

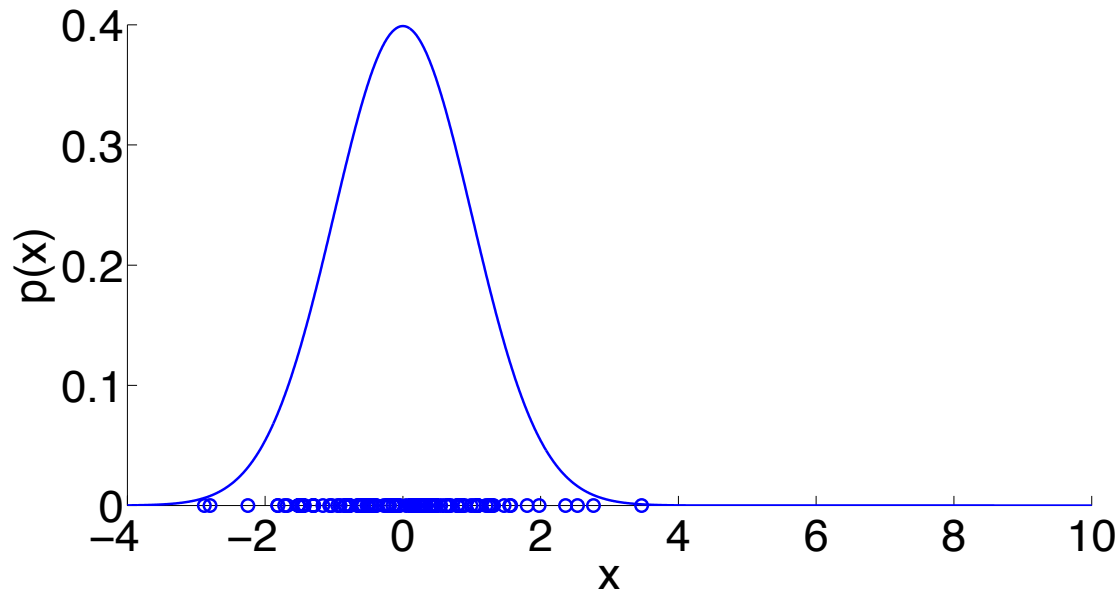
# EE3731C Pattern Recognition 2

BT Thomas Yeo

ECE, CIRC, Sinapse, Duke-NUS, HMS

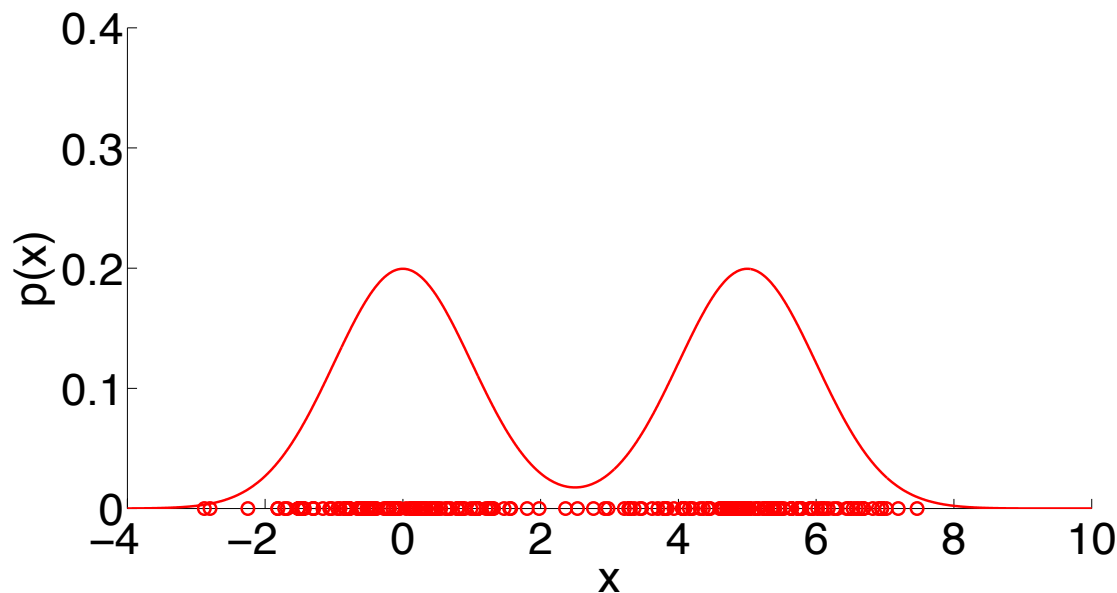
# Overview

- Problem 1: density estimation
  - Given data samples  $x_1, \dots, x_N$ , estimate  $p(x)$
  - Suppose we assume  $p(x)$  is Gaussian with variance 1, then from previous class, ML estimate of mean  $\mu_{ML} = \frac{1}{N} \sum_n x_n$ , and so estimate of  $p(x)$  is  $\mathcal{N}(\mu_{ML}, 1)$



# Overview

- Problem 1: density estimation
  - Given data samples  $x_1, \dots, x_N$ , estimate  $p(x)$
  - Suppose we assume  $p(x)$  is Gaussian with variance 1, then from previous class, ML estimate of mean  $\mu_{ML} = \frac{1}{N} \sum_n x_n$ , and so estimate of  $p(x)$  is  $\mathcal{N}(\mu_{ML}, 1)$
  - What if we do not know what kind of distribution  $p$  is?



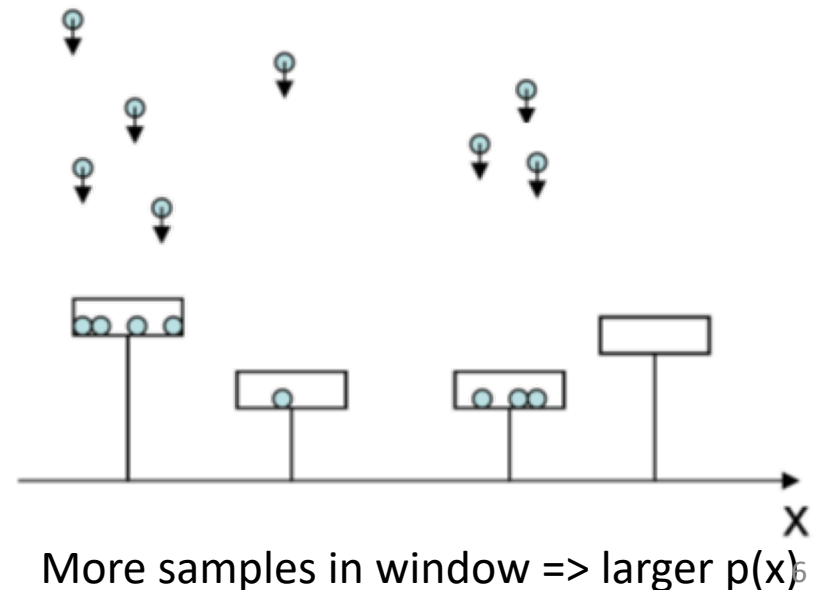
# Overview

- Problem 1: density estimation
  - Given data samples  $x_1, \dots, x_N$ , estimate  $p(x)$
  - Suppose we assume  $p(x)$  is Gaussian with variance 1, then from previous class, ML estimate of mean  $\mu_{ML} = \frac{1}{N} \sum_n x_n$ , and so estimate of  $p(x)$  is  $\mathcal{N}(\mu_{ML}, 1)$
  - What if we do not know what kind of distribution  $p$  is?
- Problem 2: classification
  - Given input-output pairs  $D = \{x_n, y_n\}_{n=1:N}$ , learn mapping  $y = f(x)$
  - What if we do not know what kind of function  $f$  is?
- Today: Non-parametric approach

