Supervised Learning: The Setup

Lecture 2

Machine Learning
Fall 2015

THE UNIVERSITY OF UTAH

Last lecture

- We saw
  - What is learning?
    - Learning as generalization
  - The badges game

This lecture

- More badges
- Formalizing supervised learning

Some slides based on lectures from Tom Dietterich, Dan Roth

The badges game

Let’s play

Name	Label
Claire Cardie	–
Peter Bartlett	+
Eric Baum	–
Haym Hirsh	+
Shai Ben-David	–
Michael I. Jordan	+

(Full data on the class website, you can stare at it longer if you want)

Let’s play

Name	Label
Claire Cardie	–
Peter Bartlett	+
Eric Baum	–
Haym Hirsh	+
Shai Ben-David	–
Michael I. Jordan	+

What is the label for “Peyton Manning”?

What about “Eli Manning”?

(Full data on the class website, you can stare at it longer if you want)

Let’s play

Name	Label
Claire Cardie	–
Peter Bartlett	+
Eric Baum	–
Haym Hirsh	+
Shai Ben-David	–
Michael I. Jordan	+

How were the labels generated?

(Full data on the class website, you can stare at it longer if you want)

Let's play

Name	Label
Claire Cardie	–
Peter Bartlett	+
Eric Baum	–
Haym Hirsh	+
Shai Ben-David	–
Michael I. Jordan	+

How were the labels generated?

```
If second letter of first name is a vowel, then + else -
```

(Full data on the class website, you can stare at it longer if you want)

Questions

1. Are you sure you got the correct function?
2. How did you arrive at it?
3. Learning issues:
   - Is this prediction or just modeling data?
   - How did you know that you should look at the letters?
   - What are vowels? Background knowledge?
   - What "learning algorithm" did you use?

What is supervised learning?

Instances and labels

X: Instance Space

The set of examples that need to be classified

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

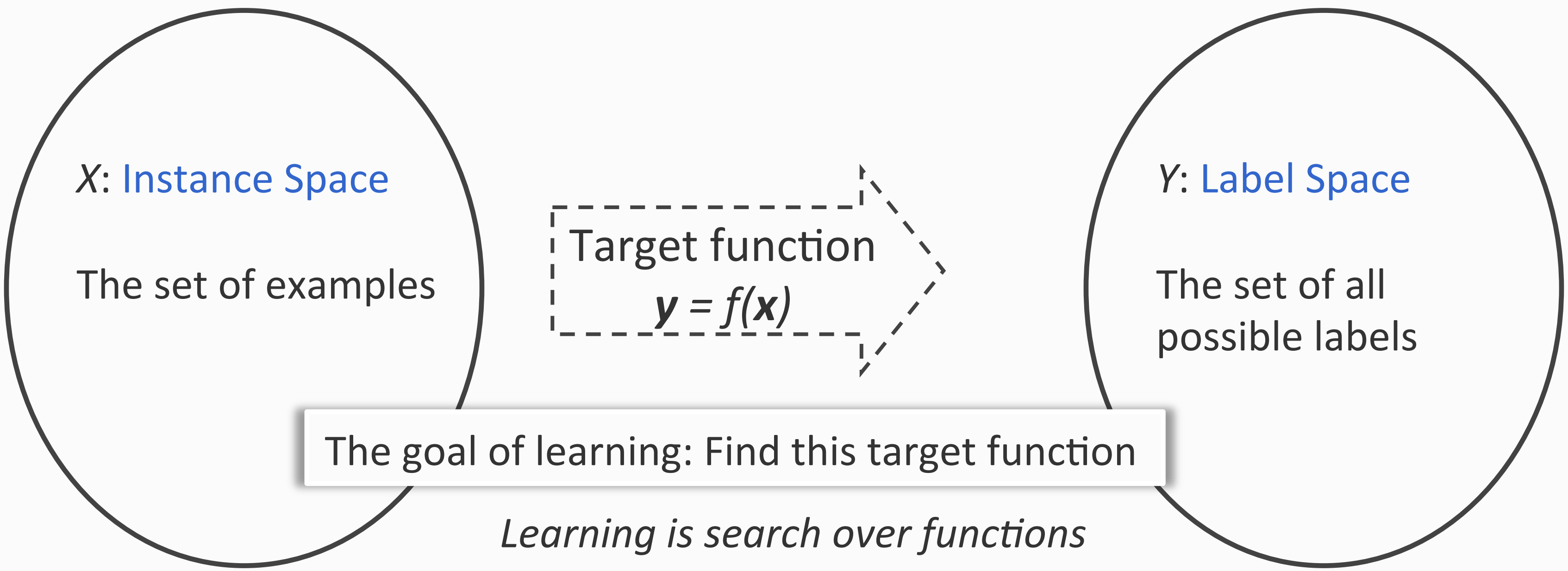
Y: Label Space

The set of all possible labels

Eg: The set of all possible names, sentences, images, emails, etc

Eg: {`Spam`, `Not-Spam`}, {+,-}, etc.

