Московский государственный технический университет им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

Лабораторная работа №4

по дисциплине

«Методы машинного обучения»

на тему

Выполнила:

студентка Ли Лююй

группы ИУ5И-21М

Москва — 2024 г.

1. Цель лабораторной работы

Цель лабораторной работы: ознакомление с базовыми методами обучения с подкреплением.

2. Задание

На основе рассмотренного на лекции примера реализуйте алгоритм Policy Iteration для любой среды обучения с подкреплением (кроме рассмотренной на лекции среды Toy Text / Frozen Lake) из библиотеки Gym (или аналогичной библиотеки).

3. Ход выполнения работы

3.1. Текстовое описание набора данных

- Шаг 1: Установите и импортируйте необходимые библиотеки
- Шаг 2: Определите алгоритм итерации стратегии
- Шаг 3: Обучите и оцените агента итерации стратегии
- Шаг 4: Запустите обученную стратегию

3.2. Основные характеристики набора данных

第一步：安装并导入必要的库

Шаг 1: Установите и импортируйте необходимые библиотеки

```
# 安装必要的库
!pip install gym numpy matplotlib

# 导入必要的库
import gym
import numpy as np
import matplotlib.pyplot as plt
from pprint import pprint
```

```
Requirement already satisfied: gym in /usr/local/lib/python3.10/dist-packages (0.25.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (1.25.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: cloudpickle>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from gym) (2.2.1)
Requirement already satisfied: gym-notices>=0.0.4 in /usr/local/lib/python3.10/dist-packages (from gym) (0.0.8)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.53.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (24.0)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

Шаг 2: Определите алгоритм итерации стратегии

```
class PolicyIterationAgent:
    def __init__(self, env, gamma=0.99, theta=1e-6, max_iterations=1000):
        self.env = env
        self.gamma = gamma
        self.theta = theta
        self.max_iterations = max_iterations

        self.policy = np.ones([env.observation_space.n, env.action_space.n]) / env.action_space.n
        self.value_function = np.zeros(env.observation_space.n)

    def policy_evaluation(self):
        for _ in range(self.max_iterations):
            delta = 0
            for state in range(self.env.observation_space.n):
                v = 0
                for action, action_prob in enumerate(self.policy[state]):
                    for prob, next_state, reward, done in self.env.P[state][action]:
                        v += action_prob * prob * (reward + self.gamma * self.value_function[next_state])
                delta = max(delta, np.abs(v - self.value_function[state]))
                self.value_function[state] = v
            if delta < self.theta:
                break

    def policy_improvement(self):
        policy_stable = True
        for state in range(self.env.observation_space.n):
            chosen_action = np.argmax(self.policy[state])
            action_values = np.zeros(self.env.action_space.n)
            for action in range(self.env.action_space.n):
                for prob, next_state, reward, done in self.env.P[state][action]:
                    action_values[action] += prob * (reward + self.gamma * self.value_function[next_state])
            best_action = np.argmax(action_values)
            if chosen_action != best_action:
                policy_stable = False
            self.policy[state] = np.eye(self.env.action_space.n)[best_action]
        return policy_stable

    def policy_iteration(self):
        for i in range(self.max_iterations):
            self.policy_evaluation()
            if self.policy_improvement():
                print(f"Policy iteration converged at iteration {i+1}")
                break

    def visualize_policy(self):
        print("Policy (state -> action):")
        for state in range(self.env.observation_space.n):
            print(f"{state} -> {np.argmax(self.policy[state])}")

    def visualize_value_function(self):
        print("Value Function:")
        print(self.value_function)
```

第三步：训练并评估策略迭代代理

Шаг 3: Обучите и оцените агента итерации стратегии

```
env = gym.make("Taxi-v3")
agent = PolicyIterationAgent(env)

agent.policy_iteration()
agent.visualize_policy()
agent.visualize_value_function()
```

```
Policy iteration converged at iteration 13
Policy (state -> action):
0 -> 4
1 -> 4
2 -> 4
3 -> 4
4 -> 0
5 -> 0
```

