Data mining & Machine Learning

CS 373
Purdue University

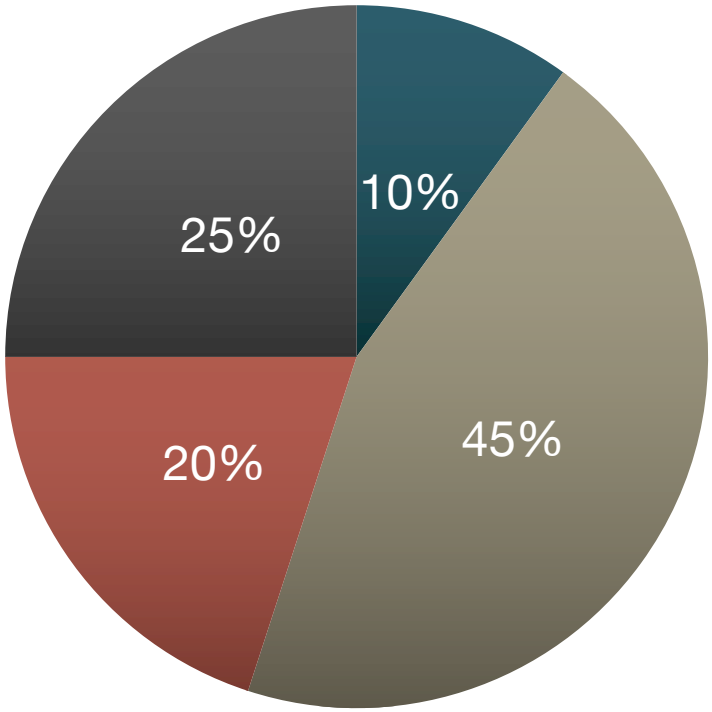
Dan Goldwasser
dgoldwas@purdue.edu

Today's Lecture

A deeper look into what is machine learning

- *Define the data mining process*
- *Key ML Principles: Model, Protocol, Algorithm*
- *Specific algorithm – K Nearest Neighbors*

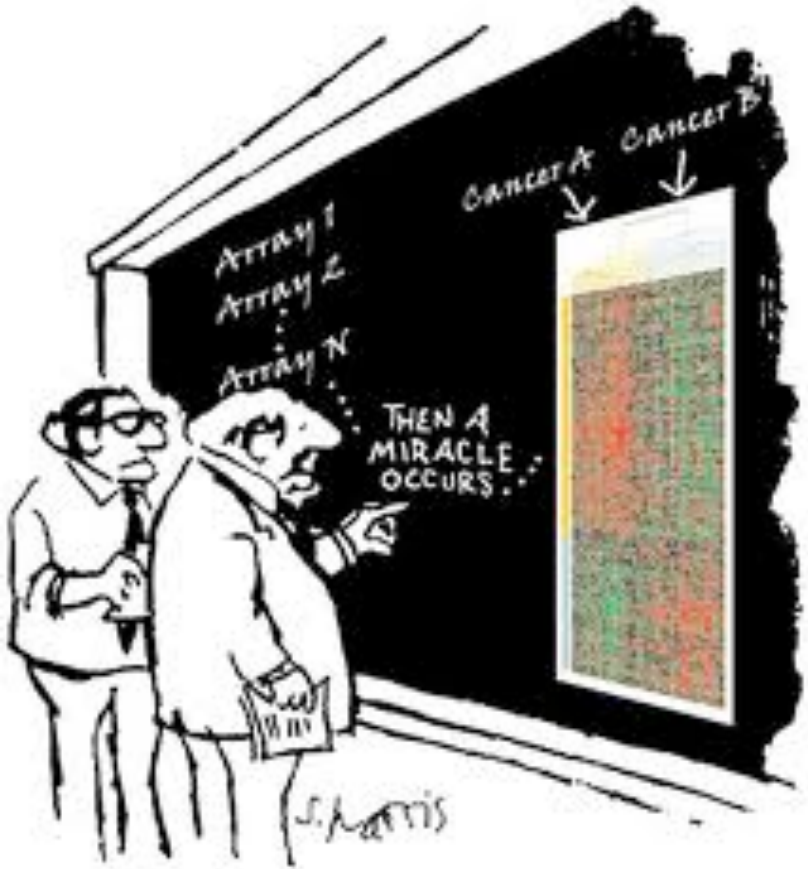
Grading


10%
45%
20%
25%
Quizzes and participation
Homework
Midterm
Final

Machine Learning – short version

- A new paradigm for *telling a computer what to do!*
- **Traditionally:**
  - Write lines of code for different tasks
- **Machine learning**
  - Write code once (learning algorithm)
  - Supply **data** for different tasks
- Key issue: **Generalization**
  - How to generalize from training examples to new data
  - Identify patterns in the data and abstract from ***training*** examples to ***testing*** examples

Let's demystify the process



"I THINK YOU SHOULD BE MORE EXPLICIT HERE IN STEP TWO."

The Process

Data

Selection

Target data

Preprocessing

Processed data

Knowledge

Interpretation evaluation

Machine Learning

Patterns

Mining

Noisy Data

- We want to build a content recommendation system
  - *Our data consists of all the movies order on netflix.*
- After analyzing the data, we conclude:
  - *The taste of 3 year-old kids is most similar to the taste to 5 year-old kids*
  - *But also to 45 year-old adults.*
- How can you explain it?

Noise in the data can be **real noise** (measurement error), but it can also be the results of making the **wrong assumptions** about the data.

Key Principles of ML

- *We will look into several learning **protocols**, using different learning **algorithms**, for learning different **models**.*

- **Model**: function mapping inputs to outputs
- **Algorithm**: used for constructing a model, based on data
- **Protocol**: the settings in which the algorithm learns.

Learning Model

- We think about learning as producing a function mapping input to outputs, based on data
  - E.g., **spam: email → Boolean**

- A **model** is the type of function the learner uses
  - Linear functions, non-linear functions
  - Decision trees
  - Ensembles of classifier

- The key question is **expressivity -** what can the model represent

Learning Models: Representation

- We think about learning as producing a function mapping input to outputs, based on data
  - E.g., **spam: email → Boolean**

What is the domain and range of this function?

