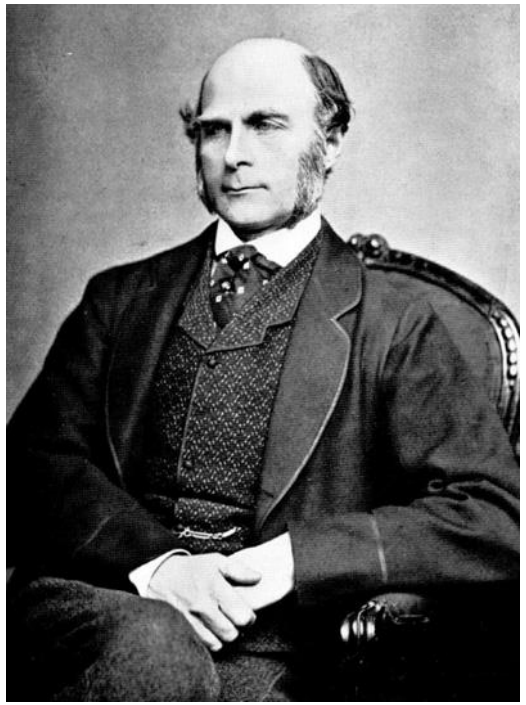
데이터 기반한 AI분석

- 예제로 알아보는 회귀분석
- 회귀분석 알고리즘 이해와 구현
- 머신러닝 라이브러리 사이킷런 소개

회귀 분석의 역사

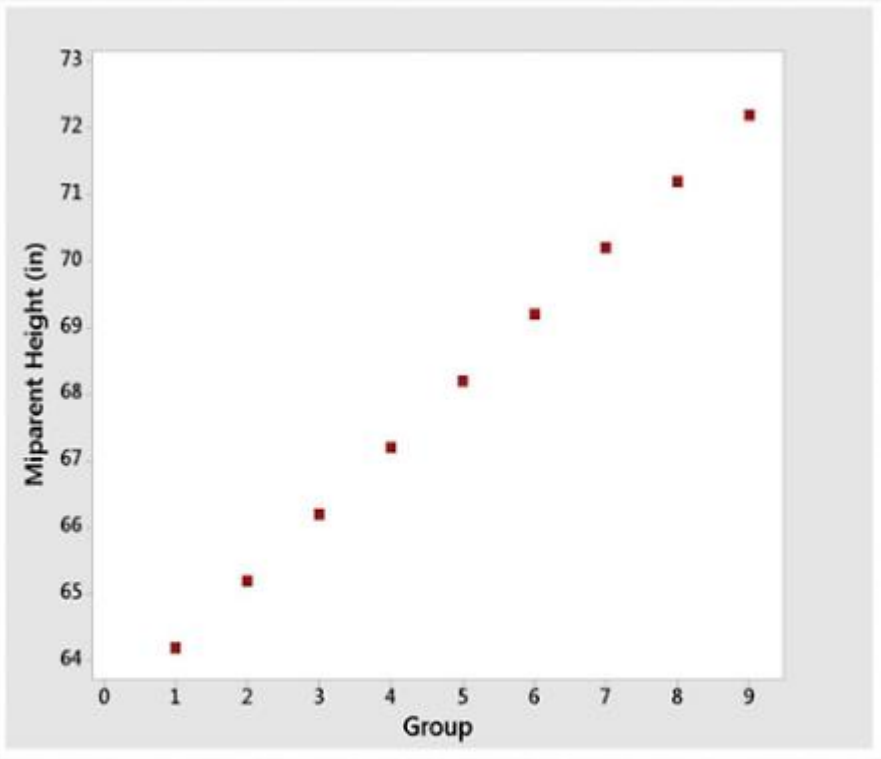
프랜시스 골턴

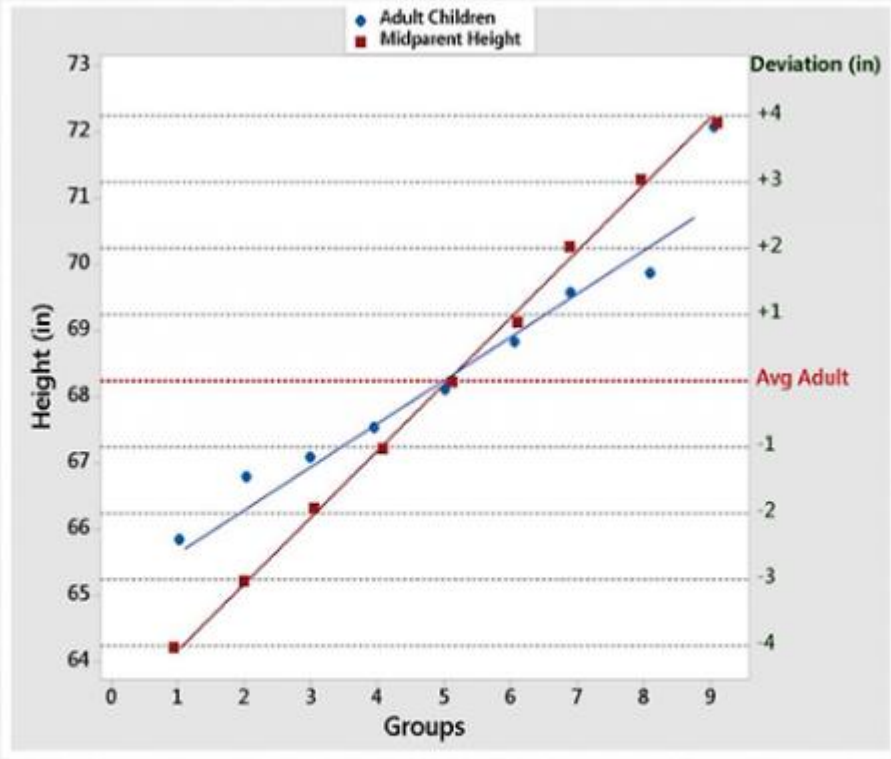


Regression toward the mean

- 회귀regression
  - 두 변수의 관계가 일반화된 선형 관계의 평균으로 돌아간다
  - 통계분야에서 프랜시스 골턴이 처음 사용한 용어
  - 19세기 말 영국. 사회와 경제의 변화 예측 설명하는 방법을 필요로함.
  - 인류 유전의 원리와 사회적 성격에 대한 연구의 일환으로 키와 유전적 특성을 연구
  - 205쌍의 부모와 자식의 키 측정.
- 회귀 분석은 대표적인 지도학습 알고리즘
  - 관측된 데이터를 통해 독립변수와 종속변수 사이의 숨어있는 관계를 추정하는 것

회귀분석의 역사





- 그 다음으로 각 그룹에 해당하는 자식의 실제 키(ADULT CHILDREN)를 표시하였습니다.
- 성인 평균 키를 기준으로 하여 둘 간의 관계를 살펴 보았습니다. 흥미로운 점은 그래프와 같이 성인 평균 키 보다 큰 부모 그룹의 자식은 오히려 부모보다 키가 작으며, 반대로 키가 작은 부모 그룹의 자식의 키는 크다는 것을 발견하였습니다

