Task 1

Predicting stock price of **Apple** based on closing price using LSTM

```
import pandas as pd
df = pd.read_csv('AAPL.csv')

df
```

```
             Date        Open        High         Low       Close
Adj Close  \
0      1980-12-12    0.128348    0.128906    0.128348    0.128348
0.099450
1      1980-12-15    0.122210    0.122210    0.121652    0.121652
0.094261
2      1980-12-16    0.113281    0.113281    0.112723    0.112723
0.087343
3      1980-12-17    0.115513    0.116071    0.115513    0.115513
0.089504
4      1980-12-18    0.118862    0.119420    0.118862    0.118862
0.092099
...           ...         ...         ...         ...         ...
...
10765  2023-08-25  177.380005  179.149994  175.820007  178.610001
178.610001
10766  2023-08-28  180.089996  180.589996  178.550003  180.190002
180.190002
10767  2023-08-29  179.699997  184.899994  179.500000  184.119995
184.119995
10768  2023-08-30  184.940002  187.850006  184.740005  187.649994
187.649994
10769  2023-08-31  187.839996  189.119995  187.479996  187.869995
187.869995

          Volume
0      469033600
1      175884800
2      105728000
3       86441600
4       73449600
...          ...
10765   51418700
10766   43820700
10767   53003900
10768   60813900
10769   60735600

[10770 rows x 7 columns]
```

```
df = df[['Date', 'Close']]
```

```
df
```

```
             Date       Close
0      1980-12-12    0.128348
1      1980-12-15    0.121652
2      1980-12-16    0.112723
3      1980-12-17    0.115513
4      1980-12-18    0.118862
...           ...         ...
10765  2023-08-25  178.610001
10766  2023-08-28  180.190002
10767  2023-08-29  184.119995
10768  2023-08-30  187.649994
10769  2023-08-31  187.869995

[10770 rows x 2 columns]
```

```
import datetime

def str_to_datetime(s):
  split = s.split('-')
  year, month, day = int(split[0]), int(split[1]), int(split[2])
  return datetime.datetime(year=year, month=month, day=day)
```

```
df
```

```
             Date       Close
0      1980-12-12    0.128348
1      1980-12-15    0.121652
2      1980-12-16    0.112723
3      1980-12-17    0.115513
4      1980-12-18    0.118862
...           ...         ...
10765  2023-08-25  178.610001
10766  2023-08-28  180.190002
10767  2023-08-29  184.119995
10768  2023-08-30  187.649994
10769  2023-08-31  187.869995

[10770 rows x 2 columns]
```

```
df['Date'] = df['Date'].apply(str_to_datetime)
```

```
<ipython-input-7-82fc7f804c29>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['Date'] = df['Date'].apply(str_to_datetime)
```

```
df['Date']
```

```
0       1980-12-12
1       1980-12-15
2       1980-12-16
3       1980-12-17
4       1980-12-18
           ...
10765   2023-08-25
10766   2023-08-28
10767   2023-08-29
10768   2023-08-30
10769   2023-08-31
Name: Date, Length: 10770, dtype: datetime64[ns]
```

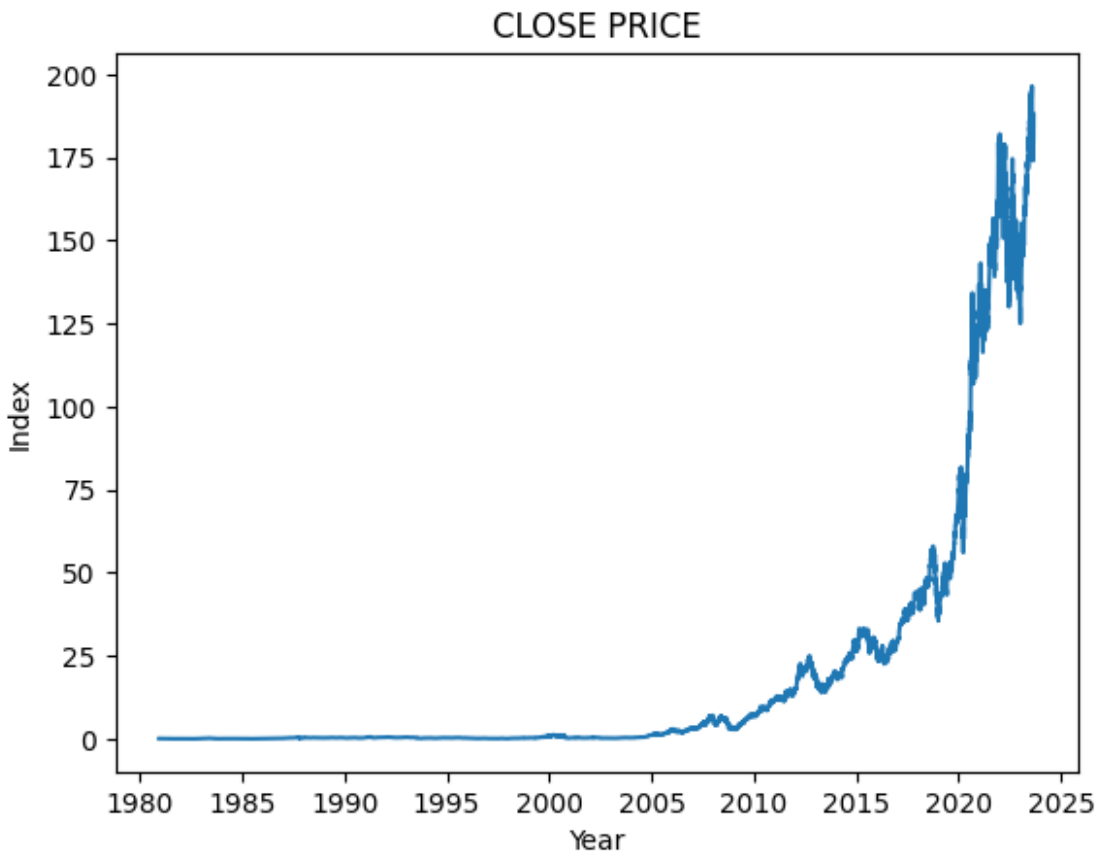
```
df.index = df.pop('Date')
```

```
df
```

```
                 Close
Date
1980-12-12    0.128348
1980-12-15    0.121652
1980-12-16    0.112723
1980-12-17    0.115513
1980-12-18    0.118862
...                ...
