Gaussian Process Regression with Local Explanation

Yuya Yoshikawa
STAIR Lab.
Chiba Institute of Technology
Chiba, Japan
yoshikawa@stair.center

Tomoharu Iwata
NTT Communication Science Laboratories
Kyoto, Japan
tomoharu.iwata.gy@hco.ntt.co.jp

Abstract

Gaussian process regression (GPR) is a fundamental model used in machine learning. Owing to its accurate prediction with uncertainty and versatility in handling various data structures via kernels, GPR has been successfully used in various applications. However, in GPR, how the features of an input contribute to its prediction cannot be interpreted. Herein, we propose GPR with local explanation, which reveals the feature contributions to the prediction of each sample, while maintaining the predictive performance of GPR. In the proposed model, both the prediction and explanation for each sample are performed using an easy-to-interpret locally linear model. The weight vector of the locally linear model is assumed to be generated from multivariate Gaussian process priors. The hyperparameters of the proposed models are estimated by maximizing the marginal likelihood. For a new test sample, the proposed model can predict the values of its target variable and weight vector, as well as their uncertainties, in a closed form. Experimental results on various benchmark datasets verify that the proposed model can achieve predictive performance comparable to those of GPR and superior to that of existing interpretable models, and can achieve higher interpretability than them, both quantitatively and qualitatively.

1 Introduction

Gaussian processes (GPs) have been well studied for constructing probabilistic models as priors of nonlinear functions in the machine learning (ML) community. They have demonstrated great success in various problem settings, such as regression [1, 2], classification [1, 3], time-series forecasting [4], and black-box optimization [5]. A fundamental model on GPs is Gaussian process regression (GPR) [1]; owing to its high predictive performances and versatility in using various data structures via kernels, it has been used in not only the ML community, but also in various other research areas, such as finance [6], geostatistics [7], material science [8] and medical science [9, 10].

GPR is defined on an infinite-dimensional feature space via kernel functions. Therefore, it requires the values of the kernels defined on two samples, i.e., a Gram matrix as an input, rather than the samples themselves. Owing to the nonlinearity of the kernel, GPR enables nonlinear predictions; however, it cannot explain what features contribute to the predictions, like linear regression models. Therefore, users of GPR cannot judge whether the predictions are reasonable and performed by fair decision.

For the interpretability of ML, several methodologies that explain the features that contribute to the outputs of prediction models, including GPR, have been proposed; in this case, the prediction models are regarded as black boxes [11, 12]. Their representative methods are local interpretable model-agnostic explanations (LIME) [13] and Shapley additive explanations (SHAP) [14], which approximate the prediction for each test sample by a locally linear explanation model. By observing the weights of the learned explanation model, the feature contributions to the prediction can be

Preprint. Under review.

understood. However, some limitations exist in these methods. First, because the forms of the prediction and explanation models differ, it is unsure whether the estimated feature contributions reflect those of the prediction model. Furthermore, because the explanation model is learned on each test sample, it may not obtain consistent explanations on similar samples.

To overcome the aforementioned limitations, we propose a novel framework for GP-based regression models, *Gaussian process regression with local explanation*, called *GPX*, which reveals the feature contributions to the prediction for each sample, while maintaining the predictive performance of GPR. In GPX, both the prediction and explanation for each sample are performed using an easy-to-interpret locally linear model. Therefore, no gap exists between the prediction and explanation. The weight vector of the locally linear model is assumed to be generated from multivariate GP priors [16]. As the multivariate GP priors have a covariance function defined as kernels on the samples, GPX ensures that similar samples have similar weights. The hyperparameters of GPX are estimated by maximizing the marginal likelihood, in which the weight vectors for all the training samples are integrated out. For a test sample, the predictions with their uncertainties of the target variable and weight vector are obtained by computing their predictive distributions. The explanation for the predicted target variable is provided using the estimated weight vector with uncertainty, as shown in Figure 1. Depicting the explanation with uncertainty helps users of GPX judge the reasonability of the predicted weights.

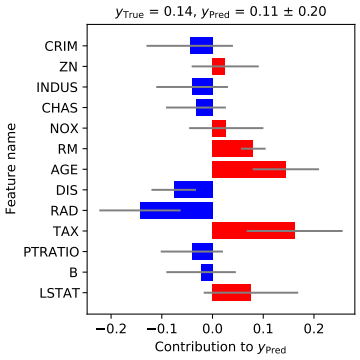


Figure 1: Example of explanation for prediction by GPX for a sample on the Boston housing dataset [15]. We provide further examples and feature description in Appendix A of the supplementary material.

In experiments, we evaluated GPX both qualitatively and quantitatively in terms of predictive performance and interpretability on various benchmark datasets. The experimental results show that 1) GPX can achieve predictive errors comparable to GPR and lower errors compared with existing interpretable methods, 2) it can outperform model-agnostic interpretable methods and locally linear methods in terms of three interpretability measurements, and 3) the feature contributions produced by GPX are appropriate.

2 Related Work

Linear regression models are simple types of interpretable models. A number of studies introducing various regularizations have been conducted to produce methods such as the ridge regression [17] and lasso [18] methods. These models have *global* weight vectors that are shared across all samples. In kernel methods, Automatic Relevance Determination (ARD), which considers the global relevance of each feature contained in kernel functions, is adopted [19, 20]. The above approaches have a significant issue in explainability, i.e., when a feature has inconsistent weight/relevance, for example, the weight/relevance of a feature is changed by the influence of another feature, global weights/relevances cannot be coped with it.

On the other hand, some *locally* linear models for regression have been proposed, such as the network lasso [21] and localized lasso [22], which have a weight vector for each sample. Therefore, these methods can avoid the drawbacks of globally linear models. To receive the benefit, we focus on generating predictions with explanations using locally linear models. In the network and localized lasso, the weights of locally linear models are estimated via optimization with network-based regularization, where the network must be defined on samples in advance. If the network is not provided, as assumed in standard regression problems, we can construct a k-nearest-neighbor graph of samples to create a network, where k is a hyperparameter that must be optimized via cross validation. Meanwhile, GPX can estimate weights and their uncertainties without constructing graphs by assuming that weights are determined by functions generated from GPs.

With regard to research on deep neural networks (DNNs), a number of studies have been conducted on making predictions generated by DNNs interpretable [23, 24, 25]. Some of these studies have developed methods that make interpretable predictions by generating locally linear models for each

sample using DNNs [26, 27, 28]. These concepts inspired our study, but we formalize our model without DNNs. To the best of our knowledge, our study is the first to develop a GP-based regression model with local explanations. Unlike DNN-based locally linear models, GPX is beneficial for tasks in which GPR can be used advantageously, such as Bayesian optimization [5, 29].

3 Proposed Model

In this section, we describe the proposed model, i.e., *Gaussian process regression with local explanation*, called *GPX*.

We consider a scalar-valued regression problem. Suppose that training data $\mathcal{D} = \{(\boldsymbol{x}_i, y_i, \boldsymbol{z}_i)\}_{i=1}^n$ containing n samples is provided. $\boldsymbol{x}_i \in \mathcal{X}$ is an original input representing the ith sample, where $\mathcal{X}$ is an original input space. Although a typical representation for $\boldsymbol{x}_i$ is a vector, it can be any data representation on which kernel functions are defined, such as graphs [30] and sets [31, 32]. $y_i \in \mathbb{R}$ is a target variable for the sample. $\boldsymbol{z}_i \in \mathbb{R}^d$ is a d-dimensional vector of simplified representation for $\boldsymbol{x}_i$. Because GPX explains the prediction via a simplified representation, the meaning of each dimension of $\boldsymbol{z}_i$ should be easily understood by humans, e.g., tabular data and bag-of-words representation for text. $\boldsymbol{z}_i$ is an optional input; therefore, if $\boldsymbol{x}_i$ can be used as a simplified representation, one can define $\boldsymbol{z}_i = \boldsymbol{x}_i$. Let us denote $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^n$, $\boldsymbol{y} = (y_i)_{i=1}^n \in \mathbb{R}^n$ and $\boldsymbol{Z} = (\boldsymbol{z}_i)_{i=1}^n \in \mathbb{R}^{n \times d}$.

In GPX, both the prediction of target variables $\boldsymbol{y}$ and their explanations are performed via easy-to-interpret locally linear models, i.e., target variable y_i for the ith sample is assumed to be obtained using locally linear function $f_i : \mathbb{R}^d \to \mathbb{R}$, defined as follows:

$$f_i(\boldsymbol{z}_i) = \boldsymbol{w}_i^\top \boldsymbol{z}_i + \epsilon_{\mathrm{y}}, \tag{1}$$

where $\boldsymbol{w}_i \in \mathbb{R}^d$ is a d-dimensional weight vector for the ith sample, and $\epsilon_{\mathrm{y}} \sim \mathcal{N}(0, \sigma_{\mathrm{y}}^2)$ is a Gaussian noise with variance $\sigma_{\mathrm{y}}^2 > 0$. Here, the explanation for the ith sample is obtained using either weight vector $\boldsymbol{w}_i$ or feature contributions $\boldsymbol{\phi}_i = (w_{il} z_{il})_{l=1}^d$.