상관분석

◆ Correlation analysis

◆ **두 연속형 변수 사이 상관관계가** 존재하는지를 파악하고, 상관관계의 정도를 확인 하는 것

◆ 상관계수(Correlation coefficient)로 선형적 상관도를 확인하여 정도 파악

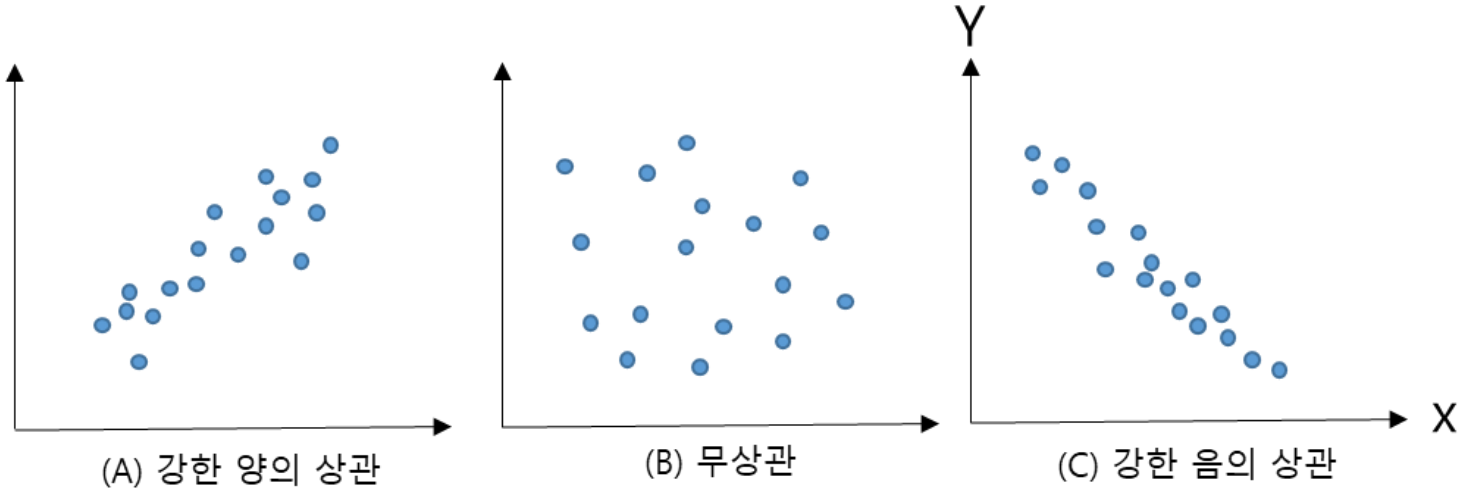
student_no	class	science	english	math	sex
1	A	92	98	97	m
2	A	62	66	65	w
3	A	81	86	84	w
4	A	73	72	71	m
5	B	65	66	69	w

① **산점도(Scatter) 두 변수 상관 파악**

② **상관계수 확인**

③ **의사결정**

상관분석

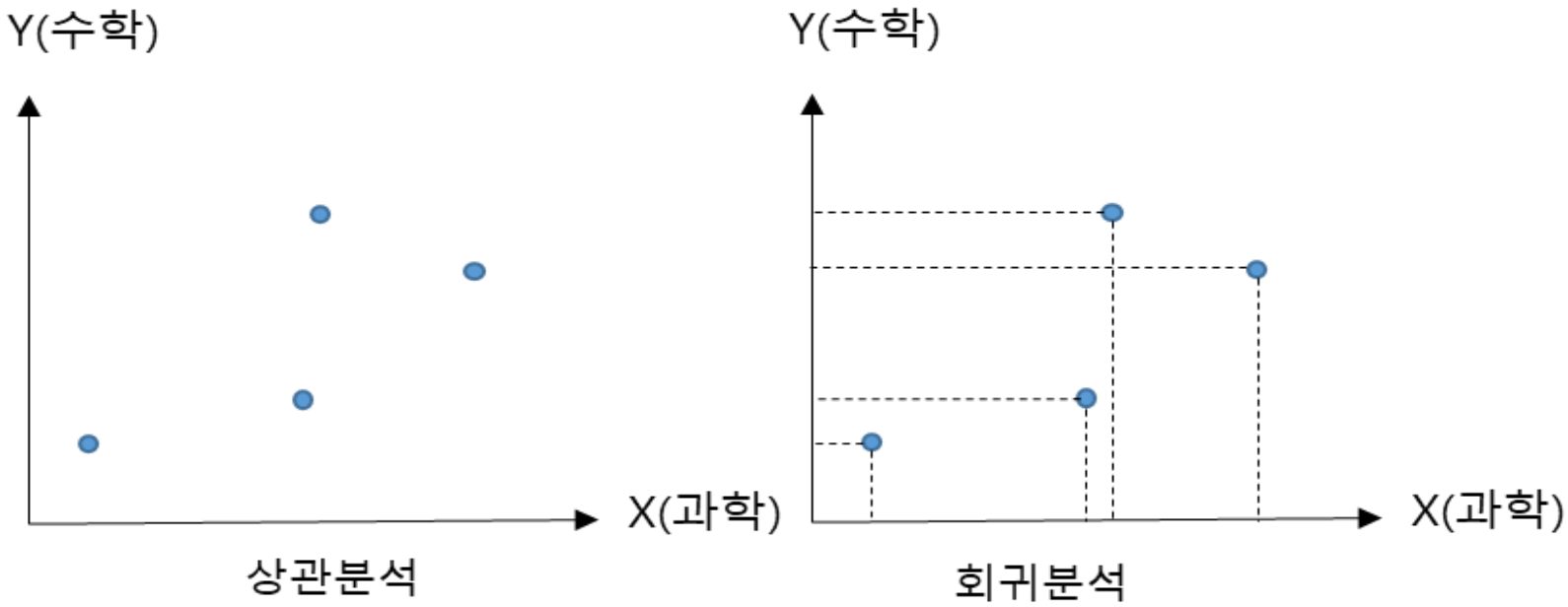


상관	상관계수
음의 상관	-1.0 ~ -0.7 이면, 강한 음의 상관관계 – 그림C -0.7 ~ -0.3 이면, 뚜렷한 음의 상관관계 -0.3 ~ -0.1 이면, 약한 음의 상관관계
무상관	-0.1 ~ +0.1 이면, 없다고 할 수 있는 상관관계 - 그림 B
양의 상관	+0.1 ~ +0.3 이면, 약한 양의 상관관계 +0.3 ~ +0.7 이면, 뚜렷한 양의 상관관계 +0.7 ~ +1.0 이면, 강한 양의 상관관계 – 그림A

회귀분석

- Regression Analysis
- 상관분석은 두 연속형 변수 X(과학)와 Y(수학)의 상관 정도만 파악 가능하고 인과관계는 알 수 없음.
- 회귀분석은 두 연속형 변수 X와 Y를 독립변수와 종속변수라고 하는 인과관계를 파악할 수 있음
- 회귀분석은 데이터에 기반하여 새로운 입력 데이터에 대한 추론(예측)이 가능한 알고리즘이다.
  - '과학 점수가 좋으면 수학점수가 좋을 확률은?'

회귀분석



X	Y
독립변수,설명변수,원인변수	종속변수,반응변수,결과변수
다른 변수에 영향을 주는 원인	다른 변수에 영향을 받는 결과

회귀분석과정

1. 변수사이의 상관관계 분석(시각화 포함)
2. 상관관계가 있는 변수 정의(종속=독립)
3. 회귀 모형 선택(회귀방정식)
4. 모델 평가

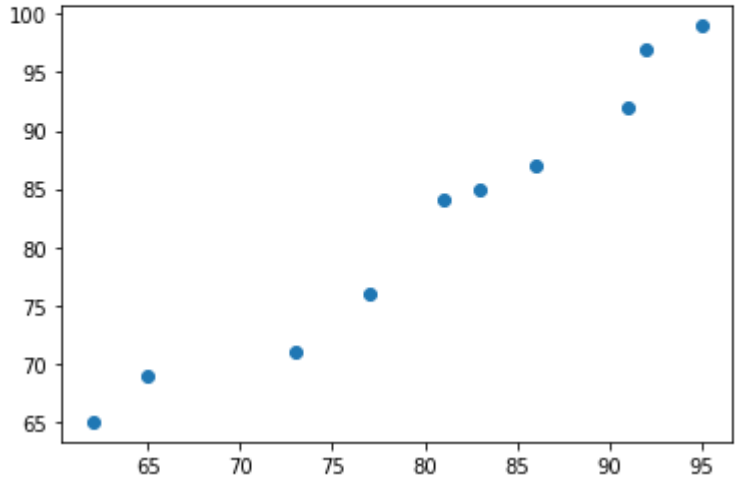
회귀분석실습

```python
#판다스 라이브러리 불러오기
import pandas as pd
import matplotlib as mpl
import matplotlib.pylab as plt
import numpy as np
import seaborn as sns

#데이터불러오기
file_path= '/content/drive/MyDrive/Colab Notebooks/ml_2024/data/exam_sample_cor.csv'

#read csv()함수로 데이터프레임 변환
df = pd.read_csv(file_path)
df
```

	student_no	class	science	english	math	sex
0	1	A	92	98	97	m
1	2	A	62	66	65	w
2	3	A	81	86	84	w
3	4	A	73	72	71	m
4	5	B	65	66	69	w
5	6	B	86	89	87	m
6	7	B	91	90	92	m
7	8	B	77	78	76	w
8	9	C	95	98	99	w
9	10	C	83	82	85	w

```python
# 두 연속형 변수 스캐터플롯 그리기
plt.scatter(df.science, df.math)
```

```
<matplotlib.collections.PathCollection at 0x1e1400f29c8>
```



