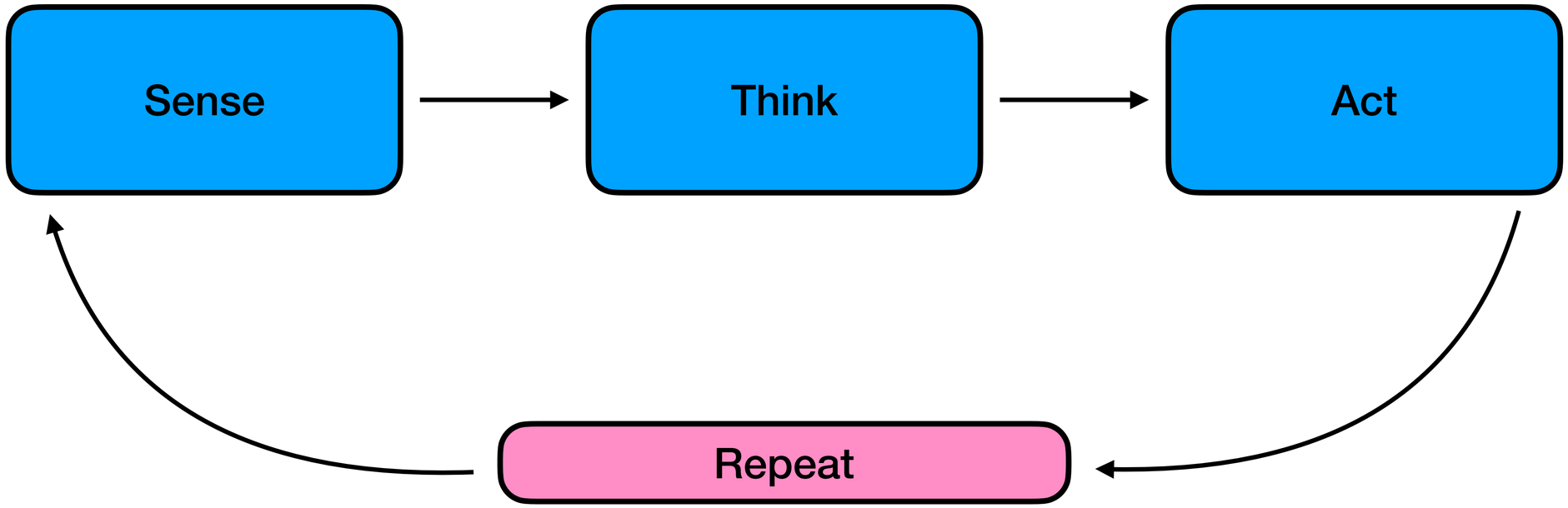
Introduction to Robot Intelligence
[Spring 2023]

Introduction to Learning

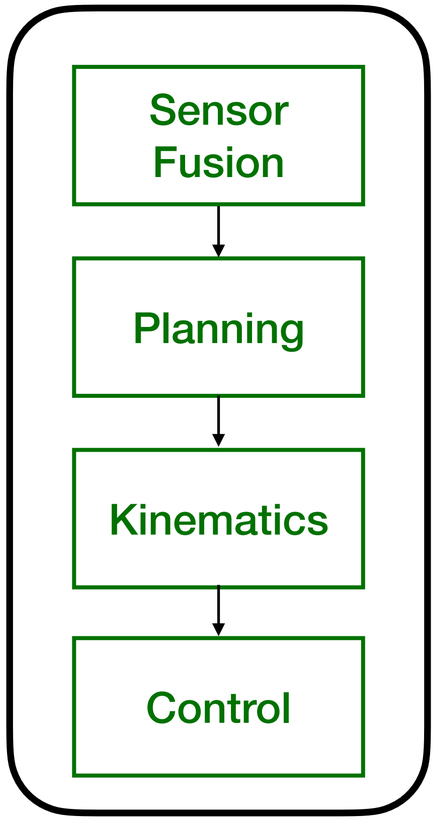
May 2, 2023

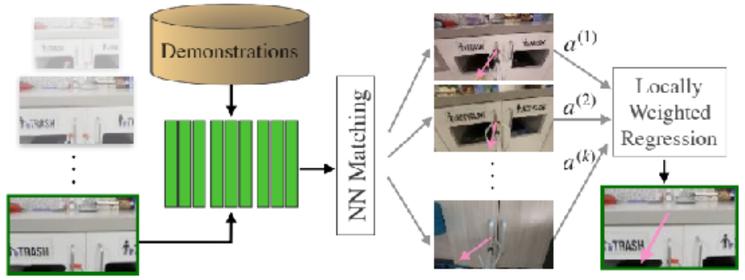
Lerrel Pinto

What is a robot?

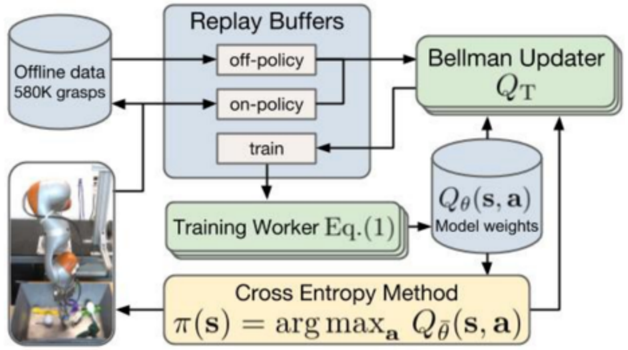


The role of learning

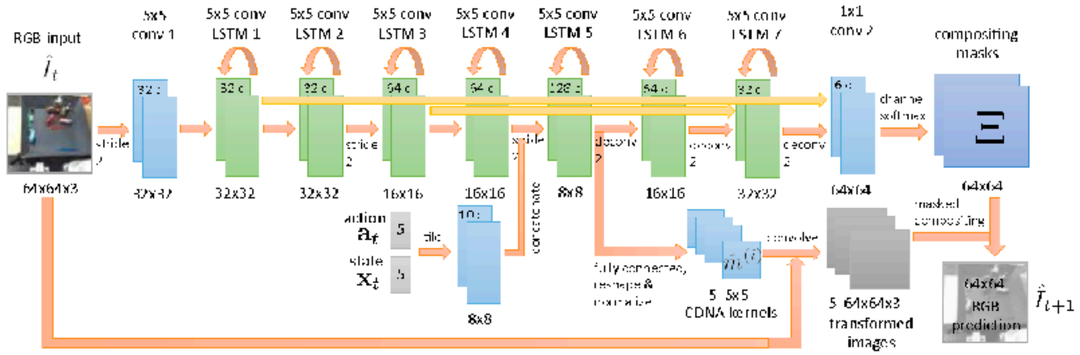




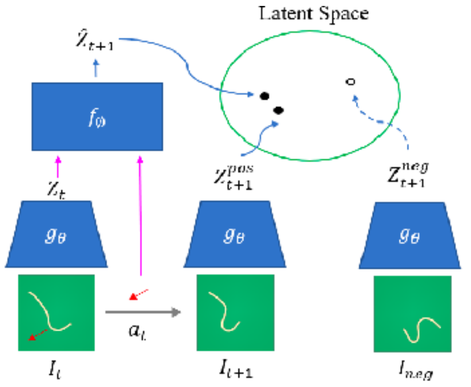
Pari et al. 2021



Kalashnikov et al. 2018



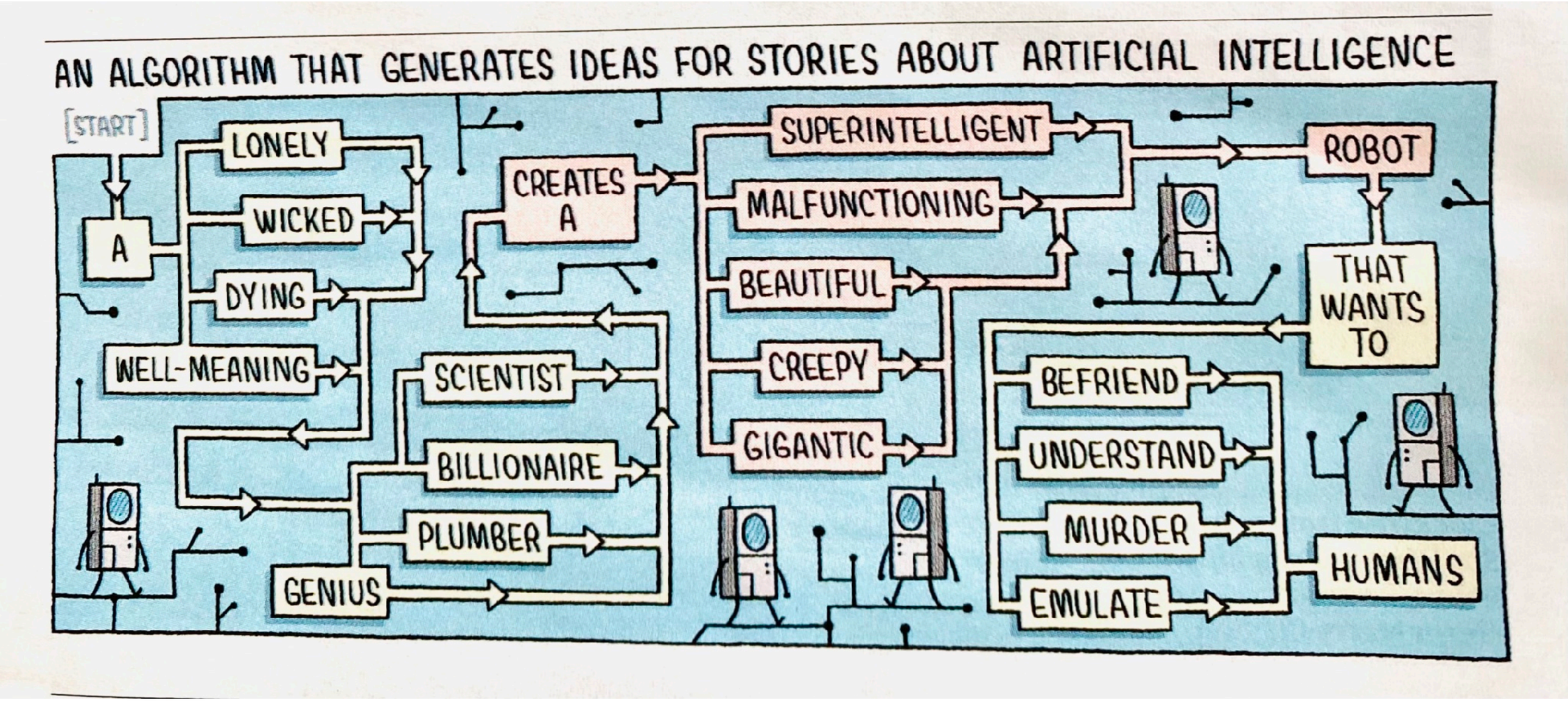
Finn et al. 2016



Yan et al. 2020

What is Machine Learning?

