ESC 21FALL Week 1

~~다들 개강 화이팅~~🤧

SEP 9, 2021

SOOYON KIM 👩🏻‍💻 + KYUMIN LEE 👩🏻‍💻

Goal for today!

[ESL CHAPTER 6] KERNEL SMOOTHING METHOD

KERNEL ?

LOCAL REGRESSION ?

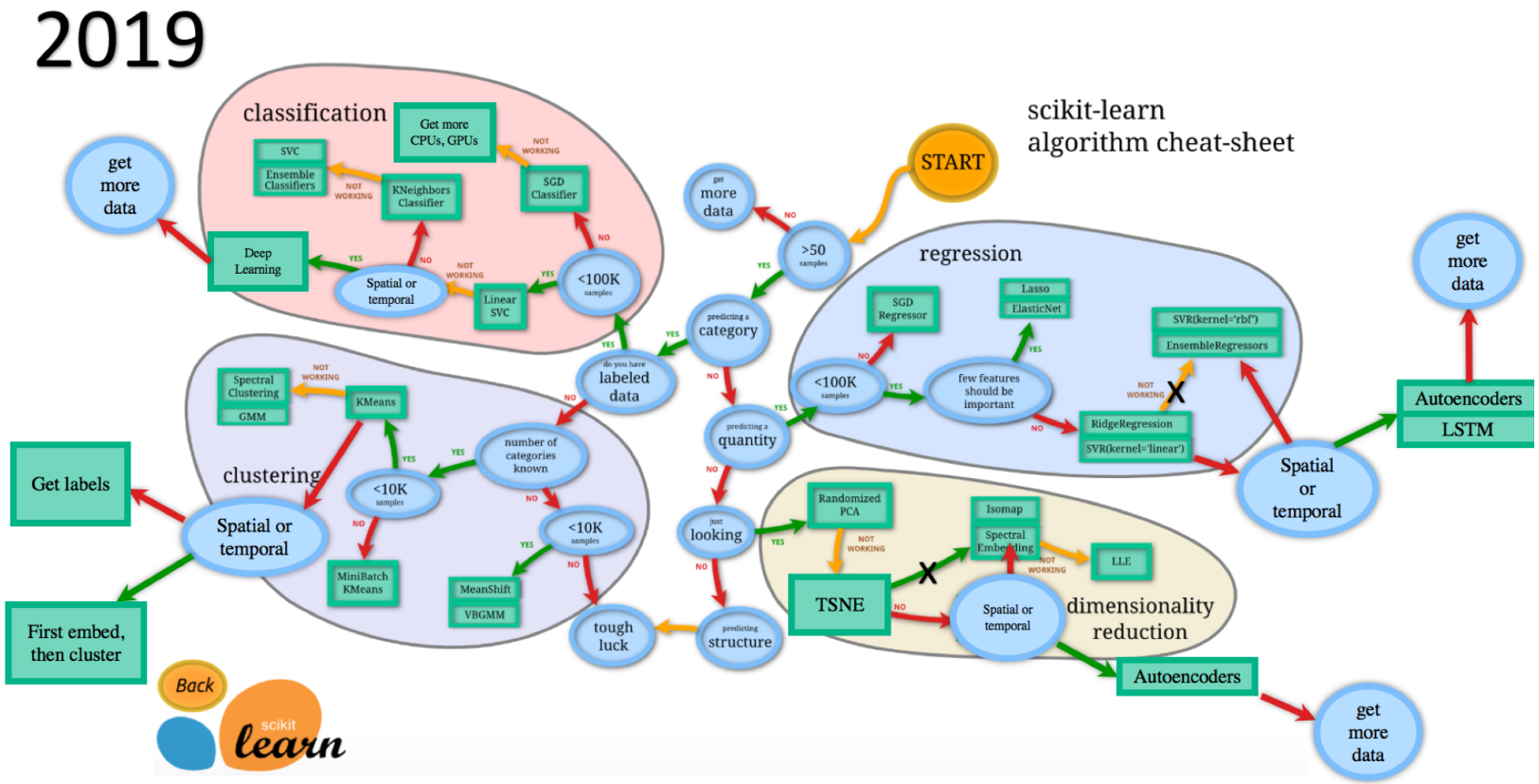
KERNEL DENSITY ESTIMATION ?

NAÏVE BAYES CLASSIFIER ?

BEFORE WE START…

Machine Learning ? Statistical Machine Learning ??

- Statistical learning theory : framework for machine learning drawing from the fields of statistics and functional analysis.

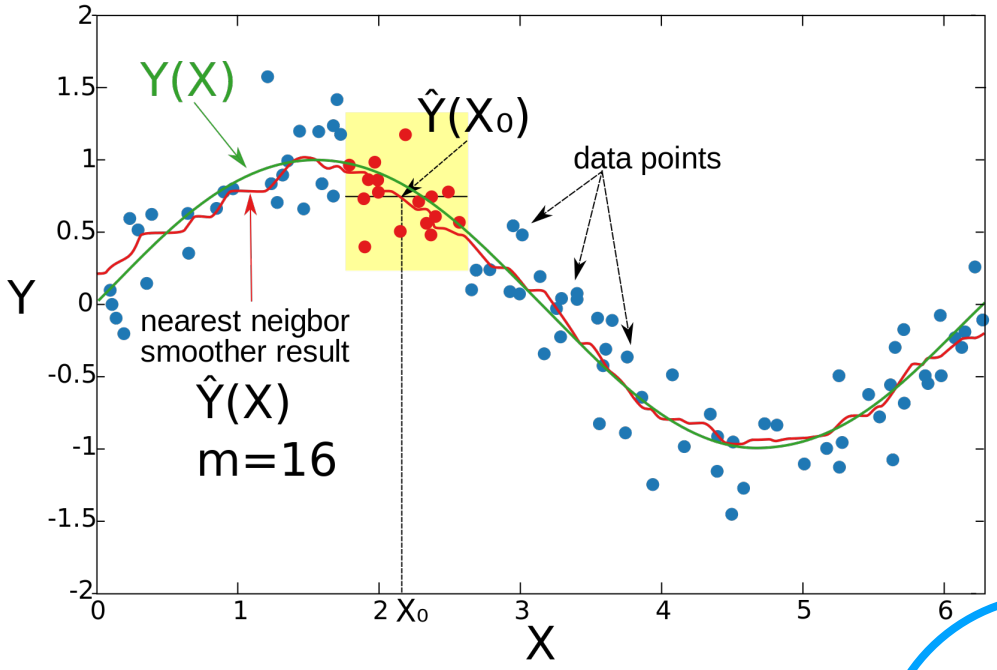


INTRO TO KERNEL METHODS

kernel ?

- kind of “filter”
- 주어진 data point들을 훑고 지나가면서

  주위의 data point들에 가중치를 부여한다.



kernel smoother ?

A **kernel smoother** is a statistical technique to estimate a real valued function $f: \mathbb{R}^p \rightarrow \mathbb{R}$ as the weighted average of neighboring observed data.

https://thepracticaldev.s3.amazonaws.com/i/2yiftvb3e8riv1ujo54g.gif

WHY DO WE NEED KERNEL?

local regression (kernel estimate) : Part 1

Actually, this is nothing but weighted-average
(with weird name called Nadaraya-Watson weighted average)

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{N} K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^{N} K_\lambda(x_0, x_i)}$$

우리가 공부해왔던 regression model이 β coefficient들을 사용한 parametric한 예측모델이었다면 kernel function을 사용하는 local regression model은 오로지 (x_i, y_i) data point들만을 사용해서 function **f** 를 추정한다! (No distributional assumption)

Parametric Model vs. Nonparametric Model? (Drawbacks of parametric model)

Modern social / natural phenomena are so complicated that it is often impossible to explain them using finite-dimensional probabilistic models.

WHY DO WE NEED KERNEL?

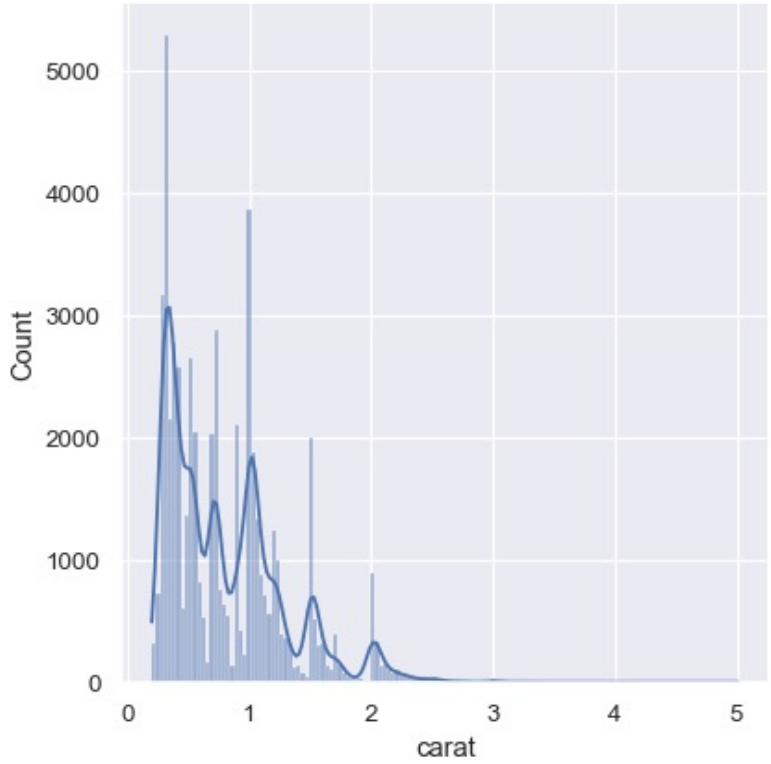
📊 **kernel density estimation (KDE)** : Part 2

- ➡️ probability density estimation
- ➡️ EDA (think of "kde" option in R or Python!)

우리가 데이터 / 모집단에 대해 어떠한 가정도 하지 않은 상태에서 그 분포 / 데이터에 대해 알고 싶을 때!

👩🏻‍💻 **kernel trick** : used broadly in ML!

- ➡️ kNN, SVM (next week!), Spline (Basis), etc.



Kernel Methods and Local Regression

(PART I)

ESL 6.1 / 6.2

KERNEL METHODS AND LOCAL REGRESSION

- providing estimates of regression function (conditional expectation E[Y|X])
in the nature of **local neighborhood** and of the class of regular functions **that are fitted locally**.

- 이 때 local neighborhood는 kernel function이란 것에 의해 정의된다.

$$K_\lambda(x_0, x)$$

Why do we need kernel?

- give weights to point x in a region around x_0.

ex) Gaussian Kernel

$$K_\lambda(x_0, x) = \frac{1}{\lambda}\exp[-\frac{||x-x_0||^2}{2\lambda}]$$

λ : variance of the Gaussian density (**controls the width of the neighborhood**)

➡️ kernel function assigns weights to points lying around x_0 (we say this query point!)
exponentially with their squared Euclidean distance from x_0.

There are many different kernels! (more to come)

According to ESL, ...

kernel smoothing method ?

➡️ regression techniques that has **flexibility** in estimating the regression function f(x) by **fitting different but simple model separately at each query point** x_0

- 정해진 parameter에 의존하는 모델 대신 각 점에서 fitting하겠다!

- 이 때 모든 data point를 다 쓰지 않고 현재 estimating 하고 있는 x_0 (a.k.a. target point)에서 가까운 데이터만 사용!

We call this **localization**

- localization은 **kernel function** 을 통해서 이루어진다.

➡️ assign a weight to neighbors based on its distance from query point.

➡️ Non-parametrical culture : model = data itself

6.1 ONE-DIMENSIONAL KERNEL SMOOTHERS

💡 Local Averaging

Setting : we don't know p(x, y)... We only have a sample

Problem : How to estimate the conditional expectation $f(x) = E(y|x)$?

