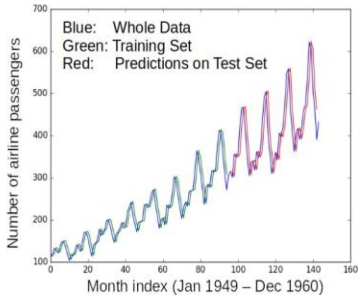
Intro to Machine Learning

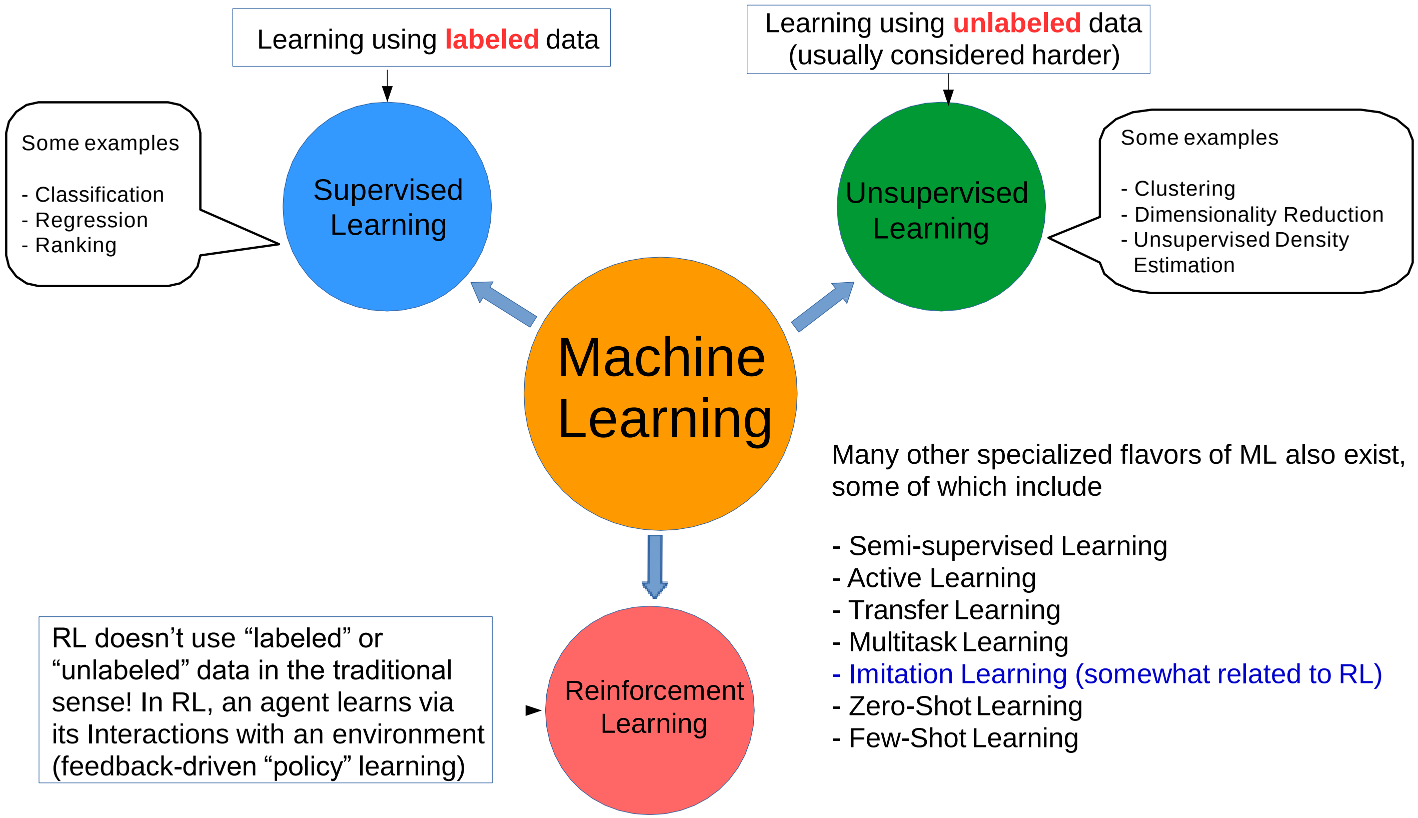
Machine Learning (ML)

- Designing algorithms that ingest data and learn a (hypothesized) model of the data
- The learned model can be used to
  - Detect patterns/structures/themes/trends etc. in the data
  - Make predictions about future data and make decisions



- Modern ML algorithms are heavily "data-driven"
  - No need to pre-define and hard-code all the rules (usually infeasible/impossible anyway) The
  - rules are not "static"; can adapt as the ML algo ingests more and more data

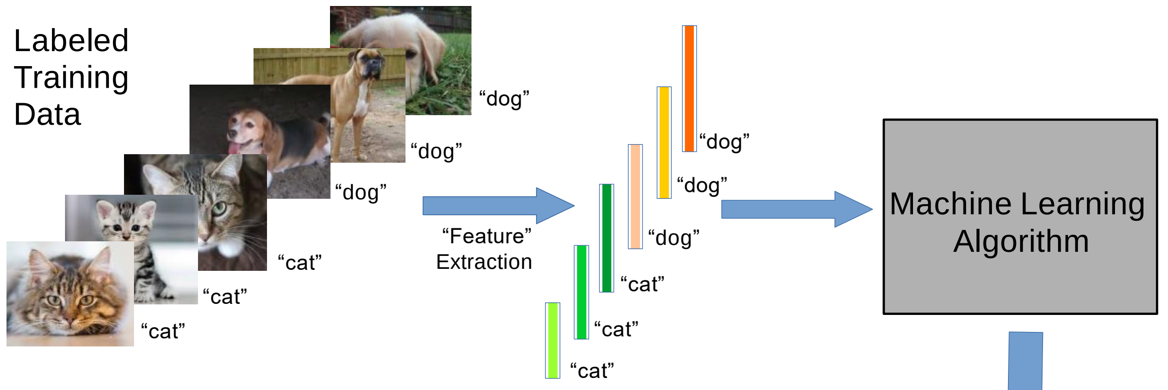
A Loose Taxonomy for ML



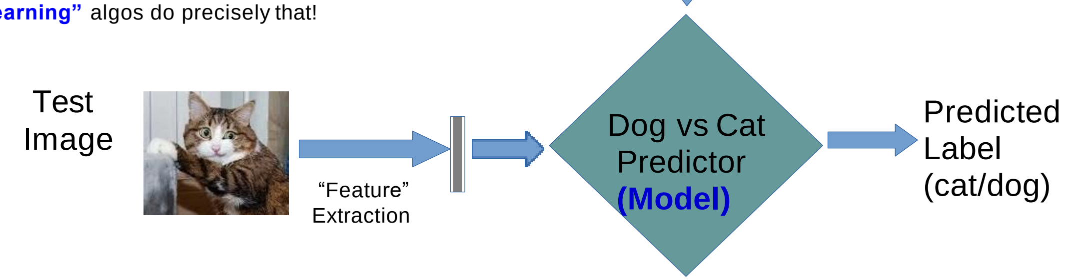
Many other specialized flavors of ML also exist, some of which include

- Semi-supervised Learning
- Active Learning
- Transfer Learning
- Multitask Learning
- Imitation Learning (somewhat related to RL)
- Zero-Shot Learning
- Few-Shot Learning

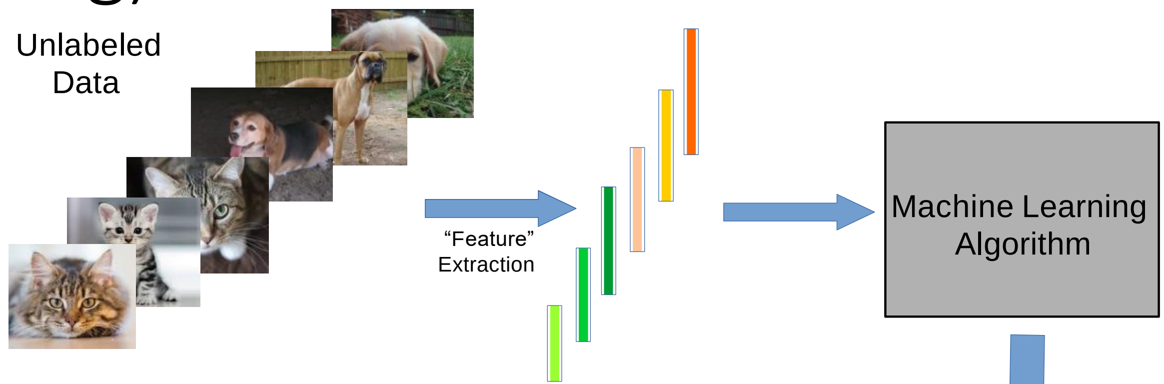
A Typical Supervised Learning Workflow (for Classification)

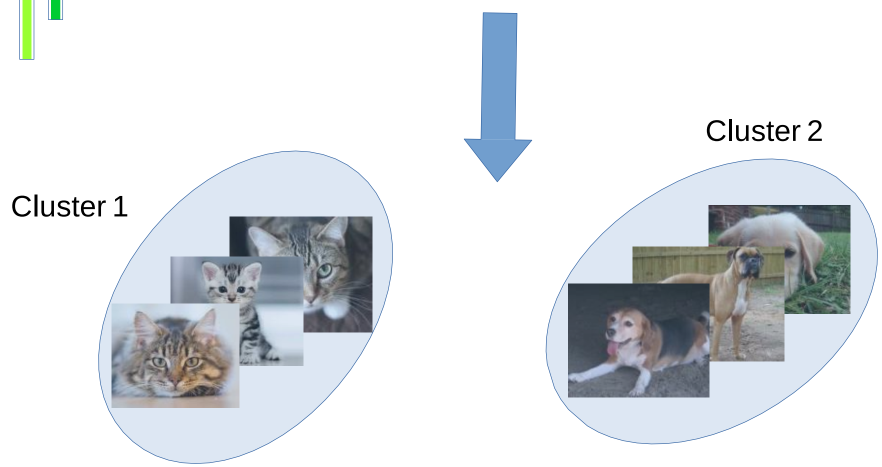


Note: The feature extraction phase may be part of the machine learning algorithm itself
(referred to “feature learning” or “representation learning”)
Modern **“deep learning”** algos do precisely that!



