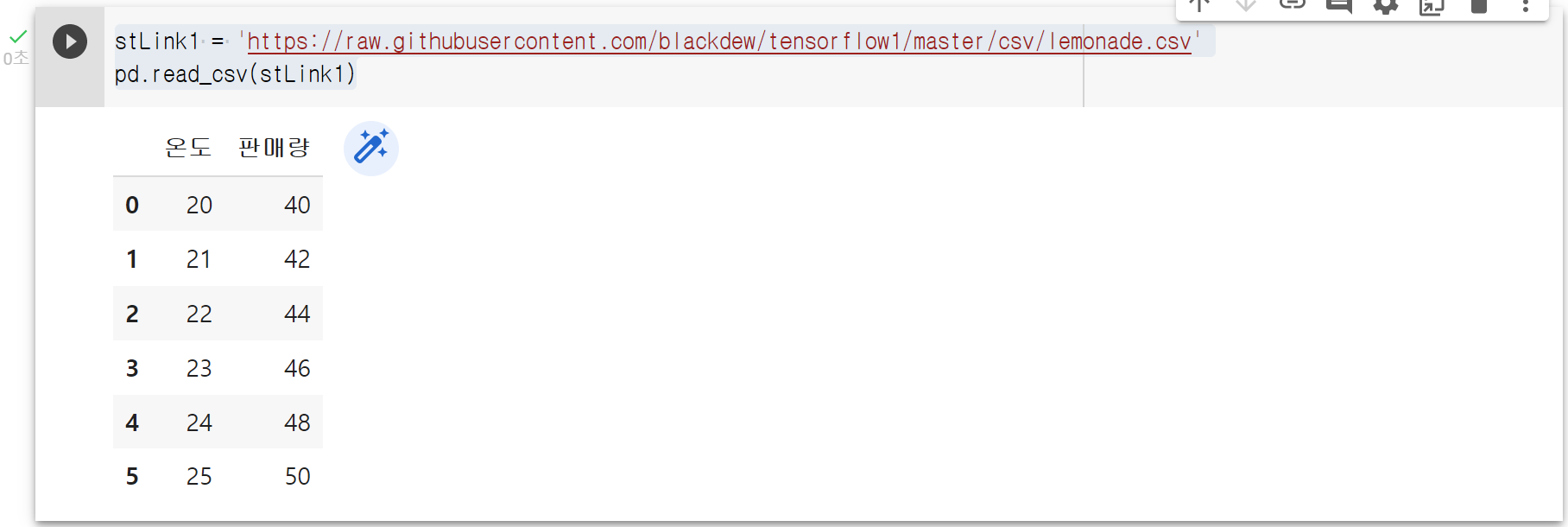
9. 딥러닝 실습 예제 1 - 레모네이드 판매 예측

1) 독립변수와 종속변수

지도학습을 진행하기 위해서는 ‘독립변수’와 ‘종속변수’에 대해서 알아야 합니다. 지금까지 ‘변수’라는 용어는 ‘특정한 값을 담아둔 것(정확히는 메모리 위치입니다)’을 변수라고 칭했습니다. 하지만 이 변수들 사이에서도 관계가 있을 수 있습니다.



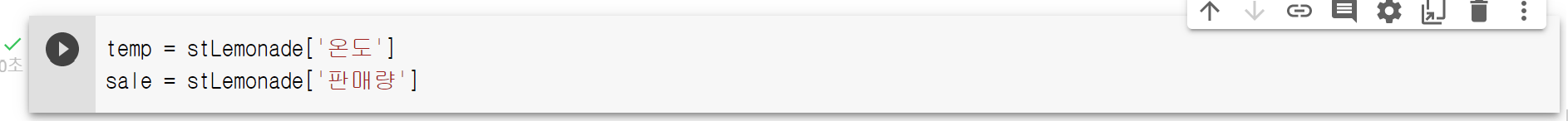
레모네이드 판매량을 보면 온도가 1도씩 오를 때마다 레모네이드가 2잔씩 더 팔리는 것을 알 수 있습니다. 원인은 온도, 결과는 판매량입니다. 이렇게 원인이 되는 변수를 ‘독립변수’, 결과가 되는 변수를 ‘종속변수’라고 합니다.

딥러닝 프로그램 제작을 위해 독립변수와 종속변수를 저장해줍시다.

