



Artificial Intelligence Tutorial

박 경 규



학습목표

프로그래밍과 코딩의 중요성을 이해합니다.

AI 기본 개념을 이해하고 설명할 수 있습니다.

머신러닝 이론을 이해하고 주요 알고리즘을 활용할 수 있습니다.

딥러닝의 원리와 파악하고 심층신경망 모델을 사용할 수 있습니다.

자료

실습파일 https://github.com/kgpark88/deeplearning/blob/master/ai_modeling.ipynb

파이썬 <https://github.com/kgpark88/python/>

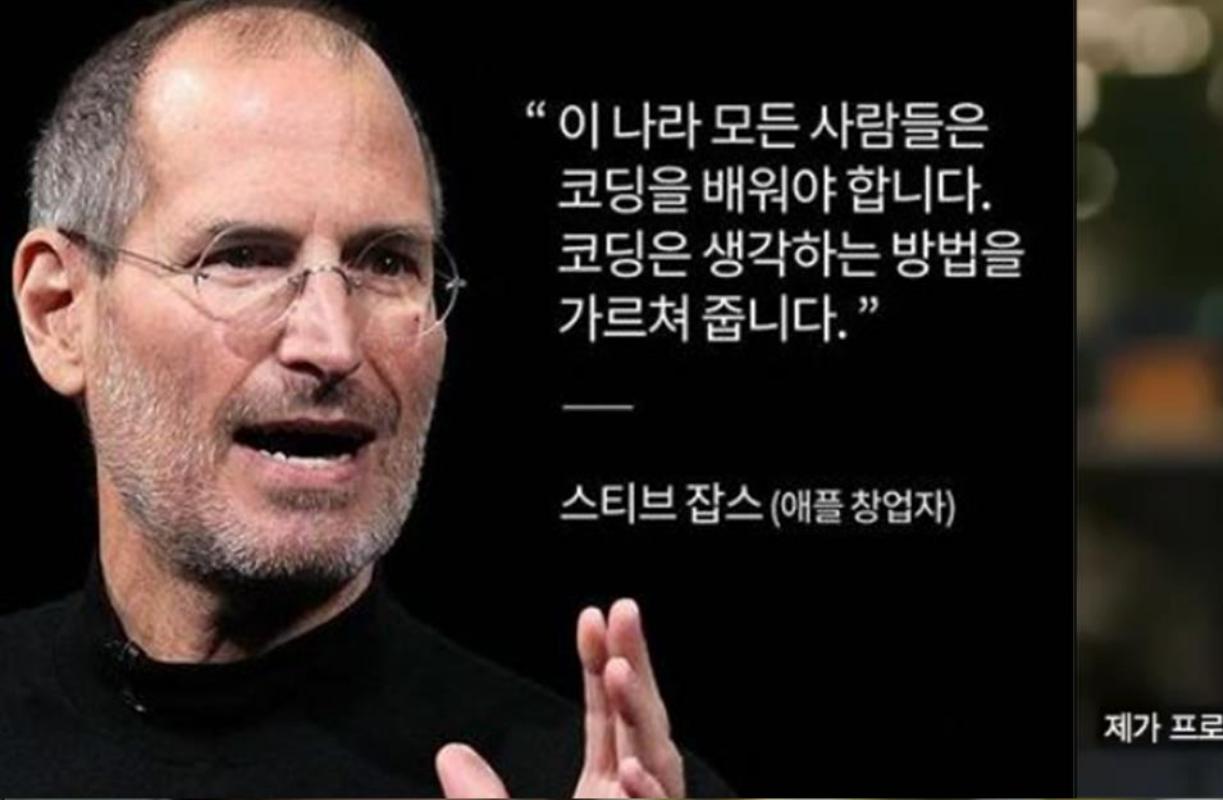
딥러닝 <https://github.com/kgpark88/deeplearning>

목 차

1. 프로그래밍을 배워야 하는 이유
2. AI 이해하기
3. AI 솔루션 개발
4. 머신러닝 이론과 주요모델
5. 딥러닝 원리와 심층신경망

1 프로그래밍을 배워야 하는 이유





“이 나라 모든 사람들은
코딩을 배워야 합니다.
코딩은 생각하는 방법을
가르쳐 줍니다.”

스티브 잡스 (애플 창업자)



MARK
CREATED facebook

제가 프로그래밍을 처음 배우기 시작한 이유는 컴퓨터 공학을 완습하거나 그분야의 최고가 되고 싶어서가 아니라 저를 포함한 저희집 남매들이 서로 같이 재밌게 할 수 있는 뭔가를 만들고 싶었거든요.



차세대 프로그래머는 미래의 마법사입니다. 다른 사람과 비교했을 때 마치 마법 능력이 있는 것처럼 보여질 거예요.

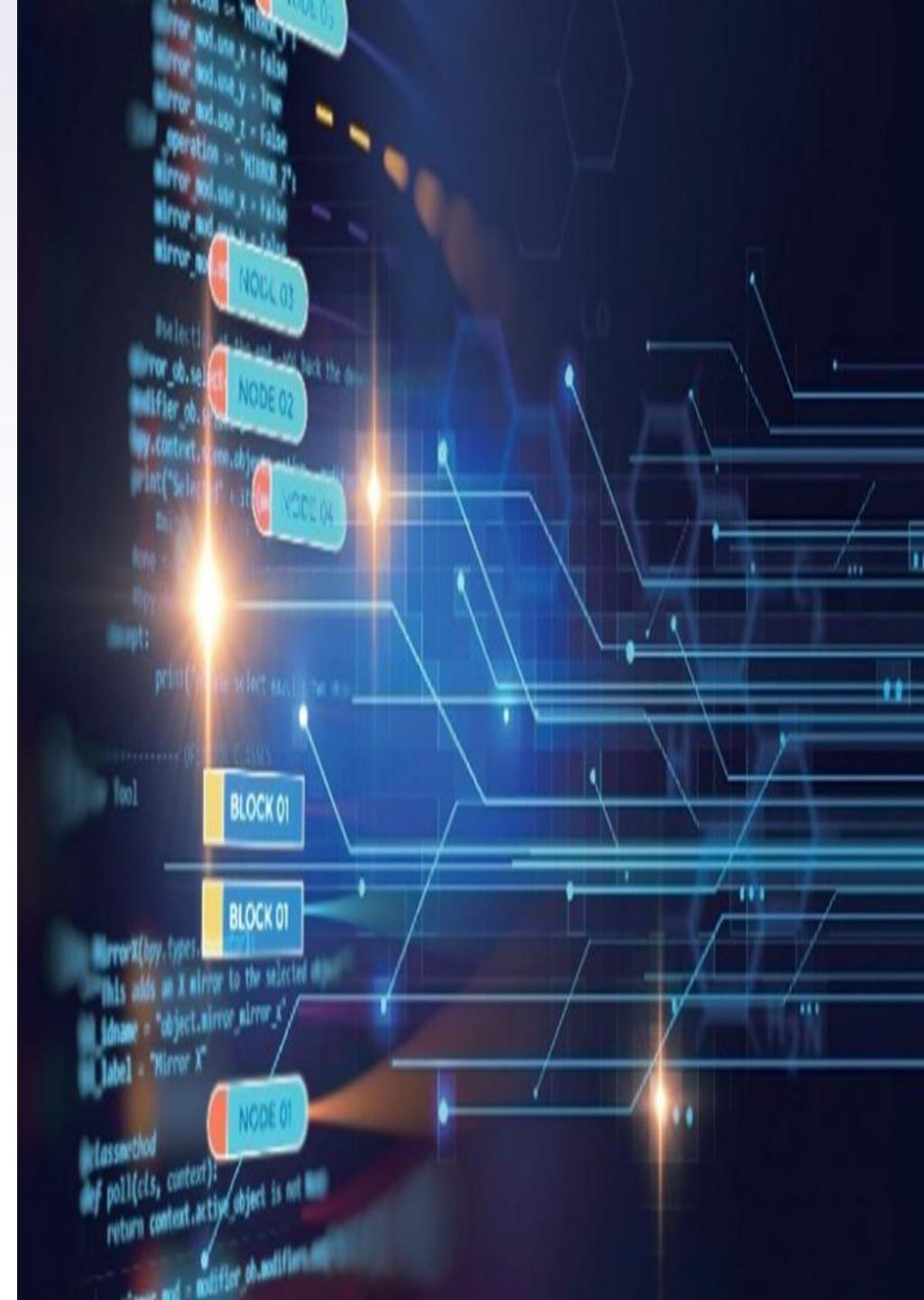


HADI
CREATED CODE.ORG

목표가 돈을 많이 버는 것이건 세상을 바꾸는 것이건 상관없이
컴퓨터 프로그래밍 능력은 당신에게 어마어마한 힘을 제공할 것입니다.

프로그래미란?

- 컴퓨터 프로그램은 특정 문제를 해결하기 위해 고안된 **특정 작업을 수행하기 위한 일련의 명령문의 집합체**
- 스마트폰, 태블릿 등에서는 ‘앱’이라는 용어를 사용
- 소프트웨어는 하드웨어의 반대 개념으로서의 의미이지만, 일반적으로는 프로그램과 같은 의미
- 프로그래밍은 주어진 문제를 해결하기 위해 컴퓨터 프로그램을 만들고 실행하는 전 과정
절차 : 문제분석 → 입출력설계 → 알고리즘설계
→ 코딩 → 프로그램실행



코딩이란?

- 코딩[Coding]은 컴퓨터 프로그램 언어로 프로그램을 작성하는 것 입니다.
좁은 의미의 프로그래밍입니다.
- 영어를 배우는 것 만큼 코딩을 배우는 게 중요합니다.
- 프로그램 언어 규칙에 따라 글을 쓰는 것만으로 컴퓨터에게 원하는 일을 시킬 수 있습니다.
- 코딩교육을 통해 소프트웨어시대의 경쟁력을 갖출 수 있게 될 것입니다.

코딩은 문제해결을 위한
절차적 사고와 논리적 사고력을
키워 줍니다.



코딩을 배워야 하는 이유

- 소프트웨어의 발전, 특히 소프트웨어를 만드는 소프트웨어 개발툴의 발전으로
모든 사람이 코딩을 할 수 있는 시대가 되었습니다.
- 코딩은 프로그래머 직종에 한정돼 있지 않으며,
데이터 분석, 과학, 의학, 엔지니어링, 영업, 농작물 재배 등에도 코딩 능력이 필요합니다.
- 전 산업이 소프트웨어를 이용해서 자동화 지능화로 혁신하고 있으며,
모든 소프트웨어는 코딩으로 만들어집니다.



You Can Create Anything You Want



Instant Scalability



Good Income

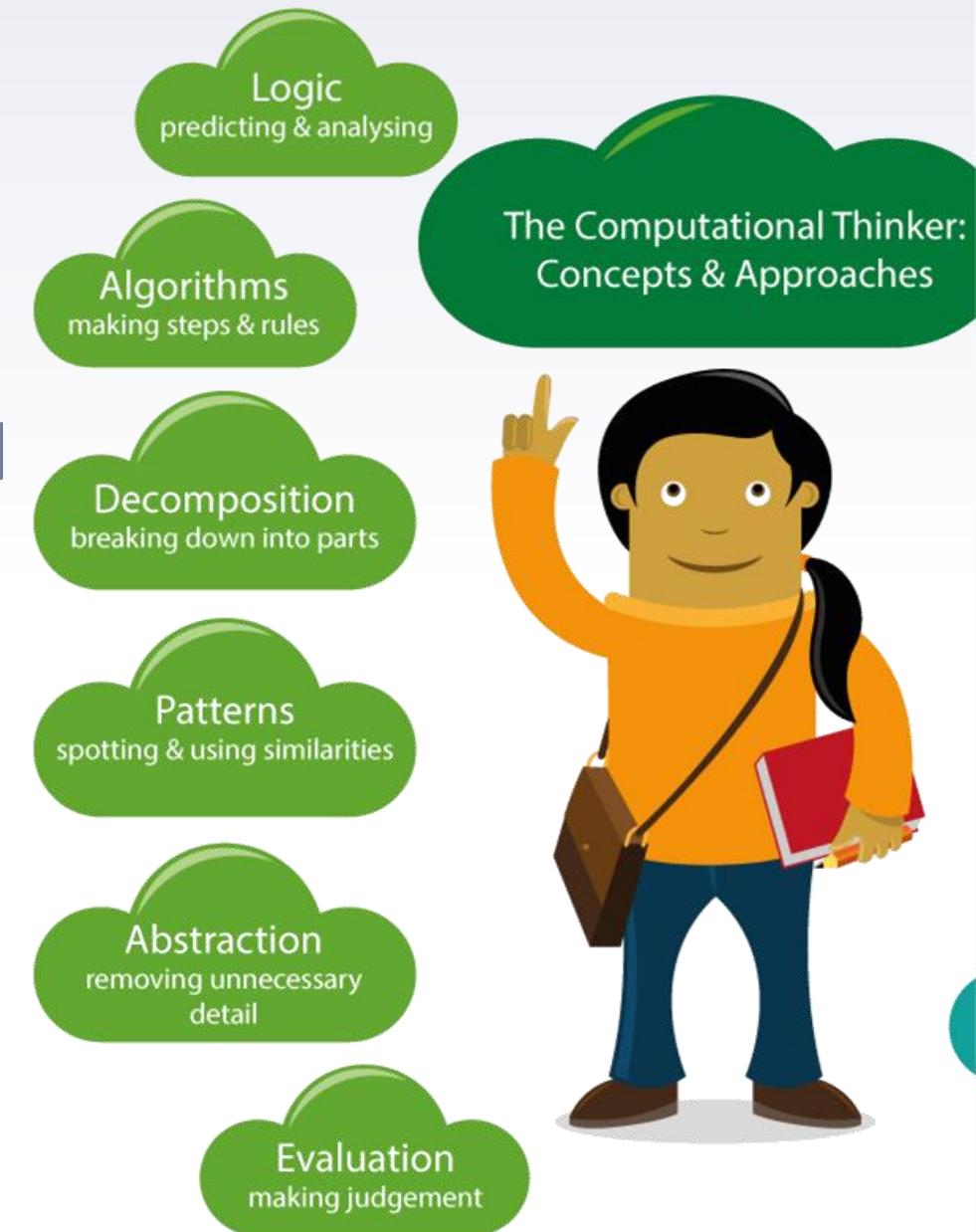
컴퓨팅 사고



- 2006년 미국 카네기멜런 대학교 교수인 지넷 윙 교수가 ACM 저널에 기고한 “Computational Thinking” 글에서 확산
- 컴퓨팅 사고는 컴퓨터 과학자뿐만이 아니라 누구나 갖춰야 하는 기본적인 역량
- 컴퓨팅 사고는 컴퓨터과학의 이론, 기술, 도구를 활용하여 현실의 복잡한 문제를 해결하는 사고방식
- 스크래치를 만든 MIT 미디어 랩, 영국의 BBC, 구글, 국내 교육부 등 다양한 기관과 학자들이 컴퓨팅 사고의 개념과 중요성을 논의함

컴퓨팅 사고력의 구성요소

- 문제를 컴퓨터로 해결할 수 있는 형태로 **구조화하기**
- 자료를 **분석하고 논리적으로 조직하기**
- 모델링이나 시뮬레이션 등의 **추상화**를 통해 자료를 표현하기
- 알고리즘적 사고를 통하여 해결방법을 **자동화하기**
- 효율적인 해결방법을 **수행하고 검증하기**
- 문제 해결 과정을 다른 문제에 적용하고 **일반화하기**



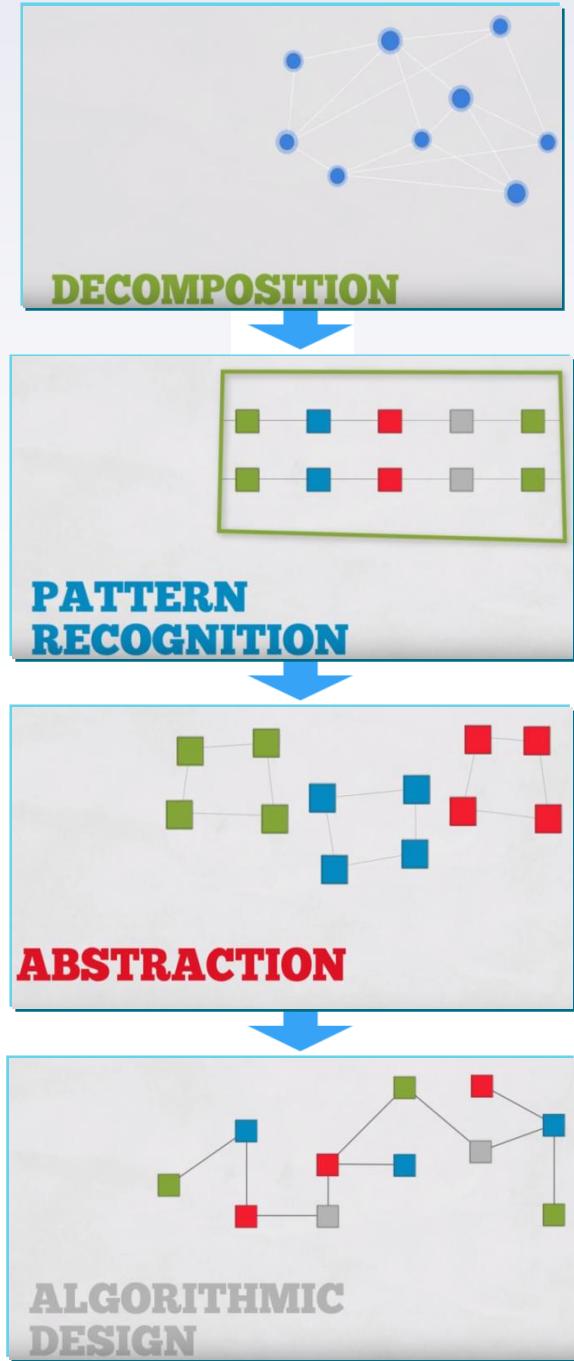
문제해결방법

① 문제 분해 : 복잡한 문제를 작은 문제로 나눕니다.

② 패턴 인식 : 문제 안에서 유사성을 발견합니다.

③ 추상화 : 문제의 핵심에만 집중하고, 부차적인 것은 제외합니다.

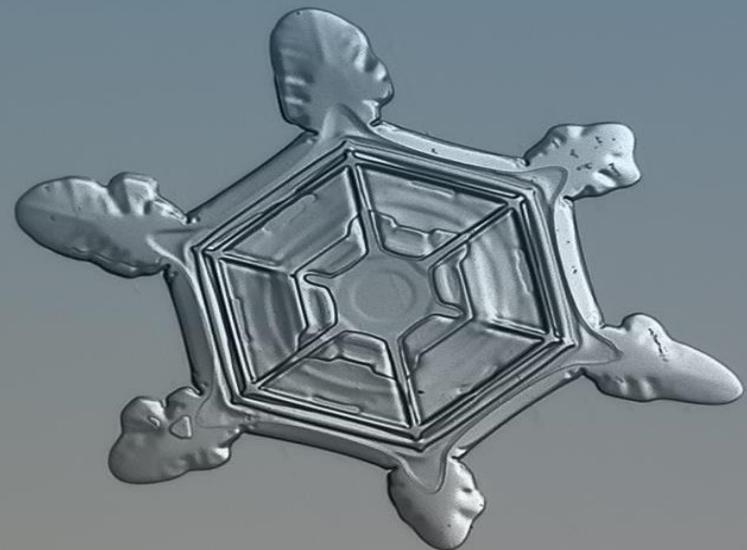
④ 알고리즘 : 이렇게 정의한 문제를 해결하는 절차입니다.



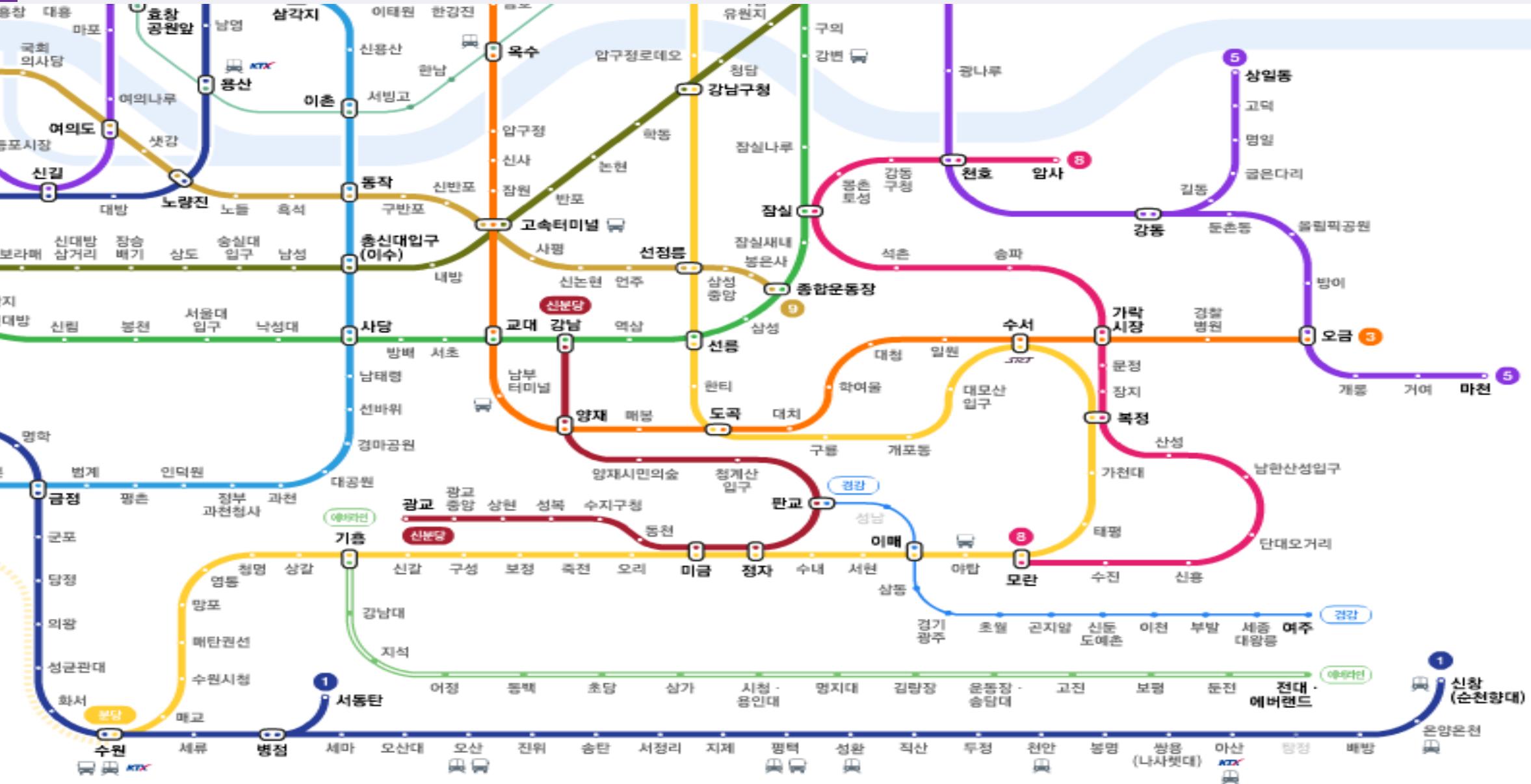
문제 분해(Decomposition)



패턴 인식(Patterns)



추상화(Abstract)



알고리즘(Algorithms)



