CHAPTER 8:

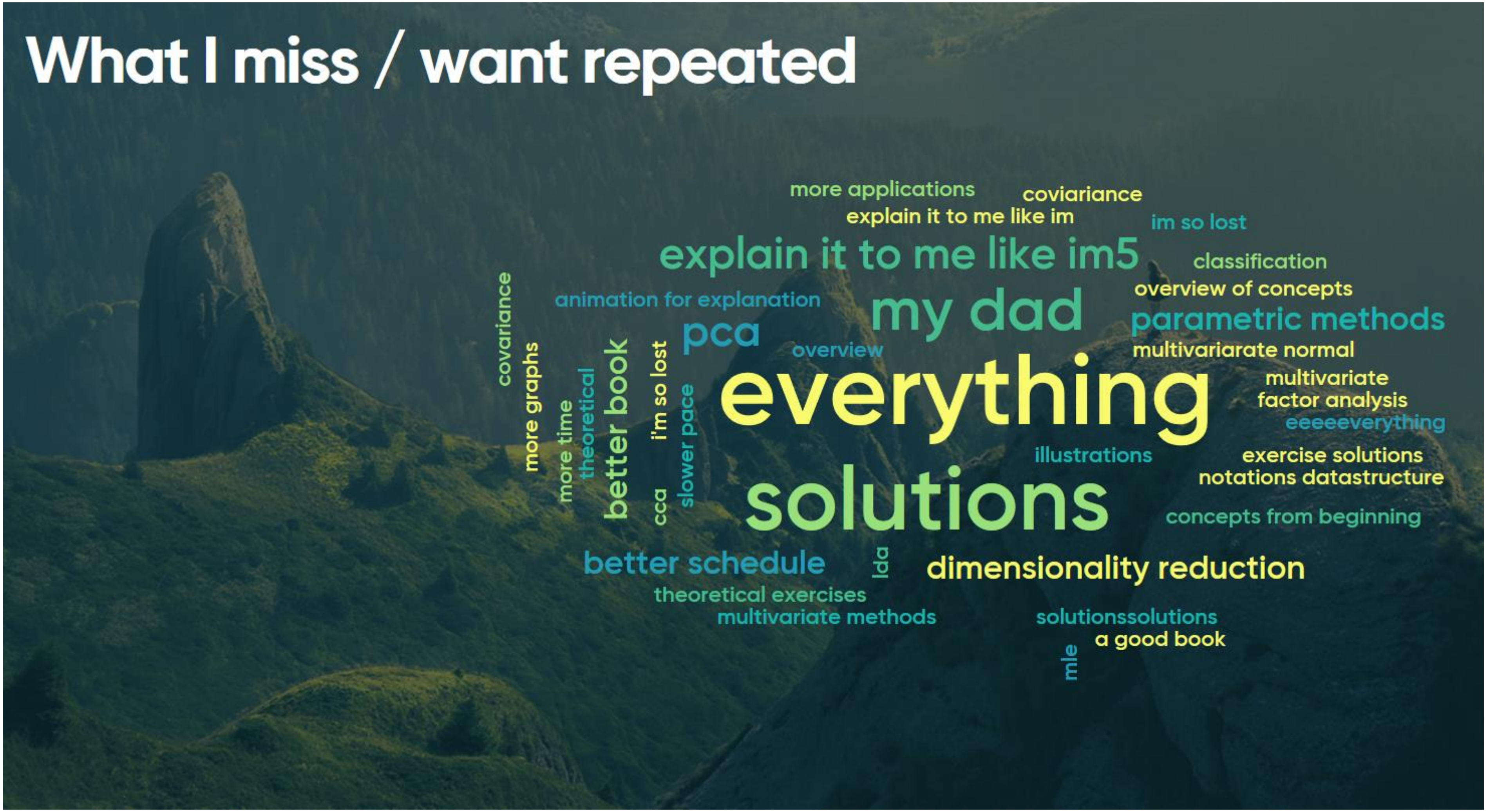
NONPARAMETRIC METHODS

Stella Grasshof

Overview

1. Repetition
2. Intro
3. Probabiliy Density Function (pdf)
4. Properties and Usages
5. Nonparametric Estimation
   a) … of pdf
   b) … of class-conditional pdf
   c) … for regression
6. Summary

Repetition?

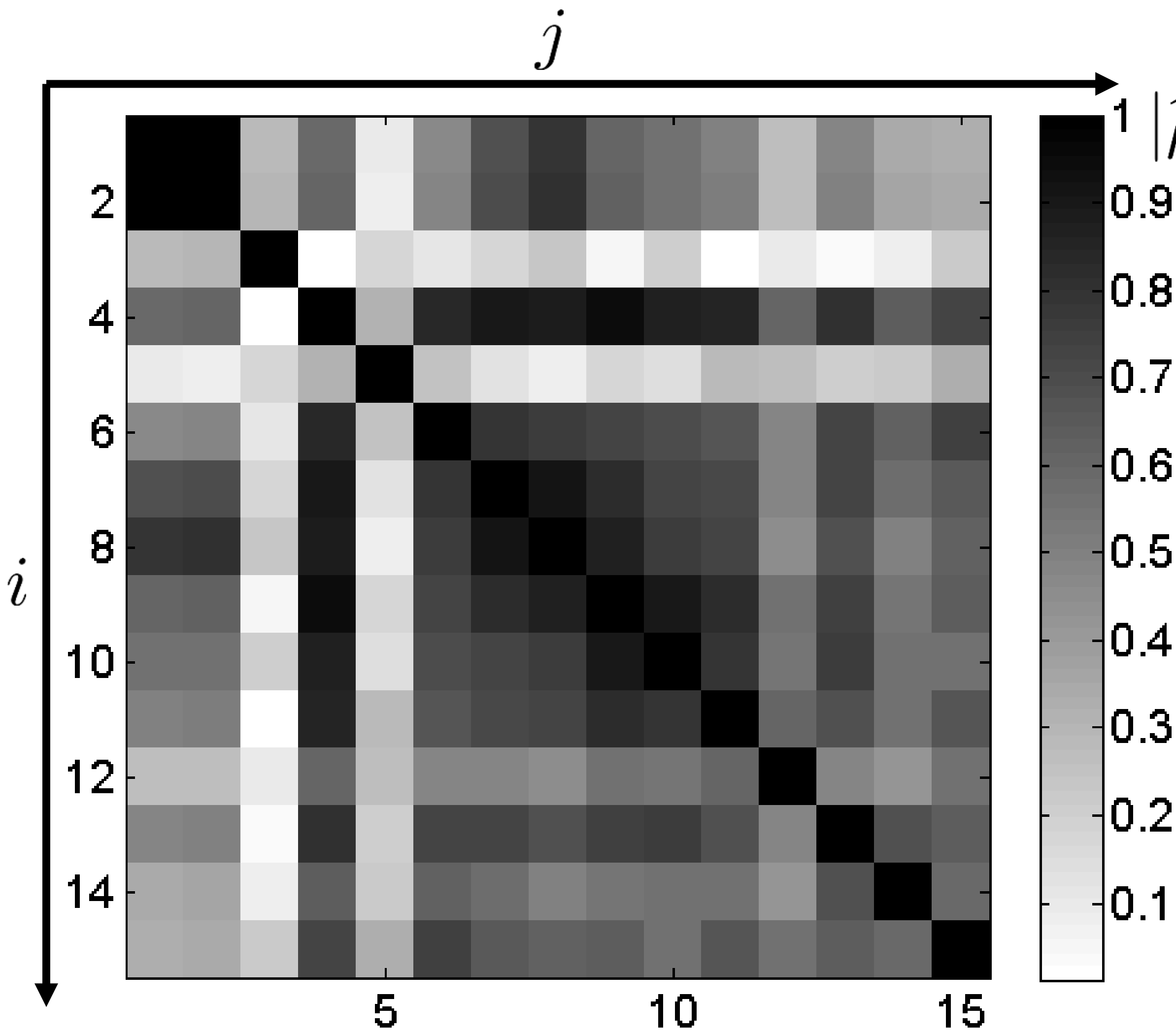


Book: *Pattern Recognition And Machine Learning*
today: Covariance, Correlation, PCA

Revisit: Correlation = scaled Cov

Example: absolute values of correlation matrix based on: $|\widehat{\boldsymbol{\Sigma}}|$

$\widehat{\boldsymbol{\Sigma}} = \widehat{\boldsymbol{\Sigma}}^{\mathrm{T}}$



Correlation matrix consists of entries $\boldsymbol{\Sigma}$

$\rho_{ij} = \mathrm{Corr}(\boldsymbol{x}_i, \boldsymbol{x}_j)$

pairwise correlation between features

$\boldsymbol{x}_i,\ \boldsymbol{x}_j,$

$i, j = 1, \ldots, D$

=> the darker the pixel at (i,j), the larger $|\rho_{ij}|$

Principal Components Analysis (PCA)

$$\mathbf{X} \in \mathbb{R}^{N \times D}$$

$$\mathrm{E}[\boldsymbol{x}] = \boldsymbol{\mu} = [\mu_1, \ldots, \mu_D]^{\mathrm{T}} \in \mathbb{R}^D$$

1. Subtract mean $\boldsymbol{X} - \boldsymbol{M}, \quad \boldsymbol{M} = (\widehat{\boldsymbol{\mu}}, \ldots, \widehat{\boldsymbol{\mu}})^{\mathrm{T}} \in \mathbb{R}^{N \times D}$
2. Compute covariance matrix $\widehat{\boldsymbol{\Sigma}} = \frac{1}{N}(\boldsymbol{X} - \boldsymbol{M})^{\mathrm{T}}(\boldsymbol{X} - \boldsymbol{M})$
3. Compute eigenvectors of covariance matrix

$$\widehat{\boldsymbol{\Sigma}} \boldsymbol{w}_k = \lambda_k \boldsymbol{w}_k, \; k = 1, ..., D, \; \lambda_i \geq \lambda_j, \; i > j$$

$$\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_j = \begin{cases} 1 & , i = j \\ 0 & , i \neq j \end{cases}$$

$$[\boldsymbol{w}_1, \ldots, \boldsymbol{w}_D] = \boldsymbol{W}^{\mathrm{T}}$$

$\boldsymbol{w}_k$ are **principal components**

$$[\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K] = \boldsymbol{W}_K^{\mathrm{T}}$$

4. New variables $\boldsymbol{z}_n = \boldsymbol{W}^{\mathrm{T}}(\boldsymbol{x}_n - \widehat{\boldsymbol{\mu}})$
5. Reconstrution $\widehat{\boldsymbol{x}}_n = \boldsymbol{W}_K \boldsymbol{z}_n + \widehat{\boldsymbol{\mu}}$

Principal Components Analysis (PCA)

