Gaussian Process

Loïc Tetrel

Keywords: Generative model; Filtering; Regression

1 Introduction

Daniel G. Kringe was the first to define a Gaussian Process in a new interpolation method to evaluate mining ressources in geostatistics. In this time, we were talking about Kriging [1]. Later, Rasmussen[2] propose to develop the Gaussian Process ($\mathcal{GP}$) theory to do machine learning. Since then, it is a famous generative model for AI applications.

2 Theory

The function f is a $\mathcal{GP}$ if a subset of the outputs $\{y(x_0), ..., y(x_N)\}$ has a multivariate gaussian distribution :

$$f(x_0, ..., x_N) = \mathcal{N}(\mu, \Sigma), \tag{1}$$

This process is entirely define by its mean function μ ans its covariance Σ.

$$\mu = \begin{bmatrix} \mu_0 \\ \vdots \\ \mu_N \end{bmatrix}; \qquad \Sigma = \begin{bmatrix} \Sigma(x_0, x_0) & \dots & \Sigma(x_0, x_N) \\ \vdots & \ddots & \vdots \\ \Sigma(x_N, x_0) & \dots & \Sigma(x_N, x_N) \end{bmatrix}. \tag{2}$$

Usually, a gaussian noise is employed to model the uncertainty behind the model, so the output becomes :

$$y = f(x) + \mathcal{N}(0, \sigma_y^2). \tag{3}$$

The covariance allows us to model the non-linearity of the inputs (known as the kernel trick). A widely used kernel is the gaussian one, which will ensure that close inputs produce output values that are close together :

$$\Sigma(x, x') = \sigma_x \exp\left(\frac{\|x - x'\|^2}{2l^2}\right). \tag{4}$$

Where σ_x is the noise from the inputs, and l is the lengthscale of the covariance. This length has to be choose carefully to improve accuracy of the prediction.

An important thing to understand is to distinguish the prior and the posterior of the $\mathcal{GP}$. The prior is observable whenever we have training values, the posterior will change in consequence with new data coming into the process.

3 Code example

Below you can see the effect on the $\mathcal{GP}$ when some inputs comes. You may notice that the outputs from the prior (vertical axis) are coming from a gaussian distribution :

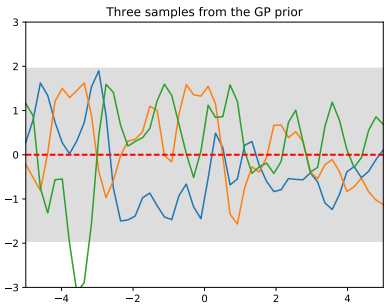

Fig. 1: Three random function taken from the prior. The mean function (red) is zero, and the covariance function constant.

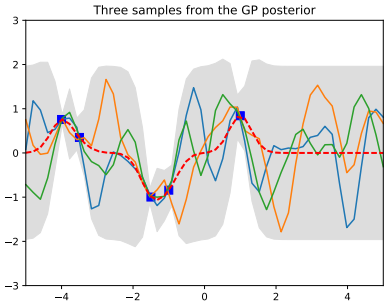

Fig. 2: With 5 test data (blue squares), three different samples are taken from the posterior. We can now evaluate the mean function (red) and the covariance at 95% (grey).

To generate the figure, you can use this code (slightly modified from `http://katbailey.github.io/post/gaussian-processes-for-dummies/`):

```
import matplotlib.pyplot as plt
```

```
import numpy as np

# Test data
n = 50
Xtest = np.linspace(-5, 5, n).reshape(-1,1)

# Define the kernel function
def kernel(a, b, param):
    sqdist = np.sum(a**2,1).reshape(-1,1) + np.sum(b**2,1) - 2*np.dot(a, b.T)
    return np.exp(-.5 * (1/param) * sqdist)

param = 0.1
K_ss = kernel(Xtest, Xtest, param)

# Get cholesky decomposition (square root) of the
# covariance matrix
L = np.linalg.cholesky(K_ss + 1e-15*np.eye(n))
# Sample 3 sets of standard normals for our test points,
# multiply them by the square root of the covariance matrix
f_prior = np.dot(L, np.random.normal(size=(n,3)))

# Now let's plot the 3 sampled functions.
plt.plot(Xtest, f_prior)
plt.gca().fill_between(Xtest.flat, -1.96, 1.96, color="#dddddd")
plt.plot(Xtest, np.zeros((n, 1)), 'r--', lw=2)
plt.axis([-5, 5, -3, 3])
plt.title('Three samples from the GP prior')
plt.show()

# Noiseless training data
Xtrain = np.array([-4, -3.5, -1.5, -1, 1]).reshape(5,1)
ytrain = np.sin(Xtrain)

# Apply the kernel function to our training points
K = kernel(Xtrain, Xtrain, param)
L = np.linalg.cholesky(K + 0.00005*np.eye(len(Xtrain)))

# Compute the mean at our test points.
K_s = kernel(Xtrain, Xtest, param)
Lk = np.linalg.solve(L, K_s)
mu = np.dot(Lk.T, np.linalg.solve(L, ytrain)).reshape((n,))

# Compute the standard deviation so we can plot it
s2 = np.diag(K_ss) - np.sum(Lk**2, axis=0)
stdv = np.sqrt(s2)
```

```
# Draw samples from the posterior at our test points.
L = np.linalg.cholesky(K_ss + 1e-6*np.eye(n) - np.dot(Lk.T, Lk))
f_post = mu.reshape(-1,1) + np.dot(L, np.random.normal(size=(n,3)))

plt.figure()
plt.plot(Xtrain, ytrain, 'bs', ms=8)
plt.plot(Xtest, f_post)
plt.gca().fill_between(Xtest.flat, mu-1.96*stdv, mu+1.96*stdv, color="#dddddd
plt.plot(Xtest, mu, 'r--', lw=2)
plt.axis([-5, 5, -3, 3])
plt.title('Three samples from the GP posterior')
plt.show()
```

4 Study case :

5 Conclusion

Acknowledgment

Thanks

References

1. Krige, D.: A statistical approach to some mine valuation and allied problems on the Witwatersrand (1951)
2. Rasmussen, C.E., Williams, C.K.: Gaussian processes for machine learning, vol. 1. MIT press Cambridge (2006)