미국 주가 추이 예측

Data-Science stock prediction

2019108078
신민호

CONTENTS

01

개요

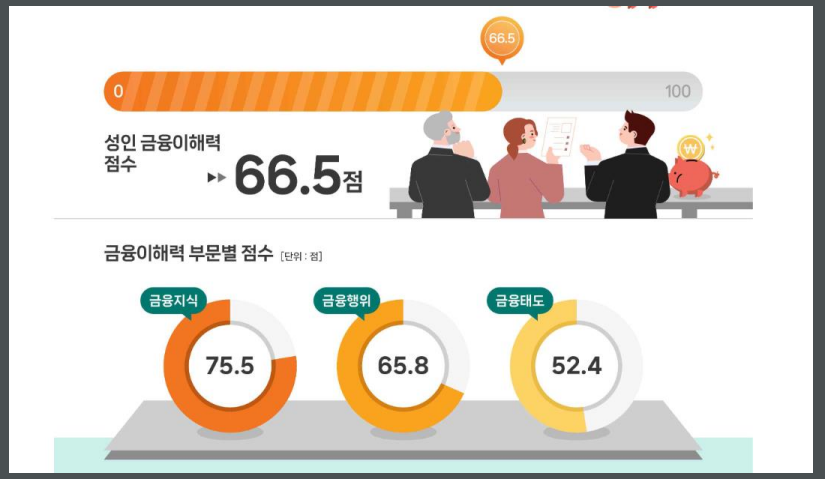
개요

[한국금융신문] 한국인 1070세대 ‘금융생활지식’ 10점 만점에 6.73

2018년 스탠더드앤드푸어스(S&P)가 발표한 '세계 금융이해력 조사'에서 한국은 142개국 중 77위를 차지하며 하위권에 머물렀다.

[문화일보] 한국 국민 경제 이해력 평균 56.3점... 금융 문맹률은 67%에 달해

2021년 기획재정부의 국민 경제 이해력 조사 결과, 국민은 평균 56.3점을 받았다. 2018년 세계 금융 이해력 조사(S&P)에서 한국은 142개국 중 77위를 기록했고, 금융 문맹률이 67% 수준으로 나타났다.



개요

자동화된 의사결정

리스크 관리

개인화된 전략

02

관련연구

관련연구

Efficient Market Hypothesis (EMH)
효율적 시장 가설(效率的市場假說)

가격은 상품에 대해 얻을 수 있는 모든 정보를
빠르게 반영하며 이로 인해서 가격 변동을 예측 할 수 없음.

Time-series analysis
시계열 분석(時系列分析)

시계열자료를
분석하고 여러 변수들 간의 인과관계를 분석하는 방법론이다.

Machine Learning
기계분석(機械學習)

03

데이터 분석 & 기계학습

데이터 분석 & 기계학습

```
import numpy as np
import pandas as pd
import seaborn as sns
sns.set_style("whitegrid")
from pandas.plotting import autocorrelation_plot
import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use("ggplot")

#필요한 라이브러리 호출 및 스타일 정의
```

✓ 0.0s

데이터 분석 & 시각화
라이브러리 호출

```
aapl = pd.read_csv("./AAPL.csv")

aapl.head()
```

✓ 0.0s

	Date	Low	Open	Volume	High	Close	Adjusted Close
0	12-12-1980	0.128348	0.128348	469033600	0.128906	0.128348	0.099874
1	15-12-1980	0.121652	0.122210	175884800	0.122210	0.121652	0.094663
2	16-12-1980	0.112723	0.113281	105728000	0.113281	0.112723	0.087715
3	17-12-1980	0.115513	0.115513	86441600	0.116071	0.115513	0.089886
4	18-12-1980	0.118862	0.118862	73449600	0.119420	0.118862	0.092492

데이터 호출

데이터 분석 & 기계학습

```
aapl.dtypes
```

✓ 0.0s

```
Date              object
Low              float64
Open             float64
Volume             int64
High             float64
Close            float64
Adjusted Close   float64
dtype: object
```

```
aapl.isna().sum()

# 결측치 확인
```

