week8. 텍스트를 위한 인공 신경망

순차 데이터

: 텍스트나 시계열 데이터와 같이 순서에 의미가 있는 데이터. 글, 대화, 일자별 날씨, 일자별 판매 실적 등

순환 신경망

: 순차 데이터에 잘 맞는 인공 신경망. 순차 데이터를 처리하기 위해 고안된 순환층을 1개 이상 사용한 신경망을 순환 신경망이라고 함.

: 순환 신경망에서는 순환층을 '셀' 이라고 한다. 하나의 셀은 여러 개의 뉴런으로 구성된다.

: 순환 신경망에서는 셀의 출력을 '은닉 상태' 라고 부른다. 은닉 상태는 다음 층으로 전달될 뿐만 아니라 셀이 다음 타임스텝의 데이터를 처리할 때 재사용 된다.

```
from tensorflow.keras.datasets import imdb

(train_input, train_target), (test_input, test_target) = imdb
    num_words=200)

print(train_input.shape, test_input.shape)
-> (25000,) (25000,)

print(len(train_input[0]))
-> 218

print(len(train_input[1]))
-> 189

print(train_input[0])
-> [1, 14, 22, 16, 43, 2, 2, 2, 2, 65, 2, 2, 66, 2, 4, 173, 3
```

```
print(train_target[:20])
-> [1 0 0 1 0 0 1 0 1 0 1 0 0 0 0 0 1 1 0 1]
```

말뭉치

: 자연어 처리에서 사용하는 텍스트 데이터의 모음, 즉 훈련 데이터 셋.

토큰

: 텍스트에서 공백으로 구분되는 문자열. 종종 소문자로 변환하고 구둣점은 삭제.

원-핫 인코딩

: 어떤 클래스에 해당하는 원소만 1이고 나머지는 모두 0인 벡터.

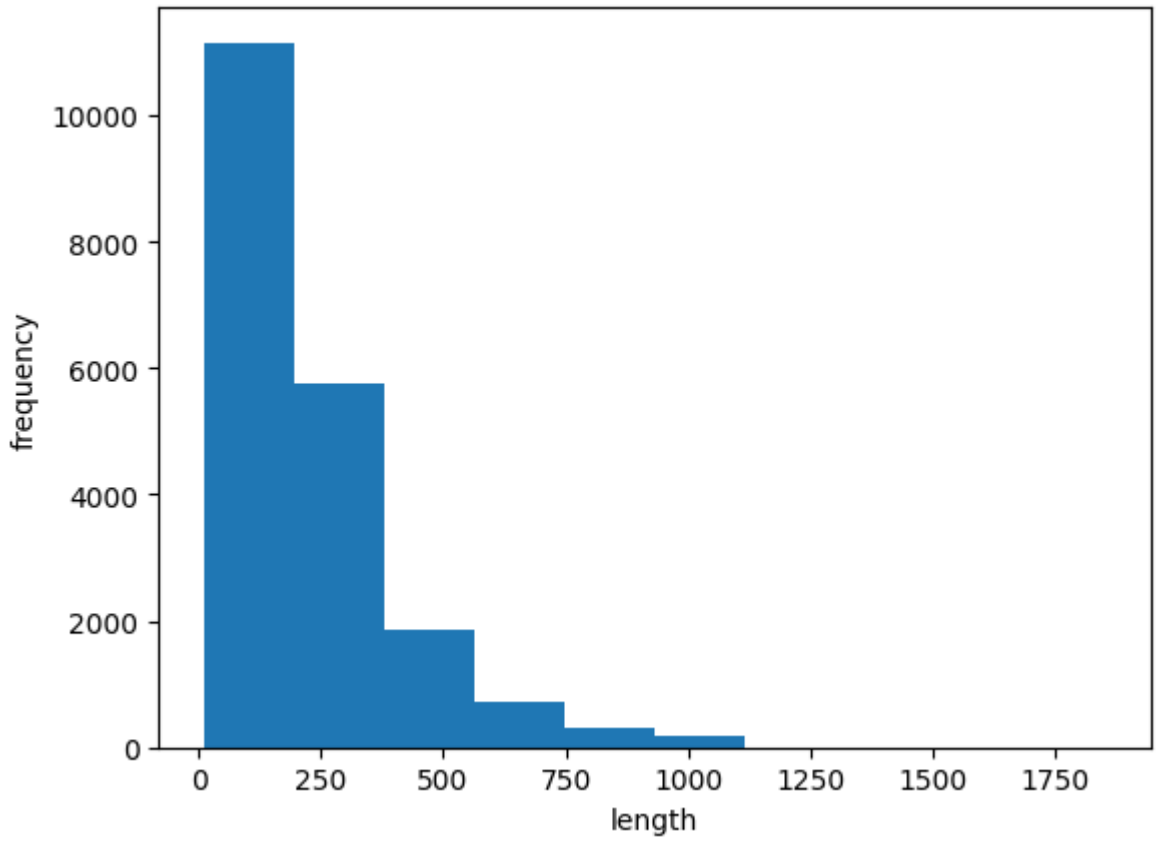
정수로 변환된 토큰을 원-핫 인코딩으로 변환하려면 어휘 사전 크기의 벡터가 만들어짐.