Estimating $\boldsymbol{W} = (\boldsymbol{w}_i)_{i=1}^n \in \mathbb{R}^{n \times d}$ without any constraints is an ill-posed problem because the number of free parameters in $\boldsymbol{W}$, nd is larger than that of target variable n. To avoid this problem in GPX, we assume that functions determining $\boldsymbol{W}$ are generated from a multivariate GP. More specifically, weight vector $\boldsymbol{w}_i$ for the ith sample is obtained as follows:

$$\boldsymbol{w}_i = \boldsymbol{g}(\boldsymbol{x}_i) + \boldsymbol{\epsilon}_{\mathrm{w}}, \tag{2}$$

where $\boldsymbol{\epsilon}_{\mathrm{w}} \sim \mathcal{N}(\boldsymbol{0}, \sigma_{\mathrm{w}}^2 \boldsymbol{I}_d)$ is a d-dimensional Gaussian noise with variance $\sigma_{\mathrm{w}}^2 > 0$, and $\boldsymbol{I}_d$ is an identity matrix of order d. Here, vector-valued function $\boldsymbol{g} : \mathcal{X} \to \mathbb{R}^d$ is a function that determines the weight vector for each sample, and each element of $\boldsymbol{g}$ is generated from a univariate GP independently, as follows:

$$\boldsymbol{g}(\boldsymbol{x}) = (\, g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \cdots, g_d(\boldsymbol{x}) \,)^\top, \quad g_l(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k_\theta(\boldsymbol{x}, \boldsymbol{x}')). \tag{3}$$

$m(\boldsymbol{x})$ is the mean function, and $k_\theta(\boldsymbol{x}, \boldsymbol{x}')$ is the covariance function with set of parameters θ. We use zero mean function for $m(\boldsymbol{x})$ as standard regularizers such as ℓ_1 and ℓ_2 assume. Covariance function $k_\theta(\boldsymbol{x}, \boldsymbol{x}')$ is a kernel function defined on two inputs $\boldsymbol{x}$, $\boldsymbol{x}' \in \mathcal{X}$. For example, one can use a scaled RBF kernel with parameters $\theta = \{\theta_1, \theta_2\}$ as the kernel function when $\boldsymbol{x}, \boldsymbol{x}'$ are vectors, defined as follows:

$$k_\theta(\boldsymbol{x}, \boldsymbol{x}') = \theta_1 \exp\left(-\frac{1}{\theta_2} \|\boldsymbol{x} - \boldsymbol{x}'\|_2^2\right), \quad \text{where} \quad \theta_1, \theta_2 > 0. \tag{4}$$

By using $\boldsymbol{g}$ generated as such, GPX ensures that two similar samples, i.e., those having a large kernel value, have similar weight vectors.

We let $\boldsymbol{G} = (\boldsymbol{g}(\boldsymbol{x}_i)^\top)_{i=1}^n \in \mathbb{R}^{n \times d}$. Based on the generative process above, the joint distribution of GPX is written as follows:

$$p(\boldsymbol{y}, \boldsymbol{W}, \boldsymbol{G} \mid \boldsymbol{X}, \boldsymbol{Z}) = p(\boldsymbol{G} \mid \boldsymbol{X}) \prod_{i=1}^n p(y_i \mid \boldsymbol{w}_i, \boldsymbol{z}_i) p(\boldsymbol{w}_i \mid \boldsymbol{G}_{i,\cdot}), \tag{5}$$

$$p(y_i \mid \boldsymbol{w}_i, \boldsymbol{z}_i) = \mathcal{N}(y_i \mid \boldsymbol{w}_i^\top \boldsymbol{z}_i, \sigma_{\mathrm{y}}^2), \quad p(\boldsymbol{w}_i \mid \boldsymbol{G}_{i,\cdot}) = \mathcal{N}(\boldsymbol{w}_i \mid \boldsymbol{g}(\boldsymbol{x}_i), \sigma_{\mathrm{w}}^2 \boldsymbol{I}_d), \tag{6}$$

$$p(\boldsymbol{G} \mid \boldsymbol{X}) = \prod_{l=1}^d \mathcal{N}\left(\boldsymbol{G}_{\cdot,l} \mid \boldsymbol{0}, \boldsymbol{K}\right), \tag{7}$$

where $\boldsymbol{G}_{i,\cdot}$ and $\boldsymbol{G}_{\cdot,l}$ denote the ith row and lth column vectors of $\boldsymbol{G}$, respectively, and $\boldsymbol{K} = (k_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j=1}^n$ is a Gram matrix of order n, which is identical to the requirement in GPR.

4 Training and Prediction

In this section, we describe the derivation of the marginal likelihood of GPX, the hyperparameter estimation for GPX, and the derivation of the predictive distributions of target variables and weight vectors for test samples.

Marginal likelihood. To ease the derivation of the marginal likelihood, we first modified the formulation of the joint distribution (5), while maintaining its mathematical meanings, as follows:

$$p(\boldsymbol{y}, \boldsymbol{W}, \boldsymbol{G} \mid \boldsymbol{X}, \boldsymbol{Z}) = \mathcal{N}(\mathsf{vec}(\boldsymbol{G}) \mid \boldsymbol{0}, \bar{\boldsymbol{K}})\, \mathcal{N}(\mathsf{vec}(\boldsymbol{W}) \mid \mathsf{vec}(\boldsymbol{G}), \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd})\, \mathcal{N}(\boldsymbol{y} \mid \bar{\boldsymbol{Z}}\mathsf{vec}(\boldsymbol{W}), \sigma_{\mathrm{y}}^2 \boldsymbol{I}_{nd}), \tag{8}$$

where $\bar{\boldsymbol{K}}$ is a block diagonal matrix of order nd whose block is $\boldsymbol{K}$, and $\mathsf{vec}(\cdot)$ is a function that flattens the input matrix in a column-major order. Here, $\bar{\boldsymbol{Z}} = (\mathsf{diag}(\boldsymbol{Z}_{\cdot,1}), \mathsf{diag}(\boldsymbol{Z}_{\cdot,2}), \cdots, \mathsf{diag}(\boldsymbol{Z}_{\cdot,d})) \in \mathbb{R}^{n \times nd}$, where $\mathsf{diag}(\cdot)$ is a diagonal matrix whose diagonal elements possess the values of the input vector. In (5), d functions that output n-dimensional column vectors in $\boldsymbol{W}$ are generated from GPs; however, in (8), it is rewritten such that a single function that outputs an nd-dimensional flatten vector $\mathsf{vec}(\boldsymbol{W})$ is generated from a single GP. Consequently, the likelihood of target variables $\boldsymbol{y}$ can be rewritten as a single multivariate normal distribution.

Subsequently, we derived the marginal likelihood by integrating out $\boldsymbol{G}$ and $\boldsymbol{W}$ in (8). Owing to the property of normal distributions, it can be obtained analytically, as follows:

$$p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{Z}) = \iint p(\boldsymbol{y}, \boldsymbol{W}, \boldsymbol{G} \mid \boldsymbol{X}, \boldsymbol{Z}) d\boldsymbol{W} d\boldsymbol{G} = \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{0}, \boldsymbol{C}\right), \tag{9}$$

where $\boldsymbol{C} = \sigma_{\mathrm{y}}^2 \boldsymbol{I}_n + \bar{\boldsymbol{Z}} \left(\bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}\right) \bar{\boldsymbol{Z}}^\top = \sigma_{\mathrm{y}}^2 \boldsymbol{I}_n + (\boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n) \odot \boldsymbol{Z}\boldsymbol{Z}^\top$.

Hyperparameter estimation. If $k_\theta(\boldsymbol{x}, \boldsymbol{x}')$ is differentiable with respect to θ, all the hyperparameters, i.e., θ, σ_{w}, and σ_{y}, can be estimated by maximizing the logarithm of the marginal likelihood with respect to them for the training data using gradient-based optimization methods, e.g., L-BFGS [33].

Predictive distributions. For a new test sample $(\boldsymbol{x}_*, \boldsymbol{z}_*)$, our goal is to infer the predictive distributions of target variable y_* and weight vector $\boldsymbol{w}_*$. First, the predictive distribution of y_* is obtained similarly as in the standard GPR, as follows:

$$p(y_* \mid \boldsymbol{x}_*, \boldsymbol{z}_*, \mathcal{D}) = \mathcal{N}\left(y_* \mid \boldsymbol{c}_*^\top \boldsymbol{C}^{-1} \boldsymbol{y}, c_{**} - \boldsymbol{c}_*^\top \boldsymbol{C}^{-1} \boldsymbol{c}_*\right), \tag{10}$$

where $\boldsymbol{c}_* = \left(k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_i) \boldsymbol{z}_*^\top \boldsymbol{z}_i\right)_{i=1}^n \in \mathbb{R}^n$ and $c_{**} = \sigma_{\mathrm{y}}^2 + \left(k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma_{\mathrm{w}}^2\right) \boldsymbol{z}_*^\top \boldsymbol{z}_* \in \mathbb{R}$.

