```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, Dropout

# Simulate a dataset
def simulate_data(num_samples, num_features):
    np.random.seed(42)
    features = np.random.rand(num_samples, num_features)
    target = np.sin(np.linspace(0, 20, num_samples)) + 0.1 * np.random.randn(num_samples)
    return features, target

# Preprocess the simulated dataset
def preprocess_data(features, target):
    scaler = MinMaxScaler()
    features_scaled = scaler.fit_transform(features)
    return features_scaled, target, scaler

def create_sequences(features, target, time_steps):
    X, y = [], []
    for i in range(len(features) - time_steps):
        X.append(features[i:i+time_steps])
        y.append(target[i+time_steps])
    return np.array(X), np.array(y)

# Define the LSTM model
def build_model(input_shape):
    model = Sequential([
        LSTM(50, return_sequences=True, input_shape=input_shape),
        Dropout(0.2),
        LSTM(50, return_sequences=False),
        Dropout(0.2),
        Dense(25, activation='relu'),
        Dense(1)
    ])
    model.compile(optimizer='adam', loss='mse')
    return model

# Train and evaluate the model
def train_model(model, X_train, y_train, X_val, y_val, epochs, batch_size):
    history = model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=epochs, batch_size=batch_size, verbose=1)
    return history

def evaluate_model(model, X_test, y_test):
    loss = model.evaluate(X_test, y_test, verbose=0)
    print(f"Test Loss: {loss}")
    return loss

# Predict and visualize results
def predict_and_plot(model, X_test, y_test, scaler):
    predictions = model.predict(X_test)
    predictions_rescaled = scaler.inverse_transform(predictions)
    y_test_rescaled = scaler.inverse_transform(y_test.reshape(-1, 1))

    plt.figure(figsize=(10, 6))
    plt.plot(y_test_rescaled, label="True Values")
    plt.plot(predictions_rescaled, label="Predicted Values")
    plt.legend()
    plt.title("True vs Predicted Values")
    plt.xlabel("Time")
    plt.ylabel("Weather Parameter")
    plt.show()

# Main script
def main():
    # Configuration
    num_samples = 1000
    num_features = 4
    time_steps = 30
    epochs = 20
    batch_size = 32
```

```
        # Simulate and preprocess data
        features, target = simulate_data(num_samples, num_features)
        features, target, scaler = preprocess_data(features, target)
        X, y = create_sequences(features, target, time_steps)

        # Split data into training, validation, and test sets
        X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3, random_state=42)
        X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, random_state=42)

        # Build and train the model
        model = build_model(input_shape=(X_train.shape[1], X_train.shape[2]))
        train_model(model, X_train, y_train, X_val, y_val, epochs, batch_size)

        # Evaluate the model
        evaluate_model(model, X_test, y_test)

        # Predict and plot results
        predict_and_plot(model, X_test, y_test, scaler)

    if __name__ == "__main__":
        main()
```

```
Epoch 1/20
/usr/local/lib/python3.10/dist-packages/keras/src/layers/rnn/rnn.py:204: UserWarning: Do not pass an `input_shape`/`input_dim` argume
  super().__init__(**kwargs)
22/22 ━━━━━━━━━━━━━━━━━━━━ 4s 48ms/step - loss: 0.5411 - val_loss: 0.4726
Epoch 2/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - loss: 0.5151 - val_loss: 0.4744
Epoch 3/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 36ms/step - loss: 0.5385 - val_loss: 0.4784
Epoch 4/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 34ms/step - loss: 0.5329 - val_loss: 0.4696
Epoch 5/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - loss: 0.5312 - val_loss: 0.4746
Epoch 6/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - loss: 0.5244 - val_loss: 0.4793
Epoch 7/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - loss: 0.5111 - val_loss: 0.4828
Epoch 8/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 57ms/step - loss: 0.4994 - val_loss: 0.4489
Epoch 9/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 40ms/step - loss: 0.4758 - val_loss: 0.4686
Epoch 10/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 34ms/step - loss: 0.4899 - val_loss: 0.4289
Epoch 11/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - loss: 0.4886 - val_loss: 0.4614
Epoch 12/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - loss: 0.5256 - val_loss: 0.4501
Epoch 13/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - loss: 0.5040 - val_loss: 0.4057
Epoch 14/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - loss: 0.4939 - val_loss: 0.4355
```

Next steps: Explain error