A Typical Unupervised Learning Workflow (for Clustering)





Note: Unsupervised Learning too can have (and often has) a "test" phase. E.g., in this case, given a new cat/dog image, predict which of the two clusters it belongs to.

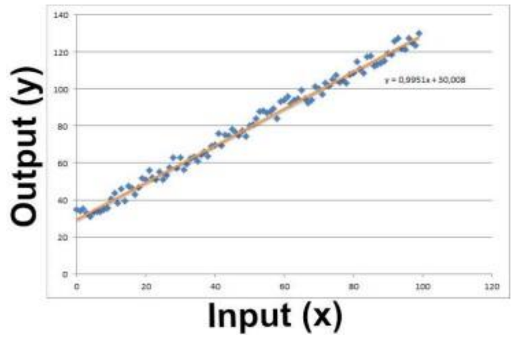
Can do it by assigning the image to the cluster with closer centroid

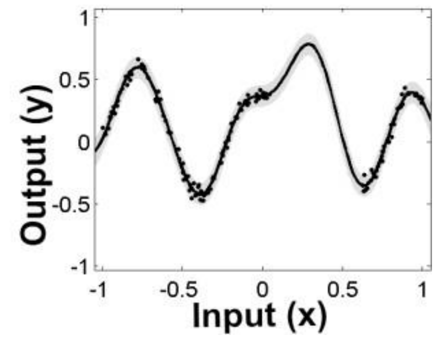
Geometric View of Some Basic ML Problems

Regression

Supervised Learning: Learn a line/curve (the "model") using training data consisting of Input-output pairs (each output is a real-valued number)

Use it to predict the outputs for new "test" inputs

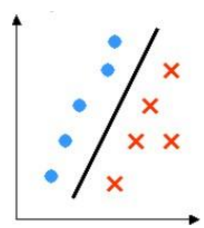




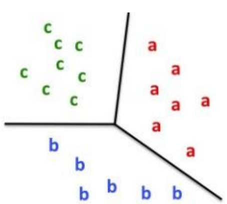
Classification

Supervised Learning: Learn a linear/nonlinear separator (the "model") using training data consisting of input-output pairs (each output is discrete-valued "label" of the corresponding input)

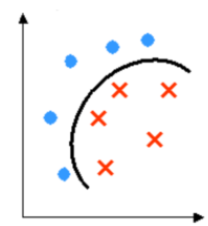
Use it to predict the labels for new "test" inputs



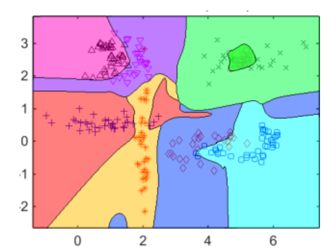
Two-Class (binary) Linear Classification



Multi-Class Linear Classification



Two-Class (binary) Nonlinear Classification

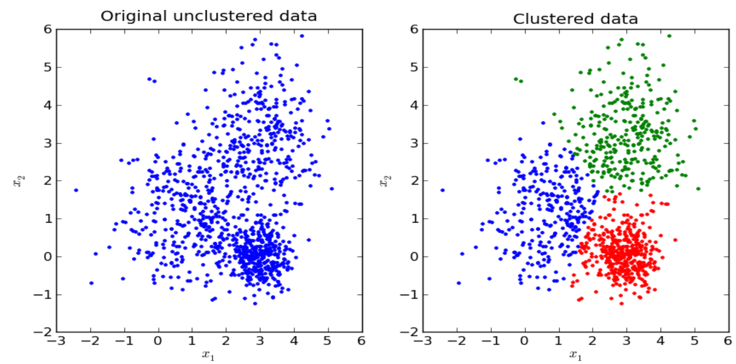


Multi-Class Nonlinear Classification

Geometric View of Some Basic ML Problems

Clustering

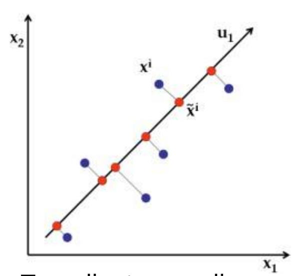
Unsupervised Learning: Learn the grouping structure for a given set of unlabeled inputs



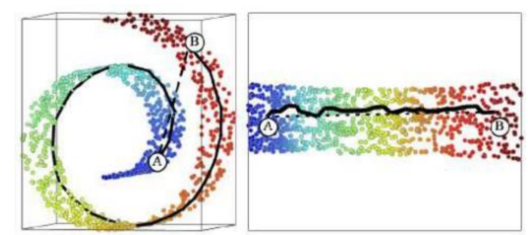
Dimensionality Reduction

Unsupervised Learning: Learn a Low-dimensional representation for a given set of high-dimensional inputs

Note: DR also comes in supervised flavors (supervised DR)



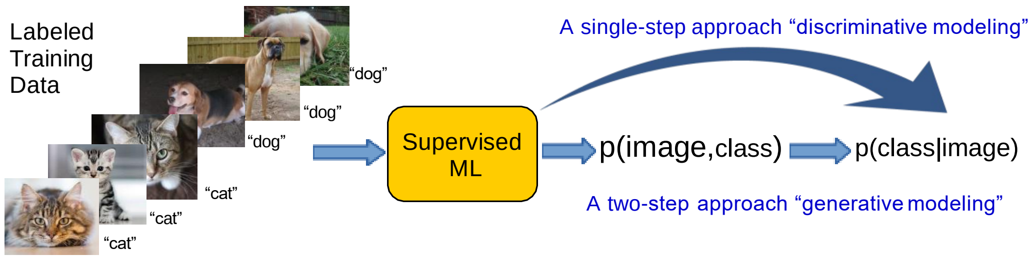
Two-dim to one-dim linear projection



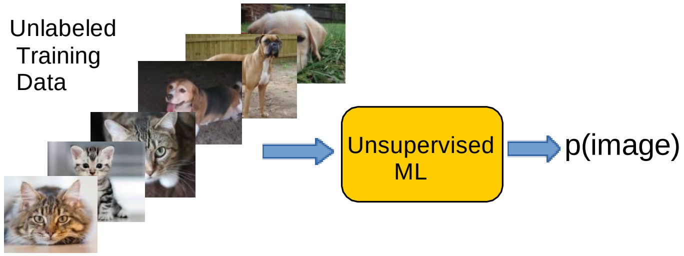
Three-dim to two-dim nonlinear projection (a.k.a. manifold learning)

Machine Learning = Probability Density Estimation

- Supervised Learning ("predict y given x") can be thought of as estimating $p(y|x)$



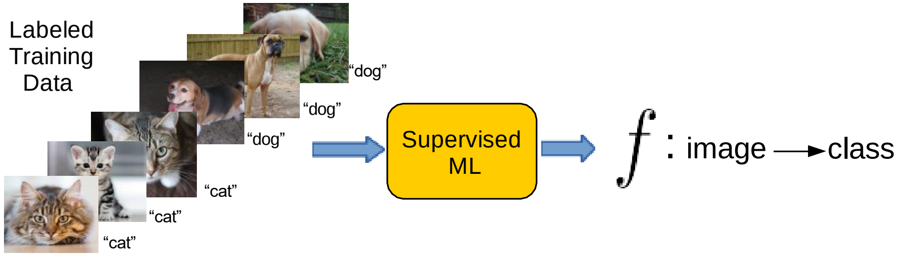
- Unsupervised Learning ("model x") can also be thought of as estimating $p(x)$



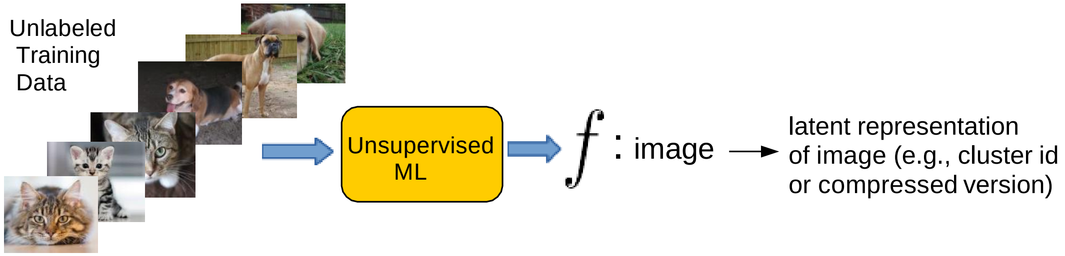
- Harder for Unsupervised Learning because there is no supervision y
- Other ML paradigms (e.g., Reinforcement Learning) can be thought of as learning prob. density

