[기업 연계를 위한 AI-IoT 과정]

인공지능

AI, 세 번째 날

2019.08.14.

첫 번째 날

- 인공지능 개론
- 인공지능 사례
- 머신러닝 개론
- Classification 모델 생성

두 번째 날

- Regression 모델 생성
- 머신러닝 알고리즘
- Scikit-learn 모델 생성

세 번째 날

- 딥러닝 이론
- 이미지 분류
- Keras를 활용한 이미지 분류 모델 생성

인공지능 기술을 이해하고

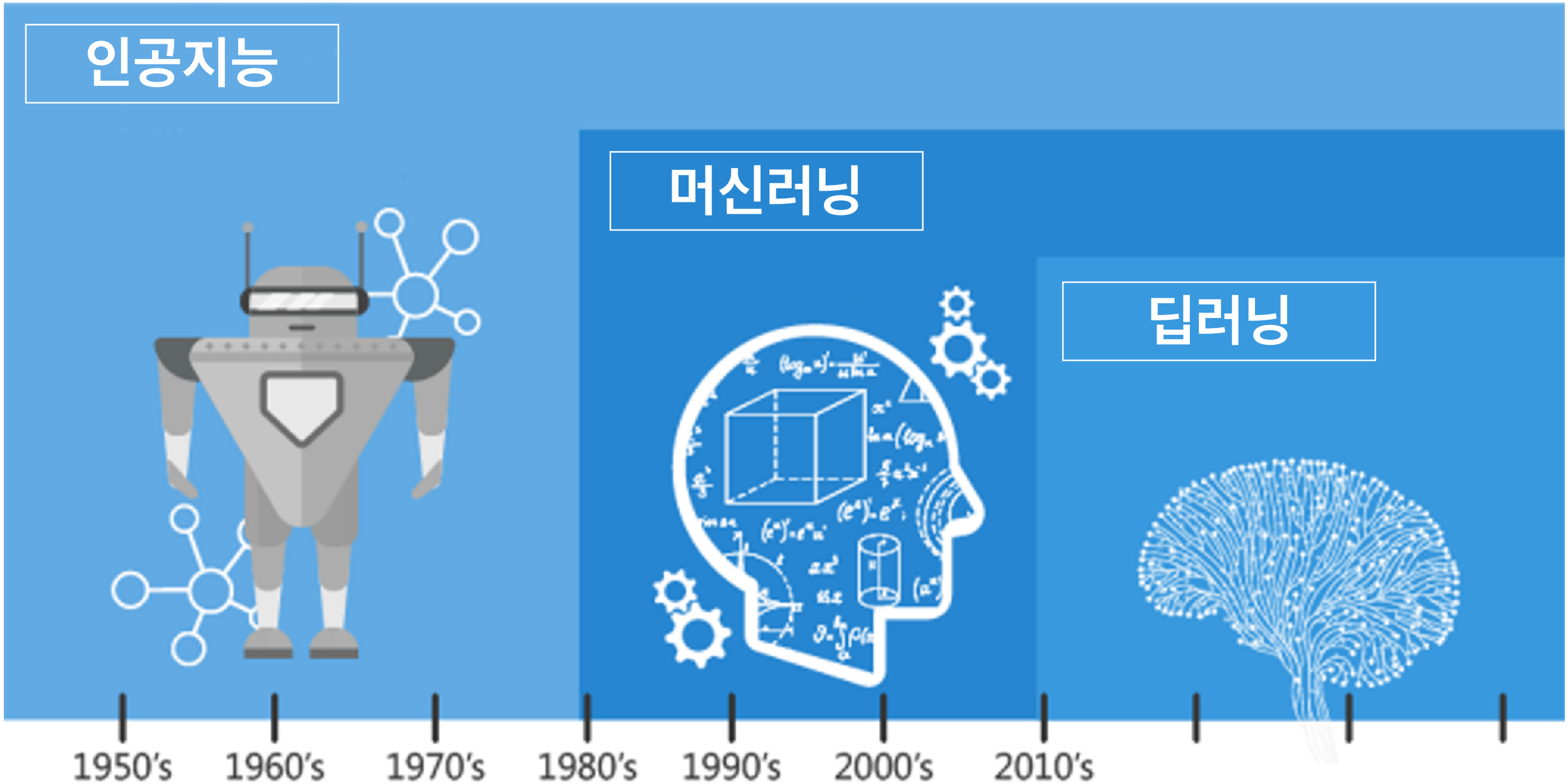
생활과 연결할 수 있는 방법을

구상하고 제작할 수 있다

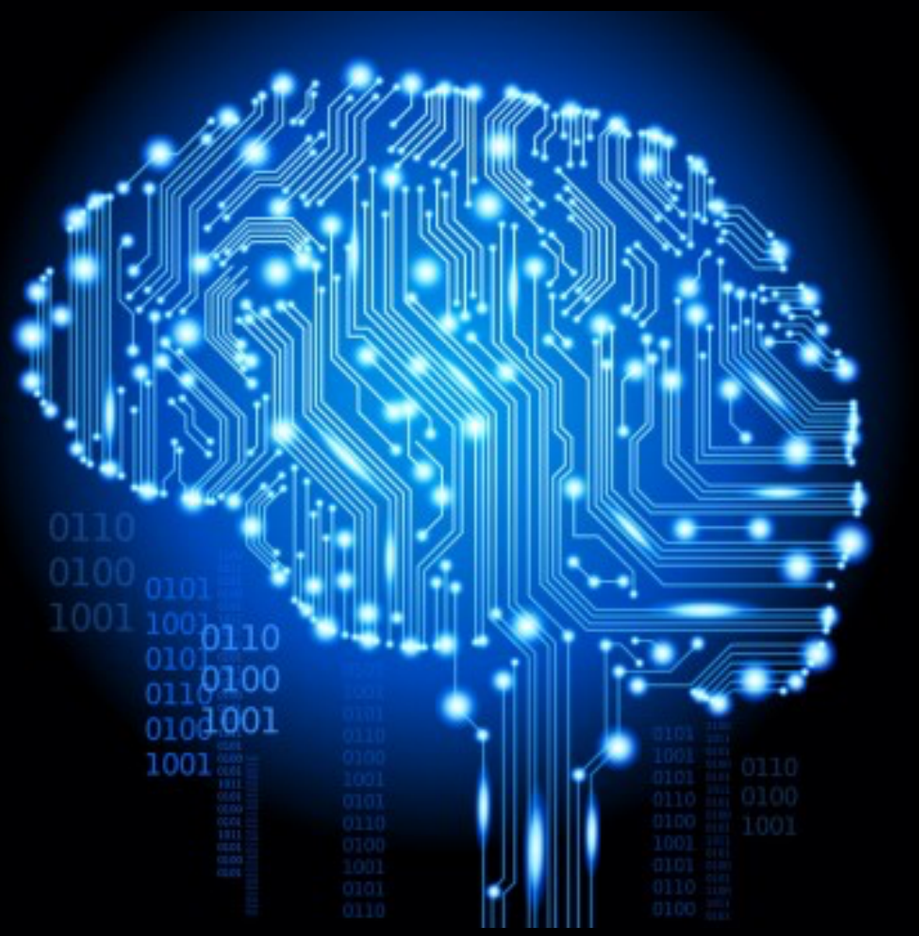
Deep Learning

(Deep) Neural Network

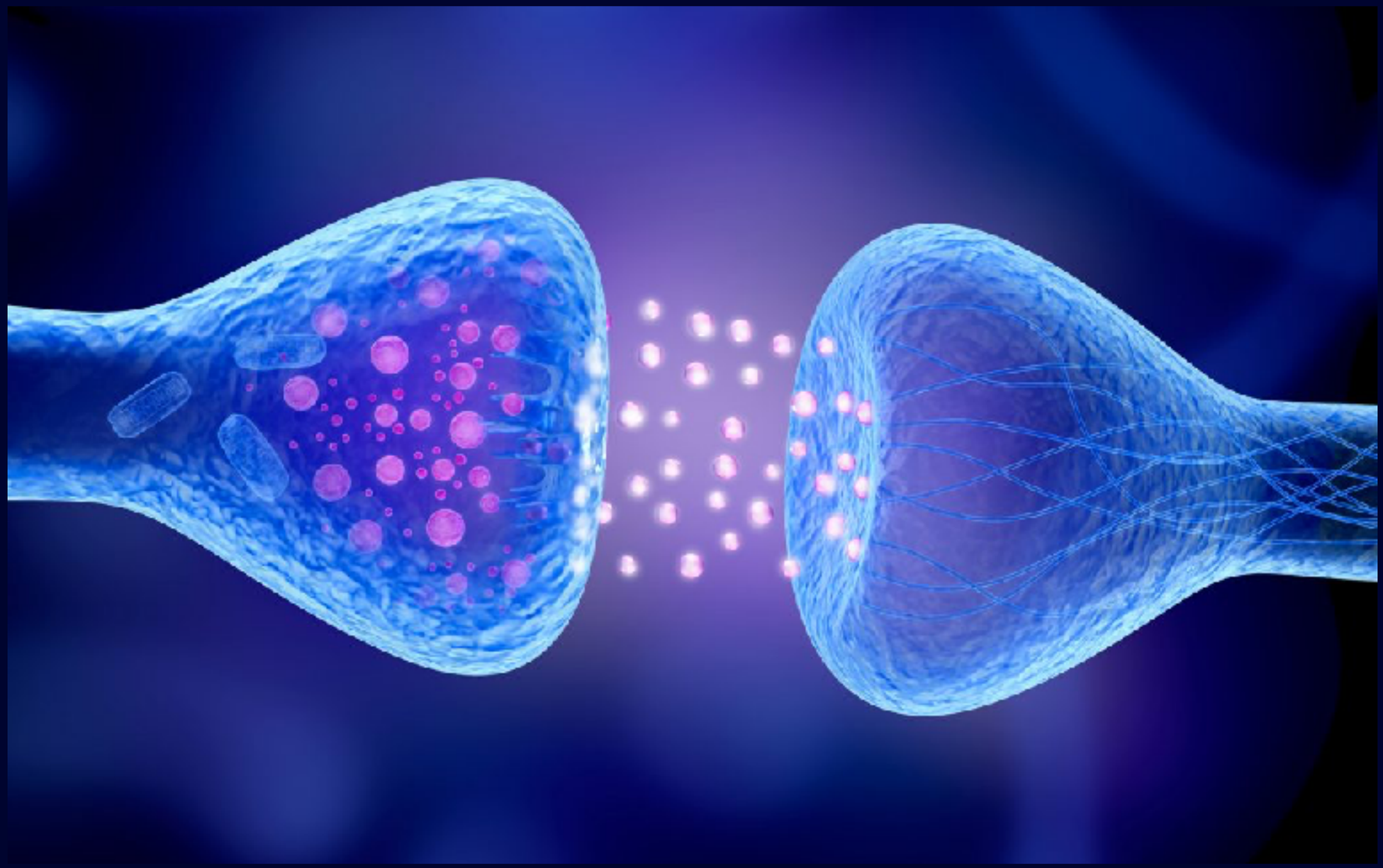
MLP(Multi Layer Perceptron)



http://bisintek.com/science/2017/12/27/knowing-basic-artificial-intelligence/

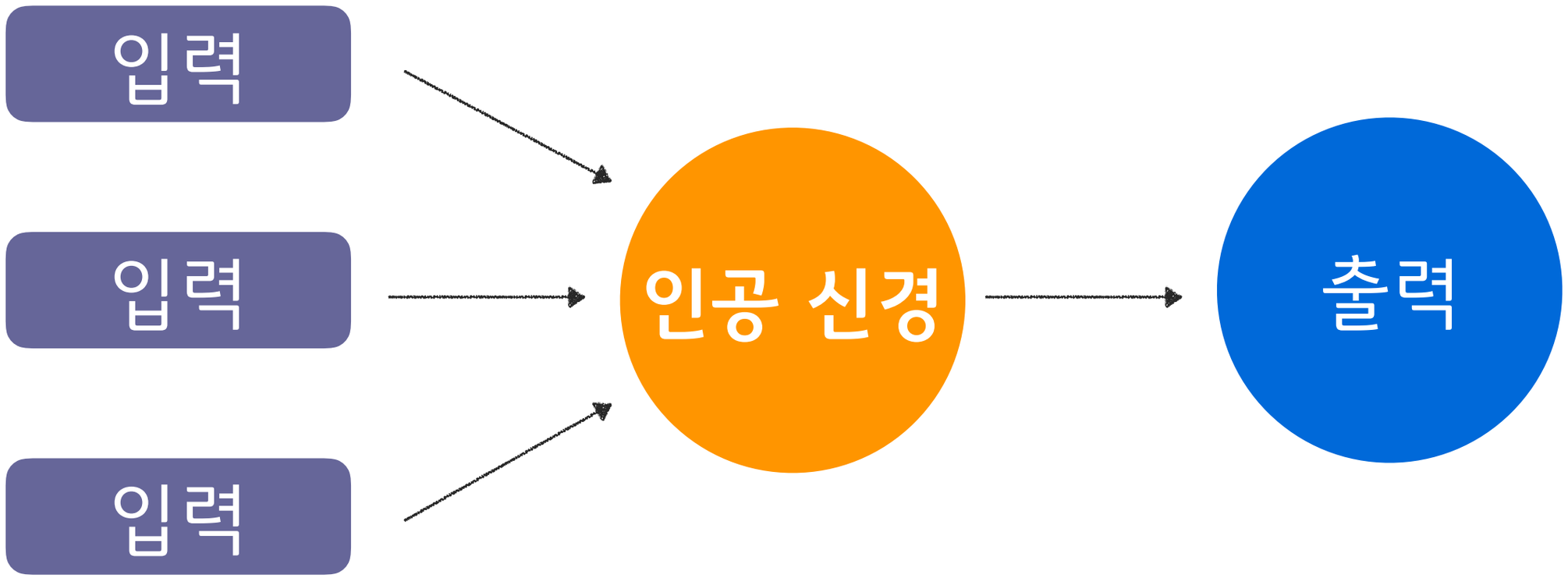


https://www.extremetech.com/wp-content/uploads/2015/12/Brain-Machine-2-640x356.jpg
https://www.alzheimersreadingroom.com/2016/07/alzheimer-tau-protein-spreads-through-brain.html



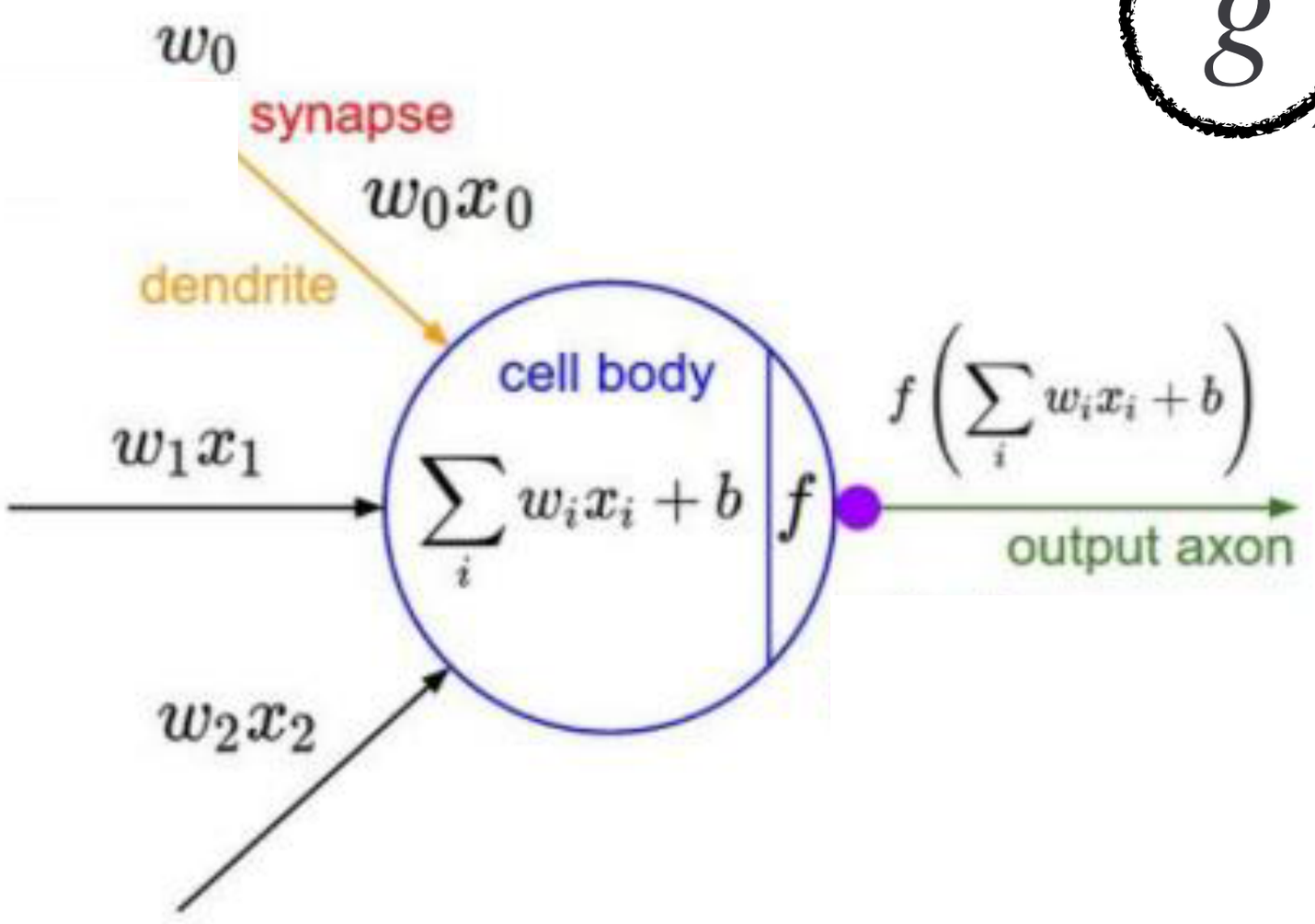
https://lh6.googleusercontent.com/4iAXl2yxBcbWVDXDaHOGeAg-SDjcG0AkmS48PpoocApztM9eXY6p9d-zPiztVlotO_SRl4Y9LZ2-dKCw-D_jl5DA_5pqYD6WPFo8NDMc0hPv8SiXYQ5bjx5ZMLBUghOVJv1zsIeG

인공 신경



$$w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_nx_n$$

$$g \sum (w_ix_i + b)$$



Cost Function

= loss function

= error function

= objective function

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

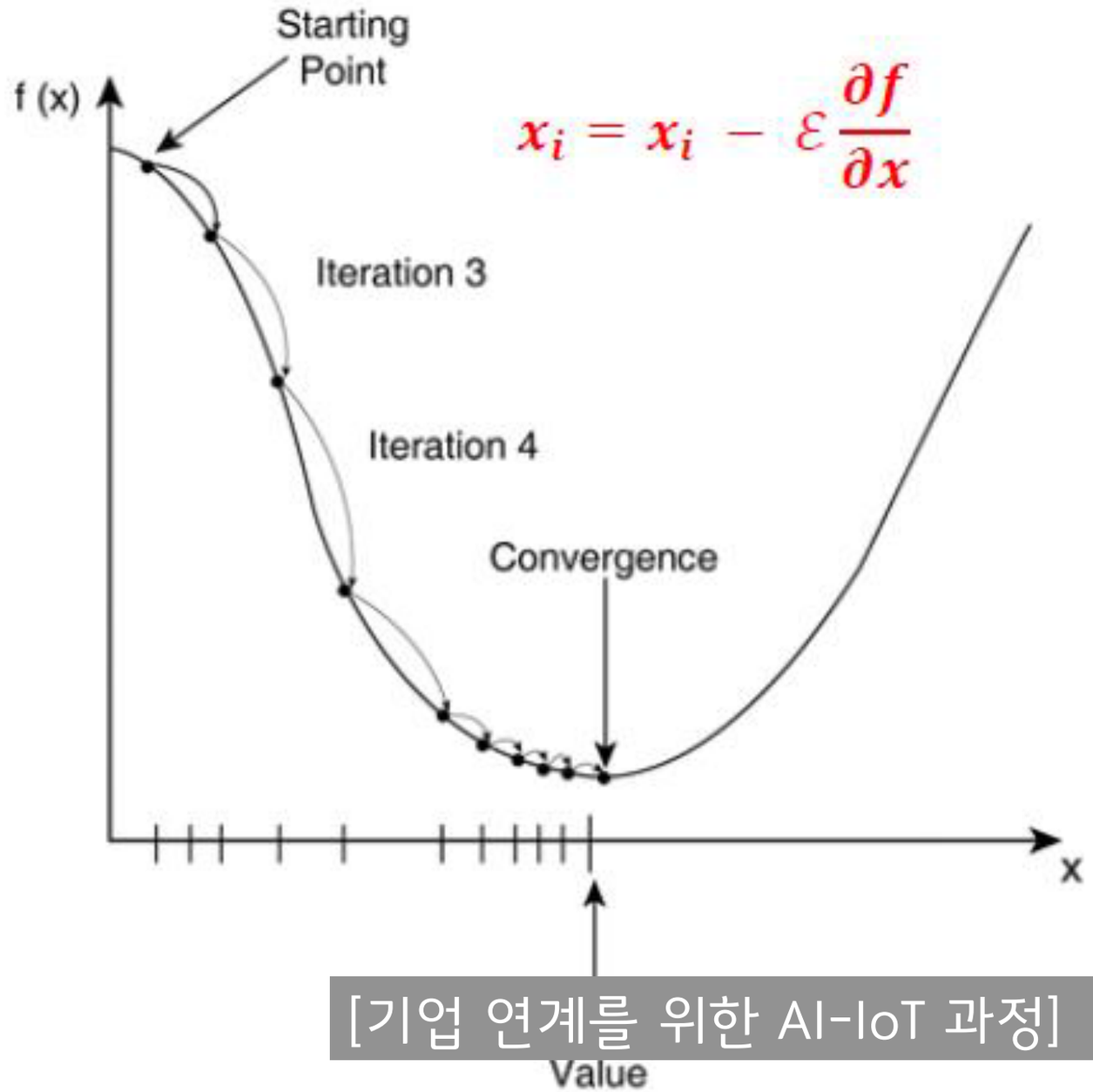
예측 값과 실제 값의 차이를 기반으로

모델의 정확도(성능)을 판단하기 위한 함수

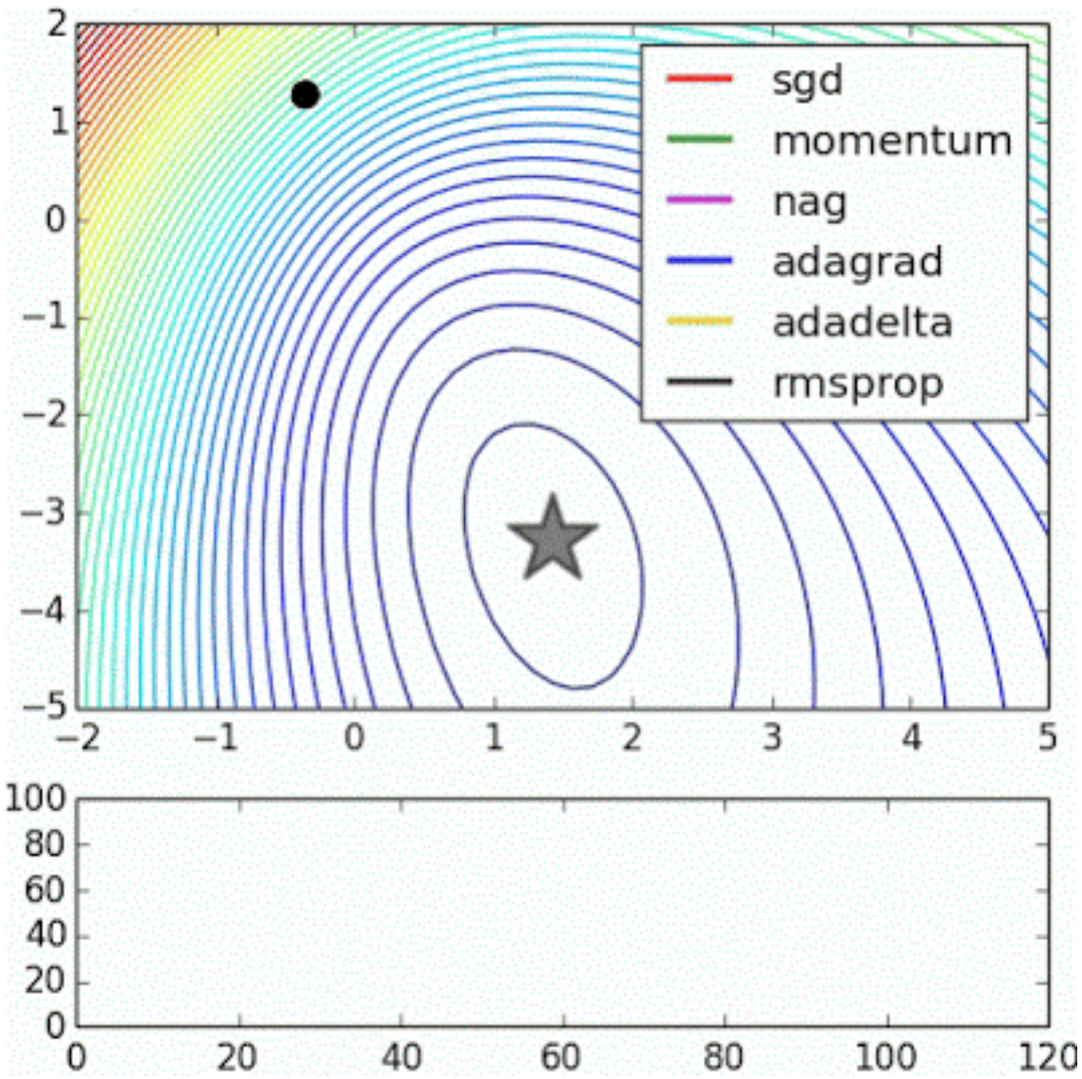
Gradient Descent

cost function 미분 (= 기울기)

기울기 < 0



Optimizer



http://2.bp.blogspot.com/-q6l20Vs4P_w/VPmIC7sEhnI/AAAAAAAACC4/g3UOUX2r_yA/s400/s25RsOr%2B-%2BImgur.gif

