Deep Learning
Deep Neural Network

Yoon Joong Kim,
Hanbat National University

Deep Learning

RNN(3)
Recurrent Neural Network

RNN(1) : RNN,RNN Applications, Language model, RNN models

RNN(2) : (1)정현파신호 샘플의 예측 모델(SimpleRNN) in keras
(2)문자 기반 신경 언어 모델(many-to-one RNN)-sixpence in keras

RNN(3) : (1)문자 기반 신경 언어 모델(many-to-many RNN)-sixpence in keras
(2) 주가예측 모델

Yoonjoong Kim

Department of Computer Engineering, Hanbat National University

yjkim@hanbat.ac.kr

Agenda

1. RNNs
    1. From feed-forward to RNNs
    2. Simple Recurrent Neural Network (SRNN)
    3. RNNs in the context of NLP
    4. The problem with RNNs
    5. LSTM
2. RNN Applications
    1. Language Modeling
    2. Character-level Language Modeling
    3. Neural Machine Translation(Google Research's blog)
    4. Text Summarization
    5. Image Captioning
3. Language Modeling
    1. Language Modeling DEMO Character-level, Language Modeling
4. RNN models
5. Examples in Keras
    1. 정현파신호 샘플의 예측 모델(SimpleRNN) in keras
    2. 문자 기반 신경 언어 모델(many-to-one RNN)-sixpence in keras
    3. 문자 기반 신경 언어 모델(many-to-many RNN)-sixpence in keras
    4. 주가예측 모델

Example 3. 문자 기반 신경 언어 모델(may to many RNN)-sixpence

How to Develop a Character-Based Neural Language Model in Keras
Photo by hedera.baltica, some rights reserved.

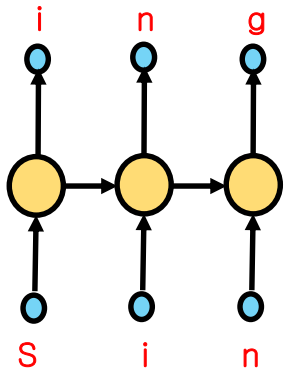
Sing a song of sixpence, a pocket full of rye,
Four and twenty blackbirds baked in a pie.
When the pie was opened the birds began to sing,
Oh wasn't that a dainty dish to set before the king?

The king was in his counting house counting out his money,
The queen was in the parlour eating bread and honey
The maid was in the garden hanging out the clothes,
When down came a blackbird and pecked off her nose!

6펜스 노래를 부르자, 주머니 가득 호밀이 있지.
찌르레기 24마리는 파이 안에서 구워 졌네.
파이를 잘랐을 때 새들은 노래를 부르기 시작했지
오, 저건 정말 왕에게 드릴만한 진미가 아닌가?

왕은 금고에서 돈을 세고 있었고,
왕비는 거실에서 꿀을 바른 빵을 먹고 있었네.
하녀는 정원에서 빨래를 널고 있는데
찌르레기 한 마리가 날아와 선 하녀의 코를 쪼았지.

- 문제
  - 영국 동요 sixpence (sixpence.txt)로 문자기반언어모델을 학습시켜서 말하는 모델 만들기
  - Many to many rnn-model 로



Example 3. 문자 기반 신경 언어 모델(many to many RNN)

- Dataset X_ohe, y_ohe 생성

```
text    Sing a song of sixpence, a pocket full of rye, Four and twen
```

Sequence text (399,10)

1234567890	1234567890
Sing a son	ing a song
ing a song	ng a song
ng a song	g a song o
g a song o	a song of
X	Y

```
sequence (399,10)
  =[[8, 19, 23, 17, 0, 11, 0, 28, 24,  23 ]
    [19, 23, 17, 0, 11, 0, 28, 24, 23, 17]
    [ 23, 17, 0, 11, 0, 28, 24, 23, 17, 0]
    …
   ]
```

```
X (398,10)
  =[[8, 19, 23, 17, 0, 11, 0, 28, 24,  23 ]
    [19, 23, 17, 0, 11, 0, 28, 24, 23, 17]
    …
   ]
```

```
Y (398,10)
  =[[19, 23, 17, 0, 11, 0, 28, 24,  23 17]
    [ 23, 17, 0, 11, 0, 28, 24, 23, 17,0]
    …
   ]
```

