업적평가(50)	역량평가(40)	정성평가(10)	합계
영업팀	홍길동	45	35	10	90
영업팀	임꺽정	40	32	9	81
총무팀	심장	48	38	10	96
생산팀	장길산	40	32	9	81
생산팀	손오공	40	35	8	83
지원팀	사오정	48	38	10	96

- 사람이 잘 해석할 수 있는 데이터
- HLP가 baseline에 적절

- 사람이 잘 해석하지 못하는 데이터
- HLP가 baseline에 적절하지 않음

*HLP(Human Level Performance)

# 모델링 파이프라인

II

## Establish a baseline 방법

Human Level Performance(HLP)

최신 기술에 대한 문헌 검색 및 오픈 소스 결과 비교

무작정 구현하기

- 어떤 것이 가능/불가능할 지에 대한 감각을 얻을 수 있음-

(이미 배포된 경우) 이전 시스템과의 성능 비교

# 모델링 파이프라인

## 모델링 시작을 위한 Tips



- 실용적인 생산 시스템 구축이 목표라면 최신 연구에 집착하지 않기
  - 사용 가능한 오픈 소스를 찾고 baseline 잡기
  - 좋은 데이터가 포함된 합리적인 알고리즘을 통해 좋은 성능 보여주기
- 
- 모델 선택할 때 컴퓨팅 제약 조건, 배포 제약 조건을 고려해야 하는가?
    - Baseline이 설정, 프로젝트 작동에 대한 확신이 있을 경우 조건을 고려
    - Baseline 미설정, 프로젝트 미작동 할 경우 먼저 baseline을 설정하고 무엇이 가능한지 결정하는 것이 우선
- 
- 적은 시간과 데이터만으로 버그 찾기 가능
  - 예1) 음성 인식 시스템
    - 하나의 data에 대해 null이 반환된다면 제대로 수행되고 있지 않기 때문에 훈련 무의미
  - 예2) 이미지 분할
    - 하나의 이미지에 대해 훈련하여 결과 확인
  - 예3) 이미지 분류
    - 10, 100개 이미지 하위 집합에서 훈련

# 모델링 파이프라인

## 오류 분석(1): 태그 설정

### 잘못된 예측에 태그를 설정하여 오류 분석

예: 음성 인식 시스템

Example	Label	Prediction	Car Noise	People Noise
1	"Stir fried lettuce recipe"	"Stir fry lettuce recipe"	v	
2	"Sweetened coffee"	"Swedish coffee"		v
3	"Sail away song"	"Sell away song"		v
4	"Let's catch up"	"Let's ketchup"	v	v

- Example 1~4에 대해 잘못된 예측
- "Car Noise", "People Noise" 태그를 설정하여 어떤 소음이 있었는지 체크
- 태그를 통해 오류 분석
- "Low Bandwidth" 새로운 태그 설정
- [잘못된 예측 정리 → 태그 설정 → 오류 분석 → 새로운 태그 설정] 프로세스 반복

# 모델링 파이프라인

## II 오류 분석(1): 태그 설정

### 잘못된 예측에 태그를 설정하여 오류 분석

예: 음성 인식 시스템

Example	Label	Prediction	Car Noise	People Noise	Low Bandwidth
1	"Stir fried lettuce recipe"	"Stir fry lettuce recipe"	v		
2	"Sweetened coffee"	"Swedish coffee"		v	v
3	"Sail away song"	"Sell away song"		v	
4	"Let's catch up"	"Let's ketchup"	v	v	v

- Example 1~4에 대해 잘못된 예측
- "Car Noise", "People Noise" 태그를 설정하여 어떤 소음이 있었는지 체크
- 태그를 통해 오류 분석
- ["Low Bandwidth" 새로운 태그 설정](#)
- [잘못된 예측 정리 → 태그 설정 → 오류 분석 → 새로운 태그 설정] [프로세스 반복](#)

# 모델링 파이프라인

## II 오류 분석(1): 태그 설정

### 태그 기반 우선 순위 지정

예: 음성 인식 시스템

Type	Accuracy	HLP	Gap to HLP	% of data
Clean Speech	94%	95%	1%	60%
Car Noise	89%	93%	4%	4%
People Noise	87%	89%	2%	30%
Low Bandwidth	70%	70%	0%	6%

#### 성능 향상 예측

- “Clean Speech”: 1% 성능 향상 기대 → 60% 데이터를 1% 성능 향상 기대 = 0.6% 향상
- “Car Noise”: 4% 성능 향상 기대 → 4% 데이터를 4% 성능 향상 기대 = 0.16% 향상
- “People Noise”: 2% 성능 향상 기대 → 30% 데이터를 2% 성능 향상 기대 = 0.6% 향상
- “Low Bandwidth”: 0% 성능 향상 기대 → 6% 데이터를 0% 성능 향상 기대 = 0% 향상

→ 개선의 여지를 파악하여 우선 순위 설정

# 모델링 파이프라인

## 오류 분석(1): 태그 설정

개선의 여지

해당 태그가 얼마나 자주 출현하는지

해당 태그에서 정확도를 향상시키는 것이 얼마나 쉬운지

해당 태그에서 성능을 향상시키는 게 얼마나 중요한지

데이터 추가 수집

데이터 증강

데이터 품질 개선

레이블 품질 개선

# 모델링 파이프라인

## 오류 분석(2): Confusion matrix

		Actual(정답)	
		Negative	Positive
Prediction(예측)	Negative	TN(True Negative) : 실제 정답이 Negative 일 때, 예측도 Negative	FN(False Negative) : 실제 정답이 Positive 일 때, 예측이 Negative
	Positive	FP(False Positive) : 실제 정답이 Negative 일 때, 예측이 Positive	TP (True Positive) : 실제 정답이 Positive 일 때, 예측도 Positive

$$\text{Precision} = \frac{TP}{TP+FP}$$

- 임계치가 낮아서 Positive라고 예측한 비율이 높아질 경우: TP, FP 증가 & FN 감소  
→ Recall 증가

$$\text{Recall} = \frac{TP}{TP+FN}$$

- 임계치가 높아서 Negative라고 예측한 비율이 높아질 경우: TN, FN 증가 & FP 감소  
→ Precision 증가

# 모델링 파이프라인

## 오류 분석(2): Confusion matrix

		Actual(정답)	
		Negative	Positive
Prediction(예측)	Negative	TN(True Negative) : 실제 정답이 Negative 일 때, 예측도 Negative	FN(False Negative) : 실제 정답이 Positive 일 때, 예측이 Negative
	Positive	FP(False Positive) : 실제 정답이 Negative 일 때, 예측이 Positive	TP (True Positive) : 실제 정답이 Positive 일 때, 예측도 Positive

$$\text{Precision} = \frac{TP}{TP+FP}$$

- 모델이 Negative만 반환한다면?

