Feature Selection in Gaussian Process Regression

Fall 2022 project - Final Report

Author
Romain Bourgeois
Supervisor
Motonobu Kanagawa

Fall semester 2022
https://github.com/romainremyb/GPR_ARD

Abstract

Using systemic experiments to control for varous regression problem conditions, this project compares methods for optimizing critical lengthscale values for Automatic Relevance Determination in GPRs. It also proposes an explanatory analysis of the factors impacting optimal threshold levels. The main components investigated are the number of variables and their relevance distribution to the target variable, the level of covariances between features and their non-linearity levels.

Contents

Chapter 1

Introduction

There are many reasons why one would want to only model the relevant variables. First, overfitting is a big issue that arises with supervised learning problems. Modelling irrelevant variables in any models would cause the algorithm to memorize on those features and generalization on new data could be deteriorated. Another reason could be the associated costs with retrieving, measuring or modelling with many variables. Finally, it is important to rely on interpretable models in general. The interpretability issue that arises in Gaussian Processes is the reason why these models are considered to be black box algorithms.

Chapter 2

Background

2.1 Gaussian Process Models

Gaussian processes (GPs) are nonparametric models that define a prior distribution directly in the space of latent functions f(x), where x is a k-dimensional input vector. The form of the functions generated by a GP relies on its covariance function k(x,x) between the latent function values at the input points x and x. This may also be seen as a similarity measure that goes to zero for unsimilar vectors. The prior is typically assumed to have zero mean:

$$p(f(X)) = p(f) = N(f|0, K), \tag{2.1}$$

where K is the covariance matrix between the la- tent function values at the training inputs $X = (x^{(1)}, ..., x^{(n)})$ such that $K_{ij} = k(x^{(i)}, x^{(j)})$.

When dealing with regression problems that involve Gaussian observation models, it is possible to obtain an analytically tractable Gaussian process (GP) posterior distribution for both noisy and noise-free observations. This can be achieved by conditioning the joint normal distribution of test and training outputs on the data that has been observed (Osborne, Ebden et al, 2013):

$$f_*|X, y, X_* \sim N(\bar{f}_*, cov(f_*)) \tag{2.2}$$

where $\bar{f}_* \equiv E[f_*|X, y, X_*] = K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} y$,
$cov(f_*) = K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} K(X, X_*)$.

2.2 Automatic Relevance Determination with a Gaussian kernel

This project uses the Gaussian kernel also called RBF:

$$k(x_1, x_2) = \sigma_f \exp(-\frac{1}{2}(x_1 - x_2)^T \Theta^2 (x_1 - x_2)) \tag{2.3}$$

where Θ is the lengthscale parameter and σ_f is an optional output scale parameter. In this Gaussian Processes Regression setting, Automatic Relevance Determination (ARD) essentially means extending the singular lengthscale parameter to all features:

$$k(x_1, x_2) = \sigma_f \exp(-\frac{1}{2}\Sigma_{i=1}^{k}\frac{(x_{1,i} - x_{2,i})^2}{t_i^2}) \tag{2.4}$$

Kernels are covariance functions measuring similarity between data points and ranges from 0 to 1. ARD relies on the idea that optimizing these parameters could display feature importance in measuring similarity between data points in the context of predicting a target variable. In order to account for the fact that the expected level of the squared difference of feature data points is the variance of the feature, normalizing features is required to enable lengthscale comparisons. Therefore, the higher a lengthscale is, the less it contributes to the kernel value and the model.

Chapter 3

Methodology

3.1 Dataset generation in experiments

3.1.1 Method

The goal is to test the feature selection methods under various regression problem conditions. There are 200 experiment trials with each unit having a random number of features and the variables are generated from a multivariate Gaussian distribution. Covariates are selected randomly but a method was built to ensure that these covariances make sense statistically. For k features, the method actually randomly produces (k-1) covariates and generates (k-1) features using (k-1) two-variate Gaussians. Then, covariances of the features are estimated and the final k-feature dataset is produced using these estimates from a multivariate Gaussian.