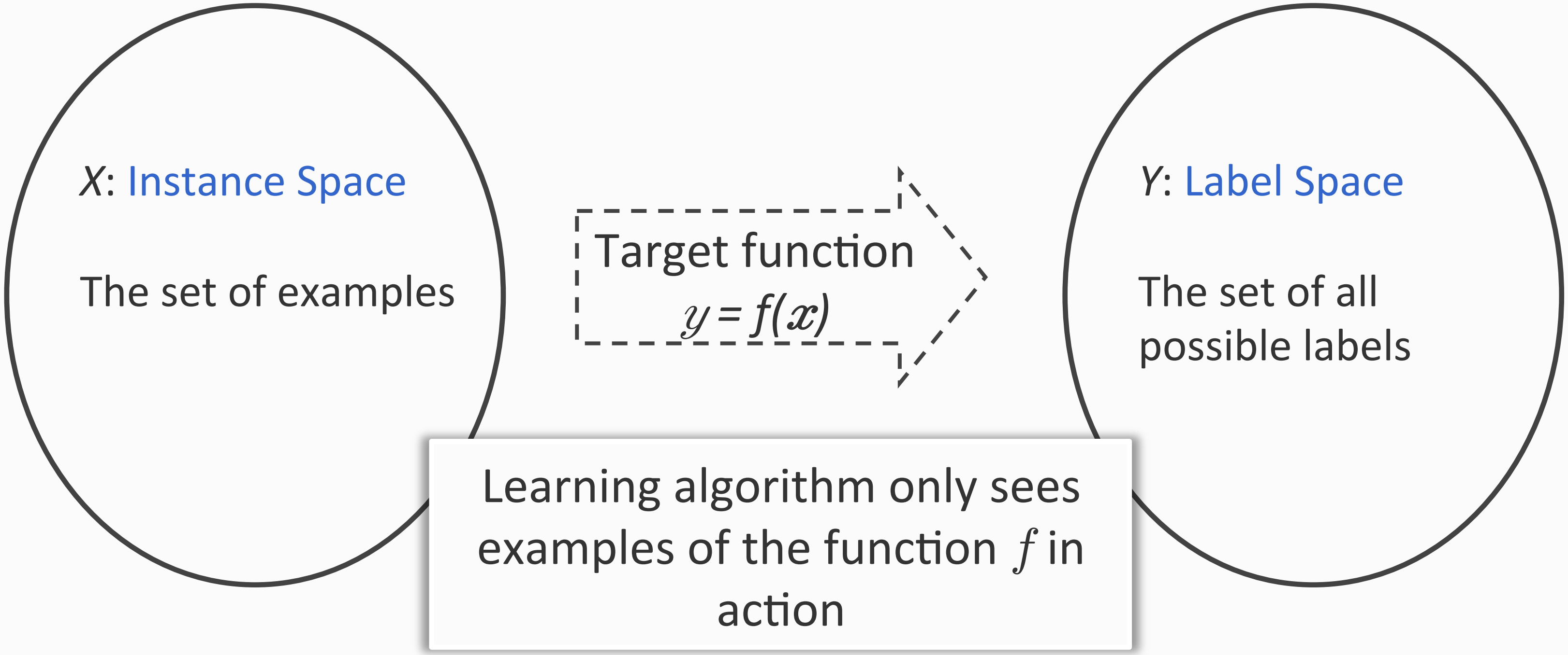
Instances and labels



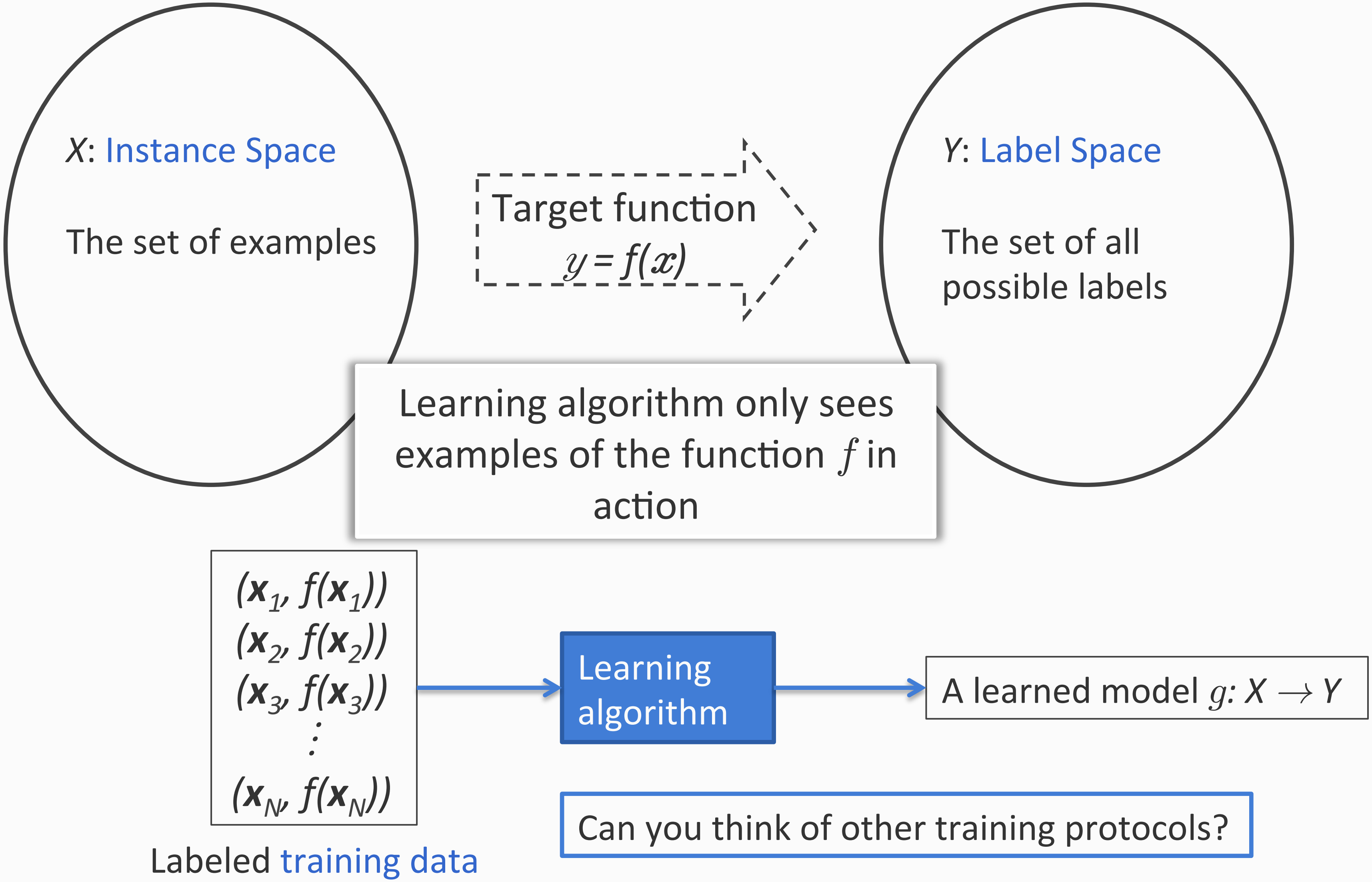
Eg: The set of all possible names, sentences, images, emails, etc

