

# Non-parametric methods

**Kernel density estimate,  
kNN classifier, kernel regression**

Aarti Singh & Barnabas Poczos

Machine Learning 10-701  
Feb 13, 2014



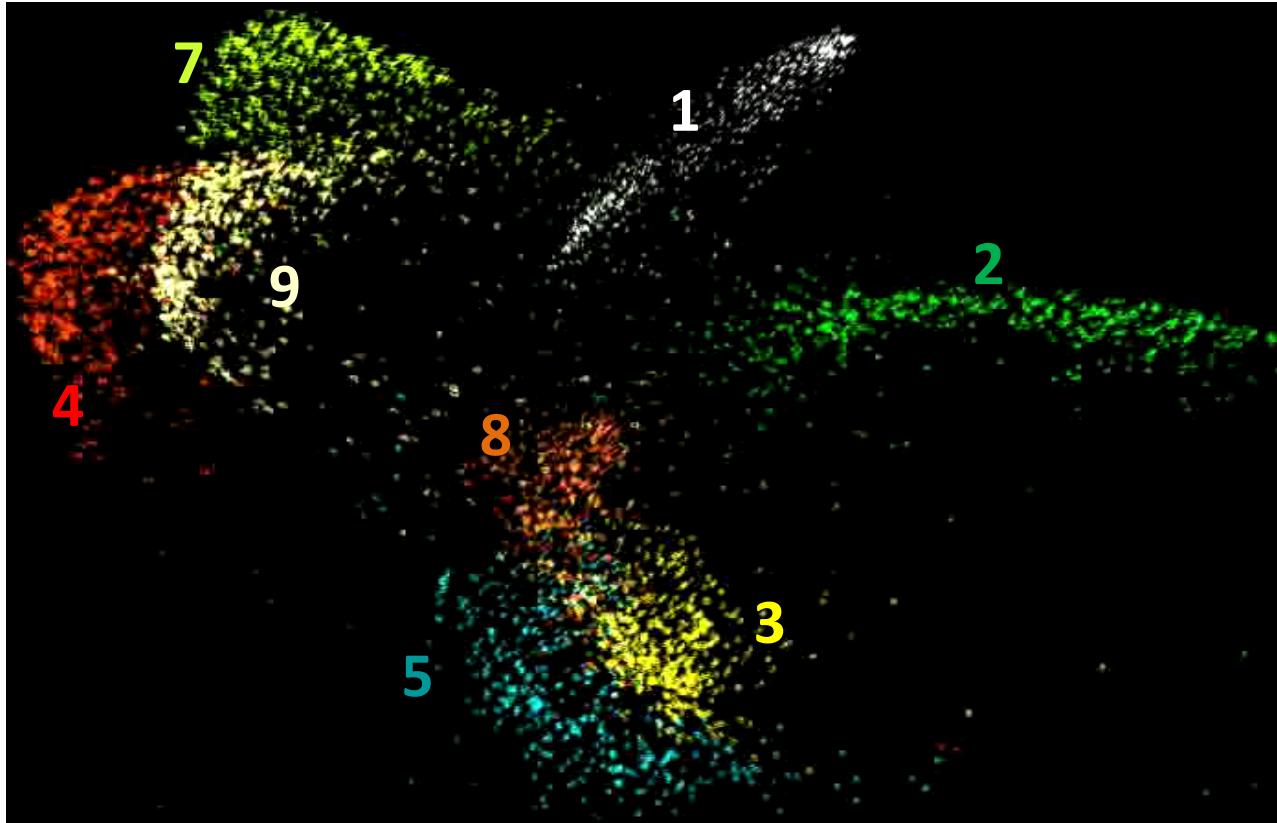
MACHINE LEARNING DEPARTMENT

Carnegie Mellon.  
School of Computer Science

# Parametric methods

- Assume some functional form (Gaussian, Bernoulli, Multinomial, logistic, Linear, Quadratic) for
  - $P(X_i|Y)$  and  $P(Y)$  as in Naïve Bayes
  - $P(Y|X)$  as in Logistic, Linear and Nonlinear regression, SVM
- Estimate parameters ( $\mu, \sigma^2, \theta, w, \beta$ ) using MLE/MAP and plug in
- **Pro** – need few data points to learn parameters
- **Con** – Strong distributional assumptions, not satisfied in practice

# Example



Hand-written digit images  
projected as points on a two-dimensional (nonlinear) feature spaces

# Non-Parametric methods

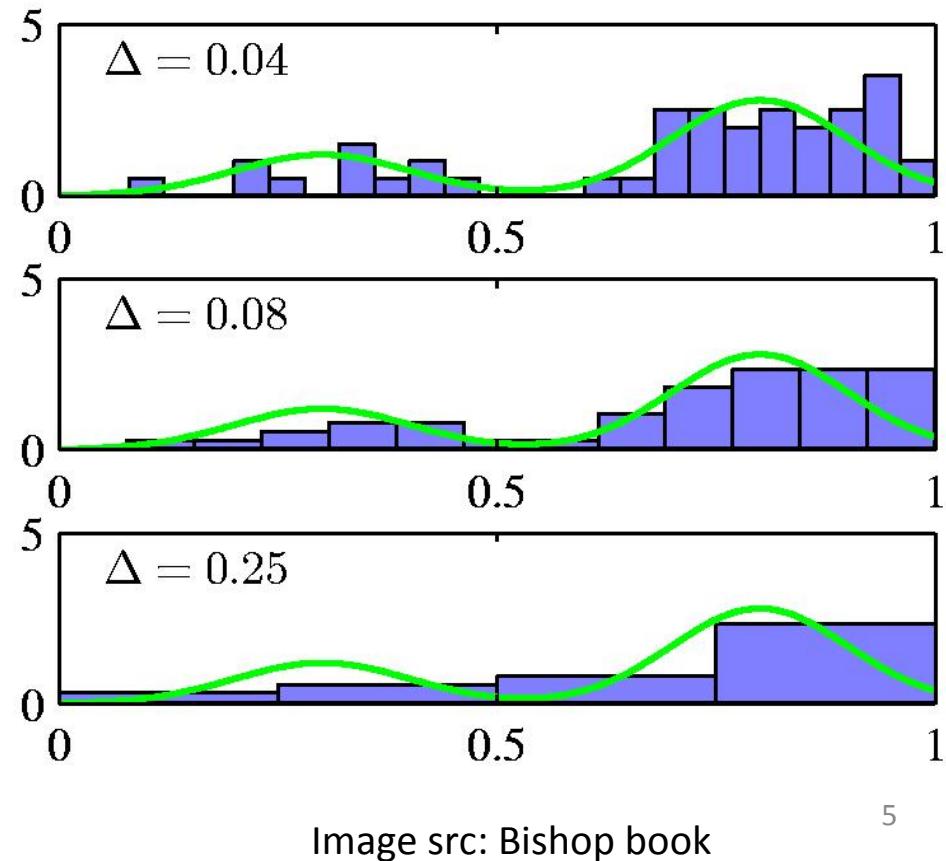
- Typically don't make any distributional assumptions
- As we have more data, we should be able to learn more complex models
- Let number of parameters scale with number of training data
- Today, we will see some nonparametric methods for
  - Density estimation
  - Classification
  - Regression

# Histogram density estimate

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

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

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

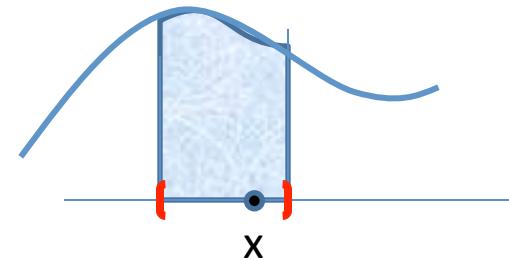


# Effect of histogram bin width

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

$$\# \text{ bins} = 1/\Delta$$

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



Bias of histogram density estimate:

$$\mathbb{E}[\hat{p}(x)] = \frac{1}{\Delta} P(X \in \text{Bin}_x) = \frac{1}{\Delta} \int_{z \in \text{Bin}_x} p(z) dz \approx \frac{p(x)\Delta}{\Delta} = p(x)$$



**Assuming density it roughly constant in each bin  
(holds true if  $\Delta$  is small)**

# Bias – Variance tradeoff

$$|\hat{p}(x) - p(x)| = |\hat{p}(x) - \mathbb{E}[\hat{p}(x)] + \mathbb{E}[\hat{p}(x)] - p(x)|$$



A horizontal blue bracket spans the entire expression  $|\hat{p}(x) - p(x)|$ . Below the bracket, the word "Variance" is centered under the first term  $|\hat{p}(x) - \mathbb{E}[\hat{p}(x)]|$ , and the word "Bias" is centered under the second term  $|\mathbb{E}[\hat{p}(x)] - p(x)|$ .

