Over-Fitting in Model Selection with Gaussian Process Regression

Rekar O. Mohammed(✉) and Gavin C. Cawley

University of East Anglia, Norwich, UK
rekarmajidi@gmail.com, g.cawley@uea.ac.uk
http://theoval.cmp.uea.ac.uk/

Abstract. Model selection in Gaussian Process Regression (GPR) seeks to determine the optimal values of the hyper-parameters governing the covariance function, which allows flexible customization of the GP to the problem at hand. An oft-overlooked issue that is often encountered in the model process is over-fitting the model selection criterion, typically the marginal likelihood. The over-fitting in machine learning refers to the fitting of random noise present in the model selection criterion in addition to features improving the generalisation performance of the statistical model. In this paper, we construct several Gaussian process regression models for a range of high-dimensional datasets from the UCI machine learning repository. Afterwards, we compare both MSE on the test dataset and the negative log marginal likelihood (nlZ), used as the model selection criteria, to find whether the problem of overfitting in model selection also affects GPR. We found that the squared exponential covariance function with Automatic Relevance Determination (SEard) is better than other kernels including squared exponential covariance function with isotropic distance measure (SEiso) according to the nLZ, but it is clearly not the best according to MSE on the test data, and this is an indication of over-fitting problem in model selection.

Keywords: Gaussian process · Regression · Covariance function · Model selection · Over-fitting

1 Introduction

Supervised learning tasks can be divided into two main types, namely classification and regression problems. Classification is usually used when the outputs are categorical (discrete class labels), whereas, regression is concerned with the prediction of continuous quantities. Gaussian process is defined as a distribution over functions, and inference takes place directly in the space of functions, i.e. the function-space view. Gaussian process regression is not a new area of study, it has been extensively used in research areas such as machine learning, statistics and engineering. In the literature, Gaussian process regression has been widely used for many real-world problems, including time series analysis. For instance, Duvenaud et al. (2013) applied GPR to the total solar irradiance dataset and

© Springer International Publishing AG 2017
P. Perner (Ed.): MLDM 2017, LNAI 10358, pp. 192–205, 2017.
DOI: 10.1007/978-3-319-62416-7_14

obtained good results, and Williams and Rasmussen (2006) also used GPR for modelling atmospheric CO2 concentrations.

Model selection approaches for GPR seek to determine good values for the hyper-parameters of the model, typically via maximising the marginal likelihood or via cross validation (Williams and Rasmussen 2006). Cawley and Talbot (2007) discusses an over-fitting issue that arises in model selection with Gaussian processes classification. They claim that for GP classification, covariance functions with large parameters clearly demonstrate the over-fitting issue, where reducing the value of the model selection criterion results in a model with worse generalisation performance. This is because the model selection criterion is evaluated over a finite set of data, and hence is a performance estimate with a non-negligible variance.

In this paper, we first describe the background methodology for applications of Gaussian progress regression, and then give some examples of covariance functions commonly used in GPR. The reminder of the paper the describes model selection practices for GPR, and the causes of over-fitting in model selection, how one can detect it, and how this issue can be avoided. Finally we present empirical results using UCI benchmark datasets (2013), showing that over-fitting the model selection criterion is a potential pit-fall in practical applications and GPR, and present our conclusions.

2 Background

Regression analysis is a vital tool in applied statistics as well as in machine learning. It aims to investigate the influence of certain variables X on a certain outcome y (Walter and Augustin 2010).

The linear regression model is one of the most common models used to study the linear relationship between a dependent variable y and one or more independent variables X. The reason for its popularity is due to both the conceptual and computational simplicity of fitting a linear model. However, linear regression is dependent on some assumptions (Briegel and Tresp 2000), for example, the true relationship in the data must be approximately linear for good prediction using a linear model, but unfortunately this often is not the case for real-life data. Therefore, standard linear regression is generalized in many ways and here we use Bayesian linear regression as a treatment to the linear model (the following exposition is based on that given by Williams and Rasmussen 2006).

