6. 머신러닝 모델부터 예측까지 기초 이론

KOREATECH 한국기술교육대학교

KOREATECH 능력개발교육원

CONTENTS

KOREATECH 한국기술교육대학교

KOREATECH HRDI 능력개발교육원

01 선형 회귀 (Linear Regression)

KOREATECH 한국기술교육대학교

KOREATECH 능력개발교육원

선형 회귀(Linear Regression)란?

- 변수 사이의 선형적인 관계를 모델링 한 것.
- 통계학: 관찰된 연속형 변수들에 대해 두 변수 사이의 모형을 구한뒤 적합도를 측정해 내는 분석 방법
- 선형: 직선적이다
- 회귀: 돌아오다

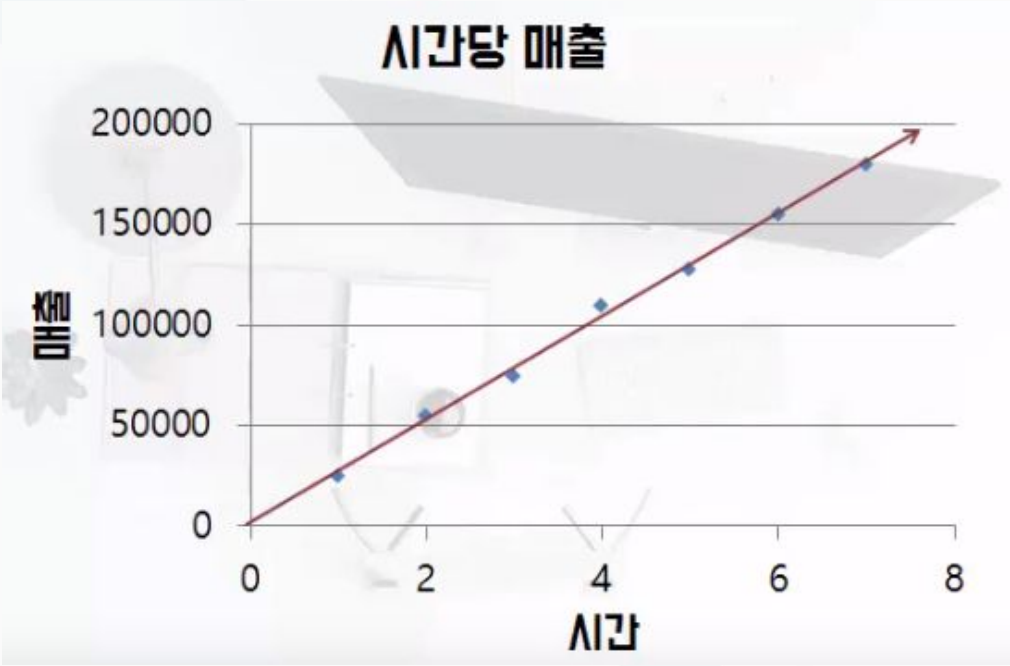
선형 회귀(Linear Regression)란?

- 대표적으로 '일하는 시간과 매출액', '공부한 시간과 성적' 등이 선형적인 관계를 가지고 있습니다. 예를 들어 장사꾼의 노동 시간과 매출 데이터가 다음과 같다고 가정해봅니다.

하루 노동 시간	하루 매출
1	25,000
2	55,000
3	75,000
4	110,000
5	128,000
6	155,000
7	180,000

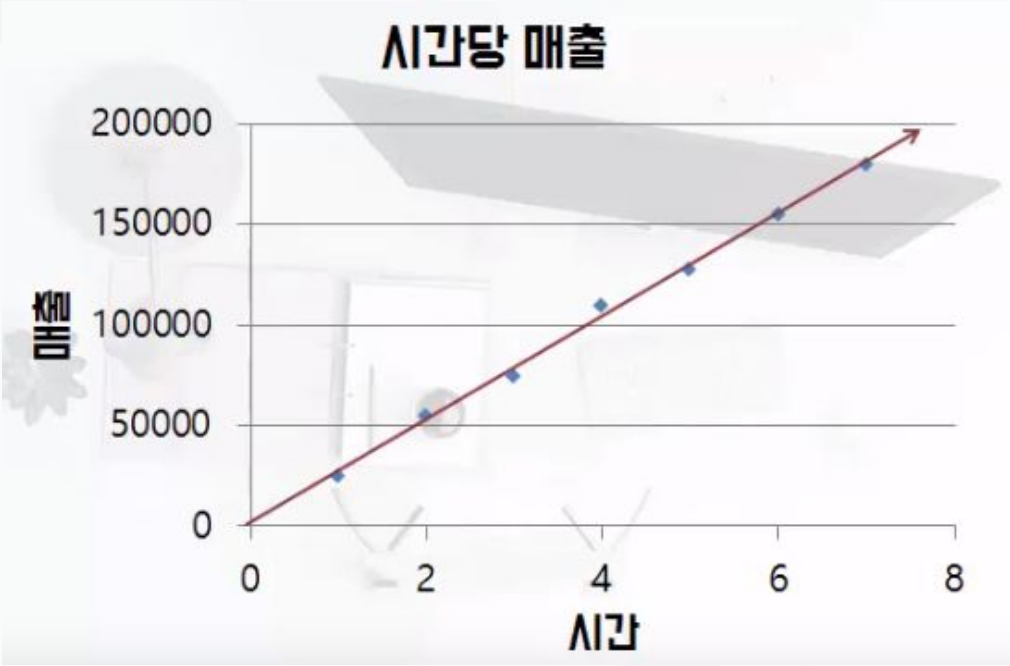
선형 회귀(Linear Regression)란?

- 선이 그려짐
- 학습을 시킨다 - 주어진 데이터를 학습시켜서 가장 합리적인 '직선'을 찾아내는 것
- 데이터는 3개 이상일 때 의미가 있슴 (데이터가 2개라면 단순히 두 점을 잇는 직선이 되어버림)



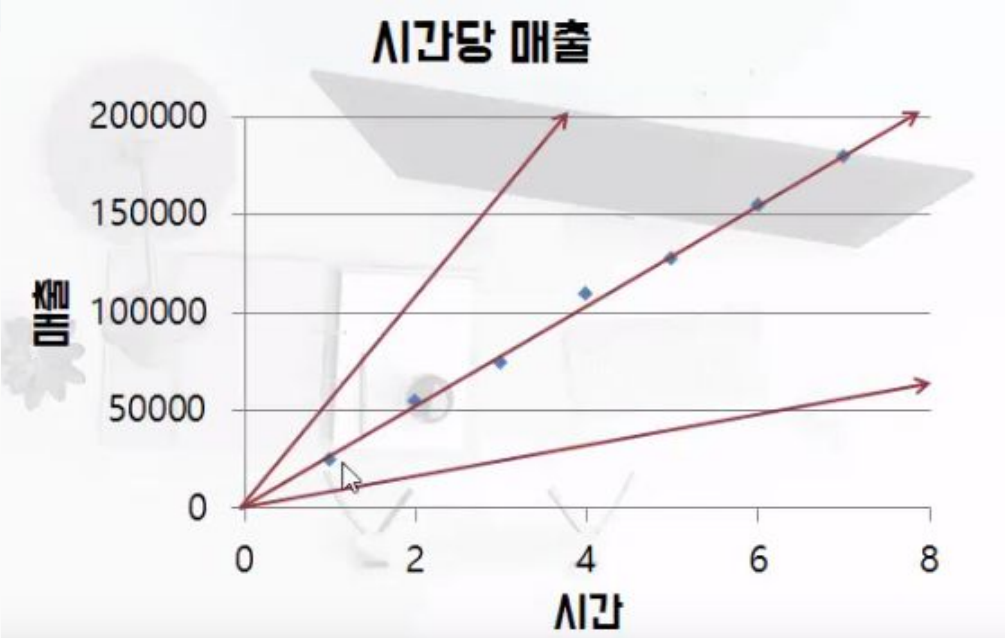
선형 회귀(Linear Regression)란?

- 가설을 수정해가며 가장 합리적인 식을 찾아내는 과정이 선형 회귀 모델을 이용한 기계학습



선형 회귀(Linear Regression)

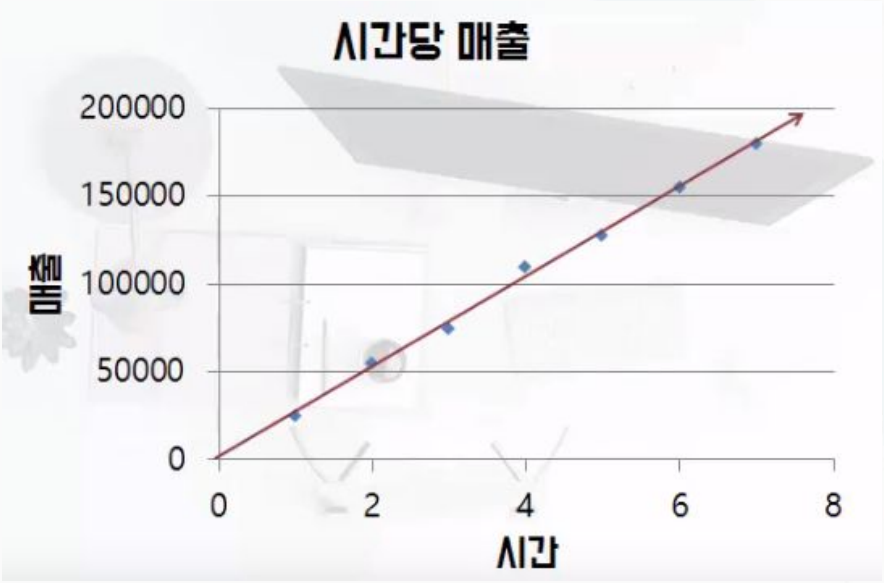
가장 합리적인 선이란?



가설

방정식을 이용해 직선을 표현

$$H(x) = Wx + b$$

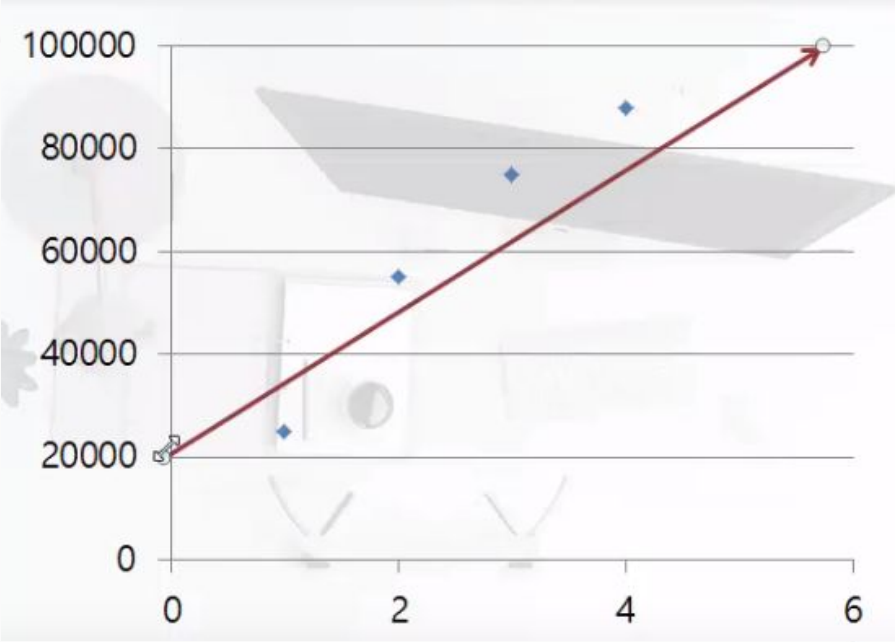


선형 회귀 비용(Cost)

- 가설이 얼마나 정확한 지 판단하는 기준
- 비용은 알고리즘 분야에서 굉장히 중요한 개념
- 해당 직선이 얼마나 '정확한' 데이터인지 판단하기 위해선 비용을 계산

