Introduction to Machine Learning

Lecture 5 Non-Linear Regression - Kernel Regression and Gaussian Process

Hongteng Xu

中国人民大學 RENMIN UNIVERSITY OF CHINA

高瓴人工智能学院 Gaoling School of Artificial Intelligence

Additional details of Lecture 4;)

Can We Learn Complicated Models from Sparse Data?

- Overfitting: Model complexity $\gg$ data complexity
  - The number of model parameters is larger than that of data points
  - **Case 1:** The model is wrongly complicated $\Rightarrow$ we need to simplify the model
  - **Case 2:** The model is with reasonable complexity but the data are insufficient $\Rightarrow$ more common, and we need to introduce more side information.

Can We Learn Complicated Models from Sparse Data?

- Overfitting: Model complexity $\gg$ data complexity
  - The number of model parameters is larger than that of data points
  - **Case 1:** The model is wrongly complicated $\Rightarrow$ we need to simplify the model
  - **Case 2:** The model is with reasonable complexity but the data are insufficient $\Rightarrow$ more common, and we need to introduce more side information.
- Underfitting: Model complexity $\ll$ data complexity
  - The number of model parameters is smaller than that of data points
- To learn complicated models from sparse data, we need to impose side information on the model parameters (as **regularizers**)

Lasso: MSE with L1 Regularization

Lasso (Least Absolute Shrinkage and Selection Operator)

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_1 \tag{1}$$

- It is also called "Basis pursuit" in the field of signal processing.
- A Bayesian Viewpoint of Lasso: $\boldsymbol{w} \sim \text{Laplace}(0, b\boldsymbol{I}_D)$, so that $p(\boldsymbol{w}) = \frac{1}{(2b)^D}\exp(-\frac{\|\boldsymbol{w}\|_1}{b})$.
- MAP:

$$\begin{aligned} &\max_{\boldsymbol{w}} p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) \propto \max_{\boldsymbol{w}} p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) \underbrace{p(\boldsymbol{w}|\boldsymbol{X})}_{p(\boldsymbol{w})} \Rightarrow \max_{\boldsymbol{w}} \prod_n p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) p(\boldsymbol{w}) \\ &\Rightarrow \min_{\boldsymbol{w}} -\sum_n \log p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) - \log p(\boldsymbol{w}) \Rightarrow \min_{\boldsymbol{w}} \frac{1}{2\sigma^2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \frac{1}{b}\|\boldsymbol{w}\|_1 + C. \end{aligned} \tag{2}$$

Optimization Methods of Lasso Regression

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_1 \tag{3}$$

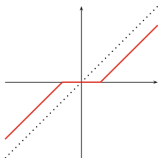
Soft-thresholding: When $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_D] \in \mathbb{R}^{N\times D}$ are orthonormal ($\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{I}_D$):

- The solution of ordinary least squares (OLS) is

$$\hat{\boldsymbol{w}}^{(OLS)} = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 = (\boldsymbol{X}^T\boldsymbol{X})\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{I}_D\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{X}^T\boldsymbol{y}. \tag{4}$$

- The solution of lasso also has a closed form:

$$\hat{w}_d = S_\lambda(\hat{w}_d^{(OLS)}) = \text{sign}(\hat{w}_d^{(OLS)}) \max\{0, |\hat{w}_d^{(OLS)}| - \lambda\}, \quad \forall d = 1, ..., D. \tag{5}$$



Optimization Methods of Lasso Regression

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_1, \quad \text{where } \boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_D] \tag{6}$$

Iterative soft-thresholding for general situations: Although $\boldsymbol{X}^T\boldsymbol{X} \neq \boldsymbol{I}_D$, we can construct orthonormal vectors column-wisely and update parameters iteratively.

- In the t-th iteration, for $d = 1, ..., D$:

$$\begin{aligned}
\hat{w}_d^{(t+1)} &= \arg\min_w \frac{1}{2}\|\boldsymbol{y} - \underbrace{\sum\nolimits_{i\neq d} \boldsymbol{x}_i w_i^{(t)}}_{\boldsymbol{X}_{-d}\boldsymbol{w}_{-d}^{(t)}} - \boldsymbol{x}_d w\|_2^2 + \lambda|w| \\
&= \arg\min_w \frac{1}{2}\left\| \frac{1}{\|\boldsymbol{x}_d\|_2}(\boldsymbol{y} - \boldsymbol{X}_{-d}\boldsymbol{w}_{-d}^{(t)}) - \underbrace{\frac{\boldsymbol{x}_d}{\|\boldsymbol{x}_d\|_2}}_{\text{orthonormal}} w \right\|_2^2 + \frac{\lambda}{\|\boldsymbol{x}_d\|_2^2}|w| \qquad (7) \\
&= S_{\frac{\lambda}{\|\boldsymbol{x}_d\|_2^2}}\left(\frac{\boldsymbol{x}_d^T(\boldsymbol{y} - \boldsymbol{X}_{-d}\boldsymbol{w}_{-d}^{(t)})}{\|\boldsymbol{x}_d\|_2^2} \right)
\end{aligned}$$

Outline

Review

- **Generalized linear model:** Definition, exponential family of distribution.
- **Bias and variance of estimation:** Definitions, their trade-off, the relations to over-fitting and under-fitting.
- **Regularization:** Ridge regression, lasso, their Bayesian interpretations, and optimization methods.
- **MAE-based loss function:** Iteratively reweighted least squares (IRLS)

Outline

Review

- **Generalized linear model:** Definition, exponential family of distribution.
- **Bias and variance of estimation:** Definitions, their trade-off, the relations to over-fitting and under-fitting.
- **Regularization:** Ridge regression, lasso, their Bayesian interpretations, and optimization methods.
- **MAE-based loss function:** Iteratively reweighted least squares (IRLS)

Today

- Non-linear regression, parametric or nonparametric
- Kernel, kernel regression, and representer theorem
- Gaussian process (Optional)

From Linear to Nonlinear Regression

- Linear regression: $y = \boldsymbol{x}^T \boldsymbol{w} + \epsilon$
- Nonlinear regression: $y = f_{\boldsymbol{w}}(\boldsymbol{x}) + \epsilon$

From Linear to Nonlinear Regression

- Linear regression: $y = \boldsymbol{x}^T \boldsymbol{w} + \epsilon$
- Nonlinear regression: $y = f_{\boldsymbol{w}}(\boldsymbol{x}) + \epsilon$

Learning task:

$$\min_{\boldsymbol{w}} \sum_{n=1}^{N} |y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n)|^2 = \min_{\boldsymbol{w}} \|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2. \tag{8}$$

From Linear to Nonlinear Regression

- Linear regression: $y = \boldsymbol{x}^T \boldsymbol{w} + \epsilon$
- Nonlinear regression: $y = f_{\boldsymbol{w}}(\boldsymbol{x}) + \epsilon$

Learning task:

$$\min_{\boldsymbol{w}} \sum_{n=1}^{N} |y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n)|^2 = \min_{\boldsymbol{w}} \|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2. \tag{8}$$

Question:

- Can nonlinear regression models be convex functions?

From Linear to Nonlinear Regression

- Linear regression: $y = \boldsymbol{x}^T \boldsymbol{w} + \epsilon$
- Nonlinear regression: $y = f_{\boldsymbol{w}}(\boldsymbol{x}) + \epsilon$

Learning task:

$$\min_{\boldsymbol{w}} \sum_{n=1}^{N} |y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n)|^2 = \min_{\boldsymbol{w}} \|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2. \tag{8}$$

Question:

- Can nonlinear regression models be convex functions?
- Can neural networks be nonlinear regression models?

