Non-parametric methods

Aarti Singh

Machine Learning 10-315
Feb 23, 2022

Parametric methods

- Assume some model (Gaussian, Bernoulli, Multinomial, logistic, network of logistic units, Linear, Quadratic) with fixed number of parameters
  - Gaussian Bayes, Naïve Bayes, Logistic Regression, Neural Networks

- Estimate parameters ($\mu, \sigma^2, \theta, w, \beta$) using MLE/MAP and plug in

- **Pro** – need few data points to learn parameters
- **Con** – Strong modeling assumptions, not satisfied in practice

Non-Parametric methods

- Typically don't make any modeling assumptions
- As we have more data, we should be able to learn more complex models
- Let number of parameters scale with number of training data

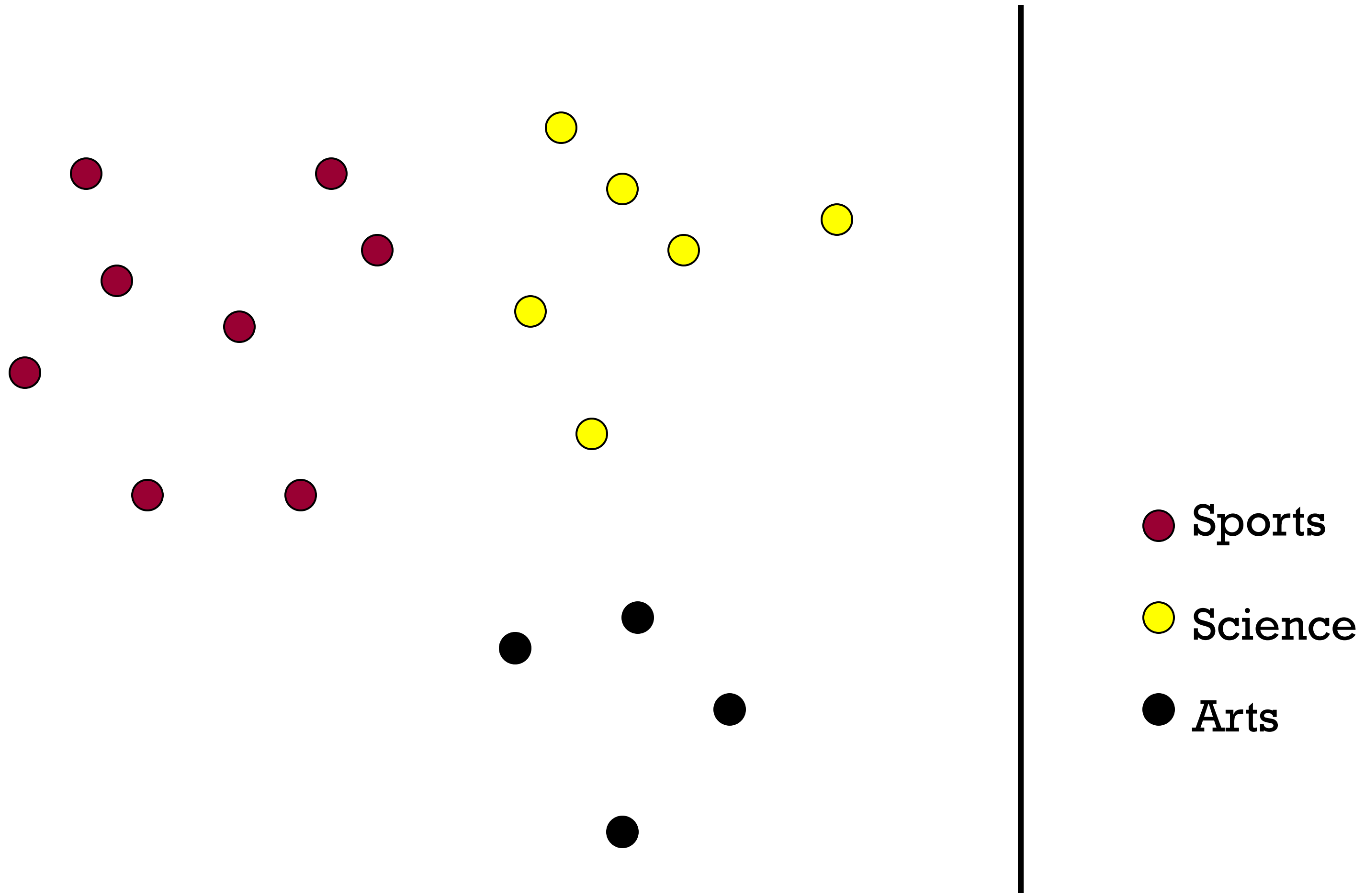
- Some nonparametric methods

  Classification: Decision trees, k-NN (k-Nearest Neighbor) classifier

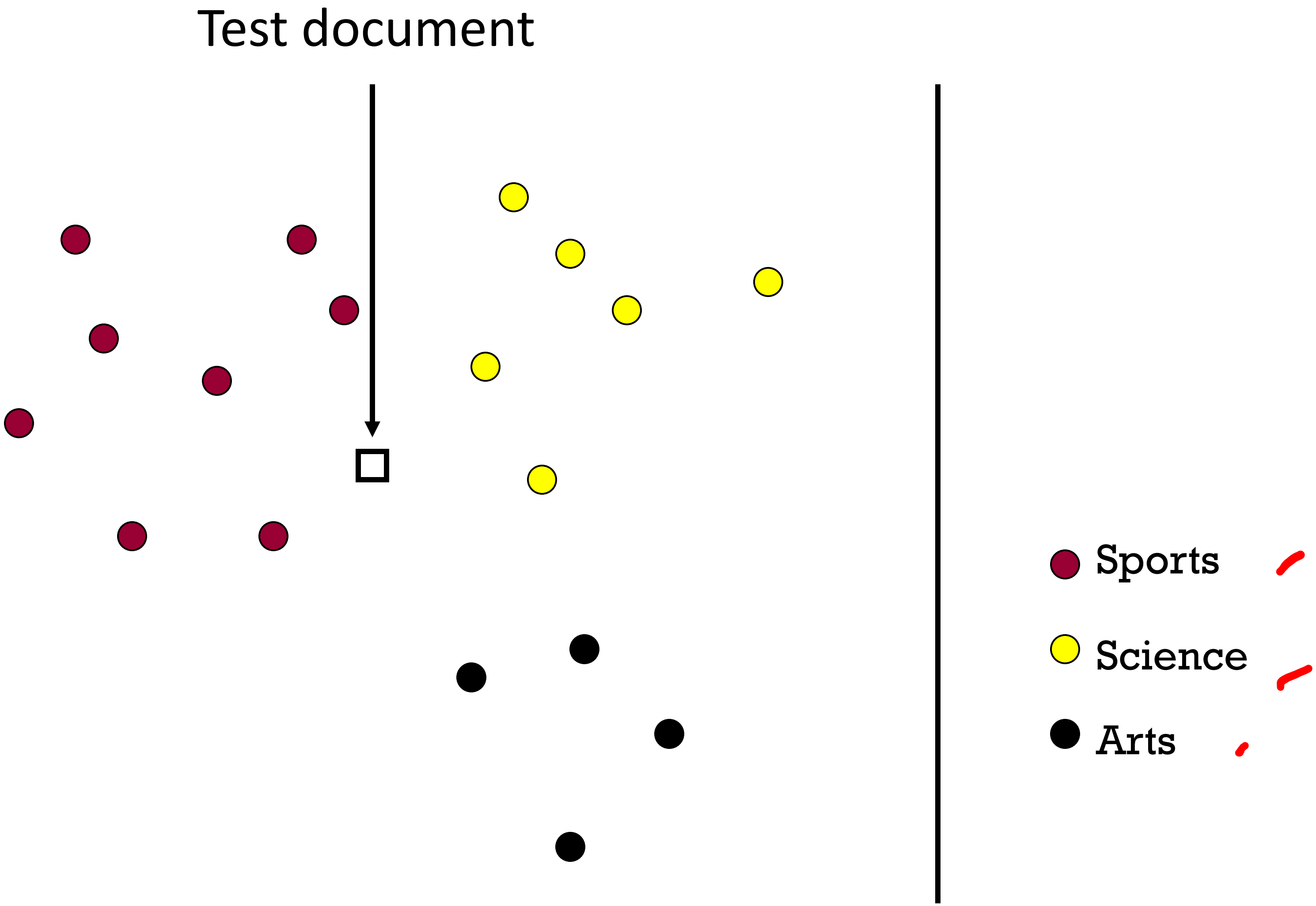
  Density estimation: k-NN, Histogram, Kernel density estimate

