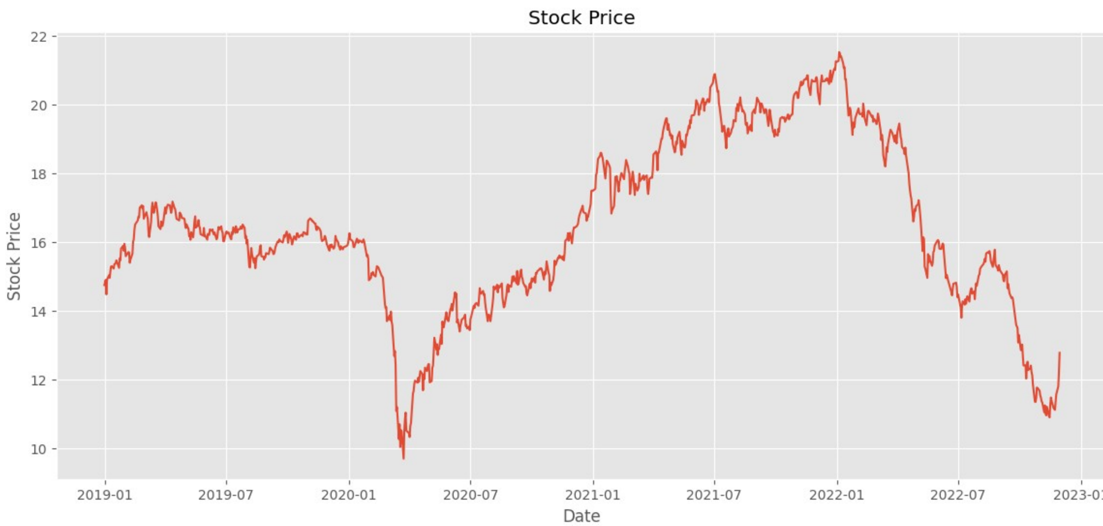
```python
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
from sklearn.preprocessing import MinMaxScaler
from tensorflow import keras
import keras.backend as K
from keras.models import Sequential
from keras.layers import Dense, Dropout, LSTM, Layer, Bidirectional,
Attention
from keras import optimizers
from sklearn.metrics import mean_squared_error

seed = 1234
np.random.seed(seed)
plt.style.use('ggplot')

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

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

dataset = pd.DataFrame(data_raw['Close'])
print(' Count row of data: ',len(dataset))

fig = plt.figure(figsize=(14, 6))
plt.plot(dataset)
plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.title('Stock Price')
plt.show()
```

```
 Count row of data:  988
```




```
dataset_norm = dataset.copy()
dataset[['Close']]
scaler = MinMaxScaler()
dataset_norm['Close'] = scaler.fit_transform(dataset[['Close']])
dataset_norm
```

```
               Close
Date
2018-12-31  0.426881
2019-01-02  0.439560
2019-01-03  0.404057
2019-01-04  0.440406
2019-01-07  0.451395
...              ...
2022-11-23  0.120034
2022-11-25  0.158073
2022-11-28  0.177515
2022-11-29  0.211327
2022-11-30  0.260355

[988 rows x 1 columns]
```

```
totaldata = dataset.values
totaldatatrain = int(len(totaldata)*0.75)
totaldataval = int(len(totaldata)*0.1)
totaldatatest = int(len(totaldata)*0.15)

# Store data into each partition
training_set = dataset_norm[0:totaldatatrain]
val_set=dataset_norm[totaldatatrain:totaldatatrain+totaldataval]
test_set = dataset_norm[totaldatatrain+totaldataval:]

# Initiaton value of lag
lag = 2
```

```python
# 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_val_set = np.array(val_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)
x_train = np.reshape(x_train, (x_train.shape[0],x_train.shape[1],1))
# Create sliding windows into validation data
x_val,y_val =
create_sliding_windows(array_val_set,len(array_val_set),lag)
x_val = np.reshape(x_val, (x_val.shape[0],x_val.shape[1],1))
# Create sliding windows into test data
x_test,y_test =
create_sliding_windows(array_test_set,len(array_test_set),lag)
x_test = np.reshape(x_test, (x_test.shape[0],x_test.shape[1],1))

# Hyperparameters
learning_rate = 0.0001
hidden_unit = 64
batch_size=64
epoch = 100

# Architecture Gated Recurrent Unit
model = Sequential()

# First GRU layer with dropout
model.add(Bidirectional(LSTM(units=hidden_unit, return_sequences=True,
input_shape=(x_train.shape[1],1), activation = 'relu')))
model.add(Dropout(0.2))
# Second GRU layer with dropout
model.add(Bidirectional(LSTM(units=hidden_unit, return_sequences=True,
activation = 'relu')))
model.add(Dropout(0.2))
# Third GRU layer with dropout
model.add(Bidirectional(LSTM(units=hidden_unit,
return_sequences=False, activation = 'relu')))
model.add(Dropout(0.2))
```

```python
# Output layer
model.add(Dense(units=1))

# Compiling the Gated Recurrent Unit
model.compile(optimizer=tf.keras.optimizers.Adam(lr=learning_rate),los
s='mean_squared_error')

# Fitting ke data training dan data validation
pred = model.fit(x_train, y_train, validation_data=(x_val,y_val),
batch_size=batch_size, epochs=epoch)
```

```
WARNING:absl:`lr` is deprecated, please use `learning_rate` instead,
or use the legacy optimizer, e.g.,tf.keras.optimizers.legacy.Adam.

Epoch 1/100
12/12 [==============================] - 8s 78ms/step - loss: 0.3242 -
val_loss: 0.5151
Epoch 2/100
12/12 [==============================] - 0s 16ms/step - loss: 0.1615 -
val_loss: 0.1053
Epoch 3/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0311 -
val_loss: 0.0032
Epoch 4/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0185 -
val_loss: 0.0484
Epoch 5/100
12/12 [==============================] - 0s 11ms/step - loss: 0.0151 -
val_loss: 0.0071
