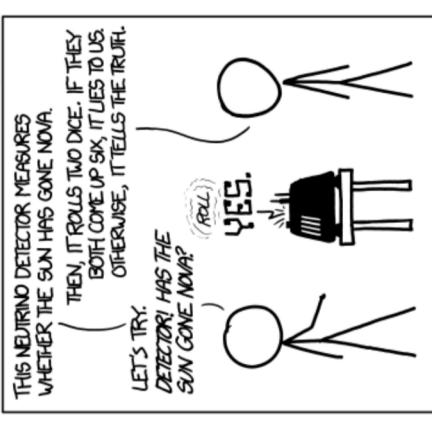


# Frequentist vs Bayesian

DID THE SUN JUST EXPLODE?  
(IT'S NIGHT, SO WE'RE NOT SURE.)



FREQUENTIST STATISTICIAN:

THE PROBABILITY OF THIS RESULT HAPPENING BY CHANCE IS  $\frac{1}{36} = 0.027$ . SINCE  $P < 0.05$ , I CONCLUDE THAT THE SUN HAS EXPLODED.

BAYESIAN STATISTICIAN:

BET YOU \$50  
IT HASN'T.

- We are often interested in learning probabilistic models of a given dataset
  - a probabilistic model describes a probabilistic process that generates the data
  - e.g. Gaussian distributions form a class of probabilistic model for real-valued observations
- The frequentist approach picks a probabilistic model that best fits the dataset
  - e.g., naive Bayes classifier
- The Bayesian approach assigns a weight to each candidate probabilistic model by using the Bayes' rule to combine
  - prior subjective assessment on how likely the model is, and
  - how well the model explains the dataset.

# Bayesian Learning

## Frequentist learning

- Suppose we have a dataset  $D$ , and we have a family of probabilistic models  $\{p(\cdot \mid \theta) : \theta \in \Theta\}$ , where  $\theta$  is the parameter vector of  $p(D \mid \theta)$ , and  $\Theta$  is the parameter space.
- In the frequentist approach, we often learn a single model  $p(\cdot \mid \theta)$  by maximizing the likelihood

$$\max_{\theta} p(D \mid \theta),$$

where  $p(D \mid \theta)$  is the probability that  $D$  is generated by the model  $p(\cdot \mid \theta)$ , and often called the likelihood.

- The likelihood is a measure of the compatibility between the model  $\theta$  and the data  $D$ .

## Bayes' Theorem (aka Bayes' law or Bayes' rule)

- For two events  $A$  and  $B$ , if  $P(B) \neq 0$ , then

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}.$$

- Interpretation

- $B$ : the observation/evidence
- $P(A)$ : the prior, or the initial belief for  $A$
- $P(B | A)$ : the likelihood
- $P(A | B)$ : the posterior, or the belief for  $A$  after observing  $B$

## Bayesian learning

- In the Bayesian approach, instead of learning a single model, we learn a distribution on all the models in  $\Theta$ .
- Specifically, we assume a prior distribution  $p(\theta)$  on  $\Theta$ , and given a dataset  $D$ , we compute a posterior

$$\overbrace{p(\theta \mid D)}^{\text{posterior}} = p(\theta)p(D \mid \theta)/Z \propto \overbrace{p(\theta)p(D \mid \theta)}^{\text{prior likelihood}},$$

where the normalization constant  $Z$  is

$$Z = \begin{cases} \sum_{\theta \in \Theta} p(\theta)p(D \mid \theta), & \text{if } p(\theta) \text{ is discrete}, \\ \int_{\Theta} p(\theta)p(D \mid \theta)d\theta, & \text{if } p(\theta) \text{ is continuous}. \end{cases}.$$

- The posterior distribution  $p(\theta \mid D)$  can be used in various ways when performing inference.

## Inference problems

- Compute the MAP (maximum a posterior) model:

$$\theta_{\text{MAP}} = \underset{\theta \in \Theta}{\operatorname{argmax}} p(\theta \mid D).$$

- Compute the (posterior) predictive distribution:

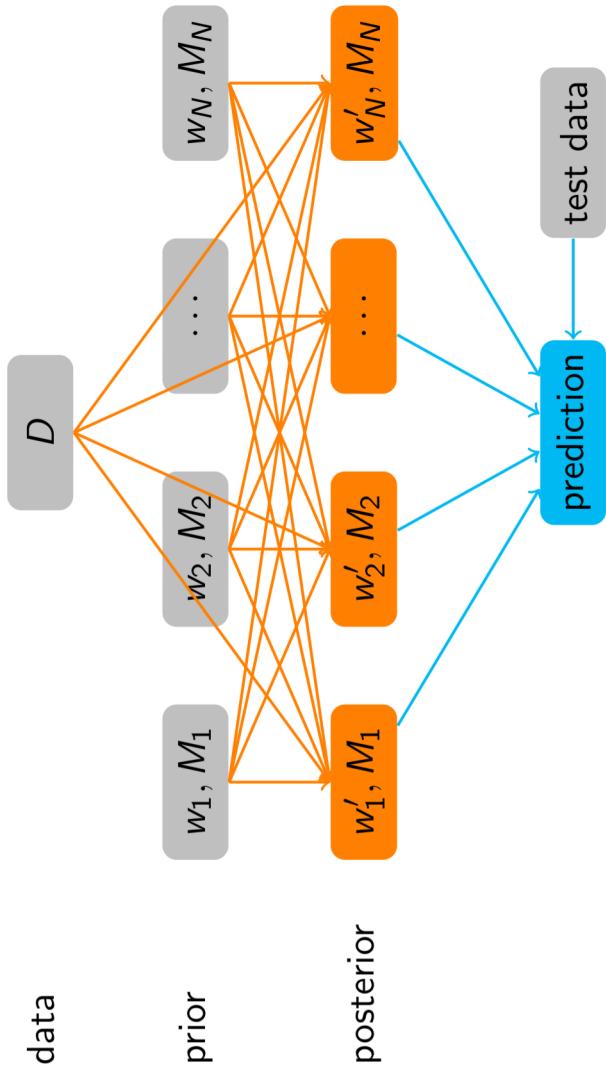
$$p(y \mid D, x) = \int p(y \mid \theta, x) p(\theta \mid D) d\theta.$$

- Compute posterior mean and variance of  $Y$  given  $x$ :

$$\text{posterior mean } \mu_x = \mathbb{E}(Y \mid x, D) = \int y p(y \mid D, x) dy$$

$$\text{posterior variance } \sigma_x^2 = \text{Var}(Y \mid x, D) = \int (y - \mu_x)^2 p(y \mid D, x) dy$$

## Bayesian method as an ensemble method



- Learning (computing posterior): construct a weighted ensemble of (often infinitely many) models using the Bayes' rule .
- Prediction: aggregate the ensemble's predictions (e.g., by computing the weighted average prediction).

