Primena učenja sa potkrepljivanjem (Reinforcment Learning) na rešavanje kartaških igara - primer BlackJack igre primenom Monte Carlo algoritma

Nikola Matijević - IN02-2018 - Marko Todorčević - IN03-2018

1. Učenje sa potkrepljivanjem (Reinforcement Learning - RL)

Predstavlja paradigmu mašinskog učenja koja trenira polisu agenta, tako da može da donese niz odluka. Cilj agenta je da proizvede akcije u skladu sa zapažanjem koje vrši u svom okruženju. Ove radnje onda dovode do više zapažanja i nagrada. Obuka uključuje brojne pokušaje i greške dok agent stupa u interakciju sa okruženjem i u svakoj iteraciji je u stanju da poboljša polisu.

Ovaj pristup učenju postao je veoma popularan poslednjih godina, pošto ovi agenti zaista dobro obavljaju složene zadatke, izazivajući istinske pomake u oblasti mašinskog učenja. Postoji mnogo slučajeva upotrebe u svetu koji koristi ovu tehnologiju, u oblastima kao što su robotika, hemija i još mnogo toga.

U kontekstu igara, agent koji preduzima radnje ili vrši ponašanje je agent igre. Razmislite sada o liku, igraču ili botu u igri, on mora da razume stanje igre, gde su igrači, a zatim na osnovu ovog zapažanja treba da donese odluku na osnovu situacije u igri. U RL, odluke su vođene nagradama, koje bi u igri mogle biti obezbeđene kao visok rezultat, ili novi nivo za postizanje određenog cilja. Na primer, moglo bi da se nauči šta da radi kada je napadnut ili kako da se ponaša da bi postigao određeni cilj.

Od ovih dostignuća, međutim, učenje sa potkrepljivanjem je još uvek nezrelo i nestabilno u primenama sa više agenata, velikim prostorom stanja ili malom nagradom.

2. Monte Carlo Metoda

Monte Carlo metode, zapravo, čini grupa računarskih algoritama koji se oslanjaju na ponavljanje slučajnih pokušaja da bi se dobili numerički rezultati. Često se koriste u rešavanju fizičkih i matematičkih problema i veoma su korisni u slučajevima kada je nemoguće koristiti druge matematičke metode.

Ove metode se najčešće koriste u tri klase slučaja: optimizaciji, numeričkoj integraciji i generisanju uzoraka kod raspodele verovatnoće.

Monte Carlo metode variraju, ali teže da isprate određeni obrazac:

1. Definiše se domen ulaza,
2. Generiše se niz slučajnih ulaza pomoću raspodele verovatnoće,
3. Izvrše se deterministički proračuni nad ulazima i
4. Izvrši se sažimanje rezultata.

Monte Carlo metod, koji se oslanja na stohastički način generisanja numeričkih vrednosti za determinističke probleme koji su teški ili čak nerešivi primenjivanjem drugih metoda i pristupa rešavanju istih, pojavljuje se čak 1940. godine.

Na primer, uzimimo u obzir krug upisan u kvadrat. S obzirom na to da je odnos ovih površina π/4, vrednost π se može aproksimirati upotrebom Monte Carlo metode:

1. Nacrtati kvadrat, zatim upisati četvrtinu kruga u isti
2. Ravnomerno prosuti neke objekte (zrna peska ili pirinča, na primer) jednake veličine preko kvadrata
3. Prebrojati ukupan broj zrna unutar kvadranta kruga
4. Odnos između prebrojanog broja zrna i ukupnog broja je procena odnosa dve površine, π/4. Rezultat se onda pomnoži sa 4, čime se dobije broj π.

U ovoj proceduri domen ulaza je kvadrat u koji je upisan kvadrant kruga. Generišemo broj ulaza prosipanjem preko kvadrata, zatim izvršimo proračune na svaki ulaz (pitamo se da li je zrno palo u krug). Napokon, sažimamo rezultate da bi dobili konačan rezultat, aproksimaciju π. Ovde imamo dve bitne napomene: prvo, ako zrna nisu ravnomerno raspodeljena, onda je naša aproksimacija nepotpuna. Drugo, mora da postoji veliki broj ulaza. Aproksimacija je generalno loša samo ako je par zrna palo u ceo kvadrat. U proseku kvalitet procene se povećava sa većim brojem zrna u kvadratu. Upotreba Monte Carlo metode zahteva veliku količinu brojeva, što je prouzrokovalo formiranje generatora pseudobrojeva čijom upotrebom je proračunavanje olakšano.