In Bayesian linear regression, we need to have a prior belief regarding the values of the model parameters that is combined with the likelihood function, describing the distribution of the data, to find the posterior distribution over the parameters. We can write down a generative model for our data.

$$f(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w}, \quad y = f(\boldsymbol{x}) + \varepsilon,$$

where $f(\boldsymbol{x})$ is our modelling function, ε is some form of additive noise, and y is the observed target values. The input vector is defined as $\boldsymbol{x}$ and parameter vector of the linear model as $\boldsymbol{w}$. We also assume that ε are an independent

and identically distributed (i.i.d.) sample from a zero-mean normal distribution, i.e. $N(0, \sigma_n^2)$. It follows that $y = \boldsymbol{x}^T\boldsymbol{w} + \varepsilon : \varepsilon \sim N(0, \sigma_n^2)$. Both noise and model assumptions enable us to identify the probability density of the observations given the parameters which is known as the Likelihood function, which is given by

$$\begin{aligned} p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{w}) &= \prod_{i=1}^{n} p(y_i \mid x_i, \boldsymbol{w}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{(y_i - \boldsymbol{x}_i^T\boldsymbol{w})^2}{2\sigma_n^2}\right), \\ &= \frac{1}{(2\pi\sigma_n^2)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma_n^2}|\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{w}|^2\right) = N(\boldsymbol{X}^T\boldsymbol{w}, \sigma_n^2\boldsymbol{I}), \\ &\propto \exp\left(-\frac{1}{2\sigma_n^2}(\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{w})^T(\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{w})\right). \end{aligned}$$

In Bayesian linear regression, we assume that a prior distribution over the parameters is also given. For example, a typical choice is $\boldsymbol{w} : N(0, \Sigma_p)$

$$\begin{aligned} p(\boldsymbol{w}) &= \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma_p|} \exp\left(-\frac{1}{2}\boldsymbol{w}^T\Sigma_p^{-1}\boldsymbol{w}\right), \\ &\propto \exp\left(-\frac{1}{2}\boldsymbol{w}^T\Sigma_p^{-1}\boldsymbol{w}\right). \end{aligned}$$

Now, by using Bayes' rule, we can obtain the posterior distribution for the parameters, which is given by

$$p(\boldsymbol{w} \mid \boldsymbol{y}, \boldsymbol{X}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w})}{\int p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w}}.$$

The denominator is known as the marginal likelihood $p(\boldsymbol{y} \mid \boldsymbol{X})$ and does not involve the parameters (weights), hence it can often be neglected. In the following steps, we get closer to the computation of the posterior distribution for the parameters.

$$\begin{aligned} p(\boldsymbol{w} \mid \boldsymbol{y}, \boldsymbol{X}) &\propto \exp\left(-\frac{1}{2\sigma_n^2}(\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{w})^T(\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{w})\right) \exp\left(-\frac{1}{2}\boldsymbol{w}^T\Sigma_p^{-1}\boldsymbol{w}\right), \\ &\propto \exp\left[-\frac{1}{2}(\boldsymbol{w} - \bar{\boldsymbol{w}})^T\left(\frac{1}{\sigma_n^2}\boldsymbol{X}\boldsymbol{X}^T + \Sigma_p^{-1}\right)(\boldsymbol{w} - \bar{\boldsymbol{w}})\right]. \end{aligned}$$

Therefore, the posterior is recognised as a Gaussian distribution with $\bar{\mathbf{w}} = \sigma_n^{-2} A^{-1} X\mathbf{y}$ as a mean and as a covariance matrix $A^{-1} = (\frac{1}{\sigma_n^2} XX^T + \Sigma_p^{-1})^{-1}$, i.e.

$$p(\boldsymbol{w} \mid \boldsymbol{y}, \boldsymbol{X}) : N(\bar{\boldsymbol{w}}, \boldsymbol{A}^{-1}).$$

Having specified $\boldsymbol{w}$, making predictions about unobserved values, $f(\boldsymbol{x}_*)$, at coordinates, $\boldsymbol{x}_*$, is then only a matter of drawing samples from the predictive distribution $p(f_* \mid \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})$ which is defined as:

$$p(f_* \mid \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int p(f_* \mid \boldsymbol{x}_*, \boldsymbol{w})p(\boldsymbol{w} \mid \boldsymbol{y}, \boldsymbol{X})d\boldsymbol{w}.$$

The predictive posterior is once again Gaussian:

$$p(f_* \mid \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \sim N(\sigma_n^{-2}\boldsymbol{x}_*^T\boldsymbol{A}^{-1}\boldsymbol{X}\boldsymbol{y}, \boldsymbol{x}_*^T\boldsymbol{A}^{-1}\boldsymbol{x}_*).$$

In fact, both the parameter posterior and posterior predictive distribution provide a useful way to quantify our uncertainty in model estimates, and to exploit our knowledge of this uncertainty in order to make more robust predictions on new test points (Do 2007).

