Cornell University

Instance-Based Learning

CS4780/5780 – Machine Learning
Fall 2011

Thorsten Joachims
Cornell University

Reading: Mitchell Chapter 1 & Sections 8.1 - 8.2

Concept Learning

- Definition:

  Acquire an operational definition of a general category of objects given positive and negative training examples.

Cornell University

Concept Learning Example

correct (3)	color (2)	original (2)	presentation (3)	binder (2)	A+Homework
complete	yes	yes	clear	no	yes
complete	no	yes	clear	no	yes
partial	yes	no	unclear	no	no
complete	yes	yes	clear	yes	yes

Instance Space X: Set of all possible objects described by attributes (often called features).

Concept c: Subset of objects from X (c is unknown).

Target Function f: Characteristic function indicating membership in c based on attributes (i.e. label)(f is unknown).

Training Data S: Set of instances labeled with target function.

Concept Learning as Learning a Binary Function

- Task:
  - Learn (to imitate) a function $f: X \rightarrow \{+1,-1\}$
- Training Examples:
  - Learning algorithm is given the correct value of the function for particular inputs → **training examples**
  - An **example** is a pair (x, y), where *x* is the input and $y=f(x)$ is the output of the target function applied to *x*.
- Goal:
  - Find a function

    $$h: X \rightarrow \{+1,-1\}$$

    that approximates

    $$f: X \rightarrow \{+1,-1\}$$

    as well as possible.

K-Nearest Neighbor (KNN)

- Given: Training data $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$
  - Attribute vectors: $\vec{x}_i \in X$
  - Label: $y_i \in Y = \{-1, +1\}$
- Parameter:
  - Similarity function: $K : X \times X \longrightarrow \Re$
  - Number of nearest neighbors to consider: k
- Prediction rule
  - New example x' with
  - K-nearest neighbors: k train examples with largest $K(\vec{x}_i, \vec{x}')$

$$h(\vec{x}') = \arg\max_{y \in Y} \left\{ \sum_{i \in knn(\vec{x}')} 1_{[y_i = y]} \right\}$$

Cornell University

KNN Example

	correct (3)	color (2)	original (2)	presentation (3)	binder (2)	A+Homework
1	complete	yes	yes	clear	no	yes / +1
2	complete	no	yes	clear	no	yes / +1
3	partial	yes	no	unclear	no	no / -1
4	complete	yes	yes	clear	yes	yes / +1

- How will new examples be classified?
  - Similarity function?
  - Value of k?

$$h(\vec{x}') = \operatorname*{arg\,max}_{y \in Y} \left\{ \sum_{i \in knn(\vec{x}')} 1_{[y_i = y]} \right\}$$

Cornell University

Weighted K-Nearest Neighbor

- Given: Training data $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$
  - Attribute vectors: $\vec{x}_i \in X$
  - Target attribute: $y_i \in \{-1, +1\}$
- Parameter:
  - Similarity function: $K : X \times X \longrightarrow \Re$
  - Number of nearest neighbors to consider: k
- Prediction rule
  - New example x'
  - K-nearest neighbors: k train examples with largest $K(\vec{x}_i, \vec{x}')$

