Michael Teti

Gaussian Process Regression

12/13/2020

Introduction

In this report I will discuss Gaussian Process Regression (GPR), which is quickly becoming a popular tool in the machine learning toolkit for reasons that will be discussed here. First, I will motivate the use of GPR by first describing a simple, nonlinear regression problem with one predictor and one response (mainly because I don't fully understand how to incorporate multiple predictors at this point). Next, I will provide some examples and prerequisite material that are helpful in garnering an intuition for GPR, which will help build up to a formal description of the algorithm before showing how GPR can be used to model the nonlinear regression problem. Finally, I will end with a comparison against standard linear regression.

Nonlinear Regression

First, let us consider a nonlinear regression problem (**Fig. 1**) with one predictor x and one response variable y, where we have a training set, D, with n pairs of predictors and responses (x_1, y_1), (x_2, y_2), … (x_n, y_n). Given these pairs, the general goal is to predict the value of y given a new value of x (indicated by the question mark in **Fig. 1**). Typically, one would perform polynomial regression, which works relatively well in many settings, to obtain a polynomial which describes something like the red line in **Fig. 1**.

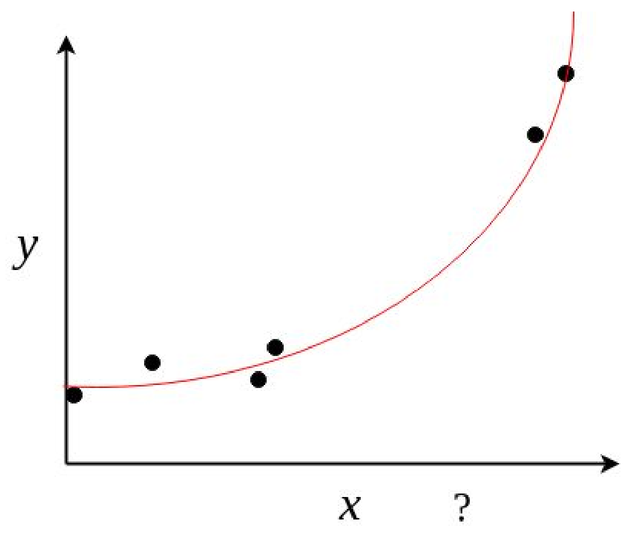

Fig. 1. Input-output pairs sampled from a nonlinear function.

To predict the response, y, given a new predictor, x, one would just insert the value of x into the equation of the best-fit line obtained during training. In the example illustrated in **Fig. 1**, there are not many training points around the value we wish to predict a response for, so it is possible that our best-fit line is not actually the best model for the true underlying data. In fact, by sampling more training points in this region, it is possible that the best-fit line would change dramatically from the one illustrated depending on where these new samples fell. One way to remedy this would be to collect a training set by densely sampling across the entire range of x values (or at least most of the range). However, this may not always be possible in practice for various reasons. The GPR model, alternatively, is able to elegantly handle the issue described above because it does not describe a single function (i.e. a single line in this case), but a *distribution over functions* [1]. It is probably not clear yet what this means, but my hope is that this will become more clear by the end of this report.

Multivariate Gaussian Distributions

Before moving forward, it is necessary to understand what a multivariate Gaussian distribution is, general properties of these distributions, and finally, how these properties relate to GPR and the nonlinear regression problem described above.

To begin, we will first describe a random variable X, which consists of components X_1, X_2, ..., X_k, as having a multivariate Gaussian distribution if every combination of its k components is normally distributed. Mathematically, this means that for any constant vector $a \in R^k$, $Y = a_1X_1 + a_2X_2 + \ldots a_kX_k$ is normally distributed. Just as a univariate Gaussian distribution is parameterized by its mean μ and standard deviation $\boldsymbol{\sigma}$, a multivariate Gaussian distribution is parameterized by a mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. An example of a 2D Gaussian distribution is displayed in **Fig. 2** via a contour plot. In this example, $\boldsymbol{\mu}$ is [0, 0] because the distribution for both X_1 and X_2 are centered at 0. Along the diagonal of $\boldsymbol{\Sigma}$ is the variance (^{2}) of each univariate distribution, whereas the off-diagonal elements indicate the covariance (similar to correlation) between X_1 and X_2. The covariance is positive and relatively strong, which is why the multivariate Gaussian is rather narrow and slanting toward the top right of the graph. Since both distributions are centered on the mean in this example, we only need the covariance matrix to compute probabilities. To do this, we use the following equation:

$$\mathrm{p}\,(x \mid \Sigma) \propto e^{(-\frac{1}{2}y^T\Sigma^{-1}y)} \qquad (1)$$

where $\boldsymbol{x}$ is the 2D vector of values corresponding to X_1 and X_2, and $\propto$ means proportional to up to a constant.