Natural idea : Local averaging

- kind of relaxed knn (그냥 가까이에 있는 것들만 사용해서 평균내겠다)

$$\hat{f}(x) = mean(y_i \mid |x - x_i| \leq h)$$

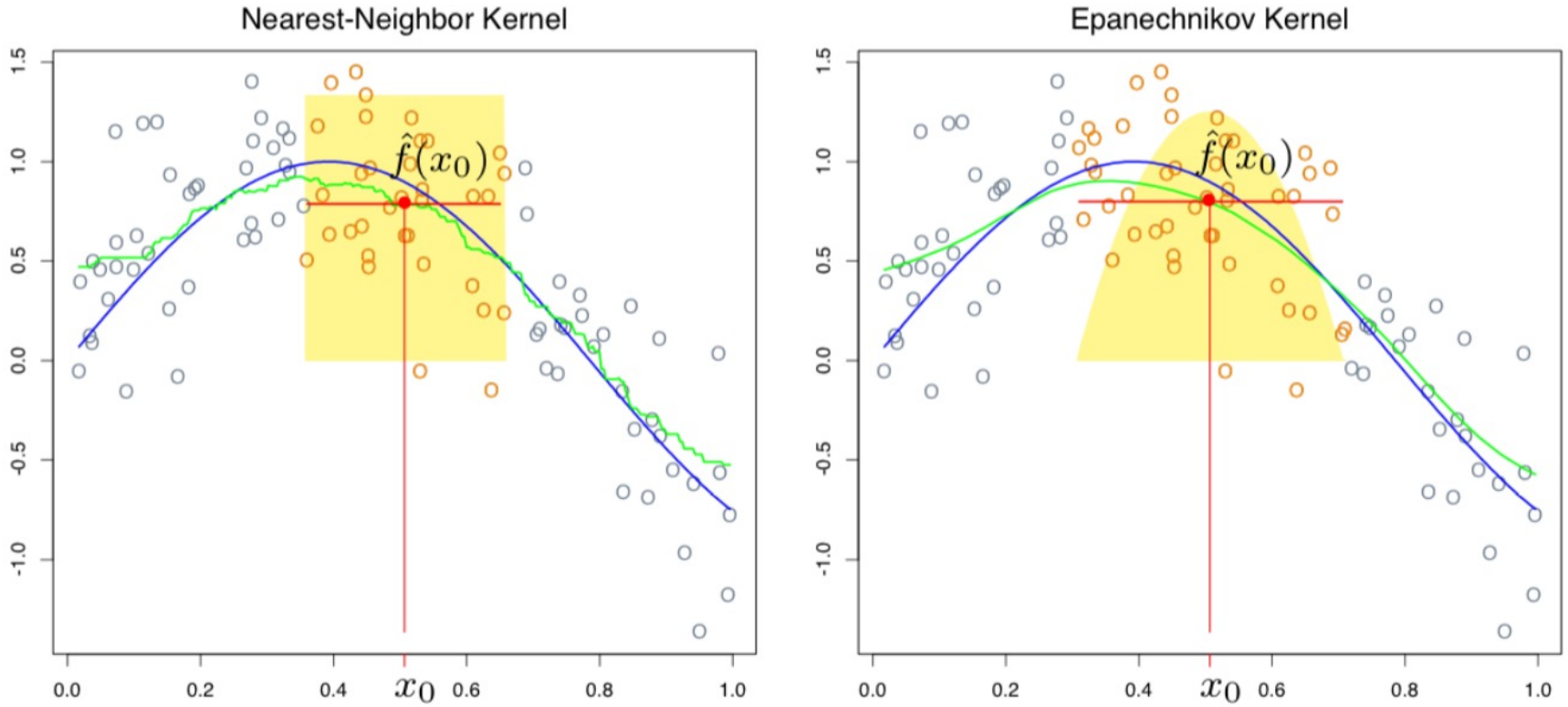
(bandwidth : choice of h controls degree of smoothness)

Can we do better ?

➡️ Give more weight to training observation x_i close to query x

$$\Phi_\sigma(x) = \text{Gaussian density with mean 0 and sd } \sigma$$

$$\hat{f}(x) = \frac{\sum_i y_i \Phi_\sigma(x - x_i)}{\sum_i \Phi_\sigma(x - x_i)}$$



⬆️ data points generated from blue true density

- red circles : data contributed for fitting $\hat{f}(x_0)$
- green line : smoothed line
- yellow box : kernel / filter / weight

왼쪽은 discrete하고 울퉁불퉁하다.. 안예쁨!

"We want to assign weights that die off smoothly with distrance from the target point"

📍 ESL 책에서는 nearest-neighbor 방법을 raw moving average라고 하고,
kernel based method를 smoothly varying locally weighted average라고 표현함

KERNEL FUNCTION

General form

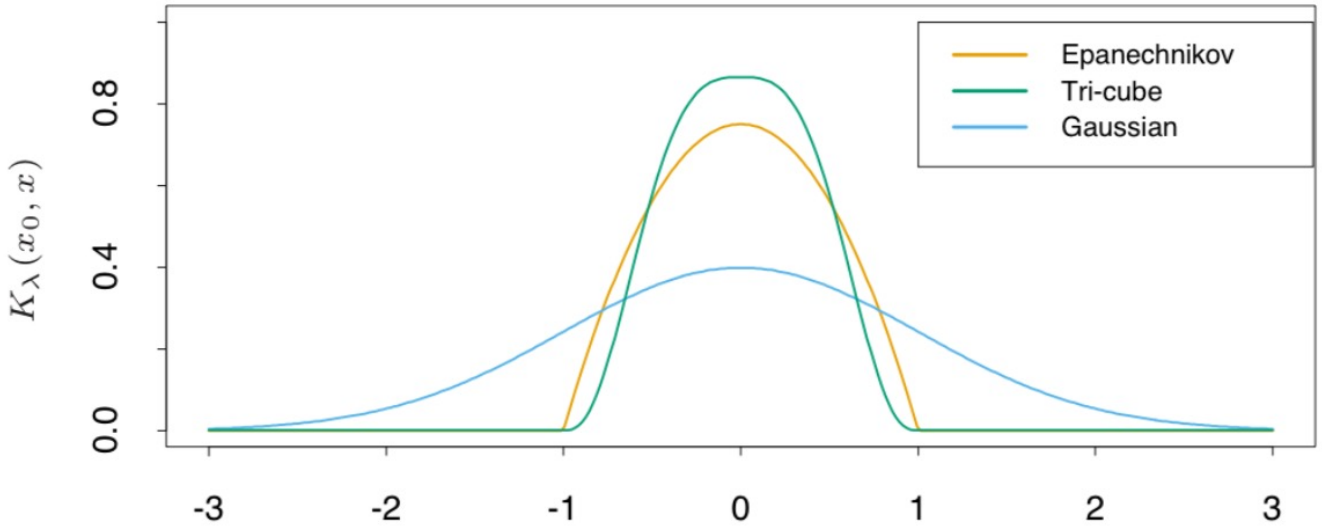
- h : width function (λ : smoothing parameter)

  ➡️ larger λ Implies lower variance (average over more obs) but higher bias

- D : distance metric $$K_\lambda(x_0, x) = D\left(\frac{|x - x_0|}{h_\lambda(x_0)}\right)$$

Choice of kernel function

1. $K(x)$ is symmetric
2. $\int K(x)dx = 1$: not necessary but to guarantee that the KDE is pdf.
3. $\lim_{x\to-\infty} K(x) = \lim_{x\to+\infty} K(x) = 0$

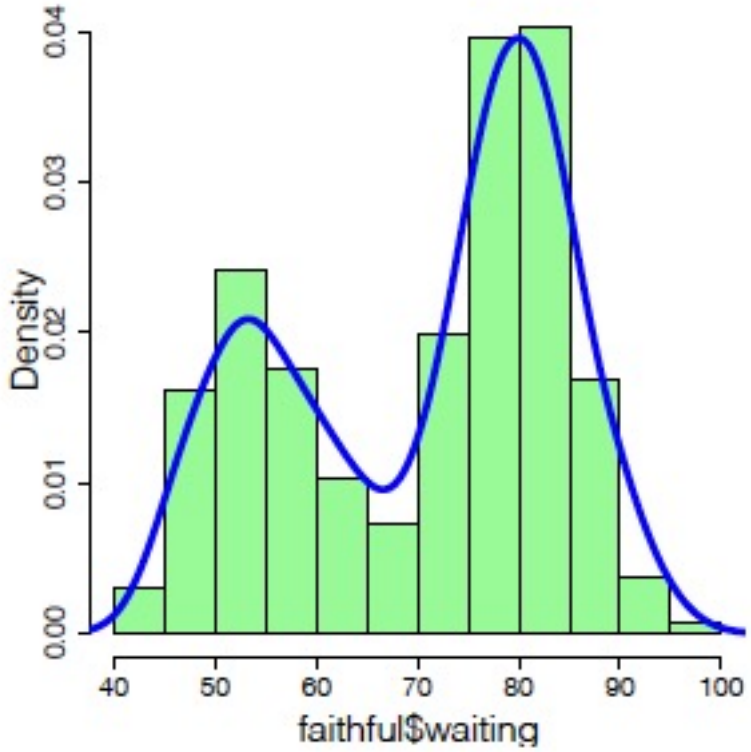


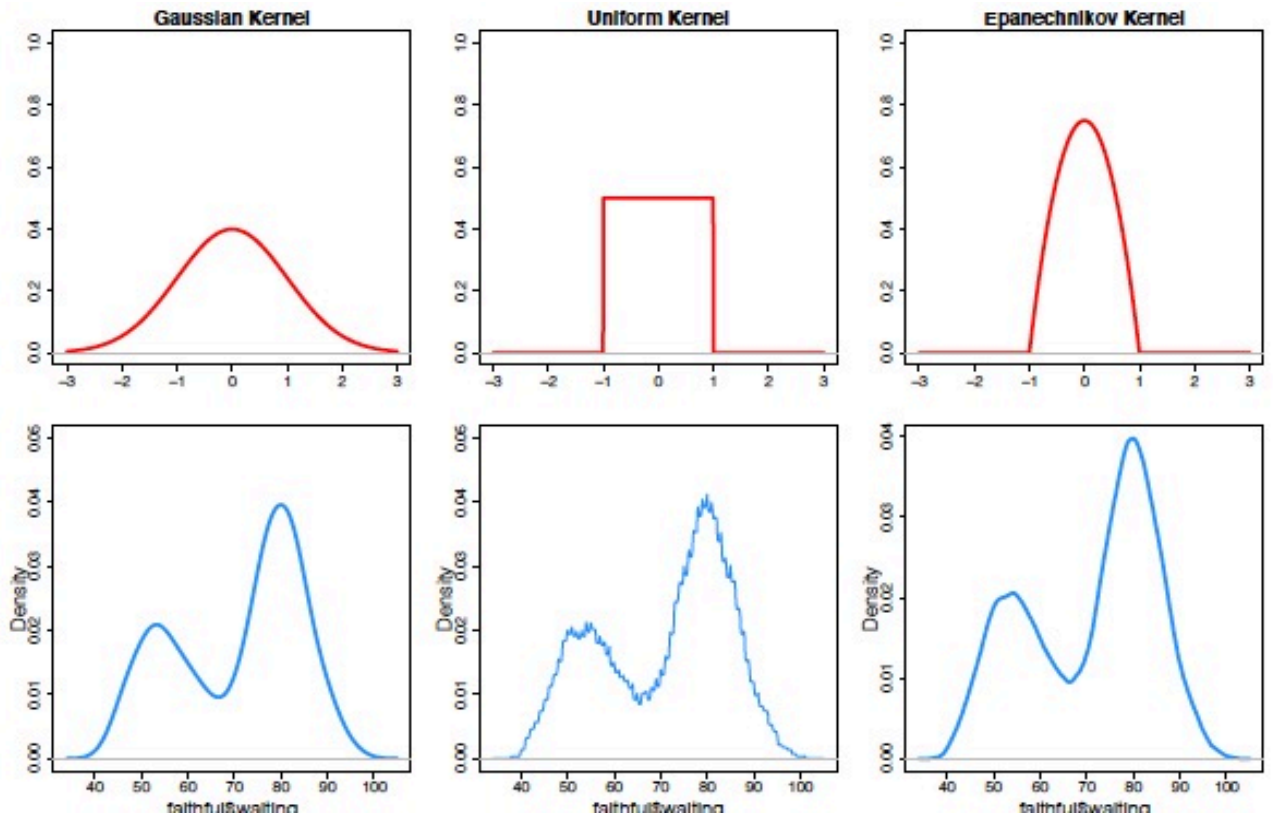
COMPARING DIFFERENT KERNELS

$$\text{Gaussian} \quad K(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}},$$