Model

모델

```
from keras.models import Sequential

model = Sequential()
```

Model
모델

네트워크
network

목표함수
objective function

최적화기
optimizer

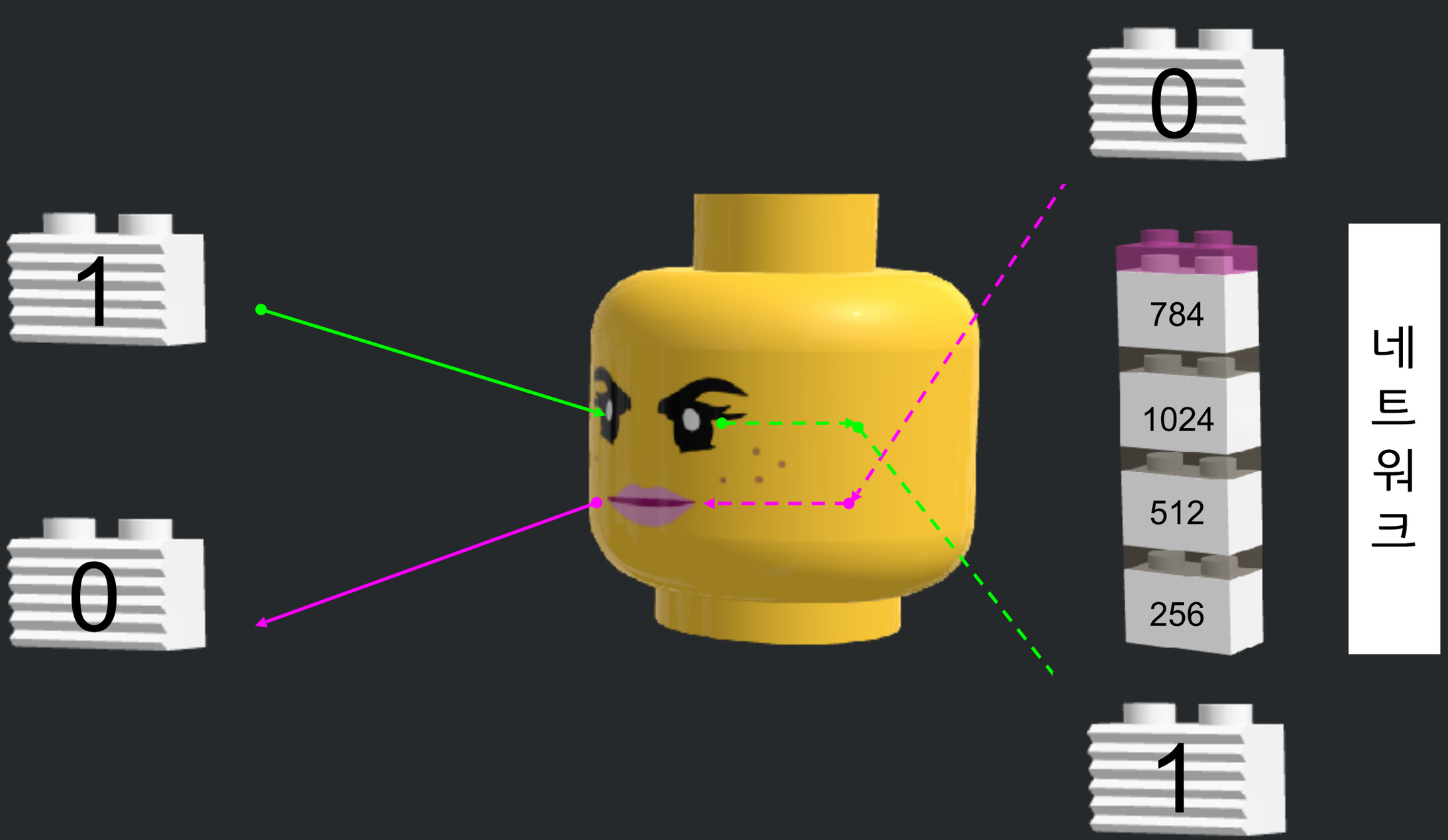
Network
네트워크

Objective Function
목표함수

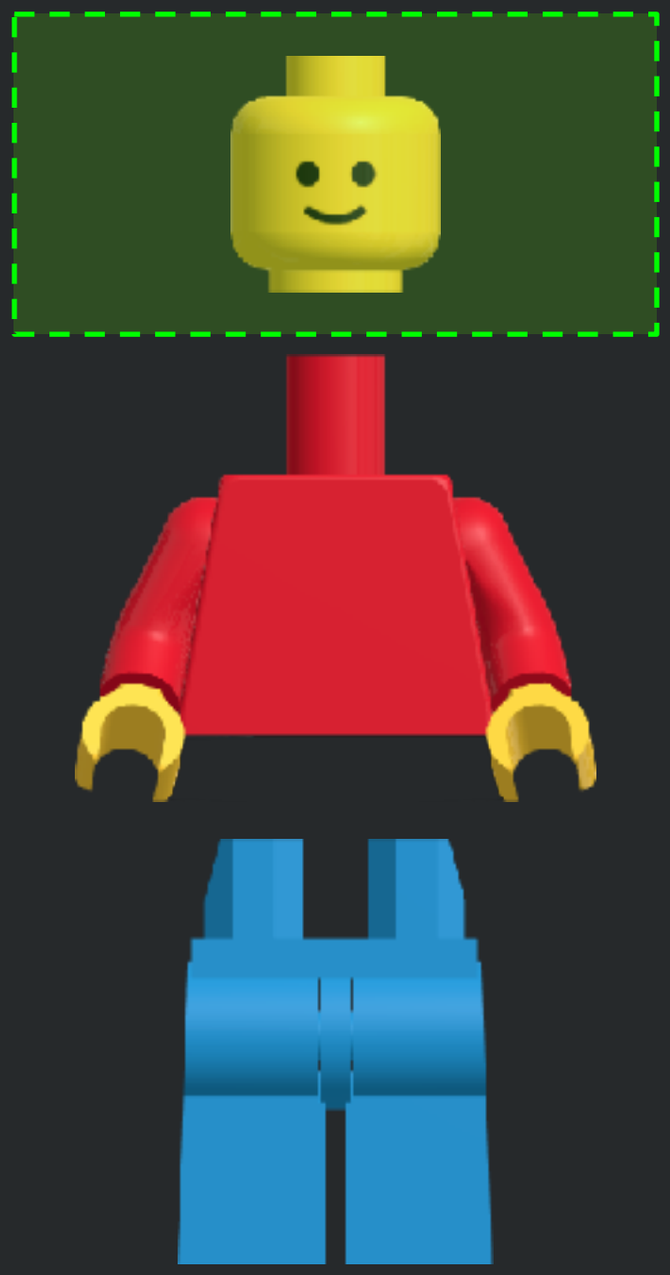
Optimizer
최적화기

Network
네트워크

```
from keras.layers import Dense

model.add(Dense(units=64, activation='relu', input_dim=100))
model.add(Dense(units=10, activation='softmax'))
```



블록과 함께하는 파이썬 딥러닝 케라스, (주)인스페이스 기술이사 김태영

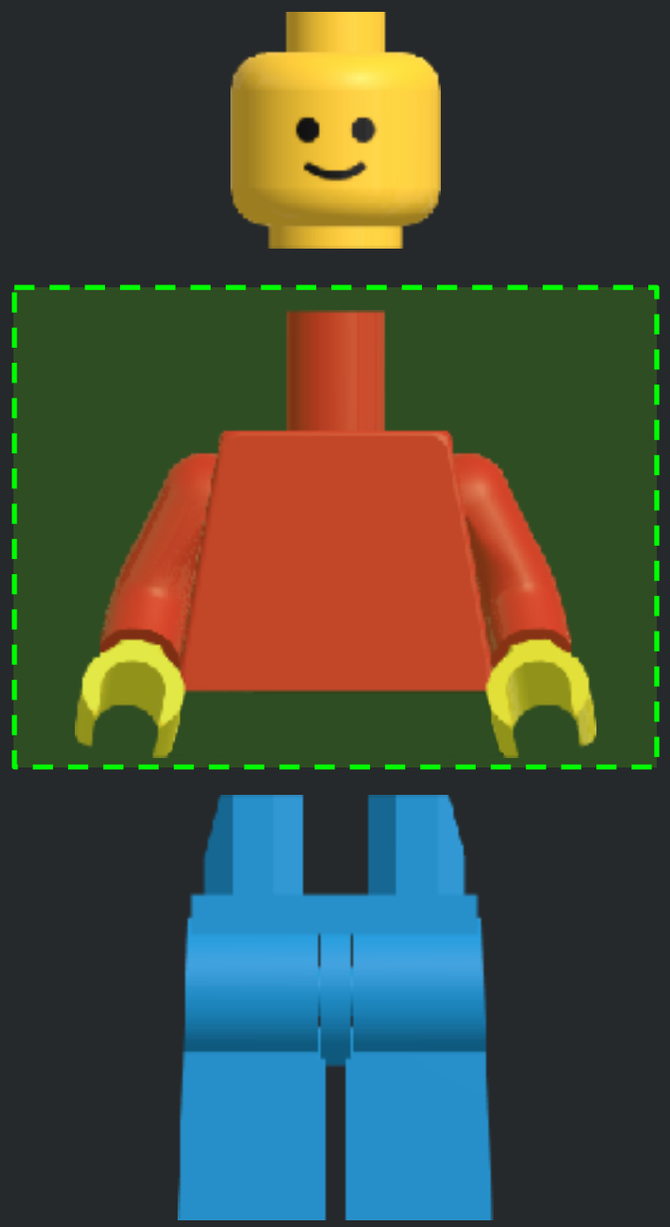


https://keras.io/applications/

- Xception (88 MB, 126)
- VGG16 (528 MB, 23)
- VGG19 (549 MB, 26)
- ResNet50 (99 MB, 168)
- InceptionV3 (92 MB, 159)
- InceptionResNetV2 (215 MB, 572)
- MobileNet (17 MB, 88)
- DenseNet121 (33 MB, 121)
- DenseNet169 (57 MB, 169)
- DenseNet201 (80 MB, 201)
- NASNet


Xception
VGG16
VGG19
ResNet 50
Inception V3
Inception ResNet V2
Mobile Net
Dense Net
NAS Net

Objective Function
목표함수



https://keras.io/losses/

- mean_squared_error
- mean_absolute_error
- mean_absolute_percentage_error
- mean_squared_logarithmic_error
- squared_hinge
- hinge
- categorical_hinge
- logcosh
- categorical_crossentropy
- sparse_categorical_crossentropy
- binary_crossentropy
- kullback_leibler_divergence
- poisson
- cosine_proximity

Cost Function

= loss function

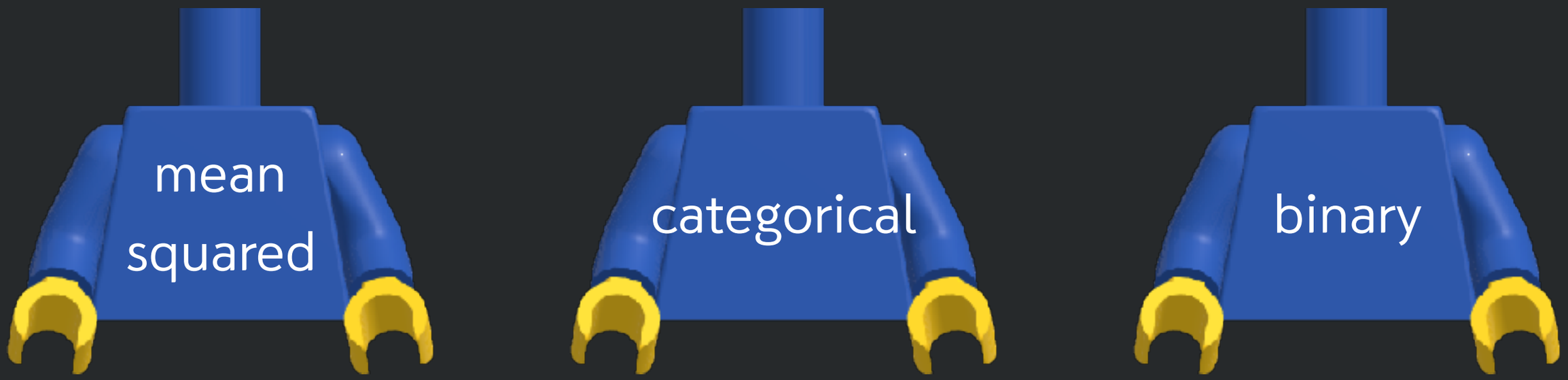
= error function

= objective function

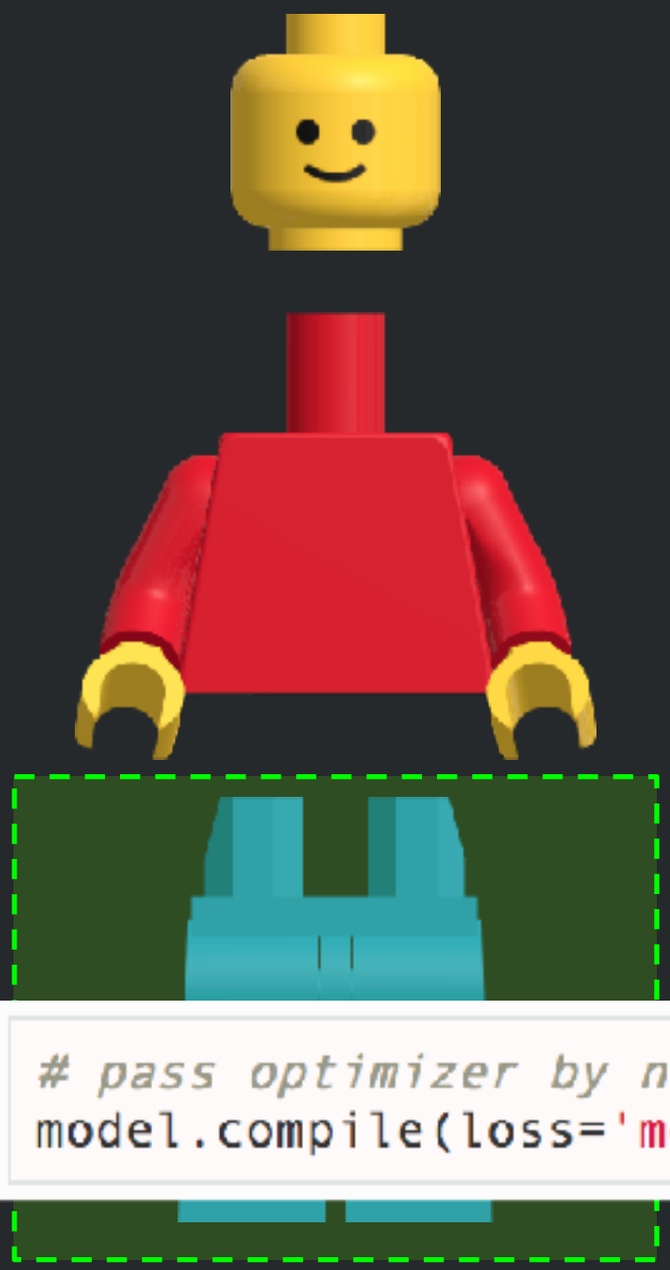
$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

예측 값과 실제 값의 차이를 기반으로

모델의 정확도(성능)을 판단하기 위한 함수


mean
squared
categorical
binary

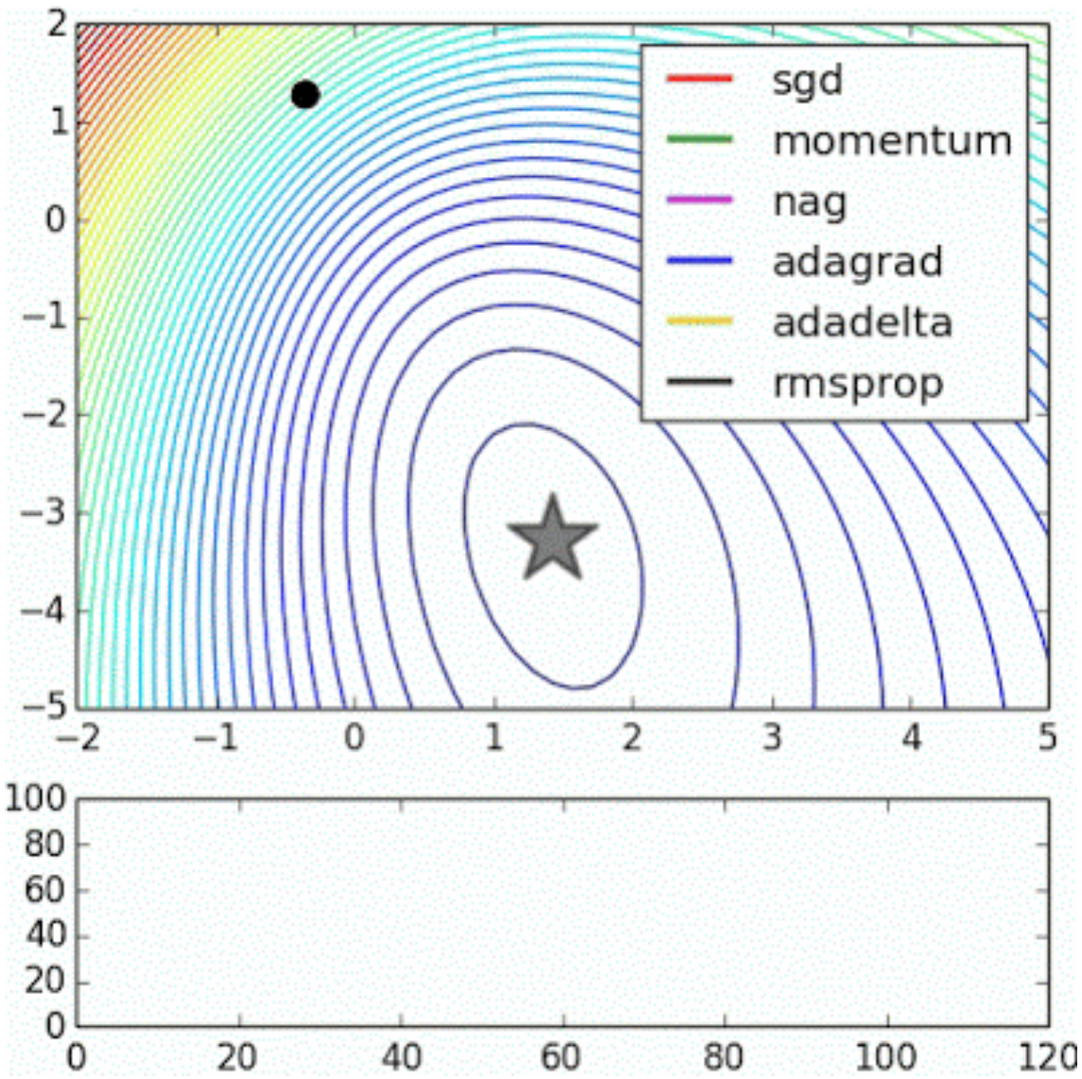
Optimizer
최적화기



https://keras.io/optimizers/

- SGD
- RMSprop
- Adagrad
- Adadelta
- Adam
- Adamax
- Nadam

```
# pass optimizer by name: default parameters will be used
model.compile(loss='mean_squared_error', optimizer='sgd')
```



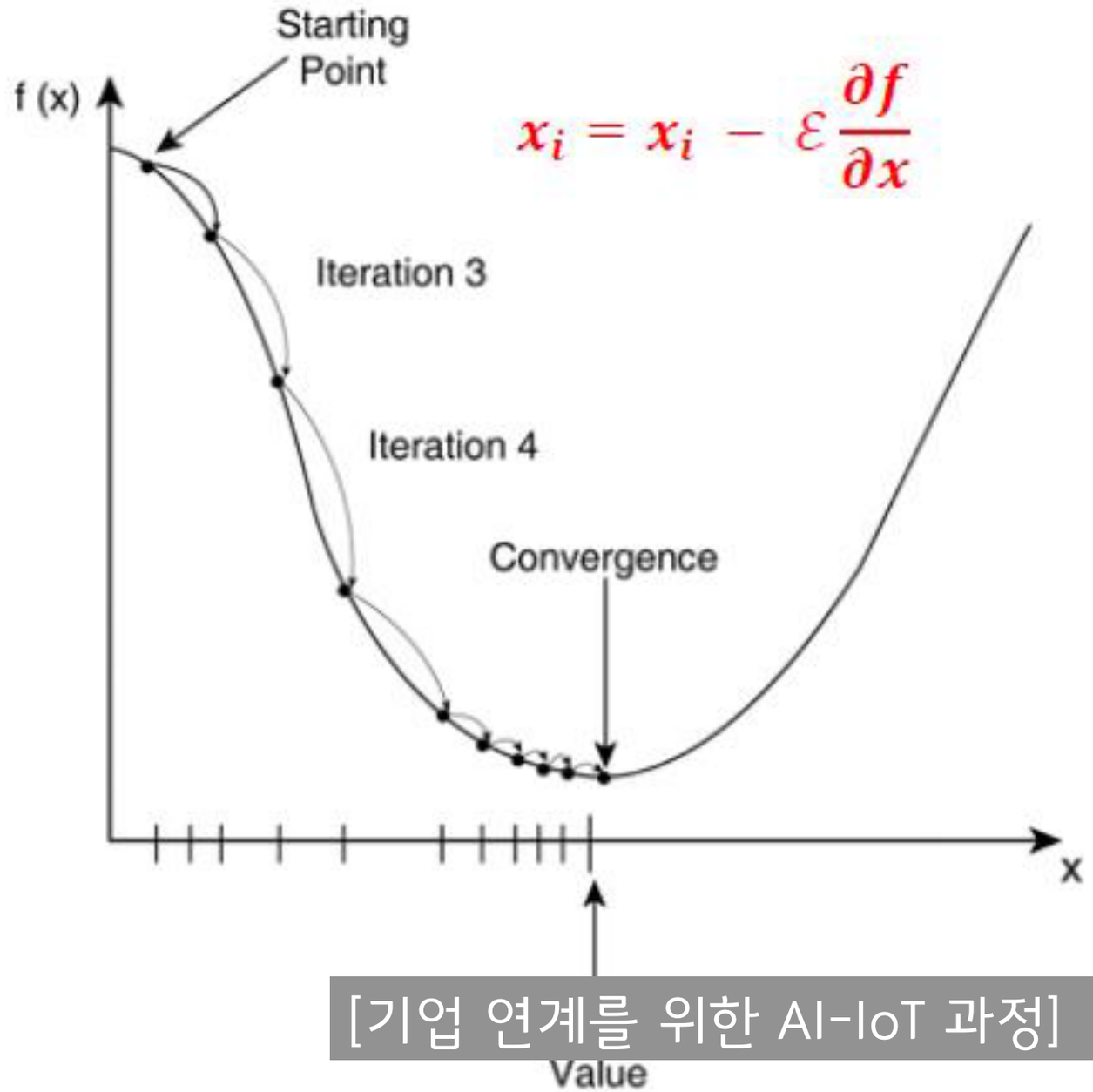
Azure

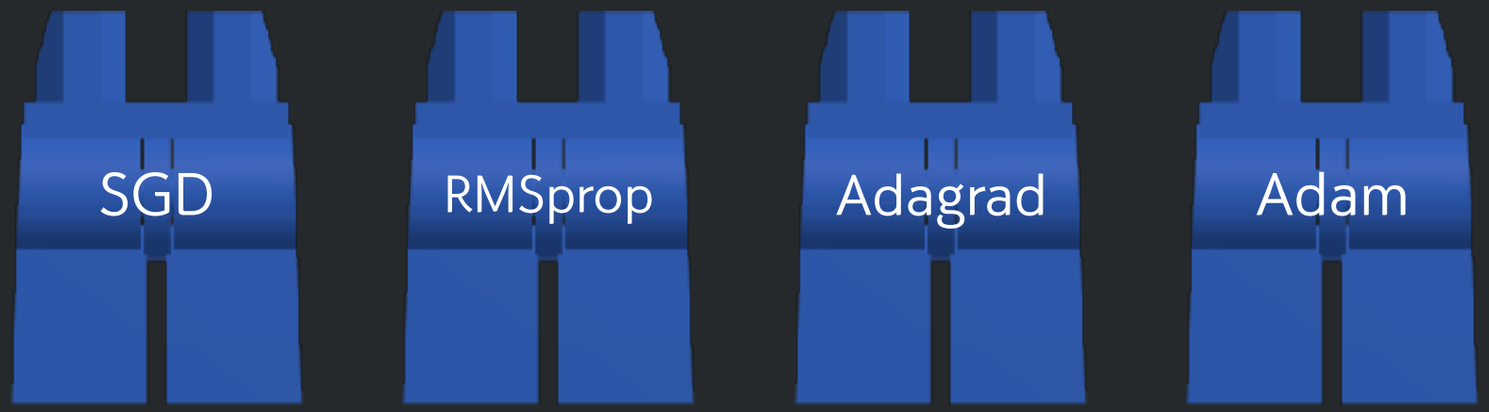
http://2.bp.blogspot.com/-q6l20Vs4P_w/VPmIC7sEhnI/AAAAAAAACC4/g3UOUX2r_yA/s400/s25RsOr%

Gradient Descent

cost function 미분 (= 기울기)