Epoch 6/100
12/12 [==============================] - 0s 11ms/step - loss: 0.0118 -
val_loss: 0.0155
Epoch 7/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0099 -
val_loss: 0.0035
Epoch 8/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0070 -
val_loss: 0.0036
Epoch 9/100
12/12 [==============================] - 0s 19ms/step - loss: 0.0051 -
val_loss: 6.0298e-04
Epoch 10/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0040 -
val_loss: 9.1905e-04
Epoch 11/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0042 -
val_loss: 5.5649e-04
Epoch 12/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0037 -
val_loss: 8.0275e-04
```

```
Epoch 13/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0035 -
val_loss: 7.0017e-04
Epoch 14/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0037 -
val_loss: 5.0966e-04
Epoch 15/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0035 -
val_loss: 5.5732e-04
Epoch 16/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0034 -
val_loss: 5.5892e-04
Epoch 17/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0032 -
val_loss: 5.7585e-04
Epoch 18/100
12/12 [==============================] - 0s 17ms/step - loss: 0.0032 -
val_loss: 6.4938e-04
Epoch 19/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0031 -
val_loss: 8.1603e-04
Epoch 20/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0030 -
val_loss: 8.5383e-04
Epoch 21/100
12/12 [==============================] - 0s 11ms/step - loss: 0.0033 -
val_loss: 7.6687e-04
Epoch 22/100
12/12 [==============================] - 0s 11ms/step - loss: 0.0030 -
val_loss: 6.2468e-04
Epoch 23/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0031 -
val_loss: 5.5692e-04
Epoch 24/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0029 -
val_loss: 7.0514e-04
Epoch 25/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0029 -
val_loss: 7.8985e-04
Epoch 26/100
12/12 [==============================] - 0s 18ms/step - loss: 0.0033 -
val_loss: 7.8555e-04
Epoch 27/100
12/12 [==============================] - 0s 19ms/step - loss: 0.0031 -
val_loss: 0.0011
Epoch 28/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0031 -
val_loss: 7.6222e-04
Epoch 29/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0031 -
```

```
val_loss: 6.4975e-04
Epoch 30/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0030 -
val_loss: 6.2320e-04
Epoch 31/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0028 -
val_loss: 0.0011
Epoch 32/100
12/12 [==============================] - 0s 17ms/step - loss: 0.0029 -
val_loss: 7.0678e-04
Epoch 33/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0026 -
val_loss: 5.7406e-04
Epoch 34/100
12/12 [==============================] - 0s 18ms/step - loss: 0.0029 -
val_loss: 6.1102e-04
Epoch 35/100
12/12 [==============================] - 0s 19ms/step - loss: 0.0028 -
val_loss: 7.0406e-04
Epoch 36/100
12/12 [==============================] - 0s 21ms/step - loss: 0.0030 -
val_loss: 6.5866e-04
Epoch 37/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0026 -
val_loss: 6.8819e-04
Epoch 38/100
12/12 [==============================] - 0s 11ms/step - loss: 0.0028 -
val_loss: 6.5058e-04
Epoch 39/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0031 -
val_loss: 6.6101e-04
Epoch 40/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0029 -
val_loss: 5.8701e-04
Epoch 41/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0028 -
val_loss: 5.9732e-04
Epoch 42/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0030 -
val_loss: 7.8076e-04
Epoch 43/100
12/12 [==============================] - 0s 22ms/step - loss: 0.0027 -
val_loss: 6.4241e-04
Epoch 44/100
12/12 [==============================] - 0s 19ms/step - loss: 0.0024 -
val_loss: 0.0012
Epoch 45/100
12/12 [==============================] - 0s 23ms/step - loss: 0.0024 -
val_loss: 8.5488e-04
Epoch 46/100
```

