Marcu-Nicolescu Cezar-George

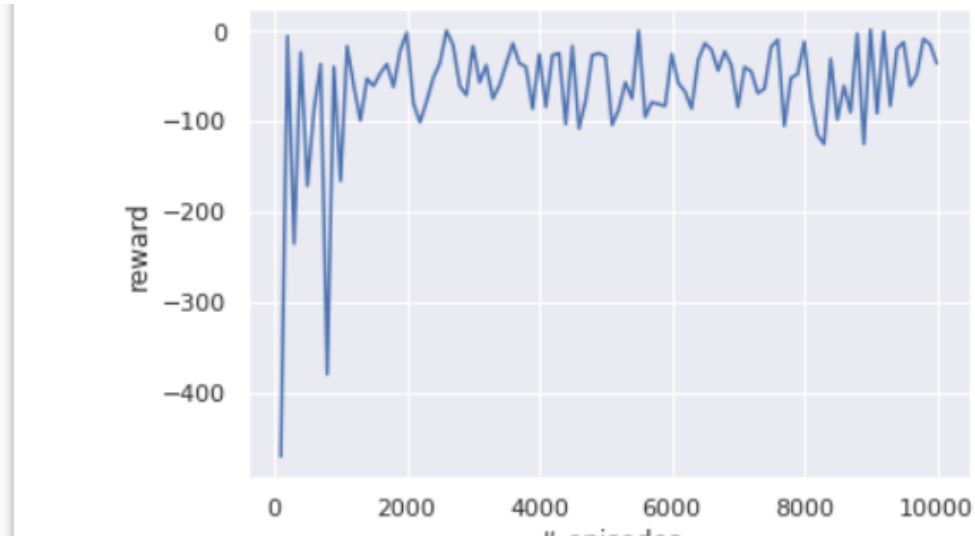
341C2

Tema 2 – Taxi Driver

Am inceput aceasta tema prin a ma documenta din resursele puse la dispozitie. Am observant ca variabila Q in care sunt stocate perechile stare – actiuni este un dictionar compus din mai multe stari ce au la randul lor un alt dictionar format din perechea actiuni posibile (N, S , E, W, pickup, dropoff) si rewardul pt fiecare stare in parte. Asadar, pentru a face choose action am realizat un random si am ales una din acele actiuni in baza probabilitatii (daca era indeplinita conditia epsilon < randomul respectiv atunci se explora si se putea alege aleator una din cele 6 posibilitati). Pentru best action am folosit argmax pe Q[state], deoarece imi intorcea actiunea cu reward-ul cel mai mare.

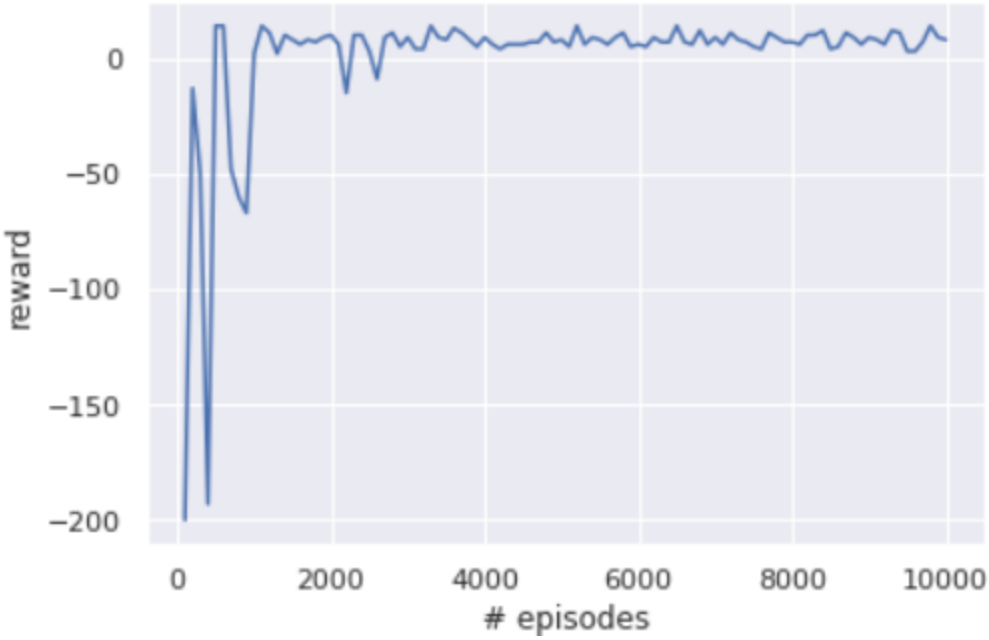
Pentru implementarea Q-learning-ului am ales actiunea cu choose action, apoi am facut pasul “inainte” in urmatorul state a actiunii proaspat alese. Am verificat cea mai buna actiune posibila a acestei stari si am retinut intr-o noua variabila reward-ul celei mai buni actiuni posibile din next state pentru a-l aplica in forma de calcul a reward-ului current.

