Consider the scenario in which a shop has several promo codes to incentivize the customers that buy an item to buy a different item. The customers can belong to different classes and the promo codes can provide different discounts.

The essence of this problem is a multi-arm slot machine problem. We need to use algorithms to determine the price and discount distribution that will get the most profit. Among them, in terms of pricing, arm refers to the price of the product, and in terms of matching, arms are the best promotion task for each customer category.

**Scenario**

Consider a scenario where a retailer of videogames and related products sells two products. The first product a game, denoted as product 1. The second product is a DLC bundle of the game, denoted as product 2. To increase sales of the DLC bundle, the retailer offers a promotional discount to customers who purchase the game. This means that all customers who purchase the game will receive one of the different promotional offers for the DLC bundle.

The number of promotional discounts is defined by the retailer’s business department and provided to customers based on the customer category to which they belong, as the promotional discounts are limited.

There are a total of four types of customers, and each customer is divided into one of the following four categories:

1. Casual Player

Casual players are interested in buying the game, but they are not willing to spend too much for games, also if they receive a discount for the complete DLC bundle after the purchase, they will probably buy it.

1. Family member

Family members buys games for gifts to a member of the family, typically their sons, they have good budget and are willing to pay for the game, but unless there is a good discount on the DLC bundle, they might not buy it.

1. Hardcore gamer

They are fans of the game, they are willing to pay for the game and the DLC bundle even with no discounts, but they often have tight budgets, with better discounts, they will probably make the purchase.

1. Collector

Collectors have a big budget and if they are collecting this game, they are willing to pay higher prices for getting the game and the DLC bundle even if the discounts are not attractive.

The game price is 30$ per unit. The conversion rate is related to the category of customers who purchased the product, the price paid, and promotional discounts, so we constructed the conversion rate into a functional relationship between these items, which has been given by code. The following table shows the price and profit of product 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Game Price | 40 | 45 | 50 | 55 |
| Profit | 10 | 15 | 20 | 25 |

For retailers, the cost of the DLC Bundle is $10 each. The following table lists the price and profit of each original price and promotion. Consider a customer buying the game, and then giving the customer a promotion. The promotion includes four different discount levels on the DLC Bundle.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | D$ | Price1 | Pr. | Price2 | Pr. | Price3 | Pr. | Price4 | Pr. |
| P0 | -0$ | 20 | 10 | 24 | 14 | 28 | 18 | 32 | 22 |
| P1 | -3$ | 17 | 7 | 21 | 11 | 25 | 15 | 29 | 19 |
| P2 | -6$ | 14 | 4 | 18 | 8 | 22 | 12 | 26 | 16 |
| P3 | -9$ | 11 | 1 | 15 | 5 | 19 | 9 | 23 | 13 |

Pr. – Profit, D$ - Discount

The following assumptions are made in this scenario:

• On average, 1,000 customers arrive at the store every day

• The average number of customers in each category are:

Distribution of customers categories 1, 2, 3 and 4 [400, 300, 200, 100]

• Promotional distribution settings: [0.3, 0.3, 0.15, 0.25] are respectively for promotion levels P1, P2 and P3.

P0 is unrestricted in all cases because it corresponds to no discount in all cases

**Step 1.**

First, match the appropriate promotion method for each type of customer:

Parameters indicate:

，Cost matrix, customer type i allocates the cost of promotion j;

，Binary allocation matrix, if and only if the promotion j is allocated to customer i 







**After determining the matching method, calculate the profit in the corresponding situation:**

Parameters indicate:

：It is the average number of customers of this category who arrive at the store every day.

：Is the price selected by the store for product 1 at time t.

:Is the price selected by the store for product 2 at time t.

:It refers to the profit when the store sells a product 1 at price P.

：It refers to the profit when the store sells a product 2 at price Q and promotion F.

：It is the conversion rate of type i of the first product at time t, assuming the price is P.

:It is the conversion rate of category i of the second product at time T, taking into account price Q and promotion F.

:Is the score of promotion f provided to category i at time t.

