```
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
from sklearn.preprocessing import MinMaxScaler
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout,GRU
from keras import optimizers
from sklearn.metrics import mean_squared_error
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

```
data_raw = pd.read_csv("VNM.csv", index_col="Date",
parse_dates=["Date"])
data_raw
```

```
             Open   High    Low  Close  Adj Close   Volume
Date
2018-12-31  14.85  14.97  14.70  14.75  14.372322   237500
2019-01-02  14.69  15.11  14.65  14.90  14.518480   267900
2019-01-03  14.76  14.83  14.40  14.48  14.109235   203300
2019-01-04  14.64  14.92  14.61  14.91  14.528225   187600
2019-01-07  14.94  15.09  14.84  15.04  14.654897   264400
...           ...    ...    ...    ...        ...      ...
2022-11-23  11.11  11.16  11.06  11.12  11.018559   450500
2022-11-25  11.45  11.60  11.37  11.57  11.464455   528600
2022-11-28  11.82  12.00  11.77  11.80  11.692356  1242400
2022-11-29  12.18  12.22  12.09  12.20  12.088708  1546400
2022-11-30  12.39  12.78  12.38  12.78  12.663417  1618300

[988 rows x 6 columns]
```

```
dataset = pd.DataFrame(data_raw["Close"])
```

```
from sklearn.preprocessing import MinMaxScaler
dataset_norm = data_raw.copy()
data_raw[['Close']]
scaler = MinMaxScaler()
dataset_norm['Close'] = scaler.fit_transform(data_raw[['Close']])
dataset_norm
```

```
             Open   High    Low     Close  Adj Close   Volume
Date
2018-12-31  14.85  14.97  14.70  0.426881  14.372322   237500
2019-01-02  14.69  15.11  14.65  0.439560  14.518480   267900
2019-01-03  14.76  14.83  14.40  0.404057  14.109235   203300
2019-01-04  14.64  14.92  14.61  0.440406  14.528225   187600
2019-01-07  14.94  15.09  14.84  0.451395  14.654897   264400
...           ...    ...    ...       ...        ...      ...
```

```
2022-11-23  11.11  11.16  11.06  0.120034  11.018559   450500
2022-11-25  11.45  11.60  11.37  0.158073  11.464455   528600
2022-11-28  11.82  12.00  11.77  0.177515  11.692356  1242400
2022-11-29  12.18  12.22  12.09  0.211327  12.088708  1546400
2022-11-30  12.39  12.78  12.38  0.260355  12.663417  1618300

[988 rows x 6 columns]
```

```
x =  dataset_norm.drop(["Volume"],axis=1).values
x = dataset_norm.drop(["Close"],axis=1).values
print(x)
```

```
[[1.4850000e+01 1.4970000e+01 1.4700000e+01 1.4372322e+01
2.3750000e+05]
 [1.4690000e+01 1.5110000e+01 1.4650000e+01 1.4518480e+01
2.6790000e+05]
 [1.4760000e+01 1.4830000e+01 1.4400000e+01 1.4109235e+01
2.0330000e+05]
 ...
 [1.1820000e+01 1.2000000e+01 1.1770000e+01 1.1692356e+01
1.2424000e+06]
 [1.2180000e+01 1.2220000e+01 1.2090000e+01 1.2088708e+01
1.5464000e+06]
 [1.2390000e+01 1.2780000e+01 1.2380000e+01 1.2663417e+01
1.6183000e+06]]
```