2023-08-25  178.610001
2023-08-28  180.190002
2023-08-29  184.119995
2023-08-30  187.649994
2023-08-31  187.869995

[10770 rows x 1 columns]
```

```
import matplotlib.pyplot as plt
plt.plot(df.index, df['Close'])
plt.title('CLOSE PRICE')
plt.xlabel('Year')
plt.ylabel('Index')
```

```
Text(0, 0.5, 'Index')
```




```
import numpy as np

def df_to_windowed_df(dataframe, first_date_str, last_date_str, n=3):
  first_date = str_to_datetime(first_date_str)
  last_date  = str_to_datetime(last_date_str)

  target_date = first_date

  dates = []
  X, Y = [], []

  last_time = False
  while True:
    df_subset = dataframe.loc[:target_date].tail(n+1)

    if len(df_subset) != n+1:
      print(f'Error: Window of size {n} is too large for date
{target_date}')
      return

    values = df_subset['Close'].to_numpy()
    x, y = values[:-1], values[-1]
```

```
    dates.append(target_date)
    X.append(x)
    Y.append(y)

    next_week =
dataframe.loc[target_date:target_date+datetime.timedelta(days=7)]
    next_datetime_str = str(next_week.head(2).tail(1).index.values[0])
    next_date_str = next_datetime_str.split('T')[0]
    year_month_day = next_date_str.split('-')
    year, month, day = year_month_day
    next_date = datetime.datetime(day=int(day), month=int(month),
year=int(year))

    if last_time:
      break

    target_date = next_date

    if target_date == last_date:
      last_time = True

  ret_df = pd.DataFrame({})
  ret_df['Target Date'] = dates

  X = np.array(X)
  for i in range(0, n):
    X[:, i]
    ret_df[f'Target-{n-i}'] = X[:, i]

  ret_df['Target'] = Y

  return ret_df

windowed_df = df_to_windowed_df(df,
                                '2022-08-31',
                                '2023-08-31',
                                n=3)
windowed_df
```

	Target Date	Target-3	Target-2	Target-1	Target
0	2022-08-31	163.619995	161.380005	158.910004	157.220001
1	2022-09-01	161.380005	158.910004	157.220001	157.960007
2	2022-09-02	158.910004	157.220001	157.960007	155.809998
3	2022-09-06	157.220001	157.960007	155.809998	154.529999
4	2022-09-07	157.960007	155.809998	154.529999	155.960007
..	...	...	...	...	...
247	2023-08-25	177.229996	181.119995	176.380005	178.610001
248	2023-08-28	181.119995	176.380005	178.610001	180.190002
249	2023-08-29	176.380005	178.610001	180.190002	184.119995
250	2023-08-30	178.610001	180.190002	184.119995	187.649994

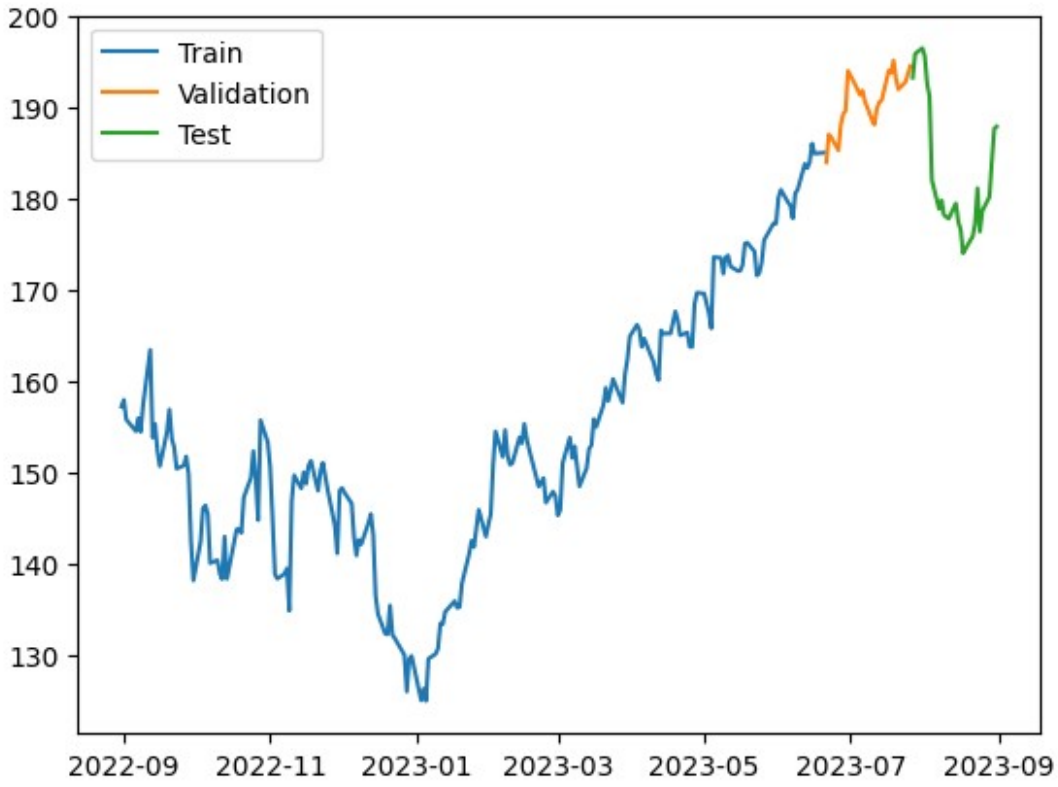
```
251  2023-08-31  180.190002  184.119995  187.649994  187.869995

[252 rows x 5 columns]
```

```
def windowed_df_to_date_X_y(windowed_dataframe):
  df_as_np = windowed_dataframe.to_numpy()

  dates = df_as_np[:, 0]

  middle_matrix = df_as_np[:, 1:-1]
  X = middle_matrix.reshape((len(dates), middle_matrix.shape[1], 1))

  Y = df_as_np[:, -1]

  return dates, X.astype(np.float32), Y.astype(np.float32)

dates, X, y = windowed_df_to_date_X_y(windowed_df)

dates.shape, X.shape, y.shape
```

```
((252,), (252, 3, 1), (252,))
```

```
q_80 = int(len(dates) * .8)
q_90 = int(len(dates) * .9)

dates_train, X_train, y_train = dates[:q_80], X[:q_80], y[:q_80]

dates_val, X_val, y_val = dates[q_80:q_90], X[q_80:q_90], y[q_80:q_90]
dates_test, X_test, y_test = dates[q_90:], X[q_90:], y[q_90:]

