k-NN (k-Nearest Neighbors), Kernel Regression

Pradeep Ravikumar

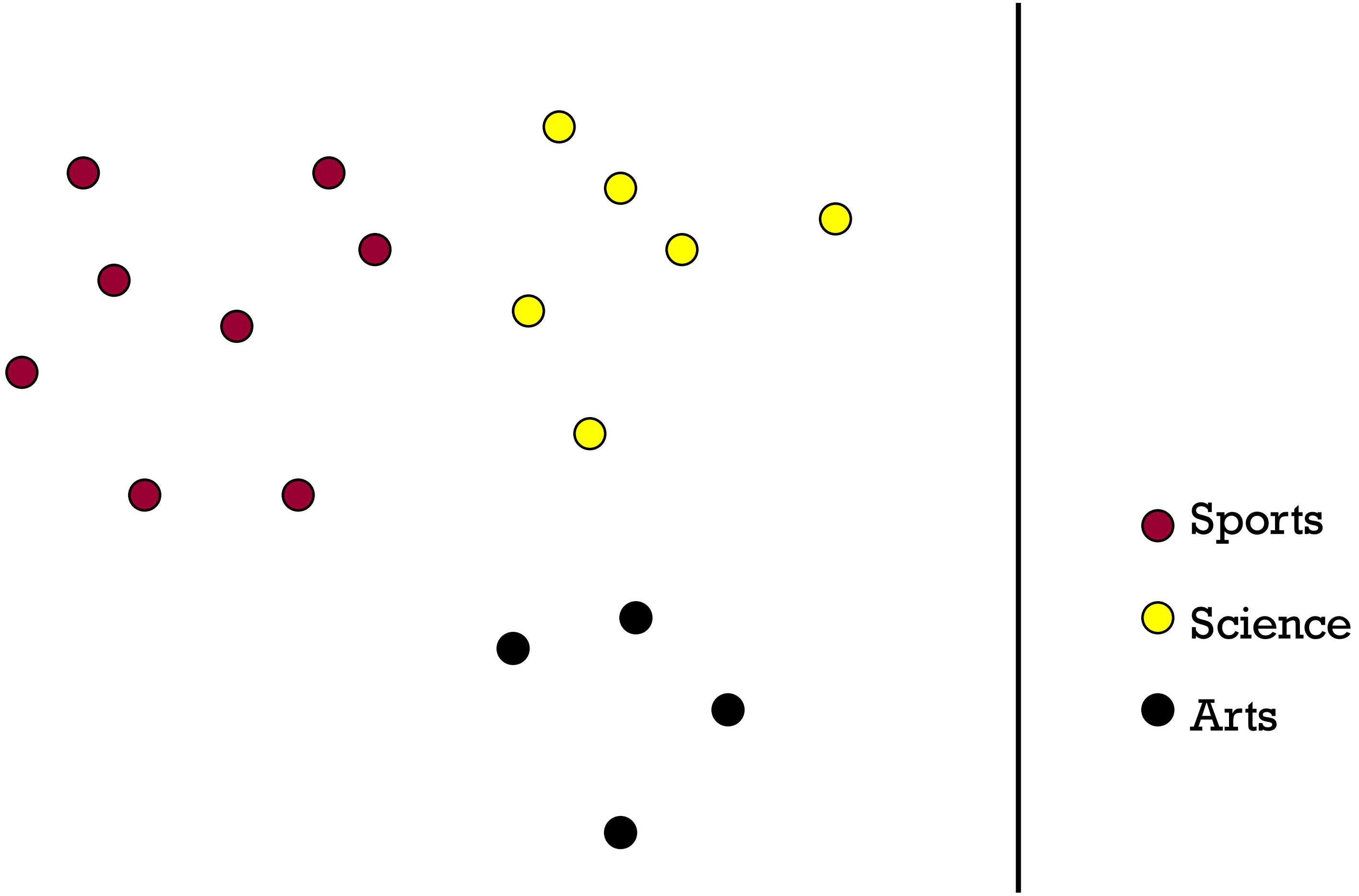
Co-instructor: Manuela Veloso

Machine Learning 10-701

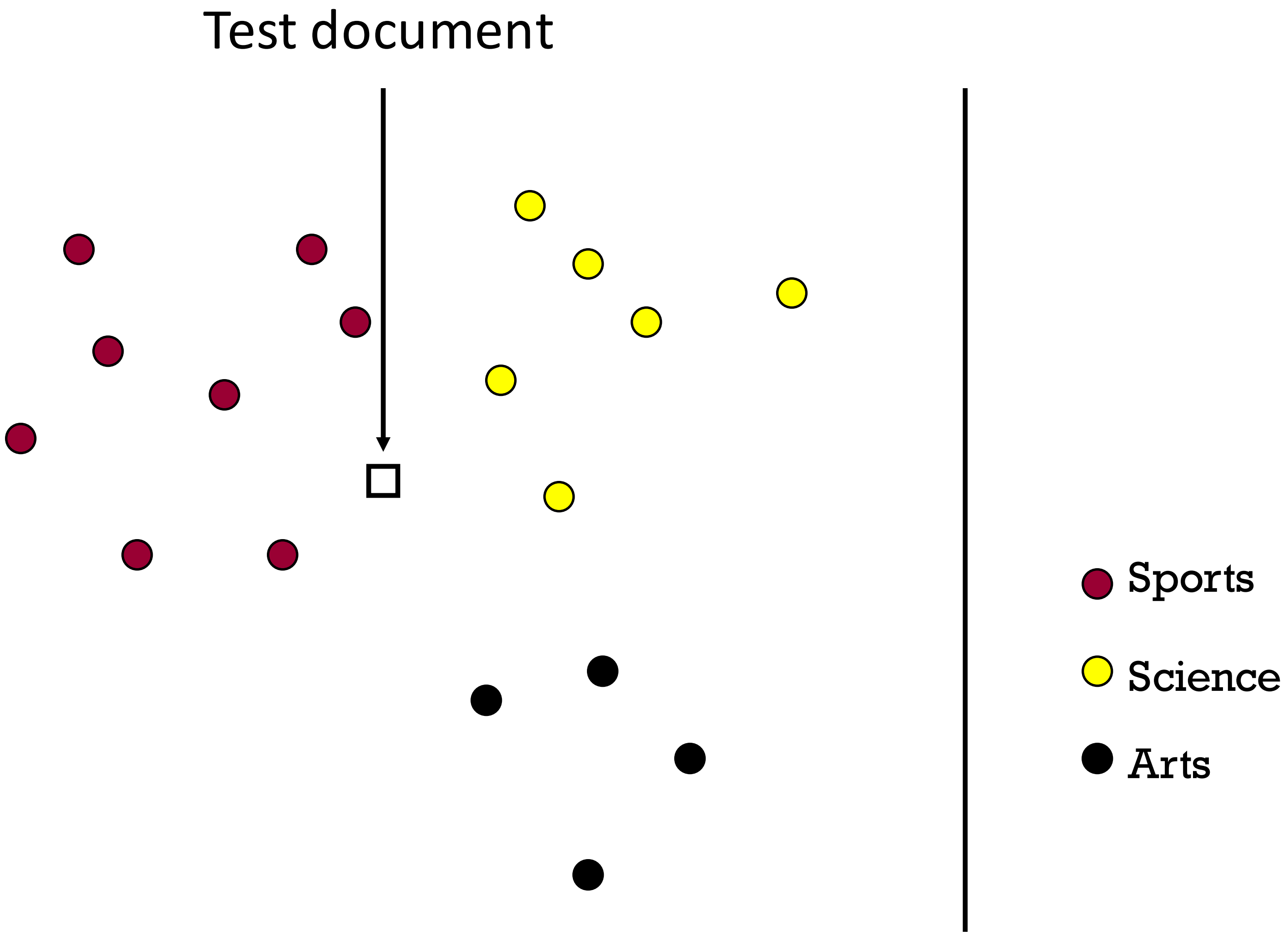
MACHINE LEARNING DEPARTMENT

