
머신러닝 실습



실습 목표

- 파이썬 머신러닝 모델 개발환경 이해
- 파이썬 기본문법 학습
- 머신러닝 모델개발 실습 : 분류(Classification), 회귀(Regression)



Contents

- 1. 환경설정
- 2. 파이썬 문법
- 3. 머신러닝 실습 1 : 분류
- 4. 머신러닝 실습 2 : 회귀

Contents

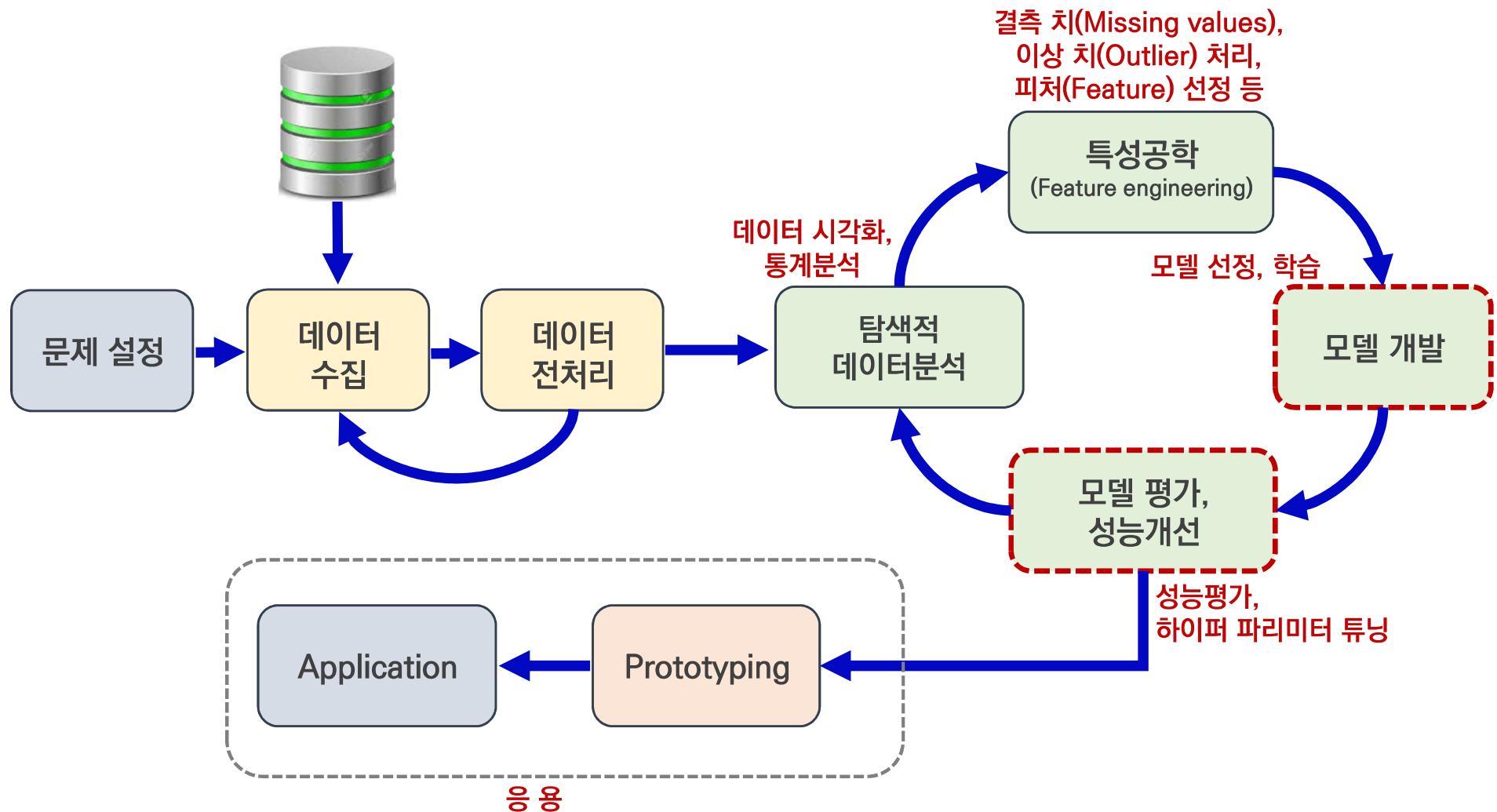
- 1. 환경설정
- 2. 파이썬 문법
- 3. 머신러닝 실습 1 : 분류
- 4. 머신러닝 실습 2 : 회귀