2.1 Gaussian Processes in Regression

Over the past few years, there has been a tremendous interest in applying non-parametric approaches to real-world problems. Numerous studies have been devoted to Gaussian processes (GPs) because of their flexibility when compared with parametric models. These techniques use Bayesian learning, which usually leads to analytically intractable posteriors (Csató 2002), however that is not the case for GPR.

A Gaussian distribution is a distribution over random variables, $\boldsymbol{x} \in \mathbb{R}^n$, which is completely specified by a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$,

$$p(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right].$$

We can write this as $\boldsymbol{x} \sim G(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Gaussian random variables are very useful in statistics and machine learning because they are very commonly used for modelling noise in statistical algorithms (Do 2007).

According to Rasmussen (2004), a Gaussian process (GP) is defined as "a collection of random variables, any finite number of which have (consistent) joint Gaussian distributions". A Gaussian process is a distribution over functions which is fully specified by the mean function, $m(x)$, and a covariance function, $k(x, x')$, of a process $f(x)$, where

$$m(x) = E[f(x)], \tag{1}$$
$$k(x, x') = E[(f(x) - m(x))(f(x') - m(x'))]. \tag{2}$$

We can now obtain a GP from the Bayesian linear regression model in which, $f(x) = \boldsymbol{\phi}(x)^T\boldsymbol{w}$, with $\boldsymbol{w} : (0, \Sigma_p)$. Both mean function and covariance function are obtained as

$$E[f(x)] = \boldsymbol{\phi}(\boldsymbol{x})^T E(\boldsymbol{w}) = 0, \tag{3}$$
$$E[f(x)f(x')] = \boldsymbol{\varphi}(x)^T E[\boldsymbol{w}\boldsymbol{w}^T]\boldsymbol{\varphi}(x)^T\boldsymbol{\Sigma_p}\boldsymbol{\varphi}(x'). \tag{4}$$

Hence, $f(x)$ and $f(x')$ are jointly Gaussian with zero mean and covariance function $\boldsymbol{\varphi}(x)^T\boldsymbol{\Sigma_p}\boldsymbol{\varphi}(x')$.

The mean function is commonly defined to be zero, "which is not a strong limitation if the data is centred in preprocessing" (Blum and Riedmiller 2013). The covariance function defines the similarity between values of the function

as a function of the data points and plays an important role in controlling the properties of Gaussian Processes (Williams and Rasmussen 2006). Gaussian processes are a technique for expressing prior distributions over functions for one or more input variables. Given a set of inputs, $x^{(1)}, \ldots, x^{(n)}$, we can draw samples $f(x^{(1)}), \ldots, f(x^{(n)})$ from the GP prior:

$$f(x^{(1)}), \ldots, f(x^{(n)}) : (0, K).$$

Although drawing random functions from the prior is important, we want to extract the information that the training data delivers about the function.

Given a noise-free training data,

$$D = \{(x^{(i)}, y^{(i)}) \mid i = 1, \ldots, n\} = \{X, f\}.$$

according to GP prior, the joint distribution of the training outputs, f, and the test outputs f_* is given by

$$\begin{bmatrix} f \\ f_* \end{bmatrix} : \left(0, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right).$$

In order to make predictions, we need to obtain the posterior distribution over functions. It is also necessary to restrict the prior to contain only functions which agree with D. The posterior distribution is obtained from the condition $\{X_*, f_*\}$ on $D = X, f$, and it is Gaussian.

$$f_* \mid X_*, X, f : N(K(X_*,X)K(X,X)^{-1}f, K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*))$$

However, the data of real world problems are typically noisy. Thus we need to define a GP for noisy observations.

$$D = \{X, \boldsymbol{y}\}, \text{ where } \boldsymbol{y} = f + \epsilon.$$

We assume additive noise, $\epsilon \sim N(0, \sigma^2 \boldsymbol{I})$, and can derive the predictive distribution by conditioning on $D = \{\boldsymbol{X}, \boldsymbol{y}\}$ that gives a Gaussian with

$$\mu = K(X_*,X)[K(X,X) + \sigma^2 \boldsymbol{I}]^{-1}\boldsymbol{y}, \tag{5}$$

$$\Sigma = K(X_*,X_*) - K(X_*,X)[K(X,X) + \sigma^2 \boldsymbol{I}]^{-1}K(X,X_*). \tag{6}$$

Now if we give a new 'test' input $\boldsymbol{x}_*$, the predictive distribution of the corresponding $f(\boldsymbol{x})$ is readily obtained. In practice, the predictive mean, denoted μ, of the GP is used as a point estimate for the function output, while the variance can be interpreted as uncertainty bounds ($\pm 2\sigma$ error-bars) on this estimate (Girard and Murray-Smith 2005).