# Non-parametric Density Estimation: Parzen Window

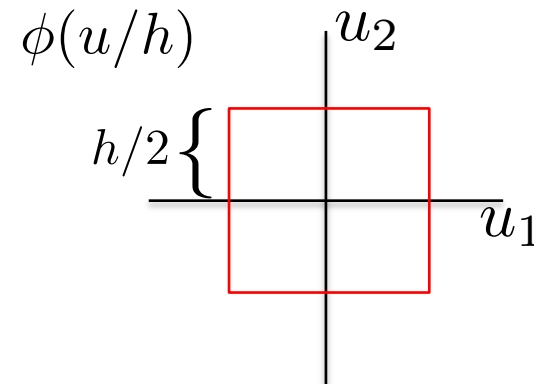
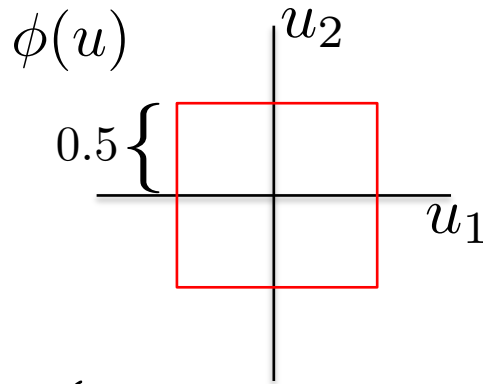
# Overview

- Previously assume parametric probability distributions (e.g. Gaussians), but now assume non-parametric
  - Does not mean no parameter!
  - Number and nature of parameters change with data
  - Less assumptions, but need much more data to work well
- Idea: Given  $x_1, \dots, x_N$  i.i.d. sampled from unknown  $p(x)$ 
  - For particular  $x$ , count number of samples  $k$  falling in window of volume  $V$  centered at  $x$
  - $p(x) \approx \frac{k/N}{V}$
- Two approaches
  - Parzen windows: fix  $V$ , estimate  $k$
  - KNN: fix  $k$ , determine  $V$



# Parzen Window

- Square cube:  $\phi(u) = \begin{cases} 1 & |u_j| \leq 0.5, j = 1, \dots, d \\ 0 & \text{otherwise} \end{cases}$



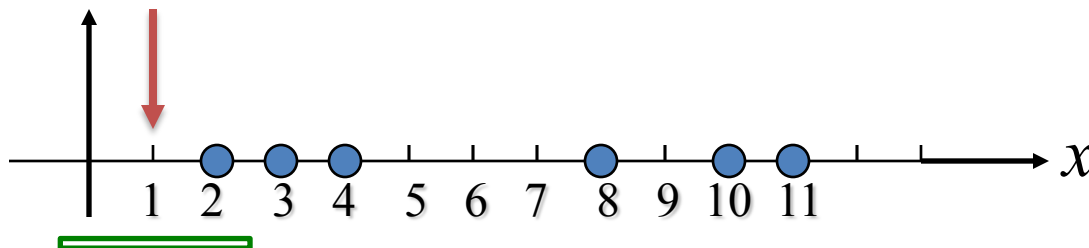
- $\phi\left(\frac{x-x_i}{h}\right) = \begin{cases} 1 & x_i \text{ inside hypercube of width } h \text{ centered at } x \\ 0 & \text{otherwise} \end{cases}$
- $k(x) = \sum_i \phi\left(\frac{x-x_i}{h}\right) = \# \text{ samples in hypercube}$
- Parzen window estimation:

$$p(x) = \frac{k(x)/N}{V} = \frac{\sum_i \phi\left(\frac{x-x_i}{h}\right)/N}{h^d} = \frac{1}{N} \sum_{i=1}^N \frac{1}{h^d} \phi\left(\frac{x-x_i}{h}\right)$$

# Parzen Window as Histogramming

Parzen window estimate:  $p(x) = \frac{1}{N} \sum_{i=1}^N \frac{1}{h^d} \phi\left(\frac{x-x_i}{h}\right)$

- Interpretation 1: At  $x$ , count fraction of samples in hypercube centered at  $x$
- Example:
  - 6 samples at 2, 3, 4, 8, 10, 11



- $d = 1, h = 3, x = 1$ :

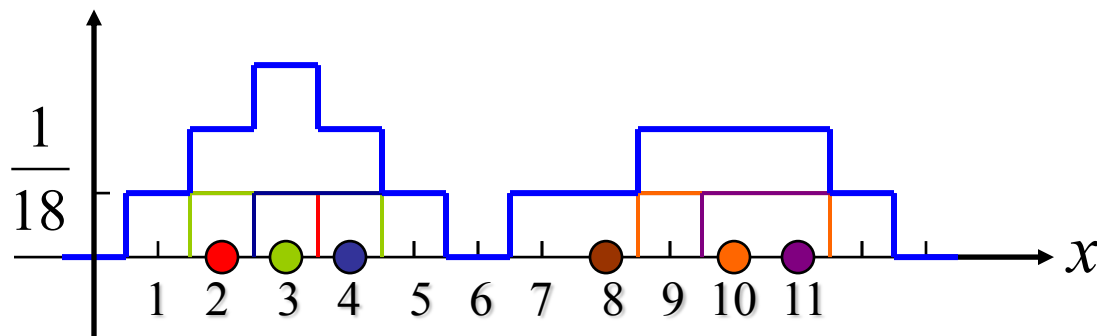
$$p(1) = \frac{1}{6} \sum_{i=1}^6 \frac{1}{3} \phi\left(\frac{1-x_i}{3}\right) = \frac{1}{18} [1 + 0 + 0 + 0 + 0 + 0] = \frac{1}{18}$$



# Parzen Window as Interpolation

Parzen window estimate:  $p(x) = \frac{1}{N} \sum_{i=1}^N \frac{1}{h^d} \phi\left(\frac{x-x_i}{h}\right)$