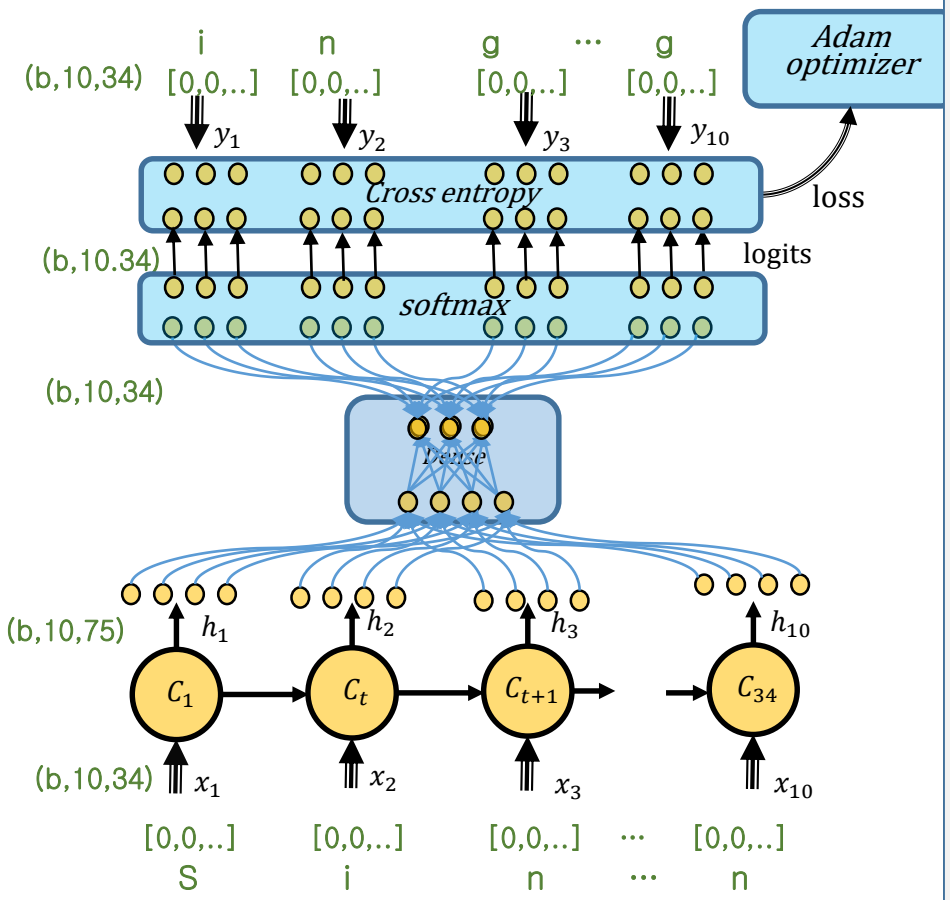
```
X_ohe (398,10,34)
  =[[[0 0 0 0 1 0..], [..],m…[…]]
    [[0 0 0 0 1 0..], [..],m…[…]]
    …
   ]
```

```
Y_ohe (398,10,34)
  =[[[0 0 0 0 1 0..], [..],m…[…]]
    [[0 0 0 0 1 0..], [..],m…[…]]
    …
   ]
```

```
with  open('data/rnn_sixpence_data.txt','r') as f:
      raw_text= f.read()
      #Sing a song of sixpence, a pocket full of rye,
      #Four and twenty blackbirds baked in a pie.
      #...
text = raw_text.split()   #strip all of the new line characters , separated only by white space
text = ' '.join(text)
    #['Sing', 'a', 'song', 'of', 'sixpence,', 'a', 'pocket', 'full', 'of', 'rye,', 'Four', 'and', 'twenty'
    #Sing a song of sixpence, a pocket full of rye, Four and twenty

alphabet = sorted(list(set(text))) #extract alphabet
vocab_size = len(alphabet)        #34
    #[' ', '!', '"', ',', '.', '?', 'F', 'O', 'S', 'T', 'W', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'k', 'l', 'm', 'n', 'o',
'p', 'q', 'r', 's', 't', 'u', 'w', 'x', 'y']
mapping_c2i = {c:i for i, c in enumerate(alphabet)}
    #{' ': 0, '!': 1, '"': 2, ',': 3, '.': 4, '?': 5, 'F': 6, 'O': 7, 'S': 8, 'T': 9, 'W': 10, 'a': 11, …,'y': 33}
text_encoded = [mapping_c2i[c] for c in text]          #(399,11)
    #[8, 19, 23, 17, 0, 11, 0, 28, 24, 23, 17, 0, 24, 16, 0, 28, 19, 32, 25, 15, 23, 13,

length = 10
sequence=np.array([text_encoded[i:i+length] for i in range(0, len(text_encoded)-
length)]) #399,10

X,Y=sequence[:-1],sequence[1:]                          #(399,10),(399,10)
X_ohe =to_categorical(X, num_classes=vocab_size)#(399,10,34)
Y_ohe = to_categorical(Y, num_classes=vocab_size)#(399,10,34)
```

Example 3. 문자 기반 신경 언

- Model 생성




```
import numpy as np
from pickle import dump
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import Dense,LSTM

#데이터 생성
with  open('data/rnn_sixpence_data.txt','r') as f:
raw_text= f.read()
text = raw_text.split() #['Sing', 'a', 'song', 'of', 'sixpence,', 'a', 'pocket', 'full', 'of', 'rye,', 'Four', 'and'
text = ' '.join(text)      #Sing a song of sixpence, a pocket full of rye, Four and twenty blackbirds ...\
alphabet = sorted(list(set(text)))
vocab_size = len(alphabet) #34
    #[' ', '!', '"', ',', '.', '?', 'F', 'O', 'S', 'T', 'W', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'k', 'l', 'm', 'n', 'o',  … 'y']
mapping_c2i = {c:i for i, c in enumerate(alphabet)}
text_encoded = [mapping_c2i[c] for c in text]
length = 10
sequence=np.array([text_encoded[i:i+length] for i in range(0, len(text_encoded)-length)]) #399,10
X,Y=sequence[:-1],sequence[1:]                          #(399,10),(399,10)
X_ohe =to_categorical(X, num_classes=vocab_size)#(399,10,34)
Y_ohe = to_categorical(Y, num_classes=vocab_size)#(399,10,34)

#모델생성  may to may rnn lstm
model = Sequential(name="may-to-many RNN")
Model.add(LSTM(75,
              input_shape=(X_ohe.shape[1], X_ohe.shape[2]),
              return_sequences=True))    #(none,10,75)
Model.add(Dense(vocab_size,
              activation='softmax'))          #(None,10,34)
print(model.summary())
Model.compile(loss='categorical_crossentropy',
              optimizer='adam', metrics=['accuracy'])

#모델학습 및 저장
Model.fit(X_ohe, Y_ohe, epochs=500, verbose=2)
 model.save('model_many2many_sixpence.h5')
dump(mapping_c2i, open('mapping.pkl', 'wb'))
```

```
Model: "many-to-many RNN with sixpense"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 10, 75)            33000
_________________________________________________________________
dense_1 (Dense)              (None, 10, 34)            2584
=================================================================
Total params: 35,584
Trainable params: 35,584
Non-trainable params: 0
_________________________________________________________________
 - 0s - loss: 0.3169 - accuracy: 0.8658
Epoch 496/500
 - 0s - loss: 0.3164 - accuracy: 0.8678
Epoch 497/500
 - 0s - loss: 0.3173 - accuracy: 0.8686
Epoch 498/500
 - 0s - loss: 0.3164 - accuracy: 0.8676
Epoch 499/500
 - 0s - loss: 0.3163 - accuracy: 0.8688
Epoch 500/500
 - 0s - loss: 0.3165 - accuracy: 0.8681
```

