Pattern Information Processing: Linear Models and Least-Squares

Masashi Sugiyama
(Department of Computer Science)

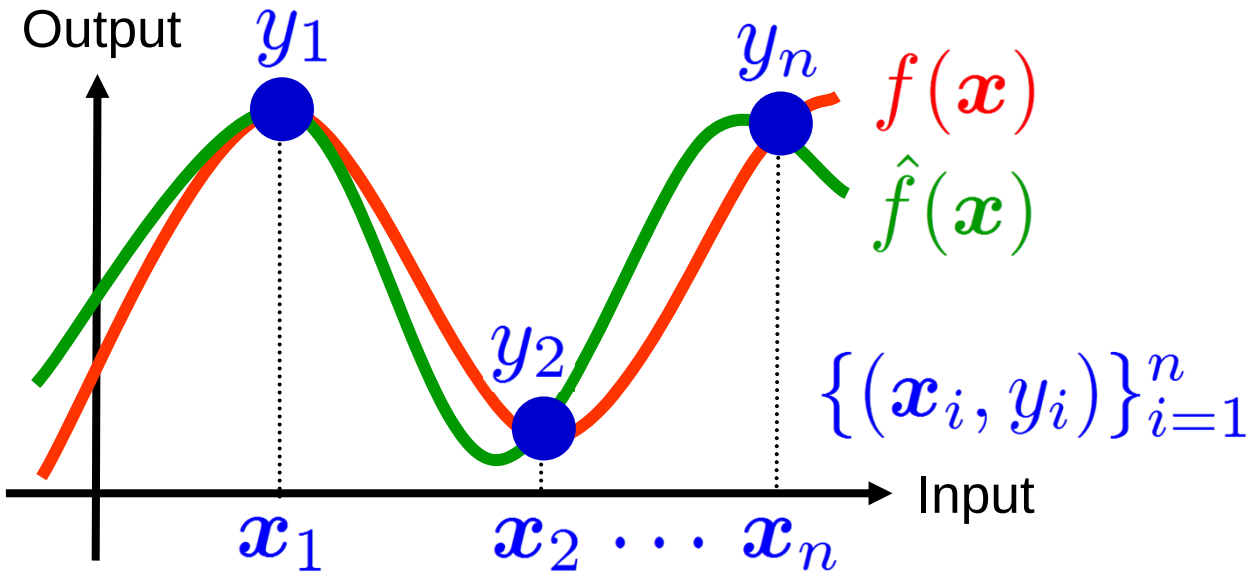
Contact: W8E-505
sugi@cs.titech.ac.jp
http://sugiyama-www.cs.titech.ac.jp/~sugi/

Supervised Learning: Learning from Examples

- There exists a function $y = f(\boldsymbol{x})$.
- We do not know $f(\boldsymbol{x})$, but we are given its samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$.
- The goal of supervised learning is to obtain an approximation $\hat{f}(\boldsymbol{x})$ to $f(\boldsymbol{x})$ from $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$.

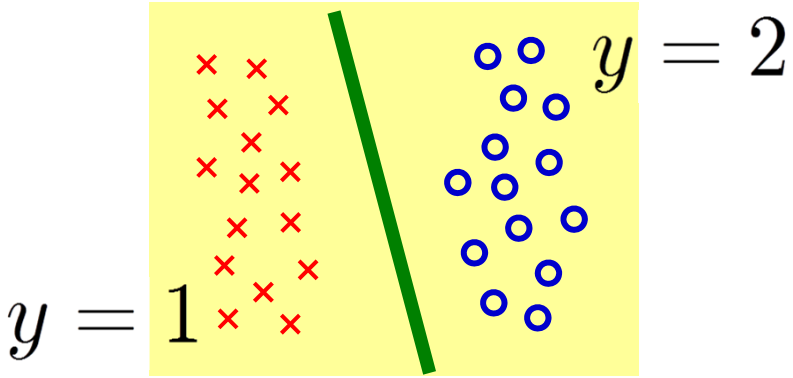
Regression: Real Outputs

- For real output $y \in \mathbb{R}$, the supervised learning problem is called regression.