Second, the predictive distribution of $\boldsymbol{w}_*$ is obtained by solving the following integral:

$$p(\boldsymbol{w}_* \mid \boldsymbol{x}_*, \boldsymbol{z}_*, \mathcal{D}) = \int p(\boldsymbol{w}_* \mid \boldsymbol{W}, \boldsymbol{X}, \boldsymbol{x}_*) p(\boldsymbol{W} \mid \mathcal{D}) d\boldsymbol{W}, \tag{11}$$

$$p(\boldsymbol{w}_* \mid \boldsymbol{W}, \boldsymbol{X}, \boldsymbol{x}_*) = \mathcal{N}(\boldsymbol{w}_* \mid \boldsymbol{A}\mathsf{vec}(\boldsymbol{W}), \bar{\boldsymbol{c}}_{**} - \boldsymbol{A}\bar{\boldsymbol{k}}_*), \quad p(\boldsymbol{W} \mid \mathcal{D}) = \mathcal{N}(\mathsf{vec}(\boldsymbol{W}) \mid \sigma_{\mathrm{y}}^{-2} \boldsymbol{S} \bar{\boldsymbol{Z}}^\top \boldsymbol{y}, \boldsymbol{S}), \tag{12}$$

where we define $\boldsymbol{A} = \bar{\boldsymbol{k}}_*^\top \left(\bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}\right)^{-1}$, $\boldsymbol{S} = \boldsymbol{L} - \boldsymbol{L}\bar{\boldsymbol{Z}}^\top \boldsymbol{C}^{-1} \bar{\boldsymbol{Z}} \boldsymbol{L}$, $\boldsymbol{L} = \bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}$, and $\bar{\boldsymbol{c}}_{**} = \left(k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma_{\mathrm{w}}^2\right) \boldsymbol{I}_d$. $\bar{\boldsymbol{k}}_*$ is an nd-by-d block matrix, where each block is an n-by-1 matrix, and (l, l)-block of the block matrix is $(k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_i))_{i=1}^n$ for $l = 1, 2, \cdots, d$, and the other blocks are zero matrices. Solving the integral analytically according to the property of the normal distributions, we obtain

$$p(\boldsymbol{w}_* \mid \boldsymbol{x}_*, \boldsymbol{z}_*, \mathcal{D}) = \mathcal{N}(\boldsymbol{w}_* \mid \sigma_{\mathrm{y}}^{-2} \boldsymbol{A}\boldsymbol{S}\bar{\boldsymbol{Z}}^\top \boldsymbol{y}, \bar{\boldsymbol{c}}_{**} - \boldsymbol{A}\bar{\boldsymbol{k}}_* + \boldsymbol{A}\boldsymbol{S}\boldsymbol{A}^\top). \tag{13}$$

We provide the detailed derivation of predictive distributions (10) and (29) in Appendix B of the supplementary material.

Accordingly, the marginal likelihood (9) and the predictive distribution for y_* (10) are similar to those of GPR, except that GPX can obtain the predictive distribution for $\boldsymbol{w}_*$ (29). Since GPX can be used with the same input as GPR if $\boldsymbol{Z} = \boldsymbol{X}$, it can be employed in existing ML models, instead of GPR.

Computational efficiency. As with ordinary GPR, the computational cost of GPX is dominated by the inverse computation. The computation of $\boldsymbol{A}$ requires inverting a square matrix of order nd, $\bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}$. However, because the matrix is block diagonal and every diagonal block comprises $\boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n$, a square matrix of order n, $\boldsymbol{A}$ can be obtained by inverting $\boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n$ only once. The remaining inverse matrix $\boldsymbol{C}^{-1}$ is of order n. Therefore, all the inverse matrices appearing in GPX can be obtained using a naive implementation with a computational complexity of $\mathcal{O}(n^3)$, which is the same as that in GPR. To significantly reduce the computational cost, efficient computation methods for GPR, such as the inducing variable method [34] and KISS-GP [35], can be used for GPX. In addition, because $\bar{\boldsymbol{k}}_*$, $\bar{\boldsymbol{K}}$ and $\bar{\boldsymbol{Z}}$ are sparse matrices, one can obtain the predictive distributions efficiently using libraries for sparse matrix computation.

5 Experiments

In this section, we demonstrate the effectiveness of the proposed model, GPX, quantitatively and qualitatively, by comparing various interpretable models. Through a quantitative evaluation, we evaluated the models based on the following perspectives:

- **Accuracy:** How accurate is the prediction of the interpretable model?
- **Faithfulness:** Are feature contributions indicative of "true" importance?
- **Sufficiency:** Do k-most important features reflect the prediction?
- **Stability:** How consistent are the explanations for similar or neighboring examples?

In addition, we qualitatively evaluated whether the feature contributions produced by the models were appropriate by visualizing them.

All the experiments were done with a computer with Intel Xeon Gold 6132 2.6GHz CPU with 16 cores, and 120GB of main memory.

5.1 Preparation

Datasets. We used eight datasets in the UCI machine learning repository [36], referred to as Digits [37], Abalone [38], Diabetes [39], Boston [15], Fish [40], Wine [41], Paper [42] and Drug [43] in our experiments. We provide the details of the datasets in Appendix C of the supplementary material. Digits dataset is originally a classification dataset for recognizing handwritten digits from 0 to 9. To use it as a regression problem, we transformed the labels into target variables $\boldsymbol{y}$ of scalar values, i.e., the target variables for the labels from 0 to 4 were -1, and those for the remaining labels were 1. With Paper and Drug datasets whose samples were represented as sentences, the original input $\boldsymbol{X}$ and the simplified input $\boldsymbol{Z}$ differed, i.e., we used the 512-dimensional sentence vectors obtained using Sentence Transformers [44] as $\boldsymbol{X}$, while we used bag-of-words binary vectors for the sentences as $\boldsymbol{Z}$. Each of the remaining datasets had the same $\boldsymbol{X}$ and $\boldsymbol{Z}$. In all the datasets, the values of $\boldsymbol{X}$ and $\boldsymbol{y}$ were standardized before training and prediction. For a quantitative evaluation of each dataset, we evaluated the average scores over five experiments performed on different training/test splittings, where the training set was 80% of the entire dataset, whereas the remaining was the test set.

GPX setup. In GPX, we consistently used a scaled RBF kernel defined as (4). The hyperparameters of GPX were estimated based on the method described in Section 4, where they were initialized with $\theta_1 = 1.0$, $\sigma_{\mathrm{y}} = 0.1$ and $\sigma_{\mathrm{w}} = 0.1$. In addition, we initialized bandwidth parameter θ_2 using median heuristics [45].

Comparing methods. We compared GPX with several methods with globally or locally linear weights that can used as interpretable feature contributions for predictions. Lasso [18] and Ridge [17] are standard linear regression models with ℓ_1 and ℓ_2 regularizers, respectively, where their weights are globally shared across all samples. The network lasso ("Network" for short) is a locally linear model that regularizes the weights of nodes such that neighboring nodes in a network have similar weights [21]. In our case, each node represents a sample, and the network is a k-nearest neighbor graph on the samples based on the cosine similarity on $\boldsymbol{X}$. The localized lasso ("Localized" for short) is an extension of the network lasso; it can estimate the sparse and exclusive weights of each sample by further incorporating an $\ell_{1,2}$ regularizer into the network lasso [22]. Furthermore, to compare model-agnostic interpretable methods with GPX, we used LIME [13] and Kernel SHAP [14], which

Table 1: Average and standard deviation of mean squared errors (MSEs) for the predictions of target variables on each dataset (lower is better). The value of bold typeface indicates the best value in each dataset, whereas that underlined indicates that the value is the best in paired t-test with significant level $p < 0.05$. "NA" indicates that their performances cannot be measured owing to memory or computational time limitations. As GPR is not an interpretable model, we only included its performances in this table to show that those of GPX and GPR are similar.

	GPX (ours)	Lasso	Ridge	Localized	Network	GPR
Digits	$\mathbf{0.078 \pm 0.010}$	0.399 ± 0.028	0.398 ± 0.024	0.135 ± 0.042	0.163 ± 0.033	0.074 ± 0.008
Abalone	$\mathbf{0.428 \pm 0.036}$	0.477 ± 0.052	0.477 ± 0.053	0.519 ± 0.023	0.534 ± 0.016	0.427 ± 0.034
Diabetes	$\mathbf{0.493 \pm 0.041}$	0.504 ± 0.039	0.503 ± 0.040	0.610 ± 0.062	0.667 ± 0.091	0.490 ± 0.048
Boston	$\mathbf{0.116 \pm 0.053}$	0.293 ± 0.078	0.284 ± 0.081	0.208 ± 0.070	0.233 ± 0.062	0.116 ± 0.052
Fish	$\mathbf{0.370 \pm 0.061}$	0.437 ± 0.055	0.437 ± 0.055	0.479 ± 0.051	0.523 ± 0.054	0.375 ± 0.066
Wine	$\mathbf{0.579 \pm 0.048}$	0.723 ± 0.039	0.712 ± 0.044	NA	NA	0.605 ± 0.046
Paper	$\mathbf{0.806 \pm 0.054}$	0.821 ± 0.047	0.936 ± 0.057	0.981 ± 0.058	0.919 ± 0.088	0.762 ± 0.087
Drug	$\mathbf{0.835 \pm 0.027}$	0.875 ± 0.037	0.911 ± 0.036	NA	NA	0.844 ± 0.033

Table 2: Average and standard deviation of faithfulness scores on each dataset (higher is better). This table can be interpreted similarly as Table 1. For Wine dataset, we could not measure the scores of the methods except for GPX owing to computational time limitations. For Paper and Drug datasets, we did not evaluate the scores as changes in $\boldsymbol{Z}$ cannot reflect $\boldsymbol{X}$.