The main aim of using Gaussian processes regression is for prediction. In the case of having D-dimensional input vector $\boldsymbol{x}$ mapped onto an N-dimensional feature space, $\boldsymbol{m}$ is an $n \times 1$ vector and $\boldsymbol{\Sigma}$ is an $n \times n$ matrix. More computational power is needed for implementing Gaussian processes regression when we have multivariate inputs.

The covariance function of the Gaussian process, that allows the model to find the high-level description of the data properties, can be specified as a hierarchical prior. For example, covariance function is used to identify the inputs that are useful in predicting the response. Inference for these covariance hyper-parameters can be performed using Markov chain sampling (Bernardo et al. 1998).

2.2 The Covariance Functions

There are three main concerns in Gaussian processes regression, namely the choice of the covariance function, the selection of variables, and the choice of good values of hyper-parameters which effectively control the complexity of the model (Shi and Choi 2011). Choosing a suitable covariance kernel is crucial because it determines almost all generalization properties of a Gaussian processes model (MacKay 1999).

There are a variety of different covariance functions that can be used in a Gaussian processes regression model, including stationary and non-stationary covariance functions. Stationary covariance functions, which are invariant under translation, are the most often used in GPR. One can simply assume that the mean is constant (zero), which means the process is stationary (Shi and Choi 2011). Stationary covariance functions depend only on the distance between the inputs, $\boldsymbol{x}$, such that the covariance function expresses the covariance between y_p and y_q (Williams and Rasmussen 2006). The formula is written as,

$$\operatorname{cov}(f(\boldsymbol{x}_p), f(\boldsymbol{x}_q)) = k(\boldsymbol{x}_p, \boldsymbol{k}_q) = \exp\left(-\frac{1}{2}|\boldsymbol{x}_p - \boldsymbol{x}_q|^2\right).$$

1. Squared Exponential Covariance Function (SE):
   This function is a smooth function of the inputs and is a common choice of covariance function because it has some nice properties, namely it can be integrated against most functions that we need in Gaussian processes.
   The form is given by

$$k_{\text{SE}}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \sigma_f^2 \exp\left(-\frac{(\boldsymbol{x}_p - \boldsymbol{x}_q)^2}{2r^2}\right) + \sigma_\epsilon^2 \delta_{pq},$$

   where σ_f^2 is the magnitude, r is the length scale that characterize variation, and σ_ϵ^2 represents noise.
2. Automatic Relevance Determination Covariance Function (SE-ARD):
   The SE-ARD covariance function for multi-dimensional inputs is considered as a more general form of the squared exponential kernel:

$$k_{\text{SE-ARD}}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D} \frac{(\boldsymbol{x}_p^{(d)} - \boldsymbol{x}_q^{(d)})^2}{r_d^2}\right),$$

   The parameter r_d is the characteristic length scale of dimension d. The relevancy of input feature can be determined by r_d, for instance, If r_d is very large, then the feature is irrelevant (Snelson 2006)

3. The Matérn Covariance Function:
   The formula of this type of covariance function is given by

$$k_{\text{Matérn}}(x, x') = \frac{2^{1-v}}{\Gamma(v)} \left(\frac{\sqrt{2v}|\boldsymbol{x} - \boldsymbol{x}'|}{r} \right)^{v} K_v \left(\frac{\sqrt{2v}|\boldsymbol{x} - \boldsymbol{x}'|}{r} \right),$$

   where both v and r are positive parameters, v determines the smoothness and K_v is an amended Bessel function (Abramowitz 1966). When $v \to \infty$, then $k_{\text{Matérn}}(\boldsymbol{x}, \boldsymbol{x}')$ becomes squared exponential covariance function.
4. The Rational Quadratic Covariance Function (RQ):
   This kernel is equivalent to adding many SE kernels together with different length-scales. The form of the rational quadratic (RQ) covariance function is;

$$K_{\text{RQ}}(\boldsymbol{x}, \boldsymbol{x}') = \left(1 + \frac{|\boldsymbol{x} - \boldsymbol{x}'|^2}{2\alpha r^2} \right)^{-\alpha},$$

   where α determines the smoothness and r is the characteristic length, when $\alpha \to \infty$ then RQ is identical to the SE.
5. Polynomial Covariance Function:
   The Polynomial kernel is a non-stationary kernel that takes the following form

$$k_{\text{Poly}}(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} \cdot \boldsymbol{x}' + \sigma_0^2)^p,$$

   where $\sigma_0^2 > 0$ is a constant, trading off the effect of higher-order against lower-order terms in the polynomial, and the kernel is known as a homogeneous polynomial when $\sigma_0^2 = 0$, $p > 0$ is the polynomial degree, which is a natural number.

