UVA CS 6316: Machine Learning

Lecture 9: K-nearest-neighbor

Dr. Yanjun Qi

University of Virginia
Department of Computer Science

Course Content Plan → Six major sections of this course

- ~~Regression (supervised)~~ — Y is a continuous
- Classification (supervised) — Y is a discrete
- Unsupervised models — NO Y
- Learning theory — About f()
- Graphical models — About interactions among X1,… Xp
- Reinforcement Learning — Learn program to Interact with its environment

Last Recap: Supervised Classifiers

Task

Representation

Score Function

Search/Optimization

Models, Parameters

Classification

Output Y is categorical

Separation Boundary / Tree
Local Smooth / Generative

?

?

ACC / F1 /
Many metric

Three major sections for classification

- We can divide the large variety of classification approaches into roughly three major types

1. Discriminative
   directly estimate a decision rule/boundary
   e.g., support vector machine, decision tree, logistic regression,
   e.g. neural networks (NN), deep NN

2. Generative:
   build a generative statistical model
   e.g., Bayesian networks, Naïve Bayes classifier

3. Instance based classifiers
   - Use observation directly (no models)
   - e.g. K nearest neighbors

(1) K-Nearest Neighbor

Task	Classification / Regression
Representation	Local Smoothness / training samples
Score Function	EPE with L2 loss ➔ conditional mean (Extra)
Search/Optimization	NA
Models, Parameters	Training Samples

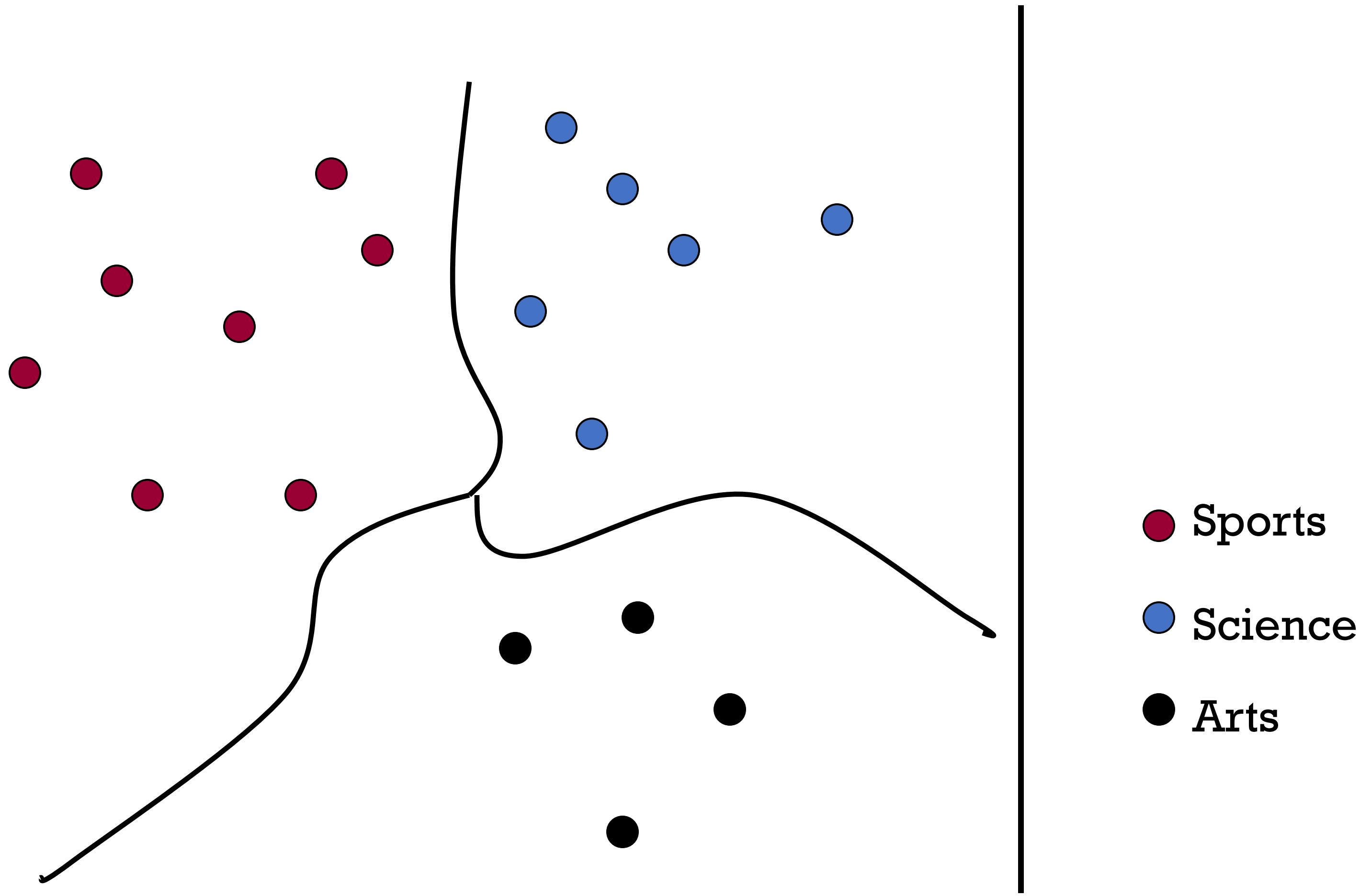
For example: Vector Space Representation of Text

- Each document is a vector, one component for each term (= word).

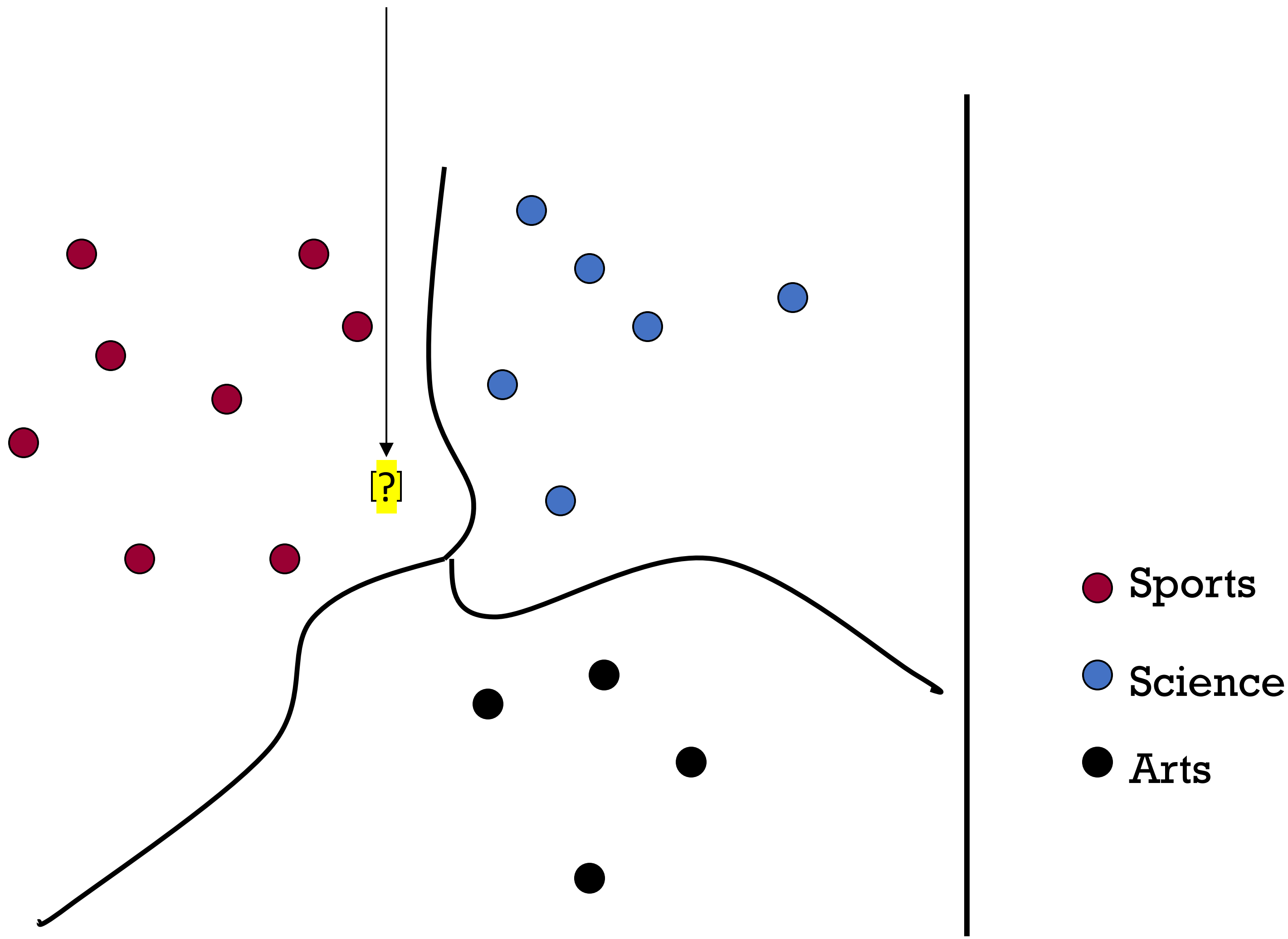
	Doc 1	Doc 2	Doc 3	...
Word 1	3	0	0	...
Word 2	0	8	1	...
Word 3	12	1	10	...
...	0	1	3	...
...	0	0	0	...

- High-dimensional vector space:
  - Terms are axes, 10,000+ dimensions, or even 100,000+
  - Docs are vectors in this space
  - Normally Normalize to unit length.