- Interpretation 2: At  $x$ , sum of  $N$  hypercubes centered at  $x_i$



- Instead of hypercube, can use other windows (e.g., Gaussians):

$$\phi(u) \geq 0 \text{ and } \int \phi(u) du = 1,$$

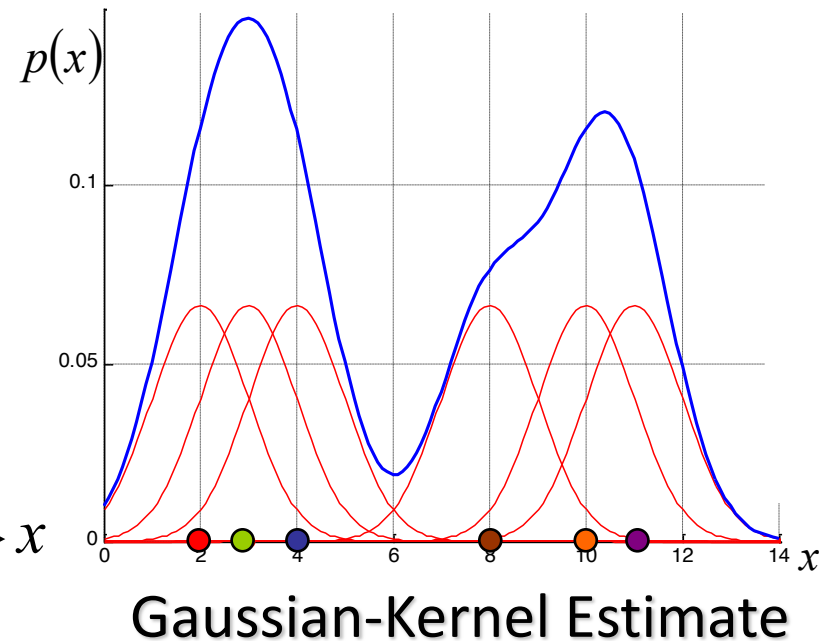
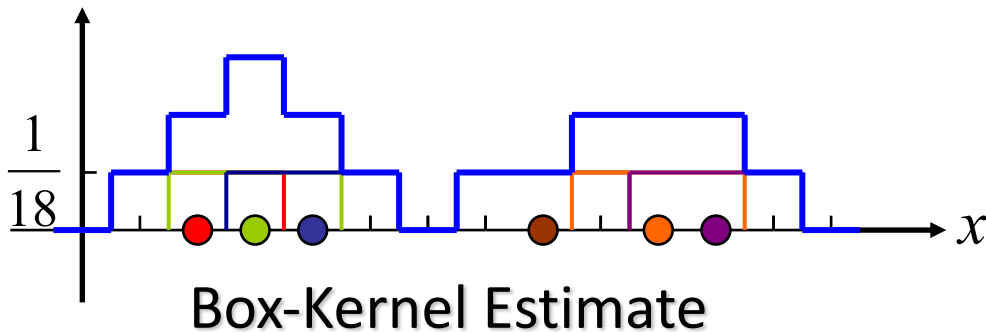
guaranteeing  $p(x) \geq 0$  and integrates to 1 because  $\frac{1}{h^d} \int \phi\left(\frac{x-x_i}{h}\right) dx = 1$

- Window function used for interpolation:  $p(x)$  is weighted average of samples

# Smooth Windows

Parzen window estimate:  $p(x) = \frac{1}{N} \sum_{i=1}^N \frac{1}{h^d} \phi\left(\frac{x-x_i}{h}\right)$

- With smooth windows (e.g., Gaussians), resulting  $p(x)$  is smooth

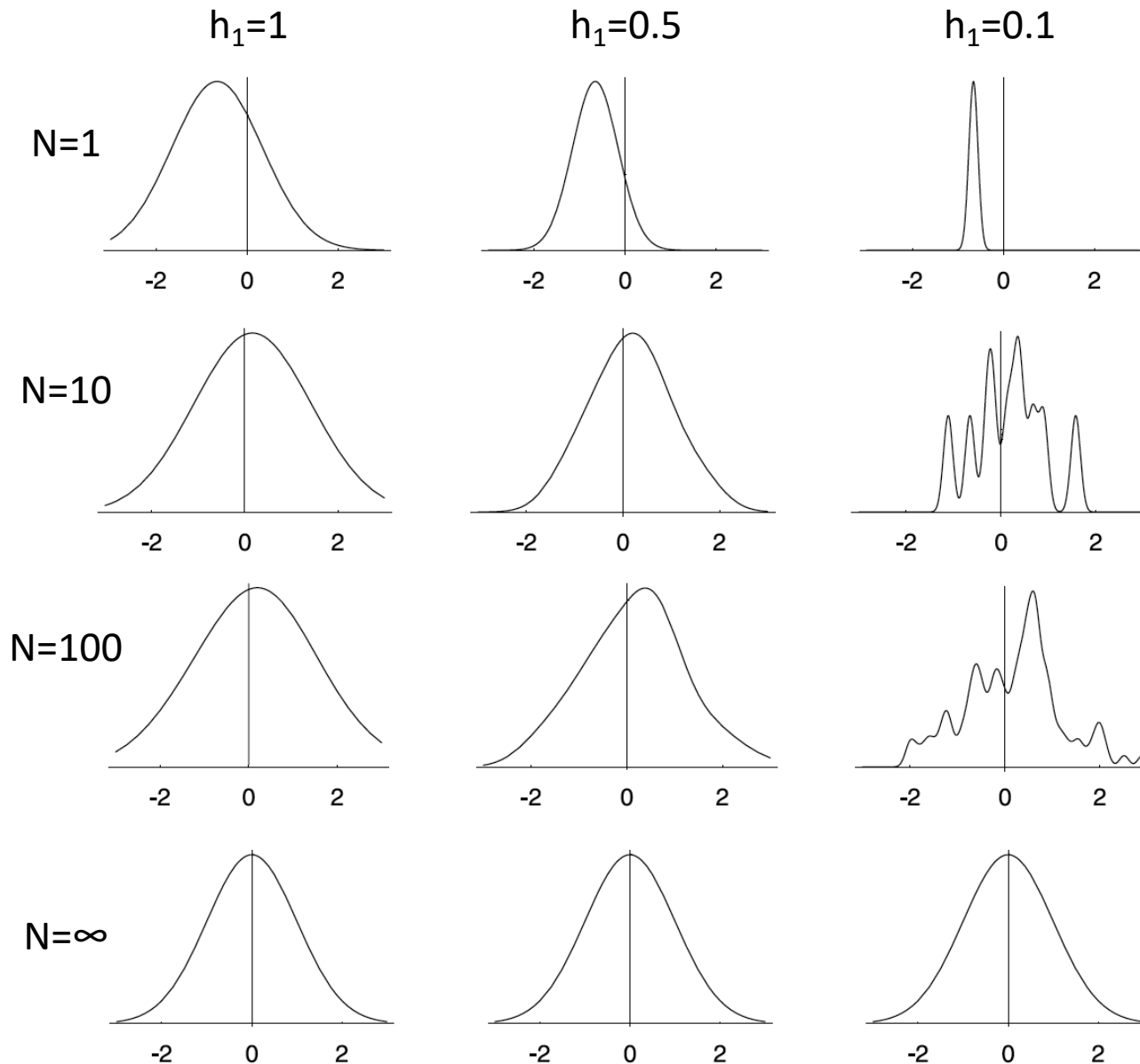


# Convergence Conditions

Parzen window estimate:  $p_N(x) = \frac{k_N(x)/N}{V_N} = \frac{1}{N} \sum_{i=1}^N \frac{1}{h_N^d} \phi\left(\frac{x-x_i}{h_N}\right)$