2.3 Model Selection for GP Regression

As mentioned previously, Gaussian processes are specified by their mean and covariance functions. The purpose of covariance function is to determine the similarity between data points that involved some free parameters known as hyper-parameters. Indeed, the hyper-parameters are useful since they allow for flexible customization of the GP to the problem. Therefore, it is necessary to select the covariance functions and its hyper-parameters appropriately by the so-called model selection process (Blum and Riedmiller 2013).

In literature, two techniques are most often discussed for model selection in Gaussian process regression, namely marginal likelihood maximisation and cross validation (Williams and Rasmussen 2006). We only describe the Marginal Likelihood method of selecting the model for GP regression, as that is the approach we adopt in our experiments.

A reliable framework for inference over the hyper-parameters is obtained via the Bayesian approach but good approximations are not easily derived, due to the required complex integrals over the hyper-parameters being analytically

intractable. In fact, it is not easy to know what the parameters of the model are because Gaussian process model is a non-parametric model.

One can obtain the probability of the data given the hyper-parameters $p(\boldsymbol{y} \mid \boldsymbol{X}, \theta)$ for GPs regression with Gaussian noise by marginalization over the function values f. The log marginal likelihood is given by

$$\log p(\boldsymbol{y} \mid \boldsymbol{X}, \theta) = -\frac{1}{2}\boldsymbol{y}^T \boldsymbol{K}_y^{-1}\boldsymbol{y} - \frac{1}{2}\log|\boldsymbol{K}_y| - \frac{n}{2}\log 2\pi.$$

where $\boldsymbol{K}_y = \boldsymbol{K}_f + \sigma_n^2 \boldsymbol{I}$ is the covariance function for the noisy output, $\boldsymbol{y}$, and $\boldsymbol{K}_f$ is the covariance function for the noise-free latent function, f. The first term from the above equation is known as a data-fit term, the second term is a complexity penalty, and the last term is a normalizing constant (Blum and Riedmiller 2013).

In order to tune hyper-parameters by maximizing the marginal likelihood, the derivatives of the log marginal likelihood with respect to the hyper-parameters are required:

$$\frac{\partial}{\partial \theta_j}\log p(\boldsymbol{y} \mid \boldsymbol{X}, \theta) = \frac{1}{2}\operatorname{tr}\left[(\boldsymbol{\alpha}\boldsymbol{\alpha}^T - \boldsymbol{K_y}^{-1})\frac{\partial K_y}{\partial \theta_j}\right], \text{ where } \boldsymbol{\alpha} = \boldsymbol{K}^{-1}\boldsymbol{y}.$$

From the above equation, "any gradient based optimization algorithm can be used to obtain the hyper-parameters that maximize the marginal likelihood of a GP. We will call this optimization procedure *training* the GP" (Blum and Riedmiller 2013).

3 Over-Fitting in Model Selection with Gaussian Processes in Regression

In this section, we first define an over-fitting issue that rises in the context of model selection in machine learning. Afterwards, the reasons for the occurrence of this problem will be discussed; we will also explain how one can detect this over-fitting issue in model selection with Gaussian processes algorithms. The methods of preventing this problem will also be described. Finally, results obtained on a suite of eleven real-world benchmark data sets will be demonstrated.

3.1 Over-Fitting in Model Selection

Over-fitting in machine learning refers to the fitting of a random noise in the data in addition to it's underlying structure by a statistical model. Over-fitting usually occurs when a model is too complicated, for example, when the parameters are excessively more than the number of observations. The potential consequence of an over-fitted model is poor predictive performance, as it can amplify very small fluctuations in the data (Joshi 2013). While the dangers of over-fitting in determining the parameters of a model (training) are well documented, the risk of over-fitting in tuning the hyper-parameters (model selection) is less well appreciated.