From Linear to Nonlinear Regression

- Given the learning task:

$$\min_{\boldsymbol{w}} \underbrace{\|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2}_{L(f_{\boldsymbol{w}})}. \tag{9}$$

- First-order optimization (gradient descent):

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \tau \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}_t})$$

From Linear to Nonlinear Regression

- Given the learning task:

$$\min_{\boldsymbol{w}} \underbrace{\|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2}_{L(f_{\boldsymbol{w}})}. \tag{9}$$

- First-order optimization (gradient descent):

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \tau \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}_t}) \tag{10}$$

- Chain-rule for the gradient of composite function:

$$\nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}}) := \frac{\partial L(f_{\boldsymbol{w}})}{\partial \boldsymbol{w}} = \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial \boldsymbol{w}}$$

From Linear to Nonlinear Regression

- Given the learning task:

$$\min_{\boldsymbol{w}} \underbrace{\|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2}_{L(f_{\boldsymbol{w}})}. \tag{9}$$

- First-order optimization (gradient descent):

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \tau \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}_t}) \tag{10}$$

- Chain-rule for the gradient of composite function:

$$\nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}}) := \frac{\partial L(f_{\boldsymbol{w}})}{\partial \boldsymbol{w}} = \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial \boldsymbol{w}}$$
$$= \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial f_{\boldsymbol{w}}(x_n)} \frac{\partial f_{\boldsymbol{w}}(x_n)}{\partial \boldsymbol{w}}$$

From Linear to Nonlinear Regression

- Given the learning task:

$$\min_{\boldsymbol{w}} \underbrace{\|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2}_{L(f_{\boldsymbol{w}})}. \tag{9}$$

- First-order optimization (gradient descent):

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \tau \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}_t}) \tag{10}$$

- Chain-rule for the gradient of composite function:

$$\begin{aligned} \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}}) &:= \frac{\partial L(f_{\boldsymbol{w}})}{\partial \boldsymbol{w}} = \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial \boldsymbol{w}} \\ &= \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial f_{\boldsymbol{w}}(x_n)} \frac{\partial f_{\boldsymbol{w}}(x_n)}{\partial \boldsymbol{w}} = \sum_{n=1}^{N} 2(f_{\boldsymbol{w}}(\boldsymbol{x}_n) - y_n) \frac{\partial f_{\boldsymbol{w}}(x_n)}{\partial \boldsymbol{w}}. \end{aligned} \tag{11}$$

From Linear to Nonlinear Regression

- Given the learning task:

$$\min_{\boldsymbol{w}} \underbrace{\|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2}_{L(f_{\boldsymbol{w}})}. \tag{9}$$

- First-order optimization (gradient descent):

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \tau \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}_t}) \tag{10}$$

- Chain-rule for the gradient of composite function:

$$\begin{aligned} \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}}) &:= \frac{\partial L(f_{\boldsymbol{w}})}{\partial \boldsymbol{w}} = \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial \boldsymbol{w}} \\ &= \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial f_{\boldsymbol{w}}(x_n)} \frac{\partial f_{\boldsymbol{w}}(x_n)}{\partial \boldsymbol{w}} = \sum_{n=1}^{N} 2(f_{\boldsymbol{w}}(\boldsymbol{x}_n) - y_n) \frac{\partial f_{\boldsymbol{w}}(x_n)}{\partial \boldsymbol{w}}. \end{aligned} \tag{11}$$

- Replacing N with B leads to stochastic gradient descent.

From Linear to Nonlinear Regression

- Given the learning task:

$$\min_{\boldsymbol{w}} \underbrace{\|\boldsymbol{y} - f_{\boldsymbol{w}}(\boldsymbol{X})\|_2^2}_{L(f_{\boldsymbol{w}})}. \tag{9}$$

- First-order optimization (gradient descent):

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \tau \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}_t}) \tag{10}$$

- Chain-rule for the gradient of composite function:

$$\begin{aligned} \nabla_{\boldsymbol{w}} L(f_{\boldsymbol{w}}) &:= \frac{\partial L(f_{\boldsymbol{w}})}{\partial \boldsymbol{w}} = \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial \boldsymbol{w}} \\ &= \sum_{n=1}^{N} \frac{\partial (y_n - f_{\boldsymbol{w}}(\boldsymbol{x}_n))^2}{\partial f_{\boldsymbol{w}}(x_n)} \frac{\partial f_{\boldsymbol{w}}(x_n)}{\partial \boldsymbol{w}} = \sum_{n=1}^{N} 2(f_{\boldsymbol{w}}(\boldsymbol{x}_n) - y_n) \frac{\partial f_{\boldsymbol{w}}(x_n)}{\partial \boldsymbol{w}}. \end{aligned} \tag{11}$$

- Replacing N with B leads to stochastic gradient descent.
- (**Not a standard derivation!**)

Chain Rule of Derivative

- Lagrange's notation for composite function $h(x) = f(g(x))$:

$$\frac{\mathrm{d}h}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}g}\frac{\mathrm{d}g}{\mathrm{d}x} \quad \Leftrightarrow \quad h'(x) = f'(g(x))g'(x).$$

Chain Rule of Derivative

- Lagrange's notation for composite function $h(x) = f(g(x))$:

$$\frac{\mathrm{d}h}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}g}\frac{\mathrm{d}g}{\mathrm{d}x} \quad \Leftrightarrow \quad h'(x) = f'(g(x))g'(x). \tag{12}$$

- Multivariable functions: $f : \mathbb{R}^M \mapsto \mathbb{R}, \quad g : \mathbb{R}^N \mapsto \mathbb{R}^M$

$$y = f(\boldsymbol{u}) = f(g(\boldsymbol{x}))$$

Chain Rule of Derivative

- Lagrange's notation for composite function $h(x) = f(g(x))$:

$$\frac{\mathrm{d}h}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}g}\frac{\mathrm{d}g}{\mathrm{d}x} \quad \Leftrightarrow \quad h'(x) = f'(g(x))g'(x). \tag{12}$$

- Multivariable functions: $f : \mathbb{R}^M \mapsto \mathbb{R}, \quad g : \mathbb{R}^N \mapsto \mathbb{R}^M$

$$y = f(\boldsymbol{u}) = f(g(\boldsymbol{x})) \tag{13}$$

- The chain rule of composite multivariable function:

$$\frac{\partial y}{\partial x_n} = \sum_{m=1}^{M} \frac{\partial y}{\partial u_m}\frac{\partial u_m}{\partial x_n} \quad \Leftrightarrow \quad \frac{\partial y}{\partial x_n} = \langle \nabla_{\boldsymbol{u}} y, \frac{\partial \boldsymbol{u}}{\partial x_n} \rangle, \ \forall m = 1, ..., M, \ n = 1, ..., N.$$

Chain Rule of Derivative

- Lagrange's notation for composite function $h(x) = f(g(x))$:

$$\frac{\mathrm{d}h}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}g}\frac{\mathrm{d}g}{\mathrm{d}x} \quad \Leftrightarrow \quad h'(x) = f'(g(x))g'(x). \tag{12}$$

- Multivariable functions: $f : \mathbb{R}^M \mapsto \mathbb{R}, \quad g : \mathbb{R}^N \mapsto \mathbb{R}^M$

$$y = f(\boldsymbol{u}) = f(g(\boldsymbol{x})) \tag{13}$$

- The chain rule of composite multivariable function:

$$\frac{\partial y}{\partial x_n} = \sum_{m=1}^{M} \frac{\partial y}{\partial u_m}\frac{\partial u_m}{\partial x_n} \quad \Leftrightarrow \quad \frac{\partial y}{\partial x_n} = \langle \nabla_{\boldsymbol{u}} y, \frac{\partial \boldsymbol{u}}{\partial x_n} \rangle, \ \forall m = 1, ..., M, \ n = 1, ..., N. \tag{14}$$

- Why are both $\nabla_{\boldsymbol{u}} y$ and $\frac{\partial \boldsymbol{u}}{\partial x_n}$ column vectors?

