



PART II-03

머신러닝 기본 실습

시간 계획

- 오늘도 파이팅 입니다.^^

시간	학습내용
09:00~10:00	Scikit learn으로 머신러닝 모델 학습 기본 익히기
10:20~11:20	머신러닝 모델 학습 기본 익히기
11:40~12:40	EDA 탐색
12:40~14:00	즐거운 점심 시간
14:00~15:00	학습데이터 성능데이터 분리, 학습모델 만들기
15:20~16:20	데이터 전처리, 결정트리 학습모델, 예측하기
16:40~17:50	머신러닝 모델 학습 예측, 평가

강의 내용

구분	주제	내용
Part1	지도학습 분류(Classification)	머신러닝 기본 이해 DecisionTree, SGD 등 학습모델 성능 평가
Part2	지도학습 회귀(Regression)	LinearRegression, Ridge, Lasso, ElasticNet Kfold 교차 검증 학습모델 성능 평가
Part3	지도학습 앙상블	배깅, 부스팅 기법 이해 및 실습 랜덤포레스트, XGBoost

학습 목표

데이터 분석을 위한 머신러닝의 기본 개념과 특징을 이해한다

Scikit-learn을 활용해, 머신러닝 학습모델을 구현한다.

학습 데이터와 테스트 데이터의 의미를 알고 생성한다.

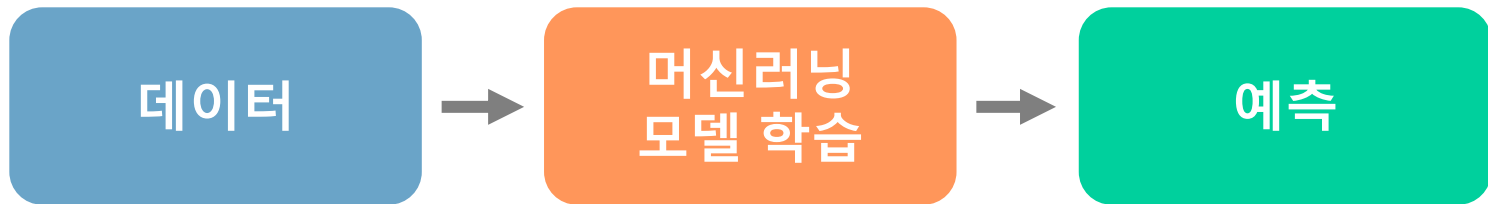
머신러닝 학습모델을 만드는 과정을 안다.

분류(Classification) 학습모델의 객체를 생성하고 데이터를 학습 및 예측을 수행한다.

머신러닝 개요

머신러닝이란?

- 명시적인 프로그래밍 없이 기계가 데이터를 이용해서 학습을 하고 예측을 수행하는 알고리즘을 구현하는 기법
 - Arthur Samuel
- 과거 경험에서 학습을 통해 얻은 지식을 미래의 결정에 이용하는 컴퓨터 과학의 한 분야
- 관측된 패턴을 일반화하거나 주어진 샘플을 통해 새로운 규칙을 생성하는 목표를 가짐



머신러닝이 필요한 이유

머신러닝 방법론을 이용할 경우,

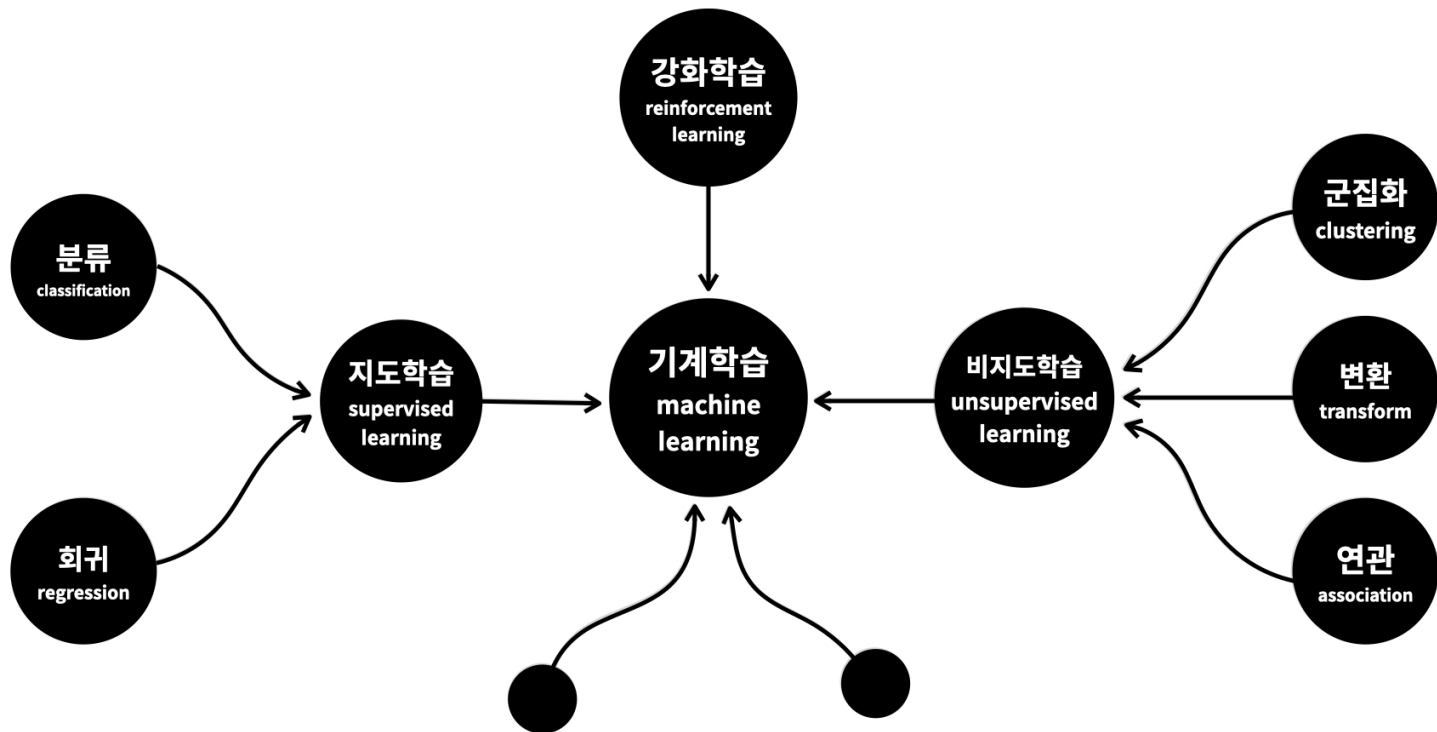
- 인간이 정확히 하나하나 로직을 지정해주기 **어려운 복잡한 문제를 데이터에 기반한 학습을 통해서 해결할 수 있음.**
- 예)
 - 어떤 사용자에게 무슨 광고를 보여주는 것이 최적의 광고 배분 전략일까?
 - 어떤 카드 사용내역이 사기에 의한 사용 내역이고, 어떤 사용 내역이 정상 사용 내역일까?
 - 사용자의 성향에 맞는 영화를 추천하면 도움이 될까?