라면 맛있게 끓이는 법

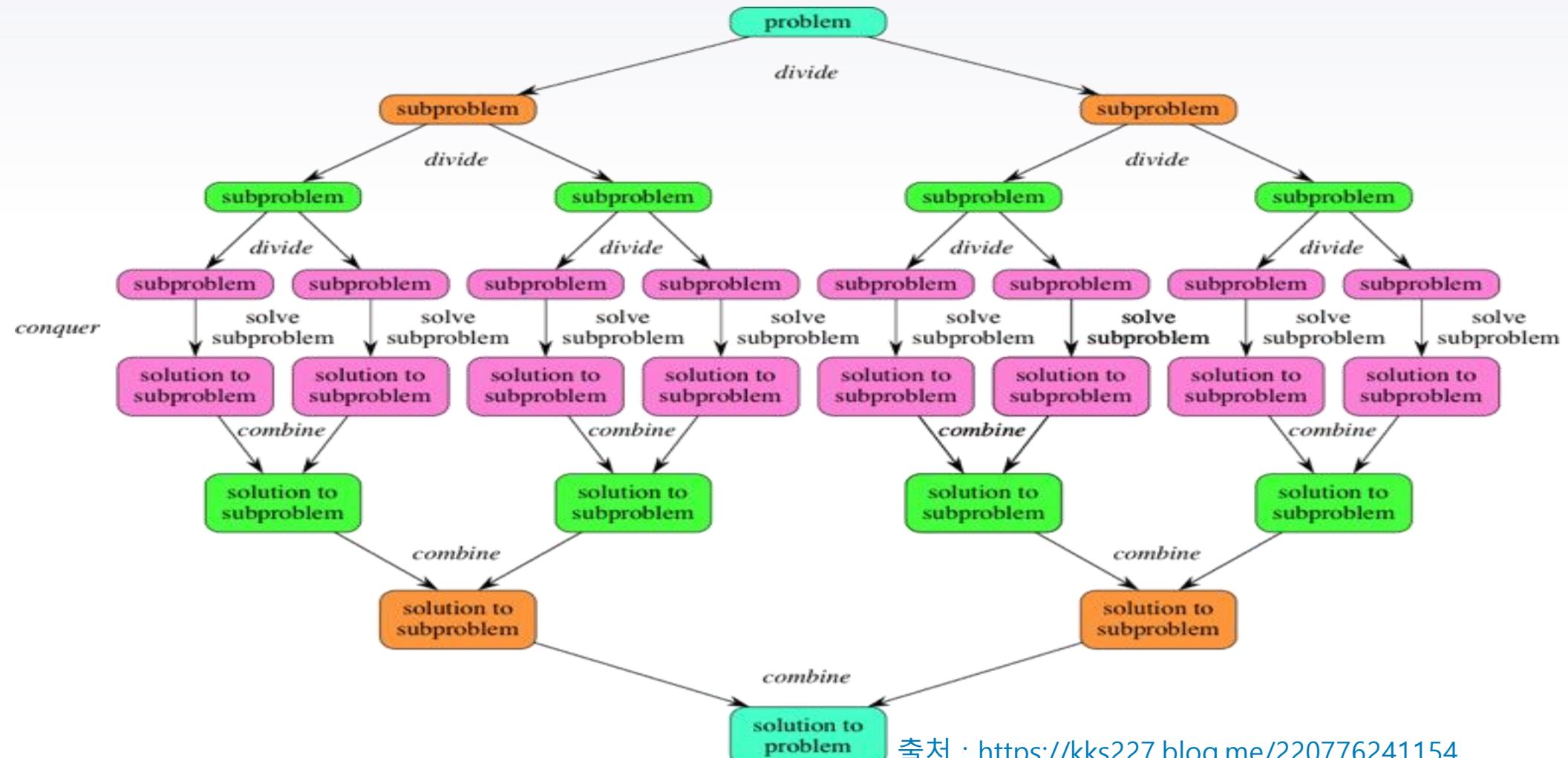
물의 양	550ml : 550ml 생수병 한 병을 다 끓고 10분의 1을 더 끓는다
조리시간	물이 끓으면 건더기스프 등을 넣고 4분30초 더 끓인다.
불의 세기	양은냄비를 사용해 센 불에 끓인다

[이미지 출처 : <http://news.hankyung.com/article/2013031367871>]

알고리즘설계

▪ 분할 정복 알고리즘(Divide and conquer algorithm)

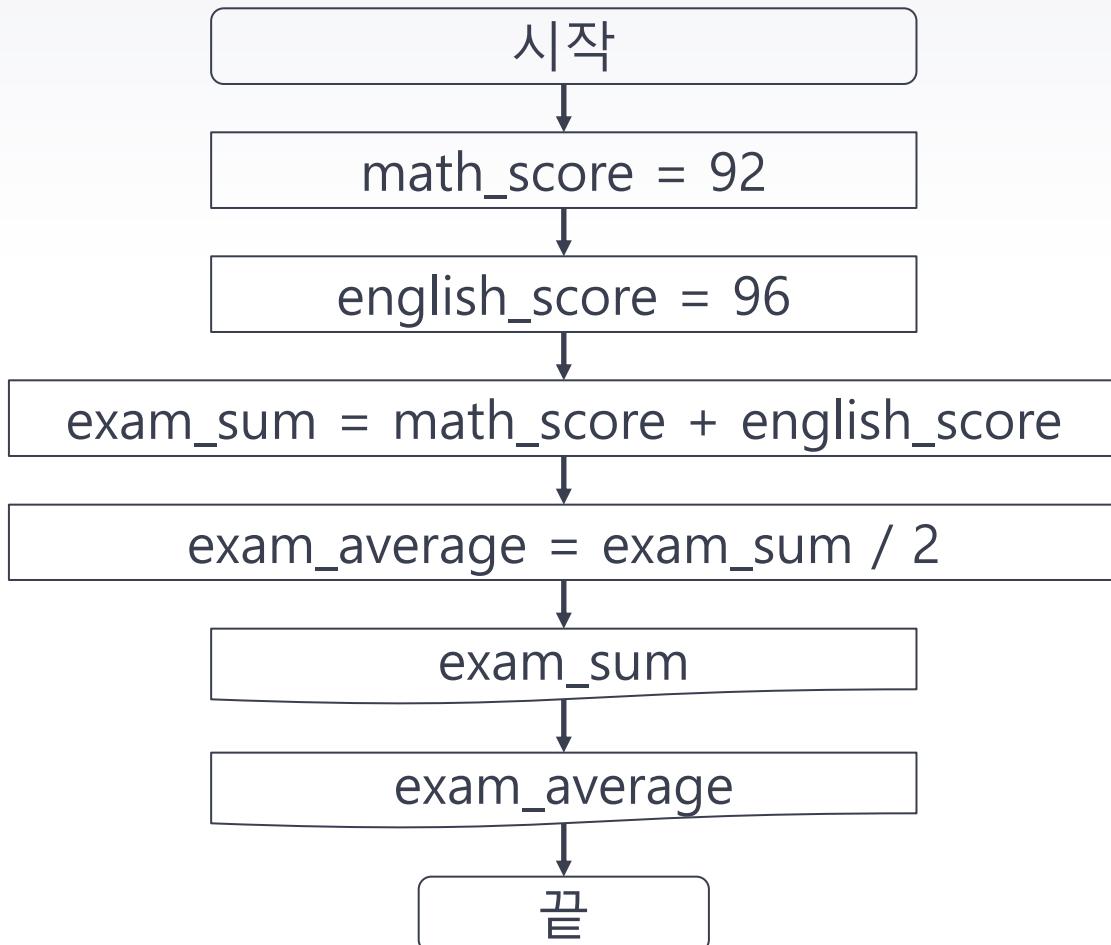
기본이 되는 해결방법으로 크고 방대한 문제를 작은 문제로 분할하여 해결



알고리즘의 구현, 코딩

■ 성적 총점과 평균 구하기

순서도



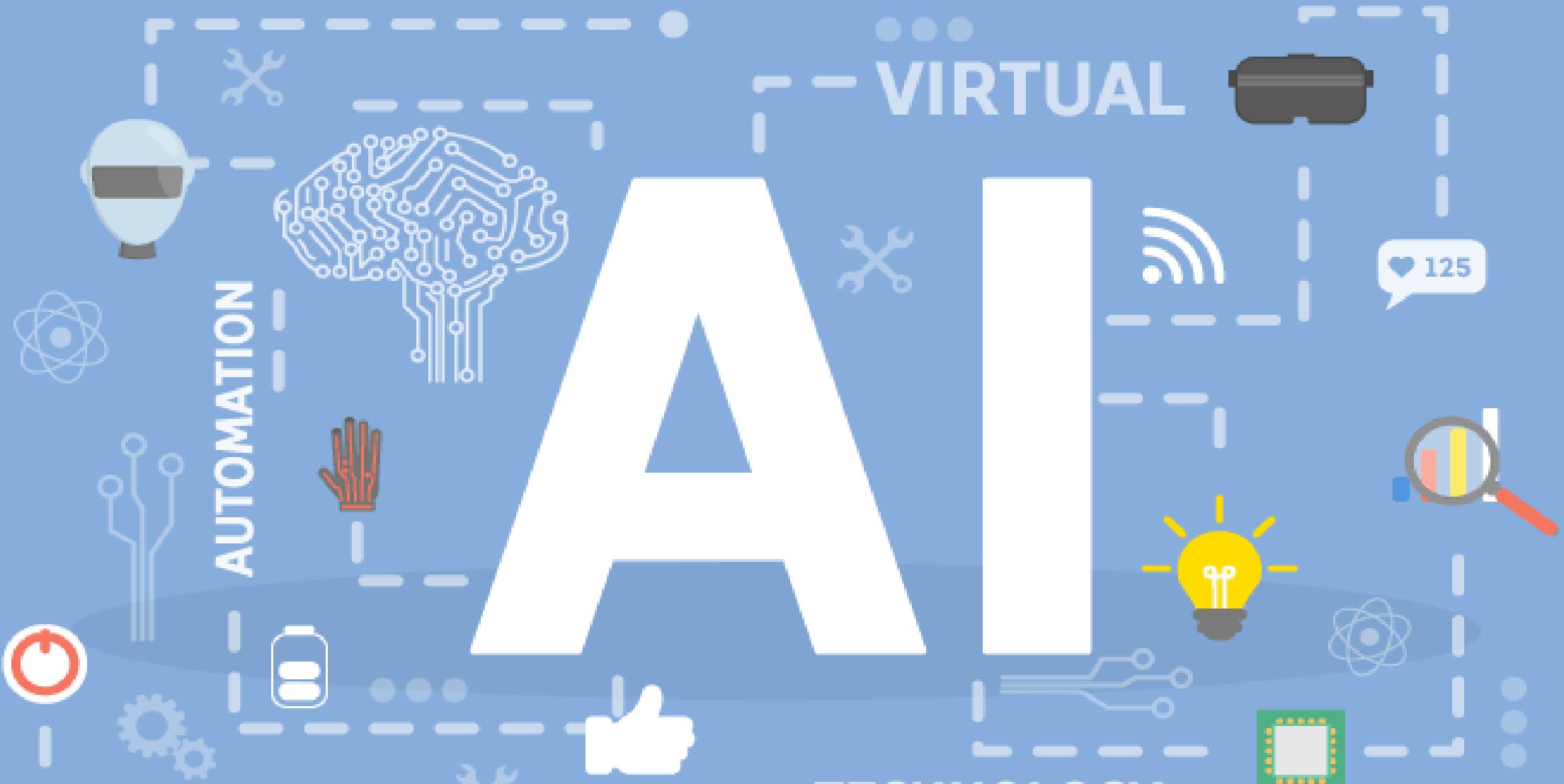
코딩

```
math_score = 92
english_score = 96
exam_sum = math_score + english_score
exam_average = exam_sum / 2
print("총점", exam_sum)
print("평균", exam_average)
```

실행결과

```
총점 188
평균 94.0
```

2 AI 이해하기



iamai

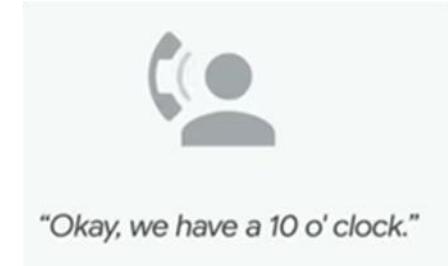
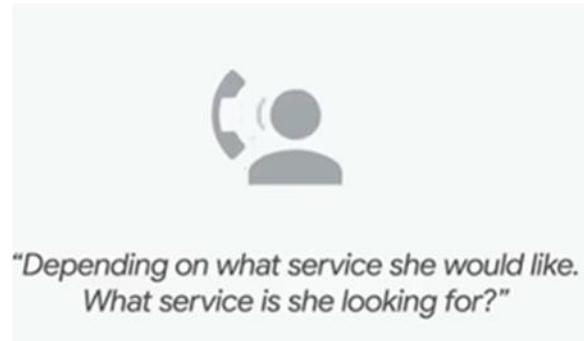
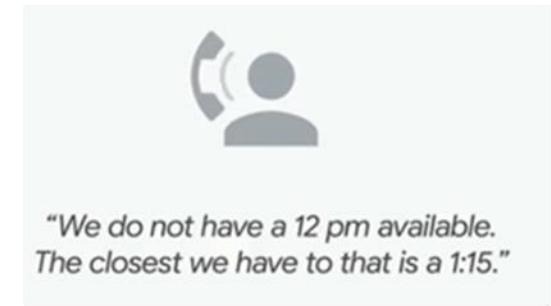
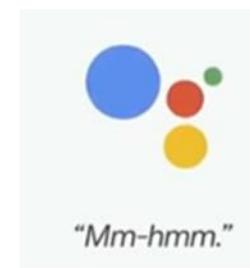
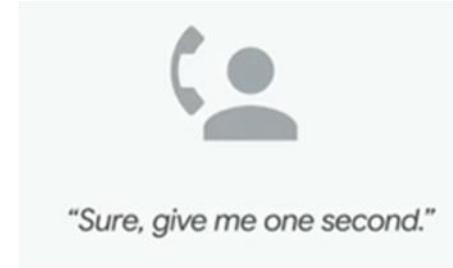


"머신러닝과 인공지능은
불과 몇 년 전만해도
우리가 상상할 수 없었던
역량을 열어주었습니다"

Sundar Pichai, CEO, Google Inc.



AI 비서



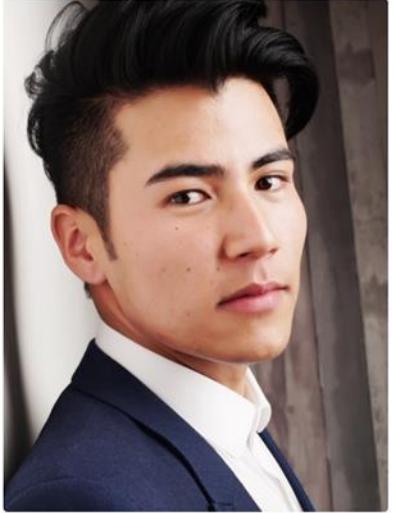
AI 작곡

A screenshot of a digital audio workstation (DAW) interface, likely Ableton Live, used for composing cinematic music. The top section shows a score editor with multiple tracks and regions labeled "Arrangement: Arrangement". The tracks include "Celli", "Basses", and various string instruments like "Violins 1", "Violins 2", "Violas", "Cellos", and "Bassoon". Regions are color-coded by theme: blue for "Intro", purple for "Theme A", orange for "Theme B", red for "Theme A (Repeat)", and green for "Theme B (Repeat)". The bottom section shows a piano roll editor for the "Choir Women (Legato)" track. The piano roll displays notes over four measures (59-62). On the left, there are various controls for Kontakt 5, including Time Quantize (classic), Scale Quantize (E Major), and Velocity. On the right, a video feed shows a young man wearing a red hoodie and headphones, sitting at a desk with a microphone, presumably the composer. The DAW interface includes a toolbar at the top with various functions like "Edit", "Functions", "View", and "Score". The status bar at the bottom shows the time as 01:56:22.07, tempo as 105.0000, and other settings like "Keep Tempo", "4/4", "No In", "No Out", and "CPU".

출처 : <https://www.aiva.ai/>

AI 화가

ORIGINAL PHOTO



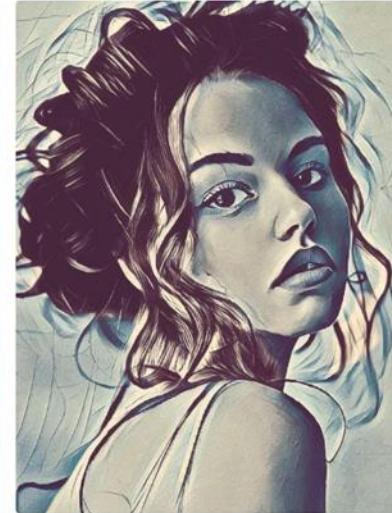
REWORKED PHOTO



ORIGINAL PHOTO



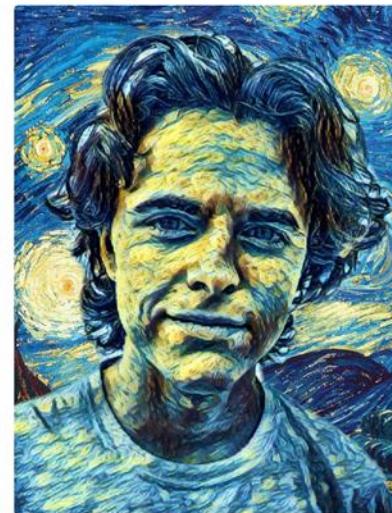
REWORKED PHOTO



ORIGINAL PHOTO



REWORKED PHOTO



ORIGINAL PHOTO



REWORKED PHOTO



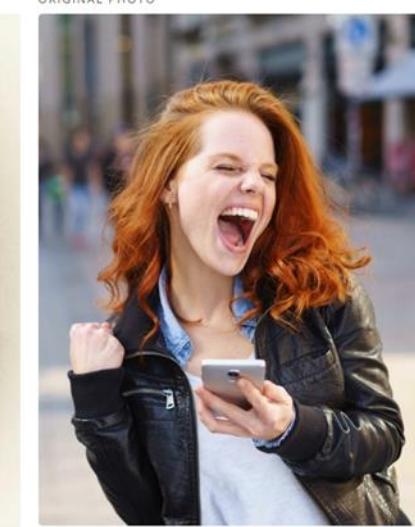
ORIGINAL PHOTO



REWORKED PHOTO



ORIGINAL PHOTO



REWORKED PHOTO



Talk to Transformer

See how a modern neural network completes your text. Type a custom snippet or try one of the examples. [Learn more](#) below.

 Follow @AdamDanielKing for updates and other demos like this one.

Custom prompt

How to unlock creativity and generate great ideas?

COMPLETE TEXT



Completion

How to unlock creativity and generate great Ideas?

Here you will learn how to discover cool ideas and make cool ideas.

How to make cool ideas?

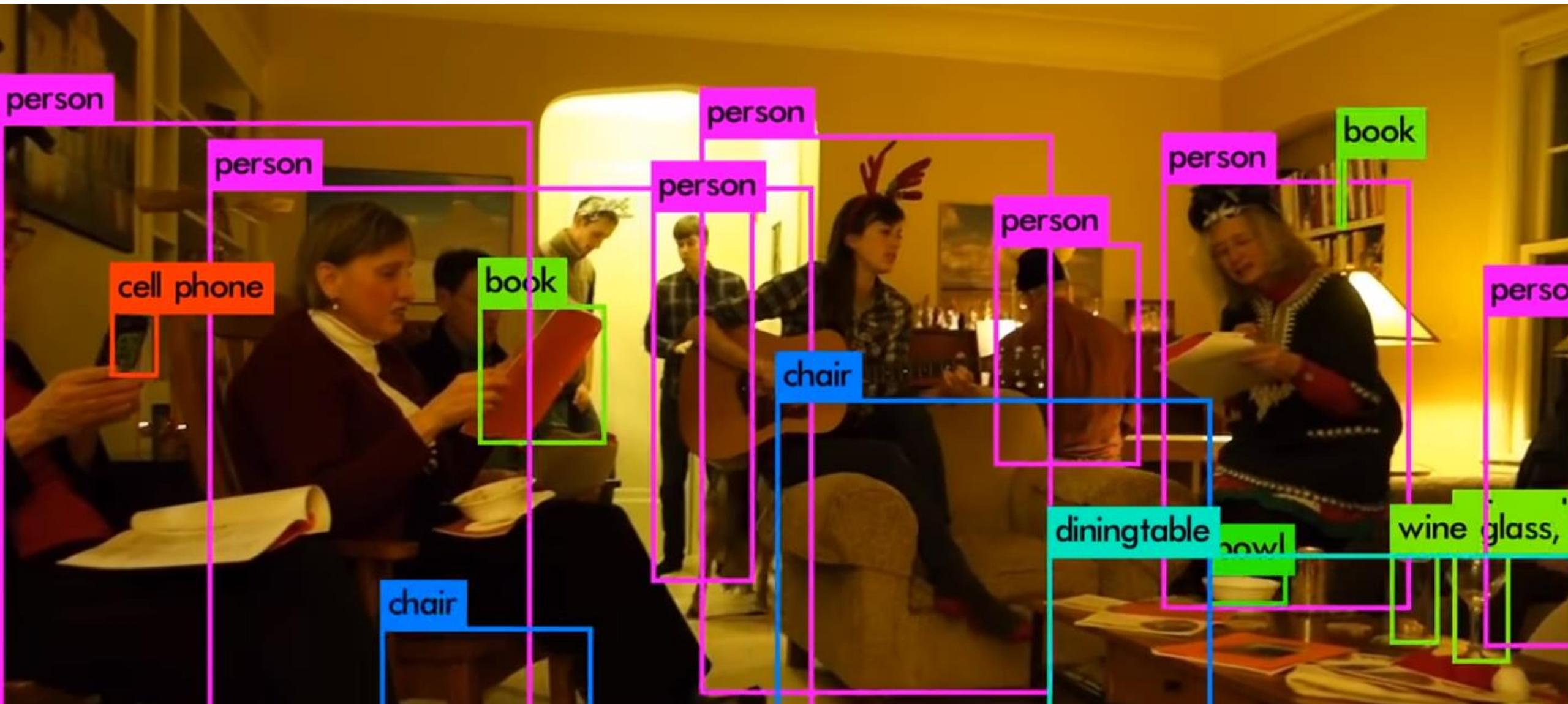
Here you will learn how to create cool ideas in no time using the "The 8 Habits of Highly Effective People"

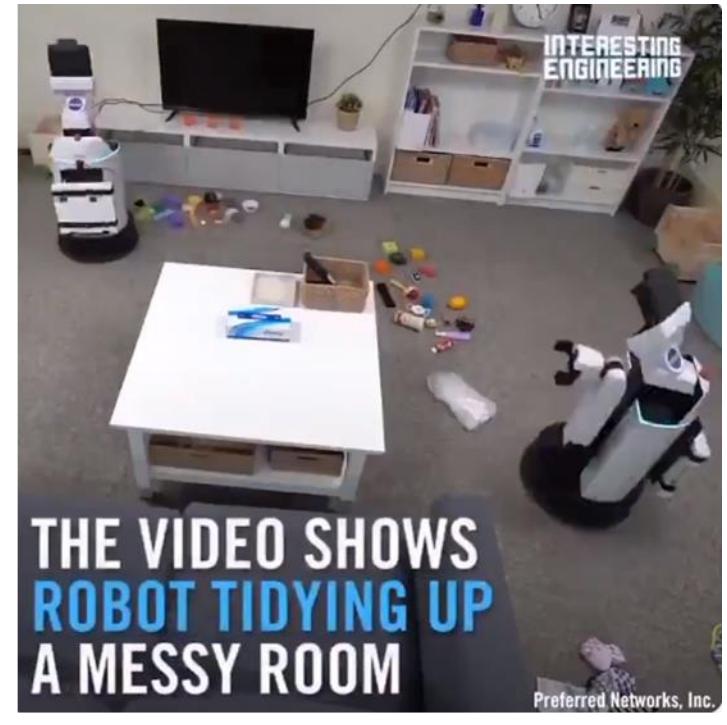
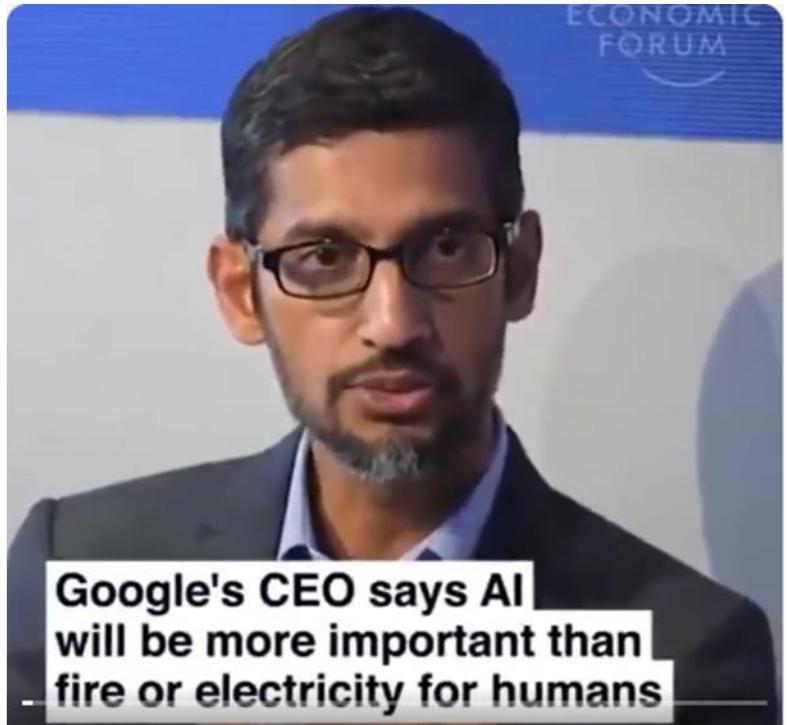
What are the 8 Habits of Highly Effective People?

