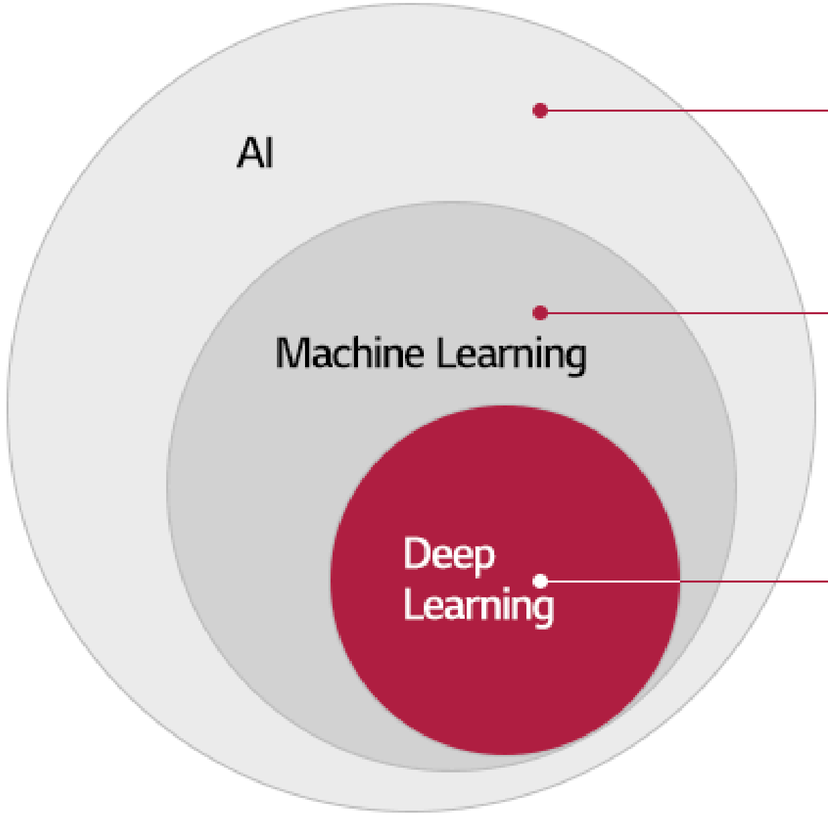
1학기 2차 정기시험 대비

2026 인공지능 모델링

딥러닝(Deep Learning)

인간의 뇌 구조에서 영감을 받아 설계된 **인공신경망(Artificial Neural Networks, ANN)**을 기반으로, 기계가 방대한 데이터를 통해 스스로 학습하고 복잡한 패턴을 인식하는 **머신러닝(기계학습)의 한 분야**



인공지능(Artificial Intelligence)

사람이 해야 할 일을 기계가
대신할 수 있는 모든 자동화에 해당

머신러닝(Machine Learning)

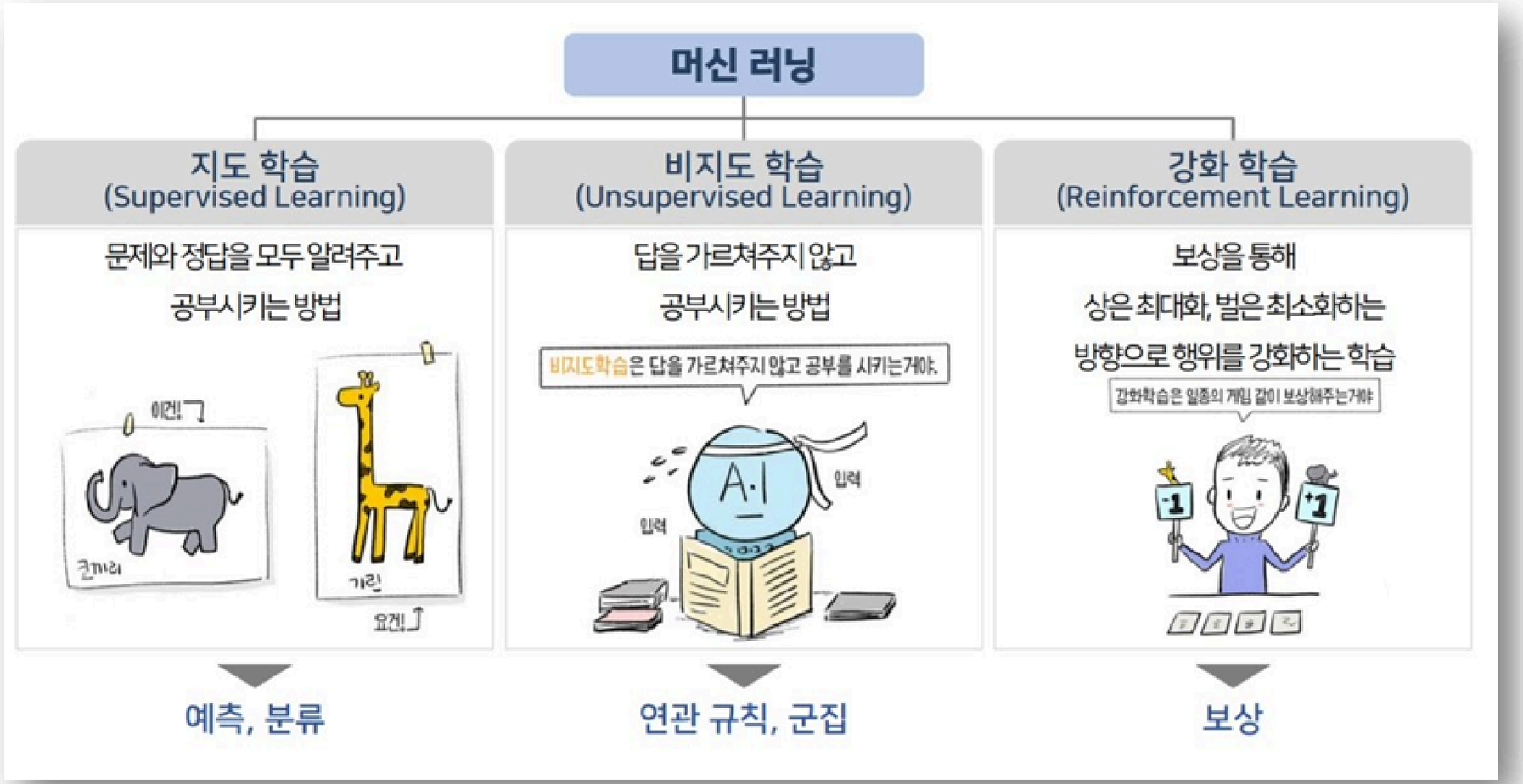
명시적으로 규칙을 프로그래밍하지 않고
데이터로부터 의사결정을 위한 패턴을
기계가 스스로 학습

- Linear Regression
- Logistic Regression
- kNN, SVM

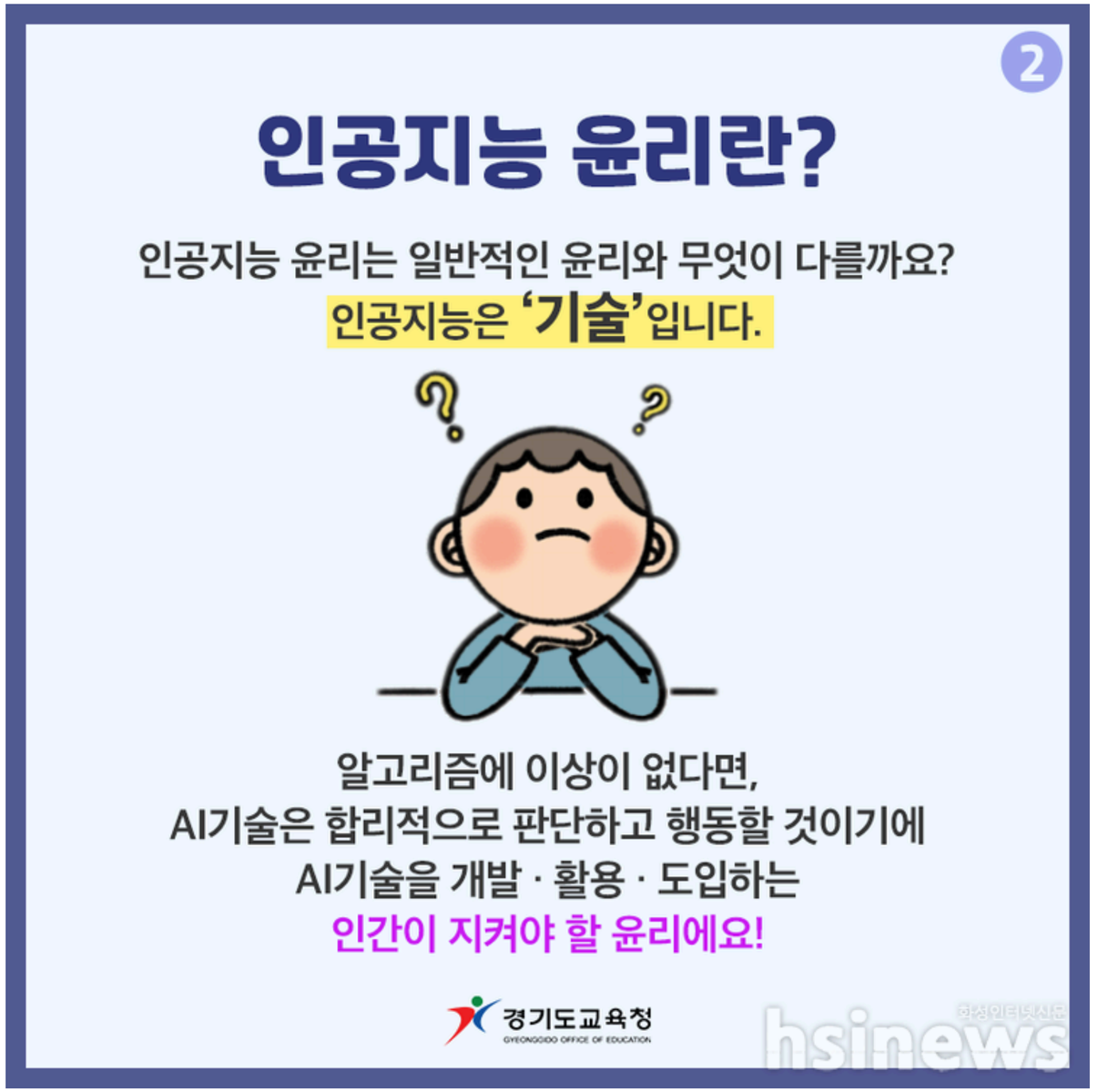
딥러닝(Deep Learning)

