ISSN: 2832-790X
DOI: 10.33552/OJRAT.2023.02.000528
Online Journal of Robotics & Automation Technology
Iris Publishers

Review Article

Comparative Study of Linear Kernel and Gaussian Kernel in Gaussian Process Regression

Hao Yaxian[1], XieTao[2] and Xing Yanyuan[3]*

[1]*School of Mathematics and Computer Science, Shanxi Normal University, Taiyuan, China*

[2]*Kyungil University, Daegu, korea*

[3]*Department of Mathematics, Changzhi University, Changzhi, China*

***Corresponding author:** Xing Yanyuan, Department of Mathematics, Changzhi University, Changzhi, China.

Received Date: May 24,2023

Published Date: June 08,2023

Abstract

Gaussian process regression is a powerful Bayesian theory that can effectively help predict data. In this paper, the influence of different kernel functions on the prediction results of house price data set in Gaussian process regression is studied. Linear kernel function and Gaussian kernel function are used to predict Gaussian process regression. Through comparison experiment, it is found that the choice of kernel function has great influence on the prediction result. By analyzing the convergence rate of the training set and the accuracy of the verification set, it is concluded that better results can be obtained by using linear kernel function for Gaussian process regression for this data set Index Terms—Gaussian Process Regression;Kernel function, Likehood.

Introduction

Machine learning [1, 2] is the core of artificial intelligence. Its main idea is to use the existing data that has been mastered, analyze and process the data, and seek rules from it. Finally, predict the unknown through the analysis of known data, and the most critical one is machine learning algorithm. In recent years, Gaussian process [3,4,5] has attracted wide attention due to its effectiveness. Gaussian process is an effective machine

learning method [6,7], which combines both Bayesian theory and kernel theory, and thus has the advantages

of both machine learning methods. Gaussian process can realize both classification and regression [7], which has certain advantages. Gaussian process model [8,9], as an excellent machine learning model in the field of regression and classification, has been widely concerned, and many excellent successes have been born at the right moment. Gaussian process has a lot of applications in practical problems [10,11]. The second part describes the process from unary Gaussian distribution to multivariate Gaussian distribution and then to Gaussian, and introduces the kernel function used in this paper. The third part strictly deduces the Gauss process regression model.

The fourth part is the experimental part of the algorithm. Three evaluation criteria are used to analyze the

performance of the model with different kernel functions, and the results are obtained.

Gaussian Process Regression

Gaussion distribution

The simplest and most common unary Gaussian distribution [12], the probability density function is

$$p(x)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \quad (1)$$

μ is mean and σ is variance, and the probability density function is drawn as the familiar bell curve. From unary Gaussian distribution to multivariate Gaussian distribution [13] [14], it is assumed that each dimension

is independent of each other

$$p(x_1,x_2,.....,x_n)=\prod_{i=1}^{n}p(x_i)$$

$$=\frac{1}{(2\pi)^{\frac{n}{2}}\sigma_1\sigma_2....\sigma_n}\exp\left(-\frac{1}{2}\left[\frac{(x_1-\mu_1)^2}{\sigma_1^2}+.....+\frac{(x_n-\mu_n)^2}{\sigma_n^2}\right]\right) \quad (2)$$

u_1, u_2, ... and σ_1, σ_2, ... is mean and variance. Make

$$x-\mu=\left[x_1-\mu_1,x_2-\mu_2,...x_n-\mu_n\right]^T$$

$$K=\begin{bmatrix}\sigma_1^2 & 0 & \cdots & 0\\ 0 & \sigma_2^2 & \cdots & 0\\ \vdots & \vdots & \ddots & 0\\ 0 & 0 & 0 & \sigma_n^2\end{bmatrix}$$

$$\sigma_1\sigma_2...\sigma_n=|K|^{\frac{1}{2}}$$

$$\frac{(x_1-\mu_1)^2}{\sigma_1^2}+\frac{(x_2-\mu_2)^2}{\sigma_2^2}+...+\frac{(x_n-\mu_n)^2}{\sigma_n^2}=(x-\mu)^T K^{-1}(x-\mu)$$

$$p(x)=(2x)^{-\frac{n}{2}}|K|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(x-\mu)^T K^{-1}(x-\mu)\right) \quad (3)$$

