Artificial Intelligence

CS 165A

Jun 8, 2023

Instructor: Prof. Yu-Xiang Wang

→ Final Review
Remaining time today

- Review for the final

- Types of questions that you may encounter
Tips for studying for the final

• Focus on the lectures and concepts
  – Solve all quiz questions
  – So you don’t get tricked in T/F, MCQ questions.

• No need to make a cheatsheet, but knowing which lecture to find which topic could be useful.

• For any concepts that you are confused, check the textbook (Again, books are random access, you don’t have to read chapters from the beginning to the end)
  – Stick to AIMA book for Search and Logic
  – Stick to the Sutton and Barto book for RL.
## We’ve come a long way…

<table>
<thead>
<tr>
<th>Week</th>
<th>Topic</th>
</tr>
</thead>
</table>
| 1    | Course Overview & Intelligent Agents  
     | Machine Learning |
| 2    | Machine Learning  
     | Machine Learning |
| 3    | Probabilistic Graphical Models  
     | Probabilistic Graphical Models |
| 4    | Search: Problem solving with search  
     | Search: Search algorithms |
| 5    | Search: Minimax search and game playing  
     | Midterm Review |
| 6    | Midterm  
     | RL: Intro / Markov Decision Processes |
| 7    | RL: Solving MDPs  
     | RL: Bandits and Exploration |
| 8    | RL: Bandits and Exploration  
     | RL: Reinforcement Learning Algorithms |
| 9    | RL: Reinforcement Learning Algorithms  
     | Logic: Propositional Logic |
| 10   | Logic: First Order Logic  
     | Responsible AI / Final Review |
| 11   | Final Exam |

### Topics:
- **Machine Learning**
- **Probabilistic Reasoning**
- **Search**
- **Reinforcement Learning**
- **Logic**
- **Responsible AI**
Lecture 1: AI Overview

- AI for problem solving
- Rational agents
- Examples of AI in the real world
- Modelling-Inference-Learning Paradigm
Modeling-Inference-Learning

Modeling

Inference

Learning
Structure of the course

Probabilistic Graphical Models / Deep Neural Networks

Classification / Regression
Bandits

Search
game playing

Markov Decision Processes
Reinforcement Learning

Logic, knowledge base
Probabilistic inference

Reflex Agents

Planning Agents

Reasoning agents

Low-level intelligence

High-level intelligence

Machine Learning

Potential question in the final: what type of agents are suitable to a given problem?
Our view of AI

• So this course is about designing rational agents
  – Constructing $f$
  – For a given class of environments and tasks, we seek the agent (or class of agents) with the “best” performance
  – Note: Computational limitations make complete rationality unachievable in most cases

• In practice, we will focus on problem-solving techniques (ways of constructing $f$), not agents per se
Different Ways of Looking at the AI

- Agent types / level of intelligence
  - Low-level: Reflex agents
  - Mid-level: Goal-based / Utility-based agents: planning agents
  - High-level: Knowledge-based: Logic agents

- Optimization view
  - Everything is an optimization problem

- Theoretical aspects
  - Time/space complexity
  - Algorithms and data structures
  - Statistical properties: # of free parameters / how easily can we learn them with data
Optimization perspective of AI

• A rational agent $\max_{a_1, \ldots, a_T} \text{Utility}(a_1, \ldots, a_T)$
  
  - Modelling tools: Features / Hypothesis, PGM, State-space abstraction, agent categorization
  - Constraints: Computation, Data, Storage

• Discrete optimization $\min_{\rho \in \text{Paths}} \text{Distance}(\rho)$
  
  - Algorithmic tool: Search / Dynamic programming

• Continuous optimization $\min_{\theta \in \mathbb{R}^d} \text{TrainingError}(\theta)$
  
  - Algorithmic tool: Gradient descent / Stochastic gradient descent
Different objectives to optimize in AI (first half of the course)

- **PGM:**
  - MLE: Maximize the log-likelihood function
  - Classifier / decision: max the posterior distribution

- **Search and planning:**
  - Find valid solutions with smallest path cost.
  - Minimax search / games: Maximize your worst-case pay-off (assuming your opponent plays optimally)

- **Machine Learning**
  \[
  \min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta, (x_i, y_i)) + \lambda r(\theta)
  \]
  - (Regularized) Empirical Risk Minimization (ERM):
  - But the goal is to minimize the (unseen) expected loss.
Different objectives to optimize in AI (second half of the course)

- **Markov Decision Processes / RL**
  - Maximize the cumulative expected reward of “decision policy”
  - Balance Exploration and Exploitation.