	GPX (ours)	GPR+LIME	GPR+SHAP	Localized	Network
Digits	$\mathbf{0.888 \pm 0.003}$	0.384 ± 0.038	0.651 ± 0.013	0.300 ± 0.029	0.352 ± 0.021
Abalone	$\mathbf{0.898 \pm 0.017}$	0.775 ± 0.024	NA	0.432 ± 0.042	0.497 ± 0.033
Diabetes	$\mathbf{0.966 \pm 0.008}$	0.844 ± 0.027	0.928 ± 0.010	0.340 ± 0.074	0.365 ± 0.086
Boston	$\mathbf{0.898 \pm 0.026}$	0.693 ± 0.035	0.869 ± 0.009	0.562 ± 0.075	0.525 ± 0.039
Fish	$\mathbf{0.902 \pm 0.016}$	0.672 ± 0.043	0.826 ± 0.033	0.480 ± 0.046	0.394 ± 0.090

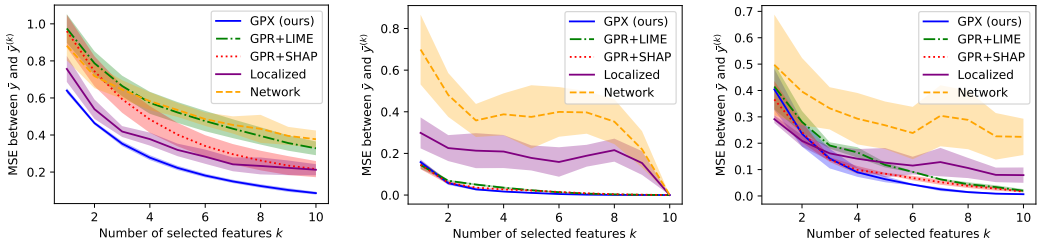
produce a locally linear model for each test sample to explain the prediction by a black-box prediction model. For a fair comparison, we used GPR as the prediction model. The hyperparameters of GPR were estimated similarly as for GPX. Meanwhile, those of the remaining comparing methods were optimized by grid search. We provide the detailed description of the comparing methods in the Appendix D of the supplementary material.

5.2 Results

Accuracy. First, we demonstrate the predictive performances of GPX and the comparing methods in Table 1. We found that GPX achieved the lowest predictive errors on all the datasets, compared with the globally or locally linear models. In addition, we found that the predictive errors of GPX and GPR were comparable on all the datasets. This result indicates that GPR can be replaced by GPX to achieve similar predictive performances.

Faithfulness. Assessing the correctness of the estimated contribution of each feature to a prediction requires a reference "true" contribution for comparison. As this is rarely available, a typical approach for measuring the faithfulness of the contributions produced by interpretable models is to rely on the proxy notion of the contributions: observing the effect of removing features on the model's prediction. Following previous studies [26, 46], we computed the faithfulness score by removing features one-by-one, measuring the differences between the original predictions and the predictions from the inputs without the removed features, and calculating the correlation between the differences and the contributions of the removed features.

Table 2 shows the faithfulness scores of GPX and the comparing methods. Here, we denote the results of LIME and Kernel SHAP using GPR as the black-box prediction model by GPR+LIME and GPR+SHAP, respectively. We found that GPX achieved the best faithfulness scores on all the datasets. As GPX predicts and explains using a single locally linear model for each test sample, when removing a feature from the input, the contribution of the feature is subtracted from the prediction directly. Meanwhile, because GPR+LIME and GPR+SHAP have different prediction and explanation models, a gap may exist between the estimated contribution in the explanation model and the latent



(a) Digits (b) Diabetes (c) Boston

Figure 2: Average sufficiency scores on Digits, Diabetes, and Boston datasets (lower is better). The filled area on each line indicates its standard deviation.

Table 3: Average and standard deviation of stability score on each dataset (lower is better). This table can be interpreted similarly as Table 1. For Wine and Drug datasets, owing to computational time limitations, we could not measure the scores of the methods except for GPX.

	GPX (ours)	GPR+LIME	GPR+SHAP	Localized	Network
Digits	**0.034 ± 0.011**	0.122 ± 0.027	0.073 ± 0.019	0.117 ± 0.065	0.133 ± 0.095
Abalone	**0.338 ± 0.056**	2.820 ± 1.445	NA	3.094 ± 1.783	5.994 ± 3.383
Diabetes	**0.015 ± 0.002**	0.482 ± 0.084	0.196 ± 0.030	0.758 ± 0.581	1.271 ± 0.261
Boston	**0.117 ± 0.038**	0.545 ± 0.234	0.258 ± 0.087	0.233 ± 0.268	0.455 ± 0.670
Fish	**0.222 ± 0.068**	0.814 ± 0.384	0.398 ± 0.095	1.414 ± 1.318	1.994 ± 1.923
Paper	**0.001 ± 0.000**	0.074 ± 0.014	0.041 ± 0.019	0.014 ± 0.006	0.223 ± 0.045

contribution in the prediction. Because the predictions by GPX and GPR were performed using similar calculations, their faithfulness differences were likely due to the gap.

Sufficiency. In general, the inputs contain many irrelevant features that do not contribute to the predictions, and discovering important features in all the features is difficult for users of the models. Therefore, a desirable property of the interpretable models is that it can assign high contributions only for important features that affect the predictions well. To quantify how each method satisfies the property, we define the sufficiency score at k, where k is the number of important features. In particular, the sufficiency score at k was computed by identifying k important features in the descending order of the absolute values of their estimated contributions, predicting from the inputs having only k important features, and comparing them against the original predictions. Because the number of important features varied according to the sample and dataset, we evaluated them at $k = 1, 2, \cdots, 10$.

Figure 2 shows the sufficiency scores of GPX and the comparing methods. We found that GPX outperformed the others on Digits dataset, whereas GPX, GPR+LIME, and GPR+SHAP produced the best sufficiency scores on Diabetes dataset. These results indicate that GPX was appropriately assigned high contributions for the important features. On Boston dataset, we found that GPX was slightly inferior to the localized lasso at $k = 1, 2$, although GPX outperformed it at $k \geq 3$. This was because the localized lasso has a regularizer that induces sparse weights. This result suggests that GPX can be further improved by employing the mechanism for generating sparse weights.

Stability. To generate meaningful explanations, interpretable methods must be robust against local perturbations from the input, as explanations that are sensitive to slight changes in the input may be regarded as inconsistent by users. In particular, flexible models such as locally linear models might be sensitive to such changes for achieving better predictions. As with Alvarez–Melis and Jaakkola [26], we used the following quantity for measuring the stability of the estimated weights for test sample $(\boldsymbol{x}_*, \boldsymbol{z}_*)$, as follows:

$$L(\boldsymbol{x}_*, \boldsymbol{z}_*) = \max_{\boldsymbol{x}'_*, \boldsymbol{z}'_* \in \mathcal{B}_\epsilon(\boldsymbol{x}_*)} \frac{\|\boldsymbol{w}'_* - \boldsymbol{w}_*\|_2}{\|\boldsymbol{z}'_* - \boldsymbol{z}_*\|_2}, \quad \text{where,} \quad \mathcal{B}_\epsilon(\boldsymbol{x}_*) = \{(\boldsymbol{x}'_*, \boldsymbol{z}'_*) \in \mathcal{D}_{\text{te}} \mid \frac{1}{m}\|\boldsymbol{x}'_* - \boldsymbol{x}_*\|_2 < \epsilon\}, \tag{14}$$

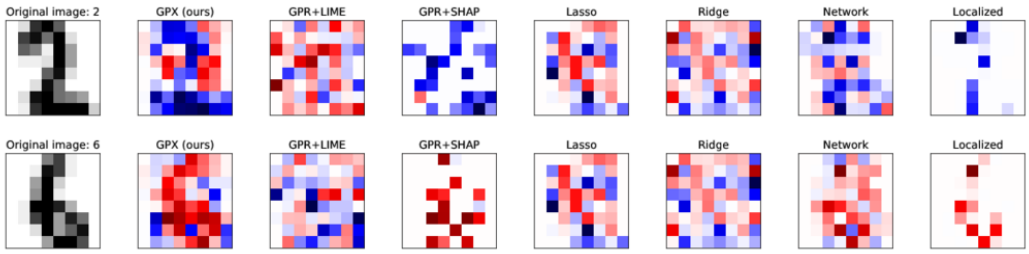


Figure 3: Examples of estimated weights of each model on Digits dataset. The upper row shows the weights for the sample with digit two ($y = -1$), whereas the bottom one displays those for the sample with digit six ($y = 1$). Red and blue denote positive and negative weights, respectively, and their color strengths represent their magnitudes.

where, $\mathcal{D}_{\text{te}} = \{(\boldsymbol{x}_*, \boldsymbol{z}_*)\}$ is a set of test samples; $\boldsymbol{w}_*$ and $\boldsymbol{w}'_*$ are the estimated weights associated with test samples $(\boldsymbol{x}_*, \boldsymbol{z}_*)$ and $(\boldsymbol{x}'_*, \boldsymbol{z}'_*)$, respectively; $\epsilon > 0$ is a parameter that determines neighboring samples; m is the dimensionality of $\boldsymbol{x}_*$. We set $\epsilon = 0.05$ in our experiments. Intuitively, the stability score will be high when the estimated weights for the sample and its neighboring samples are similar. Subsequently, we computed the stability score on a dataset by averaging the quantity (14) on all the test samples in the dataset.