Machine Learning = Function Approximation

- Supervised Learning ("predict y given x") can be thought learning a function that maps x to y



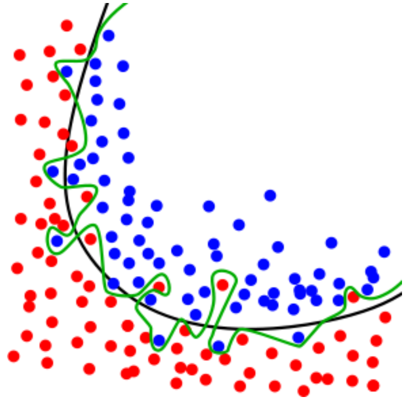
- Unsupervised Learning ("model x") can also be thought of as learning a function that maps x to some useful latent representation of x



- Harder for Unsupervised Learning because there is no supervision y
- Other ML paradigms (e.g., Reinforcement Learning) can be thought of as doing function approx.

Overfitting and Generalization

- Doing well on the training data is not enough for an ML algorithm

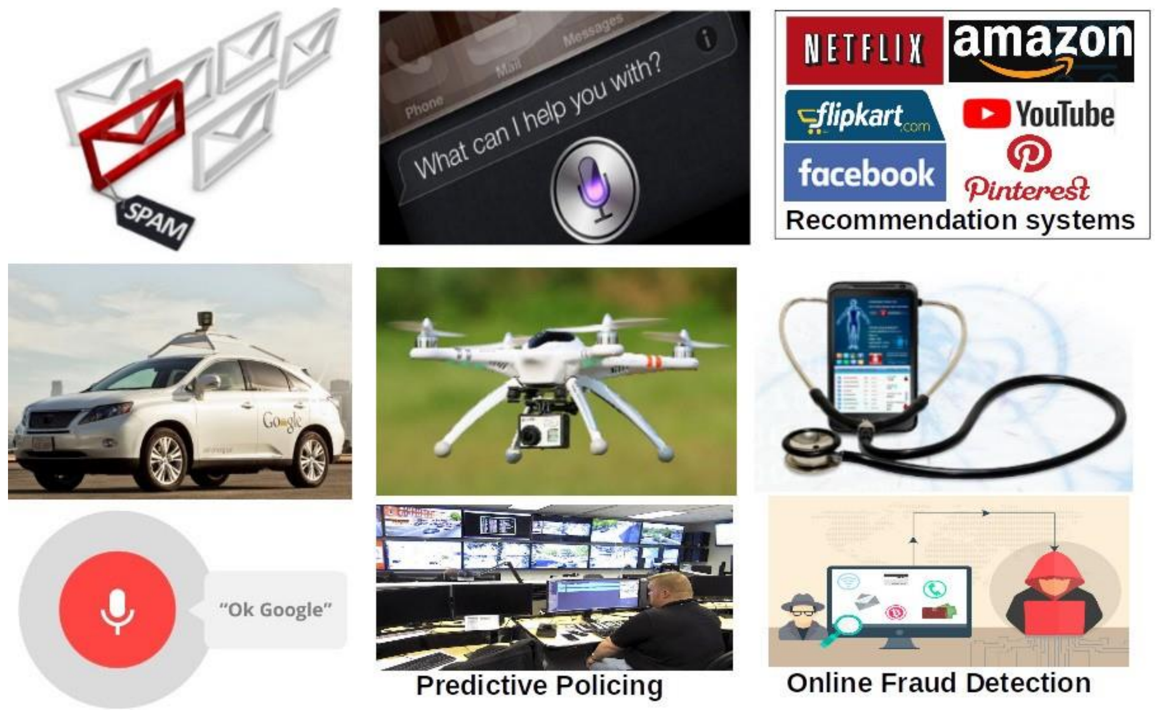


- Trying to do too well (or perfectly) on training data may lead to bad "generalization"
- Generalization: Ability of an ML algorithm to do well on future "test" data
- Simple models/functions tend to prevent overfiting and generalize well: A key principle in designing ML algorithms (called "regularization"; more on this later)

Picture courtesy: Wikipedia

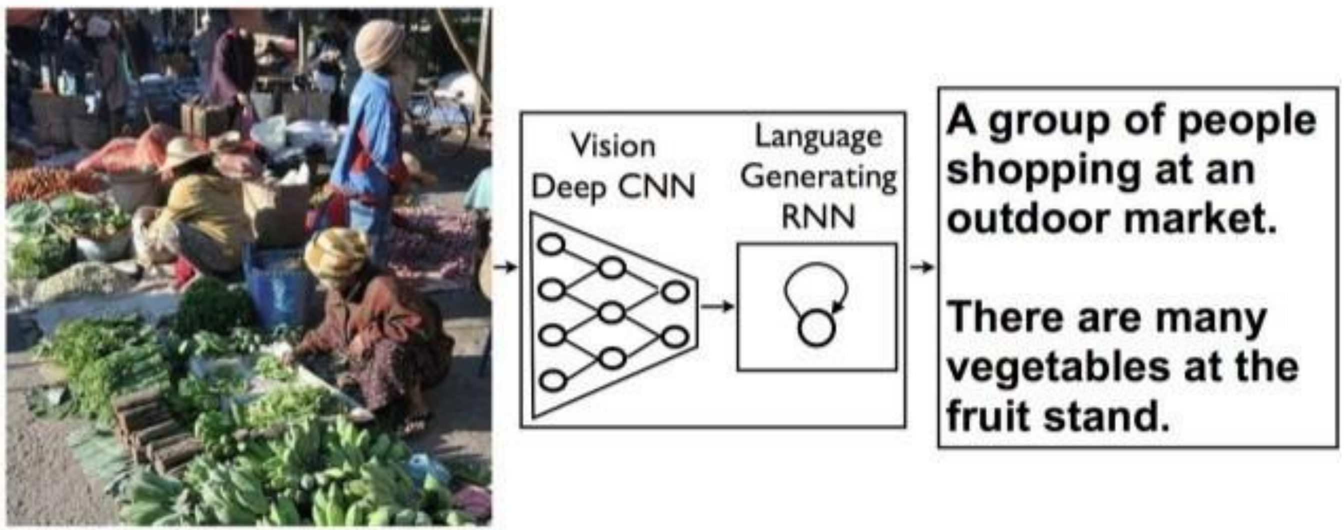
Machine Learning in the real-world

Broadly applicable in many domains (e.g., internet, robotics, healthcare and biology, computer vision, NLP, databases, computer systems, finance, etc.).



Machine Learning helps Computer Vision

ML algorithms can learn to generate captions for images

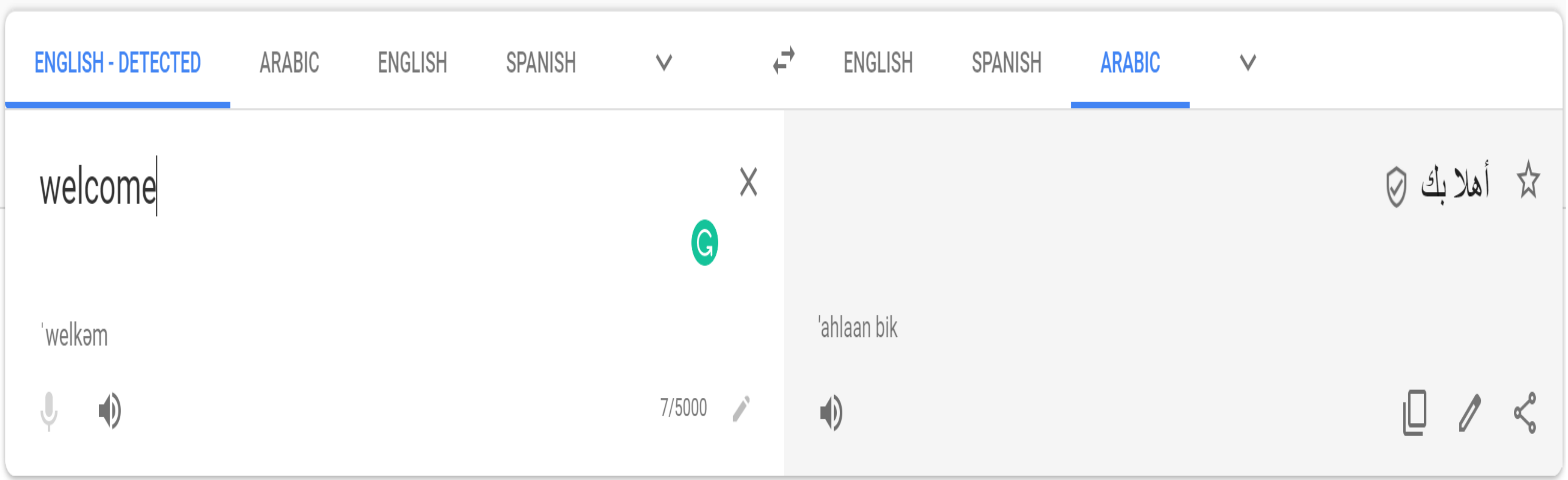


http://arxiv.org/abs/1411.4555 “Show and Tell: A Neural Image Caption Generator”

