

ARTIFICIAL INTELLIGENCE

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas at dolor nunc. consequat a gravida non, lacinia vel mi. Fusce semper ex vitae bibendum lacinia.

read more

DELPHI

인공지능

환경설정_딤러닝 개요

BRAINSTORM

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas at dolor nunc. consequat a gravida non, lacinia vel mi. Fusce semper ex vitae bibendum lacinia.

read more

강성관 (silicon1@hanmail.net)

환경 설정




Anaconda 설치하기

- Anaconda에는 Python인 다양한 라이브러리가 포함되어 있어 Python을 설치하지 않았다면 Anaconda를 설치하는 것을 권장함.
- <https://www.anaconda.com/download/#windows>에서 윈도우 버전(64비트)에 맞는 파이썬을 다운로드 한다.
- Tensorflow가 64비트만 지원하므로 아나콘다도 64비트용을 설치한다.


Anaconda 5.3.1 For Windows Installer

Python 3.7 version *

 Download

[64-Bit Graphical Installer \(633 MB\) ?](#)
[32-Bit Graphical Installer \(510 MB\)](#)

Python 2.7 version *

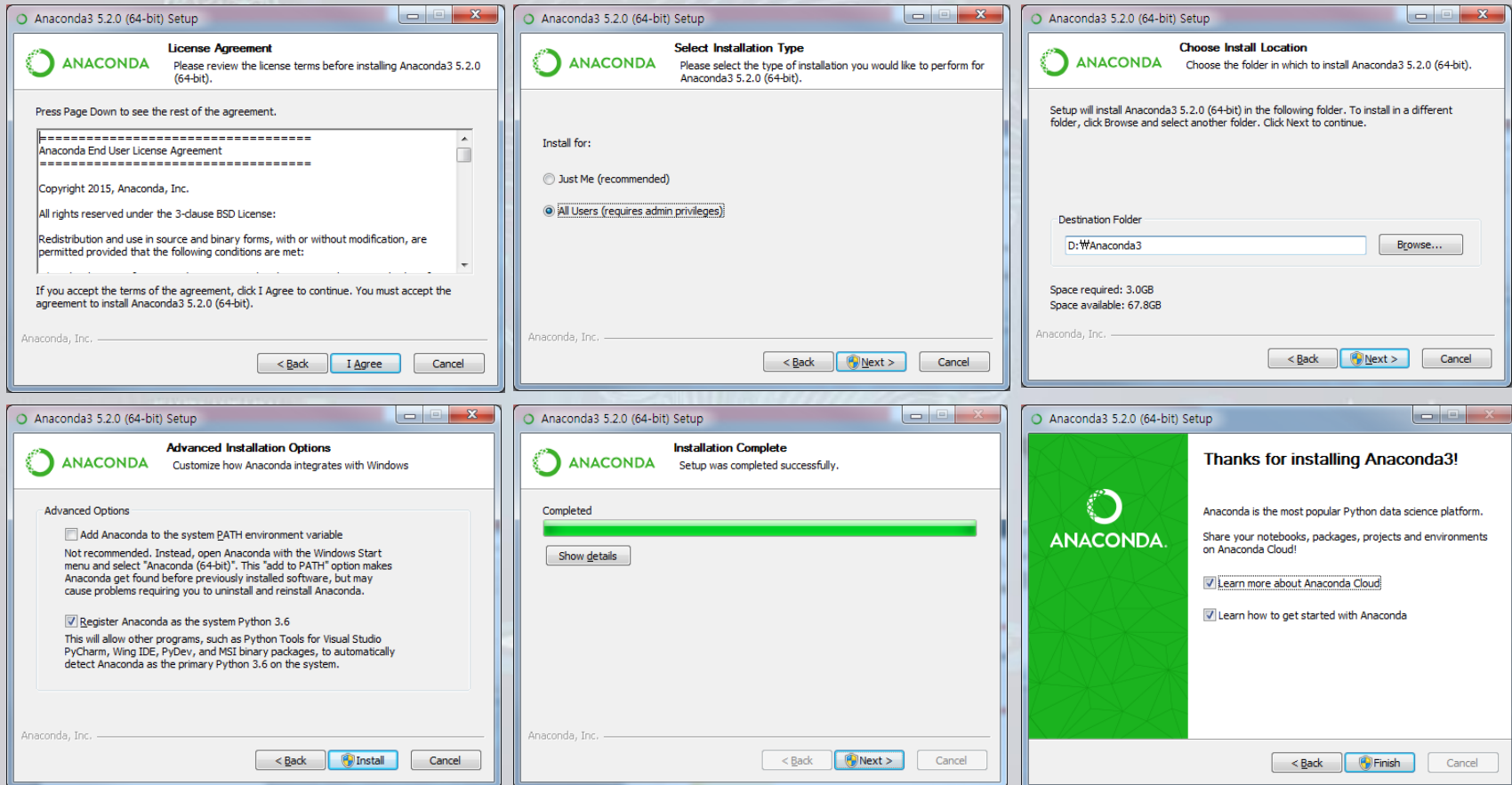
 Download

[64-Bit Graphical Installer \(580 MB\) ?](#)
[32-Bit Graphical Installer \(458 MB\)](#)



Anaconda 설치하기

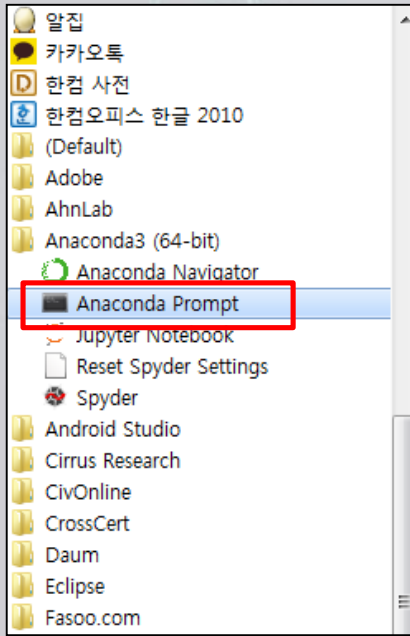
설치





Anaconda 설치하기

- 윈도우 아이콘을 클릭하고 설치한 아나콘다 폴더에서 Anaconda Prompt를 실행
- 파이썬 및 사용할 라이브러리를 설치한다



작업환경이름

파이썬 버전

```
conda create -n lecture01 python=3.5
```

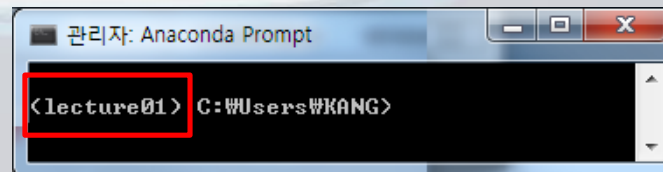
- 작업환경 확인

```
conda env list
```

- 작업환경 활성화 : 프롬프트 맨 앞에 환경이름이 표시

```
activate lecture01
```

```
source activate lecture01
```





Anaconda 설치하기

라이브러리 설치

```
pip install numpy scipy matplotlib pandas seaborn scikit-learn h5py
```

- **numpy** : 과학 계산을 (다차원 배열 계산에 유용)
- **scipy** : 과학 기술 계산을 (**numpy**의 상위 라이브러리)
- **matplotlib** : 시각화 라이브러리
- **spyder** : 아나콘다 배포본에 포함된 파이썬 IDE
- **pandas** : 데이터분석 라이브러리
- **seaborn** : 시각화 라이브러리 (색상, 통계용 차트 등)
- **scikit-learn** : 머신러닝 라이브러리
- **h5py** : HDF5 이진 데이터 포맷 라이브러리

설치된 라이브러리 목록보기

```
conda list
```

```
관리자: Anaconda Prompt

<lecture01> C:\Users\WKANG>conda list
# packages in environment at D:\Anaconda3\envs\lecture01:
#
# Name                   Version             Build           Channel
# -----
alabaster                0.7.12              <pip>
astroid                  2.1.0               <pip>
Babel                    2.6.0               <pip>
backcall                 0.1.0               <pip>
bleach                   3.0.2               <pip>
certifi                  2018.8.24           py35_1
chardet                  3.0.4               <pip>
cloudpickle              0.6.1               <pip>
colorama                 0.4.1               <pip>
cyclor                   0.10.0              <pip>
decorator                4.3.0               <pip>
defusedxml               0.5.0               <pip>
docutils                 0.14                <pip>
entrypoints              0.2.3               <pip>
h5py                     2.8.0               <pip>
idna                     2.7                 <pip>
imagesize                1.1.0               <pip>
ipykernel                5.1.0               <pip>
ipython                  7.2.0               <pip>
ipython-genutils         0.2.0               <pip>
isort                    4.3.4               <pip>
jedi                     0.13.1              <pip>
```




Anaconda 설치하기

Tensorflow 설치

- 데이터 흐름 그래프를 사용하는 수치 연산용 오픈소스 소프트웨어 라이브러리 (cafe, pytorch 등)
- 구글에서 머신러닝 및 심층신경망 연구용을 개발되었다가 공개
- **Tensor** : 다차원 데이터 배열

```
pip install tensorflow
```

설치 테스트

```
python
```

```
>>import tensorflow as tf  
>>print(tf.__version__)
```

```
관리자: Anaconda Prompt - python  
<lecture01> C:\Users\WKANG>python  
Python 3.5.6 |Anaconda, Inc.| (default, Aug 26 2018, 16:05:27) [MSC v.1900 64 bi  
t (AMD64)] on win32  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import tensorflow as tf  
>>> print(tf.__version__)  
1.12.0  
>>>
```



