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

```
# Load the data
df = pd.read_csv('/content/apple_share_price.csv')
```

```
# Preprocess the data
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(df['Close'].values.reshape(-1, 1))
# Split the data into training and testing sets
train_size = int(len(scaled_data) * 0.8)
train_data = scaled_data[:train_size]
test_data = scaled_data[train_size:]
# Build the LSTM model
model = Sequential()
model.add(LSTM(units=100, return_sequences=True, input_shape=(train_data.shape[1], 1)))
model.add(LSTM(units=100))
model.add(Dense(units=1))
# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')
```

```
# Train the model
model.fit(train_data, train_data, epochs=100, batch_size=32)
```

```
Epoch 1/100
42/42 [==============================] - 5s 7ms/step - loss: 0.1196
Epoch 2/100
42/42 [==============================] - 0s 6ms/step - loss: 0.0142
Epoch 3/100
42/42 [==============================] - 0s 6ms/step - loss: 0.0060
Epoch 4/100
42/42 [==============================] - 0s 6ms/step - loss: 0.0014
Epoch 5/100
42/42 [==============================] - 0s 7ms/step - loss: 2.3532e-04
Epoch 6/100
42/42 [==============================] - 0s 7ms/step - loss: 1.9342e-04
Epoch 7/100
42/42 [==============================] - 0s 7ms/step - loss: 1.8461e-04
Epoch 8/100
42/42 [==============================] - 0s 6ms/step - loss: 1.6503e-04
Epoch 9/100
42/42 [==============================] - 0s 7ms/step - loss: 1.5227e-04
Epoch 10/100
42/42 [==============================] - 0s 7ms/step - loss: 1.3810e-04
Epoch 11/100
42/42 [==============================] - 0s 6ms/step - loss: 1.2899e-04
Epoch 12/100
42/42 [==============================] - 0s 6ms/step - loss: 1.1201e-04
Epoch 13/100
42/42 [==============================] - 0s 7ms/step - loss: 1.0194e-04
Epoch 14/100
42/42 [==============================] - 0s 6ms/step - loss: 8.6564e-05
Epoch 15/100
42/42 [==============================] - 0s 6ms/step - loss: 7.4738e-05
Epoch 16/100
42/42 [==============================] - 0s 7ms/step - loss: 6.6129e-05
Epoch 17/100
42/42 [==============================] - 0s 9ms/step - loss: 5.6781e-05
Epoch 18/100
42/42 [==============================] - 0s 8ms/step - loss: 4.5316e-05
Epoch 19/100
42/42 [==============================] - 0s 7ms/step - loss: 3.6202e-05
Epoch 20/100
42/42 [==============================] - 0s 6ms/step - loss: 2.9028e-05
Epoch 21/100
42/42 [==============================] - 0s 6ms/step - loss: 2.3230e-05
Epoch 22/100
42/42 [==============================] - 0s 9ms/step - loss: 1.7867e-05
Epoch 23/100
42/42 [==============================] - 0s 9ms/step - loss: 1.3394e-05
Epoch 24/100
42/42 [==============================] - 0s 9ms/step - loss: 1.0150e-05
Epoch 25/100
42/42 [==============================] - 0s 9ms/step - loss: 7.4527e-06
Epoch 26/100
42/42 [==============================] - 0s 9ms/step - loss: 5.2898e-06
Epoch 27/100
42/42 [==============================] - 0s 9ms/step - loss: 3.6703e-06
```

```
     Epoch 28/100
     42/42 [==============================] - 0s 9ms/step - loss: 2.5751e-06
     Epoch 29/100
```

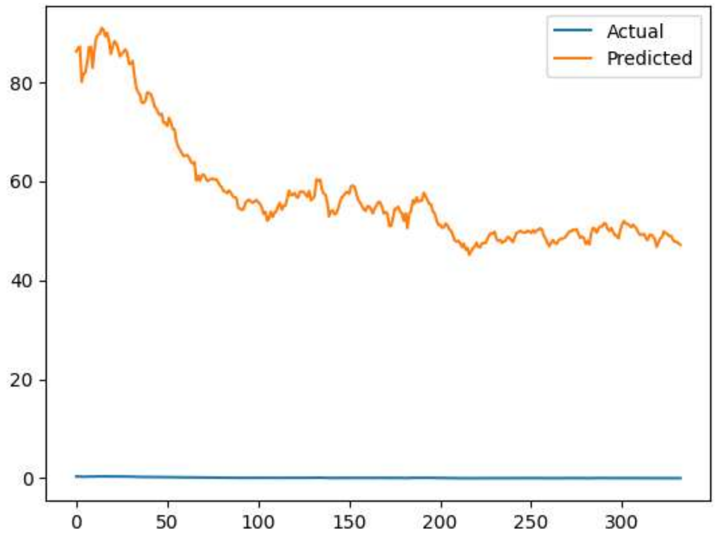
```
# Evaluate the model
predictions = model.predict(test_data)
predictions = scaler.inverse_transform(predictions)
```

```
     11/11 [==============================] - 1s 4ms/step
```

```
# Calculate the mean absolute error
mae = mean_absolute_error(test_data, predictions)
print('MAE:', mae)
```

```
     MAE: 57.61431422574825
```

```
# Plot the results
plt.plot(test_data, label='Actual')
plt.plot(predictions, label='Predicted')
plt.legend()
plt.show()
```



Colab paid products - Cancel contracts here

✓ 0s completed at 11:14 PM