CS446 Introduction to Machine Learning (Spring 2015)
University of Illinois at Urbana-Champaign
http://courses.engr.illinois.edu/cs446

LECTURE 2: SUPERVISED LEARNING

Prof. Julia Hockenmaier
juliahmr@illinois.edu

Class admin

Are you on Piazza?

Is everybody registered for the class?

HW0 is out (not graded)
http://courses.engr.illinois.edu/cs446/Homework/HW0/HW0.pdf

Email alias for CS446 staff:
cs446-staff@mx.uillinois.edu

Learning scenarios

Supervised learning:

The focus of CS446

Learning to predict labels from correctly labeled data

Unsupervised learning:

Learning to find hidden structure (e.g. clusters) in input data

Semi-supervised learning:

Learning to predict labels from (a little) labeled
and (a lot of) unlabeled data

Reinforcement learning:

Learning to act through feedback for actions
(rewards/punishments) from the environment

The Badges game

+ Naoki Abe

- Eric Baum

Conference attendees to the 1994 Machine Learning conference were given name badges labeled with + or –.

What function was used to assign these labels?

The supervised learning task

Given a labeled training data set of N items $\mathbf{x}_n \in \mathcal{X}$ with labels $y_n \in \mathcal{Y}$

$$\mathcal{D}^{\text{train}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$$

(y_n is determined by some unknown target function $f(\mathbf{x})$)

Return a model $g: \mathcal{X} \longmapsto \mathcal{Y}$ that is a good approximation of $f(\mathbf{x})$

(g should assign correct labels y to unseen $\mathbf{x} \notin \mathcal{D}^{\text{train}}$)

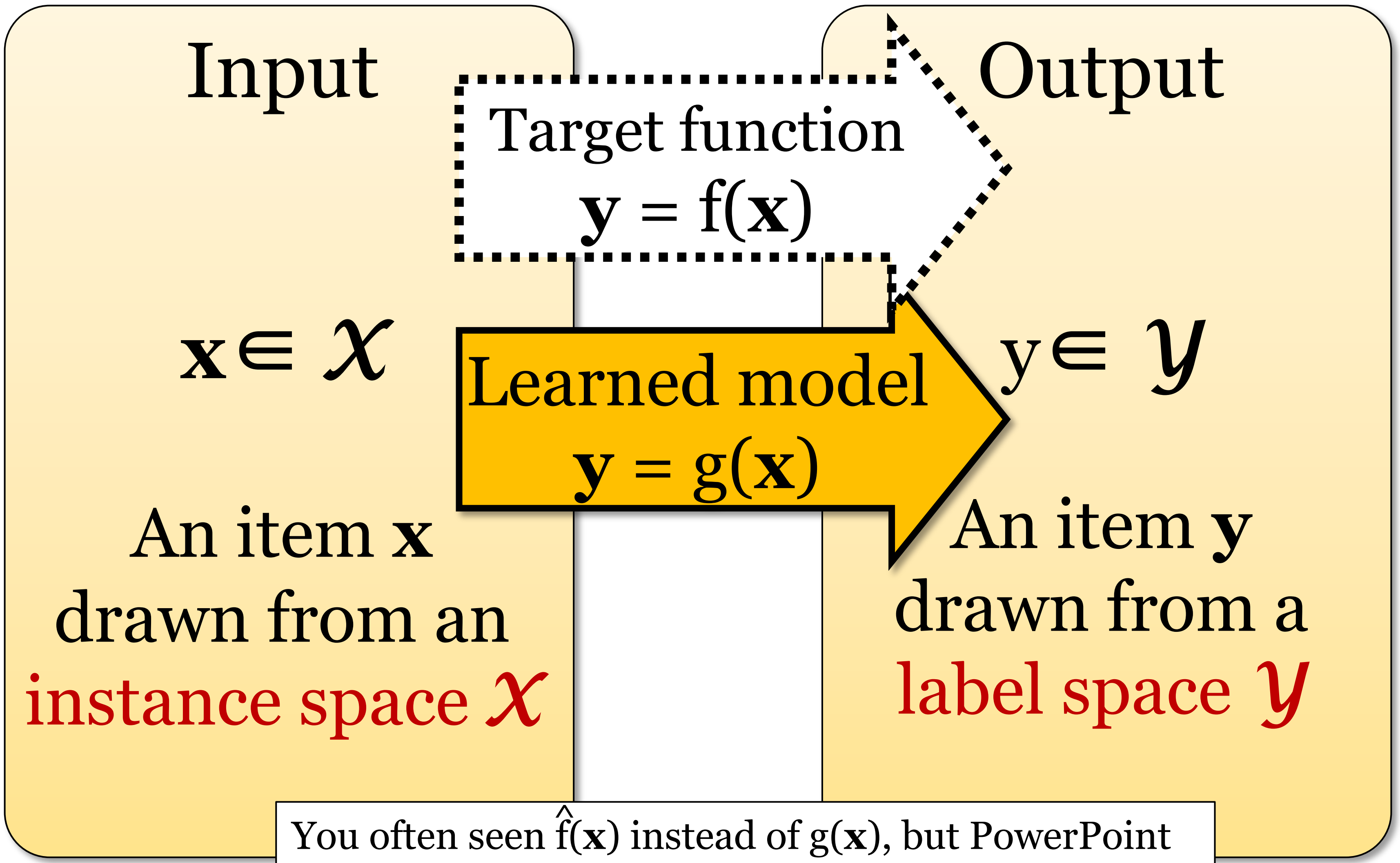
Supervised learning terms

Input items/data points $\mathbf{x}_n \in \mathcal{X}$ (e.g. emails) are drawn from an instance space $\mathcal{X}$

Output labels $y_n \in \mathcal{Y}$ (e.g. 'spam'/'nospam') are drawn from a label space $\mathcal{Y}$

Every data point $\mathbf{x}_n \in \mathcal{X}$ has a *single* correct label $y_n \in \mathcal{Y}$, defined by an (unknown) target function $f(\mathbf{x}) = y$