이번엔 목차 8에서 사용했던 코랩 노트를 계속 쓸 것입니다.

temp = stLemonade[[‘온도’]]

sale = stLemonade[[‘판매량’]]

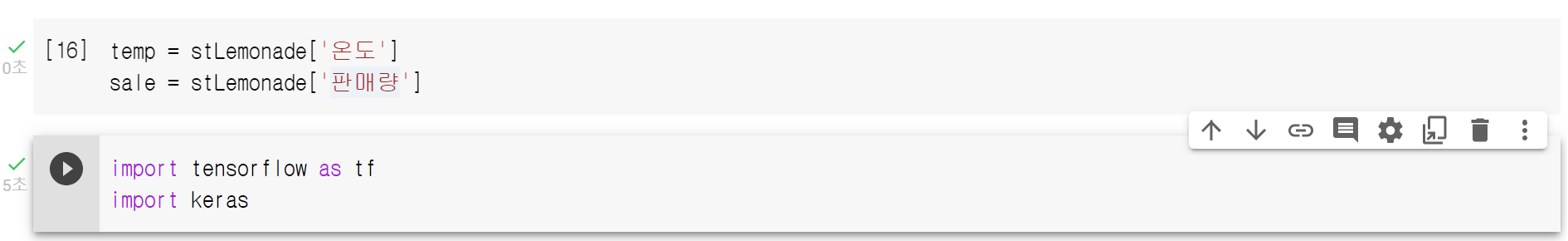


2) 본격적으로 머신러닝 모델 만들기

시작하기 전에 사용할 라이브러리를 먼저 준비합시다

import tensorflow as tf

import keras



여기서 머신러닝의 과정과 엮어서 우리가 해야 하는 일을 생각해봅시다.

1. 데이터 준비 - 이미 temp, sale이 준비되어 있습니다
2. 모델 구조 만들기 - 지금해야 하는 일입니다.
3. 모델을 이용해 학습하기 - 다행히도 이것은 라이브러리가 쉽게 해줄 것입니다.
4. 학습된 모델을 이용하여 값 예측하기

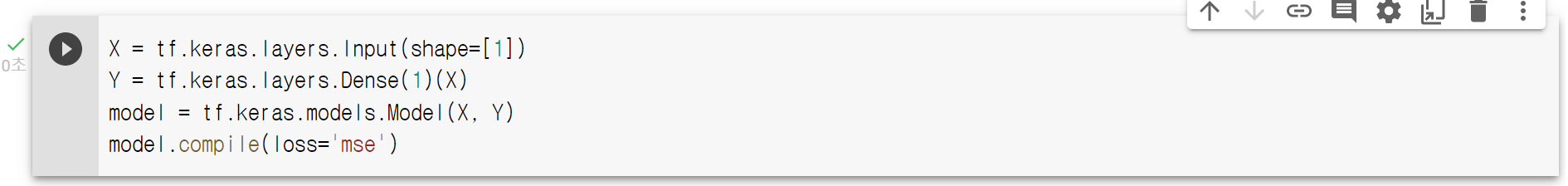
모델의 구조를 만들어야 하겠군요. 먼저 코드로 봅시다

X = tf.keras.layers.Input(shape=[1])

Y = tf.keras.layers.Dense(1)(X)

model = tf.keras.models.Model(X, Y)

model.compile(loss=’mse’)



갑자기 너무 어렵나요? 한 줄 한 줄 간단히 설명해드리겠습니다.

X = tf.keras.layers.Input(shape=[1])

1개의 입력을 받는 입력층을 구성한다. 독립변수를 말합니다. 여기선 독립변수가 temp 1개니 1로 합시다.

Y = tf.keras.layers.Dense(1)(X)

Dense는 영어로 밀접한, 조밀하게 모인 같은 뜻입니다. 딥러닝에서는 뉴런의 입력과 출력을 연결하는 역할을 합니다. 여기선 1개의 입력(X)로부터 1개의 출력을 만들어낸다는 뜻입니다.

model = tf.keras.models.Model(X, Y)

케라스의 모델을 만드는 함수를 이용. X를 입력 값으로, Y를 결과값으로 이용한다는 뜻입니다.

model.compile(loss=’mse’)

손실함수를 mse라는 것으로 한다는 뜻입니다. 여기서 손실함수란 예측 값과 정답의 오차를 말하며 이 손실함수를 최소화하는 방향으로 학습하게 됩니다. 손실에 대한 내용은 뒤에 더 자세하게 다룰 예정입니다.