Table 3 shows the stability scores on each dataset. We found that GPX achieved the best stability scores on all the datasets. For GPR+LIME and GPR+SHAP, we found that the stability scores were lower than that of GPX, although the prediction powers of GPX and GPR were comparable. This would be because LIME and Kernel SHAP estimated the weights independently over the test samples. With the localized and network lasso, we found that the variance of the stability score over the datasets was large.

Qualitative comparison. Finally, we qualitatively compared the estimated weights using GPX and the comparing methods on Digits dataset, in which the appropriate contributions for predictions were apparent. For this comparison, we rescaled the inputs $\boldsymbol{X}$ and $\boldsymbol{Z}$ to be within $[0, 1]$.

Figure 3 shows the estimated weights on two samples. We provide the results for all the digits in Appendix E of the supplementary material. On this dataset, the appropriate weights can be obtained by assigning weights having the same sign with the target variable to black pixels. We found that the methods except for GPX and the localized lasso could not estimate reasonable weights. Meanwhile, the weights estimated by GPX and the localized lasso were appropriate, although they exhibited different characteristics, i.e., dense weights from GPX, whereas sparse ones from the localized lasso. The task determines the better explanation; however, as showing important regions rather than pixels is meaningful for images, the estimated weights using GPX would be easier to interpret on Digits dataset. Furthermore, the degree of sparsity in the localized lasso can be changed as a hyperparameter; if the value of the hyperparameter is zero, the localized lasso is identical to the network lasso. However, because the estimated weights using the network lasso were inappropriate, those using GPX cannot be mimicked by the localized lasso.

6 Conclusion

We proposed a GP-based regression model with sample-wise explanations. The proposed model assumes that each sample has a locally linear model, which is used for both prediction and explanation, and the weight vector of the locally linear model are generated from multivariate GP priors. The hyperparameters of the proposed models were estimated by maximizing the marginal likelihood, in which all the weight vectors were integrated out. Subsequently, for a test sample, the proposed model predicted its target variable and weight vector with uncertainty. In the experiments, we confirmed that the proposed model outperformed the existing globally and locally linear models and achieved comparable performances with the standard GPR in terms of predictive performance. We confirmed that the proposed model was superior to the existing methods, including model-agnostic interpretable methods, in terms of three interpretability measurements. Subsequently, we showed that the feature weights estimated by the proposed model were appropriate as the explanation.

In future studies, we will confirm the effectiveness of the proposed model by applying its concept into various problems in which GPs have been successfully used, such as time-series forecasting and black-box optimization, In addition, we will extend the proposed model for further improvements in interpretability, e.g., by employing the mechanism of inducing sparsity for the weight vectors.

Broader Impact

The proposed model could be applied to a wide range of applications in which Gaussian process regression has been used, such as finance [6], geostatistics [7], material science [8] and medical science [9, 10]. The proposed model could be used to make a non-linear prediction with an explanation for an individual sample, e.g., company, country, material object, and patient in these applications.

Using the proposed model brings in some benefits. For example, the explanation provides the opportunities for users of the proposed model to judge whether the prediction is reasonable and whether it is performed by fair decision. Furthermore, the uncertainties in the prediction and explanation in which the proposed model estimates help improve the correctness of the judgment. Consequently, if the users feel that the explanation is unreasonable or unfair, they could fix the model or the training data to avoid such a false explanation next time. On the other hand, the proposed model could face risk by increasing predictability and explainability, i.e., when the users unduly trust the proposed model or ignore the large uncertainties, the users could trust the prediction and the explanation even when they are wrong; consequently, they could make the wrong, unfair or biased decision making.

Evaluating the reasonability of the explanation needs expert knowledge in the applications, and it is rarely available in general. Therefore, we encourage research to investigate whether the explanation produced by the proposed model is reasonable based on the expert knowledge in the applications. For the aforementioned risk, we encourage research to investigate the influence of the explanation by the interpretable models, including the proposed model to the trustworthiness of the models and users' decision making.

References

[1] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[2] Andrew Gordon Wilson, David A Knowles, and Zoubin Ghahramani. Gaussian process regression networks. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1139–1146, 2012.

[3] Lehel Csató, Ernest Fokoué, Manfred Opper, Bernhard Schottky, and Ole Winther. Efficient approaches to gaussian process classification. In *Advances in neural information processing systems*, pages 251–257, 2000.

[4] Stephen Roberts, Michael Osborne, Mark Ebden, Steven Reece, Neale Gibson, and Suzanne Aigrain. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984):20110550, 2013.

[5] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[6] Joan Gonzalvez, Edmond Lezmi, Thierry Roncalli, and Jiali Xu. Financial applications of gaussian processes and bayesian optimization. *arXiv preprint arXiv:1903.04841*, 2019.

[7] Gustau Camps-Valls, Jochem Verrelst, Jordi Munoz-Mari, Valero Laparra, Fernando Mateo-Jimenez, and Jose Gomez-Dans. A survey on gaussian processes for earth-observation data analysis: A comprehensive investigation. *IEEE Geoscience and Remote Sensing Magazine*, 4(2):58–78, 2016.

[8] Yichi Zhang, Daniel W Apley, and Wei Chen. Bayesian optimization for materials design with mixed quantitative and qualitative variables. *Scientific Reports*, 10(1):1–13, 2020.

[9] Li-Fang Cheng, Gregory Darnell, Bianca Dumitrascu, Corey Chivers, Michael E Draugelis, Kai Li, and Barbara E Engelhardt. Sparse multi-output gaussian processes for medical time series prediction. *arXiv preprint arXiv:1703.09112*, 2017.

[10] Joseph Futoma. *Gaussian process-based models for clinical time series in healthcare*. PhD thesis, Duke University, 2018.

[11] Christoph Molnar. *Interpretable Machine Learning*. 2019. `https://christophm.github.io/interpretable-ml-book/`.

[12] Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pages 883–892, 2018.

[13] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144. ACM, 2016.

[14] Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

[15] David Harrison Jr and Daniel L Rubinfeld. Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5:81–102, 1978.

[16] Mauricio A Álvarez, Lorenzo Rosasco, and Neil D Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, 4(3):195–266, 2012.

[17] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[18] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[19] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[20] David P Wipf and Srikantan S Nagarajan. A new view of automatic relevance determination. In *Advances in neural information processing systems*, pages 1625–1632, 2008.

[21] David Hallac, Jure Leskovec, and Stephen Boyd. Network Lasso: Clustering and Optimization in Large Graphs. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 387–396. ACM, 2015.

[22] Makoto Yamada, Takeuchi Koh, Tomoharu Iwata, John Shawe-Taylor, and Samuel Kaski. Localized Lasso for High-Dimensional Regression. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 325–333. PMLR, April 2017.

[23] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. In *Advances in Neural Information Processing Systems*, pages 8928–8939, 2019.

[24] Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. " what is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12(8), 2017.

[25] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems*, pages 9240–9251, 2019.

[26] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*, pages 7775–7784, 2018.

[27] Patrick Schwab, Djordje Miladinovic, and Walter Karlen. Granger-causal attentive mixtures of experts: Learning important features with neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4846–4853, 2019.

[28] Yuya Yoshikawa and Tomoharu Iwata. Neural generators of sparse local linear models for achieving both accuracy and interpretability. *arXiv preprint arXiv:2003.06441*, 2020.

[29] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D Sculley. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1487–1495, 2017.

[30] S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11(Apr):1201–1242, 2010.

[31] Krikamol Muandet, Kenji Fukumizu, Francesco Dinuzzo, and Bernhard Schölkopf. Learning from distributions via support measure machines. In *Advances in neural information processing systems*, pages 10–18, 2012.

[32] Yuya Yoshikawa, Tomoharu Iwata, and Hiroshi Sawada. Latent support measure machines for bag-of-words data classification. In *Advances in neural information processing systems*, pages 1961–1969, 2014.

[33] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[34] Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.

[35] Andrew Wilson and Hannes Nickisch. Kernel interpolation for scalable structured gaussian processes (kiss-gp). In *International Conference on Machine Learning*, pages 1775–1784, 2015.

[36] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[37] Optical recognition of handwritten digits data set. `https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits`.

[38] Abalone data set. `https://archive.ics.uci.edu/ml/datasets/Abalone`.

[39] Diabetes data set. `https://archive.ics.uci.edu/ml/datasets/diabetes`.

[40] Qsar fish toxicity data set. `https://archive.ics.uci.edu/ml/datasets/QSAR+fish+toxicity`.

[41] Wine quality data set. `https://archive.ics.uci.edu/ml/datasets/wine+quality`.

[42] Paper reviews data set. `https://archive.ics.uci.edu/ml/datasets/Paper+Reviews`.

[43] Drug review dataset (druglib.com) data set. `https://archive.ics.uci.edu/ml/datasets/Drug+Review+Dataset+%28Druglib.com%29#`.

[44] Nils Reimers and Iryna Gurevych. Making monolingual sentence embeddings multilingual using knowledge distillation. *arXiv preprint arXiv:2004.09813*, 2020. `https://github.com/UKPLab/sentence-transformers`.

