

A background image of a kitchen wall. At the top, a wooden shelf holds several teapots and cups in various colors like blue, red, and white. Below the shelf, a small framed picture hangs on the wall. To the left, a wooden rack holds several knives. In the foreground, a large, abstract painting with green and brown tones is leaning against the wall.

---

**Machine Learning**

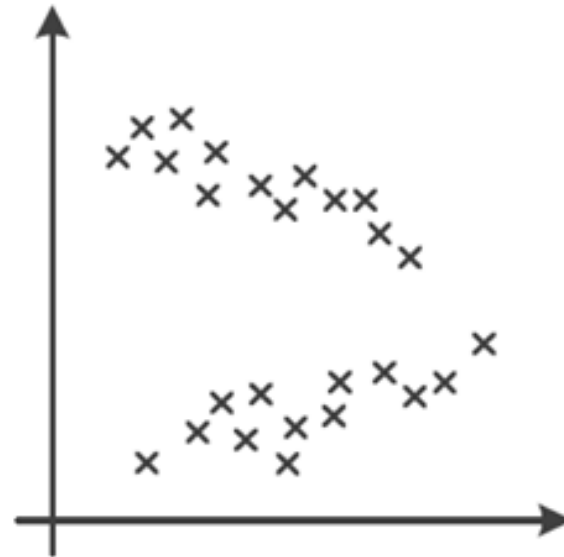
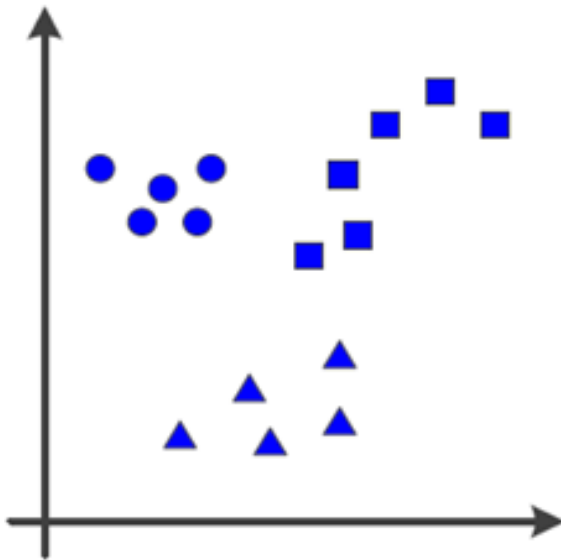
# Linear Regression

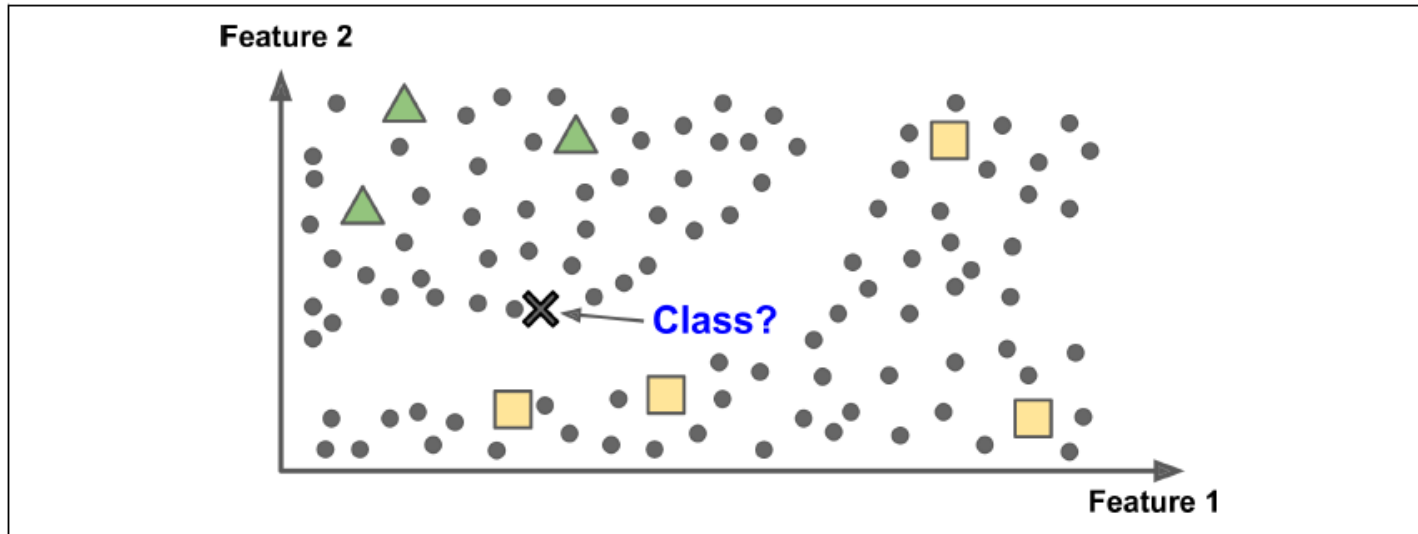
---

김선녕(ksycafe@gmail.com)