1. Consider 2D data shall be mapped to 1D

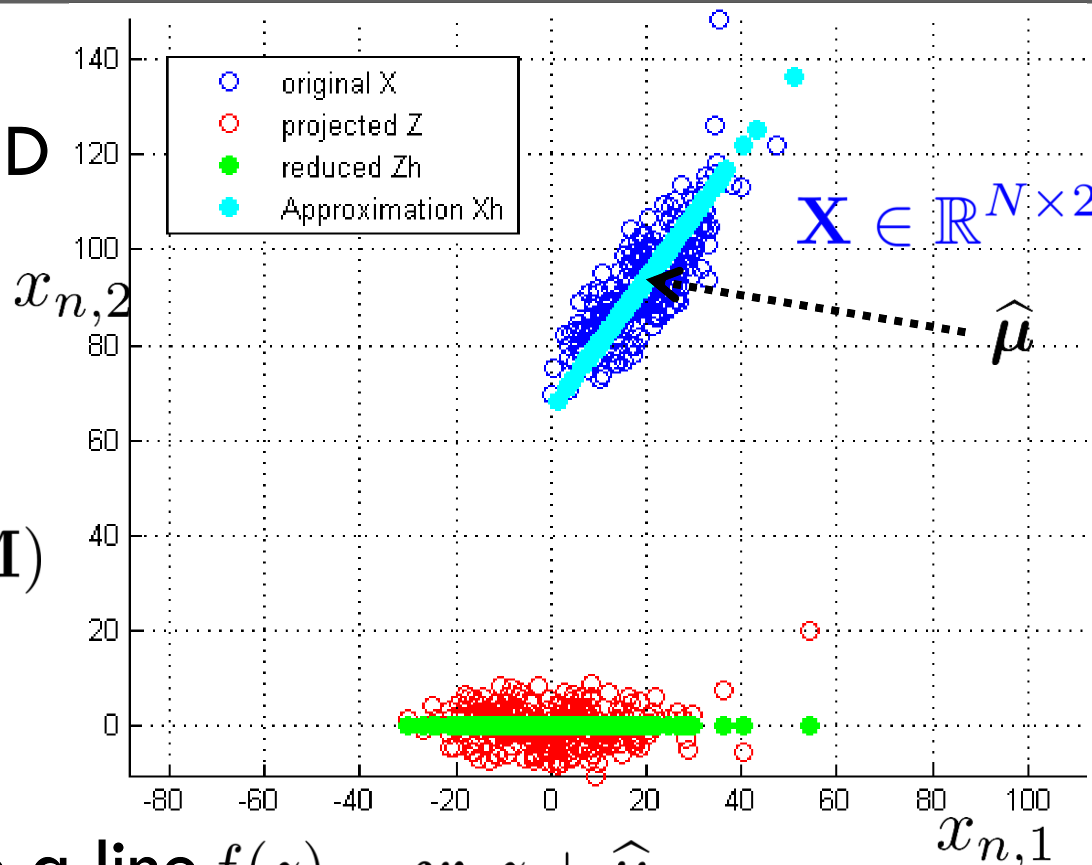
$$\boldsymbol{x}_n \in \mathbb{R}^2 \mapsto \boldsymbol{z}_n \in \mathbb{R}^1$$

2. Covariance of mean centerd data

$$\widehat{\Sigma} = \frac{1}{N}(\mathbf{X} - \mathbf{M})^T(\mathbf{X} - \mathbf{M})$$

3. Eigenvectors

$$\widehat{\boldsymbol{\Sigma}}\boldsymbol{w}_k = \lambda_k \boldsymbol{w}_k, \ k = 1, 2$$



4. Choose k=1 to define a line $f(z) = \boldsymbol{w}_1 z + \widehat{\boldsymbol{\mu}}$ leads to

- Reduction $z_n = \boldsymbol{w}_1^{\mathrm{T}}(\boldsymbol{x}_n - \widehat{\boldsymbol{\mu}}) \in \mathbb{R}^1$
- Reconstruction $\widehat{\boldsymbol{x}}_n = \boldsymbol{w}_1 z_n + \widehat{\boldsymbol{\mu}} \in \mathbb{R}^2$

END Repetition

Nonparametric Estimation

- **Parametric:**
  single global model
- **Semiparametric:**
  small number of local models
- **Nonparametric:**
  similar inputs have similar outputs
  - Keep the training data:
    "let the data speak for itself"
  - Given x: interpolate from the closest training samples
  - Aka lazy/memory-based/case-based/instance-based learning

Probability Density Function (pdf)

Suppose random variable X has pdf f

Properties of

- Non-negative $f(x) \geq 0, \quad \forall x \in \mathbb{R}$
- Normalized $\int_{-\infty}^{\infty} f(x)\ dx = 1$
- A pdf and a cumuliative distribution function relate

$$F(x) = P(X \leq x) = \int_{-\infty}^{x} f(t)\ dt \qquad \frac{d}{dx}F(x) = f(x)$$

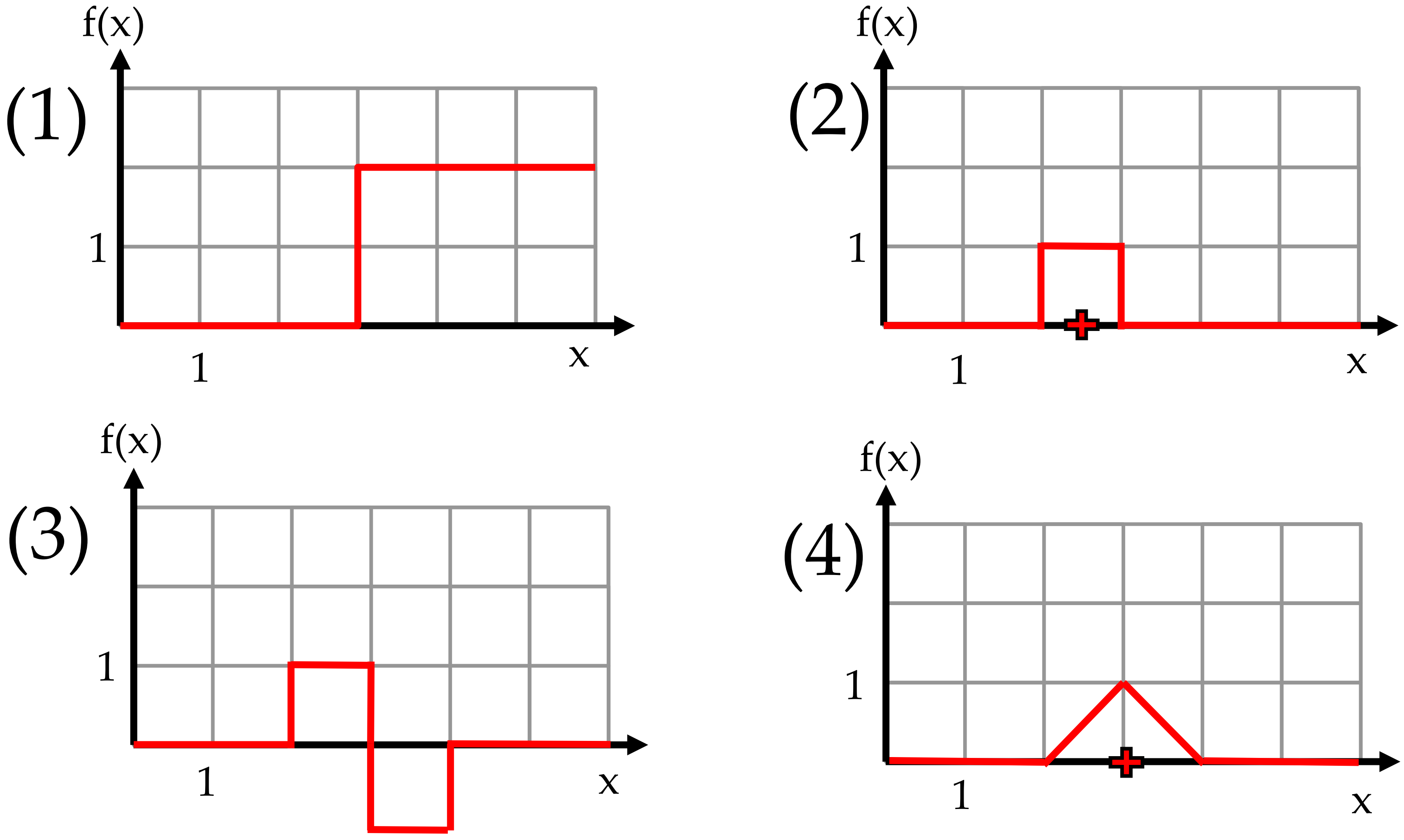
- Pdf relates area and probability

$$P(a \leq X \leq b) = \int_{a}^{b} f(x)\ dx$$

=> Usages?

Probability Density Function (pdf)

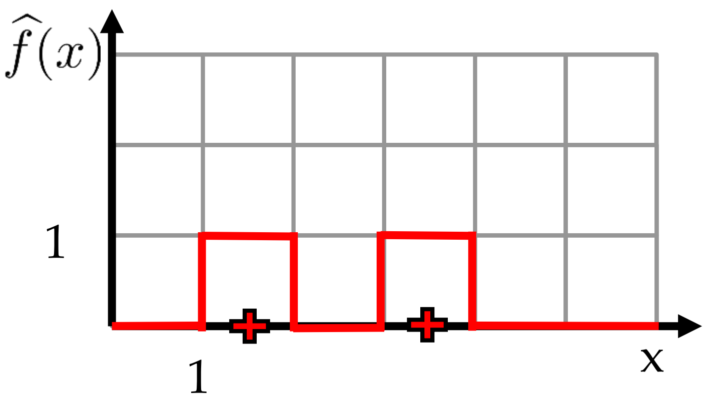
Which is a pdf?



