Machine Learning
Lecture 1

Yang Yuan

Today’s plan

- Course Logistics
- History of AI
- Supervised learning
- Unsupervised learning
- Semi-Supervised learning

What's my teaching goal for this course?

- My goal includes:
  - Make sure you are equipped with basic knowledge of ML
  - Speak ML language
  - Understand the ML landscape
  - Ready to do research
  - Ready to take future courses
    - Deep learning/CV/NLP
    - ……
- Not:
  - Help you become a fantastic machine learning engineer

What will be taught in this course?

- The elements and their connections/relationships
- A map for machine learning world
  - So you'll know what's inside
  - What are all the elements (roughly)
  - If there's anything new, where to put it
- What -> Why -> How
  - I care more on whether you understand what this tool is
  - Less on whether you know how to use this tool (this is supposed to be easy)
  - **If you really understand what, you may use it, modify it, or invent new**

What will not be taught in this course?

- How to write programs to solve real world problems
  - You will have (relatively simple) coding homework, but that's it
  - This course focuses on the conceptual part
- Too much specific details
  - I first list the continents: Asia、Europe、America、Africa、Oceania. You want to know the countries in Asia
  - I then list the countries in Asia: China, India, Japan, Thailand, …… You ask for provinces in China.
  - I then list the provinces in China: Jiangsu, Guangdong, Henan, …… You ask for cities in Sichuan.
  - I then say, stop! That's too much details. If you are interested, you may dig further: city -> district -> road -> community -> building -> layer -> apartment -> room -> person …… I only plan to teach the countries in this course.

Note on exam & homework

- What is the teaching goal of your high school teacher?
- (mostly) clear:
  - Help you pass the college entrance exam
- If that's the goal
  - Final exams become rehearsals of the entrance exam
  - Homework become rehearsals of final exams
- If you are ML models, ideally, homework/final exams/entrance exam are training/validation/test datasets
  - **All from the (almost) same distribution**

Note on exam & homework

- I am a teacher at Tsinghua, not in your high school
  - I am not interested in your GPA
  - Or, about 30% of you get A/A+, I don't care who they are
- I care about whether you've learned what I wanted to teach
- I am an actor with a small tool set
  - Lecture, homework, coding, exams

- My challenge is: how to use these tools to help you learn better?

Note on exam & homework

- Lectures: present thought flows for different notions, connect them together; present proofs, emphasize the important details
  - talking to you
- Homework: let you get your hands dirty, better understand some of the materials taught in class.
  - assigning (difficult) tasks to you
  - **Not a rehearsal for the final exam!**
- Coding: make sure you know how to run ssh, play GPU and use git
  - making you feel you can do anything you want with access to github + server

What is the goal of exams?

- Two goals:
  - Evaluation (secondary)
    - I need to give proper grade for everyone, so I need to evaluate what you have learned
    - Since this is my **secondary goal**, I don't want to let you train with homework
  - Review of the materials, recap what was taught (**primary goal**)
    - I want to make sure you understand the materials as much as possible, equipped with basic knowledge of ML
- Not: see who is smart enough to solve really hard problems with the tools taught in class, during the exams
- Time constraint
  - I only have 1.5 hours for mid-term, about 2 hours for final

Strategy for designing homework/exams

- Accurate evaluation?
  - The whole landscape has n questions to ask, $q_1, \cdots, q_n$
    - Each student j may be able to solve q_i with probability p_j^i
    - Assume p_j^i are independent with respect to i
    - Want to estimate the grade $\mathrm{p_j} = E_i[p_j^i]$
    - We may pick a set of questions $\{q_i\}_{\{i \in I\}}$ as exam questions, what's the best strategy?
  - Concentration inequality tells us
    - Larger $|I|$ gives us more accurate estimation for p_j
    - But each q_i has different answering time! Time constraint for exams
    - So I tend to pick q_i that are "quick" for exams
  - Leave "slow" questions for homework, **different distribution!!**
  - Moreover, push you review everything you learned in class

Speed problem

- Many people complain I was talking too fast
- Solution
  - 雨课堂 advised by Wanrong He in Nov 2020
  - Anonymous vote, to check whether we need to go back

投票 最多可选1项

设置

Is it clear? Do we need to go back?

A It is clear

B Let us go back

提交

Course Logistics

- Class
  - Monday 15:20 - 16:55
  - Wednesday 13:30 - 15:05
  - Room 6A215
- Office hour
  - Tuesday, 9am-10am, by appointment
  - FIT 4-6007
- TA:
  - Yifan Zhang
  - Zhenru Lin

Course Logistics

- Grading:
  - 20% for the homework + coding.
  - 35% for the mid-term.
  - 45% for the final.
- This year’s ppts & papers
  - https://cloud.tsinghua.edu.cn/d/60f1966420b7474a82ec/

Course Logistics

- Homework: (12%)
  - Once every 1-2 weeks
  - Homework should be written in Latex!
- Coding: (8%)
  - Once every 1-2 weeks
  - Python + GPU + github
- About GPU
  - We have 20 2080TI for you to use (Course GPU)
  - Everyone can access 1 machine / 4 GPU
  - Send your public key and username to TA, use your private key to log in

Homework late policy

- After submission deadline (both coding & problem sets), grading discount by 0.95/day
- After 10 days, you get ~0.59 discount

TA's job

- Set assignment (problem sets, coding homework)
- Grading homework + exams
- Recitation class
- Answer questions
  - TA should answer your question within 3 working days
  - Do not expect TA answer your question immediately at 2am!