For the first product, the total revenue of the store in a day will be equal to the price profit it chooses multiplied by the number of customers who bought it, which is equivalent to the average number of customers in each category multiplied by the conversion rate of the category at the design price with. This means that the total profit can be expressed as:



The number of customers of each type who can purchase the second product only corresponds to the part of the first product, and then the total number of customers who purchase the second product is:



We need to aggregate all promotions to get the total amount. If we add up all categories and include the profit of the second product, we will get the second part of the total daily profit of the store, corresponding to:



In a one-year time interval, by adding the two total profit items, we get the total revenue of the store in one year:



This is the amount the store is trying to maximize, and the variables they need to modify are the price and quantity of the two products and the price and quantity of promotions offered for each category. Then the store needs to solve the following optimization problems:





The only restriction we need to add is that the sum score of the promotions offered is equal to 1.

# Step 2.

We have 3 random-defined variables.

* First, the number and types of customers who arrive at the store every day.
* Then how many of these customers will buy the first product,
* How many of the customers who buy the first product will buy the second product.

The number of customers arriving each day can be modeled by supporting any distribution of natural numbers. We will represent the distribution of the number of customers in Ci, t（θi，t）, where θi, tare weight are the corresponding distribution parameters.

The number of customers who bought the first product can be modeled as the result of the Bernoulli experiment, where the number of experiments is the number of customers that arrive, and the probability of success is the category conversion rate calculated at the corresponding price. Then, the number of buyers for the first product can be modeled as a binomial distribution, with the number of trials equal to the number of customers, and the probability of success equal to the conversion rate.

Finally, the sales volume of the second product can also be modeled as a Bernoulli test, but in this case, the number of trials corresponds to the number of buyers of the first product, and the probability of success is the conversion rate of the second product. In addition, we need to consider where the customer’s promotion is provided.

Then, the deterministic optimization problem in Part 1 can be modified to the following stochastic optimization problem:











In the formula, Ci, t（θ） is any probability distribution that supports natural numbers and parameters （θ）), and B (c, p) is a binomial distribution without the number of trials and the probability of success. In this case, ci，t，χ1i，t and χ2i, t is a random variable, modeling the number of customers who arrive at the store in one day, the number of people who bought the first product, and the number of people who bought the second product.

Since this optimization problem is not deterministic, we cannot use the same method as in the previous section, we need to use online learning methods. For this type of problem, we can use the multi-arm bandit method, where each arm is a combination of the prices of two products and the effective distribution of promotional activities, and the parameters are updated every day.

## Step 3.

Consider the case in which the assignment of promos is fixed, and the price of the second item is fixed, and the goal is to learn the optimal price of the Product 1 (Game), by using a fixed price for the product 2 (DLC Bundle). Assuming that the number of customers, and their distribution per class is known as well as the conversion rate associated with the second item. Is adopted both an upper-confidence bound approach and a Thompson-sampling approach and compare their performance.

Daily customers number is extracted from the Gaussian distribution. Since the number of customers is known, the number of rounds per day is known.

## Solution

Use the upper limit of confidence method (UCB) and Thompson sampling method to learn the optimal price of the first item and compare its performance. With this method, the number of arms is equal to the number of candidate prices for product 1, which is 4 in total. The number of customers arriving at the store every day is set to Class\_i = {400,300,200,100}. The conversion rate of product 2 is the same for each arm. Using an offline environment, two learners are used to simulate the learning process for 365 days a day. Every day, the arms of two independent learners will be pulled, the reward of the selected arm will be calculated, and the learner will be updated.

**Learning Results:**

* Thompson learner converges to price 45$ for product 1
* UCB learner converges to price 40$ for product 1

### Reward

* Total margin collected by UCB: 276.6535643701261
* Total margin collected by Thompson Sampling: 293.2198688059615

The following chart shows the cumulative rewards collected by these two algorithms. The cumulative reward collected by the Thompson sampling algorithm is slightly larger than the UCB algorithm.

Gráfico, Gráfico de líneas, Gráfico de dispersión

Descripción generada automáticamente

The following chart shows the average of the expected reward for each algorithm.

Gráfico

Descripción generada automáticamente

The three algorithms used are

* Thompson sampling algorithm.
* UCB algorithm.
* perspective clairvoyant algorithm.

The perspective algorithm is the optimal solution of the problem calculated when all the parameters in the problem are known. According to the data, the solution produced by Thompson's sampling algorithm is closer to the optimal solution.

