Nama : Saila Juila

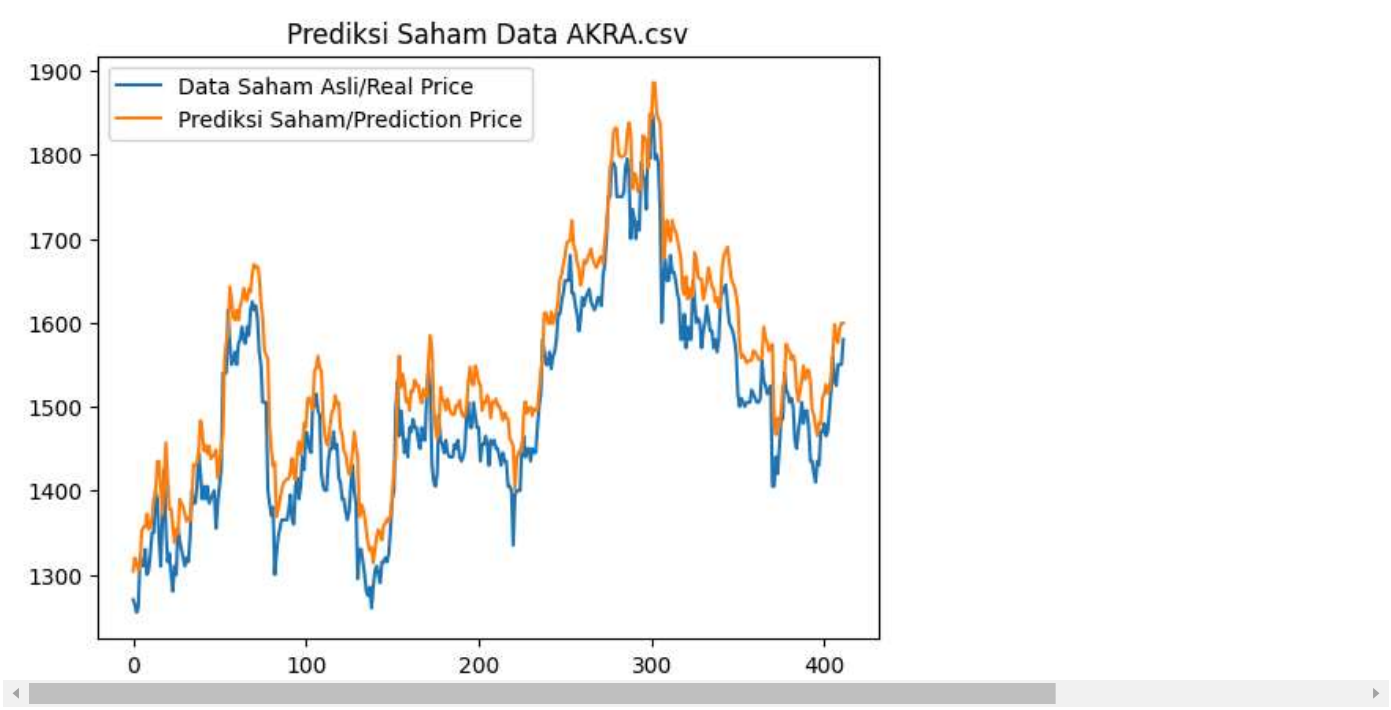
NIM : 20220040082

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_absolute_error
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.layers import LSTM, Dense

# Membaca data saham dari file CSV
#jika menggunakan data selain saham yahoo finance, bisa ubah disini
data = pd.read_csv('/content/AKRA.csv')
data = data['close'].values #cek pada masing2 data, karena terkadang atribut close = CLOSE/Close

# Normalisasi data
scaler = MinMaxScaler()
data = scaler.fit_transform(data.reshape(-1, 1))

# Fungsi untuk membagi dataset menjadi input dan output
def create_dataset(dataset, look_back=1):
    X, y = [], []
    for i in range(len(dataset) - look_back):
        X.append(dataset[i:(i+look_back)])
        y.append(dataset[i+look_back])
    return np.array(X), np.array(y)