plt.plot(dates_train, y_train)
plt.plot(dates_val, y_val)
plt.plot(dates_test, y_test)

plt.legend(['Train', 'Validation', 'Test'])
```

```
<matplotlib.legend.Legend at 0x7f22a08b0fd0>
```




```
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import layers

model = Sequential([layers.Input((3, 1)),
                    layers.LSTM(64),
                    layers.Dense(32, activation='relu'),
                    layers.Dense(32, activation='relu'),
                    layers.Dense(1)])

model.compile(loss='mse',
              optimizer=Adam(learning_rate=0.001),
              metrics=['mean_absolute_error'])

model.fit(X_train, y_train, validation_data=(X_val, y_val),
epochs=100)
```

```
Epoch 1/100
7/7 [==============================] - 3s 103ms/step - loss:
23772.3809 - mean_absolute_error: 153.5365 - val_loss: 36275.6875 -
val_mean_absolute_error: 190.4390
Epoch 2/100
7/7 [==============================] - 0s 10ms/step - loss: 23568.2109
- mean_absolute_error: 152.8699 - val_loss: 36039.4219 -
val_mean_absolute_error: 189.8177
```

```
Epoch 3/100
7/7 [==============================] - 0s 13ms/step - loss: 23386.8750
- mean_absolute_error: 152.2769 - val_loss: 35812.3125 -
val_mean_absolute_error: 189.2185
Epoch 4/100
7/7 [==============================] - 0s 12ms/step - loss: 23189.0801
- mean_absolute_error: 151.6253 - val_loss: 35532.1094 -
val_mean_absolute_error: 188.4766
Epoch 5/100
7/7 [==============================] - 0s 9ms/step - loss: 22939.0566
- mean_absolute_error: 150.7984 - val_loss: 35159.5117 -
val_mean_absolute_error: 187.4856
Epoch 6/100
7/7 [==============================] - 0s 9ms/step - loss: 22607.5898
- mean_absolute_error: 149.6932 - val_loss: 34686.6953 -
val_mean_absolute_error: 186.2204
Epoch 7/100
7/7 [==============================] - 0s 11ms/step - loss: 22149.6836
- mean_absolute_error: 148.1495 - val_loss: 33991.1445 -
val_mean_absolute_error: 184.3432
Epoch 8/100
7/7 [==============================] - 0s 12ms/step - loss: 21413.2520
- mean_absolute_error: 145.6511 - val_loss: 32792.6445 -
val_mean_absolute_error: 181.0633
Epoch 9/100
7/7 [==============================] - 0s 9ms/step - loss: 20399.0625
- mean_absolute_error: 142.1100 - val_loss: 31362.6348 -
val_mean_absolute_error: 177.0704
Epoch 10/100
7/7 [==============================] - 0s 10ms/step - loss: 19166.0879
- mean_absolute_error: 137.6978 - val_loss: 29552.6016 -
val_mean_absolute_error: 171.8831
Epoch 11/100
7/7 [==============================] - 0s 13ms/step - loss: 17498.6895
- mean_absolute_error: 131.5102 - val_loss: 27053.1445 -
val_mean_absolute_error: 164.4517
Epoch 12/100
7/7 [==============================] - 0s 12ms/step - loss: 15536.9912
- mean_absolute_error: 123.8320 - val_loss: 24413.9180 -
val_mean_absolute_error: 156.2218
Epoch 13/100
7/7 [==============================] - 0s 9ms/step - loss: 13553.0850
- mean_absolute_error: 115.5581 - val_loss: 21716.7441 -
val_mean_absolute_error: 147.3363
Epoch 14/100
7/7 [==============================] - 0s 10ms/step - loss: 11319.2100
- mean_absolute_error: 105.3407 - val_loss: 18396.5820 -
val_mean_absolute_error: 135.6015
Epoch 15/100
```

```
7/7 [==============================] - 0s 9ms/step - loss: 8803.1670 -