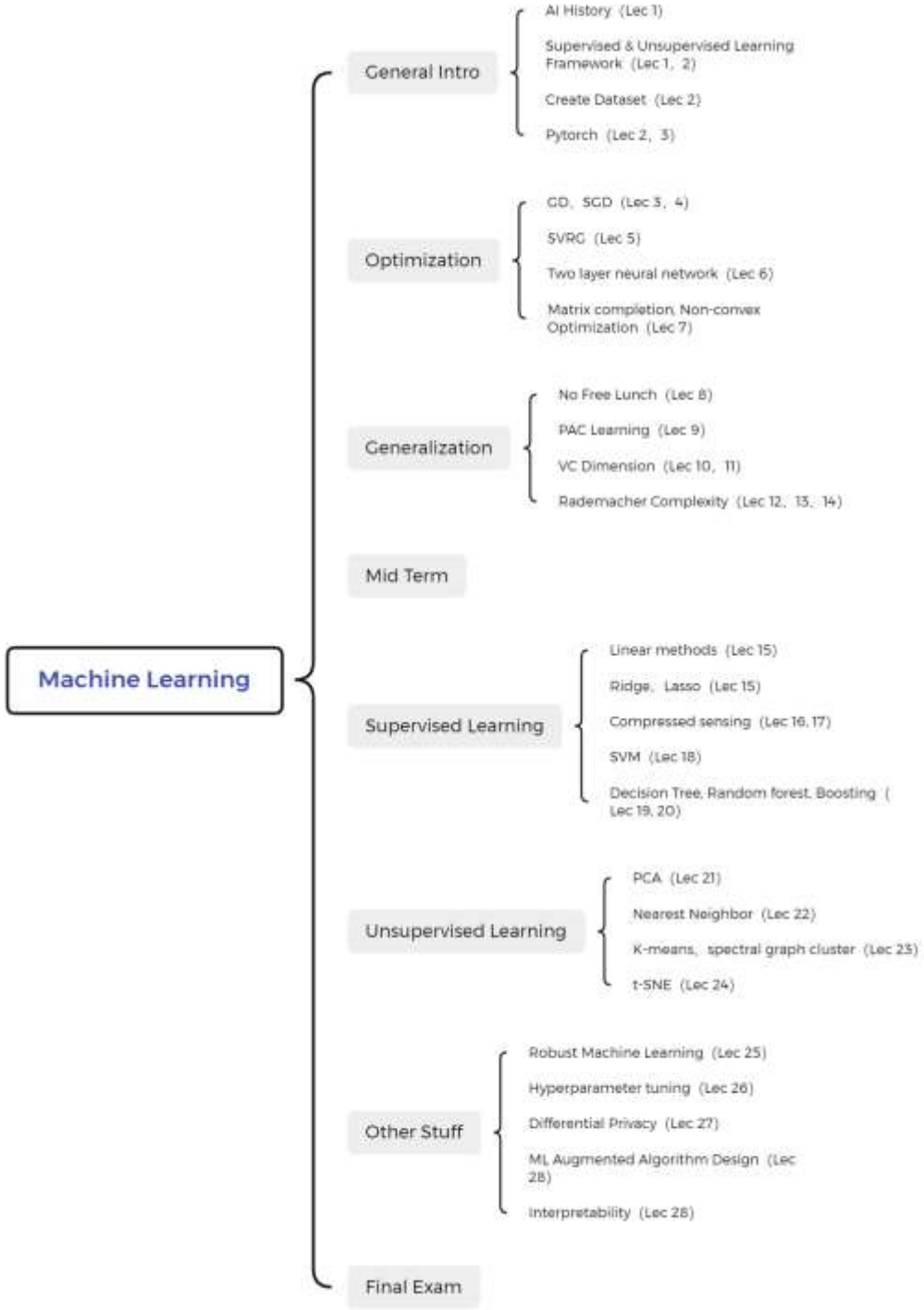
About asking questions

- Do not directly ask me questions through Wechat
  - Too much workload for me
- You may:
  - Ask TA during recitation
    - If TA cannot answer, I can help
  - Make office hour appointment with me
  - Or ask questions in wechat group (I only answer the same question once)

On Prerequisites

- I assume all of you know the basics on Analysis, Algebra and Probability
- If you don’t have the prerequisites, I recommend to drop the course
- But if you insist, do not ask questions on the basics
  - **I may refuse to answer**
  - We have 80 students in class, if I spend 5 minutes answering a question on the basics, that’s wasting 400 minutes for everyone.
- Similarly, I may refuse to answer the same question again in class (unless it is really confusing)