```
y = dataset_norm["Close"].values
print(y)
```

```
[0.42688078 0.4395604  0.40405745 0.44040571 0.45139472 0.44463225
 0.46069311 0.47252743 0.47337274 0.46745558 0.47844459 0.4792899
 0.48098052 0.48774298 0.4691462  0.49281484 0.48774298 0.51648347
 0.52324594 0.51141162 0.52831779 0.5190194  0.49788669 0.50549446
 0.50803039 0.50718508 0.48182583 0.4894336  0.50380385 0.53338964
 0.5393068  0.56297552 0.57565507 0.58579868 0.59509716 0.59340646
 0.61707518 0.62299234 0.61538456 0.62130164 0.59002531 0.59509716
 0.60608625 0.59678778 0.5908707  0.55874881 0.54522396 0.58241753
 0.61622987 0.62975481 0.61453917 0.60439555 0.63060012 0.62299234
 0.61200341 0.59763309 0.57142844 0.56466597 0.58326284 0.57227375
 0.57227375 0.57988152 0.61792049 0.60270494 0.61369386 0.6204565
 0.62468296 0.62299234 0.61284872 0.60439555 0.61369386 0.63229073
 0.61622987 0.61369386 0.60101423 0.59002531 0.58579868 0.60524095
 0.60101423 0.59932379 0.59171601 0.5908707  0.58918    0.57480976
 0.56720198 0.57650037 0.56466597 0.54691458 0.55114104 0.53846149
 0.55705836 0.54437857 0.56635667 0.58579868 0.59594247 0.56804729
 0.57396445 0.58664399 0.57142844 0.56804729 0.55198643 0.54860528
 0.56720198 0.54691458 0.54860528 0.53846149 0.55283174 0.55114104
 0.55029573 0.56382083 0.56043951 0.56382083 0.55452235 0.55114104
 0.55621306 0.5393068  0.54353326 0.55874881 0.56973799 0.56382083
 0.57142844 0.5595942  0.55621306 0.53508034 0.53423495 0.55536766
 0.55874881 0.55452235 0.55621306 0.54184281 0.5393068  0.55705836
```

```
0.55198643 0.5688926  0.55536766 0.57142844 0.5595942  0.55621306
0.56382083 0.56466597 0.5705833  0.56720198 0.56551128 0.57565507
0.56720198 0.54268812 0.54775989 0.5291631  0.53846149 0.47168212
0.46999151 0.49704138 0.51817409 0.50295854 0.48182583 0.49281484
0.46830089 0.48436175 0.49704138 0.50126792 0.50549446 0.51986471
0.52493656 0.49788669 0.49619607 0.4894336  0.49619607 0.49535076
0.50549446 0.50295854 0.50972101 0.5190194  0.51648347 0.51056631
0.50295854 0.50380385 0.50972101 0.51986471 0.53254433 0.53423495
0.53254433 0.5393068  0.53592565 0.53254433 0.53169903 0.5376161
0.54945059 0.54522396 0.55874881 0.54691458 0.53085372 0.54099742
0.55283174 0.5393068  0.52662717 0.53254433 0.5393068  0.54691458
0.53846149 0.55367705 0.55452235 0.54945059 0.54437857 0.54945059
0.55029573 0.55029573 0.54691458 0.55705836 0.55452235 0.55114104