- Precision = 0 / 0 + 0

- Recall = 0 / 0 + FN = 0 → FN 영향력 무시

$$\text{Recall} = \frac{TP}{TP+FN}$$

# 모델링 파이프라인

## 오류 분석(2): Confusion matrix

		Actual(정답)	
		Negative	Positive
Prediction(예측)	Negative	TN(True Negative) : 실제 정답이 Negative 일 때, 예측도 Negative	FN(False Negative) : 실제 정답이 Positive 일 때, 예측이 Negative
	Positive	FP(False Positive) : 실제 정답이 Negative 일 때, 예측이 Positive	TP (True Positive) : 실제 정답이 Positive 일 때, 예측도 Positive

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

- 모델이 Negative만 반환한다면?
  - Precision = 0 / 0 + 0
  - Recall = 0 / 0 + FN = 0 → FN 영향력 무시

# 모델링 파이프라인

## 오류 분석(2): Confusion matrix

		Actual(정답)	
		Negative	Positive
Prediction(예측)	Negative	TN(True Negative) : 실제 정답이 Negative 일 때, 예측도 Negative	FN(False Negative) : 실제 정답이 Positive 일 때, 예측이 Negative
	Positive	FP(False Positive) : 실제 정답이 Negative 일 때, 예측이 Positive	TP (True Positive) : 실제 정답이 Positive 일 때, 예측도 Positive

$$\text{Precision} = \frac{TP}{TP+FP}$$

- 모델이 Negative만 반환한다면?

- Precision = 0 / 0 + 0

- Recall = 0 / 0 + FN = 0 → FN 영향력 무시

$$\text{Recall} = \frac{TP}{TP+FN}$$

# 모델링 파이프라인

## 오류 분석(2): Confusion matrix

	Precision(P)	Recall(R)	F ₁ score
Model 1	88.3	79.1	83.4
Model 2	97.0	7.3	13.6

$$F_1\text{Score} = \frac{2}{\frac{1}{P} + \frac{1}{R}}$$

- Precision 과 Recall 결합
- Precision과 Recall 사이 조화평균을 취해 더 나쁜 지표를 강조

# 모델링 파이프라인

## 성능 검사

시스템이 잘못될 수 있는 다양한 상황에 대해 브레인스토밍

문제에 대해 데이터의 일부분을 사용하여 알고리즘 성능을 평가하기 위한 메트릭 설정

클라이언트에게 가장 걱정할 만한 문제, 문제에 대해 합리적으로 평가할 메트릭에 대한  
동의 얻기

# Data-centric AI development

## Data-centric AI 개념

### Data-centric AI

#### 개념

- : Data 품질이 중심이 되는 AI 모델
- : 오류 분석, 데이터 증강과 같은 도구를 사용하여 데이터 품질을 체계적으로 개선 중요
- : 다양한 모델에서 잘 작동

# Data-centric AI development

## II

## Data Augmentation(비정형 데이터)

Audio	Signal + Noise	<p>고려 사항</p> <ul style="list-style-type: none"> <li>• 노이즈 유형 선택</li> <li>• 노이즈는 볼륨 크기</li> </ul> <p>체계적인 고려 필요</p>
Image		<p>방법</p> <ul style="list-style-type: none"> <li>• 뒤집기</li> <li>• 대비 변경</li> <li>• 밝기 변경 → 너무 어두우면 인간도 인식 불가</li> <li>• 포토샵 사용</li> <li>• GAN 사용</li> </ul>
<ul style="list-style-type: none"> <li>• 목표: 알고리즘에서 잘 작동하지 않지만 인간이나 다른 baseline은 잘 수행할 수 있는 실제와 같은 샘플 만들기</li> <li>• 이미 잘 작동하는 샘플은 만들 필요가 없음</li> <li>• 인간도 해석할 수 없는 데이터는 필요가 없음</li> <li>• Check-list: 인간이 인식할 수 있는가? 사실적인가?</li> </ul>		