Classification: Categorical Outputs

- For categorical output $y \in \{1, \ldots, m\}$, the problem is called classification.



- For the moment, let us focus on regression problems.

Notations

- $\mathcal{D} \subset \mathbb{R}^d$: Input domain
- $f(\boldsymbol{x})$:Learning target function ($\mathcal{D} \to \mathbb{R}$)
- $\boldsymbol{x}_i \in \mathcal{D}$:Training input point
- $y_i \in \mathbb{R}$:Training output value
- $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$:Training examples
- $\hat{f} \in \mathcal{M}$:Learned function
- $\mathcal{M}$:Model (a set of functions)

Today's Plan

- Model:
  - Linear model
  - Kernel model
- Learning method:
  - Least-squares learning

Linear/Non-Linear Models

- Model is a set of functions from which learning result functions are searched.
- We use a parameterized family of functions

$$\{f_{\boldsymbol{\alpha}}(\boldsymbol{x}) \mid \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_b)^\top\}$$

- Linear model: $f_{\boldsymbol{\alpha}}(\boldsymbol{x})$ is linear with respect to $\boldsymbol{\alpha}$

  (Note: not necessarily linear with respect to $\boldsymbol{x}$)
- Non-linear model: Otherwise

Linear Model

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{i=1}^{b} \alpha_i \varphi_i(\boldsymbol{x})$$

- $\{\varphi_i(\boldsymbol{x})\}_{i=1}^{b}$:Linearly independent basis functions
- For example, when $d = 1$:
  - Polynomial bases

$$1, x, x^2, \ldots, x^{b-1}$$

  - Sinusoidal bases

$$1, \sin x, \cos x, \ldots, \sin kx, \cos kx$$

$$b = 2k + 1$$

Multi-Dimensional Linear Model

- For multidimensional input $(d > 1)$, a product model could be used.

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{i_1=1}^{c} \cdots \sum_{i_d=1}^{c} \alpha_{i_1,\ldots,i_d} \varphi_{i_1}(x^{(1)}) \cdots \varphi_{i_d}(x^{(d)})$$

$$\boldsymbol{x} = (x^{(1)}, \ldots, x^{(d)})^\top$$

- The number of parameters is $b = c^d$, growing exponentially with respect to d.
- Infeasible for large d !

Additive Model

- For large d, we have to reduce the number of parameters.
- Additive model:

$$\boldsymbol{x} = (x^{(1)}, \ldots, x^{(d)})^\top$$

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{j=1}^{d} \sum_{i=1}^{c} \alpha_{i,j} \varphi_i(x^{(j)})$$

- The number of parameters is only $b = cd$.
- However, additive model is too simple so its representation capability may not be rich enough in some application.

Kernel Model

- Linear model:

  $\{\varphi_i(\boldsymbol{x})\}_{i=1}^{b}$ do not depend on $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$

- Kernel model:

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i)$$

  - Example: Gaussian kernel

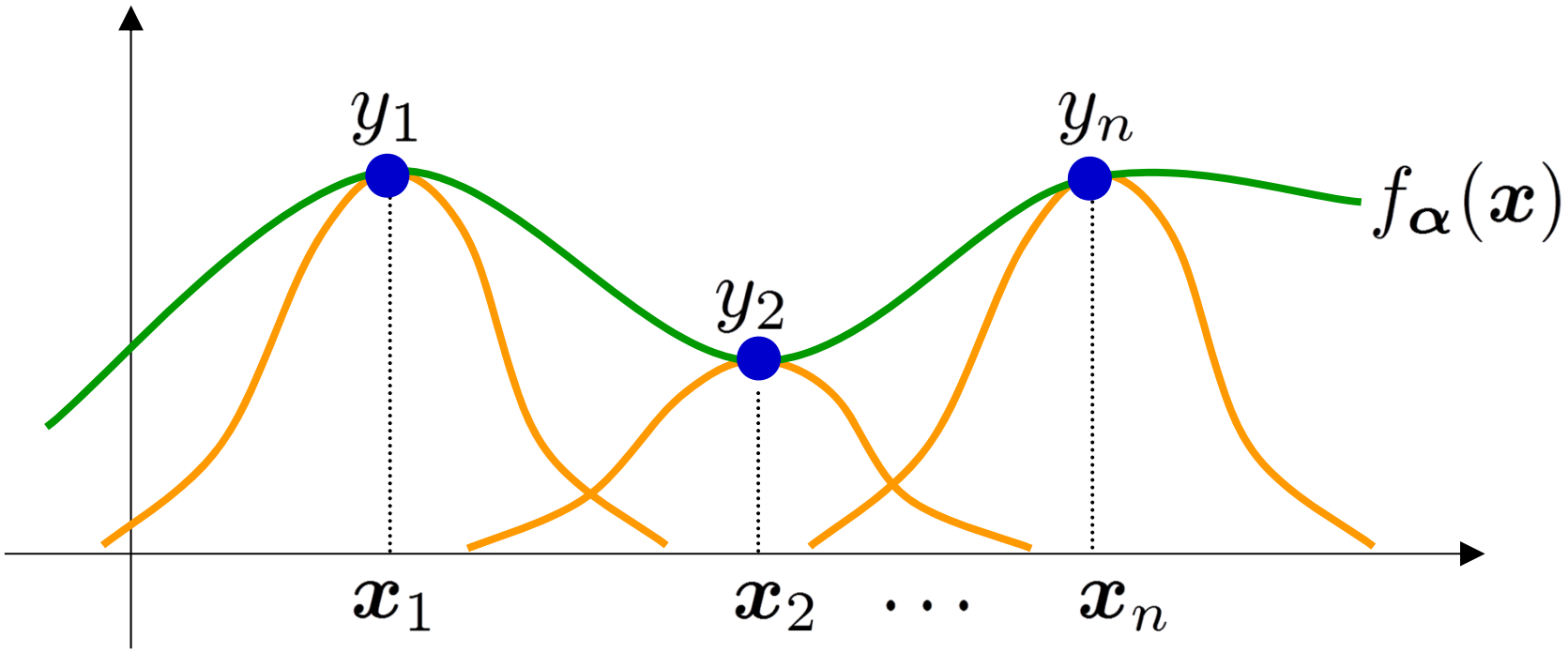
$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2h^2}\right)$$