Supervised learning



Input

$\mathbf{x} \in \mathcal{X}$

An item **x** drawn from an instance space $\mathcal{X}$

Target function
$\mathbf{y} = f(\mathbf{x})$

Learned model
$\mathbf{y} = g(\mathbf{x})$

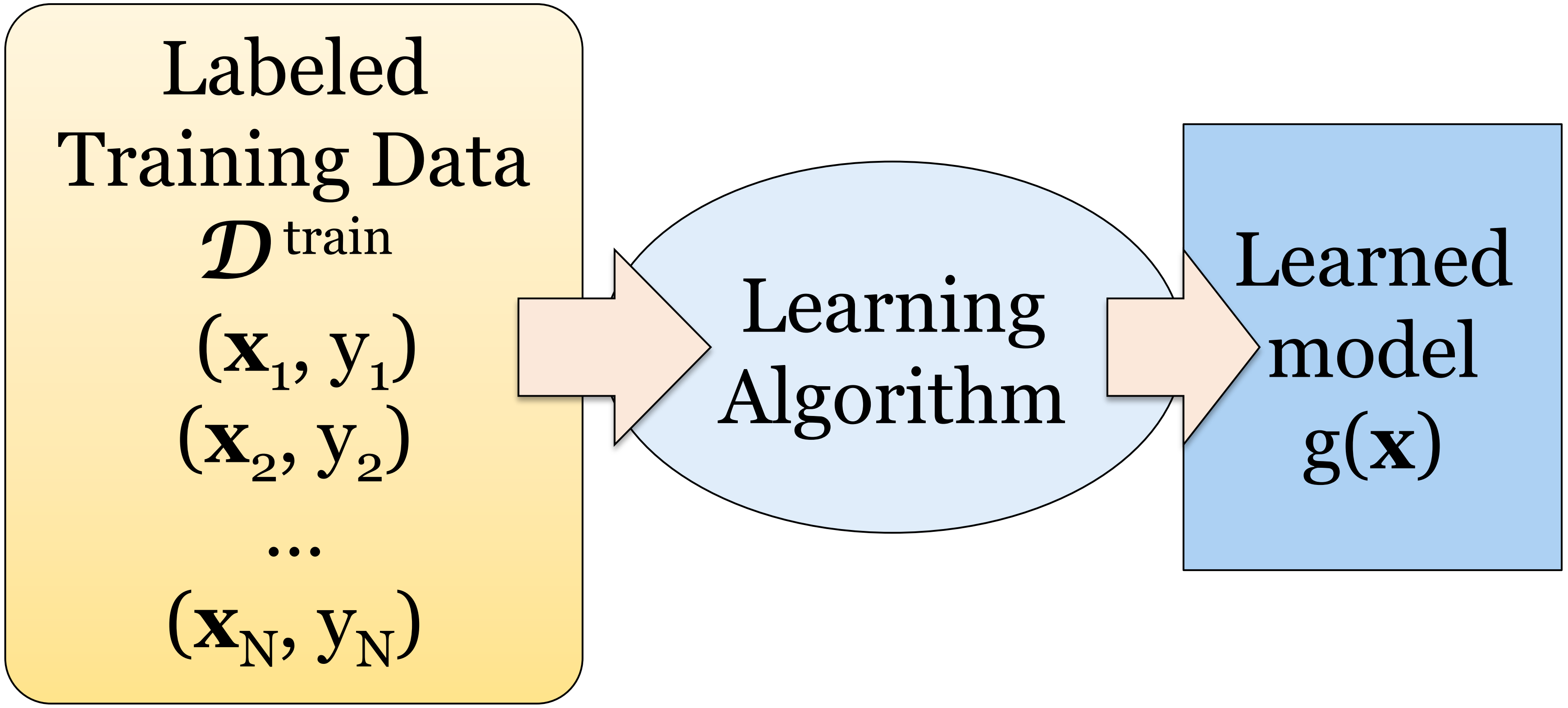
Output

$y \in \mathcal{Y}$

An item **y** drawn from a label space $\mathcal{Y}$

You often seen $\hat{f}(\mathbf{x})$ instead of $g(\mathbf{x})$, but PowerPoint can't really typeset that, so $g(\mathbf{x})$ will have to do.

Supervised learning: Training



Give the learner examples in $\mathcal{D}^{\text{train}}$

The learner returns a model $g(\mathbf{x})$

Training data

+ Naoki Abe
- Myriam Abramson
+ David W. Aha
+ Kamal M. Ali
- Eric Allender
+ Dana Angluin
- Chidanand Apte
+ Minoru Asada
+ Lars Asker
+ Javed Aslam
+ Jose L. Balcazar
- Cristina Baroglio

+ Peter Bartlett
- Eric Baum
+ Welton Becket
- Shai Ben-David
+ George Berg
+ Neil Berkman
+ Malini Bhandaru
+ Bir Bhanu
+ Reinhard Blasig
- Avrim Blum
- Anselm Blumer
+ Justin Boyan

+ Carla E. Brodley
+ Nader Bshouty
- Wray Buntine
- Andrey Burago
+ Tom Bylander
+ Bill Byrne
- Claire Cardie
+ John Case
+ Jason Catlett
- Philip Chan
- Zhixiang Chen
- Chris Darken