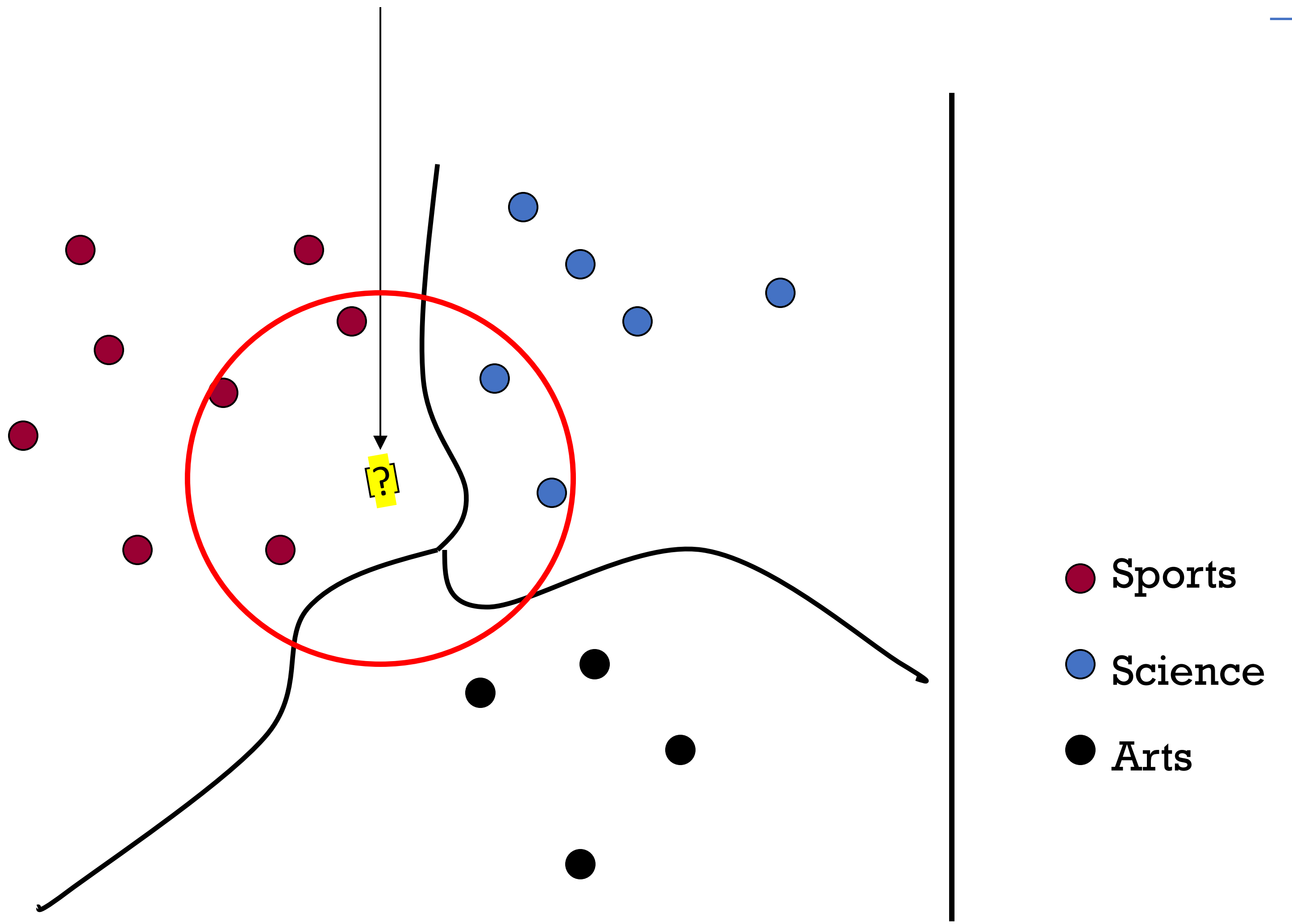
Multiple Classes in a Vector Space



Test Document = ?



K-Nearest Neighbor (kNN) classifier



Instance-based Learning

- Simplest form of learning:
  - Training instances are searched for those that most closely resembles new instance
  - The instances themselves represent the knowledge

- Similarity function defines what's "learned"
- Instance-based learning is *lazy* learning

From witten&eibe

Instance-based Learning

- K-Nearest Neighbor Algorithm
- Weighted Regression
- Case-based reasoning

What makes an Instance-Based Learner?

- A distance metric
- How many nearby neighbors to look at?
- A weighting function (optional)
- How to relate to the local points?

Popular Distance Metric

- Euclidean $$D(x,x') = \sqrt{\sum_i \sigma_i^2 (x_i - x_i')^2}$$
- Or equivalently,

$$D(x,x') = \sqrt{(x-x')^T \Sigma (x-x')}$$

- Other metrics:
  - L_1 norm: |x-x'|
  - L_∞ norm: max |x-x'| (elementwise …)
  - Mahalanobis: where Σ is full, and symmetric
  - Correlation
  - Angle
  - Hamming distance, Manhattan distance
  - …

Feature Scaling in Nearest neighbor method

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5 m to 1.8 m
    - weight of a person may vary from 90 lb to 300 lb
    - income of a person may vary from \$10K to \$1M

$d(\vec{x}_i, \vec{x}_j)$

The relative scalings in the distance metric affect region shapes.

What makes an Instance-Based Learner?

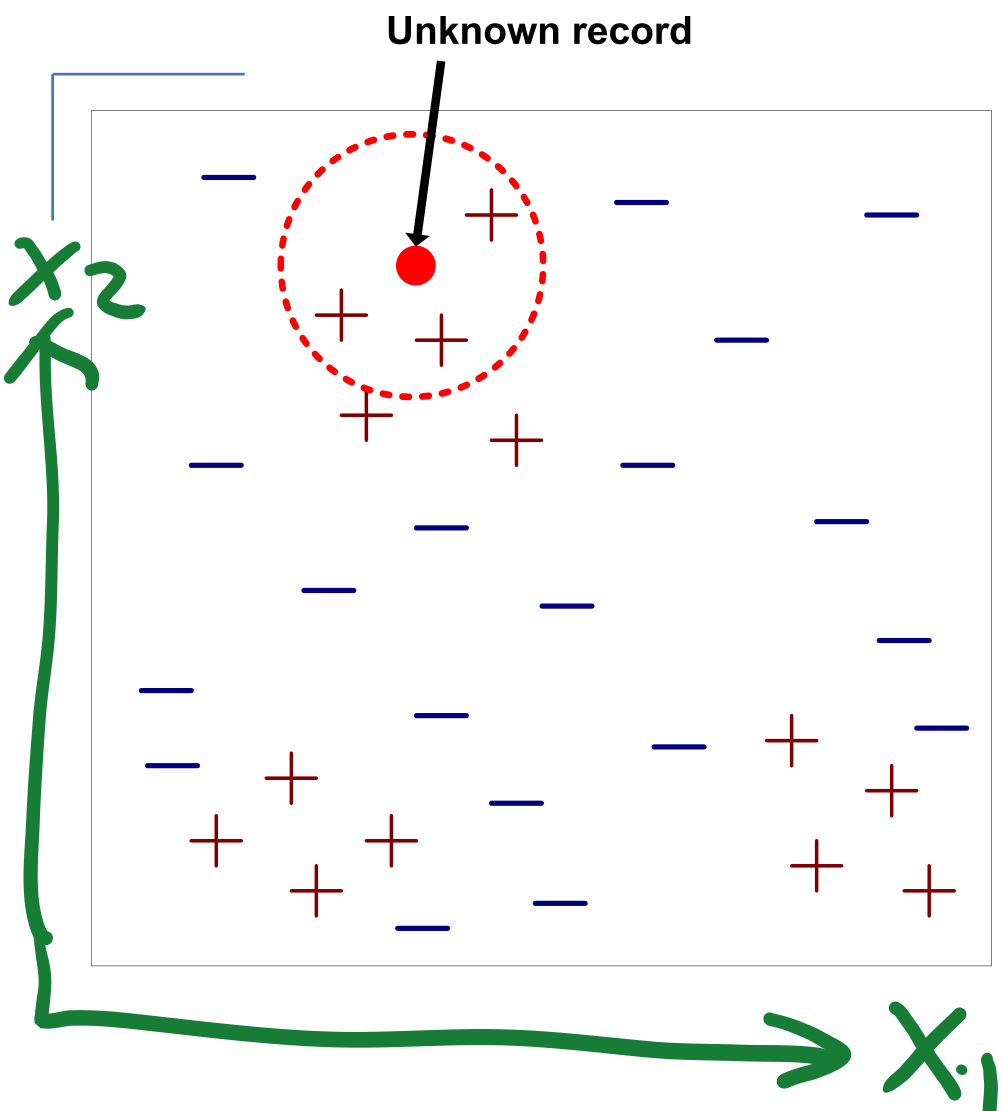
- A distance metric
- How many nearby neighbors to look at?
- A weighting function (optional)
- How to relate to the local points?