Carnegie Mellon.
School of Computer Science

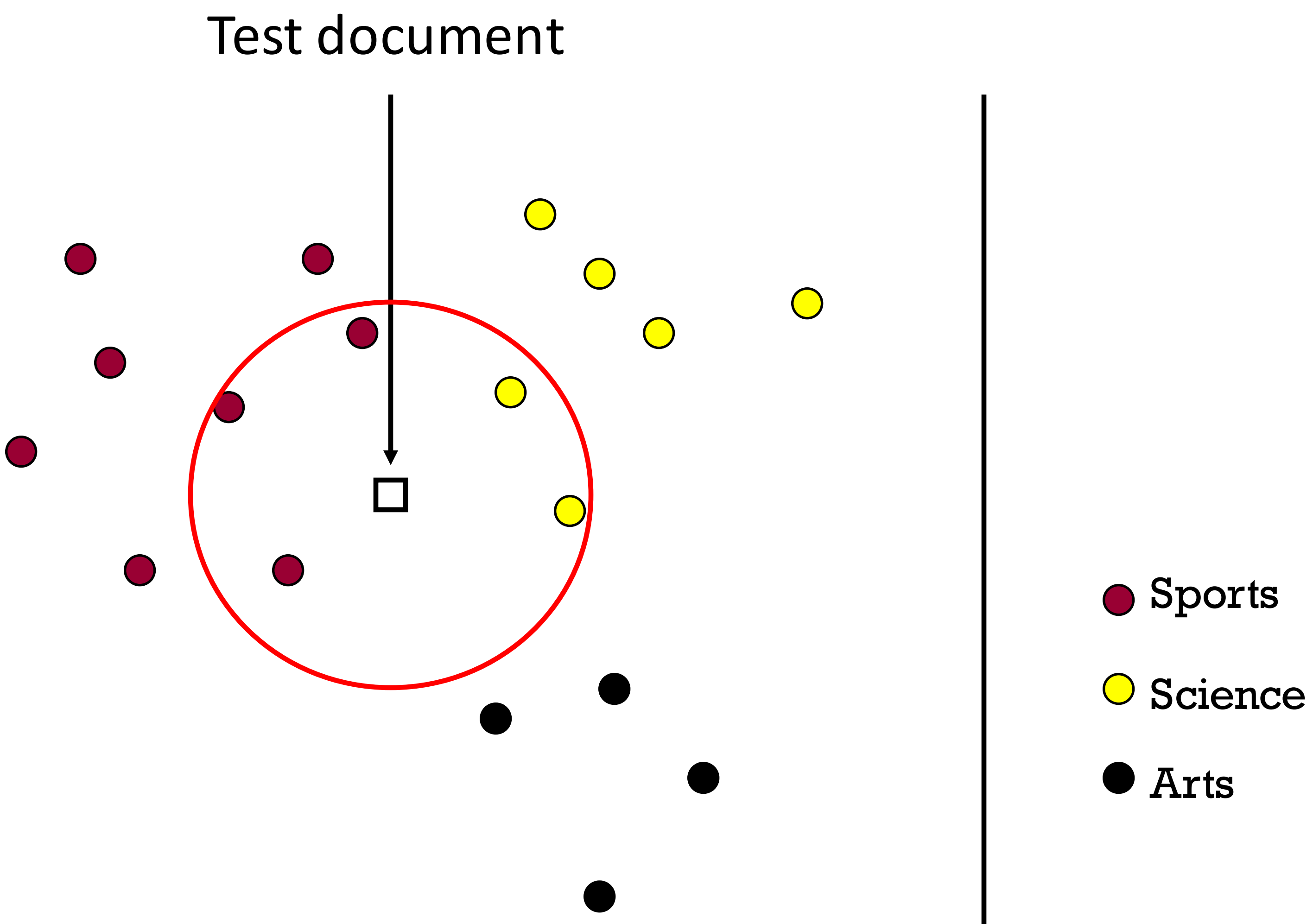
k-NN classifier



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

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