mean_absolute_error: 92.6271 - val_loss: 14739.4111 -
val_mean_absolute_error: 121.3691
Epoch 16/100
7/7 [==============================] - 0s 10ms/step - loss: 6135.0068
- mean_absolute_error: 76.9480 - val_loss: 10927.7539 -
val_mean_absolute_error: 104.4941
Epoch 17/100
7/7 [==============================] - 0s 10ms/step - loss: 3862.5093
- mean_absolute_error: 60.2852 - val_loss: 7576.2588 -
val_mean_absolute_error: 86.9899
Epoch 18/100
7/7 [==============================] - 0s 13ms/step - loss: 1963.5808
- mean_absolute_error: 41.7689 - val_loss: 4699.5923 -
val_mean_absolute_error: 68.4901
Epoch 19/100
7/7 [==============================] - 0s 9ms/step - loss: 838.9727 -
mean_absolute_error: 24.9081 - val_loss: 2769.0913 -
val_mean_absolute_error: 52.5356
Epoch 20/100
7/7 [==============================] - 0s 11ms/step - loss: 271.3718 -
mean_absolute_error: 12.6475 - val_loss: 1285.2456 -
val_mean_absolute_error: 35.7275
Epoch 21/100
7/7 [==============================] - 0s 9ms/step - loss: 220.4550 -
mean_absolute_error: 12.4415 - val_loss: 895.9570 -
val_mean_absolute_error: 29.7875
Epoch 22/100
7/7 [==============================] - 0s 9ms/step - loss: 263.4748 -
mean_absolute_error: 13.9528 - val_loss: 889.0671 -
val_mean_absolute_error: 29.6717
Epoch 23/100
7/7 [==============================] - 0s 9ms/step - loss: 242.1751 -
mean_absolute_error: 13.3374 - val_loss: 1065.2030 -
val_mean_absolute_error: 32.5046
Epoch 24/100
7/7 [==============================] - 0s 9ms/step - loss: 211.9445 -
mean_absolute_error: 12.1712 - val_loss: 1275.1309 -
val_mean_absolute_error: 35.5876
Epoch 25/100
7/7 [==============================] - 0s 10ms/step - loss: 198.1655 -
mean_absolute_error: 11.3666 - val_loss: 1410.2869 -
val_mean_absolute_error: 37.4384
Epoch 26/100
7/7 [==============================] - 0s 10ms/step - loss: 199.0915 -
mean_absolute_error: 11.1740 - val_loss: 1490.2368 -
val_mean_absolute_error: 38.4920
Epoch 27/100
7/7 [==============================] - 0s 9ms/step - loss: 195.2226 -
```

```
mean_absolute_error: 10.9538 - val_loss: 1486.1117 -
val_mean_absolute_error: 38.4402
Epoch 28/100
7/7 [==============================] - 0s 10ms/step - loss: 189.7764 -
mean_absolute_error: 10.8014 - val_loss: 1417.2126 -
val_mean_absolute_error: 37.5343
Epoch 29/100
7/7 [==============================] - 0s 9ms/step - loss: 186.2268 -
mean_absolute_error: 10.7974 - val_loss: 1347.2640 -
val_mean_absolute_error: 36.5917
Epoch 30/100
7/7 [==============================] - 0s 9ms/step - loss: 182.4146 -
mean_absolute_error: 10.6858 - val_loss: 1341.0714 -
val_mean_absolute_error: 36.5092
Epoch 31/100
7/7 [==============================] - 0s 9ms/step - loss: 180.6131 -
mean_absolute_error: 10.6935 - val_loss: 1260.8054 -
val_mean_absolute_error: 35.3919
Epoch 32/100
7/7 [==============================] - 0s 10ms/step - loss: 179.0275 -
mean_absolute_error: 10.6949 - val_loss: 1287.7272 -
val_mean_absolute_error: 35.7725
Epoch 33/100
7/7 [==============================] - 0s 10ms/step - loss: 177.8291 -
mean_absolute_error: 10.7017 - val_loss: 1212.7625 -
val_mean_absolute_error: 34.7073
Epoch 34/100
7/7 [==============================] - 0s 11ms/step - loss: 176.2862 -
