stock-market-predection

July 29, 2023

[1]:
```
import pandas as pd

df = pd.read_csv('/content/stock price predection.csv')

df
```

[1]:
```
            Date        Open        High         Low       Close   Adj Close  \
0     2018-02-05  262.000000  267.899994  250.029999  254.259995  254.259995
1     2018-02-06  247.699997  266.700012  245.000000  265.720001  265.720001
2     2018-02-07  266.579987  272.450012  264.329987  264.559998  264.559998
3     2018-02-08  267.079987  267.619995  250.000000  250.100006  250.100006
4     2018-02-09  253.850006  255.800003  236.110001  249.470001  249.470001
…            …           …           …           …           …           …
1004  2022-01-31  401.970001  427.700012  398.200012  427.140015  427.140015
1005  2022-02-01  432.959991  458.480011  425.540009  457.130005  457.130005
1006  2022-02-02  448.250000  451.980011  426.480011  429.480011  429.480011
1007  2022-02-03  421.440002  429.260010  404.279999  405.600006  405.600006
1008  2022-02-04  407.309998  412.769989  396.640015  410.170013  410.170013

        Volume
0     11896100
1     12595800
2      8981500
3      9306700
4     16906900
…          …
1004  20047500
1005  22542300
1006  14346000
1007   9905200
1008   7782400

[1009 rows x 7 columns]
```

[2]:
```
df = df[['Date', 'Close']]

df
```

[2]:
```
           Date       Close
0    2018-02-05  254.259995
1    2018-02-06  265.720001
2    2018-02-07  264.559998
3    2018-02-08  250.100006
4    2018-02-09  249.470001
…           …         …
1004 2022-01-31  427.140015
1005 2022-02-01  457.130005
1006 2022-02-02  429.480011
1007 2022-02-03  405.600006
1008 2022-02-04  410.170013

[1009 rows x 2 columns]
```

[3]:
```
df['Date']
```

[3]:
```
0       2018-02-05
1       2018-02-06
2       2018-02-07
3       2018-02-08
4       2018-02-09
           …
1004    2022-01-31
1005    2022-02-01
1006    2022-02-02
1007    2022-02-03
1008    2022-02-04
Name: Date, Length: 1009, dtype: object
```

[29]:
```
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)

datetime_object = str_to_datetime('2018-02-09')
datetime_object
```

[29]:
```
datetime.datetime(2018, 2, 9, 0, 0)
```

[5]:
```
df
```

[5]:
```
           Date       Close
0    2018-02-05  254.259995
1    2018-02-06  265.720001
```

```
2     2018-02-07  264.559998
3     2018-02-08  250.100006
4     2018-02-09  249.470001
…            …           …
1004  2022-01-31  427.140015
1005  2022-02-01  457.130005
1006  2022-02-02  429.480011
1007  2022-02-03  405.600006
1008  2022-02-04  410.170013

[1009 rows x 2 columns]
```

[6]:
```
df['Date'] = df['Date'].apply(str_to_datetime)
df['Date']
```

```
<ipython-input-6-f6fc52bb0fa5>: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)
```

[6]:
```
0      2018-02-05
1      2018-02-06
2      2018-02-07
3      2018-02-08
4      2018-02-09
          …
1004   2022-01-31
1005   2022-02-01
1006   2022-02-02
1007   2022-02-03
1008   2022-02-04
Name: Date, Length: 1009, dtype: datetime64[ns]
```

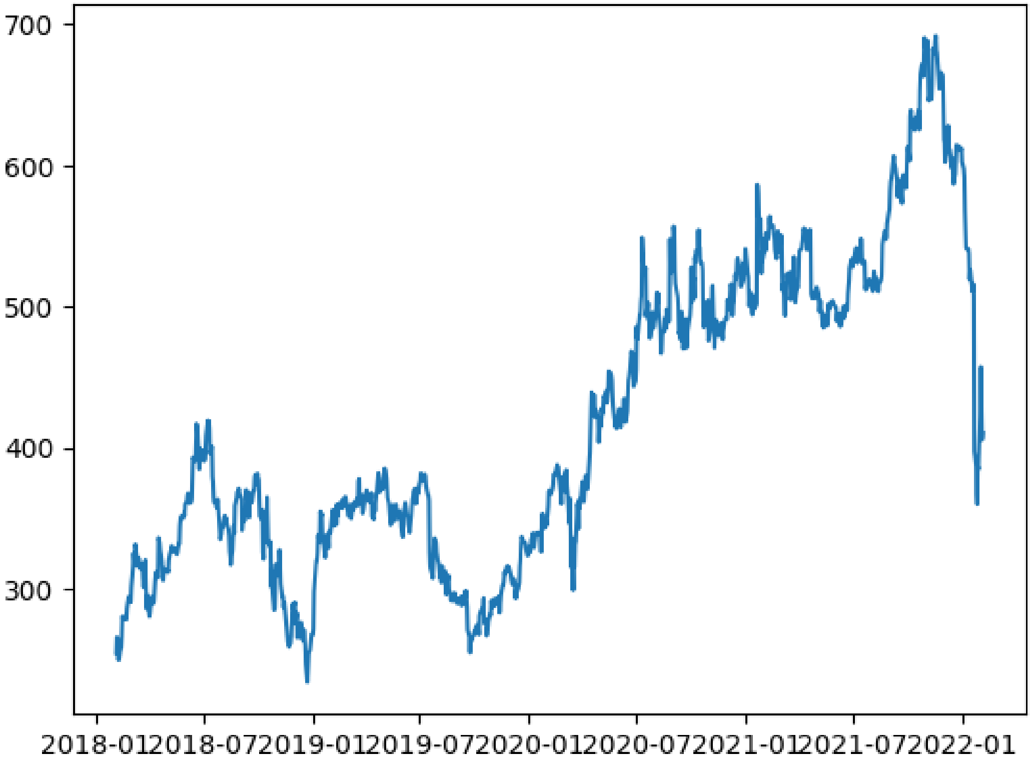
[7]:
```
df.index = df.pop('Date')
df
```