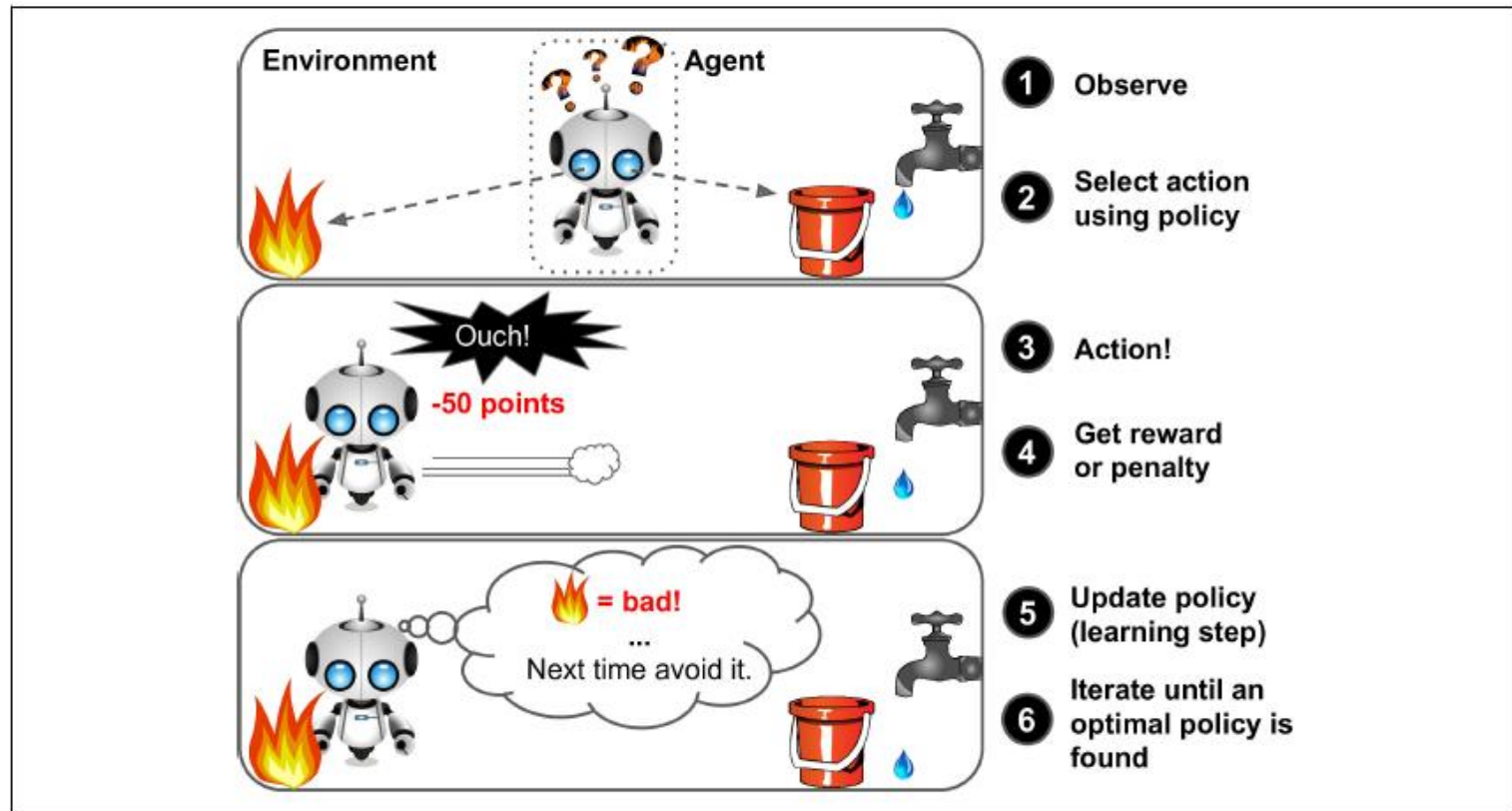
---

- 지도학습(*Supervised Learning*)
- 비지도학습(*Unsupervised Learning*)
- 준지도 학습(*Semi – supervised Learning*)
- 강화학습(*Reinforcement Learning*)





Ref : hands on machine learning 2nd - oreily




Ref : hands on machine learning 2nd - oreily

- 비지도 학습(*Unsupervised Learning*)
  - 군집(clustering)
    - $k$  – 평균( $k$  – *means*)
    - 계층군집분석(*hierarchical cluster analysis*)
    - 이상치탐지(*outlier detection*)
  - 차원 축소(*dimension reduction*)
- 지도 학습(*Supervised Learning*)
  - $k$  – 최근접 이웃( $k$  – *nearest neighbors*, *KNN*)
  - 회귀(*Regression*)
    - 선형회귀(*linear regression*)
    - 로지스틱회귀(*logistic regression*)
  - 서포트벡터머신(*support vector machine*, *SVM*)
  - 결정트리(*decision tree*)와 랜덤 포레스트(*random forest*)
  - 신경망(*neural networks*)
    - 합성곱신경망(*convolutional neural network*, *CNN*)
    - 순환신경망(*recurrent neural network*, *RNN*)

- *Iris Database*(<https://archive.ics.uci.edu/ml/datasets/iris>)
- *Attribute(feature)* : 150개 샘플 각각에 대해 꽃받침 길이, 꽃받침 너비, 꽃잎 길이, 꽃잎 너비를 측정 기록
  - 1. *sepal length in cm*
  - 2. *sepal width in cm*
  - 3. *petal length in cm*
  - 4. *petal width in cm*
  - 5. *class(label): setosa, versicolour, virginica*

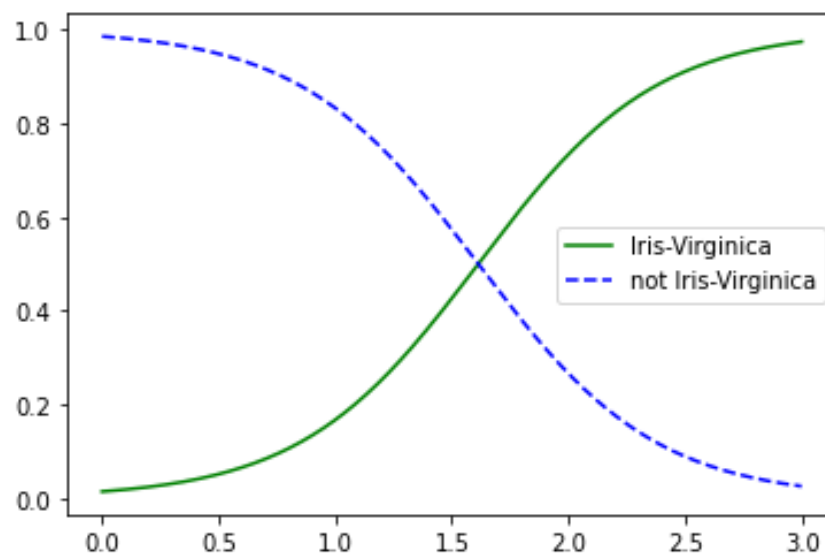
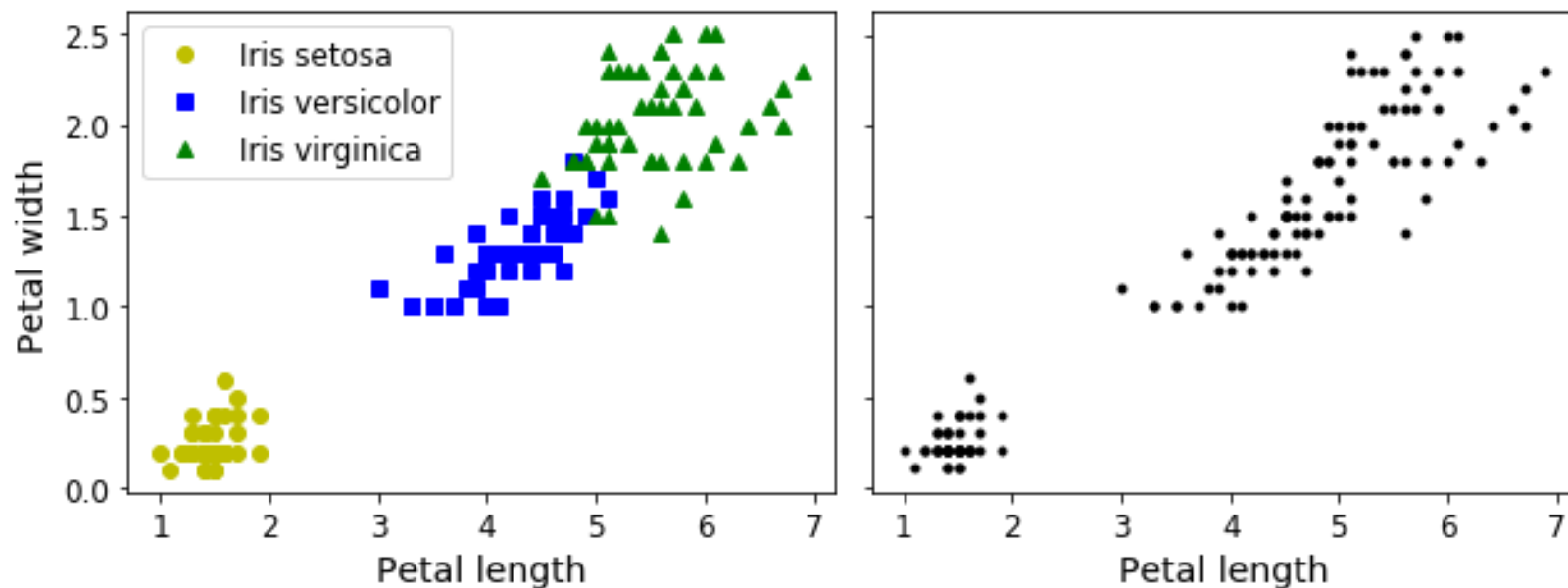