단어 임베딩

: 정수로 변환된 토큰을 비교적 작은 크기의 실수 밀집 벡터로 변환. 밀집 벡터는 단어 사이의 관계를 표현할 수 있기 때문에 자연어 처리에서 좋은 성능을 발휘.

```
import numpy as np

lengths = np.array([len(x) for x in train_input])

print(np.mean(lengths), np.median(lengths))
-> 239.00925 178.0

import matplotlib.pyplot as plt

plt.hist(lengths)
plt.xlabel('length')
```

```
plt.ylabel('frequency')
plt.show()
```



pad_sequences ( )

: 시퀀스 길이를 맞추기 위해 패딩을 추가. 이 함수는 (샘플 개수, 타임스텝 개수) 크기의 2차원 배열을 기대.

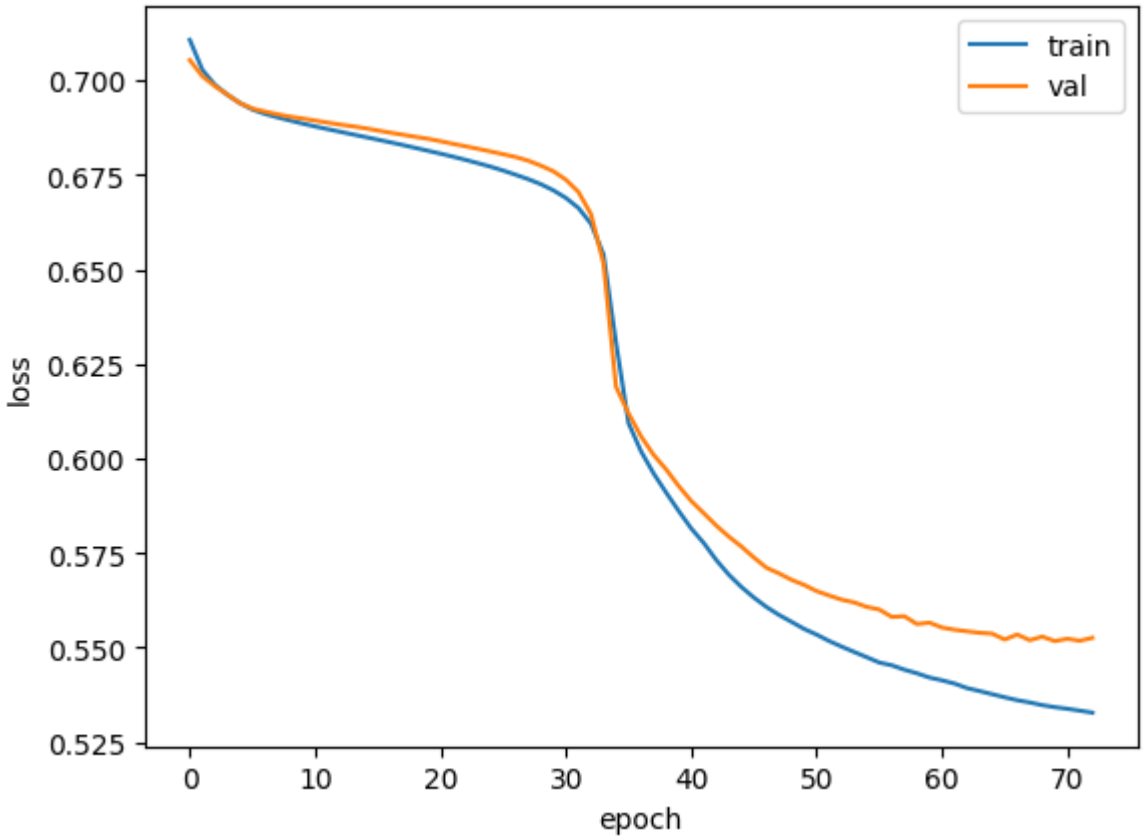
maxlen 매개변수로 패딩을 추가할 위치를 지정. 기본값인 'pre' 는 시퀀스 앞에 패딩을 추가하고 'post' 는 시퀀스 뒤에 패딩을 추가

