Non-parametric Regression and LOESS

Alexander Trofimov
PhD, professor, NRNU MEPhI

lab@neuroinfo.ru
http://datalearning.ru

Course “Machine Learning”

April 2022

Non-parametric Regression Analysis

Regression model:

$$Y|x = \varphi(x) + \varepsilon(x)$$

where $\varphi(x) = \mathrm{M}[Y|x]$ is a regression function, $\varepsilon(x)$ is a random error (noise)

Non-parametric regression analysis doesn't require any explicit assumptions about conditional distribution $F_Y(y|x)$ or regression function $\varphi(x)$

How to estimate the regression function using the data only?

In very large samples it is possible to estimate $\varphi(x)$ by directly examining the conditional distribution of Y given the x, $x \in \mathscr{X}$

For discrete explanatory variables:
Estimating conditional means for each discrete value of X

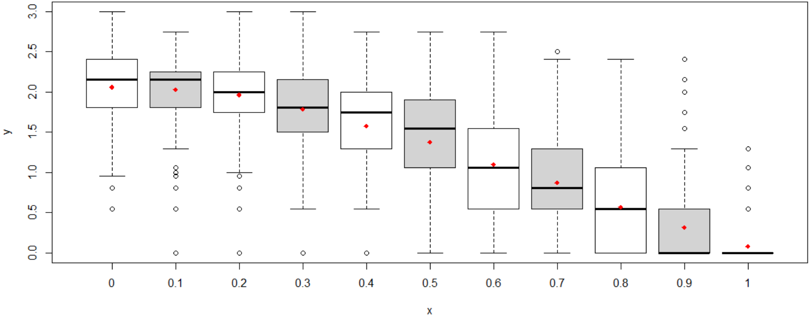
For continuous explanatory variables:
Dissecting the $X_1, ..., X_k$ into a large number of narrow bins

Conditional Means. Illustration

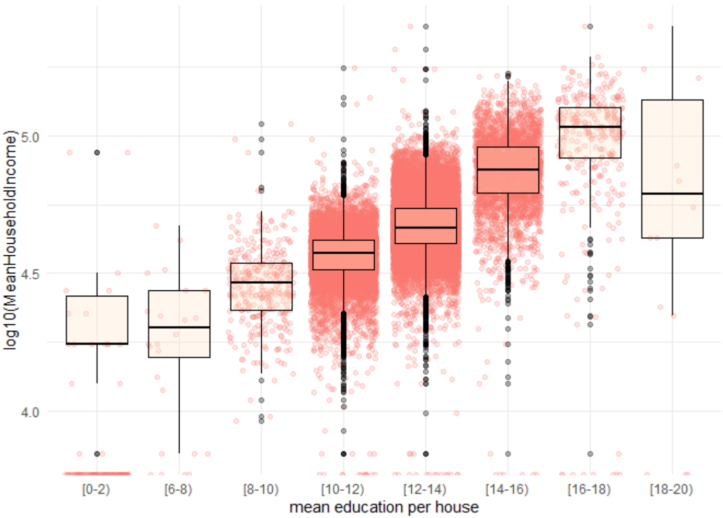
The conditional mean estimator:

$$\hat{\varphi}(x) = \hat{m}_{Y|x} = \frac{1}{n(x)} \sum_{i:x_i=x} y_i = \frac{1}{\sum_{i=1}^{n} [x_i = x]} \sum_{i=1}^{n} y_i [x_i = x]$$

where $n(x)$ is a number of observations at x



Dissection of Explanatory Variable. Illustration



Binned Regression

The input space is divided into equal-sized intervals named bins

Given an origin x_0 and a bin width h, the bins are the intervals

$$\Delta_1 = [x_0, x_0 + h), ..., \Delta_k = [x_0 + (k-1)h, x_0 + kh)$$

where k is the number of bins

The regression estimator:

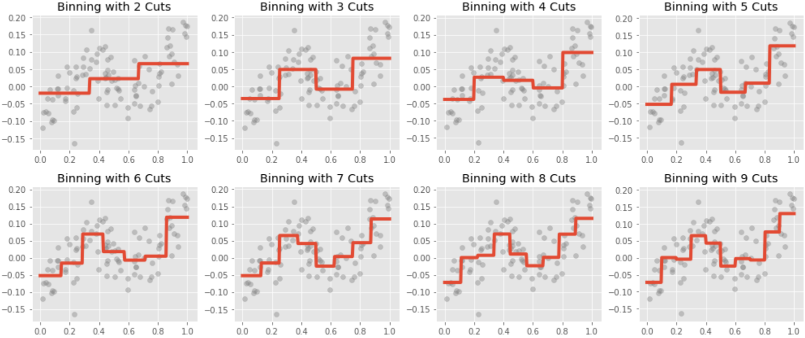
$$\hat{\varphi}(x) = \frac{1}{n_l} \sum_{i:x_i \in \Delta_l} y_i = \frac{\sum\limits_{i=1}^{n} y_i [x_i \in \Delta_l]}{\sum\limits_{i=1}^{n} [x_i \in \Delta_l]}, \quad \text{for } x \in \Delta_l$$

where n_l is the number of data points in interval Δ_l, $l \in \{1, ..., k\}$

The estimator requires two parameters: bin width h and origin x_0

Binned Regression. Illustration

The plot of binned regression is called as regressogram



$$regressogram = regression + histogram$$

Binned Regression. Notes

Advantages:

- It's easy to discover, implement, and explain
- Captures non-linear behaviour of the predictor-response relationship

Drawbacks:

- The regression function $\varphi(x)$ is assumed to be constant inside bins
- The estimated regression function $\hat{\varphi}(x)$ is a piecewise constant function
- The discontinuities of the estimate
- The $\hat{\varphi}(x)$ depends on the origin x_0
- The curse of dimensionality: the number of bins grows exponentially with the number of regressors

Naive Regression

Naive estimator of $\varphi(x)$:

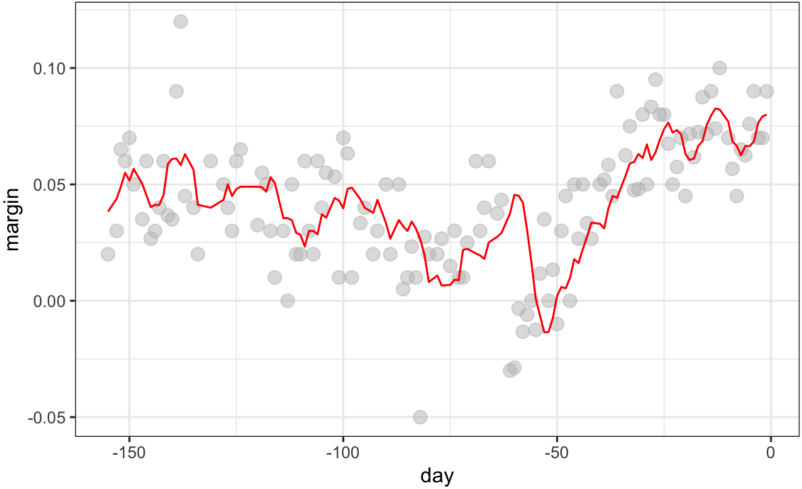
$$\hat{\varphi}(x) = \frac{\sum\limits_{i=1}^{n} y_i[x - h/2 \le x_i < x + h/2]}{\sum\limits_{i=1}^{n}[x - h/2 \le x_i < x + h/2]} = \sum_{i=1}^{n} w_i(x) y_i$$

where

$$w_i(x) = \frac{[x - h/2 \le x_i < x + h/2]}{\sum\limits_{l=1}^{n}[x - h/2 \le x_l < x + h/2]} = \frac{\left[\left|\frac{x-x_i}{h}\right| < \frac{1}{2}\right]}{\sum\limits_{l=1}^{n}\left[\left|\frac{x-x_l}{h}\right| < \frac{1}{2}\right]}$$
$$= \begin{cases} \frac{1}{n(x)}, & \left|\frac{x-x_i}{h}\right| < \frac{1}{2}, \\ 0, & otherwise \end{cases}$$

where $n(x)$ is the number of points in the neighbourhood around x

Naive Regression. Illustration



Naive regression plot (.gif)

Box Kernel

The weights of the naive estimator can be written as:

$$w_i(x) = \frac{\left[\left|\frac{x-x_i}{h}\right| < \frac{1}{2}\right]}{\sum\limits_{l=1}^{n} \left[\left|\frac{x-x_l}{h}\right| < \frac{1}{2}\right]} = \frac{K(\frac{x-x_i}{h})}{\sum\limits_{l=1}^{n} K(\frac{x-x_l}{h})}$$

where $K(u) = \begin{cases} 1, & |u| < 1/2 \\ 0, & otherwise \end{cases}$ is the box kernel function

Naive regression estimator:

$$\hat{\varphi}(x) = \sum_{i=1}^{n} w_i(x) y_i = \frac{\sum\limits_{i=1}^{n} y_i K(\frac{x-x_i}{h})}{\sum\limits_{l=1}^{n} K(\frac{x-x_l}{h})}$$

Naive Regression. Notes

- The weights $w_i(x)$, $i = \overline{1, n}$, are not continuous functions and have jumps at $x_i \pm h/2$
- The estimate $\hat{\varphi}(x)$ has discontinuities
- The naive regression acts like a moving average filter, the moving average window size is defined by the smoothing parameter h
- The regression function $\varphi(x)$ is assumed to be a constant inside the moving window, $\varphi(x) \equiv c \;\; \forall x \in \Delta_h(x)$. Under this assumption, the estimate $\hat{\varphi}(x)$ is unbiased:

  $$\mathrm{M}[\hat{\varphi}(x)] = \mathrm{M}\left[\frac{1}{n(x)} \sum_{x_i \in \Delta_h(x)} Y|x_i\right] = \varphi(x)$$

  where $\Delta_h(x) = [x - h/2, x + h/2)$
- The estimates at the ends of the data range are unreliable

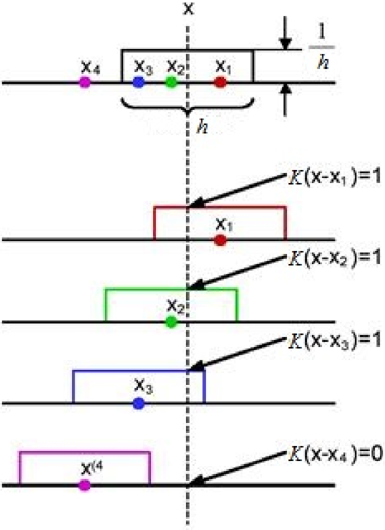
Region of Influence. Illustration

$$w_i(x) = \frac{K(\frac{x-x_i}{h})}{\sum\limits_{l=1}^{n} K(\frac{x-x_l}{h})}, \; i = \overline{1, n}$$

Each x_i has a symmetric region of influence of size h around it and contributes 1 for an x falling in its region

The weight $w_i(x)$ is influence of x_i on x among overall influences of $x_1, ..., x_n$ on x

For box kernel the influences from all $x_1, ..., x_n$ are hard (0 or 1)



Kernel Estimator

Kernel regression estimator:

$$\hat{\varphi}(x) = \sum_{i=1}^{n} w_i(x) y_i = \frac{\sum\limits_{i=1}^{n} y_i K(\frac{x-x_i}{h})}{\sum\limits_{l=1}^{n} K(\frac{x-x_l}{h})}$$

where $K(u)$ is the kernel function, h is the smoothing parameter also called the bandwidth

Kernel regression estimator was proposed in 1964 and it's also called as Nadaraya-Watson estimator

The naive estimator is a particular case of kernel estimator with box kernel function

For naive estimator, the estimate $\hat{\varphi}(x)$ has discontinuities

How can this problem be solved?

Properties of Kernel Function

Idea: instead of giving equal weight to every point in the region of influence of x_i, let's assign a weight which decays toward zero in a continuous fashion as we get further away from x_i, $i = 1, ..., n$

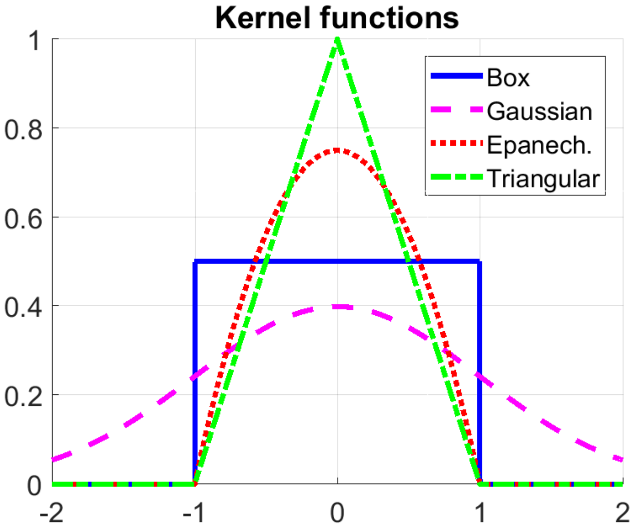
Kernel function $K(u)$ is usually chosen as a symmetric probability density function satisfying the conditions:

- $K(u) \geq 0 \quad \forall u \in \mathbb{R}$
- $\int_{-\infty}^{\infty} K(u)du = 1$
- $K(u) = K(-u) \quad \forall u \in \mathbb{R}$
- $\int_{-\infty}^{\infty} uK(u)du = 0$
- $\int_{-\infty}^{\infty} u^2 K(u)du = \sigma_K^2 < \infty$

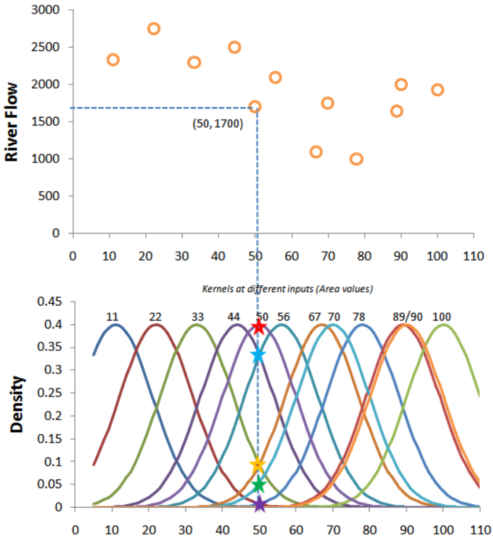
Types of Kernel Functions

- Box (uniform) kernel: $K(u) = \begin{cases} \frac{1}{2}, & |u| < 1 \\ 0, & otherwise \end{cases}$
- Gaussian kernel: $K(u) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right)$
- Epanechnikov kernel: $K(u) = \begin{cases} \frac{3}{4}(1-u^2), & |u| < 1 \\ 0, & otherwise \end{cases}$
- Triangular kernel: $K(u) = \begin{cases} 1-|u|, & |u| < 1 \\ 0, & otherwise \end{cases}$

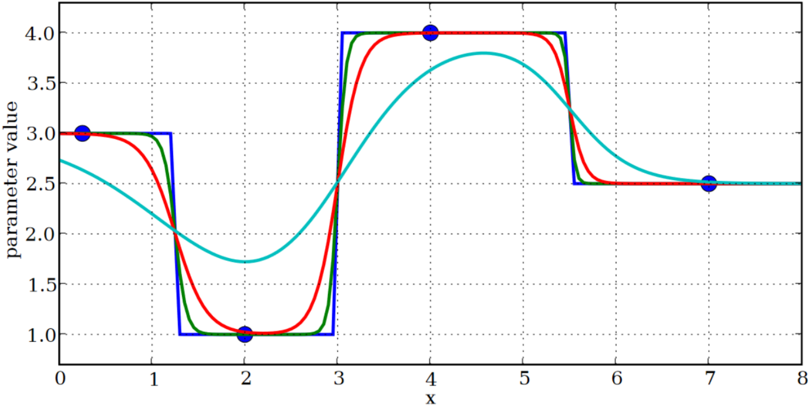
Kernel Functions. Illustration



Gaussian Kernel Regression. Illustration 1

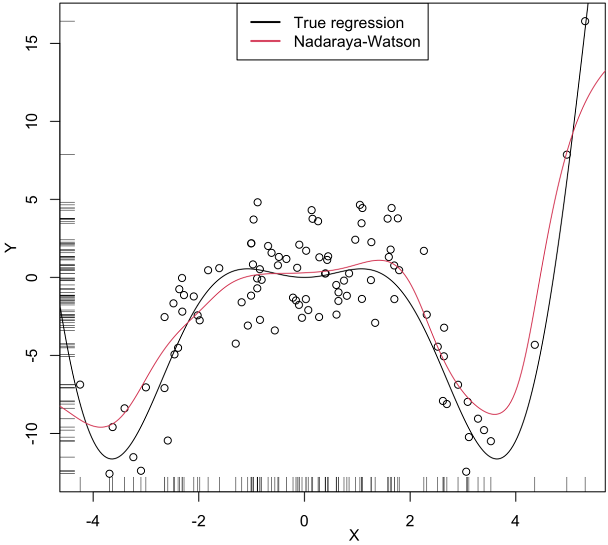


Gaussian Kernel Regression. Illustration 2



$h = 0.1$ (blue), $h = 0.3$ (green), $h = 0.5$ (red), $h = 1$ (cyan)

Gaussian Kernel Regression. Illustration 3



Kernel Regression. Notes

- The estimate $\hat{\varphi}(x)$ is a weighted sum of responses $y_1, ..., y_n$ with weights $w_1(x), ..., w_n(x)$, where

$$\sum_{i=1}^{n} w_i(x) = 1 \text{ and } w_i(x) > 0,\ i = \overline{1, n}$$

- It's a kind of linear smoothing techniques
- It's a kind of local regression techniques: only the observations close to the query point are considered for regression computation
- An observation point gets a weight that decreases as its distance from the query point increases
- No need for offline training
- It's a memory-based technique as it requires entire training set to be available while regressing

KNN Regression

The idea of K nearest neighborhood regression (KNN regression) is to identify K training observations that are closest to the query point and to average responses over them:

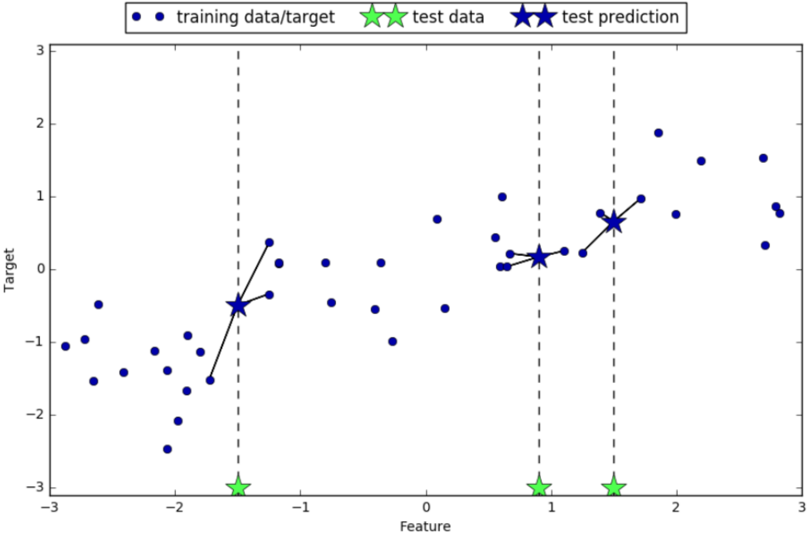
$$\hat{\varphi}(x) = \frac{1}{K} \sum_{x_i \in \Delta_K(x)} y_i$$

where $\Delta_K(x)$ is the neighborhood of x that contains K closest points from the training data $x_1, ..., x_n$

