인공지능을 위한

# 파이썬 코딩 기초

## 인공지능과 텐서플로우

김루진 강사

소속

인공지능을 위한

# 파이썬 코딩 기초

## 학습 내용

1. 텐서플로어
2. 텐서
3. 텐서플로어 수행
4. CNN 예

# 텐서플로우(Tensorflow)

## 엔드 투 엔드 머신러닝 플랫폼

# 텐서플로우(Tensorflow)

**머신러닝&딥러닝을 위한 고성능 수치 계산 패키지**


- 한 개 이상의 CPU또는 GPU를 사용하여 병렬처리 가능
- 구글의 브래인팀이 개발하여 오픈 API로 배포

- www.tensorflow.org

# 텐서플로우(Tensorflow)

## 데이터 플로우

- 노드(Mathematical Operation) 및 에지(Multidimensional data array) 즉 Tensor로 구성
- 모든 데이터는 Tensor로 표현
- 파이토치와 함께 가장 많이 사용되고 있는 인공지능 프레임워크

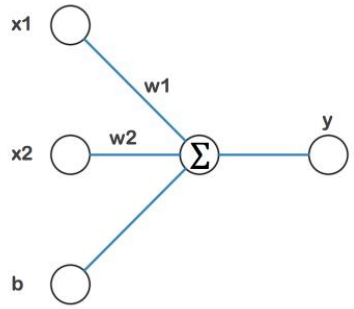
# 텐서플로우(Tensorflow)

## 텐서플로우의 이점, '추상화'

### 개발자는 애플리케이션의 전체적인 논리에만 집중

- TensorFlow는 다양한 수준의 추상화를 제공하므로 사용자는 자신의 요구에 맞는 수준을 선택할 수 있습니다.
- 상위 수준의 Keras API를 사용하여 모델을 빌드하고 학습시키세요.
- 그러면 TensorFlow 및 머신러닝을 쉽게 시작할 수 있습니다.