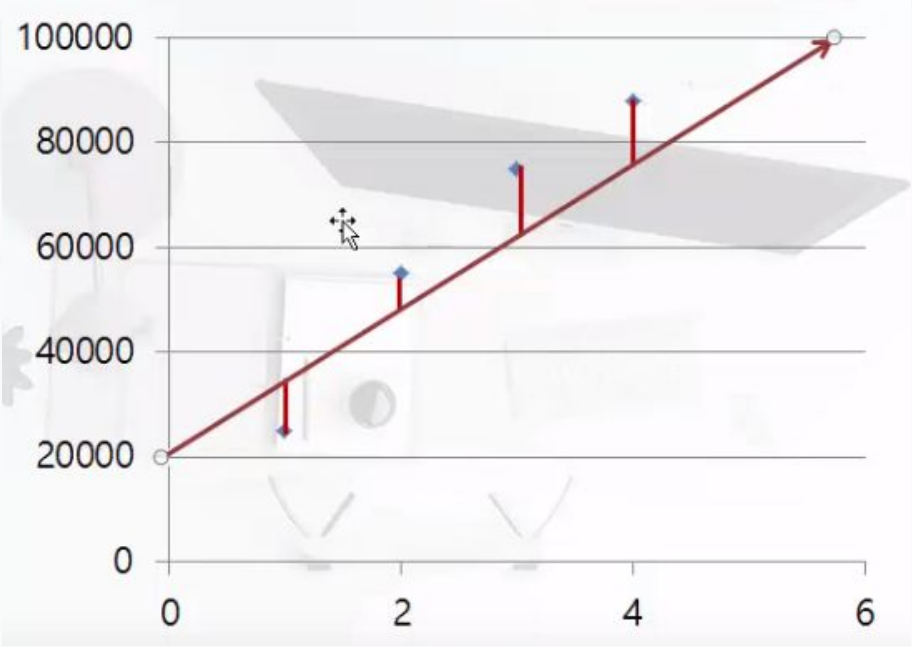
선형 회귀 비용(Cost)

각 데이터에 아래의 직선을 그렸다고 가정



선형 회귀 비용(Cost)

- 비용을 계산할 때는 '데이터와 직선과의 거리'를 구해서 계산
- 예측 값과 실제 값의 차이점이 각 점에 대한 비용



선형 회귀 비용(Cost)

전체 Cost = (예측 값 - 실제 값)^2 의 평균

- H(x) = Wx + b에서, 현재의 W, b값과 데이터를 이용하면 비용함수를 구함
- 비용 함수로 구한 비용이 적을수록 좋음

$$cost(W,b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2$$

선형 회귀 비용(Cost)

다음 chapter에서 배울 인공신경망에서

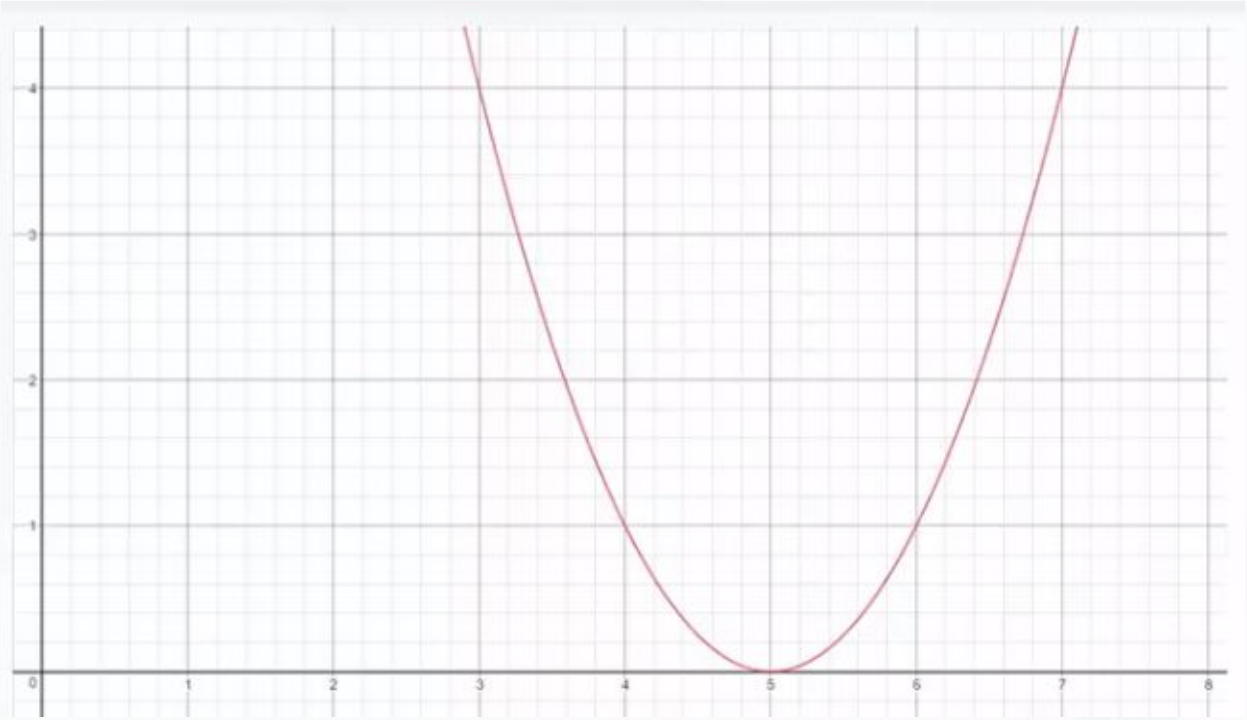
Cost function = Loss function

경사 하강(Gradient Descent)

H(x) = Wx + b을 간단히 H(x) = Wx로 바꿀때

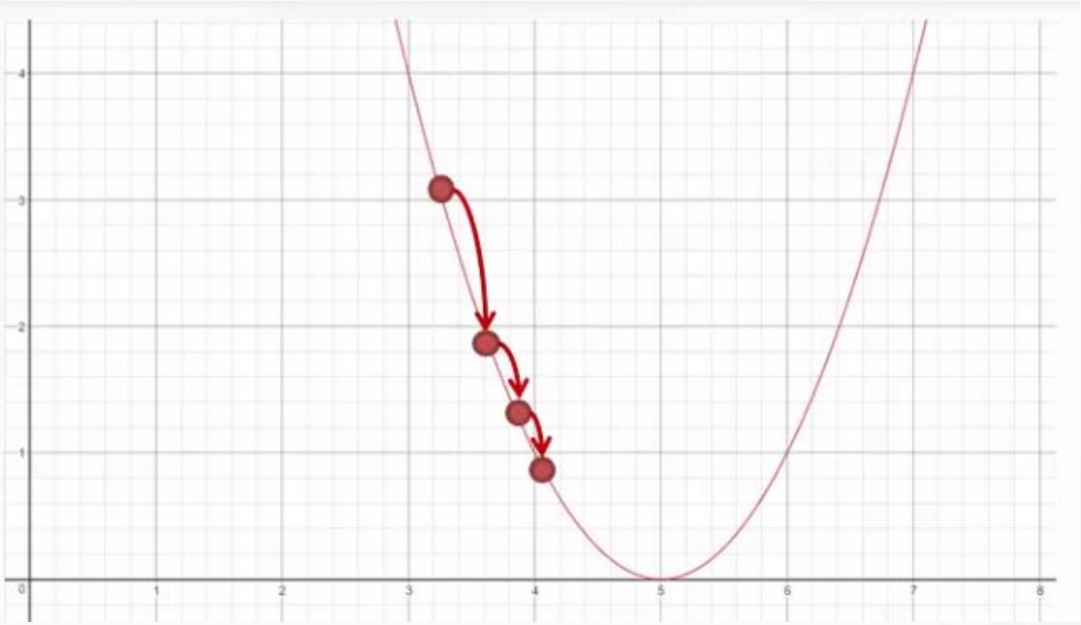
$$\text{비용함수} = (Wx - y)^2$$

경사 하강(Gradient Descent)



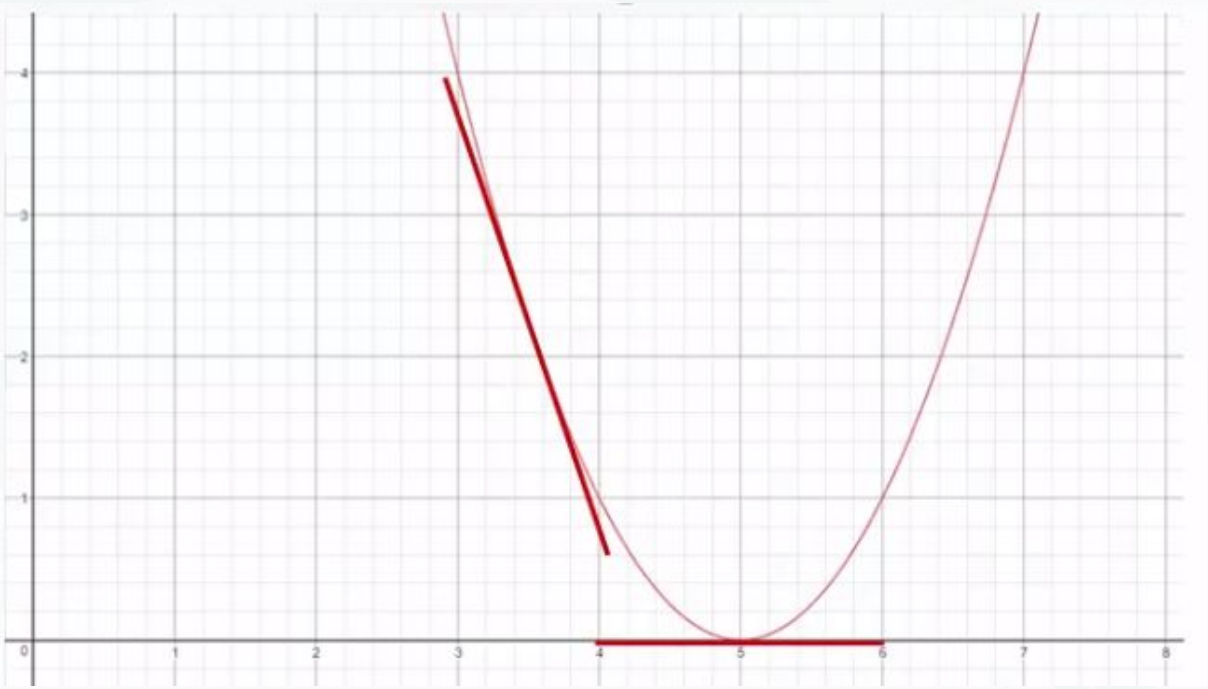
경사 하강(Gradient Descent)

점프를 하면서 학습하는 과정에서 점프(Jump) 폭을 적당히 조절



미분(derivative)과 기울기(slope)

미분을 수행해서 곡선의 골짜기로 도달하도록 합니다.