KNN regression can be seen as box kernel regression with varying bandwidth $h(x)$ that depends on x so that the symmetric window around x contains K data points:

$$\hat{\varphi}(x) = \sum_{i=1}^{n} w_i(x) y_i = \frac{\sum\limits_{i=1}^{n} y_i K\left(\frac{x-x_i}{h(x)}\right)}{\sum\limits_{i=1}^{n} K\left(\frac{x-x_i}{h(x)}\right)}$$

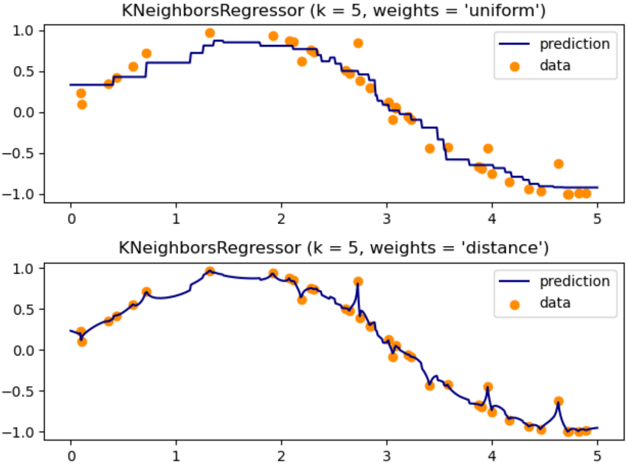
KNN Regression. Illustration



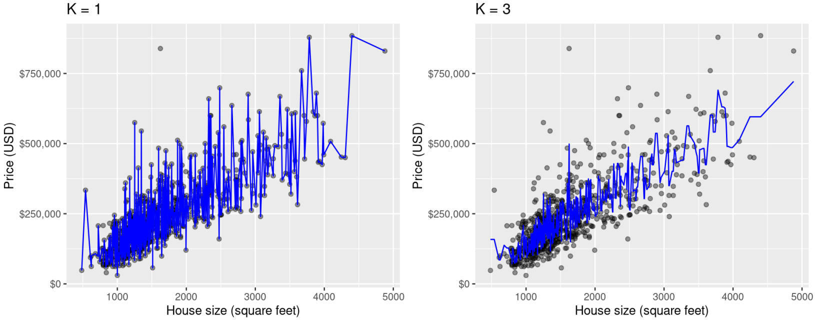
KNN Regression. Notes

- The estimate $\hat{\varphi}(x)$ has discontinuities
- KNN regression can be used with other kernels (Gaussian, triangular, etc.)
- It's a kind of linear smoothing techniques
- It's a kind of local regression techniques
- In KNN regression the amount of smoothing is adapted with according to the density of the predictor
- A small value of K provides the most flexible fit, which will have low bias but high variance
- Larger values of K provide a smoother and less variable fit, too large K results to high bias

KNN Regression. Illustration 1



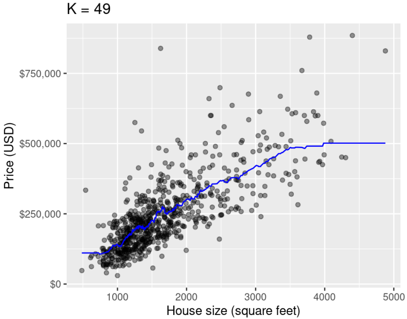
KNN Regression. Illustration 2



For $K = 1$ the estimate passes through all training data

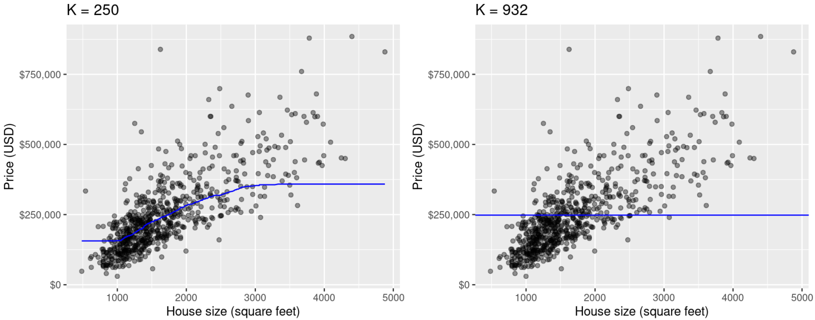
Lower K results to overfitting, the model is influenced too much by the noisy data

KNN Regression. Illustration 3



For larger K, KNN with box kernel still has discontinuities, the estimate is not rather smooth

KNN Regression. Illustration 4



For $K = n$ the estimated regression function is a horizontal line

Larger K results to underfitting, the model is not influenced enough by the training data

Smoothing

The non-parametric regression is related to the concept of data smoothing

Data smoothing (curve fitting, or low pass filtering) is the process of removing noise from dataset that allows important patterns in data to stand out

The regression model of response Y for a given $x \in \mathscr{X}$:

$$Y|x = \varphi(x) + \varepsilon(x)$$

where $\varphi(x) = \mathrm{M}[Y|x]$ is regression function, $\varepsilon(x)$ is a noise, $\mathrm{M}[\varepsilon(x)] = 0$, $\forall x \in \mathscr{X}$, and $\mathrm{D}[\varepsilon(x)] = \sigma_x^2$

By removing the noise $\varepsilon(x)$, we obtain the estimate $\hat{\varphi}(x)$ of the regression function $\varphi(x)$

The regression estimators based on data smoothing are called as smoothers

Linear Smoothing

Definition

For a given sample $(x_1, y_1), ..., (x_n, y_n)$, an estimator $\hat{\varphi}(x)$ of regression function $\varphi(x)$ is a linear smoother if, for each $x \in \mathscr{X}$, there exists a vector $w(x) = (w_1(x), ..., w_n(x))$ such that

$$\hat{\varphi}(x) = w(x)y = \sum_{j=1}^{n} w_j(x) y_j$$

The linearly smoothed sample:

$$\hat{y}_i = \hat{\varphi}(x_i) = \sum_{j=1}^{n} w_j(x_i) y_j, \quad i = 1, ..., n$$

where $w_j(x_i)$ is a contribution of observation y_j to the smoothed value $\hat{y}_i$