**Bias** – how close is the mean of estimate to the truth

**Variance** – how much does the estimate vary around mean

- Choice of bin-width  $\Delta$  or #bins  $\# \text{ bins} = 1/\Delta$ 
  - $\mathbb{E}[\hat{p}(x)] \approx p(x)$  if  $\Delta$  is small ( $p(x)$  approx constant per bin)
  - $\mathbb{E}[\hat{p}(x)] \approx \hat{p}(x)$  if  $\Delta$  is large (more data per bin,  
stable estimate)

Small  $\Delta$ , large #bins  “**Small bias, Large variance**”

Large  $\Delta$ , small #bins  “**Large bias, Small variance**”

# Choice of #bins

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

$$\# \text{ bins} = 1/\Delta$$

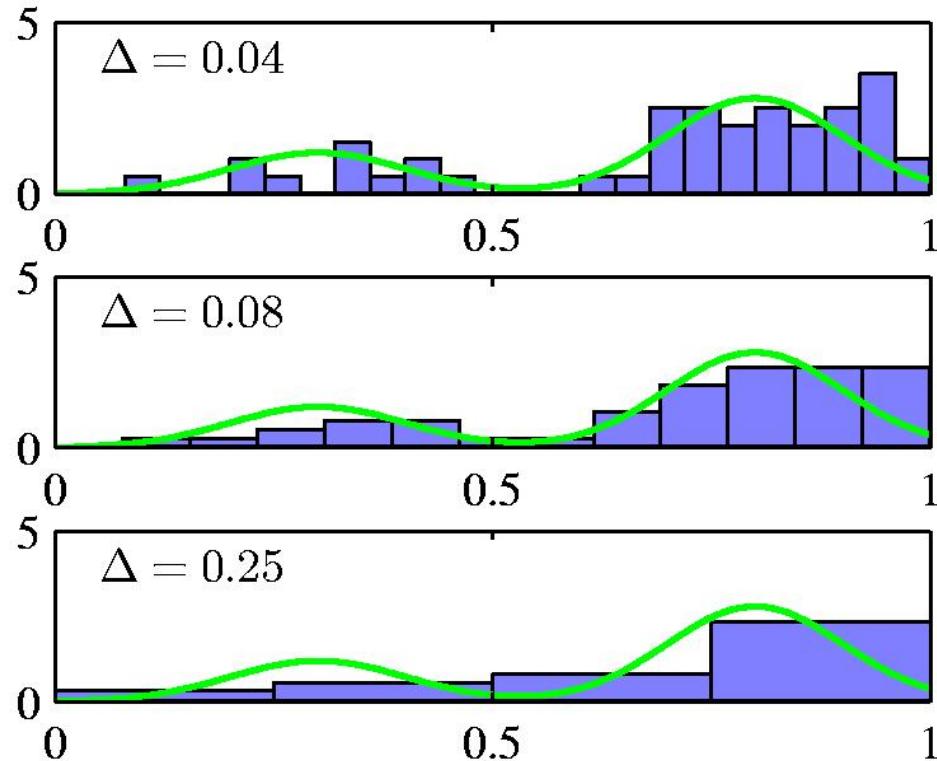


Image src: Bishop book

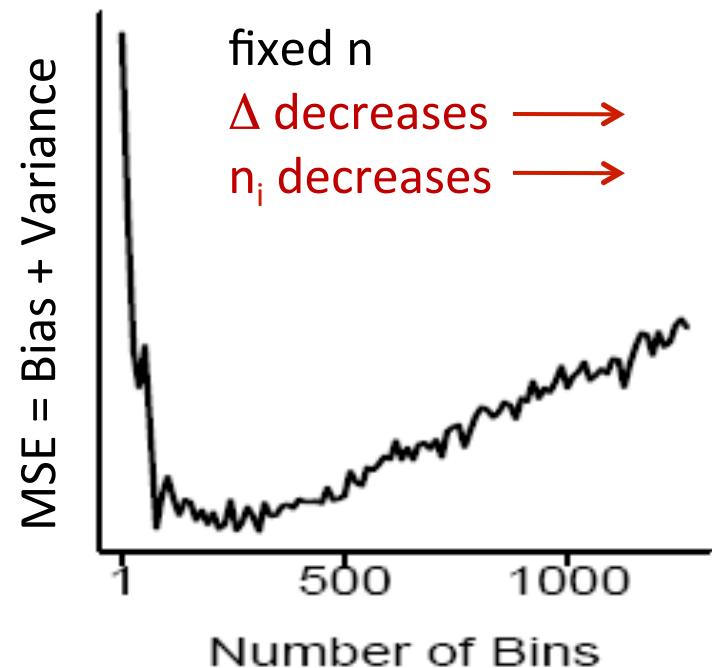


Image src: Larry book

# 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  $p_j$

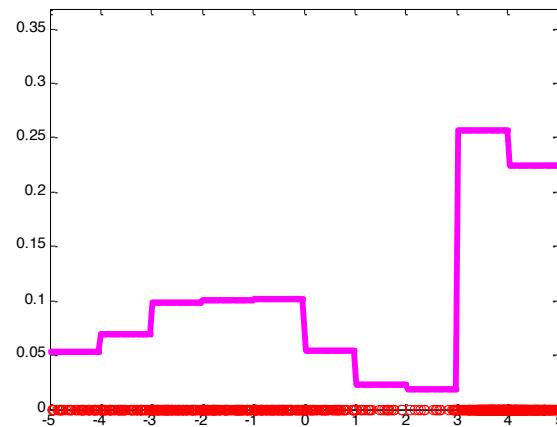
$$\hat{p}(x) = \arg \max_{\{p_j\}} P(X_1, \dots, X_n; \{p_j\}_{j=1}^{1/\Delta}) \quad \text{s.t.} \quad \sum_j p_j = 1/\Delta$$

- Show that histogram density estimate is MLE under this model – HW/Recitation

# Kernel density estimate

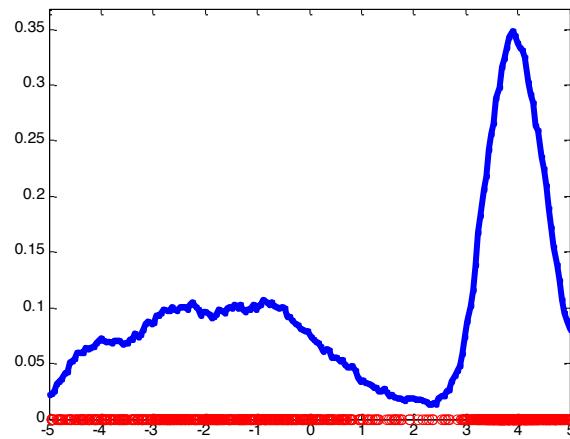
- Histogram – blocky estimate

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



- Kernel density estimate aka “Parzen/moving window method”

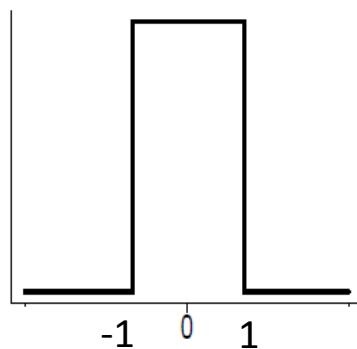
$$\hat{p}(x) = \frac{1}{\Delta} \frac{\sum_{j=1}^n \mathbf{1}_{||X_j - x|| \leq \Delta}}{n}$$



# Kernel density estimate

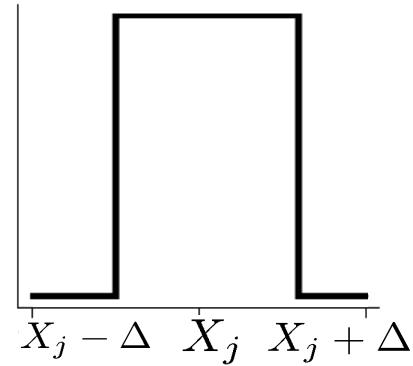
- $\hat{p}(x) = \frac{1}{\Delta} \frac{\sum_{j=1}^n K\left(\frac{X_j - x}{\Delta}\right)}{n}$  more generally

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



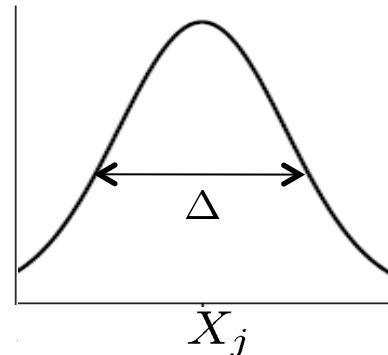
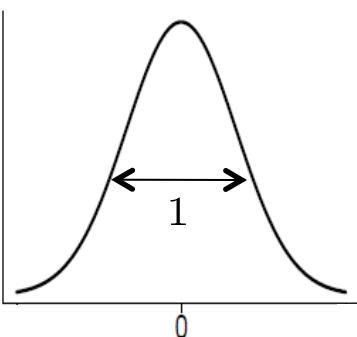
boxcar kernel :

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



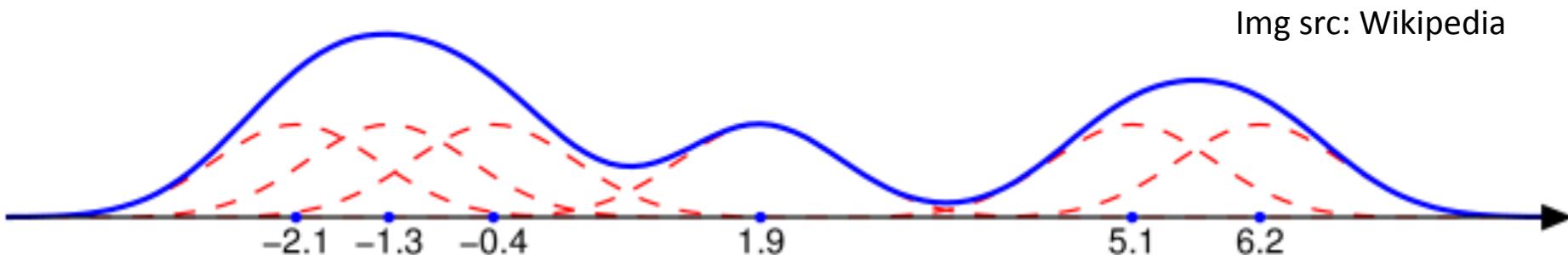
Gaussian kernel :

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



# Kernel density estimation

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



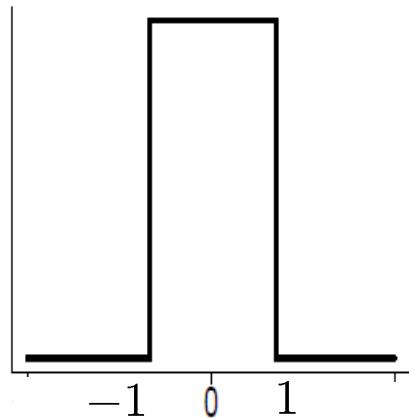
Gaussian bumps (red) around six data points and their sum (blue)

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

# Kernels

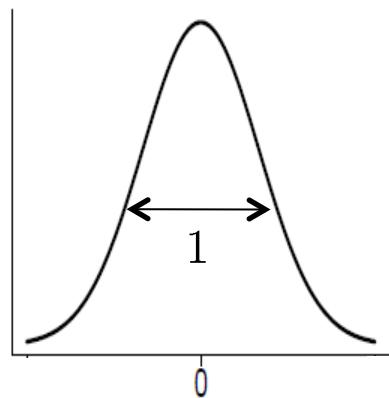
boxcar kernel :

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



Gaussian kernel :