U poređenju sa dinamičkim programiranjem, Monte Carlo može da uči direktno u interakciji sa okolinom, a pritom mu nije neophodan potpun model, nije mu potrebno da poznaje sva stanja igre. Jedini problem na koji treba obratiti pažnju je održavanje odgovarajućeg odnosa između exploration-a (istaživanja okoline i stanja) i exploitation-a (iskorišćavanja poznatih i profitabilnih stanja)

3. Exploration-exploitation

Prilikom obuke agenta da uči u nasumičnom okruženju, izazovi istraživanje i eksploatacije odmah se javljaju. Agent prima nagrade dok komunicira se okruženjem u okviru povratih informacija. Da bi maksimizirao svoje nagrade, tipično je za agenta da ponovi radnje koje je pokušao u prošlosti i koje su proizvele “povoljne” nagrade. Međutim, da bi pronašao ove radnje koje vode do nagrada, agent će morati da uzme uzorak iz skupa radnji i isproba različite radnje koje nisu prethodno odabrane. Pri tome, agent takođe mora da isporba prethodno neizabrane radnje, u suprotnom, neće uspeti da otkrije bolje akcije.

Istraživanje je kada agent mora da uzorkuje radnje iz skupa akcija da bi dobio bolje nagrade. Eksploatacija je, s druge strane, kada agent koristi ono što već zna u ponavljanju radnji koje vode do “povoljnih” dugoročnih nagrada.

U stohastičkom okruženju, akcije će morati da budu dovoljno dobro uzorkovane da bi se dobila očekivana procena nagrade. Agent koji se bavi isključivo istraživanjem ili isključivo eksploatacijom postaje gori od agenta zasnovanog na čistoj slučajnosti.

4. BlackJack

Cilj BlackJack igre jeste pobediti dilera i postoje tri načina da to učinimo:

1. Dilera pobeđujemo ako je vrednost naše ruke veća od njegove, a pritom da suma vrednosti karte u našoj ruci ne premaši 21.
2. Dilera pobeđujemo ako dilerova suma vrednosti ruke nakon povlačenja karte premaši 21 (“bust“).

Izgubili smo:

1. Kada suma karata u našoj ruci premaši 21 (“bust”).
2. Kada na kraju runde dilerov suma premaši našu.

BlackJack pravila:

Igrač u BlackJack igri može povlači odluke koje želi, dok sa druge strane diler je ograničen da igra po unapred postavljenim pravilima.

BlackJack pravila za igrača:

- Nakon deljenja prve 2 karte, kada je igrač na potezu može birati da li želi da stane (stand) ili izvuče još jednu kartu - udari (hit)
- Nakon što je igrač odigrao jednu od opcija, diler će reagovati i uporediti njegovu ruku sa igračovom da bi odredio ko je pobednik.

BlackJack za dilera:

- Diler će se zaustaviti ako je suma njegovih karata 17 ili više.
- Diler će nastaviti sa izvlačnjem dok god suma njegovih karata ne bude bar 17.

Slikama dodeljujemo vrednost 10, As-u dodeljujemo 1 ili 11 u zavisnosti od odluke igrača, ostalim kartama dodeljujemo vrednost napisanu na njima.

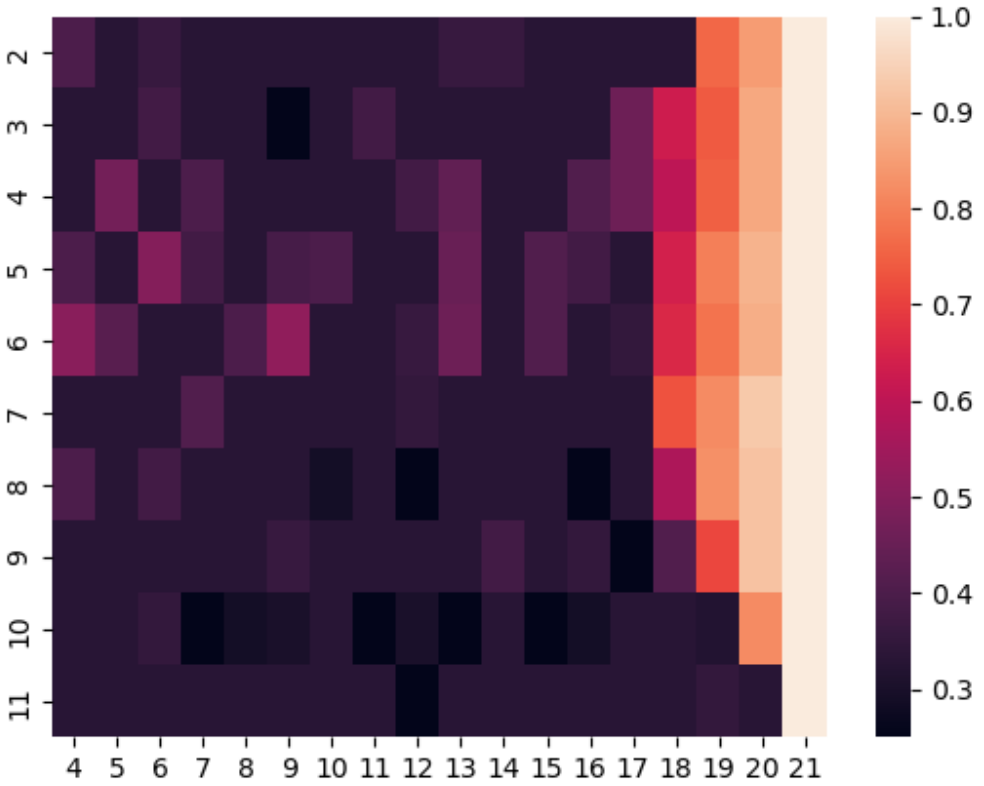
5. Monte Carlo primenjen na kartašku igru BlackJack

