دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

گزارش تمرین شماره چهار
درس یادگیری تعاملی
پاییز ۱۴۰۰

نام و نام خانوادگی	محمد عظیم پور
شماره دانشجویی	۸۱۰۱۹۷۶۵۷

فهرست

چکیده

در این تمرین دریاچه ای یخ زده به عنوان محیط مدل شده است که Agent یادگیر باید از یک گوشه به گوشه دیگر آن برود. در هرکدام از بخش های تمرین، نوعی از الگوریتم های یادگیری را در این محیط امتحان می کنیم. در بخش اول الگوریتم های Model Based، در بخش دوم الگوریتم های Model Free و در بخش الگوریتم هایی که ترکیب حالت Model Based و Model Free هستند بررسی می شوند.

سوال ۱ – سوال پیاده‌سازی

هدف سوال

هدف از این سوال پیاده سازی الگوریتم Value Iteration به عنوان یک روش Model Based برای پیدا کردن ارزش هر Action در هر State است.

توضیح پیاده سازی

برای پیاده سازی مساله، کلاسی به نام Q1Agent تعریف شد که در آن توابع مورد نیاز برای به روز کردن مقادیر ارزش ها و الگوریتم Value Iteration (با توجه به کد نمونه ای که بارگذاری شده بود) پیاده سازی شده است. سپس یک نمونه از این کلاس گرفته شد و تابع مربوط به الگوریتم فراخوانی شد. همچنین توابعی برای چاپ مقادیر ارزش ها و سیاست در این کلاس وجود دارد.

نتایج

پس از اتمام اجرای الگوریتم با استفاده از توابع گفته شده در قسمت قبل مقادیر ارزش ها و Action بهینه در هر خانه چاپ شد که نتیجه آن به شکل زیر است:

```
-----------------------------
| 0.000 | 0.001 | 0.845 | 0.056 |
-----------------------------
| 0.193 | 0.001 | 0.323 | 0.180 |
-----------------------------
| 0.521 | 0.001 | 0.001 | 0.001 |
-----------------------------
| 0.506 | 0.698 | 0.870 | 0.000 |
-----------------------------

Values:
143.54648991757648      169.52307853573032      24.716922654999998      188.39727599269787
137.6516901359796       203.05980139320138      163.83591735448792      238.0034302947592
95.11481836277686       239.52207499365974      286.60357181827453      340.9410419358146
38.50214169819134       61.35819021134699       43.02212424308588       339.1585102072932

Policy:
2       1       2       1
2       1       1       1
2       2       2       1
3       3       2       1
```

سوال ۲ - سوال پیاده‌سازی

هدف سوال

هدف از این سوال پیاده سازی الگوریتم های Off-Policy MC، Q-Learning، SARSA و 2-Step Tree Backup به عنوان الگوریتم های Model Free برای یافتن ارزش هر Action در هر State است.

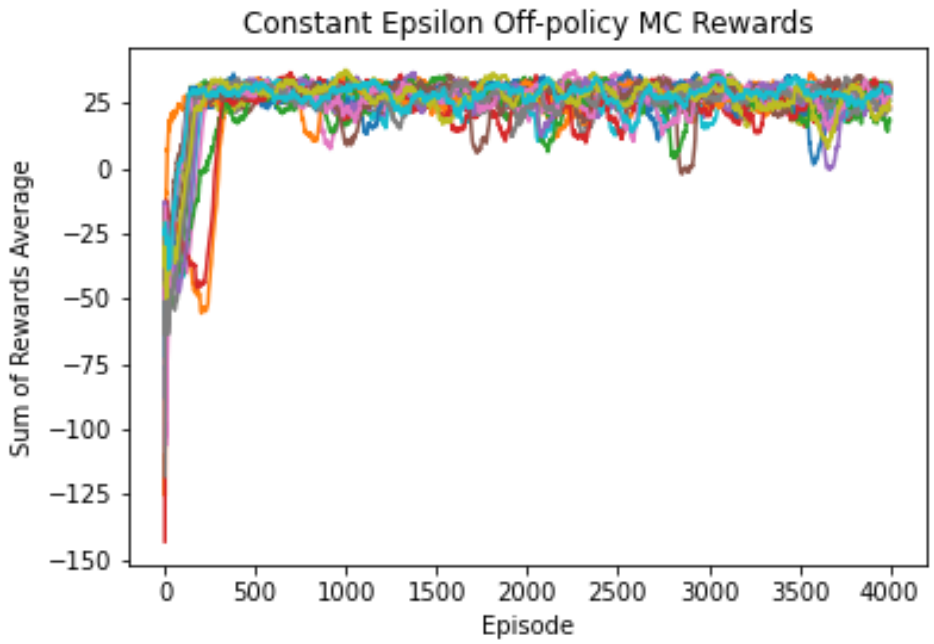
توضیح پیاده سازی

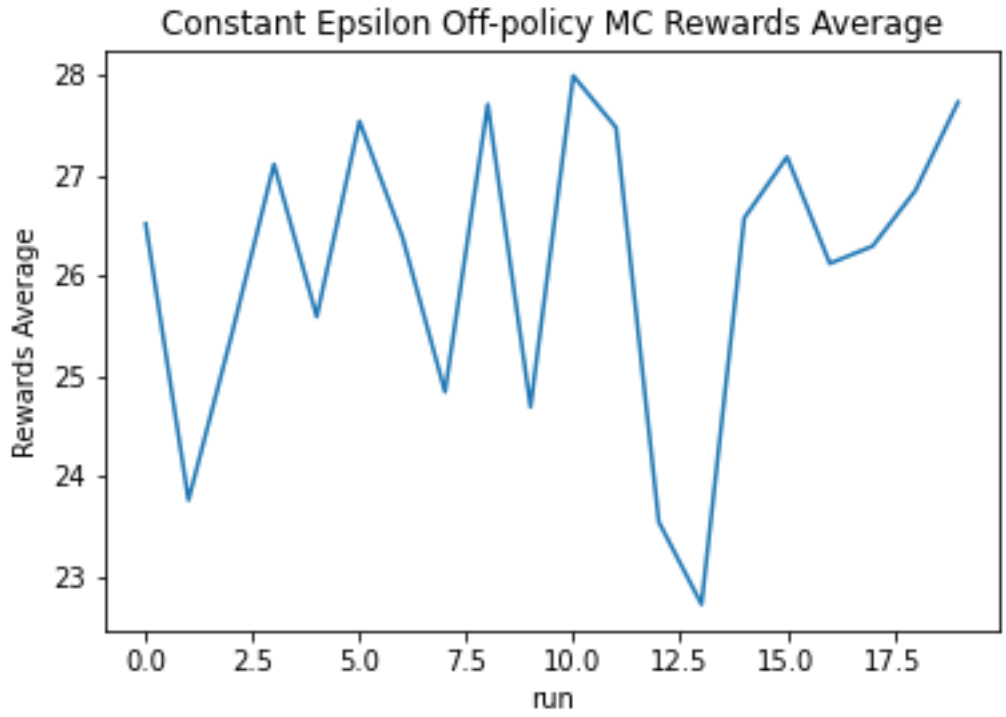
برای هرکدام از این ۴ الگوریتم یک کلاس Agent متناظر تعریف شده است که توابع و مقادیر مورد نیاز هر الگوریتم را در خود دارد. همه این ۴ کلاس، تابعی به نام run دارند که الگوریتم متناظرشان در این تابع پیاده سازی شده است و به ازای تعداد دفعات تکرار و تعداد اپیزودهای مورد نظر ما (که ورودی تابع هستند) اجرا می شود.