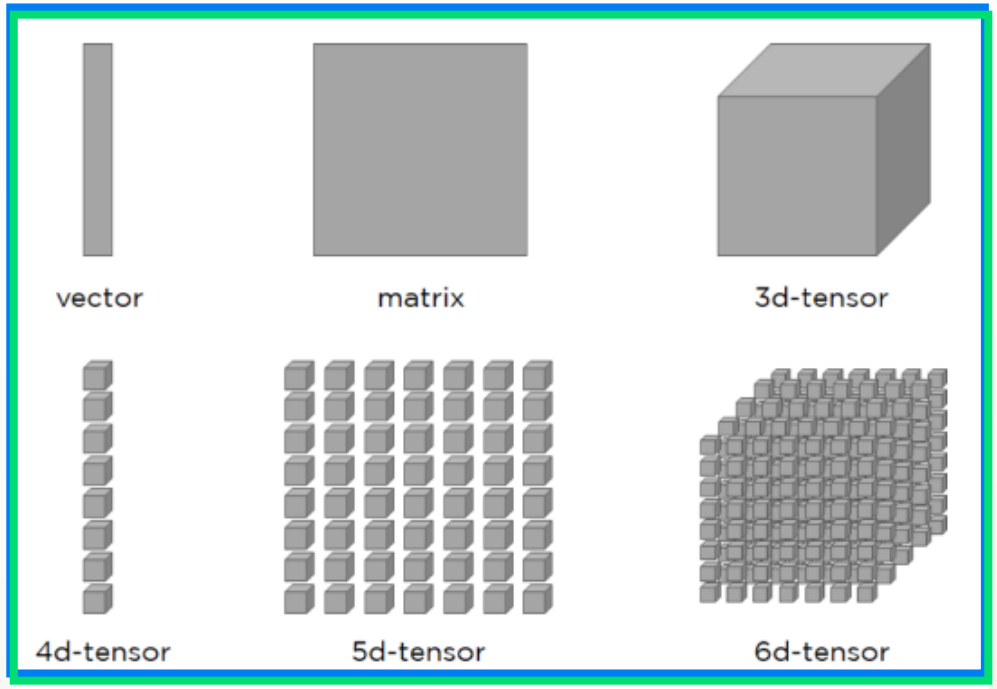
$$y = XW + b$$



# 텐서플로우(Tensorflow)

## 텐서

- 텐서는 다차원 배열,넘파이(NumPy) ndarray 객체와 비슷




| RANK | TYPE | EXAMPLE |
|---|---|---|
| 0 | scalar | [1] |
| 1 | vector | [1,1] |
| 2 | matrix | [[1,1],[1,1]] |
| 3 | 3-tensor | [[[1,1],[1,1]],[[1,1],[1,1]],[[1,2],[2,1]]] |
| n | n-tensor | |

출처 https://rekt77.tistory.com/102

# 딥러닝 구현

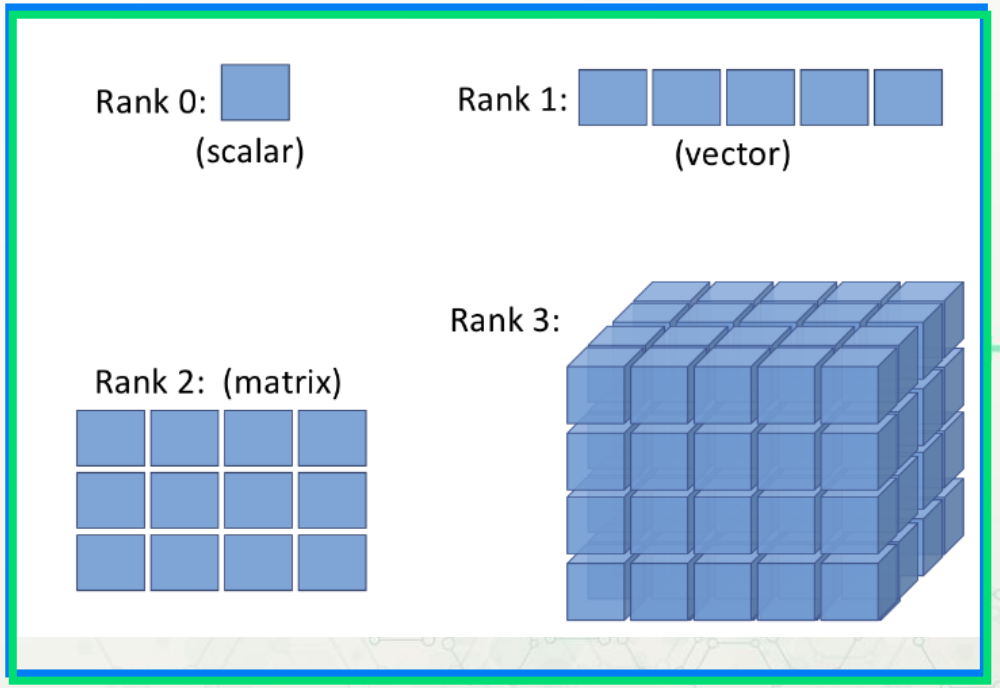
- TensorFlow
  - 텐서(Tensor)를 흘려 보내면서(Flow) 딥러닝 알고리즘 수행
- 케라스(Keras)
  - TensorFlow 2.0 부터는 직관적인 High-Level API