✓ 0.0s

```
Date             0
Low              0
Open             0
Volume           0
High             0
Close            0
Adjusted Close   0
dtype: int64
```

데이터 분석

```
aapl.describe()
```

✓ 0.0s

	Low	Open	Volume	High	Close	Adjusted Close
count	10590.000000	10590.000000	1.059000e+04	10590.000000	10590.000000	10590.000000
mean	16.141083	16.323966	3.279583e+08	16.509548	16.332137	15.660945
std	34.495971	34.904787	3.379551e+08	35.327844	34.928463	34.596768
min	0.049107	0.049665	0.000000e+00	0.049665	0.049107	0.038213
25%	0.280134	0.286663	1.215564e+08	0.293699	0.286830	0.237230
50%	0.477500	0.484375	2.151240e+08	0.493304	0.484375	0.401782
75%	15.800536	16.009286	4.070262e+08	16.179196	15.968750	13.831446
max	179.119995	182.630005	7.421641e+09	182.940002	182.009995	180.959732

데이터 기초 통계량 파악

데이터 분석 & 기계학습

```
plt.figure(figsize=(16,6))
plt.title('AAPL Close Price History')
plt.plot(aapl['Close'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.show()

# 데이터 시각화
```

✓ 0.4s

원본 데이터 시각화

```
data = aapl.filter(['Close'])
# 종가로 필터링

dataset = data.values
# Numpy 배열로 저장

training_data_len = int(np.ceil( len(dataset) * .80 ))
# 트레이닝 데이터 비율 조절

training_data_len
```

✓ 0.0s

```
8472
```

훈련 데이터 분리

데이터 분석 & 기계학습

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(dataset)

scaled_data
```

✓ 0.0s

```
array([[4.35483657e-04],
       [3.98684577e-04],
       [3.49613595e-04],
       ...,
       [7.83689774e-01],
       [7.80996942e-01],
       [7.81436607e-01]])
```

데이터 스케일링

```
train_data = scaled_data[0:int(training_data_len), :]

x_train = []
y_train = []

for i in range(60, len(train_data)):
    x_train.append(train_data[i-60:i, 0])
    y_train.append(train_data[i, 0])

x_train, y_train = np.array(x_train), np.array(y_train)

x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
```

✓ 0.0s

훈련 데이터 분리

데이터 분석 & 기계학습

```
from keras.models import Sequential
from keras.layers import Dense, LSTM

model = Sequential()
model.add(LSTM(128, return_sequences=True, input_shape= (x_train.shape[1], 1)))
model.add(LSTM(64, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))

model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(x_train, y_train, batch_size=1, epochs=1)
model.save("my_model.keras")
```

✓ 1m 33.2s

```
8412/8412 [==============================] - 93s 11ms/step - loss: 3.2930e-05
```

모델 생성(LSTM) 및 훈련

```
test_data = scaled_data[training_data_len - 60: , :]

x_test = []
y_test = dataset[training_data_len:, :]
for i in range(60, len(test_data)):
    x_test.append(test_data[i-60:i, 0])

x_test = np.array(x_test)

x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1], 1 ))

predictions = model.predict(x_test)
predictions = scaler.inverse_transform(predictions)