# Membagi dataset menjadi data latih dan data uji
train_size = int(len(data) * 0.67)
train_data, test_data = data[0:train_size], data[train_size:]

look_back = 5
train_X, train_y = create_dataset(train_data, look_back)
test_X, test_y = create_dataset(test_data, look_back)

# Reshape data agar sesuai dengan input LSTM
train_X = np.reshape(train_X, (train_X.shape[0], 1, train_X.shape[1]))
test_X = np.reshape(test_X, (test_X.shape[0], 1, test_X.shape[1]))

# Membangun model LSTM
model = Sequential()
model.add(LSTM(4, input_shape=(1, look_back)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')

# Melatih model
model.fit(train_X, train_y, epochs=10, batch_size=1, verbose=2)

# Melakukan prediksi pada data uji
predicted = model.predict(test_X)
predicted = scaler.inverse_transform(predicted)
test_y = scaler.inverse_transform(test_y)

# Menghitung Mean Absolute Error (MAE)
mae = mean_absolute_error(test_y, predicted)
print(f'Mean Absolute Error (MAE): {mae}')
```

```
Epoch 1/10
/usr/local/lib/python3.10/dist-packages/keras/src/layers/rnn/rnn.py:204: UserWarning: Do not pass an `input_shape`
```

```
  super().__init__(**kwargs)
839/839 - 3s - 3ms/step - loss: 0.0358
Epoch 2/10
839/839 - 3s - 3ms/step - loss: 0.0030
Epoch 3/10
839/839 - 2s - 3ms/step - loss: 0.0026
Epoch 4/10
839/839 - 1s - 2ms/step - loss: 0.0024
Epoch 5/10
839/839 - 1s - 1ms/step - loss: 0.0023
Epoch 6/10
839/839 - 1s - 1ms/step - loss: 0.0022
Epoch 7/10
839/839 - 1s - 1ms/step - loss: 0.0021
Epoch 8/10
839/839 - 1s - 2ms/step - loss: 0.0020
Epoch 9/10
839/839 - 1s - 2ms/step - loss: 0.0019
Epoch 10/10
839/839 - 2s - 2ms/step - loss: 0.0018
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step
Mean Absolute Error (MAE): 73.02419036568948
```

```
# Plot hasil prediksi
plt.plot(test_y, label='Data Saham Asli/Real Price')
plt.plot(predicted, label='Prediksi Saham/Prediction Price')
plt.legend()
plt.title('Prediksi Saham Data AKRA.csv') #sesuaikan dengan dataset
plt.show()
```

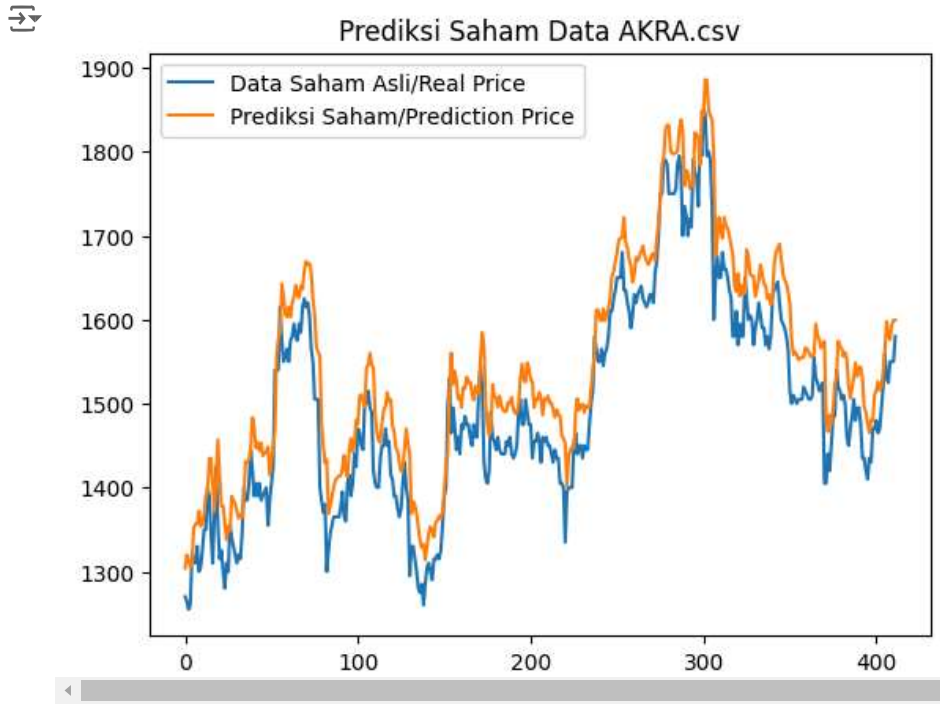



