Üzleti Intelligencia

Megerősítéses tanulás - beadandó feladatok

2023

A feladatok 1-3 skálán vannak osztályozva nehézség szerint, ahol 1 a legkönnyebb és 3 a legnehezebb. A feladatok véletlenszerűen sorsolódnak ki a 6. gyakorlaton. A feladatok beadása Coospace felületen történik, ahol csak egy linket kell beküldeni, ami a feladatot megvalósító Git tárhelyre mutat. Késő beadás nem lehetséges, a dátum beadásakor feltöltött anyagok fognak leosztályozásra kerülni. Minden további információ a tantárgyi útmutatóban és az órán elhangzottakban találhatóak.

1 Duna

Az ügynök Esztergomban áll, és olyan gyorsan kell eljutnia Pestre, amennyire csak lehet. Két út van, amelyekkel elhagyhatja a várost, az 10-es és 11-es út Mindkét út különböző helyre vezet.

Miután megérkezett Budára, két hídon juthat át a Duna folyón, hogy eljusson Pestre. A forgalom kiszámíthatatlan, így előfordulhat, hogy a hídon amit választott forgalmi dugó alakul ki.

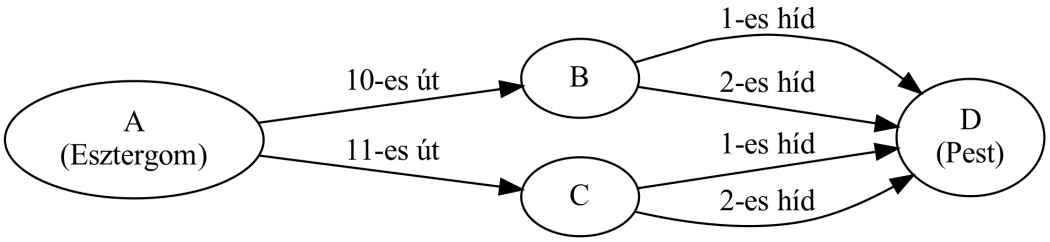
Néha pedig átirányítódik a másik hídra annak ellenére, hogy a másik útvonalat vagy hidat választotta.

A cél az, hogy olyan stratégiát találjon az ügynök, ami a leghamarabb eljuttatja a céljához. A jutalmak a negatív időt jelentik, amelyre szüksége van az adott út/híd átlépéséhez.

Az A, B, C állapotokhoz és minden cselekvéshez a jutalom a következőképpen számolódik ki:

$$R(s,a,s') = \begin{cases} Ha\ nincs\ dugó & R(s,norm) \\ Ha\ dugó\ van & R(s,dugó) \end{cases}$$

Előfordulhat, hogy az ügynök választ egy irányt, de átirányítják a másikra. Tehát ha $P(s,a,s') = 0.9$, tehát 90% valószínűséggel abba az irányba megy, amit választott, de 10% valószínűséggel átirányítják a másik útra.



A környezet dinamikája ($P(s,a,s'), R(s,a,s')$):

$P(A,10,B) = 0.79$, $P_{\text{dugó}}(A,10,B) = 0.9$, $R_{\text{dugó}}(A,10,B) = -60$, $R_{\text{norm}}(A,10,B) = -21$
$P(A,11,C) = 0.89$, $P_{\text{dugó}}(A,11,C) = 0.56$, $R_{\text{dugó}}(A,11,C) = -44$, $R_{\text{norm}}(A,11,C) = -1$
$P(B,1,D) = 0.84$, $P_{\text{dugó}}(B,1,D) = 0.43$, $R_{\text{dugó}}(B,1,D) = -20$, $R_{\text{norm}}(B,1,D) = -4$
$P(B,2,D) = 0.7$, $P_{\text{dugó}}(B,2,D) = 0.84$, $R_{\text{dugó}}(B,2,D) = -40$, $R_{\text{norm}}(B,2,D) = -11$
$P(C,1,D) = 0.98$, $P_{\text{dugó}}(C,1,D) = 0.75$, $R_{\text{dugó}}(C,1,D) = -65$, $R_{\text{norm}}(C,1,D) = -5$
$P(C,2,D) = 0.77$, $P_{\text{dugó}}(C,2,D) = 0.53$, $R_{\text{dugó}}(C,2,D) = -58$, $R_{\text{norm}}(C,2,D) = -10$

1.1 Dinamikus programozás (2)

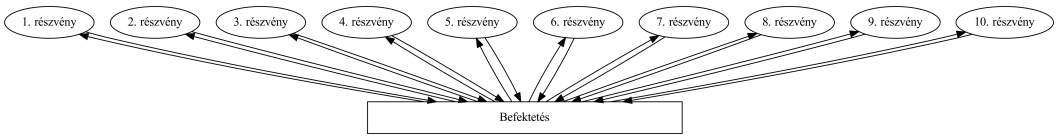
Oldja meg a problémát dinamikus programozással. Ehhez tartozóan implementálja a politika iteráció eljárását (politika kiértékelés, politika javítás). Futtassa a szimulációt adott konvergencia kritériumig. Mutassa meg, hogy az ügynök megtanulta optimalizálni a jutalmat. Írja le röviden az eljárás elméleti alapjait és a saját tapasztalatait egy Jupyter markdown cellában.