rmse = np.sqrt(np.mean(((predictions - y_test) ** 2)))
rmse
```

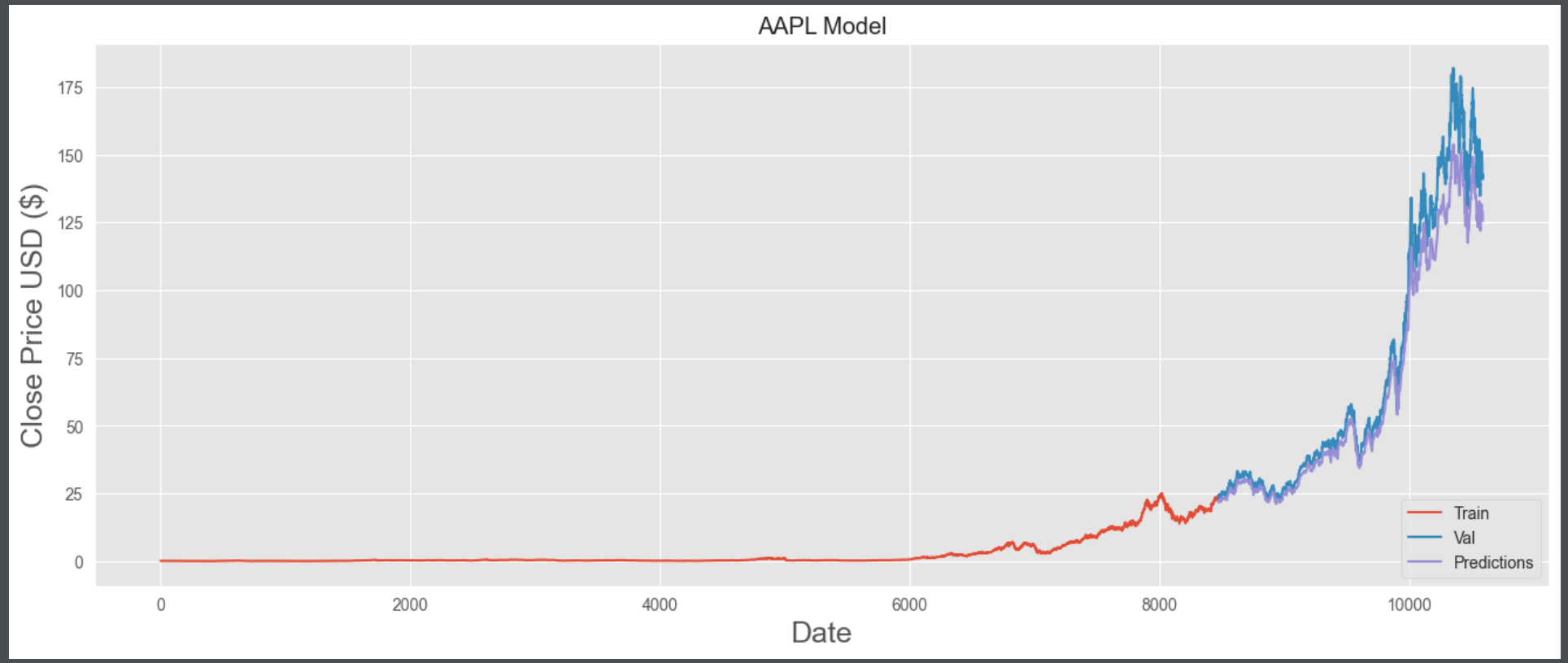
✓ 1.3s

```
67/67 [==============================] - 1s 12ms/step

10.158330477464006
```

데이터 예측 및 성능 확인

데이터 분석 & 기계학습




```
train = data[:training_data_len]
valid = data[training_data_len:]
valid['Predictions'] = predictions
# Visualize the data
plt.figure(figsize=(16,6))
plt.title('AAPL Model')
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
plt.legend(['Train', 'Val', 'Predictions'], loc='lower right')
plt.show()
```

✓ 0.1s

예측 데이터 비교 시각화

데이터 분석 & 기계학습