### Example. Learning the probability of Heads

- Peter has two coins: the probability of Heads for one is 0.5, and 0.8 for the other. He chooses a coin, tosses it twice and observes one Head and one Tail. What's the probability of Heads of the chosen coin?
- The parameter space is  $\Theta = \{0.5, 0.8\}$ , the dataset  $D$  is a sequence of two Heads, and the likelihood is

$$p(D | \theta) = \theta(1 - \theta).$$

## The frequentist solution

- We have

$$p(D \mid \theta = 0.5) = 0.25,$$

$$p(D \mid \theta = 0.8) = 0.16.$$

- Thus  $\theta = 0.5$  is more compatible with the observations, and we may believe that the probability of Heads for the chosen coin is 0.5.

## The Bayesian solution

- We heard from a close friend of Peter that he likes the biased coin and chooses it with probability 0.9, that is, our prior is  $p(\theta = 0.5) = 0.1$  and  $p(\theta = 0.8) = 0.9$ .
- We have  $p(\theta = 0.5)p(D \mid \theta = 0.5) = 0.025$ , and  $p(\theta = 0.8)p(D \mid \theta = 0.8) = 0.144$ , thus the posterior distribution is

$$p(\theta \mid D) = \begin{cases} 25/169 & \theta = 0.5 \\ 144/169 & \theta = 0.8. \end{cases}$$

The MAP model is  $\theta = 0.8$ , thus we may believe that the probability of Heads for the chosen coin is 0.8.

- The posterior mean of  $\theta$  is  $0.5 \times 25/169 + 0.8 \times 144/169 = 0.76$ .  
The standard deviation of  $\theta$  given  $D$  is 0.18 (exercise).
- The probability distribution of the outcomes of next two tosses is

outcome	HH	HT	TH	TT
$p$	0.5823	0.1733	0.1733	0.0711

## Bayesian regression and classification

- In Bayesian regression and classification methods,
  - the probabilistic model  $p(D \mid \theta)$  is often much more complex than a simple Bernoulli distribution, and
  - the prior  $p(\theta)$  is much more complex than a discrete distribution.
- Two challenges
  - Specifying a good prior can be hard.
  - The inference problems are often computationally hard.
- We focus on the Gaussian processes, which
  - support a wide range of priors on all possible functions,
  - allow elegant algorithms for the inference problems.

# From SVM to Gaussian Process

## Support vector regression

- Recall: in binary support vector classifier, the discriminant function is of the form

$$f(\mathbf{x}) = \sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}).$$

$\mathbf{x}$  is predicted to be positive if  $f(\mathbf{x}) > 0$  and negative otherwise.

- SVMs can be used for regression too, and the regressor is of the form

$$f(\mathbf{x}) = \sum_i \alpha_i k(\mathbf{x}_i, \mathbf{x}).$$

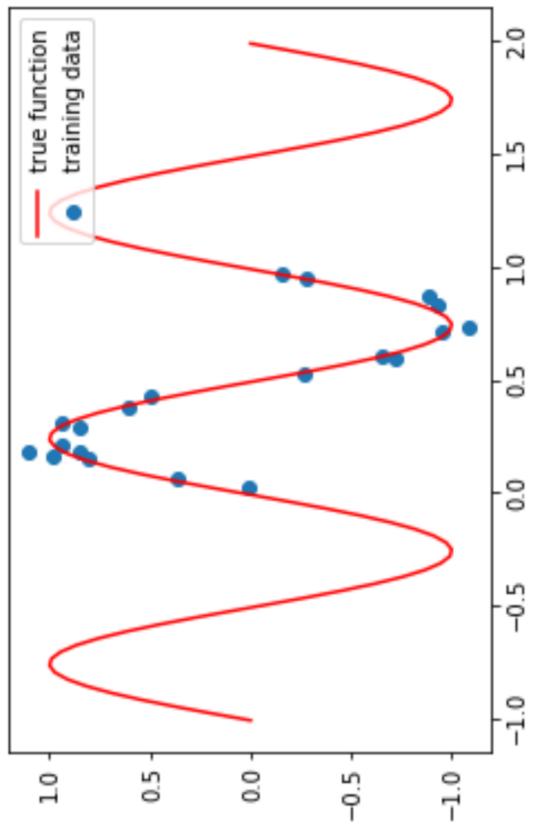
## Gaussian processes (GPs)

- Gaussian processes also produce regression estimates of the same form as SVMs:

$$f(\mathbf{x}) = \sum_i \alpha_i k(\mathbf{x}_i, \mathbf{x}).$$

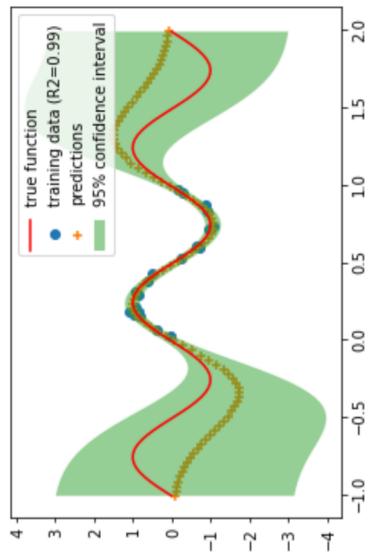
- However, there are a few important differences
  - SVM predicts a single estimated value, but GP predicts a distribution on the possible values.
  - in SVM, the kernel hyperparameters are often tuned by using methods like cross validation to choose the best values from a small set of candidate values; in GP, the hyperparameters can be optimized over all possible values using numerical optimization methods.