 # training pts of class y amongst k NNs of x

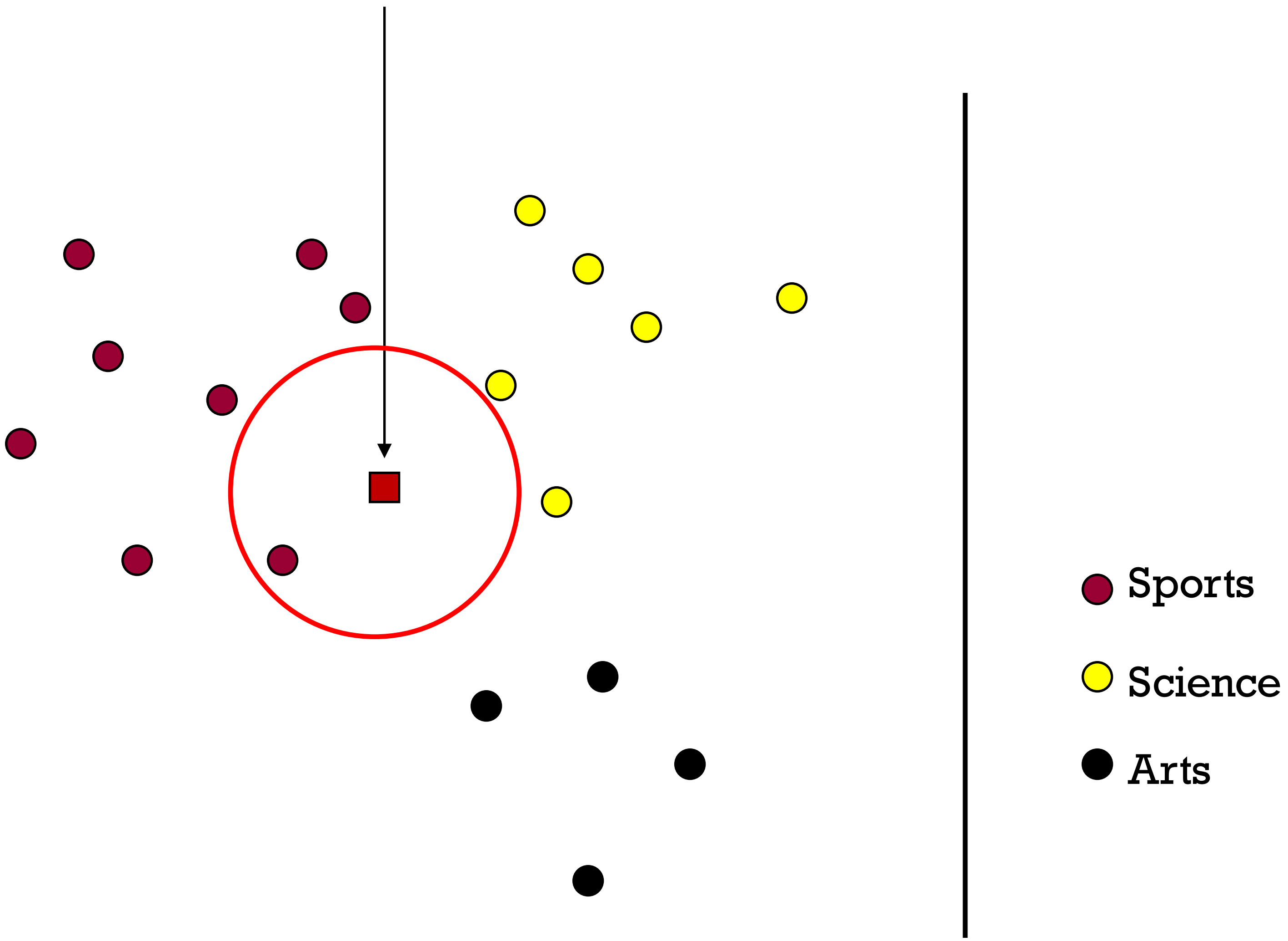
 # total training pts of class y

$$\sum_y k_y = k$$

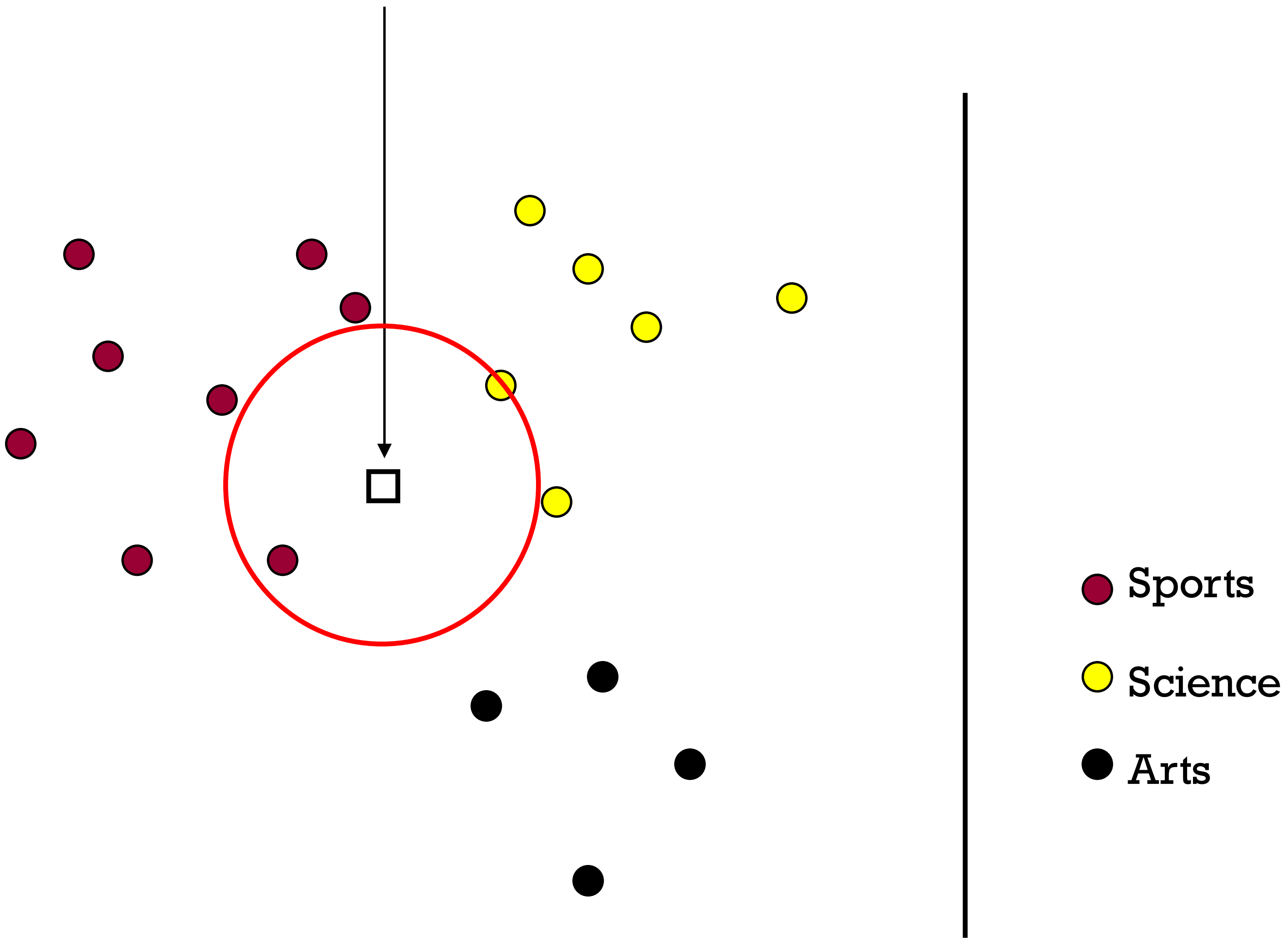
no. of training pts like x with label y / no. of training pts with label y