- **Logic / Knowledge based agent:**
  - Solve a feasibility problem, find a “proof”.
  - Determining “Valid, Satisfiable, Unsatisfiable”
A lot of these problems are computationally / statistically hard, but so what?

• Get help from human:
  – Use a model
  – Use abstractions at the right level
  – Use features
  – Use heuristic functions

• Get help from mathematics and computer science theory:
  – Solve an approx. solution
  – Approximate inference of a PGM
  – More iterations with less accuracy per SGD
  – A* Search
  – TD learning and Bootstrapping
# Modeling-Learning-Inference Paradigm

<table>
<thead>
<tr>
<th></th>
<th><strong>Modeling</strong></th>
<th><strong>Learning</strong></th>
<th><strong>Inference</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier agent</td>
<td>Feature engineering</td>
<td>Minimize Error rate</td>
<td>Prediction on new data points</td>
</tr>
<tr>
<td>(Spam filter)</td>
<td>Hypothesis class</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Probabilistic</strong></td>
<td>Joint distribution</td>
<td>Fitting the CPTs to Data</td>
<td>Marginalization</td>
</tr>
<tr>
<td><strong>Inference agent</strong></td>
<td>Draw edges in BN</td>
<td></td>
<td>(conceptually easy)</td>
</tr>
<tr>
<td>(Sherlock)</td>
<td>Conditional independences</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Search agents</strong></td>
<td>State-Space-diagram</td>
<td>Environment given</td>
<td>Uninformed Search</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(learn edge weights)</td>
<td>/ A* Search etc…</td>
</tr>
<tr>
<td><strong>Game playing agent</strong></td>
<td>State-space-diagram</td>
<td>Learn opponent model?</td>
<td>Minimax Search</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Learn evaluation functions?</td>
<td>/ Expetimax</td>
</tr>
</tbody>
</table>
# Modeling-Learning-Inference Paradigm

<table>
<thead>
<tr>
<th></th>
<th>Modeling</th>
<th>Learning</th>
<th>Inference</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Planning Agent</strong></td>
<td>MDPs: State-representation / reward design</td>
<td>Not much</td>
<td>Value Iteration / Policy iterations /</td>
</tr>
<tr>
<td><strong>Reinforcement Learning Agent</strong></td>
<td>Same as above. But also function approximation</td>
<td>Model based: Estimate MDP parameter + VI / PI Model free: Monte Carlo. SARSA. Q-Learning (with linear function approx) . Policy Gradient</td>
<td></td>
</tr>
<tr>
<td><strong>Logic Agent</strong></td>
<td>Formal logic: Syntax / Semantics Representing the knowledge using FOL</td>
<td>n.a.</td>
<td>Logic inference: CNF / Resolution Modus Ponens</td>
</tr>
</tbody>
</table>
Lecture 2-4: Machine Learning

- Machine learning
  - Types of machine learning models
  - Focus on Supervised Learning ---- classifier agents.
- What is a feature vector?
  - Feature engineering, feature extraction
- Hypothesis class and free parameters
  - How many are there? How to evaluate a classifier?
  - Error? On training data or on new data?
  - Overfitting, underfitting?
- How to learn (optimize)?
  - Surrogate losses, Gradient Descent, SGD
Illustration of GD vs SGD

**Observation:** With the time gradient descent taking one step, SGD would have already moved many steps.
One natural stochastic gradient to consider in machine learning

• Recall that

\[
\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta, (x_i, y_i))
\]

• Pick a \textbf{single} data point \(i\) uniformly at random

  – Use \( \nabla_{\theta} \ell(\theta, (x_i, y_i)) \)

  – Show that this is an unbiased estimator!
  – Know which part of the expression is random!
  – Know how to apply the definition of expectation.
Lecture 5-6: Probabilities and BayesNet

- Modelling the world with a joint probability distribution
  - Number of parameters?