Nearest neighbor is instance-based method

Requires three inputs:

1. The set of stored training samples
2. Distance metric to compute distance between samples
3. The value of *k*, i.e., the number of nearest neighbors to retrieve

Nearest neighbor classifiers

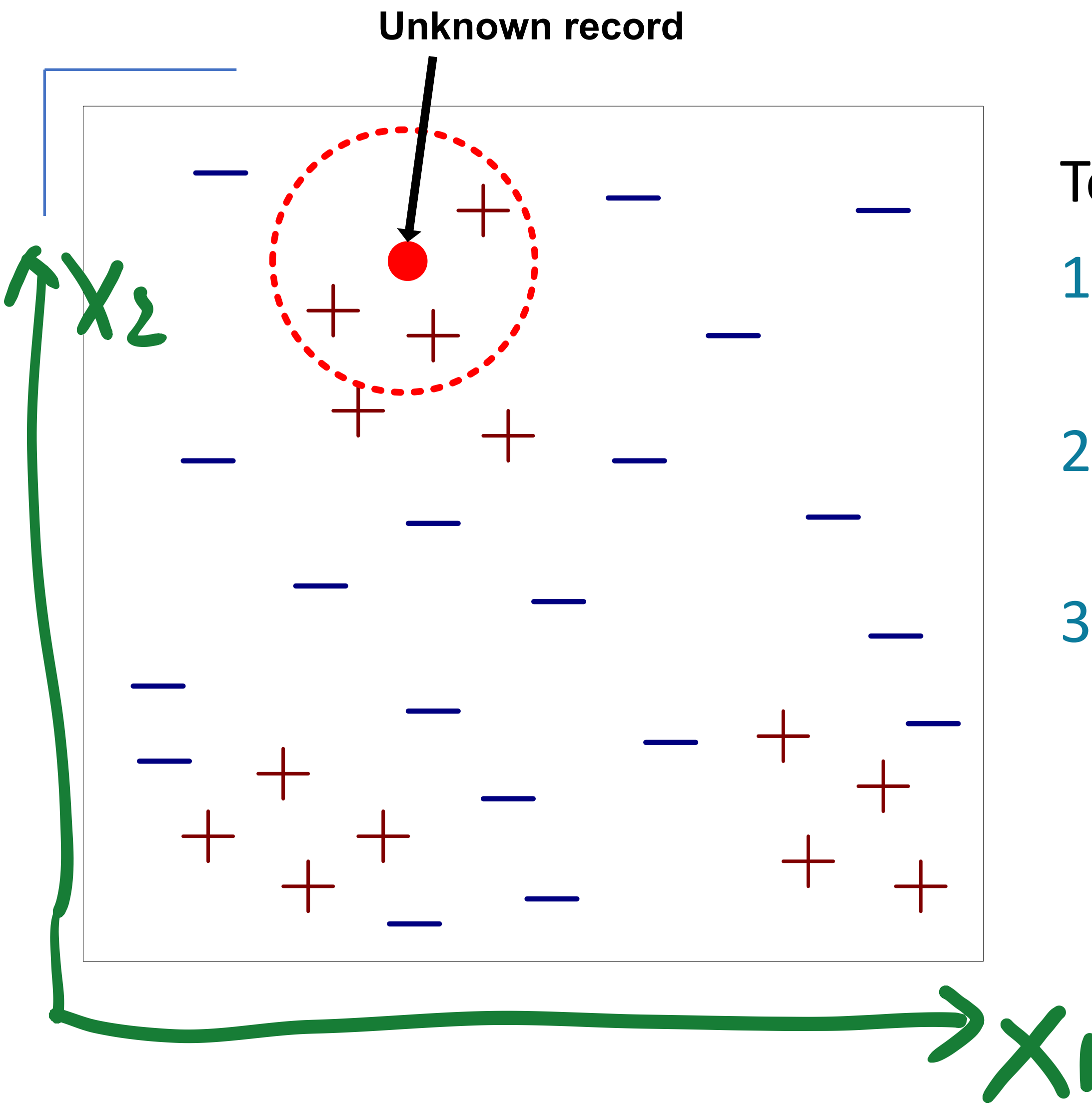


Requires three inputs:

1. The set of stored training samples
2. Distance metric to compute distance between samples
3. The value of *k*, i.e., the number of nearest neighbors to retrieve

$y \in \{+, -\}$

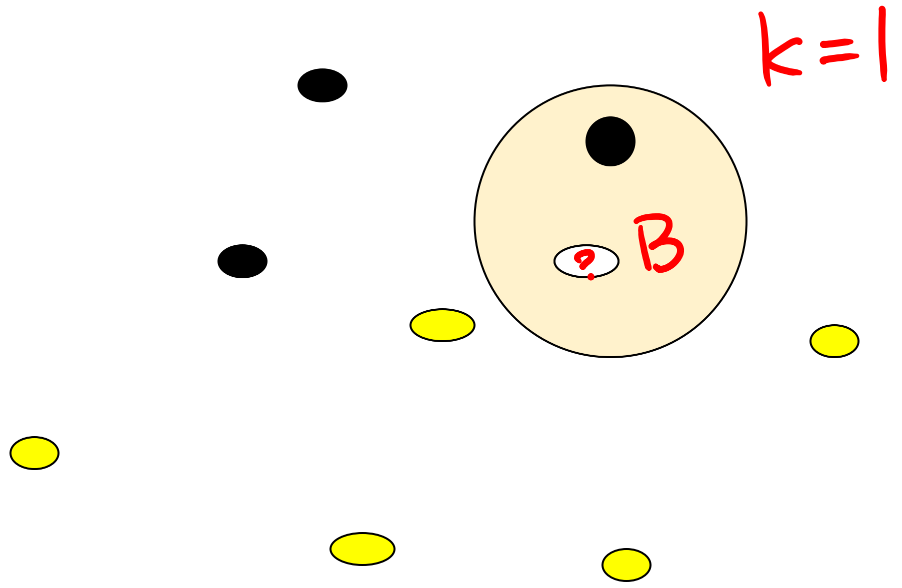
Nearest neighbor classifiers



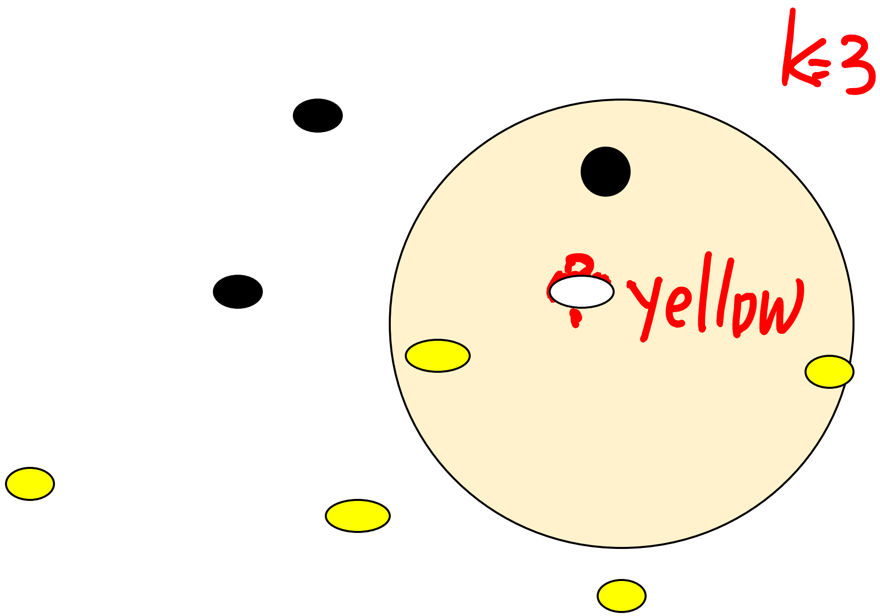
To classify unknown sample:

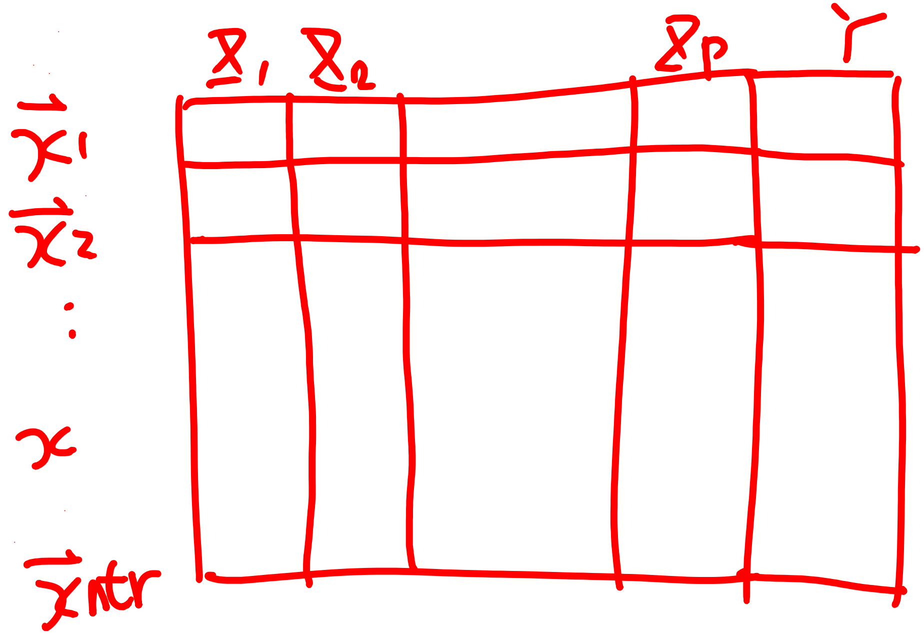
1. Compute distance to training records
2. Identify *k* nearest neighbors
3. Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

1-Nearest Neighbor



3-Nearest Neighbor





$\vec{x}_?$ 

① $d(\vec{x}_i, \vec{x}_j)$

② K

→ Step1: $O(n_{tr}p)$

$$\begin{cases} d(\vec{x}_?, \vec{x}_1) \\ d(\vec{x}_?, \vec{x}_2) \\ \vdots \\ d(\vec{x}_?, \vec{x}_{n_{tr}}) \end{cases}$$

→ step2:
pick top K from n_{tr}

$O(n_{tr})$

→ step3

$$y_? = \frac{1}{K}\sum_{j \in NN(x_?)} y_j$$

K-Nearest Neighbor: How to decide:

- Decision of output value is delayed till a new instance arrives
- Target variable may be discrete or real-valued
  - When target is discrete, the naïve prediction is the majority vote