mean_absolute_error: 10.7111 - val_loss: 1223.4611 -
val_mean_absolute_error: 34.8628
Epoch 35/100
7/7 [==============================] - 0s 10ms/step - loss: 173.9620 -
mean_absolute_error: 10.6186 - val_loss: 1216.4478 -
val_mean_absolute_error: 34.7629
Epoch 36/100
7/7 [==============================] - 0s 9ms/step - loss: 168.1552 -
mean_absolute_error: 10.2726 - val_loss: 1248.4769 -
val_mean_absolute_error: 35.2240
Epoch 37/100
7/7 [==============================] - 0s 12ms/step - loss: 160.0060 -
mean_absolute_error: 10.0026 - val_loss: 1173.2343 -
val_mean_absolute_error: 34.1375
Epoch 38/100
7/7 [==============================] - 0s 10ms/step - loss: 157.9902 -
mean_absolute_error: 10.0968 - val_loss: 1133.7172 -
val_mean_absolute_error: 33.5552
Epoch 39/100
7/7 [==============================] - 0s 10ms/step - loss: 153.1323 -
mean_absolute_error: 9.8947 - val_loss: 1116.8041 -
```

```
val_mean_absolute_error: 33.3053
Epoch 40/100
7/7 [==============================] - 0s 9ms/step - loss: 144.7677 -
mean_absolute_error: 9.5169 - val_loss: 1115.5244 -
val_mean_absolute_error: 33.2907
Epoch 41/100
7/7 [==============================] - 0s 9ms/step - loss: 131.7771 -
mean_absolute_error: 9.0439 - val_loss: 999.1364 -
val_mean_absolute_error: 31.4937
Epoch 42/100
7/7 [==============================] - 0s 12ms/step - loss: 116.0849 -
mean_absolute_error: 8.5526 - val_loss: 903.3187 -
val_mean_absolute_error: 29.9322
Epoch 43/100
7/7 [==============================] - 0s 10ms/step - loss: 110.1095 -
mean_absolute_error: 8.1813 - val_loss: 893.1309 -
val_mean_absolute_error: 29.7685
Epoch 44/100
7/7 [==============================] - 0s 10ms/step - loss: 101.5848 -
mean_absolute_error: 7.8464 - val_loss: 840.3434 -
val_mean_absolute_error: 28.8725
Epoch 45/100
7/7 [==============================] - 0s 10ms/step - loss: 91.6610 -
mean_absolute_error: 7.6787 - val_loss: 790.1449 -
val_mean_absolute_error: 27.9943
Epoch 46/100
7/7 [==============================] - 0s 12ms/step - loss: 82.7905 -
mean_absolute_error: 7.0207 - val_loss: 671.5276 -
val_mean_absolute_error: 25.7824
Epoch 47/100
7/7 [==============================] - 0s 12ms/step - loss: 75.1025 -
mean_absolute_error: 6.9285 - val_loss: 695.5529 -
val_mean_absolute_error: 26.2594
Epoch 48/100
7/7 [==============================] - 0s 11ms/step - loss: 62.9342 -
mean_absolute_error: 6.0518 - val_loss: 601.8782 -
val_mean_absolute_error: 24.4037
Epoch 49/100
7/7 [==============================] - 0s 9ms/step - loss: 59.9346 -
mean_absolute_error: 6.2469 - val_loss: 567.6635 -
val_mean_absolute_error: 23.7071
Epoch 50/100
7/7 [==============================] - 0s 10ms/step - loss: 50.1106 -
mean_absolute_error: 5.6382 - val_loss: 485.5687 -
val_mean_absolute_error: 21.9042
Epoch 51/100
7/7 [==============================] - 0s 13ms/step - loss: 36.4637 -
mean_absolute_error: 4.6938 - val_loss: 417.1528 -
val_mean_absolute_error: 20.2825
```

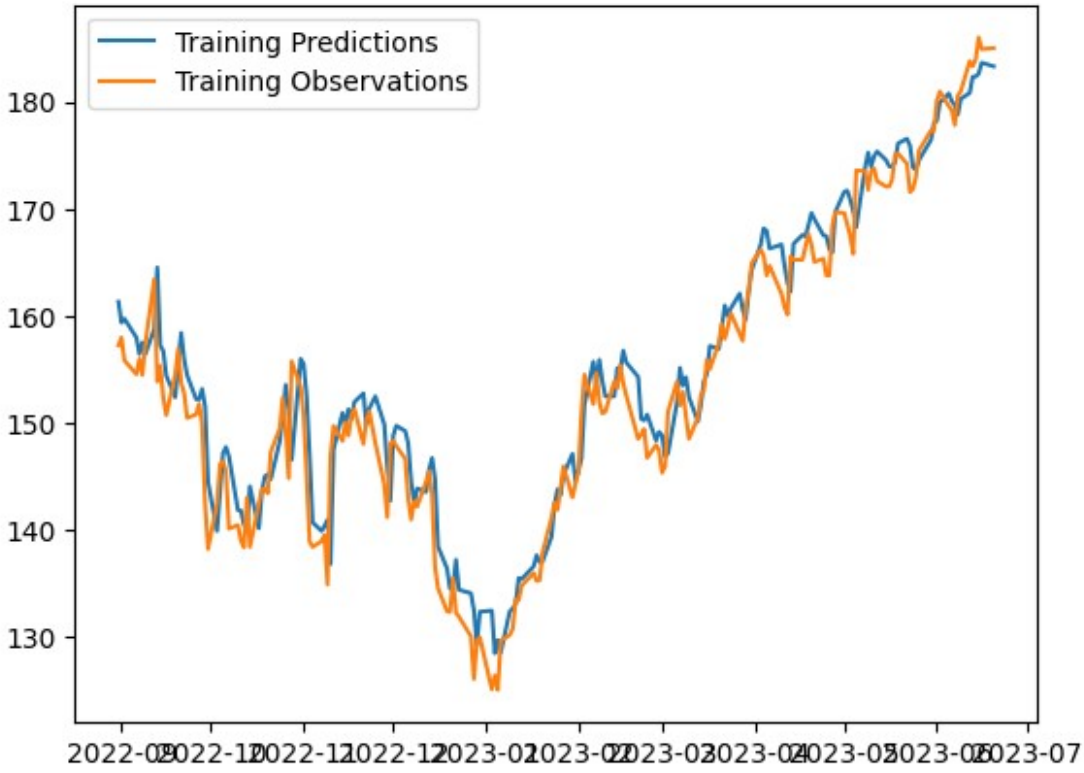
```
Epoch 52/100
7/7 [==============================] - 0s 9ms/step - loss: 31.4773 -
mean_absolute_error: 4.3614 - val_loss: 348.1501 -
val_mean_absolute_error: 18.5022
Epoch 53/100
