

다음과 같은 XOR 을 나타내는 Training Data에 대해서 머신러닝의 Logistic Regression 이용하여 구현한 후에, Test Data 이용하여 prediction 수행한 후, 결과를 확인 하시오

x1	x2	t
0	0	0
0	1	1
1	0	1
1	1	0

Training Data

x1	x2
0	0
0	1
1	0
1	1

Test Data



## – Basic Architecture –

박성호 (neowizard2018@gmail.com)

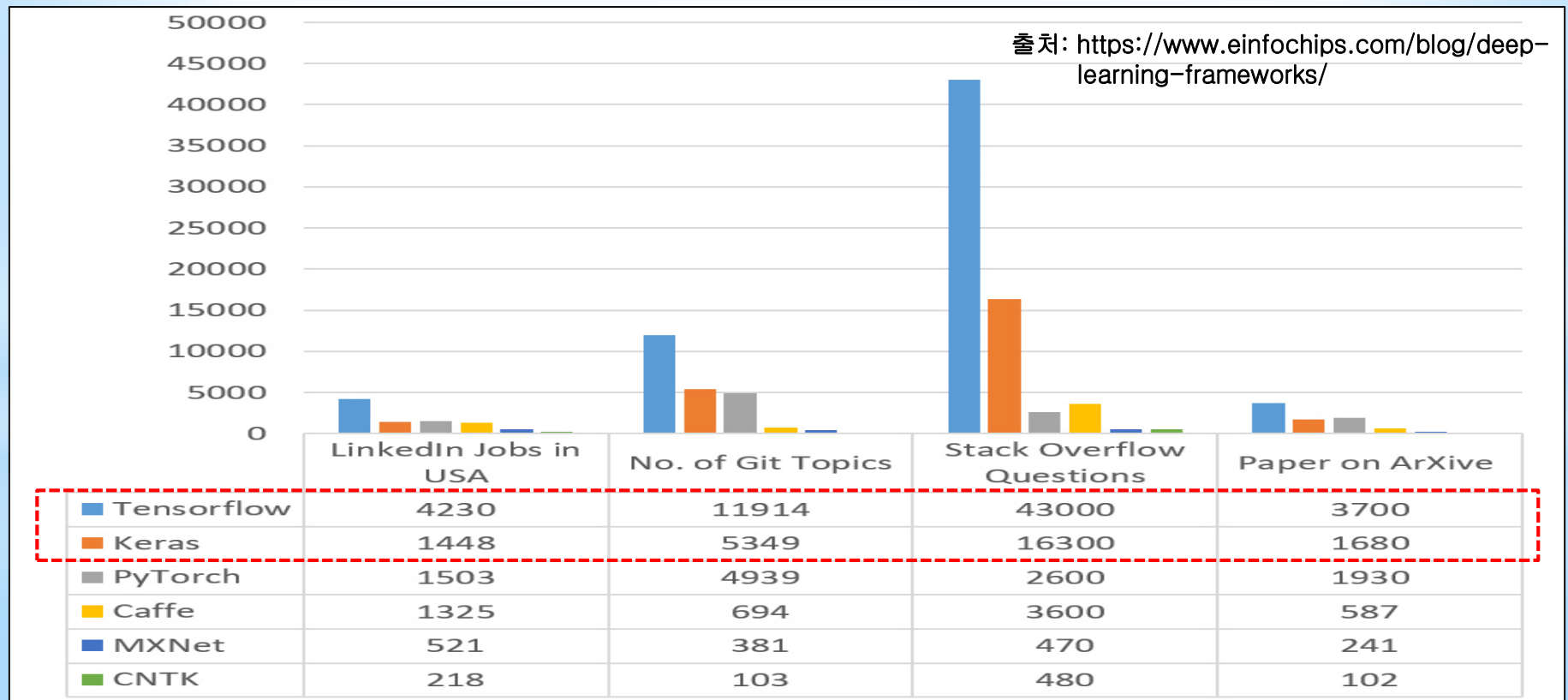
# Overview – TensorFlow 2.x

➤ TensorFlow는 텐서(Tensor)를 흘려보내면서(Flow) 딥러닝 알고리즘을 수행하는 프레임워크.

