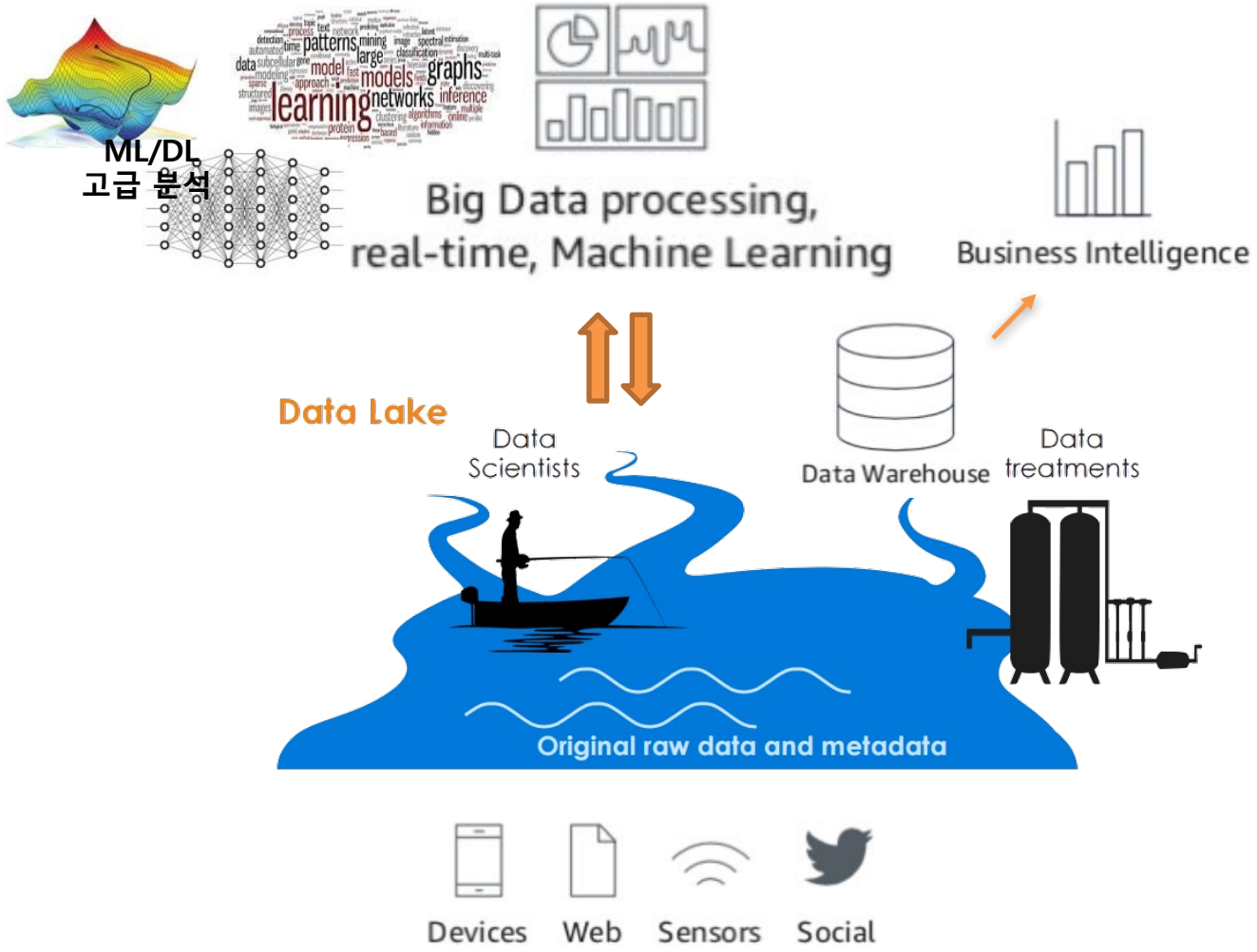
데이터 분석 과정

데이터 분석 절차 및 주요 개념 이해

SK 주식회사 C&C

데이터 분석이란?

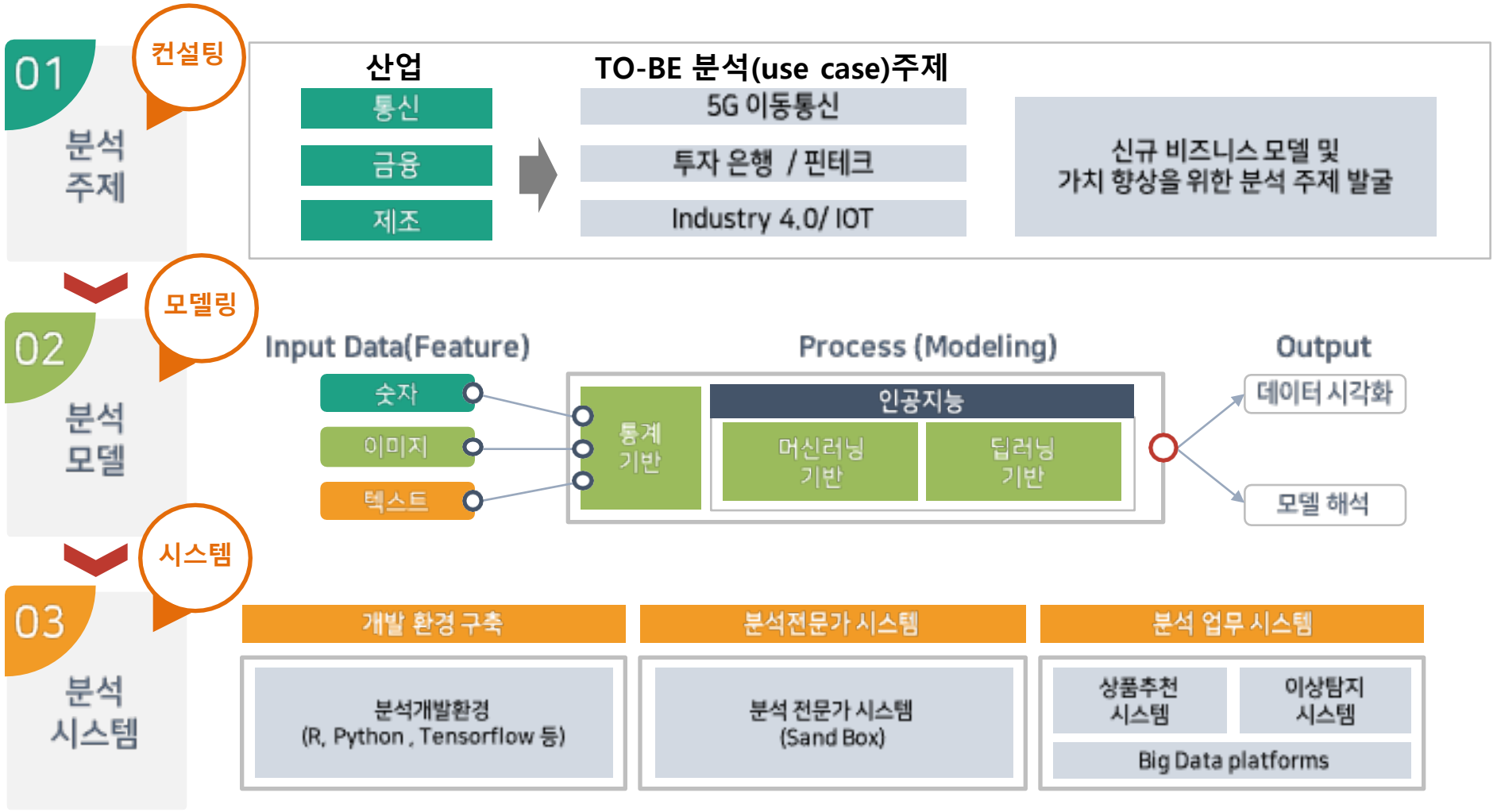
데이터 분석은 대량의 데이터를 분석하여 비즈니스 가치가 있는 정보를 추출하는 것
데이터 유형/기술 발전으로 기존에 다루지 못했던 분석기법의 적용이 가능해짐



- 다양한 유형의 정형, 반정형, 비정형 데이터
- 지속적으로 고도화되는 분석 기법 (ML/DL 고급 분석)
- 분산처리, Cloud, Platform 기술 등 낮은 비용과 빠른 속도로 데이터 저장과 분석이 용이
- 실시간 분석에 대한 요구

분석 업무의 구분

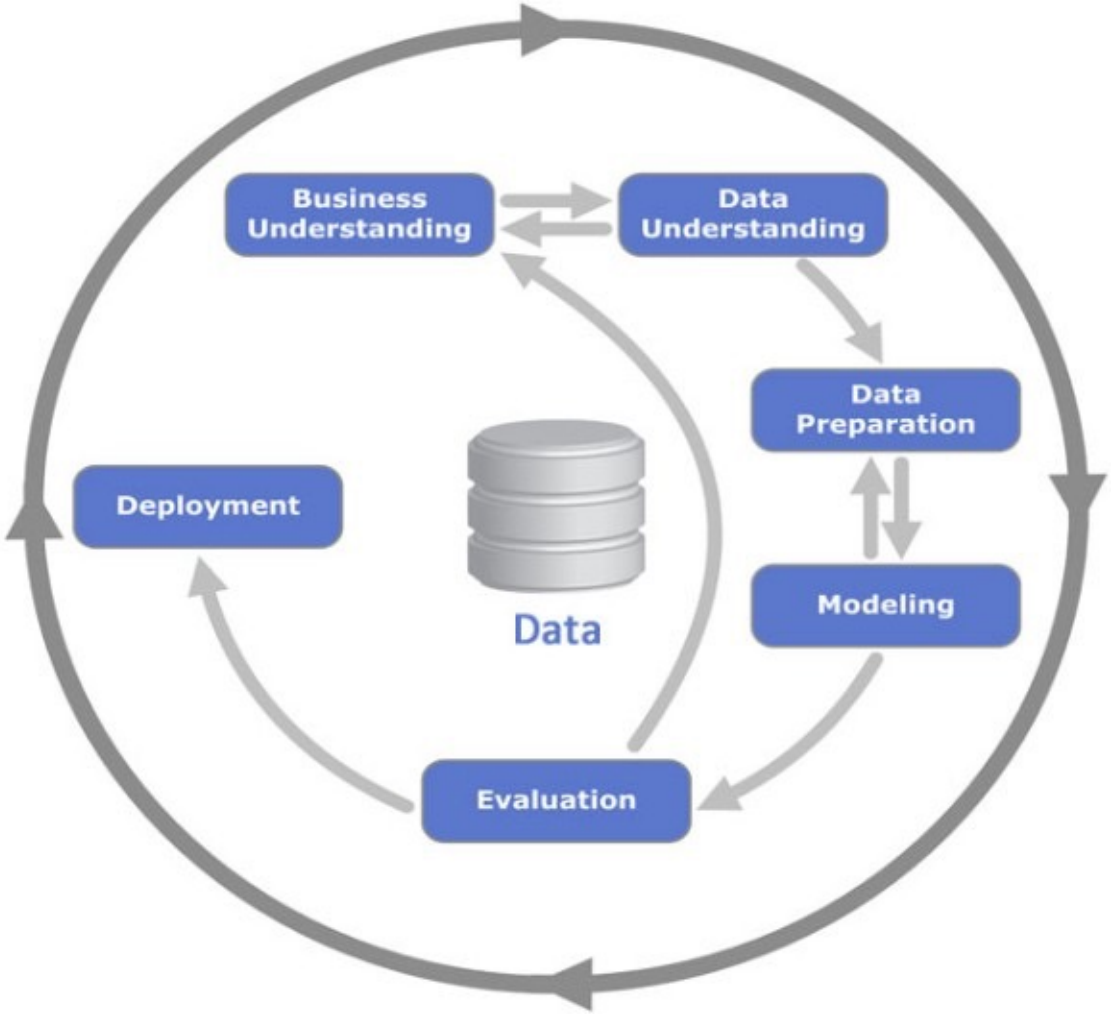
분석 업무는 분석 주제 발굴, 분석 모델링, 시스템 구축 영역으로 구분됨



데이터 분석 절차

CRISP-DM(Cross Industry Standard for Data Mining) Process Diagram

* 통계기반 분석 방법론으로, 업계에서 준용함



Biz. Understanding

비즈니스의 목적과 분석 목표를 수립하고 프로젝트 계획을 수립함
업무 문제를 분석을 위한 문제정의로 전환

Data Understanding

분석에 필요한 초기 데이터를 수집하고 품질을 검토함.(분석용 데이터 확보를 위한 준비 단계)

Data Preparation

데이터를 획득하여 선별,통합,정제 과정을 통해 분석용 Data Set을 편성(EDA, 전처리 등)

Modeling

다양한 분석 기법을 활용하여 최적의 모델을 찾기위한 트레이닝 과정을 반복하고, 테스트 계획에 따라 평가함

Evaluation

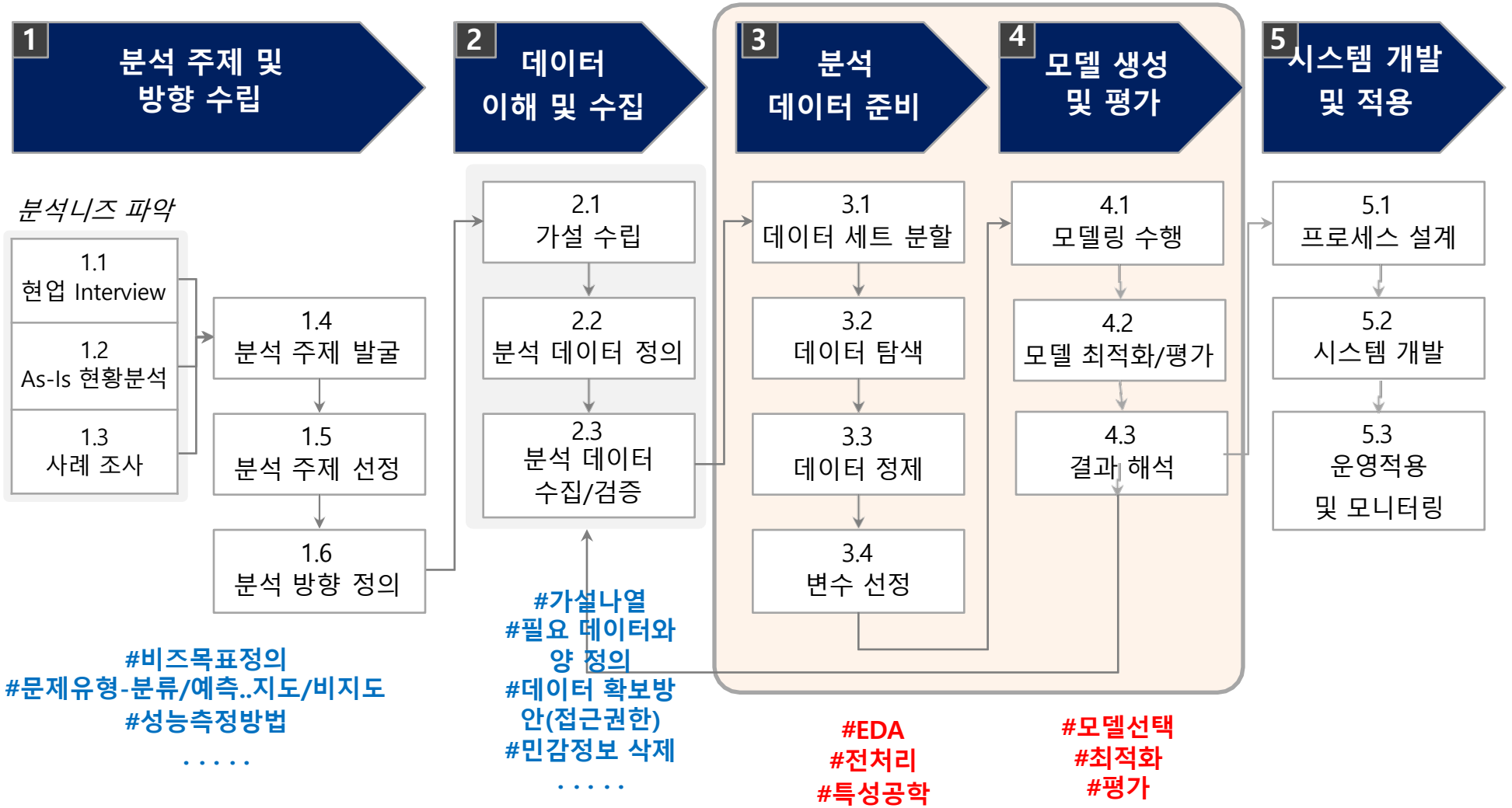
분석 결과를 평가하고 과정을 검토함

Deployment

전개 및 모니터링 계획을 수립하고 과제를 종료함

데이터 분석 절차 (Phase-Process)

