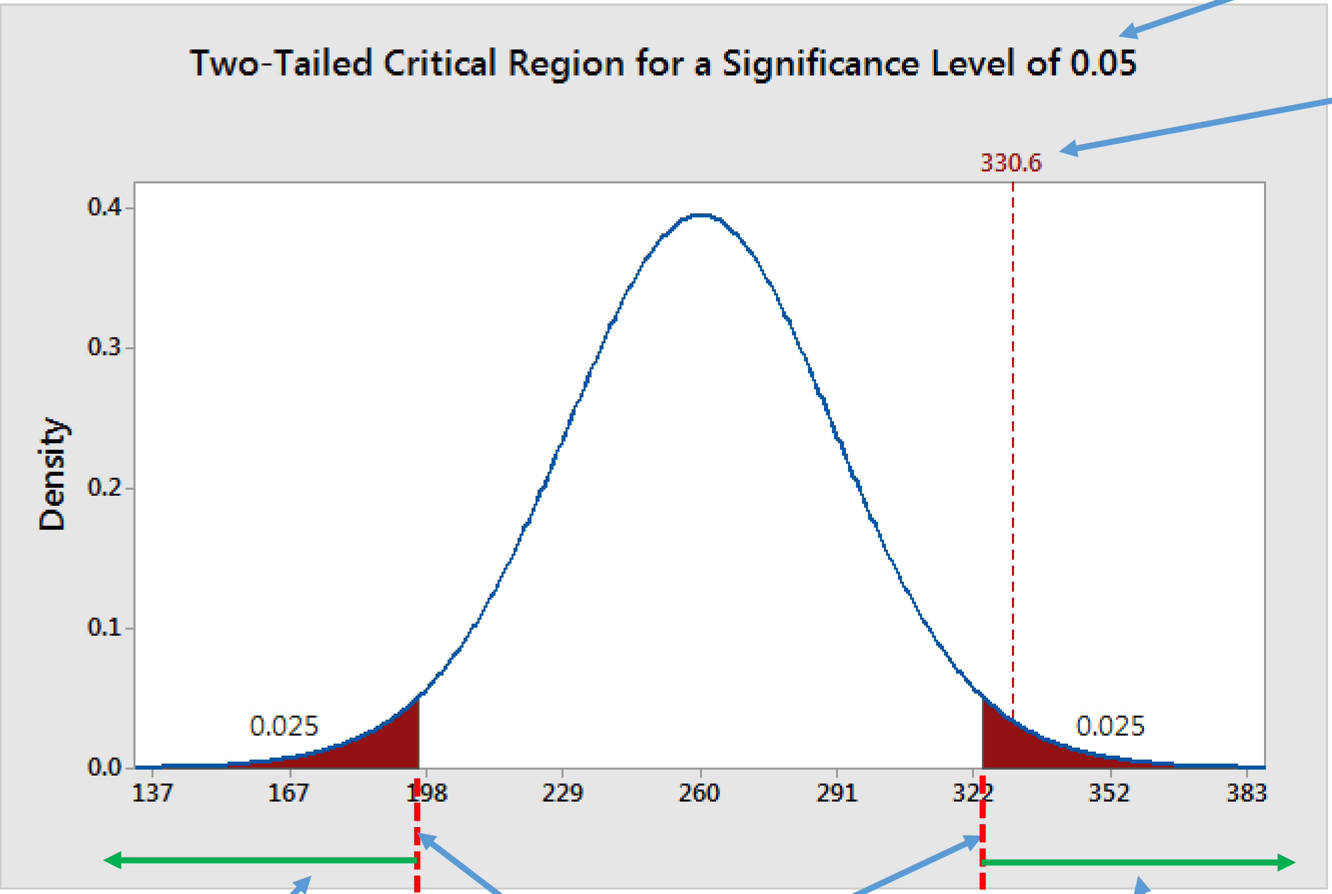
Topic No. 6

1. **Kernel Density Estimation (KDE)**

PDF



α : shaded area

t_{obs} value : calculated t value

critical region

t_{crt} value : from the table or from pdf(cdf)

critical region

PDF

A continuous probability density function $p(x)$ should satisfy the following properties

1. It is non-negative for all real x
2. Probability that x is between two points a and b is

$$P(a \leq x \leq b) = \int_a^b p(x)dx$$

3. The integration of the probability function is 1,

$$\int_{-\infty}^{\infty} p(x)dx = 1$$

Gaussian PDF $$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

PDF

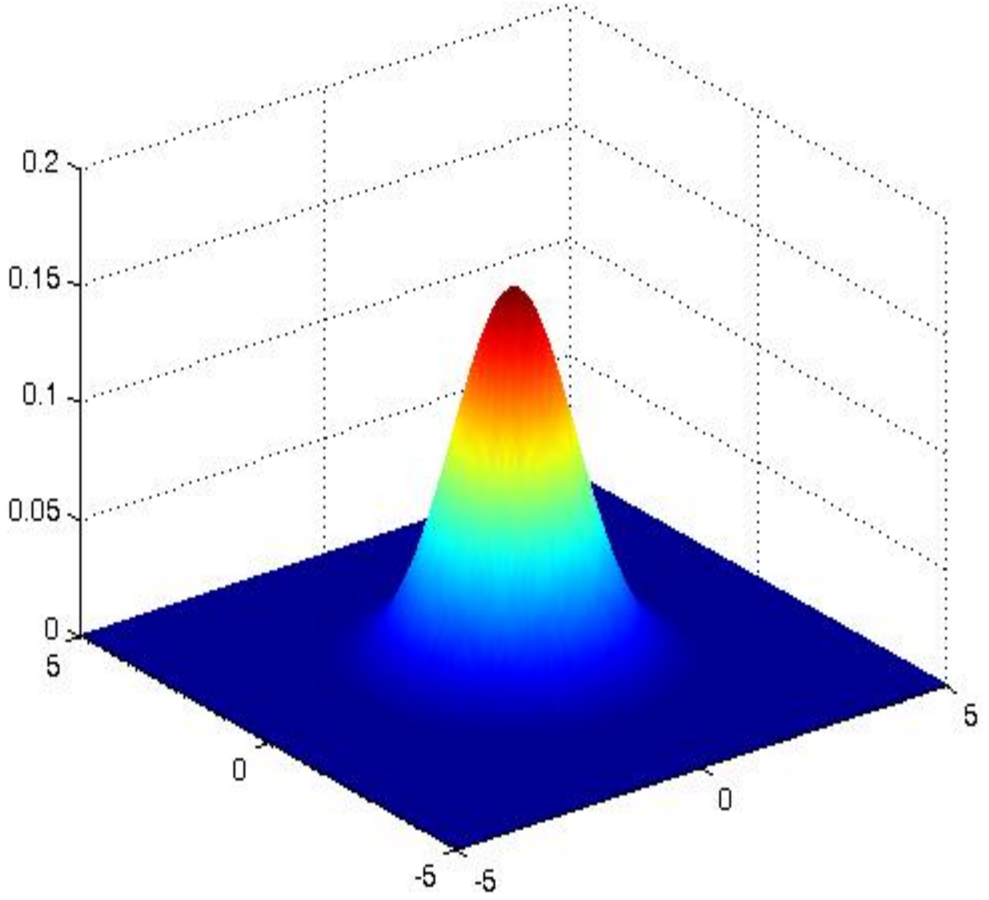
A continuous probability density function $p(x)$ in *multi-dimensional space* ($\boldsymbol{x}$ is a vector)

1. It is non-negative for all real $\boldsymbol{x}$
2. Probability that $\boldsymbol{x}$ is inside of a region $\mathcal{R}$

$$P = \int_{\mathcal{R}} p(\boldsymbol{x}) d\boldsymbol{x}$$

3. The integration of the probability function is 1,

$$\int_{-\infty}^{\infty} p(\boldsymbol{x}) d\boldsymbol{x} = 1$$



Multivariate Gaussian PDF

$$p(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^n |\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (x-\boldsymbol{\mu})}$$

Density Estimation

Density estimation is estimating the density function $p(\boldsymbol{x})$, so that we can output $p(\boldsymbol{x})$ for any new sample $\boldsymbol{x}$ for a set of n data samples $x_1,\dots, x_n$.

