20210908

Wednesday, September 8, 2021 3:27 PM

- TA lead discussion section
- Graders
- 1st goal: data science and machine learning
  - Breadth, not depth
- 2nd goal
  - If your code takes 3 days to run, you need to produce good code
  - Productive workflow
    - Allowing for collaboration
  - How do you write code in a sustainable way
  - Agile and Scrum workflows in final projects
    - To learn to collaborate with team of engineers
- 3rd goal:
  - Think critically
  - Learning from data
  - No unbiased data, no perfect model
  - Garbage in Garbage out
  - Need to know how to hold yourself accountable
  - Future data scientist perspective
    - AND consumer perspective
      - Who is this built for?
      - Who is this built by?
      - Are there any conflicts on interest?
      - Will bring guest lecturer (try) expect on topic
- You'll get a lot of technical skills and soft skills
  - Lots of opportunities to work on skills
  - Network
  - You'll get a lot out of this class if you put in the work
  - Focus on what you'll get out of the course skillwise, instead of the grade
    - (Some about skills and grading I didn't understand)
- Office hours 6-7pm remotely Wed
- Manyuan will be Monday 3-4pm
- Course load
  - 4 or 5 hws
  - 1 midterm
  - Final project
  - No final exam
  - All the HWs are already out
    - If you are late, you'll get 0

        - Plan ahead, get organized
    - Midterm
        - Kaggle competition
        - Data science
        - Between A1 and B1
        - Will be given a dataset, your goal will be prediction
        - He'll have the true values
        - Grading:
            - 20% will be based on ranking
            - Live leaderboard
            - 100s of submissions of predictions OK
            - Midterm will last about 10 days
            - Can go nuts
            - Can try different tools
            - 80% will be based on your report, thought process, and what you tried
                - Will account of some discrepancy in programming ability
                - Describe process in general, and what gets you the best results
    - HW
        - Submit own work
        - Can discuss and help each other
        - Some of us are CS, some are math people
        - Both sections have the same HWs
    - Final project is team based
- Final project: most important, and one of the biggest components
    - Can do BU sparks
        - Provides a project manager and will connect with the client
        - Team will have Team lead whose job will be to break down the tasks into individual achievable block
        - Higher expectations with spark project, but you'll get more out of it
            - Soft skills, networking, internships
        - HAVE to deliver
        - Scope is much more realistic
    - Have to submit your own project proposal by next Friday
        - Check out the google doc
        - Needs to be approved before you can start working on it
        - Can submit a vague idea, and they will work with us to

    - scope it down
  - Teams 3-5 students
  - Submit scrum files to project repo
    - So they can see if we are having challenges
    - And so we'll get organized
    - Want to get us to the place where we can show this on our github repo
  - Can also email for help from prof, TA, or project manager
- Grading
  - There is no curve for this class
  - If we all due well, we can all get an A
  - If there is an issue…
  - 5% extra credit
    - There are a few ways
    - If we submit notes after this class, that is one path forward to getting extra credit
- Read the expectations
- Help each other, there is no curve no reason not to
- There are separate projects per section, can't cross over
- There are 6 spark projects per section
  - There will be teams working on the same project, that is intended and OK
  - Everyone is guaranteed a project with Spark
  - Team matching algorithm will be based on a form and your preferences
    - Will try to maximize the overall preferences for the class
  - Will try to incorporate team mate preferences, but that may be too complicated
    - If you are interested in coding this up for future classes…
  - Project managers are posted
- Participation is part of extra credit: how much you participate in discussions, contributing to repo, etc…
- Cannot just put all the work in at the last week, you need to be consistently working throughout the semester and working with your team, submitting deliverables, etc.
- Give him at least 24 hours to respond to email
- He works full time, so meetings will be virtual
  - Can propose a few times when you email him, or send him a poll, and then he'll send you an invite
- Re-grades:
  - You have 48 hours to appeal a grade
  - Email Peter Tang who is taking care of all the grading

- Variety of platforms, no blackboard
- Get ahead: do git workshop
- Coarse repo
- Create github profile if you don't already have it
- First 3 labs are posted
    - Will be done during lab, but you can do it ahead
- Labs
    - Meant to get us all on the same page
    - Then we'll implement some functions
    - There is a template available if you want to make your own python package
- Discussion sections
    - Office hours for projects & to get technical advice
    - Personal projects, will get some project management advice
    - If we don't finish lab during discussion section
        - You should be able to, but get it done if you can't finish during the discussion section
- Schedule
    - Early insight project presentation
        - Meant to make sure you've done some work, asked and answered some classes
    - Final project presentations will likely be remote due to logistics
    - Pitch day on the 20th will be pitched by the clients themselves
        - That will likely be remote as well (each with 5min pitch)
    - Room for extra topics, remind him to post a poll so we can vote
- Project deliverables
- The labs will be the same for both lectures, but he'll check if we can swap
- Lectures WILL NOT BE RECORDED
    - Don't come if you are sick
    - Can email if sick for resources
    - Also wants all of us to upload our notes for this reason

==========

- Won't be office hours today, he'll have to rush home after this class in order to make that time

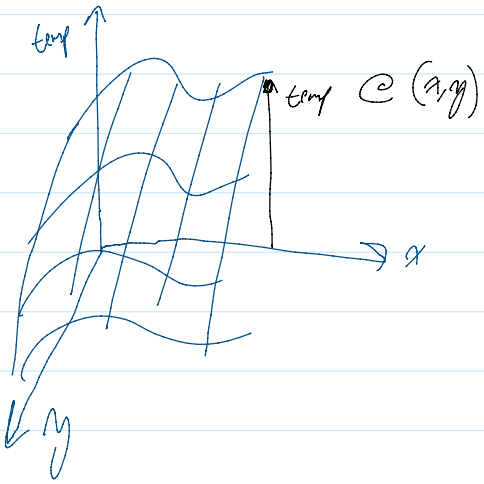
==========

==========

- Knowledge has to take the form of testable predictions

- Metal plate will eventually reach equilibrium
- A physicist may split the plate into coordinates and record temperatures over intervals of time
  - (x, y)

At time t_0



$f(x, y, t_i) \Rightarrow$ temperature

- t_i → time
- x, y → coordinate

What is the knowledge gained from this model?
Falsifiable, testable prediction

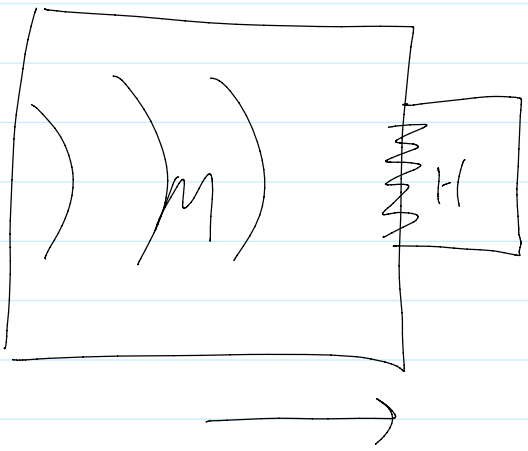
Alternative way to look @ this:

In hindsight, model predicted how heat propagated through plate.

↳ you could say "magic", and there is no way

↳ you could say "magic", and there is no way to test & verify

If 2 models can predict the same thing, they are the same theory ?



What if heat flowed from metal plate to heat source?
↳ our model didn't predict this

Theory of magical wizard
↳ if equally good @ explaining every outcome, then you have <u>zero</u> knowledge

Theory must be falsifiable to gain some knowledge
↳ need to <u>try</u> to falsify the theory

Rule that governs triples of integers.

(a, b, c)

(2, 4, 6) follows the rule

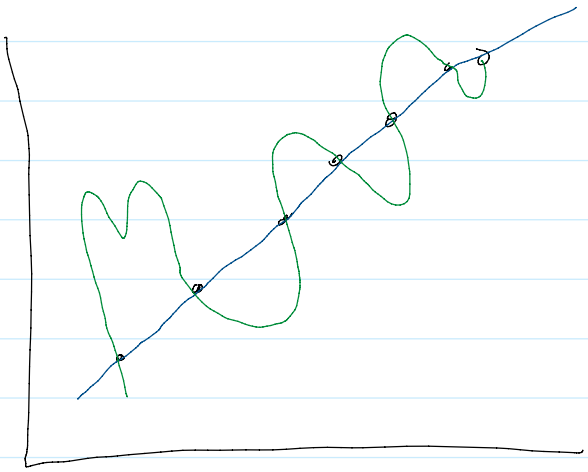
Participant A: does (2, 4, 3) work? No
(10, 12, 14) Yes
(5, 7, 9) Yes

Part A then guesses rule ⟶ hyp $h \to (x, x+2, x+4)$

Poll: which do you want to try?
A (100, 102, 104)
B (11, 13, 15)
C (1, 2, 3) ← most voted for

There are some inherent ... to this game

- Not all guesses have same amount of info
- Set of examples may no be representative of the rule
- There are ∞ # of rule that can fit a finite amount of examples



Try Occam's razor: generally prefer simpler explanation

Try Occam's razor: generally prefer simpler explanation

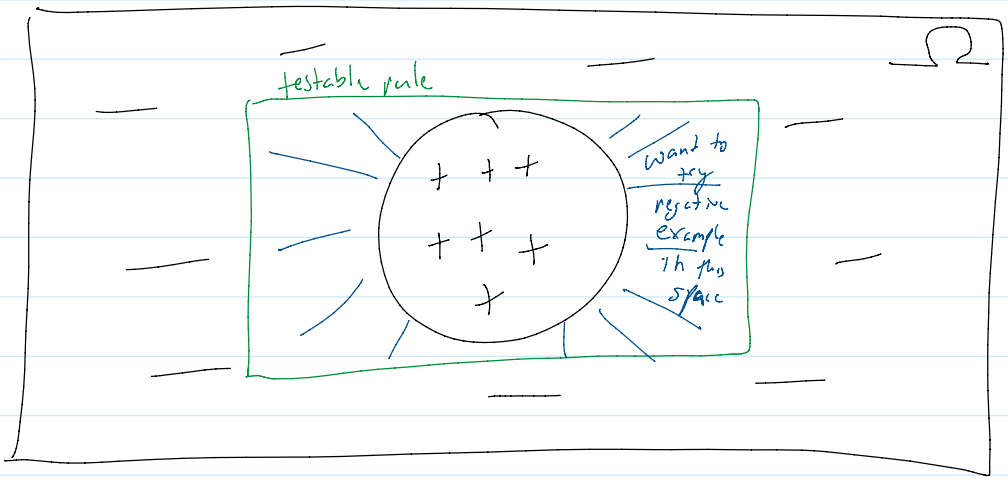
e.g. the straight line

Given particular hypothesis:

- positive examples → expect positive result
- negative examples → expect negative result

The Rule was: $a < b < c$

→ you must try negative examples to guess this rule



set of all possible examples

In neuropsych class, only 20% of students guessed rule correctly.

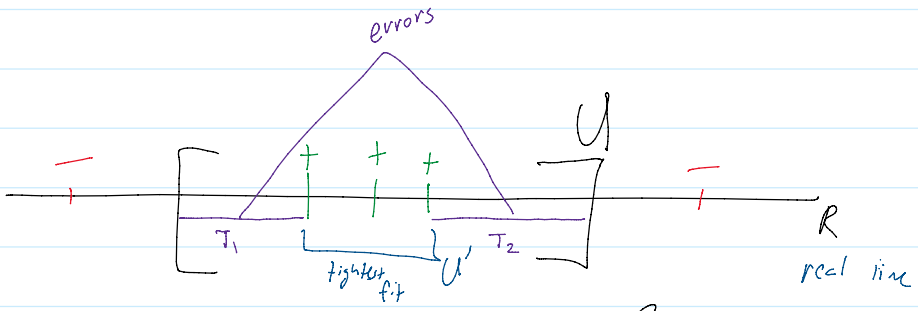
Can't test everything?

How do you come up w/ hypotheses & the algorithm for testing?

Positive bias: "the inventors curse"

Computational learning theory

If you come up w/ a hypothesis for rule
& how many time to test random examples?



What is the interval U?

tightest fit method

U' has an error rate of at most ε w/ probability ρ missed something

↳ want ↑ probability of ↓ ε

↳ What does this depend on?

Here we have algorithm

What is the error rate of this method?

$P(T_1 \cup T_2) \leq \varepsilon$ ← is this right?

Is there a mathematical framework for learning

Computational Learning Theory is beyond scope of the course

Will do review of probability if needed

==========

(Try to get slides)

Git and Github

- Are you familiar with rebasing?

Github vs Git

Motivation

Minimal progress loss

- Regular save points (commits)

Demo: next time

Iterating on Different Versions

- Want to iterate on different versions
  - May be easier to add feature on a simpler version
- Now you have a conflict on the history
- We need a way to preserve both versions of history if we want to
- Want to overwrite history if we choose
- So branch off a commit and make changes that way
  - Now can upload branch 2 and maintain that history separately
  - And its nice a clean
- Edge of branch is head of branch, the start point of branch is the base
- Master Main tend to be the main version/product
- Name branches based on the feature you are working on OR the code base…

    - This naming convention is important
- Can delete branches without effecting changes on other branches
- At some point you may want to clean up branches and merge them together
- Sometimes the base of one will be head of another and that is easy to merge
- But otherwise, need to merge the branches
    - This is rebasing and is complicated
    - Normally a manual version of process
    - Will need to resolve a lot of conflicts manually