Nonparametric Methods

단국대학교 산업공학과

홍정윤

jyoon333@dankook.ac.kr

COIS COMPUTATIONAL OPTIMIZATION FOR INTELLIGENT SYSTEMS

CONTENTS

01 Nonparametric Methods

- **Parametric Methods**
  - ✓ 특정 분포를 가정한 뒤, density estimation 진행
  - ✓ Ex) Linear regression, Logistic regression, Neural network 등

- **Nonparametric Methods**
  - ✓ 특정 분포를 가정하지 않고, density estimation 진행
  - ✓ Ex) Histograms, Kernel Density Estimation, Nearest Neighbor 등

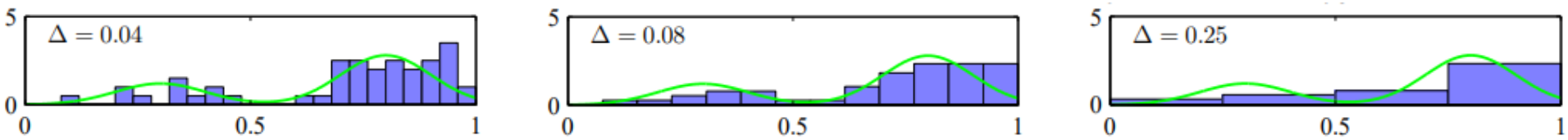
02 Histogram Estimation

- **Density estimation for histogram**
  - ✓ 데이터로부터 histogram 생성 후, PDF를 구함

$$p_i = \frac{n_i}{N\Delta_i}$$

$p(x)$: PDF 함수
p_i : i 구간 내, PDF 추정 수식
n_i : i 구간에 속하는 데이터 수
N : 전체 데이터 수
Δ_i : 일정한 구간으로 나눈 폭

  - ✓ Δ값에 따른 분포 형태



02 Histogram Estimation

- **Density estimation for histogram**
  - ✓ 한계점
    1. 구간의 경계에서 불연속성 발생
    2. 고차원 데이터의 경우 차원의 저주(curse of dimensionality) 문제 발생
       - D차원의 공간에서 각 차원에 대해 M개의 구간으로 나누면, 전체 구간의 수 = M^D개