Remember that

$$P = \int_{\mathcal{R}} p(\boldsymbol{x})d\boldsymbol{x}$$

If we assume $\mathcal{R}$ small enough such that $p(\boldsymbol{x})$ does not vary much within $\mathcal{R}$, then

$$P = \int_{\mathcal{R}} p(\boldsymbol{x})d\boldsymbol{x} \approx p(\boldsymbol{x}) \int_{\mathcal{R}} d\boldsymbol{x} = p(\boldsymbol{x})V$$

where V is volume of $\mathcal{R}$.

Density Estimation

Suppose we n data samples $x_1,\ldots, x_n$ are independently sampled from the probability density function $p(\boldsymbol{x})$, and there are k samples out of n samples are in the region $\mathcal{R}$. Then, Probability that $\boldsymbol{x}$ is inside of region $\mathcal{R}$ is

$$P = \frac{k}{n}$$

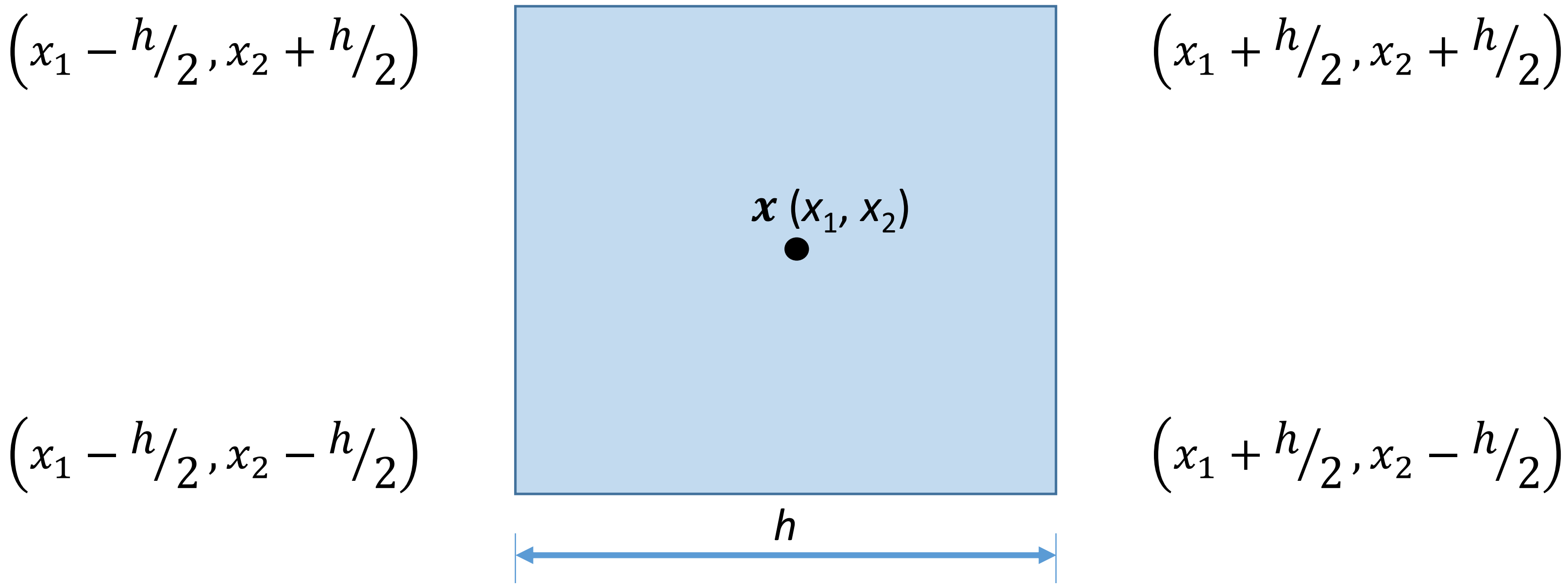
Now the probability density function $p(\boldsymbol{x})$ becomes

$$p(\boldsymbol{x}) = \frac{P}{V} = \frac{k/n}{V}$$

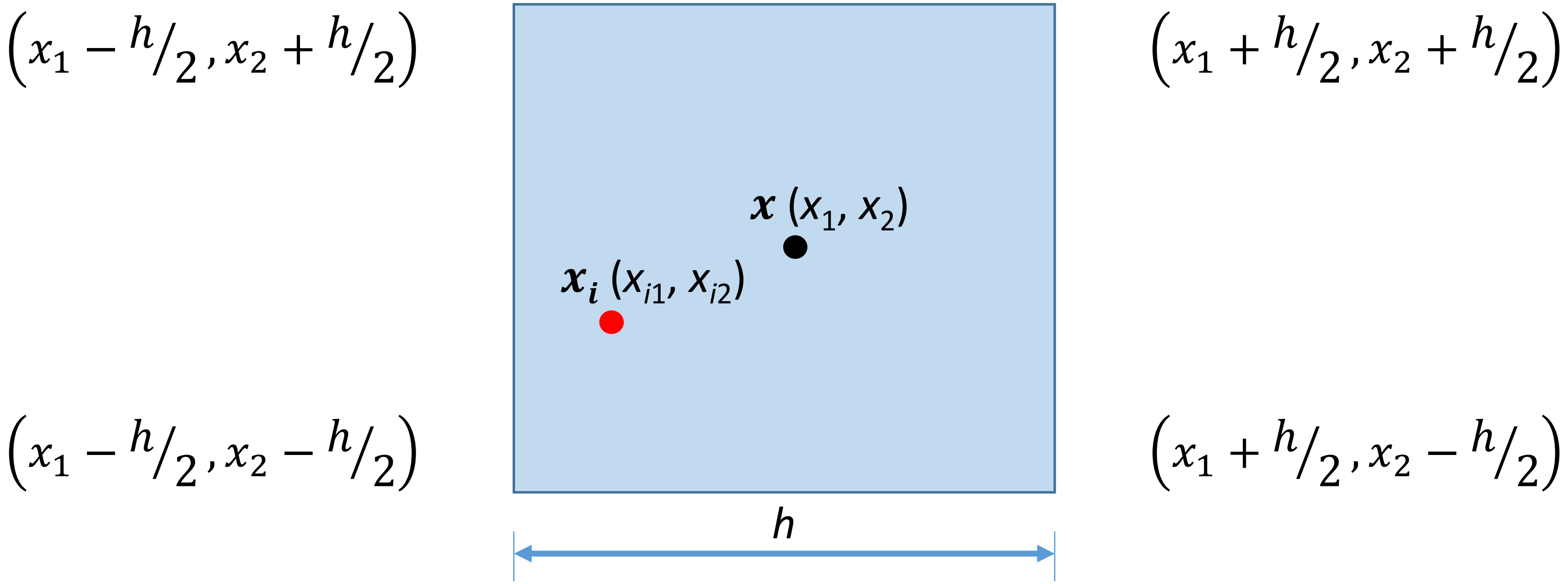
Density Estimation with Parzen window

Parzen window density estimation is a non-parametric estimation

Assume the region $\mathcal{R}$ is a hypercube centered at $\boldsymbol{x}$, and *h* be the length of the hypercube. Then, $V = h^d$, where *d* is the dimension of the hypercube.
For example of 2D, hypercube is a square with length of *h* and the volume (in this case, it is area) $V = h^2$. If the dimension is 3, $V = h^3$.