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

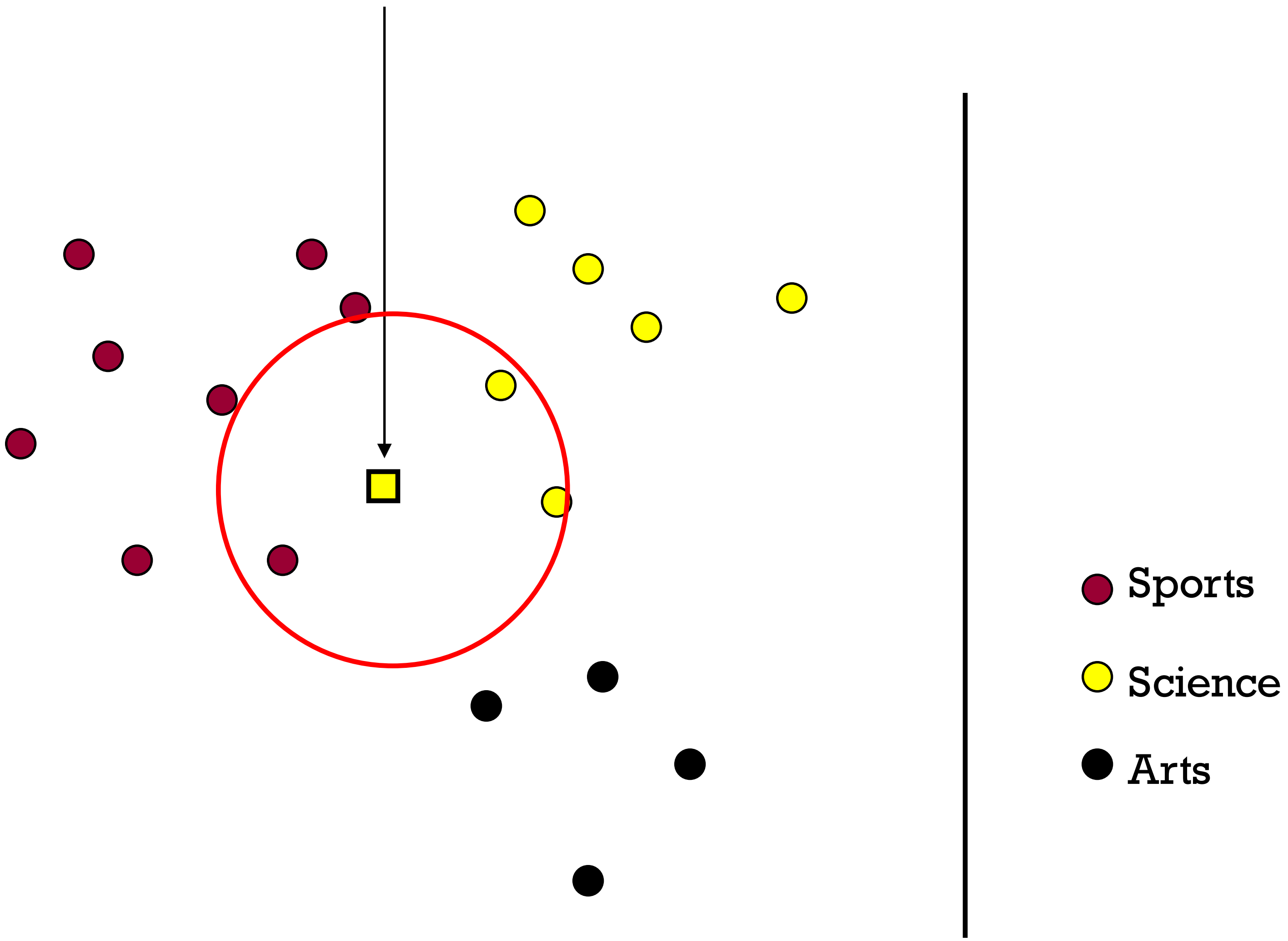
1-Nearest Neighbor (kNN) classifier



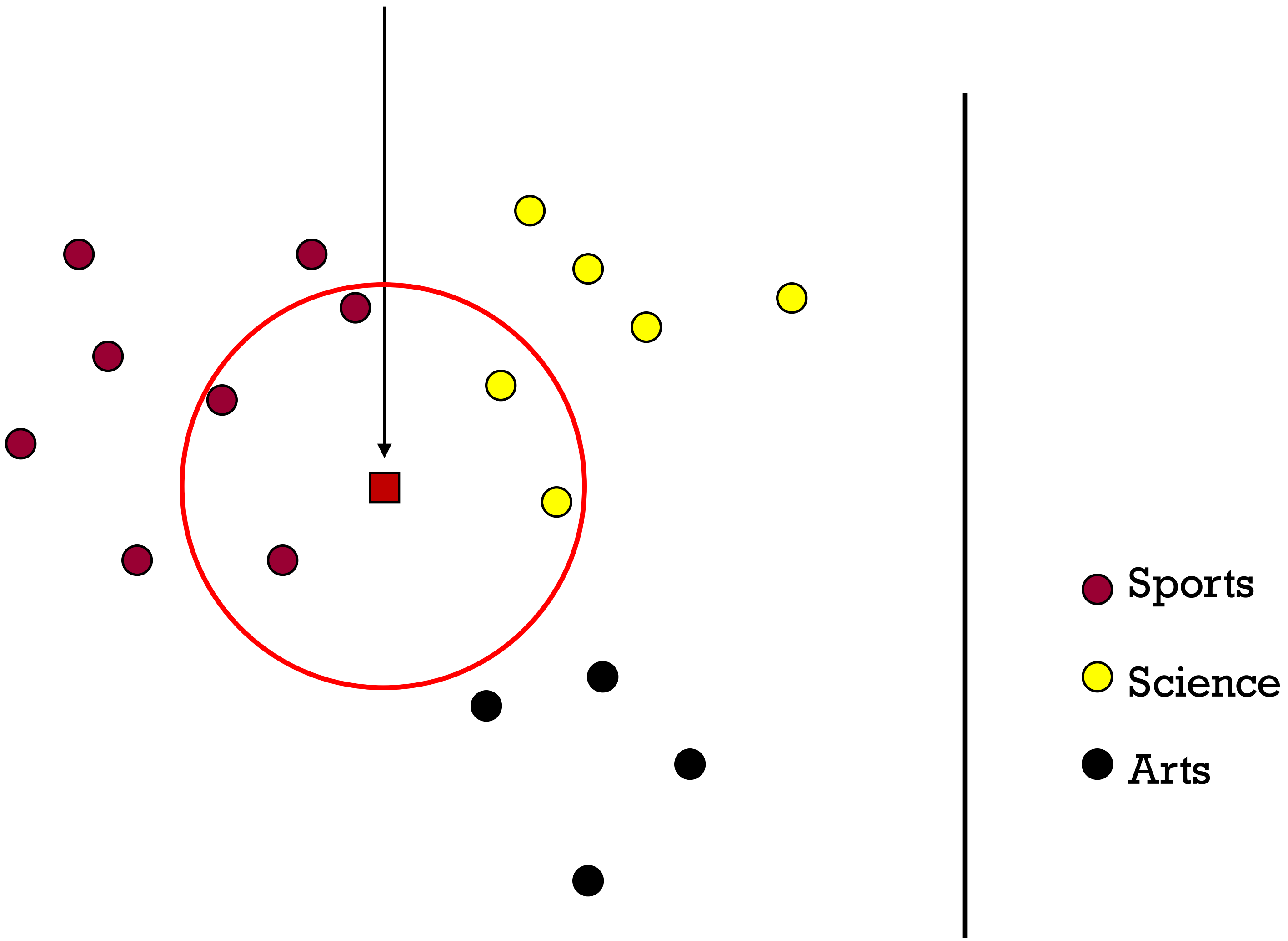
2-Nearest Neighbor (kNN) classifier



3-Nearest Neighbor (kNN) classifier

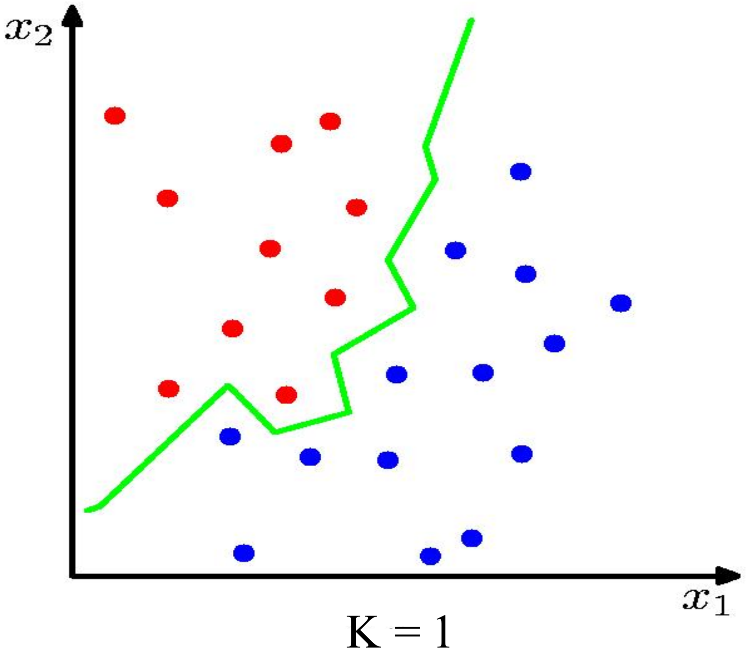


5-Nearest Neighbor (kNN) classifier

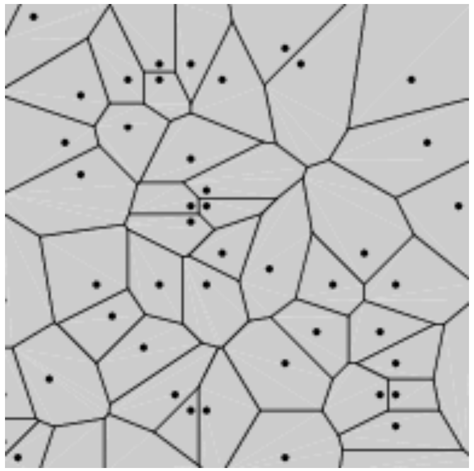


What is the best k?