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



Any kernel function that satisfies

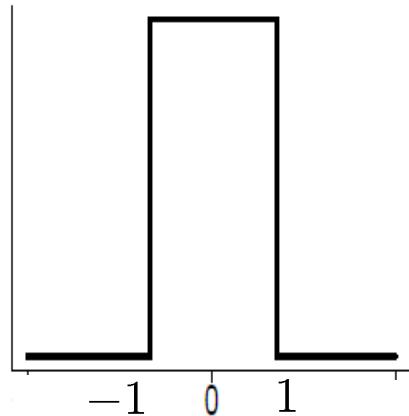
$$K(x) \geq 0,$$

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

# Kernels

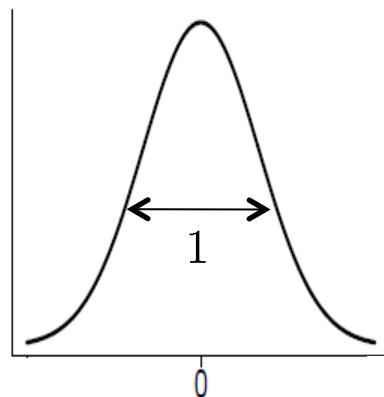
boxcar kernel :

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



Gaussian kernel :

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



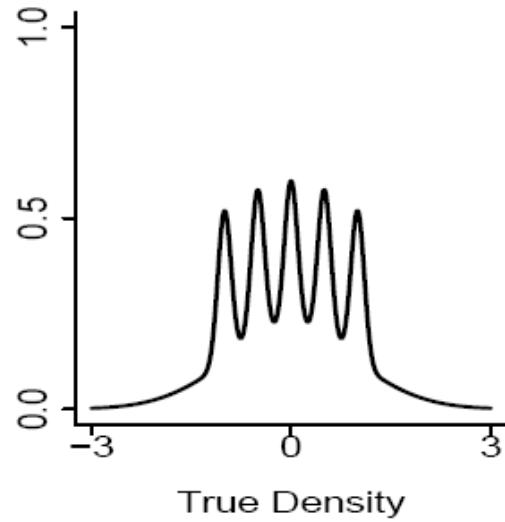
**Finite support**

- only need local points to compute estimate

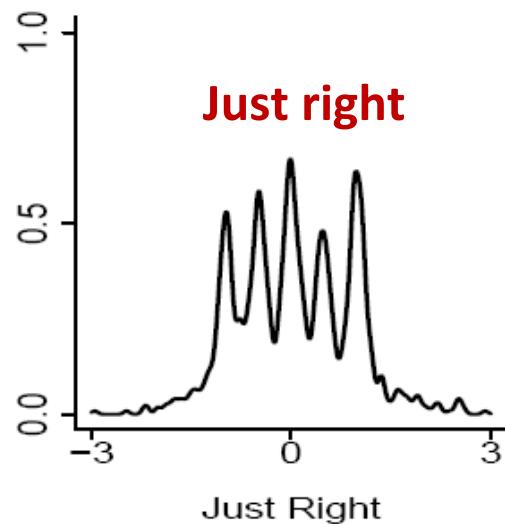
**Infinite support**

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

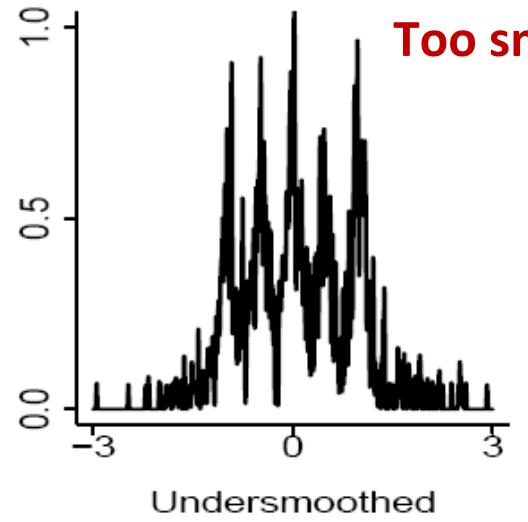
# Choice of kernel bandwidth



True Density



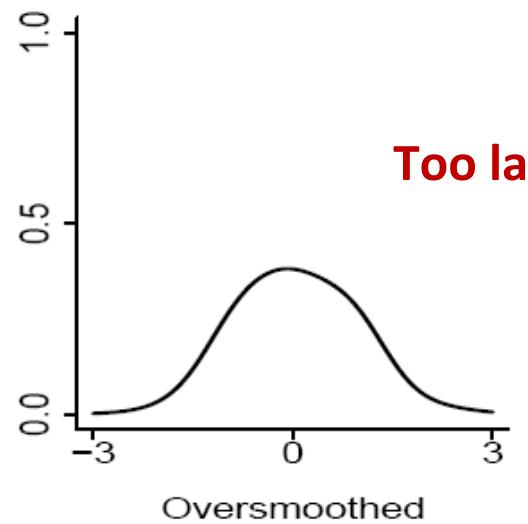
Just Right



Undersmoothed

Too small

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

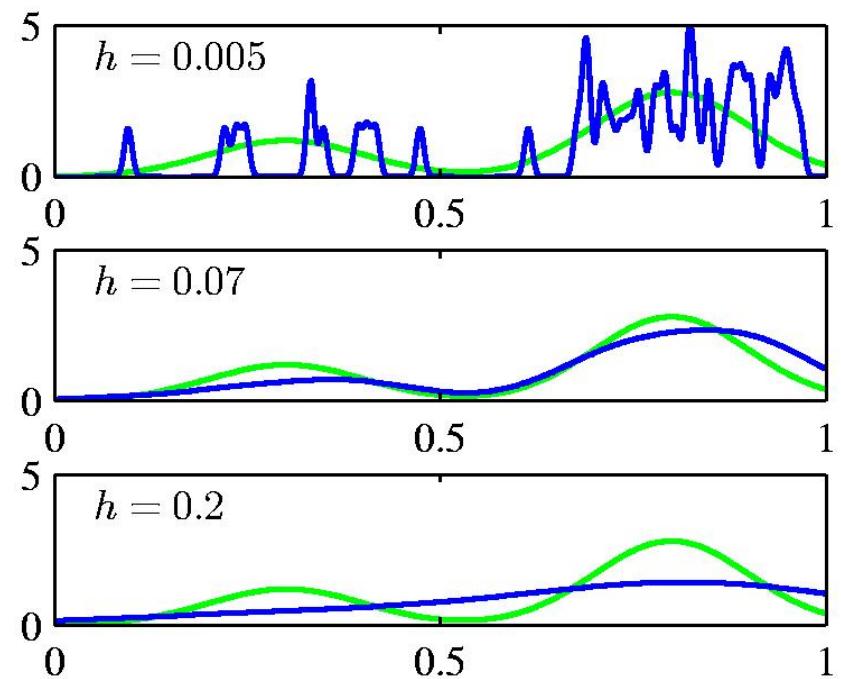
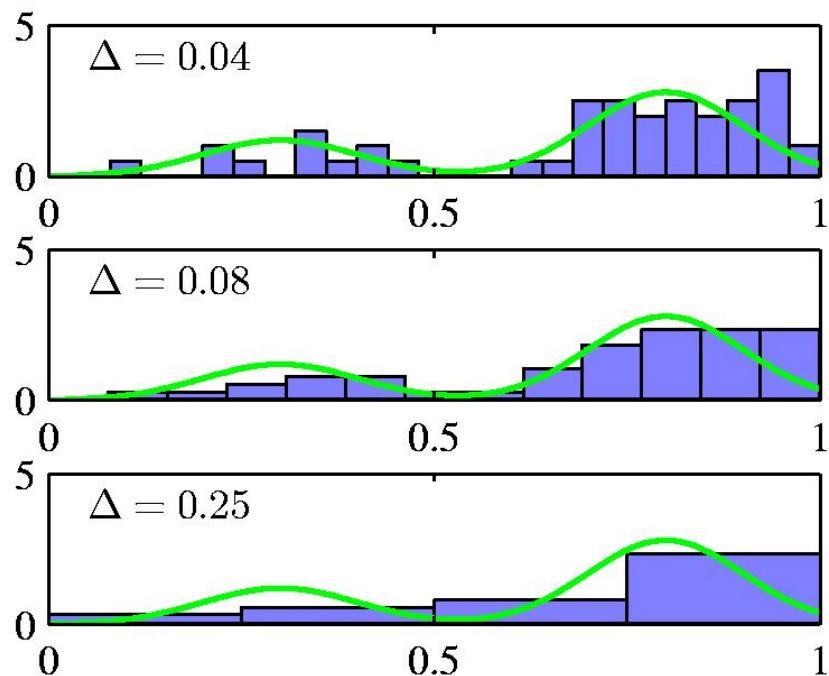


Oversmoothed

Too large

Bart-Simpson  
Density

# Histograms vs. Kernel density estimation



$\Delta = h$  acts as a smoother.

# k-NN (Nearest Neighbor) density estimation

- Histogram

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

- Kernel density est

$$\hat{p}(x) = \frac{n_x}{n\Delta}$$

Fix  $\Delta$ , estimate number of points within  $\Delta$  of  $x$  ( $n_i$  or  $n_x$ ) from data

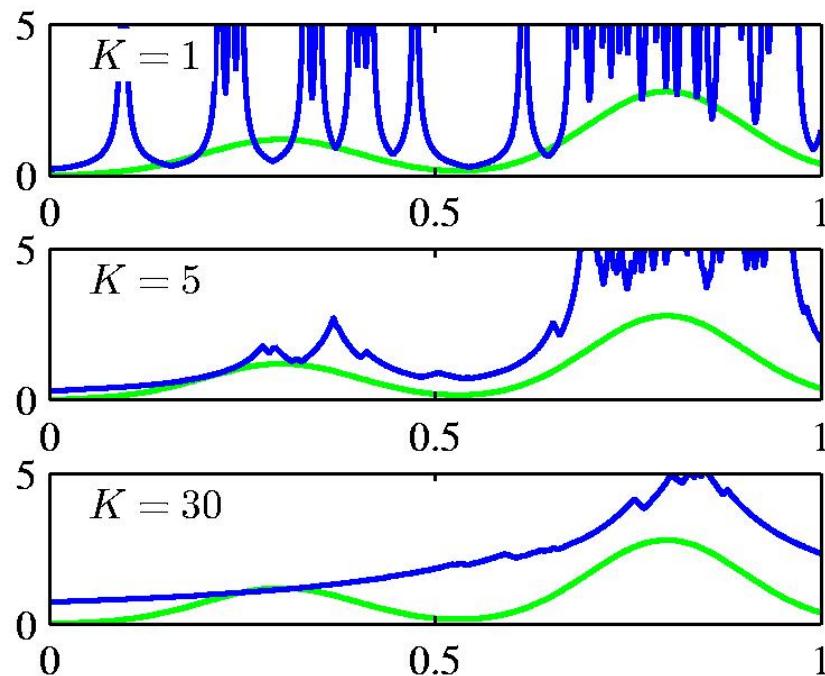
Fix  $n_x = k$ , estimate  $\Delta$  from data (volume of ball around  $x$  that contains  $k$  training pts)

- k-NN density est

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

# k-NN density estimation

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



$k$  acts as a smoother.