Good Old Fashioned AI (GOFAI)



Good Old Fashioned AI (GOFAI)

Good Old Fashioned AI (GOFAI)

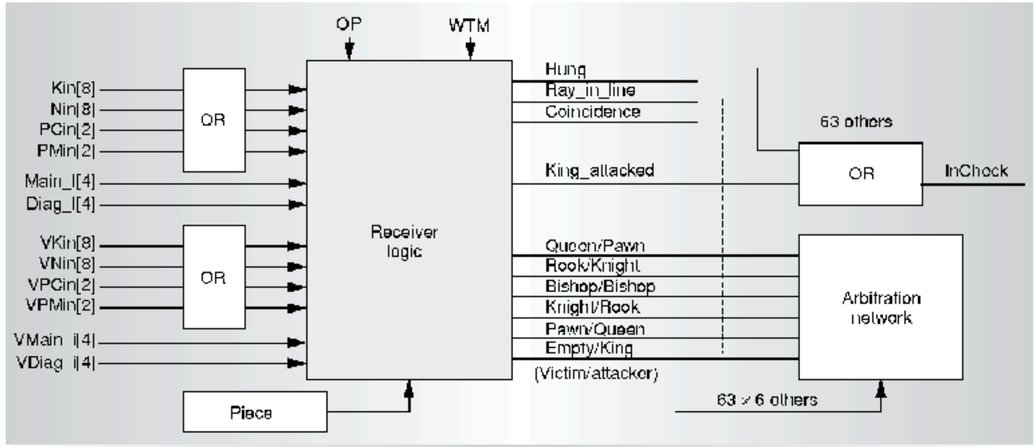


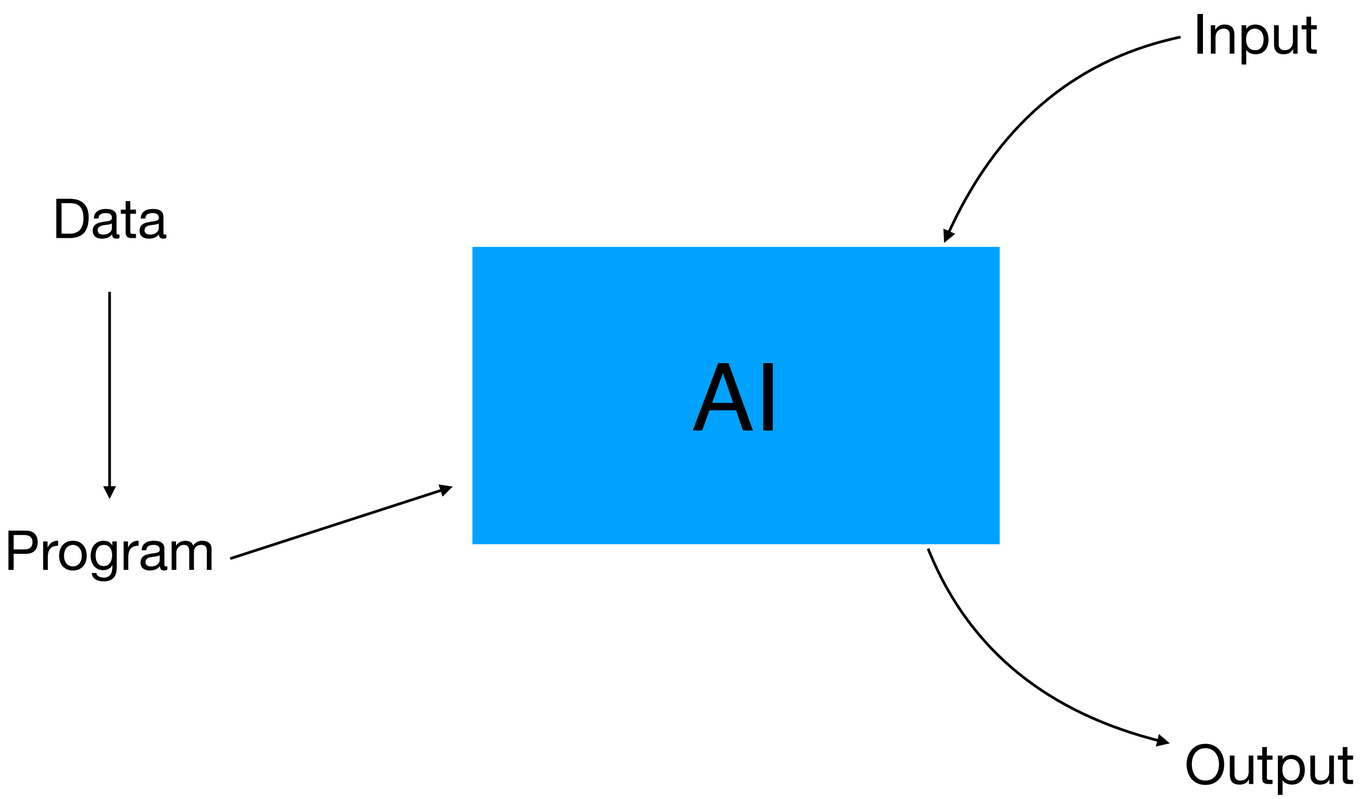
Figure 4. The receiver.

Software [edit]

Deep Blue's evaluation function was initially written in a generalized form, with many to-be-determined parameters (e.g., how important is a safe king position compared to a space advantage in the center, etc.). Values for these parameters were determined by analyzing thousands of master games. The evaluation function was then split into 8,000 parts, many of them designed for special positions. The opening book encapsulated more than 4,000 positions and 700,000 grandmaster games, while the endgame database contained many six-piece endgames and all five and fewer piece endgames. An additional database named the "extended book" summarizes entire games played by Grandmasters. The system combines its searching ability of 200 million chess positions per second with summary information in the extended book to select opening moves.[43]

Before the second match, the program's rules were fine-tuned by grandmaster Joel Benjamin. The opening library was provided by grandmasters Miguel Illescas, John Fedorowicz, and Nick de Firmian.[44] When Kasparov requested that he be allowed to study other games that Deep Blue had played so as to better understand his opponent, IBM refused, leading Kasparov to study many popular PC chess games to familiarize himself with computer gameplay.[45]

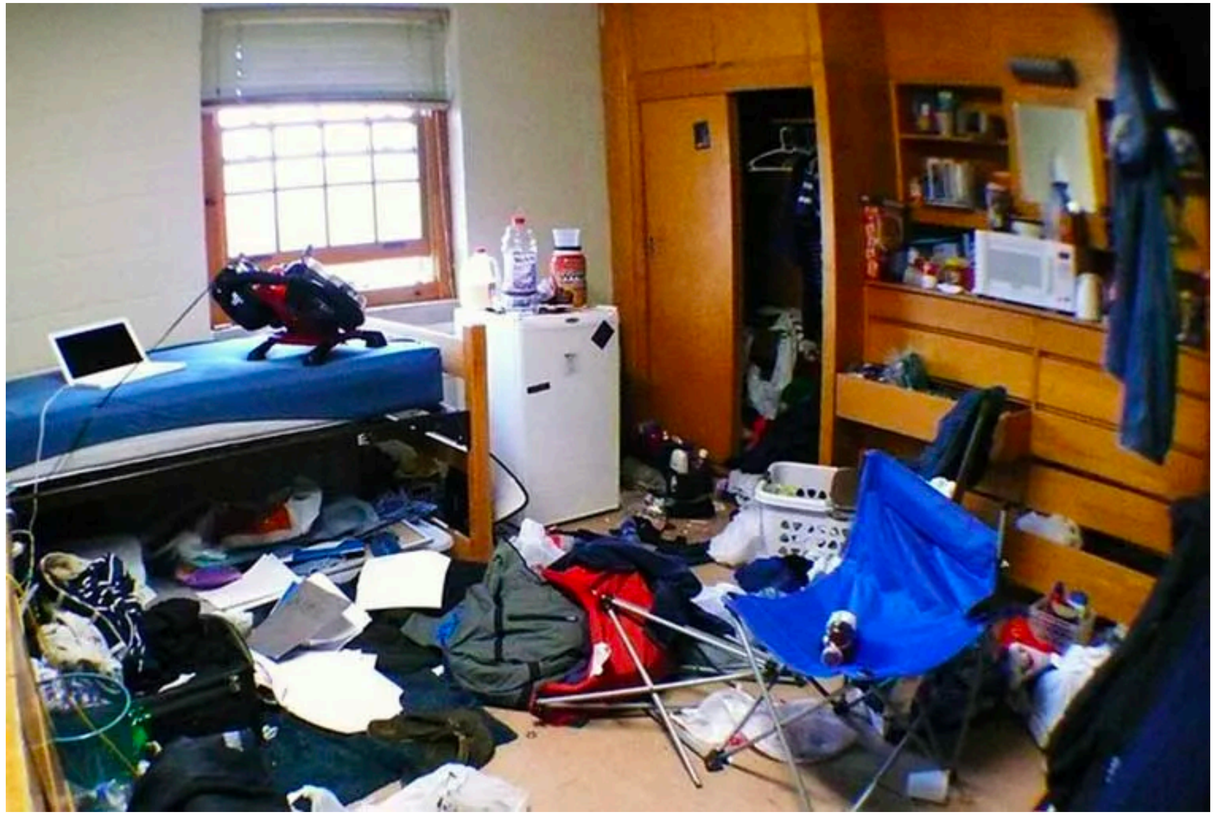
Good Old Fashioned AI (GOFAI)