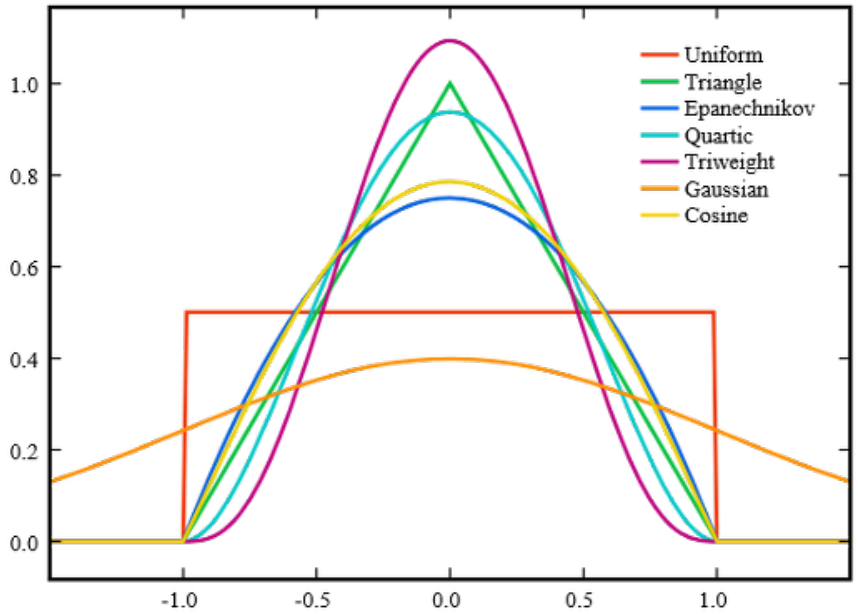
03 Kernel Density Estimation

- **Kernel function**

$$\int_{-\infty}^{\infty} K(u)du = 1$$

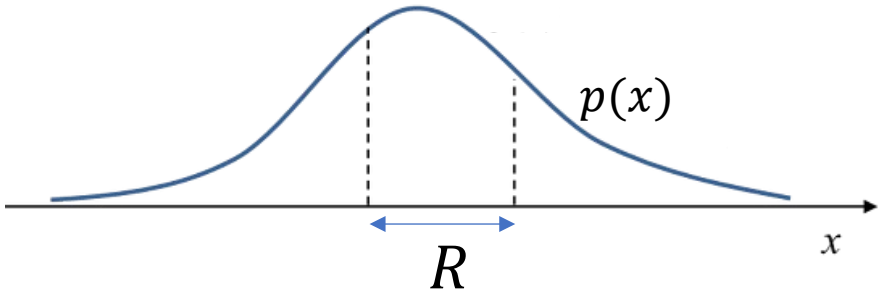
$$K(u) = -K(u)$$

$$K(u) \geq 0$$



- **Kernel Density Estimation**
  - ✓ $p(x)$로부터 추출된 데이터가, R에 속할 확률

$$P = \int_R p(x)dx$$



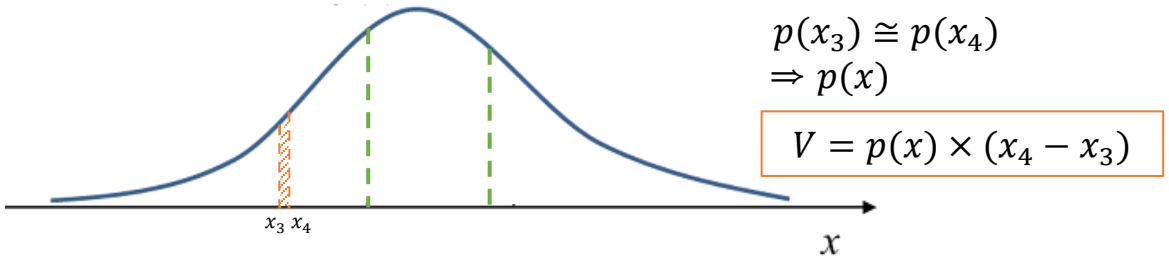
  - ✓ $P(k) = \binom{N}{K} P^k (1-P)^{N-k}$, $\quad E[k] = \mathrm{N}P, Var[k] = NP(1-P)$

$$E\left[\frac{k}{N}\right] = P, Var\left[\frac{k}{N}\right] = \frac{P(1-P)}{N}$$

03 Kernel Density Estimation

- **Kernel Density Estimation**
  - ✓ $N \rightarrow \infty$, $P \cong \frac{k}{N}$
  - ✓ R이 충분히 작아서 $p(x)$가 급격하게 변하지 않는다고 가정하면,

$$P = \int_R p(x)dx \cong p(x)V$$



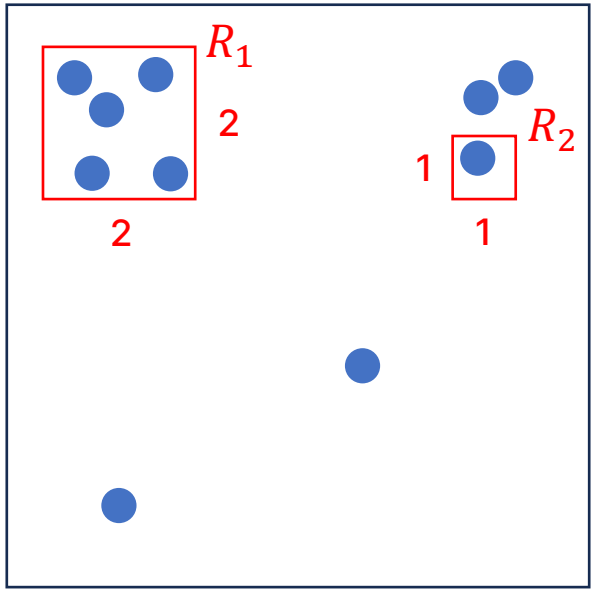
  - ✓ $P = \int_R p(x)dx \cong p(x)V = \frac{k}{N}$
    $\Rightarrow p(x) = \frac{k}{NV}$