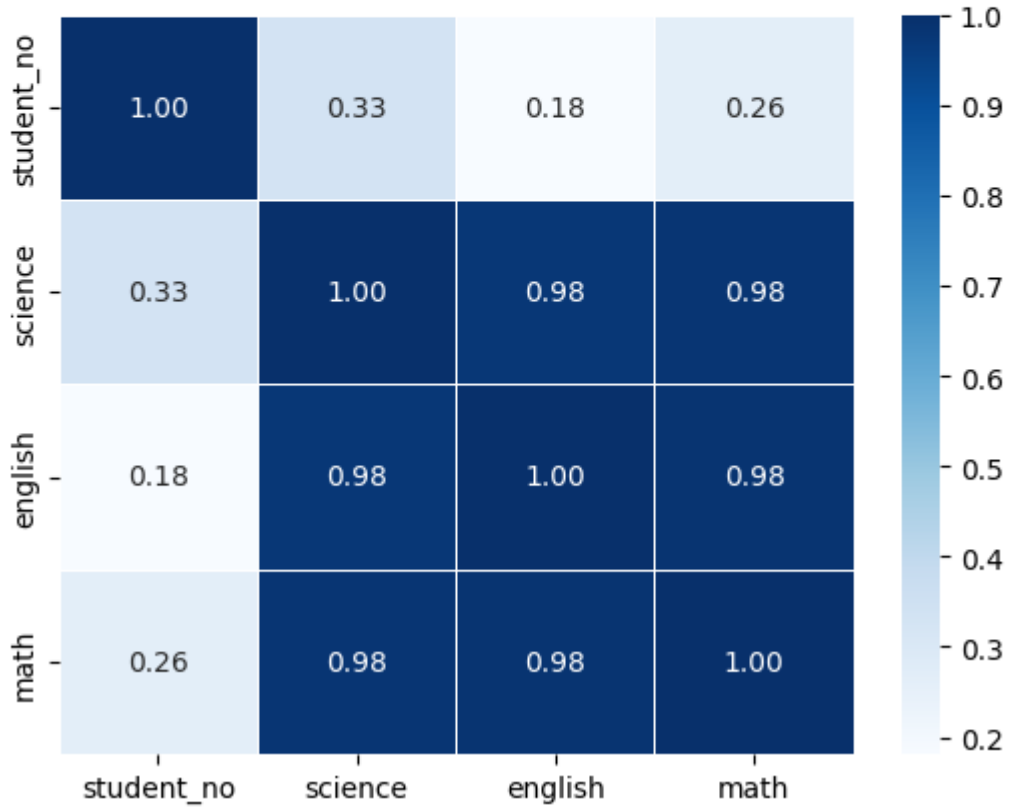
[과학과 수학 점수의 상관 관계 시각화 예]

회귀분석실습

```python
# 상관계수 확인
numerical_df = df.select_dtypes(include=['number'])
corr = numerical_df.corr() #(method = 'pearson')
```

```python
# 히트맵으로 상관도 시각화
sns.heatmap(data = corr , annot=True, fmt = '.2f', linewidths=.5, cmap='Blues')
```

```
<Axes: >
```

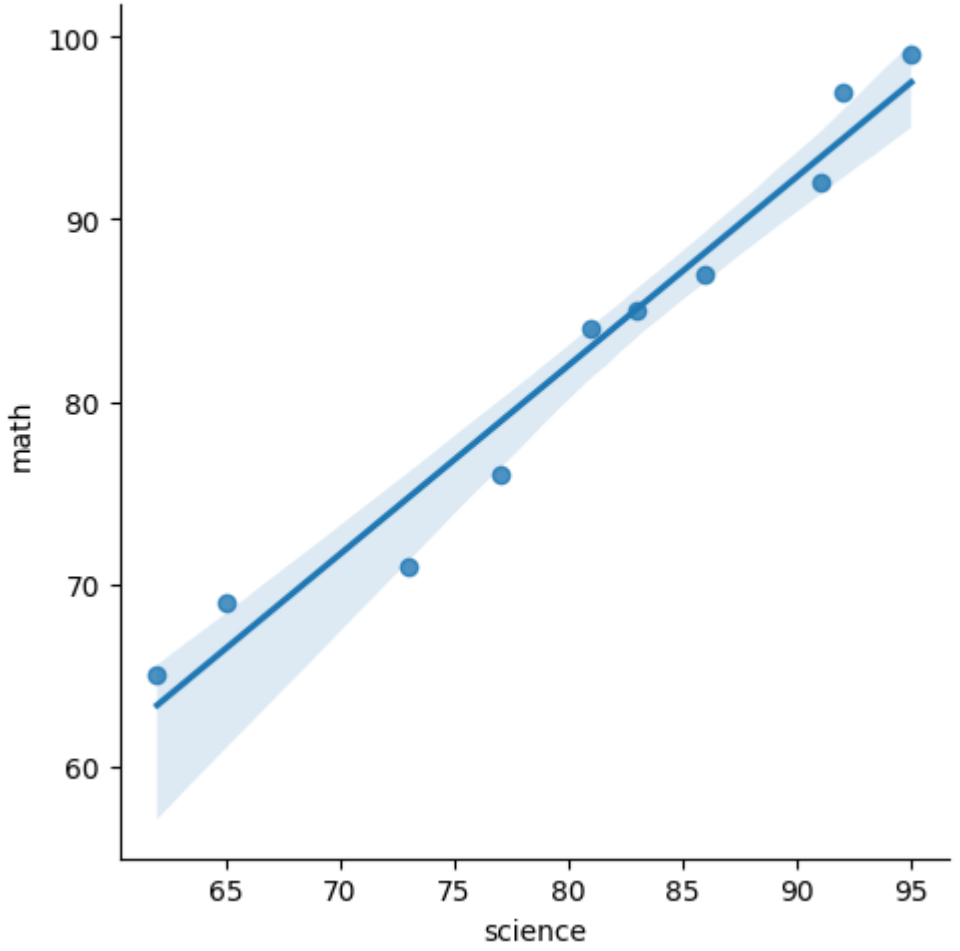


회귀분석실습

- 과학, 수학 점수 상관 분석 그래프

```
sns.lmplot(x='science', y='math',data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7bafd6b43940>
```




```
# 회귀분석을 위해 종속(Y=수학), 독립(X=과학)
# 과학점수를 알면 수학점수를 예상 할 수 있다.
# 단순선형회귀 모형
import statsmodels.api as sm
lin_reg = sm.OLS.from_formula("math ~ science", df).fit()
lin_reg.summary()
```

OLS Regression Results

Dep. Variable:	math	R-squared:	0.965
Model:	OLS	Adj. R-squared:	0.960
Method:	Least Squares	F-statistic:	218.5
Date:	Sat, 29 Jun 2024	Prob (F-statistic):	4.32e-07
Time:	07:10:20	Log-Likelihood:	-21.673
No. Observations:	10	AIC:	47.35
Df Residuals:	8	BIC:	47.95
Df Model:	1		
Covariance Type:	nonrobust		

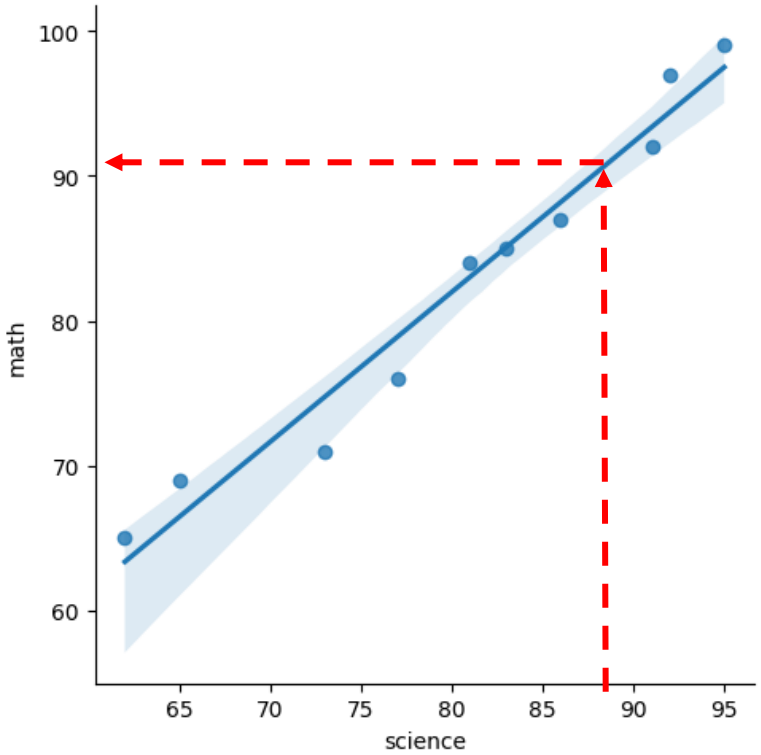
	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-0.7527	5.682	-0.132	0.898	-13.855	12.349
science	1.0342	0.070	14.781	0.000	0.873	1.196

Omnibus:	1.170	Durbin-Watson:	2.261
Prob(Omnibus):	0.557	Jarque-Bera (JB):	0.811
Skew:	-0.402	Prob(JB):	0.667
Kurtosis:	1.859	Cond. No.	618.

