

머신러닝 기본이론

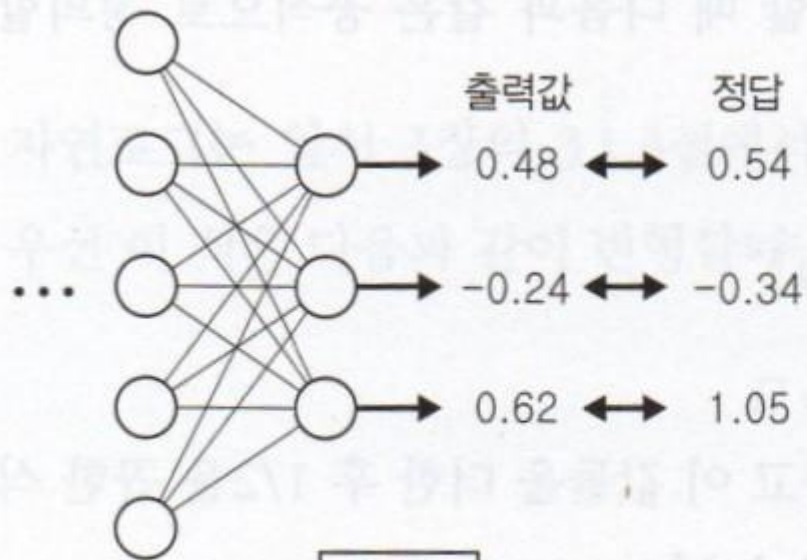
Anaconda 패키지 설치

- 아나콘다 설치페이지 접속 (**individual** 이 무료임)
 - <https://www.anaconda.com/products/individual>
- **Default** 옵션을 선택하며 계속 **Next** ➔ 설치완료
- **Anaconda Prompt** 실행하고 아래 3가지를 각각 입력 및 엔터함.
 - `pip install numpy scipy matplotlib ipython scikit-learn pandas pillow`
 - `pip install tensorflow`
 - `pip install mglearn`
- **Jupyter Notebook**을 실행하고 사용시작!!

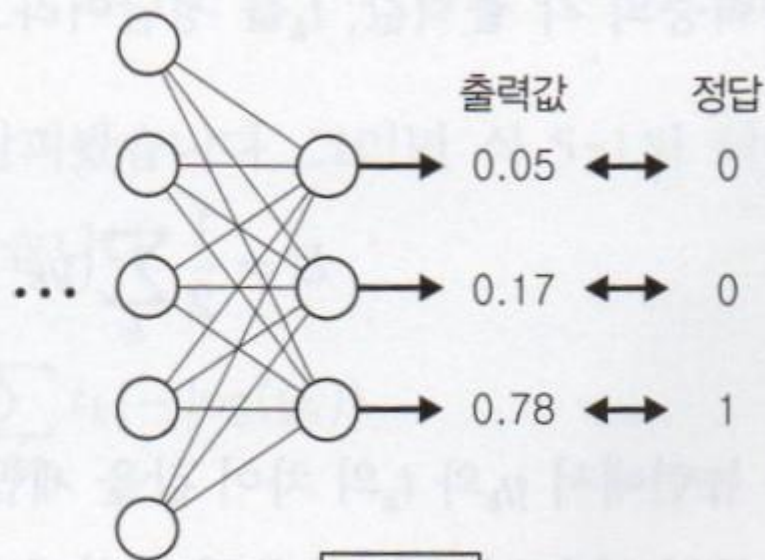
구글 코랩 설치

- 구글 계정으로 로그인 후 코랩 접속
 - <https://colab.research.google.com>
- 단축키
 - 코드셀 실행 후 셀 정지 : **Ctrl+Enter**
 - 코드셀 실행 후 셀 이동 : **Shift+Enter**
 - 코드셀 실행 후 셀 삽입 : **Alt+Enter**
 - 셀 삭제 : **Ctrl+MD** 단축키 열람 및 수정 : **Ctrl+MH**

회귀와 분류



회귀



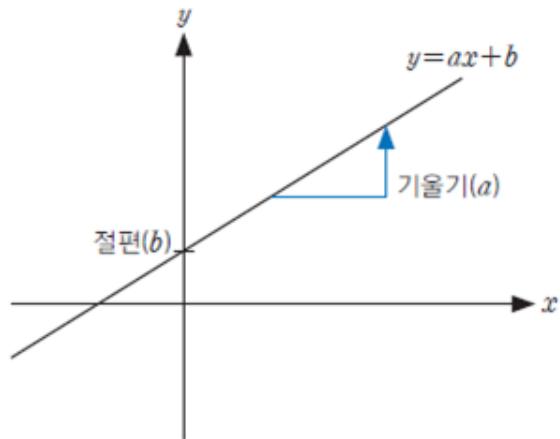
분류

선형회귀

- 회귀
 - 회귀는 변수와 다른 변수간의 관계를 측정한 것
 - 공부 시간과 시험 점수의 관계
 - 출석횟수 및 숙제 제출횟수와 학점의 관계
 - 회귀 분석은 변수 간의 관계를 추정하기 위한 통계적인 방법
 - 회귀는 학습데이터를 이용해 결과값을 예측하는 것을 의미
 - 가장 대중적인 회귀로는 로지스틱 회귀(binary)가 있음

선형회귀

- 선형회귀의 목표
 - 입력데이터(**x**)와 목표데이터(**y**)를 통해 기울기(**a**)와 절편(**b**)을 찾는 것
- 선형회귀의 특징
 - 선형회귀는 아주 간단한 **1차** 함수로 표현 가능



선형회귀와 경사하강법

- 경사하강법

- 모델이 데이터를 잘 표현할 수 있도록 기울기(변화율)를 사용하여 모델을 조금씩 조정하는 최적화 알고리즘
- 어떤 손실 함수(**loss function**)가 정의되었을 때 손실 함수의 값이 최소가 되는 지점을 찾는 방법

- 변화율

- 모델의 가중치와 절편을 효율적으로 업데이트시키기 위한 값

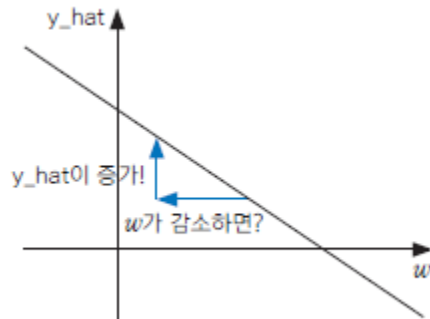
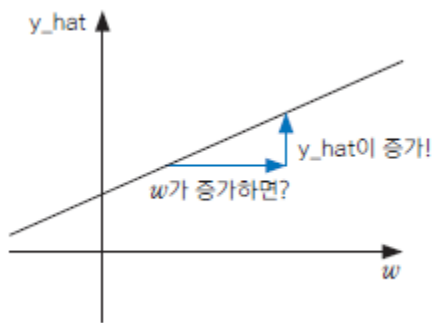
선형회귀와 경사하강법

- 변화율 고찰

- 변화율은 단순한 방법으로 가중치를 업데이트할 수 있게 해 줌
- 변화율이 **0**보다 크거나 작은 경우 모두 가중치에 변화율을 더하기 **(+)**만

하면 OK

- $w_{\text{new}} = w + w_{\text{rate}}$



선형회귀와 경사하강법

- 가중치 및 절편의 변화율
 - **y_hat**: 머신러닝 모델의 예측값
 - 가중치 변화율 = **x** 절편 변화율 = **1**

$$w_{\text{rate}} = \frac{y_{\text{hat}_{\text{inc}}} - y_{\text{hat}}}{w_{\text{inc}} - w} = \frac{(x \cdot w_{\text{inc}} + b) - (x \cdot w + b)}{w_{\text{inc}} - w} = x$$

$$b_{\text{rate}} = \frac{y_{\text{hat}_{\text{inc}}} - y_{\text{hat}}}{b_{\text{inc}} - b} = \frac{(x \cdot w + b_{\text{inc}}) - (x \cdot w + b)}{b_{\text{inc}} - b} = 1$$

선형회귀와 경사하강법

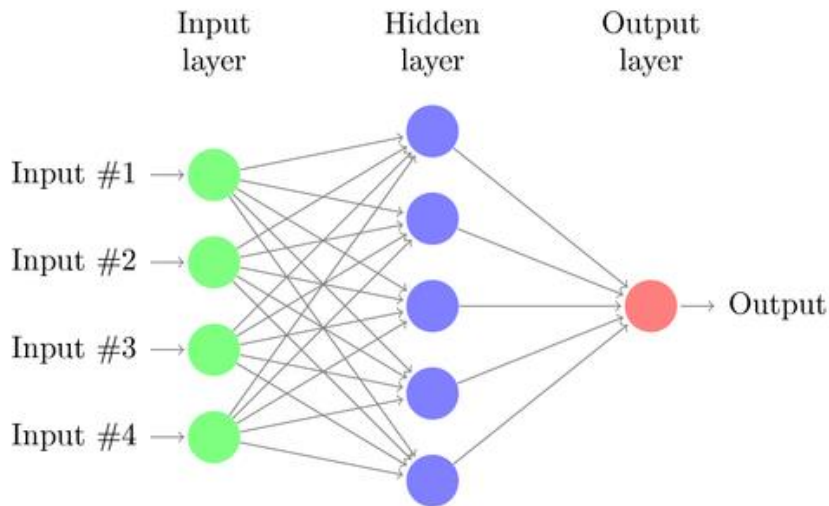
- 손실함수와 경사하강법
 - 경사 하강법에서는 손실 함수를 가중치나 절편에 대해 편미분한 다음, 변화율에서 빼는 방법을 사용함
 - 제곱오차 손실함수를 편미분함