Teaching plan

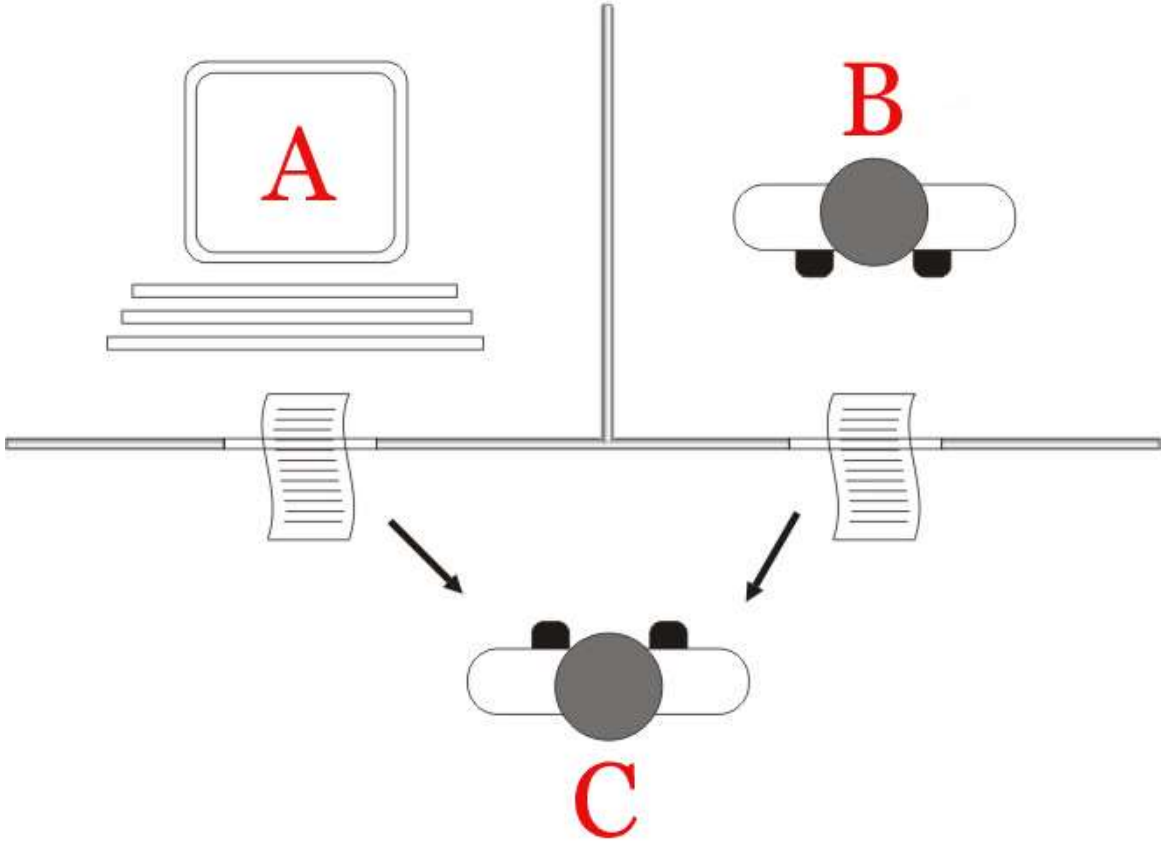


Course Logistics

- Questions?
- Comments?

History of AI

- 1950, Turing test
  - I propose to consider the question, “Can machines think?”
  - “thinking” is difficult to define, Turing chooses new question:
  - **“Are there imaginable digital computers which would do well in the imitation game?”**
  - C needs to distinguish A and B
  - **Not the golden rule!**
    - No longer used for testing AI



SYSTEM PROMPT (HUMAN-WRITTEN)	*In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.*
MODEL COMPLETION (MACHINE-WRITTEN, 10 TRIES)	The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science. Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved. Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow. Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez. Pérez and his friends were astonished to see the unicorn herd. These creatures could be seen from the air without having to move too much to see them – they were so close they could touch their horns. While examining these bizarre creatures the scientists discovered that the creatures also spoke some fairly regular English. Pérez stated, "We can see, for example, that they have a common 'language,' something like a dialect or dialectic."

History of AI

- 1951, Marvin Minsky designed SNARC, a randomly connected network of about 40 Hebb synapses.
  - Inspired by biology
  - First “artificial neural network”
- 1951, Christopher Strachey wrote a checkers program and Dietrich Prinz wrote one for chess
  - Then Game AI becomes a measure for intelligence
- 1956, John McCarthy coined the name **“Artificial Intelligence”**
- 1955, Allen Newell and Herbert Simon created “Logic Theorist” for proving theorems.