```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_absolute_error
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.layers import LSTM, Dense

# Membaca data saham dari file CSV
#jika menggunakan data selain saham yahoo finance, bisa ubah disini
data = pd.read_csv('/content/AKRA.csv')
```

```
data = data['close'].values #cek pada masing2 data, karena terkadang atribut close = CLOSE/Close

# Normalisasi data
scaler = MinMaxScaler()
data = scaler.fit_transform(data.reshape(-1, 1))

# Fungsi untuk membagi dataset menjadi input dan output
def create_dataset(dataset, look_back=1):
    X, y = [], []
    for i in range(len(dataset) - look_back):
        X.append(dataset[i:(i+look_back)])
        y.append(dataset[i+look_back])
    return np.array(X), np.array(y)

# Membagi dataset menjadi data latih dan data uji
train_size = int(len(data) * 0.67)
train_data, test_data = data[0:train_size], data[train_size:]

look_back = 5
train_X, train_y = create_dataset(train_data, look_back)
test_X, test_y = create_dataset(test_data, look_back)

# Reshape data agar sesuai dengan input LSTM
train_X = np.reshape(train_X, (train_X.shape[0], 1, train_X.shape[1]))
test_X = np.reshape(test_X, (test_X.shape[0], 1, test_X.shape[1]))

# Membangun model LSTM
model = Sequential()
model.add(LSTM(4, input_shape=(1, look_back)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')

# Melatih model
model.fit(train_X, train_y, epochs=50, batch_size=1, verbose=2)

# Melakukan prediksi pada data uji
predicted = model.predict(test_X)
predicted = scaler.inverse_transform(predicted)
test_y = scaler.inverse_transform(test_y)

# Menghitung Mean Absolute Error (MAE)
mae = mean_absolute_error(test_y, predicted)
print(f'Mean Absolute Error (MAE): {mae}')
```

```
Epoch 34/50
839/839 - 1s - 1ms/step - loss: 0.0013
Epoch 35/50
839/839 - 1s - 1ms/step - loss: 0.0013
Epoch 36/50
839/839 - 1s - 1ms/step - loss: 0.0013
Epoch 37/50
839/839 - 1s - 1ms/step - loss: 0.0013
Epoch 38/50
839/839 - 2s - 2ms/step - loss: 0.0013
Epoch 39/50
839/839 - 2s - 2ms/step - loss: 0.0013
Epoch 40/50
839/839 - 1s - 2ms/step - loss: 0.0012
Epoch 41/50
839/839 - 1s - 1ms/step - loss: 0.0014
Epoch 42/50
839/839 - 1s - 2ms/step - loss: 0.0013
Epoch 43/50
839/839 - 1s - 1ms/step - loss: 0.0013
Epoch 44/50
839/839 - 1s - 1ms/step - loss: 0.0014
Epoch 45/50
839/839 - 1s - 1ms/step - loss: 0.0013
Epoch 46/50
839/839 - 1s - 1ms/step - loss: 0.0013
Epoch 47/50
839/839 - 2s - 2ms/step - loss: 0.0013
Epoch 48/50
839/839 - 2s - 2ms/step - loss: 0.0013
Epoch 49/50
839/839 - 2s - 2ms/step - loss: 0.0013
Epoch 50/50
839/839 - 1s - 1ms/step - loss: 0.0013
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step
Mean Absolute Error (MAE): 22.364973049719357
```

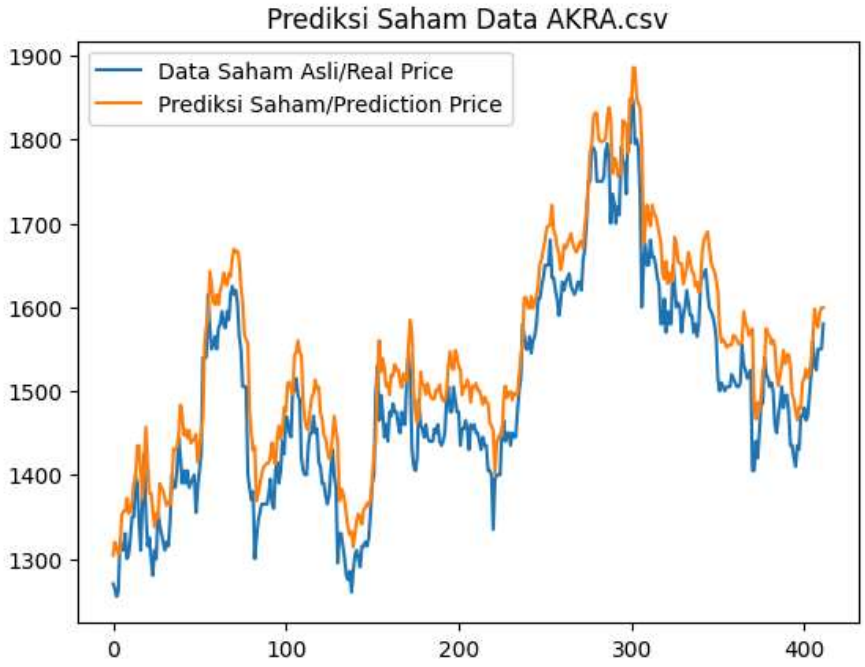
```
# Plot hasil prediksi
plt.plot(test_y, label='Data Saham Asli/Real Price')
plt.plot(predicted, label='Prediksi Saham/Prediction Price')
plt.legend()
plt.title('Prediksi Saham Data AKRA.csv') #sesuaikan dengan dataset
plt.show()
```