$$SE = (y - \hat{y})^2$$

다층신경망

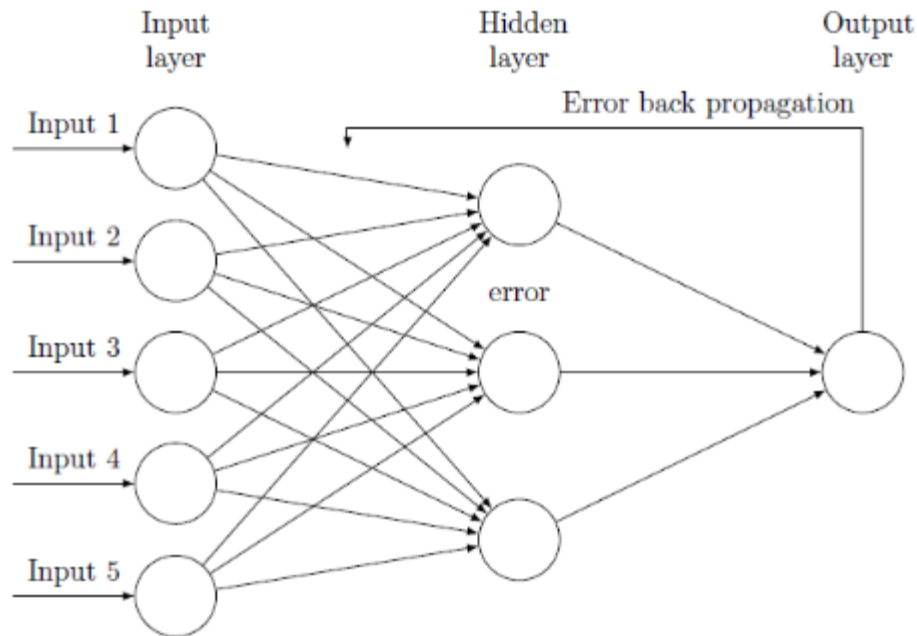
- MLP(Multi Layer Perceptron)

- 입력과 출력 사이에 층이 더 있음
- 개별 **perceptron**의 결과를 다음 층의 입력으로 사용하고 결과적으로 선형 분리의 제약을 극복



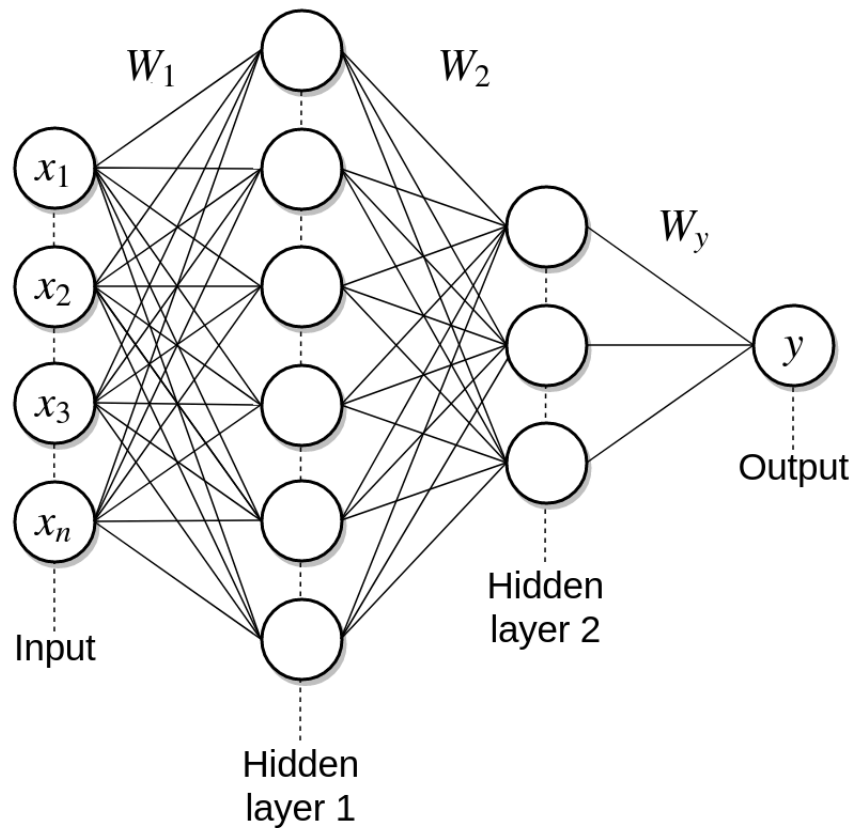
다층신경망

- 2계층 신경망
 - 입력층, 은닉층, 출력층 용어를 사용함



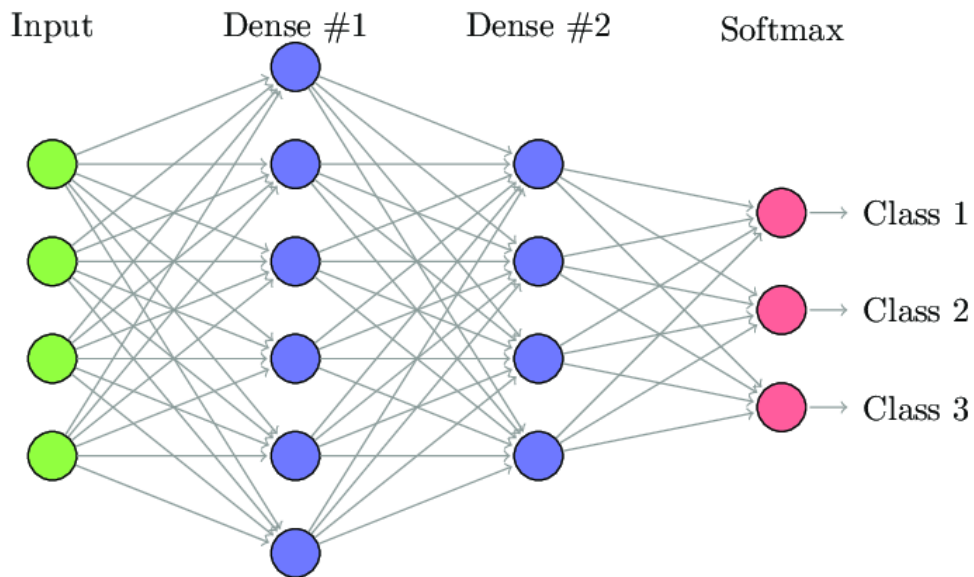
다층신경망

- 완전 연결 신경망
 - **fully connected**
neural network
 - 모든 뉴런이 연결된
네트워크



다중분류

- 다중분류 신경망의 구조



다중분류

- 원핫인코딩 변환
 - (정의) 단 하나의 값만 **True**이고 나머지는 모두 **False**인 인코딩.
즉, **1**개만 **Hot(True)**이고 나머지는 **Cold(False)**임.
 - (특징) 보통 디코딩할 때는 **numpy**의 **argmax**를 사용함.
저장공간 측면에서는 매우 비효율적임.

소프트맥스함수

- 정의
 - 입력받은 값을 출력으로 **0~1**사이의 값으로 모두 정규화
 - 출력 값들의 총합은 항상 **1**이 되는 특성을 가진 함수

$$\frac{e^{z_i}}{e^{z_1} + e^{z_2} + e^{z_3}}$$

소프트맥스함수

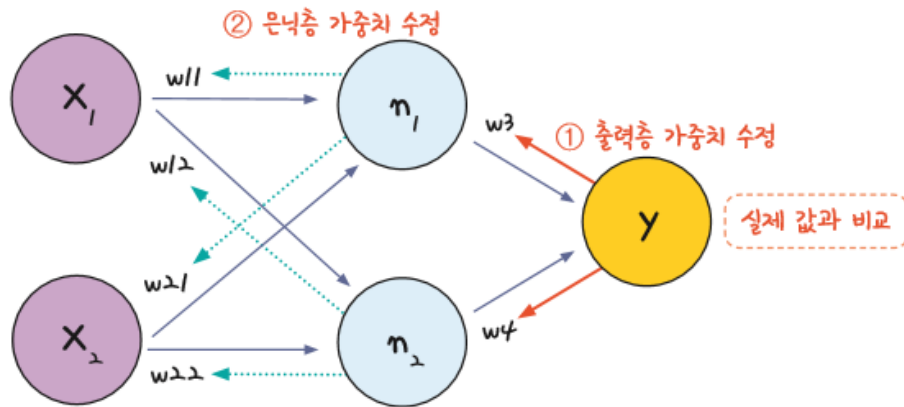
- 특징
 - 로지스틱 회귀는 두 가지로만 분류가 가능하지만, 소프트맥스 회귀는 n 개의 분류로 구분이 가능함
 - 출력의 총합이 **1**이므로 출력을 확률로 해석할 수 있어 문제를 확률적으로 대응할 수 있음
 - 소프트맥스 함수를 적용해도 각 원소의 대소관계는 변하지 않음 (지수함수 관계임)

오류역전파 (Backpropagation)

- 개념
 - 신경망을 학습시킬 때 이용하는 알고리즘
 - 출력값과 정답의 오차를 네트워크에서 역전파시켜 네트워크의 가중치와 편향을 최적화시킴
 - 모든 딥러닝에서 필수불가결한 가장 중요한 알고리즘임

오류역전파 (Backpropagation)

- 다층 퍼셉트론 결과값의 오차를 구해 이를 토대로 하나 앞선 가중치를 차례로 거슬러 올라가며 조정해 감



새 가중치는 현 가중치에서 '가중치에 대한 기울기'를 뺀 값

$$W(t+1) = W_t - \frac{\partial \text{오차}}{\partial W}$$

오류역전파 (Backpropagation)

- 최적화의 계산 방향이 출력층에서 시작해 앞으로 진행됨
- 오류 역전파 구동 방식
 - 1) 임의의 초기 가중치를 준 뒤 결과를 계산한다
 - 2) 계산 결과와 우리가 원하는 값 사이의 오차를 구한다
 - 3) 경사 하강법을 이용해 바로 앞 가중치를 오차가 작아지도록 업데이트
 - 4) 1~3 과정을 더 이상 오차가 줄어들지 않을 때까지 반복한다

오류역전파 (Backpropagation)

- ‘오차가 작아지는 방향으로 업데이트한다’ 의미
 - 미분 값이 **0**에 가까워지는 방향으로 나아간다는 뜻
- ‘기울기가 **0**이 되는 방향으로 진행’ 의미
 - 가중치에서 기울기를 뺏을 때 가중치의 변화가 전혀 없는 상태를 뜻함
- 따라서 오차 역전파를 다른 방식으로 표현하면
 - 가중치에서 기울기를 빼도 값의 변화가 없을 때까지 계속해서 가중치 수정 작업을 반복하는 것

손실함수 (Loss Function)

- 손실함수 정의
 - 예측값(출력값)과 실제값(정답)과의 오차를 정의하는 함수
 - 손실함수 = 오차함수 = 비용함수
- 손실함수 목적
 - $h(x)-y$, 즉 잔차(residue)가 작으면 주어진 데이터를 잘 대표하는 모형임. 즉 이 차이를 가능한 한 **0**에 가깝게 만들어야 함.
 - 학습데이터 개수가 여러 개일 경우, 잔차합의 평균을 **0**에 가깝게 만들

손실함수 (Loss Function)

- MSE (Mean Squared Error, 평균제곱오차)

- 오차(예측과 실제의 차이)를 모두 제곱한 후 평균을 낸 것.

$$MSE = \frac{1}{N} \sum_i (y_i - \hat{y}_i)^2$$

- 분산과 유사함 → $Var(y) = \frac{1}{N} \sum_i (y_i - \bar{y})^2$

- 장점

- 지표가 직관적이고 단순함.

