인공지능 기본 저녁반 스터디
머신러닝 모델 만들기

주식 데이터를 활용한 주가 예측 모델 만들기

Dong jo Lim / Yu rim Choi

2022. 05. 28.

CONTENTS

주제 선정 및 개요

"삼성전자 국내 주주 1천만명 시대, 머신러닝으로 주가는 예측할 수 없을까?"

주제 선정 및 개요

"되든 안 되든, 주가 예측 모델을 구현해보자"

수업을 통해 배운대로 데이터를 수집 / 분석 / 모델 구현을 해보는 도전

데이터 수집

Kaggle
Yahoo
Naver
Youtube

데이터 분석

데이터 확인
결측치 처리
데이터 정규화

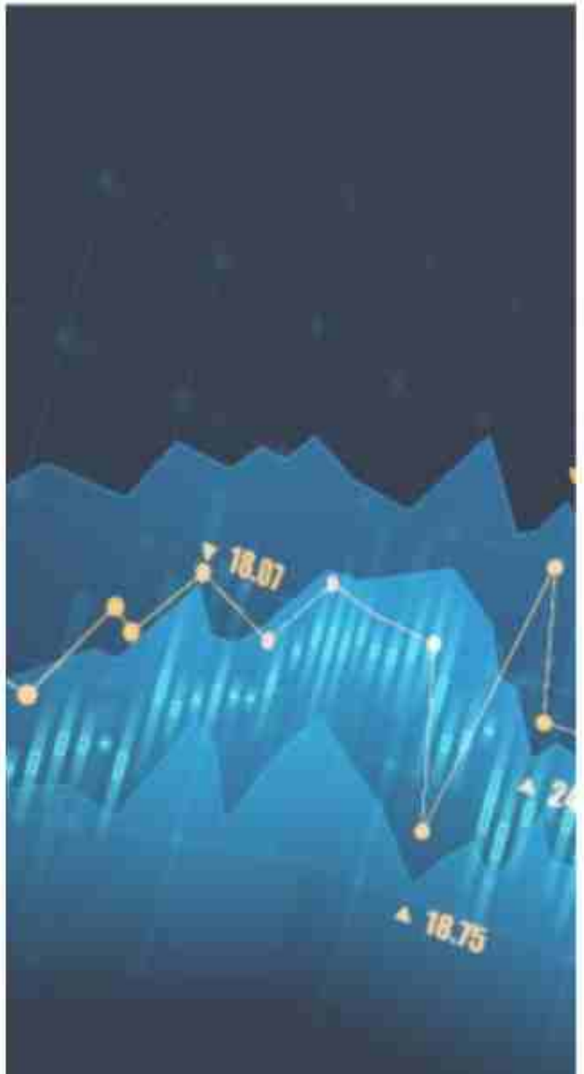
머신러닝 모델 구현

first model

정확도 높이기

add model

LSTM
Data Shuffle
Simple RNN



STEP1. 데이터 수집

데이터 셋 구하기 : kaggle

https://www.kaggle.com/datasets

/caesarmario/samsung-electronics-stock-historical-price

데이터 파일 불러오기 - 데이터 확인

* 005930.KS.csv : 삼성전자(005930)의 daily 주가
* 2019년 6월 ~ 2022년 4월 (724개의 Data)
* Date : 날짜 (yyyy-mm-dd)
* Open : 시가
* High : 고가
* Low : 저가
* Close : 종가
* Adj Close : 조정종가
* Volume : 거래량

In [2]:
```
stock_data = pd.read_csv("005930.KS.csv")
data = stock_data
data.shape
```

Out[2]:
```
(724, 7)
```

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 724 entries, 0 to 723
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       724 non-null    object
 1   Open       724 non-null    float64
 2   High       724 non-null    float64
 3   Low        724 non-null    float64
 4   Close      724 non-null    float64
 5   Adj Close  724 non-null    float64
 6   Volume     724 non-null    int64
dtypes: float64(5), int64(1), object(1)
memory usage: 39.7+ KB
```

In [4]:
```
data.head()
```

Out[4]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2019-06-03	42950.0	43900.0	42500.0	43800.0	40043.343750	15466580
1	2019-06-04	43400.0	43700.0	43000.0	43450.0	39723.363281	9913497

STEP2. 데이터 분석 - 전처리

결측치 처리 : isna / isnull / drop / dropna

isna( ) / isnull( ) 을 통해 결측치를 확인

drop( ) / dropna( )를 통해 결측치를 제거

정규화 작업 Normalization

- 전체 주가의 분포는 중간중간 튀는 데이터가 있을 수 있음
- 정규화는 모든 데이터를 0과 1 사이로 정렬시켜서 모델이 더 예측을 잘 하게 하는 과정
- 정규화를 끝낸 데이터는 result에 저장
- 이 중 90%를 train data로 나머지 10%는 test data로 설정

In [14]:
```
### Normalize Data 데이터 정규화작업