Example 3. 문자 기반 신경 언어 모델(many to many RNN)(cont.)

- 모델 평가
  - 모델을 load하고
  - 모델에게 seedtext를 제시하고
  - 문자열(20개)를 생성하게 한다.
- 실험
  - 학습된 시작부분 텍스트로부터
    20개의 문자의 텍스트(연속문자열) 생성시도
  - 학습된 중간 부분 텍스트로부터
    20개문자의 텍스트 생성 시도
  - 학습된 않은 텍스트로부터
    20개문자의 텍스트 생성시도

```
from pickle import load
from keras.models import load_model
from keras.utils import to_categorical
from keras.preprocessing.sequence import pad_sequences

# generate a sequence of nb characters with a language model
def generate_seq(model,seq_length, seed_text='Sing a son', nb_chars=20):
    in_text = seed_text
    for _ in range(nb_chars):
        encoded = [mapping_c2i[char] for char in in_text]                    #(n,)     Sing a son   => [8,19,...]
        encoded = pad_sequences([encoded], maxlen=seq_length, truncating='pre')#(1,10)   [[ 8, 19, 23, 1..., 24, 23]]
        encoded_ohe = to_categorical(encoded, num_classes=len(mapping_c2i))   #(1,10,34)
        yhat = model.predict_classes(encoded_ohe, verbose=0)                 #[[19, 23, 17,  0, 11,  0, 28, 24, 23, 17]] (1,10)
        yhatc=yhat[0,-1]                                                     #17
        char=mapping_i2c[yhatc]                                              #'g'
        in_text += char
    return in_text

# load the model and mapping
model = load_model('model_rnn_many2many_sixpence.h5')
mapping_c2i = load(open('mapping.pkl', 'rb'))          #{' ':0,...}
mapping_i2c={v:k for k,v in mapping_c2i.items()}   #{0:' ',...}

# test start, mid-line of rhyme, not in origin
print(generate_seq(model,10, 'Sing a son', 20))        #Sing a song of sixpence, a poc
print(generate_seq(model, 10, 'king was i', 20))       #king was in his counting out h
print(generate_seq(model,10, 'hello worl', 20))        #hello worl When down came a bl
print(generate_seq(model,10, 'hello Sing', 20))        #hello Sing a song of sixpence,
print(generate_seq(model,10, 'hello model ', 20))      #hello model of rye, Four and twe
```

Example 3. 문자 기반 신경 언어 모델(many to many RNN)(cont.)

- 비교 검토

Sing a song of sixpence, a pocket full of rye,
Four and twenty blackbirds baked in a pie.
When the pie was opened the birds began to sing,
Oh wasn't that a dainty dish to set before the king?

The king was in his counting house counting out his money,
The queen was in the parlour eating bread and honey
The maid was in the garden hanging out the clothes,
When down came a blackbird and pecked off her nose!

```
#many to one RNN
print(generate_seq(model,10, 'Sing a son', 20))    #Sing a song of sixpence, a poc
print(generate_seq(model, 10, 'king was i', 20))   #king was in his counting house
print(generate_seq(model,10, 'hello worl', 20))    #hello worl,, The aeen was in t
```

```
#many to many RNN
print(generate_seq(model,10, 'Sing a son', 20))        #Sing a song of sixpence, a poc
print(generate_seq(model, 10, 'king was i', 20))       #king was in his counting out h
print(generate_seq(model,10, 'hello worl', 20))        #hello worl When down came a bl
print(generate_seq(model,10, 'hello Sing', 20))        #hello Sing a song of sixpence,
print(generate_seq(model,10, 'hello model ', 20))      #hello model of rye, Four and twe
```

Example 4. 주가예측 모델

- 문제
  - GE의주가를 이용하여 모델을 학습하고 주가를 예측한다.
- 데이터셋 생성
  - 미국 General Electronics의 주가 를 로드 한다.
    - us.ge.txt 을 다운로드한다.
  - 정규화 한다
  - 학습 및 평가용 데이터 셋을 구성
- 주가예측 모델 생성
  - 모델 구성 및 학습
- 모델 평가
- 주가 데이터 분석방법