# Data-centric AI development

## II

## Adding features(정형 데이터)

### 예: 레스토랑 추천 시스템

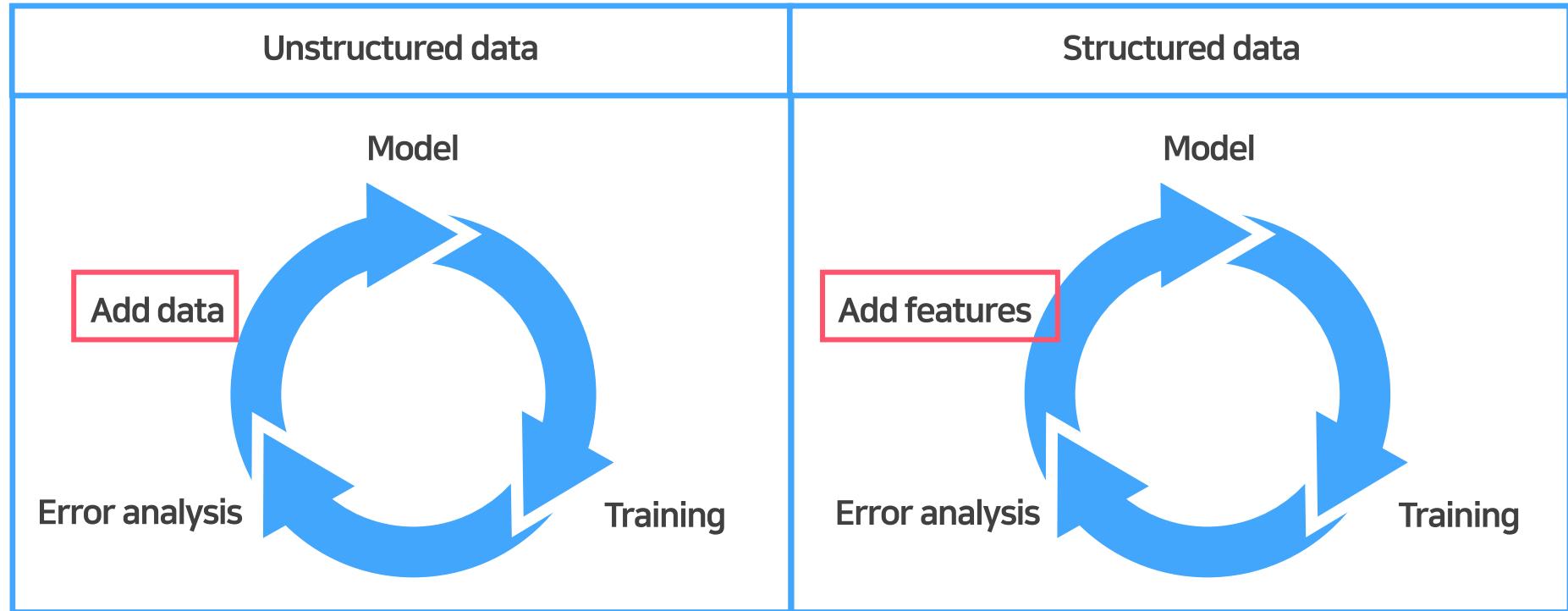
Input	사용자 정보, 레스토랑 정보
Output	식당 추천 여부
Error	채식주의자에게 육류 옵션이 있는 레스토랑 추천
Solution	새로 추가할 feature가 있는지 확인

• Collaborative filtering: 사용자를 살펴보고 사용자와 유사한 사람을 파악한 후 추천하는 접근 방식  
→ 시간이 필요

• Content based filtering: 식당 설명, 메뉴를 기반으로 식당이 사용자에게 맞는지 알아보고 추천 여부 결정  
→ 새로운 데이터에도 사용 가능

# Data-centric AI development

## Data iteration



MLOps[1]

Introduction to Machine Learning in Production

III

# Data Definition and Baseline

1. Data definition
2. HLP
3. Data pipeline
4. Metadata
5. Balanced dataset

# Data definition

## III Data 정의 요소

Input x

- 밝기, 대비, 해상도 고려
- Feature가 알고리즘 성능에 미치는 영향

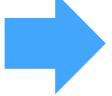
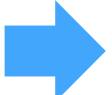


Target label y

- 일관된 레이블 제공 보장

# Data definition

## III Data 문제 분석

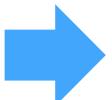
	Unstructured	Structured	
Small data	100개 훈련 데이터에서 스크래치 학습	집의 크기 포함 다른 features를 기반으로 주택 가격을 예측하려는 52개의 구조화 데이터	
Big data	5천만 훈련 데이터에서 음성 인식	백만명의 사용자가 있는 온라인 쇼핑 권장 시스템	

- 깨끗하고 일관된 레이블을 갖는 것이 중요
  - 레이블 지정자 간의 대화를 통해 일관된 레이블링
  - 전체적으로 데이터세트를 확인하고 잘못된 레이블 수정

프로세스에 중심을 두어 진행

# Data definition

## III Data 문제 분석

	Unstructured	Structured	
Small data	100개 훈련 데이터에서 스크래치 학습	집의 크기 포함 다른 features를 기반으로 주택 가격을 예측하려는 52개의 구조화 데이터	
Big data	5천만 훈련 데이터에서 음성 인식	백만명의 사용자가 있는 온라인 쇼핑 권장 시스 템	

깨끗하고 일관된 레이블을 갖는 것이 중요

- 레이블 지정자 간의 대화를 통해 일관된 레이블링
- 전체적으로 데이터세트를 확인하고 잘못된 레이블 수정

프로세스에 중심을 두어 진행

+ 내가 하려는 작업을 확인하고 해당 작업을 진행했던 사람에게 조언듣기

# Data definition

## III

## Label 일관성 개선 방법

여러 레이블 지정자가 동일한 예에 대해 레이블 지정하기

불일치할 경우 레이블 지정자 간 토론을 통해 레이블 정의 약속  
- 계약을 문서화하는 것을 추천-

Input x가 레이블 지정에 있어서 충분한 정보를 포함하지 못할 경우 Input x 수정  
- 어두운 사진이라 식별이 안 되는 경우-

레이블 지정 프로세스를 반복하여 합의 도달

클래스를 병합하여 학습 알고리즘 작업 단순화

불확실성을 포착하기 위한 새로운 클래스/레이블 생성

Ground Truth Label	Inspector(인간 검사관)
1	1
1	0
1	1
0	0
0	0
0	1

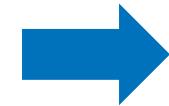


HLP: 66.7%

### 한계

1. 클라이언트는 과연 우리가 만드는 모델이 인간의 성능을 뛰어남을 수 있을까라는 불확실성이 생김
2. 실제로 HLP는 두 사람이 하나의 예제에 대해 동의하는 비율을 나타내는 것
3. 따라서 인간과 기계를 벤치마킹 하는 데 사용하면 이와 같은 부작용 발생

Scratch length	Ground Truth Label	Inspector(인간 검사관)
0.7	1	1
0.2	1 → 0	0
0.5	1	1
0.2	0	0
0.1	0	0
0.1	0	1 → 0



HLP: 100%

### Raising HLP

1. 사람이 지정한 레이블이 GT일 경우, HLP < 100%라면 [모호한 레이블 지정 지침](#)일 수 있음
2. 레이블 일관성을 개선하면 HLP가 높아짐
3. ML이 HLP를 능가하기 어렵게 만들지만 일관된 레이블은 ML 성능을 높여 궁극적으로 실제 애플리케이션에 도움