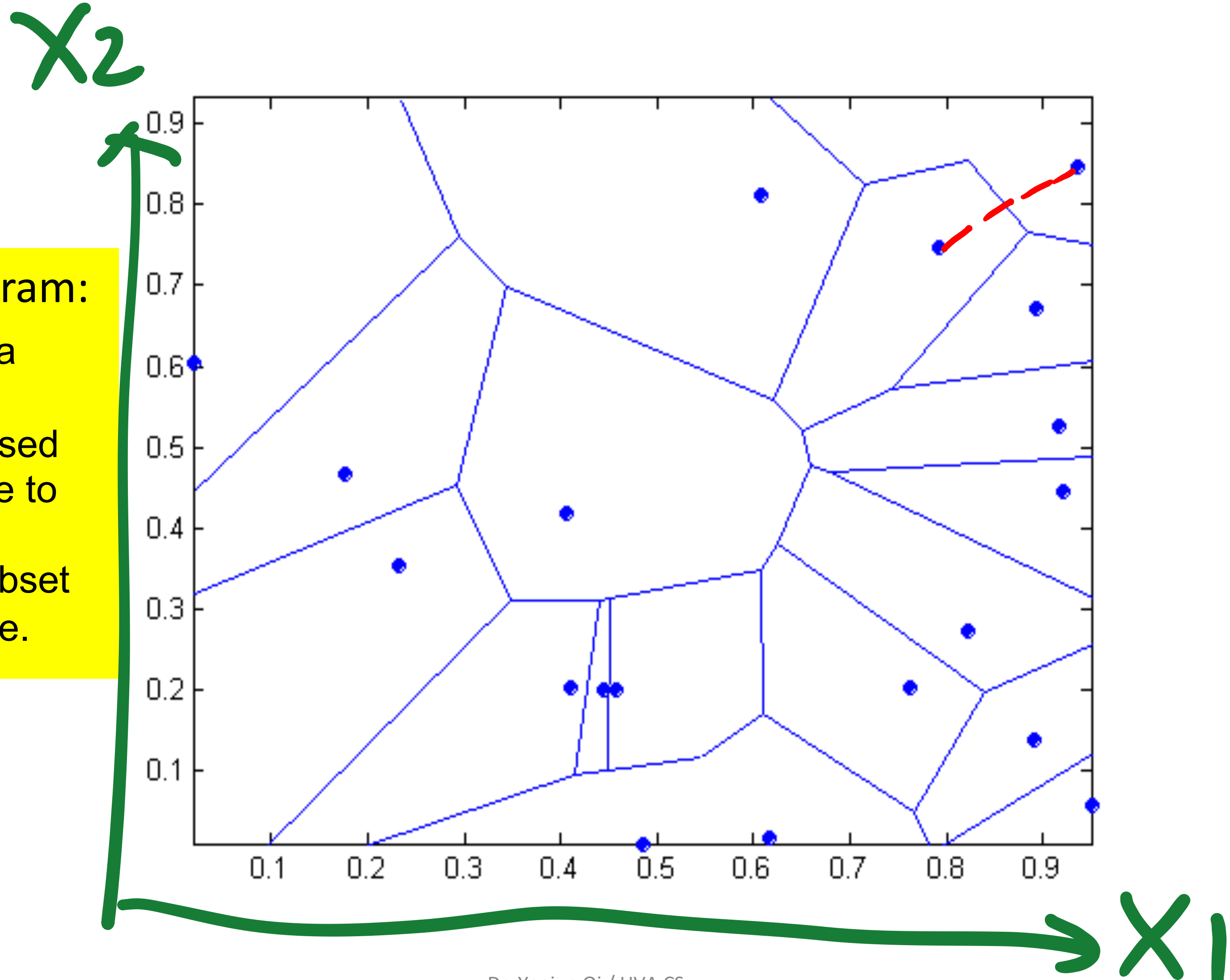
$$y_? = \frac{1}{k} \sum_{j \in kNN(x_?)} y_j$$

$y \in \{0, 1\}$

① majority vote: If $(y_i > t)$

$t = 0.5$

e.g., 1-nearest neighbor



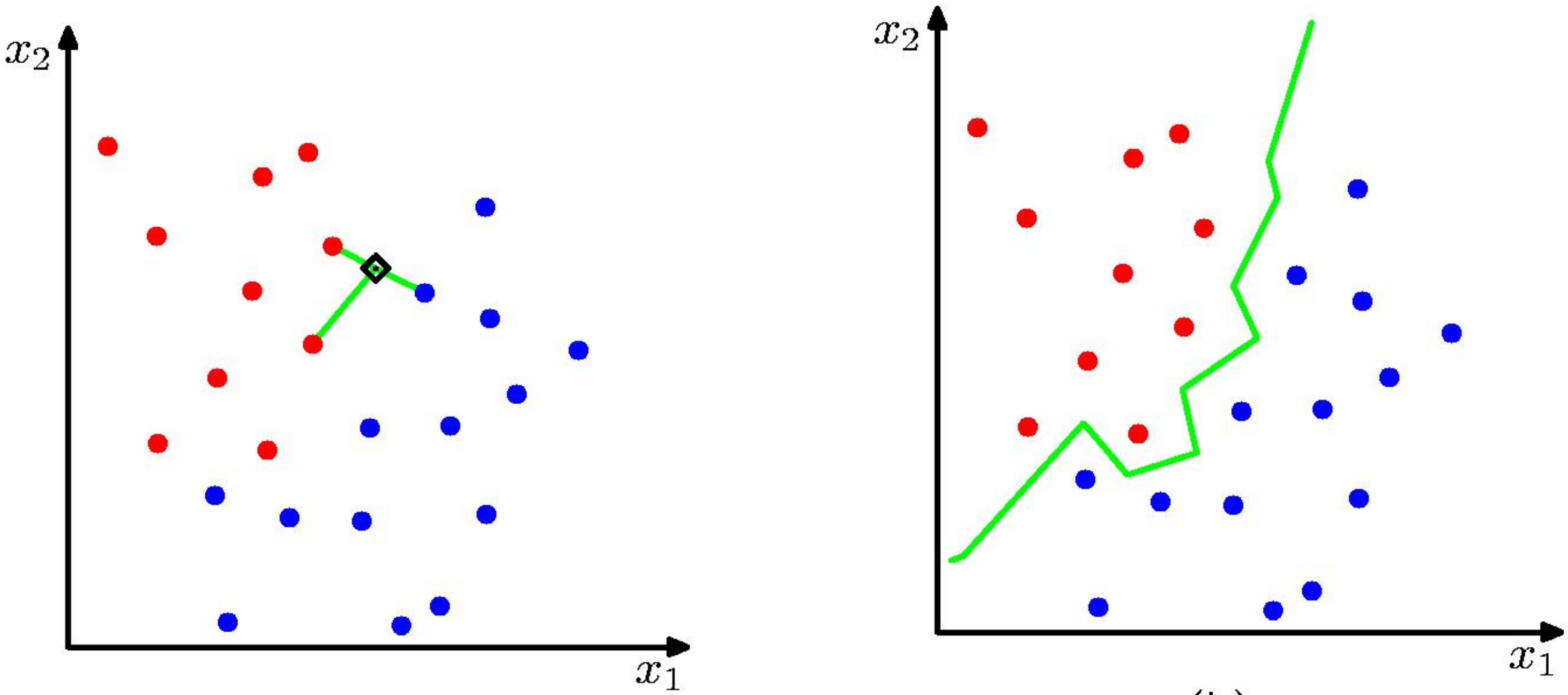
Voronoi diagram:

partitioning of a plane into regions based on distance to points in a specific subset of the plane.

e.g. Decision boundary implemented by 3NN

The boundary is always the perpendicular bisector of the line between two points (Vornoi tesselation)

k-nearest neighbors of a sample x are datapoints that have the *k* smallest distances to x