Density Estimation with Parzen window



Introduce a new function $\emptyset(x_i; \boldsymbol{x}, h)$

$$W(\boldsymbol{x}_i; \boldsymbol{x}, h) = \boldsymbol{W}\left(\frac{\boldsymbol{x_i} - \boldsymbol{x}}{h}\right) = \begin{cases} 1 & |x_{ik} - x_k| \leq \dfrac{h}{2} \\ 0 & otherwise \end{cases}$$

$where\ k = 1, \ldots, d$
$\qquad\quad i = 1, \ldots, n$

Density Estimation with Parzen window

$\boldsymbol{W}\left(\frac{\boldsymbol{x_i}-\boldsymbol{x}}{h}\right)$ indicates whether $\boldsymbol{x_i}$ is inside of the hypercube (in 2D, square), and h is bandwidth

Therefore, the total number of k samples out of n samples are in the region $\mathcal{R}$ is

$$k = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{W}\left(\frac{\boldsymbol{x_i}-\boldsymbol{x}}{h}\right)$$

The probability density function $p(\boldsymbol{x})$ is

$$p(\boldsymbol{x}) = \frac{P}{V} = \frac{k/n}{V} = \frac{1}{nV}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{W}\left(\frac{\boldsymbol{x_i}-\boldsymbol{x}}{h}\right)$$

In 2D example, the probability density function $p(\boldsymbol{x})$ is

$$p(\boldsymbol{x}) = \frac{1}{nh^2}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{W}\left(\frac{\boldsymbol{x_i}-\boldsymbol{x}}{h}\right)$$

Density Estimation with Parzen window

$\boldsymbol{W}\left(\frac{\boldsymbol{x_i}-\boldsymbol{x}}{h}\right)$ is called as a window function.

Generalization:

1. Window function can be a circle in 2D.
2. Instead of counting if a data point is inside of the window, we can apply varying weights depending the location of the data point.
   1) Data point is near the center of the window→ higher weight
   2) Data point is near the boundary of the window → lower weight

→ Gaussian

With 1D Gaussian, the probability density function $p(x)$ is

$$p(x) = \frac{1}{n}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-x)^2}{2\sigma^2}}$$

Density Estimation with Parzen window: Example.

Estimate PDF at x=3, using Gaussian function with $\sigma = 1$ for given 5 data points $x_1 = 6, x_2 = 2.5, x_3 = 1, x_4 = 3, x_5 = 2.$

$$p(\boldsymbol{x}) = \frac{1}{n}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i - x)^2}{2\sigma^2}}$$

1) For x_1, $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-x)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}} e^{-\frac{(6-3)^2}{2}} = 0.0044$
2) For x_2, $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-x)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}} e^{-\frac{(2.5-3)^2}{2}} = 0.3521$
3) For x_3, $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-x)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}} e^{-\frac{(1-3)^2}{2}} = 0.0540$
4) For x_4, $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-x)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}} e^{-\frac{(3-3)^2}{2}} = 0.3989$
5) For x_5, $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-x)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}} e^{-\frac{(2-3)^2}{2}} = 0.2420$

Density Estimation with Parzen window: Example.

Estimate PDF at x=3, using Gaussian function with $\sigma = 1$ for given 5 data points $x_1 = 6, x_2 = 2.5, x_3 = 1, x_4 = 3, x_5 = 2.$

So, PDF at x=3 is

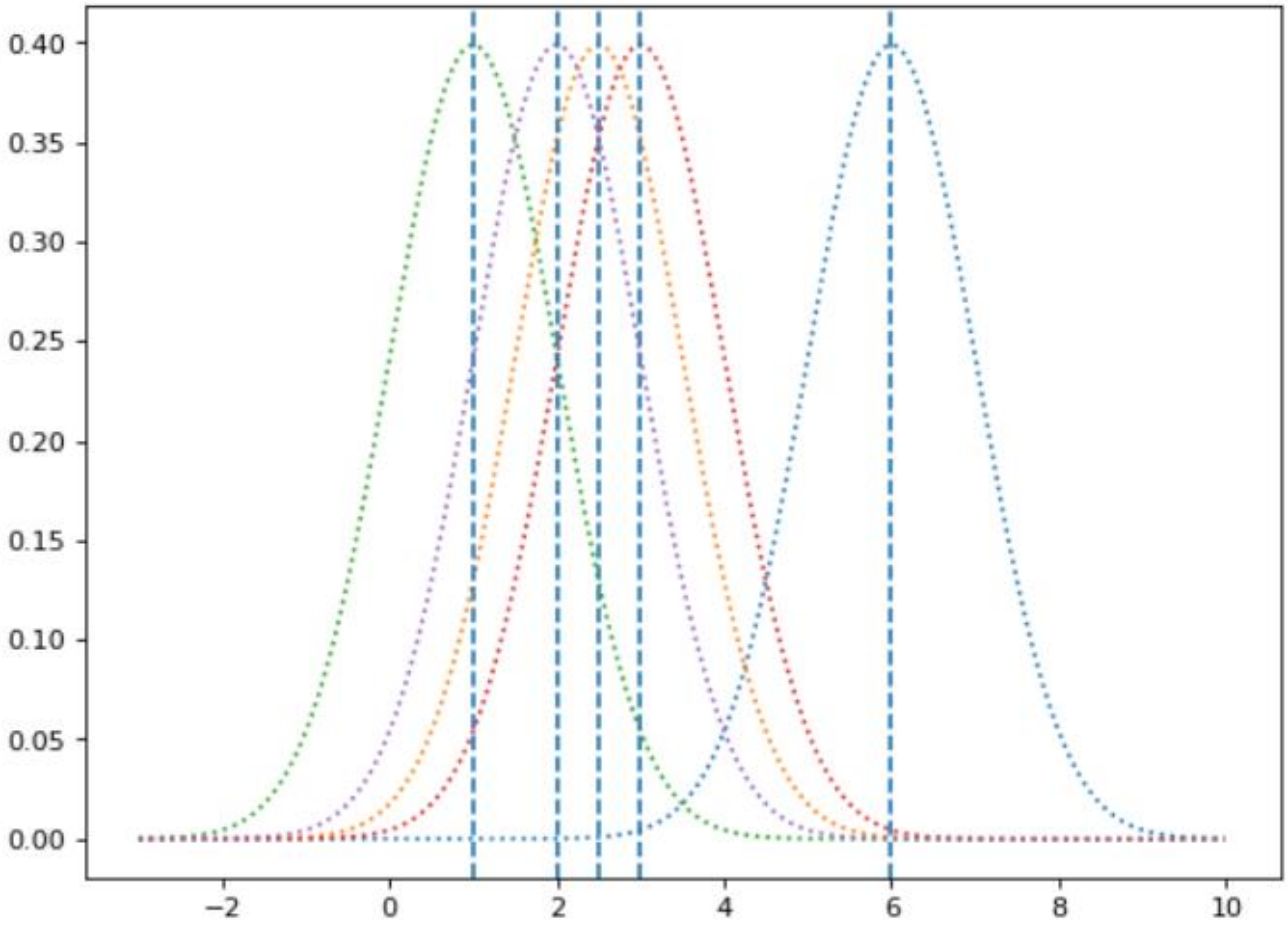
$$p(x = 3) = \frac{1}{5}\sum_{i=1}^{5} \frac{1}{\sqrt{2\pi}} e^{-\frac{(x_i - x)^2}{2}}$$