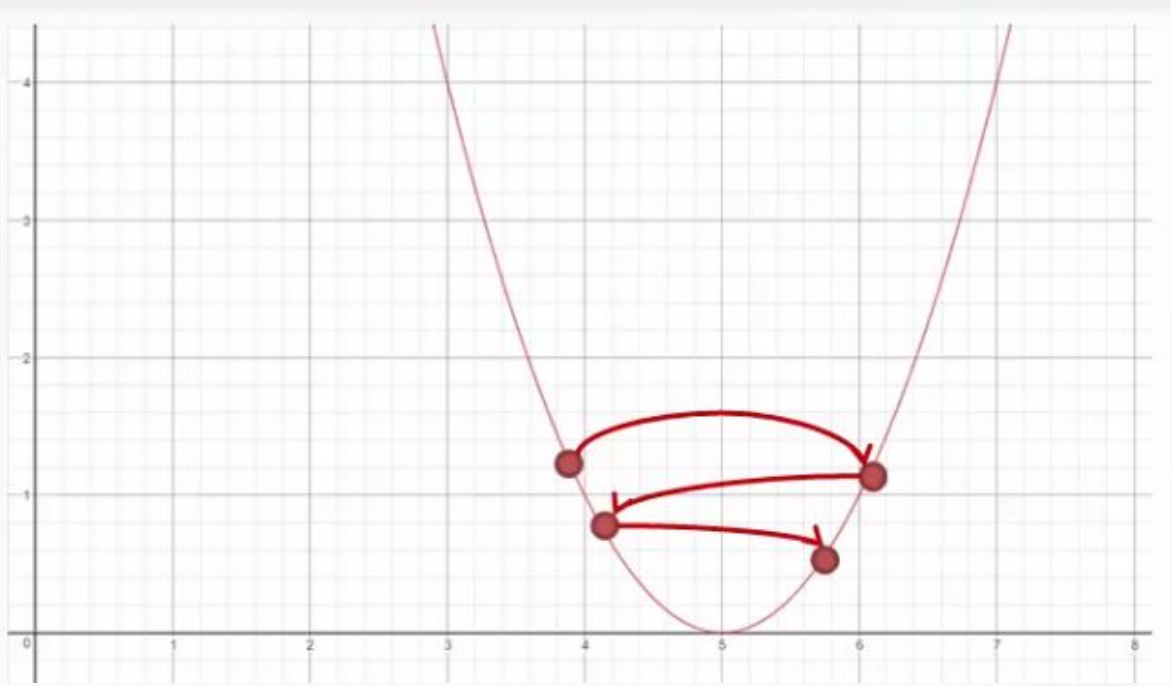


Gradient Descent - 점프

- 곡선의 특성상 초반에 많은 폭으로 변화
- 너무 작게 점프하면 오랫동안 학습해야 함
- 너무 크게 점프하면 학습 결과가 부정확할 수 있슴

Gradient Descent - 점프

너무 크게 점프한 경우



Gradient Descent

- 많은 머신러닝 라이브러리는 경사 하강 라이브러리를 효과적으로 제공
- 수학적 수식을 이용해 직접 경사 하강을 구현할 필요가 없음
- 간단히 실제 프로그램에 적용 가능

sklearn LinearRegression 구현하기(implementation)

- import하기
- https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

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

sklearn LinearRegression

- 모델을 생성하고, 그 안에 X, y 데이터를 fit

```
line_fitter = LinearRegression()
line_fitter.fit(X, y)
```

sklearn LinearRegression

```
line_fitter = LinearRegression()
line_fitter.fit(X, y)
```

fit() 메서드는 선형 회귀 모델에 얻을수 있는 값

- 기울기: line_fitter.coef_
- 절편: line_fitter.intercept_

예: y = 5x + 2

- 5 - coefficient(기울기)
- 2 - (y-)intercepts(절편)

기울기 - H, 절편 - b

`fit(X, y, sample_weight=None)`

Fit linear model.

Parameters:	**X :** ***{array-like, sparse matrix} of shape (n_samples, n_features)*** Training data **y :** ***array-like of shape (n_samples,) or (n_samples, n_targets)*** Target values. Will be cast to X's dtype if necessary **sample_weight :** ***array-like of shape (n_samples,), default=None*** Individual weights for each sample *New in version 0.17:* parameter *sample_weight* support to LinearRegression.
Returns:	**self :** ***returns an instance of self.***

*싸이킷런 단순 선형회귀에서는 최소제곱법(Ordinary Least Squares)을 활용

02 뉴럴 네트워크 (Neural Network)

KOREATECH 한국기술교육대학교

KOREATECH 능력개발교육원

어떤 장난꾸러기 학생이 선생님의 부등호를 지웠네요.

어떤 부등호였들까요?

Math Quiz #1 - Teacher's Answer Key

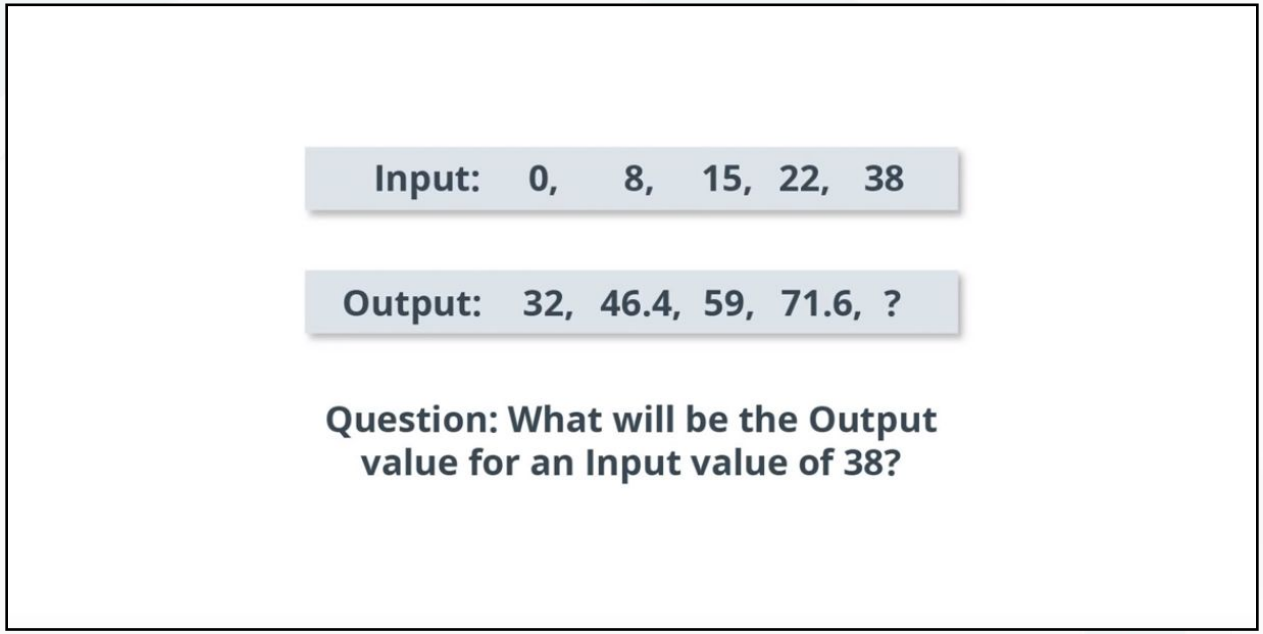
1) 2 4 5 = 3	5) 6 2 2 = 10
2) 5 2 8 = 2	6) 3 1 1 = 2
3) 2 2 1 = 3	7) 5 3 4 = 11
4) 4 2 2 = 6	8) 1 8 1 = 7

아래와 같은 input과 output이 있다고 가정합니다.

0을 넣었을때 32가 나오고, 8을 넣었을때 46.4가 나옵니다.

Input:	0,	8,	15,	22
Output:	32,	46.4,	59,	71.6

38을 넣었을때 어떤 숫자가 나올까요?



100.4가 나옵니다.

그럼, 이건 어떤 공식에 의해 나온 output일까요?

Input:	0,	8,	15,	22,	38
Output:	32,	46.4,	59,	71.6,	100.4

F = C * 1.8 + 32는 화씨를 섭씨로 바꾸는 공식입니다.

Input:	0,	8,	15,	22,	38
Output:	32,	46.4,	59,	71.6,	100.4

$$F = C * 1.8 + 32$$

F = *Fahrenheit*
C = *Celsius*

입력값은 섭씨(C = Celsius)로 표시됩니다.

Input:	0,	8,	15,	22,	38
Output:	32,	46.4,	59,	71.6,	100.4

$$F = C * 1.8 + 32$$

F = *Fahrenheit*
C = *Celsius*

출력값은 화씨(F = Fahrenheit)로 표시됩니다.

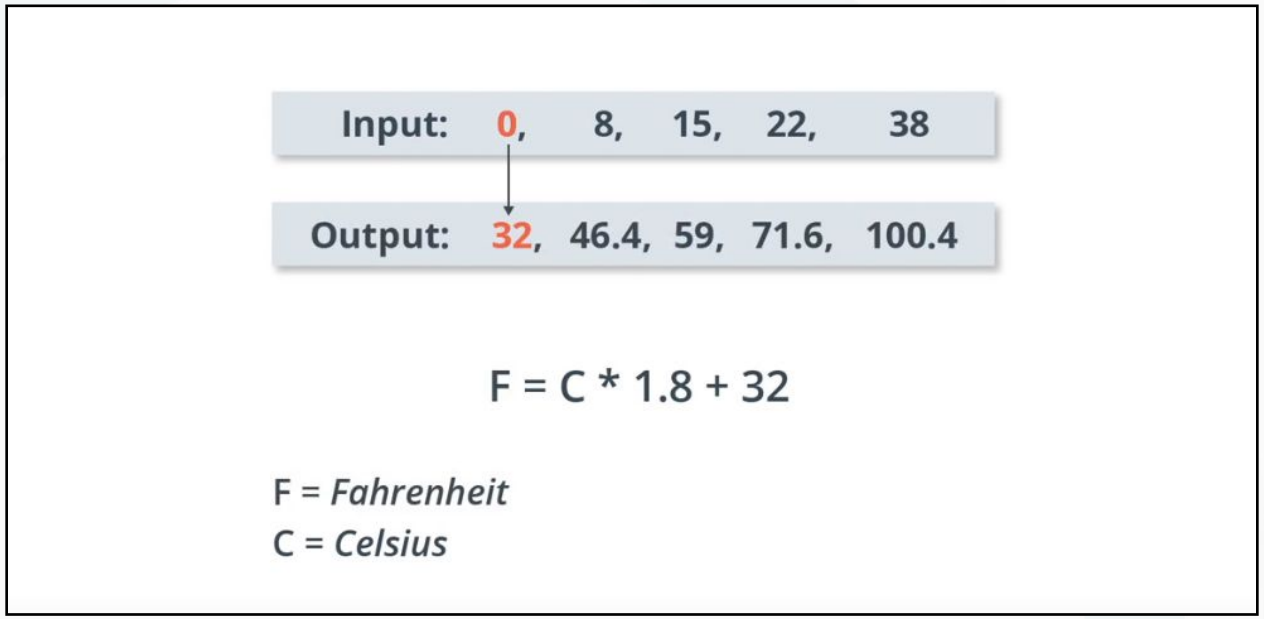
Input: 0, 8, 15, 22, 38

Output: 32, 46.4, 59, 71.6, 100.4

$$F = C * 1.8 + 32$$

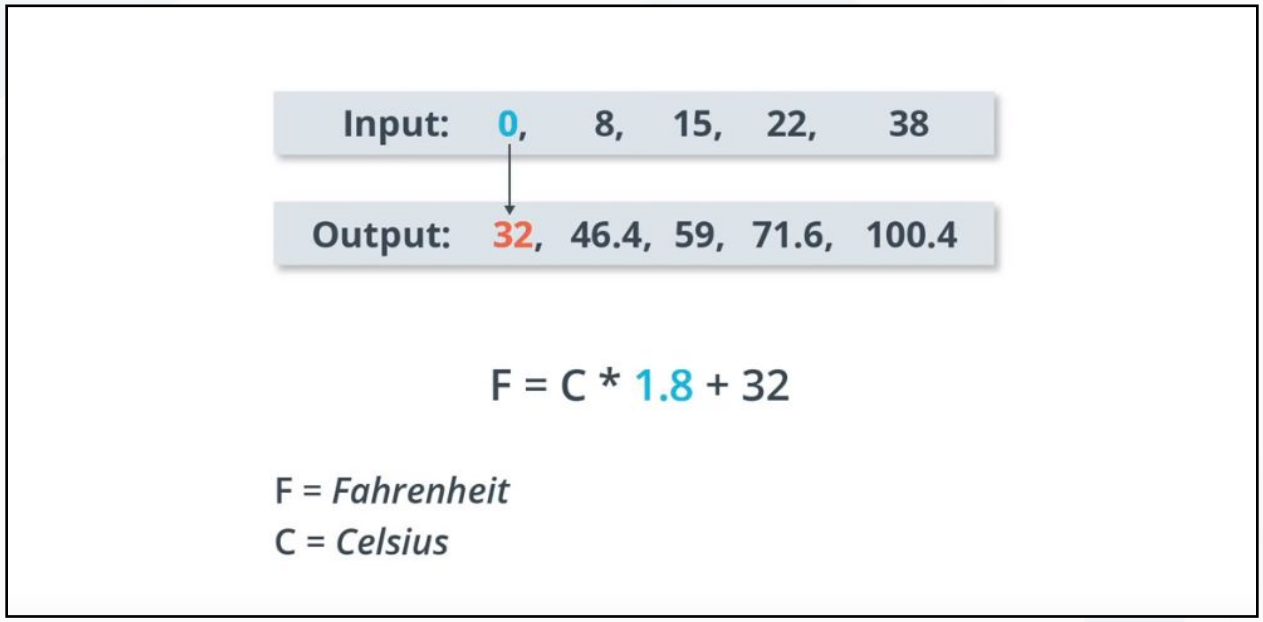
F = *Fahrenheit*
C = *Celsius*

다시 말해 입력은 섭씨 온도의 온도 값을 나타내고 출력은 화씨로 나타낸 해당 온도를 나타냅니다.