기울기 < 0




SGD
RMSprop
Adagrad
Adam

Compile
컴파일

Network
네트워크

Objective Function
목표함수

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

Model
모델

네트워크
network

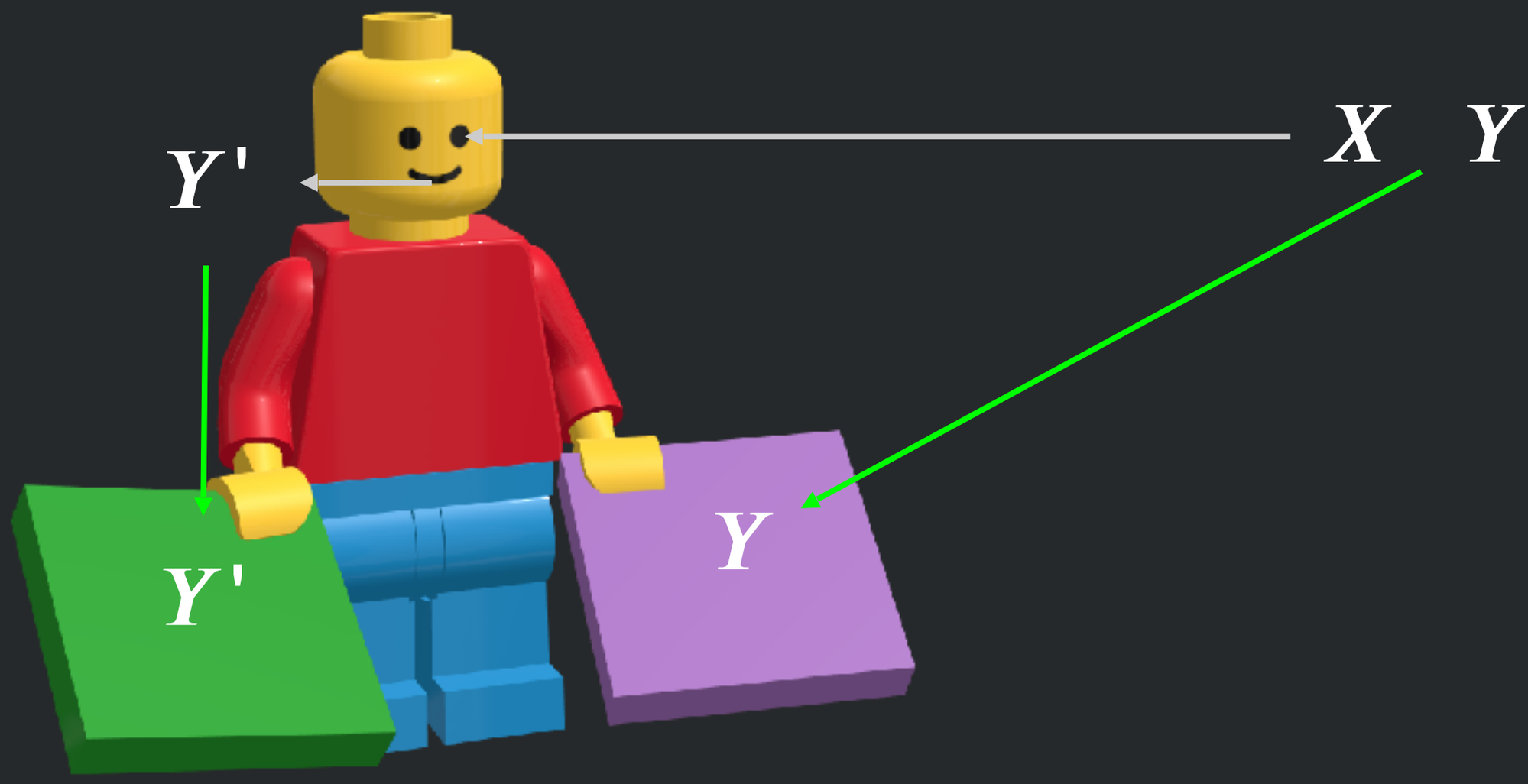
목표함수
objective function

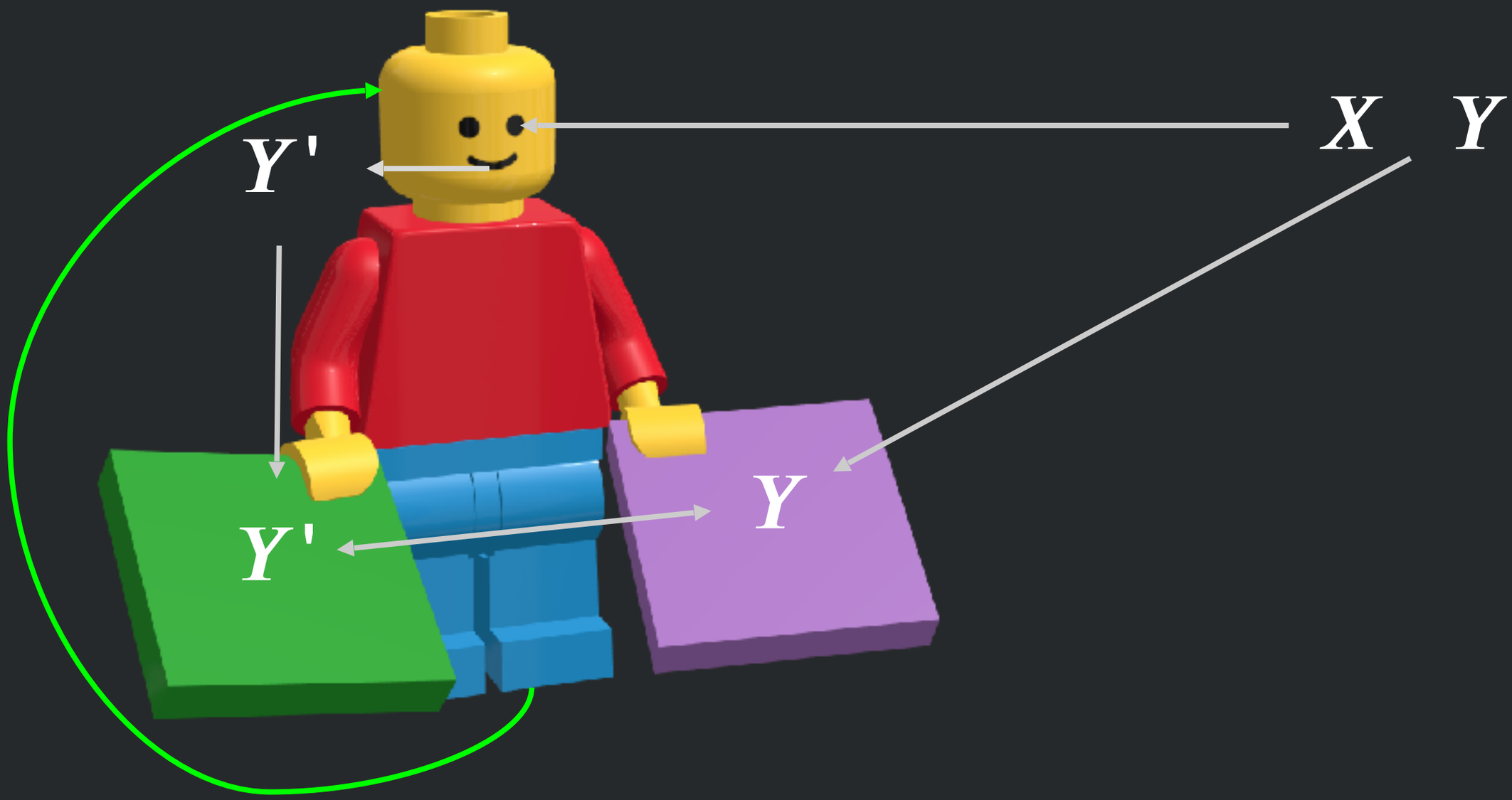
최적화기
optimizer


binary
adam

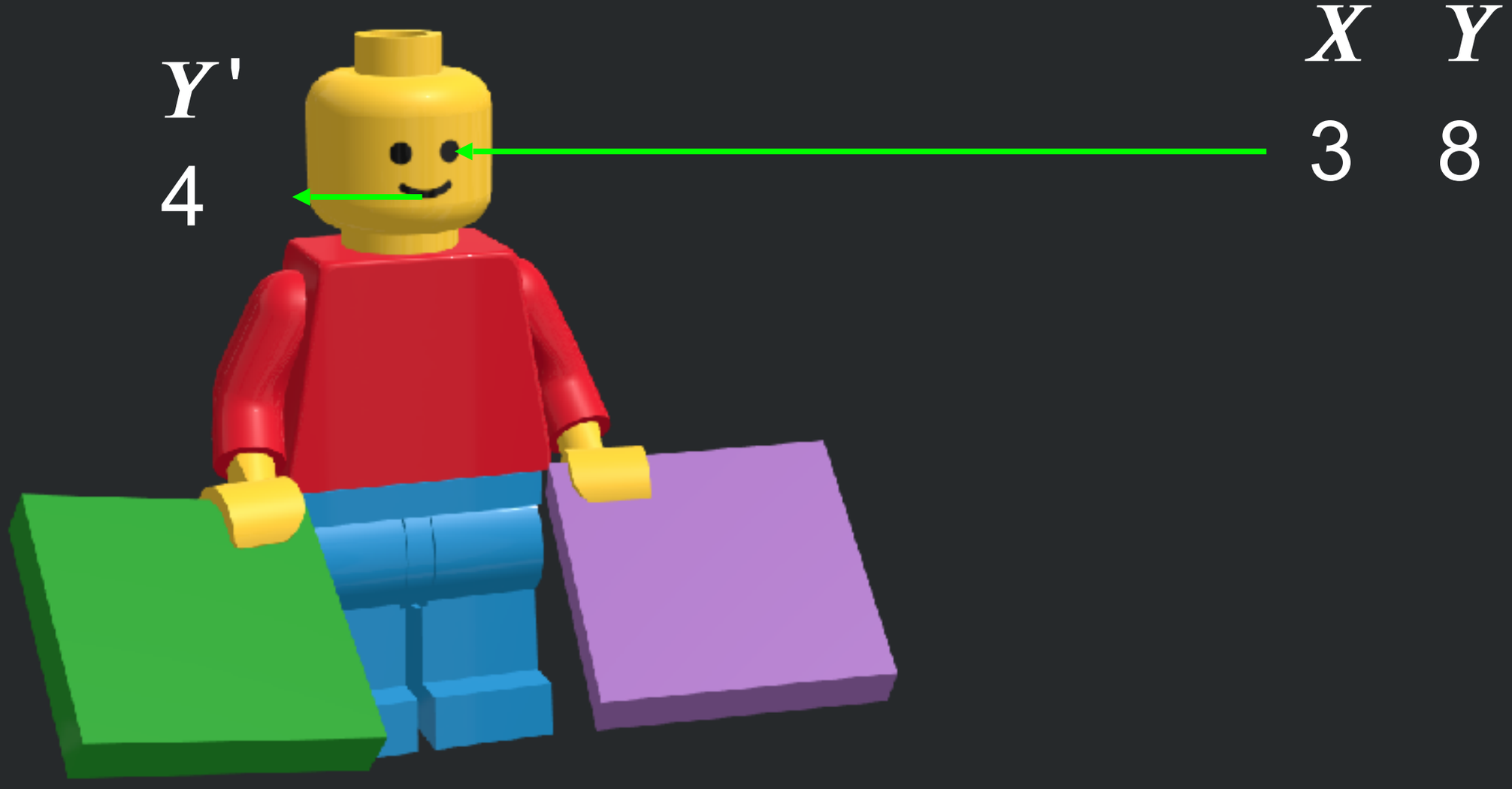

categorical
sgd

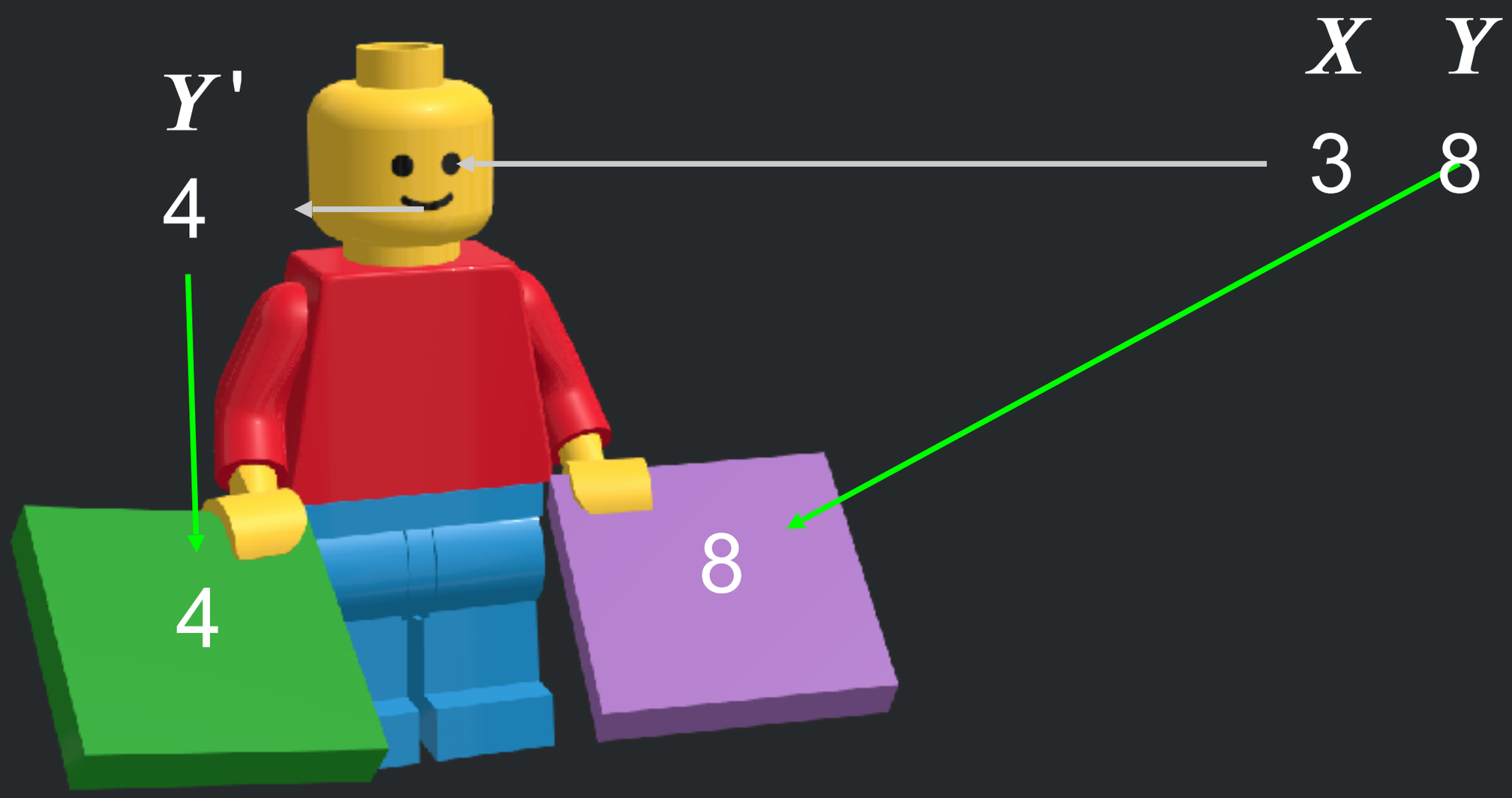

Y'
X
Y

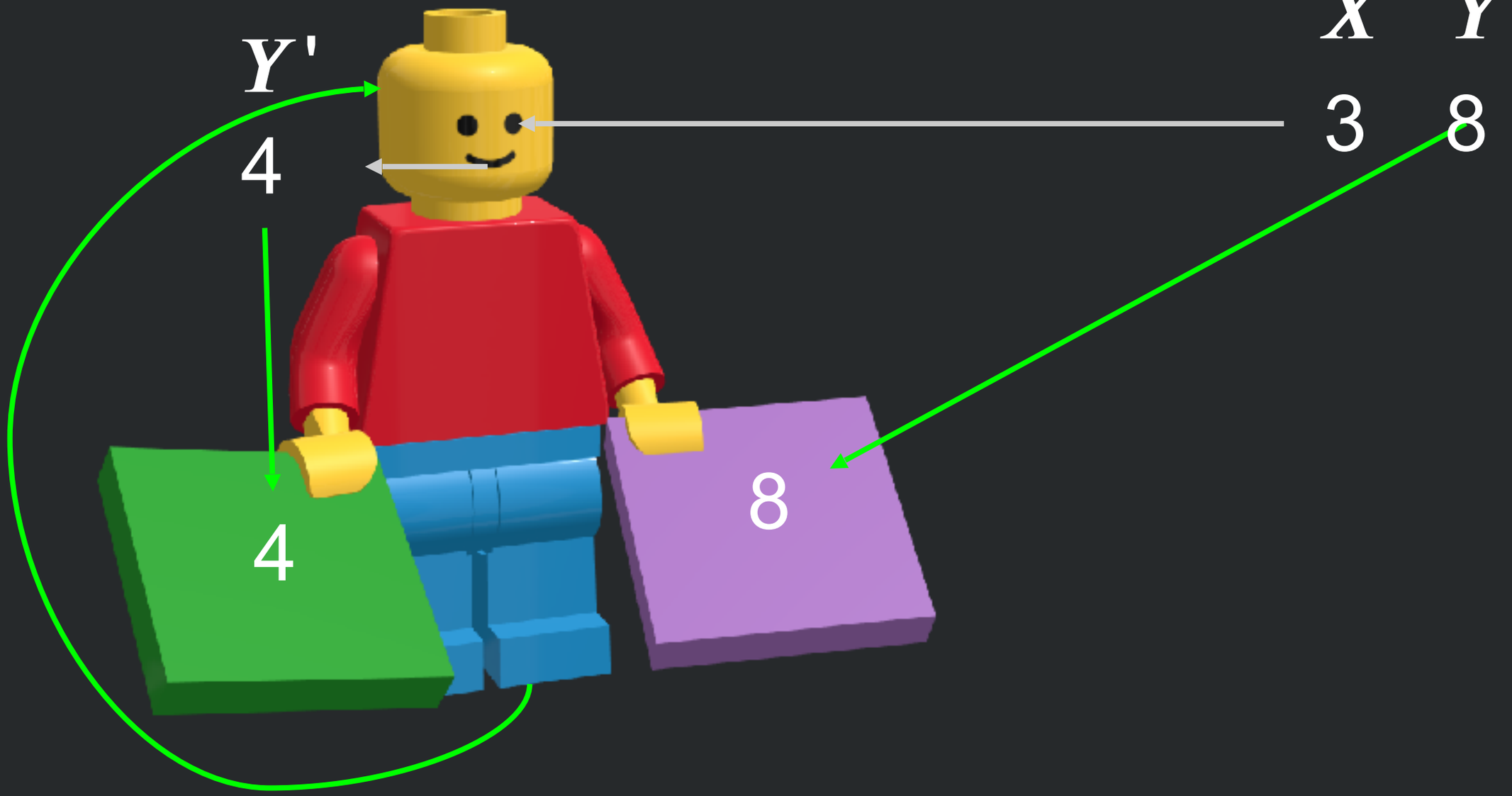

Y'
X
Y
Y'
Y



블록과 함께하는 파이썬 딥러닝 케라스, (주)인스페이스 기술이사 김태영


Y'
4
X Y
3 8


Y'
4
X
3
Y
8
4
8


Y'
4
X Y
3 8
4
8

Deep Learning

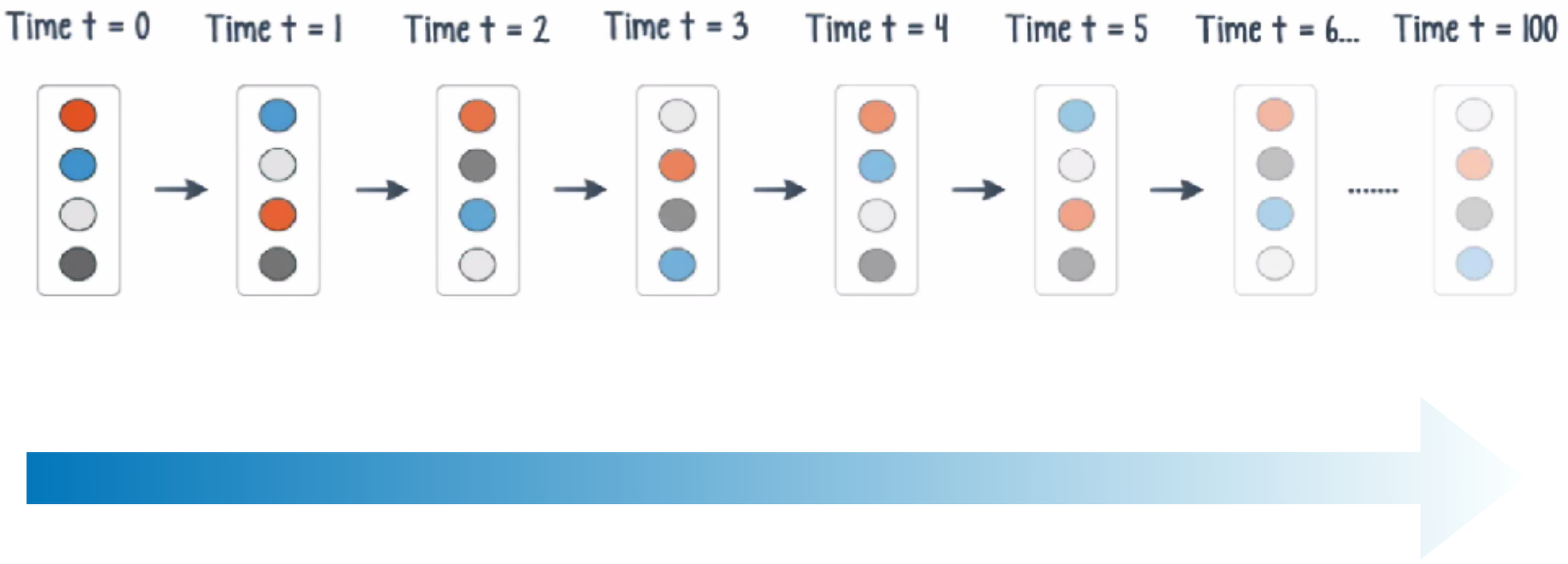
(Deep) Neural Network

MLP(Multi Layer Perceptron)

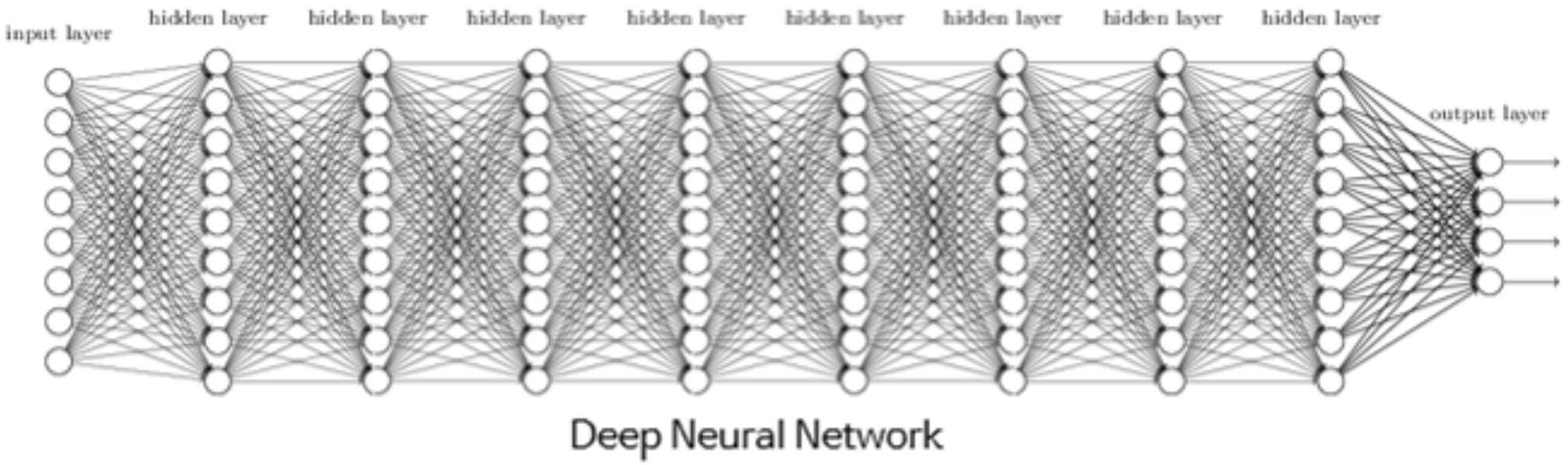
Vanishing Gradient

Overfitting

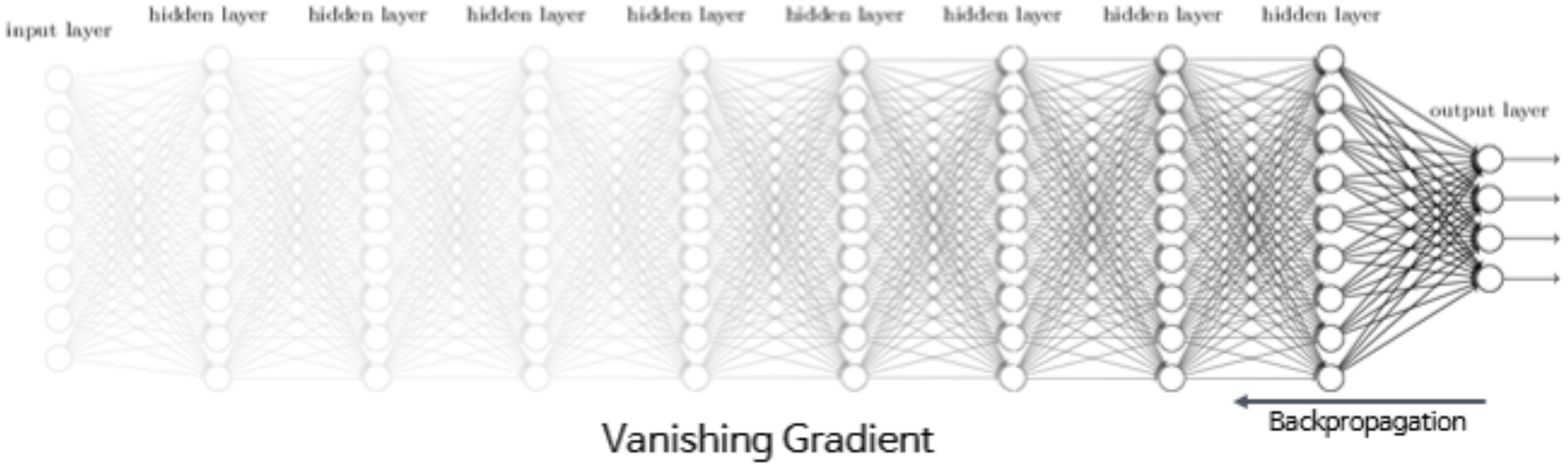
Decay of information through time



https://medium.com/@anishsingh20/the-vanishing-gradient-problem-48ae7f501257



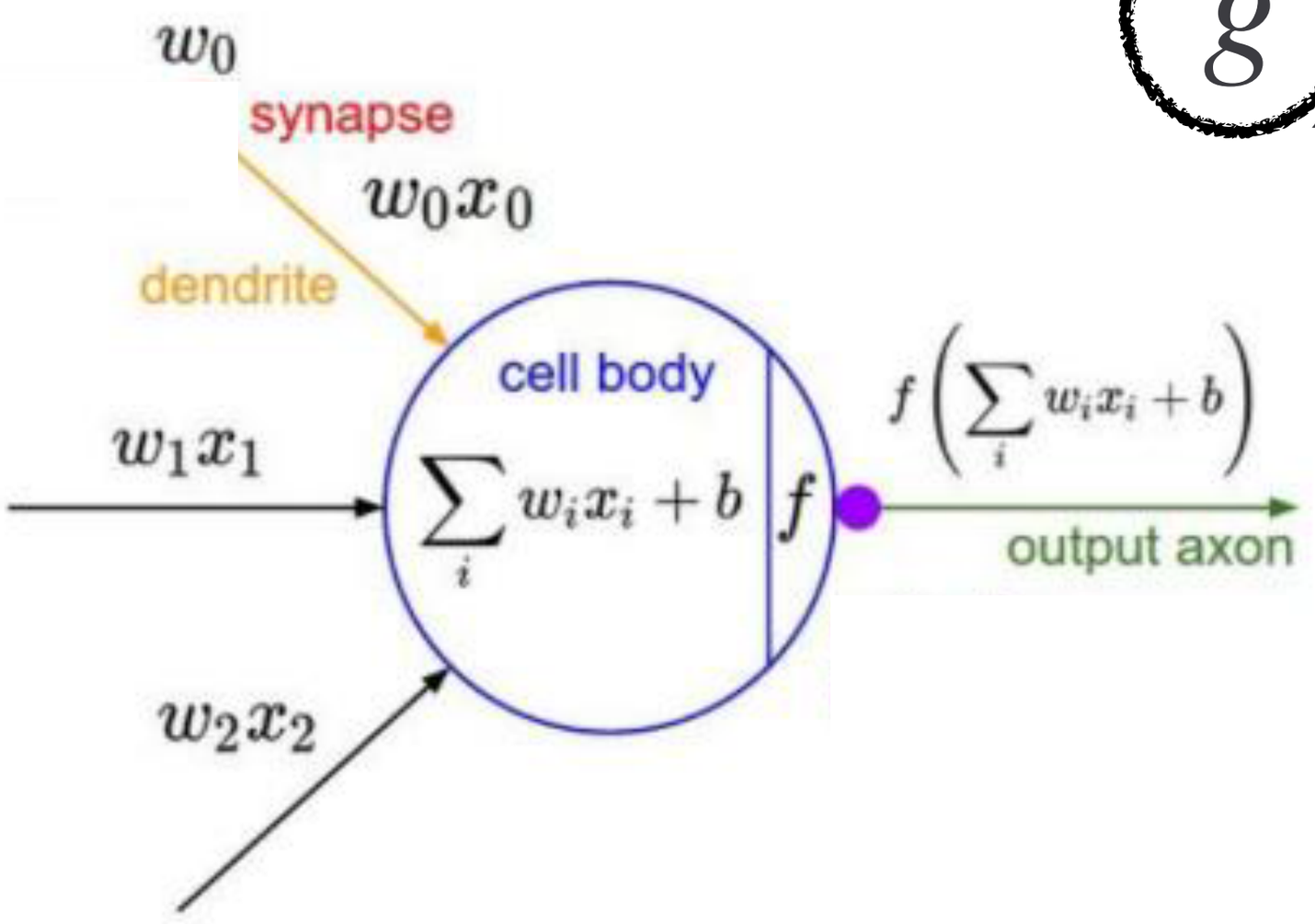
Deep Neural Network

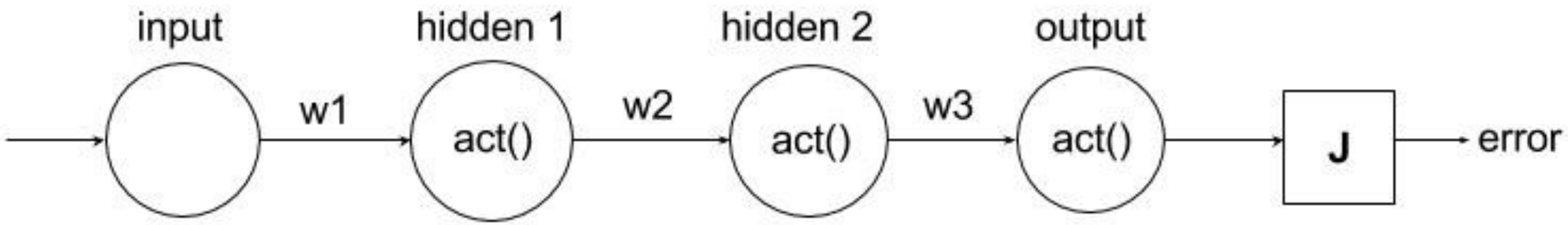


Vanishing Gradient

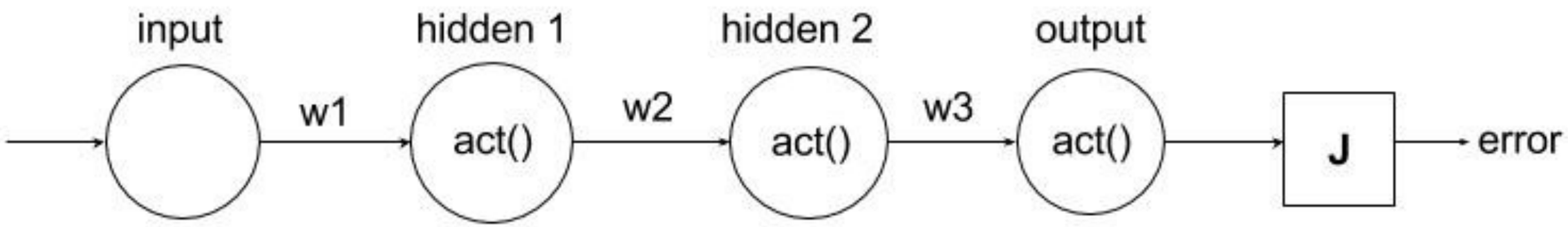
$$w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_nx_n$$

