بخش اول

یک شرکت داریم که دارای سه روش برای آموزش به دانش‌آموزان است و میخواهد ببیند که کدام یک از این روش ها، تاثیر بهتری بر روی یادگیری دانش‌آموزان دارد. پس اگر بخواهیم این مسئله را بصورت یک Multi-Armed-Bandit مدلسازی کنیم، بازوهایی که برای این مسئله تعریف می‌شوند، بصورت زیر هستند:

A = روش آموزشی اول
B = روش آموزشی دوم
C = روش آموزشی سوم

در مورد پاداش هر یک از روشهای آموزشی (بازوها) هم باید گفت که طبق صورت سوال می‌توان فرض کرد که در انتهای هر هفته یک آزمون مشترک از دانش آموزان گرفته می‌شود و میزان پاداش هرروش بصورت میانگین نمره‌ی دانش‌آموزانی که با آن روش خاص آموزش دیده اند، محاسبه می‌شود. با توجه به توضیحات داده شده در سوال دوم (کدها)، می‌توان گفت که پاداش هر کدام از این روش ها بصورت تصادفی از یک توزیع نرمال که بین 0 و 100 قرار دارد، می‌آید و پارامترهای این توزیع برای هر کدام از این روش ها برابرند با :

A = Normal (μ=40, σ=40)
B = Normal (μ=60, σ=10)
C = Normal (μ=50, σ=20)

بخش دوم

در این بخش با توجه به مدلسازی ای که در بخش قبل انجام شد، به این صورت عمل می‌کنیم که در هر هفته از یک برنامه آموزشی برای آموزش هر دانش آموز استفاده می‌کنیم. برای انتخاب برنامه آموزشی در هر مرحله، طبق صورت سوال، از روش مبتنی بر گرادیان استفاده می‌کنیم که به اینصورت عمل می‌کند که ابتدا مقدار پارامتر ترجیح را برای هر کدام از روش ها برابر با صفر در نظر می‌گیریم و در هر مرحله آن را بر اساس پاداشی که دریافت کردیم، بروز می‌کنیم و انتخاب سیاست هر در هر مرحله بر اساس روش softmax انجام می‌شود. در زیر می‌توانید جزئیات الگوریتم استفاده شده برای حل این مسئله را مشاهده کنید :

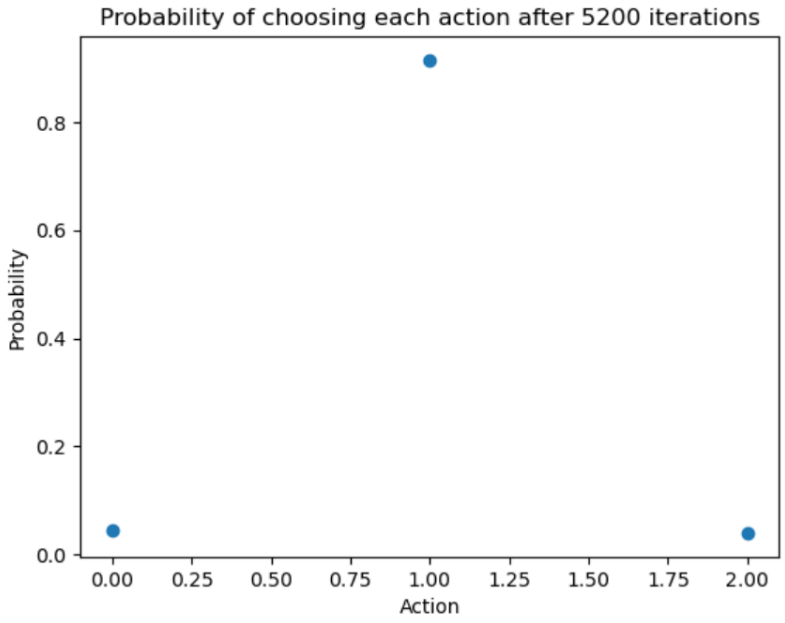
احتمال انتخاب هر عمل :