Good Old Fashioned AI (GOFAI) – Pitfalls

Good Old Fashioned AI (GOFAI) – Pitfalls

Good Old Fashioned AI (GOFAI) – Pitfalls

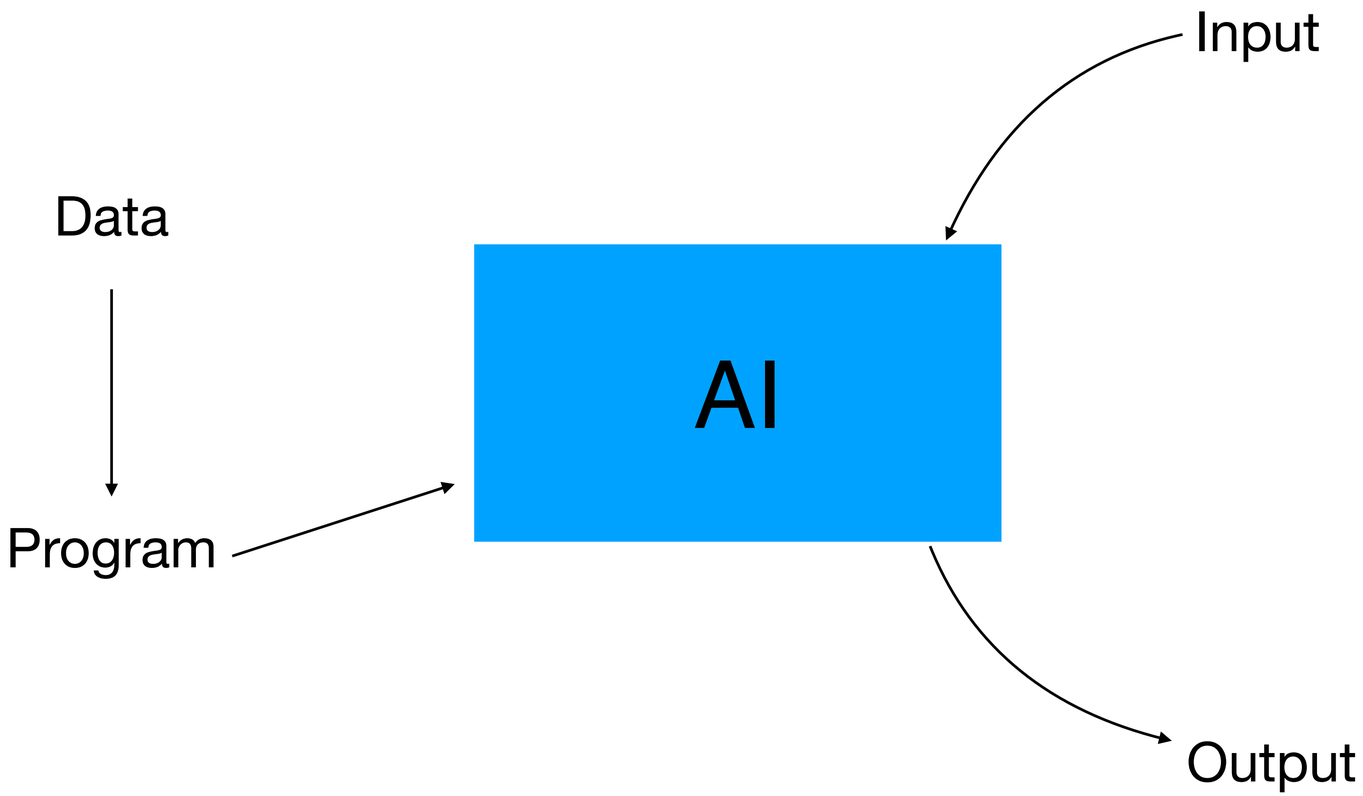


Machine Learning

Wikipedia def.: Machine learning (ML) is the study of computer algorithms that can improve automatically through experience and by the use of data.

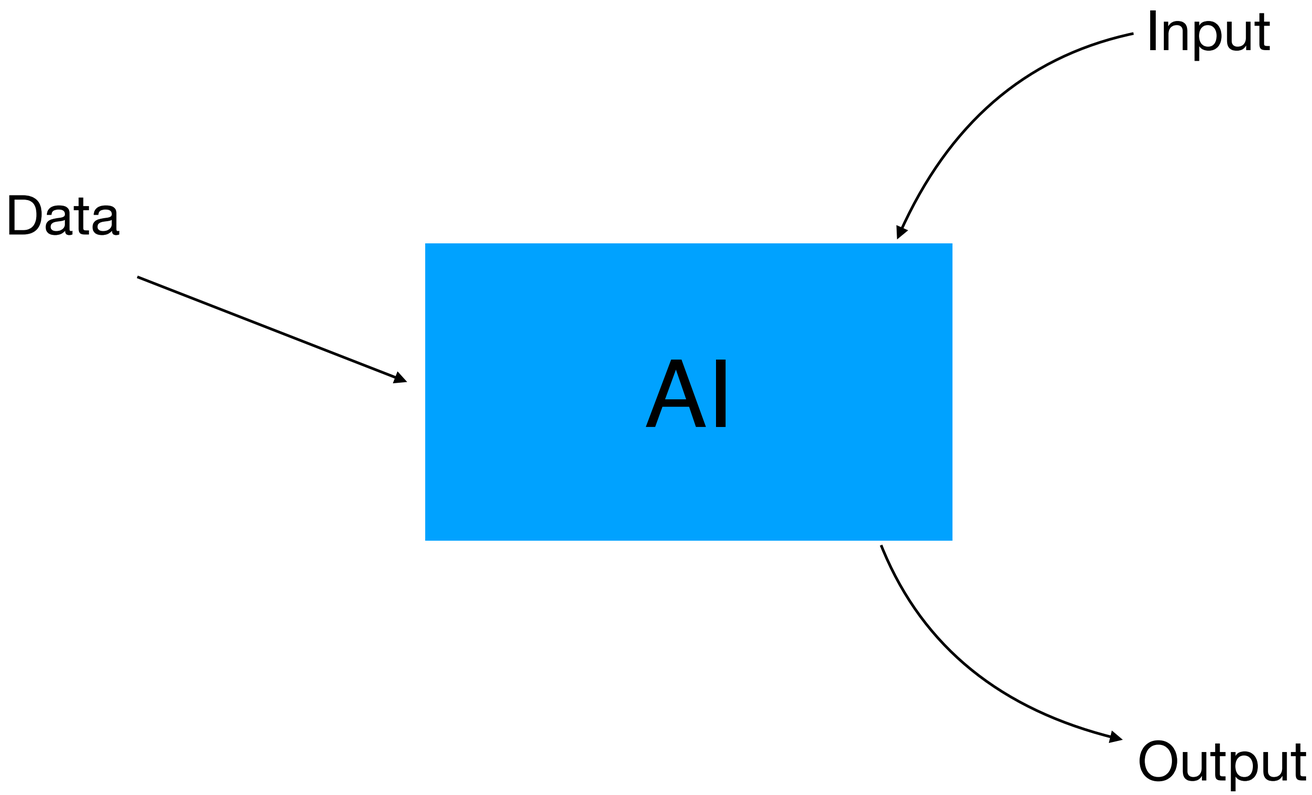
Machine Learning

Wikipedia def.: Machine learning (ML) is the study of computer algorithms that can improve automatically through experience and by the use of data.



Machine Learning

Wikipedia def.: Machine learning (ML) is the study of computer algorithms that can improve automatically through experience and by the use of data.



Machine Learning

Wikipedia def.: Machine learning (ML) is the study of computer algorithms that can improve automatically through experience and by the use of data.

Definition by Tom Mitchell (1998):

Machine Learning is the study of algorithms that

- improve their performance P
- at some task T
- with experience E.

A well-defined learning task is given by <P, T, E>.

Types of Machine Learning

- Supervised Learning
    - Given: Training data with output labels.

Types of Machine Learning

- Supervised Learning
    - Given: Training data with output labels.
- Unsupervised Learning
    - Given: Training data without output labels.

Types of Machine Learning

- Supervised Learning
    - Given: Training data with output labels.
- Unsupervised Learning
    - Given: Training data without output labels.
- Reinforcement Learning
    - Given: Access to an environment with ‘reward’ labels.

Case study 1: Linear Regression (Supervised Learning)

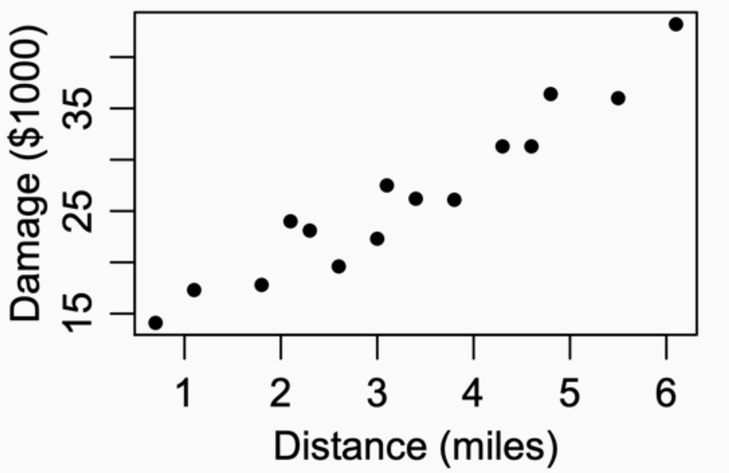
Regression Problem

- Given $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots, (x_n, y_n)\}$,
- Return $y = f(x)$.

Regression Problem

- Given $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots, (x_n, y_n)\}$,
- Return $y = f(x)$.

Distance (mile)	Damage ($1000)
0.7	14.1
1.1	17.3
1.8	17.8
2.1	24.0
2.3	23.1
2.6	19.6
3.0	22.3
3.1	27.5
3.4	26.2
3.8	26.1
4.3	31.3
4.6	31.3
4.8	36.4
5.5	36.0
6.1	43.2

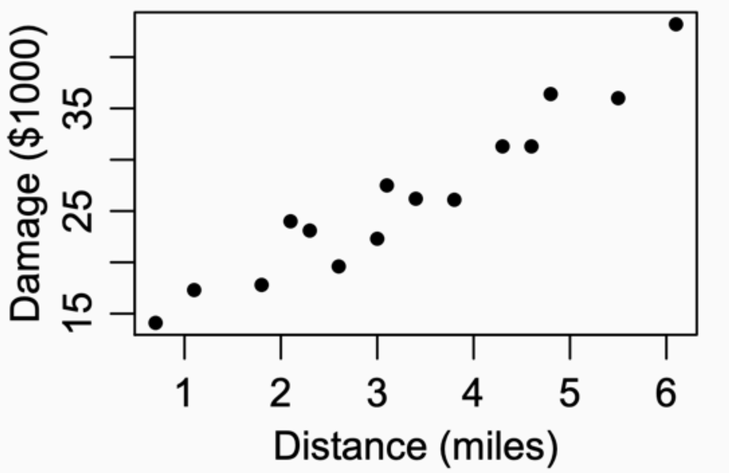


Source: https://galton.uchicago.edu/~yibi/teaching/stat220/17aut/Lectures/L24.pdf

Regression Problem

- Lets assume $f(x)$ is linear.
- $f(x) = ax + b$

Distance (mile)	Damage ($1000)
0.7	14.1
1.1	17.3
1.8	17.8
2.1	24.0
2.3	23.1
2.6	19.6
3.0	22.3
3.1	27.5
3.4	26.2
3.8	26.1
4.3	31.3
4.6	31.3
4.8	36.4
5.5	36.0
6.1	43.2

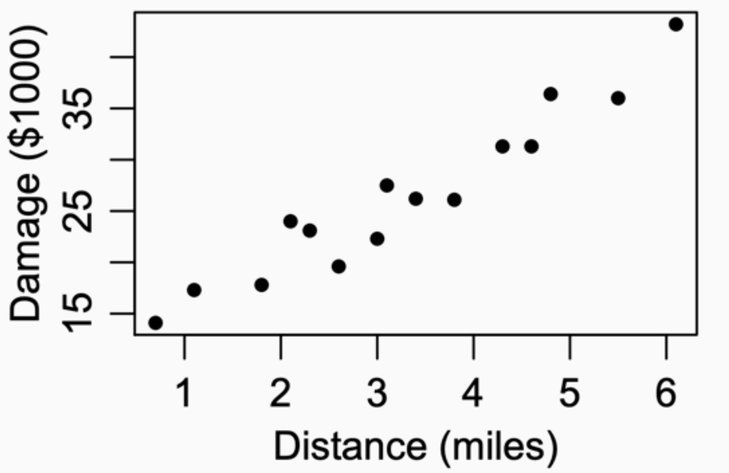


Source: https://galton.uchicago.edu/~yibi/teaching/stat220/17aut/Lectures/L24.pdf

Regression Problem

- Lets assume $f(x)$ is linear.
- $f(x) = ax + b$

Distance (mile)	Damage ($1000)
0.7	14.1
1.1	17.3
1.8	17.8
2.1	24.0
2.3	23.1
2.6	19.6
3.0	22.3
3.1	27.5
3.4	26.2
3.8	26.1
4.3	31.3
4.6	31.3
4.8	36.4
5.5	36.0
6.1	43.2



Source: https://galton.uchicago.edu/~yibi/teaching/stat220/17aut/Lectures/L24.pdf

Regression Problem

- Lets assume $f(x)$ is linear.
- What if x and y are multi-dimensional?