Eg: {`Spam`, `Not-Spam`}, {+,-}, etc.

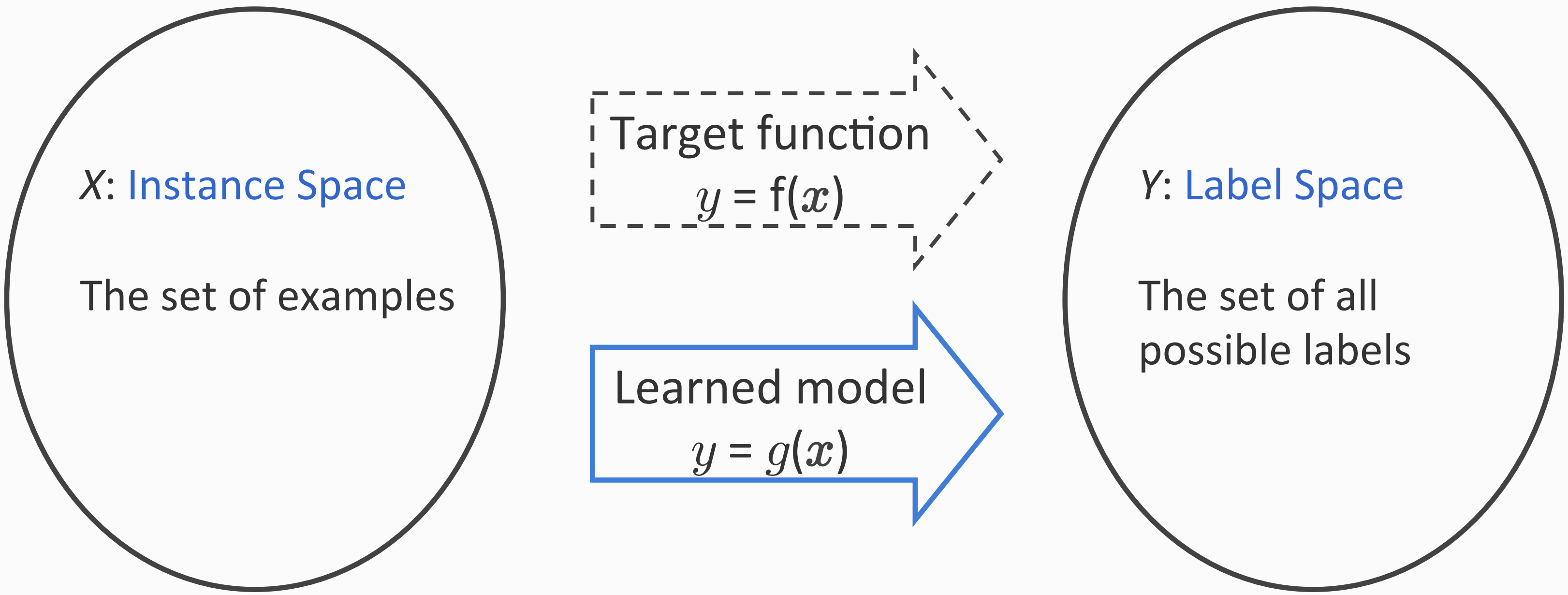
Supervised learning



Supervised learning: Training



Supervised learning: Evaluation



Draw test example $x \in X$ → $f(x)$, $g(x)$ → Are they different?

Apply the model to many test examples and compare to the target's prediction

Can you use test examples during training?

Supervised learning: General setting

- Given: Training examples of the form <x, $f(x)$>
  - The function f is an unknown function
- Typically the input x is represented in a *feature space*
  - Example: $x \in \{0,1\}^n$ or $x \in \Re^n$
  - A deterministic mapping from objects in your problem (emails) to features
- For a training example x, the value of $f(x)$ is called its *label*
- Goal: Find a good approximation for f
- The label determines the kind of problem we have
  - Binary classification: $f(\mathbf{x}) \in \{-1,1\}$
  - Multiclass classification: $f(\mathbf{x}) \in \{1, 2, 3, \cdots, K\}$
  - Regression: $f(\mathbf{x}) \in \Re$

Questions?