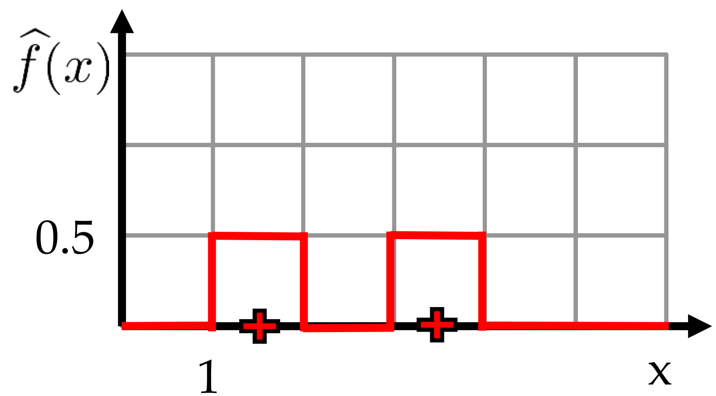
Probability Density Function (pdf)

Suppose random variable X has pdf f

it should be estimated from samples: ✚ $= \boldsymbol{x}_n, \ n = 1, \ldots, N$

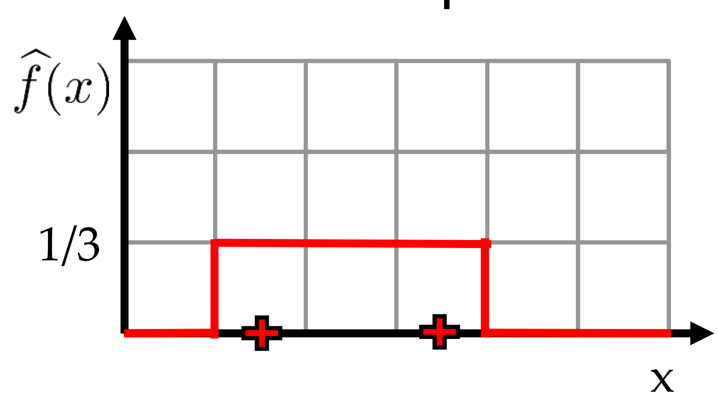


not a pdf



this is a pdf

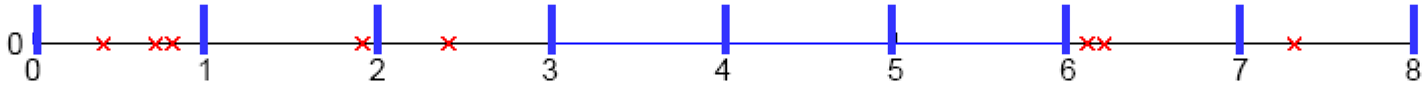
Different possible estimates for pdf from the samples



Density Estimation

- Given the training set $\boldsymbol{x}_n, \; n = 1, \ldots, N$ drawn iid from distribution described by pdf p
- => estimate pdf $p(x) \approx \widehat{p}(x)$
- Divide data into bins of size *h*

$N = 8$
$h = 1$

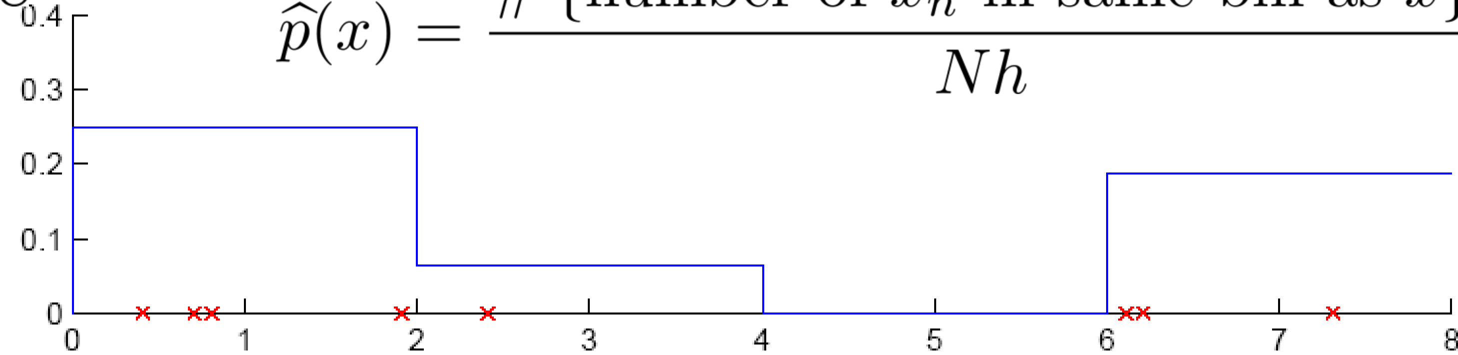


Histogram Estimator

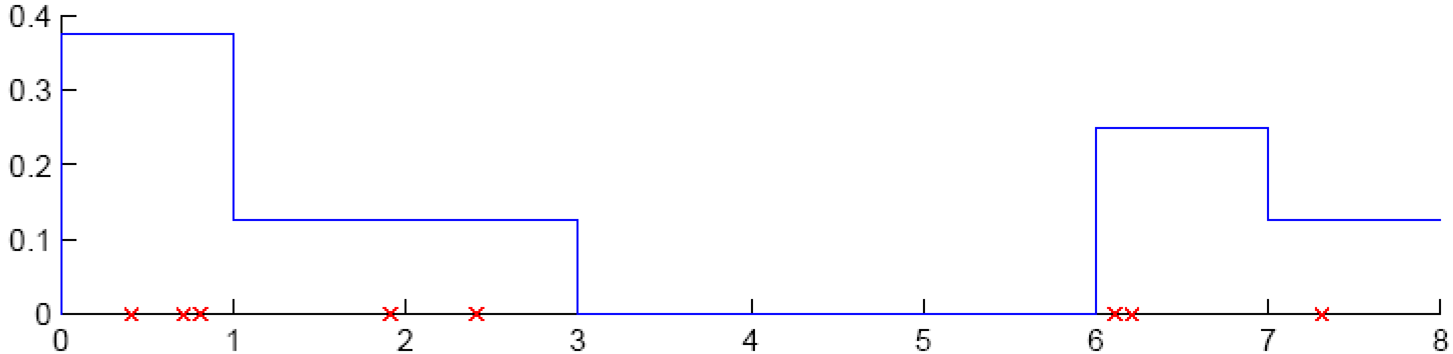
$N = 8$

$$\widehat{p}(x) = \frac{\#\,\{\text{number of } x_n \text{ in same bin as } x\}}{Nh}$$

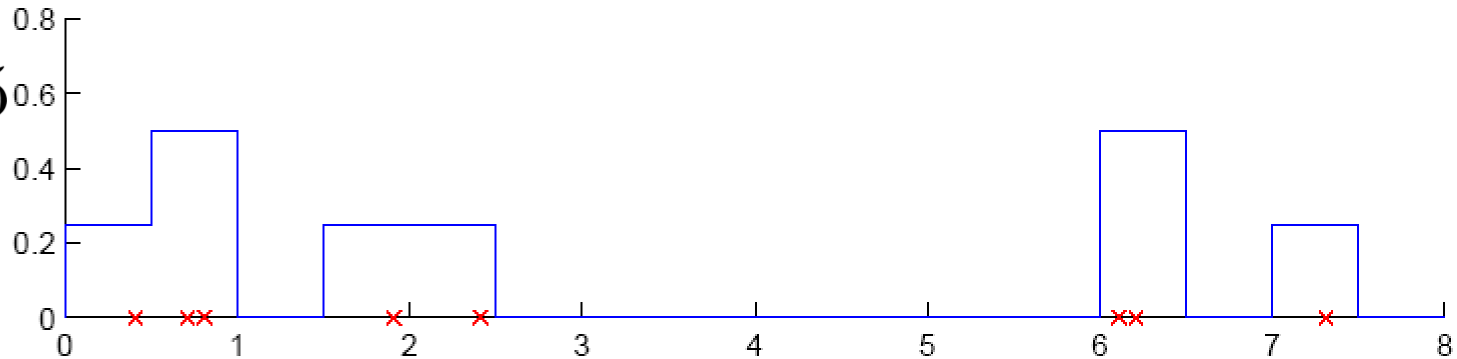
$h = 2$



$h = 1$



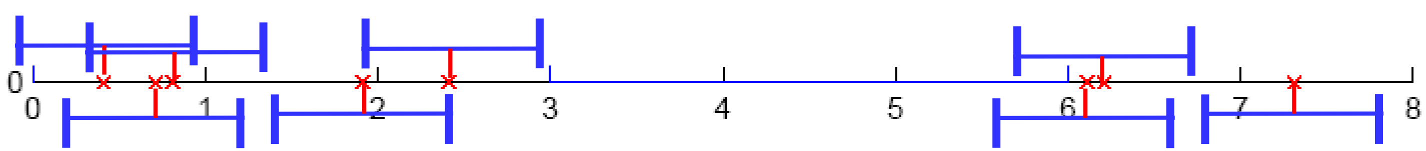
$h = 0.5$



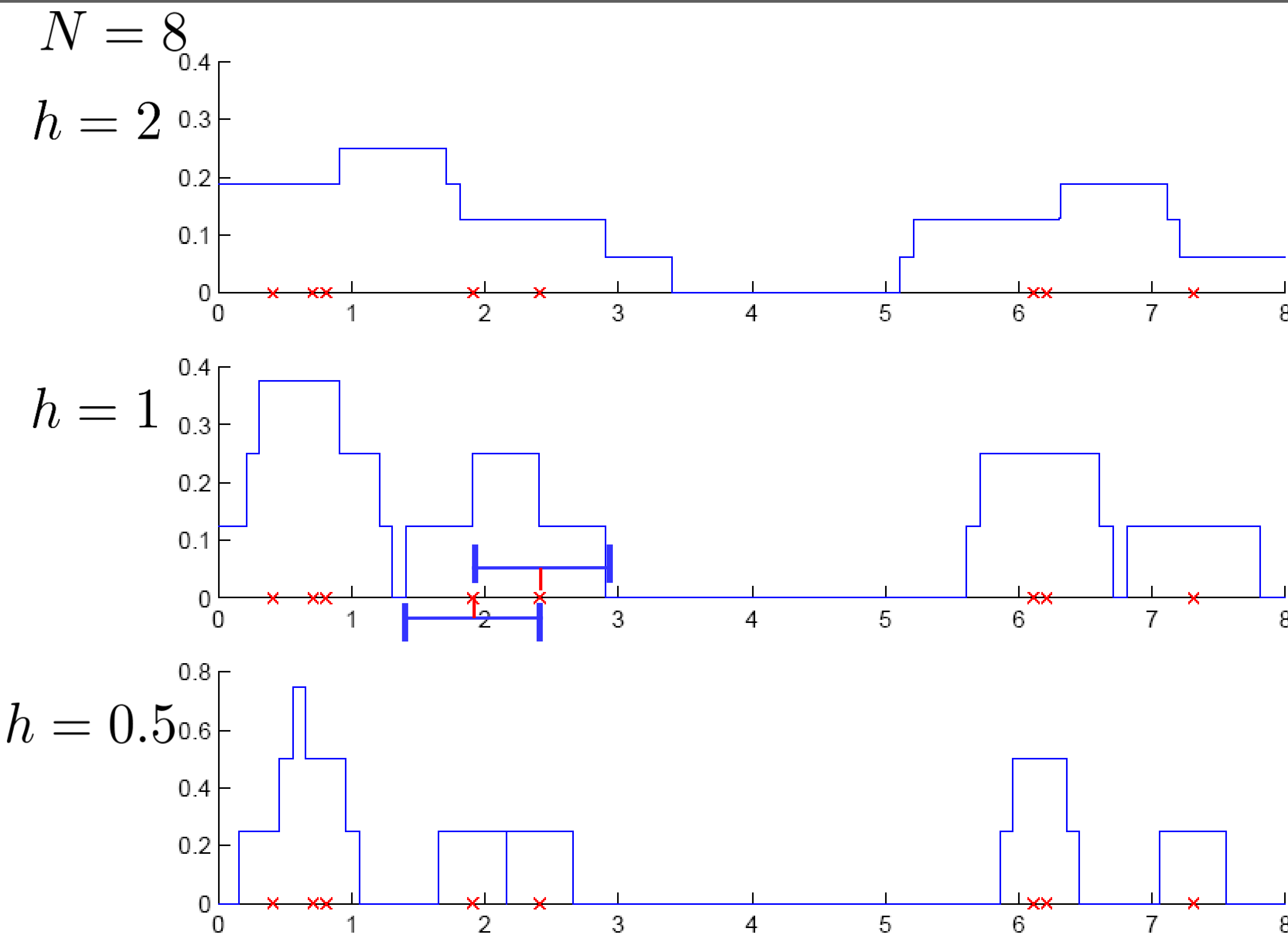
Density Estimation

- Given the training set $\boldsymbol{x}_n, \; n = 1, \ldots, N$ drawn iid from distribution described by pdf p
- => estimate pdf $p(x) \approx \widehat{p}(x)$
- Divide data into bins of size *h*
- *Center at data points*