손실함수 (Loss Function)

- MSE (Mean Squared Error, 평균제곱오차)

- 단점

- 오차가 커질수록 손실이 제곱으로 커짐 (예외적 데이터에 약함)
 - 스케일에 의존적임

(예) 100만원, 10만원 주가의 MSE 같을 때, 과연 오차율이 동일한가?

- 오차를 제곱하므로 1 미만의 오차는 작아지고, 그 이상은 커짐
 - 제곱을 하므로 오차가 양수인지 음수인지 방향성을 상실

손실함수 (Loss Function)

- MAE (Mean Absolute Error, 평균절대오차)

- 오차(예측과 실제의 차이)의 절대값의 평균을 낸 것.

$$MAE = \frac{1}{N} \sum_i |y_i - \hat{y}_i|$$

- 장점
 - 예외적 데이터에 상대적으로 강인함.
 - 지표가 직관적이고 단순함.
- 단점 : 스케일에 의존적이고 오차의 방향성 상실.

손실함수 (Loss Function)

- 최소제곱법 필요성
 - 아래의 3가지 방법이 존재함 (**ME** → **MAE** → **MSE**)
 - $h_{\theta}(x) - y$ 을 최소화하는 방법
 - $|h_{\theta}(x) - y|$ 를 최소화하는 방법
 - $(h_{\theta}(x) - y)^2$ 를 최소화하는 방법 ✓
 - 첫 번째 방법(**ME**)은 양수 잔차와 음수 잔차가 상쇄되므로 사용불가
 - 잔차의 정규성을 고려하여 제곱합을 최소화하는 세번째 **MSE**를 채택

손실함수 (Loss Function)

- 최소제곱법 공식

- 잔차 제곱합의 평균을 최소화하는 것이 손실함수의 목적

$$\sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$


- 잔차 제곱합의 평균을 구하되, 경사하강법 미분 적용 시 발생하는 **2**를 상쇄하기 위해, **2m**으로 나누는 방법을 선택함

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

손실함수 (Loss Function)

- 머신러닝에서 사용하는 손실함수 2가지
 - 오차제곱합 (Sum of Squares for Error)



$$E = \frac{1}{2} \sum_k (y_k - t_k)^2$$

- 미분을 쉽게 하기 위해서 앞에 $\frac{1}{2}$ 을 붙임
- y_k : 예측값(출력값) t_k : 실제값(정답)
- 오차제곱합은 연속적 수치에 잘 맞음  회귀 문제에서 자주 사용됨

손실함수 (Loss Function)

- 머신러닝에서 사용하는 손실함수 2가지
 - 교차엔트로피 오차 (Cross Entropy Error)

$$E = \sum_k t_k (-\log y_k)$$

- 두 분포 간의 차이를 나타내는 척도  분류 문제에서 자주 사용됨
- 분류문제에서 정답 1은 하나이고 나머지는 모두 0 (원핫인코딩)
 우변 시그마 내부에서 $t_k=1$ 인 항만 오차 **E**에 영향을 주고,
나머지는 **0**라서 의미 없음

손실함수 (Loss Function)

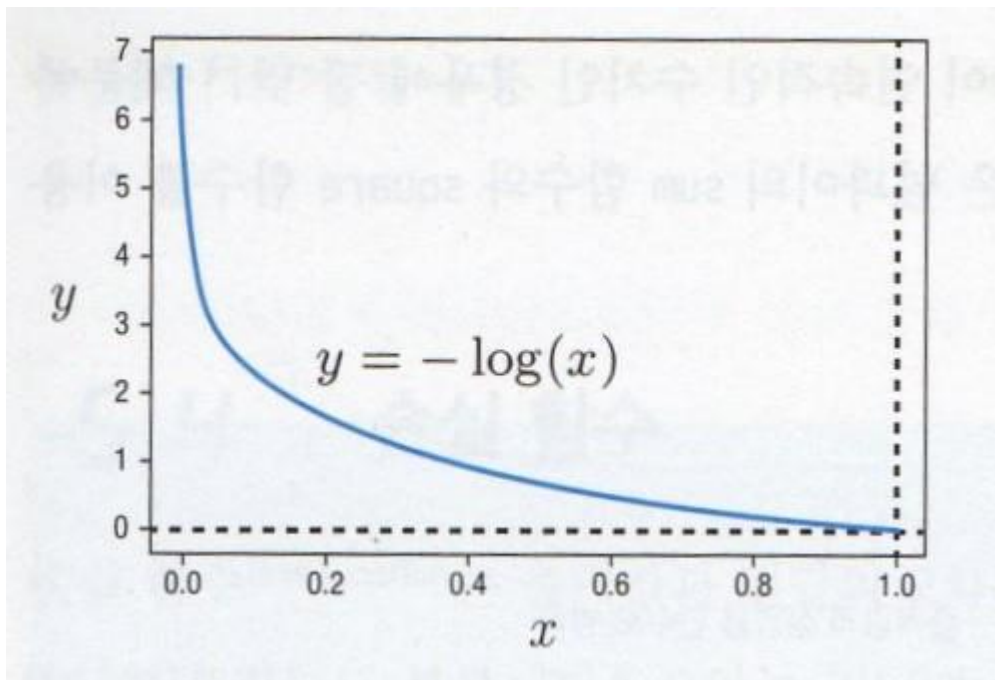
- 머신러닝에서 사용하는 손실함수 2가지
 - 교차엔트로피 오차 (Cross Entropy Error)

$$E = \sum_k t_k (-\log y_k)$$

- 정답 $t_k = 1$ 일 때, y_k 가 정답 1에 근접할수록 오차 E는 0에 가까워짐
 - 👉 $-\log y_k$ 는 $y_k = 1$ 일때 0이고, y_k 가 0에 근접할수록 무한대로 커짐
- 출력값과 정답의 차가 크면 오차는 무한대로 커짐 👉 학습속도 빠름

손실함수 (Loss Function)

- 교차엔트로피 오차 (Cross Entropy Error)



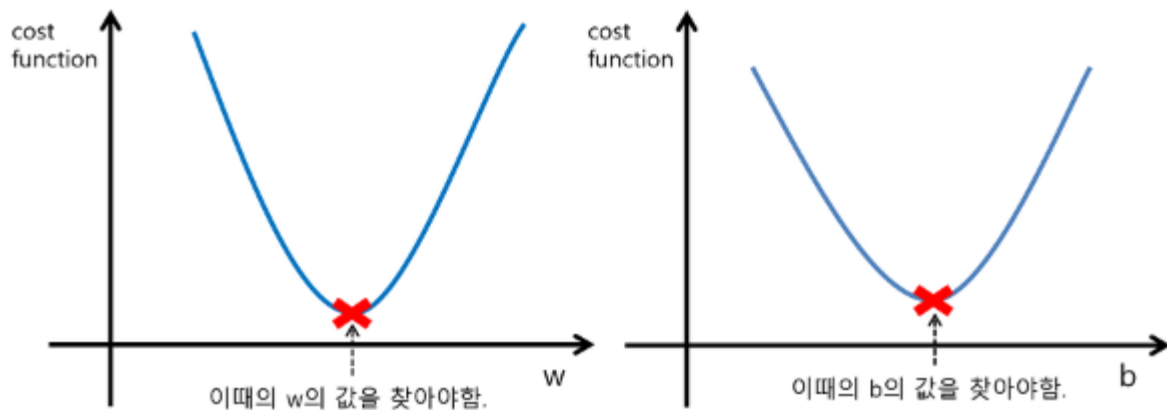
경사하강법

[1] 평균제곱오차 = 손실함수 (Loss Function)

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

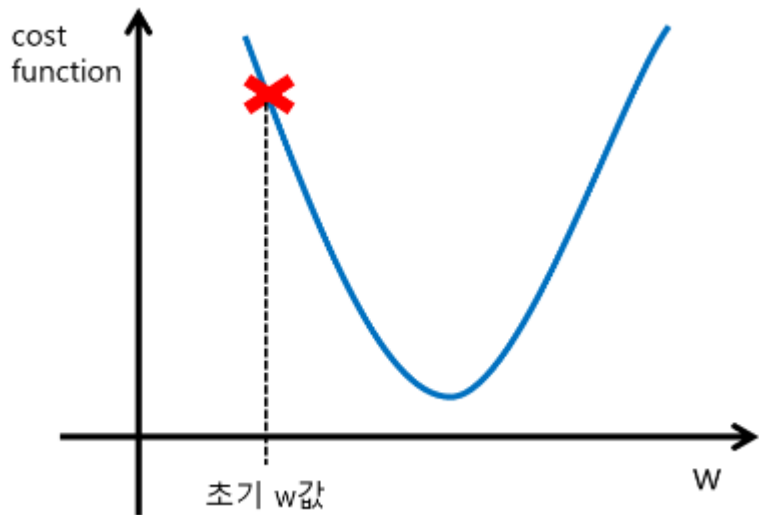
...(공식1: 평균제곱오차, 비용함수)