1-NN classifier decision boundary



Voronoi Diagram



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

What is the best k?

Approximation vs. Stability Tradeoff

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

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, Perceptron, Neural Networks

- Estimate parameters (μ,σ^2,θ,w,β) using MLE/MAP and plug in

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

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

- Some nonparametric methods
  - Decision Trees
  - k-NN (k-Nearest Neighbor) Classifier

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 requires storing and computing with the entire data set.

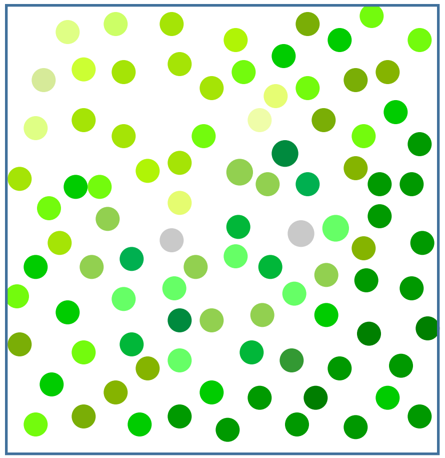
  Parametric models, once fitted, are much more efficient in terms of storage and computation.

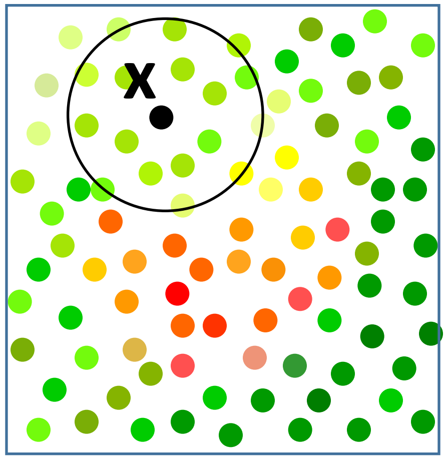
Local, Kernel Regression

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

Nadaraya-Watson Kernel Regression

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

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

with box-car kernel

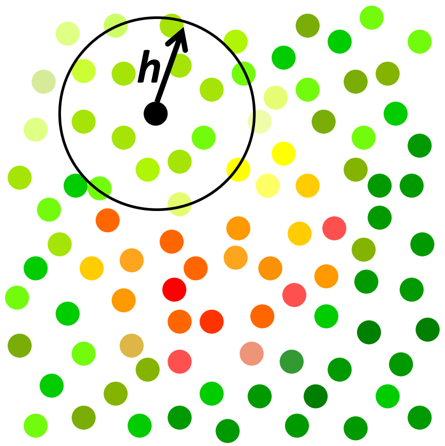
$$= \frac{1}{n_X^h} \underbrace{\sum_{i=1}^{n} Y_i \, \mathbf{1}_{|X-X_i|\leq h}}_{\text{Sum of Ys in h ball around X}}$$

#pts in h ball around X

Sum of Ys in h ball around X

boxcar kernel :

$$K\left(\frac{X-X_i}{h}\right) = \mathbf{1}_{|X-X_i|\leq h}$$



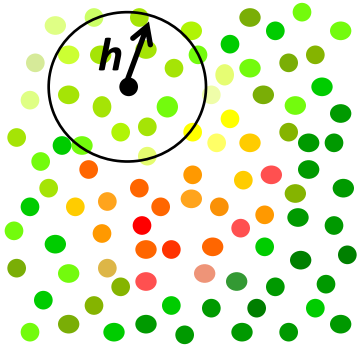
Recall: NN classifier with majority vote

Here we use Average instead

Local Kernel Regression

- Nonparametric estimator akin to kNN
- Nadaraya-Watson Kernel Estimator

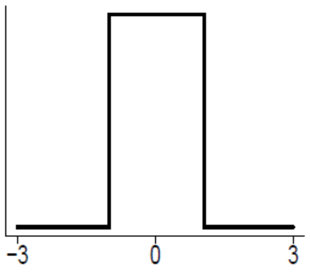
$$\widehat{f}_n(X) = \sum_{i=1}^{n} w_i Y_i \quad \text{Where} \quad w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X-X_i}{h}\right)}$$



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

boxcar kernel :

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



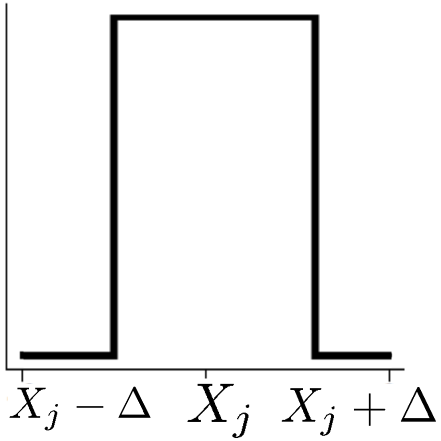
Kernels

$$\begin{aligned} K(x) &\geq 0, \\ \int K(x)dx &= 1 \end{aligned}$$

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

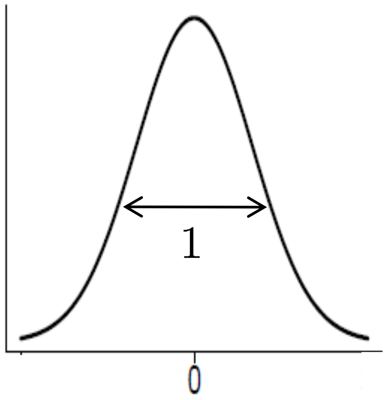
boxcar kernel :

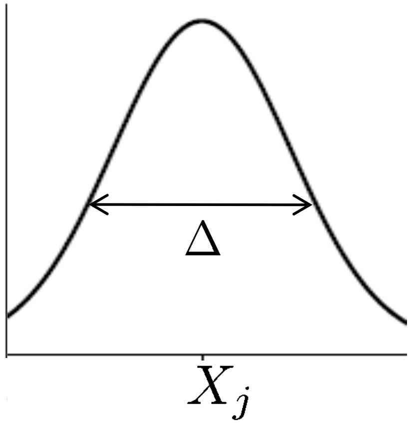
$$K(x) = \frac{1}{2}I(x),$$



Gaussian kernel :