$$\text{Uniform} \quad K(x) = \frac{1}{2} I(-1 \le x \le 1),$$

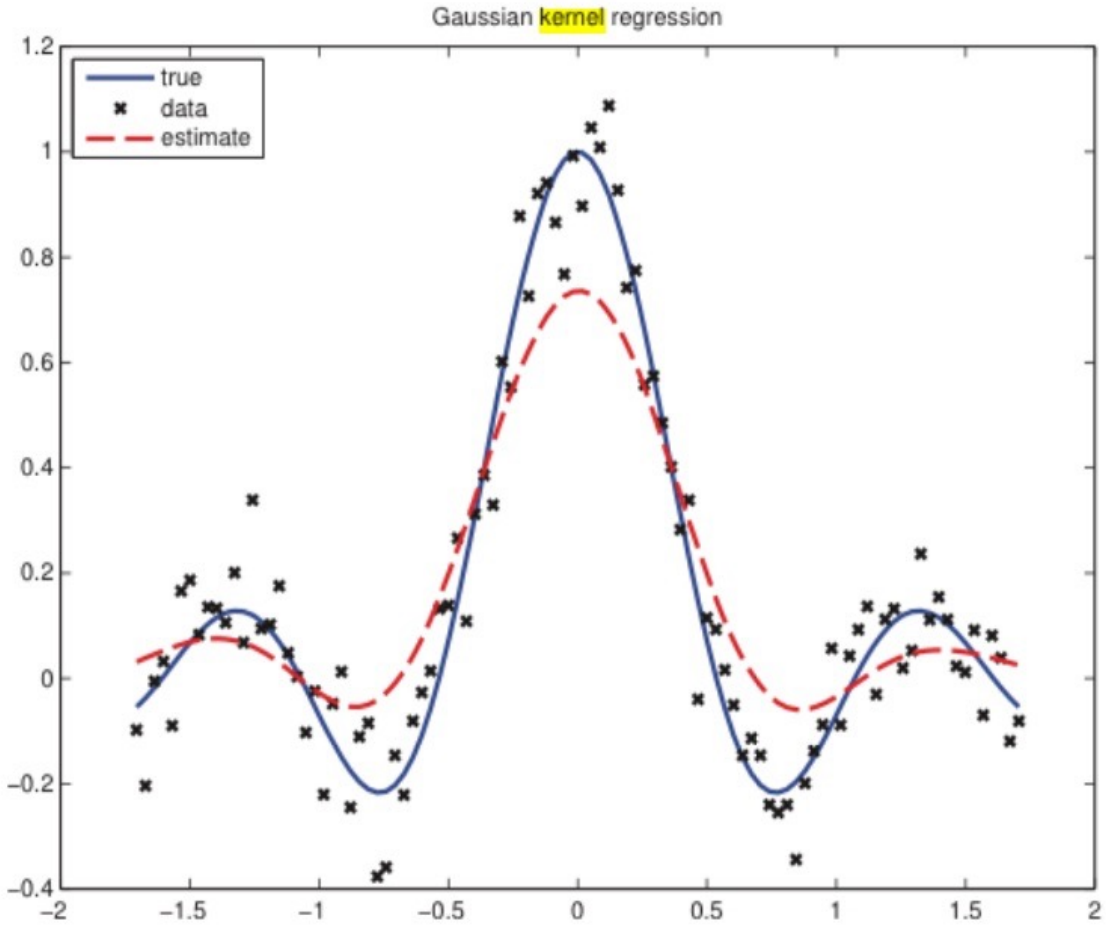
$$\text{Epanechnikov} \quad K(x) = \frac{3}{4} \cdot \max\{1 - x^2, 0\}.$$



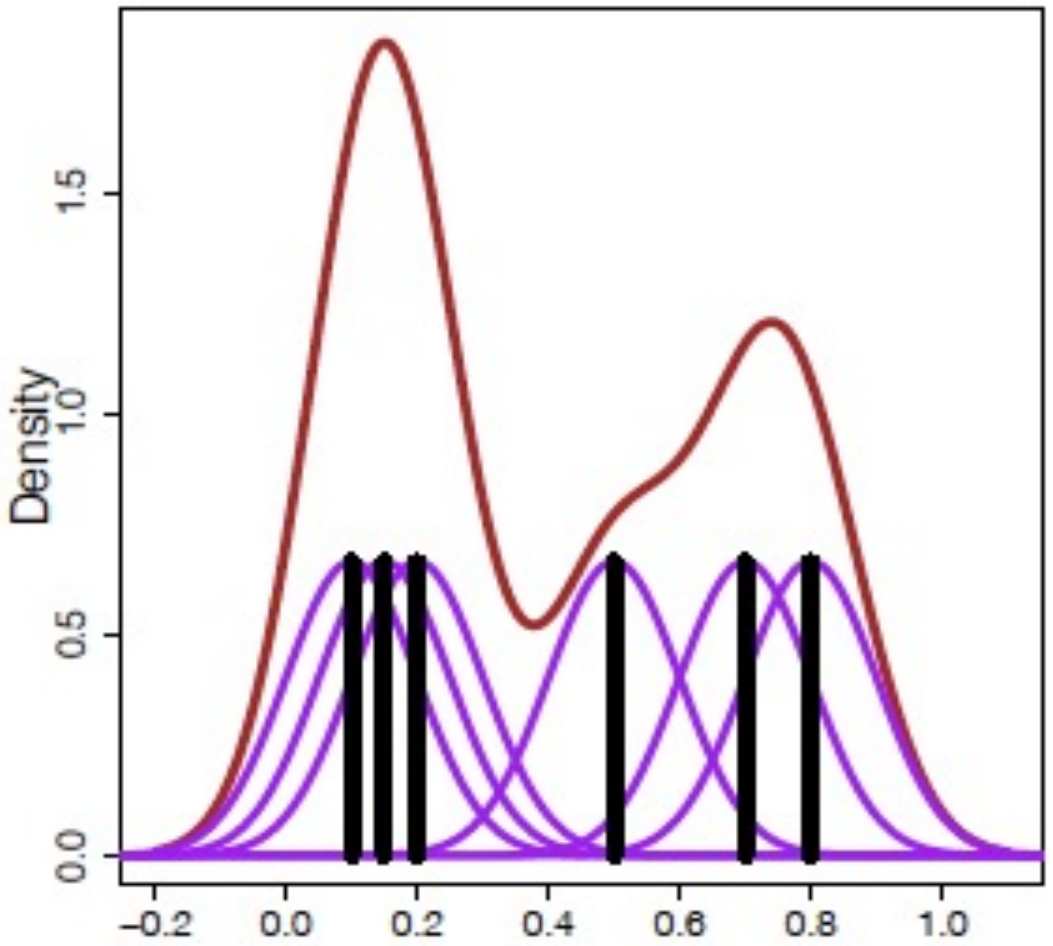


EX) SMOOTHING WITH GAUSSIAN KERNEL

그래서 Gaussian kernel을 사용하면 어떻게 되는데?



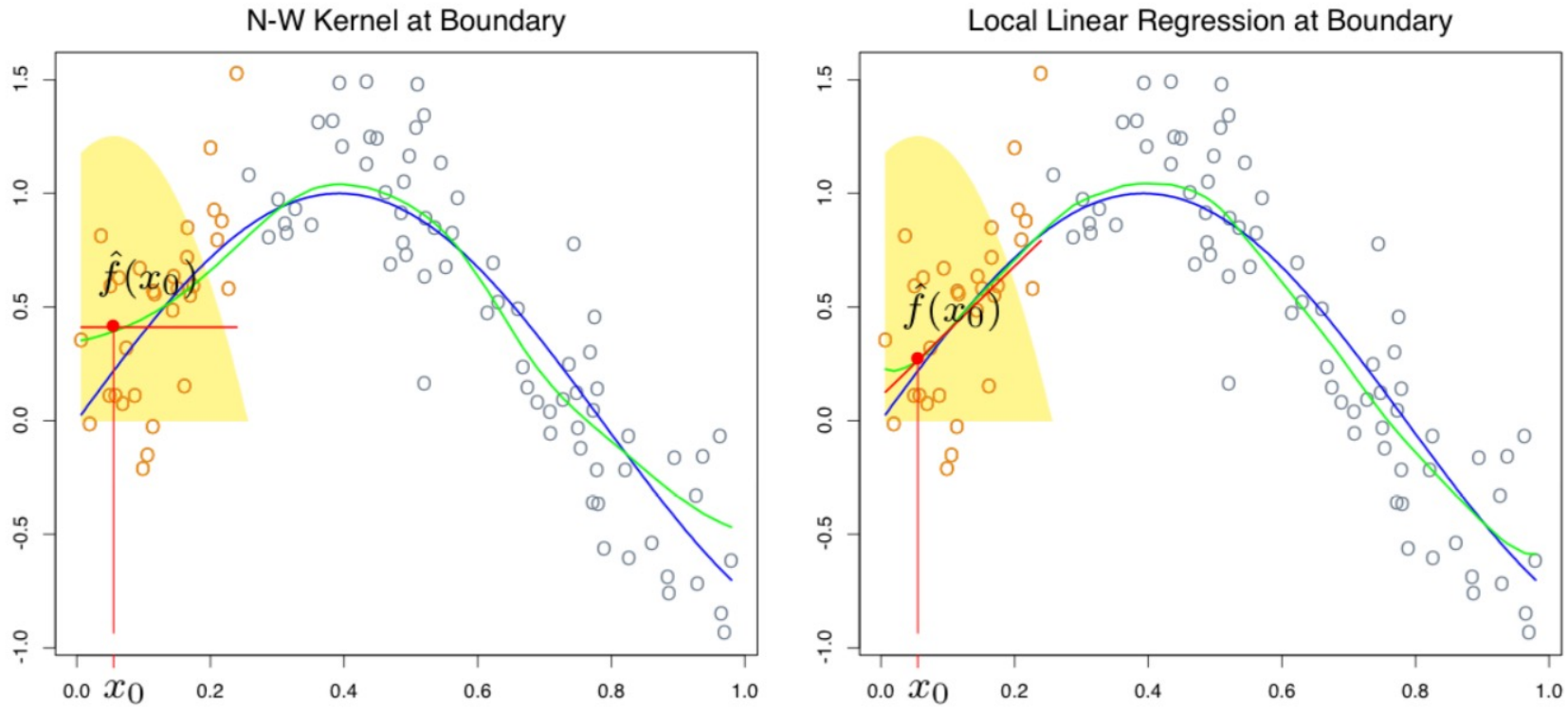
HOW DENSITY ESTIMATION WORKS



6.1.1 LOCAL LINEAR REGRESSION

- kernel function으로 부드럽게 fitting되었더라도 boundary에서는 매우 크게 bias되는 문제가 발생

(아래 그림의 왼쪽처럼!)



- This is because of the **asymmetry of the kernel in that region**.

HOW TO FIX?

- By fitting straight lines rather than constants locally,
  we can remove this bias exactly to first order (이전 슬라이드의 오른쪽 그림!)