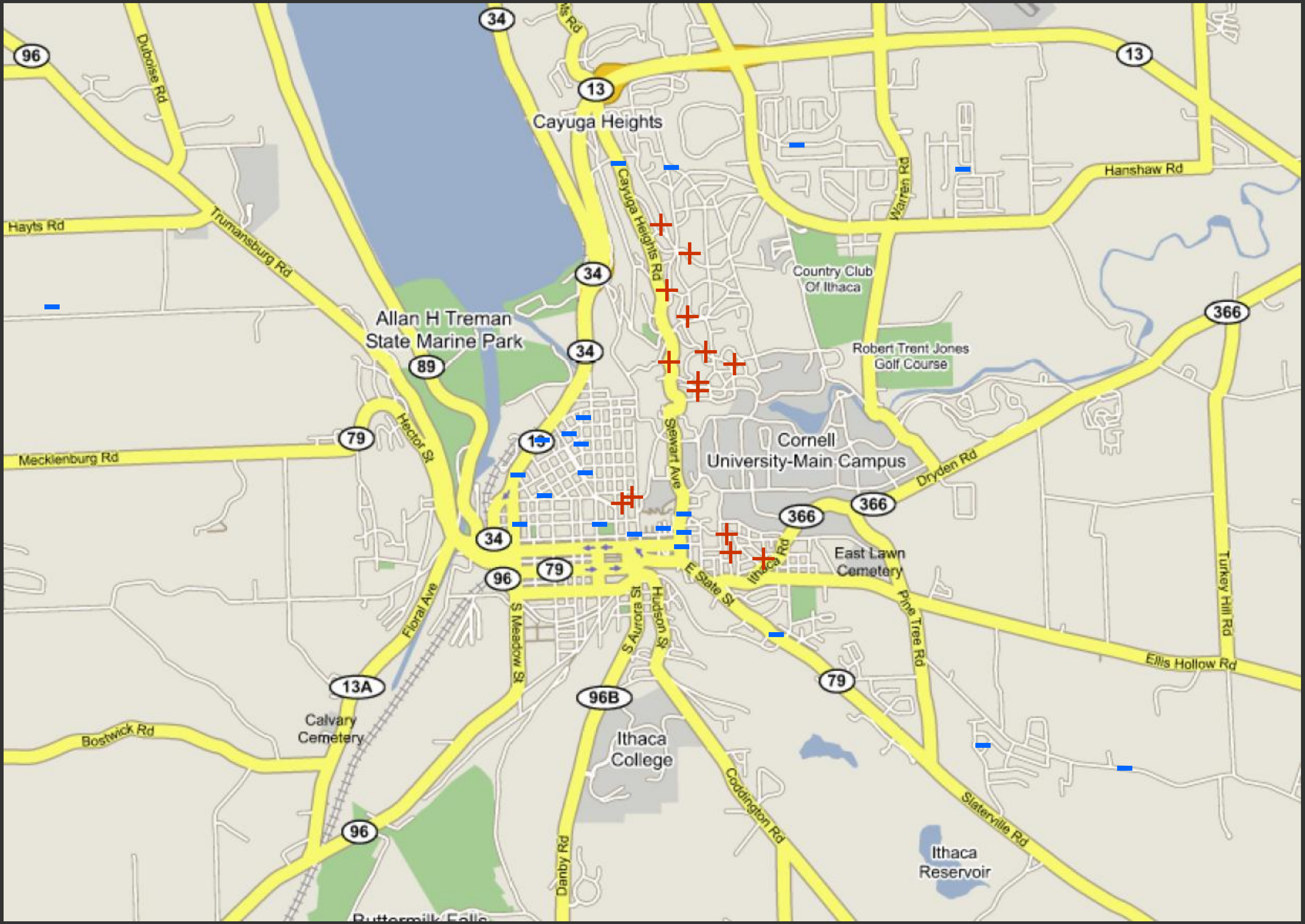
$$h(\vec{x}') = \arg\max_{y \in Y} \left\{ \sum_{i \in knn(\vec{x}')} 1_{[y_i = y]} K(\vec{x}_i, \vec{x}') \right\}$$