Chain Rule of Derivative

- Lagrange's notation for composite function $h(x) = f(g(x))$:

$$\frac{\mathrm{d}h}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}g}\frac{\mathrm{d}g}{\mathrm{d}x} \quad \Leftrightarrow \quad h'(x) = f'(g(x))g'(x). \tag{12}$$

- Multivariable functions: $f : \mathbb{R}^M \mapsto \mathbb{R}, \quad g : \mathbb{R}^N \mapsto \mathbb{R}^M$

$$y = f(\boldsymbol{u}) = f(g(\boldsymbol{x})) \tag{13}$$

- The chain rule of composite multivariable function:

$$\frac{\partial y}{\partial x_n} = \sum_{m=1}^{M} \frac{\partial y}{\partial u_m}\frac{\partial u_m}{\partial x_n} \quad \Leftrightarrow \quad \frac{\partial y}{\partial x_n} = \langle \nabla_{\boldsymbol{u}} y, \frac{\partial \boldsymbol{u}}{\partial x_n} \rangle, \ \forall m = 1, ..., M, \ n = 1, ..., N. \tag{14}$$

- Why are both $\nabla_{\boldsymbol{u}} y$ and $\frac{\partial \boldsymbol{u}}{\partial x_n}$ column vectors? (**Because one is gradient while the other is Jacobian matrix.**)

Gradient, Jacobian Matrix, and Chain Rule

- Given a function $f : \mathbb{R}^M \mapsto \mathbb{R}^N$, i.e., $\boldsymbol{y} = f(\boldsymbol{x})$, where $\boldsymbol{y} = [f_1(\boldsymbol{x}), ..., f_N(\boldsymbol{x})]^T \in \mathbb{R}^N$ and $\boldsymbol{x} = [x_1, ..., x_M]^T \in \mathbb{R}^M$, its **Jacobian matrix** is

$$\boldsymbol{J}_f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \dots & \frac{\partial f_1(\boldsymbol{x})}{\partial x_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_N(\boldsymbol{x})}{\partial x_1} & \dots & \frac{\partial f_N(\boldsymbol{x})}{\partial x_M} \end{bmatrix} \in \mathbb{R}^{N \times M} \tag{15}$$

Gradient, Jacobian Matrix, and Chain Rule

- Given a function $f: \mathbb{R}^M \mapsto \mathbb{R}^N$, i.e., $\boldsymbol{y} = f(\boldsymbol{x})$, where $\boldsymbol{y} = [f_1(\boldsymbol{x}), ..., f_N(\boldsymbol{x})]^T \in \mathbb{R}^N$ and $\boldsymbol{x} = [x_1, ..., x_M]^T \in \mathbb{R}^M$, its **Jacobian matrix** is

$$\boldsymbol{J}_f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\boldsymbol{x})}{\partial x_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_N(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_N(\boldsymbol{x})}{\partial x_M} \end{bmatrix} \in \mathbb{R}^{N \times M} \tag{15}$$

- Given a composite function $h(\boldsymbol{x}) = f(g(\boldsymbol{x}))$, where $g: \mathbb{R}^M \mapsto \mathbb{R}^N$, $f: \mathbb{R}^N \mapsto \mathbb{R}^L$, the **Chain rule of their Jacobian matrix** is

$$\underbrace{\boldsymbol{J}_h(\boldsymbol{x})}_{\in \mathbb{R}^{L \times M}} = \underbrace{\boldsymbol{J}_f(g(\boldsymbol{x}))}_{\in \mathbb{R}^{L \times N}} \underbrace{\boldsymbol{J}_g(\boldsymbol{x})}_{\in \mathbb{R}^{N \times M}}.$$

Gradient, Jacobian Matrix, and Chain Rule

- Given a function $f: \mathbb{R}^M \mapsto \mathbb{R}^N$, i.e., $\boldsymbol{y} = f(\boldsymbol{x})$, where $\boldsymbol{y} = [f_1(\boldsymbol{x}), ..., f_N(\boldsymbol{x})]^T \in \mathbb{R}^N$ and $\boldsymbol{x} = [x_1, ..., x_M]^T \in \mathbb{R}^M$, its **Jacobian matrix** is

$$\boldsymbol{J}_f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\boldsymbol{x})}{\partial x_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_N(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_N(\boldsymbol{x})}{\partial x_M} \end{bmatrix} \in \mathbb{R}^{N \times M} \tag{15}$$

- Given a composite function $h(\boldsymbol{x}) = f(g(\boldsymbol{x}))$, where $g: \mathbb{R}^M \mapsto \mathbb{R}^N$, $f: \mathbb{R}^N \mapsto \mathbb{R}^L$, the **Chain rule of their Jacobian matrix** is

$$\underbrace{\boldsymbol{J}_h(\boldsymbol{x})}_{\in \mathbb{R}^{L \times M}} = \underbrace{\boldsymbol{J}_f(g(\boldsymbol{x}))}_{\in \mathbb{R}^{L \times N}} \underbrace{\boldsymbol{J}_g(\boldsymbol{x})}_{\in \mathbb{R}^{N \times M}}. \tag{16}$$

- **The relation between gradient and Jacobian matrix:**

$$\nabla_{\boldsymbol{x}} f = \boldsymbol{J}_f^T(\boldsymbol{x}) \tag{17}$$

Are models (and their parameters) necessary?

Parametric Model v.s Nonparametric Model

Parametric model

- $x \sim p(X|\theta)$
- The statistics (e.g., mean, variance, high-order moments) are functions of model parameters.

Parametric Model v.s Nonparametric Model

Parametric model

- $x \sim p(X|\theta)$
- The statistics (e.g., mean, variance, high-order moments) are functions of model parameters.
- Sometimes the model parameters themselves are the statistics, while in most cases they are used to construct the statistics (recall GLM).

Parametric Model v.s Nonparametric Model

Parametric model

- $x \sim p(X|\theta)$
- The statistics (e.g., mean, variance, high-order moments) are functions of model parameters.
- Sometimes the model parameters themselves are the statistics, while in most cases they are used to construct the statistics (recall GLM).

Nonparametric model

- Distribution-free: do not rely on assumptions that the data are draw from a given parametric family of probability distributions.
- The statistics (e.g., mean, variance, high-order moments) are defined to be **functions of samples**, no dependency on any parameters.

Parametric Model v.s Nonparametric Model (An Example)

Parametric model

- Linear regression: $y \sim \mathcal{N}(\boldsymbol{x}^T \boldsymbol{w}, \sigma^2)$
- Non-linear regression: $y \sim \mathcal{N}(\boldsymbol{f}_{\boldsymbol{w}}(\boldsymbol{x}), \sigma^2)$

Parametric Model v.s Nonparametric Model (An Example)

Parametric model

- Linear regression: $y \sim \mathcal{N}(\boldsymbol{x}^T \boldsymbol{w}, \sigma^2)$
- Non-linear regression: $y \sim \mathcal{N}(f_{\boldsymbol{w}}(\boldsymbol{x}), \sigma^2)$
- What we care is $\mathbb{E}[Y|X] = g^{-1}(X\beta)$ (Recall GLM).

Parametric Model v.s Nonparametric Model (An Example)

Parametric model

- Linear regression: $y \sim \mathcal{N}(\boldsymbol{x}^T\boldsymbol{w}, \sigma^2)$
- Non-linear regression: $y \sim \mathcal{N}(f_{\boldsymbol{w}}(\boldsymbol{x}), \sigma^2)$
- What we care is $\mathbb{E}[Y|X] = g^{-1}(X\beta)$ (Recall GLM).
- MSE: $\mathbb{E}_{y|\boldsymbol{w},\boldsymbol{x}}[(y - \hat{y})^2]$ (Recall bias and variance).