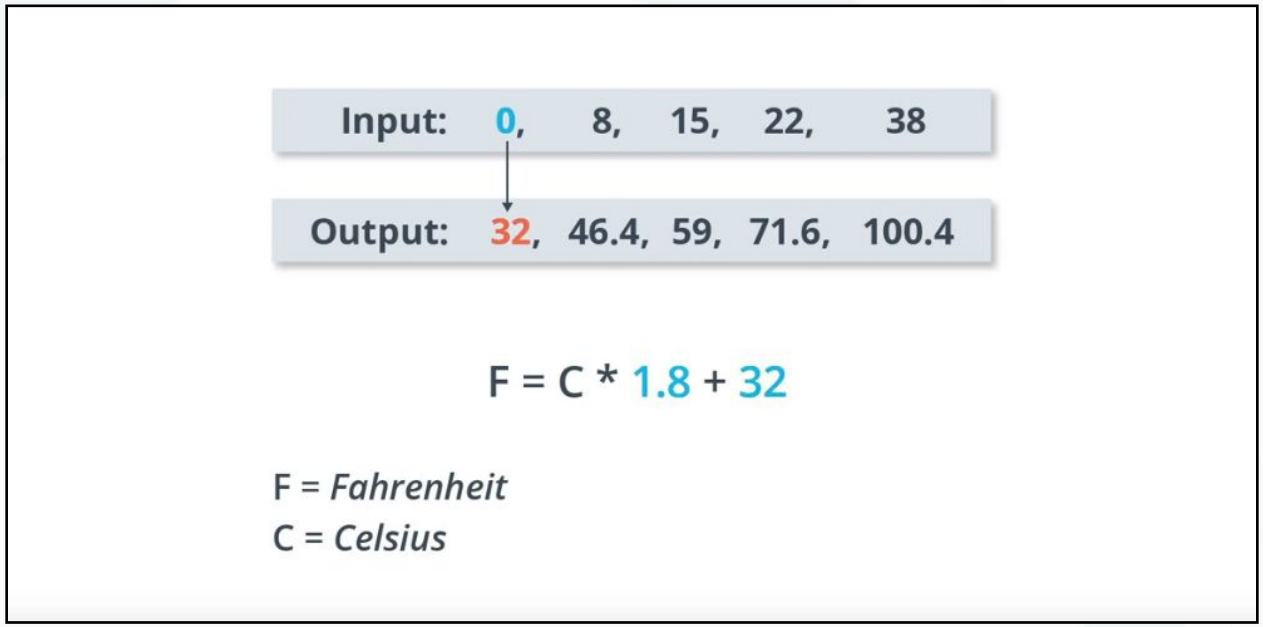
예를 들어, 0의 입력 값은 화씨 32도에 해당하는 섭씨 0 도의 온도를 나타냅니다.

이 공식에서 보는 것처럼 32에 0을 곱하고 32를 더합니다.



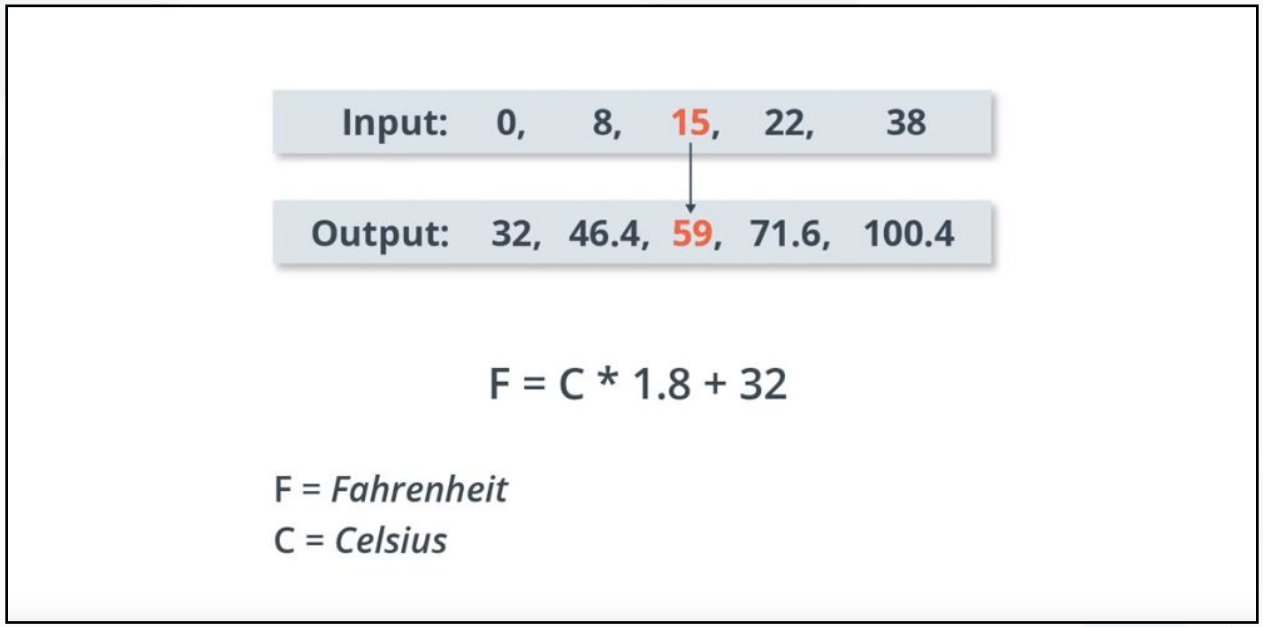
예를 들어, 0의 입력 값은 화씨 32도에 해당하는 섭씨 0 도의 온도를 나타냅니다.

이 공식에서 보는 것처럼 32에 0을 곱하고 32를 더합니다.



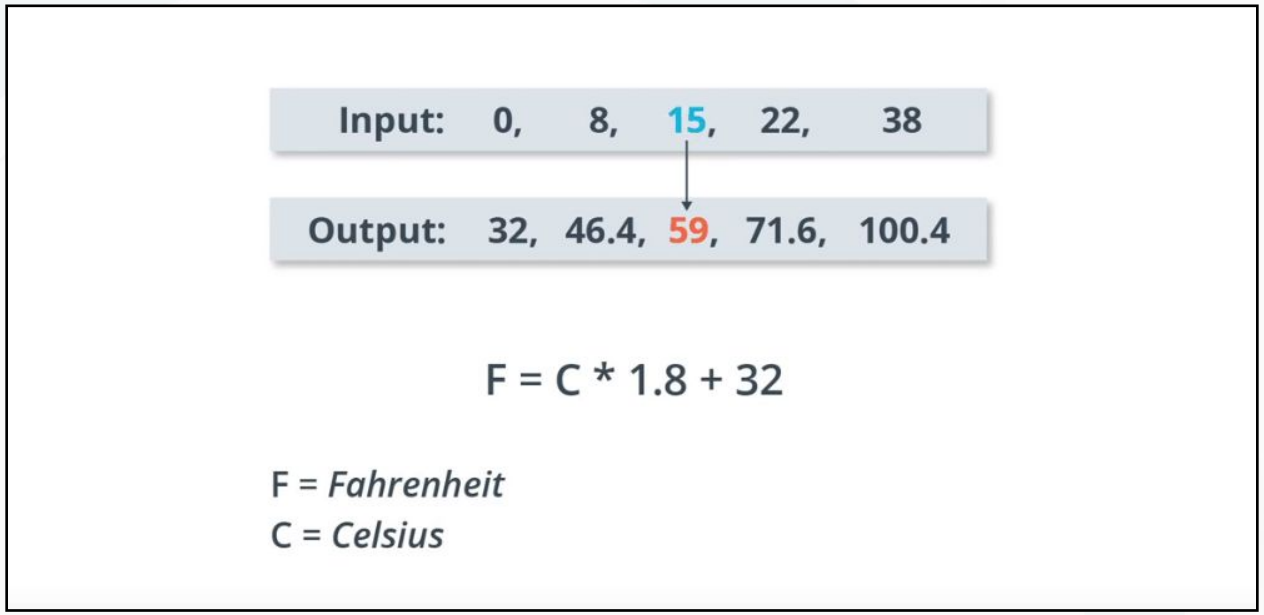
유사하게, 입력 값 15는 15섭씨에 59화씨에 해당합니다.

이 공식에 적용하면 15에 1.8을 곱하고 32를 더하여 59를 얻습니다.



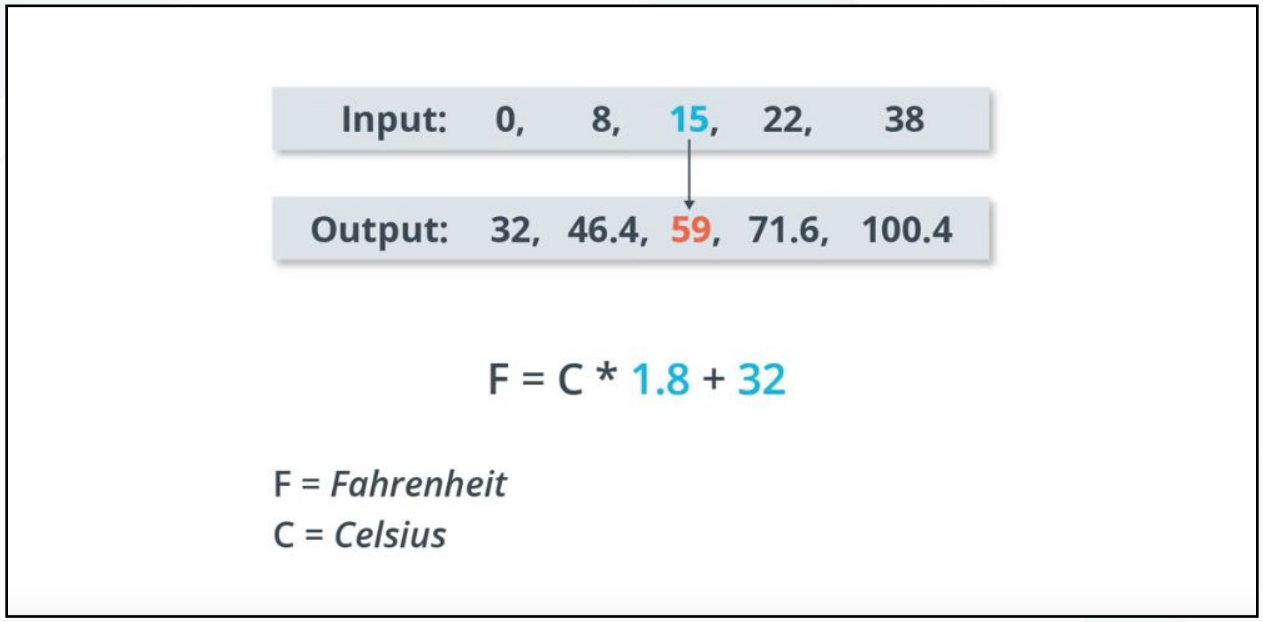
유사하게, 입력 값 15는 15섭씨에 59화씨에 해당합니다.

이 공식에 적용하면 15에 1.8을 곱하고 32를 더하여 59를 얻습니다.



유사하게, 입력 값 15는 15섭씨에 59화씨에 해당합니다.

이 공식에 적용하면 15에 1.8을 곱하고 32를 더하여 59를 얻습니다.



머신러닝이란?