### Regrets

* Total expected regret of UCB: 25.934721860568168
* Total expected regret of Thompson Sampling: 10.581789530499975

# Step 4.

Like Step 3, but the conversion rates associated with product 2 and the number of daily customers is unknown. The goal is to provide solutions to the pricing problems for the game and DLC Bundle (products 1 and 2). Daily customers are unknown and comes from a Gaussian distribution.

Daily customers number is extracted from the Gaussian distribution, and the customer categories that arrive at the shop are randomly selected according to the predefined category distribution.

## Solution.

To find the optimal price of the first product and the second product, the upper limit of confidence method (UCB) is used. Unlike Step 3, the product 2 introduces 4 candidate prices. Like on step 3, the product promotion distributions are fixed. The number of arms among the learners of product 1 and product 2 will be 4 to each, which is equal to the number of candidate prices.

In the learning process, the number of customers in each category is sampled from a normal distribution. In the simulation, the customer arrival distribution is a randomly selected class in which there are customers remaining on the day. Pull the arm of product 1 and observe the reward. If the reward of product 1 is positive, that is, the customer purchases the first product, pull the arm of the second product to calculate the reward. Then update the two learners with the corresponding arm and calculate the cumulative reward.

**Learning Results:**

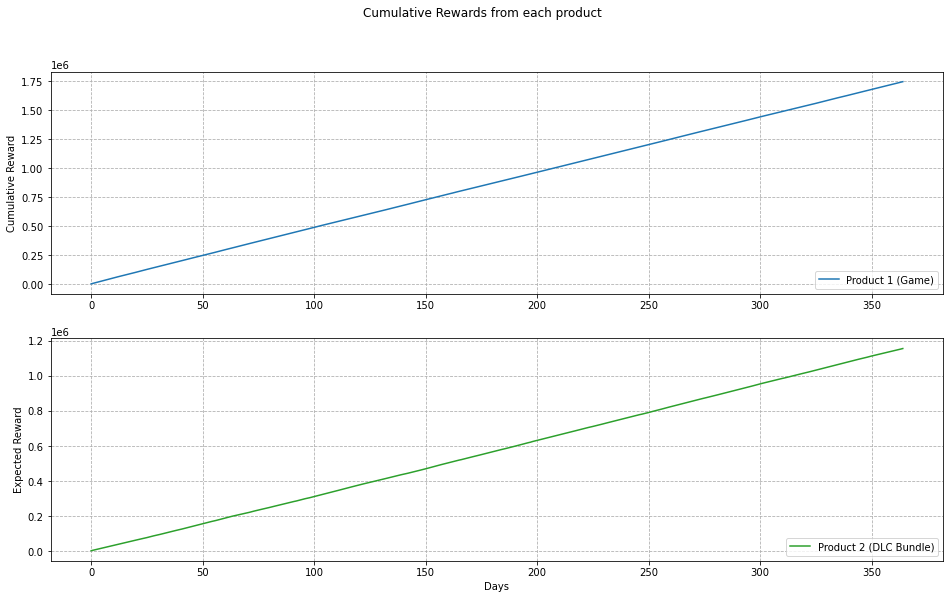
* learner 1 converges to price 40$ for product 1.
* learner 2 converges to price 28$ for product 2.

### Rewards

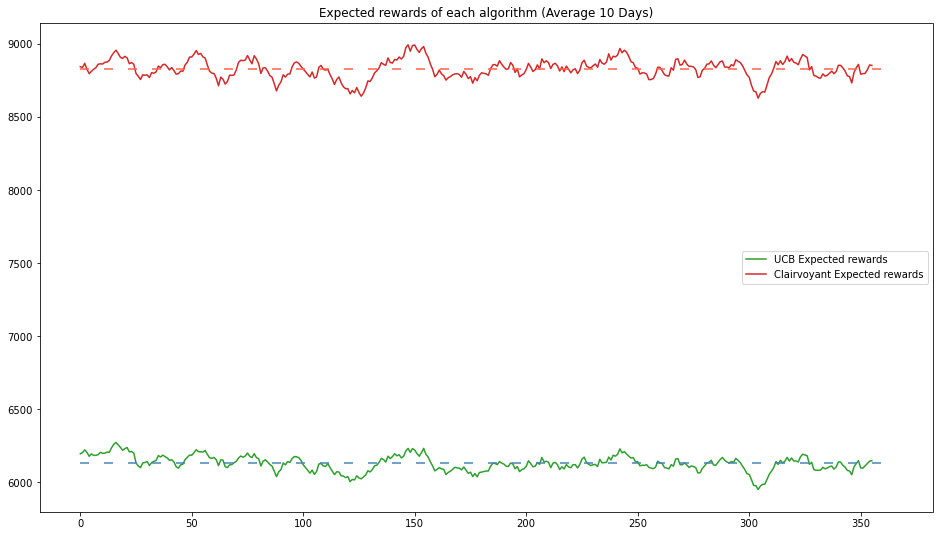
Total profit collected from product 1: 1’748.095$