Where $\mu\in\mathbb{R}^n$ is the mean vector, and $K\in\mathbb{R}^{n\times n}$ is the covariance matrix. Since we assume that each dimension is directly independent of each other, K is a diagonal matrix. When variables of each imension are correlated, the form of the above equation is still the same, but in this case, the covariance matrix K is no longer a diagonal matrix and only has the properties of semi-positive definite and symmetric. This is also commonly abbreviated as

$$x\sim N(\mu,K)$$

Therefore,A mean and a variance determine a onedimensional Gaussian distribution, and a mean vector

and a covariance matrix determine a multidimensional Gaussian distribution

Gaussion Process

When we look at sampling from the perspective of function and understand that each sampling with infinite dimension is equivalent to sampling a function, the original probability density function is no longer a distribution of points, but a distribution of functions. This infinite element Gaussian distribution is called a Gaussian process. Gaussian processes are formally defined as: for all

$$x=(x_1,x_2,...,x_n).f(x)=\left[f(x_1),f(x_2),...,f(x_n)\right]$$

all obey the multi-element Gaussian distribution, Then *f* is said to be a Gaussian process, expressed as

$$f(x)\sim N(\mu(x),k(x,x))$$

The $\mu(x)\in\mathbb{R}^n\to\mathbb{R}^n$ indicates the Mean function and returns the mean of each dimension; The $\kappa(x,x)$: $\mathbb{R}^n\times\mathbb{R}^n\to\mathbb{R}^{n\times n}$ is a covariance function Covariance Function (also known as the Kernel). Function returns the ovariance matrix between the dimensions of two vectors. A Gaussian process is uniquely defined as a mean function and a covariance function, and subsets of the finite dimensions of a Gaussian process all obey a multivariate Gaussian distribution (for ease of understanding, we can imagine a binary Gaussian distribution in which each dimension obeys a Gaussian distribution).A Gaussian process can be determined by a mean function and a covariance function!

kernel Function

The kernel takes the vectors in the original space as input vectors and returns the function of the dot product

of the vectors in the feature space (converted data space, possibly higher dimensional) [15,16].

Definition 1. Kernel Function : X is subset of $\mathbb{R}^n$,∃ mapping $\phi:\phi:\mapsto\phi(\xi)\in$ H, H is Hilbert Space, for $\forall x,y\in$ X, there is $\kappa(x,y):(\phi(x),\phi(y))=\phi(x)^T\phi(y)$, the $\kappa(x,y)$ is Kernel Function

Linear kernel

Computes a covariance matrix based on the Linear kernel between inputs X_1 and X_2 :

$$k_{Linear}(X_1,X_2)=\upsilon X_1^T X_2. \quad (4)$$

where - v is a variance parameter.

RBF

Computes a covariance matrix based on the RBF (squared exponential) kernel between inputs X_1 and X_2 :

$$k_{RBF}(X_1,X_2)=\exp\left(-\frac{1}{2}(X_1-X_2)^T\Theta^{-2}(X_1-X_2)\right) \quad (5)$$

where Θ is a lengthscale parameter. See gpytorch. kernels. Kernel for descriptions of the lengthscale options.

Gaussion Process Regression

GPR

The formula of Gaussian process regression is derived as follows.

Gaussian process a priori said

$$p\left(f\left(x\right)\middle|x\right)=f\left(x\right)\mathrm{N}\left(\mu_f,K_{xx}\right)$$

This is a prior distribution representing the output y we expect to get after input x before looking at any data.

After that, we import some training data with input x and output $y=f\left(x\right)$.Next, we have some new input x^* and need to compute $y^*=f\left(x^*\right)$

If now we observe some data $\left(x^*,y^*\right)$ and assume that y^* and f(x) obey joint Gaussian distribution

$$\begin{bmatrix}f(x)\\y^*\end{bmatrix}\sim N\left(\begin{bmatrix}\mu_f\\\mu_y\end{bmatrix},\begin{bmatrix}k_{xx}&k_{xx^*}\\k_{xx^*}^T&k_{xx^*}\end{bmatrix}\right)\qquad(6)$$

Among them

$$K_{xx}=k\left(x,x\right),K_{xx^*}=k\left(x,x^*\right),K_{x^*x^*}=k\left(x^*,x^*\right)$$

Have x, y as the training data, x^* for the input data), now, the model is $p\left(Y^*\middle|X^*,X,Y\right)$, What we need is y^*

This becomes a given joint Gaussian distribution, finding the conditional probability $P\left(y^*\middle|x^*,x,y\right)$:

The above formula indicates that the distribution *f* of the function after given data (x*, y*) is still a Gaussian

process. The specific derivation is as follows:

Here are some definitions and theorems:

Definition 2. Set $U=\left(U_1,\ldots U_q\right)'$ for random vector, $U_1,\ldots U_q$,Uqare independent of each other and with

N(0,1) distribution; Let μ be *p* dimensional constant vector and *A* be $p\times q$ constant matrix, then the distribution of X = AU +μ is called p dimensional normal distribution or X is called *p* dimensional normal random vector. Write it as $X\sim N_p\left(\mu,AA'\right)$.