예측 모델(Prediction Model)의 필요성

데이터 분석을 통한 정교한 예측 모델을 갖고 있을 경우,

- 중요한 비즈니스적 의사결정을 안정적이고 계획적으로 수행 가능
- 예)
 - 다음달 휴대폰 판매량은 얼마나 될까?
=> 생산계획, 재고관리 전략 수립 가능
 - 광고비를 200만원 더 집행하면 얼마나 많은 유저를 추가적으로 획득할 수 있을까?
=> 목표로 하는 유저 획득 수에 따른 광고비 집행 전략 수립 가능

머신러닝 학습 방법 분류



머신러닝 - 지도학습(Supervised Learning)

- 학습 데이터가 입력과 출력 쌍으로 제공
 - 입력 : 특징 행렬, Feature, 독립변수
 - 출력 : 대상 벡터, 정답, 레이블, Target
- 머신러닝 학습 모델의 목표는 **입력 특징 행렬과 출력 대상 벡터를 매핑**시키는 규칙을 찾는 것임
- **입력 데이터와 출력 정답을 알려줘서 학습하므로 지도라고 함**
- 딥러닝은 95%이상이 지도학습이 사용되고 있음(CNN, RNN)

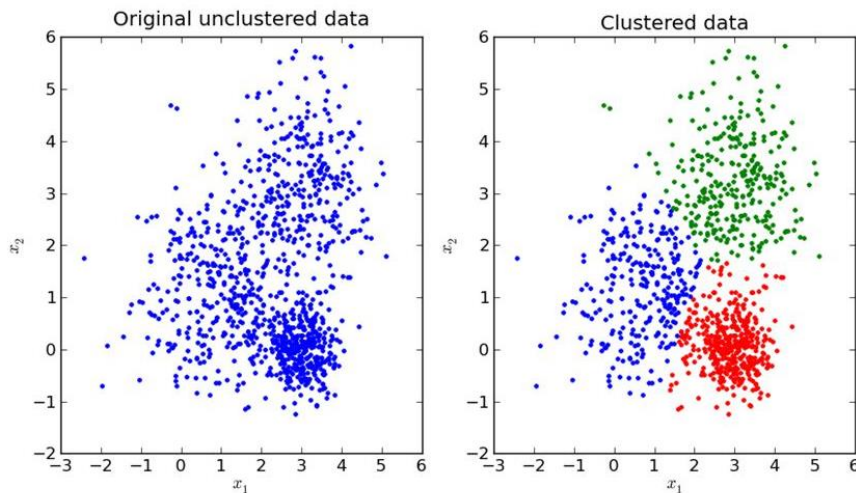
머신러닝 - 지도학습 - Classification vs. Regression

머신러닝은 크게 분류문제와 회귀문제로 나뉨

- 분류(Classification)문제 : 예측하는 결과값이 이산값인 경우
예) 강아지인지 고양이 인지?, 이미지의 숫자가 1인지 2인지?
- 회귀(Regression)문제 : 예측하는 결과값이 연속값인 경우
예) 3개월 뒤 이 아파트 가격은 4억일 것인가? 5억일 것인가?
증가차 가격은 얼마일까?, 코스피 종합주가가 지수가 3500까지 갈까?
- 알고리즘 : KNN, 선형 회귀, 로지스틱 회귀, SVM, 결정트리, 랜덤 포레스트, 신경망

머신러닝 – Unsupervised Learning

- 학습 데이터 x 만을 이용해서 학습하는 방법
- 데이터의 숨겨진 특징(hidden feature)을 찾아내는 것이 목적임.
- 데이터가 무작위로 분포되어 있을 때,
- 예) 블로그 글의 주제 구분, 고객들을 취향이 비슷한 그룹으로 묶기,
비정상적인 웹사이트 접근 탐지

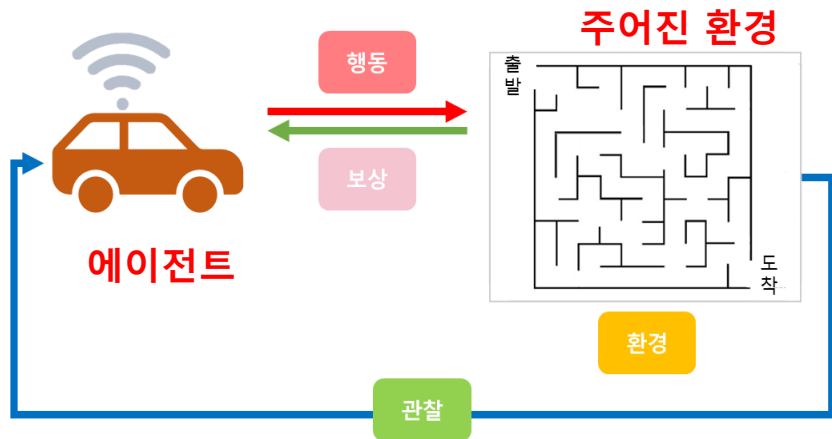


머신러닝 – Unsupervised Learning

- 비지도학습 단독으로 사용하기 보다는 비지도학습으로 파악한 데이터의 숨겨진 특징을 원본 데이터 대신 지도학습의 입력 데이터로 활용해서 지도학습의 성능을 더욱 끌어올리는 용도로 많이 활용됨
- 대표적인 학습방법
 - 주성분분석(PCA) : 머신러닝 비지도학습을 위해 사용
 - 오토인코더(Autoencoder) : 딥러닝 비지도학습을 위함 많이 사용

머신러닝 – Reinforcement Learning

- 피드백을 바탕으로 성능을 평가하고, 그에 따라 반응하는 학습법
- 시스템이 어떤 목표를 달성하기 위해 동적인 조건에 대응하도록 함
- 학습하는 에이전트(Agent)가 주어진 환경(State)에서 어떤 행동(Action)을 취하고 이에 대한 보상(Reward)을 얻으면서 학습 진행
- 시간이 경과하면서 가장 큰 보상을 얻기 위해 최상의 전략(정책)을 스스로 학습함



Scikit learn 소개

Scikit-learn 사용



- 설치
 - `python3.8 -m pip install -U scikit-learn`
 - colab은 설치 되어 있음
- import 방법
 - `from sklearn.linear_model import LinearRegression`
 - `from sklearn.model_selection import train_test_split`

머신러닝을 위한 파이썬 패키지



sklearn

numpy

pandas

seaborn

matplotlib

Scikit-learn 소개

<https://scikit-learn.org/>

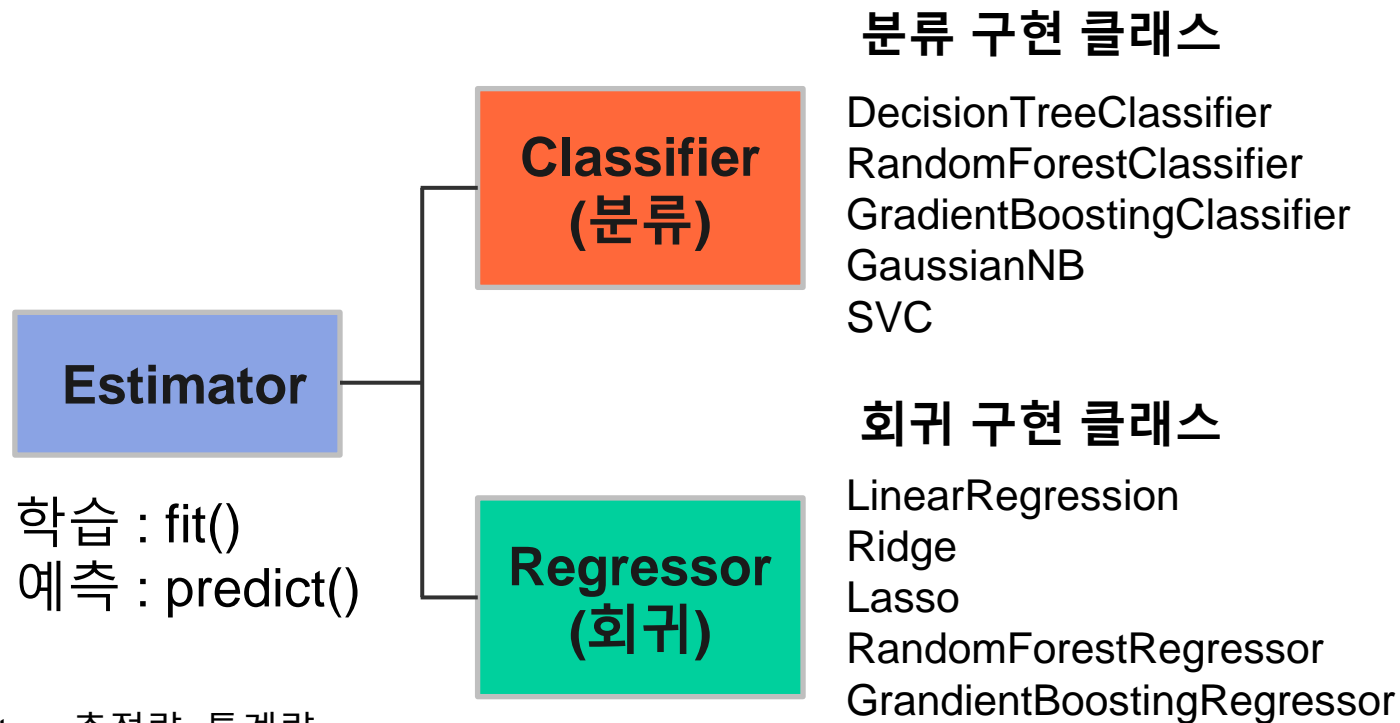
The screenshot shows the Scikit-learn website homepage. At the top is a navigation bar with links for 'Install', 'User Guide', 'API', 'Examples', and 'More'. Below this is the 'scikit-learn' logo and the tagline 'Machine Learning in Python'. A secondary navigation bar contains links for 'Getting Started', 'Release Highlights for 0.24', and 'GitHub'. The main content area is divided into three columns, each representing a machine learning task:

- Classification:** Identifying which category an object belongs to. Applications include spam detection and image recognition. Algorithms include SVM, nearest neighbors, random forest, and more. An example image shows a 4x4 grid of handwritten digits.
- Regression:** Predicting a continuous-valued attribute associated with an object. Applications include drug response and stock prices. Algorithms include SVR, nearest neighbors, random forest, and more. An example plot shows a 'Boosted Decision Tree Regression' model fitting a noisy sine wave.
- Clustering:** Automatic grouping of similar objects into sets. Applications include customer segmentation and grouping experiment outcomes. Algorithms include k-Means, spectral clustering, mean-shift, and more. An example image shows 'K-means clustering on the digits dataset' with clusters represented by different colors.

- 머신러닝 패키지 중 가장 쉽고 파이썬스러운 API 제공, 오픈소스
- 머신러닝을 위한 매우 다양한 알고리즘과 편리한 API 제공
- 오랜 기간 실전 환경에서 검증된 성숙한 라이브러리
- 정형데이터 예측 학습모델 만들기에 여전히 많이 쓰임
- 산업현장, 학계에 널리 사용됨
- Numpy와 Scipy 기반으로 구축된 라이브러리

Scikit-learn 프레임워크 소개

- Scikit-learn의 지도학습 알고리즘



* Estimator : 추정량, 통계량

머신러닝 지도학습 어떤 Estimator(학습기) 써야 할까요?

- 코로나19 백신 2차까지 접종, 전국민 70%까지 접종이 완료 되었다.
With 코로나 상황, 이제 마음껏 여행을 다닐 수 있다. 오늘 금요일, 내일 가족 또는 애인과 여행을 가기로 했다.
- 내일 비가 올까 안 올까? 기온은 몇 도가 될까?



Scikit-learn

학습모델 만들고 예측하기

머신러닝 과정

데이터를 기반으로 **패턴을 학습**하여 **결과를 예측**하는 것

Data

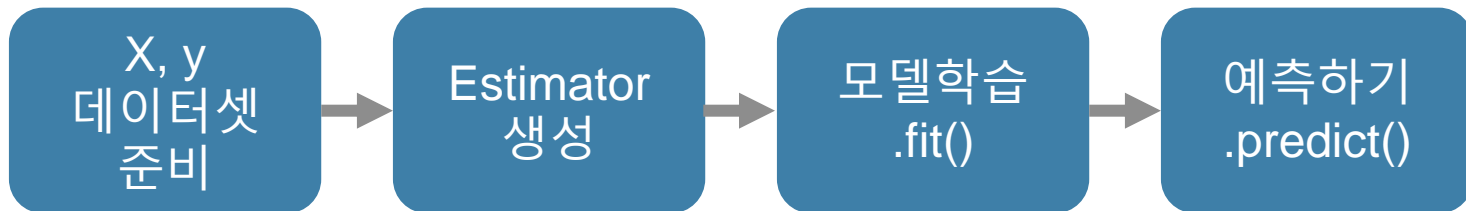
Model

Prediction



[실습1] scikit-learn으로 머신러닝 모델 만들기

- 입력 데이터(X) : 0~29까지 숫자 데이터
- $\text{target}(y) = 2 * X + 1$ 의해 결정됨
- 해결 문제 :
 - 머신러닝 학습모델에, 새로운 데이터 x를 입력 했을 때, 결과 y를 예측할 수 있을까?
- workflow



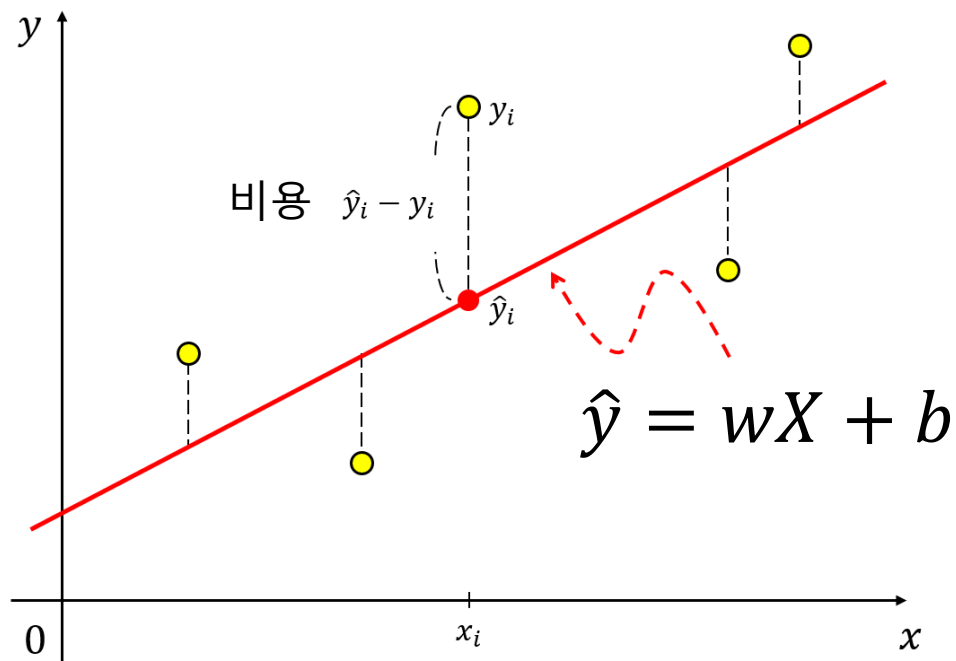
실습파일 : 3-2-1.sklearn의_개요-학습(fit)_예측(predict).ipynb