Anaconda 설치하기

• Keras 설치

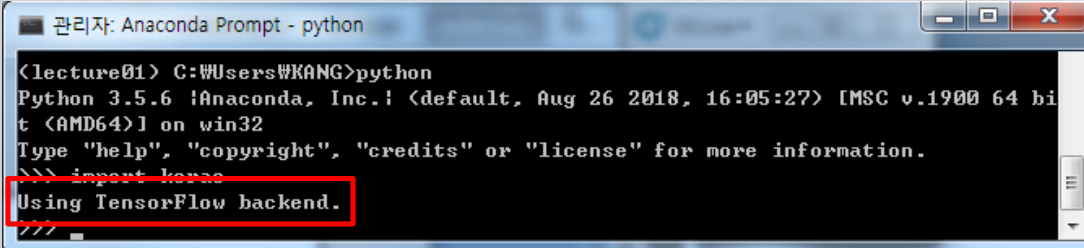
- 파이썬으로 작성된 오픈소스 신경망 라이브러리 (딥러닝 지원)

```
pip install keras
```

• 설치 테스트

```
python
```

```
>>import keras
```



```
관리자: Anaconda Prompt - python

<lecture01> C:\Users\WKANG>python
Python 3.5.6 |Anaconda, Inc.| <default, Aug 26 2018, 16:05:27> [MSC v.1900 64 bi
t (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import keras
Using TensorFlow backend.
>>>
```

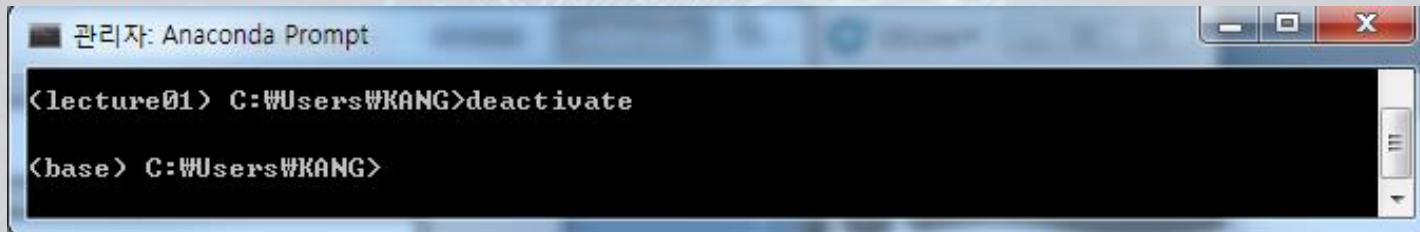



Anaconda 설치하기

- 작업환경 비활성화

```
deactivate
```

```
source deactivate
```

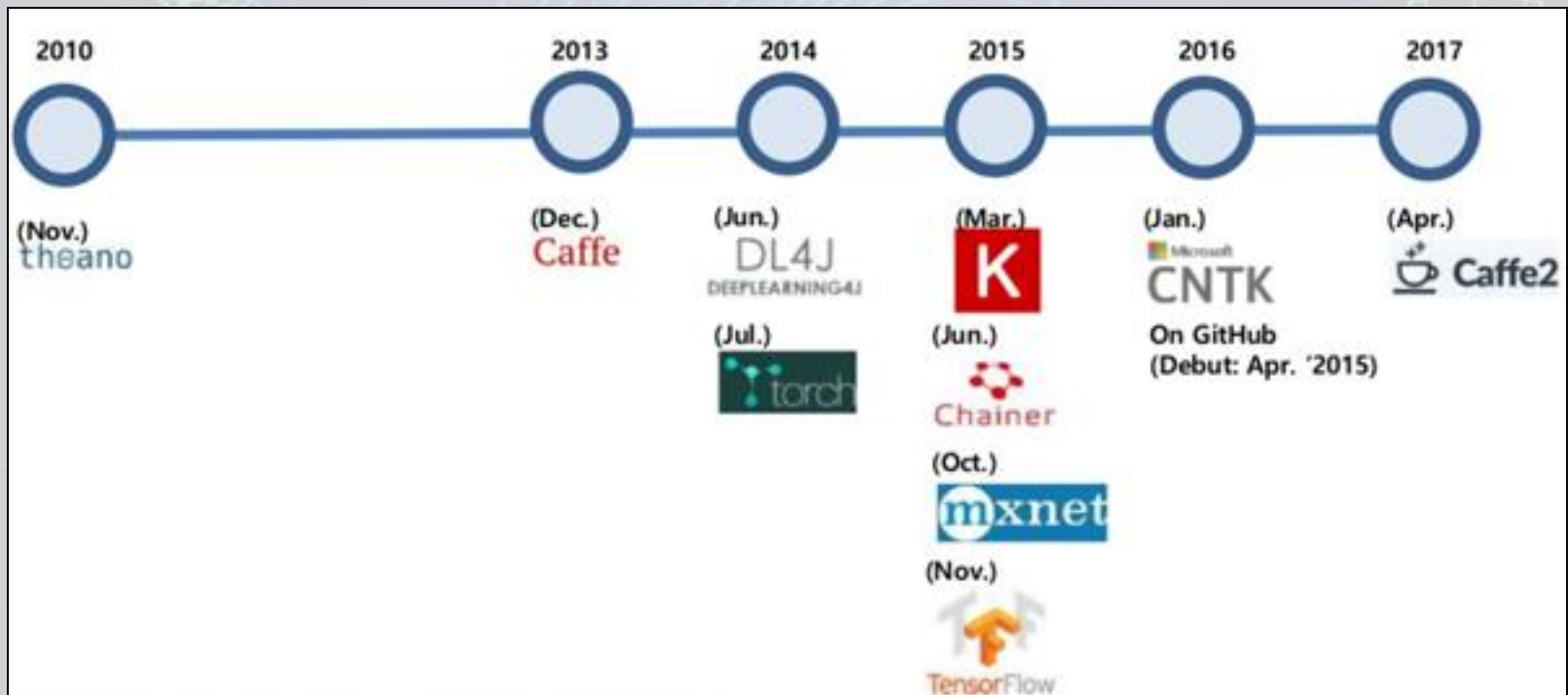


```
관리자: Anaconda Prompt  
<lecture01> C:\Users\WKANG>deactivate  
  
<base> C:\Users\WKANG>
```



딥러닝 프레임워크

역사





Caffe

Caffe



Caffe2

- 버클리대학 인공지능 연구실에서 개발, C/C++ 기반으로 고속 CNN(Convolution Neural Network)을 구현해놓은 것으로 C++로 직접 사용할 수도 있지만 Python과 Matlab도 잘 구현
- Tensorflow와 비교하면 caffe가 성능상 우세 → 모든 코드가 C++ / C 로 작성되어, 파이썬 인터프리터보다 우수
- <http://caffe.berkeleyvision.org/>
- 특징
 - 이미지 처리와 머신러닝에 특화, 프로그래밍 대신에 설정 파일로 학습 방법을 정의
 - Caffe Model Zoo를 통한 다양한 Pre-trained Model 제공
 - 텍스트, 사운드 등의 데이터 처리에는 부적합
 - 윈도우에서 개발환경을 갖추기 어렵고, 코딩 하는 것 자체가 생각보다 어려움
 - 새로운 기능 추가가 어렵고 문서화가 잘 안되어 있음
- 최근 Facebook과 NVidia를 비롯한 많은 회사와 협력하여 Caffe2가 출시 (<https://caffe2.ai/>)
- 적용사례 : Facebook, Adobe, Microsoft, Samsung, Flickr, Tesla, Yelp, Pinterest 등



Chainer



- 일본에서 제작된 딥러닝 프레임워크로 다이내믹 컴퓨테이션 그래픽을 지원
- <http://docs.chainer.org/en/latest/index.html>
- 특징
 - 거의 모든 딥러닝 알고리즘을 직관적인 Python 코드로 구현할 수 있음
 - 자유도가 매우 높음
 - 대다수의 다른 라이브러리들과는 다르게 "Define-by-Run" 형태로 구현되어 있어서, forward 수만 정의해주면 네트워크 구조가 자동으로 정해진다는 점이 특이 (유연성 제공)
 - 협소한 사용자 커뮤니티
- 적용사례 : Toyota motors, Panasonic, FANUC



CNTK



- Microsoft의 딥러닝 오픈 소스 프레임워크
- DNN, CNN, 회귀 등의 알고리즘이 포함되어 있으며 C++ 기반에 Python API를 제공
- <https://www.microsoft.com/en-us/cognitive-toolkit/>
- <https://github.com/Microsoft/CNTK>
- 특징
 - 처리 성능의 선형성
 - 협소한 사용자 커뮤니티
- 적용사례 : Microsoft's speech recognition engine, Skype's Translator



DL4J



- Java와 스칼라를 위해 작성된 상용화 수준의 딥러닝 프레임워크
- <https://deeplearning4j.org/>
- 특징
 - 쉬운 이식성과 엔터프라이즈 시스템 수준의 안정성 제공
 - Spark 기반의 분산 처리 지원
 - 문서화가 잘 되어 있음
 - 학습 디버깅을 위한 시각화 도구 DL4J UI 제공
 - Java 언어로 인한 학습 및 테스트 과정이 번거러움
 - 협소한 사용자 커뮤니티와 예제
- 적용사례 : 은행 Fraud Detection



Keras



K Keras

- Theano 기반 Python으로 작성된 딥러닝 프레임워크로 Torch 처럼 모듈화가 잘되어 있어서 사용하기 쉽고 업데이트와 빠르게 발전하는 라이브러리
- Keras에서는 다양한 딥러닝 모델을 지원해주고 있으므로 전반적인 네트워크 구조를 생각하고 작성한다면 빠른 시간 내에 코딩을 할 수 있는 엄청난 장점
- 현재는 Tensorflow 위에서 Keras가 동작하도록 설계되어 있고, 완전히 포함되지는 않았지만 Keras를 Tensorflow 안에 아예 집어넣으려고 하는 노력을 진행 중
- <https://keras.io/>
- 특징
 - 직관적인 API 인터페이스
 - Caffe, Torch, TensorFlow, CNTK, Theano 등 다양한 프레임워크 모델 import 기능 제공
 - 문서화가 잘되어 있음
 - Theano 프레임워크에서 문제 발생시 디버깅이 어려움
- 적용사례 : Tensorflow