0.55536766 0.55114104 0.58495354 0.5908707  0.58918    0.58833469
0.58410823 0.58410823 0.57819108 0.57819108 0.56466597 0.56213013
0.5705833  0.56551128 0.56213013 0.5579035  0.54015211 0.53508034
0.5393068  0.54775989 0.54437857 0.52747248 0.51479286 0.51141162
0.52578187 0.51986471 0.52662717 0.51648347 0.51817409 0.52493656
0.54775989 0.54015211 0.53169903 0.52240063 0.52071002 0.51394755
0.52240063 0.51479286 0.51986471 0.52155532 0.52155532 0.52493656
0.53169903 0.55452235 0.54015211 0.53592565 0.53338964 0.52071002
0.51986471 0.52493656 0.54099742 0.53846149 0.53085372 0.53338964
0.53677079 0.53169903 0.53592565 0.5393068  0.53338964 0.50718508
0.49788669 0.50126792 0.49196953 0.43871509 0.44463225 0.45646657
0.45308534 0.4590025  0.45224003 0.44801349 0.46238373 0.47337274
0.47168212 0.47083682 0.45393065 0.4497041  0.44801349 0.44463225
0.38038881 0.37024511 0.37278103 0.33812339 0.34150462 0.35249363
0.34065931 0.36179202 0.33727808 0.32882499 0.25274723 0.26458155
0.2299239  0.11749788 0.12679627 0.04818258 0.08453085 0.02874049
0.0701606  0.06762468 0.         0.06677937 0.09721047 0.11327133
0.06931529 0.06424344 0.06255283 0.05325443 0.0803043  0.08791208
0.16060861 0.16652577 0.18343194 0.19188502 0.18681317 0.1994928
0.1893491  0.20287403 0.21555366 0.20794588 0.16821638 0.20710057
0.19526626 0.22400674 0.21555366 0.22654267 0.23245983 0.23245983
0.18765848 0.19103971 0.22316144 0.22907859 0.26035501 0.29754858
0.26796278 0.28148772 0.25528316 0.27049871 0.2696534  0.30515636
0.28233303 0.33727808 0.32713438 0.32290783 0.36010141 0.34319524
0.34234993 0.33812339 0.36770918 0.37109042 0.38038881 0.36432795
0.37616227 0.4091293  0.39814029 0.40574806 0.33474215 0.34234993
0.31276413 0.32713438 0.33389684 0.34234993 0.34319524 0.34826709
0.35418425 0.32628907 0.32882499 0.32037191 0.32459845 0.31614537
0.34319524 0.34826709 0.37616227 0.37024511 0.3795435  0.38123412
0.38292474 0.37616227 0.39560436 0.419273   0.40743868 0.40574806
0.41251053 0.40236683 0.40490276 0.38377004 0.37024511 0.33812339
0.35418425 0.3499577  0.35249363 0.33812339 0.37785288 0.39391375
0.42349954 0.41758238 0.41673707 0.42772608 0.4091293  0.42519016
0.42011831 0.42265423 0.43110732 0.39560436 0.38461535 0.37193573
0.37447165 0.40321214 0.42349954 0.42857139 0.40997461 0.42603547
0.42265423 0.43533386 0.44801349 0.43617917 0.4488588  0.43110732
0.46153842 0.42772608 0.43026201 0.4590025  0.46491966 0.44632287
```