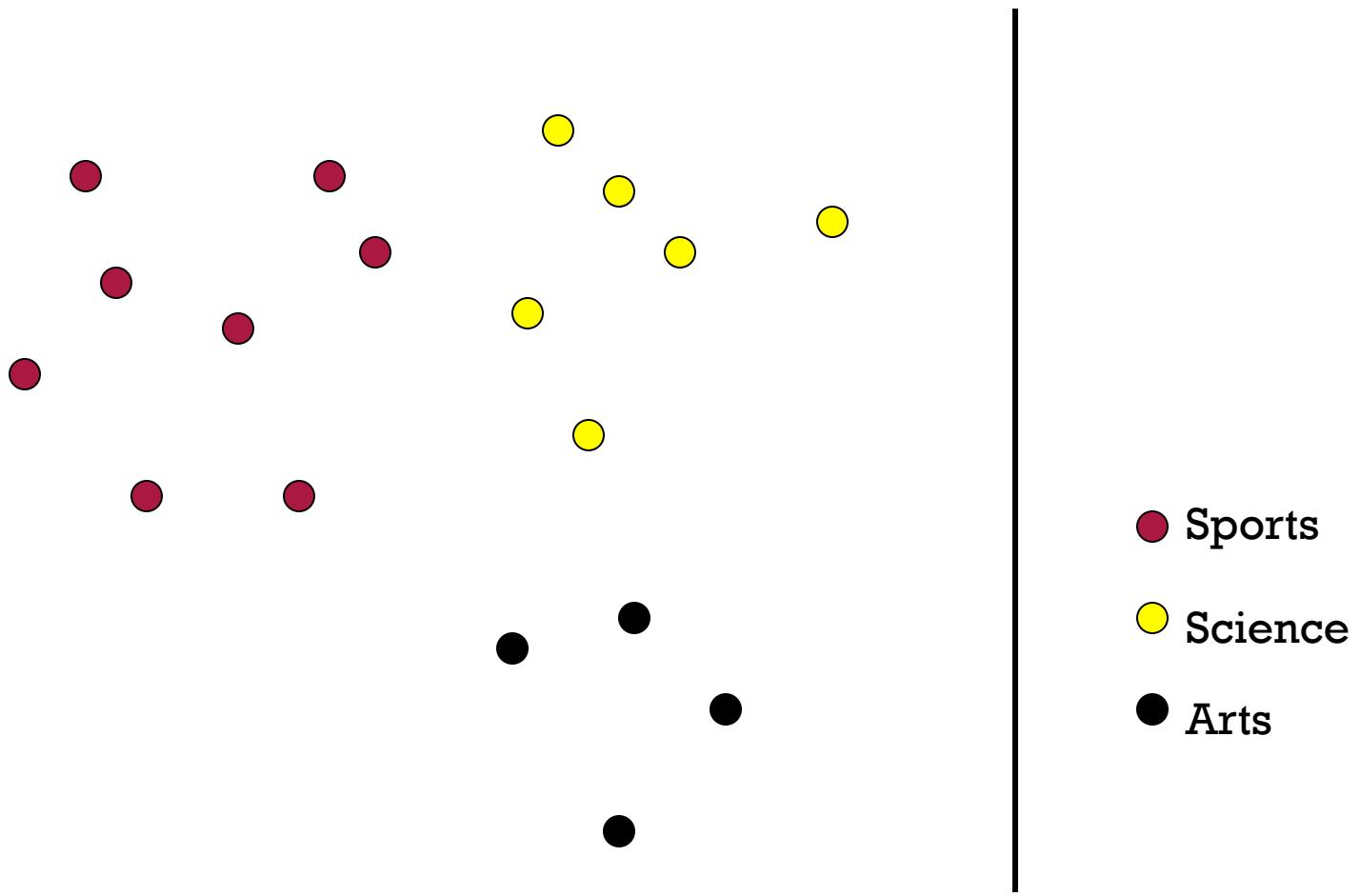
Not very popular for density estimation – spiked estimates

But a related version  
for classification quite popular

...

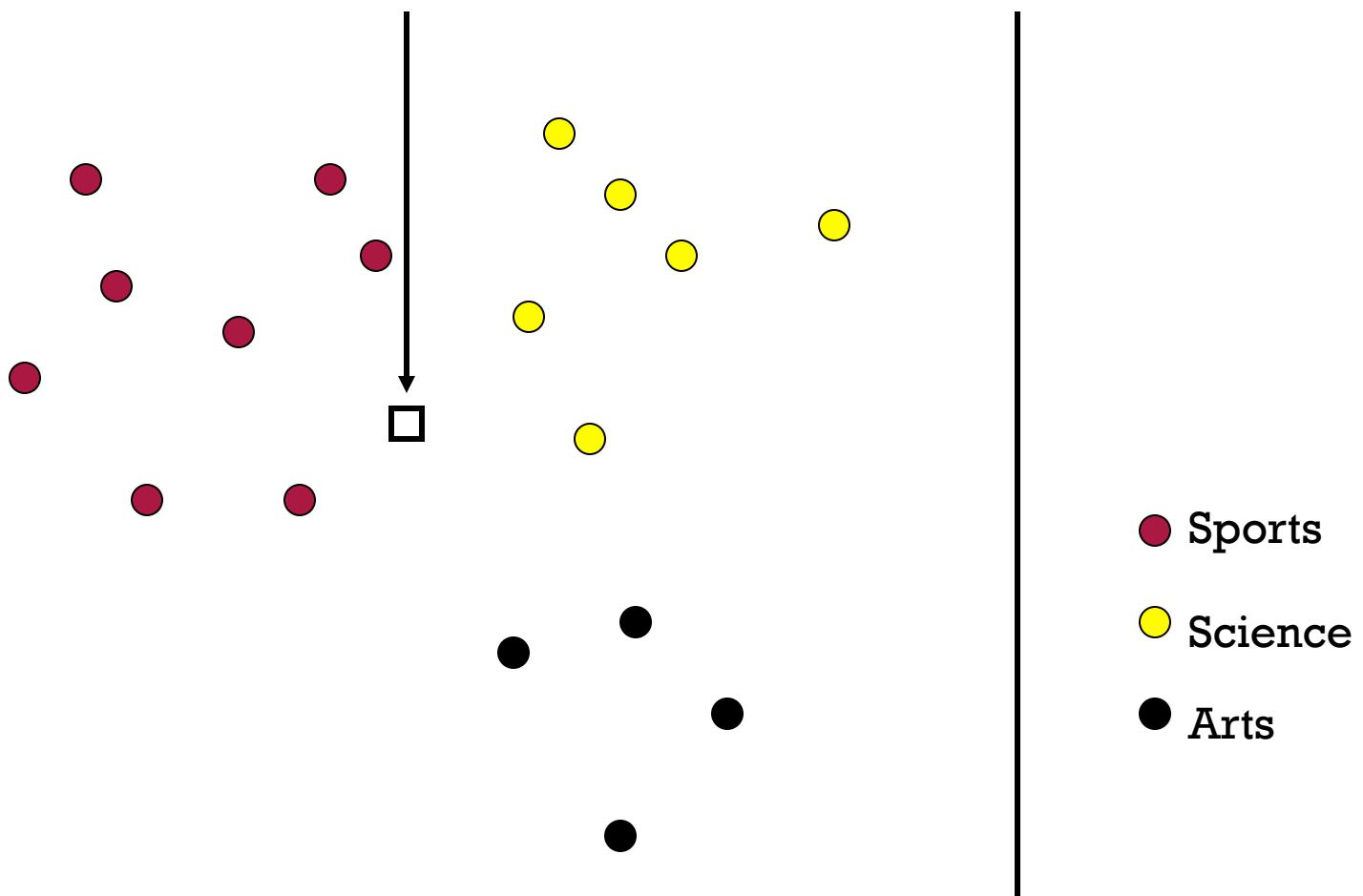
From  
Density estimation  
to  
Classification