Each feature contributes to the model with its associated response function and the variables have no interaction effects to the target variable. The datasets are made up of four different types of response functions. Each time a feature is generated, its parameters are calculated given a randomly picked level of variance of its response function that is used as a proxy for feature relevance. In order to pull a metric on their level of non-linearity, linear regressions of the responses are fitted and the R square is used as the metric. One type of function is a non-predictive one with f(x)=0. The other three will then be explained.

3.1.2 Linear function responses

$$f(x) = w \times x \tag{3.1}$$

with $Var(f(X)) = w^2 \times Var(X)$

3.1.3 Polynomial function responses

$$f(x) = a \times x^2 + b \times x \tag{3.2}$$

with $Var(f(X)) = a^2Var(X^2) + 2abVar(X) + b^2Var(X)$.

Approximating given that X is normally distributed gives $Var(X^2) \approx 4\mu^2\sigma^2 + 2\sigma^4$.

(a,b) are computed such that the variance of the response function equals the randomly picked level. Setting b=0 we may calculate the corresponding a^*. a is then generated from a uniform law in $[-a^*, a^*]$. Finally, solving the resulting system enables to get b.

3.1.4 Sinusoidal function responses

$$f(x) = A \times sin(w \times x) \tag{3.3}$$

with $Var(f(X)) \approx h'(\mu)^2\sigma^2 + 0.5h''(\mu)^4$

If the variance of x is 1, the function reaches extreme non linearity when w goes to π and $-\pi$. Therefore w was uniformly picked between $[-\frac{\pi}{Var(X)}, \frac{\pi}{Var(X)}]$. The scale factor A was hence calculated to ensure that the response variance is at the expected level.

3.1.5 Additional comments

Parameters	Generation method
Number of features	$\sim U(5, 12)$
Sample size	$int(X), X \sim N(2500, 1000)$
Intercept	$\sim N(0, 3)$
Feature means	$\sim N(0, 3)$
Feature standard deviations	$abs(X), X \sim N(0, 3)$
(k-1) covariances to feature 1	$\sim N(0, 0.15) with - 0.7 < X < 0.7$
Mean of response function variances	$abs(X), X \sim N(1, 0.5)$
Variance of response function variances	$abs(X), X \sim N(0.25, 0.1)$
p-value for Kolmogorov-Smirnov test on target	1%
Added noise (as fraction of σ_y)	$abs(X), X \sim N(0.05, 0.1)$

Table: Hyper parameters for generating the datasets.

The table displays the hyper parameters used for generating the datasets in the experiment. It shall be noted a conditioned loop is also set up to check for non-positive semi-definite matrices. A dataset also needed to pass the Kolmogorov-Smirnov test to be validated.

3.2 Feature selection methods

3.2.1 Model specifications

The models used in the experiment rely on the above kernel and are implemented with GPyTorch both with and without an global output scale parameter σ_f.

Each of these models is to be trained with and without l1 and l2 regularization. The experiment consists of fitting each of the six models to the 200 experiment trials.

The models are run with early stopping and a maximum number of iterations. 20% of the datasets are used as validation data and training stops when the marginal log likelihood deteriorates for two consecutive iterations.

Furthermore, a learning rate (LR) grid search with early stopping has been implemented to decide on the appropriate LR and on the maximum epoch hyperparameters. Models with output scale will use a learning rate of 0.35 and a maximum iteration of 35 steps. The trials without output scale will on the other side use 0.2 and 40.

3.2.2 Regularization

From equation (1.2), it was shown that a high lengthscale suggests that its corresponding feature could be irrelevant. Both l_1 and l_2 regularization will be done on the lengthscale effect, that is $\frac{1}{l_i^2}$.

$$l_1 \times \Sigma_{i=1}^{k} |\frac{1}{l_i^2}| \tag{3.4}$$

$$l_2 \times \Sigma_{i=1}^{k} |\frac{1}{l_i^2}|^2 \tag{3.5}$$

The l_1 regularization is added to the marginal log likelihood function and is responsible for adding sparsity to the parameters. This means getting as many near zero values and few non-zero. It encourages the model to focus on a smaller set of features that are most relevant for predicting the outcome, while ignoring the rest.