Parametric Model v.s Nonparametric Model (An Example)

Parametric model

- Linear regression: $y \sim \mathcal{N}(\boldsymbol{x}^T\boldsymbol{w}, \sigma^2)$
- Non-linear regression: $y \sim \mathcal{N}(f_{\boldsymbol{w}}(\boldsymbol{x}), \sigma^2)$
- What we care is $\mathbb{E}[Y|X] = g^{-1}(X\beta)$ (Recall GLM).
- MSE: $\mathbb{E}_{y|\boldsymbol{w},\boldsymbol{x}}[(y - \hat{y})^2]$ (Recall bias and variance).

Nonparametric model

- $\mathbb{E}[Y|X] = f(X)$, without parametric statistics (i.e., β)

Parametric Model v.s Nonparametric Model (An Example)

Parametric model

- Linear regression: $y \sim \mathcal{N}(\boldsymbol{x}^T\boldsymbol{w}, \sigma^2)$
- Non-linear regression: $y \sim \mathcal{N}(f_{\boldsymbol{w}}(\boldsymbol{x}), \sigma^2)$
- What we care is $\mathbb{E}[Y|X] = g^{-1}(X\beta)$ (Recall GLM).
- MSE: $\mathbb{E}_{y|\boldsymbol{w},\boldsymbol{x}}[(y - \hat{y})^2]$ (Recall bias and variance).

Nonparametric model

- $\mathbb{E}[Y|X] = f(X)$, without parametric statistics (i.e., β)
- Typical models: histogram, KNN classification, and kernel regression, semi-parametric regression, ...

Nonparametric Model: Nadaraya–Watson Kernel Regression

- Given $\{x_n, y_n\}_{n=1}^N$, for arbitrary input x, its output y can be estimated by

$$\hat{y} = \hat{f}_h(x) = \frac{\sum_{n=1}^N \kappa_h(x - x_n) y_n}{\sum_{n=1}^N \kappa_h(x - x_n)} \tag{18}$$

- $\kappa_h(x)$ is a kernel function with bandwidth h.

Nonparametric Model: Nadaraya–Watson Kernel Regression

- Given $\{x_n, y_n\}_{n=1}^N$, for arbitrary input x, its output y can be estimated by

$$\hat{y} = \hat{f}_h(x) = \frac{\sum_{n=1}^N \kappa_h(x - x_n) y_n}{\sum_{n=1}^N \kappa_h(x - x_n)} \tag{18}$$

- $\kappa_h(x)$ is a kernel function with bandwidth h.
- Nonparametric model also owns parameters (e.g., h) but the parameters do not determine the statistics (e.g., $\mathbb{E}[Y|X]$, or equivalently, $f(X)$) uniquely.

What Is Kernel Function?

Nonparametric Statistics

- A kernel is a **non-negative real-valued integrable** function
  - Normalization: $\int_{x \in \mathcal{X}} \kappa(x) \mathrm{d}x = 1$
  - Symmetry: $\kappa(x) = \kappa(-x), \forall x \in \mathcal{X}$.

What Is Kernel Function?

Nonparametric Statistics

- A kernel is a **non-negative real-valued integrable** function
  - Normalization: $\int_{x \in \mathcal{X}} \kappa(x) \mathrm{d}x = 1$
  - Symmetry: $\kappa(x) = \kappa(-x), \forall x \in \mathcal{X}$.

Bayesian Statistics (More Generalized)

- Kernel is a function associated with a probability density function, which omits the factors irrelevant to target variable.

What Is Kernel Function?

Nonparametric Statistics

- A kernel is a **non-negative real-valued integrable** function
  - Normalization: $\int_{x \in \mathcal{X}} \kappa(x) \mathrm{d}x = 1$
  - Symmetry: $\kappa(x) = \kappa(-x), \forall x \in \mathcal{X}$.

Bayesian Statistics (More Generalized)

- Kernel is a function associated with a probability density function, which omits the factors irrelevant to target variable.

$$p(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x-\mu)^2}{2\sigma^2}) \quad \Rightarrow \quad \kappa(x) = \exp(-\frac{(x-\mu)^2}{2\sigma^2}), \tag{19}$$

where x is the target variable.

What Is Kernel Function?

Nonparametric Statistics

- A kernel is a **non-negative real-valued integrable** function
  - Normalization: $\int_{x\in\mathcal{X}} \kappa(x)\mathrm{d}x = 1$
  - Symmetry: $\kappa(x) = \kappa(-x), \forall x \in \mathcal{X}$.

Bayesian Statistics (More Generalized)

- Kernel is a function associated with a probability density function, which omits the factors irrelevant to target variable.

$$p(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{(x-\mu)^2}{2\sigma^2}) \quad \Rightarrow \quad \kappa(x) = \exp(-\frac{(x-\mu)^2}{2\sigma^2}), \tag{19}$$

where x is the target variable.

Functional Analysis (Much More Generalized and Insightful)

- A function associated with a **reproducing kernel Hilbert space**

Reproducing Kernel Hilbert Space

- RKHS ($\mathcal{H}$) is a Hilbert space of functions satisfying $\forall f, g \in \mathcal{H}$

$$\|f - g\| \to 0 \quad \Rightarrow \quad |f(x) - g(x)| \to 0, \ \forall x \tag{20}$$

- Here, the commonly-used $L_p$norm of function can be defined defined as

$$\|f\|_{L_p} = \left(\int_{x \in \mathcal{X}} |f(x)|^p \mathrm{d}x \right)^{1/p}$$

Reproducing Kernel Hilbert Space

- RKHS ($\mathcal{H}$) is a Hilbert space of functions satisfying $\forall f, g \in \mathcal{H}$

$$\|f - g\| \to 0 \quad \Rightarrow \quad |f(x) - g(x)| \to 0, \ \forall x \tag{20}$$

- Here, the commonly-used $L_p$norm of function can be defined defined as

$$\|f\|_{L_p} = \left(\int_{x \in \mathcal{X}} |f(x)|^p \mathrm{d}x \right)^{1/p} \tag{21}$$

- **Reproducing kernel of $\mathcal{H}$:** An RKHS $\mathcal{H}$ is associated with a kernel that reproduces every function in the space: Continuous evaluation functional on $\mathcal{H}$: $L_x : f \mapsto f(x), \ \forall f \in \mathcal{H}$

$$f(x) := L_x(f) = \langle f, K_x \rangle_{\mathcal{H}}$$

Reproducing Kernel Hilbert Space

- RKHS ($\mathcal{H}$) is a Hilbert space of functions satisfying $\forall f, g \in \mathcal{H}$

$$\|f - g\| \to 0 \quad \Rightarrow \quad |f(x) - g(x)| \to 0, \ \forall x \tag{20}$$

- Here, the commonly-used $L_p$norm of function can be defined defined as

$$\|f\|_{L_p} = \left(\int_{x \in \mathcal{X}} |f(x)|^p \mathrm{d}x \right)^{1/p} \tag{21}$$

- **Reproducing kernel of $\mathcal{H}$:** An RKHS $\mathcal{H}$ is associated with a kernel that reproduces every function in the space: Continuous evaluation functional on $\mathcal{H}$: $L_x : f \mapsto f(x), \ \forall f \in \mathcal{H}$

$$f(x) := L_x(f) = \langle f, K_x \rangle_{\mathcal{H}} = \int_{x' \in \mathcal{X}} f(x') \bar{K}_x(x') \mathrm{d}x'.$$

Reproducing Kernel Hilbert Space

- RKHS ($\mathcal{H}$) is a Hilbert space of functions satisfying $\forall f, g \in \mathcal{H}$

$$\|f - g\| \to 0 \quad \Rightarrow \quad |f(x) - g(x)| \to 0, \ \forall x \tag{20}$$