회귀분석실습

◆ 수학과 과학 성적 간의 관계를 **직선의 방정식으로 표현한다는 가설** 하에, 주어진 **데이터로부터 가중치 a(weight)와 편향 b(bias)**를 찾아 데이터를 가장 잘 표현하는 직선을 찾음.

수학예측값 = 1.0342 * 과학 – 0.7527



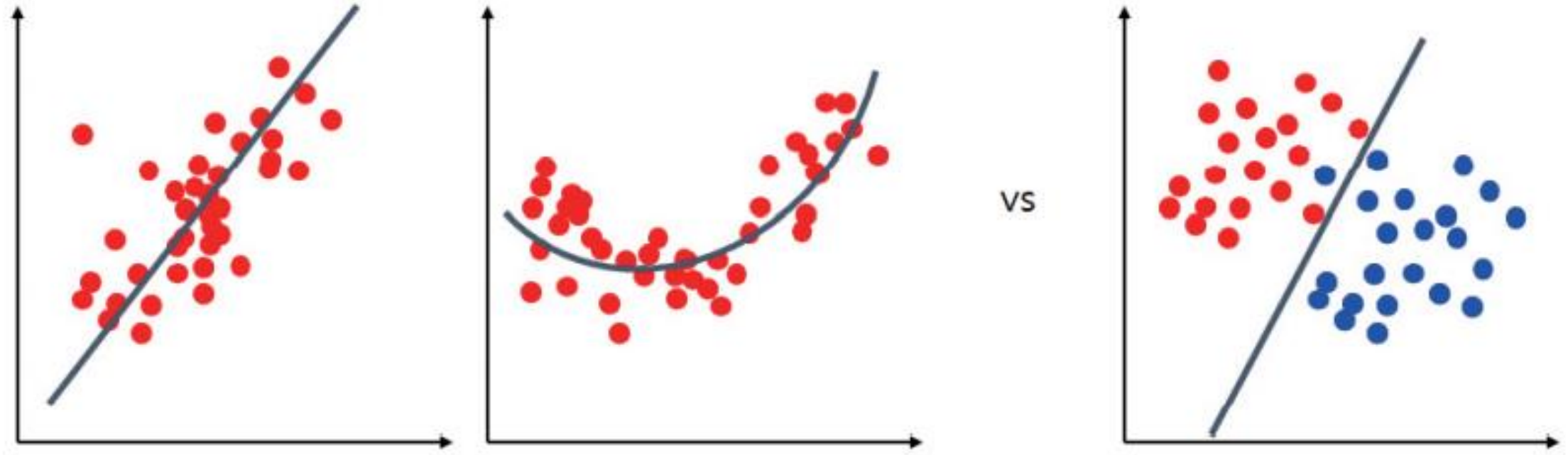
✓ 과학점수가 88점 받을 때, 수학점수는?

```
lin_reg.predict()
```

회귀분석 알고리즘 이해와 구현

회귀분석 종류

- 회귀Regression 입력 데이터 하나 하나에 대응하는 출력값을 예측
- 분류classification 는 입력 데이터를 몇 가지 이산적인 범주category 중의 하나로 대응



회귀(Regression)

분류(Classification)

회귀분석 종류

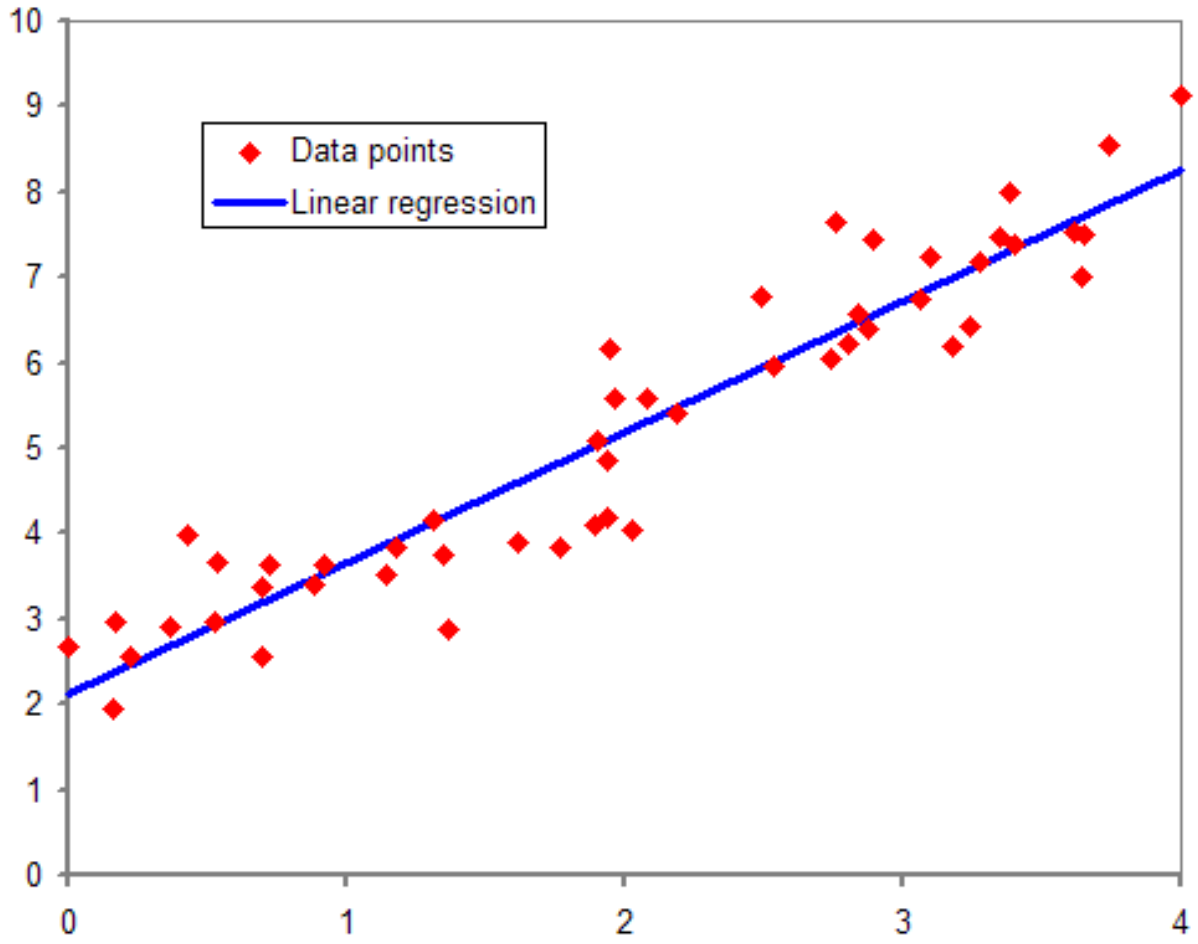
- 선형 회귀 Linear Regression: 수치 예측
  - 연속변수에 대한 분석 모형
  - 단순 선형 회귀: 독립변수 1개
  - 다중 선형 회귀: 독립변수 2개 이상
  - 다항 회귀(Polynomial Regression)
    - 독립변수를 다항식으로 변환하여 선형 회귀 수행

- 로지스틱 회귀(Logistic Regression): 범주 예측
  - 범주형 변수에 대한 분석 모형
  - 이항 로지스틱 회귀: 이항 분류
  - 다항 로지스틱 회귀: 다항 분류

선형회귀 알고리즘

◆ Linear Regression Algorithm

좌표평면 위에 표현된 **다수의 데이터들**을 바탕으로 **입력 변수와 출력 변수 간의 관계를 가장 잘 대표할 수 있는 선형 (선형 결합) 상관 관계를 모델 하나로 모델링하는 알고리즘**