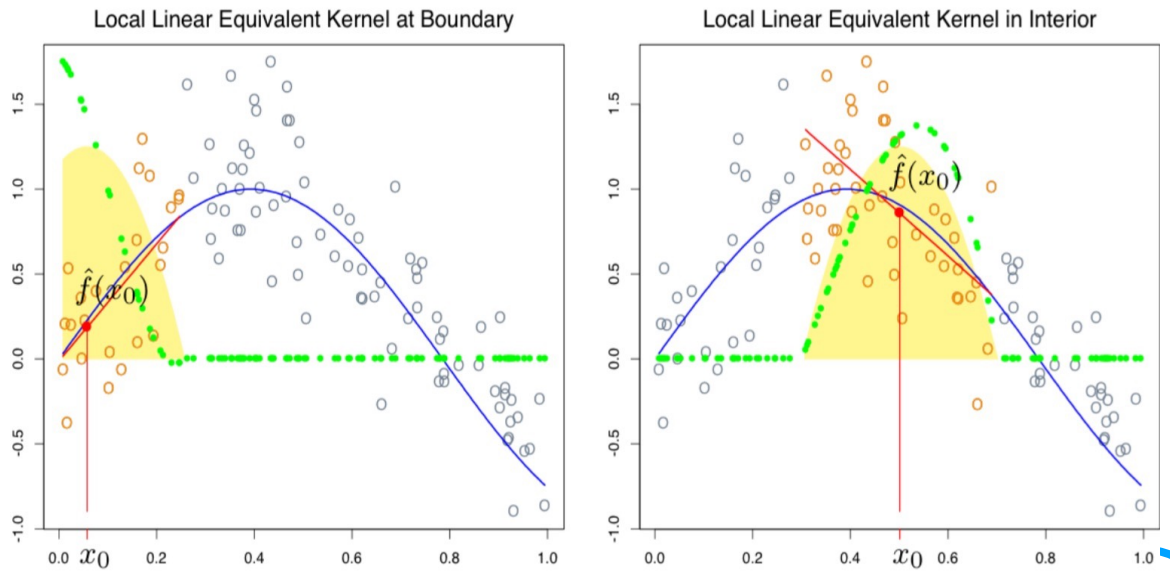
- local regression 방법은 x_0에서의 bias 문제를 (1차항 수준에서) 개선할 수 있다.

$$\min_{\alpha(x_0),\beta(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i)\left[y_i - \alpha(x_0) - \beta(x_0)x_i\right]^2 .$$

$$\hat{f}(x_0) = \hat{\alpha}(x_0) + \hat{\beta}(x_0)x_i \text{ vs. } \hat{f}(x) = \frac{\sum_i y_i \Phi_\sigma(x-x_i)}{\sum_i \Phi_\sigma(x-x_i)}$$

local regression vs. locally weighted averaging

(linear) vs. (single term)



DOES THIS ACTUALLY "FIX" THE PROBLEM?

local average kernel methods were corrected by modifying the kernel considering (asymptotic) MSE.

We can show it by both graphical plot and mathematical expansion!

$$\begin{aligned}\hat{f}(x_0) &= b(x_0)^T(\mathbf{B}^T\mathbf{W}(x_0)\mathbf{B})^{-1}\mathbf{B}^T\mathbf{W}(x_0)\mathbf{y} \\ &= \sum_{i=1}^{N} l_i(x_0)y_i.\end{aligned}$$

(Note that $bias = E\,\hat{f}\,(x_0) - f(x_0)$ and $\sum l_i(x_0) = 1$, $\sum l_i(x_0)(x_i - x_0) = 0$)

$$\begin{aligned}\mathrm{E}\hat{f}(x_0) &= \sum_{i=1}^{N} l_i(x_0)f(x_i) \\ &= f(x_0)\sum_{i=1}^{N} l_i(x_0) + f'(x_0)\sum_{i=1}^{N}(x_i - x_0)l_i(x_0) \\ &\quad + \frac{f''(x_0)}{2}\sum_{i=1}^{N}(x_i - x_0)^2 l_i(x_0) + R,\end{aligned}$$

"Simple" Linear Regression

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x = \hat{f}$$

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})\, y_i}{S_{xx}}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

[Matrix Notation]

$$\hat{f} = \hat{y} = X\hat{\beta}$$

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

vs.

"Local" Linear Regression

$$\hat{f} = \hat{\alpha}(x_0) + \hat{\beta}(x_0)\, x$$

$$\hat{\alpha}(x_0), \hat{\beta}(x_0) = \underset{\alpha(x_0), \beta(x_0)}{\operatorname{argmin}} \sum_{i=1}^{N} K_\lambda(x_0, x_i) [y_i - \alpha(x_0) - \beta(x_0) x_i]^2$$

a.k.a. $\hat{\beta}_0$, $\hat{\beta}_1$

Take MSE into account!

모든 data 다 쓰면 갔다와

번거로운 식

weight = 0인 애들은 사용되지 않을 것!

vs.

$$\hat{f}(x_0) = b(x_0)^T (B^T W(x_0) B)^{-1} B^T W(x_0) y$$

evaluating (predicting) at specific query point x_0

$x \approx B$

Kernel (weight)

6.1.2 LOCAL POLYNOMIAL REGRESSION 멈춰!!

📍 Can't we do better?

➡️ Why linear? We can fit of any degree d !

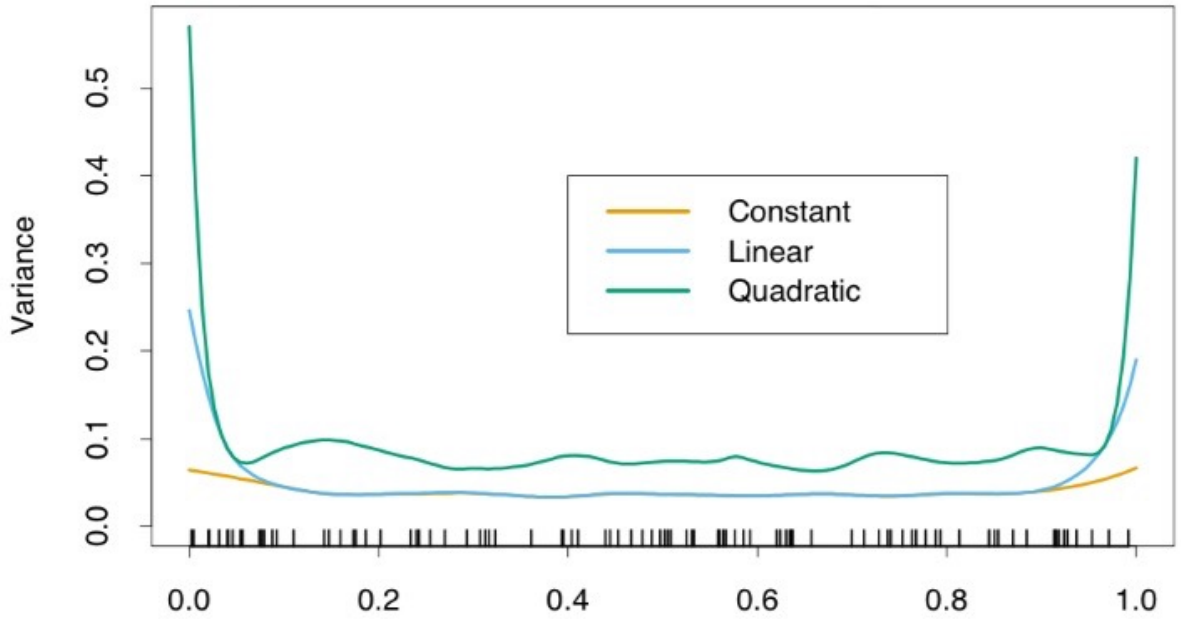
$$\min_{\alpha(x_0),\beta_j(x_0),\ j=1,\ldots,d} \sum_{i=1}^{N} K_\lambda(x_0, x_i) \left[y_i - \alpha(x_0) - \sum_{j=1}^{d} \beta_j(x_0) x_i^j \right]^2$$

Then the estimated function becomes...

$$\hat{f}(x_0) = \hat{\alpha}(x_0) + \sum_{j=1}^{d} \hat{\beta}_j(x_0) x_i^j$$

- 이렇게 되면 bias는 d+1 order이상의 term들로만 이루어진다 :D

SUMMARY



- Local **linear** fit : 적당한 cost의 variance로 boundary에서의 bias를 크게 줄일 수 있다.

그치만 interior region에서 curve를 메워버림..

"tend to be biased in regions of curvature of the true function, a phenomenon referred to as *trimming the hills* and *filling the valleys*"

- Local **quadratic** fit : domain의 대부분을 차지하는 interior region에서의 curvature를 잘 모델링할 수 있기 때문에 bias reduction 측면에서는 가장 효율적이다.

그치만 boundary에서는 bias를 줄이는 것에 비해 variance가 크게 증가함.

- 결론적으로 목적에 따라 적당한 degree를 알아서 잘 선택해야 한다.

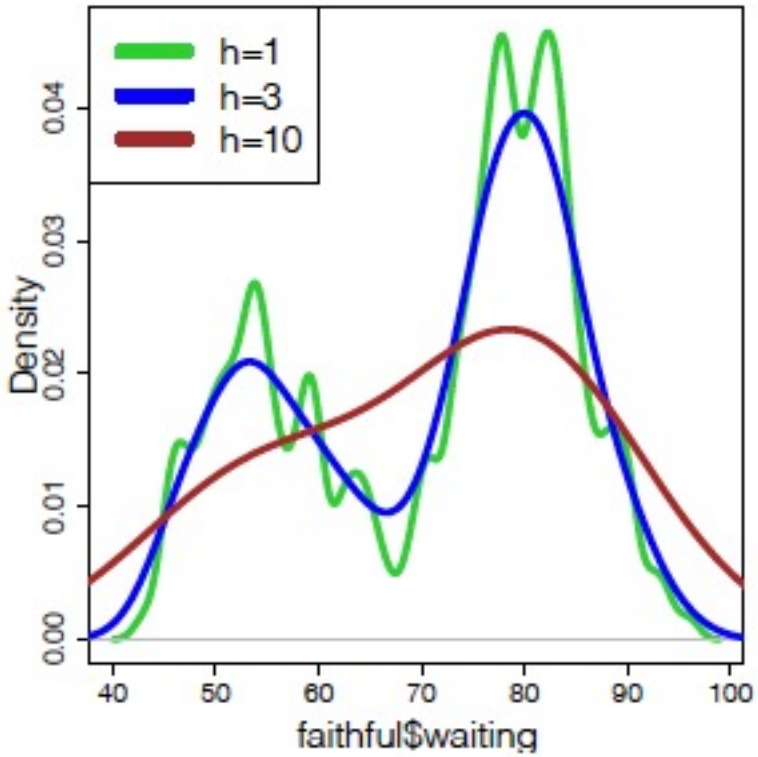
6.2 SELECTING THE WIDTH OF THE KERNEL

💡**a.k.a. "optimal bandwidth problem"**

λ **: control the width of kernel**

- 어떤 커널을 사용하냐에 따라서 구체적으로 람다가 뭔지는 달라진다.

!! Bias-Variance Tradeoff !!



SELECTING BANDWIDTH

With narrow window,

$\hat{f}$ is composed of small number of y_is.