Supervised learning: Testing

Labeled

Test Data

$\mathcal{D}^{\text{test}}$

$(\mathbf{x}'_1, y'_1)$

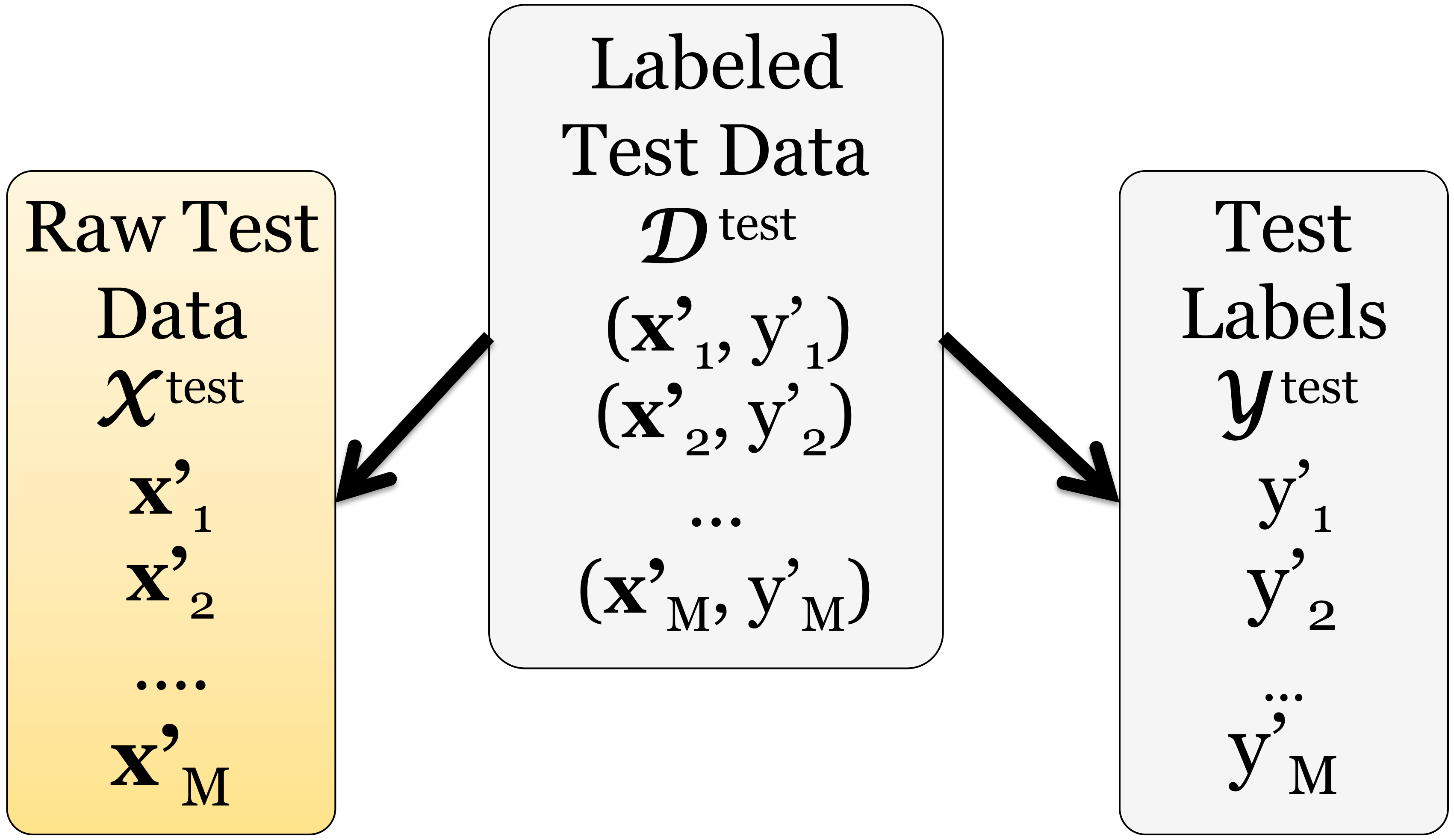
$(\mathbf{x}'_2, y'_2)$

…

$(\mathbf{x}'_M, y'_M)$

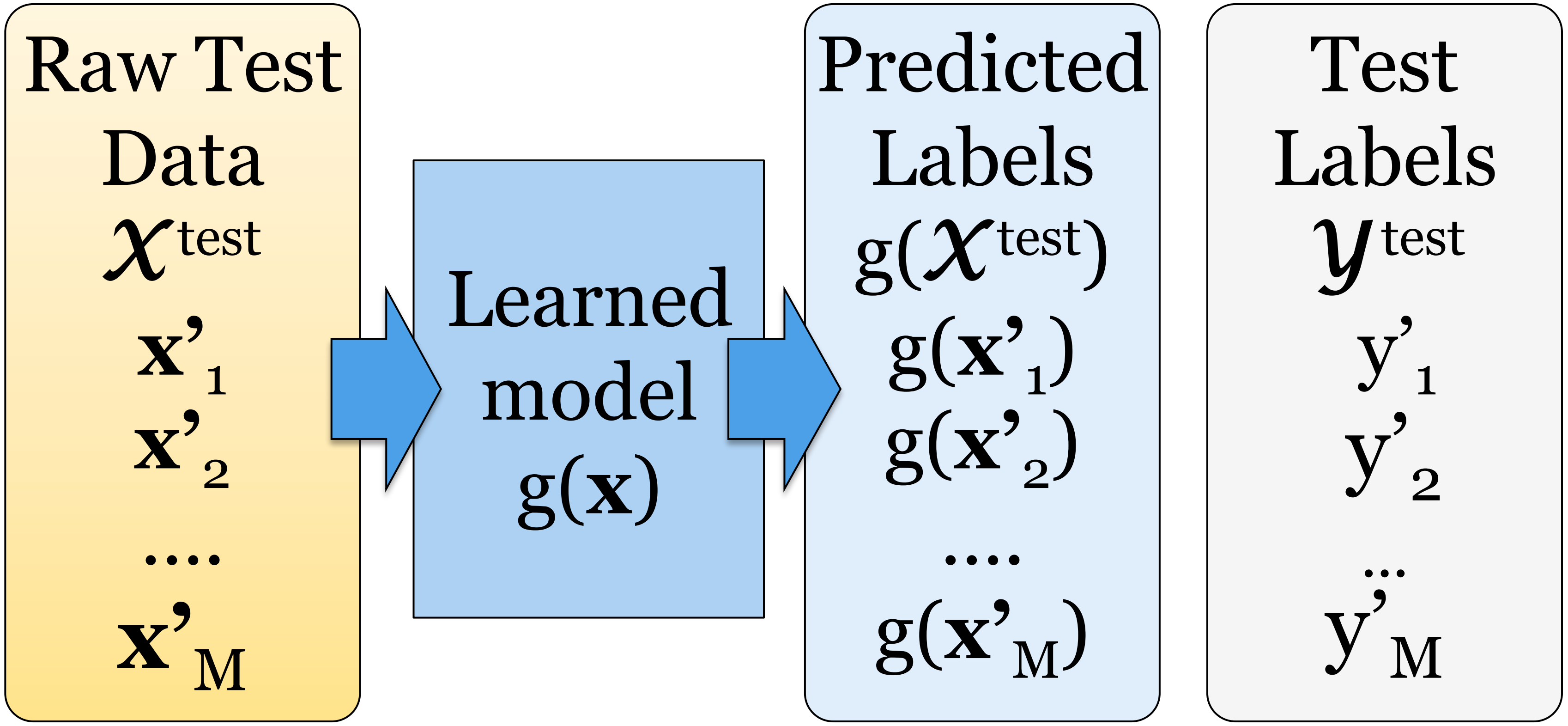
Reserve some labeled data for testing

Supervised learning: Testing



Supervised learning: Testing

Apply the model to the raw test data



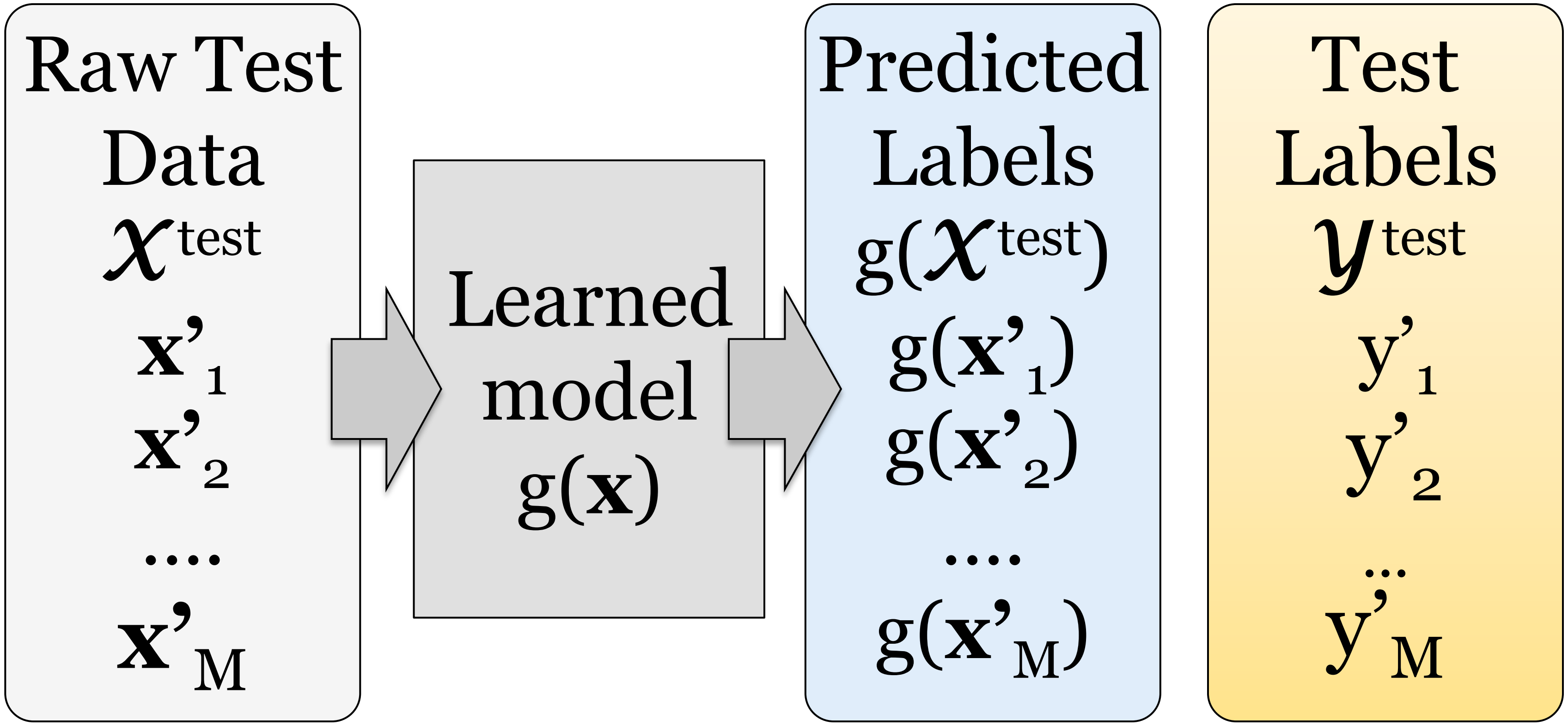
Raw test data

Gerald F. DeJong
Chris Drummond
Yolanda Gil
Attilio Giordana
Jiarong Hong

J. R. Quinlan
Priscilla Rasmussen
Dan Roth
Yoram Singer
Lyle H. Ungar

Supervised learning: Testing

Evaluate the model by comparing the predicted labels against the test labels



Labeled test data

+ Gerald F. DeJong
- Chris Drummond
+ Yolanda Gil
- Attilio Giordana
+ Jiarong Hong
- J. R. Quinlan
- Priscilla Rasmussen
+ Dan Roth
+ Yoram Singer
- Lyle H. Ungar

Evaluating supervised learners

Use a test data set that is disjoint from $\mathcal{D}^{\text{train}}$

$\mathcal{D}^{\text{test}} = \{(\mathbf{x}'_1, y'_1), \ldots, (\mathbf{x}'_M, y'_M)\}$

- The learner has not seen the test items during learning.
- Split your labeled data into two parts: test and training.

Take all items $\mathbf{x}'_i$ in $\mathcal{D}^{\text{test}}$ and compare the predicted $f(\mathbf{x}'_i)$ with the correct y'_i .