## Example. Learning the sine function

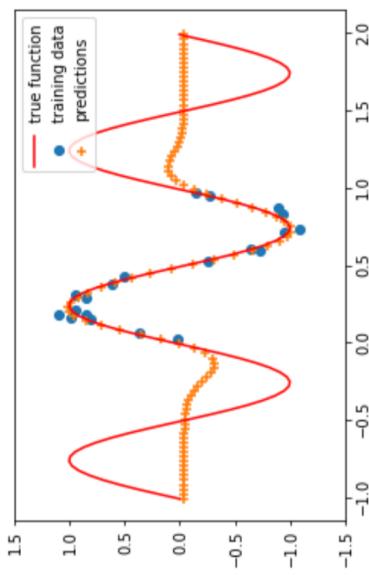


- $Y = \sin(2\pi x) + \epsilon$ , where  $\epsilon \sim N(0, 0.1^2)$ .
- Training set:  $x$  sampled from  $[0, 1]$
- Prediction:  $x$  sampled from  $[-1, 2]$   
     $\Rightarrow$  we can observe how well an algorithm interpolates and extrapolates.

Gaussian process (RBF kernel)



SVM (RBF kernel with  $\gamma = 2000$ )



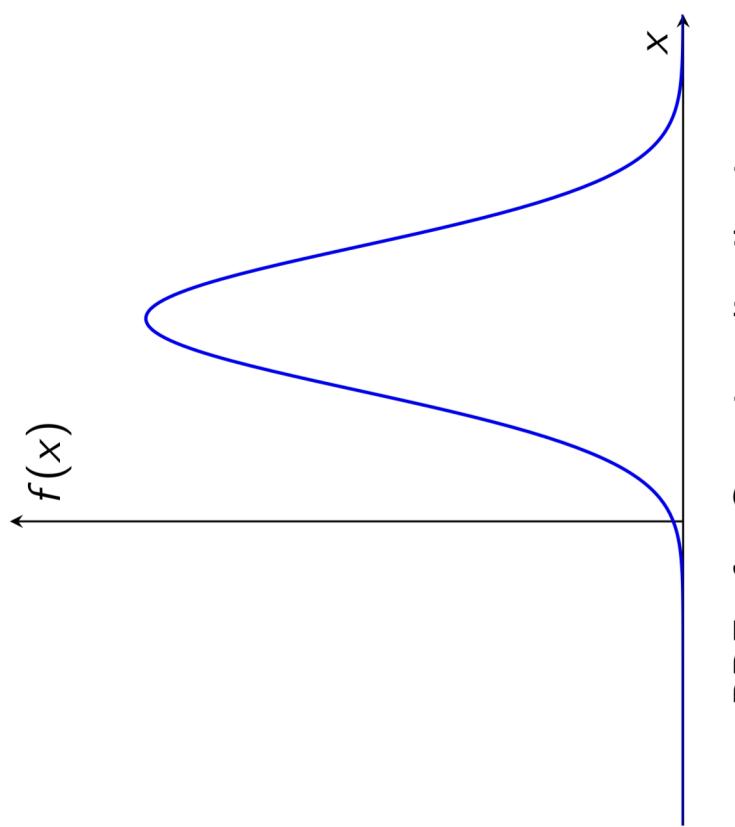
# Gaussian Distributions 101 102

## Univariate Gaussian distribution

- A random variable  $Y$  is said to follow a univariate Gaussian distribution  $N(\mu, \sigma^2)$  if its probability density function (PDF) is

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

- We often write this as  $Y \sim N(\mu, \sigma^2)$ , and use  $N(y; \mu, \sigma^2)$  to denote the PDF.



PDF of a Gaussian distribution

## Multivariate Gaussian distribution

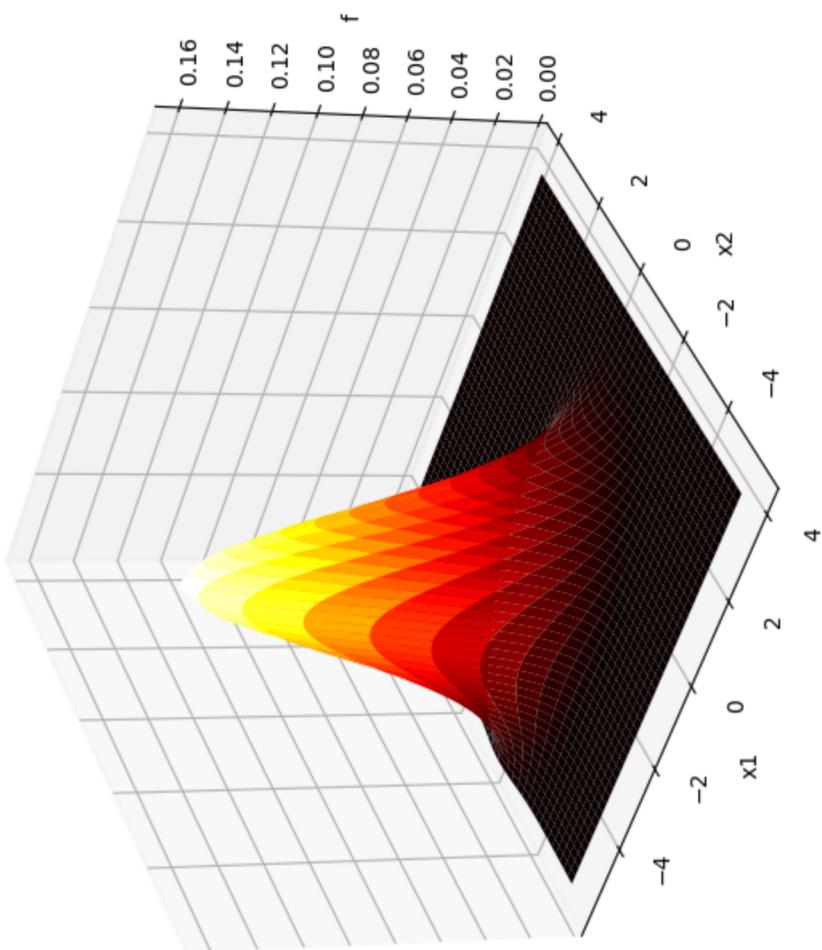
- A random vector  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$  is said to follow a multivariate Gaussian distribution  $N(\mu, \Sigma)$  with mean  $\mu$  and covariance matrix  $\Sigma$  if its PDF is

$$f(\mathbf{y}) = \frac{1}{\sqrt{|2\pi\Sigma|}} \exp\left(-\frac{1}{2}(\mathbf{y} - \mu)^\top \Sigma^{-1}(\mathbf{y} - \mu)\right), \quad (2)$$

where we use the notation  $|A|$  to denote the determinant of a matrix  $A$ .