1. 환경설정

1.1 머신러닝 모델 개발절차

✓ 머신러닝 모델 개발 절차

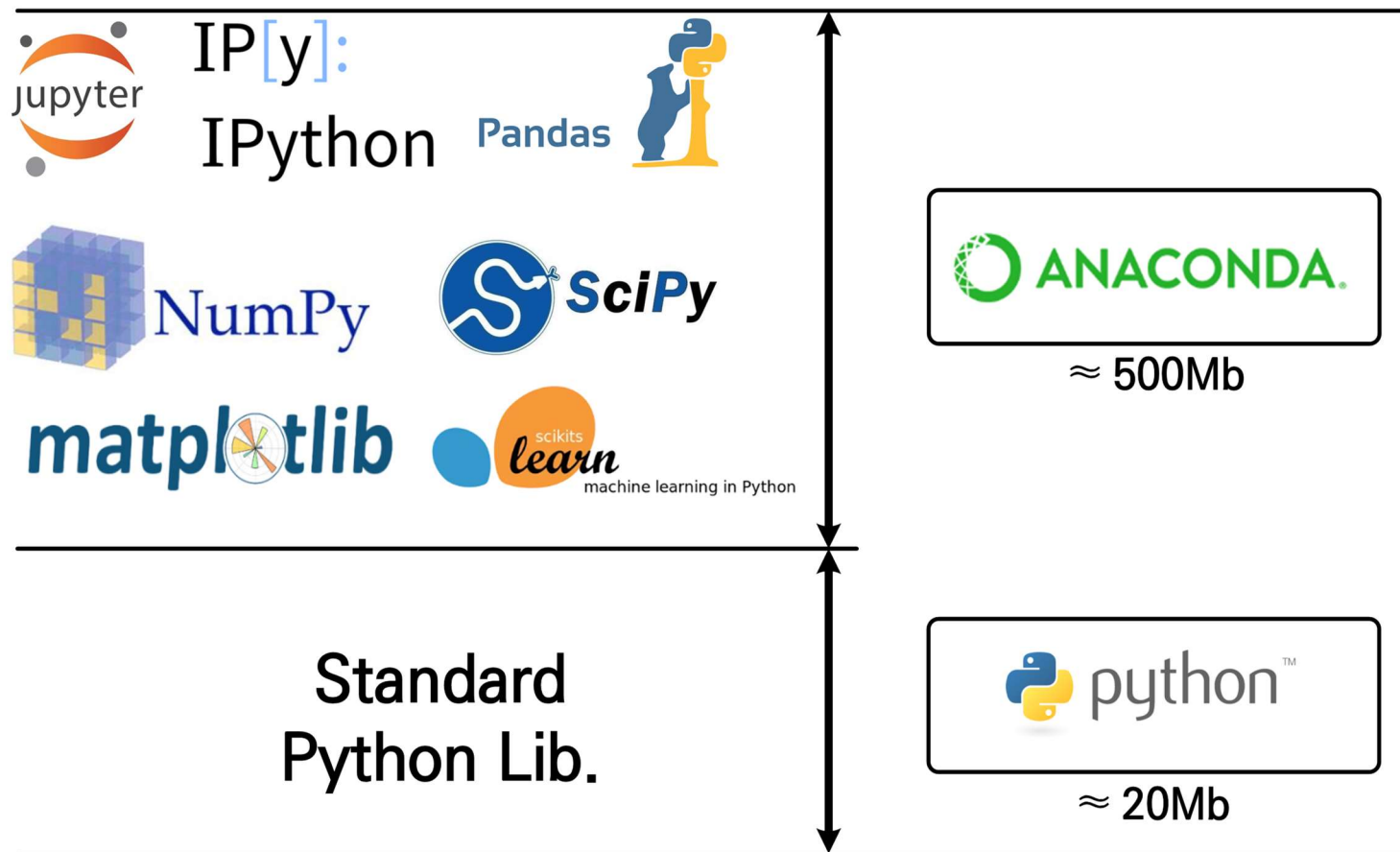
 실습 범위



1. 환경설정

1.2 아나콘다

✓ www.anaconda.com → Download, Install



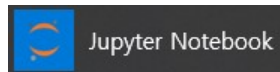
1. 환경설정

1.3 주피터 노트북

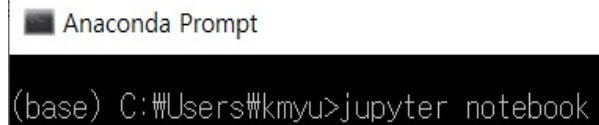
✓ 주피터 노트북(Jupyter Notebook)

- 웹 브라우저에서 파이썬 코드를 작성하고 실행해볼 수 있는 개발 도구
- 실행방법

- 방법1) 아이콘 클릭



- 방법2) 커맨드 창에서 'jupyter notebook' 입력

The image shows a terminal window titled "Anaconda Prompt". The prompt is "(base) C:\Users\kmyu>". The command "jupyter notebook" has been entered after the prompt.

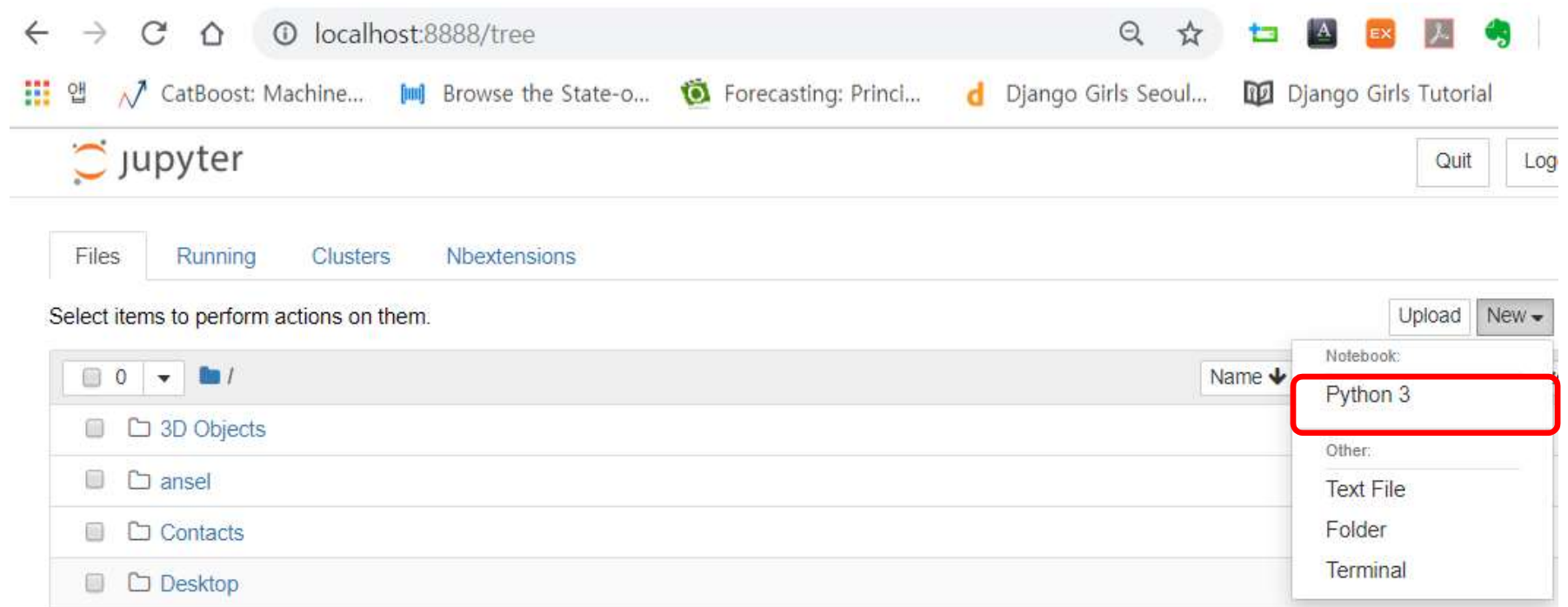
```
(base) C:\Users\kmyu>jupyter notebook
```

1. 환경설정

1.3 주피터 노트북

✓ 주피터 노트북(Jupyter Notebook)

- 웹 브라우저에서 파이썬 코드를 작성하고 실행해볼 수 있는 개발도구

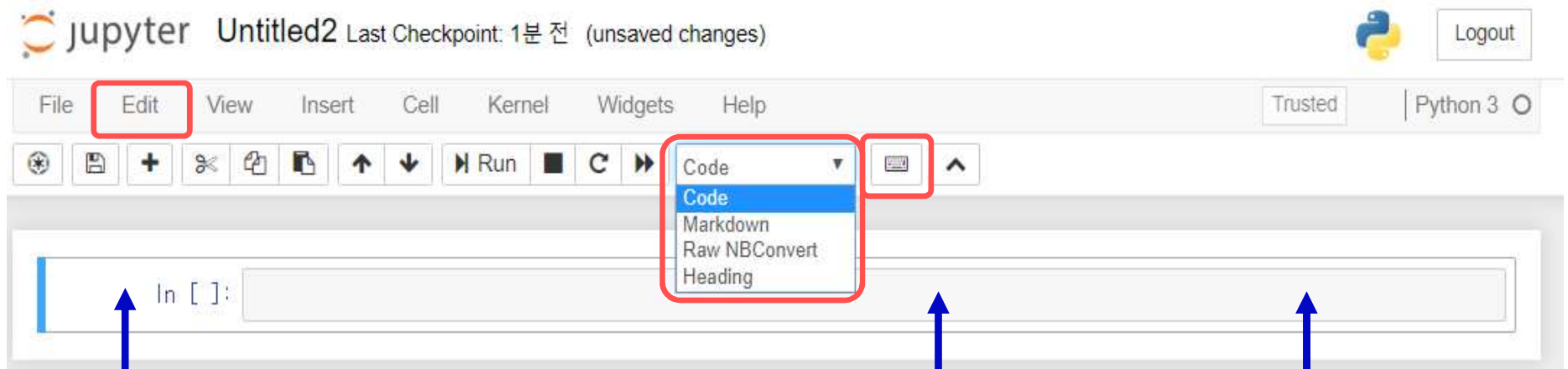


1. 환경설정

1.3 주피터 노트북

✓ 주피터 노트북(Jupyter Notebook)

- 웹 브라우저에서 파이썬 코드를 작성하고 실행해볼 수 있는 개발 도구



커맨드 모드

마크다운 언어 또는
파이썬 코드 입력

에디트 모드

1. 환경설정

실습 1-1

✓ 마크다운 언어(Mark Down)

- 일반 텍스트 문서의 양식을 편집하는 문법
- HTML에 비해 사용이 간단함

한국전력 신입사원 교육

한국전력 신입사원 교육

한국전력 신입사원 교육

한국전력 신입사원 교육

- 한국전력 신입사원 교육
- 입소를 축하드립니다.