Smoothing Matrix

Linear smoothing in matrix form:

$$\hat{y} = Wy$$

where

$y = (y_1, ..., y_n)^T$ is a vector of responses,

$\hat{y} = (\hat{\varphi}(x_1), ..., \hat{\varphi}(x_n))^T$ is the vector of fitted values at $x_1, ..., x_n$,

W is the smoother matrix:

$$W = \begin{pmatrix} w_1(x_1) & ... & w_n(x_1) \\ ... & ... & ... \\ w_1(x_n) & ... & w_n(x_n) \end{pmatrix}$$

In parametric regression analysis the smoother matrix is called as hat matrix, $W = H = X(X^TX)^{-1}X^T$

Linear regression, kernel regression and KNN regression are kinds of linear smoothers

Unbiasedness of Linear Smoother

Is the estimate $\hat{\varphi}(x)$ unbiased?

$$\mathrm{M}[\hat{\varphi}(x)] = \sum_{i=1}^{n} w_i(x)\mathrm{M}[Y|x_i] \overset{?}{=} \varphi(x)$$

If the contributions $w_i(x)$, $i = \overline{1, n}$, are non-zero only in a small neighbourhood $\Delta_h(x)$ around x where the regression function is locally constant, $\mathrm{M}[Y|x_i] = c,\ \forall x_i \in \Delta_h(x)$, then the estimate $\hat{\varphi}(x)$ is unbiased

We need small bandwidth h to hold assumptions of approximately constant $\varphi(x)$, but a small number of data points will be used to average and obtain estimate $\tilde{\varphi}(x)$

Is it possible to use larger window sizes without the bias to be increased?

Locally Linear Regression

Idea: instead of assuming the regression function $\varphi(x)$ is approximately constant in a window, let's assume it is locally linear

Taylor expansion of $\varphi(x)$ in the neighborhood of x:

$$\varphi(x_i) = \varphi(x) + \varphi'(x)(x_i - x) + o(x_i - x), \quad x_i \in \Delta_h(x)$$

Since $\varphi(x)$ and $\varphi'(x)$ are unknown, let's fit them!

Local OLS criterion:

$$\begin{aligned} E(x) &= \sum_{x_i \in \Delta_h(x)} (y_i - \varphi(x_i))^2 \approx \sum_{x_i \in \Delta_h(x)} [y_i - (\varphi(x) + \varphi'(x)(x_i - x))]^2 \\ &= \sum_{x_i \in \Delta_h(x)} [y_i - (\beta_0 + \beta_1(x_i - x))]^2 \to \min_{\beta_0, \beta_1} \end{aligned}$$

where $\beta_0 = \varphi(x)$, $\beta_1 = \varphi'(x)$

Local OLS Criterion

Local OLS criterion is the sum of errors over the neighbourhood $\Delta_h(x)$, let's rewrite:

$$E(x) = \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)\left[y_i - (\beta_0 + \beta_1(x_i - x))\right]^2 \to \min_{\beta_0,\beta_1}$$

In matrix form:

$$E(x) = (y - X\beta)^T V (y - X\beta) \to \min_{\beta}$$

where $y = (y_1, ..., y_n)^T$ is vector of responses, $\beta = (\beta_0, \beta_1)^T$ is vector of parameters, X is design matrix:

$$X = \begin{pmatrix} 1 & x_1 - x \\ ... & ... \\ 1 & x_n - x \end{pmatrix}$$

and V is weight matrix: $V = diag\left[K\left(\frac{x-x_1}{h}\right), ..., K\left(\frac{x-x_n}{h}\right)\right]$

Locally Linear Estimator

The local OLS problem $E(x) \to \min_{\beta}$ is indeed a weighted least squares (WLS) problem

The solution:

$$\hat{\beta} = (X^T V X)^{-1} X^T V y$$

The estimate $\hat{\varphi}(x)$:

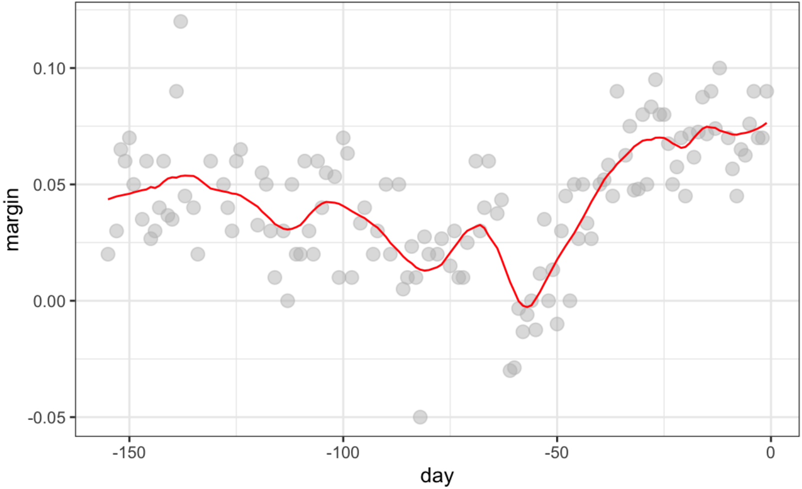
$$\hat{\varphi}(x) = \hat{\beta}_0 = (1\ 0)\hat{\beta} = (1\ 0)(X^T V X)^{-1} X^T V y$$
$$= w(x) y = \sum_{j=1}^{n} w_j(x) y_j$$

where

$$w(x) = (1\ 0)(X^T V X)^{-1} X^T V$$

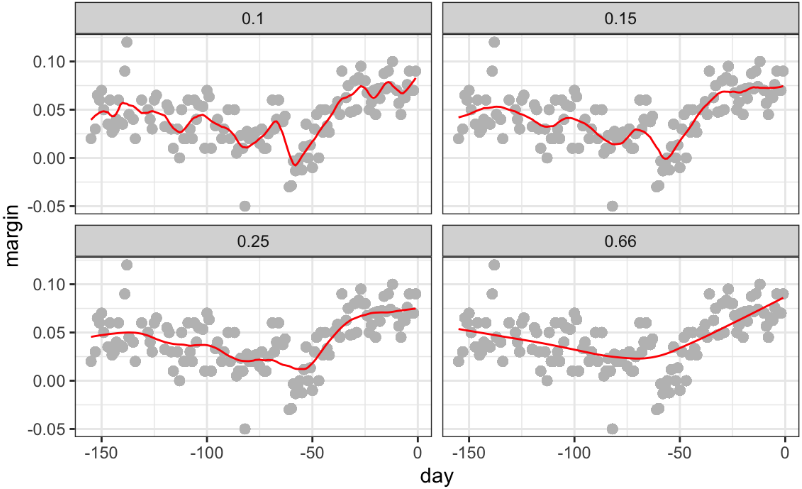
The locally linear estimator is a weighted linear combination of the responses, and like Nadaraya–Watson estimator, it's a kind of linear smoothers