normalized_data = []
normalized_data_price = []

for window in result:
  normalized_window = [((float(p) / float(window[0])) for p in window]

  normalized_data.append(normalized_window)

  normalized_data_price.append(normalized_data_price)

result = np.array(normalized_data)        #result = np.array(normalized_data_price)

row = int(round(result.shape[0] * 0.9))  # split train(90%) and test(10%) data
train = result[:row, :]

X_train = train[:, :-1]                   # 앞에서 부터 50개
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
y_train = train[:, -1]                    # 맨 뒤 1개

X_test = result[row:, :-1]                # 50개
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
y_test = result[row:, -1]                 # 1개

X_train.shape, X_test.shape
```

Out[14]: ((606, 50, 1), (67, 50, 1))

STEP3. 모델 구현

RNN 구조

- 데이터를 연속적으로 읽어 다음 데이터를 파악

LSTM (Long Short Term Memory) layer를 사용하여 모델 정의

- keras.layers.LSTM( )
- 장/단기 기억 가능(주로 시계열처리, 자연어처리에 사용)

In [7]:
```
seq_len = 50
sequence_length = seq_len + 1

result = []
for index in range(len(mid_price) - sequence_length):
  result.append(mid_price[index: index + sequence_length])

## 데이터를 연속적으로 읽어 다음 데이터를 파악하는 RNN 구조의 데이터 만들어주기
## cnn과일의 최근 50일 동안의 데이터를 보고 다음 날의 주식 가격을 예측해본다.
```

In [15]:
```
model = Sequential()

model.add(LSTM(50, return_sequences=True, input_shape=(50, 1)))

model.add(LSTM(64, return_sequences=False))

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

model.compile(loss = 'mse', optimizer = 'rmsprop')

model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 50, 50)            10400
_________________________________________________________________
lstm_3 (LSTM)                (None, 64)                29440
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 39,905
Trainable params: 39,905
Non-trainable params: 0
_________________________________________________________________
```

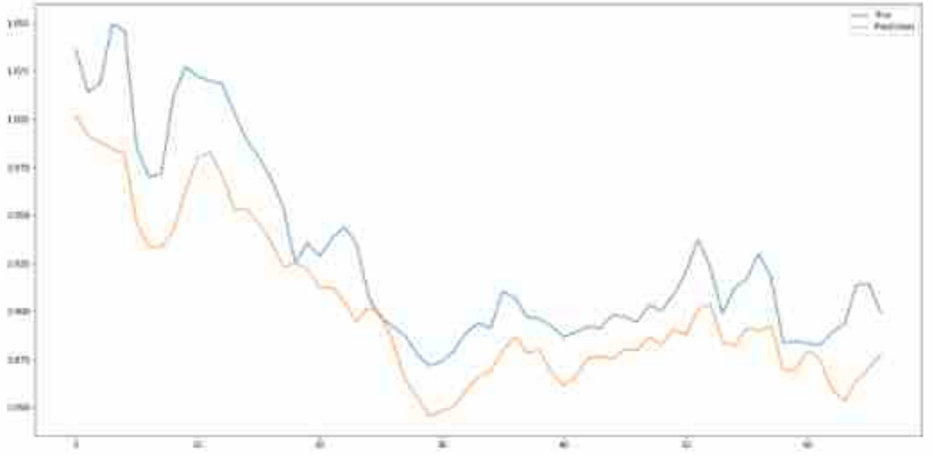
STEP3. 모델 구현

Model Train 모델 트레이닝 시키기

Prediction 예측하기

시각화 - [그래프 그리기]

- 파란색 : 실제 데이터
- 주황색 : 예측 데이터
- 파란색과 주황색 선이 잘 맞을수록 예측을 잘 한 것인데, 많이 겹치지 않음.



In [16]:
```
model.fit(X_train, y_train,
          validation_data= (X_test, y_test),
          batch_size=20,
          epochs = 20)
```