Q-learning fara decay :



Implementarea decay-ului am implementat-o in felul urmator : la fiecare DECAY_EPS_EPISODES epsilon = epsilon * decay_epsilon (am preferat inmultire deoarece, intr-un nr mare de episode acesta ar fi pozitiv)

Q-learning cu decay :

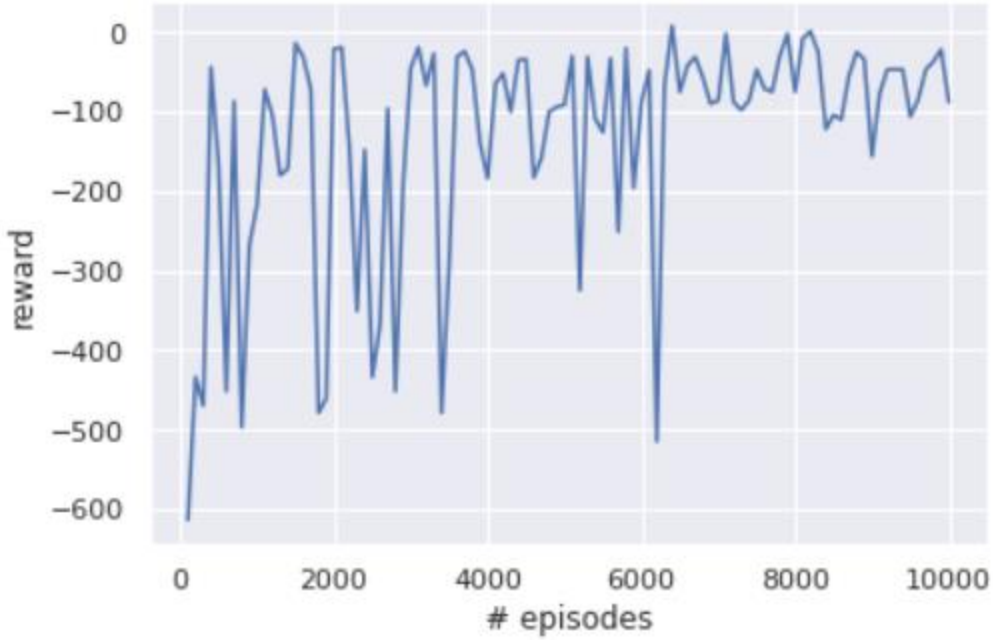


O concluzie : La inceput este importanta explorarea, si agentul trebuie sa invete environment-ul, insa in timp, rezultatele ar trebui sa fie primordiale, asadar prin decay se mentine accentul pe rezultate in timp (exploatarea dupa invatare).

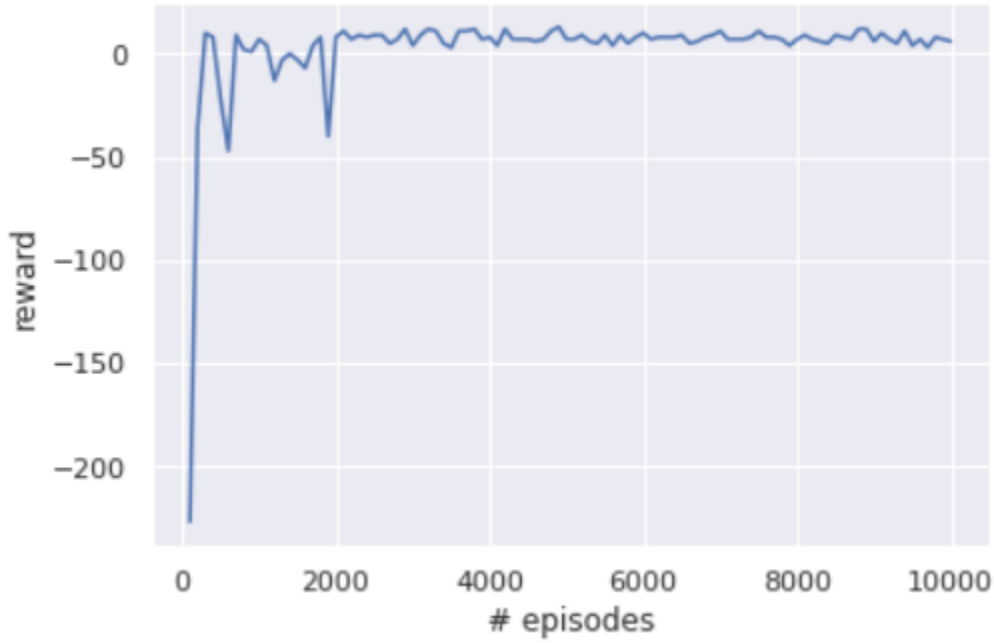
Am implementat algoritmul Sarsa in felul urmator : dupa ce se alege actiunea cu choose action, se trece in state-ul respectiv acestei actiuni, iar apoi se verifica urmatoarea actiune posibila (tot cu choose state, deci ramane la probabilitatea random-ului daca alege best action). Se preia reward-ul perechii next_state, next_action si se calculeaza reward-ul curent in baza celui preluat din actiunea viitoare. Actualizez state-ul cu next state pentru a avansa in computatie.

Ca o parere personala : consider Sarsa ca un Q-learning mult mai realist.