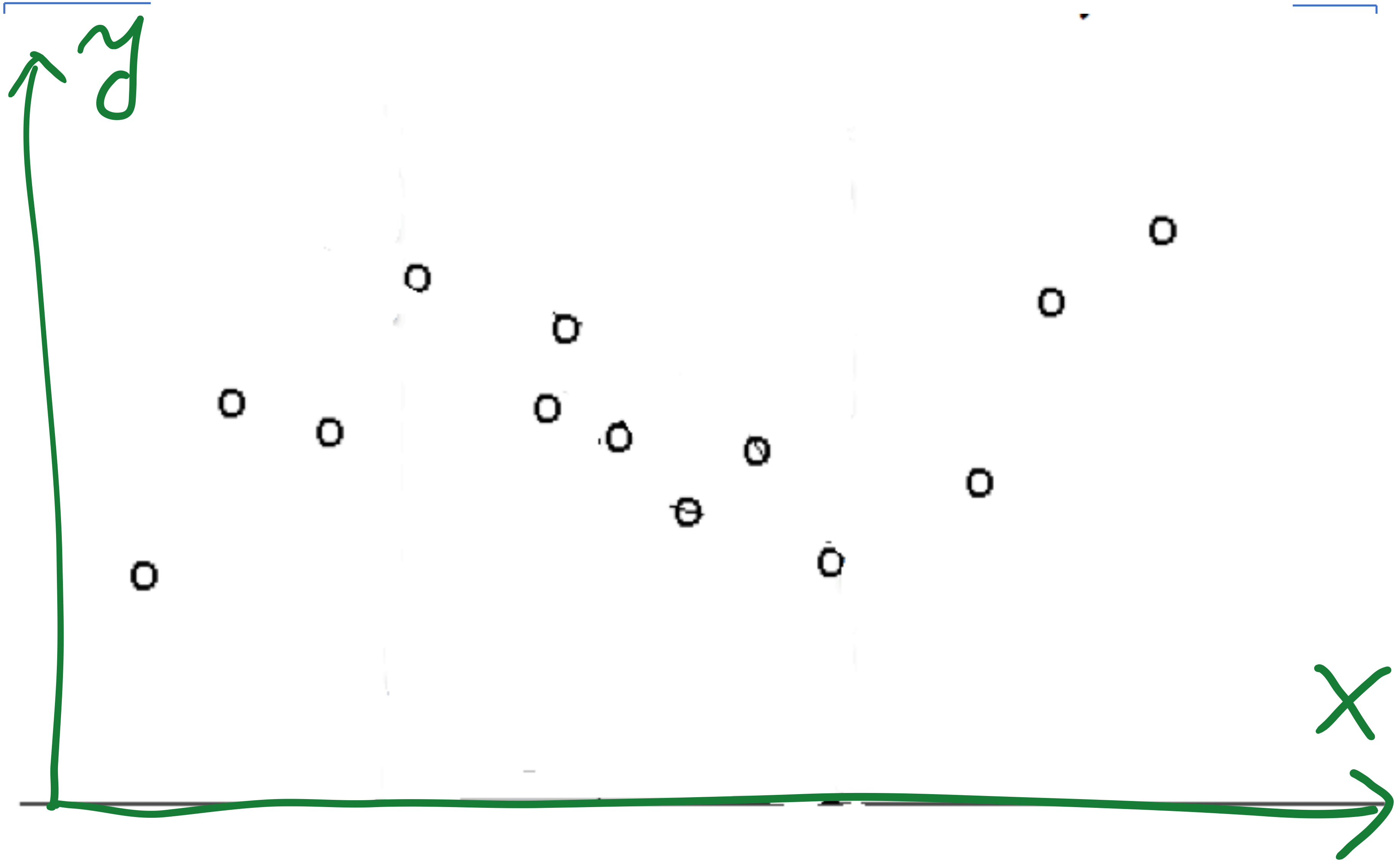
(a) (b)

K-Nearest Neighbor: How to decide:

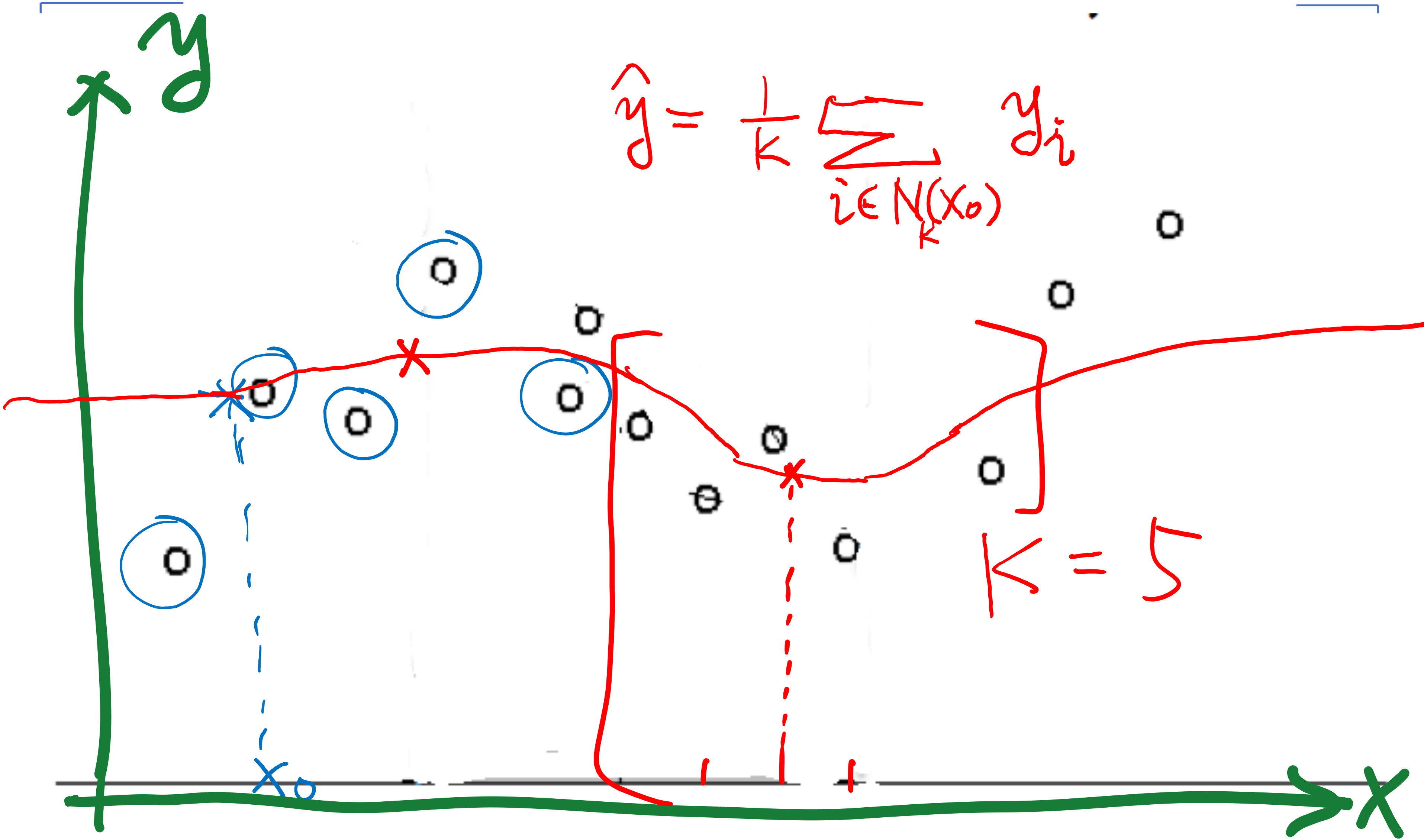
- Decision of output value is delayed till a new instance arrives
- Target variable may be discrete or real-valued
  - When target is discrete, the naïve prediction is the majority vote
  - When target is continuous, the naïve prediction is the mean value of the k nearest training examples

$$\hat{y} = \frac{1}{k} \sum_{i \in N_k(x_0)} y_i$$

Nearest Neighbor (1D input) for Regression

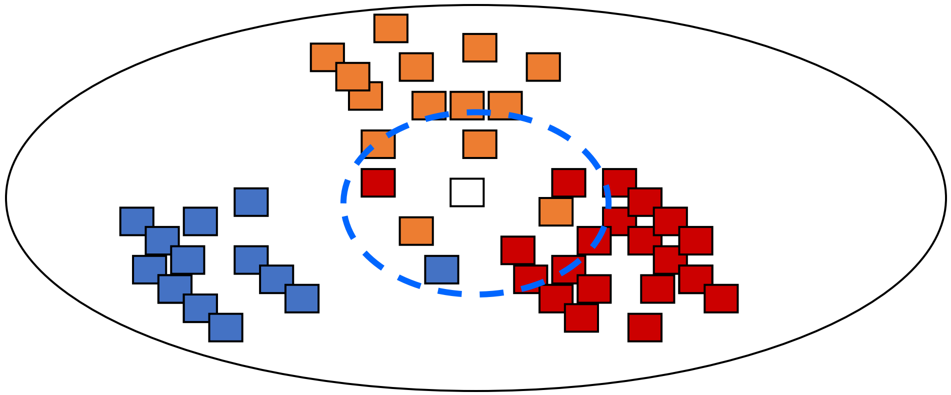


K=5-Nearest Neighbor (1D input) for Regression



Probabilistic Interpretation of KNN

- Estimate conditional probability Pr(y|x)
  - Count of data points in class y in the neighborhood of x



Summary of Nearest neighbor methods

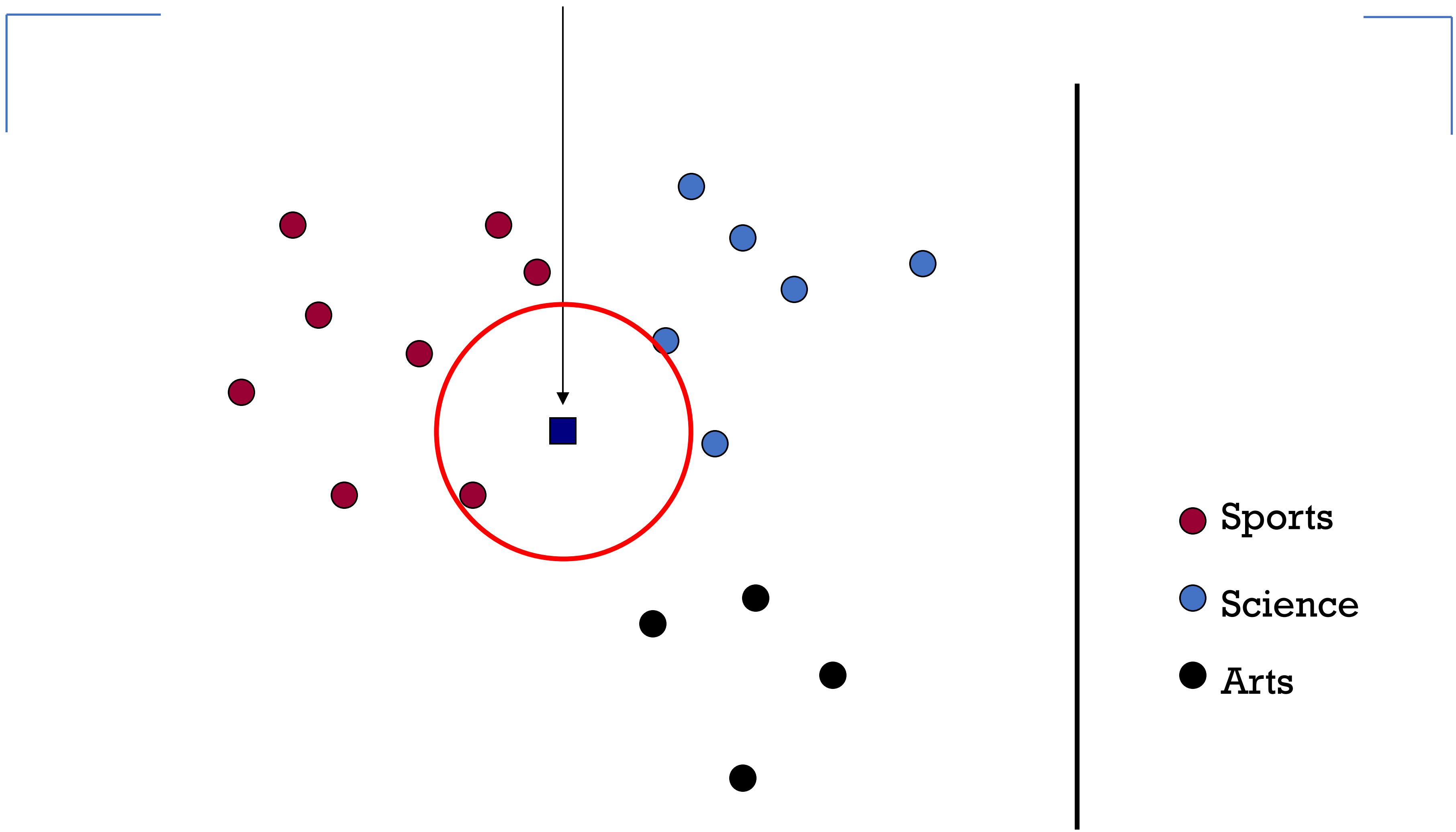
- For regression, average the predictions of the K nearest neighbors.