Machine Learning helps Computer Vision

ML algorithms can learn to answer questions about images (Visual QA)

Machine Learning helps NLP



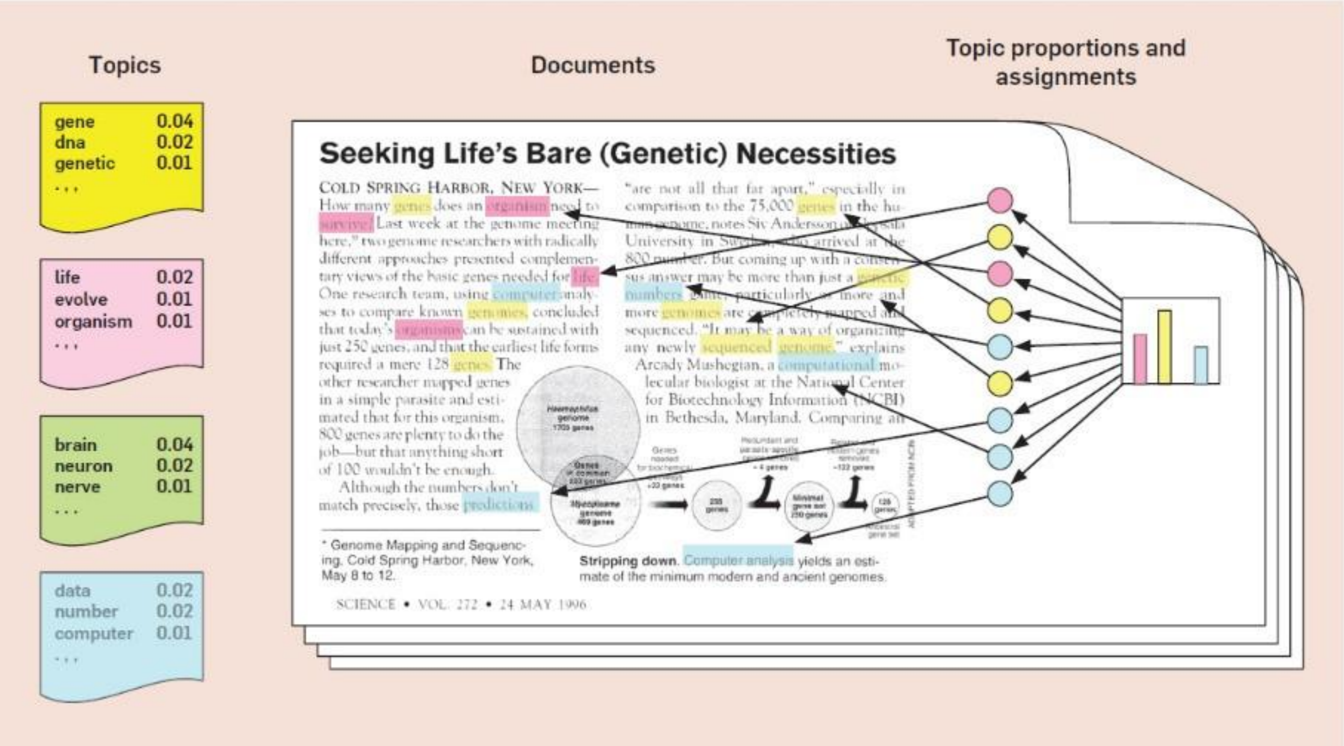
Machine Learning helps NLP

ML algorithms can learn to summarize text

Input: Article 1st sentence	Model-written headline
metro-goldwyn-mayer reported a third-quarter net loss of dlrs 16 million due mainly to the effect of accounting rules adopted this year	mgm reports 16 million net loss on higher revenue
starting from july 1, the island province of hainan in southern china will implement strict market access control on all incoming livestock and animal products to prevent the possible spread of epidemic diseases	hainan to curb spread of diseases
australian wine exports hit a record 52.1 million liters worth 260 million dollars (143 million us) in september, the government statistics office reported on monday	australian wine exports hit record high in september

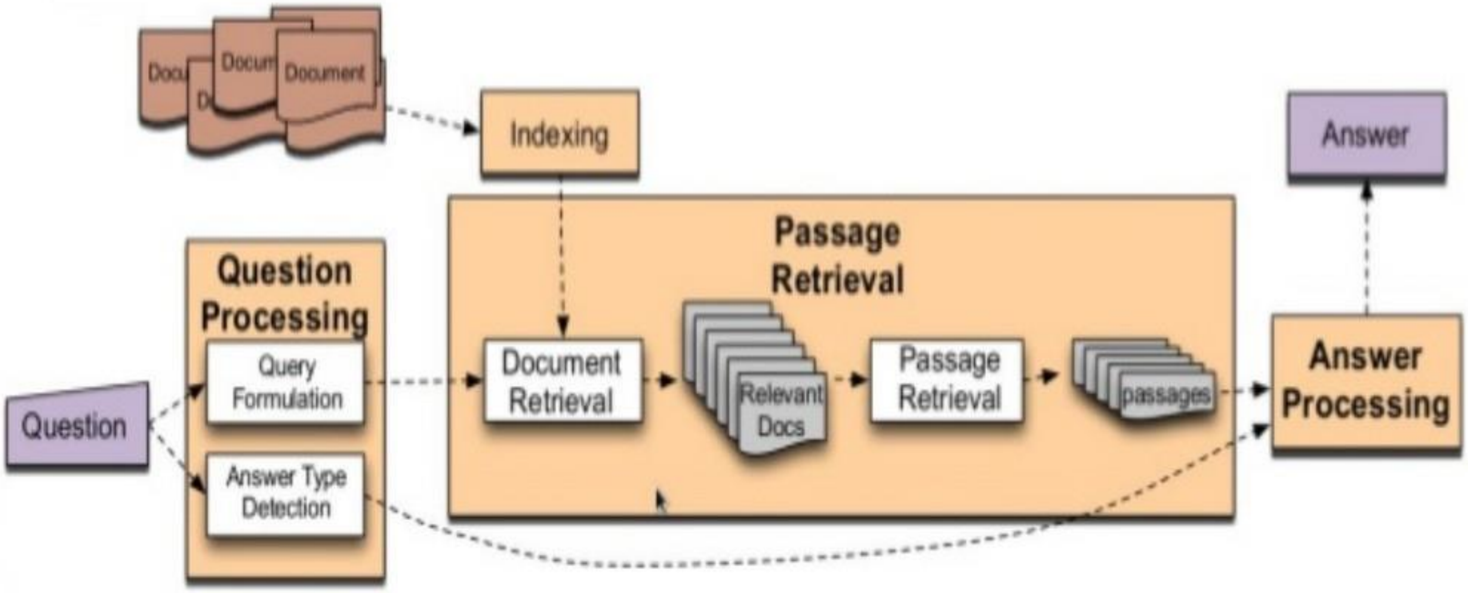
Machine Learning helps NLP

ML algorithms can learn the topics in a text corpus ("Topic Modeling")



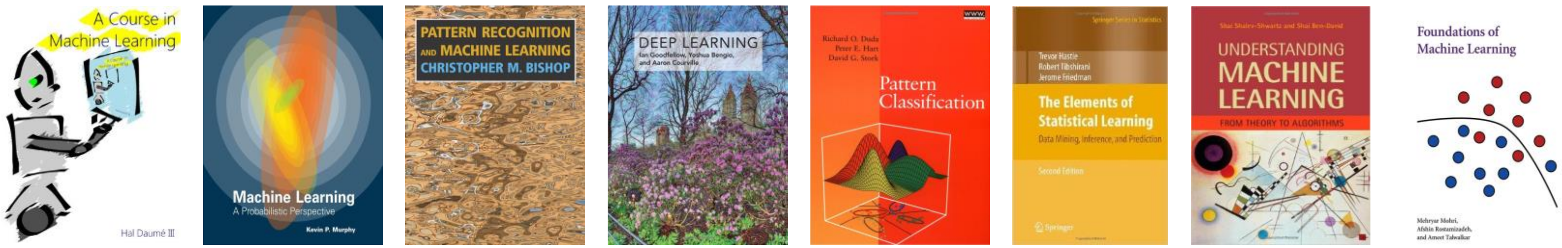
Machine Learning helps Search and Info Retrieval

ML algorithms can learn to search for the answer to a given question from a large database of documents



Textbook and References

- Many excellent texts but none "required". Some of them include (list not exhaustive)



- Different books might vary in terms of
  - Set of topics covered
  - General approach taken e.g., classical statistics, deep learning, probabilistic/Bayesian, theory
  - Terminology and notation (beware of this especially)
- Avoid using too many sources until you have developed a reasonable understanding of a concept
- We will provide you the reading material from the relevant sources

What is Learning

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?

