DRL Course Домашнее задание 3

Алексей Матушевский

Динамическое программирование

Policy и Value Итерации

Марковский процесс решения задачи (Markov decision process, MDP) является полезным инструментом для моделирования и анализа систем с разными формами неопределенности. Он используется для оптимизации решений в ситуациях, где известно текущее состояние системы и возможные действия, но неизвестны вероятности перехода из одного состояния в другое.

Одним из эффективных методов решения MDP является итерация политики и итерация ценности (Policy and Value Iteration), которая последовательно обновляет и улучшает политику действий для системы. В этой работе мы будем изучать
- влияние параметра gamma на результаты работы алгоритма Policy Iteration.
- виляние начального состояния Value на конечную политику
- реализацию Value Iteration

В этой домашней работы я постарался запрограммировать практику самостоятельно, без просмотра видео. Мне частично это удалось. Итоговый код не сильно отличается от кода разработанного в видео лекции. Главное отличии — это использование 3х мерной матрицы, вместо dict. В сравнении реализаций, я заметил что мой код работает чуть быстрее. Видимо разница в неиспользовании словаря дает преимущество.

Задание 1

Параметр Gamma

Параметр gamma, как известно, определяет вес дальнейших вознаграждений в функции оценки стоимости состояния. В этой работе мы будем исследовать, как изменение значения gamma влияет на результаты работы MDP и что это означает для поиска оптимального стратегического плана. Для этого мы будем использовать метод политики итераций, один из наиболее эффективных методов для решения MDP.

Для изучения параметра, мы будет рассматривать различные диапазоны, также фокусироваться на более интересных отрезках значений. Для поиска политики мы будем использовать следующие параметры

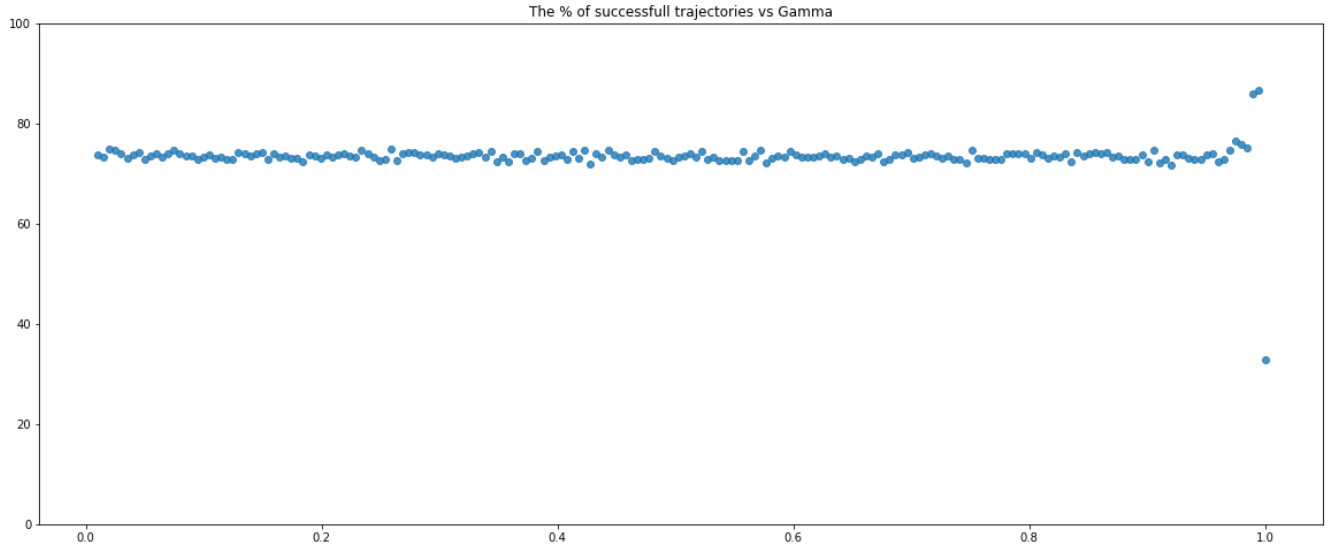
K = 100 итераций Policy Improvement с

L = 100 итераций Policy Evaluation

Для тестирования полученной политики мы проведем 5000 тестов (траекторией). В каждой из траекторий мы посчитаем среднее % успешно завершенных траекторий. Хотя в домашнем задании говориться о подсчете средней оценки, что по факту является абсолютно идентичным параметром. То есть - За успех мы получаем 1 бал награды, а за провал 0 балов. Посчитав среднее значение награды, сводится к подсчету количества_успешных_траекторий/5000. С точки зрения повествования мне кажется метрика % успешных завершенных траекторий более простым, понятным.