- CPTs
  - Count number of independent numbers to represent a CPT

- Conditional, Marginal, Probabilistic Inference with Bayes Rule

- Read off conditional independences from the graph
  - d-separation
  - Bayes ball algorithm
  - Markov Blanket
Tradeoffs in our model choices

Fully Independent

\[ P(X_1, X_2, \ldots, X_N) = P(X_1) P(X_2) \ldots P(X_N) \]

\[ O(N) \]

Fully general

\[ P(X_1, X_2, \ldots, X_N) \]

\[ O(e^N) \]

Space / computation efficiency

Expressiveness

Idea:

1. Independent groups of variables?
2. Conditional independences?
What are the CPTs? What are their dimensions?

Question: How to fill values into these CPTs?
Ans: Specify by hands. Learn from data (e.g., MLE).
Big question: Is there a general way that we can answer questions about conditional independences by just inspecting the graphs?

• Turns out the answer is “Yes!”
What are the probabilistic graphical models for topics we learned in the second half?

- Expectimax
- MDP
- Bandits / Contextual Bandits
- Reinforcement Learning
Lecture 7-10: Search

- Problem solving by search
  - Abstraction, problem formulation
  - State-space diagram
  - Count the number of states, number of actions.

- Uniformed Search algorithms
  - Four evaluation criteria

- Informed (heuristic) search
  - Admissible / consistent heuristics
  - Tree-search vs graph search

- Minimax Search and Game playing
  - Know how to do minimax / expectimax by hand!
  - Pruning
Minimax example

Which move to choose?

The \textit{minimax decision} is move $A_1$
Lecture 11-16 Reinforcement Learning

• Markov Decision Processes
  – New concepts: reward, value function, policy, transition dynamics
  – Bellman equations
  – Iterative algorithms for finding the optimal policy

• Bandits / Contextual bandits
  – The notion of regret
  – Explore-exploit

• Reinforcement Learning
  – Model-based learning
  – Model-free learning
  – Bootstrapping with Temporal Difference Learning
Reinforcement learning problem setup

- State, Action, Reward
- Unknown reward function, unknown state-transitions.
- Agents might not even observe the state
Let us tackle different aspects of the RL problem one at a time

- **Markov Decision Processes:**
  - Dynamics are given no need to learn

- **Bandits:** Explore-Exploit in simple settings
  - RL without dynamics

- **Full Reinforcement Learning**
  - Learning MDPs
Tabular MDP

- **Discrete State, Discrete Action, Reward and Observation**
  
  \[ S_t \in S \quad A_t \in A \quad R_t \in \mathbb{R} \quad O_t \in O \]

- **Policy:**
  
  - When the state is observable: \( \pi : S \rightarrow A \)
  
  Or when the state is not observable:

  \[ \pi_t : (O \times A \times \mathbb{R})^{t-1} \rightarrow A \]

- **Learn the best policy that maximizes the expected reward**

  - Finite horizon (episodic) MDP: \( \pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}[\sum_{t=1}^{T} R_t] \) \( T: \) horizon

  - Infinite horizon MDP: \( \pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} R_t] \) \( \gamma: \) discount factor
State-space diagram representation of an MDP: An example with 3 states and 2 actions.

\[
r(s_2, a_1, s_1) = -2
\]

\[
r(s_2, a_1, s_2) = 50
\]

\[
r(s_2, a_2, s_3) = -1
\]

* The reward can be associated with only the state s’ you transition into.
* Or the state that you transition from s and the action a you take.
* Or all three at the same time.
Reward function and Value functions