- *The **input** space defines the data representation*
- *The **output** space* defines the learning task.
  - *Binary, multiclass, continuous (regression), structure..*

Data Representation

- *Choice of **data structure** for representing individual and collections of measurements*
  - **Individual measurements**: single observations (e.g., person's date of birth, product price)
  - **Collections of measurements**: sets of observations that describe an instance (e.g., person, product)
- Choice of representation determines applicability of algorithms and can impact modeling effectiveness
- **Issues**: data sampling, data cleaning, feature construction

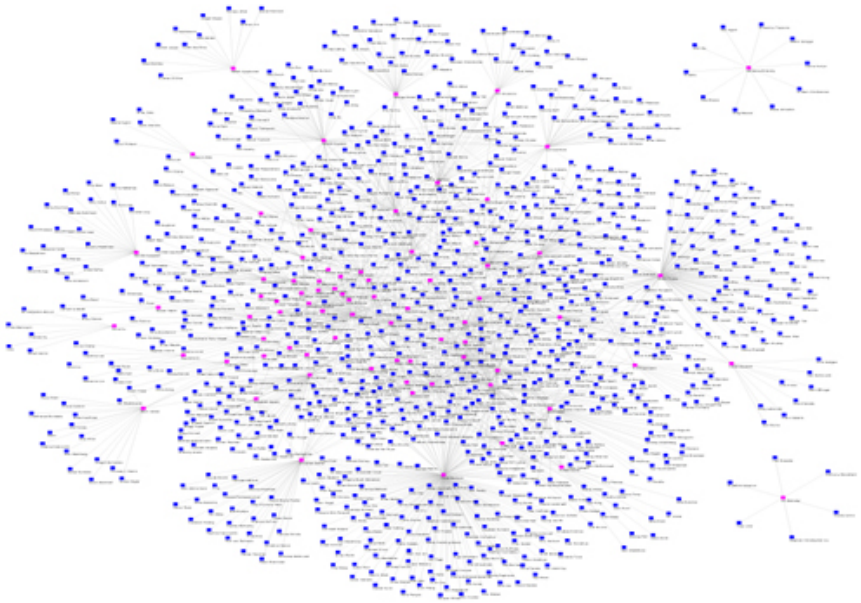
Data representation: Table/vectors

Fraud	Age	Degree	StartYr	Series7
+	22	Y	2005	N
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y
+	24	N	2006	N
-	29	N	2003	N

N instances X *p* attributes

Data Representation

- *In realistic settings we usually do not get a data table that is to work with.*
- *We have to identify relevant properties in the data*
  - *Also known as* ***features***
- ***What are the options we had for the Badges game?***
- ***Open question****: can we* ***learn*** *a good representation?*
  - *What are the properties of good representations?*



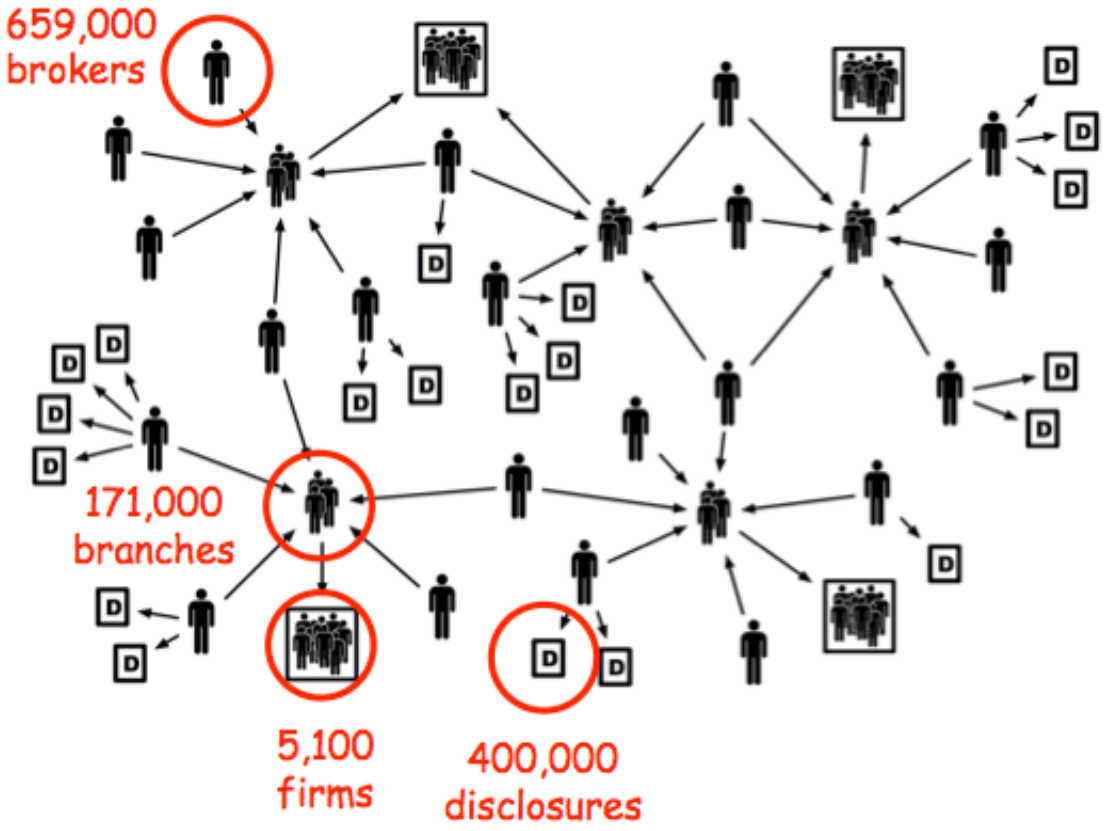
$$\begin{bmatrix}
0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&1&0&0&0&1&0&0&0&0\\
0&1&0&0&0&1&0&0&0&0&1&0&0&0&1&0&0&0\\
0&0&1&1&0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&1\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0
\end{bmatrix}$$

?