# k-NN classifier

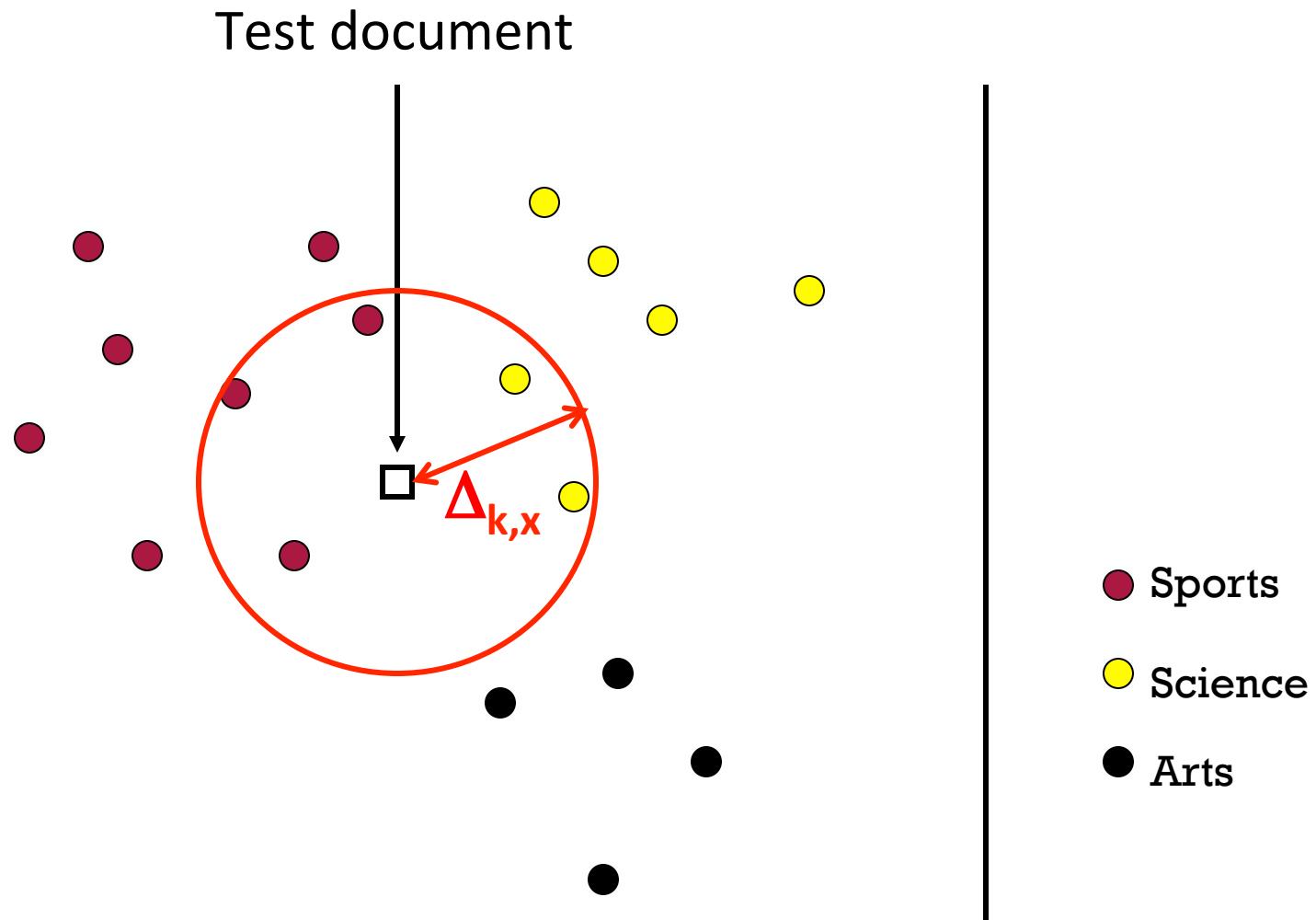


# k-NN classifier

Test document



# k-NN classifier (k=5)

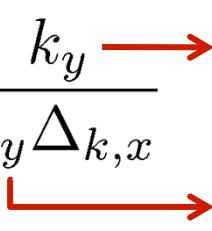


What should we predict? ... Average? Majority? Why?

# k-NN classifier

- Optimal Classifier: 
$$\begin{aligned} f^*(x) &= \arg \max_y P(y|x) \\ &= \arg \max_y p(x|y)P(y) \end{aligned}$$
- k-NN Classifier: 
$$\begin{aligned} \hat{f}_{kNN}(x) &= \arg \max_y \hat{p}_{kNN}(x|y)\hat{P}(y) \\ &= \arg \max_y k_y \end{aligned}$$

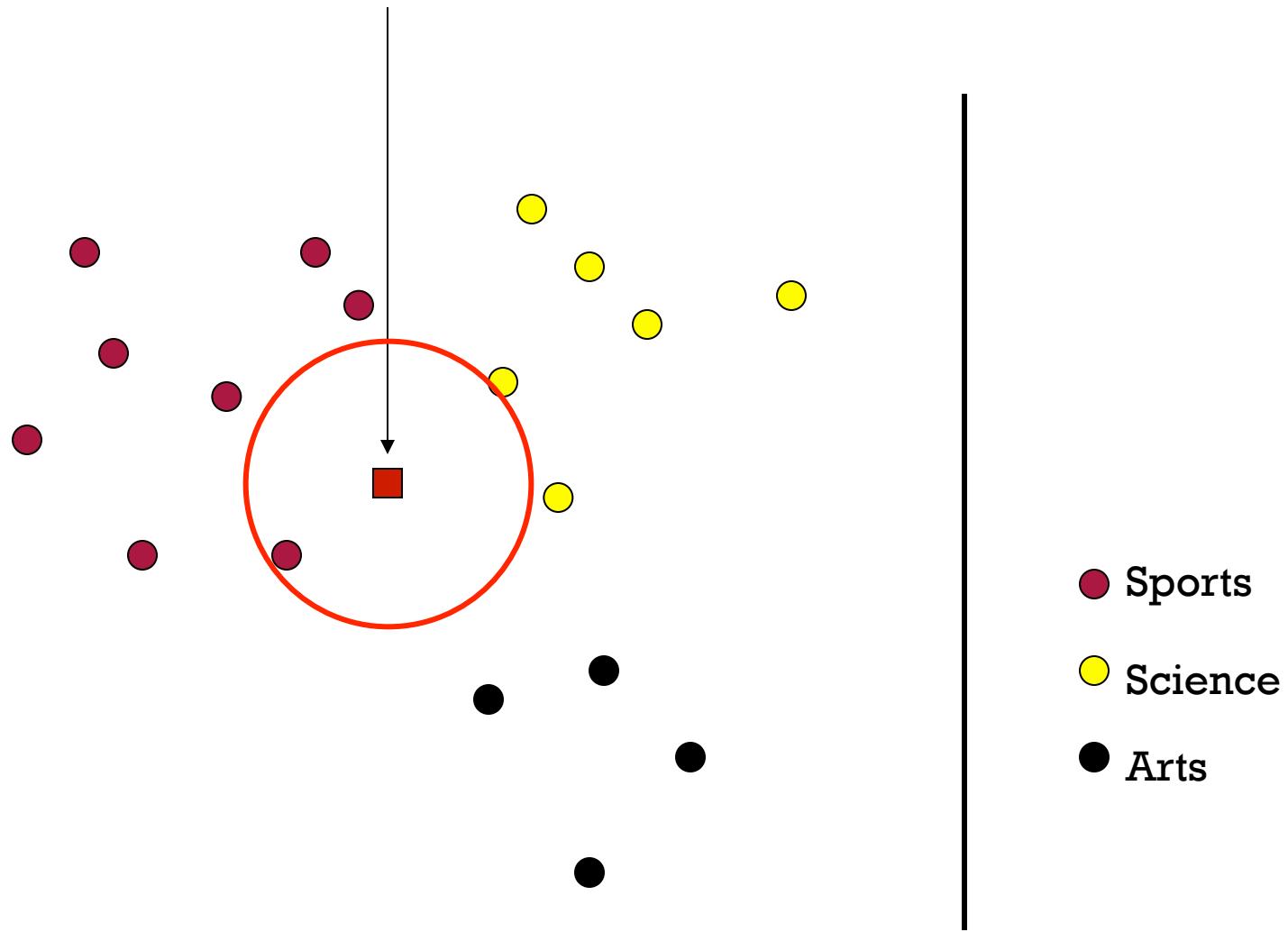
$$\hat{p}_{kNN}(x|y) = \frac{k_y}{n_y \Delta_{k,x}} \quad \begin{array}{l} \text{# training pts of class y} \\ \text{that lie within } \Delta_k \text{ ball} \end{array} \quad \sum_y k_y = k$$