```
0.44294164 0.43279793 0.42265423 0.42434485 0.40574806 0.40152152
0.4091293  0.43110732 0.419273   0.43533386 0.44378694 0.43533386
0.44632287 0.43279793 0.45646657 0.45477595 0.46069311 0.46491966
0.46745558 0.46745558 0.46830089 0.46407435 0.44040571 0.45984781
0.45308534 0.46153842 0.48520706 0.45731188 0.45139472 0.41251053
0.43279793 0.42603547 0.4395604  0.45393065 0.48689768 0.47590867
0.47844459 0.49366014 0.49366014 0.4995773  0.49366014 0.49788669
0.49027891 0.49788669 0.48774298 0.50549446 0.53592565 0.54099742
0.55452235 0.53592565 0.55536766 0.5291631  0.54522396 0.56128482
0.5688926  0.56804729 0.57142844 0.57396445 0.57734577 0.58918
0.59763309 0.61284872 0.61622987 0.62214695 0.60693139 0.60524095
0.60270494 0.58495354 0.59678778 0.59425185 0.62552826 0.6584953
0.6593406  0.6584953  0.66356706 0.69822471 0.70160603 0.71935742
0.73710898 0.74471676 0.75232453 0.75147923 0.74387145 0.73879968
0.6889264  0.7109045  0.73288252 0.73119182 0.72020273 0.71513096
0.64666098 0.60270494 0.61031271 0.62130164 0.65004213 0.69315294
0.6889264  0.69484356 0.65680459 0.68047332 0.68469977 0.69568887
0.70245134 0.68723578 0.70836849 0.71851228 0.73457297 0.71766697
0.70667779 0.70160603 0.65088752 0.66779369 0.70583248 0.69315294
0.66779369 0.64835168 0.67624686 0.6593406  0.66356706 0.66779369
0.70076072 0.68300933 0.68977179 0.69399825 0.69653426 0.68554508
0.69230755 0.69653426 0.66441237 0.65088752 0.66356706 0.68808109
0.69230755 0.6906171  0.71005911 0.74809791 0.75570568 0.74387145
0.7092138  0.75147923 0.75232453 0.77937441 0.78698218 0.78698218
0.80135243 0.8106509  0.83178362 0.83685538 0.83685538 0.8089602
0.82333053 0.79458996 0.79797111 0.78698218 0.79458996 0.77345725
0.75316993 0.76162284 0.77937441 0.77430264 0.79036333 0.80388827
0.78782749 0.759087   0.74725277 0.78191042 0.76584947 0.76500416
0.77345725 0.7971258  0.79458996 0.81910382 0.81234135 0.83262875
0.82502098 0.84361785 0.84530855 0.85460677 0.86052393 0.88165665
0.8689771  0.84530855 0.86052393 0.85883349 0.87066772 0.88588327
0.88419266 0.85545217 0.87573949 0.87151303 0.88419266 0.88334735
0.87658488 0.8799662  0.91293314 0.92138623 0.92392207 0.93744717
0.94420964 0.94590009 0.90194422 0.90448006 0.87320373 0.86475056
0.82333053 0.80388827 0.80726959 0.81741337 0.81825851 0.76331354
0.7988165  0.80642428 0.81234135 0.79205403 0.80135243 0.8089602
0.81318674 0.81656806 0.83262875 0.82924761 0.86221464 0.85798809
0.87235834 0.85460677 0.88841911 0.86982241 0.86644126 0.85376147
0.85714278 0.846999   0.82248514 0.81994921 0.82586629 0.79966181
0.81741337 0.8098056  0.82079452 0.80557897 0.85038032 0.85883349
0.84953501 0.86390525 0.87066772 0.88757397 0.87320373 0.85122563
0.87320373 0.87320373 0.85629748 0.84953501 0.85545217 0.85967879
0.85714278 0.81825851 0.82924761 0.85798809 0.84784439 0.83516476
0.80388827 0.80304313 0.79205403 0.8089602  0.79797111 0.79458996
0.81149604 0.80388827 0.82586629 0.83685538 0.84023653 0.83770077
0.83685538 0.83939122 0.8284023  0.83939122 0.84615369 0.84615369
0.83431946 0.84023653 0.84615369 0.846999   0.8791208  0.89095512
0.89602689 0.90194422 0.8985629  0.88672867 0.88757397 0.90617068
0.92730339 0.91800491 0.92054092 0.92138623 0.93068454 0.93406586
0.93322055 0.94251894 0.94251894 0.91631446 0.89433644 0.91800491
```