$$g \sum (w_ix_i + b)$$

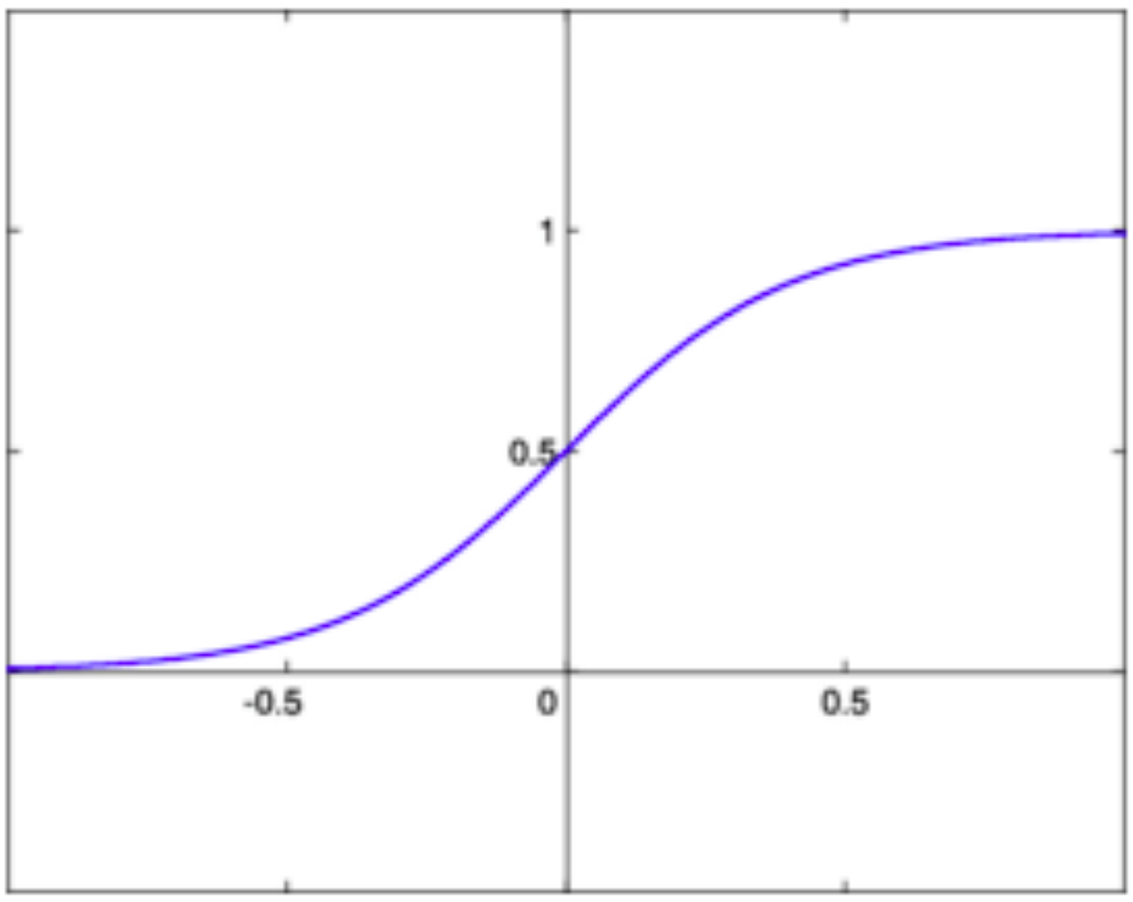


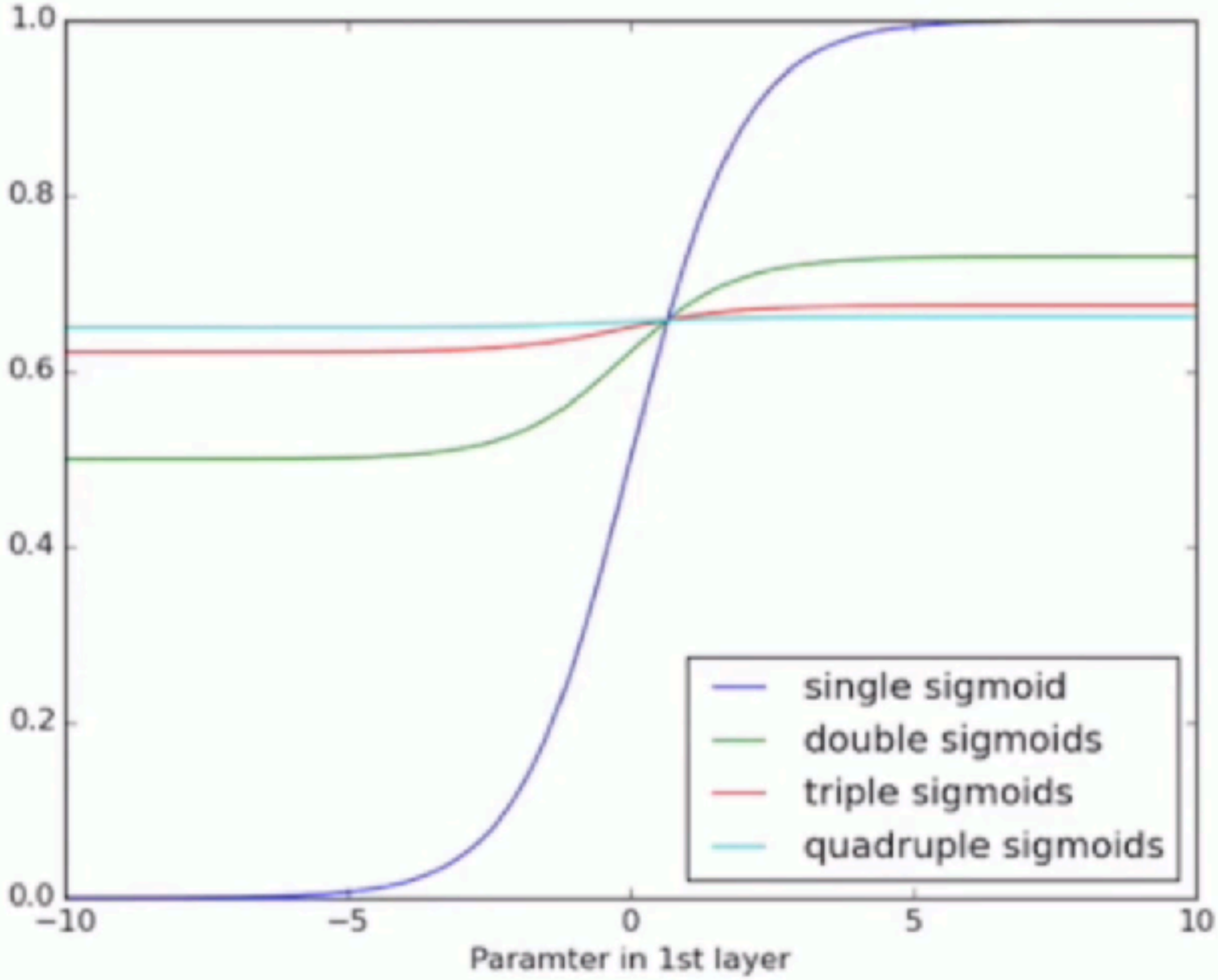


https://ayearofai.com/rohan-4-the-vanishing-gradient-problem-ec68f76ffb9b



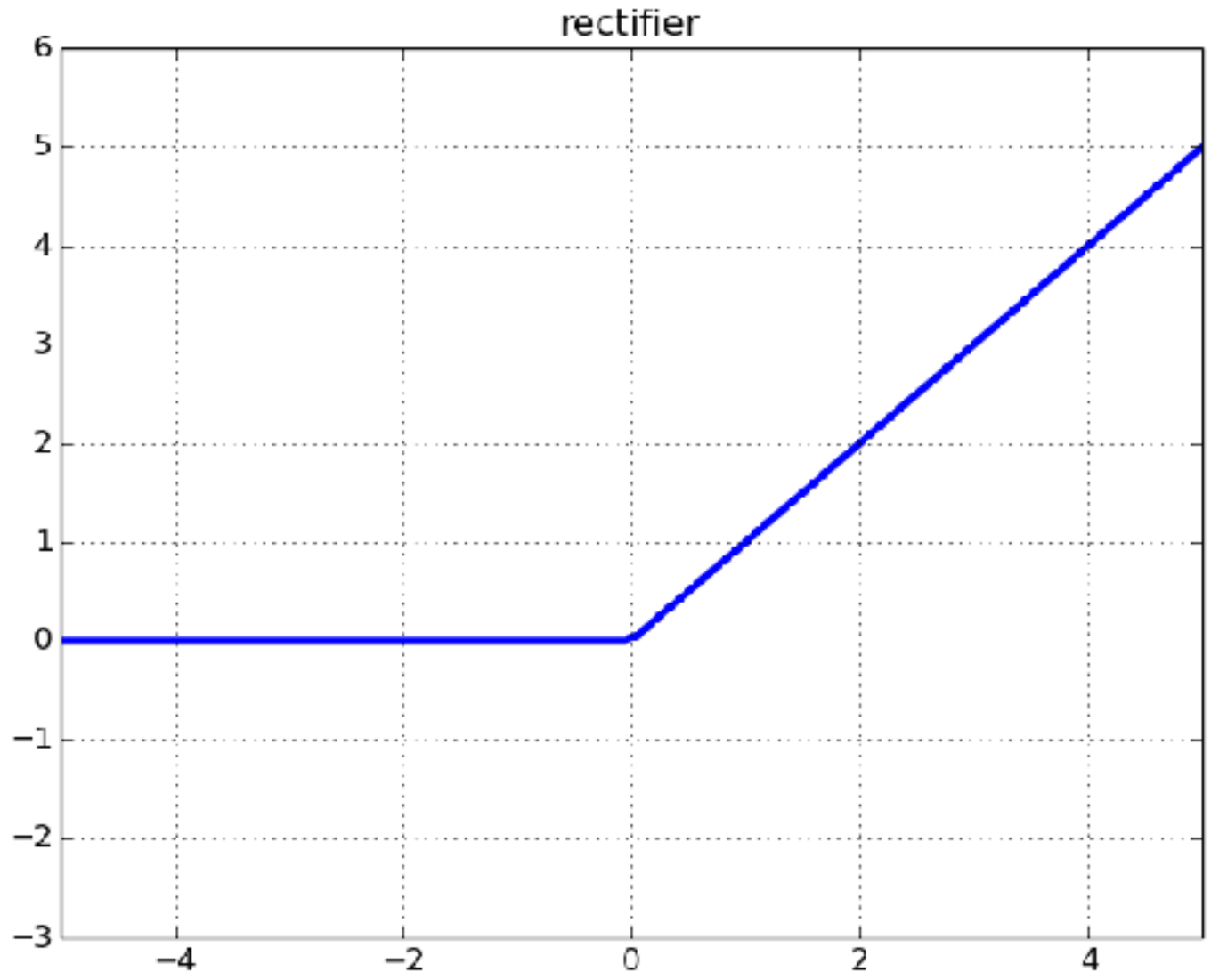
$$Sigmoid = S(\alpha) = \frac{1}{1+e^{-\alpha}}$$





http://skymind.ai/wiki

$$RELU(x) = \begin{cases} 0 \ if \ x < 0 \\ x \ if \ x >= 0 \end{cases}$$

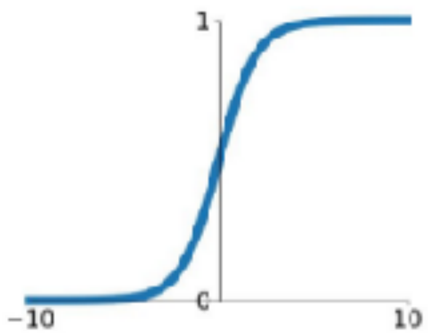


https://ayearofai.com/rohan-4-the-vanishing-gradient-problem-ec68f76ffb9b

Activation Functions

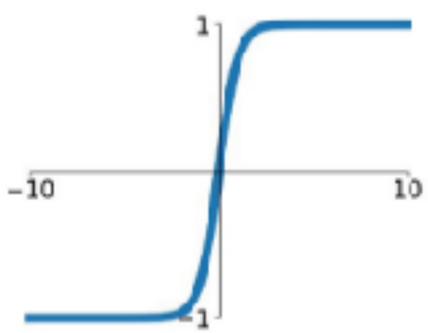
Sigmoid

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



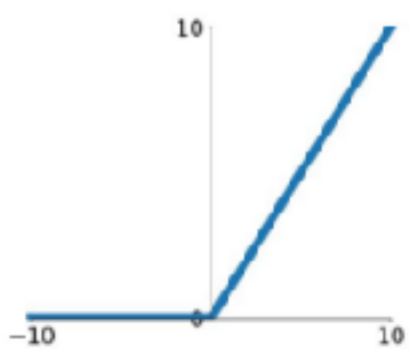
tanh

$\tanh(x)$



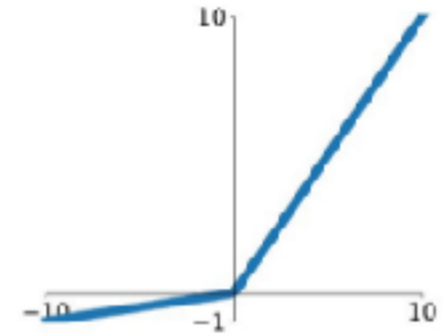
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

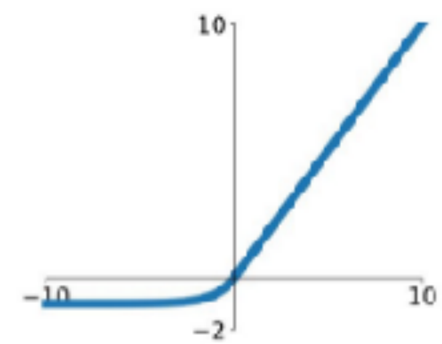


Maxout

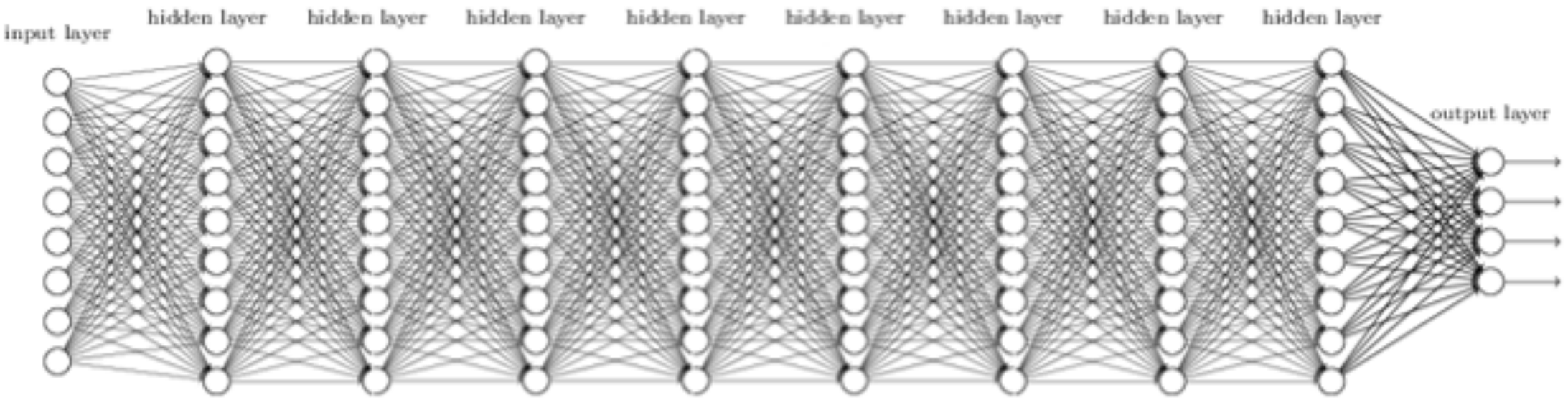
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$



https://www.quora.com/Why-is-ReLU-the-most-common-activation-function-used-in-neural-networks