• Immediate reward function $r(s,a,s')$
  – expected immediate reward
    $$r(s, a, s') = \mathbb{E}[R_1 | S_1 = s, A_1 = a, S_2 = s']$$
    $$r^\pi(s) = \mathbb{E}_{a \sim \pi(a|s)}[R_1 | S_1 = s]$$

• state value function: $V^\pi(s)$
  – expected long-term return when starting in $s$ and following $\pi$
    $$V^\pi(s) = \mathbb{E}_\pi[R_1 + \gamma R_2 + \ldots + \gamma^{t-1} R_t + \ldots | S_1 = s]$$

• state-action value function: $Q^\pi(s,a)$
  – expected long-term return when starting in $s$, performing $a$, and following $\pi$
    $$Q^\pi(s, a) = \mathbb{E}_\pi[R_1 + \gamma R_2 + \ldots + \gamma^{t-1} R_t + \ldots | S_1 = s, A_1 = a]$$
Bellman equations – the fundamental equations of MDP and RL

- An alternative, recursive and more useful way of defining the V-function and Q function
  - $V^\pi(s)$ function Bellman equation
    \[
    V^\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V^\pi(s')]
    \]
  - $Q^\pi(s,a)$ function Bellman equation
    \[
    Q^\pi(s,a) = \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma \sum_{a'} \pi(a'|s')Q^\pi(s',a')]
    \]
  - $V^*$ function Bellman (optimality) equation
    \[
    V^*(s) = \max_a \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V^*(s')]
    \]
  - $Q^*$ function Bellman (optimality) equation
    \[
    Q^*(s,a) = \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma \max_{a'} Q^*(s',a')]
    \]
Let’s work out the Value function for a specific policy

actions: UP, DOWN, LEFT, RIGHT

- reward +1 at [4,3], -1 at [4,2]
- reward -0.04 for each step

\[ V^\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V^\pi(s')] = \sum_a \pi(a|s)Q^\pi(s,a) \]

1.0 + 0.8 * (+1-0.04 + 0) + 0.1 * (-0.04 + V^\pi([3,2])) + 0.1 * (-0.04 + V^\pi([3,3]))
Inference problem: given an MDP, how to compute its optimal policy?

- It suffices to compute its $Q^*$ function, because:

  $$\pi^*(s) = \arg \max_a Q^*(s, a)$$

- It suffices to compute its $V^*$ function, because:

  $$Q^*(s, a) = \sum_{s'} P(s'|s, a)[r(s, a, s') + \gamma V^*(s')]$$
MDP inference problem: Policy Evaluation (prediction) vs Policy Optimization (control)

- **Policy Evaluation (prediction):**
  - Simulate Bellman equation w.r.t. policy $\pi$ until it converges

- **Policy Optimization (control):**
  - Policy evaluation, policy improvement, PE, PI, …
  - Value iterations: simulate Bellman optimality equation
How to calculate value functions given MDP parameters? Policy Iterations and Value Iterations

• What are these algorithms for?
  – Algorithms of computing the \( V^* \) and \( Q^* \) functions from MDP parameters

• Policy Iterations
  \[
  \pi_0 \rightarrow^E V^{\pi_0} \rightarrow^I \pi_1 \rightarrow^E V^{\pi_1} \rightarrow^I \ldots \rightarrow^I \pi^* \rightarrow^E V^*
  \]

• Value iterations
  \[
  V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s, a)[r(s, a, s') + \gamma V_k(s')]
  \]

• How do we make sense of them?
  – Recursively applying the Bellman equations until convergence.
Multi-arm bandits: Problem setup

• No state. k-actions \( a \in \mathcal{A} = \{1, 2, \ldots, k\} \)

• You decide which arm to pull in every iteration \( A_1, A_2, \ldots, A_T \)

• You collect a cumulative payoff of \( \sum_{t=1}^{T} R_t \)

• The goal of the agent is to maximize the expected payoff.
  – For future payoffs?
  – For the expected cumulative payoff?
How do we measure the performance of an online learning agent?

- The notion of “Regret”:
  - I wish I have done things differently.
  - Comparing to the best actions in the hindsight, how much worse did I do.