Say simply, by q a independent standard normal random variable of some linear group of the distribution of the random vector, is referred to as multiple normal distributed.

Theorem 1. Let $X\sim N_p\left(\mu,\Sigma\right),B$ is $s\times p$ constant matrix, *d* is sconstant vector, let $Z=B\mu+d$, then $Z\sim N_s\left(B\mu+d,B\Sigma B'\right)$ Proof. Prove that $\sum_d\geq 0,\Sigma$ can be decomposed into: $\Sigma=AA'$, is defined by 2.1 as $X=AU+\mu(1)\,A$ is $P\times q$ matrix),

Where $U=\left(U_1,\ldots,U_q\right)'$, and $U_1,\ldots,U_q$ is independent distribution with *N* (0, 1) again

$$Z=BX+d\overset{d}{=}B\left(AU=\mu\right)+d=BAU+\left(B\mu+d\right).$$

By definition, 1 $Z\sim N_8\left(B\mu+d,\left(BA\right)\left(BA\right)'\right)$,

$$Z\sim N_s\left(B\mu+d,B\Sigma B'\right).$$

Lemma 1.*Set* $X=\begin{bmatrix}X^{(1)}\\X^{(2)}\end{bmatrix}_{p-r}^{r}\sim N_p\left(\mu,\Sigma\right),where$

$$\mu=\begin{bmatrix}\mu^{(1)}\\\mu^{(2)}\end{bmatrix}_{p-r}^{r},\Sigma=\begin{bmatrix}\Sigma_{11}^{r}&\Sigma_{12}^{p-r}\\\Sigma_{21}&\Sigma_{22}\end{bmatrix}_{p-r}^{r},$$

$$Is\ X^{(1)}\sim N_r\left(\mu^{(1)},\Sigma_{11}\right),X^{(2)}\sim N_{p-r}\left(\mu^{(2)},\Sigma_{22}\right).$$

Theorem 2. *set* $X=\begin{bmatrix}X^{(1)}\\X^{(2)}\end{bmatrix}_{p-r}^{r}\sim N_p\left(\mu,\Sigma\right)\left(\Sigma>0\right),$

$X^{(2)}$ is given, the conditional distribution of $X^{(1)}$ is Among them

$$\left(X^{(1)}\middle|X^{(2)}\right)\sim N_r\left(\mu_{1.2},\Sigma_{11.2}\right),$$

$$\mu_{1.2}=\mu^{(1)}+\Sigma_{12}\Sigma_{22}^{-1}\left(x^{(2)}-\mu^{(2)}\right),$$

$$\Sigma_{11.2}=\Sigma_{11}-\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}.$$

Proof. Prove that for a nonsingular linear transformation, let

$$Z=\begin{bmatrix}Z^{(1)}\\Z^{(2)}\end{bmatrix}$$

$$=\begin{bmatrix}X^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}X^{(2)}\\X^{(2)}\end{bmatrix}$$

$$=\begin{bmatrix}I_r&-\Sigma_{12}\Sigma_{22}^{-1}\\0&I_{p-r}\end{bmatrix}\begin{bmatrix}X^{(1)}\\X^{(2)}\end{bmatrix}$$

$$=BX.$$

By 2.2 the property of 2 obviously has

$Z\sim N_p\left(\begin{bmatrix}\mu^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}\mu^{(2)}\\\mu^{(2)}\end{bmatrix},\begin{bmatrix}\Sigma_{11.2}&0\\0&\Sigma_{22}\end{bmatrix}\right)$and $D(Z)=\begin{bmatrix}\Sigma_{11.2}&0\\0&\Sigma_{22}\end{bmatrix}$ so $Z^{(1)}$and are independent. The joint density of Z is

$$g\left(z^{(1)},z^{(2)}\right)=g_1\left(z^{(1)}\right).g_2\left(z^{(2)}\right)=g_1\left(z^{(1)}\right).f_2\left(z^{(2)}\right).$$

and $Z^{(2)}=X^{(2)}$, so $g_2\left(z^{(2)}\right)=f_2\left(z^{(2)}\right)\left(f_2\left(\cdot\right)\right)$ denisty of $X^{(2)}$). because Z = BX, Using the integral transformation formula, the density function *f*(x) of *X* can be represented by g(z), i.e

$$f\left(x^{(1)},x^{(2)}\right)=g\left(Bx\right).J\left(z\to x\right)$$

$$=g_1\left(x^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}x^{(2)}\right).g_2\left(x^{(2)}\right).\left|\frac{\partial z'}{\partial x}\right|_+$$

$$=g_1\left(x^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}x^{(2)}\right).f_2\left(x^{(2)}\right),$$