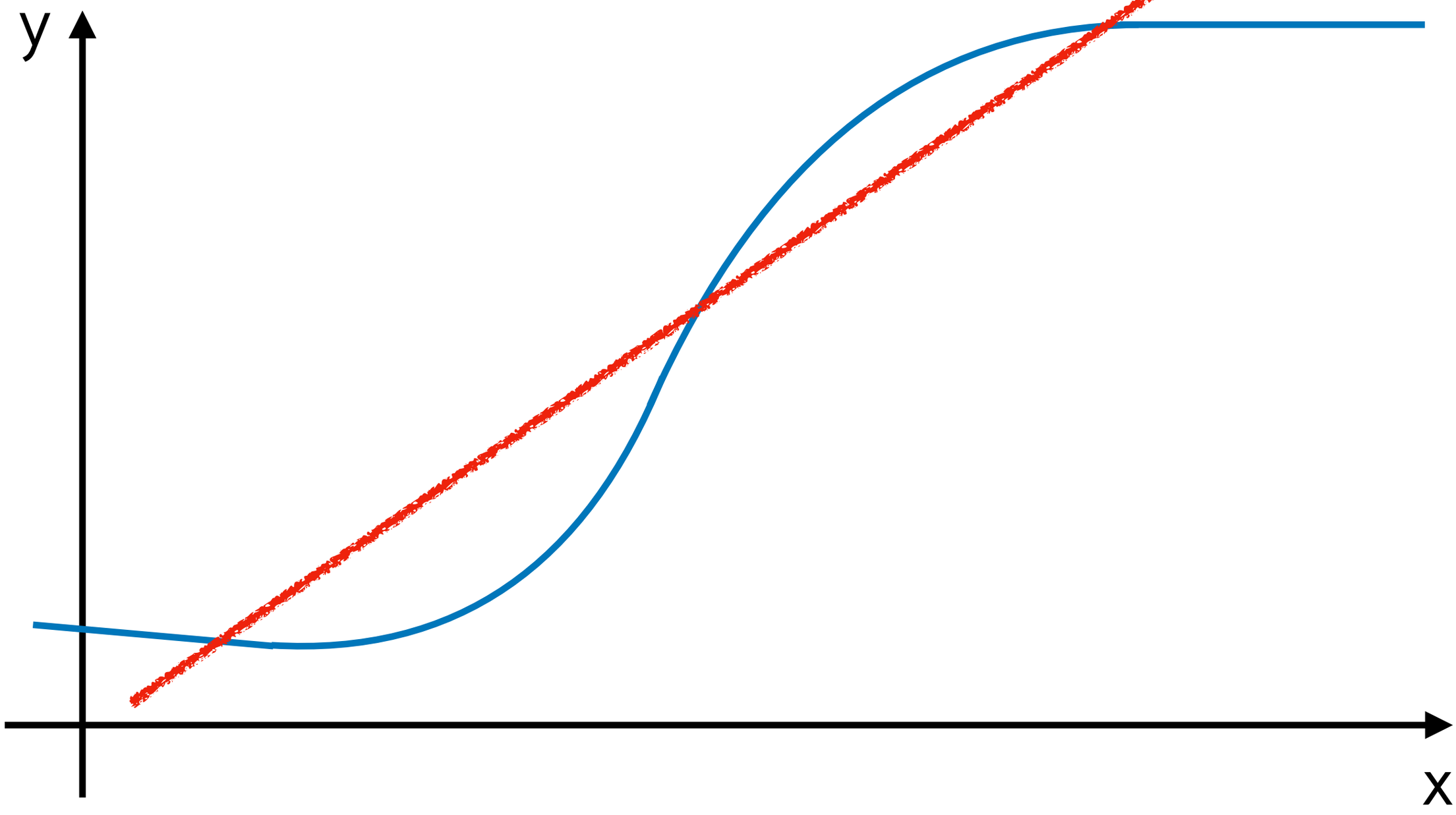
Deep Neural Network

Vanishing Gradient

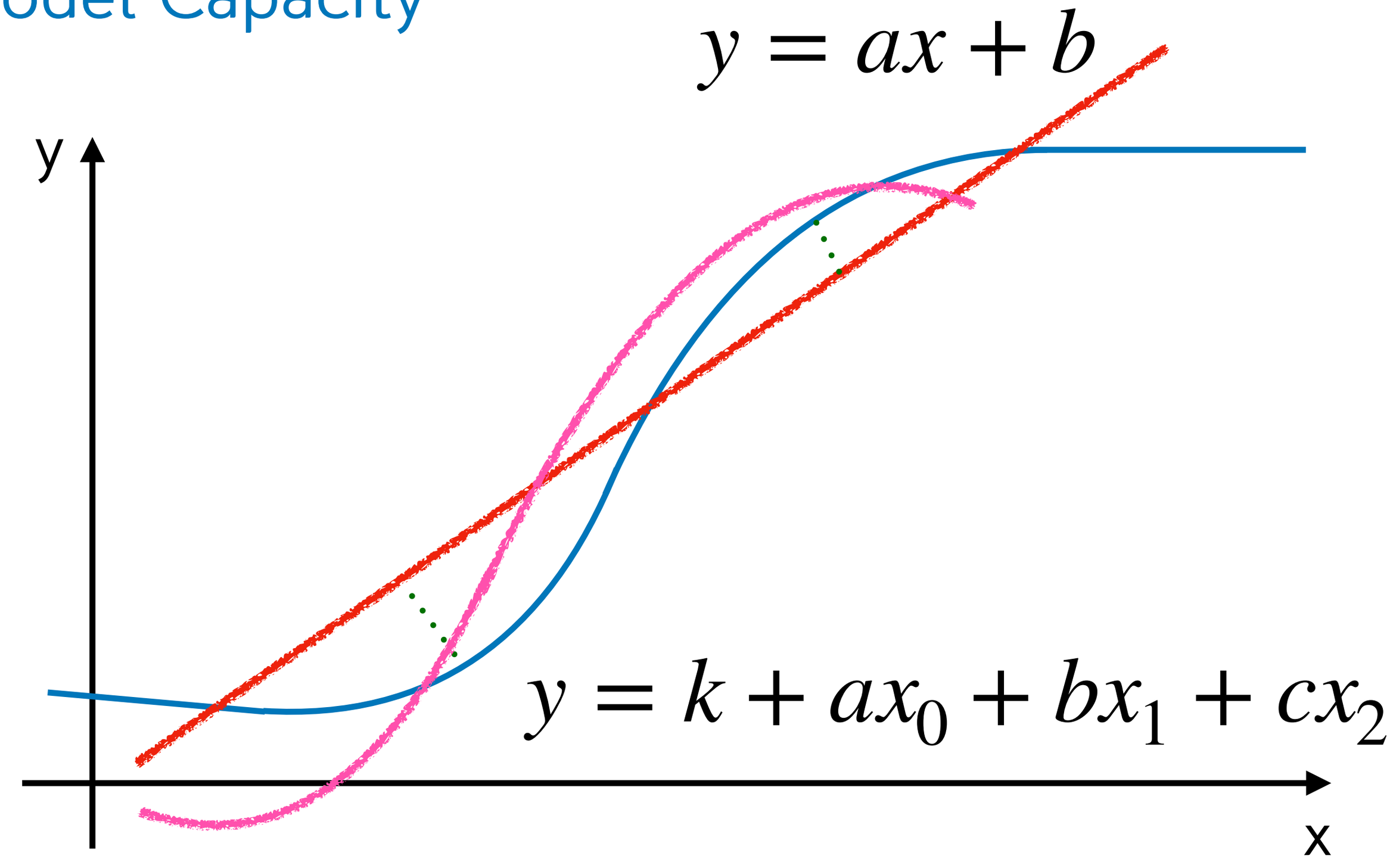
Overfitting

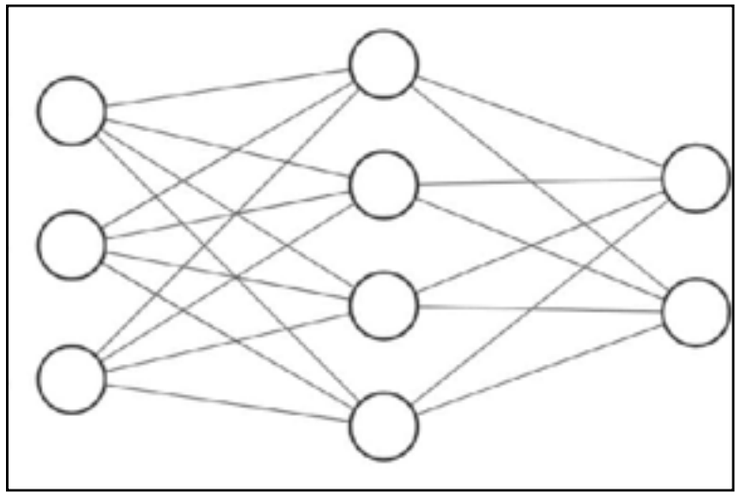
Model Capacity

$$y = ax + b$$

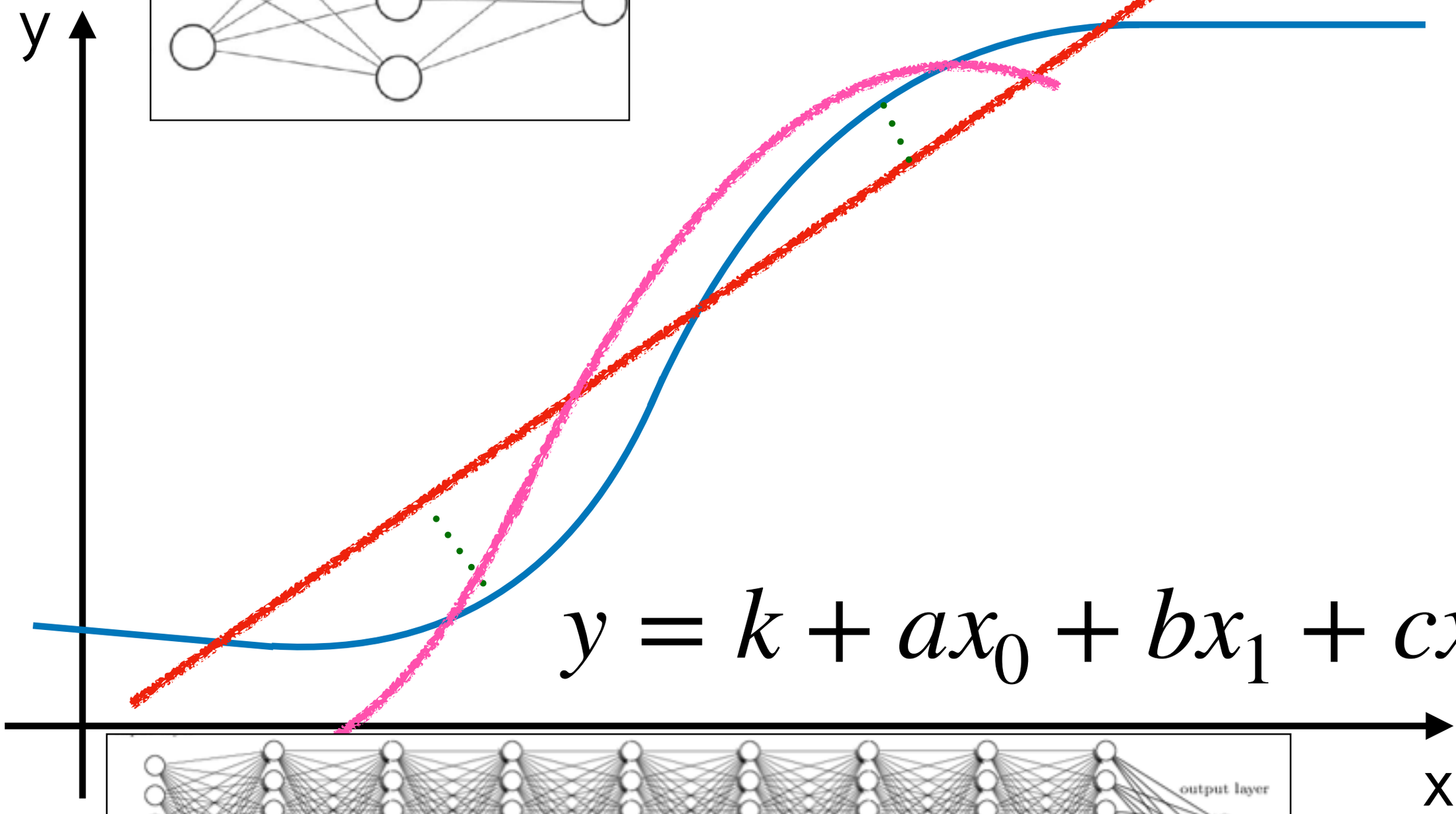


Model Capacity

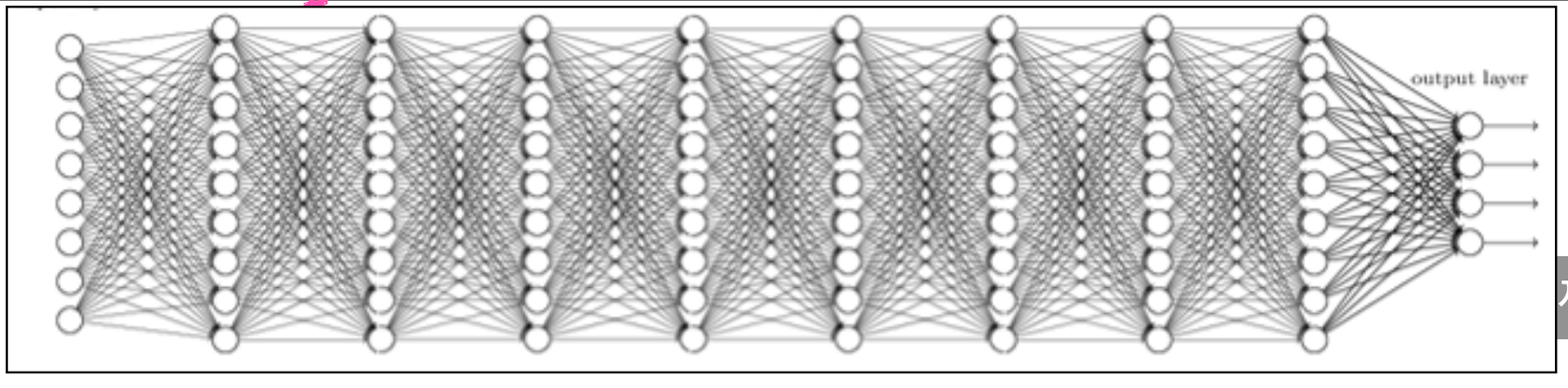


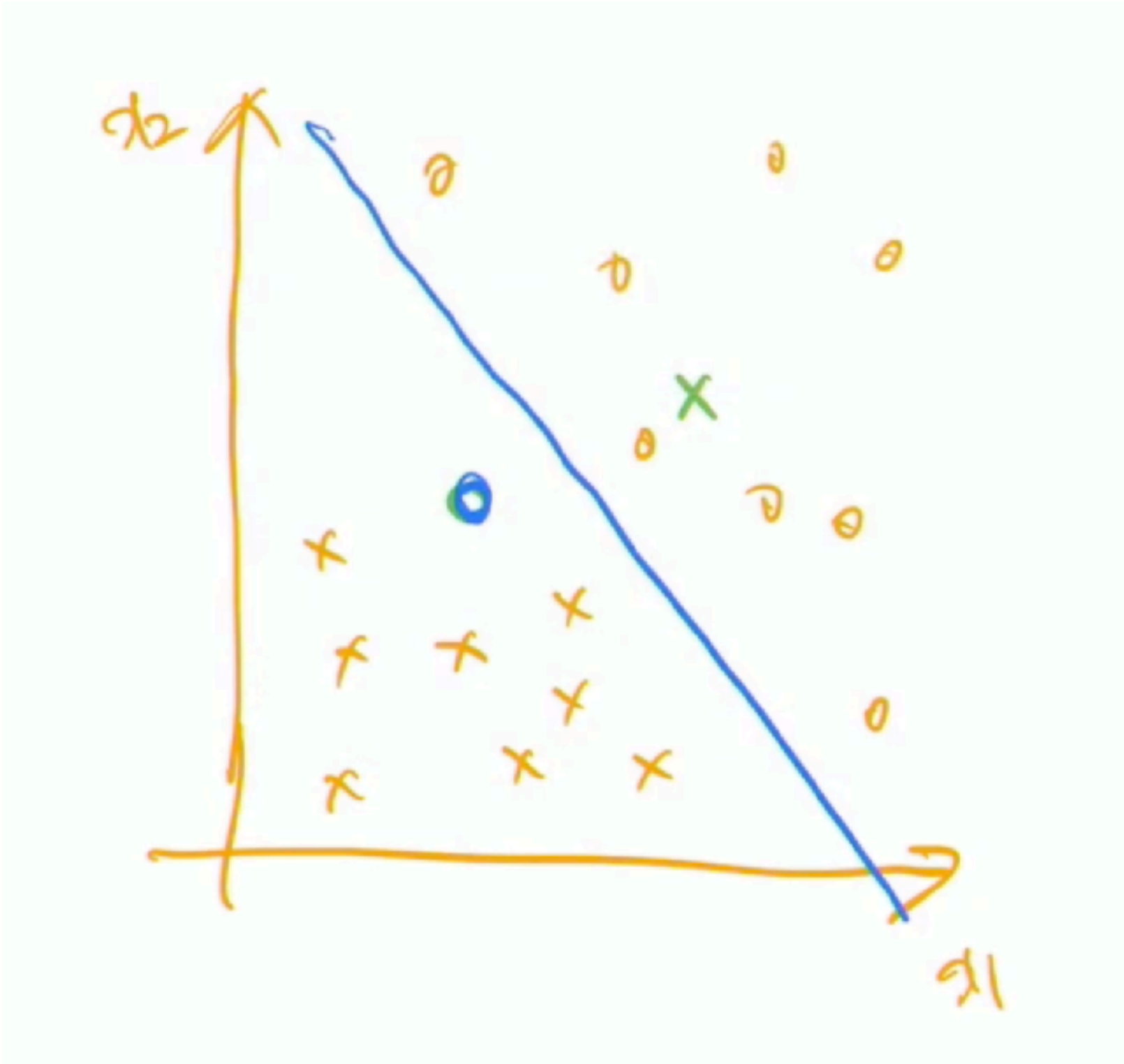


$$y = ax + b$$



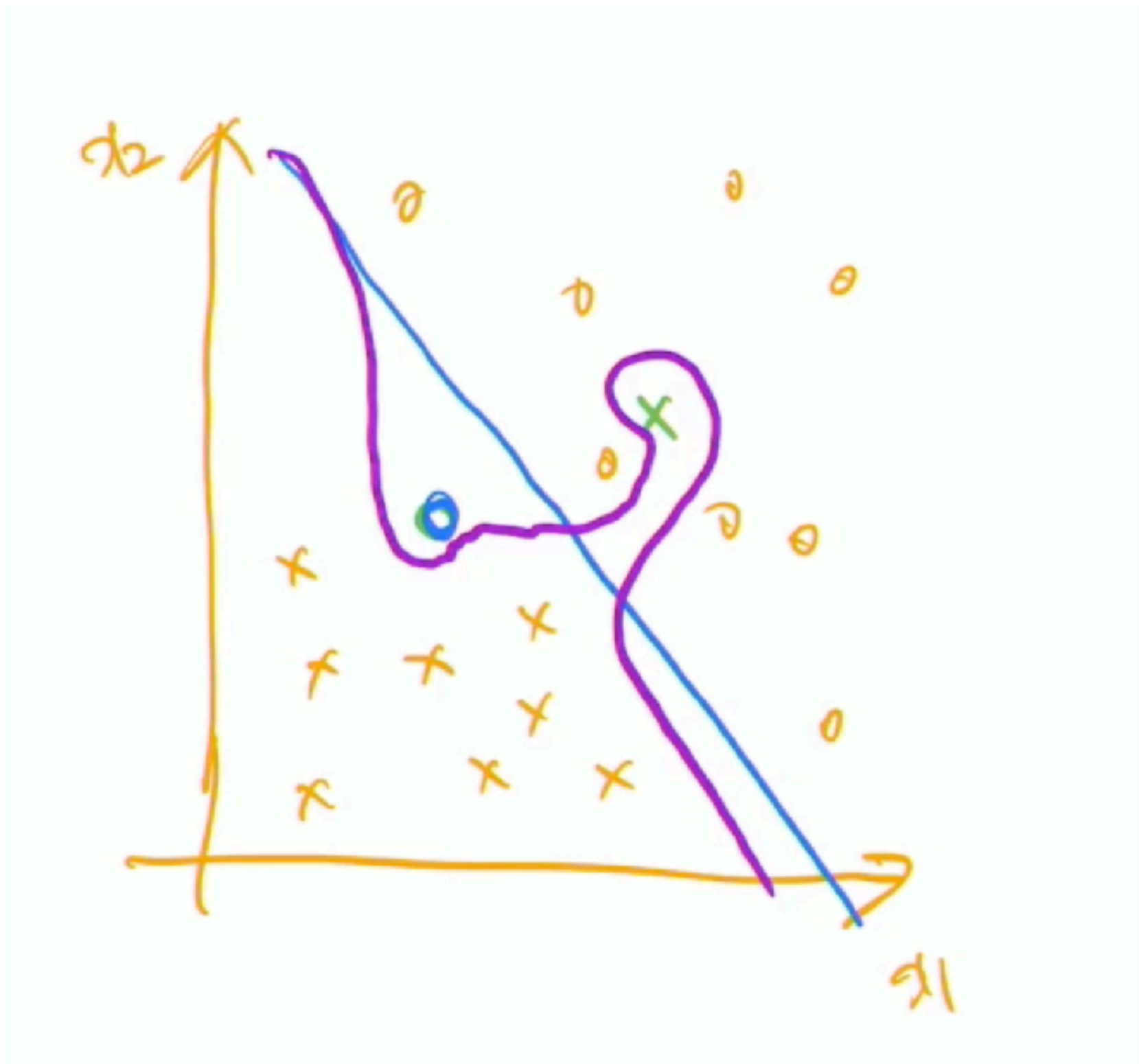
$$y = k + ax_0 + bx_1 + cx_2$$





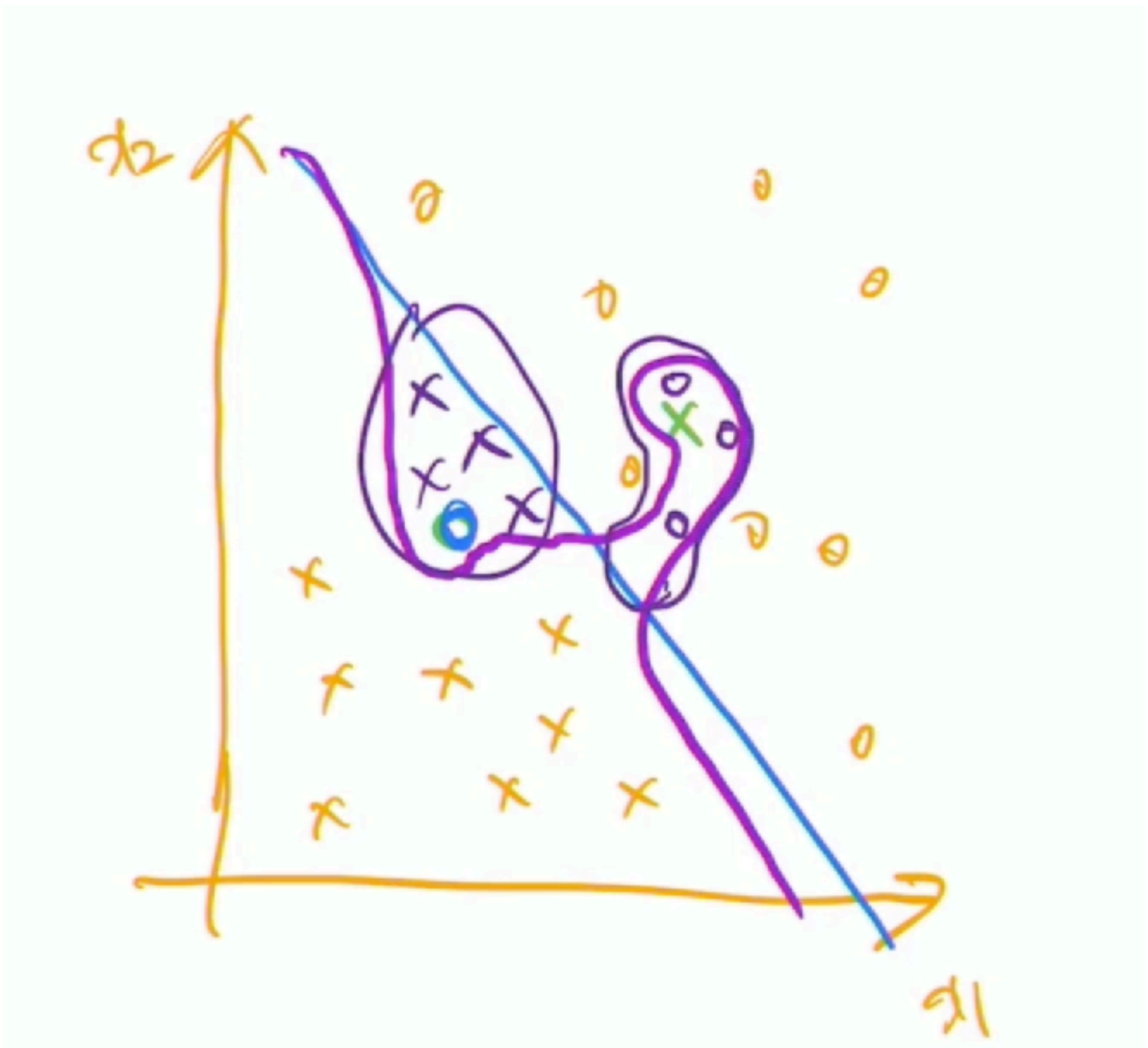
https://youtu.be/_sz3KTyB9Lk

[기업 연계를 위한 AI-IoT 과정] AI | 세번째 날 | 전미정



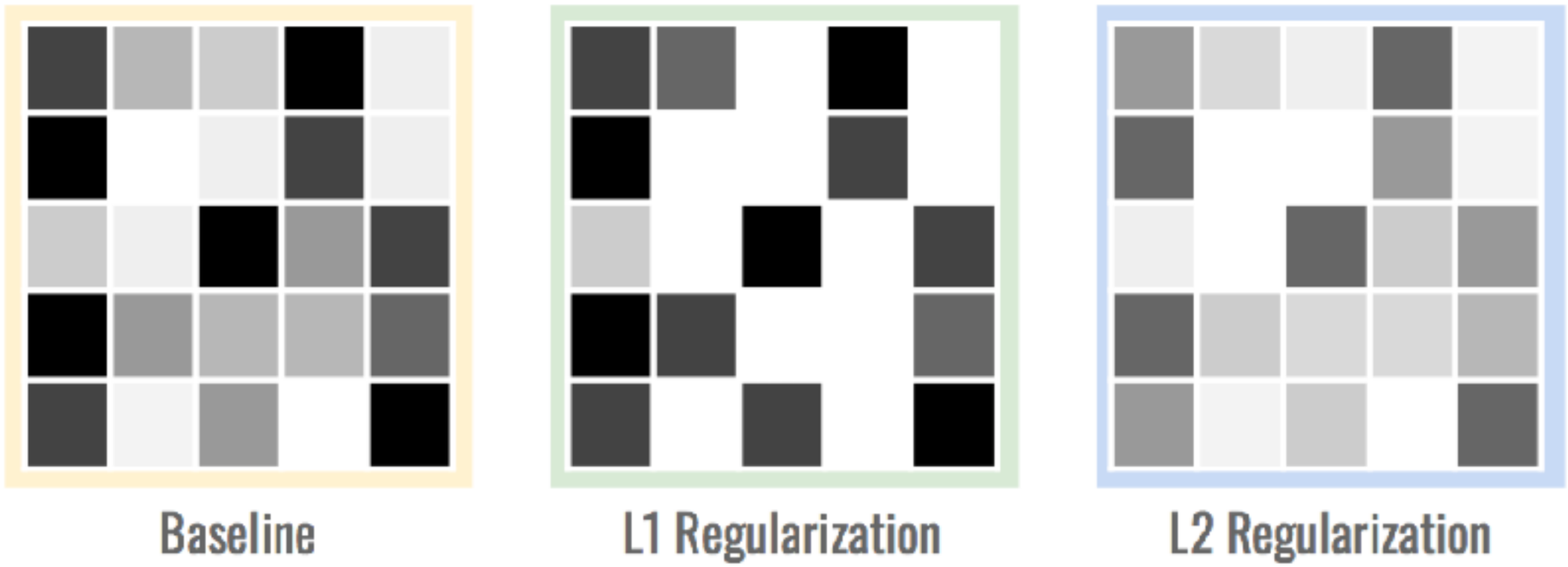
https://youtu.be/_sz3KTyB9Lk

[기업 연계를 위한 AI-IoT 과정] AI | 세번째 날 | 전미정



https://youtu.be/_sz3KTyB9Lk

L1/L2 Regularization



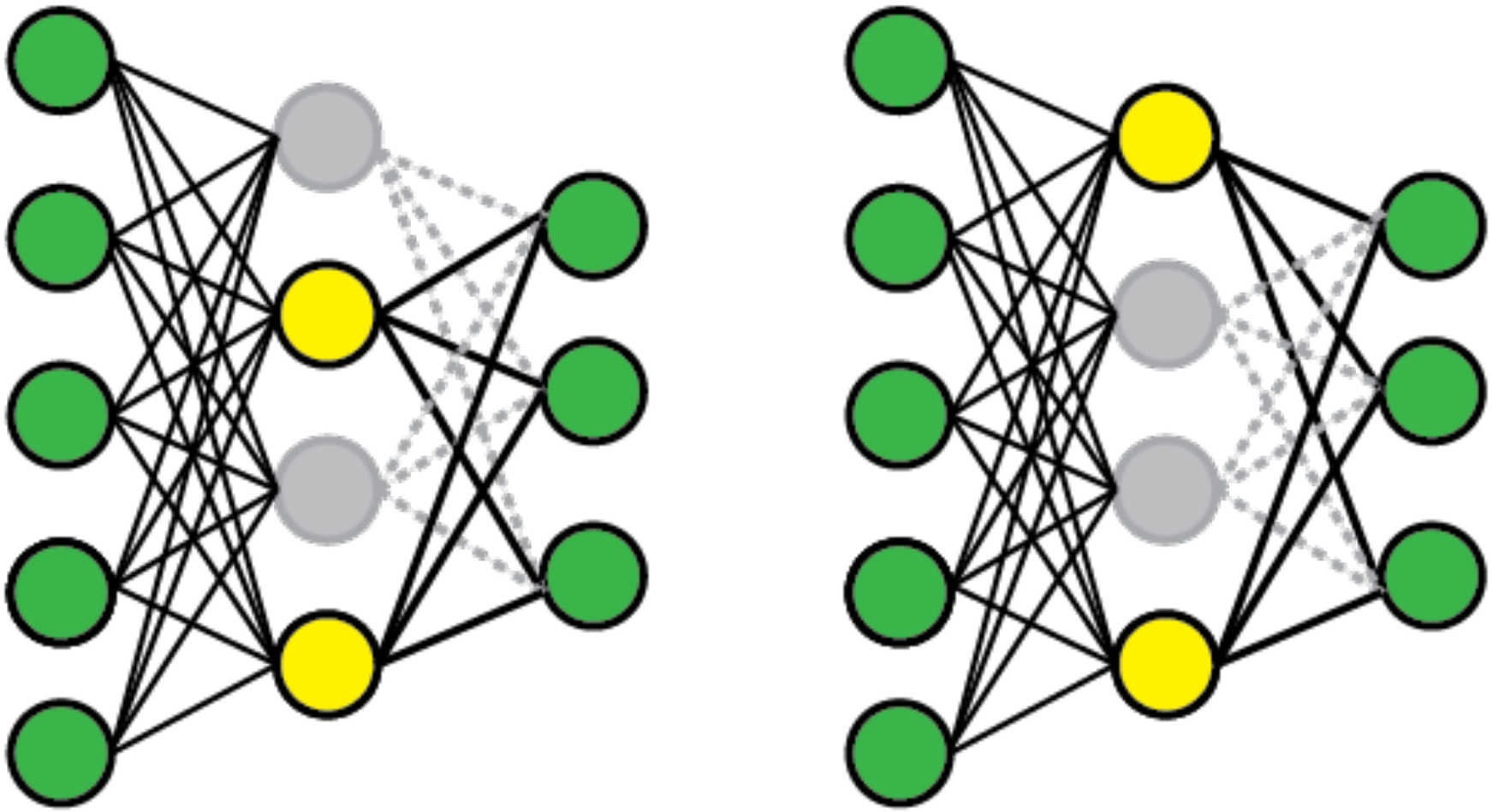
Baseline L1 Regularization L2 Regularization

$$\lambda \sum_{j=0}^{M} |W_j| \qquad \lambda \sum_{j=0}^{M} W_j^2$$

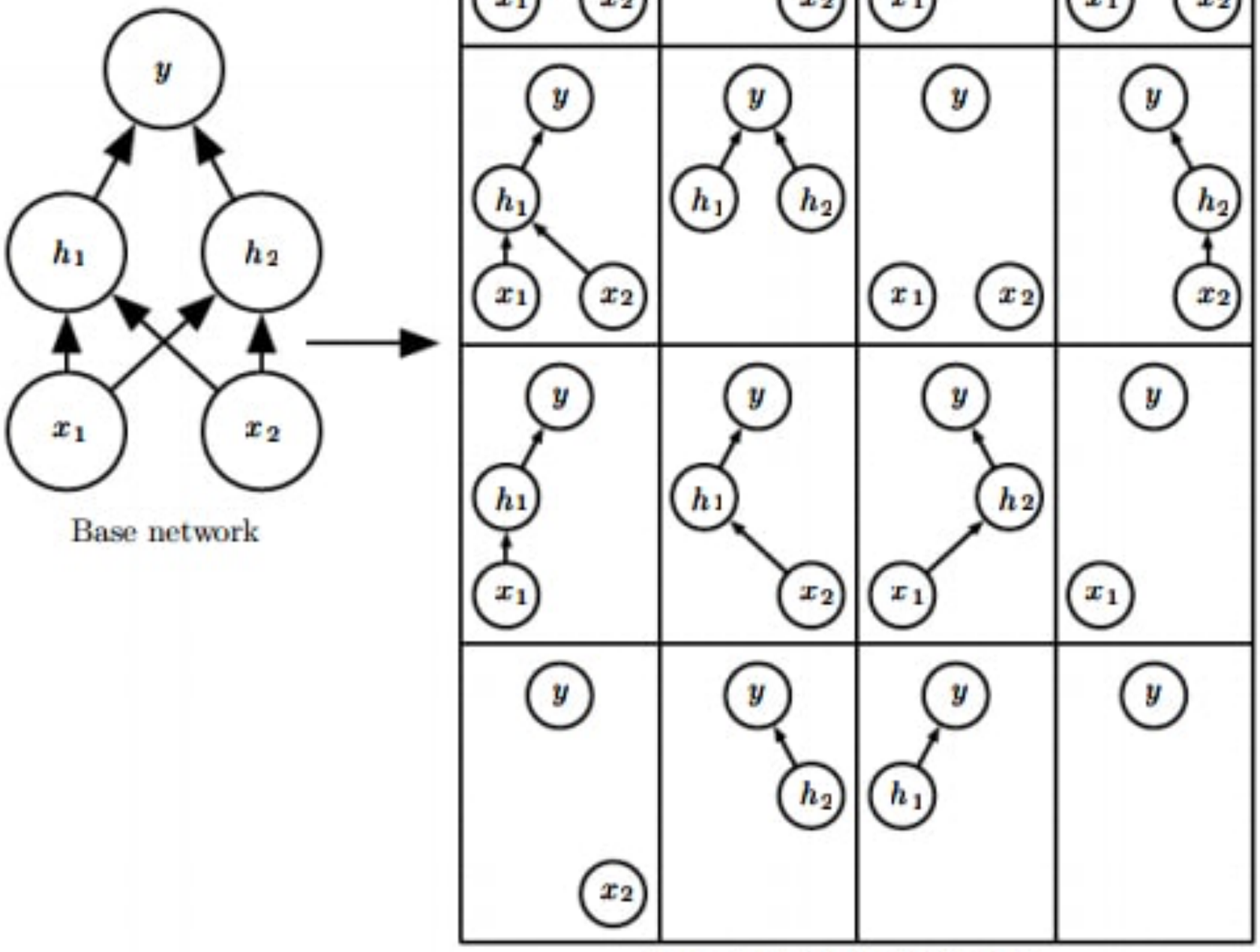
L1 Penalty L2 Penalty

https://hackernoon.com/memorizing-is-not-learning-6-tricks-to-prevent-overfitting-in-machine-learning-820b091dc42

