Nonparametric Methods

Prof. Ziping Zhao

School of Information Science and Technology
ShanghaiTech University, Shanghai, China

CS182: Introduction to Machine Learning (Fall 2021)
http://cs182.sist.shanghaitech.edu.cn

Outline

Outline

Introduction

Nonparametric Density Estimation

Generalization to Multivariate Case

Nonparametric Classification

Nonparametric Regression

Parametric, Semiparametric, and Nonparametric Methods

- Parametric:
  - $p(\mathbf{x} \mid C_i)$ is represented by a single global parametric model.
  - Topic 3 (Parameter Estimation for Generative Models)
- Semiparametric:
  - $p(\mathbf{x} \mid C_i)$ is represented by a small number of local parametric models.
  - Topic 10 (Clustering and Mixture Models)
- Nonparametric:
  - $p(\mathbf{x} \mid C_i)$ cannot be represented by a single parametric model or a mixture model; the data speaks for itself.
  - Assumption: similar inputs have similar outputs, i.e., smooth functions (e.g., probability density functions, discriminant functions, regression functions).
  - Given a test instance, find a small number of nearest (or most similar) training instances and interpolate from them.
  - A.k.a. instance-based, memory-based, case-based or lazy learning algorithms.

Outline

Introduction

Nonparametric Density Estimation

Generalization to Multivariate Case

Nonparametric Classification

Nonparametric Regression

Nonparametric Density Estimation: Univariate Case

- Sample $\mathcal{X} = \{x^{(\ell)}\}_{\ell=1}^{N}$, drawn i.i.d. from some unknown probability density $p(x)$, with cumulative distribution function $F(x)$.
- Estimator $\hat{F}(x)$ for $F(x)$:

$$\hat{F}(x) = \frac{\#\{x^{(\ell)} \leq x\}}{N}$$

- Estimator $\hat{p}(x)$ for $p(x)$:

$$\hat{p}(x) = \frac{1}{h}\left[\frac{\#\{x^{(\ell)} \leq x + h\} - \#\{x^{(\ell)} \leq x\}}{N}\right]$$

where h is the length of the interval and instances $x^{(\ell)}$ that fall in this interval are assumed to be "close enough".

Histogram Estimator

- The input space is divided into equal-sized intervals called bins:

$$\left[x_0 + mh, x_0 + (m+1)h\right)$$

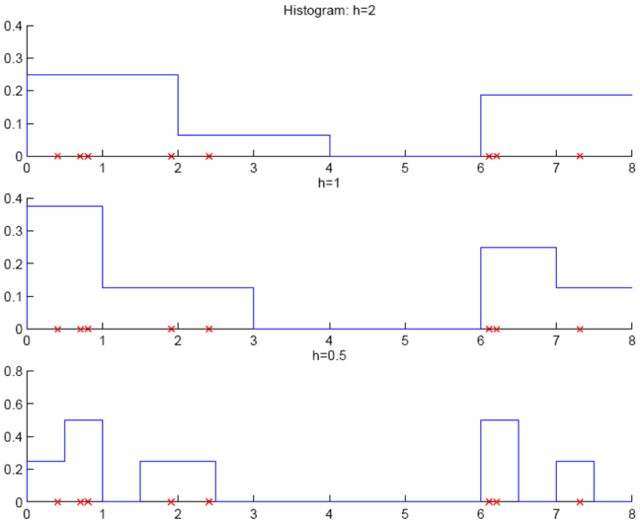
  where x_0 is the origin, h is the bin width, and m is an integer.

- Histogram estimator:

$$\hat{p}(x) = \frac{\#\{x^{(\ell)} \text{ in the same bin as } x\}}{Nh}$$

- Once the bin estimates are calculated and stored, we do not need to retain the training set.

