



# STAT 453: Introduction to Deep Learning and Generative Models

---

Ben Lengerich

Lecture 01: What are Machine Learning and Deep Learning?

September 3, 2025




# Today

---

1. **Course overview**
2. What is machine learning?
3. The broad categories of ML
4. The supervised learning workflow
5. Necessary ML notation and jargon
6. About the practical aspects and tools



A short teaser:  
what you'll be able to do after this course



# 3D Convolutional Networks

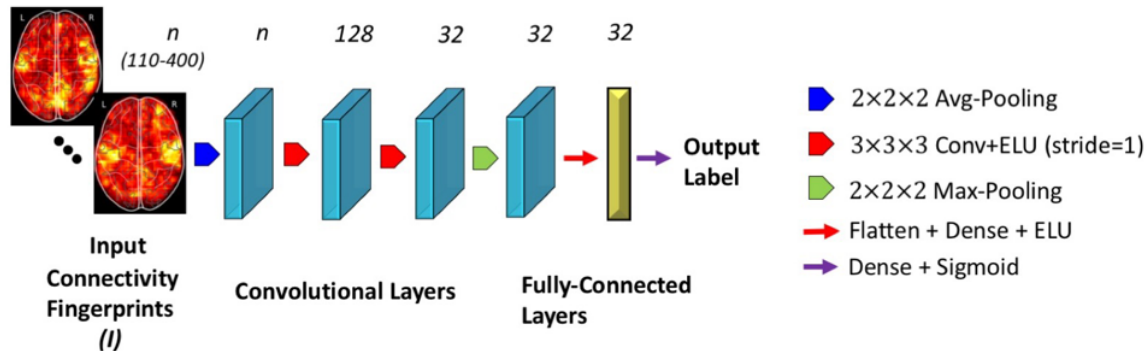
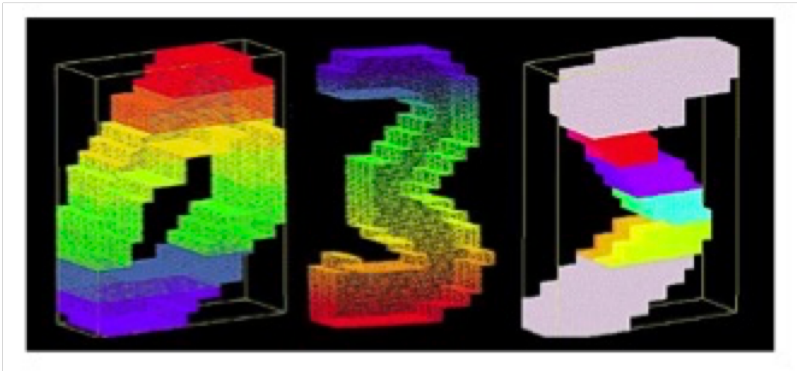
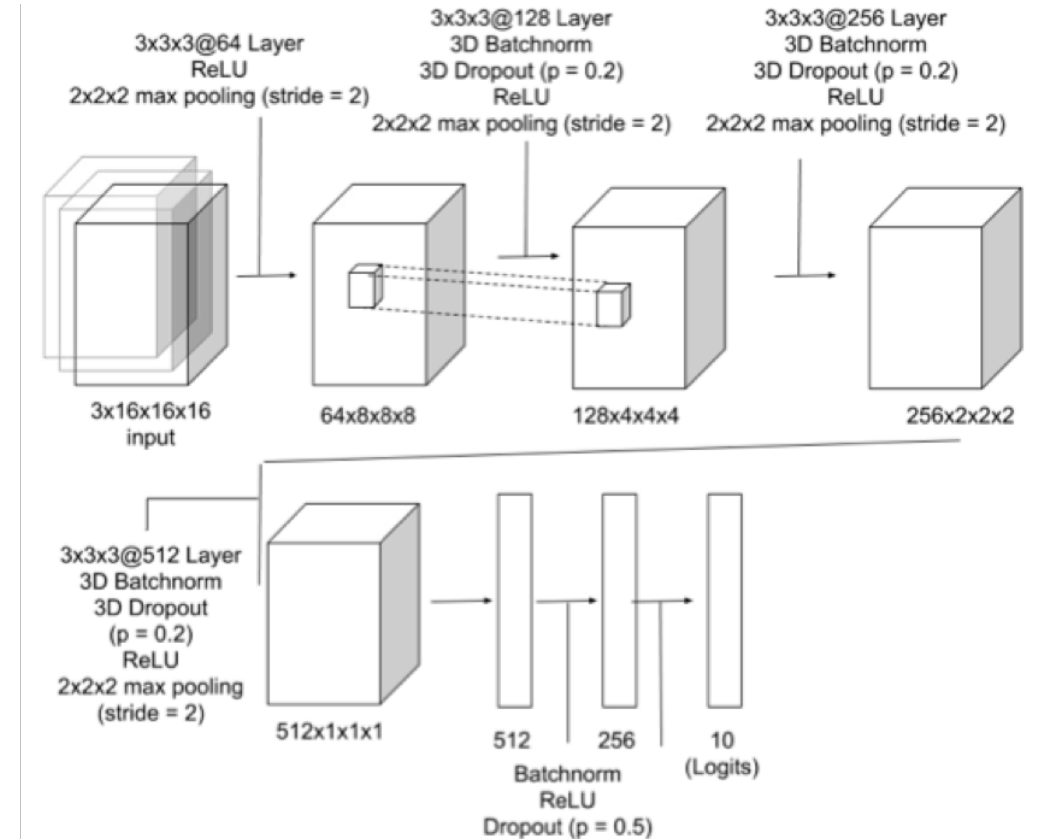


Image Source: 3D Convolutional Neural Networks for Classification of Functional Connectomes. <https://arxiv.org/abs/1806.04209>



# Audio Classification Using Convolutional Neural Networks

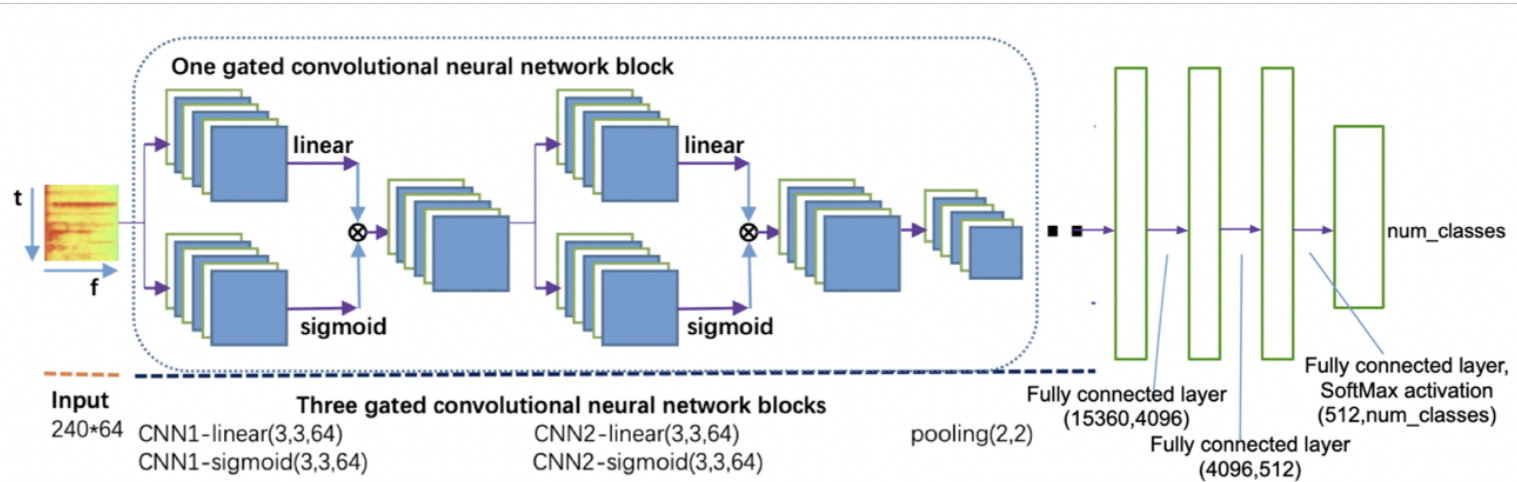
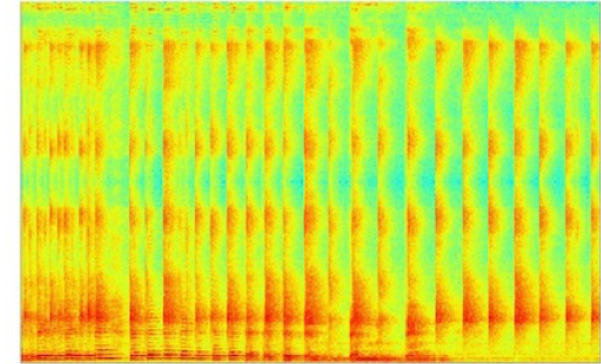
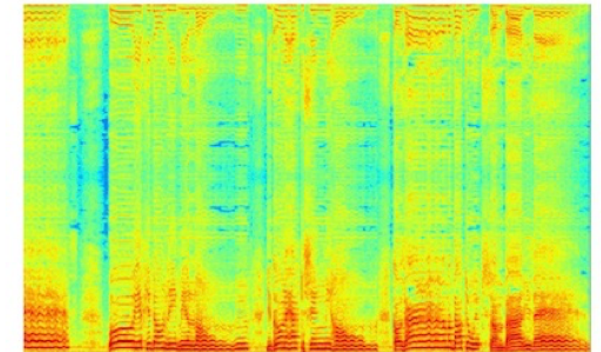


Figure 2. Gated Convolutional Neural Network with Multi-Layered Classification Model

Spectrogram of an audio clip corresponding to "finger snapping:"



Spectrogram of an audio clip corresponding to "synthetic singing:"



# Photographic Style Transfer with Deep Learning



Guernica



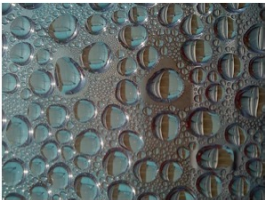
Model Result



Sandstone



Model Result



Water Drop



Model Result

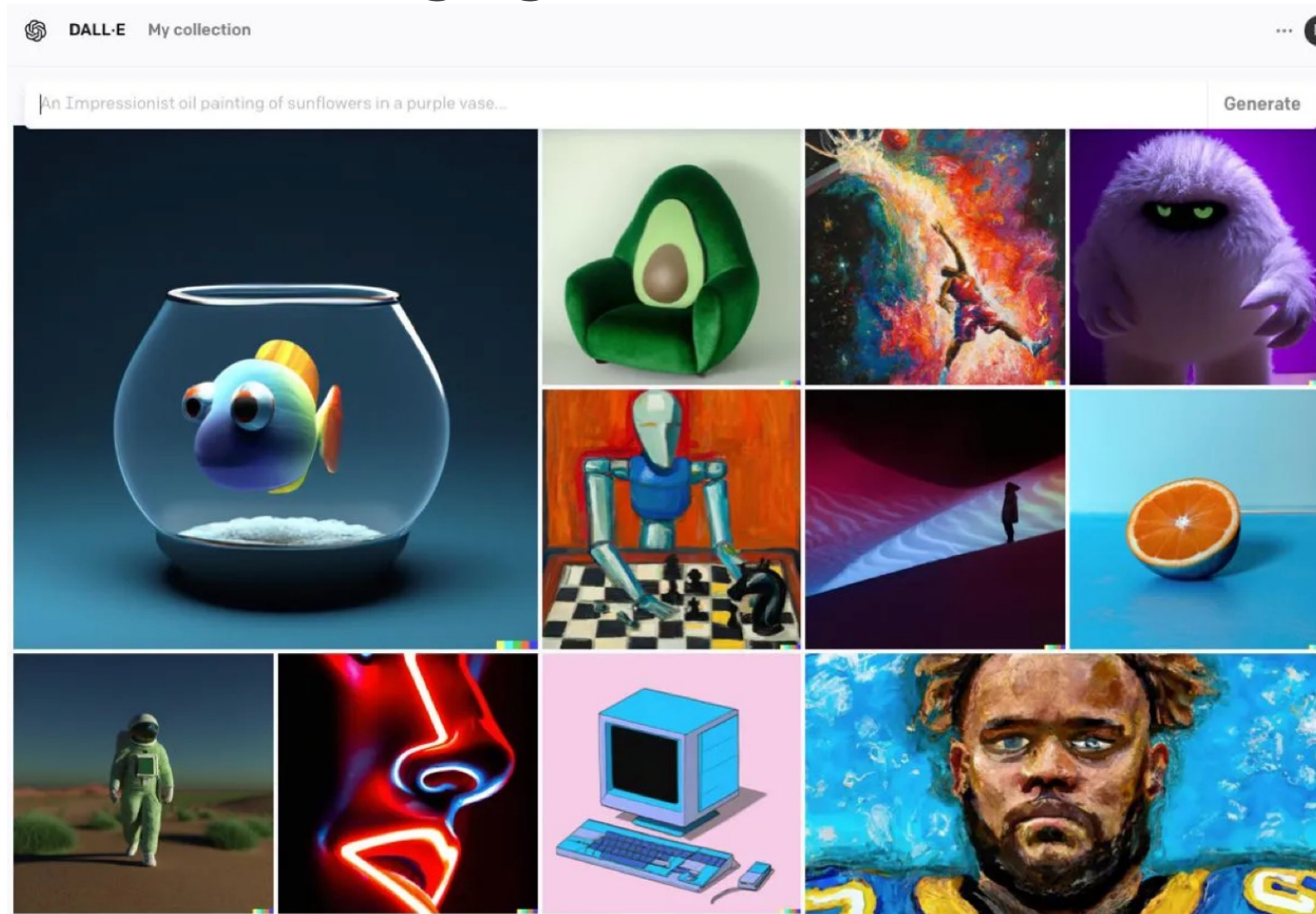


Skrik



Model Result

# Diffusion model and image generation



# Large language model and agents

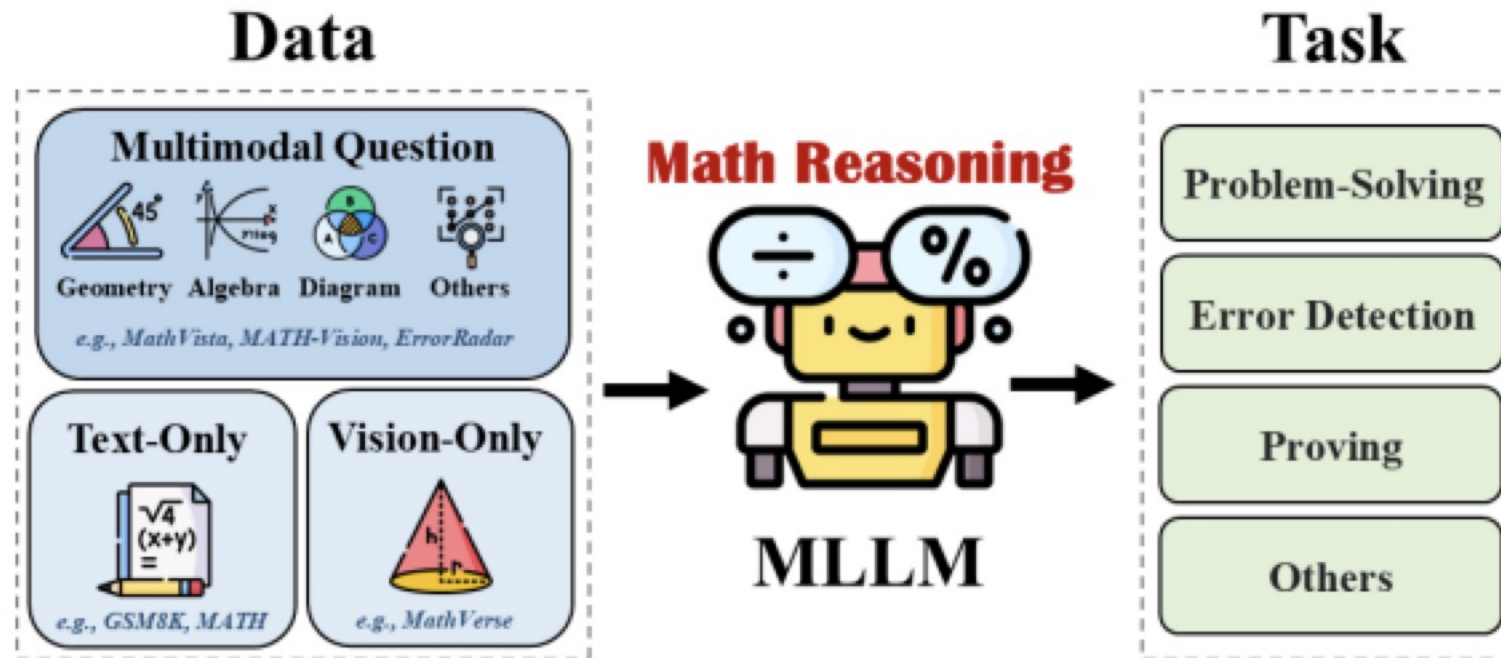


Figure 1: The illustration of our research scope (*i.e.*, investigating the MLLM's math reasoning capability).

# Some course projects from prior years

---

- Spring 2023
  - Breast Cancer detection using ultrasound imaging
  - LSTM music generation
  - Predicting stock prices using LSTMs
  - Creating Surrealism Artworks with DCGAN
  - Exploring the Impact of Activation Functions and Normalization Techniques
- Spring 2024
  - Gradio framework with self-supervised learning
  - Song generation
  - Diagnosis of Chest X-ray Images
  - Audio-to-video image animation
  - Movie recommendations
  - Sentiment analysis using BERT



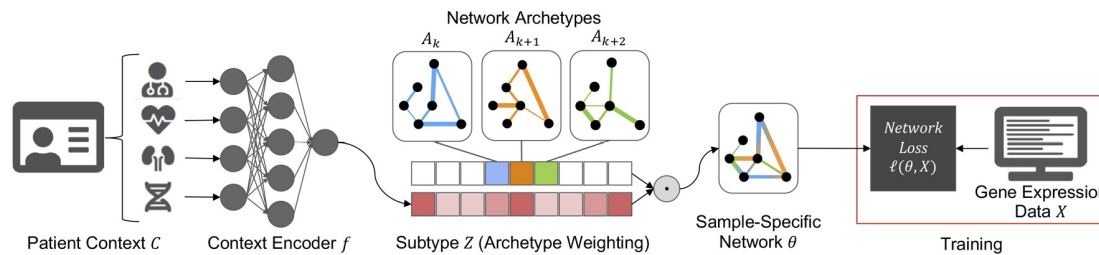
# A bit about your instructor



# My research interests

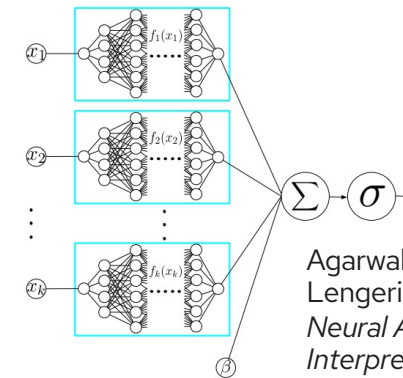
Research Group: [adaptinfer.org](https://adaptinfer.org)

## Contextualized models



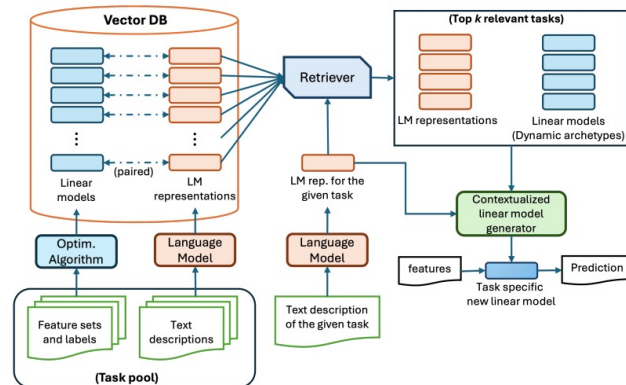
Ellington, Lengerich, Watkins et al. (2025)  
*Learning to estimate sample-specific transcriptional networks for 7000 tumors.*  
 Proceedings of the National Academy of Sciences (PNAS).

## Interpretability and Modularity



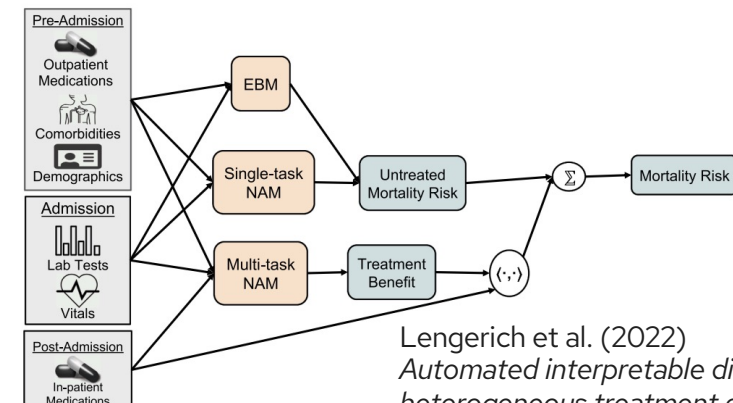
Agarwal, Melnick, Frosst, Zhang, Lengerich, Caruana, Hinton (2021)  
*Neural Additive Models: Interpretable Machine Learning with Neural Nets.* NeurIPS 2021.

## Foundation models as context



Mahbub, Ellington, Alinejad, Wen, Luo, Lengerich, Xing. (2024)  
*From One to Zero: RAG-IM Adapts Language Models for Interpretable Zero-Shot Clinical Predictions*  
 NeurIPS Workshop on Adaptive Foundation Models.

## Biomedical Applications

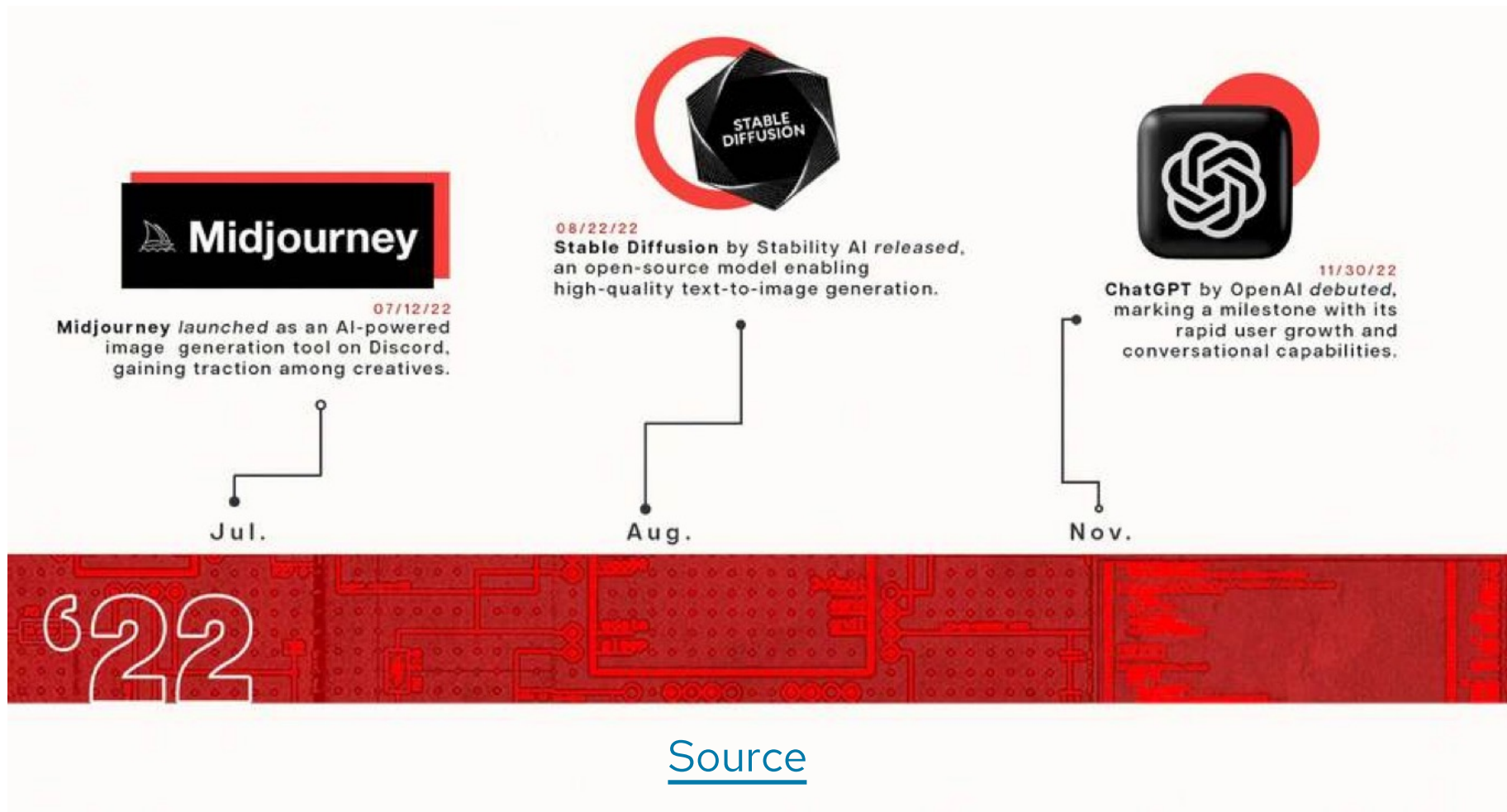


Lengerich et al. (2022)  
*Automated interpretable discovery of heterogeneous treatment effectiveness.*  
 Journal of Biomedical Informatics

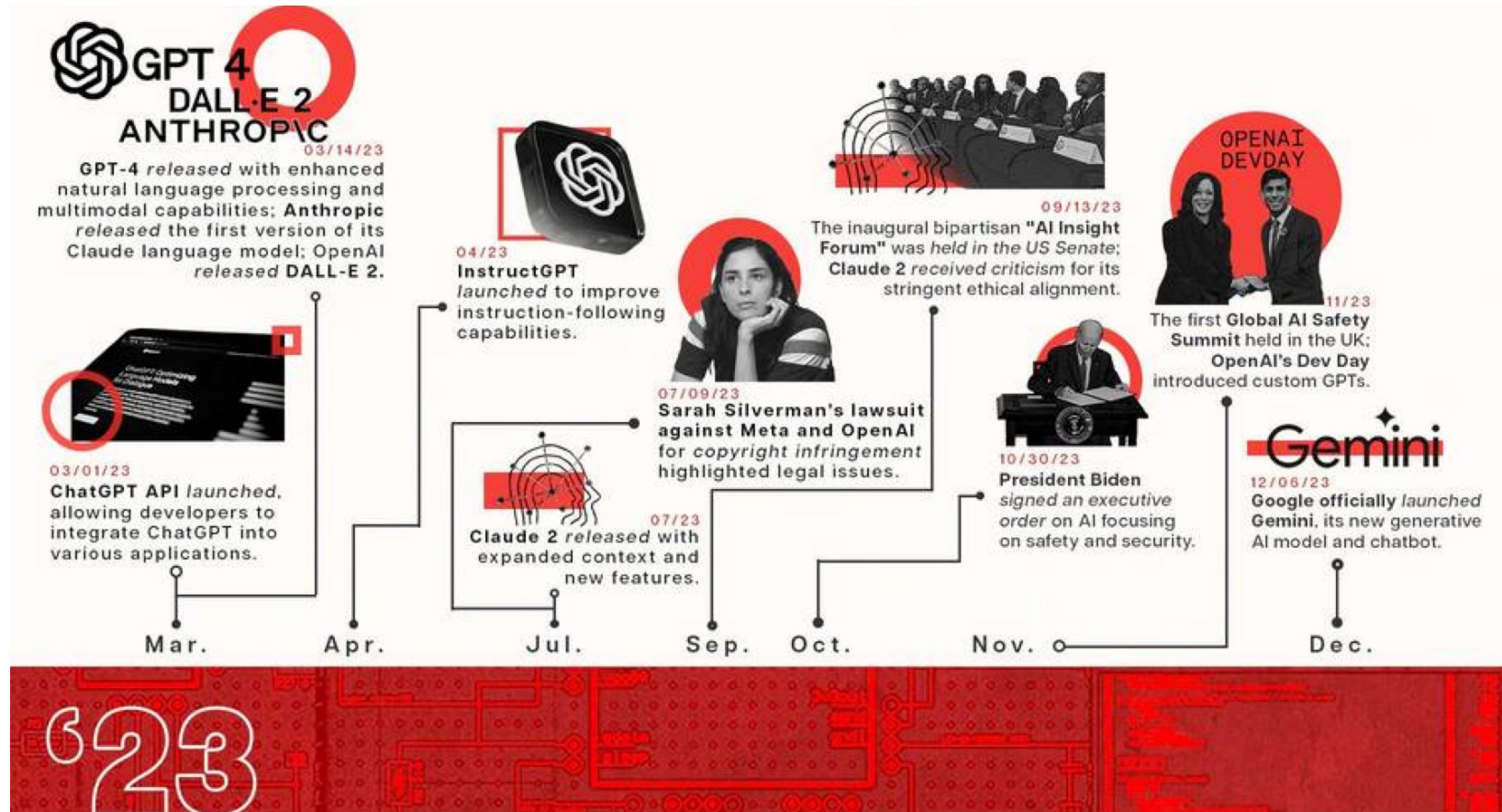


# About the course

# The AI Revolution

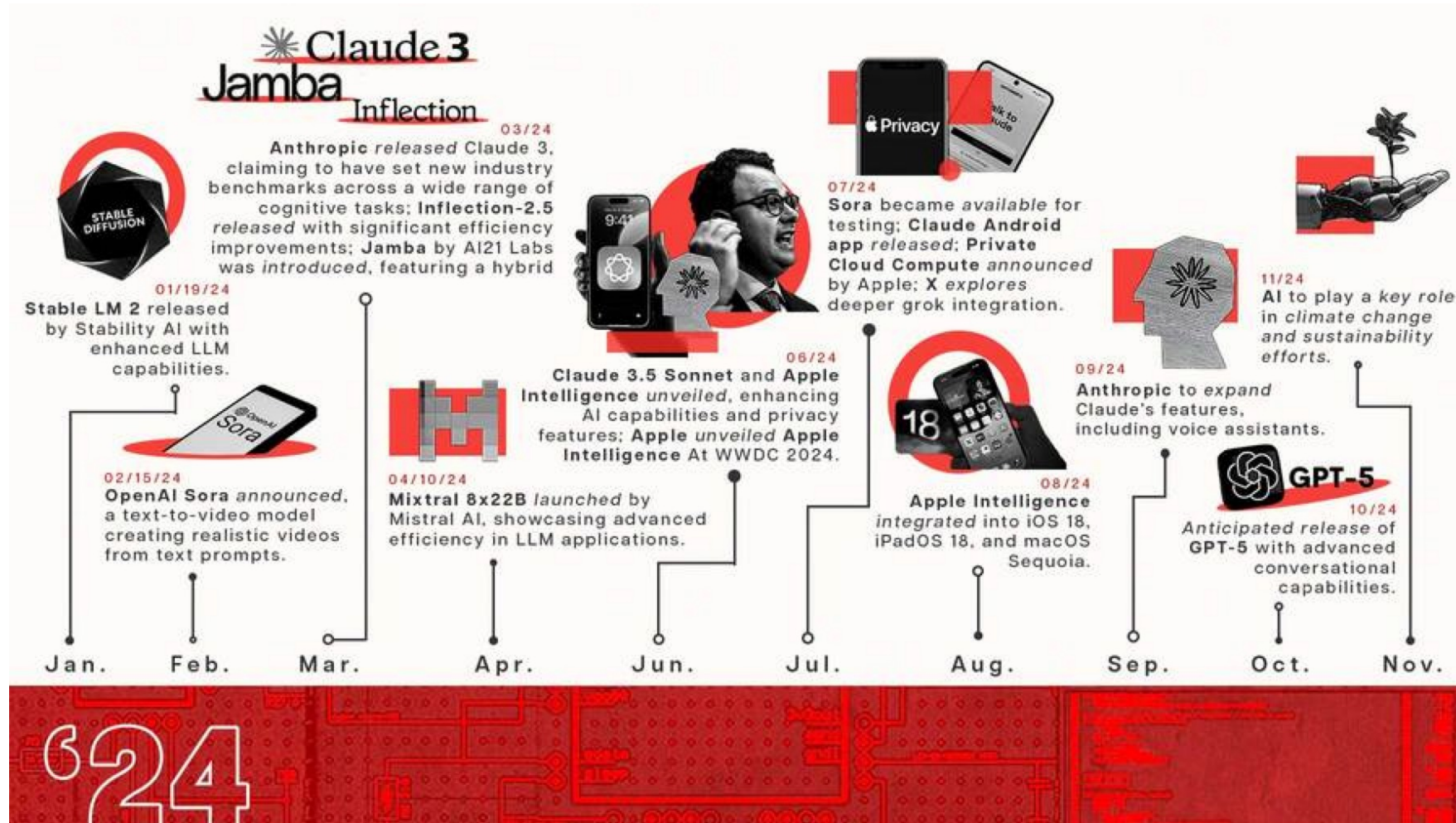


# The AI Revolution



[Source](#)

# The AI Revolution



[Source](#)

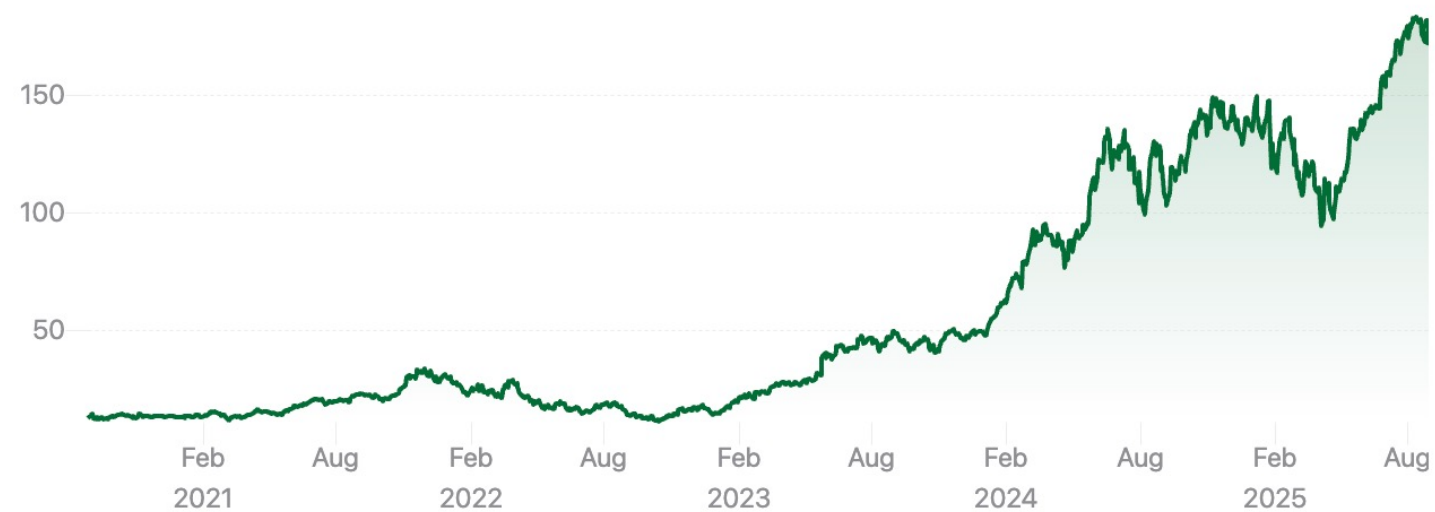
# The AI Revolution

NVIDIA Corporation · NASDAQ:NVDA

**174.11** USD **+161.55 (+1,279.10%)** ↑

Friday, 4:00 PM GMT-4 · Disclaimer

1 Day 5 Days 1 Month Ytd 1 Year **5 Years** Max



Open	178.11	Mkt Cap	4.25T	Prev close	180.17
High	178.15	P/E ratio	48.99	52W high	184.48
Low	173.15	Volume	243M	52W low	86.62



# Course Schedule / Calendar

Week	Lecture Dates	Topic	Assignments
Module 1: Introduction and Foundations			
1	9/3	Course Introduction	
2	9/8, 9/10	A Brief History of DL, Statistics / linear algebra / calculus review	HW1
3	9/15, 9/17	Single-layer networks Parameter Optimization and Gradient Descent	
4	9/22, 9/24	Automatic differentiation with PyTorch, Cluster and cloud computing resources	HW 2
Module 2: Neural Networks			
5	9/29, 10/1	Multinomial logistic regression, Multi-layer perceptrons and backpropagation	
6	10/6, 10/8	Regularization Normalization / Initialization	HW 3
7	10/13, 10/15	Optimization, Learning Rates CNNs	Project Proposal
8	10/20, 10/22	Review, <b>Midterm Exam</b>	In-class Exam


Week	Lecture Dates	Topic	Assignments
Module 3: Intro to Generative Models			
9	10/27, 10/29	A Linear Intro to Generative Models, Factor Analysis, Autoencoders, VAEs	
10	11/3, 11/5	Generative Adversarial Networks, Diffusion Models	Project Midway Report
Module 4: Large Language Models			
11	11/10, 11/12	Sequence Learning with RNNs Attention, Transformers	HW4
12	11/17, 11/19	GPT Architectures, Unsupervised Training of LLMs	
13	11/24, 11/26	Supervised Fine-tuning of LLMs, Prompts and In-context learning	HW5
14	12/1, 12/3	Foundation models, alignment, explainability Open directions in LLM research	
15	12/8, 12/10	<b>Project Presentations</b>	Project Final Report
16	12/17	<b>Final Exam</b>	Final Exam

# Course webpage

[adaptinfer.org/dgm-fall-2025](https://adaptinfer.org/dgm-fall-2025)

STAT 453


logistics lectures notes calendar homework project



## Deep Learning and Generative Models

STAT 453 • Fall 2025 • UW-Madison

- **Time:** Monday/Wednesday 8:00-9:15am
- **Location:** Morgridge 1524
- **Discussion:** [Canvas](#)
- **HW submission:** [Canvas](#)
- **Contact:** Students should ask all course-related questions on [Canvas](#), where you will also find announcements. Individual enquires can be directed to TA/instructor emails.



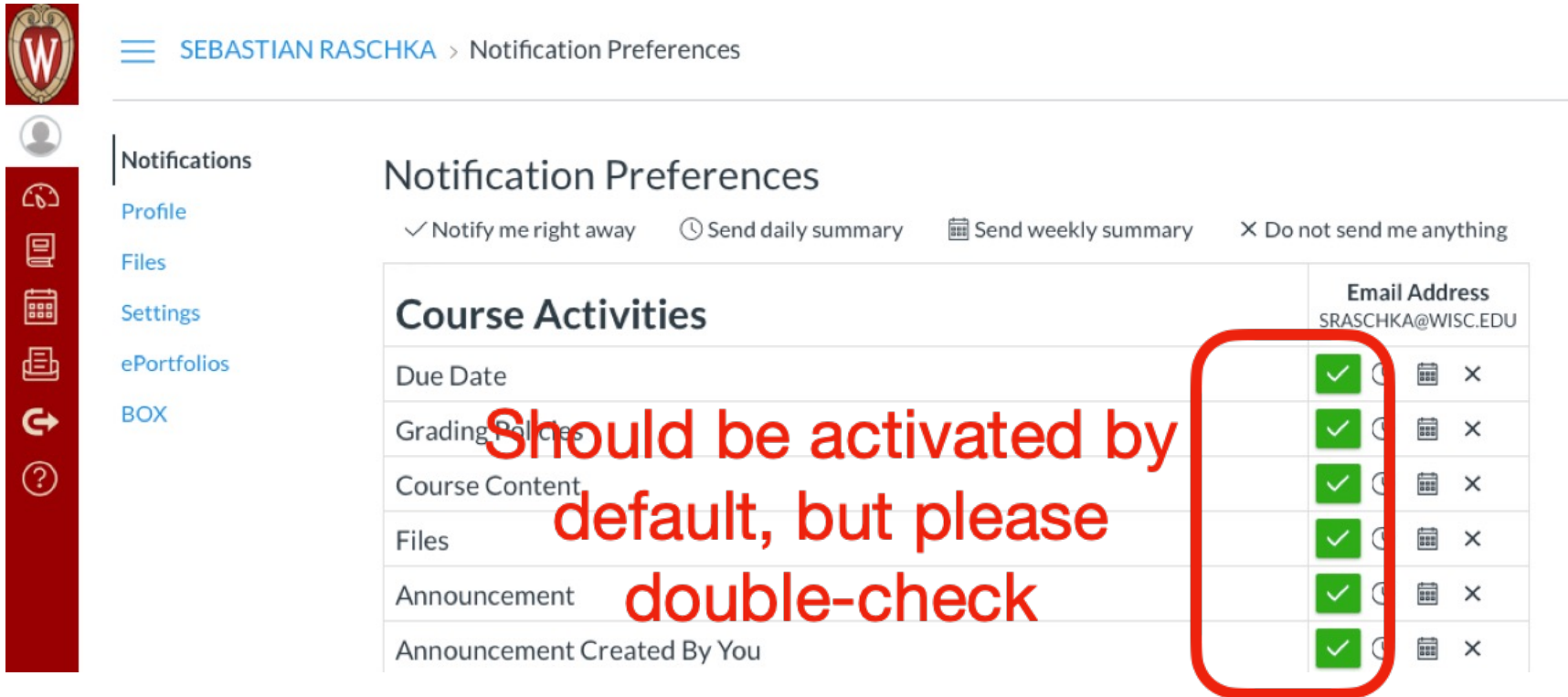
Instructor [Benjamin Lengerich](#)  
Email: [lengerich@wisc.edu](mailto:lengerich@wisc.edu)  
Office hours: TBD

TA [Baiheng Chen](#)  
Email: [bchen342@wisc.edu](mailto:bchen342@wisc.edu)  
Office hours: TBD

+ **Canvas for assignments / submissions / grades**



# Check your settings on Canvas



The screenshot shows the Canvas Notification Preferences page for Sebastian Raschka. The left sidebar contains navigation links: Notifications, Profile, Files, Settings, ePortfolios, and BOX. The main content area is titled "Notification Preferences" and includes a summary of notification settings: "Notify me right away" (checked), "Send daily summary" (clock icon), "Send weekly summary" (calendar icon), and "Do not send me anything" (X icon). Below this is a table of notification preferences for the email address SRASCHKA@WISC.EDU. The table has columns for the notification type, a green checkmark indicating it is active, a clock icon for daily summaries, a calendar icon for weekly summaries, and an X icon for disabling notifications. A red box highlights the first six rows of the table, which are all active. A red text overlay reads: "Should be activated by default, but please double-check".

SEBASTIAN RASCHKA > Notification Preferences

Notifications

Profile

Files

Settings

ePortfolios

BOX

## Notification Preferences

✓ Notify me right away   ⌚ Send daily summary   📅 Send weekly summary   ✕ Do not send me anything

Course Activities	Email Address
	SRASCHKA@WISC.EDU
Due Date	✓ ⌚ 📅 ✕
Grading Policies	✓ ⌚ 📅 ✕
Course Content	✓ ⌚ 📅 ✕
Files	✓ ⌚ 📅 ✕
Announcement	✓ ⌚ 📅 ✕
Announcement Created By You	✓ ⌚ 📅 ✕

Should be activated by default, but please double-check

# Logistics

- Textbook:
  - Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python. Sebastian Raschka, Yuxi (Hayden) Liu, Vahid Mirjalili. Packt Publishing; 1st edition February 25, 2022
  - Used as a reference – exams will be based on lecture material
- Class Announcements: **Canvas**
- Assignment Submissions: **Canvas**
- Instructor: **Ben Lengerich**
  - Office Hours: Mondays 10am-11am, Morgridge 5683
  - Email: [lengerich@wisc.edu](mailto:lengerich@wisc.edu)
- TA: **Baiheng Chen**
  - Office Hours: TBD
  - Email: [bchen342@wisc.edu](mailto:bchen342@wisc.edu)



# Grading

---

- **20% HW Assignments**
- **20% Midterm Exam**
  - End of Module 2 (~10/22)
  - Format: In-class, Open-notes. No calculator / phone / AI allowed.
- **30% Final Exam**
  - 12/17 5:05PM
  - Format: Location TBA. Open-notes. No calculator / phone / AI allowed.
- **30% Final Project**
  - Milestones dues along the way
- **Lecture Notes** (up to 5% extra credit)

# Extra Credit: Lecture Notes

- **Opportunity:** Sign up to write lecture notes for each lecture. If the notes meet quality standards, students will receive +2% extra credit added to their final grade.
  - **Sign-up sheet**
  - You are expected to work together with the other scribe(s) to submit 1 note document for the lecture.
  - Accepted notes will be hosted on [our course webpage](#) (optionally with your name attached, your choice).
- **Submission:** Submit via GitHub Pull Request [for the course webpage](#).
  - Lecture notes must be submitted within one week.
- **Template** is available [on the webpage](#).

## Extra Credit: Lecture Notes (cont.)

- **Opportunity:** You can also receive +1% by making a material improvement to a classmate's lecture note.
- Notes will be posted on the [course's GitHub page](#); if you see an opportunity to improve a classmate's lecture note, submit a GitHub Pull Request with the improvement.
- You can do this up to 3 times for credit.
- Max extra credit is 5%:
  - 2% for a planned lecture note
  - 3 x 1% for improvements to classmate's lecture notes

# Grading Scale

---

- **No curving:**
  - A: 93–100%
  - AB: 88–92%
  - B: 83–87%
  - BC: 78–82%
  - C: 70–77%
  - D: 60–69%
  - F: Below 60%





# Homework Policies

---

- Assignments must be submitted via Canvas by the deadline (typically Fridays at 11:59 PM).
- **Late submissions** will incur a penalty of 10% per day, up to three days, at which they will not be accepted.
- **Collaboration:** Students may collaborate on understanding the problems, but all submitted work must be individual. Use of AI tools is allowed, but answers must reflect your own understanding. You are responsible for ensuring no plagiarism.



# Project

---

- **Proposal** (5%)
- **Midway Report** (5%)
- **Presentation** (5%)
- **Report** (15%)
- **Collaboration:** Teams of up to four students are allowed.
- **Honors Optional Component:** Individual extension to your project. Email me!



# Attendance / Participation

---

- Attendance and participation are expected, though they are not part of the formal grade calculation.
- Contact the instructor in advance of extended absences.

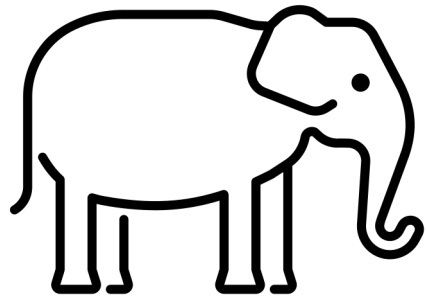
# Calendar



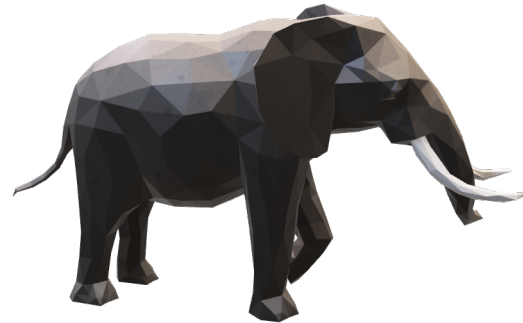
Today < > September 2025						
Month						
SUN 31	MON Sep 1	TUE 2	WED 3 First day of class • 8am STAT 453 L	THU 4	FRI 5	SAT 6
7	8 • 8am STAT 453 L • 10am Prof. Lenç	9	10 • 8am STAT 453 L	11	12	13
14	15 • 8am STAT 453 L • 10am Prof. Lenç	16	17 • 8am STAT 453 L	18	19	20
21	22 • 8am STAT 453 L • 10am Prof. Lenç	23	24 • 8am STAT 453 L	25	26	27
28	29 • 8am STAT 453 L • 10am Prof. Lenç	30	Oct 1 • 8am STAT 453 L	2	3	4
STAT 453 Fall 2025						
Events shown in time zone: (GMT-05:00) Central Time - Chicago						
<a href="#">Add to Google Calendar</a>						
Google Calendar						

# My philosophy

---



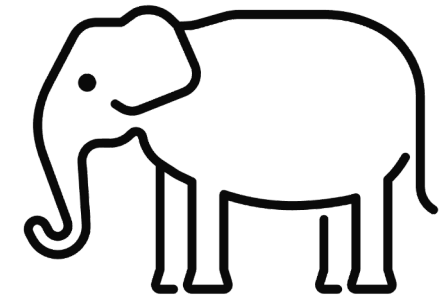
Lectures



Readings



Homeworks



Quizzes/Exams



# Academic Integrity

---

- By virtue of enrollment, each student agrees to uphold the high academic standards of the University of Wisconsin–Madison; academic misconduct is behavior that negatively impacts the integrity of the institution. Cheating, fabrication, plagiarism, unauthorized collaboration and helping others commit these previously listed acts are examples of misconduct which may result in disciplinary action. Examples of disciplinary sanctions include, but are not limited to, failure on the assignment/course, written reprimand, disciplinary probation, suspension or expulsion.



# Mental Health & Wellbeing

- Students often experience stressors that can impact both their academic experience and personal well-being. These may include mental health concerns, substance misuse, sexual or relationship violence, family circumstances, campus climate, financial matters, among others.
- UW–Madison students are encouraged to learn about and utilize the university's mental health services and/or other resources as needed. Students can visit **uhs.wisc.edu** or call **University Health Services** at **(608) 265-5600** to learn more.



**Questions about Course Logistics?**



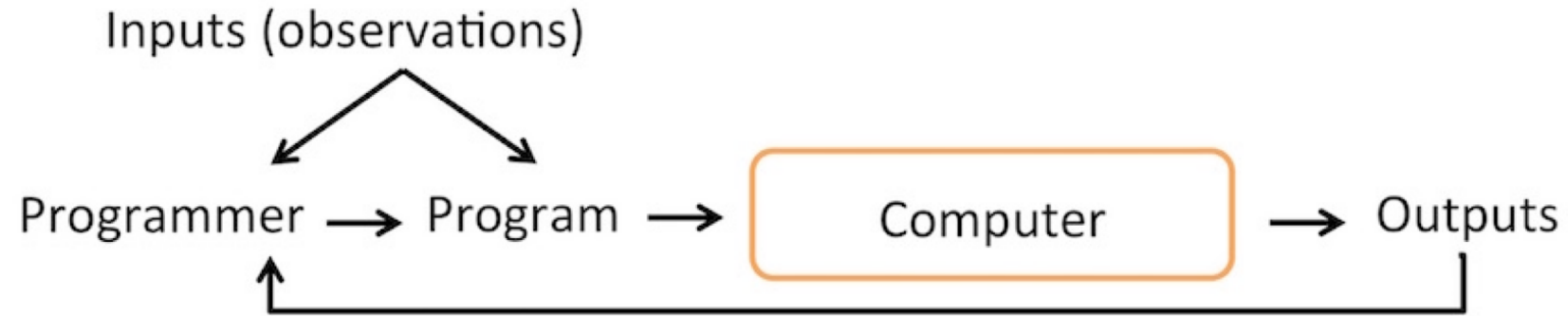
# Today

---

1. Course overview
- 2. What is machine learning?**
3. The broad categories of ML
4. The supervised learning workflow
5. Necessary ML notation and jargon
6. About the practical aspects and tools

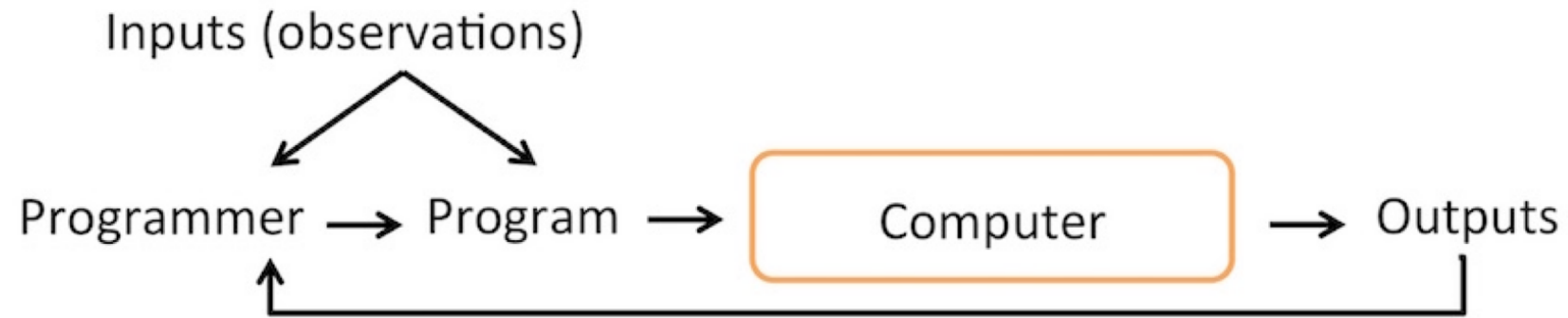
# What is Machine Learning?

## The Traditional Programming Paradigm

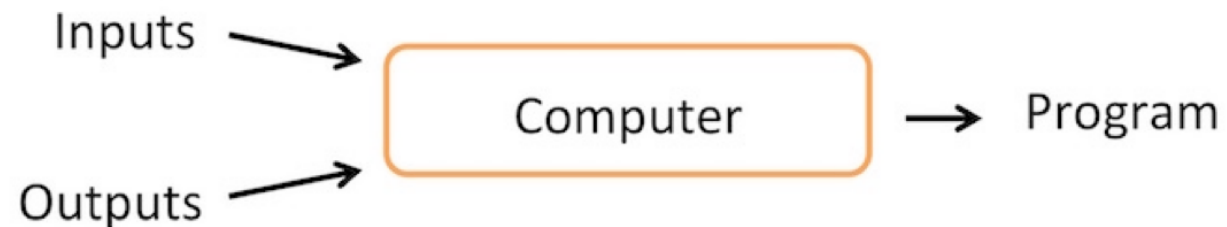


# What is Machine Learning?

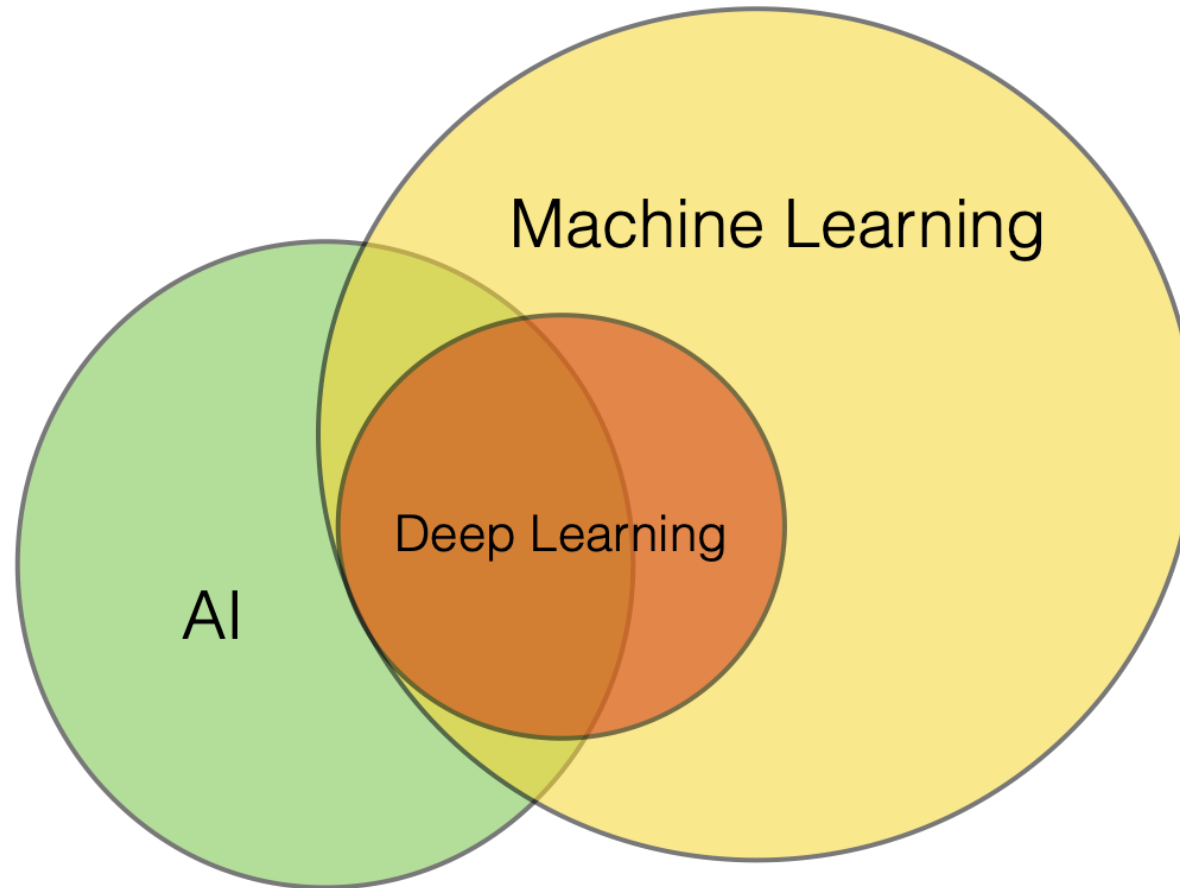
## The Traditional Programming Paradigm



## Machine Learning

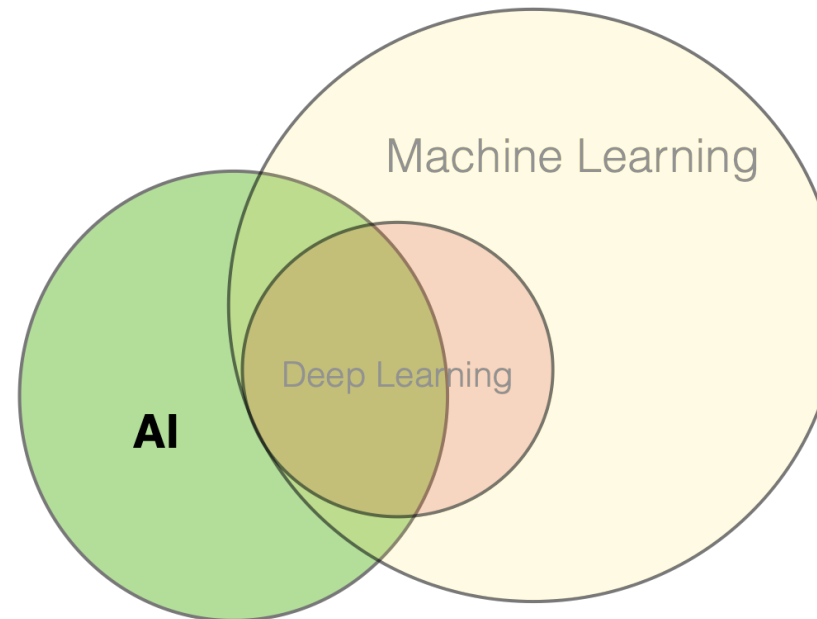


# The Connection Between Fields

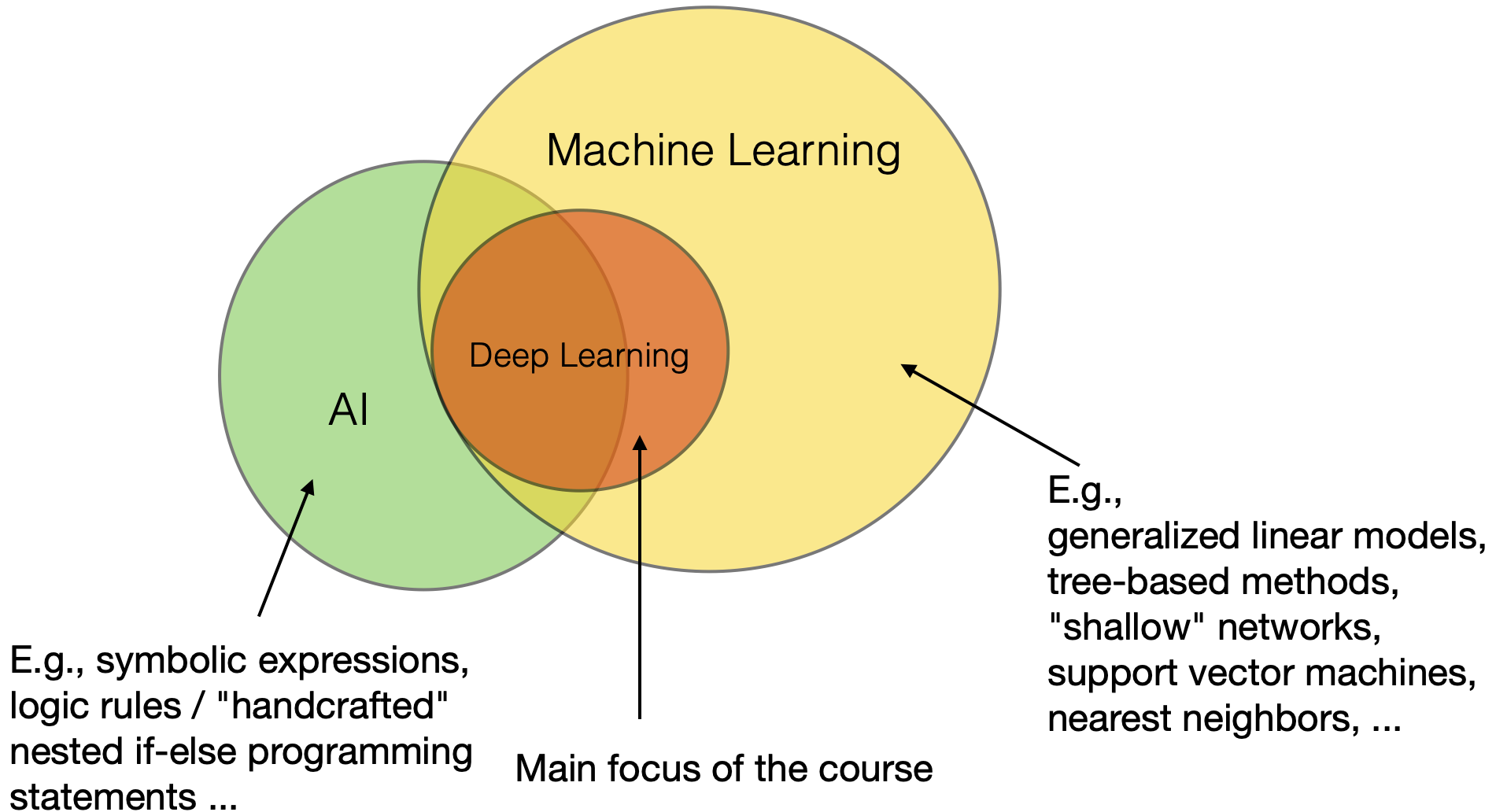


# Not all AI Systems involve Machine Learning

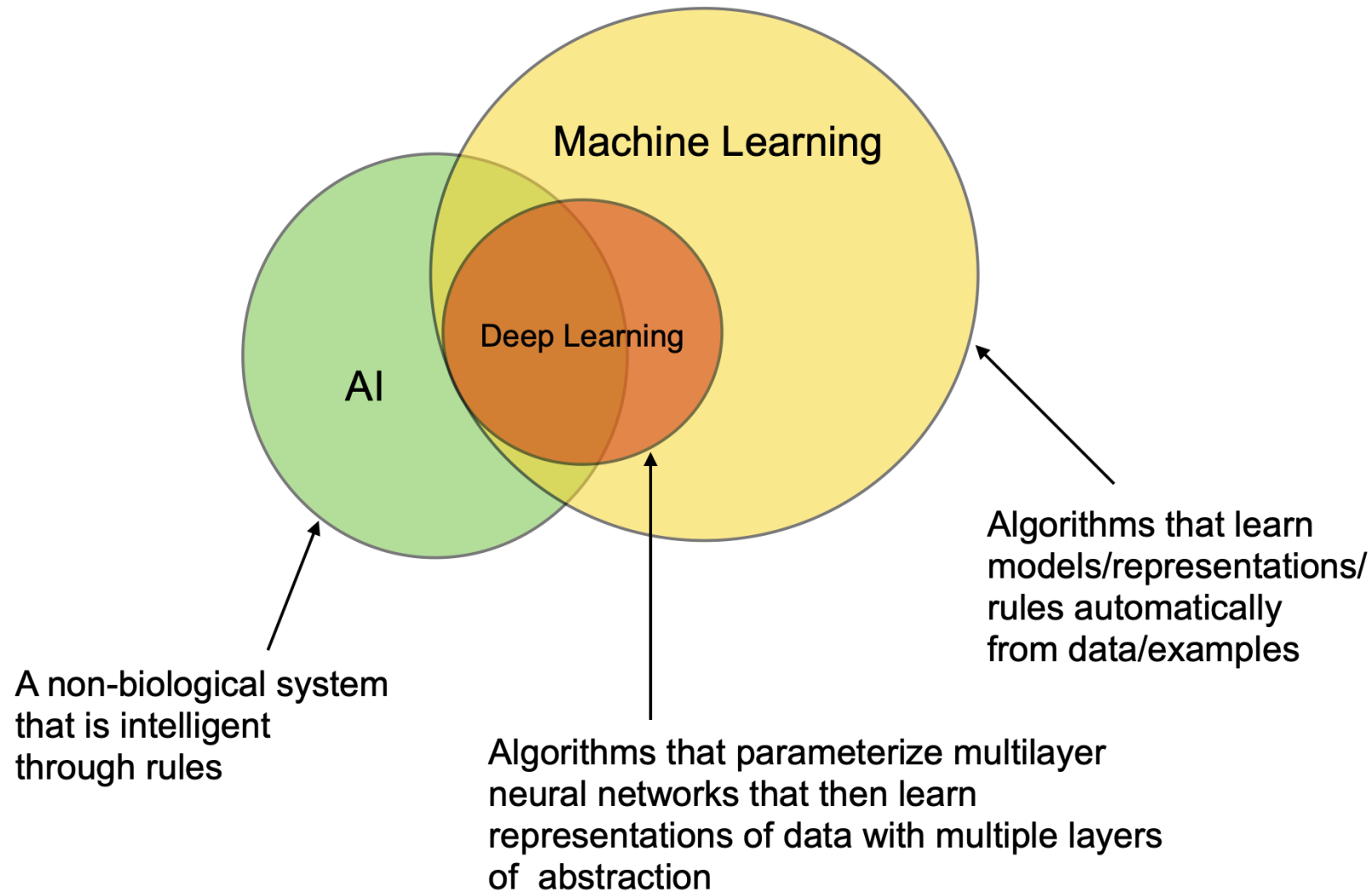
Deep Blue used custom VLSI chips to execute the **alpha-beta search** algorithm in parallel, an example of GOFAI (Good Old-Fashioned Artificial Intelligence).



# What This Course is About



# Example from the Three Related Areas



# Definition of Machine Learning

Formally, a computer program is said to **learn** from experience  $\mathcal{E}$  with respect to some task  $\mathcal{T}$  and performance measure  $\mathcal{P}$  if its performance at  $\mathcal{T}$  as measured by  $\mathcal{P}$  improves with  $\mathcal{E}$ .

# The Fundamental Questions

- Representation
  - How to encode our domain knowledge/assumptions/constraints?
  - How to capture/model uncertainties in possible worlds?
- Inference
  - How do I answer questions/queries according to my model and/or based on observed data?

$$\text{e.g. } P(X_i|D)$$

- Learning
  - What model is "right" for my data?

$$\text{e.g. } M = \operatorname{argmax}_{M \in \mathcal{H}} F(D; M)$$

# The Fundamental Questions in Probabilistic Form

- Representation
  - What is the joint probability distribution of multiple variables?  
 $P(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8)$
  - Assume Boolean  $X_i$ . How many state configurations of the joint probability?
    - $2^8$
  - Can we disallow certain state configurations, and instead focus on a subset?

# The Fundamental Questions in Probabilistic Form

- Inference
  - How do I answer questions/queries according to my model and/or based on observed data?

e.g.  $P(X_i|D)$

- Let's try  $P(X_8|X_1)$ :

$$\begin{aligned} P(X_8|X_1) &= \frac{P(X_8, X_1)}{P(X_1)} \\ &= \frac{\sum_{X_2} \sum_{X_3} \cdots \sum_{X_7} P(X_1, \dots, X_8)}{\sum_{X_2} \sum_{X_3} \cdots \sum_{X_8} P(X_1, \dots, X_8)} \end{aligned}$$

- Require summing over  $2^6$  configurations of unobserved variables
- On the other hand, if  $X_i$  all independent:  $P(X_8|X_1) = P(X_8)$
- Graphical form allows us to work in between.

# The Fundamental Questions in Probabilistic Form

- Learning

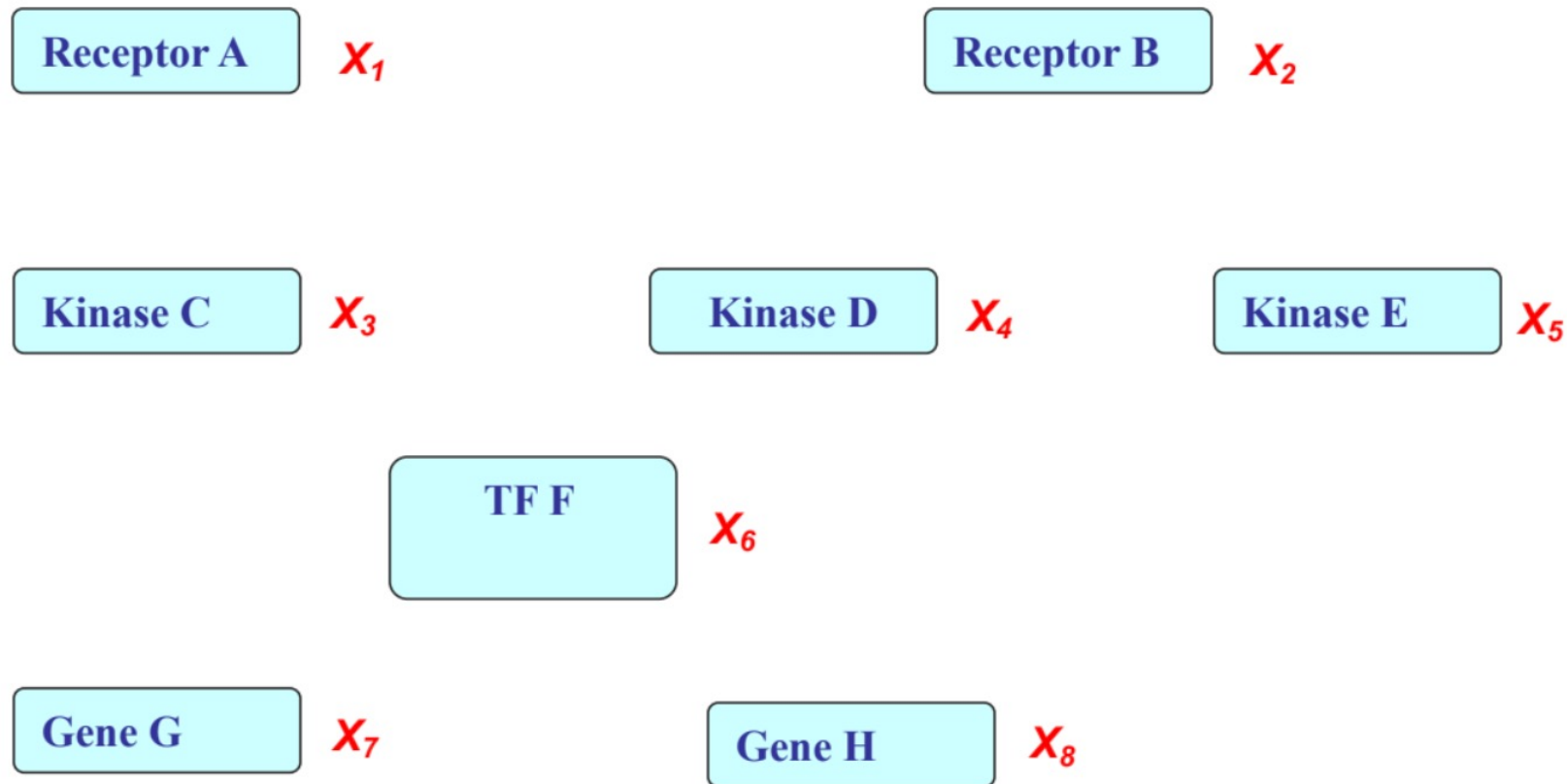
- What model is "right" for my data?

$$\text{e.g. } M = \operatorname{argmax}_{M \in \mathcal{H}} F(D; M)$$

- How can we constrain the hypothesis space  $\mathcal{H}$  so that the search for the argmax is efficient?

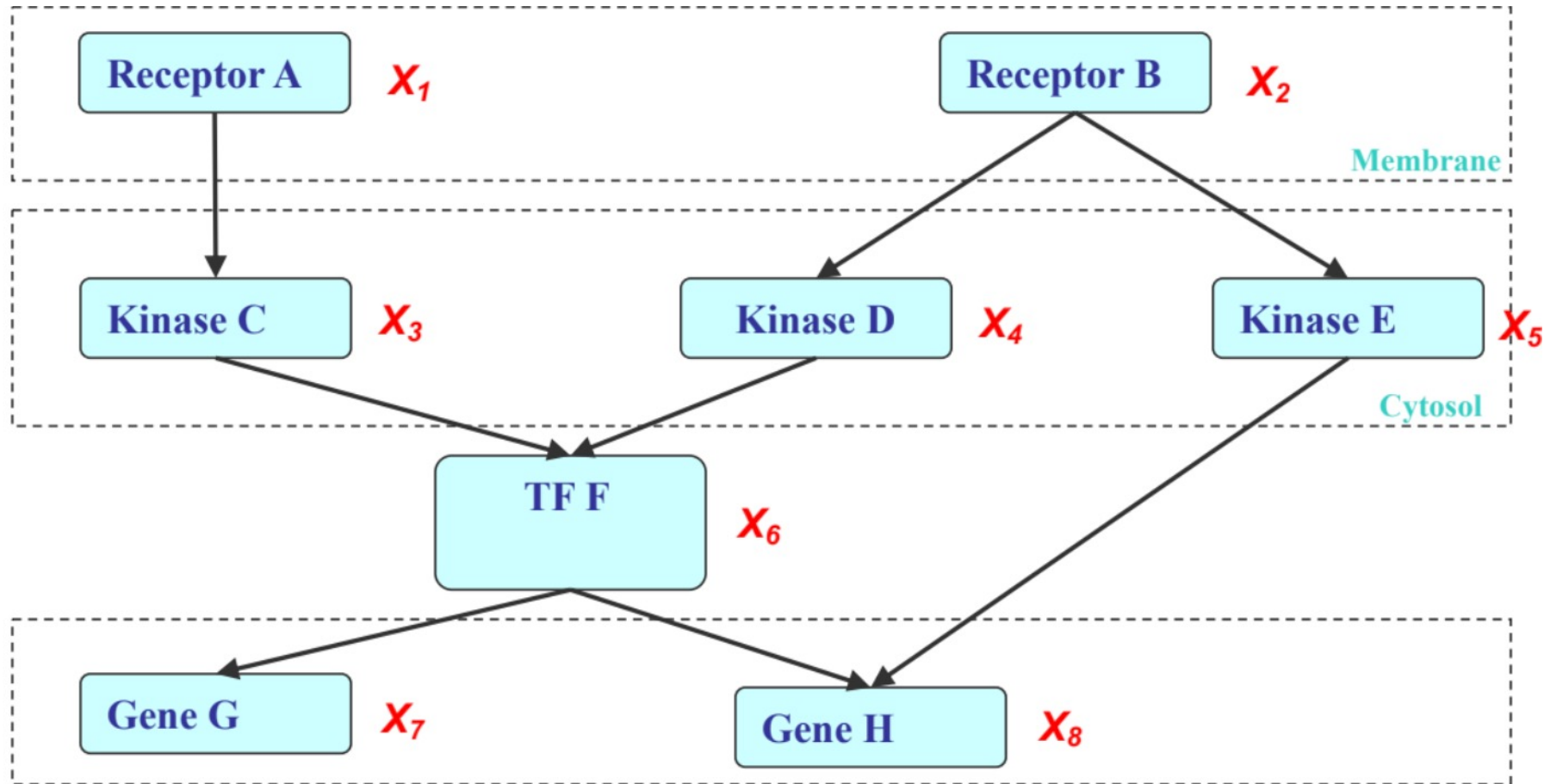
# An example

A possible world for cellular signal transduction:



# An example

Imposing structure of dependencies simplifies representation

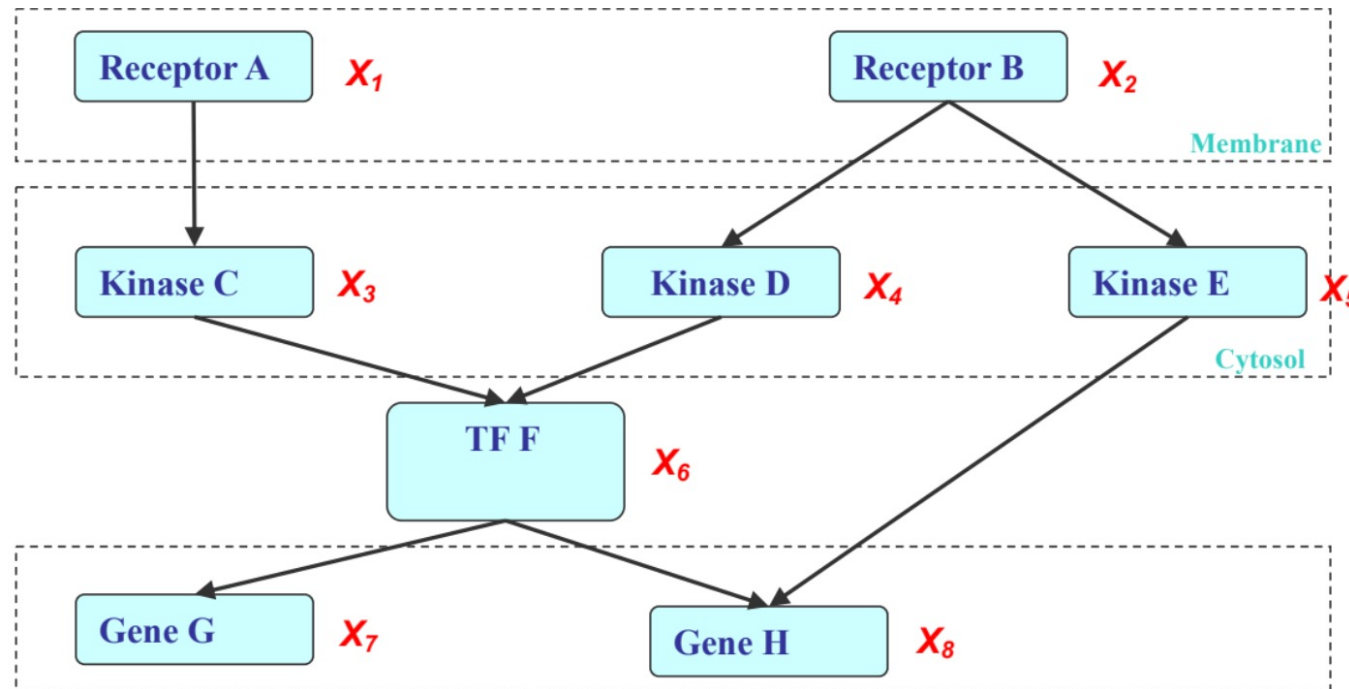


# An example

If the  $X_i$ s are conditional independence (as encoded by a PGM), then the joint can be factorized into a product of simpler terms:

$$P(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8) = P(X_1)P(X_2)P(X_3|X_1)P(X_4|X_2)P(X_5|X_2)P(X_6|X_3, X_4)P(X_7|X_6)P(X_8|X_6, X_5)$$

Reduced from  $2^8$  to 18 parameters!





# Today

---

1. Course overview
2. What is machine learning?
- 3. The broad categories of ML**
4. The supervised learning workflow
5. Necessary ML notation and jargon
6. About the practical aspects and tools

# 3 Broad Categories of ML

## Supervised Learning

- > Labeled data
- > Direct feedback
- > Predict outcome/future

## Unsupervised Learning

- > No labels/targets
- > No feedback
- > Find hidden structure in data

## Reinforcement Learning

- > Decision process
- > Reward system
- > Learn series of actions

**Source:** Raschka and Mirjalily (2019). *Python Machine Learning, 3rd Edition*

# 3 Broad Categories of ML

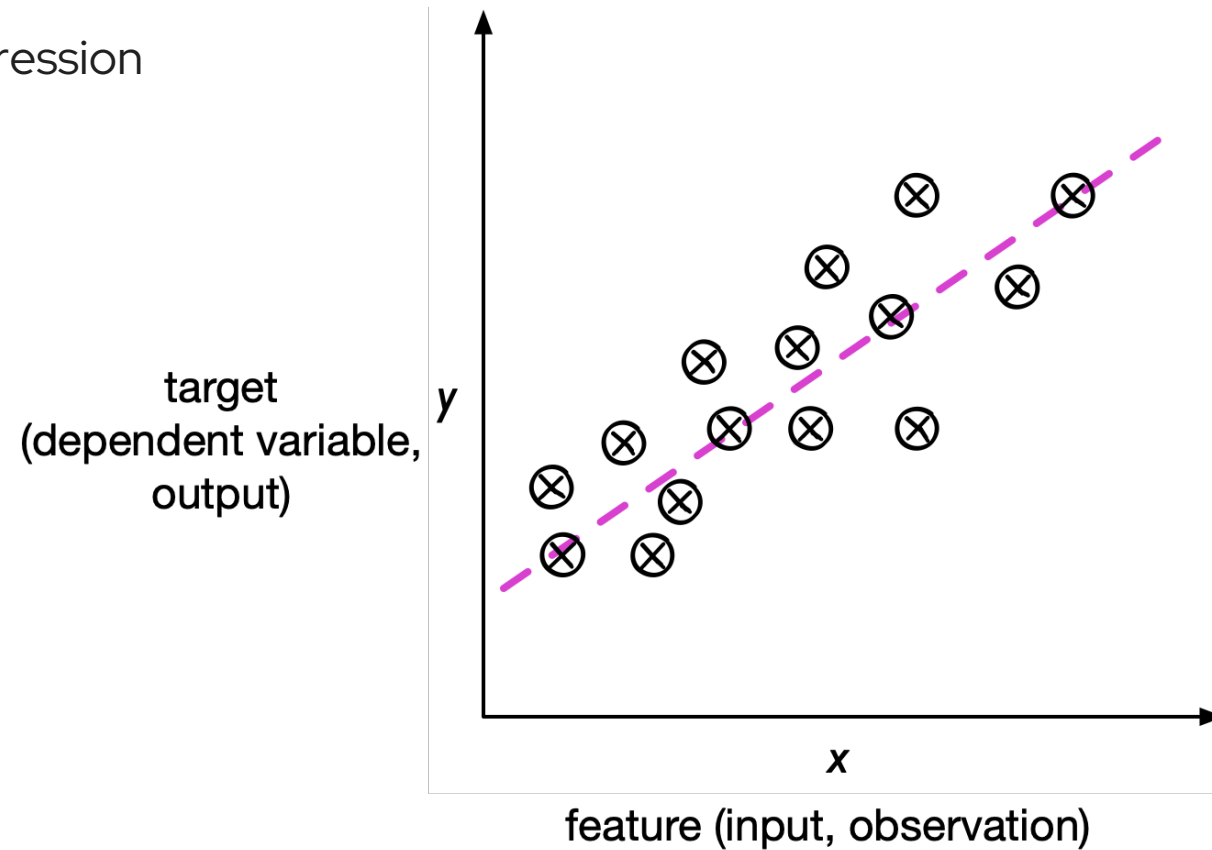
---

## Supervised Learning

- Labeled data
- Direct feedback
- Predict outcome/future

# Supervised Learning

Ex: Regression

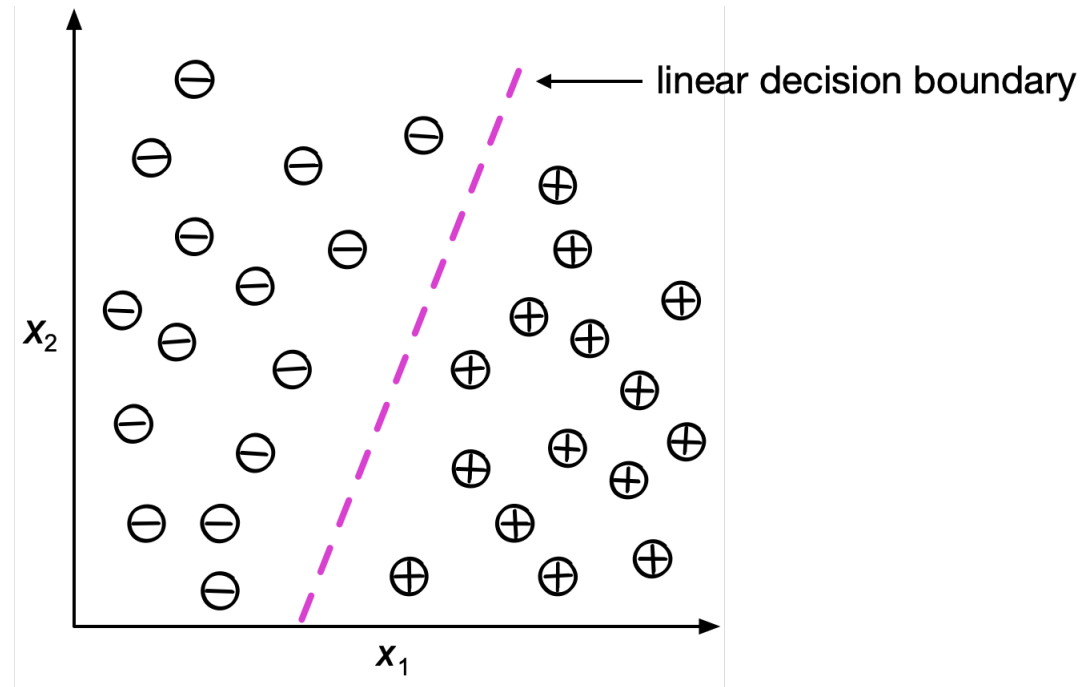


**Source:** Raschka and Mirjalili (2019). *Python Machine Learning, 3rd Edition*

# Supervised Learning

Ex: Classification

**What are the  
class labels (y's)?**



**Source:** Raschka and Mirjalily (2019). Python Machine Learning, 3rd Edition

# Supervised Learning

Recall our prior definition of machine learning:

Formally, a computer program is said to **learn** from experience  $\mathcal{E}$  with respect to some task  $\mathcal{T}$  and performance measure  $\mathcal{P}$  if its **performance at  $\mathcal{T}$  as measured by  $\mathcal{P}$  improves with  $\mathcal{E}$ .**

So **supervised** machine learning is:

- |                               |   |
|-------------------------------|---|
| • Task $\mathcal{T}$ :        | Learn a function $h: \mathcal{X} \rightarrow \mathcal{Y}$ |
| • Experience $\mathcal{E}$ :  | Labeled samples $\{(x_i, y_i)\}_{i=1}^n$                  |
| • Performance $\mathcal{P}$ : | A measure of how good $h$ is                              |

# Unsupervised Learning

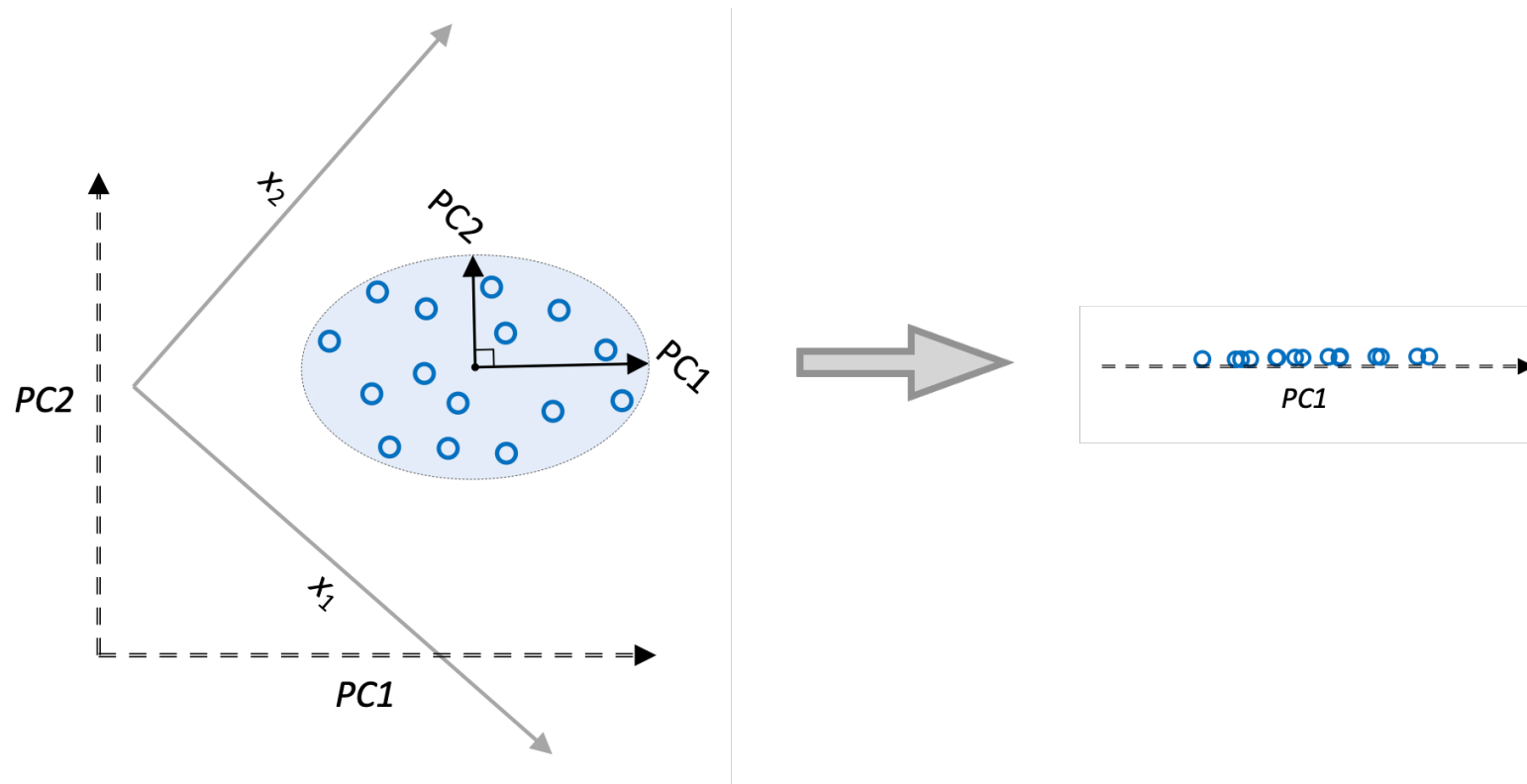
## Unsupervised Learning

- No labels/targets
- No feedback
- Find hidden structure in data

**Source:** Raschka and Mirjalily (2019). Python Machine Learning, 3rd Edition

# Unsupervised Learning

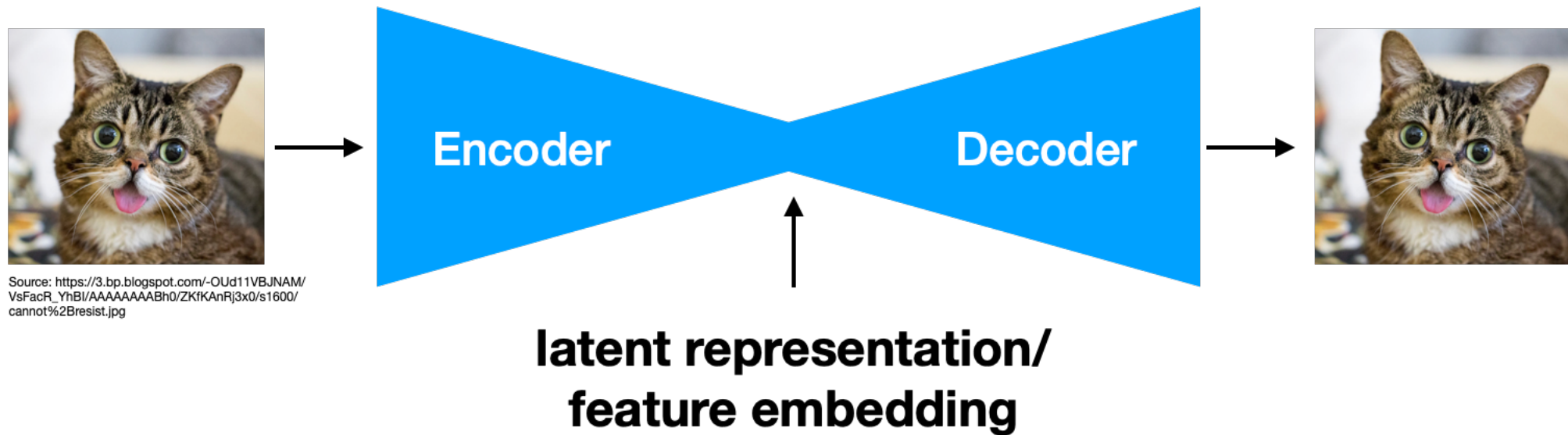
Ex: Representation Learning / Dimensionality Reduction with PCA



Source: Raschka and Mirjalily (2019). Python Machine Learning, 3rd Edition

# Unsupervised Learning

Ex: Representation Learning / Dimensionality Reduction with Autoencoders

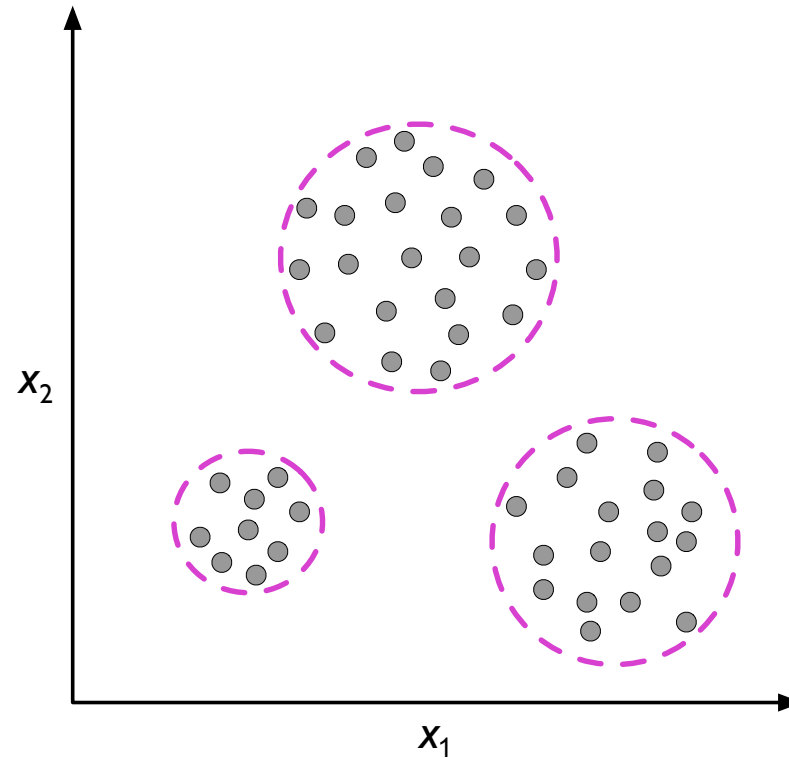


Fun fact: PCA = linear Autoencoder

Source: Raschka and Mirjalily (2019). Python Machine Learning, 3rd Edition

# Unsupervised Learning

Ex: Clustering



**Source:** Raschka and Mirjalily (2019). Python Machine Learning, 3rd Edition

# Unsupervised Learning

Recall our prior definition of machine learning:

Formally, a computer program is said to **learn** from experience  $\mathcal{E}$  with respect to some task  $\mathcal{T}$  and performance measure  $\mathcal{P}$  if its **performance at  $\mathcal{T}$  as measured by  $\mathcal{P}$  improves with  $\mathcal{E}$ .**

So **unsupervised** machine learning is:

- |                               |  |
|-------------------------------|--|
| • Task $\mathcal{T}$ :        | Discover structure in data                   |
| • Experience $\mathcal{E}$ :  | Unlabeled samples $\{\mathbf{x}_i\}_{i=1}^n$ |
| • Performance $\mathcal{P}$ : | Measure of fit or utility                    |

# Reinforcement Learning

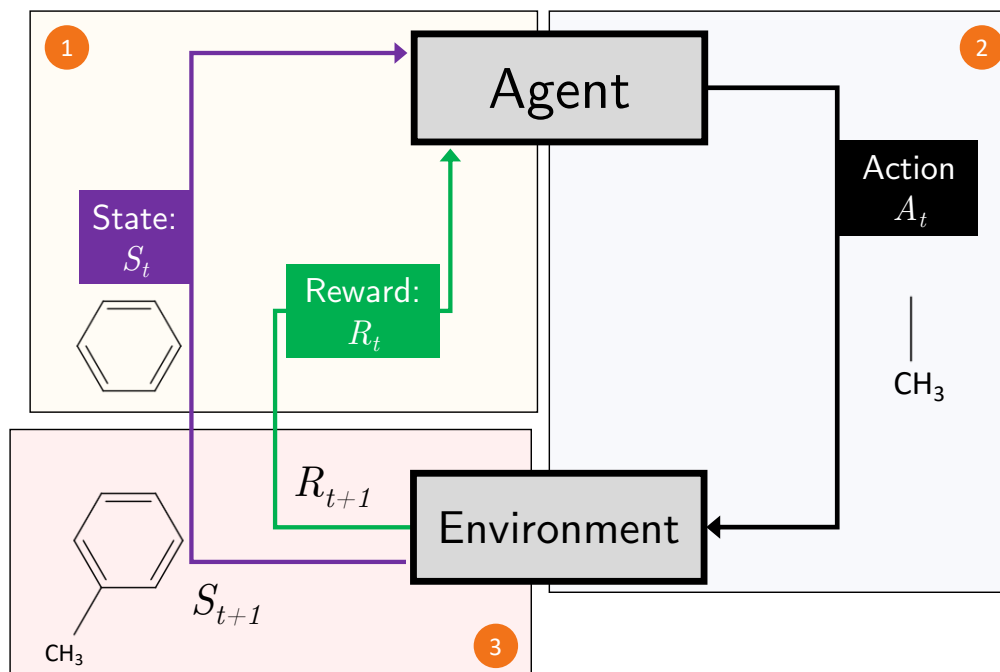


Reinforcement Learning

- > Decision process
- > Reward system
- > Learn series of actions

**Source:** Raschka and Mirjalily (2019). *Python Machine Learning, 3rd Edition*

# Reinforcement Learning

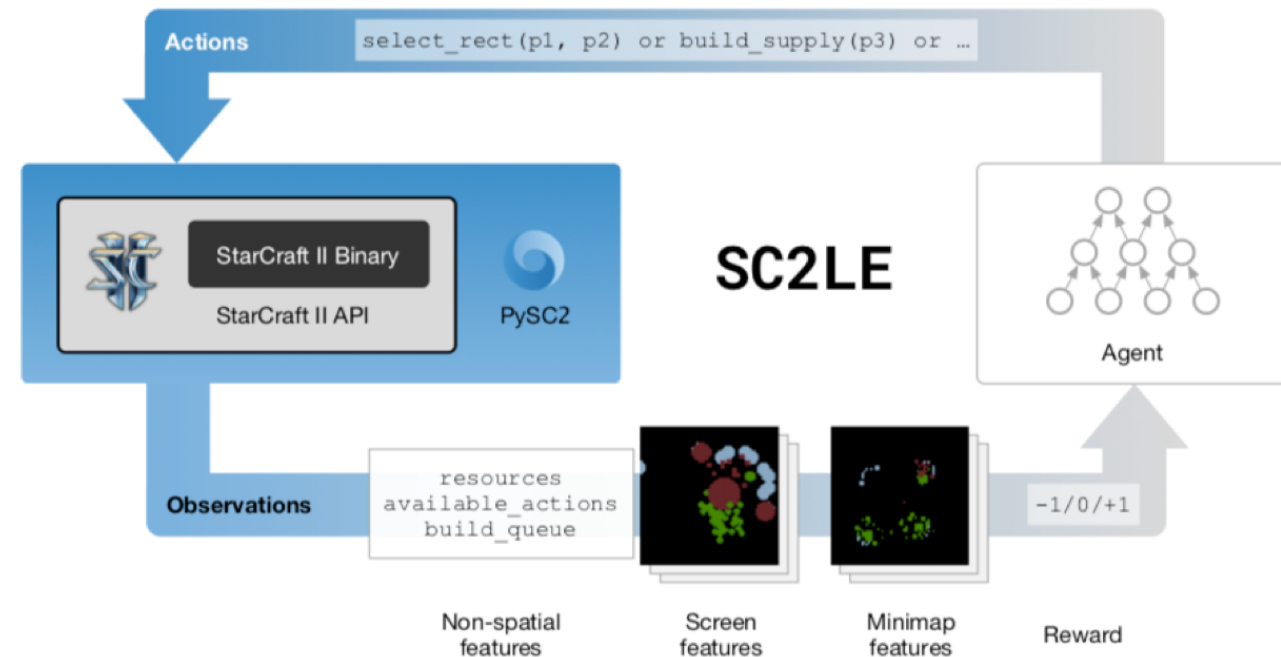


**Figure 5:** Representation of the basic reinforcement learning paradigm with a simple molecular example. (1) Given a benzene ring (state  $S_t$  at iteration  $t$ ) and some reward value  $R_t$  at iteration  $t$ , (2) the agent selects an action  $A_t$  that adds a methyl group to the benzene ring. (3) The environment considers this information for producing the next state ( $S_{t+1}$ ) and reward ( $R_{t+1}$ ). This cycle repeats until the episode is terminated.

Source: Sebastian Raschka and Benjamin Kaufman (2020)

Machine learning and AI-based approaches for bioactive ligand discovery and GPCR-ligand recognition

# Reinforcement Learning



Vinyals, Oriol, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani et al. "Starcraft II: A new challenge for reinforcement learning." *arXiv preprint arXiv:1708.04782* (2017).

# Reinforcement Learning

Recall our prior definition of machine learning:

Formally, a computer program is said to **learn** from experience  $\mathcal{E}$  with respect to some task  $\mathcal{T}$  and performance measure  $\mathcal{P}$  if its **performance at  $\mathcal{T}$  as measured by  $\mathcal{P}$  improves with  $\mathcal{E}$ .**

So **reinforcement** learning is:

- |                               |                                       |
|-------------------------------|---------------------------------------|
| • Task $\mathcal{T}$ :        | Learn a policy $\pi: S \rightarrow A$ |
| • Experience $\mathcal{E}$ :  | Interaction with environment          |
| • Performance $\mathcal{P}$ : | Expected reward                       |

# The 3 Broad Categories of ML

## Supervised Learning

- > Labeled data
- > Direct feedback
- > Predict outcome/future

## Unsupervised Learning

- > No labels/targets
- > No feedback
- > Find hidden structure in data

## Reinforcement Learning

- > Decision process
- > Reward system
- > Learn series of actions

**Source:** Raschka and Mirjalily (2019). *Python Machine Learning, 3rd Edition*

# Semi-Supervised Learning

- Mix between supervised and unsupervised learning
- Some training examples contain outputs, but some don't

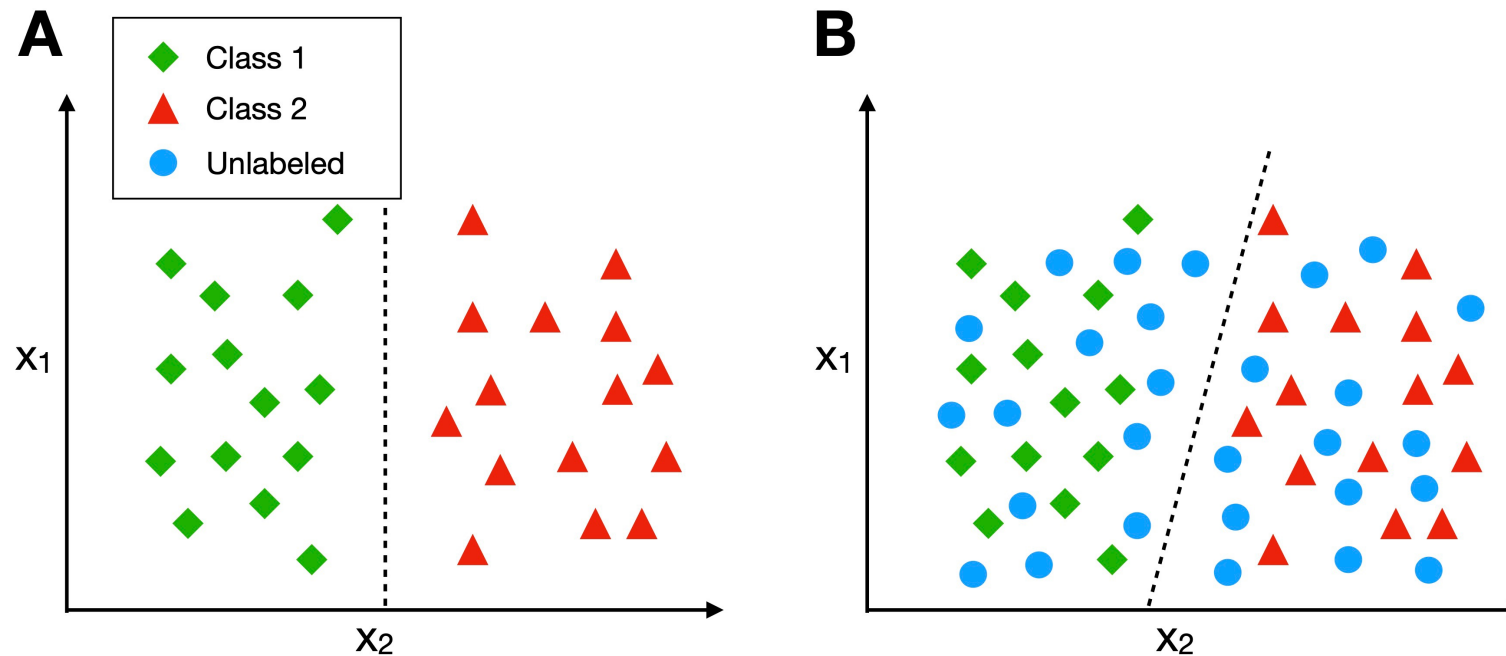
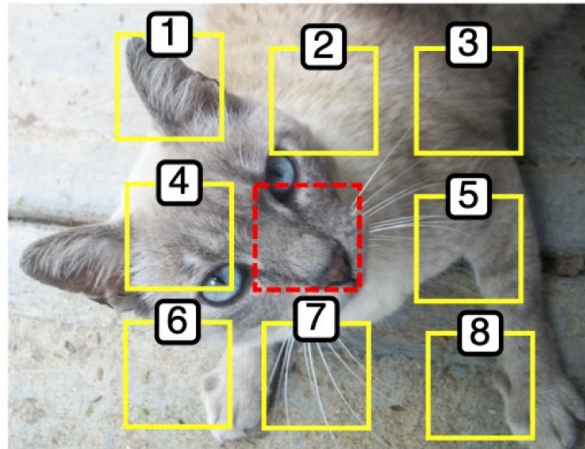


Illustration of semi-supervised learning incorporating unlabeled examples. (A) A decision boundary derived from the labeled training examples only. (B) A decision boundary based on both labeled and unlabeled examples.

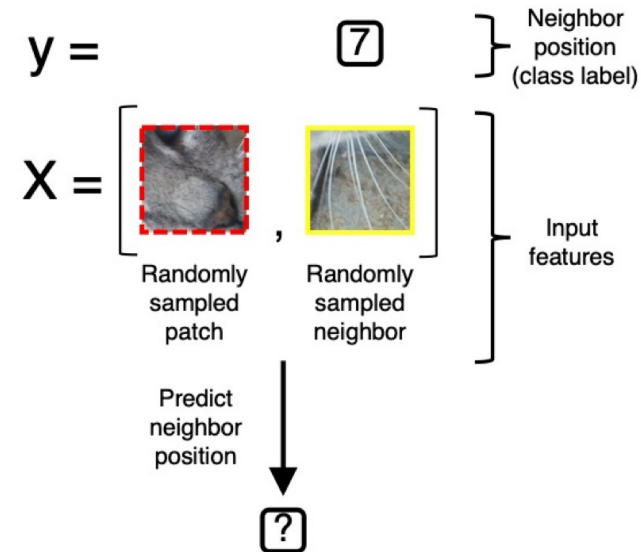
# Self-Supervised Learning

- A process of deriving and using label information directly from the data itself rather than having humans annotating it

A



B



Self-supervised learning via context prediction. (A) A random patch is sampled (red square) along with 9 neighboring patches. (B) Given the random patch and a random neighbor patch, the task is to predict the position of the neighboring patch relative to the center patch (red square).

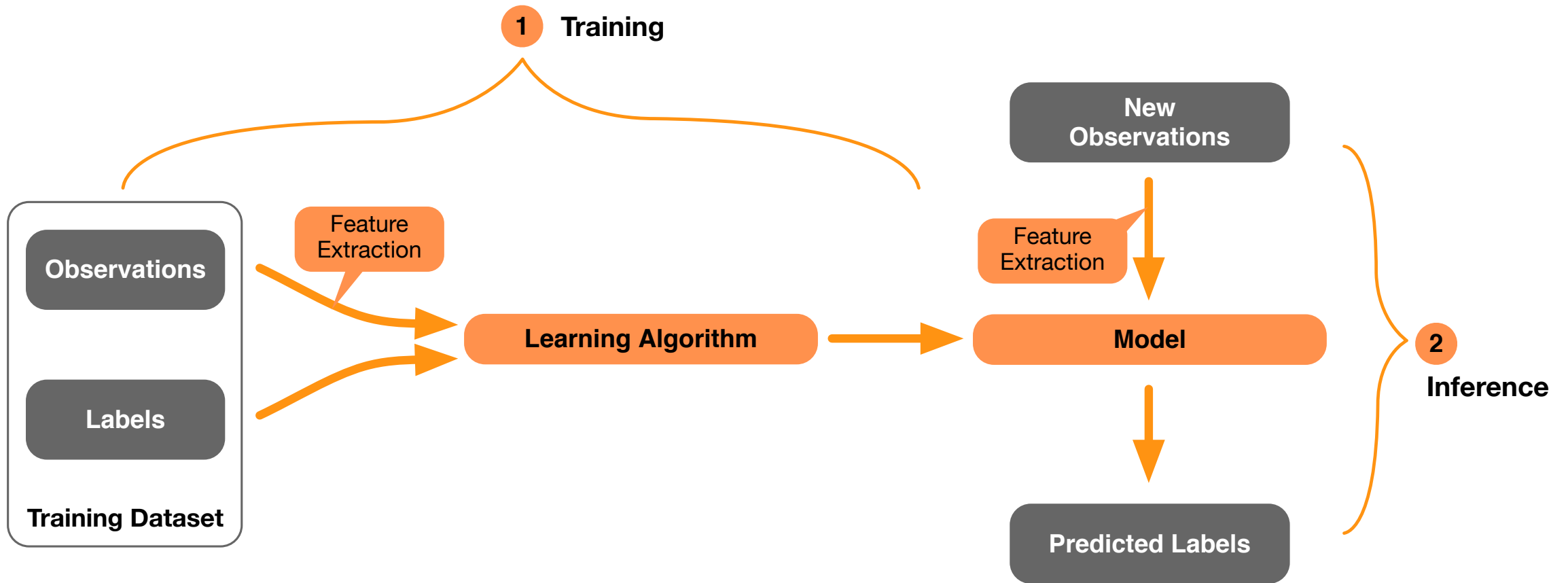


# Today

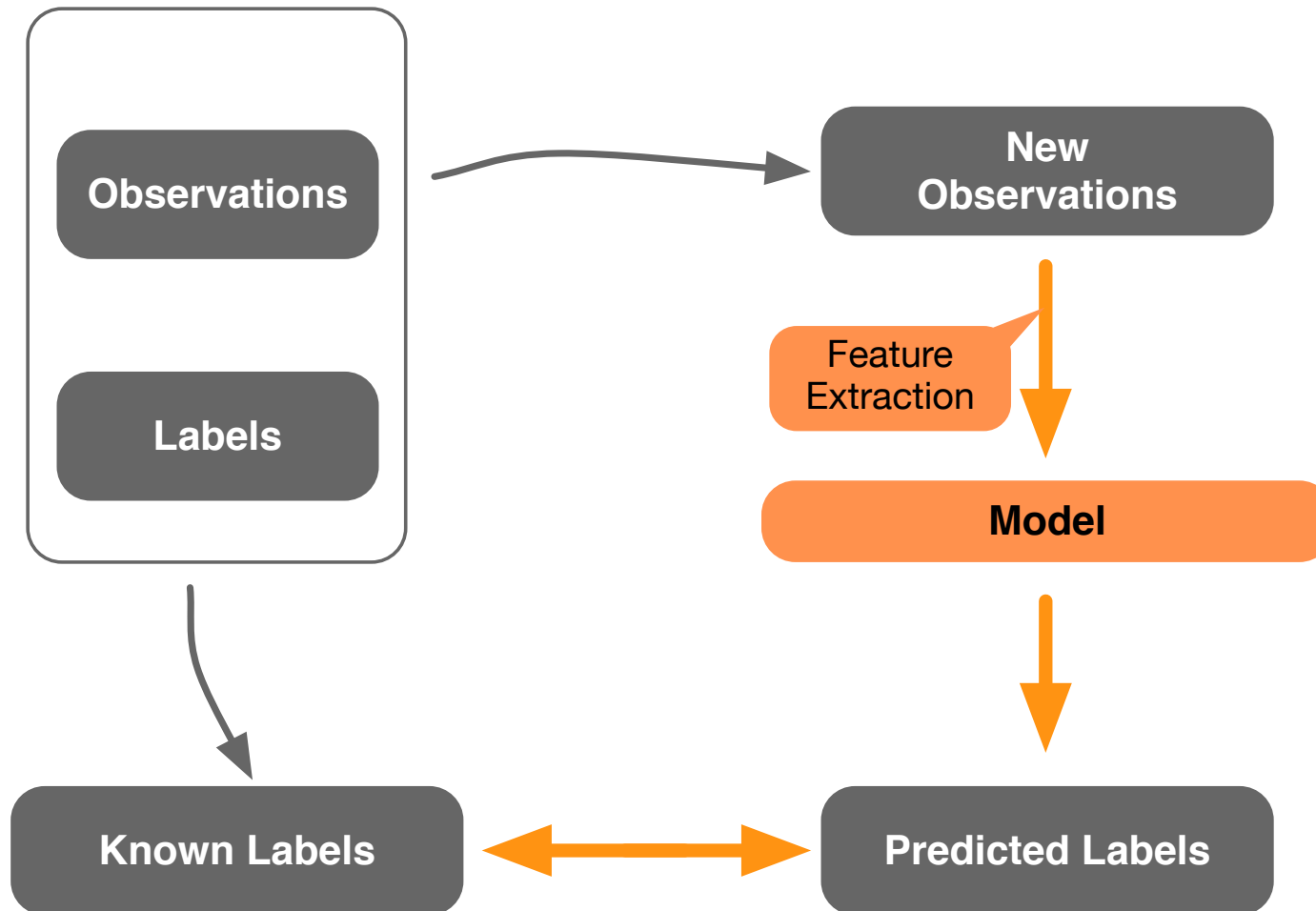
---

1. Course overview
2. What is machine learning?
3. The broad categories of ML
- 4. The supervised learning workflow**
5. Necessary ML notation and jargon
6. About the practical aspects and tools

# Supervised Learning Workflow



# Using a test dataset to evaluate performance



# Machine Learning vs Deep Learning

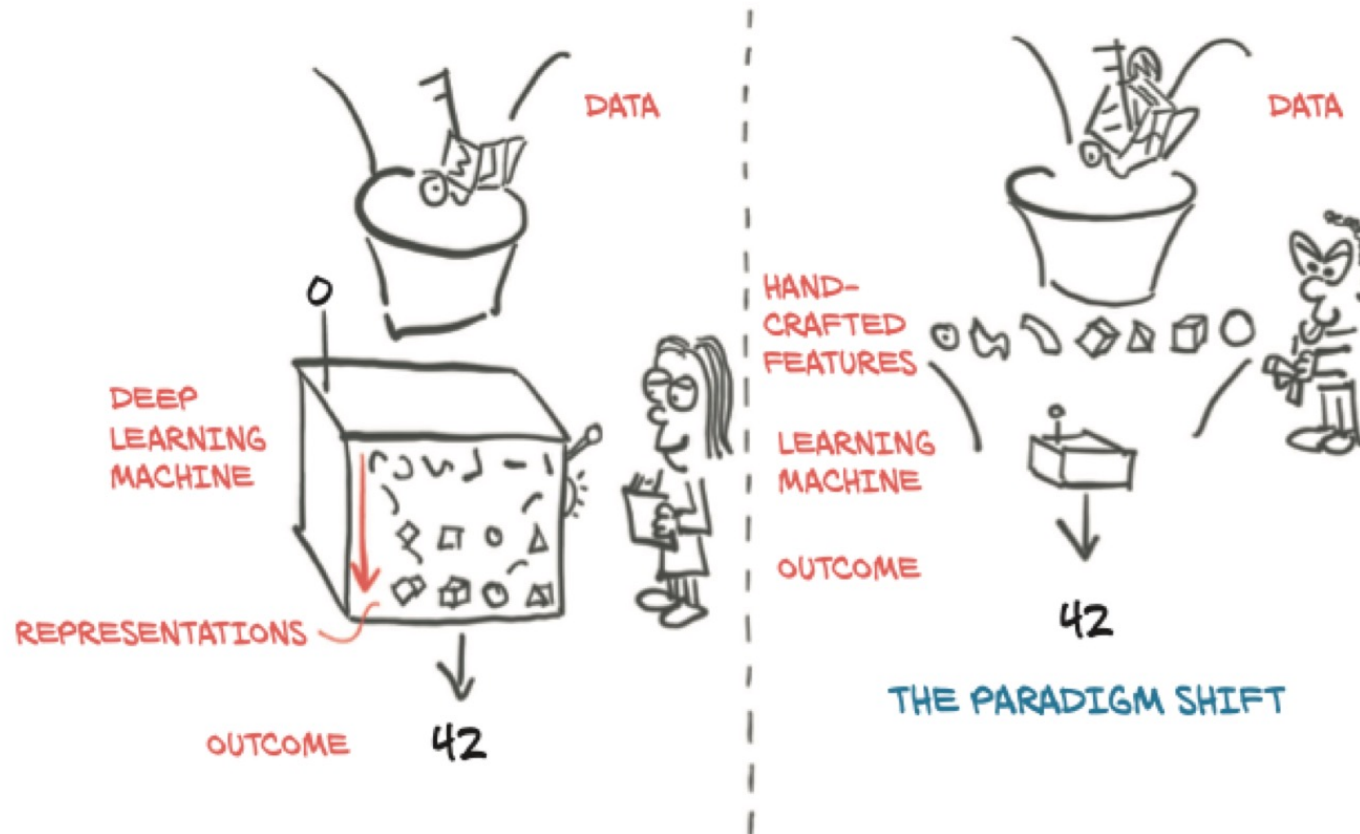


Image source: Stevens et al., Deep Learning with PyTorch. Manning, 2020

# Structured vs Unstructured Data

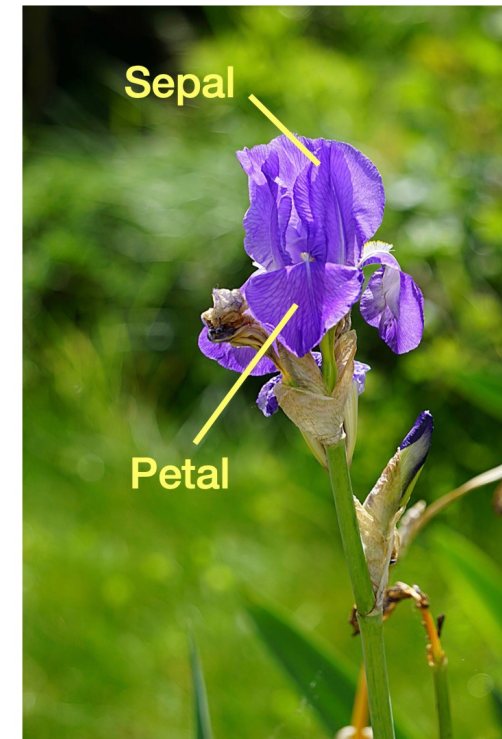
A

Feature vector of the 1st training example

Class label

Index	Sepal length	Sepal width	Petal length	Petal width	Species
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
...	...	...	...	...	...
150	5.9	3	5.1	1.8	Iris-virginica

B





# Today

---

1. Course overview
2. What is machine learning?
3. The broad categories of ML
4. The supervised learning workflow
- 5. Necessary ML notation and jargon**
6. About the practical aspects and tools

# Machine Learning Jargon

---

- **Supervised learning**
  - Learning function to map input  $x$  (features) to output  $y$  (targets)
- **Structured data**
  - Databases, spreadsheets/csv files, etc
- **Unstructured data**
  - Features like image pixels, audio signals, text sentences

# Supervised Learning (More Formal Notation)

Training set:  $D = \{(x^i \ y^i), i = 1, \dots, n\},$

Unknown function:  $f(x) = y$

Hypothesis:  $h(x) = \hat{y}$

Classification

$$h: \mathcal{R}^m \rightarrow \mathcal{Y}, \mathcal{Y} = \{1, \dots, k\}$$

Regression

$$h: \mathcal{R}^m \rightarrow \mathcal{R}$$

# Data Representation

---

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

Feature vector

# Data Representation

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

Feature vector

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix}$$

Feature Matrix / Design Matrix

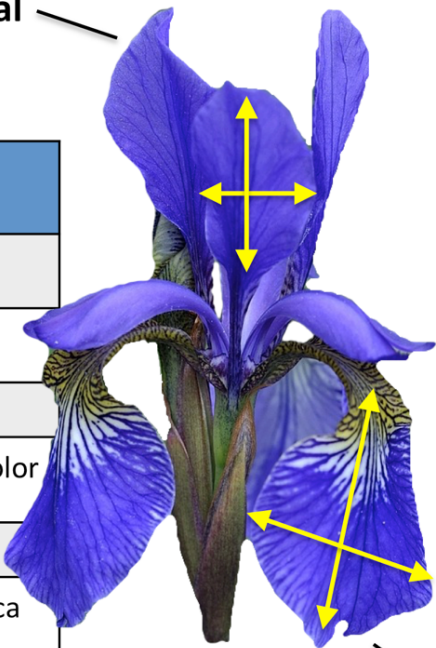
$$\mathbf{X} = \begin{bmatrix} x_1^{[1]} & x_2^{[1]} & \cdots & x_m^{[1]} \\ x_1^{[2]} & x_2^{[2]} & \cdots & x_m^{[2]} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{[n]} & x_2^{[n]} & \cdots & x_m^{[n]} \end{bmatrix}$$

Feature Matrix / Design Matrix

# Data Representation (structured data)

$m =$  \_\_\_\_\_

$n =$  \_\_\_\_\_



	Sepal length	Sepal width	Petal length	Petal width	
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

# Data Representation (unstructured data; images)

## Convolutional Neural Networks

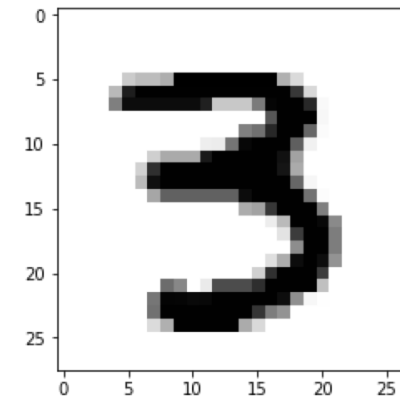
Image batch dimensions: `torch.Size([128, 1, 28, 28])` ← "NCHW" representation (more on that later)

Image label dimensions: `torch.Size([128])`

```
print(images[0].size())
torch.Size([1, 28, 28])
```

```
images[0]
```

```
tensor([[[[0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.5020, 0.9529, 0.9529, 0.9529,
          0.9529, 0.9529, 0.9529, 0.8706, 0.2157, 0.2157, 0.2157, 0.5176,
          0.9804, 0.9922, 0.9922, 0.8392, 0.0235, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.6627, 0.9922, 0.9922, 0.9922, 0.0314, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.4980, 0.5529,
          0.8471, 0.9922, 0.9922, 0.5961, 0.0157, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0667, 0.0745, 0.5412, 0.9725, 0.9922,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000]]]])
```



# Machine Learning Jargon

- **Training a model** = fitting a model = parameterizing a model = learning from data
- **Training example**, synonymous to training record, training instance, training sample (in some contexts, sample refers to a collection of training examples)
- **Feature**, synonymous to observation, predictor, variable, independent variable, input, attribute, covariate
- **Target**, synonymous to outcome, ground truth, output, response variable, dependent variable, (class) label (in classification)
- **Output / Prediction**, use this to distinguish from targets; here, means output from the model



# Today

---

1. Course overview
2. What is machine learning?
3. The broad categories of ML
4. The supervised learning workflow
5. Necessary ML notation and jargon
- 6. About the practical aspects and tools**

# Main Scientific Python Libraries

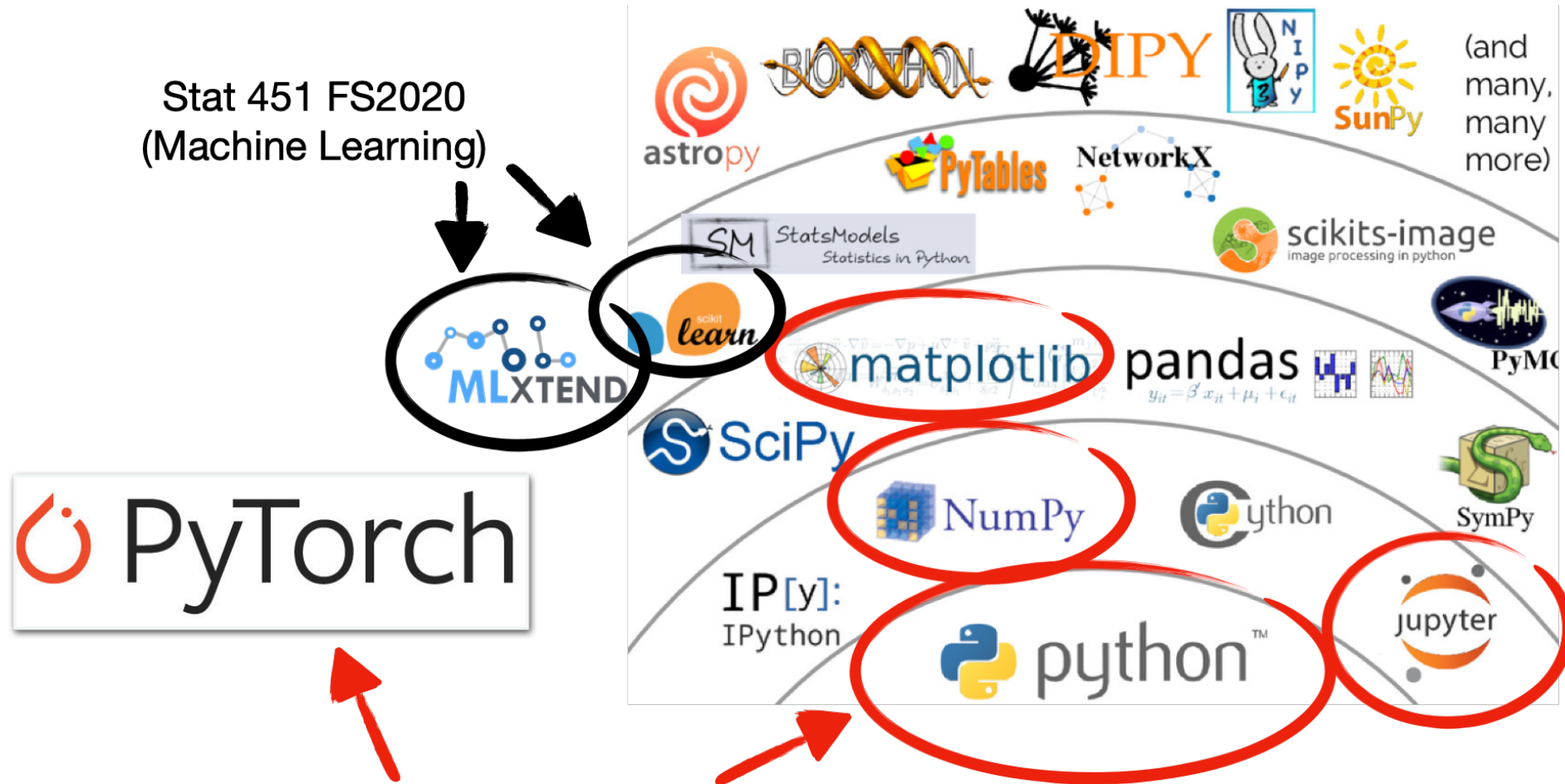
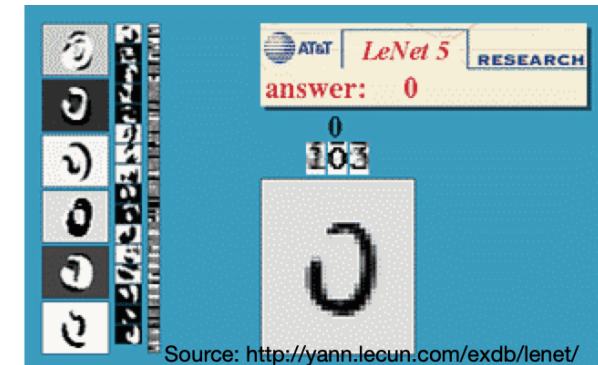
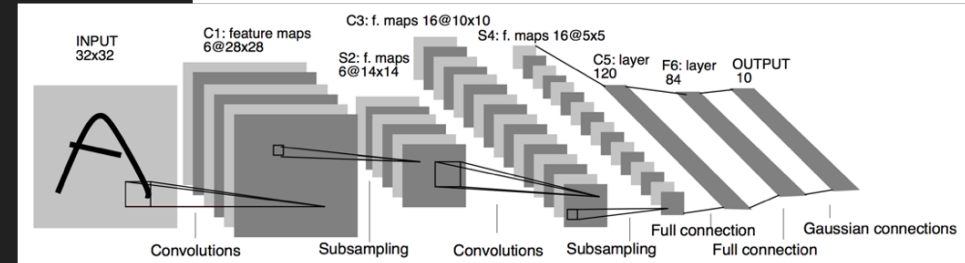


Image by Jake VanderPlas. Source:  
<https://speakerdeck.com/jakevdp/the-state-of-the-stack-scipy-2015-keynote?slide=8>

```

34 #####
35 # ## MODEL
36 #####
37
38 class LeNet5(torch.nn.Module):
39
40     def __init__(self, num_classes):
41         super().__init__()
42
43         self.features = torch.nn.Sequential(
44             torch.nn.Conv2d(1, 6, kernel_size=5),
45             torch.nn.Tanh(),
46             torch.nn.MaxPool2d(kernel_size=2),
47             torch.nn.Conv2d(6, 16, kernel_size=5),
48             torch.nn.Tanh(),
49             torch.nn.MaxPool2d(kernel_size=2)
50         )
51
52         self.classifier = torch.nn.Sequential(
53             torch.nn.Linear(16*5*5, 120),
54             torch.nn.Tanh(),
55             torch.nn.Linear(120, 84),
56             torch.nn.Tanh(),
57             torch.nn.Linear(84, num_classes),
58         )
59
60     def forward(self, x):
61         x = self.features(x)
62         x = torch.flatten(x, 1)
63         logits = self.classifier(x)
64         probas = torch.nn.functional.softmax(logits, dim=1)
65         return logits, probas
66
67

```

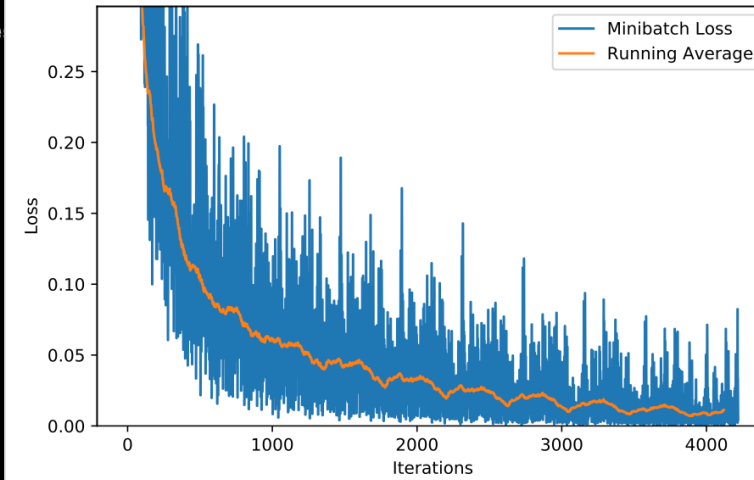


<https://code.visualstudio.com>

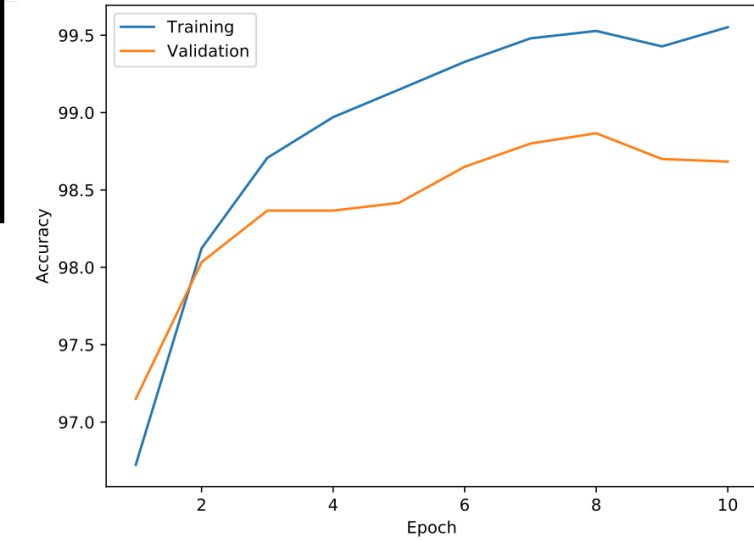
<https://github.com/rasbt/stat453-deep-learning-ss21/tree/main/L01/code>

```
(base) raschka@lambda-quad:~/code/stat453-ss21-exp$ python simple_cnn.py
PyTorch version: 1.7.0
Using cuda:0
[W Context.cpp:69] Warning: torch.set_deterministic is in beta, and its de
on operator())
Epoch: 001/010 | Batch 0000/0422 | Loss: 2.2935
Epoch: 001/010 | Batch 0050/0422 | Loss: 0.5462
Epoch: 001/010 | Batch 0100/0422 | Loss: 0.3154
Epoch: 001/010 | Batch 0150/0422 | Loss: 0.2551
Epoch: 001/010 | Batch 0200/0422 | Loss: 0.1792
Epoch: 001/010 | Batch 0250/0422 | Loss: 0.2210
Epoch: 001/010 | Batch 0300/0422 | Loss: 0.1551
Epoch: 001/010 | Batch 0350/0422 | Loss: 0.2155
Epoch: 001/010 | Batch 0400/0422 | Loss: 0.2306
Epoch: 001/010 | Train: 96.72% | Validation: 97.15%
Time elapsed: 0.09 min
Epoch: 002/010 | Batch 0000/0422 | Loss: 0.1028
Epoch: 002/010 | Batch 0050/0422 | Loss: 0.1167
Epoch: 002/010 | Batch 0100/0422 | Loss: 0.0660
Epoch: 002/010 | Batch 0150/0422 | Loss: 0.1024
Epoch: 002/010 | Batch 0200/0422 | Loss: 0.0847
Epoch: 002/010 | Batch 0250/0422 | Loss: 0.0905
Epoch: 002/010 | Batch 0300/0422 | Loss: 0.1024
Epoch: 002/010 | Batch 0350/0422 | Loss: 0.0719
Epoch: 002/010 | Batch 0400/0422 | Loss: 0.1302
Epoch: 002/010 | Train: 98.12% | Validation: 98.03%
Time elapsed: 0.18 min
Epoch: 003/010 | Batch 0000/0422 | Loss: 0.0720
Epoch: 003/010 | Batch 0050/0422 | Loss: 0.0984
Epoch: 003/010 | Batch 0100/0422 | Loss: 0.0373
Epoch: 003/010 | Batch 0150/0422 | Loss: 0.0685
Epoch: 003/010 | Batch 0200/0422 | Loss: 0.0511
Epoch: 003/010 | Batch 0250/0422 | Loss: 0.0617
Epoch: 003/010 | Batch 0300/0422 | Loss: 0.0748
```

```
Epoch: 010/010 | Batch 0000/0422 | Loss: 0.0095
Epoch: 010/010 | Batch 0050/0422 | Loss: 0.0113
Epoch: 010/010 | Batch 0100/0422 | Loss: 0.0135
Epoch: 010/010 | Batch 0150/0422 | Loss: 0.0028
Epoch: 010/010 | Batch 0200/0422 | Loss: 0.0019
Epoch: 010/010 | Batch 0250/0422 | Loss: 0.0049
Epoch: 010/010 | Batch 0300/0422 | Loss: 0.0132
Epoch: 010/010 | Batch 0350/0422 | Loss: 0.0114
Epoch: 010/010 | Batch 0400/0422 | Loss: 0.0270
Epoch: 010/010 | Train: 99.55% | Validation: 98.68%
Time elapsed: 0.88 min
Total Training Time: 0.88 min
Test accuracy 98.66%
```



Epochs



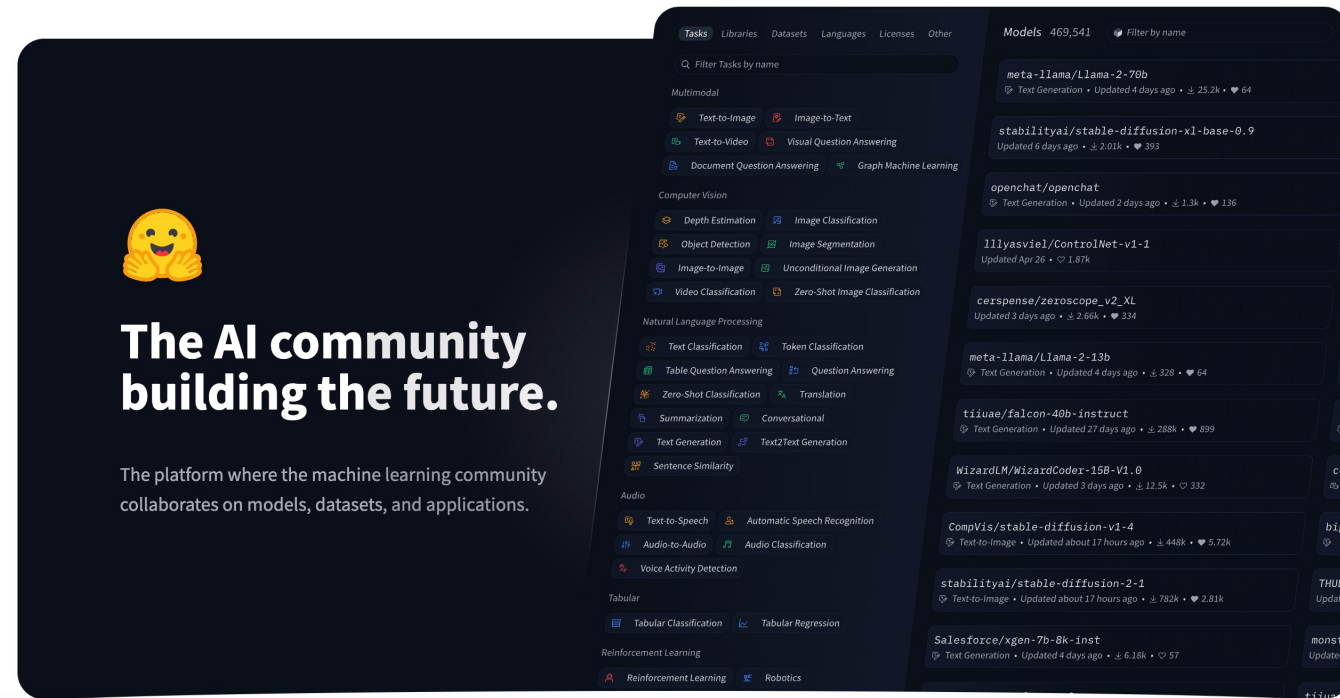
# Further Resources and Reading Materials

- "Introduction to Machine Learning and Deep Learning", article <https://sebastianraschka.com/blog/2020/intro-to-dl-ch01.html>
- STAT451 FS2021: Intro to machine Learning, lecture notes: [https://github.com/rasbt/stat451-machine-learning-fs20/blob/master/L01/01-ml-overview\\_\\_notes.pdf](https://github.com/rasbt/stat451-machine-learning-fs20/blob/master/L01/01-ml-overview__notes.pdf)
- *Python Machine Learning*, 3rd Ed. Packt 2019. Chapter 1.

# Many more open-sourced models and libraries

- Huggingface, especially language-model related:

[huggingface.co](https://huggingface.co)



Questions?