Locally Linear Regression. Illustration 1



Locally linear regression plot (.gif)

Locally Linear Regression. Illustration 2



Locally linear regression for different bandwidths (.gif)

Locally Linear Regression. Notes

Advantages:

- Locally linear regression is more general that locally constant (Nadaraya-Watson) regression, and it has lower bias
- It relies on the local data structure when performing the local fitting
- The process of fitting a model to the sample data does not require the specification of a regression function $\varphi(x)$
- Logical simplicity of the method

Drawbacks:

- It does not produce an estimate $\hat{\varphi}(x)$ that is easily represented by a mathematical formula
- Can be inefficient at discovering some relatively simple (e.g., linear) structures in data
- Computationally intensive method

LOESS Estimator

The locally linear regression can be generalized to locally polynomial regression:

$$\varphi(x_i) \approx \varphi(x)+\varphi'(x)(x_i-x)+\frac{\varphi''(x)}{2}(x_i-x)^2+\ldots+\frac{\varphi^{(k)}(x)}{k!}(x_i-x)^k$$

The regression function $\varphi(x)$ is locally approximated by a k-order polynomial in the neighbourhood $\Delta_h(x)$, $x \in \mathscr{X}$:

$$E = \sum_{x_i \in \Delta_h(x)} (y_i - \varphi(x_i))^2 \approx \sum_{x_i \in \Delta_h(x)} (y_i - x_i'\beta)^2 \to \min_{\beta}$$

where

$$x_i' = (1, x_i - x, \ldots, \tfrac{1}{k!}(x_i - x)^k), \quad \beta = (\varphi(x), \varphi'(x), \ldots, \varphi^{(k)}(x))^T$$

The local polynomial estimator was proposed by W. Cleveland in 1979, further developed in 1988, and called as LOESS estimator (LOcally Estimated Scatterplot Smoothing)

LOESS Estimator. Notes

- The idea of local polynomial regression was proposed by A. Savitzky and M. Golay in 1964
- The polynomial order k shouldn't be large. The idea is that regression function can be well approximated in a small neighborhood by a low-order polynomial and that simple models can be fitted to data easily
- High-degree polynomials would tend to overfit the data in each window and are numerically unstable, making accurate computations difficult
- Nadaraya-Watson regression is a particular case of locally polynomial regression if $k = 0$
- Locally linear regression is a particular case of locally polynomial regression if $k = 1$
- In LOESS regression the bandwidth h is usually controlled by the number of points in the moving window

Bias-Variance Decomposition

How to choose the bandwidth h?

Selecting the bandwidth h for the kernel estimator is primarily a matter of trial and error – we want h to be small enough to reveal details but large enough to suppress random noise

Expectation of squared error estimate at given x:

$$e^2(x) = \mathrm{M}\left[(\tilde{\varphi}(x) - \varphi(x))^2\right] = \mathrm{M}\left[\tilde{\varphi}^2(x)\right] - 2\varphi(x)\mathrm{M}\left[\tilde{\varphi}(x)\right] + \varphi^2(x)$$

Bias-Variance Decomposition

How to choose the bandwidth h?

Selecting the bandwidth h for the kernel estimator is primarily a matter of trial and error – we want h to be small enough to reveal details but large enough to suppress random noise

Expectation of squared error estimate at given x:

$$\begin{aligned} e^2(x) = \mathrm{M}\left[(\tilde{\varphi}(x) - \varphi(x))^2\right] &= \mathrm{M}\left[\tilde{\varphi}^2(x)\right] - 2\varphi(x)\mathrm{M}\left[\tilde{\varphi}(x)\right] + \varphi^2(x) \\ &= \mathrm{D}\left[\tilde{\varphi}(x)\right] + (\mathrm{M}\left[\tilde{\varphi}(x)\right])^2 - 2\varphi(x)\mathrm{M}\left[\tilde{\varphi}(x)\right] + \varphi^2(x) \end{aligned}$$

Bias-Variance Decomposition

How to choose the bandwidth h?

Selecting the bandwidth h for the kernel estimator is primarily a matter of trial and error – we want h to be small enough to reveal details but large enough to suppress random noise

Expectation of squared error estimate at given x:

$$\begin{aligned} e^2(x) = \mathrm{M}\left[(\tilde\varphi(x)-\varphi(x))^2\right] &= \mathrm{M}\left[\tilde\varphi^2(x)\right] - 2\varphi(x)\mathrm{M}\left[\tilde\varphi(x)\right] + \varphi^2(x) \\ &= \mathrm{D}\left[\tilde\varphi(x)\right] + \left(\mathrm{M}\left[\tilde\varphi(x)\right]\right)^2 - 2\varphi(x)\mathrm{M}\left[\tilde\varphi(x)\right] + \varphi^2(x) \\ &= \mathrm{D}\left[\tilde\varphi(x)\right] + \left(\mathrm{M}\left[\tilde\varphi(x)\right] - \varphi(x)\right)^2 = Var(x) + Bias^2(x) \end{aligned}$$

Two sources of error in regression estimation:

- Bias

  It's a systematic error incurred in the estimation
- Variance

  It's a random error incurred in the estimation

Bias, Variance and Bandwidth

It can be shown that

$$Var(x) \approx \frac{R(K)}{nhf_X(x)}\sigma^2(x)$$

$$\text{for } k=0: \quad Bias(x) \approx \frac{\mu_2(K)}{2}\left[\varphi''(x) + 2\frac{\varphi'(x)f'_X(x)}{f_X(x)}\right]h^2$$