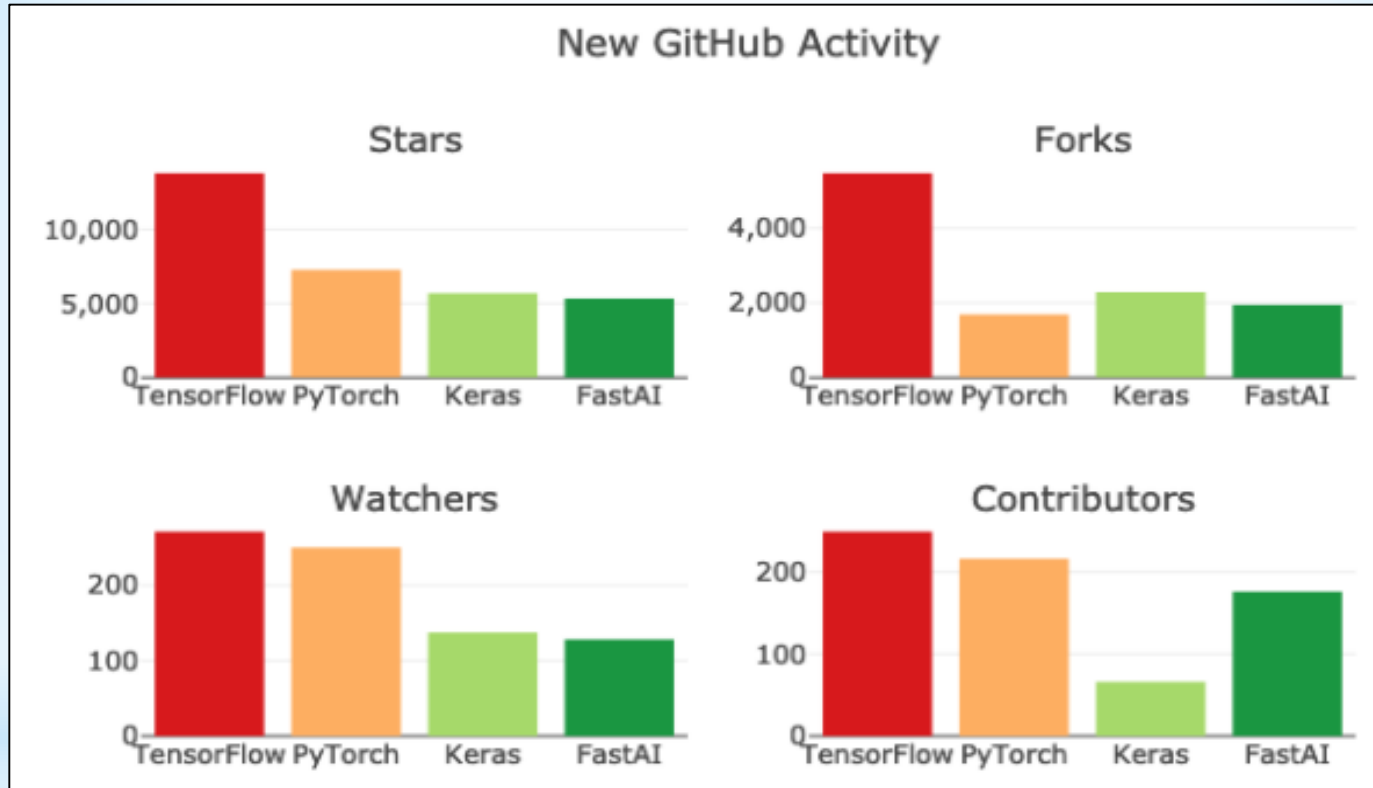
– 2019년 9월 30일에 TensorFlow 2.0 정식 Release 되었으며 1.x 버전과 비교하면

① 사용자 친화적(Keras as High Level API) ② 코드 가독성과 직관성을 높이는 Eager Execution 적용

“If you have started with TensorFlow 2.0 and have never seen TensorFlow 1.x, then you are lucky.” ,  
Deep Learning with TensorFlow 2 and Keras, 2<sup>nd</sup> Edition, Packt, 2020.04



# Overview – GitHub Activity (Developer Trend)



# Overview – Keras vs TensorFlow vs PyTorch



Keras is most suitable for:

- Rapid Prototyping
- Small Dataset
- Multiple back-end support



TensorFlow is most suitable for:

- Large Dataset
- High Performance
- Functionality
- Object Detection



PyTorch is most suitable for:

- Flexibility
- Short Training Duration
- Debugging capabilities

# Keras as High Level API

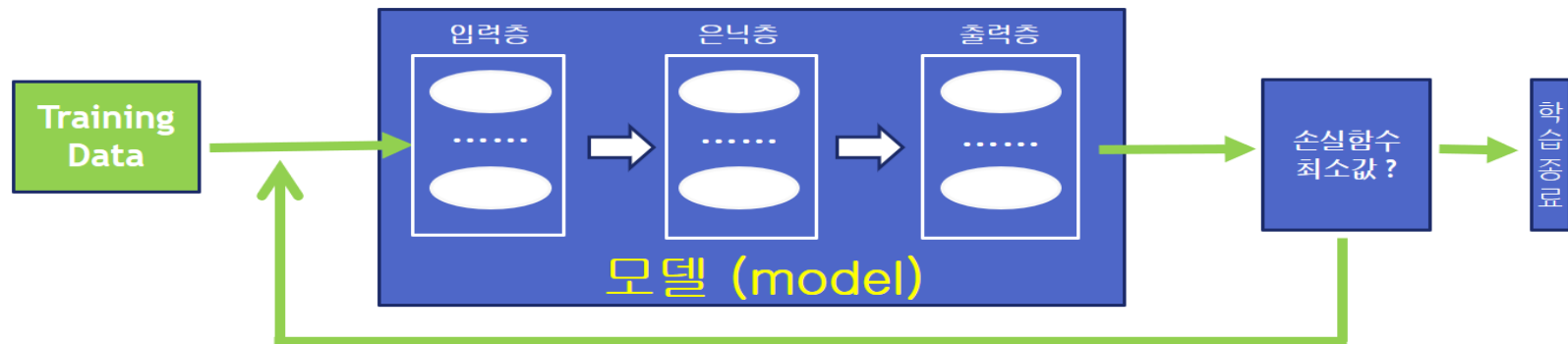
## ➤ Keras in TensorFlow 2.0

- Keras 창시자 프랑소와 솔레(François Chollet)가 TF 2.0 개발에 참여하였고, TF 2.0 에서 공식적이고 유일한 High-Level API 로서 Keras가 선정되었음
- 또한 프랑소와 솔레는 앞으로 native Keras 보다는 `tf.keras`를 사용할 것을 권장하고 있음

## ➤ Keras 특징

- 사용자 친근성 (User Friendliness) : 직관적인 API를 이용하면 ANN, CNN, RNN 또는 이를 조합한 딥러닝 **모델**을 쉽게 구축 할 수 있음
- 모듈성 (Modularity) : Keras에서 제공하는 모듈은 독립적으로 설정 가능함. 즉 신경망 층, 손실함수, 활성화 함수, 최적화 알고리즘, 정규화 기법 등은 모두 독립적인 모듈이기 때문에 이러한 모듈을 서로 조합하기만 하면 새로운 딥러닝 **모델**을 쉽게 만들고 학습시킬 수 있음

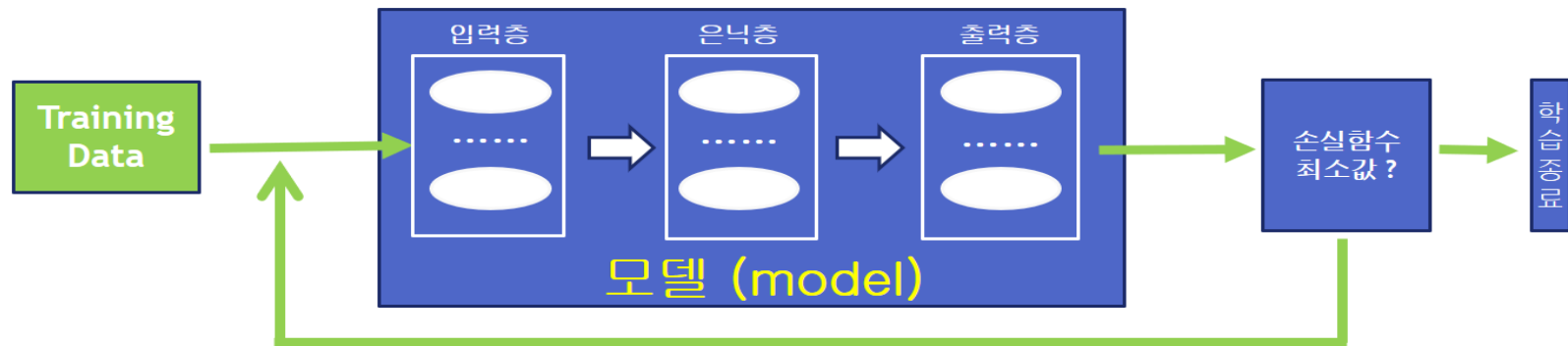
# Keras – model / layer ( <https://youtu.be/Ke70Xxj2EJw> )