Nature of applications

- There is no human expert
  - Eg: Identify DNA binding sites
- Humans can perform a task, but can't describe how they do it
  - Eg: Object detection in images
- The desired function is hard to obtain in closed form
  - Eg: Stock market

Binary classification

- Spam filtering
  - Is an email spam or not?
- Recommendation systems
  - Given user’s movie preferences, will she like a new movie?
- Malware detection
  - Is an Android app malicious?
- Time series prediction
  - Will the future value of a stock increase or decrease with respect to its current value?

On using supervised learning

We should be able to decide:

1. What is our instance space?
   What are the inputs to the problem? What are the features?

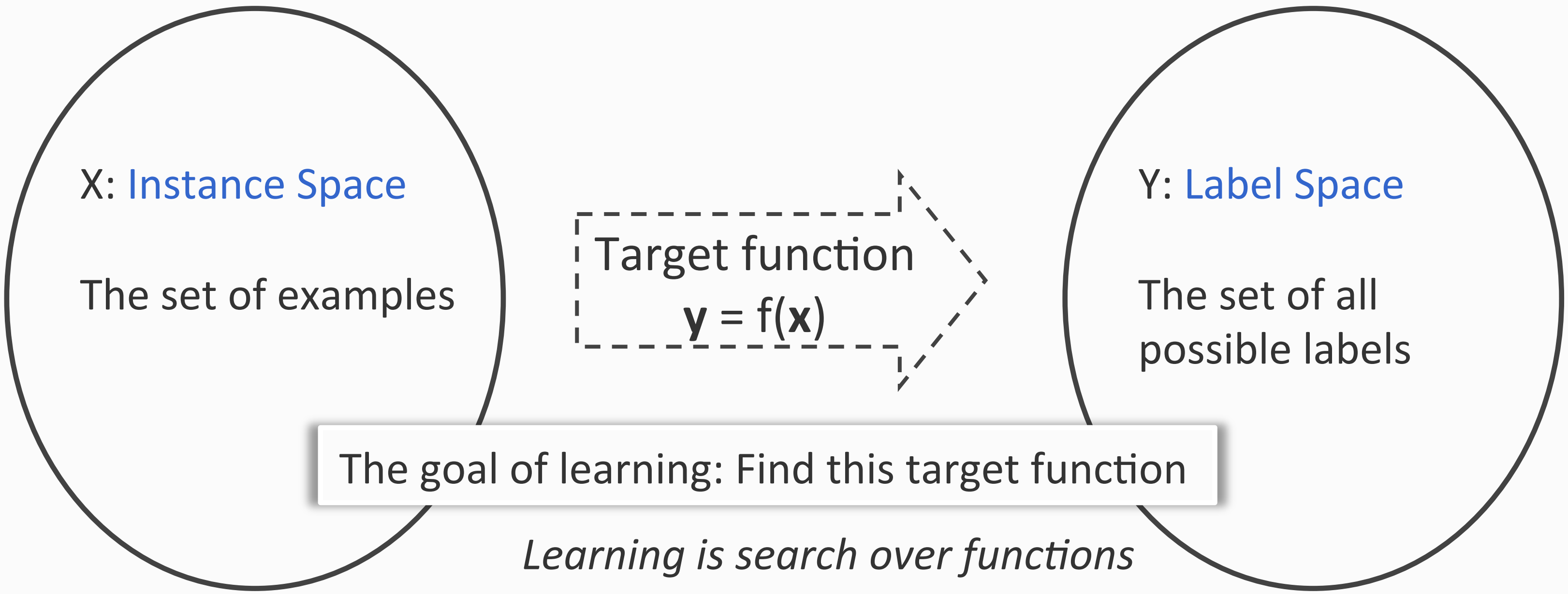
2. What is our label space?
   What is the learning task?

3. What is our hypothesis space?
   What functions should the learning algorithm search over?

4. What is our learning algorithm?
   How do we learn from the labeled data?

5. What is our loss function or evaluation metric?
   What is success?

1. The Instance Space *X*



Eg: The set of all possible names, sentences, images, emails, etc

Eg: {`Spam`, `Not-Spam`}, {+,-}, etc.

1. The Instance Space *X*

X: Instance Space

The set of examples

Ta

The goal of lear

Learn

Eg: The set of all possible names, sentences, images, emails, etc

Designing an appropriate instance space is crucial

Instances x ∈ X are defined by features/attributes

- Examples: Boolean features
  - Does the email contain the word "free"?
- Examples: Real valued features
  - What is the height of the person?
  - What was the stock price yesterday?

Eg: {Spam, Not-Spam}, {+,-}, etc.