Luke, I am your father.

(0,0,1,0,0,2,0,..,0)

Data representation: Relational/graph data

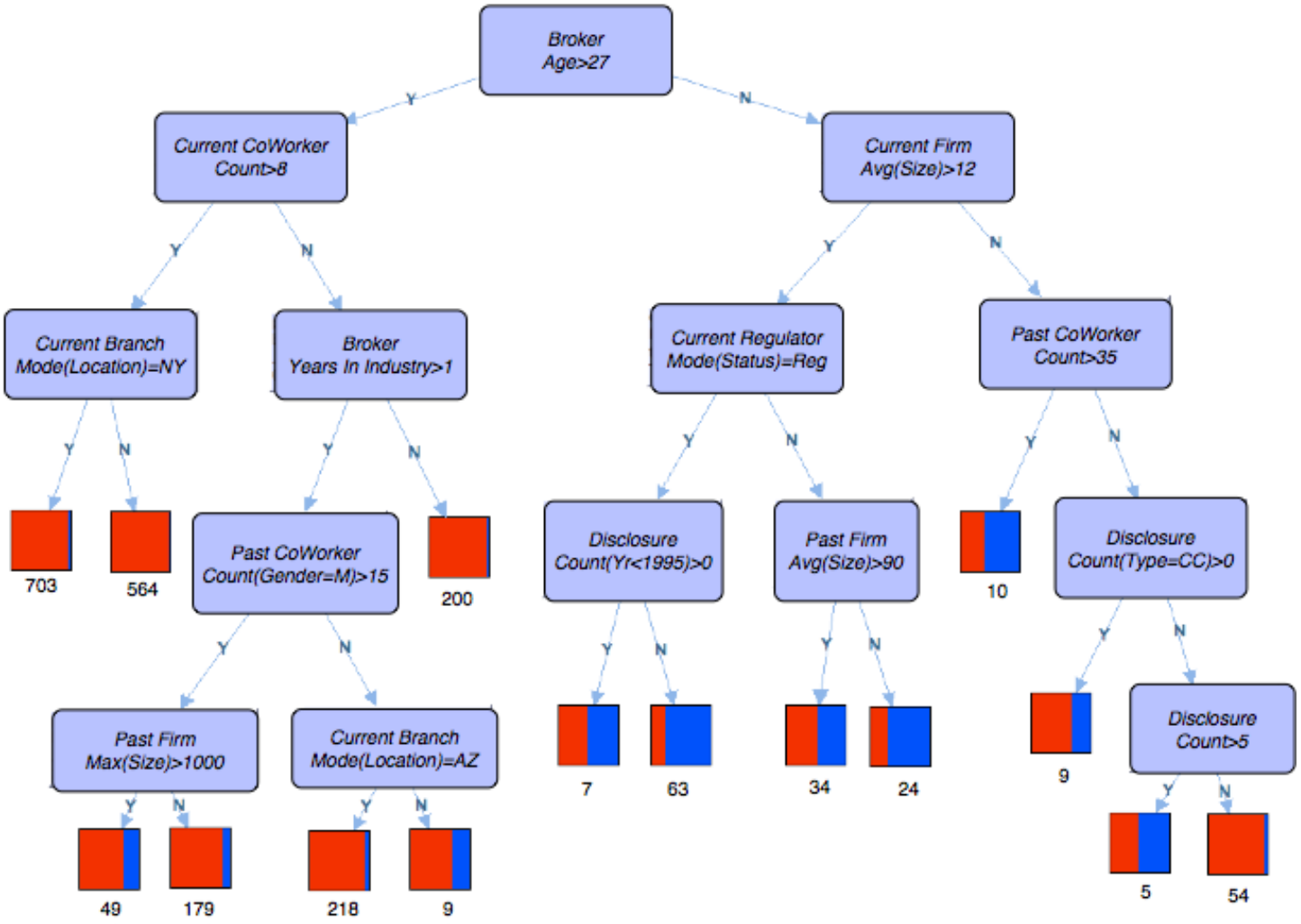


Learning Model (*Hypothesis* Space)

- *Underlying structure of the model or patterns that we seek from the data*
  - Specifies the models/patterns that could be returned as the results of the learning algorithm
  - Defines the **model space** that algorithms search over (i.e., all possible models/patterns)

- Examples:
  - **If-then rule**
    ```
    If short closed car then toxic chemicals
    ```
  - **Conditional probability distribution**
    ```
    P(fraud |age,degree, series7,startYr )
    ```
  - **Decision tree**

Learning Model: Classification tree



Each node corresponds to a feature; each leaf a class label or probability distribution

Learning Model: Regression model

$$y = \beta_1 x_1 + \beta_2 x_2 \ldots + \beta_0$$

- X are predictor variables
- Y is response variable
- Example:
  - Predict number of disclosures given income and trading history

Learning Protocol

- **Supervised learning**
  - Human (*teacher)* supplies a labeled examples
  - *Learner* has to learn a model using this data
- **Unsupervised learning**
  - No *teacher*, *learner* has only unlabeled examples
  - Data mining: finding patterns in unlabeled data
- **Semi-supervised learning**
  - *Learner* has access to *both* labeled and unlabeled examples
- **Weakly-supervised learning**
  - We have access to noisy labels

Learning Protocol

- **Active learning**
  - *Learner* and *teacher* interact
  - *Learner* can ask questions
- **Reinforcement learning**
  - Learner learns by interacting with the environment

- Why is Reinforcement learning **different/similar** to Supervised learning? Active learning?

Supervised Learning: Classification

- **Classification**: mapping data into categories
  - *Write a face recognition program*
  - *Determine if an English sentence is grammatical*
  - *Distinguish between normal and cancerous cells*

- **Can't we just write code?**

  - Provide **labeled examples** and let a classifier distinguish between the two classes
  - What are the labeled examples in each case?