- 기계(computer)가 이 특정한 입력과 출력 사이의 관계를 이해
- 기계(computer)가 올바른 공식(알고리즘)을 파악.

전통 소프트웨어(Traditional Softare) 개발 (Development)

입력값과 알고리즘이 정해진 상태에서 기능/함수(function)을 구현해서 출력값을 만들어 냄.

- 입력값
- 로직(logic)을 적용
- 출력값

머신러닝

입력값과 출력값을 알고 있지만 출력값을 만들어 내는 알고리즘은 알지 못함.

- 입력값과 출력값
- 알고리즘을 배움

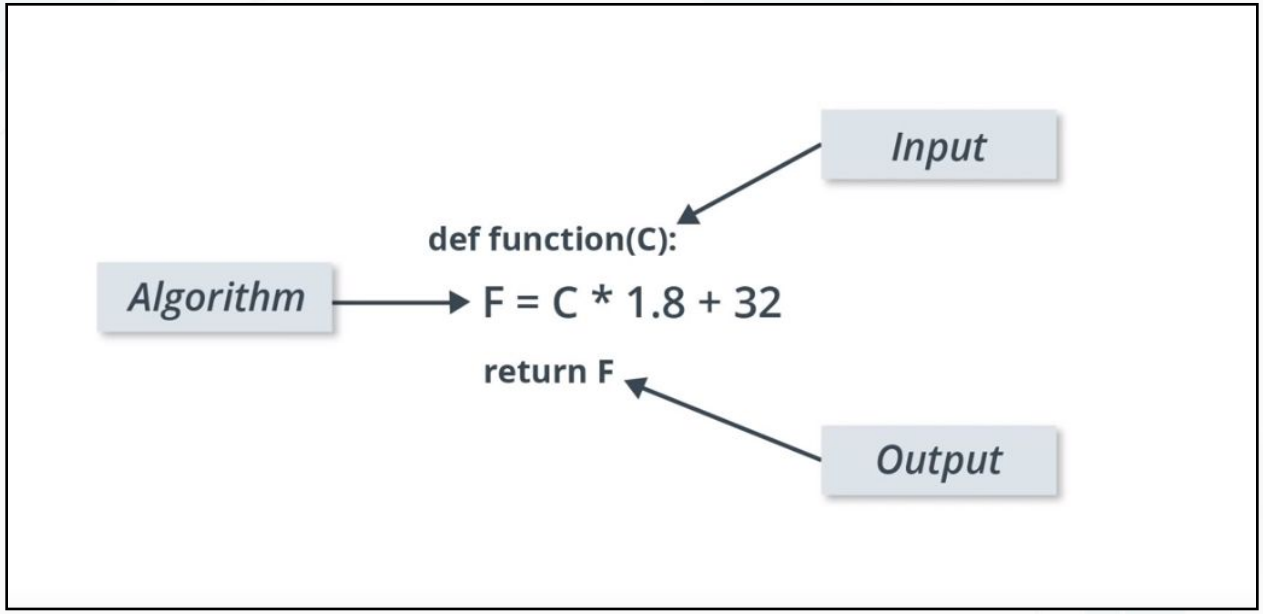
이 공식을 함수로 컴퓨터 프로그램을 작성하려고 한다고 가정해 봅니다.
전통적인 소프트웨어(Traditional Software) 개발에서는 이 관계를 함수를 사용하여 모든 프로그래밍 언어로 쉽게 구현할 수 있습니다.

$$F = C * 1.8 + 32$$

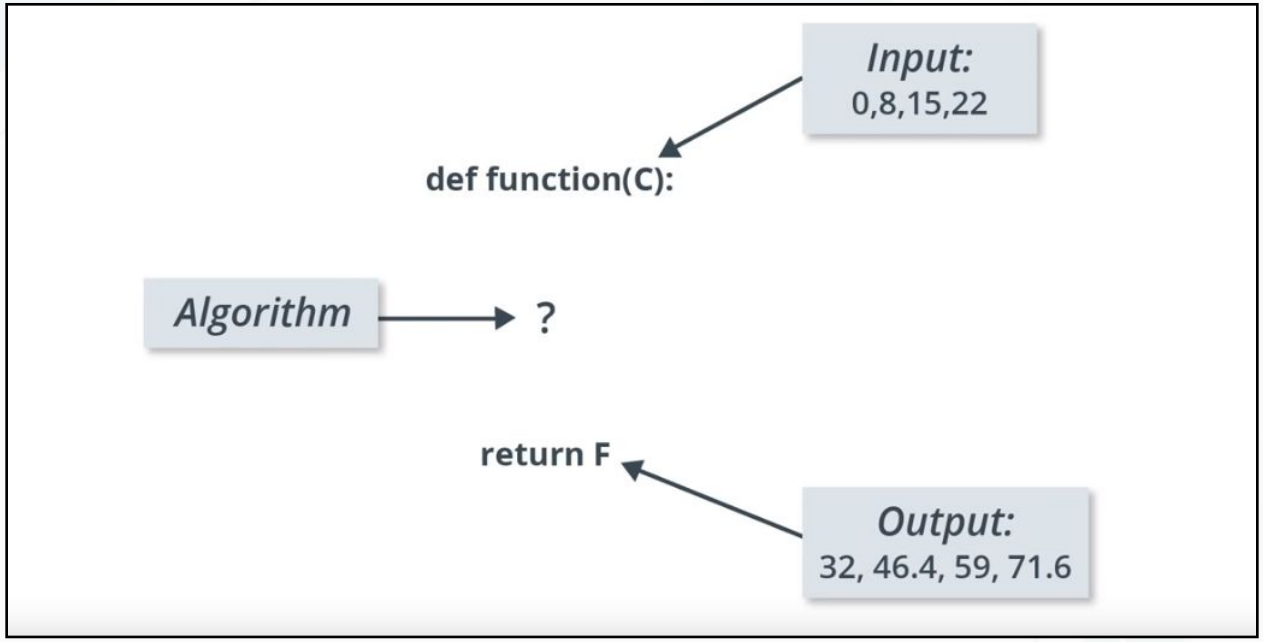
Python을 사용하는 경우.

```
def function(C):
    F = C * 1.8 + 32
    return F
```

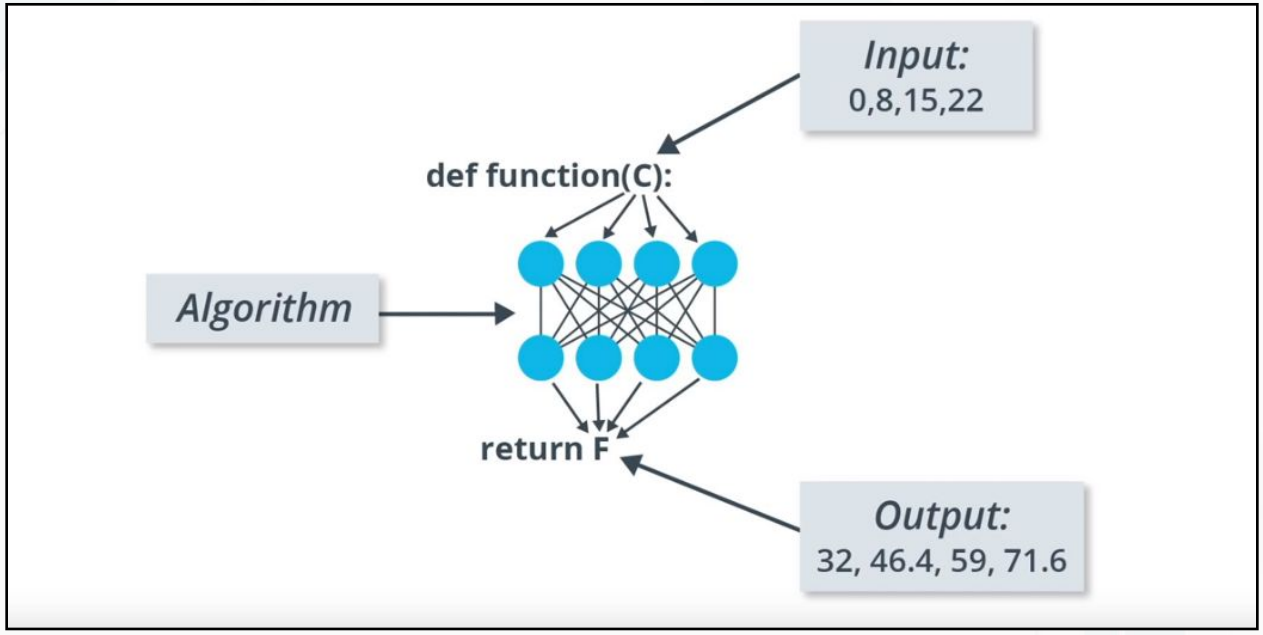
함수는 입력값 C를 받음 -> 알고리즘을 사용하여 출력 값 F를 계산 -> F값을 출력



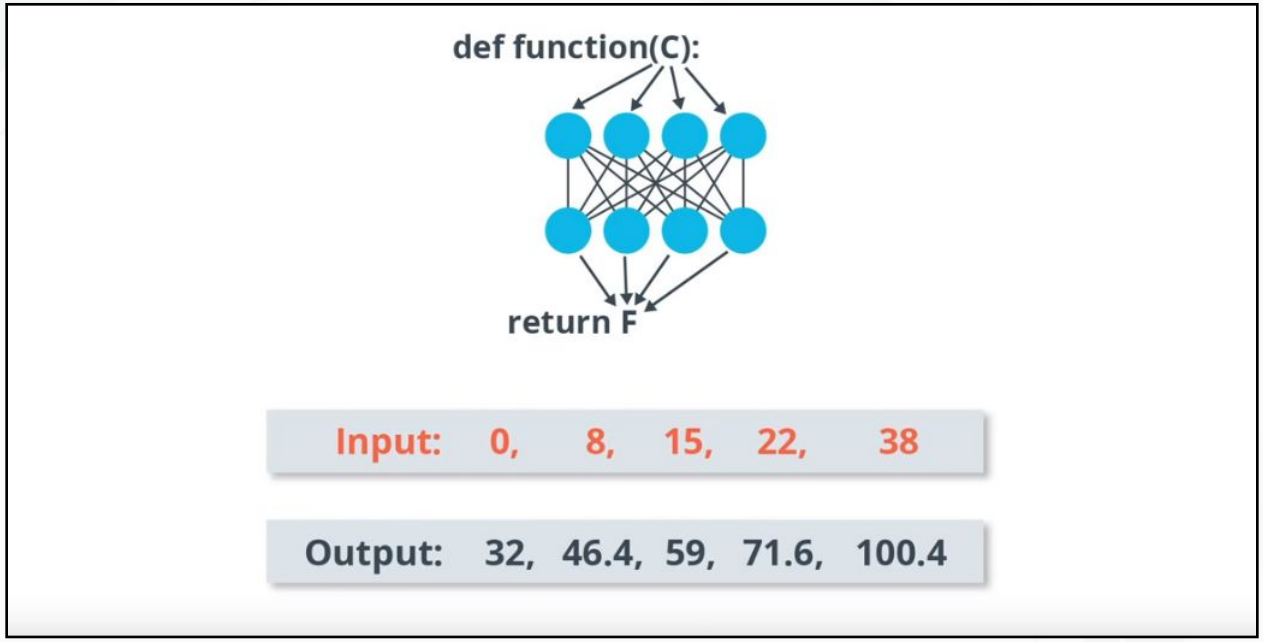
머신러닝은 입력값과 출력값은 있지만 알고리즘이 없습니다.