Histogram Estimator with Different Bin Sizes



Naive Estimator

- Unlike the histogram estimator, this estimator frees us from setting an origin.
- Naive estimator:

$$\hat{p}(x) = \frac{\#\{x - h/2 < x^{(\ell)} \le x + h/2\}}{Nh}$$

- The bin is of size h and x is always at its center.
- Alternative form of estimator:

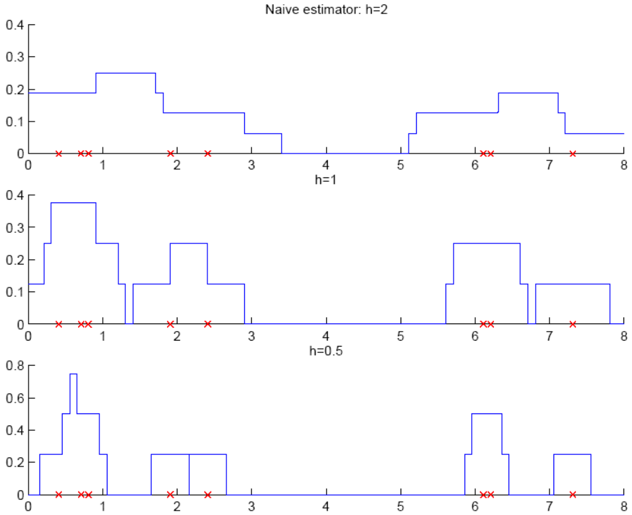
$$\hat{p}(x) = \frac{1}{Nh} \sum_{\ell=1}^{N} w\left(\frac{x - x^{(\ell)}}{h}\right)$$

  with weight function:

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

- Each $x^{(\ell)}$ has a symmetric region of influence of size h around it and contributes 1 for an x falling in its region. The nonparametric estimate is the sum of influences of $x^{(\ell)}$ whose regions include x, i.e., sum of "boxes."

Naive Estimator with Dieffrent Bin Sizes



Kernel Estimator

- Histogram estimator and naive estimator are not smooth at bin boundaries.
- To get a smooth estimator, a smooth weight function called kernel function is used, e.g., Gaussian kernel:

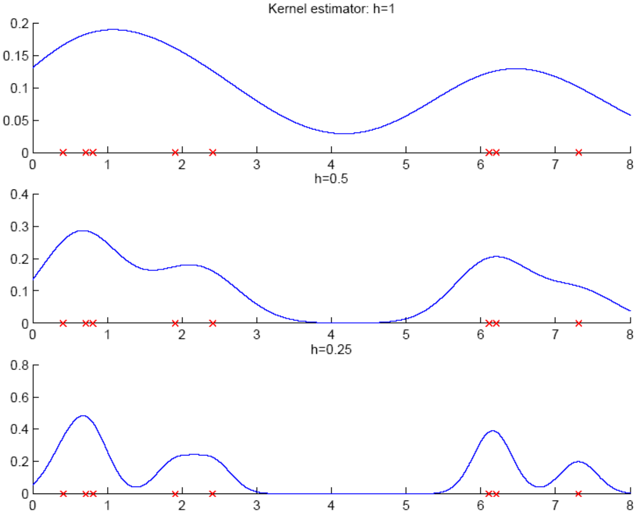
$$K(u) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right)$$

- Kernel estimator (a.k.a. Parzen windows):

$$\hat{p}(x) = \frac{1}{Nh} \sum_{\ell=1}^{N} K\left(\frac{x - x^{(\ell)}}{h}\right)$$

  where $K(\cdot)$ determines the shape of the influences and h determines the width. $K(\cdot)$ should be everywhere nonnegative and integrates to 1.
- It is a sum of N smooth local functions.

Kernel Estimator with Different Window Widths



Properties of Kernel Estimator

- All the $x^{(\ell)}$ have an effect on the estimate at x and this effect decreases smoothly as $|x - x^{(\ell)}|$ increases.
- When h is small, each training instance has a large effect in a small region.
- When h is large, there is more overlap of the kernels and the estimator is smoother.
- One problem with this estimator is that the window width h is fixed across the entire input space.

k-Nearest Neighbor Estimator I

- While kernel estimator uses the same window width everywhere, the nearest neighbor class of estimators adapts the amount of smoothing to the local density of data.
- The degree of smoothing is controlled by $k(\ll N)$, the number of neighbors taken into account.
- k-nearest neighbor (k-NN) estimator:

$$\hat{p}(x) = \frac{k}{2Nd_k(x)}$$

  where $d_k(x)$ is the distance from x to the kth nearest instance.
- This is like a naive estimator with $h = 2d_k(x)$, the difference being that instead of fixing h and checking how many samples fall in the bin, we fix k, the number of observations to fall in the bin, and compute the bin size.
- When the data density is high, the bins are small; when it is low, the bins are larger.