Supervised Learning: Classification

Class	Outlook	Temperature	Windy?
Play	Sunny	Low	Yes
No play	Sunny	High	Yes
No play	Sunny	High	No
Play	Overcast	Low	Yes
Play	Overcast	High	No
Play	Overcast	Low	No
No play	Rainy	Low	Yes
Play	Rainy	Low	No

- Predict the **class label** (*play tennis)*
- **Features**: outlook, temperature, windy
- **Feature values**: can be binary, categorical, continuous

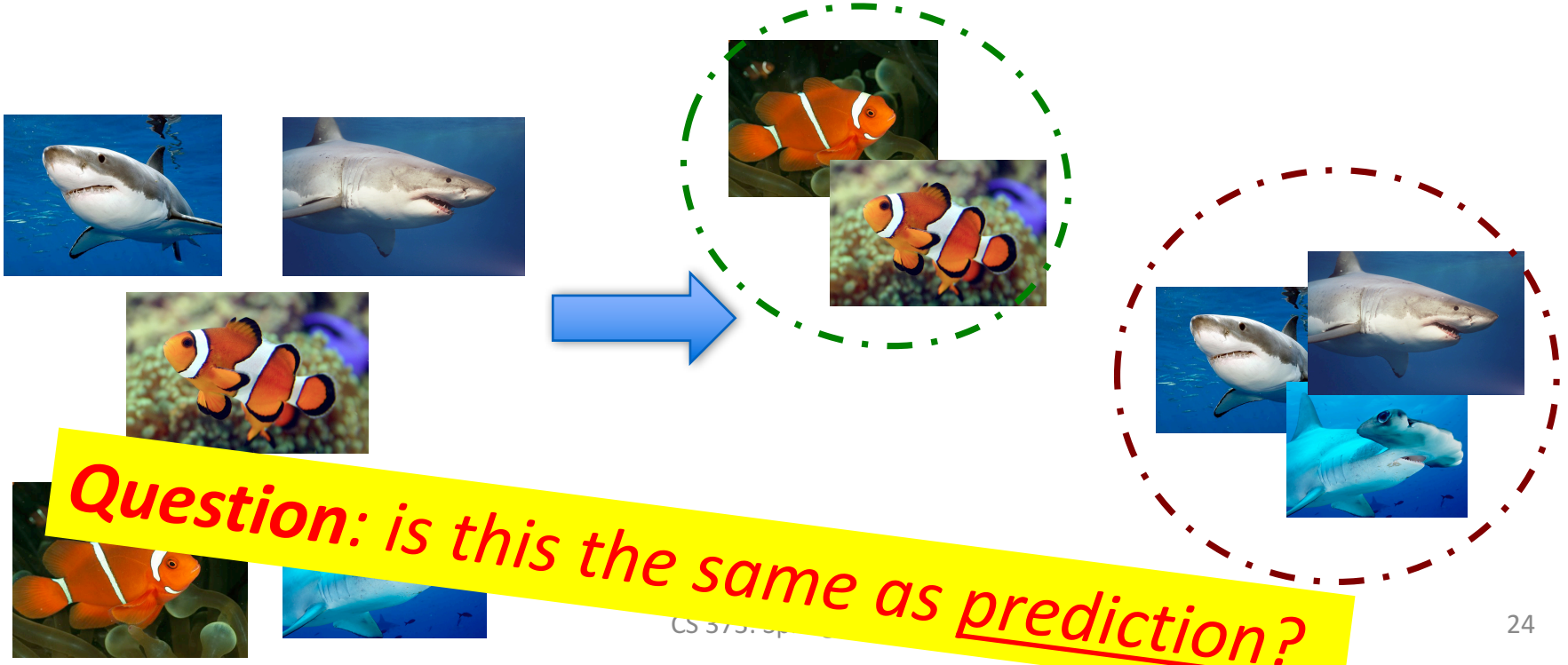
Supervised Learning: Classification

- A labeled dataset is a collection of (x,y) pairs
  - x refers to input examples
  - y refers to output labels
- Our goal is to build a model to predict new examples
- **Generalization**: can we make reliable predictions?

Class	Outlook	Temperature	Windy?
???	Sunny	Low	No

Supervised (Predictive) vs. Unsupervised (Descriptive)

- Unsupervised Learning: Learn properties of the data
- **Clustering:** group similar instance
- Define a similarity metric between instances



***Question**: is this the same as prediction?*

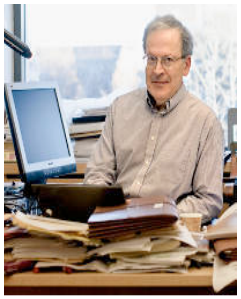
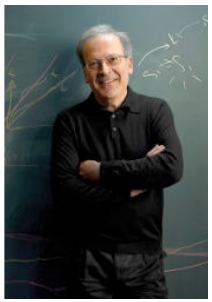
Clustering