[7]:
```
                 Close
Date
2018-02-05  254.259995
2018-02-06  265.720001
2018-02-07  264.559998
2018-02-08  250.100006
2018-02-09  249.470001
…                  …
```

```
2022-01-31  427.140015
2022-02-01  457.130005
2022-02-02  429.480011
2022-02-03  405.600006
2022-02-04  410.170013

[1009 rows x 1 columns]
```

[8]:
```
import matplotlib.pyplot as plt

plt.plot(df.index, df['Close'])
```

[8]:
```
[<matplotlib.lines.Line2D at 0x7b73fd8a63b0>]
```



[31]:
```
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,
                                '2018-02-09',
                                '2022-02-04',
                                n=3)
windowed_df
```

[31]:

```
     Target Date    Target-3    Target-2    Target-1      Target
0     2018-02-09  265.720001  264.559998  250.100006  249.470001
1     2018-02-12  264.559998  250.100006  249.470001  257.950012
2     2018-02-13  250.100006  249.470001  257.950012  258.269989
3     2018-02-14  249.470001  257.950012  258.269989  266.000000
4     2018-02-15  257.950012  258.269989  266.000000  280.269989
…            …           …           …           …           …
1000  2022-01-31  359.700012  386.700012  384.359985  427.140015
1001  2022-02-01  386.700012  384.359985  427.140015  457.130005
1002  2022-02-02  384.359985  427.140015  457.130005  429.480011
1003  2022-02-03  427.140015  457.130005  429.480011  405.600006
1004  2022-02-04  457.130005  429.480011  405.600006  410.170013

[1005 rows x 5 columns]
```

[14]:

```
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
```

[14]:

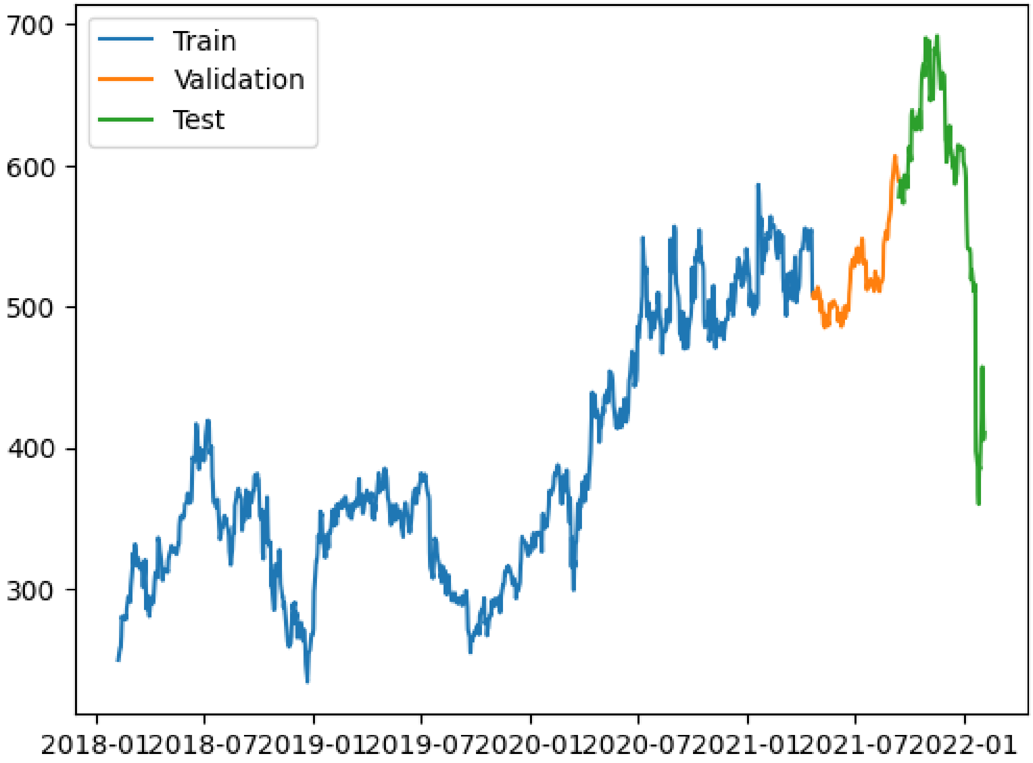
```
((1005,), (1005, 3, 1), (1005,))
```