3.2 The Causes of Over-Fitting in Model Selection

When selecting a model, over-fitting often occurs due to the variance of the model selection criteria. Models are typically trained via performance maximization based on a finite set of training data, the efficiency of the model on the other hand is not dictated based on the performance of the model using the training data. It is instead established using the success and effectiveness of the model of handling unseen data. The problem of over-fitting is encountered when a model begins to memorize training data as opposed to learning to generalize from the observed trend in the training data. For instance, if the number of parameters is the equal to or greater than the number of data points available, a basic linear model or learning process will be able to perfectly estimate the training data merely through memorization of the entire training data set. However, such elemental models and processes will frequently fail significantly when estimating new data. As the basic model has not learned to generalize to any degree, we experience the over-fitting problem (Joshi 2013).

According to Dietterich (1995) the major complication of over-fitting usually emerges from the structure of the machine-learning tasks. A learning algorithm is trained on a training dataset, but then applied to provide estimations using new unseen data points. We are not necessarily concerned with the algorithm's accuracy on the training data, but instead achieving optimal predictive accuracy on these unseen data points. The scenario of "over-fitting" arises when we try too hard to find the very best fit to the training data (or to the model selection criteria) and thus risk that noise will be consumed in the data due to the model memorizing particular characteristics of the training data instead of discovering a general predictive rule.

3.3 Detecting Over-Fitting in Model Selection

According to Cawley and Talbot (2010), fitting a Gaussian process with the non-ARD (Auto Relevance determination) equivalent covariance function (the Radial Basis Function (RBF) covariance function) and comparing the test error rates, would seem like the most straightforward progression to do. For several reasons, the ARD covariance function fails to perform as well as the non-ARD covariance function due to the over-fitting in tuning the hyper-parameters. The RBF is a special case of ARD where parameters constrained to be equal. Having fewer parameters gives less scope for over-fitting.

3.4 Avoiding Over-Fitting in Model Selection

Over-fitting mainly occurs when a small dataset is used. Therefore, it is always better to have a large data set. Thus, by using a lot of patterns the problem can potentially be avoided. However, having an excessively high number of data points, the algorithm is obliged to generalize and come up with a good model to fits all the points, without having sufficient capacity to model the noise. The convenience of choosing a large database does not always exist. There are

times where a small database is the only available option, limiting our choice of model development. In such cases, a technique called cross validation can be used. This technique divides the dataset into training and testing datasets. The model is developed using the training dataset and the validity of the model is tested using the testing database. This process is then repeated using various partitions of training and testing datasets. As a result of this technique, a fairly good approximation of the underlying model is given, due to the fact that it is tested on several partitions to achieve generalization at the maximum possible degree (Joshi 2013).

According to Cawley and Talbot (2010) over-fitting in model selection may seem logical, if a model selection criterion estimated over a specific number of data observations is directly optimized. For example, over-fitting in model selection, similarly to over-fitting in training, can be significantly harmful when the data sample is small and the population of hyper-parameters to be tuned is large. Similarly, under the assumption that further data are unavailable, possible solutions to the over-fitting the model selection criterion may be analogous to the solutions for the over-fitting the training criterion which has been tried and tested.

4 UCI Benchmark Datasets Used in Empirical Demonstrations

In this section, we use eleven benchmark data sets from the UCI machine learning repository (Bache and Lichman 2013) to examine the problem of over-fitting in model selection for Gaussian processes regression. Table 1 shows the details of the datasets, including the number of features, and test patterns for each dataset.

Table 1. Details of data sets used in empirical comparison.

Data set	Training patterns	Testing patterns	Number of replications	Input features
Airfoil self noise	1353	150	100	5
Community crime	1792	199	100	99
Concrete	927	103	100	8
dat	203	22	100	2
Energy Efficiency	692	76	100	8
Fertility	90	10	100	8
Housing	456	50	100	13
Istanbul Stock Exchange	483	53	100	8
Mpg	359	39	100	7
Servo	151	16	100	4
Yacht Hydrodynamics	278	30	100	6

4.1 Results and Discussion

In order to examine whether the problem of over-fitting during model selection is encountered with Gaussian processes regression, we find both mean squared error (MSE) and negative log marginal likelihood (nLZ) of seven kernel functions over a suite of eleven benchmark datasets. MSE is found based on the test set as a performance evaluation criteria, while nLZ is evaluated over the training set and used as a model selection criteria. Afterwards, the Friedman test is used to determine whether there are statistically significant differences in either MSE or nLZ for different covariance functions. This test is illustrated by critical difference diagrams (Friedman test with Post-Hoc test) (Demšar 2006), which shows the average ranks of seven kernels, as shown in Fig. 1.