History of AI

- 1956, Dartmouth conference, **the birth of AI**, as the name “Artificial intelligence” was accepted
- Golden years: 1956-1974
  - Search algorithms via path elimination
  - NLP: First chatterbot ELIZA
  - Robotics: stack blocks, walk with lower limbs
  - …
- Lots of money were invested

Optimistic Predictions

- Predictions:
  - 1958, H. A. Simon and Allen Newell: "within ten years a digital computer will be the world's chess champion" and "within ten years a digital computer will discover and prove an important new mathematical theorem."
  - 1965, H. A. Simon: "machines will be capable, within twenty years, of doing any work a man can do."
  - 1967, Marvin Minsky: "Within a generation ... the problem of creating 'artificial intelligence' will substantially be solved."
  - 1970, Marvin Minsky (in Life Magazine): "In from three to eight years we will have a machine with the general intelligence of an average human being."

First AI winter 1974-1980

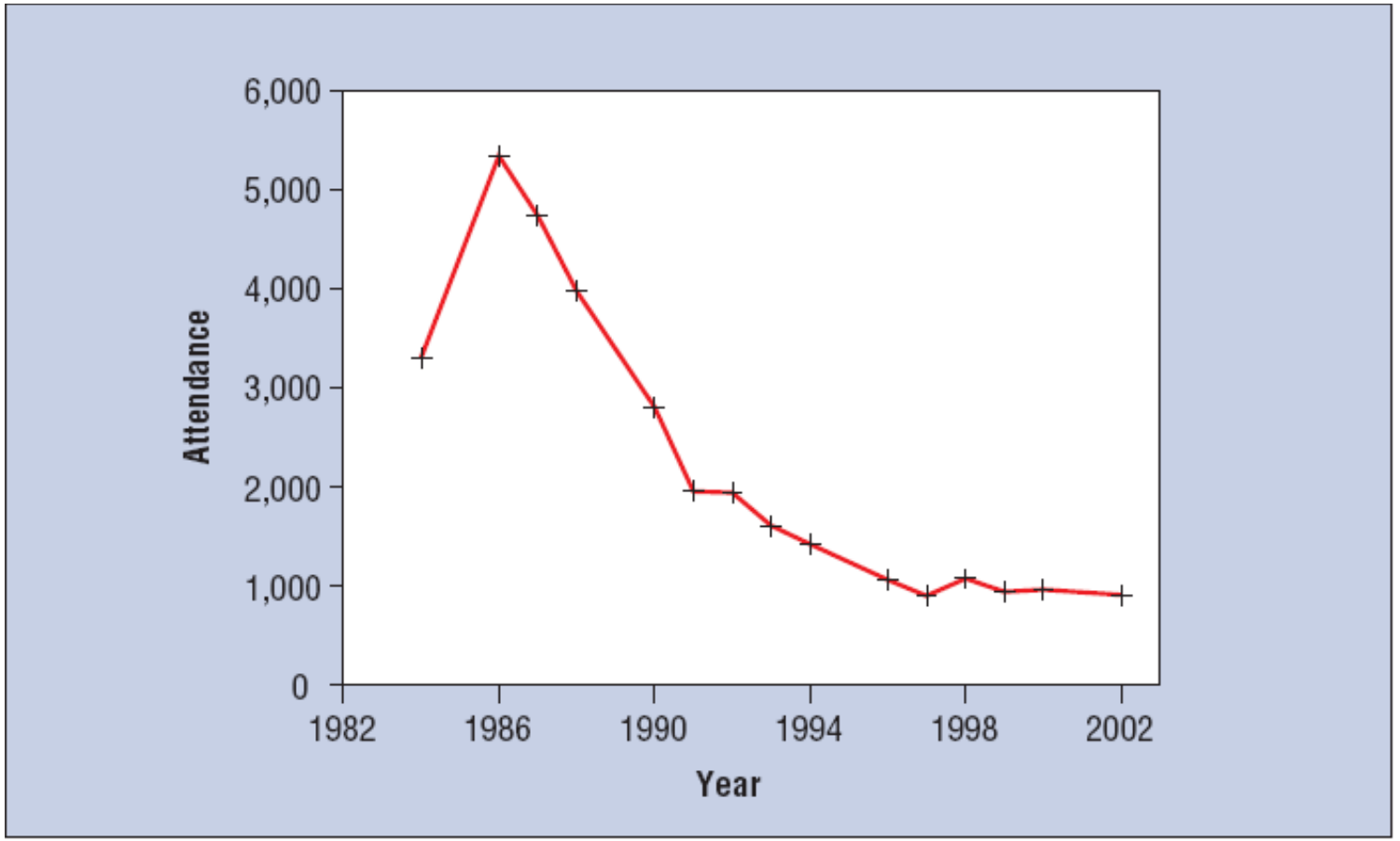
- Tremendous optimism had raised expectations impossibly high, and promised results failed to materialize
  - Funding for AI disappeared
- Neural networks were shutdown for 10 years
- A few reasons
  - Limited computer power
  - Search space explosion
  - Commonsense knowledge and reasoning
  - Logic reasoning is not powerful enough