- For MAB, the regret is defined as follow

\[
T \max_{a \in [k]} \mathbb{E}[R_t | a] - \sum_{t=1}^{T} \mathbb{E}_{a \sim \pi} \left[ \mathbb{E}[R_t | a] \right]
\]
Regret of different MAB algorithms

- Greedy: $O(T)$
- ExploreFirst: $O(T^{2/3}k^{1/3})$
- eps-Greedy: $O(T^{2/3}k^{1/3})$
- Upper Confidence Bound: $O(T^{1/2}k^{1/2})$
RL algorithms

- Model-based approach (plug-in an empirically estimated MDP, run VI / PI)

- Model-free approach:
  - Monte Carlo (average converges to mean) e.g., First visit Monte Carlo
  - Combining Monte Carlo with Dynamic Programming (e.g., VI)
  - Temporal difference learning
Revisit the dynamic programming approach

- Policy Evaluation

\[ V_{k+1}^{\pi}(s) \leftarrow \sum_{a} \pi(a|s) \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V_{k}^{\pi}(s')] \]

- Policy improvement

\[ \pi'(s) = \text{arg max}_{a} Q^{\pi}(s,a) \]
\[ = \text{arg max}_{a} \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V_{k}^{\pi}(s')] \]

- Value iterations

\[ V_{k+1}(s) \leftarrow \max_{a} \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V_{k}(s')] \]

*We do not have the MDP parameters in RL!"
Reinforcement learning agents have “online” access to an environment

- State, Action, Reward
- Unknown reward function, unknown state-transitions.
- Agents can “act” and “experiment”, rather than only doing offline planning.
TD policy optimization (TD-control)

- **SARSA (On-Policy TD-control)**
  - Update the Q function by bootstrapping Bellman Equation
  \[
  Q(S, A) \leftarrow Q(S, A) + \alpha \left[ R + \gamma Q(S', A') - Q(S, A) \right]
  \]
  - Choose the next A’ using Q, e.g., eps-greedy.

- **Q-Learning (Off-policy TD-control)**
  - Update the Q function by bootstrapping Bellman Optimality Eq.
  \[
  Q(S, A) \leftarrow Q(S, A) + \alpha \left[ R + \gamma \max_a Q(S', a) - Q(S, A) \right]
  \]
  - Choose the next A’ using Q, e.g., eps-greedy, or any other policy.
Q-Learning with function approximation

\[ Q(s, a) = w_1 f_1(s, a) + w_2 f_2(s, a) + \ldots + w_n f_n(s, a) \]

- Q-learning with linear Q-functions:

  transition = (s, a, r, s')
  difference = \[ r + \gamma \max_{a'} Q(s', a') \] - Q(s, a)

  \[ Q(s, a) \leftarrow Q(s, a) + \alpha \text{[difference]} \]

  Exact Q's

  \[ w_i \leftarrow w_i + \alpha \text{[difference]} f_i(s, a) \]

  Approximate Q's

- Intuitive interpretation:
  - Adjust weights of active features
  - E.g., if something unexpectedly bad happens, blame the features that were on: disprefer all states with that state’s features

- Formal justification: online least squares (Read the textbook!)
Most important concepts in MDP / RL

• Make sure you understand
  – Problem setup, evaluation criteria
  – Definition of the policy, state, action, immediate reward, value, value function …
  – Shapes of the MDP parameters

• Bellman equations
  – How do Policy / Value iterations work?

• RL algorithms:
  – Model-based estimation of MDP parameters
  – Monte Carlo estimate of V function and Q function
  – Q-Learning: SGD-style Stochastic simulation of Bellman equations
Lecture 16-17: Logic

- Logic agent
  - Know how to play, e.g., Minesweeper and know how to explain your reasons.