$$= \frac{(0.0044 + 0.3521 + 0.0540 + 0.3989 + 0.2420)}{5}$$

$$= 0.2103$$

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i - x)^2}{2\sigma^2}}$$

Graphical Illustration



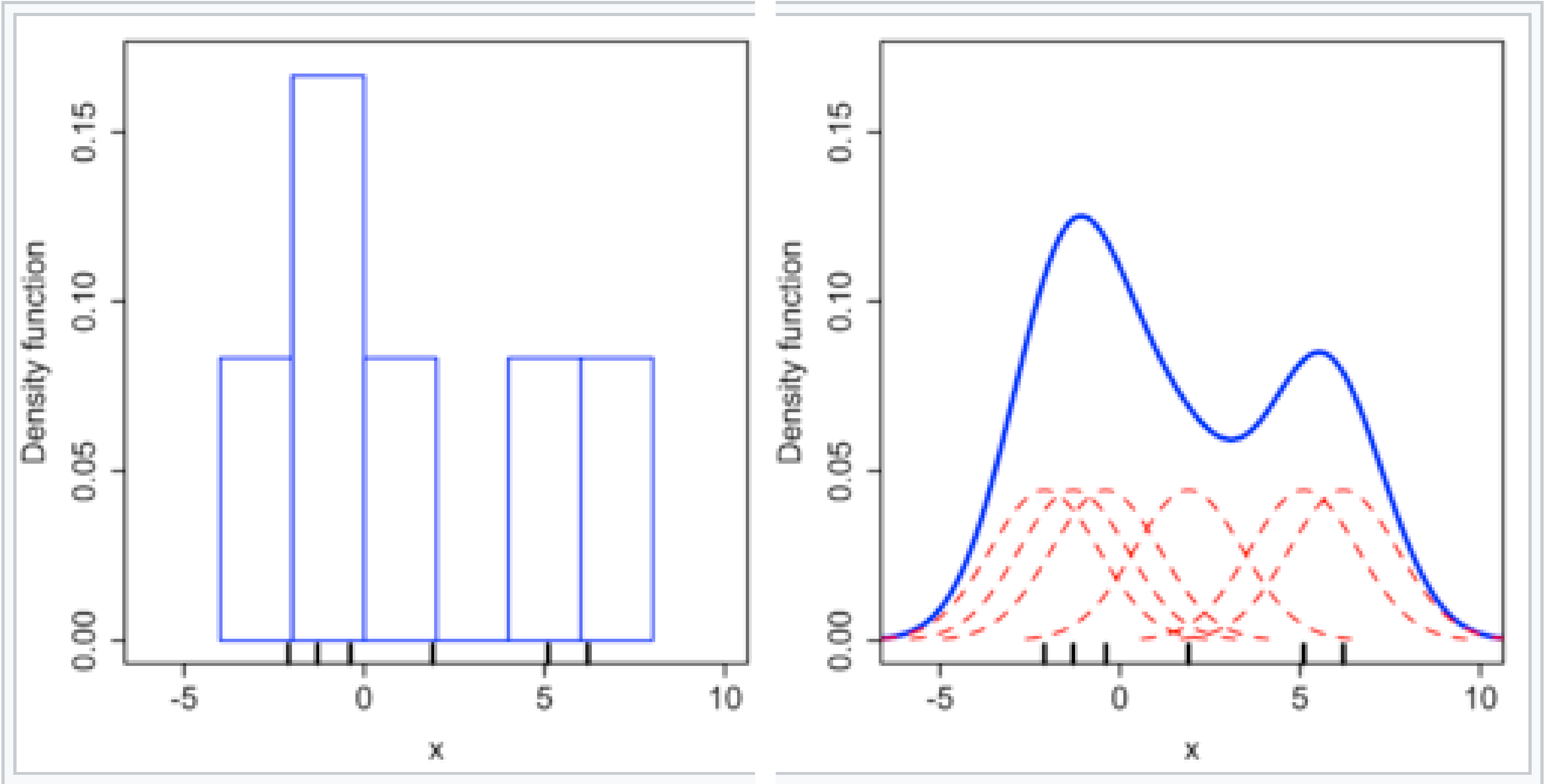
```
# example of Parzen window probability density estimation
from matplotlib import pyplot
from scipy.stats import norm
import numpy as np
from matplotlib.pyplot import figure

figure(figsize=(8, 6), dpi=80)
sample_std = 1
sample_points = [6, 2.5, 1, 3, 2]
pr = np.zeros(len(np.arange(-3, 10, 0.01)))
for me in sample_points:
    dist = norm(me, sample_std)
    values = [value for value in np.arange(-3, 10, 0.01)]
    probabilities = [dist.pdf(value) for value in values]
    pr += probabilities
    pyplot.plot(values, probabilities,'--')
    pyplot.axvline(x=me, linestyle='--')

pyplot.plot(values, pr/len(sample_points), linewidth=3)
pyplot.show()
```

Graphical Illustration

Sample	1	2	3	4	5	6
Value	-2.1	-1.3	-0.4	1.9	5.1	6.2



https://en.wikipedia.org/wiki/Kernel_density_estimation

Bandwidth selection

The bandwidth *h*

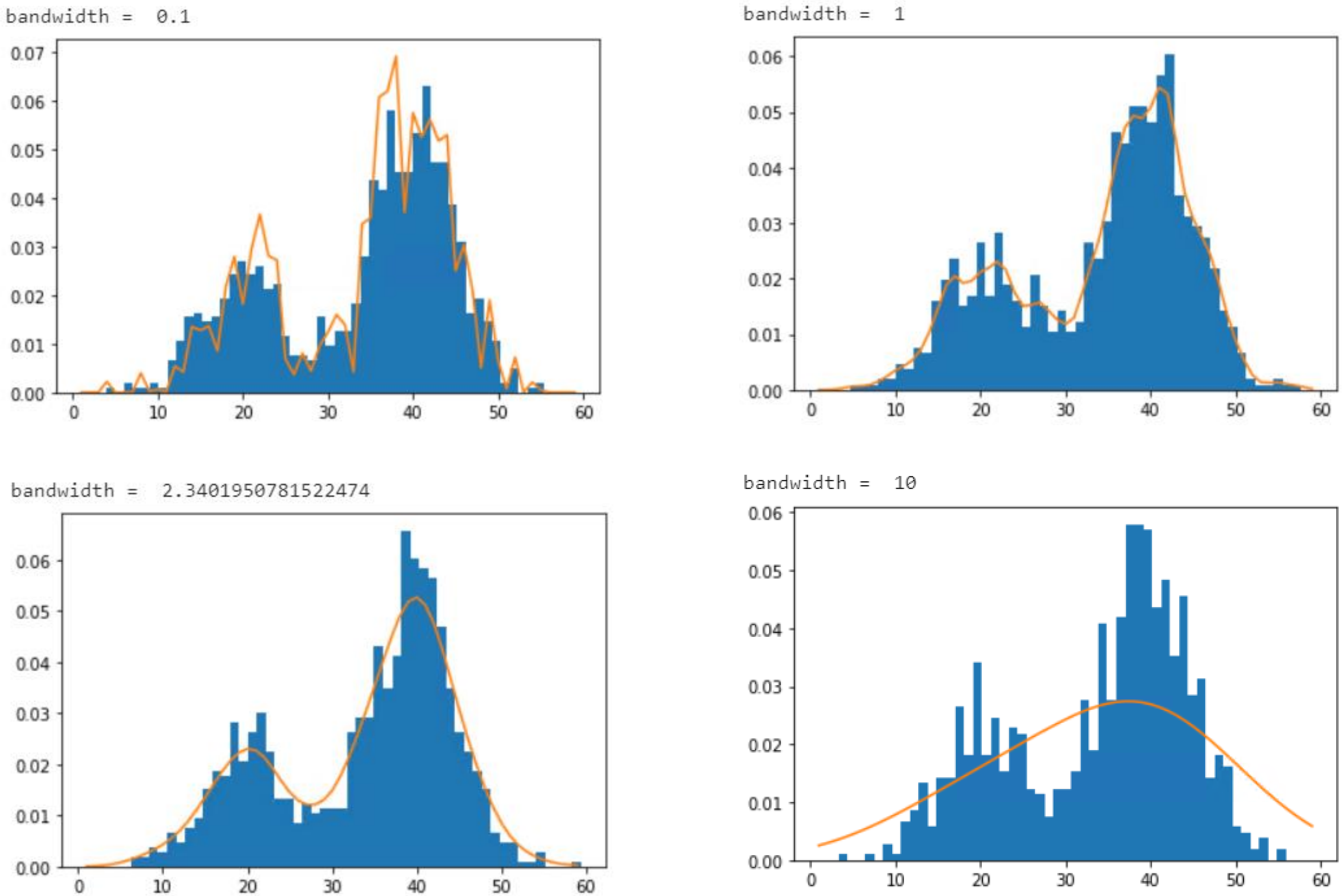
1. Optimization : not trivial problem due to unknown density function and its second derivatives
2. A rule-of-thumb bandwidth estimator