  Regression: Kernel regression

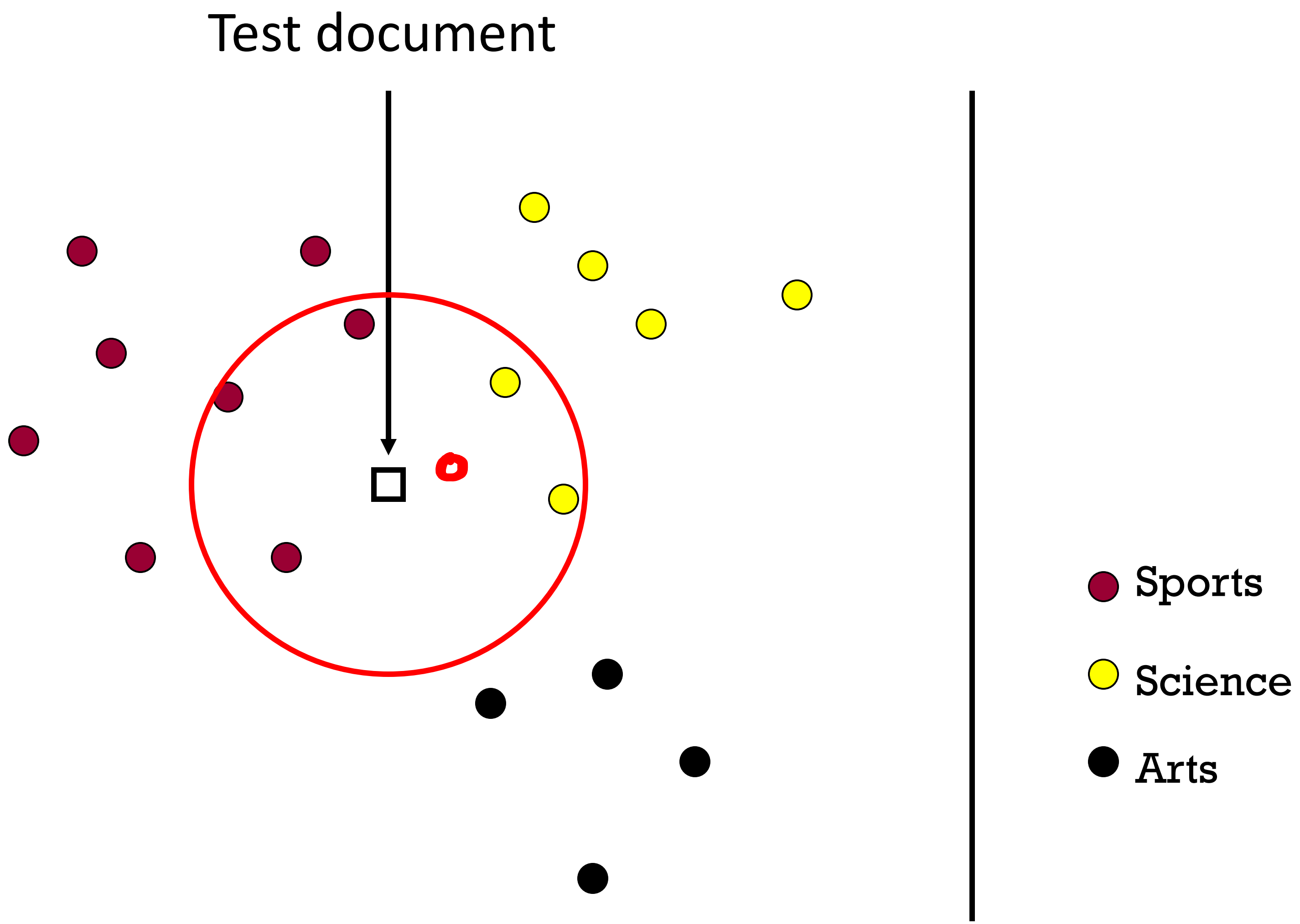
k-NN classifier



k-NN classifier



k-NN classifier (k=5)



What should we predict? … Average? Majority? Why?

k-NN classifier

- Optimal Classifier: $$f^*(x) = \arg\max_y P(y|x) = \arg\max_y P(x|y)P(y)$$

- k-NN Classifier: $$\widehat{f}_{kNN}(x) = \arg\max_y \widehat{P}_{kNN}(x|y)\widehat{P}(y) = \arg\max_y k_y$$

$$\widehat{P}_{kNN}(x|y) = \frac{k_y}{n_y}$$

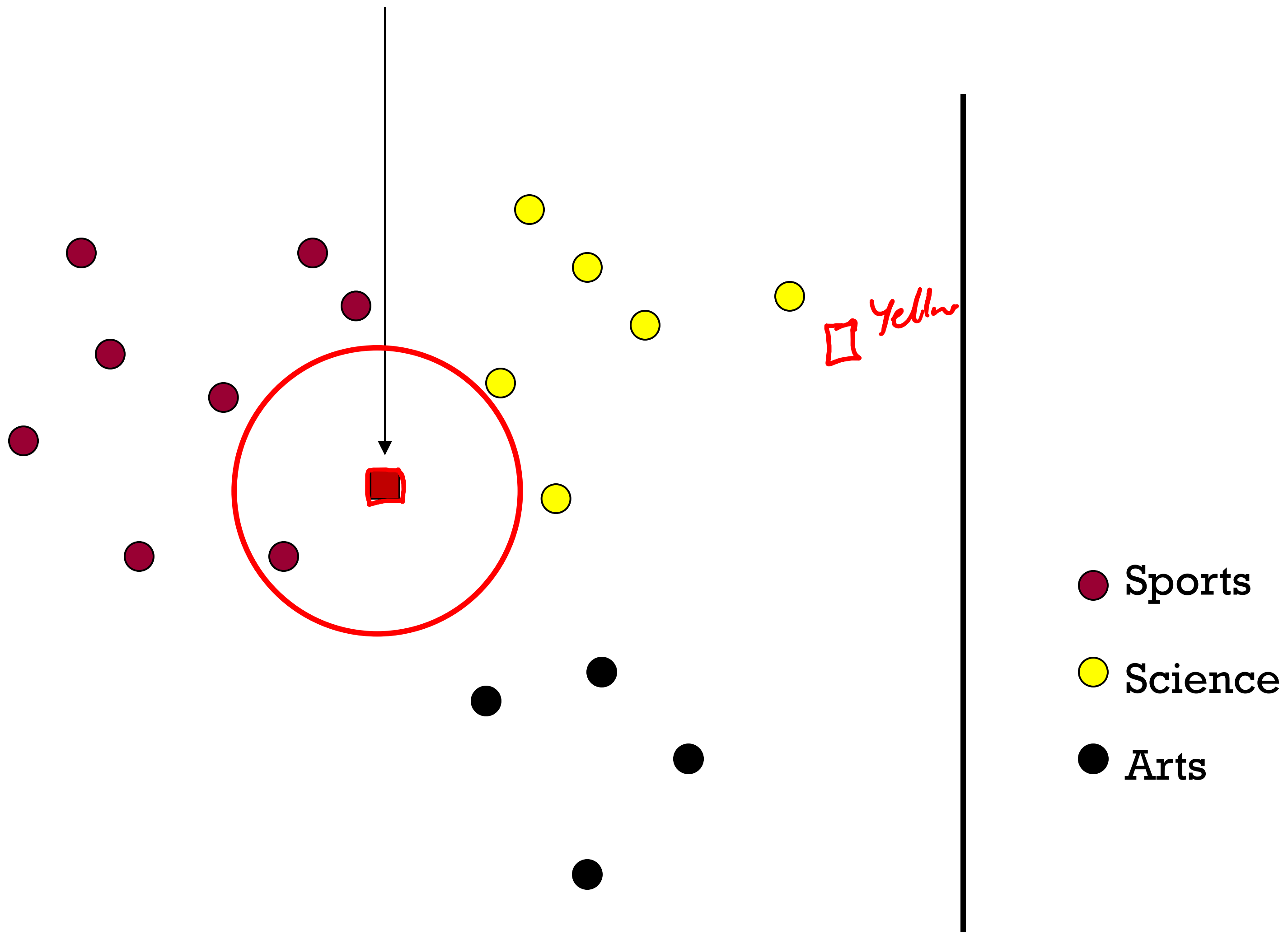
k_y → # training pts of class y amongst k NNs of x