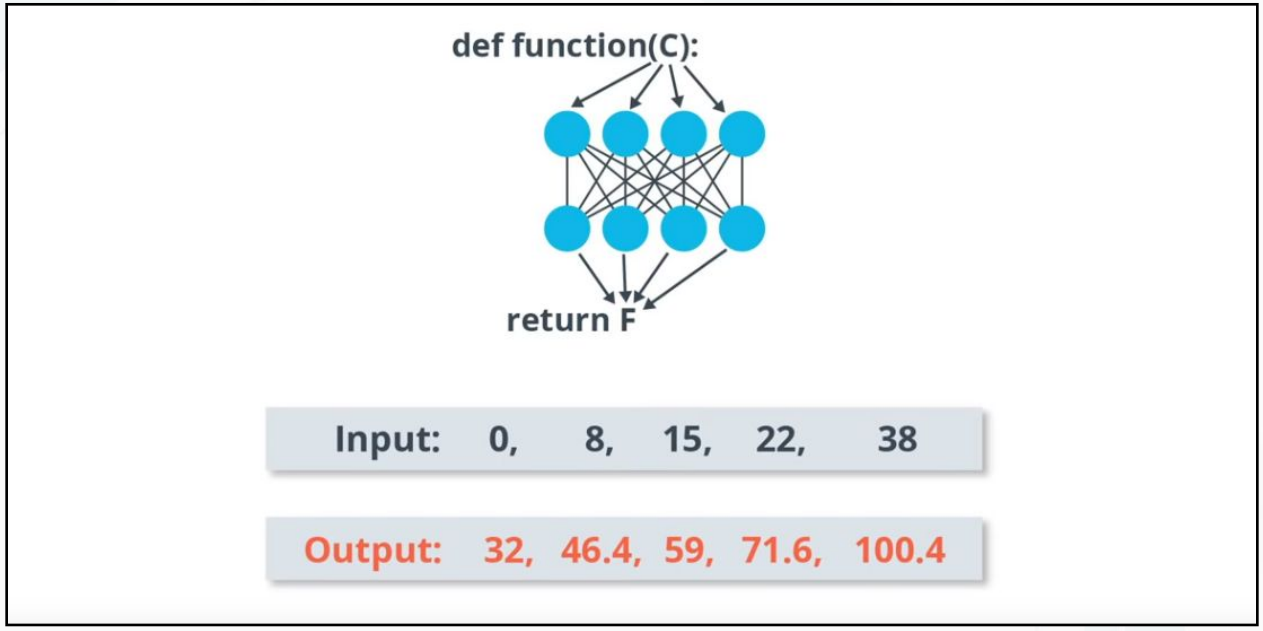
머신러닝은 이러 입력과 출력 간의 관계를 학습하는 신경망(neural network)을 사용함.
신경망은 사전 정의 된 수학(predefined Math) 및 내부 변수(internal variables)로
구성된 계층들(layers)의 스택(stacks)으로 생각할 수 있슴.



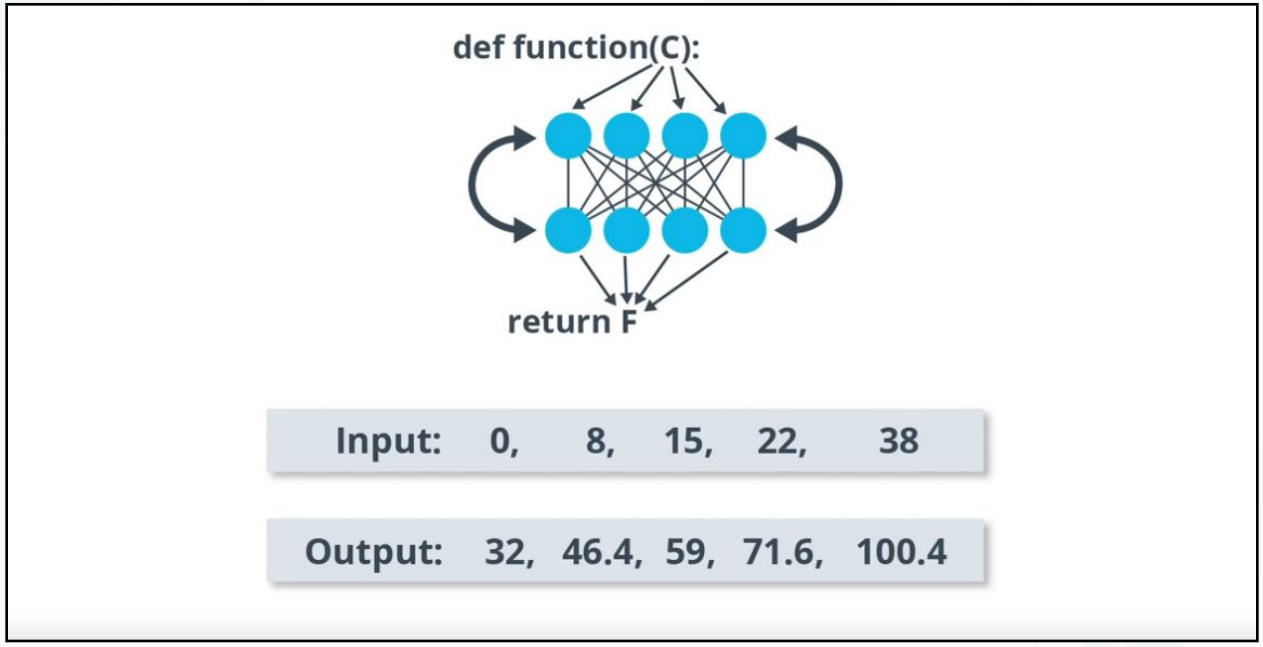
입력 값은 신경망에 공급되고 레이어(layer) 스택(stack)을 따라 이동.



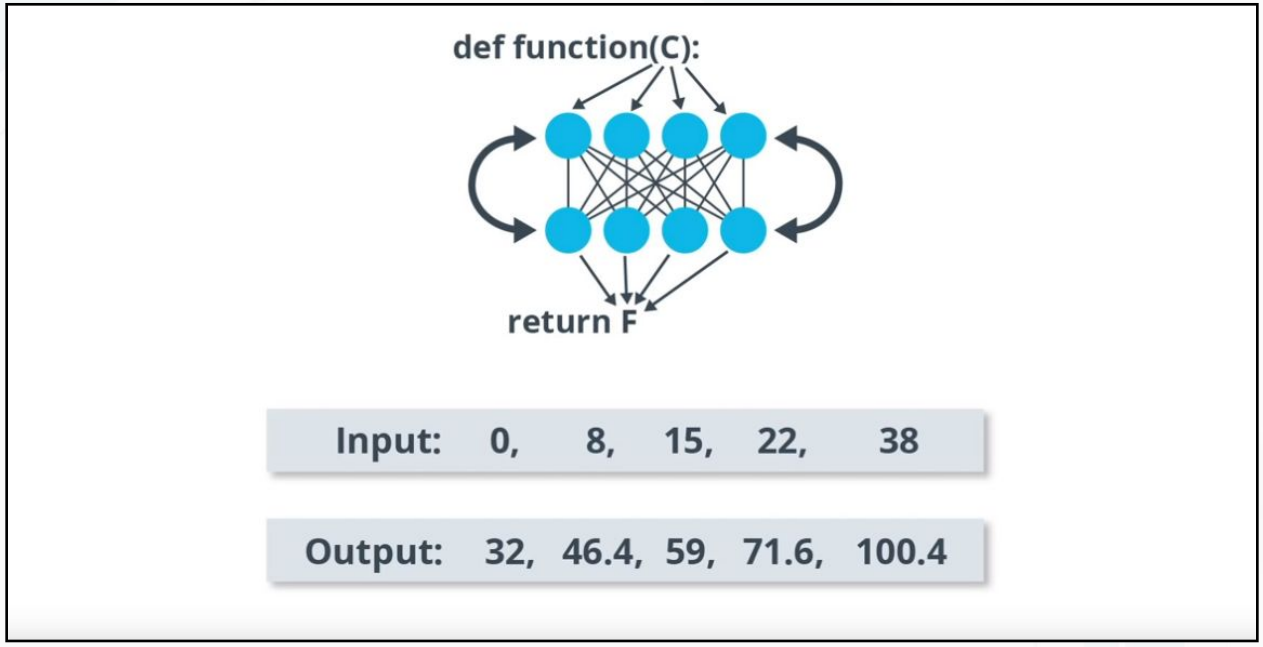
수학(Math)과 내부 변수(internal variables)가 적용되고 결과값 출력이 생성.



학습(training) - 뉴럴 네트워크(neural network)가 입력과 출력 사이의 올바른 관계를 알기 위해서는 훈련시켜야 합니다. 반복적으로 네트워크가 입력을 출력에 매핑(mapping) 시키도록함으로써 신경 네트워크를 훈련시킵니다.

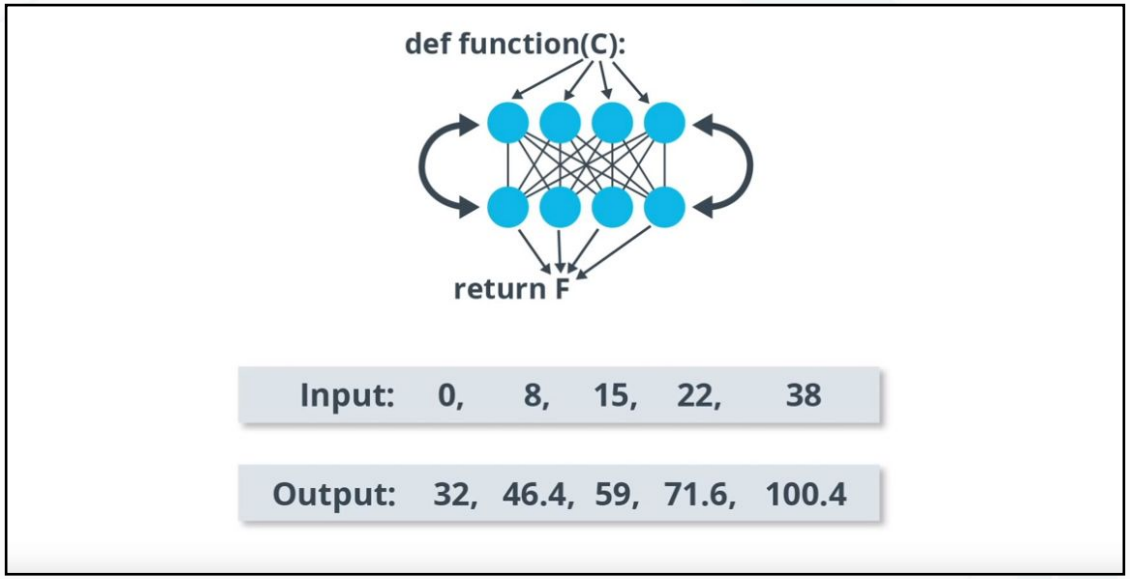


학습(training)을 하는 동안 네트워크(network)가 입력을 받아 출력을 생성 할 때까지 레이어(layer)의 내부 변수(internal variables)를 조정합니다.



네트워크(network)를 조정하여 내부 변수를 조정하는 학습(training) 과정은 **수천** 또는 **수백만** 입력 및 출력 데이터에 대해 반복함.

간단히 정리하면, 머신러닝 알고리즘은 변수를 조정하여 일부 입력을 일부 출력에 올바르게 맵핑(mapping) 할 수있는 함수(function)라 할수 있슴.



03 K-평균 (K-Means)

KOREATECH 한국기술교육대학교

KOREATECH 능력개발교육원

클러스터링(Clustering)

클러스터링(Clustering) 이란?

- 여러 개의 데이터가 있을 때 데이터를 군집화
- 비슷한 데이터끼리 묶으면 관리하기가 쉽다는 장점
- 예) 우리가 학원을 운영한다고 했을 때 학생을 고급반, 중급반, 초급반으로 나누어 적절히 분류
- 이러한 클러스터링 기법은 적용할 사례가 매우 많다는 특징

클러스터링(Clustering)

- K-means 알고리즘으로 대표적인 비지도학습(Unsupervised Learning) 알고리즘
- K-means에서 파생된 알고리즘 및 흡사한 알고리즘은 매우 다양
- 또한 다양한 분야에서 응용될 수 있으므로 제대로 이해하는 것이 중요

K-means 사전 준비

- 클러스터링을 수행할 데이터의 주제를 결정
  - 예) 수학 학원에서 학생들의 성적에 따라 반을 구분하기)
- 얼마나 많은 클러스터를 만들지 고민해야 함
  - 예) 고급반, 중급반, 초급반, ...)
- 데이터를 준비
  - 데이터가 정확할수록 유리
- 클러스터링을 수행하기 위한 방법은 다양
  - 예) 무작위 중심(Centroid) 값 선택, K-means++, ...