[45] Damien Garreau, Wittawat Jitkrittum, and Motonobu Kanagawa. Large sample analysis of the median heuristic. *arXiv preprint arXiv:1707.07269*, 2017.

[46] Umang Bhatt, Adrian Weller, and José MF Moura. Evaluating and aggregating feature-based model explanations. *arXiv preprint arXiv:2005.00631*, 2020.

[47] K. B. Petersen and M. S. Pedersen. The matrix cookbook, 2012. Version: November 15, 2012.

[48] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[49] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

Supplementary Material: Gaussian Process Regression for Local Explanation

A Feature Description and Additional Examples for the Boston Housing Dataset

Table 4: Feature names and their descriptions for the Boston housing dataset.

Feature name	Description
CRIM	Per capita crime rate by town
ZN	Proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS	Proportion of non-retail business acres per town.
CHAS	Charles River dummy variable (1 if tract bounds river; 0 otherwise)
NOX	Nitric oxides concentration (parts per 10 million)
RM	Average number of rooms per dwelling
AGE	Proportion of owner-occupied units built prior to 1940
DIS	Weighted distances to five Boston employment centers
RAD	Index of accessibility to radial highways
TAX	Full-value property-tax rate per $10,000
PTRATIO	Pupil-teacher ratio by town
B	1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
LSTAT	% lower status of the population
MEDV	Median value of owner-occupied homes in $1000's

The Boston housing dataset, referred to as "Boston" in our experiments, contains information collected by the U.S. Census Service regarding housing in the area of Boston, Massachusetts [15] and is used for predicting house prices based on the information. Table 4 lists the names of the features and their descriptions for the Boston housing dataset.

Figure 4 presents four examples of feature contributions estimated by GPX. We found that each of these examples has different feature contributions, although some of the features, such as "AGE" and "DIS," had consistent positive or negative contributions, respectively.

B Detailed Derivation of Predictive Distributions

In this appendix, we describe the derivation of predictive distributions in detail. For a new test sample $(\boldsymbol{x}_*, \boldsymbol{z}_*)$, our goal is to infer the predictive distributions of the target variable y_* and weight vector $\boldsymbol{w}_*$.

Predictive distribution of y_*. The predictive distribution of y_* is obtained similarly to the standard GPR [1]. In Section 4, we demonstrated that the marginal distribution of training target variables $\boldsymbol{y}$ for GPX is defined as

$$p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{Z}) = \iint p(\boldsymbol{y}, \boldsymbol{W}, \boldsymbol{G} \mid \boldsymbol{X}, \boldsymbol{Z}) d\boldsymbol{W} d\boldsymbol{G} = \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{0}, \boldsymbol{C}\right), \tag{15}$$

where $\boldsymbol{C} = \sigma_{\mathrm{y}}^2 \boldsymbol{I}_n + (\boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n) \odot \boldsymbol{Z}\boldsymbol{Z}^\top$. According to (15), the joint marginal distribution of $\boldsymbol{y}$ and y_* is defined as

$$p(\boldsymbol{y}, y_* \mid \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{x}_*, \boldsymbol{z}_*) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix} \middle| \boldsymbol{0}, \begin{bmatrix} \boldsymbol{C} & \boldsymbol{c}_* \\ \boldsymbol{c}_*^\top & c_{**} \end{bmatrix} \right), \tag{16}$$

where $\boldsymbol{c}_* = \left(k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_i) \boldsymbol{z}_*^\top \boldsymbol{z}_i\right)_{i=1}^n \in \mathbb{R}^n$, and $c_{**} = \sigma_{\mathrm{y}}^2 + \left(k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma_{\mathrm{w}}^2\right) \boldsymbol{z}_*^\top \boldsymbol{z}_* \in \mathbb{R}$. The predictive distribution of y_* is the conditional distribution of y_* given $\boldsymbol{y}$ with training and testing inputs. Therefore, it can be obtained by applying the formula of conditional distributions for normal distributions [47, Eq. (354)] to (16) as follows:

$$p(y_* \mid \boldsymbol{x}_*, \boldsymbol{z}_*, \mathcal{D}) = \mathcal{N}\left(y_* \mid \boldsymbol{c}_*^\top \boldsymbol{C}^{-1} \boldsymbol{y}, c_{**} - \boldsymbol{c}_*^\top \boldsymbol{C}^{-1} \boldsymbol{c}_*\right). \tag{17}$$

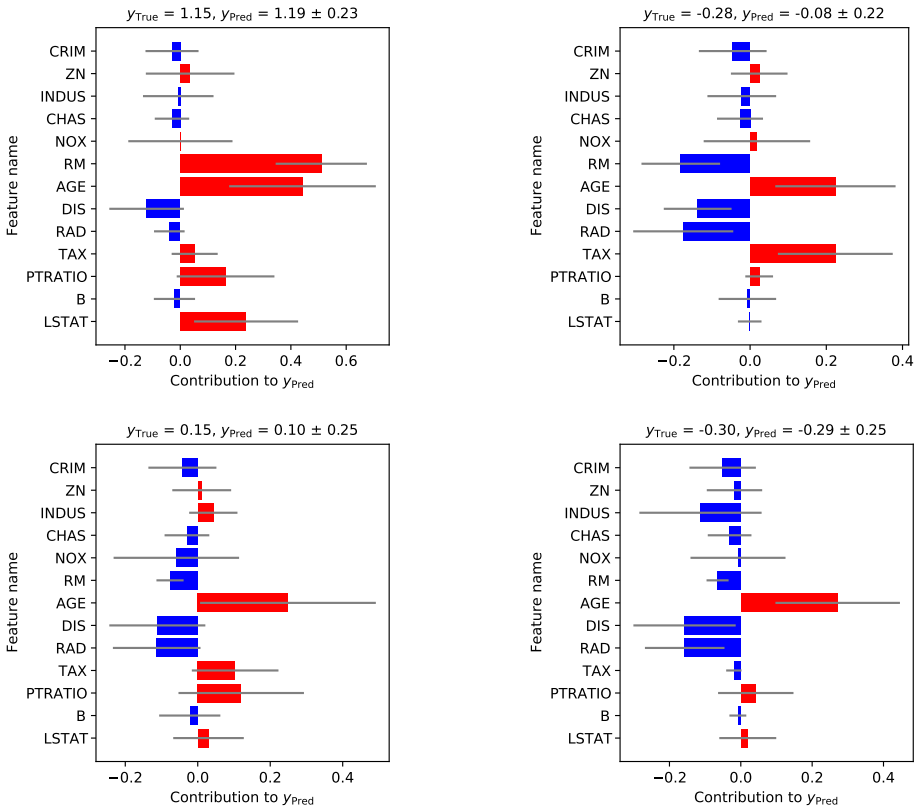


Figure 4: Examples of feature contributions estimated by GPX for the Boston housing dataset. Here, the error bars denote the standard deviations or feature contributions.

Predictive distribution of $\boldsymbol{w}_*$. The predictive distribution of $\boldsymbol{w}_*$ can be obtained by solving the following equation:

$$p(\boldsymbol{w}_* \mid \boldsymbol{x}_*, \boldsymbol{z}_*, \mathcal{D}) = \int p(\boldsymbol{w}_* \mid \boldsymbol{W}, \boldsymbol{X}, \boldsymbol{x}_*) p(\boldsymbol{W} \mid \mathcal{D}) d\boldsymbol{W}, \tag{18}$$

where the first integrand $p(\boldsymbol{w}_* \mid \boldsymbol{W}, \boldsymbol{X}, \boldsymbol{x}_*)$ is the conditional distribution of $\boldsymbol{w}_*$ and the second integrand $p(\boldsymbol{W} \mid \mathcal{D})$ is the posterior distribution of $\boldsymbol{W}$. The conditional distribution of $\boldsymbol{w}_*$ is derived similarly to the conditional distribution of y_* (17). The distribution of $\boldsymbol{W}$ in which the functions $\boldsymbol{G}$ are integrated out is given by

$$p(\boldsymbol{W} \mid \boldsymbol{X}) = \int p(\boldsymbol{W} \mid \boldsymbol{G}) p(\boldsymbol{G} \mid \boldsymbol{X}) d\boldsymbol{G} = \prod_{l=1}^{d} \mathcal{N}\left(\boldsymbol{W}_{\cdot,l} \mid \boldsymbol{0}, \boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n\right), \tag{19}$$

where $\boldsymbol{W}_{\cdot,l}$ is the lth column vector of $\boldsymbol{W}$. According to this, the joint distribution of $\boldsymbol{W}$ and $\boldsymbol{w}_*$ is defined as

$$p(\boldsymbol{W}, \boldsymbol{w}_* \mid \boldsymbol{X}, \boldsymbol{x}_*) = \prod_{l=1}^{d} \mathcal{N}\left(\begin{bmatrix} \boldsymbol{W}_{\cdot,l} \\ w_{*,l} \end{bmatrix} \middle| \boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n & \boldsymbol{k}_* \\ \boldsymbol{k}_*^\top & k_{**} \end{bmatrix}\right), \tag{20}$$

where we let $\boldsymbol{k}_* = (k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_i))_{i=1}^n$ and $k_{**} = k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma_{\mathrm{w}}^2$. Subsequently, we can obtain the conditional distribution of $\boldsymbol{w}_*$ by applying the formula of conditional distributions for normal distributions [47, Eq. (354)] to (20) as follows:

$$p(\boldsymbol{w}_* \mid \boldsymbol{W}, \boldsymbol{X}, \boldsymbol{x}_*) = \prod_{l=1}^{d} \mathcal{N}\left(w_{*,l} \,\middle|\, \boldsymbol{k}_*^\top \left(\boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{W}_{\cdot,l}, k_{**} - \boldsymbol{k}_*^\top \left(\boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{k}_*\right). \tag{21}$$