7/7 [==============================] - 0s 9ms/step - loss: 28.7694 -
mean_absolute_error: 4.1870 - val_loss: 321.6772 -
val_mean_absolute_error: 17.7773
Epoch 54/100
7/7 [==============================] - 0s 10ms/step - loss: 22.9562 -
mean_absolute_error: 3.6597 - val_loss: 293.5599 -
val_mean_absolute_error: 16.9723
Epoch 55/100
7/7 [==============================] - 0s 12ms/step - loss: 20.6193 -
mean_absolute_error: 3.4551 - val_loss: 268.4052 -
val_mean_absolute_error: 16.2185
Epoch 56/100
7/7 [==============================] - 0s 12ms/step - loss: 18.9022 -
mean_absolute_error: 3.3070 - val_loss: 227.8057 -
val_mean_absolute_error: 14.9125
Epoch 57/100
7/7 [==============================] - 0s 9ms/step - loss: 17.6522 -
mean_absolute_error: 3.2769 - val_loss: 197.2581 -
val_mean_absolute_error: 13.8495
Epoch 58/100
7/7 [==============================] - 0s 11ms/step - loss: 16.6624 -
mean_absolute_error: 3.1313 - val_loss: 193.3367 -
val_mean_absolute_error: 13.7200
Epoch 59/100
7/7 [==============================] - 0s 9ms/step - loss: 16.9198 -
mean_absolute_error: 3.2442 - val_loss: 193.8337 -
val_mean_absolute_error: 13.7503
Epoch 60/100
7/7 [==============================] - 0s 9ms/step - loss: 14.4918 -
mean_absolute_error: 2.9819 - val_loss: 160.8454 -
val_mean_absolute_error: 12.4853
Epoch 61/100
7/7 [==============================] - 0s 11ms/step - loss: 12.3734 -
mean_absolute_error: 2.7061 - val_loss: 154.1432 -
val_mean_absolute_error: 12.2194
Epoch 62/100
7/7 [==============================] - 0s 11ms/step - loss: 12.6811 -
mean_absolute_error: 2.7832 - val_loss: 136.2751 -
val_mean_absolute_error: 11.4612
Epoch 63/100
7/7 [==============================] - 0s 9ms/step - loss: 11.7470 -
mean_absolute_error: 2.6639 - val_loss: 131.1218 -
val_mean_absolute_error: 11.2413
Epoch 64/100
```

```
7/7 [==============================] - 0s 10ms/step - loss: 11.0823 -
mean_absolute_error: 2.5916 - val_loss: 134.5838 -
val_mean_absolute_error: 11.4053
Epoch 65/100
7/7 [==============================] - 0s 9ms/step - loss: 12.2693 -
mean_absolute_error: 2.7773 - val_loss: 117.9387 -
val_mean_absolute_error: 10.6471
Epoch 66/100
7/7 [==============================] - 0s 9ms/step - loss: 11.2256 -
mean_absolute_error: 2.5688 - val_loss: 108.8859 -
val_mean_absolute_error: 10.2135
Epoch 67/100
7/7 [==============================] - 0s 10ms/step - loss: 10.9425 -
mean_absolute_error: 2.5793 - val_loss: 96.9112 -
val_mean_absolute_error: 9.6054
Epoch 68/100
7/7 [==============================] - 0s 10ms/step - loss: 10.1744 -
mean_absolute_error: 2.4771 - val_loss: 98.1279 -
val_mean_absolute_error: 9.6773
Epoch 69/100
7/7 [==============================] - 0s 9ms/step - loss: 10.3190 -
mean_absolute_error: 2.4961 - val_loss: 91.3156 -
val_mean_absolute_error: 9.3169
Epoch 70/100
7/7 [==============================] - 0s 9ms/step - loss: 11.0626 -
mean_absolute_error: 2.5645 - val_loss: 80.6446 -
val_mean_absolute_error: 8.7217
Epoch 71/100
7/7 [==============================] - 0s 13ms/step - loss: 10.4899 -
mean_absolute_error: 2.4942 - val_loss: 77.8625 -
val_mean_absolute_error: 8.5666
Epoch 72/100
7/7 [==============================] - 0s 10ms/step - loss: 10.1722 -