truncating 매개변수는 긴 시퀀에서 잘라버릴 위치를 지정. 기본값이 'pre'는 시퀀스 앞부분을 잘라내고 'post' 는 시퀀스 뒷부분을 잘라냄

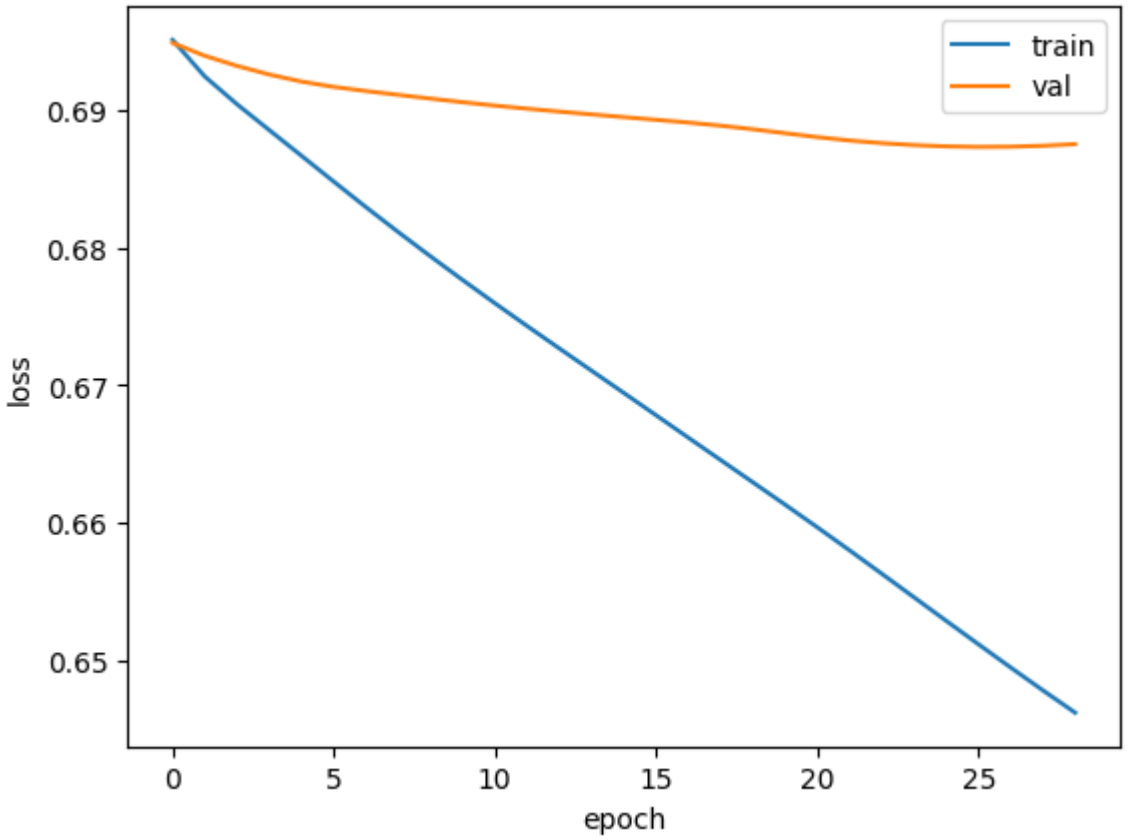
```
from tensorflow.keras.preprocessing.sequence import pad_seque

train_seq = pad_sequences(train_input, maxlen=100)
```

```
print(train_seq.shape)
print(train_seq[0])
print(train_input[0][-10:])
print(train_seq[5])

->
(20000, 100)
[ 10   4  20   9   2   2   2   5  45   6   2   2  33   2   8
   5   2  17  73  17   2   5   2  19  55   2   2  92  66 104
  76   2 151  33   4  58  12 188   2 151  12   2  69   2 142
   2   7   2   2 188   2 103  14  31  10  10   2   7   2   5
   2  30   2  34  14  20 151  50  26 131  49   2  84  46  50
   6   2  46   7  14  20  10  10   2 158]
[6, 2, 46, 7, 14, 20, 10, 10, 2, 158]
[  0   0   0   0   1   2 195  19  49   2   2 190   4   2   2
  10  13  82  79   4   2  36  71   2   8   2  25  19  49   7
   2   2   2  10  10  48  25  40   2  11   2   2  40   2   2
   2  95  14   2  56 129   2  10  10  21   2  94   2   2   2
  24   2   2   7  94   2   2  10  10  87   2  34  49   2   7
   2   2   2   2  46  48  64  18   4   2]
```

```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(['train', 'val'])
plt.show()
```




```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(['train', 'val'])
plt.show()
```