```
Epoch 1/20
31/31 [==============================] - 2s 77ms/step - loss: 0.0963 - val_loss: 4.9
Epoch 2/20
31/31 [==============================] - 1s 41ms/step - loss: 0.0175 - val_loss: 3.0
Epoch 3/20
31/31 [==============================] - 1s 38ms/step - loss: 0.0110 - val_loss: 0.0
31/31 [==============================] - 1s 46ms/step - loss: 0.0030 - val_loss: 0.0
Epoch 19/20
31/31 [==============================] - 1s 45ms/step - loss: 0.0039 - val_loss: 5.0
Epoch 20/20
31/31 [==============================] - 1s 39ms/step - loss: 0.0035 - val_loss: 0.0
```

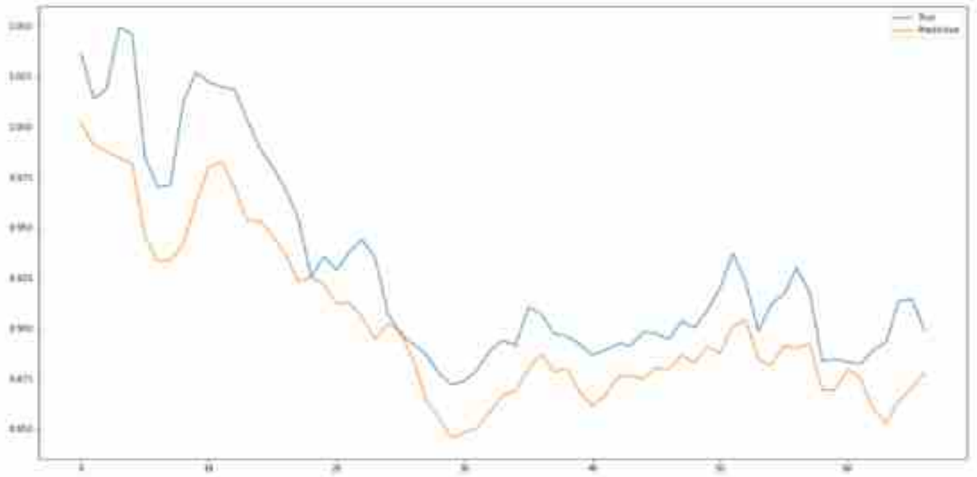
Out[15]:
```
<tensorflow.python.keras.callbacks.History at 0x202605dfe80>
```

In [17]:
```
# test data 예측
pred = model.predict(X_test)