- Knowledge Base
  - Tell operation
  - Ask operation

- Components of a formal logic system
  - Syntax, Semantics

- Definition of Valid / Unsatisfiable

- First order logic: translate natural language into FOL sentences
KB Agents

- Need a formal logic system to work
- Need a data structure to represent known facts
- Need an algorithm to answer ASK questions
Syntax and semantics

• Two components of a logic system

• Syntax --- How to construct sentences
  – The symbols
  – The operators that connect symbols together
  – A precedence ordering

• Semantics --- Rules the assignment of sentences to truth
  – For every possible worlds (or “models” in logic jargon)
  – The truth table is a semantics
Entailment

A is entailed by B, if A is true in all possible worlds consistent with B under the semantics.
Inference procedure

- Inference procedure
  - Rules (algorithms) that we apply (often recursively) to derive facts from other facts.
  - Could be specific to a particular set of semantics, a particular realization of the world.

- Soundness and completeness of an inference procedure
  - Soundness: All truth discovered are valid.
  - Completeness: All truth that are entailed can be discovered.
Propositional Logic

• Syntax:
  – True, false, propositional symbols
  – ( ), ¬ (not), ∧ (and), ∨ (or), ⇒ (implies), ⇔ (equivalent)

• Semantics:
  – Five rules (the following truth table)

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>Q</td>
<td>¬P</td>
<td>P ∧ Q</td>
<td>P ∨ Q</td>
<td>P ⇒ Q</td>
<td>P ⇔ Q</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>----</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>

• Inference rules:
  – Modus Pronens etc / Resolution
Propositional logic agent

- **Representation:** Conjunctive Normal Forms
  - Represent them in a data structure: a list, a heap, a tree?
  - Efficient TELL operation

- **Inference:** Solve ASK question
  - Use “Resolution” only on CNFs is Sound and Complete.
  - Equivalent to SAT, NP-complete, but good heuristics / practical algorithms exist

- **Possible answers to ASK:**
  - Valid, Satisfiable, Unsatisfiable
First order logic

• More expressive language
  – Relations and functions of objects.
  – Quantifiers such as, All, Exists.

• Easier to construct a KB.
  – Need much smaller number of sentences to capture a domain.

• Follow the same structure: Symbols, Semantics

• Dedicated inference algorithms

• (FOL inference is not covered in the Final)
Potential types of questions in FOL

- Translate FOL sentence to natural language or the other way round.

- Translate the rule of a simple game to FOL.
  - e.g., how would you describe the rules of Wumpus world using FOL?
FOL Description of a Wumpus world

- What are the rules? How to write them down in FOL?

"A breeze is smelt at a location if there is a pit in an adjacent location"

\[ \forall y \ Breezy(y) \implies \exists x \ Pit(x) \land \text{Adjacent}(x, y) \]

\[ \forall x, y \ Pit(x) \land \text{Adjacent}(x, y) \implies Breezy(y) \]

\[ \forall y \ Breezy(y) \iff [\exists x \ Pit(x) \land \text{Adjacent}(x, y)] \]
Lecture 18: Responsible AI

- What are the typical pitfalls in AI applications
  - Privacy: Data sharing, data use, data ownership
  - Fairness of AI Decision making: Recidivism prediction, Admission / Recruiting
  - Polarizing effects of recommendation systems
  - Fake news / fake videos
  - Social impacts: unemployment / rich gets richer
  - Robustness and Safety: adversarial examples, self-driving cars
  - Generative models: ethics and copyright

- What are your thoughts on overcoming them?
  - Potentially a case / short essay question.
Courses to take next on the AI track

• CS165B Machine Learning (almost every quarter)
  – Undergraduate level intro to ML
  – Expanded version of Part 1 in this course
  – I am teaching 165B this coming Fall 2023

• Other advanced topics courses in AI:
  – Deep Learning, Natural Language Processing, Computer vision, Convex Optimization, Reinforcement Learning
Thank you and stay in touch!

• It’s my pleasure to work with you!
• I hope the course is / will be useful.

• AI Research at UCSB
  – Machine Learning Lab
  – Natural Language Processing Lab
  – Center for Responsible Machine Learning
  – Center for Information Technology and Society
  – The Mellichamp Initiative in Mind & Machine Intelligence
  – Data Science Initiative