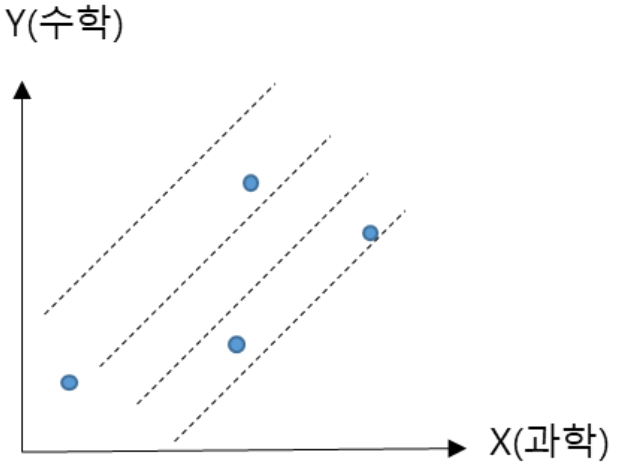
선형회귀 모델을 구하기(모델링) 위한 공식

- 주어진 데이터를 대표하는 회귀선(하나의 직선)을 찾는 것
  - 선형 회귀선의 함수식 – 선형회귀식
  - Y = aX + b

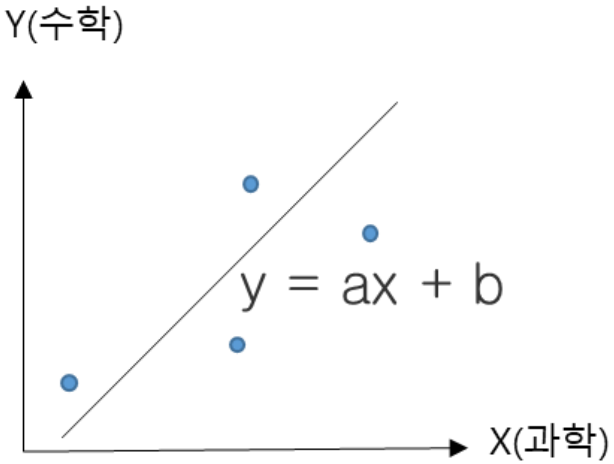
- 선형회귀 과정은,

입력변수(x)와 출력 변수(y) 사이의 관계를 선으로 모델링(표현) 하는 함수의 절편(b)과 기울기(a)를 찾아가는 과정이다!!

- 선형회귀를 통해 얻게 되는 함수 f를 선형회귀 모델이라고 한다.



(a)



(b)

선형회귀 모델을 구하기(모델링) 위한 공식

◆ 선형회귀 직선의 방정식 변수 x와 y의 예시

"학생들의 중간고사 성적이 [ ]에 따라 다르다."

[ ] 부분에 시험 성적을 좌우할 만한, 공부한 시간, 시험 당일의 컨디션, 사교육비 지출액 등의 요소들이 들어갈 수 있음

성적을 변하게 할 수 있는 요소는 입력 변수 x, 그 값에 따라 종속적으로 변하는 성적은 출력 변수 y임

입력 변수 x를 독립변수, 출력 변수 y를 종속변수란 용어 사용

선형회귀 모델을 구하기(모델링) 위한 공식

‘선형회귀는 입력 변수와 출력 변수 간의 관계를 가장 잘 대표하는 선형 상관 관계를 모델링’

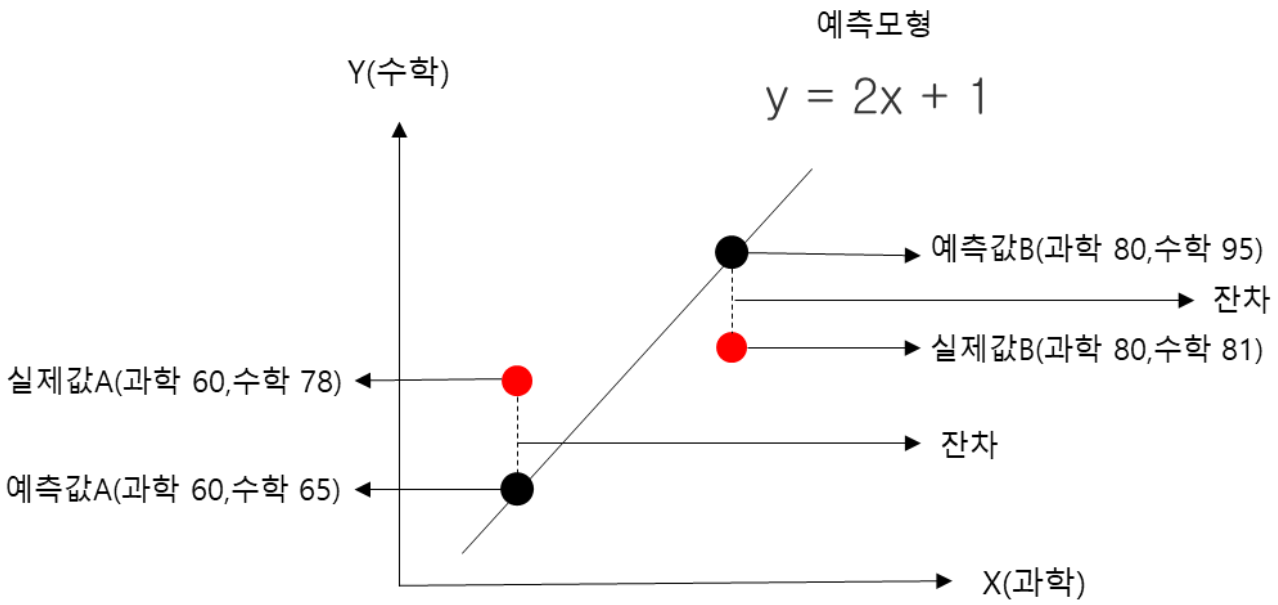
=> 여기서 ‘가장 잘 대표하는’의 기준은?

$y = wx + b$의 경우, 좋은 선형회귀 모델은,

입력 변수 x에 따른 출력 변수 y의 예측값 f(x)와 실제 출력값 y의 차이인, **오차**가 최소가 되도록 하는 w와 b값의 모델임

예측값 f(x)와 실제 출력값 y의 차이인 **오차(E)**가 가장 작은 모델?!

선형회귀 모델을 구하기(모델링) 위한 공식



예측값 f(x)와 실제 출력값 y의 차이,

오차(E)를 최소화하는 방법

- **오차의 제곱합을 구하는 방법**

오차는 양수일 수도 있고 음수일 수도 있으며, 오차의 크기가 얼마나 큰지 가늠하기 위해서는 부호를 제거해야 하므로 오차 값을 제곱해서 사용함

선형회귀 모델을 구하기(모델링) 위한 공식

최소제곱법(LSM) 으로 선형회귀모델 구하기

- 잔차 제곱의 합이 최소로 하는 직선을 회귀선으로 결정

$$E = \sum_{i=1}^{n} [y_i - ax_i - b]^2$$

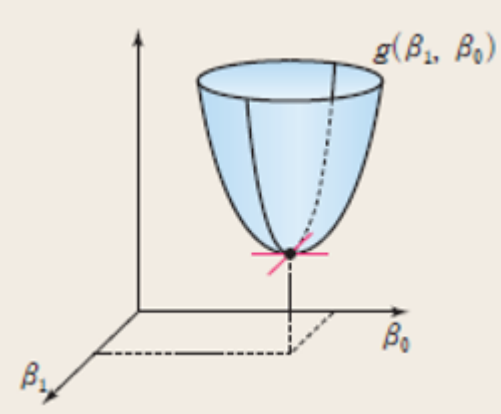
✓ **즉, 최소제곱법은 선형모델인 직선 y=wx + b의 기울기 w와 y절편 b를 추정한다.**

최소제곱법(LSM)

- $y=wx+b$에서 w와 b를 구하는 방법!

$\sum_i^n(y_i-\hat{y_i})^2=\sum_i^n(y_i-(wx_i+b))^2$가 최소가 되는 w와 b를 구하기 위하여

위 식을 변수가 2 개인 함수 $g(w,b)$라고 하고,



함수의 그래프가 왼쪽 그림과 같다면,

각 변수의 편미분 값이 0이 되어야 최소가 됨

$$\frac{\partial g}{\partial w}=\frac{\partial g}{\partial b}=0$$

최소제곱법(LSM)

$$\begin{cases} \dfrac{\partial g}{\partial w} = \sum 2(y_i - wx_i + b)(-x_i) = 2\left(w\sum {x_i}^2 + b\sum x_i - \sum x_i y_i\right) = 0 & \cdots ① \\ \dfrac{\partial g}{\partial b} = \sum 2\,(y_i - wx_i + b)(-1) = 2\left(w\sum x_i + b\sum 1 - \sum y_i\right) = 0 & \cdots ② \end{cases}$$

변수 w와 b를 기준으로 각각 편미분한 후, 편미분 값을 0으로 두고 식을 전개 정리하면 ①과 ②의 w와 b에 대한 연립방정식을 얻을 수 있다.