# 딥러닝 구현

## Tensor

- n차원의 배열로 numpy와 호환하며 사용

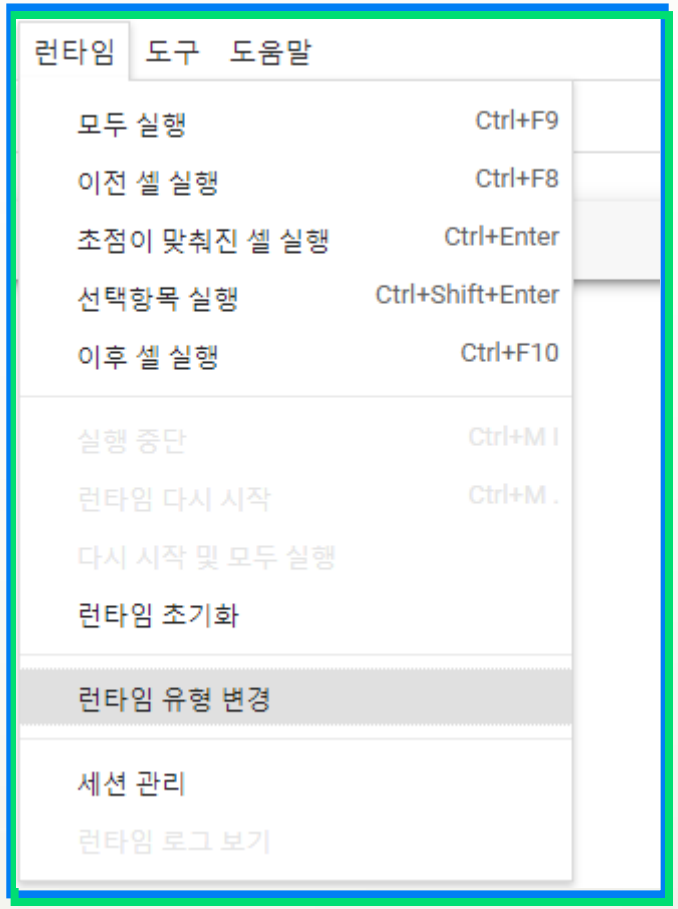
| | |
|---|---|
| Rank | Tensor의 차원수 |
| Shape | 차원의 모양 |
| Data Type | int,float,double... |



# 텐서플로우 사용하기

## Colab 주소

- https://colab.research.google.com/ 활용(64bit)




- Colab에서 GPU를 사용하는 방법
  - 새 노트에 진입했을 경우
    - 상단에서 런타임 > 런타임 유형 변경 클릭

# 텐서플로우 사용하기

- 필요한 패키지 설치
  - pip install tensorflow
  - pip install keras
  - pip install gensim

# 텐서플로우 사용하기

- 함수를 사용하여 리스트나 넘파이 배열에서 텐서 만들기
  - tf.convert_to_tensor 함수
    - 리스트나 넘파이 배열에서 텐서 만드는 함수

# 텐서플로우 사용하기

## 텐서 만들기

```
a = np.array([1, 2, 3], dtype=np.int32)
b = [4, 5, 6]
t_a = tf.convert_to_tensor(a)
t_b = tf.convert_to_tensor(b)


print(t_a)
print(t_b)
```

```
>>> tf.Tensor([1 2 3], shape=(3,), dtype=int32)
```

# 텐서플로우 사용하기

## 속성 확인하기

```
t_ones = tf.ones((2, 3))


t_ones.shape
```

>>>TensorShape([2, 3])

```
t_ones.numpy()
```

>>> array([[1., 1., 1.], [1., 1., 1.]], dtype=float32)

# 텐서플로우

## TensorFlow 2.0

- 2019년 9월 30일, TensorFlow 2.0 정식 Release1.x 버전과 비교
- 즉시 실행 모드로 불리는 Eager Execution 적용되어 코드의 직관성이 높음