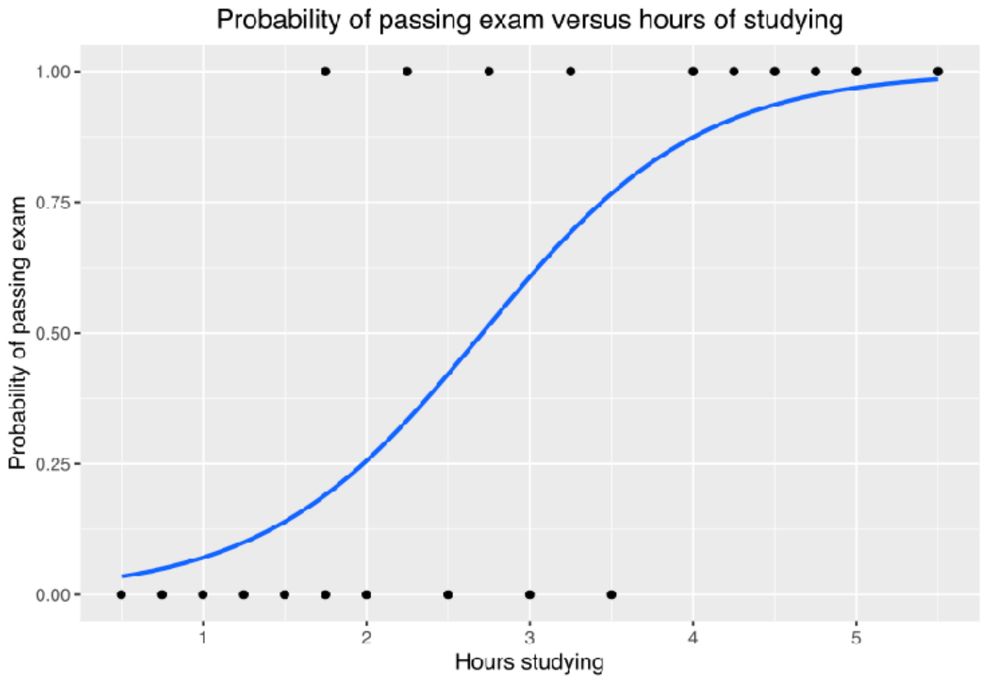
Case study 2: Logistic Regression (Supervised Learning)

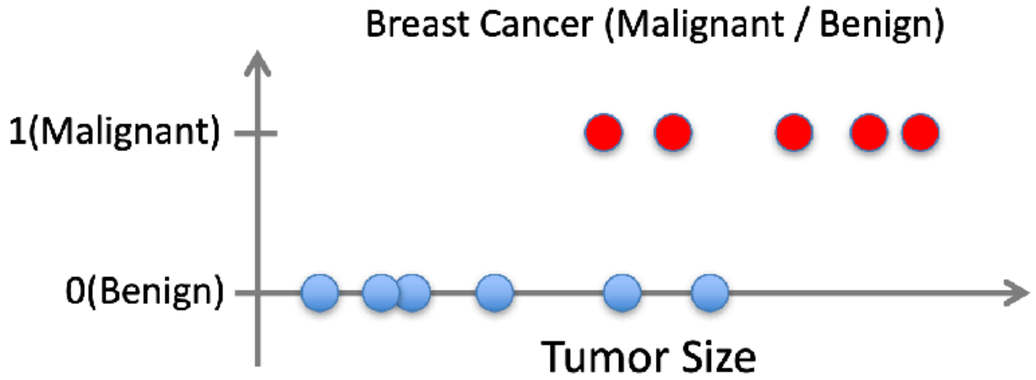
Classification Problem

- Given $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots, (x_n, y_n)\}$,
- Return $y = f(x). y \in \{0,1\}$

Classification Problem

- Given $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots, (x_n, y_n)\}$,
- Return $y = f(x). y \in \{0,1\}$



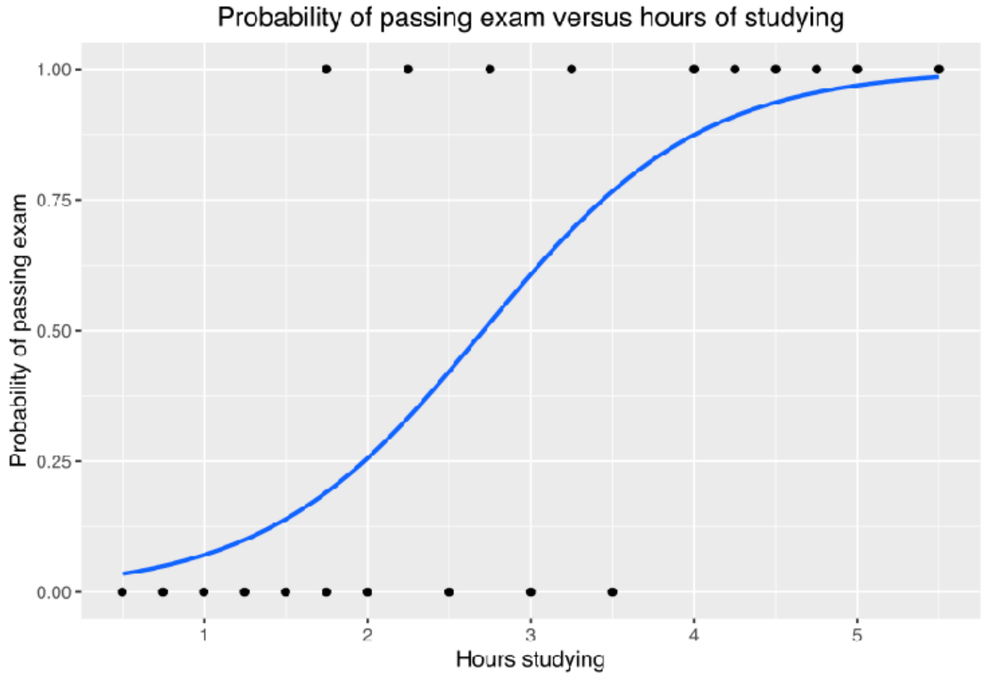


Source: https://en.wikipedia.org/wiki/Logistic_regression; Andrew Ng.

Classification Problem

- Assume $y \equiv p(x)$ is a logistic function.
- $p(x) = \dfrac{1}{1 + e^{-(ax+b)}}$

Hours (x_k)	0.50	0.75	1.00	1.25	1.50	1.75	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	4.00	4.25	4.50	4.75	5.00	5.50
Pass (y_k)	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	1	1	1	1	1



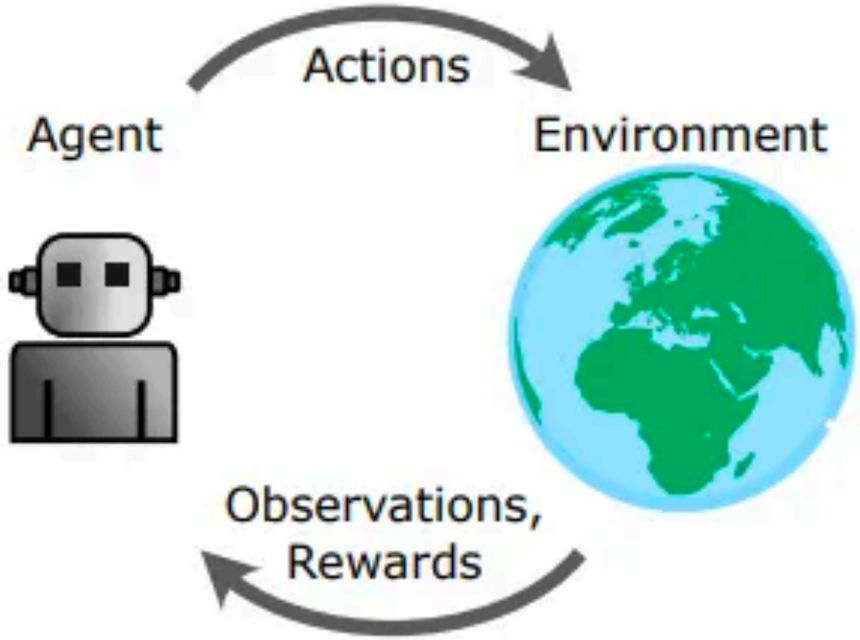
Source: https://en.wikipedia.org/wiki/Logistic_regression.

Classification Problem

- Assume $y \equiv p(x)$ is a logistic function.
- $p(x) = \dfrac{1}{1 + e^{-(ax+b)}}$

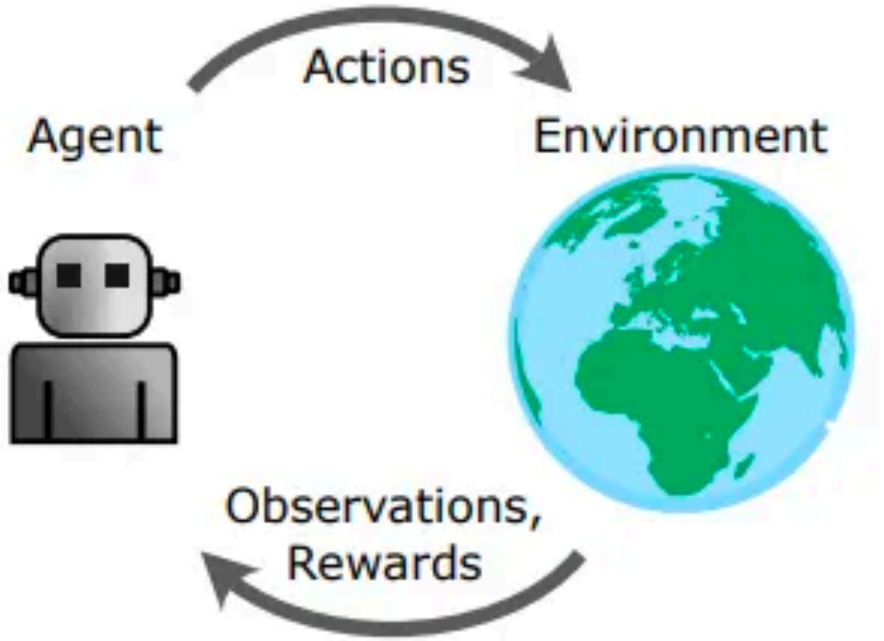
Source: https://en.wikipedia.org/wiki/Logistic_regression.