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

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

Better choice

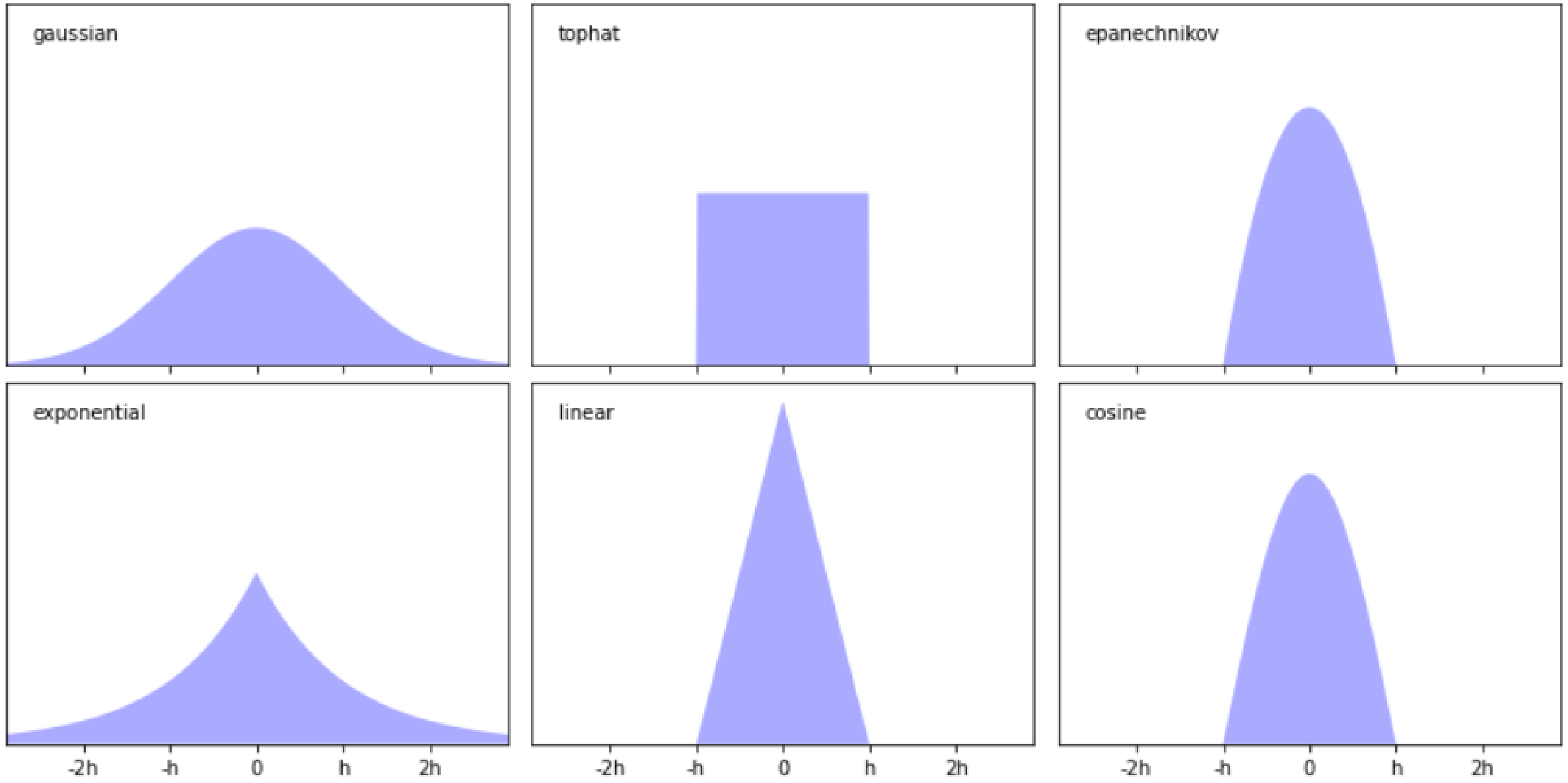
3. It should be chosen correctly
   1) Too small → less smoothing
   2) Too big → too much smoothing
4. Resources
   1) https://sebastianraschka.com/Articles/2014_kernel_density_est.html


bandwidth = 0.1
bandwidth = 1
bandwidth = 2.3401950781522474
bandwidth = 10

```
# example of kernel density estimation for a bimodal data sample
from matplotlib import pyplot
from numpy.random import normal
from numpy import hstack
from numpy import asarray
from numpy import exp
import numpy
import scipy
from sklearn.neighbors import KernelDensity
# generate a sample
sample1 = normal(loc=20, scale=5, size=300)
sample2 = normal(loc=40, scale=5, size=700)
sample = hstack((sample1, sample2))
# fit density
# bw=10
bw = 0.9*min(std(sample),scipy.stats.iqr(sample)/1.34)*(numpy.power(len(sample),(-1/5)))
print('bandwidth = ', bw)
model = KernelDensity(bandwidth=bw, kernel='gaussian')
sample = sample.reshape((len(sample), 1))
model.fit(sample)
# sample probabilities for a range of outcomes
values = asarray([value for value in range(1, 60)])
values = values.reshape((len(values), 1))
probabilities = model.score_samples(values)
probabilities = exp(probabilities)
# plot the histogram and pdf
pyplot.hist(sample, bins=50, density=True)
pyplot.plot(values[:], probabilities)
pyplot.show()
```