SARSA fara decay :



SARSA cu decay :



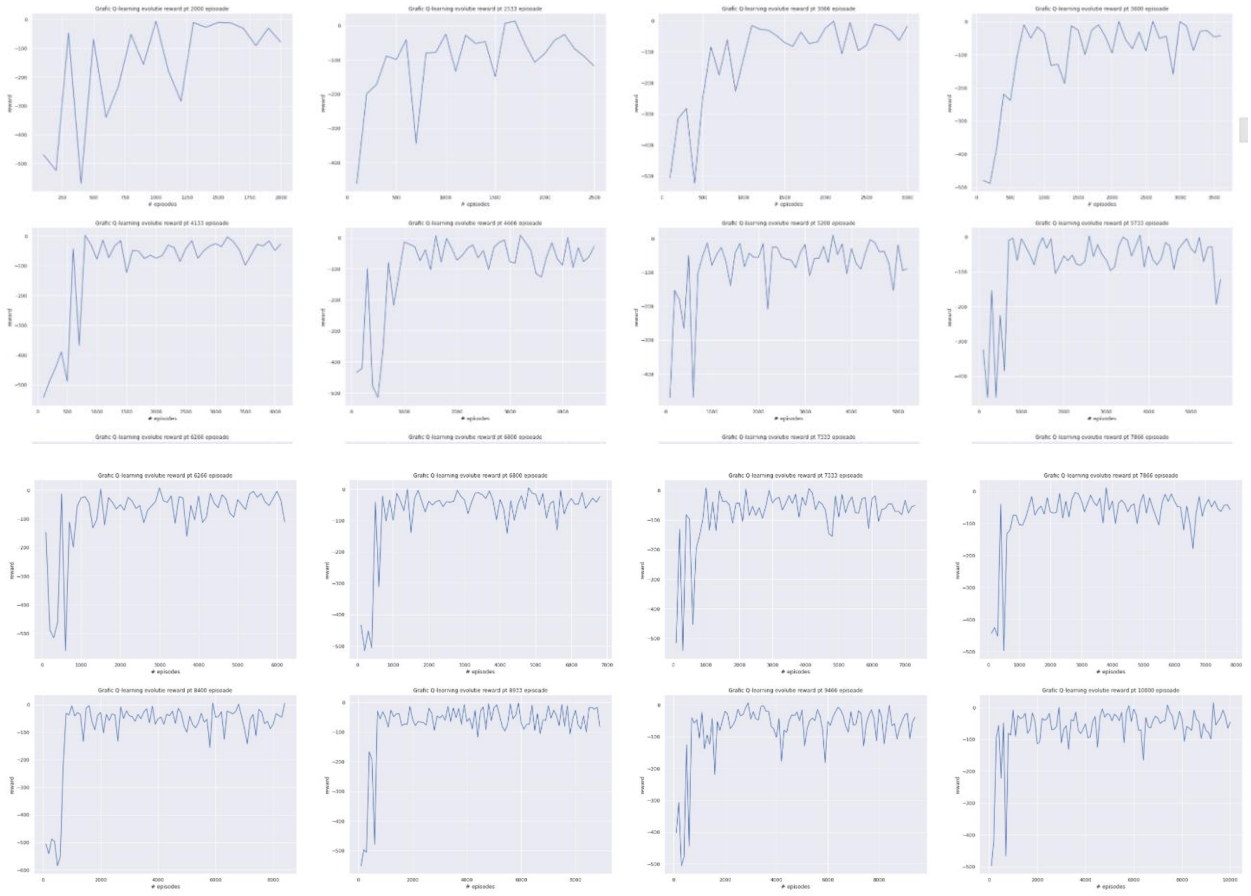
Concluzie :

In momentul in care SARSA incepe sa aleaga best action mai des (asta se intampla dupa cateva episoade de explorare si cateva decay-uri cu 5%) acesta incepe sa isi actualizeze informatiile mult mai relevant decat Q-learning, avand experiente mai “realiste” si o cunostinta asupra environment-ului mult mai clara.

Graficele pe care le voi adauga incat sa evidentiez performantele algoritmilor in baza parametrilor vor fi fara decay pentru a evidentia un impact clar fara alte imbunatatiri.