RL - preliminaries

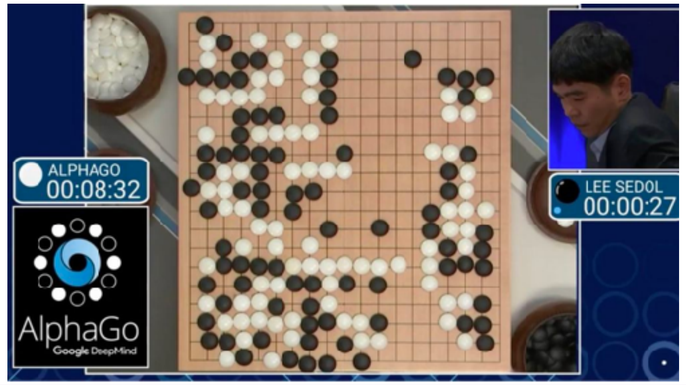


Maximize Rewards

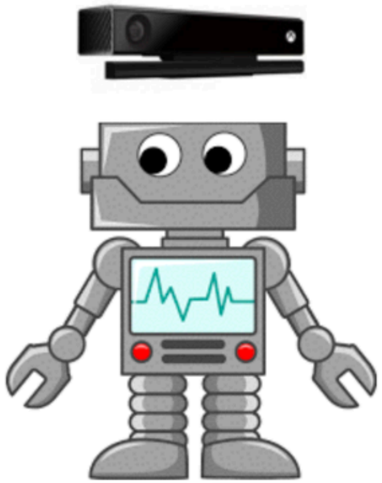
RL - preliminaries



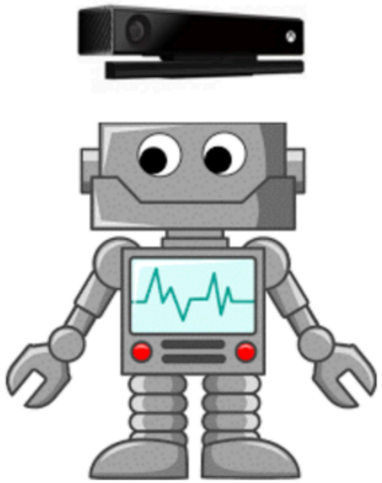
Maximize Rewards



Life as a Robot

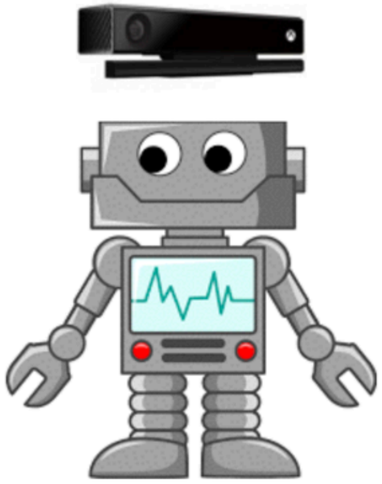


Life as a Robot



Life is a sequence of what it ‘observes’ and what it ‘does’

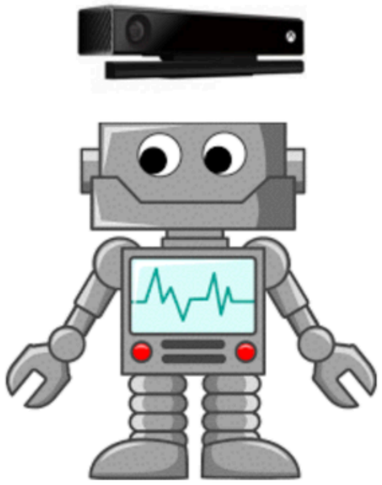
Life as a Robot



Life is a sequence of what it 'observes' and what it 'does'

$$(o_1, a_1), (o_2, a_2), ..., (o_T, a_T)$$

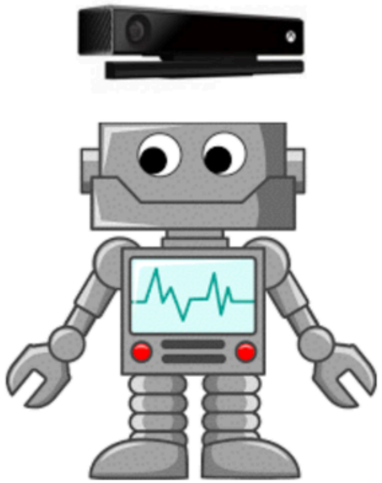
Life as a Robot



The effect of an action can be lifelong

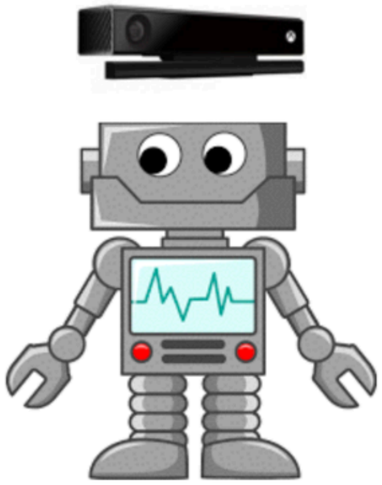
$$(o_1, a_1) \rightarrow (o_2, a_2) \rightarrow\rightarrow\rightarrow (o_T, a_T)$$

Life as a Robot



- To be successful our robot needs to make good decisions

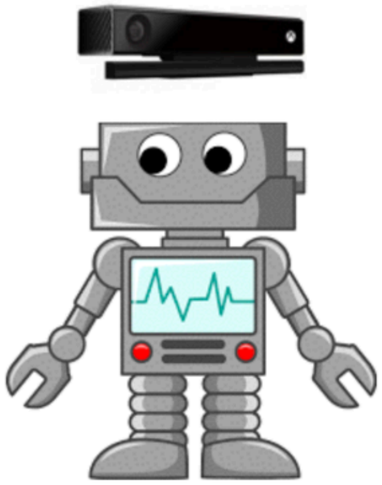
Life as a Robot



- To be successful our robot needs to make good decisions
- In the general case, at time t use

  $(o_1, o_2, ..., o_t)$ to make decision a_t

Life as a Robot

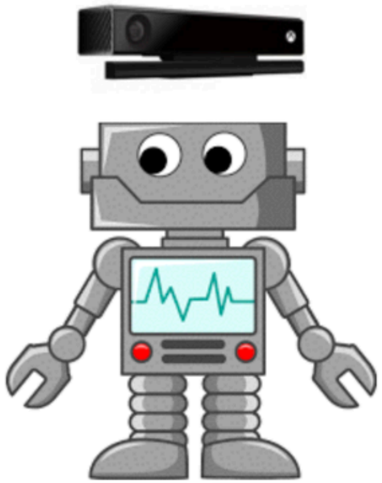


- To be successful our robot needs to make good decisions
- In the general case, at time t use

  $(o_1, o_2, ..., o_t)$ to make decision a_t

- The world also gives some notion of success through 'rewards' r_t

Life as a Robot



- To be successful our robot needs to make good decisions
- In the general case, at time t use

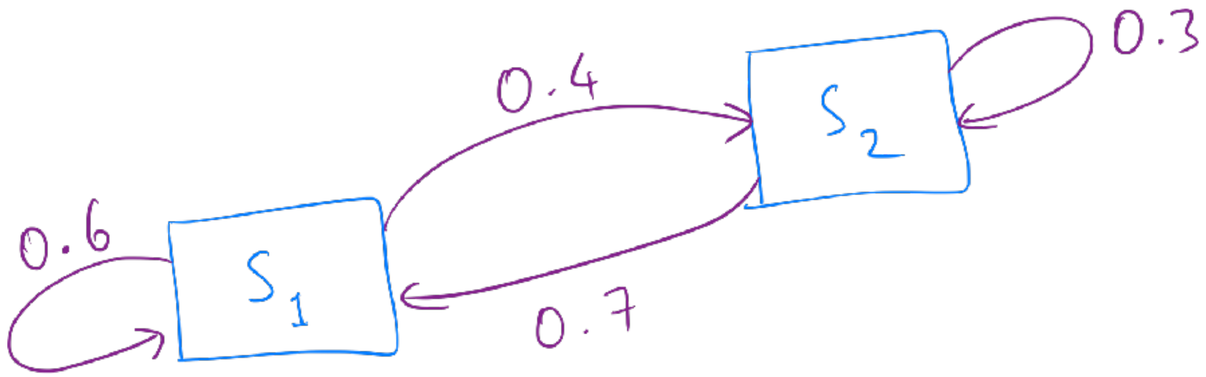
  $(o_1, o_2, ..., o_t)$ to make decision a_t

- The world also gives some notion of success through 'rewards' r_t

In general case maximize $\sum_{t=0}^{T} r_t$

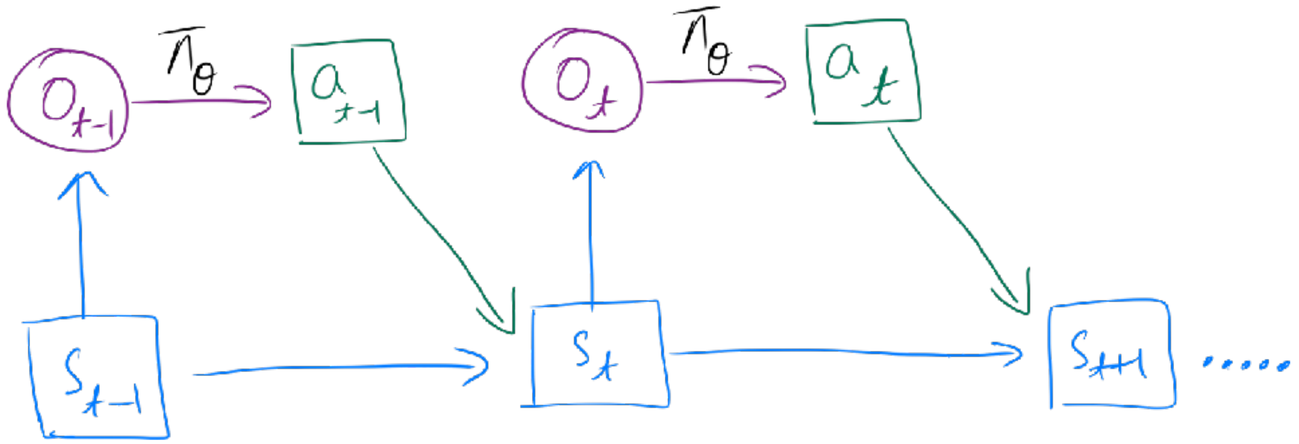
Markov Decision Process (MDP)

Markov Chain

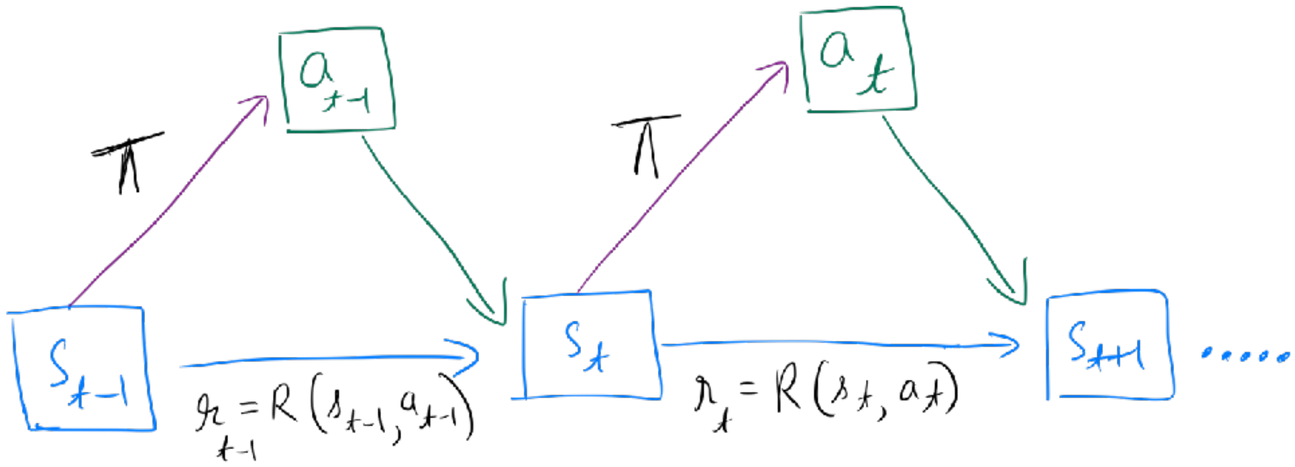


What is missing for decision making?