The $\left|\frac{\partial z'}{\partial x}\right|_+=\left|B'\right|=1.$

$Z^{(1)}\sim N_r\left(\mu^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}\mu^{(2)},\Sigma_{11.2}\right)$, therefore

$$f_1\left(x^{(1)}\middle|x^{(2)}\right)=\frac{f\left(x^{(1)},x^{(2)}\right)}{f_2\left(x^{(2)}\right)}=g_1\left(x^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}x^{(2)}\right)$$

$$=\frac{1}{\left(2\pi\right)^{r/2}\left|\Sigma_{11.2}\right|^{1/2}}.$$

$$\exp\left[-\frac{1}{2}\left(x^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}x^{(2)}-\left(\mu^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}\mu^{(2)}\right)\right)'\right.$$

$$\left.\Sigma_{11.2}^{-1}\left(x^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}x^{(2)}-\left(\mu^{(1)}-\Sigma_{12}\Sigma_{22}^{-1}\mu^{(2)}\right)\right)\right]$$

$$=\frac{1}{\left(2\pi\right)^{r/2}\left|\Sigma_{11.2}\right|^{1/2}}.\exp\left[-\frac{1}{2}\left(x^{(1)}-\mu_{1.2}\right)'\Sigma_{11.2}^{-1}\left(x^{(1)}-\mu_{1.2}\right)\right],$$

And

$$\Sigma_{11.2}=\Sigma_{11}-\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}.$$

So theorem 3.4 tells us that

$$p\left(y^*\middle|X^*,X,Y\right)=N\left(y^*\middle|\mu^*,\Sigma^*\right)$$

$$\mu^*=\mu_y+k_{xx*}^Tk_{xx*}^{-1}\left(X-\mu f\right)$$

$$\Sigma^*=K_{yy}-k_{xx*}^Tk_{xx*}^{-1}k_{xx*}$$

$$y^*=f\left(x*\right)=p\left(y^*\middle|X^*,X,Y\right)$$

$$\sim N\left(k_{xx*}^Tk_{xx*}^{-1}\left(x-\mu_f\right)+\mu_y,k_{xx}-k_{xx*}^Tk_{xx*}^{-1}k_{xx*}\right)$$

This is the posterior distribution of y_* calculated based on the prior distribution and the observations. In the case of GPR, this formula helps us get the predicted value, and in most cases u = 0

Marginal maximum likelihood (MLL)

These are modules to compute the marginal log likelihood (MLL) of the GP model when applied to data [17,18,19]. I.e., given a $GP\ \ f\sim G\left(\mu,K\right)$, and data X, y, these modules compute

$$\iota=pf\left(y\middle|x\right)=\int p\left(y\middle|f\left(X\right)\right)p\left(f\left(X\right)\middle|X\right)df \tag{8}$$

This is computed exactly when the GP inference is computed exactly . It is approximated for GP models that

use approximate inference.

These models are typically used as the ”loss” functions for GP models (though note that the output of these

functions must be negated for optimization).

From 2.2 ,we know that:

$$p\left(f\left(x\right)\middle|x\right)=$$

$$\left(2\pi\right)^{-\frac{n}{2}}\left(\left|K_{xx}\right|\right)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}f\left(x\right)^T\ K_{xx}^{-1}f\left(x\right)\right) \tag{9}$$

$$p\left(y\middle|f\left(x\right)\right)=\left(2\pi\right)^{-\frac{n}{2}}\left(\left|K_{xx}\right|\right)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}y^TK_{xx}^{-1}f\left(x\right)\right) \tag{10}$$

So:

$$pf\left(y\middle|X\right)=\int\left(2\pi\right)^{-\frac{n}{2}}\left(\left|K_{xx}\right|\right)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}y^TK_{xx}^{-1}y\right)\bullet$$

$$\left(2\pi\right)^{-\frac{n}{2}}\left(\left|K_{xx}\right|\right)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}f\left(X\right)^T\ K_{xx}^{-1}f\left(x\right)\right)df\left(x\right) \tag{11}$$

we know

$$\int\left(2\pi\right)^{-\frac{n}{2}}\left(\left|K_{xx}\right|\right)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}f\left(X\right)^T\ K_{xx}^{-1}f\left(x\right)\right)df\left(x\right)=1$$

$$pf\left(y\middle|X\right)=\left(2\pi\right)^{-\frac{n}{2}}\left(\left|K_{xx}\right|\right)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}y^TK_{xx}^{-1}y\right) \tag{12}$$

So, the loglikehood is

$$\log pf\left(y\middle|X\right)=-\frac{1}{2}y^TK_{xx}^{-1}y-\frac{1}{2}\log\left|K_{xx}\right|-\frac{n}{2}\log2\pi \tag{13}$$