▶ Run

or

Ctrl + Enter

or

Shift + Enter

한국전력 신입사원 교육

한국전력 신입사원 교육

한국전력 신입사원 교육

한국전력 신입사원 교육

- 한국전력 신입사원 교육
- 입소를 축하드립니다.

1. 환경설정

실습 1-2

✓ 파이썬 코드

```
company = '한국전력'  
blank = ''  
rank = '신입사원'  
print(company+blank+rank)
```

executed in 6ms, finished 22:14:43 2019-10-26

한국전력 신입사원

```
a = 10  
b = 2  
c = a+b  
print(a+b)
```

executed in 5ms, finished 22:18:44 2019-10-26

12

```
print('result is', a/b)
```

executed in 4ms, finished 22:18:44 2019-10-26

result is 5.0

```
print('곱셈: {0}, 나눗셈: {1:.2f}'.format(a*b, a/b))
```

executed in 3ms, finished 22:20:27 2019-10-26

곱셈: 20, 나눗셈: 5.00

Contents

- 1. 환경설정
- 2. 파이썬 문법
- 3. 머신러닝 실습1 : 분류
- 4. 머신러닝 실습2 : 회귀

2. 파이썬 문법

2.1 내장 자료형(built-in types)

파이썬 자료형 : Boolean(True/ False), Numeric, Sequence(list, tuple), Text(string), Dictionary 등

✓ Boolean

- 참과 거짓을 나타내는 자료형
- True, False 두 가지 값만 가짐

```
a = True  
b = False
```

```
type(a) # 변수의 자료형(타입)을 확인하는 함수
```

```
bool
```

✓ Numeric

- Int(정수형), float(실수형)
- 산술연산 가능(+, -, *, /, **)

```
a = 5  
a
```

```
5
```

```
b = 3.5  
b
```

```
3.5
```

✓ String

- 문자열

```
x2 = 'Python is fun.'  
x2
```

```
'Python is fun.'
```

2. 파이썬 문법

2.1 내장 자료형(built-in types)

✓ List

- 순서가 있는 원소들의 집합
- 대괄호 []로 묶어주고 콤마(,) 로 원소를 구분

```
a = [1, 2, 3, 4, 5]
```

✓ Dictionary

- key와 value로 구성
- 중괄호 {}로 묶어주고, 콤마(,) 로 원소를 구분, 콜론(:)으로 key와 value를 구분

```
# 한 줄로 써도 됩니다.
```

```
dic = {'name': 'Tom',  
       'phone': '01012345678',  
       'birth': '900101'}
```

```
dic
```

```
{'name': 'Tom', 'phone': '01012345678', 'birth': '900101'}
```

2. 파이썬 문법

2.2 조건절 if

✓ if문

- 어떤 조건을 판단해 그에 해당하는 상황을 수행하도록 하는 것
- if, elif, else로 나타냄(elif와 else는 생략가능)
- 수행할 내용은 들여쓰기를 해야함

```
if 조건:
    <수행할 내용>
elif 조건:
    <수행할 내용>
else:
    <수행할 내용>
```

```
temp = 28

if temp > 30:
    print("아이스크림")
elif temp > 27 and temp < 30:
    print("물")
else:
    print("빈 손")
```

물

2. 파이썬 문법

2.3 반복문 for

✓ for 문

- 코드를 반복해서 수행해야 할 경우에 사용

for 변수 in 리스트(또는 문자열, 튜플):
 <수행할 내용>

```
lst = ['one', 'two', 'three']  
  
for i in lst:  
    print(i)
```

```
one  
two  
three
```

```
lst = [i for i in range(0, 10)]  
lst
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```


2. 파이썬 문법

2.4 함수

✓ 함수

- 반복적으로 사용되는 부분을 묶은 것
- 입력 값을 받아 어떤 일을 수행한 후 그 결과를 돌려줌

```
def 함수명(매개변수):  
    <수행할 내용>
```

```
def add(a, b):  
    return a + b
```

```
c = add(1, 3)  
c
```

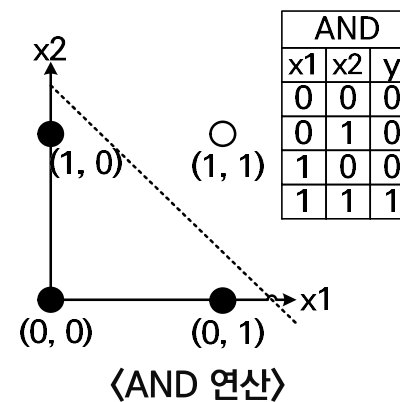
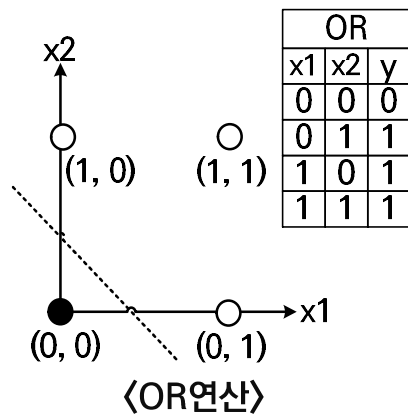
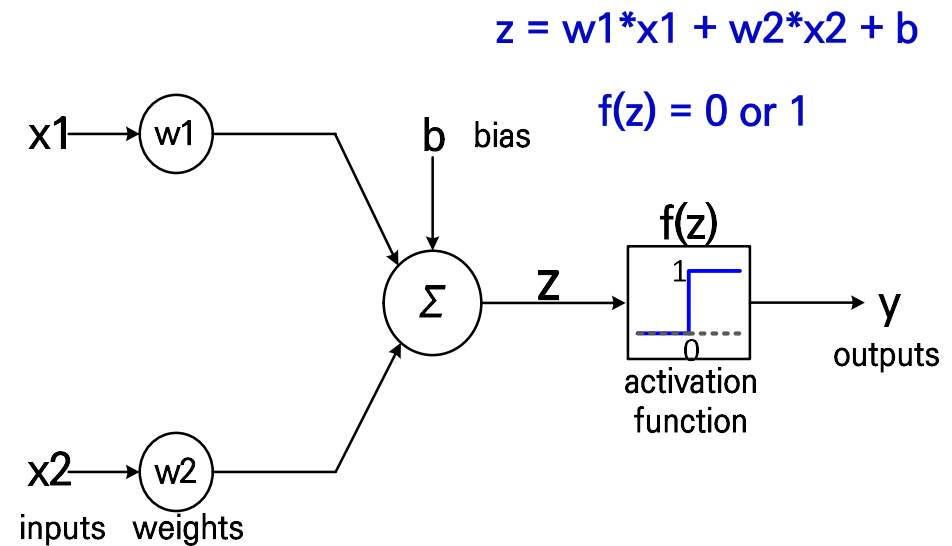
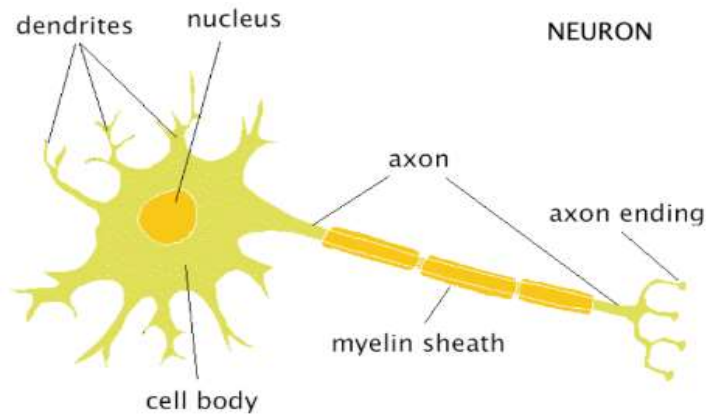
4

2. 파이썬 문법

실습 2-1 : MCP 뉴런 구현하기

✓ MCP Neuron

- AND, OR, NAND 논리회로 만들기



2. 파이썬 문법

실습 2-1 : MCP 뉴런 구현하기

✓ Numpy

- 행렬이나 다차원 배열을 쉽게 처리할 수 있도록 하는 라이브러리



