CSCE 633: Machine Learning

Lecture 32: Reinforcement Learning

Texas A&M University

11-6-19

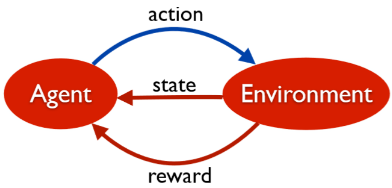
Goals of this lecture

- Reinforcement Learning
- Source: Mohri text and slides

Reinforcement Learning

- We have an Actions and Environment that do not passively collected labeled data
- The learner, called an Agent, gets two kinds of information to learn from: The current state of the environment, and a real-valued reward
- The objective of the learning problem is for the agent to maximize its reward
- It does this through finding the best course of actions - called a policy

Reinforcement Learning



- Exploration - search unknown states and actions to gain reward information
- Exploitation - search known states to optimize reward

Reinforcement Learning vs. Supervised Learning

- No fixed distribution that instances are drawn from
- Environment may not be fixed!
- Training and testing phases are mixed.
- Planning Problem: When the environment model is known - objective is to maximize reward
- Learning Problem: Environment model is unknown
- We will explore both

Applications

- Robot control e.g., Robocup Soccer Teams (Stone et al., 1999).
- Board games, e.g., TD-Gammon (Tesauro, 1995).
- Elevator scheduling (Crites and Barto, 1996).
- Ads placement.
- Telecommunications.
- Inventory management.
- Dynamic radio channel assignment.

Reinforcement Learning: Markov Decision Process

- Set of epochs $\{0, \cdots, T\}$
- a set of states S, possibly infinite!
- an initial state $s_0 \in S$
- Actions A, also possibly infinite
- Transition Probability $P(s'|s, a)$ which is the distribution over destination states $s' = \delta(s, a)$
- Reward Probability $P(r'|s, a)$ which is the distribution over rewards returned $r' = r(s, a)$

Policy

- MDP wants to determine what action to take at each state - that is - the policy
- $\pi : S \rightarrow \Delta(A)$ where $\Delta(A)$ is the set of probably distributions of actions A.
- A policy is deterministic if, for any s, there exists a unique $a \in A$ such that $\pi(s)(a) = 1$
- Stationary Policy - choice of distribution of actions does not depend upon time.

Policy: Non-stationary

- Finite horizon ($T < \text{inf}$): $\sum_{t=0}^{T} r(s_t, \pi(s_t))$
- Infinite horizon ($T = \text{inf}$): $\sum_{t=0}^{+\text{inf}} \gamma^t r(s_t, \pi(s_t))$ where $\gamma^t \in [0, 1)$ (discounts future rewards - earlier rewards are more important)

Policy: Value

- Value $V_\pi(s)$ is:
- Finite horizon: $V_\pi(s) = \mathbb{E}_{a_t \;\; \pi(s_t)} \left[\sum_{t=0}^{T} r(s_t, a_t) | s_0 = s \right]$
- Infinite horizon: $V_\pi(s) = \mathbb{E}_{a_t \;\; \pi(s_t)} \left[\sum_{t=0}^{+\inf} \gamma^t r(s_t, a_t) | s_0 = s \right]$
- These expectations are over the random selection of an action a_t according to a distribution $\pi(s_t)$
- We want to find a policy π that maximizes value over all states.
- A policy π^* is optimal if its value is maximal for every state s.
- For any MDP there exists a deterministic optimal policy.

State Action Value Function

- Q state-action value function associated to a policy π for all $(s, a) \in S \times A$
- $Q_\pi(s, a) = \mathbb{E}[r(s, a)] + \mathbb{E}_{a_t \ \pi(s_t)}[\sum_{t=0}^{+\inf} \gamma^t r(s_t, a_t) | s_0 = s, a_0 = a]$
- We find that $\mathbb{E}_{a \ \pi(s)}[Q_\pi(s, a)] = V_\pi(s)$
- This is by the Bellman Equations: $V_\pi(s) = \mathbb{E}[r(s, \pi(s))] + \gamma \sum_{s'} \mathbb{P}[s' | s, \pi(s)] V_\pi(s')$
- What does all this mean? That an optimal policy exists and can be determined. (Existence and Uniqueness)

Bellman Equation - Existence and Uniqueness

- Notation:
  - transition probability matrix $\mathbf{P}_{s,s'} = \Pr[s'|s, \pi(s)]$.
  - value column matrix $\mathbf{V} = V_\pi(s)$.
  - expected reward column matrix: $\mathbf{R} = \mathrm{E}[r(s, \pi(s)]$.
- Theorem: for a finite MDP, Bellman's equation admits a unique solution given by

$$\mathbf{V}_0 = (\mathbf{I} - \gamma \mathbf{P})^{-1} \mathbf{R}.$$

Bellman Equation - Existence and Uniqueness

- **Proof**: Bellman's equation rewritten as

$$\mathbf{V} = \mathbf{R} + \gamma \mathbf{P} \mathbf{V}.$$

  - $\mathbf{P}$ is a stochastic matrix, thus,

$$\|\mathbf{P}\|_\infty = \max_s \sum_{s'} |\mathbf{P}_{ss'}| = \max_s \sum_{s'} \Pr[s'|s, \pi(s)] = 1.$$

  - This implies that $\|\gamma \mathbf{P}\|_\infty = \gamma < 1$. The eigenvalues of $\gamma \mathbf{P}$ are all less than one and $(\mathbf{I} - \gamma \mathbf{P})$ is invertible.