Sepal length ⬆	Sepal width ⬆	Petal length ⬆	Petal width ⬆	Species ⬆		
5.2	3.5	1.4	0.2	<i>I. setosa</i>		
4.9	3.0	1.4	0.2	<i>I. setosa</i>		
4.7	3.2	1.3	0.2	<i>I. setosa</i>		
4.6	3.1	1.5	0.2	<i>I. setosa</i>		
	7.0	3.2	4.7	1.4	<i>I. versicolor</i>	
	6.4	3.2	4.5	1.5	<i>I. versicolor</i>	
	6.9	3.1	4.9	1.5	<i>I. versicolor</i>	
	5.5	2.3	4.0	1.3	<i>I. versicolor</i>	
		6.3	3.3	6.0	2.5	<i>I. virginica</i>
		5.8	2.7	5.1	1.9	<i>I. virginica</i>
		7.1	3.0	5.9	2.1	<i>I. virginica</i>
		6.3	2.9	5.6	1.8	<i>I. virginica</i>



Virginica

Sepal



- 회귀의 유래

- 서양에서는 영국의 유명한 유전학자 갈톤(*F. Galton*, 1822 ~ 1891) 에 의해서 회귀라는 용어가 등장
  - 유전에 관한 논문 「Family Likeness in Stature (1886)」에서 처음 회귀에 대하여 정의.
  - 보편적 회귀법칙 (law of universal regression)  
"일반적으로 키가 큰 부모에게서 키 큰 자녀가, 키가 작은 부모에게서 키 작은 자녀가 태어난다. 그렇지만 자녀들의 평균 키는 전체 인구의 평균 키로 회귀하는 경향이 있다.  
즉, 키가 큰 부모이든 작은 부모이든 그들에게서 태어난 자녀들의 평균 키는 전체 평균 키 수준에 접근하는 현상으로 나타난다는 것이다"

- 회귀분석

- 변수와 변수 사이의 상관관계를 알아보기 위한 통계적 분석방법, 독립변수의 값에 의하여 종속변수의 값을 예측하기 위함
  - 독립변수(*independent variable*) : 종속변수에 영향을 미치는 변수
  - 종속변수(*dependent variable*) : 분석의 대상이 되는 변수
- 머신러닝 : 독립변수는 특징(*feature*), 종속변수는 결정 값(*target, predict*)
- 단순회귀분석(단변량, *simple regression analysis*)
  - 하나의 종속변수와 하나의 독립변수의 관계를 분석
- 다중회귀분석(다변량, *multiple regression analysis*):
  - 하나의 종속변수와 둘 이상의 독립변수간의 관계를 분석

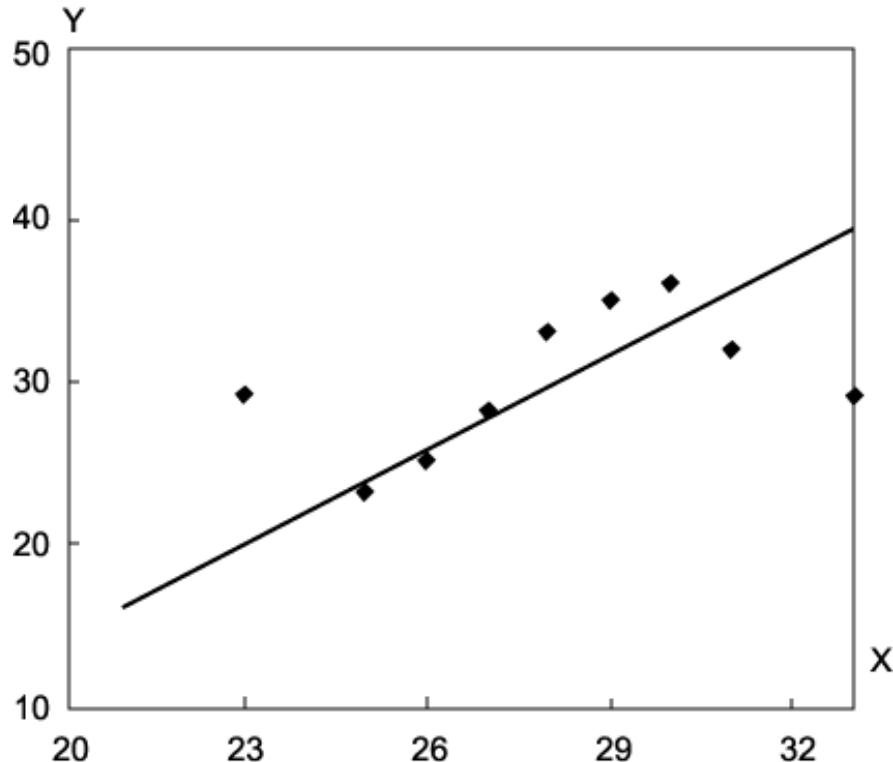


## Example

- 기온이 섭씨 1도 올라감에 따라 아이스크림 판매량에 미치는 영향(예측)은?