Therefore, the optimized parameter model can be obtained [20] (take RBF kernel function as an example).

$$\frac{\partial}{\partial\Theta}\log p\left(y\middle|X,\Theta\right)$$

$$=\frac{1}{2}y^TK_{xx}^{-1}\frac{\partial K_{xx}}{\partial\Theta}K_{xx}^{-1}y-\frac{1}{2}tr\left(K_{xx}^{-1}\frac{\partial K_{xx}}{\partial\Theta}\right)$$

$$= \frac{1}{2} tr\left(\left(\alpha\alpha^T - K_{xx}^{-1} \right) \frac{\partial K_{xx}}{\partial \Theta} \right)$$

$$where \ \ \alpha = K_{xx}^{-1} y \qquad (14)$$

The random gradient descent algorithm is carried out on the parameters Θ [21,22]

$$\Theta = \Theta - \gamma \frac{\partial}{\partial \Theta} \log p\left(y \middle| X, \Theta \right) \qquad (15)$$

Experimental Evaluation

Dataset

The data set contains information about housing prices in Boston, Massachusetts collected by the U.S. Census Bureau. The data set contains 506 samples.It contains 13 characteristic variables and a target value: the average house price.Below is a brief introduction to the housing price data setGaussian process regression was performed on the above Boston housing price data set, and the dataset was randomly divided into a training set and a test set in a 1:1. We assume that the data of the training set obey the Gaussian process, so as to predict the average housing price (Figure 1).

CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2
0.02985	0	2.18	0	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43	22.9
0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.9	19.15	27.1
0.21124	12.5	7.87	0	0.524	5.631	100	6.0821	5	311	15.2	386.63	29.93	16.5
0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.1	18.9
0.22489	12.5	7.87	0	0.524	6.377	94.3	6.3467	5	311	15.2	392.52	20.45	15
0.11747	12.5	7.87	0	0.524	6.009	82.9	6.2267	5	311	15.2	396.9	13.27	18.9
0.09378	12.5	7.87	0	0.524	5.889	39	5.4509	5	311	15.2	390.5	15.71	21.7
0.62976	0	8.14	0	0.538	5.949	61.8	4.7075	4	307	21	396.9	8.26	20.4
0.63796	0	8.14	0	0.538	6.096	84.5	4.4619	4	307	21	380.02	10.26	18.2
0.62739	0	8.14	0	0.538	5.834	56.5	4.4986	4	307	21	395.62	8.47	19.9
1.05393	0	8.14	0	0.538	5.935	29.3	4.4986	4	307	21	386.85	6.58	23.1
0.7842	0	8.14	0	0.538	5.99	81.7	4.2579	4	307	21	386.75	14.67	17.5

Figure 1: Boston dataset.

GPR Algorithm

According to the Dataset, we can get the input training data as follows:

$$X = \left\{ X_{1,} X_{2,\ldots} X_{253} \right\}^T$$

$$X_i = \left\{ X_i^1, X_i^2, \ldots X_i^{13} \right\}, \ i = 1, 2, \ldots\ldots 253$$

The target data is :

$$Y = \left\{ y_{1,} y_{2,\ldots} y_{253} \right\}^T$$

The predict data is:

$$\hat{Y} = f(x) \in, \in \sim N\left(0, \sigma^2\right)$$

The training set was trained for 5 epochs, every epoch was iterated 1000 times, and the MLL function was used to calculate the loss of the predicted value and the target value. Fig 2 shows the loss function image of the model against the training data set after Gaussian process regression uses linear kernel function and RBF kernel function respectively.

As can be seen from the figure, the loss function value of the model using linear kernel function is lower than that using RBF kernel function in the same training batch, and the rate of decline is faster. When the loss function converges, the loss value of the model using linear kernel function is lower than that using RBF kernel function. Therefore, from the convergence of loss function, the model using linear kernel function performs better. Table 1 and Table 2 show the partial parameter values of the model in the training process (Table 1,2 & Figure 2).

Table 1: Parameter.

Linear	RBF	noise
0.693147	0.693147	0.693247
0.96436	0.975312	1.300361
1.238469	1.323533	2.016961
1.471993	1.7366	2.702634
1.648751	2.206498	3.295374

Table 2: Parameter.