Dropout



https://i.stack.imgur.com/CewjH.png



https://deepestdocs.readthedocs.io/en/latest/004_deep_learning_part_2/image/0042_fig0.jpg

Vanishing Gradient

Overfitting

Deep Learning for Image Classification

[기업 연계를 위한 AI-IoT 과정] AI | 세번째 날 | 전미정

Classification

CAT

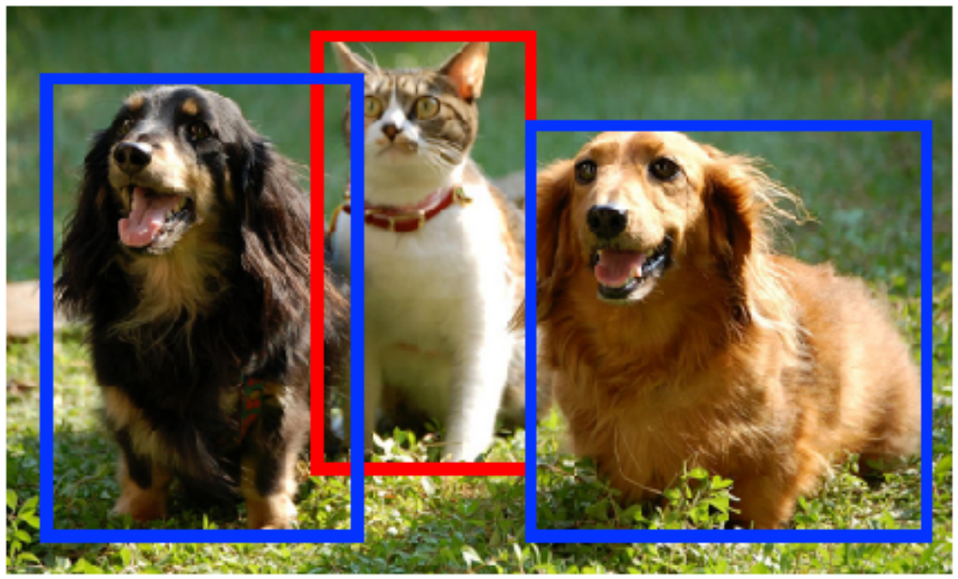
Classification
+ Localization

CAT

Single object

https://www.oreilly.com/library/view/python-advanced-guide/9781789957211/c9adbb35-809e-4667-af5f-d5a58095c828.xhtml

Object Detection



CAT, DOG

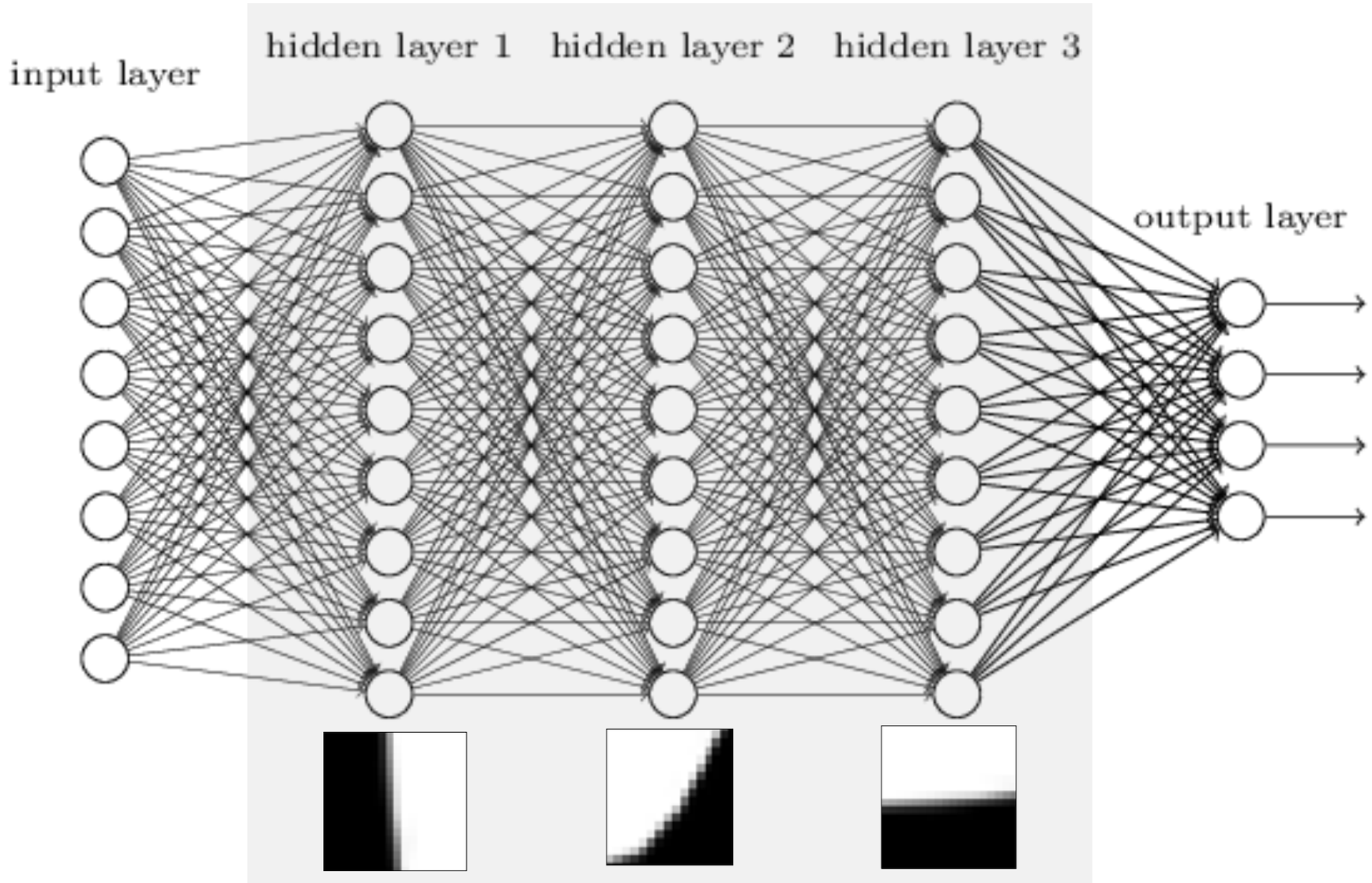
Instance Segmentation

CAT, DOG

Multiple objects

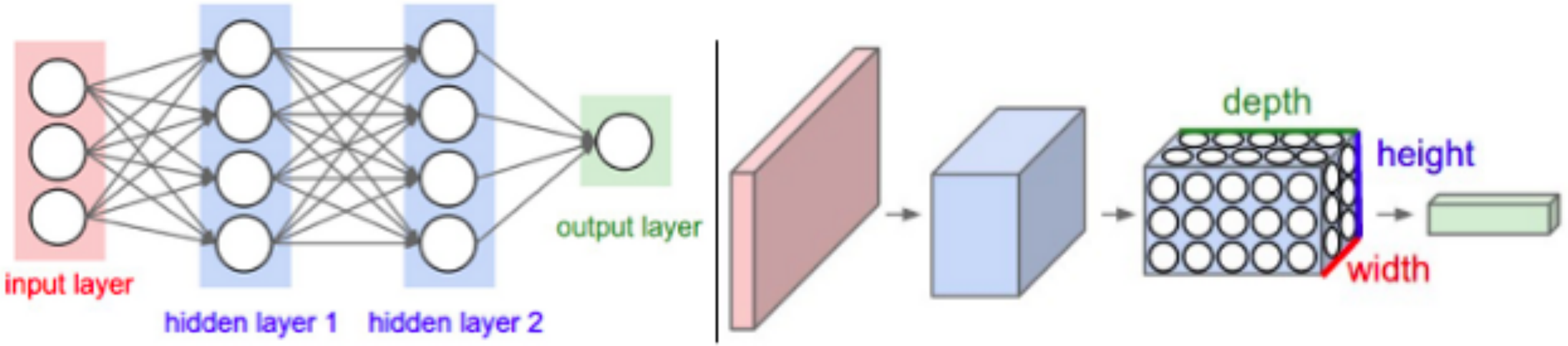
https://www.oreilly.com/library/view/python-advanced-guide/9781789957211/c9adbb35-809e-4667-af5f-d5a58095c828.xhtml

Deep neural network



http://neuralnetworksanddeeplearning.com/chap5.html

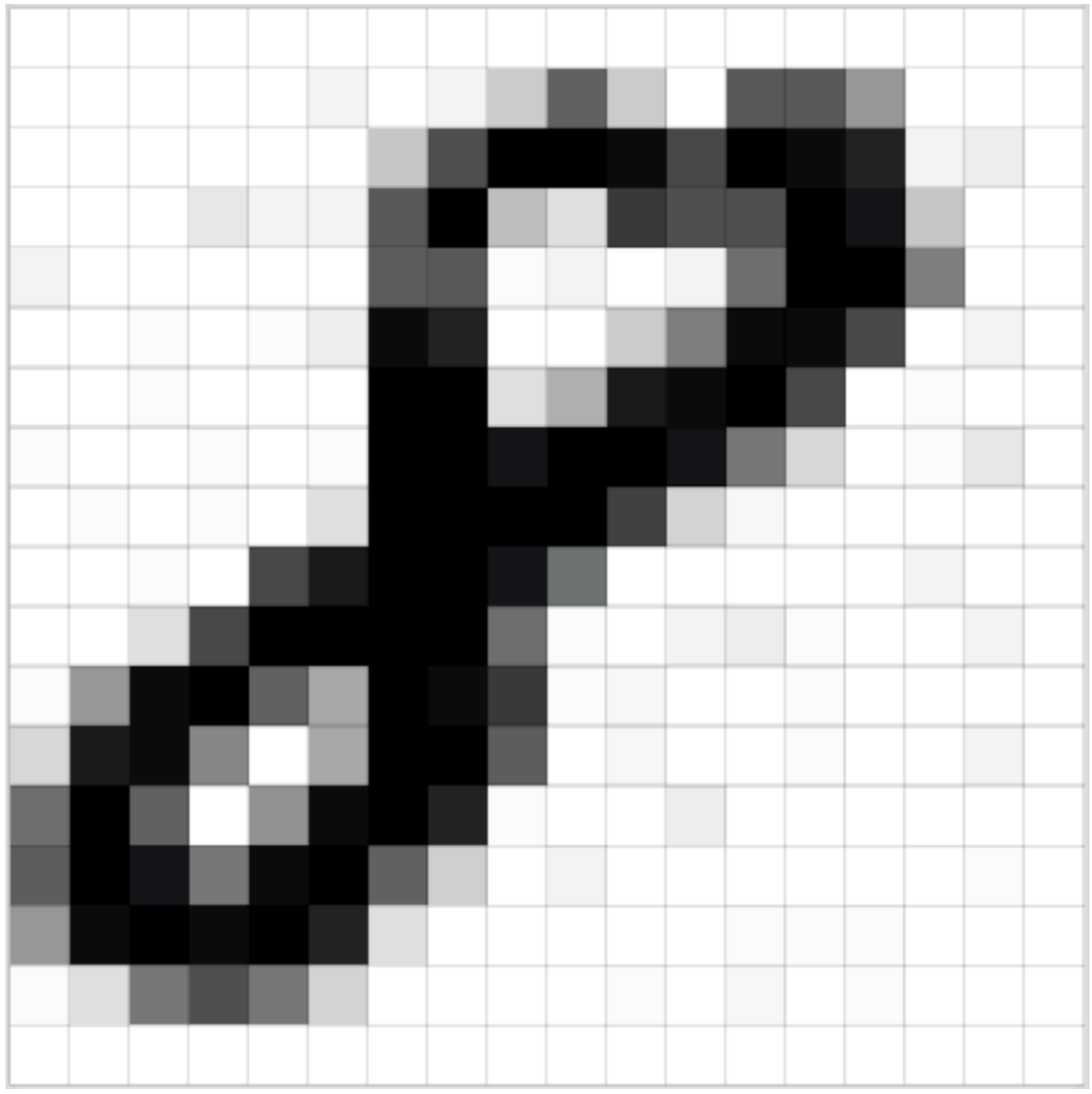
Neural Networks



DNN

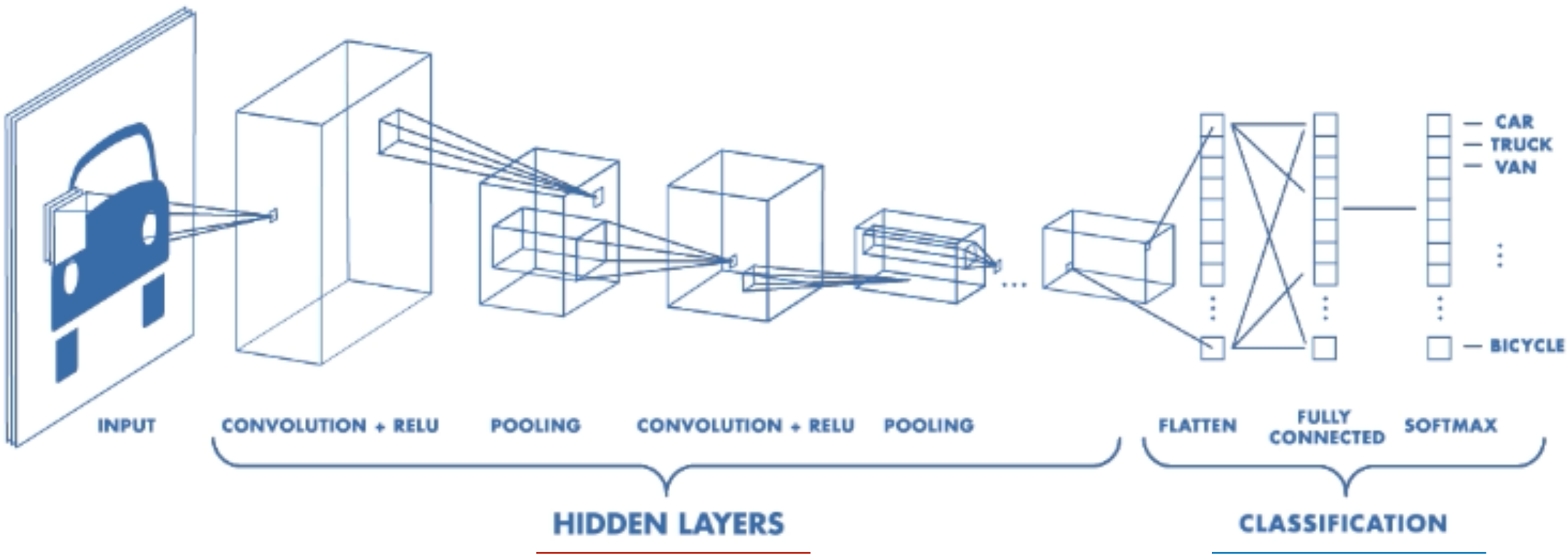
CNN

http://cs231n.github.io/convolutional-networks/



https://medium.com/@ageitgey

Convolutional Neural Networks



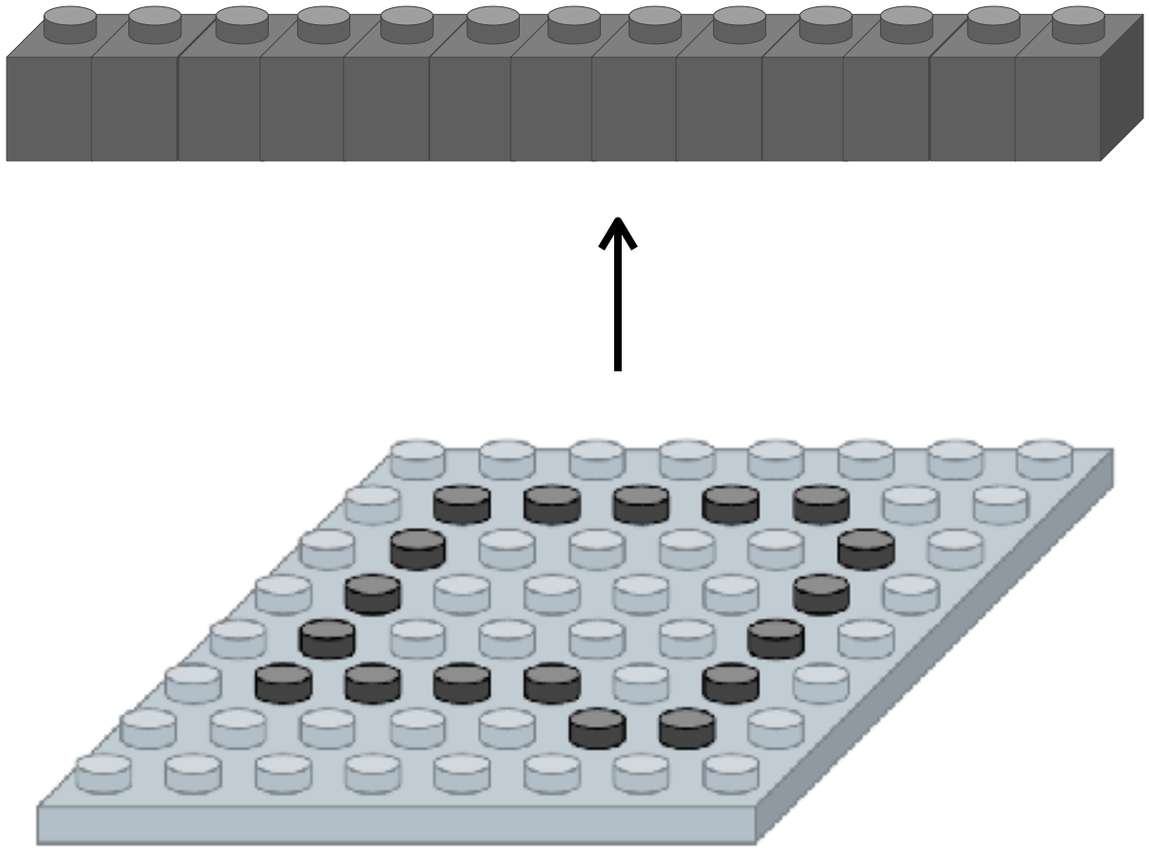
https://www.mathworks.com/videos/introduction-to-deep-learning-what-are-convolutional-neural-networks--1489512765771.html

Cifar 10


airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck

목표 모델

이미지 -> 10 분류

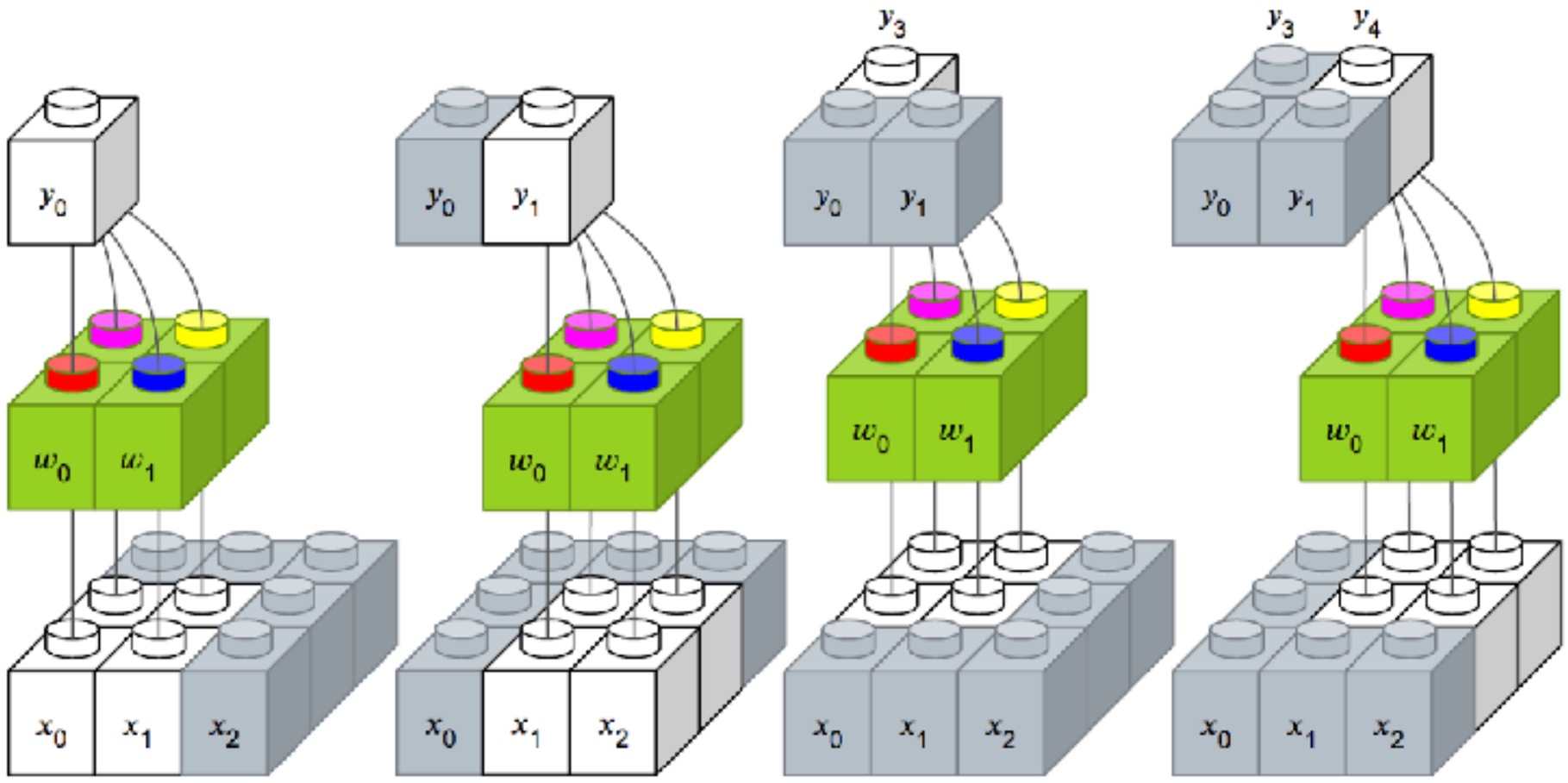


김태영의 케라스 블로그*: https://tykimos.github.io/*

CNN Layers

Convolution Layer

이미지 특징 학습 필터



김태영의 케라스 블로그*: https://tykimos.github.io/*

Convolution Layer

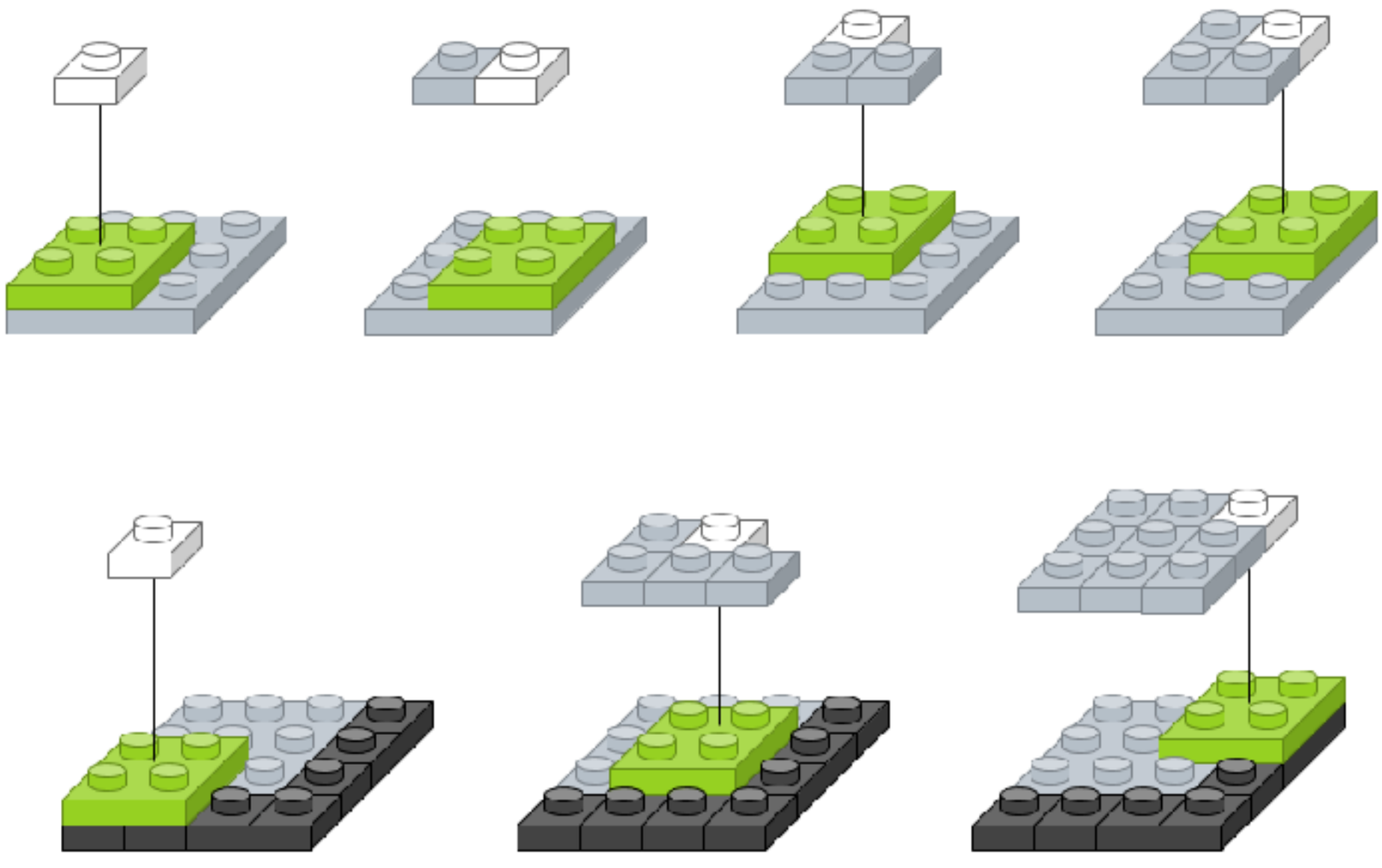
1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