인공신경망 기반의 모델로, 비정형
데이터로부터 특징 추출 및 판단까지
기계가 한 번에 수행

머신러닝


머신 러닝
지도 학습
(Supervised Learning)
문제와 정답을 모두 알려주고
공부시키는 방법
이건 코끼리
이건 기린
비지도 학습
(Unsupervised Learning)
답을 가르쳐주지 않고
공부시키는 방법
비지도학습은 답을 가르쳐주지 않고 공부를 시키는거야.
A.I
입력
입력
강화 학습
(Reinforcement Learning)
보상을 통해
상은 최대화, 벌은 최소화하는
방향으로 행위를 강화하는 학습
강화학습은 일종의 게임 같이 보상해주는거야
+1
-1
예측, 분류
연관 규칙, 군집
보상

인공지능 윤리


2
인공지능 윤리란?
인공지능 윤리는 일반적인 윤리와 무엇이 다를까요?
인공지능은 '기술'입니다.
알고리즘에 이상이 없다면,
AI기술은 합리적으로 판단하고 행동할 것이기에
AI기술을 개발 · 활용 · 도입하는
인간이 지켜야 할 윤리에요!
경기도교육청
GYEONGGIDO OFFICE OF EDUCATION
hsinews

인공지능 윤리

2020년 과학기술정보통신부 “국가 인공지능 윤리 기준“ 발표

인공지능 윤리 기준 3대 기본 원칙

인간의 존엄성

인간은 기계 제품과 교환 불가능한 가치가 있다.

인공지능의 개발 및 활용은 인간에게 해가 되지 않도록 해야 한다.

사회의 공공선

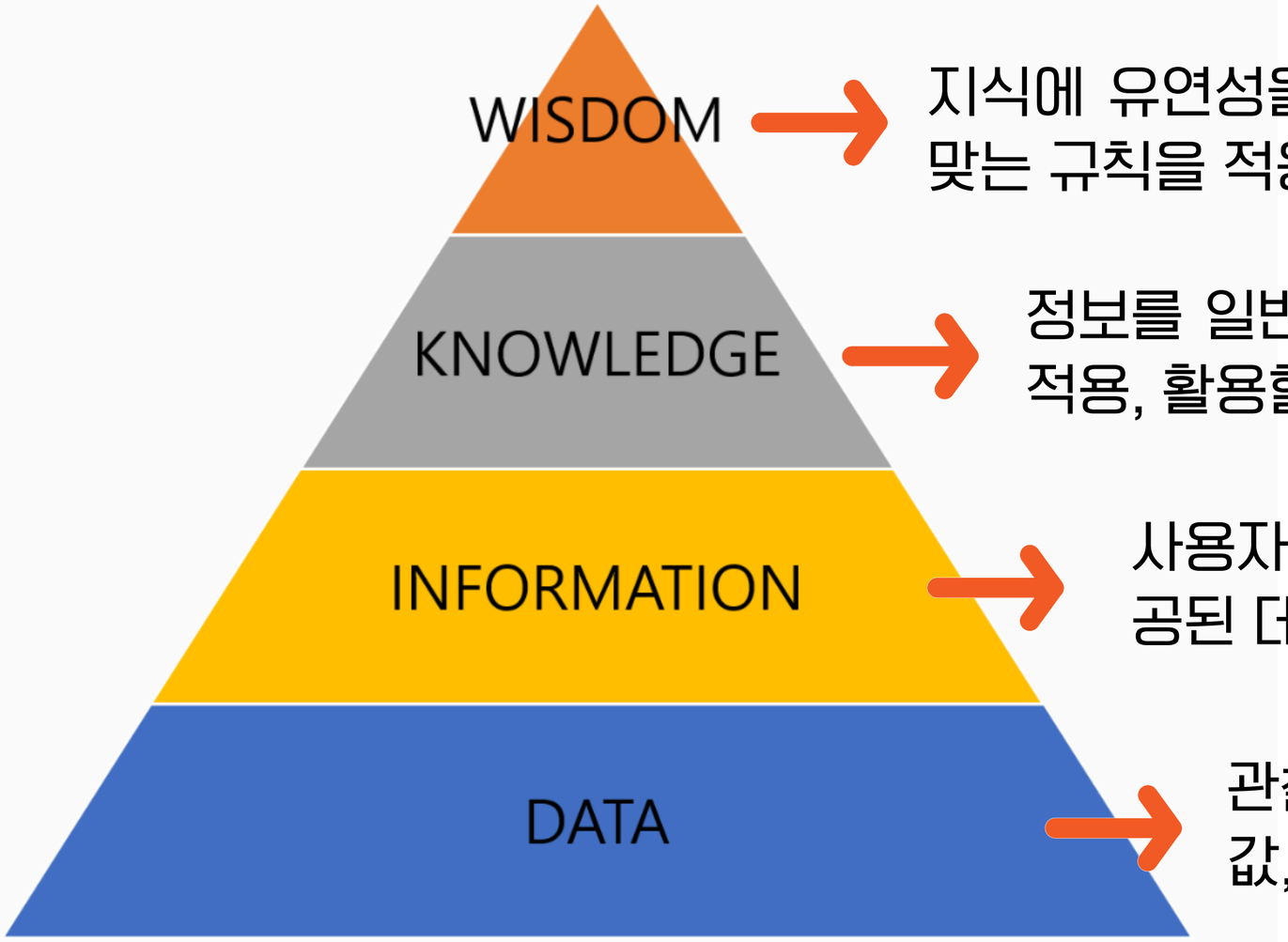
가능한 많은 사람의 행복을 추구한다.

인공지능의 개발 및 활요은 인류의 보편적 복지를 향상시켜야 한다.

기술의 합목적성

인공지능 기술은 인간의 삶에 필요한 도구로서 목적에 맞게 개발 및 활용되어야 하며 그 과정도 윤리적이어야 한다.

데이터 피라미드



지식에 유연성을 더하고, 상황과 맥락에 맞는 규칙을 적용하는 것을 의미

정보를 일반화하고 체계화하여 바로 적용, 활용할 수 있도록 만든 것

사용자의 필요에 따라 정제되고 가공된 데이터

관찰, 측정을 통해 수집된 사실이나 값, 수치, 문자 등

데이터의 형태에 따른 종류

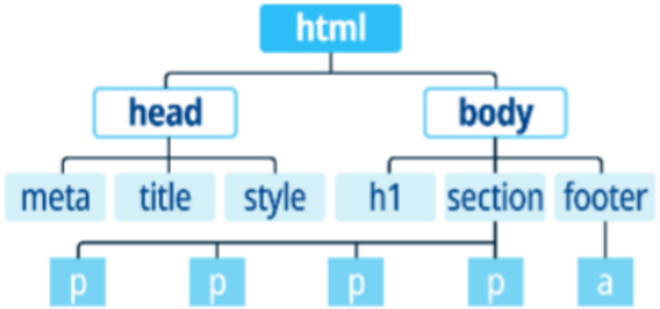
정형데이터

ID	Name	AGE	SEX
01	KIM	32	M
02	LEE	26	F
03	PARK	72	F
04	CHOI	15	M

미리 정해놓은 형식과 구조에 따라 저장되도록 구성하여 고정된 필드에 저장되는 데이터

예) 관계형DB, 스프레드시트 등

반정형데이터



데이터의 구조 정보를 데이터와 함께 제공하는 데이터로 데이터의 구조와 형식이 변경될 수 있는 데이터

예) HTML, NoSQL, JSON

비정형데이터

정해진 구조가 없는 데이터 주로 소셜미디어 플랫폼에서 생

예) 이미지, 동영상, 오디오, 텍스트

회귀(Regression)

날짜	요일	온도	판매량
2월 5일	일	-3	30
2월 6일	월	-5	50
2월 7일	화	-8	80
2월 8일	수	-6	?

최소제곱법

- 데이터들 사이의 관계를 가장 잘 설명하는 모델(보통 직선이나 곡선)을 찾기 위해, 실제 관측값과 모델 예측값 사이의 오차(잔차)를 제곱하여 합한 값을 최소화

$h_\theta = \theta_0 + \theta_1 x$ 일 때,

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

$$\theta_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

최소제곱법의 한계

- 역행렬을 구하기 위해서는 정사각행렬이어야 함. 만약 정사각행렬이 아니면 유사역행렬(Pseudo-Inverse)를 사용해야 하는데 일반 역행렬보다 연산이 복잡해 짐.
- 행렬 연산은 데이터 개수나 특성의 개수가 많아질수록 계산 비용이 급격히 증가함.
- 만약 데이터가 수억개라면??

→ **경사하강법(Gradient Descent)**

경사하강법(Gradient Descent)

- 임의의 가중치를 정한 후 MSE함수의 기울기가 0이 되는 방향으로 가중치를 업데이트 해나간다.
- 기울기가 0이 되는 지점이 최적 값이다.
- 기울기가 클 수록 최적의 값과 멀리 떨어져 있다.

MSE의 기울기가 0인 지점을 찾는 방법

1. 임의의 가중치 θ_j 를 정한 후 예측값을 계산하고 실제값과의 차이를 계산한다.
2. θ_j 를 보정하여 실제값과 예측값의 차이가 작아지도록 한다.

❓ 어떻게 오차가 적은 가중치 값으로 보정할 수 있을까?

최소제곱법 vs 경사하강법

최소제곱법

- 단순한 계산
- 독립변수의 개수가 많아질 수록 행렬의 크기가 커지고 연산이 복잡해 진다.

경사하강법

- 함수가 너무 복잡해 미분 계수를 구하기 어려운 때 사용한다.
- 데이터 양이 많은 경우 효율적으로 계산하기 위해 사용한다.