```
data = list(range(5))
print(data)
print(data*2)
print([n*2 for n in data])
```

executed in 5ms, finished 00:06:08 2019-11-22

```
[0, 1, 2, 3, 4]
[0, 1, 2, 3, 4, 0, 1, 2, 3, 4]
[0, 2, 4, 6, 8]
```

〈List 연산〉

```
import numpy as np
arr1d = np.array(data)
print(arr1d)
print(arr1d*2)
```

executed in 5ms, finished 00:08:15

```
[0 1 2 3 4]
[0 2 4 6 8]
```

〈Numpy array 연산〉

```
data2 = [[0, 1, 2], [3, 4, 5], [6, 7, 8]]
arr2d = np.array(data2)
print(arr2d)
```

executed in 4ms, finished 00:10:53 2019-11-22

```
[[0 1 2]
 [3 4 5]
 [6 7 8]]
```

✓ pandas

- 데이터의 수정, 조작 등 다양한 기능을 제공하는 라이브러리
- 데이터 프레임(Data Frame)

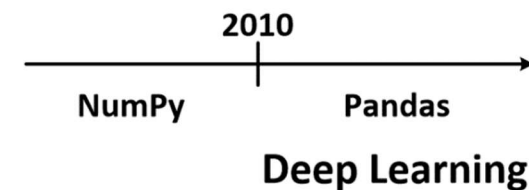
```
data.head()
```

executed in 24ms, finished 20:05:51 2019-11-21

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2



Pandas

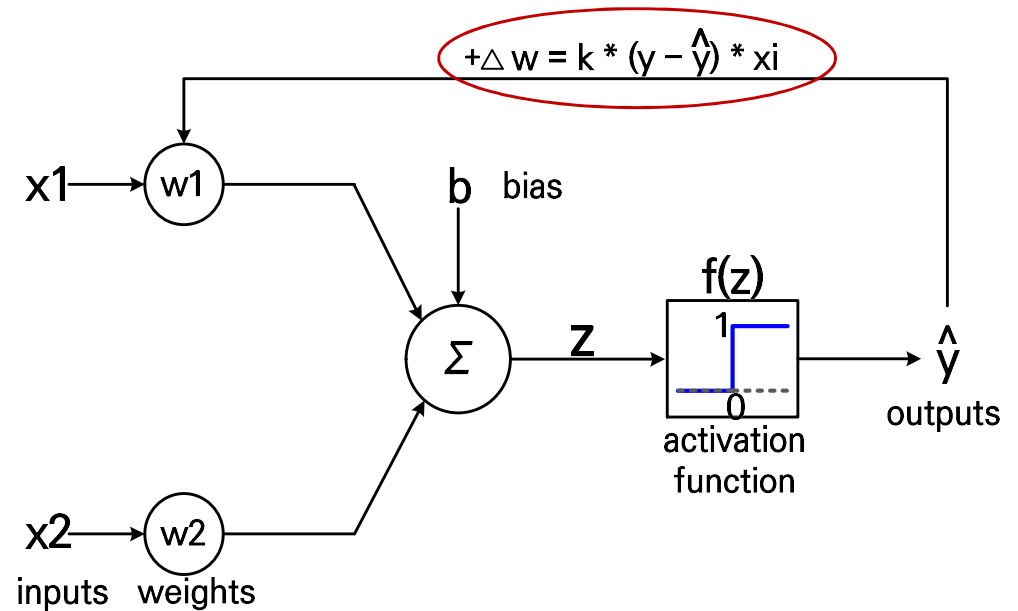
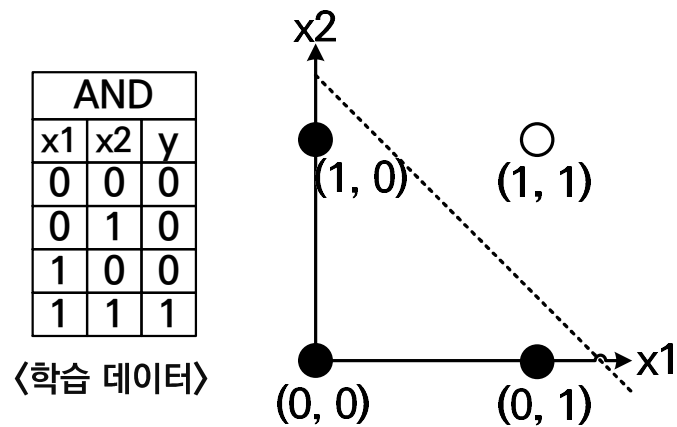


2. 파이썬 문법

실습 2-2 : Perceptron 구현하기

✓ Perceptron

- AND, OR, NAND 논리회로 만들기



2. 파이썬 문법

실습 2-2 : Perceptron 구현하기

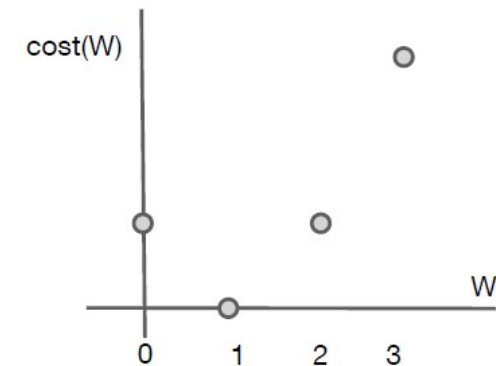
✓ Perceptron

- 가중치 W 업데이트(학습) 방법 : 경사하강법(Gradient descent)을 활용한 비용함수(cost function) 최적화

$$\text{cost}(W) = (1/m) \sum_{i=1}^m (W \cdot x_i - y_i)^2$$

x	y
1	1
2	2
3	3

- $W = 0, \text{cost}(W) = 4.67$
 $\frac{1}{3}((0 \cdot 1 - 1)^2 + (0 \cdot 2 - 2)^2 + (0 \cdot 3 - 3)^2)$
- $W = 1, \text{cost}(W) = 0$
 $\frac{1}{3}((1 \cdot 1 - 1)^2 + (1 \cdot 2 - 2)^2 + (1 \cdot 3 - 3)^2)$
- $W = 2, \text{cost}(W) = 4.67$