to_categorical() : 정수 시퀀스를 원-핫 인코딩으로 변환. 토큰을 원-핫 인코딩하거나 타깃 값을 원-핫 인코딩 할 때 사용.

num_classes 매개변수에서 클래스 개수를 지정. 지정하지 않으면 데이터에서 자동으로 찾음.

SimpleRNN : 케라스의 기본 순환층 클래스

첫 번째 매개변수에 뉴런의 개수를 지정.

activation 매개변수에서 활성화 함수를 지정. 기본값은 하이퍼볼릭 탄젠트인 'tanh'

dropout 매개변수에서 입력에 대한 드롭아웃 비율을 지정할 수 있음

return_sequences 매개변수에서 모든 타임스텝의 은닉 상태를 출력할지 결정.

기본값은 False

Embedding : 단어 임베딩을 위한 클래스

첫 번째 매개변수에서 어휘 사전의 크기를 지정.

두 번째 매개변수에서 Embeding 층이 출력할 밀집 벡터의 크기를 지정.

input_length 매개변수에서 입력 시퀀스의 길이를 지정. 이 매개변수는 Embedding 층 바로 뒤에 Flattern 이나 Dense 클래스가 올 때 꼭 필요함.

LSTM 셀 : LSTM 셀을 사용한 순환층 클래스

→ 타임스텝이 긴 데이터를 효과적으로 학습하기 위해 고안된 순환층.

입력 게이트, 삭제 게이트, 출력 게이트 역할을 하는 작은 셀이 포함되어 있음.

→ 은닉 상태 외에 셀 상태를 출력함. 셀 상태는 다음 층으로 전달되지 않으며, 현재 셀에서만 순환.