# Data pipeline

## 데이터 수집

데이터 수집에 많은 시간을 쓰지 않기

사용자 개인 정보를 존중하고 규제 고려 사항 따르기

이미 알고 있는 작업이라면, 데이터 세트 수집에 더 많은 시간을 쓰기 가능  
- 이미 오류 분석을 통해 파악한 정보이기 때문이다-

# Data pipeline

## III 데이터 수집 - 소스 목록 작성

예: 음성 인식 데이터 세트 소스 목록

Source	Amount	Cost(\$)	Time
Owned	100h	0	0
Crowdsourced – Reading	1000h	10000	14 days
Pay for labels	100h	6000	7 days
Purchase data	1000h	10000	1 day

시간이 촉박한 경우,  
이 분석을 기반으로 이미 소유한 데이터와 데이터 구매를 통해 빠르게 진행 가능

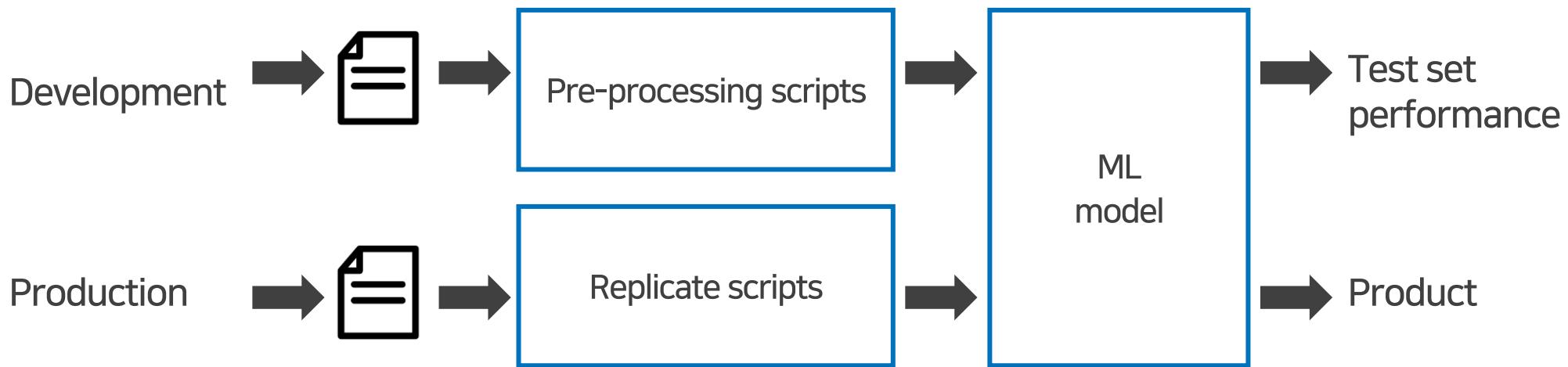
# Data pipeline

## Labeling data

- Labeling 옵션: 회사 내 vs. 아웃 소싱 vs. 크라우드 소싱
- MLE(Machine Learning Engineer)가 직접 labeling: 시간이 오래 걸릴 경우 비효율적
- Labeling 품질
  - 음성 인식: 누구나 오디오를 듣고 텍스트 전환 가능
  - 공장 검사 또는 의료 이미지: SME(Subject matter expert)가 진행
  - 추천 시스템 : 좋은 레이블 할당이 어려움 → 사용자의 구매 데이터를 하나의 레이블로 의존
- 데이터는 한 번에 10배 이상 늘리지 않기
  - 데이터세트 크기가 10배 이상으로 증가할 때 어떤 일이 일어날지 예측하기 어려움
  - 데이터를 무식하게 많이 수집하는 것은 유용한 방법이 아님

# Data pipeline

## III Data preprocessing



동일한 ML 모델에 동일한 입력이어야 하므로 동일한 Pre-processing 진행해야 함  
→ 복제가능성 여부가 매우 중요

# Metadata

## III Metadata 개념 및 특징

### Metadata

#### 개념

- : 데이터에 대한 데이터
- : 데이터 출처와 계보 데이터 저장

#### 메타데이터 정보를 미리 저장

- 오류 분석에 매우 유용
- 예상하지 못한 효과나 태그 또는 비정상적으로 성능이 좋지 않은 데이터 카테고리를 찾아 시스템 개선 가능

# Balanced train/dev/test splits

## Dataset splits

예: Visual inspection example = positive(30개) + negative(70개)

### 1) Random split 경우

- 1) Train:dev:test = 60% : 20% : 20%
- 2) Positive Train:dev:test = 21개:2개:7개
- 3) Positive example rate = 35%:10%:35%
- 4) 불균형!

### 2) Balanced split 경우

- 1) Positive Train:dev:test = 18개:6개:6개
- 2) Positive example rate = 30%:30%:30%
- 3) 균형을 유지하면 ML 프로세스를 크게 개선할 수 있음