Example 4. 주가예측 모델

- 데이터셋 생성, X,y

```
#주가데이터 로드
df_ge = pandas.read_csv("data\\us.ge.txt", engine='python')  #주가데이터 로드
print(df_ge)

#시가 고가 저가 종가 거래량 추출
Train_cols = [ " Open " , " High " , " Low " , " Close " , " Volume " ]
stock=df_ge.loc[:,train_cols].values

#데이터 주가 정규화(0.0~1.0)
Min_max_scaler = MinMaxScaler()
stock = min_max_scaler.fit_transform(stock)

#학습 및 평가 데이터 셋 구성
TIME_STEPS=3
 X=[]; y=[]
for i in range(0,stock.shape[0]-TIME_STEPS-1,1) :
            X.append(stock[i:i+TIME_STEPS])          #["Open","High","Low","Close","Volume"
            y.append(stock[i+TIME_STEPS,Close_idx])#"Close“
X=np.array(X)          #(14054,3,5)
y=np.array(y)          #(14054,)
 X_train, X_val, y_train, y_val = train_test_split(X, y, train_size=0.8, test_size=0.2,
      shuffle=False)
#(11235,3,5) (2811,3,5),  (11235,) (2811,)
```

```
         Date     Open     High      Low    Close     Volume  OpenInt
0  1962-01-02   0.6277   0.6362   0.6201   0.6201    2575579        0
1  1962-01-03   0.6201   0.6201   0.6122   0.6201    1764749        0
2  1962-01-04   0.6201   0.6201   0.6037   0.6122    2194010        0
3  1962-01-05   0.6122   0.6122   0.5798   0.5957    3255244        0
```

```
[[0.00356678 0.00352766 0.00358385 0.00338425 0.02108267]
 [0.00340607 0.00319219 0.00341628 0.00338425 0.01365459]
 [0.00340607 0.00319219 0.00323598 0.00321827 0.01758709]]
```

```
Stock (14045,5)

[[11  12  13  14 15]
 [21  22  23  24 25]
 [31  32  33  34 35]
 [41  42  43  44 45]
 [51  52  53  54 55]
 [61  62  63  64 65]
 [71  72  73  74 75]
 [81  82  83  84 85]
 [91  92  93  94 95]
 ...
]
```

```
X (14045,3,5)

[ [ [11 12 13 14 15]  [21 22 23 24 25]  [31 32 33 34 35] ]
  [ [21 22 23 24 25]  [31 32 33 34 35]  [41 42 43 44 45] ]
  [ [31 32 33 34 35]  [41 42 43 44 45]  [51 52 53 54 55] ]
  [ [41 42 43 44 45]  [51 52 53 54 55]  [61 62 63 64 65] ]
  [ [51 52 53 54 55]  [61 62 63 64 65]  [71 72 73 74 75] ]
  [ [61 62 63 64 65]  [71 72 73 74 75]  [81 82 83 84 85] ]
  [ [71 72 73 74 75]  [81 82 83 84 85]  [91 92 93 94 95] ]
 ...
]
```

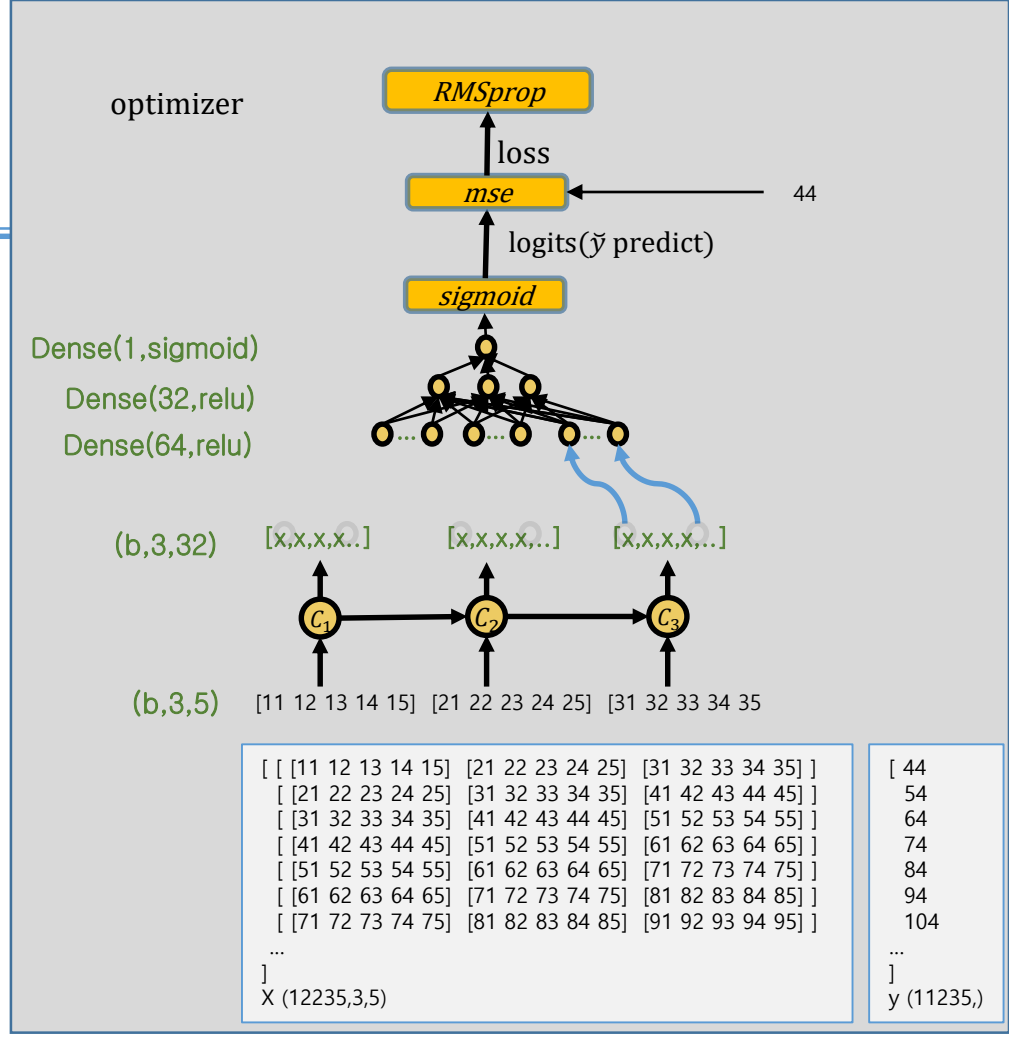
```
y (14045)

[ 44
  54
  64
  74
  84
  94
  104
...
]
```