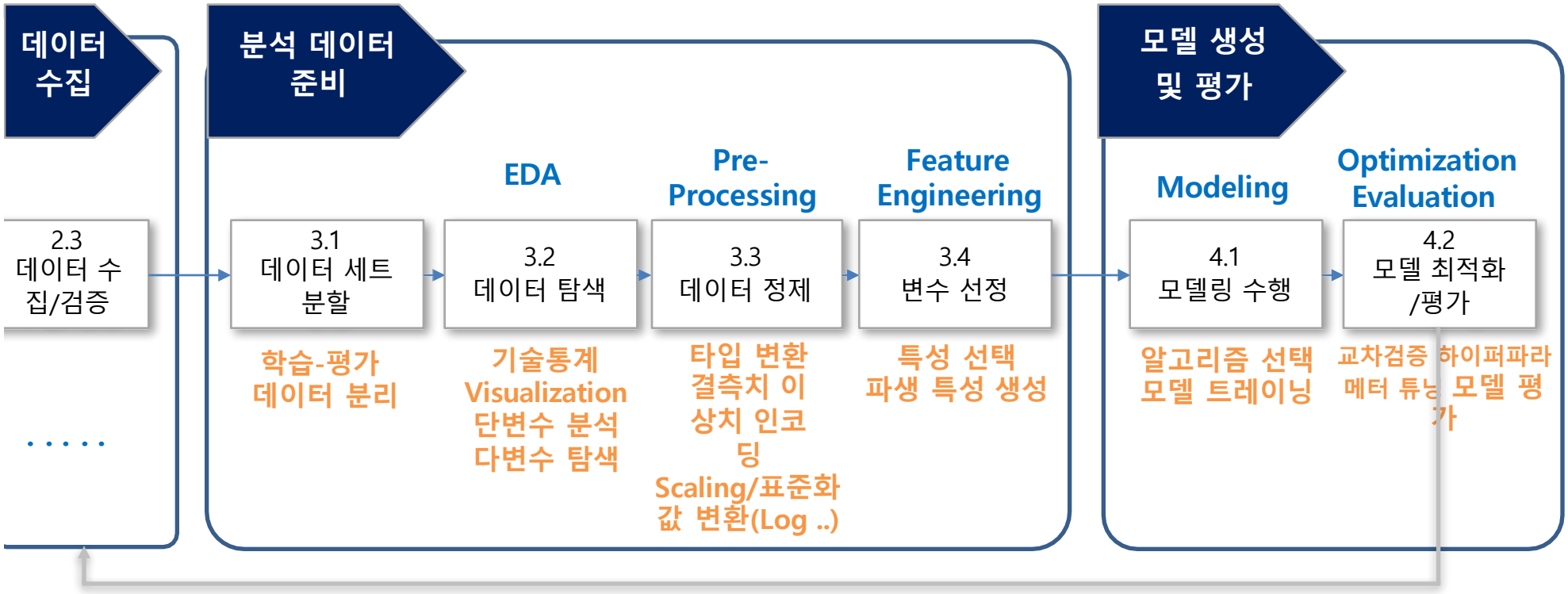
전반적인 데이터 분석절차는 분석기법과 관계 없이 유사하며, [4] 모델 생성 및 평가 부분은 분석기법에 따라 상세 내용이 달라짐



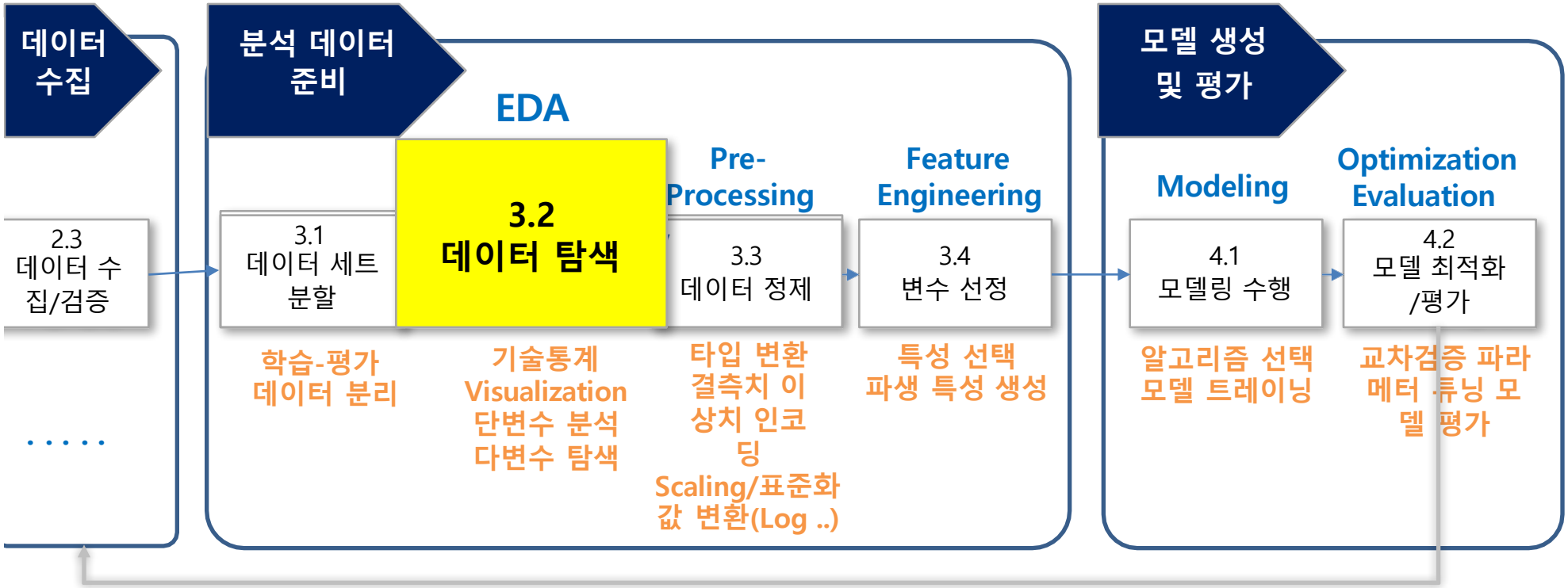
Machine Learning 절차 및 Tasks

Machine Learning은 아래 방식으로 진행됨

- ✓ 수집한 데이터의
- ✓ 탐색적 데이터 분석(EDA-Exploratory Data Analysis)을 통한 이해를 바탕으로,
- ✓ 모델링에 사용할 데이터를 가공하는 전처리(Pre-processing)
- ✓ 모델 트레이닝을 위한 위한 특성을 선택하는 특성 공학(Feature Engineering)
- ✓ 학습 데이터를 기반으로 머신러닝 알고리즘을 적용해 모델을 학습(Modeling) /최적화(Optimization) 하고, 테스트 데이터로 모델에 대한 평가(Evaluation)를 수행



Machine Learning 절차



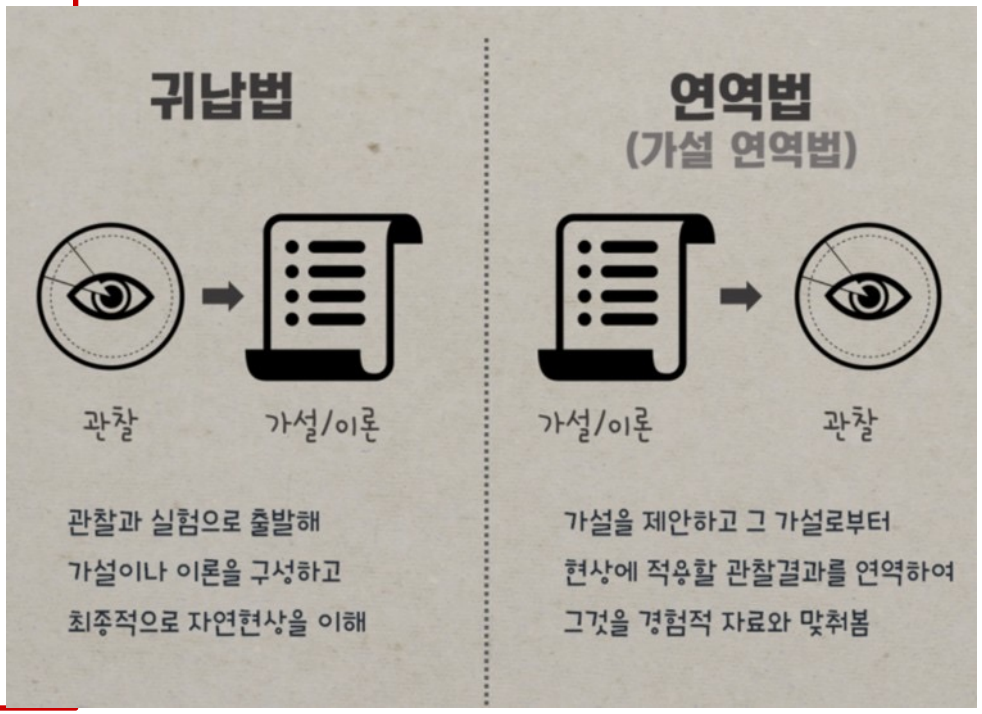
EDA(Exploratory Data Analysis) 개요

탐색적 데이터 분석(EDA)은 분석의 첫 단계로서 데이터의 형태/구조 및 변수간 관계 등을 파악

"데이터가 말하려는 것을 수학이나 그래프를 이용하여 탐색하는 것" *(by Tukey, 1977)*

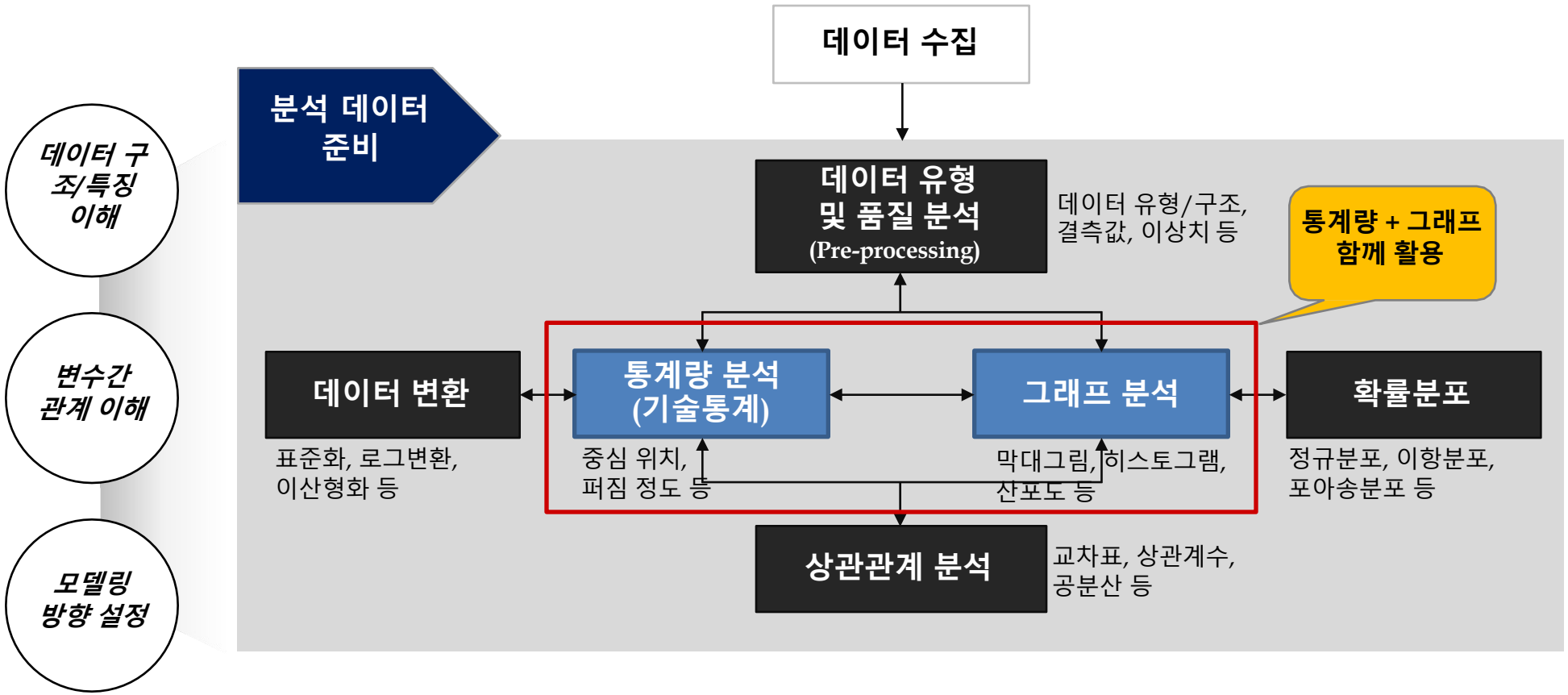
	탐색적 데이터 분석 (EDA: Exploratory Data Analysis)	vs.	확증적 데이터 분석 (CDA: Confirmatory Data Analysis)
정의	• 기 수집된 데이터로부터 데이터의 형태, 관계 파악 • '경찰이 증거를 찾는 일' • 귀납적		• 모집단 추정과 가설의 검정 • '배심원이나 판사가 증거의 강도를 평가하는 일' • 연역적
주요 분석 내용	• 데이터 요약(중앙값, 사분위)와 그래프(산포도 등) • 데이터를 재표현(re-expression, transformation) • 데이터가 어떤 분포에 적합한지 알아보는 방법 (적합성 검증) • 통계적 가설 설정 과정 없음(기술 통계)		
활용	• 기술통계량과 경험(Biz 노하우)에 따른 결론 유추 • 분석의 첫 단계 - 데이터 분포의 적합성 검증 • 통계적 가설이나 모형 수립		

모델링 방향에 대한 감잡기!!



EDA(Exploratory Data Analysis)의 목적

탐색적 데이터 분석을 기반으로 데이터의 특성, 변수간의 관계를 이해하고 모델링 방향을 설정함



탐색적 데이터 분석

EDA
탐색적 데이터 분석
(Exploratory Data Analysis)

=

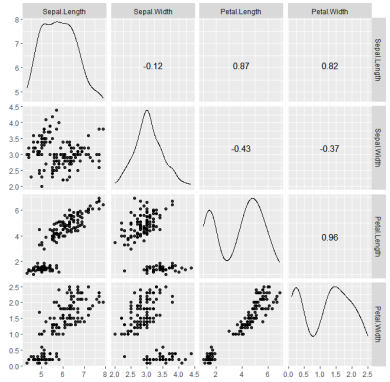
통계량 분석
(기초 통계량)

with

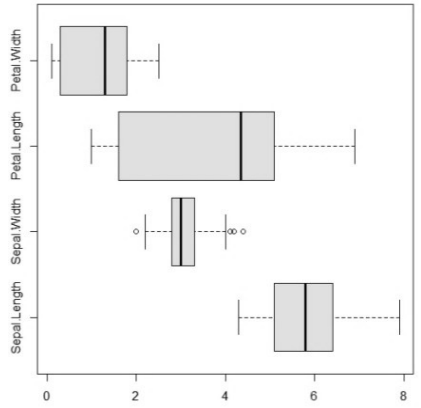
그래프 분석

중심화	평 균: mean( ) 중 앙 값: median( )
퍼짐(분산)	표준편차: std( ) 변동계수: std() / mean() 범위 : range( ) 최 소 : min( ) 최 대 : max( )
분포/대칭	왜 도: skew ( ) 첨 도: kurtosis( )

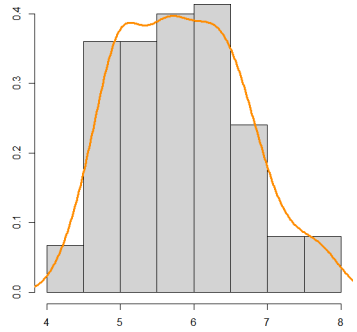
Scatter (Density)



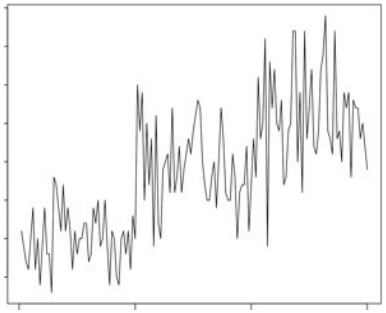
Boxplot



Histogram + Density



Line (Trend)