MXNet



- 빠르고 확장 가능한 교육 및 추론 프레임워크로서 기계 학습을 위해 사용이 쉽고 간단한 API가 제공
- <http://mxnet.incubator.apache.org/>
- 특징
 - 다양한 프로그래밍 인터페이스 지원
 - 모바일 지원
 - 빠르게 발전
 - 저레벨 / 고레벨 API 모두 제공
 - Imperative / Graph 프로그래밍 모델 모두 지원
 - 다소 속도가 느림
- 적용사례 : AWS



TensorFlow



- 구글에서 제작한 데이터 흐름 그래프를 사용하는 수치 연산용 오픈소스 소프트웨어 라이브러리
- 구글 사내에서만 사용되고 있다가 문서 작성, 디버깅, 버전 관리가 잘 안되는 부분이 있어서 오픈 소스로 공개
- 딥러닝 프레임워크 커뮤니티 중에서도 가장 활성화가 되어 있는 프레임워크
- 전반적인 사용은 python 을 통해 이루어지지만, 많은 부분이 뒤의 C++ 로 작성되어 있어 속도 면에서도 그렇게 뒤쳐지지 않게끔 설계되었고, 아마도 가장 오래 인기 있을 딥러닝 프레임워크
- <https://www.tensorflow.org/>
- 특징
 - 추상화된 그래프 모델, 학습 디버깅을 위한 시각화 도구 TensorBoard 제공
 - 모바일 지원, 저레벨/고레벨 API 모두 제공, 방대한 사용자 커뮤니티
 - Define-and-Run 모델 / 런타임 그래프 변경 안됨, Torch에 비해 느림
- 적용사례 : Google (Search Signals, Email auto-responder, Photo Search)



Theano



Theano

- 최초의 딥러닝 프레임워크로 몬테리얼 대학교에서 사용, 파이썬 라이브러리
- 초창기 버전은 속도에 치중한 나머지 프레임워크를 일반 사용자가 사용하기에는 어려움 (2017년에 공식 지원 마감)
- 딥러닝 알고리즘을 파이썬으로 쉽게 구현할 수 있도록 해주는데, Theano 기반 위에 얹어서 더 사용하기 쉽게 구현된 여러 라이브러리 제공 (Keras, Pylearn2, Lasagne, Blocks 등) -
- <http://deeplearning.net/software/theano/index.html>
- 특징
 - 저레벨을 제어할 수 있는 API, 추상화된 그래프 모델 지원, 빠르고 유연함
 - Keras, Lasagne, Blocks 등 Wrapper 프레임워크 기반 프레임워크
 - 저레벨 API가 복잡함
- 적용사례 : Keras, Lasagne, Blocks



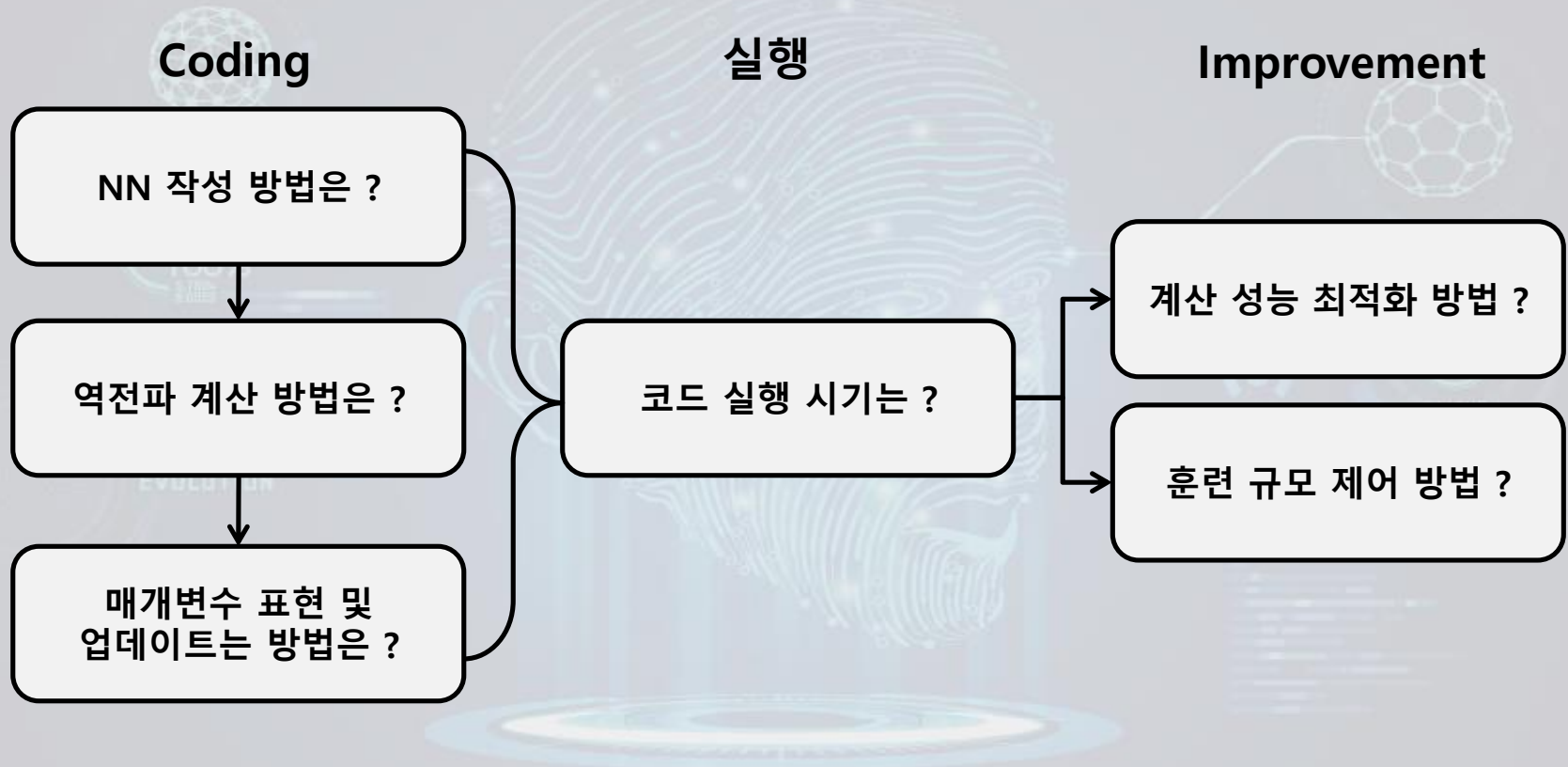
Torch



- 초창기에는 Lua 라는 언어로 개발되었고, GPU를 사용할 수 있으며, 사용자가 사용하기에 편한 것에 치중
- 페이스북과 구글 딥마인드 등 많은 기업들 및 개발자들에서 사랑받고 있고, 최근 PyTorch 파이썬 라이브러리를 제공
- Keras와 비교하자면, Keras는 코드를 막 작성해서 굴리기 편한 반면에, Torch 는 좀 더 개발자스럽게 customize 할 수 있는 부분이 많고, 객체지향적으로 모델을 만드는 형식임.
- <http://torch.ch/>
- 특징
 - 필요한 모든 기능이 잘 구현되어 사용이 용이, 다양한 데이터 전처리 및 시각화 유틸리티 제공
 - 스크립트 언어인 Lua를 사용하기 때문에 쉽게 사용 가능, OpenCL 및 모바일 지원
 - 파이썬 인터페이스가 없음 (PyTorch 별도 존재)
 - 문서화가 잘 안되어 있음, 협소한 사용자 커뮤니티, 연구용으로 적합
- 적용사례 : Facebook, Google, Twitter, Element Inc. 등



딥러닝 프레임워크의 선택





딥러닝 프레임워크의 선택

• NN을 어떻게 작성할 것인가 ?

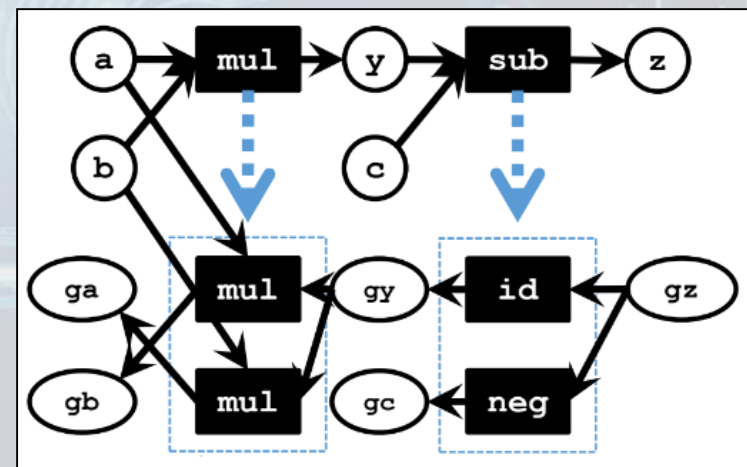
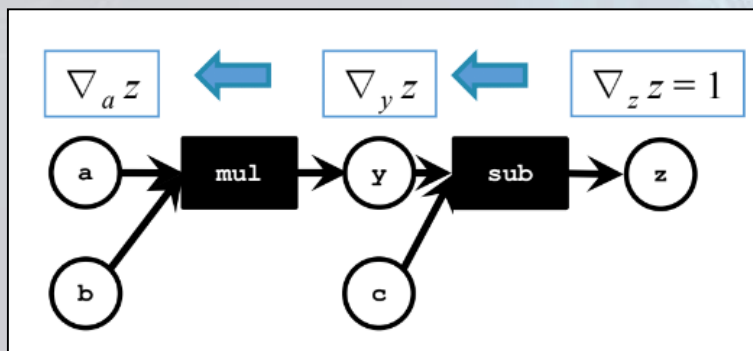
방법	특징	설명	프레임워크
선언적 구성 파일에 NN 작성	휴대성 높음 유연성 낮음	- 구문 분석이 쉽고 다른 프레임워크에서 재사용 가능 - 구조화하기 어려워 복잡한 NN 작성 힘들	caffe Pylearn2 (YAML)
절차적 스크립팅으로 NN 작성	휴대성 낮음 유연성 높음	- 다른 프레임워크로 이식이 어려움 - 스크립티 언어의 추상화 기능 사용 가능	others



딥러닝 프레임워크의 선택

• 역전파 계산 방법은 ?