```
499 -> 3
Value Function:
[944.72361809 864.01317574 903.55733909 873.75068256 789.53804327
 864.01317574 789.53804327 816.7669381  864.01317574 826.0272102
 903.55733909 835.3810204  807.59926872 826.0272102  807.59926872
 873.75068256 955.27638191 873.75068256 913.69428191 883.58654804
 934.27638191 854.37304398 893.5217657  864.01317574 798.52327603
 873.75068256 798.52327603 826.0272102  854.37304398 816.7669381
 893.5217657  826.0272102  816.7669381  835.3810204  816.7669381
 883.58654804 944.72361809 883.58654804 903.55733909 893.5217657
 883.58654804 807.59926872 844.82931354 816.7669381  844.82931354
 923.93361809 844.82931354 873.75068256 844.82931354 807.59926872
 883.58654804 816.7669381  826.0272102  844.82931354 826.0272102
 893.5217657  893.5217657  934.27638191 893.5217657  903.55733909
 873.75068256 798.52327603 835.3810204  807.59926872 854.37304398
 934.27638191 854.37304398 883.58654804 835.3810204  798.52327603
 873.75068256 807.59926872 835.3810204  854.37304398 835.3810204
 903.55733909 883.58654804 944.72361809 883.58654804 913.69428191
 864.01317574 789.53804327 826.0272102  798.52327603 864.01317574
 944.72361809 864.01317574 893.5217657  826.0272102  789.53804327
 864.01317574 798.52327603 826.0272102  844.82931354 826.0272102
 893.5217657  873.75068256 955.27638191 873.75068256 903.55733909
 934.27638191 854.37304398 893.5217657  864.01317574 798.52327603
 873.75068256 798.52327603 826.0272102  873.75068256 835.3810204
 913.69428191 844.82931354 816.7669381  835.3810204  816.7669381
 883.58654804 944.72361809 883.58654804 923.93361809 893.5217657
 923.93361809 844.82931354 883.58654804 854.37304398 807.59926872
 883.58654804 807.59926872 835.3810204  864.01317574 826.0272102
 903.55733909 835.3810204  826.0272102  844.82931354 826.0272102
 893.5217657  934.27638191 893.5217657  913.69428191 903.55733909
 893.5217657  816.7669381  854.37304398 826.0272102  835.3810204
 913.69428191 835.3810204  864.01317574 854.37304398 816.7669381
 893.5217657  826.0272102  835.3810204  854.37304398 835.3810204
 903.55733909 903.55733909 923.93361809 903.55733909 913.69428191
 883.58654804 807.59926872 844.82931354 816.7669381  844.82931354
 923.93361809 844.82931354 873.75068256 844.82931354 807.59926872
 883.58654804 816.7669381  844.82931354 864.01317574 844.82931354
 913.69428191 893.5217657  934.27638191 893.5217657  923.93361809
 873.75068256 798.52327603 835.3810204  807.59926872 854.37304398
 934.27638191 854.37304398 883.58654804 835.3810204  798.52327603
 873.75068256 807.59926872 835.3810204  854.37304398 835.3810204
 903.55733909 883.58654804 944.72361809 883.58654804 913.69428191
 923.93361809 844.82931354 883.58654804 854.37304398 807.59926872
 883.58654804 807.59926872 835.3810204  883.58654804 844.82931354
 923.93361809 854.37304398 826.0272102  844.82931354 826.0272102
 893.5217657  934.27638191 893.5217657  934.27638191 903.55733909
 913.69428191 835.3810204  873.75068256 844.82931354 816.7669381
 893.5217657  816.7669381  844.82931354 873.75068256 835.3810204
 913.69428191 844.82931354 835.3810204  854.37304398 835.3810204
 903.55733909 923.93361809 903.55733909 923.93361809 913.69428191
 903.55733909 826.0272102  864.01317574 835.3810204  826.0272102
 903.55733909 826.0272102  854.37304398 864.01317574 826.0272102
 903.55733909 835.3810204  844.82931354 864.01317574 844.82931354
 913.69428191 913.69428191 913.69428191 913.69428191 923.93361809
 893.5217657  816.7669381  854.37304398 826.0272102  835.3810204
 913.69428191 835.3810204  864.01317574 854.37304398 816.7669381
 893.5217657  826.0272102  854.37304398 873.75068256 854.37304398
```

第四步：运行训练好的策略

Шаг 4: Запустите обученную стратегию

```python
def run_agent(env, agent, episodes=5):
    for episode in range(episodes):
        state = env.reset()
        done = False
        total_reward = 0
        while not done:
            action = np.argmax(agent.policy[state])
            state, reward, done, info = env.step(action)
            total_reward += reward
            env.render()
        print(f"Episode {episode + 1}: Total Reward: {total_reward}")

env = gym.make("Taxi-v3")
agent = PolicyIterationAgent(env)
agent.policy_iteration()

run_agent(env, agent)
env.close()
```

```
Policy iteration converged at iteration 13
/usr/local/lib/python3.10/dist-packages/gym/utils/passive_env_checker.py:241: DeprecationWarning: `np.bool8` is a deprecated alias for `np.bool_`.  (Deprecated NumPy 1
  if not isinstance(terminated, (bool, np.bool8)):
/usr/local/lib/python3.10/dist-packages/gym/core.py:49: DeprecationWarning: WARN: You are calling render method, but you didn't specified the argument render_mode at e
If you want to render in human mode, initialize the environment in this way: gym.make('EnvName', render_mode='human') and don't call the render method.
See here for more information: https://www.gymlibrary.ml/content/api/
  deprecation(
Episode 1: Total Reward: 7
Episode 2: Total Reward: 10
Episode 3: Total Reward: 9
Episode 4: Total Reward: 11
Episode 5: Total Reward: 6
```