데이터 탐색의
기본 개념/용어를 이해하기 위해서,

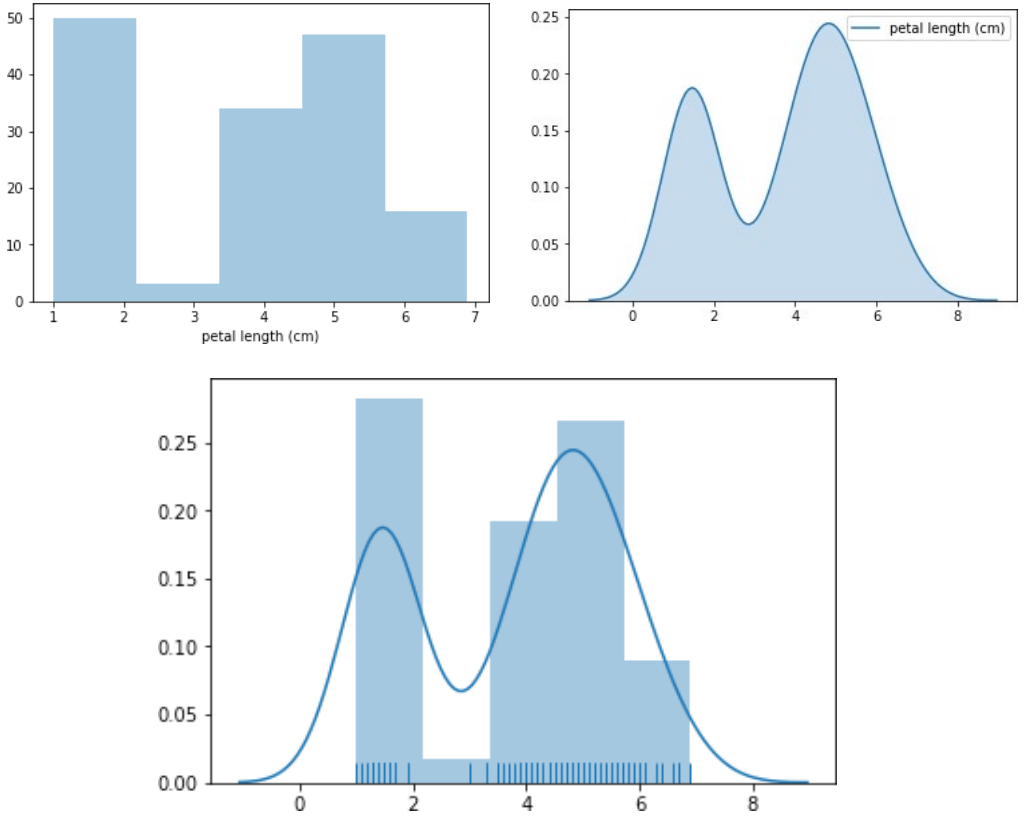
"기술통계 및 그래프 분석" 참고

단변수(단변량) 탐색

한 변수의 값이 가지는 분포를 파악함. 정규분포 가정, 이상치 검출 등에 이용

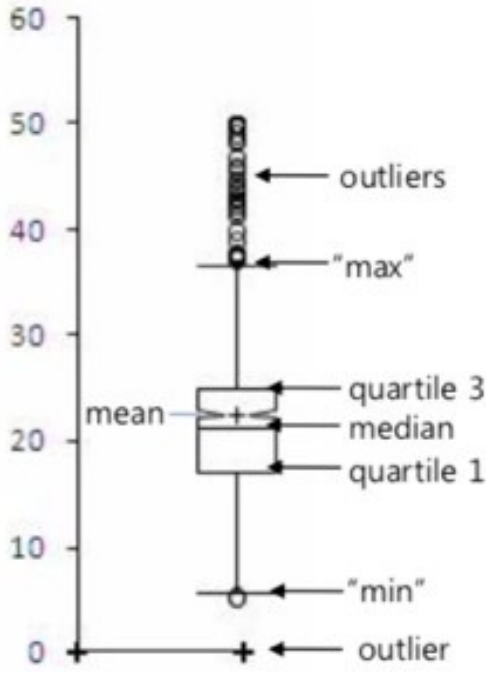
Histogram + Density

- 한 변수의 값이 갖는 분포를 파악
- 정규분포 가정 등에 이용



BoxPlot

- 한 변수의 값이 갖는 분포를 파악
- 사분위수 범위(quartile3-quartile1)의 1.5배 멀리 떨어진 관측치는 이상치로 판단

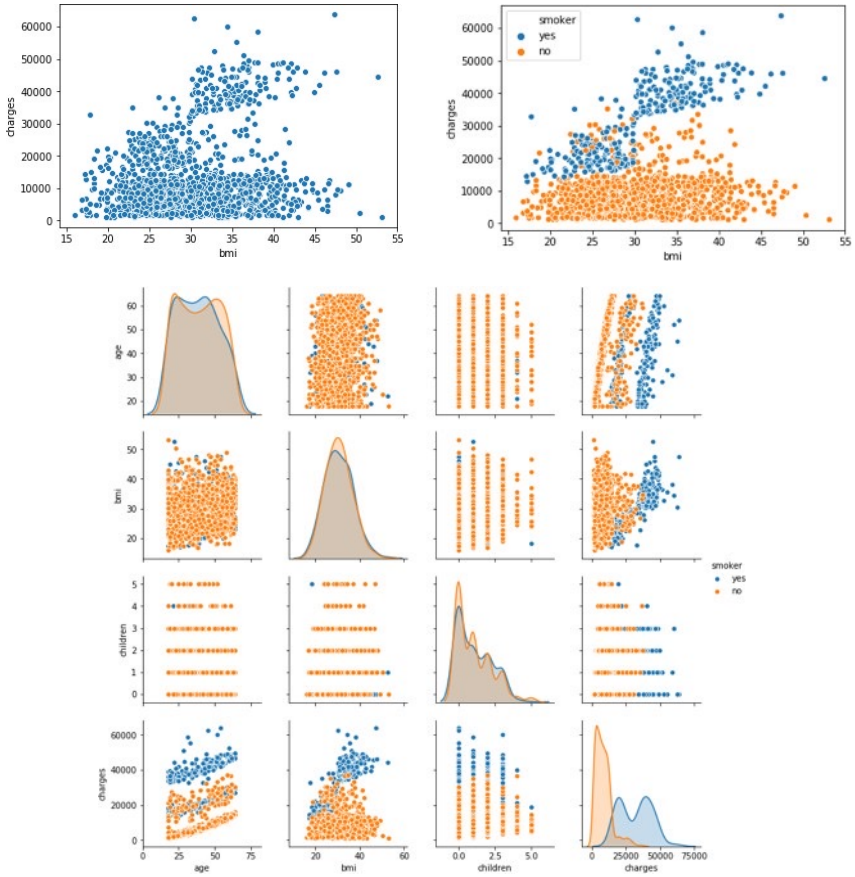


다변수 탐색

이변수 탐색, 다변수 탐색 . 어떤 변수들이 서로 상관관계가 높은지 탐색하여, 대표성을 갖는 적은 수의 변수를 선택할 때 사용 가능 (회귀분석은 X, Y의 선형성 전제)

Scatter

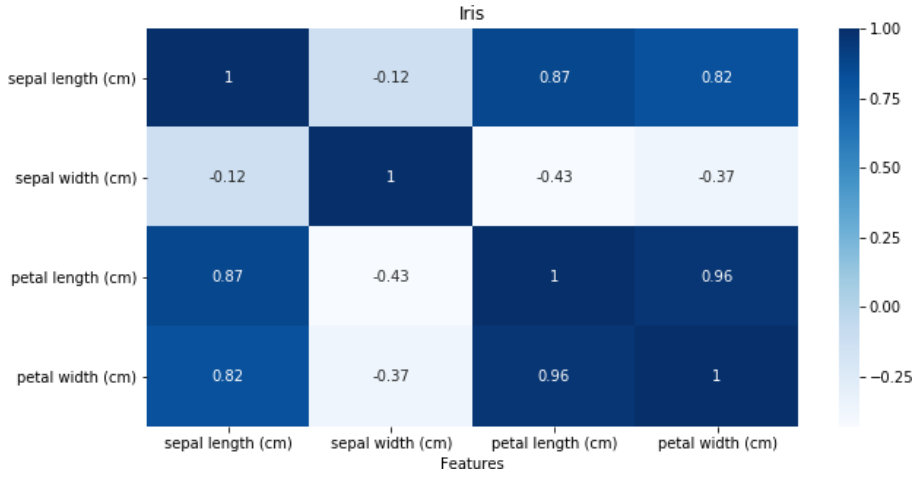
- 두 변수의 상관관계를 파악



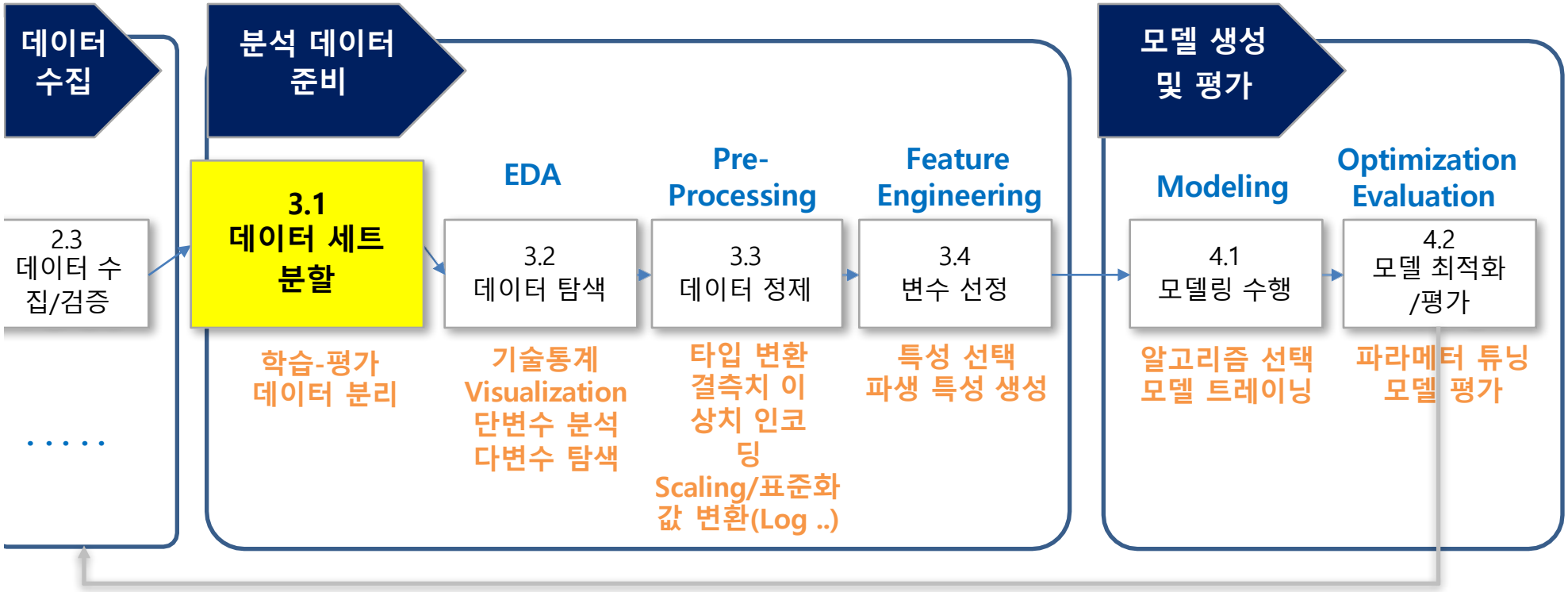
상관계수 + Heatmap

- 데이터 셋이 너무 크지 않을 때는 모든 특성 간의 상관계수(Correlation Coefficient)를 쉽게 확인해 볼 수 있음

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
sepal length (cm)	1.000000	0.742547	0.267176	0.278098
sepal width (cm)	0.742547	1.000000	0.177700	0.232752
petal length (cm)	0.267176	0.177700	1.000000	0.331630
petal width (cm)	0.278098	0.232752	0.331630	1.000000

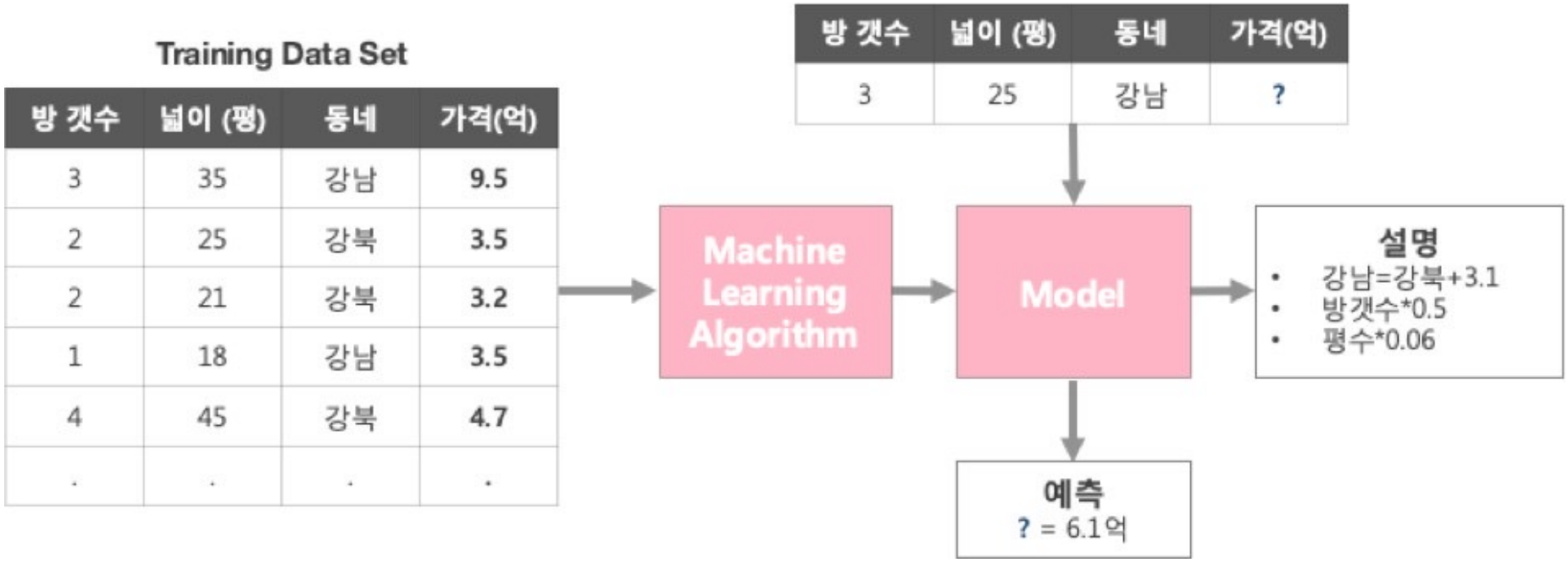


머신러닝 절차



모델

데이터 분석은 데이터 속에 있는 차이를 확인하고 설명(Model)하는 일임
데이터의 차이를 설명할 수 있으면 미지의 입력에 대한 예측/추론도 가능함

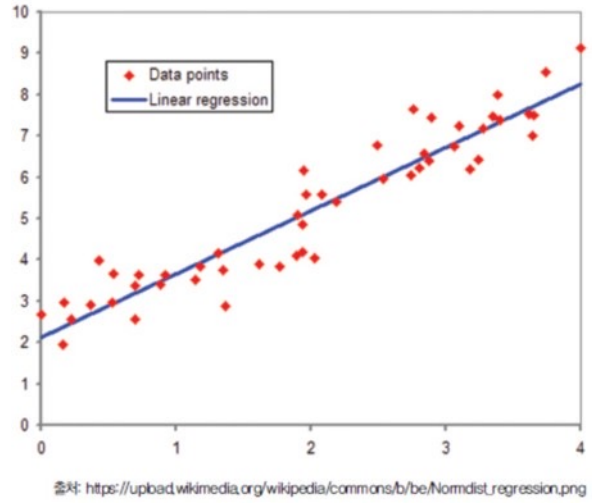


간단한 모델 vs. 복잡한 모델

좋은 모델은 표현력과 유연성(일반화) 사이에 균형이 필요

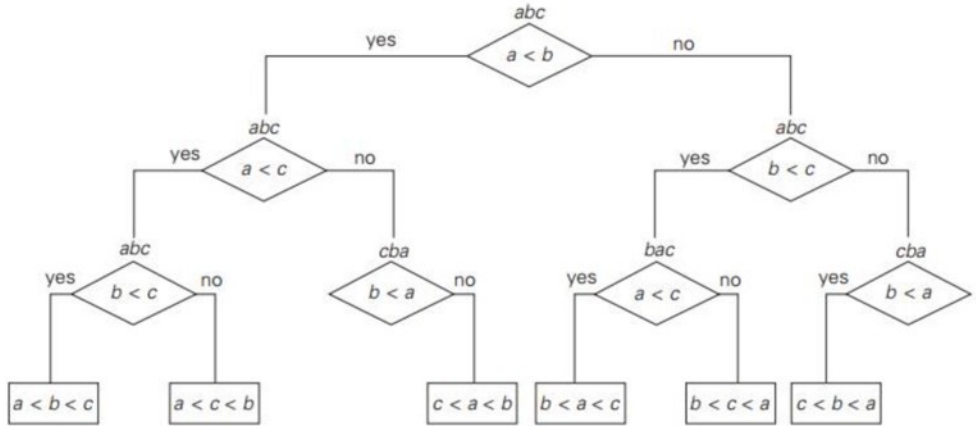
$$y = f(x)$$

모델은 데이터를 설명함(표현)