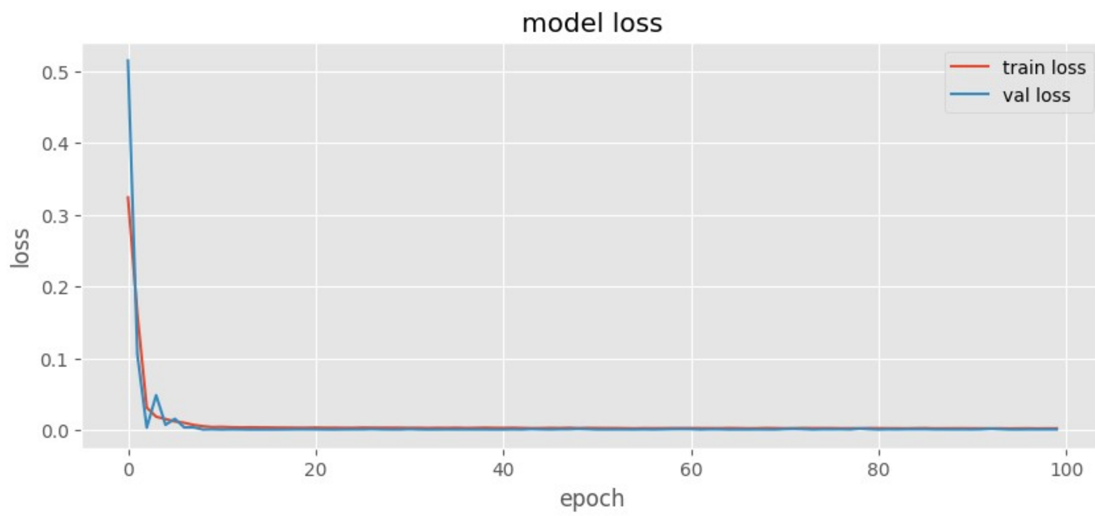
```
12/12 [==============================] - 0s 18ms/step - loss: 0.0027 -
val_loss: 6.5623e-04
Epoch 47/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0026 -
val_loss: 9.3385e-04
Epoch 48/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0030 -
val_loss: 0.0010
Epoch 49/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0025 -
val_loss: 0.0015
Epoch 50/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0027 -
val_loss: 0.0011
Epoch 51/100
12/12 [==============================] - 0s 17ms/step - loss: 0.0026 -
val_loss: 6.2857e-04
Epoch 52/100
12/12 [==============================] - 0s 18ms/step - loss: 0.0025 -
val_loss: 6.3086e-04
Epoch 53/100
12/12 [==============================] - 0s 19ms/step - loss: 0.0025 -
val_loss: 6.9913e-04
Epoch 54/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0023 -
val_loss: 6.9443e-04
Epoch 55/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0021 -
val_loss: 6.5038e-04
Epoch 56/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0024 -
val_loss: 8.1234e-04
Epoch 57/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0023 -
val_loss: 6.2960e-04
Epoch 58/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0023 -
val_loss: 8.4343e-04
Epoch 59/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0025 -
val_loss: 0.0011
Epoch 60/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0025 -
val_loss: 0.0012
Epoch 61/100
12/12 [==============================] - 0s 20ms/step - loss: 0.0025 -
val_loss: 0.0012
Epoch 62/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0024 -
val_loss: 6.3914e-04
```

```
Epoch 63/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0024 -
val_loss: 9.1721e-04
Epoch 64/100
12/12 [==============================] - 0s 19ms/step - loss: 0.0023 -
val_loss: 9.2544e-04
Epoch 65/100
12/12 [==============================] - 0s 20ms/step - loss: 0.0026 -
val_loss: 6.1067e-04
Epoch 66/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0024 -
val_loss: 6.4710e-04
Epoch 67/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0022 -
val_loss: 6.2876e-04
Epoch 68/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0022 -
val_loss: 7.4592e-04
Epoch 69/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0026 -
val_loss: 6.6616e-04
Epoch 70/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0024 -
val_loss: 6.1225e-04
Epoch 71/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0022 -
val_loss: 0.0012
Epoch 72/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0023 -
val_loss: 0.0015
Epoch 73/100
12/12 [==============================] - 0s 17ms/step - loss: 0.0026 -
val_loss: 0.0011
Epoch 74/100
12/12 [==============================] - 0s 18ms/step - loss: 0.0024 -
val_loss: 6.3207e-04
Epoch 75/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0024 -
val_loss: 9.4824e-04
Epoch 76/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0024 -
val_loss: 9.2989e-04
Epoch 77/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0022 -
val_loss: 0.0010
Epoch 78/100
12/12 [==============================] - 0s 17ms/step - loss: 0.0022 -
val_loss: 7.5611e-04
Epoch 79/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0023 -
```

```
val_loss: 0.0019
Epoch 80/100
12/12 [==============================] - 0s 17ms/step - loss: 0.0024 -
val_loss: 0.0011
Epoch 81/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0023 -
val_loss: 6.3124e-04
Epoch 82/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0022 -
val_loss: 8.7704e-04
Epoch 83/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0022 -
val_loss: 8.2500e-04
Epoch 84/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0021 -
val_loss: 0.0010
Epoch 85/100
12/12 [==============================] - 0s 17ms/step - loss: 0.0023 -
val_loss: 0.0010
Epoch 86/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0024 -
val_loss: 0.0011
Epoch 87/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0021 -
val_loss: 7.9519e-04
Epoch 88/100
12/12 [==============================] - 0s 16ms/step - loss: 0.0021 -
val_loss: 7.7954e-04
Epoch 89/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0022 -
val_loss: 7.2426e-04
Epoch 90/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0022 -
val_loss: 7.7557e-04
Epoch 91/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0021 -
val_loss: 7.0264e-04
Epoch 92/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0021 -
val_loss: 9.4707e-04
Epoch 93/100
12/12 [==============================] - 0s 14ms/step - loss: 0.0021 -
val_loss: 0.0017
Epoch 94/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0021 -
val_loss: 0.0011
Epoch 95/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0019 -
val_loss: 6.6568e-04
Epoch 96/100
```

```
12/12 [==============================] - 0s 12ms/step - loss: 0.0021 -
val_loss: 6.4345e-04
Epoch 97/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0021 -
val_loss: 7.8933e-04
Epoch 98/100
12/12 [==============================] - 0s 12ms/step - loss: 0.0019 -
val_loss: 7.2604e-04
Epoch 99/100
12/12 [==============================] - 0s 13ms/step - loss: 0.0021 -
val_loss: 7.7786e-04
Epoch 100/100
12/12 [==============================] - 0s 15ms/step - loss: 0.0021 -
val_loss: 7.4949e-04
```

```
fig = plt.figure(figsize=(10, 4))
plt.plot(pred.history['loss'], label='train loss')
plt.plot(pred.history['val_loss'], label='val loss')
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(loc='upper right')
plt.show()
```




```
learningrate_parameter = learning_rate
train_loss=pred.history['loss'][-1]
validation_loss=pred.history['val_loss'][-1]
learningrate_parameter=pd.DataFrame(data=[[learningrate_parameter,
train_loss, validation_loss]],
                                    columns=['Learning Rate',
'Training Loss', 'Validation Loss'])
learningrate_parameter.set_index('Learning Rate')
```

```
               Training Loss  Validation Loss
Learning Rate
0.0001              0.002097         0.000749
```

```
y_pred_test = model.predict(x_test)

# Invert normalization min-max
y_pred_invert_norm = scaler.inverse_transform(y_pred_test)
```