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





Choice of kernel bandwidth h

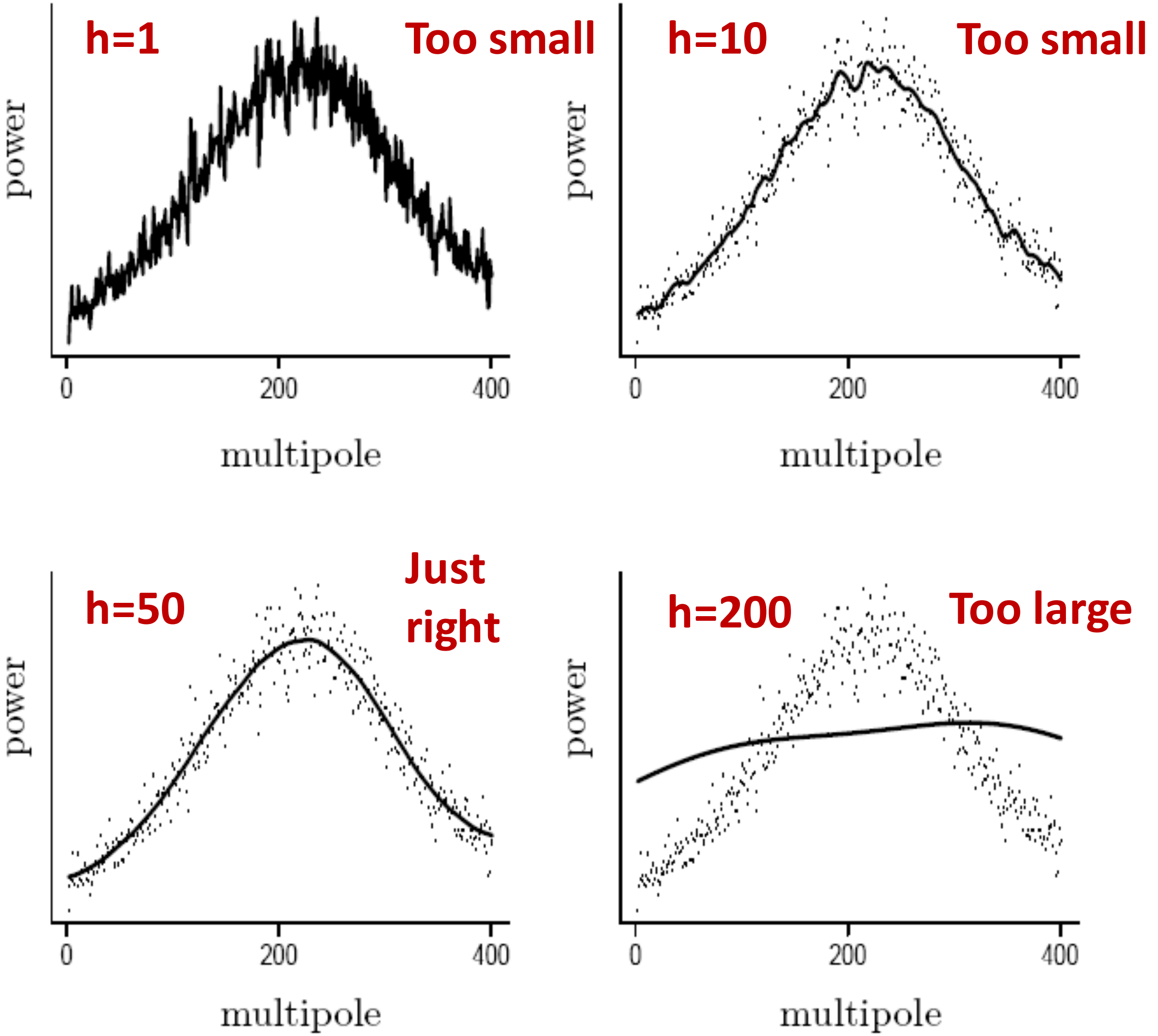


Image Source: Larry's book – All of Nonparametric Statistics

Choice of kernel is not that important

Kernel Regression as Weighted Least Squares

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

Weighted Least Squares

Weigh each training point based on distance to test point

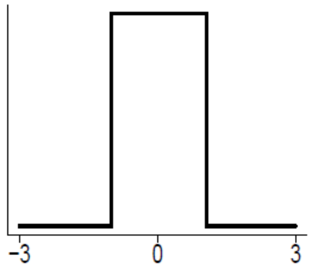
$$w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X-X_i}{h}\right)}$$

K – Kernel

h – Bandwidth of kernel

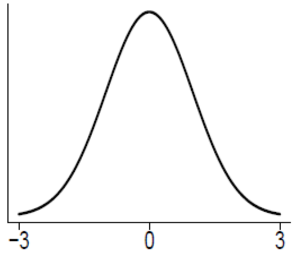
boxcar kernel :

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



Gaussian kernel :

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



Kernel Regression as Weighted Least Squares

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

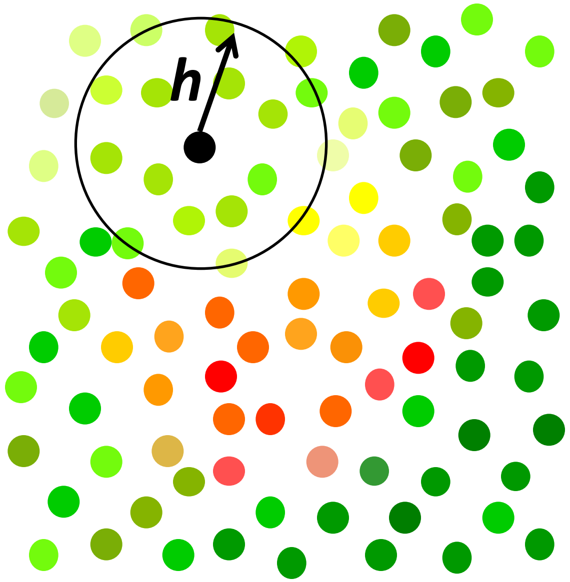
$$\min_{\beta} \sum_{i=1}^{n} w_i(\underbrace{\beta}_{\text{constant}} - Y_i)^2 \qquad w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X-X_i}{h}\right)}$$

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

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

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

Choice of Bandwidth



Should depend on n, # training data
(determines variance)

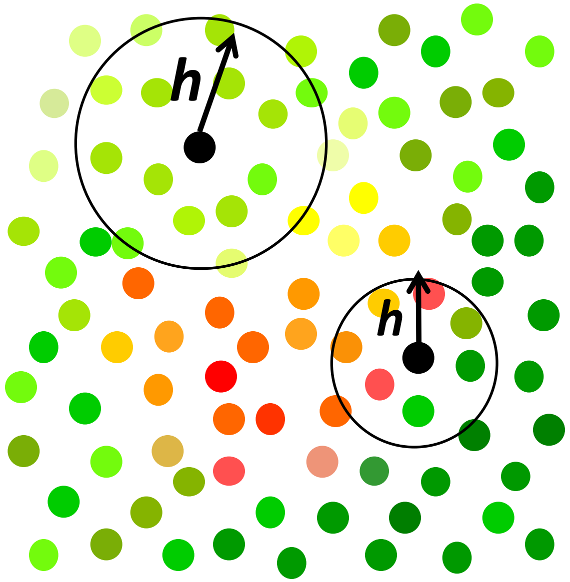
Should depend on smoothness of function
(determines bias)

Large Bandwidth – average more data points, reduce noise (Lower variance)

Small Bandwidth – less smoothing, more accurate fit (Lower bias)

Bias – Variance tradeoff

Spatially adaptive regression



If function smoothness varies spatially, we want to allow bandwidth h to depend on X