Linear	RBF	noise
1.770648	2.720727	3.787125
1.846764	3.267597	4.189744
1.887529	3.83662	4.519375
1.902391	4.417646	4.791023
1.899158	5.000462	5.017116

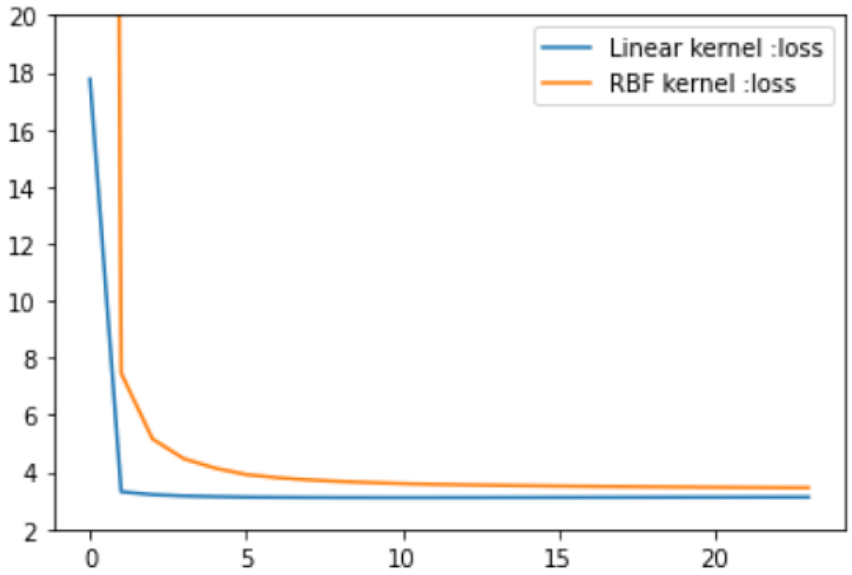
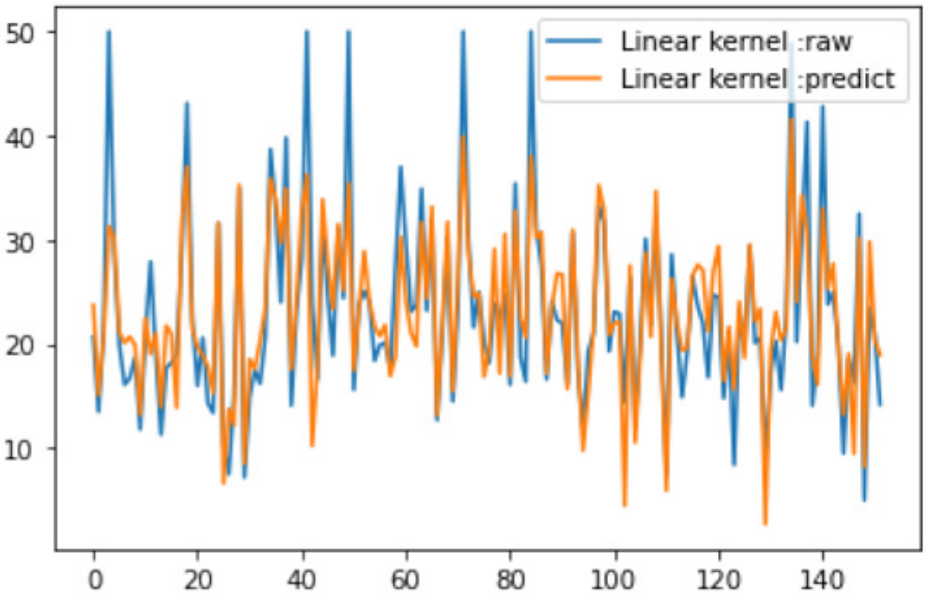
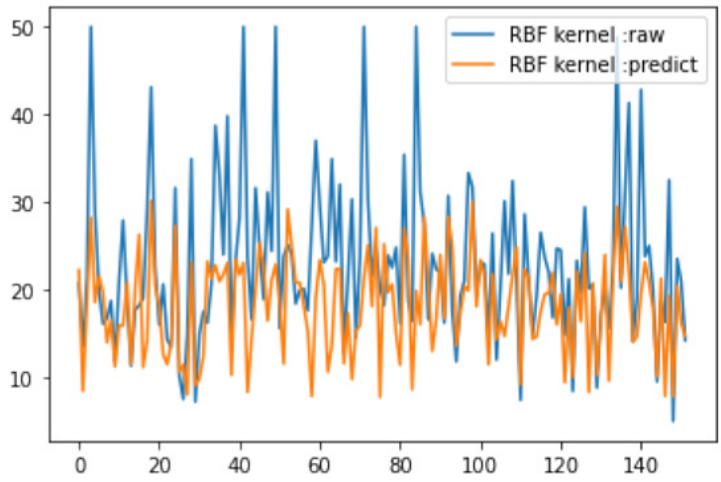


Figure 2: Trainloss.