- For classification, pick the class with the most votes.
    - How should we break ties?
    - E.g., Let the k’th nearest neighbor contribute a count that falls off with k. For example, $1+\frac{1}{2^k}$

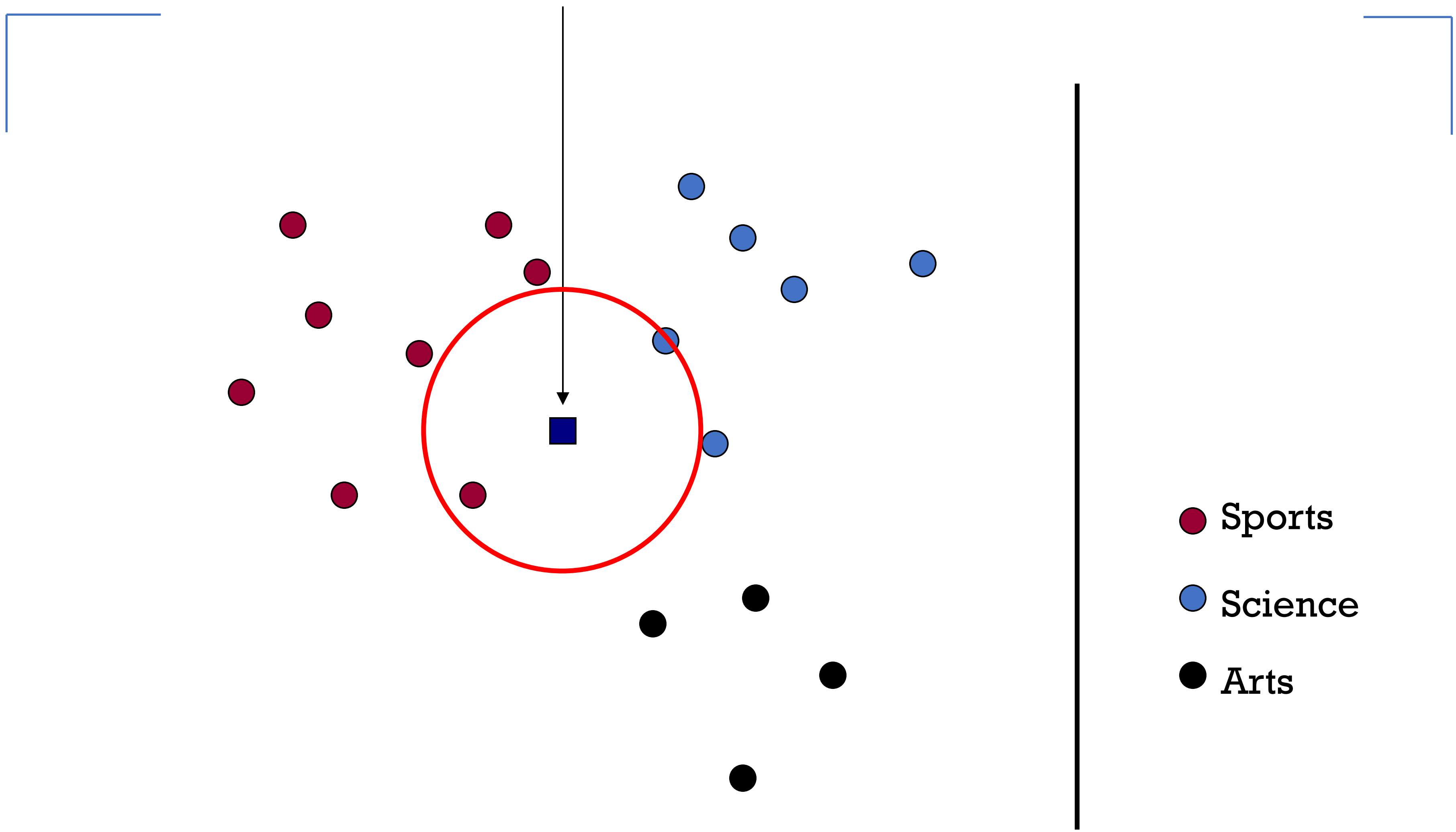
What makes an Instance-Based Learner?

- A distance metric
- How many nearby neighbors to look at?
- A weighting function (optional)
- How to relate to the local points?