$h(\geq 0)$: Bandwidth

Gaussian Kernel Model

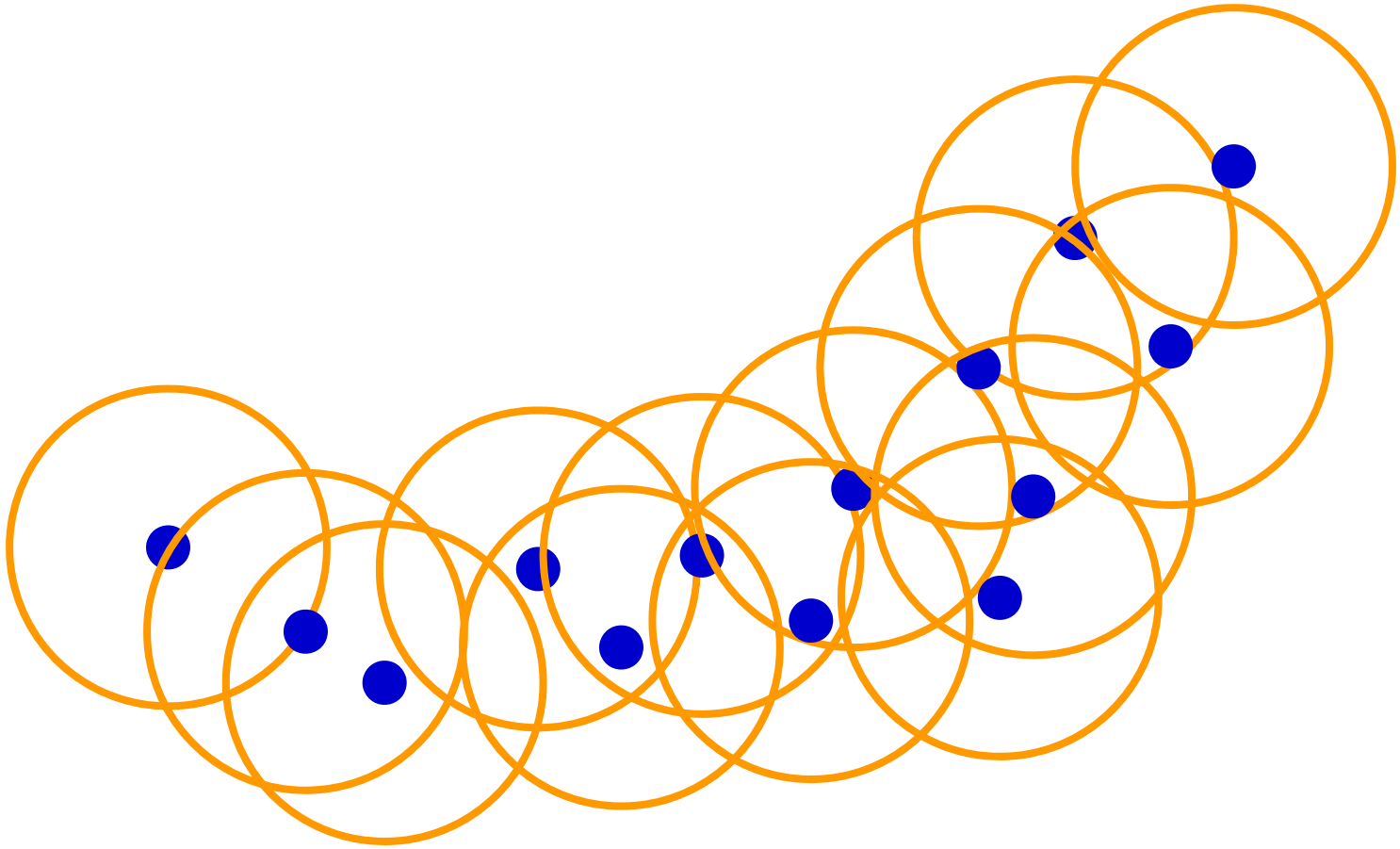
$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{2h^2}\right)$$

- Put kernel functions at training input points.



Gaussian Kernel Model (cont.)

- When training inputs are unevenly distributed, Gaussian kernel model automatically focuses on the region where training inputs exist.



Kernel Model (cont.)

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i)$$

- The number of parameters is n, which is independent of the input dimensionality d.
- Although kernel model is linear w.r.t. $\boldsymbol{\alpha}$, the number of parameters grows as the number of training samples increases.
- Mathematical treatment could be different from ordinary linear models (called a “non-parametric model” in statistics).

Summary of Linear Models

- Linear model (product):
  High flexibility, high complexity
- Linear model (additive):
  Low flexibility, low complexity
- Kernel model:
  Moderate flexibility, moderate complexity

- Good model depends on applications.
- Later, we discuss how to choose an appropriate model ("model selection").

Learning Methods

- Linear learning method:

  Parameter vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_b)^\top$ is estimated linearly with respect to

  $$\boldsymbol{y} = (y_1, \ldots, y_n)^\top$$

- Non-linear learning methods: Otherwise

Linear Learning for Linear Models / Kernel Models

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{i=1}^{b} \alpha_i \varphi_i(\boldsymbol{x})$$

- In linear learning methods, a learned parameter vector is given by

$$\hat{\boldsymbol{\alpha}} = \boldsymbol{L}\boldsymbol{y}$$

$\boldsymbol{L}$:Learning matrix

Least-Squares Learning

- Learn $\boldsymbol{\alpha}$ such that the squared error at training input points is minimized:

$$\hat{\boldsymbol{\alpha}}_{LS} = \underset{\boldsymbol{\alpha} \in \mathbb{R}^b}{\operatorname{argmin}} J_{LS}(\boldsymbol{\alpha})$$

$$J_{LS}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \left(f_{\boldsymbol{\alpha}}(\boldsymbol{x}_i) - y_i\right)^2$$

$$= \|\boldsymbol{X}\boldsymbol{\alpha} - \boldsymbol{y}\|^2$$

$\boldsymbol{X}_{i,j} = \varphi_j(\boldsymbol{x}_i)$:Design matrix $(n \times b)$