$N = 8$
$h = 1$



Naive Estimator



Density Estimation

- Given the training set $\boldsymbol{x}_n, \; n = 1, \ldots, N$ drawn iid from *p*(*x*)
- Divide data into bins of size *h*
- Histogram:

$$\widehat{p}(x) = \frac{\#\left\{\text{number of } x_n \text{ in same bin as } x\right\}}{Nh}$$

- Naive estimator:

$$\widehat{p}(x) = \frac{\#\left\{x - h/2 < x_n \leq x + h/2\right\}}{Nh}$$

or

$$\widehat{p}(x) = \frac{1}{Nh} \sum_{n=1}^{N} w\left(\frac{x - x_n}{h}\right), \quad w(u) = \begin{cases} 1 & , \; |u| < 1/2 \\ 0 & , \; \text{else} \end{cases}$$

Drawback: Discontinuities

Kernel Estimator

- Replace:

$$w(u) = \begin{cases} 1 & , \ |u| < 1/2 \\ 0 & , \ \text{else} \end{cases}$$

- by another kernel function, e.g. Gaussian kernel:

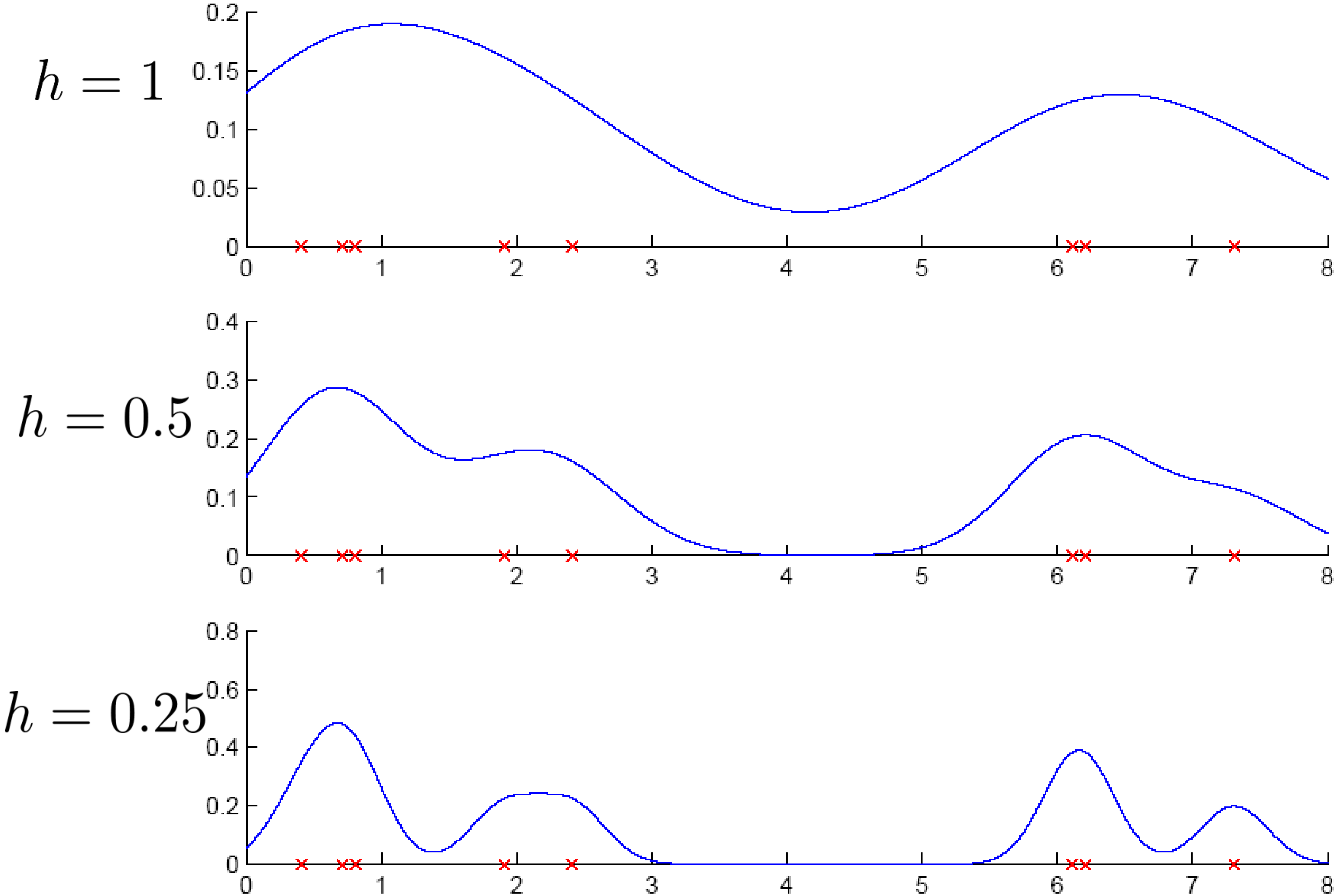
$$\varphi(x) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{x^2}{2}\right] \qquad \text{this is: } \mathcal{N}(0,1)$$

- Gives kernel estimator (Parzen windows)

$$\widehat{p}(x) = \frac{1}{Nh} \sum_{n=1}^{N} \varphi\left(\frac{x - x_n}{h}\right)$$

… is continuous, differentiable

Kernel Estimator

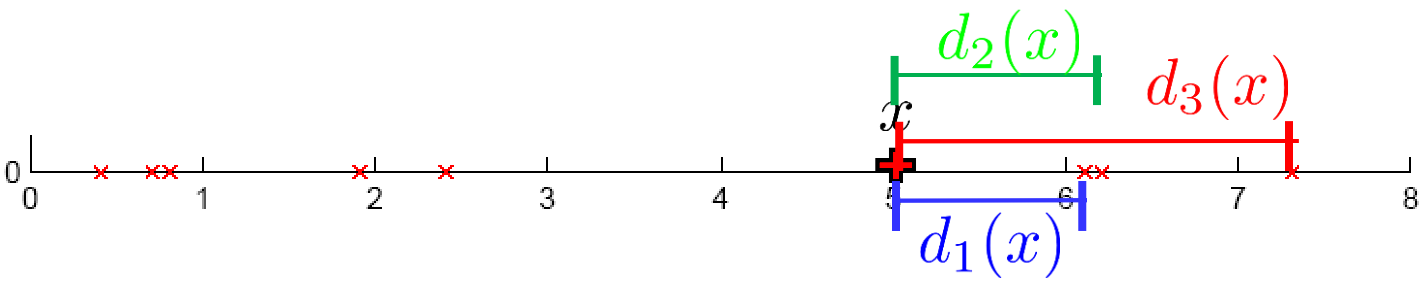


k-Nearest Neighbor Estimator

Instead of fixing bin width *h*
and counting the number of instances
fix the instances (neighbors) *k* and check bin width