Gamma = [0.01, 1] 200 значений

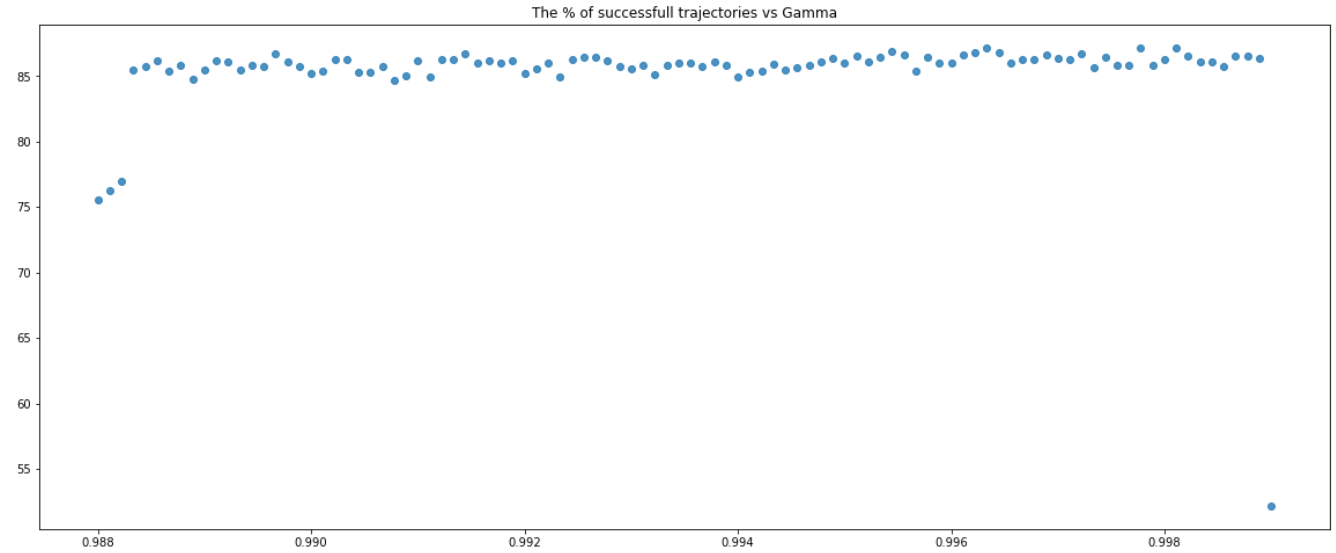
```
gammas = np.linspace(0.011,1, num=200)
```



Мы видим что в среднем значение успешных траекторий = 0.7343

в конце диапазона последние 3 значений резко отличаются от 197 предыдущих. Проверка этих диапазонов показал что самые высокие результаты получаются в диапазоне 0.98833333 до 0.99888889

```
gammas = np.linspace(0.988,0.999, num=100)
```