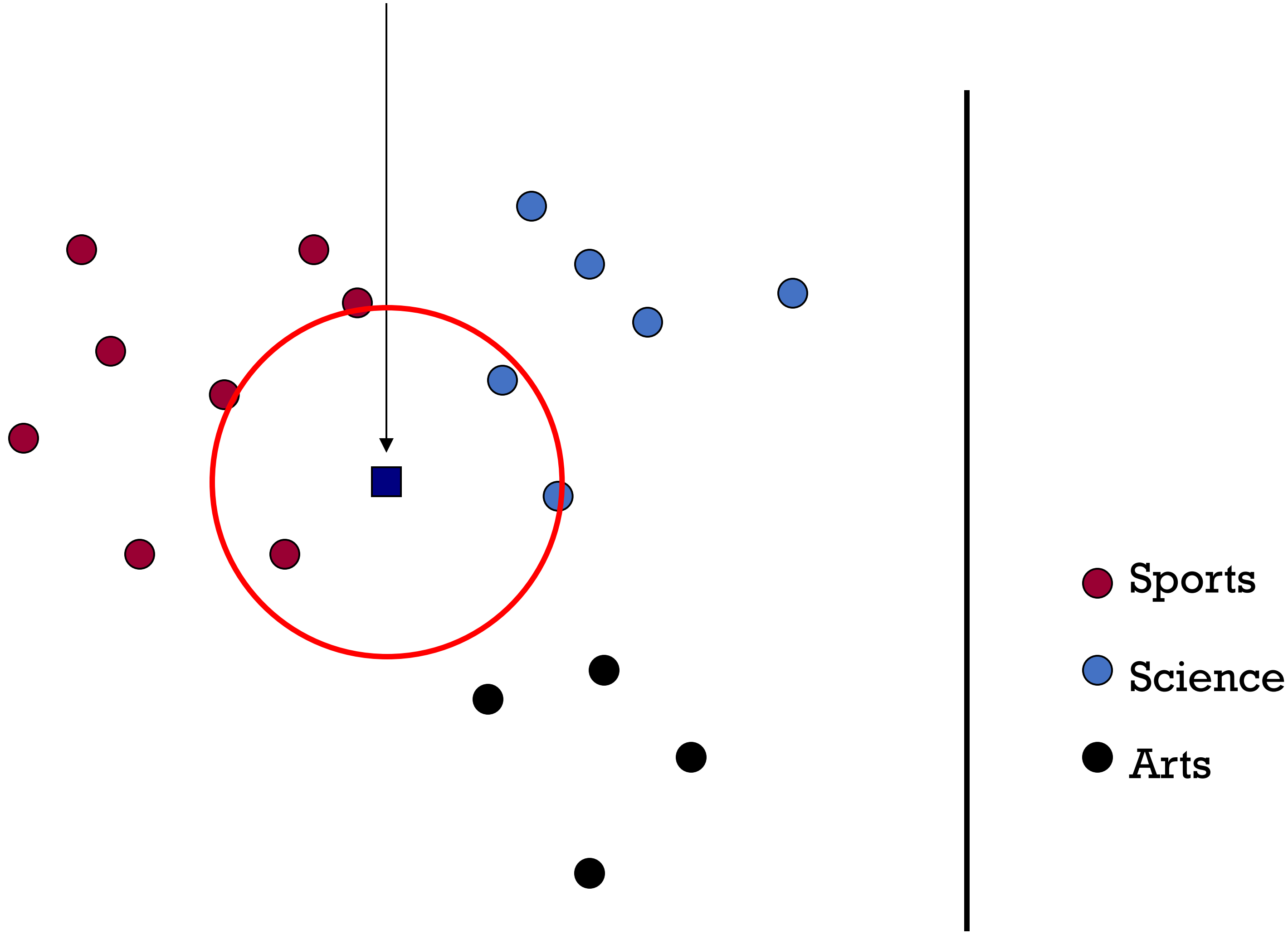
1-Nearest Neighbor (kNN) classifier



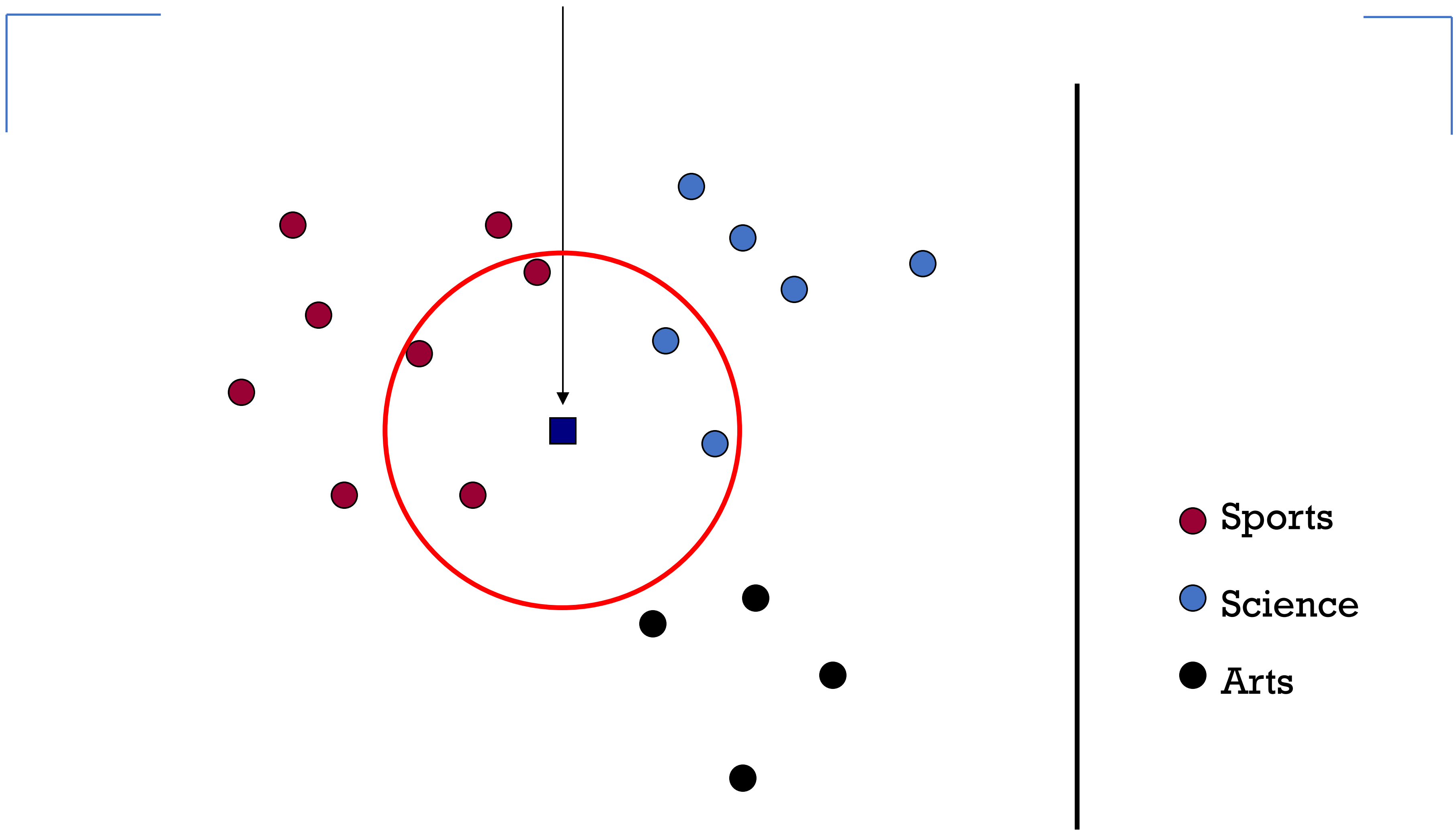
2-Nearest Neighbor (kNN) classifier

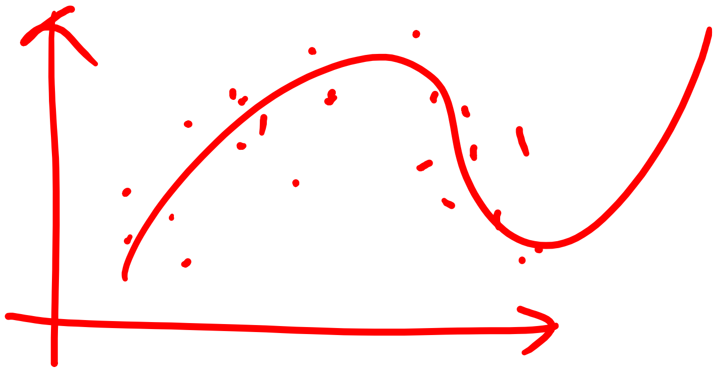


3-Nearest Neighbor (kNN) classifier



5-Nearest Neighbor (kNN) classifier





$d \uparrow$ complexity $\uparrow$

$d \downarrow$ complexity $\downarrow$

Ridge

$\lambda \uparrow$ complexity $\downarrow$

$\lambda \downarrow$ complexity $\uparrow$

$K \uparrow$ complexity $\downarrow$

$K \downarrow$ complexity $\uparrow$

ploy Reg d $\nearrow$

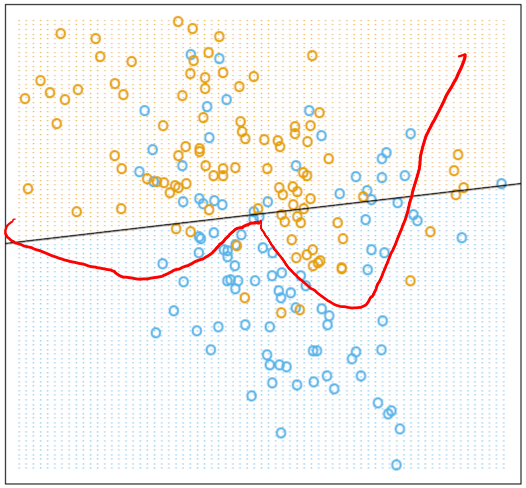
Ridge λ $\swarrow$

KNN k $\swarrow$

model complex small

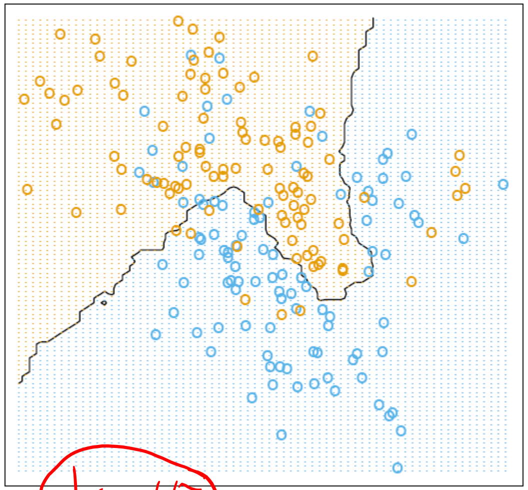
model complexity large

Decision boundaries in Linear vs. kNN models (Later)



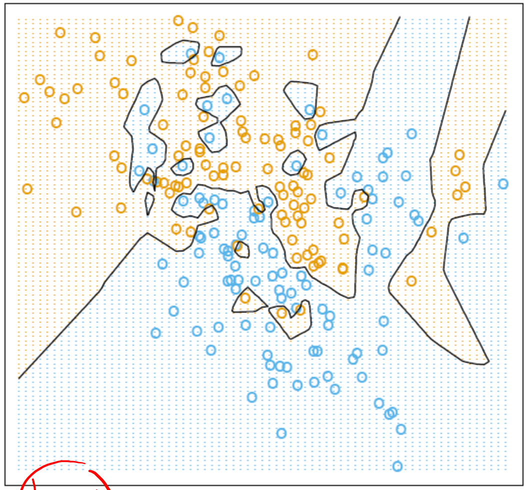
Low Variance / High Bias

linear regression
- global
- stable
- can be inaccurate



k=15

15-nearest neighbor



k=1

1-nearest neighbor

KNN
- local
- accurate
- unstable

Low Bias / High Variance

What ultimately matters: ***GENERALIZATION***

Model Selection for Nearest neighbor classification

- Choosing the value of *k*:
  - If *k* is too small, sensitive to noise points
  - If *k* is too large, neighborhood may include points from other classes

K ↓ flexible / varies a lot

K ↑ Smooth / varies little

- Bias and variance tradeoff
  - A small neighborhood → large variance → unreliable estimation
  - A large neighborhood → large bias → inaccurate estimation

What makes an Instance-Based Learner?

- A distance metric
- How many nearby neighbors to look at?
- A weighting function (optional)
- How to relate to the local points?