방법	특징	설명	프레임워크
그래프 기반 역전파	낮은 유연성 구현이 쉽고 단순	- 순전파의 그래프만 작성하고 그래프를 역추적하여 역전파 수행	Torch Caffe Chainer PyTorch
확장 그래프 기반 역전파	높은 유연성 구현 복잡	- 순전파뿐만 아니라 역전파를 위한 그래프 프도 작성	Theano MXNet Tensorflow

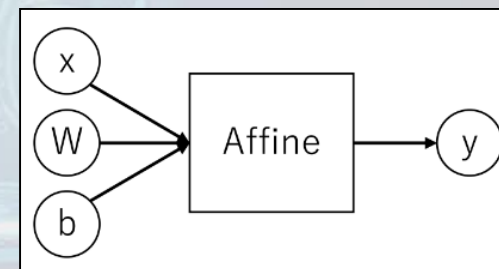




딥러닝 프레임워크의 선택

매개변수 표현 방법 ?

방법	특징	설명	프레임워크
연산자 노드의 일부로 표현	직관적 낮은 유연성 낮은 재사용성	<ul style="list-style-type: none"> - 매개 변수가 연산자 노드 (컨볼루션 계층)에 포함되며 그래프에 나타나지 않음 - 변수 노드처럼 매개변수를 사용할 수 없음 	Torch Caffe MXNet
개별 노드로 표현	높은 유연성 높은 재사용성	<ul style="list-style-type: none"> - 매개 변수를 별도의 변수 노드로 표시 - 변수 노드에서 수행할 수 있는 모든 작업을 매개 변수에서 적용 가능 	Theano Chainer Tensorflow PyTorch

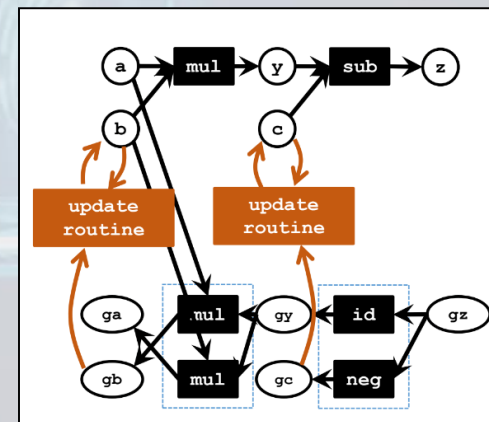
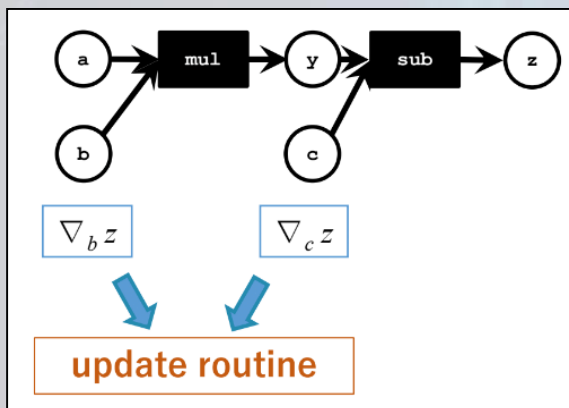




딥러닝 프레임워크의 선택

매개변수 업데이트 방법 ?

방법	특징	설명	프레임워크
그래프 외부의 자체 루틴으로 업데이트	구현 용이 낮은 무결성	- 업데이트 수식은 백엔드 배열 라이브러리를 사용하여 자체 루틴을 직접 구현 - 배열 기능 활용 가능, 그래프 미 통합	Torch Caffe MXNet Chainer PyTorch
그래프의 일부로 업데이트	구현 복잡 높은 무결성	- 업데이터 수식은 계산 그래프의 일부로 작성 - 계산 그래프 내에 변경 가능한 연산 지원 필요	Theano Tensorflow

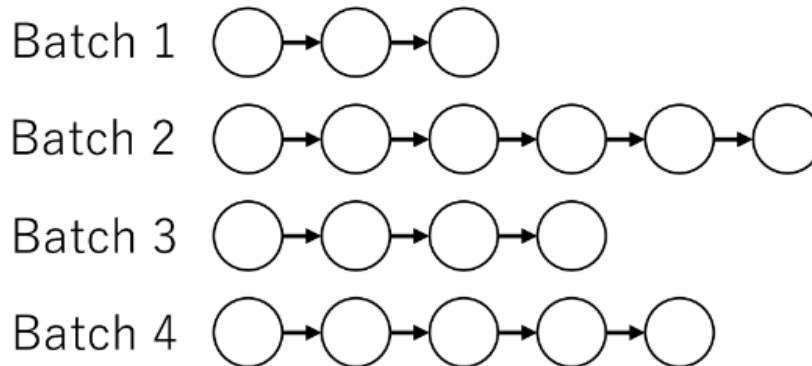




딥러닝 프레임워크의 선택

• 코드 실행 시기는 ?

방법	특징	설명	프레임워크
정적 계산 그래프	최적화 용이 낮은 유연성 낮은 유용성	-루프에 들어가기 전에 그래프 작성 (Define-and-Run) -프레임워크 생성 시 최적화 가능, 반복마다 동일한 그래프를 사용해야 함	others
동적 계산 그래프	최적화 어려움 높은 유연성 높은 유용성	-매 루프마다 그래프 작성 (Define-by-Run) -반복 시마다 최적화 수행 어려움 -반복 시마다 다른 그래프 작성 가능	autograd Chainer PyTorch





딥러닝 프레임워크의 선택

• 계산 성능의 최적화 방법 ?

방법	설명	프레임워크
그래프 변환	-계산 그래프 최적화 기능 제공	Theano Tensorflow
맞춤형 운영자 노드 작성	-자신의 목적에 맞게 최적화된 자체 운영자 노드 작성	Torch MXNet Chainer Pytorch



딥러닝 프레임워크의 선택

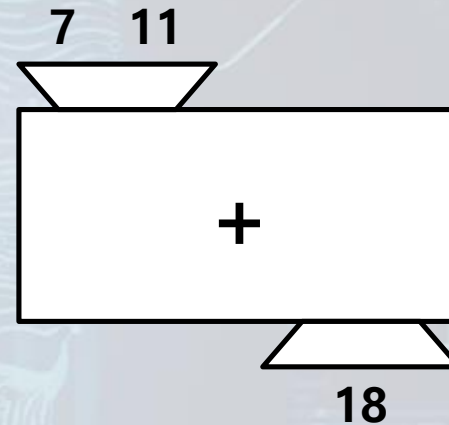
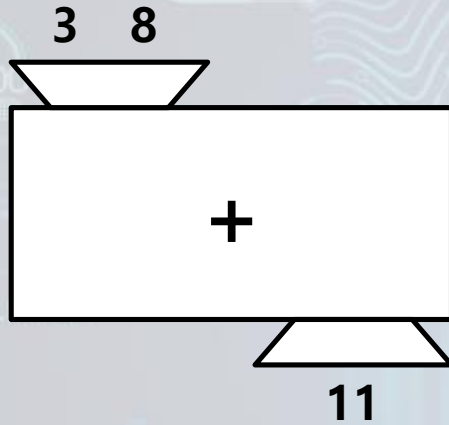
• 훈련 규모 제어 방법 ?

방법	설명	프레임워크
다중 GPU 병렬 처리	-다중 GPU 계산 지원	All
다중 노드 병렬 처리 (분산 처리)	-학습 규모를 확장하기 위해 분산 계산을 지원	MXNet Tensorflow CNTK Chainer DL4J PyTorch

딥러닝 (Deep Learning) 개요



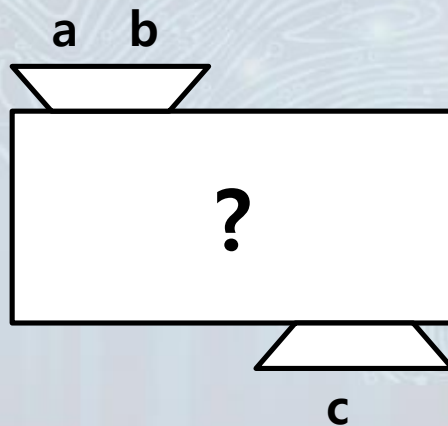
머신러닝





머신러닝

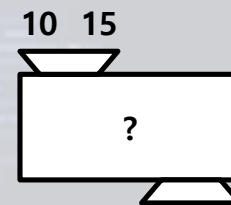
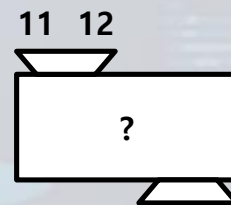
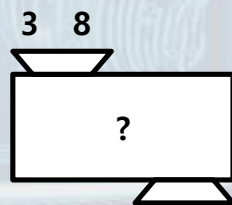
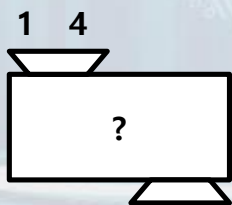
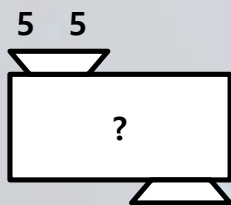
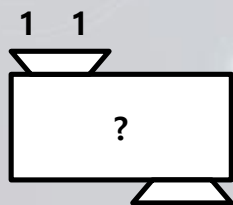
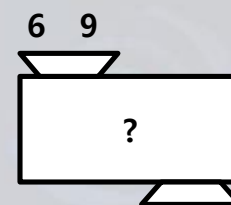
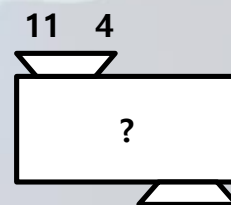
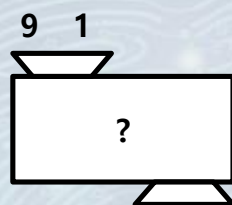
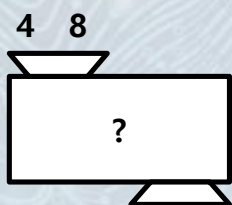
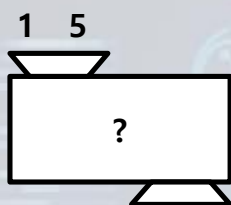
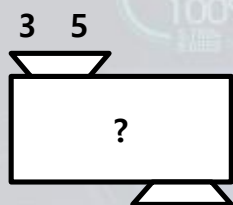
안에서 무슨 일이 일어나서 c 가 나오는 걸까요 ?