The 8 Habits for Highly Effective People:

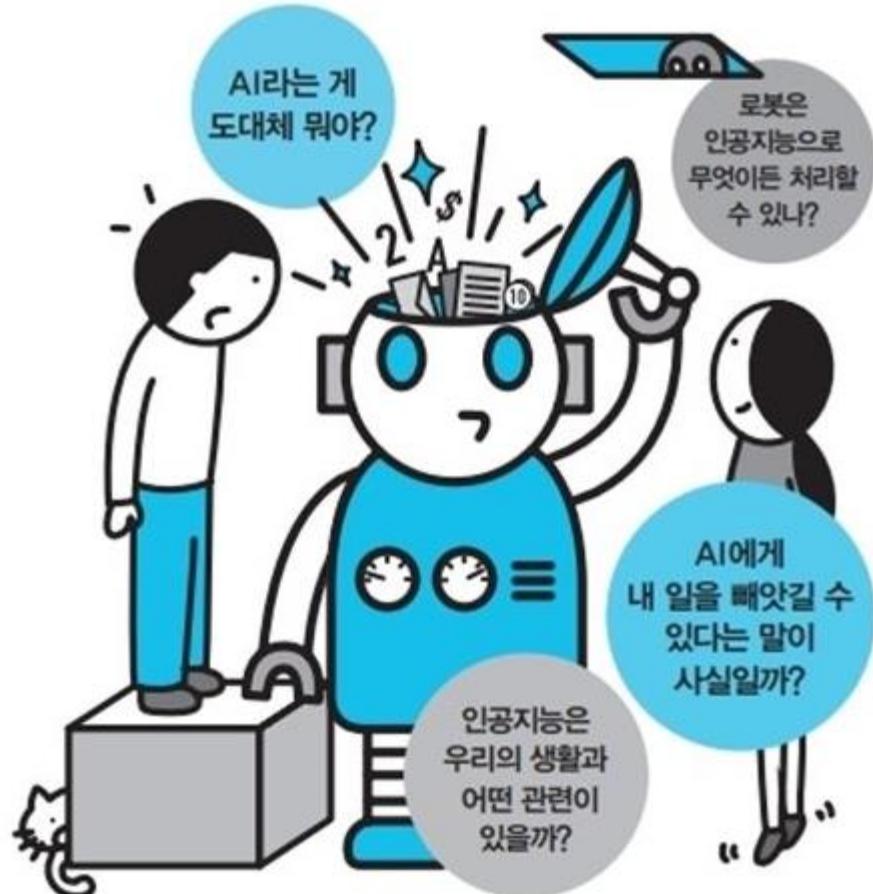
1. Know what people want
2. Be sure of your goals.
3. Be willing to compromise
4. Be curious about what people need
5. Never give in to pressure
6. Have fun
7. Do what you love.
8. Love working for others.

실시간 오브젝트 감지





AI가 뭔가요?



- AI가 뭐예요?
- 굳이 설명하자면 스스로 생각하고 판단할 수 있는 컴퓨터예요 [→ P.012].
- AI에게 내 일을 빼앗길 수 있다는 말이 사실인가요?
- 모든 일이 AI로 대체되지는 않아요. 하지만 AI 때문에 없어지는 일도 있고 새롭게 생겨나는 일도 있어요 [→ P.166].
- AI를 활용할 때 중요한 점은 뭘까요?
- ① 문제의 본질을 파악하는 능력,
② 데이터를 만드는 능력.
이렇게 두 가지예요 [→ P.150].

인공지능(Artificial Intelligent)



인공 지능

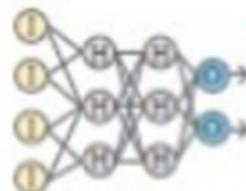
인간의 지적능력(추론, 인지)을 구현하는 모든 기술



머신 러닝

알고리즘으로 데이터를 분석, 학습하여 판단이나 예측을 하는 기술

선형회귀
로지스틱회귀
K-최근접 이웃
결정트리
랜덤포레스트
서포트 벡터 머신



클러스터링 차원축소

딥러닝

인공신경망 알고리즘을 활용하는 머신러닝 기술

심층신경망
(DNN)

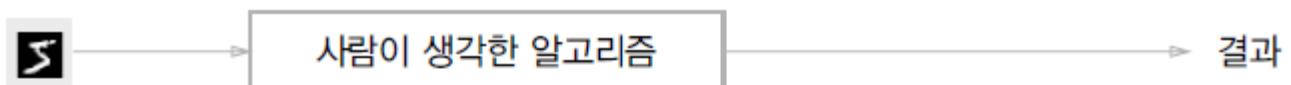
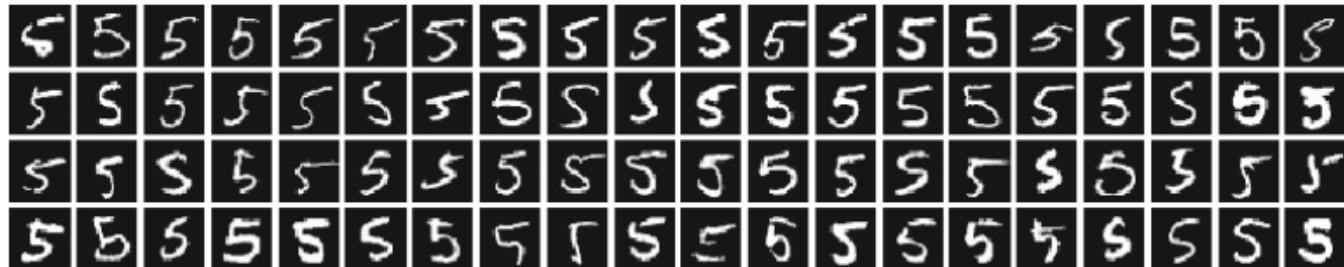
합성곱 신경망
(CNN)

순환 신경망
(RNN)

강화 학습

머신러닝과 딥러닝의 차이점

자유분방한 손글씨 이미지를 보고 5인지 아닌지를 구분하는 프로그램을 만들려고 합니다.
어떻게 하면 될까요?

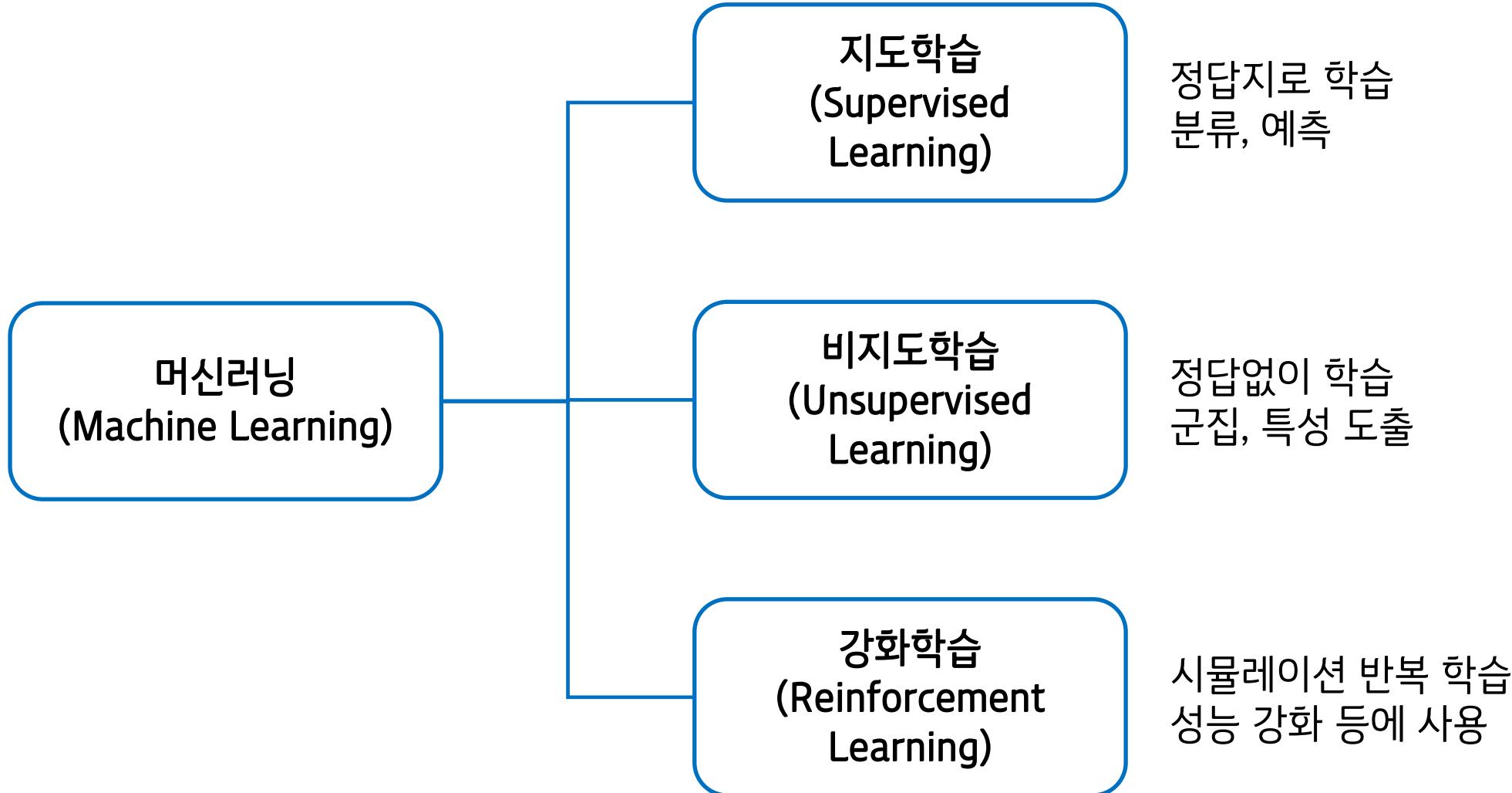


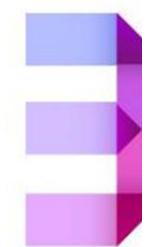
기계학습에서는 데이터로부터 규칙을 찾아내는 역할을 '기계'가 담당하므로, 문제에 적합한 특징을 쓰지 않으면 좀처럼 좋은 결과를 얻을 수 없다.



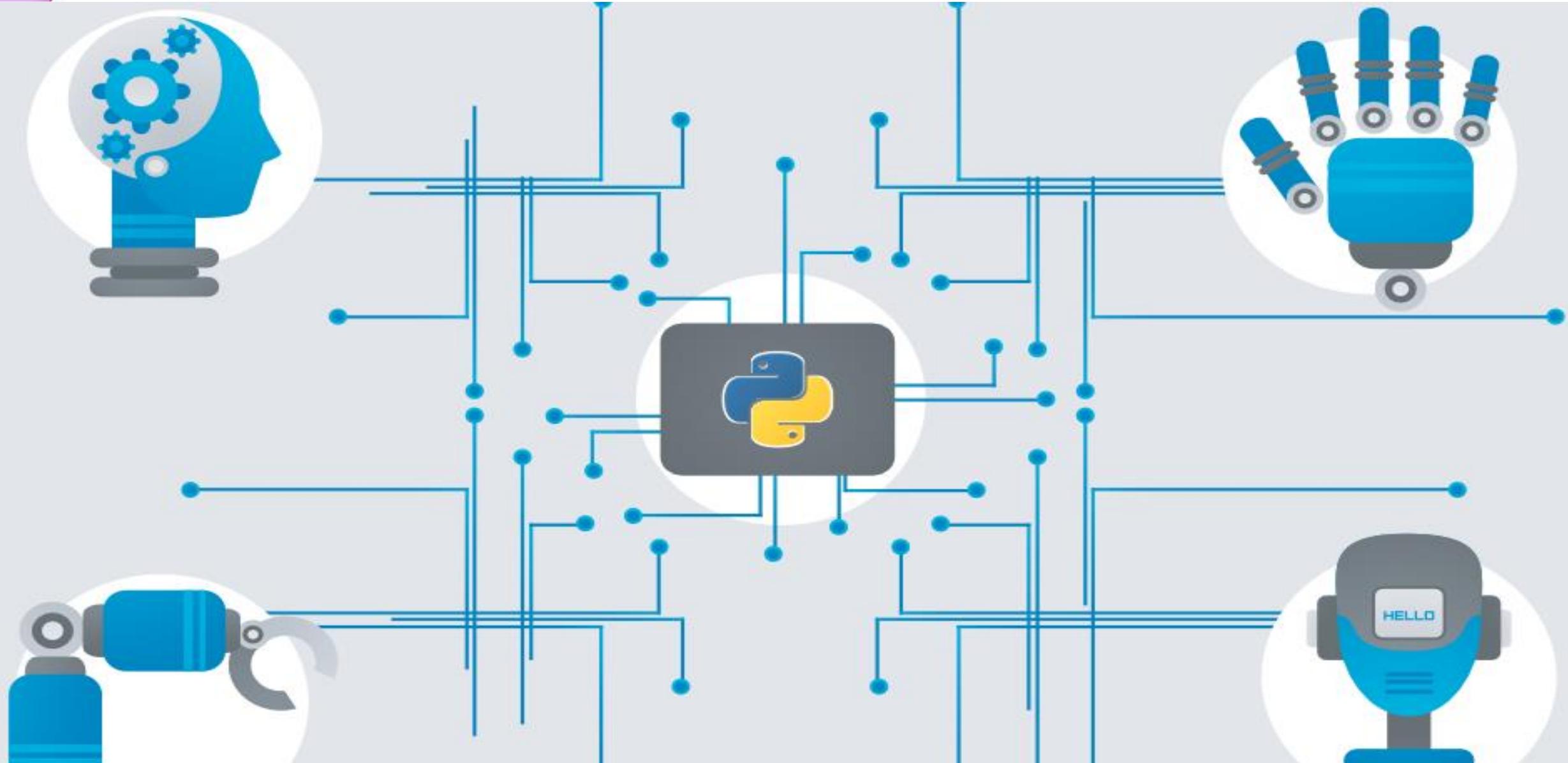
신경망은 이미지를 '있는 그대로' 학습하며, 이미지에 포함된 중요한 특징까지도 '기계'가 스스로 학습한다.

머신러닝/딥러닝 종류





AI 솔루션 개발



AI 솔루션 개발 언어 - 파이썬(Python)

1) Python the Best programming language in AI

2) Community support

3) Good visualization options

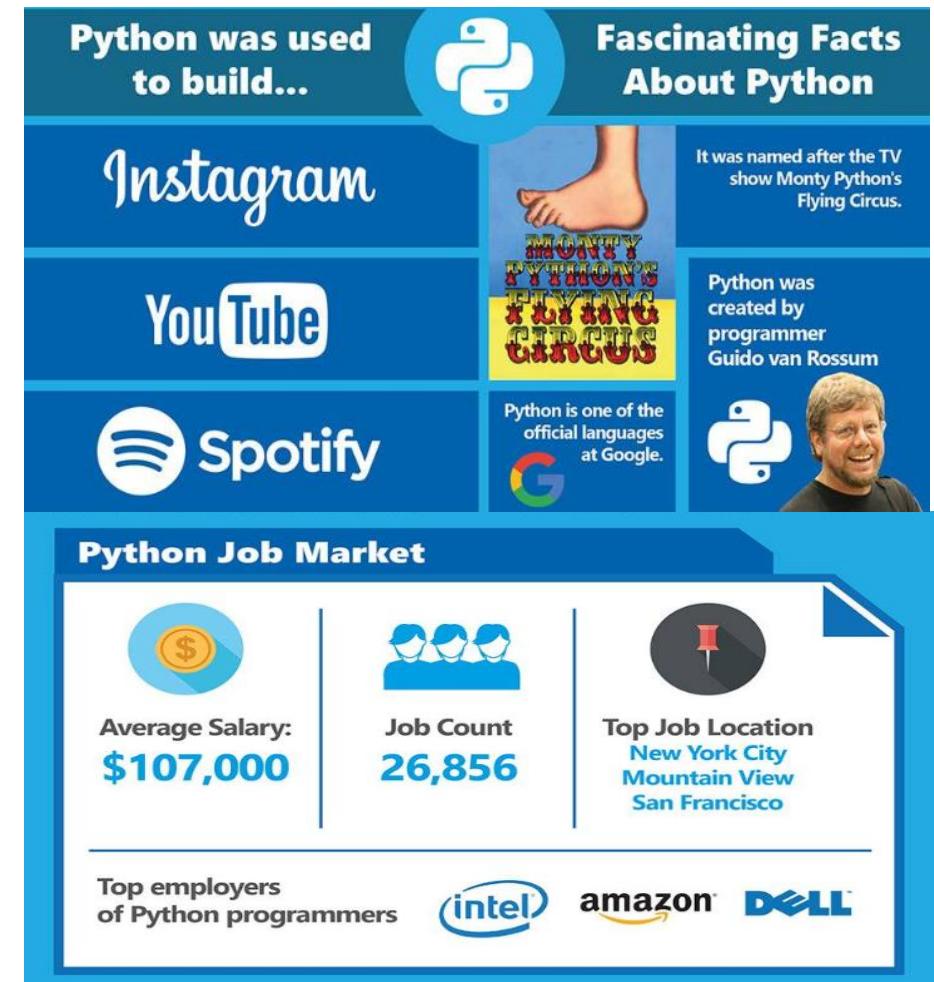
4) Easy Readability

5) Platform independence

6) Flexibility

7) Easy Coding

8) A rich Python Libraries for AI Projects



스터디 자료

<https://github.com/kgpark88/python/>

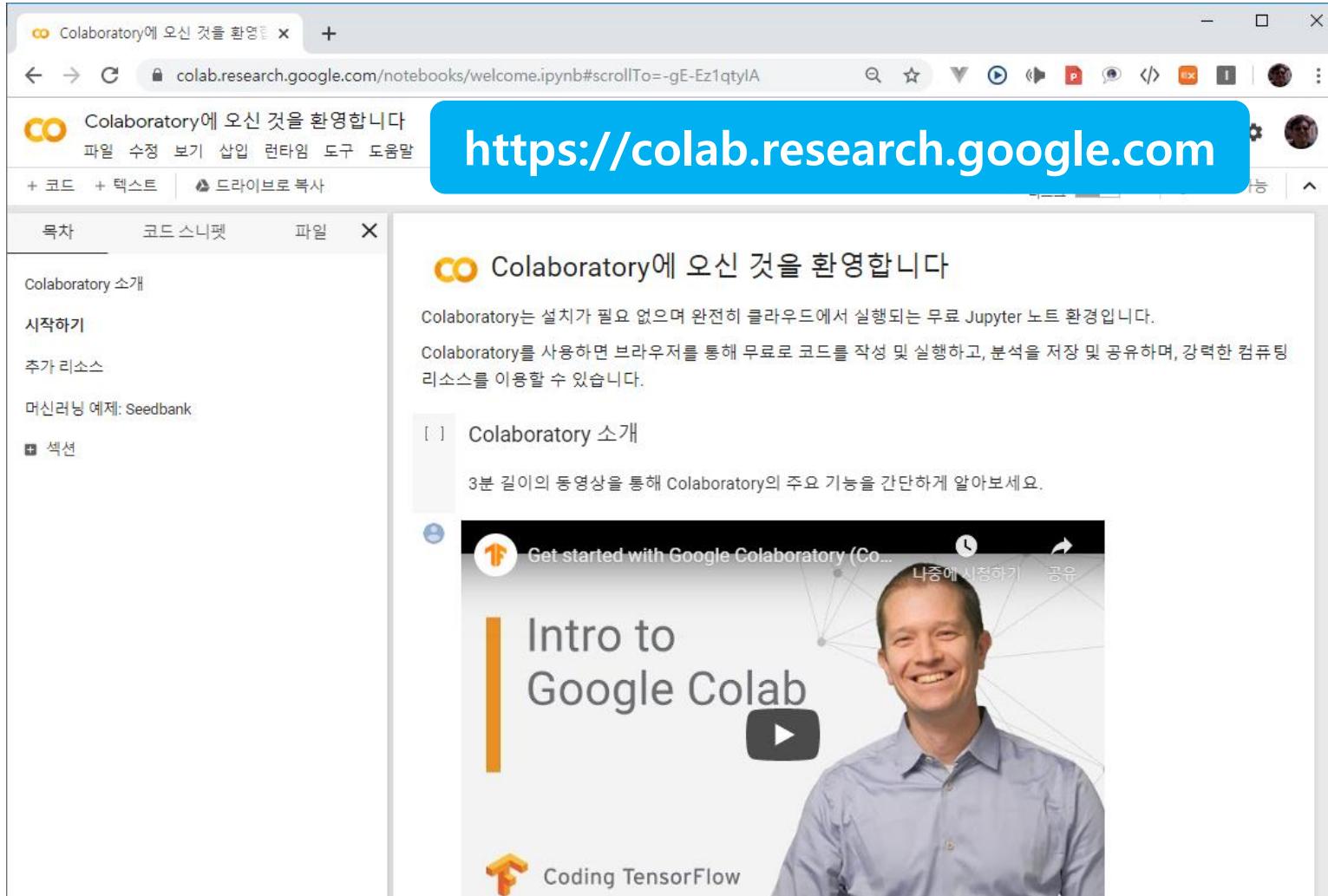
-  01_우리가 프로그래밍을 배워야 하는 .pdf
-  02_컴퓨팅사고력.pdf
-  03_왜 파이썬인가.pdf
-  04_통합개발환경에서 코딩하기.pdf
-  05_변수와데이터타입.pdf
-  06_시퀀스데이터타입과딕셔너리.pdf
-  07_프로그램제어문.pdf
-  08_IDE보다 간편한 코드에디터.pdf

<https://github.com/kgpark88/deeplearning>

-  1_1_0_AI.pdf
-  1_1_0_PythonForAI.pdf
-  1_1_classification.py
-  1_2_0_딥러닝필수개념.pdf
-  1_3_0_인공신경망소개.pdf
-  1_3_1_classification.py
-  1_3_2_perceptron.py
-  1_3_3_regression.py

구글 코랩(Google Colab)

프로그램 설치 없이 클라우드에서 AI 프로그래밍을 할 수 있는 환경으로 GPU를 포함한 가속 컴퓨팅 환경을 선택사항으로 제공합니다. 런타임 유형을 GPU로 변경하여 GPU를 이용할 수 있습니다.



The screenshot shows the Google Colaboratory landing page in a web browser. The URL in the address bar is <https://colab.research.google.com>. The main content area displays the title "Colaboratory에 오신 것을 환영합니다" and a brief introduction about the service. Below the introduction, there is a video thumbnail titled "Intro to Google Colab" featuring a man smiling. The sidebar on the left contains links for "Colaboratory 소개", "시작하기", "추가 리소스", "머신러닝 예제: Seedbank", and "섹션".



파이썬 기본기

■ 변수 할당(Variable Assignment)

```
x = 2
```

```
y = 3
```

```
z = x + y
```

■ 출력

```
x = 'hello'
```

```
x = "hello"
```

```
x
```

```
[Out] 'hello'
```

Single Quotation
작은 따옴표

Double Quotation
쌍 따옴표

```
print(x)
```

```
[Out] 'hello'
```

■ 리스트(List)

```
[1, 2, 3]
```

```
my_list = ['a', 'b', 'c']
```

```
my_list.append('d')
```

```
my_list[0]
```

```
my_list[:-1]
```

```
my_list[-1]
```

Bracket
대괄호

■ 딕셔너리(Dictionary)

```
d = {'key1': 'item1', 'key2': 'item2'}
```

```
d['key1']
```

```
[Out] 'item1'
```

Brace
중괄호

실습 자료

https://github.com/kgpark88/deeplearning/blob/master/ai_modeling.ipynb

The screenshot shows a GitHub repository page for 'kgpark88/deeplearning'. The repository name is 'deeplearning / ai_modeling.ipynb'. The page includes navigation links like Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. A red box highlights the 'Open in Colab' button, which is located in the main content area below the file statistics (4009 lines, 410 KB). The content area contains sections for 'AI 모델링 실습' and '1. 파이썬(Python)', with a code cell showing variable assignments: In [1]: x = 2, y = 3, z = x + y.

파이썬 주요 패키지



NumPy

행렬과 다차원 배열을 쉽게 처리 할 수 있게 해주는 패키지

pandas

데이터를 처리하고 분석하는 데 효과적인 패키지

matplotlib

데이터를 차트나 플롯(Plot)으로 그려주는 시각화 라이브러리 패키지

seaborn

matplotlib을 기반으로 다양한 색상 테마와 통계용 차트 등의 기능을 추가한 시각화 패키지

scikit
learn

교육 및 실무를 위한 머신러닝 라이브러리 패키지

TensorFlow

구글에서 만든 오픈소스 딥러닝 라이브러리 패키지

넘파이(Numpy)

NumPy(Numerical Python)는 파이썬 선형대수 라이브러리입니다. 데이터 분석, 수학/과학연산을 위한 파이썬 기본 패키지로 고성능의 다차원 배열 객체와 다양한 객체에 대해 고속 연산을 가능하게 합니다.

```
data = np.array([1,2,3])
```

data
1
2
3

data
1
2
3
.max() = 3

data
1
2
3
.min() = 1

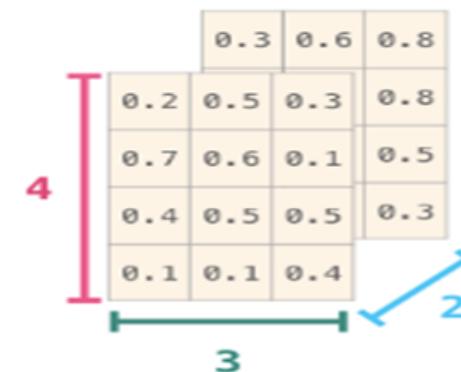
data
1
2
3
.sum() = 6

```
np.array([[1,2],[3,4],  
[[5,6],[7,8]]])
```



1	2	5	6
3	4	8	

```
np.random.random((4,3,2))
```



넘파이(Numpy)

■ Numpy 라이브러리 임포트

```
import numpy as np
```

■ Numpy Array 생성

```
my_list = [1, 2, 3]  
np.array(my_list)  
[Out] array([1, 2, 3])
```

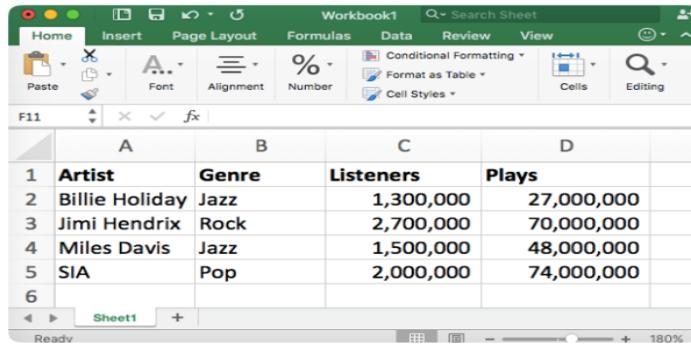
```
my_matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]  
np.array(my_matrix)  
[Out] array([[1, 2, 3],  
           [4, 5, 6],  
           [7, 8, 9]])
```

판다스(Pandas)

Pandas는 데이터 분석을 위해 널리 사용되는 파이썬 라이브러리 패키지입니다.

