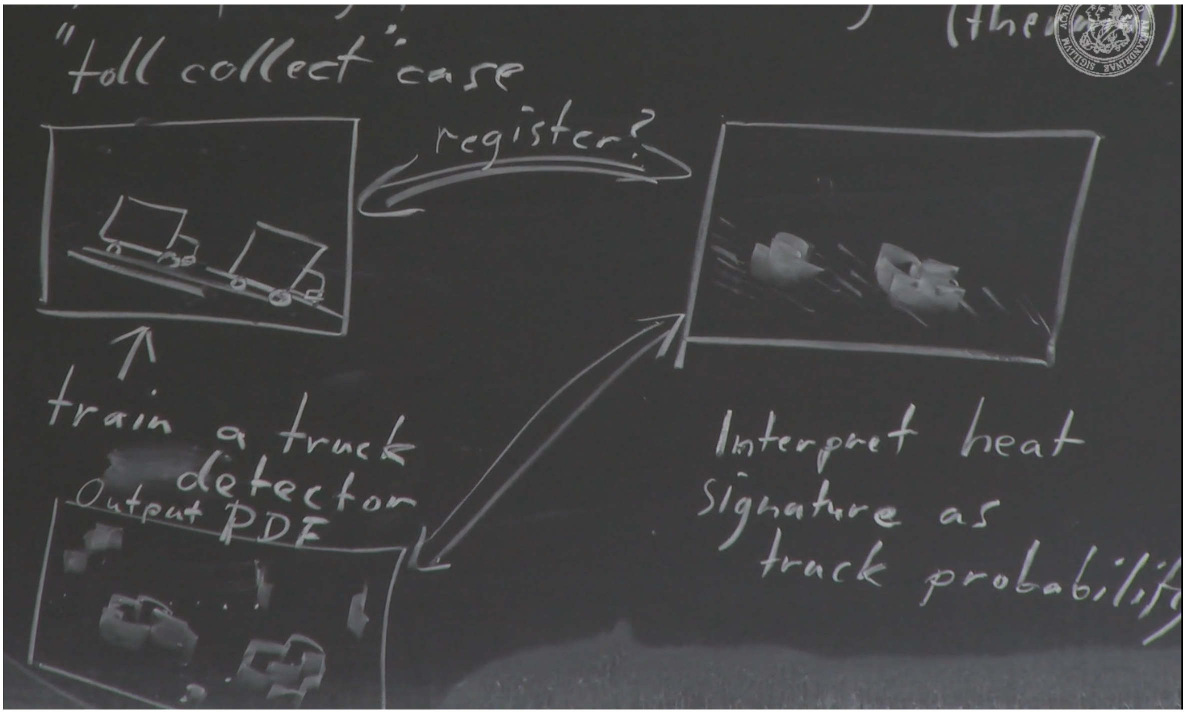
PA-2018 - 03: Non-parasitic density function

why might i want to estimate a probability density function (PDF) from some discreate observation?

- compute statistical measures, for example in Mutual information
    - image from modality 1
        - e.g. a photograph
    - image from modality 2
        - e.g. infrared image



- sample(draw) new observations with the same distribution as the actual(measurement)

Parzen Window Estimator

Idea: Given a set of discrete observations "smear" them out to obtain a PDF
Let $S = \{x_1, x_2, x_3, \ldots\ldots x_N\}$ denote the set of observations.
Let PR denote the probability that x is falling into region R.

$$PR = \int_R p(x)\partial x \tag{1}$$

if we assume $p(x)$ is approximately constant in R,

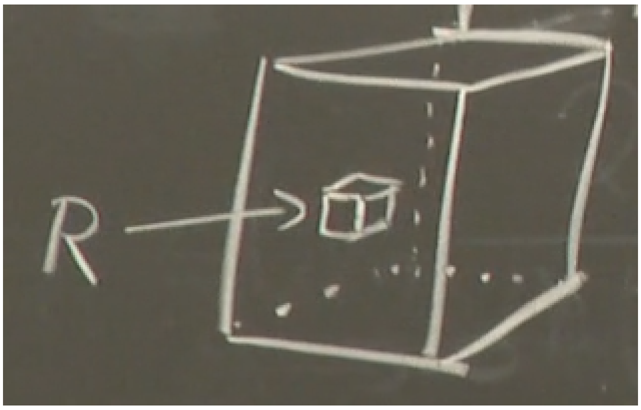
then $PR \approx p(x) * \int_R \partial x$

Here $\int_R \partial x = V_R$ is the "Volume" of R

$\Rightarrow PR = p(x) * V_R = \frac{k_R}{N} * V_R$

N is the number of features that fall in R over # of all features.

