Introduction to Gaussian Processes - RAW

Filipe P. Farias
Teleinformatics Engineering Department
Federal University of Ceará
filipepfarias@fisica.ufc.br

Abstract

The Gaussian processes have proven to be a powerfull framework for robust estimation and a flexible model for non-linear *regression*, case which will be the main object of this work, with some implementations of real situations.

1 Introduction

First, we'll overview some initial concepts that will set the background for the GP framework. Let's suppose a data set $\mathcal{D} = \{(x_i, t_i)\}_{i=0}^{N-1}$ which we denote the input as x, the output (or target) as t and N as the number of observations. The first step in our workflow is define a *training* set, i.e. some data that is given to make our first assumptions of the *model*. The model y can be defined as a guess of the law that rules the phenomenon of which our data was observed. This law can be, by example a senoid function as represented in Figure 1. Given this training data we wish to make *predictions* for new inputs x_* that we have not observed in the training set.

We'll assume an parametric approach, then the model is said to contain *parameters* $\mathbf{w}$, that will be adjusted during the *training phase*, when those are modified aiming to reduce the mismatch with the training set. In general we define a *loss function* $L(y,t)$ which increases itself as the mismatch becomes larger, in other words the *error* of the model. Then our work will be to reduce this error such that the smallest one will be the which defines when our model has learned the parameters of the law of the phenomenon. This turns possible to make our predictions where the data was not observed.

Unfortunately, in this trying of obtain the model by the smallest error, we may lose the capability of generalize it, i.e. our model could learned well for the training set only. So, if new data arrive or a new realization of the phenomenon occurs, that smallest error may increase for the same model. With this we define that our model isn't flexible. Then we can increase this flexibility by accepting some *uncertainty* above it. More, sometimes a good first assumption can make the difference to the estimation, and one may want to put its beliefs in the model even before to observe the data, i.e. make a *prior* assumption. These both strategies of uncertainty and prior assumptions are well defined by the Bayesian inference and could help if we assume a *probabilistic model*.

The Bayesian inference can handle with the classical approaches of search the model which has the smallest error, but our objective is achieve one step ahead. We can not only obtain one model, but a *distribution* of possible models. And with this, all the probabilistic meaning of distribution is carried with it, that is we can obtain both the model which *minimize* the error or the statistics of the distribution of models. A more explaining view of what this really means will be given in the next sections.

Furthermore, the concept of *infer* is similar to what we have done since beginning. We maded a guess of the law which rules the phenomenon, i.e. a prior assumption. Then we turns our model more plausible by reducing its error, or more *likely*. After, we obtained a result of these assumptions, a *posterior* assumption. These steps are similar in concept when dealing with Bayesian inference, except that, as we will deal with probability distributions, then some rules must be established for the method to be concise.

Finally, we'll deal with a specific class of models in which we assume not a distribution of parameters but functions in general. By example, in a space with infinite possible functions, we'll evaluate how much possible which one are to be generated the data by its statistics, what is similar to what was done for the parameters. And in this part we make the fully use of the Gaussian process.

2 Linear Regression

The following scenario is given. The data set of observations is given by $\mathcal{D} = \{(x_i, t_i)\}_{i=0}^{N-1}$, hence we make a guess about the law that rules the phenomenon behind the data. In general, we define that model as a mathematical function y whose parameters $\mathbf{w}$ will be adjusted in trying of learn the phenomenon law. By example, we may choose a polynomial as a model and try to adjust its coefficients, that will be the parameters for the model in this case.

$$y\left(x, \mathbf{w}\right) = w_0 x^0 + \cdots + w_{M-1} x^{M-1} \tag{2.1}$$

This approach is not linear in the inputs, i.e. the model output y it isn't a linear transformation of the inputs. But we can say that is *linear on the parameters.* To visualize this, considering the parameters array $\mathbf{w} = \left(w_0, \cdots, w_{M-1}\right)^\top$, we put in the matrix form as

$$y\left(x, \mathbf{w}\right) = \mathbf{w}^\top \boldsymbol{\phi}(x). \tag{2.2}$$

Here we have defined the array $\boldsymbol{\phi}(x) = \left(x^0, \cdots, x^{M-1}\right)^\top$ in order to keep the linearity of the model. This approach becomes more clear if we take $\boldsymbol{\phi}$ for all the space where we have the observations, by this we mean evaluate $\boldsymbol{\phi}$ for all the $\{x_i\}_{i=0}^{N-1}$ inputs, and we'll obtain

$$\begin{pmatrix} x_0^0 & \cdots & x_0^{M-1} \\ x_1^0 & \cdots & x_1^{M-1} \\ \vdots & \ddots & \vdots \\ x_{N-1}^0 & \cdots & x_{N-1}^{M-1} \end{pmatrix}. \tag{2.3}$$

This matrix is called the *design matrix* Φ. Hence for the vector with all the outputs, we would obtain of $\mathbf{y} = \mathbf{w}^\top \Phi$. These tools make possible gain some insight that help us in the follow steps. Backing to the data, let‘s take the targets in the vector form, this is $\mathbf{t} = \left(t_0, \cdots, t^{N-1}\right)^\top$, so we can consider that $\mathbf{t}$ is a *vector in space* [Bishop, 2006]. If we take the columns of Φ, denoting each one by $\boldsymbol{\varphi}_j$, what defines the vector $\mathbf{y}$ by its linear combination. If we take a polynomial of M^{th} order, being M smaller than the number of points N, than we say that $\boldsymbol{\varphi}$ will span a linear subspace $\mathcal{S}$ of dimensionality M. With this, our problem turns to actually train the parameters, what can be done defining some metric that say to us when the model is missing, as a *loss function.*

Maybe an intuitive way to say to the model if he's going far away from the target is measure this error, i.e. how much the model is *distant* from the target. We may generalize the distance itself by the *Minkowsky distance* L_p between $\mathbf{u}$ and $\mathbf{v}$, defined as

$$L_p\left(\mathbf{u}, \mathbf{v}\right) = \left(\sum_{i=1}^{n} |u_i - v_i|^p\right)^{1/p} \tag{2.4}$$

where we have the Euclidean distance for $p = 2$ and the also known Manhattan distance for $p = 1$. The question here is define what distance use, by example the Euclidean distance is the most common due to its analytical tractability and practical importance. With this choice the error between the model and the targets will be define as the distance between $\mathbf{y}$ and $\mathbf{t}$, and where the orthogonal projection of $\mathbf{t}$ onto the subspace $\mathcal{S}$ is the minimal distance. The *learning* step of the parameters $\mathbf{w}$ lies on choose those whose make that distance minimal. Hence, being $\langle \cdot, \cdot \rangle$ the inner product, we have

$$L_2(\mathbf{t} - \mathbf{y}) = \langle \mathbf{t} - \mathbf{y}, \mathbf{t} - \mathbf{y} \rangle^{1/2} \tag{2.5}$$

References

[Bishop, 2006] Bishop, C. M. (2006). Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag, Berlin, Heidelberg.

[Hennig, 2013] Hennig, P. (2013). Gaussian processes. Machine Learning Summer School 2013.