Nakon implementacije igre BlackJack u python-u, potrebno je odrediti polisu igranja, način za postizanje adekvatnog odnosa exploration-exploitation.

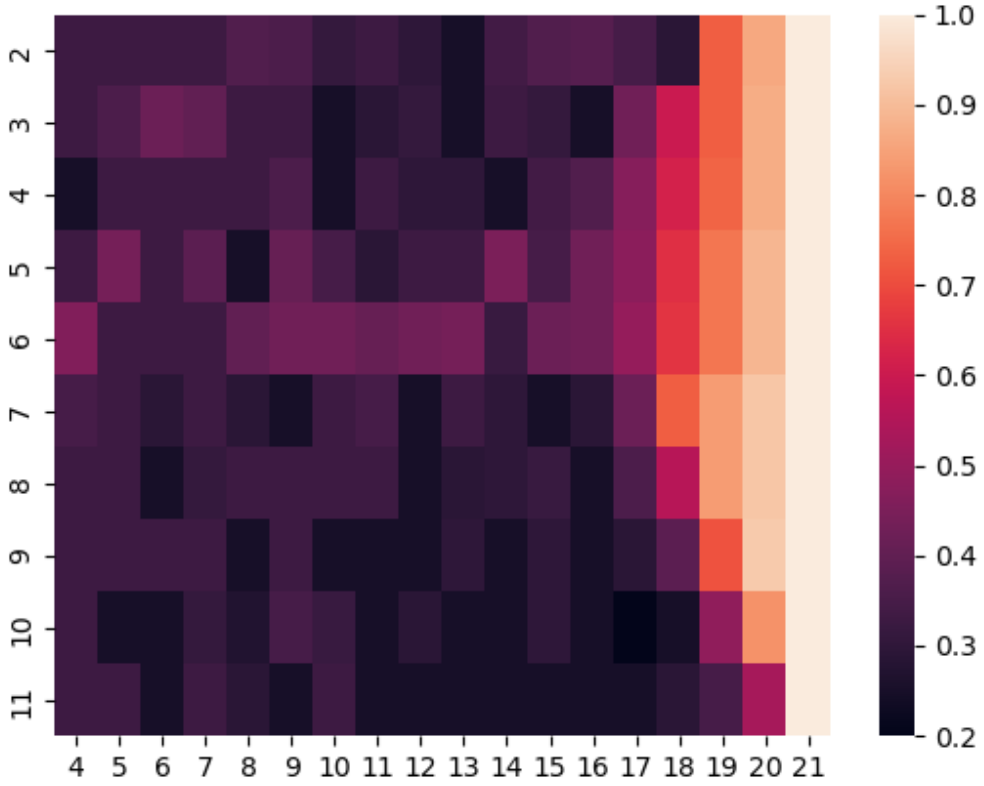
Prostor stanja igre je ograničen na dve dimnzije, prvom predstavljamo sumu vrednosti karata samog dilera, dok drugom predstavljamo sumu vrednosti karata igrača.

Uvodimo tri tabele za beleženje rezultata simulacija: hit_play, check_win, check_play. Dimenzije ovih tabela odgovaraju prostoru stanja. hit_play tabela predstavlja broj pogrešno odigranih hit akcija, tj. broj odigranih hit akcija koje su dovele do bust-a i time dovela do pobede dilera. check_win tabela predstavlja broj pobeda nekog stanja za igrača nakon što je odigrao akciju check, dok tabela check_play predstavlja broj odigranih check akcija za svako stanje.