Various Kernels

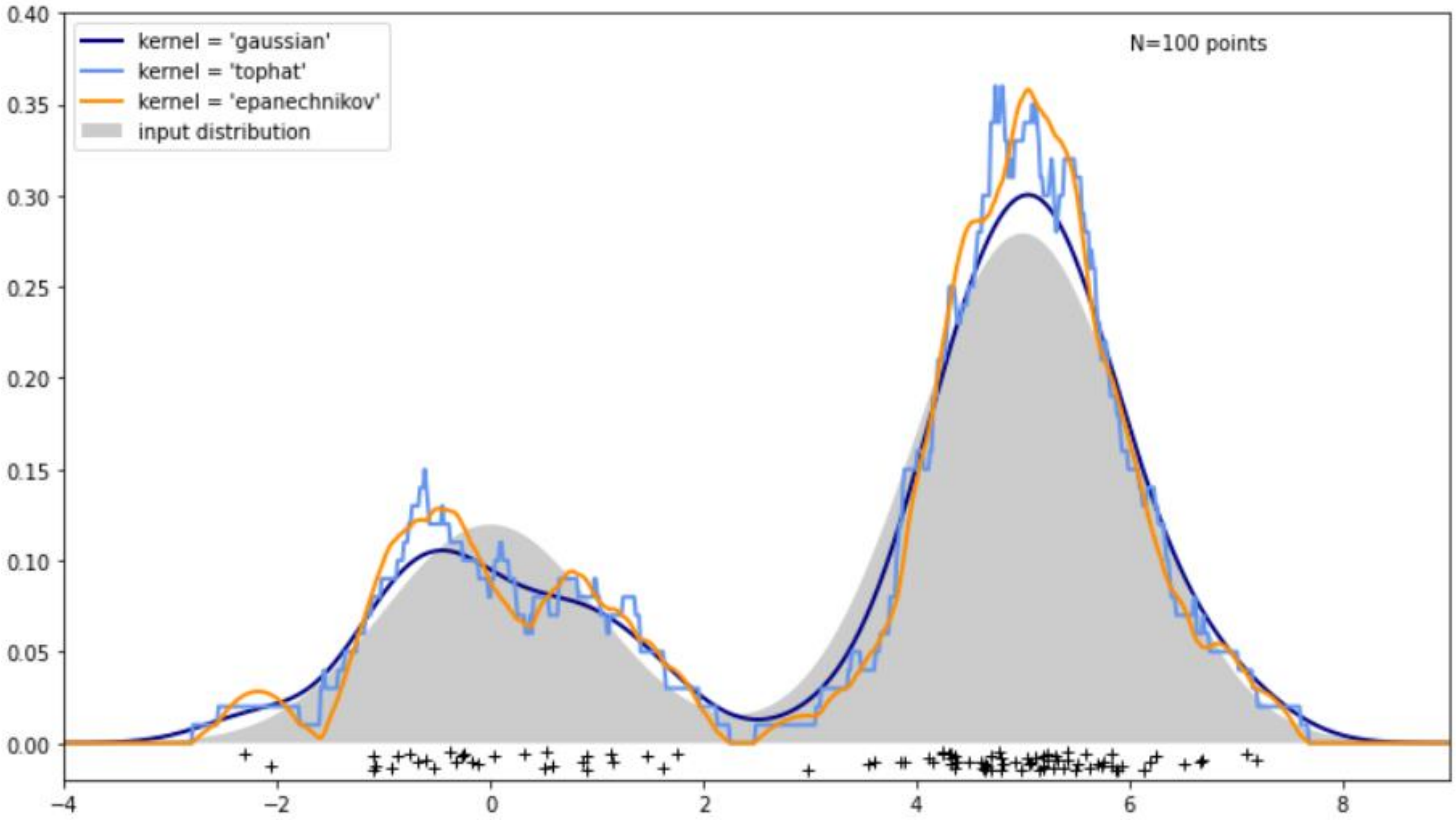
sklearn.neighbors.KernelDensity



https://scikit-learn.org/stable/auto_examples/neighbors/plot_kde_1d.html#sphx-glr-auto-examples-neighbors-plot-kde-1d-py

Various Kernels

sklearn.neighbors.KernelDensity



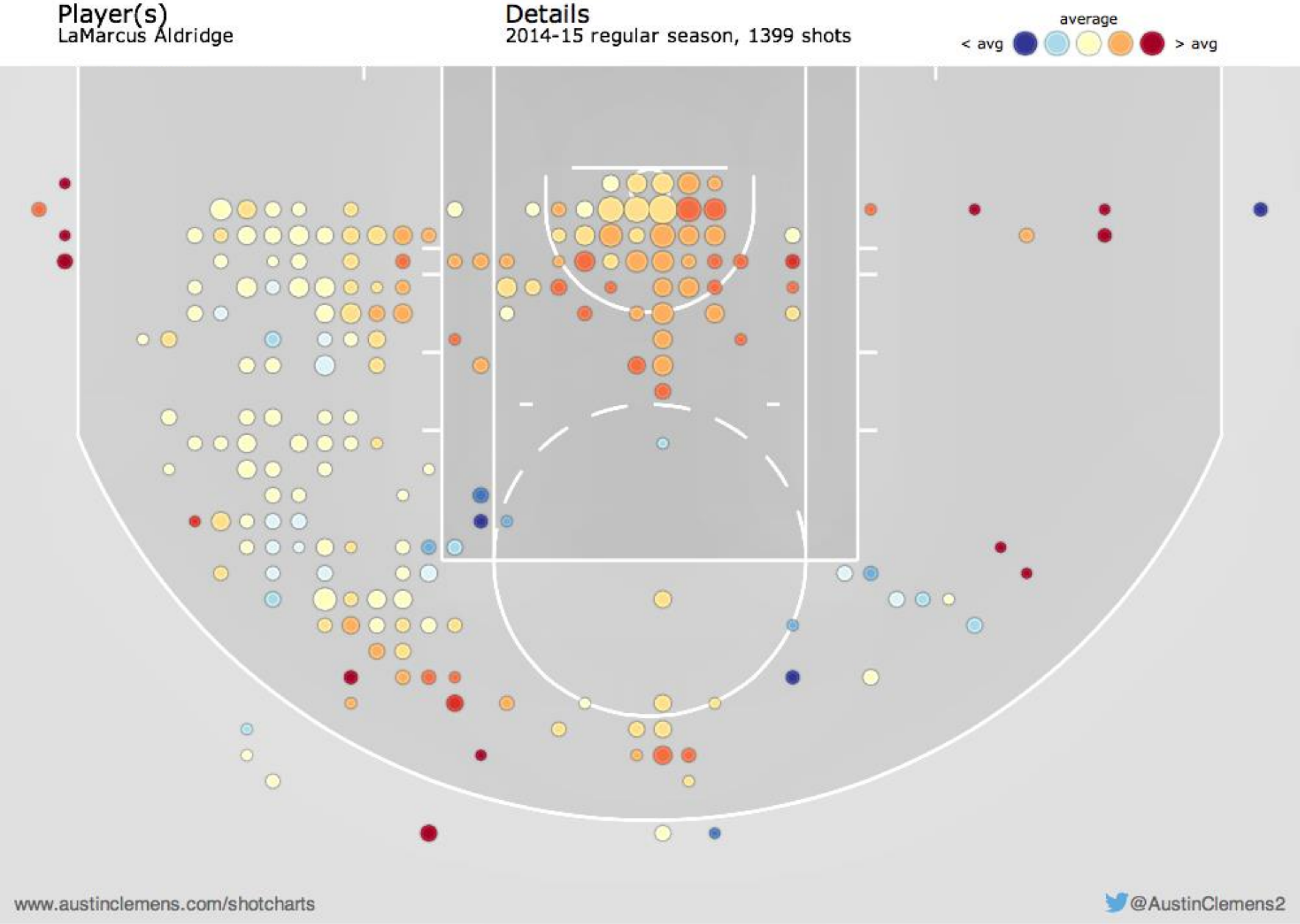
https://scikit-learn.org/stable/auto_examples/neighbors/plot_kde_1d.html#sphx-glr-auto-examples-neighbors-plot-kde-1d-py

Kernel Density Estimation

Motivating Example

First, acquire shooting data about each player (e.g., using code from Savvas Tjortjoglou's post).

Second, plot the shooting data.