Total profit collected from product 2: 1’156.760$

The next charts show the cumulative reward for each product in the two product pricing problems.



The next shows the 10-day moving average of the expected returns of the UCB- and perspective algorithms. The UCB algorithm has never achieved the high average expected return of the clairvoyance algorithm.



### Regret

Total expected regret: 984794.4165627654

# Step 5.

The goal is to find the best distribution of promotions for each customer category.

The price is fixed.

## Solution.

The method is very similar to the previous step. The main difference is that arms are no longer price candidates, but they are the best promotional tasks for each customer category. This boils down to the use of an allocation algorithm on the matrix, where the rows represent customer categories, and the columns represent promotions. Since each category can be assigned to the first promotion, three more promotion categories are introduced.

For each day, we will determine the number of promotional activities based on the number of customers. Once the arm of Product 1 is pulled and the observed reward is positive, we solve the matching problem. If there are no more promotions for a certain type, the cost of this column in the matrix will be as high as possible, so this column will never be selected. After the matching problem is resolved, we determine which promotion has been selected for the arriving customer. Learners will be updated with the observed rewards, and the number of daily promotions for matching types will be reduced.

**Learning Results**

The number of times a class was offered each promo level is shown below:

[[5.0900e+02 6.1710e+04 0.0000e+00 0.0000e+00]

[2.9472e+04 1.1000e+01 2.2334e+04 9.0000e+00]

[2.2100e+03 9.0000e+00 1.0000e+00 3.6367e+04]

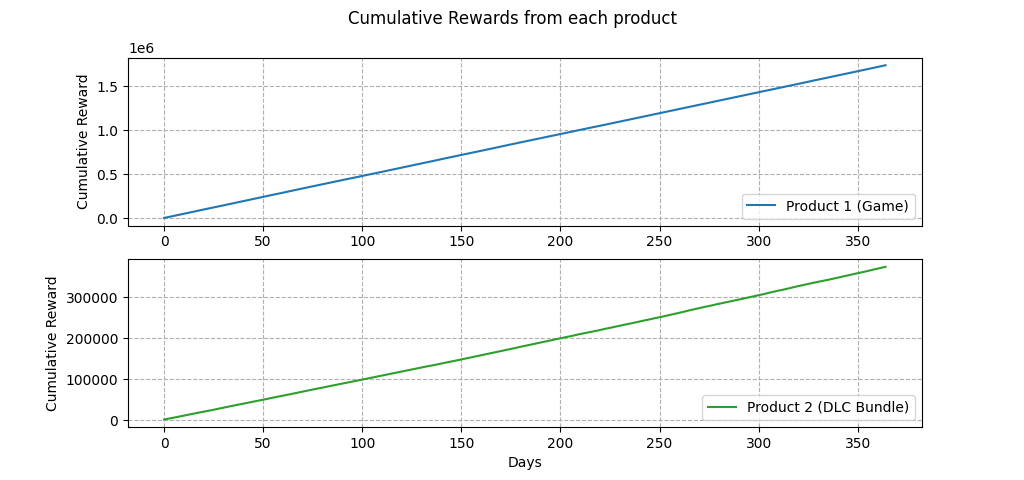
[5.8800e+02 4.0900e+02 1.8105e+04 1.8350e+03]]

### Reward

Total profit collected from product 1: 1’735.690$

Total profit collected from product 2: 374.792$

The chart shows the cumulative reward.



And the next chart displays the difference between the 365-day average of UCB and fluoroscopy.

Gráfico, Gráfico de líneas

Descripción generada automáticamente

### Regret

Total expected regret: 126237.23955625683

# Step 6

Like Step 5, but prices of product 1 and product 2 is no longer fixed.