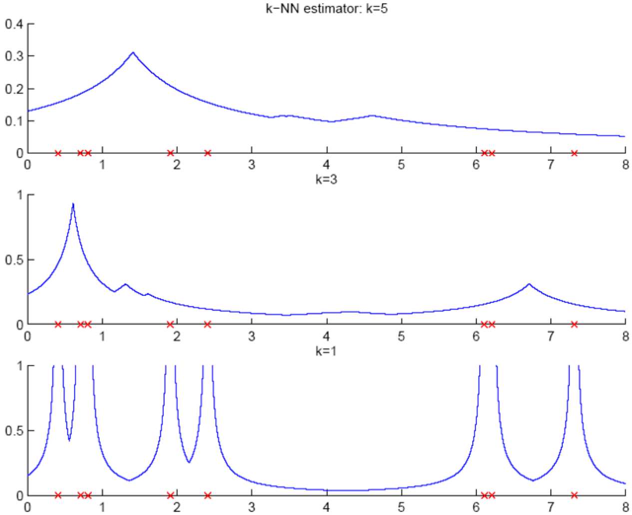
k-Nearest Neighbor Estimator II

- The k-NN estimator is not continuous and hence is not a probability density function since it integrates to ∞, not 1.
- k-nearest neighbor (k-NN) estimator with a kernel function:

$$\hat{p}(x) = \frac{1}{N d_k(x)} \sum_{\ell=1}^{N} K\left(\frac{x - x^{(\ell)}}{d_k(x)}\right)$$

  where $K(\cdot)$ is typically chosen to be the Gaussian kernel.
- This estimator is like a kernel estimator with adaptive smoothing parameter $h = d_k(x)$.

k-Nearest Neighbor Estimator with Different k Values



Outline

Generalization to Multivariate Case I

- A sample of *d*-dimensional observations $\mathcal{X} = \{\mathbf{x}^{(\ell)}\}_{\ell=1}^{N}$
- Multivariate kernel density estimator:

$$\hat{p}(x) = \frac{1}{Nh^d} \sum_{\ell=1}^{N} K\Big(\frac{\mathbf{x} - \mathbf{x}^{(\ell)}}{h}\Big)$$

  with the requirement that

$$\int_{\mathbb{R}^d} K(\mathbf{x}) d\mathbf{x} = 1$$

- Multivariate Gaussian kernel:

$$K(\mathbf{u}) = \Big(\frac{1}{\sqrt{2\pi}}\Big)^d \exp\Big(-\frac{\|\mathbf{u}\|^2}{2}\Big)$$

Generalization to Multivariate Case II

- Instead of using a single smoothing parameter h for all dimensions which corresponds to using the Euclidean distance, generalization to Mahalanobis distance gives the multivariate ellipsoidal Gaussian kernel:

$$K(\mathbf{u}) = \frac{1}{(2\pi)^{d/2}|\mathbf{S}|^{1/2}} \exp\left(-\frac{1}{2}\mathbf{u}^T\mathbf{S}^{-1}\mathbf{u}\right)$$

  where $\mathbf{S}$ is the (general) sample covariance matrix.
- Curse of dimensionality: nonparametric estimation in high-dimensional spaces may require many bins, most of which end up being empty.

Outline

Nonparametric Classification I

- Classification based on density estimation:
  - **Step 1**: estimate the class-conditional densities $p(\mathbf{x} \mid C_i)$ (parametric or nonparametric approach).
  - **Step 2**: use Bayes' rule to compute the posterior class probabilities and make optimal decision.
- Kernel estimator of class-conditional densities:

$$\hat{p}(\mathbf{x} \mid C_i) = \frac{1}{N_i h^d} \sum_{\ell=1}^{N} K\left(\frac{x - x^{(\ell)}}{h}\right) r_i^{(\ell)}$$

where

$$r_i^{(\ell)} = \begin{cases} 1 & \text{if } \mathbf{x}^{(\ell)} \text{ is in } C_i \\ 0 & \text{otherwise} \end{cases}$$

and $N_i = \sum_\ell r_i^{(\ell)}$.