4		

https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

Convolution Layer

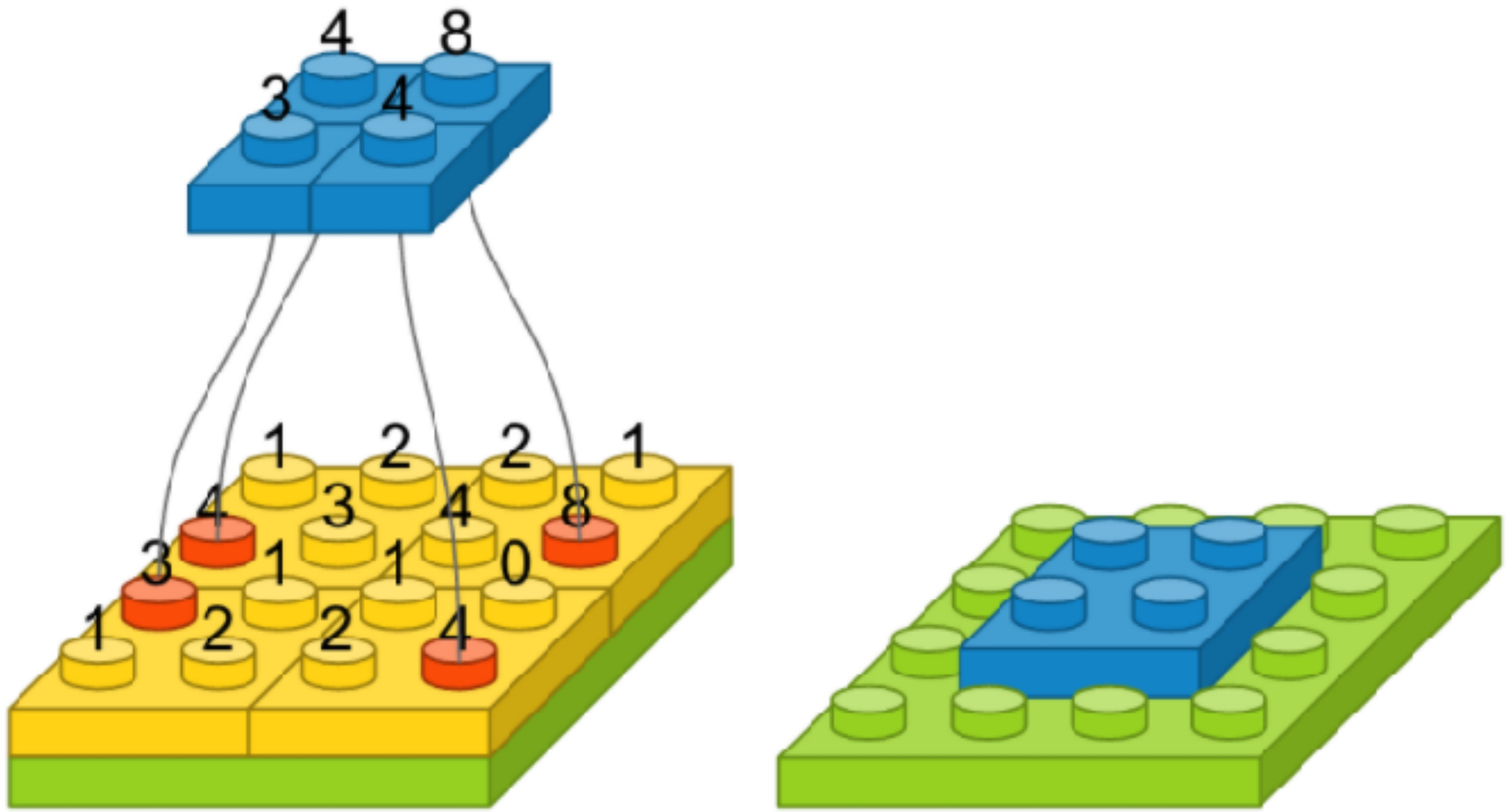
경계 처리 방법(padding)



김태영의 케라스 블로그*: https://tykimos.github.io/*

MaxPooling Layer

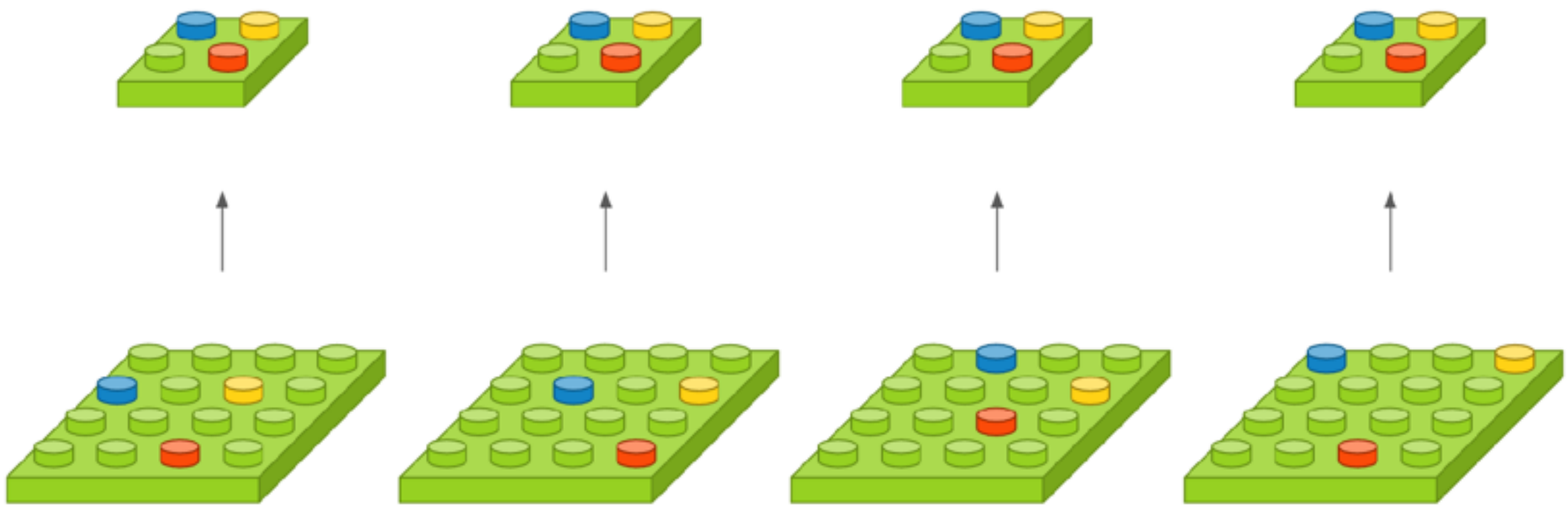
이미지의 큰 특징만 추출



김태영의 케라스 블로그*: https://tykimos.github.io/*

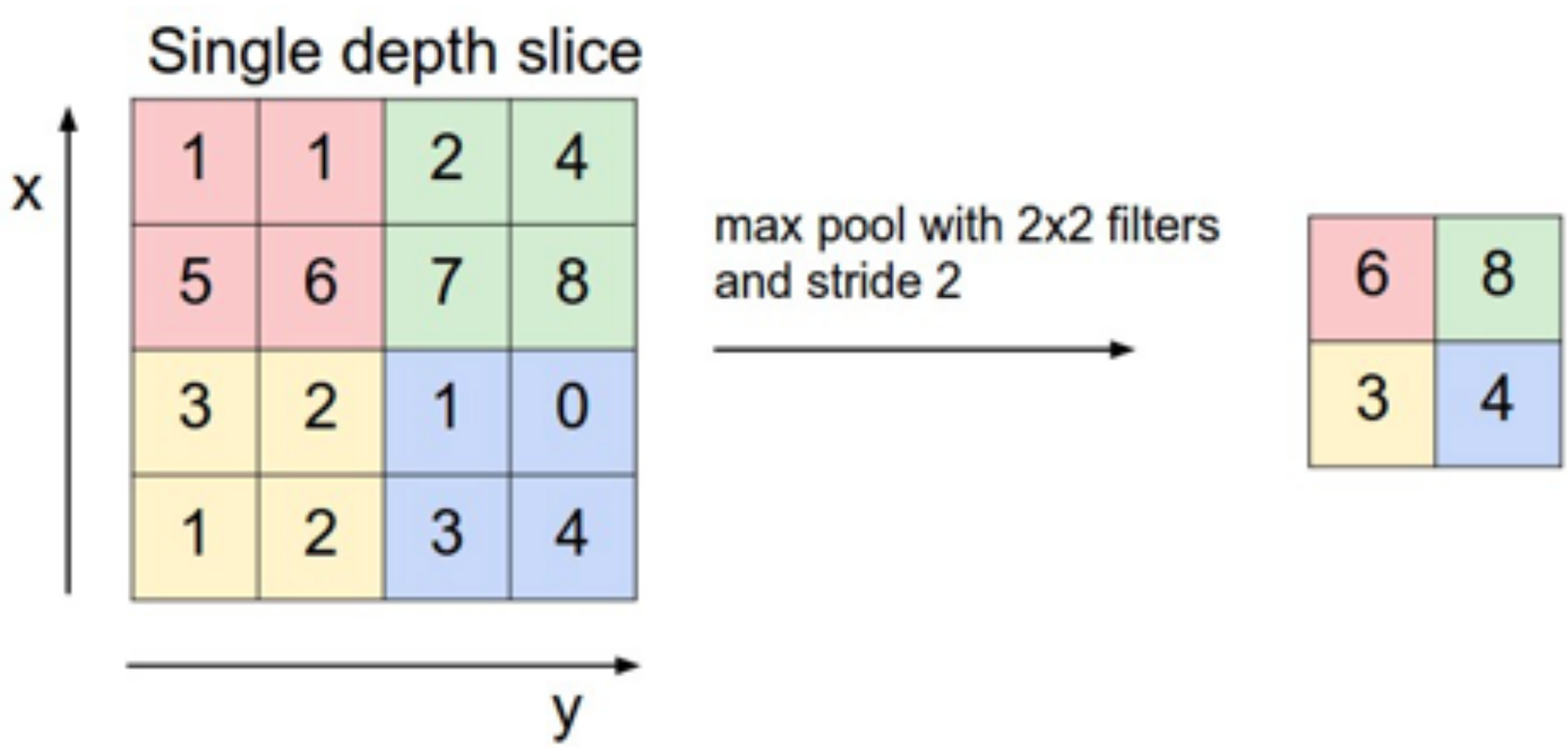
MaxPooling Layer

이미지 사이즈 축소, 특징 일반화



김태영의 케라스 블로그*: https://tykimos.github.io/*

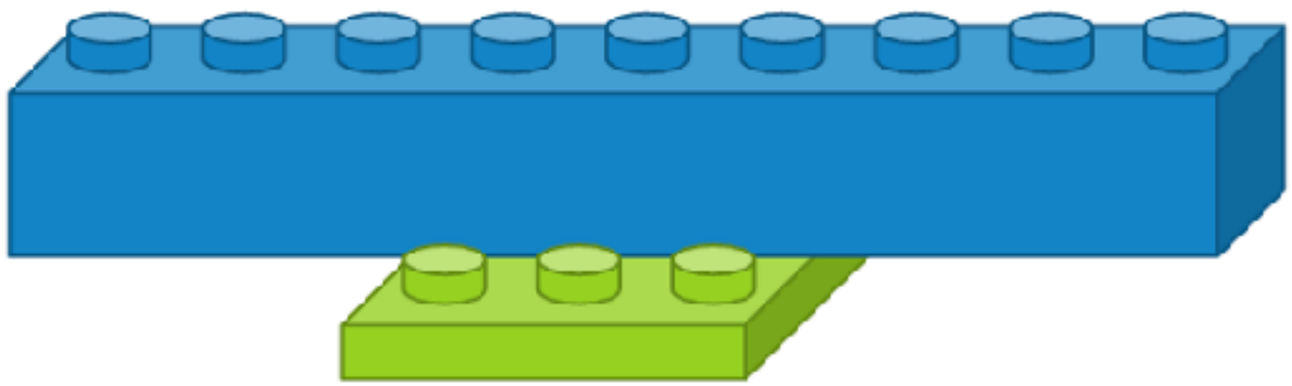
MaxPooling Layer



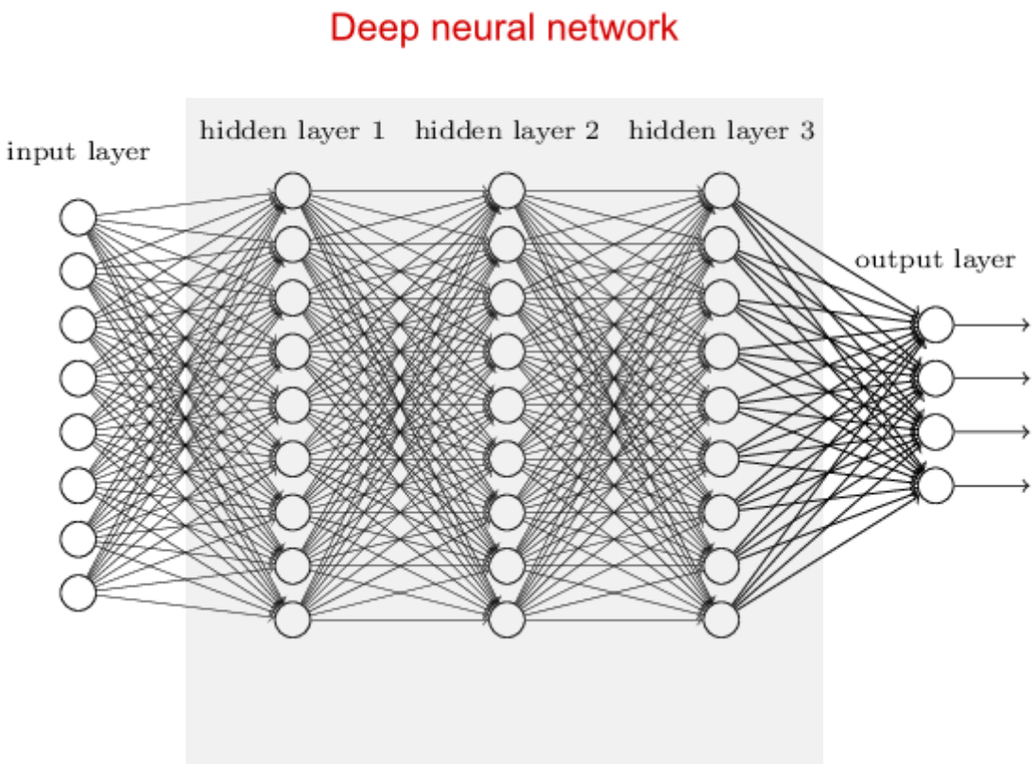
http://cs231n.github.io/convolutional-networks/

Flatten Layer

영상(2차원) -> 문자열(1차원)

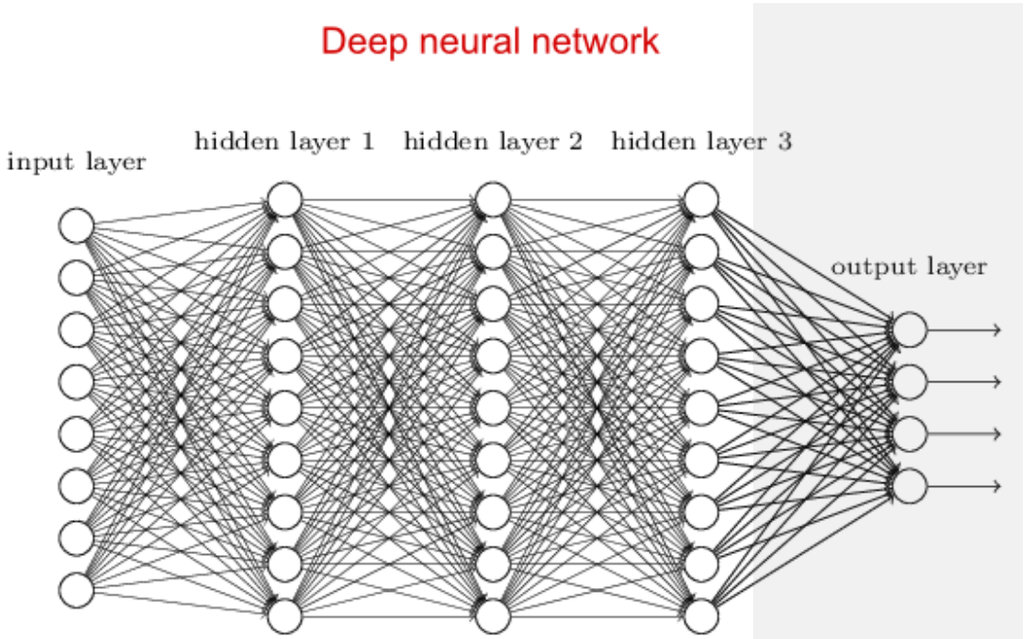


Activation Layer



은닉층		
ReLU	**Leaky ReLu**	**ELU**
Rectifier linear	Leaky ReLU	Exponential Linear Units

Activation Layer



출력층		
linear	**sigmoid**	**softmax**
특정 값 예측	이진 클래스 예측	다중 클래스 예측

세 번째 날

- 딥러닝 이론
- 이미지 분류
- Keras를 활용한 이미지 분류 모델 생성

문제 정의

데이터 셋 준비

모델 설정

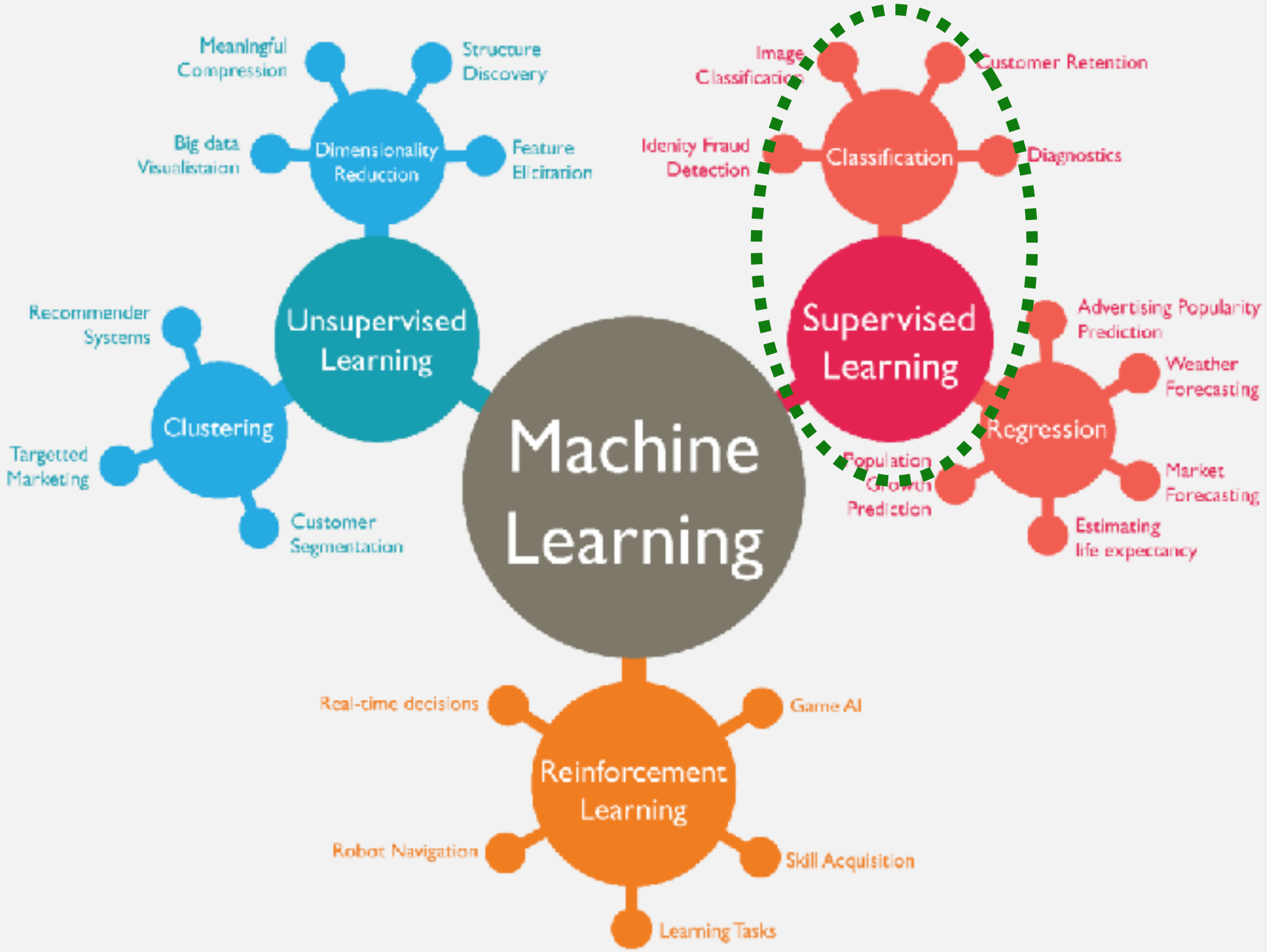
모델 훈련 / 평가

모델 활용

MACTORES

Data Science Team

Machine Learning Bubble Chart



https://www.mactores.com

Cifar 10

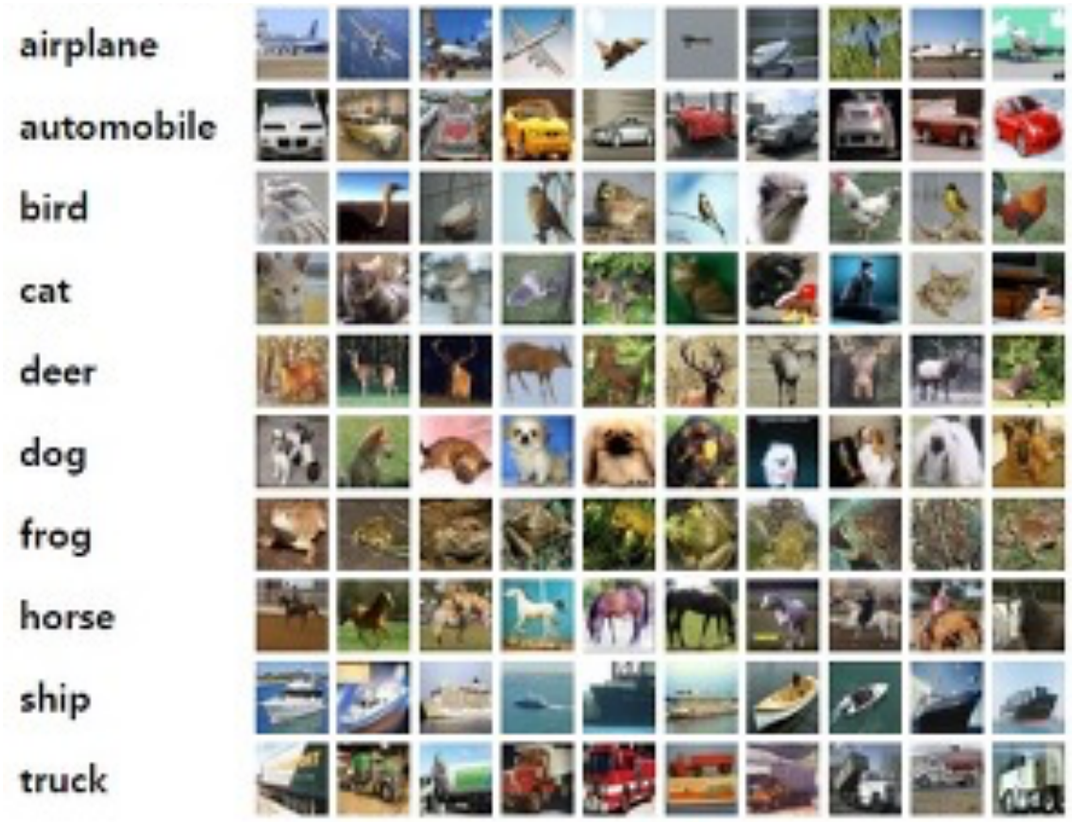


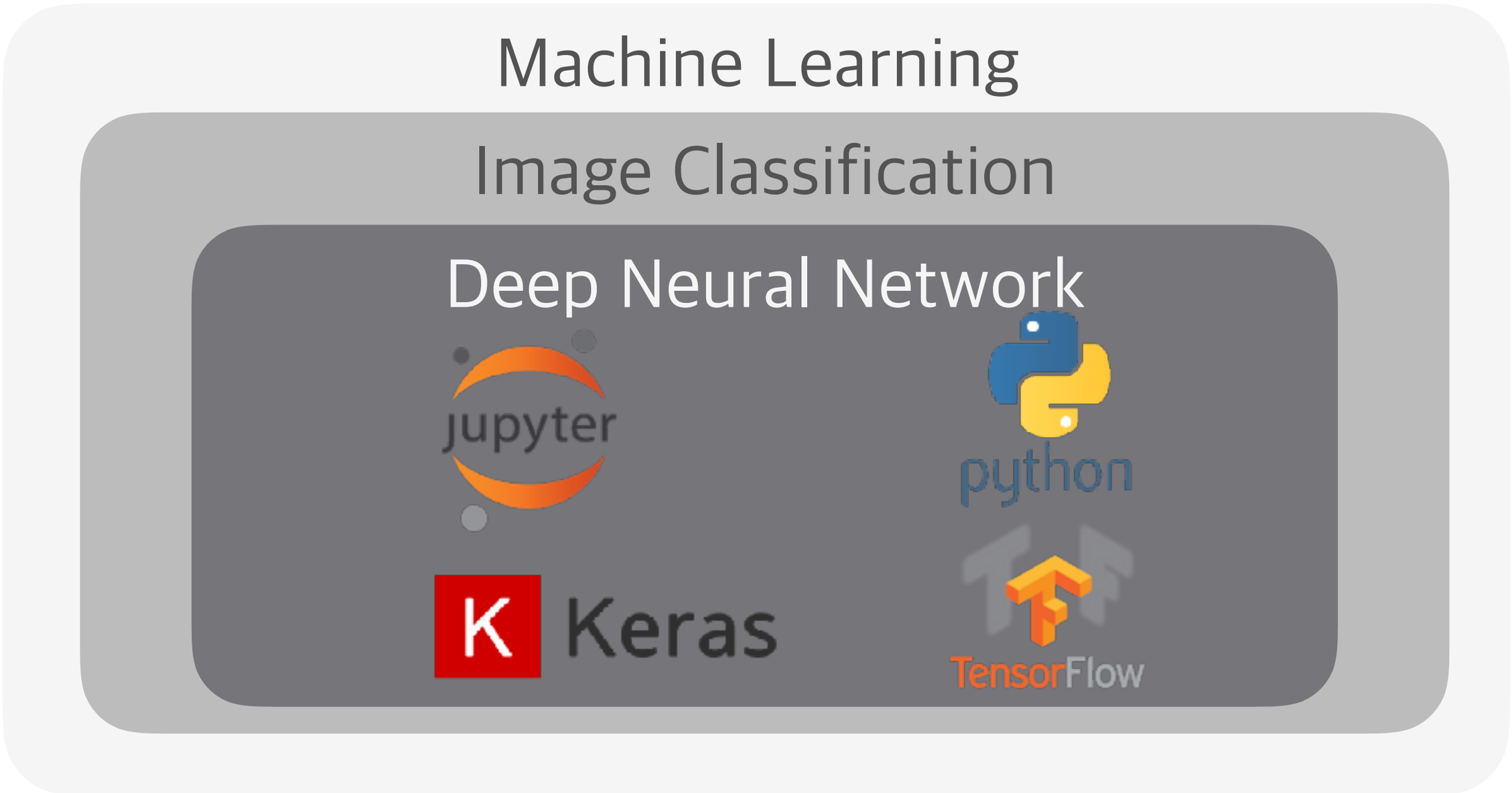
Image Dataset

32 x 32

6000 장

10 classes

- Linear Regression
- Regression Trees
- SVM(Support Vector Machine)
- Decision Tree
- Ensemble
- Bayes Classification
- Logistic Regression
- SOM(Self-Organizing Map)
- k-Means Clustering
- Sequence Analysis
- Link Analysis
- Text mining
- Neural Network
- .....