# 그래프 그리기
fig = plt.figure(facecolor='white', figsize=(20, 10))
ax = fig.add_subplot(111)
ax.plot(y_test, label='True')
ax.plot(pred, label='Prediction')
ax.legend()
plt.show()
```

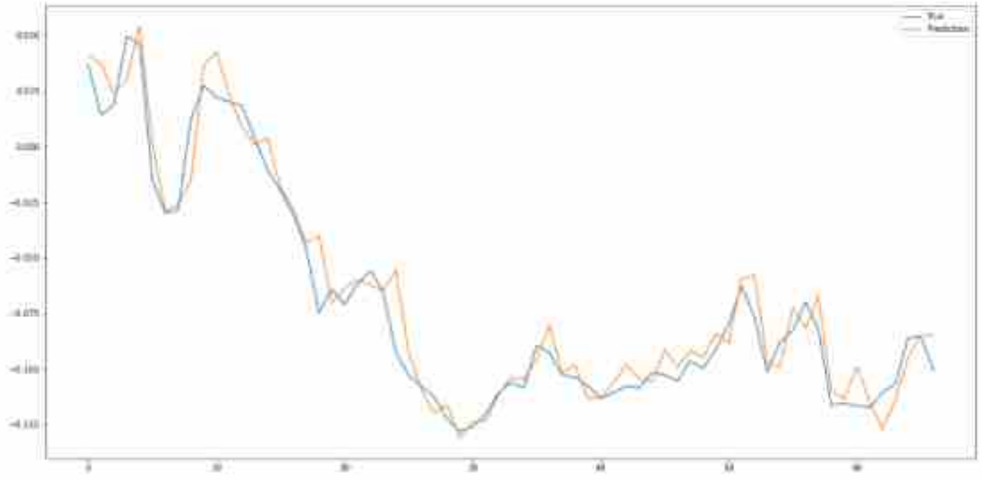
STEP3. 모델 구현 - add (1)

Shuffle 추가 `np.random.shuffle(train)`

- 데이터가 순서대로 들어가게 되면 모델이 데이터에 익숙해지기 때문에 train data를 섞어주었다.



Shuffle 전

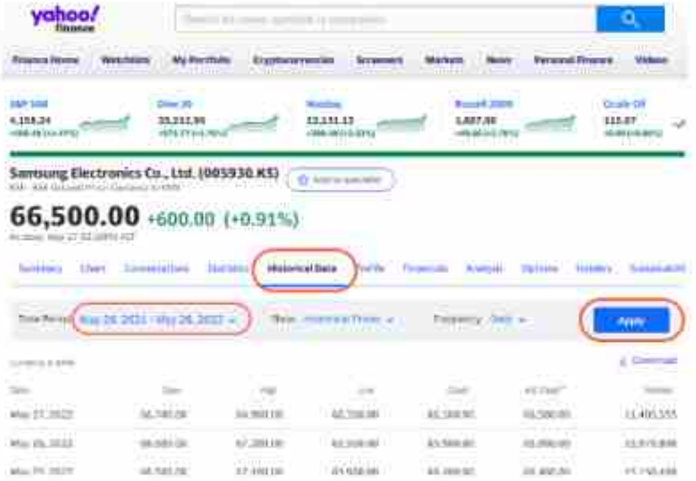


Shuffle 후

STEP3. 모델 구현 - add(2)

DataSet 추가

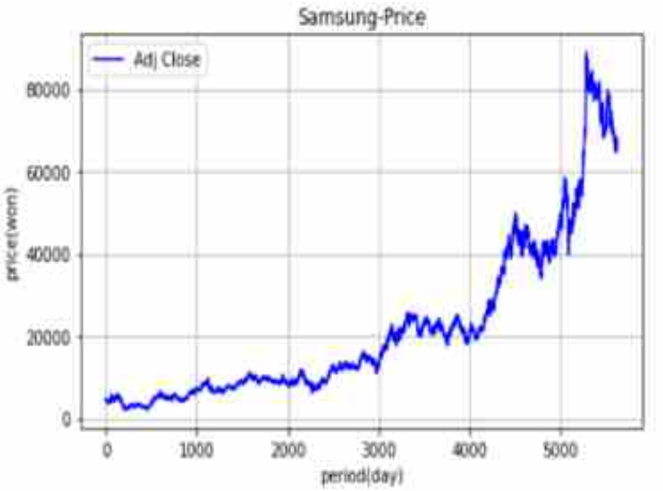
- 기존 데이터는 724개로 데이터 양이 적어 데이터를 추가로 수집
- 야후 파이넨셜에서 삼성전자 주가를 기간별로 검색 후 .csv로 다운로드
- 2000년 1월 ~ 2022년 5월 (5,623개)



데이터 확인 (시각화)

- 수정종가 기준
- 전체 기간(2000년 1월 ~ 2022년 5월) 주가의 상승 / 하락 시각화

```
[ ]  data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5623 entries, 0 to 5622
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       5623 non-null   object
 1   Open       5623 non-null   float64
 2   High       5623 non-null   float64
 3   Low        5623 non-null   float64
 4   Close      5623 non-null   float64
 5   Adj Close  5623 non-null   float64
 6   Volume     5623 non-null   int64
dtypes: float64(5), int64(1), object(1)
memory usage: 307.6+ KB
```



STEP3. 모델 구현 - add(2)

데이터 확인 (결측치)

- 거래량(Volume)을 보면 거래가 없는 날의 데이터도 포함되어 있음
- 비어있는 값인 결측치는 찾아서 평균값으로 결측치 처리를 해준다.
- 특이값(outlier)는 통계적으로 비정상적으로 크거나 작은 데이터를 말하는데,
  딥러닝에서는 이 특이값을 적절한 값으로 바꾸거나 삭제하는 등의 처리가 필요하다.