On the other hand l_2 regularization has an opposite effect towards sparsity and the model would penalize the features that contribute the most to the target level and variance.

This effects are controlled by the coefficients l_1 and l_2 and are set arbitrarily at 0.1. l_1 regularization is expected to work better in AMD.

3.2.3 Parameters pulled during the experiment

Symbol	Parameters
meanR^2	Mean R^2 score, weighted with response function variances
stdR^2	Standard deviation of R^2 scores, weighted with response function variances
meanCovs	Mean of covariances between features
stdCovs	Mean of standard deviation of features
$meanCovs_{ir,r}$	Mean covariance between relevant and irrelevant features, weighted with response function variances
$stdCovs_{ir,r}$	Standard deviation of covariance between relevant and irrelevant features, weighted with response function variances
Nf	Number of features
R_{ir}	Ratio of irrelevant features to total number of variables
σ_y	Standard deviation of the target variable
$p(y\|X_{test})$	Marginal log-likelihood of the model on testing dataset

Table: Parameters pulled during experiment units

3.2.4 AMD scores and optimizing for threshold/critical values

In order to evaluate the general accuracy of the ARD method, one must determine lengthscale critical/threshold values that are general to most regression problem configurations. Even though the features are standardized, threshold values must be determined as a function of the number of parameters. This project proposes to scale features' lengthscale effects to the sum of the effects over all variables:

$$score_j = \frac{\frac{1}{l_j^2}}{\sum_{i=1}^{k} \frac{1}{l_i^2}} \tag{3.6}$$

This AMD technique suggest that an irrelevant feature could be detected if its associated score is small enough. This project investigate optimum threshold values for such a decision. To do so, it will first optimize the critical value on all experiment units. It will then optimize the threshold to each units individually in order to conduct an explanatory analysis.

Two optimizing functions will be used. The first one consists of a binary function, taking one if the threshold takes out all irrelevant features and does not remove relevant ones.

Another function is a distance measure that accounts for how good and bad a threshold actually is on the overall dataset. The loss on an experiment unit is defined as follows:

$$loss_i = (S_{min(l_i)_{nr}} - t)^2 + w \times (1 - Rvar(max(l_i)_r)) \times (S_{max(l_i)_r} - t)^2 \tag{3.7}$$

where $S_{min(l_i)_{nr}}$ is the feature score of the smallest (least probable) irrelevant lengthscale. $S_{max(l_i)_r}$ is the feature score of the biggest (less probable) relevant lengthscale. $Rvar(max(l_i)_r)$ is the variance ratio of the biggest relevant lengthscale, its distance effect hence goes to 1 as the metric goes to zero (point of non-relevance). t is the threshold score and w controls for the weight to the relevant feature distances. As removing relevant variables should be more costly, the parameter typically takes a value smaller than one.

The critical parameter values are optimized with random search algorithms. More specifically, it uniformally tries thresholds on a given ranges and saves the best fits. It is followed from a random search around the mean of the best scores. The noise parameter is determined with the maximum distance to the mean from the best scores array.

3.2.5 In sample (experiment unit) critical value optimization

Developing the distance optimization function gives:

$$loss_i = (1 + wRvar(max(l_i)_r))t^2 - 2(S_{min(l_i)_{nr}} + wRvar(max(l_i)_r)S_{max(l_i)_r})t + S^2_{min(l_i)_{nr}} + wRvar(max(l_i)_r)S^2_{max(l_i)_r}) \tag{3.8}$$

$1 + wRvar(max(l_i)_r) > 0$ therefore critical values are maximized when

$$t^* = \frac{S_{min(l_i)_{nr}} + wRvar(max(l_i)_r)S_{max(l_i)_r}}{1 + wRvar(max(l_i)_r)} \tag{3.9}$$