=> $d_k(x)$ distance to *k*th closest instance to *x*

aim: small distance = high pdf



$$\widehat{p}(x) = \frac{k}{2N d_k(x)}$$

Drawback: Discontinuities and is NOT pdf
improvement by kernel function

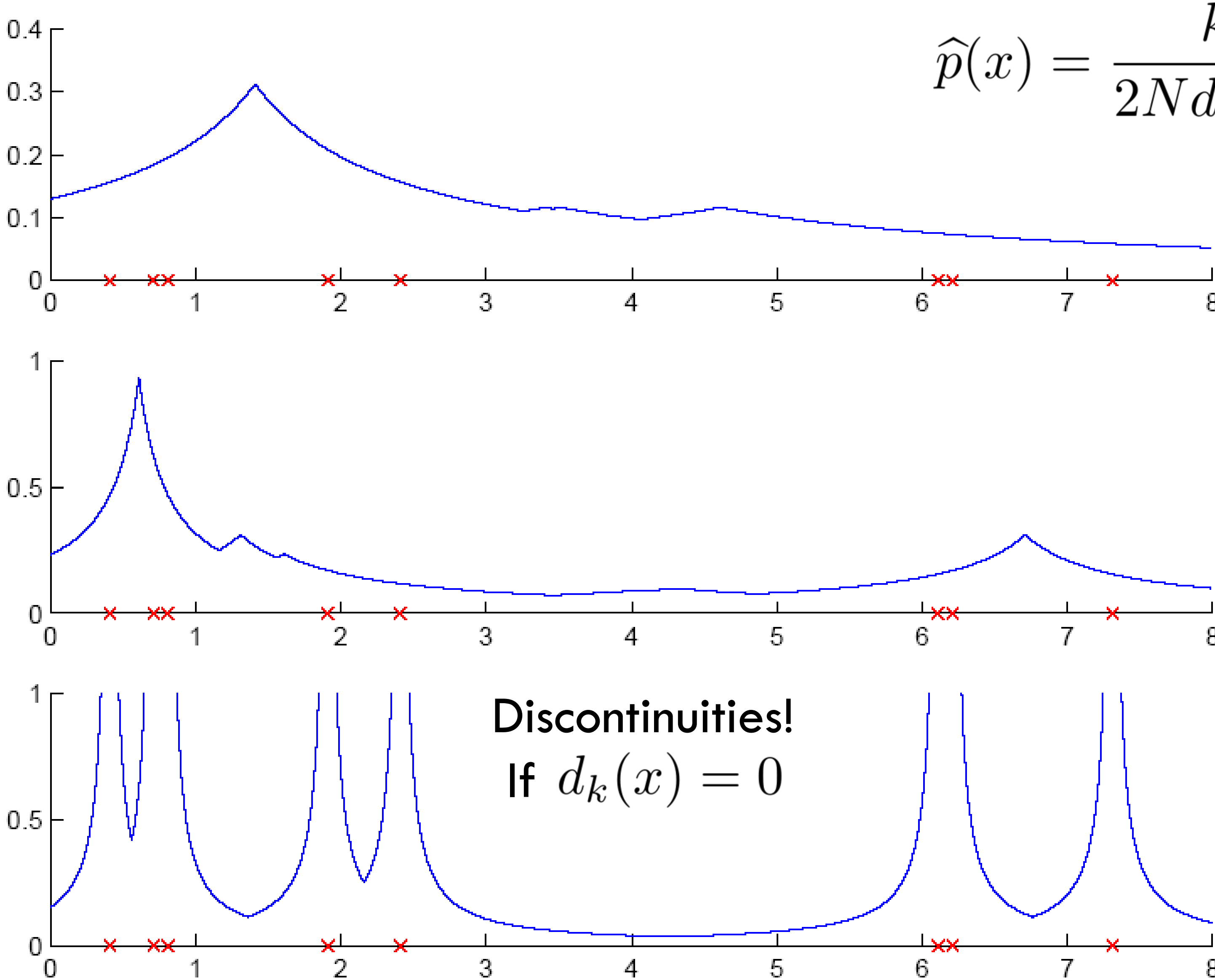
k-Nearest Neighbor Estimator

$$\widehat{p}(x) = \frac{k}{2N d_k(x)}$$

$k = 5$

$k = 3$

$k = 1$



Discontinuities!

If $d_k(x) = 0$

Multivariate Data

$\boldsymbol{x}, \boldsymbol{x}_n \in \mathbb{R}^D$

Kernel density estimator

$$\widehat{p}(\boldsymbol{x}) = \frac{1}{Nh^D} \sum_{n=1}^{N} \varphi\left(\frac{\boldsymbol{x} - \boldsymbol{x}_n}{h}\right)$$

Multivariate Gaussian kernel

- spheric $\mathcal{N}\left(\mathbf{0}, \boldsymbol{I}\right)$

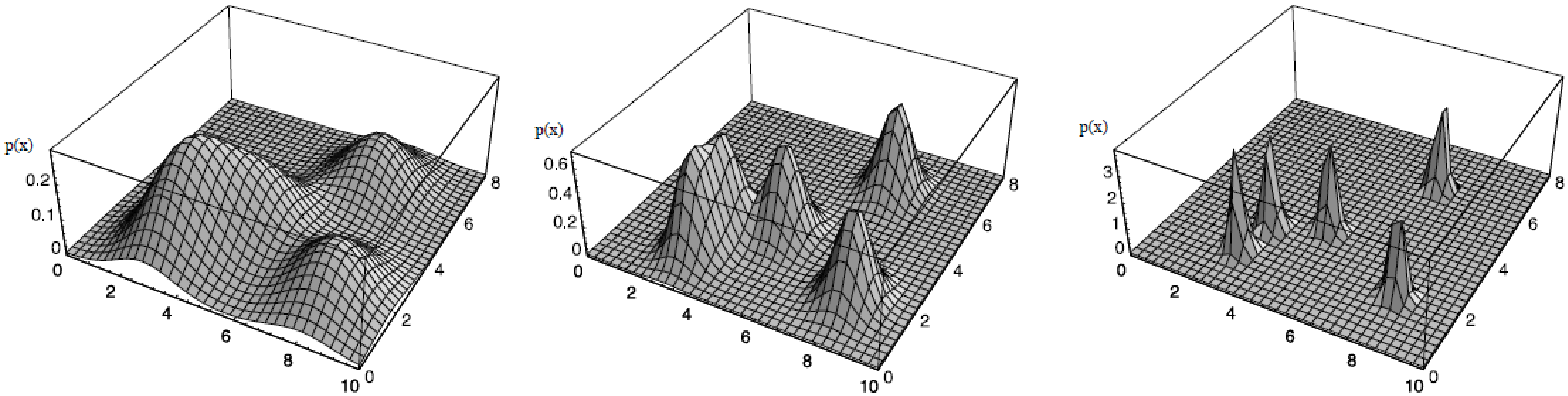
$$\varphi\left(\boldsymbol{x}\right) = \frac{1}{(2\pi)^{D/2}} \exp\left[-\frac{\left\|\boldsymbol{x}\right\|^2}{2}\right]$$

- ellipsoid $\mathcal{N}\left(\mathbf{0}, \boldsymbol{S}\right)$

$$\varphi\left(\boldsymbol{x}\right) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{S}|^{1/2}} \exp\left[-\frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{S}^{-1}\boldsymbol{x}\right]$$

Kernel Estimator

Estimation from 5 points with varying width



“Pattern Classification“, Duda et al., 2001

Nonparametric Classification

- Discriminant for class *k*: $\widehat{g}_i(\boldsymbol{x}) = \widehat{P}(C_i)\widehat{p}(\boldsymbol{x}|C_i)$
- Class probability as before: $\widehat{P}(C_i) = \dfrac{N_i}{N}$
- Estimate conditional pdf $p(\boldsymbol{x}|C_i) \approx \widehat{p}(\boldsymbol{x}|C_i)$
  - Kernel estimator (same as before, but select cases)

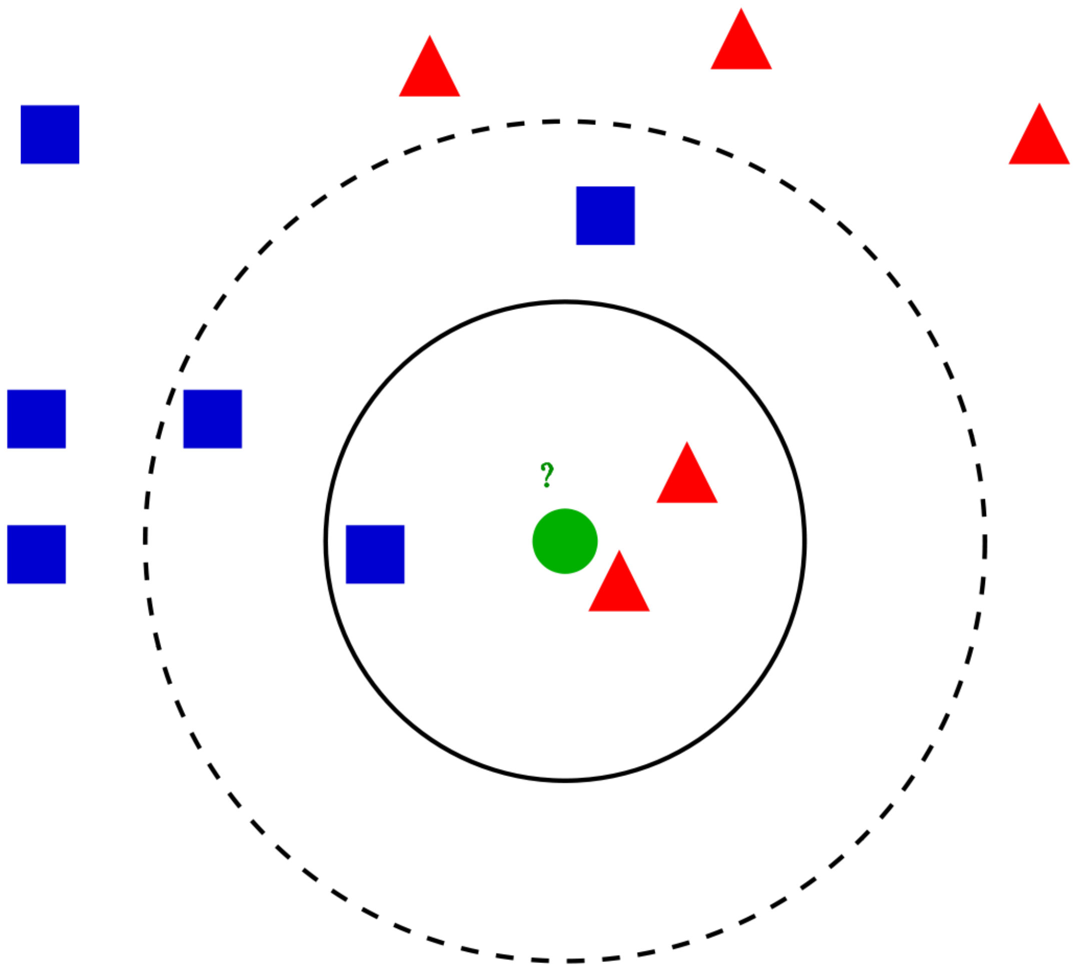
$$\widehat{p}(\boldsymbol{x}|C_i) = \frac{1}{N_i h^D} \sum_{n=1}^{N} 1(\boldsymbol{x}_n \in C_i)\varphi\left(\frac{\boldsymbol{x} - \boldsymbol{x}_n}{h}\right)$$

  - *k*-NN estimator: get *k* nearest neighbors of $\boldsymbol{x}$
    - k_i of k neighbors belong to class C_i
    - $V^k(\boldsymbol{x})$ volume of D-dimensional hypersphere centered at $\boldsymbol{x}$

then: $\widehat{p}(\boldsymbol{x}|C_i) = \dfrac{k_i}{N_i V^k(\boldsymbol{x})}$

Decision is majority vote: $\widehat{P}(C_i|\boldsymbol{x}) = \dfrac{k_i}{k}$

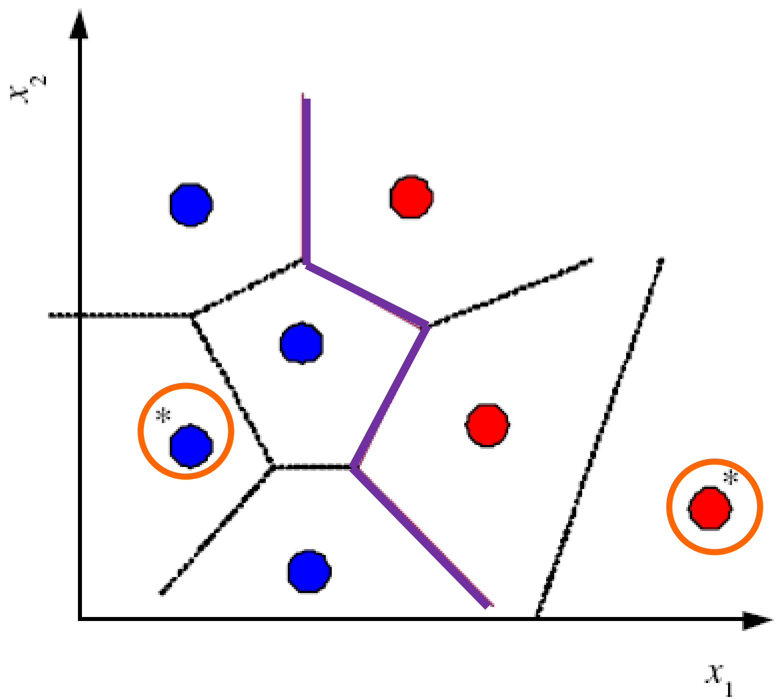
k-Nearest Neighbor Classifier



https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

Condensed Nearest Neighbor

- Find a subset Z of X that is small and accurate in classifying X (Hart, 1968)
- The points do not contribute to the decision boundary => delete them from X receive smaller set Z