* mse는 (예측 – 정답)^2의 공식으로 오차를 계산합니다.

3) 딥러닝 학습과 예측

이제 딥러닝 모델을 만들었으니 모델을 학습시킬 차례입니다.

model.fit(temp, sale, epochs = 1000)

텍스트이(가) 표시된 사진

자동 생성된 설명

뭔가 굉장히 많이 진행되는 것을 확인하실 수 있습니다. 코드부터 하나하나 천천히 살펴보도록 하죠

model.fit(temp, sale, epochs = 1000)

model은 방금 model = tf.keras.models.Model(X, Y)로 만든 변수를 의미합니다.

fit은 모델을 학습시키기 위한 명령입니다.

temp는 독립변수, sale은 위에서 다뤘듯이 종속변수입니다.

epochs = 1000은 학습 횟수를 1000번으로 설정하겠다는 뜻입니다.

우리가 어려운 수학 공식을 접했을 때, 문제를 여러 번 풀어보고 학습하듯이 컴퓨터도 비슷한 과정이 필요합니다. 우리가 문제를 많이 풀수록 공식을 더 잘 알게 되듯 컴퓨터도 마찬가지입니다.

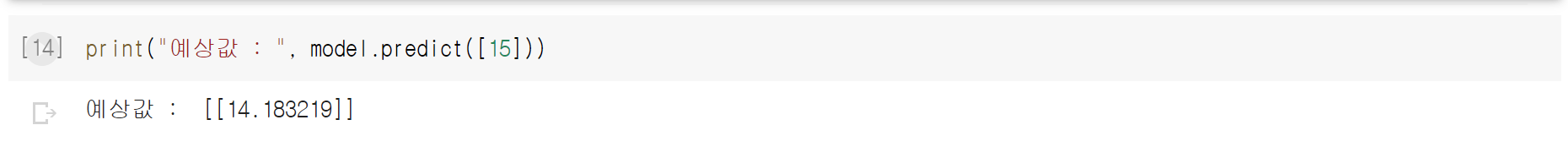
이전에 학습한 텐서플로 플레이그라운드를 통해 학습을 여러 번 진행하게 될수록 결과가 정답에 가까워진다는 것을 알 수 있었습니다. 모델을 완성하기 위해서는 충분한 학습이 필요하지요.

이제 모델을 학습시켰으니 이용해 제대로 완성됐나 확인할 차례입니다. 위에서 본 표를 다시 봅시다.



우리는 직감적으로 온도\*2 = 판매량인 것을 알 수 있습니다. 그럼 온도가 15도라면 판매량은 30이 나와야 합니다. 확인해봅시다.

print(“예상값 : ”, model.pridict([15]))



여러분들은 어떻게 나오셨나요? 우리가 예상한 값과 많이 다를 수도, 그렇지 않을 수도 있습니다.

저의 경우엔 완전 딴판의 값이 나왔습니다. 당황하지 말고 1000회 더 학습시킨 후 다시 예측을 실행해보았습니다.



이제서야 제대로 나왔습니다.

* 2000회 학습 후에도 오차가 여전히 클 수 있습니다. 당황하지 말고 적당한 값이 나올 때까지 계속 반복을 시켜보세요

4) 손실의 의미

첫번째 예측 값이 오차가 크게 발생한 이유가 무엇일까요? 그 이유는 아래 사진에서 알 수 있습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

우리가 주목해야 할 값은 loss입니다.