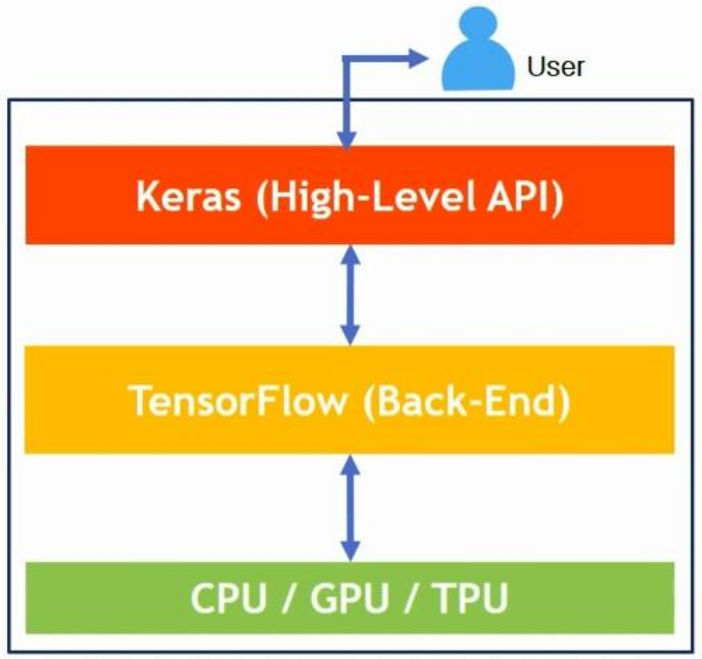
Eager Execution

```
W = tf.Variable(tf.random.normal([1]))  # 가우시안 분포
print('initial W = ', W.numpy())
print('=========================================')

# session 생성 없이 즉시 실행 (Eager Execution)
# numpy() 메서드 사용하면 numpy 값을 리턴해줌

for step in range(2):
    W = W + 1.0
    print('step = ', step, ' , W = ', W.numpy())
```

```
initial W =  [0.588869]
=========================================
step =  0 , W =  [1.588869]
step =  1 , W =  [2.588869]
```




- 사용자 친화적이어서 쉽게 배울 수 있는 Keras만을 High-Level API로 공식 지원

# 텐서플로우

## TensorFlow 2.0

### Eager Execution (즉시 실행 모드)

```
import tensorflow as tf
import numpy as np

tf.__version__
```

Eager Execution

```
'2.2.0'
```

```
a = tf.constant(10)
b = tf.constant(20)

c = a + b
d = (a+b).numpy()

print(type(c))
print(c)
print(type(d), d)

d_numpy_to_tensor = tf.convert_to_tensor(d)

print(type(d_numpy_to_tensor))
print(d_numpy_to_tensor)
```

numpy() 메서드는 numpy 값을 리턴

tf.convert_to_tensor() 메서드는 numpy 값을 tensor 값으로 변환

```
<class 'tensorflow.python.framework.ops.EagerTensor'>
tf.Tensor(30, shape=(), dtype=int32)
<class 'numpy.int32'> 30
<class 'tensorflow.python.framework.ops.EagerTensor'>
tf.Tensor(30, shape=(), dtype=int32)
```