데이터프레임 자료구조를 사용하여, 데이터 분석에 있어 높은 수준의 성능을 발휘합니다.

music.csv

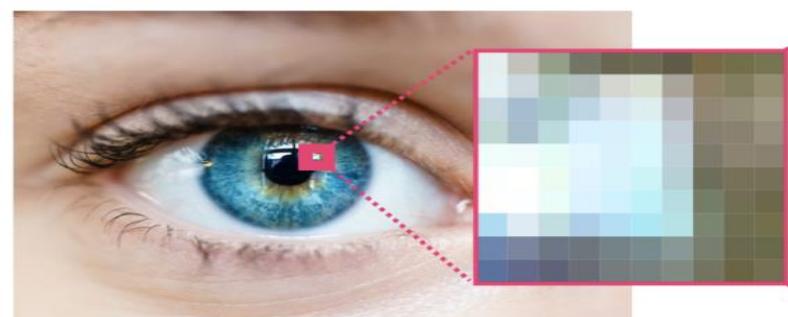


pandas.read_csv('music.csv')

	Artist	Genre	Listeners	Plays
0	Billie Holiday	Jazz	1,300,000	27,000,000
1	Jimi Hendrix	Rock	2,700,000	70,000,000
2	Miles Davis	Jazz	1,500,000	48,000,000
3	SIA	Pop	2,000,000	74,000,000

df['Avg Plays'] = df['Plays']/df['Listeners']

	Artist	Genre	Listeners	Plays	Avg Plays
0	Billie Holiday	Jazz	1,300,000	27,000,000	20
1	Jimi Hendrix	Rock	2,700,000	70,000,000	25
2	Miles Davis	Jazz	1,500,000	48,000,000	32
3	SIA	Pop	2,000,000	74,000,000	37



233	188	137	96	90	95	63	73	73	82
237	202	159	120	105	110	88	107	112	121
226	191	147	110	101	112	98	123	110	119
221	191	176	182	203	214	169	144	133	145
185	160	161	184	205	223	186	137	147	161
181	174	189	207	206	215	194	136	142	151
246	237	237	231	208	206	192	122	143	144
254	254	241	224	199	192	181	99	122	117
239	248	232	207	187	182	184	110	114	110
193	215	193	167	158	164	181	114	112	111
113	119	110	111	113	123	135	120	108	106
93	97	91	103	107	111	122	112	104	114
									113

판다스 데이터프레임(DataFrame)

컬럼명
(Column Names)

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService
인덱스 Index	0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
	1	5575-GNVDE	Male	0	No	No	34	Yes	No
	2	3668-QPYBK	Male	0	No	No	2	Yes	No
	3	7795-CFOCW	Male	0	No	No	45	No	No phone service
	4	9237-HQITU	Female	0	No	No	2	Yes	Fiber optic

데이터

판다스(Pandas)

■ Pandas 라이브러리 임포트

```
import pandas as pd
```

■ 파일에서 데이터를 로드하는 방법

```
df = pd.read_csv('data.csv')
```

■ 데이터 확인

```
df.head()
```

```
df.head(20)
```

```
df.tail()
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
1	5575-GNVDE	Male	0	No	No	34	Yes	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service

판다스(Pandas)

■ 자료구조 파악

`df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   customerID      7043 non-null    object  
 1   gender          7043 non-null    object  
 2   SeniorCitizen   7043 non-null    int64  
 3   Partner         7043 non-null    object  
 4   Dependents     7043 non-null    object  
 5   tenure          7043 non-null    int64  
 ....., 
 18  MonthlyCharges 7043 non-null    float64 
 19  TotalCharges   7043 non-null    object  
 20  Churn          7043 non-null    object  
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

■ 데이터 타입 확인

`df.dtypes`

```
customerID        object
gender            object
SeniorCitizen    int64
Partner           object
Dependents        object
tenure            int64
PhoneService      object
MultipleLines     object
InternetService   object
OnlineSecurity    object
OnlineBackup      object
DeviceProtection  object
TechSupport       object
StreamingTV       object
StreamingMovies   object
Contract          object
PaperlessBilling  object
PaymentMethod     object
MonthlyCharges    float64
TotalCharges      object
Churn             object
dtype: object
```

■ Null 데이터 확인

`df.isnull().sum()`

```
customerID      0
gender          0
SeniorCitizen   0
Partner         0
Dependents      0
tenure          0
PhoneService    0
MultipleLines   0
InternetService 0
OnlineSecurity  0
OnlineBackup    0
DeviceProtection 0
TechSupport     0
StreamingTV     0
StreamingMovies 0
Contract        0
PaperlessBilling 0
PaymentMethod   0
MonthlyCharges  0
TotalCharges    0
Churn          0
dtype: int64
```

판다스(Pandas)

■ 통계 정보

df.describe()

	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692
std	0.368612	24.559481	30.090047
min	0.000000	0.000000	18.250000
25%	0.000000	9.000000	35.500000
50%	0.000000	29.000000	70.350000
75%	0.000000	55.000000	89.850000
max	1.000000	72.000000	118.750000

■ 데이터 상관관계 분석

df.corr()

	SeniorCitizen	tenure	MonthlyCharges
SeniorCitizen	1.000000	0.016567	0.220173
tenure	0.016567	1.000000	0.247900
MonthlyCharges	0.220173	0.247900	1.000000

■ 판다스(Pandas)

■ 데이터 전처리

입력 데이터에서 제외: drop()

Null 데이터 처리: dropna(), fillna()

누락데이터 처리: replace()

데이터타입 변환: astype()

특성 추출 (feature engineering) : df['new_feature'] = df['f_1']/df['f_2']

```
df.drop('customerID', axis=1, inplace=True)
```

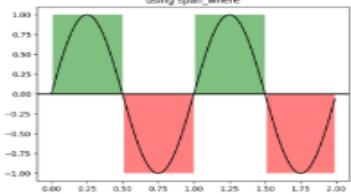
```
df['TotalCharges'].replace([' '], [0], inplace=True)
```

```
df['TotalCharges'] = df['TotalCharges'].astype(float)
```

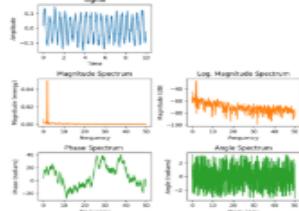
```
df['Churn'].replace(['Yes', 'No'], [1, 0], inplace=True)
```

matplotlib

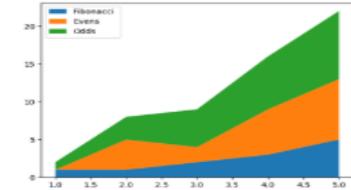
파이썬에서 데이터를 차트나 플롯(Plot)으로 그려주는 라이브러리 패키지로서
가장 많이 사용되는 데이터 시각화(Data Visualization) 패키지입니다.



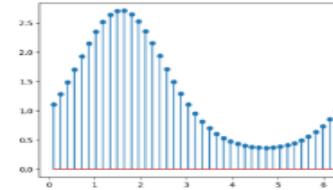
Using span_where



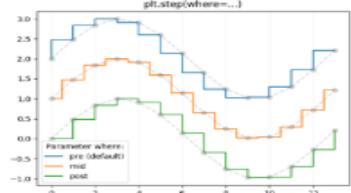
Spectrum Representations



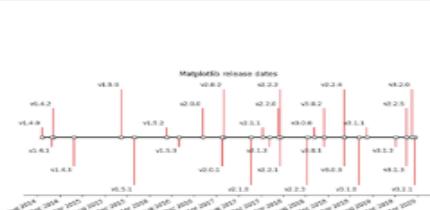
Stackplot Demo



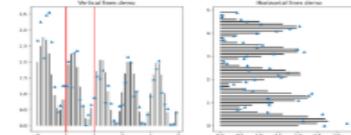
Stem Plot



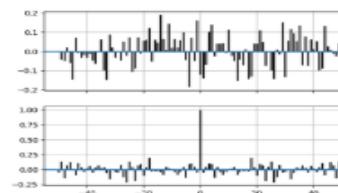
Step Demo



Creating a timeline with lines, dates, and text



hlines and vlines



Cross- and Auto-Correlation Demo

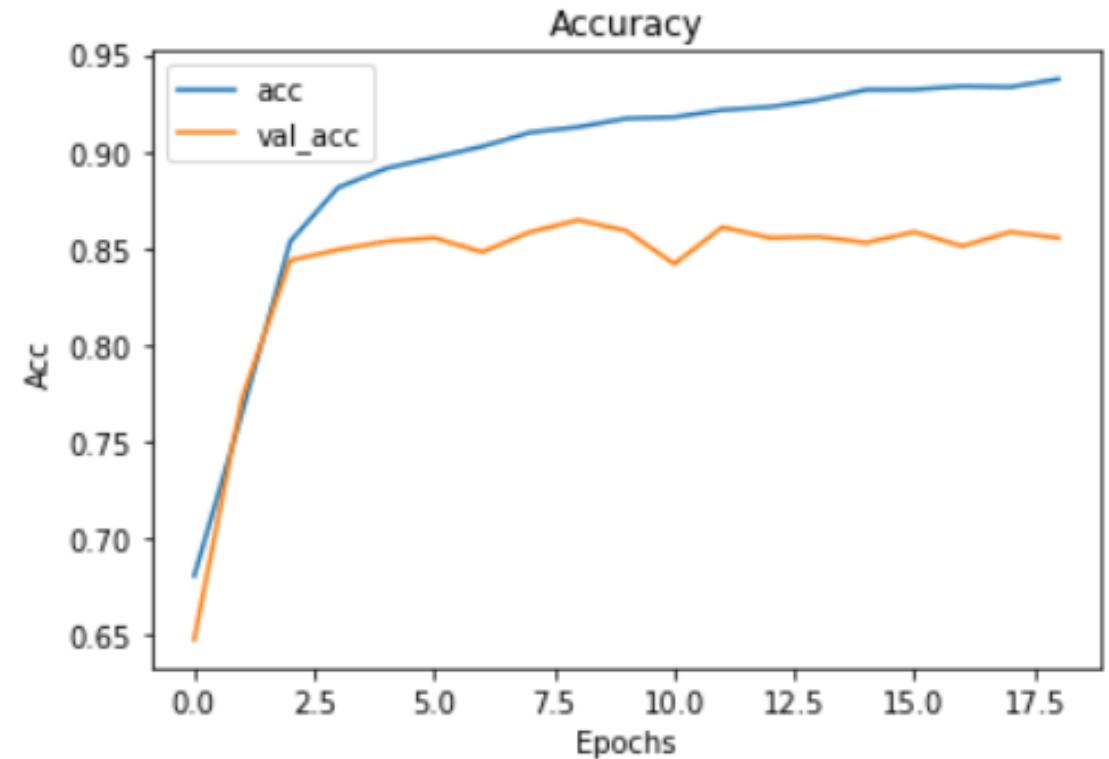
■ 맷플롯립(Matplotlib)

■ 라이브러리 임포트

```
import matplotlib.pyplot as plt  
%matplotlib inline
```

■ Matplotlib 사용법(예시)

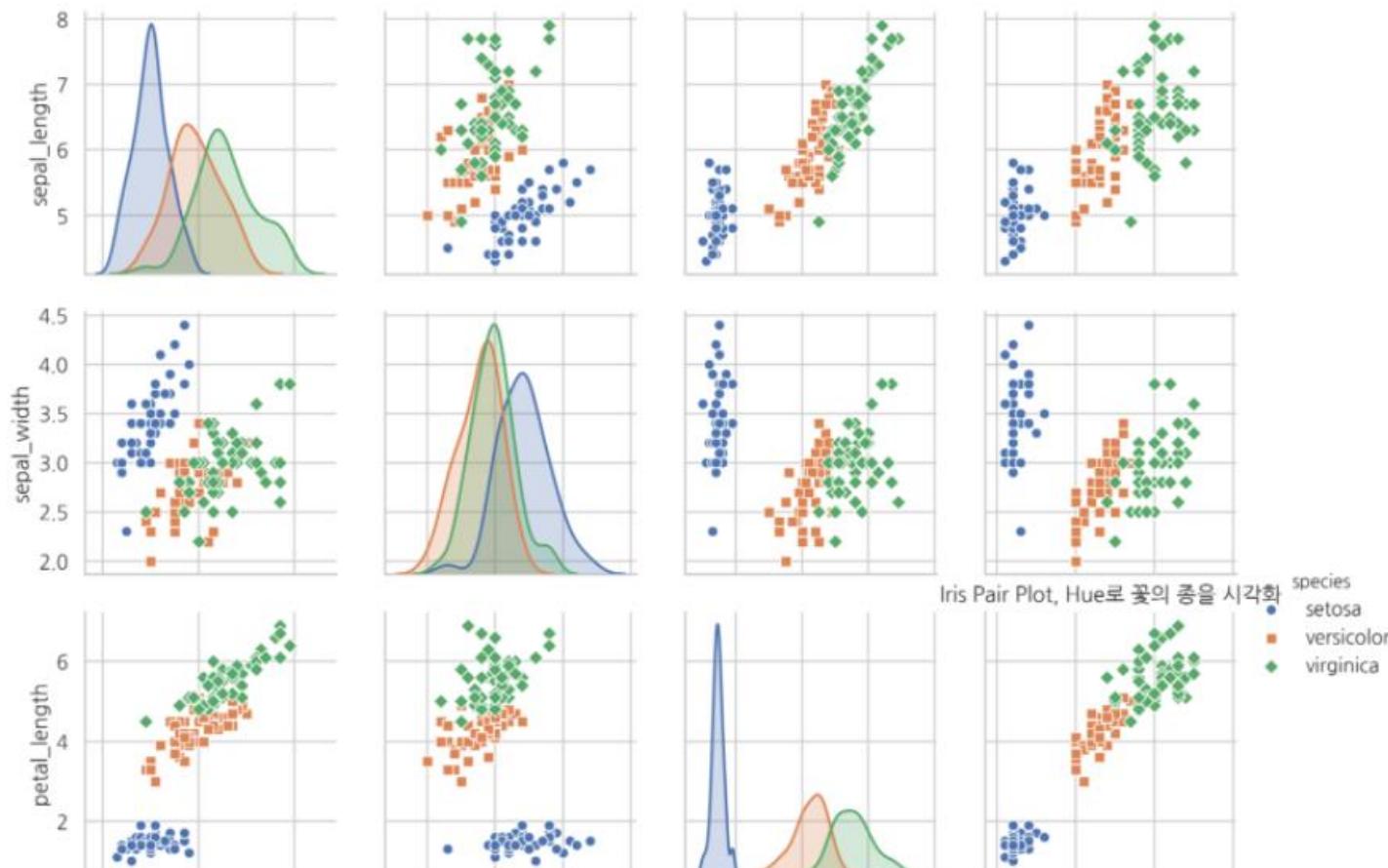
```
plt.plot(history.history['acc'])  
plt.plot(history.history['val_acc'])  
plt.title('Accuracy')  
plt.xlabel('Epochs')  
plt.ylabel('Acc')  
plt.legend(['acc', 'val_acc'])  
plt.show()
```



씨본(Seaborn)

Matplotlib을 기반으로 다양한 색상 테마와 통계용 차트 등의 기능을 추가한 시각화 패키지입니다.

```
sns.pairplot(iris, hue="species", markers=["o", "s", "D"])
plt.title("Iris Pair Plot, Hue로 꽃의 종을 시각화")
plt.show()
```



출처 : <https://bit.ly/3mqQsNY>

씨본(Seaborn)

■ 패키지 설치

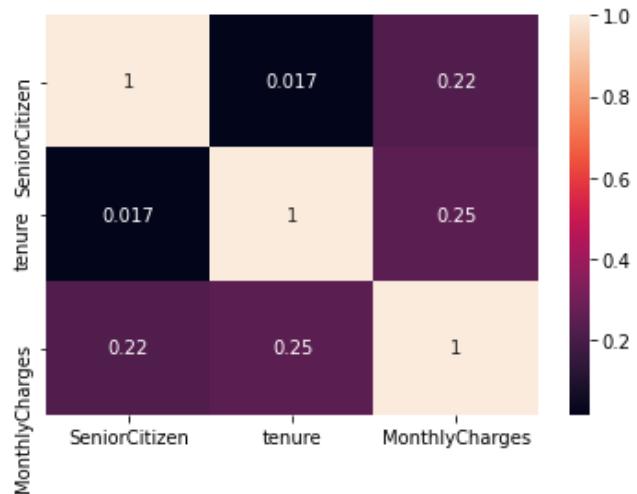
```
!pip install seaborn
```

■ 라이브러리 임포트

```
import seaborn as sns
```

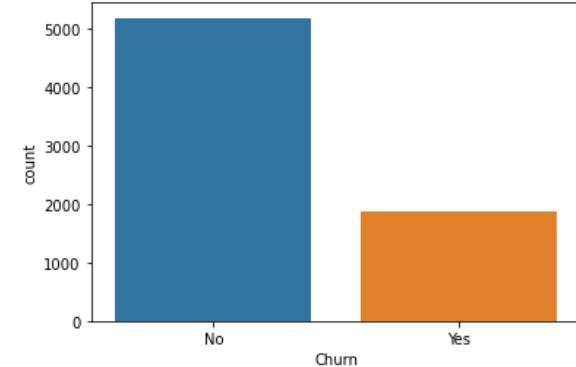
■ 상관관계 히트맵

```
sns.heatmap(df.corr(), annot=True)
```



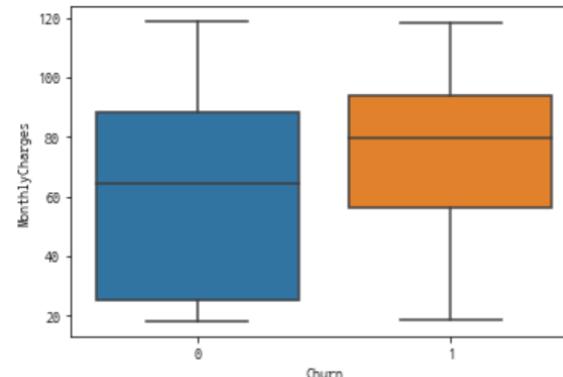
■ 카운트 플롯