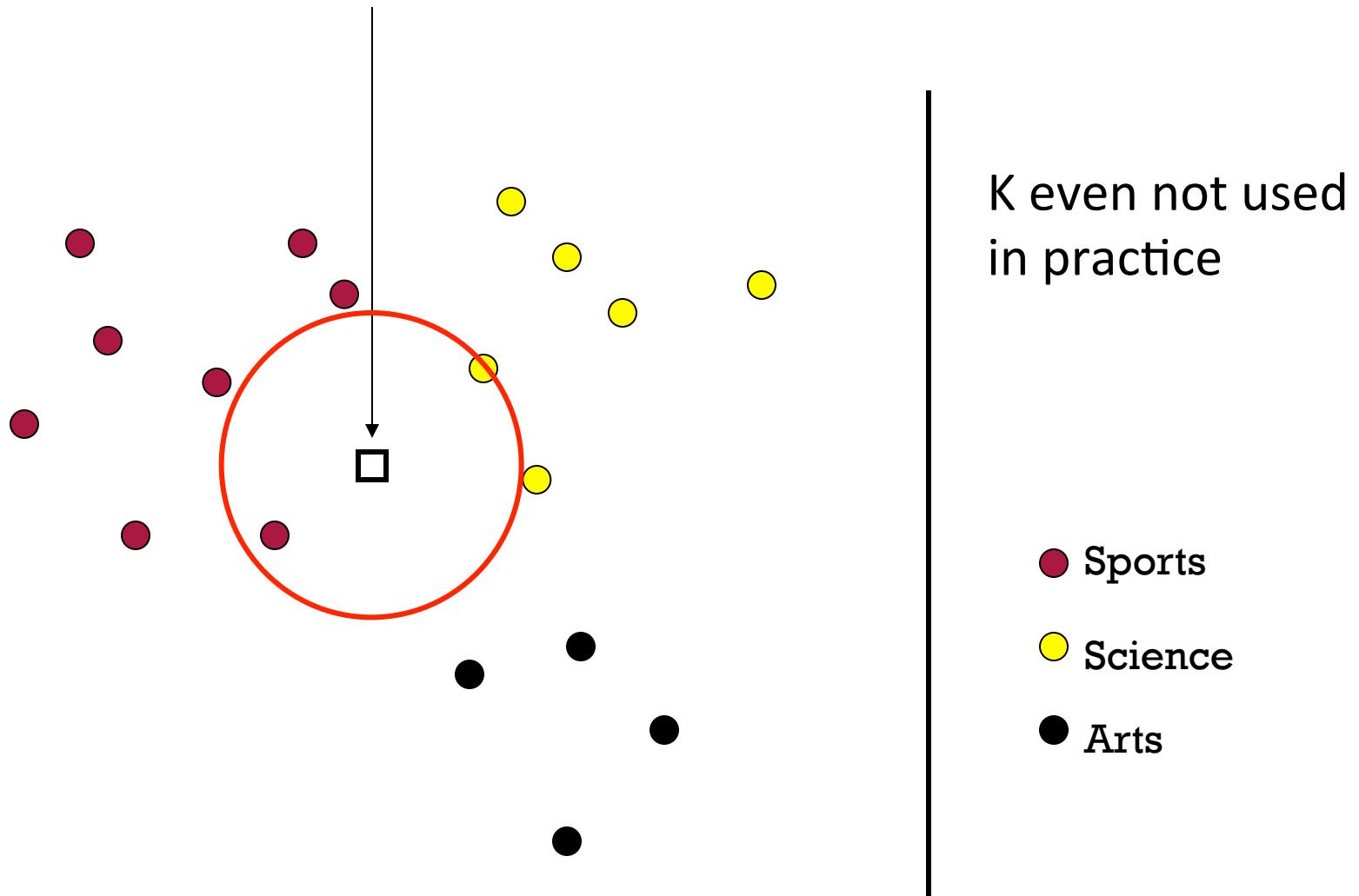
**# total training pts of class y**

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

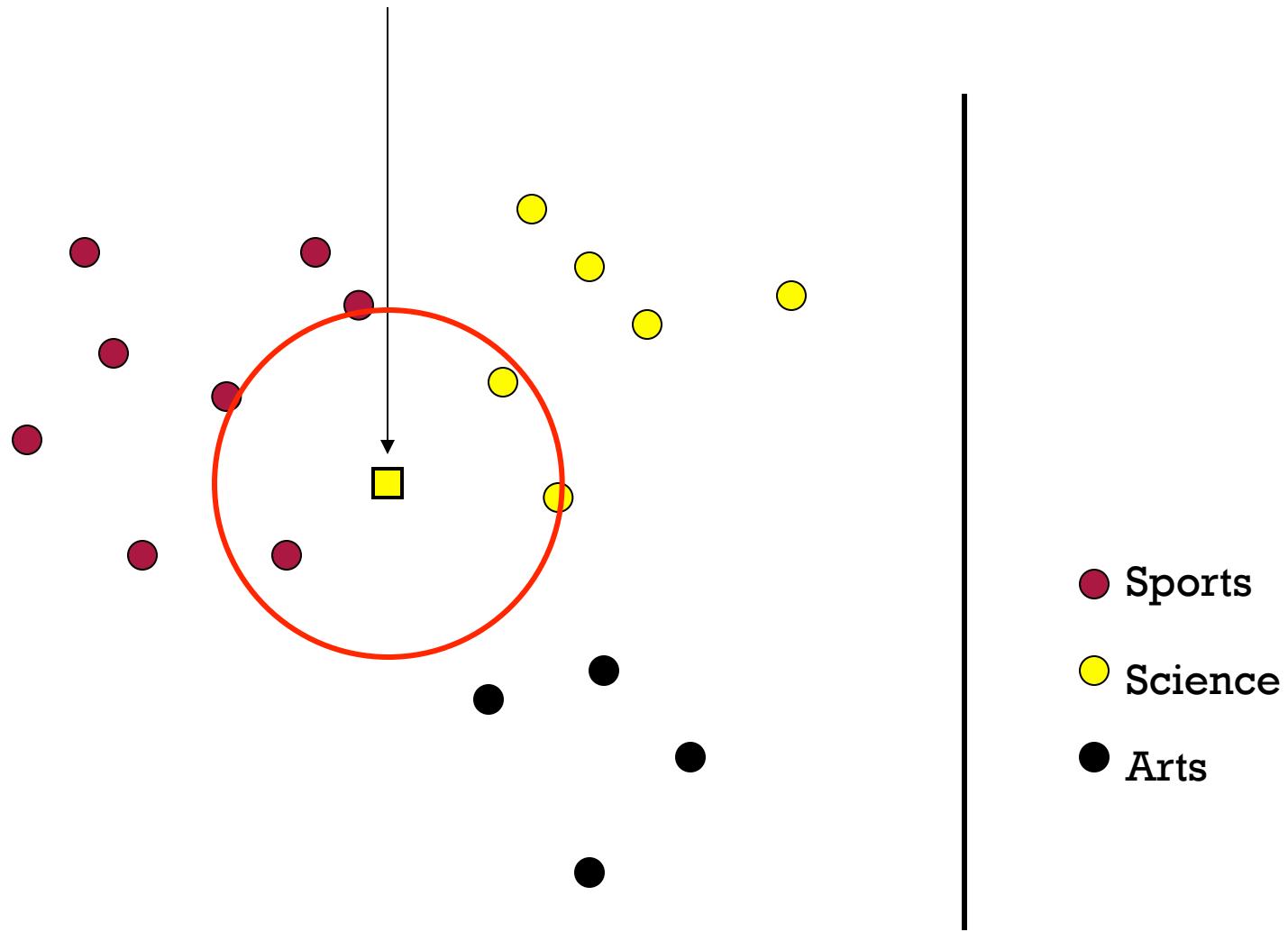
# 1-Nearest Neighbor (kNN) classifier



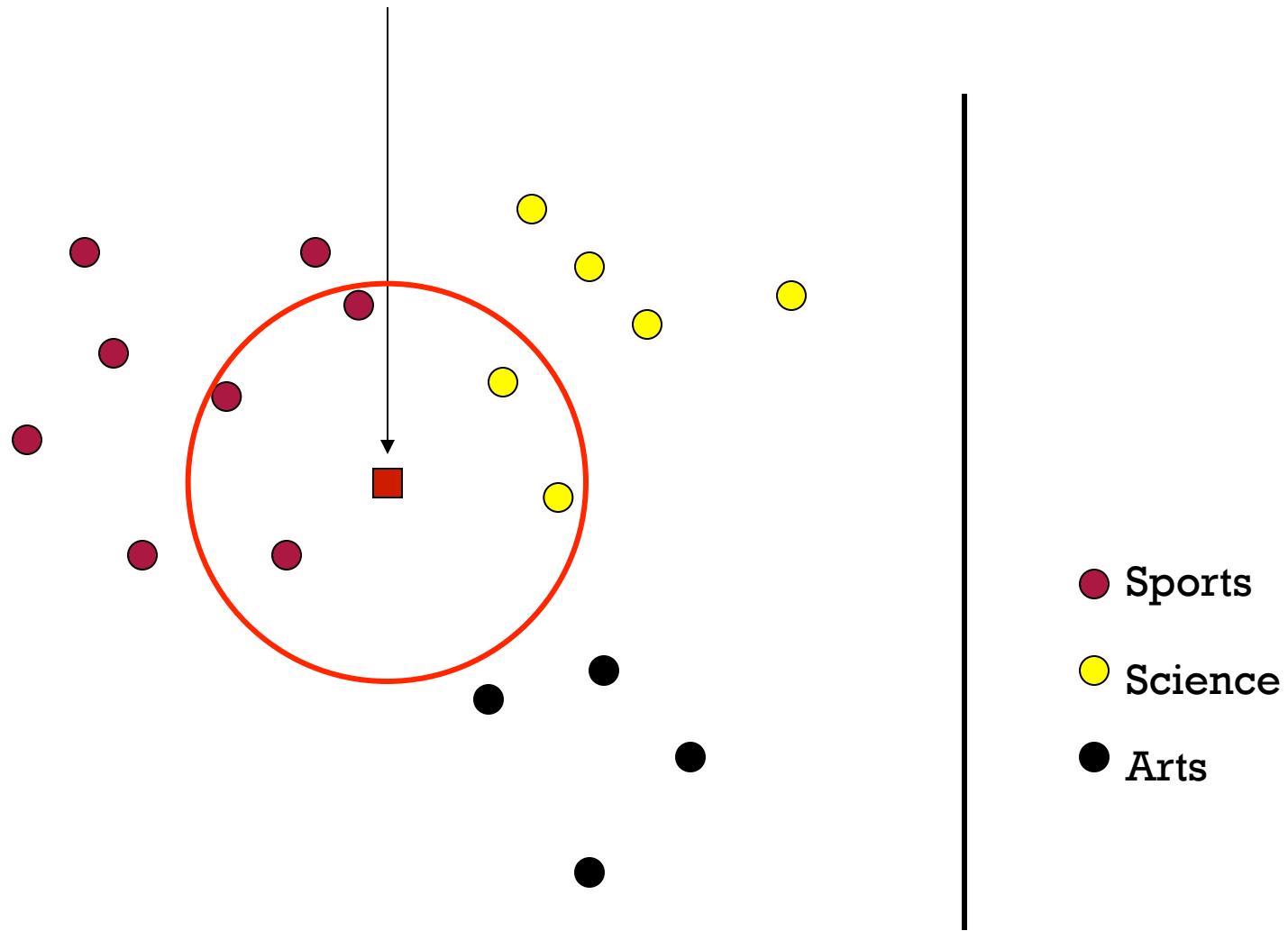
# 2-Nearest Neighbor (kNN) classifier



# 3-Nearest Neighbor (kNN) classifier



# 5-Nearest Neighbor (kNN) classifier



# What is the best K?

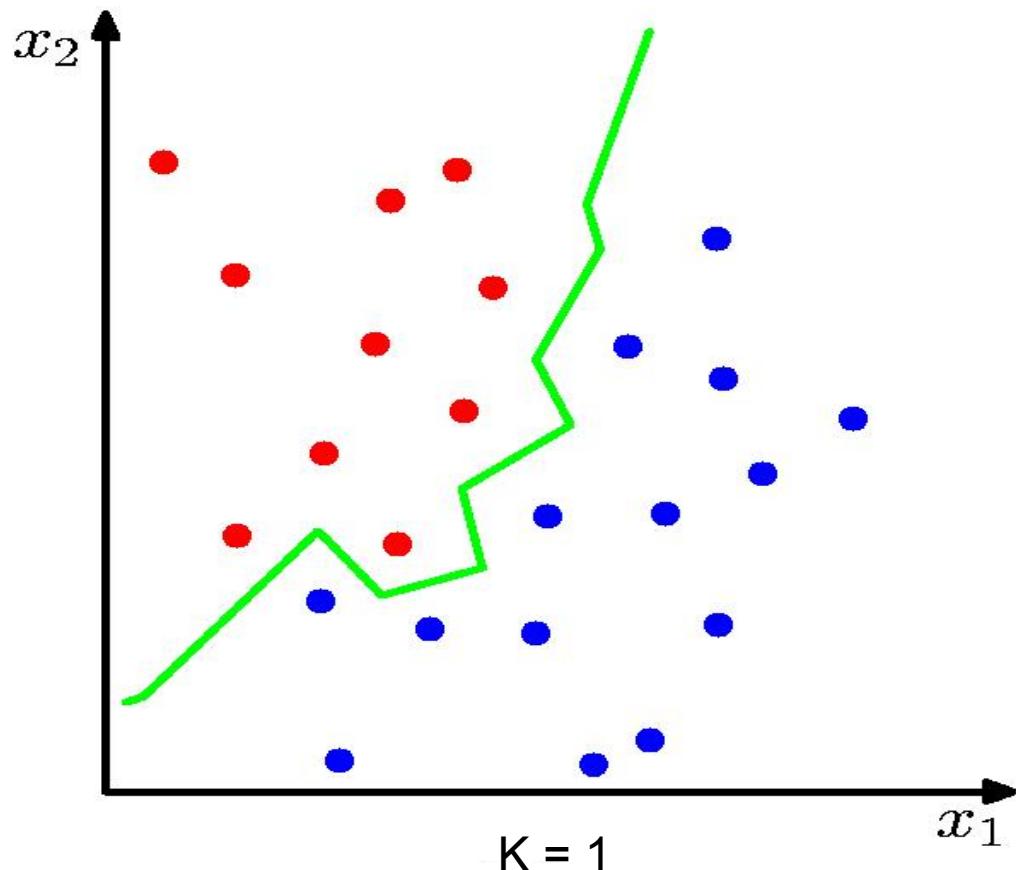
Bias-variance tradeoff

Larger K => predicted label is more stable

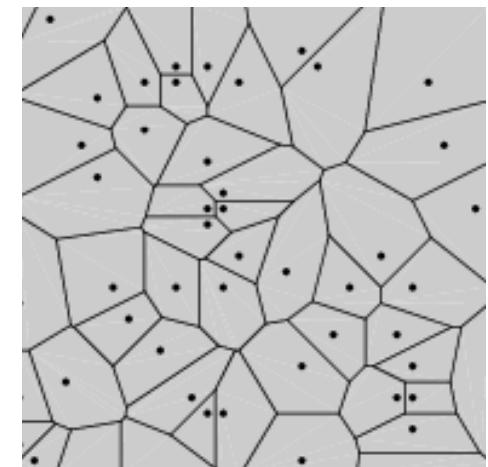
Smaller K => predicted label is more accurate

Similar to density estimation

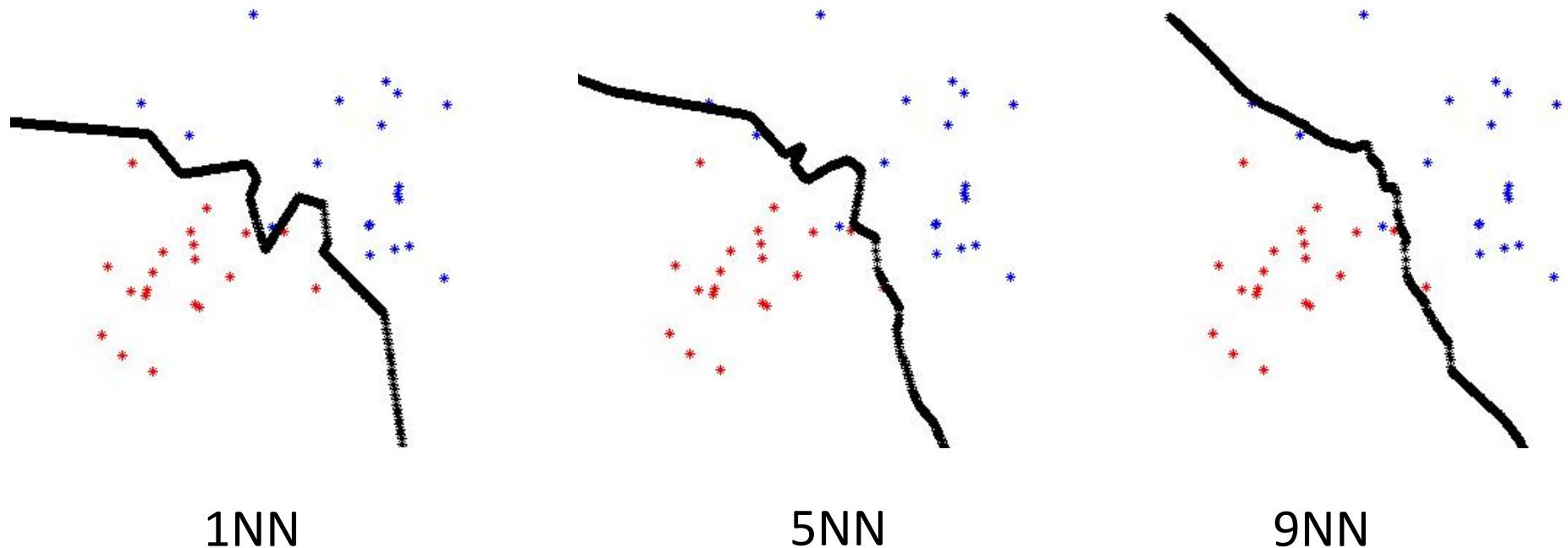
# 1-NN classifier – decision boundary



Voronoi  
Diagram



# k-NN classifier – decision boundary



- K acts as a smoother (Bias-variance tradeoff)

