What is Machine Learning?

A few useful viewpoints about ML

"Learning is any process by which a system improves performance from experience."

Herbert Simon (1916-2001)

Tom Mitchell (1951-)

Machine Learning is the study of algorithms that

- improve their performance P
- at some task T
- with experience, or "training data" $\mathcal{D}$

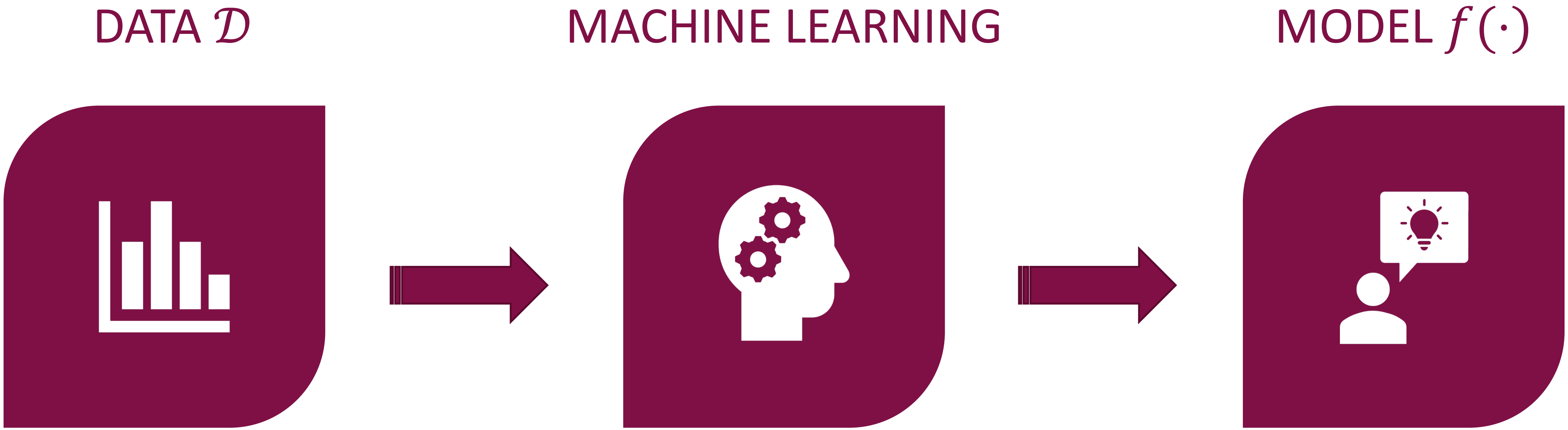
A well-defined learning task is given by $(P, T, \mathcal{D})$

Examples:

T: Classify emails as legitimate or spam
P: Percentage of emails labeled correctly
$\mathcal{D}$: Repository of emails, some with human-specified labels

T: Playing Chess (or Go)
P: Percent games won against an opponent
$\mathcal{D}$: Playing games against itself

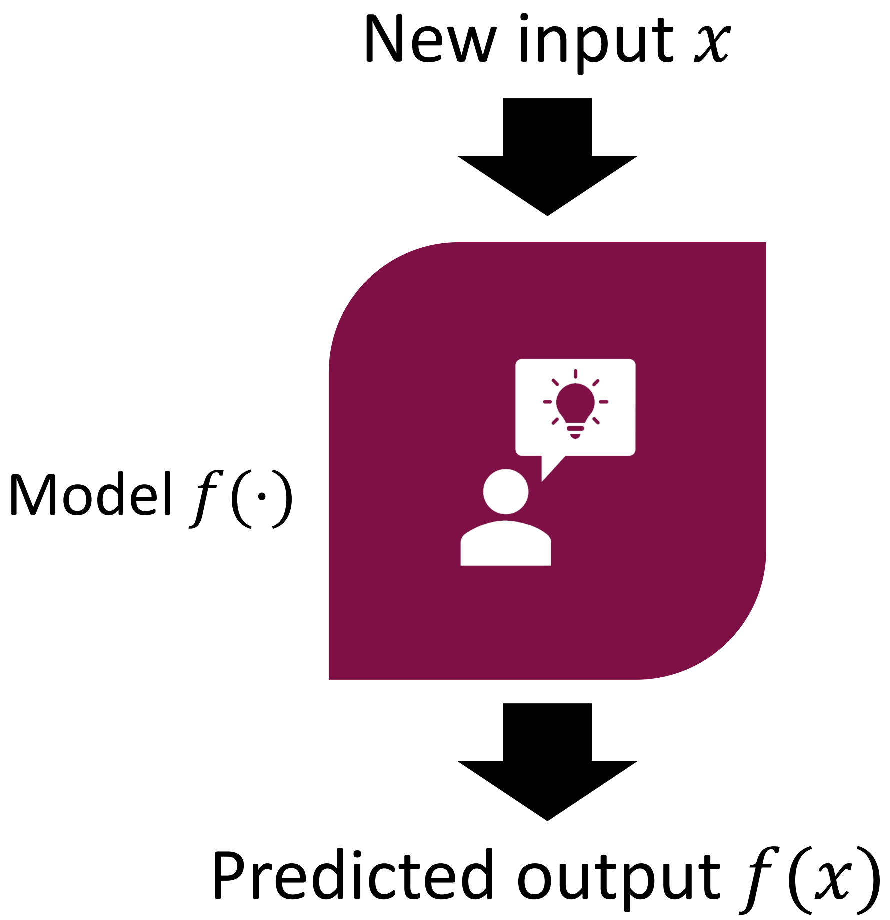
Data In, Model Out



ML automates the generation of “models” from data

Examples of “models”: mathematical models like Newton’s laws of motion that predict how objects will move, conceptual models like a flowchart specifying how to treat a patient, etc.

Machine Learning for Prediction



ML automates the generation of "models" from data

Examples of "models": mathematical models like Newton's laws of motion that predict how objects will move, conceptual models like a flowchart specifying how to treat a patient, etc.

Example: Rediscovering Newton's 2^{nd} Law

How can we predict the acceleration a of an object when we push it?

Framing this as an ML problem:

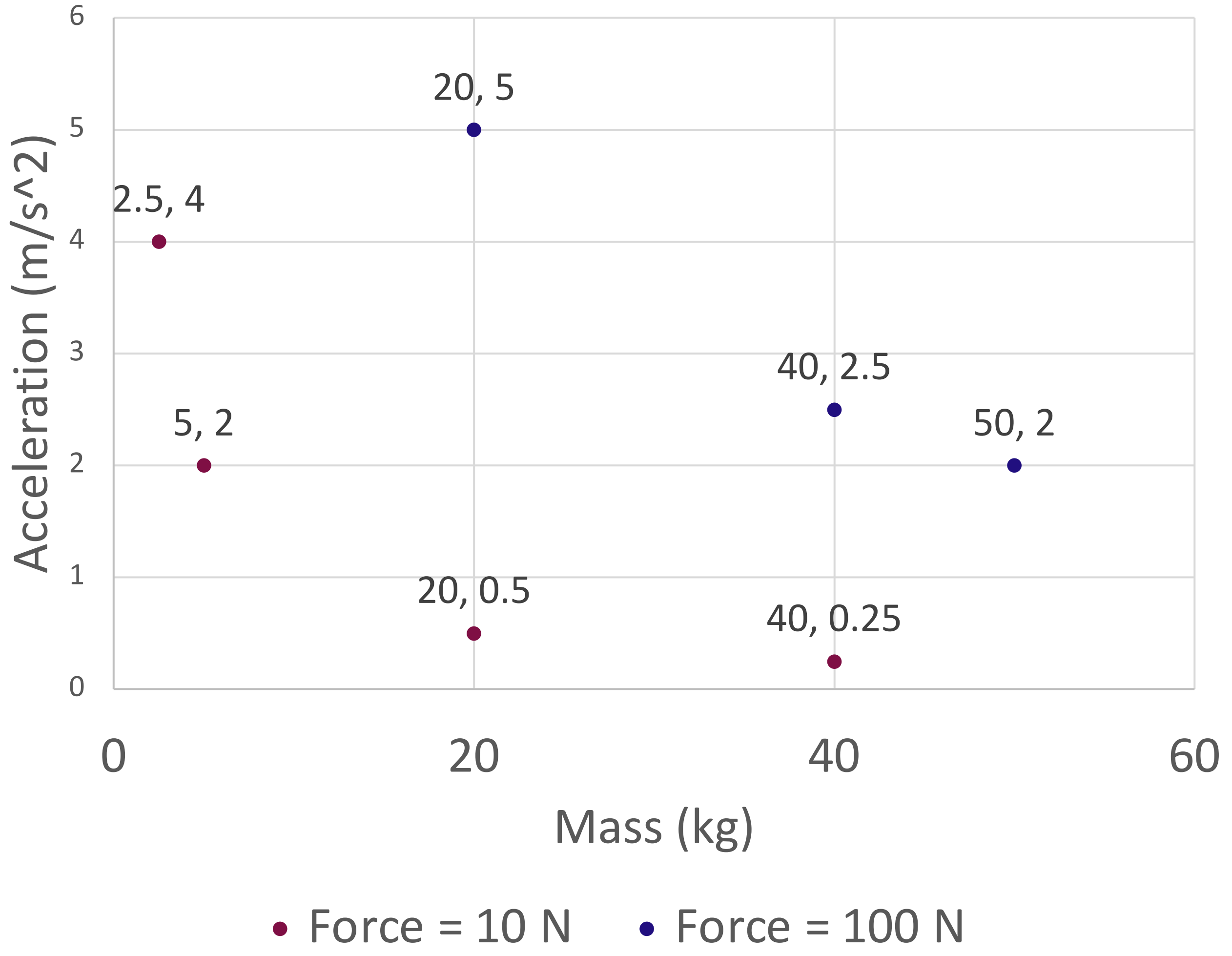
Q: How do we know to record these "features"? Why no others?

Task (T)	Predict acceleration a given pushing force F, mass m
Performance Measure (P)	Error in predicted a
Experience / Data ($\mathcal{D}$)	Perform some object pushing experiments!

Data In: $\mathcal{D}$

Push a few objects with known masses at two force values 10 N and 100 N.

Force	Mass	Acceleration
10	2.5	4
10	5	2
10	20	0.5
10	40	0.25
100	40	2.5
100	20	5
100	50	2



Data In, Model Out

Data In, Model Out

DATA $\mathcal{D}$

Force	Mass	Acceleration
10	2.5	4
10	5	2
10	20	0.5
10	40	0.25
100	40	2.5
100	20	5
100	50	2

MACHINE LEARNING

MODEL $f(\cdot)$

Data In, Model Out

DATA $\mathcal{D}$

Force	Mass	Acceleration
10	2.5	4
10	5	2
10	20	0.5
10	40	0.25
100	40	2.5
100	20	5
100	50	2

MACHINE LEARNING

MODEL $f(\cdot)$

$$\text{acceleration } a = \frac{\text{Force } F}{\text{mass } m}$$

We would like to recover a model like this!

Data In, Model Out

DATA $\mathcal{D}$ MACHINE LEARNING MODEL $f(\cdot)$

Force	Mass	Acceleration
10	2.5	4
10	5	2
10	20	0.5
10	40	0.25
100	40	2.5
100	20	5
100	50	2

$$\text{acceleration } a = \frac{\text{Force } F}{\text{mass } m}$$

ML Design (hypothesis class, loss function, optimizer, hyperparameters, features, …)

Example hypothesis class:
(for varying values of $w_0, \dots w_4$)

$$a = w_0 + w_1 F + w_2 m + w_3 (F * m) + w_4 (\frac{F}{m})$$

Learning = finding “good” values for the *weights* $w_0, w_1, \dots, w_4$

$$a = 0 + 0F + 0m + 0(F * m) + 1(\frac{F}{m})$$

ML Design Choices

The class of functions from which the ML procedure must pick one to fit the data

What it means to fit the data: a function that is high for bad fits, low for good fits

How to search for the function that best fits the data

ML Design (hypothesis class, loss function, optimizer, hyperparameters, features, …)

Natural laws are concise descriptions of empirical observations.