 $\frac{1}{3}((2 \cdot 1 - 1)^2 + (2 \cdot 2 - 2)^2 + (2 \cdot 3 - 3)^2)$
- $W = 3, \text{cost}(W) = 18.67$
 $\frac{1}{3}((3 \cdot 1 - 1)^2 + (3 \cdot 2 - 2)^2 + (3 \cdot 3 - 3)^2)$

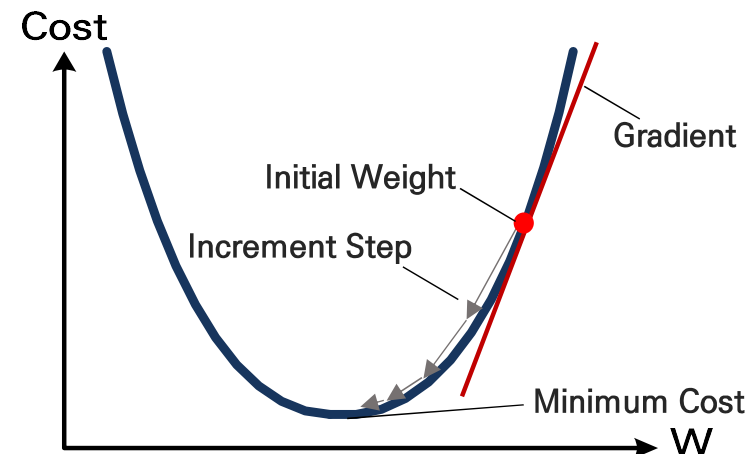


- 최적화 : minimize cost(W)
- 경사하강법을 통한 업데이트 Rule

$$W := W - \alpha \frac{\partial}{\partial W} \text{cost}(W)$$

$$W := W - \alpha \frac{1}{m} \sum_{i=1}^m (W(x_i) - y_i)x_i$$

↑
Increment Step
(Learning rate)



Contents

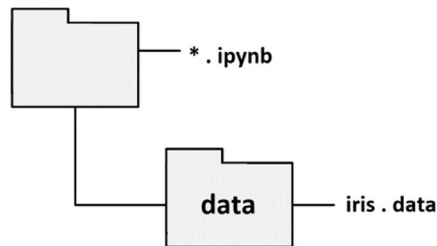
- 1. 환경설정
- 2. 파이썬 문법
- 3. 머신러닝 실습1 : 분류
- 4. 머신러닝 실습2 : 회귀

3. 머신러닝 실습 1 : 분류

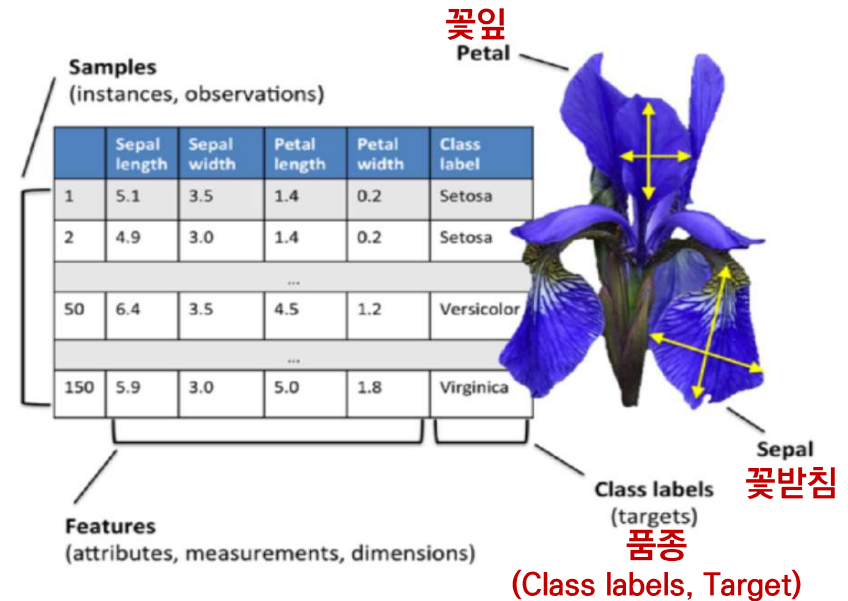
실 습 3-1 : IRIS(붓꽃) 품종 분류

✓ 데이터 셋 확인

- features, samples, class labels



〈데이터 저장위치〉



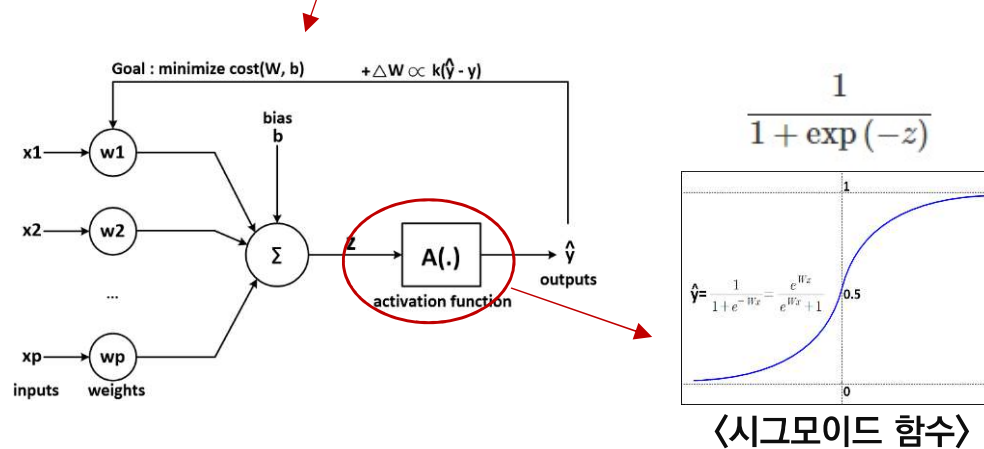
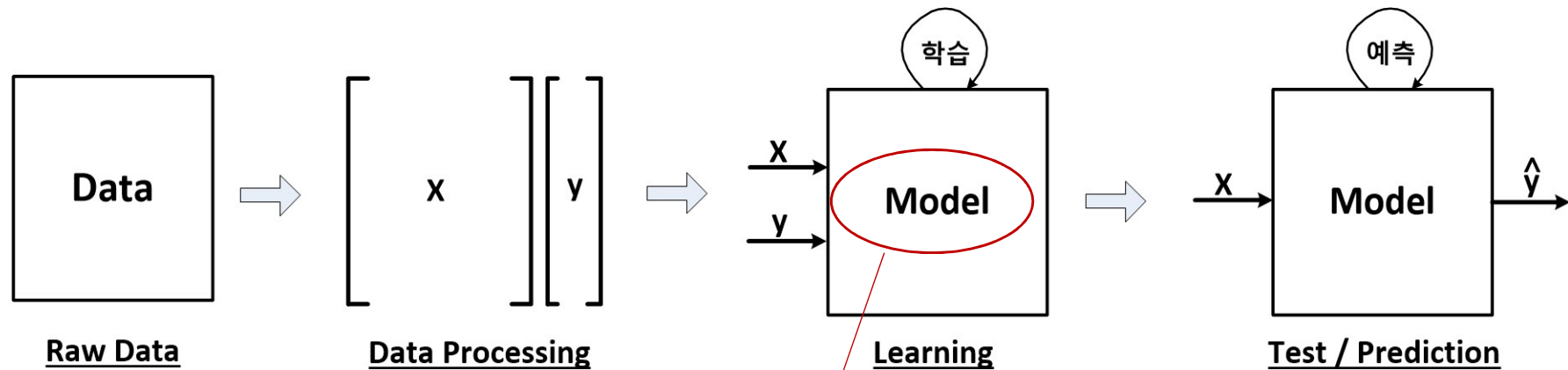
features (X)					
	0	1	2	3	4
sample 0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

3. 머신러닝 실습 1 : 분류

실 습 3-1 : IRIS(붓꽃) 품종 분류

✓ 첫번째 머신러닝 모델

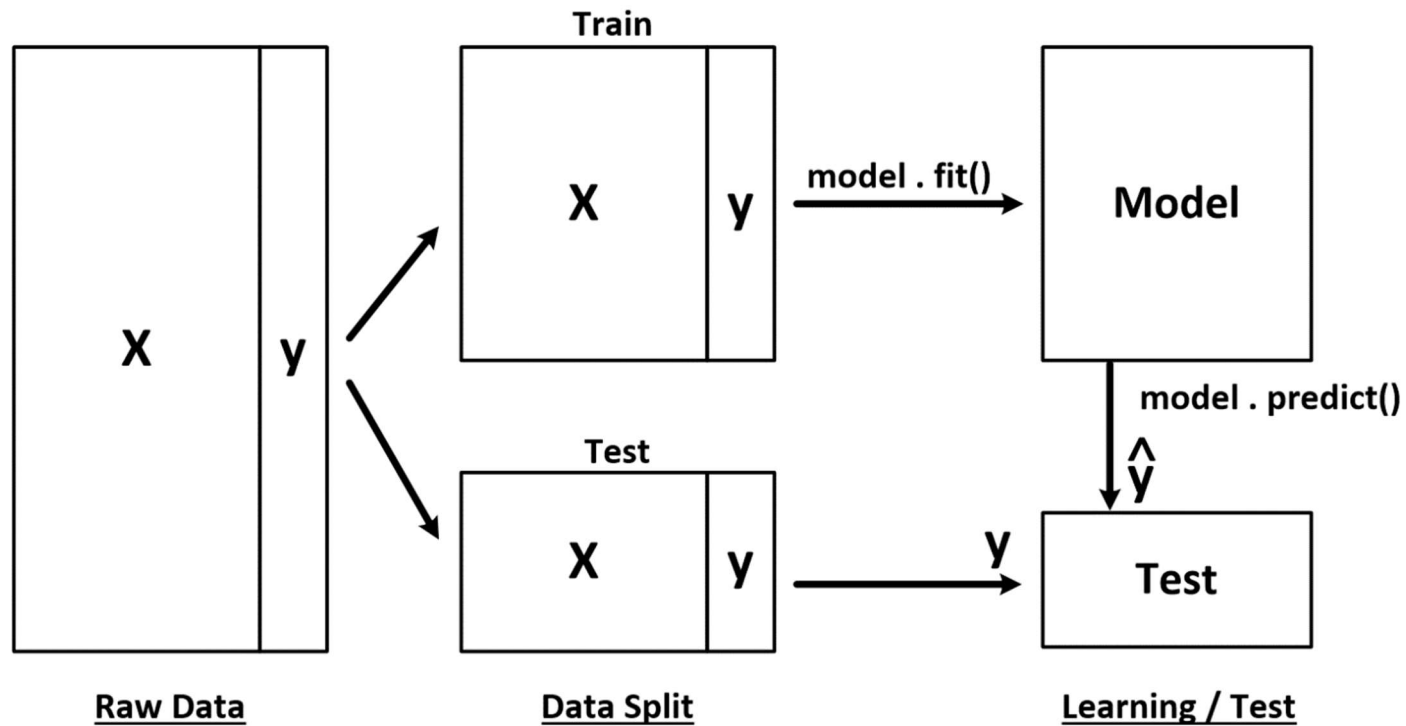
- 로지스틱 회귀(Logistic Regression)



3. 머신러닝 실습 1 : 분류

실 습 3-1 : IRIS(붓꽃) 품종 분류

- ✓ 데이터 셋을 분리하여 학습/ 검증하기



3. 머신러닝 실습 1 : 분류

실 습 3-2 : 와인 등급 분류(Option)

✓ 데이터 셋 확인

- features, samples, class labels

Y		X								
	Class label	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavnoid phenols	Proant
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	
2	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	...
3	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	
4	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	