[2] 손실함수값이 최소가 되는 w와 b를 찾아야 함



경사하강법

[3] 가중치 w 업데이트 방법



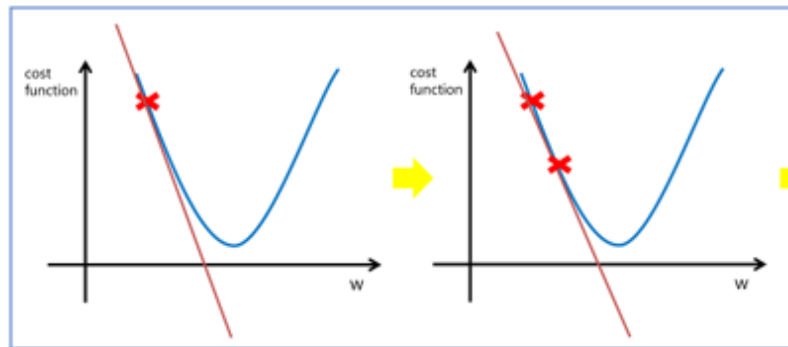
최적의 w 값을 찾아가기 위해서 손실함수를 w 에 대해서 편미분한 것에 학습률 (learning rate)로 불리는 파라미터 알파를 곱한 것을 초기 설정된 w 값에서 빼준다.

[4] 가중치 w 업데이트 공식

$$w := w - \alpha \frac{\partial}{\partial w} MSE$$

...(공식2: w 값 업데이트하기)

경사하강법



- ✓ 바로 이렇게 점차 접선의 기울기가 감소하는 것이 경사하강법임
- ✓ 접선의 기울기가 0이 될 때가 최적의 w 값이 됨



경사하강법

- 미니 배치 경사 하강법 (mini-batch gradient descent)

[정의]

- 배치 경사하강법처럼 에포크마다 전체 데이터를 사용하는 것이 아님.
- 조금씩 나누어서 정방향계산 → 그레이디언트계산 → 가중치업데이트를 수행함.

[고려사항]

- 미니배치의 크기 : 보통 16, 32, 64 등 2의 배수를 사용함.
- 만약 미니배치의 크기가,
10이면 → 확률적 경사하강법
전체데이터 크기 → 배치 경사하강법

경사하강법

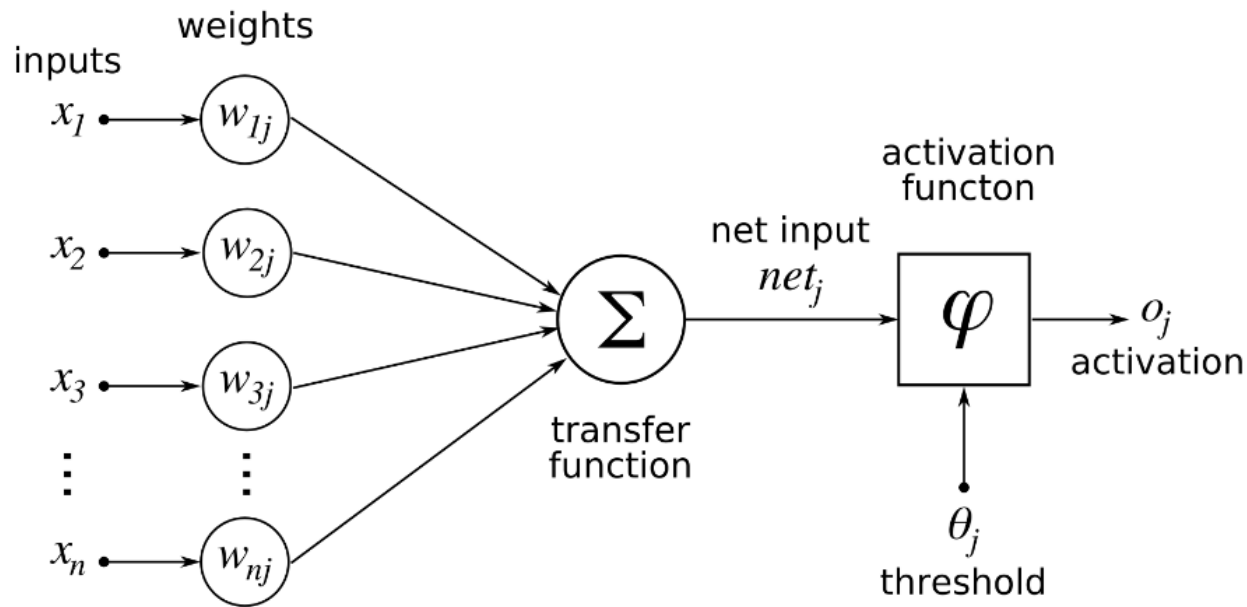
- 확률적 경사 하강법 **vs** (미니) 배치 경사 하강법
 - 확률적 경사 하강법
 - : 계산 비용 낮음, 가중치가 최적값에 수렴하는 과정이 불안정
 - (미니) 배치 경사 하강법
 - : 계산 비용 높음, 가중치가 최적값에 수렴하는 과정이 안정적임

활성화함수

- 활성화함수 정의
 - 입력신호의 총합을 출력신호로 변환하는 함수
 - 입력받은 신호를 얼마나 출력할 지를 결정함
- 활성화함수 종류
 - 계단함수 (Step function)
 - 시그모이드함수 (Sigmoid function)
 - ReLU 함수

활성화함수

- 활성화함수 구성도



활성화 함수

- 활성화 함수 비선형성
 - 활성화 함수는 비선형 함수를 사용해야 함
 - 아래 식에서 선형 함수 **a**와 동일한 구조의 식이 나오므로 활성화 함수의 의미 없어짐

The diagram illustrates the derivation of a linear function from a linear combination of inputs. It consists of three equations arranged vertically, connected by red curved arrows pointing downwards. The first equation is $a = w_1x_1 + w_2x_2 + \dots + w_nx_n$. A red arrow points from this equation to the second equation, $y = ka$. Another red arrow points from the second equation to the third equation, $y = k(w_1x_1 + \dots + w_nx_n)$. This sequence shows how a linear combination of inputs, when passed through a linear activation function, results in a linear output, which contradicts the purpose of using a non-linear activation function.

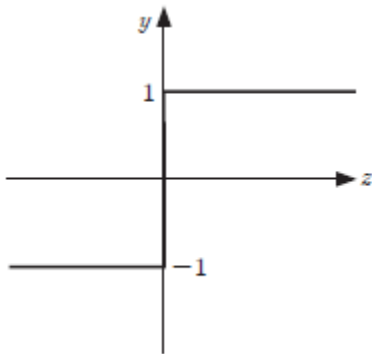
$$\begin{aligned} a &= w_1x_1 + w_2x_2 + \dots + w_nx_n \\ y &= ka \\ y &= k(w_1x_1 + \dots + w_nx_n) \end{aligned}$$

활성화 함수

- 계단 함수 (Step function)

- 출력 z 가 0보다 크거나 같으면 **1**로, 0보다 작으면 **-1**로 분류함

$$y = \begin{cases} 1 & (z > 0) \\ -1 & \end{cases}$$

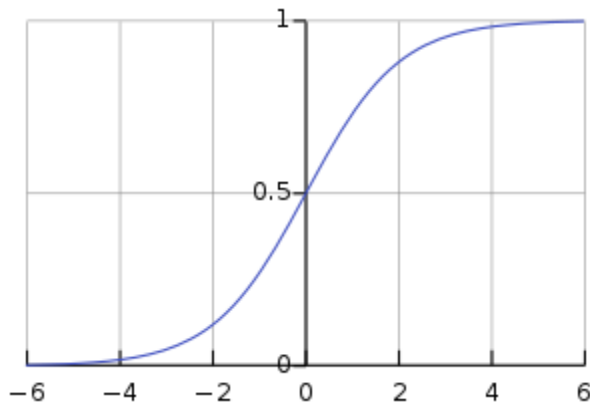


활성화 함수

- 시그모이드 함수 (Sigmoid function)

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}$$

- S자 곡선의 형태를 가짐
- X축 범위 : 실수 전체
- Y축 범위 : 0 ~ 1
- 입력값이 커질수록 1에 수렴하고, 입력값이 작을수록 0에 수렴함



활성화함수

- 기울기소실 문제 (**Vanishing Gradient Problem**) – 개념 및 발생원인
 - (개념) 역전파 학습과정에서 입력층으로 갈수록 기울기(**Gradient**)가 점차 작아져서 입력층에 가까운 층들에서 가중치가 잘 업데이트 안됨
 - (원인) 활성화함수로 시그모이드 함수를 사용하기 때문임.
시그모이드의 미분값은 **0~0.25** 사이값만 표현가능. 즉 역전파 가중치 계산시 **1/4** 감소되고, 세 번 이상 미분계산 반복 시 **0**에 가까운 값
☞ 출력값과 멀어질수록 학습이 되지 않는 현상이 발생함.