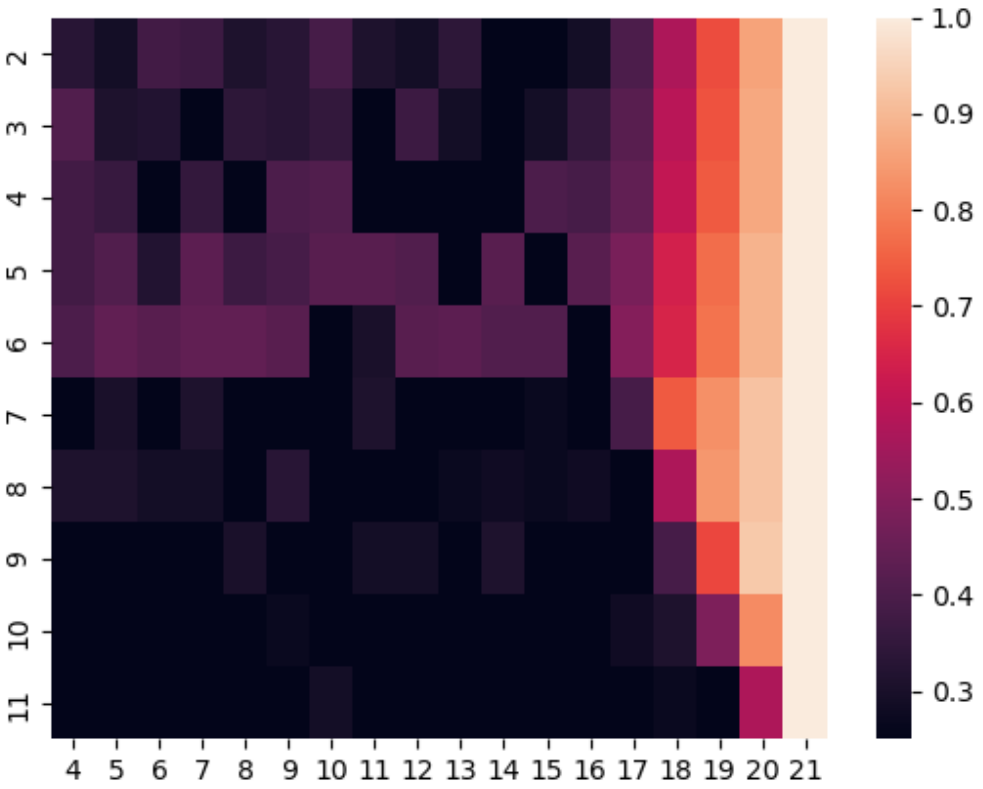
Nakon nekoliko iteracija kroz moguće polise odlučivanja za igrača, došli smo do zaključka da polisa koja predstavlja odnos check_win i check_play donosi najbolje rezultate, agent za svako stanje igre saznaje verovatnoću pobede ako odigra akciju check.



Slika 1. Mapa odlučivanja agenta nakon 100 000 simulacija

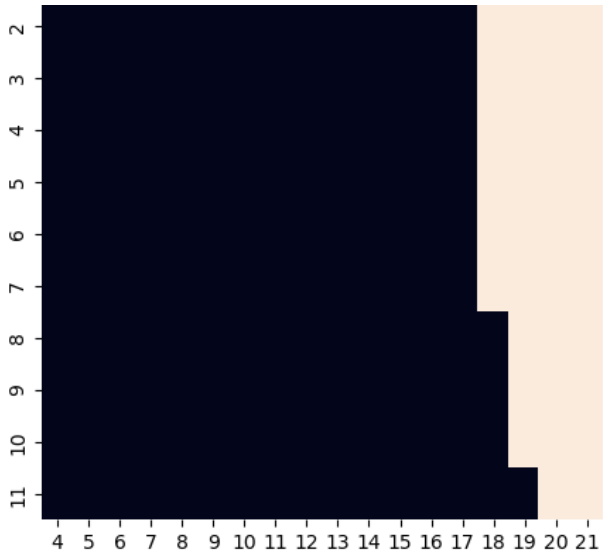


Slika 2. Mapa odlučivanja agenta nakon 1 000 000 simulacija



Slika 3. Mapa odlučivanja agenta nakon 10 000 000 simulacija

Nakon simulacija možemo uočiti, da naš agent igra veoma slično čoveku. Kada je vrednost ruke agenta visoka (>17) najčešće bira da odigra akciju check, jer akcija hit predstavlja velik rizik od bust-a, dok pri nižim vrednostima ruke agent bira akciju hit, kako bi prišao sumi ruke 21. Kao tablicu odluka za našeg agenta dobili smo sledeću:



Slika 4. Matrica odluke agenta treniranog na 10 000 000 simulacija