How to Obtain A Solution

- Extreme-value condition:

$$\nabla J_{LS}(\hat{\boldsymbol{\alpha}}_{LS}) = 2\boldsymbol{X}^\top(\boldsymbol{X}\hat{\boldsymbol{\alpha}}_{LS} - \boldsymbol{y}) = \boldsymbol{0}$$

$$\Rightarrow \quad \hat{\boldsymbol{\alpha}}_{LS} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{y}$$

(We assume $(\boldsymbol{X}^\top\boldsymbol{X})^{-1}$ exists.)

- Therefore, LS is linear learning.

$$\hat{\boldsymbol{\alpha}}_{LS} = \boldsymbol{L}_{LS}\boldsymbol{y}$$

$$\boldsymbol{L}_{LS} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top$$

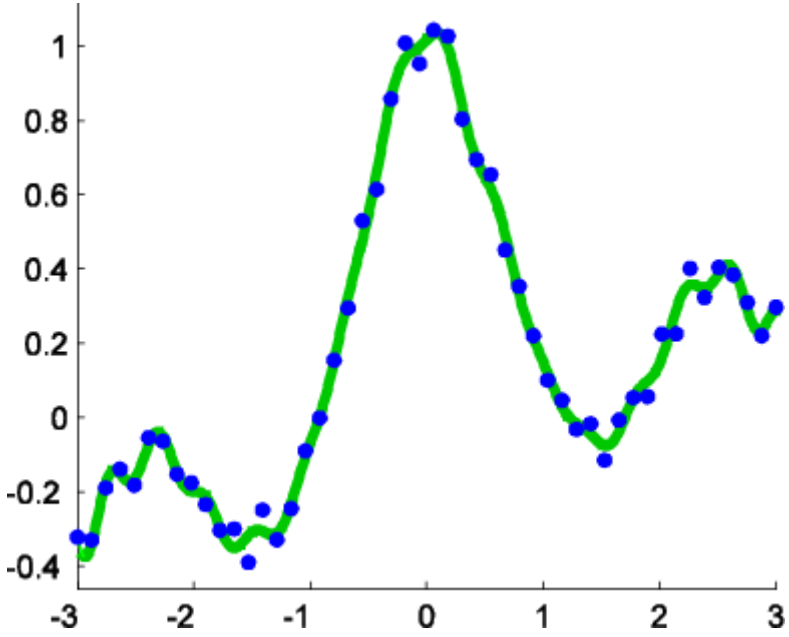
If you are not familiar with vector-derivatives, see e.g, "Matrix Cookbook" (http://matrixcookbook.com)

Example of LS

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{i=1}^{b} \alpha_i \varphi_i(\boldsymbol{x})$$

- Trigonometric polynomial model:

$$1, \sin x, \cos x, \ldots, \sin 15x, \cos 15x \quad (b = 31)$$



Homework

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{i=1}^{b} \alpha_i \varphi_i(\boldsymbol{x})$$

1. Prove that the LS solution in kernel models is given by

$$\hat{\boldsymbol{\alpha}}_{LS} = \boldsymbol{L}_{LS}\boldsymbol{y}$$

$$\boldsymbol{L}_{LS} = \boldsymbol{K}^{-1}$$

$$\boldsymbol{K}_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

(Kernel matrix)

Homework (cont.)

2. For your own toy 1-dimensional data, perform simulations using
   - Gaussian kernel models and least-squares learning

   and analyze the results when, e.g.,
   - target functions, number of samples, noise level, and width of Gaussian kernel

   are changed.

- Tips: If matrix $\boldsymbol{K}$ is unstable to invert, you may add a small positive constant to diagonals.
- Deadline: Next class