Newton didn't quite discover his 2nd law with ML, but Johannes Kepler played data scientist with 30 years of Tycho Brahe's astronomical observations to discover his celebrated laws of planetary motion, which later led to Newton's laws!

You will play with some of Tycho's data in HW0!

[Data-Driven Discovery of Physical Laws - Langley - 1981 - Cognitive Science - Wiley Online Library]

The Machine Learning Workflow

ML Workflow

ML Design (hypothesis class, loss function, optimizer, hyperparameters, features)

ML Workflow

- ML Design (hypothesis class, loss function, optimizer, hyperparameters, features)
- Train model

ML Workflow

- ML Design (hypothesis class, loss function, optimizer, hyperparameters, features)
- Train model
- Validate / Evaluate

Main focus of this class

ML Workflow

- Framing an ML problem (Mitchell's P, T, D)
- ML Design (hypothesis class, loss function, optimizer, hyperparameters, features)
- Train model
- Validate / Evaluate

Main focus of this class

ML Workflow

- Framing an ML problem (Mitchell's P, T, D)
- Data curation (sourcing, scraping, collection, labeling)
- Data analysis / visualization
- ML Design (hypothesis class, loss function, optimizer, hyperparameters, features)
- Train model
- Validate / Evaluate

Project

Main focus of this class

ML Workflow

- Framing an ML problem (Mitchell’s P, T, D)
- Data curation (sourcing, scraping, collection, labeling)
- Data analysis / visualization
- ML Design (hypothesis class, loss function, optimizer, hyperparameters, features)
- Train model
- Validate / Evaluate
- Deploy (and generate new data)
- Monitor performance on new data

Project

Main focus of this class

Learning as Compression

Learning is “Compression”

DATA $\mathcal{D}$

MACHINE LEARNING

MODEL $f(\cdot)$

Force	Mass	Acceleration
10	2.5	4
10	5	2
10	20	0.5
10	40	0.25
100	40	2.5
100	20	5
100	50	2

$$\text{acceleration } a = \frac{\text{Force } F}{\text{mass } m}$$

$$a = 0 + 0F + 0m + 0(F * m) + 1(\frac{F}{m})$$

(This is one of the functions in the hypothesis class!)

Dataset size?

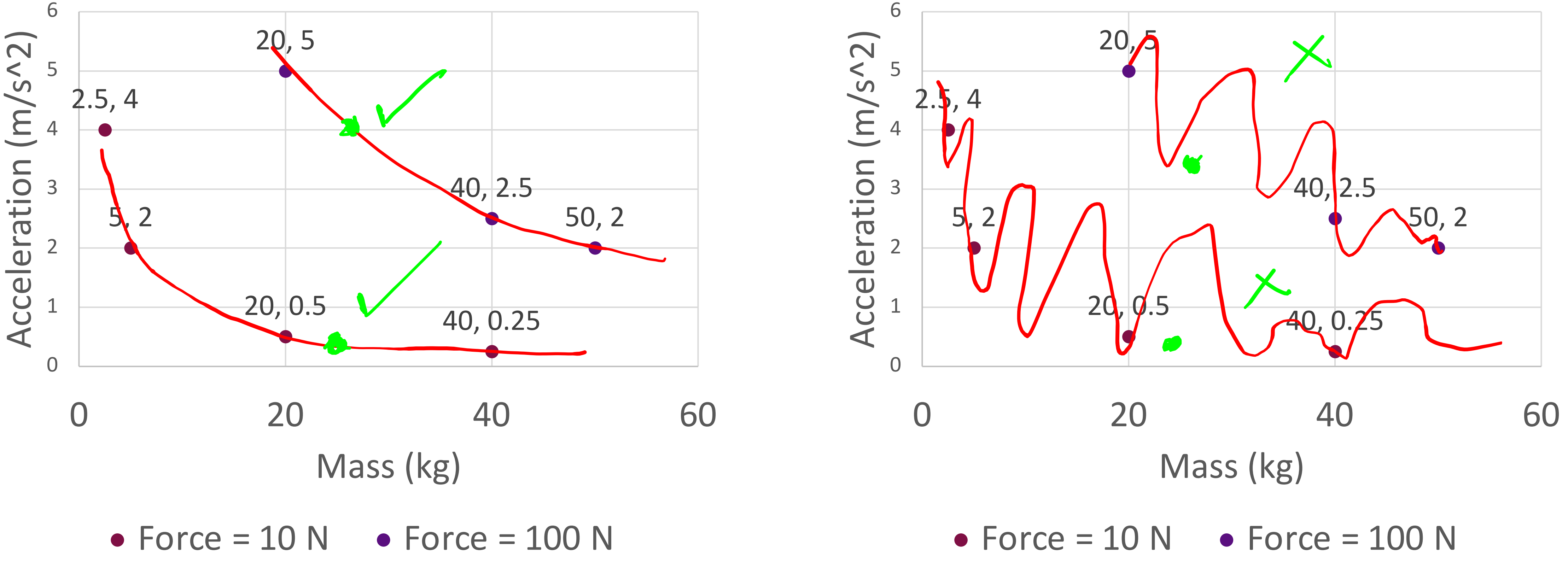
(N=7 data samples) * (D=2 features + 1 label) = 21 floats

“Model” / learned function size?

# of learned parameters = 5 floats

(Note: N, D are standard notation for num samples and num features)

What If We Use More Model Parameters?



Q: How do we know this is a bad result?

A: Collect some more data!

ML Workflow

- ML Design (hypothesis class, loss function, optimizer, hyperparameters, features)
- Train model
- Validate / Evaluate

(Always on held-out data)

Main focus of this class

Learning as Programming by Examples

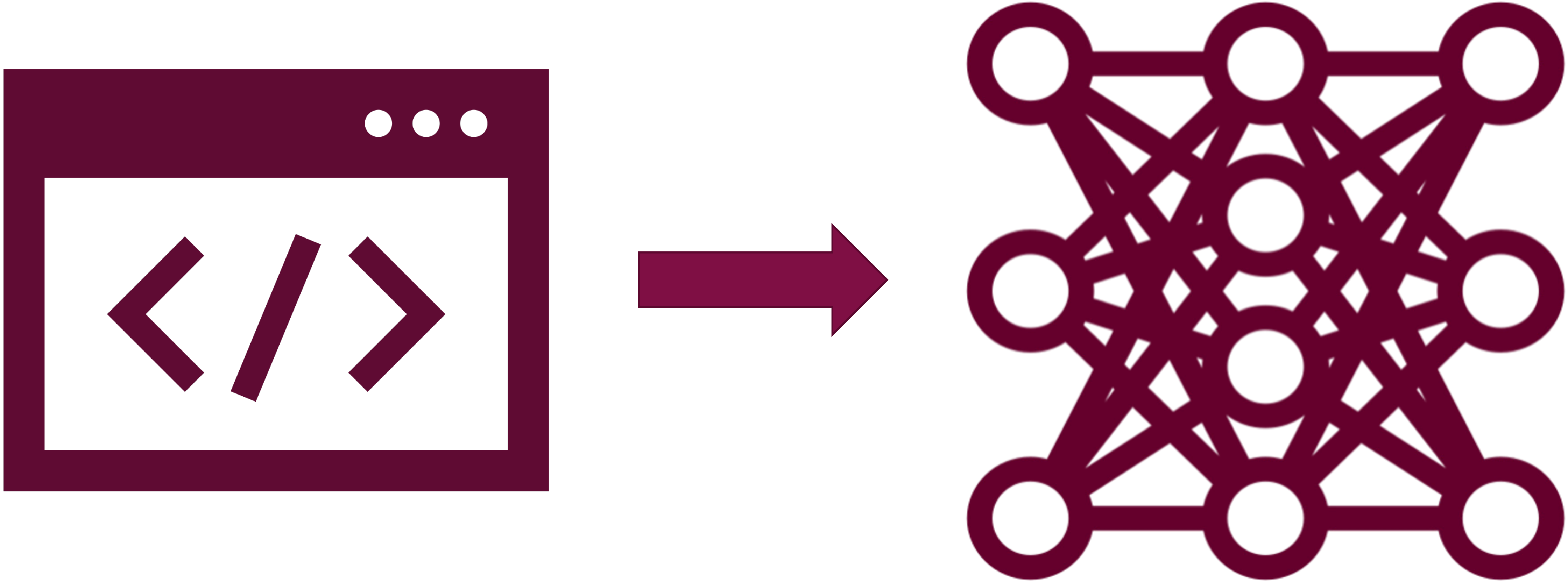
“Machine learning … gives computers the ability to learn without being explicitly programmed.”

Arthur Samuel

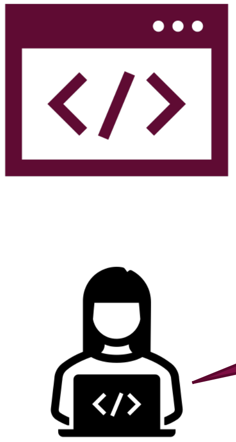
Machine learning is Programming 2.0

Traditional Programming

Machine learning (ML)

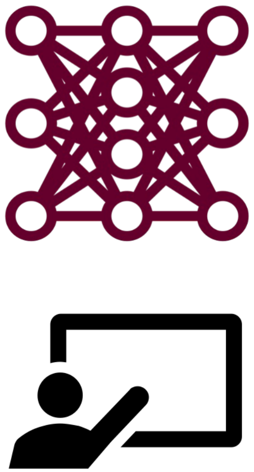


Task specification in ML: programs → examples



Here is a program to implement Newton's second law of motion

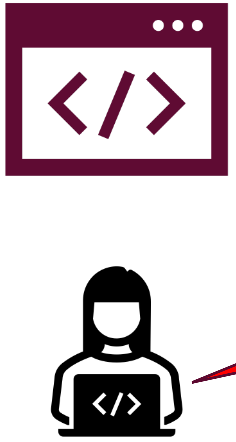
```
def compute_force(m, a):
      '''
      returns force (in N) needed to
      move mass m (in kg) at
      acceleration a (in m/s^2)
      '''

      F = m * a

      return F
```



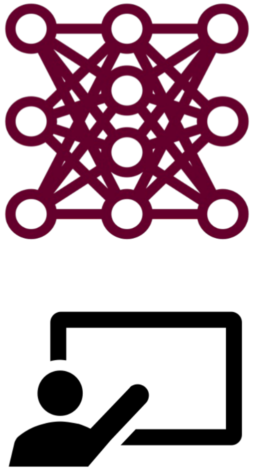
Here are some examples. Try to imitate them.