The problem now is pricing and matching and the following assumptions are made:

* The environment is fixed
* The arrival of customers is continuous, which means that the rewards returned for each customer are based on level and price, rather than rewards throughout the day.

The number of customers per day comes from a Gaussian distribution. A learning process is applied every day for each customer that arrives. According to a specific category distribution, the customer category to which the arriving customer belongs is randomly selected. The promotion level distribution is set to a small fraction of the total customers that arrive and is constant. There are two optional settings for these scores, as described in the scenario section.

Fully Revised until this point

## Solution.

The methods used are the UCB method for the first product pricing problem and the matching UCB method for the second product matching problem. The number of arms of the pricing learner is equal to the number of candidate prices, and there are a total of 4 candidate prices.

For the second matching problem of product 2, the number of arms should be equal to 112, but to use the linear sum allocation algorithm to solve our optimization problem, another 84 arms were created. These arms are additional copies of the P0 discount. As before, the number of customers in each category is sampled from a normal distribution and truncated to 0 to avoid negative numbers.

The customer arrival is simulated by selecting categories randomly with remaining customers on the day. As in the previous part, we pull the arms for the first product and observe the rewards. If the reward is positive, we will pull the second price arm, otherwise the reward for the second product is set to zero. Then update the learner. For each customer, the rewards calculated are aggregated into cumulative rewards, and the expected rewards for each day are calculated from this.

LEARNING RESULTS

class 1 learners converged to price 1000 for product 1 and price 100 for price 2

class 2 learners converged to price 1100 for product 1 and price 100 for price 2

class 3 learners converged to price 900 for product 1 and price 100 for price 2

class 4 learners converged to price 900 for product 1 and price 110 for price 2

With the following shows the number of times a class was assigned each promo level:

[[13553. 0. 0. 0.]

[47257. 0. 0. 0.]

[55115. 0. 0. 0.]

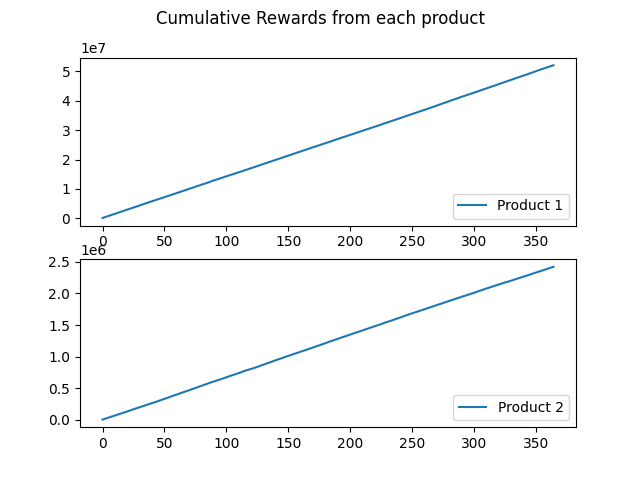
[36441. 0. 0. 0.]]

### reward

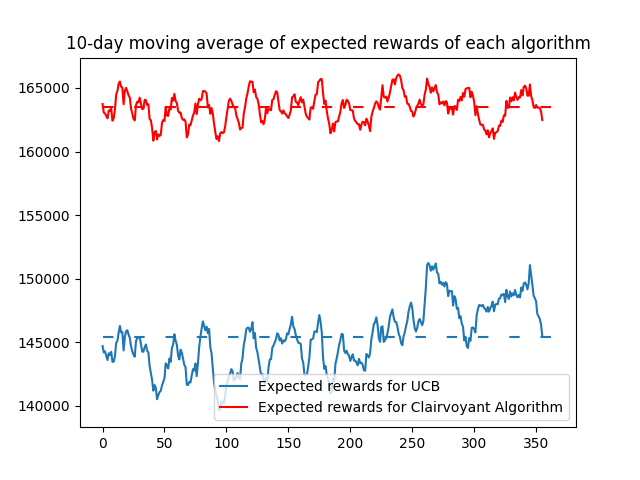
Total profit collected from product 1: 52102500.0

Total profit collected from product 2: 2420140.0

The next chart shows the cumulative reward for each product.



The next chart shows the 10-day moving average expected rewards of UCB and the perspective algorithm.