نتایج

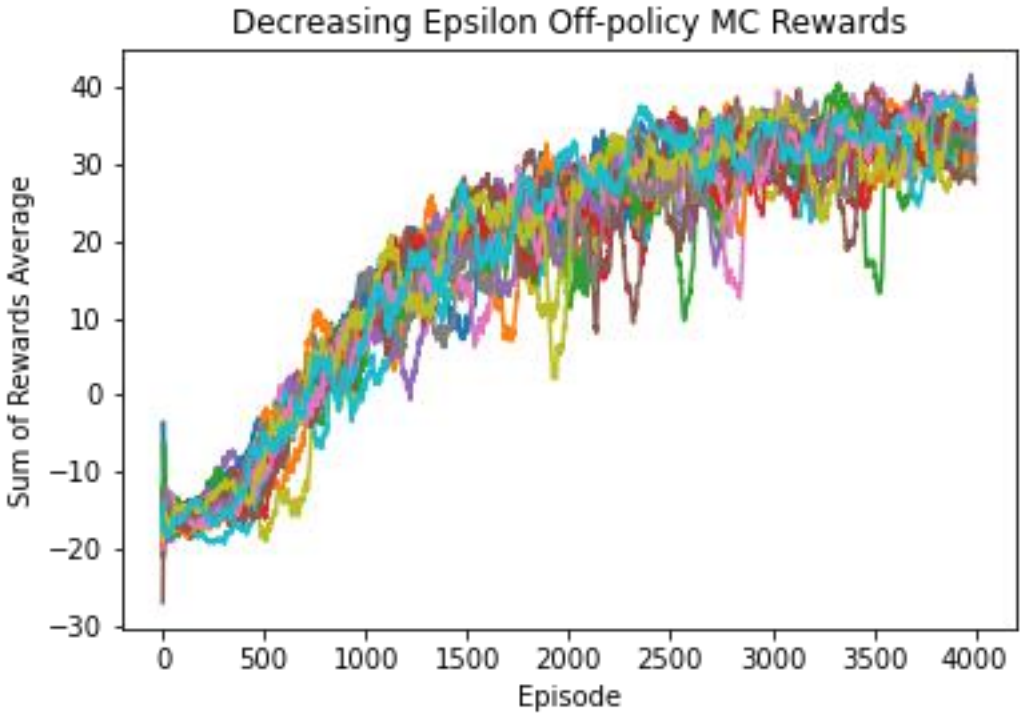
۱ – الگوریتم Off-Policy MC)

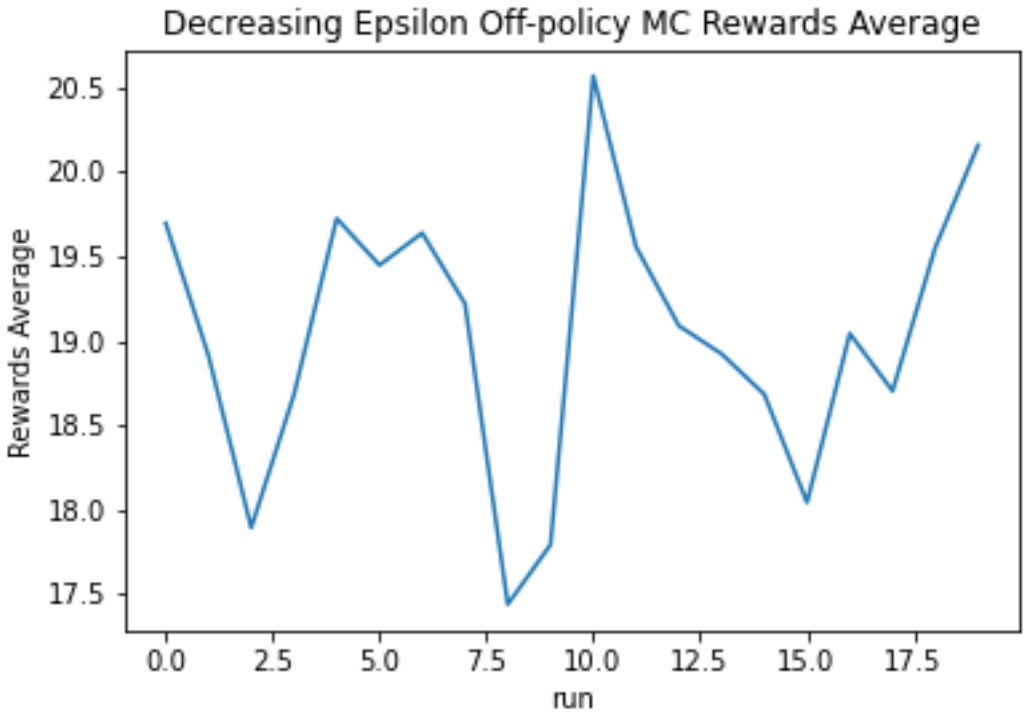
برای این الگوریتم، در حالت اپسیلون ثابت یادگیری انجام می شود ولی از آن جا که اپسیلون ثابت است پس از اتمام یادگیری نیز احتمال انتخاب Action غیر بهینه از بین نمی رود و در نتیجه نوساناتی در نمودار پاداش دریافتی آن مشاهده می شود.





در حالت اپسیلون کاهشی یادگیری با سرعت بسیار کمی انجام می شود ولی در نهایت Agent می تواند Actionهای بهینه تری را نسبت به حالت اپسیلون ثابت پیدا کند.