Condensed Nearest Neighbor

Incremental algorithm: Add instance if needed

$\mathcal{Z} \leftarrow \emptyset$

Repeat

For all $\boldsymbol{x} \in \mathcal{X}$ (in random order)

Find $\boldsymbol{x}' \in \mathcal{Z}$ s.t. $\|\boldsymbol{x} - \boldsymbol{x}'\| = \min_{\boldsymbol{x}^j \in \mathcal{Z}} \|\boldsymbol{x} - \boldsymbol{x}^j\|$

If class($\boldsymbol{x}$)$\neq$class($\boldsymbol{x}'$) add $\boldsymbol{x}$ to $\mathcal{Z}$

Until $\mathcal{Z}$ does not change

Distance-based Classification

- Find a distance function $\mathcal{D}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ such that:
  - if $\boldsymbol{x}_i, \boldsymbol{x}_j$ belong to the same class: $\mathcal{D}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ small
  - if $\boldsymbol{x}_i, \boldsymbol{x}_j$ belong to different classes: $\mathcal{D}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ large
- Assume a parametric model and learn its parameters using data, e.g.,

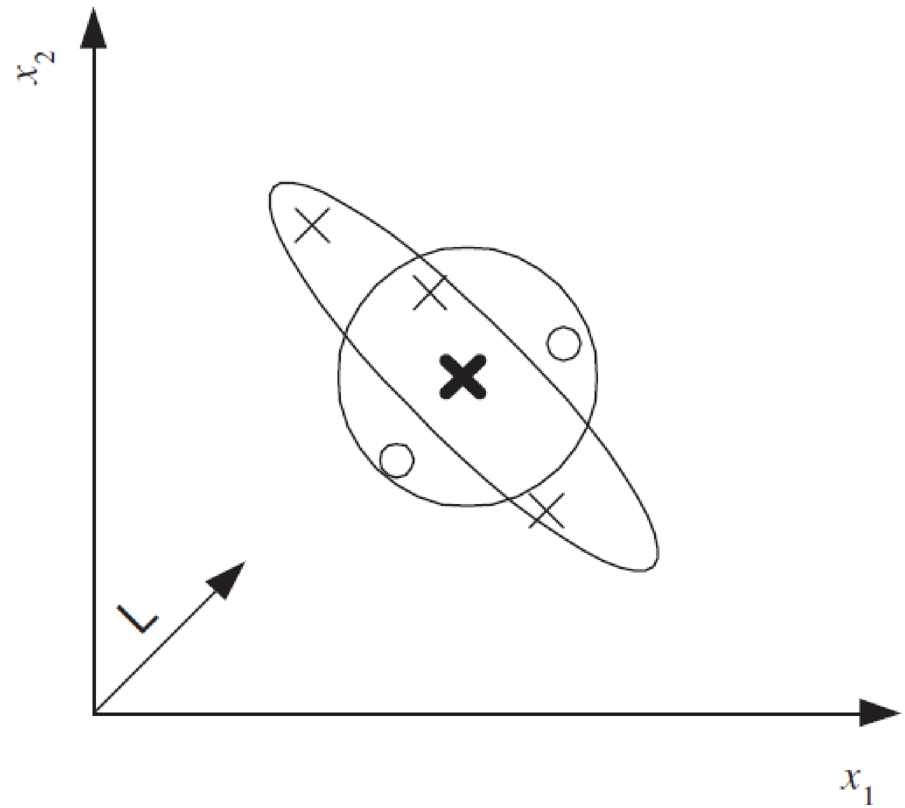
$$\mathcal{D}(\boldsymbol{x}_i, \boldsymbol{x}_j | \boldsymbol{M}) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j)$$

Learning a Distance Function

- Distances in high dimensions, can be represented as euclidean distance in lower dimensions
- Consider dimensionality reduction: $\boldsymbol{z}_i = \boldsymbol{L}\boldsymbol{x}_i$
- **M**=**L**^{T}**L** is *D*x*D* and **L** is *K*x*D*

$$\begin{aligned}\mathcal{D}(\boldsymbol{x}_i, \boldsymbol{x}_j | M) &= (\boldsymbol{x}_i - \boldsymbol{x}_j)^{\mathrm{T}} \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j) = \ldots \\ &= ||\boldsymbol{z}_i - \boldsymbol{z}_j||_2^2\end{aligned}$$

- Similarity-based representation using similarity scores



Euclidean distance (circle) is not suitable,

Mahalanobis distance using an **M** (ellipse) is suitable.

After the data is projected along **L**, Euclidean distance can be used.

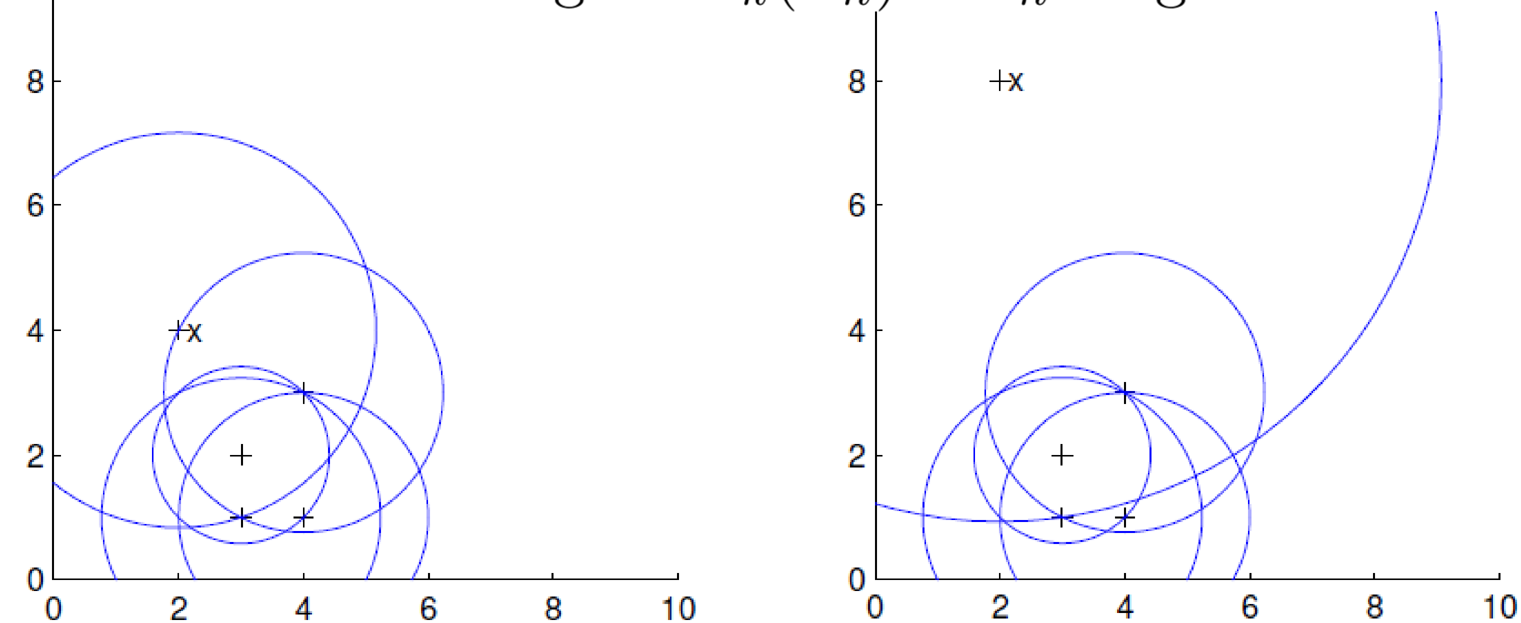
Outlier Detection

- Find outlier/novelty points
- Not a two-class problem!
  - Outliers vary, are seldom and unlabeled
- Instead: find instances with low probability
- In nonparametric case:
  Find instances far away from other instances

Local Outlier Factor (LOF)

- Circle around each point, containing k=3 next neighbors
- Is the distance to the 3rd nearest neigbor similar to other points in that area?
  - Yes: LOF near 1
  - No: LOF bigger than 1 => Outlier

$$LOF(\boldsymbol{x}) = \frac{d_k(\boldsymbol{x})}{\text{average of } d_k(\boldsymbol{x}_n) : \ \boldsymbol{x}_n \text{ neighbors of } \boldsymbol{x}}$$



Nonparametric Regression

- Regression:
  - Given training data: $(\boldsymbol{x}_n, y_n)$
  - Estimate function such that $y_n = g(\boldsymbol{x}_n) + \epsilon$

$$y_n \approx g(\boldsymbol{x}_n)$$

- Nonparametric regression = smoothing models, e.g.

$$g(\boldsymbol{x}) := \text{average of } y_n \text{ from } \boldsymbol{x}_n \text{ in the bin of } \boldsymbol{x}$$

- Regressogram:
  - Divide data into bins of width h
  - Define value for each bin as:
    - average of points inside of bin training data
    - Line through points
    - …