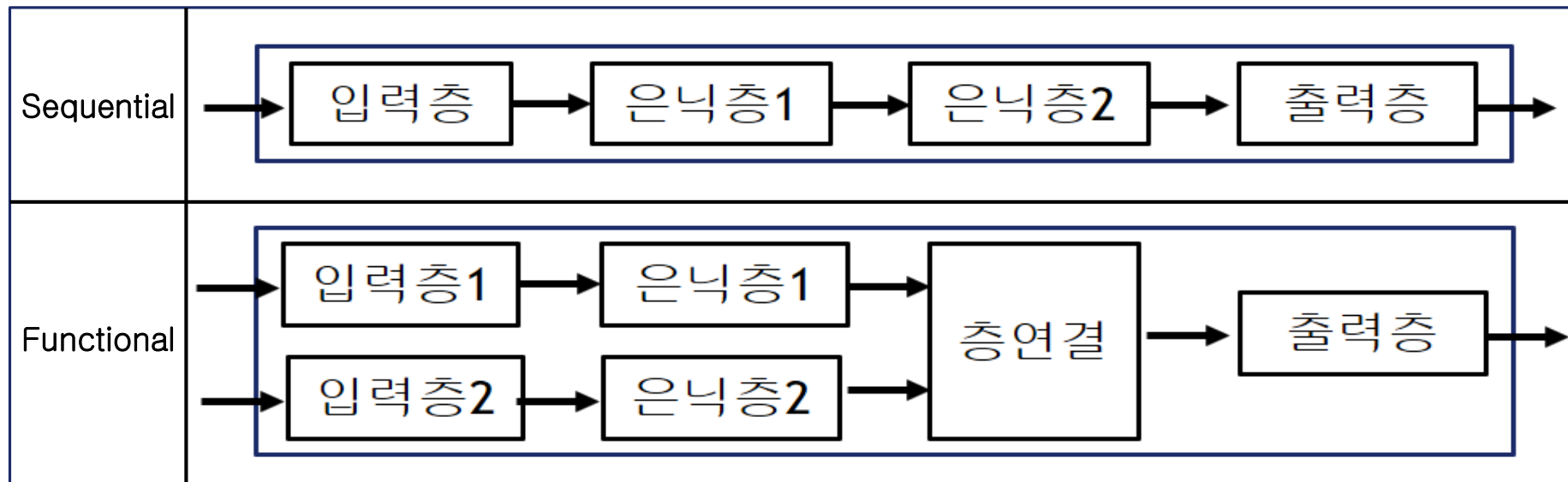
– 케라스의 모델(model)은 신경망 자체이며, 모든 모델의 기본 단위는 층(layer)으로 나타냄

<code>tf.keras.layers.Flatten()</code>	입력데이터(텐서)를 1차원 vector로 만들어주는 역할을 수행함
<code>tf.keras.layers.Dense(100, activation='relu')</code> <code>tf.keras.layers.Dense(100, activation='sigmoid')</code> <code>tf.keras.layers.Dense(10, activation='softmax')</code>	은닉층과 출력층을 의미하는 완전 연결층을 나타내며, 1 <sup>st</sup> 파라미터는 출력 노드 수이며, 활성화 함수는 <code>activation='...'</code> 나타냄