(단위 : 도, 십만 개)

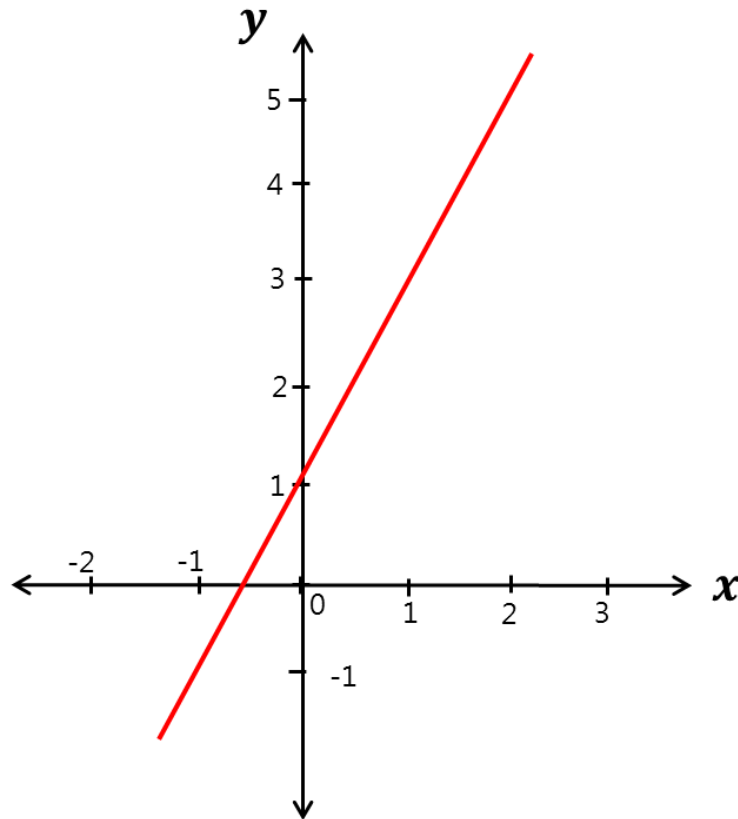
온 도 (X)	23	25	26	27	28	29	30	31	33
판매량 (Y)	29	23	25	28	33	35	36	32	29



- 두 변수 사이에 positive(+) or negative(-)의 관계가 존재
- 두 변수는 선형관계 또는 비선형 관계
- 두 변수 사이에 존재하는 관련성(함수관계)

# 1차 방정식

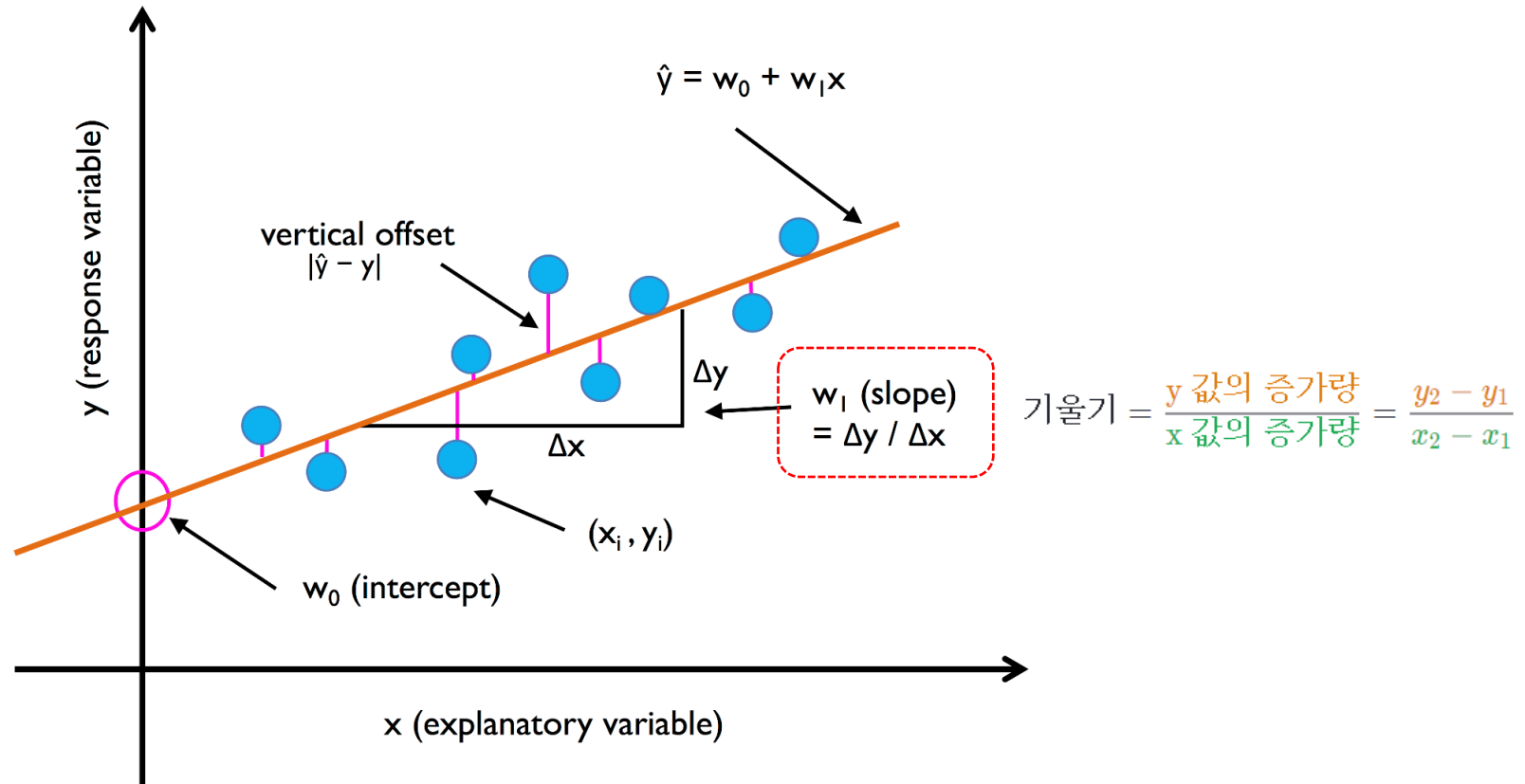
- $y = ax + b$
- 기울기(slope) :  $a$ ,  $y$ 절편(intercept) :  $b$