머신러닝 기본 개념

가설함수, 비용, 손실;error 함수 Hypothesis, Cost, Loss Function

머신러닝 기본 개념

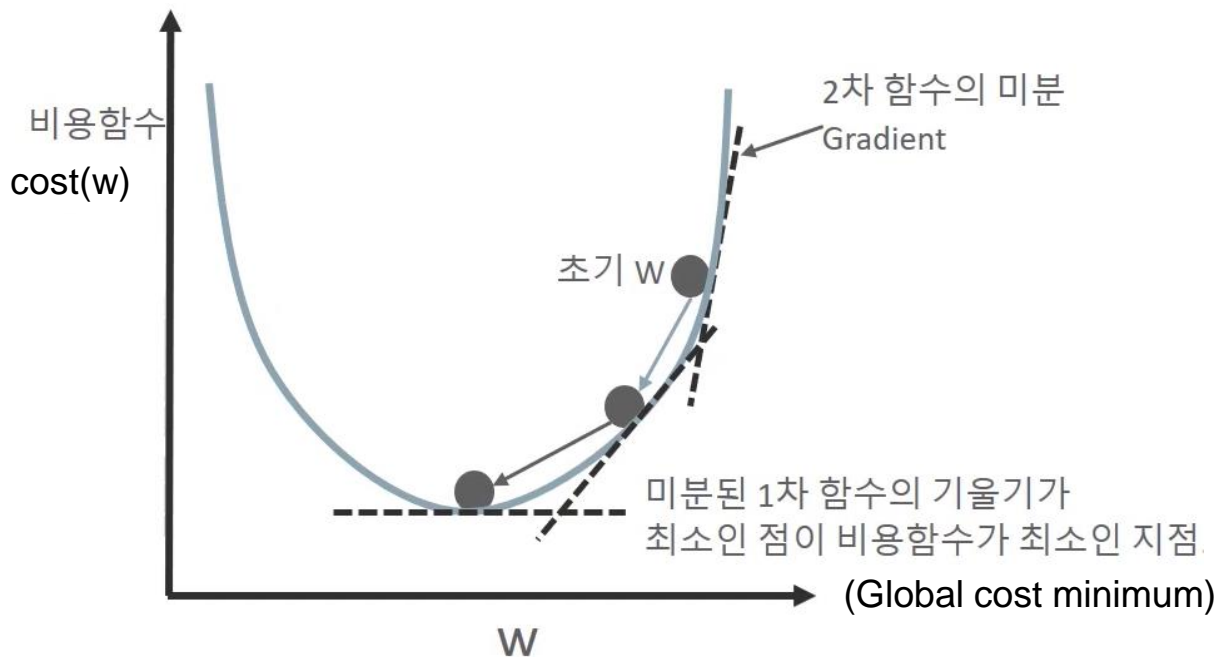
- 가설함수 $H(x) = \hat{y}$



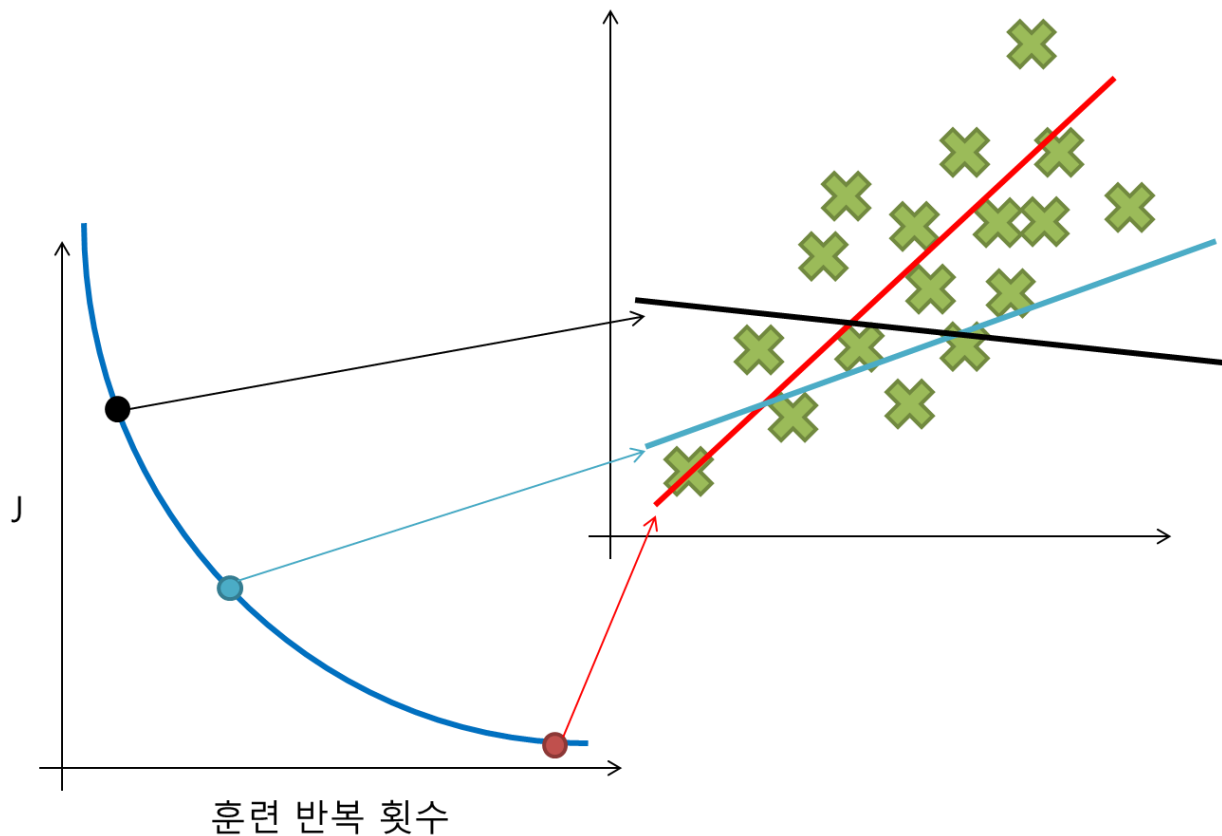
최소 오차 지점인 최적의 W 찾기

- 머신러닝의 성능을 높이는 방법은? 오차를 최소화 하는 것

$$cost(w) = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$



훈련 횟수에 따른 손실함수와의 관계



머신러닝 기본 용어와 학습과정 살펴보기

머신러닝 모델 학습 및 예측 과정 정리

1. 머신러닝 모듈 import

```
from sklearn.linear_model import LinearRegression
```

2. 학습모델 객체 생성

```
model_lr = LinearRegression()
```

3. 모델 데이터 학습

```
model_lr.fit(test_x, test_y)
```

4. 새로운 데이터로 예측

```
pred = model_lr.predict(X2)
```

학습 데이터, 예측데이터
(features, label, train, test)

머신러닝 용어정리 - feature, label

```
from sklearn.linear_model import LinearRegression
model_lr = LinearRegression()
model_lr.fit(X_train, y_train)
prediction = model_lr.predict(x2)
```

- X : - **features**(특징행렬, 설명변수, 독립변수)
- 변수 : X_train, X_test
 - 학습을 위한 데이터 세트(X_train), 예측을 위한 테스트 값(X_test)
- y : - **label(target**, 정답, 종속변수, 결정값, Class)
- 변수 : y_train, y_test
 - 학습 데이터의 정답(y_train), 예측 후 평가하기 위한 결과(y_test)

머신러닝 - train, test 데이터

학습을 위한 데이터
= Training data set
X_train, y_train

예측을 위한 데이터
= Test data set
X_test

- 모델이 학습하기 위해 필요함
- **feature/label(target)**이 모두 존재

- 모델이 예측하기 위한 데이터
- **feature**만 존재

[실습2]titanic 사고 데이터로 생존자 예측

- 입력 데이터 : sex, age, sibsp ...
 - 12Dimension (survived, embark_town, alive 컬럼 제외)
- Target data : survived 컬럼
 - 0 (사망), 1(생존)
 - Binary Classification
- 학습 모델 알고리즘(Estimator)
 - 확률적 경사하강법(SGDClassifier)
- 추가적인 적용기법
 - EDA(Exploratory Data Analysis)
 - 성능평가

[실습2]데이터 EDA 및 학습모델 성능 비교

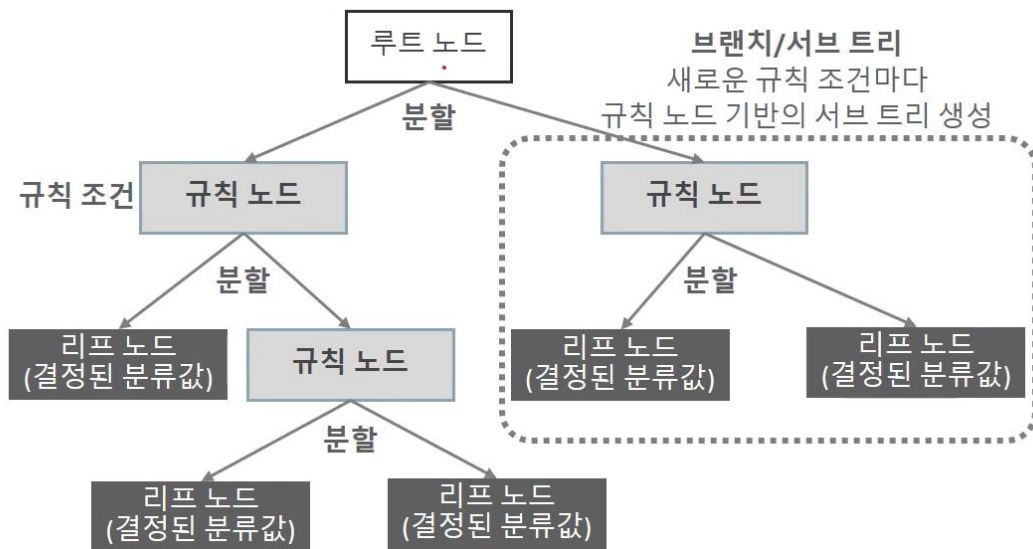
- 3-2-2.지도학습_classification_EDA_성능평가_titanic_clean_data사용.ipynb