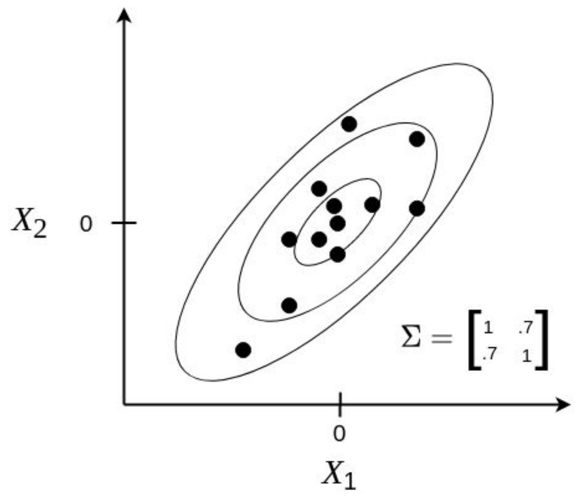

Fig. 2. A contour plot of a 2D Gaussian distribution. All points on a single contour line are constant probability. Probability increases while traveling toward the center, which is illustrated by the black dots which represent samples drawn from this distribution. Only zero axis labels are shown to illustrate that the mean is centered at zero for both distributions

It is also possible to compute the probability of one of the random variables conditioned on a value of the other given the following equation:

$$\mathrm{p}(X_2 \mid \Sigma, X_1) \propto e^{-\frac{1}{2}(X_2 - \mu_*)\Sigma_*^{-1}(X_2 - \mu_*)} \qquad (2)$$

where the conditional distribution is also a Gaussian, μ_* depends linearly on Σ and is the mean of this conditional distribution, and Σ_* represents the spread of the conditional distribution. As the off-diagonal entries in the covariance matrix tend toward zero and the variables become less and less correlated, μ_* tends toward the marginal mean of X_2 (i.e. 0 in this case), and Σ_* tends toward the marginal variance of X_2, (which is 1 in this case). Specifically, Σ_* increases as the correlation decreases because there is more uncertainty about the value of X_2 given X_1 as the two become less correlated. On the other hand, as the off-diagonal entries in the covariance matrix become more correlated, then Σ_*

tends toward zero and μ_* tends toward the mean of the conditional distribution. In the extreme case that the two variables were perfectly correlated, Σ_* would be zero. Essentially, there is no uncertainty because the second variable would be easily predicted by matching the value of the first variable. Therefore, the covariance between these two variables allows us to determine the range of values that one of them could take on with the other one fixed.

Gaussian Processes

So far, we have determined that if we have two normally distributed variables (or more), know how they are correlated, and fix one (or more) of them, then we have an estimate about the possible range of values the other one can take on and the distribution of these values. However, these examples have been dealing only with two variables with a pre-defined covariance matrix. To make this more general, we can define a covariance function:

$$\Sigma(X_1, X_2) = K(X_1, X_2) + I_y^2 \qquad (3)$$

where the second term describes the range of the response variable at a given input x and can be learned via Maximum Likelihood Estimation, and K is some covariance function chosen as a prior. Typically, it is chosen as the radial basis function

$$K(X_1, X_2) = {}^2e^{-\frac{1}{2l^2}(X_1 - X_2)^2} \qquad (4)$$

One thing to notice is that as the value of X_1 approaches the value of X_2, then K tends toward σ^2. The other thing to notice is that we have transitioned from being given a covariance matrix to having a function to be able to create one from any arbitrary real values along the function we are trying to fit. In other words, we can plug in any two continuous x values in our training dataset in **Fig. 1**, and we can then determine how the responses covary at those points. This particular covariance function chosen contains the parameter l, which stands for length and can roughly be interpreted as controlling the amount of expressivity in the model. For example, if l is larger, the line will fluctuate less, and if it is smaller, the line will fluctuate more. This parameter can also be found through Maximum Likelihood Estimation.