- 하나의 특성( $x$ )과 연속적인 타겟( $y$ )사이의 관계를 모델링

$$\hat{y} = w_0 + w_1 x$$

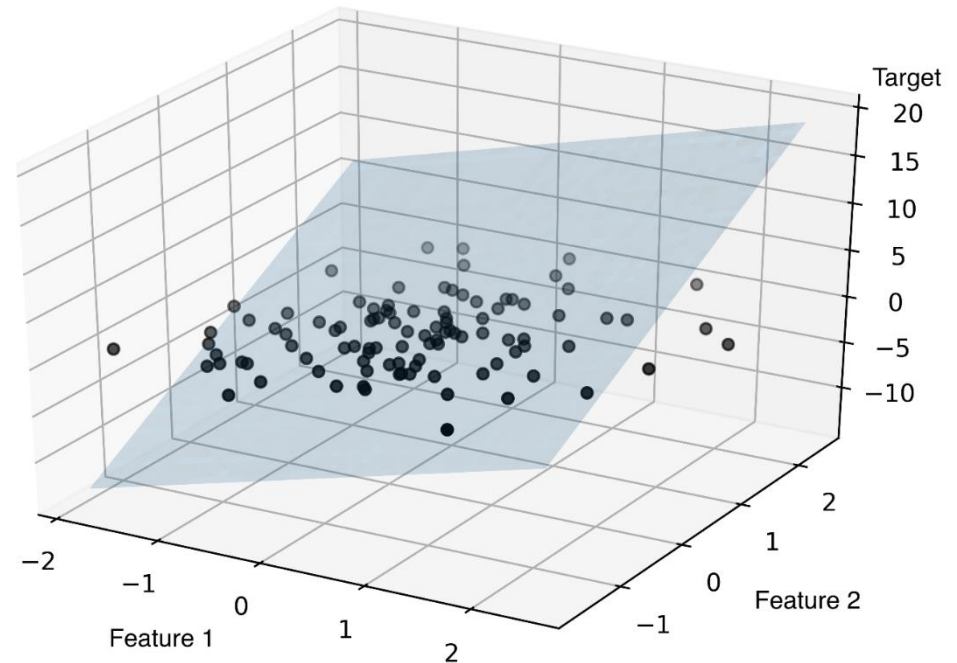
- 회귀선(regression line) : 데이터에 가장 잘 맞는 직선
- 오차(error) : 회귀선과 데이터 사이의 직선 거리



특성이 여러 개 :  $x_1, x_2, \dots, x_n$

$$\begin{aligned}\hat{y} &= w_0x_0 + w_1x_1 + \dots + w_nx_n \\ &= \sum_{i=0}^n w_ix_i = \mathbf{w}^T \mathbf{x}\end{aligned}$$

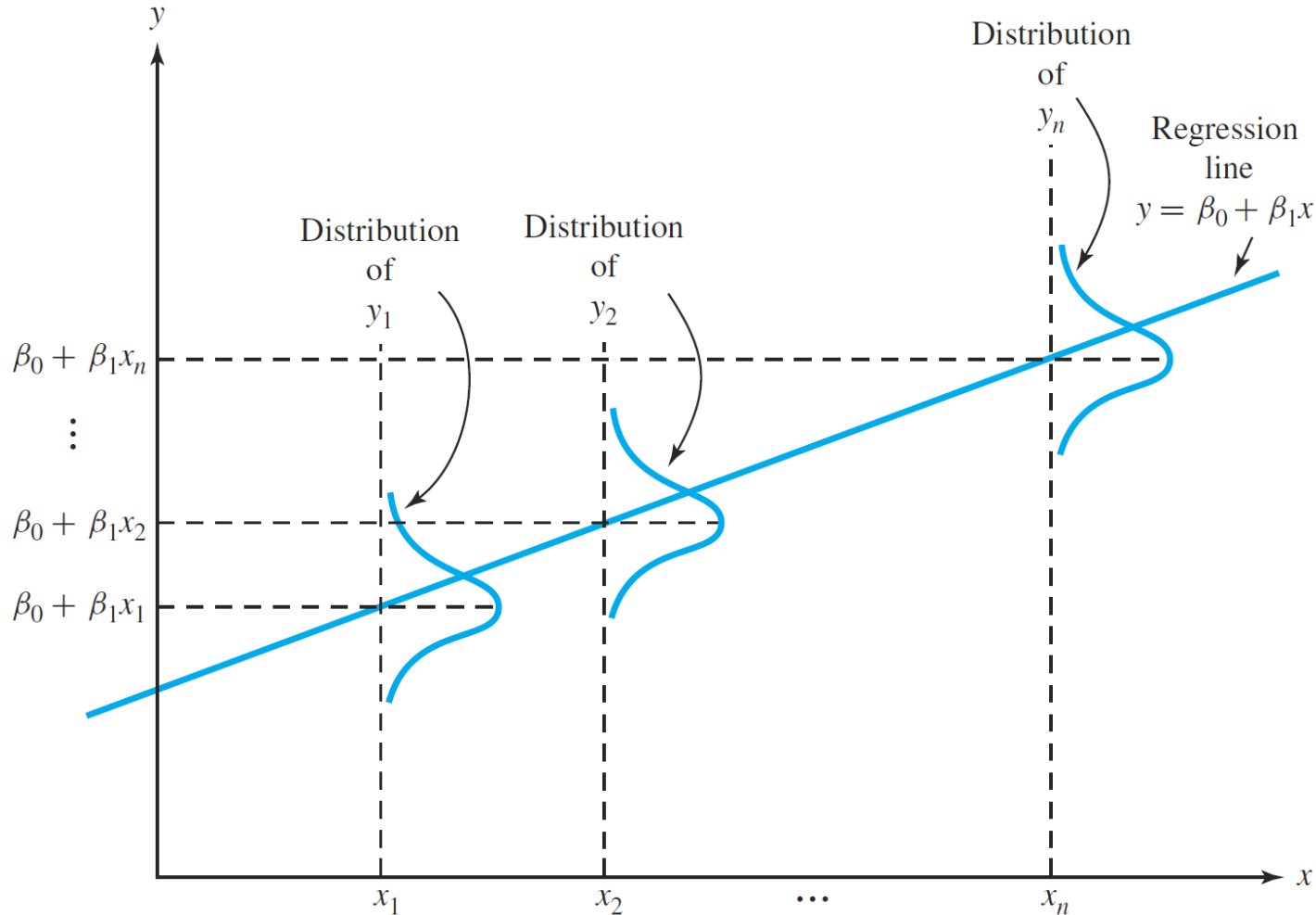
$$\mathbf{v} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$$



$$y = (w_1 \quad w_2 \quad w_3) \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = (w_1x_1 + w_2x_2 + w_3x_3)$$