1. The Instance Space *X*

Let’s brainstorm some features for the badges game

Instances as feature vectors

An input to the problem
(Eg: emails, names, images)

Feature function

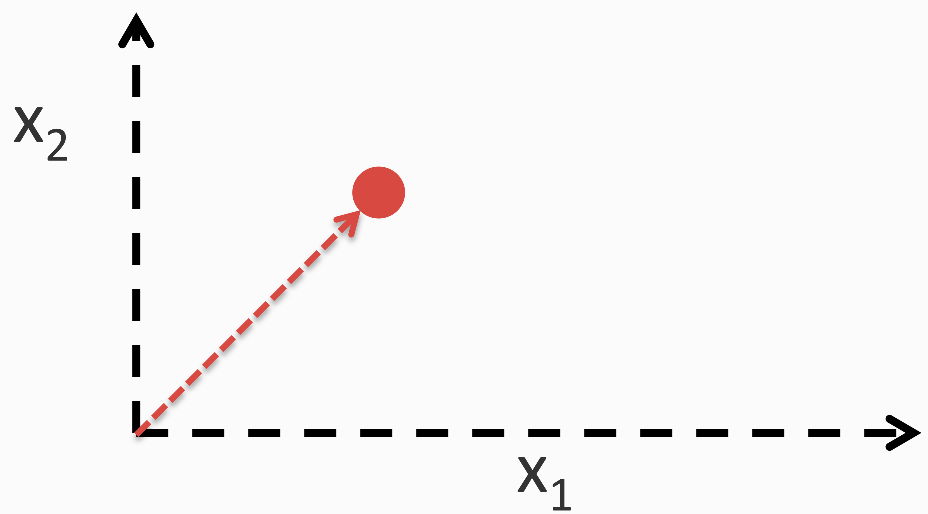
A feature vector

Feature functions a.k.a feature extractors

- Deterministic (for the most part)
- Convert the examples a collection of attributes
  - Very often easy to think of them as vectors
- Important part of the design of a learning based solution

Instances as feature vectors

- Features functions operate on inputs and produce a Boolean or a real number
  - Given a collection of feature functions, there is a deterministic mapping from instances to collections of Booleans or real numbers
- The instance space X ***is*** a N-dimensional vector space (e.g $\Re^N$ or $\{0,1\}^N$)
  - *Each dimension is one feature*
- Each $\mathbf{x} \in X$ is a feature vector
  - Each $\mathbf{x} = [x_1, x_2, \cdots, x_N]$ is a point in the vector space



Feature functions produce feature vectors

When designing feature functions, think of them as templates

- Feature extractor: *"The second letter of the name"*
  - `Naoki` $\rightarrow$ [1 0 0 0 …]
  - `Abe` $\rightarrow$ [0 1 0 0 …]
  - `Manning` $\rightarrow$ [1 0 0 0 …]
  - `Scrooge` $\rightarrow$ [0 0 1 0 …]
- Feature extractor: *"The length of the name"*
  - `Naoki` $\rightarrow$ 5
  - `Abe` $\rightarrow$ 3

Question: What is the length of this feature vector?

26 (One dimension per letter)

Good features are essential

- Good features decide how well a task can be learned
  - Eg: A bad feature function the badges game
    - "Is there a day of the week that begins with the last letter of the first name?"
- Much effort goes into designing features
  - Or maybe learning them
- Will touch upon general principles for designing good features
  - But feature definition largely domain specific
  - Comes with experience

On using supervised learning

✓ What is our instance space?
    What are the inputs to the problem? What are the features?

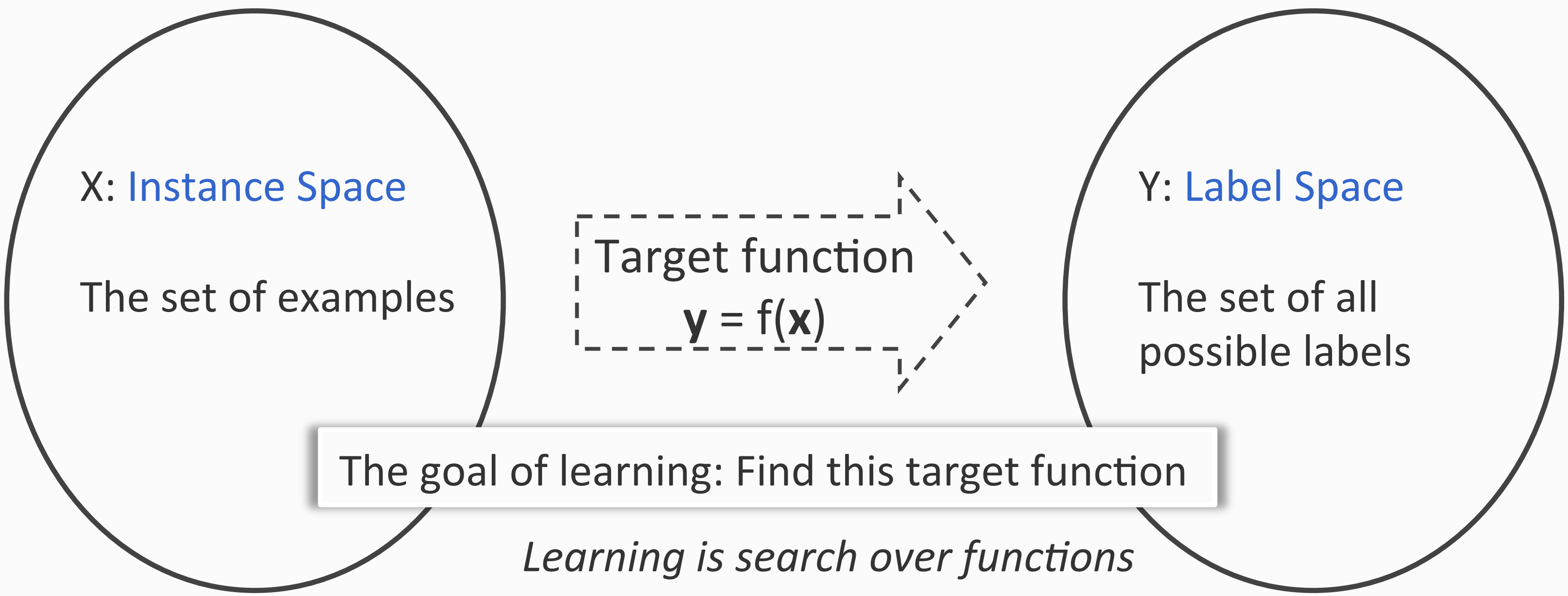
2. What is our label space?
    What is the learning task?