5	1	14.20	1.76	2.45	15.2	112	3.27	3.39	0.34	

$$\begin{array}{c} \text{분류개수} \downarrow \\ \begin{bmatrix} w_{1.1} & w_{1.2} & \dots & w_{1.p} \\ w_{2.1} & w_{2.2} & \dots & w_{2.p} \\ w_{3.1} & w_{3.2} & \dots & w_{3.p} \end{bmatrix} \\ W \end{array} \xrightarrow{\text{특징개수} \rightarrow} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_p \end{bmatrix} X$$

Contents

- 1. 환경설정
- 2. 파이썬 문법
- 3. 머신러닝 실습1 : 분류
- 4. 머신러닝 실습2 : 회귀

4. 머신러닝 실습 2 : 회귀

실습 4-1 : 보스턴 주택가격 예측

✓ 데이터 셋 확인

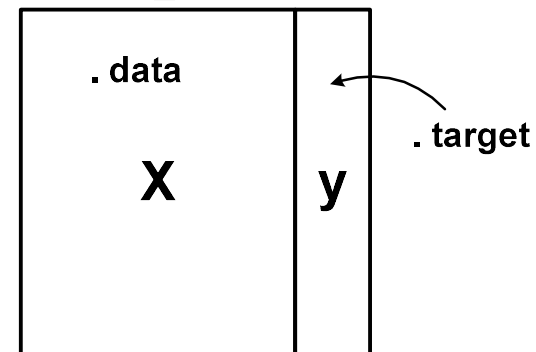
- Bunch 데이터 셋

	X													y
	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2

Bunch

- . data → X
- . target → y
- . feature_names
- . DESCR → Readme

. feature_names

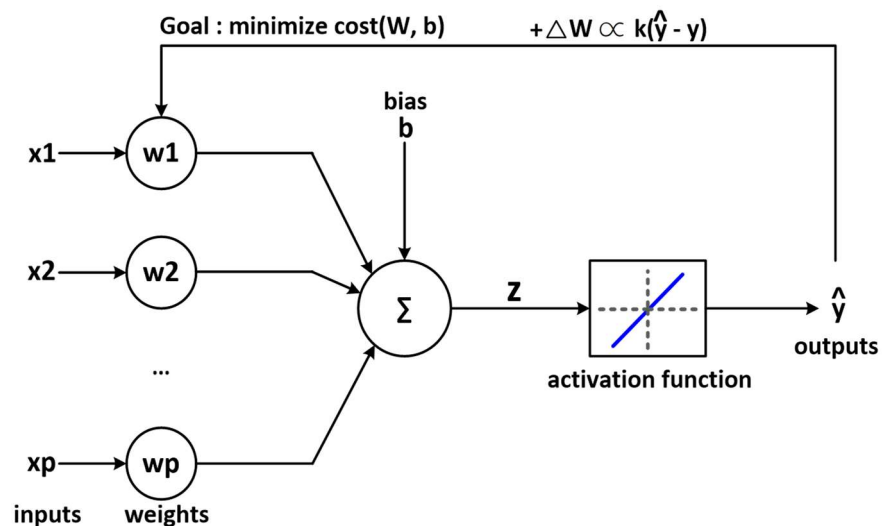


4. 머신러닝 실습 2 : 회귀

실 습 4-1 : 보스턴 주택가격 예측

✓ 회귀용 머신러닝 모델

· 선형회귀(Linear regression)



$$\hat{y} = [x_1 \ x_2 \ \dots \ x_n] \cdot \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_n \end{bmatrix}$$
$$= w_1 x_1 + w_2 x_2 + \dots + w_n x_n$$

· 평가지표

$$score_{regression} = R^2 = 1 - \frac{\sum_{i=1}^m (y_i - \hat{y}_i)^2}{\sum_{i=1}^m (y_i - \bar{y})^2}$$

⟨R² score⟩

$$score_{classification} = \text{mean}(y == \hat{y})$$

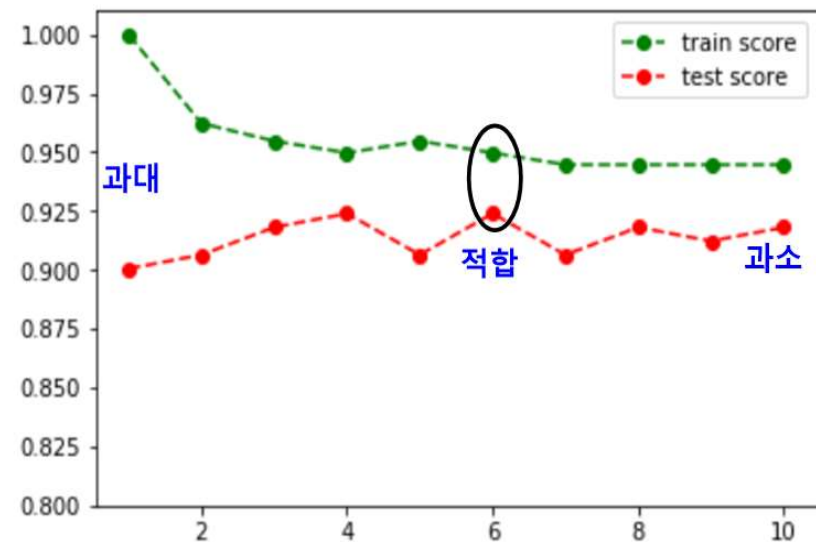
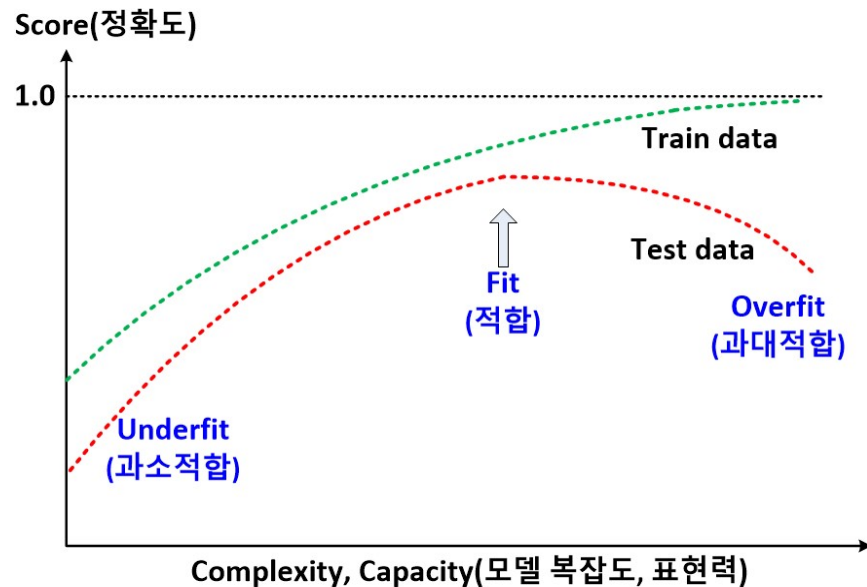
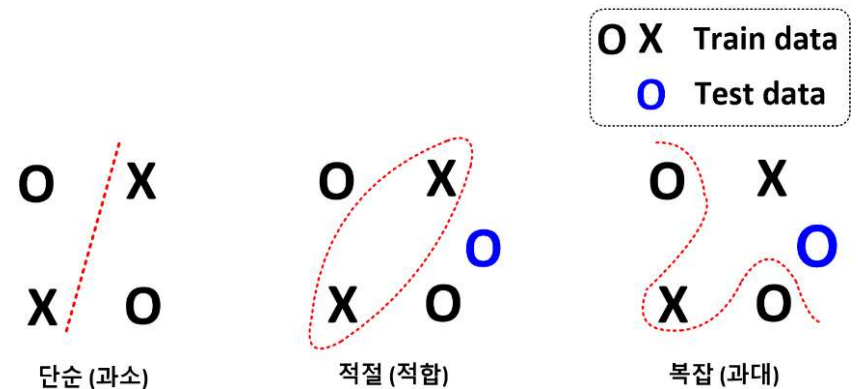
⟨Accuracy⟩

4. 머신러닝 실습 2 : 회귀

과대적합과 과소적합

✓ 좋은 모델이란?

- 학습/ 시험 데이터 모두에서 성능이 우수 해야함(최적화)
- 학습/ 시험 데이터 점수가 유사 해야함(일반화)

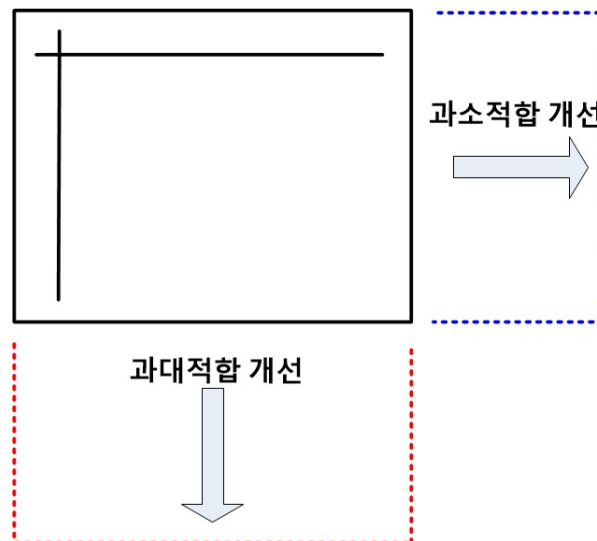


4. 머신러닝 실습 2 : 회귀