https://

Example 4. 주가예측 모델

- 모델 생성 및 학습

```
#모델생성
model = Sequential(name='주식예측모델')
model.add(LSTM(32, input_shape=(X_train.shape[1],X_train.shape[2])))
model.add(Dropout(0.2))
model.add(Dense(64,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(32,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1),activation='sigmoid'))

optimizer = keras.optimizers.RMSprop(lr=0.001)
model.compile(loss='mean_squared_error', optimizer=optimizer)

# 모델 학습,  logging file(.log)
csv_logger = keras.callbacks.CSVLogger('logs/stock_model.log'), append=True)
history = model.fit(X_train,y_train,              #train data
          epochs=300,
          batch_size=1204,                        #
          verbose=2,                              # set output amount
          shuffle=False,                          #no shuffle
          validation_data=(X_val,y_val),          #evaluation data
          callbacks=[csv_logger])                 #set logging callback
```



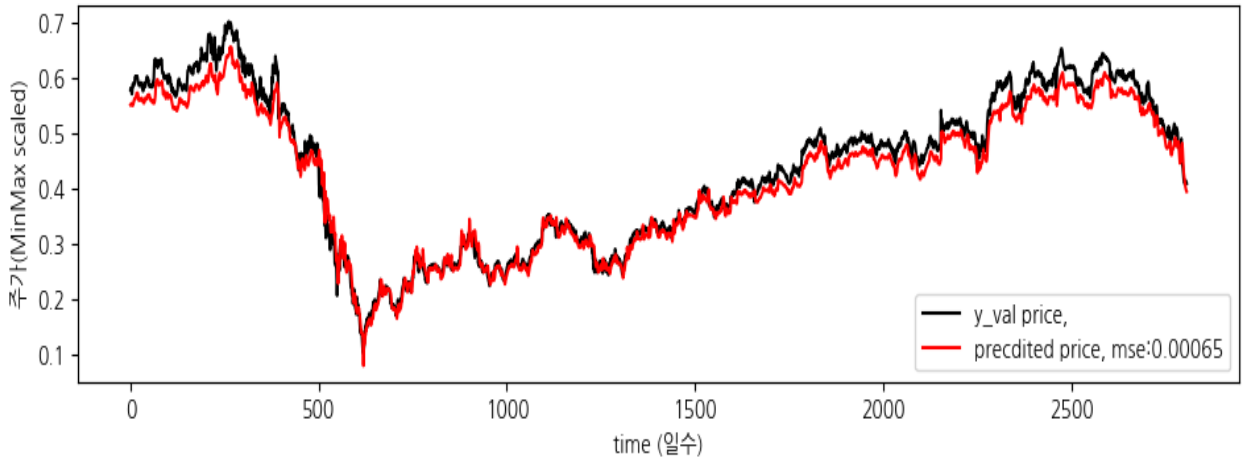

```
Epoch 293/300
 - 0s - loss: 5.1985e-04 - val_loss: 0.0010
Epoch 294/300
 - 0s - loss: 7.1853e-04 - val_loss: 0.0013
Epoch 295/300
 - 0s - loss: 5.7094e-04 - val_loss: 4.2705e-04
Epoch 296/300
 - 0s - loss: 5.1088e-04 - val_loss: 3.8838e-04
Epoch 297/300
 - 0s - loss: 5.0382e-04 - val_loss: 6.6422e-04
Epoch 298/300
 - 0s - loss: 5.9464e-04 - val_loss: 0.0014
Epoch 299/300
 - 0s - loss: 5.9211e-04 - val_loss: 7.3366e-04
Epoch 300/300
 - 0s - loss: 5.3272e-04 - val_loss: 6.5282e-04
```