$$\Pr\{A_t = a\} = \frac{e^{H_t(a))}}{\sum_{b=1}^{n} e^{H_t(b)}} = \pi_t(a),$$

نحوه بروز رسانی پارامتر H (ترجیح) برای هر عمل

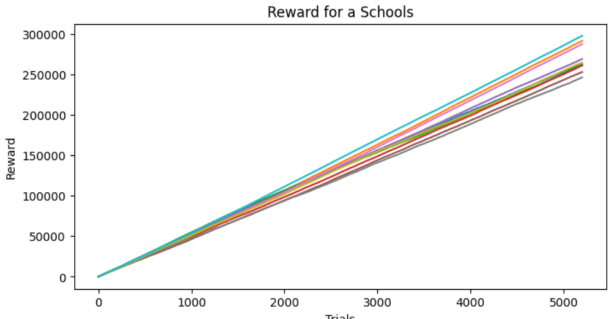
$$H_{t+1}(A_t) \doteq H_t(A_t) + \alpha\big(R_t - \bar{R}_t\big)\big(1 - \pi_t(A_t)\big), \qquad \text{and}$$
$$H_{t+1}(a) \doteq H_t(a) - \alpha\big(R_t - \bar{R}_t\big)\pi_t(a), \qquad \text{for all } a \neq A_t$$

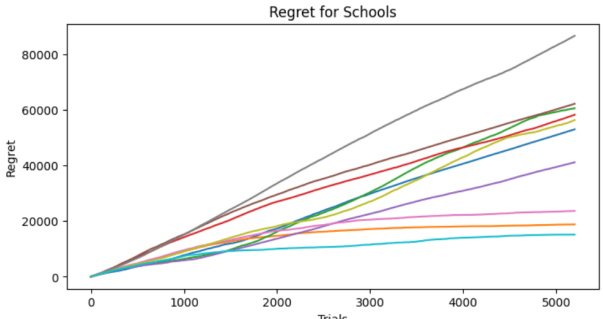
در شکلی که در ادامه آمده، میتوانید ببینید که در پایان افق زمانی، احتمال انتخاب روش دوم تقریبا نزدیک به یک و احتمال انتخاب روش اول و سوم، نزدیک به صفر است و این همان چیزی بود که ما انتظار داشتیم، چرا که روش دوم دارای میانگین بالاتر و انحراف معیار کمتر نسبت به روش های دیگر است.

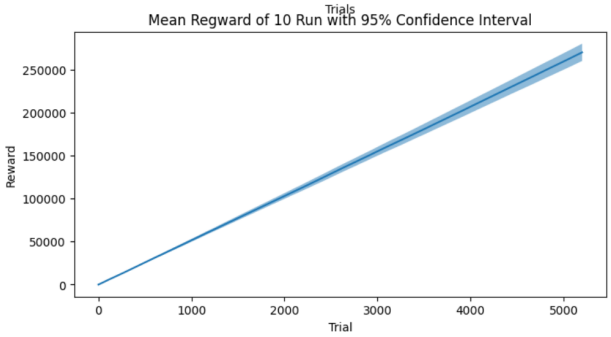


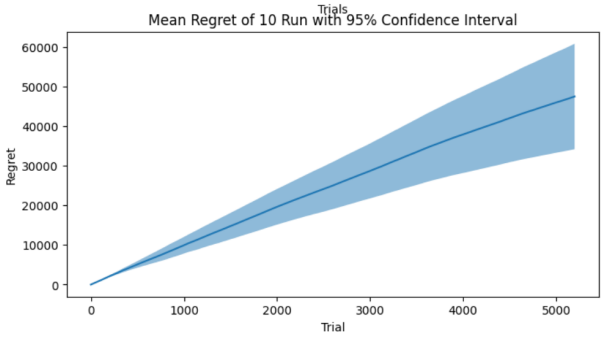
در ادامه نمودارهای مقدار پاداش و پشیمانی را به ازای نرخ های یادگیری ۰.۰۰۱ ، ۰.۰۱ و ۰.۱ ، برای ده مدرسه متفاوت آمده است و بعد از آن هم نمودار پاداش و پشیمانی با یک بازه اطمینان ۹۵٪ آمده که شامل میانگین این پارامترها و ابری در اطراف این مقدار است که بیان می‌کند به احتمال ۹۵٪ مقدار این پارامتر ها در بازه های نشان داده شده قرار میگیرد.