간단한 모델

- 데이터가 복잡하지 않고 간단할 것이라 가정
- 결과를 이해하기 쉬움
- 학습이 쉬움
- 가정 자체가 강력해서 **모델의 표현력에 제약**



복잡한 모델

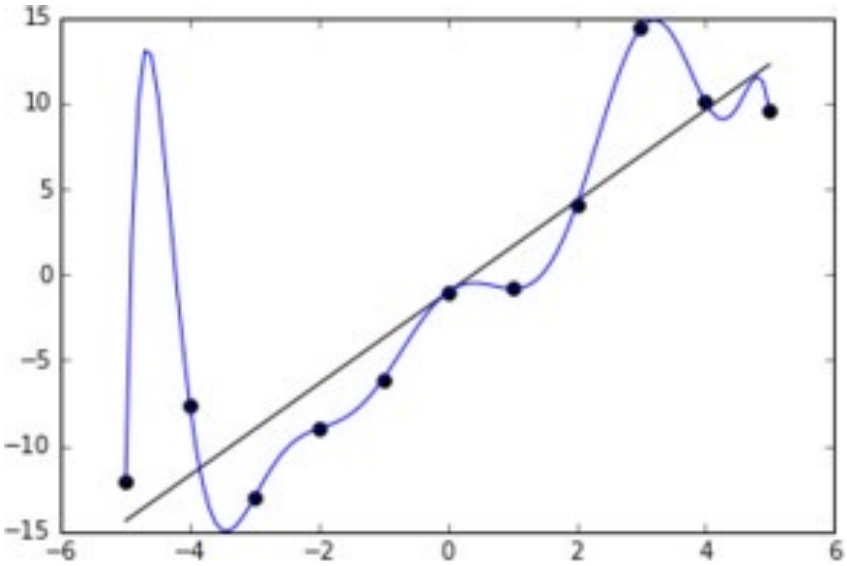
- 데이터가 어떻게(간단?복잡?) 생겼을 거라는 가정 자체가 별로 없음
- 결과를 이해하기 어려울 수 있음
- 학습이 복잡함
- 한정된 데이터를 그대로 학습하여 새로운 데이터에 대해 성능이 떨어질 수 있음(**Overfitting**)

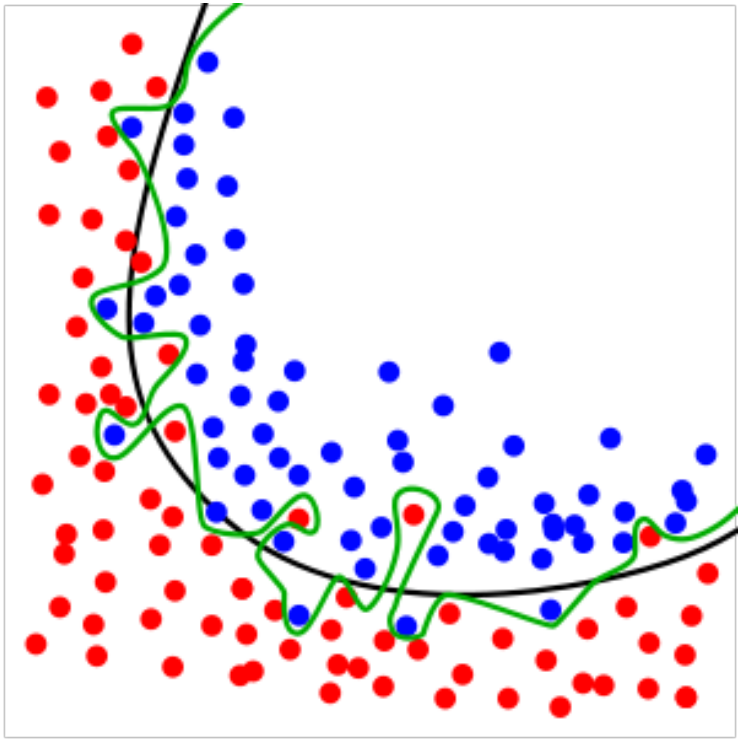
과적합(Overfitting)

머신러닝의 가장 골치 아픈 문제는 단연 과적합

학습데이터에서 완벽하게 작동하는 것 같지만 학습에 사용하지 않은 데이터를 사용하면 잘 작동하지 않는, 일반화를 못하는 문제

모델이 학습용 데이터를 단순히 외운것인지, 아니면 실제로 유용한 패턴을 학습했는지 검증이 필요함





복잡한 모델은 데이터에서 미묘한 패턴을 감지할 수 있지만,
훈련 데이터 셋에 잡음이 많거나, 데이터셋이 너무 적으면(샘플링 잡음 발생)
잡음이 섞인 패턴을 감지하게 됨

과적합(Overfitting) 원인 및 방안

More Data, Simple Model, Generalization performance measure

과적합의 원인 및 증상

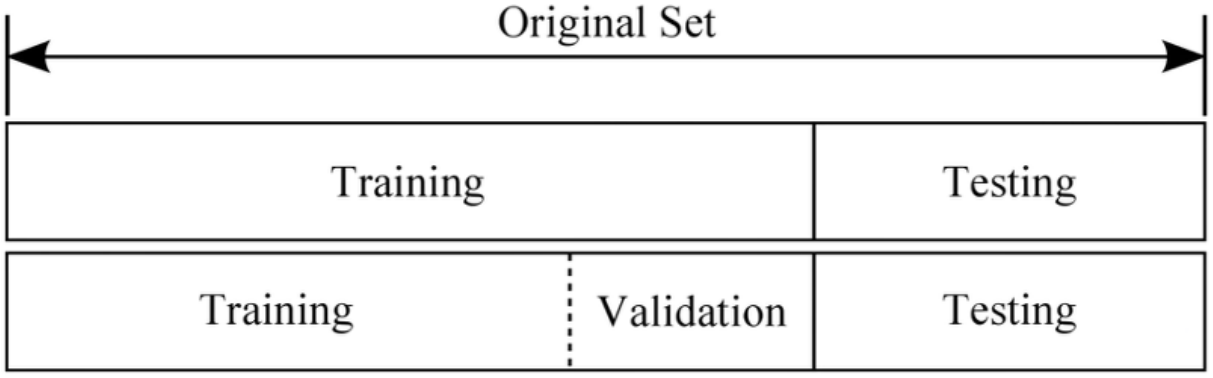
- 부족한 데이터 (insufficient data)
- 소음에 적합 (fit to noise)
- 불필요하게 과도하게 복잡한 모델
- 과적합은 모델이 훈련 데이터 자체를 암기하기 시작할 때 발생
  훈련 데이터에 대해 완벽한 성능
  ➔ Unseen data, New data에 대해서는 성능 폭락

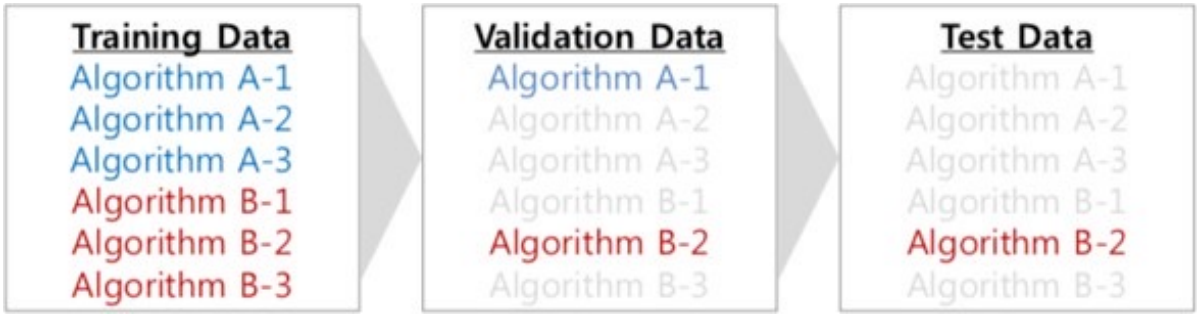
과적합 방지 방안

- 추가 데이터 확보
  - The more, the better ↔ 비용
- 모델 복잡도 통제
  - 변수 선별 (Feature Selection)
  - 변수 차원 축소 (Dimension Reduction)
  - Regularization (Ridge, LASSO)
  - Early stopping, Pruning
- 모델 일반화(Generalization) 성능 평가
  - Unseen data를 가지고 모델 성능 Test
  - Cross-validation
    ➔ 일반화 성능 좋은 모델 parameter 선정

학습-평가 데이터 나누기

과적합(Overfitting)을 방지하기 위해서 Training에 사용하는 학습 데이터와 모델의 성능을 검증하기 위한 평가(Test) 데이터를 철저히 분리하여 검증해야 함





학습데이터(Training Set)

✓ 모델을 학습하는데 사용

검증 데이터(Validation Set)

✓ 모델의 최적 파라메터를 선택하는데 사용

테스트 데이터(Test Set)

✓ 새로운 데이터를 적용하여 모델의 실제 예측력을 평가하는데 사용

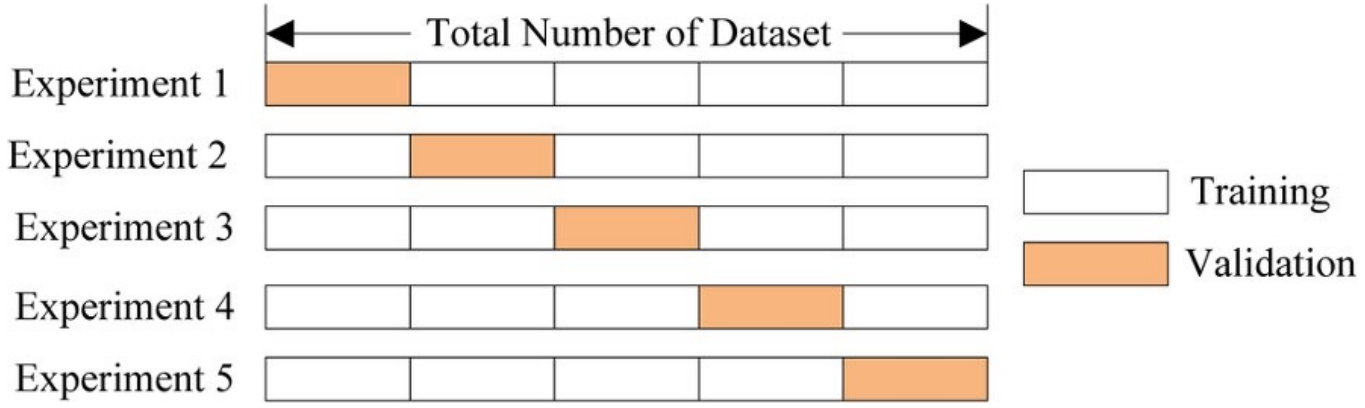
교차검증

데이터 수가 적은데 검증과 테스트에 데이터를 더 뺏기면 성능이 미달되는 모델이 학습됨.
총 데이터 갯수가 적은 데이터 셋에 대해서 교차검증을 적용하여 정확도를 향상시킬 수 있음

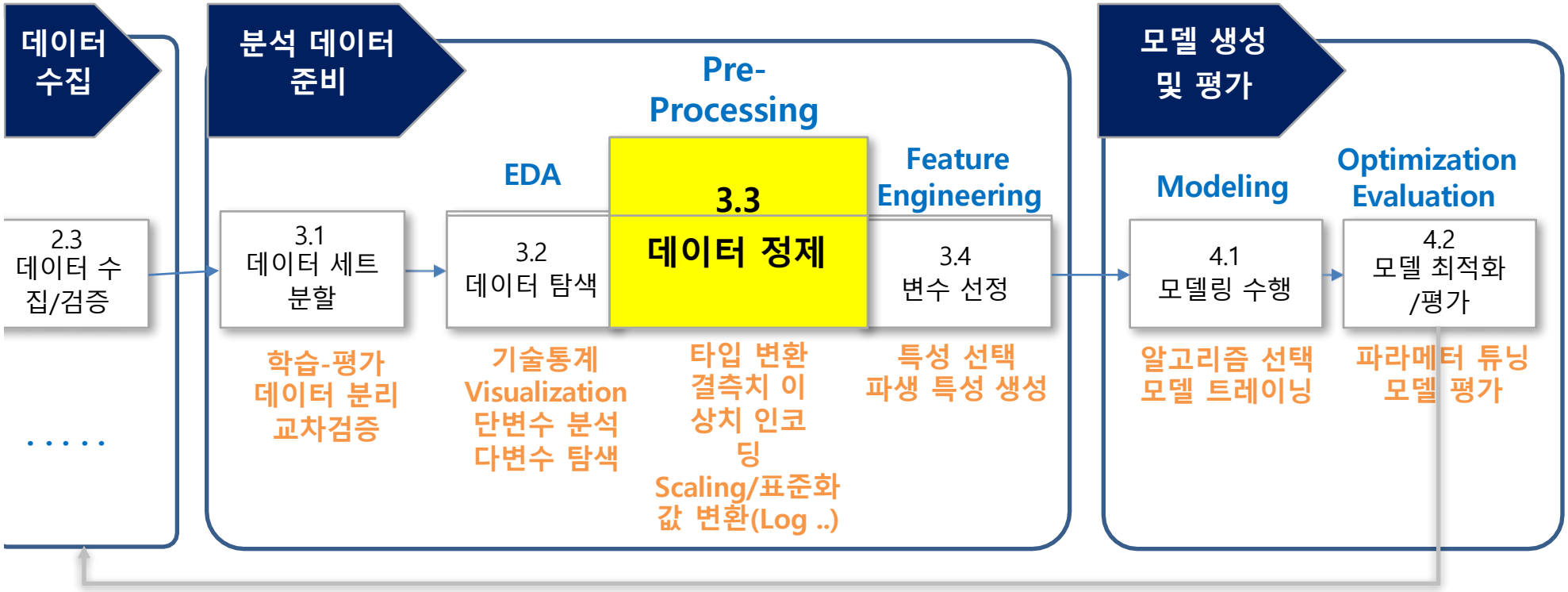
K겹 교차검증(K-Fold Cross Validation)

- ✓ 데이터셋을 K개로 나눕니다.
- ✓ 그 중 첫번째 세트를 제외하고 나머지에 대해 모델을 학습합니다. 그리고 첫번째 세트를 이용해서 평가를 수행합니다. (다음 세트를 이용하여 이 과정을 반복)
- ✓ 각 세트에 대해 구한 평가 결과의 평균을 구함

※ 교차검증은 여러 부분을 학습과 평가로 사용한 결과로 일반화 특성을 평가하므로 더 안정적이고 정확함. 하지만 여러번 학습하고 평가하는 과정을 거치기 때문에 계산량이 많음



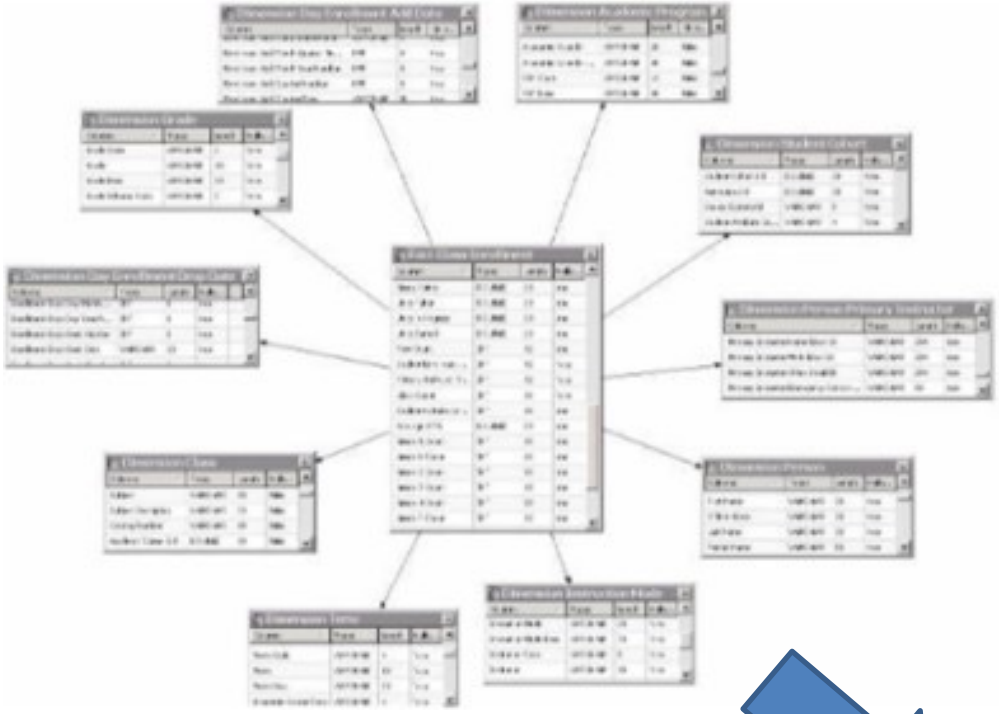
Machine Learning 절차



데이터 준비 - 전처리

원본 데이터를 분석에 사용하기 좋은 형태로 바꾸는 일.