지도학습 – 결정 트리 타이타닉 생존자 예측

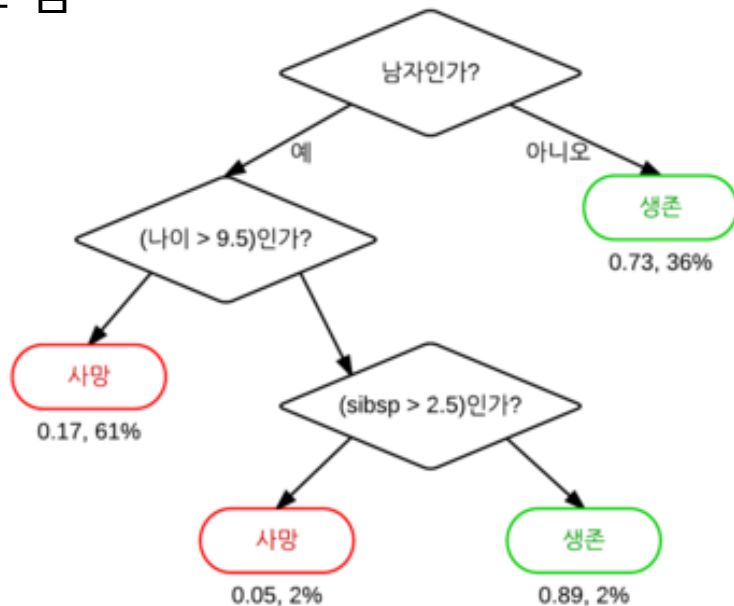
결정 트리(Decision Tree)

- 데이터에 있는 규칙을 학습을 통해 자동으로 찾아내 트리 기반으로 분류 규칙을 만듦(if else 기반 규칙)
- 데이터의 어떤 기준을 바탕으로 규칙을 만들어야 가장 효율적인 분류가 될 것 인가가 알고리즘의 성능을 크게 좌우함.



결정 트리(Decision Tree)


- 데이터 마이닝에서 일반적으로 사용되는 방법론
- 몇몇 입력변수를 바탕으로 목표 변수의 값을 예측하는 모델을 생성하는 것을 목표로 함



입력데이터값

<https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=moonsoo5522&logNo=220888886396>

결정 트리의 장점과 단점



장점

- 알고리즘의 동작과정이 직관적임
- 학습시간이 빠름
- 개별 Feature들을 개별적으로 판단하므로 Feature Normalization 등이 필요하지 않음.

단점

- Tree의 depth가 깊어지면, 오버피팅(Overfitting)에 빠지기 쉬움

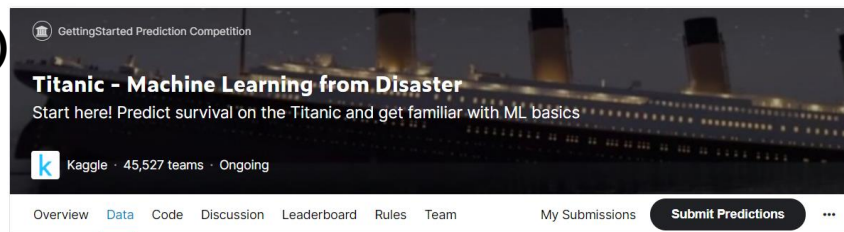
Scikit-learn의 DecisionTree Estimator



- **sklearn.tree.DecisionTreeClassifier**(분류문제에 사용)
- **sklearn.tree.DecisionTreeRegressor**(회귀문제에 사용)

[실습2]titanic 사고 데이터 – 생존자 예측

- 데이터 URL : <http://www.kaggle.com/c/titanic/data>
- 1912년 타이타닉 사고 당시의 승객에 대한 데이터
- Feature 데이터 columns : survived, pclass, sex, age, sibsp, ...
 - 15Dimension
- **Target Value : 1(생존자), 0(사망자)**
 - BinaryClassification
- 데이터 개수 : 891명의 승객



Data Description

Overview

The data has been split into two groups:

- training set (train.csv)
- test set (test.csv)

The training set should be used to build your machine learning models. For the training set, we provide the outcome (also known as the "ground truth") for each passenger. Your model will be based on "features" like passengers' gender and class. You can also use [feature engineering](#) to create new features.

*This contest and associated leaderboard are for educational purposes only. The contest is not a competition and does not constitute a contest or competition.

[실습2]titanic 사고 데이터 Features 살펴보기

Feature	Definition	Value
survived	생존여부	생존 여부, 1: 생존, 0: 사망
pclass	티켓 등급(1등석, 2등석, 3등석)	1=1 st , 2=2 nd , 3 = 3 rd
sex	성별	male : 남성, female : 여성
age	나이	숫자
sibsp	함께 탑승한 형제, 자매, 배우자 수의 합	숫자
parch	함께 탑승한 부모, 자식 수의 합	숫자
fare	운임 요증(티켓 가격)	숫자
embarked	탑승 항구(첫글자)	C=cherbourg, Q=Queenstown, S=Southampton
class	티켓 등급(단어로)	first, Second, Third
who	남성/여성/아이 구분	man, woman, child
adult_male	성인 남성인지의 여부	true, False
deck	선박에서 배정받은 좌석의 구역	A, B, C, D, E, F, G, 빈 값
embark_town	출항지(폴네임)	도시 이름
alive	생존여부	yes, no
alone	혼자인지 여부	True, False

Categorical column vs Numerical Column

Categorical column : 범주형 데이터 값

- [1,2,3], ["내부", "외부"]와 같이 몇 가지 분류로 한정되는 데이터임
- ex) categorical_cols = ["sex", "embarked", "class", "who", "adult_male", "deck", "embark_town", "alive", "alone"]

Numerical column : 수치형 데이터 값

- 1, 2, 3, ... 1.2, 4.51, 3.145 와 같이 숫자 축으로 무한히 위치할 수 있는 데이터
- ex) numerical_cols = ["age", "sibsp", "parch", "fare"]

데이터 인코딩 - LabelEncoder

- 머신러닝 알고리즘은 string(=object) 형태의 값을 처리할 수 없기 때문에 이를 숫자형 값으로 변경해 주어야 함.
- scikit-learn의 preprocessing.LabelEncoder 클래스 활용
- string(=object) 형태의 값을 숫자형 값을 변경할 수 있음
- 공식 문서 : <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html>

```
>>> le = preprocessing.LabelEncoder()
>>> le.fit(["paris", "paris", "tokyo", "amsterdam"])
LabelEncoder()
>>> list(le.classes_)
['amsterdam', 'paris', 'tokyo']
>>> le.transform(["tokyo", "tokyo", "paris"])
array([2, 2, 1]...)
>>> list(le.inverse_transform([2, 2, 1]))
['tokyo', 'tokyo', 'paris']
```

데이터 인코딩 - LabelEncoder

- 원본 데이터

상품 분류	가격
TV	1,100,000
냉장고	1,800,000
전자레인지	250,000
컴퓨터	900,000
선풍기	100,000
믹서	100,000
믹서	100,000



- 상품 분류를 레이블 인코딩 한 데이터

상품 분류	가격
0	1,100,000
1	1,800,000
4	250,000
5	900,000
3	100,000
2	100,000
2	100,000