برای Learning rate = 0.001 :

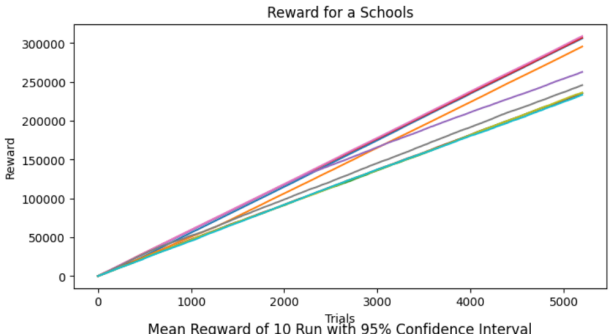


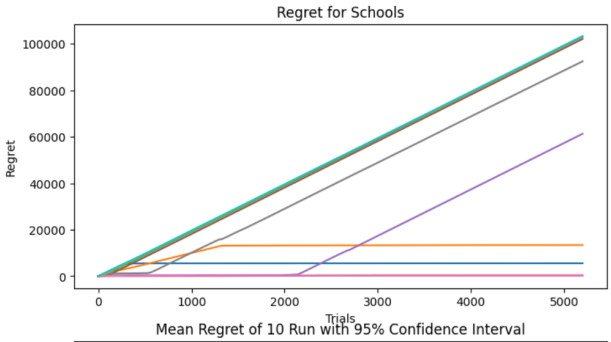


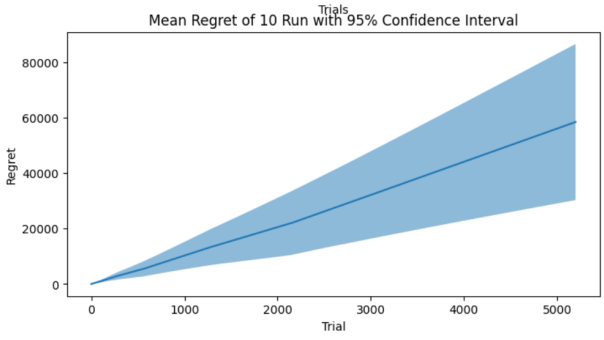




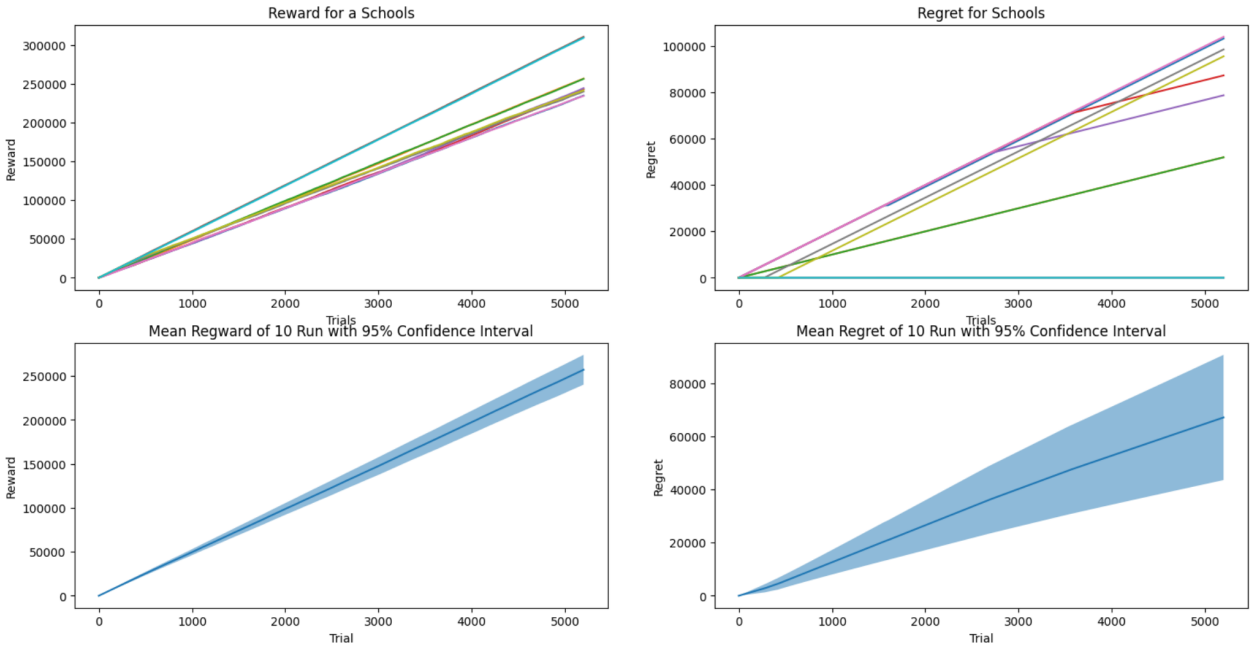
برای Learning rate = 0.01 :