3. What is our hypothesis space?
    What functions should the learning algorithm search over?

4. What is our learning algorithm?
    How do we learn from the labeled data?

5. What is our loss function or evaluation metric?
    What is success?

2. The Label Space *Y*



Eg: The set of all possible names, sentences, images, emails, etc

Eg: {Spam, Not-Spam}, {+,-}, etc.

2. The Label Space *Y*

X: Instance Space

The set of examples

Target function
y = f(**x**)

Y: Label Space

The set of all possible labels

The goal of learning: Find this target function

Learning is search over functions

Eg: The set of all possible names, sentences, images, emails, etc

Eg: {`Spam`, `Not-Spam`}, {+,-}, etc.

2. The Label Space Y

Classification: The outputs are categorical

- Binary classification: Two possible labels
  - We will see a lot of this
- Multiclass classification: K possible labels
  - We may see a bit of this
- Structured classification: Graph valued outputs
  - A different class

Classification is the primary focus of this class

2. The Label Space *Y*

- The output space can be numerical
  - Regression:
    - Y is the set (or a subset) of real numbers
  - Ranking
    - Labels are ordinal
    - That is, there is an ordering over the labels
    - Eg: A Yelp 5-star review is only slightly different from a 4-star review, but very different from a 1-star review

On using supervised learning

✓ What is our instance space?

What are the inputs to the problem? What are the features?

✓ What is our label space?

What is the learning task?

3. What is our hypothesis space?

   What functions should the learning algorithm search over?

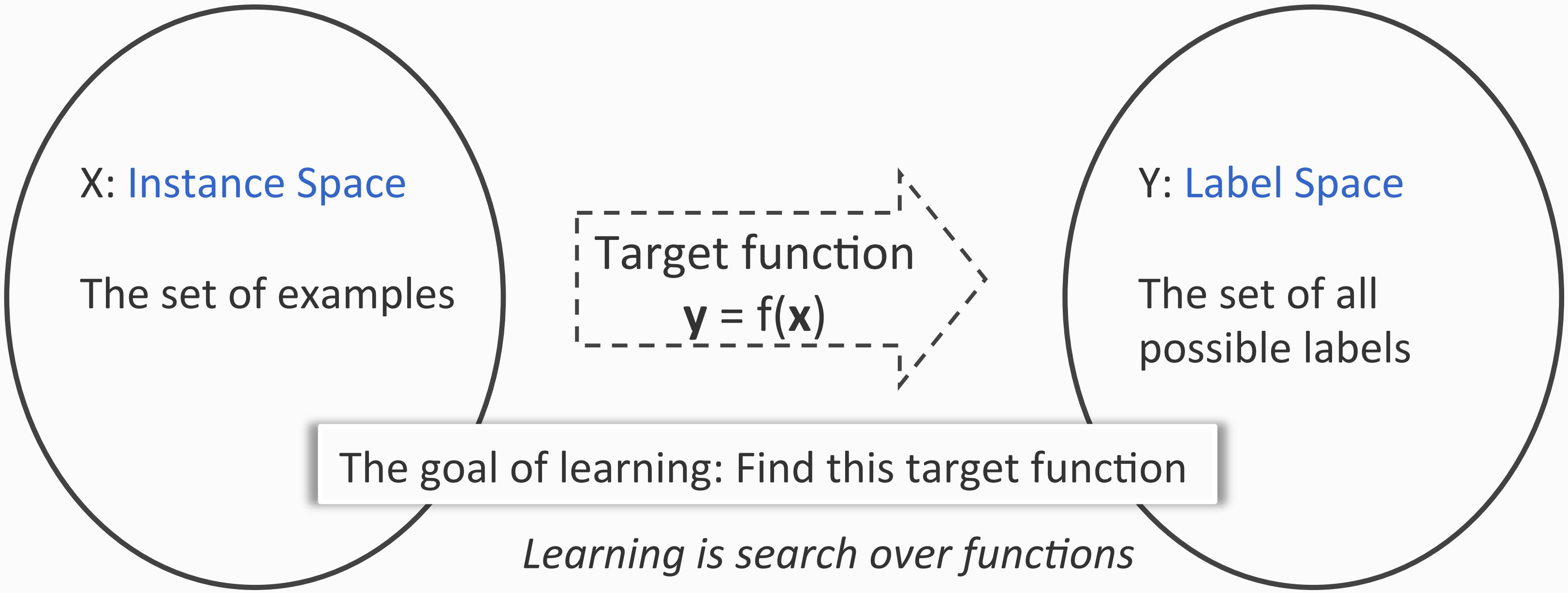
4. What is our learning algorithm?

   How do we learn from the labeled data?

5. What is our loss function or evaluation metric?

   What is success?