Machine Learning
Image Classification
Deep Neural Network
jupyter
python
Keras
TensorFlow

Frameworks

Caffe2

Chainer

Cognitive Toolkit

mxnet

PyTorch

PaddlePaddle

ML.NET

MathWorks

dmlc XGBoost

ML

scikit learn

K

κέρας **= 그리스어 “뿔”**


상아
거짓
뿔
진실


우리가 만든 모델이 진실을 말해주길...

keras.io

Keras Documentation

Search docs

- Home
- Keras: The Python Deep Learning library
- You have just found Keras.
- Guiding principles
- Getting started: 30 seconds to Keras
- Installation
- Switching from TensorFlow to CNTK or Theano
- Support
- Why this name, Keras?
- Getting started
- Guide to the Sequential model
- Guide to the Functional API
- FAQ
- Models

Docs » Home

Keras: The Python Deep Learning library

You have just found Keras.

Keras is a high-level neural networks API, written in Python and capable of running Theano. It was developed with a focus on enabling fast experimentation. *Being able least possible delay is key to doing good research.*

Use Keras if you need a deep learning library that:

- Allows for easy and fast prototyping (through user friendliness, modularity, and extensibilit
- Supports both convolutional networks and recurrent networks, as well as combinations of t
- Runs seamlessly on CPU and GPU.

Read the documentation at Keras.io.

Keras is compatible with: **Python 2.7-3.6.**

You have just found Keras

[기업 연계를 위한 AI-IoT 과정] AI | 세번째 날 | 전미정

TensorFlow

```
############################
###DNN in TensorFlow Only###
############################

import tensorflow as tf
from tensorflow.examples.tutorials.mnist import input_data

###1. Load data set, and split it if necessary
mnist = input_data.read_data_sets("MNIST_data/")

###2. we create a holder, a container to place the computation activities
in tensorflow
###identifying format and tensor's r/c, null means any kind
VISIBLE_NODES = 784
HIDDEN_NODES = 400
x = tf.placeholder("float", shape=[None, VISIBLE_NODES])
y = tf.placeholder("float", shape=[None, 10])

###3. We identify weights and biases with tensor shape, start with 0
weights = tf.Variable(tf.random_normal((VISIBLE_NODES, HIDDEN_NODES),
    mean=0.0, stddev=1. / VISIBLE_NODES))
hidden_bias = tf.Variable(tf.zeros([HIDDEN_NODES]))
visible_bias = tf.Variable(tf.zeros([VISIBLE_NODES]))

###4. set up the sigmoid model and multiply x and W with matmul function,
building the
###hidden layer and reconstruction layer
hidden_activation = tf.nn.sigmoid(tf.matmul(x, weights) + hidden_bias)
visible_reconstruction = tf.nn.sigmoid(tf.matmul(hidden_activation,
tf.transpose(weights))
    + visible_bias)
final_hidden_activation = tf.nn.sigmoid(tf.matmul(visible_reconstruction,
weights)
    + hidden_bias)

###5. This process can be understood as being two phases of learning
###positive and negative or, more poetically, waking and sleeping
positive_phase = tf.matmul(tf.transpose(x), hidden_activation)
negative_phase = tf.matmul(tf.transpose(visible_reconstruction),
final_hidden_activation)
LEARNING_RATE = 0.01
weight_update = weights.assign_add(LEARNING_RATE *
    (positive_phase - negative_phase))
visible_bias_update = visible_bias.assign_add(LEARNING_RATE *
    tf.reduce_mean(x - visible_reconstruction, 0))
hidden_bias_update = hidden_bias.assign_add(LEARNING_RATE *
    tf.reduce_mean(hidden_activation - final_hidden_activation, 0))
###6. Now we create the operations for scaling the hidden and visible
biases, with loss
###function feedback
train_op = tf.group(weight_update, visible_bias_update,
hidden_bias_update)
loss_op = tf.reduce_sum(tf.square(x - visible_reconstruction))

###7. We start the session
session = tf.Session()
session.run(tf.global_variables_initializer())
current_epochs = 0

###8.Run the session
for i in range(20):
    total_loss = 0
    while mnist.train.epochs_completed == current_epochs:
        batch_inputs, batch_labels = mnist.train.next_batch(100)
        _, reconstruction_loss = session.run([train_op, loss_op],
feed_dict={input_placeholder: batch_inputs})
        total_loss += reconstruction_loss

    print("epochs %s loss %s" % (current_epochs, reconstruction_
    current_epochs = mnist.train.epochs_completed
```

VS

Keras

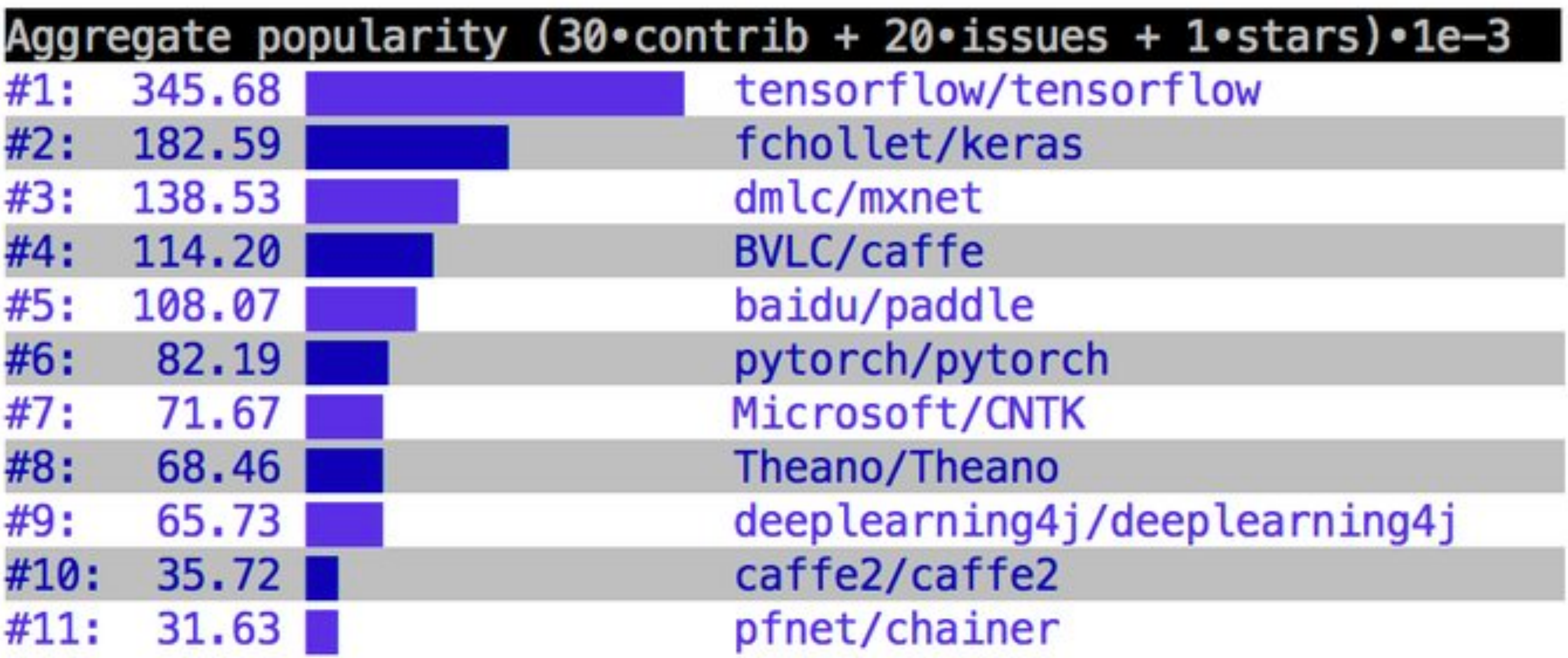
```
##############################
###DNN in TensorFlow KeRas ###
##############################
###1. Load Data and Splot Data
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers.core import Dense, Activation
from keras.utils import np_utils

(X_train, Y_train), (X_test, Y_test) = mnist.load_data()

###2.Preprocess
X_train = X_train.reshape(60000, 784)
X_test = X_test.reshape(10000, 784)
X_train = X_train.astype('float32')
X_test = X_test.astype('float32')
X_train /= 255
X_test /= 255
classes = 10
Y_train = np_utils.to_categorical(Y_train, classes)
Y_test = np_utils.to_categorical(Y_test, classes)

###3. Set up parameters
input_size = 784
batch_size = 100
hidden_neurons = 400
epochs = 30

###4.Build the model
model = Sequential()
model.add(Dense(hidden_neurons, input_dim=input_size))
model.add(Activation('relu'))
model.add(Dense(classes, input_dim=hidden_neurons))
model.add(Activation('softmax'))

model.compile(loss='categorical_crossentropy',
    metrics=['accuracy'], optimizer='adadelta')
model.fit(X_train, Y_train, batch_size=batch_size, epochs=epochs,
verbose=1)

###5.Test
score = model.evaluate(X_test, Y_test, verbose=1)
print('\n''Test accuracy:', score[1])
#Test accuracy: 0.983
```

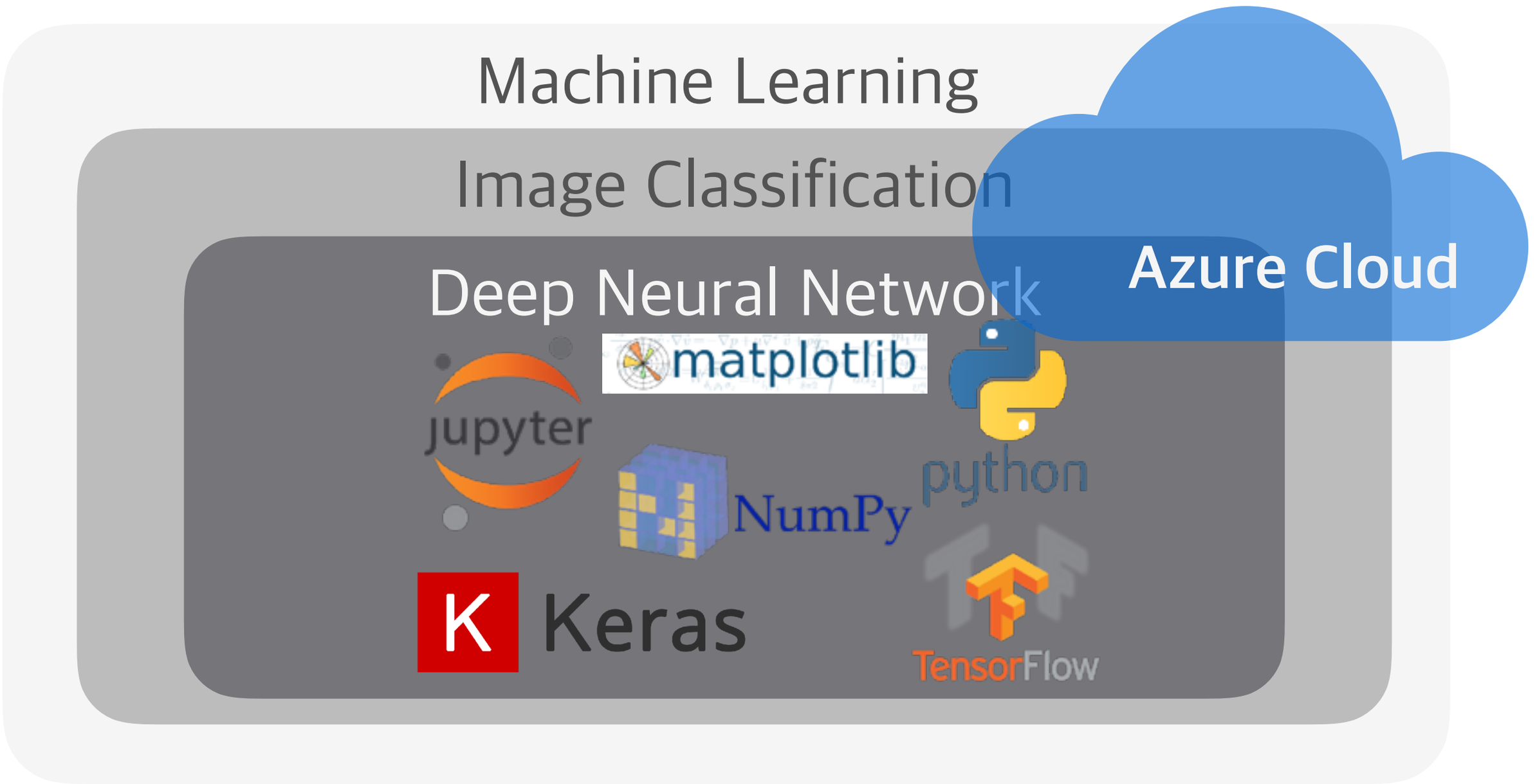
Library	Rank	Overall	Github	Stack Overflow	Google Results
tensorflow	1	10.87	4.25	4.37	2.24
keras	2	1.93	0.61	0.83	0.48
caffe	3	1.86	1.00	0.30	0.55
theano	4	0.76	-0.16	0.36	0.55
pytorch	5	0.48	-0.20	-0.30	0.98
sonnet	6	0.43	-0.33	-0.36	1.12
mxnet	7	0.10	0.12	-0.31	0.28
torch	8	0.01	-0.15	-0.01	0.17
cntk	9	-0.02	0.10	-0.28	0.17
dlib	10	-0.60	-0.40	-0.22	0.02
caffe2	11	-0.67	-0.27	-0.36	-0.04

François Chollet @fchollet

And here are the GitHub popularity metrics at this time.
pic.twitter.com/o4eoGTNdl0

♡ 167 2:02 PM - Mar 9, 2018




Machine Learning
Image Classification
Deep Neural Network
Azure Cloud
matplotlib
jupyter
NumPy
python
Keras
TensorFlow

Azure


SQL

Resource

Resource Group

Subscription

Resource

Region

[기업 연계를 위한 AI-IoT 과정] AI | 세번째 날 | 정미정

https://azure.microsoft.com/en-us/global-infrastructure/regions/

[https://azure.microsoft.com/en-us/global-infrastructure/regions/

날 | 전미정





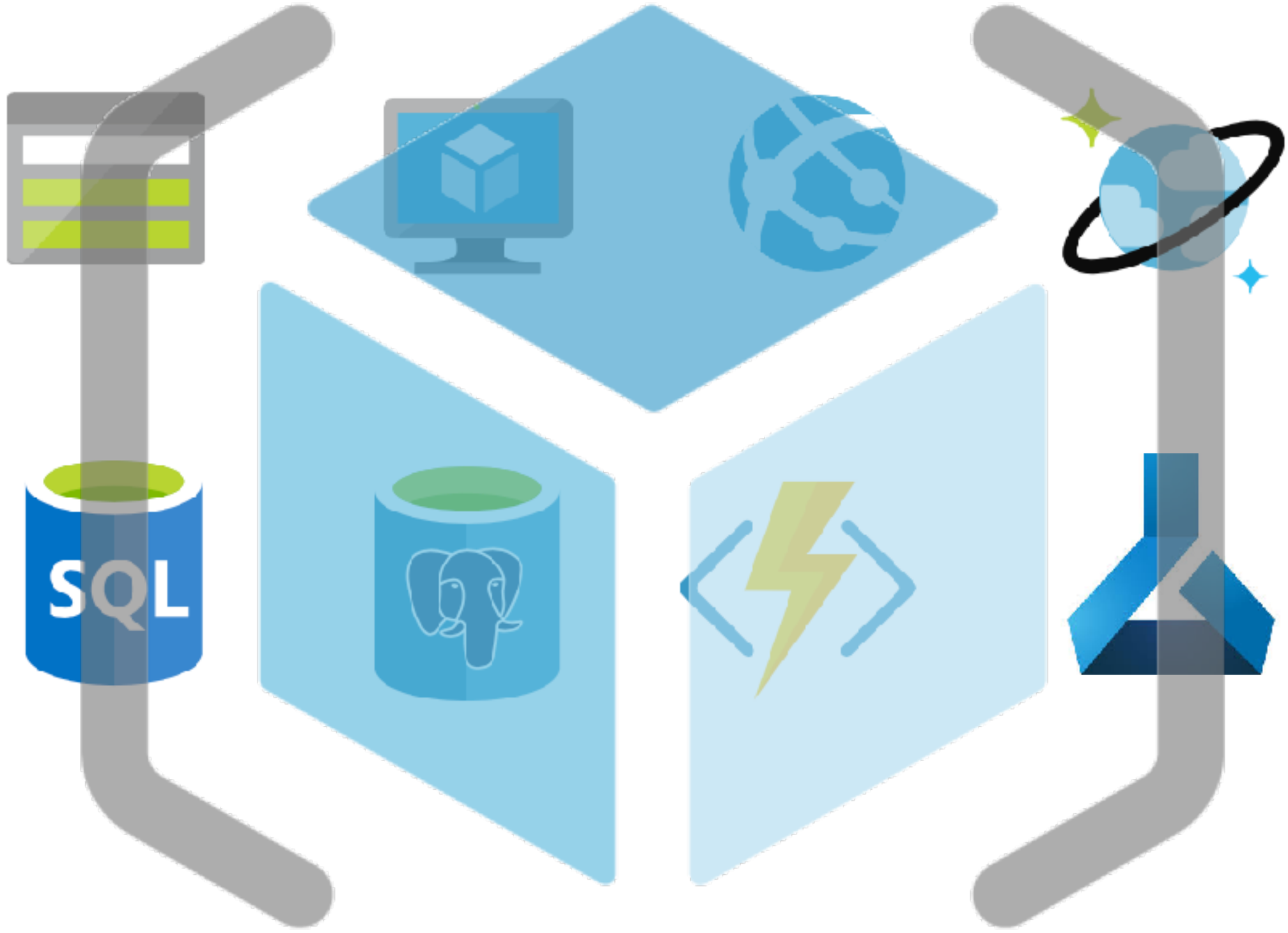


Region US East

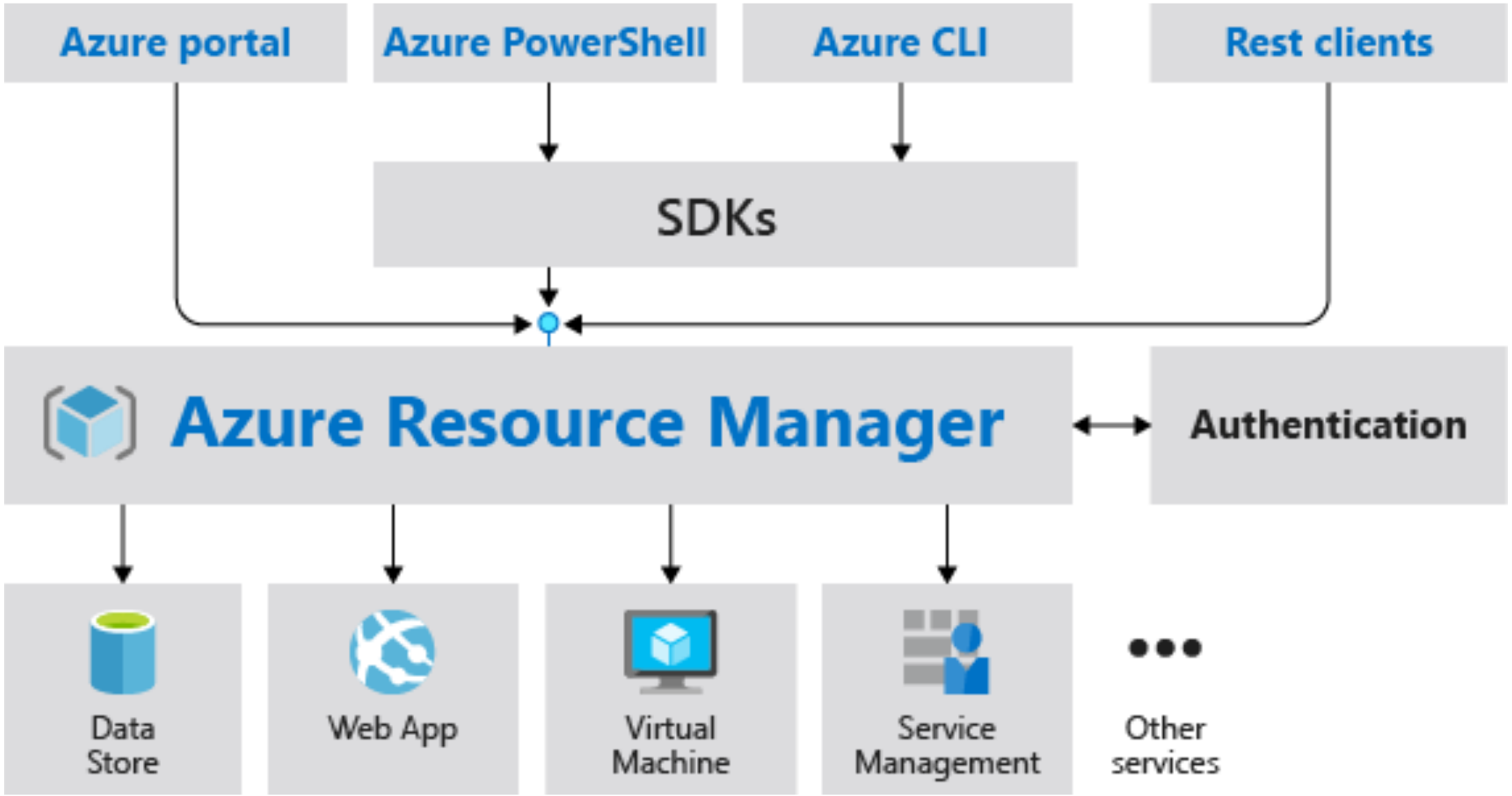
Resource

Resource Group

Subscription



Region US East


Azure portal
Azure PowerShell
Azure CLI
Rest clients
SDKs
Azure Resource Manager
Authentication
Data Store
Web App
Virtual Machine
Service Management
Other services

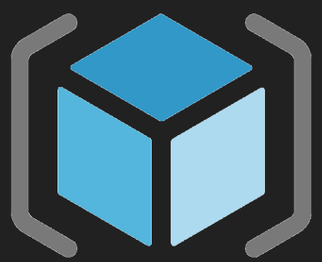
Resource

Resource Group

Subscription

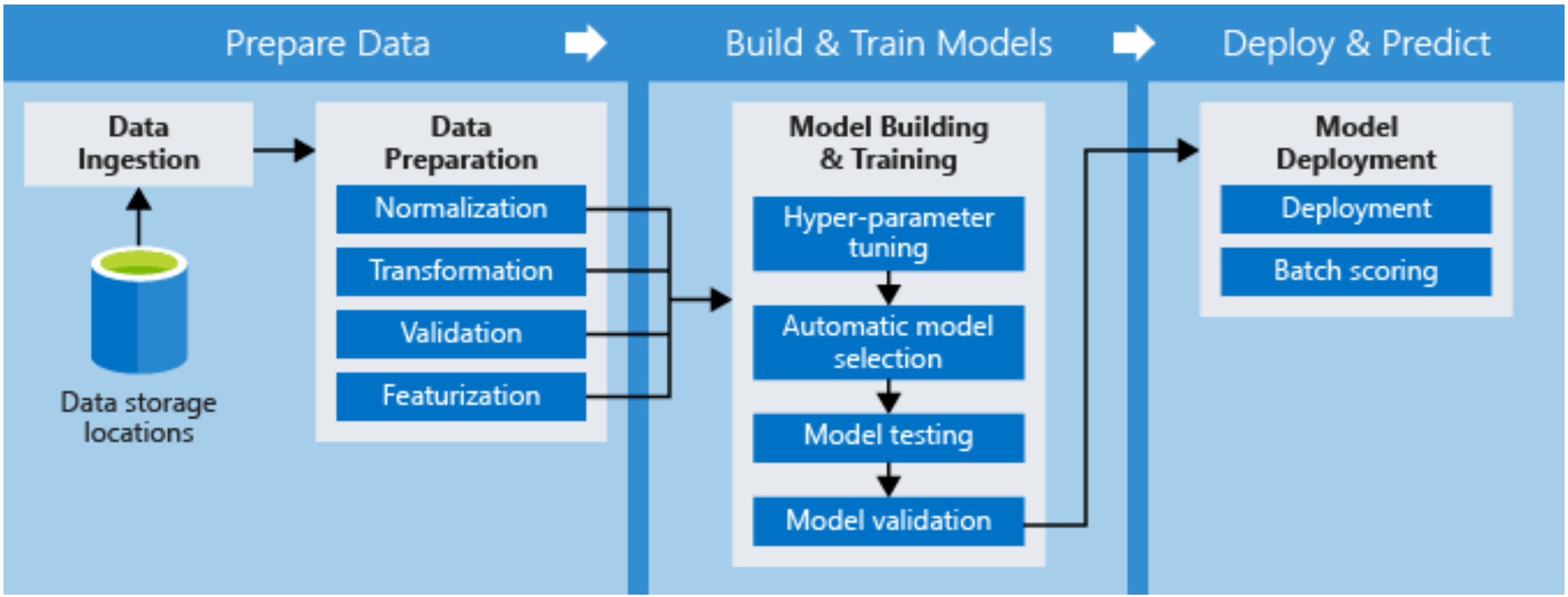
Pay As You Go

Resource Machine Learning service



Resource Group AutoML_DLCAT

Subscription Microsoft Sponsorship



rning/service/media/concept-ml-pipelines/pipelines.png

네 번째 날

다섯 번째 날

AI on Android

Webs for more ML

- 캐글 데이터셋
- UCI 데이터셋
- Azure 갤러리
- Azure ML Studio에서 R 사용하기 실습
- Azure Machine Learning Service
- Azure Notebook
- Azure Cognitive Sevice

Books for more ML

입문자가 읽기 좋은 **머신러닝** 책

- 핸즈온 머신러닝_오렐리앙 제롱(한빛미디어)
- 처음 배우는 머신러닝_김승연,정용주(한빛미디어)
- 모두의 딥러닝_조태호(길벗)

그리고 **딥러닝**에 대해 더 공부하고 싶다면

- 블록과 함께하는 파이썬 딥러닝 케라스 이야기_김태영(디지털북스)
- 케라스 창시자에게 배우는 딥러닝_프랑소와 숄레(길벗)
- 밑바닥부터 시작하는 딥러닝_사이토 고키(한빛미디어)

수고하셨습니다 

ninevincentg@gmail.com