(Likely to overfit to the query point)

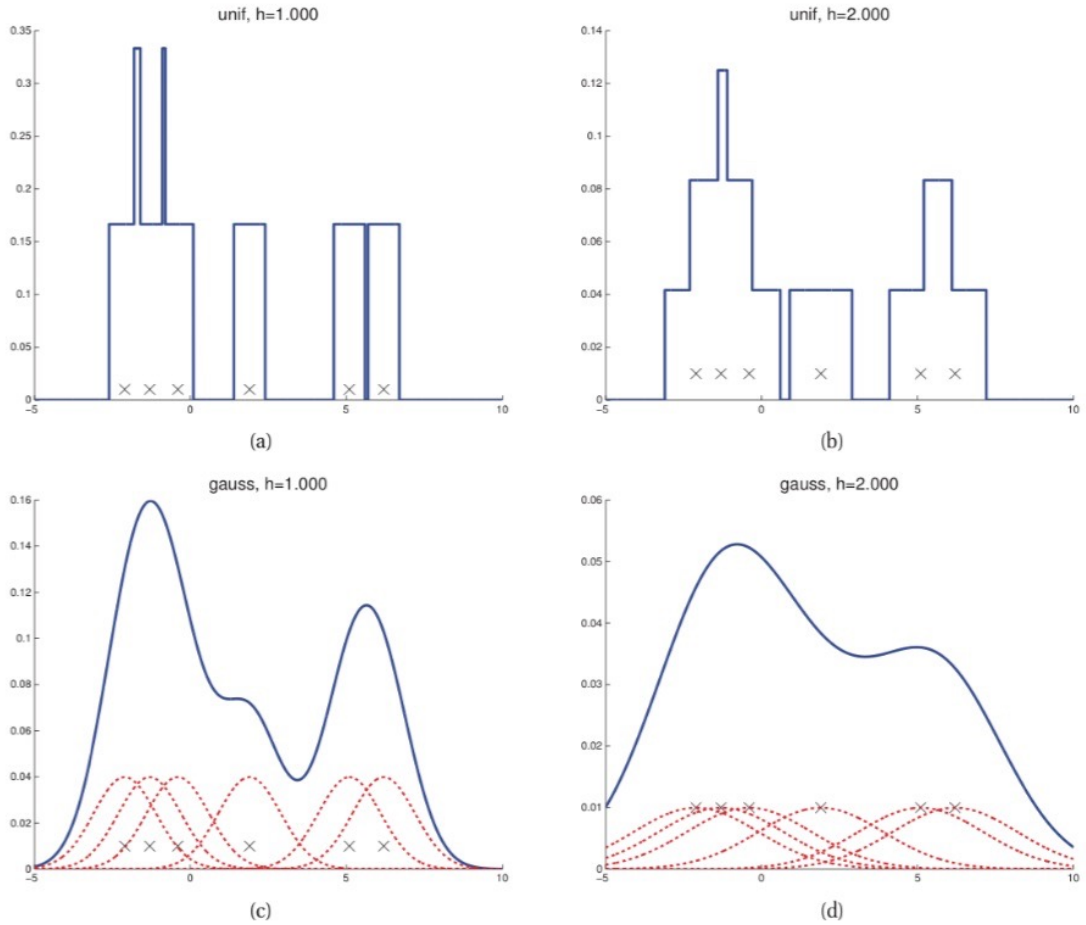
➡️ Small bias, but Large variance

With wide window,

$\hat{f}$ is composed of large number of y_is.

(Smooth a LOT!!)

➡️ Large bias, but Small variance



Selecting Bandwidth

- We can choose optimal bandwidth by performing k-fold cross validation or LOOCV

KERNEL IN GENERAL ML WORLD 🌏

💡 kernel function = measure of similarity

(between two objects which belong to same abstract space)

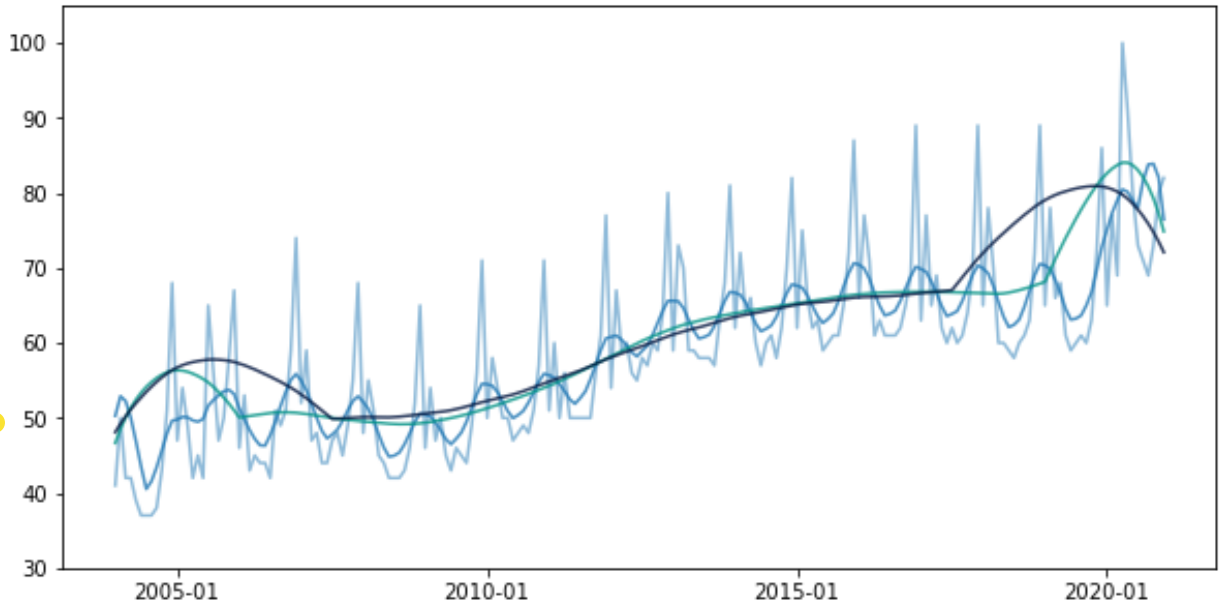
- 넓은 의미에서 cosine similarity (text analysis), gram-matrix (computer vision), linear kernel (내적) 등등 전부 다 kernel이다.
- 오만가지 커널이 있음. probability에서 파생된 것들도 있고, Fisher kernel도 있고, ...

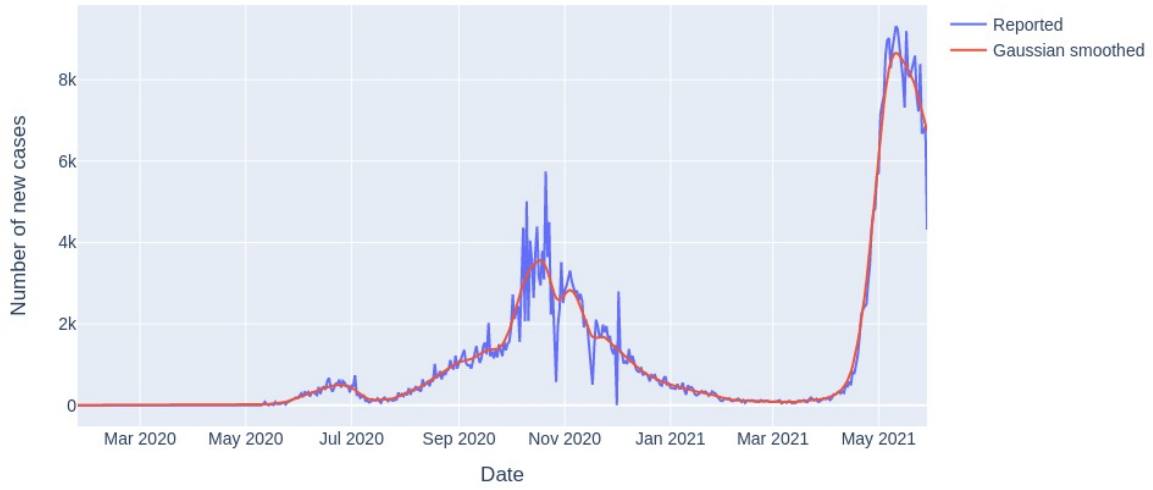
- 우리가 공부한 kernel은 Kernels for building generative models라고 한다.
(non-parametric density estimate p(y, **x**)를 생성하는데 사용되기 때문..🧠)

➡️ Thus belongs to unsupervised density estimation

OTHER PRACTICAL USAGES?

- smoothing time-series data
  - https://towardsdatascience.com/gaussian-smoothing-in-time-series-data-c6801f8a4dc3
  - https://towardsdatascience.com/kernel-density-estimation-and-non-parametric-regression-ecebebc75277

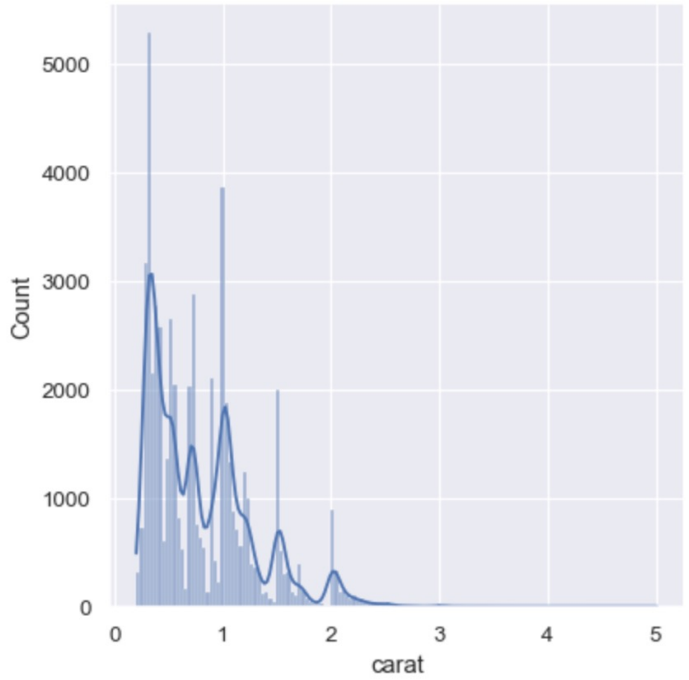




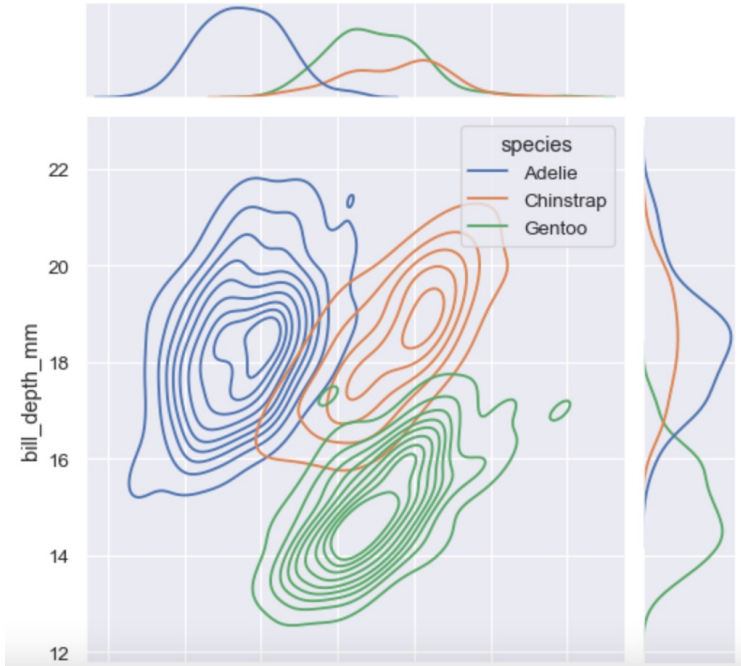
OTHER PRACTICAL USAGES?