# Keras – Sequential API / Functional API



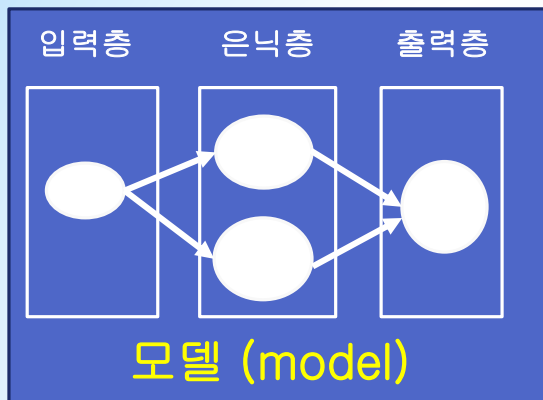
- 케라스 모델은 다양한 층(layer)로 이루어지는데, 이러한 층(layer)을 구성하는 방식에는 크게 Sequential API, Functional API, Subclassing API 등으로 정의할 수 있음
- 초급 개발자 또는 간단한 모델은 일반적으로 Sequential API, 전문가 또는 복잡한 모델은 Functional API 기반의 모델이 주로 사용됨





# Keras Sequential model

모델  
구축



```
model = Sequential()
model.add(Flatten(input_shape=(1,)))
model.add(Dense(2, activation='sigmoid'))
model.add(Dense(1, activation='sigmoid'))
model.add(Dense(2, activation='sigmoid', input_shape=(1,)))
```

①

②

컴파일

[예1] `model.compile(optimizer=SGD(learning_rate=0.1), loss='mse', metrics=['accuracy'])`

[예2] `model.compile(optimizer=Adam(learning_rate=1e-4), loss='categorical_crossentropy')`

※ 손실함수 종류는 정답이 실수 'mse', 정답이 0 또는 1 인 이항분류 'binary\_crossentropy', 다중 분류시 one-hot encoding 한 후에 넣어주는 경우 'categorical\_crossentropy', 다중 분류시 one-hot encoding 하지않고 정수로 넣어주는 경우 'sparse\_categorical\_crossentropy'

학습

[예] `model.fit(x_train, t_train, epochs=10, batch_size=100, verbose=0, validation_split=0.2)`

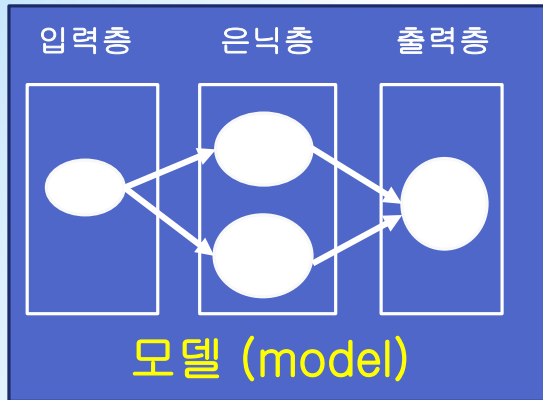
평가,  
예측,  
저장,  
로드

`model.evaluate(x_test, t_test)` , `model.predict(x_input_data)`

`model.save("model_name.h5")` , `model = tensorflow.keras.models.load_model("model_name.h5")`

# Keras Functional model ( <https://youtu.be/KugOEdth3iE> )

모델  
구축



```
in_ = Input(shape=(1,))
```

```
x = Dense(2, activation='sigmoid')(in_)
```

```
out_ = Dense(1, activation='sigmoid')(x)
```

```
model = Model(inputs=in_, outputs=out_)
```

컴파일

[예1] `model.compile(optimizer=SGD(learning_rate=0.1), loss='mse', metrics=['accuracy'])`

[예2] `model.compile(optimizer=Adam(learning_rate=1e-4), loss='categorical_crossentropy')`

※ 손실함수 종류는 정답이 실수 'mse', 정답이 0 또는 1 인 이항분류 'binary\_crossentropy',

다중 분류시 one-hot encoding 한 후에 넣어주는 경우 'categorical\_crossentropy',

다중 분류시 one-hot encoding 하지않고 정수로 넣어주는 경우 'sparse\_categorical\_crossentropy'

학습

[예] `model.fit(x_train, t_train, epochs=10, batch_size=100, verbose=0, validation_split=0.2)`

평가,  
예측,  
저장,  
로드

`model.evaluate(x_test, t_test)` , `model.predict(x_input_data)`

`model.save("model_name.h5")` , `model = tensorflow.keras.models.load_model("model_name.h5")`

# Keras - 딥러닝 모델 구축 (API 흐름도)

API  
흐름도

Sequential model API

또는

Functional model API

model.compile()

model.summary()