Table 5: Specification of datasets.

	n	d
Digits	1,797	64
Abalone	4,177	10
Diabetes	442	10
Boston	506	13
Fish	908	6
Wine	6,497	11
Paper	399	2,990
Drug	3,989	2,429

Here, we can rewrite (21) as a single d-dimensional multivariate normal distribution as follows:

$$p(\boldsymbol{w}_* \mid \boldsymbol{W}, \boldsymbol{X}, \boldsymbol{x}_*) = \mathcal{N}\left(\boldsymbol{w}_* \;\middle|\; \bar{\boldsymbol{k}}_*^\top \left(\bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}\right)^{-1} \mathsf{vec}(\boldsymbol{W}),\ \bar{\boldsymbol{c}}_{**} - \bar{\boldsymbol{k}}_*^\top \left(\bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}\right)^{-1} \bar{\boldsymbol{k}}_*\right), \tag{22}$$

where $\bar{\boldsymbol{K}}$ is a block diagonal matrix of order nd whose block is $\boldsymbol{K}$, $\mathsf{vec}(\cdot)$ is a function that flattens the input matrix in column-major order, and $\bar{\boldsymbol{c}}_{**} = \left(k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma_{\mathrm{w}}^2\right) \boldsymbol{I}_d$. $\bar{\boldsymbol{k}}_*$ is an nd-by-d block matrix, where each block is an n-by-1 matrix the and (l, l)-block of the block matrix is $(k_\theta(\boldsymbol{x}_*, \boldsymbol{x}_i))_{i=1}^n$ for $l = 1, 2, \cdots, d$, while the other blocks are zero matrices. By letting $\boldsymbol{A} = \bar{\boldsymbol{k}}_*^\top \left(\bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}\right)^{-1}$, we obtain

$$p(\boldsymbol{w}_* \mid \boldsymbol{W}, \boldsymbol{X}, \boldsymbol{x}_*) = \mathcal{N}(\boldsymbol{w}_* \mid \boldsymbol{A}\mathsf{vec}(\boldsymbol{W}), \bar{\boldsymbol{c}}_{**} - \boldsymbol{A}\bar{\boldsymbol{k}}_*). \tag{23}$$

To derive the posterior distribution of $\boldsymbol{W}$, $p(\boldsymbol{W} \mid \mathcal{D})$, we first consider the joint distribution of $\boldsymbol{W}$ and $\mathcal{D}$. This distribution is straightforwardly obtained as

$$p(\boldsymbol{W}, \mathcal{D}) = \prod_{l=1}^{d} \mathcal{N}\left(\boldsymbol{W}_{\cdot,l} \mid \boldsymbol{0}, \boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n\right) \prod_{i=1}^{n} \mathcal{N}(y_i \mid \boldsymbol{w}_i^\top \boldsymbol{z}_i, \sigma_{\mathrm{y}}^2), \tag{24}$$

which can be rewritten as

$$p(\boldsymbol{W}, \mathcal{D}) = \mathcal{N}\left(\mathsf{vec}(\boldsymbol{W}) \mid \boldsymbol{0}, \bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}\right) \mathcal{N}\left(\boldsymbol{y} \mid \bar{\boldsymbol{Z}}\mathsf{vec}(\boldsymbol{W}), \sigma_{\mathrm{y}}^2 \boldsymbol{I}_n\right), \tag{25}$$

where $\bar{\boldsymbol{Z}} = (\mathsf{diag}(\boldsymbol{Z}_{\cdot,1}), \mathsf{diag}(\boldsymbol{Z}_{\cdot,2}), \cdots, \mathsf{diag}(\boldsymbol{Z}_{\cdot,d})) \in \mathbb{R}^{n \times nd}$. By applying the formula of conditional distributions of normal distributions [48, Eqs. (2.113)–(2.117)] to (25), we can obtain

$$p(\boldsymbol{W} \mid \mathcal{D}) = \mathcal{N}(\mathsf{vec}(\boldsymbol{W}) \mid \sigma_{\mathrm{y}}^{-2} \boldsymbol{\Sigma} \bar{\boldsymbol{Z}}^\top \boldsymbol{y}, \boldsymbol{\Sigma}), \quad \text{where} \quad \boldsymbol{\Sigma} = \left(\boldsymbol{L}^{-1} - \bar{\boldsymbol{Z}}^\top (\sigma_{\mathrm{y}}^{-2} \boldsymbol{I}_n) \bar{\boldsymbol{Z}}\right)^{-1}, \tag{26}$$

where we let $\boldsymbol{L} = \bar{\boldsymbol{K}} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_{nd}$. Here, the computation of $\boldsymbol{\Sigma}$ requires inverting a square matrix of order nd with a computational complexity of $\mathcal{O}(n^3 d^3)$. By using the Woodbury identity [47, Eq. (156)] to compute this inversion efficiently, we can transform $\boldsymbol{\Sigma}$ into $\boldsymbol{S} = \boldsymbol{L} - \boldsymbol{L}\bar{\boldsymbol{Z}}^\top \boldsymbol{C}^{-1} \bar{\boldsymbol{Z}} \boldsymbol{L}$, which requires inverting a matrix of order n, $\boldsymbol{C} = \sigma_{\mathrm{y}}^2 \boldsymbol{I}_n + (\boldsymbol{K} + \sigma_{\mathrm{w}}^2 \boldsymbol{I}_n) \odot \boldsymbol{Z}\boldsymbol{Z}^\top$. Consequently, we obtain

$$p(\boldsymbol{W} \mid \mathcal{D}) = \mathcal{N}(\mathsf{vec}(\boldsymbol{W}) \mid \sigma_{\mathrm{y}}^{-2} \boldsymbol{S} \bar{\boldsymbol{Z}}^\top \boldsymbol{y}, \boldsymbol{S}). \tag{27}$$

From (23) and (27), one can see that (18) can be represented by the following equation:

$$p(\boldsymbol{w}_* \mid \boldsymbol{x}_*, \boldsymbol{z}_*, \mathcal{D}) = \int \mathcal{N}(\boldsymbol{w}_* \mid \boldsymbol{A}\mathsf{vec}(\boldsymbol{W}), \bar{\boldsymbol{c}}_{**} - \boldsymbol{A}\bar{\boldsymbol{k}}_*)\, \mathcal{N}(\mathsf{vec}(\boldsymbol{W}) \mid \sigma_{\mathrm{y}}^{-2} \boldsymbol{S} \bar{\boldsymbol{Z}}^\top \boldsymbol{y}, \boldsymbol{S}) d\boldsymbol{W}. \tag{28}$$

This integral can be obtained in a closed form, as shown in [48, Eqs. (2.113)–(2.117)]. Therefore, we can obtain the predictive distribution of $\boldsymbol{w}_*$ as follows:

$$p(\boldsymbol{w}_* \mid \boldsymbol{x}_*, \boldsymbol{z}_*, \mathcal{D}) = \mathcal{N}(\boldsymbol{w}_* \mid \sigma_{\mathrm{y}}^{-2} \boldsymbol{A}\boldsymbol{S}\bar{\boldsymbol{Z}}^\top \boldsymbol{y}, \bar{\boldsymbol{c}}_{**} - \boldsymbol{A}\bar{\boldsymbol{k}}_* + \boldsymbol{A}\boldsymbol{S}\boldsymbol{A}^\top). \tag{29}$$

C Specification of Datasets

We considered eight datasets from the UCI machine learning repository [36], which were referred to as Digits [37], Abalone [38], Diabetes [39], Boston [15], Fish [40], Wine [41], Paper [42], and Drug [43] in our experiments.

The first six datasets consisted of tabular data. We treated the original inputs $\boldsymbol{X}$ and simplified inputs $\boldsymbol{Z}$ identically in our experiments. The Digits dataset was originally developed as a classification dataset for recognizing handwritten digits from zero to nine. As described in Section 5.1, we used this dataset for a regression problem by transforming the digit labels into binary values of 1 or -1. Here, we used only the testing set from the original Digits dataset because that is how scikit-learn [49] distributes this dataset. The Abalone dataset is a dataset for predicting the age of abalone based on physical measurements. The Diabetes dataset is a dataset for predicting the onset of diabetes based on diagnostic measures. The Boston dataset is a dataset for predicting house prices, as described in Appendix A. The Fish dataset is a dataset for predicting acute aquatic toxicity toward the fish pimephales promelas for a set of chemicals. The Wine dataset is a dataset for predicting the quality of white and red wines based on physicochemical tests. The remaining two datasets are text datasets. The Paper dataset is a dataset for predicting evaluation scores for papers based on review texts written mainly in Spanish. The Drug dataset is a drug review dataset for predicting 10-star ratings for drugs based on patient review texts. For each dataset, $\boldsymbol{X}$ and $\boldsymbol{Z}$ are different. Specifically, we used the 512-dimensional sentence vectors obtained using sentence transformers [44] as $\boldsymbol{X}$ and used bag-of-words binary vectors of the sentences as $\boldsymbol{Z}$, where the cutoff frequencies for words were set to two and five for the Paper and Drug datasets, respectively. Table 5 lists the number of samples n and number of features d in each dataset.