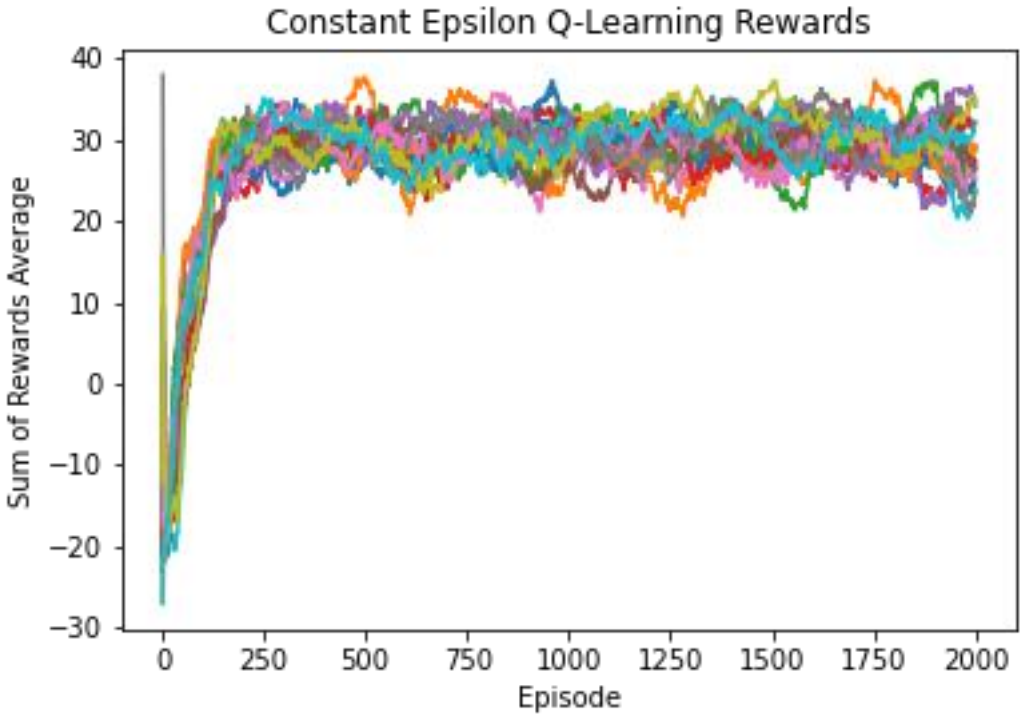


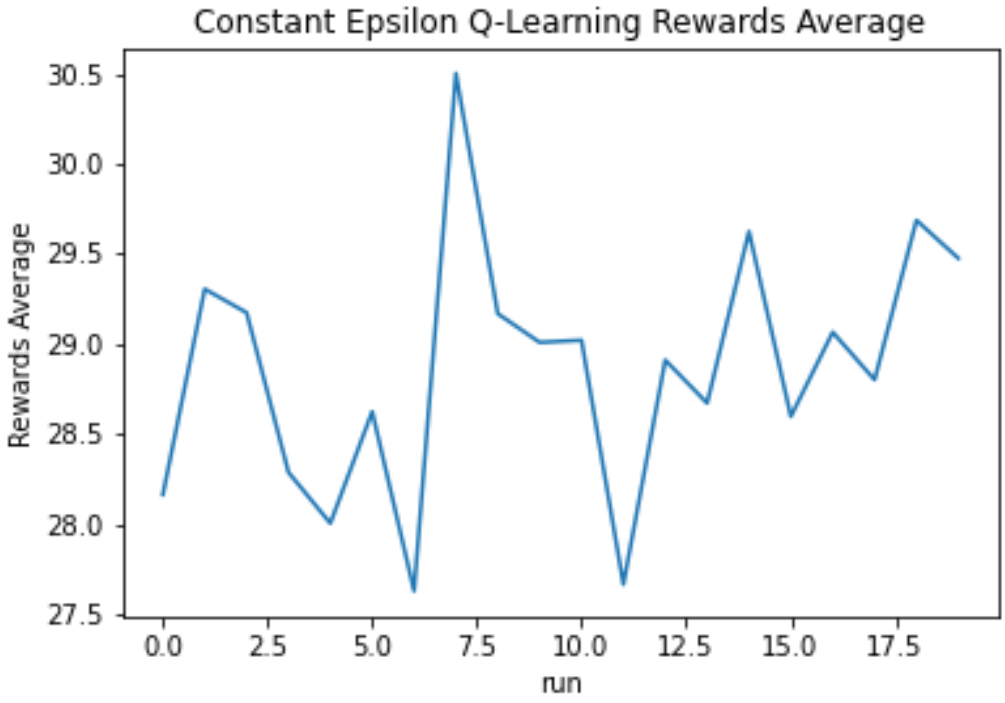


در نهایت مقایسه میزان پشیمانی در دو حالت به صورت زیر است:

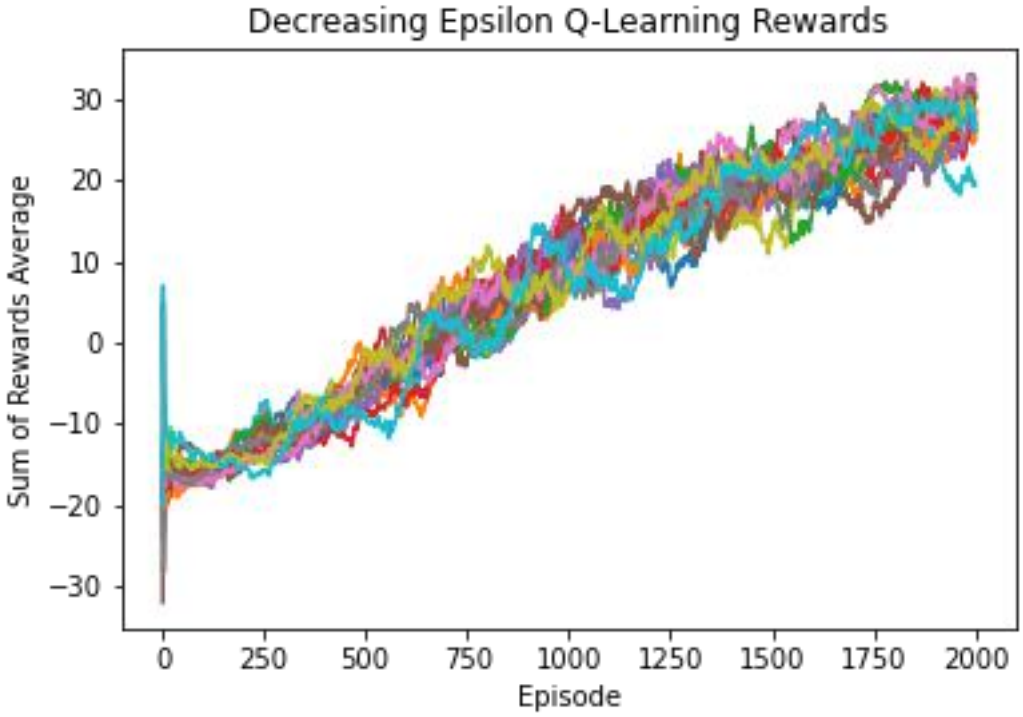
۲ – الگوریتم Q-Learning)