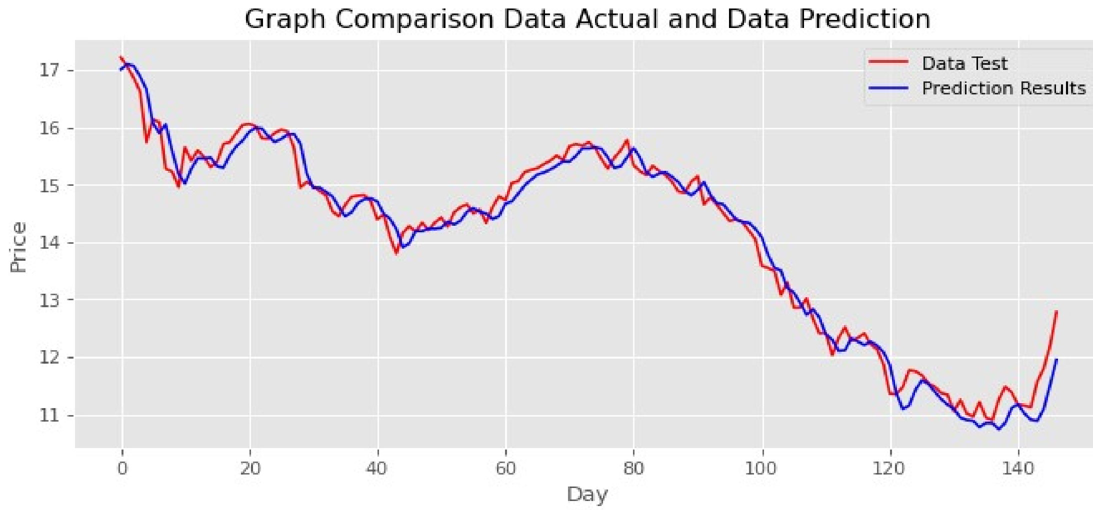
```
5/5 [==============================] - 1s 4ms/step
```