- We often write this as  $\mathbf{Y} \sim N(\mu, \Sigma)$ , and use  $N(\mathbf{y}; \mu, \Sigma)$  to denote the PDF.

PDF of a bivariate Gaussian



- Notations: Let  $I = \{i_1, i_2, \dots, i_k\}$  and  $J = \{j_1, \dots, j_l\}$  be ordered sets/sequences. Then  $x_I$  denotes  $(x_{i_1}, \dots, x_{i_k})^\top$ , and  $\Sigma_{IJ}$  denotes
$$\begin{pmatrix} \sigma_{i_1 j_1} & \sigma_{i_1 j_2} & \cdots & \sigma_{i_1 j_l} \\ \vdots & \ddots & & \vdots \\ \sigma_{i_k j_1} & \sigma_{i_k j_2} & \cdots & \sigma_{i_k j_l} \end{pmatrix}$$
, where  $\sigma_{ij}$  is the  $(i, j)$ th element of  $\Sigma$ .

## Marginal distribution

- The marginal distribution of a Gaussian distribution is also a Gaussian distribution.
- Specifically, we partition  $\{1, \dots, d\}$  into two disjoint subsets  $I_1$  and  $I_2$  with  $n_1$  and  $n_2$  elements respectively, and let

$$\mathbf{Y}_i = Y_{I_i}, \quad \mu_i = \mu_{I_i}, \quad \Sigma_{ij} = \Sigma_{I_i I_j}.$$

- Then the marginal distribution of  $\mathbf{Y}_1$  is

$$f_1(\mathbf{y}_1) = N(\mathbf{y}_1; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}). \tag{3}$$

## Conditional distribution

- The conditional distribution of a Gaussian distribution is also a Gaussian distribution.
- Specifically, the distribution of  $\mathbf{Y}_2$  given  $\mathbf{Y}_1 = \mathbf{y}_1$  is

$$f_{2|1}(\mathbf{y}_2|\mathbf{y}_1) = N(\mathbf{y}_2; \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}(\mathbf{y}_1 - \boldsymbol{\mu}_1), \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}). \quad (4)$$

- We often drop the subscripts in  $f_1$  and  $f_{2|1}$  when there is no confusion.

### Example. Bivariate Gaussian

- Let  $Y_1$  and  $Y_2$  be the returns for two investments. They are known to have a joint distribution

$$\begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} \sim N \left( \begin{pmatrix} -1 \\ -2 \end{pmatrix}, \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix} \right).$$

- Then the marginal distributions are

$$Y_1 \sim N(-1, 1), \quad Y_2 \sim N(-2, 5).$$

- The conditional distribution of  $Y_1$  given  $Y_2 = 3$  has mean  $-1 + 2 \cdot \frac{1}{5} \cdot (3 - (-2)) = 1$  and variance  $1 - 2 \cdot \frac{1}{5} \cdot 2 = \frac{1}{5}$ , that is,

$$Y_1 \mid Y_2 = 3 \sim N(1, 1/5).$$

The conditional distribution of  $Y_2$  given  $Y_1 = 2$  has mean  $-2 + 2 \cdot \frac{1}{1} \cdot (2 - (-1)) = 4$  and variance  $5 - 2 \cdot \frac{1}{1} \cdot 2 = 1$ , that is,

$$Y_2 \mid Y_1 = 2 \sim N(4, 1).$$

### Example. Trivariate Gaussian

- Let  $Y_1, Y_2, Y_3$  be the returns for three investments. They are known to have a joint distribution

$$\begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \end{pmatrix} \sim N \left( \begin{pmatrix} -1 \\ -2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 1 \\ 2 & 5 & 2 \\ 1 & 2 & 4 \end{pmatrix} \right).$$

- The conditional distribution of  $Y_1, Y_2$  given  $Y_3 = 2$  has mean  $\begin{pmatrix} -1 \\ -2 \end{pmatrix} + \begin{pmatrix} 1 \\ 2 \end{pmatrix}(4)^{-1}(2-1) = \begin{pmatrix} -3/4 \\ -3/2 \end{pmatrix}$ , and covariance matrix  $\begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \end{pmatrix}(4)^{-1}(1-2) = \begin{pmatrix} 3/4 & 3/2 \\ 3/2 & 4 \end{pmatrix}$ , thus

$$\begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} \Big| Y_3 = 2 \sim N \left( \begin{pmatrix} -3/4 \\ -3/2 \end{pmatrix}, \begin{pmatrix} 3/4 & 3/2 \\ 3/2 & 4 \end{pmatrix} \right)$$

# Gaussian Processes (GPs)

## A generalization of multivariate Gaussians

- Specifically, a Gaussian process (GP) is a collection of random variables such that any finite subset of which follows a (multivariate) Gaussian distribution.
- Recall: if  $(Y_1, \dots, Y_n)$  follows a multivariate Gaussian distribution, then any subset of them follows a multivariate Gaussian distribution.  
⇒ a multivariate Gaussian distribution is a GP.

## Mean and kernel

- A GP can be specified in terms of the mean function  $m$  and the covariance function (aka kernel)  $k$ , defined by

$$\begin{aligned}m(Y) &= \mathbb{E}(Y), \\k(Y, Y') &= \text{cov}(Y, Y'),\end{aligned}$$

- where  $Y$  and  $Y'$  are any two random variables in the GP
- For example, if the GP under consideration is a multivariate Gaussian  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top \sim N(\boldsymbol{\mu}, \Sigma)$ , then

$$\begin{aligned}m(Y_i) &= \mu_i, \\k(Y_i, Y_j) &= \sigma_{ij}.\end{aligned}$$

## GPs as Distributions on Functions

- In many cases, each random variable in a GP can be considered as the output on an input.
- In particular, we often consider a GP  $\{Y(\mathbf{x}) : \mathbf{x} \in \mathbf{R}^d\}$ , where  $\mathbf{x}$  denotes an input feature vector, and  $Y(\mathbf{x})$  denotes the output for  $\mathbf{x}$ .
- If we define a random function  $F$  such that  $F(\mathbf{x})$  is  $Y(\mathbf{x})$ , then the GP is the probability distribution for  $F$ , and we write

$$F \sim GP(m, k),$$

where  $m$  and  $k$  are the mean function and the covariance function of the GP.