Local polynomials, splines, wavelets, regression trees …

Local Linear/Polynomial Regression

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

Weighted Least Squares

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

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

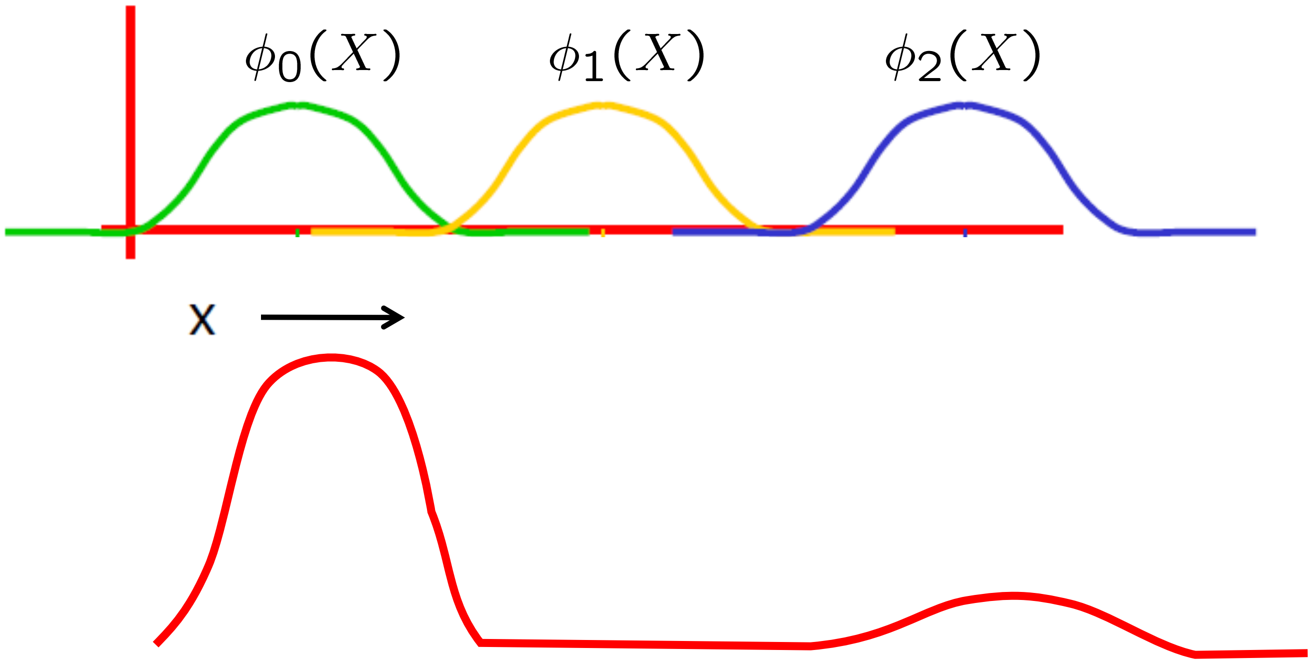
(local polynomial of degree p around X)

Local Regression

$$f(X) = \sum_{j=0}^{m} \beta_j \underbrace{\phi_j(X)}$$

Basis coefficients

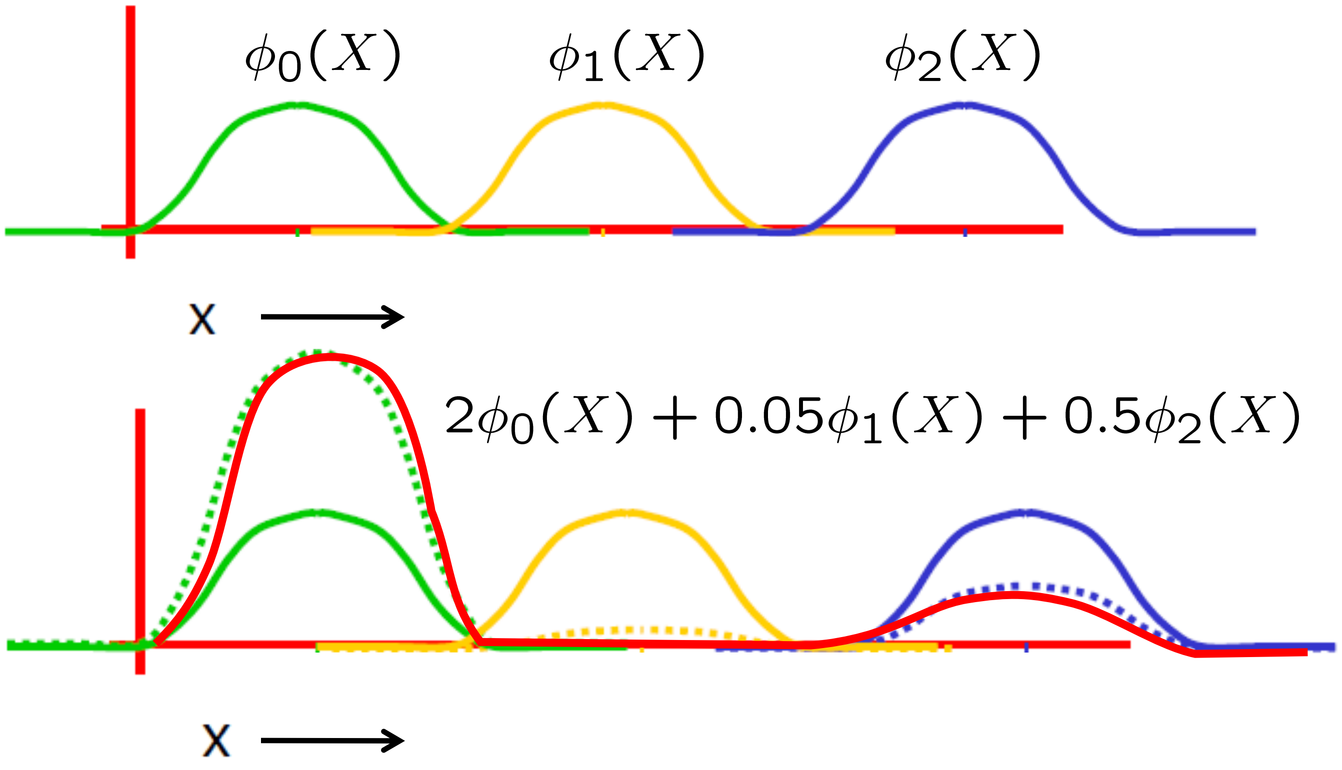
Nonlinear features/basis functions



Globally supported basis functions (polynomial, fourier) will not yield a good representation