```
 0.93152985 0.9281487  0.92730339 0.93406586 0.93829231 0.93152985
 0.90194422 0.87151303 0.91377845 0.9298394  0.94251894 0.9281487
 0.9281487  0.92899409 0.93322055 0.93068454 0.93575647 0.92138623
 0.94505478 0.95435326 0.92645808 0.95604388 0.95942502 0.95604388
 0.97717659 0.97548597 0.9780219  1.         0.99661868 0.98816551
 0.98901091 0.97210482 0.96027033 0.96280634 0.93237524 0.93322055
 0.84361785 0.84953501 0.86305994 0.85714278 0.79628066 0.82671168
 0.81403205 0.81741337 0.83685538 0.85122563 0.85629748 0.86136924
 0.85038032 0.846999   0.8486897  0.84108192 0.87320373 0.85967879
 0.83854608 0.81994921 0.84108192 0.85460677 0.85545217 0.85291633
 0.84023653 0.84361785 0.8284023  0.83601007 0.82924761 0.82248514
 0.83262875 0.8486897  0.84023653 0.80642428 0.78360086 0.79628066
 0.78360086 0.74640746 0.71851228 0.73541837 0.76584947 0.75401524
 0.77345725 0.80219782 0.8089602  0.80473358 0.80304313 0.7988165
 0.77937441 0.79543535 0.78613687 0.77514795 0.80219782 0.82417584
 0.8089602  0.79120864 0.78529157 0.76669478 0.7599324  0.74894322
 0.76500416 0.74725277 0.70160603 0.67624686 0.66018591 0.64750637
 0.6390532  0.58326284 0.59340646 0.61369386 0.6086221  0.6204565
 0.62552826 0.63567188 0.62214695 0.60524095 0.58495354 0.51056631
 0.54437857 0.54015211 0.47168212 0.46745558 0.44463225 0.50295854
 0.48351644 0.498732   0.48858829 0.47421805 0.48182583 0.50803039
 0.51056631 0.52409125 0.53592565 0.5376161  0.53423495 0.51648347
 0.51563816 0.52409125 0.5291631  0.52662717 0.50295854 0.44378694
 0.45224003 0.44632287 0.43871509 0.43279793 0.4091293  0.40152152
 0.41842769 0.43026201 0.43195263 0.43279793 0.42349954 0.39729498
 0.40405745 0.37109042 0.34657647 0.37700758 0.38715128 0.37869819
 0.39222313 0.38123412 0.39222313 0.3998309  0.38630597 0.40743868
 0.41504646 0.419273   0.40574806 0.41166522 0.39222313 0.41504646
 0.43110732 0.42519016 0.45054941 0.45308534 0.46661027 0.46999151
 0.47252743 0.47844459 0.48351644 0.49112422 0.48436175 0.50464915
 0.50803039 0.50549446 0.51056631 0.50211323 0.48605237 0.47168212
 0.48774298 0.498732   0.51394755 0.47675397 0.46830089 0.46238373
 0.47590867 0.46830089 0.46238373 0.45224003 0.43786979 0.43617917
 0.45224003 0.46069311 0.419273   0.4294167  0.41842769 0.40659337
 0.39475905 0.39729498 0.39306844 0.38038881 0.36770918 0.32882499
 0.32544376 0.32121722 0.28571426 0.30431105 0.26711748 0.26711748
 0.28064241 0.25105661 0.22907859 0.2299239  0.19695687 0.22231613
 0.23837699 0.21808958 0.22231613 0.22907859 0.21301773 0.20540996
 0.18343194 0.14032121 0.1394759  0.1496196  0.17497885 0.17328824
 0.16737108 0.15553676 0.15046491 0.14116651 0.13863059 0.11411664
 0.13102281 0.11073541 0.10650887 0.12764158 0.10481825 0.10143702
 0.13186812 0.15046491 0.14201182 0.12426034 0.12256973 0.1200338
 0.15807268 0.17751478 0.21132712 0.26035501]
```