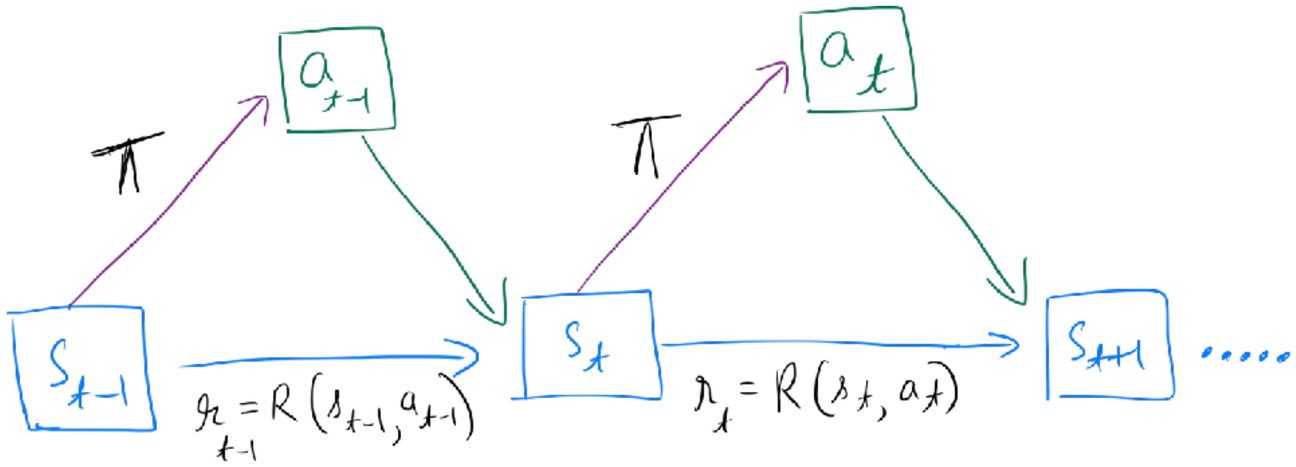
MDPs



MDPs



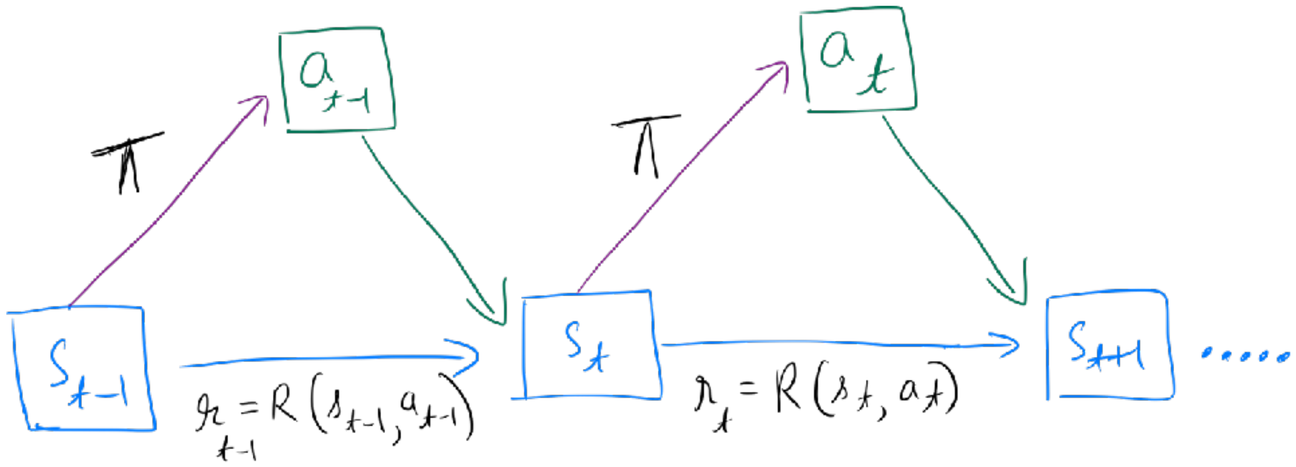
MDPs



- Finite set of states S
- Finite set of actions A
- Immediate reward function $R : S \times A \rightarrow$ Reals
- Transition (next-state) function $T: S \times A \rightarrow S$

More generally, R and T are treated as stochastic

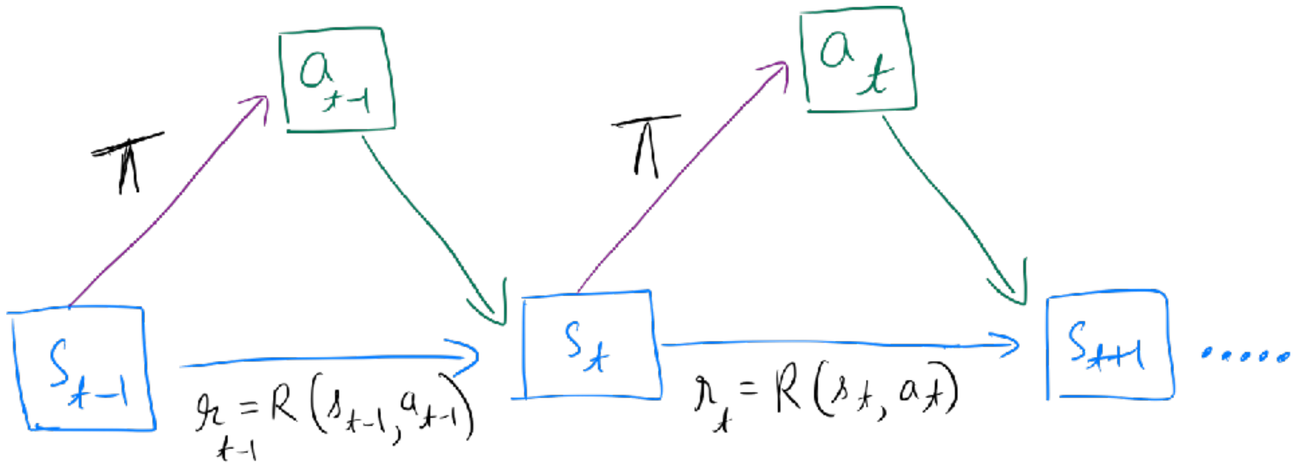
MDPs



Markov Decision Processes:

- A fundamental framework for reinforcement learning

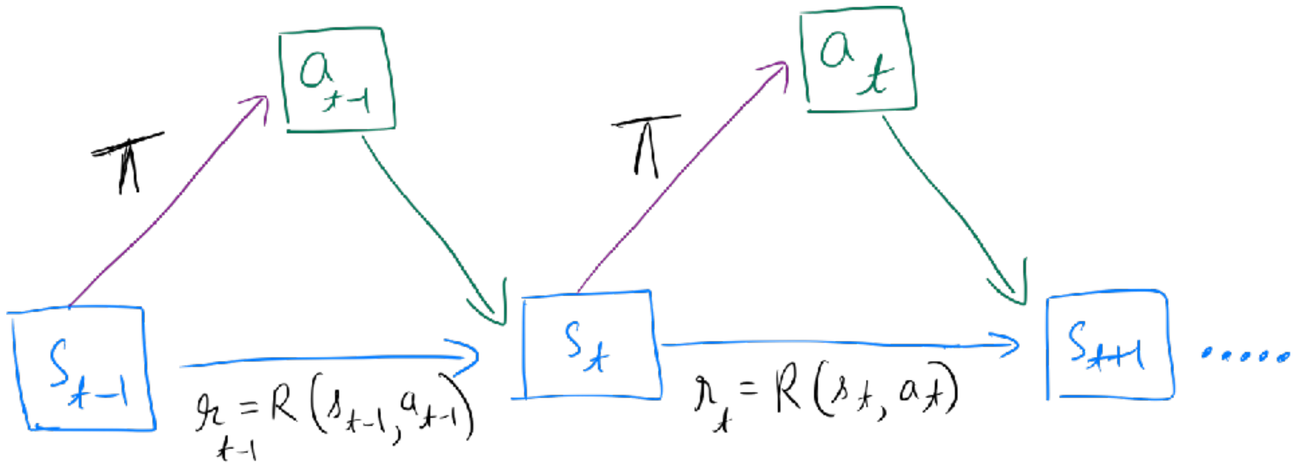
MDPs



Markov Decision Processes:

- A fundamental framework for reinforcement learning
- History
  - 1950s: early works of Bellman and Howard
  - 50s-80s: theory, basic set of algorithms, applications
  - 90s: MDPs in AI literature

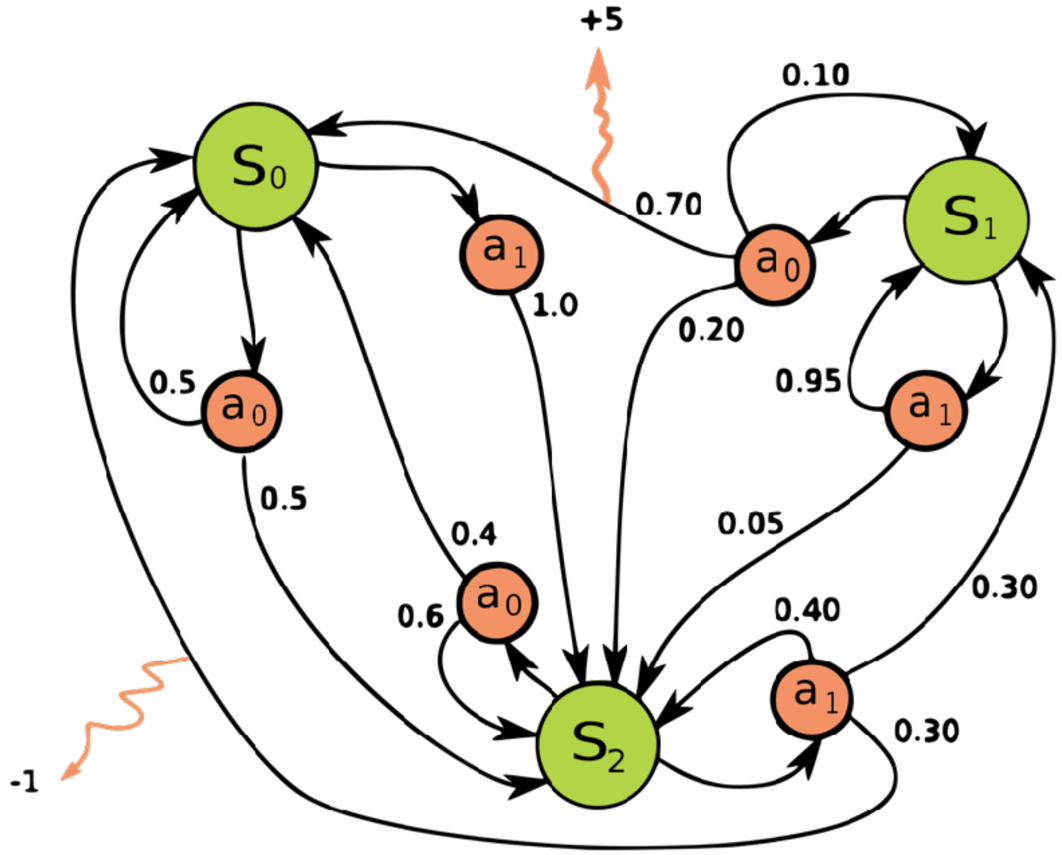
MDPs



Markov Decision Processes:

- A fundamental framework for reinforcement learning
- History
  - 1950s: early works of Bellman and Howard
  - 50s-80s: theory, basic set of algorithms, applications
  - 90s: MDPs in AI literature
- MDPs in AI
  - reinforcement learning
  - probabilistic planning

MDPs

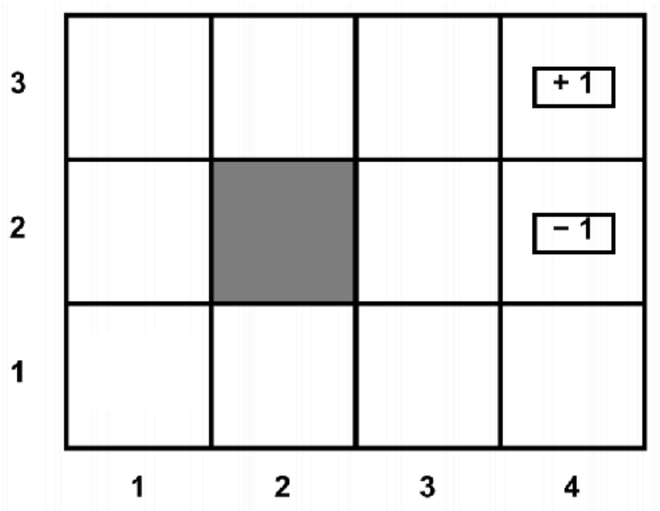


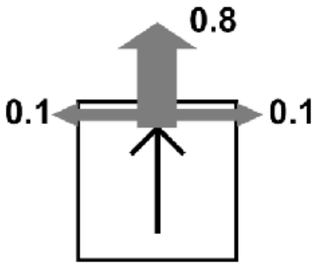
Number of states: 3
Number of actions: 2

Slide credits: Waldo Alvarez

MDPs

- The agent lives in a grid
- Walls block the agent's path
- The agent's actions do not always go as planned:
  - 80% of the time, the action North takes the agent North (if there is no wall there)
  - 10% of the time, North takes the agent West; 10% East
  - If there is a wall in the direction the agent would have been taken, the agent stays put
- Big rewards come at the end



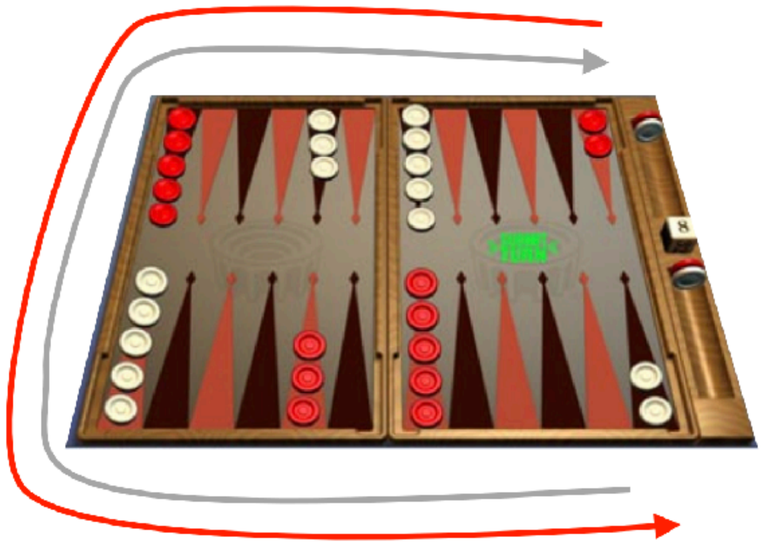


Number of states: 11
Number of actions: 4

Slide credits: Pieter Abbeel

MDPs

Backgammon



STATES: configurations of the playing board ($\approx 10^{20}$)

ACTIONS: moves

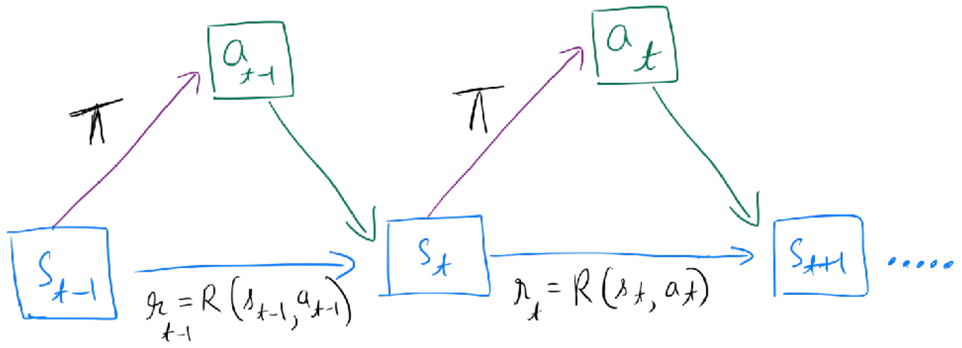
REWARDS: win: +1
lose: −1
else: 0

a "big" game

Slide credits: Rich Sutton

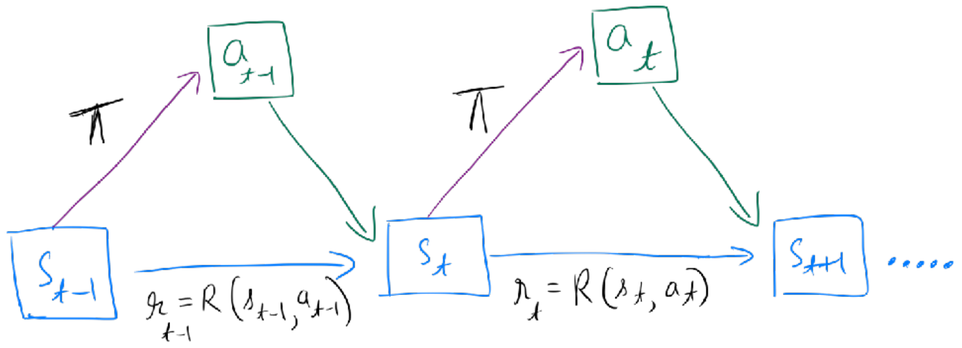
Questions?

MDPs



Whats a Policy?

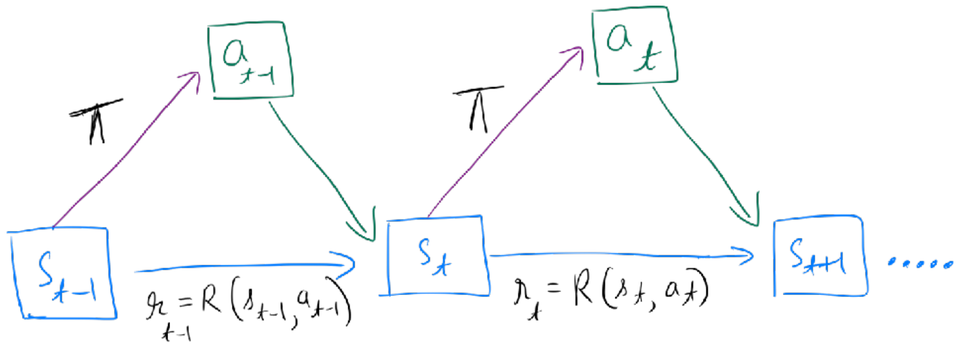
MDPs



Whats a Policy?

Overall objective is to determine a policy $\pi: S \rightarrow A$
such that some measure of cumulative reward is optimized

MDPs

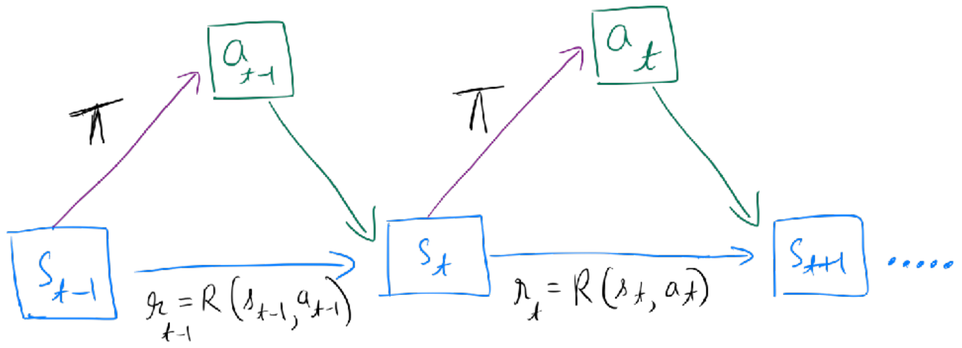


Whats a Policy?

Overall objective is to determine a policy $\pi: S \rightarrow A$
such that some measure of cumulative reward is optimized

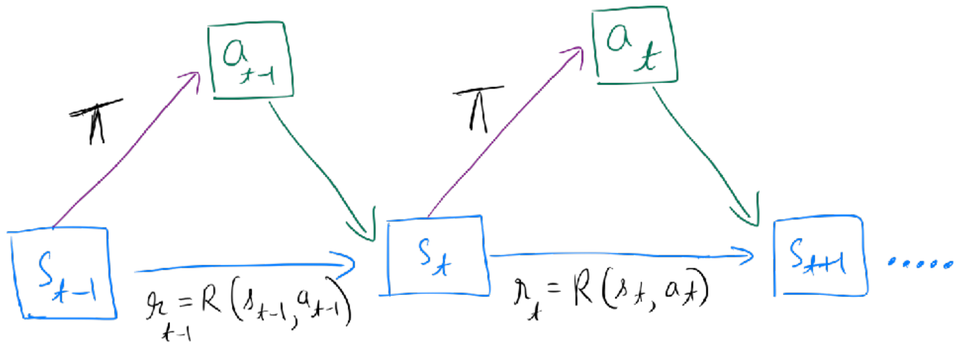
If agent in state:	Apply action:
s_1	a_5
s_2	a_1
s_3	a_9
s_4	a_1

MDPs



Objective function?