Now that we have described the covariance function, we can define the Gaussian Process. Formally, a Gaussian process is a collection of random variables, any finite number of which have a (consistent) Gaussian distribution, or in other words, it is a multivariate Gaussian distribution with infinitely many variables. It is fully specified by the mean function and covariance function, and it is typically written as follows

$$f(x) \sim GP(m(x),\ K(x,\ x')) \qquad (5)$$

where *m(x)* takes in the input x and returns the mean at that point, and it doesn't have to be continuous like you would expect in gradient-based models.

Gaussian Process Regression

Now that we have defined a Gaussian Process and described some characteristics of it, we can look at how to actually use them in a regression context to fit data and make predictions. First, let's revisit our nonlinear regression problem in **Fig. 1**. We have six observations which we would like to inform our model about what function or family of functions would be appropriate to model this data. In GPR, this set of known observations are typically called $\boldsymbol{y}_2$, and the rest of the points in between are called $\boldsymbol{y}_1$. To find these points, GPR formulates the problem using Bayes rules

$$p(y_1 \mid y_2) = \frac{p(y_1, y_2)}{p(y_2)} \qquad (6)$$

where the term on the left hand side of the equation is the conditional probability of the unobserved $\boldsymbol{y}_1$ points given the observed $\boldsymbol{y}_2$ points, and the denominator in the fraction is the marginal probability of the known observations. The numerator in the fraction on the right hand side is the joint probability of both y_1 and y_2 and expressed as

$$p\left(y_1, y_2\right) = \mathcal{N}\left(\begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}\right) \quad (7)$$

where a is the marginal mean of the distribution of y_1, A would be the covariance of the distribution of samples in y_1, and B would be the covariance between y_1 and y_2, for example. Since both of the terms on the right hand side of the equation are Gaussian, then $p(y_1 \mid y_2)$ is also Gaussian, which means that we can directly solve this with the equation

$$p(y_1 \mid y_2) = N(\mathrm{a} + BC^{-1}(y_2 - b),\ A - BC^{-1}B^T) \quad (8)$$

One thing to point out from this equation is that it involves taking the inverse of a large matrix (n x n if you have n known observations in y_2), which can be very computationally costly for n greater than about 10,000 or so. This is in contrast to the algorithms used for linear regression and its variants, where they are typically optimized iteratively through gradient descent and can be made to use mini-batches of samples at each iteration, and they don't involve taking the inverse of large matrices like GPR. From the equation above, we can interpret the first argument as being the conditional mean of y_1 given y_2. By looking at the second parameter, we can interpret this as reducing the uncertainty because A is the covariance (or uncertainty) of the points on the function we don't know (or prior uncertainty), and $BC^{-1}B^T$ can be interpreted as the reduction in uncertainty given the points we know. As $BC^{-1}B^T$ grows larger, there is less uncertainty about y_1.

Relationship to Standard Regression Models

GPR is essentially a non-parametric version of standard linear or nonlinear regression. For example, by looking at **Eq. 3** we see that it bears some semblance to the typical formulation for linear regression

$$y(x) = f(x) + \epsilon_y \qquad (9)$$

where $f(x) = b_0 + b_1x_1 + b_2x_2 + . . . b_px_p$ and p equals the number of predictor variables, the second term on the right hand side of the equation describes how much the responses y are spread out around the mean at each value of x, and ϵ is drawn from a normal distribution with mean zero and variance one. Another difference between the two models is that standard regression models are typically optimized by minimizing some estimate of the error, such as mean-squared error, through gradient descent, whereas GPR models attempt to minimize uncertainty through Bayesian methods.

References

1. Williams, Christopher KI, and Carl Edward Rasmussen. Gaussian processes for machine learning. Vol. 2. No. 3. Cambridge, MA: MIT press, 2006.
2. https://thegradient.pub/gaussian-process-not-quite-for-dummies/