n_y → # total training pts of class y

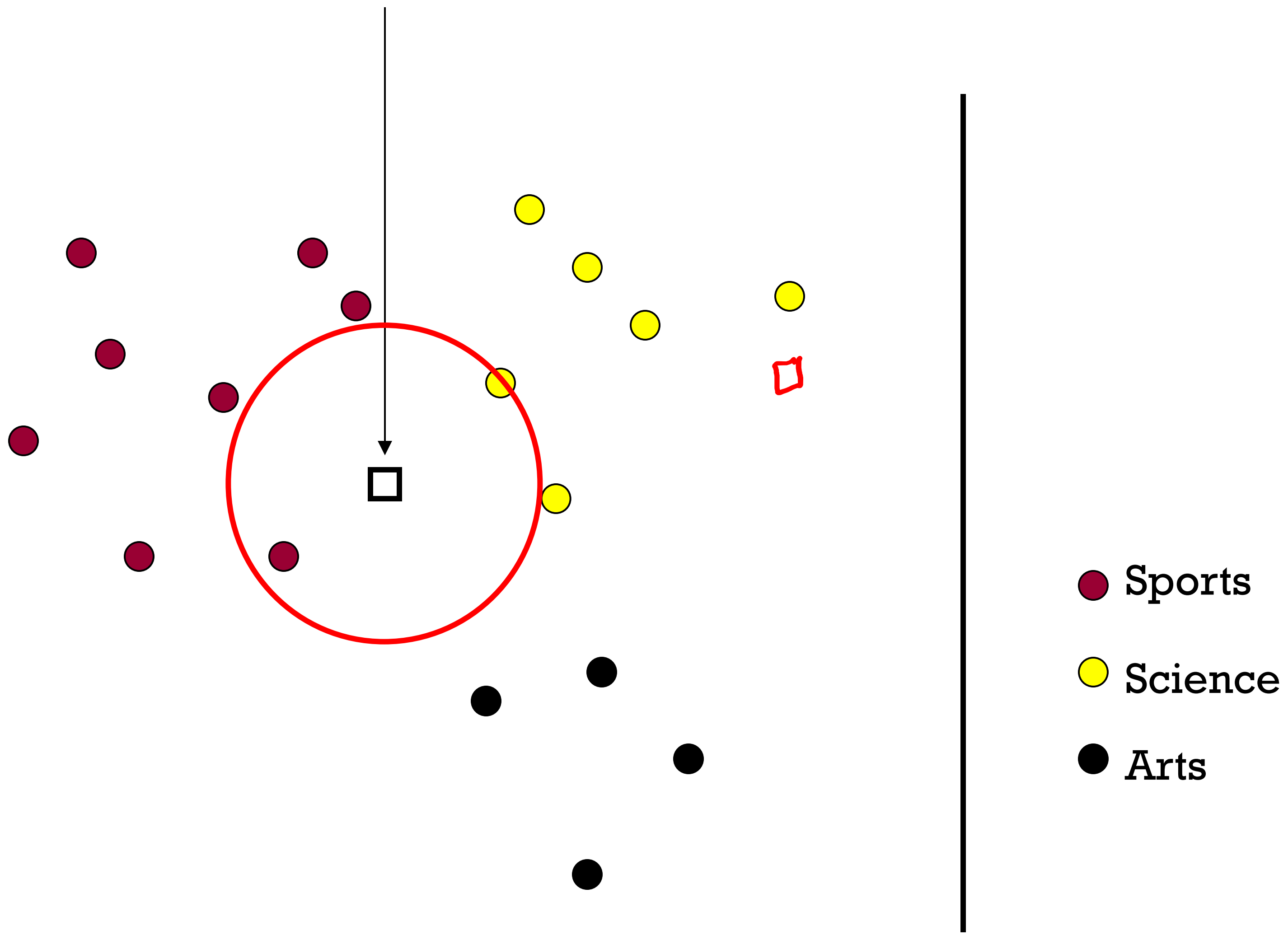
$$\sum_y k_y = k$$

$$\widehat{P}(y) = \frac{n_y}{n}$$

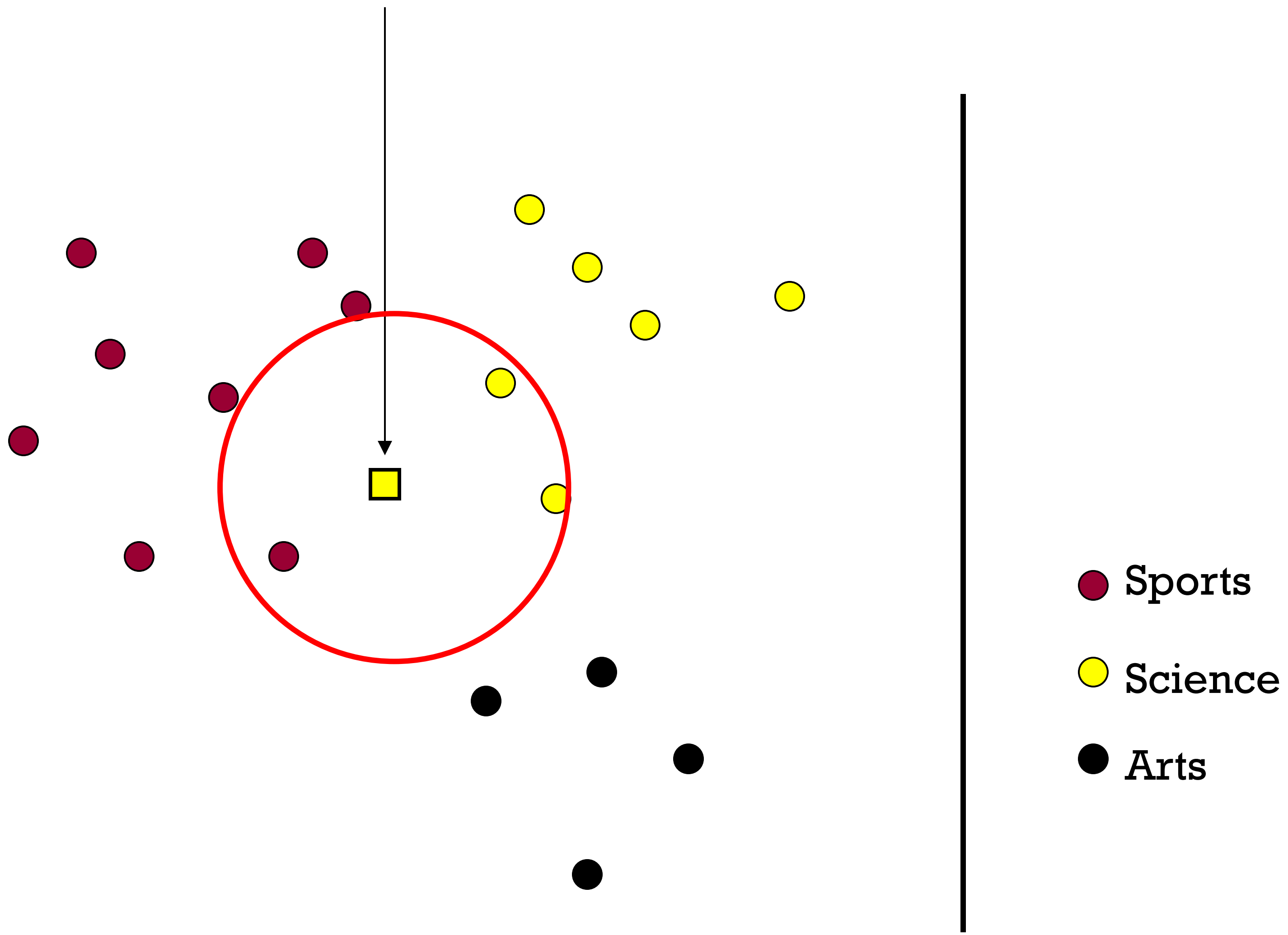
1-Nearest Neighbor (kNN) classifier



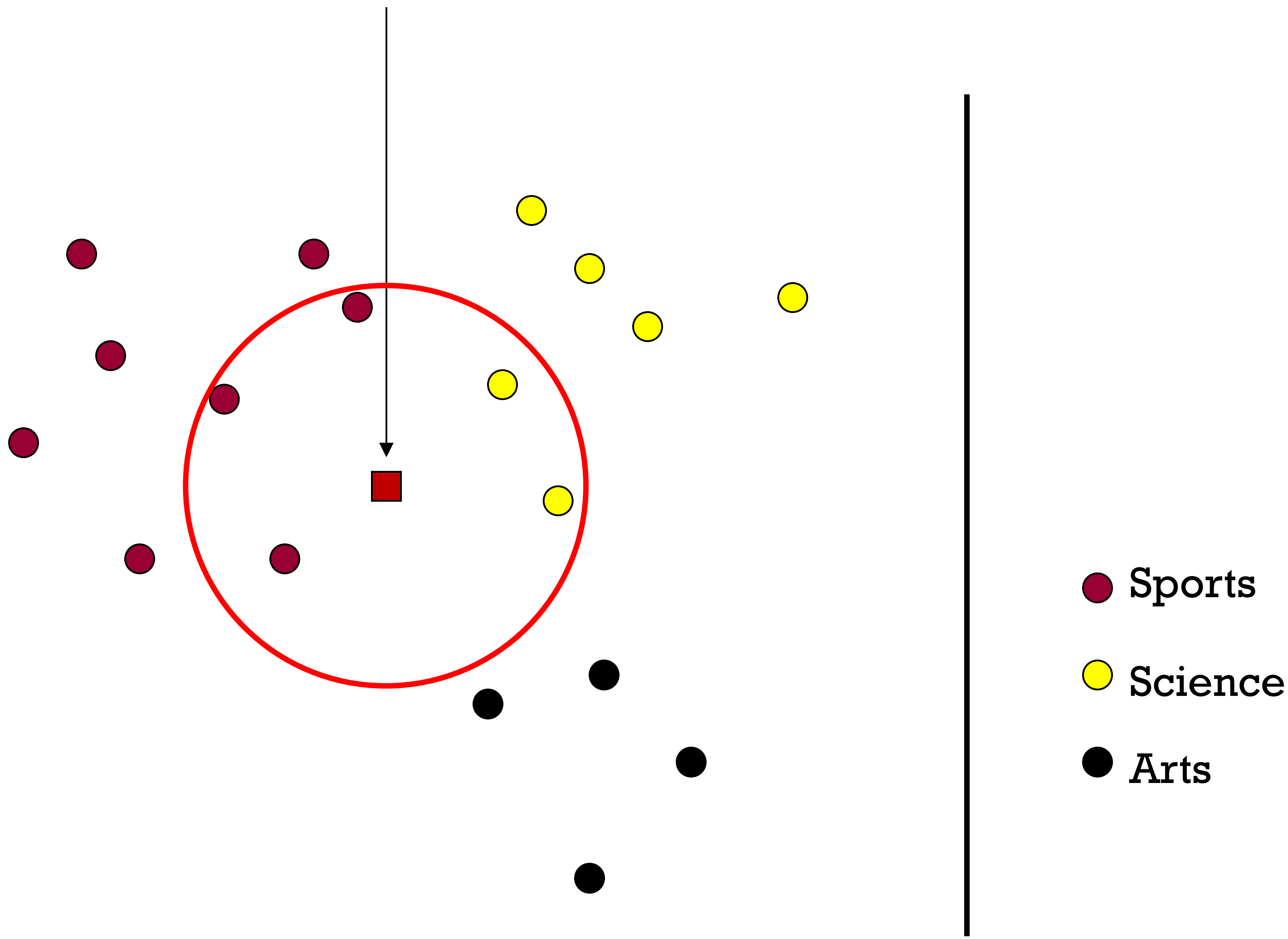
2-Nearest Neighbor (kNN) classifier



3-Nearest Neighbor (kNN) classifier

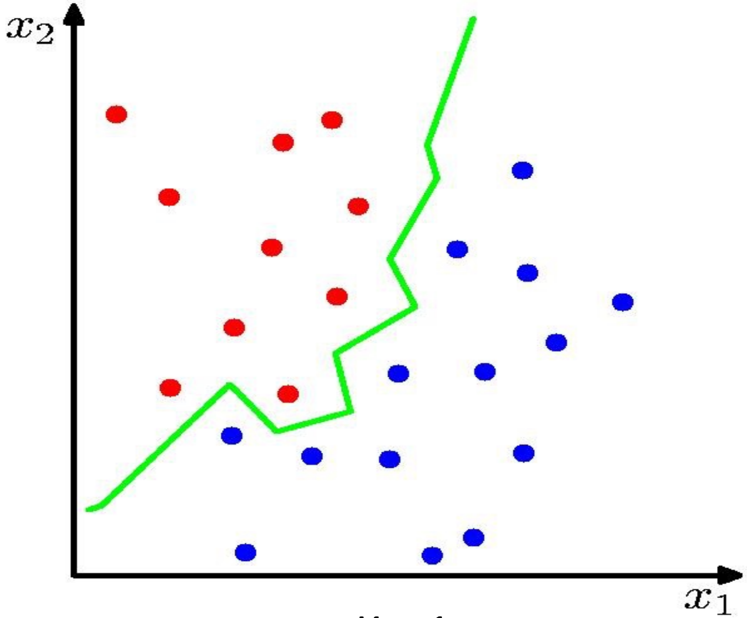


5-Nearest Neighbor (kNN) classifier



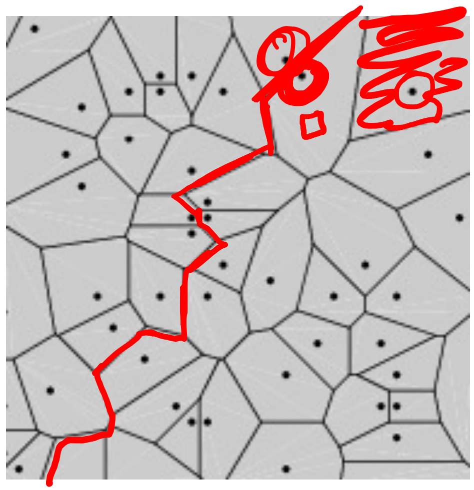
What is the best k?

1-NN classifier decision boundary