**MLOps[1]**  
**Introduction to Machine Learning in Production**

IV

# Scoping

1. Scoping process
2. 실현 가능성

# Scoping Process

## Scoping Process 단계

비즈니스/애플리케이션 문제 브레인스토밍

문제를 해결하기 위한 AI 솔루션 브레인스토밍

잠재적인 솔루션 실현 가능성과 가치 평가

Milestone 정의

Milestone: 프로젝트 진행 과정에서 반드시 거쳐야 하는 중요한 지점

리소스와 예산 책정

# 실현 가능성과 가치 평가

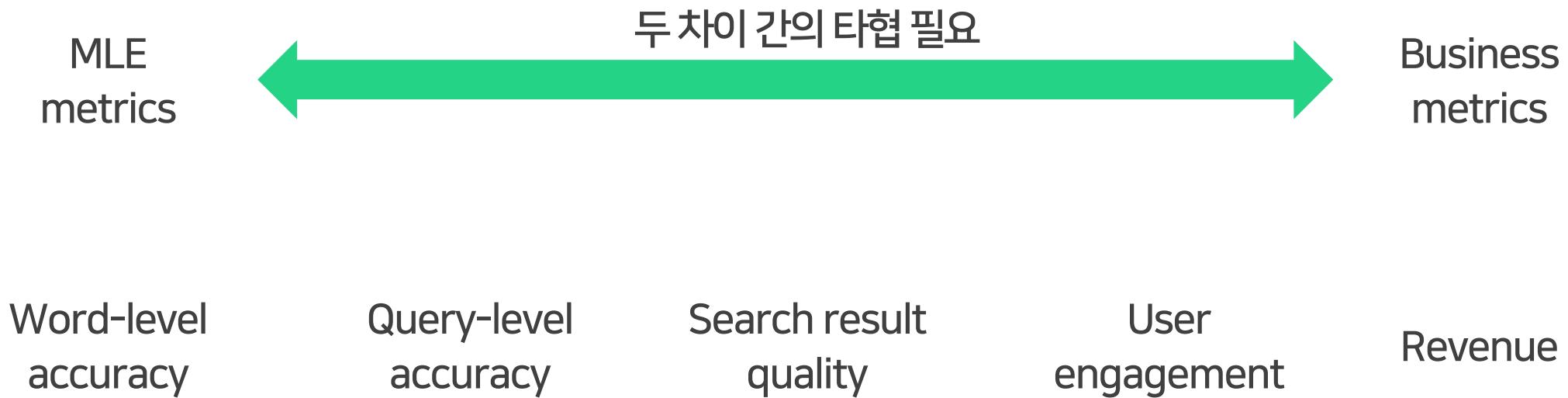
## 실현 가능성

	Unstructured	Structured
New	<p>HLP → 프로젝트 가능 여부 추측 가능</p>	사용 가능한 예측 feature
Existing	<ul style="list-style-type: none"><li>- HLP</li><li>- 기존 기록 확인 → 미래 진행 상황 예측</li></ul>	<ul style="list-style-type: none"><li>- 기존 기록 확인 → 미래 진행 상황 예측</li><li>- 예측을 돋는 feature 추가</li></ul>

# 실현 가능성과 가치 평가

## 가치평가

예: 음성 인식 시스템 구축



# 실현 가능성과 가치 평가

## 윤리적 고려사항

### Ethical considerations

- 프로젝트가 사회적 가치를 만들어 내는가?
- 프로젝트가 공정하고 편향으로부터 자유로운가?
- 윤리적 문제를 공개적으로 토론했는가?

MLOps[2]

Machine Learning Data Lifecycle in Production

I

# Collecting, Labeling and Validating Data

1. Machine Learning Engineering in Production
2. Collecting Data
3. Labeling Data
4. Validating Data

# Machine Learning Engineering in Production

## ML 모델링 vs. Production 모델링

### Research ML

Data

### Production ML

Dynamic – Shifting

Static

Highest overall accuracy

Fast inference, good  
interpretability

Optimal tuning and training

Priority for design

Continuously assess  
and retrain

Very important

Model training

Crucial

High accuracy algorithm

Fairness

Entire system

ML 모델 성능 개선에 집중

ML 개발 + SW 개발

# Machine Learning Engineering in Production

## I ML 개발 vs. SW 개발

### ML 개발

- 데이터 + 예측 품질 관련된 문제에 집중
  - 레이블이 정확한가?
  - 데이터가 존재하는가?
  - Feature vector 차원 축소  
→ 시스템 성능 최적화  
→ 데이터 예측 정보 유지 및 향상
  - 공정성 고려
  - 회귀 클래스에 대해 고려 및 측정

### SW 개발

- Scalability: 규모 확장/축소 가능한가?
- Extensibility: 기능 추가가 가능한가?
- Configuration: 명확하고 잘 정의되었는가?
- Consistency & reproducibility
- Safety & Security
- Modularity
- Testability
- Monitoring
- Best practices

# Collecting Data

## I 데이터 수집 고려 사항

- 데이터와 사용자, 레이블을 적용하는 방법을 이해해야 함
  - 어떤 종류/얼마나 많은 데이터를 사용할 수 있는가?
  - 데이터에 대한 세부정보와 문제점은 무엇인가?
    - 일관성 고려
  - 예측 features는 무엇인가?
    - 데이터 기반 feature 추출
  - 레이블이 무엇인가?
  - 모델 성능을 측정하는데 사용할 메트릭이 무엇인가?
- 데이터 보안 및 개인 정보 보호
  - GDPR과 같은 정책 필요
- 공정성 고려

# Labeling Data

## 데이터 문제

### Data 변화

- 추세 및 계절성
- Feature 분포의 변화
- Feature의 상대적 중요도 변화

### 현실세계의 변화

- 스타일의 변화
- 범위와 프로세스 변경
- 경쟁자의 변화
- 사업 변화

### Data collection 문제

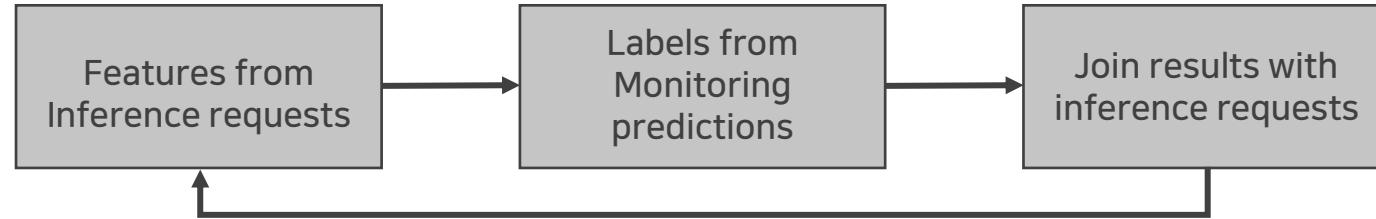
- Bad sensor/camera
- Bad log data
- 이동하거나 비활성화 된 센서/카메라

### System problems

- Bad software update
- 네트워크 연결
- System down
- 잘못된 자격 증명

# Labeling Data

## Process labeling



Process Labeling	
방법	<ol style="list-style-type: none"> <li>예측 요청에서 Feature 추출</li> <li>시스템 모니터링 + 시스템 피드백을 통해 추출한 feature에 대한 labeling</li> <li>기존 input x와 feature 결합</li> </ol> <p>1~3 과정을 반복(강화학습과 비슷한 보상방법)</p>
장점	<ol style="list-style-type: none"> <li>훈련 데이터세트를 지속적으로 생성 가능</li> <li>레이블이 빠르게 진화</li> <li>강한 레이블 신호 포착</li> </ol>
단점	<ol style="list-style-type: none"> <li>문제 고유한 특성으로 인해 방해를 받음</li> <li>대부분 모델에 맞춤형 설계</li> </ol>