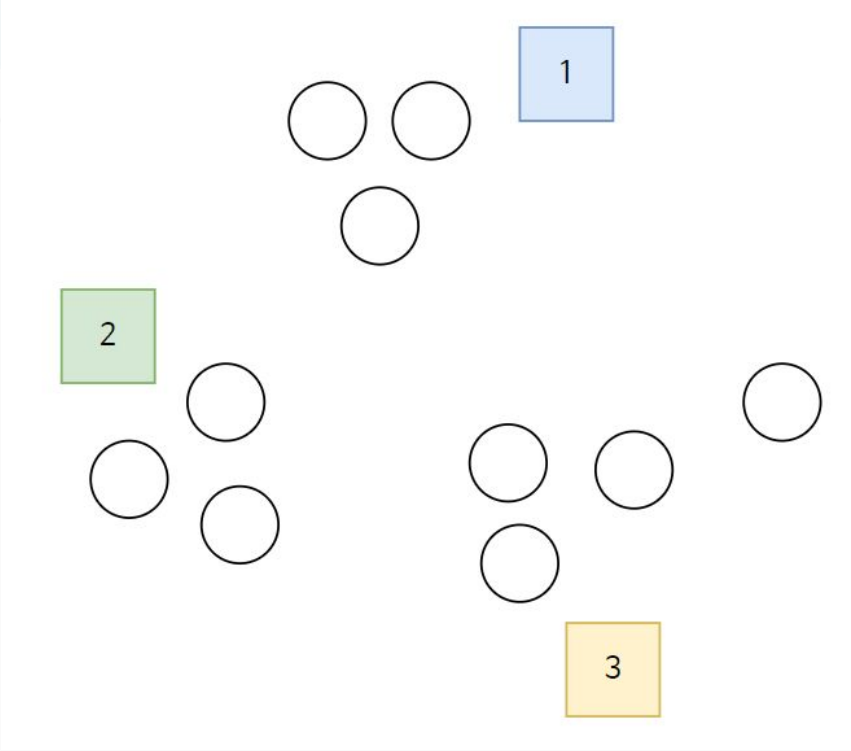
K-means 수행 과정

- 중심(Centroid)에 가까운 데이터를 클러스터에 포함
- 중심(Centroid)을 클러스터의 중앙으로 이동

K-means는 위 두 과정을 반복 수행하면 됩니다. 그러면 결과적으로 완전하게 군집화된 클러스터들을 얻을 수 있습니다. 더이상 중심(Centroid)의 위치가 변하지 않을 때까지 반복하는 것이 일반적입니다.

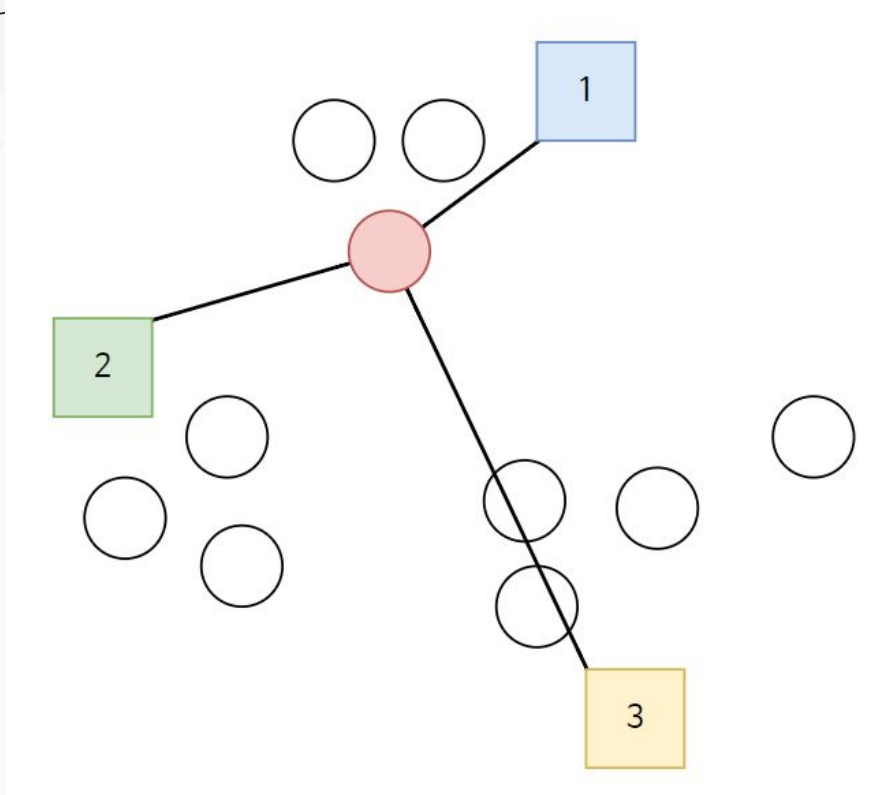
클러스터링 예시 – 무작위 중심(Centroid) 값 선택 알고리즘

데이터가 있을 때 1, 2, 3 세 개의 클러스터가 존재한다고 무작위로 설정을 합니다.



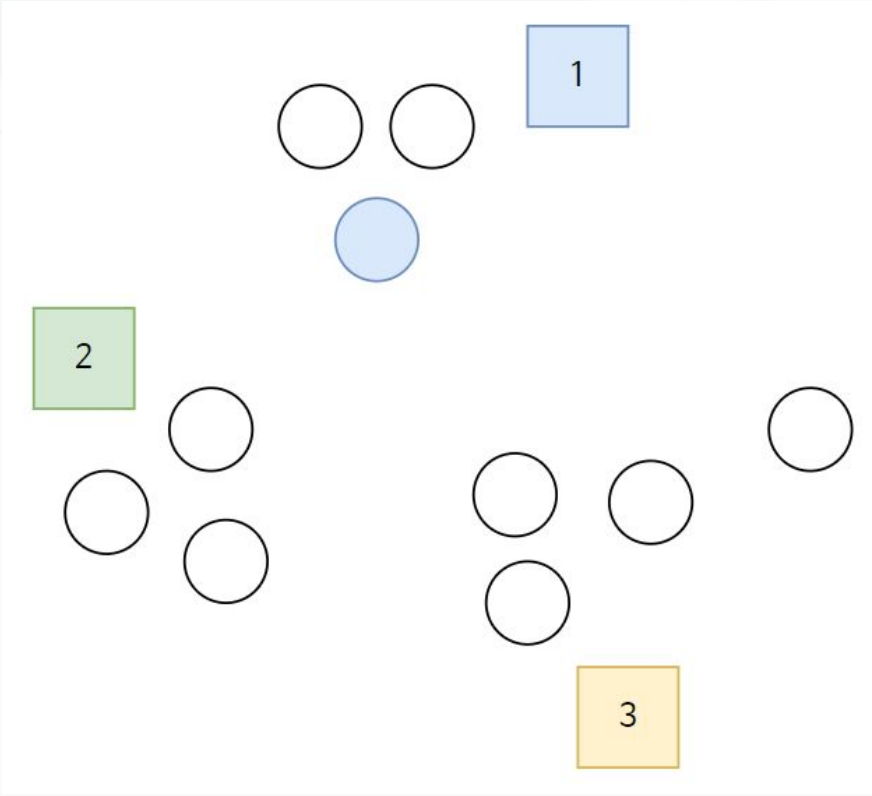
클러스터링 예시 – 무작위 중심(Centroid) 값 선택 알고리즘

하나의 데이터를 선택해봅시다. 이 때 세 개의 클러스터 중에서 무엇에 제일 가깝나요? 바로 1에 가장 가깝습니다.



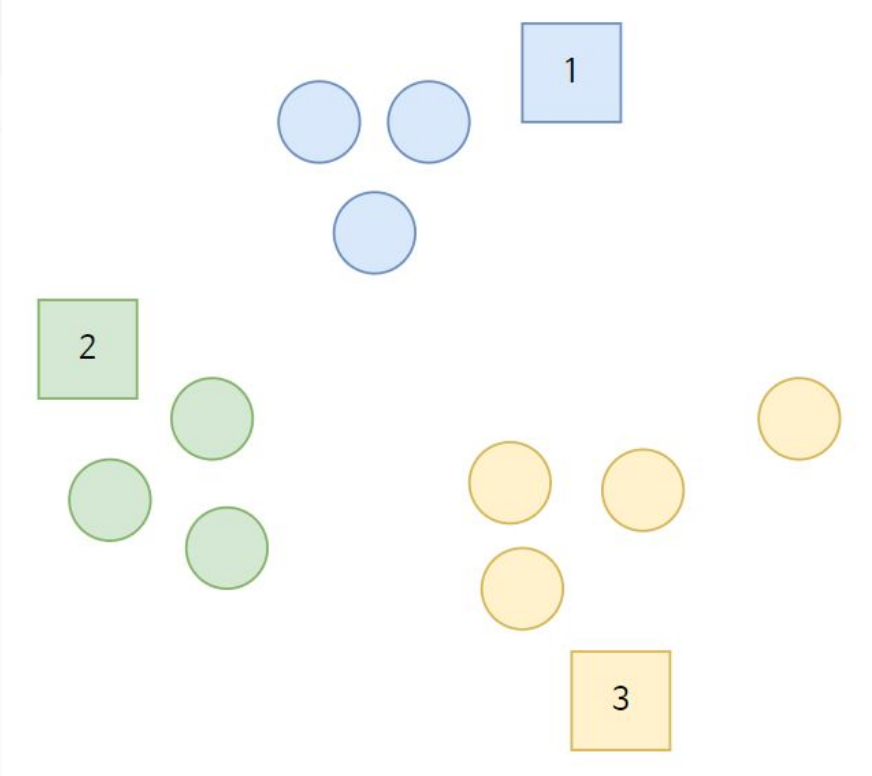
클러스터링 예시 – 무작위 중심(Centroid) 값 선택 알고리즘

그러므로 해당 데이터를 클러스터 1에 속하도록 만듭니다.



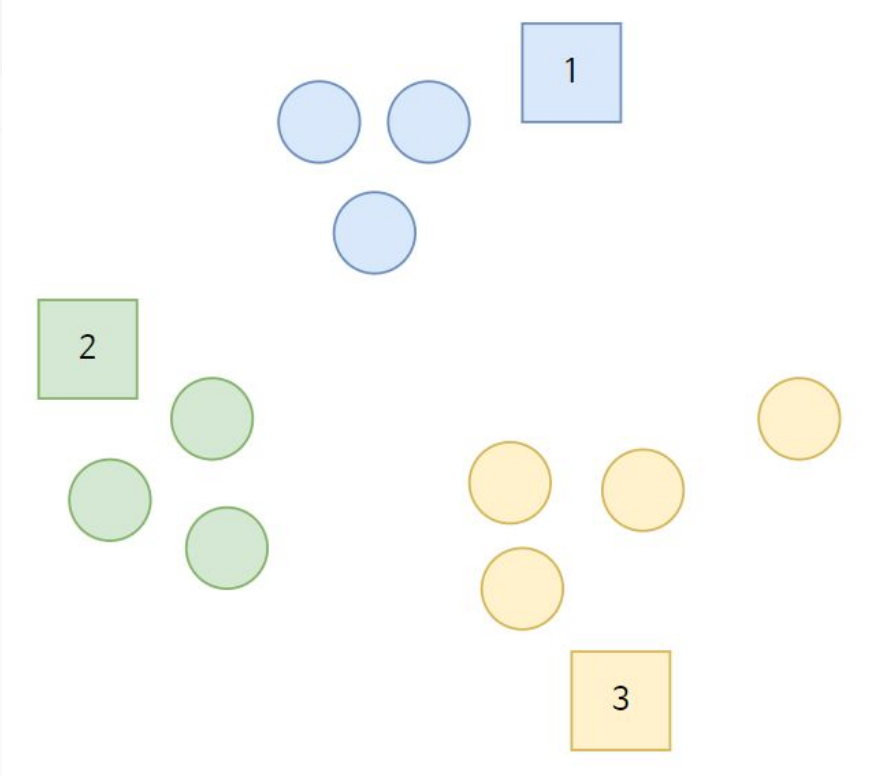
클러스터링 예시 – 무작위 중심(Centroid) 값 선택 알고리즘

이러한 과정을 전체 데이터에 대해 수행하면 다음과 같습니다.