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

Raw test data

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

Labeled test data

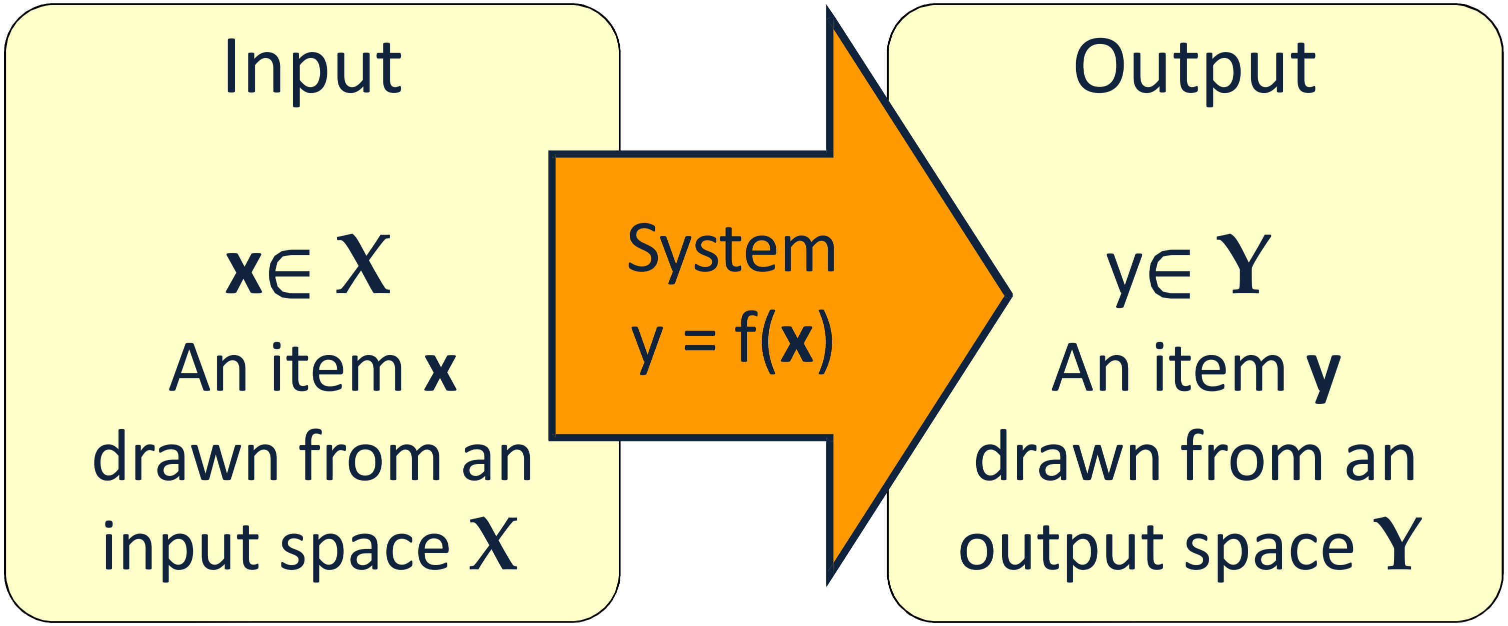
? Shivani Agarwal
+ Gerald F. DeJong
- Chris Drummond
+ Yolanda Gil
- Attilio Giordana
+ Jiarong Hong

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

What is Learning

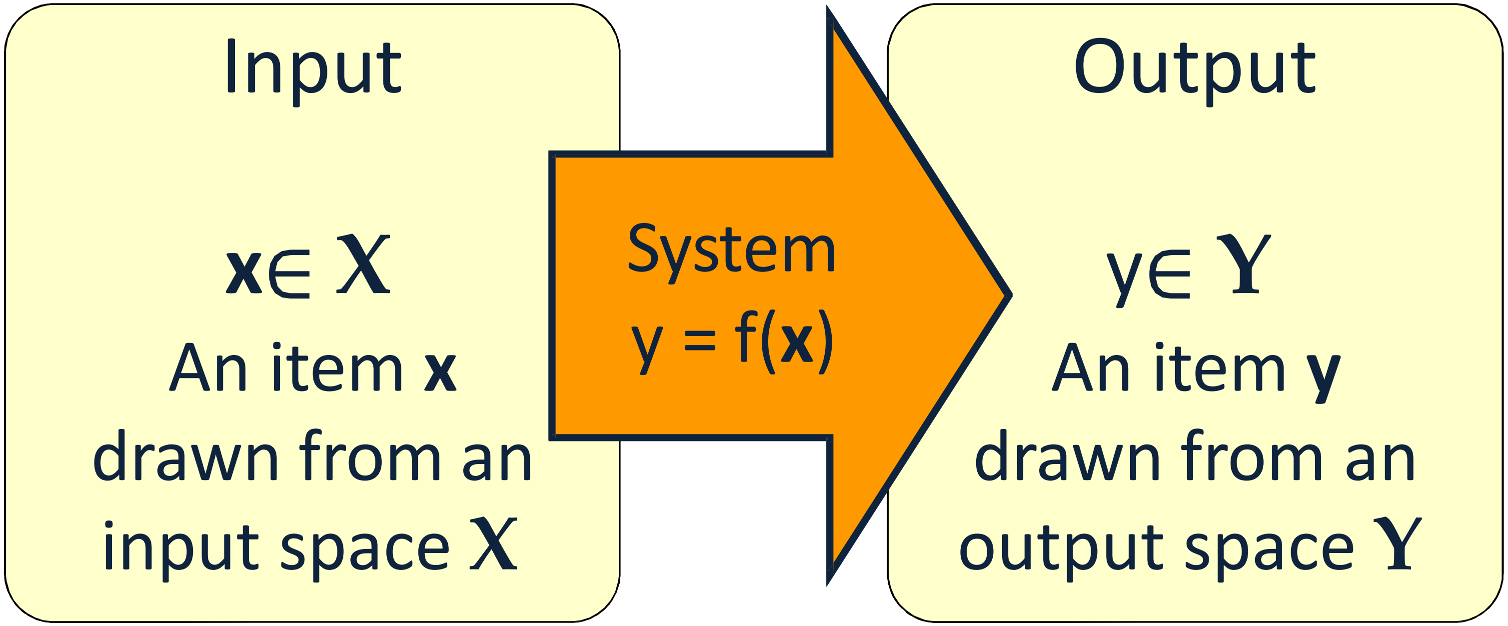
- The Badges Game... ...
  - This is an example of the key learning protocol: supervised learning
- First question: Are you sure you got it?
  - Why?
- Issues:
  - Which problem was easier?
  - Representation
  - Algorithm: can you write a program that takes this data as input and predicts the label for your name?

Supervised Learning



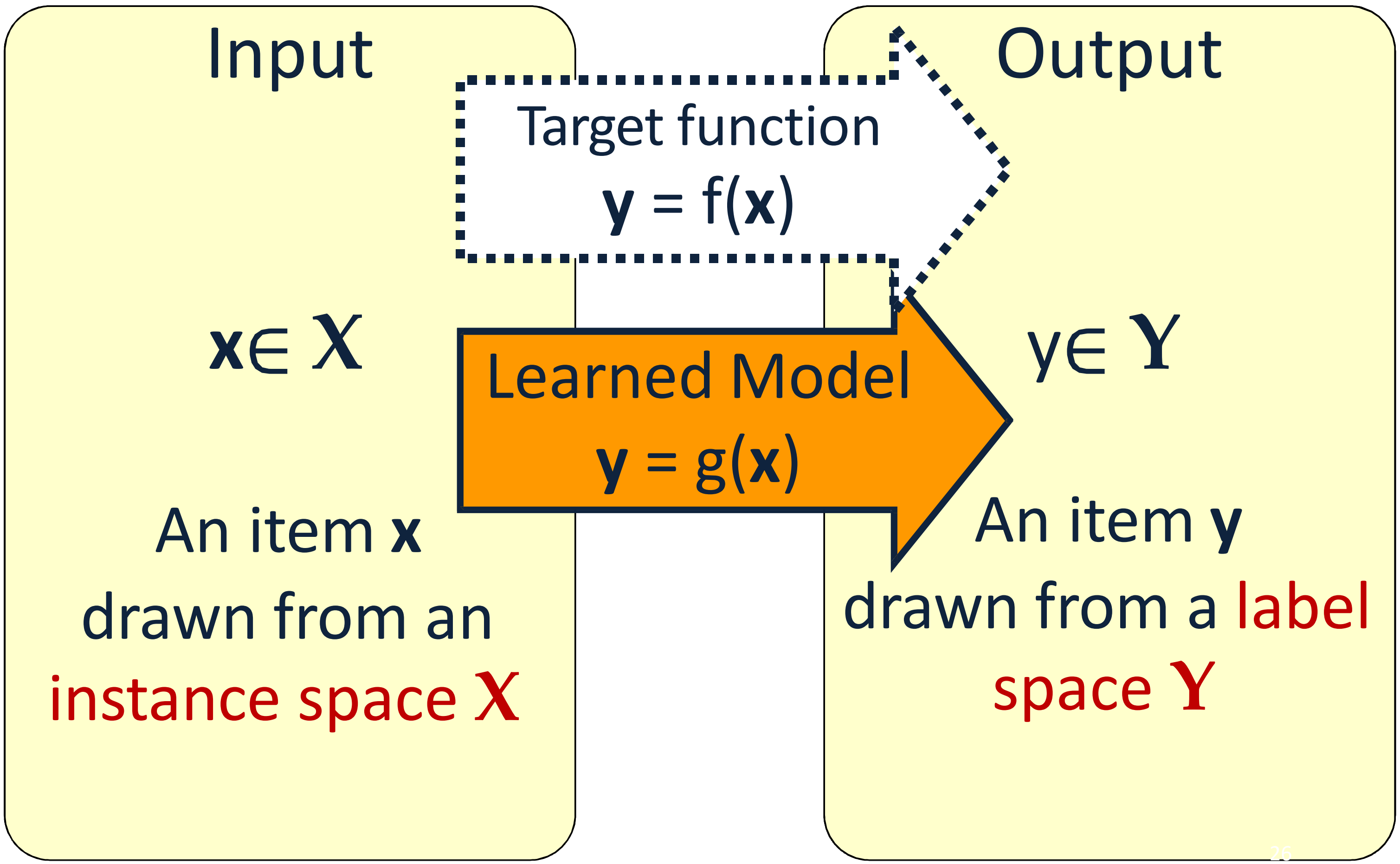
- We consider systems that apply a function f() to input items **x** and return an output **y** = f(**x**).