Задание 2

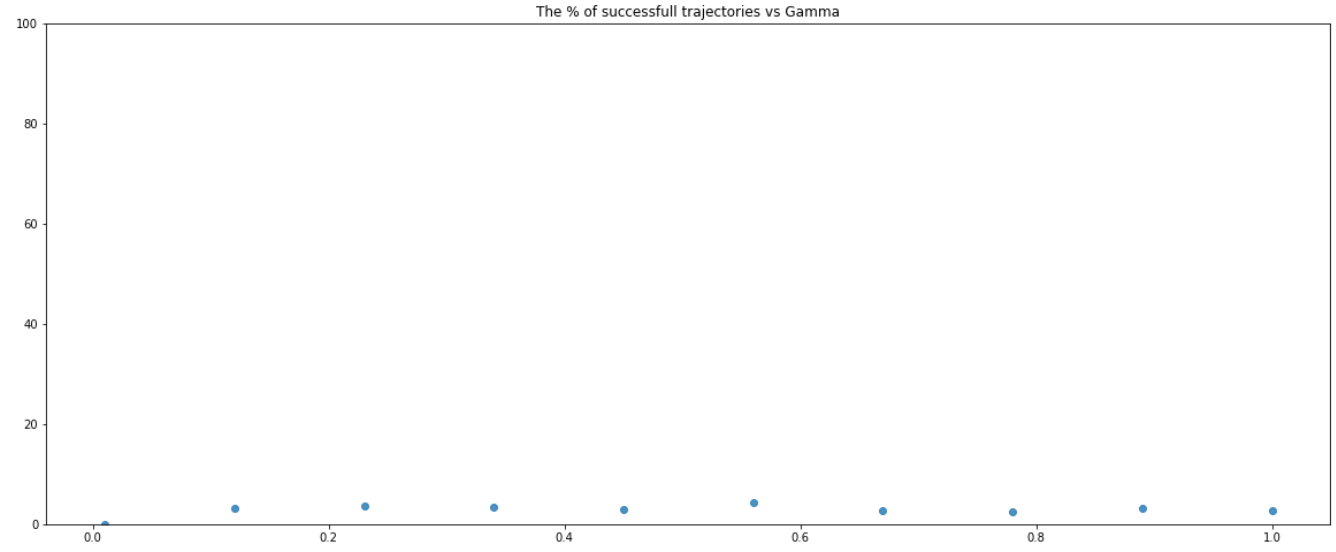
На шаге Policy Evaluation мы каждый раз начинаем с нулевых values. А что будет если вместо этого начинать с values обученных на предыдущем шаге? Будет ли алгоритм работать? Если да, то будет ли он работать лучше?

Для выполнения этой задачи мы изменим буквально одну строку:

```
def policy_evaluation_step(policy, values, gamma):
    new_values = values.copy()
    # вместо: new_values = np.zeros(policy.shape[0]*policy.shape[1])
```

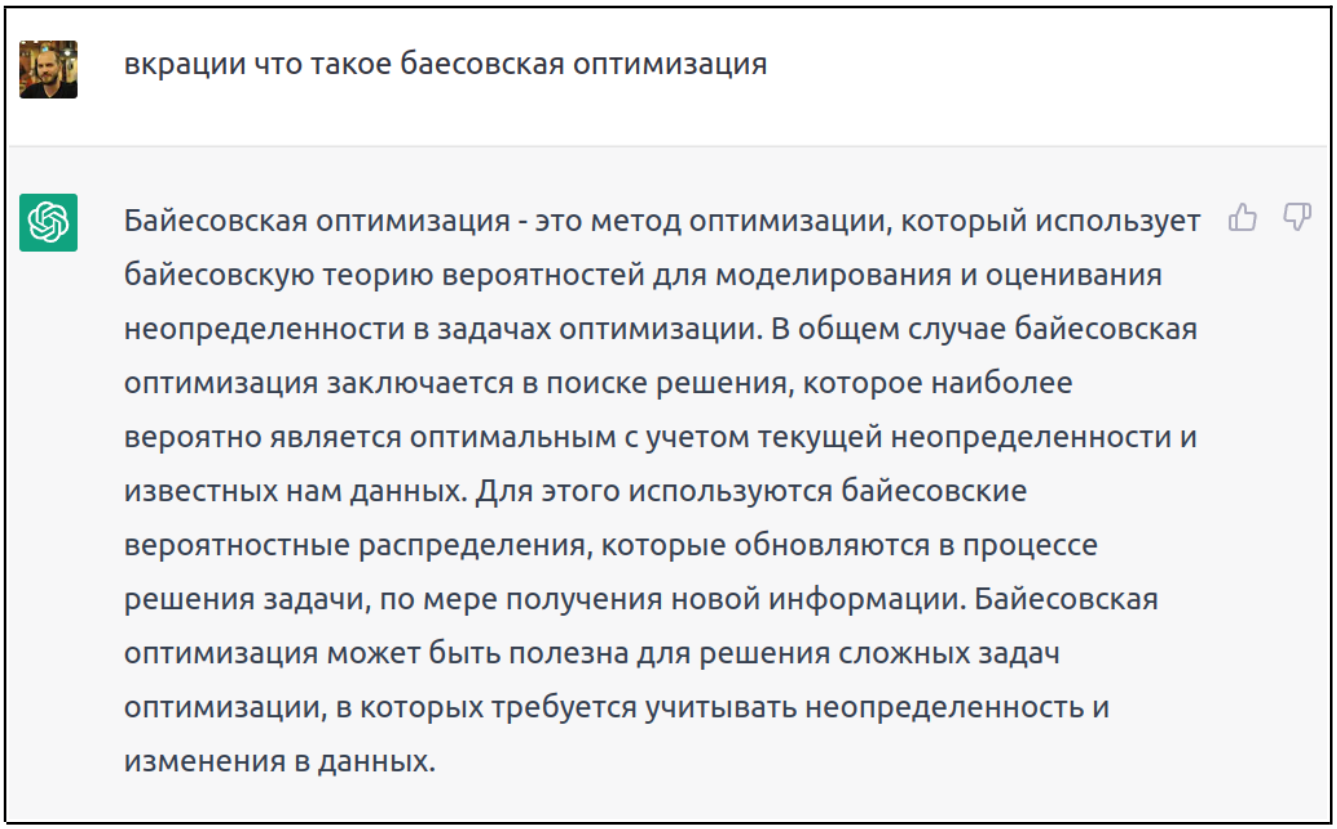
таким образом мы будем использовать предыдущие значения.