```
sns.countplot(x='Churn', data=df)
```



■ 박스 플롯

```
sns.boxplot(x='Churn', y='MonthlyCharges', data=df)
```





4 머신러닝 이론과 주요모델

데이터

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4



머신러닝

머신러닝을 실현하기 위한 방법론이 여러개가 있지만, 그 중 지도학습(Supervised Learning)과 비지도학습(Un-Supervised Learning)이 대표적입니다.

■ 지도학습(supervised learning)

- 학습시 정답을 알려 주면서 진행하는 학습으로, 학습시 데이터와 레이블(정답)이 함께 제공됩니다.
- **레이블(Label)** = 정답, 실제값, 타깃, 클래스, y
- 예측된 값 = 예측값, 분류값, \hat{y} (y hat)
- 데이터마다 레이블을 달기 위해 많은 시간을 투자해야 합니다.
- 지도학습 모델에는 **분류모델**(이진분류, 다중분류)와 **예측(회귀)모델**(주가예측 등)이 있습니다.

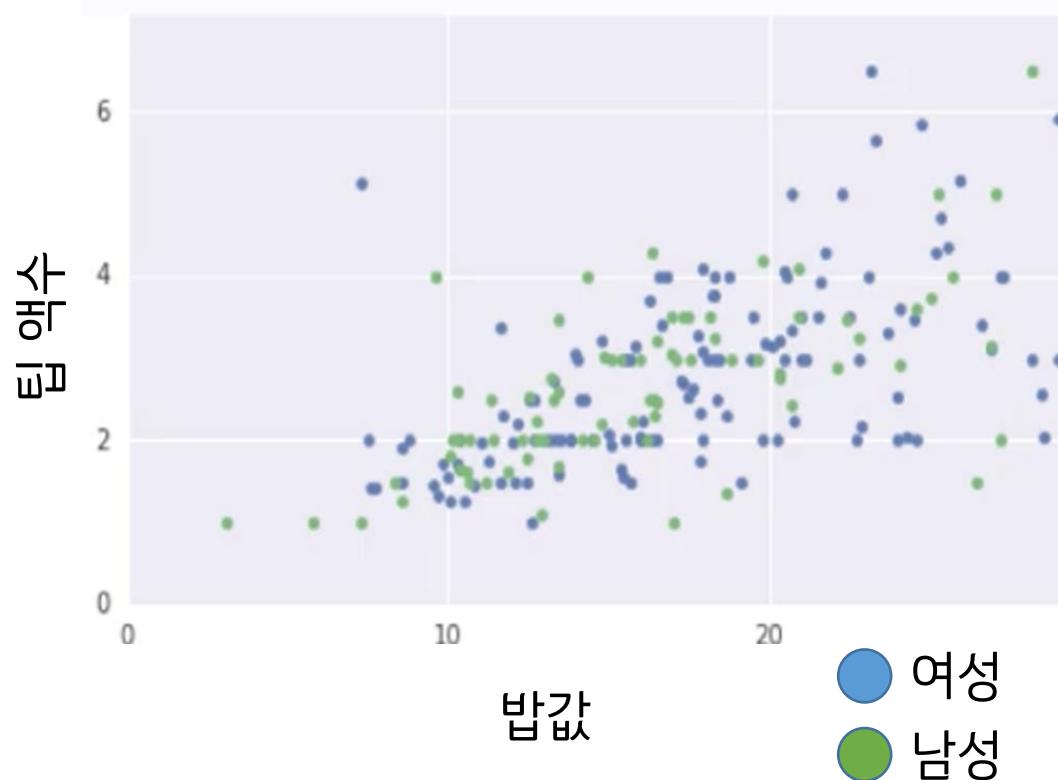
■ 비지도학습(unsupervised learning)

- 레이블(정답) 없이 진행되는 학습으로, 데이터 자체에서 패턴을 찾아내야 할 때 사용합니다.
- 비지도학습의 대표적인 예는 군집화(clustering)와 차원축소가 있습니다.

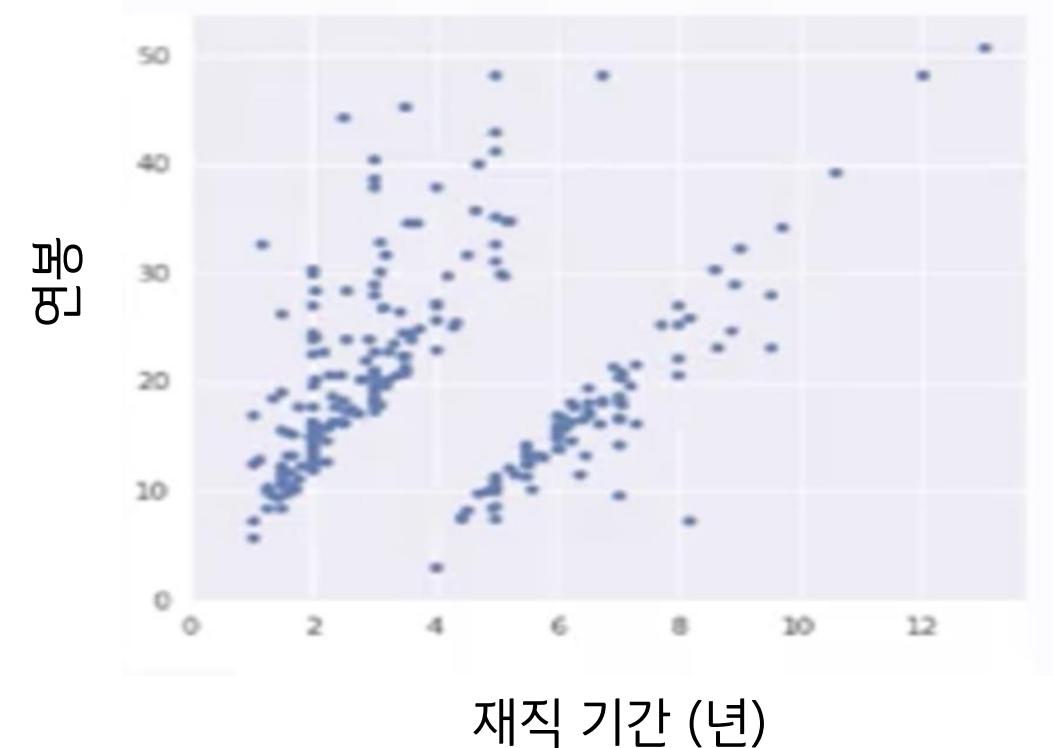
지도학습 vs 비지도학습

지도학습에서는 데이터가 레이블링 되어 있으며, 축적된 데이터로부터 클래스를 분류 또는 미래의 값을 예측합니다. 비지도학습에선 데이터가 레이블링 되어 있지 않습니다.

성별에 따른 팁 액수의 크기



패스트트랙(Fast-track) 인가?



지도학습 모델 종류

모델은 데이터들의 패턴을 대표할 수 있는 함수로, 입력을 독립변수 출력을 종속변수라고 합니다.

■ 모델이란?

- 데이터들의 패턴을 대표할 수 있는 함수, 예) $f(x) = ax + b$
- 함수의 입력은 독립변수이고 출력은 종속변수로, 독립변수들에 의해 출력값이 정해집니다.

■ 분류 모델(Classification)

- 레이블의 값들이 이산적으로 나눠질 수 있는 문제에 사용합니다.
- 예) 밥값이 많이 나오면 남성, 적게 나오면 여성

■ 예측 모델(Regression, 회귀모델)

- 레이블의 값들이 연속적인 문제에 사용합니다.
- 예) 밥값이 많이 나올수록, 팁의 크기도 계속 커진다.

지도학습 모델 종류

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

예측 모델(회귀 모델)
팁의 크기를 예측

분류 모델
손님의 성별을 예측

지도학습 데이터셋 구조

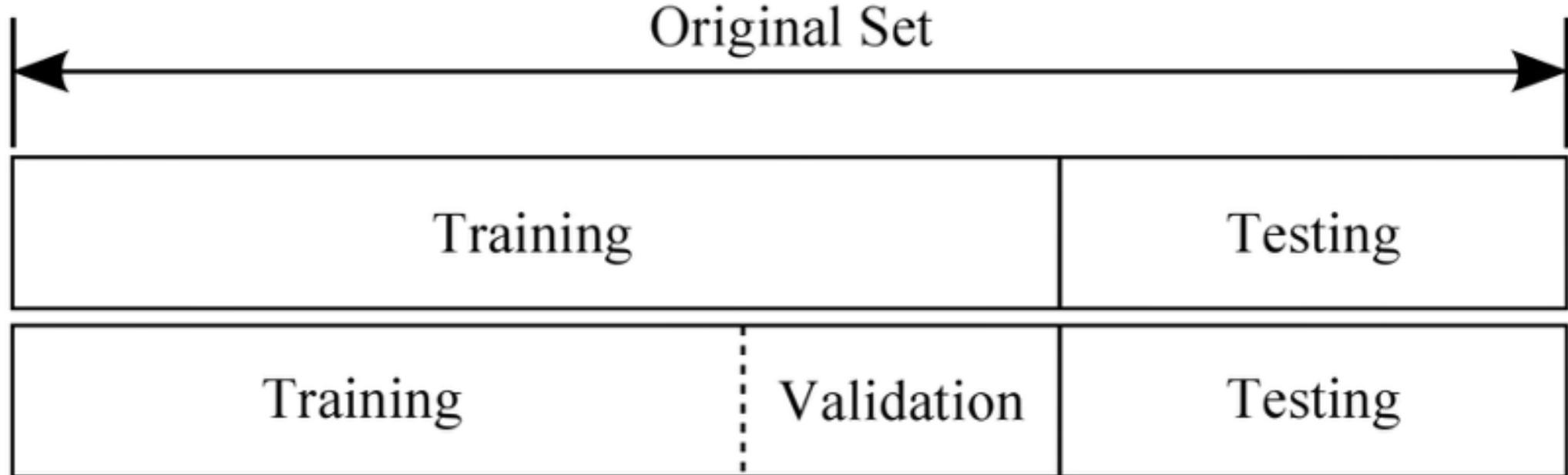
각 열(column)을 특징/속성(feature) 이라고 합니다.

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

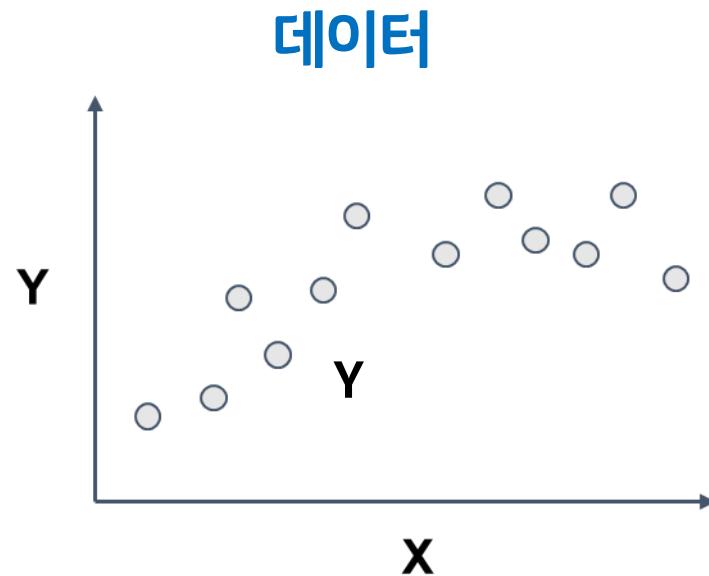
각 행(row)을
예제(Example)
데이터라고
합니다

데이터 컬럼(column)중에 하나를 선택해서 레이블로 사용합니다.

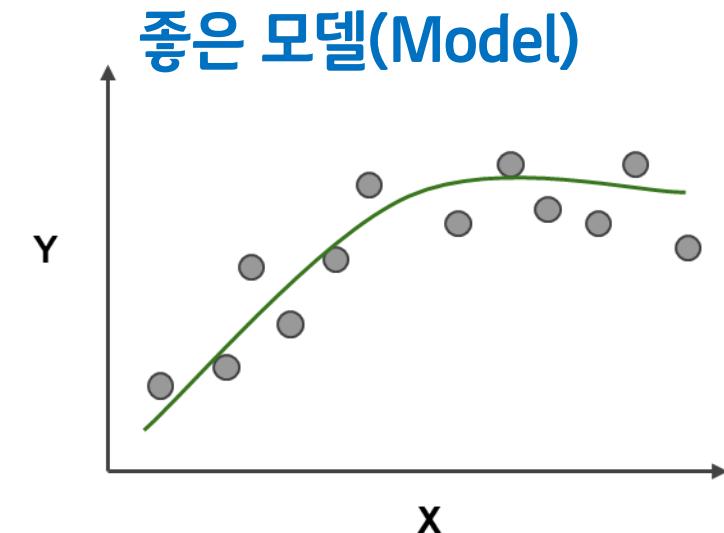
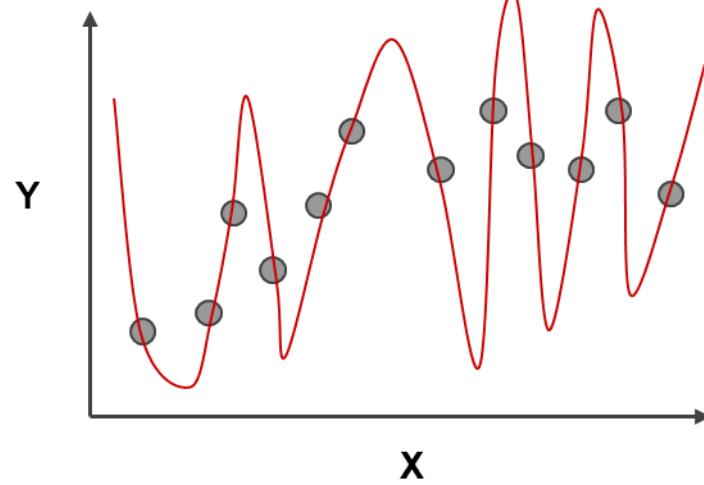
데이터셋 분리



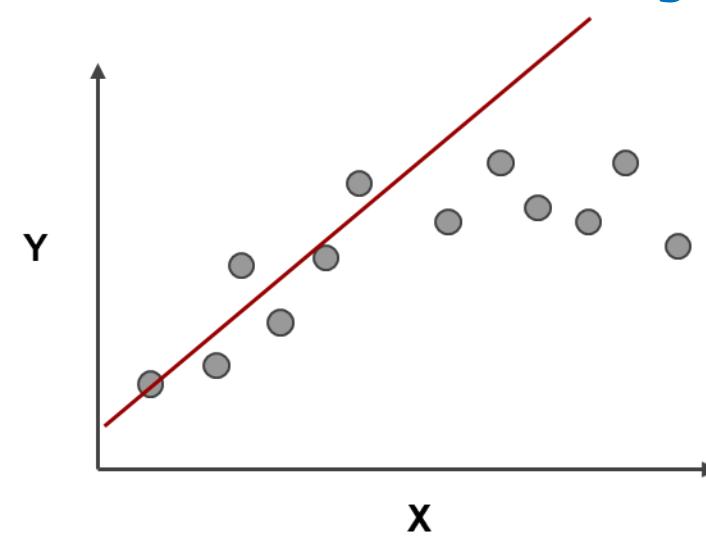
모델 선택



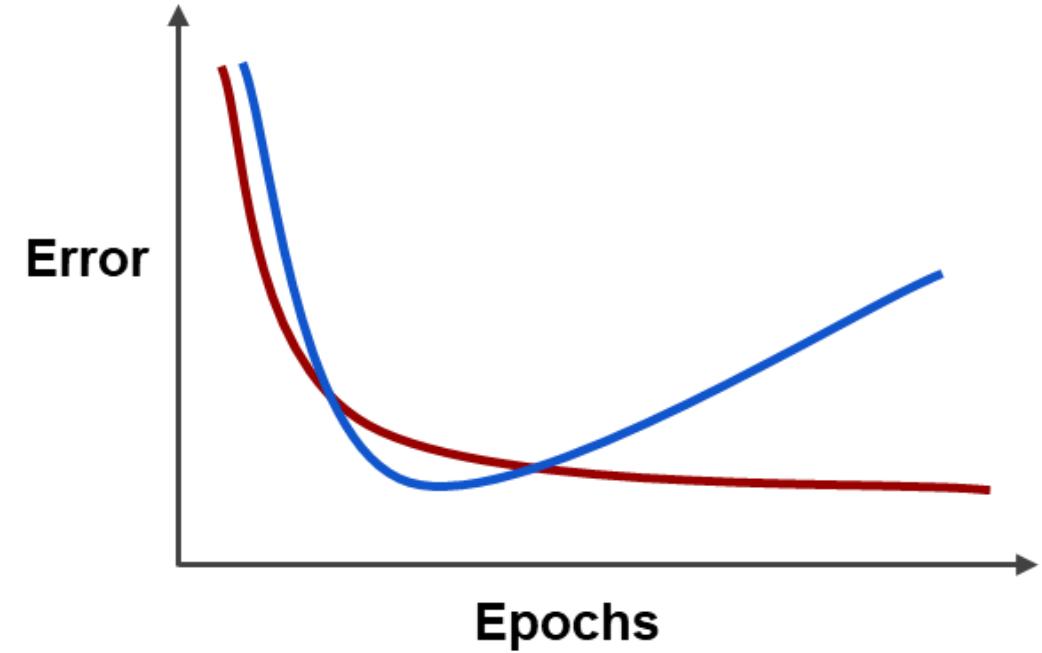
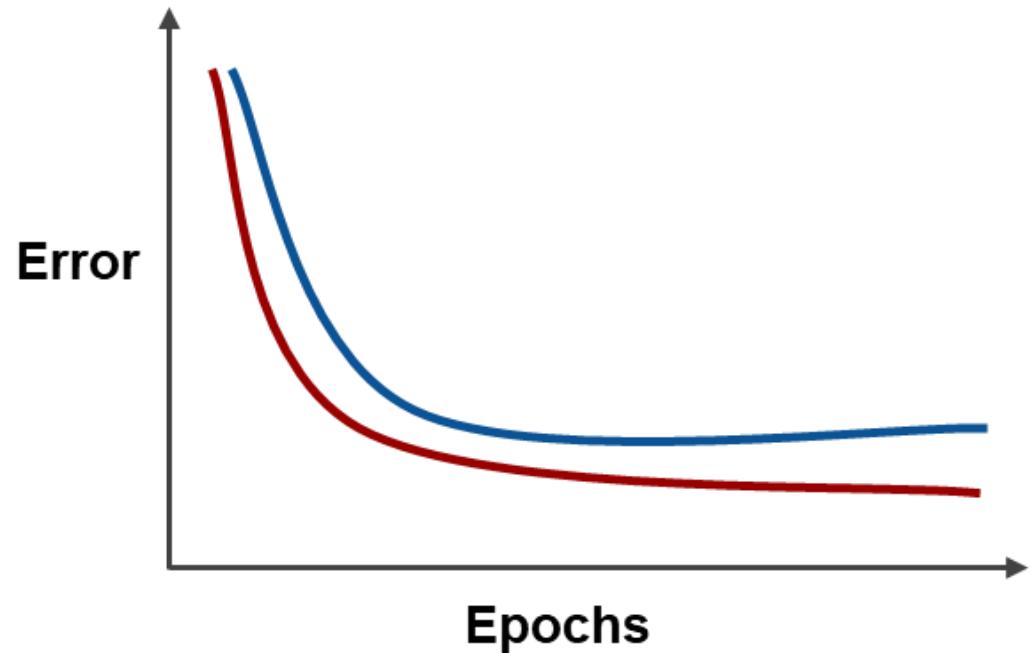
과적합(Overfitting)



과소적합(Underfitting)



모델 성능 평가

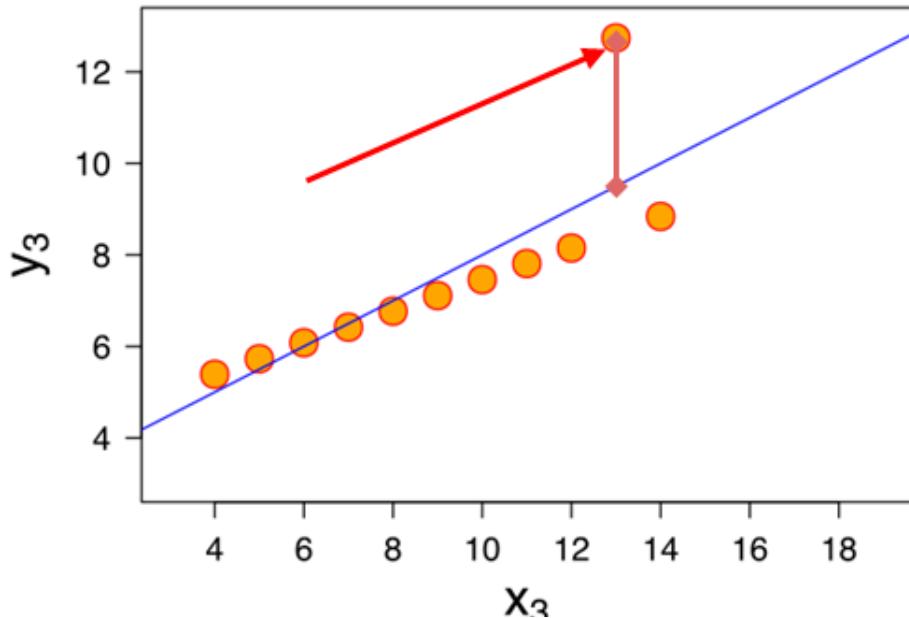


training set test set

손실함수

Loss Function은 모델의 손실을 측정하는 함수로 학습을 진행하면서 지속적으로 측정이 됩니다.
예측 모델의 손실측정함수는 MSE, 분류 모델의 손실측정함수는 Cross Entropy가 사용됩니다.

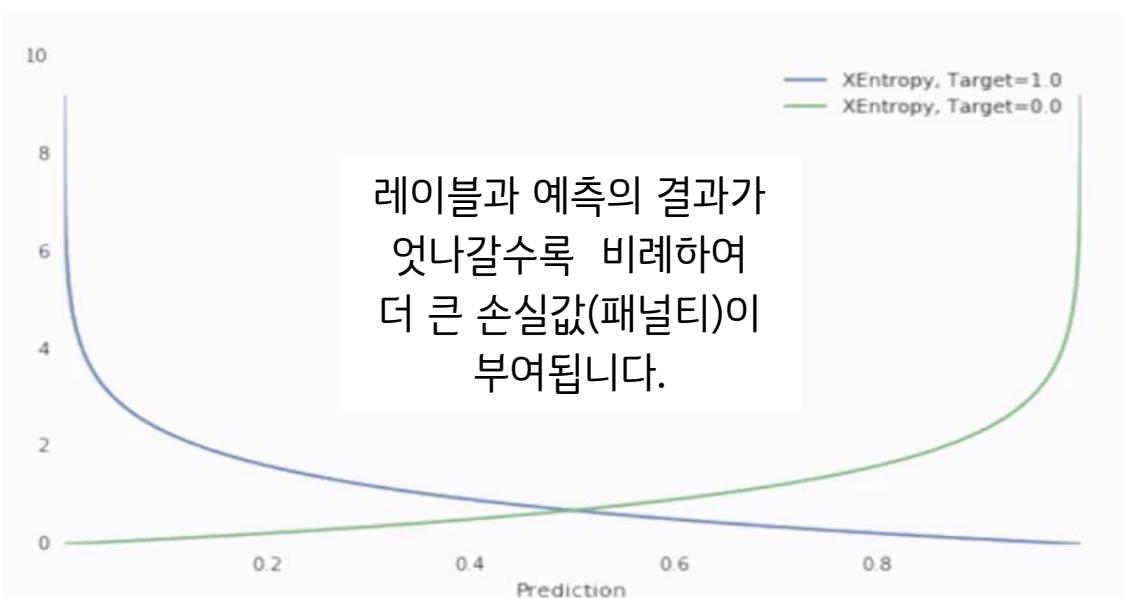
■ MSE(Mean Squared Error)



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

레이블 값 (실제값) \downarrow \hat{y}_i (모델이 예측한 값) \downarrow

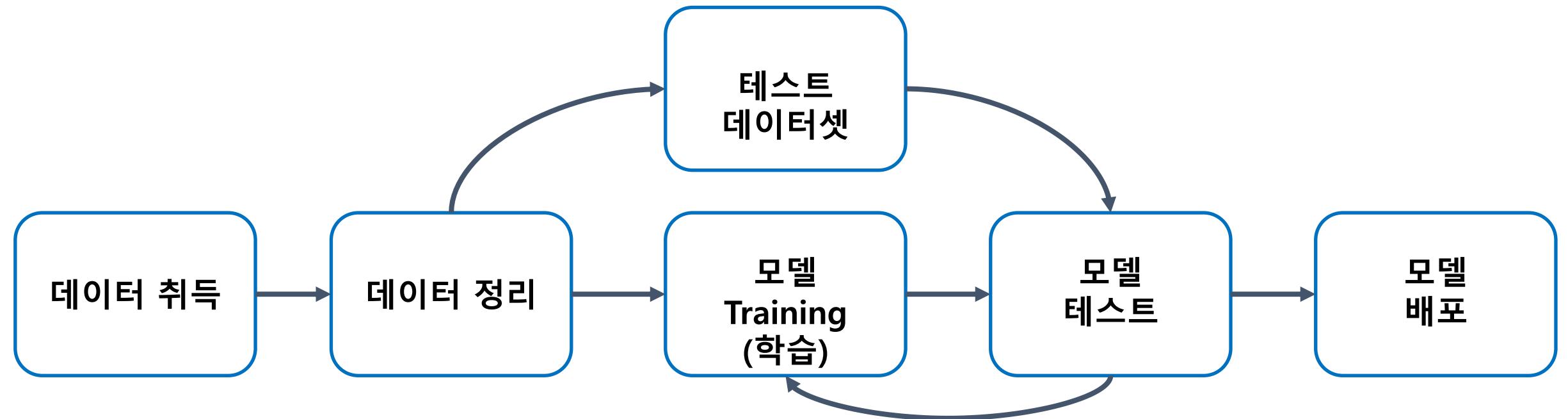
■ Cross Entropy



$$\frac{-1}{N} \times \sum_{i=1}^N y_i \times \log(\hat{y}_i) + (1 - y_i) \times \log(1 - \hat{y}_i)$$

※ 크로스 엔트로피 참고자료
<https://youtu.be/r3iRRQ2ViQM>
<https://youtu.be/Jt5BS71uVfl>

머신러닝 프로세스



사이킷런(Scikit-learn)

가장 인기있는 머신러닝 패키지이며, 많은 머신러닝 알고리즘이 내장되어 있습니다.

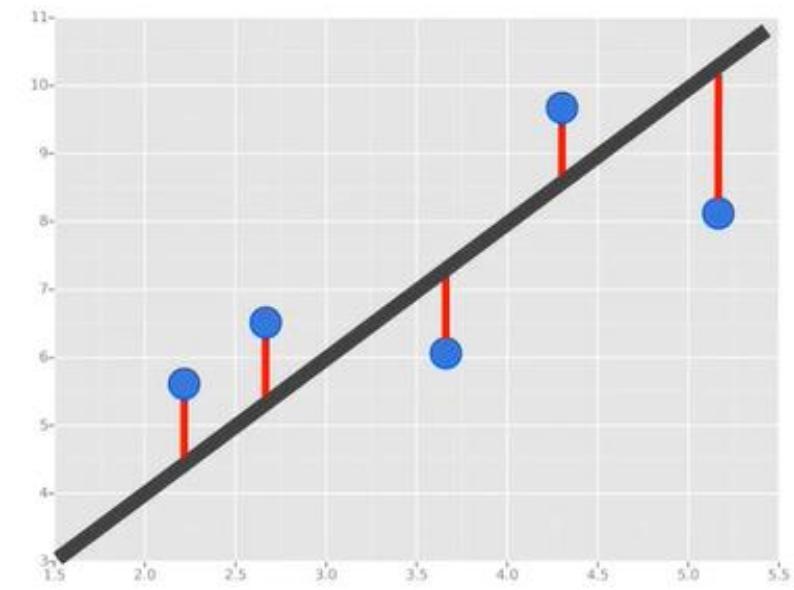
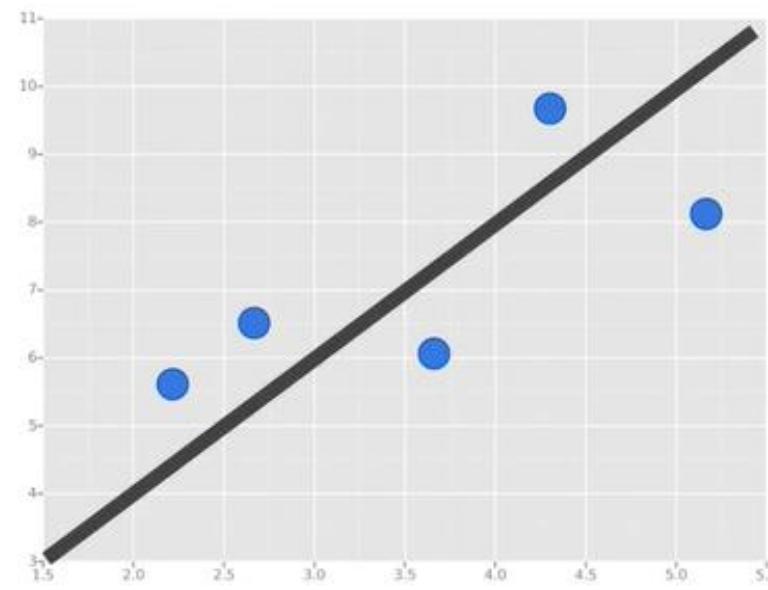
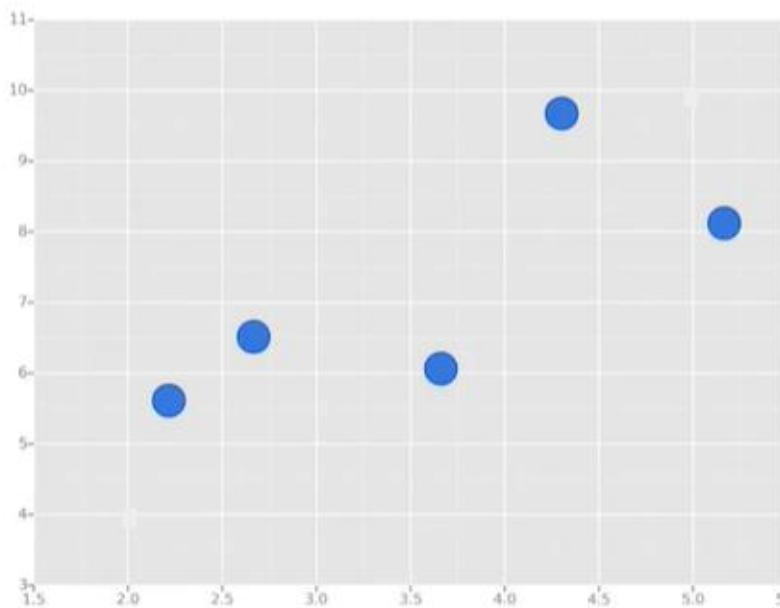
The screenshot shows the official website for scikit-learn (<https://scikit-learn.org>). The top navigation bar includes links for 'Install', 'User Guide', and 'API'. Below the header, there's a main title 'scikit-learn' and subtitle 'Machine Learning in Python'. A blue banner highlights the following features:

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

The page is divided into three main sections: Classification, Regression, and Clustering, each with a brief description and associated applications and algorithms.

- Classification:** Identifying which category an object belongs to. Applications include spam detection and image recognition. Algorithms include SVM, nearest neighbors, random forest, and more. It features a grid of 9x6 plots comparing various classification models like K-Nearest Neighbors, Logistic Regression, and Decision Trees against a baseline model.
- Regression:** Predicting a continuous-valued attribute associated with an object. Applications include drug response and stock prices. Algorithms include SVR, nearest neighbors, random forest, and more. It features a plot titled 'Boosted Decision Tree Regression' showing a green line (n_estimators=1) and a red line (n_estimators=300) fitting a series of black dots.
- Clustering:** Automatic grouping of similar objects into sets. Applications include customer segmentation and grouping experiment outcomes. Algorithms include k-Means, spectral clustering, mean-shift, and more. It features a scatter plot of digits data points colored by cluster assignment, with white crosses marking the centroids.

선형 회귀(Linear Regression)

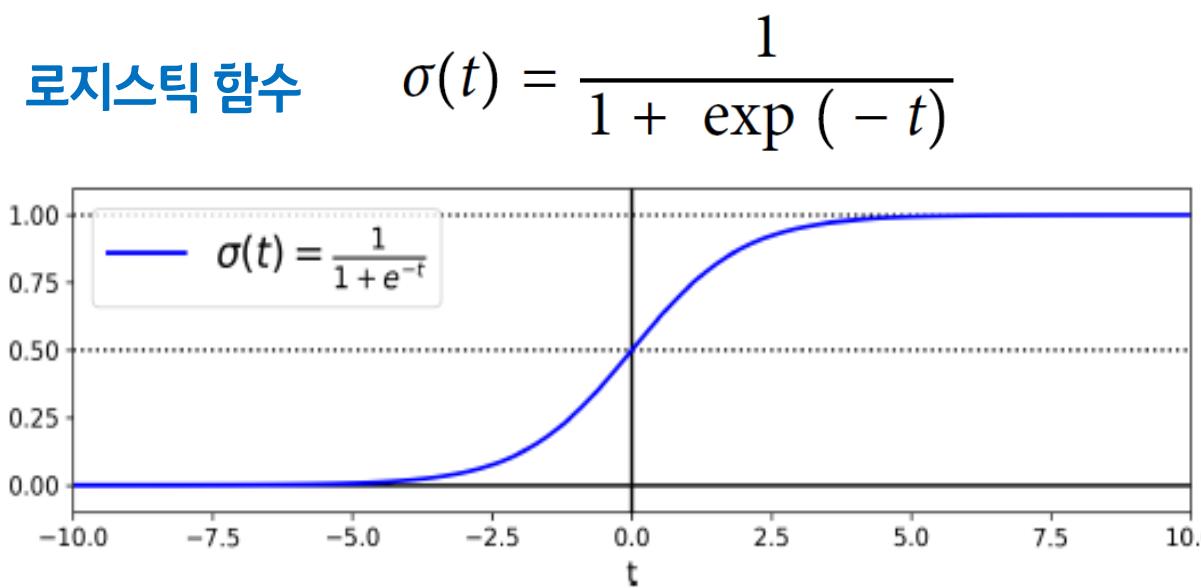
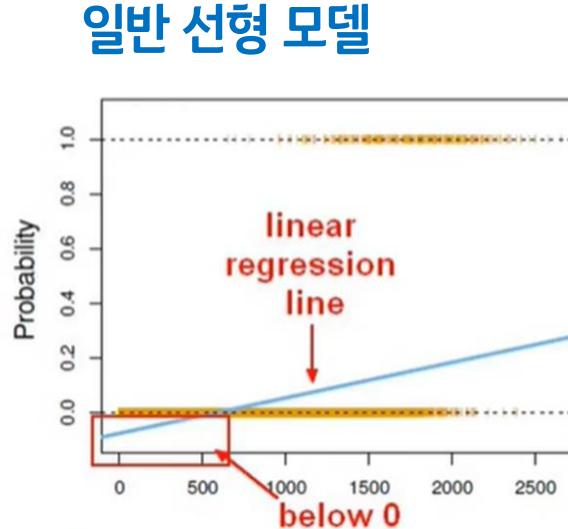


선형 회귀(Linear Regression)