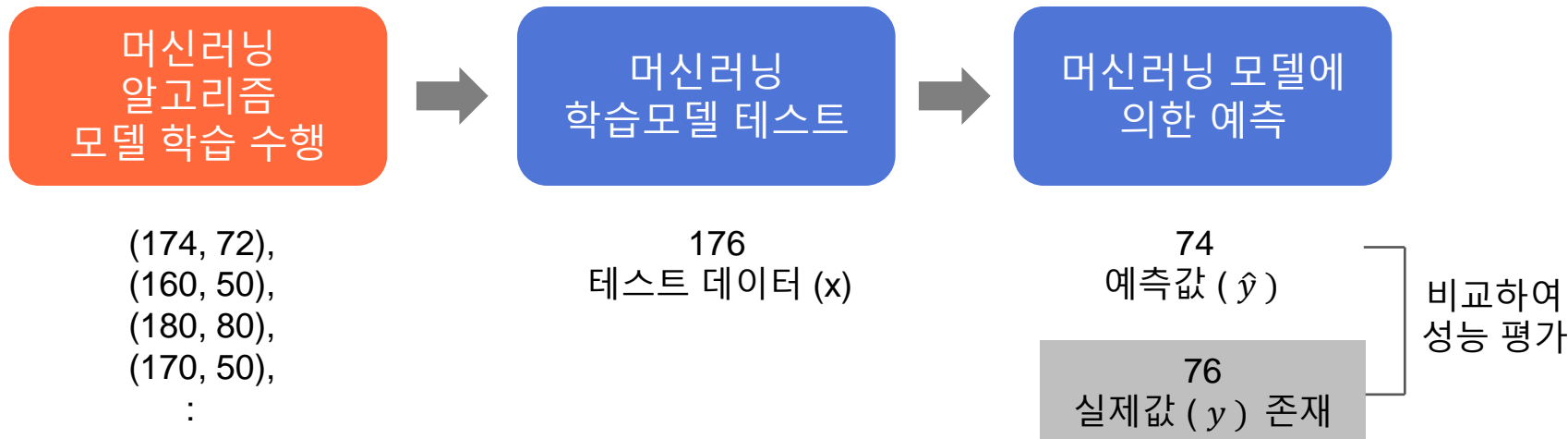
training data, test data 나누기

```
from sklearn.model_selection import train_test_split  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

- training data, test data를 나누는 메소드
 - X_train, X_test, y_train, y_test : 순서 꼭 지키기
 - test_size : default 0.25(train data 75%, test 25%)로 나눔
 - X_train, y_train : 학습데이터
 - X_test : 예측을 위한 데이터
 - y_test : 성능 평가를 위한 데이터

Training Data, Test Data

- Training Data(학습용 데이터)
 - 머신러닝 모델을 학습시킬 때 사용하는 데이터
- Test Data(테스트 데이터)
 - 학습모델의 성능을 테스트 할 때 사용하는 데이터
 - Training Data로 사용하지않은 새로운 데이터



[실습2]titanic 사고 데이터로 생존자 예측

- 입력 데이터 : sex, age, sibsp ...
 - 12Dimension (survived, embark_town, alive 컬럼 제외)
- Target data : survived 컬럼
 - 0 (사망), 1(생존)
 - Binary Classification
- 학습 모델 알고리즘(Estimator)
 - DecisionTreeClassifier
- 추가적인 적용기법
 - EDA(Exploratory Data Analysis)
 - Data Cleansing(결측치 처리)

[실습3]Decision Tree로 titanic 사고 생존자 예측

- 3-2-3.지도학습_classification_decision_tree_titanic_alive_pred.ipynb



train_test_split()

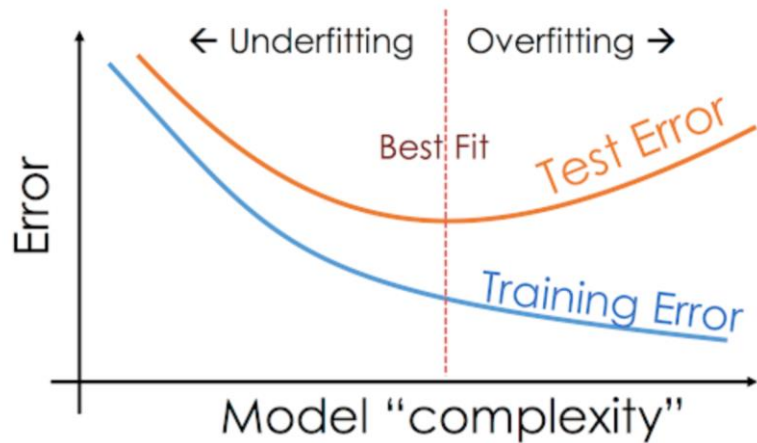
sklearn.model_selection의 train_test_split() 함수의 파라미터