[15]:

```
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'])
```

```
[15]: <matplotlib.legend.Legend at 0x7b73fb83d210>
```



```
[16]: 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
26/26 [==============================] - 3s 27ms/step - loss: 154996.9375 -
mean_absolute_error: 384.2372 - val_loss: 270641.6875 - val_mean_absolute_error:
519.4163
Epoch 2/100
26/26 [==============================] - 0s 6ms/step - loss: 152540.8594 -
mean_absolute_error: 381.0434 - val_loss: 265031.1875 - val_mean_absolute_error:
513.9865
Epoch 3/100
26/26 [==============================] - 0s 5ms/step - loss: 145159.7969 -
mean_absolute_error: 371.1794 - val_loss: 247883.6875 - val_mean_absolute_error:
497.0260
Epoch 4/100
26/26 [==============================] - 0s 5ms/step - loss: 128101.0859 -
mean_absolute_error: 347.2981 - val_loss: 217547.3438 - val_mean_absolute_error:
465.5093
Epoch 5/100
26/26 [==============================] - 0s 6ms/step - loss: 102066.1719 -
mean_absolute_error: 307.2246 - val_loss: 173090.4375 - val_mean_absolute_error:
415.0206
Epoch 6/100
26/26 [==============================] - 0s 6ms/step - loss: 68615.9922 -
mean_absolute_error: 246.6671 - val_loss: 115762.0938 - val_mean_absolute_error:
338.9886
Epoch 7/100
26/26 [==============================] - 0s 5ms/step - loss: 33097.1055 -
mean_absolute_error: 158.6124 - val_loss: 60488.9531 - val_mean_absolute_error:
244.2132
Epoch 8/100
26/26 [==============================] - 0s 5ms/step - loss: 12174.9805 -
mean_absolute_error: 82.5767 - val_loss: 26947.6699 - val_mean_absolute_error:
161.5529
Epoch 9/100
26/26 [==============================] - 0s 6ms/step - loss: 7535.0234 -
mean_absolute_error: 71.3012 - val_loss: 18550.2520 - val_mean_absolute_error:
133.0515
Epoch 10/100
26/26 [==============================] - 0s 5ms/step - loss: 7334.0537 -
mean_absolute_error: 72.5959 - val_loss: 20946.7480 - val_mean_absolute_error:
142.0623
Epoch 11/100
26/26 [==============================] - 0s 5ms/step - loss: 5710.9946 -
mean_absolute_error: 58.6846 - val_loss: 15614.4160 - val_mean_absolute_error:
121.8209
Epoch 12/100
```

```
26/26 [==============================] - 0s 6ms/step - loss: 4388.4004 -
mean_absolute_error: 52.4315 - val_loss: 14112.3740 - val_mean_absolute_error:
116.4280
Epoch 13/100
26/26 [==============================] - 0s 6ms/step - loss: 3056.8621 -
mean_absolute_error: 42.8179 - val_loss: 7808.3057 - val_mean_absolute_error:
84.5798
Epoch 14/100
26/26 [==============================] - 0s 6ms/step - loss: 1602.6653 -
mean_absolute_error: 29.4872 - val_loss: 3988.1030 - val_mean_absolute_error:
57.9745
Epoch 15/100
26/26 [==============================] - 0s 5ms/step - loss: 702.4193 -
mean_absolute_error: 17.7832 - val_loss: 1873.0781 - val_mean_absolute_error:
35.7536
Epoch 16/100
26/26 [==============================] - 0s 5ms/step - loss: 343.6540 -
mean_absolute_error: 12.5222 - val_loss: 1048.8779 - val_mean_absolute_error:
23.6797
Epoch 17/100
26/26 [==============================] - 0s 6ms/step - loss: 233.2283 -
mean_absolute_error: 10.6512 - val_loss: 734.0581 - val_mean_absolute_error:
18.5219
Epoch 18/100
26/26 [==============================] - 0s 6ms/step - loss: 183.5535 -
mean_absolute_error: 9.7191 - val_loss: 505.4833 - val_mean_absolute_error:
14.3068
Epoch 19/100
26/26 [==============================] - 0s 5ms/step - loss: 173.3000 -
mean_absolute_error: 9.7049 - val_loss: 498.0577 - val_mean_absolute_error:
14.6273
Epoch 20/100
26/26 [==============================] - 0s 5ms/step - loss: 152.0413 -
mean_absolute_error: 8.8392 - val_loss: 386.7752 - val_mean_absolute_error:
12.6382
Epoch 21/100
26/26 [==============================] - 0s 5ms/step - loss: 176.3841 -
mean_absolute_error: 9.9926 - val_loss: 358.3921 - val_mean_absolute_error:
12.2324
Epoch 22/100
26/26 [==============================] - 0s 5ms/step - loss: 143.2870 -
mean_absolute_error: 8.7372 - val_loss: 373.6602 - val_mean_absolute_error:
14.5344
Epoch 23/100
26/26 [==============================] - 0s 5ms/step - loss: 161.2704 -
mean_absolute_error: 9.5224 - val_loss: 316.6615 - val_mean_absolute_error:
11.4183
Epoch 24/100
```

```
26/26 [==============================] - 0s 5ms/step - loss: 134.2278 -
mean_absolute_error: 8.5973 - val_loss: 438.6989 - val_mean_absolute_error:
15.7713
Epoch 25/100
26/26 [==============================] - 0s 5ms/step - loss: 133.1034 -
mean_absolute_error: 8.5429 - val_loss: 292.7980 - val_mean_absolute_error:
10.9124
Epoch 26/100
26/26 [==============================] - 0s 6ms/step - loss: 134.4724 -
mean_absolute_error: 8.5050 - val_loss: 326.9892 - val_mean_absolute_error:
12.1983
Epoch 27/100
26/26 [==============================] - 0s 5ms/step - loss: 132.6534 -
mean_absolute_error: 8.3531 - val_loss: 262.6467 - val_mean_absolute_error:
10.8086
Epoch 28/100
26/26 [==============================] - 0s 6ms/step - loss: 124.8328 -
mean_absolute_error: 8.1036 - val_loss: 271.5528 - val_mean_absolute_error:
10.3862
Epoch 29/100
26/26 [==============================] - 0s 6ms/step - loss: 145.1362 -
mean_absolute_error: 9.0476 - val_loss: 327.0142 - val_mean_absolute_error:
12.7272
Epoch 30/100
26/26 [==============================] - 0s 6ms/step - loss: 153.6421 -
mean_absolute_error: 9.4326 - val_loss: 241.7687 - val_mean_absolute_error:
10.1551
Epoch 31/100
26/26 [==============================] - 0s 5ms/step - loss: 144.6448 -
mean_absolute_error: 9.1171 - val_loss: 334.4837 - val_mean_absolute_error:
13.7688
Epoch 32/100
26/26 [==============================] - 0s 5ms/step - loss: 145.9703 -
mean_absolute_error: 8.9010 - val_loss: 240.0051 - val_mean_absolute_error:
10.6825
Epoch 33/100