```
totaldata = data_raw.values
totaldatatrain = int(len(totaldata)*0.75)
totaldatatest = int(len(totaldata)*0.25)

training_set = dataset_norm[0:totaldatatrain]
test_set = dataset_norm[totaldatatrain:]
```

```
#Sliding windows
lag = 2
# sliding windows function
def create_sliding_windows(data,len_data,lag):
    x=[]
    y=[]
    for i in range(lag,len_data):
        x.append(data[i-lag:i,0])
        y.append(data[i,0])
    return np.array(x),np.array(y)

# Formating data into array for create sliding windows
array_training_set = np.array(training_set)
array_test_set = np.array(test_set)

# Create sliding windows into training data
x_train, y_train =
create_sliding_windows(array_training_set,len(array_training_set),
lag)
# Create sliding windows into test data
x_test,y_test =
create_sliding_windows(array_test_set,len(array_test_set),lag)
```

```
from sklearn.linear_model import LinearRegression
ml=LinearRegression()
ml.fit(x_train, y_train)
```

```
LinearRegression()
```

```
y_pred=ml.predict(x_test)
print(y_pred)
```

```
[20.76468449 20.76917438 20.83053297 20.64739597 20.67282827
20.77690392
 20.72400306 20.5277084  20.73382164 20.86572844 20.7414974
20.76353459
 20.98411237 21.01570522 21.00761109 21.24155729 21.27503094
21.51085713
 21.54432807 21.49106424 21.4041179  21.2691812  21.04823892
21.2344032
 20.92175996 20.74243735 20.06791023 19.74633876 19.90973674
19.84816901
 19.36993718 19.33009393 19.53881688 19.3526526  19.39427246
19.51829608
 19.792503   19.89203408 19.81824602 19.79172003 19.78984032
19.68143042
 19.95020751 19.87093783 19.80545223 19.52879078 19.60900231
19.64738313
 19.82185045 19.80357224 19.61009752 19.85234619 19.18398204
19.5181381
 19.48816385 19.55291611 19.42492527 19.83542711 19.5724476
```

```
19.5043523
 19.13019502 19.01035039 19.18257341 18.93034909 18.43237733
18.26605774
 18.43452106 18.67013811 18.71979732 19.15625475 19.22847382
19.29453124
 19.28304443 19.20889206 18.9755204  19.06042978 19.01750457
19.05703439
 19.14304169 19.33218297 19.42325461 19.15975081 18.97797285
18.88972106
 18.66538349 18.62475837 18.73128675 18.72449697 18.36125457
17.85388352
 17.68756392 17.52176528 17.478683   16.68989877 16.83397541
16.91167862
 16.95345554 17.05658853 17.0578937  17.13826049 17.04734693
17.00990453
 16.73606301 16.06644387 16.14425275 16.07798428 15.52748059
15.10068638
 15.0531144  15.52194676 15.63406266 15.54126739 15.69354145
15.43040396
 15.40126555 15.70842436 15.76748486 15.90581595 16.11866409
16.20258326
 16.03751461 15.95641691 15.9859736  15.86325624 16.03077952
16.07667993
 15.83557945 15.28622404 15.12329803 14.93133699 14.95770843
14.82689693
 14.66825238 14.52673605 14.58584943 14.69274393 14.98313992
15.00324425
 14.87582714 14.56454313 14.41608175 14.21414668 14.00469203
14.08432802
 14.16694011 14.28286869 14.32125033 14.25179751 14.27138017
14.2832341
 14.40161712 14.43190462 14.62350025 14.69744367 14.62177662 14.604283
 14.49493406 14.42945018 14.70386705 14.72397137 14.93081533
15.0789113
 15.15530916 15.20705921 15.22079304 15.37531239 15.4545301
15.5333824
 15.45959524 15.5595447  15.69730124 15.7194946  15.75657088
15.6428876
 15.54032744 15.33217814 15.42894234 15.56763883 15.64403669
15.67771858
 15.31014104 15.22643273 15.35724423 15.27196855 15.30679953
15.18502211
 14.90084035 14.86360781 14.9981791  15.11165324 14.98648143
14.8730073
 14.77232703 14.60934814 14.43075553 14.41383645 14.3867342
14.29357353
 14.13059464 13.84359303 13.61664476 13.44238648 13.28186203
13.0861412
 12.99506957 12.91527732 12.98003039 12.74362975 12.51950131
12.57109511
```

```
 12.16304689 12.21015008 12.21051549 12.4896321  12.32905477
12.4899975
 12.36425115 12.29500748 11.90993631 11.491027   11.3792765
11.29045016
 11.76079701 11.77447795 11.6984455  11.80836898 11.60377033
11.5089389
 11.39301032 11.2462197  11.13274556 11.19467879 11.25640287
11.09039499
 11.09963822 11.0301854  11.15817705 11.47943509 11.61076826
11.3049148
 11.18167577 11.13613995 11.43953897 11.80575828 12.16576336]
```

