In [1]:

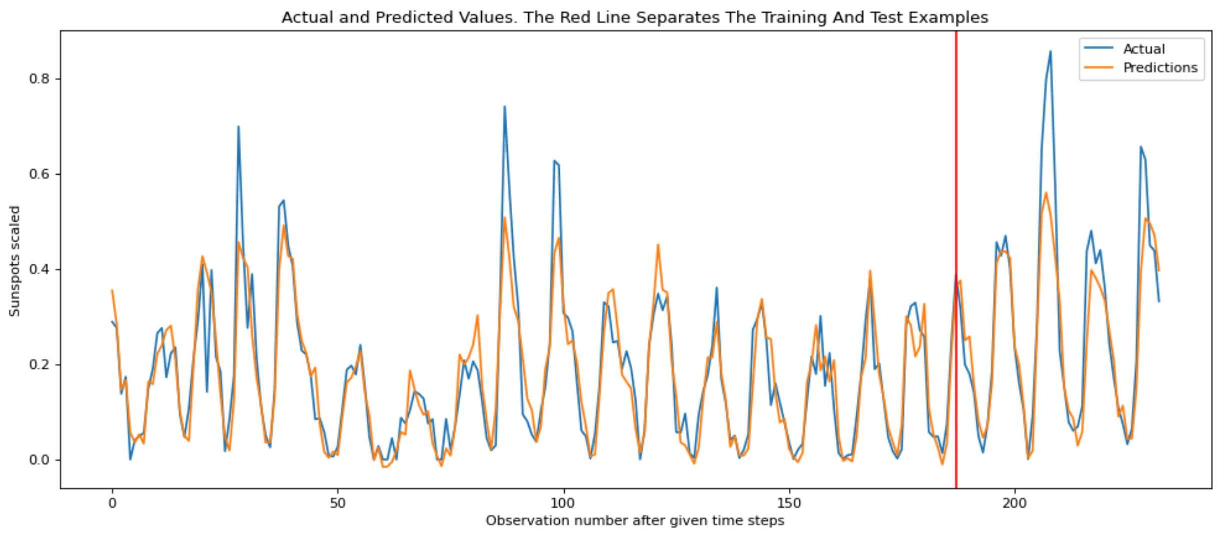
```python
from pandas import read_csv
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, SimpleRNN
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
import math
import matplotlib.pyplot as plt
```