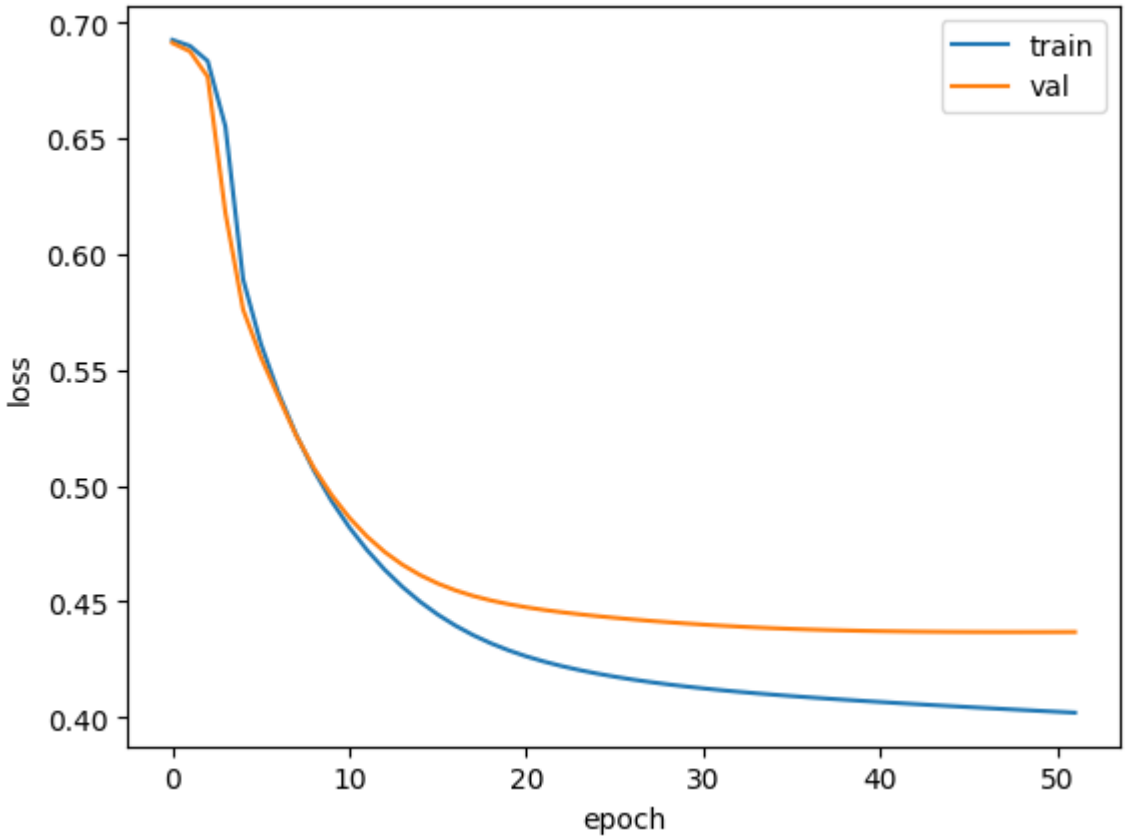
→ 첫 번째 매개변수에 뉴런의 개수를 지정

→ dropout 매개변수에서 입력에 대한 드롭아웃 비율을 지정할 수 있음

→ return_sequences 매개변수에서 모든 타임스텝의 은닉 상태를 출력할지 결정. 기본 값은 False

```
import matplotlib.pyplot as plt

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(['train', 'val'])
plt.show()
```

→



GRU 셀 : GRU 셀을 사용한 순환층 클래스

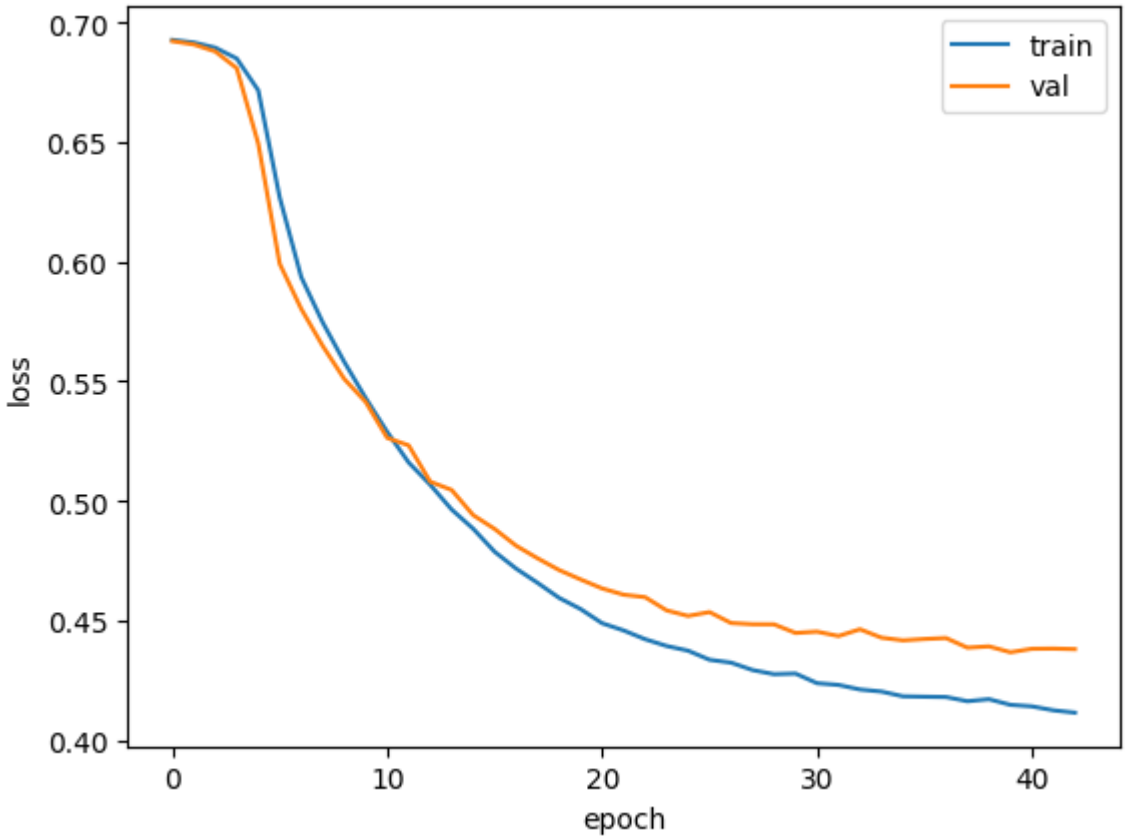
→ LSTM의 간소화 버전. 생략할 수 있으나, LSTM 셀에 못지 않는 성능.