mean_absolute_error: 2.4502 - val_loss: 78.4187 -
val_mean_absolute_error: 8.6040
Epoch 73/100
7/7 [==============================] - 0s 9ms/step - loss: 9.8120 -
mean_absolute_error: 2.4397 - val_loss: 68.5188 -
val_mean_absolute_error: 8.0021
Epoch 74/100
7/7 [==============================] - 0s 12ms/step - loss: 11.1284 -
mean_absolute_error: 2.6438 - val_loss: 70.0688 -
val_mean_absolute_error: 8.1096
Epoch 75/100
7/7 [==============================] - 0s 9ms/step - loss: 9.5330 -
mean_absolute_error: 2.4328 - val_loss: 78.2827 -
val_mean_absolute_error: 8.6158
Epoch 76/100
7/7 [==============================] - 0s 9ms/step - loss: 9.5403 -
```

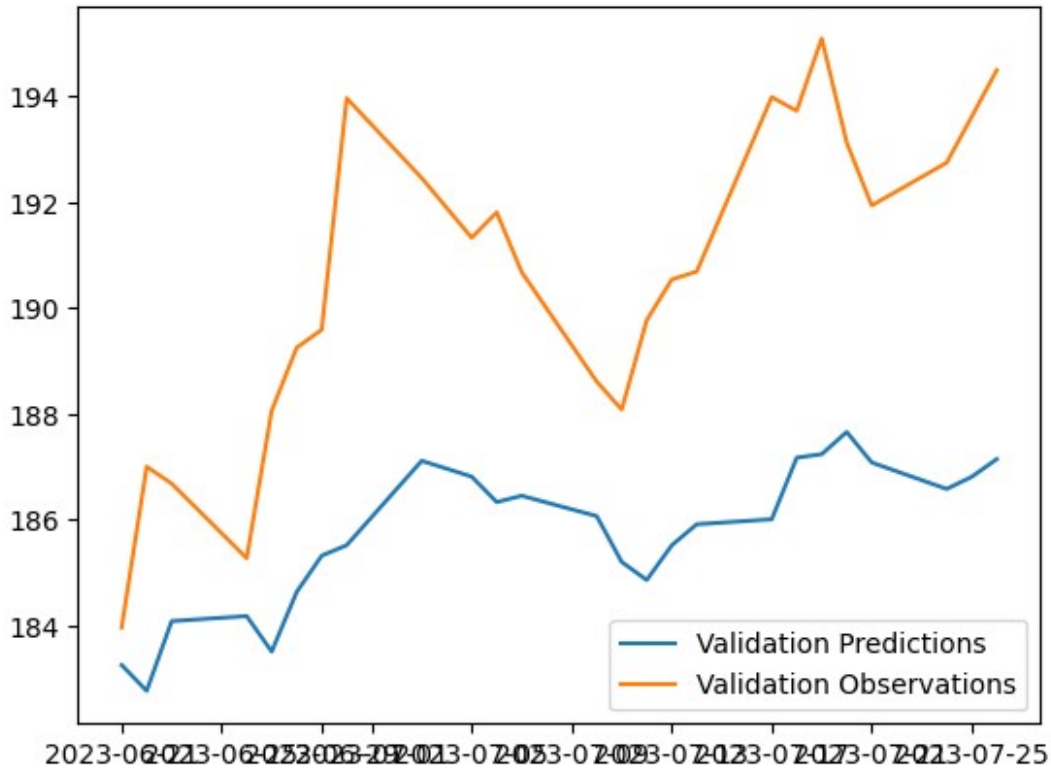
```
mean_absolute_error: 2.4107 - val_loss: 75.0244 -
val_mean_absolute_error: 8.4246
Epoch 77/100
7/7 [==============================] - 0s 15ms/step - loss: 10.4302 -
mean_absolute_error: 2.5052 - val_loss: 77.1762 -
val_mean_absolute_error: 8.5593
Epoch 78/100
7/7 [==============================] - 0s 9ms/step - loss: 13.1435 -
mean_absolute_error: 2.8788 - val_loss: 66.9169 -
val_mean_absolute_error: 7.9325
Epoch 79/100
7/7 [==============================] - 0s 9ms/step - loss: 9.4024 -
mean_absolute_error: 2.3768 - val_loss: 66.5101 -
val_mean_absolute_error: 7.9121
Epoch 80/100
7/7 [==============================] - 0s 10ms/step - loss: 9.8418 -
mean_absolute_error: 2.4214 - val_loss: 55.8333 -
val_mean_absolute_error: 7.1919
Epoch 81/100
7/7 [==============================] - 0s 10ms/step - loss: 9.8656 -
mean_absolute_error: 2.4177 - val_loss: 61.1910 -
val_mean_absolute_error: 7.5639
Epoch 82/100
7/7 [==============================] - 0s 12ms/step - loss: 9.0151 -
mean_absolute_error: 2.3437 - val_loss: 63.0086 -
val_mean_absolute_error: 7.6902
Epoch 83/100
7/7 [==============================] - 0s 9ms/step - loss: 9.3005 -
mean_absolute_error: 2.3216 - val_loss: 52.6990 -
val_mean_absolute_error: 6.9806
Epoch 84/100
7/7 [==============================] - 0s 12ms/step - loss: 9.5189 -
mean_absolute_error: 2.3936 - val_loss: 44.5514 -
val_mean_absolute_error: 6.3591
Epoch 85/100
7/7 [==============================] - 0s 9ms/step - loss: 9.9476 -
mean_absolute_error: 2.3782 - val_loss: 41.2438 -
val_mean_absolute_error: 6.0900
Epoch 86/100
7/7 [==============================] - 0s 8ms/step - loss: 11.7243 -
mean_absolute_error: 2.6219 - val_loss: 41.9635 -
val_mean_absolute_error: 6.1565
Epoch 87/100
7/7 [==============================] - 0s 9ms/step - loss: 11.2818 -
mean_absolute_error: 2.6764 - val_loss: 50.0701 -
val_mean_absolute_error: 6.8020
Epoch 88/100
7/7 [==============================] - 0s 12ms/step - loss: 9.0579 -
mean_absolute_error: 2.3278 - val_loss: 43.0414 -
```

```
val_mean_absolute_error: 6.2572
Epoch 89/100
7/7 [==============================] - 0s 11ms/step - loss: 9.2308 -
mean_absolute_error: 2.3371 - val_loss: 41.7234 -
val_mean_absolute_error: 6.1523
Epoch 90/100
7/7 [==============================] - 0s 12ms/step - loss: 8.9410 -
mean_absolute_error: 2.3197 - val_loss: 50.9233 -
val_mean_absolute_error: 6.8756
Epoch 91/100
7/7 [==============================] - 0s 9ms/step - loss: 9.5612 -
mean_absolute_error: 2.3545 - val_loss: 57.2259 -
val_mean_absolute_error: 7.3294
