ID :20231214 ID:20233002
ID:20231072 ID:241090158
ID:20231039 ID:20233616
ID:20233008 ID:20233052
ID:20232333 ID:20233398
ID:20231290
ID:241090046

AI Project: Probabilistic Reasoning in the Monty Hall Problem

Part 1: AI Project – Probabilistic Reasoning

1. Introduction & Problem Definition

The Monty Hall Problem is a classic probability puzzle named after the host of the television game show Let's Make a Deal. In the field of Artificial Intelligence, this problem serves as a fundamental benchmark for evaluating how agents update their beliefs when presented with new, non-random information.

The Scenario

An AI agent acting as a contestant is presented with three identical doors:

* Prize Distribution: One door conceals a car; the other two conceal "goats" (non-prizes).

* Initial Selection: The agent makes a preliminary choice (e.g., Door 1).

* Agent Interaction: The host, who has "insider information" regarding the car's location, opens one of the remaining doors to reveal a goat.

* The Decision Point: The agent must decide between two strategies: Stay with the original choice or Switch to the remaining closed door.

2. AI Implementation Approaches

To solve this in an AI context, we can model the agent using different algorithmic frameworks:

A. Bayesian Inference (Probabilistic Modeling)

AI agents use Bayes' Theorem to update the probability of a hypothesis as more evidence becomes available.

* Prior Probability: Initially, P(Car_at_Door_i) = 1/3 for all doors.

* The Evidence: The host's action is not random; he must choose a door that is neither the agent's pick nor the winning door.

* Posterior Update: After the goat is revealed, the probability that the car is behind the other unopened door increases to 2/3, while the original choice remains 1/3.

B. Monte Carlo Simulation

We can implement a Monte Carlo agent to simulate thousands of iterations of the game to empirically verify the optimal strategy.

* Simulation Logic: The AI runs N=10,000 trials for both "Stay" and "Switch" strategies.

* Result Analysis: The agent observes that the "Switch" strategy consistently converges to a 66.7% win rate, while "Stay" remains at 33.3%.

C. Reinforcement Learning (Q-Learning)

The problem can be modeled as a Markov Decision Process (MDP) where a Reinforcement Learning agent learns through rewards:

* State: Which doors are closed/open.

* Action: Stay or Switch.

* Reward: +1 for winning the car, 0 for a goat.

* Learning: Over time, the agent's Q-values will favor the "Switch" action as it yields a higher expected return.

3. Critical AI Insight: Intuition vs. Logic

The Monty Hall problem is a "pretty big deal" in AI because it highlights a common failure in human-like heuristic thinking. Many humans incorrectly assume a 50/50 split once one door is removed. An AI system built on strict probabilistic logic avoids this bias, demonstrating that "switching" is the mathematically superior decision for any rational agent.

Part 2: AI Work 2 – Rules and Dynamics

Section 1: Rules of the Monty Hall Problem

Rule 1: There are exactly three doors. No more, no less. One door hides a car (the prize), and two doors hide goats (no prize).

Rule 2: The contestant picks one door first. This choice is random, and the contestant does not know what is behind any door.

* Initial probability: The chosen door has a 1/3 chance of winning. The other two doors combined have a 2/3 chance.

Rule 3: The host knows where the car is. Monty is not confused, he is not guessing, and he has insider information.

Rule 4: The host always opens a door with a goat. He never opens the car door, never opens a door by accident, and makes no mistakes. This is critical because it gives real information.

Rule 5: The host never opens the contestant’s chosen door. Even if your door hides a goat, your door stays untouched until the end.

Rule 6: The host always offers the option to switch. He doesn't skip it, he doesn't base it on your personality, and he always asks: "Do you want to switch?".

Rule 7: Only one unopened door remains besides your own. After Monty opens a goat door, you now have:

* Your original door: Still 1/3.

* The other unopened door: Now 2/3.

Rule 8: The contestant makes a final choice.

* Stay: Keeps the original 1/3 chance.

* Switch: Takes the 2/3 chance.

* Switching doubles your odds.

Section 2: Importance of Host Behaviour

The host's behavior is crucial to the puzzle. The host never opens a door at random—he already knows everything. His deliberate choice to reveal a goat provides valuable information and changes the effective probabilities.

Key dynamics:

* Initial pick: 1/3 chance car is behind your door, 2/3 behind the other two combined.

* Host always reveals a goat from the remaining doors.

* Your original door's probability stays 1/3 (no new information about it).

* The remaining unopened door absorbs the full 2/3 probability.

If the host ever behaved differently (opened doors randomly, sometimes revealed the car, or didn't always offer a switch), the whole advantage of switching would disappear or change dramatically.

Conclusion: The predictable, knowledgeable, and consistent behavior of the host is what makes switching the winning strategy.

Section 3: Initial Probabilities (1/3 vs 2/3)

Initial Choice: At the beginning of the game, there are three doors. Only one hides the car. When the player chooses a door, the probability that this door has the car is 1/3 (≈ 33.3%).

What This Means:

* In 1 out of 3 cases, the player picked the car.

* In 2 out of 3 cases, the player picked a goat.

* On the first choice, the player is twice as likely to be wrong as right.

Probability of the Other Doors: The two doors not chosen by the player together have a 2/3 (≈ 66.7%) chance of hiding the car. This 2/3 probability belongs to the group of the two unchosen doors—not yet to any single one of them.

Important Point: These probabilities are fixed at the start—before the host opens any door. The host revealing a goat does NOT change the original 1/3 probability of your door. It only concentrates the remaining 2/3 probability onto the one other unopened door.

Part 3: AI Work 3 – Strategy Analysis: Switch or Stay?

