



# 7 머신러닝 모델부터 예측까지 기초 실습 1

# CONTENTS

01

선형 회귀 (skip)  
(Linear Regression)

02

K-평균 (K-means)



# 01 선형 회귀 - 실습 (Linear Regression)

## 같이하기

**scikit-learn.org**

[https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LinearRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html)

## Step 1. 필요한 라이브러리 import

- 넘파이
- `sklearn.linear_model -> LinearRegression`

## 같이하기

## scikit-learn.org

## Step 2. 데이터 준비하기

- $y = (1 * x_1) + (2 * x_2) + 3$  <- 이런 공식을 만들려고 하네요.
- $X = 2$ 차원  $[1, 1], [1, 2], [2, 2], [2, 3]$
- $y =$  공식을 사용한 값
- $X$ 를 출력
- $y$ 를 출력



```
y = np.dot(X, np.array([1, 2])) + 3
```

## 같이하기

**scikit-learn.org**

Step 3. 모델만들기와 학습하기 - **reg**란 이름으로 **LinearRegression** 모델을 만든후 학습

- 학습: `fit(X, y)`
- `reg(모델)`를 출력해서 살펴볼까요?

## 같이하기

## scikit-learn.org

## Step 4. 기울기와 절편 알아보기

- 기울기: coefficient
- 절편: intercept

```
reg.coef_
```

```
reg.intercept_
```



## 같이하기

### scikit-learn.org

Step 5. 예측하기 - predict를 사용해서 예측

- 예측: predict
- tip: 넘파이 형식의 값을 사용해야 합니다



## 실습: weight-height 데이터를 사용한 선형 회귀

실습하기전 필요한 데이터및 Tool을 준비하세요.

- 실습 중 종종 Save하시기 바랍니다.

## 실습

## weight-height 데이터를 사용한 선형 회귀

## 데이터 수집

- Kaggle weight-height data
- 캐글에서 살펴보기 - <https://www.kaggle.com/mustafaali96/weight-height>
- 깃헙(github)으로 다운받고 업로드 후 준비된 상태
  - <https://github.com/yyoo79/ktech2020winter/blob/main/weight-height.csv>

## 실습

**weight-height 선형 회귀**

라이브러리 준비하기 (import하기)

- (from) sklearn.linear\_model - (import) linearRegression
- pandas (alias pd)
- numpy (alias np)
- matplotlib.pyplot (alias plt)

## 실습

## weight-height 선형 회귀

1. 판다스의 데이터 프레임으로 파일 불러오기
  - 변수이름 = df
  - 판다스 함수 = read\_csv
  - 파일경로 = <https://raw.githubusercontent.com/yyoo79/ktech2020winter/main/weight-height.csv>
2. dataframe의 처음 5개 data 출력. 잘 불러왔는지 확인
3. 구성(shape)도 알아보세요.
4. 총 몇개인지 알아보세요. hint: 몇개인지 세어볼까요?

## 실습

## weight-height 선형 회귀

matplotlib으로 시각화

1. X축 = Height 지정
2. y축 = Weight 지정
3. matplotlib의 plot함수를 사용 시각화
  - X, y, 'o'의 매개변수 사용
  - show() 사용

## 실습

## weight-height 선형 회귀

모델 생성 &amp; 데이터 fit

- 모델이름 = line\_fitter 모델은 LinearRegression
- **Warning:** X는 그대로 사용할 수 없네요. 그 이유는 X는 2차원 array 형태여야 하기 때문입니다.
- .reshape 함수를 사용해서 변경이 필요

## 실습

## weight-height 선형 회귀

## 예측하기

- 키가 70인 사람으로 예측
- 다른 값으로 예측



## 실습

## weight-height 선형 회귀

기울기와 절편

- 기울기 - coefficient 구하기 / 출력하기
- 절편 - intercept 구하기 / 출력하기

## 실습

**weight-height 선형 회귀**

선그리기- 기존  $x$  값으로  $y$ 를 예측하게 해서 그래프 그리기

1. 우선, 점선 'o'를 그리세요
2. 예측해서 나온 값을  $y$  대신 넣어보세요
3. 그래프 보여주기 - `show()`

## 실습

## diabetes dataset 선형 회귀

[https://scikit-learn.org/stable/auto\\_examples/linear\\_model/plot\\_ols.html#sphx-glr-auto-examples-linear-model-plot-ols-py](https://scikit-learn.org/stable/auto_examples/linear_model/plot_ols.html#sphx-glr-auto-examples-linear-model-plot-ols-py)



## 02 K-평균 (K-means)

## 같이하기

### K-means

#### Step 1. 필요한 라이브러리 import

- 넘파이
- matplotlib
- sklearn.cluster (import) KMeans

## 같이하기

### K-means

#### Step 2. 데이터 만들기

- `x = [1, 5, 1.5, 8, 1, 9]`
- `y = [2, 8, 1.8, 8, 0.6, 11]`
- matplotlib에 `scatter`를 사용해서 x 와 y값을 넣습니다.
- `plt`을 사용해서 그래프를 보여줍니다.