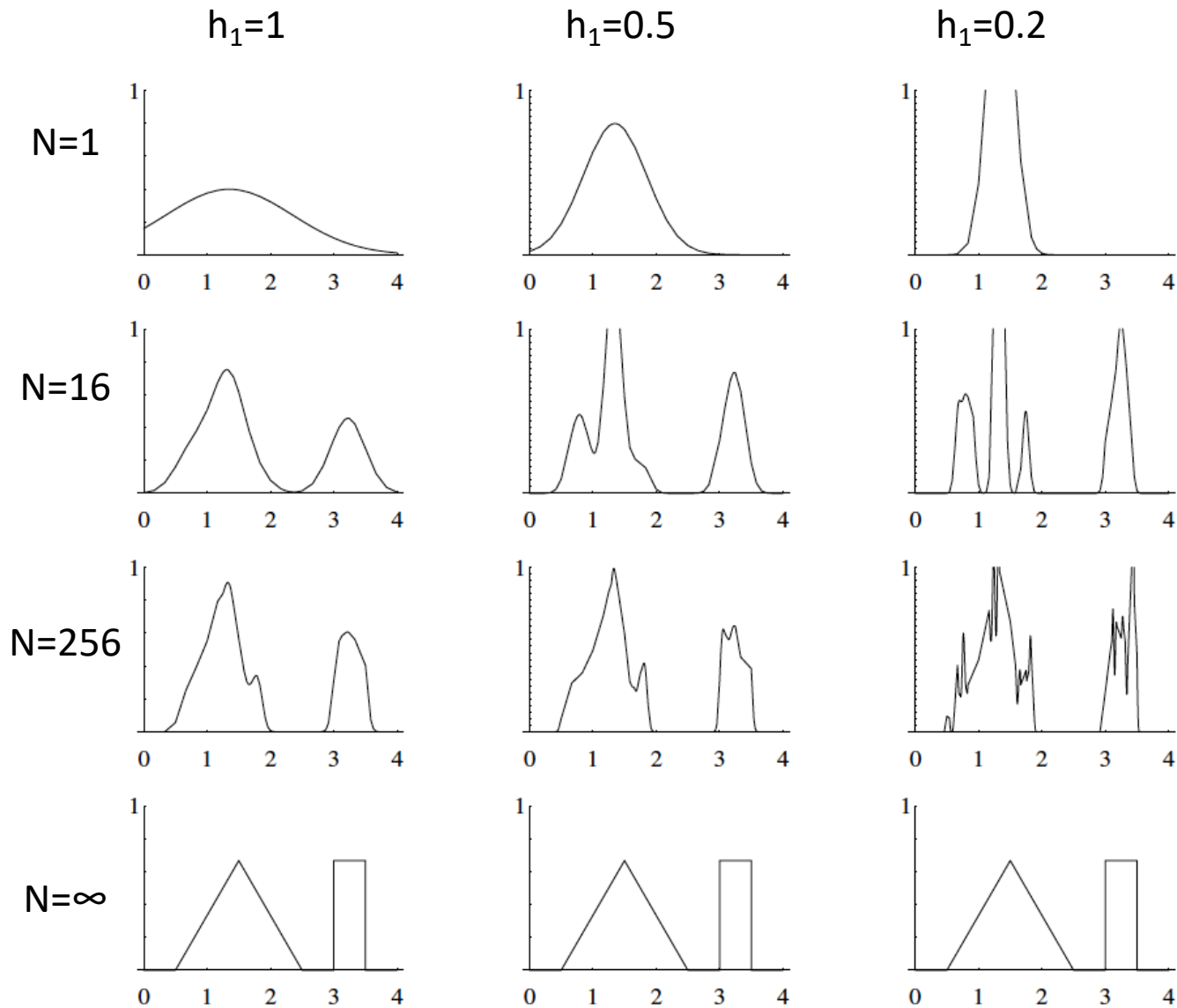
- As  $N \rightarrow \infty$ , want  $p_N(x) \rightarrow \text{true } p(x)$
- 4 sufficient conditions
  - $\sup_u \phi(u) < \infty$
  - $\lim_{||u|| \rightarrow \infty} \phi(u) \prod_{i=1}^d u_i = 0$
  - $\lim_{N \rightarrow \infty} V_N = 0$
  - $\lim_{N \rightarrow \infty} NV_N = \infty$
- First two conditions easy to satisfy
- Last two conditions  $\implies$  volume around data sample must fall to 0, but at a rate slower than  $1/N$

# $N(0, 1)$ Window to Estimate $N(0, 1)$



$$h_N = h_1 / \sqrt{N}$$

# $N(0, 1)$ Window to Estimate Bimodal Distribution



# Setting Window Size $h$

Parzen window estimate:  $p(x) = \frac{1}{N} \sum_{i=1}^N \frac{1}{h^d} \phi\left(\frac{x-x_i}{h}\right)$

- In practice,  $N$  fixed, need to set  $h$
- Can use cross-validation
  - Divide training data into training and validation set
  - For different  $h$ , compute estimate  $p_h(x)$  from training set
  - “Test” on validation set:  $\sum_i \log p_h(x_i)$ , where  $x_i \in$  validation set
  - Pick  $h$  with highest log probability on validation set

# K-Nearest Neighbor

# K-Nearest Neighbors (KNN) Density Estimation

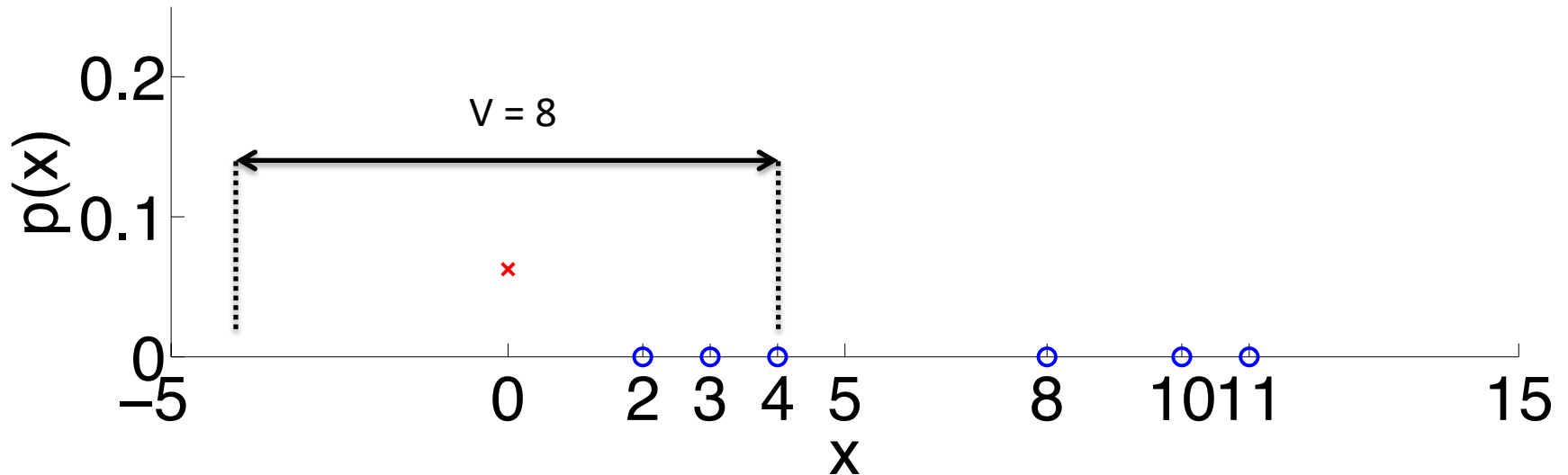
KNN estimate:  $p_N(x) = \frac{k_N(x)/N}{V_N}$