연립방정식은 두 가지 방법(**행렬방정식으로 변환하여 푸는 방법**과 연립방정식을 그대로 푸는 방법)으로 풀 수 있음

최소제곱법(LSM)

방법 1/2. 행렬방정식 풀이

연립방정식을 정리한 식, $\begin{cases} w\sum_{i=1}^{n} {x_i}^2 + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \\ w\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} 1 \quad = \sum_{i=1}^{n} y_i \end{cases}$ 에서

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \qquad A = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix}, \qquad X = \begin{pmatrix} w \\ b \end{pmatrix}$$

Y, A, X를 위와 같이 놓으면, 연립방정식을

$$A^T AX = A^T Y$$

인 하나의 행렬방정식으로 간단하게 변환할 수 있다.

최소제곱법(LSM)

※ 좌변 A^TAX

$$A^TA = \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ 1 & 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix} = \begin{pmatrix} {x_1}^2 + {x_2}^2 + \cdots + {x_n}^2 & x_1 + x_2 + \cdots + x_n \\ x_1 + x_2 + \cdots + x_n & 1 + 1 + \cdots + 1 \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{i=1}^{n} {x_i}^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} 1 \end{pmatrix}, \qquad A^TAX = \begin{pmatrix} \sum_{i=1}^{n} {x_i}^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} 1 \end{pmatrix} \begin{pmatrix} w \\ b \end{pmatrix}$$

※ 우변 A^TY

$$A^TY = \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ 1 & 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} x_1y_1 + x_2y_2 + \cdots + x_ny_n \\ y_1 + y_2 + \cdots + y_n \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{n} x_iy_i \\ \sum_{i=1}^{n} y_i \end{pmatrix}$$

최소제곱법(LSM)

※ 전체 식

$$A^TAX = A^TY \quad \Rightarrow \quad \begin{pmatrix} \sum_{i=1}^{n} {x_i}^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} 1 \end{pmatrix} \begin{pmatrix} w \\ b \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i \end{pmatrix}$$

위 행렬방정식은 연립방정식 $\begin{cases} w\sum_{i=1}^{n} {x_i}^2 + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \\ w\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} 1 = \sum_{i=1}^{n} y_i \end{cases}$ 와 같다.

행렬방정식을 정리하면,

$$X = (A^TA)^{-1}A^TY$$

최소제곱법(LSM)

방법 2/2. 연립방정식 풀이

연립방정식의 ②번 식 $2(w\sum x_i + b\sum 1 - \sum y_i) = 0$을 b에 대해 정리하면

$bn = -(w\sum x_i - \sum y_i)$이고 $b = -w\bar{x} + \bar{y}$ (이 수식을 ③으로 함)이므로,

※ 양변을 n으로 나누면, $\sum y_i$ 의 경우 y의 합을 n으로 나눈 것은 y값의 평균이므로 $\bar{y}$ 로 표기함

①번 식 $2(w\sum {x_i}^2 + b\sum x_i - \sum x_i y_i) = 0$에 ③번 식을 대입하여 w에 대해 정리하면

$$w\sum {x_i}^2 + (\bar{y} - w\bar{x})\sum x_i - \sum x_i y_i = 0$$

$$w(\sum x_i(x_i - \bar{x})) = \sum x_i(y_i - \bar{y})$$

$$w = \frac{\sum x_i(y_i - \bar{y})}{\sum x_i(x_i - \bar{x})}$$

최소제곱법(LSM)

여기서 편차의 총합인 $\sum(x_i - \bar{x}) = \sum(y_i - \bar{y}) = 0$이므로

$$w = \frac{\sum x_i(y_i - \bar{y})}{\sum x_i(x_i - \bar{x})} = \frac{\sum x_i(y_i - \bar{y}) - \sum \bar{x}(y_i - \bar{y})}{\sum x_i(x_i - \bar{x}) - \sum \bar{x}(x_i - \bar{x})} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})(x_i - \bar{x})}$$

$$= \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

를 얻을 수 있으며, 이 w를 ③번 식에 대입하면 b의 값도 구할 수 있음

$$w = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \qquad b = \bar{y} - w\bar{x}$$