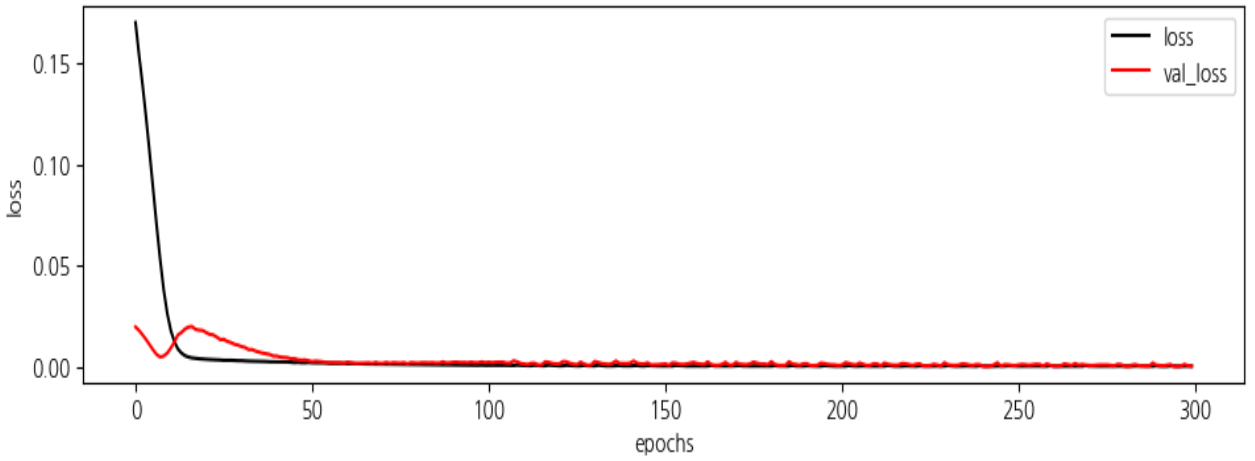
Example 4-3. 주가예측 모델 (cont.)

- 모델 검증

```
y_hat  =model.predict(X_val)        #(2808,3,5)=>(2808,1)
y_hat1=np.reshape(y_hat,(-1))       #((2808,)

score_tr=model.evaluate(X_train,y_train) #(11242,3,5),(11242,)=>0.00047
score   =model.evaluate(X_val,y_val)     #(2808,3,5), (2808,)     =>0.00105

print('evaluation score (x_train,y_train ) : ',score_tr)
print('evaluation score (x_val,  y_val)    : ',score)

plt.subplot(211)
plt.plot(y_val,  'k',label='real price')
plt.plot(y_hat1,'r',label='precdited price, mse:{}'.format(score))
plt.legend()

plt.subplot(212)
plt.plot(history.history['loss'],     'k', label='loss')
plt.plot(history.history['val_loss'],'r', label='val_loss')
plt.legend()
plt.show()
```

```
evaluation score (x_train,y_train ) :  0.0003012311564353952
evaluation score (x_val,  y_val)    :  0.0006528214721985564
```





Example 3. 주가예측 모델 (cont.)

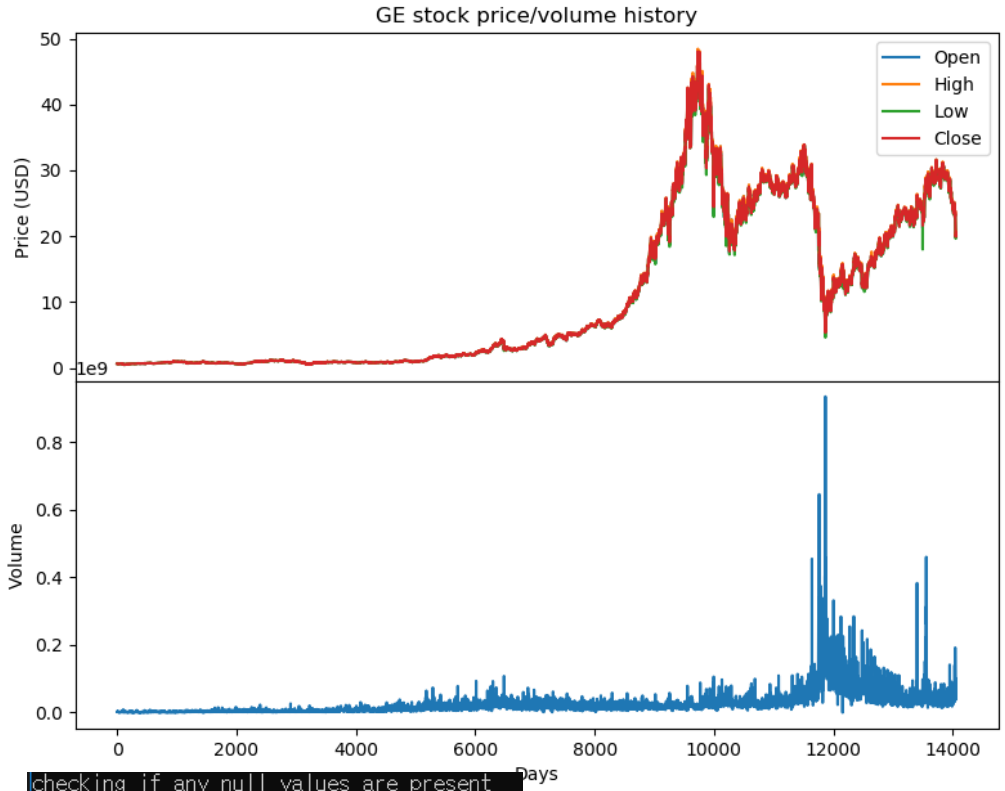
- 주가 데이터 분석
  - **General Electronic ‘corp. us.ge.txt’**

```
import pandas as pd
df_ge = pd.read_csv("data\\us.ge.txt", engine='python')
print(df_ge))   #(14058,7)
```

```
def plot_stock() :
        fig,axes=plt.subplots(2,1,sharex=True)
        axes[0].plot(df_ge["Open"],label='Open’)
        axes[0].plot(df_ge["High"],label='High’)
        axes[0].plot(df_ge["Low"],label='Low’)
        axes[0].plot(df_ge["Close"],label='Close’)
        axes[0].set_ylabel('Price (USD)’)
        axes[0].set_xlabel('Days’)
        axes[0].legend()

        axes[1].plot(df_ge["Volume"])
        axes[1].set_ylabel('Volume’)
        axes[1].set_xlabel('Days’)
        plt.subplots_adjust(hspace=0)
        axes[0].title.set_text('GE stock price/volume history’)
        plt.show()
plot_stock()
```

```
print("checking if any null values are present\n", df_ge.isna().sum())
```