회귀(Regression) - 캘리포니아 집값 예제

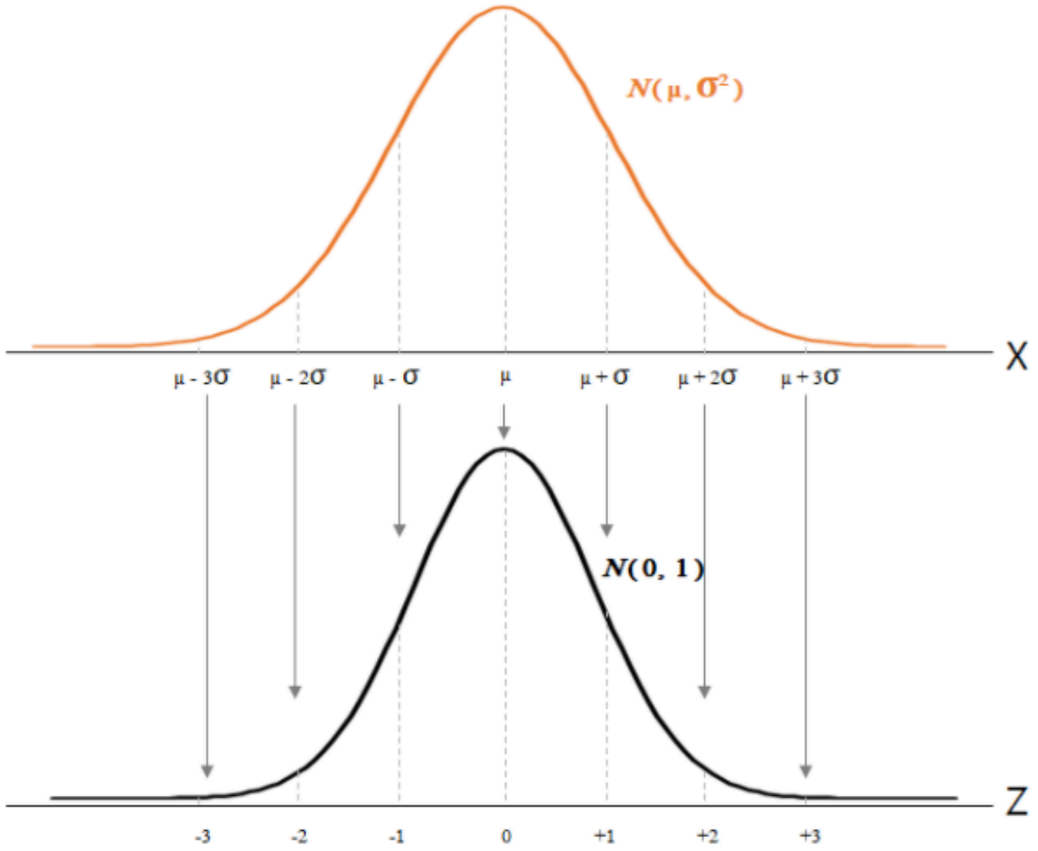
https://colab.research.google.com/drive/1T-H8b1062ZH5Zwx4g9CI2nds_prYHJYJ?usp=sharing

표준화(Standardization/z-score Scaling)

- 각 데이터에 대해 평균을 빼고 표준표차로 나눈다.

$$z = \frac{x - \mu}{\sigma}$$

- 데이터를 평균 0, 표준편차 1인 표준정규분포 형태로 변환시킨다.
- 이상치에 강하며, 범위가 정해져 있지는 않다.
- 평균과 표준편차가 중요한 알고리즘(선형회귀, SVM 등)에 주로 사용



정규화(Normalization/Min-Max Scaling)

- 통계 분포보다는 데이터의 범위 자체에 집중하는 방식
- 모든 데이터를 0 ~ 1 또는 -1 ~ 1의 값으로 변환

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- 이상치에 매우 취약
- 거리기반 알고리즘(knn,k-means)에서 유리

표준화와 정규화

	표준화	정규화
주요 목적	데이터의 분포를 표준정규분포로 변경	데이터의 범위를 [0, 1]로 변경
CLT와 관련성	데이터를 정규분포를 따른다고 가정할 때 효과	분포와 상관없이 범위를 맞출 때 사용
이상치 영향	작음	큼
추천모델	로직스틱 회귀, SVM, 신경	KNN, K-means

- 데이터에 이상치가 많다.
- 데이터 분포가 정규분포에 가깝다.

→ 표준화 사용

- 거리기반 알고리즘이다.

→ 정규화 사용

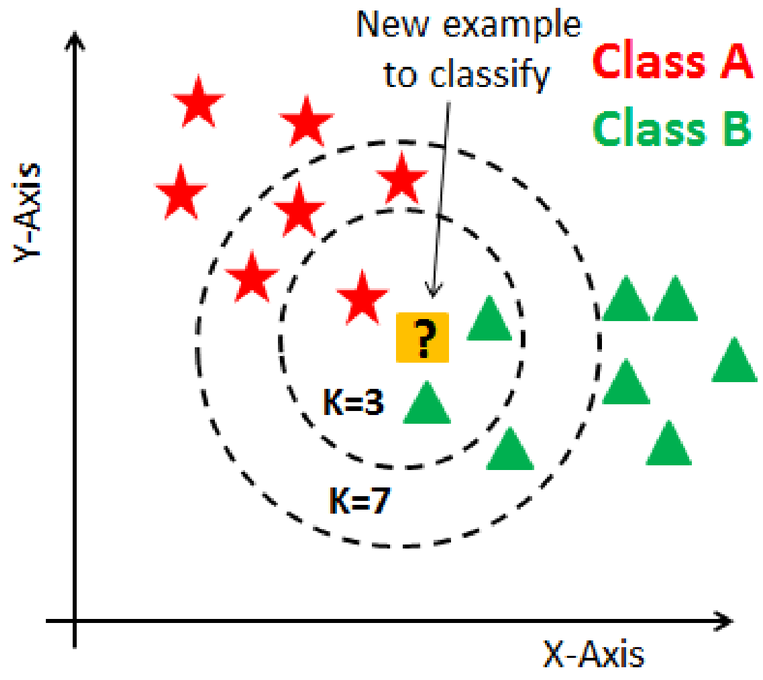
탈모 데이터 과제12

https://colab.research.google.com/drive/1poFse1-0NN2c4UdOxhpp2t7b1rMsnv-E?usp=sharing

kNN(k-Nearest Neighbor)

- 기존 데이터 중 가장 가까운 k개를 바탕으로 새로운 데이터를 예측하고 분류하는 알고리즘
- 판별하려는 데이터와 인접한 데이터 k개를 찾아 그 중 빈도수가 가장 높은 데이터를 범주로 분류
- 거리를 측정할 때에는 주로 Euclidean distance를 사용

K 정하기



k = 3

새로운 샘플이 Class B로 분류됨.

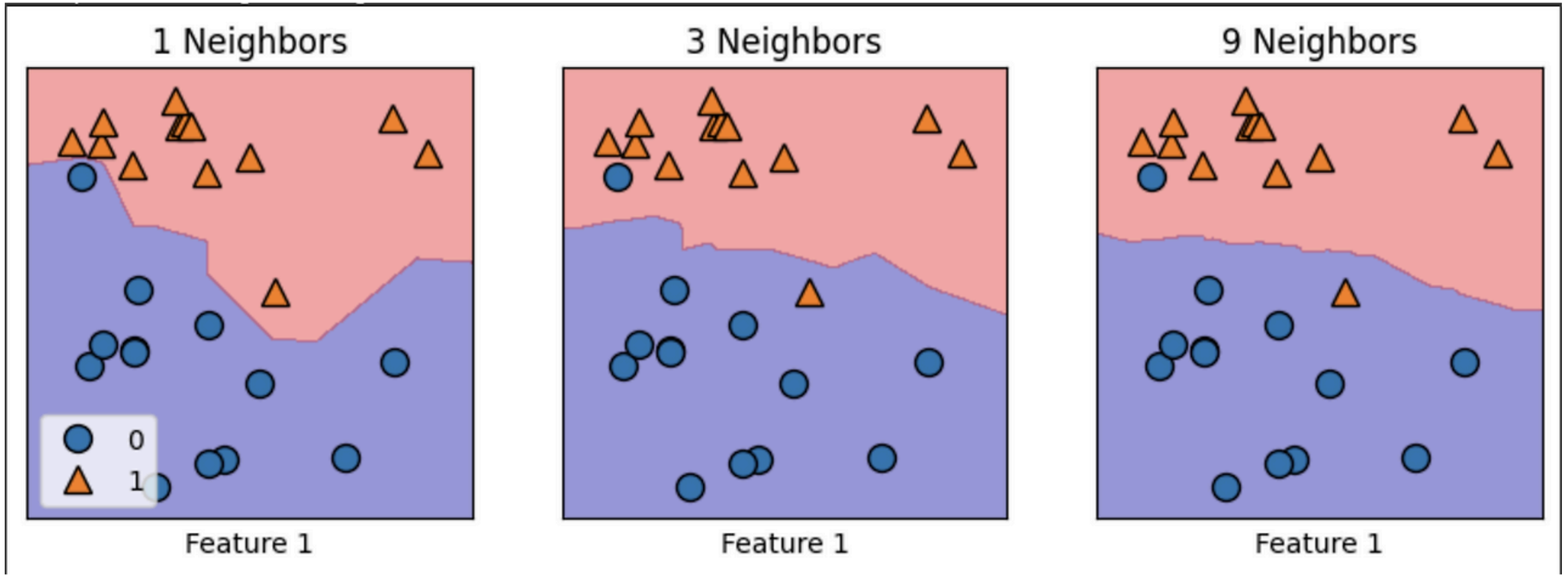
k = 7

새로운 샘플이 Class A로 분류됨.

k값 정할 때 유의사항

- 동점이 되지 않도록 k값은 홀수로 지정
- k값이 너무 크면 미세 경계 부분 분류가 안됨.
  ➡ **underfitting**
- k값이 너무 작으면 이상치 영향을 많이 받고 패턴이 직관적이지 않음.
  ➡ **overfitting**

k값에 따른 결정 경계 변화



k값이 작으면 경계가 복잡하고 구불구불해 지며(Overfitting), k값이 커지면 경계가 단순해지고(Underfitting) 집단간 세밀한 차이가 무시된다.

경계가 매끄러우며 군집의 패턴을 정확히 파악하는 것이 이상적임.