26/26 [==============================] - 0s 6ms/step - loss: 141.2333 -
mean_absolute_error: 8.7895 - val_loss: 231.1074 - val_mean_absolute_error:
10.5566
Epoch 34/100
26/26 [==============================] - 0s 6ms/step - loss: 150.0935 -
mean_absolute_error: 9.1135 - val_loss: 252.3089 - val_mean_absolute_error:
11.9935
Epoch 35/100
26/26 [==============================] - 0s 5ms/step - loss: 125.5260 -
mean_absolute_error: 8.1293 - val_loss: 259.5168 - val_mean_absolute_error:
10.8996
Epoch 36/100
```

```
26/26 [==============================] - 0s 6ms/step - loss: 123.0203 -
mean_absolute_error: 8.0662 - val_loss: 215.0604 - val_mean_absolute_error:
9.2880
Epoch 37/100
26/26 [==============================] - 0s 5ms/step - loss: 119.3579 -
mean_absolute_error: 7.9570 - val_loss: 228.2355 - val_mean_absolute_error:
9.7337
Epoch 38/100
26/26 [==============================] - 0s 6ms/step - loss: 143.9167 -
mean_absolute_error: 8.9914 - val_loss: 212.9570 - val_mean_absolute_error:
10.6698
Epoch 39/100
26/26 [==============================] - 0s 8ms/step - loss: 124.8427 -
mean_absolute_error: 8.1303 - val_loss: 201.5243 - val_mean_absolute_error:
9.1840
Epoch 40/100
26/26 [==============================] - 0s 8ms/step - loss: 122.2687 -
mean_absolute_error: 8.0143 - val_loss: 285.9185 - val_mean_absolute_error:
14.0102
Epoch 41/100
26/26 [==============================] - 0s 8ms/step - loss: 128.5870 -
mean_absolute_error: 8.4487 - val_loss: 241.0711 - val_mean_absolute_error:
10.7622
Epoch 42/100
26/26 [==============================] - 0s 9ms/step - loss: 144.6851 -
mean_absolute_error: 9.0708 - val_loss: 269.5070 - val_mean_absolute_error:
12.1778
Epoch 43/100
26/26 [==============================] - 0s 9ms/step - loss: 144.7680 -
mean_absolute_error: 8.9673 - val_loss: 281.2650 - val_mean_absolute_error:
12.3390
Epoch 44/100
26/26 [==============================] - 0s 9ms/step - loss: 166.0449 -
mean_absolute_error: 9.7934 - val_loss: 251.4678 - val_mean_absolute_error:
12.1553
Epoch 45/100
26/26 [==============================] - 0s 8ms/step - loss: 122.9152 -
mean_absolute_error: 8.1216 - val_loss: 315.1829 - val_mean_absolute_error:
13.8814
Epoch 46/100
26/26 [==============================] - 0s 8ms/step - loss: 134.7073 -
mean_absolute_error: 8.6140 - val_loss: 232.0075 - val_mean_absolute_error:
9.6392
Epoch 47/100
26/26 [==============================] - 0s 12ms/step - loss: 123.3405 -
mean_absolute_error: 8.1037 - val_loss: 234.4566 - val_mean_absolute_error:
12.0321
Epoch 48/100
```

```
26/26 [==============================] - 1s 27ms/step - loss: 139.1393 -
mean_absolute_error: 8.7122 - val_loss: 226.2381 - val_mean_absolute_error:
9.7101
Epoch 49/100
26/26 [==============================] - 0s 8ms/step - loss: 129.9930 -
mean_absolute_error: 8.4888 - val_loss: 400.7219 - val_mean_absolute_error:
17.0841
Epoch 50/100
26/26 [==============================] - 0s 6ms/step - loss: 134.7233 -
mean_absolute_error: 8.6200 - val_loss: 196.5728 - val_mean_absolute_error:
10.4564
Epoch 51/100
26/26 [==============================] - 0s 5ms/step - loss: 115.5324 -
mean_absolute_error: 7.6912 - val_loss: 179.8604 - val_mean_absolute_error:
8.5853
Epoch 52/100
26/26 [==============================] - 0s 6ms/step - loss: 126.7636 -
mean_absolute_error: 8.2791 - val_loss: 285.3101 - val_mean_absolute_error:
13.0291
Epoch 53/100
26/26 [==============================] - 0s 6ms/step - loss: 174.1700 -
mean_absolute_error: 9.8724 - val_loss: 198.6720 - val_mean_absolute_error:
9.3454
Epoch 54/100
26/26 [==============================] - 0s 5ms/step - loss: 135.8474 -
mean_absolute_error: 8.6971 - val_loss: 248.7760 - val_mean_absolute_error:
11.2760
Epoch 55/100
26/26 [==============================] - 0s 5ms/step - loss: 141.7928 -
mean_absolute_error: 9.0761 - val_loss: 179.8068 - val_mean_absolute_error:
9.2702
Epoch 56/100
26/26 [==============================] - 0s 5ms/step - loss: 131.4438 -
mean_absolute_error: 8.4372 - val_loss: 185.8106 - val_mean_absolute_error:
8.7765
Epoch 57/100
26/26 [==============================] - 0s 5ms/step - loss: 164.7827 -
mean_absolute_error: 9.4662 - val_loss: 208.1839 - val_mean_absolute_error:
9.3379
Epoch 58/100
26/26 [==============================] - 0s 5ms/step - loss: 120.1789 -
mean_absolute_error: 8.0302 - val_loss: 370.0333 - val_mean_absolute_error:
16.0233
Epoch 59/100
26/26 [==============================] - 0s 5ms/step - loss: 146.5215 -
mean_absolute_error: 9.0395 - val_loss: 204.1731 - val_mean_absolute_error:
10.5463
Epoch 60/100
```

```
26/26 [==============================] - 0s 6ms/step - loss: 127.8752 -
mean_absolute_error: 8.2559 - val_loss: 177.2521 - val_mean_absolute_error:
8.4993
Epoch 61/100
26/26 [==============================] - 0s 5ms/step - loss: 127.3617 -
mean_absolute_error: 8.0215 - val_loss: 170.1115 - val_mean_absolute_error:
8.3845
Epoch 62/100
26/26 [==============================] - 0s 6ms/step - loss: 121.1888 -
mean_absolute_error: 7.9866 - val_loss: 186.3846 - val_mean_absolute_error:
8.6958
Epoch 63/100
26/26 [==============================] - 0s 5ms/step - loss: 128.4416 -
mean_absolute_error: 8.4885 - val_loss: 256.2840 - val_mean_absolute_error:
12.2310
Epoch 64/100
26/26 [==============================] - 0s 6ms/step - loss: 117.2819 -
mean_absolute_error: 7.7766 - val_loss: 174.5816 - val_mean_absolute_error:
8.4774
Epoch 65/100
26/26 [==============================] - 0s 6ms/step - loss: 119.6409 -
mean_absolute_error: 7.9041 - val_loss: 249.5272 - val_mean_absolute_error:
12.0038
Epoch 66/100
26/26 [==============================] - 0s 5ms/step - loss: 138.1725 -
mean_absolute_error: 8.8449 - val_loss: 174.5928 - val_mean_absolute_error:
9.3277
Epoch 67/100