- This requires an evaluation metric (e.g. accuracy).

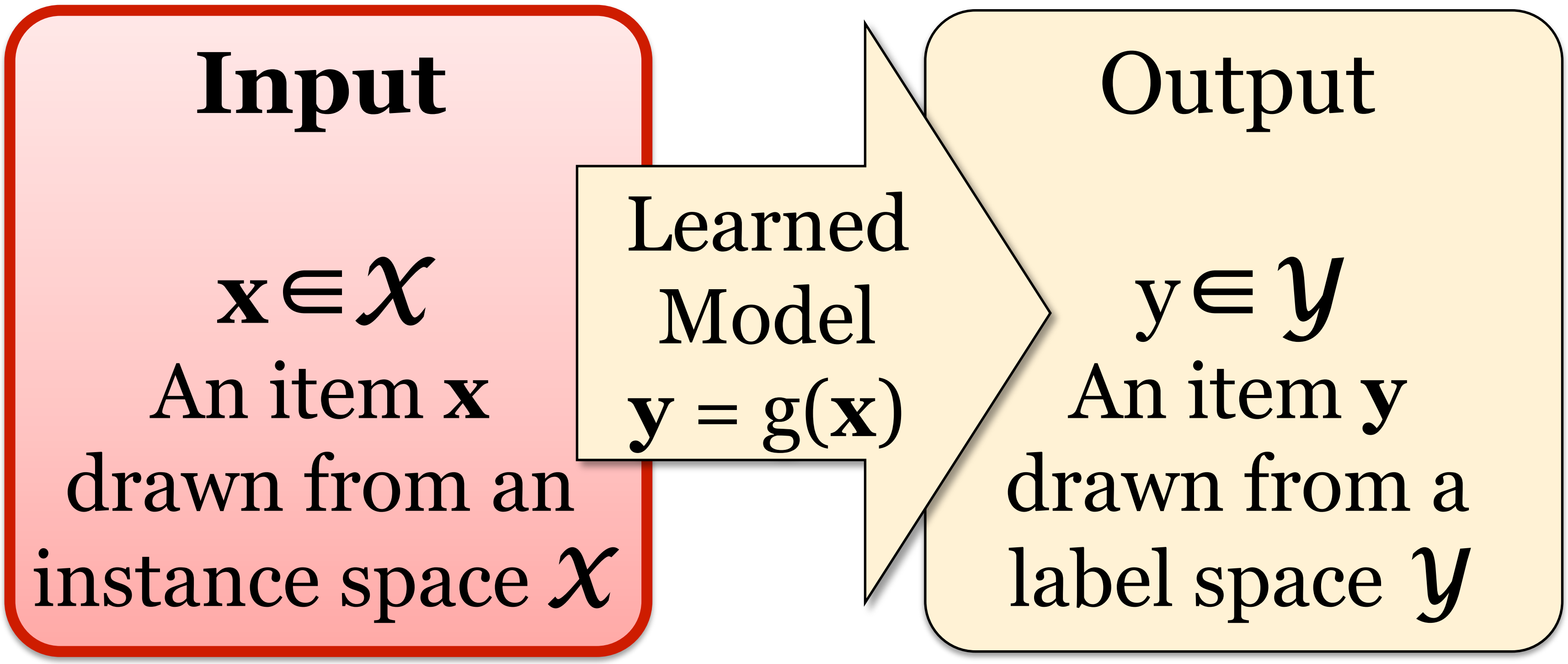
Using supervised learning

- What is our instance space?
  Gloss: What kind of features are we using?
- What is our label space?
  Gloss: What kind of learning task are we dealing with?
- What is our hypothesis space?
  Gloss: What kind of model are we learning?
- What learning algorithm do we use?
  Gloss: How do we learn the model from the labeled data?

(What is our loss function/evaluation metric?)
  Gloss: How do we measure success?

1. The instance space

1. The instance space $\mathcal{X}$


Input
$\mathbf{x} \in \mathcal{X}$
An item **x** drawn from an instance space $\mathcal{X}$
Learned Model
$\mathbf{y} = g(\mathbf{x})$
Output
$y \in \mathcal{Y}$
An item **y** drawn from a label space $\mathcal{Y}$

Designing an appropriate instance space $\mathcal{X}$ is crucial for how well we can predict y.

1. The instance space $\mathcal{X}$

When we apply machine learning to a task, we first need to *define* the instance space $\mathcal{X}$.

Instances $\mathbf{x} \in \mathcal{X}$ are defined by features:

– Boolean features:

Does this email contain the word 'money'?

– Numerical features:

How often does 'money' occur in this email?
What is the width/height of this bounding box?

What's $\mathcal{X}$ for the Badges game?