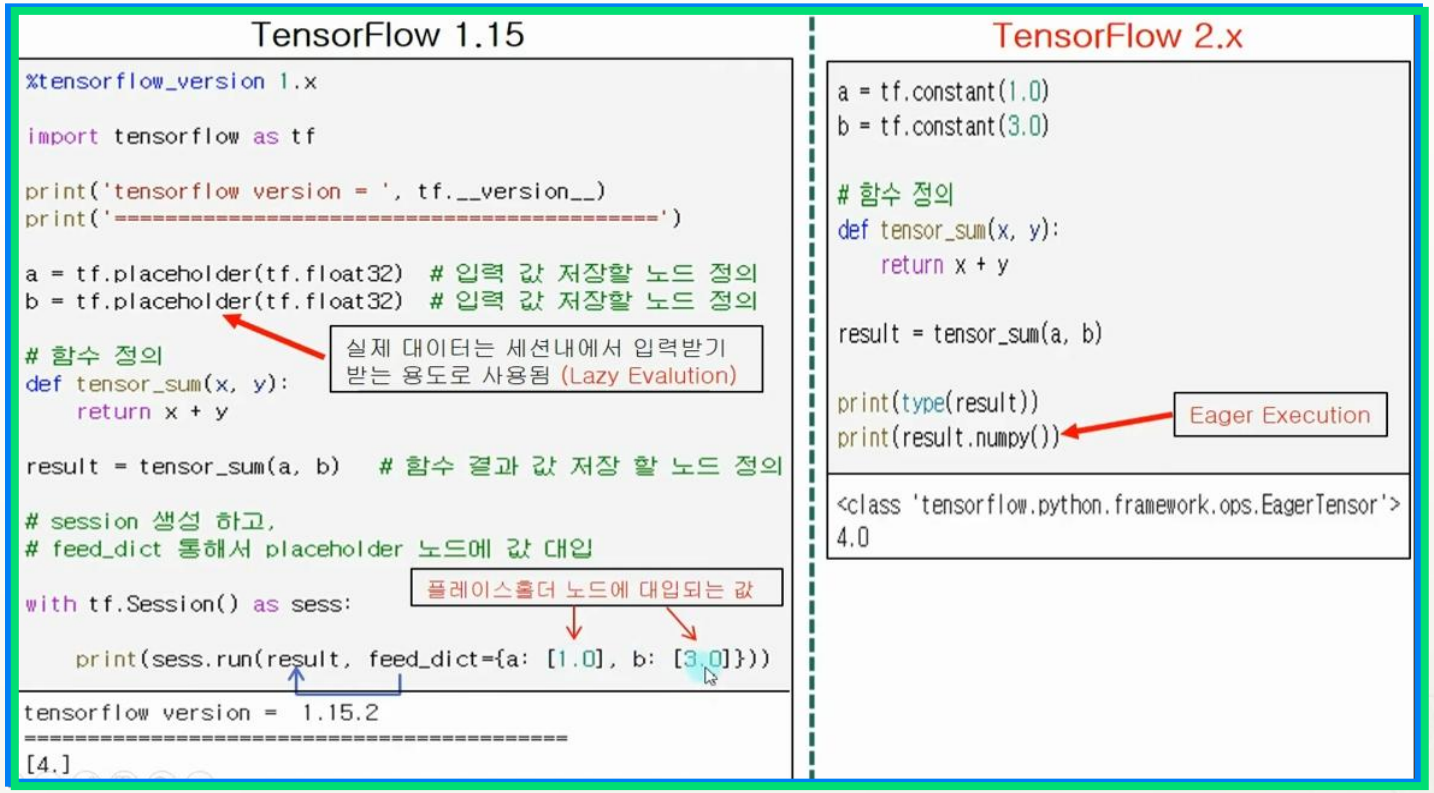
- numpy() 함수를 이용하면 파이썬의 넘파이 타입으로 변환 가능

→ Eager Execution 기능을 통해 텐서플로우를 파이썬처럼 사용 가능

# 텐서플로우

## TensorFlow 2.0

### Eager