26/26 [==============================] - 0s 5ms/step - loss: 120.9965 -
mean_absolute_error: 7.8849 - val_loss: 333.8981 - val_mean_absolute_error:
15.2993
Epoch 68/100
26/26 [==============================] - 0s 5ms/step - loss: 173.5322 -
mean_absolute_error: 9.8697 - val_loss: 273.9355 - val_mean_absolute_error:
13.6015
Epoch 69/100
26/26 [==============================] - 0s 6ms/step - loss: 169.2060 -
mean_absolute_error: 9.8738 - val_loss: 192.1674 - val_mean_absolute_error:
8.8002
Epoch 70/100
26/26 [==============================] - 0s 6ms/step - loss: 127.6288 -
mean_absolute_error: 8.3910 - val_loss: 161.2447 - val_mean_absolute_error:
8.1436
Epoch 71/100
26/26 [==============================] - 0s 6ms/step - loss: 116.2683 -
mean_absolute_error: 7.7953 - val_loss: 169.5639 - val_mean_absolute_error:
8.6035
Epoch 72/100
```

```
26/26 [==============================] - 0s 6ms/step - loss: 140.9998 -
mean_absolute_error: 8.9246 - val_loss: 235.6805 - val_mean_absolute_error:
12.4371
Epoch 73/100
26/26 [==============================] - 0s 6ms/step - loss: 147.0506 -
mean_absolute_error: 9.1186 - val_loss: 260.6047 - val_mean_absolute_error:
13.4693
Epoch 74/100
26/26 [==============================] - 0s 6ms/step - loss: 147.9171 -
mean_absolute_error: 9.1503 - val_loss: 334.0437 - val_mean_absolute_error:
13.3708
Epoch 75/100
26/26 [==============================] - 0s 6ms/step - loss: 132.3647 -
mean_absolute_error: 8.4266 - val_loss: 173.8935 - val_mean_absolute_error:
8.5375
Epoch 76/100
26/26 [==============================] - 0s 6ms/step - loss: 116.6217 -
mean_absolute_error: 7.7895 - val_loss: 160.1826 - val_mean_absolute_error:
8.1590
Epoch 77/100
26/26 [==============================] - 0s 6ms/step - loss: 119.7691 -
mean_absolute_error: 7.8931 - val_loss: 164.9925 - val_mean_absolute_error:
8.2591
Epoch 78/100
26/26 [==============================] - 0s 7ms/step - loss: 117.7961 -
mean_absolute_error: 7.8479 - val_loss: 158.4522 - val_mean_absolute_error:
8.1553
Epoch 79/100
26/26 [==============================] - 0s 6ms/step - loss: 124.6997 -
mean_absolute_error: 8.1172 - val_loss: 158.8931 - val_mean_absolute_error:
8.1078
Epoch 80/100
26/26 [==============================] - 0s 6ms/step - loss: 172.3972 -
mean_absolute_error: 9.8032 - val_loss: 206.4727 - val_mean_absolute_error:
9.2698
Epoch 81/100
26/26 [==============================] - 0s 6ms/step - loss: 139.3251 -
mean_absolute_error: 8.7868 - val_loss: 206.3963 - val_mean_absolute_error:
10.2427
Epoch 82/100
26/26 [==============================] - 0s 6ms/step - loss: 123.9896 -
mean_absolute_error: 8.2499 - val_loss: 270.0038 - val_mean_absolute_error:
13.6096
Epoch 83/100
26/26 [==============================] - 0s 7ms/step - loss: 138.7762 -
mean_absolute_error: 8.9411 - val_loss: 184.7938 - val_mean_absolute_error:
9.8043
Epoch 84/100
```

```
26/26 [==============================] - 0s 6ms/step - loss: 157.5565 -
mean_absolute_error: 9.3807 - val_loss: 181.1399 - val_mean_absolute_error:
8.6562
Epoch 85/100
26/26 [==============================] - 0s 6ms/step - loss: 113.8167 -
mean_absolute_error: 7.7927 - val_loss: 429.5895 - val_mean_absolute_error:
18.1913
Epoch 86/100
26/26 [==============================] - 0s 6ms/step - loss: 136.2842 -
mean_absolute_error: 8.5810 - val_loss: 187.9289 - val_mean_absolute_error:
9.4659
Epoch 87/100
26/26 [==============================] - 0s 7ms/step - loss: 124.2476 -
mean_absolute_error: 8.0992 - val_loss: 189.0924 - val_mean_absolute_error:
10.4338
Epoch 88/100
26/26 [==============================] - 0s 7ms/step - loss: 120.9756 -
mean_absolute_error: 7.9098 - val_loss: 190.0829 - val_mean_absolute_error:
8.8320
Epoch 89/100
26/26 [==============================] - 0s 6ms/step - loss: 141.3713 -
mean_absolute_error: 9.0235 - val_loss: 195.8291 - val_mean_absolute_error:
8.9304
Epoch 90/100
26/26 [==============================] - 0s 9ms/step - loss: 119.0125 -
mean_absolute_error: 8.0502 - val_loss: 232.4824 - val_mean_absolute_error:
12.3532
Epoch 91/100
26/26 [==============================] - 0s 6ms/step - loss: 150.5720 -
mean_absolute_error: 9.1180 - val_loss: 189.7423 - val_mean_absolute_error:
8.7747
Epoch 92/100
26/26 [==============================] - 0s 6ms/step - loss: 129.7334 -
mean_absolute_error: 8.3602 - val_loss: 270.1428 - val_mean_absolute_error:
14.0312
Epoch 93/100
26/26 [==============================] - 0s 6ms/step - loss: 161.9862 -
mean_absolute_error: 9.7016 - val_loss: 244.2034 - val_mean_absolute_error:
10.7154
Epoch 94/100
26/26 [==============================] - 0s 6ms/step - loss: 128.9421 -
mean_absolute_error: 8.3233 - val_loss: 168.5362 - val_mean_absolute_error:
9.2309
Epoch 95/100
26/26 [==============================] - 0s 6ms/step - loss: 143.4112 -
mean_absolute_error: 8.8294 - val_loss: 202.5568 - val_mean_absolute_error:
10.6321
Epoch 96/100
```

```
26/26 [==============================] - 0s 7ms/step - loss: 131.5090 -
mean_absolute_error: 8.5372 - val_loss: 174.9052 - val_mean_absolute_error:
8.7091
Epoch 97/100
26/26 [==============================] - 0s 6ms/step - loss: 121.0583 -
mean_absolute_error: 8.0119 - val_loss: 197.3560 - val_mean_absolute_error:
11.1168
Epoch 98/100
26/26 [==============================] - 0s 6ms/step - loss: 141.8681 -
mean_absolute_error: 8.8961 - val_loss: 203.8097 - val_mean_absolute_error:
9.6278
Epoch 99/100
26/26 [==============================] - 0s 5ms/step - loss: 133.8441 -
mean_absolute_error: 8.5034 - val_loss: 176.0943 - val_mean_absolute_error:
9.6984
Epoch 100/100
26/26 [==============================] - 0s 6ms/step - loss: 132.1107 -
mean_absolute_error: 8.3304 - val_loss: 289.9989 - val_mean_absolute_error:
14.7398
```