```
amzn = pd.read_csv('./AMZN.csv')

amzn.head()
```

✓ 0.0s

	Date	Low	Open	Volume	High	Close	Adjusted Close
0	15-05-1997	0.096354	0.121875	1443120000	0.125000	0.097917	0.097917
1	16-05-1997	0.085417	0.098438	294000000	0.098958	0.086458	0.086458
2	19-05-1997	0.081250	0.088021	122136000	0.088542	0.085417	0.085417
3	20-05-1997	0.081771	0.086458	109344000	0.087500	0.081771	0.081771
4	21-05-1997	0.068750	0.081771	377064000	0.082292	0.071354	0.071354

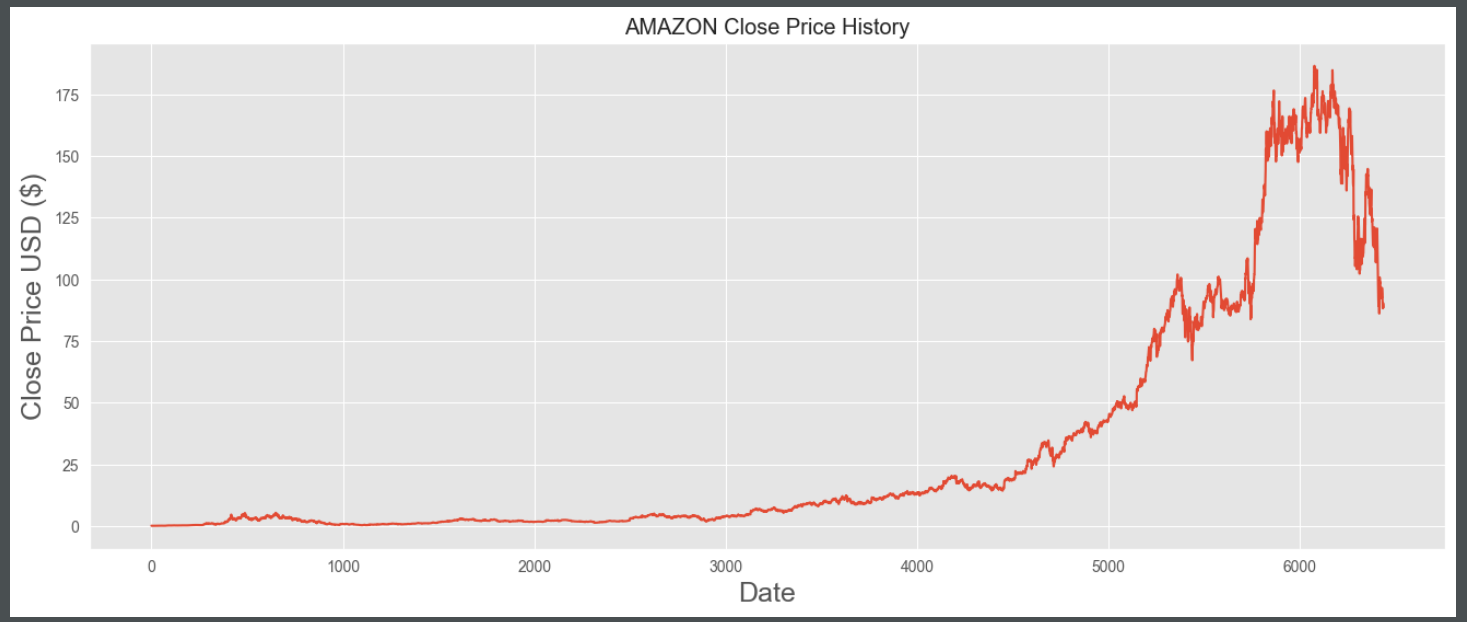
```
amzn.isna().sum()
```

✓ 0.0s