```
X_train, X_test, y_train, y_test = train_test_split(iris_data.data, iris_data.target, test_size=3.0, random_state=121)
```

- iris_data.data : 학습을 위한 전체 feature 데이터 셋
- iris_data.target : 학습을 위한 전체 label 값
- test_size : 전체 데이터에서 테스트 데이터 셋 크기를 얼마로 할지 결정
디폴트 0.25%
- train_size : 전체 데이터에서 학습용 데이터 셋 크기를 얼마로 할지 결정
- shuffle : 테스트를 분리하기 전에 데이터를 미리 섞을지 결정
디폴트 True
- random_state : 호출할 때마다 동일한 학습/테스트용 데이터 셋을 생성하기 위해 random seed값 설정

지도학습 학습모델 성능평가

Overfitting, Underfitting

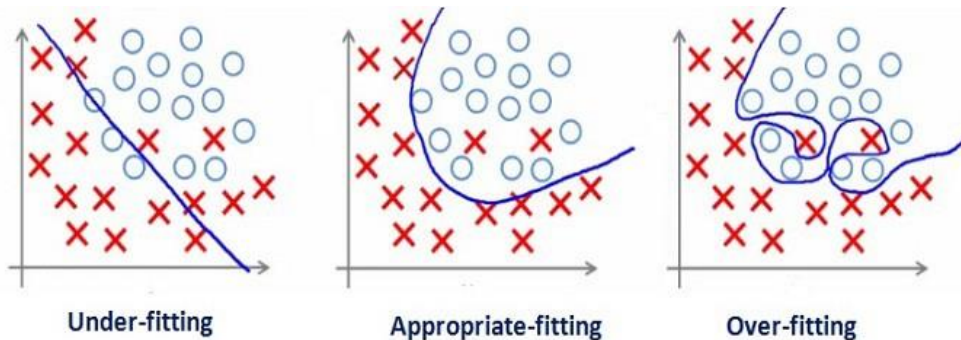


[트레이닝 에러와 검증 에러 출력]

- 처음에는 트레이닝 에러와 검증 에러가 모두 작아지지만 일정 회수 이상 반복할 경우 트레이닝 에러는 작아지지만 검증 에러는 커지는 오버피팅(Overfitting, 과최적화)에 빠지게 됨
- 트레이닝 에러는 작아지지만 검증에러는 커지는 지점에서 업데이터를 중지 (Early Stopping)하면 최적의 파라미터를 얻을 수 있음

Overfitting, Underfitting

- 오버피팅(Overfitting)
 - 학습 과정에서 머신러닝 알고리즘의 파라미터가 트레이닝 데이터에 과도하게 최적화되어 트레이닝 데이터에 대해서는 잘 동작하지만 새로운 데이터인 테스트 데이터에 대해서는 잘 동작하지 못하는 현상
 - 오버피팅을 방지하기 위한 기법을 Regularization 이라고 함.
- 언더피팅(Underfitting)
 - 모델의 표현력이 부족해서 트레이닝 데이터도 제대로 예측하지 못하는 상황



지도학습 성능 평가 방법

Classification (분류)

- 정확도(accuracy)
- 정밀도(precision)
- 재현율(recall)
- F1-score
- ROC / AUC

Regression (회귀)

- MAE(Mean Absolute Error)
- RMSE(Root Mean Squared Error)
- R2 Score(Coefficient of Determination, 결정계수)
- ROC / AUC

[실습4]지도학습 성능평가 정리



- 3-2-4.지도학습_classification_학습모델_평가_정리.ipynb

분류(Classification) 알고리즘들

- 데이터의 피처와 레이블 값으로 구성하여 머신러닝 학습 모델 생성
- 생성된 모델에 새로운 데이터의 피처가 주어졌을 때 레이블을 예측

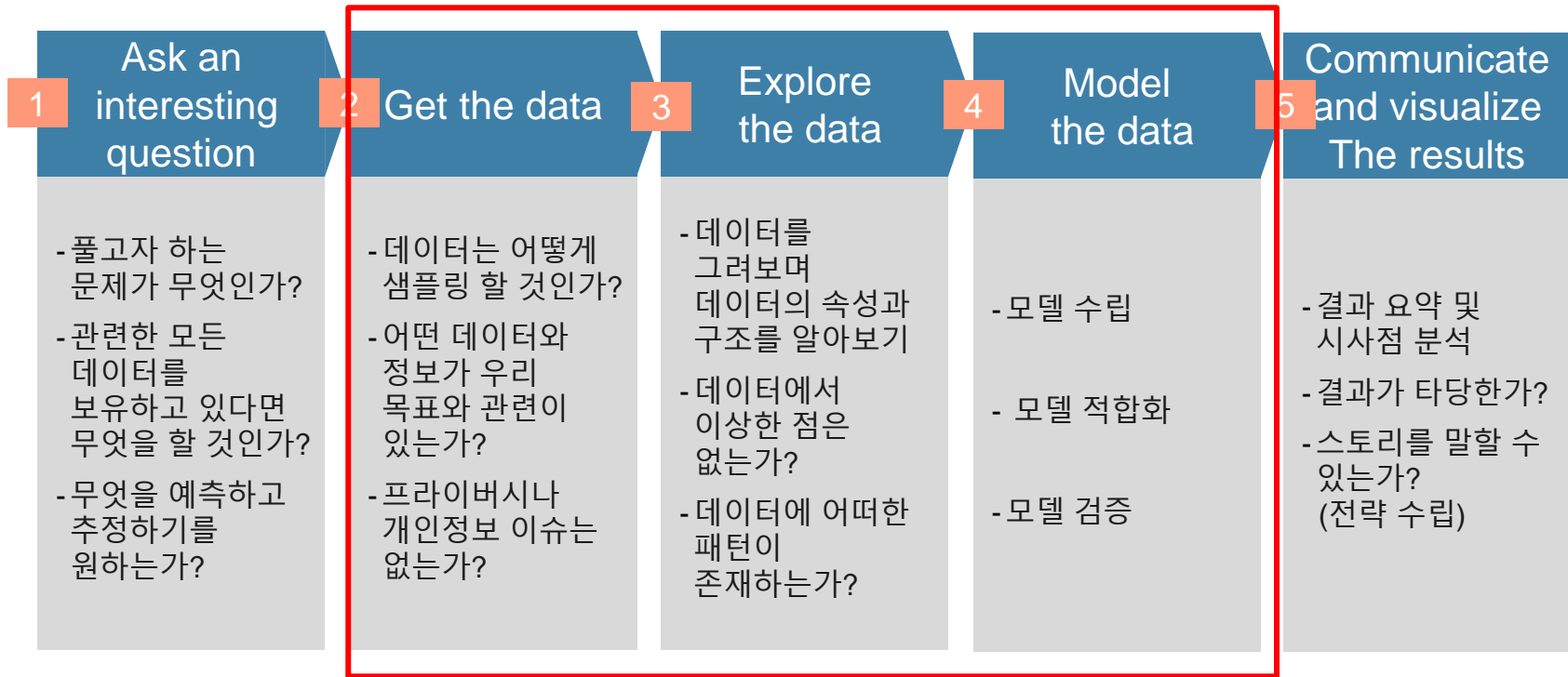
대표적인 분류 알고리즘

- 결정트리 : 데이터 균일도에 따른 규칙 기반
- SGDClassifier : 확률적 경사 하강법에 기반한 분류
- 로지스틱 회귀 : 독립변수와 종속변수의 선형 관계성에 기반
- 나이브 베이즈 : 베이즈(Bayes) 통계와 생성 모델에 기반
- SVM(서포트 벡터 머신) : 개별 클래스 간의 최대 분류 마진을 효과적으로 찾아 줌