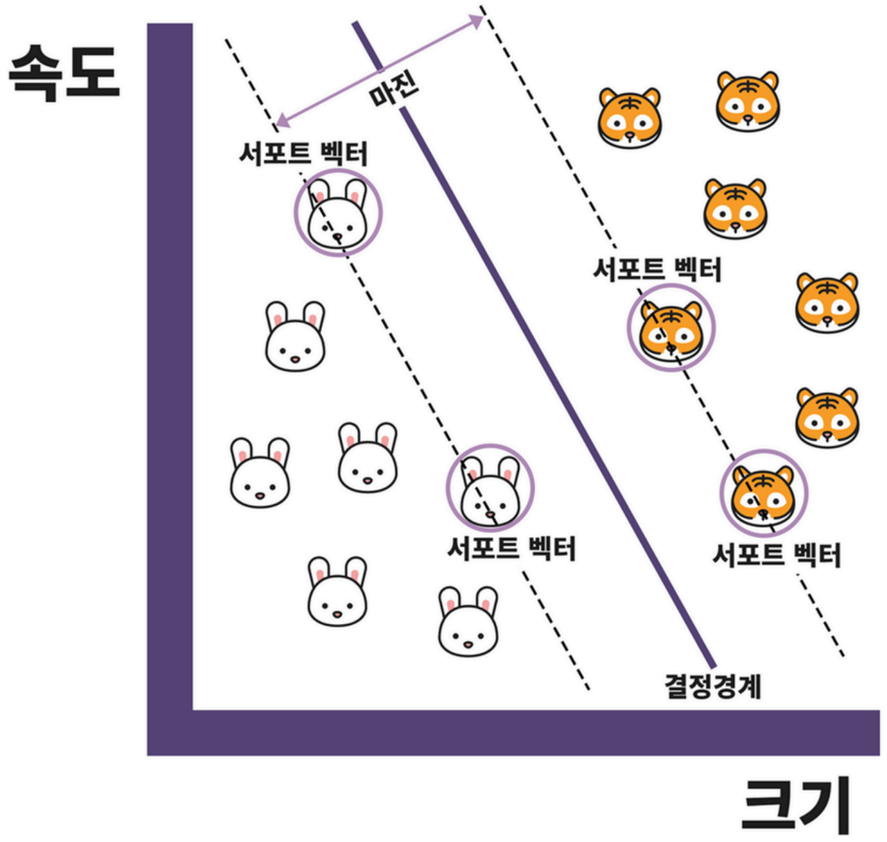
통계적으로 데이터수 N에 대해 N의 제곱근(루트N)부터 탐색 시작

Support Vector Machine(SVM)

- 딥 러닝이 등장하기 전까지 가장 성능 좋은 머신러닝 알고리즘
- 두 벡터(Support Vector)를 선정하고 중간에 결정 경계를 구하고, 그것을 중심으로 클래스를 분류하는 것
- 두 클래스로부터 최대한 멀리 떨어져 있는 결정 경계를 찾는 분류 기법
- 선형이나 비선형 분류, 회귀, 이상치 탐색에 사용할 수 있음.
- 데이터셋이 작거나 중간 크기에 적합하며 복잡한 분류에 잘 맞음.

SVM의 아이디어

데이터가 어느 카테고리에 속할지 판단하기 위해 가장 적절한 경계를 찾는 선형 모델



결정경계(초평면)

집단을 분할하는 기준이며, 마진의 중앙을 통과하는 경계

서포트벡터

경계상에 있는 데이터들

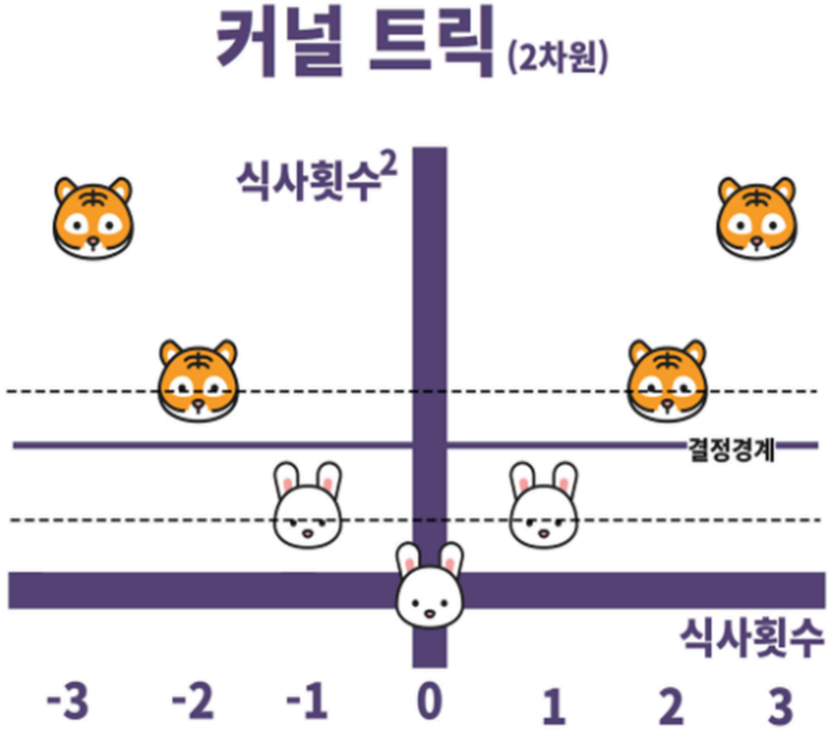
마진

결정경계와 벡터 사이의 거리

→ **유클리드 거리 공식**과 **점과 직선 사이의 거리를 구하는 공식**으로 두 벡터와 결정 경계를 구함.

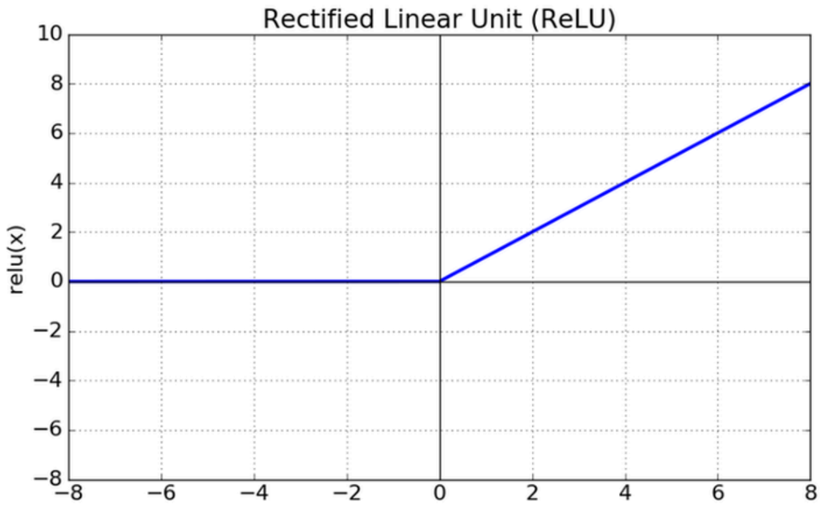
커널 트릭

- 저차원 공간에서는 선형으로 분리되지 않는 데이터를 고차원의 공간으로 직접 보내지 않고도, 마치 고차원에 있는 것처럼 계산하여 데이터를 분류하는 머신러닝 기법



Activation Function의 종류

ReLu(Rectified Linear Unit)



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

특징

- "Rectified(정류)"는 원래 전기공학 용어로, 교류(AC)에서 음의 전류를 잘라내 단방향으로 만드는 것을 의미
- x > 0이면 기울기가 1, x < 0이면 기울기가 0이다.
- sigmoid나 tanh에 비해 연산 비용이 적고, 구현이 간단하다.
- sigmoid나 tanh에 비해 학습 속도가 빠르다.
- 가장 많이 사용된다.

단점

- **Dying ReLu** : x < 0인 값에 대해 기울기가 0이므로 뉴런들이 죽을 수 있다.
- Dying Relu 보완을 위해 **Leakly ReLu** 제안!

PyTorch와 Numpy의 차이점

자동미분(Autograd)

- 텐서에 대한 모든 연산의 미분값을 자동으로 계산해주는 강력한 기능인 '자동 미분(Autograd)'을 내장
- 경사 하강법(Gradient Descent)을 구현하는 데 사용
- 텐서의 미분값을 계산하고 싶다면, 텐서를 생성할 때 **requires_grad=True** 옵션을 설정

Autograd 동작원리

- 계산 그래프를 구축하여 작동
- **backward() 메서드**로 미분 값 계산
- PyTorch는 계산 그래프를 역전파하며 각 텐서의 미분값을 자동으로 계산하며 미분값은 **.grad**에 저장

```
import torch

# x 텐서 정의 (미분 계산을 위해 requires_grad=True)
x = torch.tensor(2.0, requires_grad=True)

# 계산 그래프를 생성하는 연산
y = x**2
z = 3 * y

# z를 x에 대해 미분
# dz/dx = 3 * (2x) = 6x
# x=2 이므로, dz/dx = 12
z.backward()

# x의 미분값(기울기) 확인
# z에 대한 x의 기울기(gradient)
print(f"x의 미분값(dz/dx): {x.grad}") # 출력: 12.0
```

Autograd 예제

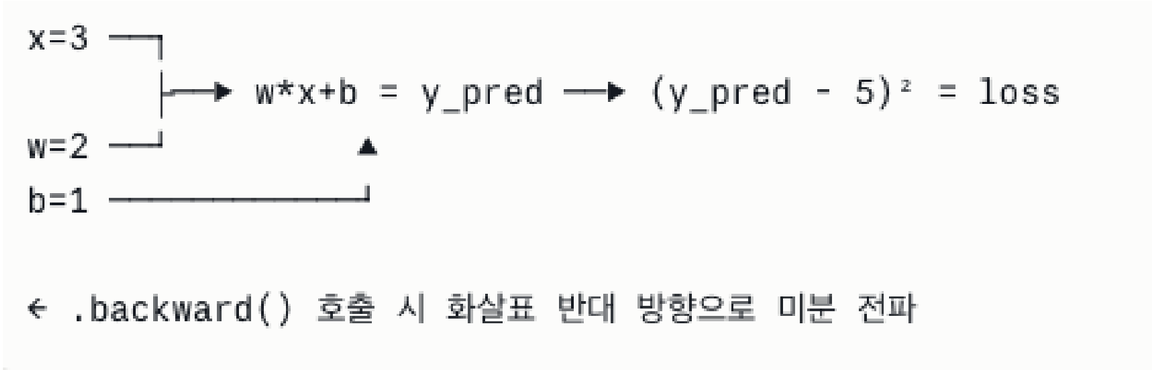
```
w = torch.tensor(2.0, requires_grad=True)   # 학습할 파라미터
b = torch.tensor(1.0, requires_grad=True)

x = torch.tensor(3.0)   # 입력값 (미분 불필요)

# 순전파: 계산 그래프가 자동으로 생성됨
y_pred = w * x + b       # y = 2*3 + 1 = 7
loss   = (y_pred - 5) ** 2  # (7-5)² = 4

# 역전파: 그래프를 거슬러 올라가며 미분 계산
loss.backward()

print(w.grad)   # d(loss)/dw = 2*(y_pred-5)*x = 2*2*3 = 12
print(b.grad)   # d(loss)/db = 2*(y_pred-5)*1 = 2*2   =  4
```