Nonparametric Classification II

- MLE of prior probabilities:

$$\hat{p}(C_i) = \frac{N_i}{N}$$

- Discriminant functions:

$$g_i(\mathbf{x}) = \hat{p}(\mathbf{x} \mid C_i)\hat{P}(C_i) = \frac{1}{Nh^d}\sum_{\ell=1}^{N} K\Big(\frac{x - x^{(\ell)}}{h}\Big) r_i^{(\ell)}$$

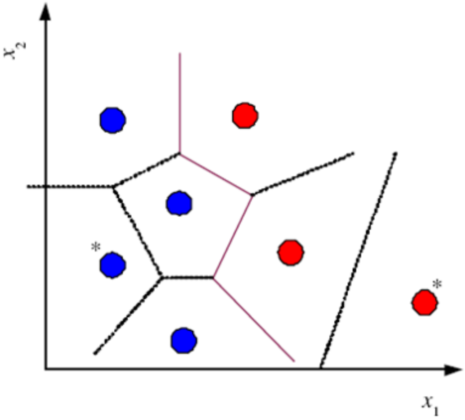
where the common factor $1/(Nh^d)$ can be ignored.

k-NN Classifier

- k-NN estimator:
$$\hat{p}(\mathbf{x} \mid C_i) = \frac{k_i}{N_i V^k(\mathbf{x})}$$
where k_i is the number of neighbors that belong to C_i and $V^k(\mathbf{x})$ is the volume of the d-dimensional hypersphere centered at $\mathbf{x}$ with radius $r = \|\mathbf{x} - \mathbf{x}_{(k)}\|$ where $\mathbf{x}_{(k)}$ is the k-th nearest observation to $\mathbf{x}$ (among all neighbors from all classes of $\mathbf{x}$).
- Posterior class probabilities:
$$\hat{P}(C_i \mid \mathbf{x}) = \frac{\hat{p}(\mathbf{x} \mid C_i)\hat{P}(C_i)}{\sum_j \hat{p}(\mathbf{x} \mid C_j)\hat{P}(C_j)} = \frac{k_i/NV^k(\mathbf{x})}{\sum_j k_j/NV^k(\mathbf{x})} = \frac{k_i}{k}$$
- k-NN classifier: assigns the input $\mathbf{x}$ to the class C_i having most examples among the k neighbors of $\mathbf{x}$, i.e.,
$$i = \arg\max_j \hat{P}(C_j \mid \mathbf{x}) = \arg\max_j k_j$$

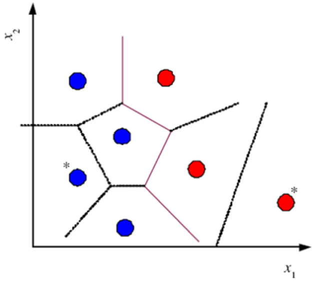
Nearest Neighbor Classifier

- Nearest neighbor classifier: special case of k-NN classier with $k = 1$.
- Voronoi tessellation formed in input space:



Condensed Nearest Neighbor

- Time/space complexity of nonparametric methods (e.g., k-NN): $O(N)$
- Condensing methods: find a small (hopefully smallest) subset $\mathcal{Z}$ of $\mathcal{X}$ such that the error does not increase when $\mathcal{Z}$ is used in place of $\mathcal{X}$.
- Condensed nearest neighbor classier: only the instances that define the discriminant need to be kept but those inside the class regions can be removed (cf. support vector machines).



Outline

Introduction

Nonparametric Density Estimation

Generalization to Multivariate Case

Nonparametric Classification

Nonparametric Regression

Nonparametric Regression

- Nonparametric regression is a.k.a. smoothing models.
- Regression problem:
$$y^{(\ell)} = g(\mathbf{x}^{(\ell)}) + \epsilon$$
where $y^{(\ell)} \in \mathbb{R}$.
- Nonparametric regression is needed when we cannot find an appropriate parametric model (e.g., polynomial) for $g(\cdot)$.
- Nonparametric regression estimators (a.k.a. smoothers):
  - Running mean smoother
  - Kernel smoother
  - Running line smoother
- Here we consider the univariate case, which can be extended easily to the multivariate case.