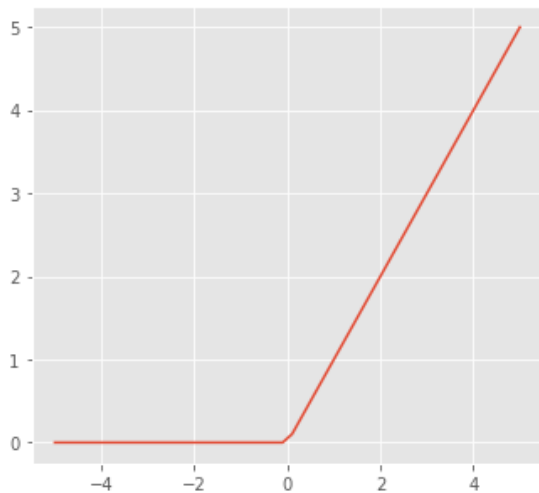
활성화 함수

- 기울기 소실 문제 (**Vanishing Gradient Problem**) - 해결 방안
 - 은닉층의 활성화 함수로 시그모이드 대신에 미분값 범위가 **0~1**인 하이퍼볼릭탄젠트 함수, **ReLU**, **Leaky ReLU**를 사용함.
 - **ReLU**는 **x**가 **0**보다 작을 때는 모든 값을 **0**으로 처리하고, **0**보다 큰 값은 **x**를 그대로 사용하는 방법. 이 방법을 쓰면 **x**가 **0**보다 크기만 하면 미분값이 **1**이 됨. 여러 은닉층을 거치며 곱해지더라도 맨 처음 층까지 사라지지 않고 남아 있을 수 있음

활성화 함수

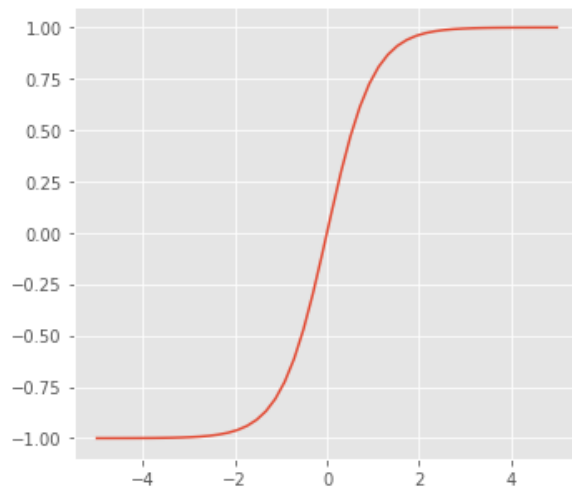
- ReLU 함수

$$f(x) = \max(0, x)$$



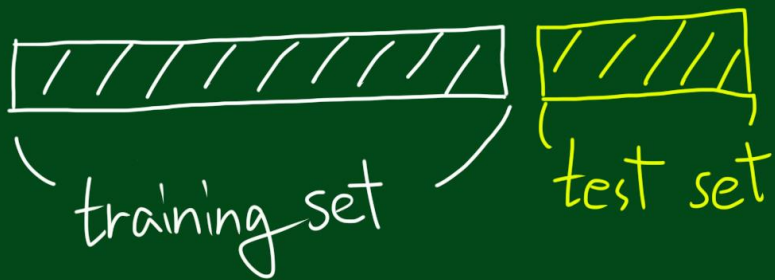
- Tanh 함수

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



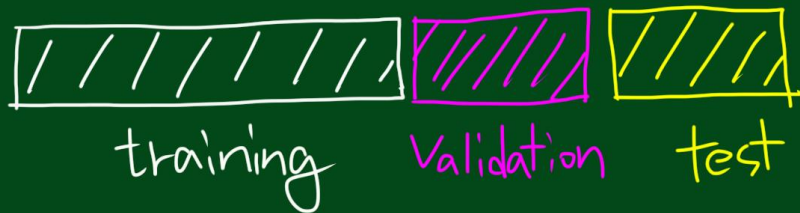
데이터 세트

- 학습데이터 세트
 - **Training set** : 모델을 학습시키기 위해 사용함
 - **Test set** : 모델의 성능을 평가하기 위해 사용함



데이터 세트

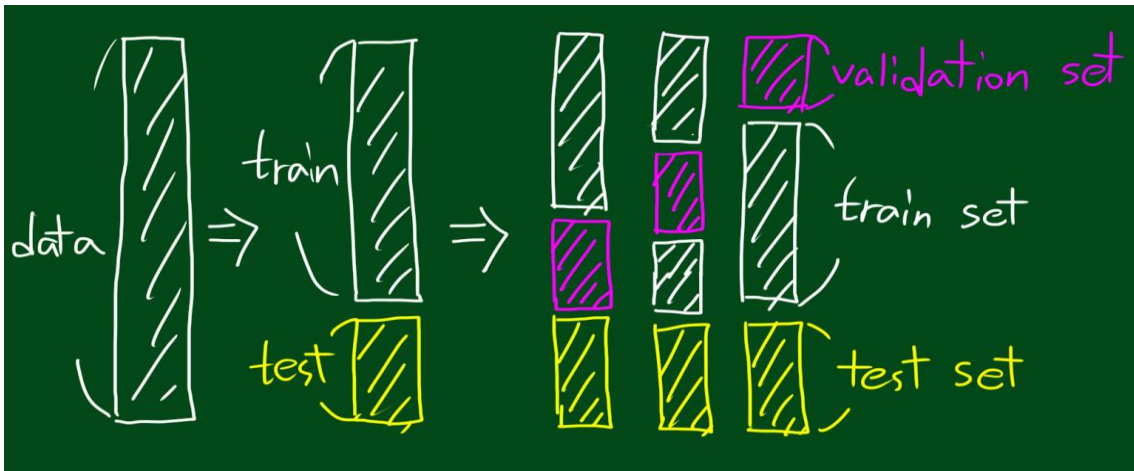
- 검증데이터 세트
 - **Test set** 으로 파라미터 튜닝 → **Test set** 으로 성능평가 시 과적합 발생
 - **Training set** 일부를 잘라 **Validation set**을 만들어 튜닝을 실시해야 함



데이터 세트

- K-fold cross validation

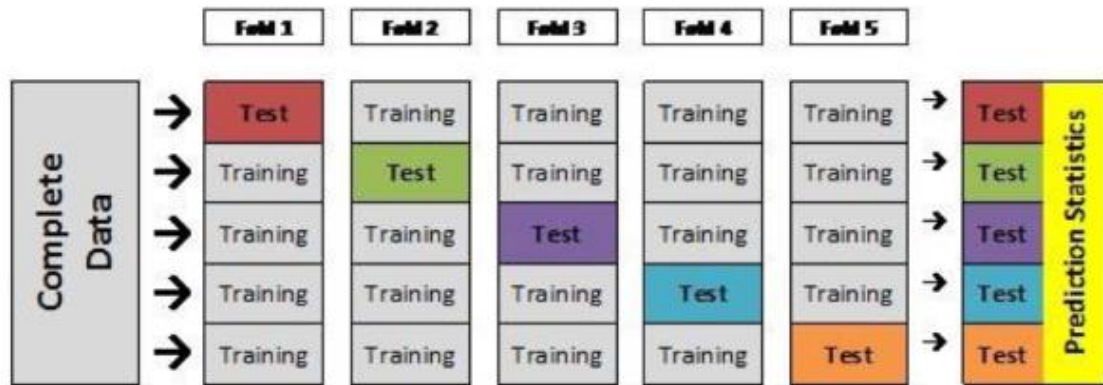
- **validation set** 을 확보해 줄 만큼 학습데이터가 충분하지 못할 경우
- **Training set** 을 k등분하고 **validation set**을 바꾸어 가며 반복 시행함



데이터 세트

- K-fold cross validation

- (장점) 모든 데이터를 학습과 테스트에 다 사용 가능함
- (단점) 수행시간이 오래 걸림



데이터 스케일링

- 데이터 스케일링 필요성
 - 데이터의 스케일이 다른 경우 학습이 수렴하지 않거나 발산할 수 있음
 - 아래 도표에서, 두개의 **feature** (성적과 키)는 값의 범위가 크게 다름

성명	성적 (등수)	키 (cm)
홍길동	3	180
제갈량	1	170
제임스	2	190

데이터 스케일링

- 정규화(Normalization)

- 입력된 x 값들을 모두 **0**과 **1** 사이의 값으로 변환하는 것

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad \dots (\text{정규화 공식})$$

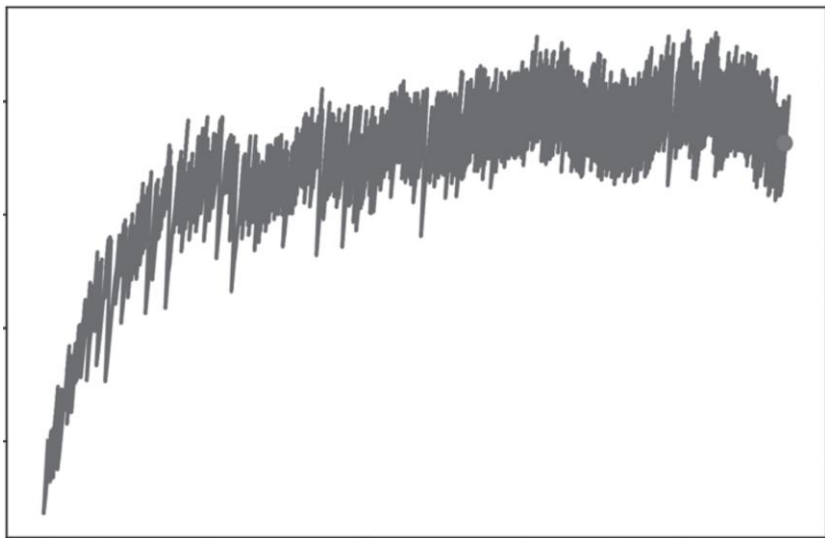
- 표준화(Standardization)

- 입력된 x 들의 정규분포를 평균 **0**, 분산 **1**인 표준정규분포로 변환하는 것

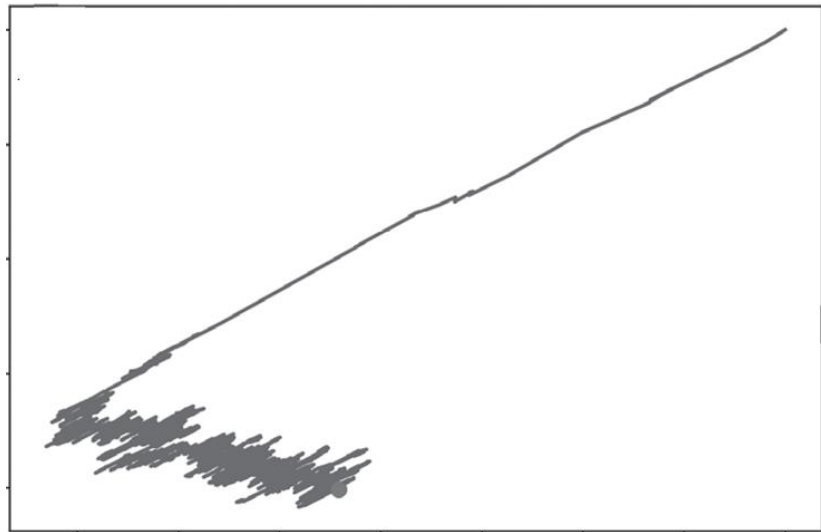
$$X' = \frac{X - \mu}{\sigma} \quad \dots (\text{표준화 공식})$$