The Setup

You are presented with three doors. Behind one door is a brand new car (the prize), and behind the other two doors are goats (no prize).

* The Choice: You pick a door. At this specific moment, you have a 1 in 3 (33.3%) chance of having the car.

* The Twist: The host (Monty), who knows what is behind the doors, opens one of the remaining doors to reveal a goat.

* The Question: Monty asks, "Do you want to keep Door 1, or switch to Door 2?".

The Staying Strategy

What Staying Means: You keep your original door after Monty opens a door showing a goat.

* Initial Odds: When you first choose a door, there is a 1 in 3 (33.3%) chance that you picked the car.

* Why the Odds Don't Change: Monty's action does not give any new information about your chosen door. He always opens a goat door and never opens your door.

* The Result: The probability that your original door has the car remains 1/3.

* The Verdict: Staying wins 1 out of 3 times (33%).

The Advantage of Switching

Why You Should Switch:

* The Math: When you made your first pick, there was a 66.6% (2/3) chance the car was behind one of the other two doors.

* The Reveal: When Monty opens a losing door, he doesn't change the initial probabilities. He simply gives you a clue.

* The Shift: The 1/3 chance stays with your original door. The massive 2/3 chance shifts entirely to the single remaining unpicked door.

The Verdict:

* Staying: Wins 1 out of 3 times (33%).

* Switching: Wins 2 out of 3 times (67%).

Strategy Comparison Table

| Strategy | What You Do | Chance of Winning |

|---|---|---|

| Stay | Keep your original door | 1 out of 3 (33%) |

| Switch | Change to the other unopened door | 2 out of 3 (67%) |

Final Verdict: Switching doubles your chance of winning compared to staying.

Part 4: AI Assignment – Probability and Implementation

1. Conditional Probability Framework

Definition and Formula

Conditional probability is the probability of an event occurring given that another event has already occurred or is known to occur.

* Mathematically, it is expressed as the formula:

* Where P(A|B) is the probability of A given B, and P(B) must be greater than 0.

Application to the Monty Hall Problem

The problem becomes conditional on Monty's action.

* Initial State: Your first pick has a 1/3 chance of being correct, while the other two doors together have a 2/3 chance.

* The Update: Monty always reveals a goat and never the car.

* The Outcome:

  * If you stay, your probability remains 1/3.

  * If you switch, you get the combined probability of both other doors (2/3), because Monty has effectively shown you which of those two was empty.

Why Switching Works:

* If your initial pick was wrong (2/3 probability), Monty must reveal the only other goat, so switching wins.

* If your initial pick was right (1/3 probability), switching loses.

2. Pseudocode Representation

Below is the algorithmic logic used to simulate the problem and prove the probabilities empirically.

```
FUNCTION simulate_monty_hall(num_trials, switch_strategy):
    wins = 0
    FOR trial in range(num_trials):
```

```
        # Setup: doors 0, 1, 2, car behind one random door
        car_door = random_integer(0, 2)
 # Player makes initial choice
        player_choice = random_integer(0, 2)
 # Monty reveals a goat from remaining doors
        possible_reveal_doors = [0, 1, 2] excluding player_choice and
car_door
        monty_reveals = random_choice(possible_reveal_doors)
   # Determine remaining door if switching
        remaining_doors = [0, 1, 2] excluding player_choice and
monty_reveals
        switch_door = remaining_doors[0] # only one remains
  # Determine final choice based on strategy
        IF switch_strategy = True:
            final_choice = switch_door
        ELSE:
            final_choice = player_choice
  # Check if win
        IF final_choice = car_door:
            wins = wins + 1
 win_percentage = (wins / num_trials) * 100
    RETURN win_percentage
```

END FUNCTION

Simulation Output:

* Running simulate_monty_hall(10000, False) yields a "Stay" win rate of approximately 33%.

* Running simulate_monty_hall(10000, True) yields a "Switch" win rate of approximately 66%.

3. Deep Dive: Relevance to Artificial Intelligence

The Monty Hall problem is not just a puzzle; it introduces foundational concepts used in AI.

* Bayesian Inference (Belief Updating): AI systems often work with incomplete information (e.g., self-driving cars). The problem demonstrates how an AI agent must update its "Belief State" when new evidence is introduced. An agent that fails to update its probabilities based on the host's action would be considered a "dumb" agent.

* Reinforcement Learning (RL): In RL, an agent learns by trial and error. If programmed to play millions of times, an agent would learn that the policy \pi(s)=Switch yields a higher "Expected Value" (reward) than Staying. This is similar to how AlphaGo determines actions that maximize future rewards.

* Causal Reasoning: Modern AI focuses on understanding why things happen. The problem relies on the causal constraint that the host cannot open the car door. If the host acted randomly, the math would change to 50/50. AI needs to understand these constraints to make correct decisions.

4. Project Conclusion

This project establishes that the Monty Hall problem is a demonstration of conditional probability and the importance of new information in decision making.

* Mathematical Certainty: The counter intuitive nature comes from the human tendency to view remaining doors as equal (50/50). However, the host's behavior is the critical variable; his "insider information" concentrates the rem consistent aining 2/3 probability onto the specific unchosen door.

* AI Benchmarking: The problem serves as a benchmark for how intelligent agents handle uncertainty.

* It illustrates Bayesian Inference by requiring belief updates.

* It demonstrates Reinforcement Learning as agents learn the higher expected value of switching.

* It shows how algorithmic reasoning can Overcome Bias, successfully identifying the winning strategy where human intuition often fails.

Ultimately, in a controlled environment where the host is knowledgeable and, the decision to switch is mathematically superior