برای Learning rate = 0.1 :



بخش سوم

قبل از شروع به توضیح دادن راجع به این بخش، نیاز است به این مورد اشاره کنم که با توجه به این مشکل که بعضا promt های داده شده به Chat-GPT، ارسال نمی‌شد، که در بخش bug-report هم به آن اشاره کردیم، ما برای انجام این بخش، از سایر وبسایت هایی که api از Chat-GPT در اختیار ما می‌گذاشتند مثل سایت poe.com هم استفاده کردیم.

در اینجا ما براساس یافته های خود از مقاله Planning Problem Steps with Reinforcement Learning Models پیش رفتیم. اگر بخواهیم به صورت خلاصه، به روشی که در این مقاله برای حل مسائل یادگیری تقویتی اشاره شده است بپردازیم، اصولا این روش شامل گام های زیر می‌شود:

1. مسئله را به عنوان یک توصیف متنی برای ChatGPT فرموله می‌کنیم:

"سه برنامه آموزشی به نام‌های m1، m2 و m3 وجود دارد. پاداش برنامه m1 از توزیع نرمال با میانگین 40 و انحراف معیار 40 پیروی می‌کند. پاداش برنامه m2 از توزیع نرمال با میانگین 60 و انحراف معیار 10 پیروی می‌کند. پاداش برنامه m3 از توزیع نرمال با میانگین 50 و انحراف معیار 20 پیروی می‌کند. برای 100 دانش‌آموز به ازای هر هفته در طی 52 هفته، یکی از این برنامه‌ها را پیشنهاد دهید تا جمع پاداش کلی را بیشینه کنید. روشی بر اساس الگوریتم های یادگیری تقویتی ارائه داده و جزئیات استراتژی انتخاب را هفته به هفته توضیح دهید."

2. ChatGPT را با استفاده از مجموعه داده‌ای از مسائل multi-armed bandit و راه حل‌ها بهینه‌سازی کنید.

3. ChatGPT به عنوان مدل آموزش دیده به مسئله مطرح شده پاسخ دهید و از آن بخواهید استراتژی انتخاب هفتگی را برای بیشینه کردن پاداش کل مورد انتظار ارائه دهد.

به عنوان مثال:

"بر اساس توصیف مسئله، در هفته ۱ به چه برنامه‌ای پیشنهاد می‌دهید؟ هفته ۲؟ ..."

4. استراتژی ChatGPT را تجزیه و تحلیل کنید و با شبیه‌سازی انتخاب‌ها، پاداش کل مورد انتظار را اعتبارسنجی کنید. سوالات فرعی را بپرسید تا پیشنهادات غیربهینه را بهبود ببخشید.

به عنوان مثال:

"چرا در هفته ۳ برنامه m2 را پیشنهاد دادید؟"

"پاداش کل مورد انتظار با برنامه شما برابر با X است. آیا می‌توانید یک برنامه بهبود یافته ارائه دهید تا پاداش کل را افزایش دهید؟"

5. تا زمان رسیدن به سیاست بهینه و یا نزدیک به بهینه، سیاست را آپدیت کنید.