# Labeling Data

## I Human labeling

### Human labeling

People (“raters”) to examine data and assign labels manually



평가자가 데이터를 조사하고 레이블을 할당하도록 요청

# Labeling Data

## Human labeling

• • •

Human Labeling	
방법	<ol style="list-style-type: none"><li>레이블이 지정되지 않은 데이터 수집</li><li>평가자 모집</li><li>레이블 지정 가이드 제공</li><li>평가자에게 데이터를 나누고 할당</li><li>평가자 사이 동일한 데이터에 대해 불일치 발생 시, 불일치 해결</li></ol>
장점	<ol style="list-style-type: none"><li>더 많은 레이블</li><li>순수 지도 학습</li></ol>
단점	<ol style="list-style-type: none"><li>품질 일관성 문제</li><li>평가자에게 데이터, 레이블 지침서 제공 → 많은 시간 소요</li><li>인건비</li><li>많은 시간과 비용으로 큰 데이터세트를 얻기 어려움 → 작은 데이터세트로 끝남</li></ol>

# Validating Data

## Data issues

- Drift
  - 시간에 따라 변화되는 데이터
- Skew
  - 서로 다른 소스에서 가져온 데이터 버전의 차이
  - 개념적으로 동일하지만 소스가 다름

# Validating Data

## Skew 종류

I

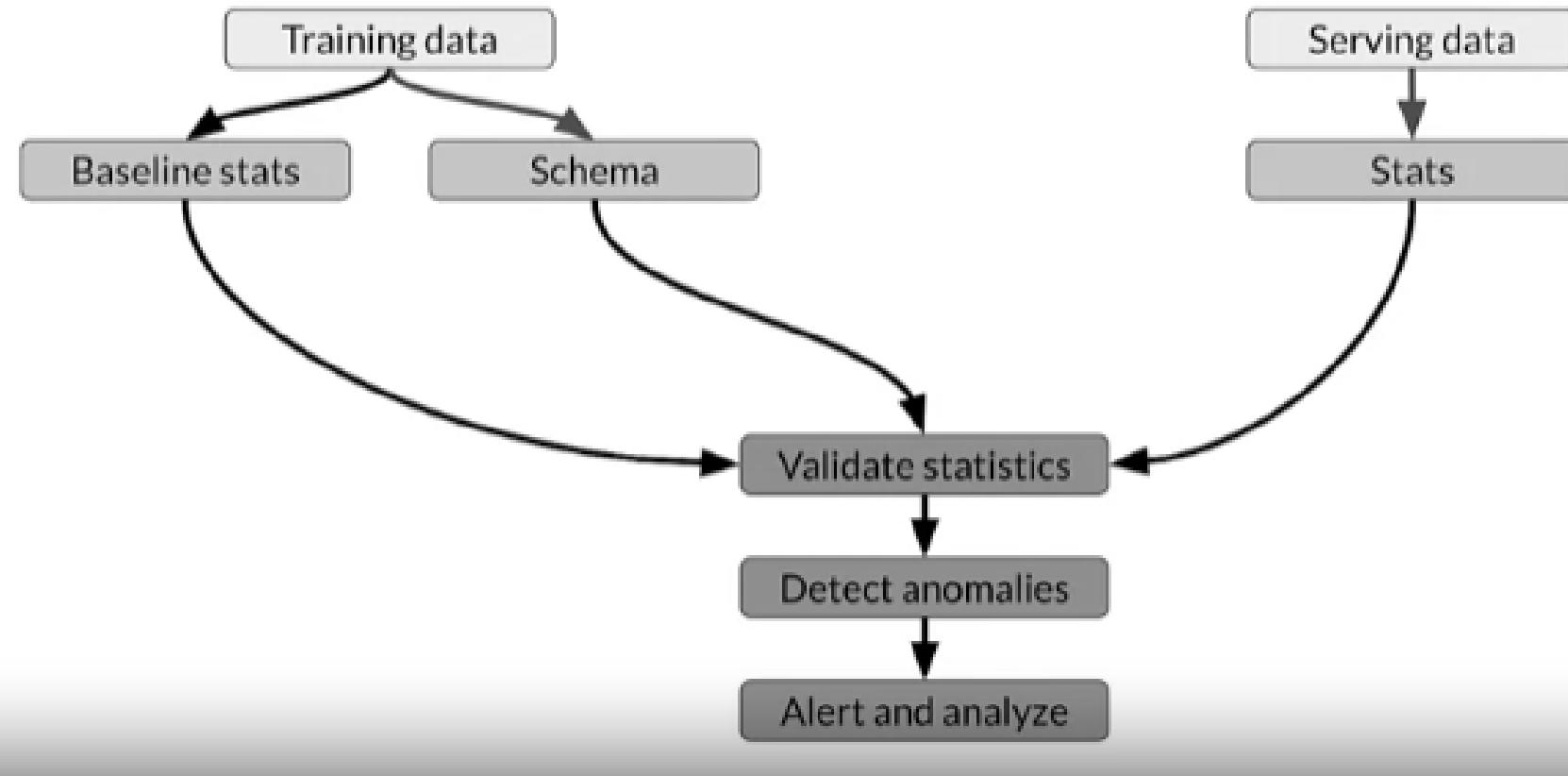
• • •

Schema skew	Feature skew	Distribution skew
제공 데이터와 학습 데이터가 동일한 스키마를 따르지 않을 때 발생	<ul style="list-style-type: none"><li>훈련 데이터와 제공 데이터 사이 feature values 변화</li><li>시스템이 훈련 및 제공 데이터 중 다른 데이터 소스를 사용하거나 상황이 변경될 때 발생</li></ul>	Dataset 개별 feature 분포 변화

# Validating Data

## Skew detection workflow

### Skew detection workflow



# Validating Data

## TensorFlow Data Validation(TFDV)

### TensorFlow Data Validation(TFDV)



- 개발자가 대규모 ML 데이터 이해, 검증 및 모니터링하는데 도움을 줌
- 현재 생산 중인 수백 또는 수천 개의 서로 다른 애플리케이션에서 페타바이트 규모의 데이터 분석 및 검증에 사용
- TFX 사용자가 ML 파이프라인 상태를 유지하는데 도움을 줌