Execution – tf.placeholder 삭제

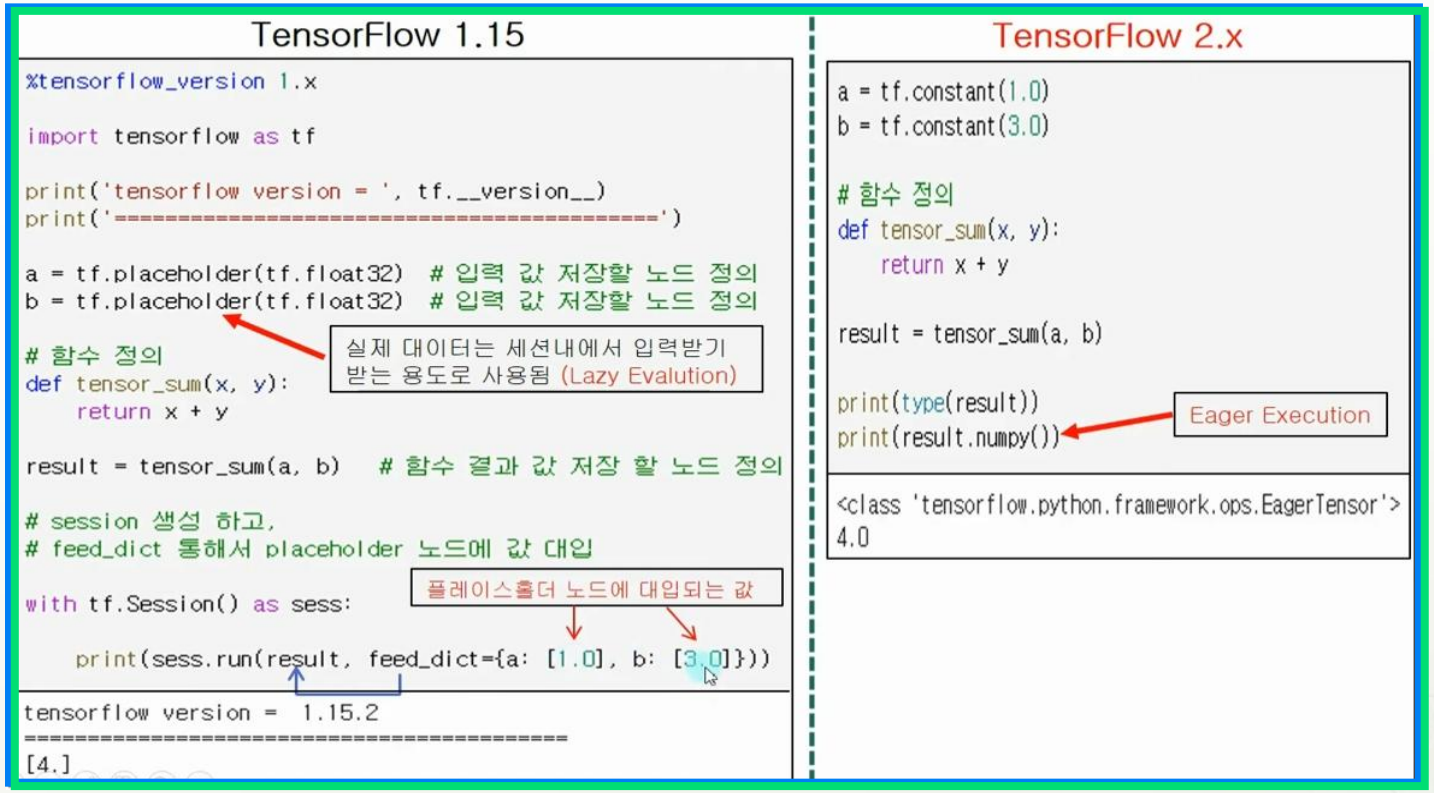



- TF 1.x 버전
  - 함수를 실행하여 결과를 얻기 위해서는 tf.placeholder()에 입력 값을 주고 그 값을 이용해 함수에서 정의된 연산을 실행