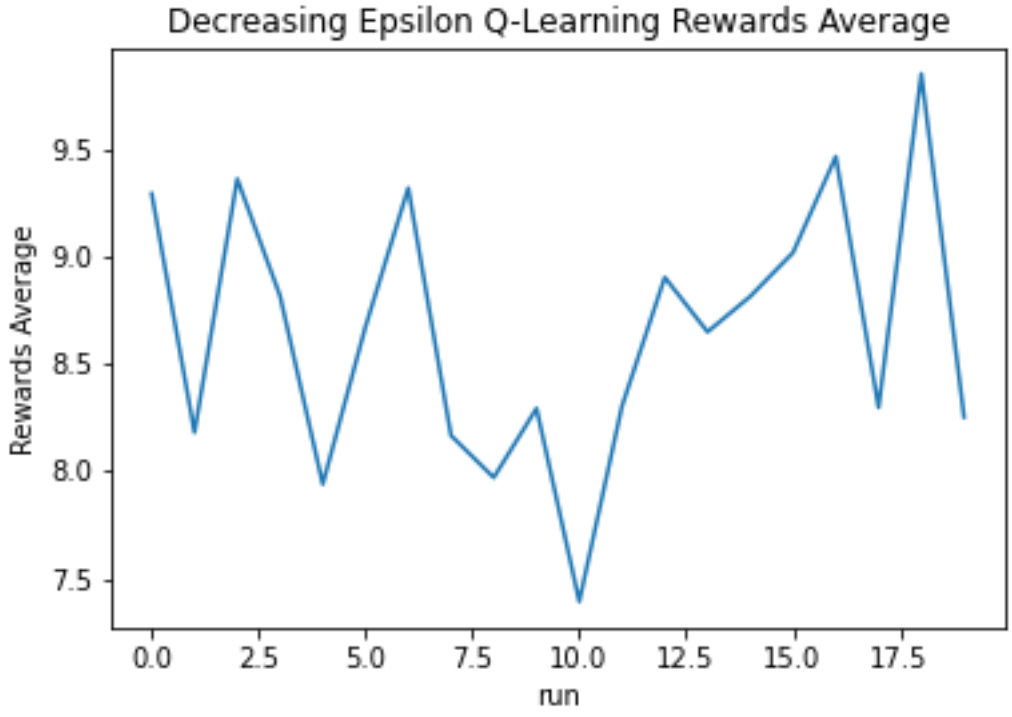
در این حالت نیز در حالت اپسیلون ثابت یادگیری انجام می شود ولی از آن جا که اپسیلون ثابت است پس از اتمام یادگیری نیز احتمال انتخاب Actionهای غیر بهینه از بین نمی رود و در نتیجه نوساناتی در نمودار پاداش دریافتی آن مشاهده می شود.





در حالت اپسیلون کاهشی هم مانند الگوریتم Off-Policy MC یادگیری با سرعت بسیار کمی انجام می شود ولی بر خلاف حالت قبل در نهایت سیاست Agent به Actionهای کم ارزش تری نسبت به حالت اپسیلون ثابت همگرا می شود.



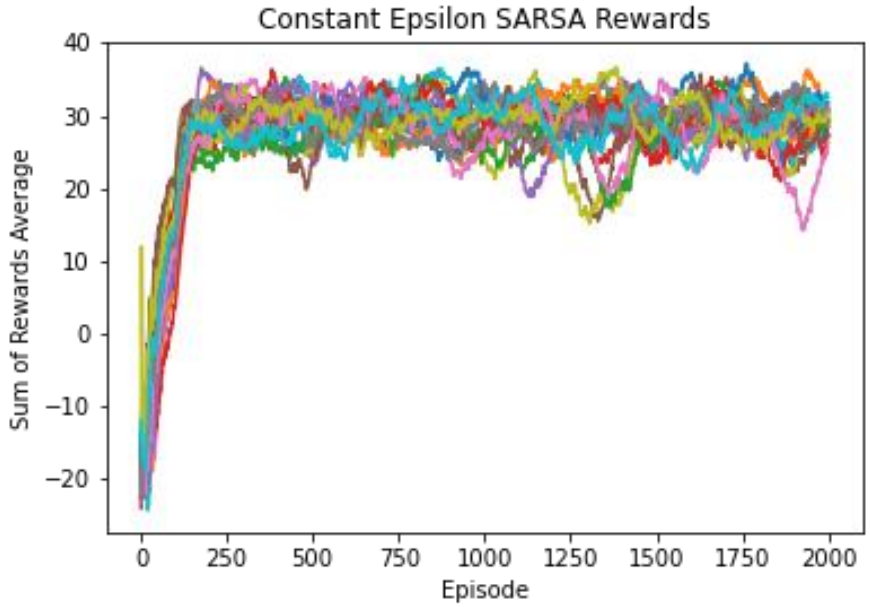


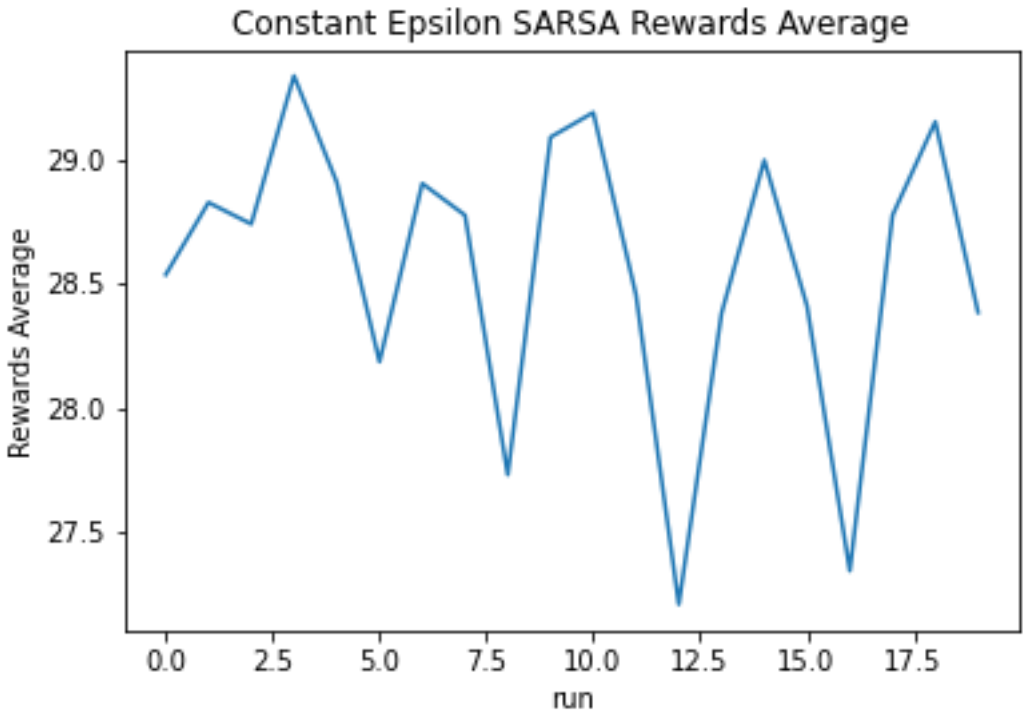
در نهایت مقایسه میزان پشیمانی در دو حالت به صورت زیر است:

۳ – الگوریتم SARSA و 2-Step Tree Backup)

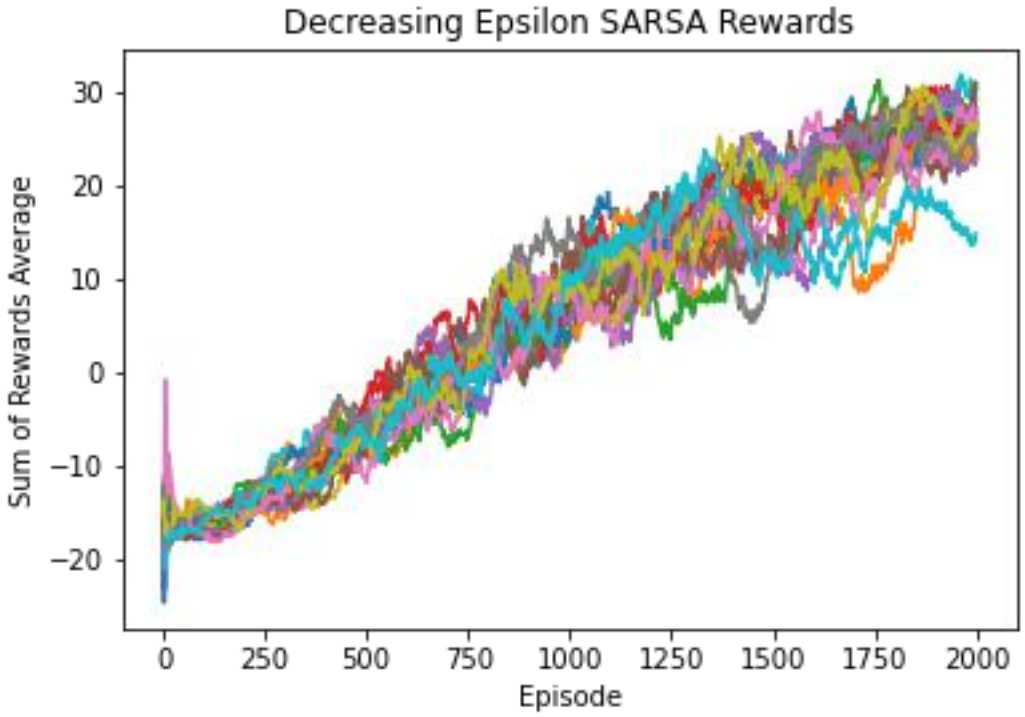
این دو الگوریتم نیز در دو حالت اپسیلون ثابت و کاهشی بررسی شده اند.

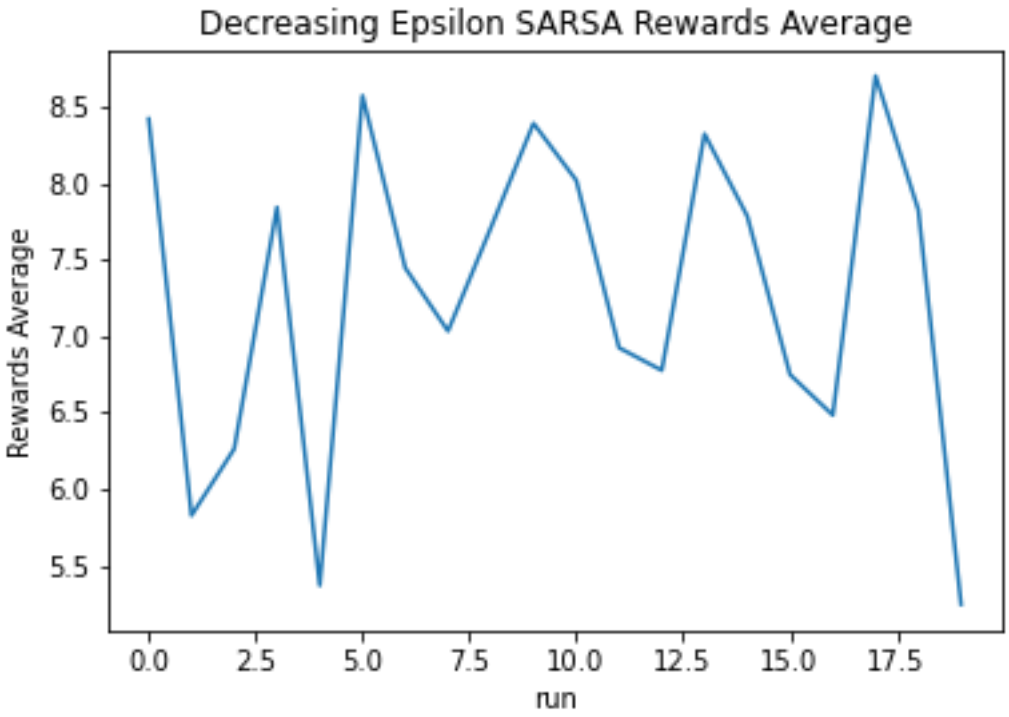
الگوریتم SARSA در حالت اپسیلون ثابت مانند الگوریتم های قبل از آن جا که اپسیلون ثابت است و پس از اتمام یادگیری نیز احتمال انتخاب Action غیر بهینه از بین نمی رود نوساناتی در نمودار پاداش دریافتی آن مشاهده می شود.