데이터 스케일링

- 데이터 스케일링을 하지 않은 경우
 - 가중치를 찾는 경로가 크게 흔들림



- 데이터 스케일링을 한 경우
 - 가중치를 찾는 경로가 매끄러움



규제

- **L1** 규제

- **L1 norm**

$$||w||_1 = \sum_{i=1}^n |w_i|$$

- 손실 함수에 **L1 norm** 추가

$$L = -(y \log(a) + (1-y) \log(1-a)) + \alpha \sum_{i=1}^n |w_i|$$

- α (**hyper parameter**) : 규제의 양을 조절함

규제

- **L1** 규제를 적용한 손실 함수 미분
 - **L1** 규제는 α 에 크게 의존하게 됨
 - **w**는 부호만 결정함

$$\frac{\partial}{\partial w} L = -(y - a)x + \alpha \times \text{sign}(w)$$

규제

- L2 규제

- L2 norm

$$||w||_2 = \sqrt{\sum_{i=1}^n |w_i|^2}$$

- 손실 함수에 L2 norm의 제곱을 더함

$$L = -(y \log(a) + (1-y) \log(1-a)) + \frac{1}{2} \alpha \sum_{i=1}^n |w_i|^2$$

규제

- L2 규제를 적용한 손실 함수 미분
 - L1 규제에 비해 \mathbf{w} 의 영향도 상승




$$\frac{\partial}{\partial w} L = -(y - a)x + a \times w$$

머신러닝 학습결과 평가지표

- 분류 결과 평가지표
 - 정확도 (Accuracy)
 - 정밀도 (Precision)
 - 재현율 (Recall)
 - F-점수 (F-measure)
 - 혼동행렬 (Confusion matrix) : 보조개념임
- 회귀 모델 평가지표
 - 평균제곱근오차 (RMSE, Root Mean Squared Error)
 - 결정계수 (Coefficient of determination)

머신러닝 학습결과 평가지표

- 분류 결과 평가지표 - 정확도 (Accuracy)

- 정확도 = $\frac{\text{정답과 일치한 수}}{\text{전체 데이터 수}}$
- 스팸메일과 정상메일을 분류하는 이진분류작업 사례 분석
 - 수신메일 **100**건을 사람이 직접 분류한 결과 스팸 **60**개, 정상 **40**개
 - 분류기가 모든 메일을 스팸으로 분류하였다면,
 -  분류기의 정확도는 **60%**가 됨
- 무작위로 선택한 결과를 최저성능으로 삼음  이진분류 시 50%
- 그런데 위의 사례에서 모두 스팸이라 했는데도 정확도 60% ??
 -  정확도만으로는 의미가 없는 경우임

머신러닝 학습결과 평가지표

- 분류 결과 평가지표 - 정밀도 (Precision)
 - 출력결과가 정답을 얼마나 맞추었는지를 나타내는 지표
 - 스팸메일과 정상메일을 분류하는 이진분류작업 사례 분석
 - 수신메일 **100**건을 사람이 직접 분류한 결과 스팸 **60**개, 정상 **40**개
 - 분류기가 스팸으로 판정한 메일이 **80**개이고 이중 진짜 스팸이 **55**개라면,
 - 정밀도는 분류기가 스팸으로 판정한 메일 중에서, 진짜 스팸의 비율
 - 따라서 분류기의 정밀도는 **$55/80=0.68$**

머신러닝 학습결과 평가지표

- 분류 결과 평가지표 - 재현율 (Recall)
 - 출력결과가 실제 정답 중에서 얼마나 맞추었는지를 나타내는 지표
 - 스팸메일과 정상메일을 분류하는 이진분류작업 사례 분석
 - 수신메일 **100**건을 사람이 직접 분류한 결과 스팸 **60**개, 정상 **40**개
 - 분류기가 스팸으로 판정한 메일이 **80**개이고 이중 진짜 스팸이 **55**개라면,
 - 재현율은 전체 데이터에 포함된 실제 스팸 중에서 분류기가 스팸으로 판정한 메일 개수
 - 따라서 분류기의 재현율은 $55/60=0.92$
 - 정밀도보다 재현율이 **1**에 가까움 🖐 이 분류기는 재현율을 중시

머신러닝 학습결과 평가지표

- 정밀도를 중시하는 경우
 - 빠뜨리는 개수가 많더라도 정확한 예측이 필요한 경우
 - 가끔 스팸을 보아도 좋으니 정상 메일이 스팸으로 걸러지지 않도록 하는 것을 희망하는 경우
- 재현율을 중시하는 경우
 - 오탐율이 높더라도 놓치는 것이 없도록 하는 경우
 - 나중에 전체 데이터를 놓고 볼 때 놓친 개수가 얼마나 되는 지를 중시

머신러닝 학습결과 평가지표

- 분류 결과 평가지표 - F-점수 (F-measure)
 - 정밀도와 재현율의 상충관계를 평가에 반영
 - 실제 분류기를 비교하는 데 사용되는 지표이며, 조화평균에 해당함
 - $$F1\text{점수} = \frac{2}{\frac{1}{\text{정밀도}} + \frac{1}{\text{재현율}}} = \frac{2}{\frac{1}{0.69} + \frac{1}{0.92}} = 0.79$$
 - 재현율과 정밀도가 균형을 이룰 때 F점수는 높아짐
 - F점수가 높다는 것은 재현율과 정밀도가 고르게 높다는 뜻임

머신러닝 학습결과 평가지표

- 혼동행렬 (Confusion matrix)

		예측 결과	
		양성(스팸)	음성(정상)
실제 정답	양성 (스팸)	진짜 양성 (True Positive)	거짓 음성 (False Negative)
	음성 (정상)	거짓 양성 (False Positive)	진짜 음성 (True Negative)

Evaluation of predictive performance in machine learning

- Confusion matrix

Confusion matrix		Predictions	
		Positive	Negative
Correct answers	Positive	TP (True Positive)	FN (False Negative)
	Negative	FP (False Positive)	TN (True Negative)

머신러닝 학습결과 평가지표

- 혼동행렬 (Confusion matrix)

		예측 결과	
		양성(스팸)	음성(정상)
실제 정답	양성 (스팸)	55 (TP)	5 (FN)
	음성 (정상)	25 (FP)	15 (TN)

- 정밀도

$$55 / (55+25) = 0.69$$

- 재현율

$$55 / (55+5) = 0.92$$

- 정확도

$$(55+15) / (55+5+25+15) = 0.7$$

머신러닝 학습결과 평가지표

- 회귀 모델 평가지표
 - 평균제곱근오차 (RMSE, Root Mean Squared Error)

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^n (x_{1,i} - x_{2,i})^2}{n}}$$

머신러닝 학습결과 평가지표

- 회귀 모델 평가지표 - 결정계수

$$R^2 = \frac{SSE}{SST} = 1 - \frac{SSR}{SST}$$

계산하기 전에 SST, SSE, SSR에 대해서 먼저 알 필요가 있습니다.

1 - SST(Total Sum of Squares)

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

2 - SSE(Explained Sum of Squares)

$$SSE = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2$$

SSE는 추정값에서 관측값의 평균(혹은 추정치의 평균)을 뺀 결과의 총합입니다.

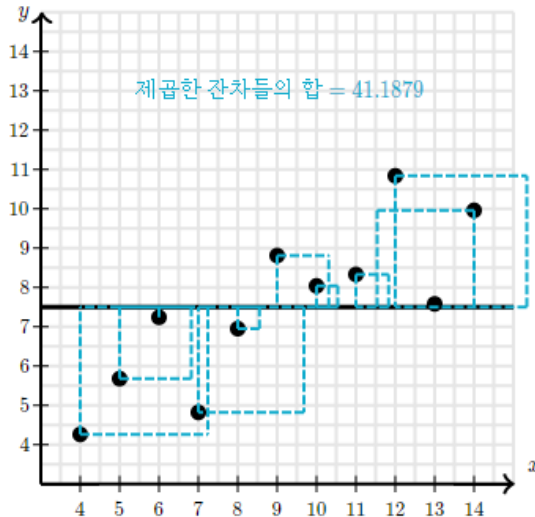
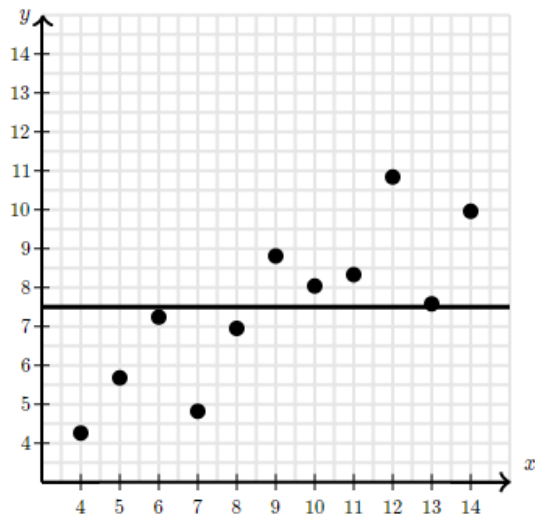
3 - SSR(Residual Sum of Squares)

$$SSR = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

SSR은 관측값에서 추정값을 뺀 값, 즉 잔차(Residual)의 총합입니다.