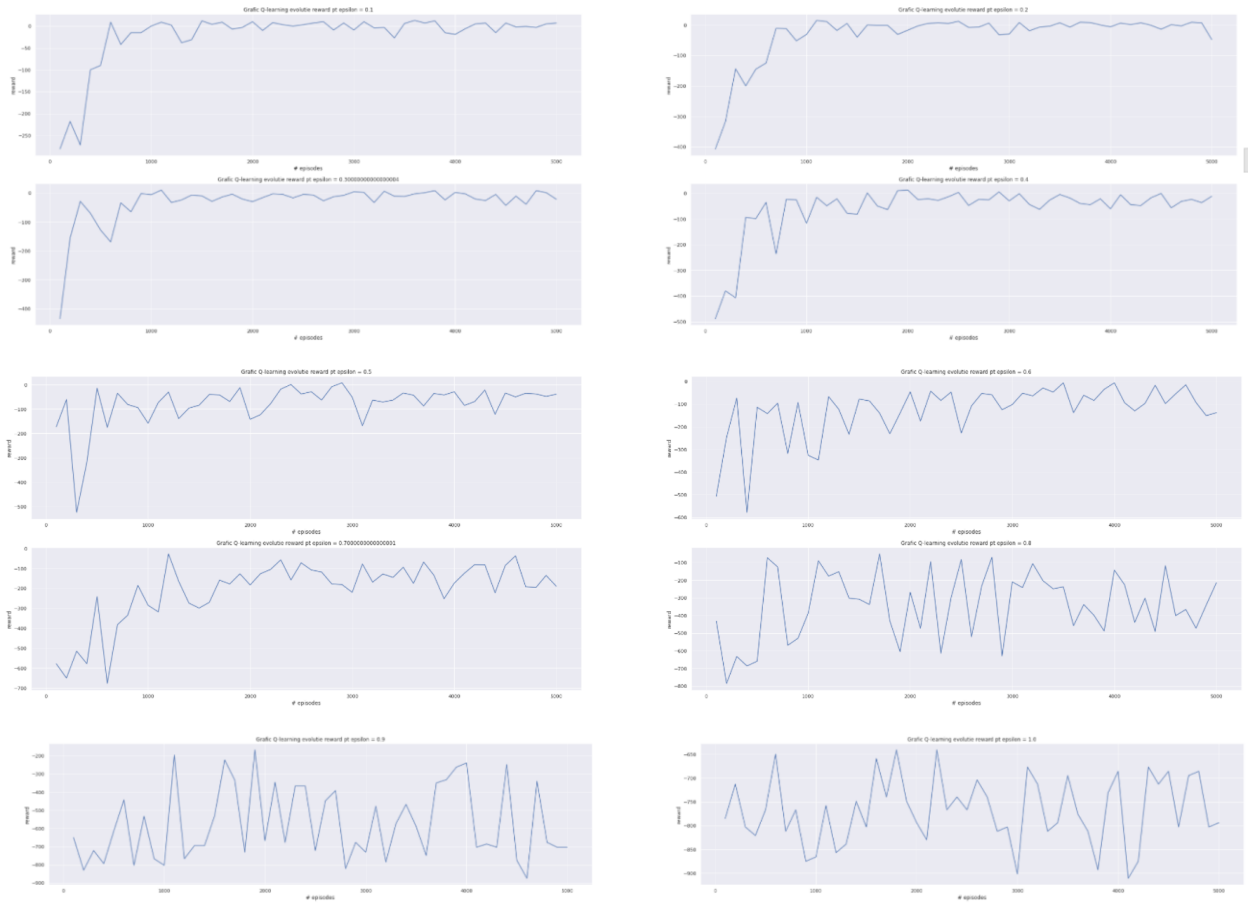
Pentru a crea graficele am format functii ale Q-learning-ului si ale algoritmului SARSA pentru usurinta.

Grafice in functie de numarul de episoade pt Q-learning:



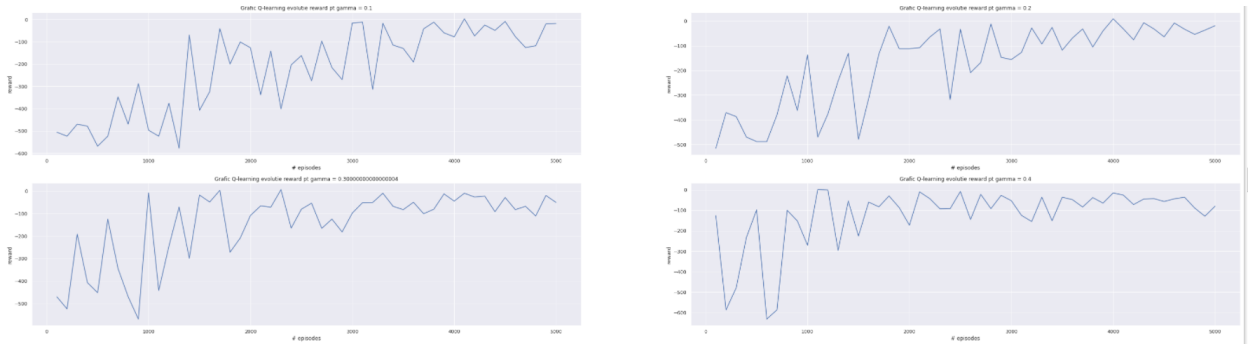
Se poate observa ca algoritmul converge spre stabilitate dupa un numar de episoade. Inca sunt fluctuatii maricele ale reward-ului deoarecele el nu face constant best action ci, merge si pe actiuni random. Dupa aproximativ 1000 de episoade el gaseste reward-urile si incepe sa inteleaga foarte bine scena.

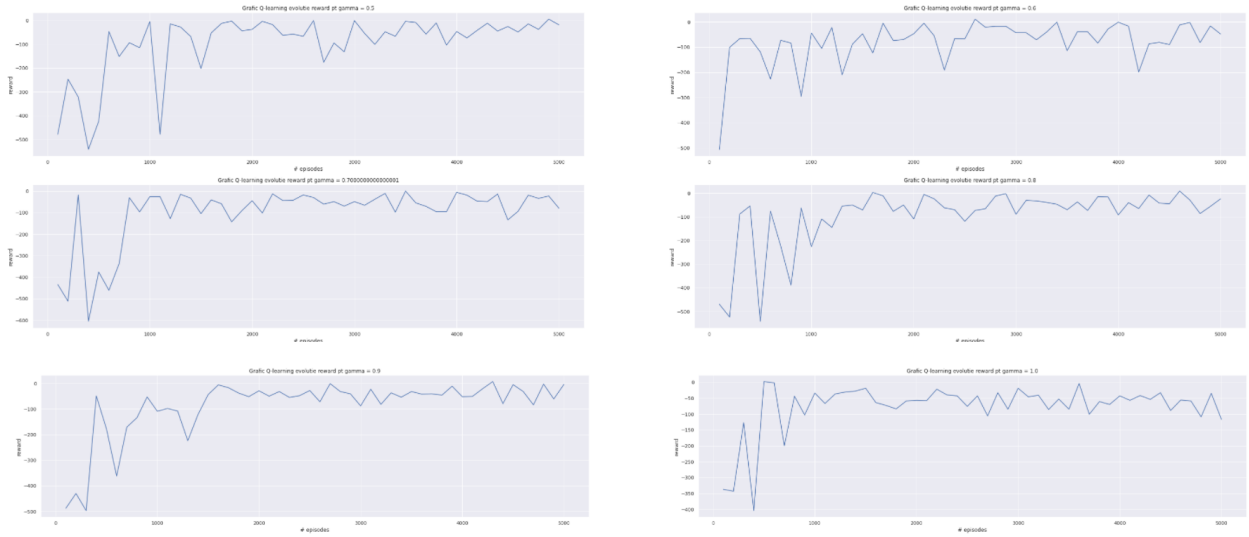
Grafice Q-learning pentru valori diferite ale epsilon-ului:



Putem observa discrepanta din ce in ce mai marita a reward-urilor in aceste grafice. Cu cat crestem epsilon-ul cu atat explorarea devine din ce in ce mai scazuta, algoritm-ul are nevoie de cunostinte despre scena si de situatii care sunt “mai putin importante” pentru a diferentia mai eficient deciziile pe viitor. Asadar, diferenta dintre explorare si exploatarea trebuie foarte bine definita, caci daca algoritmul functioneaza pe baza unei politici extrem de lacome, sistemul este destabilizat.

Grafic Q-learning pe baza valorilor diferite ale lui gamma:

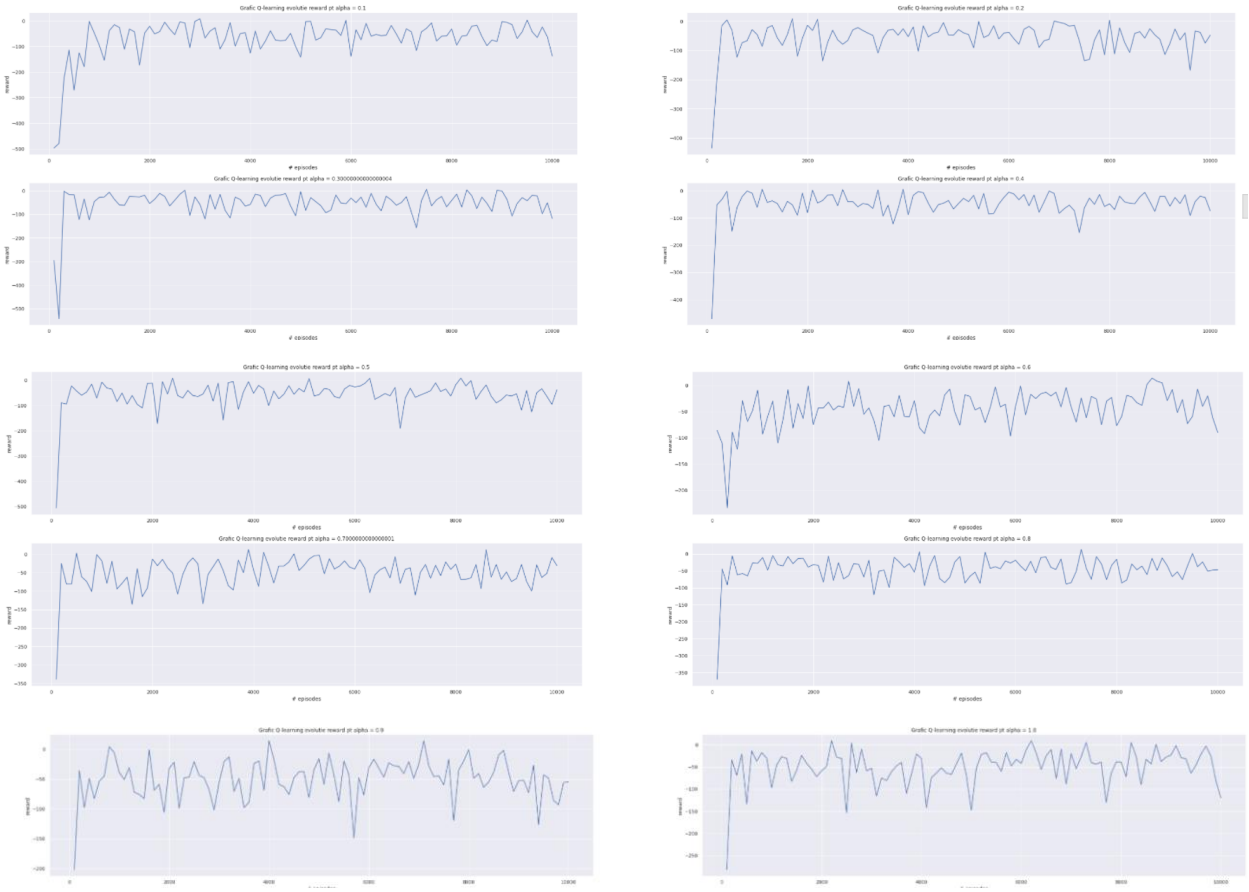




Eu observ acest parametru ca impactul cunostintelor proaspat dobandite asupra prezentului :