- Here, the commonly-used $L_p$norm of function can be defined defined as

$$\|f\|_{L_p} = \left(\int_{x \in \mathcal{X}} |f(x)|^p \mathrm{d}x \right)^{1/p} \tag{21}$$

- **Reproducing kernel of $\mathcal{H}$:** An RKHS $\mathcal{H}$ is associated with a kernel that reproduces every function in the space: Continuous evaluation functional on $\mathcal{H}$: $L_x : f \mapsto f(x), \ \forall f \in \mathcal{H}$

$$f(x) := L_x(f) = \langle f, K_x \rangle_{\mathcal{H}} = \int_{x' \in \mathcal{X}} f(x') \bar{K}_x(x') \mathrm{d}x'. \tag{22}$$

- The reproducing kernel of $\mathcal{H}$ is a function $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$

$$K(x, y) = \langle K_x, K_y \rangle_{\mathcal{H}} =? \text{ (Derive it)} \tag{23}$$

Domain, Hilbert Space, and RKHS

- Sample space $\mathcal{X}$, works as the domain of functions
- $f : \mathcal{X} \mapsto \mathbb{R}$, a function defined on $\mathcal{X}$.

Domain, Hilbert Space, and RKHS

- Sample space $\mathcal{X}$, works as the domain of functions
- $f : \mathcal{X} \mapsto \mathbb{R}$, a function defined on $\mathcal{X}$.
- Hilbert space $\mathcal{H}$ is the space of all valid functions on $\mathcal{X}$, *i.e.*, $\mathcal{H} = \{f | f : \mathcal{X} \mapsto \mathbb{R}\}$.

Domain, Hilbert Space, and RKHS

- Sample space $\mathcal{X}$, works as the domain of functions
- $f : \mathcal{X} \mapsto \mathbb{R}$, a function defined on $\mathcal{X}$.
- Hilbert space $\mathcal{H}$ is the space of all valid functions on $\mathcal{X}$, *i.e.*, $\mathcal{H} = \{f | f : \mathcal{X} \mapsto \mathbb{R}\}$.
- When the Hilbert space is associated with a reproducing kernel K, such that

$$f(x) := L_x(f) = \langle f, K_x \rangle_{\mathcal{H}}, \ \forall f \in \mathcal{H}, \ x \in \mathcal{X}, \tag{24}$$

Then, it becomes a RKHS

Domain, Hilbert Space, and RKHS

- **Why is the kernel $K(x, y)$ called "Reproducing kernel"?**

Domain, Hilbert Space, and RKHS

- **Why is the kernel $K(x, y)$ called "Reproducing kernel"?**
- Because its value is reproduced by the inner product of its marginal functions in the Hilbert space — Recall its definition:

$$K: \ \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$$

$$K_x: \ \mathcal{X} \mapsto \mathbb{R} \quad (\text{A marginal of } K \text{ at } x, \text{ i.e., } K_x = K(x, :).)$$

$$K(x, y) = \langle K_x, K_y \rangle_{\mathcal{H}}$$

Revisit (Nadaraya-Watson) Kernel Regression

- The data we observed are $\{x_n, y_n\}_{n=1}^N$
- The model $f(x) : \mathcal{X} \mapsto \mathcal{Y}$ is a sample in a RKHS $\mathcal{H}$.

Revisit (Nadaraya-Watson) Kernel Regression

- The data we observed are $\{x_n, y_n\}_{n=1}^N$
- The model $f(x) : \mathcal{X} \mapsto \mathcal{Y}$ is a sample in a RKHS $\mathcal{H}$.
- Therefore, for each point $x \in \mathcal{X}$, we have

$$f(x) = \langle f, K_x \rangle_{\mathcal{H}} = \int_{x' \in \mathcal{X}} f(x') \bar{K}_x(x') \mathrm{d}x'$$

Revisit (Nadaraya-Watson) Kernel Regression

- The data we observed are $\{x_n, y_n\}_{n=1}^N$
- The model $f(x) : \mathcal{X} \mapsto \mathcal{Y}$ is a sample in a RKHS $\mathcal{H}$.
- Therefore, for each point $x \in \mathcal{X}$, we have

$$
\begin{aligned}
f(x) = \langle f, K_x \rangle_{\mathcal{H}} &= \int_{x' \in \mathcal{X}} f(x') \bar{K}_x(x') \mathrm{d}x' \\
&\approx \sum_{n=1}^{N} f(x_n) K_x(x_n)
\end{aligned}
$$

Revisit (Nadaraya-Watson) Kernel Regression

- The data we observed are $\{x_n, y_n\}_{n=1}^N$
- The model $f(x) : \mathcal{X} \mapsto \mathcal{Y}$ is a sample in a RKHS $\mathcal{H}$.
- Therefore, for each point $x \in \mathcal{X}$, we have

$$
\begin{aligned}
f(x) = \langle f, K_x \rangle_{\mathcal{H}} &= \int_{x' \in \mathcal{X}} f(x') \bar{K}_x(x') \mathrm{d}x' \\
&\approx \sum_{n=1}^{N} f(x_n) K_x(x_n) = \sum_{n=1}^{N} f(x_n) \frac{\kappa_h(x - x_n)}{\sum_{i=1}^{N} \kappa_h(x - x_i)} = \hat{f}(x)
\end{aligned}
\tag{25}
$$

where $y_n = f(x_n)$ and $K_x(x_n) := \frac{\kappa_h(x - x_n)}{\sum_{i=1}^{N} \kappa_h(x - x_i)} \propto \kappa_h(x - x_n)$.

Revisit (Nadaraya-Watson) Kernel Regression

- The data we observed are $\{x_n, y_n\}_{n=1}^N$
- The model $f(x) : \mathcal{X} \mapsto \mathcal{Y}$ is a sample in a RKHS $\mathcal{H}$.
- Therefore, for each point $x \in \mathcal{X}$, we have

$$
\begin{aligned}
f(x) = \langle f, K_x \rangle_{\mathcal{H}} &= \int_{x' \in \mathcal{X}} f(x') \bar{K}_x(x') \mathrm{d}x' \\
&\approx \sum_{n=1}^{N} f(x_n) K_x(x_n) = \sum_{n=1}^{N} f(x_n) \frac{\kappa_h(x - x_n)}{\sum_{i=1}^{N} \kappa_h(x - x_i)} = \hat{f}(x)
\end{aligned}
\tag{25}
$$

where $y_n = f(x_n)$ and $K_x(x_n) := \frac{\kappa_h(x - x_n)}{\sum_{i=1}^{N} \kappa_h(x - x_i)} \propto \kappa_h(x - x_n)$.

Why does the approximation based on finite points work?

The Rationality of Kernel Method: Representer Theorem

- **Representer Theorem:** A minimizer f^* of a regularized empirical risk functional defined over a reproducing kernel Hilbert space can be represented as a finite linear combination of kernel products evaluated on the input points in the training set data.