6. استراتژی نهایی Chat-GPT را به صورت یک نگاشت از فضای حالت به عمل را برای اجرا به صورت رسمی مشخص کنید.

بعد از اینکه مسئله برای ChatGPT بیان شد، توانست سوال را به خوبی بفهمد متوجه شود و الگوریتمی که برای حل این روش ارائه شد، الگوریتم Upper Confidence Bound بود. توضیحی که از الگوریتم توسط Chat-GPT ارائه شد، توضیح دقیق و مبتنی بر شهودی بود که ما از این روش سابقا داشتیم یعنی میدانست که پارامترهای این الگوریتم چیست، چه پارامترهایی باید در طول یادگیری بهینه شوند و چه مواردی نیازی به بروزرسانی ندارند.
همچنین می دانست چگونه باید در هر مرحله عملی را انتخاب کند که بر اساس الگوریتم UCB باشد.
سپس سعی کردیم در تعدادی محدودی از مراحل، عملکرد Chat-GPT را در انتخاب عملی که باید بر اساس الگوریتم UCB باشد و طوری که این عامل پارامتر ها را بروز میکند را بررسی کنیم. این اتفاق تا حد خوبی به درستی صورت می‌گرفت ولی یک مشکل بزرگ که در نحوه عملکرد این عامل بود این بود که در ابتدا مقدار پارامتر ubc را برای همه برنامه ها برابر با صفر در نظر می‌گرفت و اتفاق بدی که در اینجا می‌افتاد، این بود که اگر در مرحله اول، یک عمل تصادفی را انتخاب کنیم، بعد از دریافت پاداش و بروزرسانی مقدار ucb برای آن عمل بر اساس پاداش بدست آمده، می‌توان دید که مقدار حاصل، بزرگتر از صفر خواهد شد و در نتیجه از آنجایی که مقدار ubc برای سایر عمل هایی که تاکنون انتخاب نشده برابر با صفر است، پس همواره عملی که در ابتدا انتخاب شده انتخاب می‌شود و هرگز عمل های دیگر انتخاب نمی‌شوند. و در اینجا کاری که ما کردیم این بود که این موضوع را برای عامل که در اینجا همان Chat-GPT باشد، مشخص کردیم که در ابتدا پارامتر ucb برای هر عمل برابر با مقدار ∞ است و بعد از اینکه یکبار یک عمل خاص انتخاب شد، مقدار ucb اش را به همان طریقی که از قبل می‌دانست، آپدیت میکنیم و با این کار باعث می‌شویم تا از آن اتفاق که فقط یک عمل انتخاب شود، جلوگیری کند.
مشکل اصلی عامل در یادگیری مدل، همین مورد بود و به جز این مورد، چند مورد جزئی از قبیل اینکه نمی‌تواست پارامتر های مدل را به خوبی به خاطر بسپارد بود که این مشکل هم بعد از اینکه چندین بار، تکرار های مختلف را به همراه عامل پیش رفتیم، به نتیجه ی خوبی منجر شد.