Mass m (kg)	Acceleration a (m/s^2)	Force F (N)
2.5	4	10
5	2	10
20	0.5	10
40	0.25	10
40	2.5	100
20	5	100
50	2	100

Task specification in ML: programs → examples



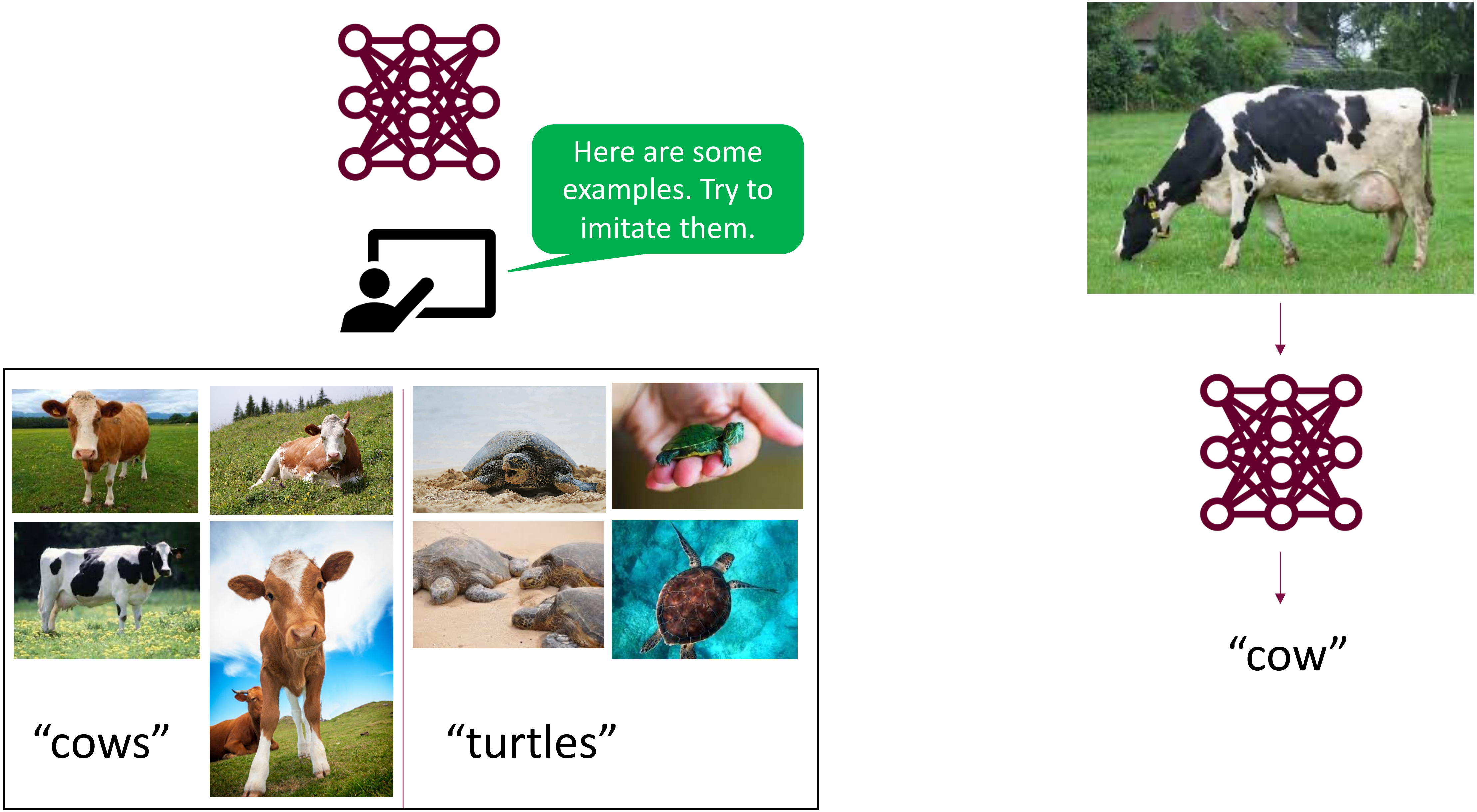
Here is a program to recognize an image as a cow or a turtle



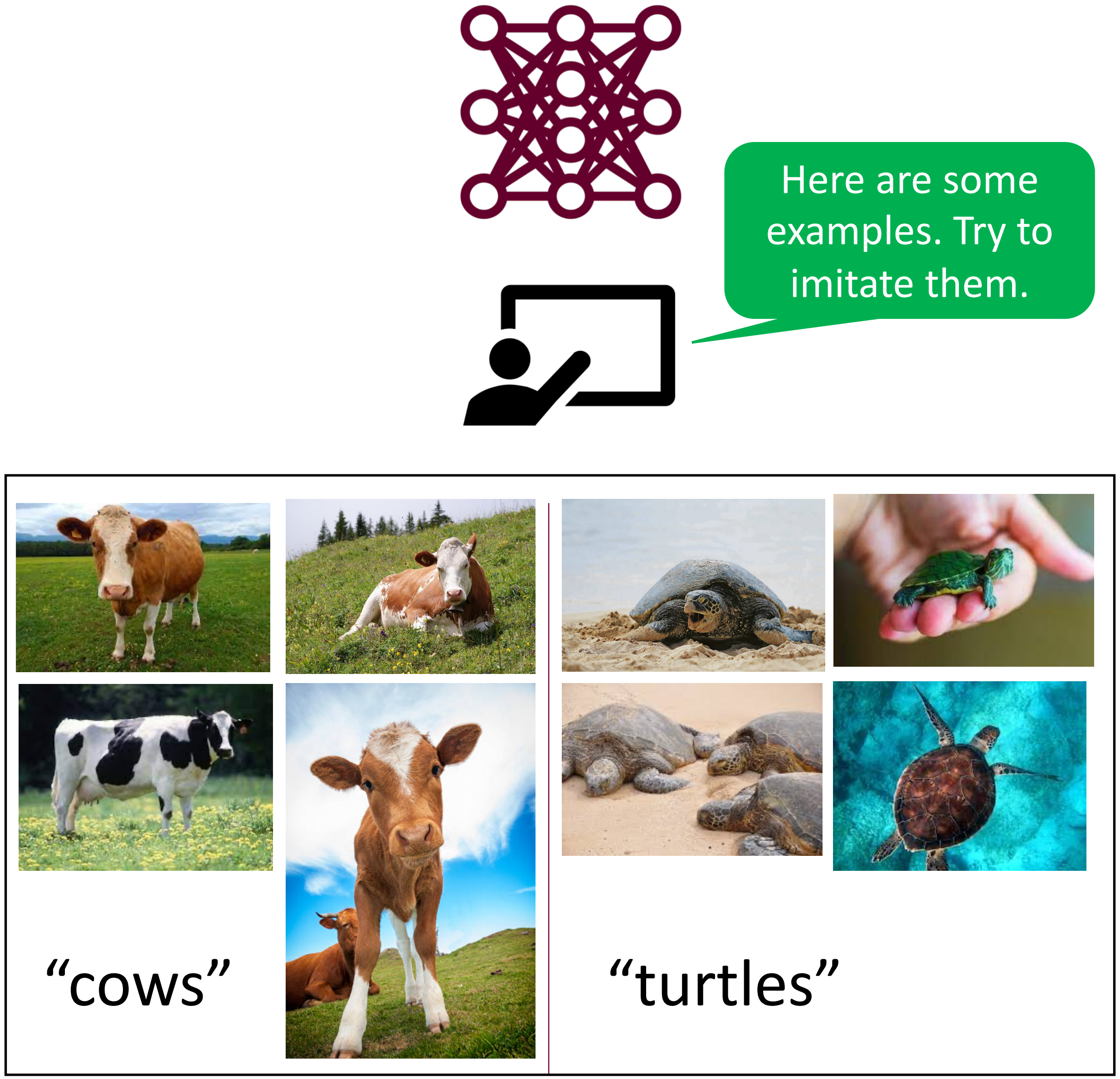
Here are some examples. Try to imitate them.

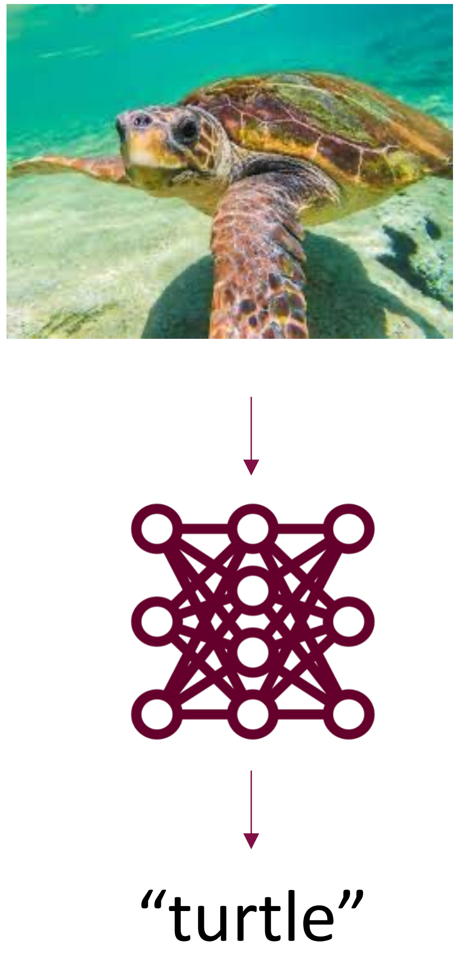
```
def cow_or_turtle(image):

        ???
```

Putting a trained ML system to use



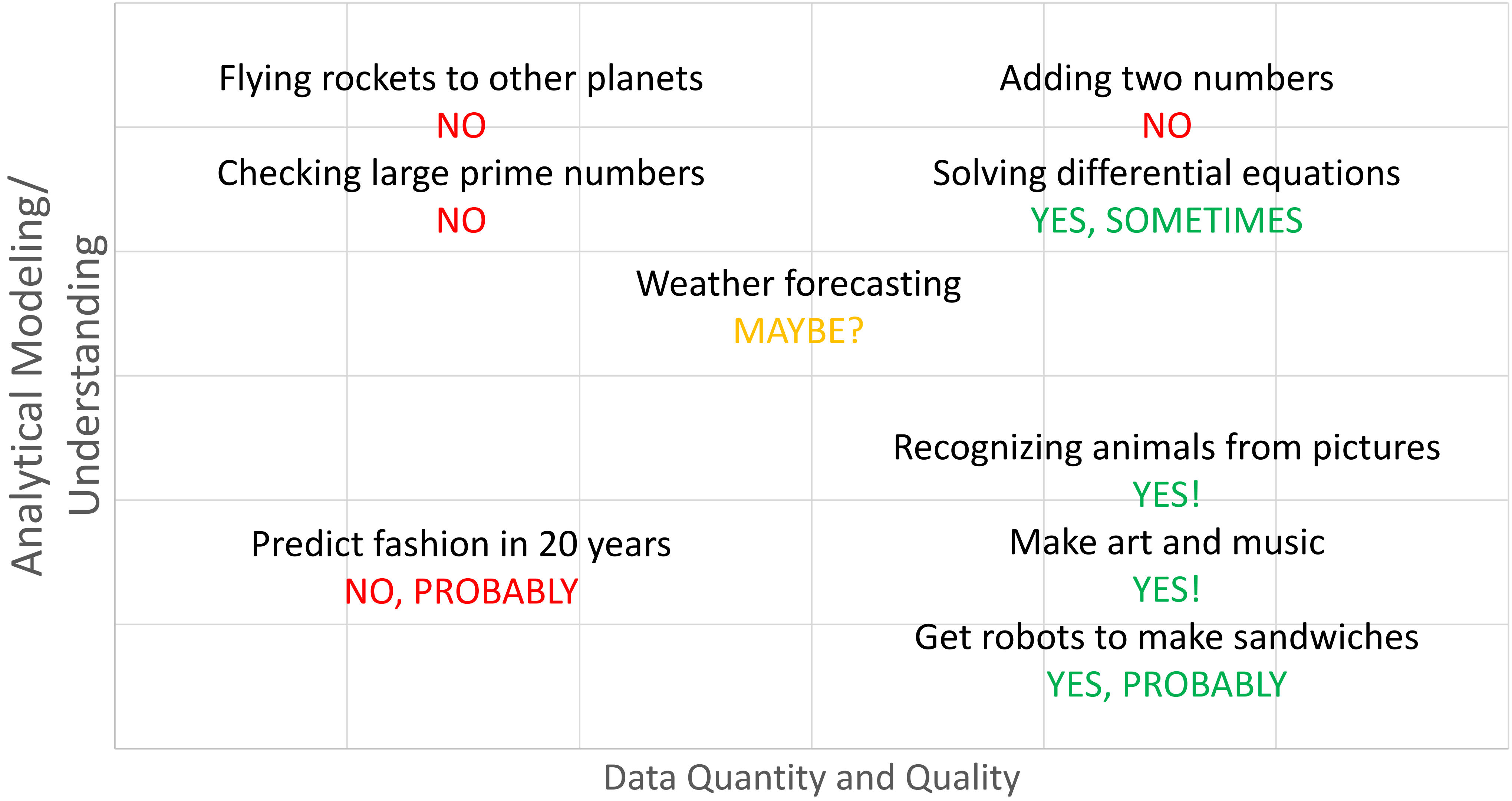
Putting a trained ML system to use





When should we use machine learning …?