모델 성능 평가 지표

회귀

- 평균 절대 오차(MAE, Mean Absolute Error)
- 평균 제곱 오차(MSE, Mean Squared Error)
- 평균제곱근 오차(RMSE, Root Mean Squared Error)
- 결정계수(R^2, R-squared)

분류

- 정확도(Accuracy), 정밀도(Precisioni), 재현율(Recall)
- F1점수(F1 score), AUC

회귀에 대한 성능 평가 지표

R^2(R-squared, 결정 계수)

- 회귀 모델의 성능을 평가
- 모델이 평균 대비 얼마나 오차를 줄였는가를 확인
- 1에 가까우면 모델이 데이터의 변동을 완벽하게 설명한다는 의미

$$R^2 = 1 - \frac{SSR}{SST} = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2}$$

분류에 대한 성능 평가 지표

혼동행렬(confusion matrix)

		모델 예측값	
		Positive (양성)	Negative (음성)
실제값	Positive (양성)	True Positive (TP)	False Negative (FN)
	Negative (음성)	False Positive (FP)	True Negative (TN)

분류에 대한 성능 평가 지표

정확도(Accuracy)

- 모든 예측 중 올바르게 예측된 비율
- 가장 직관적인 성능지표
- 한계 : 학습 데이터에 클래스 불균형이 있는 경우 오해의 소지 있음.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

		모델 예측값	
		Positive	Negative
실제값	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

분류에 대한 성능 평가 지표

정밀도(precision)

- 모델이 양성으로 예측한 사례 중 실제로 양성인 사례의 비율
- 즉, 모델이 얼마나 정밀하게 양성 사례를 예측했는지 나타냄.

$$Precision = \frac{TP}{TP + FP}$$

		모델 예측값	
		Positive	Negative
실제값	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

분류에 대한 성능 평가 지표

정밀도(precision)

- 거짓 양상(FP)의 수를 줄이는 것이 중요한 문제에서 응용

예) 스팸 메일 필터링 : 정상 메일을 스팸으로 분류하는 것을 피해야 함.

금융 사기 탐지 : 정상 거래를 사기로 판단하는 것을 피해야 함.

- **한계** : 거짓 음성(FN) 값을 무시함.

		모델 예측값	
		암	암X
실제값	암	9	20
	암X	1	911

$$Precision = \frac{TP}{TP + FP} = \frac{9}{9 + 1} = 0.9$$

분류에 대한 성능 평가 지표

재현율(Recall)

- 실제 양성 사례들 중 모델이 양성으로 올바르게 예측한 사례의 비율
- 즉, 모델이 실제 양성 사례를 얼마나 잘 재현했는지 나타냄.
- 거짓 음성(FN)의 수를 줄이는 것이 중요할 때 유용

예) 암진단 : 암을 암이 아니라고 오진하는 것을 피해야

보안시스템 : 보안 위협이 없는 것으로 잘못 판단하는 것을 피해야

$$Recall = \frac{TP}{TP + FN}$$

		모델 예측값	
		Positive	Negative
실제값	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

분류에 대한 성능 평가 지표

재현율(Recall)

- 한계 : 거짓 양성(FP) 무시

		모델 예측값	
		Spam	Ham
실제값	Spam	9	1
	Ham	20	911

$$Recall = \frac{TP}{TP + FN} = \frac{9}{9 + 1} = 0.9$$

정밀도 - 재현율 트레이드 오프(Trade - off)

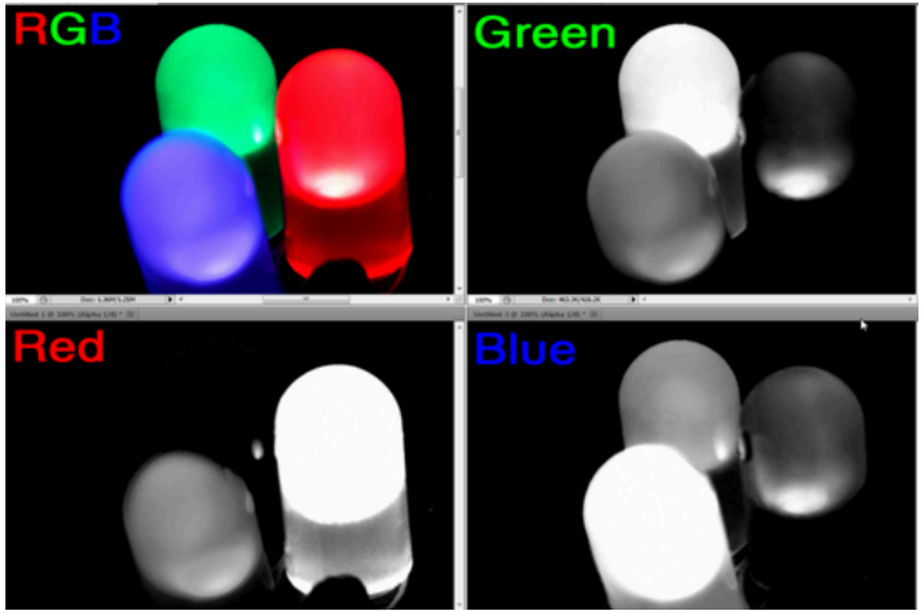
- 정밀도와 재현율은 하나가 증가하면 하나가 감소하는 Trade-off관계
- 정밀도를 높이기 위해 매우 보수적으로 양성을 예측하면 실제 양성인 사례를 높치게 되어 재현율 감소
- 재현율을 높이기 위해 가능한 많은 양성 사례를 포착하려고 하면 거짓 양성이 많아져 정밀도 감소

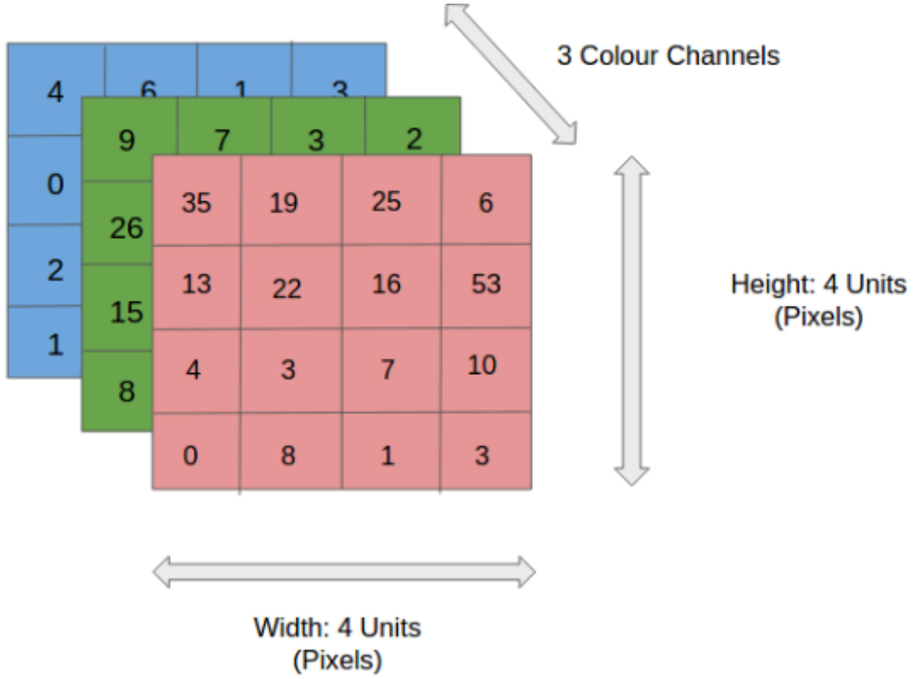
모델성능평가지표 예제 - 손글씨 데이터

https://colab.research.google.com/drive/1Hh4hjY8tzoMCA5vNm6XBT9hXuD_cVL_d?usp=sharing

이미지의 구조

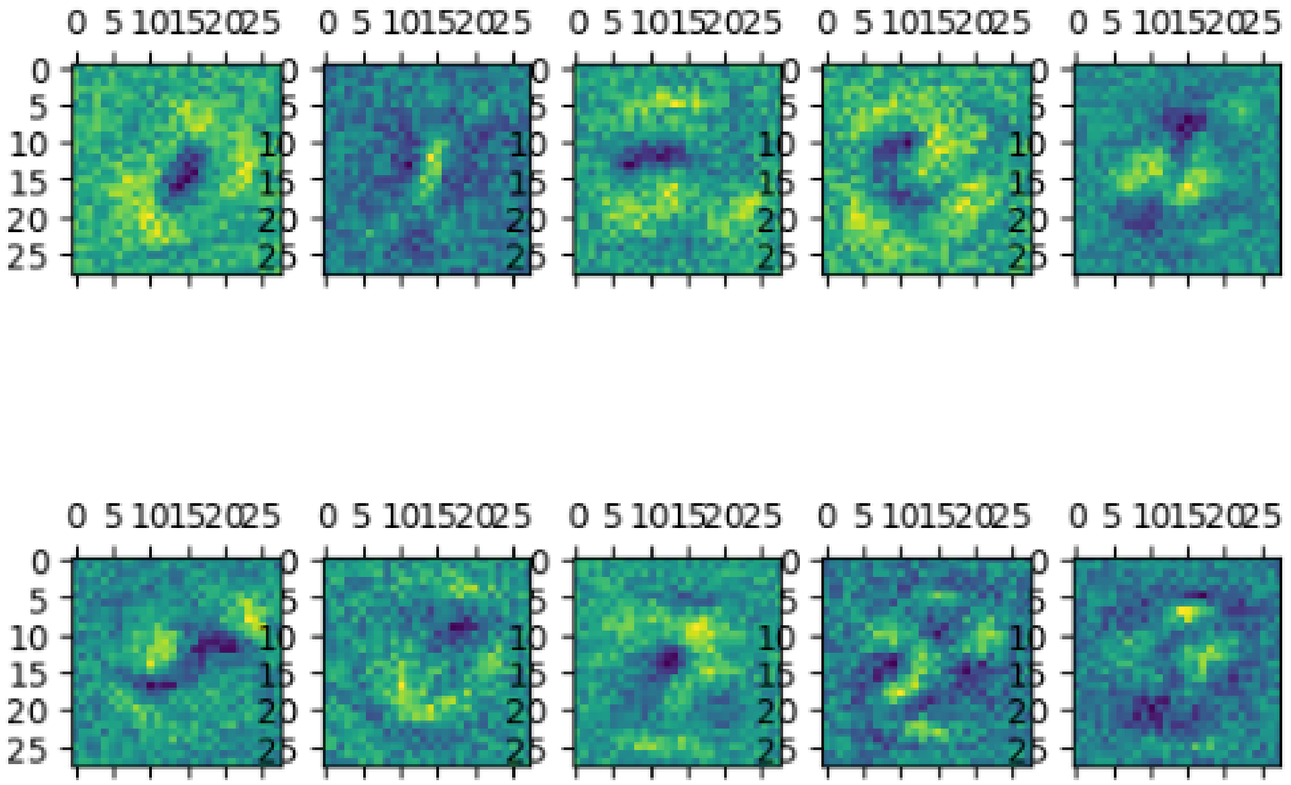
- 컴퓨터가 이미지를 인식하는 방법으로 0 ~ 255 사이의 숫자 형태로 인식
- 숫자는 색의 명암, 0에 가까울수록 어두운 색
- 흑백 이미지는 채널이 1개, 칼라 이미지는 채널이 3개(Red, Green, Blue)로 표현