# 회귀분석의 목적은 최적의 $\beta_0, \beta_1$ 를 찾는 것

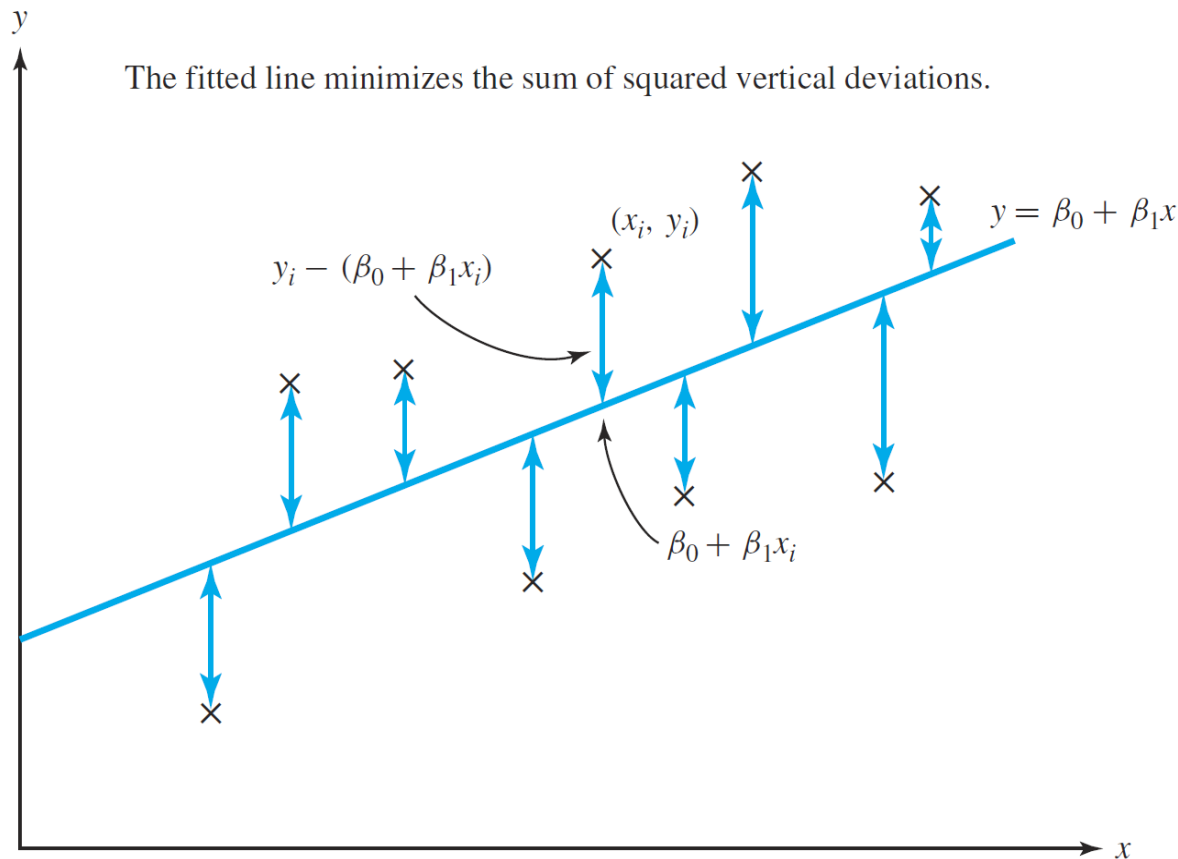
$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i \quad (\epsilon_i : \text{오차항})$$



- 오차항  $\epsilon_1, \dots, \epsilon_n$ : 오차분산  $\sigma^2$ 에 대하여 정규분포 한다고 가정  $N(0, \sigma^2)$
- 평균은 0. 실제 값이 회귀선상에 있는 점을 중심으로 분포되어 있고 분산  $\sigma^2$ 은 모든  $x$  값에 동일

- 회귀직선  $y = \beta_0 + \beta_1 x$  를 데이터  $(x_1, y_1), \dots, (x_n, y_n)$ 에 적합(*fit*)시키는 과정
- 데이터셋에 가장 근접한 직선을 찾는 과정

$$\text{minimize} = \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2$$



- 상관계수(*Correlation coefficient*) : 두 변수들 간의 연관 정도

$m_x$  : mean of  $X$

$m_y$  : mean of  $Y$

$s_x$  : standard deviation of  $X$ ,  $\sqrt{\frac{\sum_{i=1}^n (X_i - m_x)^2}{n - 1}}$

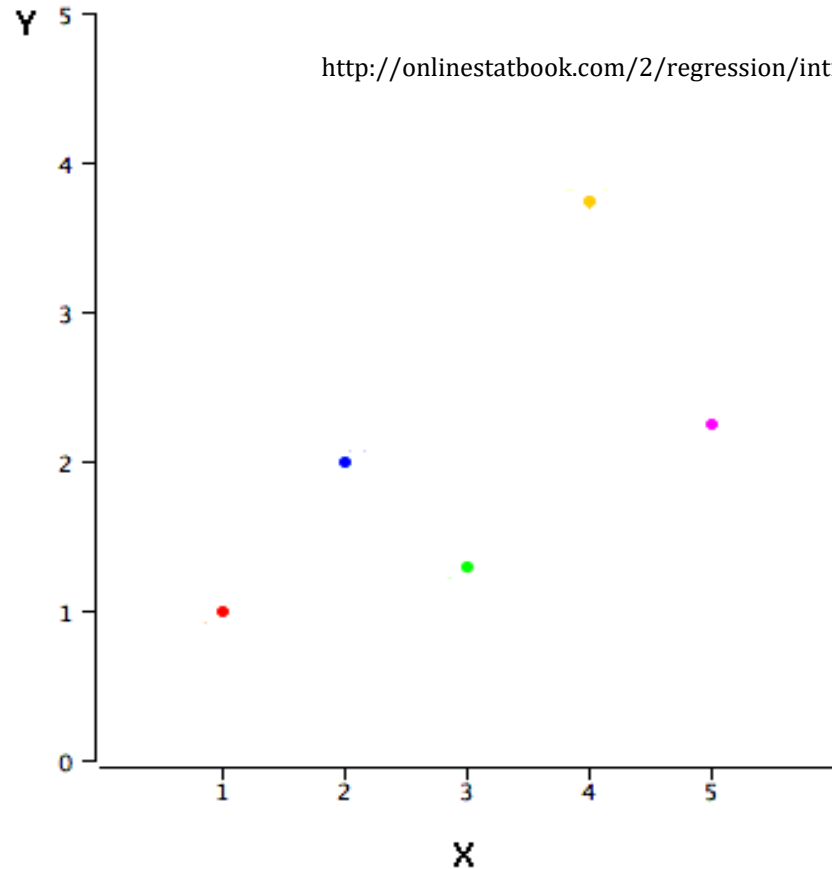
$s_y$  : standard deviation of  $Y$ ,  $\sqrt{\frac{\sum_{i=1}^n (Y_i - m_y)^2}{n - 1}}$