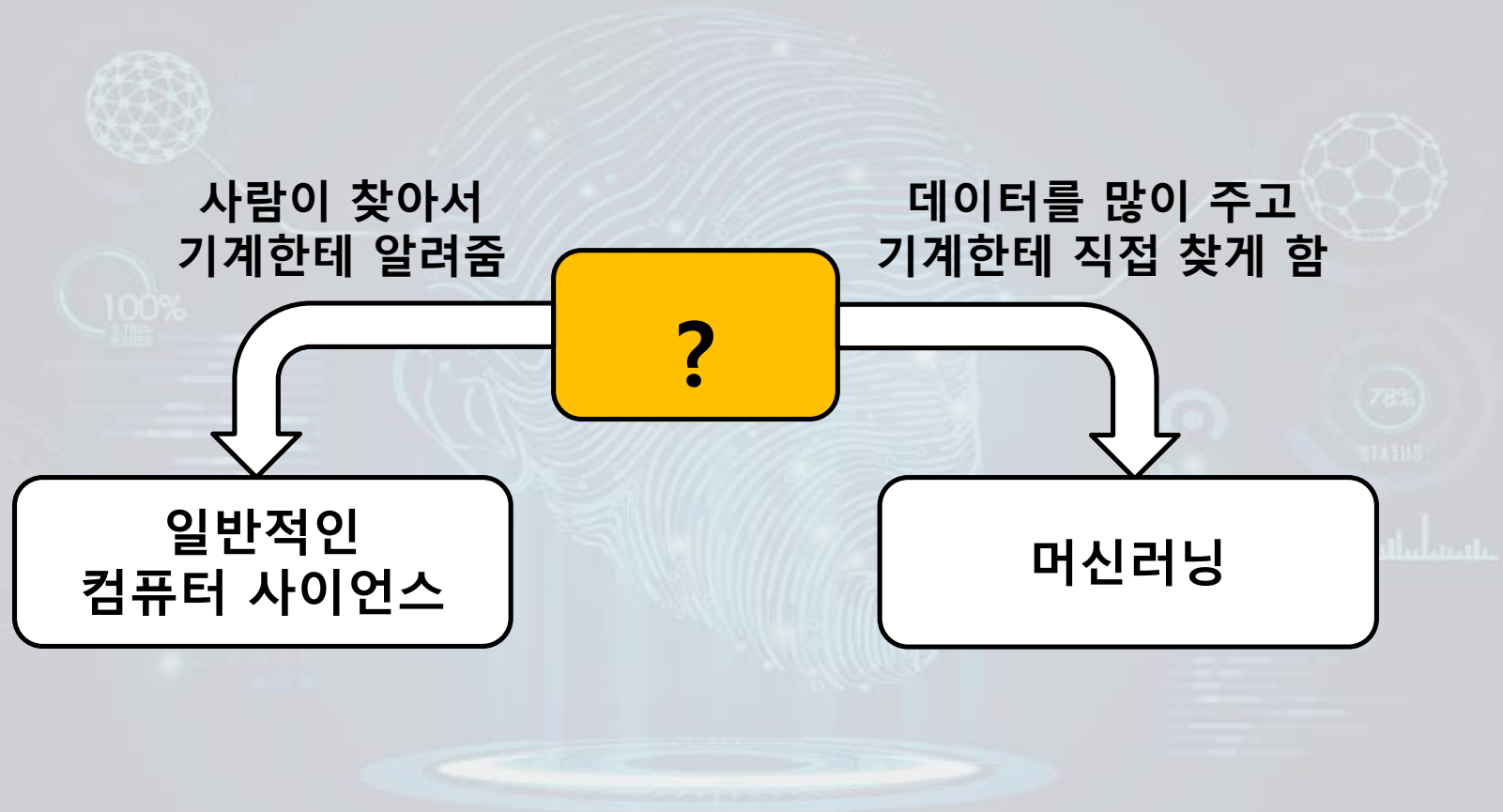
머신러닝

?는 무엇일까요 ?





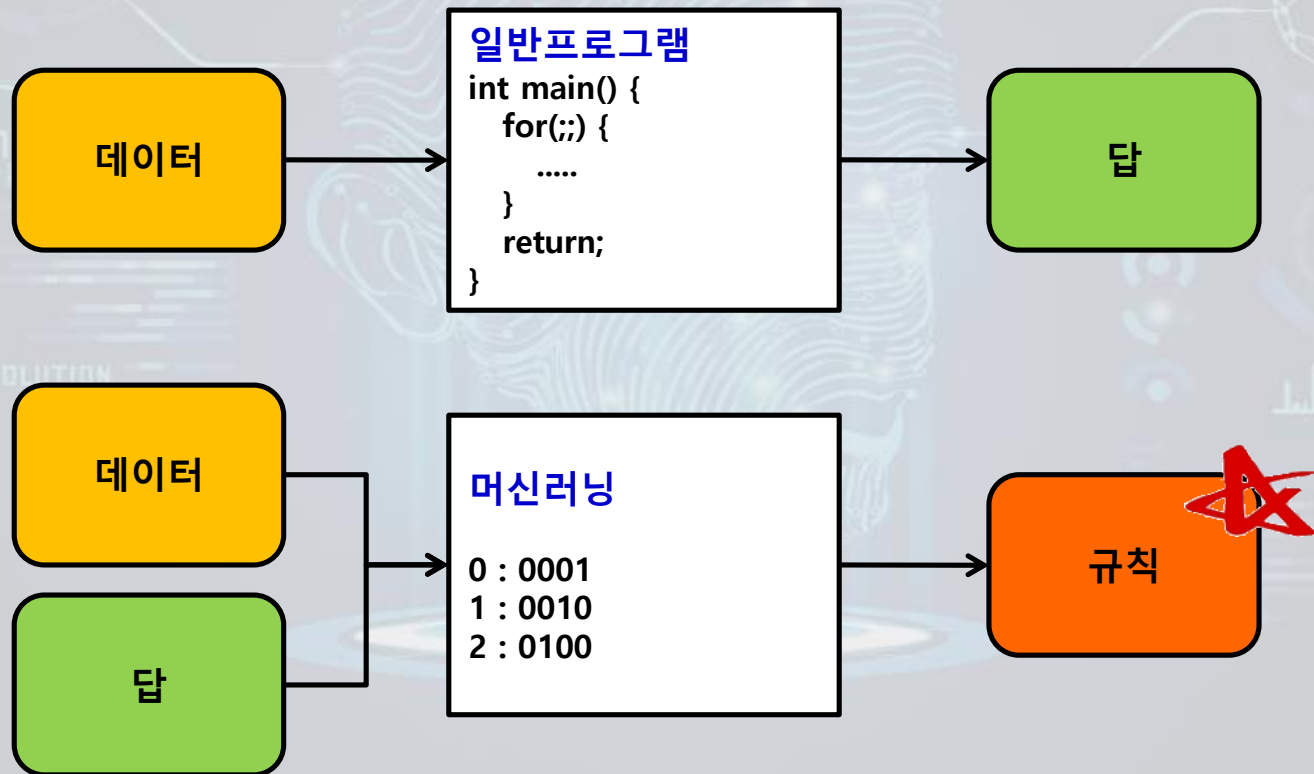
머신러닝





머신러닝

- 머신러닝은 기존 데이터를 이용하여 아직 일어나지 않는 미지의 일을 예측하기 위해 만들어진 기법





기존 환자의 데이터를 이용해 새로운 환자의 생사를 예측하는 프로그램을 만든다면



환자 데이터 입력

진료기록 1 : 사망

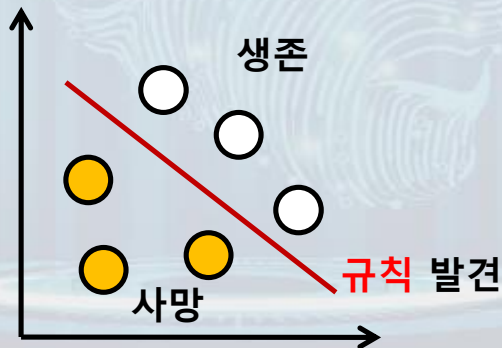
진료기록 2 : 생존

진료기록 3 : 사망

...

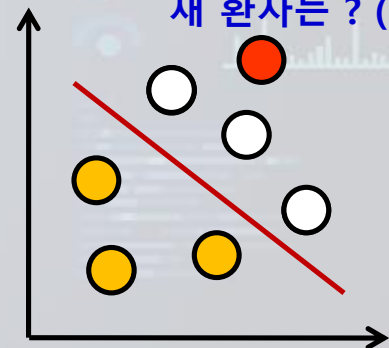


머신러닝으로 학습



새로운 환자 예측

새 환자는 ? (예측)



딥러닝이란 ?

여러분은 사람들을 어떻게 기억하고 누군지 인식하나요 ?





딥러닝이란 ?