That called relative frequency feature in R.



For example, let R be a d-dimensional hypercube and let h denote the side-length of the hypercube$V_R = h^d$

The kernel window function is

$K(x_c, x) = {}^{1}_{0}$ if $\frac{x_{i,k} - x_k}{h} \leq \frac{1}{2}$

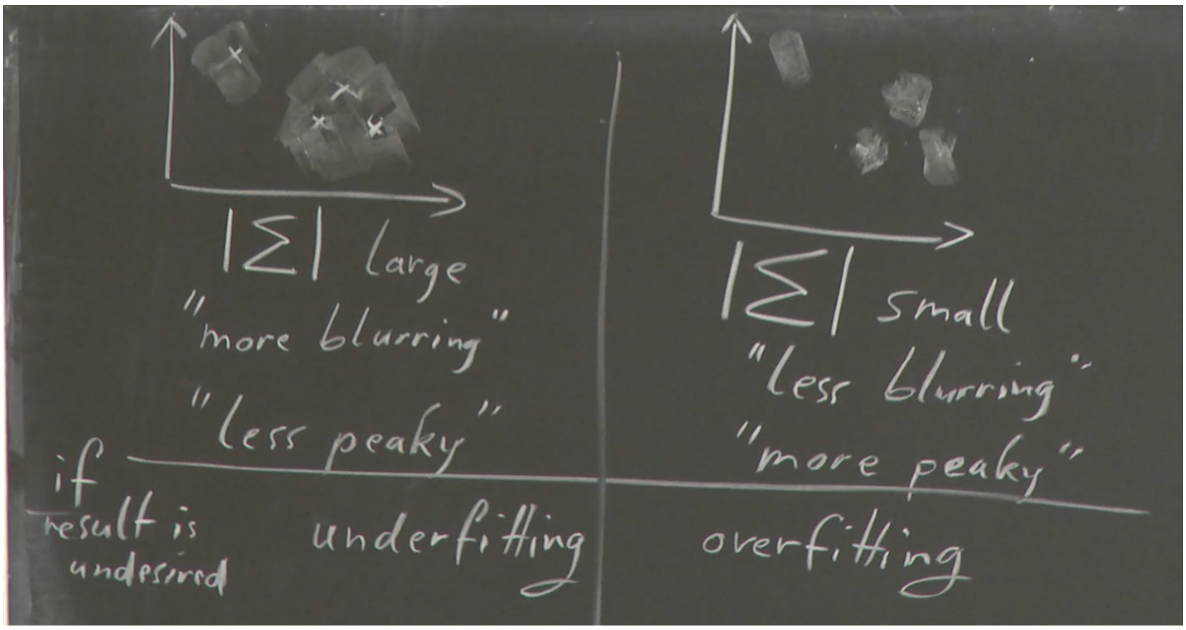
$$\Rightarrow Rewrite P(x) = \frac{1}{N} * \sum_{i=1}^{N} K(x_i, x) \tag{2}$$

Alternatively, $K(x_i, x)$ can be any other kernel for example Gaussian.
$K_{\sum}(x_i, x) = \frac{1}{\sqrt{det(\sum) * 2\pi}} * e^{-\frac{1}{2}(x_i - x)\sum^{-1}(x_i - x)}$

where$\sum$ is covariance

Question tree

Question tree: how do we obtain $\sum$?
equivalent: how do we obtain h for the hypercube?
Qualitatively, how doses the result change if $|\sum|$ become larger or smaller?



Estimation of the covariance $\sum$ or the window width h can be done via ML estimation in the case of limited training data additionally with cross-validation

Cross-validation

Let $P^i_{\lambda,N-1}$ be the PDF defined by $S = \frac{S}{x_i}$
$\lambda = \sum, h$

Then we consider the objective function

$$\hat{\lambda} = argmax_{\lambda} \prod_{i=1}^{N} P^i_{\lambda,N-1}(x_i) \tag{3}$$

$P^i_{\lambda,N-1}$ is "Trained model for all samples except of x_i"
(x_i) is Test sample

$= argmax_{\lambda} \sum_{i=1}^{N} logP^i_{\lambda,N-1}(x_i)$ log-likelihood
For a differentiable kernel, we can now compute the gradient and look for an optimun, if the kernel is non-differentiable, we have to "brute force" the solution(Note: Gaussian=differentiable, Hypercube=non-diff.)