Kernel Density Estimation

- **Kernel Density Estimation**
  - ✓ $p(x) = \frac{k}{NV}$

$$R_1 : p(x) = \frac{5}{10 \times 2^2}$$

$$R_2 : p(x) = \frac{1}{10 \times 1^2}$$



03 Kernel Density Estimation

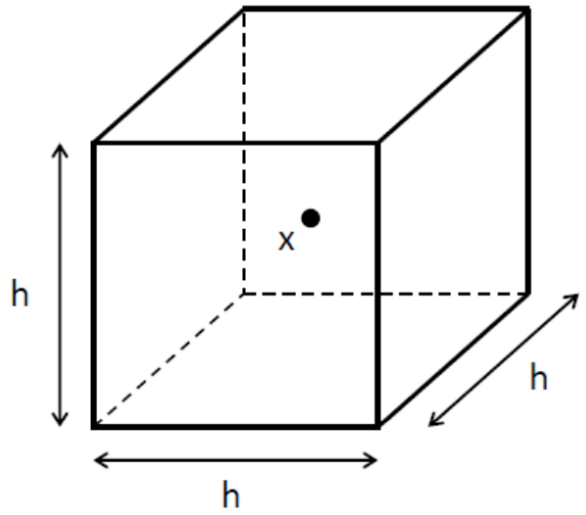
- **Kernel Density Estimation**

$$p(x) = \frac{k}{NV}$$

$V : volume\ surrounding\ x$
$N : the\ total\ number\ of\ data$
$k : the\ number\ of\ data\ inside\ V$

- ✓ N이 클수록, V가 작을수록 estimation이 정확해짐
- ✓ N은 고정값, 적절한 V를 찾아야 함
- ✓ V를 고정하고, k를 찾아서 pdf를 계산 ⇒ kernel density estimation
- ✓ k를 고정하고, V를 찾아서 pdf를 계산 ⇒ KNN density estimation

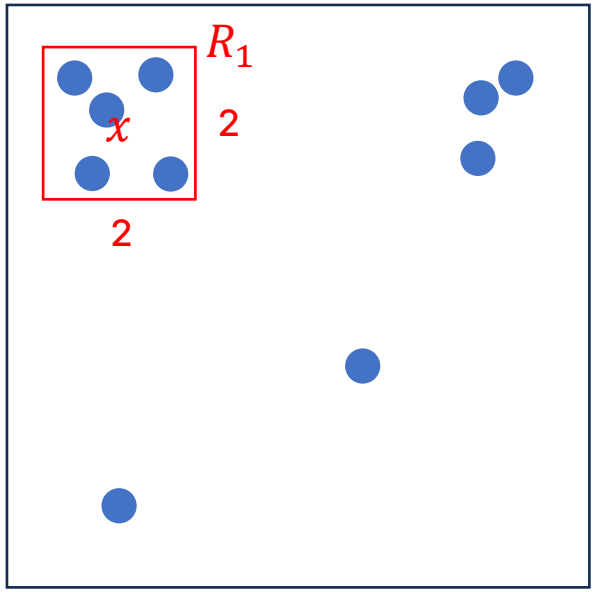
- **Parzen Window Density Estimation**
  - ✓ $p(x)$의 x가 무게중심이며, 각 변의 길이가 h인 hypercube를 정의
    - → d차원 일 때, hypercube의 $V = h^d$



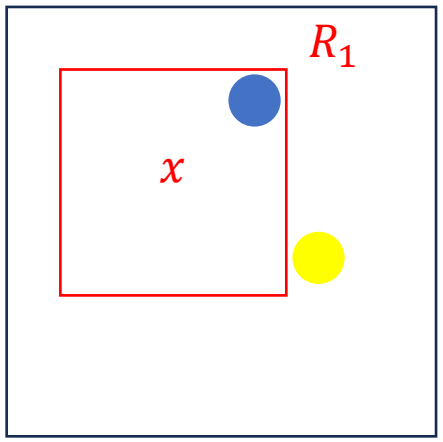
- **Parzen Window Density Estimation**
  - ✓ Kernel function - $K(u)$ 정의

