Lab Logbook Requirement:

1. *Modify the practical session LSTM model parameter from 100 to be calculated using the formula:*

   ZY + 10 , where your SID is: XXXXXZY

2. *Change the epochs to 10.*
3. *Change the patience to 3*
4. *Leave other parameters the same as in the practical session.*
5. *Compile the model.*
6. *Train your LSTM with the same datasets and demonstrate the received test MSE & MAE. Compare your test MSE & MAE with the MSE & MAE of the LSTM in the practical session.*
7. *Please only add to your Lab Logbook print-screens of:*

- *your LSTM architecture using model.summary(),*
- *the resulting test MSE & MAE and*
- *MAE detailed graph*

```
#sid =2368529 where Z=2 and Y=9
#ZY +10 = 29 + 10 = 39

model = keras.Sequential([

    keras.layers.LSTM(39, activation = 'relu', input_shape = (50, 18)),

    keras.layers.Dense(2)

])

print(model.summary())
```

Model: "sequential_3"

Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 39)	9,048
dense_3 (Dense)	(None, 2)	80

Total params: 9,128 (35.66 KB)
Trainable params: 9,128 (35.66 KB)
Non-trainable params: 0 (0.00 B)
None

[51]:
```
model.compile(optimizer = "adam", loss = "mse", metrics =["mae"])
```

[52]:
```
es = EarlyStopping(monitor='val_loss', mode='min', patience=3, verbose=1)
mc = ModelCheckpoint('best_model_LSTM_GOLD.keras', monitor='val_loss', mode='min', verbose=1, save_best_only=True)
```

[53]:
```
history = model.fit(X_train, y_train, batch_size = 20, epochs = 10, validation_split = 0.1, shuffle = True, verbose =1, callbacks = [es,mc])
```

```
Epoch 1/10
1212/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - loss: 0.1920 - mae: 0.1231
Epoch 1: val_loss improved from inf to 0.00005, saving model to best_model_LSTM_GOLD.keras
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 25s 18ms/step - loss: 0.1918 - mae: 0.1230 - val_loss: 4.9154e-05 - val_mae: 0.0049
Epoch 2/10
1212/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step - loss: 4.8671e-05 - mae: 0.0052
Epoch 2: val_loss improved from 0.00005 to 0.00003, saving model to best_model_LSTM_GOLD.keras
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 21s 17ms/step - loss: 4.8660e-05 - mae: 0.0052 - val_loss: 2.9932e-05 - val_mae: 0.0039
Epoch 3/10
1211/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step - loss: 3.3265e-05 - mae: 0.0045
Epoch 3: val_loss did not improve from 0.00003
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 21s 17ms/step - loss: 3.3264e-05 - mae: 0.0045 - val_loss: 6.5819e-05 - val_mae: 0.0068
Epoch 4/10
1212/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - loss: 3.3284e-05 - mae: 0.0045
Epoch 4: val_loss did not improve from 0.00003
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 44s 19ms/step - loss: 3.3282e-05 - mae: 0.0045 - val_loss: 6.8866e-05 - val_mae: 0.0069
Epoch 5/10
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - loss: 3.2081e-05 - mae: 0.0045
Epoch 5: val_loss improved from 0.00003 to 0.00002, saving model to best_model_LSTM_GOLD.keras
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 22s 18ms/step - loss: 3.2080e-05 - mae: 0.0045 - val_loss: 2.0336e-05 - val_mae: 0.0033
Epoch 6/10
1212/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - loss: 3.7082e-05 - mae: 0.0048
Epoch 6: val_loss did not improve from 0.00002
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 21s 17ms/step - loss: 3.7076e-05 - mae: 0.0048 - val_loss: 3.4457e-05 - val_mae: 0.0047
Epoch 7/10
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step - loss: 3.2006e-05 - mae: 0.0045
Epoch 7: val_loss did not improve from 0.00002
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 19s 15ms/step - loss: 3.2006e-05 - mae: 0.0045 - val_loss: 2.4473e-05 - val_mae: 0.0044
Epoch 8/10
1211/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step - loss: 2.4711e-05 - mae: 0.0039
Epoch 8: val_loss improved from 0.00002 to 0.00001, saving model to best_model_LSTM_GOLD.keras
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 21s 17ms/step - loss: 2.4712e-05 - mae: 0.0039 - val_loss: 1.3461e-05 - val_mae: 0.0030
Epoch 9/10
1211/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - loss: 2.2413e-05 - mae: 0.0037
Epoch 9: val_loss did not improve from 0.00001
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 45s 21ms/step - loss: 2.2414e-05 - mae: 0.0037 - val_loss: 4.0357e-05 - val_mae: 0.0050
Epoch 10/10
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - loss: 2.0940e-05 - mae: 0.0036
Epoch 10: val_loss improved from 0.00001 to 0.00001, saving model to best_model_LSTM_GOLD.keras
1213/1213 ━━━━━━━━━━━━━━━━━━━━ 24s 20ms/step - loss: 2.0939e-05 - mae: 0.0036 - val_loss: 9.3247e-06 - val_mae: 0.0026
```

```
scores = LSTM_saved_best_model.evaluate(X_test, y_test, verbose=1)
```

```
94/94 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - loss: 9.4234e-06 - mae: 0.0026
```

```
scores
```

```
[8.694913049112074e-06, 0.0024753068573772907]
```

```
print("Mean squared error (mse): %.9f " % (scores[0]))
```

```
Mean squared error (mse): 0.000008695
```

```
print("Mean absolute error (mae): %.9f " % (scores[1]))
```

```
Mean absolute error (mae): 0.002475307
```

```
history_dict = history.history

mae_values = history_dict['mae']
val_mae_values = history_dict['val_mae']

epochs = range(1, len(mae_values) + 1)
plt.figure(num=1, figsize=(15,7))
plt.plot(epochs, mae_values, 'b', label='Training Mean Absolute Error(MAE)')
plt.plot(epochs, val_mae_values, marker='o', markeredgecolor='red', markerfacecolor='yellow', label='Validation Mean Absolute Error(MAE)')
plt.xlabel('Epochs', size=18)
plt.ylabel('Mean Absolute Error(MAE)', size=18)
plt.legend()
plt.show()
```