로지스틱 회귀를 이용한 MNIST 데이터

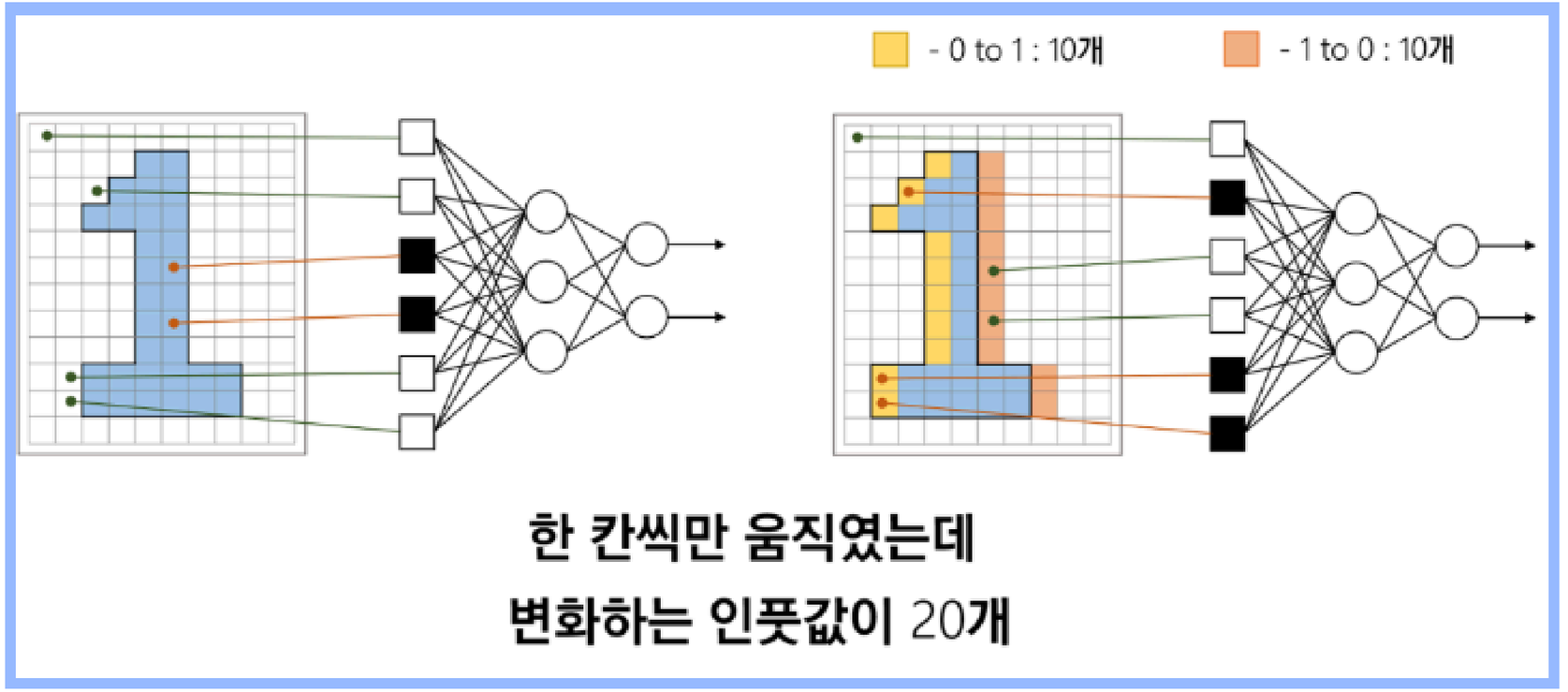
- 학습 후 아래와 같은 28x28 사이즈의 가중치를 가짐.
- 밝은 색은 양의 가중치, 어두운 색은 음의 가중치



CNN의 필요성

기존 알고리즘의 한계

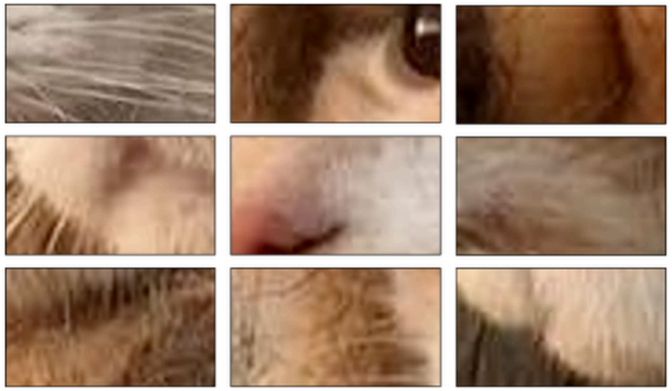
- 1을 인식하려고 할 때, 이미지가 조금만 움직여도 이미 학습한 가중치가 달라진다.



CNN의 필요성

위치나 기울기가 아닌 **이미지의 특성**을 인식하는 방법이 필요!