```
set_test = dataset["Close"]
set_test
```

```
Date
2018-12-31    14.75
2019-01-02    14.90
2019-01-03    14.48
2019-01-04    14.91
2019-01-07    15.04
              ...
2022-11-23    11.12
2022-11-25    11.57
2022-11-28    11.80
2022-11-29    12.20
2022-11-30    12.78
Name: Close, Length: 988, dtype: float64
```

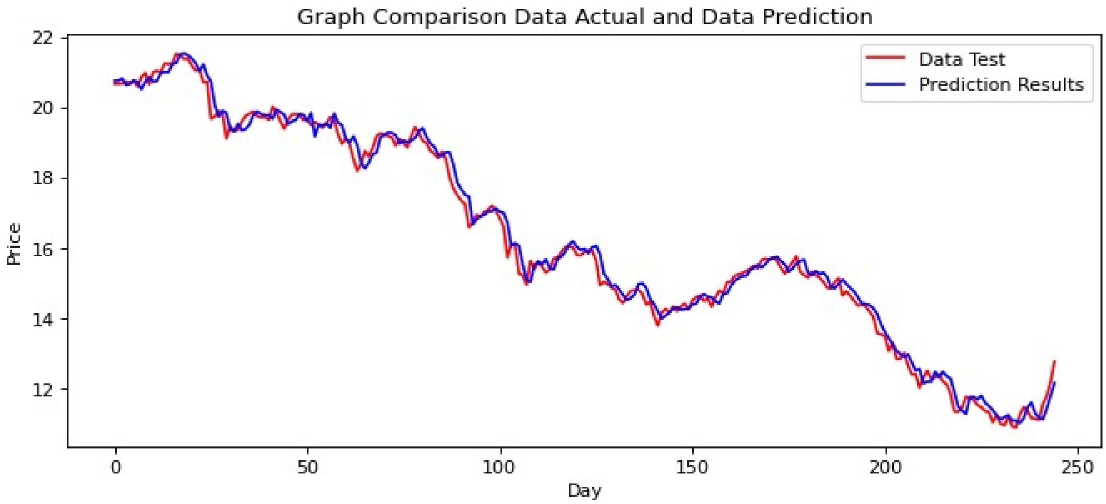
```
datacompare = pd.DataFrame()
datatest=np.array(set_test[totaldatatrain+lag:])
datapred= y_pred

datacompare['Data Test'] = datatest
datacompare['Prediction Results'] = datapred
datacompare
```

```
     Data Test  Prediction Results
0    20.680000           20.764684
1    20.680000           20.769174
2    20.690001           20.830533
3    20.740000           20.647396
4    20.709999           20.672828
..         ...                 ...
240  11.120000           11.181676
241  11.570000           11.136140
242  11.800000           11.439539
243  12.200000           11.805758
244  12.780000           12.165763

[245 rows x 2 columns]
```

```
plt.figure(num=None, figsize=(10, 4), dpi=80,facecolor='w',
edgecolor='k')
plt.title('Graph Comparison Data Actual and Data Prediction')
plt.plot(datacompare['Data Test'], color='red',label='Data Test')
plt.plot(datacompare['Prediction Results'],
color='blue',label='Prediction Results')
plt.xlabel('Day')
plt.ylabel('Price')
plt.legend()
plt.show()
```



```
def MAPE(Y_actual,Y_Predicted):
    mape = np.mean(np.abs((Y_actual - Y_Predicted)/Y_actual))*100
    return mape
MAPE(datatest, datapred)
```

```
1.3600106153506502
```

```
from sklearn.metrics import mean_squared_error
import math
MSE = mean_squared_error(datatest, datapred)
RMSE = math.sqrt(MSE)
print(RMSE)
```

```
0.2794239373188524
```

```
from sklearn.metrics import mean_absolute_error

mean_absolute_error(
    y_true=datatest,
    y_pred=datapred
)
```

```
0.21201817537491477
```