K = 1

Voronoi Diagram

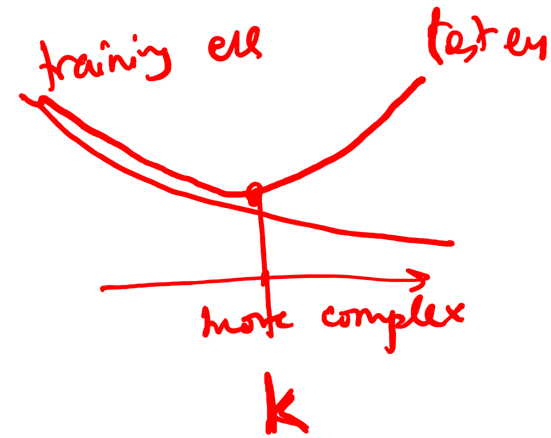


As k increases, boundary becomes smoother (less jagged).

What is the best k?

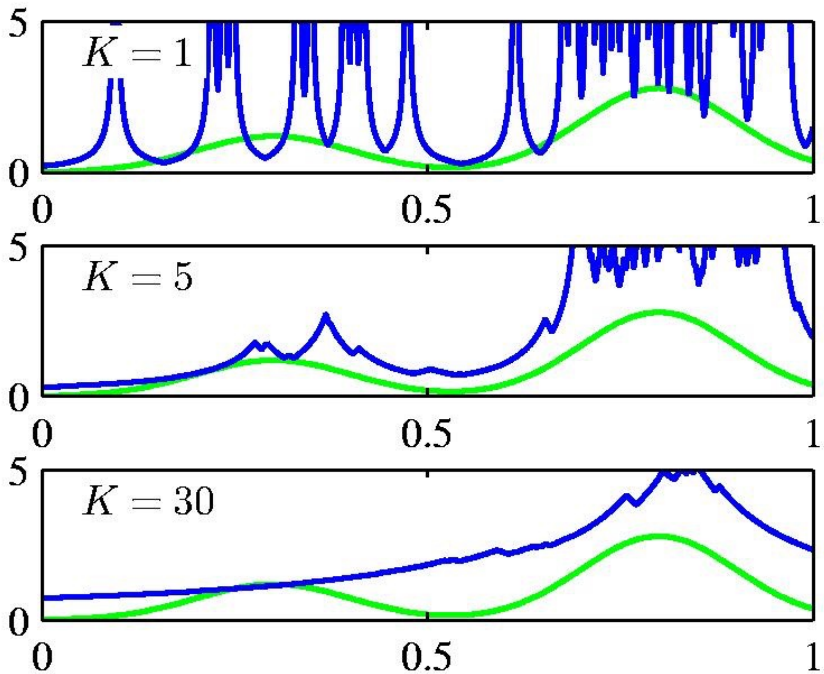
Approximation vs. Stability (aka Bias vs Variance) Tradeoff

- Larger K => predicted label is more stable
- Smaller K => predicted label can approximate best classifier well given enough data

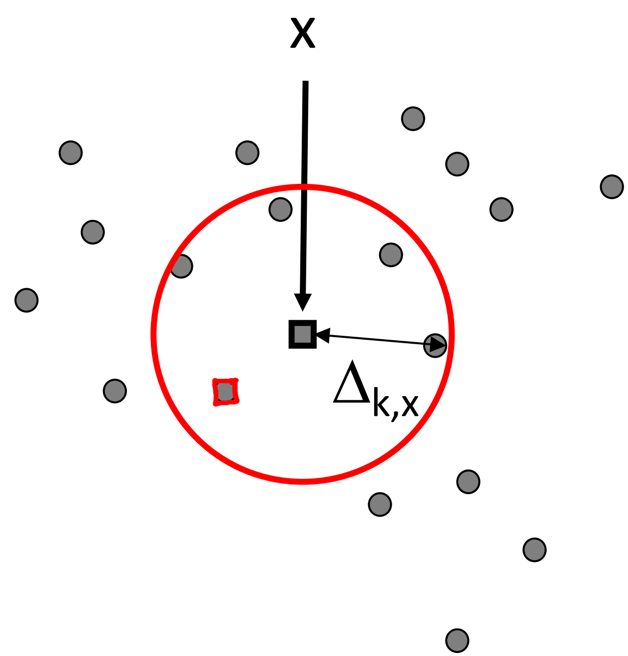


k-NN density estimation

$$\widehat{p}(x) = \frac{k}{n\Delta_{k,x}}$$



k acts as a smoother.