결과적으로 위의 공식을 얻게 됨

선형회귀알고리즘 구현

단순선형회귀

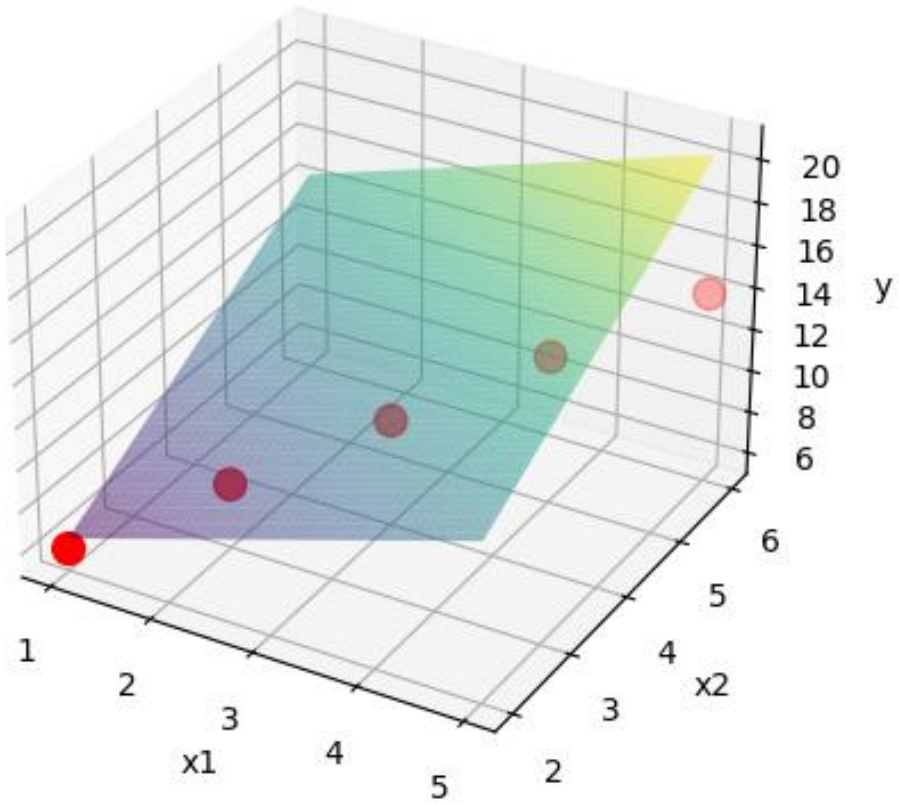
다중선형회귀

다중선형회귀

둘 이상의 입력 변수들과 출력 변수 간의 선형적인 관계를 모델링하는 회귀 분석 기법으로,

여러 개의 독립 변수들과 하나의 종속 변수 간의

선형적인 관계를 설명하는 모델을 만드는 것



다중선형회귀

다중 선형 회귀의 일반식

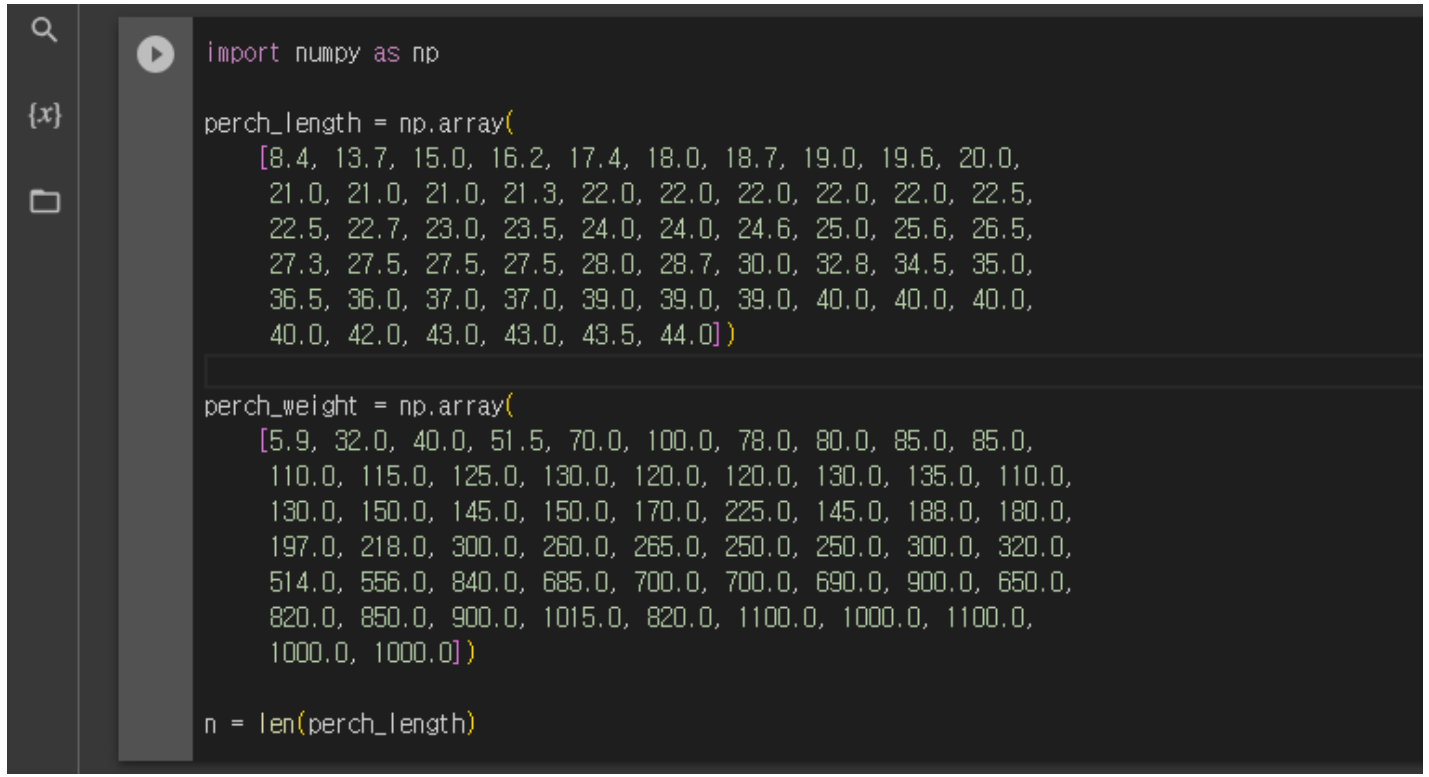
$$y = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$$

다중 선형 회귀의 가정

- 각각의 독립 변수는 종속 변수와의 선형 관계가 존재
- 독립 변수들 사이에는 높은 수준의 상관관계가 존재하지 않음
- 오차가 정규분포를 이룸

단순선형회귀 모델 실습

실습 1. 생선의 길이에 따른 무게를 예측하는 모델을 생성하는 예제



```
import numpy as np

perch_length = np.array(
    [8.4, 13.7, 15.0, 16.2, 17.4, 18.0, 18.7, 19.0, 19.6, 20.0,
     21.0, 21.0, 21.0, 21.3, 22.0, 22.0, 22.0, 22.0, 22.0, 22.5,
     22.5, 22.7, 23.0, 23.5, 24.0, 24.0, 24.6, 25.0, 25.6, 26.5,
     27.3, 27.5, 27.5, 27.5, 28.0, 28.7, 30.0, 32.8, 34.5, 35.0,
     36.5, 36.0, 37.0, 37.0, 39.0, 39.0, 39.0, 40.0, 40.0, 40.0,
     40.0, 42.0, 43.0, 43.0, 43.5, 44.0])

perch_weight = np.array(
    [5.9, 32.0, 40.0, 51.5, 70.0, 100.0, 78.0, 80.0, 85.0, 85.0,
     110.0, 115.0, 125.0, 130.0, 120.0, 120.0, 130.0, 135.0, 110.0,
     130.0, 150.0, 145.0, 150.0, 170.0, 225.0, 145.0, 188.0, 180.0,
     197.0, 218.0, 300.0, 260.0, 265.0, 250.0, 250.0, 300.0, 320.0,
     514.0, 556.0, 840.0, 685.0, 700.0, 700.0, 690.0, 900.0, 650.0,
     820.0, 850.0, 900.0, 1015.0, 820.0, 1100.0, 1000.0, 1100.0,
     1000.0, 1000.0])

n = len(perch_length)
```

먼저, 독립변수 x에 해당하는 길이 배열과

종속변수 y에 해당하는 무게 배열을 선언함

단순선형회귀 모델 실습

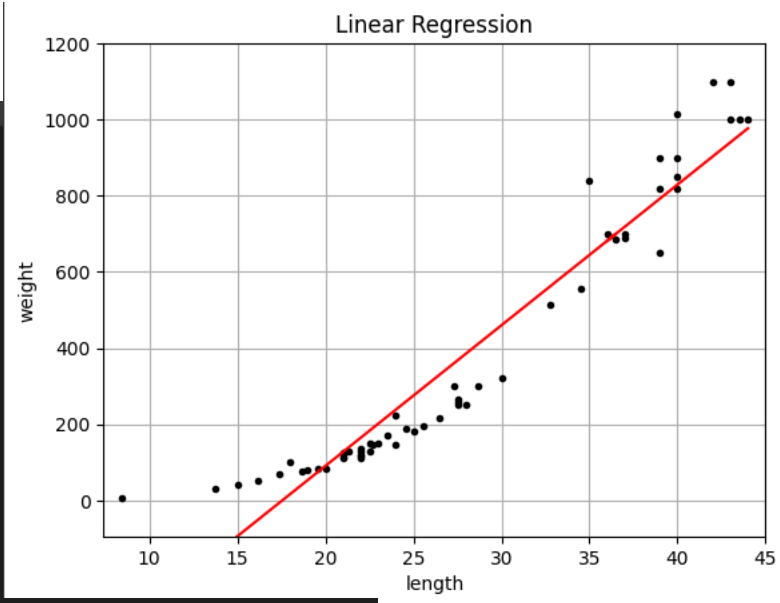
최소제곱법 공식 $w = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}$, $b = \bar{y} - w\bar{x}$ 를 통해 기울기 w와 절편 b의 값을 구함

```
[ ]  mean_x = np.mean(perch_length)
     mean_y = np.mean(perch_weight)

     # 최소제곱법 공식을 이용한 기울기와 절편 구하기

     temp1 = temp2 = 0

     for i in range(n):
         temp1 += (perch_length[i] - mean_x) * (perch_weight[i] - mean_y)
         temp2 += (perch_length[i] - mean_x) ** 2

     w = temp1 / temp2
     b = mean_y - (w * mean_x)

     print(f'기울기(w): {w}, 절편(b): {b}')

     기울기(w): 36.93837596783693, 절편(b): -648.0775582457374
```

단순선형회귀 모델 실습

구한 w와 b값을 이용한 선형 모델로 값을 예측, 그 결과를 화면에 표시함

```
[ ]  import matplotlib.pyplot as plt

    # 구한 기울기와 절편을 이용한 선형회귀 모델의 예측값 계산
    Y_pred = w * perch_length + b

    # 실제 값과 예측 값 출력
    plt.title('Linear Regression')
    plt.xlabel('length')
    plt.ylabel('weight')
    plt.plot(perch_length, perch_weight, 'k.')
    plt.plot(perch_length, Y_pred, color = 'red')
    plt.axis([min(perch_length) - 1, max(perch_length) + 1, min(perch_weight) - 100, max(perch_weight) + 100])
    plt.grid( )
    plt.show( )
```



검은 점들은 실제 데이터, 붉은 직선은 선형 회귀 모델임

다중선형회귀 모델 실습

실습 2. 생선의 길이, 너비, 높이 데이터로 무게를 예측하는 프로그램

이 문제에서 독립변수 x1, x2, x3은 생선의 길이, 너비 높이이고, 종속변수 y는 생선의 무게임

선형회귀 식은 아래와 같이 나타낼 수 있음

$$y = b + w_1 x_1 + w_2 x_2 + w_3 x_3$$

다중선형회귀 모델 실습

사용할 데이터는 실습용 생선 데이터를 코드 상에서 다운로드 받아 사용

Pandas 라이브러리의 read_csv 함수를 통해 생선의 길이, 너비, 높이의 값을 가진 데이터 로드

```
[ ]  import pandas as pd
     import numpy as np

     df = pd.read_csv('https://bit.ly/perch_csv')
     x = df.to_numpy()
     print(x)
```

종속변수 y를 직접 선언

```
[2]  Y = np.array([5.9, 32.0, 40.0, 51.5, 70.0, 100.0, 78.0, 80.0, 85.0, 85.0, 110.0,
           115.0, 125.0, 130.0, 120.0, 120.0, 130.0, 135.0, 110.0, 130.0,
           150.0, 145.0, 150.0, 170.0, 225.0, 145.0, 188.0, 180.0, 197.0,
           218.0, 300.0, 260.0, 265.0, 250.0, 250.0, 300.0, 320.0, 514.0,
           556.0, 840.0, 685.0, 700.0, 700.0, 690.0, 900.0, 650.0, 820.0,
           850.0, 900.0, 1015.0, 820.0, 1100.0, 1000.0, 1100.0, 1000.0, 1000.0])
```