MLOps[2]

Machine Learning Data Lifecycle in Production

III

# Feature Engineering, Transformation and Selection

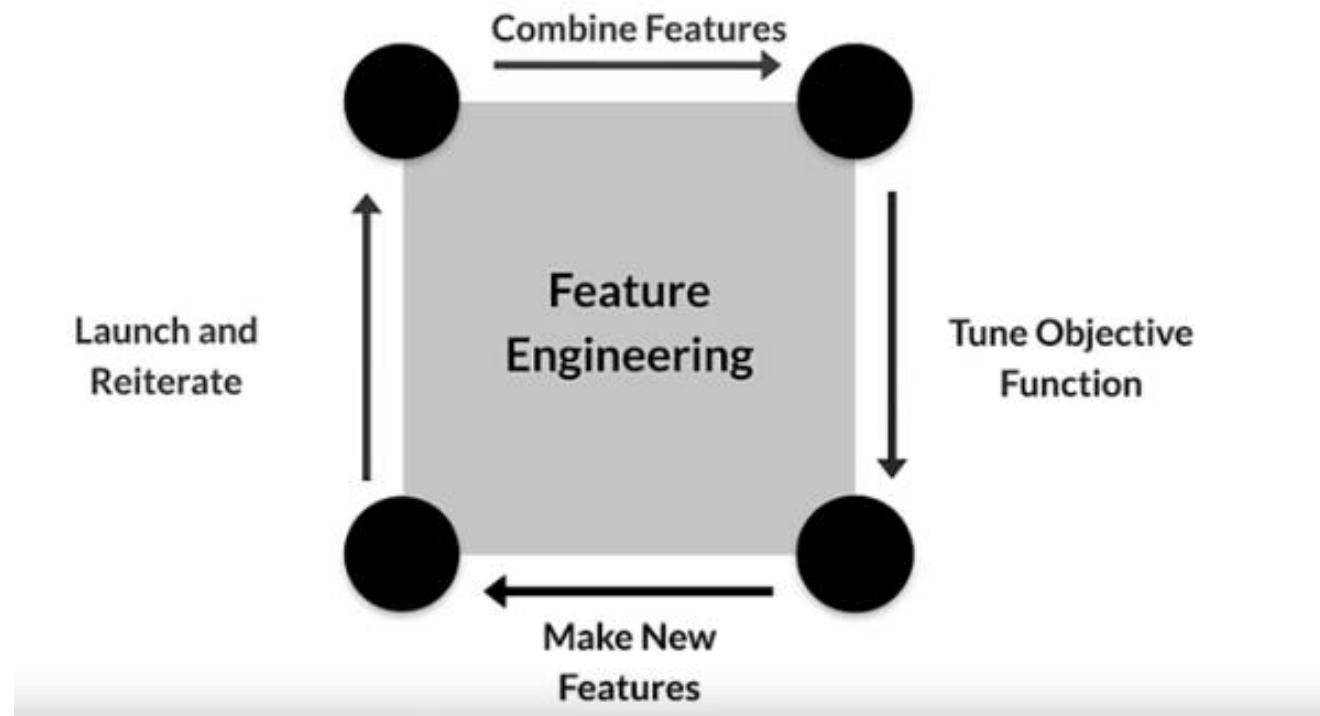
1. Feature engineering
2. Feature engineering techniques
3. Feature selection

# Feature engineering

## 개념 및 특징

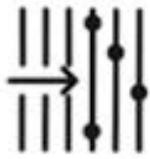
### Feature engineering

: 모델의 학습 능력을 향상시키면서 가능한 컴퓨팅 리소스를 축소



# Feature engineering

## II Preprocessing



Data cleansing



Feature tuning



Representation  
transformation



Feature  
extraction



Feature  
construction

# Feature engineering techniques

## Scaling

### Scaling

- 표준 범위로 변환
- 장점
  - 신경망이 더 빠르게 수렴하도록 유도
  - 훈련 도중 NaN 오류 해결
  - 각 features에 대해 올바른 weights 학습

