BlackJack Game Using

Vikas Rajpoot
S2021101006

Course Instructor :
Dr. Arun P V

Environment

- BlackJack is the popular casino Game in which player is playing against the Dealer.

- BlackJack :
  Ace + (Face card or 10).

Rules

- In this game ‘Number card’ weightage is equal to the their value, Face is value is 10 and Ace can be either 11 or 1.
- Player win if his score is greater than the dealer score and less then the 21. score is equal to the sum of cards in the hand.
- Player’s Actions : Hit or Stick
  Hit : Take random card from deck.
  Stick : Further player not receive any card and game is over for the player.
- Dealer keeps Hit until his score is greater than 17. If score is greater than 17 He stick and both player and dealer show their card.

Environment Implementation

```
class BlackJack:
        def __init__(self) -> None:
        def start(self):

        def play(self, action):
        def __delta_card(self):
        def __bust(self, cards, hands):
        def __score(self, cards, hands):
        def __usable_ace(self, cards, hands):
```

Environment Returns

- play function

  return {'pCards', 'nState', 'dealers_cards', 'dealers_score', 'action', 'reward', 'done'}

nState : Next state of the agent.
reward : (-1, 0, 1).
done : False / True, if game over for player then True otherwise False.
pCard : cards player have in hand.
dealers_cards : cards dealer have.
dealers_score : score of the dealer i.e. sum of cards.

Agent

- Agent in this game is the player which try to maximize the average reward over the episodes.
- I use Monte Carlo method to implement the Agent.
- Agent can take two action Hit or Stick. Accordingly It gets the reward, next state and other parameter from the Environment.

Implementation of the Agent

- For Agent implementation there are mainly two part : Policy evaluation and Policy improvement.
- Policy evaluation : I use Monte Carlo Q value evaluation.
- Policy improvement : Use ε-Greddy policy improvement.

Monte Carlo Model Free Policy Evaluation

- State-Action (S□, A□) Value evaluation for each episode.

$$N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$$

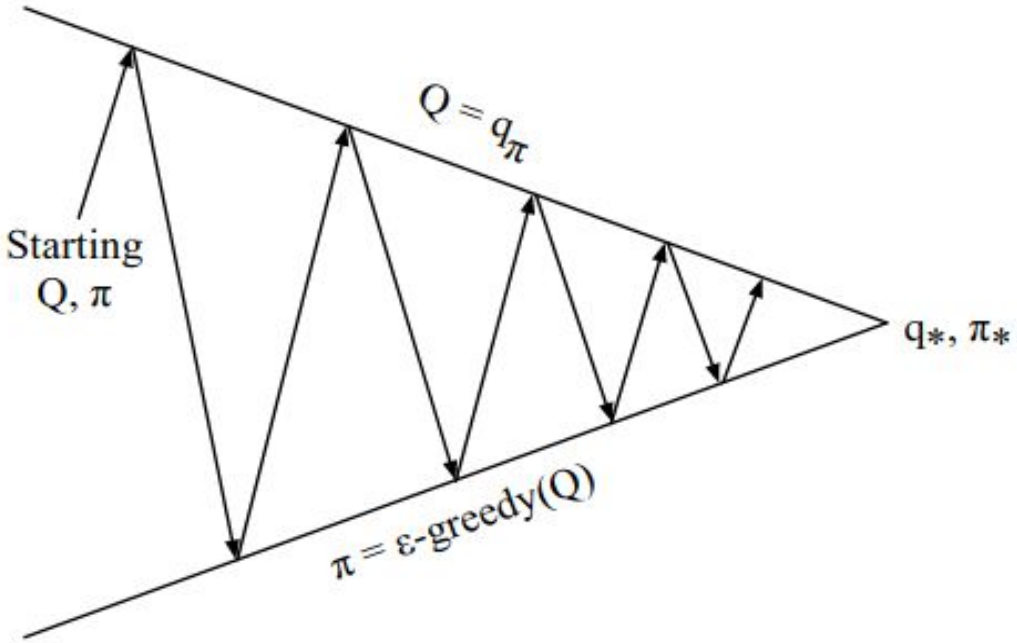
$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t)} (G_t - Q(S_t, A_t))$$

ε-Greedy policy Improvement

- ε-Greedy policy allow some exploration.

$$\pi(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon & \text{if } a^* = \underset{a \in \mathcal{A}}{\text{argmax}}\ Q(s,a) \\ \epsilon/m & \text{otherwise} \end{cases}$$