1980-1987 AI Boom

- Expert system: answer questions about a specific domain of knowledge, using logical rules from experts.
  - Health care
- 1982 John Hopfield proposes Hopfield net
- 1982 Geoffrey Hinton and David Rumelhart proposes backpropagation
- Neural network revives

However, 1987-1993 Second AI Winter

- Like the first AI winter
  - Expert systems are useful only in a few special scenarios
  - Most AI projects are not that useful
  - >300 AI companies had shutdown, gone bankrupt, or been acquired



Attendance at the National Conference of Artificial Intelligence

1993-2011 steady development

- 1997, Deep blue from IBM beats world chess champion.
- 2005, Stanford robot won DARPA grand challenge by driving autonomously for 131 miles
- However, AI researchers call AI other names
  - Informatics
  - knowledge-based systems
  - cognitive systems
  - computational intelligence
- New York Times 2005: “Computer scientists and software engineers avoided the term artificial intelligence for fear of being viewed as wild-eyed dreamers.”
- Neural network were dead again

Before 2012, other than neural networks..

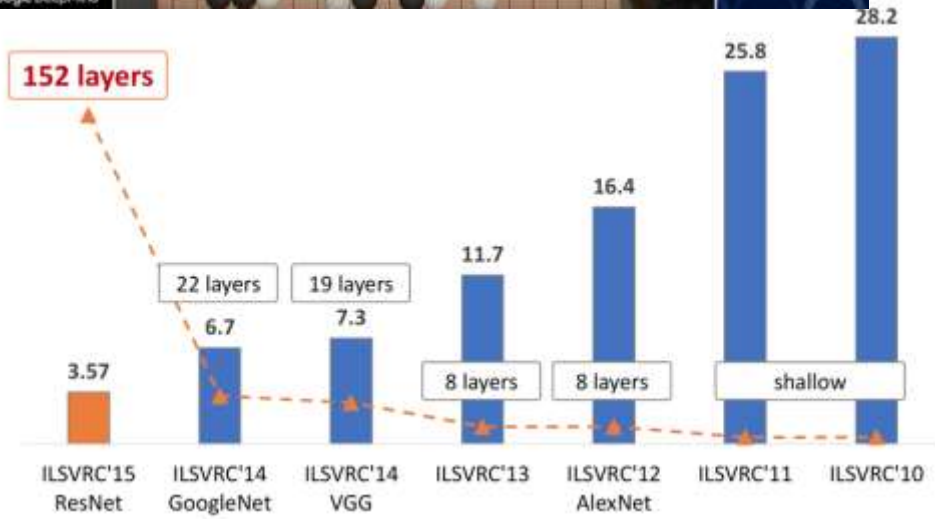
- Support vector machines (SVM)
  - We will cover it in class
  - A fancy version of linear regression
- Graphical models
  - We will not cover in this class
  - Very useful in practice
- Reinforcement learning
  - We will not cover in this class
  - Can do pretty well on Go for small size board (say, 9*9)
- ……

My undergrad.. 2008-2012

- If you search neural networks on Google
  - All the webpages are negative
  - “Neural network is dead”
- I was very interested in AI in my first year
  - But then I gave up
- I did not even learn neural network in my undergrad
  - I was convinced that it’s dead
- No one talks about neural network at that time
  - Just like it never existed

2012-now Deep learning

- 2012, Hinton's group use deep learning for imagenet
  - 16% error rate (compared with 26% 2nd best entry)
  - Beginning of deep learning era
- 2013, Deepmind beats human on Atari
- 2016, Deepmind (AlphaGo) beats human on Go
- ……

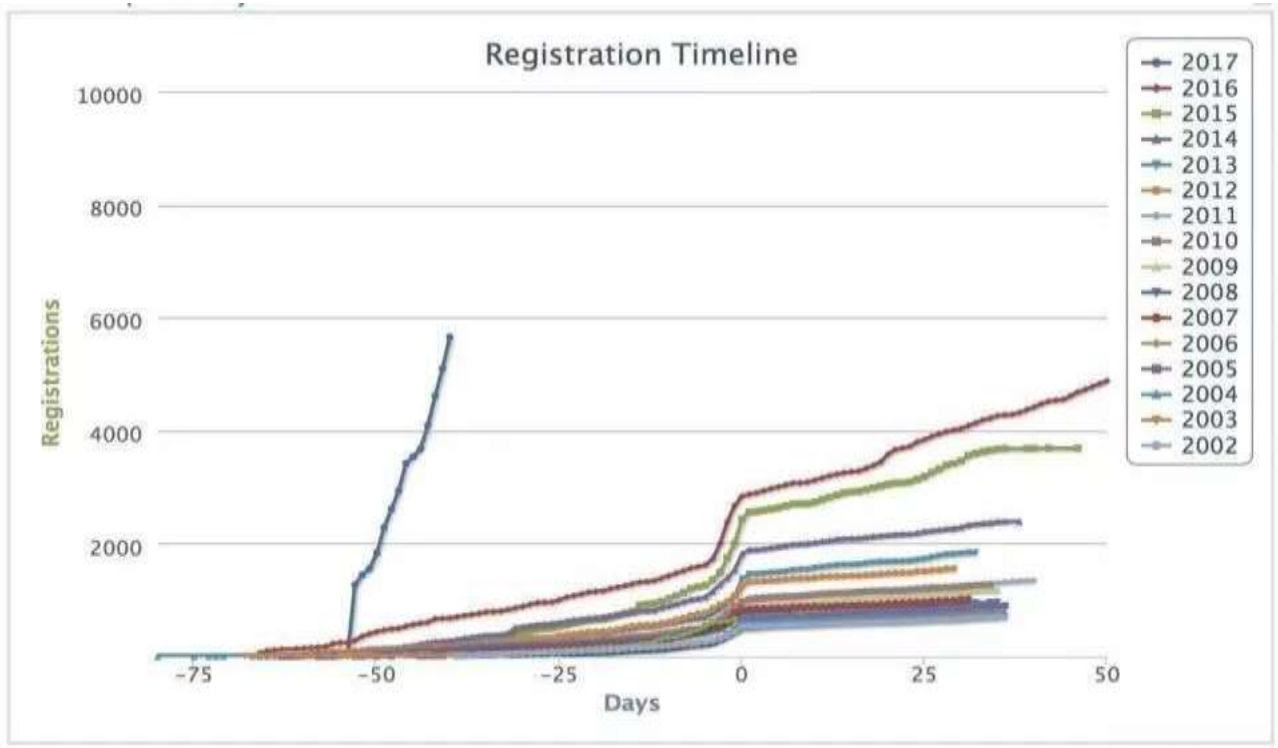


When will be the next AI winter?

- Maybe not too far away

殷鉴不远，在夏后之世

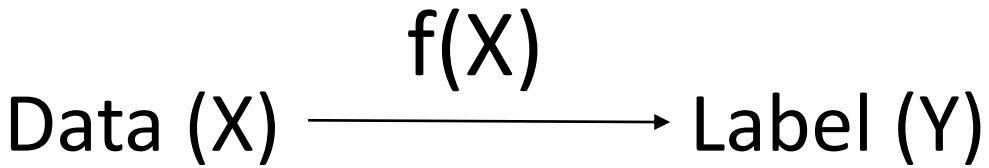
《诗经·大雅·荡》



Be prepared!

Supervised learning framework

- Supervised learning is an important subarea of machine learning
  - Input $X = (x_1, x_2, \cdots, x_N)$
  - Output $Y = (y_1, y_2, \ldots, y_N)$
  - We want to learn the a function f, such that $f(x_i) \approx y_i$
    - Sometimes exact equality is hard to get
    - Sometimes Y contains noise
  - Essentially: **learn by examples**



Example: MNIST

- Every input x is an image of 28*28 pixels
- Every output y is a label in {0, 1, …, 9}
- We want to find a f such that
  - Given an image, it outputs the correct label

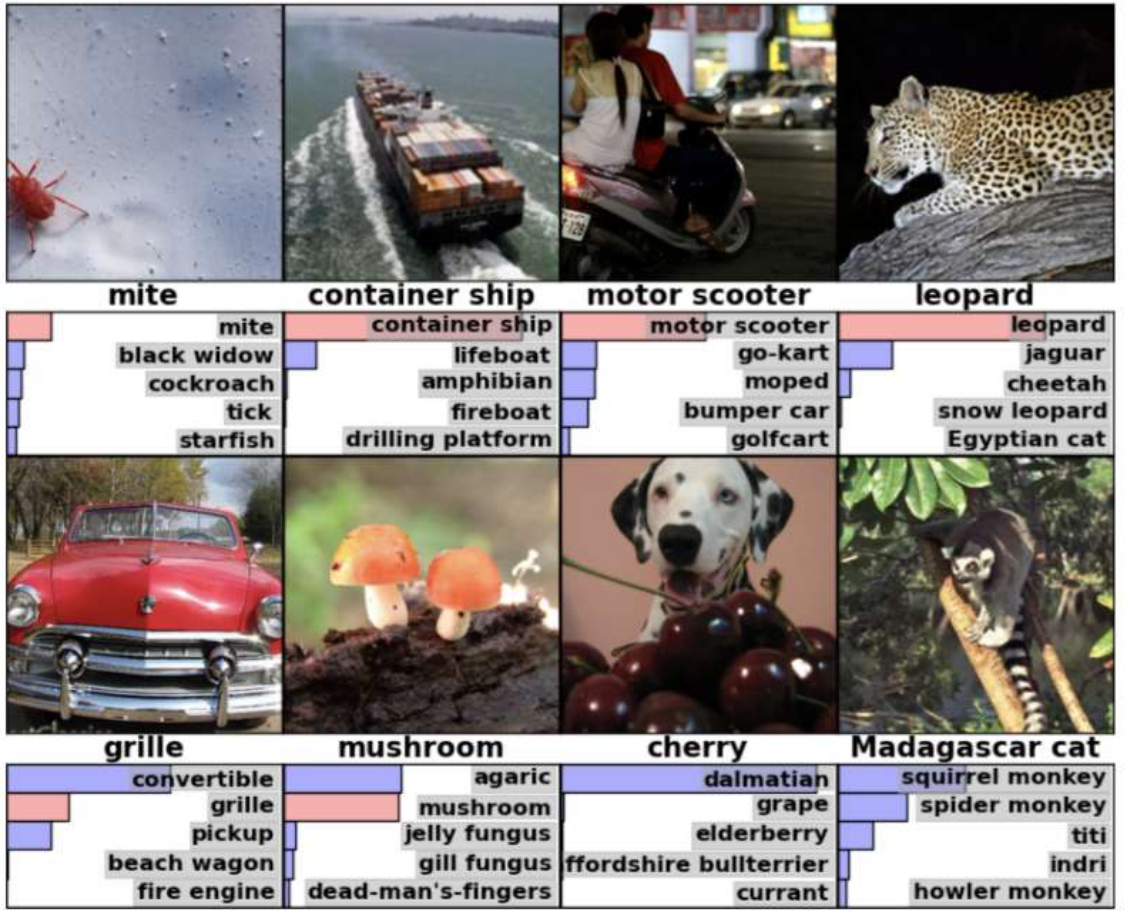
- f()=2
- f()=5

Example: Imagenet

- Every input x is an image of 224*224 pixels
- Every output y is a label in {0, 1, …, 999}
- We want to find a f such that
  - Given an image, it outputs the correct label

- f()=“cat”

- f()=“dog”



Example: Movie review

- Every input x is a movie review (text paragraph)
- Every output y is a label, “good” or “bad”
- We want to find a f such that
  - Given a review, output whether it is “good” or “bad”

- f(“I really love this movie”)=“good”
- f(“This movie is waste of your time”)=“bad”

How do you evaluate f?

- Millions of functions, which one is good, which one is bad?
- We define "loss function l"
  - Distance of prediction f(X) to Y
- For categorical target ("y=cat, dog, good, bad", **classification**)
  - $l(f, x_i, y_i) = 1$ if $f(x_i) \neq y_i$
  - $l(f, x_i, y_i) = 0$ if $f(x_i) = y_i$
  - Good, but not differentiable. We will get back to it later
- For real number target ("y=0.1, 0.55, 1.5", **regression**)
  - $l(f, x_i, y_i) = dist(f(x_i), y_i)$
  - E.g., $l(f, x_i, y_i) = (f(x_i) - y_i)^2$
- $L(f, X, Y) = \frac{1}{N}\sum_i l(f, x_i, y_i)$
  - Sum loss together.

Supervised learning goal: find a good f

- With loss L, we want to
  - Find f s.t. $f = \min_f L(f, X, Y)$
- How do we find this f?
  - This is called **optimization**: we use algorithms to minimize the loss L
  - Will get back to it later
- But minimizing L is not enough!
  - f is a complicated if-function (memorization):
    - If input = xi, output yi
    - Otherwise, output a random number
  - We get $L(f, X, Y) = 0$
  - But this f is useless!

A good f should also generalize well

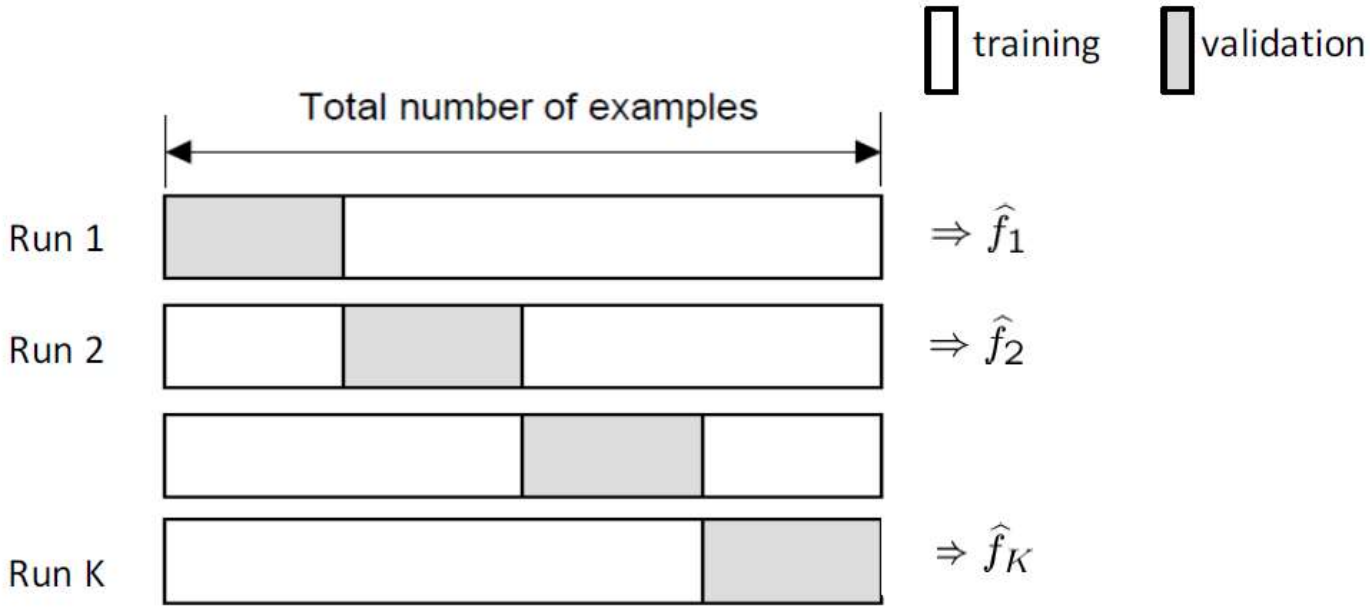
- f should "fit" the **training data** (X, Y) well → minimize L
- But also "generalize" well → doing well for unseen data
  - i.e., $L(f, X', Y')$ is small for X', Y' not seen before
  - X', Y' are called **test data**, from the **same data distribution**
  - That is, we want to have **generalization** guarantees for f
- In other words
  - We should rule out memorization functions
  - f should learn something from (X, Y), not just 死记硬背

How to ensure good generalization?

- A fundamental question in machine learning
  - We will get back to this question many times in this class
- A practical trick: validation
  - Split the training data into two parts: training+validation
    - E.g., 90% is training, 10% is validation
    - Find f using training set, and test it on validation set
  - f never sees validation set during training
- This way, we could estimate how well f generalizes

How to ensure good generalization?

- Cross validation
  - Split data into k parts (e.g., k=10)
  - Do the following for k times:
    - Use the i-th part as validation, the remaining as training
    - Train f on the training set, test it on validation set
  - Average all results together, get a more accurate estimation for f



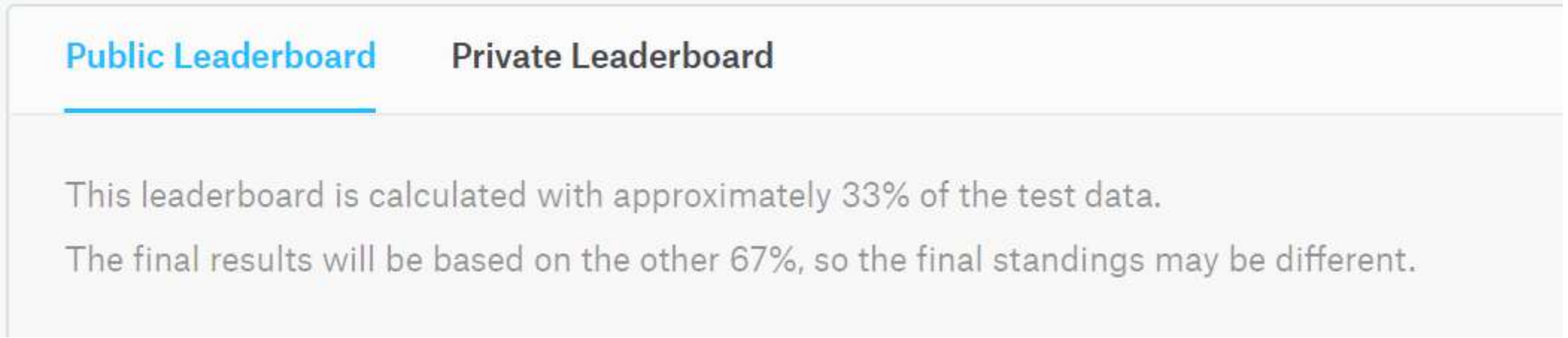
Supervised learning: step by step

1. Identify the task you want to solve (e.g., MNIST)
2. Create a dataset, containing thousands, or millions of examples
   - Input $X = (x_1, x_2, \cdots, x_N)$
   - Output $Y = (y_1, y_2, \ldots, y_N)$
3. Define a loss function L to evaluate $f(X)$
4. Learn a function f to minimize L (optimization)
5. **Minimizing L is not enough!** (generalization)
   - We need to ensure f that generalizes well
   - We could use a validation set in practice
   - But in theory, we will see some guarantees

Never touch your test set during training!!! (that's stupid)

Take Kaggle competition as an example

- It shows your performance on a validation set, on the leader board



- So do not try to do “fine tuning” to make you leader board higher
  - Maybe you are overfitting
- You can split your data to do cross validation, to get a more sensible estimation of your performance

投票 最多可选1项

设置

Is it clear? Do we need to go back?

A It is clear

B Let us go back

提交