… over traditional programming?



Summarizing

- Various conceptual views of machine learning:
  - Systems that improve with experience
  - Generating models from data
  - Programs → examples to specify a task to a computer
  - Compressing data
  - …

- All of these are correct, and it helps to think about ML in all these ways to build useful intuitions for how it works!

Types of Learning Problems

Types of Learning

- **Supervised learning**
  - **Input:** Examples of inputs and desired outputs
  - **Output:** Model that predicts output given a new input

- **Unsupervised learning**
  - **Input:** Examples of some data (no "outputs")
  - **Output:** Representation of structure in the data

- **Reinforcement learning**
  - **Input:** Sequence of agent interactions with an environment
  - **Output:** Policy that maps agent's observations to actions

Supervised Learning: Regression

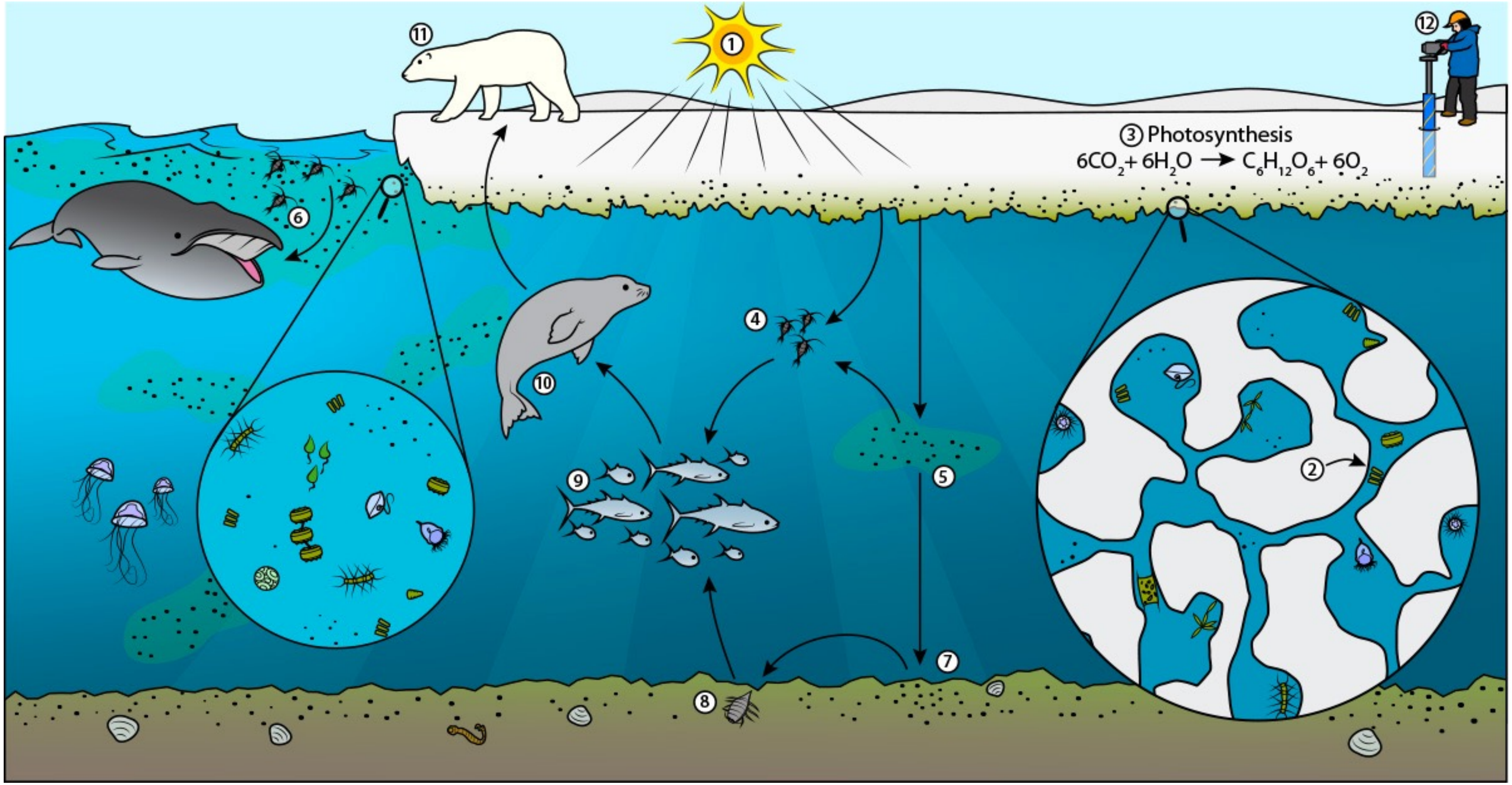


Image: https://askabiologist.asu.edu/explore/frozen-life

Supervised Learning: Regression

- Given $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
- Learn a function $f(x)$ to predict y given x
  - y is numeric == regression

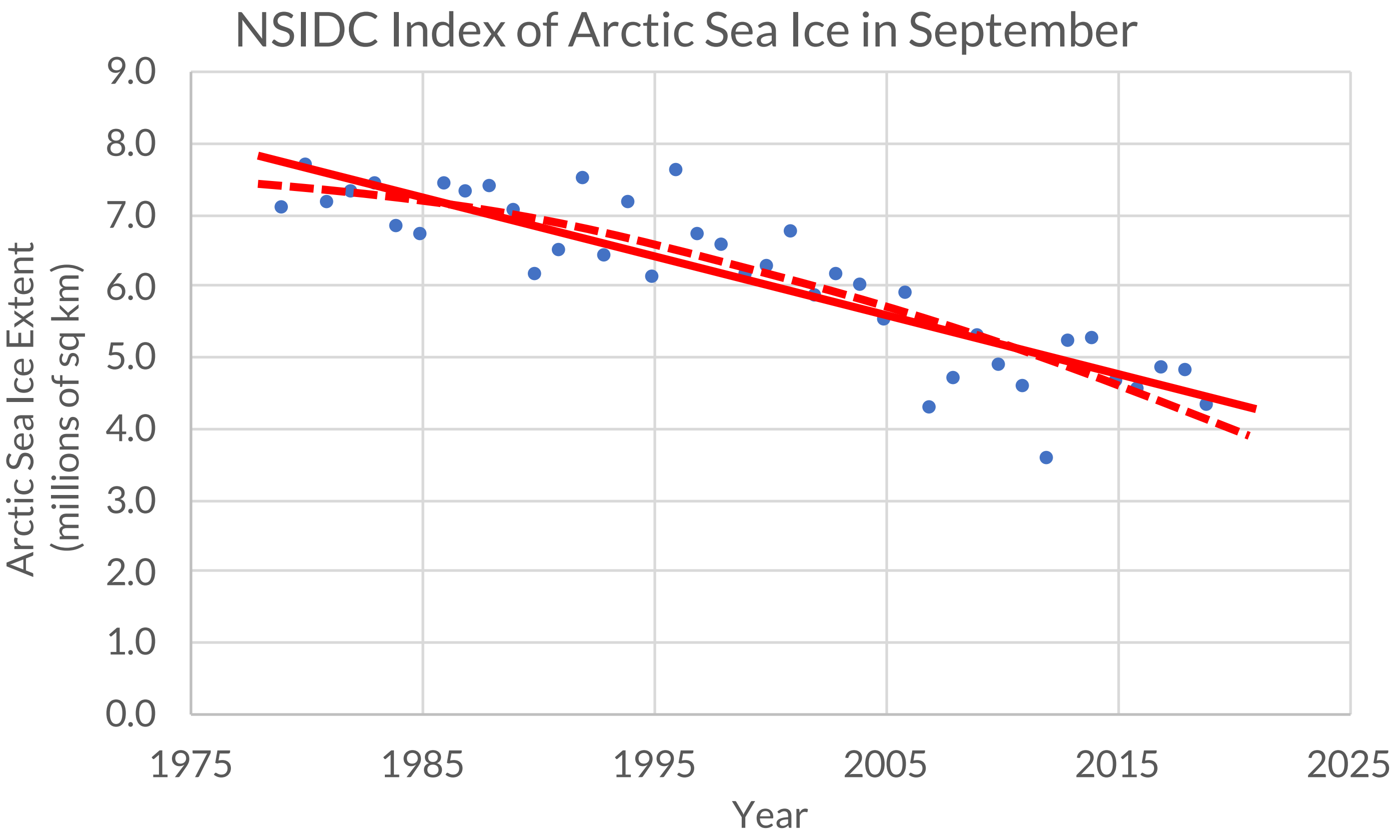
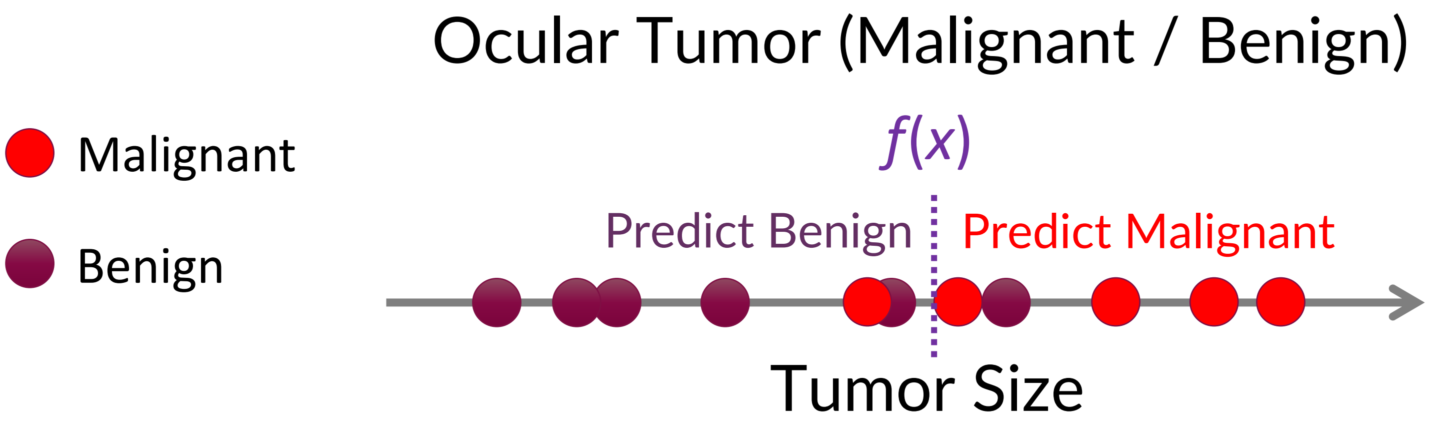


Image: https://www.flickr.com/photos/gsfc/5937599688/
Data from https://nsidc.org/arcticseaicenews/sea-ice-tools/

Supervised Learning: Classification

- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict y given x
  - y is categorical == classification