```
from sklearn.model_selection import train_test_split  
from sklearn.linear_model import LinearRegression  
  
X = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ]).reshape(-1,1)  
y = np.array([13, 25, 34, 47, 59, 62, 79, 88, 90, 100])  
  
X_train, X_test, y_train, y_test = train_test_split(X, y,  
                                                    test_size=0.3, random_state=42)  
model = LinearRegression()  
model.fit(X_train, y_train)  
predictions = model.predict(X_test)
```

로지스틱 회귀(Logistic Regression)

- 이진 분류 규칙은 0과 1의 두 클래스를 갖는 것으로, 일반 선형 회귀 모델을 이진분류에 사용할 수 없습니다.
- 대신 선형 회귀를 로지스틱 회귀 곡선으로 변환 할 수 있으며, 로지스틱 회귀 곡선은 0과 1 사이에서만 이동할 수 있으므로 분류에 사용할 수 있습니다.
- 로지스틱 회귀는 선형 회귀처럼 바로 결과를 출력하지 않고 로지스틱(logistic)을 출력합니다.
- 로지스틱 회귀는 샘플이 특정 데이터에 속할 확률을 추정(이진분류)하는 데 사용됩니다.
- 추정 확률이 50%가 넘으면 모델은 그 샘플이 해당 클래스에 속한다고 예측합니다.



로지스틱 확률모델

$$\hat{p} = h_{\theta}(\mathbf{x}) = \sigma(\mathbf{x}^T \boldsymbol{\theta})$$
$$\hat{y} = \begin{cases} 0 & \text{if } \hat{p} < 0.5 \\ 1 & \text{if } \hat{p} \geq 0.5 \end{cases}$$

로지스틱 회귀(Logistic Regression)

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

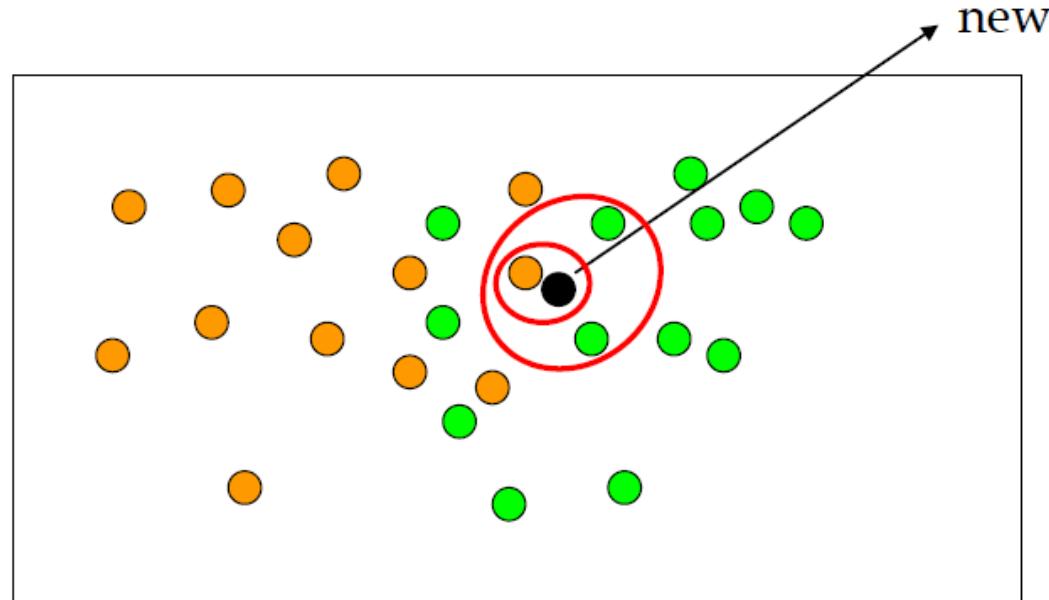
X = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ]).reshape(-1,1)
y = np.array([0, 0, 0, 0, 0, 1, 1, 1, 1, 1])

X_train, X_test, y_train, y_test = train_test_split(
    train.drop('Survived',axis=1),
    train['Survived'], test_size=0.30, random_state=42)

model = LogisticRegression()
model.fit(X_train, y_train)
predictions = model.predict(X_test)
```

K-최근접 이웃(K-Nearest Neighbor)

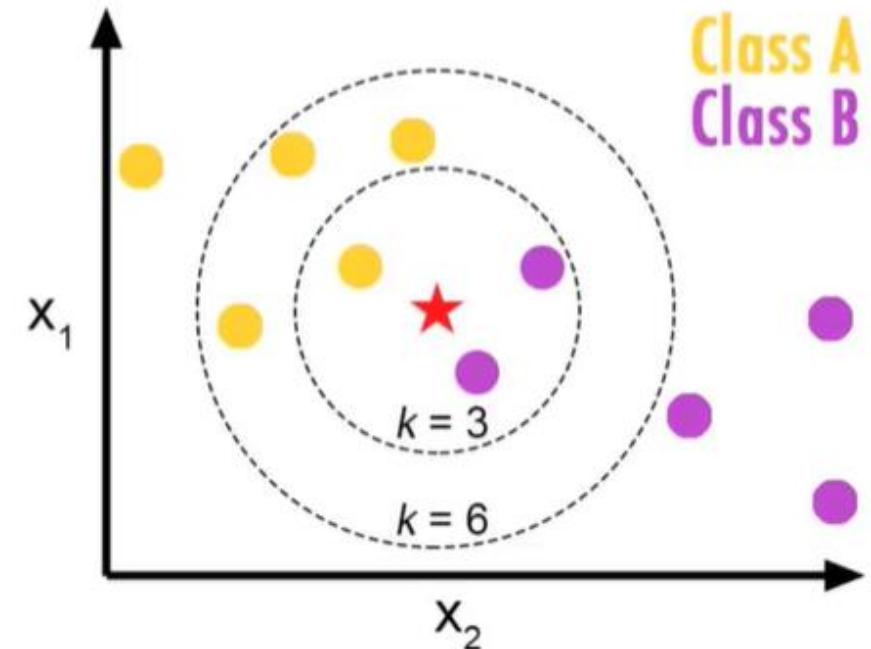
- KNN은 새로운 데이터가 주어졌을 때 기존 데이터 가운데 가장 가까운 k개 이웃의 정보로 새로운 데이터를 예측하는 방법론입니다. 아래 그림처럼 검은색 점의 범주 정보는 주변 이웃들을 가지고 추론해낼 수 있습니다.
- 만약 k가 1이라면 오렌지색, k가 3이라면 녹색으로 분류(classification)하는 것입니다.
- 만약, 회귀(regression) 문제라면 이웃들 종속변수(y)의 평균이 예측값이 됩니다.



K-최근접 이웃(K-Nearest Neighbor)

- 알고리즘이 간단하며 큰 데이터셋과 고차원 데이터에 적합하지 않은 단점이 있습니다.

```
from sklearn.neighbors import KNeighborsClassifier  
  
knn = KNeighborsClassifier(n_neighbors=3)  
  
knn.fit(X_train,y_train)  
  
pred = knn.predict(X_test)
```



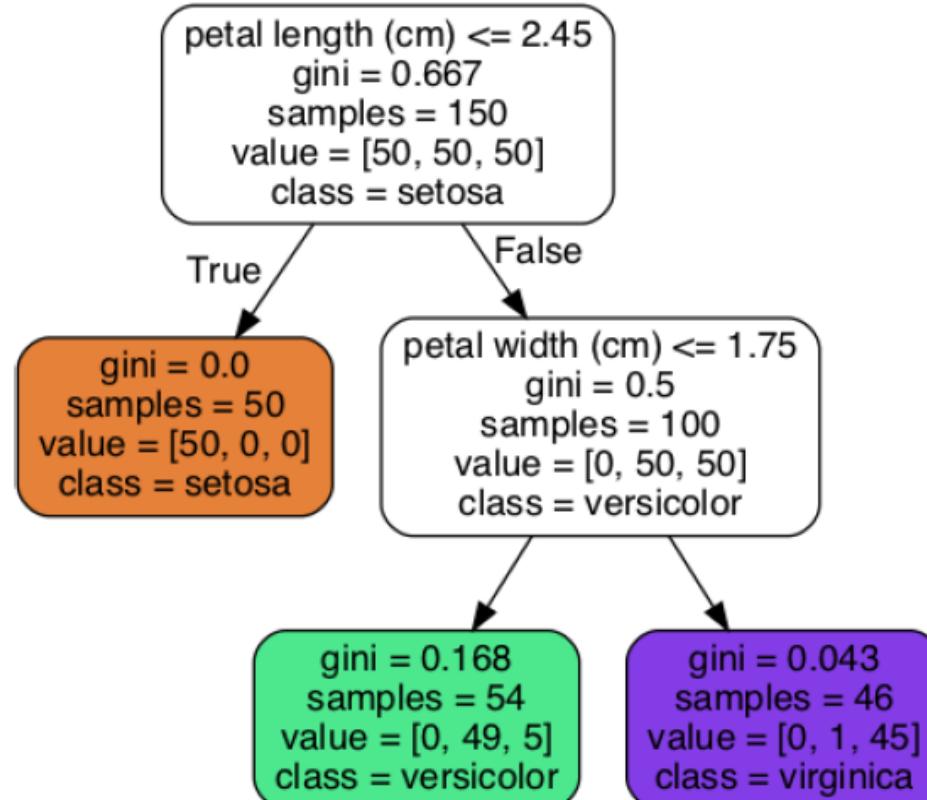
결정 트리(Decision Tree)

- 분류와 회귀 작업이 가능한 다재다능한 머신러닝 알고리즘입니다.
- 복잡한 데이터셋도 학습할 수 있으며, 강력한 머신러닝 알고리즘인 랜덤 포레스트의 기본 구성 요소입니다.

■ 붓꽃(Iris)



■ 붓꽃 분류 결정트리



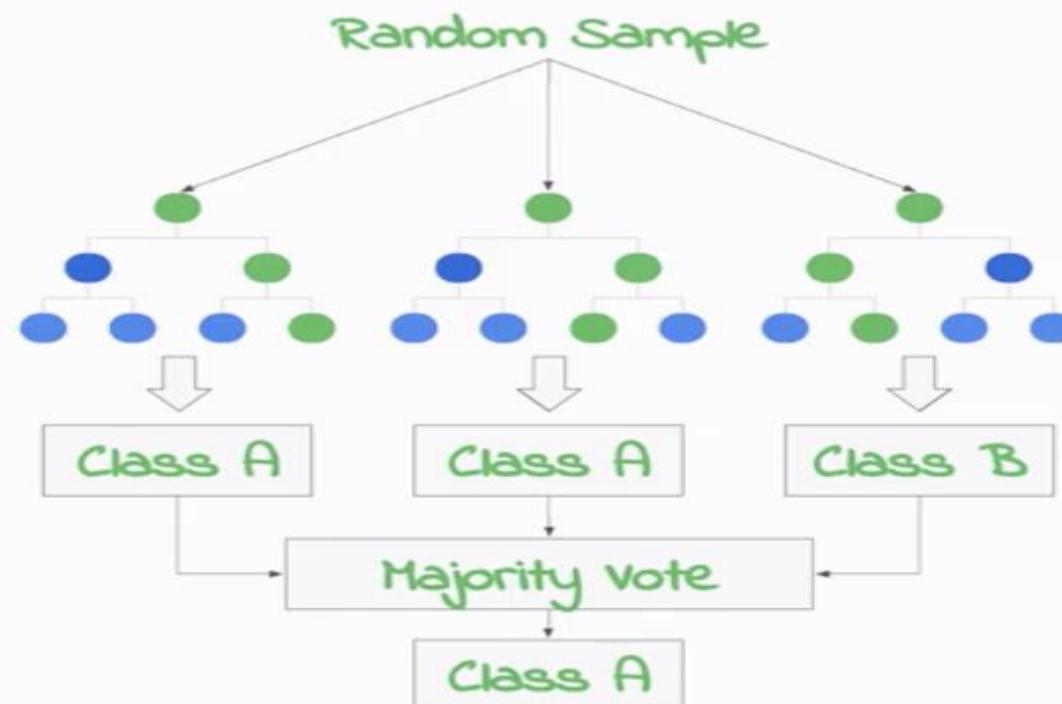
결정 트리(Decision Tree)

```
from sklearn.datasets import load_iris  
from sklearn.tree import DecisionTreeClassifier  
  
iris = load_iris()  
X = iris.data[:, 2:] # petal length and width  
y = iris.target  
  
dtree_clf = DecisionTreeClassifier(max_depth=2)  
dtree_clf.fit(X, y)  
  
dtree_clf.predict_proba([[5, 1.5]])  
[Out] array([[0. , 0.90740741, 0.09259259]])  
  
dtree_clf.predict([[5, 1.5]])  
[Out] array([1])
```

랜덤 포레스트(Random Forest)

- 일련의 예측기(분류, 회귀모델)로 부터 예측을 수집하면 가장 좋은 모델 하나보다 더 좋은 예측을 얻을 수 있습니다.
- 일련의 예측기를 앙상블이라고 부르며, 결정 트리의 앙상블을 랜덤 포레스트라고 합니다.
- 훈련 세트로 부터 무작위로 각기 다른 서브셋을 만들어 일련의 결정 트리 분류기를 훈련시킬 수 있습니다.

Random forest: Strong learner
from many weak learners



랜덤 포레스트(Random Forest)

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

iris = datasets.load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
X_train, X_test, y_train, y_test = train_test_split(
    df[iris.feature_names], iris.target, test_size=0.25,
    stratify=iris.target, random_state=42)
rf = RandomForestClassifier(n_estimators=50,
                            max_depth=20,
                            random_state=42)

rf.fit(X_train, y_train)
predicted = rf.predict(X_test)
accuracy = accuracy_score(y_test, predicted)
```

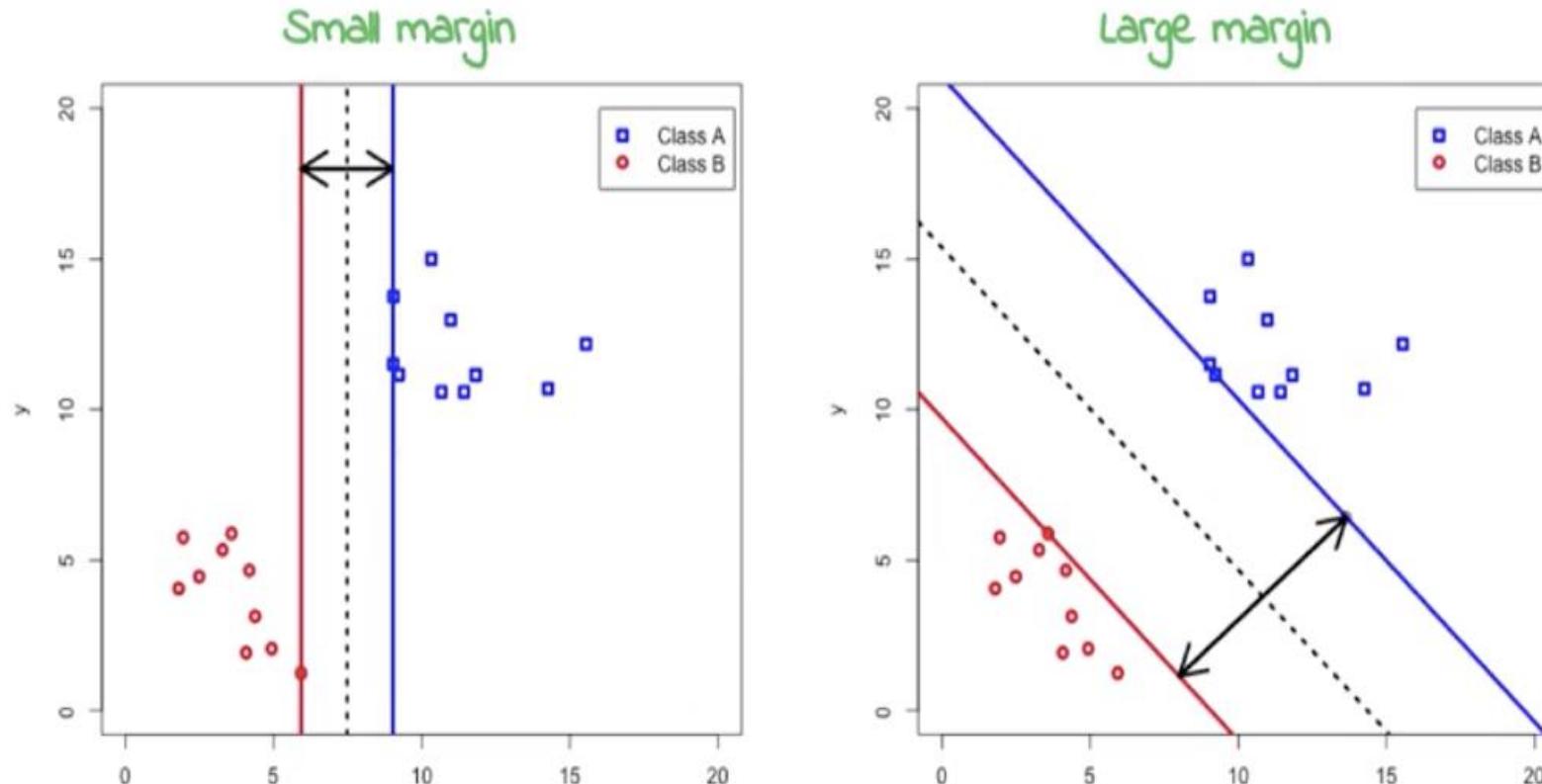
shift+tab : 함수 설명 보기