1.2 Sztochasztikus becslés (2)

Oldja meg a problémát sztochasztikus becsléssel. Ehhez tartozóan implementálja a Monte Carlo és időbeli különbségek politika keresési eljárásait. Futtassa a szimulációt adott konvergencia kritériumig. Mutassa meg, hogy az ügynök megtanulta optimalizálni a jutalmat. Írja le röviden az eljárás elméleti alapjait és a saját tapasztalatait egy Jupyter markdown cellában.

2 Tőzsde

Az ügynök egy tőzsdén akar befektetni és eladni. A tőzsde úgy működik, hogy minden reggel 10 részvény közül választhat, ami zárás után dől el, hogy jövedelmezett-e vagy nem. Minden nap csak egy részvényt választhat.



Az ügynök a jutalmát a befektetésekből kapja meg. Minden befektetésnek a jutalma egy normális eloszlásból származik, ahol μ a várható érték és σ a variancia.

Részvény	1	2	3	4	5	6	7	8	9	10
μ	0.76	-1.23	1.54	-0.48	1.94	-0.82	-1.27	0.92	-1.99	1.30
σ	0.34	0.72	0.61	0.87	0.28	0.51	0.18	0.43	0.95	0.75

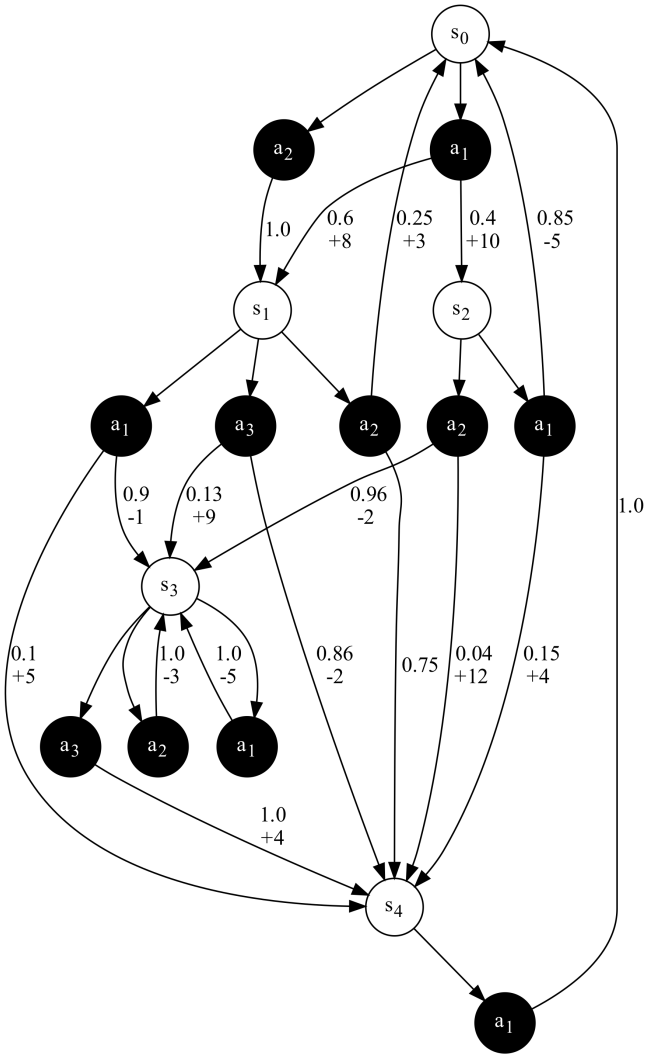
Minden nap az ügynök választhat, hogy a legjobban jövedelmező befektetést választja, vagy kockáztat és véletlenszerűen teszi meg a tétet.

2.1 ε-mohó ügynök (1)

Implementálja a befektetés környezetét és optimalizálja a befektetési stratégia ε értékét. Mutassa meg, hogy az ügynök képes optimalizálni a jutalmat a tanítási iterációk alatt. Adja meg az optimális ε értéket 2 tizedesjegy pontossággal. Hasonlítsa össze a mohó stratégia és az optimális stratégia jutalmait az egyes részvényeken. Ábrázolja az összehasonlítást. Írja le röviden az eljárás elméleti alapjait és a saját tapasztalatait, eredményeit egy Jupyter markdown cellában.

3 MDP

Implementálja az alábbi $MDP(S, A, P, R, s_0, \gamma)$ Markov döntési folyamatot. Az állapotok valószínűsége tizedes törtekként szerepel, a jutalmak pedig + és − előjellel a kapcsolatokon. Használja az alábbi paramétereket: $s_0 = 0,\ \gamma = 0.98$



3.1 Bellman egyenletek (1)

Számolja ki a $V(s)$ és $Q(s, a)$ értékeket a Bellman egyenlettel a környezeti dinamikát felhasználva. Mutassa meg, hogy az értékek a valóshoz konvergálnak. Írja le az eljárás elméleti alapjait és a saját tapasztalatait röviden egy Jupyter markdown cellában.