$r$  : the correlation between  $X$  and  $Y$  (**Pearson's  $r$** )

$$\textbf{Pearson's } r = \frac{\sum xy}{\sqrt{\sum x^2 \sum y^2}}, \quad (x = X - m_x, y = Y - m_y)$$

# 최적의 선을 찾아보자

X	Y
1.00	1.00
2.00	2.00
3.00	1.30
4.00	3.75
5.00	2.25



- $$\text{Pearson's } r = \frac{\sum xy}{\sqrt{\sum x^2 \sum y^2}}$$

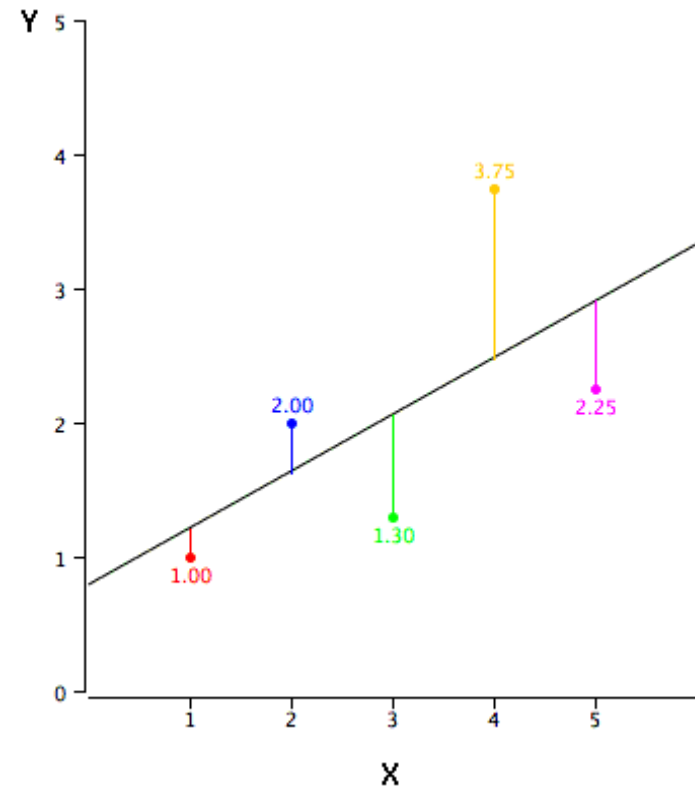
$m_x$	$m_y$	$s_x$	$s_y$	$r$
3	2.06	1.581	1.072	0.627

- $$\text{slope}(a) = r \cdot s_x / s_y = (0.627) \cdot (1.072) / 1.581 = 0.425$$
- $$\text{intercept}(b) = m_y - a \cdot m_x = 2.06 - (0.425)(3) = 0.785$$



- $\hat{Y}$  : predicted values
- $Y - \hat{Y}$  : errors of prediction
- $\hat{Y} = 0.425X + 0.786$
- *Best fitting line*  
*minimizes the sum of the squared errors of prediction.*

$X$	$Y$	$\hat{Y}$	$Y - \hat{Y}$	$(Y - \hat{Y})^2$
1.00	1.00	1.210	-0.210	0.044
2.00	2.00	1.635	0.365	0.133
3.00	1.30	2.060	-0.760	0.578
4.00	3.75	2.485	1.265	1.600
5.00	2.25	2.910	-0.660	0.436

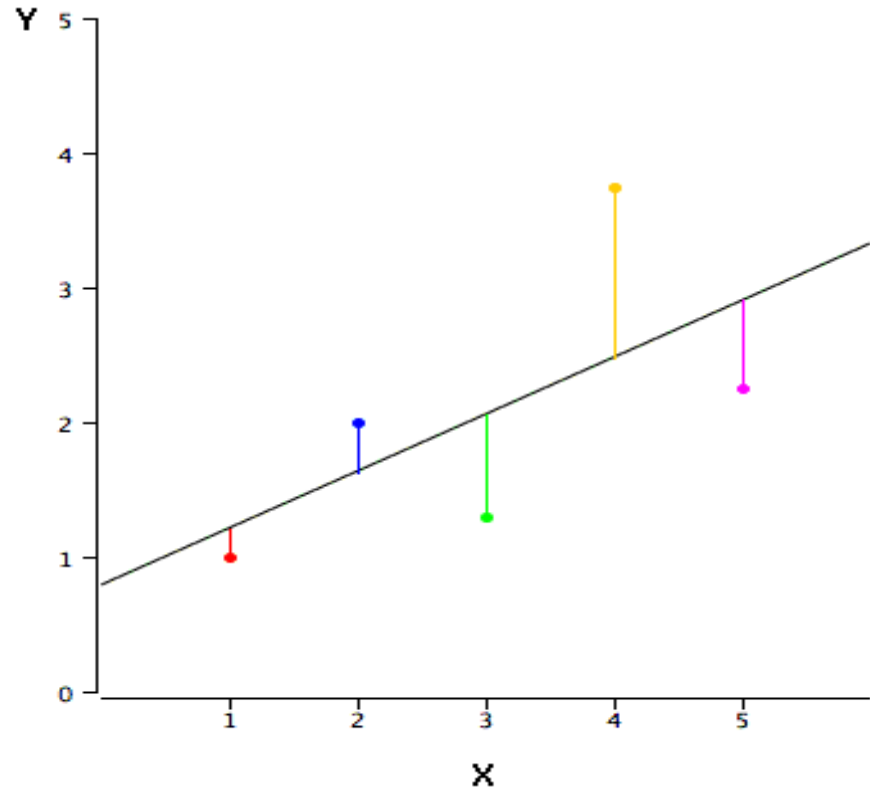


- 실제 값에서 예측 값간의 차이 제곱의 합이 최소일 때의 값을 찾는 방법

- $(\hat{Y} - Y)^2$      $\hat{y}$ : 예측값,  $Y$ : 실제 값

$$\frac{(\hat{y}^{(1)} - y^{(1)})^2 + (\hat{y}^{(2)} - y^{(2)})^2 + \dots + (\hat{y}^{(5)} - y^{(5)})^2}{5}$$