Possible features:

- Gender/age/country of the person?
- Length of their first or last name?
- Does the name contain letter 'x'?
- How many vowels does their name contain?
- Is the n-th letter a vowel?

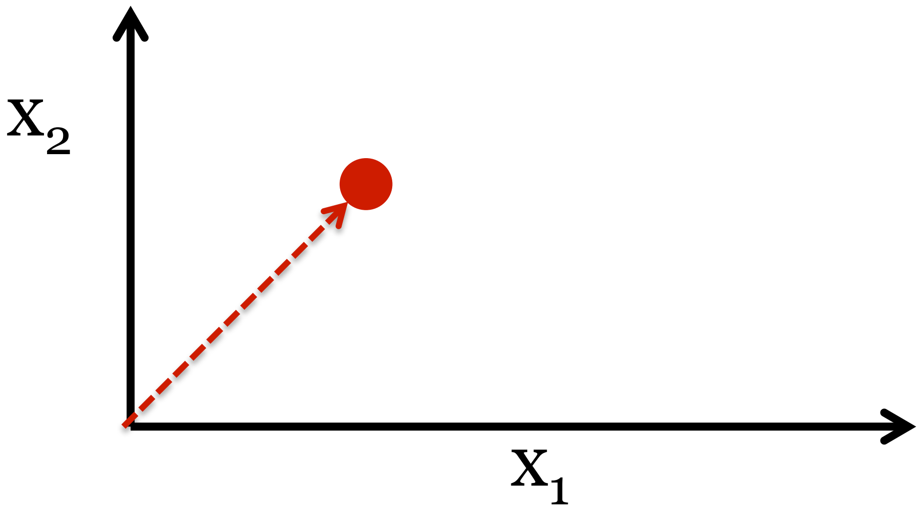
$\mathcal{X}$ as a vector space

$\mathcal{X}$ is an N-dimensional vector space (e.g. $\mathbb{R}^N$)

Each dimension = one feature.

Each **x** is a feature vector (hence the boldface **x**).

Think of **x** $= [x_1 \ldots x_N]$ as a point in $\mathcal{X}$:



From feature templates to vectors

When designing features, we often think in terms of templates, not individual features:

What is the 2nd letter?

N **a** oki → [1 0 0 0 ...]

A **b** e → [0 1 0 0 ...]

S **c** rooge → [0 0 1 0 ...]

What is the *i*-th letter?

Abe → [**1** 0 0 0 0... 0 **1** 0 0 0 0... 0 0 0 0 **1** ...]

Good features are essential

The choice of features is crucial
for how well a task can be learned.

In many application areas (language, vision, etc.),
a lot of work goes into designing suitable features.

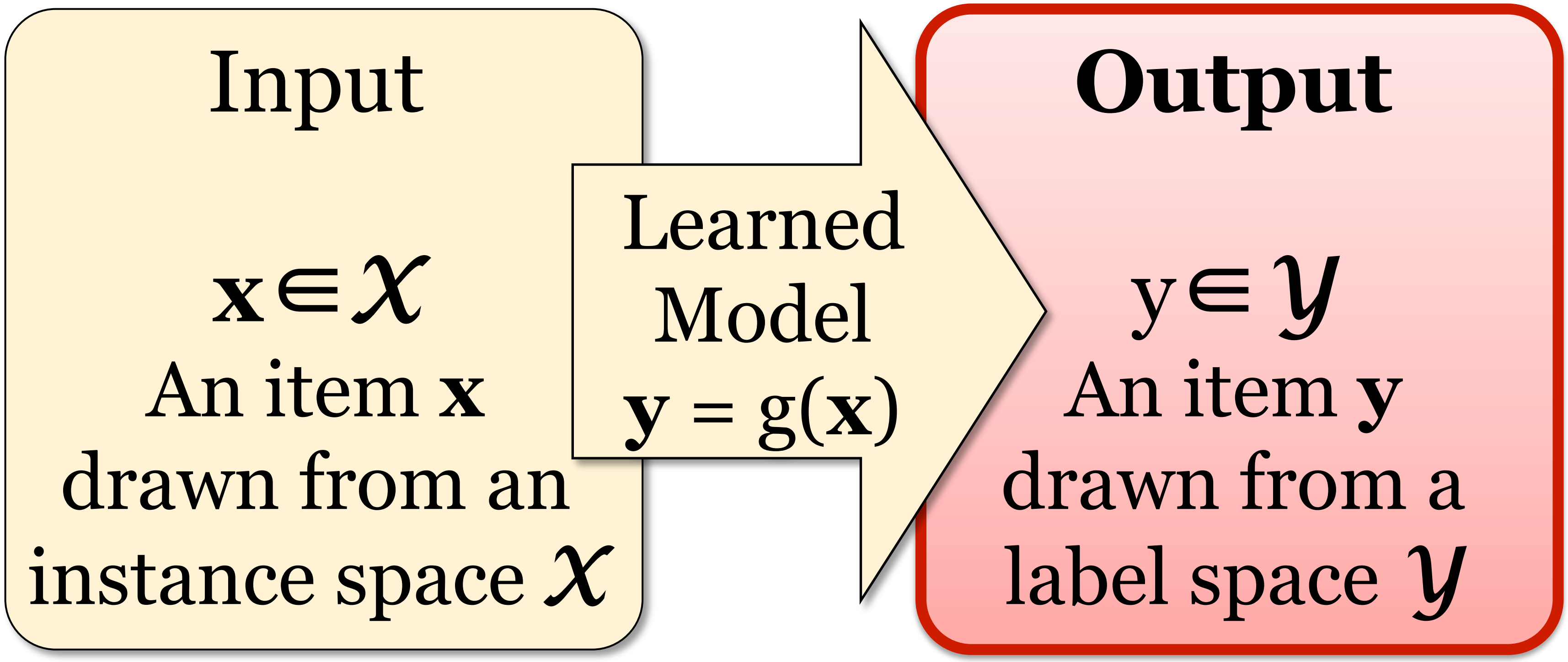
This requires domain expertise.

CS446 can't teach you what specific features
to use for your task.

But we will touch on some general principles

2. The label space

2. The label space $\mathcal{Y}$


Input
$\mathbf{x} \in \mathcal{X}$
An item $\mathbf{x}$ drawn from an instance space $\mathcal{X}$
Learned Model
$\mathbf{y} = g(\mathbf{x})$
Output
$y \in \mathcal{Y}$
An item $\mathbf{y}$ drawn from a label space $\mathcal{Y}$

The label space $\mathcal{Y}$ determines what *kind* of supervised learning task we are dealing with

Supervised learning tasks I

Output labels $y \in \mathcal{Y}$ are **categorical**: The focus of CS446

– Binary classification: Two possible labels

– Multiclass classification: k possible labels

Output labels $y \in \mathcal{Y}$ are **structured objects** (sequences of labels, parse trees, etc.)