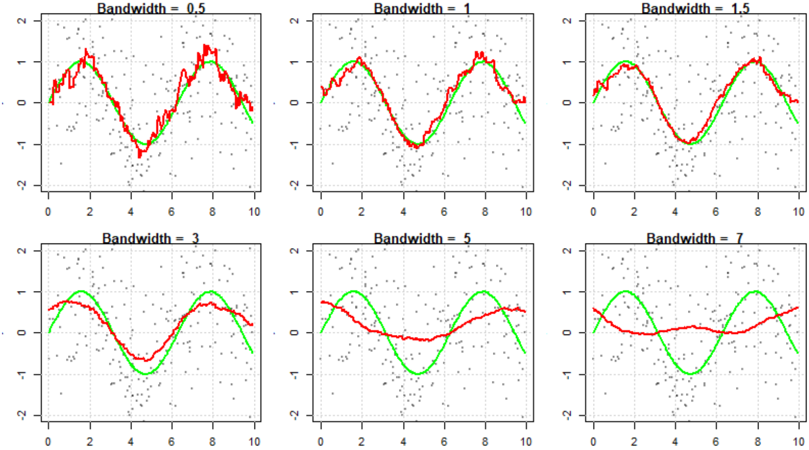
$$\text{for } k=1: \quad Bias(x) \approx \frac{\mu_2(K)}{2}\varphi''(x)h^2$$

where $\mu_2(K)$ is the 2-nd moment of kernel K, $f_X(x)$ is the pdf of X, $R(K) = \int_{-\infty}^{\infty} K^2(u)du$, $\sigma^2(x) = \mathrm{D}[Y|x]$ is the variance of noise

Trade-off between $Bias(x)$ and $Var(x)$ depends on bandwidth h:

- **Small** $h \Rightarrow$ small bias at the expense of a larger variance in the estimates for different training samples (undersmoothing)
- **Large** $h \Rightarrow$ small differences among the estimates for different training samples (oversmoothing)

Bias, Variance and Bandwidth. Illustration



Bias, Variance and Bandwidth. Notes

- The bias decreases with h quadratically both for Nadaraya-Watson regression and locally linear regression

  Small bandwidths h give estimators with low bias, whereas large bandwidths provide largely biased estimators
- The bias at x is directly proportional to $\varphi''(x)$ if $k = 1$ or affected by $\varphi''(x)$ if $k = 0$

  The higher the curvature of $\varphi(x)$ the higher the bias
- The variance is inversely proportional to the density $f_X(x)$

  The lower the density $f_X(x)$, the more variable the estimate $\tilde{\varphi}(x)$
- The variance decreases at a factor of $(nh)^{-1}$ which can be thought of as the amount of data in the neighborhood of x that is employed for performing the regression
- The locally linear regression has smaller bias than Nadaraya-Watson regression while keeping the same variance

Optimal Bandwidth

Mean integrated squared error (MISE) of estimator $\tilde{\varphi}(x)$:

$$MISE(\tilde{\varphi}) = \mathrm{M}[e^2(x)] = \int\limits_{-\infty}^{\infty} \left[Var(\tilde{\varphi}(x)) + Bias^2(x)\right] f_X(x)dx$$

For $k = 1$:

$$MISE(\tilde{\varphi}) = \frac{R(K)}{nh} \int\limits_{-\infty}^{\infty} \sigma^2(x)dx + h^4 \int\limits_{-\infty}^{\infty} \left(\frac{\mu_2(K)}{2}\varphi''(x)f_X(x)\right)^2 dx$$

The bandwidth h_{MISE} that minimizes the MISE:

$$h_{MISE} = \left(\frac{R(K)\int \sigma^2(x)dx}{\mu_2^2(K)\int \varphi''(x)^2 f_X^2(x)dx}\right)^{1/5} n^{-1/5}$$

Bandwidth Selection Approaches

- Guided trial and error
  If the fitted regression looks too rough, then try increasing the bandwidth; if it looks oversmoothed, then try decreasing it
- Assumptions about $\varphi(x)$
  Optimal bandwidth h_{MISE} depends on unknown regression function $\varphi(x)$. It can be calculated only under certain assumptions about $\varphi(x)$
- Cross-validation methods
  The optimal bandwidth $h_{CV} = \arg\min_{h>0} E_{CV}(h)$, where $E_{CV}(h)$ is average error over validation sample for a given h
- Plug-in methods
  Replace $\varphi''(x)$ in h_{MISE} expression to an estimation $\hat{\varphi}''(x)$ at pilot bandwidth $g = g(h_{MISE})$, and $\sigma^2(x)$ to the estimated variance of errors under homoscedasticity assumption

Multiple Kernel Regression

Let's consider that there is m explanatory variables $x_1, ..., x_m$

Regression model:

$$Y|x = \varphi(x) + \varepsilon(x)$$

where $x = (x_1, ..., x_m)$ is a vector of regressors, $\varphi(x) = \mathrm{M}[Y|x]$ is regression function, $\varepsilon(x)$ is a random error (noise)

Kernel regression estimator:

$$\hat{\varphi}(x) = \sum_{i=1}^{n} w_i(x) y_i = \frac{\sum\limits_{i=1}^{n} y_i K(\frac{x - x_i}{h})}{\sum\limits_{i=1}^{n} K(\frac{x - x_i}{h})}$$

The kernel function $K(u)$ should be multivariable, $K(u_1, ..., u_m)$

How to construct multivariable kernel function $K(u)$ in m-dimensional space?

Product Kernel Regression Estimator

Idea: since the kernel is a probability density function of some random vector, let's assume that it's independent random vector

Product kernel:

$$K(u) = K(u_1) \cdot ... \cdot K(u_m)$$

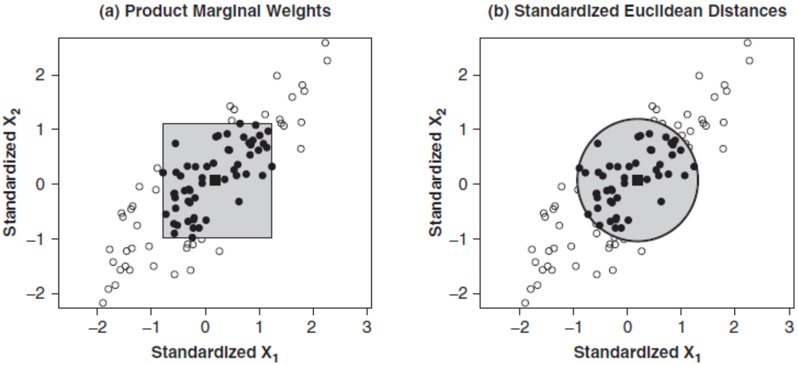
Product kernel regression estimator:

$$\hat{\varphi}(x) = \sum_{i=1}^{n} w_i(x) y_i = \frac{\sum\limits_{i=1}^{n} y_i \prod\limits_{j=1}^{m} K(\frac{x_j - x_{ji}}{h_j})}{\sum\limits_{l=1}^{n} \prod\limits_{j=1}^{m} K(\frac{x_j - x_{ji}}{h_j})}$$