The Rationality of Kernel Method: Representer Theorem

- **Representer Theorem:** A minimizer f^* of a regularized empirical risk functional defined over a reproducing kernel Hilbert space can be represented as a finite linear combination of kernel products evaluated on the input points in the training set data.
- Mathematical statement:

$$f^* := \arg\min_{f \in \mathcal{H}_K} \mathbb{E}_{x,y \sim P_{\mathcal{D}}}[\text{loss}(y, f(x))] + \mathcal{R}(f)$$

The Rationality of Kernel Method: Representer Theorem

- **Representer Theorem:** A minimizer f^* of a regularized empirical risk functional defined over a reproducing kernel Hilbert space can be represented as a finite linear combination of kernel products evaluated on the input points in the training set data.
- Mathematical statement:

$$\begin{aligned} f^* &:= \arg\min_{f \in \mathcal{H}_K} \mathbb{E}_{x,y \sim P_{\mathcal{D}}}[\text{loss}(y, f(x))] + \mathcal{R}(f) \\ &\Leftrightarrow \exists\, \boldsymbol{\alpha} \in \mathbb{R}^M, \text{ such that } f^*(x) = \sum_{n=1}^{M} \alpha_n K(x, x_n),\ M \leq |\mathcal{D}|. \end{aligned} \tag{26}$$

The Rationality of Kernel Method: Representer Theorem

- **Representer Theorem:** A minimizer f^* of a regularized empirical risk functional defined over a reproducing kernel Hilbert space can be represented as a finite linear combination of kernel products evaluated on the input points in the training set data.
- Mathematical statement:

$$\begin{aligned} f^* &:= \arg\min_{f \in \mathcal{H}_K} \mathbb{E}_{x,y \sim P_{\mathcal{D}}}[\text{loss}(y, f(x))] + \mathcal{R}(f) \\ &\Leftrightarrow \exists\, \boldsymbol{\alpha} \in \mathbb{R}^M, \text{ such that } f^*(x) = \sum_{n=1}^{M} \alpha_n K(x, x_n),\ M \leq |\mathcal{D}|. \end{aligned} \tag{26}$$

What are the conditions of a valid kernel?

Valid Kernel Functions

- The condition of a valid reproducing kernel function K: **positive definite**.

Valid Kernel Functions

- The condition of a valid reproducing kernel function K: **positive definite**.
- **Mercer's Theorem:** for all square-integrable function $g(x)$,

$$\iint g(x)K(x,x')g(x')\mathrm{d}x\mathrm{d}x' \geq 0 \tag{27}$$

Valid Kernel Functions

- The condition of a valid reproducing kernel function K: **positive definite**.
- **Mercer's Theorem:** for all square-integrable function $g(x)$,

$$\iint g(x)K(x,x')g(x')\mathrm{d}x\mathrm{d}x' \geq 0 \tag{27}$$

- **Equivalent statement:** The Gram matrix $\boldsymbol{K} = [K(x_i, x_j)] \in \mathbb{R}^{N\times N}$ is positive definite.

Valid Kernel Functions

- The condition of a valid reproducing kernel function K: **positive definite**.
- **Mercer's Theorem:** for all square-integrable function $g(x)$,

$$\iint g(x)K(x,x')g(x')\mathrm{d}x\mathrm{d}x' \geq 0 \tag{27}$$

- **Equivalent statement:** The Gram matrix $\boldsymbol{K} = [K(x_i, x_j)] \in \mathbb{R}^{N\times N}$ is positive definite.
- Can a positive definite matrix be low-rank?

Valid Kernel Functions

- The condition of a valid reproducing kernel function K: **positive definite**.
- **Mercer's Theorem:** for all square-integrable function $g(x)$,

$$\iint g(x)K(x,x')g(x')\mathrm{d}x\mathrm{d}x' \geq 0 \tag{27}$$

- **Equivalent statement:** The Gram matrix $\boldsymbol{K} = [K(x_i, x_j)] \in \mathbb{R}^{N\times N}$ is positive definite.
- Can a positive definite matrix be low-rank? asymmetric?

Valid Kernel Functions

- The condition of a valid reproducing kernel function K: **positive definite**.
- **Mercer's Theorem:** for all square-integrable function $g(x)$,

$$\iint g(x)K(x,x')g(x')\mathrm{d}x\mathrm{d}x' \geq 0 \tag{27}$$

- **Equivalent statement:** The Gram matrix $\boldsymbol{K} = [K(x_i, x_j)] \in \mathbb{R}^{N\times N}$ is positive definite.
- Can a positive definite matrix be low-rank? asymmetric?
- Recall that $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$

$$K(x,y) = \langle K_x, K_y \rangle_{\mathcal{H}}$$

Valid Kernel Functions

- The condition of a valid reproducing kernel function K: **positive definite**.
- **Mercer's Theorem:** for all square-integrable function $g(x)$,

$$\iint g(x)K(x,x')g(x')\mathrm{d}x\mathrm{d}x' \geq 0 \tag{27}$$

- **Equivalent statement:** The Gram matrix $\boldsymbol{K} = [K(x_i, x_j)] \in \mathbb{R}^{N\times N}$ is positive definite.
- Can a positive definite matrix be low-rank? asymmetric?
- Recall that $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$

$$K(x,y) = \langle K_x, K_y\rangle_{\mathcal{H}} = \langle \phi(x), \phi(y)\rangle_{\mathcal{F}} \tag{28}$$

where $\phi : \mathcal{X} \mapsto \mathcal{F}$ maps samples to the (maybe infinite-dimensional) feature space (another Hilbert space).

Commonly-used Kernels

- Polynomial kernel

$$K(x, y) = (x^T y + c)^d.$$

Commonly-used Kernels

- Polynomial kernel

$$K(x, y) = (x^T y + c)^d. \tag{29}$$

when $d = 2$, we have $\phi(x) = [\{x_i^2\}_{i=1}^n, \{\sqrt{2}x_i x_j\}_{i=2,j=1}^{n,i-1}, \{\sqrt{2c}x_i\}_{i=1}^n, c]$.

Commonly-used Kernels

- Polynomial kernel

$$K(x, y) = (x^T y + c)^d. \tag{29}$$

  when $d = 2$, we have $\phi(x) = [\{x_i^2\}_{i=1}^n, \{\sqrt{2}x_i x_j\}_{i=2,j=1}^{n,i-1}, \{\sqrt{2c}x_i\}_{i=1}^n, c]$.

- Radial basis function (RBF) kernel

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{30}$$

  In this case, ϕ leads to an infinite-dimensional space $\mathcal{V}$ (based on the Taylor expansion of $\exp(x^T y)$.)

Commonly-used Kernels

- Polynomial kernel

$$K(x, y) = (x^T y + c)^d. \tag{29}$$

  when $d = 2$, we have $\phi(x) = [\{x_i^2\}_{i=1}^n, \{\sqrt{2}x_i x_j\}_{i=2,j=1}^{n,i-1}, \{\sqrt{2c}x_i\}_{i=1}^n, c]$.

- Radial basis function (RBF) kernel

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{30}$$

  In this case, ϕ leads to an infinite-dimensional space $\mathcal{V}$ (based on the Taylor expansion of $\exp(x^T y)$.)

- If K_1 and K_2 are valid kernel functions:
  1. $a_1 K_1 + a_2 K_2$, where $a_1, a_2 > 0$;

Commonly-used Kernels

- Polynomial kernel

$$K(x, y) = (x^T y + c)^d. \tag{29}$$

  when $d = 2$, we have $\phi(x) = [\{x_i^2\}_{i=1}^n, \{\sqrt{2}x_i x_j\}_{i=2,j=1}^{n,i-1}, \{\sqrt{2c}x_i\}_{i=1}^n, c]$.

- Radial basis function (RBF) kernel

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{30}$$

  In this case, ϕ leads to an infinite-dimensional space $\mathcal{V}$ (based on the Taylor expansion of $\exp(x^T y)$.)

- If K_1 and K_2 are valid kernel functions:
  1. $a_1 K_1 + a_2 K_2$, where $a_1, a_2 > 0$; $k_1 \cdot k_2$

Commonly-used Kernels

- Polynomial kernel

$$K(x, y) = (x^T y + c)^d. \tag{29}$$

  when $d = 2$, we have $\phi(x) = [\{x_i^2\}_{i=1}^n, \{\sqrt{2}x_i x_j\}_{i=2,j=1}^{n,i-1}, \{\sqrt{2c}x_i\}_{i=1}^n, c]$.