```
[ ]  # Volume값 0을 NaN으로 모두 대체(replace)
     data['Volume'] = data['Volume'].replace(0,np.nan)
```

```
[ ]  # 각 column에 0 개수 확인
     for col in data.columns:
       missing_rows = data.loc[data[col]==0].shape[0]
       print(col+': '+str(missing_rows))
```

```
[ ]  # 모든 Missing Value 삭제
     data.isnull().sum()
```

```
[ ]  data = data.dropna()
     data.isnull().sum()
```

```
[ ]  data.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	5623.000000	5623.000000	5623.000000	5623.000000	5623.000000	5.623000e+03
mean	24899.244176	25150.339676	24642.399075	24894.479815	21745.380880	2.176056e+07
std	20120.438963	20286.150852	19946.564133	20105.923452	19754.880059	1.544958e+07
min	2540.000000	2760.000000	2420.000000	2730.000000	2078.435791	0.000000e+00
25%	10170.000000	10320.000000	10020.000000	10160.000000	7788.425538	1.179838e+07
50%	16600.000000	16800.000000	16420.000000	16600.000000	13287.426758	1.773971e+07
75%	31340.000000	31620.000000	30990.000000	31360.000000	26651.802735	2.722040e+07
max	90300.000000	96800.000000	89500.000000	91000.000000	88908.171875	1.642150e+08

```
[ ]  data.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	5496.000000	5496.000000	5496.000000	5496.000000	5496.000000	5.496000e+03
mean	25122.851164	25379.748908	24860.070961	25117.976710	21937.418132	2.226339e+07
std	20058.700081	20225.865994	19883.153007	20043.857596	19699.549180	1.526464e+07
min	2540.000000	2760.000000	2420.000000	2730.000000	2078.435791	2.765000e+04
25%	10395.000000	10560.000000	10260.000000	10420.000000	8037.501831	1.211359e+07
50%	17040.000000	17180.000000	16820.000000	17040.000000	13709.213867	1.800640e+07
75%	31645.000000	32130.000000	31340.000000	31785.000000	27021.611816	2.757765e+07
max	90300.000000	96800.000000	89500.000000	91000.000000	88908.171875	1.642150e+08

STEP3. 모델 구현 - add(2)

데이터 전처리 (정규화 Normalization)

```
[ ]  from sklearn.preprocessing import MinMaxScaler

     scaler = MinMaxScaler()
     # 정규화 대상 column 정의
     scale_cols = ['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume']

     # 정규화 수행
     scaled_df = scaler.fit_transform(data[scale_cols])

     # 리턴값은 numpy
     print(type(scaled_df), '₩n')

     # 정규화된 새로운 DataFrame 생성
     scaled_df = pd.DataFrame(scaled_df, columns=scale_cols)
     print(scaled_df)
```

```
<class 'numpy.ndarray'>

        Open      High       Low     Close  Adj Close    Volume
0   0.039426  0.035623  0.037207  0.038292   0.029636  0.451724
1   0.037147  0.035091  0.035599  0.032287   0.024989  0.454678
2   0.036577  0.032114  0.036288  0.032740   0.025340  0.331100
3   0.034412  0.030944  0.033762  0.031834   0.024638  0.245313
4   0.034868  0.032008  0.036288  0.034440   0.026655  0.285359
...      ...       ...       ...       ...        ...       ...
```

데이터 전처리 (feature - label 정의)

- 딥러닝 학습을 위한 입력데이터 feature column, 정답데이터 label column 정의
- DataFrame -> Numpy로 변환

```
[ ]  # feature 정의(입력데이터)
     feature_cols = ['Open', 'High', 'Low', 'Close', 'Adj Close']
     # label 정의(정답데이터)
     label_cols = ['Adj Close']

     label_df = pd.DataFrame(scaled_df, columns=label_cols)
     feature_df = pd.DataFrame(scaled_df, columns=feature_cols)

     print(feature_df)
     print(label_df)