- **Notes**: general shortest distance problem (MM, 2002).

Takeaways

- For a finite MDP - policy can be determined via matrix operations.
- Policy of a state is formed by policy of the prior states - forming a system of linear equations - this generates our matrices for the Bellman Equations
- But now let's solve our models by finding the solutions that these algorithms indicate exist.

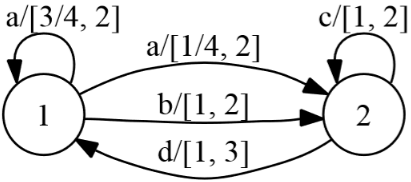
Planning Algorithms

- Recall for planning algorithms - the environment model is known and all we want to do is find the optimal policy.
- This means that $\mathbb{E}[r(s,a)]$ and $\mathbb{P}[s'|s,a]$ are known.
- Do not need to learn environment, just find the best course of action.
- Three algorithms will be covered - Value Iteration, Policy Iteration, and Linear Programming

Value Iteration

- Find optimal policy at each state, therefore, optimal across all states.
- Based upon Bellman equations.
- $V \leftarrow V_0$
- while $||V - \phi(V)|| \geq \frac{(1-\gamma)\epsilon}{\gamma}$ do
- $V \leftarrow \phi(V)$
- return $\phi(V)$
- Start with an arbitrary policy, then iteratively improve via some function.

VI Algorithm - Example



$$\mathbf{V}_{n+1}(1) = \max\left\{2 + \gamma\left(\frac{3}{4}\mathbf{V}_n(1) + \frac{1}{4}\mathbf{V}_n(2)\right), 2 + \gamma\mathbf{V}_n(2)\right\}$$

$$\mathbf{V}_{n+1}(2) = \max\left\{3 + \gamma\mathbf{V}_n(1), 2 + \gamma\mathbf{V}_n(2)\right\}.$$

For $\mathbf{V}_0(1) = -1, \mathbf{V}_0(2) = 1, \gamma = 1/2, \mathbf{V}_1(1) = \mathbf{V}_1(2) = 5/2.$

But, $\mathbf{V}^*(1) = 14/3, \mathbf{V}^*(2) = 16/3.$

Policy Iteration Algorithm

PolicyIteration(π_0)

$\pi \leftarrow \pi_0 \quad \triangleright\, \pi_0$ arbitrary policy

$\pi' \leftarrow$ NIL

while $(\pi \neq \pi')$ **do**

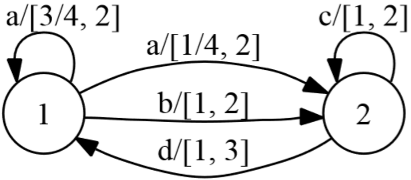
  $\mathbf{V} \leftarrow \mathbf{V}_\pi \quad \triangleright$ policy evaluation: solve $(\mathbf{I} - \gamma \mathbf{P}_\pi)\mathbf{V} = \mathbf{R}_\pi$.

  $\pi' \leftarrow \pi$

  $\pi \leftarrow \operatorname{argmax}_\pi \{\mathbf{R}_\pi + \gamma \mathbf{P}_\pi \mathbf{V}\} \quad \triangleright$ greedy policy improvement.

return π

PI Algorithm - Example



Initial policy: $\pi_0(1) = b, \pi_0(2) = c$.

Evaluation: $V_{\pi_0}(1) = 1 + \gamma V_{\pi_0}(2)$

$$V_{\pi_0}(2) = 2 + \gamma V_{\pi_0}(2).$$

Thus, $V_{\pi_0}(1) = \dfrac{1+\gamma}{1-\gamma} \quad V_{\pi_0}(2) = \dfrac{2}{1-\gamma}.$

Notes

- Policy Iteration algorithm converges faster than Value Iteration
- Each iteration of the Policy Iteration requires solving a system of linear equations which might be more expensive computationally

Primal Linear Program

- **LP formulation**: choose $\alpha(s) > 0$, with $\sum_s \alpha(s) = 1$.

$$\min_{\mathbf{V}} \sum_{s \in S} \alpha(s) V(s)$$

$$\text{subject to } \forall s \in S, \forall a \in A, V(s) \geq \mathrm{E}[r(s,a)] + \gamma \sum_{s' \in S} \Pr[s'|s,a] V(s').$$

- **Parameters**:
  - number rows: $|S||A|$.
  - number of columns: $|S|$.

Dual Linear Program

- **LP formulation:**

$$\max_{\mathbf{x}} \sum_{s\in S, a\in A} \mathrm{E}[r(s,a)]\, x(s,a)$$

$$\text{subject to} \quad \forall s \in S, \sum_{a\in A} x(s',a) = \alpha(s') + \gamma \sum_{s\in S, a\in A} \Pr[s'|s,a]\, x(s',a)$$

$$\forall s \in S, \forall a \in A, x(s,a) \geq 0.$$

- **Parameters**: more favorable number of rows.
  - number rows: $|S|$.
  - number of columns: $|S||A|$.

Takeaways and Next Time

- Next Time: More General Learning