낮은 품질의 데이터는 나쁜 모델을 만든다.



머신러닝 알고리즘이 처리 가능하고 성능을 높일 수 있는 형태로(품질,형식) 데이터를 준비하는 작업

- ✓ 변수 타입 변환
- ✓ 결측치 처리(삭제, 대체, 예측)
- ✓ 이상 데이터(Outlier) 처리
- ✓ 데이터 인코딩(Label encoding, One-Hot Encoding 등)
- ✓ 연속형 ➔ 범주화(Binning)
- ✓ 로그, 제곱근 변환 등

변수 타입 변환

데이터 의미에 맞도록 데이터 유형을 변환함

- ✓ **명목형 변수는 object 또는 String으로,**
- ✓ **수치형 변수는 int64 또는 float64 으로 변환**

예시) Titanic Data Set에서 Survived(생존여부) 와
Pclass(티켓의 클래스)는 명목형 변수(라고 가정한다면,) 이므로 "Object"로 형식 변환

- Survived : 0, 1
- Pclass : 1=1st, 2=2nd, 3=3rd

```
titanic.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId     891 non-null int64
Survived        891 non-null int64
Pclass          891 non-null int64
Name            891 non-null object
Sex             891 non-null object
Age             891 non-null float64
SibSp           891 non-null int64
Parch           891 non-null int64
Ticket          891 non-null object
Fare            891 non-null float64
Cabin           204 non-null object
Embarked        889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

단순 코드의 의미를 가지는 레이블이 크기가 의미를 가지는 숫자로 사용될 경우, 선형 회귀와 같은 알고리즘에 이를 적용하면 예측 성능이 떨어지는 경우가 있음
(크고 작음에 대한 특성이 반영)

```
titanic['Survived'] = titanic['Survived'].astype(object)
titanic['Pclass'] = titanic['Pclass'].astype(object)
```

결측치(Null Value) 처리

존재하지 않거나 관측되지 않은 값(운영/휴먼 에러). 대부분의 ML 알고리즘은 데이터에 Null 값을 허용하지 않음(연산불가). 따라서, 해당 행을 삭제하거나 다른 값으로 대체함

- ✓ 소수일 경우 평균값, 중앙값, 최빈값, 근사치 등으로 대치
- ✓ 대다수 일 경우는 해당 특성 자체를 Drop

【 NA 확인 】

> head

	PassengerId	Survived	Pclass	Ticket	Fare	Cabin
0	1	0	3	A/5 21171	7.2500	NaN
1	2	1	1	PC 17599	71.2833	C85
2	3	1	3	STON/O2. 3101282	7.9250	NaN
3	4	1	1	113803	53.1000	C123
4	5	0	3	373450	8.0500	NaN

> isna().sum()

```
1 df_titanic.isna().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

【 NA 처리 】

1 분석 제외

※ 비즈니스 관점에서 제외해도 되는지 반드시 검토 필요

- NA인 행을 제거
- 대다수의 값이 NA이고, 분석영향도가 적다면 특성 제거

2 다른 값으로 대체

- 제외되는 데이터 유실을 최소화해야 할 경우
- 시계열과 같이 데이터 흐름을 반영하여 분석해야 할 경우
- 대표값 또는 트랜드값으로 대체
  ➔ **0, 평균, 유사 개체값** 등

[주의] NA 처리 시

- **정보손실, 왜곡 이슈 잔존**
- **전처리 항목/내용에 대해 반드시 공유!**

이상치(Outlier)

대다수의 데이터와 달리 분포에서 비정상적으로 벗어난 값

값은 존재하되 실질적으로 가능하지 않은 값(나이 990살, 키 30cm)
이상치는 평균 등 중심값 수치를 왜곡시킴
기술통계량 만으로는 이상치 판별이 쉽지 않고, 통상 Visualization을 통해서 이상치를 판별함

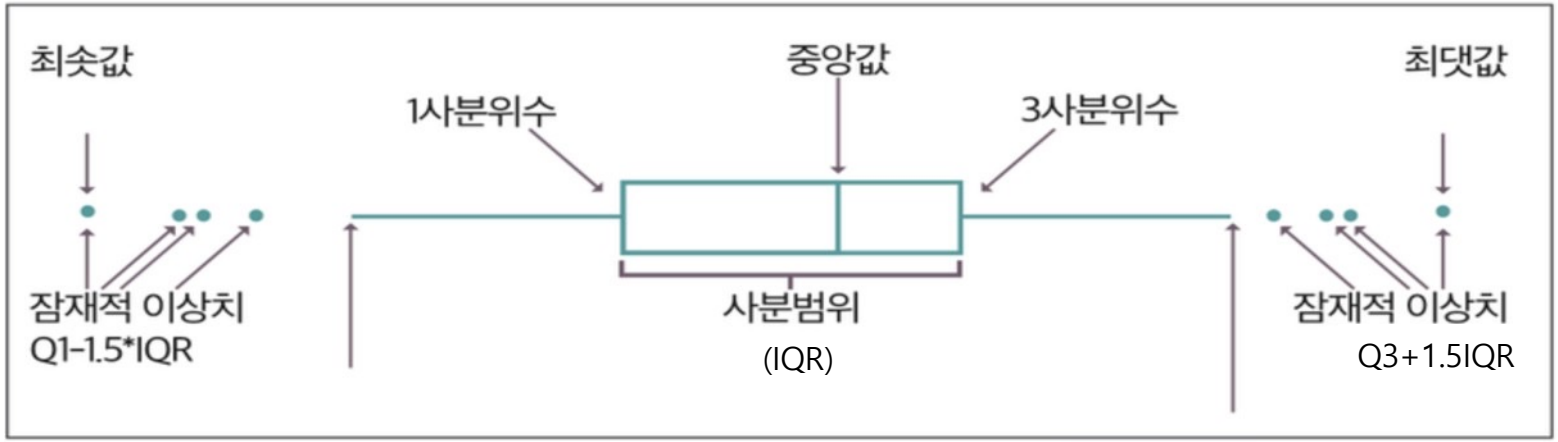
해결방안

- ✓ 레코드 수가 충분히 많으면 제거
- ✓ 레코드 수가 충분치 않으면 합리적인 값으로 대치

주의

- ✓ 이상치 처리 시 비즈니스 관점에서 필요한 데이터인지 반드시 확인 필요

Ex) Fraud Detection 분석과제



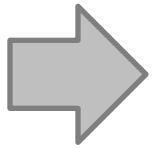
Boxplot을 이용한 이상치 확인

데이터 인코딩 – 레이블 인코딩

머신러닝 알고리즘은 문자열 값을 입력 값으로 허용하지 않음.
(연산에 사용할 수 있도록) 숫자형으로 변환해야 함

원본데이터

상품 분류	가격
TV	1,000,000
냉장고	1,500,000
전자렌지	200,000
컴퓨터	800,000
선풍기	100,000
선풍기	100,000
믹서	50,000
믹서	50,000



상품분류 레이블 인코딩 데이터

상품 분류	가격
0	1,000,000
1	1,500,000
4	200,000
5	800,000
3	100,000
3	100,000
2	50,000
2	50,000

```
1  from sklearn.preprocessing import LabelEncoder
2
3  items=['TV','냉장고','전자렌지','컴퓨터','선풍기','선풍기','믹서','믹서']
4
5  # LabelEncoder를 객체로 생성한 후 , fit( ) 과 transform( ) 으로 label 인코딩 수행.
6  encoder = LabelEncoder()
7  encoder.fit(items)
8  labels = encoder.transform(items)
9  print('인코딩 변환값:',labels)
```

```
인코딩 변환값: [0 1 4 5 3 3 2 2]
```

주의

단순 코드의 의미를 가지는 레이블이 크기가 의미를 가지는 숫자로 변경되므로

선형회귀와 같은 알고리즘에 이를 적용할 경우 예측 성능이 떨어지는 경우가 있음(크고 작음에 대한 특성이 반영)

➔ 원-핫 인코딩 (One-Hot Encoding) 방식 사용

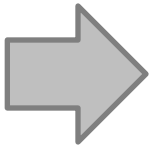
데이터 인코딩 – 원-핫 인코딩(One-Hot Encoding)

문자열 값을 숫자형으로 변환하기 위한 방법 중 하나로,

행 형태로 표현된 특성의 고유값을 열 형태로 차원을 변환하고 해당 컬럼만 1로 표시하고 나머지 컬럼은 0으로 표시하는 방식 (Dummy Coding)

원본데이터

상품 분류
TV
냉장고
전자렌지
컴퓨터
선풍기
선풍기
믹서
믹서



One-Hot Encoding

상품분류_ TV	상품분류_ 냉장고	상품분류_ 전자렌지	상품분류_ 컴퓨터	상품분류_ 선풍기	상품분류_ 믹서
1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	1	0
0	0	0	0	0	1
0	0	0	0	0	1

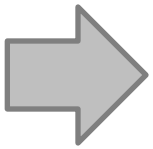
```
1  import pandas as pd
2
3  df = pd.DataFrame({'item':['TV','냉장고','전자렌지','컴퓨터','선풍기','선풍기','믹서','믹서'] })
4  pd.get_dummies(df)
```

데이터 인코딩 – 원-핫 인코딩(One-Hot Encoding)

상품명, 상품 카테고리, 성별과 같은 비수치적 데이터 사이의 유사도 계산에 원-핫 인코딩(Dummy coding) 방식을 이용할 수 있음

사용자 상품 구매내역

사용자1	상품 A	상품 C	상품 D
사용자2	상품 A	상품 B	
사용자3	상품 A	상품 C	



One-Hot Encoding

사용자	상품 A	상품 B	상품 C	상품 D
사용자1	1	0	1	1
사용자2	1	1	0	0
사용자3	1	0	1	0

사용자 1과 누가 더 유사한지를 알아보기 위해서 제곱유클리드 거리를 계산함

$$d(\text{사용자1}, \text{사용자2})^2 = (1-1)^2 + (0-1)^2 + (1-0)^2 + (1-0)^2 = 3$$
$$d(\text{사용자1}, \text{사용자3})^2 = (1-1)^2 + (0-0)^2 + (1-1)^2 + (1-0)^2 = 1$$

즉, 사용자1과 사용자3이 비슷한 상품을 구매하므로 같은 클러스터에 속할 가능성이 높다고 할 수 있음
이처럼 카테고리를 원-핫 인코딩 방식을 이용하여 수치 데이터로 변환하면 유사도 계산을 할 수 있음

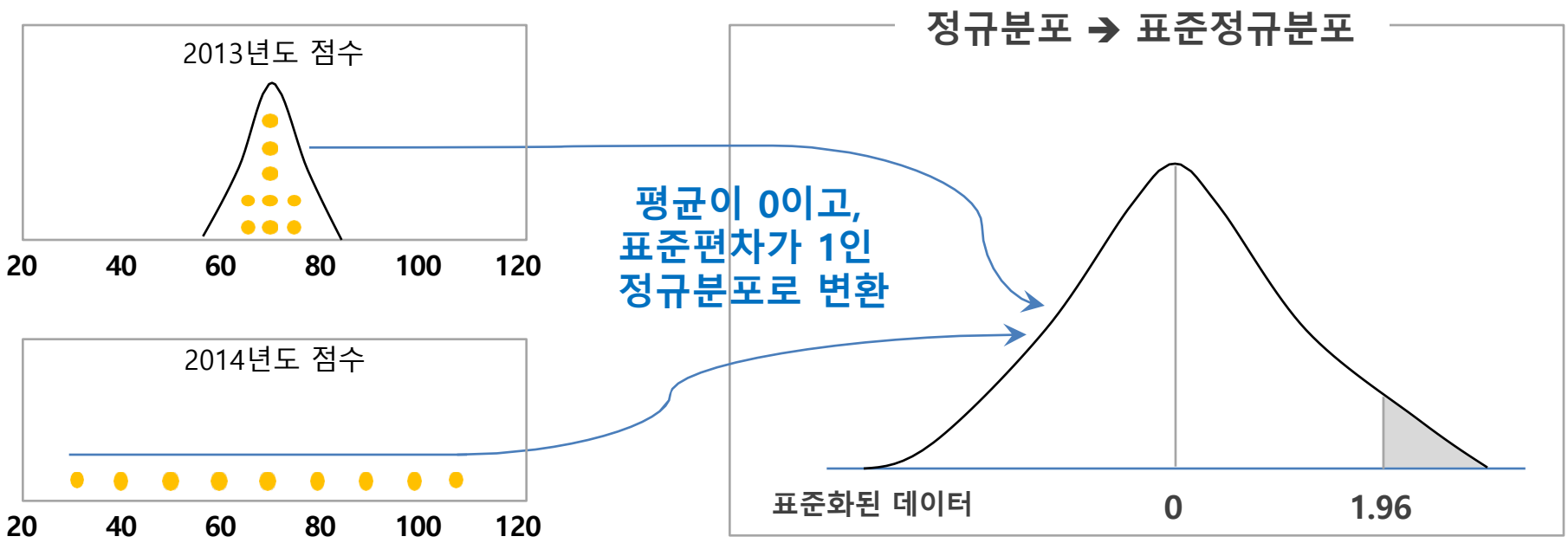
피쳐 스케일링 > 표준화(Standardization)

서로 다른 변수의 값 범위를 일정한 수준으로 맞추는 작업을 피쳐 스케일링이라고 함

표준화(Standardization)

키와 몸무게, 구매상품 수와 구매 개격 등 단위가 다른 두 수치 데이터를 직접 비교하는 것은 의미가 없음. 이럴 때는 데이터의 Feature 각각이 평균이 0이고, 분산이 1인 표준정규분포를 가진 값으로 변환하여 비교하면 효과적임 (z-score 표준화)

$$\text{Z score} = \frac{\text{측정치} - \text{평균}}{\text{표준편차}}$$



피쳐 스케일링 > 정규화(Normalization)

서로 다른 변수의 값 범위를 일정한 수준으로 맞추는 작업을 피쳐 스케일링이라고 함

Min-Max Scaling

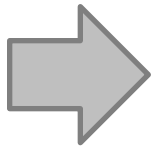
서로 다른 피처의 크기를 통일하기 위해 크기를 0~1 또는 -1~1로 변환해 주는 것

$$N = \frac{x_I - \min(x)}{\max(x) - \min(x)}$$

집값 $y = ax_1 + bx_2 + c$ (모델은 a, b, c 값을 찾는 것이 목표)

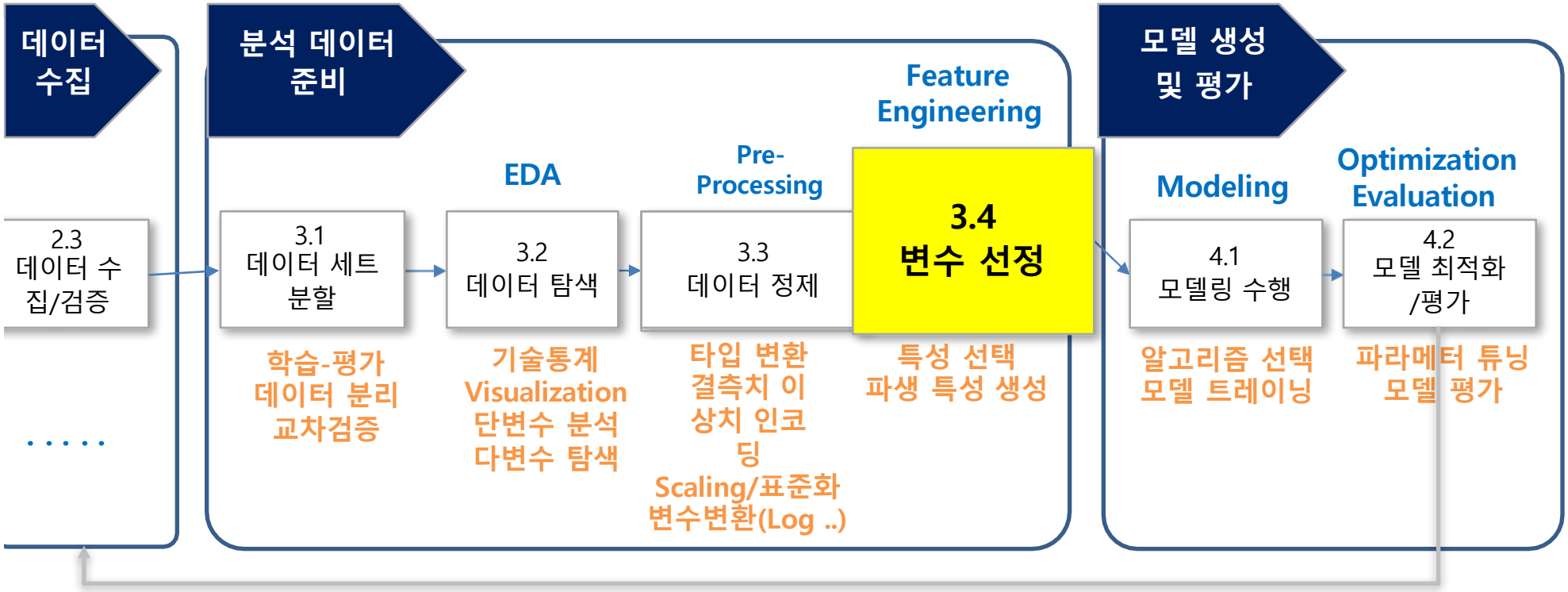
모델이 a, b, c 값을 바꿔가며 y를 계산하는데,
x_2 값이 x_1 에 비해 너무 커서 y 값이 의도치 않게 휘청거림. 따라서, 크기를 맞추어 줌