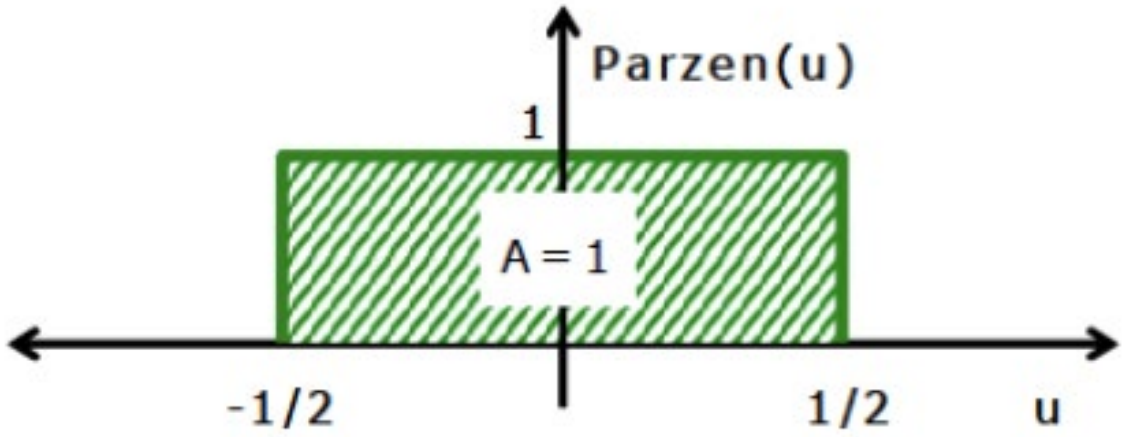
$$K(u) = \begin{cases} 1 & |u_j| < \frac{1}{2} \ \forall i = 1, \dots, d \\ 0 & otherwise \end{cases}$$

$$k = \sum_{i=1}^{N} K\left(\frac{x_i - x}{h}\right) \qquad p(x) = \frac{1}{Nh^d} \sum_{i=1}^{N} K\left(\frac{x_i - x}{h}\right)$$



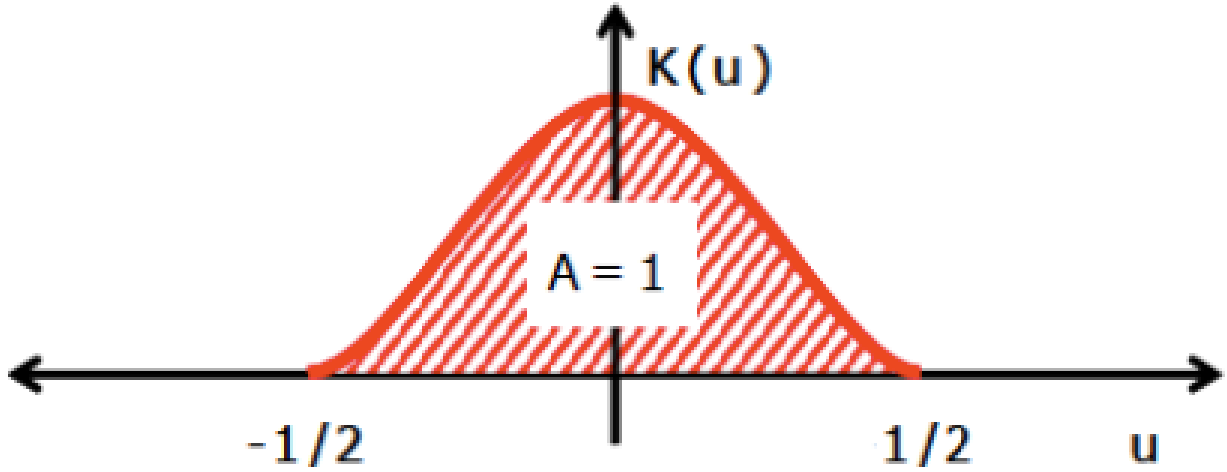
- Parzen Window Density Estimation
  - ✓ 불연속성 문제 존재