Supervised Learning



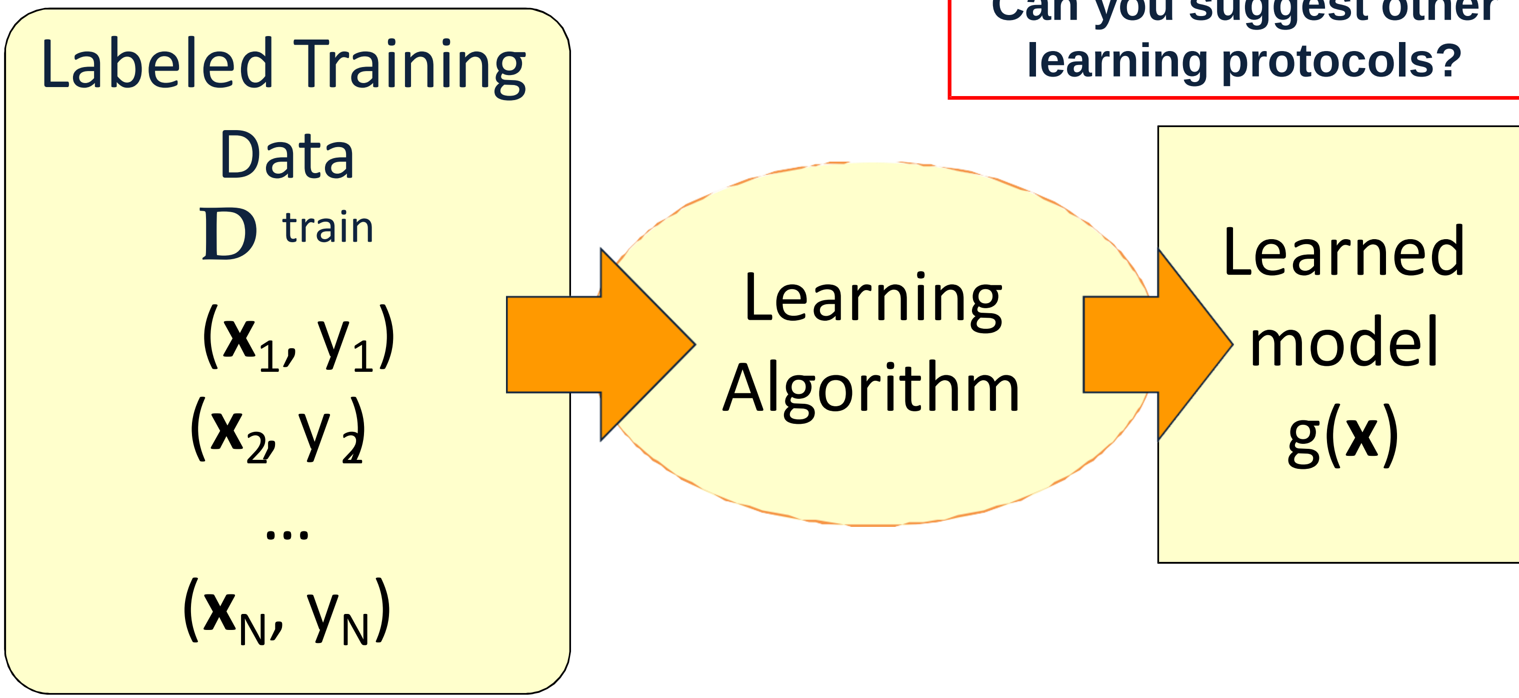
- In (supervised) machine learning, we deal with systems whose f(**x**) is learned from examples.

Supervised learning



Supervised learning: Training

Can you suggest other learning protocols?



- Give the learner examples in $\mathbf{D}^{\text{train}}$
- The learner returns a model $g(\mathbf{x})$

g(x) is the model we'll use in our application

Supervised learning: Testing

Labeled
Test Data
$\mathbf{D}^{\text{test}}$
$(\mathbf{x}'_1, y'_1)$
$(\mathbf{x}'_2, y'_2)$
…
$(\mathbf{x}'_M, y'_M)$

- Reserve some labeled data for testing

Supervised learning: Testing

Labeled Test Data

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

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

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

…

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

Raw Test Data

$\mathbf{X}^{\text{test}}$

$\mathbf{x}'_1$

$\mathbf{x}'_2$

….

$\mathbf{x}'_M$

Test Labels

$\mathbf{Y}^{\text{test}}$

y'_1

y'_2

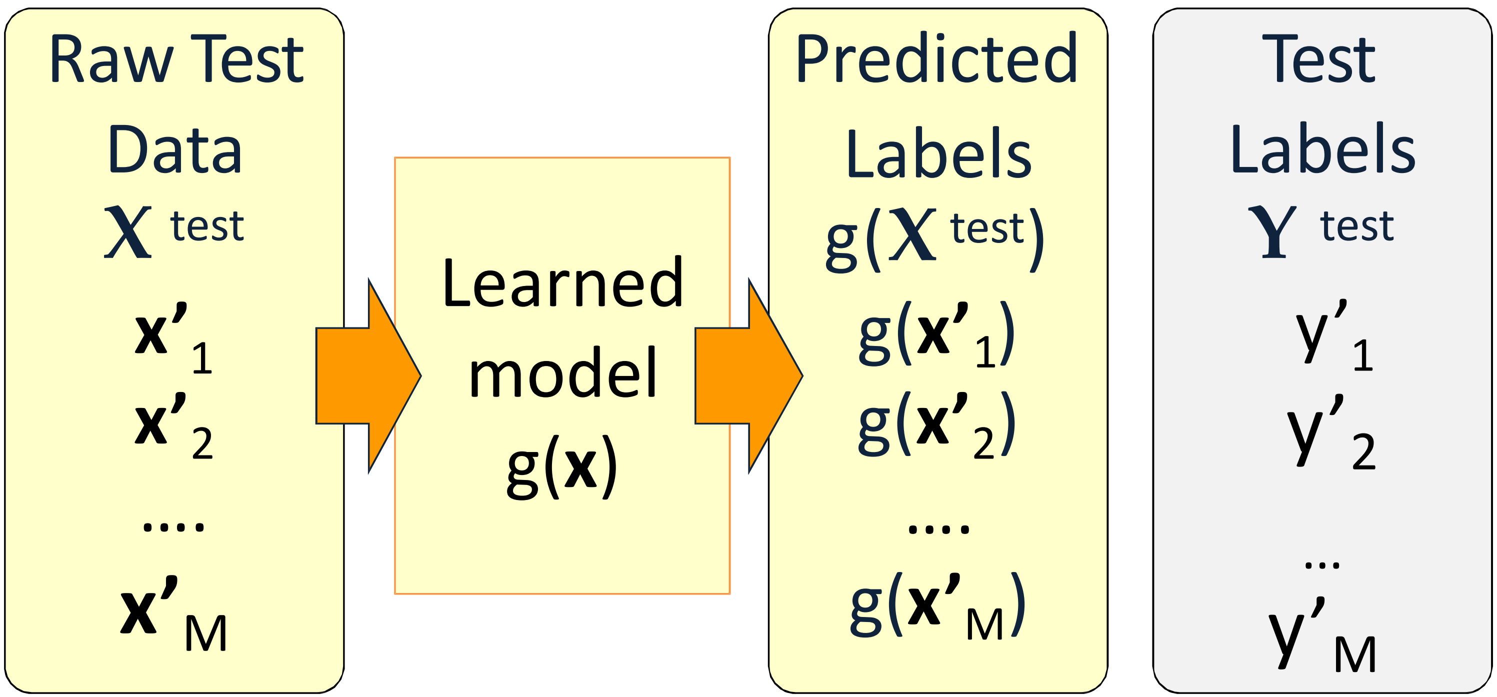
…

y'_M

Supervised learning: Testing

Can you use the test data otherwise?