	방수 x_1	면적 x_2
단위	개	m^2
최소값	0	0
최대값	10	20,000



	방수 x_1	면적 x_2
단위	-	-
최소값	0	0
최대값	1	1

Machine Learning 절차



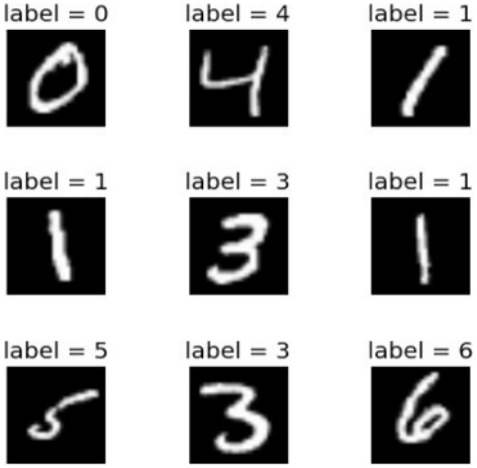
차원의 저주

특성의 수가 증가할수록 동일한 설명력을 유지하기 위해 필요한 레코드의 수는 기하급수적으로 증가함. 이는 훈련을 느리게 할 뿐 아니라, 최적의 솔루션을 찾기 어렵게 함

이 때문에
어떤 현상을 표현하기 위한 논리적인 방법이 여러가지 있다면

Simple is the Best - Occam's Razor(오컴의 면도날)

※ 이론적으로 차원의 저주를 해결하는 해결책 중 하나는 훈련 샘플의 밀도가 충분히 높아질 때가지 훈련 데이터의 크기를 키우는 것임. 하지만, 데이터가 무한정 있지 않기 때문에 차원을 축소하는 것임



MNIST 손글씨 ML 학습용 데이터
28 * 28 = 784 픽셀로 이루어져 있어 차원이 784차원이 됨

특성 선택, 파생 특성 생성

특성이 많으면 학습에 사용하는 데이터가 많아 학습률이 느려지고, 모델이 복잡해져서 과적합을 할 우려가 있음. 중요한 특성만 선택하거나, 특성을 합쳐서 수를 줄이는 방법이 있고, 어떤 방법을 사용하든 성능에 도움되는 특성은 고르고, 최소한으로 유지하는 방향으로 진행됨

♦ 특성 삭제(Feature Deletion)

불필요한 속성부터 삭제하는 것이 좋은 방법임

♦ 특성 선택(Feature Selection)

- ✓ 전체 변수 중에서 유의미한 변수만을 선택
- ✓ 원래 변수의 형태가 그대로 보존됨

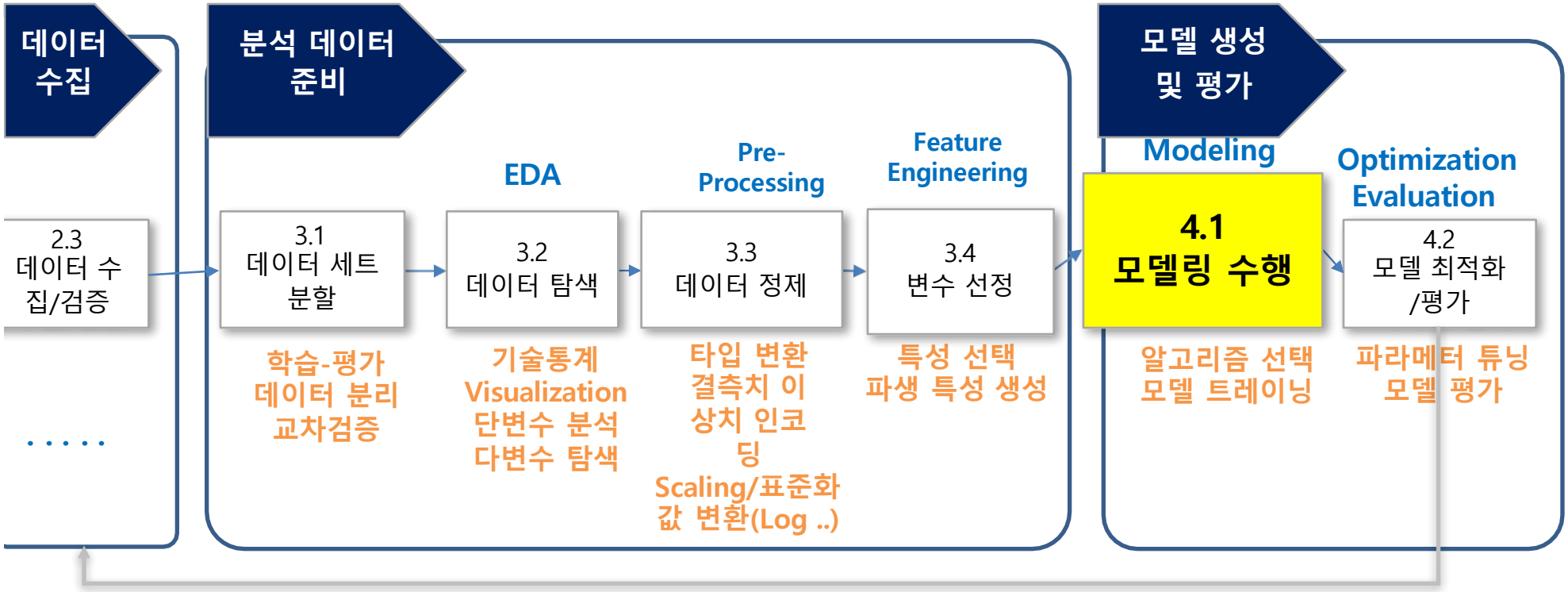
♦ 특성 추출(Feature Extraction, 파생변수 생성)

전체 데이터 집합을 잘 설명할 수 있는 적은 수의 변수를 생성

- ✓ 원래 변수의 형태가 그대로 보존되지 않음
- ✓ Original variables : Age, sex, height, weight
- ✓ Constructed variables : Age+3*I(sex=female) + 0.2*height – 0.3*weight

➔ **차원축소 알고리즘 적용 (Dimension Reduction)**

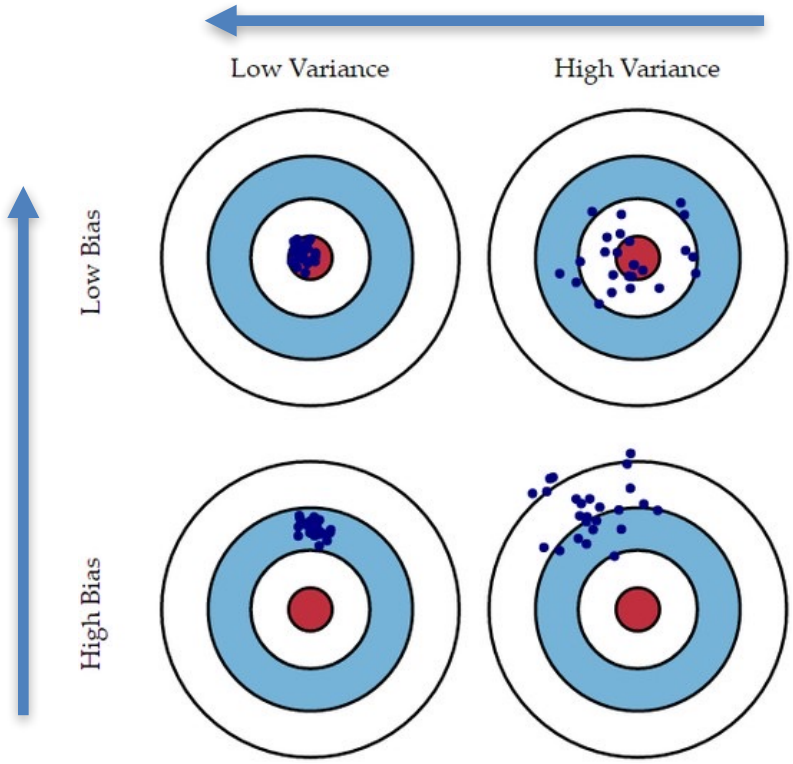
Machine Learning 절차



좋은 모델?

편향(Bias)과 분산(Variance)의 균형

- 예측값들과 정답이 대체로 멀리 떨어져 있으면 결과의 **편향(bias)**이 높음
- 예측값들이 자기들끼리 대체로 멀리 흩어져있으면 결과의 **분산(variance)**이 높음



편향은
알고리즘이 잘못된 가정을 했을 때 발생
➔ 과소적합

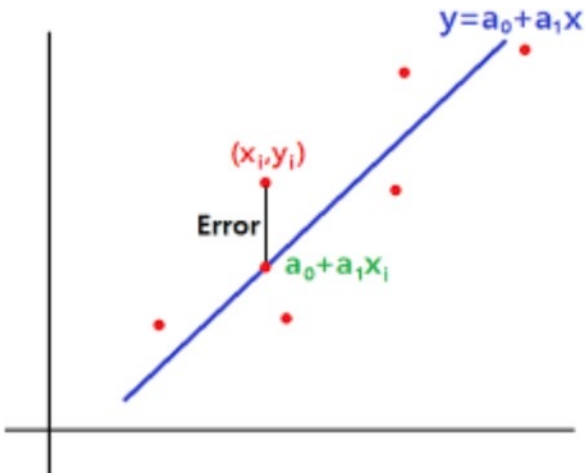
분산은
모델이 노이즈까지 학습하여 새로운 트레이닝셋에 성능이 낮게 나옴 ➔ 과적합

편향과 분산은
한쪽이 증가하면 다른 한쪽이 감소하고,
한쪽이 감소하면 다른 한쪽이 증가하는 경향을 보임

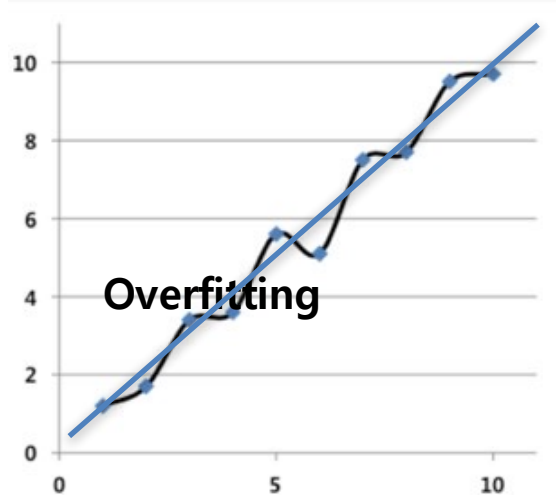
https://www.opentutorials.org/module/3653/22071

좋은 모델?

편향(Bias)과 분산(Variance)의 균형



모델이 표현력이 부족하여 이상적인 모델과 차이가 커짐
➔ **편향이 높음**



모델이 복잡도가 높아서 학습할 때마다 나타나는 모델 편차가 커짐
➔ **분산이 높음**

앙상블(Ensemble) 기법

- ✓ Boosting : 간단한 모델을 여러 개 조립하여 편향을 줄이는 방법 (표현력 보완)
- ✓ Random Forest : 복잡한 모델인 결정트리를 여러 개 조합하여 분산을 줄이는 방법 (유연성 보완)

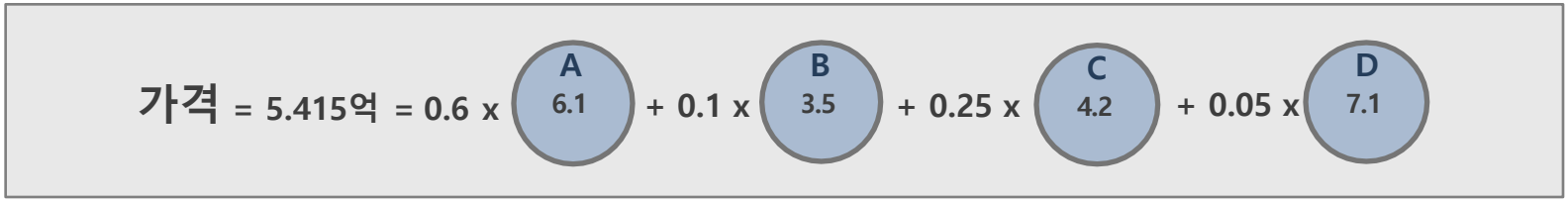
정규화(Regularization) : Ridge, LASSO

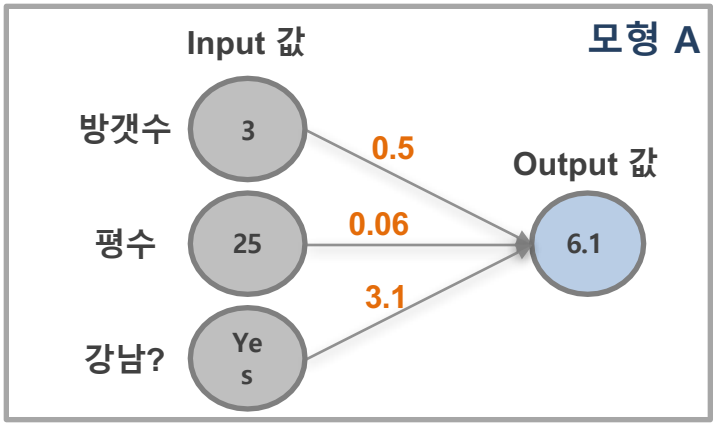
- ✓ 정해진 모델이 필요이상으로 복잡해지지 않도록 모델에 들어 있는 인자에 제한을 주는 방식
  예) $y = w_0 + w_1x_1 + w_2x_2 + w_3x_3$ (단, w_1, w_2, w_3 중에 두 개만 0이 아닌 값을 가진다.)

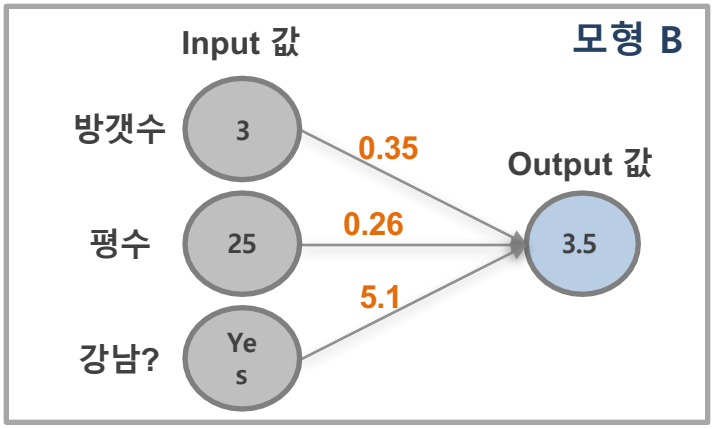
White box vs. Black box

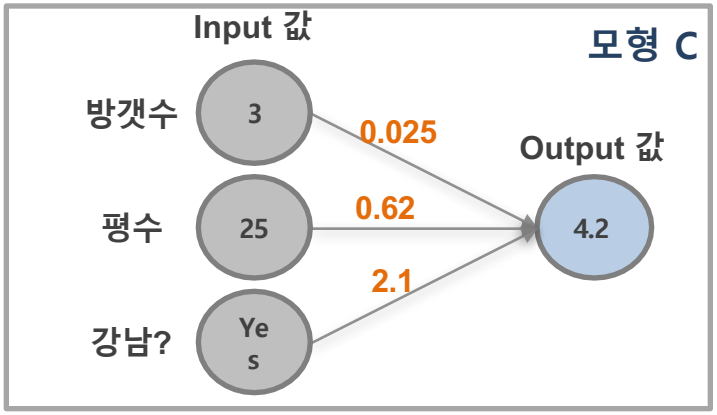
Black Box: 더 정확하지만 이해하기 힘듦. 해석력이 중요할 때는 간단한 모델 적용