Types of Attributes

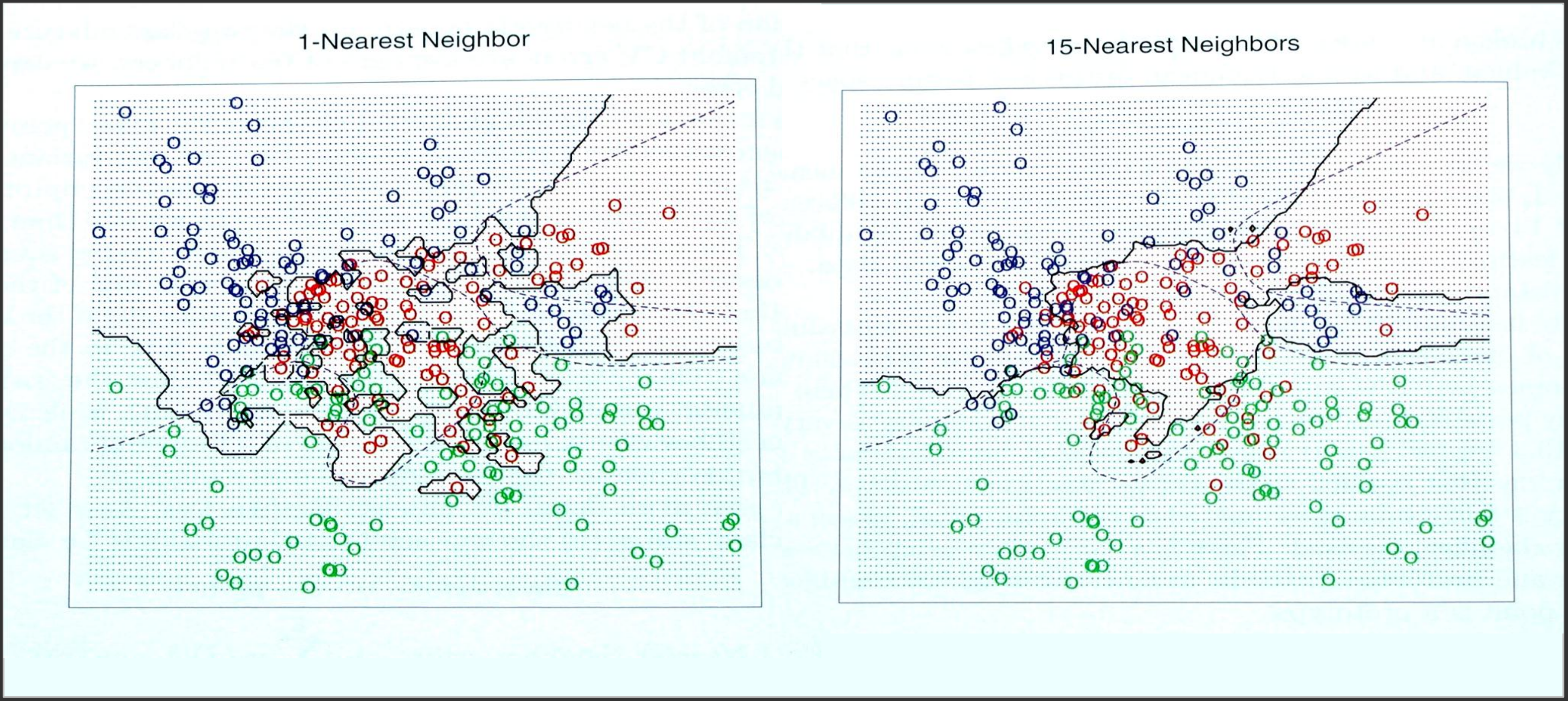
- Symbolic (nominal)
  - *EyeColor {brown, blue, green}*
- Boolean
  - *alife {TRUE,FALSE}*
- Numeric
  - Integer: *age* [0, 105]
  - Real: *length*
- Structural
  - Natural language sentence: parse tree
  - Protein: sequence of amino acids

Cornell University

Example: Expensive Housing (>$200 / sqft)



Example: Effect of k



Hastie, Tibshirani, Friedman 2001

Supervised Learning

- Task:
  - Learn (to imitate) a function f: $X \rightarrow Y$
- Training Examples:
  - Learning algorithm is given the correct value of the function for particular inputs → **training examples**
  - An **example** is a pair $(x, f(x))$, where x is the input and $f(x)$ is the output of the function applied to x.
- Goal:
  - Find a function

    $$h: X \rightarrow Y$$

    that approximates

    $$f: X \rightarrow Y$$

    as well as possible.

Weighted K-Nearest Neighbor for Regression

- Given: Training data $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$
  - Attribute vectors: $\vec{x}_i \in X$
  - Target attribute: $y_i \in \Re$
- Parameter:
  - Similarity function: $K : X \times X \longrightarrow \Re$
  - Number of nearest neighbors to consider: k
- Prediction rule
  - New example x'
  - K-nearest neighbors: k train examples with largest $K(\vec{x}_i, \vec{x}')$

$$h(\vec{x}') = \frac{\sum_{i \in knn(\vec{x}')} y_i K(\vec{x}_i, \vec{x}')}{\sum_{i \in knn(\vec{x}')} K(\vec{x}_i, \vec{x}')}$$

Cornell University

Collaborative Filtering