بخش چهارم

در مسأله N-armed bandit، اطلاعات اضافی برای بهینه‌سازی تعادل بین اکتشاف و بهره‌برداری بسیار مهم است. دانش اولیه درباره گزینه‌ها، از جمله عملکرد تاریخی و گزینه‌های مطمئن‌تر، امکان تخصیص منابع با آگاهی بیشتر فراهم می‌کند. درک توزیع پاداش برای هر گزینه در توسعه الگوریتم‌های bandit که به صورت استراتژیک بین اکتشاف و بهره‌برداری عمل کنند، کمک می‌کند. درک همبستگی بین گزینه‌ها و اطلاعات زمینه‌ای به تصمیم‌گیری شخصی‌سازی کمک می‌کنند. این دانش به فراتر از چارچوب اولیه N-armed bandit می‌رود تا به مسائل محیط‌های پویا، یادگیری انتقالی، و تخصص انسانی بپردازد و کارایی در شناسایی گزینه‌های پرثمر را افزایش دهد. مزایا شامل سازگاری با تغییرات محیط، استراتژی‌های دقیق با اطلاعات هزینه، ترویج یادگیری همکارانه، و سهیم شدن در تصمیم‌گیری شخصی‌سازی هستند. به طور کلی، یکپارچگی اطلاعات اضافی در مسأله N-armed bandit کارایی تصمیم‌گیری را ارتقا می‌دهد و احتمال شناسایی و بهره‌برداری از گزینه‌های پرثمر را افزایش می‌دهد.
اکنون به حل مساله میپردازیم. ابتدا صورت مساله و الگوریتم حل را برای مدل زبانی (chatGPT) توضیح میدهیم. سپس چندین مثال بیان میکنیم تا بداند دقیقا باید از چه جهتی به ما در تصمیم گیری کمک کند به این کار اصطلاحا (few shot) میگویند که یکی از راه های متداول برای پرامپ کردن مدل ها زبانی است. سپس سوال مد نظر را از مدل زبانی می‌پرسیم. در نهایت با کمی تحلیل و فکر کردن میتوان به پاسخ سوال با استفاده از مدل زبانی دست یافت.
در واقع اطلاعات اضافی درباره عملکرد دیگر مدارس و برنامه‌های آموزشی ارائه شده توسط آن‌ها، اطلاعات ارزشمندی را برای مدیران یک مدرسه خاص فراهم می‌کند. این اطلاعات می‌توانند در حوزه یک مسأله باندیت چندبازویی بهبود تصمیم‌گیری را فراهم کنند و به احتمال زیاد به مدیران اجازه دهند تا به بهترین نحو برنامه مطلوب را به دانش‌آموزان خود ارائه دهند. اینجا چگونگی این اطلاعات اضافی می‌تواند مفید باشد:

1. یادگیری از مدارس دیگر:
- مدیران می‌توانند داده‌های عملکرد تاریخی مدارس دیگر و برنامه‌های ارائه شده توسط آن‌ها را مشاهده کنند.
- این اطلاعات به درک این کمک می‌کند که کدام برنامه‌ها در مدارس دیگر موفق‌تر بوده‌اند.

2. استفاده از برنامه‌های موفق:
- مدیران می‌توانند از این اطلاعات بهره‌برداری کنند با ترجیح برنامه‌هایی که به طور مداوم در مدارس دیگر عملکرد خوبی داشته‌اند.
- اگر روش‌های آموزشی خاصی در مدارس دیگر نمره میانگین بالاتری داشته‌اند، مدیران ممکن است این روش‌ها را با اولویت بیشتری مورد استفاده قرار دهند تا احتمال نتایج مثبت افزایش یابد همچنین با توجه به صورت مساله چون از میزان موفق بودن برنامه ها اطلاعی در دسترس نیست باید این مساله را در نظر داشت که بیش از اندازه به داده هایی که قطعیت به آن ها موجود ندارد پرداخته نشود البته این در مراحلی است که با فضا مساله ناآشنا هستیم پس از کسب از اطلاعات از محیط میتوان نرخ استفاده از برنامه های موفق را افزایش داد.

3. تسریع فاز بهره‌برداری:
- با دانستن در مورد موفقیت برنامه‌ها در مدارس دیگر، مدیران ممکن است در بهره‌برداری از بهترین روش‌ها در مراحل ابتدایی فرآیند، با اطمینان بیشتری عمل کنند.
- این احتمالاً می‌تواند فرآیند شناسایی و اجرای برنامه‌های مطلوب برای دانش‌آموزان را تسریع بخشد.