– Structure learning (e.g. CS546)

Supervised learning tasks II

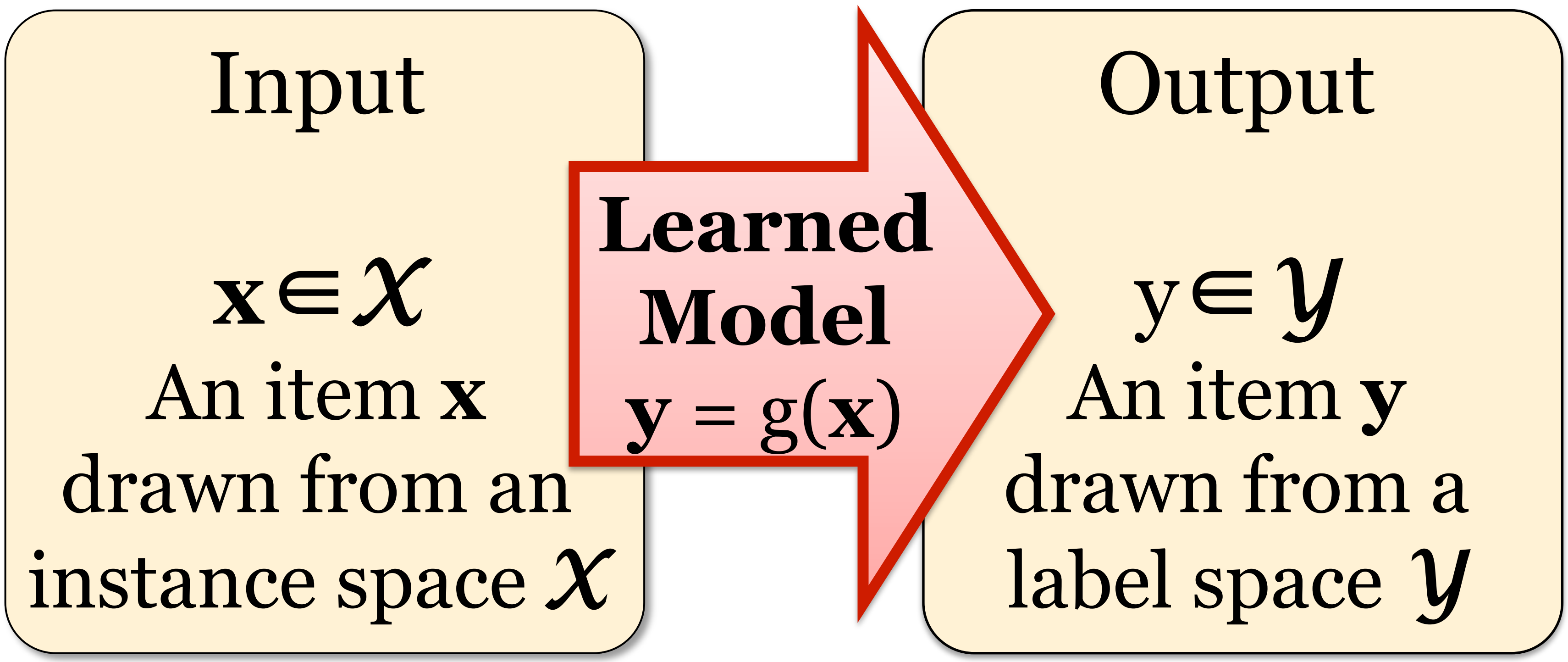
Output labels $y \in \mathcal{Y}$ are **numerical**:

- Regression (linear/polynomial):
  Labels are continuous-valued
  Learn a linear/polynomial function f(x)

- Ranking:
  Labels are ordinal
  Learn an ordering $f(x_1) > f(x_2)$ over input

3. Models
(The hypothesis space)

3. The model g(**x**)


Input
$\mathbf{x} \in \mathcal{X}$
An item **x** drawn from an instance space $\mathcal{X}$
Learned Model
$\mathbf{y} = g(\mathbf{x})$
Output
$y \in \mathcal{Y}$
An item **y** drawn from a label space $\mathcal{Y}$

We need to choose what *kind* of model we want to learn

More terminology

For classification tasks ($\mathcal{Y}$ is categorical, e.g. {0, 1}, or {0, 1, …, k}), the model is called a classifier.

For binary classification tasks ($\mathcal{Y}$ = {0, 1}), we often think of the two values of $\mathcal{Y}$ as Boolean (0 = false, 1 = true), and call the target function f(**x**) to be learned a concept

A learning problem

	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

A learning problem

Each $\mathbf{x}$ has 4 bits: $|\mathcal{X}| = 2^4 = 16$

Since $\mathcal{Y} = \{0, 1\}$, each $f(\mathbf{x})$
defines one subset of $\mathcal{X}$

$\mathcal{X}$ has $2^{16} = 65536$ subsets:
There are 2^{16} possible $f(\mathbf{x})$
(2^9 are consistent with our data)

We would need to see all of $\mathcal{X}$ to learn $f(\mathbf{x})$

A learning problem

We would need to see all of $\mathcal{X}$ to learn f($\mathbf{x}$)

– Easy with $|\mathcal{X}|=16$

– Not feasible in general
(for any real-world problems)

– Learning = generalization,
not memorization of the training data

The hypothesis space $\mathcal{H}$

There are $|\mathcal{Y}|^{|\mathcal{X}|}$ possible functions f(**x**) from the instance space $\mathcal{X}$ to the label space $\mathcal{Y}$.

Learners typically consider only a *subset* of the functions from $\mathcal{X}$ to $\mathcal{Y}$.

This subset is called the hypothesis space $\mathcal{H}$.

$$\mathcal{H} \subseteq |\mathcal{Y}|^{|\mathcal{X}|}$$

Can we restrict $\mathcal{H}$?

	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

Conjunctive clauses:

16 different conjunctions over $\{x_1, x_2, x_3, x_4\}$:

$f(\mathbf{x}) = x_1$

…

$f(\mathbf{x}) = x_1 \wedge x_2 \wedge x_3 \wedge x_4$

None is consistent with the data

Can we restrict $\mathcal{H}$?

	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

n-of-m clauses:

20 rules of the form "y=1 iff at least *m* of the following *n* x_i are 1"

Can we restrict $\mathcal{H}$?

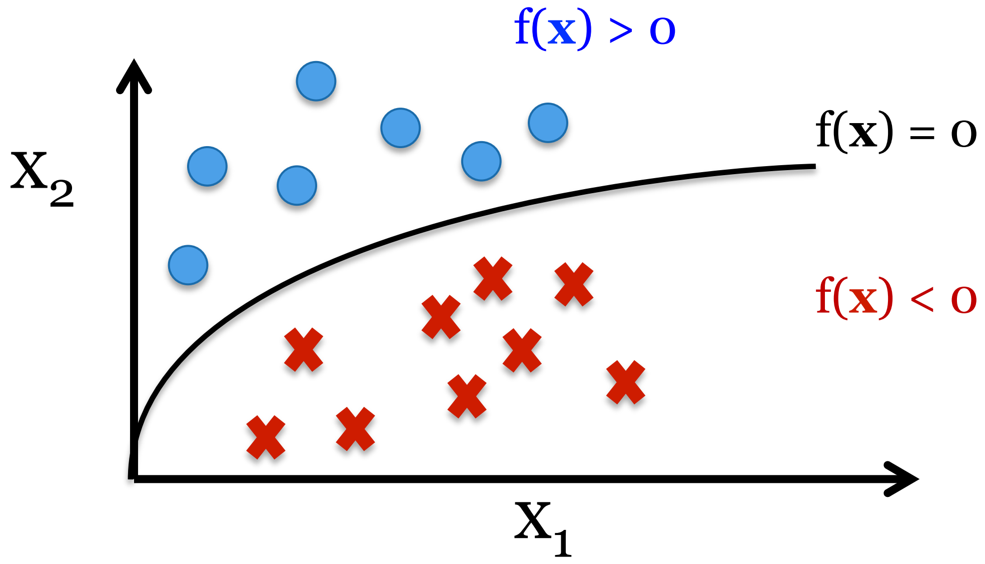
	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

n-of-m clauses:

20 rules of the form
"y=1 iff at least *m* of the following *n* x_i are 1"

Consistent hypothesis:
"y=1 if and only if
at least *2* of $\{x_1, x_3, x_4\}$ are 1"

Classifiers in vector spaces



Binary classification:

We assume f *separates* the positive and negative examples:

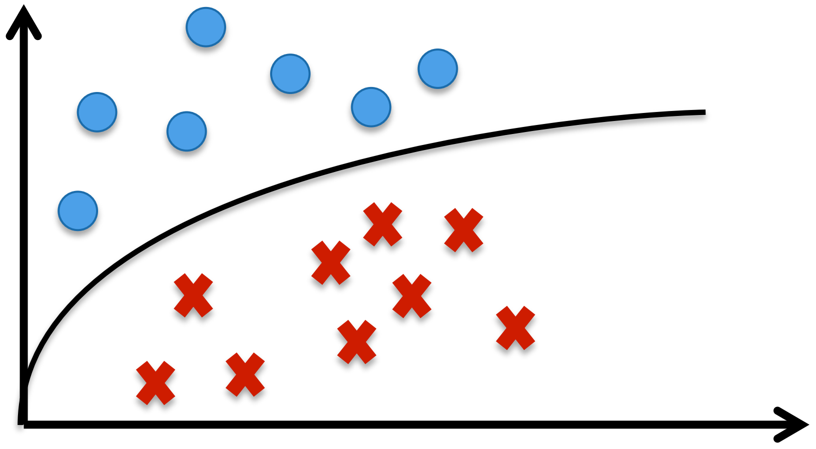
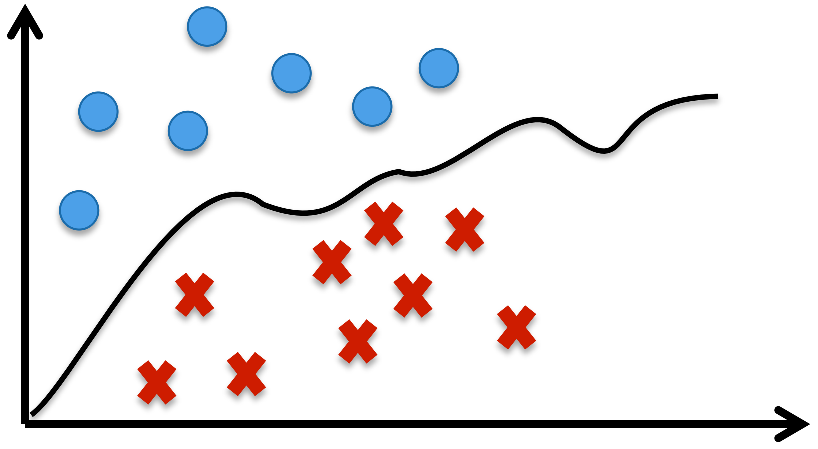
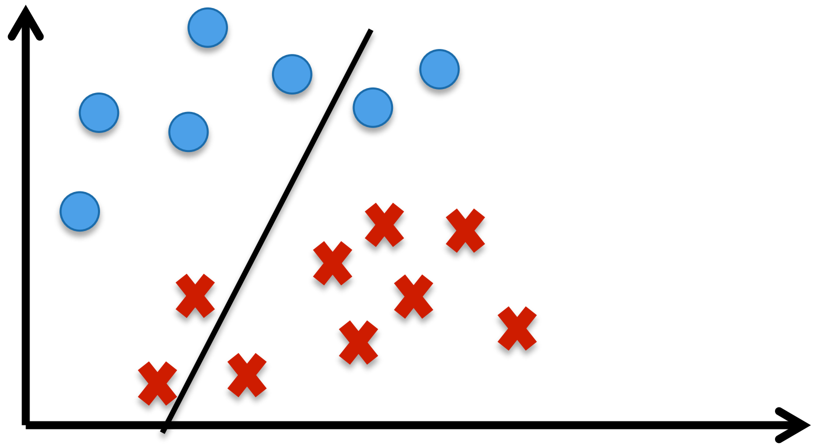
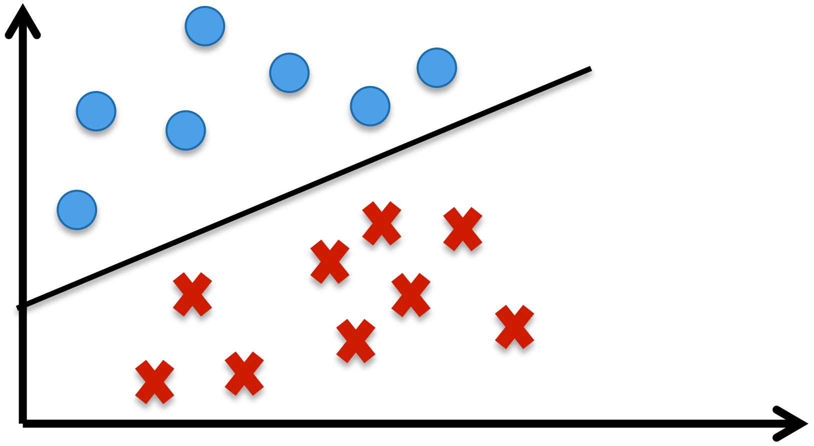
- Assign y = 1 to all **x** where f(**x**) > 0
- Assign y = 0 to all **x** where f(**x**) < 0

Learning a classifier

The learning task:
Find a function f($\mathbf{x}$) that best separates the (training) data

- What kind of function is f?
- How do we define *best*?
- How do we find f?

Which model should we pick?