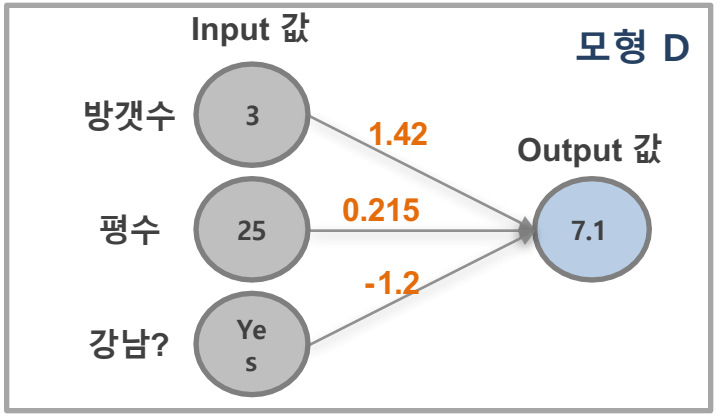
➔ 결국, 선택의 문제





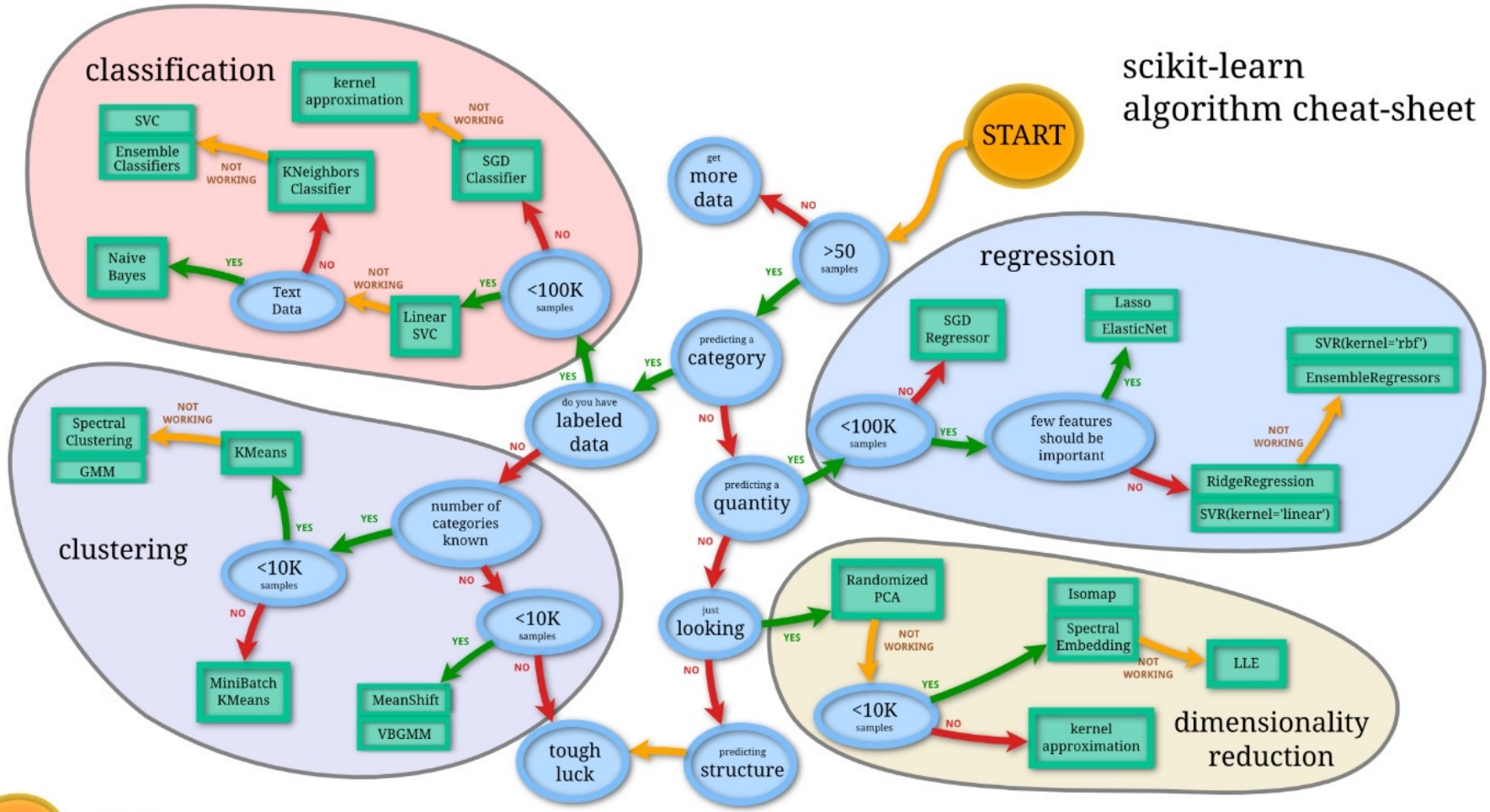






참고. Scikit-learn algorithm cheat-sheet

간단한 모델에서 복잡한 모델로

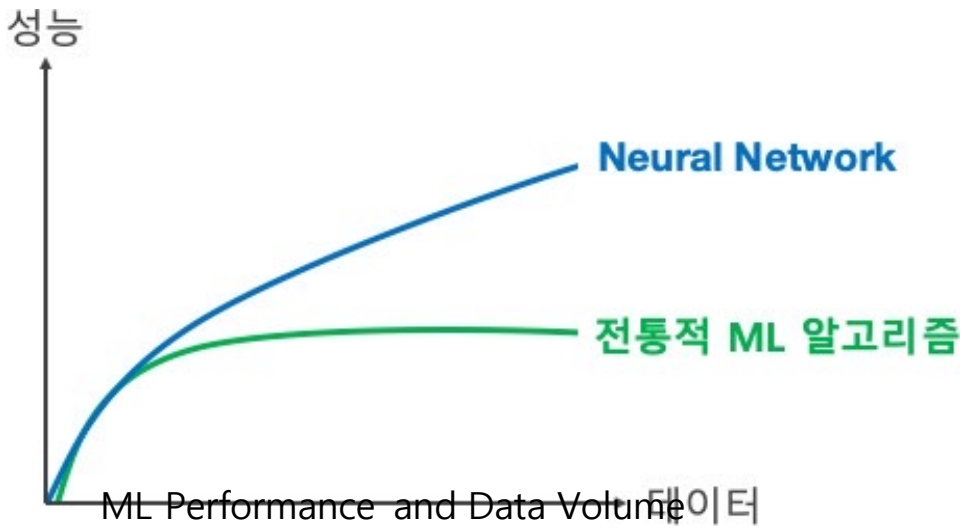


https://scikit-learn.org/stable/tutorial/machine_learning_map/

간단한 모델부터?

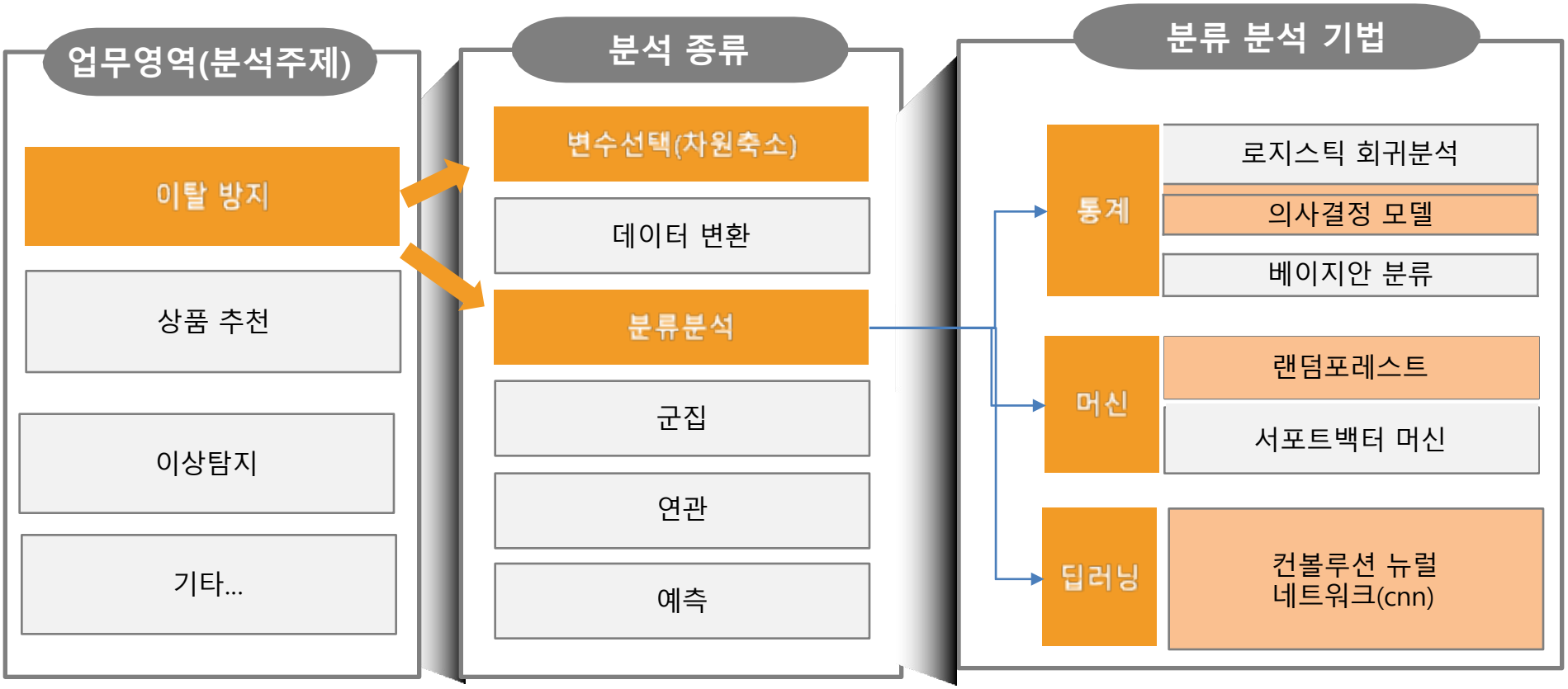
머신러닝은 모델에 따라 성능이 크게 좌우됨
딥러닝이 발전하고, 데이터양이 증가하면서 모델 자체에 대한 고민은 예전보다 적어진 듯
그럼에도 더 좋은 성능을 얻으려면 데이터에 적합한 모델을 이용한 학습이 필요함

어짜피 복잡한 모델이 더 좋은 성능을 낼 확률이 높은데
왜 굳이 시간을 낭비?

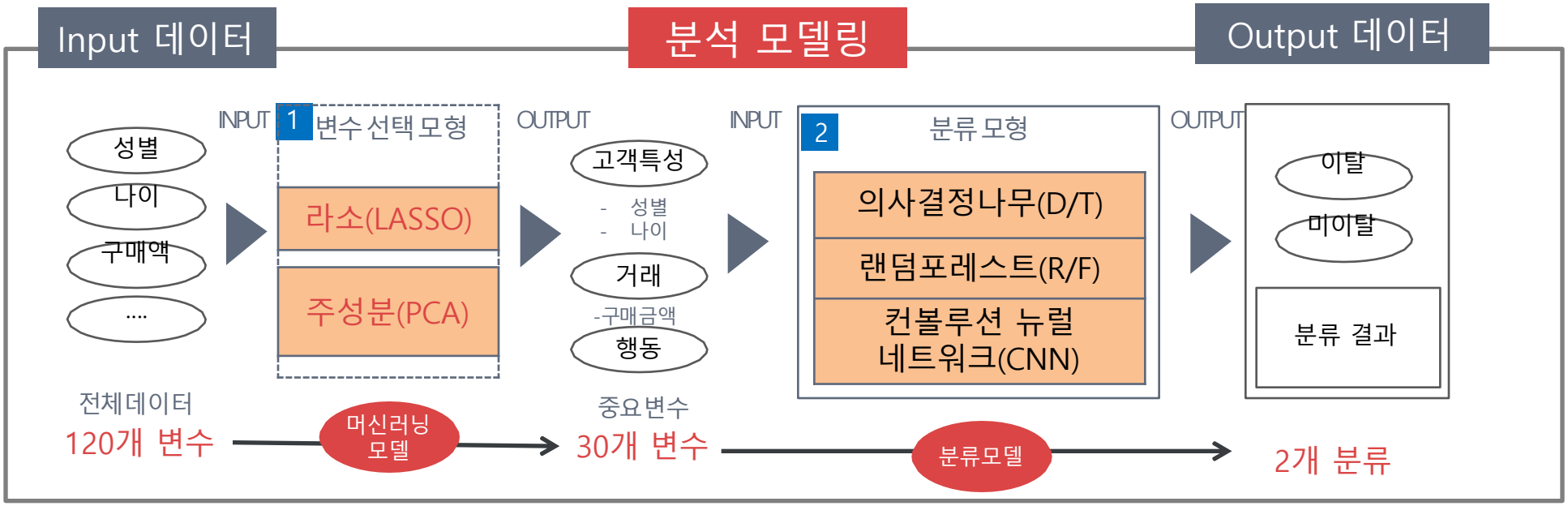


- ✓ 간단한 모델은 구현이 쉽고 학습이 쉬움
- ✓ 간단한 모델의 성능은 다른 모델이 얼마나 잘 동작하는지에 대한 지표가 됨. 즉, 간단한 모델 성능보다 복잡한 모델이 성능이 안나오면 뭔가 문제가 있는것. (개선 방법을 찾아야 함)
- ✓ 간단한 모델은 원인 찾기가 상대적으로 쉬움.

[예시] 분석 주제에 따른 분석 기법

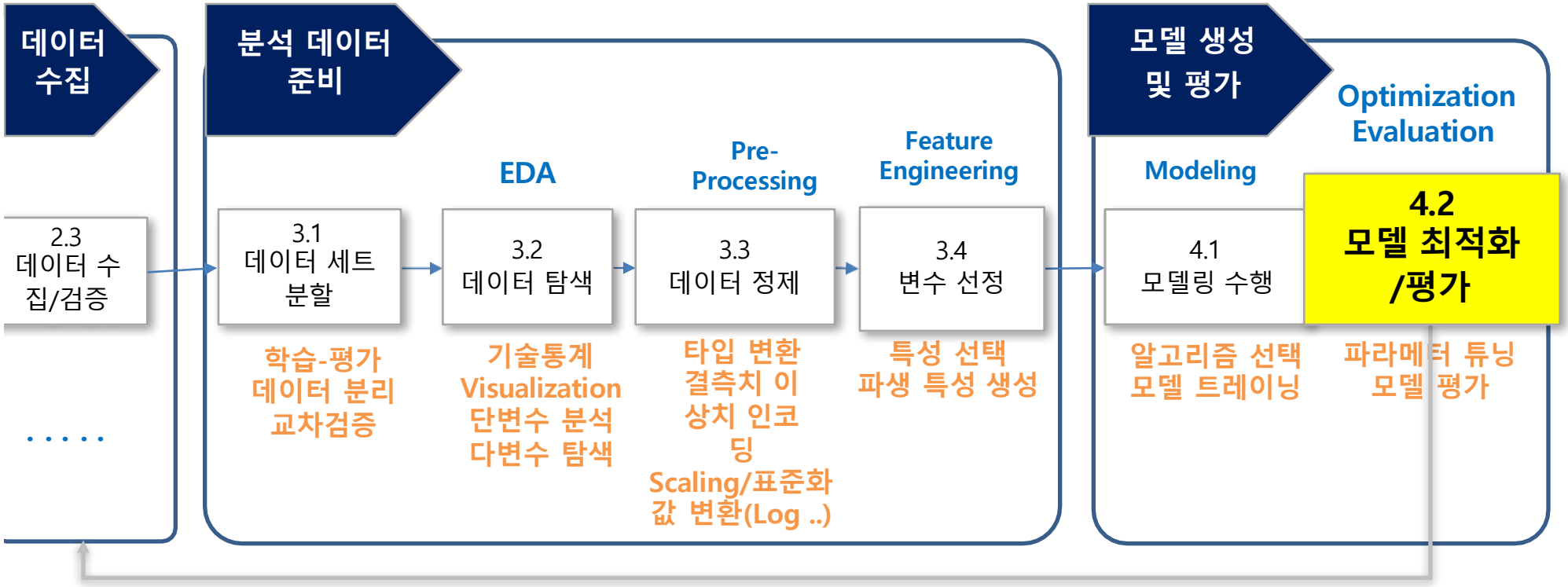


[예시] 분석 주제에 따른 분석 기법



구분	분석 기법	목적	비고
변수 선택 모형	Lasso	영향도(가중치)가 높은 변수 선택 알고리즘	2개 분석 기법 적용 후 "성능(정확도)"이 높은 알고리즘 선택
	Neural Network	상동 (1 Layer 딥러닝 기법)	
분류모형	의사결정나무	분류 기준에 따른 적합도 산출 모형	3개 분석 기법 적용 후 "성능(정확도)이 높은 알고리즘 선택
	랜덤 포레스트	의사결정나무의 앙상블 모델	
	컨볼루션 뉴럴네트워크	분류를 위한 딥러닝 분석 모델	