## 같이하기

### K-means

#### Step 2a. 데이터 만들기 2

- X, 즉 훈련데이터는 2차원 (x와 y를 묶어놓은것) 형식이 필요
- 넘파이 np를 사용해서 2차원 array를 만드세요
  - ex: `[1, 2], [5, 8]` ...
- X를 출력



## 같이하기

### K-means

#### Step 3. 모델 만들기

- 변수이름 - kmeans
- 모델 - KMeans()
  - <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>
  - 매개변수: **n\_clusters : int, default=8**
  - n\_clusters = 2 개로 지정
- centroids 출력
- labels 출력

## 같이하기

### K-means

Step 4. Coordinate 표시하고, 그래프 만들기/출력(show)

- colors 변수에 green, red, cyan, yellow 색깔에 해당하는 코드 (더하기 점(.))로 리스트 만들기
- for loop을 X 개수 만큼 반복/돌리기
  - X[i] 와 레이블[i]를 출력 - 형식:

```
coordinate: [1. 2.] label: 1
coordinate: [5. 8.] label: 0
```
  - plt.plot - 매개변수
    - x좌표 값
    - y좌표 값
    - color 값 - labels[i]에 해당되는 값을 colors list의 index로 사용
    - markersize = 10

## 같이하기

### K-means

Step 4a. Coordinate 표시하고, 그래프 만들기/출력(show) 2

- scatter graph만들기 - centroid를 그래프에 표시
  - parameter:
    - 값1 - centroid에 첫번째 값 - `centroids[:,0]`
    - 값2 - centroid에 두번째 값 - `centroids[:,1]`
    - 마커 - "x"
    - s = 150
    - 라인두께 = 5
    - zorder = 10
- 만들어진 scatter show하기

## 실습: make\_blobs를 사용한 K-means

실습하기전 필요한 데이터 및 Tool을 준비하세요.

- 실습 중 종종 Save하시기 바랍니다.

## 실습

## make\_blobs를 사용한 K-means

sklearn.datasets .make\_blobs()란?

- [https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make\\_blobs.html](https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_blobs.html)
- 이 함수를 사용하여 가우시안 분포의 점 블록을 생성
- 생성할 블록 수와 샘플 수는 물론 다른 속성들도 조절가능
- 선형 분류 문제에 적합
- 예) 멀티 클래스 분류 예측 문제 - 3개의 블록이 있는 표본의 2차원 데이터 셋을 생성. 각 관측치에는 두 개의 입력값과 0,1 또는 2개의 클래스 값이 존재
  - `X, y = make_blobs(n_samples=100, centers=3, n_features=2)`
- real world 예1 - 마라톤 선수의 자질이 있는 group으로 나눌때, 정해진 시간에 속도와 거리.
- real world 예2 - 몸무게와 키를 가지고 group으로 나눌때

## 실습

**make\_blobs를 사용한 K-means**

## Step 1. 필요한 라이브러리 import

- (from) sklearn (import) datasets
- (from) sklearn.cluster (import) KMeans
- matplotlib (alias plt)
- 넘파이 (alias np)

## 실습

## make\_blobs를 사용한 K-means

## Step 2. 데이터 준비/만들기

- datasets의 make\_blobs 함수를 사용해서 X, y 변수 생성
  - 일반적으로 `X, y = make_blobs()` 로 구현됨
    - X는 graph 좌표 값, y = label (cluster 이름)
- X의 shape과 type을 출력
- X를 출력
- y도 역시 shape, type, y 출력

## Examples

```
>>> from sklearn.datasets import make_blobs
>>> X, y = make_blobs(n_samples=10, centers=3, n_features=2,
...                   random_state=0)
```



## 실습

## make\_blobs를 사용한 K-means

## Step 2. 모델 만들기 / 훈련하기

- KMeans 모델만들기
  - 변수이름 = kmeans
  - cluster는 총 3개
- 만들어진 모델을 훈련(fit)
  - X 데이터 사용
- 훈련후, 모델의 레이블을 출력후 확인

Note: predict() 가 필요없음

## 실습

## make\_blobs를 사용한 K-means

## Step 3. 시각화하기

- 맷플롯립 plt로 scatter graph 만들기 (X값)
  - 값1 - X에 첫번째 값 (tip: 콜론(:), 0 index)
  - 값2 - X에 두번째 값
  - 색깔(color) = label array 사용
  - 마커 = 'o'
  - s = 10

## 실습

## make\_blobs를 사용한 K-means

## Step 3a. 시각화하기 2

- 맷플롯립 plt로 scatter graph 만들기 (centroids)
  - cluster\_centers\_
    - kmeans에서 얻을수 있음
    - kmeans.label\_과 같은 형식
  - 값1 - cluster\_centers\_ 의 첫번째 값 (tip: 콜론(:), 0 index)
  - 값2 - cluster\_centers\_ 의 두번째 값
  - 색깔(color) = red, black(k), blue
  - 마커 = 삼각형
  - 마커 포인트(s) = 50
- show

## 실습

## make\_blobs를 사용한 K-means

## Step 4. 예측하기

- predict로 예측
  - parameter 값은 넘파이 2차 형태여야 함.
    - 예: `np.array([[0,0]])`

## 실습

## make\_blobs를 사용한 K-means

이제 K을 값을 5로 바꿔서 하면 어떨까요?

Step 5. 모델 다시 만들기 / 훈련하기

- Step 2와 같은 방법으로
  - 모델이름 = kmeans5, n\_clusters = 5로 지정
- X data 훈련

## 실습

## make\_blobs를 사용한 K-means

## Step 6. 시각화 다시 하기

- Step 3와 동일
- 다른점:
  - `c = 모델 kmeans5의 labels_`를 사용
- Step 3a와 동일
- 다른점:
  - `c = 색깔(color)`를 2개 추가 - gray, orange
- `show`