→ 첫 번째 매개변수에 뉴런의 개수를 지정

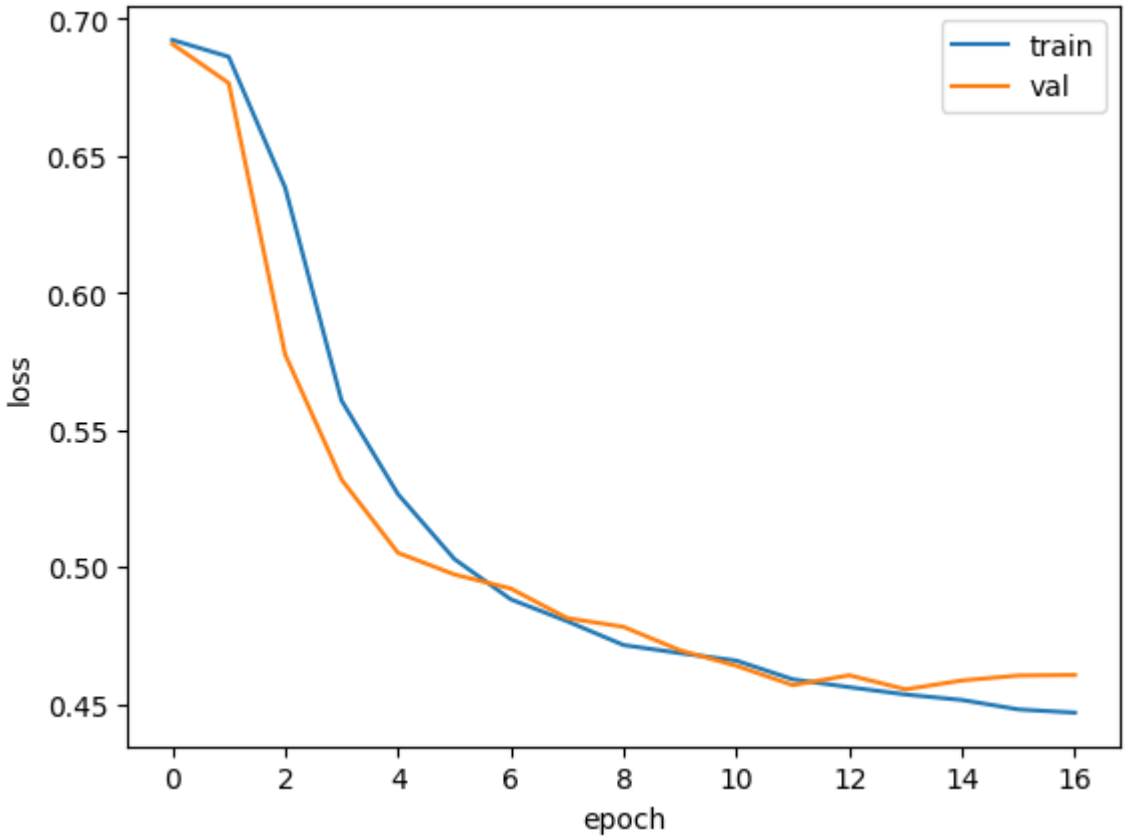
→ dropout 매개변수에서 입력에 대한 드롭아웃 비율을 지정할 수 있음.