다중선형회귀 모델 실습

최소제곱법의 행렬방정식 풀이를 위해, 입력 데이터에 1로 이루어진 절편 열을 추가하고,

공식 $X = (A^T A)^{-1} A^T Y$를 이용하여 가중치 값의 행렬 [b, w1, w2, w3]을 구함

```
[5]  # 입력 변수에 절편을 위해 1로 이루어진 열 추가
     X = np.c_[np.ones(x.shape[0]), x]
     # 최소제곱법 공식 이용하여 가중치 학습(행렬 계산)
     weights = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(Y)
```

```
     # 학습된 가중치 출력
     print("학습된 가중치:", weights)

     학습된 가중치: [-546.43979144    2.9082713    67.20469902   67.26029602]
```

다중선형회귀 모델 실습

실제 예측 결과를 점수로 표현하기 위해

절편과 기울기 값을 일반식에 대입하여 예측 Y의 값을 구하고 예측에 대한 점수를 출력함

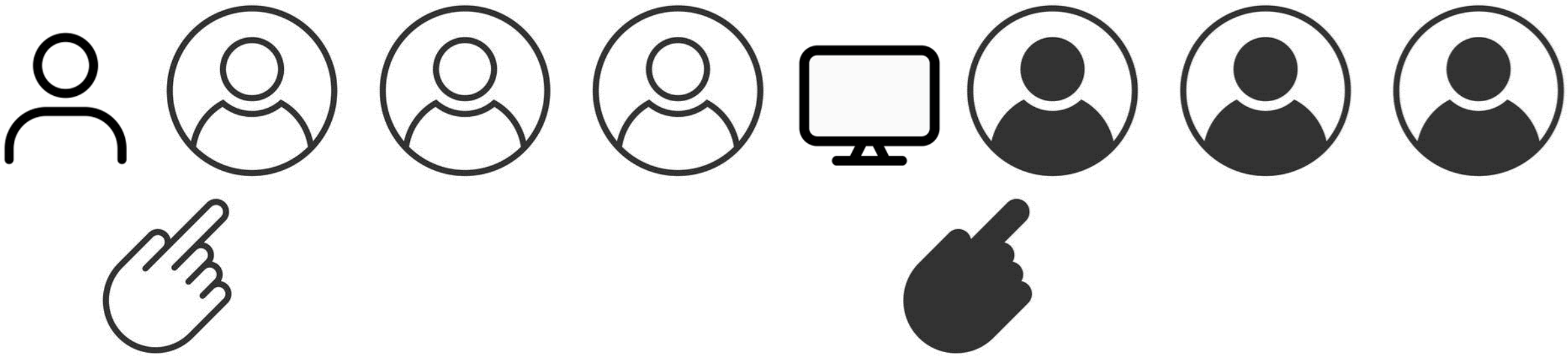
해당 예제 데이터에서는 100점 만점에 94점 정도가 나온 것을 확인할 수 있음

```
[7]  # 구한 절편과 기울기 값으로 일반식에 대입하여 예측 결과 계산
     Y_pred = weights[0] + weights[1]*x[:, 0] + weights[2]*x[:, 1] + weights[3]*x[:, 2]
```

```
# 예측 점수 출력
Y_mean = np.mean(Y)
results = 1 - (np.sum((Y - Y_pred) ** 2) / np.sum((Y - Y_mean) ** 2))

print(f'score: {results * 100:.2f}')
```

```
score: 94.23
```

회귀분석의 한계



- 선형회귀 분석은 분석을 진행하려는 데이터의 특징을 사람이 선택하고 해당 Feature(특징)을 기준으로 분석을 진행한다.
- 선형회귀식에서 최적의 w, b를 찾기 위한 과정이 머신러닝의 과정!
- 머신러닝의 딥러닝은 Feature 선택을 딥러닝-AI network model이 함.

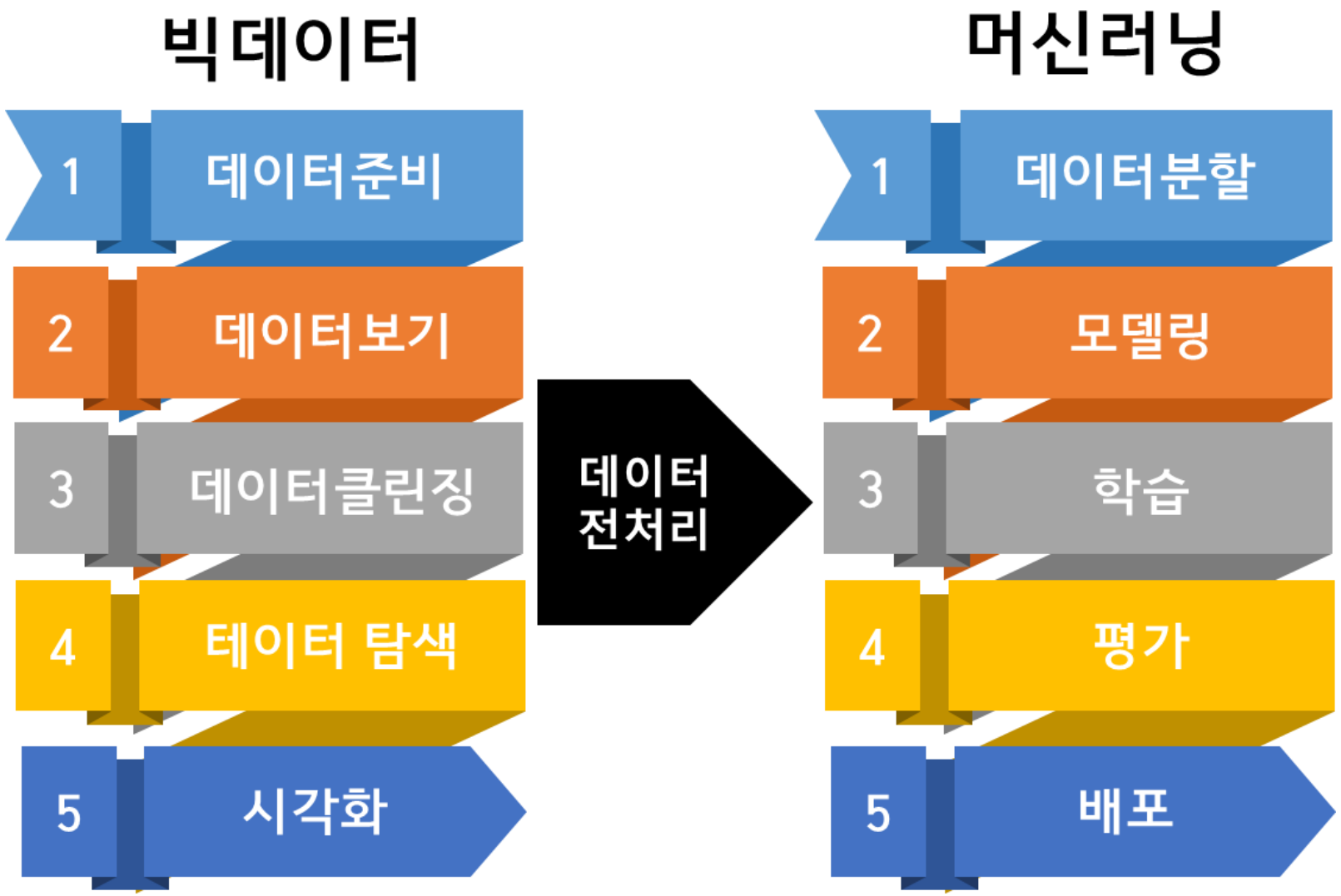
머신러닝 라이브러리

Scikit-learn

- 파이썬 환경에서 실행되는 오픈소스 머신 러닝 라이브러리
- NumPy, Matplotlib, SciPy와 같은 주요 패키지와 통합되어 있어 데이터 과학 및 머신 러닝 프로젝트에 필수적인 라이브러리

사이킷런 설치
pip install scikit-learn
또는 conda install scikit-learn
Import sklearn
Sklearn.__version__

머신러닝 파이프라인


빅데이터
1 데이터준비
2 데이터보기
3 데이터클린징
4 데이터 탐색
5 시각화
데이터
전처리
머신러닝
1 데이터분할
2 모델링
3 학습
4 평가
5 배포