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

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

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

```
     Data Test  Prediction Results
0    17.219999           17.010592
1    17.059999           17.104795
2    16.860001           17.069622
3    16.620001           16.885792
4    15.740000           16.662764
..         ...                 ...
142  11.120000           10.905536
143  11.570000           10.884079
144  11.800000           11.092619
145  12.200000           11.515539
146  12.780000           11.951453

[147 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(datatest,datapred):
    mape = np.mean(np.abs((datatest - datapred)/datatest))*100
    return mape

mape = MAPE(datatest, datapred)
print(mape)
```

13.738295078642611

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

0.2761624981131313

```
from sklearn.metrics import mean_absolute_error

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

0.2076929376191639

```
n_ahead=input("How many values do you want to predict ?");
n_ahead=int(n_ahead)
# Making the prediction list
def predict_ahead(n_ahead, X_train):
   yhat = []
   for _ in range(n_ahead):
   # Making the prediction
       fc = model.predict(X_train)
       yhat.append(fc)
```

```python
    # Creating a new input matrix for forecasting
        X_train = np.append(X_train, fc)

    # Ommitting the first variable
        X_train = np.delete(X_train, 0)

    # Reshaping for the next iteration
        X_train = np.reshape(X_train, (1, len(X_train), 1))

    return yhat
y30 = predict_ahead(n_ahead, x_test[len(x_test)-30:])
```

```
1/1 [==============================] - 0s 86ms/step
1/1 [==============================] - 2s 2s/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 46ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 38ms/step
```

```python
y30
```

```
[array([[0.21674204],
        [0.21097781],
        [0.20105417],
        [0.18288219],
```

```
       [0.14114706],
       [0.11761088],
       [0.12266176],
       [0.14602378],
       [0.16010551],
       [0.15469024],
       [0.14323843],
       [0.13223785],
       [0.12397379],
       [0.11763855],
       [0.1049497 ],
       [0.1013751 ],
       [0.1002242 ],
       [0.0910664 ],
       [0.09719525],
       [0.09686133],
       [0.08795277],
       [0.09734835],
       [0.11907981],
       [0.12442023],
       [0.11102837],
       [0.10190491],
       [0.10009122],
       [0.11771923],
       [0.15346901],
       [0.19031721]], dtype=float32),
array([[7.3751546e+19]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
array([[nan]], dtype=float32),
```

```
 array([[nan]], dtype=float32),
 array([[nan]], dtype=float32),
 array([[nan]], dtype=float32),
 array([[nan]], dtype=float32),
 array([[nan]], dtype=float32)]
```

```
y30 = [[0.21674204],
        [0.21097781],
        [0.20105417],
        [0.18288219],
        [0.14114706],
        [0.11761088],
        [0.12266176],
        [0.14602378],
        [0.16010551],
        [0.15469024],
        [0.14323843],
        [0.13223785],
        [0.12397379],
        [0.11763855],
        [0.1049497 ],
        [0.1013751 ],
        [0.1002242 ],
        [0.0910664 ],
        [0.09719525],
        [0.09686133],
        [0.08795277],
        [0.09734835],
        [0.11907981],
        [0.12442023],
        [0.11102837],
        [0.10190491],
        [0.10009122],
        [0.11771923],
        [0.15346901],
        [0.19031721]]

y30 = scaler.inverse_transform(y30)

from numpy import savetxt
savetxt('Bi-LSTM.csv', y30, delimiter=',')
```