이미지의 특징 찾기



1단계 : 세모, 동그라미, 세모, 털

2단계 : 눈, 코, 귀, 발

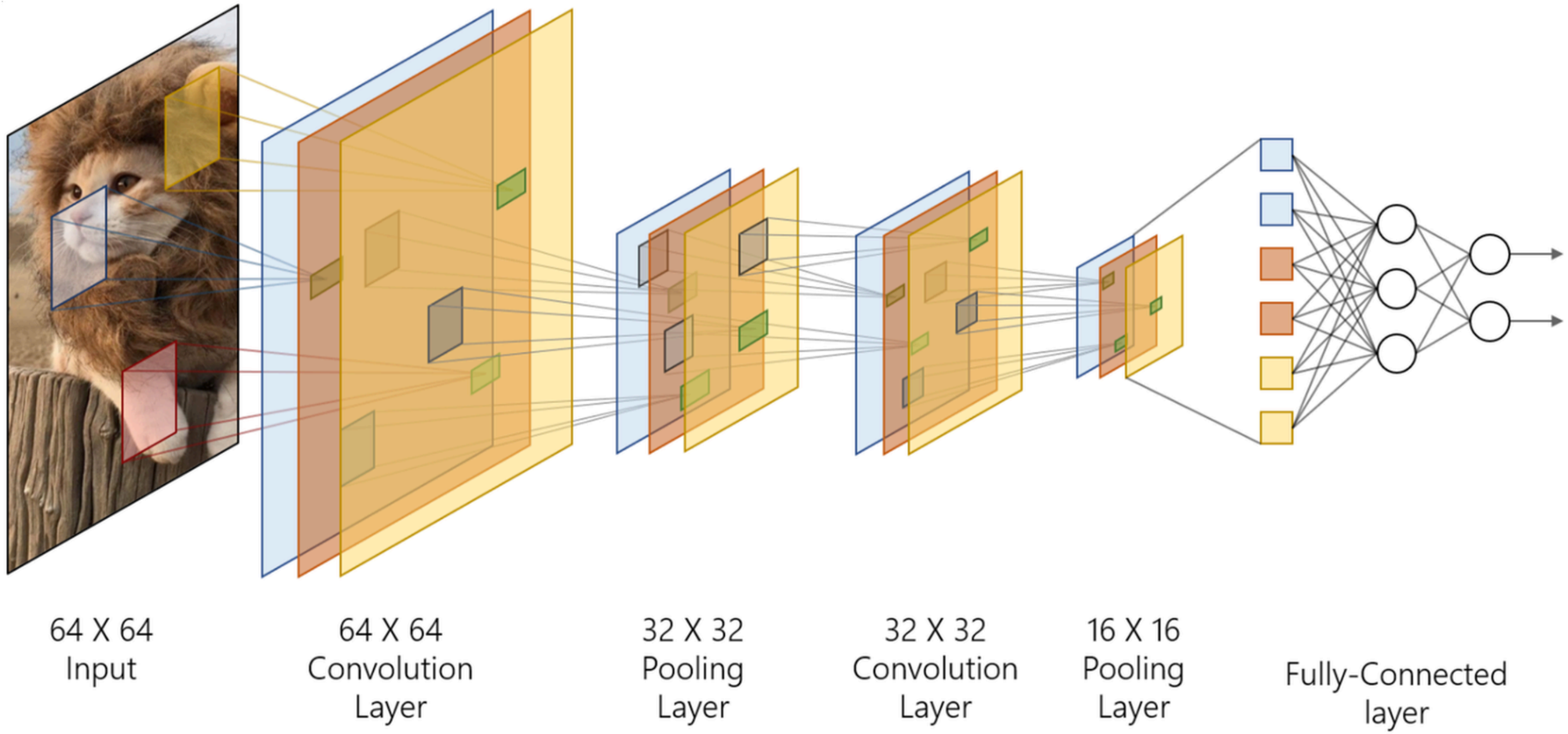
3단계 : 고양이

1단계 : 가장 기초가 되는 특징 확인

2단계 : 특징들을 조합해서 보다 복잡한 특징 존재 확인

3단계 : 물체 분류

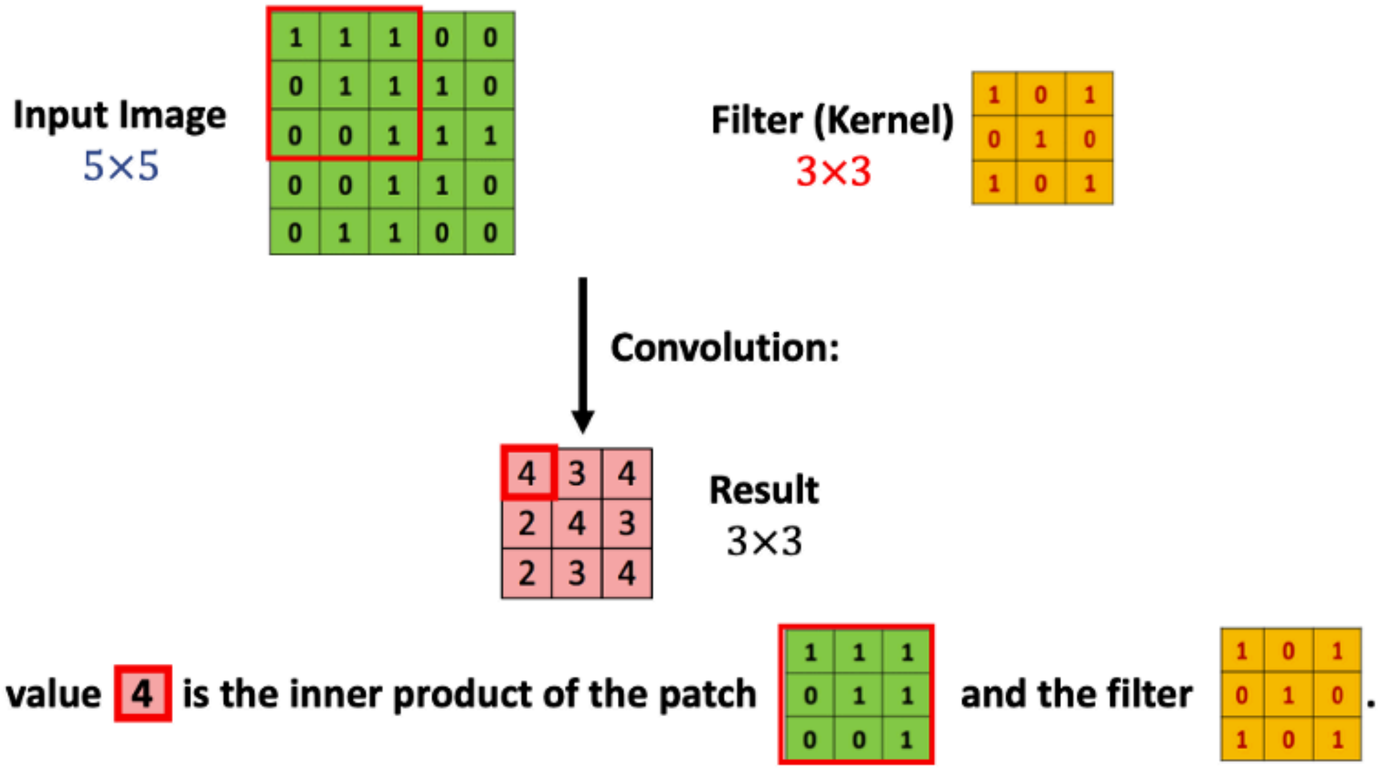
CNN의 구조



이미지를 input으로 받은 뒤 **Convolution연산**과 **Pooling 연산**을 반복하여 MLP를 거치는 것이 CNN의 기본 구조

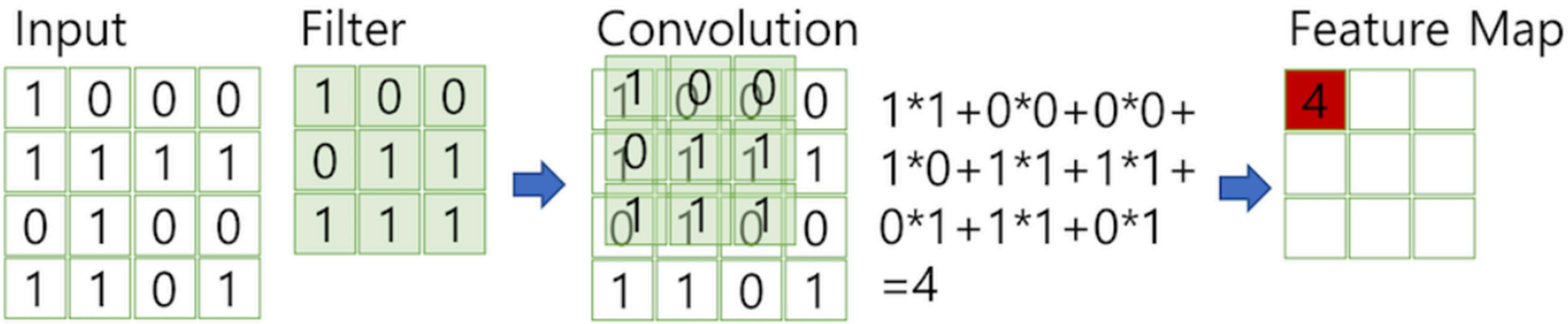
→ **이미지의 특성을 찾는 과정**

Convolution 연산



- 합성곱 연산으로 만들어진 결과를 **Feature Map**이라고 한다.
- Feature Map은 주어진 이미지에서 특징을 추출한 것이고, 합성곱 층의 최종 출력 결과를 **Activation Map**이라고 한다.