Chapter 4

Results

4.1 Global threshold optimization

Output scale	L1 regularization	L1 regularization	critical value	score in %
No	No	No	0.10088	5%
No	Yes	No	0.114607	5.5%
No	No	Yes	0.069431	4%
Yes	No	No	0.11426	4%
Yes	Yes	No	0.16885	3.5%
Yes	No	Yes	0.04479	4%

Table 4.1.a: Global optimization for the binary optimizing function

Output scale	L1 regularization	L1 regularization	critical value	loss mean score
No	No	No	0.1153	0.0052
No	Yes	No	0.11805	0.00467
No	No	Yes	0.12311	0.00245
Yes	No	No	0.11851	0.00497
Yes	Yes	No	0.12258	0.00194
Yes	No	Yes	0.12313	0.00223

Table 4.1.b: Global optimization for the distance optimizing function

In terms of binary scores, models without an output scale parameter and regularization perform better. It is unclear whether L1 outperforms L2. It shall be noted that L2 based optimal lengthscale are much smaller than for other models. It is counter-intuitive because sparser lengthscale should push up the threshold limit.

On the other hand, the distance function yields a much different prospective on the matter. L2 based regularization models yield higher than average threshold as one should expect. Additionally, models with an output scale parameter

perform better. It is also surprising to see that without output scale, it is the L2 regularization that yields the best score. Overall best score is never the less achieved with L1 regularization as expected, with the model that uses an output scale parameter.

4.2 In sample (experiment units) threshold optimization

Output scale	L1 regularization	L1 regularization	Mean critical value	Std critical value
No	No	No	0.1116	0.0419
No	Yes	No	0.1141	0.0392
No	No	Yes	0.1218	0.0348
Yes	No	No	0.1152	0.0417
Yes	Yes	No	0.1230	0.0340
Yes	No	Yes	0.1224	0.0342

Table 4.2: In sample optimization function

This table depicts the moment of the in sample optimal threshold level. It matches with the above conclusions in that the less volatile model is the one with output scale using the L1 regularization.

4.3 Explanatory analysis

A linear regression is further fitted on the variables pulled from the experiment. The regression aims to predict the optimal critical score. The model used was the one yielding the best results: L1 regularization with output scale.

Variable	coef	p-value
constant	0.0076	0.730
$meanR^2$	0.0067	0.328
$stdR^2$	-0.0494	0.003
$meanCovs_{ir,r}$	0.0701	0.584
$stdCovs_{ir,r}$	0.0216	0.839
meanCovs	-0.0013	0.964
stdCovs	-0.0109	0.487
Nf	2.727e-05	0.985
R_{ir}	0.9878	0.000
σ_y	-0.0031	0.002
$p(y\|X_{test})$	8.863e-07	0.813

Table 4.3: Coefficients and p-values

The above table displays the linear coefficients and p-values of the variables pulled from the experiment. It shows that only a small portion of these variables

have a predictive power over the score threshold level. In fact, the weighted standard error of the non-linearity level decreases the threshold value. As expected, the rate of irrelevant features increases the threshold level. Finally, the variance of the target variable reduces the critical score.

Chapter 5

Conclusion and Future works

This project has tried to assess different methods for selecting features based on AMD. The experiment suggests that both output scale and L1 regularization improve this selection method. However, it shall be noted that further tests on learning rate parameter may be conducted to improve this inference.

Additionally, it has inspected the reasons for the volatile nature of critical score values. The analysis infers that only to target variable noise, the variance in the weighted non-linearity levels and the ratio of irrelevant features were useful when predicting the threshold scores. From these variables, only the noise of target variable may be used with confidence on testing data to improve the method, the other ones being inherently unknown in the non-experimental settings. Unfortunately, this project has not tested such a model on new data.

There are several things that would have been relevant to add to the research. On one hand, it would have been useful to model score rankings based on response function variances. Non Gaussian features have not been assessed and the explanatory analysis misses the variance on response function variances.

Bibliography

[1] Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N., Aigrain, S. (2013). Gaussian processes for time-series modelling. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 371(1984), 20110550.