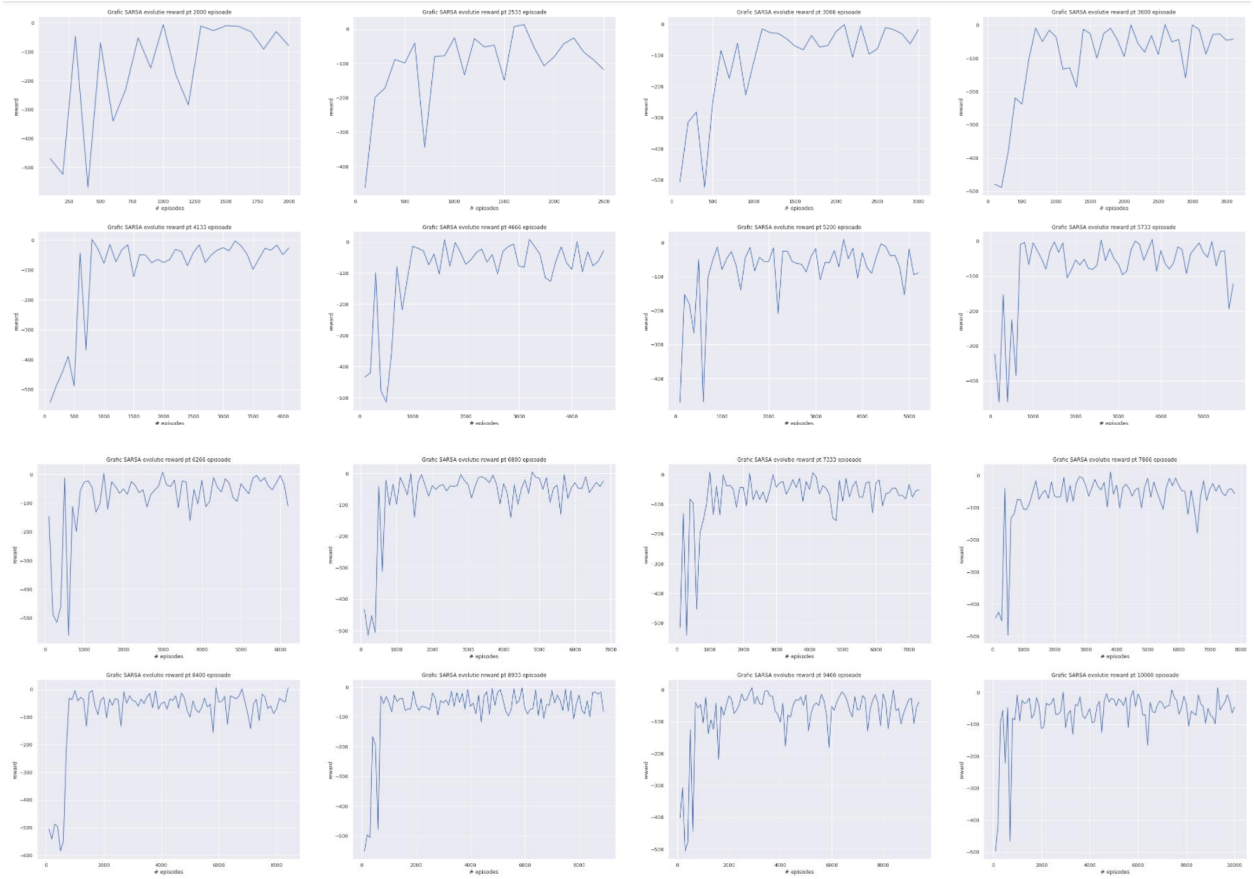
Acest parametru merge foarte bine cu Q-learning deoarece el alege best action-ul, iar atunci drumul cel mai eficient este favorizat extrem.

Grafice Q-learning cu reward-uri pt valori diferite ale parametrului alpha :