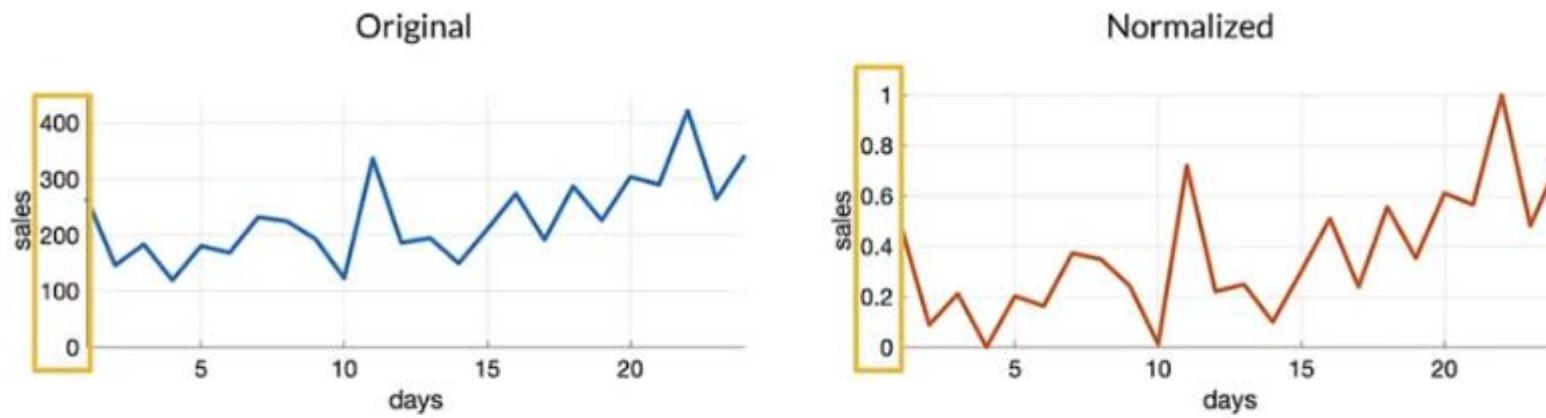
# Feature engineering techniques

## Normalization

### Normalization

$$X_{\text{norm}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

$$X_{\text{norm}} \in [0, 1]$$



# Feature engineering techniques

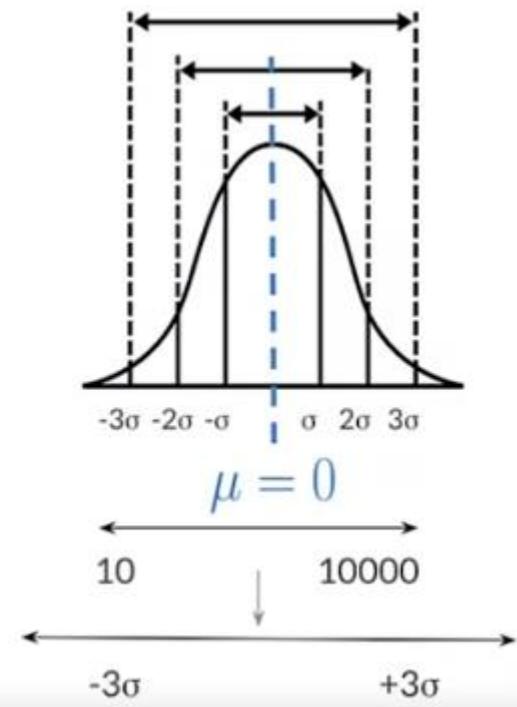
## Standardization(z-score)

### Standardization(z-score)

- Z-score relates the number of standard deviations away from the mean
- Example:

$$X_{\text{std}} = \frac{X - \mu}{\sigma} \quad (\text{z-score})$$

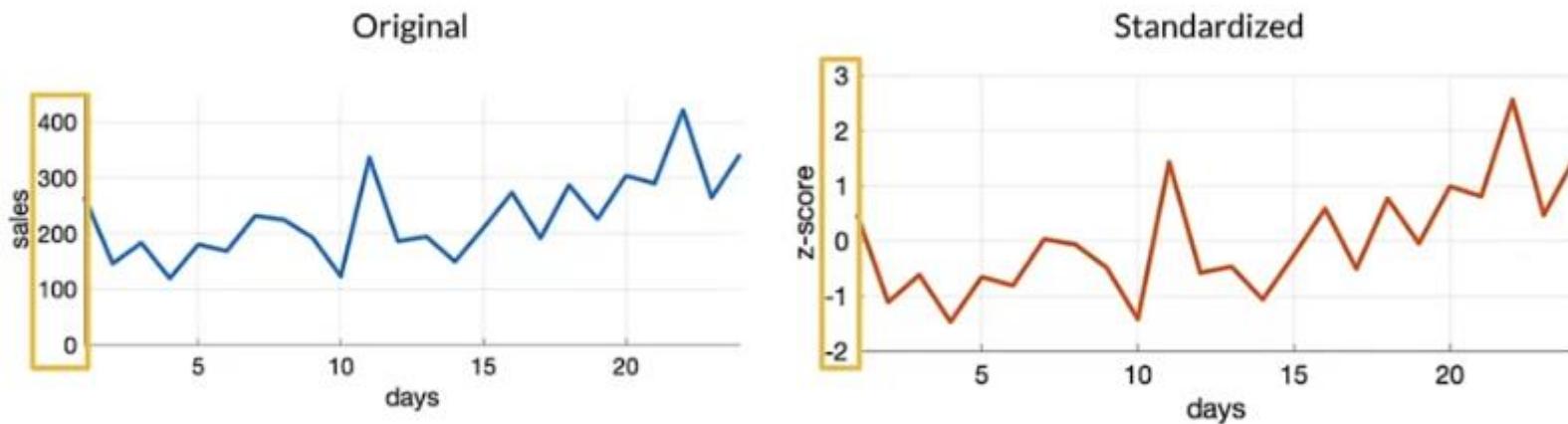
$$X_{\text{std}} \sim \mathcal{N}(0, \sigma)$$



# Feature engineering techniques

## Standardization(z-score)

### Standardization(z-score)

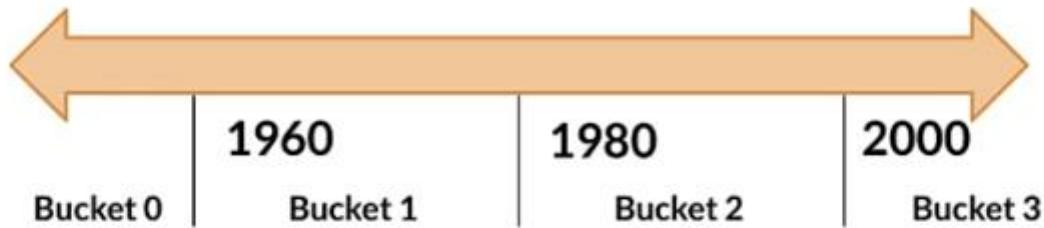


정규화와 표준화를 먼저 수행한 뒤 비교해보고 선택하는 것이 좋음

# Feature engineering techniques

## II Bucketizing/Binning

### Bucketizing/Binning



Date Range	Represented as...
< 1960	[1, 0, 0, 0]
$\geq 1960$ but $< 1980$	[0, 1, 0, 0]
$\geq 1980$ but $< 2000$	[0, 0, 1, 0]
$\geq 2000$	[0, 0, 0, 1]

# Feature Selection

## II Feature 선택 방법

All Features



Feature selection



Useful features



- 관계를 가장 잘 표현하는 feature 식별하기
- 결과에 영향을 주지 않는 feature 제거하기
- Feature space 축소하기
- 리소스 요구사항, 모델 복잡도 줄이기

# Feature Selection

## II Feature 선택 방법

All Features



Feature selection



Useful features



- 관계를 가장 잘 표현하는 feature 식별하기
- 결과에 영향을 주지 않는 feature 제거하기
- Feature space 축소하기
- 리소스 요구사항, 모델 복잡도 줄이기

상관 관계 고려

Feature 1개씩 추가/제거

Feature 중요도 계산



# 감사합니다

Thank you