Criteria for choosing models

Accuracy:

Prefer models that make fewer mistakes

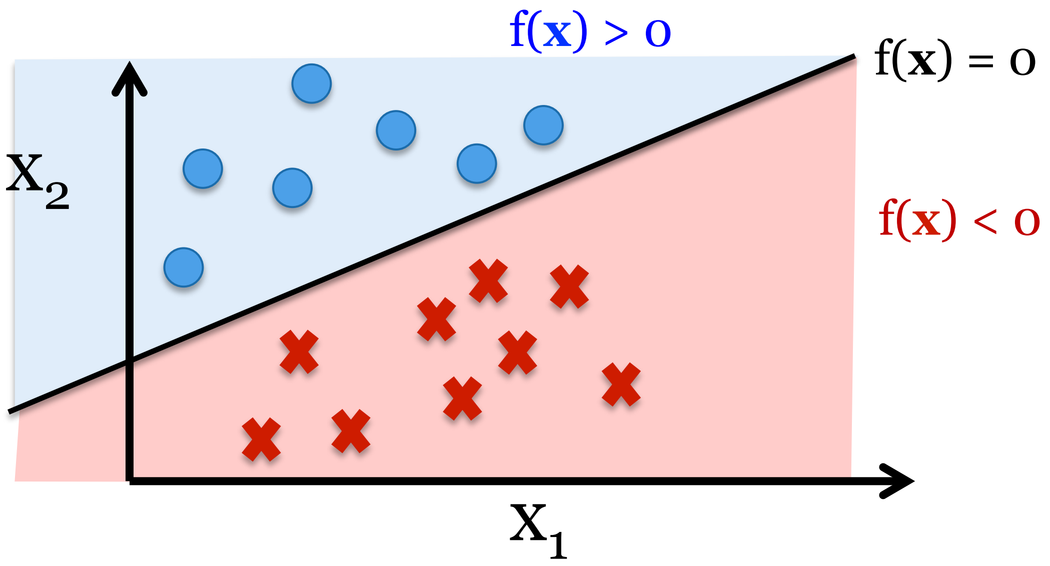
- We only have access to the training data
- But we care about accuracy on unseen (test) examples

Simplicity (Occam’s razor):

Prefer simpler models (e.g. fewer parameters).

- These (often) generalize better,
  and need less data for training.

Linear classifiers

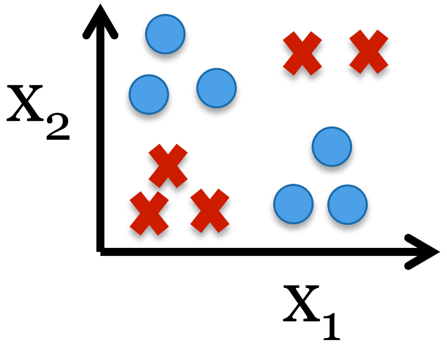


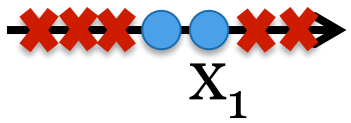
Many learning algorithms restrict the hypothesis space to **linear classifiers**:

$$f(\mathbf{x}) = w_0 + \mathbf{wx}$$

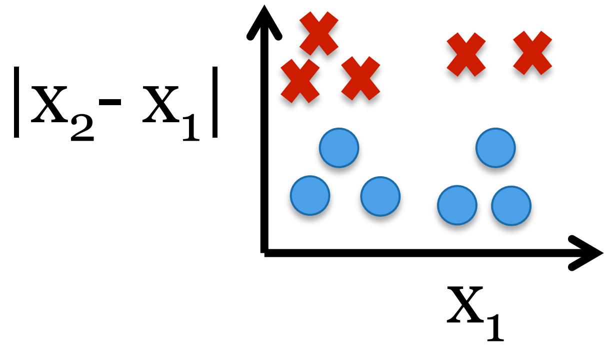
Linear Separability

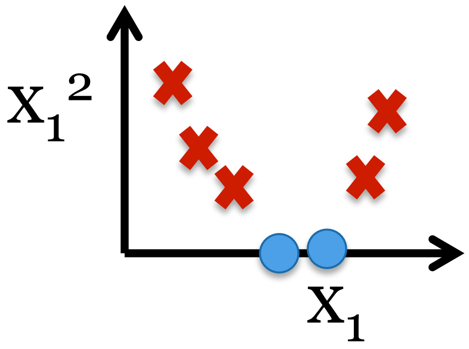
Not all data sets are linearly separable:





Sometimes, feature transformations help:





4. The learning algorithm

4. The learning algorithm

The learning task:
Given a labeled training data set

$$\mathcal{D}^{\text{train}} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$$

return a model (classifier) g: $\mathcal{X} \longmapsto \mathcal{Y}$
from the hypothesis space $\mathcal{H} \subseteq |\mathcal{Y}|^{|\mathcal{X}|}$

The learning algorithm performs a search in the hypothesis space $\mathcal{H}$ for the model g.

Batch versus online training

Batch learning:

The learner sees the complete training data, and only changes its hypothesis when it has seen the entire training data set.

Online training:

The learner sees the training data one example at a time, and can change its hypothesis with every new example

Practical issues

How to use your labeled data

Split your data into two (or three) sets:

– Training data (often 70-90%)

– Test data (often 10-20%)

– Development data (10-20%)

You need to report performance on test data, but you are not allowed to look at it.

You are allowed to look at the development data (and use it to tweak parameters)

Baseline performance

How difficult is your task?

You need to compare against a (reasonable) baseline (e.g. assign the majority class)

Ablation studies

How important are the different features (feature templates) you have designed?

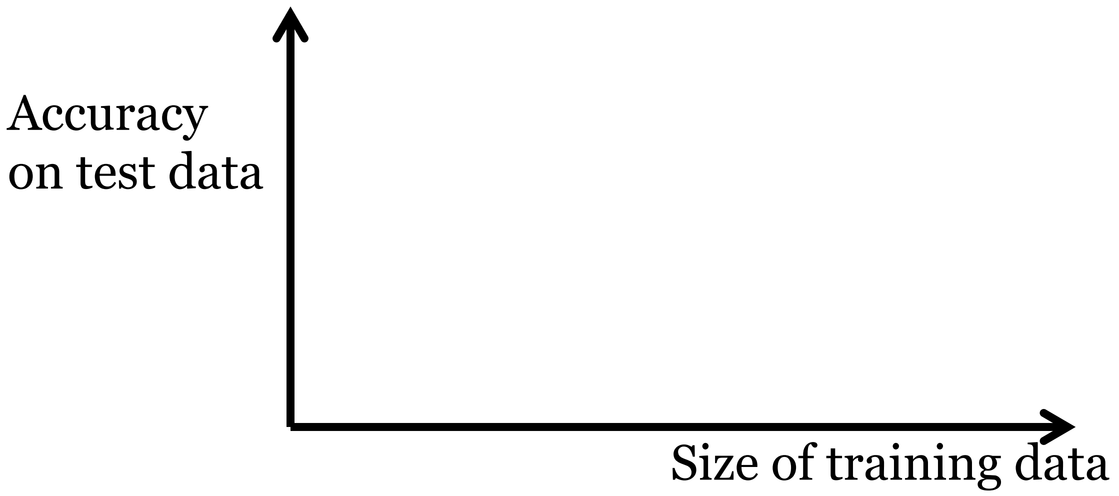
An ablation study compares models that use different subsets of the features/feature templates.

Learning curves

How much training data do you need?

Has your model converged?

How does your performance change with the amount of training data?



Today’s key concepts

Lecture 2 key concepts

- Instance space (typically a vector space):
  each instance = one feature vector $\mathbf{x} = (x_1 \ldots x_n)$
- Hypothesis space (supervised learning):
  Subset of functions from instances to labels
- Linear classifiers:
  Only consider linear functions $g(\mathbf{x}) = w_0 + \mathbf{wx}$
- Learning algorithms:
  Online vs. batch learning
- Training vs. test vs. development data