- For example, consider  $Y \sim N(\mu, \sigma^2)$ . This can be viewed as a distribution of real-valued functions defined on a set  $\{\mathbf{x}_1\}$  with a single feature vector, where the PDF of a function  $f$  defined on  $\{\mathbf{x}_1\}$  is

$$p(f) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(f(\mathbf{x}_1) - \mu)^2}{2\sigma^2}}.$$

- Similarly, if  $\mathbf{Y} = (Y_1, \dots, Y_n)^\top \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ , then it can be viewed as a distribution of real-valued functions defined on  $n$  feature vectors  $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ .

- The covariance function  $k(Y(\mathbf{x}), Y(\mathbf{x}'))$  is then a function of  $\mathbf{x}$  and  $\mathbf{x}'$  and often simply written as  $k(\mathbf{x}, \mathbf{x}')$ .
- Intuitively, the kernel controls how the outputs for  $\mathbf{x}$  and  $\mathbf{x}'$  are related with each other.
- As in SVMs, the choice of the kernel is important in GPs.

# GP Regression

## Noise-free observation model

- Consider a training set  $D = (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n) \in \mathbf{R}^d \times \mathbf{R}$ .
- In the noise-free GP model, we assume that  $D$  is generated as follows
  - sample  $f$  from  $GP(m, k)$ ,
  - for each input  $\mathbf{x}_1, \dots, \mathbf{x}_n$ , observe

$$y_i = f(\mathbf{x}_i).$$

- We want to make predictions on  $\mathbf{x}'_1, \dots, \mathbf{x}'_t$ .
- Note: we assume  $\mathbf{x}'_i$ 's and  $\mathbf{x}'_j$ 's are all different.

- Notations

notation	meaning
$\mathbf{X}$	matrix with $\mathbf{x}_i^\top$ as the $i$ th row
$\mathbf{X}'$	matrix with $\mathbf{x}'_i$ as the $i$ th row
$\mu\mathbf{x}$	$(m(\mathbf{x}_1), \dots, m(\mathbf{x}_n))^\top$ ; $\mu\mathbf{x}'$ similarly defined
$\mathbf{Y}$	$(Y(\mathbf{x}_1), \dots, Y(\mathbf{x}_n))^\top$
$\mathbf{y}$	$(y_1, \dots, y_n)^\top$
$\mathbf{Y}'$	$(Y(\mathbf{x}'_1), \dots, Y(\mathbf{x}'_t))^\top$
$K_{\mathbf{X}, \mathbf{X}'}$	matrix with $k(\mathbf{x}_i, \mathbf{x}'_j)$ as the $(i, j)$ th entry here $\mathbf{X}$ and $\mathbf{X}'$ can be any two matrices

## Prediction

- The joint distribution of  $\mathbf{Y}$  and  $\mathbf{Y}'$  is

$$\begin{pmatrix} \mathbf{Y} \\ \mathbf{Y}' \end{pmatrix} \sim N \left( \begin{pmatrix} \mu_{\mathbf{X}} \\ \mu_{\mathbf{X}'} \end{pmatrix}, \begin{pmatrix} K_{\mathbf{X}, \mathbf{X}} & K_{\mathbf{X}, \mathbf{X}'} \\ K_{\mathbf{X}', \mathbf{X}} & K_{\mathbf{X}', \mathbf{X}'} \end{pmatrix} \right).$$

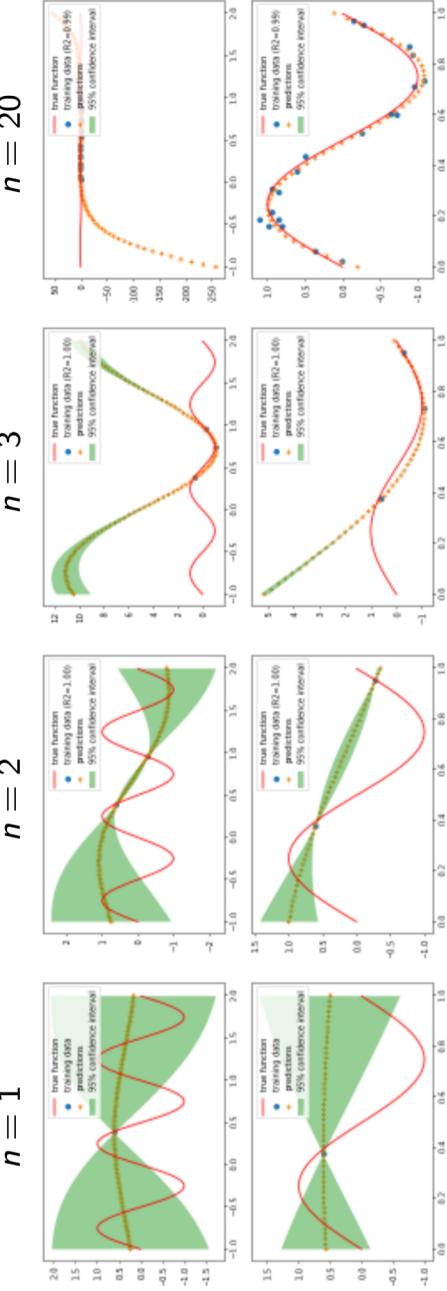
- The predictive distribution of  $\mathbf{Y}'$  is

$$\mathbf{Y}' | \mathbf{X}', \mathbf{X}, \mathbf{y} \sim N \left( \underbrace{\mu_{\mathbf{X}'} + K_{\mathbf{X}', \mathbf{X}} K_{\mathbf{X}, \mathbf{X}}^{-1} (\mathbf{y} - \mu_{\mathbf{X}})}_{\text{posterior mean}}, \underbrace{K_{\mathbf{X}', \mathbf{X}'} - K_{\mathbf{X}', \mathbf{X}} K_{\mathbf{X}, \mathbf{X}}^{-1} K_{\mathbf{X}, \mathbf{X}'}}_{\text{posterior covariance}} \right). \quad (6)$$

- Let  $(\alpha_1, \dots, \alpha_n)^\top = K_{\mathbf{X}, \mathbf{X}}^{-1} (\mathbf{y} - \mu_{\mathbf{X}})$ , then for any  $\mathbf{x}'$ , its posterior mean is

$$f(\mathbf{x}') = \mu_{\mathbf{x}'} + \sum_{i=1}^n \alpha_i k(\mathbf{x}', \mathbf{x}_i).$$

Evolution of noise-free GP with  $m = 0$  and  $k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2}\right)$



- Good training set performance with all training set  $R^2$  values nearly 1.
- Interpolation performance improves as  $n$  increases, but extrapolation performance becomes worse; in addition uncertainty estimates are often nearly 0 and not useful.