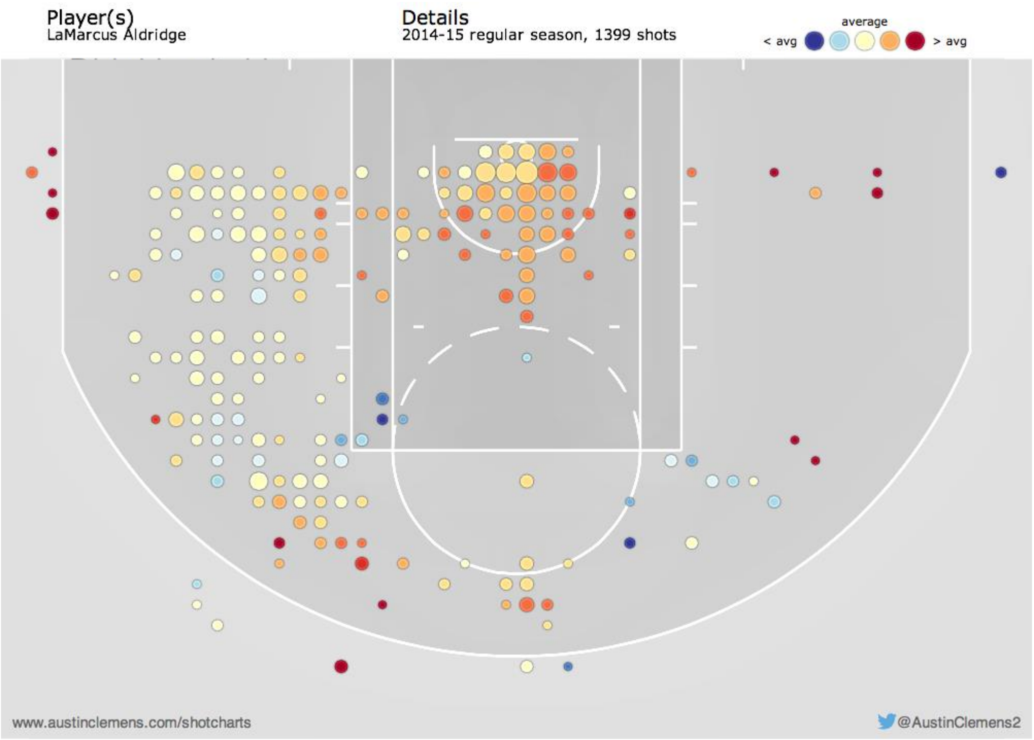


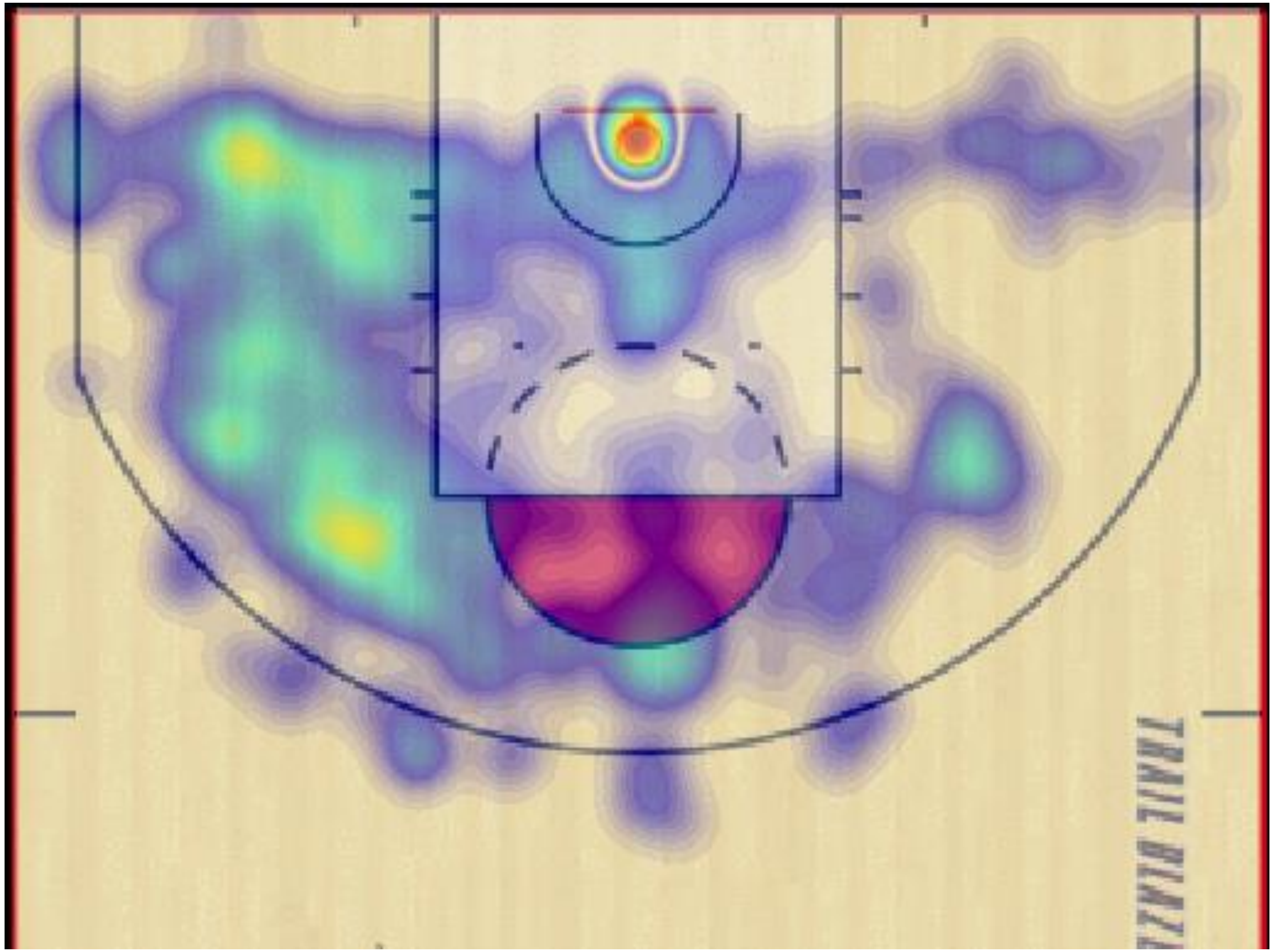
http://www.austinclemens.com/shotcharts/
http://savvastjortjoglou.com/nfl-bayesian-bootstrap.html

Kernel Density Estimation

Motivating Example – Is one visualization better than the other?

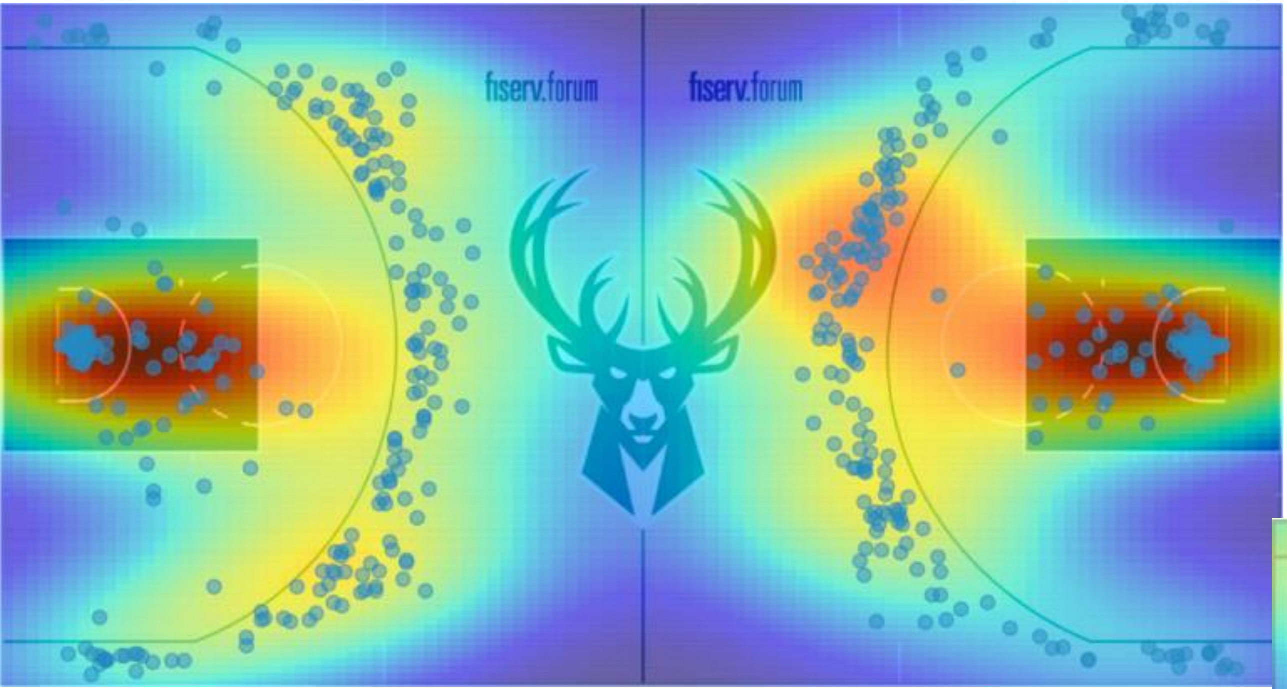
A kernel density estimator is a function that takes in the shot location and distributes variation about that location, giving it weight. A common function that spreads weight is the two-dimensional Gaussian distribution with mean centered at the shot location and a width, h, called a **bandwidth**.