Nearest neighbor Variations

- Options for determining the class from nearest neighbor list
  - 1. majority vote of class labels among the *k*-nearest neighbors
  - 2. Weight the votes according to distance
    - example: weight factor w = 1 / d^2

$$y_? = \frac{1}{k} \sum_{j \in kNN(x_?)} w_j y_j$$

$$w_j = \frac{1}{d(x_j, x_?)}$$

Spurious or less relevant points need to be downweighted

Another Weighted kNN

- Weight the contribution of each close neighbor based on their distances
- Weight function

$$w(\mathbf{x}, \mathbf{x}_i) = \exp\left(-\lambda|\mathbf{x} - \mathbf{x}_i|_2^2\right)$$

- Prediction

$$\Pr(y|\mathbf{x}) = \frac{\sum_{i=1}^{n} w(\mathbf{x}, \mathbf{x}_i)\delta(y, y_i)}{\sum_{i=1}^{n} w(\mathbf{x}, \mathbf{x}_i)}$$

$$\delta(y, y_i) = \begin{cases} 1 & y = y_i \\ 0 & y \neq y_i \end{cases}$$

Variants: Distance-Weighted k-Nearest Neighbor Algorithm

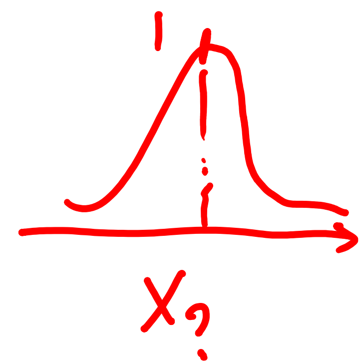
- Assign weights to the neighbors based on their "distance" from the query point
  - Weight "may" be inverse square of the distances w = 1 / d^2

- Extreme Option: All training points may influence a particular instance
  - E.g., Shepard's method/ Modified Shepard, … by Geospatial Analysis

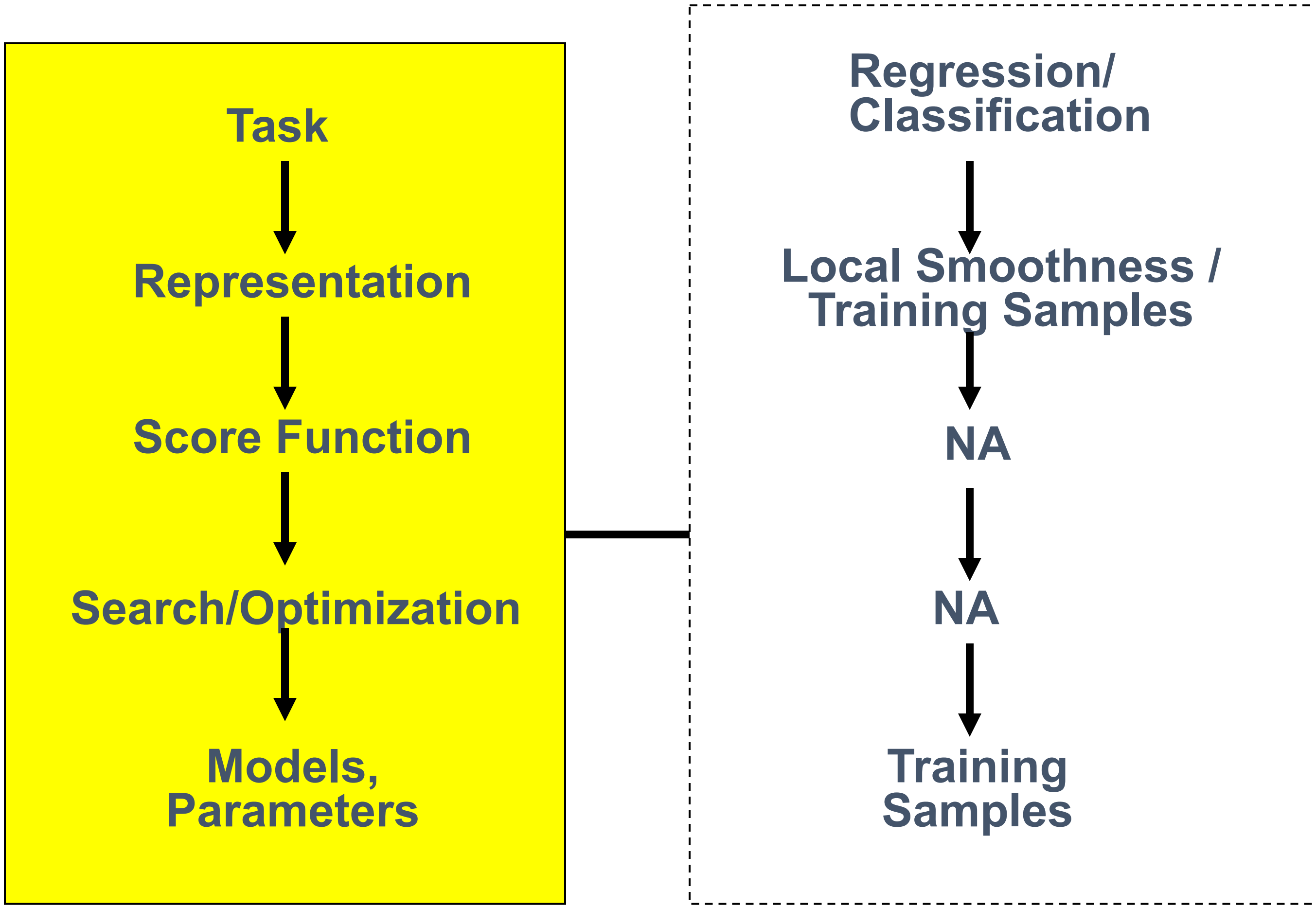
e.g. $\hat{y} = \frac{1}{K} \sum_{j \in N_k(x_0)} W_i y_j$

RBF

$K_\lambda(x_i, x_0)$



K-Nearest Neighbor



Computational Scalable?

Computational Time Cost

	Train (n)	Test (m=1)
Linear Regression	$O(np^2 + p^3)$	$O(p)$
KNN	$O(1)$	$O(np) +$ $O(\text{sort } n\text{-}K)$???

$p = 30{,}000$

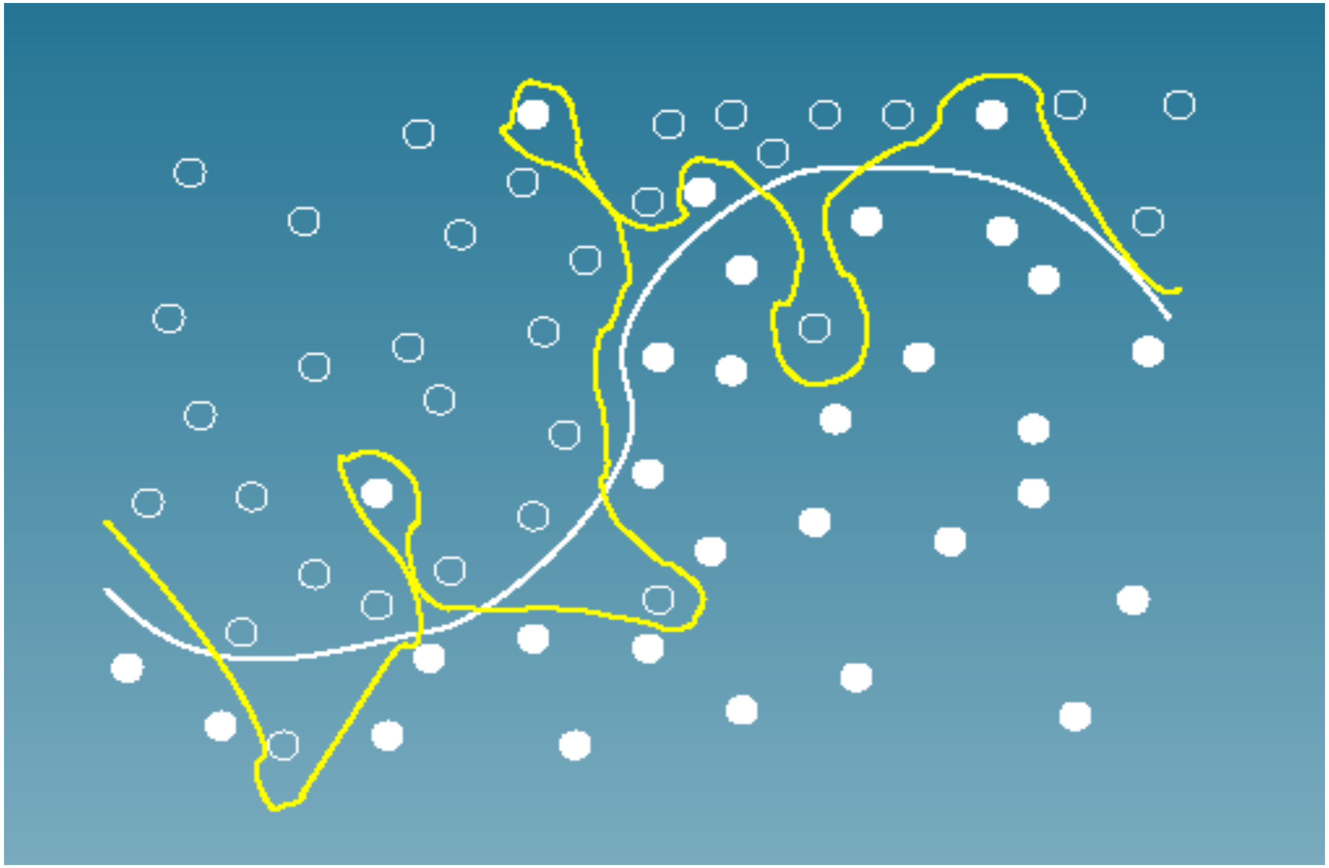
$n = 20{,}000$

K-Nearest Neighbor

Task	Regression/ Classification
Representation	Local Smoothness / Training Samples
Score Function	NA
Search/Optimization	NA
Models, Parameters	Training Samples

Asymptotically Sound?

Is kNN ideal? … See Extra



References

- Prof. Tan, Steinbach, Kumar's "Introduction to Data Mining" slide
- Prof. Andrew Moore's slides
- Prof. Eric Xing's slides
- Hastie, Trevor, et al. *The elements of statistical learning*. Vol. 2. No. 1. New York: Springer, 2009.