- Radial basis function (RBF) kernel

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{30}$$

  In this case, ϕ leads to an infinite-dimensional space $\mathcal{V}$ (based on the Taylor expansion of $\exp(x^T y)$.)

- If K_1 and K_2 are valid kernel functions:
  1. $a_1 K_1 + a_2 K_2$, where $a_1, a_2 > 0$; $k_1 \cdot k_2$
  2. $poly(K_1)$, $\exp(K_1)$

Commonly-used Kernels

- Polynomial kernel

$$K(x, y) = (x^T y + c)^d. \tag{29}$$

  when $d = 2$, we have $\phi(x) = [\{x_i^2\}_{i=1}^n, \{\sqrt{2}x_i x_j\}_{i=2,j=1}^{n,i-1}, \{\sqrt{2c}x_i\}_{i=1}^n, c]$.

- Radial basis function (RBF) kernel

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{30}$$

  In this case, ϕ leads to an infinite-dimensional space $\mathcal{V}$ (based on the Taylor expansion of $\exp(x^T y)$.)

- If K_1 and K_2 are valid kernel functions:
  1. $a_1 K_1 + a_2 K_2$, where $a_1, a_2 > 0$; $k_1 \cdot k_2$
  2. $poly(K_1)$, $\exp(K_1)$
  3. $K_3(x, x') = K_1(\phi(x), \phi(x'))$, $K_3(x, x') = f(x)K_1(x, x')f(x')$.

Revisit Nonparametric/Bayesian Kernel from Functional Analysis

Consider the RBF kernel

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{31}$$

A Viewpoint of Nonparametric Statistics

- A kernel is a **non-negative real-valued integrable** function
  - Normalization: $\int_{x \in \mathcal{X}} \kappa(x) \mathrm{d}x = 1$
  - Symmetry: $\kappa(x) = \kappa(-x), \forall x \in \mathcal{X}$.
- $\kappa(x - y) := K(x, y)$.

Revisit Nonparametric/Bayesian Kernel from Functional Analysis

Consider the RBF kernel

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{31}$$

A Viewpoint of Nonparametric Statistics

- A kernel is a **non-negative real-valued integrable** function
  - Normalization: $\int_{x \in \mathcal{X}} \kappa(x) \mathrm{d}x = 1$
  - Symmetry: $\kappa(x) = \kappa(-x), \forall x \in \mathcal{X}$.
- $\kappa(x - y) := K(x, y)$.

A Viewpoint of Bayesian Statistics

- We have

$$p(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x-\mu)^2}{2\sigma^2}) \quad \Rightarrow \quad \kappa(x) = \exp(-\frac{(x-\mu)^2}{2\sigma^2}), \tag{32}$$

- $\kappa(x) := K(x, \mu)$.

Kernel Method: Kernel Ridge Regression

- **Model:** $y = f(\boldsymbol{x}) + \epsilon, \epsilon \sim \mathcal{N}(0, \sigma^2), f \in \mathcal{H}_K$.

Kernel Method: Kernel Ridge Regression

- **Model:** $y = f(\boldsymbol{x}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, $f \in \mathcal{H}_K$.
- **Learning:** Given $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$

$$f^* = \arg\min_{f \in \mathcal{H}_K} \sum_{n=1}^{N} (y_n - f(\boldsymbol{x}_n))^2 + \lambda \underbrace{\|f\|_{\mathcal{H}}^2}_{\langle f, f \rangle_{\mathcal{H}}}.$$

Kernel Method: Kernel Ridge Regression

- **Model:** $y = f(\boldsymbol{x}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, $f \in \mathcal{H}_K$.
- **Learning:** Given $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$

$$f^* = \arg\min_{f \in \mathcal{H}_K} \sum_{n=1}^{N} (y_n - f(\boldsymbol{x}_n))^2 + \lambda \underbrace{\|f\|_{\mathcal{H}}^2}_{\langle f, f \rangle_{\mathcal{H}}}. \tag{33}$$

- **Representer Theorem:** The optimal solution must be in the following form:

$$f^*(x) = \sum_{n=1}^{N} \alpha_n K(x, x_n).$$

Kernel Method: Kernel Ridge Regression

- **Model:** $y = f(\boldsymbol{x}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, $f \in \mathcal{H}_K$.
- **Learning:** Given $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$

$$f^* = \arg\min_{f \in \mathcal{H}_K} \sum_{n=1}^{N} (y_n - f(\boldsymbol{x}_n))^2 + \lambda \underbrace{\|f\|_{\mathcal{H}}^2}_{\langle f, f \rangle_{\mathcal{H}}}. \tag{33}$$

- **Representer Theorem:** The optimal solution must be in the following form:

$$f^*(x) = \sum_{n=1}^{N} \alpha_n K(x, x_n). \tag{34}$$

- Replacing the f in (33) with (34), we have

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \|\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha}\|_2^2 + \lambda \boldsymbol{\alpha}^T \boldsymbol{K} \boldsymbol{\alpha}, \tag{35}$$

where $\boldsymbol{\alpha} = [\alpha_n]$, the Gram matrix $\boldsymbol{K} = [K(x_n, x_{n'})] \in \mathbb{R}^{N \times N}$.

Kernel Method: Kernel Ridge Regression

- **Model:** $y = f(\boldsymbol{x}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, $f \in \mathcal{H}_K$.
- **Learning:** Given $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$

$$f^* = \arg\min_{f \in \mathcal{H}_K} \sum_{n=1}^{N} (y_n - f(\boldsymbol{x}_n))^2 + \lambda \underbrace{\|f\|_{\mathcal{H}}^2}_{\langle f, f \rangle_{\mathcal{H}}}. \tag{33}$$

- **Representer Theorem:** The optimal solution must be in the following form:

$$f^*(x) = \sum_{n=1}^{N} \alpha_n K(x, x_n). \tag{34}$$

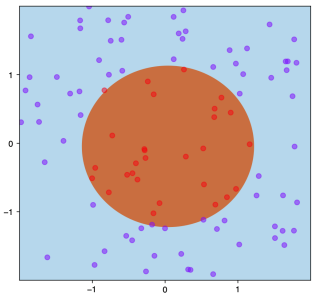
- Replacing the f in (33) with (34), we have

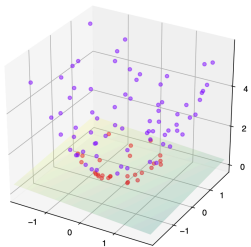
$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \|\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha}\|_2^2 + \lambda \boldsymbol{\alpha}^T \boldsymbol{K} \boldsymbol{\alpha}, \tag{35}$$

where $\boldsymbol{\alpha} = [\alpha_n]$, the Gram matrix $\boldsymbol{K} = [K(x_n, x_{n'})] \in \mathbb{R}^{N \times N}$. (When is KRR a linear regression?)

Kernel Method: From Finite Dimension to Infinite Dimension

- The power of kernel method: convert a low-dimensional linearly-inseparable problem to a high-dimensional linearly-separable problem.





Gaussian Process (GP)

- A time continuous stochastic process $\{X_t\}_{t\in\mathcal{T}}$ is Gaussian iff for every finite set of indices $\{t_n \in \mathcal{T}\}_{n=1}^{N}$, we have

$$\boldsymbol{X}_{t_1,..,t_N} = [X_{t_1}, ..., X_{t_N}]^T \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Gaussian Process (GP)

- A time continuous stochastic process $\{X_t\}_{t\in\mathcal{T}}$ is Gaussian iff for every finite set of indices $\{t_n \in \mathcal{T}\}_{n=1}^N$, we have

$$\boldsymbol{X}_{t_1,..,t_N} = [X_{t_1}, ..., X_{t_N}]^T \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{36}$$

- **Equivalent condition:** $\forall\{t_n \in \mathcal{T}\}_{n=1}^N$ and $\forall \boldsymbol{w} \in \mathbb{R}^N$, there always exists μ and σ^2, such that

$$\langle \boldsymbol{X}, \boldsymbol{w} \rangle \sim \mathcal{N}(\mu, \sigma^2) \tag{37}$$

GP in Machine Learning: Gaussian Process Regression

- “Kriging” in Statistics, interpolation governed by prior covariance.