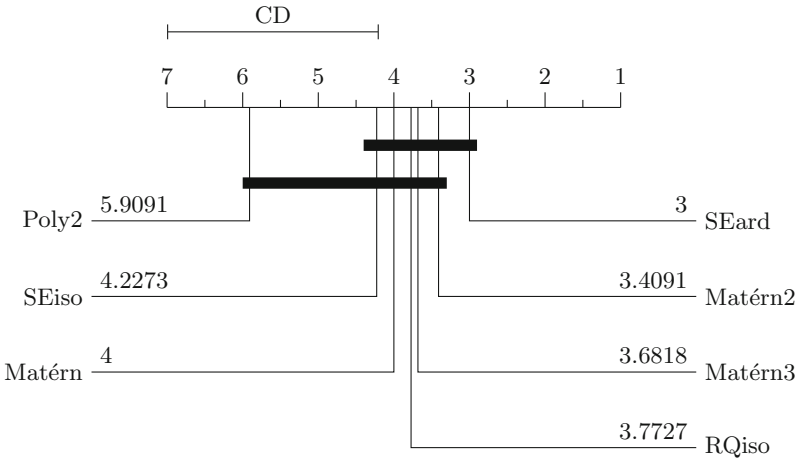

Fig. 1. Critical difference diagram showing the average ranks of seven kernels with using mean squared error (MSE)

This diagram shows the bold bars that joins the lines, such that if two or more lines (representing models with different covariance functions) are joined by a bar, it means these models are not statistically significantly different from each other. It clearly shows that only poly2 is statistically worse than SEard, in terms of generalisation performance, and the remaining differences are non-statically significant.

Figure 2 shows the average ranks of seven kernels with using negative log marginal likelihood. For the majority of the benchmarks, the lowest negative log-likelihood is obtained using SEard which is not surprising because it has more hyper-parameters. However, this is not a good result since SEard does not always give the minimum MSE compared to SEiso. This is called "over-fitting in model selection". In other words, when we have such a problem the negative log-likelihood is no longer a good indication of performance of the model. Indeed,

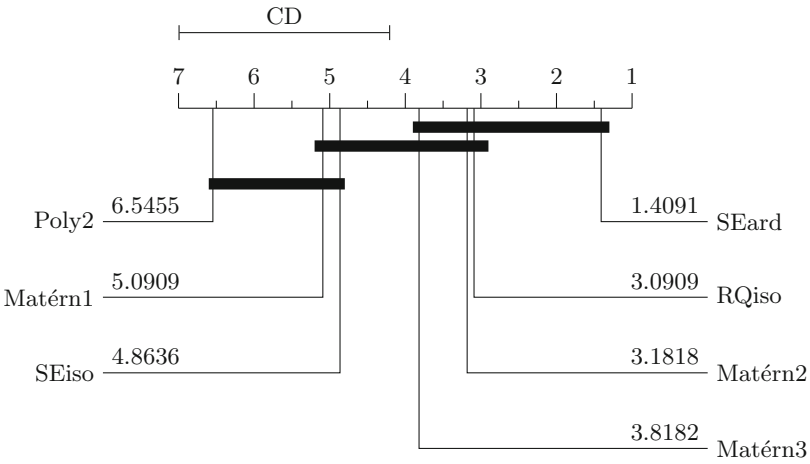

Fig. 2. Critical difference diagram showing the average ranks of seven kernels with using negative log marginal likelihood (nLZ)

the SEiso kernel is a special case of SEard kernel because both are squared exponential function. Thus, we should always obtain better negative log likelihood for SEard than SEiso simply because of having a lot of different parameters to be changed. On the other hand, sometimes the choice of hyper-parameters will result in a model over-fitting the model selection criteria or it may result in under-fitting the data rather than over-fitting it. The significantly lower negative log marginal likelihood of the SEard covariance over the SEiso is not reflected in the statistically insignificant difference in generalisation performance.

Figure 2 shows that SEiso is not significantly worse than SEard, while having fewer hyper-parameters. This is interesting result because it suggests that unlike classification datasets investigated by Cawley and Talbot (2010), the regression data sets are less susceptible to be over-fitting in model selection. Although, there is a great difference between SEard and the rest of the kernels used, SEard still performs well in terms of MSE. This suggests that over-fitting is still a problem but not as much as a problem in classification. In brief, we found that SEard kernel is better than most other kernels including SEiso according to the marginal likelihood, but it is not clearly the best according to MSE on the test datasets, and this is an indication of over-fitting problem. It is worth mentioning that the datasets used in this study were all rather small, however there are algorithms for large scale GP as it is described in the GPML web page by Williams and Rasmussen (2006), but the problem with over-fitting the model selection is most apparent with small datasets, hence there is unlikely to be a significant problem for larger datasets.