- Rather than fix window size, grows volume around  $x$  until  $k_N(x)$  samples
- If  $p(x)$  high, then volume will be small; if  $p(x)$  low, then volume will be big
- Convergence:  $p_N(x) \rightarrow p(x) \iff \begin{cases} \lim_{N \rightarrow \infty} k_N = \infty \\ \lim_{N \rightarrow \infty} k_N/N = 0 \end{cases}$



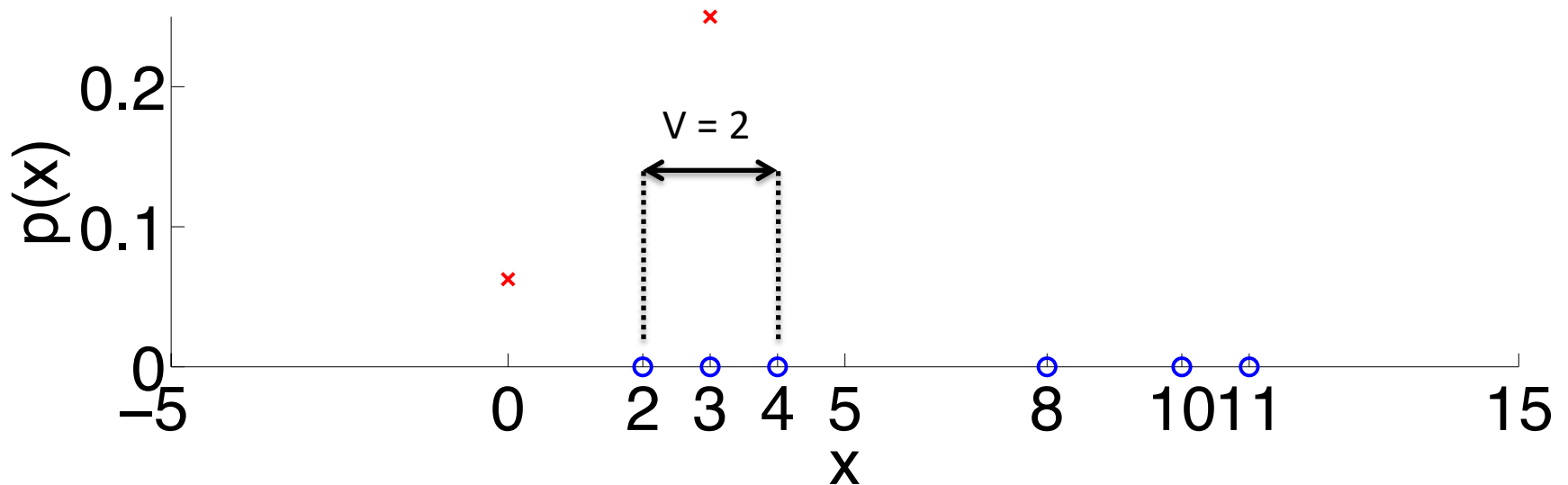
# KNN Example

- $\{x_n\}_{n=1:6} = \{2, 3, 4, 8, 10, 11\}$ ,  $K = 3$
- At  $x = 0$ ,  $V = 8$ , so  $p(x) = (K/N)/V = (3/6)/8 = 1/16$



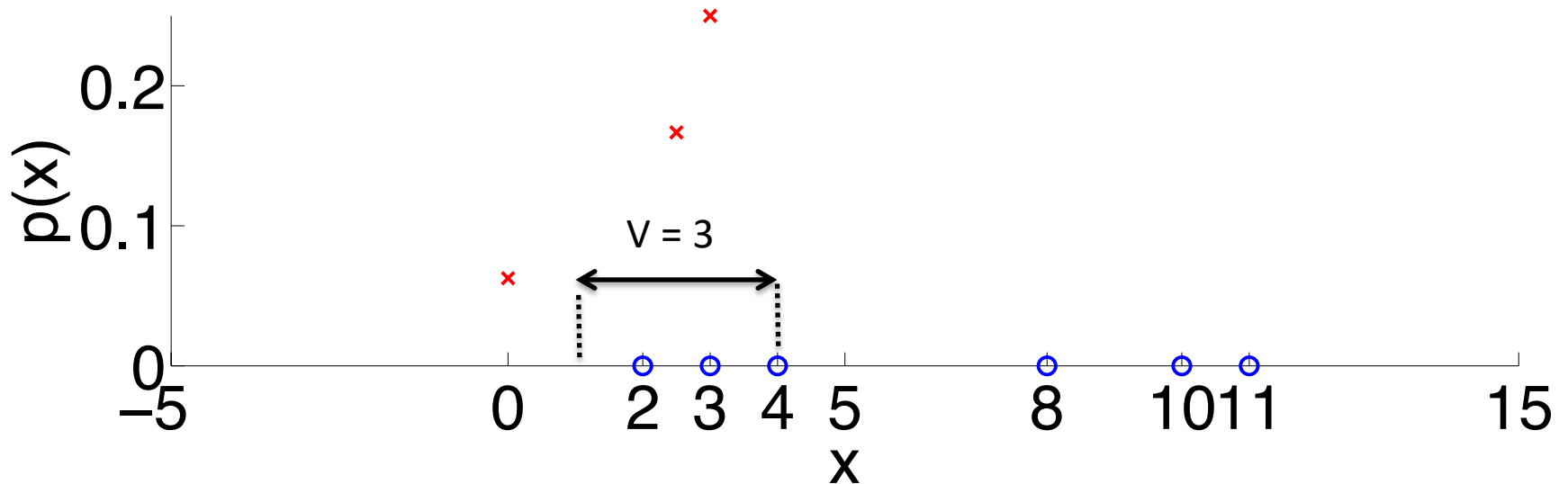
# KNN Example

- $\{x_n\}_{n=1:6} = \{2, 3, 4, 8, 10, 11\}$ ,  $K = 3$
- At  $x = 0$ ,  $V = 8$ , so  $p(x) = (K/N)/V = (3/6)/8 = 1/16$
- At  $x = 3$ ,  $V = 2$ , so  $p(x) = (K/N)/V = (3/6)/2 = 1/4$