## Noisy observation model

- In general, the observed value  $y$  is only a noisy observation of the true value.
- The noisy observation model is the same as the noise-free model, except that  $y_i$  is not  $f(\mathbf{x}_i)$ , but

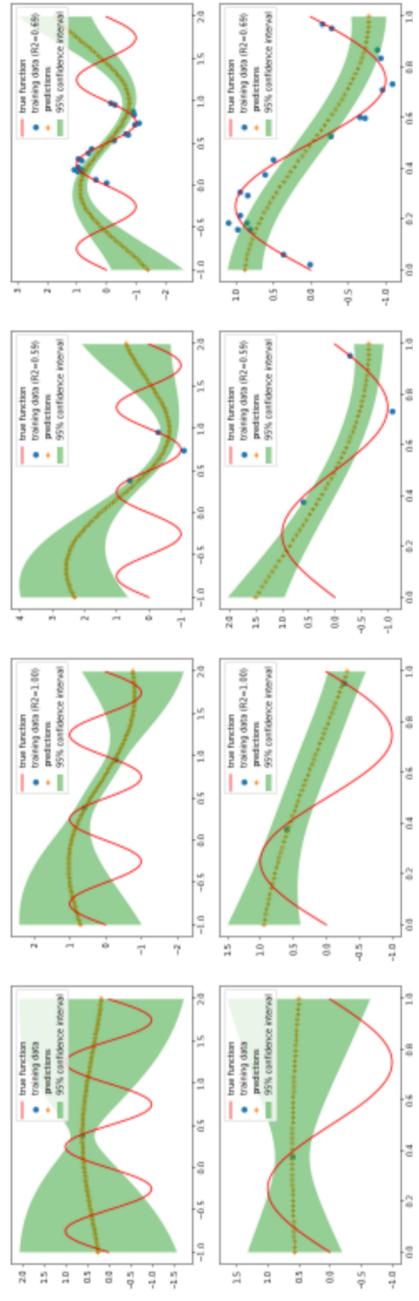
$$y = f(\mathbf{x}) + \epsilon,$$

where  $\epsilon \sim N(0, \sigma^2)$ , and  $\sigma^2$  is an unknown constant.

- The predictive distribution is

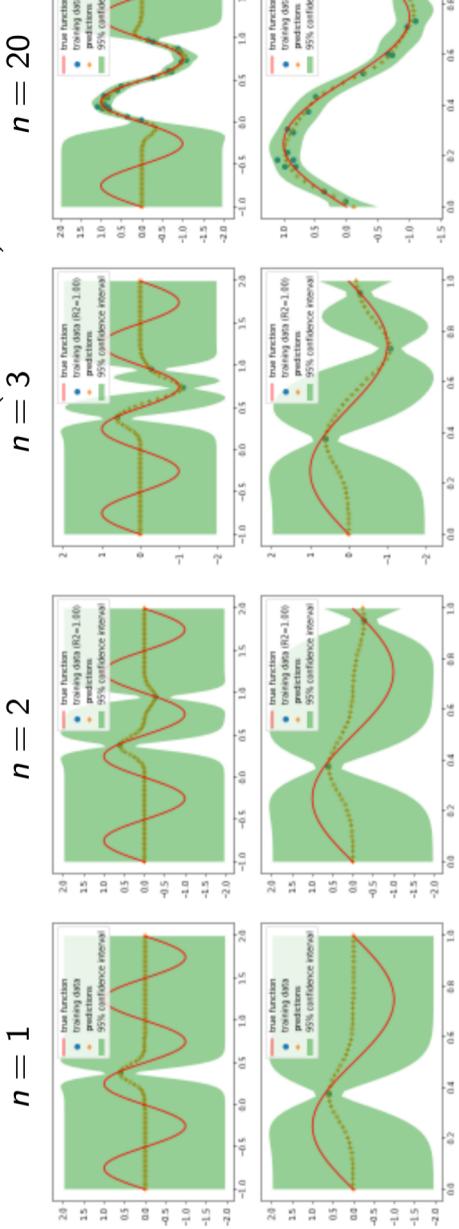
$$\mathbf{Y}' \mid \mathbf{X}', \mathbf{X}, \mathbf{y} \sim N\left(\underbrace{\mu_{\mathbf{x}'} + K_{\mathbf{x}', \mathbf{x}}(K_{\mathbf{x}, \mathbf{x}} + \sigma^2 I)^{-1}(\mathbf{y} - \mu_{\mathbf{x}})}_{\text{posterior mean}}, \underbrace{K_{\mathbf{x}', \mathbf{x}'} - K_{\mathbf{x}', \mathbf{x}}(K_{\mathbf{x}, \mathbf{x}} + \sigma^2 I)^{-1}K_{\mathbf{x}, \mathbf{x}'}}_{\text{posterior covariance}}\right). \quad (7)$$

Evolution of noisy GP with  $m = 0$ ,  $k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2}\right)$  and  $\sigma^2 = 0.01$ .



- Incorporating noise in the observation leads to weaker training set performance, but better confidence intervals and better extrapolation performance.

Evolution of noisy GP with  $m = 0$ ,  $k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{0.02}\right)$  and  $\sigma^2 = 0.01$ .



- Using an RBF kernel with a smaller length scale leads to good interpolation and extrapolation performance; and the uncertainty estimates are good, though slightly too large.