```
Init signature:
RandomForestClassifier(
    n_estimators=100,
    *,
    criterion='gini',
    max_depth=None,
    min_samples_split=2,
    min_samples_leaf=1,
    min_weight_fraction_leaf=0.0,
    max_features='auto',
```

서포트 벡터 머신(SVM)

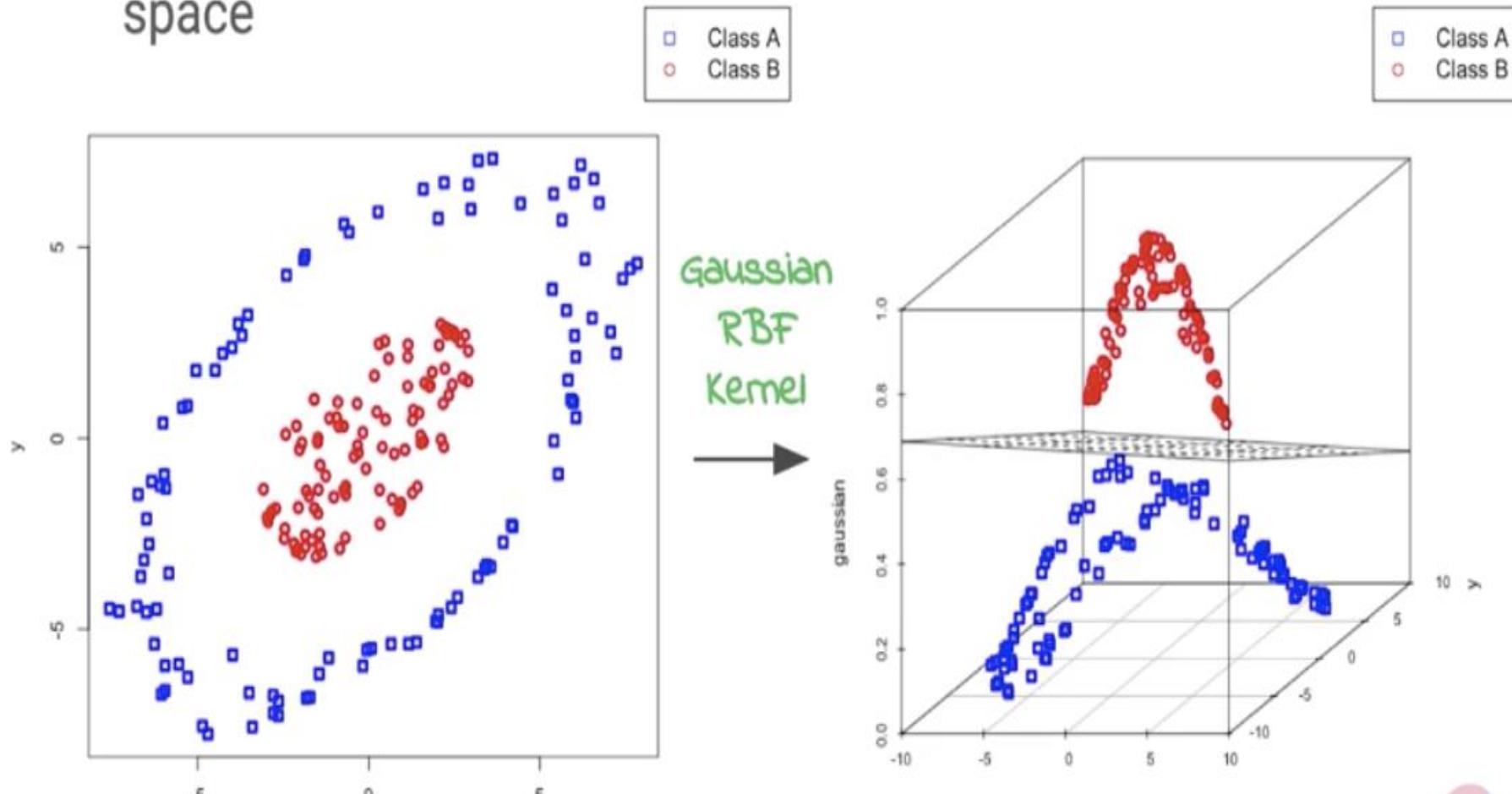
- 강력하고 선형이나 비선형 분류, 회귀, 이상치 탐색에도 사용할 수 있는 다목적 머신러닝 모델입니다.
- SVM은 특히 복잡한 분류 모델에 잘 들어 맞으며 작거나 중간 크기의 데이터셋에 적합합니다.

SVMs maximize the margin between two classes

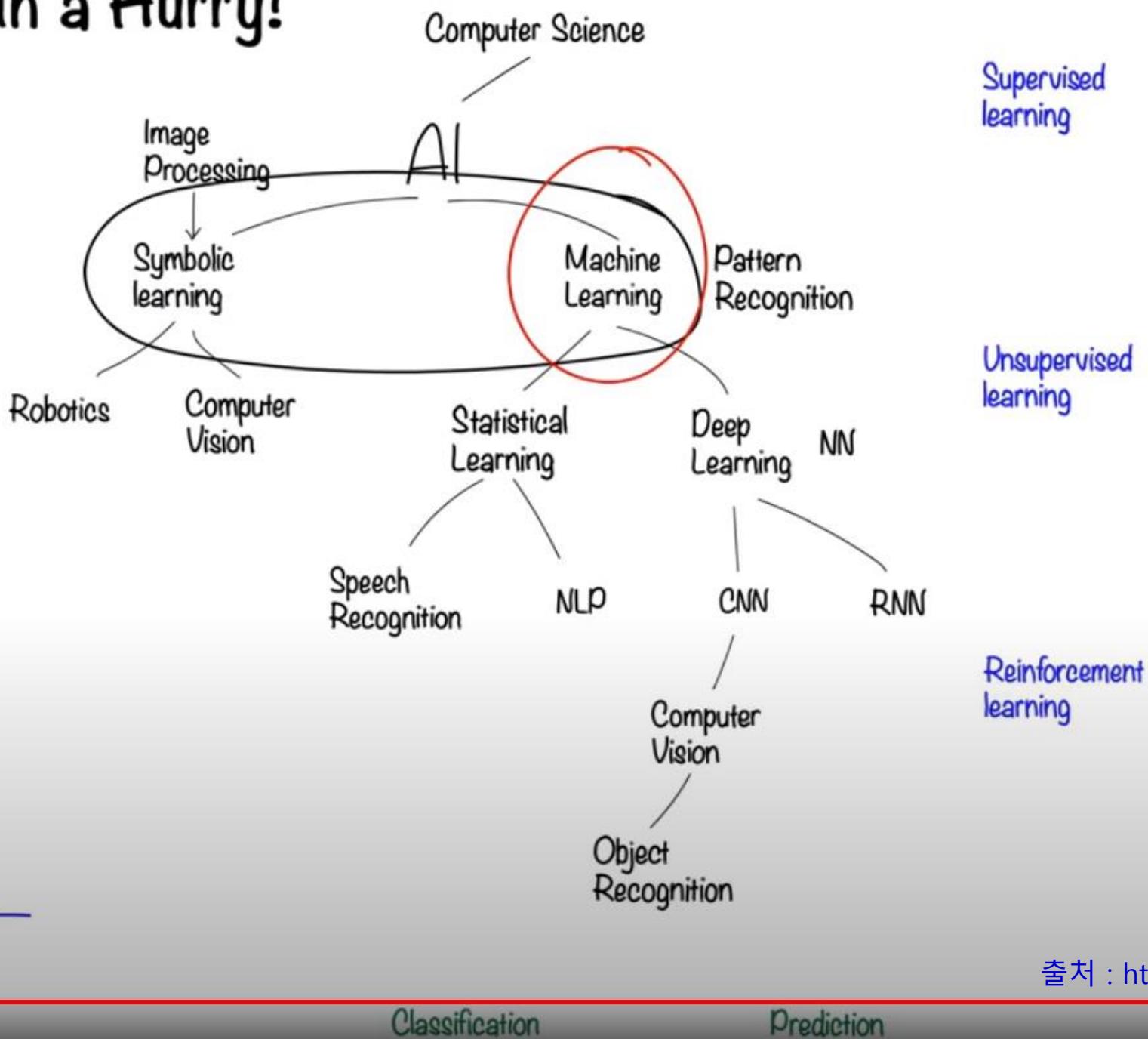


서포트 벡터 머신(SVM)

Kernels transform the input space into a more usable feature space

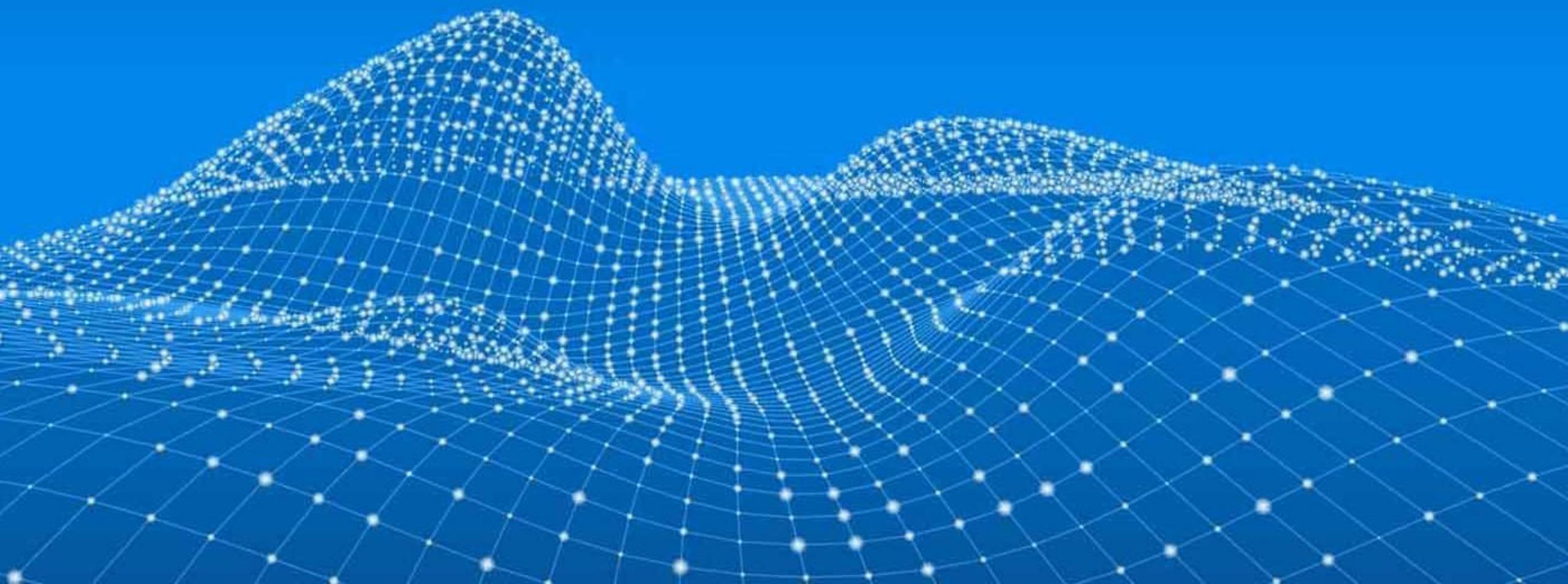


AI - For People In a Hurry!





딥러닝 원리와 심층신경망



딥러닝 이론

<https://github.com/kgpark88/deeplearning>

퍼셉트론 모델(Perceptron Model)

뉴럴 네트워크(Neural Network)

활성화 함수(Activation Function)

손실함수(Loss Function)

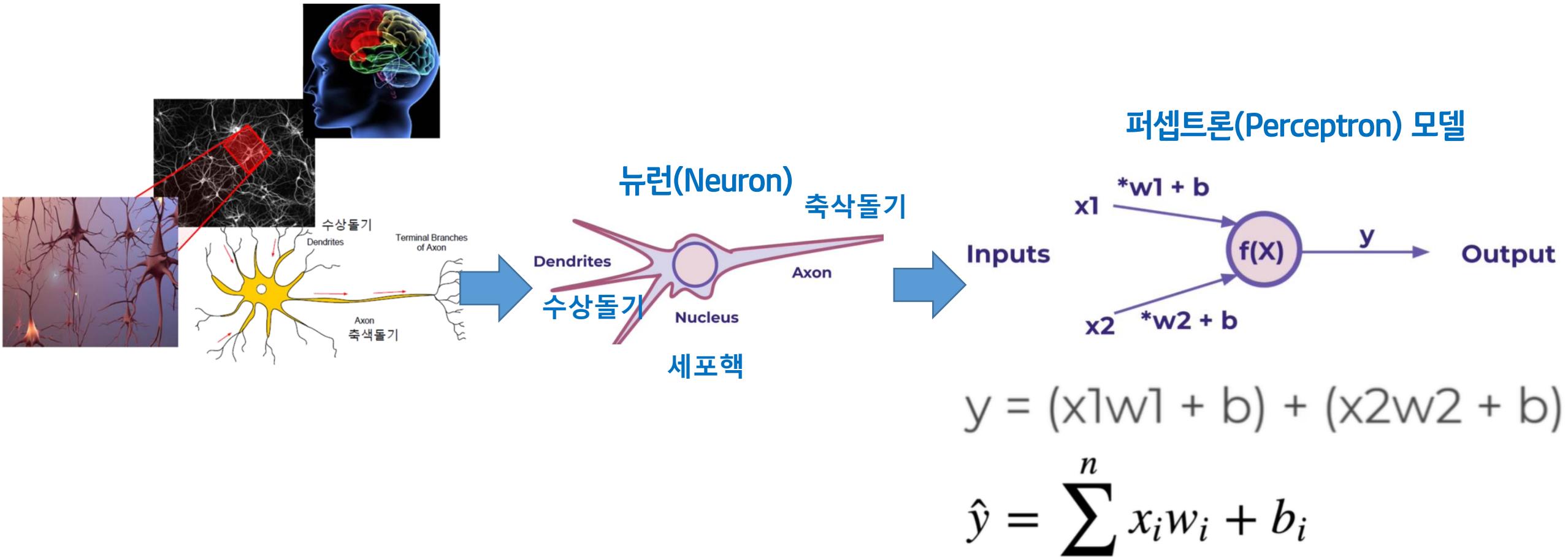
피드 포워드 신경망(Feed Forward Network)

역전파(Back Propagation)

경사하강법(Gradient Descent)

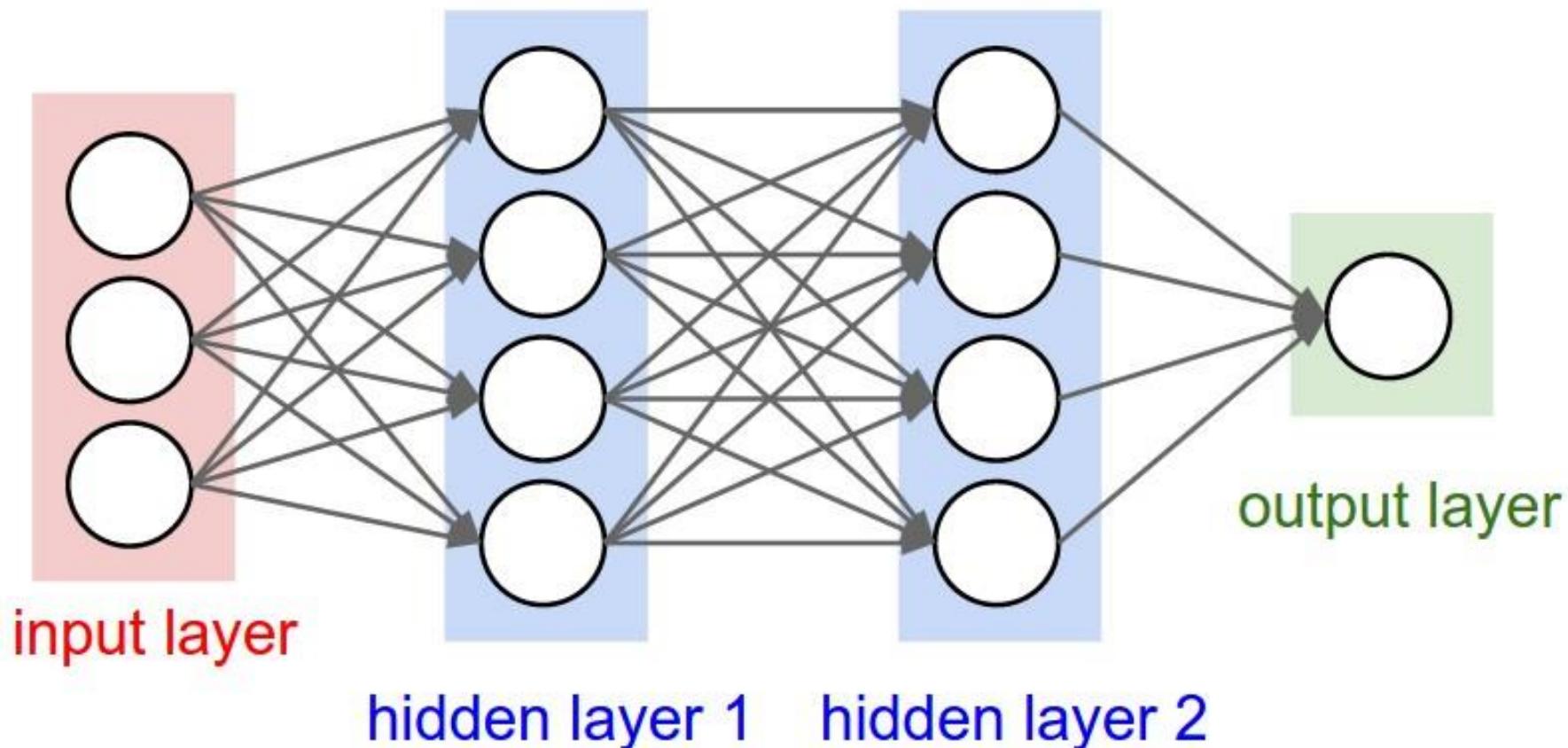
퍼셉트론(Perceptron)

퍼셉트론은 사람 두뇌에 있는 뉴런을 모델링한 것으로 간단한 함수를 학습할 수 있습니다.

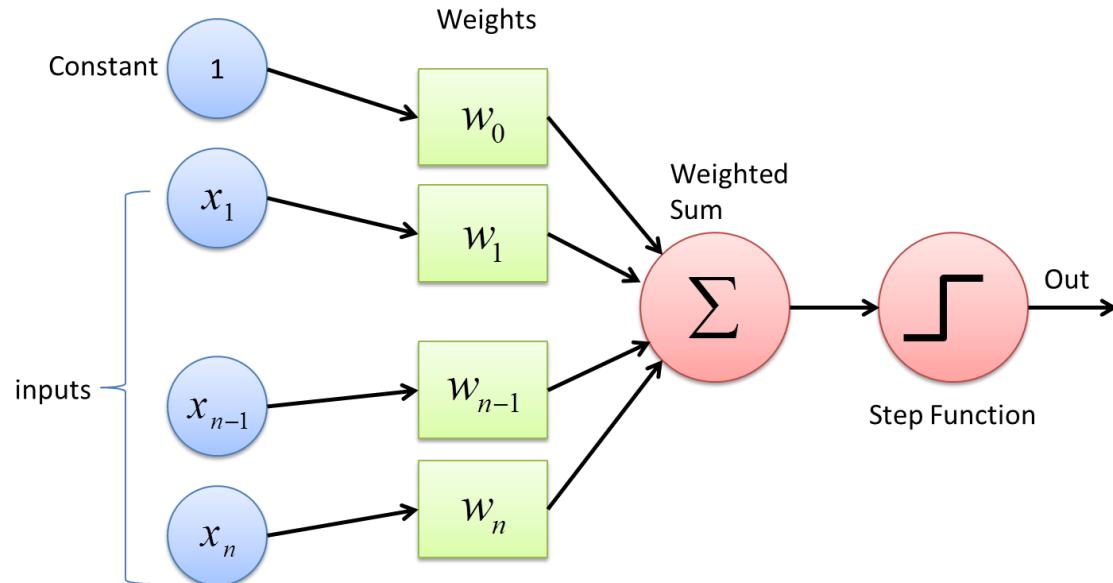


심층 신경망(Deep Neural Network)

입력층과 출력층 사이에 여러 개의 은닉층(hidden layer)로 이루어진 인공신경망으로
신경망 출력에 비선형 활성화 함수를 추가하여 복잡한 비선형 관계를 모델링 할 수 있습니다.



활성화 함수(Activation function)

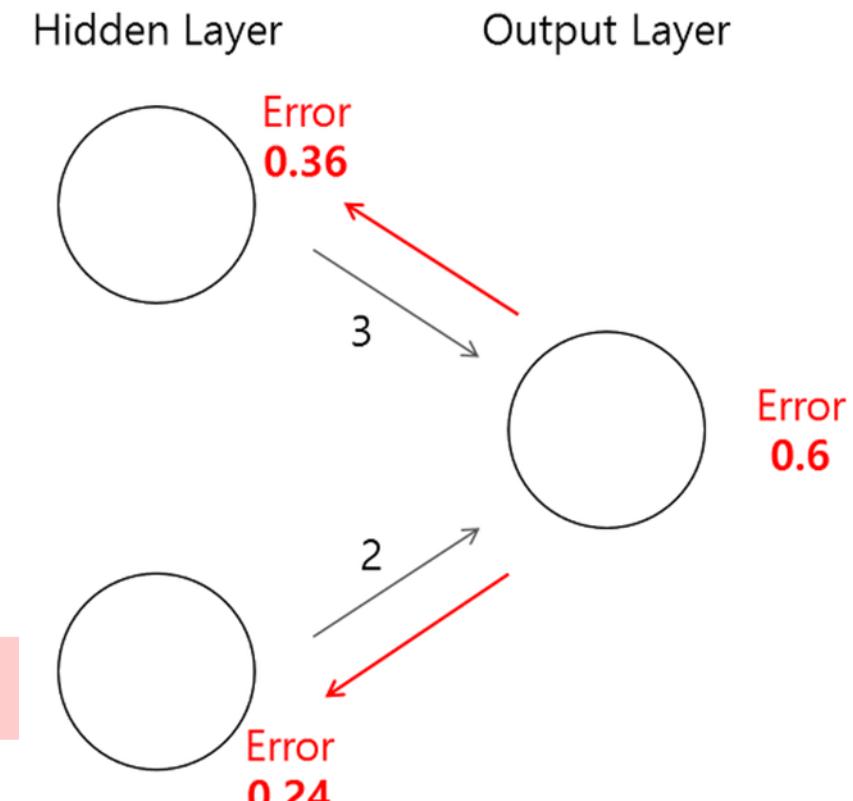
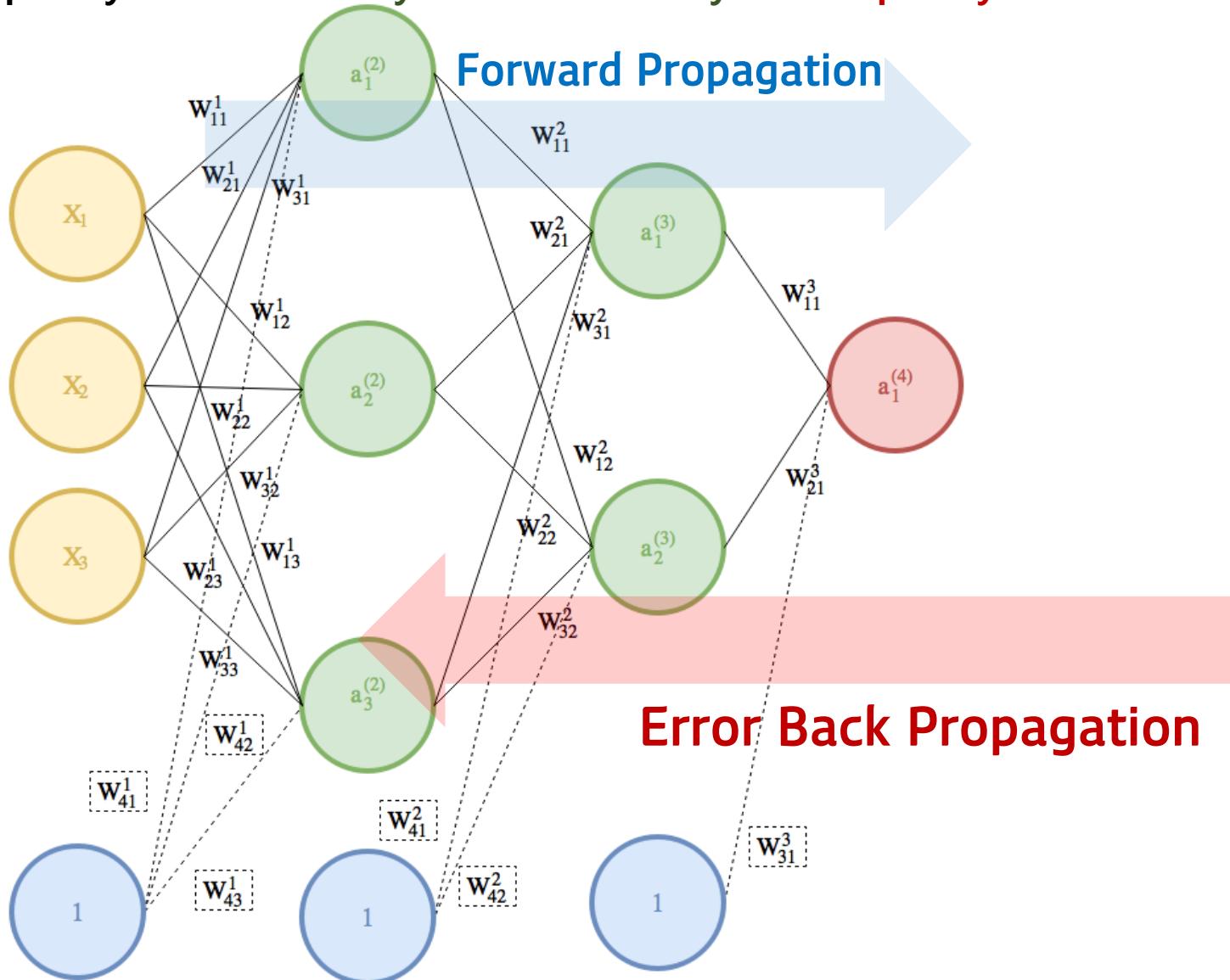


Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Logistic (a.k.a. Sigmoid or Soft step)		$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$
Rectified linear unit (ReLU) ^[12]		$f(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases} = \max\{0, x\} = x\mathbf{1}_{x>0}$

https://en.wikipedia.org/wiki/Activation_function

오차 역전파(Error Backpropagation)

Input layer hidden layer hidden layer output layer



손실함수(Loss Function)

손실함수는 신경망 학습의 목적함수로 출력값(예측값)과 정답(실제값)의 차이를 계산합니다.

■ 회귀(regression)

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

$$\text{MAE} = \frac{\sum_{i=1}^n |y_i - \hat{y}_i|}{n}$$

- 손실함수는 신경망 학습의 목적으로(목적함수) 출력값과 정답의 차이를 계산합니다.

■ 분류(classification)

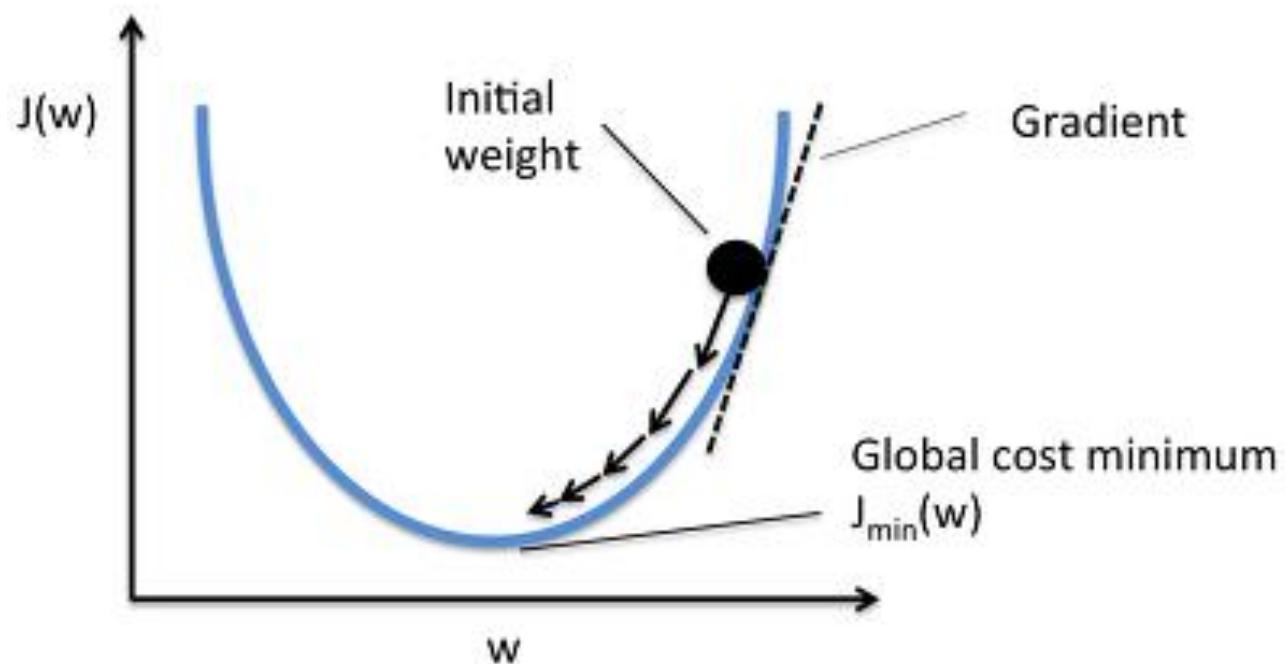
이진분류 : binary cross-entropy

$$L = -\frac{1}{N} \sum_{i=1}^N t_i \log(y_i) + (1 - t_i) \log(1 - y_i)$$

다중 분류: categorical cross-entropy $L = -\frac{1}{N} \sum_{j=1}^N \sum_{i=1}^C t_{ij} \log(y_{ij})$

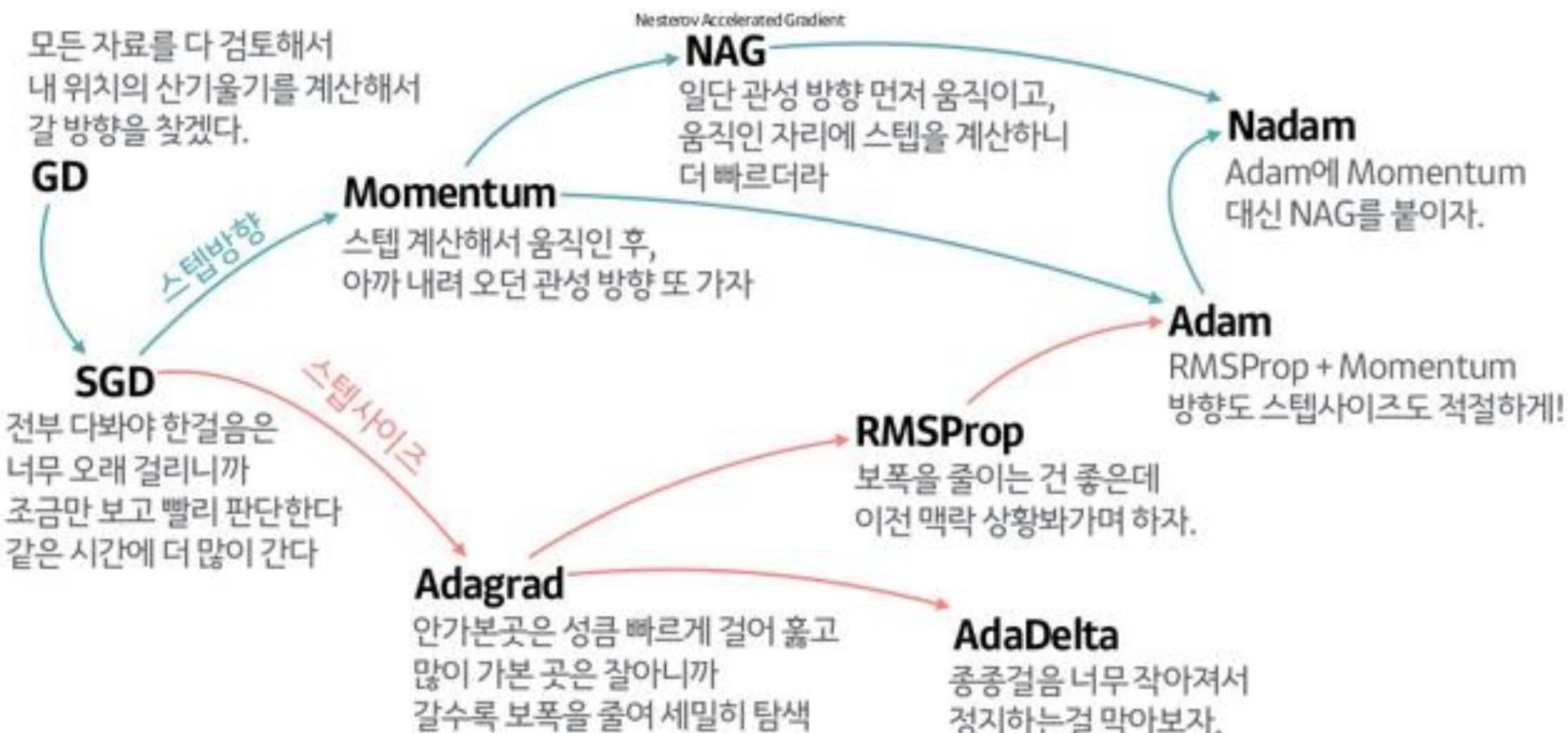
경사하강법(Gradient Descent)

뉴럴넷이 가중치 파라미터들을 최적화 하는 방법으로 손실함수의 현 가중치에서 기울기(gradient)를 구해서 Loss을 줄이는 방향으로 업데이트 해 나갑니다.



$$\text{weight의 업데이트} = \frac{\text{에러 낮추는 방향}}{\text{(decent)}} \times \frac{\text{한발자국 크기}}{\text{(learning rate)}} \times \frac{\text{현 지점의 기울기}}{\text{(gradient)}}$$
$$-\gamma \nabla F(\mathbf{a}^n)$$

최적화 알고리즘(Optimization Algorithm)



딥러닝 용어 정리

딥러닝 학습의 목표는 모델에 입력값을 넣었을 때의 출력값이 최대한 정답과 일치하게 하는 것입니다.

■ 딥러닝의 학습방법

딥러닝 모델의 매개변수(weight, bias)를 무작위로 부여한 후,
반복학습을 통해 모델의 출력값을 정답과 일치하도록 매개변수를 조금씩 조정합니다.

■ 순전파(Forward Propagation)

딥러닝 모델에 값을 입력해서 출력을 얻는 과정입니다.

■ 오차 역전파(Error Backpropagation)

실제값과 모델 결과값에서 오차를 구해서, 오차를 input 방향으로 보내서 가중치를 재업데이트 하는 과정입니다.

■ 손실함수(Loss Function)

손실함수는 신경망 학습의 목적으로(목적함수) 출력값과 정답의 차이를 계산합니다.

■ 과적합(overfitting)

생성된 모델이 학습 데이터와 지나치게 일치하여 새 데이터를 올바르게 예측하지 못하는 경우입니다.

딥러닝 용어 정리

■ 최적화(Optimization)

딥러닝 모델의 매개변수(weight, bias)를 조절해서 손실함수의 값을 최저로 만드는 과정으로 경사하강법(Gradient Descent)이 대표적입니다.

■ 경사하강법(gradient descent)

학습 데이터의 조건에 따라 모델의 매개변수를 기준으로 손실의 경사를 계산하여 손실을 최소화하는 기법입니다. 쉽게 설명하면, 경사하강법은 매개변수를 반복적으로 조정하면서 손실을 최소화하는 가중치와 편향의 가장 적절한 조합을 점진적으로 찾는 방식입니다.

■ 경사(gradients)

모든 독립 변수를 기준으로 한 편미분의 벡터입니다. 머신러닝에서 경사는 모델 함수의 편미분의 벡터입니다.

■ 일반화(generalization)

모델에서 학습에 사용된 데이터가 아닌 이전에 접하지 못한 새로운 데이터에 대해 올바른 예측을 수행하는 능력을 의미합니다.

딥러닝 용어 정리

■ 시퀀스 모델(sequence model)

입력에 순서 종속성이 있는 모델입니다

■ 밀집 연결층(Dense Layer)

완전 연결층(fully connected layer)이나 밀집 층(dense layer)라고 불리는 밀집 연결 층(densely connected layer)

■ 입력 레이어(input layer)

신경망의 첫 번째 레이어로서 입력 데이터를 수신합니다.

■ 하든 레이어(hidden layer)

신경망에서 입력 레이어(특성)와 출력 레이어(예측) 사이에 위치하는 합성 레이어입니다.

신경망에 하나 이상의 하든 레이어가 포함될 수 있습니다.

■ 드롭아웃(dropout)

신경망의 과적합을 방지하는 방법으로 단일 경사 스텝이 일어날 때마다

특정 네트워크 레이어의 유닛을 고정된 개수만큼 무작위로 선택하여 삭제합니다.

■ 출력 레이어(output layer)

신경망의 '최종' 레이어입니다. 이 레이어에 답이 포함됩니다.

딥러닝 용어 정리



1 Epoch : 모든 데이터 셋을 한 번 학습

1 iteration : 1회 학습

minibatch : 데이터 셋을 batch size 크기로 쪼개서 학습

ex) 총 데이터가 100개, batch size가 100이면,

1 iteration = 10개 데이터에 대해서 학습

1 Epoch = $100/\text{batch size} = 10 \text{ iteration}$

딥러닝 구현 절차

- ① 라이브러리 임포트(import)
- ② 데이터 가져오기>Loading the data)
- ③ 탐색적 데이터 분석(Exploratory Data Analysis)
- ④ 데이터 전처리(Data PreProcessing) : 데이터타입 변환, Null 데이터 처리, 누락데이터 처리, 카테고리 데이터, 더미특성 생성, 특성 추출 (feature engineering) 등
- ⑤ 훈련/테스트 데이터 분할(Train Test Split)
- ⑥ 데이터 정규화(Normalizing the Data)
- ⑦ 모델 개발(Creating the Model)
- ⑧ 모델 성능 평가(Evaluating Model Performance)

■ 심층신경망 구현

■ 라이브러리 임포트