5 Conclusion

The contribution of this paper is to find whether the problem of over-fitting in model selection takes place with Gaussian processes regression, both mean squared error (cross validated MSE) and negative log marginal likelihood (nLZ) were found for seven kernel functions over a suit of eleven benchmark datasets. The negative log marginal likelihood is the model selection criteria that can be optimized, whereas the MSE is the test criteria. Afterwards, Friedman test was used to determine whether there is a statistically significant difference in either MSE or nLZ for different covariance functions. For the majority of the benchmarks, the lowest negative log marginal likelihood was obtained using SEard kernel which is not surprising because it has more hyper-parameters. We found that SEard kernel was clearly better than other kernels including SEiso according to the marginal likelihood, but it was clearly not the best according to MSE on the test datasets, and this is an indication of over-fitting problem. This is because the negative log marginal likelihood is the model selection criteria thus it is always decreasing and MSE is getting worse or not improving. We conclude that over-fitting is still a problem in GPs regression but not as much as a problem in GPs classification.

References

Abramowitz, M.: Handbook of mathematical functions. Am. J. Phys. **34**(2), 177 (1966)

Bernardo, J., Berger, J., Dawid, A., Smith, A., et al.: Regression and classification using Gaussian process priors. Bayesian Stat. **6**, 475 (1998)

Blum, M., Riedmiller, M.A.: Optimization of Gaussian process hyperparameters using Rprop. In: ESANN (2013)

Briegel, T., Tresp, V.: Dynamic neural regression models, Collaborative Research Center 386, Discussion Paper 181(2000)

Cawley, G.C., Talbot, N.L.C.: Preventing over-fitting during model selection via Bayesian regularisation of the hyper-parameters. J. Mach. Learn. Res. **8**, 841–861 (2007)

Cawley, G.C., Talbot, N.L.C.: On over-fitting in model selection and subsequent selection bias in performance evaluation. J. Mach. Learn. Res. **11**, 2079–2107 (2010)

Csató, L.: Gaussian processes: iterative sparse approximations. Ph.D. thesis, Aston University (2002)

Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**(January), 1–30 (2006)

Dietterich, T.: Overfitting and undercomputing in machine learning. ACM Comput. Surv. (CSUR) **27**(3), 326–327 (1995)

Do, C.B.: Gaussian processes (2007). http://see.stanford.edu/materials/aimlcs229/cs229-gp.pdf. Accessed 28 June 2014

Duvenaud, D.K., Lloyd, J.R., Grosse, R.B., Tenenbaum, J.B., Ghahramani, Z.: Structure discovery in nonparametric regression through compositional kernel search. In: ICML, vol. 3, pp. 1166–1174 (2013)

Girard, A., Murray-Smith, R.: Gaussian processes: prediction at a noisy input and application to iterative multiple-step ahead forecasting of time-series. In: Murray-Smith, R., Shorten, R. (eds.) Switching and Learning in Feedback Systems. LNCS, vol. 3355, pp. 158–184. Springer, Heidelberg (2005). doi:10.1007/978-3-540-30560-6_7

Joshi, P.: Overfitting in machine learning (2013). https://prateekvjoshi.com/2013/06/09/overfitting-in-machine-learning/

Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml

MacKay, D.J.C.: Comparison of approximate methods for handling hyperparameters. Neural Comput. **11**(5), 1035–1068 (1999)

Rasmussen, C.E.: Gaussian processes in machine learning. In: Bousquet, O., Luxburg, U., Rätsch, G. (eds.) ML -2003. LNCS, vol. 3176, pp. 63–71. Springer, Heidelberg (2004). doi:10.1007/978-3-540-28650-9_4

Shi, J.Q., Choi, T.: Gaussian Process Regression Analysis for Functional Data. CRC Press, Boca Raton (2011)

Snelson, E.: Tutorial: Gaussian process models for machine learning. Gatsby Computational Neuroscience Unit, UCL (2006)

Walter, G., Augustin, T.: Bayesian linear regression — different conjugate models and their (in)sensitivity to prior-data conflict. In: Kneib, T., Tutz, G. (eds.) Statistical Modelling and Regression Structures, pp. 59–78. Springer, Heidelberg (2010)

Williams, C.K., Rasmussen, C.E.: Gaussian processes for machine learning. **2**(3), 4 (2006). The MIT Press. ISBN 0-262-18253-X