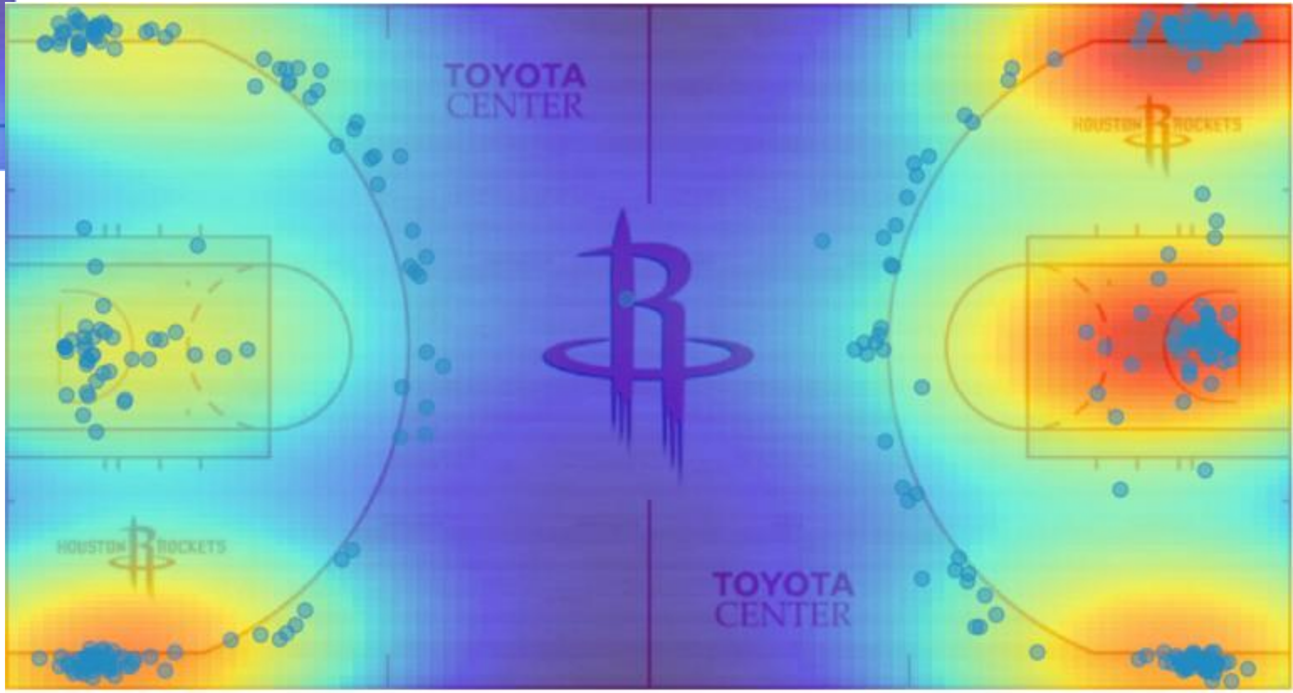




https://squared2020.com/2015/09/01/nba-shot-charts-via-kernel-density-estimation/

Kernel Density Estimation

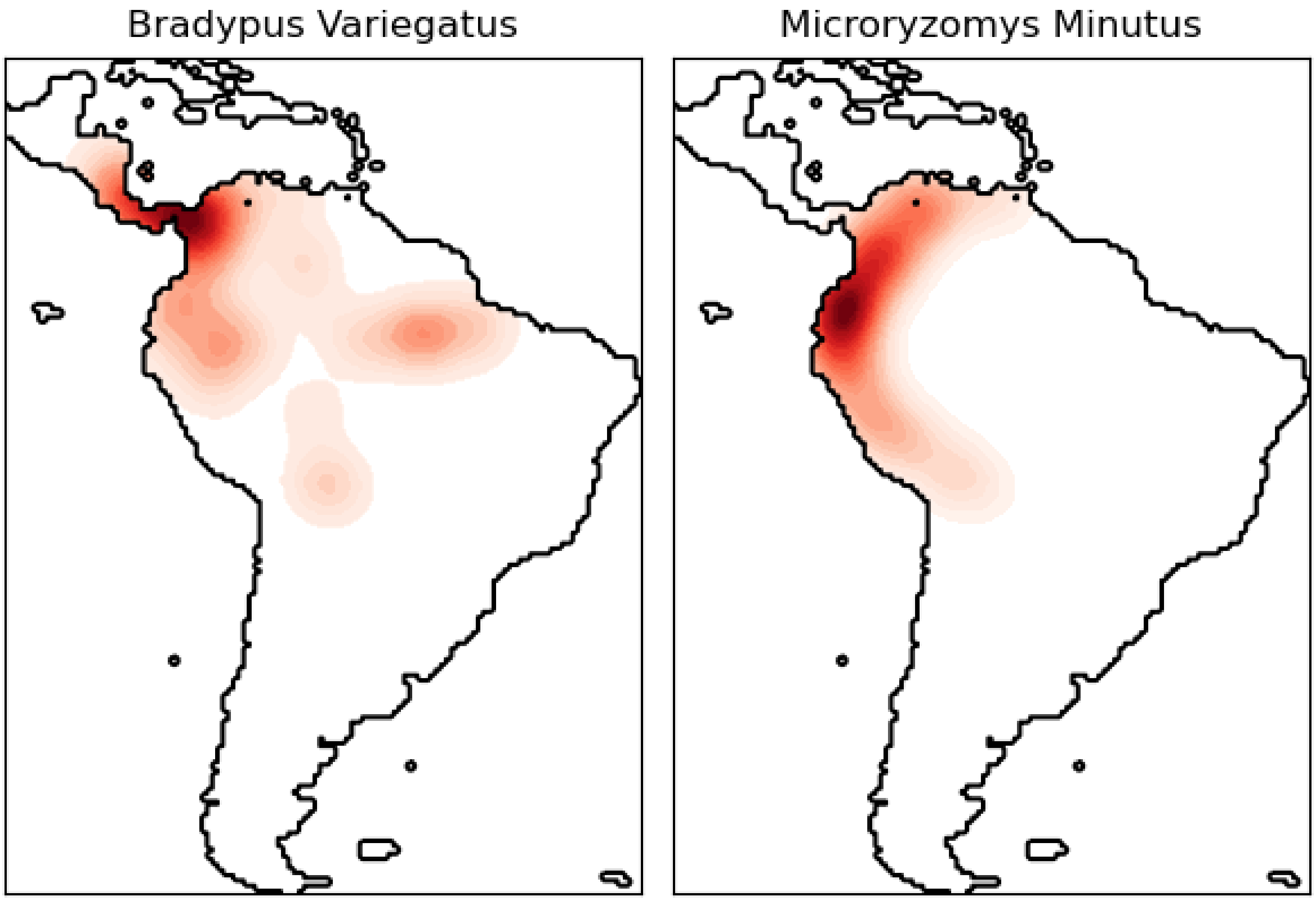




https://squared2020.com/2019/02/07/an-example-in-kullback-leibler-divergence/

Kernel Density Estimation

Motivating Example



https://scikit-learn.org/stable/auto_examples/neighbors/plot_species_kde.html#sphx-glr-auto-examples-neighbors-plot-species-kde-py