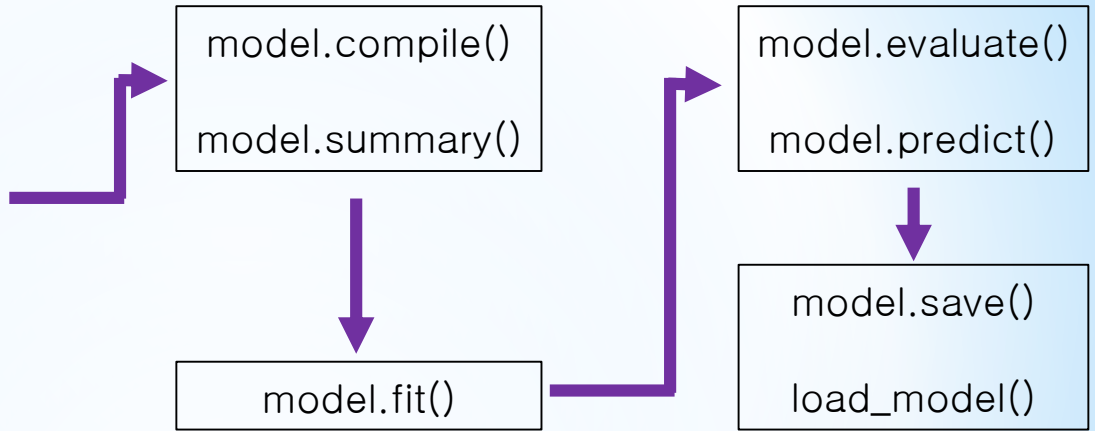
model.fit()

model.evaluate()

model.predict()

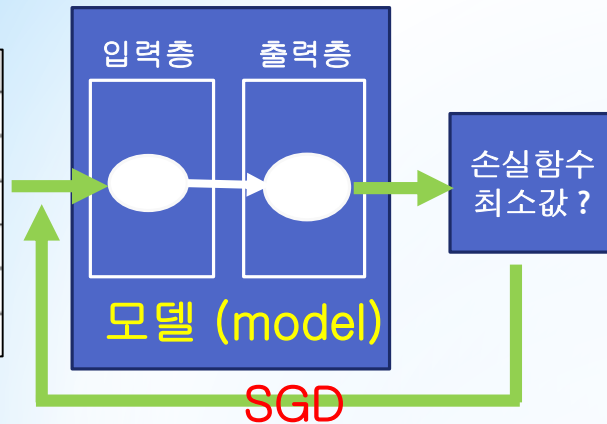
model.save()

load\_model()



# Keras – Simple LinearRegression Exercise (Sequential)

입력	정답
1	3
2	4
3	5
4	6
5	7
6	8



```
import tensorflow as tf

from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Flatten, Dense, Input
from tensorflow.keras.optimizers import SGD

import numpy as np
```

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

```
model = Sequential()      # 모델

model.add(Flatten(input_shape=(1,)))      # 입력층

model.add(Dense(1, activation='linear'))      # 출력층

# model.add(Dense(1, input_shape=(1,), activation='linear'))
```

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

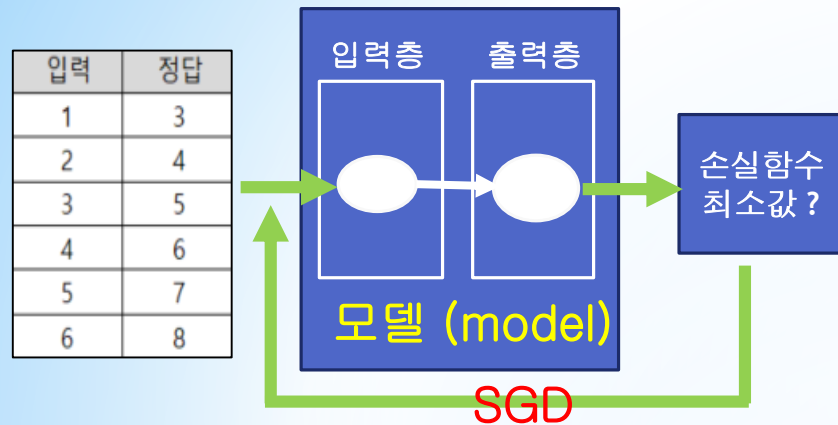
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, 1)	0
dense (Dense)	(None, 1)	2