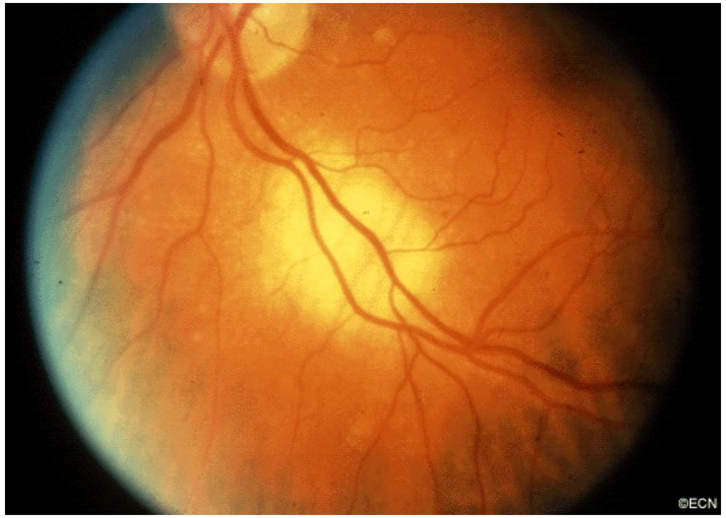
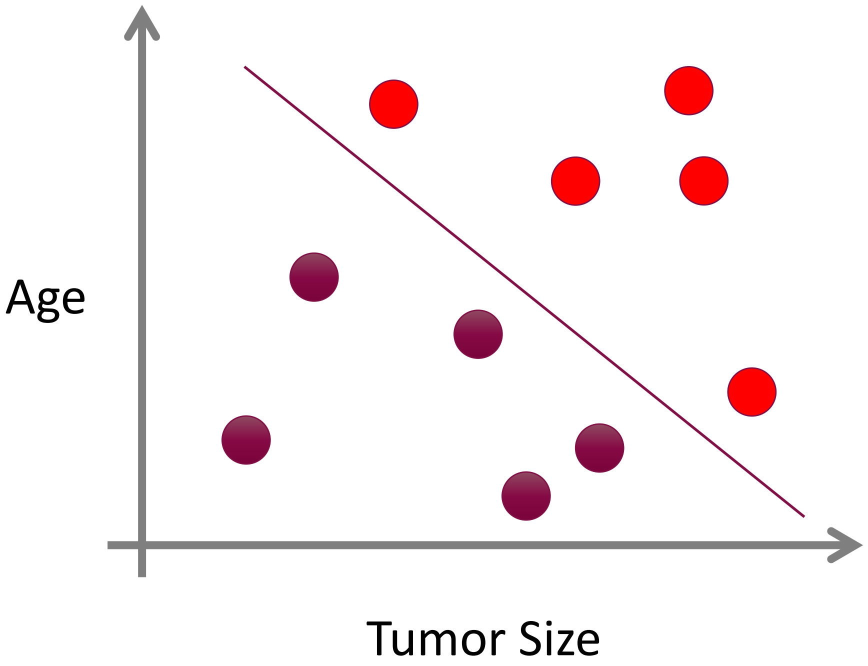


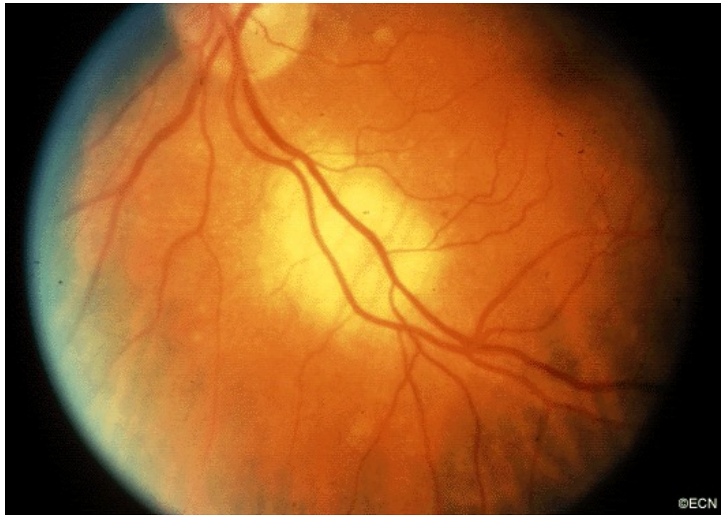
Image: https://eyecancer.com/uncategorized/choroidal-metastasis-test/

Supervised Learning

- *x* can be multi-dimensional
  - Each dimension corresponds to an attribute:



- Patient age
- Clump thickness
- Tumor Color
- Distance from optic nerve
- Cell type
- …

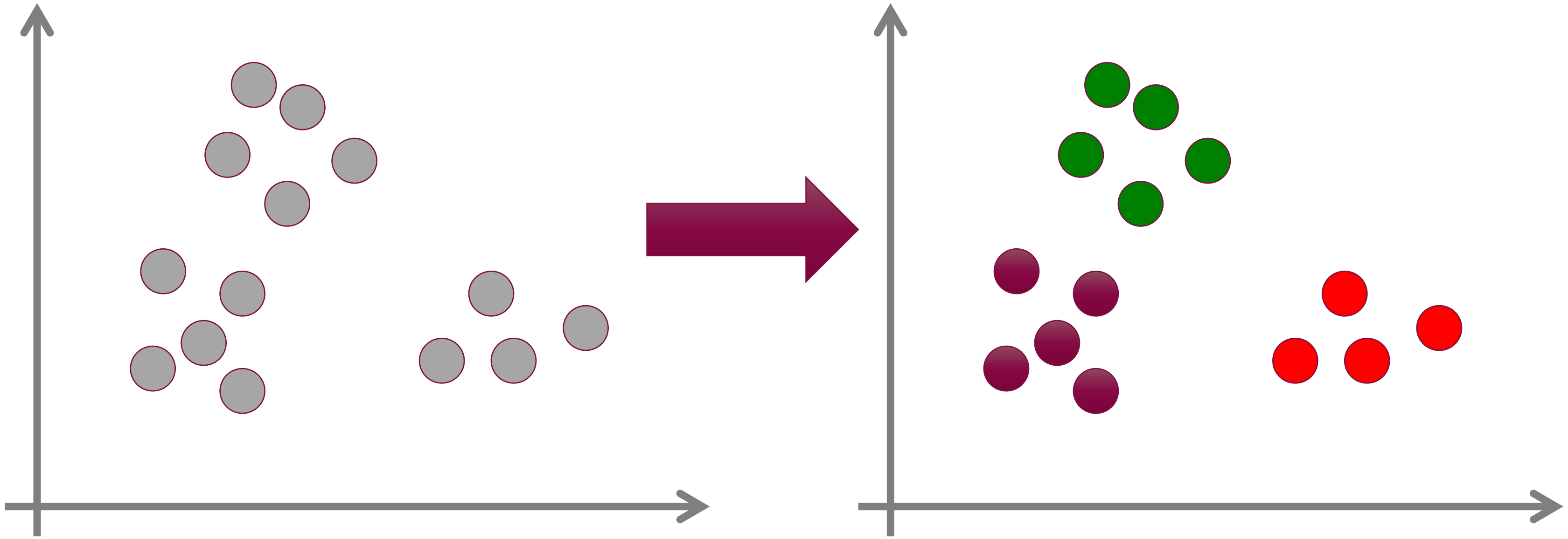


Cell type is the most telling feature, but it's risky to do a biopsy of the eye
- ML can help determine *when* a feature is needed

Image: https://eyecancer.com/uncategorized/choroidal-metastasis-test/

Unsupervised Learning

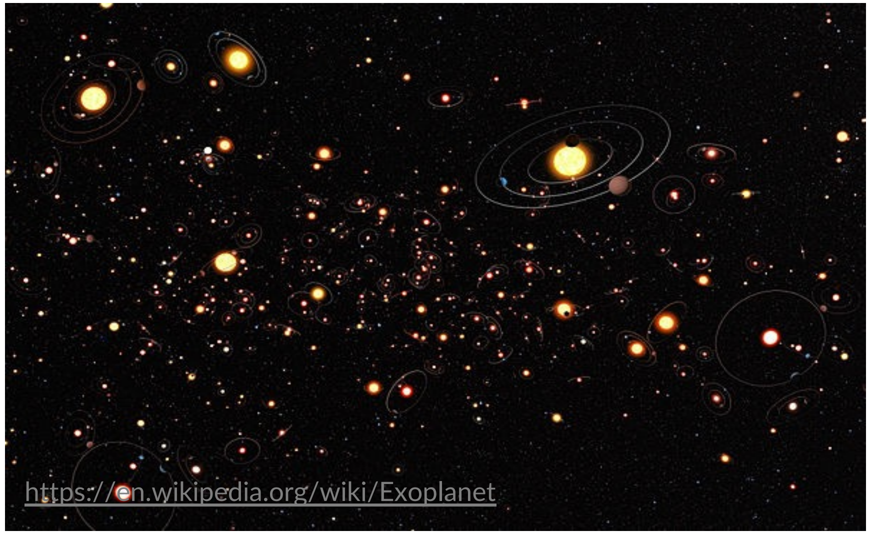
- Given $x_1, x_2, \ldots, x_n$ (without labels)
- Output hidden structure behind the x's … connected to "learning as compression"
  - E.g., clustering



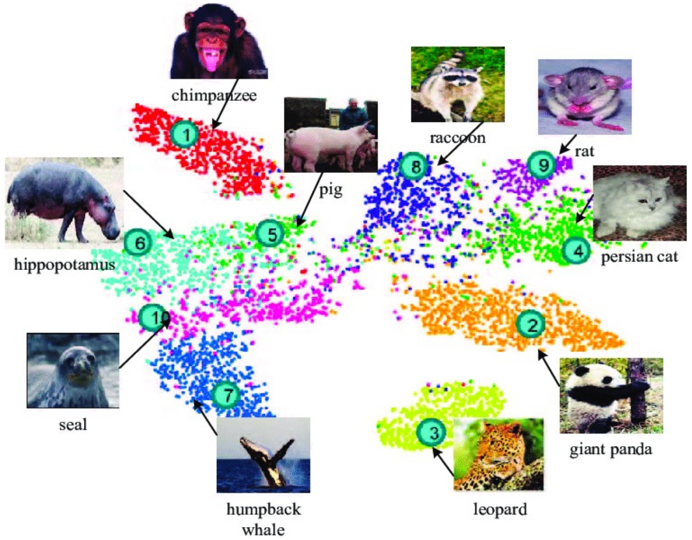
Unsupervised Learning Applications

Find Groups in Social Networks

Identify Types of Exoplanets



Visualize Data



Batch Computing Jobs

Determine Land Use

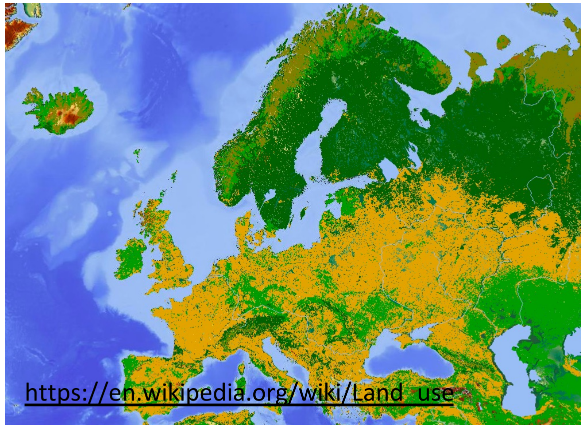
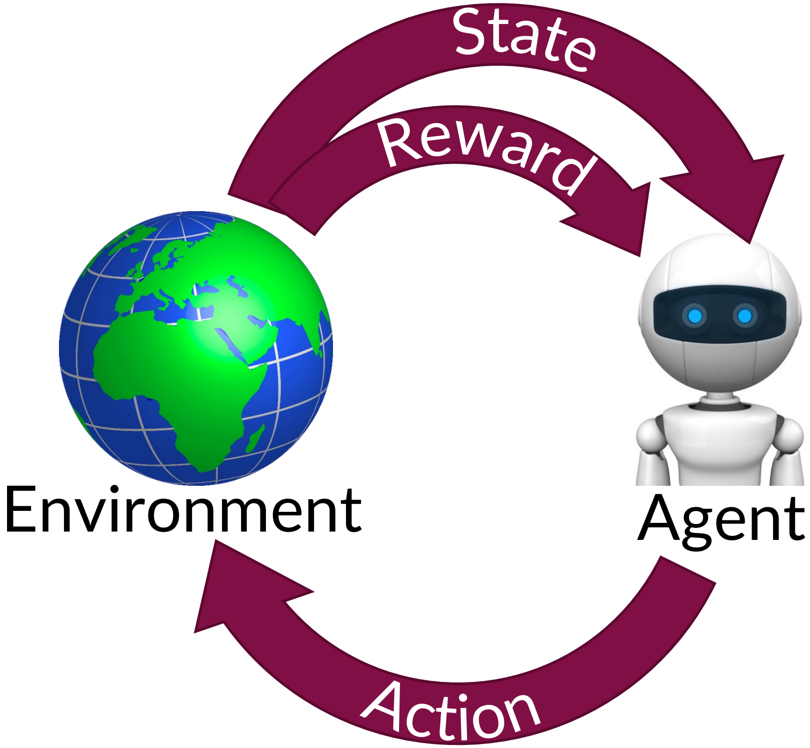