Not very popular for density estimation – spiked estimates

Histogram density estimate

Partition the feature space into distinct bins with widths Δ_i and count the number of observations, n_i, in each bin.

$$\widehat{p}(x) = \frac{n_i}{n\Delta_i} \mathbf{1}_{x \in \text{Bin}_i}$$

"Local relative frequency"

- Often, the same width is used for all bins, $\Delta_i = \Delta$.
- Δ acts as a smoothing parameter.

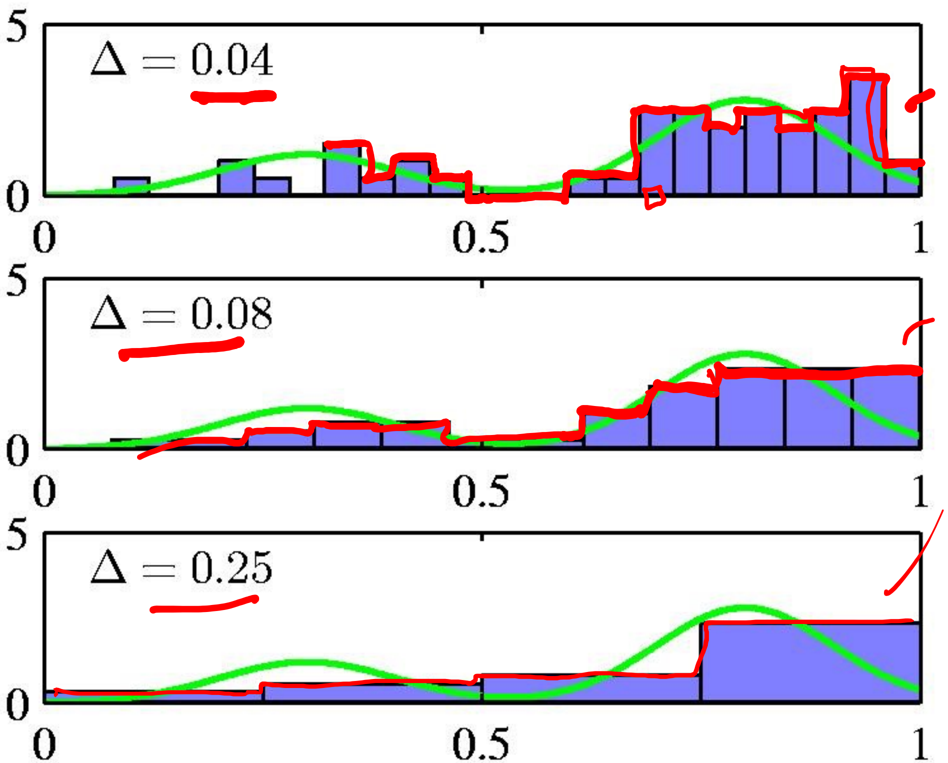
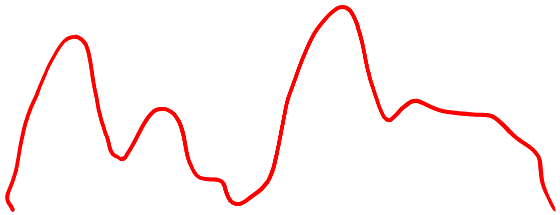


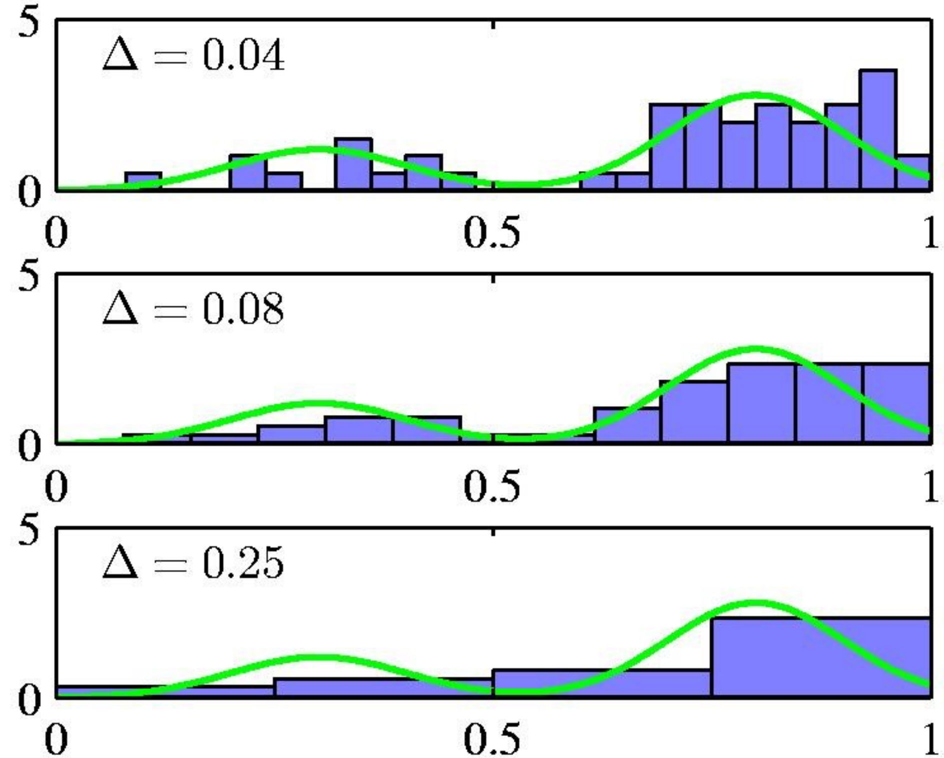
Image src: Bishop book

Effect of histogram bin width

$$\widehat{p}(x) = \frac{n_i}{n\Delta} \mathbf{1}_{x \in \text{Bin}_i}$$



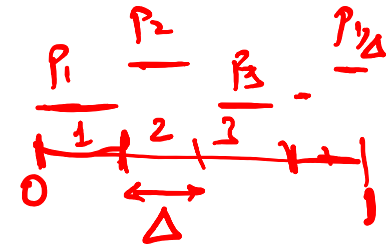
bins = 1/Δ



Small Δ, large #bins
Good fit but unstable
(few points per bin)
"**Small bias**, **Large variance**"

Large Δ, small #bins
Poor fit but stable
(many points per bin)
"**Large bias**, **Small variance**"

Histogram as MLE



- Underlying model – density is constant on each bin

  Parameters p_j : density in bin j

  Note $\sum_j p_j = 1/\Delta$ since $\int p(x)dx = 1$

- Maximize likelihood of data under probability model with parameters p_j

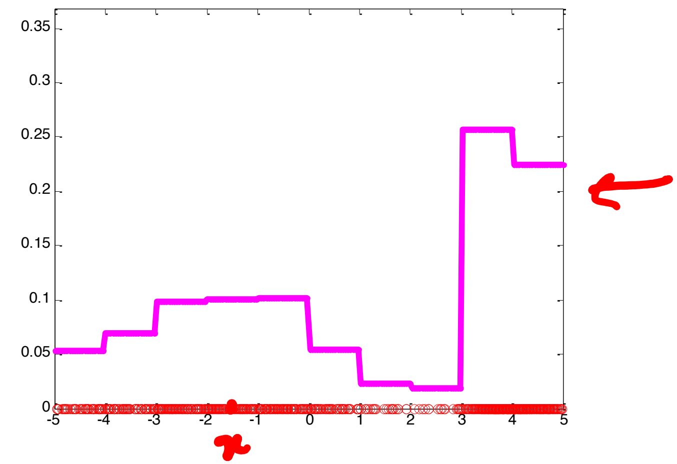
$$\hat{p}(x) = \arg\max_{\{p_j\}} P(X_1, \ldots, X_n; \{p_j\}_{j=1}^{1/\Delta}) \quad \text{s.t.} \quad \sum_j p_j = 1/\Delta$$

- Show that histogram density estimate is MLE under this model = Categorical($p_1\Delta$, $p_2\Delta$, $p_3\Delta$,…)

Kernel density estimate

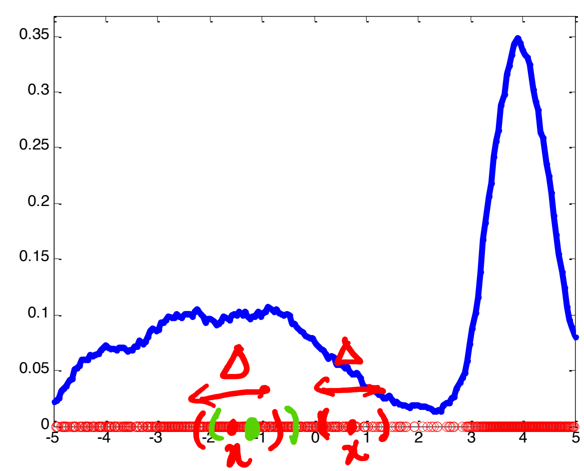
- Histogram – blocky estimate

$$\widehat{p}(x) = \frac{1}{\Delta} \frac{\sum_{j=1}^{n} \mathbf{1}_{X_j \in \text{Bin}_x}}{n}$$