Machine Learning 절차



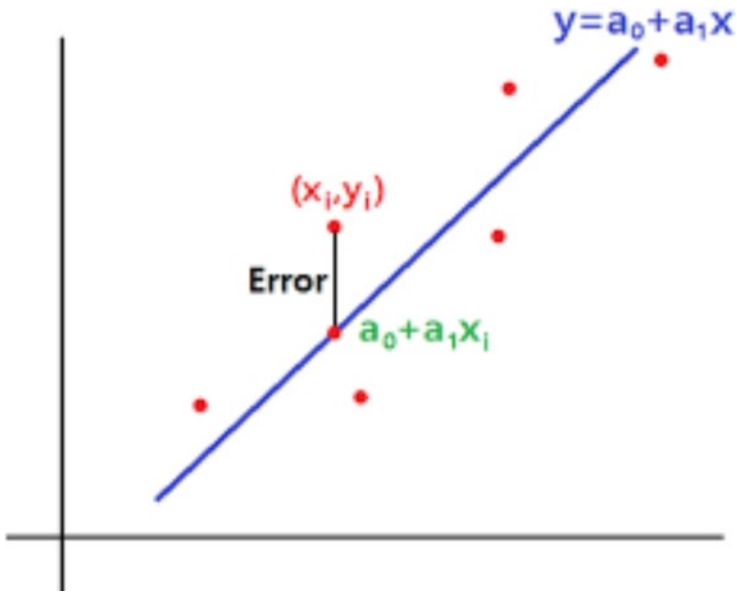
측정지표 > 손실함수(Loss Function)

모델의 수식화된 학습 목표

모델이 실제로 데이터를 바르게 표현했는지, 얼마나 예측이 정확한지(부정확한지)
수학적으로 표현하는 것이 손실함수(Loss Function)임

손실함수의 값이 작을수록 모델이 더 정확하게 학습된 것. 이때 손실함수로 얻은 결과값을 보통 에러라고 부름

손실함수의 값을 줄여가는 과정이 곧 모델을 학습하는 과정임



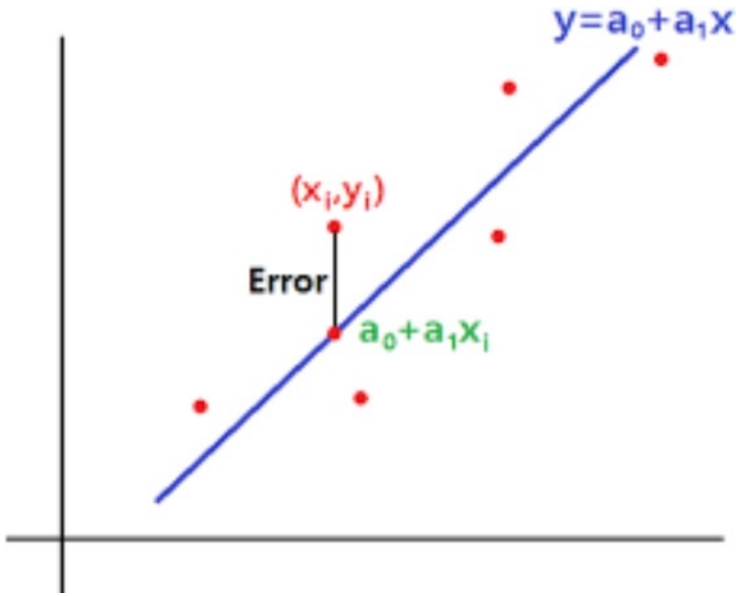
손실(Loss)?

- ✓ 실제 데이터에서 관측된 결과 vs. 모델에 의해 생성된 결과
- ✓ 둘의 차이에 의해 '손실'이 발생
- ✓ 이 '손실'이 작다면? ➔ 모델 성능이 좋음

성능 평가(Evaluation)

모델 성능 평가 지표는 일반적으로 모델이 분류냐 회귀냐에 따라 여러 종류로 나뉨

예시) 회귀의 손실함수



MSE(Mean Squared Error),
RMSE(Root Mean squared Error**)**

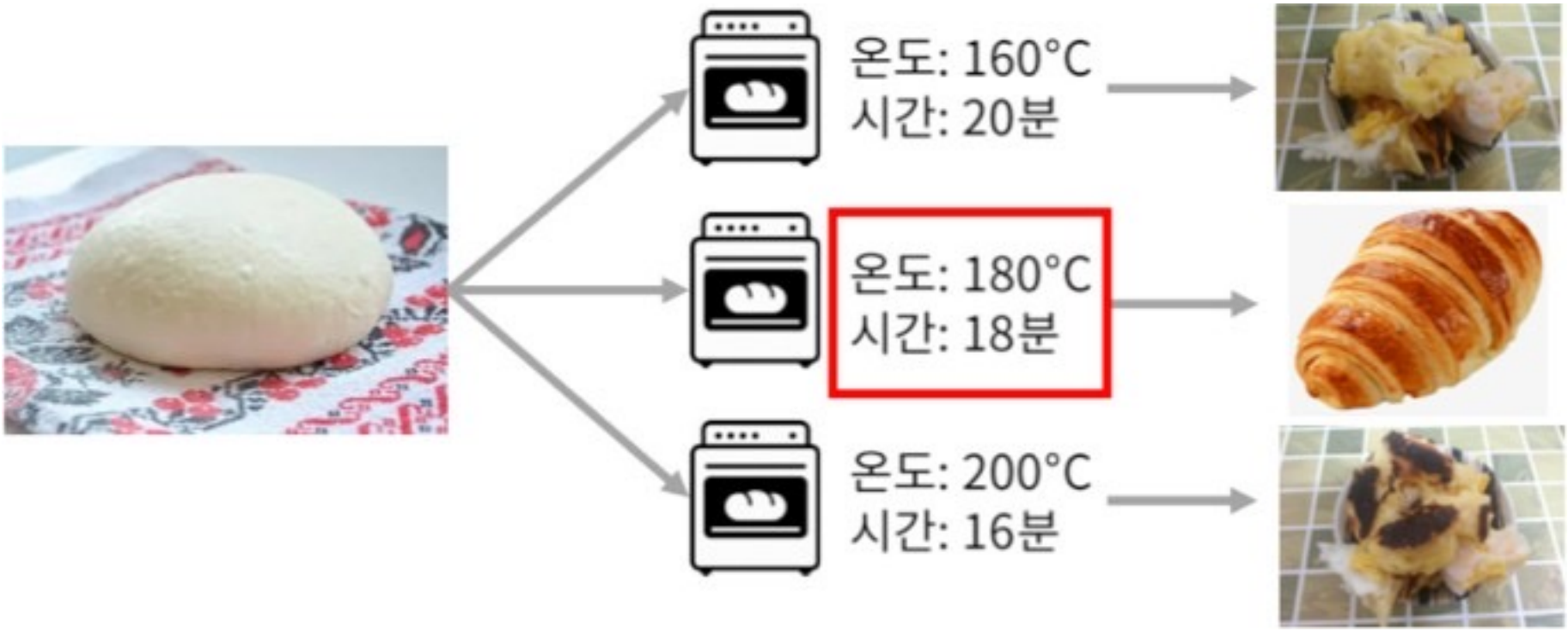
$$MSE = \frac{1}{n} \overset{\text{A}}{=} (y_i - y'_i)^2$$
(BC

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y'_i)^2}$$

회귀문제는 손실함수의 값이 작은 것이 성능이 좋다고 평가함

모델 최적화 – 하이퍼 파라메터

동일한 재료(데이터)와 동일한 오븐(알고리즘)을 사용하되,
오븐에 각각 다른 세팅을 주어 빵을 구웠을 때 최상의 세팅값을 찾는 것



모델 파라메터는 데이터로부터 학습되는 것 (아래 a, b, c)

$$y = ax_1 + bx_2 + cx_3 + \ \ldots$$

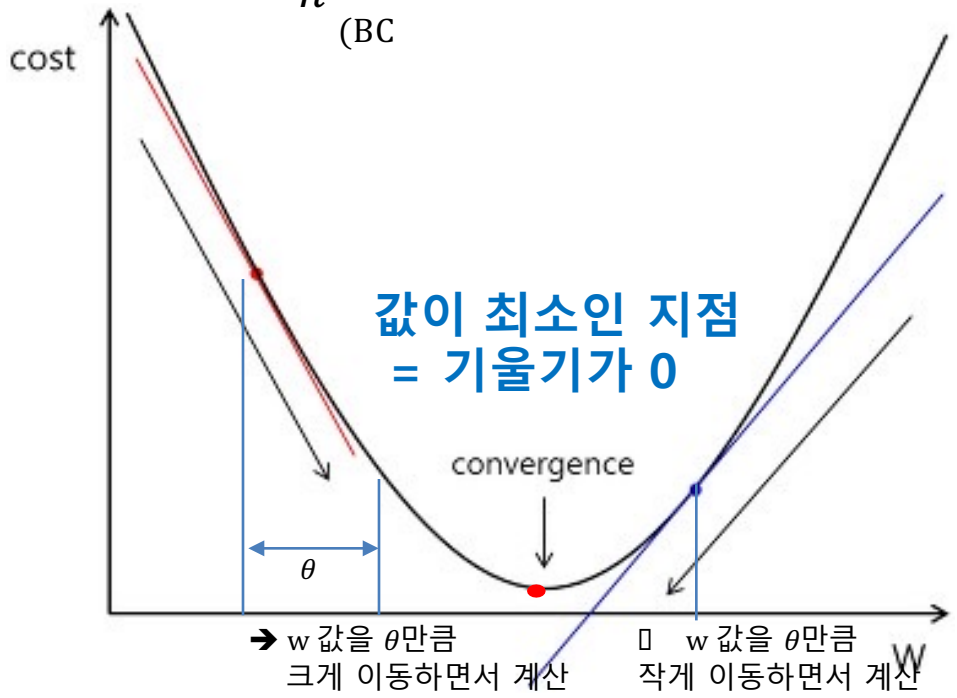
하이퍼 파라메터는 사람들이 선험적 지식으로 설정을 미리 하거나 외부 모델 매커니즘을 통해 자동으로 설정이 되는 변수. 훈련 전에 미리 설정하는 값.

모델 최적화 - 하이퍼 파라메터

예시) Learning Rate

학습 진도율이 너무 작으면 학습의 속도가 느리고, 너무 크면 학습이 안되고 발산할 수 있음. 적절한 값을 찾아야 효율적인 학습이 가능함

선형회귀 손실함수 $MSE = \frac{1}{n} \sum_{(BC}^{A} (y_i - y'_i)^2$



Learning Rate(θ)만큼 이동하면서 기울기 최소값 찾기 : 경사하강법

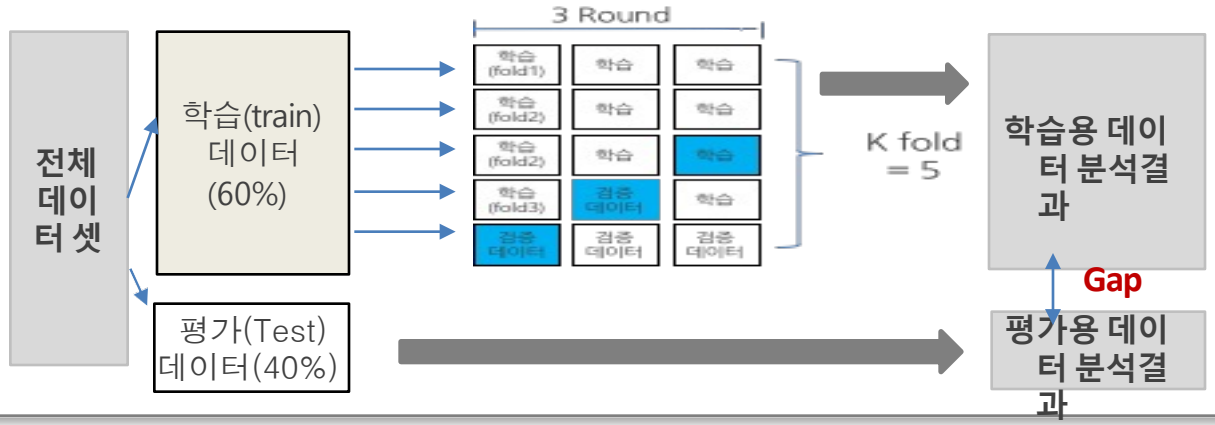
모델 평가

분석 모델의 과적합 여부와 모델의 결과값에 대한 평가(assess)지표를 정의하여 모델을 평가함

모델 평가 종류

모델 평가 내용

과적합평가



- **학습 데이터에서는 잘 맞는데, 실제 상황(검증데이터)에서도 잘 맞는가?**
- 방법 : 과적합(over-fiting)파악
- 검증 : "학습 데이터"에서는 잘 맞는데, "평가 데이터"에서 정확도의 Gap이 크면 과적합 되었음

모델 평가

예측 모델 평가 지표	분류 모델 평가 지표
■ 제곱근 오차(MSE) ■ 평균 제곱근 오차(RMSE) ■ 여러 지표 중 1개 선택하여 평가	■ 정확도(accuracy) ■ 정밀도(Precision) ■ 재현율(recall) ■ 정확도, 정밀도 증 여러 지표의 trade off를 거려한 평가

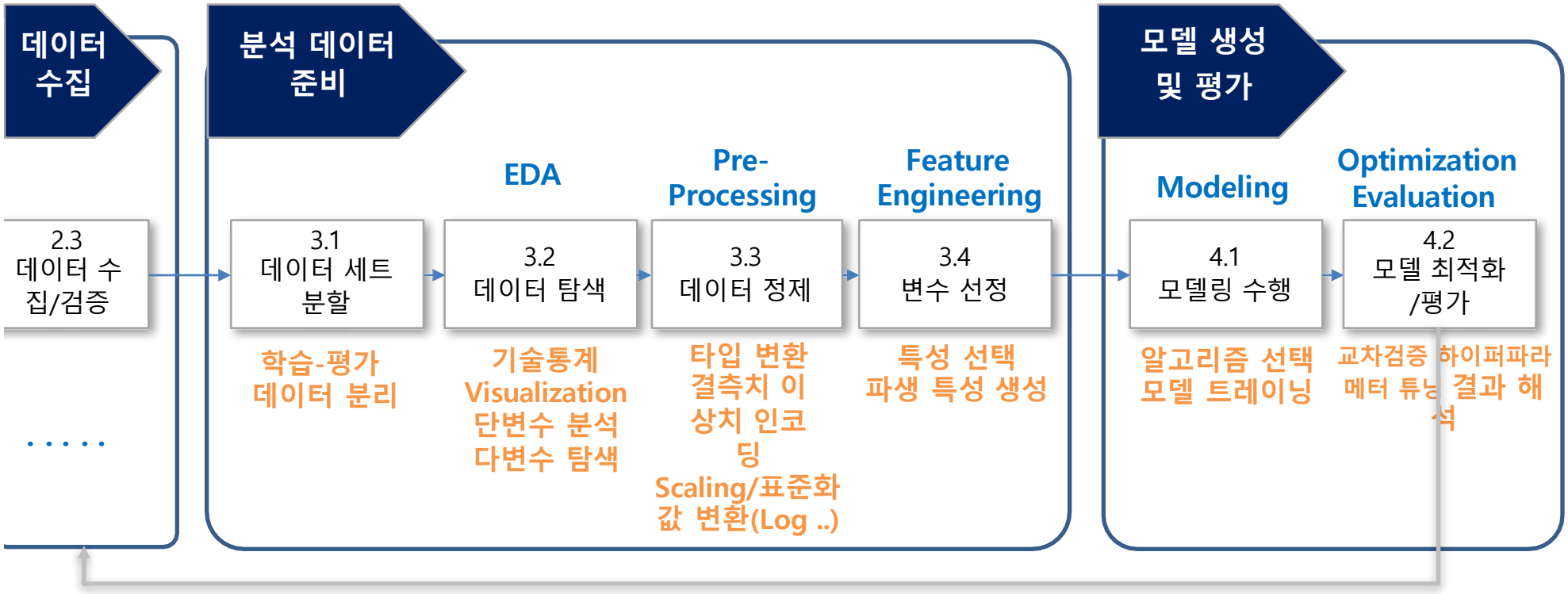
분석모델별 다양한 평가지표 존재함

- **모델이 "값"을 잘 분류또는 예측 하는가?**
- 방법 : 분석 유형/모델별 평가지표 적용
- 평가지표 :
  - 분류 : **accuracy**
  - 예측 : RMSE
  - . . . . .

머신러닝 절차 (세부 Task 중심)

Machine Learning은 아래 방식으로 진행됨

- ✓ 수집한 데이터의
- ✓ 탐색적 데이터 분석(EDA-Exploratory Data Analysis)을 통한 이해를 바탕으로,
- ✓ 모델링에 사용할 데이터를 가공하는 전처리(Pre-processing)
- ✓ 모델 트레이닝을 위한 위한 특성을 선택하는 특성 공학(Feature Engineering)
- ✓ 학습 데이터를 기반으로 머신러닝 알고리즘을 적용해 모델을 학습(Modeling) /최적화(Optimization) 하고, 테스트 데이터로 모델에 대한 평가(Evaluation)를 수행



Thank you