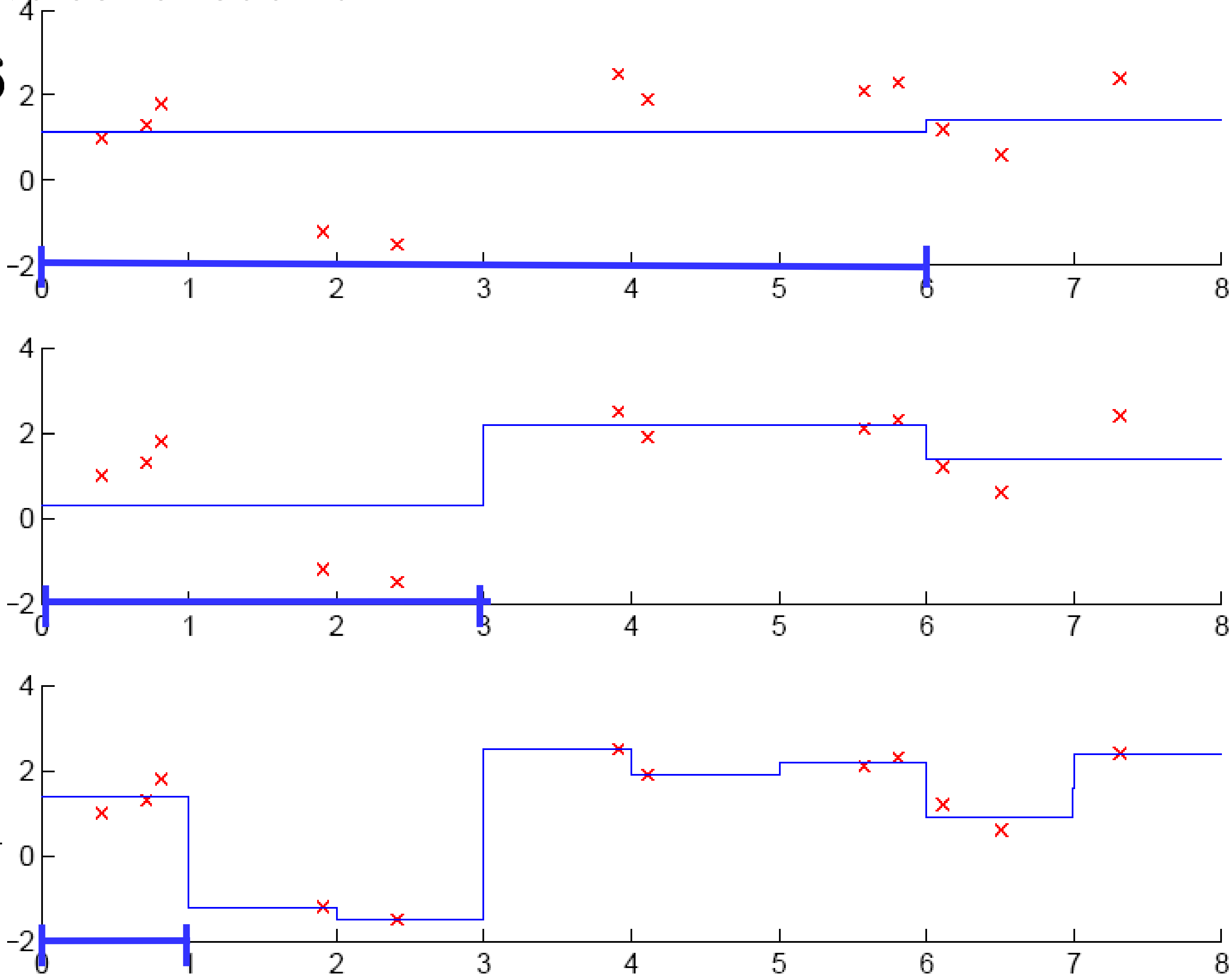
Regressogram smoother

Mean value for each bin

$h = 6$

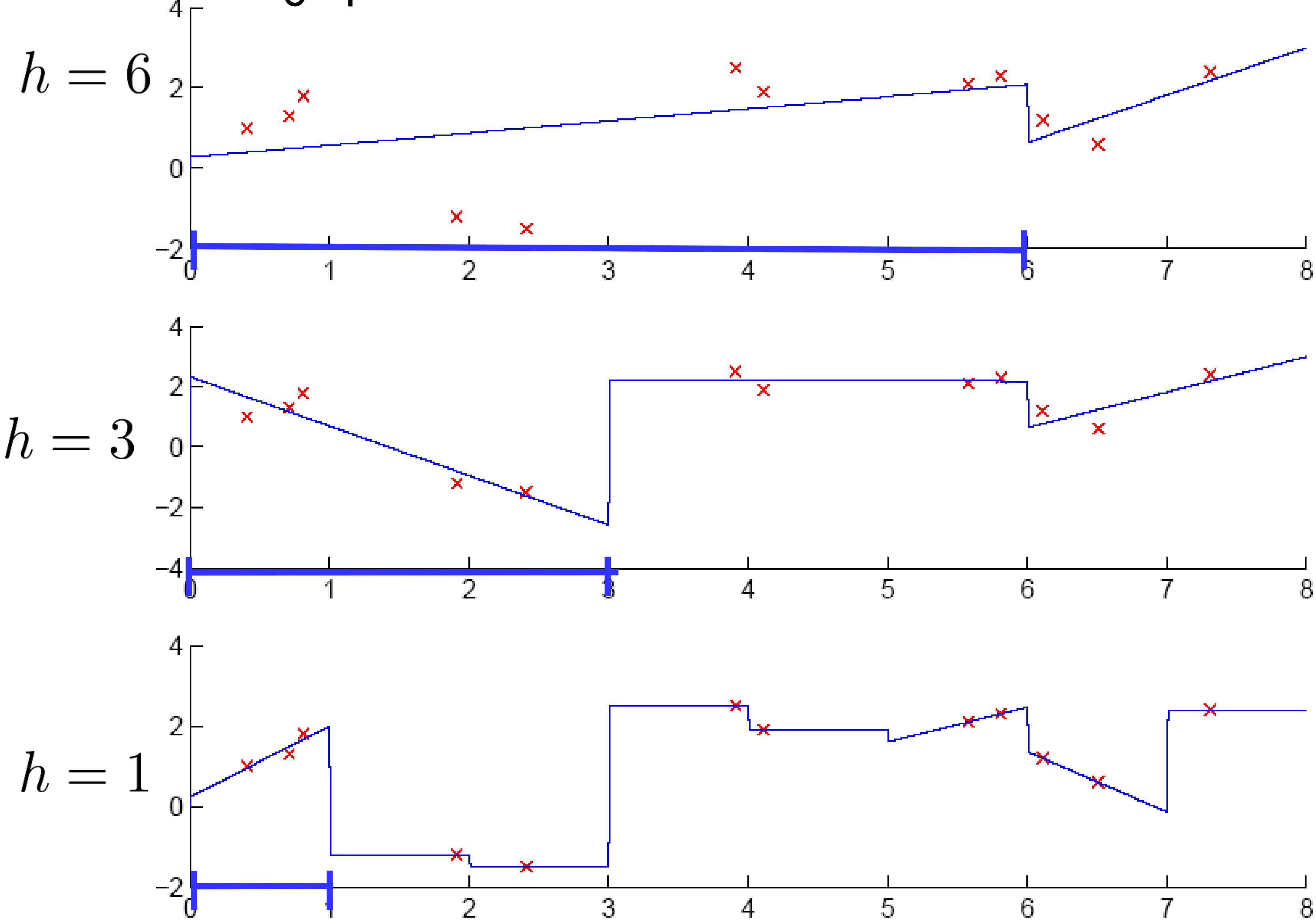
$h = 3$

$h = 1$



Nonparametric Regression

Fit line through points in each bin



Running Mean/Kernel Smoother

$$\widehat{g}(x) = \frac{\sum_{n=1}^{N} \varphi\left(\frac{x-x_n}{h}\right) y_n}{\sum_{n=1}^{N} \varphi\left(\frac{x-x_n}{h}\right)}$$

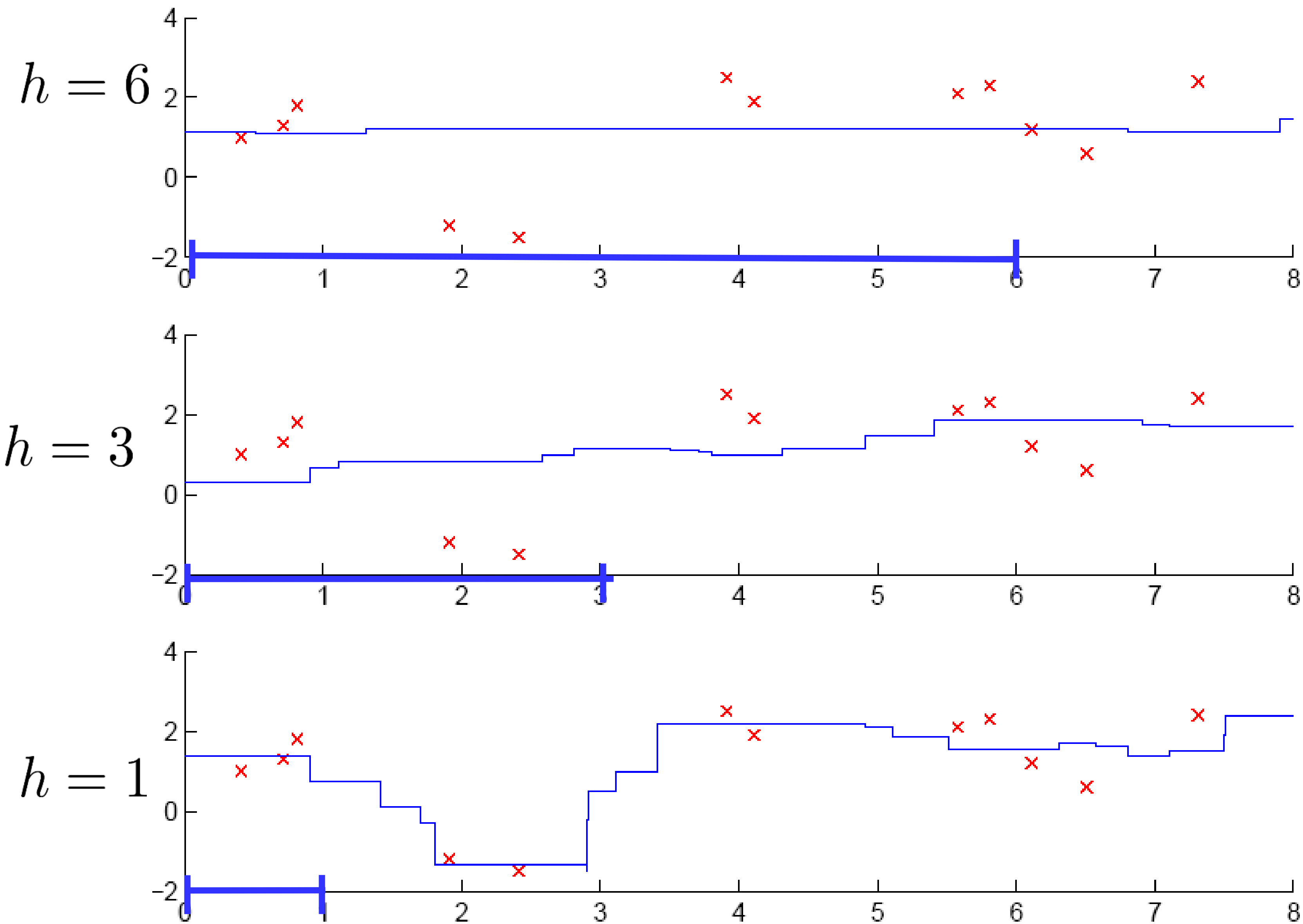
- Running mean smoother

$$\varphi(x) = \begin{cases} 1 & , \text{ if } |x| < 1 \\ 0 & , \text{ else} \end{cases}$$