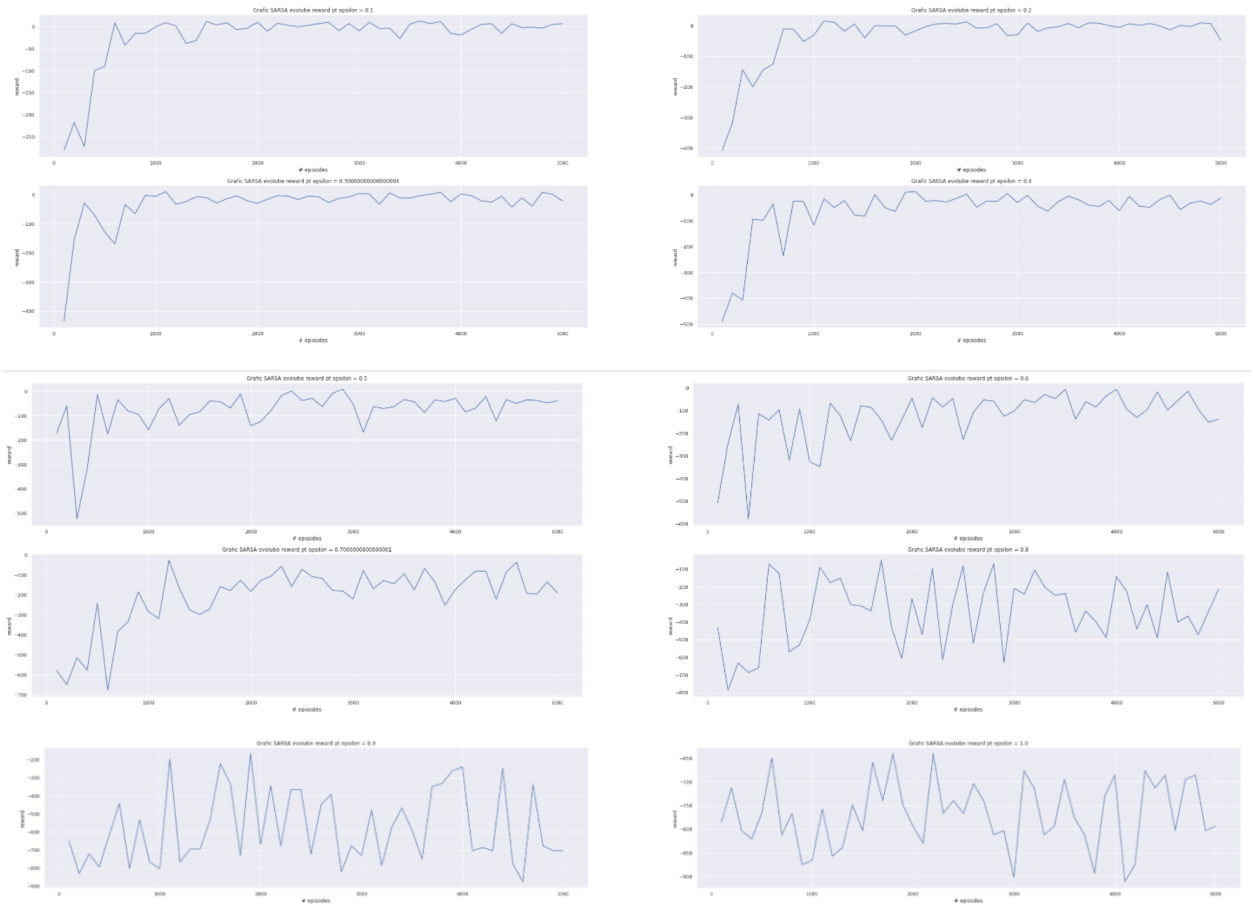
Alpha-ul il observ ca fiind coeficientul ce determina contributia totala a ceea ce am invatat, adica cat de important e sa retin atat din trecut cat si din present. Rewardurile acestuia in baza valorilor sunt determinate de ce actiuni aleg. Actiuni random ce ma duc pe cazuri cu reward negativ, atunci spike-uri negative mari daca alpha e mare, invers pentru actiuni positive. Un alpha mai micut poate determina un progres intre episoade si deciziile agentului pe baza cunostintelor acumulate.

Grafice SARSA pentru un nr diferit de episoade :



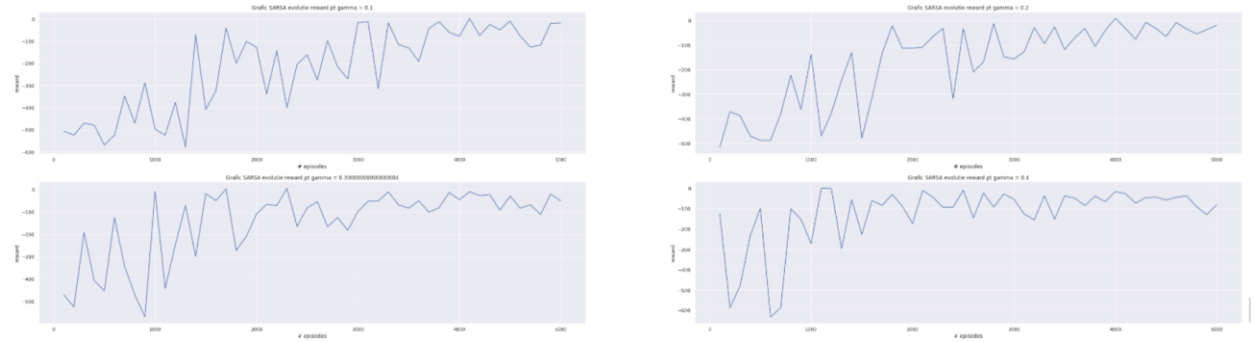
Putem observa mult mai multe fluctuatii la SARSA decat la Q-learning, iar acest lucru este datorat faptului ca acesta calculeaza reward-ul curent in baza actiunilor alese prin epsilon greedy, nu best action-uri. SARSA tinde sa aiba o curba pozitiva si sa se stabilizeze mai repede la un anume prag decat Q-learning (Acesta da de situatii pe care nu le-a mai intalnit si incepe sa aplice best action pt calculul reward-ului, insa in multe situatii nu are dreptate din prima, pe cand SARSA exploreaza si in acele situatii de la bun inceput pe cat ii permite probabilitatea)