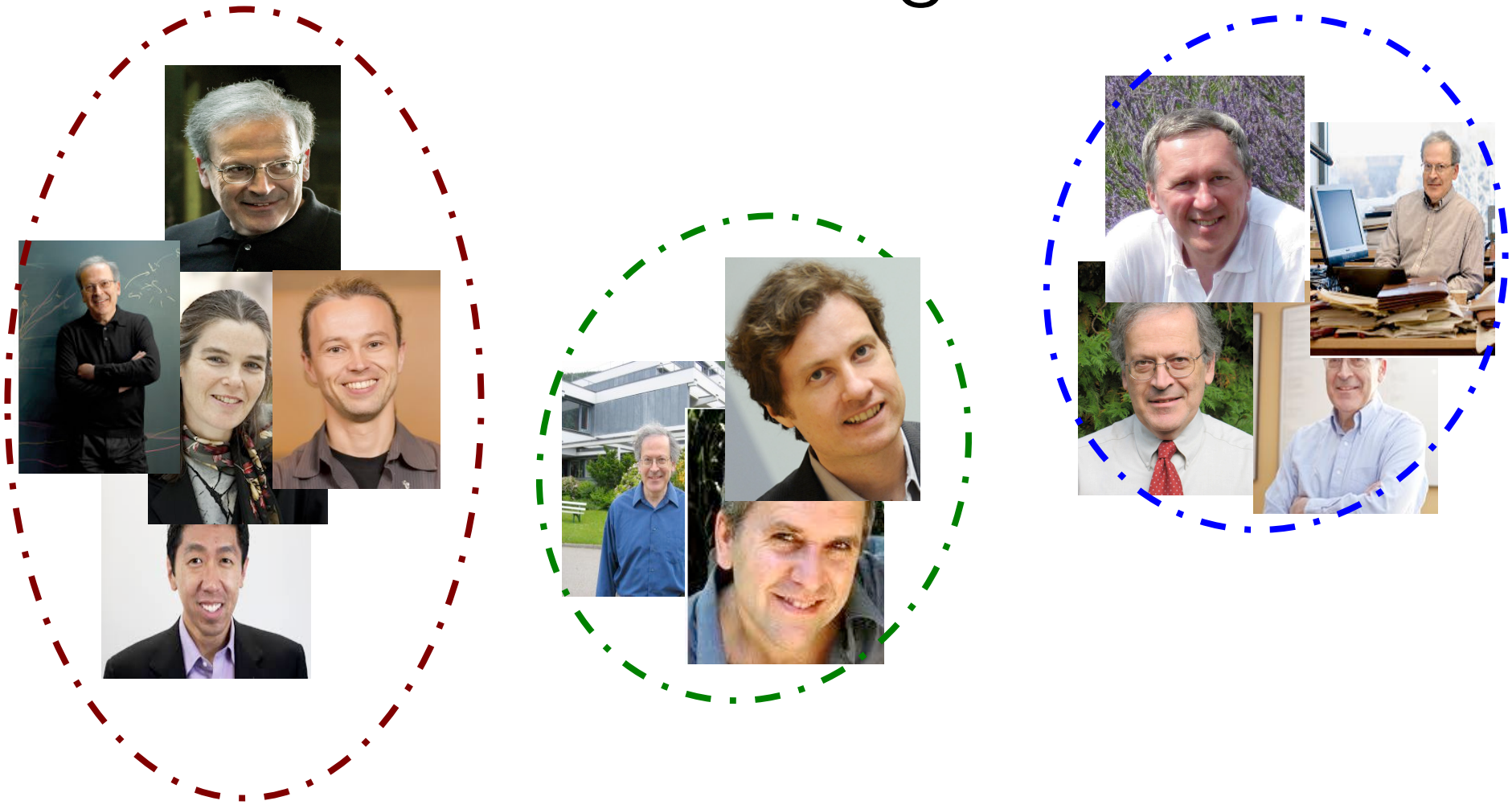






Clustering



Learning Algorithm

- Learning Algorithms generate a **model**, they work under the settings of a specific **protocol**
  - **Supervised vs. Semi-Supervised vs. Unsupervised**
  - **Online vs. Batch**
  - Online algorithm: learning is done one example at a time
    - Winnow, Perceptron,..
  - Batch algorithm: learning is done over entire dataset
    - SVM, Logistic Regression, Decision Trees, …

- How can we compare learning algorithms?

Learning Algorithm

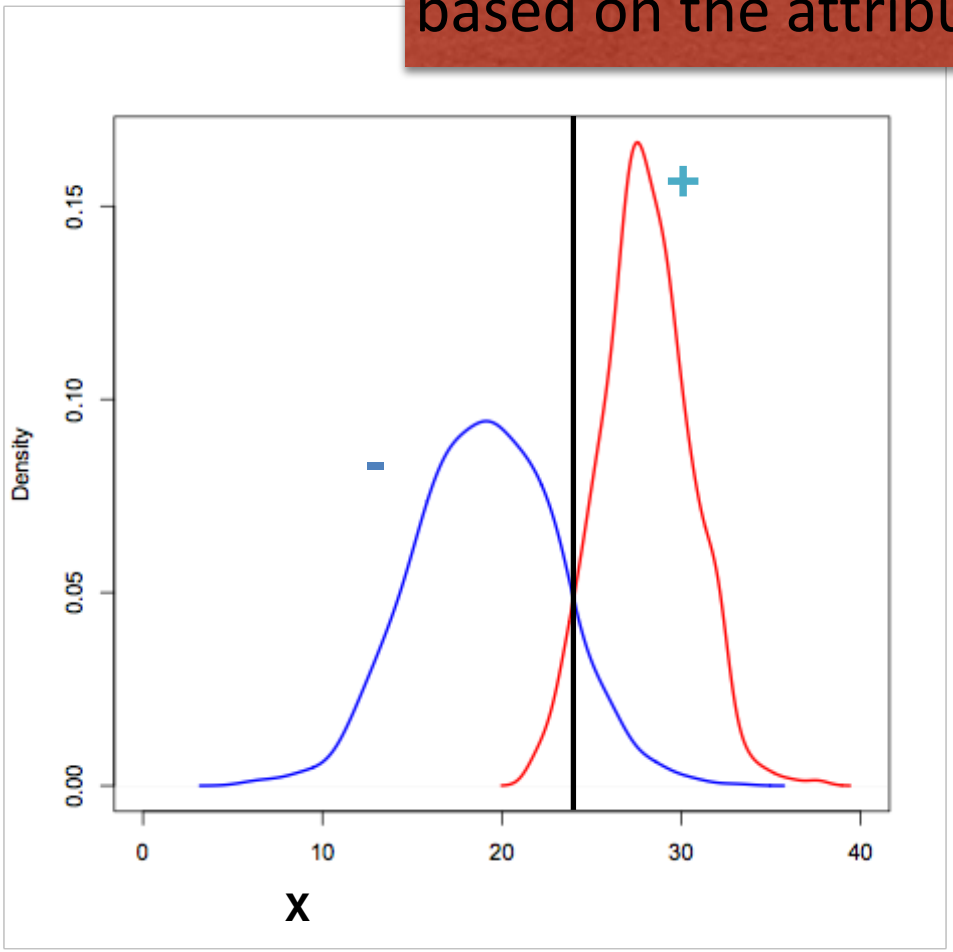
- Learning Algorithms generate a **model**, they work under the settings of a specific **protocol**
- **Learning is essentially search —**
  - Given the space of possible models in our hypothesis space
  - Search for the best model
    - Define a procedure for efficiently search the model space.
    - *Best* is defined as maximizing some scoring function, defined w.r.t the training data and other properties

Example learning problem

Task: Devise a rule to classify items based on the attribute **X**

Knowledge representation:
If-then rules

Example rule:
If x > 25 then +
Else -



What is the model space?