     # 딥러닝 학습을 위해 DataFrame -> numpy로 변환
     label_np = label_df.to_numpy()
     feature_np = feature_df.to_numpy()
```

```
        Open      High       Low     Close  Adj Close
0   0.039426  0.035623  0.037207  0.038292   0.029636
1   0.037147  0.035091  0.035599  0.032287   0.024989
2   0.036577  0.032114  0.036288  0.032740   0.025340
3   0.034412  0.030944  0.033762  0.031834   0.024638
4   0.034868  0.032008  0.036288  0.034440   0.026655
...      ...       ...       ...       ...        ...
```

STEP3. 모델 구현 - add(2)

[결과]

모델 구축

- LSTM 레이어로 모델 구축
- RNN 구조로 예측값 구하고 시각화

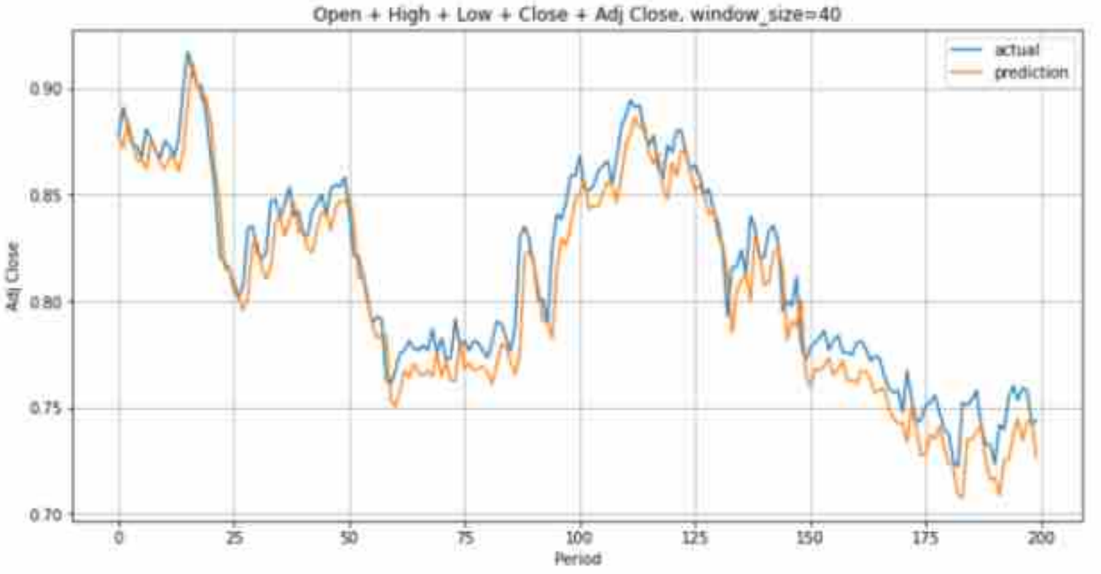
LSTM 모델 구축

```
[ ]  model = Sequential()
     # LSTM계층에 tanh를 활성화 함수로 가지는 노드 128개
     # input_shape = (40,3)
     model.add(LSTM(128,
                    activation='tanh',
                    input_shape=X_train[0].shape))

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

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_2 (LSTM)               (None, 128)               68608

 dense_2 (Dense)             (None, 1)                 129

=================================================================
Total params: 68,737
Trainable params: 68,737
Non-trainable params: 0
_________________________________________________________________
```

- SimpleRNN Example

```
pred = model.predict(X_test)

plt.figure(figsize=(12, 6))
plt.title('Open + High + Low + Close + Adj Close, window_size=40')
plt.ylabel('Adj Close')
plt.xlabel('Period')
plt.plot(y_test, label='actual')
plt.plot(pred, label='prediction')
plt.grid()
plt.legend(loc='best')

plt.show()
```



참고문헌

인공지능 기본반 교육 내용을 기본으로 진행하되
도서 / 블로그 / 유튜브 등 자료를 참고하였습니다.

[참고도서]

안드레아스 뮐러 / 세라 가이도 저(박해선 역) , "파이썬 라이브러리를 활용한 머신러닝" , 한빛미디어

프랑소와 숄레 저(박해선 역) , "케라스 창시자에게 배우는 딥러닝" , 길벗출판사

[데이터]

Kaggle 데이터셋 : https://www.kaggle.com/datasets

Yahoo 파이넨스 : https://finance.yahoo.com/

[기타 사이트]

ibovespa-stocks 브라질 주식 데이터 분석(강사님 Guihub) : https://github.com/LDJWJ/dataAnalysis

LSTM과 FinanceDataReader API를 활용한 삼성전자 주가 예측 : https://teddylee777.github.io/tensorflow/lstm-stock-forecast

Machine Learning을 이용한 간단한 주식 예측 : https://superhky.tistory.com/124

머신러닝을 활용한 주식 예측 : https://12soso12.tistory.com/84

LSTM을 활용한 삼성전자 주가예측 : https://blog.naver.com/beyondlegend/222500391711

주가 패턴 검색기로 향후 5일간 주가 예측하기 : https://www.youtube.com/watch?v=0MXDBYf0E60

감사합니다