

گزارش پروژه اول قسمت اول

پروژه Cliff Walking

درس: مبانی و کاربردهای هوش مصنوعی

استاد راهنما: دکتر حسین کارشناس نجف آبادی

اعضای گروه:

علی‌اکبر احراری- 4003613001

مهرآذین مزروق- 4003613055

پاییز 1402

فهرست

[گزارش کار الگوریتم 3](#_Toc153233330)

[**Policy\_evaluation** 3](#_Toc153233331)

[**Policy\_iteration** 5](#_Toc153233332)

[نمونه خروجی 6](#_Toc153233333)

[منابع 6](#_Toc153233334)

# گزارش کار الگوریتم

**Policy\_evaluation**

def policy\_evaluation(policy):  
 Vp = np.zeros(env.nS)  
 Qp = np.zeros((env.nS, env.nA)) / env.nA  
 converged = False  
 t = 1  
  
 while t < 1000 and not converged:  
 delta = 0  
 old\_Vp = Vp.copy()  
  
 for state in range(env.nS):  
 if cliffs.\_\_contains\_\_(state):  
 Vp[state] = -100  
 elif state == 47:  
 Vp[state] = 2000  
 else:  
 for action in range(env.nA):  
 ans = 0  
 for probability, next\_state, reward, done in env.P[state][action]:  
 ans += (1/2)\*(reward+gamma\*old\_Vp[next\_state])  
 for probability, next\_state, reward, done in env.P[state][(action - 1)%4]:  
 ans += (1/4)\*(reward+gamma\*old\_Vp[next\_state])  
 for probability, next\_state, reward, done in env.P[state][(action+1)%4]:  
 ans += (1/4)\*(reward+gamma\*old\_Vp[next\_state])  
 Qp[state][action] = ans  
  
 Vp[state] = Qp[state][int(policy[state])]  
  
 # Calculate the change in utility value  
 delta = np.max(np.abs(old\_Vp - Vp[state]))  
  
 if delta < theta:  
 converged = True  
 t+=1  
  
 return Vp, Qp

این تابع برای تخمین ارزش valueها و q\_valueها با استفاده از یک سیاست به عنوان ورودی به کار می رود. مراحل اجرای این تابع بدین صورت است که ابتدا ارزش وضعیت های مختلف را با 0 مقداردهی اولیه می کنیم. دو متغیر برای تشخیص شرایط اتمام حلقه(محدودیت تکرار و همگرایی) تعریف می کنیم. برای مقایسه کردن Vp فعلی با Vpهای جدید، از متغیر old\_Vp استفاده می کنیم.

برای هر حالت در فضای بازی، حالات مختلفی که عامل می تواند در آن قرار بگیرد را بررسی می کنیم و بنا به آن Vp مناسب را انتخاب می کنیم. برای cliffها ارزش منفی صد و برای هدف اصلی، 2000 را در نظر می گیریم. در صورتی که عامل در حالتی دیگر باشد به صورت دیگری عمل می کنیم.

این بخش از کد، برای به‌روزرسانی تخمین‌های توابع ارزش وضعیت‌ها و عمل‌ها با استفاده از معادله بلمن استفاده می‌شود. این احتمالات وضعیت‌های بعدی، پاداش‌ها و اطلاعات ترانزیشن را از محیط دریافت می‌کند و بر اساس آن‌ها مقادیر توقعی تابع ارزش عمل را محاسبه می‌کند. سپس این مقادیر با وزن‌های مختلف به‌روزرسانی شده و تابع ارزش وضعیت نیز بر اساس سیاست جاری به‌روزرسانی می‌شود.

در پایان هر دور ایتریشن، میزان تغییرات در تخمین‌های تابع ارزش بررسی شده و اگر این تغییرات کمتر از یک حد مشخص (مانند theta) باشد، الگوریتم به‌صورت همگرا معلوم می‌شود و محاسبات متوقف می‌شود.

**Policy\_iteration**

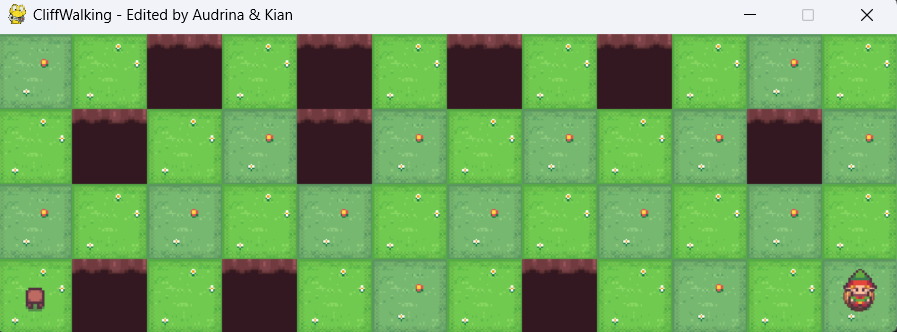
def policy\_iteration():  
  
 policy = np.zeros(env.nS)  
 t = 1  
 converged = False  
 while t < 100 and not converged:  
 old\_policy = policy.copy()  
 Vp, Qp = policy\_evaluation(policy)  
 for state in range(env.nS):  
 act = -1  
 maxQ = -np.inf  
 for action in range(env.nA):  
 if Qp[state][action] > maxQ:  
 if not (state >= 0) & (state < 12) & (action == 0):  
 if not (state > 35) & (state < 48) & (action == 2):  
 if not (state % 12 == 11) & (action == 1):  
 if not (state % 12 == 0) & (action == 3):  
 maxQ = Qp[state][action]  
 act = action  
  
 policy[state] = act  
  
 delta = np.max(np.abs(policy - old\_policy))  
 if delta < theta:  
 converged = True  
 t = t+1  
  
 return policy

در این تابع به تکرار و بهینه سازی سیاست(policy) می پردازیم. ایتدا یک مقداردهی اولیه (مقدار 0) را در نظر می گیریم. برای تشخیص شزط اتمام حلقه، دو متغییر t و converged را مقدار دهی اولیه می کینم. در صورتی که تکرار از مقداز تعیین شده بگذرد و یا سیاست مورد نظر در نقطه ای همگرا شود، حلقه پایان می یابد. با استفاده از متغییر old\_policy می توان همگرایی سیاست را بررسی کرد. با استفاده از تابع policy\_evaluation مقادیر q\_value و value ها را بدست می آوریم.

حال به ازای هر state در محیط بازی، بین تمامی action های ممکن به انتخاب actionی می پردازیم که مقدارQp را به حداکثر ممکن برساند. در این بین، شروطی برای حرکت عامل وجود دارند که از برخورد آن به دیواره ها جلوگیری می کنند. پس از آن action مورد نظر انتخاب شده و سیاست به روز می شود.

پس از هر بار به روز رسانی، بیشترین مقدار تغییرات در سیاست با استفاده از delta مورد بررسی قرار می گیرد و در صورتی که از مقدار theta کمتر باشد، شرط همگرایی برقرار شده است. در آخر، این تابع سیاست بهینه را بر می گرداند.

# نمونه خروجی



# 

# منابع

[Bing AI - Search](https://www.bing.com/search?form=NTPCHB&q=Bing+AI&showconv=1)

[Bard (google.com)](https://bard.google.com/chat)

[ChatGPT (openai.com)](https://chat.openai.com/)

Artificial Intelligence: A Modern Approach, Textbook by Peter Norvig and Stuart J. Russell