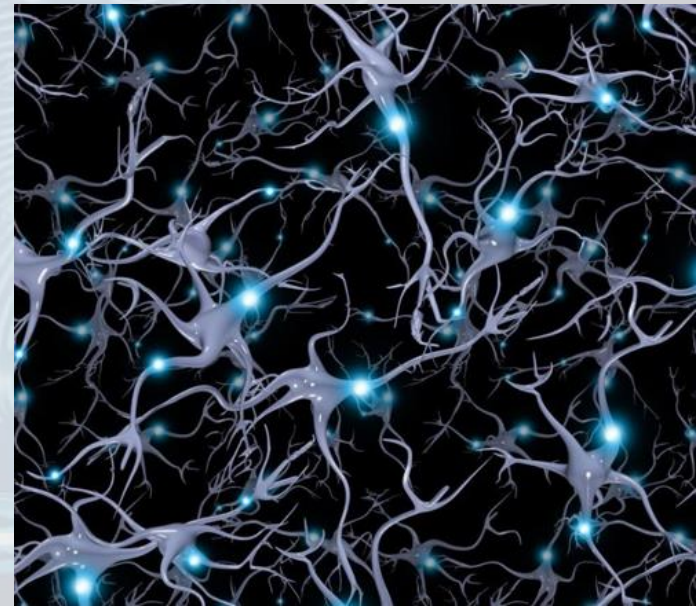
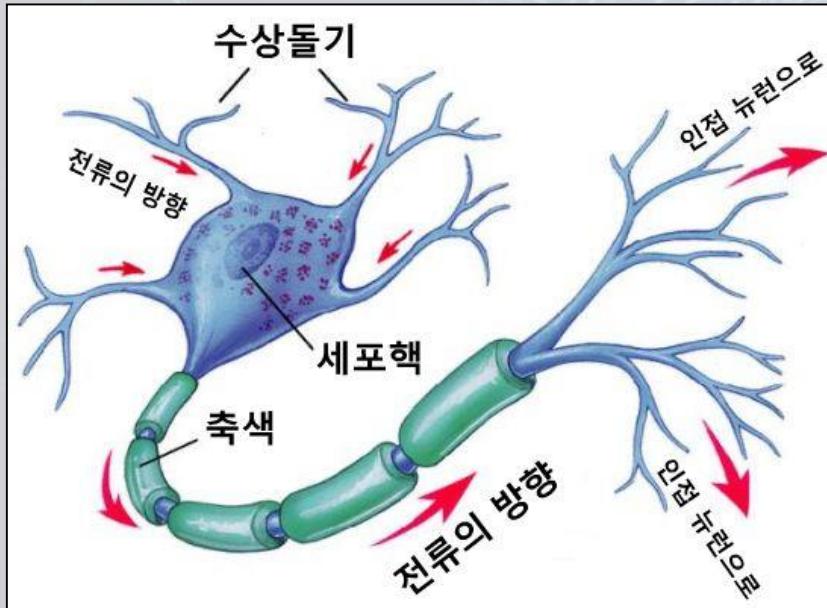
비율 확률은 어떻게 계산하는 걸까요 ?

날짜	내일(30일 토)								모레(01일 일)						
시각	24	03	06	09	12	15	18	21	24	03	06	09	12	15	
날씨															
강수확률(%)	81	90	90	89	89	90	90	72	30	20	10	0	0		
강수량	25~49mm				10~24mm				-						
최저/최고(℃)	14/16								8/14						
기온(℃)	16	15	14	15	15	15	14	13	12	11	9	11	13	14	



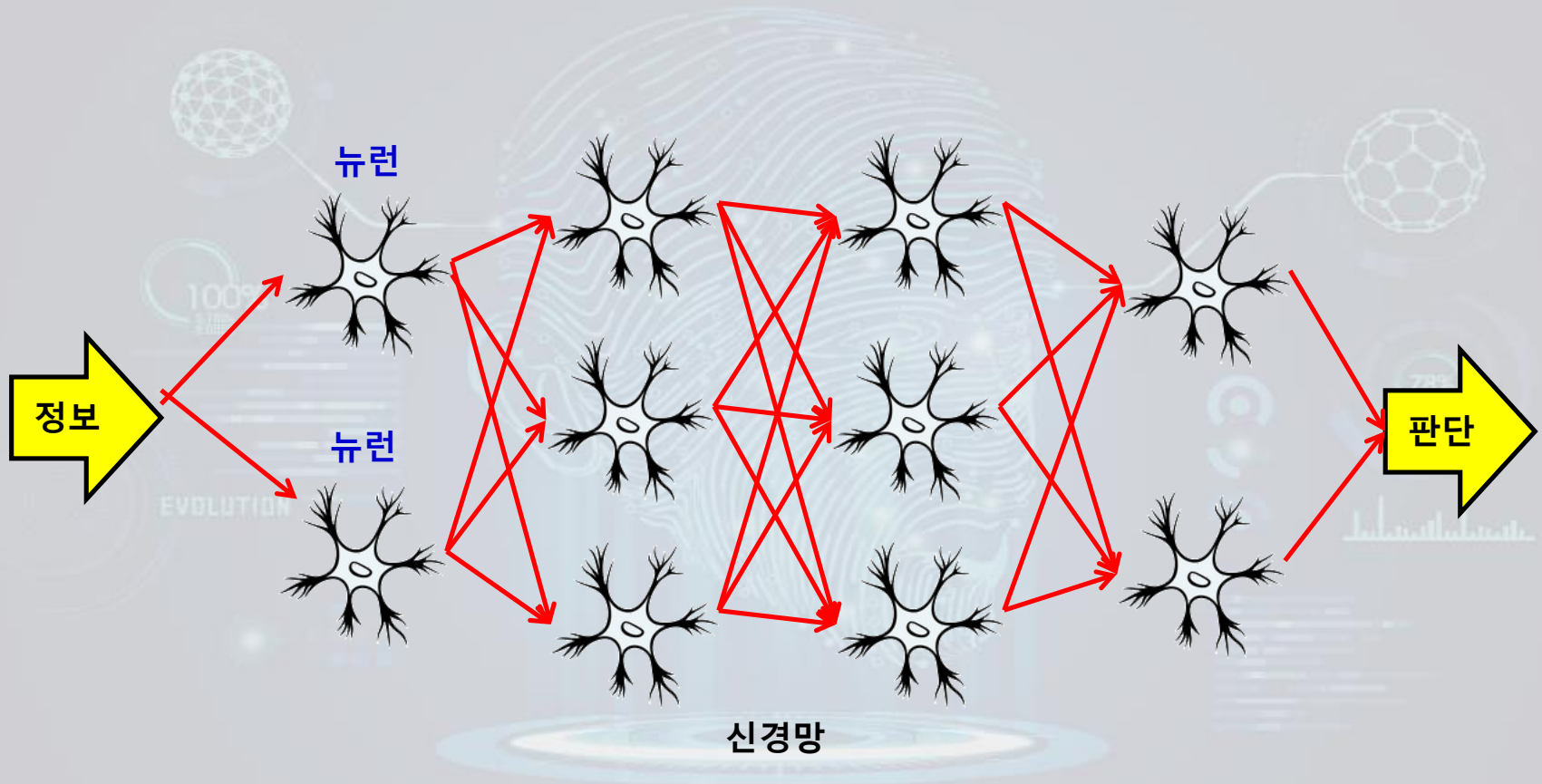
딥러닝이란 ?

여러분이 생각한 방법들이 사람은 어떻게 가능할까요 ?



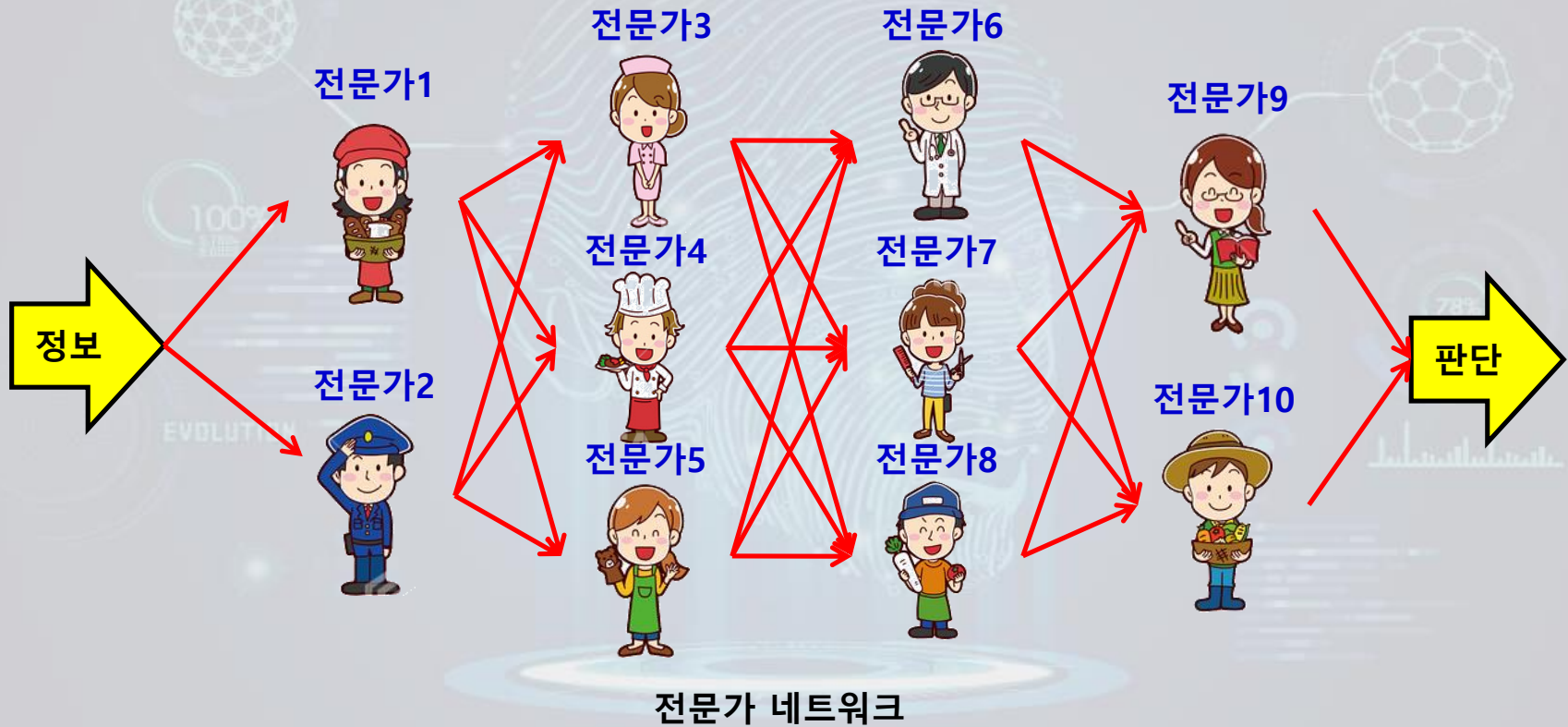


딥러닝이란 ?