D Detailed Description of Comparing Methods

In this appendix, we describe the implementation and hyperparameter search methods used for comparing methods.

We implemented GPR using PyTorch v1.5.0[1]. All hyperparameters for GPR were estimated by maximizing marginal likelihood [1], where we initialized the hyperparameters to the same values as those for GPX. For Lasso and Ridge, we used the implementations provided by scikit-learn [49]. The hyperparameters that regularize the strengths of the ℓ_1 and ℓ_2 regularizers in Lasso and Ridge, respectively, were optimized through a grid search using functions provided by scikit-learn (i.e., `sklearn.linear_model.LassoCV` and `sklearn.linear_model.RidgeCV`) with the default options. The search range for the hyperparameters for Lasso was limited to within 100 grid points such that the ratio of its minimum value to its maximum value was capped at 0.001, while that for Ridge was limited to within a range of $\{0.1, 1, 10\}$. For the localized lasso, we used the original implementation written in Python[2]. The hyperparameters and their search ranges for the localized lasso are the strength of network regularization $\lambda_1 \in \{1, 3, 5, 7\}$, strength of the $\ell_{1,2}$ regularizer $\lambda_2 \in \{0.01, 0.1, 1, 10\}$, and $k \in \{5, 10, 15\}$ for the k-nearest-neighbor graph. The hyperparameters were optimized through a grid search. The network lasso is a special case of the localized lasso. If λ_2 for the localized lasso is zero, then the localized lasso is identical to the network lasso. Therefore, we used the implementation of the localized lasso and set $\lambda_2 = 0$ for the network lasso. The hyperparameter search for the network lasso was the same as that for the localized lasso, except for the setting of λ_2. For LIME and Kernel SHAP, we used the original implementations[3].

E Additional Results for the Digits Dataset

Table 6 shows the computational times of each of the methods on the Digits dataset. First, the training times of GPX and GPR were much the same, and GPX was significantly faster than the localized and network lasso. Since the localized and network lasso requires the hyperparameter search, the actual training times of the localized and network lasso were about 48 and 12 times longer than the times

[1] `https://pytorch.org/`

[2] `https://riken-yamada.github.io/localizedlasso.html`

[3] LIME: `https://github.com/marcotcr/lime`, Kernel SHAP: `https://github.com/slundberg/shap`

Table 6: Training, prediction and total times (seconds) of each method on the Digits dataset. Here, the training times of GPR+LIME/SHAP are those of GPR, while the prediction times of GPR+LIME/SHAP are those of producing explanations for all the test samples.

	GPX	GPR+LIME	GPR+SHAP	Localized	Network
Training	5.74	5.53	5.76	105.84	106.16
Prediction	24.57	155.54	2643.06	1.59	1.61
Total	30.31	161.07	2648.82	107.43	107.77

shown in the table, respectively. Second, the prediction time of GPX was significantly faster than GPR+LIME/SHAP. In total, GPX was the fastest in the comparing methods.

Figure 5 presents additional examples of estimated weights for the Digits dataset. We found that the weights estimated by GPX were appropriately assigned such that the regions of black pixels have weights with the same signs as those of the target variables.

In terms of the stability of explanations, estimated weights for the same digit should be similar. Figure 6 presents three examples of estimated weights for the digit two. We found that GPX estimated similar weights for all three examples.

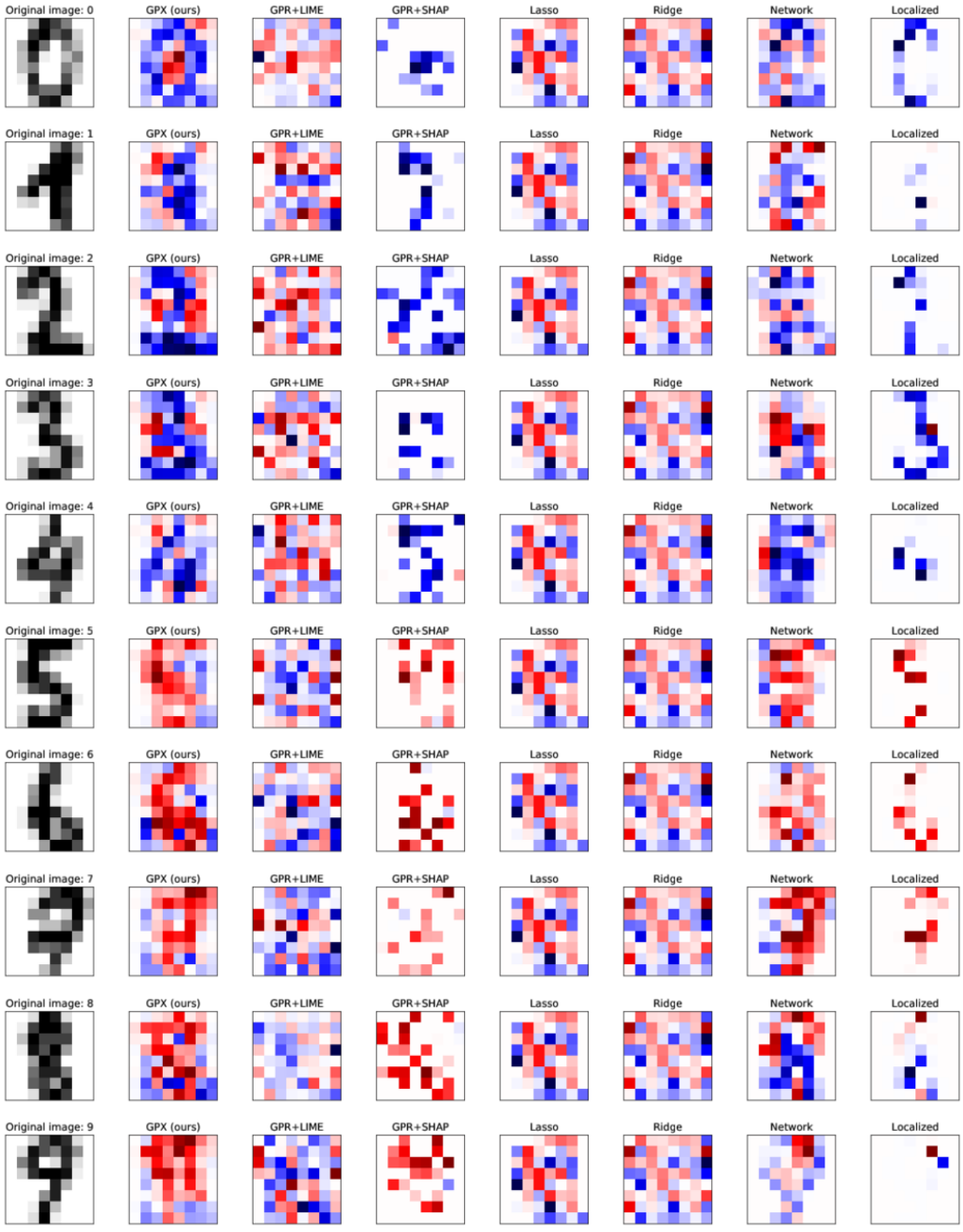


Figure 5: Examples of estimated weights for digits ranging from zero to nine for the Digits dataset. The five upper rows present the weights of samples with digits of zero to four ($y = -1$), whereas the five bottom rows present those for samples with digits from five to nine ($y = 1$). Red and blue denote positive and negative weights, respectively, and their color strengths represent their magnitudes.

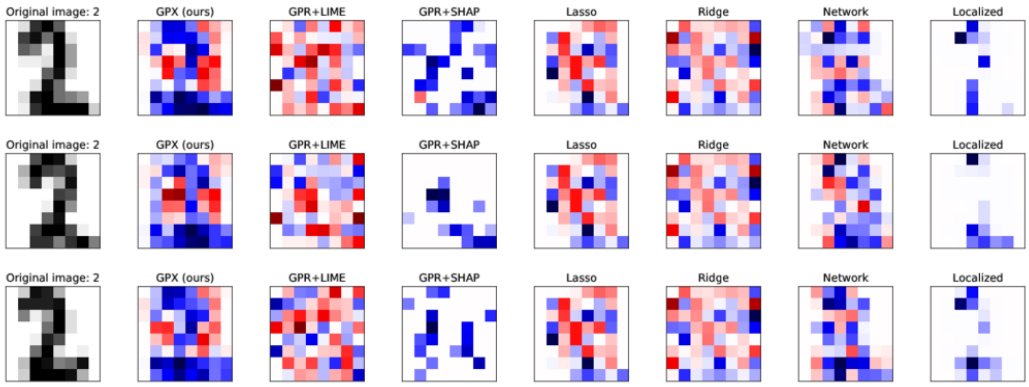


Figure 6: Different examples of estimated weights for digit two for the Digits dataset.