[16]:

```
<keras.callbacks.History at 0x7b739c205090>
```

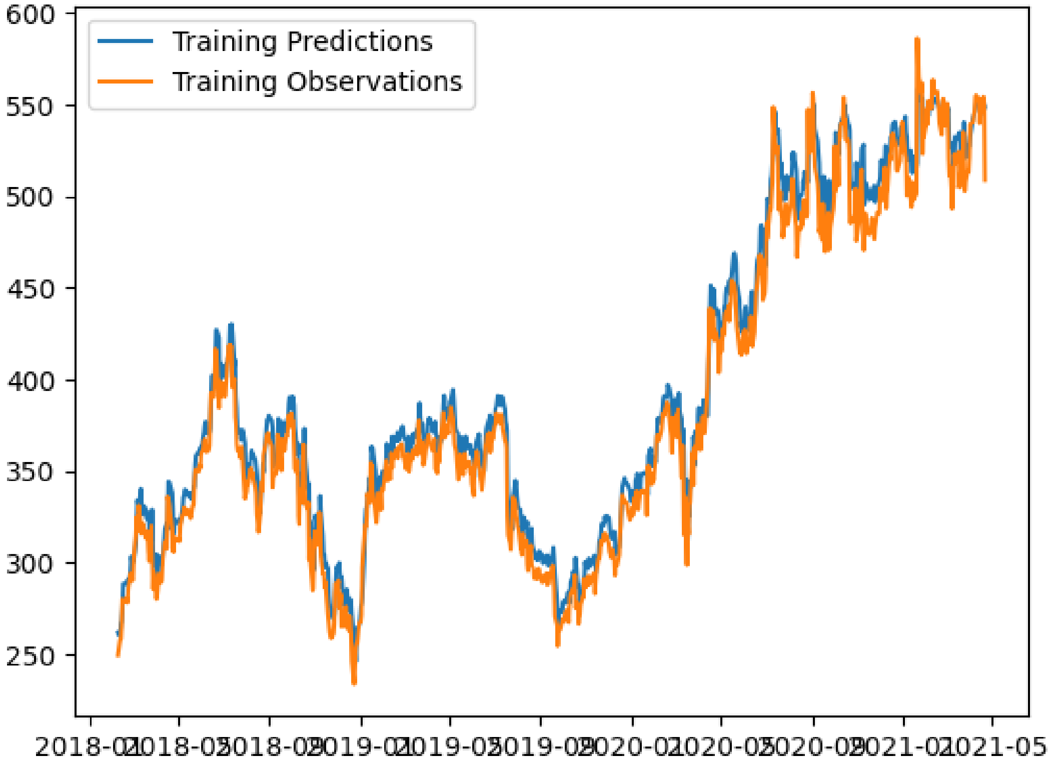
[17]:

```python
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'])
```

```
26/26 [==============================] - 1s 3ms/step
```

[17]:

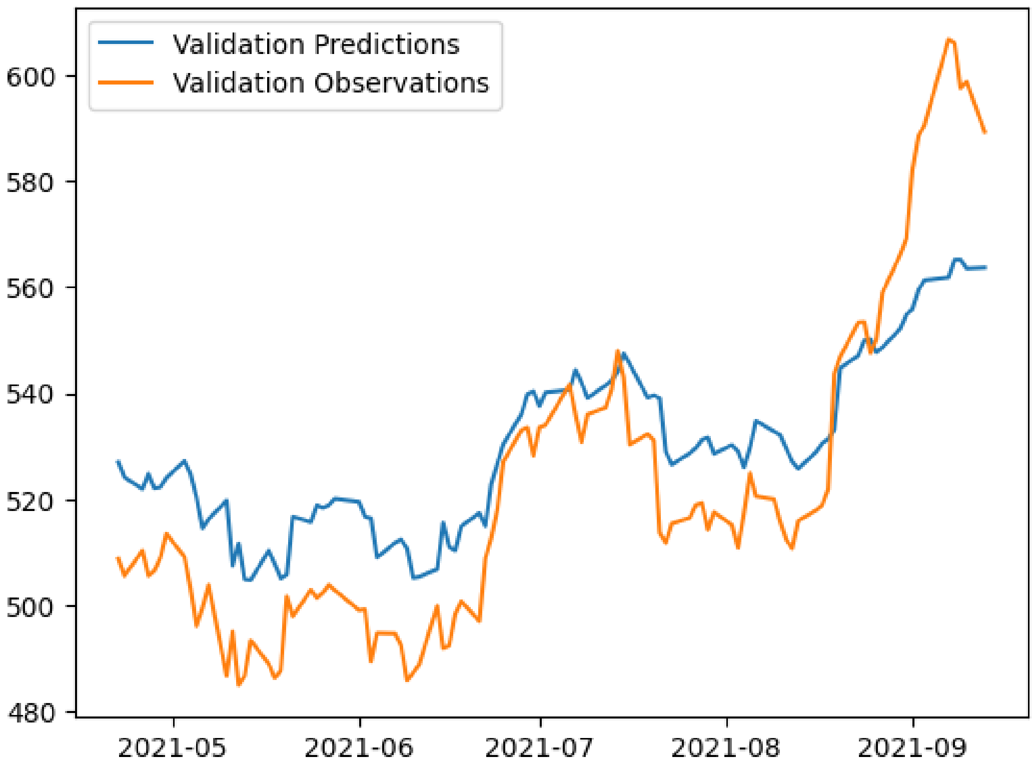
```
<matplotlib.legend.Legend at 0x7b73977763e0>
```



```
[18]: 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'])
```

```
4/4 [==============================] - 0s 4ms/step
```

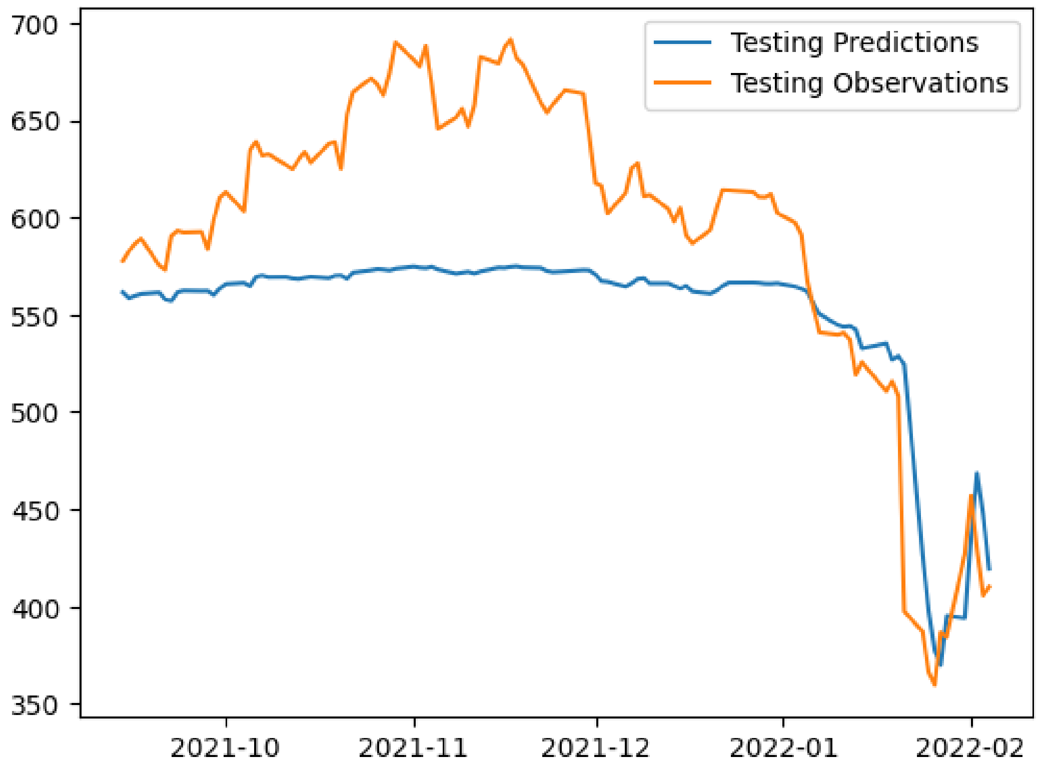
```
[18]: <matplotlib.legend.Legend at 0x7b7397626c50>
```