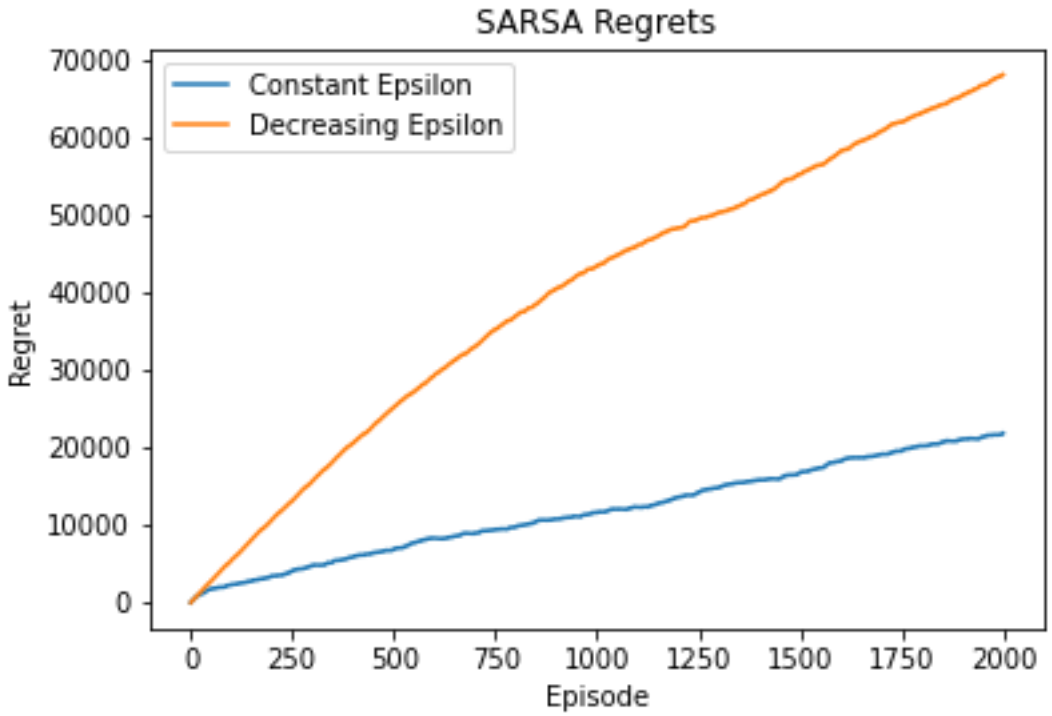


الگوریتم SARSA در حالت اپسیلون کاهشی مانند الگوریتم Q-Learning عمل می کند. به این صورت که در نهایت سیاست Agent به Actionهای کم ارزش تری نسبت به حالت اپسیلون ثابت همگرا می شود.

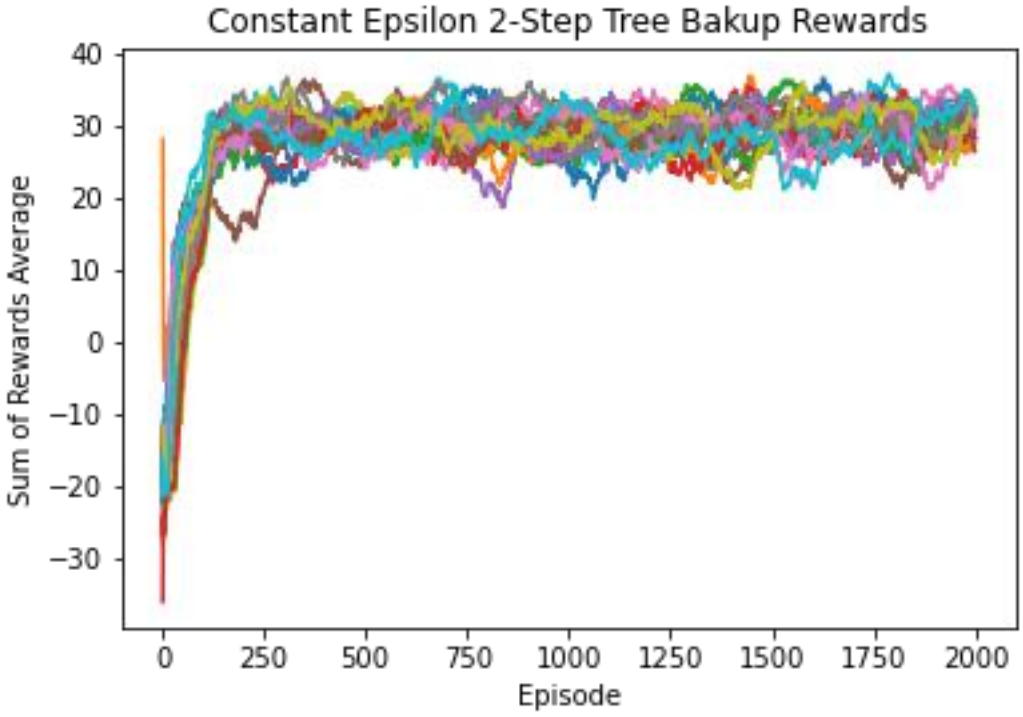


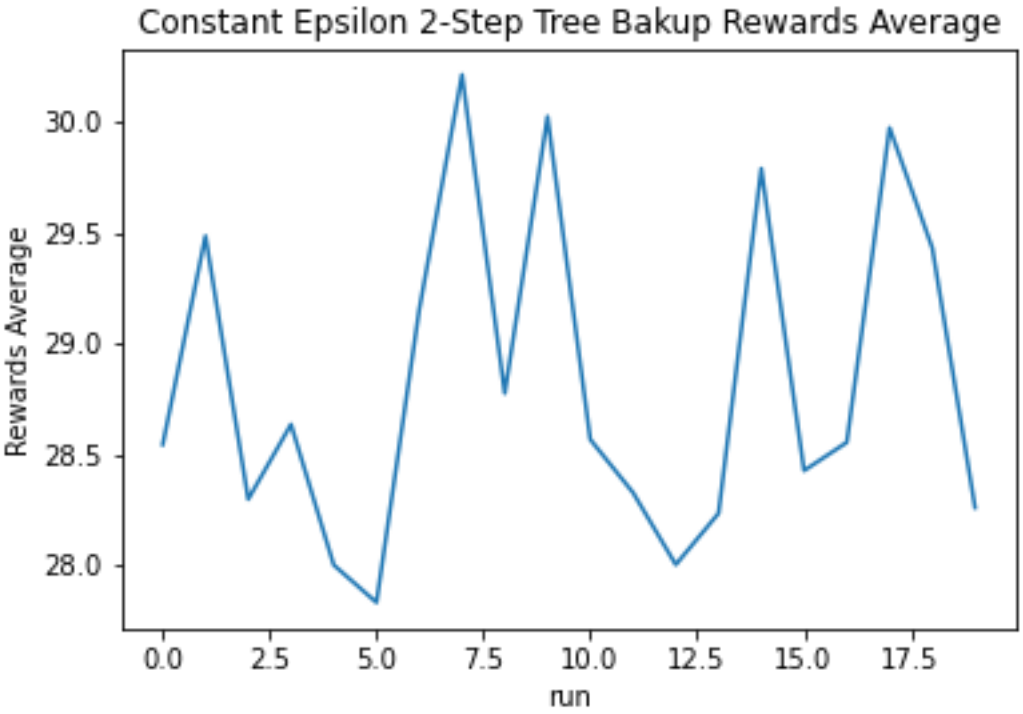


مقایسه پشیمانی دو حالت الگوریتم SARSA به صورت زیر است:

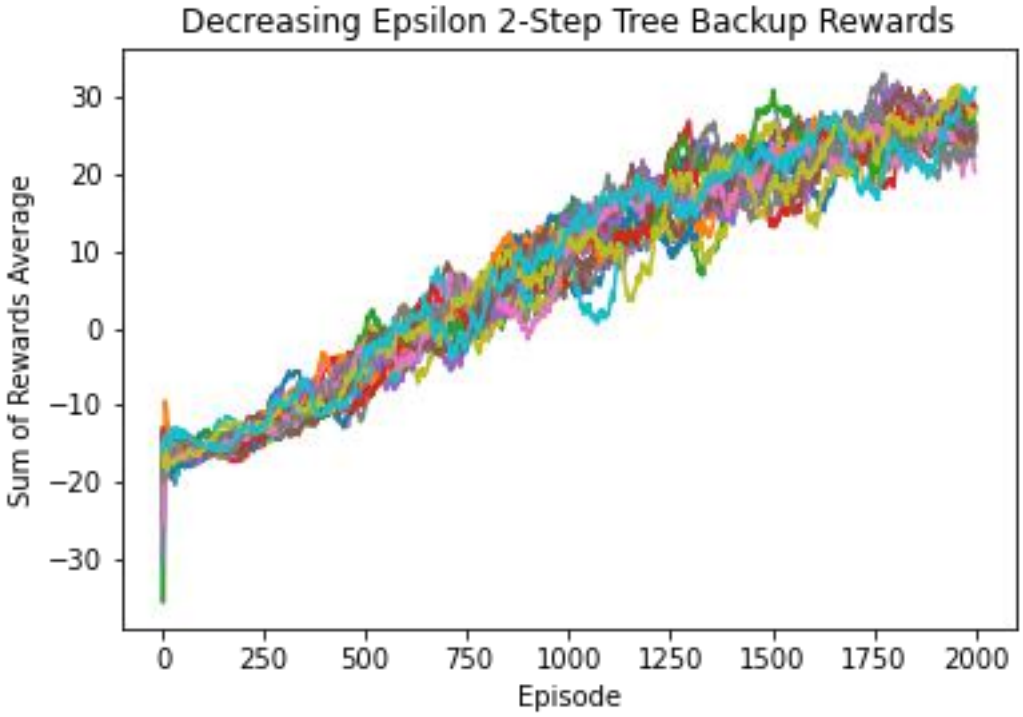


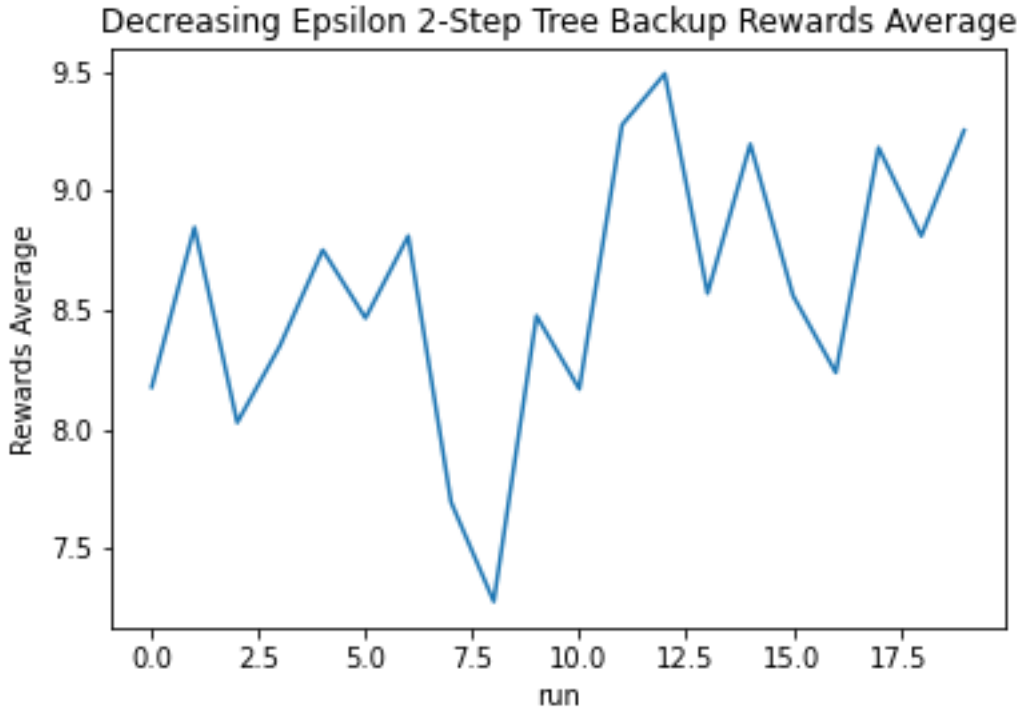
الگوریتم 2-Step Tree Backup در حالت اپسیلون ثابت در انتهای عملیات یادگیری دارای نوسانات کمتری به نسبت الگوریتم های قبلی است و به مقدار بهینه نیز همگرا می شود.



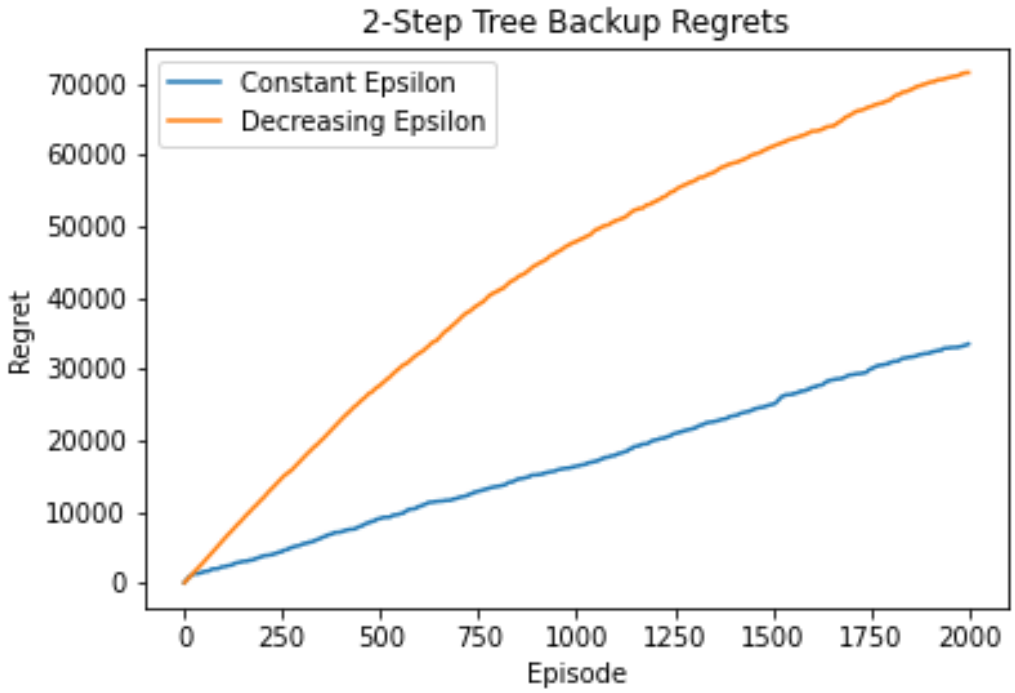


یادگیری در الگوریتم 2-Step Tree Backup نیز در حالت اپسیلون کاهشی با سرعت کمتری انجام می شود و در نتیجه در انتهای فرآیند یادگیری Agent میزان پاداش کمتری به نسبت حالت اپسیلون ثابت دریافت می کند.