- Apply the model to the raw test data
- Evaluate by comparing predicted labels against the test labels



Supervised Learning : Examples

- Disease diagnosis
  - x: Properties of patient (symptoms, lab tests)
  - f : Disease (or maybe: recommended therapy)
- Part-of-Speech tagging
  - x: An English sentence (e.g., The can will rust)
  - f : The part of speech of a word in the sentence
- Face recognition
  - x: Bitmap picture of person's face
  - f : Name the person (or maybe: a property of)
- Automatic Steering
  - x: Bitmap picture of road surface in front of car
  - f : Degrees to turn the steering wheel

Many problems that do not seem like classification problems can be decomposed to classification problems.

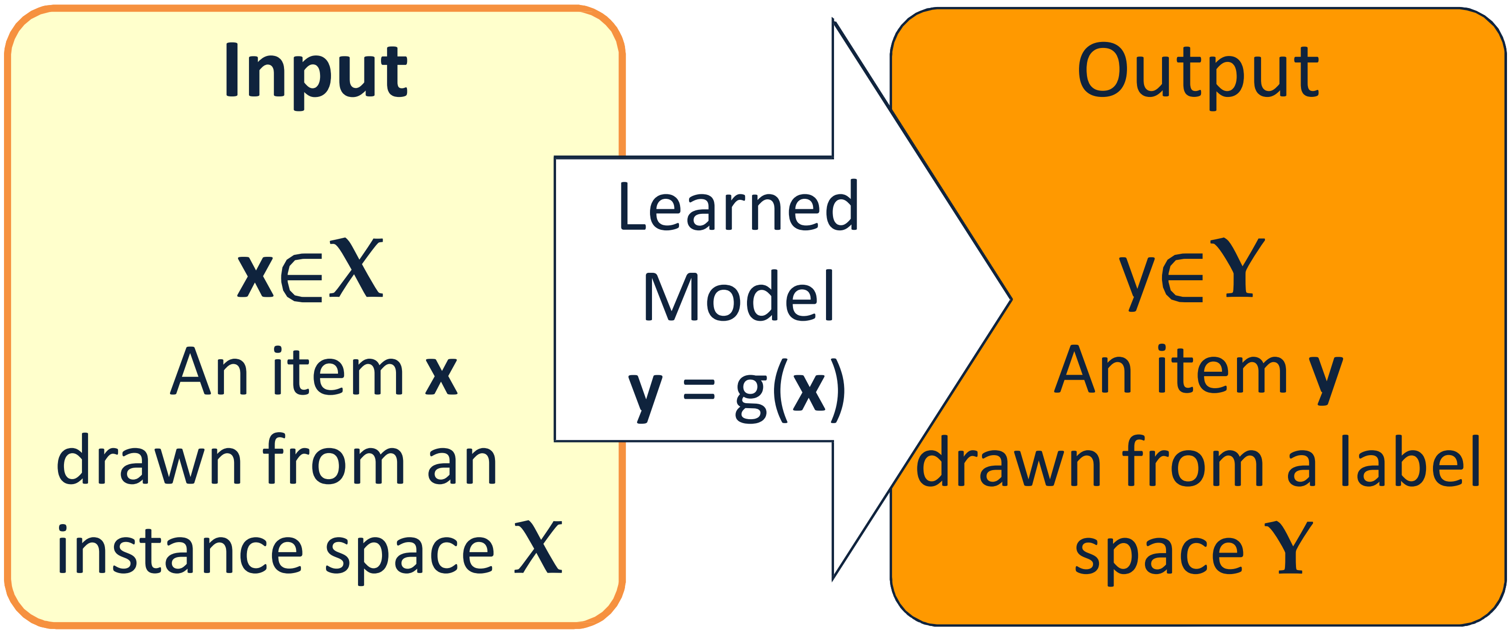
Key Issues in Machine Learning

- Modeling
  - How to formulate application problems as machine learning problems ? How to represent the data?
  - Learning Protocols (**where is the data & labels coming from**?)
- Representation
  - What **functions** should we learn (hypothesis spaces) ?
  - How to map raw **input** to an instance space?
  - Any rigorous way to find these? Any general approach?
- Algorithms
  - What are good algorithms?
  - How do we define success?
  - Generalization vs. over fitting
  - The computational problem

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 functions (models) 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? What drives learning?

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



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

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

- When we apply machine learning to a task, we first need to define the instance space $\mathbf{X}$.
- Instances $x \in \mathbf{X}$ are defined by features:
  - Boolean features:
    - Is there a folder named after the sender?
    - Does this email contains the word 'class'?
    - Does this email contains the word 'waiting'?
    - Does this email contains the word 'class' and the word 'waiting'?
  - Numerical features:
    - How often does 'learning' occur in this email?
    - What long is email?
    - How many emails have I seen from this sender over the last day/week/month?
  - Bag of tokens
    - Just list all the **tokens** in the input

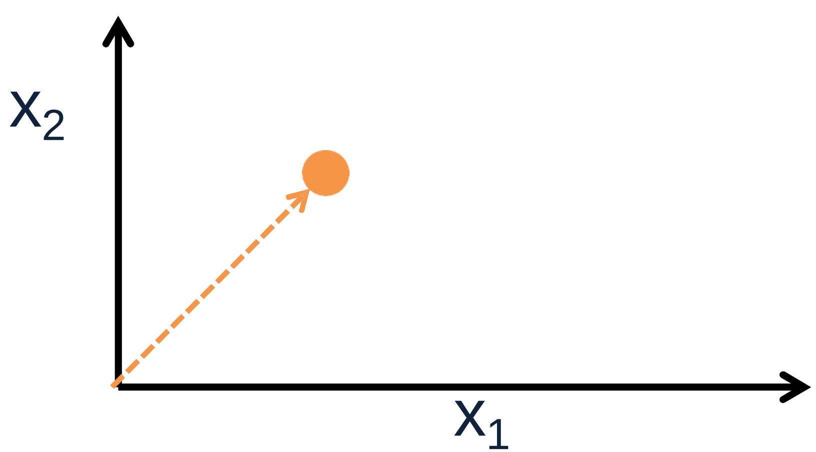
Does it add anything?

What’s 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?
  - Height;
  - Shoe size

$\mathbf{X}$ as a vector space

- $\mathbf{X}$ is an N-dimensional vector space (e.g. $<^{N}$)
  - 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 $\mathbf{X}$:



From feature templates to vectors

- When designing features, we often think in terms of templates, not individual features:
- **Encoding a name by encoding it characters:**
- **What is the *i*-th letter?**
- **Abe** → [**1** 0 0 0 0… 0 1 0 0 0 0… 0 0 0 0 **1** …]
  - 26*2 + 1 positions in each group;
  - # of groups == upper bounds on length of names

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.
- Think about the badges game – what if you were focusing on visual features?
- We can’t teach you what specific features to use for your task.
  - But we will touch on some general principles

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

Input

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

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

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

Output

$\mathbf{y} \in \mathbf{Y}$

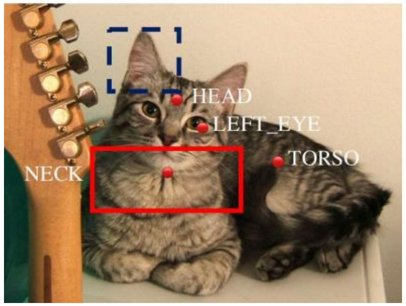
An item $\mathbf{y}$ drawn from a label space $\mathbf{Y}$

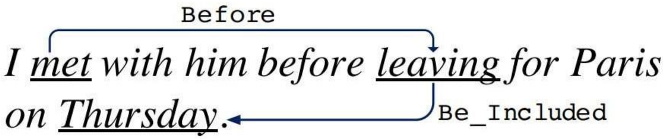
- The label space $\mathbf{Y}$ determines what *kind* of supervised learning task we are dealing with

Supervised learning tasks I

- Output labels $y \in Y$ are categorical:
  - Binary classification: Two possible labels
  - Multiclass classification: k possible labels

  - Output labels $y \in Y$ are structured objects (sequences of labels, parse trees, etc.)
  - Structure learning