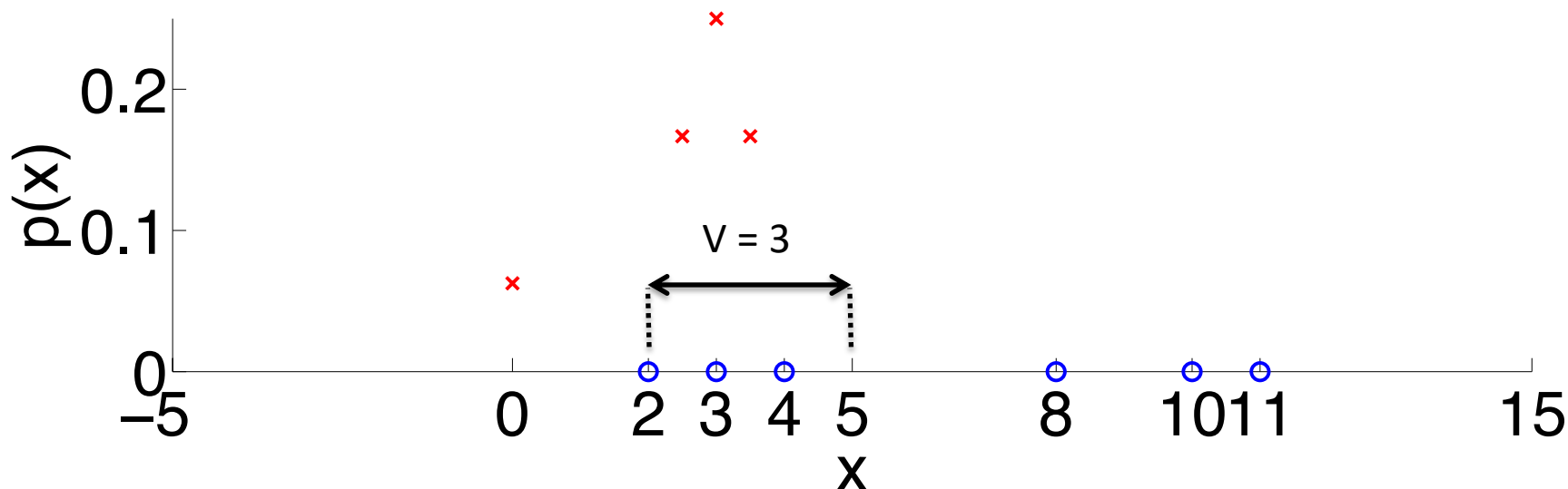
# KNN Example

- $\{x_n\}_{n=1:6} = \{2, 3, 4, 8, 10, 11\}$ ,  $K = 3$
- At  $x = 0$ ,  $V = 8$ , so  $p(x) = (K/N)/V = (3/6)/8 = 1/16$
- At  $x = 3$ ,  $V = 2$ , so  $p(x) = (K/N)/V = (3/6)/2 = 1/4$
- At  $x = 2.5$ ,  $V = 3$ , so  $p(x) = (K/N)/V = (3/6)/3 = 1/6$



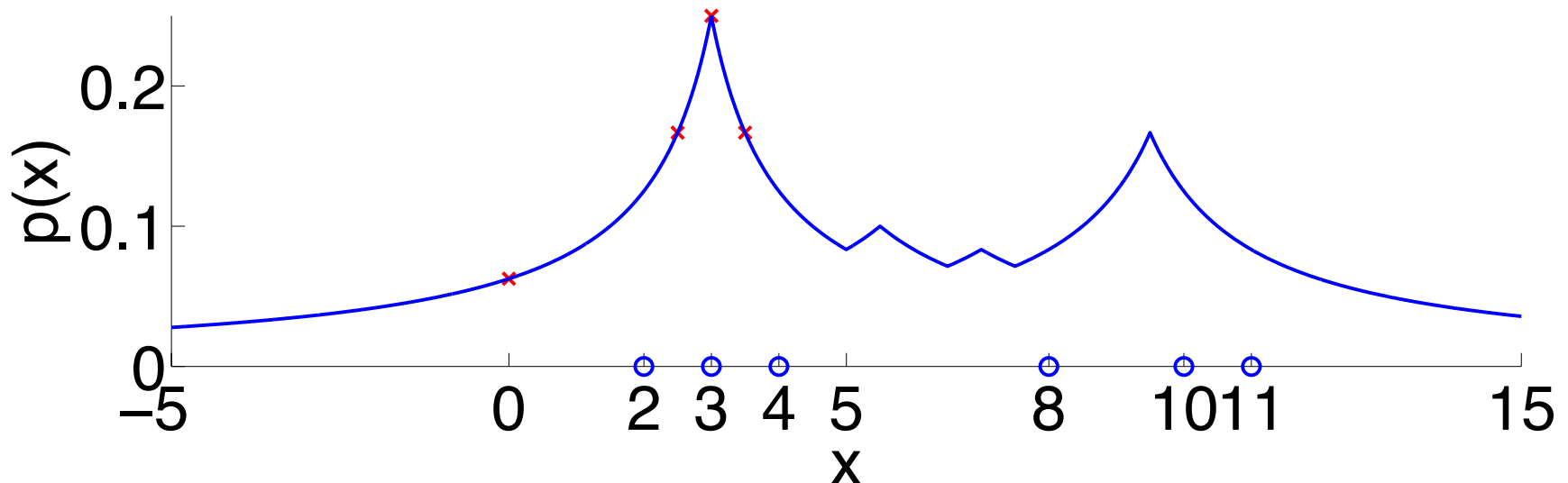
# KNN Example

- $\{x_n\}_{n=1:6} = \{2, 3, 4, 8, 10, 11\}$ ,  $K = 3$
- At  $x = 0$ ,  $V = 8$ , so  $p(x) = (K/N)/V = (3/6)/8 = 1/16$
- At  $x = 3$ ,  $V = 2$ , so  $p(x) = (K/N)/V = (3/6)/2 = 1/4$
- At  $x = 2.5$ ,  $V = 3$ , so  $p(x) = (K/N)/V = (3/6)/3 = 1/6$
- At  $x = 3.5$ ,  $V = 3$ , so  $p(x) = (K/N)/V = (3/6)/3 = 1/6$