	Date	Open	High	Low	Close	Volume	OpenInt
0	1962-01-02	0.6277	0.6362	0.6201	0.6201	2575579	0
1	1962-01-03	0.6201	0.6201	0.6122	0.6201	1764749	0
2	1962-01-04	0.6201	0.6201	0.6037	0.6122	2194010	0
3	1962-01-05	0.6122	0.6122	0.5798	0.5957	3255244	0




```
checking if any null values are present
 Date        0
Open         0
High         0
Low          0
Close        0
Volume       0
OpenInt      0
dtype: int64
```

Example 4-3. 주가예측 모델 (cont.)

- 주식의 일봉 주가 가져오기
  - 미국 종목코드는 알파벳이름으로, 우리나라 종목은 6자리 숫자 코드로 지정한다.
  - 종목코드
    - 마이크로 소프트: 'msft',
      코스피 상장사 삼성전자: '005930.KS'
      코스닥 상장사
  - 주가 호출 형식
    - import pandas_datareader as pdr
      pdr.get_data_yahoo('종목 코드', start='시작 시점', end='종료 시점')
  - 예

```
import pandas_datareader as pdr
#pdr.get_data_yahoo('종목 코드', start='시작 시점', end='종료 시점')
msft = pdr.get_data_yahoo('msft')
print('Micosoft 주가\n',msft)
sm=pdr.get_data_yahoo('005930.KS', start='2011-08-19', end='2020-01-23')
print('삼성전자 주가\n',sm)
sh=pdr.get_data_yahoo('028300.KQ')
print('에이치엘비 주가\n',sh)
```

```
Micosoft 주가
                 High         Low  ...      Volume   Adj Close
Date                               ...
2015-05-11  47.910000   47.369999  ...  24609400.0   42.712799
2015-05-12  47.680000   46.419998  ...  29928300.0   42.694778
2015-05-13  48.320000   47.570000  ...  34184600.0   42.947243
2015-05-14  48.820000   48.029999  ...  32980900.0   43.930080
2015-05-15  48.910000   48.049999  ...  28642700.0   43.551365
...               ...         ...  ...         ...         ...
2020-05-04 179.000000  173.800003  ...  30372900.0  178.839996
2020-05-05 183.649994  179.899994  ...  36839200.0  180.759995
2020-05-06 184.199997  181.630005  ...  32139300.0  182.539993
2020-05-07 184.550003  182.580002  ...  28316000.0  183.600006
2020-05-08 185.000000  183.360001  ...  30877800.0  184.679993

[1259 rows x 6 columns]
삼성전자 주가
              High      Low     Open    Close      Volume     Adj Close
Date
2011-08-19  13940.0  13440.0  13760.0  13600.0  28970600.0     10.083615
2011-08-22  14040.0  13660.0  13680.0  13820.0  20900650.0     10.246735
2011-08-23  14700.0  13980.0  14000.0  14460.0  24617300.0     10.721255
2011-08-24  14700.0  14000.0  14480.0  14160.0  18508650.0     10.498822
2011-08-25  14760.0  14420.0  14740.0  14500.0  19039800.0     10.750915
...             ...      ...      ...      ...         ...           ...
2020-01-17  62000.0  61000.0  61900.0  61300.0  16025661.0  61300.000000
2020-01-20  62800.0  61700.0  62000.0  62400.0  12528855.0  62400.000000
2020-01-21  62400.0  61200.0  62000.0  61400.0  11142693.0  61400.000000
2020-01-22  62600.0  60400.0  60500.0  62300.0  15339565.0  62300.000000
2020-01-23  61800.0  60700.0  61800.0  60800.0  14916555.0  60800.000000

[2069 rows x 6 columns]
에이치엘비 주가
              High      Low     Open     Close     Volume  Adj Close
Date
2015-05-11  22400.0  20500.0  20750.0  22400.0  1514799.0    22400.0
2015-05-12  24500.0  22200.0  22600.0  23850.0  1441091.0    23850.0
2015-05-13  24200.0  22900.0  24000.0  23300.0   906043.0    23300.0
2015-05-14  25000.0  22700.0  23400.0  24350.0   949096.0    24350.0
2015-05-15  25600.0  23750.0  24300.0  25400.0   657592.0    25400.0
...             ...      ...      ...      ...        ...        ...
2018-01-26  50400.0  48300.0  49850.0  48450.0   826790.0    48450.0
2018-01-29  48700.0  46100.0  48550.0  46200.0  1188756.0    46200.0
2018-01-30  47300.0  44750.0  45400.0  45150.0   975391.0    45150.0
2018-01-31  47100.0  43700.0  45300.0  46500.0   960708.0    46500.0
2020-05-08 101300.0  98100.0  98100.0 100400.0   510047.0   100400.0

[670 rows x 6 columns]
Press any key to continue
```