### Regret

Total expected regret: 6601082.073262578

# Step 7.

This step requires to run the algorithm in a non-stationary environment, which means that the parameters will not remain constant over time. In our example, we divide the year into two phases of equal time: in the first phase, all parameters are the same as those in the previous step; in the second phase, the conversion rates are changed due to scenario.

## Solution

As the parameters are constantly changing, learners need to be able to adapt to these changes. One method is to use the sliding window method. In this method, we choose a time window of length and only use the last day's return to train the learner. This technique allows learners to update parameters and ignore information that may be out of date. To adapt the algorithm to our approach and ensure that all customers are considered in a round, we need to use a window whose length is equal to the square root of the period times the number of expected customers in a day.

LEARNING RESULTS

Phase 1 results:

class 1 learners converged to price 1000 for product 1 and price 100 for 2

class 2 learners converged to price 1000 for product 1 and price 100 for 2

class 3 learners converged to price 1000 for product 1 and price 100 for 2

class 4 learners converged to price 1000 for product 1 and price 110 for 2

With the following shows the number of times a class was assigned each promo level:

[[ 5779. 0. 0. 0.]

[27841. 0. 0. 0.]

[23182. 0. 0. 0.]

[17185. 0. 0. 0.]]

Phase 2 results:

class 1 learners converged to price 900 for product 1 and price 100 for 2

class 2 learners converged to price 900 for product 1 and price 100 for 2

class 3 learners converged to price 1100 for product 1 and price 100 for 2

class 4 learners converged to price 1100 for product 1 and price 110 for 2

With the following shows the number of times a class was assigned each promo level:

[[ 2843. 0. 0. 0.]

[15816. 0. 0. 0.]

[22240. 0. 0. 0.]

[16568. 0. 0. 0.]]

### Reward

Total profit collected from product 1: 47152900.0

Total profit collected from product 2: 1685450.0

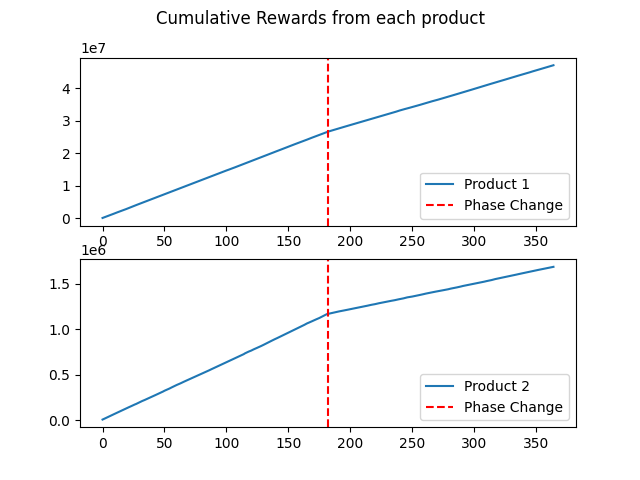


Figure-9

In Figure 9, the influence of non-stationarity can be observed. Specifically, we can observe that the cumulative reward slope of product 2 is smaller in the second stage than in the first stage. This decrease is due to the reduced conversion rate of the product, which translates into a decrease in sales.

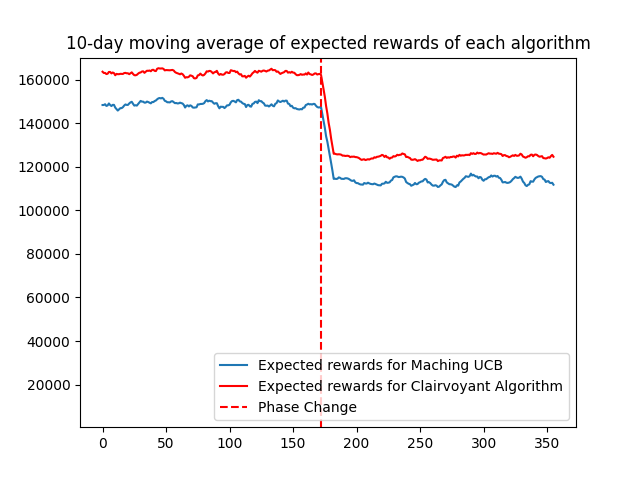


Figure-10

When we compare the results of the perspective algorithm with our results, we can clearly prove this phenomenon. As shown in Figure 10, we can see that the return of the UCB algorithm will never be close to the optimal result.