All possible thresholds

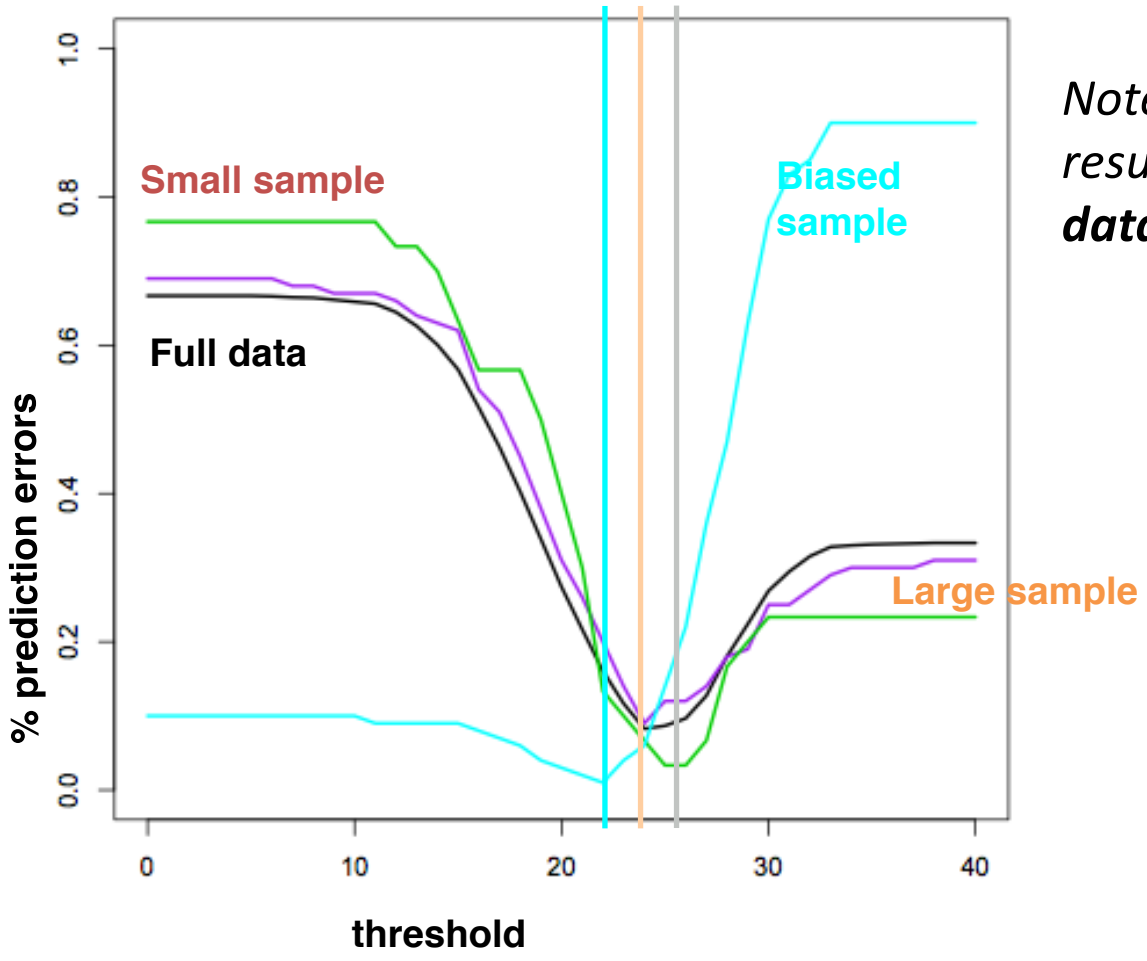
What score function?

Prediction error rate

Score function over model space

Search procedure?

Try all thresholds, select one with lowest score



*Note: learning result depends on **data***

Example: Identifying email spam

- **Task**
  - Spam detector that can differentiate between labeled emails
- **Data**
  - Table of relative word/ punctuation frequencies

TABLE 1.1. *Average percentage of words or characters in an email message equal to the indicated word or character. We have chosen the words and characters showing the largest difference between* spam *and* email.

	george	you	your	hp	free	hpl	!	our	re	edu	remove
spam	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13	0.01	0.28
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29	0.01

- **Model Space**
  - `If/then` rules with conjunctions of features

if (%george < 0.6) & (%you > 1.5) then spam
else email.

- **Learning technique**
  - **Search** over set of rules, **select** rule with maximum accuracy on training data

Your First Classifier!