- EDA (revisit!)
  - https://seaborn.pydata.org/tutorial/distributions.html

```
sns.displot(diamonds, x="carat", kde=True)
```




```
sns.jointplot(
    data=penguins,
    x="bill_length_mm", y="bill_depth_mm", hue="species",
    kind="kde"
)
```



Break Time

Kernel Density Estimation

(PART II)

ESL 6.6 / 6.7

NAÏVE BAYES CLASSIFIER

KL DIVERGENCE

Maximum entropy, given mean and standard deviation

$p(x)$: probability density function of X

$$\int p(x)dx = 1$$

$$\mu = \int p(x)xdx$$

$$\sigma^2 = \int p(x)(x-\mu)^2 dx$$

$$H(p) = -\int p(x)logp(x)dx$$

KL DIVERGENCE

Maximum entropy, given mean and standard deviation

$$L = H(p) + \lambda_1(\int p(x)dx - 1) + \lambda_2(\int xp(x)dx - \mu) + \lambda_3(\int p(x)(x-\mu)^2 dx - \sigma^2$$

Euler - Lagrange equation

$$\frac{\partial F}{\partial p} - \frac{d}{dx}\frac{\partial F}{\partial p'} = 0$$

$$p(x) = exp(-1 + \lambda_1 + \lambda_2 x + \lambda_3(x-\mu)^2)$$

$$p(x) = \frac{1}{(2\pi\sigma^2)^{1/2}} exp(-(x-\mu)^2/2\sigma^2)$$

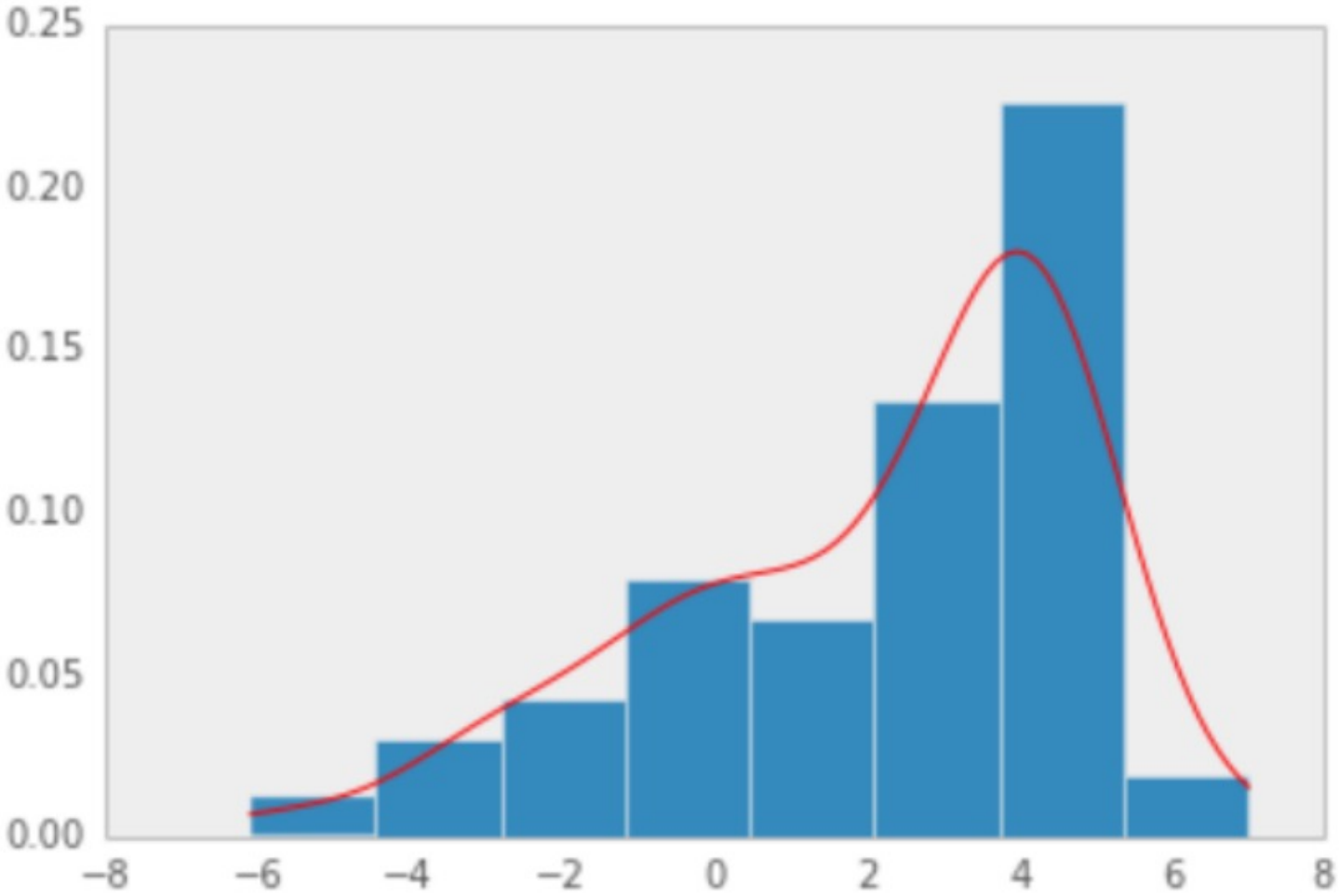
6.6 KERNEL DENSITY ESTIMATION (KDE)

Density Estimation

: from given data, we estimate probability density function of random variable

Parametric vs Nonparametric

- parametric: given pdf model, we estimate parameters
- Nonparametric: only from data!!



Simple, but not smooth, not accurate …

SO WE HAVE KERNEL DENSITY ESTIMATION!

We use kernels to estimate density function.

Ex)

Local estimate

$$\hat{f}_X(x_0) = \frac{\#x_i \in \mathcal{N}(x_0)}{N\lambda}$$

Smooth Parzen estimate (KDE)

$$\hat{f}_X(x_0) = \frac{1}{N\lambda}\sum K_\lambda(x_0, x_i) = \frac{1}{N\lambda}\sum K((x_i - x_0)/\lambda)$$

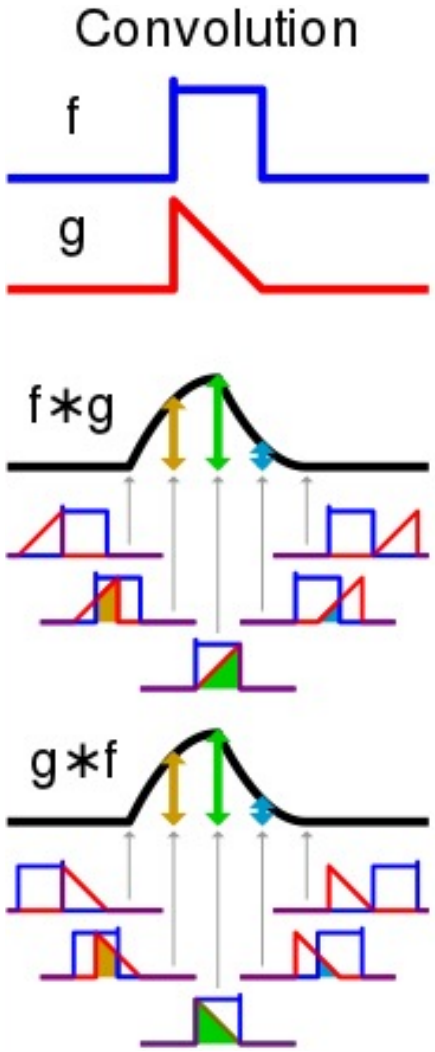
+ CONVOLUTION

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau$$

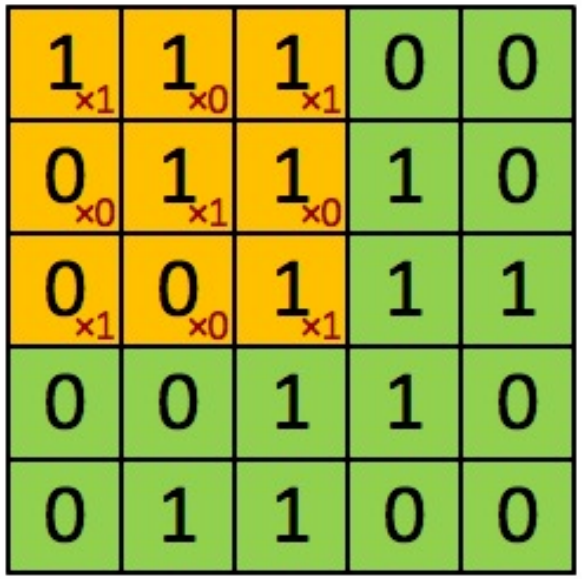
$$(f * g)(n) = \sum_{m=-\infty}^{\infty} f(m)g(n-m)$$

Ex) Gaussian kernel

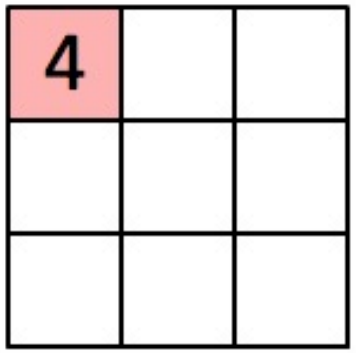
$K_\lambda(x_0, x) = \phi(|x - x_0|/\lambda)$ where ϕ is standard Gaussian pdf



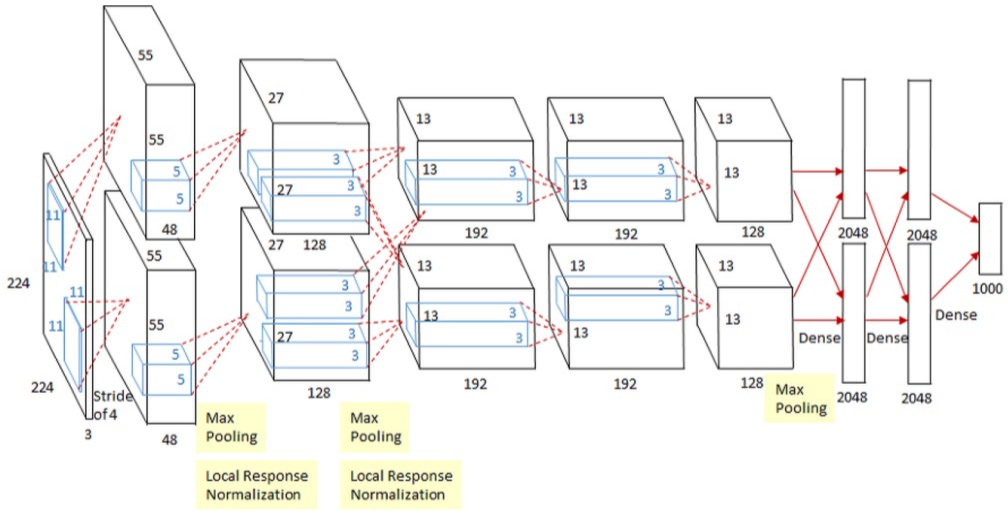
CONVOLUTIONAL NEURAL NETWORK (CNN)