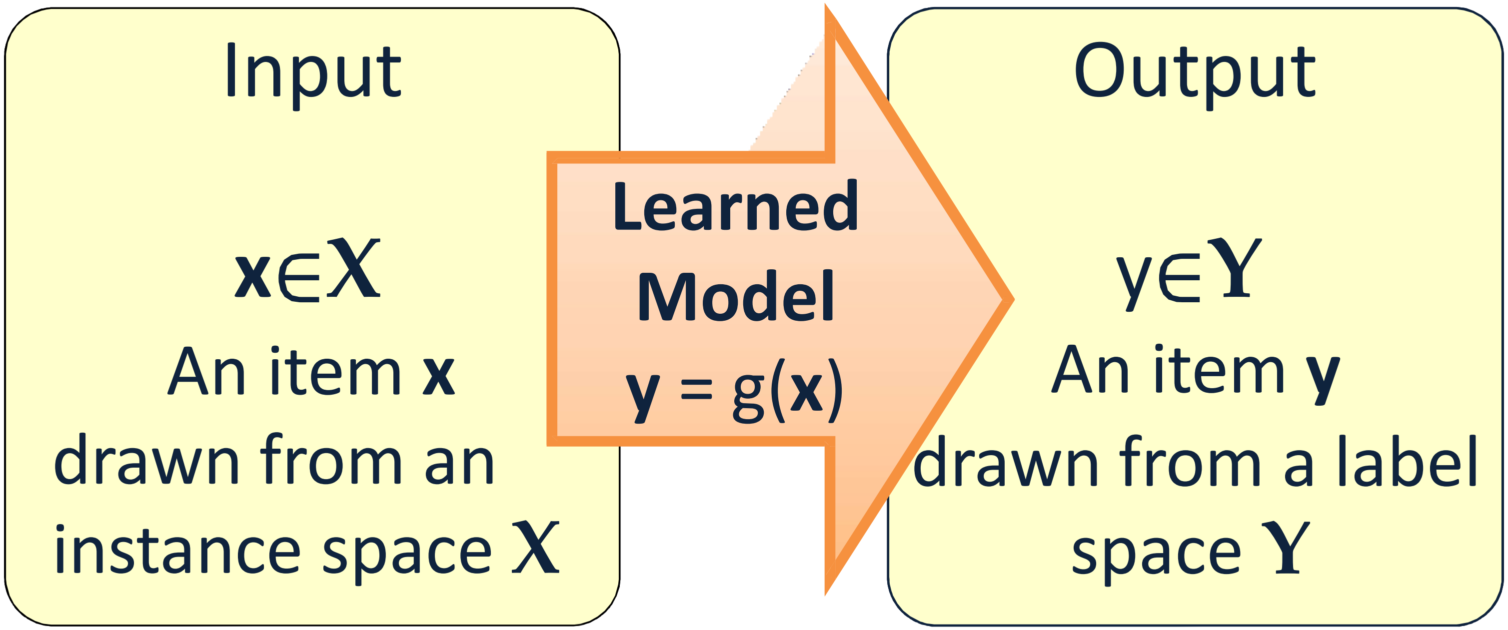




Supervised learning tasks II

- Output labels $y \in Y$ 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. The model g(**x**)



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

Hypothesis Space

Complete Ignorance:

There are $2^{16} = 65536$ possible functions over four input features.

We can't figure out which one is correct until we've seen every possible input-output pair.

After observing seven examples we still have 2^9 possibilities for f

Is Learning Possible?

Example	x_1	x_2	x_3	x_4	y
1	0	0	0	0	?
	1	0	1	1	?
	1	1	0	0	0
	1	1	0	1	?
	1	1	1	0	?
16	1	1	1	1	?

- There are $|Y|^{|X|}$ possible functions f(**x**) from the instance space **X** to the label space **Y.**
- Learners typically consider only a *subset* of the functions from **X** to **Y**, called the hypothesis space **H** . **H** $\subseteq |Y|^{|X|}$

Terminology

- Target function (concept): The true function f :X →{…Labels…}
- Concept: Boolean function. Example for which f (x)= 1 are positive examples; those for which f (x)= 0 are negative examples (instances)

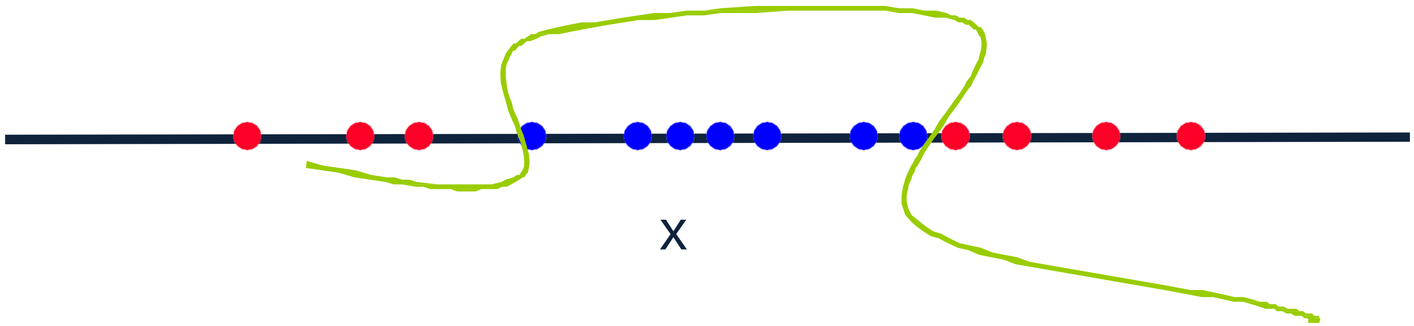
- Hypothesis: A proposed function h, believed to be similar to f. The output of our learning algorithm.
- Hypothesis space: The space of all hypotheses that can, in principle, be the output of the learning algorithm.

- Classifier: A discrete valued function produced by the learning algorithm. The possible value of f: {1,2,…K} are the classes or class labels. (In most algorithms the classifier will actually return a real valued function that we'll have to interpret).

- Training examples: A set of examples of the form {(x, f (x))}

Functions Can be Made Linear

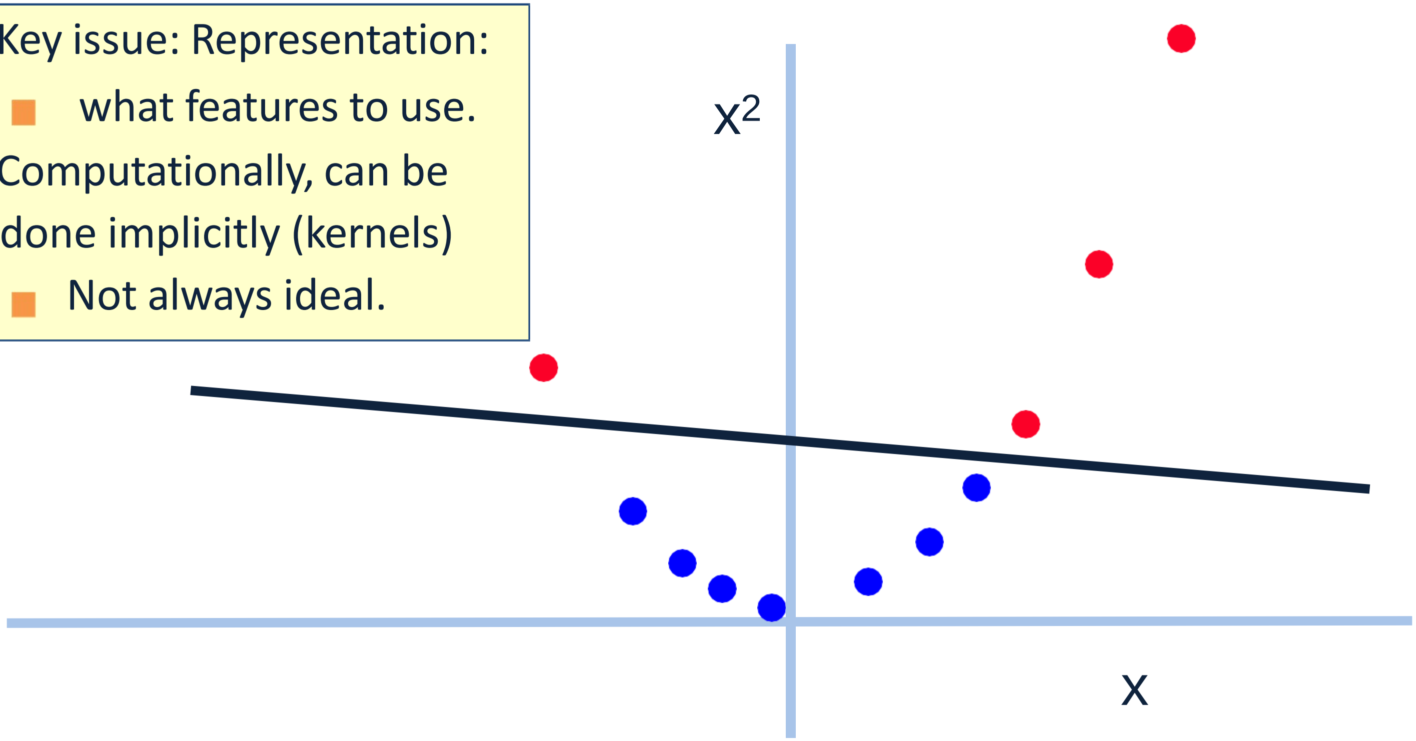
- Data points are not linearly separable in one dimension
- Not separable if you insist on using a specific class of functions (e.g., linear)



Blown Up Feature Space

- Data are separable in $<x, x^2>$ space

- Key issue: Representation:
  - what features to use.
- Computationally, can be done implicitly (kernels)
  - Not always ideal.



Exclusive-OR (XOR)

- $(x_1 \wedge x_2) \vee (\neg\{x_1\} \wedge \neg\{x_2\})$
- In general: a parity function.

- $x_i \in \{0,1\}$
- $f(x_1, x_2, \ldots, x_n) = 1$
  iff $\sum x_i$ is even

This function is not
  linearly separable.