- 최소근접(Nearest Neighbor) : 근거리를 기준 분류함
- 앙상블(Ensemble) : 서로 다른(또는 같은) 머신러닝 알고리즘을 결합함
- Neural Network : 심층연결 기반의 신경망

머신러닝 프로젝트 절차

머신러닝 프로젝트 절차

- 데이터 기반의 문제해결 5단계



[1]문제정의(Ask an interesting question)

- 현상을 정확히 파악하여 **진짜문제 정의**
- **풀고자 하는 문제**가 무엇인가?
- 관련한 모든 데이터를 보유하고 있다면 무엇을 할 것인가?
- 무엇을 예측하고 추정하기를 원하는가?

[2]데이터 확보(Get the data)



좋은 성능 => 좋은 데이터(Quality, Quantity)

데이터 가공 => 전처리(pre-processing)

**데이터 분석에 적합하게 데이터를
가공 / 변형 / 처리 / 클린징**

Garbage in, Garbage out!

[2]데이터 확보 - 전처리(pre-processing)

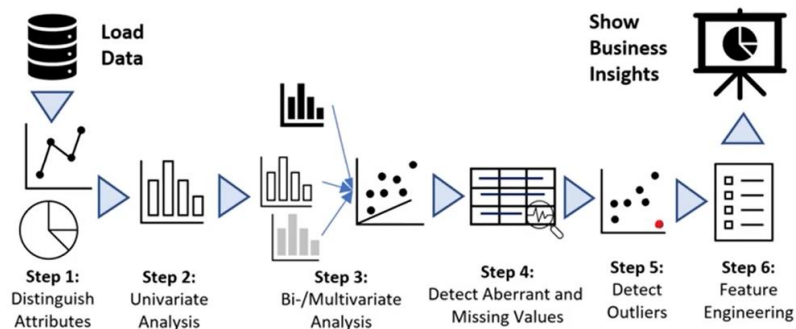
- 분석가의 80% 시간을 데이터 수집 및 전처리에 사용하고 있음.

전처리 방법

- 결측치 – imputer
- 이상치
- 정규화(Normalization)
- 표준화(Standardization)
- 샘플링(over/under sampling)
- 피처공학(Feature Engineering)
 - feature 생성/ 연산
 - 구간 생성, 스케일 변형

[3]데이터 탐색 및 이해(Explore the data)

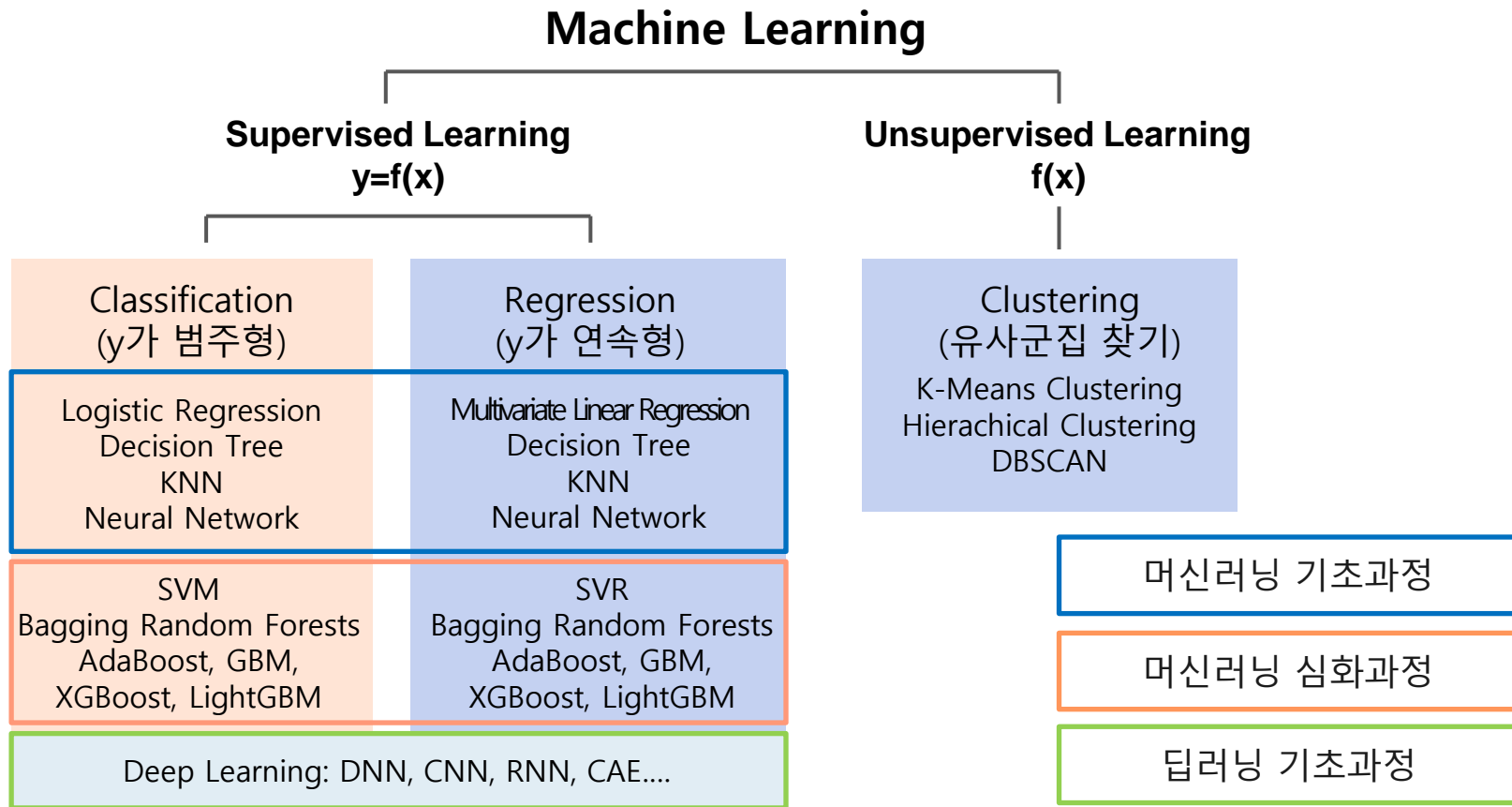
- 성급한 모델링 이전에 **충분히 데이터를 탐색하라**
 - 단변량 분석, 상관관계분석, 결측치/이상치 탐지 및 제거
 - Feature Engineering등 고전적인 데이터 전처리 방식은 여전히 유효함.
 - 데이터 시각화 툴/라이브러리 활용



Taboada, G. L., Seruca, I., Sousa, C., & Pereira, Á. (2020). Exploratory data analysis and data envelopment analysis of construction and demolition waste management in the European Economic Area. Sustainability, 12(12), 4995.

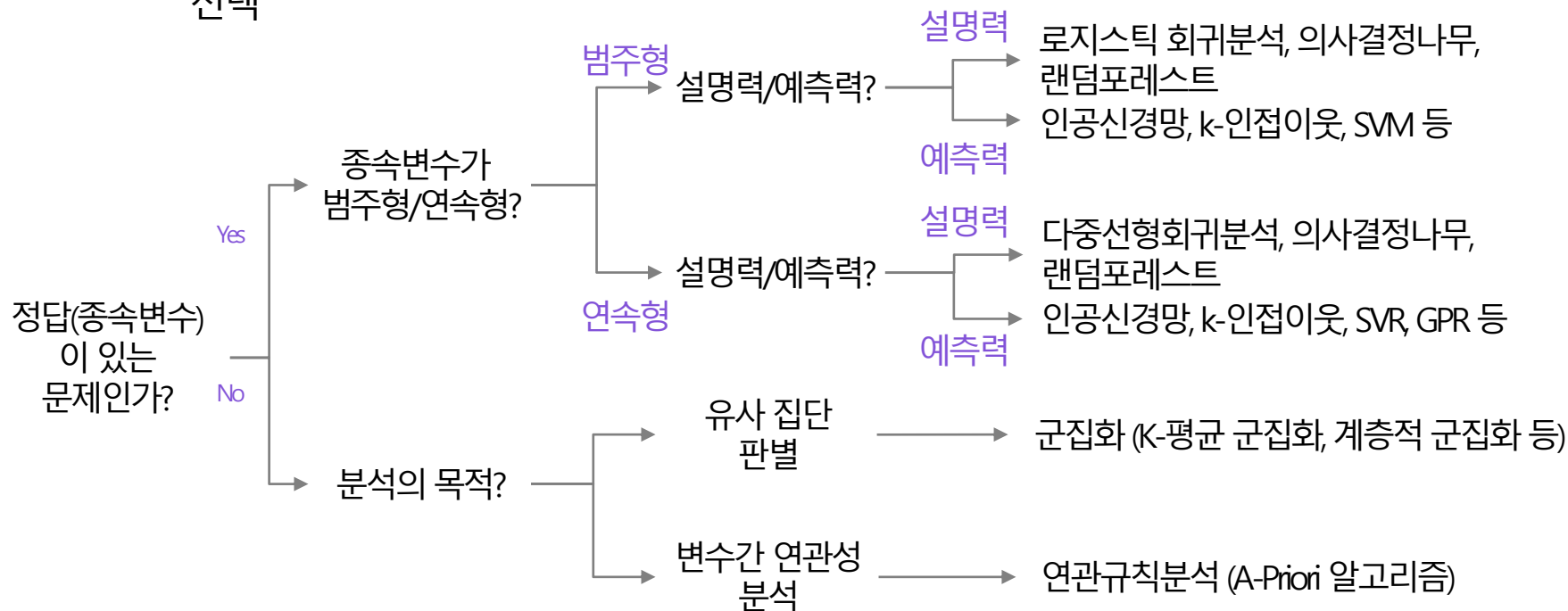
**데이터, 예측해야 할 값에 맞는
알고리즘 사용**

[4]머신러닝 학습모델(Model the data) - 모델선택



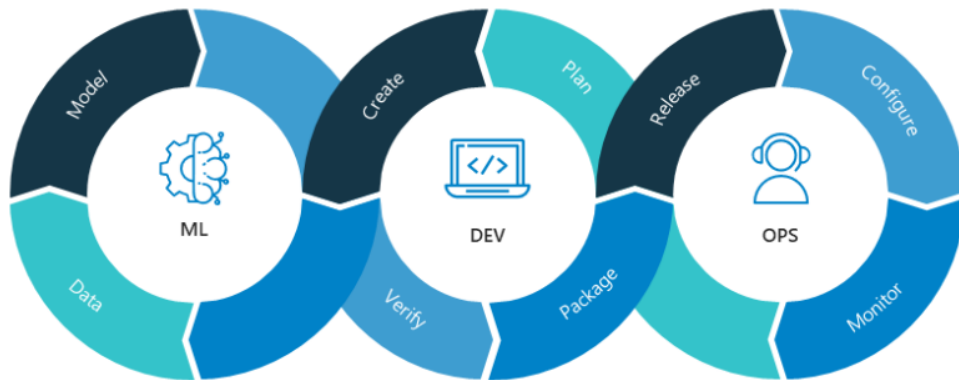
[4]머신러닝 학습모델(Model the data) - 모델선택

- 주어진 문제를 가장 잘 해결할 수 있는 모델 선택
 - 질문의 속성, 데이터의 특징, 결과의 설명력 포함 유무 등을 고려하여 적합한 분석 알고리즘 선택



[5]머신러닝 - 적용 및 개선

- MLOps: Model Design(DataOps) + Model Development + DevOps



- 문제정의
- 모델 디자인
- 현업 전문가

- ML연구자, 개발자
- 모델, 학습, 평가, 테스트, 패키징

- ML 서비스 개발자

- 1000