**Question:** Can we learn  $\sigma^2$  and other hyperparameters from data?

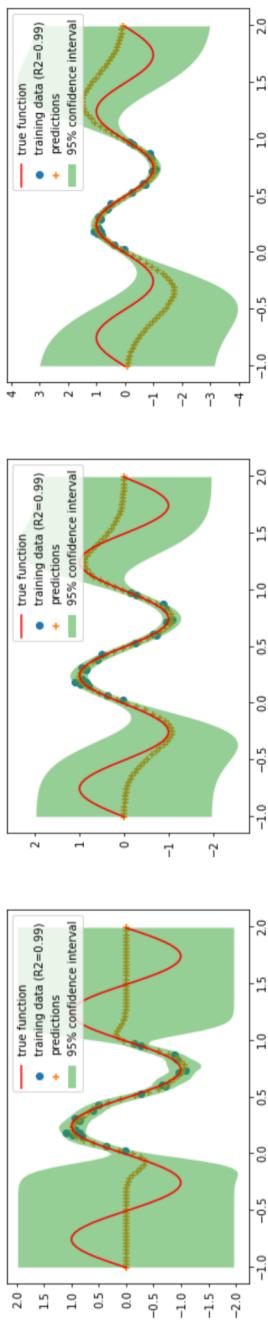
# Model Selection

- The problem of choosing the hyperparameters of a GP model is a problem of model selection, thus we can use techniques such as cross-validation.
- Let  $\varphi$  be the learnable parameters of the mean function  $m$  and the kernel function  $k$ , and the observation noise variance  $\sigma^2$ . We can choose  $\varphi$  by maximizing the likelihood function

$$L(\varphi) = p(\mathbf{y} \mid \mathbf{X}, m, k) = N(\mathbf{y}; \mu_{\mathbf{X}}, K_{\mathbf{X}, \mathbf{X}}). \quad (9)$$

- The likelihood function measures the compatibility between  $\varphi$  and the data.
- Various numerical optimization algorithms can be used to maximize the likelihood function (details beyond this course).

Scaled RBF kernel  $c \exp(-(\mathbf{x} - \mathbf{x}')^2/(2\ell^2))$ ,  $m = 0$



$$c = 1, \ell = 0.1, \sigma^2 = 0.01 \quad c = 1, \ell = 0.335, \sigma^2 = 0.00821 \quad c = 2.435, \ell = 0.335, \sigma^2 = 0.00821$$

Hyperparameters in red are learned.

## Commonly-used Kernels

- Not every function  $k(\mathbf{x}, \mathbf{x}')$  can be used as a kernel function in SVMs; this is true in GPs too.
- Using the right kernel is often important to make GPs work.
- In practice, we can try commonly used kernels, or try kernels constructed using them.

## Constant kernel

- The constant kernel is defined as

$$k(\mathbf{x}, \mathbf{x}') = c, \quad (10)$$

where  $c \geq 0$  is a constant.

- This is not really an interesting kernel on its own, but is useful when constructing new kernels using known kernels.

## Linear kernel

- The linear kernel is defined as

$$k_{\text{linear}}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}' + \sigma_0^2.$$

The linear kernel is said to be homogeneous if  $\sigma_0 = 0$  and inhomogeneous otherwise.

- The posterior mean function is a linear function, thus this kernel is suitable if the output is approximately linear in the features.

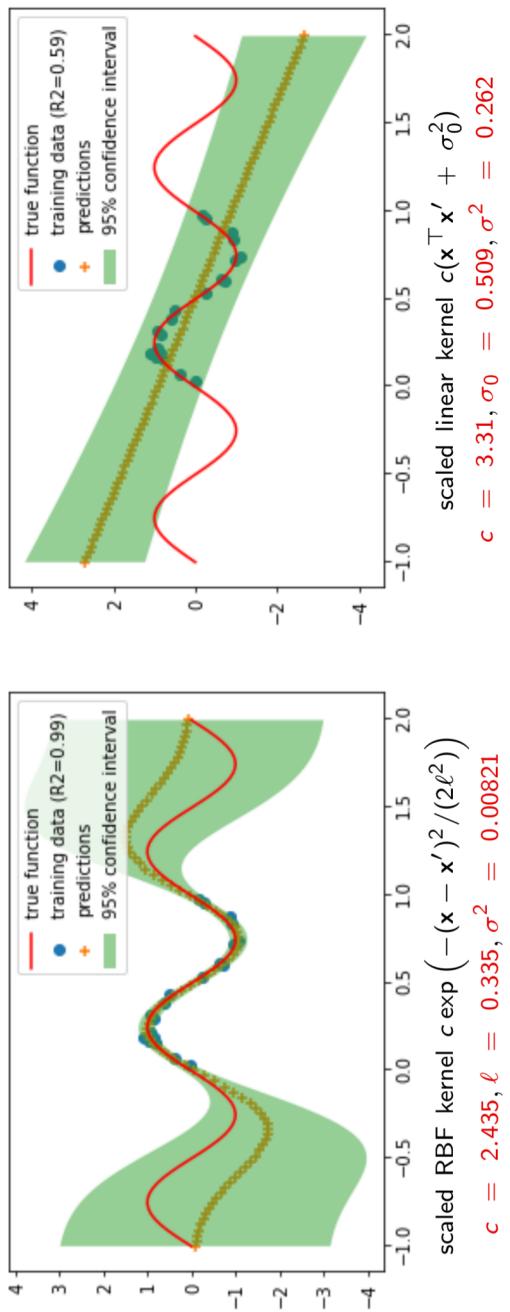
## Squared exponential kernel

- The squared exponential kernel (aka RBF kernel) is defined as

$$k_{\text{SE}}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell^2}\right),$$

where  $\ell$  is called the characteristic length scale.

- When the distance between  $\mathbf{x}$  and  $\mathbf{x}'$  decreases, the kernel value increases  $\Rightarrow$  more similar inputs lead to more correlated outputs.



Hyperparameters in red are learned;  $m = 0$ .