# 텐서플로우

## TensorFlow 2.0

### Eager Execution – tf.placeholder 삭제

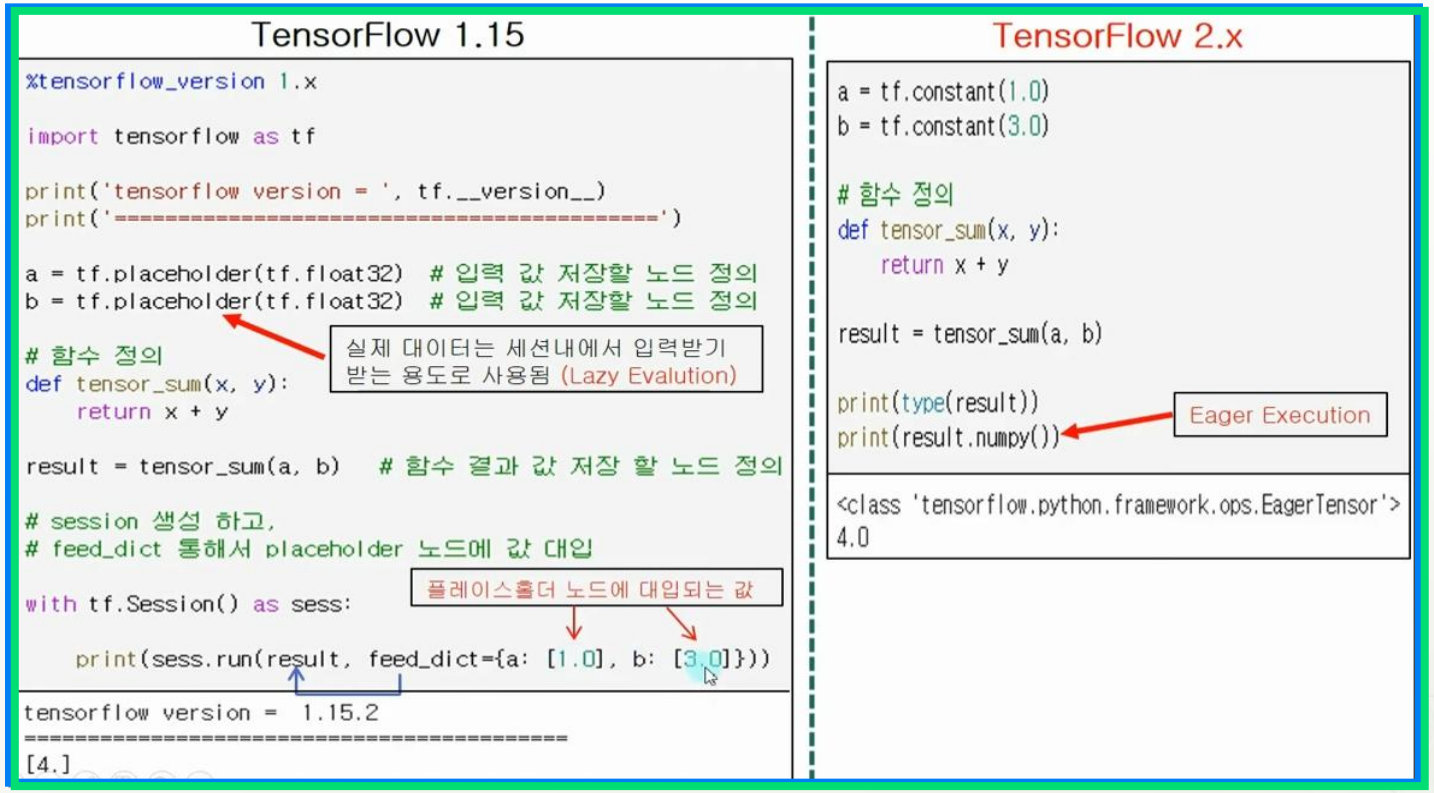



- TF 2.0
  - 일반적인 Python 코드와 마찬가지로 함수에 값을 직접 넘겨주면 즉시 결과를 얻을 수 있음

# 텐서플로우

## TensorFlow 2.0

### Eager Execution – tf.placeholder 삭제



# 텐서플로우 활용

머신러닝을 통해 까다로운 실생활 문제를 해결하도록 지원하는 전체 생태계

## 사용분야 사용되는 분야

- 필기 숫자 판별
- 이미지 인식
- 단어 임베딩
- 반복 신경망
- 기계 번역을 위한 시퀀스 투 시퀀스 모델
- 자연어 처리

# 텐서플로우 활용

머신러닝을 통해 까다로운 실생활 문제를 해결하도록 지원하는 전체 생태계

## 텐서플로우가 사용되는 곳

- 분류: 개체가 속한 범주를 식별
- 응용프로그램: 스팸 감지, 이미지 인식
- 알고리즘: SVM , 최근접 이웃 , 랜덤 포레스트
- 회귀:  객체와 관련된 연속 값 속성 예측
- 클러스터링: 유사한 개체를 세트로 자동 그룹화