- Line smoother by local regression
  locally weighted line fitting
- Kernel smoother:

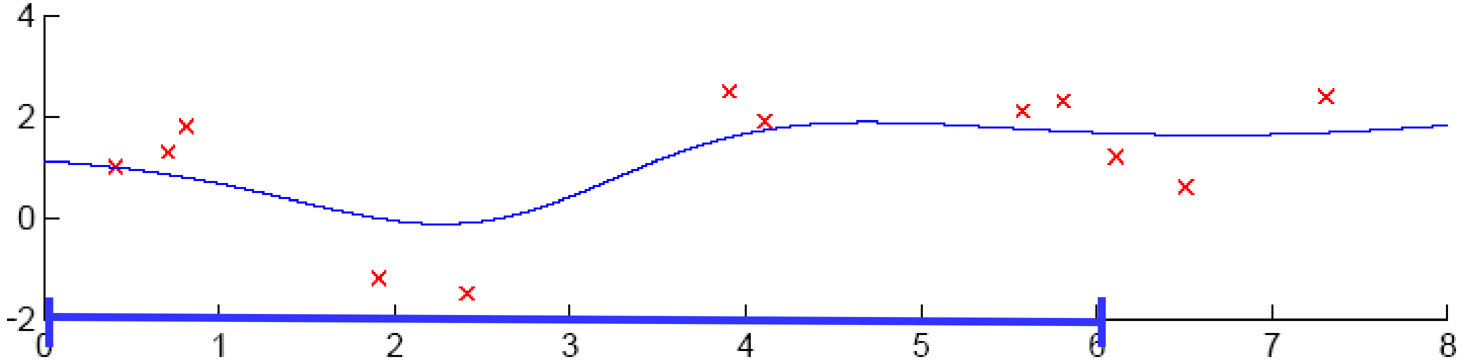
$$\varphi(x) = \mathcal{N}(0,1)$$

Running mean smoother

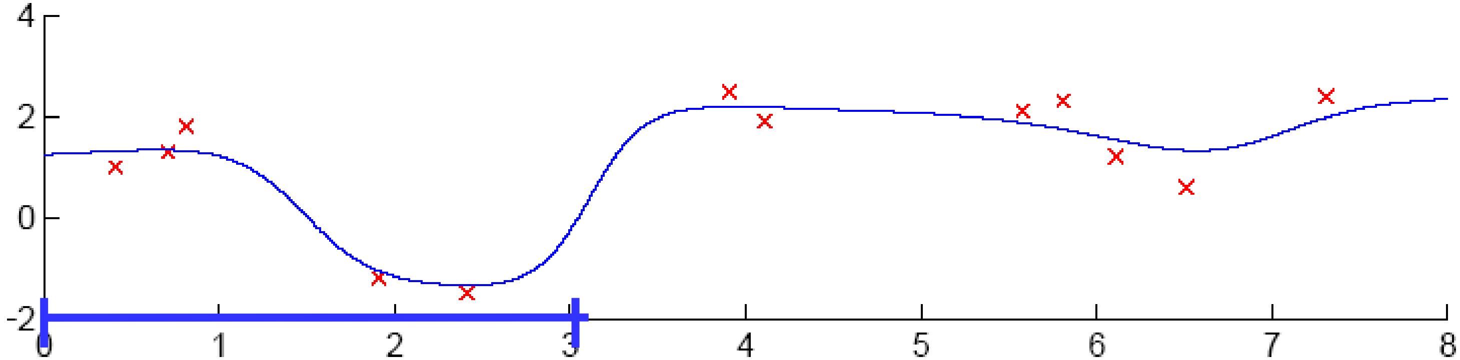


Kernel Smoother

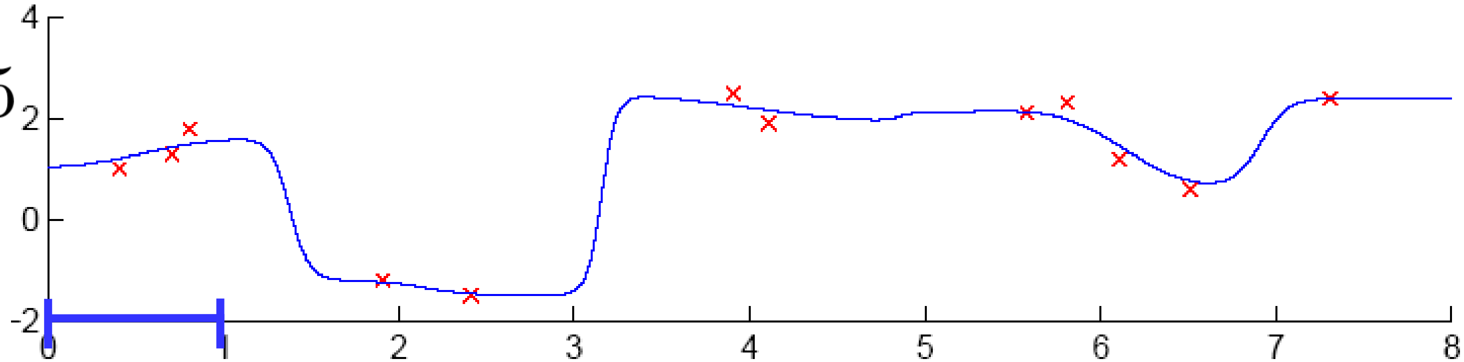
$h = 1$



$h = 0.5$



$h = 0.25$

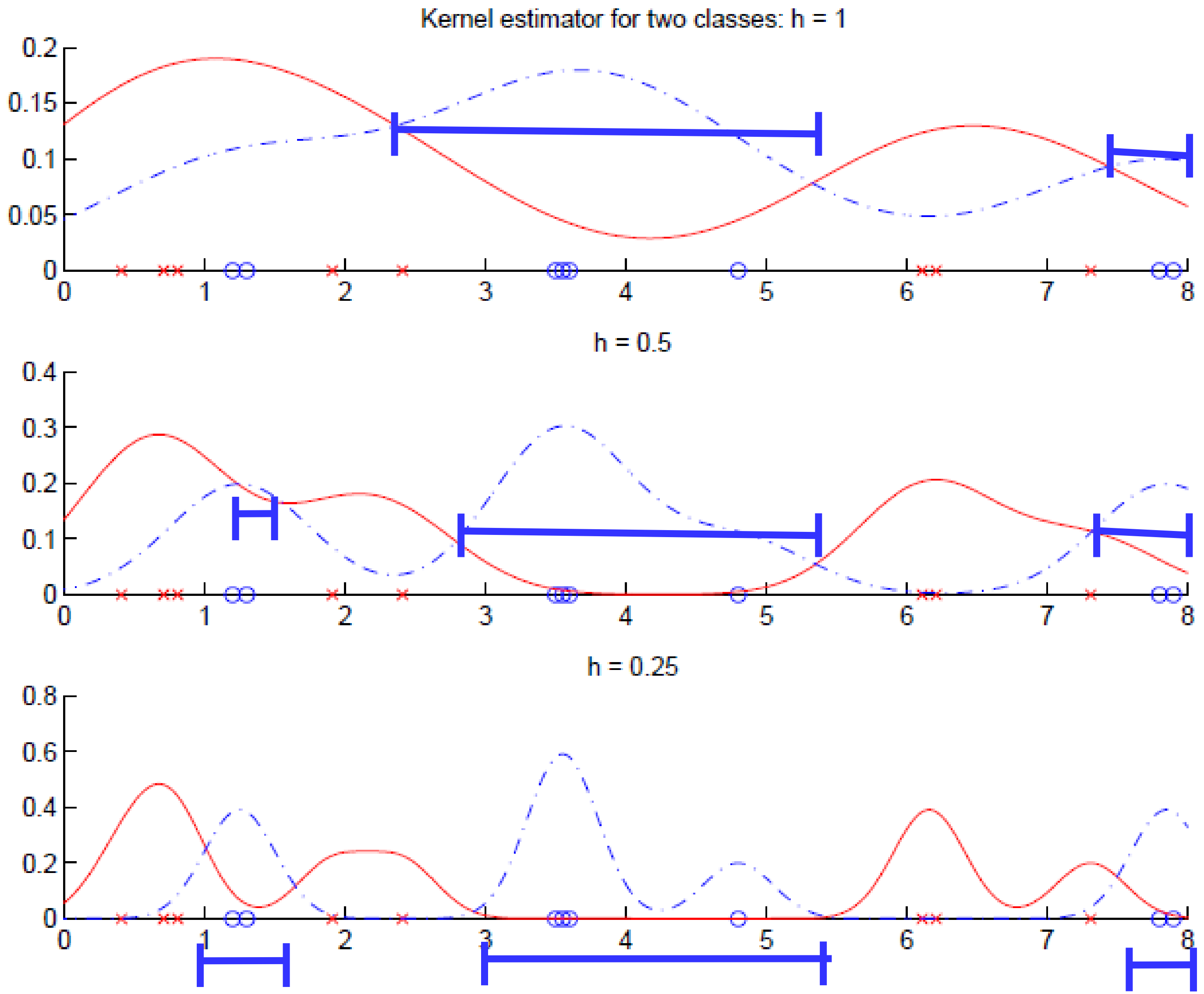


Gaussian Filtered Image

https://en.wikipedia.org/wiki/Gaussian_blur#/media/File:Cappadocia_Gaussian_Blur.svg

How to Choose *k* or *h* ?

- *k* or *h* is small:
  - single instances matter
    => bias is small, variance is large
    => undersmoothing: High complexity
- *k* or *h* increases:
  - average over more instances
    => variance decreases but bias increases
    => oversmoothing: Low complexity
- Cross-validation is used to finetune *k* or *h*



Kernel estimator for two classes: h = 1
0.2
0.15
0.1
0.05
0
0 1 2 3 4 5 6 7 8
h = 0.5
0.4
0.3
0.2
0.1
0
0 1 2 3 4 5 6 7 8
h = 0.25
0.8
0.6
0.4
0.2
0
0 1 2 3 4 5 6 7 8

Summary

Nonparametric

- No explicit model
- Use training data
  - “Smart“ subset of training data
- More robust to outliers
- Different possibilites for pdf, regression …

APPENDIX