4. کاهش اکتشاف در برخی موارد:

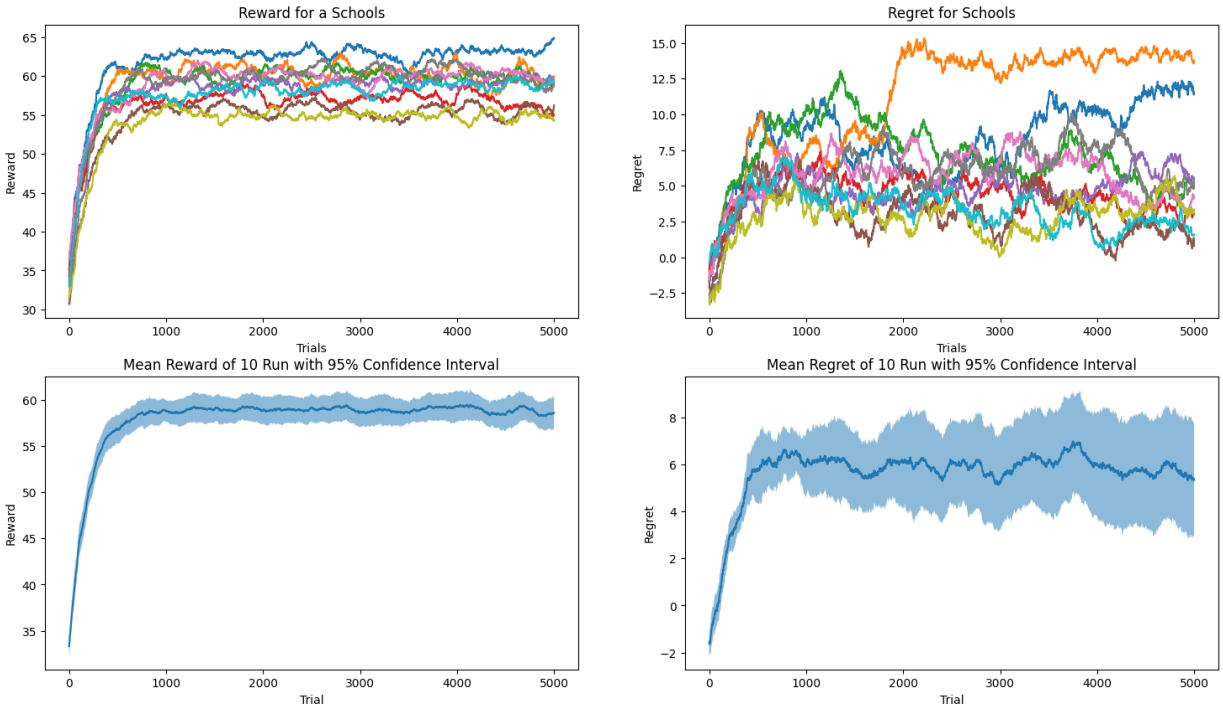
- اگر مدیران مشاهده کنند که برخی از برنامه‌ها به طور مداوم عملکرد ضعیفی داشته‌اند، ممکن است تصمیم بگیرند که اکتشاف این برنامه‌ها را کاهش دهند.
- این می‌تواند با صرفه‌جویی در زمان و منابع باعث اجتناب از برنامه‌هایی شود که در زمینه‌های مشابه کمتر کارآمدی نشان داده‌اند.

5. تنظیم استراتژی بر اساس درک کلی:
- مدیران می‌توانند استراتژی الگوریتم باندیت خود را بر اساس درک کلی که از عملکرد برنامه‌ها در چندین مدرسه به‌دست می‌آید، تنظیم کنند.
- به عنوان مثال، اگر یک برنامه خاص به طور مداوم از سایرین بهتر عمل کرده باشد، الگوریتم می‌تواند تنظیم شود تا منابع بیشتری را به آن برنامه اختصاص دهد.

6. یادگیری و تنظیم مداوم:
- سیستم می‌تواند طراحی شود تا به‌طور مداوم از عملکرد مدارس دیگر یاد بگیرد و مدل باندیت خود را به‌طور پویا تنظیم کند.
- هرچه بیشتر داده از مدارس دیگر جمع‌آوری شود، مدیران می‌توانند استراتژی خود را به‌منظور بهبود کارایی کلی برنامه‌های آموزشی تنظیم کنند.