Fig 3 and Fig 4 respectively show the fitting between the predicted value and the real value of the Gaussian process regression model under the two kernel functions. The comparison between FIG. 3 and FIG. 4 shows that the predicted value of the model under the linear kernel function better fits the original real value. Therefore, in terms of the fitting of the predicted value, the model using linear kernel function is better than that using RBF kernel function.





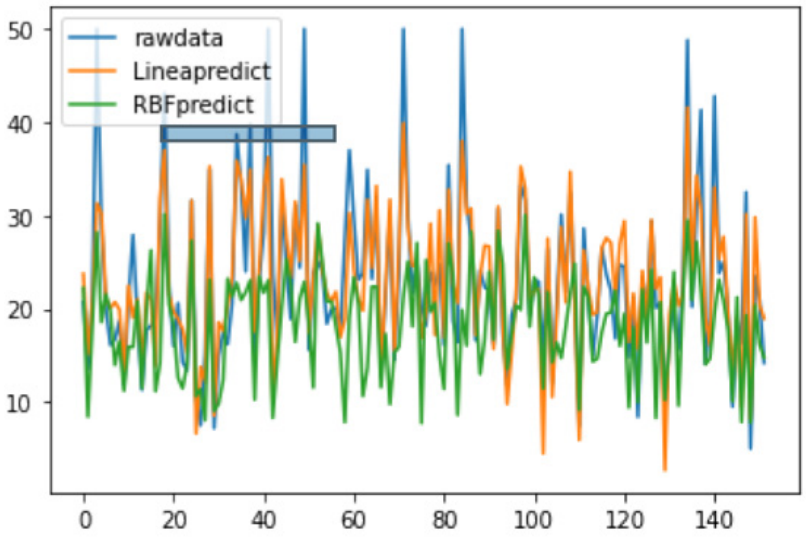
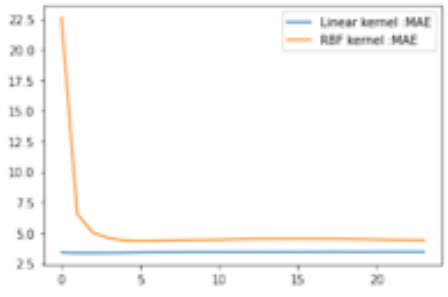


Figure 3: Comparison of predicted values.



step	Linear	RBF
0	3.3822	22.62827
200	3.3445	6.5431
400	3.3395	5.0014
600	3.3495	4.5627
800	3.3676	4.3733

Figure 4: MAE.

Metric

The evaluation criteria of three regression models are used in this paper. Mean Absolute Error(MAE): Is the average value of absolute error, which can better reflect the actual situation of predicted error.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|\hat{y}_i - y_i\right| \qquad (16)$$

Mean Square Error(MSE):Is the square of the difference between the predicted value and the true value, and then the average of the sum, generally used to detect the deviation between the predicted value and the true value of the model.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{y}_i - y_i\right)^2 \qquad (17)$$

R2 squared:Coefficient of determination. Reflects the accuracy of model fitting data, generally R^2 range is 0 to 1. The closer the value is to 1, it indicates that the variable of the equation has a stronger ability to explain y, and this model also fits the data well (Figure 3).

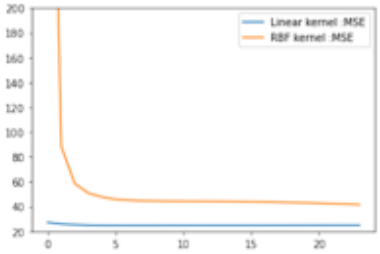
$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2} \qquad (18)$$

Fig 5 shows the MAE value of the Gaussian process regression model on the test set under the two kernel functions. MAE values for partial iterations are listed in the table. As can be seen from Figure 5, MAE under the linear kernel model is much smaller than that under the RBF kernel model. It can be seen that, from the perspective of MAE evaluation criteria, the model using linear kernel function has better performance than that using RBF kernel function (Figure 4).

The value of MSE is shown in Figure 6. As can be seen from the figure, the mean square error using the linear

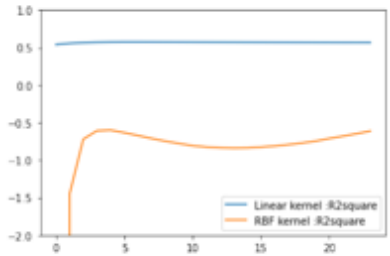
kernel model is smaller, which means that the predicted value using the linear kernel model has less deviation from the real value. Therefore, from the point of view of mean square error, it can be seen that the advantage of using linear kernel function is more obvious (Figure 5,6).

