**Codes:**

import numpy as np

from sklearn.gaussian\_process import GaussianProcessRegressor

from sklearn.gaussian\_process.kernels import RBF

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error

# 生成模拟数据集

np.random.seed(0)

n\_samples = 100

X = np.random.rand(n\_samples, 1) # 一维输入特征

y = np.sin(10 \* X) + np.cos(5 \* X) + np.random.normal(scale=0.1, size=n\_samples) # 添加噪声的目标变量

# 划分训练集和测试集

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# 定义高斯过程回归模型和核函数

kernel = RBF(length\_scale=1.0, length\_scale\_bounds=(1e-1, 10.0)) # 使用RBF核函数，并设置长度尺度的搜索范围

gpr = GaussianProcessRegressor(kernel=kernel, alpha=1e-½, optimizer='fmin\_l\_bfgs\_b', n\_restarts\_optimizer=10)

# 训练模型

gpr.fit(X\_train, y\_train)

# 预测测试集结果

y\_pred, y\_pred\_std = gpr.predict(X\_test, return\_std=True) # 获取预测均值和标准差（方差的平方根）

# 计算并打印预测误差

mse = mean\_squared\_error(y\_test, y\_pred)

print(f'Mean Squared Error (MSE): {mse:.2f}')

# 可视化结果

import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))

plt.plot(X\_train, y\_train, 'o', color='gray', label='Training Data')

plt.plot(X\_test, y\_test, '^', color='black', label='True Values')

plt.plot(X\_test, y\_pred, '-', color='red', label='Predictions')

plt.fill\_between(X\_test.ravel(),

y\_pred - y\_pred\_std,

y\_pred + y\_pred\_std,

alpha=0.2, color='pink',

label='Standard Deviation')

plt.title('Gaussian Process Regression with RBF Kernel')

plt.xlabel('Input Feature')

plt.ylabel('Target Variable')

plt.legend()

plt.show()