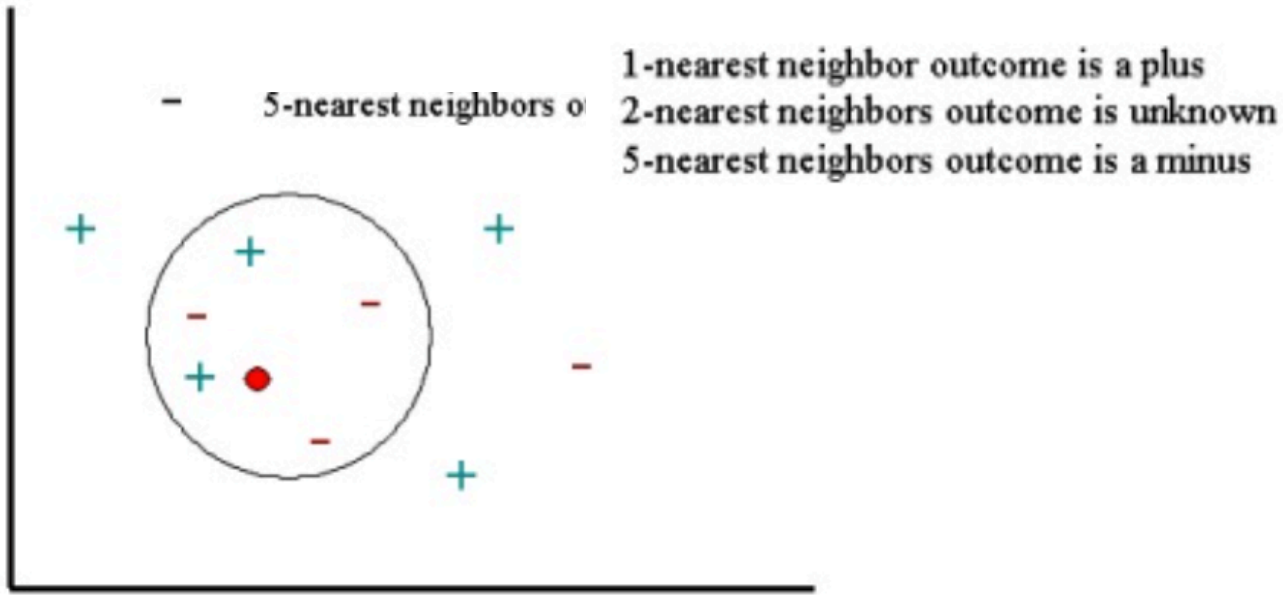
- Let's consider one of the simplest classifiers out there.
- Assume we have a training set $(x_1,y_1)…(x_n,y_n)$
- Now we get a new instance x_{new} , how can we classify it?
  - Example: Can you recommend a movie, based on user's movie reviews?
- **Simple Solution**:
  - Find the most similar example (x,y) in the training data and predict the same
    - If you liked "*Fast and Furious*" you'll like "*2 fast 2 furious*"
- Only a single decision is needed: distance metric to compute similarity

$$d(x_1, x_2) = 1 - \frac{x_1 \bigcap x_2}{x_1 \bigcup x_2}$$

$$d(x_1, x_2) = \sqrt[2]{(x_1 - x_2)^2}$$

K Nearest Neighbors

- Can you thing about a better way?
- We can make the decision by looking at several near examples, not just one. **Why would it be better**?



K Nearest Neighbors

- **Learning**: just storing the training examples
- **Prediction**:
  - Find the K training example closest to **x**
- **Predict a label:**
  - Classification: majority vote
  - Regression: mean value
- KNN is a type of *instance based learning*
- This is called *lazy* learning, since most of the computation is done at prediction time

KNN analysis

- ***What are the advantages and disadvantages of KNN?***
  - *What should we care about when answering this question?*
- ***Complexity***
  - ***Space*** *(how memory efficient is the algorithm?)*
    - *Why should we care?*
  - ***Time*** *(computational complexity)*
    - *Both at training time and at test (prediction) time*
- ***Expressivity***
  - *What kind of functions can we learn?*

KNN analysis

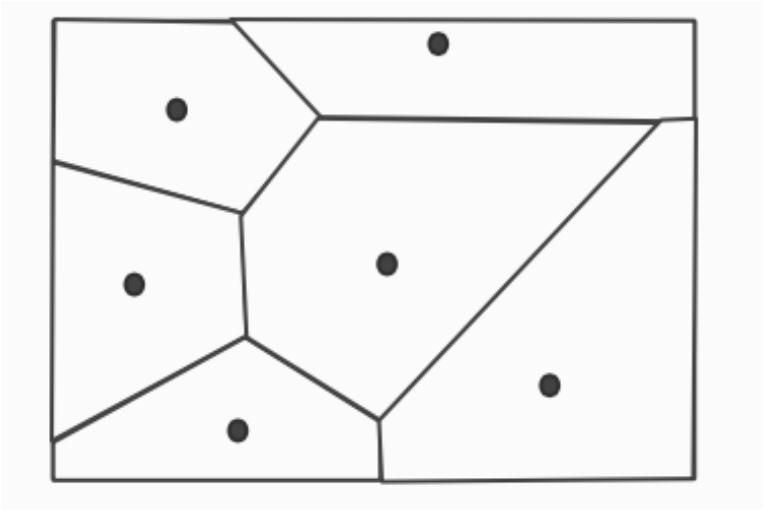
- ***What are the advantages and disadvantages of KNN?***
  - *What should we care about when answering this question?*
- ***Complexity***
  - ***Space** (how memory efficient is the algorithm?)*
    - *Why should we care?*
  - ***Time** (computational complexity)*
    - *Both at training time and at test (prediction) time*
- ***Expressivity***
  - *What kind of functions can we learn?*

KNN needs to maintain all training examples!
-Datasets can be HUGE

Training is very fast! But *prediction is slow*
- O(dN) for N examples with d attributes
- *increases* with the number of examples!

