

# Instance-Based Learning

CS4780/5780 – Machine Learning  
Fall 2013

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.

Also called: binary classification, binary supervised learning,...

# Concept Learning Example

	<b>correct</b> (complete, partial, guessing)	<b>color</b> (yes, no)	<b>original</b> (yes, no)	<b>presentation</b> (clear, unclear, cryptic)	<b>binder</b> (yes, no)	<b>A+</b>
1	complete	yes	yes	clear	no	yes
2	complete	no	yes	clear	no	yes
3	partial	yes	no	unclear	no	no
4	complete	yes	yes	clear	yes	yes

**Instance Space X:** Set of all possible objects describable 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  $\rightarrow$  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  $((\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n))$ 
  - Attribute vectors:  $\vec{x}_i \in X$
  - Labels:  $y_i \in Y$
- Parameter:
  - Similarity function:  $K : X \times X \rightarrow \mathfrak{R}$
  - Number of nearest neighbors to consider:  $k$
- Prediction rule
  - New example  $x'$
  - K-nearest neighbors:  $k$  train examples with largest  $K(\vec{x}_i, \vec{x}')$

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

# KNN Example

	<b>correct</b> (complete, partial, guessing)	<b>color</b> (yes, no)	<b>original</b> (yes, no)	<b>presentation</b> (clear, unclear, cryptic)	<b>binder</b> (yes, no)	<b>A+</b>
1	complete	yes	yes	clear	no	yes
2	complete	no	yes	clear	no	yes
3	partial	yes	no	unclear	no	no
4	complete	yes	yes	clear	yes	yes

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

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

# Weighted K-Nearest Neighbor

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

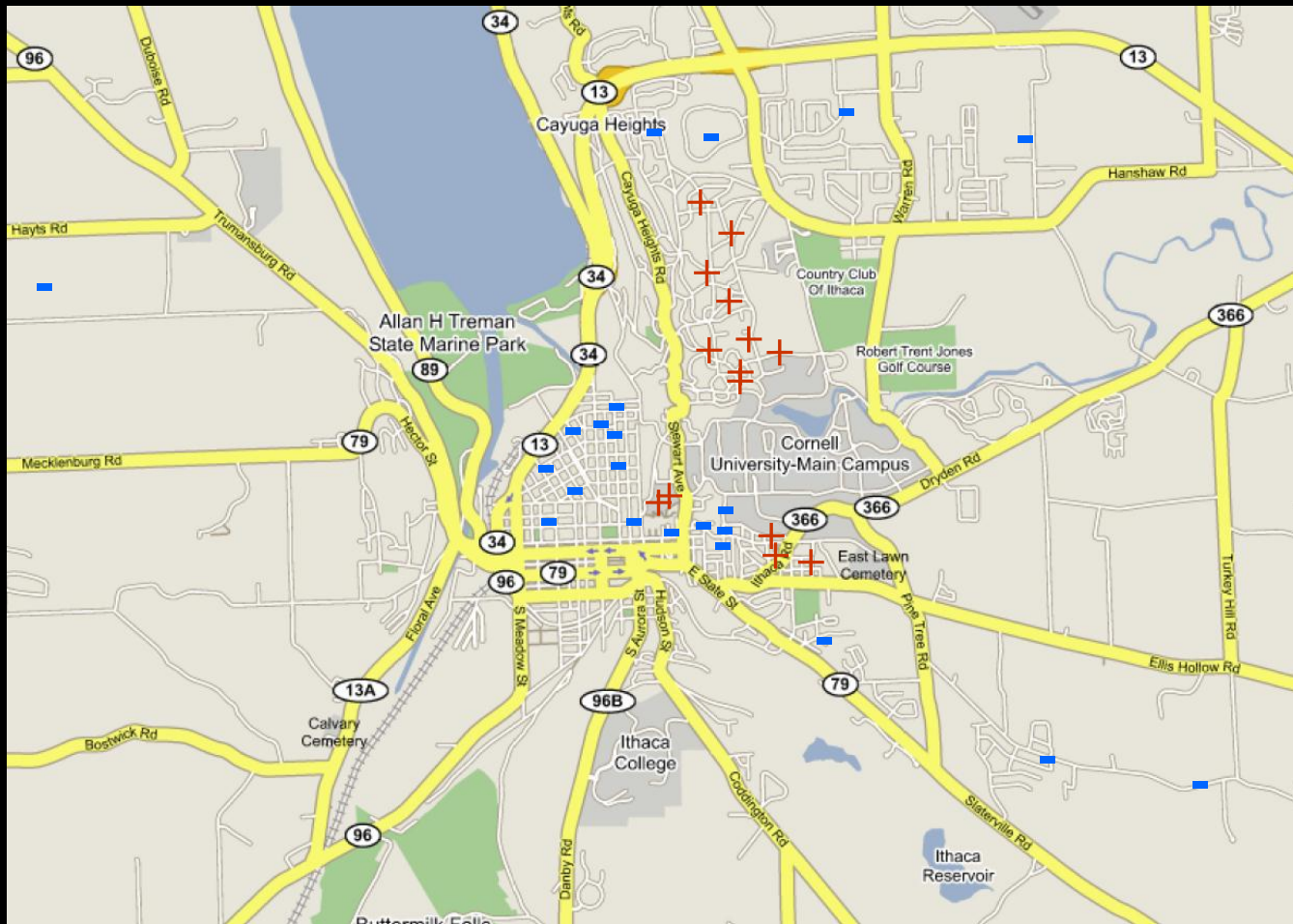
$$h(\vec{x}') = \arg \max_{y \in Y} \left\{ \sum_{i \in knn(\vec{x}')} 1_{[y_i=y]} K(\vec{x}_i, \vec{x}') \right\}$$