С такими изменениями алгоритм уже не дает результатов при K=100 и L=100.



10 экспериментов с разными гамма

Чтобы найти рабочие параметры для нового алгоритма воспользуемся Оптимизацией Байеса. Что такое ОБ, мне поможет объяснить ChatGPT



В качестве ограничений поиска зададим

gamma =[0.01, 0.999]

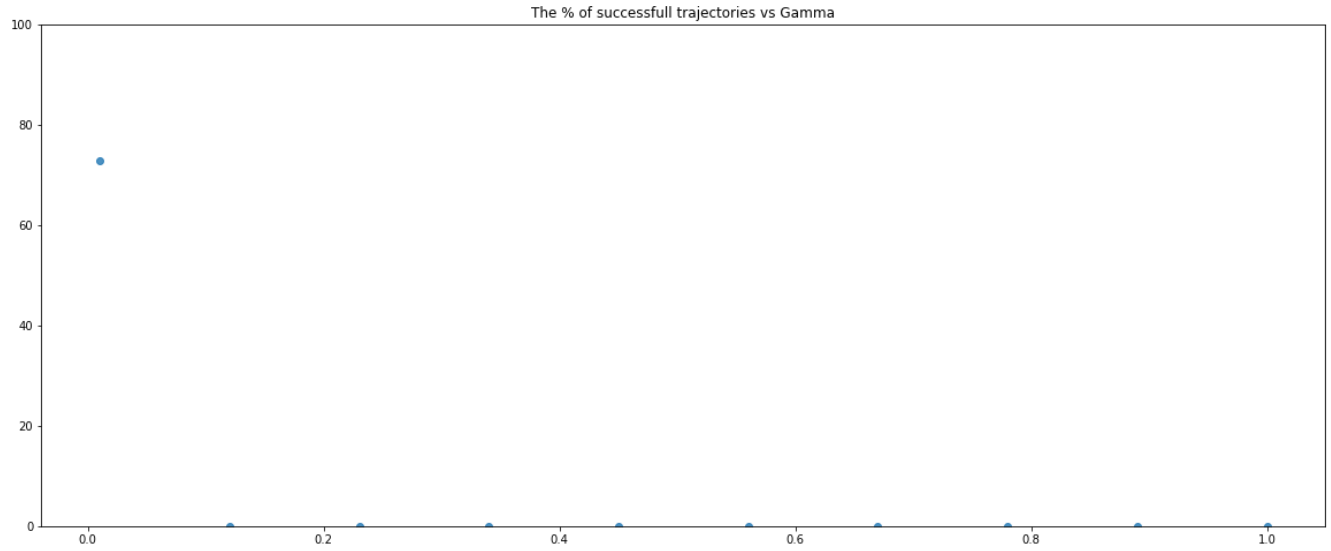
K= [1,100]

L = [1,100]