Total params: 2  
Trainable params: 2  
Non-trainable params: 0

# Keras – Simple LinearRegression Exercise (Sequential)



```
import tensorflow as tf

from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Flatten, Dense, Input
from tensorflow.keras.optimizers import SGD

import numpy as np
```

```
hist = model.fit(x_data, t_data, epochs=1000)
```

```
test_input_data = np.array([-3.1, 3.0, 3.5, 15.0, 20.1])
label_data = test_input_data + 2.0
```

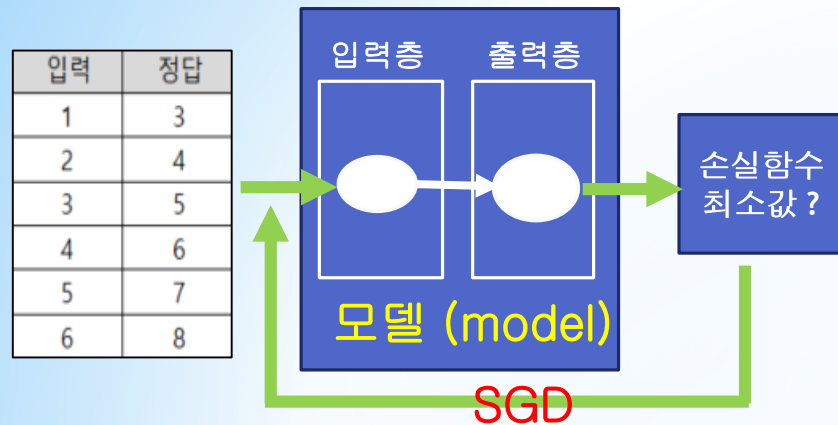
```
result = model.predict(test_input_data)
```

```
print(result)
```

```
print(label_data.reshape(5,1))
```

```
[[ -1.1591684]
 [  4.98973  ]
 [  5.4937377]
 [17.085922  ]
 [22.226805  ]]
[[-1.1]
 [ 5. ]
 [ 5.5]
 [17. ]
 [22.1]]
```

# Keras – Simple LinearRegression Exercise (Functional)



```
import tensorflow as tf

from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Flatten, Dense, Input
from tensorflow.keras.optimizers import SGD

import numpy as np
```

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

```
input_ = Input(shape=(1,))

#x = Flatten()(input_)

output_ = Dense(1, activation='linear')(input_)

model = Model(inputs=input_, outputs=output_)
```

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

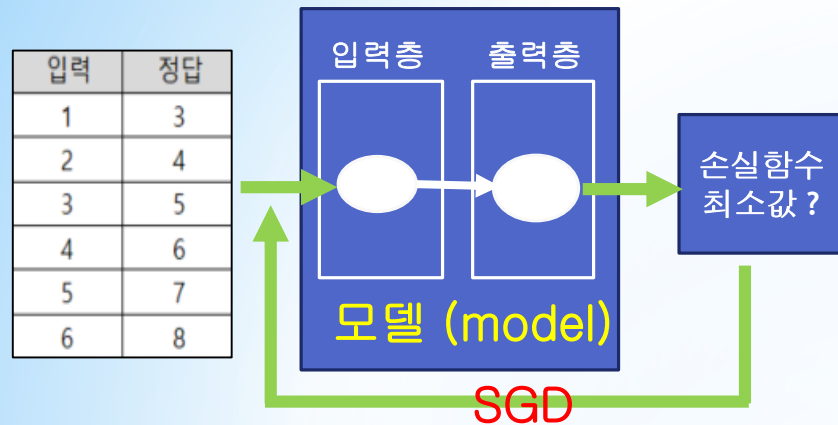
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, 1)	0
dense (Dense)	(None, 1)	2