```
import pandas as pd  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
import tensorflow as tf  
from tensorflow.keras.models import Sequential  
from tensorflow.keras.layers import Dense, Activation, Dropout
```

■ 데이터 로드

```
df = pd.read_csv('data.csv')
```

■ 심층신경망 구현

■ 데이터 분석

```
df.info()  
df.isnull().sum()  
df.describe().transpose()  
df.corr()['MonthlyCharges'][:-1].sort_values().plot(kind='bar')  
sns.pairplot(df)
```

■ 데이터 전처리

```
df.drop('customerID', axis=1, inplace=True)  
df['TotalCharges'].replace([' '], ['0'], inplace=True)  
df['TotalCharges'] = df['TotalCharges'].astype(float)  
df['Churn'].replace(['Yes', 'No'], [1, 0], inplace=True)
```

심층신경망 구현

■ 데이터 전처리

```
cols = ['gender', 'Partner', 'Dependents', 'PhoneService',
'MultipleLines', 'InternetService', 'OnlineSecurity', 'OnlineBackup',
'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies',
'Contract', 'PaperlessBilling', 'PaymentMethod']
```

```
dummies = pd.get_dummies(df[cols], drop_first=True)
df = df.drop(cols, axis=1)
df = pd.concat([df, dummies], axis=1)
```

```
# df = pd.get_dummies(df)
# cols = list(df.select_dtypes('object').columns)
```

심층신경망 구현

■ 훈련/테스트 데이터 분할

```
from sklearn.model_selection import train_test_split  
X = df.drop('Churn', axis=1).values  
y = df['Churn'].values  
  
X_train, X_test, y_train, y_test = train_test_split(  
    X, y,  
    test_size=0.3,  
    random_state=42)  
X_train.shape  
[Out] (4930, 30)  
  
y_train.shape  
[Out] (4930,)
```

심층신경망 구현

■ 데이터 정규화/스케일링(Normalizing/Scaling)

```
from sklearn.preprocessing import MinMaxScaler
```

```
scaler = MinMaxScaler()  
scaler.fit(X_train)  
X_train = scaler.transform(X_train)  
X_test = scaler.transform(X_test)
```

심층신경망 구현

■ 모델 구성

```
model = Sequential()
```

input Layer

```
model.add(Dense(64, activation='relu', input_shape=(30,)))
```

hidden Layer

```
model.add(Dense(64, activation='relu'))
```

hidden Layer

```
model.add(Dense(32, activation='relu'))
```

output Layer

```
model.add(Dense(1, activation='sigmoid'))
```

심층신경망 구현

■ 모델 구성 - 과적합 방지

```
model = Sequential()  
model.add(Dense(128, activation='relu', input_shape=(30,)))  
model.add(Dropout(0.5))  
model.add(Dense(64, activation='relu'))  
model.add(Dropout(0.5))  
model.add(Dense(64, activation='relu'))  
model.add(Dropout(0.5))  
model.add(Dense(32, activation='relu'))  
model.add(Dropout(0.5))  
  
model.add(Dense(1, activation='sigmoid'))
```

심층신경망 구현

■ 모델 컴파일 - 이진 분류 모델

```
model.compile(optimizer='adam',  
              loss='binary_crossentropy',  
              metrics=['accuracy'])
```

■ 모델 컴파일 - 다중 분류 모델

```
model.compile(optimizer='adam',  
              loss='categorical_crossentropy',  
              metrics=['accuracy'])
```

■ 모델 컴파일 - 예측 모델

```
model.compile(optimizer='adam',  
              loss='mse')
```

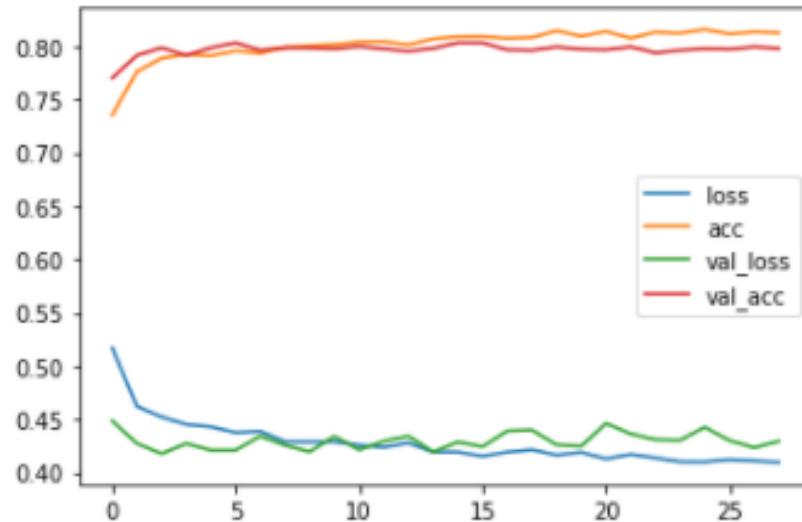
심층신경망 구현

■ 모델 훈련

```
model.fit(X_train, y_train,  
          validation_data=(X_test, y_test), epochs=20, batch_size=10)
```

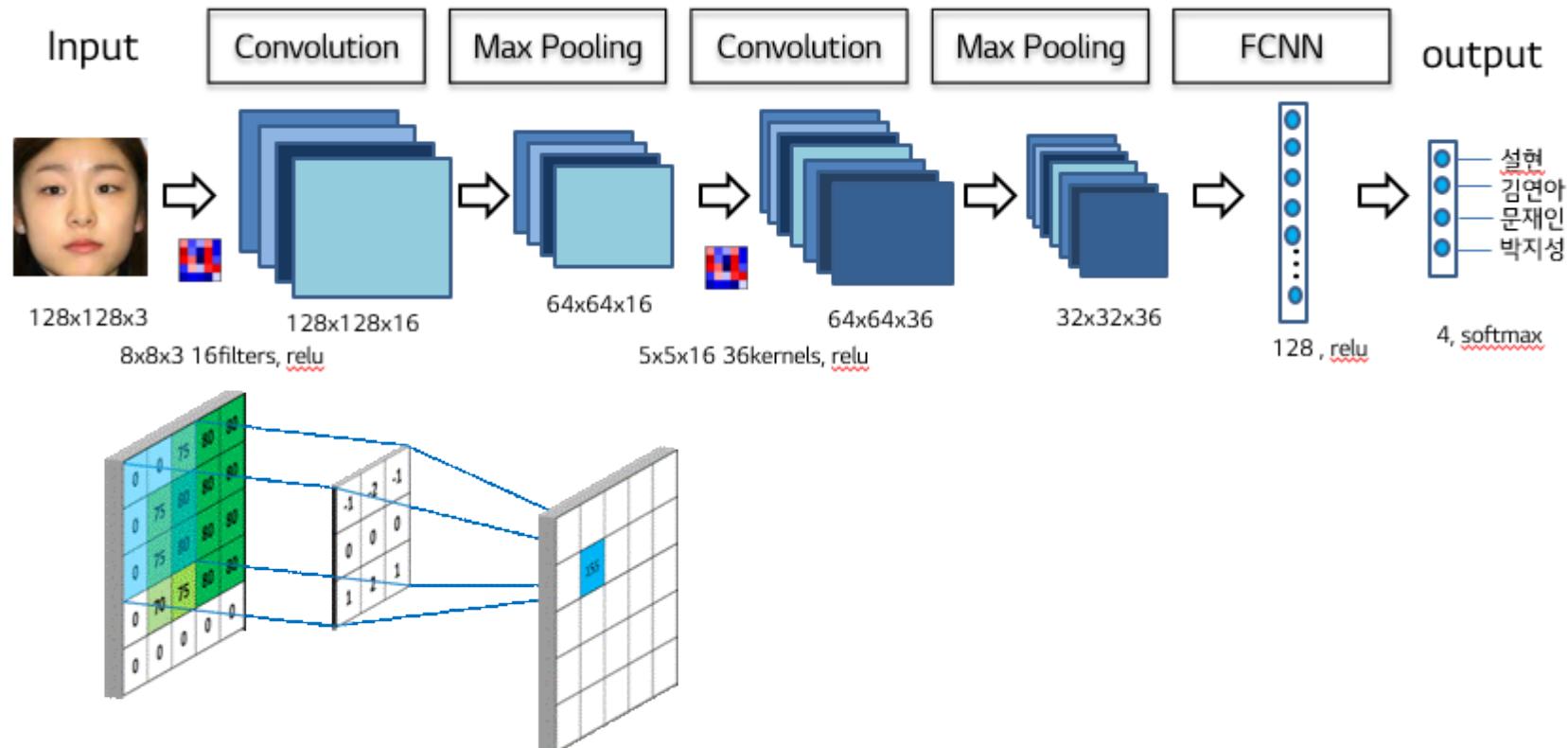
■ 모델 성능 평가

```
losses = pd.DataFrame(model.history.history)  
losses[['loss','val_loss']].plot()
```



합성곱 신경망(CNN, Convolutional Neural Network)

사람의 시각 피질 메커니즘에 영감을 받아 설계된 이미지, 영상등을 인식하는 신경망 모델
Convolution층에서는 각 filter가 입력 이미지의 픽셀 전체를 차례로 훑고 지나가며
linear combination을 진행하고 Feature Map을 구성합니다.

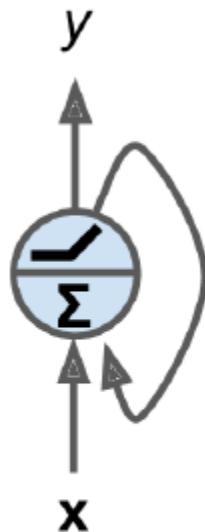


순환 신경망(RNN, Recurrent Neural Network)

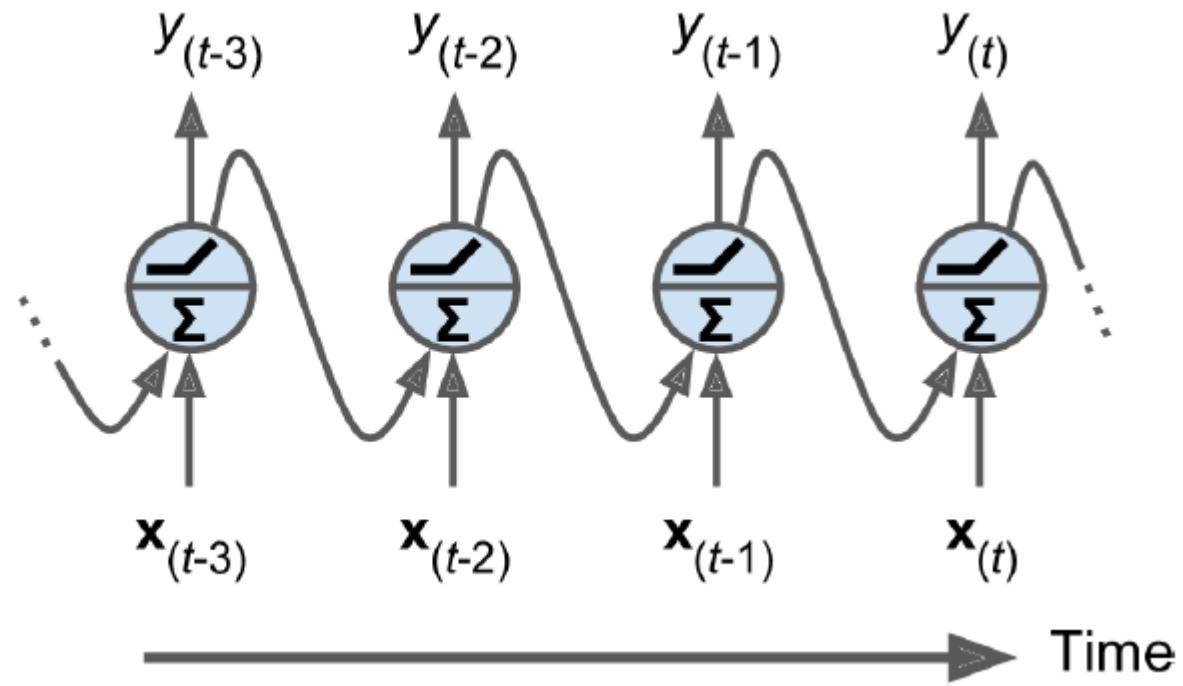
순환신경망은 고정 길이 입력이 아닌 임의 길이를 가진 시퀀스를 다룰 수 있습니다.

순환신경망은 시계열 데이터를 분석해서 미래값을 예측하고 문장, 오디오를 입력으로 받아 자동번역, 자연어처리에 유용합니다.

■ 순환뉴런



■ 순환뉴런을 타임 스텝으로 펼친 모습

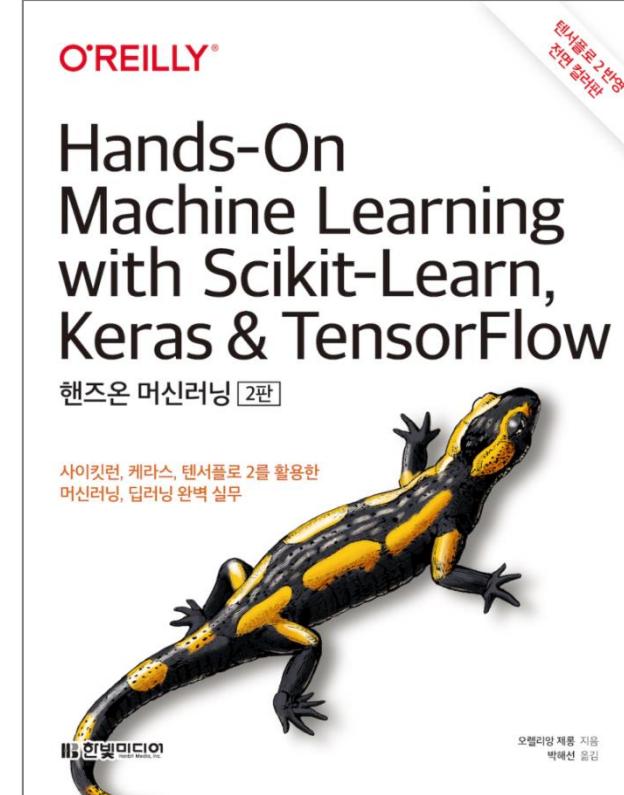


STUDY 추천자료

<https://wikibook.co.kr/mymlrev/>



<https://bit.ly/36Ltr2X>



<https://www.youtube.com/user/TheEasyoung>

<https://tensorflow.blog/handson-ml2/>



수고하셨습니다.

감사합니다.