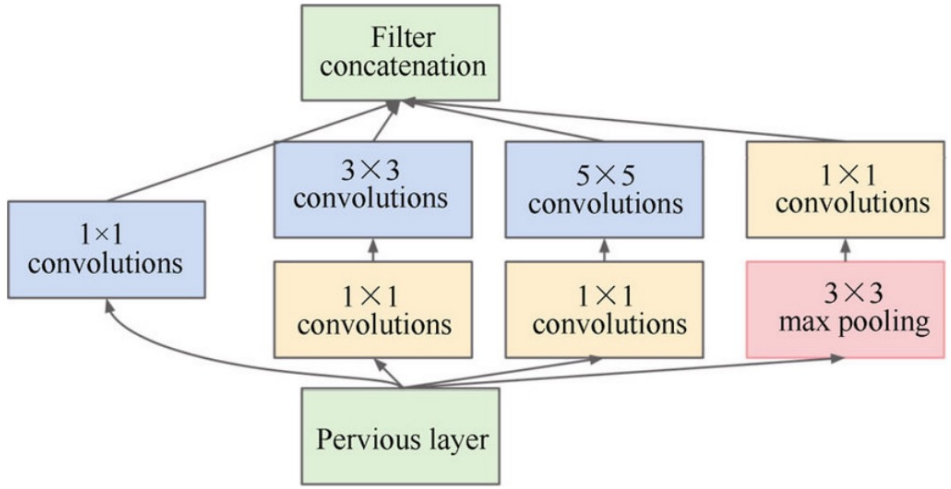


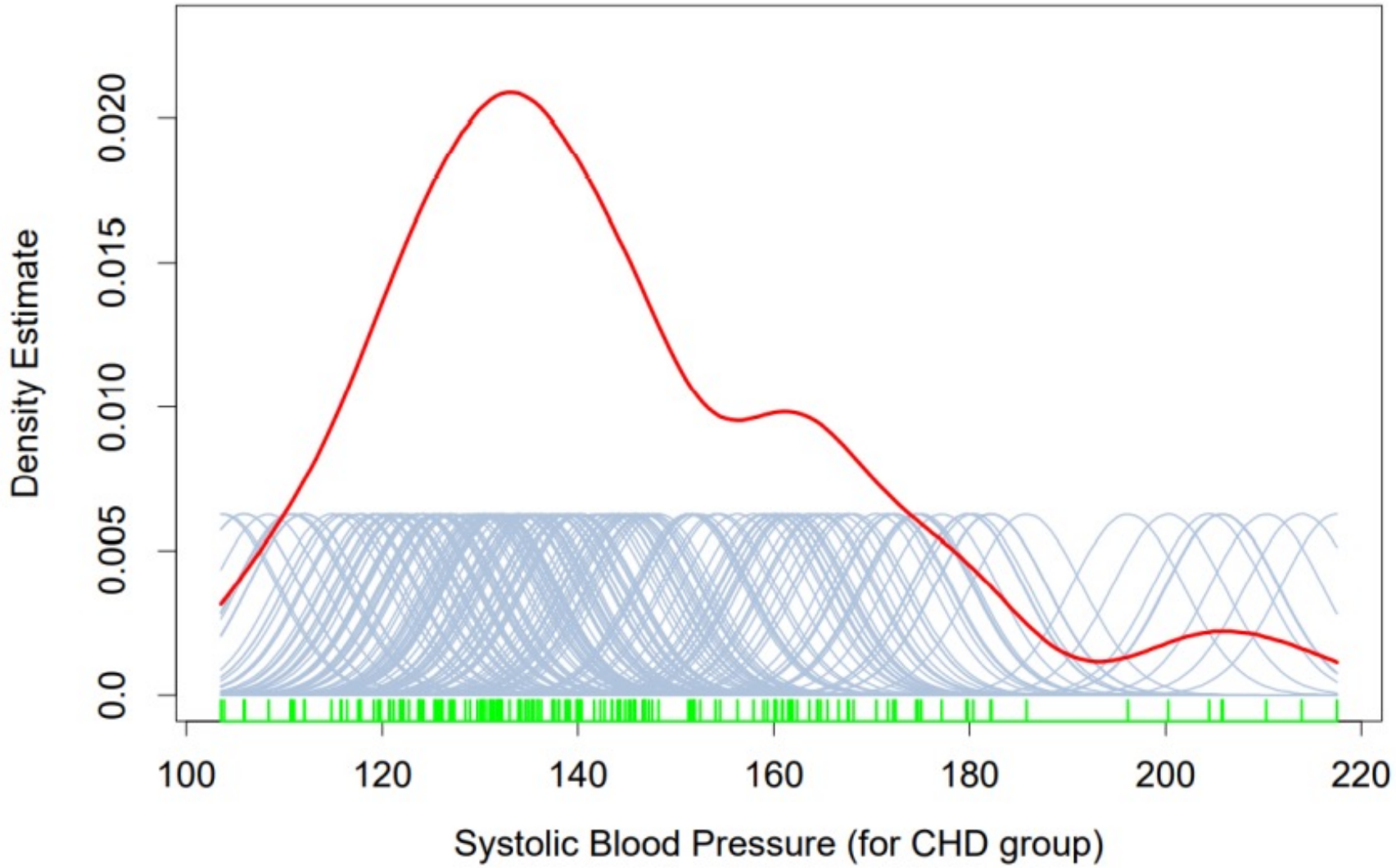
Image



Convolved Feature






Density Estimate
0.0
0.005
0.010
0.015
0.020
100
120
140
160
180
200
220
Systolic Blood Pressure (for CHD group)

THEN WHAT WE HAVE TO CHOOSE? (1)

(1) Kernel

Kernel should be ……

- Symmetric

- Generally positive

- $\int K(x)dx = 1$

- $\lim_{x\to-\infty} K(x) = \lim_{x\to\infty} K(x) = 0$

THEN IS OUR $\hat{f}$ WELL ESTIMATED?

We would check if $$\int \hat{f}(x)dx = 1$$

Also we would calculate the mean and variance of estimated density function.

- $\int K(x)dx = 1$

Let $\hat{f}_X(x_0) = \frac{1}{N\lambda}\sum K((x_i - x_0)/\lambda) = \frac{1}{N}\sum \delta(x_i - x_0)$ where $\delta(x) = \frac{1}{\lambda}K(x/\lambda)$

Then $\int \delta(x - x_i) = 1$

So $\int \hat{f}(x)dx = 1$

MEAN AND VARIANCE

mean $\int x\hat{f}(x)dx = \frac{1}{n}\sum x_i$

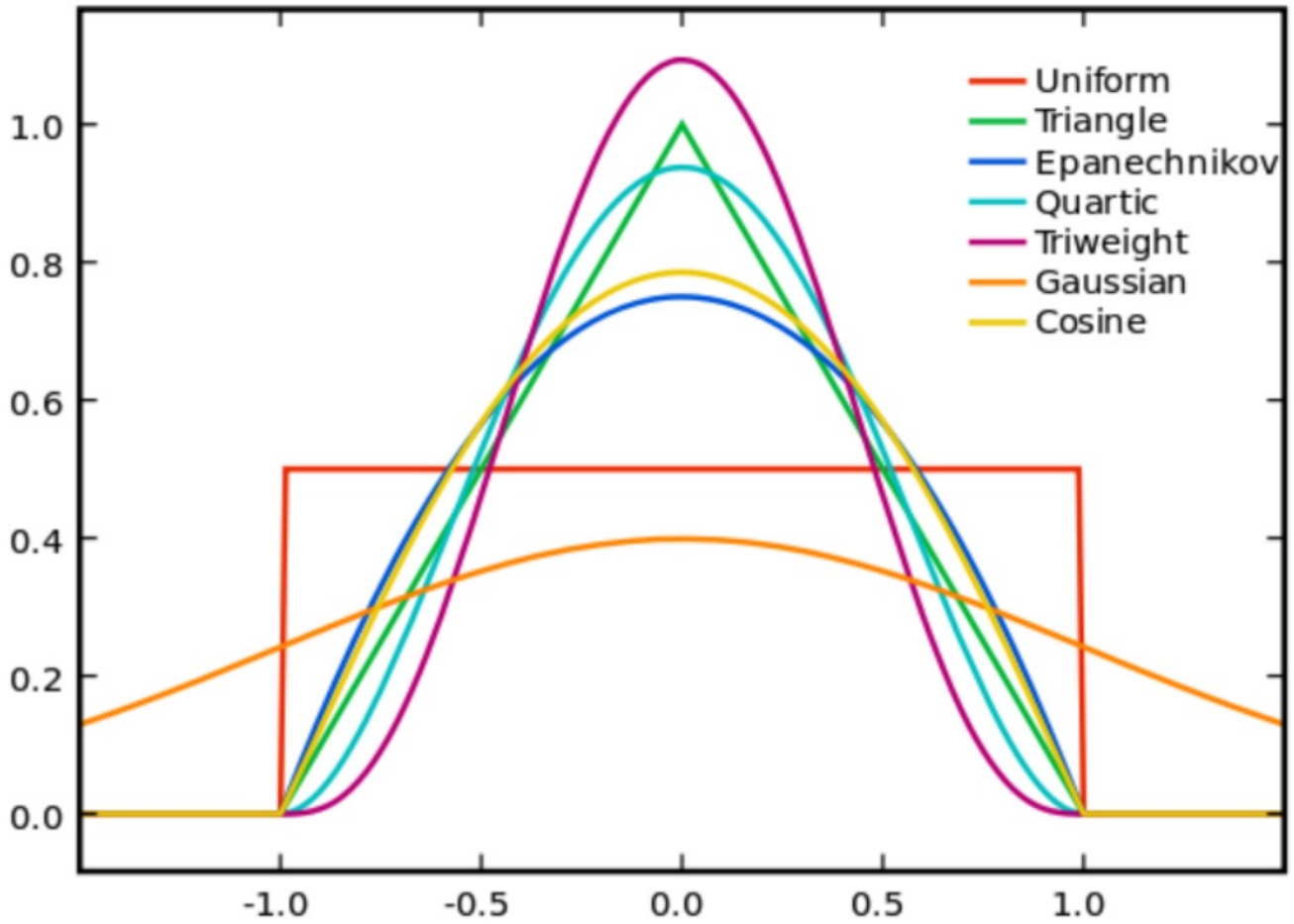
variance $\int x^2\hat{f}(x)dx = \frac{1}{n}\sum x_i^2 + \lambda^2 \int u^2 K(u)du$

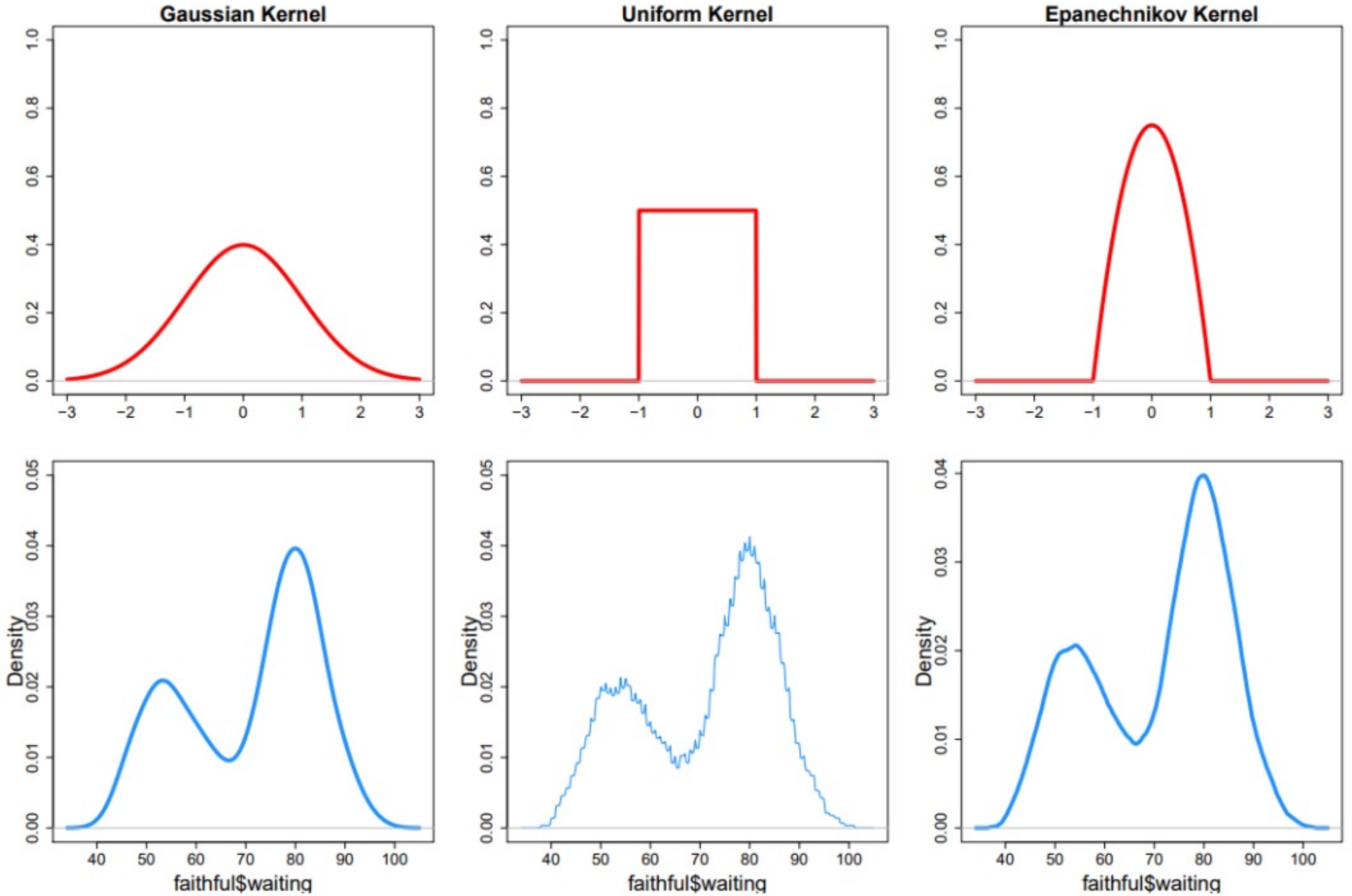
EXAMPLES OF KERNELS

Gaussian kernel: $K(x) = \phi(|x - x_0|/\lambda) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$