# Case Study: kNN for Web Classification

- Dataset
  - 20 News Groups (20 classes)
  - Download :(<http://people.csail.mit.edu/jrennie/20Newsgroups/>)
  - 61,118 words, 18,774 documents
  - Class labels descriptions

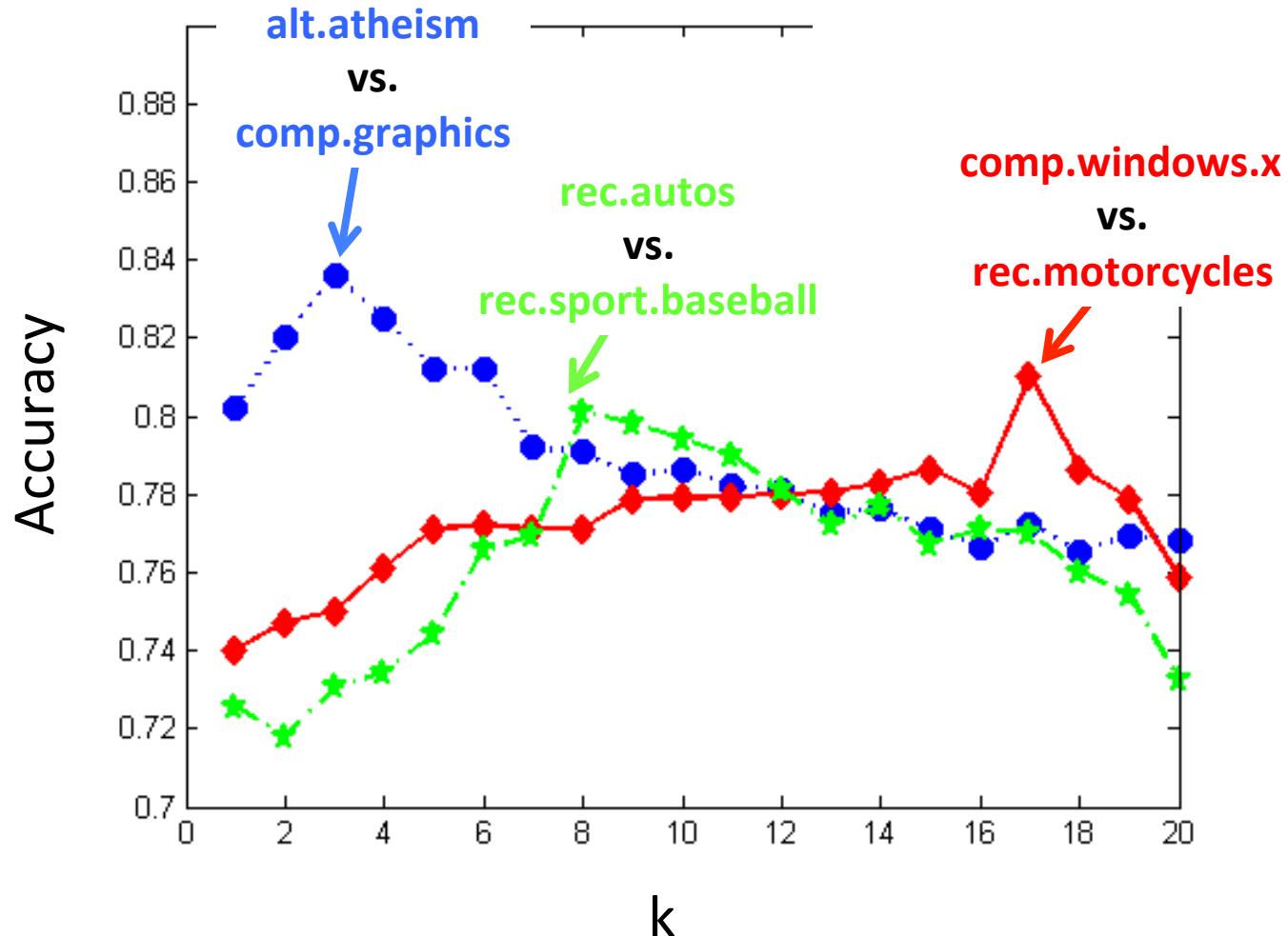
comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

# Experimental Setup

- Training/Test Sets:
  - 50%-50% randomly split.
  - 10 runs
  - report average results
- Evaluation Criteria:

$$Accuracy = \frac{\sum_{i \in \text{test set}} \mathbf{I}(\text{predict}_i = \text{true label}_i)}{\# \text{ of test samples}}$$

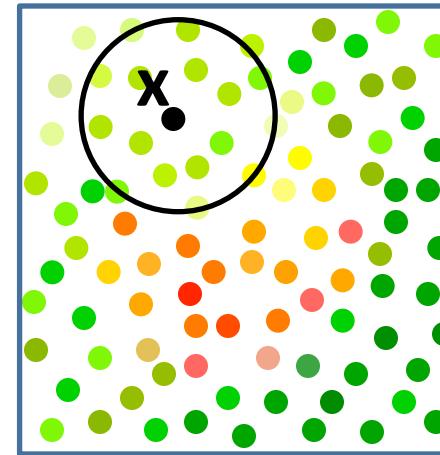
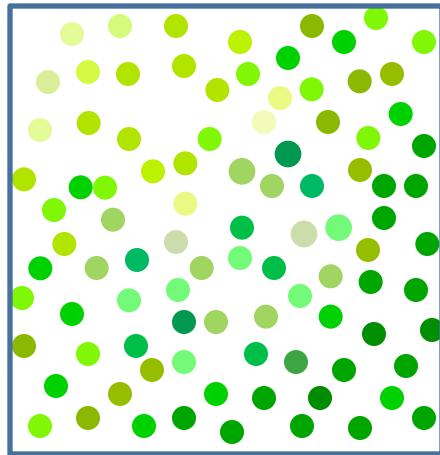
# Results: Binary Classes



From  
Classification  
to  
Regression

# Temperature sensing

- What is the temperature in the room?  
at location  $x$ ?