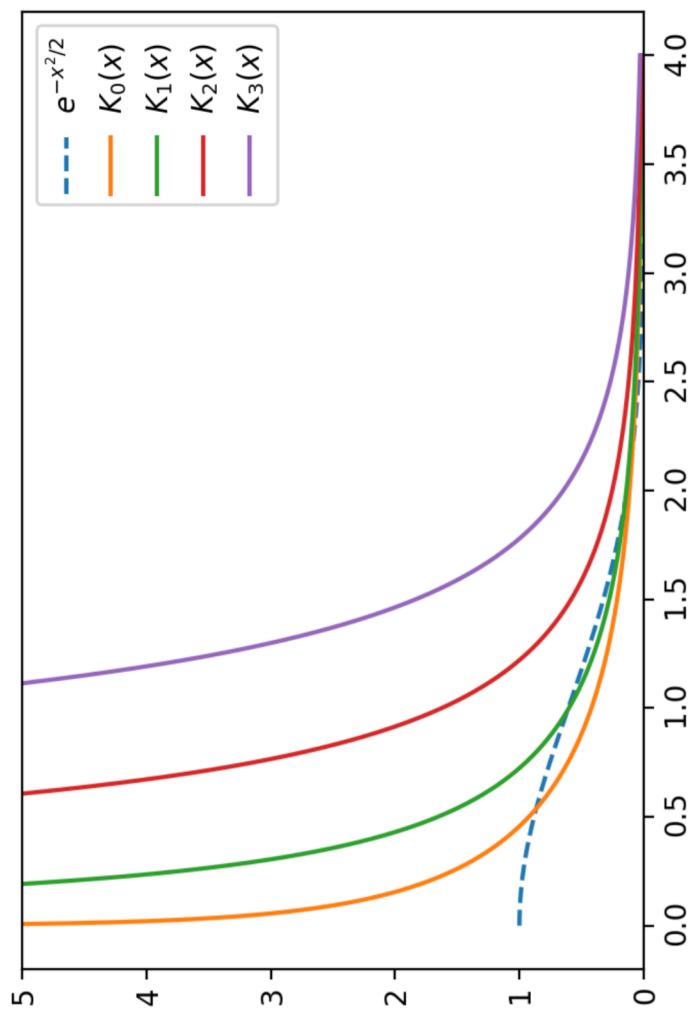
## Matérn kernel

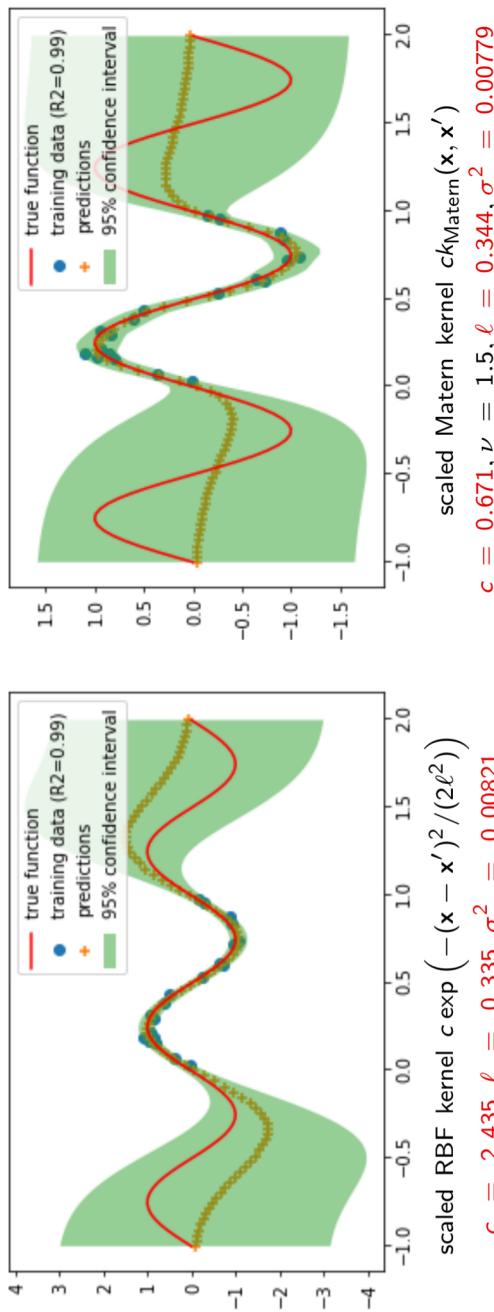
- The Matérn kernel is defined as

$$k_{\text{Matérn}}(\mathbf{x}, \mathbf{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu} \|\mathbf{x} - \mathbf{x}'\|}{\ell} \right)^\nu K_\nu \left( \frac{\sqrt{2\nu} \|\mathbf{x} - \mathbf{x}'\|}{\ell} \right),$$

with positive parameters  $\nu$  and  $\ell$ , where  $\Gamma$  is the Gamma function, and  $K_\nu$  is the modified Bessel function of the second kind.

- $K_\nu(x) \sim \sqrt{\pi/(2x)} \exp(-x)$  as  $x \rightarrow \infty$ .





Hyperparameters in red are learned;  $m = 0$ .

# Constructing New Kernels

- If  $k_1$  and  $k_2$  are kernels, then
  - $k(\mathbf{x}, \mathbf{x}') = ck_1(\mathbf{x}, \mathbf{x}')$  is a kernel for any  $c > 0$ .
  - $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$  is a kernel.
  - $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')$  is a kernel.
  - $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')^p$  is a kernel.

# GP Regression in sklearn

```
from sklearn.datasets import fetch_california_housing
from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.gaussian_process.kernels import WhiteKernel,
    ConstantKernel, Matern
from sklearn.model_selection import train_test_split

X, y = fetch_california_housing(return_X_y=True)
X_tr, X_ts, y_tr, y_ts = train_test_split(X, y, test_size=0.8,
    random_state=42)
# train and test a GP with scaled Matern kernel and noisy obs
kernel = ConstantKernel() * Matern() + WhiteKernel()
gpr = GaussianProcessRegressor(kernel=kernel, random_state=0)
gpr.fit(X_tr, y_tr)
print(gpr.score(X_ts, y_ts))
```

sklearn uses zero-mean GPs. By default, kernel hyperparameters are optimized during fitting.

# GP Classification

- GPs can be used for classification as well.
- The theory is much more involved than that for regression and is beyond this course.
- However, there are many GP libraries, and implementing a GP classifier is easy.
- As in regression, choosing the right kernel is a main consideration in getting the most out of a GP classifier.

# GP Classification in sklearn

```
from sklearn.datasets import load_digits
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import WhiteKernel,
    ConstantKernel, Matern
from sklearn.model_selection import train_test_split

X, y = load_digits(return_X_y=True)
X_tr, X_ts, y_tr, y_ts = train_test_split(X, y, test_size=0.8,
                                            random_state=42)
# train and test a GP with scaled Matern kernel and noisy obs
kernel = ConstantKernel() * Matern() + WhiteKernel()
gpr = GaussianProcessClassifier(kernel=kernel, random_state=0)
gpr.fit(X_tr, y_tr)
print(gpr.score(X_ts, y_ts))
```