GP in Machine Learning: Gaussian Process Regression

- "Kriging" in Statistics, interpolation governed by prior covariance.
- **Task:** Given $\{x_n\}_{n=1}^N$, predict a Gaussian process $f(x)$ for $x \in \mathcal{X}$.

GP in Machine Learning: Gaussian Process Regression

- "Kriging" in Statistics, interpolation governed by prior covariance.
- **Task:** Given $\{x_n\}_{n=1}^N$, predict a Gaussian process $f(x)$ for $x \in \mathcal{X}$.
- **Assumption:** the observed vector $\boldsymbol{f} = [f(x_n)] \in \mathbb{R}^N$ is just one sample from a multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}_N, \boldsymbol{\Sigma}_{N\times N})$.

GP in Machine Learning: Gaussian Process Regression

- "Kriging" in Statistics, interpolation governed by prior covariance.
- **Task:** Given $\{x_n\}_{n=1}^N$, predict a Gaussian process $f(x)$ for $x \in \mathcal{X}$.
- **Assumption:** the observed vector $\boldsymbol{f} = [f(x_n)] \in \mathbb{R}^N$ is just one sample from a multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}_N, \boldsymbol{\Sigma}_{N\times N})$.
- **Principle: Kernelized Covariance Matrix.** for $(x, x') \in \mathcal{X} \times \mathcal{X}$,

$$\boldsymbol{f} = [f(x)] \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}(x, x'; \theta)), \tag{38}$$

where $\boldsymbol{K}(x, x'; \theta)$ is the covariance matrix between all possible pairs (x, x').

GP in Machine Learning: Gaussian Process Regression

- **Learning: MLE** $\max_\theta \log p(f(x')|x,\theta)$

$$\log p(f(x')|x,\theta) = -\frac{1}{2}\boldsymbol{f}^T\boldsymbol{K}(x,x';\theta)^{-1}\boldsymbol{f} - \frac{1}{2}\log\det(\boldsymbol{K}(x,x';\theta)) - \frac{N}{2}\log 2\pi. \quad (39)$$

- Derive it [Hint: PDF of multivariate normal distribution:
$p(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \det(2\pi\boldsymbol{\Sigma})^{-\frac{1}{2}}\exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))$]

GP in Machine Learning: Gaussian Process Regression

- **Learning: MLE** $\max_\theta \log p(f(x')|x,\theta)$

$$\log p(f(x')|x,\theta) = -\frac{1}{2}\boldsymbol{f}^T\boldsymbol{K}(x,x';\theta)^{-1}\boldsymbol{f} - \frac{1}{2}\log\det(\boldsymbol{K}(x,x';\theta)) - \frac{N}{2}\log 2\pi. \tag{39}$$

- Derive it [Hint: PDF of multivariate normal distribution:
  $p(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \det(2\pi\boldsymbol{\Sigma})^{-\frac{1}{2}}\exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))$]
- **Prediction: Kernel Regression** Given $\hat{\theta}$, for new points x^*, we predict $f(x^*)$ via drawing samples from the predictive distribution
  $p(y^*|x^*,f(x),x) = \mathcal{N}(y^*|\boldsymbol{a},\boldsymbol{B})$

$$\begin{aligned}&\text{Posterior mean: } \boldsymbol{a} = \boldsymbol{K}(x^*,x;\hat{\theta})\boldsymbol{K}^{-1}(x,x';\hat{\theta})\boldsymbol{f}\\ &\text{Posterior variance: } \boldsymbol{B} = \boldsymbol{K}(x^*,x^*;\hat{\theta}) - \boldsymbol{K}(x^*,x;\hat{\theta})\boldsymbol{K}^{-1}(x,x';\hat{\theta})\boldsymbol{K}^T(x^*,x;\hat{\theta}).\end{aligned} \tag{40}$$

- Note: x^* may contain multiple data points.

GP in Machine Learning: Gaussian Process Regression

- **Learning: MLE** $\max_\theta \log p(f(x')|x,\theta)$

$$\log p(f(x')|x,\theta) = -\frac{1}{2}\boldsymbol{f}^T\boldsymbol{K}(x,x';\theta)^{-1}\boldsymbol{f} - \frac{1}{2}\log\det(\boldsymbol{K}(x,x';\theta)) - \frac{N}{2}\log 2\pi. \quad (39)$$

- Derive it [Hint: PDF of multivariate normal distribution:
  $p(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \det(2\pi\boldsymbol{\Sigma})^{-\frac{1}{2}}\exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))$]
- **Prediction: Kernel Regression** Given $\hat{\theta}$, for new points x^*, we predict $f(x^*)$ via drawing samples from the predictive distribution
  $p(y^*|x^*,f(x),x) = \mathcal{N}(y^*|\boldsymbol{a},\boldsymbol{B})$

$$\begin{aligned}&\text{Posterior mean: } \boldsymbol{a} = \boldsymbol{K}(x^*,x;\hat{\theta})\boldsymbol{K}^{-1}(x,x';\hat{\theta})\boldsymbol{f}\\ &\text{Posterior variance: } \boldsymbol{B} = \boldsymbol{K}(x^*,x^*;\hat{\theta}) - \boldsymbol{K}(x^*,x;\hat{\theta})\boldsymbol{K}^{-1}(x,x';\hat{\theta})\boldsymbol{K}^T(x^*,x;\hat{\theta}).\end{aligned} \quad (40)$$

- Note: x^* may contain multiple data points.
- **Enumerate the limitations of GP regression.**

In Summary

- Basis representation and representer theorem
- Dual form of linear regression
- Basic concepts of kernel function
- Gaussian process

Next...

- Unsuperivsed learning (Dimensionality reduction and clustering)
- Linear dimensionality reduction: Principal component analysis (PCA)

Homework 2, DDL: April 2, 2022

Python Programming

1. Lab # 3 (4 Pts, Done)
2. Lab # 4 (4 Pts)

Questions for Tech Report (6 Pts, $\leq$ 3 Pages)

1. Derive the closed form solution of generalized Tikhonov regularizer (1 Pts)

$$\min_w \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_{\boldsymbol{P}}^2 + \lambda\|\boldsymbol{w} - \boldsymbol{w}_0\|_{\boldsymbol{Q}}^2, \text{ where } \boldsymbol{P}, \boldsymbol{Q} \text{ are positive definite.} \tag{41}$$

2. Assume $y \sim \mathcal{N}(\boldsymbol{x}^T\boldsymbol{w}, \sigma^2)$, prior $p(\boldsymbol{w}) = \frac{1}{(2b)^D}\exp(-\frac{\|\boldsymbol{w}-\boldsymbol{\mu}\|_1}{b})$. Given $\{y_n, \boldsymbol{x}_n\}_{n=1}^N$, derive $p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})$ and the algorithm solving $\max \log p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})$. (2 Pts)
3. Write down your derivation from (33) to (35), and derive the chain rule of $\frac{\partial L}{\partial h}$ in the case using the RBF kernel $K(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \exp(-\|\boldsymbol{x}_n - \boldsymbol{x}_{n'}\|_2^2/h)$. (2 Pts)
4. When using the linear kernel $K(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) = \boldsymbol{x}_n^T\boldsymbol{x}_{n'}$, what is the connection between KRR and ridge regression? (1 Pts)