Diagrams



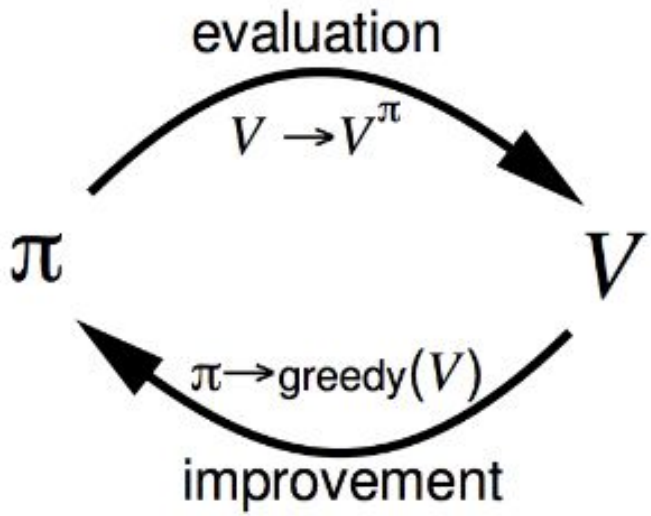


fig. Q values convergence

Programs Q value convergence

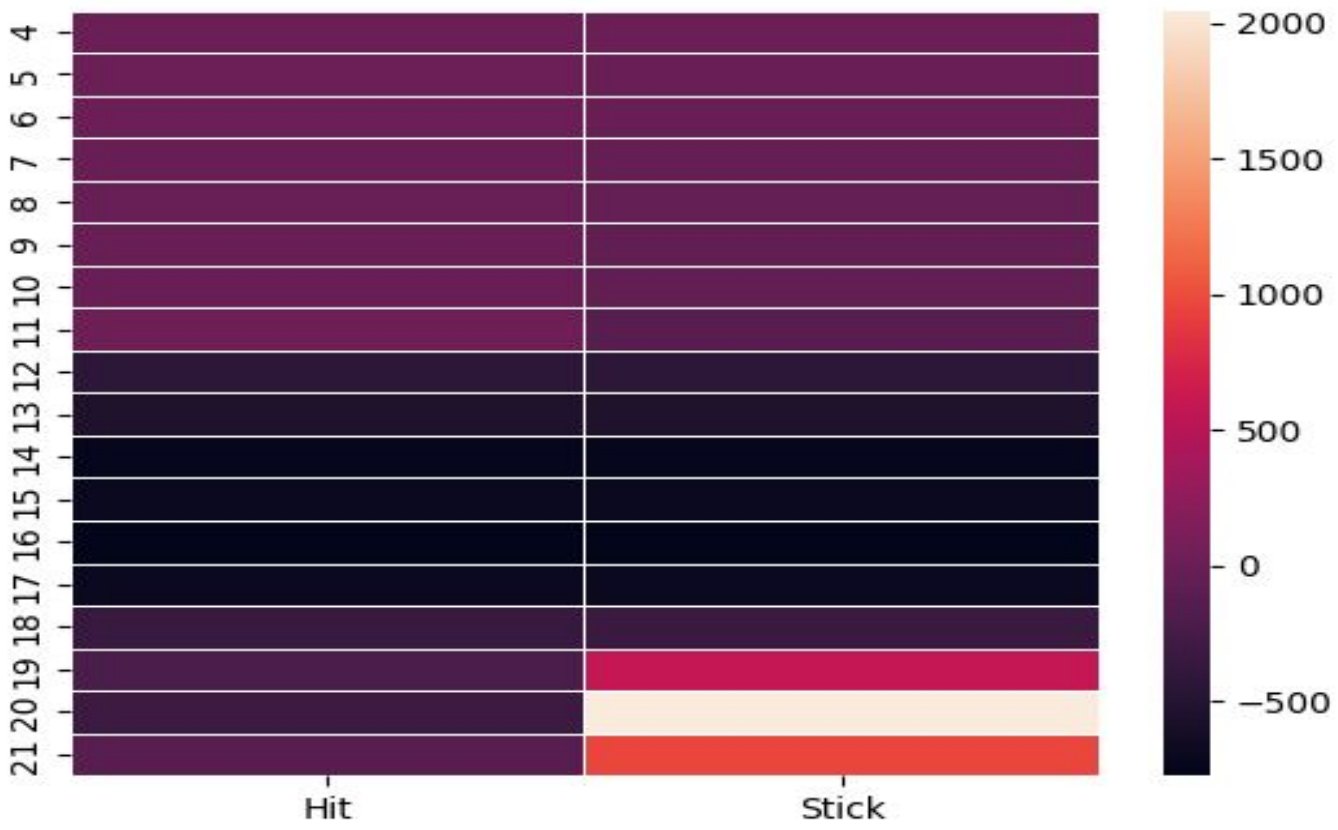


fig. Program Q values convergence

Thank You