# Types of Attributes

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

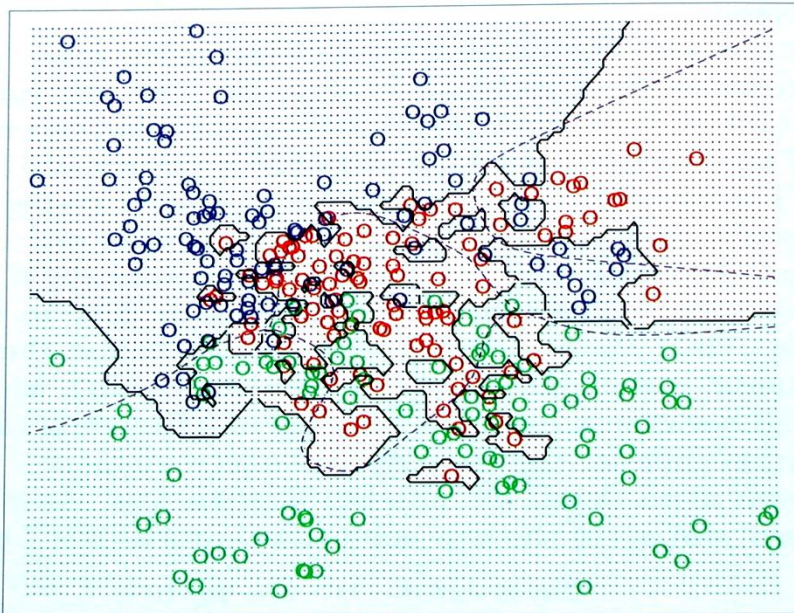


# Example: Expensive Housing (>\$200 / sqft)

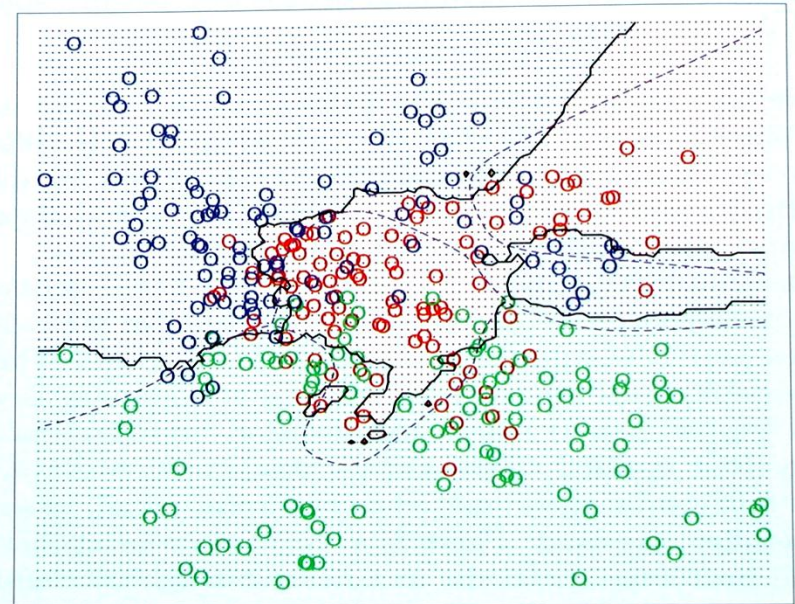


# Example: Effect of $k$

1-Nearest Neighbor



15-Nearest Neighbors



*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  $\rightarrow$  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-NN for Regression

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

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

# Collaborative Filtering



Rating Matrix	$m_1$	$m_2$	$m_3$	$m_4$	$m_5$	$m_6$
$u_1$		1	5		3	5
$u_2$		5	1	1	3	1
$u_3$		2	4		1	5
$u$	?	1	4	?	?	?