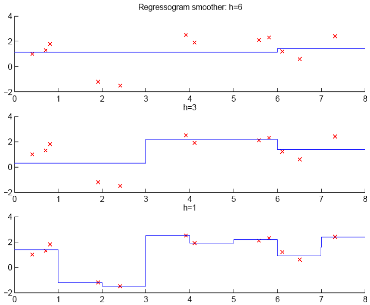
Running Mean Smoother I

- Regressogram:

$$\hat{g}(x) = \frac{\sum_{\ell=1}^{N} b(x, x^{(\ell)}) y^{(\ell)}}{\sum_{\ell=1}^{N} b(x, x^{(\ell)})}$$

where

$$b(x, x^{(\ell)}) = \begin{cases} 1 & \text{if } x^{(\ell)} \text{ is in the same bin with } x \\ 0 & \text{otherwise} \end{cases}$$



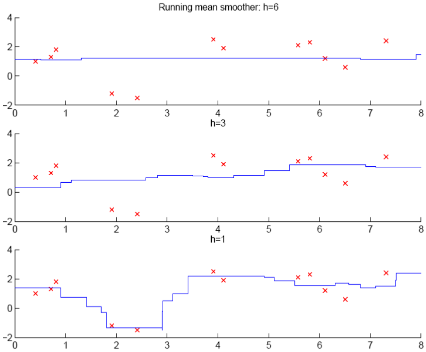
Running Mean Smoother II

- To avoid the need to fix an origin, the running mean smoother defines a bin symmetric around x:

$$\hat{g}(x) = \frac{\sum_{\ell=1}^{N} w\left(\frac{x - x^{(\ell)}}{h}\right) y^{(\ell)}}{\sum_{\ell=1}^{N} w\left(\frac{x - x^{(\ell)}}{h}\right)}$$

where

$$w(u) = \begin{cases} 1 & \text{if } |u| < 1 \\ 0 & \text{otherwise} \end{cases}$$



Kernel Smoother

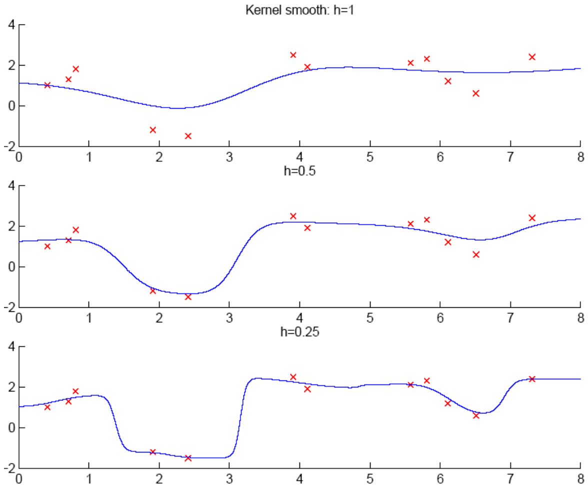
- Kernel smoother:

$$\hat{g}(x) = \frac{\sum_{\ell=1}^{N} K(\frac{x - x^{(\ell)}}{h}) y^{(\ell)}}{\sum_{\ell=1}^{N} K(\frac{x - x^{(\ell)}}{h})}$$

  where $K(\cdot)$ is a kernel, such as Gaussian kernel, that gives less weight to further points.

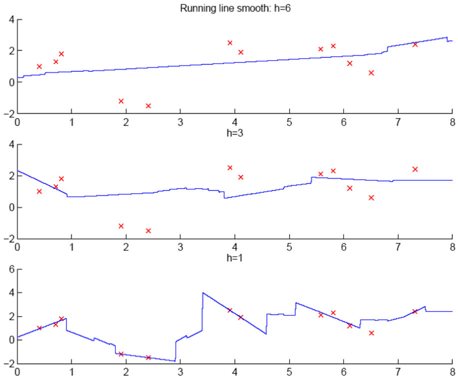
- k-NN smoother:
  Instead of fixing h, the number of neighbors k is fixed to adapt to the density around x.

Kernel Smoother with Different Bin Lengths



Running Line Smoother

- Unlike the running mean smoother which has discontinuities, the running line smoother uses continuous piecewise linear fit.



- Alternatively, kernel weighting may also be used to give the locally weighted running line smoother.

How to Choose h or k?

- Small h or k (undersmoothing): small bias but large variance.
- Large h or k (oversmoothing): large bias but small variance.
- Regularized cost function for smoothing splines:

$$\sum_{\ell} \left[y^{(\ell)} - \hat{g}(x^{(\ell)})\right]^2 + \lambda \int_a^b \left[\hat{g}''(x)\right]^2 dx$$

  - First term: error of fit
  - Second term: penalty for high variability, where $\hat{g}''(x)$ is the curvature of $\hat{g}(\cdot)$ and $[a, b]$ is the input range
  - λ: trades off error and variability and can also be determined by cross-validation.
- Cross-validation may be used to determine the best h or k.