$$= \frac{1}{n} \sum_{i=1}^n (\hat{Y}_{(i)} - Y_{(i)})^2$$



# LinearRegression

19

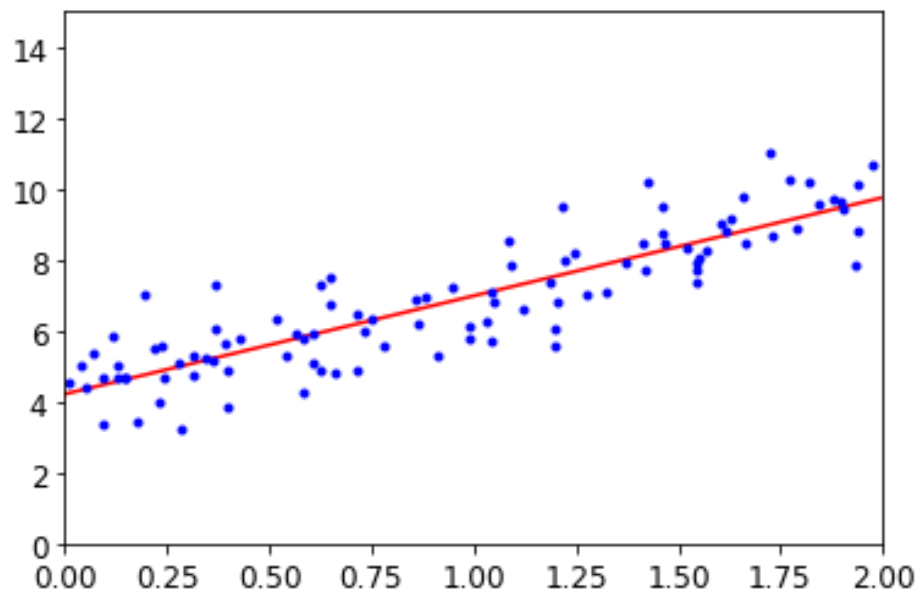
```
from sklearn.linear_model import LinearRegression
```

```
lin_reg = LinearRegression()  
lin_reg.fit(X, y)  
lin_reg.intercept_, lin_reg.coef_
```

$\beta_0, \beta_1$  (array([4.04967799]), array([[3.01093259]]))

```
X_new = np.array([[0], [2]])  
lin_reg.predict(X_new)
```

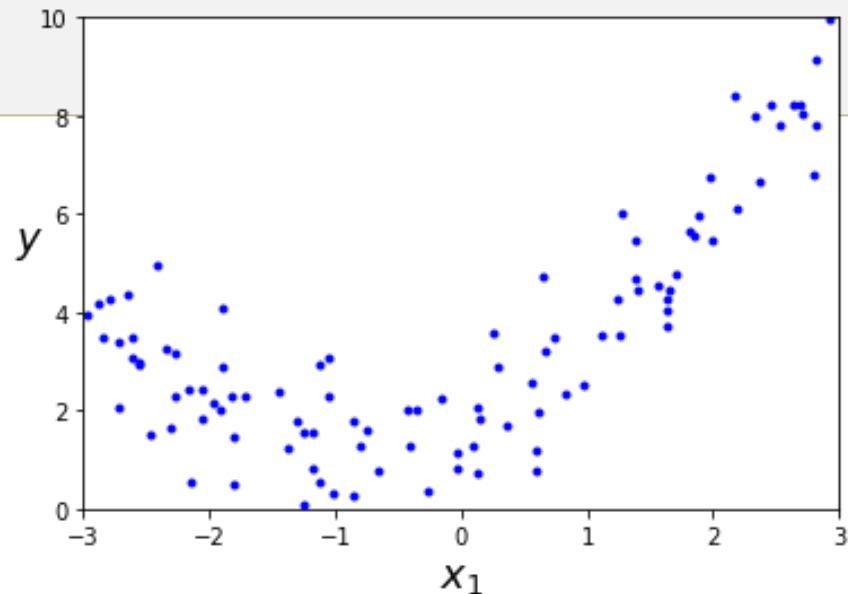
[0], [1] 의 예측 결과 : array([[ 4.04967799], [10.07154317]])



```
import numpy.random as rnd
np.random.seed(42)

# 2차 방정식으로 비선형 데이터 생성
m = 100
X = 6 * np.random.rand(m, 1) - 3
y = 0.5 * X**2 + X + 2 + np.random.randn(m, 1)

plt.plot(X, y, "b.")
plt.xlabel("$x_1$", fontsize=18)
plt.ylabel("$y$", rotation=0, fontsize=18)
plt.axis([-3, 3, 0, 10])
plt.show()
```



```
# 훈련세트에 있는 각 특성을 제공 (2차 다항)하여 새로운 특성 추가
from sklearn.preprocessing import PolynomialFeatures
poly_features = PolynomialFeatures(degree=2, include_bias=False)
X_poly = poly_features.fit_transform(X)
X[0]
```

```
array([-0.75275929])
```

```
X_poly[0]
```

```
array([-0.75275929, 0.56664654])
```

```
lin_reg = LinearRegression()
lin_reg.fit(X_poly, y) # X_poly : 원래 특성 x와 새로운 특성 포함
lin_reg.intercept_, lin_reg.coef_
```

```
(array([1.78134581]), array([[0.93366893, 0.56456263]]))
```

- 실제함수 :  $y = 0.5x_1^2 + 1.0x_1 + 2.0$
- 예측모델 :  $y = 0.56x_1^2 + 0.93x_1 + 1.78$

```
X_new=np.linspace(-3, 3, 100).reshape(100, 1)
X_new_poly = poly_features.transform(X_new)
y_new = lin_reg.predict(X_new_poly)
plt.plot(X, y, "b.")
plt.plot(X_new, y_new, "r-", linewidth=2, label="Predictions")
plt.xlabel("$x_1$", fontsize=18)
plt.ylabel("$y$", rotation=0, fontsize=18)
plt.legend(loc="upper left", fontsize=14)
plt.axis([-3, 3, 0, 10])
plt.show()
```