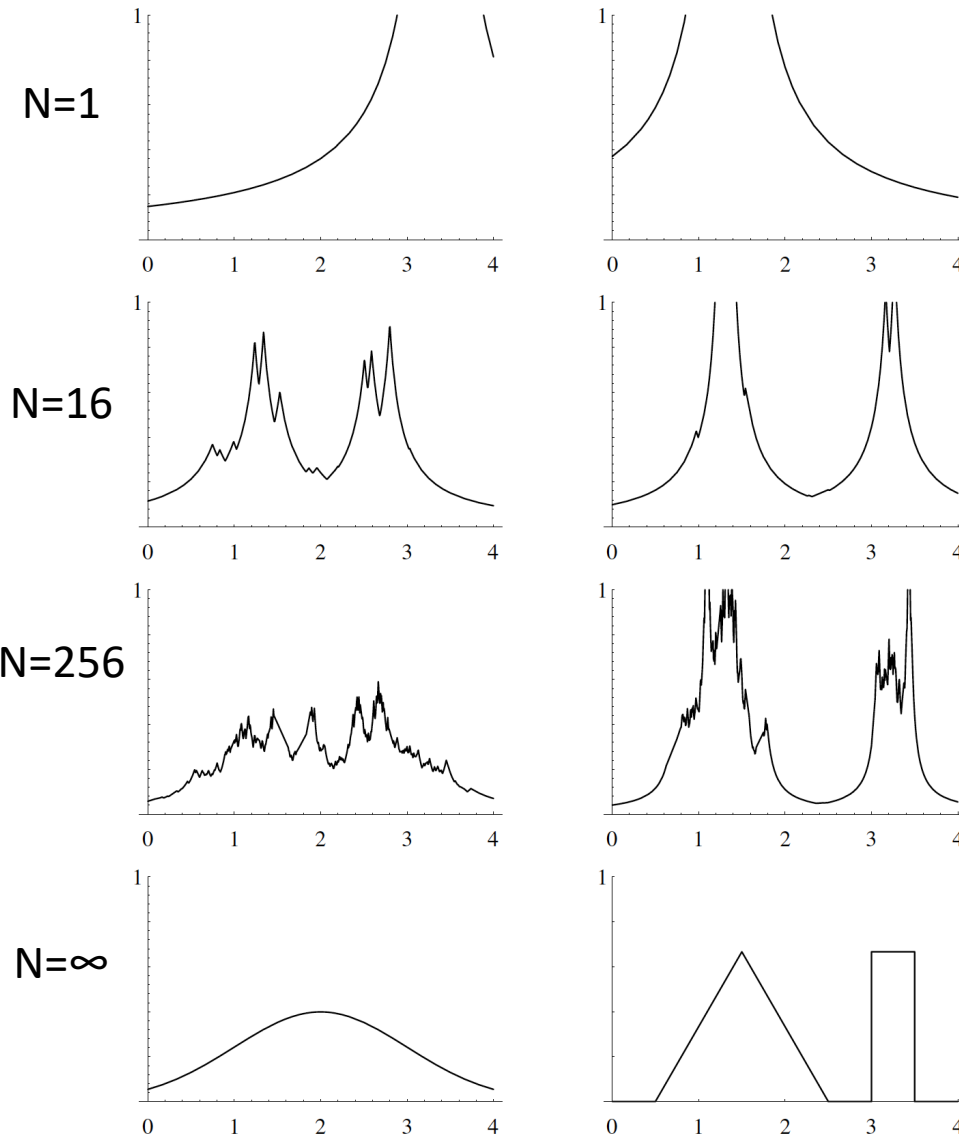


# KNN Example

- $\{x_n\}_{n=1:6} = \{2, 3, 4, 8, 10, 11\}$ ,  $K = 3$
- At  $x = 0$ ,  $V = 8$ , so  $p(x) = (K/N)/V = (3/6)/8 = 1/16$
- At  $x = 3$ ,  $V = 2$ , so  $p(x) = (K/N)/V = (3/6)/2 = 1/4$
- At  $x = 2.5$ ,  $V = 3$ , so  $p(x) = (K/N)/V = (3/6)/3 = 1/6$
- At  $x = 3.5$ ,  $V = 3$ , so  $p(x) = (K/N)/V = (3/6)/3 = 1/6$
- In general,  $p(x)$  continuous but not differentiable
- In general,  $p(x)$  may not be proper distribution ( $\int p(x)dx = \infty$ )



# Estimating $N(0, 1)$ and Bi-modal Distribution



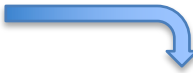
$$k_N = k_1 \sqrt{N}$$

$$k_1 = 1$$

Results do not look very impressive. For density estimation, most people use Parzen's approach. But KNN used for classification instead (see later slides)

# Non-parametric Classification

# Classification (Supervised Learning)

- Given features  $x$ , want to estimate label  $y$
- Estimate joint density  $p(x, y)$  non-parametrically
- Suppose volume  $V$  around  $x$  capture  $K$  samples,  $k_c$  were from class  $y = c$ 
  - Joint probability  $p(x, y = c) = \frac{k_c/N}{V}$  
  - Posterior  $p(y = c|x) = \frac{p(x, y=c)}{\sum_{c'=1}^C p(x, y=c')} = \frac{(k_c/N)/V}{\sum_{c'=1}^C (k_{c'}/N)/V} = \frac{k_c}{\sum_{c'=1}^C k_{c'}} = \frac{k_c}{K}$
- Therefore posterior probability of class  $c$  is simply fraction of neighbors with class label  $c$ 
  - MAP estimation: output class  $c$  with highest posterior probability
- Above can be performed using Parzen's window or KNN
  - Parzen's window approach: Fix volume  $V$  and count the number of captured samples  $K$
  - KNN approach of fixing  $K$
- In practice, most people use KNN



# KNN Example

- Suppose we have training set consisting of  $N$  images & each image is labeled as a human, cat or dog
- Given a test image  $x$ , and suppose we set  $K = 5$ 
  - Out of  $N$  training images, we find 5 images closest to the test image
  - Of these 5 images, let's say 3 images are humans (i.e.,  $k_{\text{human}} = 3$ ), 1 image is a cat (i.e.,  $k_{\text{cat}} = 1$ ) and 1 image is a dog (i.e.,  $k_{\text{dog}} = 1$ ).
  - Then  $p(y = \text{human} \mid x) = k_{\text{human}}/K = 3/5$
  - $p(y = \text{cat} \mid x) = k_{\text{cat}}/K = 1/5$
  - $p(y = \text{dog} \mid x) = k_{\text{dog}}/K = 1/5$
  - So the MAP estimate of  $x$  is human

# Setting # Neighbors K

KNN estimate:  $p(y = c|x) = \frac{k_c}{K}$

- In practice,  $N$  fixed, need to set  $K$
- Can use cross-validation
  - Given training data  $\{x_n, y_n\}_{n=1:N}$ , divide into training  $(\{x'_i, y'_i\}_{i=1:N_1})$  and validation  $(\{x''_k, y''_k\}_{k=1:N_2})$  sets
  - For different  $K$ , use training set to classify class label on validation set:  $\hat{y}''_k = \operatorname{argmax}_c p(y = c|x''_k)$ , which can then be compared with true class label  $y''_k$
  - Pick  $K$  with lowest error on validation set
  - Note that test set was never touched