loss: 623.9893

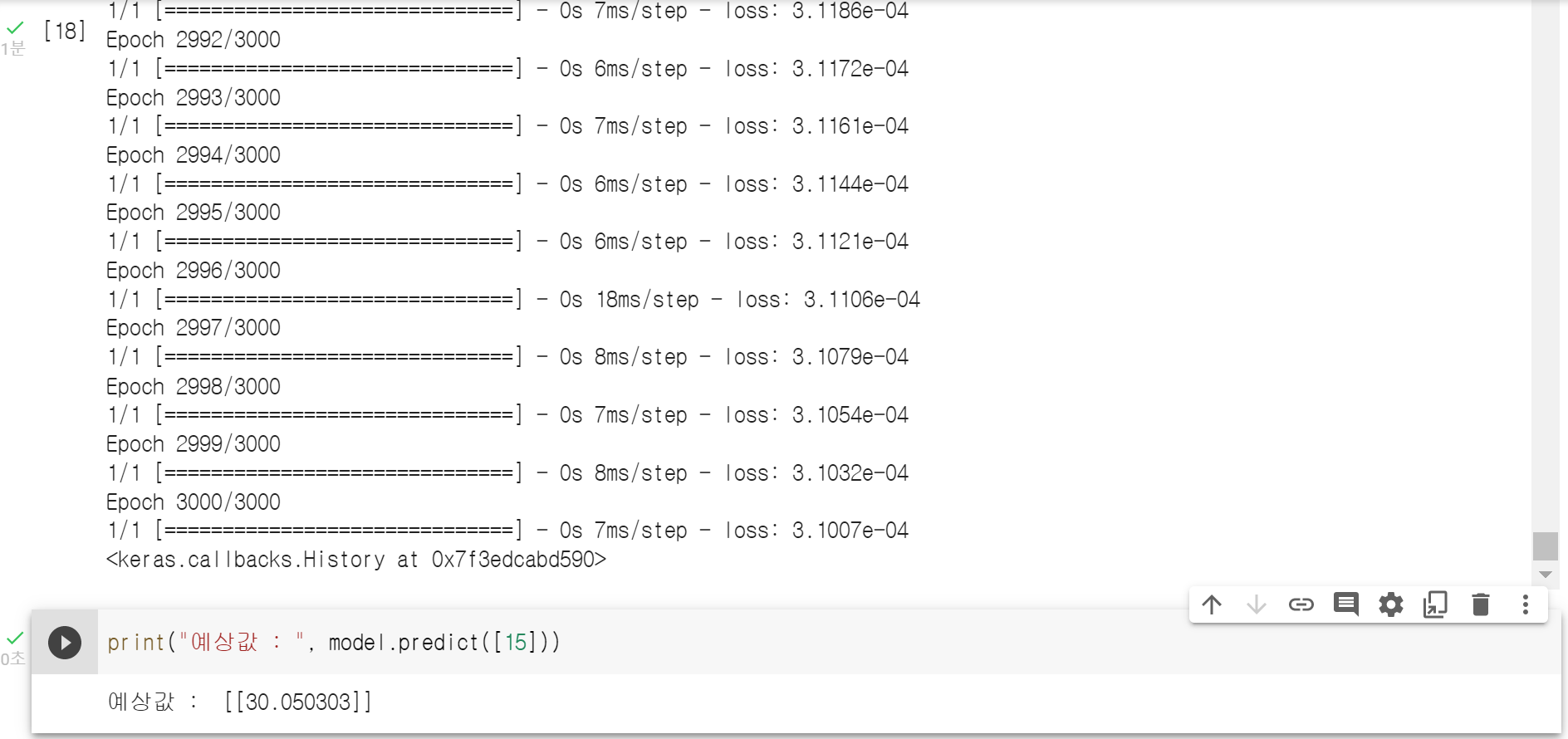
loss: 622.8226

loss: 621.6570 (이하 생략)

여기서 loss의 의미는 위에서 언급한 손실함수의 결과 즉 오차를 의미합니다. Epoch가 진행될 수록 오차가 줄어드는 것이 보이시나요? 하지만 1000회의 학습으로 오차를 줄이기엔 부족해서 1000회 더 학습을 진행했을 때는



위와 같이 오차가 유효하게 줄어드는 것을 확인할 수 있습니다. 손실이 원하는 만큼 유효하게 내려갈 때까지 반복해서 학습을 시키면 더 정확한 예측 값을 얻을 수 있겠죠. 반복해서 학습을 더 시켜보겠습니다.



loss는 0.00031007로 매우 크게 줄었고, 예상 값도 30.050303으로 정답에 훨씬 가까워졌습니다.

아직은 코드가 잘 이해되지 않았을 수 있습니다. 하지만 괜찮습니다. 우린 이제 배울 것이 더 없습니다. 우리가 만든 딥러닝 모델이 Epoch를 반복하듯이 우리도 반복을 통해 학습하다 보면 더 완벽해질 것입니다.