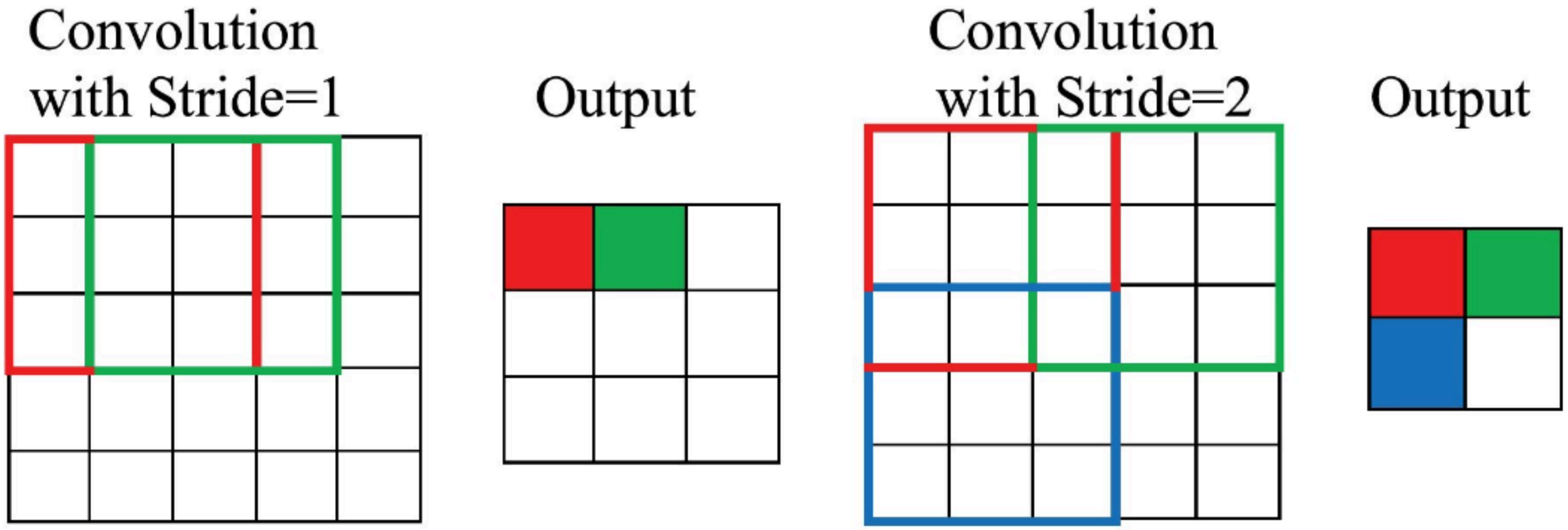
Convolution 과정



- **필터(커널)** : 이미지의 특징을 찾아내기 위한 공용 파라미터

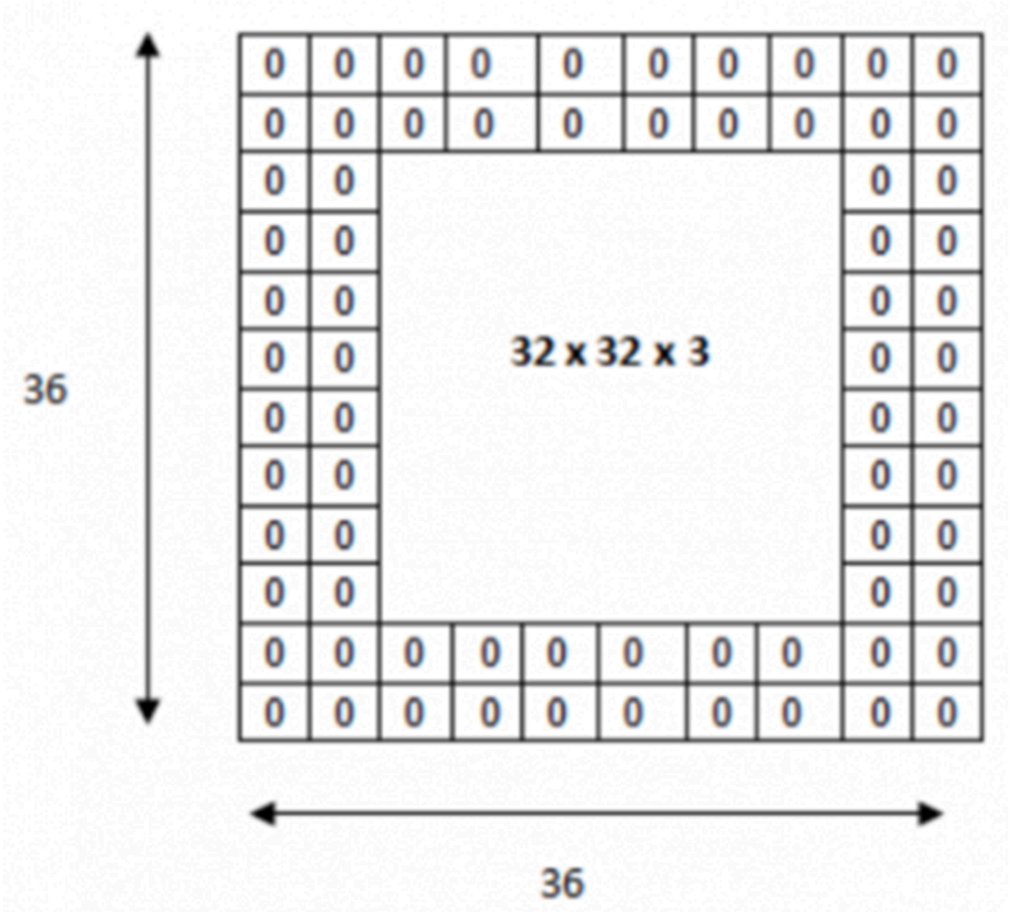
Stride

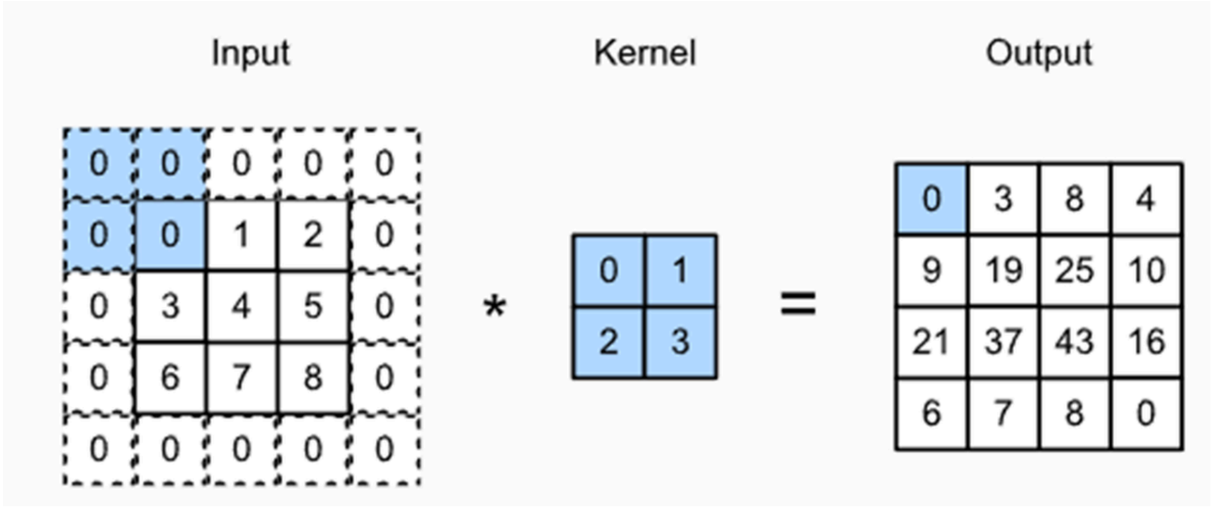
- 필터가 순회하는 간격
- convolution 연산 후 얼마나 이동할지를 결정



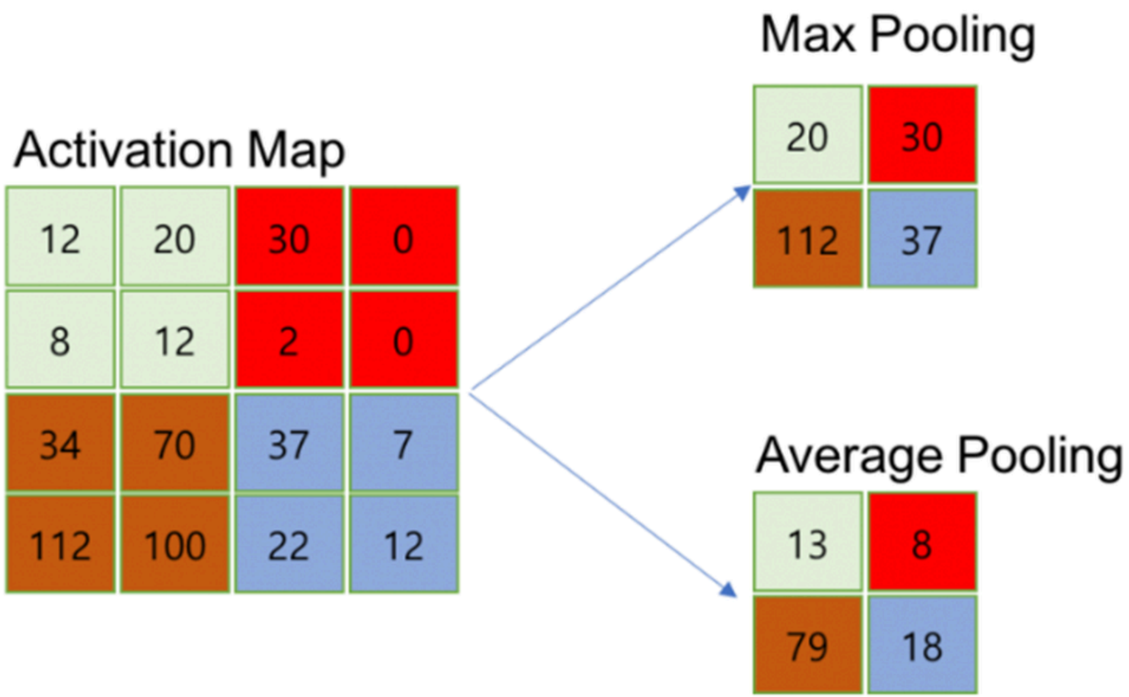
Padding

- 합성곱 연산으로 인한 데이터 손실 방지 및 경계 표현을 위한 테두리



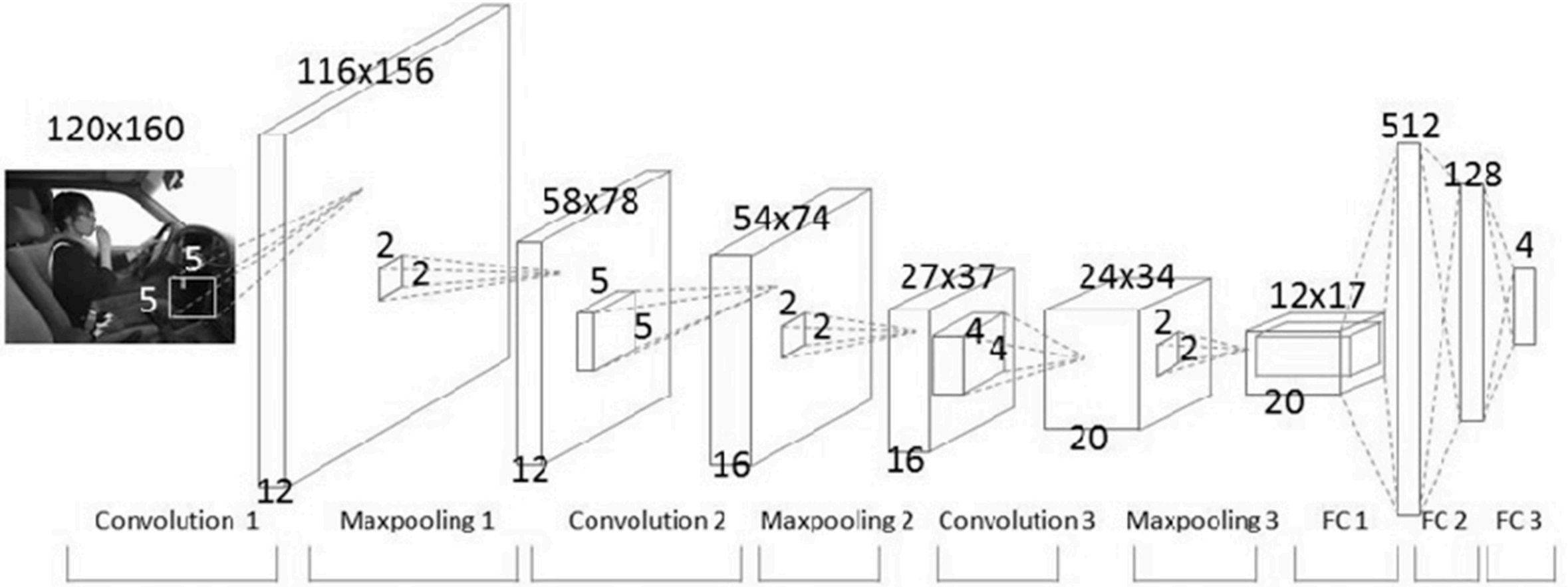


Pooling



- Convolution 연산 후 출력 Map(Activation Map)의 크기를 줄이거나 특정 데이터를 강조하기 위한 연산
- 뒤틀림, 왜곡같은 영향을 줄여주는 효과가 있음.
- 여러 픽셀을 하나의 픽셀로 줄일 때, 대표값을 구한다.
- Max Pooling을 많이 사용함.

CNN 처리 과정


120x160
116x156
58x78
54x74
27x37
24x34
12x17
512
128
4
5
5
2
2
5
5
2
2
4
4
2
2
20
20
12
12
16
16
Convolution 1
Maxpooling 1
Convolution 2
Maxpooling 2
Convolution 3
Maxpooling 3
FC 1
FC 2
FC 3

커널사이즈, stride, padding에 따른 이미지 사이즈

$$O = \left\lfloor \frac{I - K + 2P}{S} \right\rfloor + 1$$

- O : 출력 데이터 사이즈
- I : 입력 데이터 사이즈
- K : 커널 사이즈
- P : 패딩 사이즈
- S : Stride 사이즈

만약 이미지가 28x28, 커널 사이즈 = 3x3, stride = 1, padding = 0 이면?

$$O = \left\lfloor \frac{28 - 3 + 0}{1} \right\rfloor + 1 = 26$$

- 출력 데이터는 26x26