```
Date          0
Low           0
Open          0
Volume        0
High          0
Close         0
Adjusted Close    0
dtype: int64
```

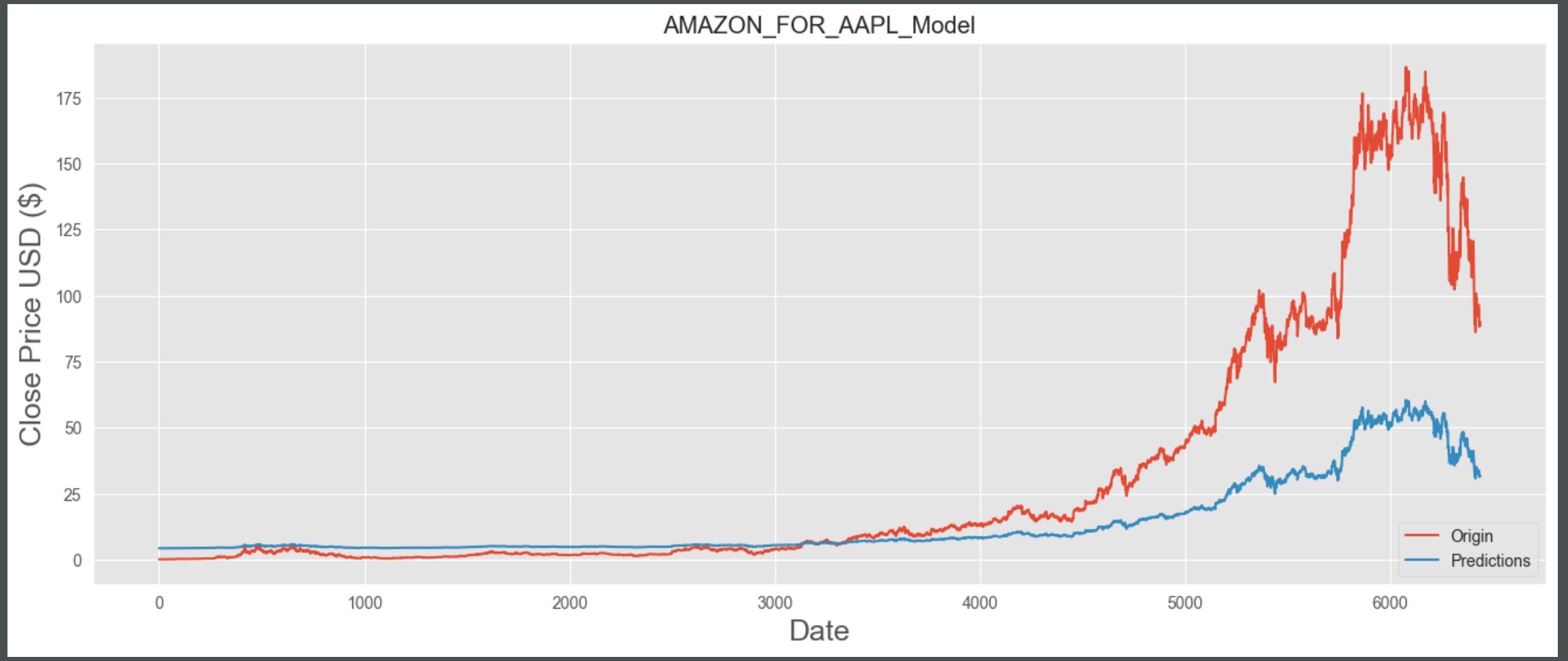
```
plt.figure(figsize=(16,6))
plt.title('AMAZON Close Price History')
plt.plot(amzn['Close'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.show()
```

✓ 0.1s



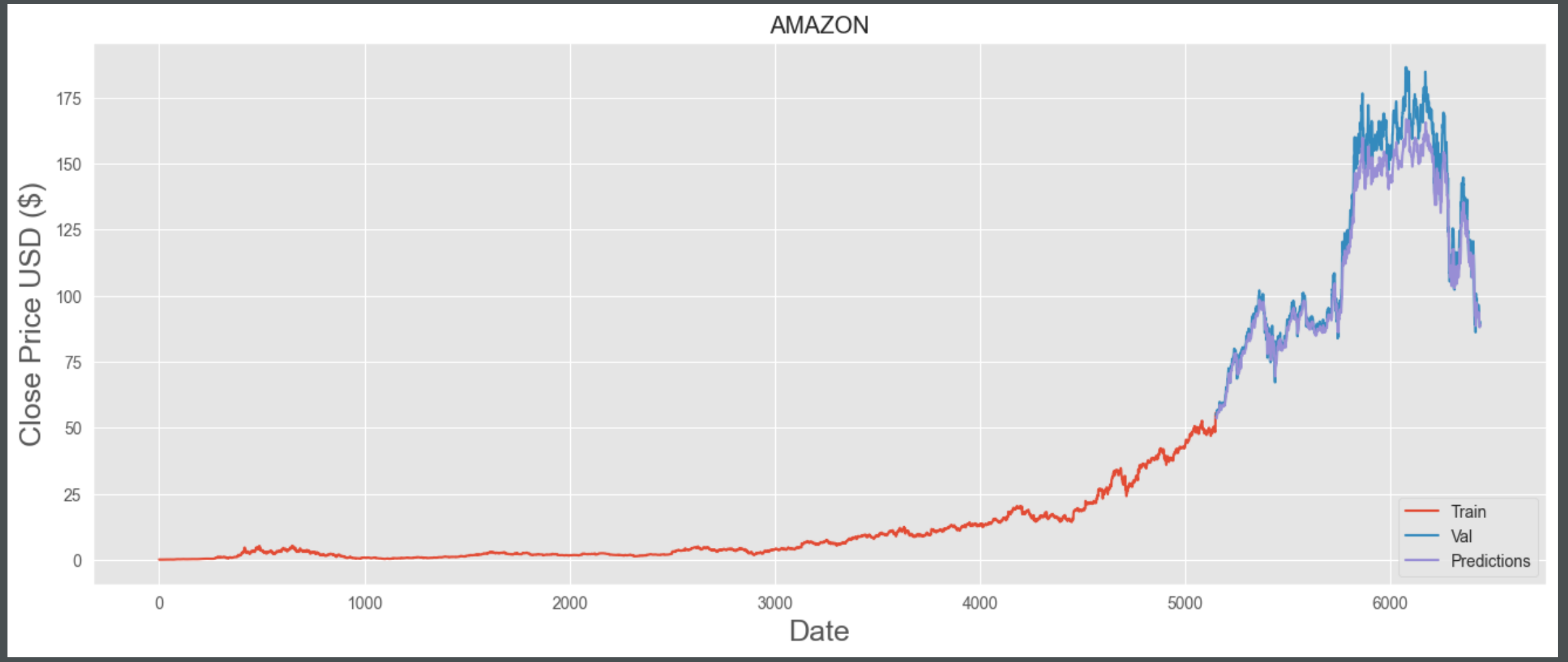
예측 테스트 (동일 모델)

데이터 분석 & 기계학습



예측 테스트 (동일 모델)

데이터 분석 & 기계학습



예측 테스트 (추가 학습)

04

결론 및 소감

결론 및 소감

결론

- 학습되지 않은 데이터에 대해서는 결과의 신뢰도가 떨어짐
- 주식을 시계열 데이터 분석, LSTM을 이용해 기계학습을 하는 것은 합리적이지 않음

소감

- 결과 보다는 경험에 의의
- 생소한 분야에 시야를 넓히게 됨

REFERENCE

Template

- https://slidesgo.com/theme/stock-pitch-deck

Code

- https://www.kaggle.com/code/hajaribrahiem/kagglex-baselines-linear-dense-lstm-cnn-arlstm/input
- https://www.kaggle.com/code/nivmeiri94/data-science-stock-prediction

DataSet

- https://slidesgo.com/theme/stock-pitch-deck

Image

- https://time.com/personalfinance/static/f597bfd818ea4639a9e90bda97ac8d63/57e17/Dividend-stocks.jpg
- https://thenounproject.com/icon/money-graph-108601/
- https://www.flaticon.com/kr/free-icon/research_2345221
- https://www.bok.or.kr/portal/bbs/P0000559/view.do?nttId=10076306&menuNo=200690

News

- https://m.weekly.khan.co.kr/view.html?med_id=weekly&artid=202305191125141&code=114
- https://munhwa.com/news/view.html?no=2023112201032005054001