Image Credits:
https://medium.com/graph-commons/finding-organic-clusters-in-your-complex-data-networks-5c27e1d4645d
https://arxiv.org/pdf/1703.08893.pdf

Reinforcement Learning

- Given a sequence of states and actions with (delayed) rewards, output a policy
  - Policy is a mapping from states → actions. It tells you what to do in a given state
- Examples:
  - Game playing
  - Robot grasping an object
  - Balance a pole on your forehead
  - Medical treatment plans for patients

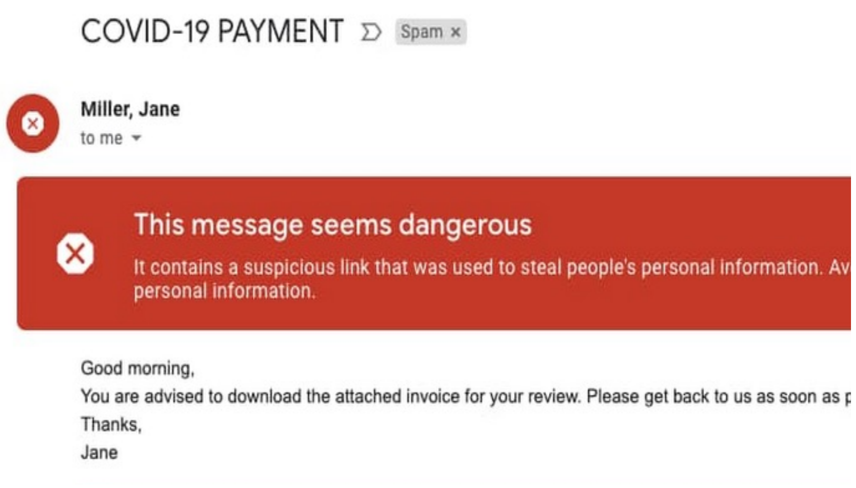


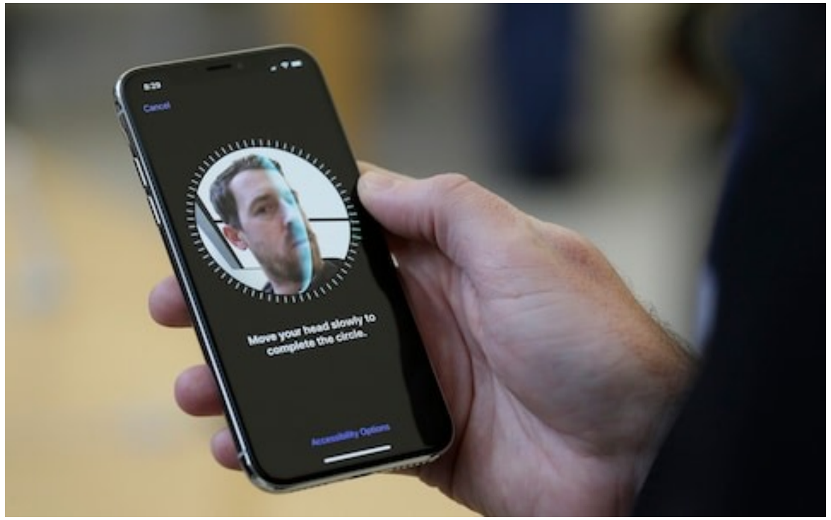
Reinforcement Learning

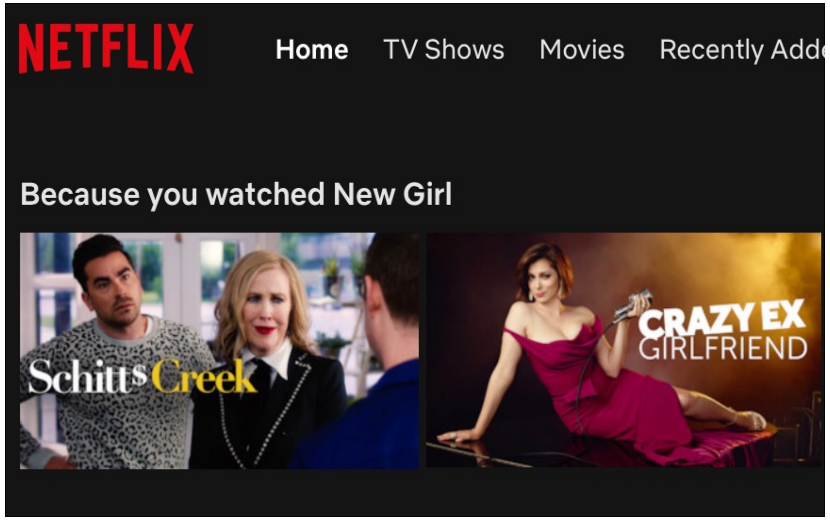
https://www.youtube.com/watch?v=iaF43Ze1oeI