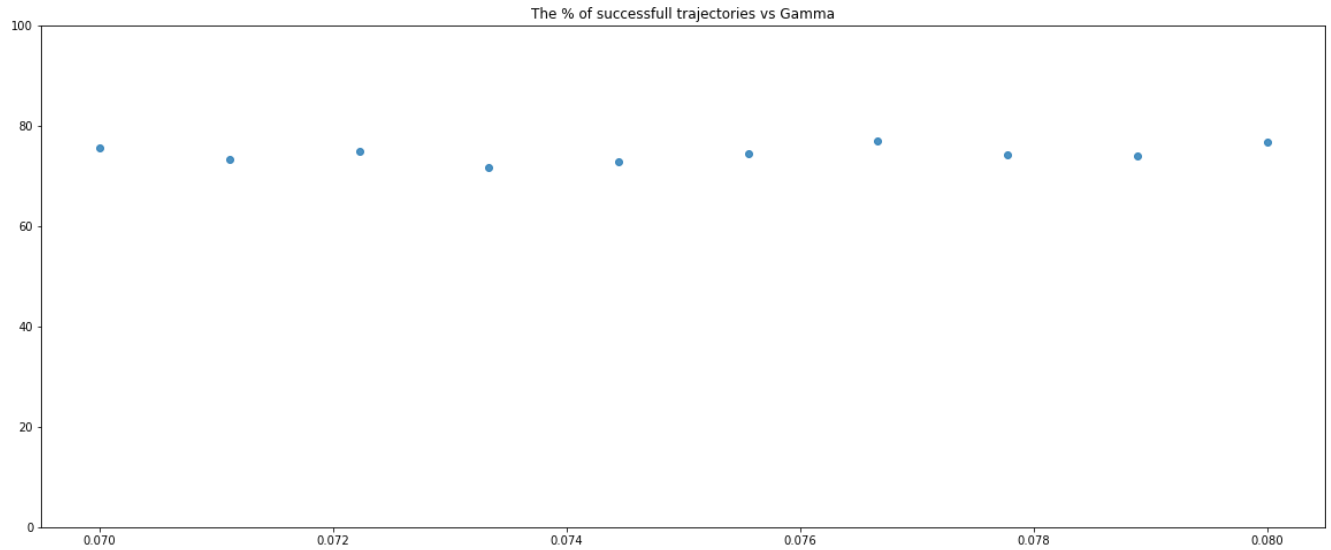
После 100 итераций ОБ выдает следящие параметры как наилучшие:

```
K = 36
L = 1
gamma = 0.0741
```

С результатом в 74.24% успешных траекторий из 5000 экспериментов.



K=36, L=1, gamma[0.01, 0.12, 0.23, 0.34, 0.45, 0.56, 0.67, 0.78, 0.89, 1.]



K=36, L=1 gamma=[0.07, 0.071, 0.072, 0.073, 0.074, 0.075, 0.076, 0.07, 0.078, 0.08]

Чтобы объяснить успех таких изменения с подобными параметрами я бы описал это так. L=1 практически приводит к тому что мы шаг Policy Evaluation — становиться просто одним шагом, а не итерацией. Низкая gamma практический сводит к минимуму предыдущий шаг. Получается что наш алгоритм свелся к

Algorithm:

Let π^0 and $K \in \mathbb{N}$. For each $k \in \overline{0, K}$,do

- (Policy Evaluation)

  $v^{k+1} = R_{\pi^k} + P_{\pi^k} v^k$

  Define $q^k(s, a)$ by $v^k(s)$

- (Policy improvement) Greedy Policy Imporvement

  $$\pi^{k+1}(a|s) = \begin{cases} 1, \text{if } a \in \operatorname{argmax}_{a' \in \mathbb{A}} q^k(s, a') \\ 0, \text{otherwise} \end{cases}$$

Такой алгоритм является частным случаем Policy Iteration при L = 1 и всецело не противоречит теореме о сходимости алгоритма в которой главным условием сходимости это множество итераций K:

Theorem

$\pi^k \to \pi_*,\ k \to \infty$. Convergence rate $O(mn^2)$

Задание 3

Имплантация Vaule Iteration
В среднейм Value Iteration деат более низкую оценку чем Policy Iteration.
0.9884

Выводы

Мы изучили работу Policy и Value iterations в среде MDP при дискретных действиях.

Источники : https://chat.openai.com/chat