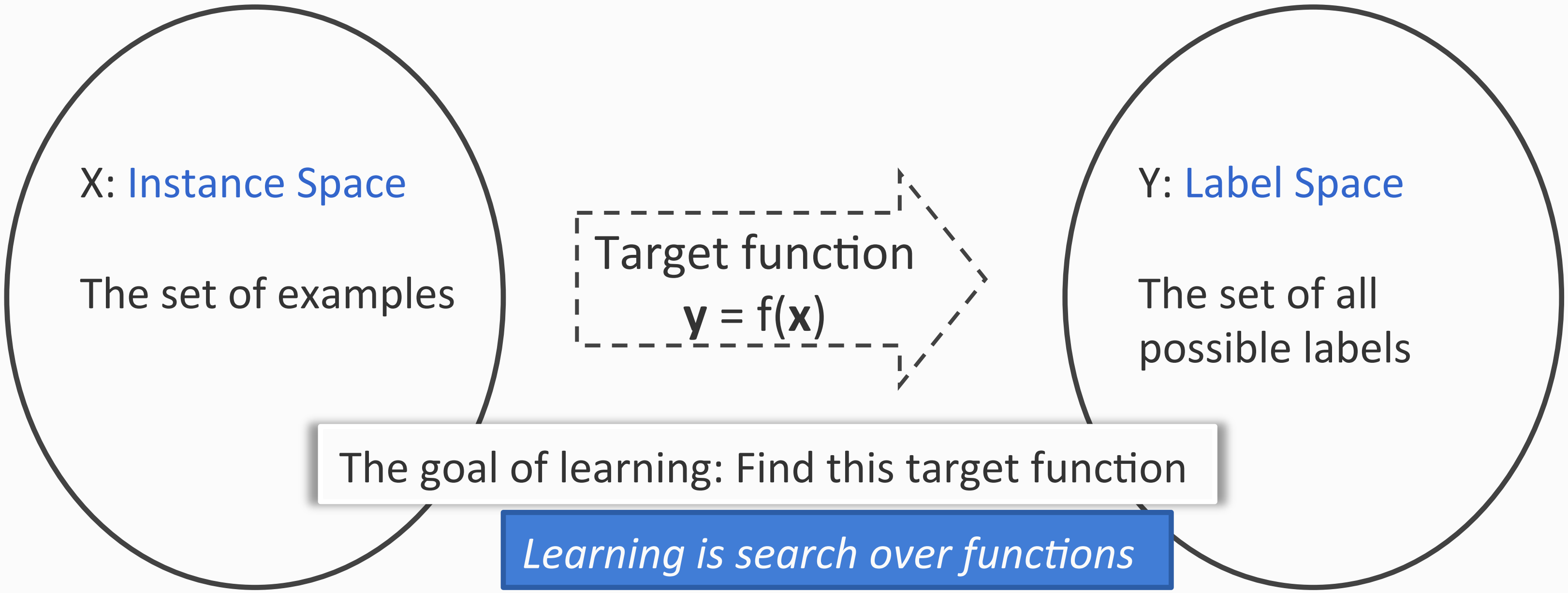
3. The Hypothesis Space



Eg: The set of all possible names, sentences, images, emails, etc

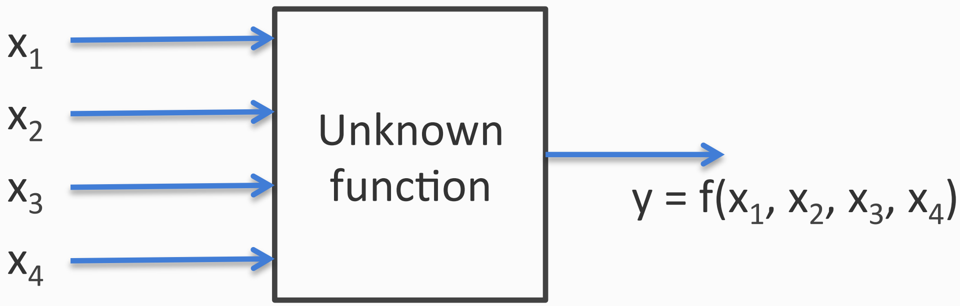
Eg: {`Spam`, `Not-Spam`}, {+,-}, etc.

3. The Hypothesis Space



The hypothesis space is the set of functions we consider for this search

The fundamental problem: Machine learning is ill-posed!



x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Can you learn this function?

What is it?

Is learning possible at all?

- There are $2^{16} = 65536$ possible Boolean functions over 4 inputs
  - Why? There are 16 possible outputs. Each way to fill these 16 slots is a different function, giving 2^{16} functions.

- We have seen only 7 outputs

- *How could we possibly know the rest without seeing every label?*
  - Think of an adversary filling in the labels every time you make a guess at the function

x_1	x_2	x_3	x_4	y
0	0	0	0	?
0	0	0	1	?
0	0	1	0	**0** $\leftarrow$
0	0	1	1	**1** $\leftarrow$
0	1	0	0	**0** $\leftarrow$
0	1	0	1	**0** $\leftarrow$
0	1	1	0	**0** $\leftarrow$
0	1	1	1	?
1	0	0	0	?
1	0	0	1	**1** $\leftarrow$
1	0	1	0	?
1	0	1	1	?
1	1	0	0	**0** $\leftarrow$
1	1	0	1	?
1	1	1	0	?
1	1	1	1	?

Is learning possible at all?

- There are $2^{16} = 65536$ possible Boolean functions over 4 inputs
  - Why? There are 16 possible outputs. Each way to fill these 16 slots is a different function, giving 2^{16} functions.