Epoch 92/100
7/7 [==============================] - 0s 9ms/step - loss: 11.2345 -
mean_absolute_error: 2.5650 - val_loss: 43.8583 -
val_mean_absolute_error: 6.3358
Epoch 93/100
7/7 [==============================] - 0s 9ms/step - loss: 8.8014 -
mean_absolute_error: 2.2741 - val_loss: 35.6865 -
val_mean_absolute_error: 5.6453
Epoch 94/100
7/7 [==============================] - 0s 14ms/step - loss: 11.0551 -
mean_absolute_error: 2.4627 - val_loss: 42.5556 -
val_mean_absolute_error: 6.2394
Epoch 95/100
7/7 [==============================] - 0s 16ms/step - loss: 8.6420 -
mean_absolute_error: 2.2748 - val_loss: 40.0935 -
val_mean_absolute_error: 6.0338
Epoch 96/100
7/7 [==============================] - 0s 17ms/step - loss: 8.6674 -
mean_absolute_error: 2.2590 - val_loss: 43.7137 -
val_mean_absolute_error: 6.3320
Epoch 97/100
7/7 [==============================] - 0s 15ms/step - loss: 8.9867 -
mean_absolute_error: 2.3053 - val_loss: 39.5028 -
val_mean_absolute_error: 5.9879
Epoch 98/100
7/7 [==============================] - 0s 17ms/step - loss: 8.9163 -
mean_absolute_error: 2.3562 - val_loss: 34.2921 -
val_mean_absolute_error: 5.5330
Epoch 99/100
7/7 [==============================] - 0s 18ms/step - loss: 9.2482 -
mean_absolute_error: 2.3514 - val_loss: 33.2577 -
val_mean_absolute_error: 5.4384
Epoch 100/100
7/7 [==============================] - 0s 17ms/step - loss: 9.1897 -
mean_absolute_error: 2.2820 - val_loss: 28.1247 -
val_mean_absolute_error: 4.9345
```

```
<keras.src.callbacks.History at 0x7f22425526e0>

train_predictions = model.predict(X_train).flatten()

plt.plot(dates_train, train_predictions)
plt.plot(dates_train, y_train)
plt.legend(['Training Predictions', 'Training Observations'])

7/7 [==============================] - 0s 3ms/step

<matplotlib.legend.Legend at 0x7f222e6efd00>
```



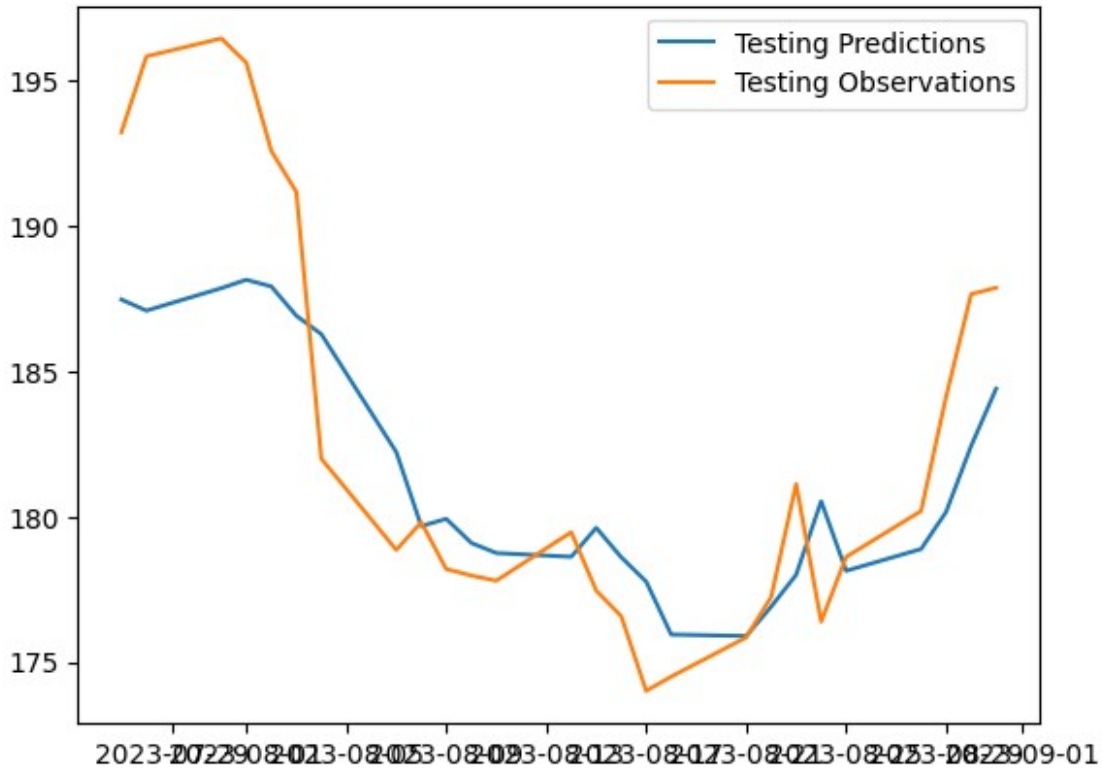

```
val_predictions = model.predict(X_val).flatten()

plt.plot(dates_val, val_predictions)
plt.plot(dates_val, y_val)
plt.legend(['Validation Predictions', 'Validation Observations'])

1/1 [==============================] - 0s 29ms/step

<matplotlib.legend.Legend at 0x7f223fabae00>
```




```
test_predictions = model.predict(X_test).flatten()

plt.plot(dates_test, test_predictions)
plt.plot(dates_test, y_test)
plt.legend(['Testing Predictions', 'Testing Observations'])
```

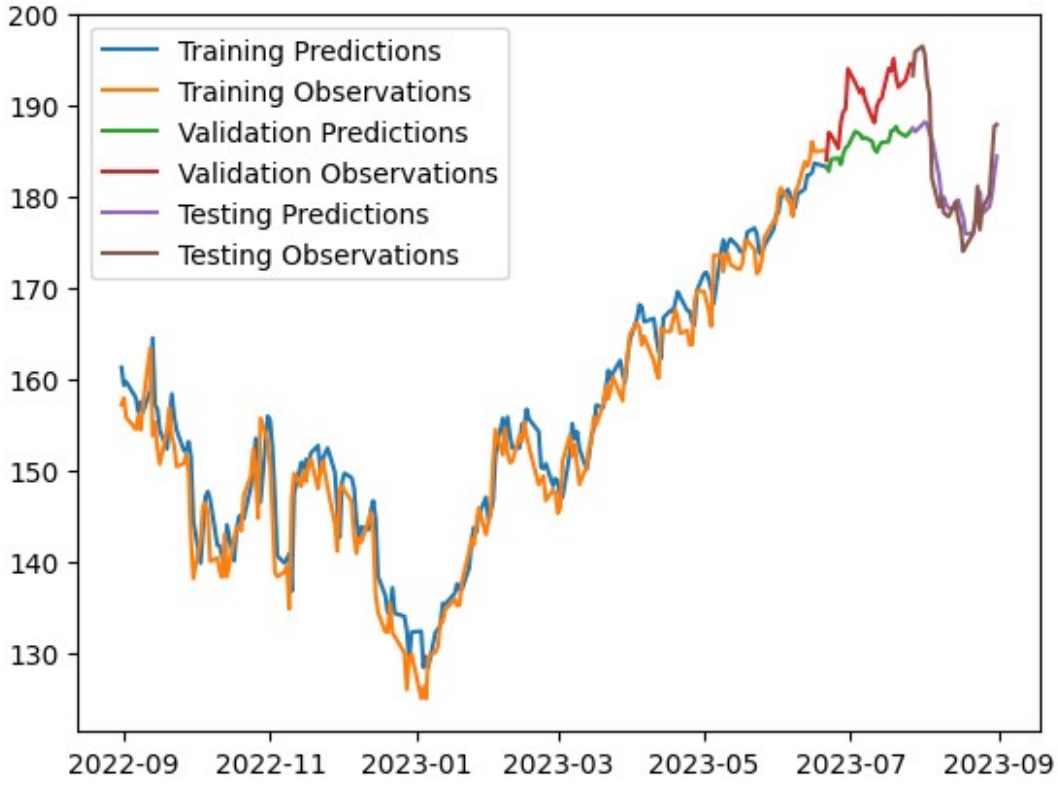
```
1/1 [==============================] - 0s 28ms/step
```

```
<matplotlib.legend.Legend at 0x7f223e4f7e50>
```




```
plt.plot(dates_train, train_predictions)
plt.plot(dates_train, y_train)
plt.plot(dates_val, val_predictions)
plt.plot(dates_val, y_val)
plt.plot(dates_test, test_predictions)
plt.plot(dates_test, y_test)
plt.legend(['Training Predictions',
            'Training Observations',
            'Validation Predictions',
            'Validation Observations',
            'Testing Predictions',
            'Testing Observations'])
```

```
<matplotlib.legend.Legend at 0x7f223ea38d30>
```


Training Predictions
Training Observations
Validation Predictions
Validation Observations
Testing Predictions
Testing Observations
200
190
180
170
160
150
140
130
2022-09
2022-11
2023-01
2023-03
2023-05
2023-07
2023-09