머신러닝 학습결과 평가지표

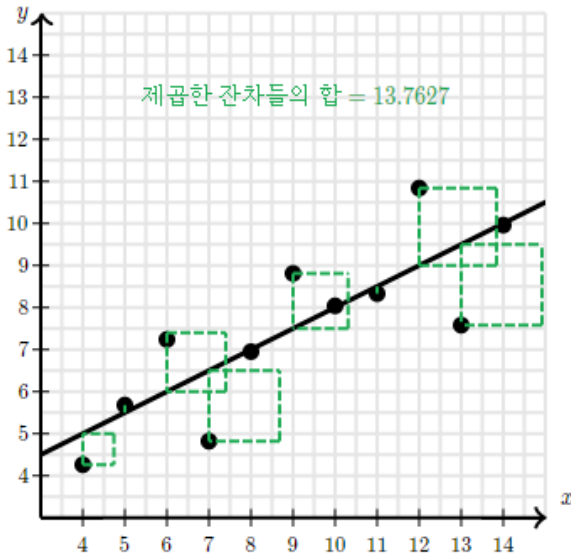
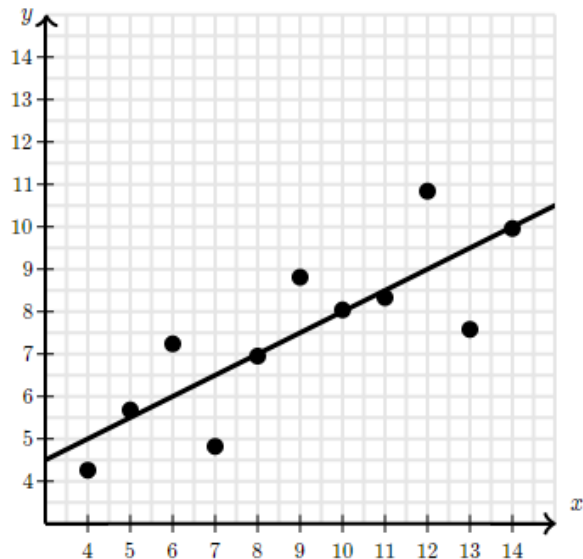
- 회귀 모델 평가지표 - 결정계수
 - 예측직선이 평균일 때 ➔ 제공한 잔차들의 합은 **41.1879**



머신러닝 학습결과 평가지표

- 회귀 모델 평가지표 - 결정계수

- 회귀로 예측했을 때 ➔ 제공한 잔차들의 합은 **13.7627**



머신러닝 학습결과 평가지표

- 회귀 모델 평가지표 - 결정계수
 - 회귀로 예측했을 때 ➔ 제공한 잔차들의 합은 **13.7627**
 - 줄어든 오차를 원래의 예측오차의 백분율로 나타낸 것이 결정계수
 - 결정계수를 통하여, 최소 제곱 회귀를 사용할 때 예측 오차가 얼마나 줄었는지 알 수 있음.

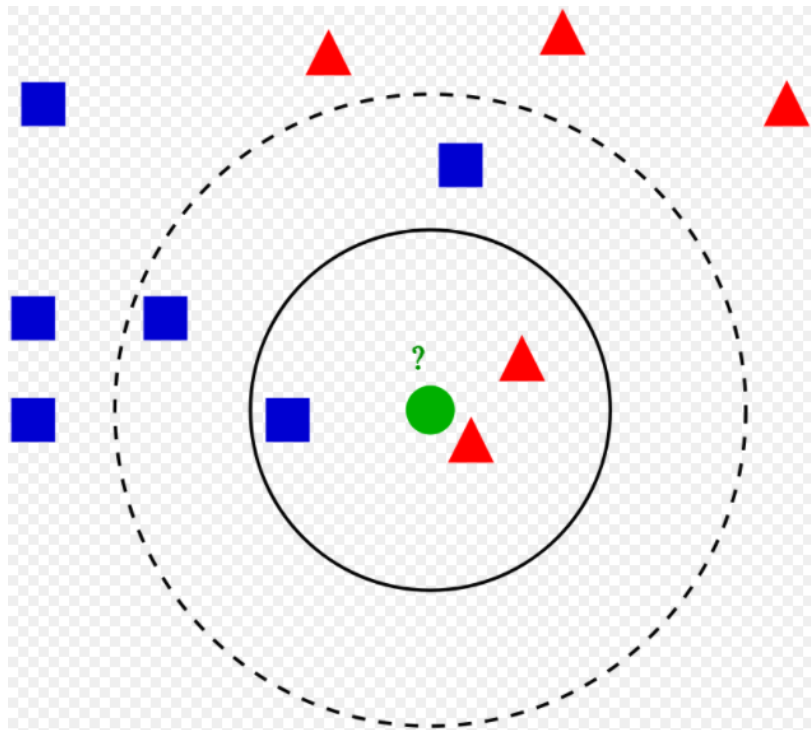
$$\frac{41.1879 - 13.7627}{41.1879} = \frac{27.4252}{41.1879} \approx 66.59\%$$

K-최근접이웃 알고리즘 (K-NN)

- 정의
 - 인접한 **k**개의 이미 학습된 데이터 중 가장 많은 수가 속하는 클래스를 예측클래스로 삼는 알고리즘
- 수행원리
 - 이미 학습된 데이터 중 새로 입력된 데이터와 거리가 가장 가까운 **K**개를 선택한다.
 - 이 **K**개 데이터의 클래스를 확인하고, 가장 많은 데이터가 속한 클래스를 찾는다.
 - 이 클래스를 새로운 데이터의 클래스로 삼는다.

K-최근접이웃 알고리즘 (K-NN)

검증 표본(초록색 원)은 첫 번째(파랑 네모)의
항목이나 두 번째(빨강 삼각형) 항목으로
분류되어야 한다. 만약 “ $k = 3$ ” (실선으로
그려진 원)이면 빨강삼각형 항목으로
할당되어야 한다. 왜냐하면 2개의 삼각형과
1개의 사각형만이 안쪽 원 안에 있기 때문이다.
만약 “ $k = 5$ ” (점선으로 그려진 원)이면
파랑네모 항목으로 분류되어야 한다. 왜냐하면
바깥쪽 원 안에는 사각형이 3개, 삼각형이
2개가 있어서, 사각형이 더 다수이기 때문이다.
(출처) 위키백과사전



KNN (K-Nearest Neighbor)

- 특징
 - 지도학습 알고리즘 중 하나이며, 매우 직관적이고 간단함
 - 어떤 데이터가 주어지면 그 주변(이웃)의 데이터를 살펴본 뒤 더 많은 데이터가 포함되어 있는 범주로 분류함 (지역성에 근거함)
 - 즉, 주변 친구들을 보면 대강 그 사람이 어떤 사람인지 알 수 있다는 식의 논리임
 - **KNN은 Lazy Model** : 평소에 열심히 훈련해서 모델을 사전에 구축해 놓거나 하지 않고, 새로운 데이터가 들어오면 그제서야 **k**개의 인접데이터를 세어보고 분류해 줌. 따라서 실시간으로 빠르게 예측이 수행되나 훈련은 필요 없는 알고리즘임.

K-최근접이웃 알고리즘 (K-NN)

- 고려사항
 - 스케일의 차이가 클 경우 학습이 제대로 되지 않을 수 있음 (예: 특징 간에 평균치 크기가 수십 배 차이 날 경우)
 - ☞ 정규화(Normalization) 필수
 - 하이퍼파라미터 **k**값의 설정이 매우 중요함

회귀분석

- 정의
 - 독립변수와 종속변수 사이의 관계를 통계적으로 분석하는 기법
 - ※ 종속변수 : 독립변수의 변화에 영향을 받는 변수
- 종류
 - 단순선형회귀 : x 1개, y 1개
 - 다중선형회귀, 다항회귀 : x 여러 개, y 1개
- $y = \beta_0 + \beta_1 x$
 - x (독립변수), y (종속변수), β_0 , β_1 (회귀계수)

단순 선형회귀

- 정의
 - 하나의 독립변수와 하나의 종속변수 사이의 회귀직선을 추론하는 기법
- 사례
 - 온도에 따른 화학물질의 반응속도를 예측
 - 온도 = x = 독립변수 **or** 설명변수 **or** 입력변수
 - 반응속도 = y = 종속변수 **or** 반응변수 **or** 출력변수
- 수식형태
 - $y = \beta_0 + \beta_1 x$

다중 선형회귀

- 개념
 - 2개 이상의 독립변수와 하나의 종속변수 사이의 관계를 추론하는 기법
 - 독립변수 간 상관관계가 높아 발생하는 다중공선성 문제처리 필요
- 사례
 - 온도, 압력, 촉매에 따른 화학물질의 반응속도를 예측
 - 온도(**x1**), 압력(**x2**), 촉매(**x3**) = 독립변수
 - 반응속도 = **y** = 종속변수
- 수식형태
 - $y = \beta_0 + \beta_1x_1 + \beta_2x_2 + \beta_3x_3$

다항 회귀

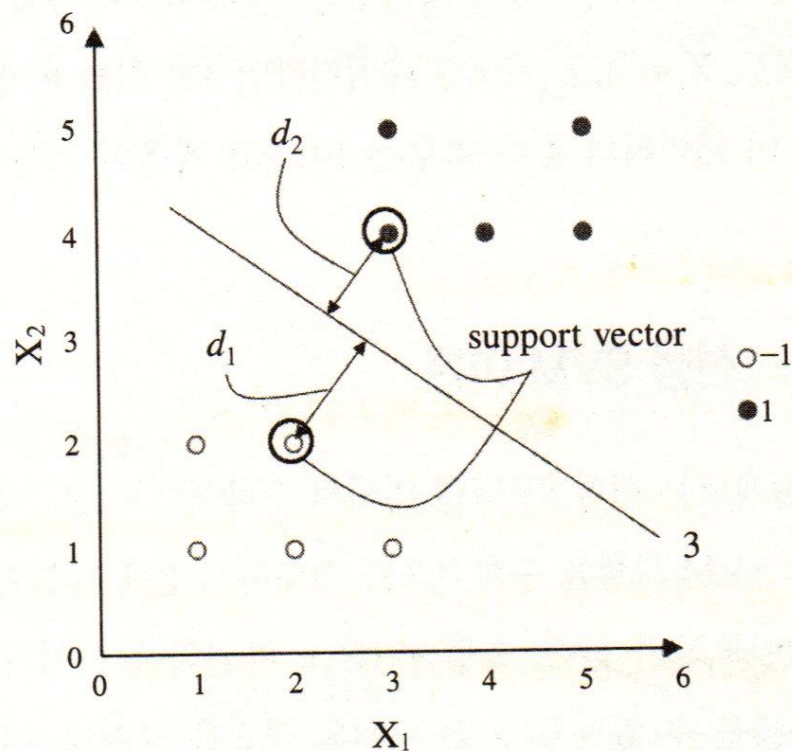
- 정의
 - 회귀가 독립변수의 단항식이 아닌 **2차, 3차**의 다항식으로 표현되는 것
 - 독립변수와 종속변수가 선형적 관계가 아니라 곡선형태
- 사례
 - 독립변수를 가로, 세로 ➔ 면적
- 수식형태
 - $y = \beta_0 + \beta_1x + \beta_2x^2$