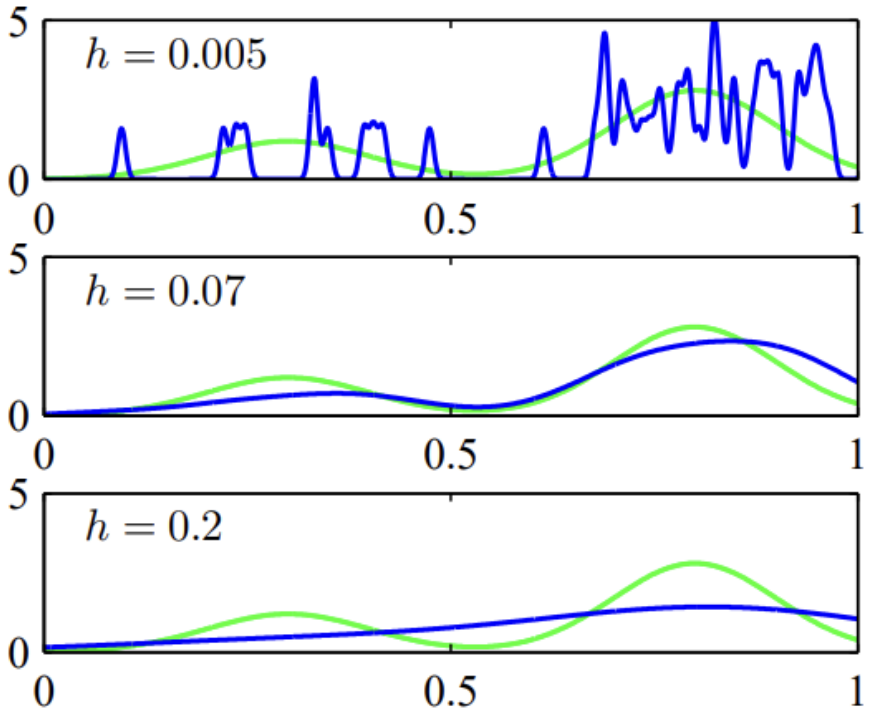
03 Kernel Density Estimation

- **Parzen Window Density Estimation**
  - ✓ Smooth kernel function
    - → Gaussian kernel



03 Kernel Density Estimation

- **Parzen Window Density Estimation**
  - ✓ Smoothing parameter : h



04 Nearest-neighbour

- **Nearest-neighbour**

$$p(x) = \frac{k}{NV}$$

$V : volume\ surrounding\ x$
$N : the\ total\ number\ of\ data$
$k : the\ number\ of\ data\ inside\ V$

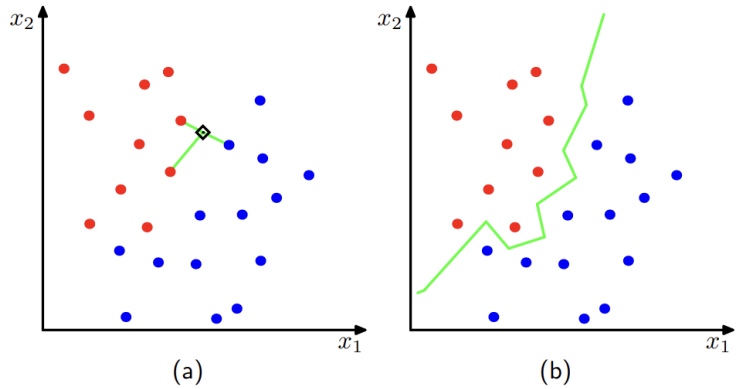
✓ k를 고정하고, V를 찾아서 pdf를 계산 ⇒ KNN density estimation