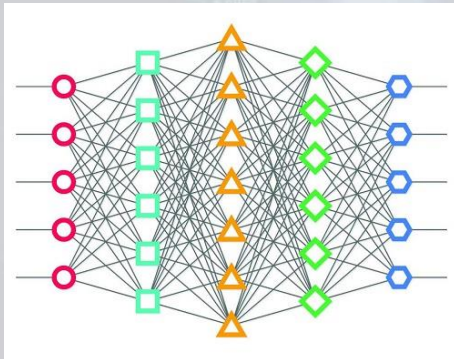
답러닝이란 ?



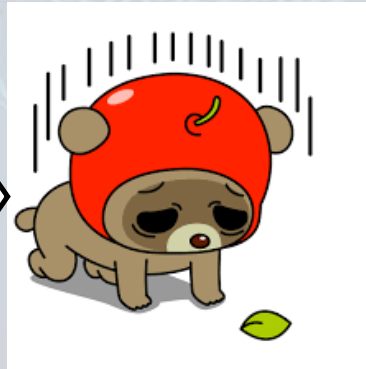


딥러닝이란 ?

1950년 전부터 있었던 신경망 개념이 최근에 갑자기 뜬 것일까요 ?



많은 데이터가 필요



딥러닝이란 ?

- 예측의 신뢰성을 높이기 위해 다양한 머신러닝 알고리즘이 등장하였으며 딥러닝은 이러한 수많은 머신러닝 알고리즘 중의 하나





실습 : 폐암 수술환자의 생존율 예측하기

• Data set의 형태 (ThoracicSurgery.csv)

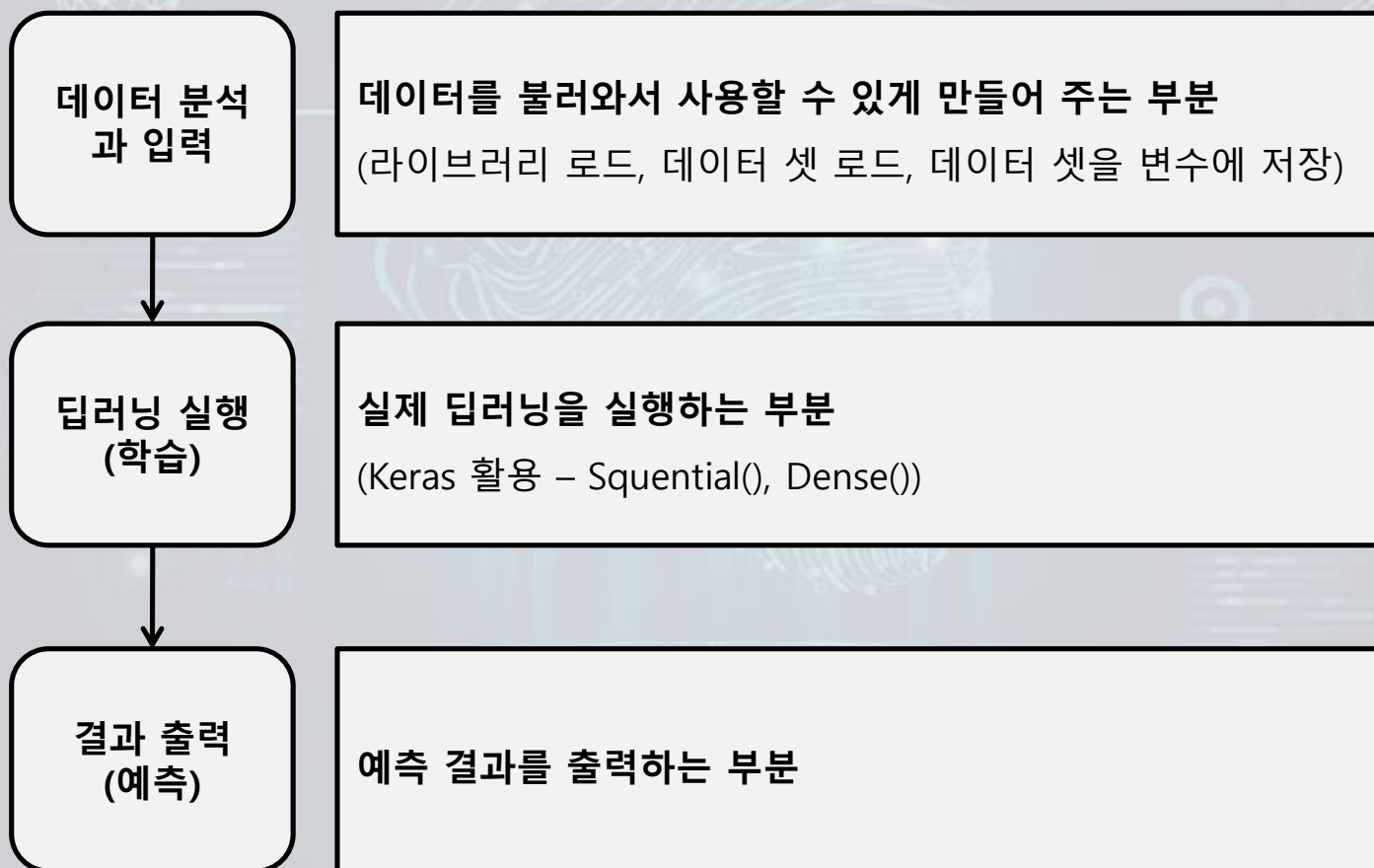
- 폴란드 브로츠와프 의과대학에서 2013년 공개한 폐암 수술 환자의 수술 전 데이터와 수술 후 생존 결과를 기록한 의료 기록 데이터
- 18개 항목으로 구성된 470개의 데이터로 구성되고 각 항목은 ,로 구분
- 종양 유형, 폐활량, 호흡곤란여부, 고통 정도, 기침, 흡연, 천식여부 등 17가지 환자 상태
- 18번째 항목은 수술 후 생존 결과 (1 : 생존, 0 : 사망)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	293	1	3.8	0	0	0	0	0	0	0	12	0	0	0	1	0	62	0
2	1	2	2.88	2.16	1	0	0	0	1	1	14	0	0	0	1	0	60	0
3	8	2	3.19	2.5	1	0	0	0	1	0	11	0	0	1	1	0	66	1
...
470	447	8	5.2	4.1	0	0	0	0	0	0	12	0	0	0	0	0	49	0



실습 : 폐암 수술환자의 생존율 예측하기

• 코드 구성 (DL01_딥러닝기초.ipynb)





실습 : 폐암 수술환자의 생존율 예측하기

- 속성 데이터와 레벨 데이터로 분리

100%

속성 (X)

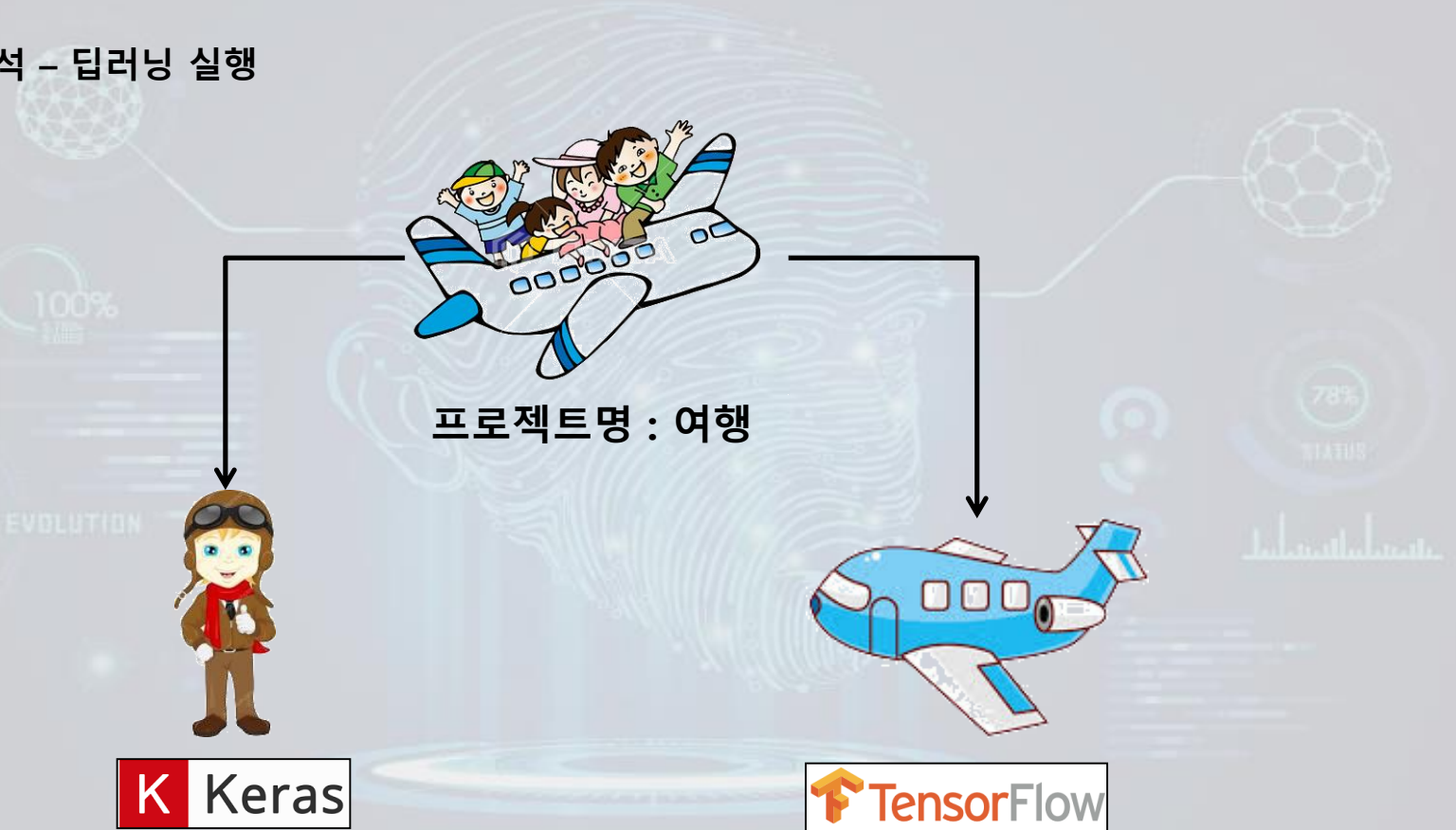
정답 / 클래스 (Y)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	293	1	3.8	0	0	0	0	0	0	0	12	0	0	0	1	0	62	0