3.2 Q-tanulás (2)

Implementálja a Q-tanulás és dupla Q-tanulás algoritmusát az adott Markov döntési folyamatra. Ábrázolja és hasonlítsa össze a Q-táblákat illetve az értékek konvergenciáját mindkét esetben. Írja le az eljárás elméleti alapjait és a saját tapasztalatait röviden egy Jupyter markdown cellában.

4 Teljes MDP

Adott az alábbi programkód, ami egy $MDP(S, A, P, R, s_0, \gamma)$ Markov döntési folyamatot definiál. Másolja be a feladat megoldásához tartozó Jupyter notebook első cellájába és végig hagyja módosítatlanul.

```
import numpy as np

random_seed = 6556
np.random.seed(random_seed)

n_S = 10
n_A = 3
S = list(range(0, n_S))
A = list(range(0, n_A))
p_s_a = np.zeros((n_S, n_S, n_A))
R = np.random.randint(0, 100, (n_S, n_A))

for a in range(n_A):
    for i, tpr in enumerate(p_s_a[:, :, a]):
        choosen_indexes = np.random.choice(n_S, 3, replace =
            False)
        p_dist = np.random.random(3)
        p_sum = sum(p_dist)
        p_dist = [i/p_sum for i in p_dist]
        for j, k in enumerate(choosen_indexes):
            tpr[k] = p_dist[j]

def step(s, a):
    sp = np.random.choice(n_S, 1, p=p_s_a[s, :, a])[0]
    r = R[s, a]
    return sp, r
```

A feladat megoldása során használja a következő paramétereket: $s_0 = 0, \gamma = 0.99$.

4.1 Mélytanulás (3)

Oldja meg a Markov Döntési folyamatot mély Q-tanulással és dupla mély Q-tanulással. Mutassa meg, hogy az ügynök képes optimalizálni a jutalmakat az iterációk során. Hasonlítsa össze az eljárásokat. Ábrázolja az ügynök viselkedését animáción egy epizódon keresztül. Írja le az eljárások elméleti alapjait és a saját tapasztalatait egy Jupyter markdown cellában.

5 FrozenLake-v1

Implementáljon megerősítéses tanulási algoritmust a FrozenLake-v1 megerősítéses tanulási környezetre:

A következő paramétereket használja: `desc=None, map_name="8x8", is_slippery=True, render_mode='rgb_array'`

5.1 Sztochasztikus becslés (2)

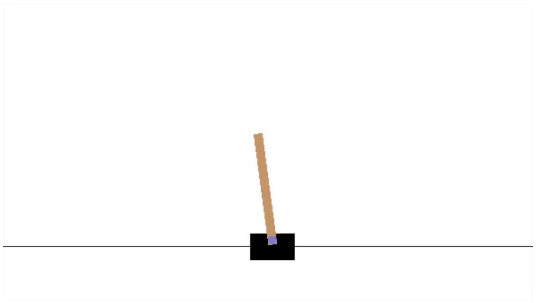
Oldja meg a FrozenLake-v1 megerősítéses tanulási környezetet Monte Carlo illetve temporális különbségek algoritmusával. Ábrázolja az állapot-értékek konvergenciáját mindkét esetben. Mutassa meg, hogy az ügynök képes optimalizálni a jutalmakat az iterációk alatt. Hasonlítsa össze az eljárásokat. Ábrázolja az ügynök viselkedését animáción egy epizódon keresztül. Írja le az eljárások elméleti alapjait és a saját tapasztalatait egy Jupyter markdown cellában.

5.2 Mélytanulás (3)

Oldja meg a környezetet dupla mély Q-tanulással és párbajozó mély Q-tanulással. Mutassa meg, hogy az ügynök képes optimalizálni a jutalmakat az iterációk során. Hasonlítsa össze az eljárásokat. Ábrázolja az ügynök viselkedését animáción egy epizódon keresztül. Írja le az eljárások elméleti alapjait és a saját tapasztalatait egy Jupyter markdown cellában.

6 CartPole-v1

Implementáljon megerősítéses tanulási algoritmust a CartPole-v1 megerősítéses tanulási környezetre:



6.1 Mélytanulás (3)

Oldja meg a környezetet dupla mély Q-tanulással és dupla mély Q-tanulással. Mutassa meg, hogy az ügynök képes optimalizálni a jutalmakat az iterációk során. Hasonlítsa össze az eljárásokat. Ábrázolja az ügynök viselkedését animáción egy epizódon keresztül. Írja le az eljárások elméleti alapjait és a saját tapasztalatait egy Jupyter markdown cellában.

6.2 Mélytanulás (3)

Oldja meg a környezetet párbajozó mély Q-tanulással és aktor-kritikus architektúrával. Mutassa meg, hogy az ügynök képes optimalizálni a jutalmakat az iterációk során. Hasonlítsa össze az eljárásokat. Ábrázolja az ügynök viselkedését animáción egy epizódon keresztül. Írja le az eljárások elméleti alapjait és a saját tapasztalatait egy Jupyter markdown cellában.