MDPs



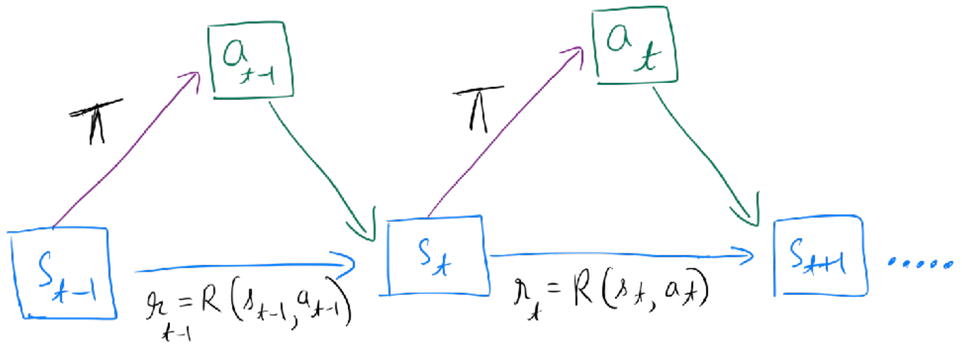
Objective function?

Goal: Learn to choose actions that maximize the cumulative reward

$$R = r_0 + \gamma r_1 + \gamma^2 r_2 + ...$$

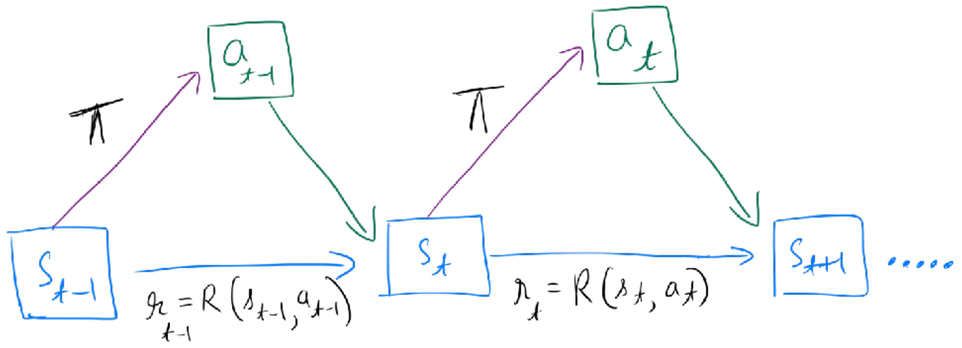
where $0 \leq \gamma \leq 1$

MDPs



Value function

MDPs

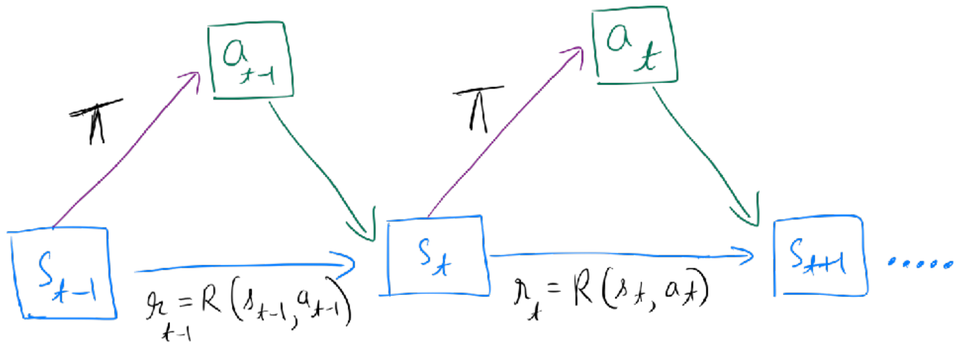


Value function

A (state) value function V is a function mapping states to real numbers:

$$V : S \rightarrow \text{Reals}$$

MDPs

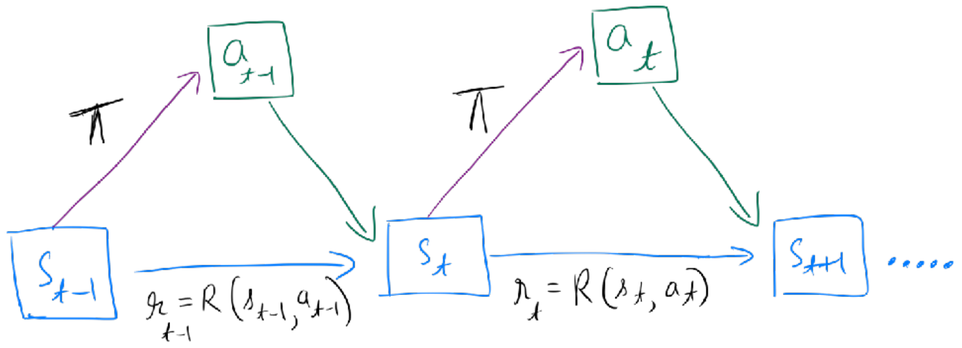


Value function

For any policy π, define the return to be the function $V : S \rightarrow$ Reals assigning to each state the quantity

$$V^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t r_t$$

MDPs



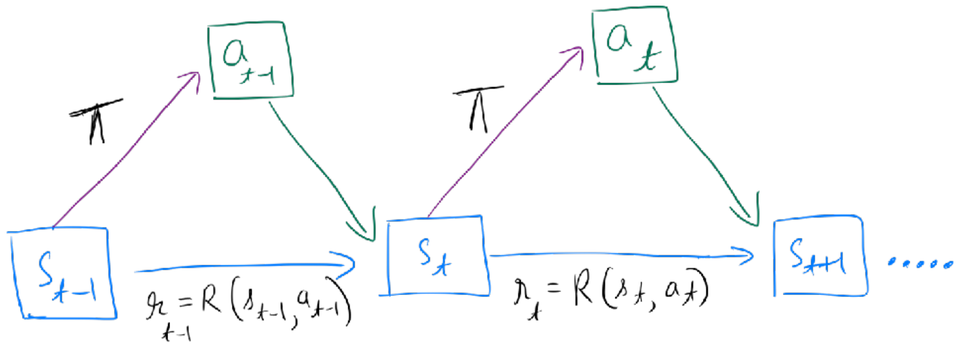
Value function

For any policy π, define the return to be the function $V : S \rightarrow$ Reals assigning to each state the quantity

$$V^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t r_t$$

Use expected values
in the stochastic case.

MDPs



Value function

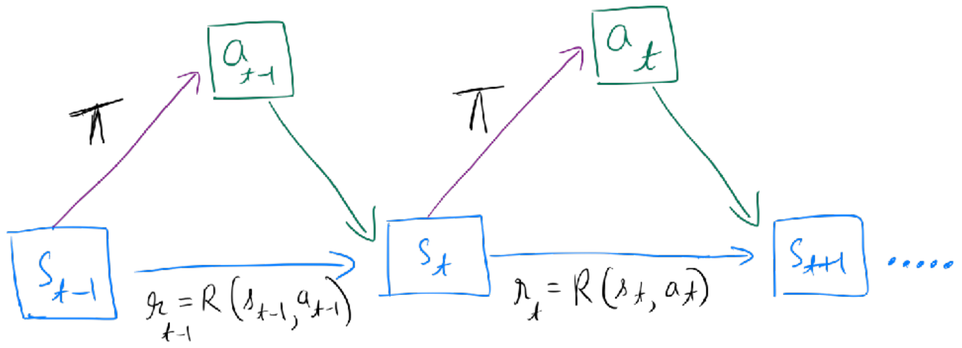
For any policy π, define the return to be the function $V : S \rightarrow$ Reals assigning to each state the quantity

$$V^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t r_t$$

Use expected values in the stochastic case.

- $s_0 = s$

MDPs



Value function

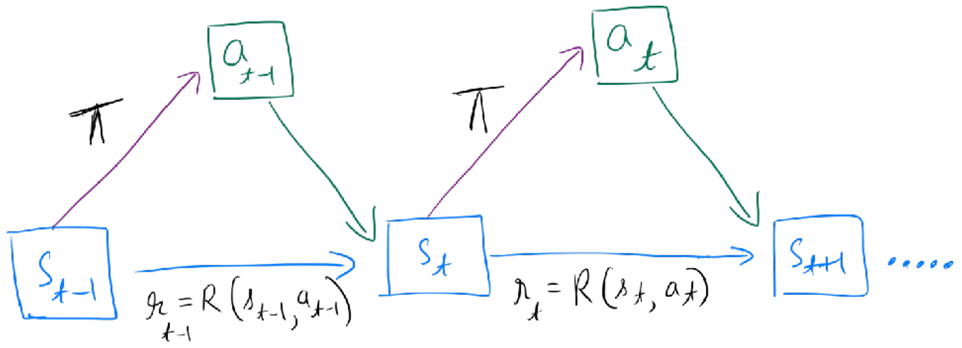
For any policy π, define the return to be the function V : S → Reals assigning to each state the quantity

$$V^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t r_t$$

Use expected values in the stochastic case.

- $s_0 = s$
- $a_t \sim \pi(a_t|s_t)$; $a_t = \pi(s_t)$

MDPs



Value function

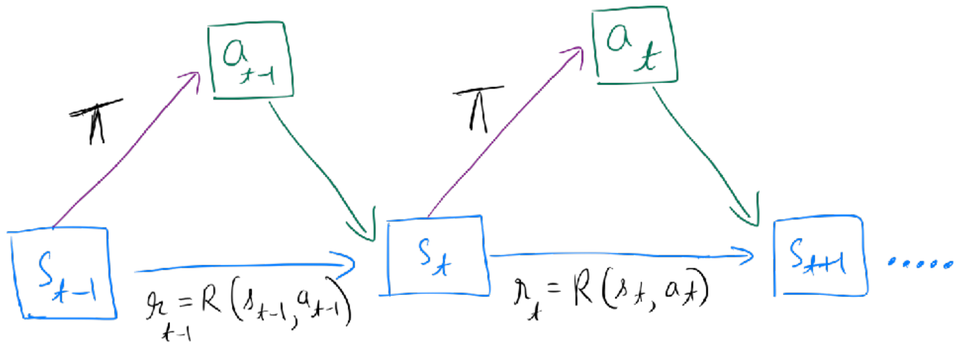
For any policy π, define the return to be the function $V : S \rightarrow$ Reals assigning to each state the quantity

$$V^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t r_t$$

Use expected values in the stochastic case.

- $s_0 = s$
- $a_t \sim \pi(a_t|s_t)$; $a_t = \pi(s_t)$
- $s_{t+1} \sim T(s_{t+1}|s_t, a_t)$; $s_{t+1} = T(s_t, a_t)$

MDPs



Value function

For any policy π, define the return to be the function $V : S \rightarrow$ Reals assigning to each state the quantity

$$V^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t r_t$$

Use expected values in the stochastic case.

- $s_0 = s$
- $a_t \sim \pi(a_t|s_t)$; $a_t = \pi(s_t)$
- $s_{t+1} \sim T(s_{t+1}|s_t, a_t)$; $s_{t+1} = T(s_t, a_t)$
- $r_t \sim R(r_t|s_t, a_t)$; $r_t = R(s_t, a_t)$

MDPs

Optimal Policy

MDPs

Optimal Policy

Goal: Learn to choose actions that maximize the cumulative reward

$$R = r_0 + \gamma r_1 + \gamma^2 r_2 + ...$$

MDPs

Optimal Policy

Goal: Learn to choose actions that maximize the cumulative reward

$$R = r_0 + \gamma r_1 + \gamma^2 r_2 + ...$$

Alternate phrasing w.r.t. policy: Find a policy π * such that
for any policy π and any state s

$$V^{\pi^*}(s) \geq V^{\pi}(s)$$

Such a policy is called an optimal policy.