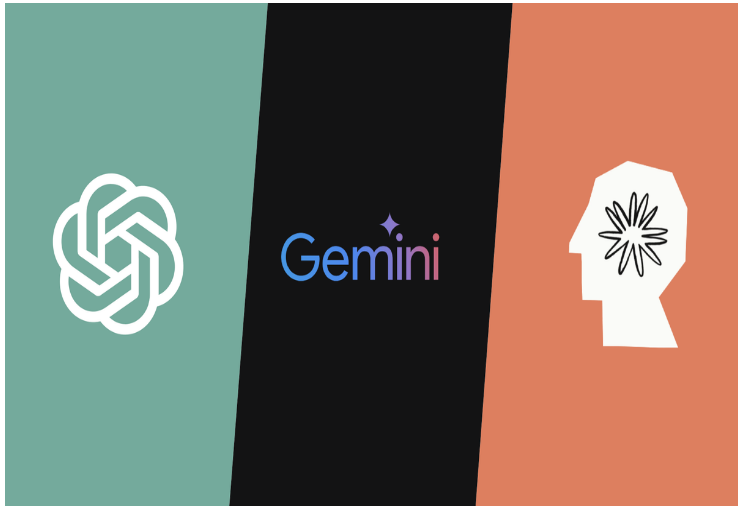
Example Applications of ML

Some everyday ML applications

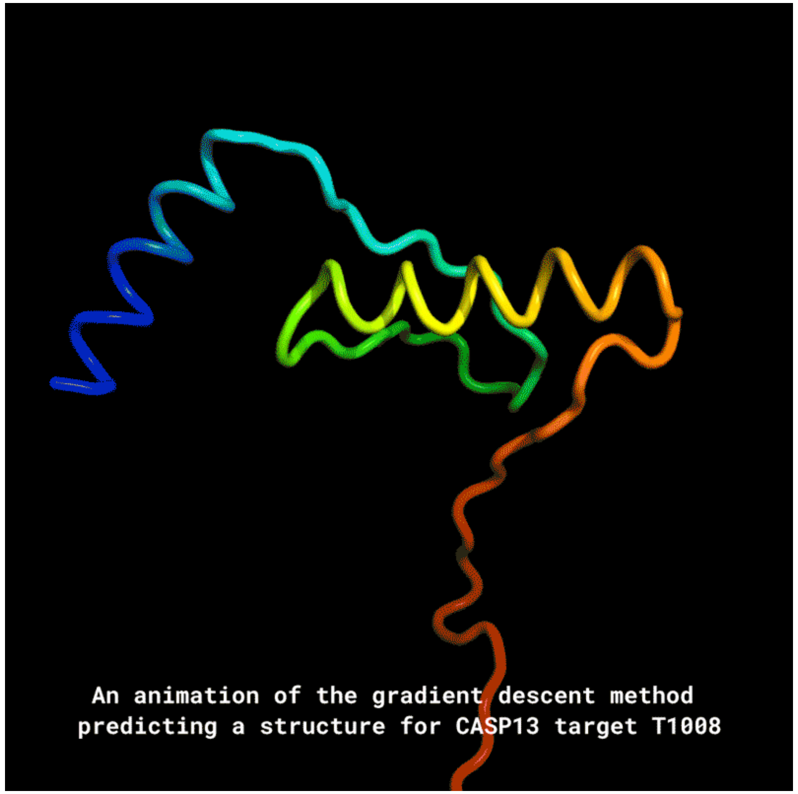




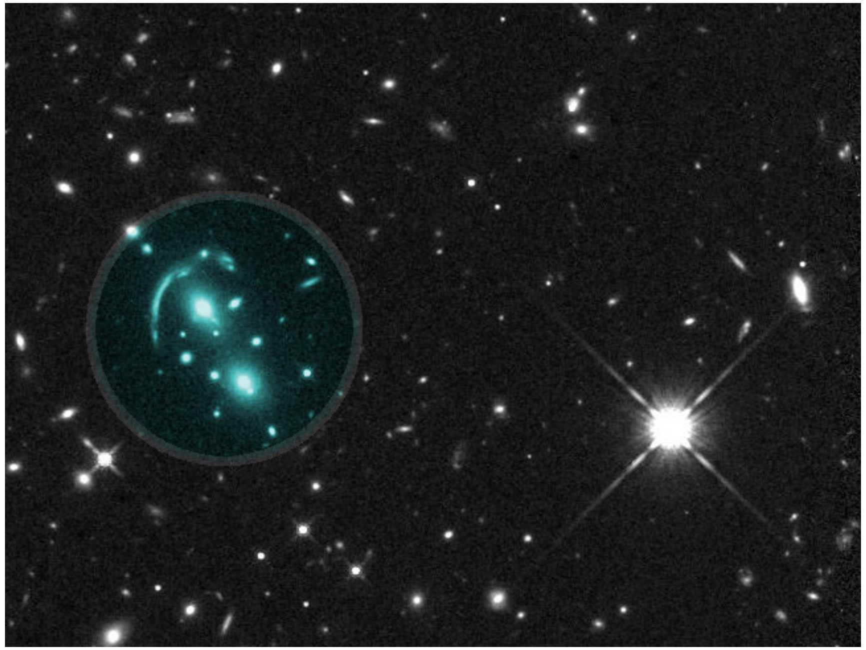




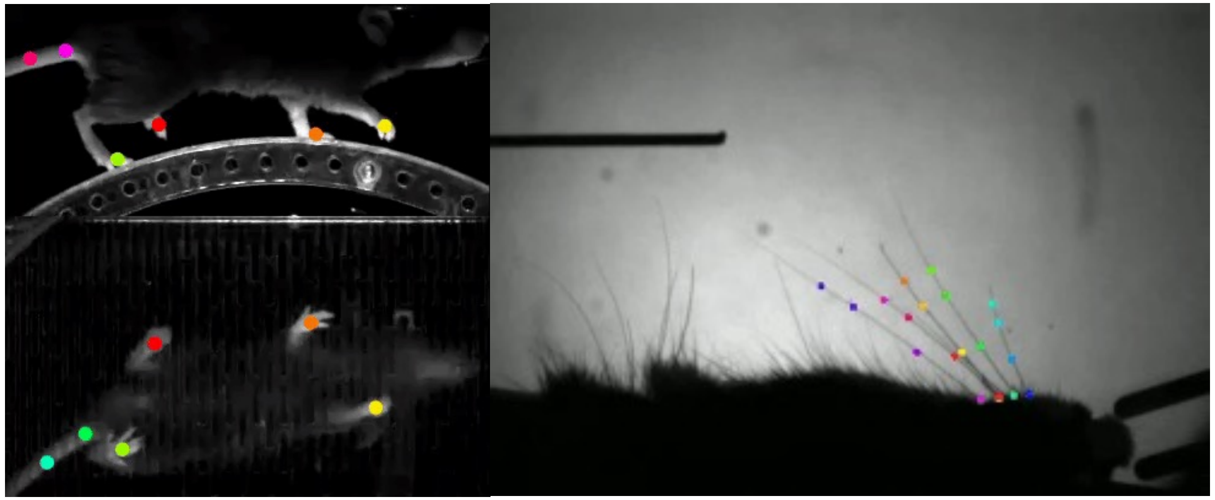
Scientific Discovery



https://deepmind.com/blog/article/AlphaFold-Using-AI-for-scientific-discovery



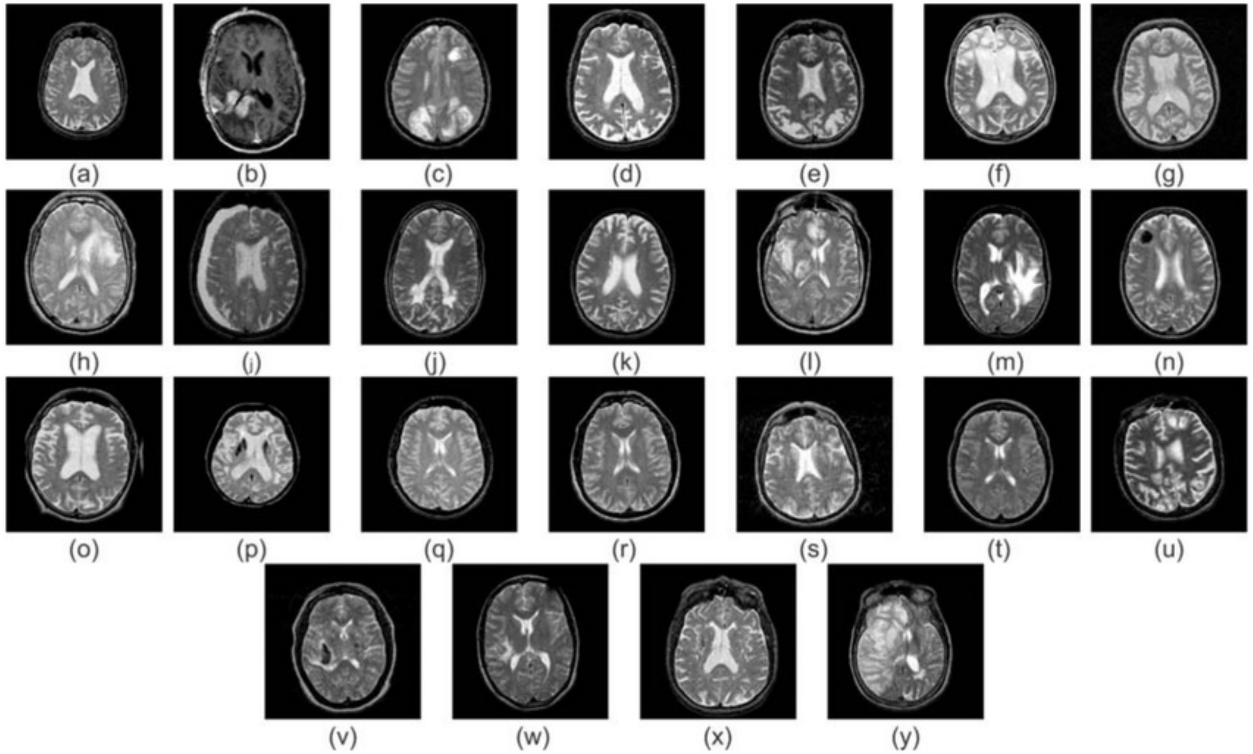
https://www.jpl.nasa.gov/edu/news/2019/4/19/how-scientists-captured-the-first-image-of-a-black-hole/



http://www.mousemotorlab.org/deeplabcut

Radiology and Medicine

Input: brain scans



Output: neurological disease labels

Machine learning studies on major brain diseases: 5-year trends of 2014–2018

Koji Sakai[1] · Kei Yamada[1]

Applications of machine learning in drug discovery and development

https://www.nature.com/articles/s41573-019-0024-5

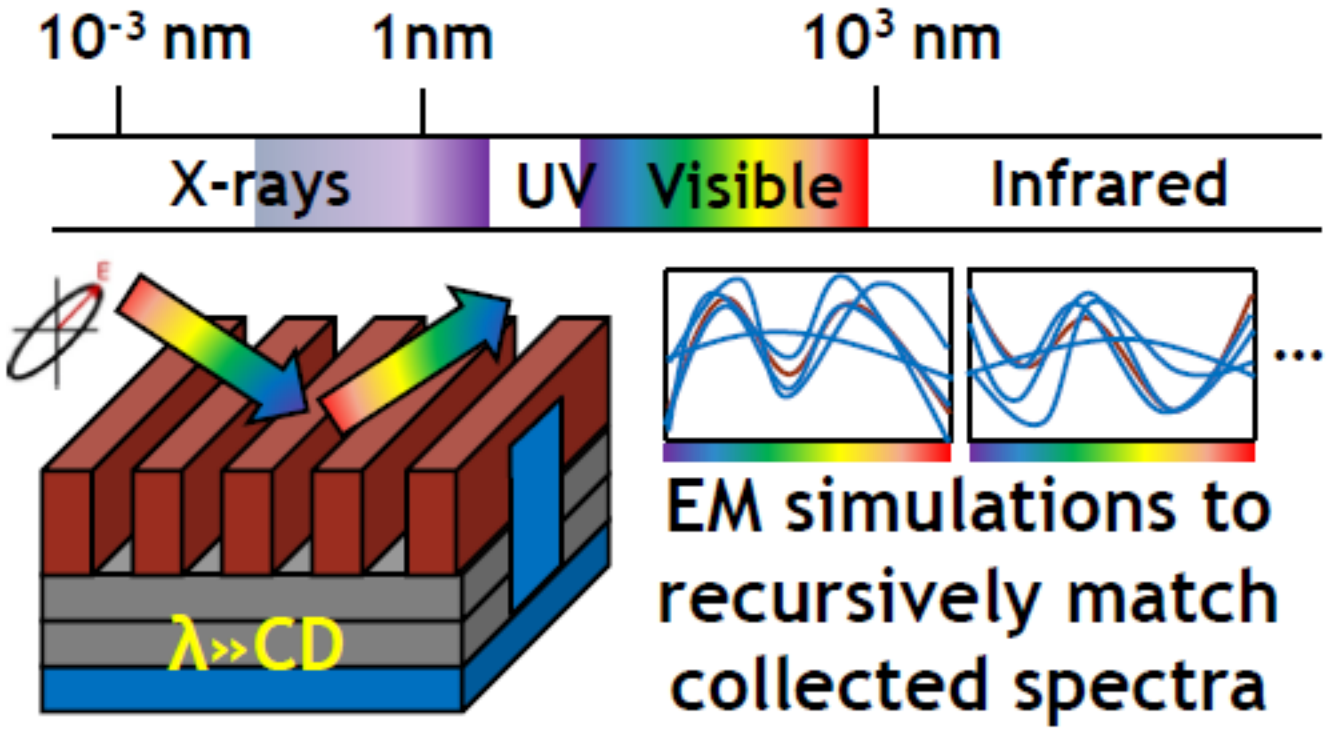
Deep learning-enabled medical computer vision

Andre Esteva, Katherine Chou, Serena Yeung, Nikhil Naik, Ali Madani, Ali Mottaghi, Yun Liu, Eric Topol, Jeff Dean & Richard Socher