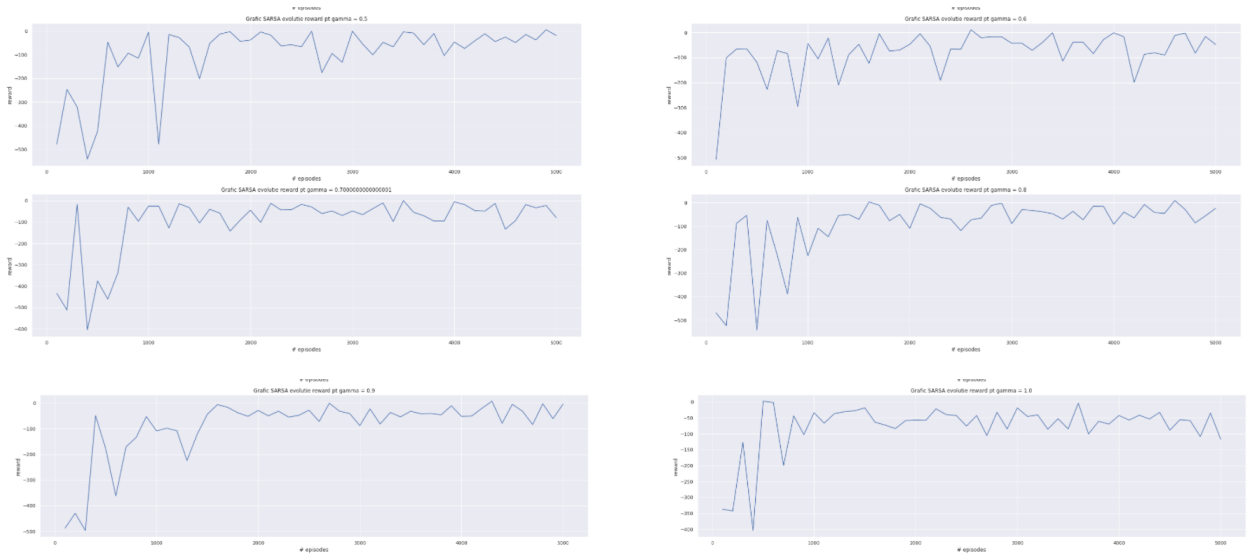
Grafice SARSA cu reward-uri in baza valorilor epsilon:



Concluzie : Prin a creste epsilon e ca si cum ai obliga SARSA sa adopte o politica mai Greedy. Dupa valorile reward-urilor se pot interpreta graficele si putem ajunge la concluzia ca fara explorare nu se pot obtine rezultate satisfacatoare.

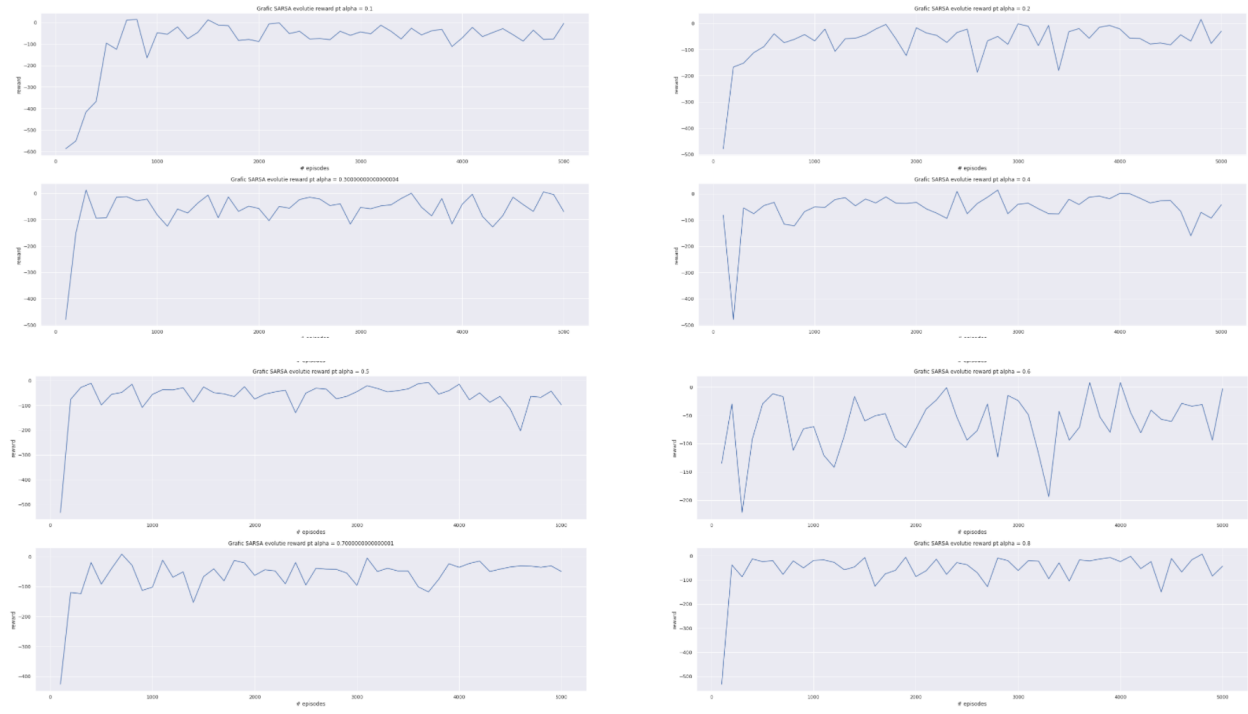
Grafice SARSA pentru valori diferite gamma:





Concluzie : actiunile din trecut au un impact mai eficient. In momentul in care nu se exploreaza reward-urile se stabilizeaza. Cu cat gamma este mai mare cu atat agentul retine cat mai multe din experientele precedente.

Grafice SARSA pentru rewardurile evidentiate de valori diferite ale parametrului alpha:



Acest parametru merge mult mai bine pe SARSA decat pe Q-learning, iar acest lucru se datoreaza faptului ca reward-ul actual se calculeaza in functie de o actiune posibila din viitor (nu neaparat best action-ul)