```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_absolute_error
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.layers import LSTM, Dense

# Membaca data saham dari file CSV
#jika menggunakan data selain saham yahoo finance, bisa ubah disini
data = pd.read_csv('/content/AKRA.csv')
data = data['close'].values #cek pada masing2 data, karena terkadang atribut close = CLOSE/Close

# Normalisasi data
scaler = MinMaxScaler()
data = scaler.fit_transform(data.reshape(-1, 1))

# Fungsi untuk membagi dataset menjadi input dan output
def create_dataset(dataset, look_back=1):
    X, y = [], []
    for i in range(len(dataset) - look_back):
        X.append(dataset[i:(i+look_back)])
        y.append(dataset[i+look_back])
    return np.array(X), np.array(y)

# Membagi dataset menjadi data latih dan data uji
train_size = int(len(data) * 0.67)
train_data, test_data = data[0:train_size], data[train_size:]

look_back = 5
train_X, train_y = create_dataset(train_data, look_back)
test_X, test_y = create_dataset(test_data, look_back)

# Reshape data agar sesuai dengan input LSTM
train_X = np.reshape(train_X, (train_X.shape[0], 1, train_X.shape[1]))
test_X = np.reshape(test_X, (test_X.shape[0], 1, test_X.shape[1]))

# Membangun model LSTM
model = Sequential()
model.add(LSTM(4, input_shape=(1, look_back)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')

# Melatih model
model.fit(train_X, train_y, epochs=100, batch_size=1, verbose=2)

# Melakukan prediksi pada data uji
predicted = model.predict(test_X)
predicted = scaler.inverse_transform(predicted)
test_y = scaler.inverse_transform(test_y)

# Menghitung Mean Absolute Error (MAE)
mae = mean_absolute_error(test_y, predicted)
print(f'Mean Absolute Error (MAE): {mae}')
```

```
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
s/step - loss: 0.0013
━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step
or (MAE): 50.03733396067203
```

```
# Plot hasil prediksi
plt.plot(test_y, label='Data Saham Asli/Real Price')
plt.plot(predicted, label='Prediksi Saham/Prediction Price')
plt.legend()
plt.title('Prediksi Saham Data AKRA.csv') #sesuaikan dengan dataset
plt.show()
```


Prediksi Saham Data AKRA.csv
Data Saham Asli/Real Price
Prediksi Saham/Prediction Price
1900
1800
1700
1600
1500
1400
1300
0
100
200
300
400