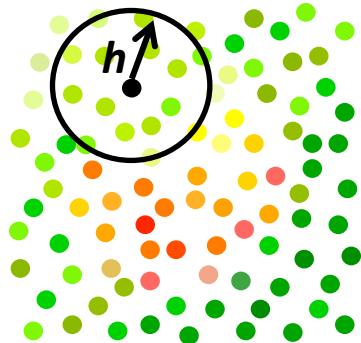
$$\hat{T} = \frac{1}{n} \sum_{i=1}^n Y_i$$

Average

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

“Local” Average

# Kernel Regression



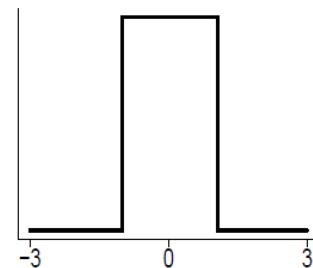
- Aka Local Regression
- Nadaraya-Watson Kernel Estimator

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

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

boxcar kernel :

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

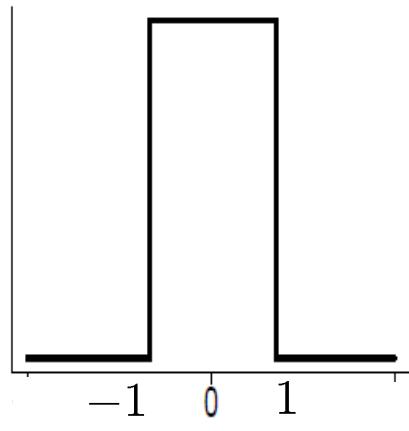


# Kernels

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

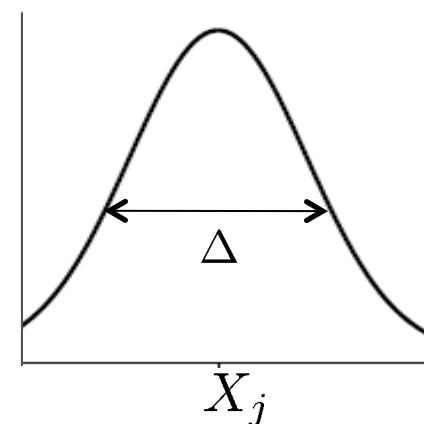
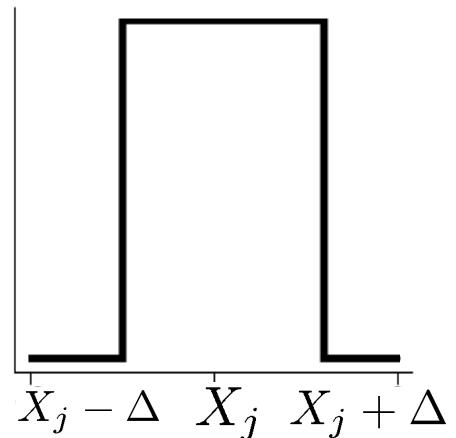
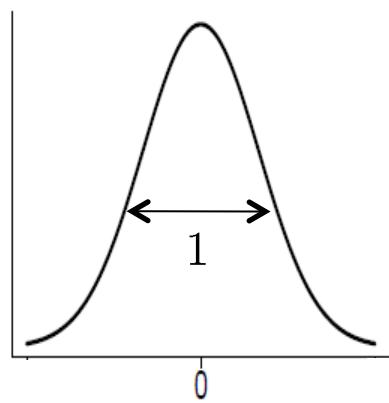
boxcar kernel :

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



Gaussian kernel :

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



# Choice of kernel bandwidth $h$

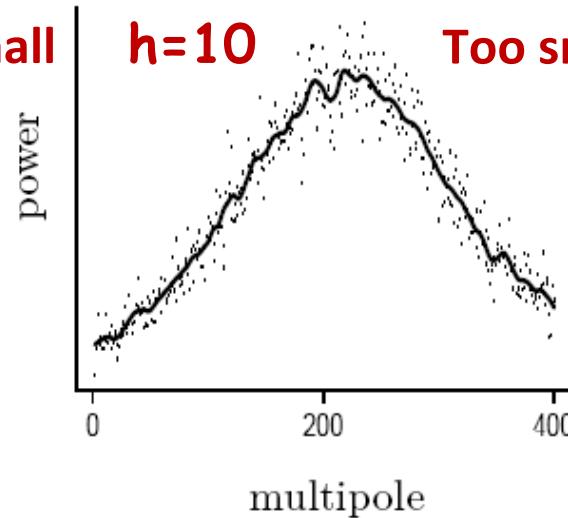
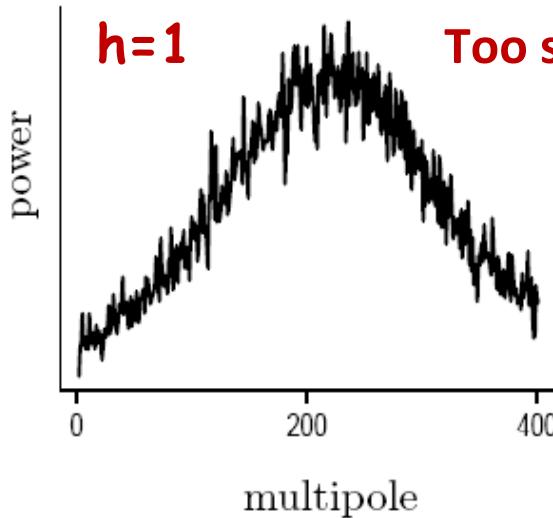
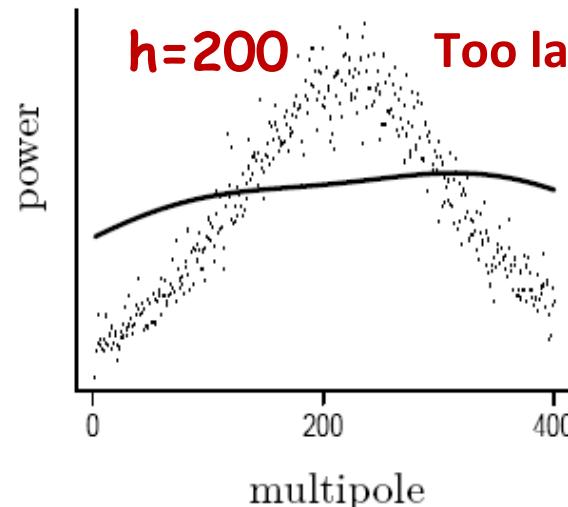
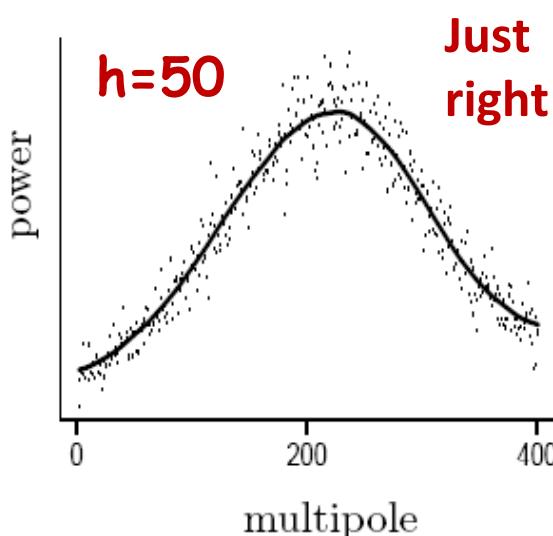


Image Source:  
Larry's book – All  
of Nonparametric  
Statistics



Choice of kernel is  
not that important

# Kernel Regression as Weighted Least Squares

$$\min_f \sum_{i=1}^n w_i (f(X_i) - Y_i)^2 \quad w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^n K\left(\frac{X-X_i}{h}\right)}$$

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)

# Kernel Regression as Weighted Least Squares

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

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

  
constant

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

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

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

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

# Local Linear/Polynomial Regression

$$\min_f \sum_{i=1}^n w_i (f(X_i) - Y_i)^2 \quad w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^n K\left(\frac{X-X_i}{h}\right)}$$

## Weighted Least Squares

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

$$f(X_i) = \beta_0 + \beta_1(X_i - X) + \frac{\beta_2}{2!}(X_i - X)^2 + \dots + \frac{\beta_p}{p!}(X_i - X)^p$$

i.e. set

(local polynomial of degree p around X)

# Summary

- Instance based/non-parametric approaches

**Four things make a memory based learner:**

1. *A distance metric,  $dist(x, X_i)$*   
**Euclidean (and many more)**
2. *How many nearby neighbors/radius to look at?*  
 **$k, \Delta/h$**
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) distributional assumptions
  - Nonparametric models (not histograms) requires storing and computing with the entire data set.  
Parametric models, once fitted, are much more efficient in terms of storage and computation.

# What you should know...

- Histograms, Kernel density estimation
  - Effect of bin width/ kernel bandwidth
  - Bias-variance tradeoff
- K-NN classifier
  - Nonlinear decision boundaries
- Kernel (local) regression
  - Interpretation as weighted least squares
  - Local constant/linear/polynomial regression

# Bias-variance tradeoff

- Simulations