다항 회귀

- 고려사항
 - 선형회귀와 비선형회귀를 구분하는 것은 독립변수의 선형성이 아니라, 회귀계수의 선형성임 ➔ 다항회귀도 선형회귀임
 - 따라서 다항회귀는 다중선형회귀의 특정사례로 간주됨
 - 다항함수로의 모델링은 선형회귀를 매우 강력하게 만들어, 모델이 데이터에 과적합하도록 만들기도 함

SVM (Support Vector Machine)

- **SVM** : 마진을 최대화하는 초평면을 찾는 알고리즘
- 초평면 (**hyperplane**) : 그림에서 3번 결정경계 직선임
- 마진 : **support vector**에서 초평면까지의 거리(**d1, d2**) 중 작은 것
- **Support Vector** : 초평면과 가장 가까운 데이터




군집 (클러스터링)

- 정의
 - 주어진 입력 데이터들을 클러스터 내부 유사성을 최대화하고 클러스터 사이 유사성을 최소화하도록 특정한 척도 공간에서 몇 개의 그룹으로 분류하는 것
- 요소 기술
 - 유사성을 측정하기 위해 사용되는 가장 유명한 척도 방법은 데이터 포인트들 간의 유클리디안 거리 (**Euclidean distance**)
 - 가장 자주 사용되는 평가 함수는 제곱오차평가 (**squared error criterion**)

K-평균(K-means) 알고리즘

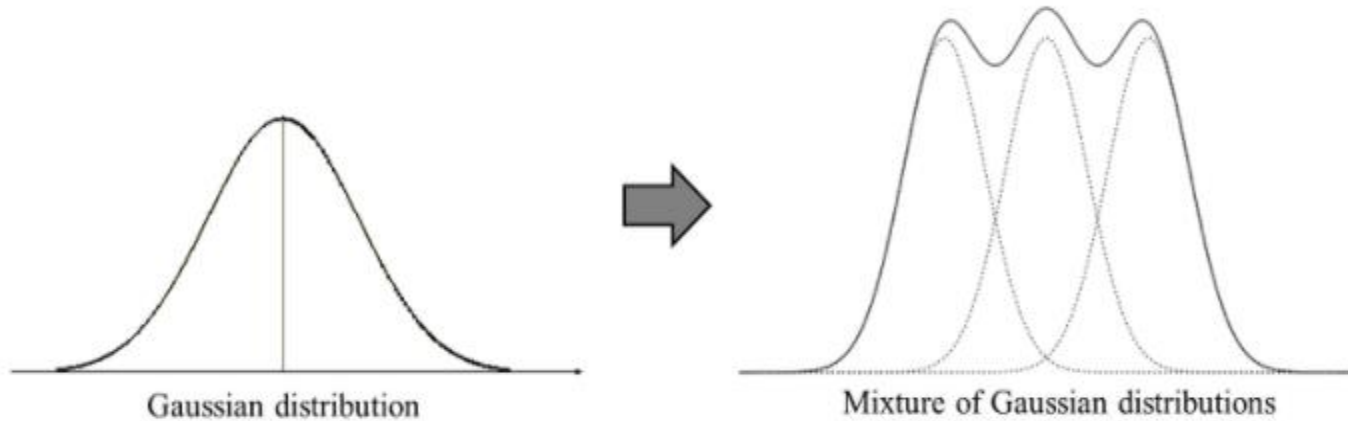
- 개념
 - 데이터를 **k**개의 클러스터로 묶는 알고리즘
- 특징
 - 각 클러스터와의 거리 차이 분산을 최소화하여 데이터를 분류
 - 센트로이드(**Centroid**) : 클러스터의 중심
 - 문제는 몇 개의 **k**를 지정해야 하는가이며, 이를 위해 등장한 것이 바로 엘보우(**Elbow**) 법

GMM (Gaussian Mixture Model)

- 개념
 - (정의) **Gaussian** 분포가 여러 개 혼합된 **clustering** 알고리즘
 - (착안) 현실에 존재하는 복잡한 형태의 확률 분포를 **K**개의 **Gaussian distribution**을 혼합하여 표현하자는 것  전체 분포에서 하위 분포가 존재한다고 보고, 데이터가 모수를 갖는 여러 개의 분포로부터 생성되었다고 가정하는 **Mixture Model** 모델.
 - **K**는 데이터를 분석하고자 하는 사람이 직접 설정해야 함.

GMM (Gaussian Mixture Model)

- Mixture Model



GMM (Gaussian Mixture Model)

- 장점
 - K 평균보다 유연하게 다양한 데이터 세트에 잘 적용됨
- 단점
 - K 평균과 달리 군집의 중심 좌표를 구할 수 없어 군집 중심 표현의 시각화가 불가능
 - 시간이 오래 걸린다.

차원 축소

- 고차원 데이터 문제점
 - 데이터에 포함된 노이즈 비율 증가
 - 모델학습과 추론 관련 계산 복잡도 증가
 - 더 많은 데이터를 요구함
- 차원축소 장점
 - 훈련속도가 빨라짐
 - 잡음 또는 불필요한 세부사항을 걸러내므로 성능을 높일 수도 있음

PCA (Principal Component Analysis)

- 개념
 - PCA(주성분 분석)는 데이터를 가장 잘 표현하는 초평면을 찾아 분산을 최대한 보존하는 축을 찾는 것
 - ➔ 원본 데이터셋과 투영된 것 사이의 평균 제곱 거리를 최소화하는 축
 - 분산이 최대한 보존되는 축을 선택 ➔ 정보가 가장 적게 손실
- 단점
 - 해당 성분이 의미하는 바를 직관적으로 설명하기 어려움

t-SNE (Stochastic Neighbor Embedding)

- 배경
 - n 차원에 분포된 이산 데이터를 $k(n \text{ 이하의 정수})$ 차원으로 축소하며 거리 정보를 보존하되, 거리가 가까운 데이터의 정보를 우선하여 보존하기 위해 고안됨
- 개념
 - 비슷한 샘플은 가까이, 비슷하지 않은 샘플은 멀리 떨어지도록 하면서 차원을 축소함.
 - 주로 시각화에 많이 사용되며, 특히 고차원 공간의 샘플군집을 시각화할 때 사용.

결정트리 (Decision Tree)

- 개념
 - 의사 결정 규칙과 그 결과들을 트리 구조로 도식화한 의사 결정 지원 도구의 일종
 - 결정 트리는 운용 과학, 그 중에서도 의사 결정 분석에서 목표에 가장 가까운 결과를 낼 수 있는 전략을 찾기 위해 주로 사용됨.

결정트리 (Decision Tree)

- 결정트리 구현 시 과적합
 - 의사결정 나무에서는 가지가 너무 많거나 나무의 크기가 클 때 (**fully grown**) 과적합 문제가 발생함
 - 결정 트리는 운용 과학, 그 중에서도 의사 결정 분석에서 목표에 가장 가까운 결과를 낼 수 있는 전략을 찾기 위해 주로 사용됨.
- 결정트리 과적합 해소기법
 - 가지치기
 - 앙상블기법

결정트리 (Decision Tree)

- 가지치기
 - (정의) 의사결정나무에서 과적합을 방지하기 위해 적절한 수준에서 **terminal node**를 결합해 주는 것
 - 사전가지치기 : 트리의 최대 깊이, 각 노드에 있어야 할 최소 관측값 수 등을 미리 지정하여 트리를 만드는 도중에 **stop** 하는 것
 - 사후가지치기 : 트리를 먼저 **full tree**로 만든 후 적절한 수준에서 **terminal node**를 결합해 주는 것

결정트리 (Decision Tree)

- 앙상블기법
 - (정의) 여러 개의 모델을 조합하여 과적합을 방지하고 예측의 정확도를 높이는 방법
 - (종류)
 - ① 투표기반
 - ② 배깅
 - ③ 부스팅

앙상블 기법

- 투표기반
 - (직접투표) 더 좋은 분류기를 만들기 위해 각 분류기의 예측을 모아 다수결 투표로 정하는 것
 - (간접투표) 개별 분류기의 예측을 평균해서 확률이 가장 높은 범주를 예측
- 배깅
 - 훈련세트 서브셋을 무작위로 나누어서 분류기를 각기 다르게 학습시키되, 훈련세트에서 중복을 허용하는 샘플링방법임 (허용하지 않는 기법은 페이스팅)

앙상블 기법

- **Random Forest**

- 나무들이 모이면 숲이 됨. 즉 형태가 조금씩 다른 **Decision Tree** 모델들이 모이면 **Random Forest**가 됨 → 분류, 회귀 분석 등에 사용되는 앙상블 학습 방법의 일종으로, 훈련 과정에서 구성한 다수의 결정 트리로부터 부류(분류) 또는 평균 예측치(회귀 분석)를 출력함으로써 동작함.

- **Ada Boost**

- 이전의 분류기에 의해 잘못 분류된 것들을 이어지는 약한 학습기들이 수정해줄 수 있음.

수고하셨습니다