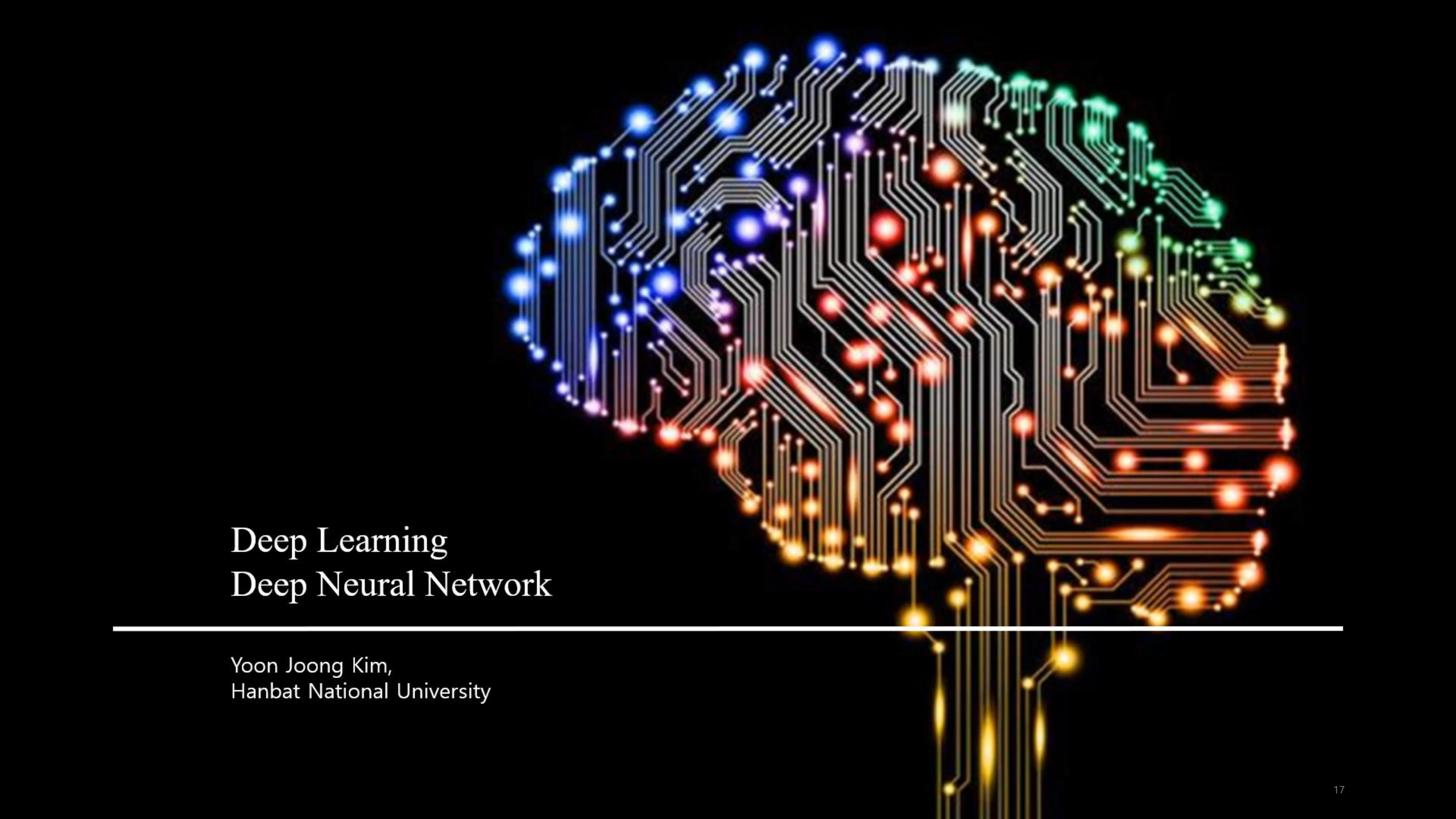
Example 4-3. 주가예측 모델 (cont.)

- 주식 종목코드

```
def get_code(name) :
    df = pd.read_html('http://kind.krx.co.kr/corpgeneral/corpList.do?method=download', header=0)[0]
    df_회사명코드=df[['회사명','종목코드']]                    #회사명,종목코드 만 추출 =>df
    회사명2코드={n:c for n,c in df_회사명코드.values} #리스트형으로 변환 후 dict형으로 로 변환
    code='{:06d}'.format(회사명2코드[name])
    return code
Print(get_code('삼성전자'))  #005930
```

Agenda

1. RNNs
    1. From feed-forward to RNNs
    2. Simple Recurrent Neural Network (SRNN)
    3. RNNs in the context of NLP
    4. The problem with RNNs
    5. LSTM
2. RNN Applications
    1. Language Modeling
    2. Character-level Language Modeling
    3. Neural Machine Translation(Google Research's blog)
    4. Text Summarization
    5. Image Captioning
3. Language Modeling
    1. Language Modeling DEMO Character-level, Language Modeling
4. RNN models
5. Examples in Keras
    1. 정현파신호 샘플의 예측 모델(SimpleRNN) in keras
    2. 문자 기반 신경 언어 모델(many-to-one RNN)-sixpence in keras
    3. 문자 기반 신경 언어 모델(many-to-many RNN)-sixpence in keras
    4. 주가예측 모델



Deep Learning
Deep Neural Network

Yoon Joong Kim,
Hanbat National University