In [4]:

```python
# Parameter split_percent defines the ratio of training examples
def get_train_test(url, split_percent=0.8):
    df = read_csv(url, usecols=[1], engine='python')
    data = np.array(df.values.astype('float32'))
    scaler = MinMaxScaler(feature_range=(0, 1))
    data = scaler.fit_transform(data).flatten()
    n = len(data)
    # Point for splitting data into train and test
    split = int(n*split_percent)
    train_data = data[range(split)]
    test_data = data[split:]
    return train_data, test_data, data

# Prepare the input X and target Y
def get_XY(dat, time_steps):
    Y_ind = np.arange(time_steps, len(dat), time_steps)
    Y = dat[Y_ind]
    rows_x = len(Y)
    X = dat[range(time_steps*rows_x)]
    X = np.reshape(X, (rows_x, time_steps, 1))
    return X, Y

def create_RNN(hidden_units, dense_units, input_shape, activation):
    model = Sequential()
    model.add(SimpleRNN(hidden_units, input_shape=input_shape, activation=activation[0]))
    model.add(Dense(units=dense_units, activation=activation[1]))
    model.compile(loss='mean_squared_error', optimizer='adam')
    return model

def print_error(trainY, testY, train_predict, test_predict):
    # Error of predictions
    train_rmse = math.sqrt(mean_squared_error(trainY, train_predict))
    test_rmse = math.sqrt(mean_squared_error(testY, test_predict))
    # Print RMSE
    print('Train RMSE: %.3f RMSE' % (train_rmse))
    print('Test RMSE: %.3f RMSE' % (test_rmse))

# Plot the result
def plot_result(trainY, testY, train_predict, test_predict):
    actual = np.append(trainY, testY)
    predictions = np.append(train_predict, test_predict)
    rows = len(actual)
    plt.figure(figsize=(15, 6), dpi=80)
    plt.plot(range(rows), actual)
    plt.plot(range(rows), predictions)
    plt.axvline(x=len(trainY), color='r')
    plt.legend(['Actual', 'Predictions'])
    plt.xlabel('Observation number after given time steps')
    plt.ylabel('Sunspots scaled')
    plt.title('Actual and Predicted Values. The Red Line Separates The Training And Test Examples')

sunspots_url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/monthly-sunspots.csv'
time_steps = 12
train_data, test_data, data = get_train_test(sunspots_url)
trainX, trainY = get_XY(train_data, time_steps)
testX, testY = get_XY(test_data, time_steps)

# Create model and train
model = create_RNN(hidden_units=3, dense_units=1, input_shape=(time_steps,1),
                   activation=['tanh', 'tanh'])
model.fit(trainX, trainY, epochs=20, batch_size=1, verbose=2)

# make predictions
```

```
train_predict = model.predict(trainX)
test_predict = model.predict(testX)

# Print error
print_error(trainY, testY, train_predict, test_predict)

#Plot result
plot_result(trainY, testY, train_predict, test_predict)
```

```
Epoch 1/20
187/187 - 1s - loss: 0.0152 - 1s/epoch - 6ms/step
Epoch 2/20
187/187 - 0s - loss: 0.0113 - 238ms/epoch - 1ms/step
Epoch 3/20
187/187 - 0s - loss: 0.0091 - 245ms/epoch - 1ms/step
Epoch 4/20
187/187 - 0s - loss: 0.0078 - 322ms/epoch - 2ms/step
Epoch 5/20
187/187 - 0s - loss: 0.0068 - 249ms/epoch - 1ms/step
Epoch 6/20
187/187 - 0s - loss: 0.0062 - 234ms/epoch - 1ms/step
Epoch 7/20
187/187 - 0s - loss: 0.0057 - 238ms/epoch - 1ms/step
Epoch 8/20
187/187 - 0s - loss: 0.0053 - 247ms/epoch - 1ms/step
Epoch 9/20
187/187 - 0s - loss: 0.0050 - 256ms/epoch - 1ms/step
Epoch 10/20
187/187 - 0s - loss: 0.0048 - 239ms/epoch - 1ms/step
Epoch 11/20
187/187 - 0s - loss: 0.0046 - 261ms/epoch - 1ms/step
Epoch 12/20
187/187 - 0s - loss: 0.0045 - 249ms/epoch - 1ms/step
Epoch 13/20
187/187 - 0s - loss: 0.0043 - 248ms/epoch - 1ms/step
Epoch 14/20
187/187 - 0s - loss: 0.0042 - 238ms/epoch - 1ms/step
Epoch 15/20
187/187 - 0s - loss: 0.0041 - 258ms/epoch - 1ms/step
Epoch 16/20
187/187 - 0s - loss: 0.0040 - 267ms/epoch - 1ms/step
Epoch 17/20
187/187 - 0s - loss: 0.0040 - 239ms/epoch - 1ms/step
Epoch 18/20
187/187 - 0s - loss: 0.0038 - 246ms/epoch - 1ms/step
Epoch 19/20
187/187 - 0s - loss: 0.0039 - 241ms/epoch - 1ms/step
Epoch 20/20
187/187 - 0s - loss: 0.0038 - 242ms/epoch - 1ms/step
6/6 [==============================] - 0s 2ms/step
2/2 [==============================] - 0s 3ms/step
Train RMSE: 0.060 RMSE
Test RMSE: 0.094 RMSE
```


Actual and Predicted Values. The Red Line Separates The Training And Test Examples
Actual
Predictions
Sunspots scaled
Observation number after given time steps