Kernel Density Estimation (KDE)

Study of the optimal bandwidth

Frank Bautista

September 30, 2020

Universidad Nacional de Colombia

Table of contents

Introduction

What is KDE?

Kernel density estimation - KDE

In statistics, the univariate kernel density estimation (KDE) is a non-parametric way to estimate the probability density function $f(x)$ of a random variable X

It is a fundamental data smoothing problem where inferences about the population are made, based on a finite data sample.

What is a kernel?

Kernel

In non-parametric statistics, a kernel is a weighting function used in non-parametric estimation techniques. Kernels are used in kernel density estimation to estimate random variables density functions $f(x)$.

In general any functions having the following assumptions can be used as a kernel:

1. $K(x) \geq 0$ and $\int_{\mathbb{R}} K(x)dx = 1$
2. Symmetric about the origin, $\int_{\mathbb{R}} xK(x)dx = 0$
3. Finite second moment $\mu_2(K) = \int_{\mathbb{R}} x^2 K(x)dx < \infty$

What is a kernel?

Kernel	$K(x;r)$	$\mathrm{R}(K)$	$\mu_2(K)$
Gaussian	$K(x;\infty)=\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right)1_{]-\infty,+\infty[}$	$1/\left(2\sqrt{\pi}\right)$	1
Epanechnikov	$K(x;2)=\frac{3}{4}\left(1-x^2\right)1_{(\|x\|\leq 1)}$	$3/5$	$1/5$
Uniform	$K(x;0)=\frac{1}{2}1_{(\|x\|\leq 1)}$	$1/2$	$1/3$
Triangular	$K(x;1)=(1-\|x\|)1_{(\|x\|\leq 1)}$	$2/3$	$1/6$
Triweight	$K(x;6)=\frac{35}{32}\left(1-x^2\right)^3 1_{(\|x\|\leq 1)}$	$350/429$	$1/9$
Tricube	$K(x;9)=\frac{70}{81}\left(1-\|x\|^3\right)^3 1_{(\|x\|\leq 1)}$	$175/247$	$35/243$
Biweight	$K(x;4)=\frac{15}{16}\left(1-x^2\right)^2 1_{(\|x\|\leq 1)}$	$5/7$	$1/7$
Cosine	$K(x;\infty)=\frac{\pi}{4}\cos\left(\frac{\pi}{2}x\right)1_{(\|x\|\leq 1)}$	$\pi^2/16$	$\left(-8+\pi^2\right)/\pi^2$

Figure 1: some important kernels [**?**]

Kernel density estimator

Kernel density estimator

Let $(X_1, X_2, ..., X_n)$ be a data sample, independent and identically distributed of a continuous random variable X, with density function $f(x)$.

$$\widehat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h\left(x - x_i\right) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right). \tag{1}$$

Here, K is the kernel, n is the number of data and h is a smother is a smoothing constant, also it is call **bandwidth**.

Bandwidth selection

Bandwidth

The bandwidth of the kernel is a free parameter which exhibits a strong influence on the resulting estimate. This parameter is very important that controls the degree of smoothing applied to the data.

Some considrations in the values of h:

- If $h \to 0$ then we will have over-fitting.
- If $h \to \infty$, we will have a density function completely smoothed.

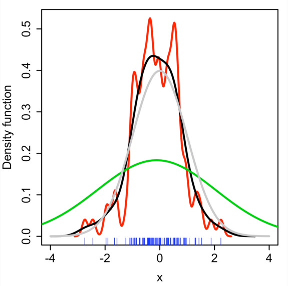

Figure 2: Different bandwidths for KDE (From: wikipedia.org/wiki/Kernel_density_estimation)

Bandwidth selection

The most common optimality criterion used to select this parameter is the expected risk function employing cross validation:

$$\mathrm{MISE}(h) = \mathrm{E}\left[\int \left(\hat{f}_h(x) - f(x)\right)^2 dx\right].$$

or the Asymptotic Mean Integrated Squared Error's Reduction Techniques:

$$\mathrm{AMISE}(h, r) = \frac{\mathrm{R}\left(K^{(r)}\right)}{nh^{2r+1}} + \frac{1}{4}h^4\mu_2^2(K)\mathrm{R}\left(f^{(r+2)}\right)$$

with optimal h as:

$$h^* = \left[\frac{(2r+1)\mathrm{R}\left(K^{(r)}\right)}{\mu_2^2(K)\mathrm{R}\left(f^{(r+2)}\right)}\right]^{1/(2r+5)} n^{-1/(2r+5)}$$

Bandwidth selection

Also is used the rule-of-thumb bandwidth estimator:

$$h = \left(\frac{4\hat{\sigma}^5}{3n}\right)^{\frac{1}{5}} \approx 1.06\hat{\sigma}n^{-1/5}.$$

Or Silverman's (1986) rule of thumb:

$$h = 0.9 \min\left(\hat{\sigma}, \frac{IQR}{1.34}\right) n^{-\frac{1}{5}}.$$

KDE and bw selection in `Python`

Kernel estimators in `Python`

There are several options available for computing kernel density estimates in Python. The selection of a estimator depends a lot on what the particular goals are. There are KDE implementations through SciPy Scikits stack:

1. `SciPy`: library `gaussian_kde`.
2. `statsmodels`: libraries `KDEUnivariate, KDEMultivariate`.
3. `Scikit-learn`: library `KernelDensity`

Each has advantages and disadvantages.

Kernel estimators in `Python`

	Bandwidth Selection	Available Kernels	Multi-dimension	Heterogeneous data	FFT-based computation	Tree-based computation
Scipy	Scott & Silverman	One (Gauss)	Yes	No	No	No
Statsmodels KDEUnivariate	Scott & Silverman	Seven	1D only	No	Yes	No
Statsmodels KDEMultivariate	normal reference cross-validation	Seven	Yes	Yes	No	No
Scikit-Learn	None built-in; Cross val. available	6 kernels x 12 metrics	Yes	No	No	Ball Tree or KD Tree

Figure 3: Comparison of KDE methods

(From https://jakevdp.github.io/blog/2013/12/01/kernel-density-estimation/)

Conclusion

Questions?

References i