https://www.nature.com/articles/s41746-020-00376-2

Optical Metrology in Semiconductor Manufacturing

Input: light spectra after bouncing off silicon wafer

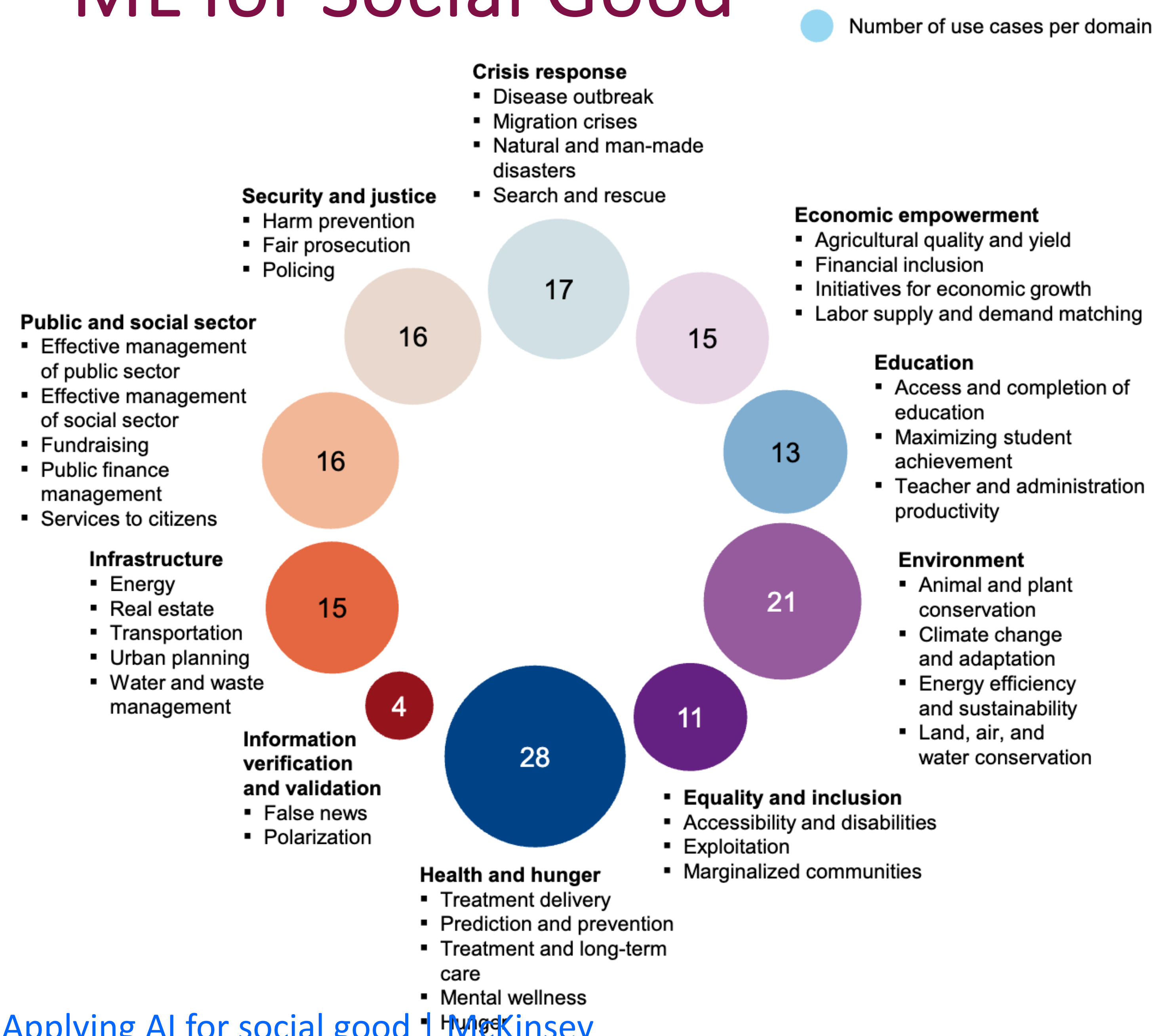


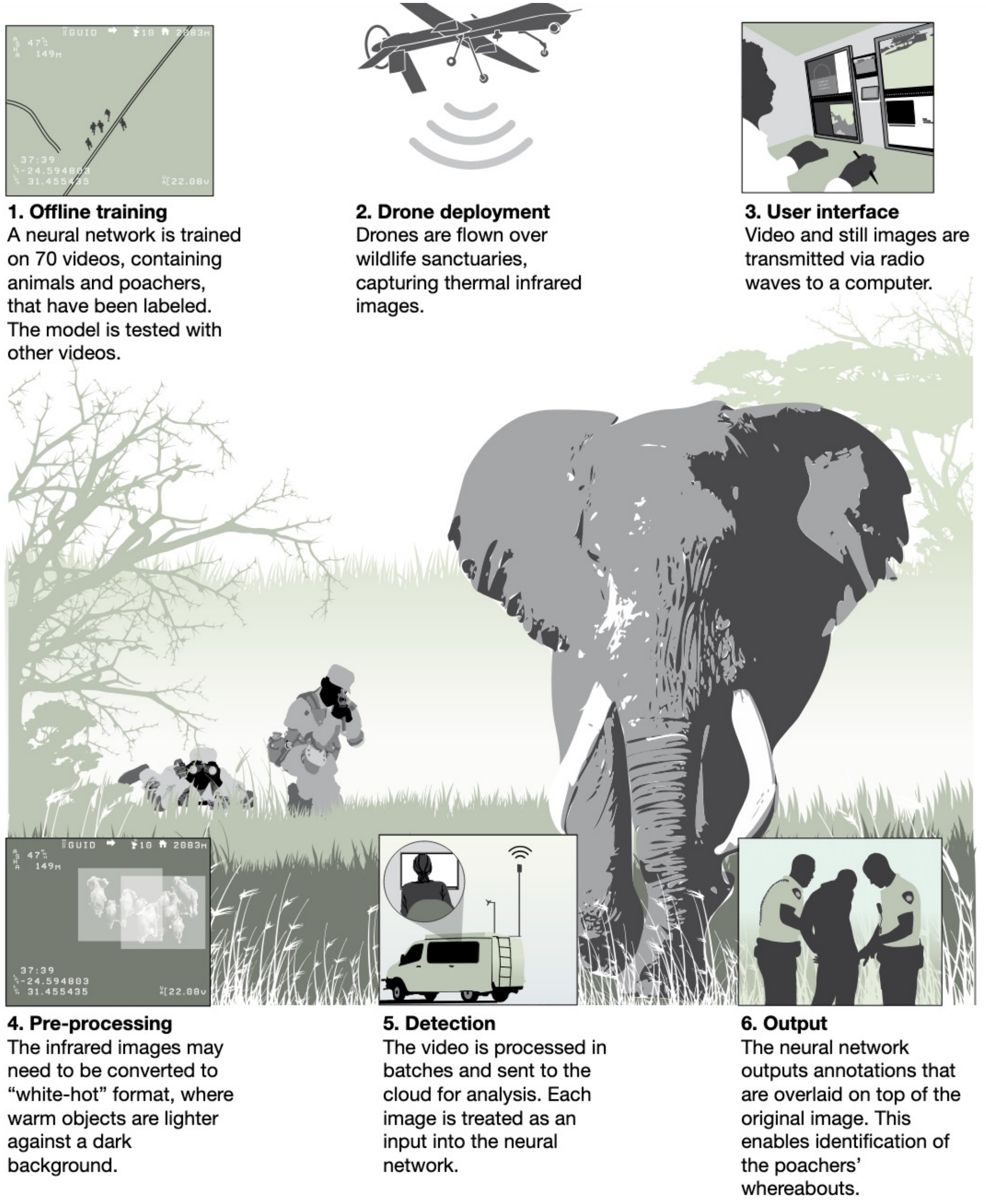
Huge gains in manufacturing throughput -> cheaper phones and computers!

Output: defective / perfect

https://semiwiki.com/semiconductor-manufacturers/287593-a-compelling-application-for-ai-in-semiconductor-manufacturing/

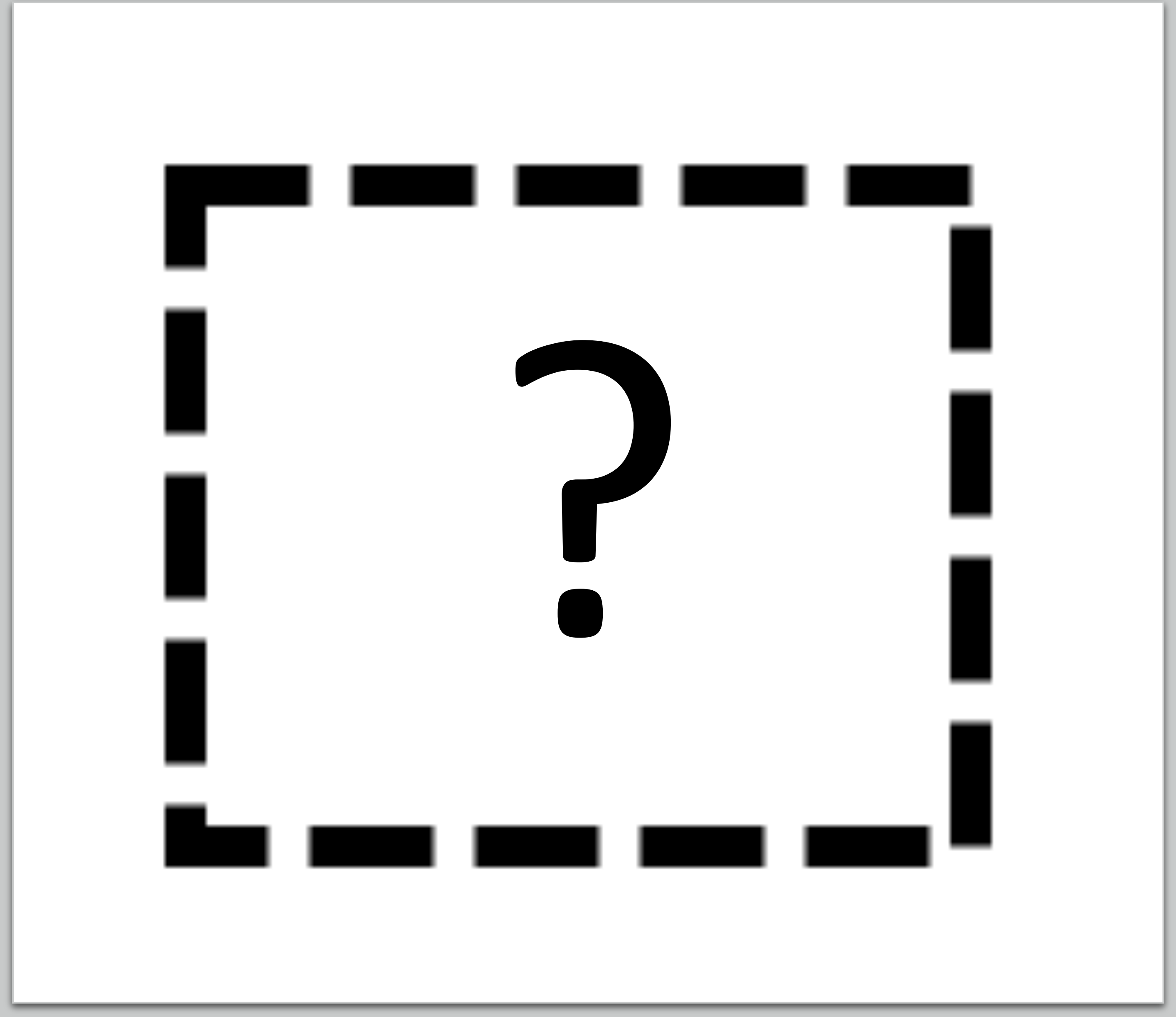
ML for Social Good





Elizabeth Bondi et al., SPOT poachers in action: Augmenting conservation drones with automatic detection in near real time, AAAI 2018

[Applying AI for social good | McKinsey]



Your ML application

Ethical Considerations

"The Pennsylvania Board of Probation and Parole has begun using machine learning forecasts to help inform parole release decisions. In this paper, we evaluate the impact of the forecasts on those decisions and subsequent recidivism."

An impact assessment of machine learning risk forecasts on parole board decisions and recidivism

Richard Berk

"In 2013, the University of Texas at Austin's computer science department began using a machine-learning system called GRADE to help make decisions about who gets into its Ph.D. program"

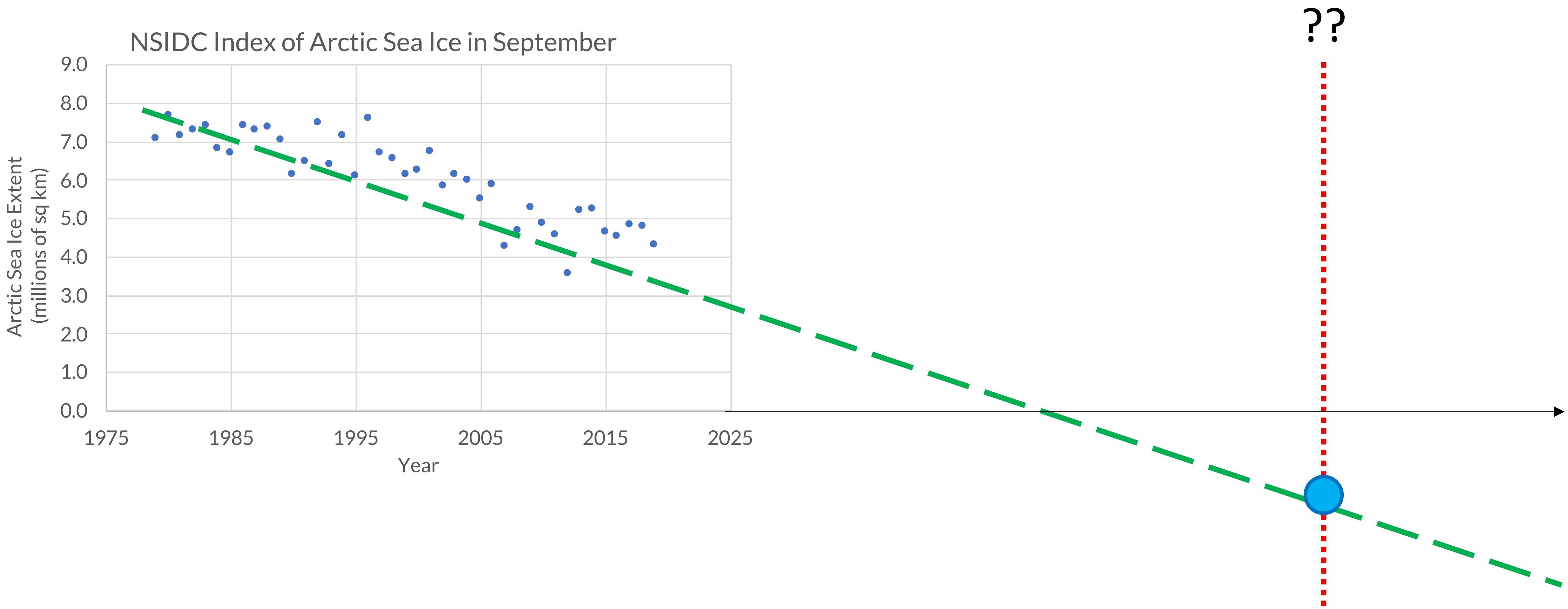
The Death and Life of an Admissions Algorithm

"Videos about vegetarianism led to videos about veganism. Videos about jogging led to videos about running ultramarathons. It seems as if you are never 'hard core' enough for YouTube's recommendation algorithm. It promotes, recommends and disseminates videos in a manner that appears to constantly up the stakes. Given its billion or so users, YouTube may be one of the most powerful radicalizing instruments of the 21st century."

YouTube, the great radicalizer

THE NEW YORK TIMES / ZEYNEP TUFEKCI / MAR 12

Danger of Out-of-Domain Machine Learning



Any time you are evaluating on data “far” from your training data, beware!