과대적합과 과소적합

✓ 모델성능 개선 방법 1

- 과소적합인 경우 : 피쳐 늘리기
- 과대적합인 : 샘플수 늘리기



· 우수한 모델 사용하기

과
대
적
합

Logistic regression
Decision tree
Random forest
Gradient boosting
Support vector machine

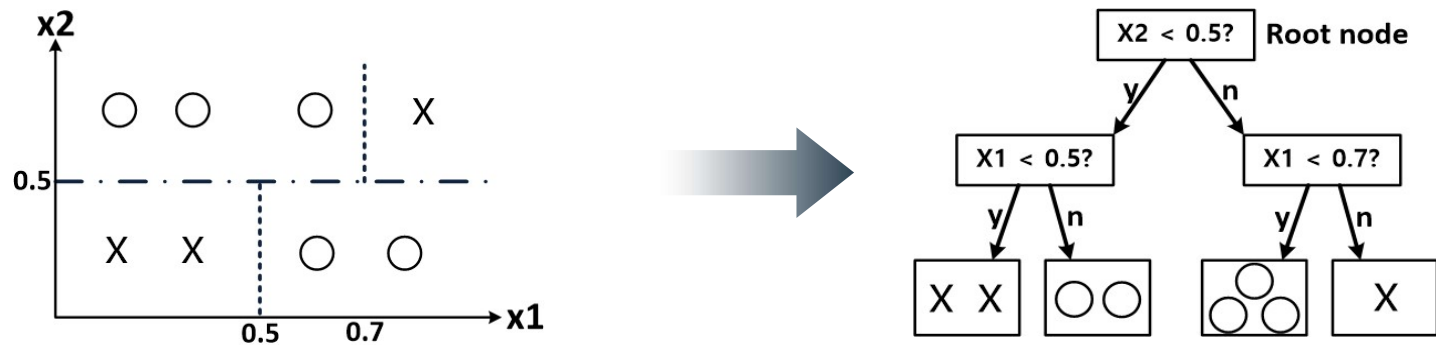
과
소
적
합

Linear regression
Decision tree
Ridge, Lasso
Gradient boosting
Random forest

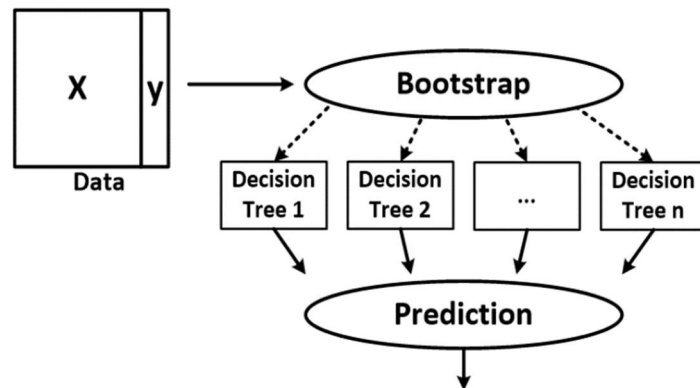
4. 머신러닝 실습 2 : 회귀

실 습 4-2 : 보스턴 주택가격 예측(Option, 앙상블 모델)

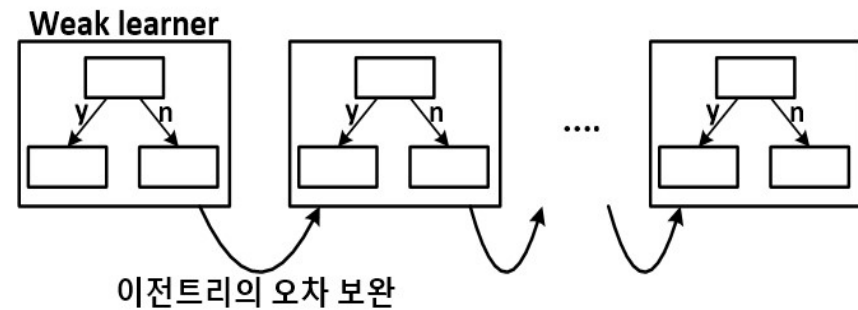
✓ 결정트리(Decision Tree)



✓ 앙상블 모델(Ensemble model)



〈랜덤 포레스트(Random forest)〉



〈그래디언트 부스팅(Gradient boosting) 트리〉

4. 머신러닝 실습 2 : 회귀

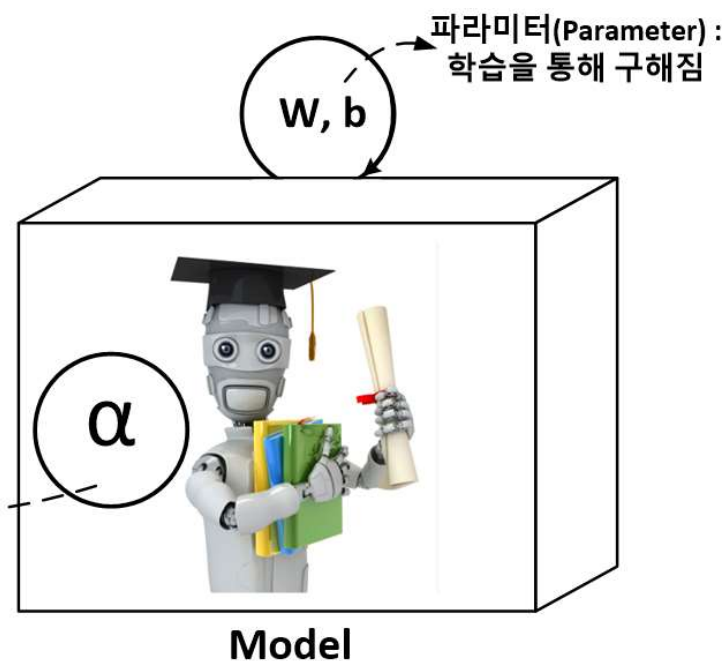
하이퍼 파라미터

✓ 모델성능 개선 방법 2

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,  
    max_features='auto', max_leaf_nodes=None,  
    min_impurity_decrease=0.0, min_impurity_split=None,  
    min_samples_leaf=1, min_samples_split=2,  
    min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,  
    oob_score=False, random_state=7, verbose=0, warm_start=False)
```

```
GradientBoostingRegressor(alpha=0.9, criterion='friedman_mse', init=None,  
    learning_rate=0.1, loss='ls', max_depth=3, max_features=None,  
    max_leaf_nodes=None, min_impurity_decrease=0.0,  
    min_impurity_split=None, min_samples_leaf=1,  
    min_samples_split=2, min_weight_fraction_leaf=0.0,  
    n_estimators=100, n_iter_no_change=None, presort='auto',  
    random_state=7, subsample=1.0, tol=0.0001,  
    validation_fraction=0.1, verbose=0, warm_start=False)
```