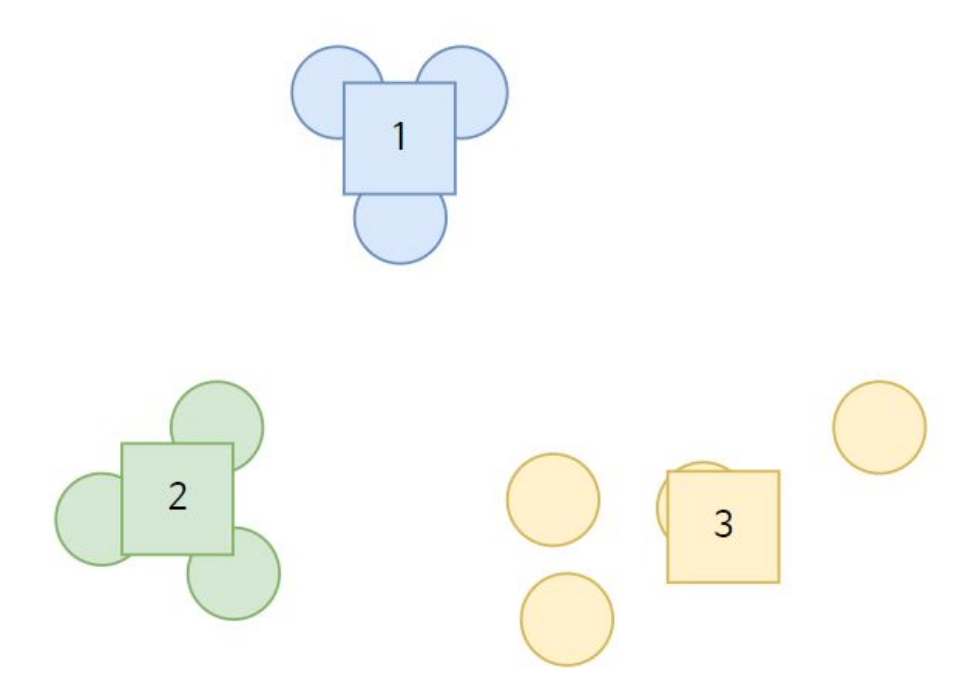
클러스터링 예시 – 무작위 중심(Centroid) 값 선택 알고리즘

대략적으로 군집화가 완료되었습니다.



클러스터링 예시 – 무작위 중심(Centroid) 값 선택 알고리즘

이제 여기에서 더 완벽한 클러스터링을 위해 중심(Centroid)의 위치를 데이터의 중간으로 이동시키게 됩니다. 그 결과는 다음과 같습니다

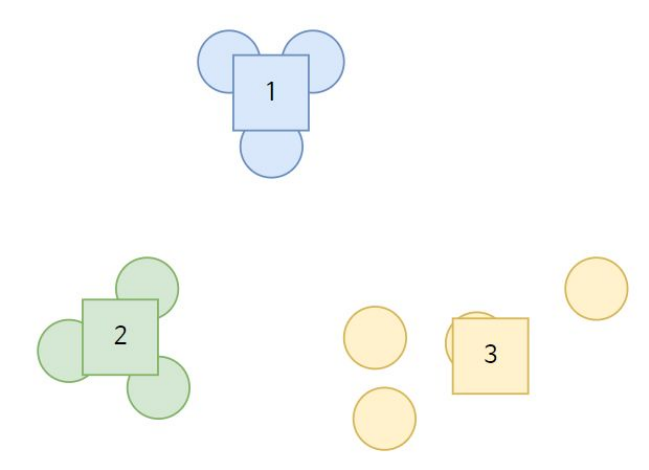


클러스터링 예시 – 무작위 중심(Centroid) 값 선택 알고리즘

이제 이렇게 중심(Centroid)의 위치를 이동시키고 다시 모든 데이터에 대해 어떠한 클러스터 중심(Centroid)에 가까운지 하나씩 체크 합니다.

만약 **특정한 데이터가 다른 클러스터에 더 가깝다면 그 클러스터에 속하도록 설정** 하게 됩니다.

우리가 다룬 위 예시에서는 클러스터 중심(Centroid)의 위치가 더이상 바뀌지 않으므로 여기에서 클러스터링이 끝납니다.



클러스터링 예시 – K-means++

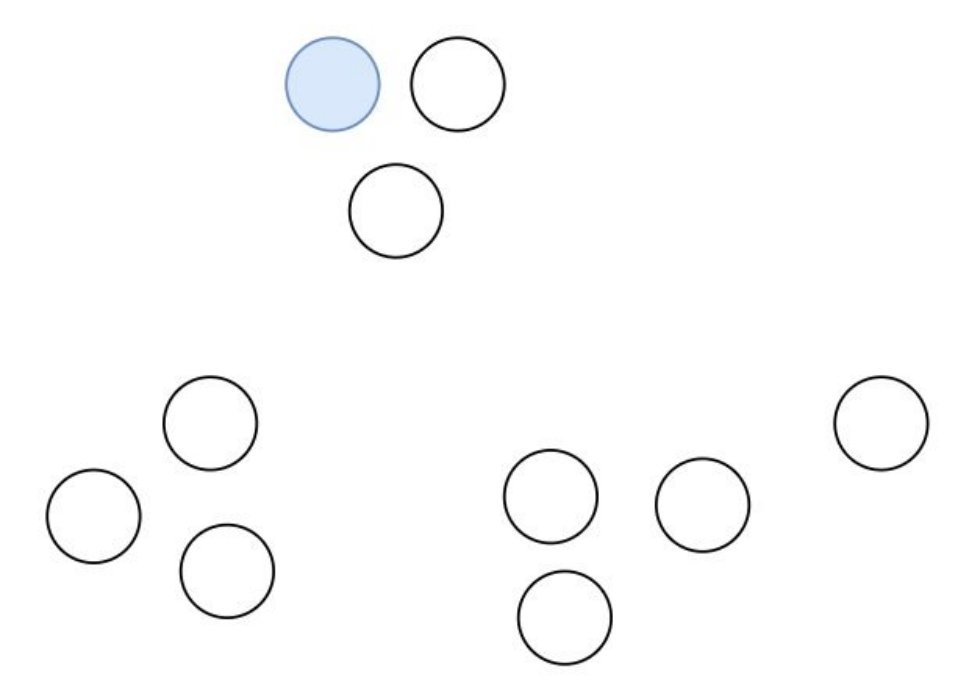
- 초반의 중심 값 설정을 다르게 하는 방법
- K-means 같이 무작위 위치에서 중심 값을 설정할 수도 있지만 직접 수동으로 지정할 수도 있슴

혹은, K-means++

K-means++란 자동으로 적절한 클러스터들의 중심 위치를 찾아주는 알고리즘

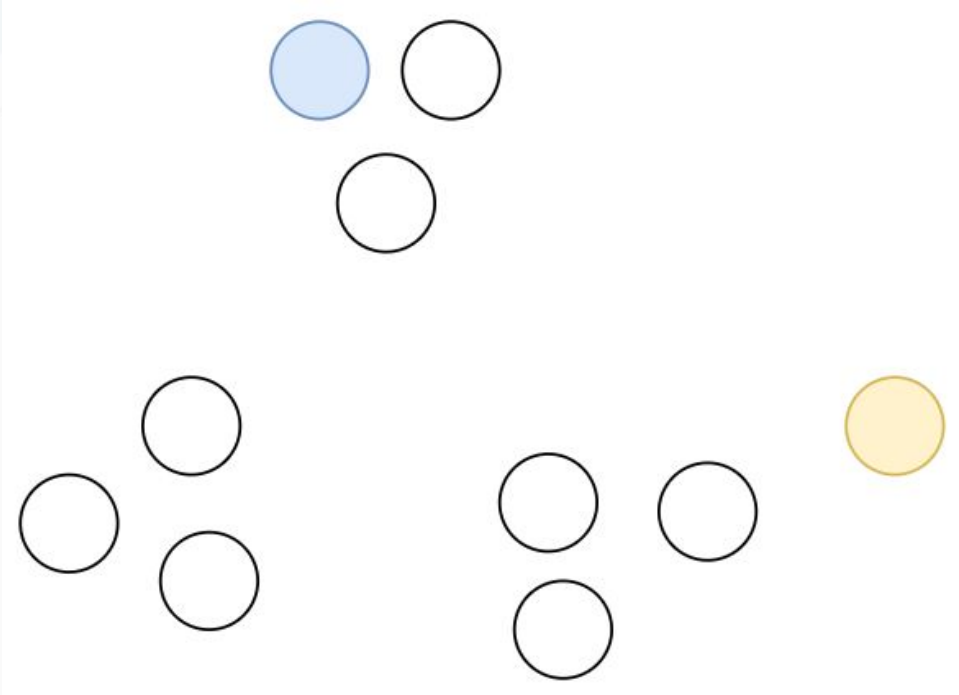
클러스터링 예시 – K-means++

가장 먼저 특정한 노드를 선택하여 클러스터의 중심으로 설정합니다.



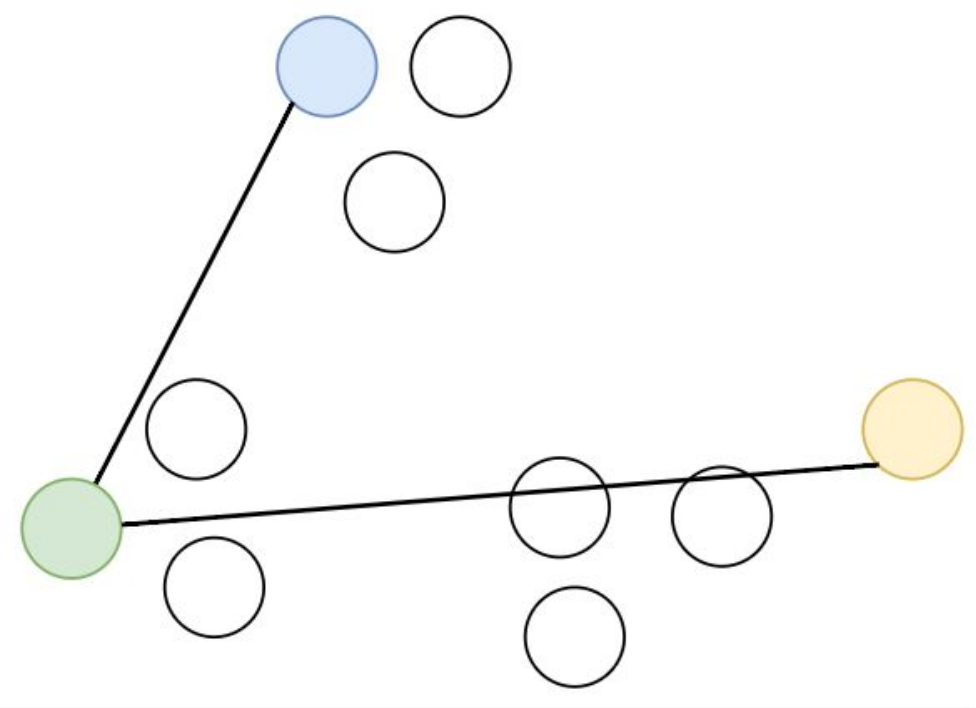
클러스터링 예시 – K-means++

이후에 해당 노드에서 가장 먼 노드를 2번째 클러스터의 중심으로 설정합니다.



클러스터링 예시 – K-means++

그 다음부터는 이미 선택된 중심점들로부터 가장 멀리 있는 노드가 중심이 됩니다.



클러스터링 예시 – K-means++

결과적으로 이렇게 초기 클러스터 중심 데이터들을 설정할 수 있었습니다.

K-means는 이와 같이 클러스터링을 수행하는 대표적인 알고리즘 중 하나입니다.