# 텐서플로우(Tensorflow)

## 자연어 처리를 통한 예제 구성

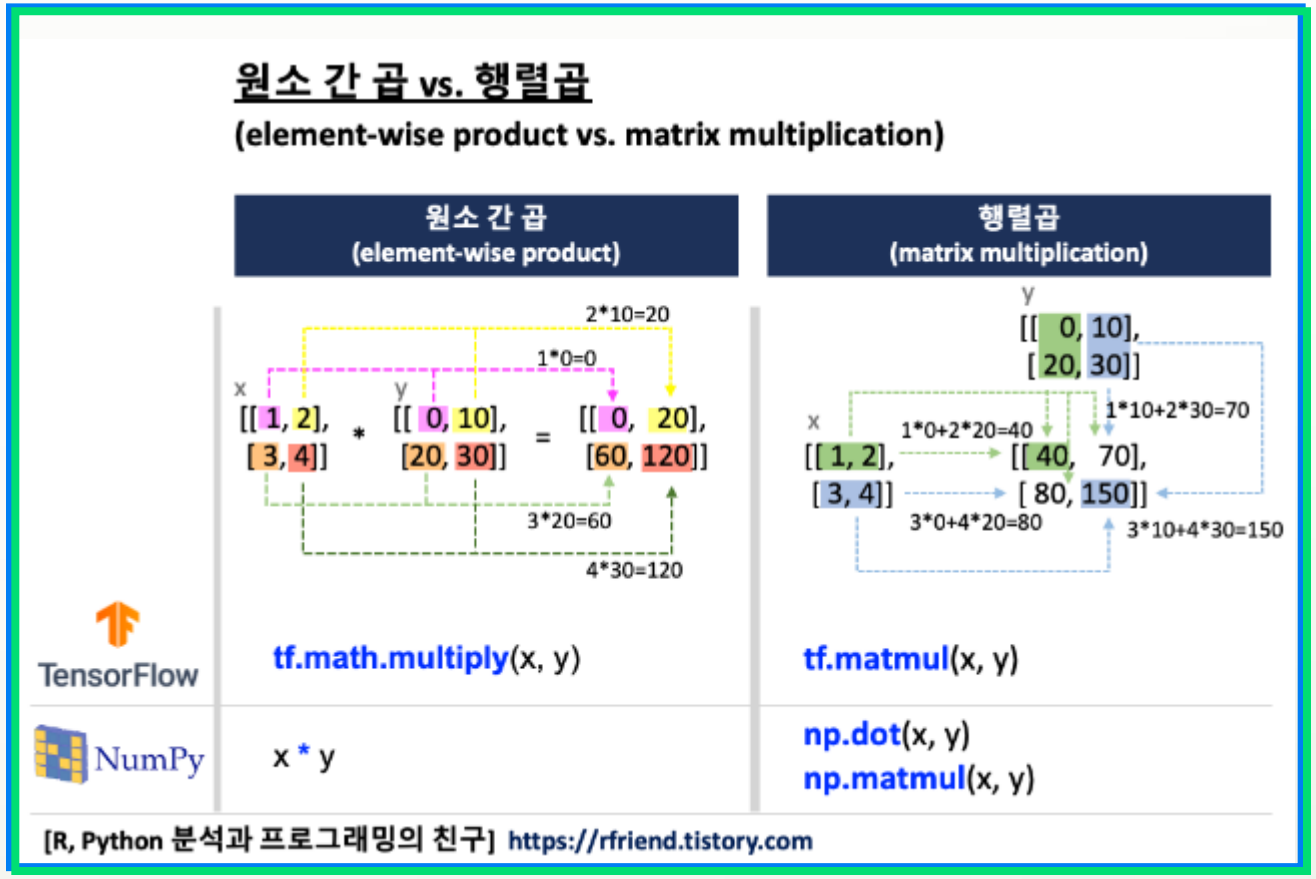
Hi King
Hi Queen
Hi Jack

| WORD | INDEX | EXAMPLE |
|---|---|---|
| hi | 0 | [1,0,0,0] |
| king | 1 | [0,1,0,0] |
| queen | 2 | [0,0,1,0] |
| jack | 3 | [0,0,0,1] |

출처 https://rekt77.tistory.com/102

# 텐서 만들기

## [Tensorflow] 행렬 원소간 곱 vs. 행렬 곱(tf.math.multiply() vs.tf.matmul())



# 감사합니다.