# KNN Computational Complexity & Metrics

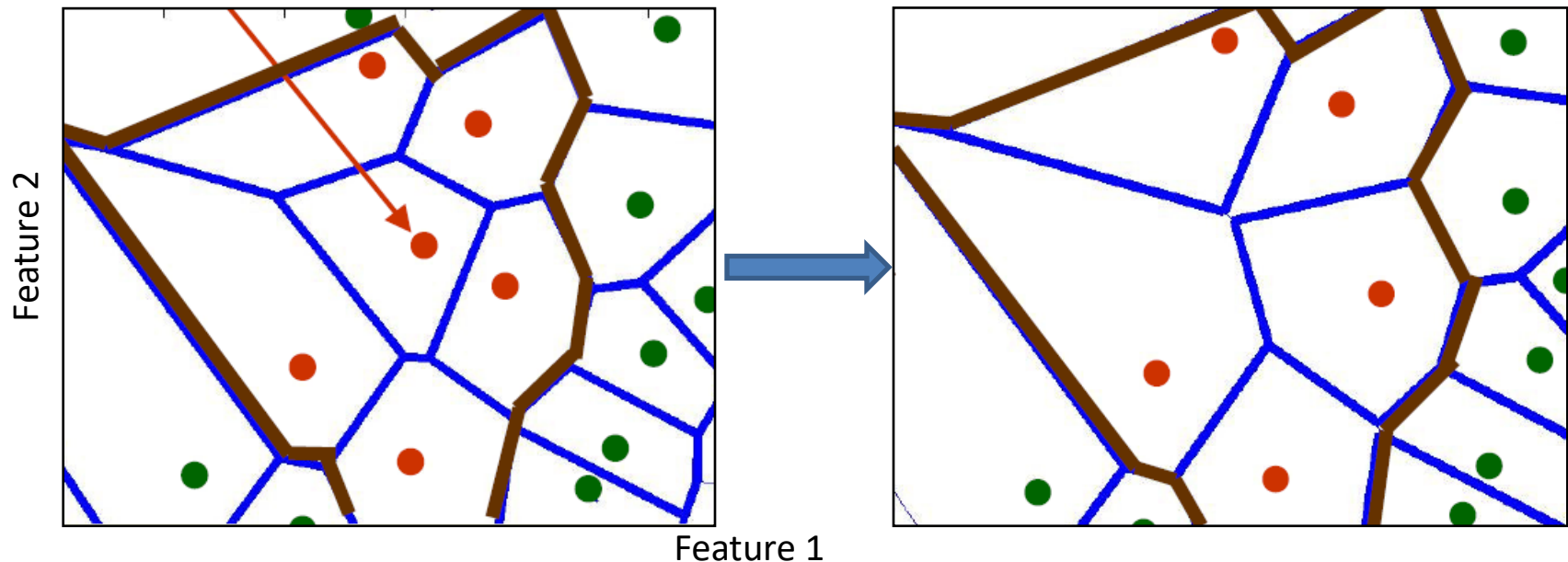
- Suppose features are D-dimensional => Compute distance between two samples require  $O(D)$  operations
- For test sample, to find closest neighbor among N training samples, need to compute distance between test sample and each training sample =>  $N \times O(D) = O(ND)$  operations
- To speed up:
  - Pruning/editing: for 1-NN, can remove data point surrounded by neighbors of the same class, resulting in subset of points (prototypes) used for classification

Red dots: class 1

Green dots: class 2

Blue lines: equidistant between 2 or more dots

Brown lines: 1-NN decision boundary



# KNN Computational Complexity & Metrics

- Suppose features are D-dimensional => Compute distance between two samples require  $O(D)$  operations
- For test sample, to find closest neighbor among N training samples, need to compute distance between test sample and each training sample =>  $N \times O(D) = O(ND)$  operations
- To speed up:
  - Pruning/editing: for 1-NN, can remove data point surrounded by neighbors of the same class, resulting in subset of points (prototypes) used for classification
  - Special data structure, e.g., [https://en.wikipedia.org/wiki/K-d\\_tree](https://en.wikipedia.org/wiki/K-d_tree)
- Distance metrics
  - Minkowski  $dist(a, b) = \left( \sum_{j=1}^D |a_j - b_j|^p \right)^{1/p}$ , where  $j$  indexes feature dimension
  - Euclidean if  $p = 2$ , Manhattan if  $p = 1$

# Summary

- Parzen window for density estimation
  - Interpretation 1: at  $x$ , count fraction of samples in volume centered at  $x$
  - Interpretation 2: at  $x$ ,  $p(x)$  is given by weighted average of neighbors, where weight is larger if neighbor is closer
- KNN neighbors for classification
  - $p(y = c|x)$  is fraction of  $K$  neighbors (in neighborhood of  $x$ ) that are from class  $c$

## Optional Reading

- Duda & Hart Chapter 4.1-4.4, 4.5.5
- Some figures taken from Dr. Tam and Dr. Sun

# Additional Material

# Convergence Proof of Parzen Window

- Let's establish some notations and explain what we mean by convergence
- Let  $\delta_n(x) = \frac{1}{h_n^d} \phi(\frac{x}{h_n})$ , where  $\phi(x) \geq 0$  and  $\int \phi(x) dx = 1$
- Parzen window estimate:  $p_n(x) = \frac{1}{n} \sum_{i=1}^n \delta_n(x - x_i)$
- $\lim_{h_n \rightarrow 0} \delta_n(x - x_i)$  is dirac delta function centered at  $x_i$  as  $h_n \rightarrow 0$ , i.e.,  $V_n \rightarrow 0$
- $p_n(x)$  depends on the samples  $x_1, \dots, x_n$ , so  $p_n(x)$  is also random with some mean  $\bar{p}_n(x)$  and variance  $\sigma_n^2(x)$ .
- We say  $p_n(x)$  converges to  $p(x)$  in the mean square sense if  $\lim_{n \rightarrow \infty} \bar{p}_n(x) = p(x)$  and  $\lim_{n \rightarrow \infty} \sigma_n^2(x) = 0$



# Convergence of Mean

- Let's consider convergence of the mean

$$\begin{aligned}\bar{p}_n(x) &= E[p_n(x)] \\ &= E\left[\frac{1}{n} \sum_{i=1}^n \delta_n(x - x_i)\right] \\ &= \frac{1}{n} \sum_{i=1}^n E[\delta_n(x - x_i)] \\ &= \int \delta_n(x - v) p(v) dv\end{aligned}$$

- From previous slide,  $\lim_{n \rightarrow \infty} \delta_n(x - v)$  is a delta function centered at  $x$  if  $\lim_{n \rightarrow \infty} V_n = 0$
- Therefore  $\lim_{n \rightarrow \infty} \bar{p}_n(x) = p(x)$  assuming  $p(x)$  continuous at  $x$

# Convergence of Variance

- $\sigma_n^2(x)$  is sum of independent variables  $\frac{1}{n}\delta_n(x - x_i)$ , therefore  $\sigma_n^2(x)$  is sum of the variance of the independent variables, so

$$\begin{aligned}\sigma_n^2(x) &= \sum_{i=1}^n E \left[ \left( \frac{1}{n}\delta_n(x - v) - \frac{1}{n}\bar{p}_n(x) \right)^2 \right] \\ &= nE \left[ \frac{1}{n^2}\delta_n^2(x - v) \right] - \frac{2}{n}\bar{p}_n^2(x) + \frac{1}{n}\bar{p}_n^2(x) \\ &= \int \frac{1}{n}\delta_n^2(x - v)p(v)dv - \frac{1}{n}\bar{p}_n^2(x) \\ &\leq \int \frac{1}{n} \left( \frac{1}{h_n^d}\phi \left( \frac{x - v}{h_n} \right) \right) \delta_n(x - v)p(v)dv \\ &\leq \frac{\sup(\phi(\cdot))}{nh_n^d} \int \delta_n(x - v)p(v)dv \\ &= \frac{\sup(\phi(\cdot))\bar{p}_n(x)}{nh_n^d} = C\end{aligned}$$

- If  $\sup_u \phi(u) < \infty$ , and  $\lim_{n \rightarrow \infty} V_n = \infty$ , then  $C \rightarrow 0$