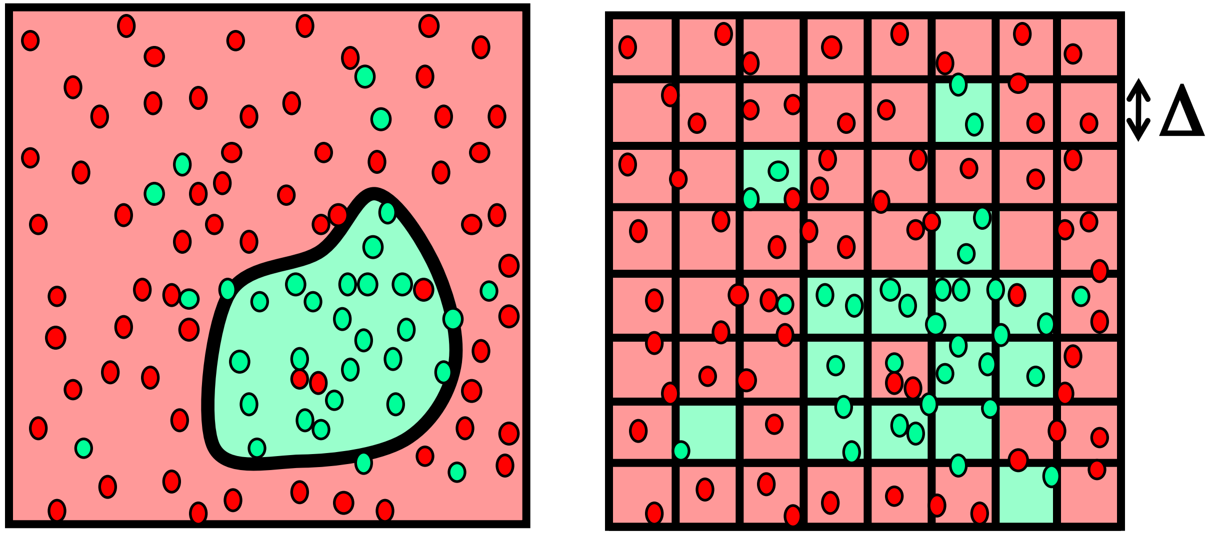
Local Regression

$$f(X) = \sum_{j=0}^{m} \underbrace{\beta_j}_{\text{Basis coefficients}} \underbrace{\phi_j(X)}_{\text{Nonlinear features/basis functions}}$$



Globally supported basis functions (polynomial, fourier) will not yield a good representation

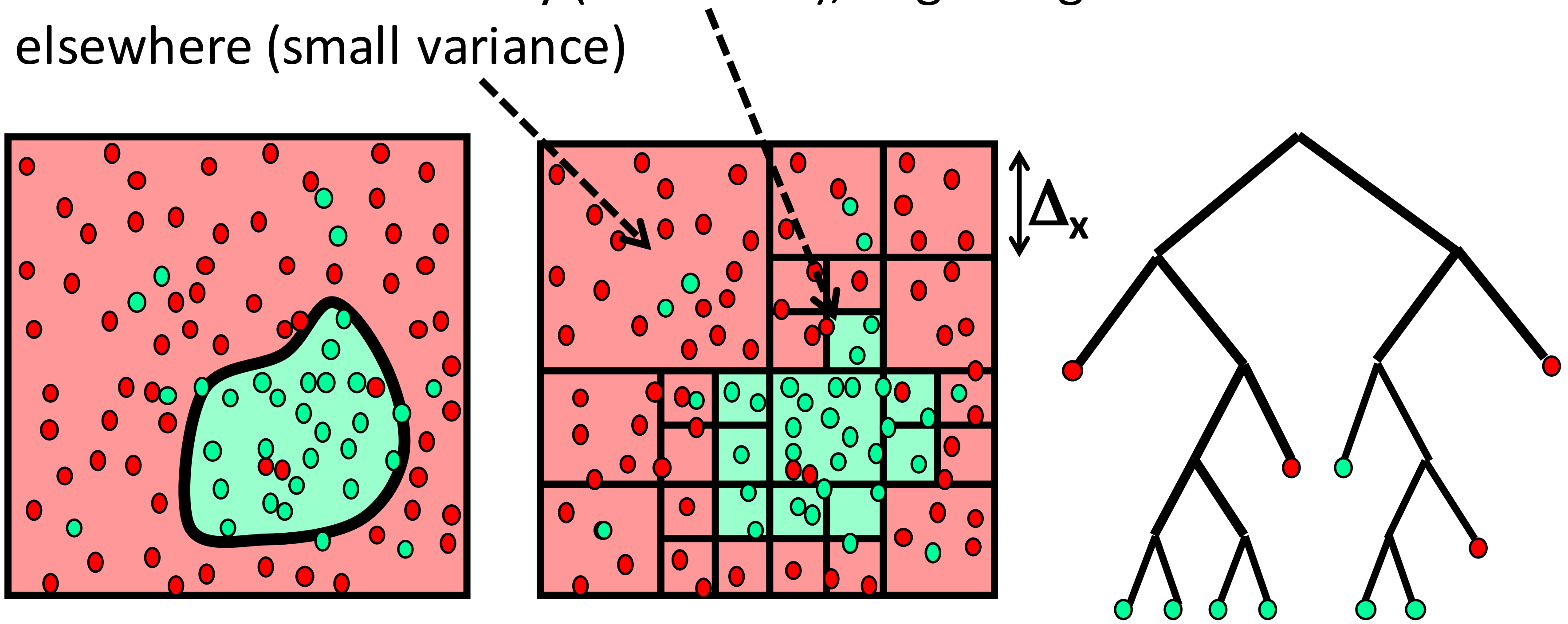
Local prediction



Histogram Classifier

Local Adaptive prediction

Let neighborhood size adapt to data – small neighborhoods near decision boundary (small bias), large neighborhoods elsewhere (small variance)



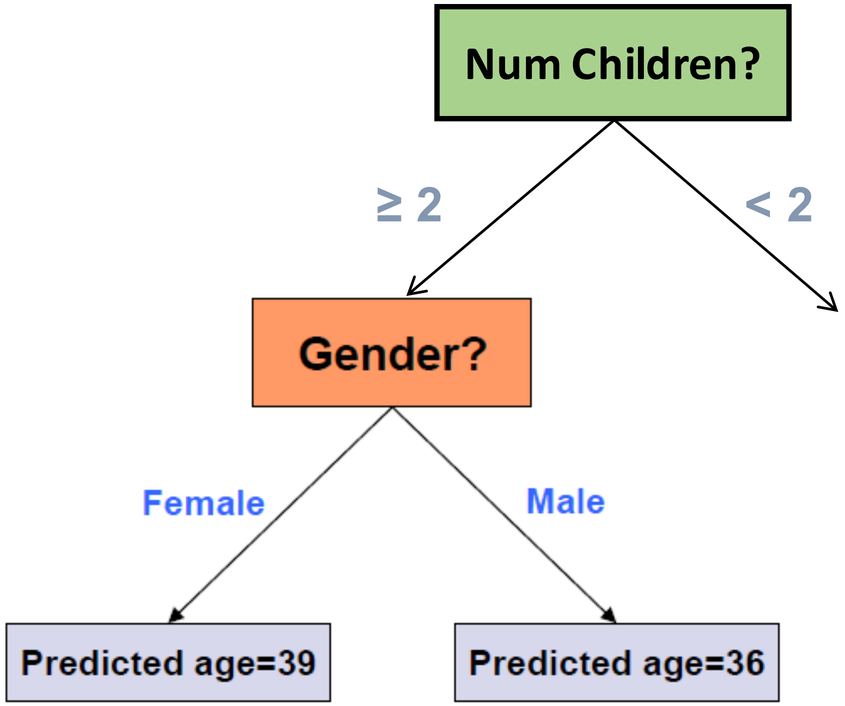
Decision Tree Classifier

Majority vote at each leaf

Regression trees

$X^{(1)}$	. . . .		$X^{(p)}$	Y
Gender	Rich?	Num. Children	# travel per yr.	Age
F	No	2	5	38
M	No	0	2	25
M	Yes	1	0	72
:	:	:	:	:

Binary Decision Tree



Average (fit a constant) on the leaves