✓ V 정의

→ x를 중심으로 k개의 데이터 포인트를 포함할 때까지의 영역을 확장

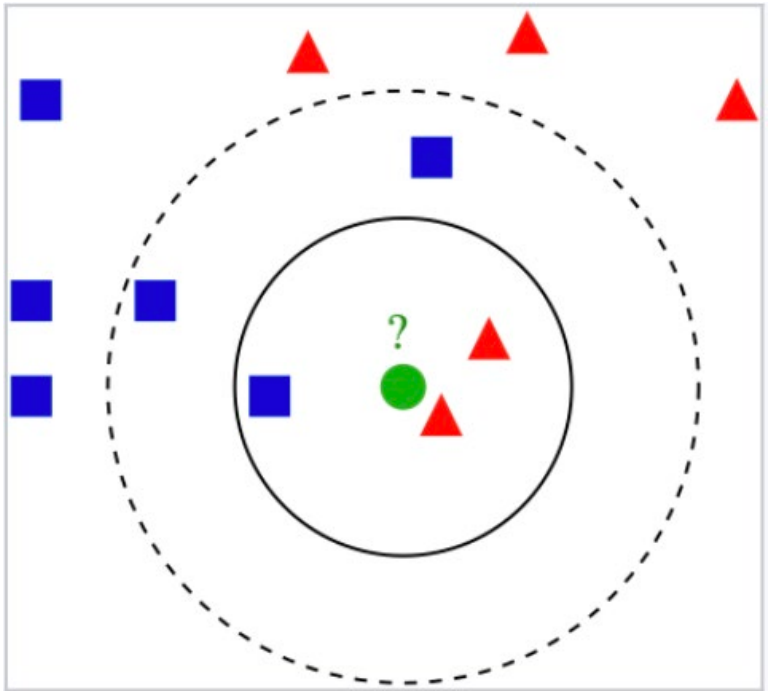
04 Nearest-neighbour

- **Nearest-neighbour**
  - ✓ Bayes' theorem
  1. 데이터 집합 가정 $C_k : \sum_k N_k = N$
  2. New point x 분류
     1. C_k로부터 K_k의 데이터 포인트를 가질 때, 각 C_k에 대한 밀도 추정 수식 $p(x|C_k) = \frac{K_k}{N_k V}$
     2. 전체 영역에 대한 밀도 추정 수식 $p(x) = \frac{K}{NV}$
     3. 각 C_k에 대한 밀도 추정 수식 $p(C_k) = \frac{N_k}{N}$
     4. Bayes' theorem 적용 $p(C_k|x) = \frac{p(x|C_k)p(C_k)}{p(x)} = \frac{K_k}{K}$



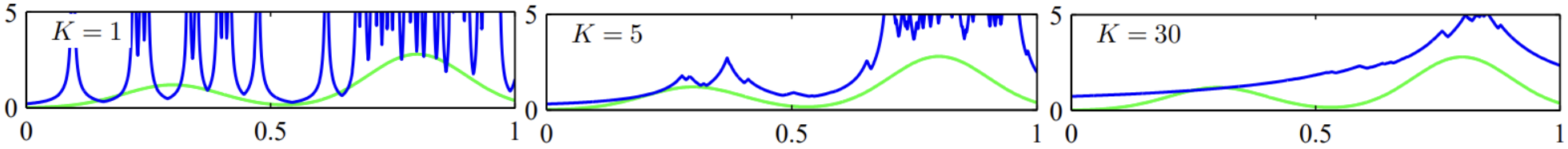
04 Nearest-neighbour

- **K-nearest neighbour**
  - ✓ Algorithm
    1. 분류할 x를 선택
    2. x로부터 인접한 k개의 학습 데이터 탐색
    3. 탐색된 k개 학습 데이터의 majority class에 할당
    4. 할당된 class를 x의 분류 결과로 반환



04 Nearest-neighbour

- **K-nearest neighbour**
  - ✓ Hyperparameter
    - → k



    - → Distance measures

04 Nearest-neighbour

- **K-nearest neighbour**
  - ✓ 장점
    - → 데이터 noise에 강건함
    - → 데이터 수가 많을 때, 효과적임
  - ✓ 한계점
    - → 최적의 hyperparameter 결정
    - → 계산시간이 오래 걸림

감사합니다.

COIS COMPUTATIONAL OPTIMIZATION FOR INTELLIGENT SYSTEMS