بخش پنجم

تفاوت این روش با یک مسئله multi-arm bandit ساده تلاش بر شخصی‌سازی بیش‌تر برای agent ها به کمک اطلاعاتی است که از آن‌ها داریم. در این الگوریتم قبل از انتخاب تصمیم به contex که در این‌جا دانش‌آموز استنگاه می‌کنیم و به کمک اطلاعاتی که از آن داریم action بهینه را انتخاب می‌کنیم و در نهایت هدف همچنان پیدا کردن سیاستی است که reward را بیشینه کند و البته regret کم‌تری هم در طول یادگیری ایجاد کند. به کمک این مفهوم در واقع با شخصی‌سازی برای agent ها سعی می‌شود برای هر مدل agent که در این‌جا سه نوع داریم، مسئله را بهتر حل کنیم. این الگوریتم برای مسائلی مانند advertisment در سایت‌ها بسیار پرکاربرد و مناسب است.
برای پیاده‌سازی این روش، با توجه به این‌که در کدهای داده شده rewardهای دانش‌آموزهای مربوط به گروه‌های مختلف از توزیع‌های متفاوتی می‌آید لذا برای حدس مدرسه در مورد نوع دانش‌آموز از rewardهایی که می‌آید استفاده کردیم. پس از حدس زده شده، می‌سنجیم که آیا حدس درست است یا نه. اگر درست بود سیاست را به روز می‌کنیم و اگر نبود این کار را نمی‌کنیم و صرفا پاداش را حساب می‌کنیم.
نمودارهای پاداش و پشیمانی در شکل زیر آمده است.
مشخص است که نمودارها نویزی است که این هم منطقی است زیرا ترتیب دانش‌آموزان از گروه‌های مختلف پشت سر هم وارد می‌شود و action بهینه برای گروهی انتخاب می‌شود که تخمین زده‌ایم لذا این پاداش‌ها و پشیمانی‌ها می‌تواند بالا و پایین شود ولی اگر به نمو نمودارها نگاه کنیم می‌بینیم که روند منطقی‌ای را طی می‌کند.
در ابتدای الگوریتم، با خروجی‌ای که می‌گیریم به تخمین گروه مربوط به هر دانش‌آموز می‌پردازیم لذا در ابتدا پاداش‌ها کم است ولی با گذشت چند تکرار مدارس یاد می‌گیرند که هر دانش‌آموز از چه نوعی است لذا تقریبا همیشه تصمیم آن‌ها درست است لذا پاداش همواره بالای ۶۰ است. در این‌جا ما دقیقا از الگوریتم سوال دوم یعنی gradient bandit استفاده کردم و contextual bandit را روی آن پیاده کردیم.
در مورد روش‌هایی که می‌توان به جای این روش استفاده کرد، به کمک chat gpt روش‌هایی که در این سناریو به جای contextual bandit می‌توان استفاده کرد را جست و جو کردیم.
در ابتدا باید توجه داشت انتخاب روش بسته به نوع تعریف مسئله است. برای مثال اگر مسئله به گونه‌ای تعریف شده بود که ما از افراد داده‌هایی از پیش در دست داشتیم، روش counterfactual evaluation روش مناسبی بود. این روش به این صورت است که با استفاده از داده‌های در دست انواع actionها را شبیه‌سازی می‌کند تا پاداش مورد انتظار هر action را براورد کند.



روش دیگر استفاده از multi arm bandit ساده است که پاداش مجموعه را بهینه می‌کند و یا استفاده از روش‌هایی همانند thompson sampling که regret را در طول زمان کمینه می‌کند. اگر اطلاعات از تعداد گروه‌ها هم در دست نباشد و بخواهیم از ابتدا خود گروه بندی‌ای ایجاد کنیم می‌توان به کمک روش‌هایی مثل A/B testing افراد را به دو یا چند گروه تقسیم کنیم و performance هر گروه را بررسی کنیم. کار دیگری که در این حالت می‌توان کرد این است که به کمک feature هایی در ابتدا با یک الگوریتم خوشه‌بندی داده‌ها را در هر مرحله جدا کرد و action بهینه را برای هر کدام از این گروه‌ها یافت.