- Kernel density estimate aka "Parzen/moving window method"

$$\widehat{p}(x) = \frac{1}{\Delta} \frac{\sum_{j=1}^{n} \mathbf{1}_{||X_j - x|| \leq \Delta}}{n}$$



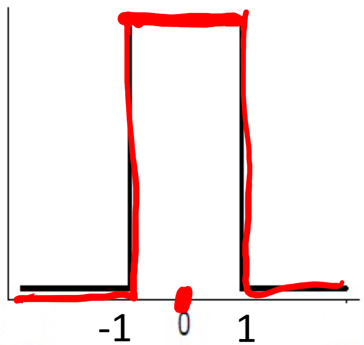
Kernel density estimate

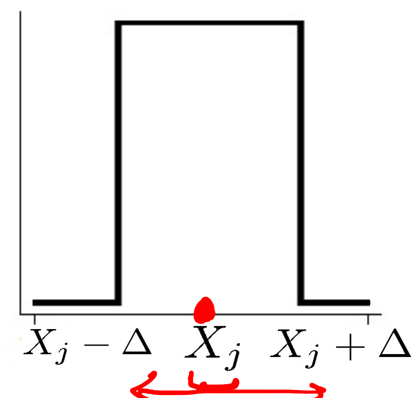
- $\widehat{p}(x) = \frac{1}{\Delta} \frac{\sum_{j=1}^{n} K\left(\frac{X_j - x}{\Delta}\right)}{n}$ more generally

$$K\left(\frac{X_j - x}{\Delta}\right)$$

boxcar kernel :

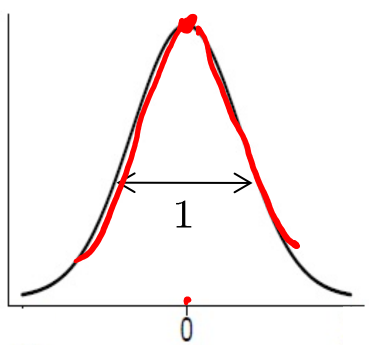
$$K(x) = \frac{1}{2} I(x),$$

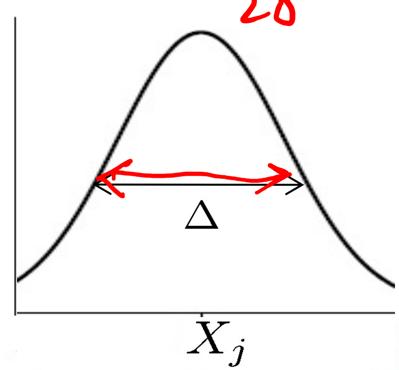




Gaussian kernel :

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

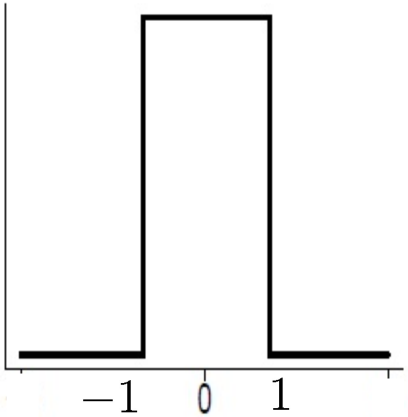




Kernels

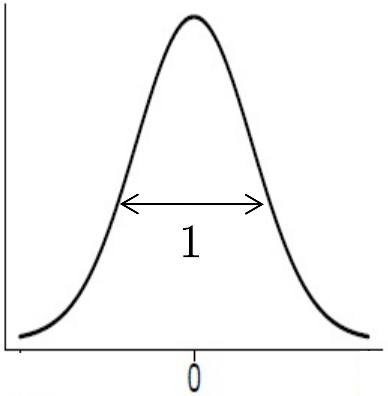
boxcar kernel :

$$K(x) = \frac{1}{2} I(x),$$



Gaussian kernel :

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$



Any kernel function that satisfies

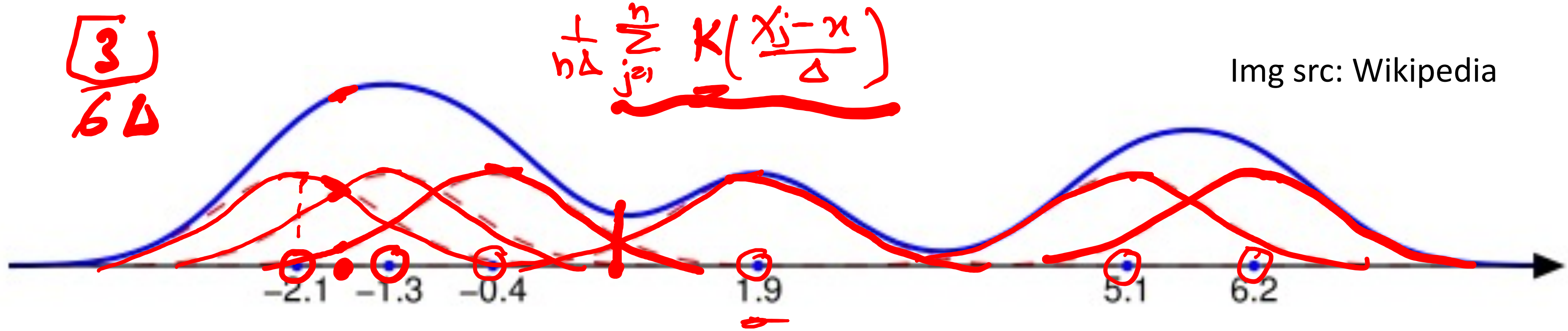
$$K(x) \geq 0,$$

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

Kernel density estimation

- Place small "bumps" at each data point, determined by the kernel function.
- The estimator consists of a (normalized) "sum of bumps”.

Img src: Wikipedia



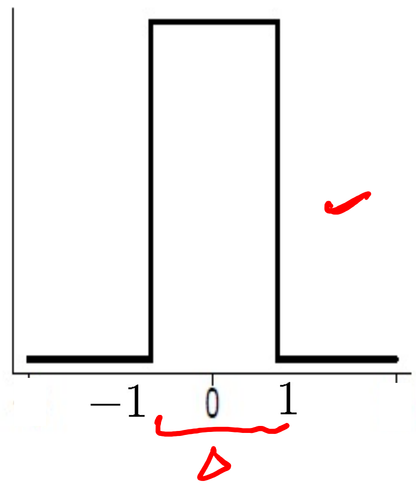
Gaussian bumps (red) around six data points and their sum (blue)

- Note that where the points are denser the density estimate will have higher values.

Choice of Kernels

boxcar kernel :

$$K(x) = \frac{1}{2} I(x),$$

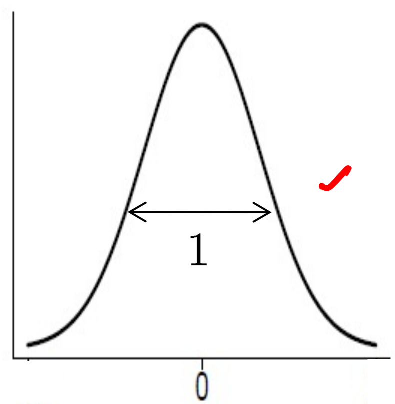


Finite support

– only need local points to compute estimate

Gaussian kernel :