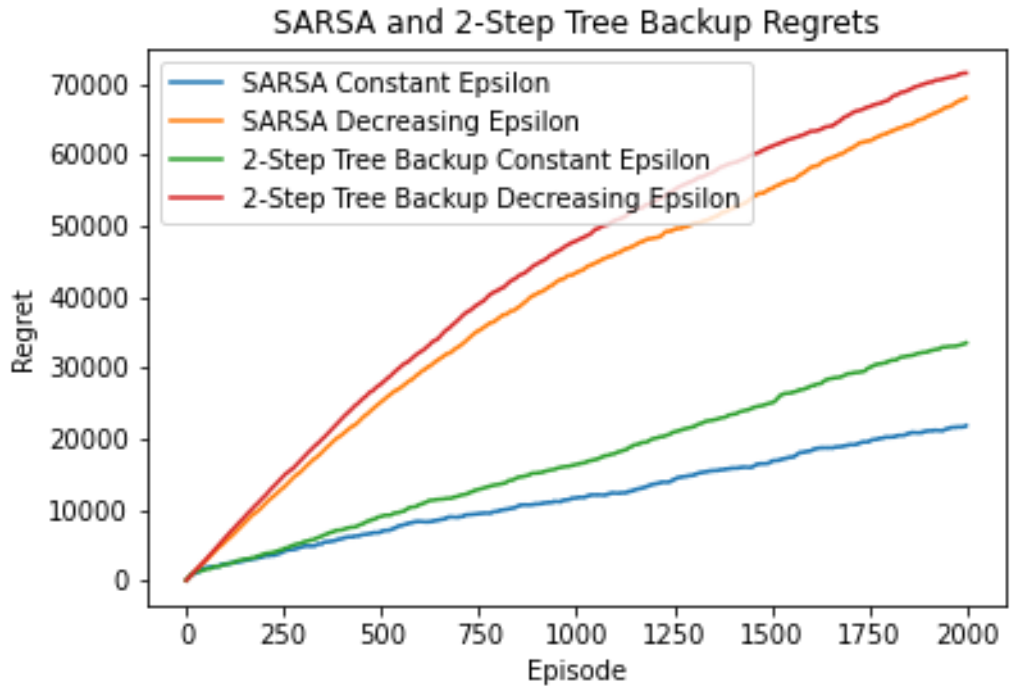




در نهایت مقایسه مقدار پشیمانی دو حالت الگوریتم به صورت زیر است:



حال اگر ۴ حالت بررسی شده در این بخش را در کنار هم بررسی کنیم به نمودار زیر می رسیم. از بین این ۴ حالت، بهترین حالت از نظر میزان پشیمانی مربوط به حالت اپسیبون ثابت الگوریتم SARSA و بدترین حالت مربوط به حالت اپسیلون کاهشی الگوریتم 2-Step Tree Backup است.



علت این که در این سوال الگوریتم های با اپسیلون کاهشی پشیمانی بیشتری نسبت به الگوریتم های با اپسیلون ثابت داشتند کم بودن تعداد اپیزودها و نرخ کم کاهش اپسیلون است. در صورتی که تعداد اپیزودها را زیادتر کنیم و یا این که اپسیلون را با نرخ بیشتری کاهش دهیم این مشکل برطرف می شود.

سوال ۳ - سوال پیاده‌سازی

هدف سوال

در این بخش فرض شده است که احتمال شکستن هر خانه دریاچه تابع زمان و متغیر است و باید از تلفیقی از روش های Model Based و Model Free برای آن استفاده کرد. الگوریتم Model Based مورد نظر الگوریتم Value Iteration و الگوریتم Model Free مورد نظر الگوریتم Q-Learning است.

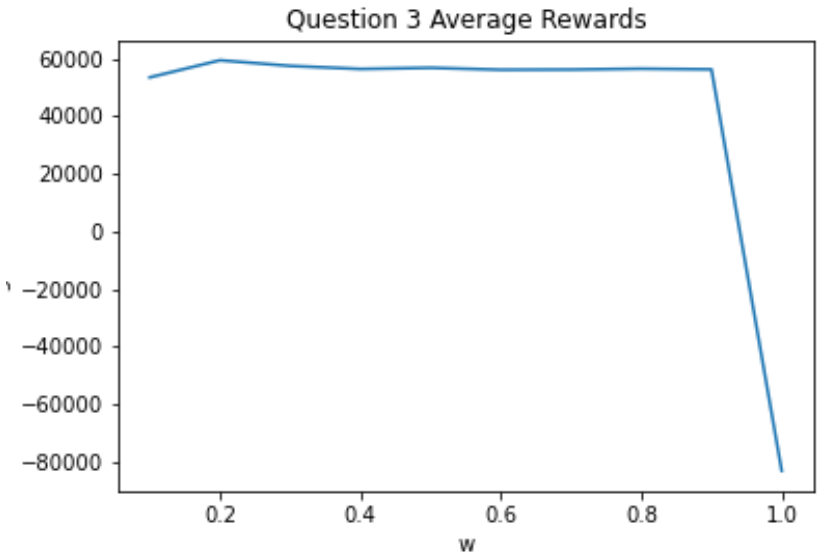
توضیح پیاده سازی

برای پیاده سازی مساله، مانند قبل یک Agent تعریف کردیم که در توابعی الگوریتم Value Iteration و توابع به روز رسانی مقادیر مطابق الگوریتم Q-Learning را در آن پیاده سازی کردیم. این کلاس هم یک تابع run دارد که به تعداد تکرار دلخواه و تعداد اپیزود دلخواه الگوریتم را اجرا می کند. این تابع پس از نمونه گیری از کلاس Agent (و یک محیط non-stationary) با w ها مختلف فراخوانی می شود.

نتایج

پس از اجرا و مشاهده نتیجه، می توان نتیجه گرفت که بهترین مقدار w برابر ۰,۲ است که Agent در آن بیشترین پاداش را دریافت می کند.

برای حالت ۰,۹، در این حالت مقدار Discount Factor مناسب است و Agent مسیر بهینه را پیدا می کند و تماما به صورت مورب به سمت گوشه بالای سمت چپ جدول حرکت می کند. در نتیجه با ۱۴ حرکت به جایزه می رسد. (پاداش = ۹۹۹,۸۶)



از آن جا که داده آخر را می توان داده پرت فرض کرد (و زمانبر بودن اجرای دوباره الگوریتم)، نمودار حاصل از حذف این داده را نیز رسم کردیم که به شکل زیر است. این نمودار بهینه بودن نقطه w=0.2 را بهتر نمایان می کند.