Uniform kernel: $K(x) = \frac{1}{2} I(-1 \leq x \leq 1)$

Epanechnikov kernel: $K(x) = \frac{3}{4} max\{1 - x^2, 0\}$

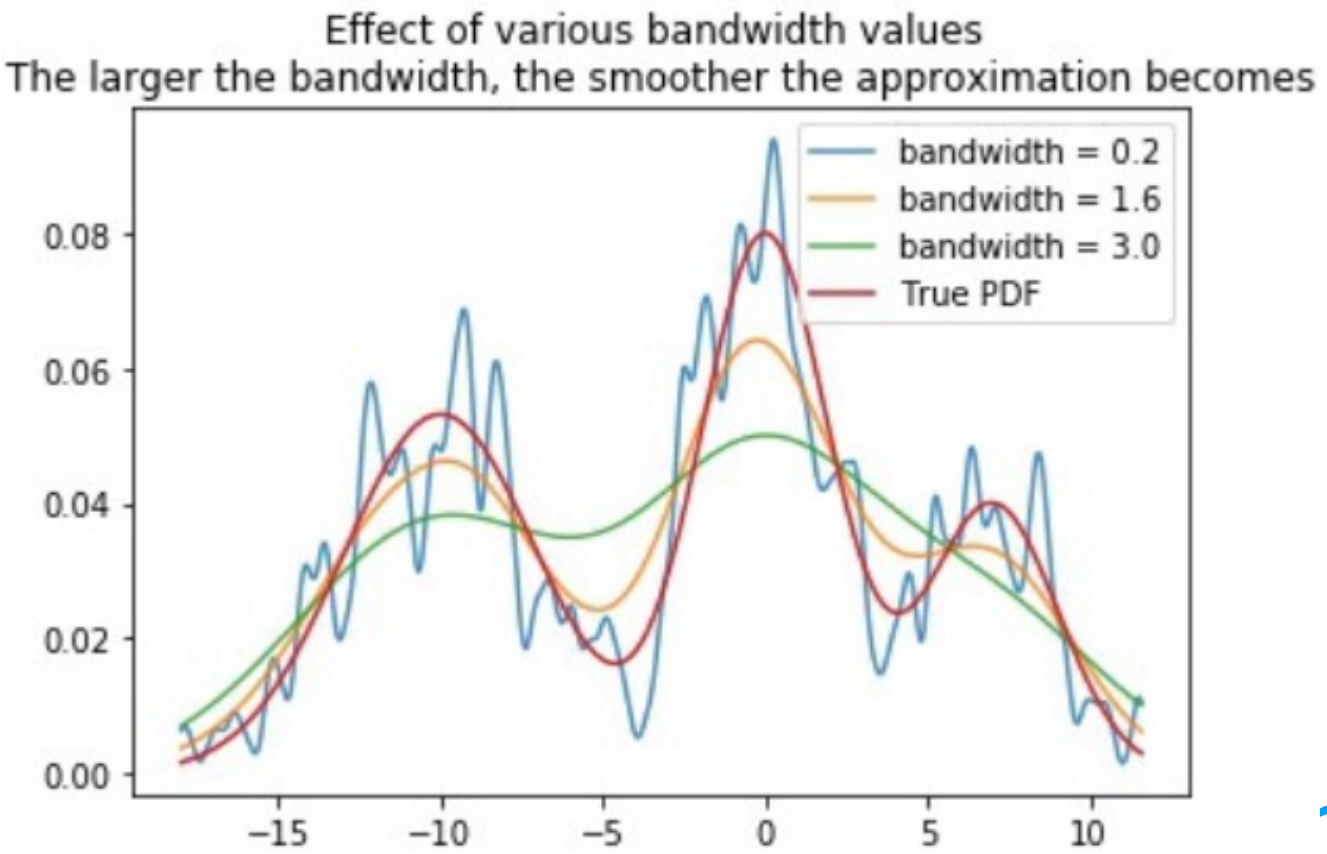



Gaussian Kernel
Uniform Kernel
Epanechnikov Kernel
Density
faithful$waiting

THEN WHAT DO WE HAVE TO CHOOSE? (2)

(2) bandwidth

- Too small → undersmoothing
- Too large → oversmoothing



IS KDE GOOD ESTIMATION?

$$\begin{aligned}\text{Expected MSE} &= E\left[(Y-\hat{Y})^2|X\right] \\ &= \sigma^2 + (E[\hat{Y}]-\hat{Y})^2 + E[\hat{Y}-\mathrm{E}[\hat{Y}]]^2 \\ &= \sigma^2 + \mathrm{Bias}^2(\hat{Y}) + \mathrm{Var}(\hat{Y}) \\ &= \text{Irreducible Error} + \mathrm{Bias}^2 + \text{Variance}\end{aligned}$$

So we calculate the bias and variance of KDE

BIAS OF KDE

$$E(\hat{f}(x_0)) - f(x_0)$$

$$= \frac{1}{\lambda} \int K((x - x_0)/\lambda) f(x) dx - f(x_0)$$

$$= \int K(y) f(x_0 + \lambda y) dy - f(x_0)$$

After Taylor expansion, we let

$$f(x_0 + \lambda y) = f(x_0) + \lambda y f'(x_0) + \frac{1}{2}\lambda^2 y^2 f''(x_0) + o(\lambda^2)$$

So, the bias becomes

$$\frac{1}{2}\lambda^2 f''(x_0) \int y^2 K(y) dy + o(\lambda^2)$$

VARIANCE OF KDE

$$Var(\hat{f}(x_0))$$

$$= Var(\frac{1}{N\lambda}\sum K((x_i - x_0)/\lambda))$$

$$\leq \frac{1}{n\lambda^2} E(K^2((x_i - x_0)/\lambda))$$

After Taylor expansion, we let

$$f(x_0 + \lambda y) = f(x_0) + \lambda y f'(x_0) + \frac{1}{2}\lambda^2 y^2 f''(x_0) + o(\lambda^2)$$

So, the variance is less than or equal to

$$\frac{1}{n\lambda} f(x_0) \int K^2(y)dy + o(1/n\lambda)$$

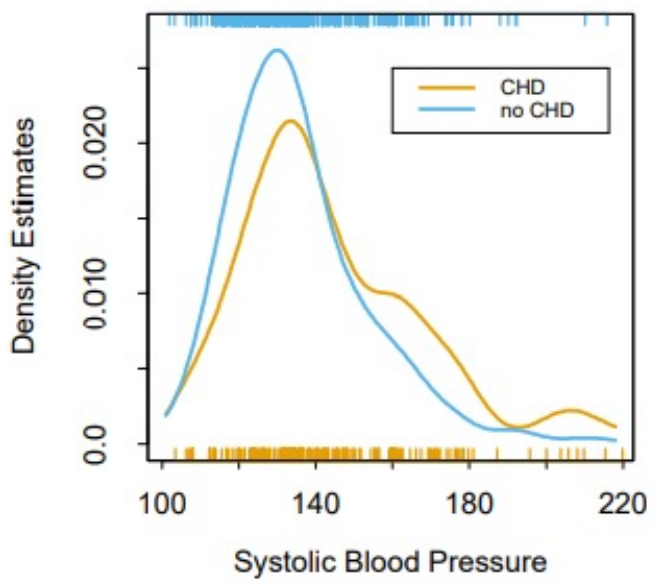
FURTHER APPLICATIONS

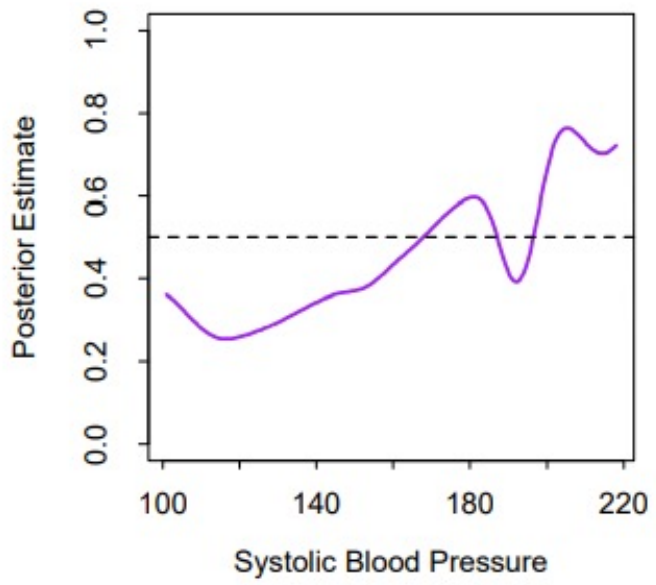
(1) Kernel Density Classification

Suppose J class so we have density estimates $\hat{f}_j(X), j = 1, \ldots J$

Let priors $\hat{\pi}_j$ be sample proportions.

$$\hat{Pr}(G = j|X = x_0) = \hat{\pi}_j \hat{f}_j(x_0) / \sum \hat{\pi}_k \hat{f}_k(x_0)$$





FURTHER APPLICATIONS

(2) Naïve Bayes Classifier

- We just assume features are independent given a class G.

$$f_j(X) = \Pi f_j k(X_k)$$

- Especially when dimension p of the feature is high

(since density estimation doesn’t work well)

p(**Dear** | **N**) = 0.47
p(**Friend** | **N**) = 0.29
p(**Lunch** | **N**) = 0.18
p(**Money** | **N**) = 0.06

p(**N**) = 0.67

0.67 × 0.47 × 0.29 = 0.09 ∝ p(**N** | **Dear Friend**)

p(**Dear** | **S**) = 0.29
p(**Friend** | **S**) = 0.14
p(**Lunch** | **S**) = 0.00
p(**Money** | **S**) = 0.57

p(**S**) = 0.33

0.33 × 0.29 × 0.14 = 0.01 ∝ p(**S** | **Dear Friend**)

HOMEWORK

💻 **HW** : kernel smoothing from scratch (using R) + experiment

✏️ Ex. 6.8 Suppose that for continuous response Y and predictor X, we model the joint density of X, Y using a multivariate Gaussian kernel estimator. Note that the kernel in this case would be the product kernel $\phi_\lambda(X)\phi_\lambda(Y)$. Show that the conditional mean $E(Y|X)$ derived from this estimate is a Nadaraya–Watson estimator. Extend this result to classification by providing a suitable kernel for the estimation of the joint distribution of a continuous X and discrete Y.

REFERENCE

- 📚 ESL 2.8
- 📚 ESL 6.1 / 6.2 / 6.6 / 6.7
- 📚 Probabilistic Machine Learning (Kevin L. Murphy) - 14. Kernels
- 🔗 https://sites.google.com/view/theostat/research/probability-density-function-estimation
  (서울대학교 박병욱 교수님 연구실 페이지)