How could we possibly learn anything?

- We have seen only 7 outputs

- *How could we possibly know the rest without seeing every label?*
  - Think of an adversary filling in the labels every time you make a guess at the function

x_1	x_2	x_3	x_4	y
0	0	0	0	?
0	0	0	1	?
0	0	1	0	**0** $\leftarrow$
0	0	1	1	**1** $\leftarrow$
0	1	0	0	**0** $\leftarrow$
[illegible]	[illegible]	[illegible]	1	**0** $\leftarrow$
[illegible]	[illegible]	[illegible]	0	**0** $\leftarrow$
0	1	1	1	?
1	0	0	0	?
1	0	0	1	**1** $\leftarrow$
1	0	1	0	?
1	0	1	1	?
1	1	0	0	**0** $\leftarrow$
1	1	0	1	?
1	1	1	0	?
1	1	1	1	?

Solution: Restrict the search space

A *hypothesis space* is the set of possible functions we consider

- We were looking at the space of all Boolean functions
- Instead choose a hypothesis space that is smaller than the space of all functions
  - Only simple conjunctions (with four variables, there are only 16 conjunctions without negations)
  - Simple disjunctions
  - m-of-n rules: Fix a set of n variables. At least m of them must be true
  - Linear functions

  ⋮

Example

Hypothesis space 1

Simple conjunctions

There are only 16 simple **conjunctive rules** of the form $g(\mathbf{x})=x_i \wedge x_j \wedge x_k$

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Is there a *consistent* hypothesis in this space?

Example

Hypothesis space 1

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Simple conjunctions

There are only 16 simple **conjunctive rules** of the form $g(\mathbf{x})=x_i \wedge x_j \wedge x_k$

Rule	Counterexample
False	1001 1
x_1	1100 0
x_2	0100 0
x_3	0110 0
x_4	0101 1
$x_1 \wedge x_2$	1100 0
$x_1 \wedge x_3$	0011 1
$x_1 \wedge x_4$	0011 1

Example

Hypothesis space 1

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Simple conjunctions

There are only 16 simple **conjunctive rules** of the form $g(\mathbf{x})=x_i \wedge x_j \wedge x_k$

Rule	Counterexample
False	1001 1
x_1	1100 0
x_2	0100 0
x_3	0110 0
x_4	0101 1
$x_1 \wedge x_2$	1100 0
$x_1 \wedge x_3$	0011 1
$x_1 \wedge x_4$	0011 1

Rule	Counterexample
$x_2 \wedge x_3$	0011 1
$x_2 \wedge x_4$	0011 1
$x_3 \wedge x_4$	1001 1
$x_1 \wedge x_2 \wedge x_3$	0011 1
$x_1 \wedge x_2 \wedge x_4$	0011 1
$x_1 \wedge x_3 \wedge x_4$	0011 1
$x_2 \wedge x_3 \wedge x_4$	0011 1
$x_1 \wedge x_2 \wedge x_3 \wedge x_4$	0011 1

Example

Hypothesis space 1

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Simple conjunctions

There are only 16 simple **conjunctive rules** of the form $g(\mathbf{x}) = x_i \wedge x_j \wedge x_k$

Rule	Counterexample	Rule	Counterexample
False	1001 1	$x_2 \wedge x_3$	0011 1
x_1			0011 1
x_2			1001 1
x_3			0011 1
x_4	0101 1	$x_1 \wedge x_2 \wedge x_4$	0011 1
$x_1 \wedge x_2$	1100 0	$x_1 \wedge x_3 \wedge x_4$	0011 1
$x_1 \wedge x_3$	0011 1	$x_2 \wedge x_3 \wedge x_4$	0011 1
$x_1 \wedge x_4$	0011 1	$x_1 \wedge x_2 \wedge x_3 \wedge x_4$	0011 1

No simple conjunction explains the data!

Our hypothesis space is too small

Solution: Restrict the search space

- A *hypothesis space* is the set of possible functions we consider
  - We were looking at the space of all Boolean functions
  - Instead choose a hypothesis space that is smaller than the space of all functions
    - Only simple conjunctions (with four variables, there are only 16 conjunctions without negations)
    - m-of-n rules: Pick a set of n variables. At least m of them must be true
    - Linear functions
- How do we pick a hypothesis space?
  - Using some prior knowledge (or by guessing)
- What if the hypothesis space is so small that nothing in it agrees with the data?
  - We need a hypothesis space that is flexible enough

Example

Hypothesis space 2

m-of-n rules

Pick a subset with n variables. Y = 1 if at least m of them are 1

Example: at least 2 of $\{x_1, x_3, x_4\}$ should be 1

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Is there a consistent hypothesis in this space?

Try to check if there is one

First, how many m-of-n rules are there for four variables?

Views of learning

- Learning is the removal of ***remaining*** uncertainty
  - If we knew that the unknown function is a simple conjunction, we could use the training data to figure out which one it is

- Requires guessing a ***good, small*** hypothesis class
  - And we could be wrong
  - We could find a consistent hypothesis and still be incorrect with a new example!

On using supervised learning

✓ What is our instance space?

What are the inputs to the problem? What are the features?

✓ What is our label space?

What is the learning task?

✓ What is our hypothesis space?

What functions should the learning algorithm search over?

4. What is our learning algorithm?

   How do we learn from the labeled data?

5. What is our loss function or evaluation metric?

   What is success?