실습 : 폐암 수술환자의 생존율 예측하기

코드 분석 - 딥러닝 실행



비행기의 이륙 및 정확한 지점까지
도착하게 해 주는 역할

목적지까지 빠르게 이동시켜주는 역할



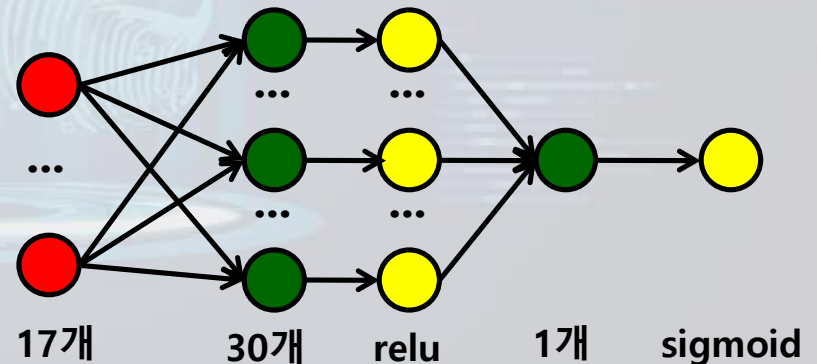
실습 : 폐암 수술환자의 생존율 예측하기

코드 분석 - 딥러닝 실행

```

16 model = Sequential()
17 model.add(Dense(30, input_dim=17, activation='relu'))
18 model.add(Dense(1, activation='sigmoid'))
  
```

- **Sequential()** : 신경망을 한층 한층 쌓는 기능 (add() 함수 사용)
- **Dense()** : 신경망 각층의 특성을 설정하는 기능
- [17] Dense(출력 뉴런 수, 입력뉴런 수, 활성화 함수 종류)
- [18] Dense(출력 뉴런 수, 활성화 함수 종류)
- **활성화 함수** : 출력으로 나오게 하는 기준값 (sigmoid, tanh, relu, softmax 등)





실습 : 폐암 수술환자의 생존율 예측하기

코드 분석 - 딥러닝 실행

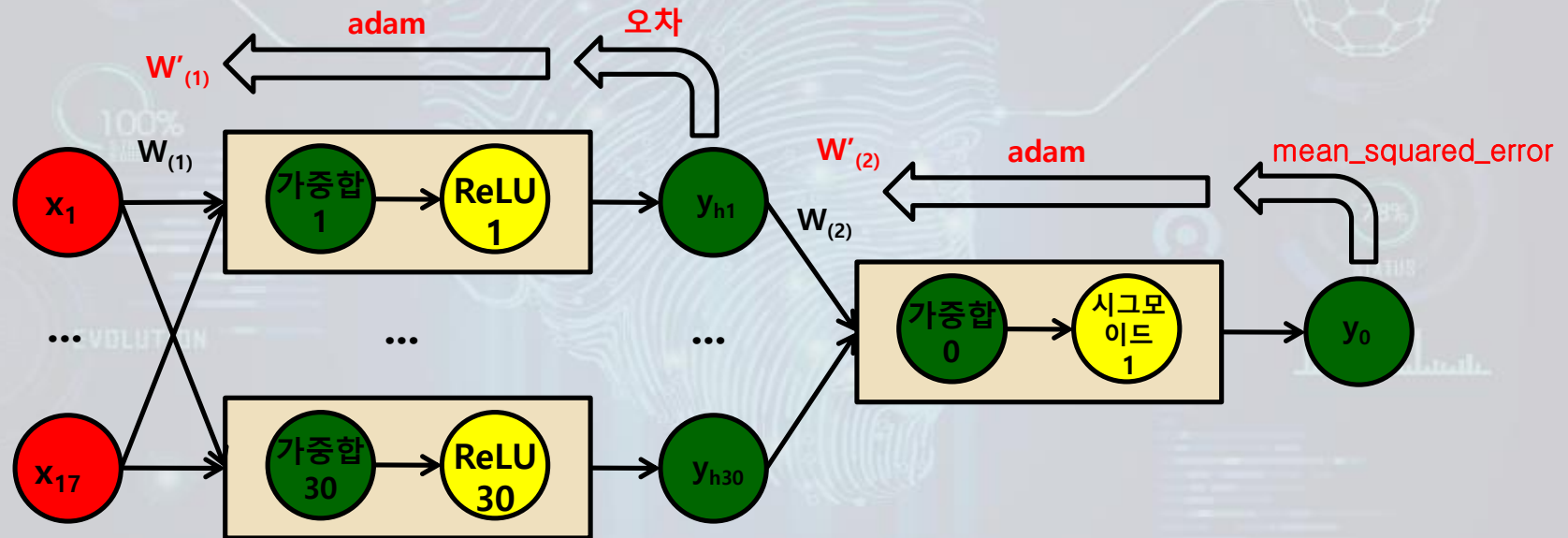
20	model.compile(loss='mean_squared_error', optimizer='adam', metrics=['accuracy'])
21	model.fit(X, Y, epochs=30, batch_size=10)

- **compile()** : 학습 프로세스를 설정하는 기능
- **fit()** : 학습 데이터로 학습하는 기능
- [20] compile(손실함수 종류, 최적화(오차감소) 방법, 평가 방법)
- [21] fit(입력 데이터, 입력 라벨, 학습 반복 횟수(30개 층에 대해 한번씩 계산), 한번에 처리할 데이터 수)
- **손실함수** : 오차 계산 방법 (MSE (최소자승오차), CEE (교차 엔트로피 오차) 등)
- **최적화 방법** : 정답을 찾아가는 방법 (SGD (확률적 경사하강법), 모멘텀, AdaGrad, Adam 등)



실습 : 폐암 수술환자의 생존율 예측하기

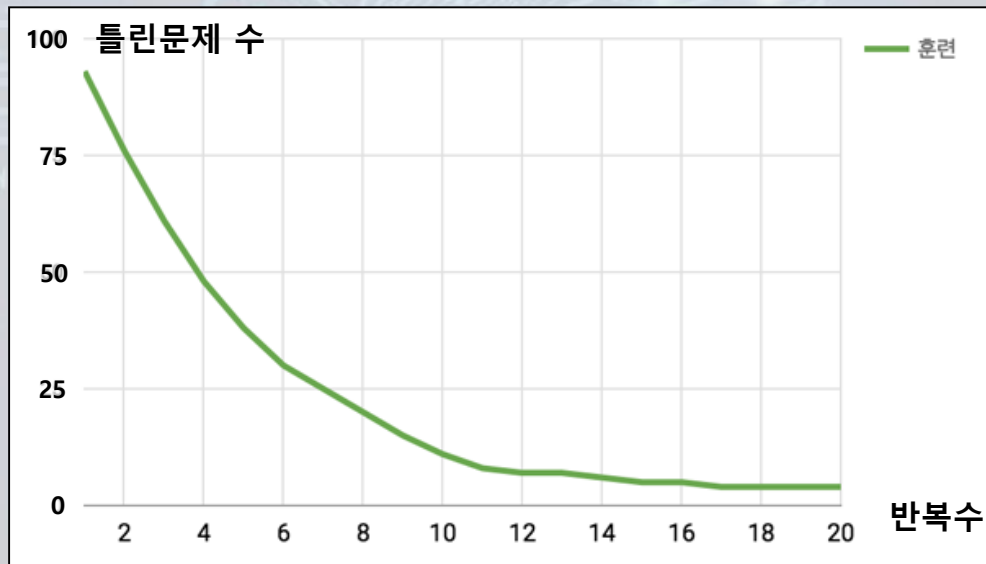
코드 분석 - 딥러닝 실행





실습 : 폐암 수술환자의 생존율 예측하기

- **batch** : 한번에 처리할 데이터의 수 (몇 문제를 풀고 해답을 확인하는 지를 의미)
- **epoch** : 학습을 몇 번 수행하는 지의 수 (문제를 몇 번 풀어 볼 것인지 의미)
- 반복 회수 (epoch) 마다 100문제씩 (batch) 풀 경우 틀린 문제의 수





실습 : 폐암 수술환자의 생존율 예측하기

● 결과 출력

```
23 print("\nAccuracy: %.4f" %(model.evaluate(X, Y)[1]))
```

- `evaluate()` : 출력값과 metrics로 설정한 결과값을 반환
- [23] metrics로 설정한 결과 값을 반환



■ 무엇을 해야 할까요 ?