### Regret

Total expected regret: 4660760.259741467

# Step 8

The goal of this section is to test the algorithm in a non-stationary environment again, but now the change detection method is used instead of the sliding window method. The rest of the settings are the same that the ones used in step 7.

## Solution

This step requires the use of an algorithm to detect changes, so CUSUM (Cumulative Sum) algorithm is being used, as it’s a good algorithm for detecting mutations. The basic idea of the CUSUM algorithm is to use the function of the observed sample as the step size of the random walk. This random walk is designed to have a positive average drift after a change point, and a negative average drift if there is no change. Therefore, if the random walk exceeds a certain positive threshold, CUSUM will send a change signal.

LEARNING RESULTS

Phase 1 results:

class 1 learners converged to price 1000 for product 1 and price 100 for 2

class 2 learners converged to price 1000 for product 1 and price 100 for 2

class 3 learners converged to price 1000 for product 1 and price 100 for 2

class 4 learners converged to price 1100 for product 1 and price 110 for 2

With the following shows the number of times a class was assigned each promo level:

[[ 6417. 0. 0. 0.]

[29376. 0. 0. 0.]

[24374. 0. 0. 0.]

[16065. 0. 0. 0.]]

Phase 2 results:

class 1 learners converged to price 900 for product 1 and price 100 for 2

class 2 learners converged to price 900 for product 1 and price 100 for 2

class 3 learners converged to price 1000 for product 1 and price 100 for 2

class 4 learners converged to price 1100 for product 1 and price 110 for 2

With the following shows the number of times a class was assigned each promo level:

[[ 3805. 0. 0. 0.]

[19484. 0. 0. 0.]

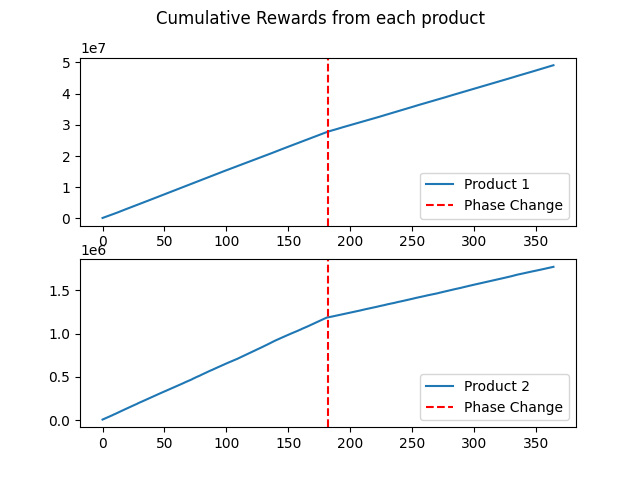
[24427. 0. 0. 0.]

[15711. 0. 0. 0.]]

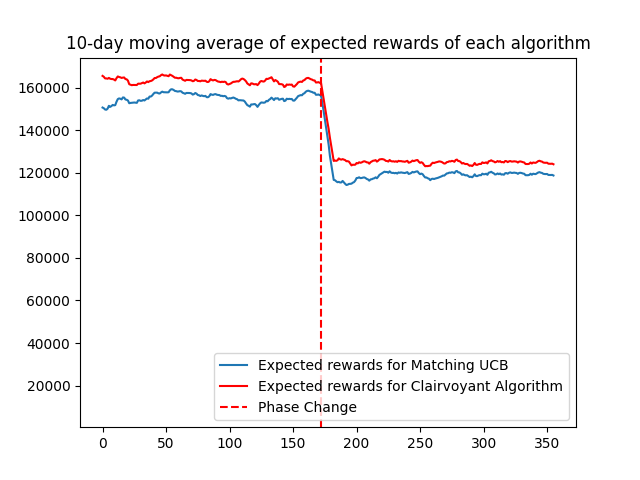
### Reward

Total profit collected from product 1: 49105950.0

Total profit collected from product 2: 1772910.0



The behavior observed in Figure 11 is like that of Figure 9. Since the conversion rate of these two products is reduced, the cumulative reward slope of these two products is smaller in the second stage



According to data we can see that the return of the UCB algorithm is better than the sliding window, and it is closer to the best result.

### Regret

Total expected regret: 2625390.366953019