[19]:
```
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'])
```

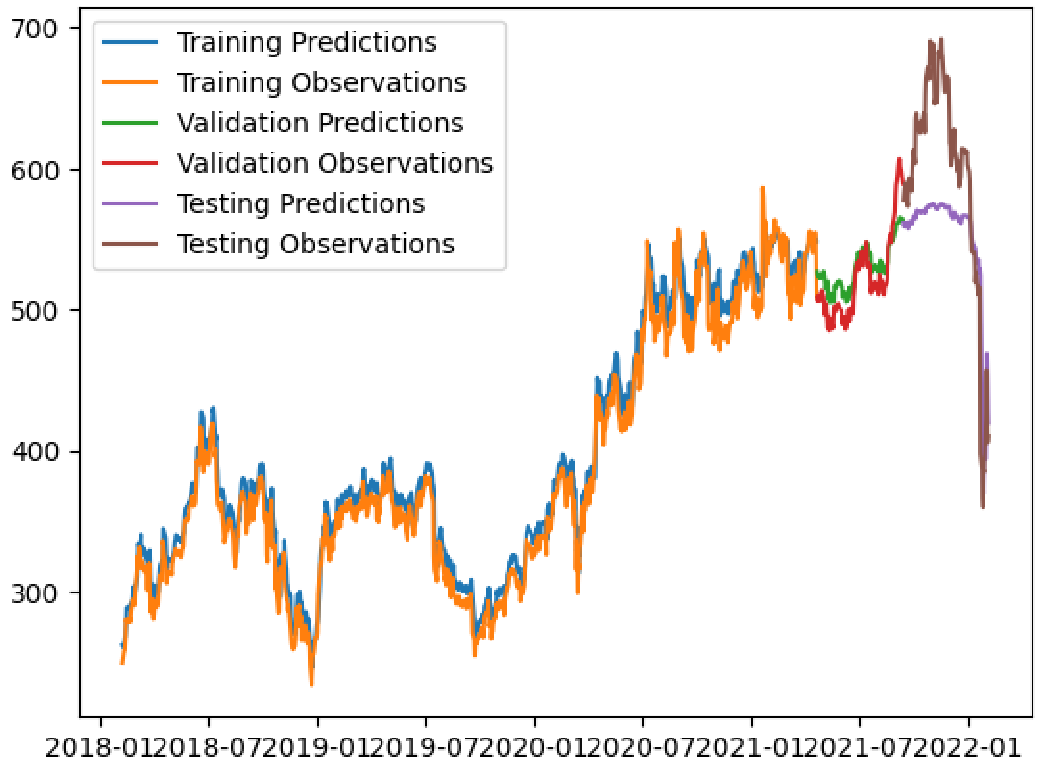
```
4/4 [==============================] - 0s 4ms/step
```

[19]:
```
<matplotlib.legend.Legend at 0x7b7397624310>
```



[20]:
```
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'])
```

[20]:
```
<matplotlib.legend.Legend at 0x7b7397519900>
```



[21]:

```
from copy import deepcopy

recursive_predictions = []
recursive_dates = np.concatenate([dates_val, dates_test])

for target_date in recursive_dates:
  last_window = deepcopy(X_train[-1])
  next_prediction = model.predict(np.array([last_window])).flatten()
  recursive_predictions.append(next_prediction)
  last_window[-1] = next_prediction
```

```
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 73ms/step
1/1 [==============================] - 0s 91ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 76ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 44ms/step
```

```
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
```

```
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 46ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 89ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 82ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 24ms/step
```

```
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 29ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 35ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 30ms/step
```

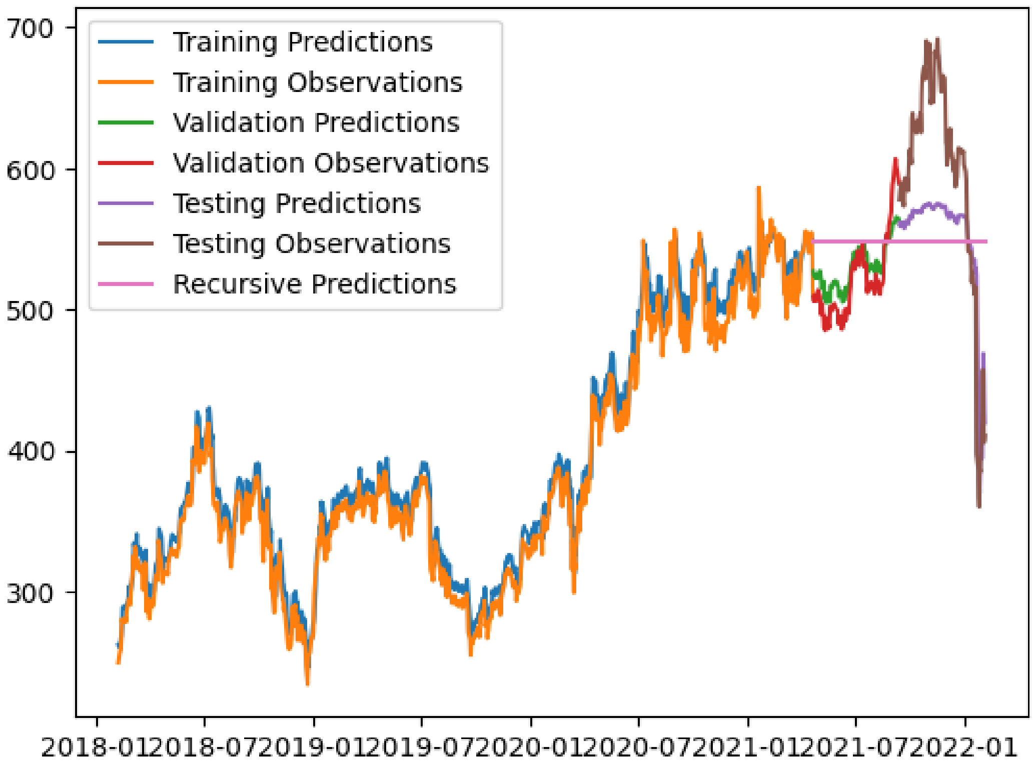
```
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 32ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 49ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 28ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
```

[22]:

```
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.plot(recursive_dates, recursive_predictions)
plt.legend(['Training Predictions',
            'Training Observations',
            'Validation Predictions',
            'Validation Observations',
            'Testing Predictions',
            'Testing Observations',
            'Recursive Predictions'])
```

[22]:

```
<matplotlib.legend.Legend at 0x7b73fb73e680>
```



[ ]: