

Hi!

CSCI 3202-002 ~~Fall~~ 2020 Intro to Artificial Intelligence

Hi!) Start @ 1:50

Instructor: Dr. Zachary Mullen

HELLO AND WELCOME!

Syllabus Material

The course Canvas page will house:

- ① 1. The course syllabus and schedule
- ② 2. Annotated lectures and their videos posted after completion of the lecture
3. Homework Assignments and their turn-ins locations
4. Grades
5. Links to In-Class Notebooks, Data sets, and course Piazza page (register!)
6. Whatever else is necessary

② Zoom

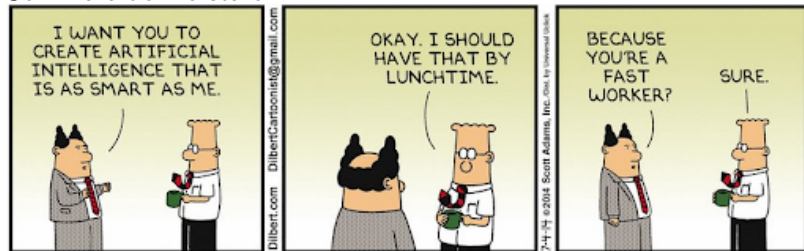
③ Piazza

Piazza: <https://piazza.com/colorado/fall2020/csci3022> ³²⁰²

1. Ask questions in Q & A forum (and answer other students' questions!)
2. Discuss work, but do not post solutions/vital code
3. Send private messages to faculty instead of email (keeps things organized)

AI!

So where do we start?



- ▶ What is Artificial Intelligence?
- ▶ To that point, what is even intelligence?

- ▶ From the text: "It's one thing to say that the mind operates, at least in part, according to logical rules, and to build physical systems that emulate some of these rules; its another to say that the mind itself is such a physical system. ...if the mind is governed entirely by physical laws, then it has no more free will than a rock "deciding" to fall toward the center of the earth."

Applications of AI

Realms	Questions
Philosophy	<p>What is knowledge and what isn't?</p> <p>How do knowledge and the mind arise from the brain?</p> <p>How does knowledge lead to action?</p>
Mathematics	<p>Can formal rules define intelligence? ($p \rightarrow q$)</p> <p>What can be computed? What can be done with incomplete information?</p>
Economics	<p>How do we make decisions to maximize payoffs?</p> <p>How does this change if others are competing with us?</p>
Life Sciences	<p>How do brains process information?</p> <p>How do humans and animals think and act?</p>

AI!

Where are we now?

1. How can we build an efficient computer?
2. How can artifacts operate under their own control?
3. How do languages relate to thoughts?



IN CS, IT CAN BE HARD TO EXPLAIN
THE DIFFERENCE BETWEEN THE EASY
AND THE VIRTUALLY IMPOSSIBLE.

Thinking or Acting?

The crux of the issue lies in the following goals:

thinking humanly	thinking rationally
acting humanly	acting rationally

In AI, we tend to live in the bottom-right quadrant. The goal is to mimic and reproduce human *behavior*, but not necessarily human thought.

Thinking or Acting?

The crux of the issue lies in the following goals:

thinking humanly	thinking rationally
acting humanly	acting rationally

In AI, we tend to live in the bottom-right quadrant. The goal is to mimic and reproduce human *behavior*, but not necessarily human thought.



Famous AI problems:

So our plan is to think “right,” even if not exactly human. We want to make the best decision possible given all available information.

Think of things “AIs” have been able to achieve already:

- ▶ The Turing Test
- ▶ The ELIZA chatbot
- ▶ Ability to beat humans in games (Deep Blue vs. Kasparov, AlphaGo vs Sedal)

Solving an AI problem:

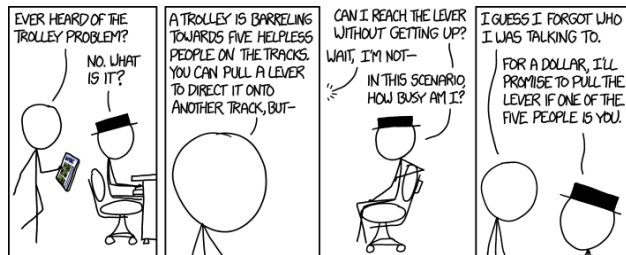
Breaking down an AI problem, then, is about the establishing rational goals.

1. An rational algorithm will try to maximize pre-existing goals. It decides *what* decisions are made, but not truly *why*.
2. Goals must be expressed into a quantifiable **utility**: being rational is being optimal according to this score.
3. **Maximize**: evaluate all options and pick the best
4. **Expected**: evaluate those decisions based on possible uncertainty or missing information (hello prob/stats!)
5. **Utility**: what do you value?

Unsolved Problems

So what do we want to solve?

1. Autonomous Cars



Unsolved Problems

So what do we want to solve?

1. Autonomous Cars
2. Visual Recognition



Which of these vehicles is a threat?



Unsolved Problems

So what do we want to solve?

1. Autonomous Cars
2. Visual Recognition
3. Text Recognition

Suppose are handed a new object:



What is this?



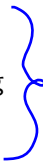


Learning Goals

At the end of this class, students should possess:

1. An understanding of the issues involved in building computer systems that behave rationally
2. The ability to design and implement standard algorithms for heuristic search, logical reasoning, probabilistic inference, machine learning and natural language processing problems
3. The ability to evaluate the performance of these algorithms on realistic datasets.

Tools

Algorithms and method we will explore include:

1. Search algorithms
 2. Search heuristics
 3. Game trees
 4. Markov decision processes
 5. Bayes nets and probabilistic reasoning
 6. Hidden Markov models
 7. Classification techniques, including basics of supervised learning and unsupervised learning
 8. Decision diagrams and information entropy
- 
- 
- 

Coding Overview

1. We will use Python 3 and in particular Numpy and Pandas
2. Lot's of great data science libraries and decent plotting
3. We'll exclusively work in Jupyter Notebooks. We strongly recommend you install local copy
4. If not, you can use Microsoft Azure or Google Colab notebooks
5. Often work on problems in groups in class
6. Bring a laptop or have a buddy with a laptop



Python

What:

Python is free high-level programming language built for flexibility and simple syntax. It is commonly used in statistical computing and graphics.

Why we're using it:

It's widely used - especially in industry - free, and has a healthy repository of packages.

Common Syntax

Function Syntax: Functions use indents to determine the stopping point after a colon. The function `def myfunction(x):` ends after the indenting stops. *Indexing:* Python is 0-indexed, and uses square brackets. For an $n \times m$ matrix named `mydata`, `mydata[3,2]` accesses the entry in the fourth row and third column. *Comments:* `#` comment

Favorite Reference: Official Documentation

Jupyter

What:

Jupyter notebook is free web application to combine running live code and visualizations.

Why we're using it:

Statistics is inherently interdisciplinary, and communication of clear results is paramount. The notebook environment encourages replicable results and a clear workflow.

Common Syntax

Cells: The notebook is divided into cells. For our purpose, expository material will be done in Markdown cells with \LaTeX compatibility. Computational work will be done in Python 3 code cells, which may also generate plots, histograms, tables, and other output.

Formatting: `#` (with varying numbers of `#` signs) can be used to create section headers in markdown cells.

Comments: `%` comment



What:

LaTeX is a typesetting software with a particular emphasis on mathematics, including matrices, greek letters, etc.

Why we're using it:

Microsoft Equation Editor is a pain. It's included in Jupyter for Markdown cells.

Common Syntax

Function Syntax: Functions use curly brackets; "`\textit{arg}`" would italicize the argument

Math mode: inputting "`$ arg $`" will apply mathematical typesetting to the argument.

Comments: `% comment`

Favorite Reference: <https://en.wikibooks.org/wiki/LaTeX>

Evaluations

Workload: *a final reason*

- ▶ (48%) Homework assignments (every 1-2 weeks, lowest dropped, late days)
- ▶ (13%) Midterm exam
- ▶ (13%) Final exam (cumulative) *} take home*
- ▶ (10%) Practicum 1 (midterm)
- ▶ (10%) Practicum 2 (final)
- ▶ (6%) Participation (Canvas) *due after each lecture:*

Read the syllabus! More details can be found there regarding course policies (see: Late days!)

*by 11:59 pm night before next lecture
(Tu / Th / Su)*

Remote Materials

All lectures for this class will be hosted remotely via Zoom from 1:50-2:40pm on MWF. The zoom link is <https://cuboulder.zoom.us/j/92837524184>, and will be open around 1:40pm most days. All lectures will be recorded and posted to the course schedule.

There will be a warmup/intro problem to complete if you arrive between ~~4:40~~^{1:40-1:50}pm.

I will try to make Zoom as interactive as possible: use Zoom reactions, raise your hand if you have questions, and answer polls as I put them out.

It is my *strong* preference to have cameras on if your bandwidth can support it. It helps people feel invested and engaged in the process!

Fridays will typically be coding/application based - you are highly encouraged to follow along and attempt the exercises *before* class. M/W will be heavier on theory: pen-and-paper exercises and annotations on slides.

Try to stay engaged! Take minute papers seriously, and ask questions through any/all mediums available (Zoom, Piazza, minute papers)

The curse of Laptops



“Results showed that students who used laptops in class spent considerable time multitasking and that the laptop use posed a significant distraction to both users and fellow students. Most importantly, the level of laptop use was negatively related to several measures of student learning, including self-reported understanding of course material and overall course performance.”

<http://www.sciencedirect.com/science/article/pii/S0360131506001436>

Also: <http://journals.sagepub.com/doi/pdf/10.1177/0956797616677314>

And: <http://www.sciencedirect.com/science/article/pii/S0272775716303454>

If at first you don't succeed...

TA/CA
: 2 GSS

1. When you're asking for help, be sure to explain...
2. what you're trying to do
3. what you think should happen
4. what you get instead (copy/pastes or screenshots work well)
5. what all you have tried
6. if you haven't tried anything, try something first

Learning New Software

There are 3 major tools to use in learning new software:

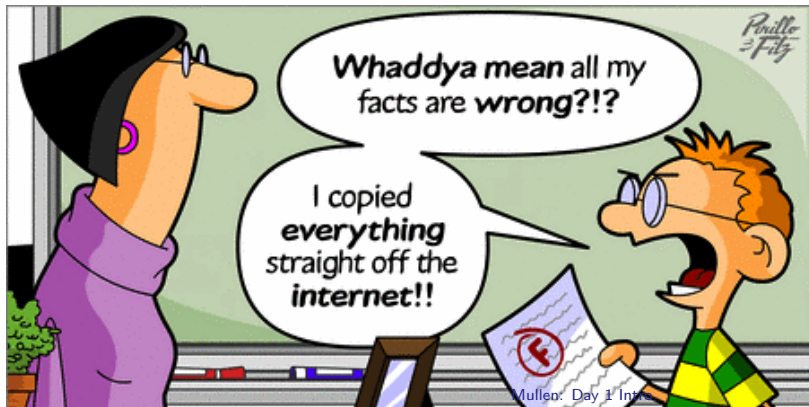
1. Pirating similar code found from course materials, etc.
2. Official documentation
3. Google searches, often directed to sites like stackexchange. (Don't Copy/Paste! Write from pseudo code, and *cite any sources* if you use them!)

Use (1.) and (2.) often, but be very careful with #3..., and don't hesitate to

1. Ask your instructor or peers for ideas on how to write specific routines, or for their syntax knowledge. Piazza is made for exactly this sort of thing!

Academic Integrity

1. See the CU Academic Integrity Policy for more details. Here are some highlights.
“Examples of cheating include: copying the work of another student during an examination or other academic exercise (includes computer programming)”
2. “Examples of plagiarism include: . . . copying information from computer-based sources”

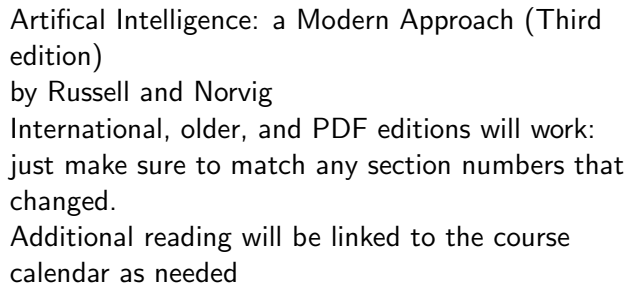


Integrity Examples

Example 1: For an assignment, Chris searches the internet for relevant codes and copy-pastes them into his Jupyter Notebook. He properly cites the source of the codes.

Example 2: For an assignment, Maciej and Felix work together to figure out how to implement the codes, but each works on their own computer and develops their own software.

Example 3: For an assignment, Rhonda has a plan for how to implement an algorithm, but isn't sure how to manipulate a Python list in a particular way that she needs to. She searches the internet, finds a fix, and implements it in her code without copying it.



Moving Forward

Let's get to work!

► Before next class:

1. Make sure you can access the Canvas page and read the syllabus
2. Set up some way to back up your work
3. Install Anaconda (or other reliable Jupyter notebook method)
4. If new to Jupyter/Python3: Review and complete Numpy/Pandas tutorials if needed