→ return_sequences 매개변수에서 모든 타임스텝의 은닉 상태를 출력할지 결정. 기본 값은 False

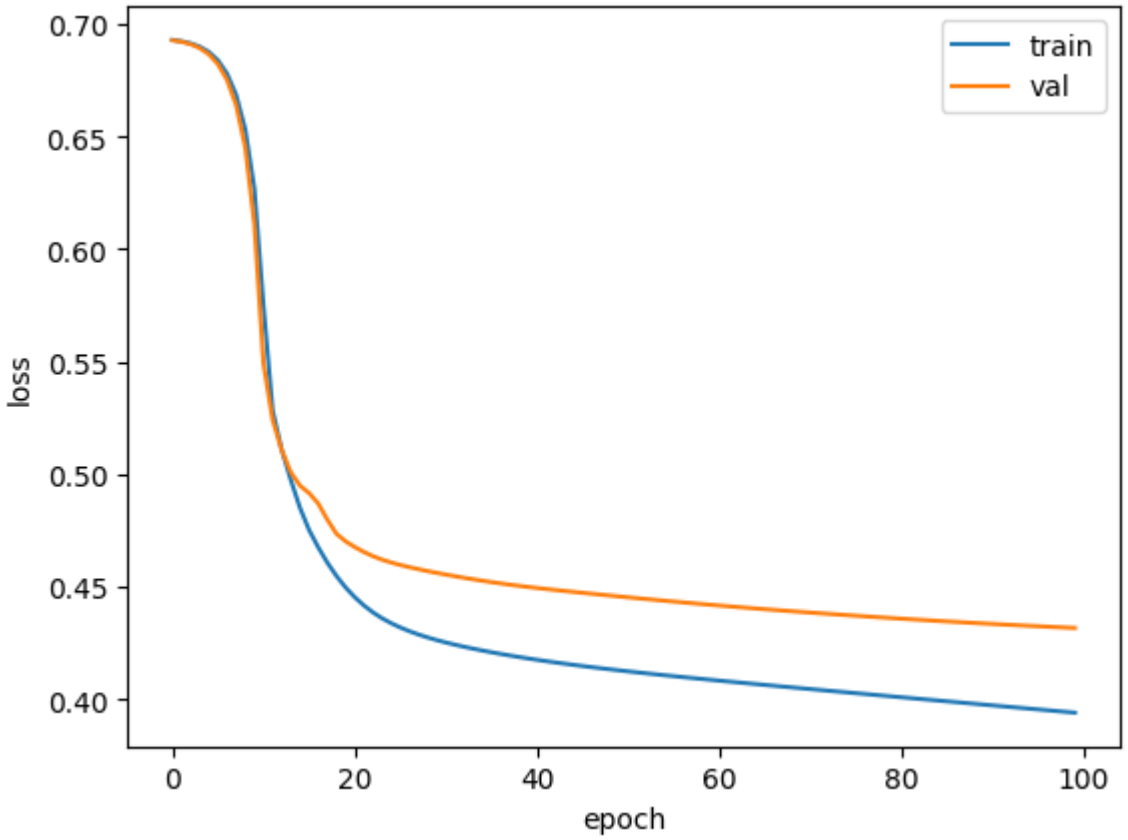
```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(['train', 'val'])
plt.show()
```




```
# 2개 층 연결하기

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(['train', 'val'])
plt.show()
```




```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(['train', 'val'])
plt.show()
```


0.70
0.65
0.60
0.55
0.50
0.45
0.40
loss
0
20
40
60
80
100
epoch
train
val