Total params: 2  
Trainable params: 2  
Non-trainable params: 0

# Keras – Simple LinearRegression Exercise (Functional)



```
import tensorflow as tf

from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Flatten, Dense, Input
from tensorflow.keras.optimizers import SGD

import numpy as np
```

```
hist = model.fit(x_data, t_data, epochs=1000)
```

```
test_input_data = np.array([-3.1, 3.0, 3.5, 15.0, 20.1])
label_data = test_input_data + 2.0
```

```
result = model.predict(test_input_data)
```

```
print(result)
```

```
print(label_data.reshape(5,1))
```

```
[[ -1.1783116]
 [  4.9864073]
 [  5.4917116]
 [17.113722 ]
 [22.267832 ]]

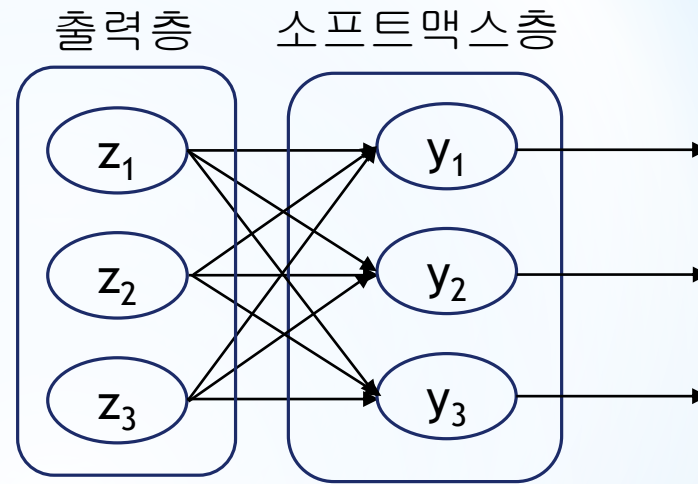
[[-1.1]
 [ 5. ]
 [ 5.5]
 [17. ]
 [22.1]]
```



## 출력층에서의 Softmax

소프트맥스 (softmax) 함수는 다음과 같이 정의됨. 여기서  $n$  은 출력층의 노드 개수,  $y_k$  는 그 중  $k$  번째 출력임을 나타냄

$$y_k = \frac{e^{z_k}}{\sum_{i=1}^n e^{z_i}}$$



소프트맥스 (softmax) 함수 출력은 0.0 ~ 1.0 이며, 또한 소프트맥스 함수 출력의 총합은 1 이 되는 것을 알 수 있음. 출력 총합이 1 이 되는 것은 소프트맥스 함수의 중요한 성질이며, 이 성질로 인해서 소프트맥스 함수 출력을 확률로 해석할 수 있음.



## 출력층에서의 Softmax

딥러닝을 이용하여 강아지, 고양이, 코알라, 다람쥐를 분류하고자 한다.

각각의 이미지는  $32 \times 32 \times 1$  shape 을 가진다고 할 경우

[1] 입력층(input layer) 노드 개수를 구하시오

[2] 출력층(output layer) 노드 개수를 구하시오 (activation function은 softmax)

[3] 학습을 마친 후, 임의의 1개 이미지에 대한 prediction 수행하였을 경우, prediction 에 대한 shape 을 구하시오

[4] 강아지 = 0, 고양이 = 1, 코알라 = 2, 다람쥐 = 3 으로 정답을 가정할 경우, prediction 결과값이  $[0.2 \quad 0.1 \quad 0.6 \quad 0.1]$  인 경우, 입력된 이미지는 무엇이며 해당 결과값을 바탕으로 softmax 를 통해 알수 있는 insight 를 정리하시오

다음과 같은 XOR 을 나타내는 Training Data에 대해서, [1] 은닉층 노드 3개를 가지는 딥러닝을 이용하여 구현한 후에, Test Data 이용하여 prediction 수행한 후, 결과를 확인 하시오.

x1	x2	t
0	0	0
0	1	1
1	0	1
1	1	0

Training Data

x1	x2
0	0
0	1
1	0
1	1

Test Data

[2] 은닉층 노드를 2개를 가지는 경우와 은닉층 노드 1개를 가지는 각각의 경우에 대해서 학습을 진행하고, Test Data 이용하여 prediction 수행한 후, 결과를 확인 하시오.

다음의 조건에 맞게 DeepLearning\_Example\_1 예제를 수정하고 학습과 예측을 진행 하시오

[1] 2개의 노드를 가지는 1번째 은닉층    [2] 3개의 노드를 가지는 2번째 은닉층

x1	x2	t
0	0	0
0	1	1
1	0	1
1	1	0

Training Data

x1	x2
0	0
0	1
1	0
1	1

Test Data