Figure 6 shows the value of R2square. It can be seen from the figure that the R2square value of the model using linear kernel function is closer to 1 than that of the model using RBF kernel function. The closer it is to 1, the stronger the interpretation ability of the model to the real value is, thus indicating the better fitting degree of the model to the data. So in this respect you can also see the advantage of using linear kernel functions.



step	Linear	RBF
0	27.2439	599.0617
200	26.0481	88.4776
400	25.3891	58.5171
600	25.0532	50.8863
800	24.9135	47.7

Figure 5: MSE.



step	Linear	RBF
0	0.5406	-657.443
200	0.5559	-1.4479
400	0.5650	-0.7212
600	0.5704	-0.6078
800	0.5729	-0.6006

Figure 6: R2square.

Gaussian process regression can effectively simulate data, but the choice of kernel function has a great impact on the simulation results of data, so how to choose a suitable kernel function when we use Gaussian regression model becomes very important.In this paper, the linear kernel and RBF kernel are compared in the Gaussian process regression model, and it is concluded that the linear kernel is better than RBF kernel in this data set. Therefore, we know that the selection of kernel function should not only consider the efficiency of its own model, but also consider the influence of data set itself. Therefore, in the future work, I hope to find a better and faster method to select the kernel function more suitable for the model, so as to achieve a better prediction effect on the data.

Acknowledgement

"Supported by: Fundamental Research Program of Shanxi Province Fund number: 202103021223379" to the article.

Conflict of Interest

No Conflict of interest.

References

1. JR Quinlan (1993) Program for machine learning.
2. C Andrieu, ND Freitas, A Doucet, MI Jordan (2003) An introduction to mcmc for machine learning. Machine Learning 50(1): 5-43.
3. D Nguyen-Tuong, J Peters (2008) Local gaussian process regression for real-time model-based robot control. In Intelligent Robots and Systems.
4. A Girard, CE Rasmussen, R Murray-Smith (2002) Gaussian process priors with uncertain inputs - multiple-step-ahead prediction. School of Computing Science.
5. AJ Smola, P Bartlett (2000) Sparse greedy gaussian process regression. In Neural Information Processing Systems.
6. H Owhadi, JL Akian, L Bonnet (2022) Learning "best" kernels from data in gaussian process regression. with application to aerodynamics. Journal of Computational Physics.
7. CE Rasmussen, Cki Williams (2006) Gaussian processes for machine learning (adaptive computation and machine learning). The MIT Press.
8. Joaquin, onero Candela, CE Rasmussen (2005) A unifying view of sparse approximate gaussian process regression. The Journal of Machine Learning Research 6: 1939-1959.
9. M Seeger, Cki Williams, ND Lawrence (2003) Fast forward selection to speed up sparse gaussian process regression. In International Conference on Artificial Intelligence and Statistics: 254-261.
10. B Lu, D Gu, H Hu, K Mcdonald-Maier (2012) Sparse gaussian process for spatial function estimation with mobile sensor networks. In Emerging Security Technologies (EST). Third International Conference on.
11. O Stegle, SV Fallert, DJC Mackay, S Brage (2008) Gaussian process robust regression for noisy heart rate data. IEEE transactions on biomedical engineering 55(9): 2143-2151.
12. Govind S Mudholkar, Ziji Yu, Saria S Awadalla (2015) The mode-centric m-gaussian distribution: A model for right skewed data. Statistics & Probability Letters 107: 1-10.
13. Stein, M Charles (1981) Estimation of the meaning of a multivariate normal distribution. Annals of Statistics 9(6): 1135-1151.
14. C Stein (1956) Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. University of California Press: 197-206.
15. S Bergman (1950) The kernel function and conformal mapping. American Mathematical Society.
16. P Li, S Xu (2005) Support vector machine and kernel function characteristic analysis in pattern recognition. Computer Engineering and Design.
17. BM Aitkin (1981) Marginal maximum likelihood estimation of item parameters: Application of an em algorithm. Psychometrika 46: 443–459.
18. JL Foulley, MS Cristobal, D Gianola, S Im (1992) Marginal likelihood and bayesian approaches to the analysis of heterogeneous residual variances in mixed linear gaussian models. Computational Statistics & Data Analysis 13(3): 291-305.
19. P Kontkanen, P Myllymaki, H Tirri (2013) Classifier learning with supervised marginal likelihood. Morgan Kaufmann Publishers Inc: 277-284.
20. Y Bazi, F Melgani (2009) Gaussian process approach to remote sensing image classification. IEEE Transactions on Geoscience and Remote Sensing 48(1): 186-197.
21. YX Hao, YR Sun, SO Sciences (2018) Knearest neighbor matrix factorization for recommender systems. Journal of Chinese Computer Systems.
22. YX Hao, J H Shi (2022) Jointly recommendation algorithm of knn matrix factorization with weights. Journal of Electrical Engineering & Technology 17: 3507-3514.