$K(u)$ is a box kernel ⇒ the region of influence is hypercube

$K(u)$ is a Gaussian kernel ⇒ the region of influence is multivariate gaussian

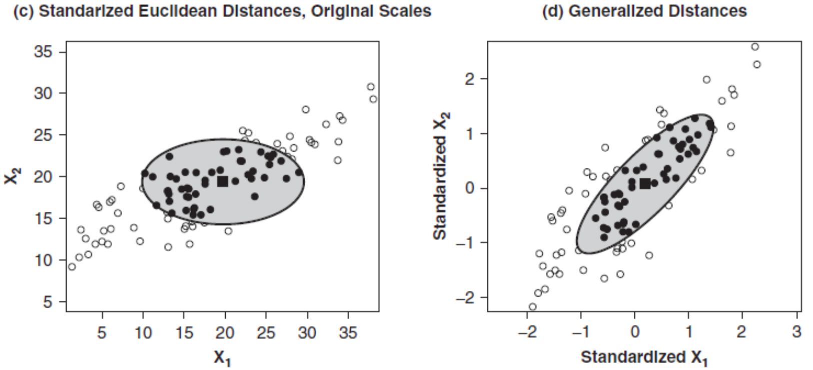
Multivariate Kernels. Illustration 1



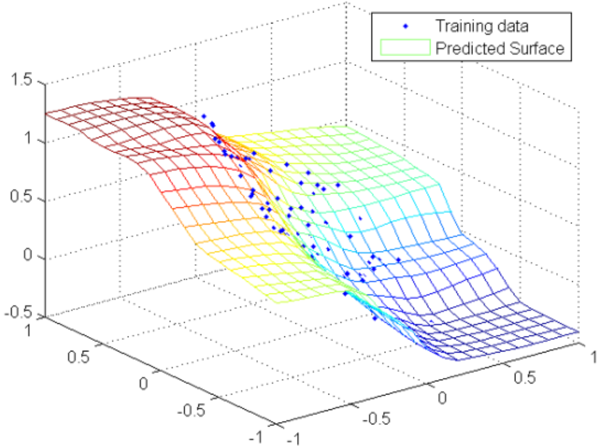
Squared point is a point of interest, grey area is the region of influence

Multivariate Kernels. Illustration 2

More sophisticated multivariate kernels can be constructed



Multiple Kernel Regression. Illustration



Multivariate kernel regression function with Gaussian kernel is a type of multivariate radial-basis function (RBF) regression

Unbiasedness of Multiple Kernel Estimator

Kernel regression estimator:

$$\hat{\varphi}(x) = \sum_{i=1}^{n} w_i(x) y_i$$

The multiple kernel estimator is a generalization of simple kernel (Nadaraya-Watson) estimator to the multidimensional case

The regression function $\varphi(x)$ should be locally constant in multidimensional area $\Delta_h(x)$, $\mathrm{M}[Y|x_i] = c, \ \forall x_i \in \Delta_h(x)$, for the estimate $\hat{\varphi}(x)$ to be unbiased

We need small bandwidths $h_1, ..., h_m$ of multivariate kernel to hold assumptions of approximately constant $\varphi(x)$, but a small number of data points will be used to average and obtain estimate $\tilde{\varphi}(x)$

Is it possible to use larger window sizes without the bias to be increased?

Locally Linear Multiple Regression

Idea: instead of assuming the regression function $\varphi(x)$ is approximately constant in a window, let's assume it is locally linear

Taylor expansion of $\varphi(x)$ in the neighborhood of x:

$$\varphi(x_i) \approx \varphi(x) + \frac{\partial \varphi(x)}{\partial x_1}(x_{1i} - x_1) + \ldots + \frac{\partial \varphi(x)}{\partial x_m}(x_{mi} - x_m)$$

Local OLS criterion:

$$E(x) = \sum_{x_i \in \Delta_h(x)} (y_i - \varphi(x_i))^2$$

$$= \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)\left[y_i - \left(\beta_0 + \sum_{j=1}^{m} \beta_j (x_{ji} - x_j)\right)\right]^2$$

where

$$\beta_0 = \varphi(x), \quad \beta_j = \frac{\partial \varphi(x)}{\partial x_j}, \quad j = \overline{1, M}$$

Multiple LOESS

Local OLS criterion in matrix form:

$$E(x) = (y - X\beta)^T V (y - X\beta) \to \min_{\beta}$$

where $y = (y_1, ..., y_n)^T$ is vector of responses, $\beta = (\beta_0, ..., \beta_m)^T$ is vector of parameters, X is design matrix:

$$X = \begin{pmatrix} 1 & x_{11} - x_1 & ... & x_{m1} - x_m \\ ... & ... & ... & ... \\ 1 & x_{1n} - x_1 & ... & x_{mn} - x_m \end{pmatrix} = \begin{pmatrix} 1 & x_1 - x \\ ... & ... \\ 1 & x_n - x \end{pmatrix}$$

and V is weight matrix:

$$K = diag\left[K\left(\frac{x - x_1}{h}\right), ..., K\left(\frac{x - x_n}{h}\right)\right]$$

The problem $E(x) \to \min_{\beta}$ is a weighted least squares problem

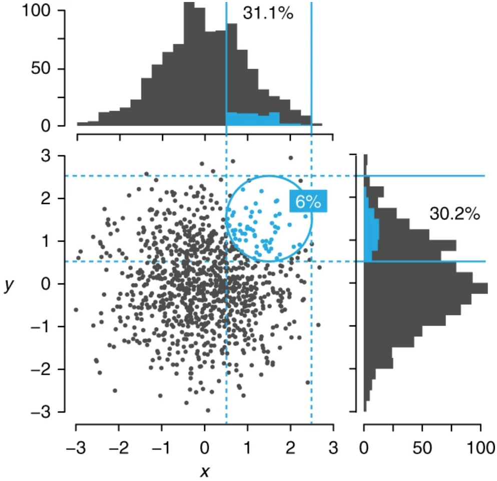
Problems of Non-parametric Multiple Regression

- **The curse of dimensionality**
  As the number of explanatory variables increases, the number of points in the neighbourhood $\Delta_h(x)$ tends to decline rapidly (the phenomenon of the empty spaces). The hypersphere of fixed radius becomes an insignificant volume in multidimensional space. The sample becomes very sparse

- **Difficulties of interpretation**
  Because non-parametric regression does not provide an equation relating the average response to the explanatory variables, we must display the response surface graphically. It can be complicated in multidimensional space

Non-parametric multiple regression (LOESS, KNN, etc.) is appropriate only for small number of explanatory variables ($m \lesssim 5$)

Data in Higher Dimensions



As m increases, the data become sparse in any given neighborhood of fixed radius