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$



Infinite support

- need all points to compute estimate
- But quite popular since smoother

Choice of kernel bandwidth

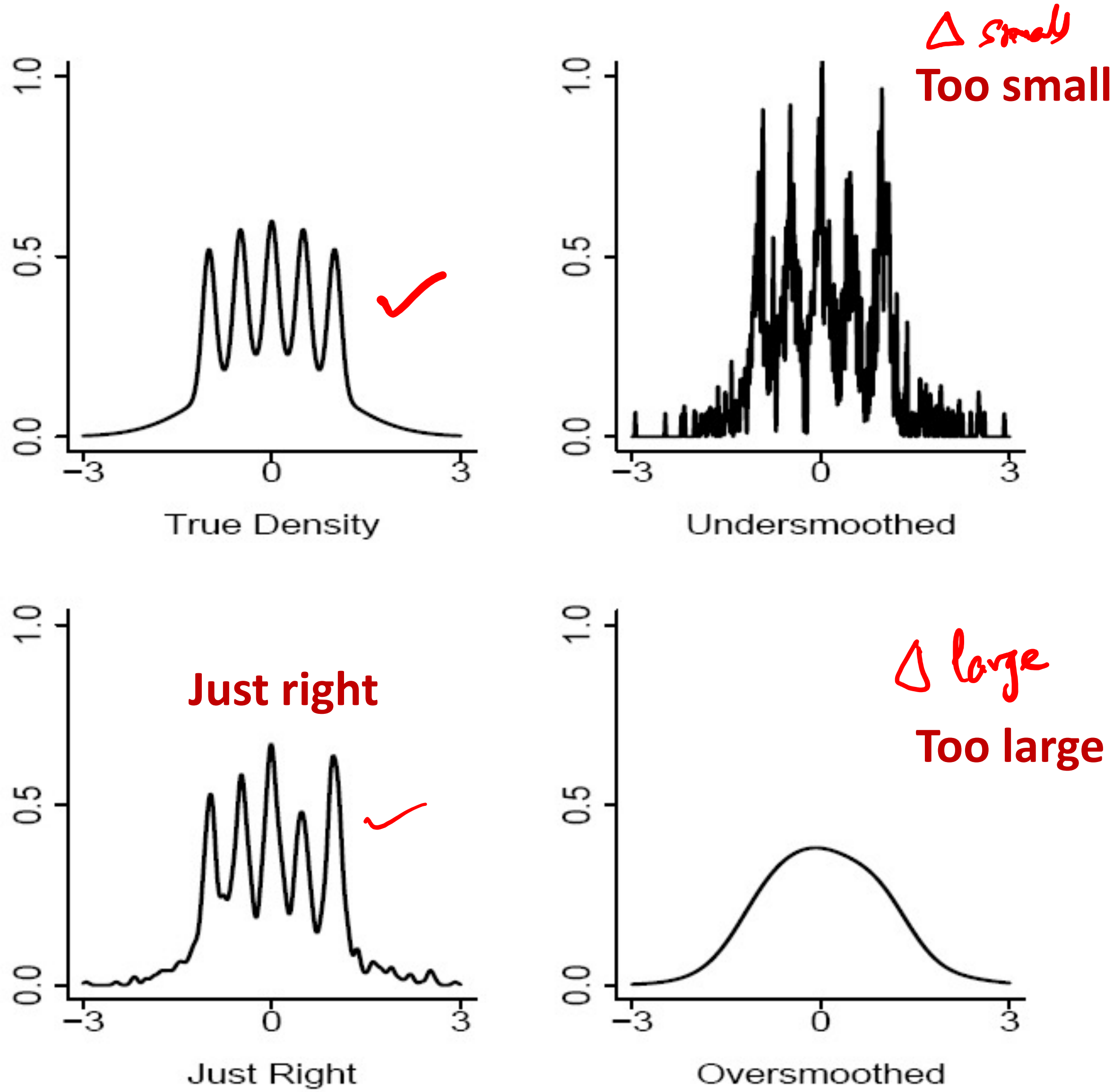
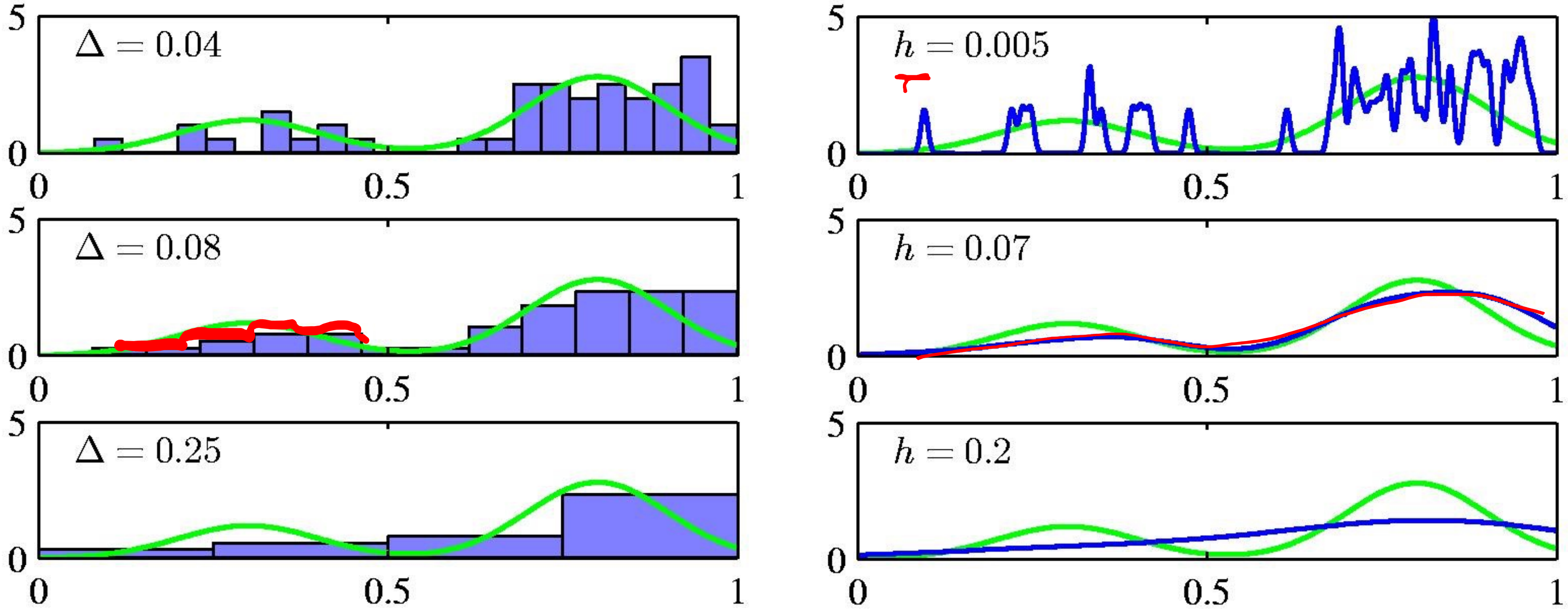


Image Source: Larry's book – All of Nonparametric Statistics

Bart-Simpson Density

Histograms vs. Kernel density estimation



$\Delta = $ h acts as a smoother.

Nonparametric density estimation

non-overlapping bins

- Histogram $$\widehat{p}(x) = \frac{n_i}{n\Delta} \mathbf{1}_{x \in \text{Bin}_i}$$
- Kernel density est $$\widehat{p}(x) = \frac{n_x}{n\Delta}$$

Fix Δ, estimate number of points within Δ of x (n_i or n_x) from data

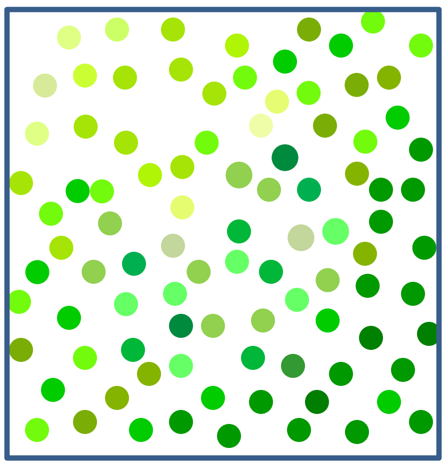
Fix $n_x = k$, estimate Δ from data (volume of ball around x that contains k training pts)

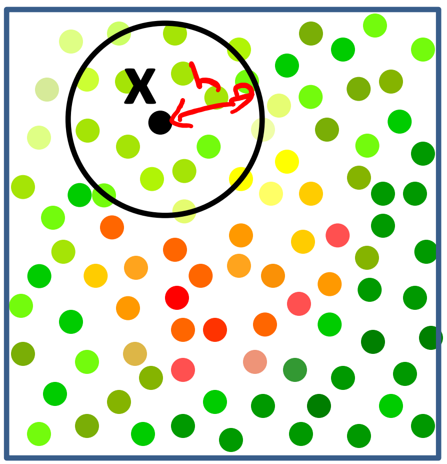
- k-NN density est $$\widehat{p}(x) = \frac{k}{n\Delta_{k,x}}$$

Local Kernel Regression

- What is the temperature in the room?

at location x?