KNN analysis

- We discussed the importance of the model space
  - Expressive (we can represent the right model)
  - Constrained (we can search effectively, using available data)
- Let's try to characterize the model space, by looking at the **decision boundary**
- **How would it look if K=1?**

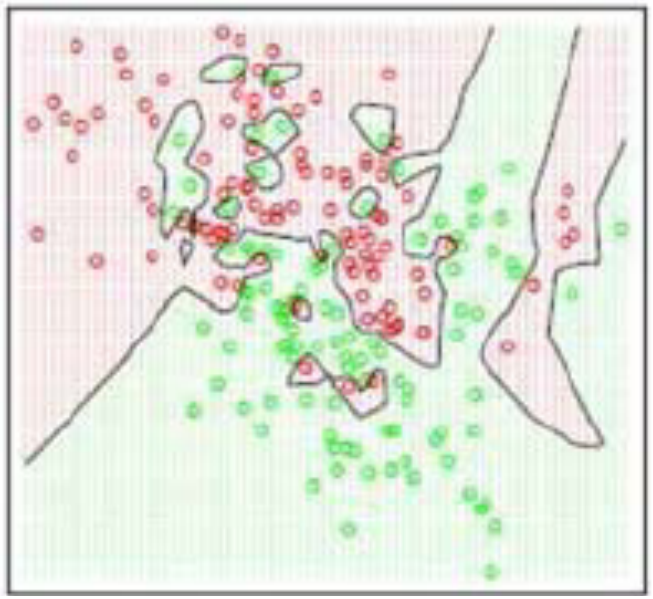


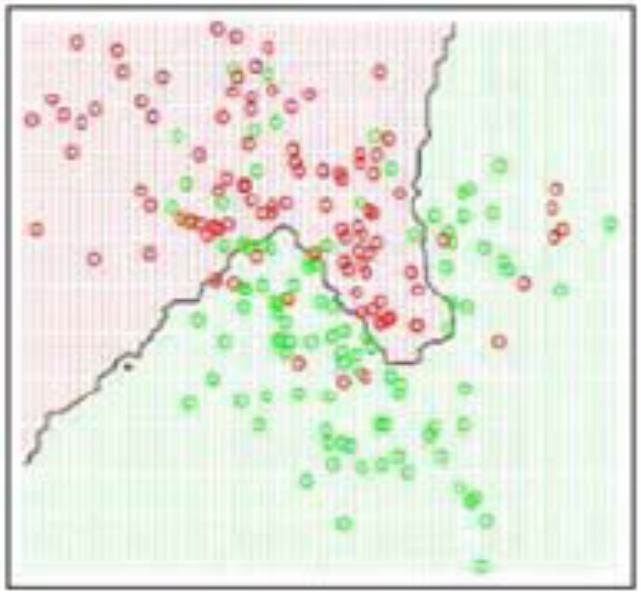
We define the model space to be our choice of K.

Does the complexity of the model space increase of decrease with K?

KNN analysis

- Which model has a higher K value?
- Which model is more complex?
- Which model is more sensitive to noise?





Questions

- We know higher K values result in a smoother decision boundary.
  - Less ”jagged” decision regions
  - Total number of regions will be smaller

What will happen if we keep increasing K, up to the point that K=n ?

n = is the number of examples we have

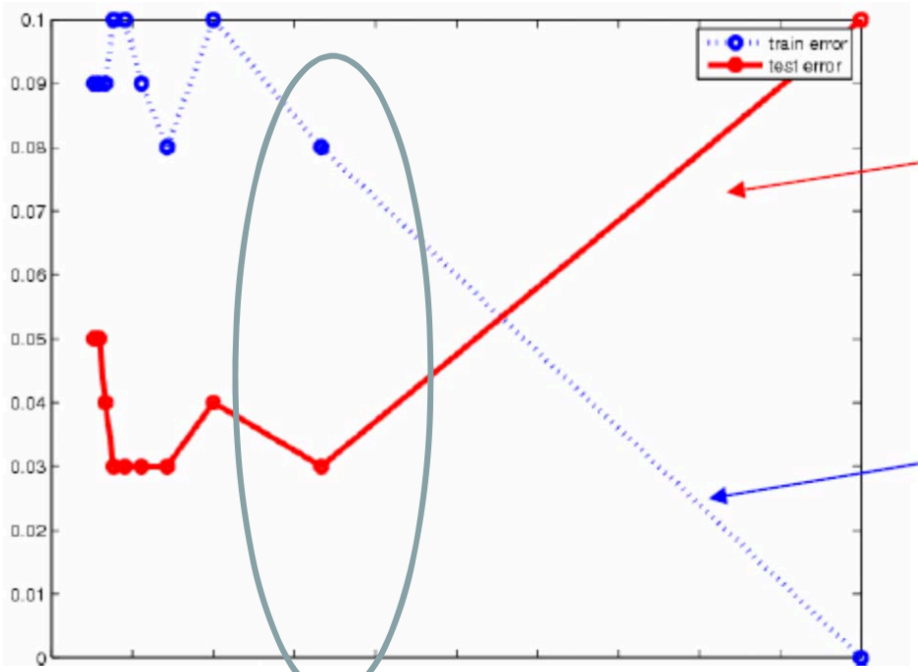
Determining the value of K

- Higher K result in less complex functions (less expressive)
- Lower K values are more complex (more expressive)
  - **How can we find the right balance between the two?**
- **Option 1**: *Find the K that minimizes the training error*.
  - Training error: after learning the classifier, what is the number of errors we get on the training data.
  - What will be this value for k=1, k=n, k=n/2?

Is this a good idea?

- **Option 2**: *Find K that minimizes the **validation error***.
  - Validation error: set aside some of the data (validation set). what is the number of errors we get on the validation data, after training the classifier.

Determining the value of K



Test error

Train error

k=20

k=1

Possibly a good choice for k

In general – using the training error to tune parameters will always result in a more complex hypothesis! **(why?)**

Practical Considerations

- Finding the right representation is key
  - KNN is very sensitive to irrelevant attributes
- Choosing the right distance metric is important
  - Many options!
  - Popular choices:
    - Euclidean distance

$$||\mathbf{x}_1 - \mathbf{x}_2||_2 = \sqrt{\sum_{i=1}^{n} (\mathbf{x}_{1,i} - \mathbf{x}_{2,i})^2}$$

    - Manhattan distance

$$||\mathbf{x}_1 - \mathbf{x}_2||_1 = \sum_{i=1}^{n} |\mathbf{x}_{1,i} - \mathbf{x}_{2,i}|$$

    - L_p-norm
      - Euclidean = L_2
      - Manhattan = L_1

$$||\mathbf{x}_1 - \mathbf{x}_2||_p = \left(\sum |\mathbf{x}_{1,i} - \mathbf{x}_{2,i}|^p \right)^{\frac{1}{p}}$$