하이퍼 파라미터
(Hyper Parameter):
학습의 대상이 아니라
학습의 환경임



4. 머신러닝 실습 2 : 회귀

실 습 4-2 : 보스턴 주택가격 예측(*Option*, 앙상블 모델)

✓ 최적의 하이퍼 파라미터 찾기

- 최적의 max_depth 찾기

```
GradientBoostingRegressor(alpha=0.9, criterion='friedman_mse', init=None,  
    learning_rate=0.1, loss='ls', max_depth=3, max_features=None,  
    max_leaf_nodes=None, min_impurity_decrease=0.0,  
    min_impurity_split=None, min_samples_leaf=1,  
    min_samples_split=2, min_weight_fraction_leaf=0.0,  
    n_estimators=100, n_iter_no_change=None, presort='auto',  
    random_state=7, subsample=1.0, tol=0.0001,  
    validation_fraction=0.1, verbose=0, warm_start=False)
```

- 파이썬 머신러닝 모델 개발환경 이해 :

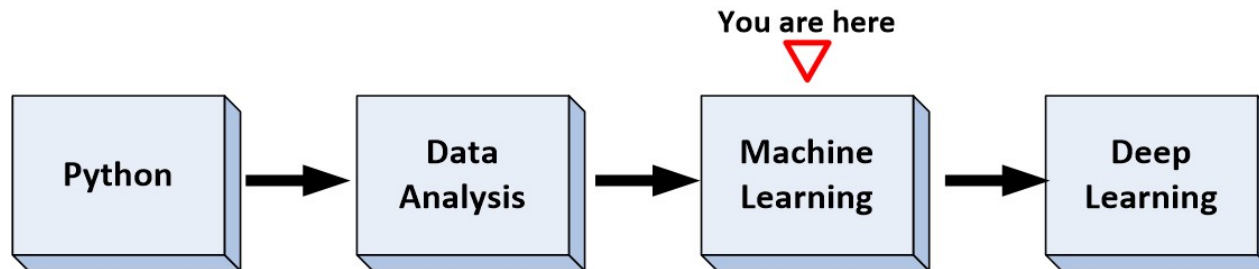
- Anaconda, Jupyter notebook, Numpy, pandas, Scikit-learn

- 파이썬 기본문법 소개

- 자료형, 조건절, 반복문, 함수 등
 - MCP 뉴런, 퍼셉트론 구현 실습

- 머신러닝 모델개발 실습

- 분 류 : IRIS 품종분류, 와인등급 분류, 데이터셋 구축방법(feature, label, train/ test data), 로지스틱 회귀
 - 회 귀 : 보스턴 집값 예측, 성능지표(R^2 , Accuracy), 과대적합/ 과소적합, 하이퍼 파라미터 선형회귀, 랜덤 포레스트, 그래디언트 부스팅



감사합니다.

Kwang Myung Yu

www.github.com/sguys99 sguys99@naver.com