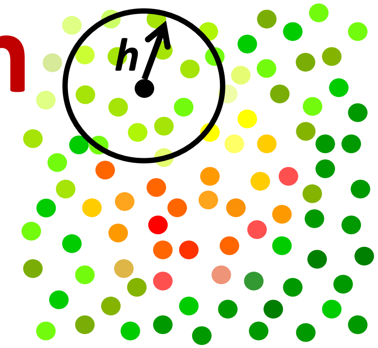
$$\widehat{T} = \frac{1}{n} \sum_{i=1}^{n} Y_i$$

Average

$$\widehat{T}(x) = \frac{\sum_{i=1}^{n} Y_i \mathbf{1}_{||X_i - x|| \leq h}}{\sum_{i=1}^{n} \mathbf{1}_{||X_i - x|| \leq h}}$$

"Local" Average

Local Kernel Regression



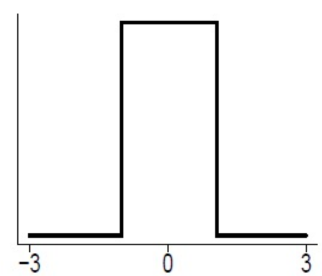
- Nonparametric estimator
- Nadaraya-Watson Kernel Estimator

$$\widehat{f}_n(X) = \sum_{i=1}^{n} w_i Y_i \quad \text{Where} \quad w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

- Weight each training point based on distance to test point
- Boxcar kernel yields local average

boxcar kernel :

$$K(x) = \frac{1}{2} I(x),$$



Choice of kernel bandwidth h

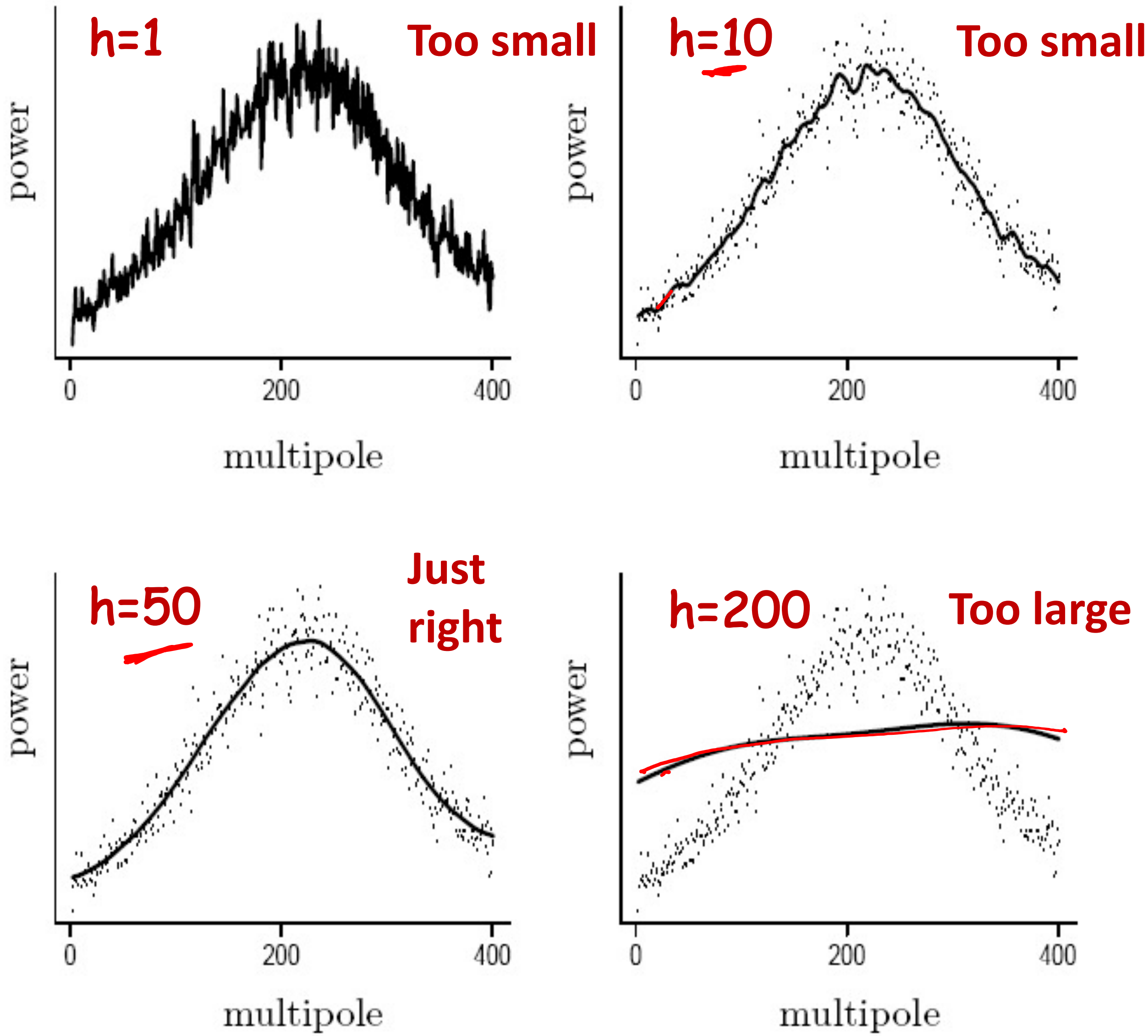


Image Source: Larry's book – All of Nonparametric Statistics

Kernel Regression as Weighted Least Squares

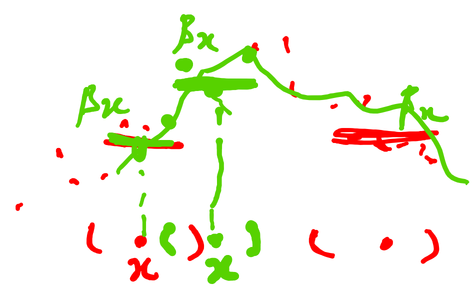
$$\min_f \sum_{i=1}^{n} w_i (f(X_i) - Y_i)^2$$

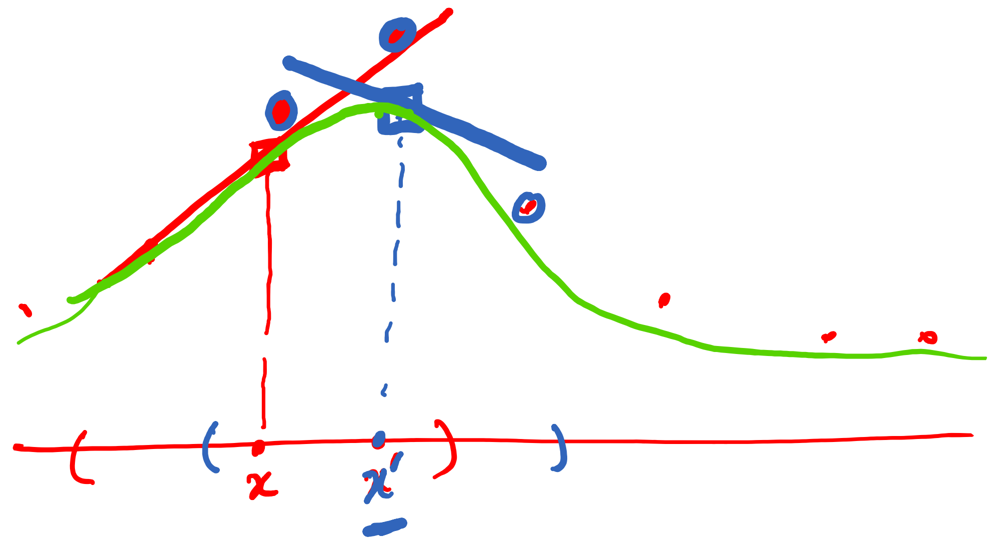
$$w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

Weighted Least Squares

Kernel regression corresponds to locally constant estimator obtained from (locally) weighted least squares

i.e. set $f(X_i) = \beta$ (a constant)




x
x'

Kernel Regression as Weighted Least Squares

set $f(X_i) = \beta$ (a constant)

WLS: $$\min_{\beta} \sum_{i=1}^{n} w_i(\beta - Y_i)^2$$

$f(X_i)$ = β → constant; $J(\beta)$

$$w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

$\beta \sum_{i=1}^{n} w_i = \sum_{i=1}^{n} w_i Y_i$

$$\frac{\partial J(\beta)}{\partial \beta} = 2\sum_{i=1}^{n} w_i(\beta - Y_i) = 0$$

Notice that $\sum_{i=1}^{n} w_i = 1$

$$\Rightarrow \widehat{f}_n(X) = \widehat{\beta}(X) = \sum_{i=1}^{n} w_i(X) Y_i$$

Local Linear/Polynomial Regression

$$\min_f \sum_{i=1}^{n} w_i(f(X_i) - Y_i)^2$$

X

$$w_i(X) = \frac{K\left(\frac{X - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X - X_i}{h}\right)}$$

Weighted Least Squares

Local Polynomial regression corresponds to locally polynomial estimator obtained from (locally) weighted least squares

i.e. set $f(X_i) = \beta_0 + \beta_1(X_i - X) + \frac{\beta_2}{2!}(X_i - X)^2 + \cdots + \frac{\beta_p}{p!}(X_i - X)^p$

(local polynomial of degree p around X)

Summary

- Non-parametric approaches

Four things make a nonparametric/memory/instance based/lazy learner:

1. *A distance metric, dist(x,X_i)* ✓ $\|x - X_i\|$
   Euclidean (and many more)
2. *How many nearby neighbors/radius to look at?*
   k, Δ/h ✓
3. *A weighting function (optional)*
   W based on kernel K ✓
4. *How to fit with the local points?*
   Average, Majority vote, Weighted average, Poly fit

Summary

- Parametric vs Nonparametric approaches
  - Nonparametric models place very mild assumptions on the data distribution and provide good models for complex data

    Parametric models rely on very strong (simplistic) modeling assumptions

  - Nonparametric models typically require storage and computation of the order of entire data set size.

    Parametric models, once fitted, are much more efficient in terms of storage and computation.