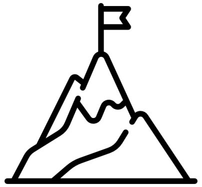
Pricing and Matching

Fabio Barbieri – Luca Castellazzi – Giulia Mussi

The problem



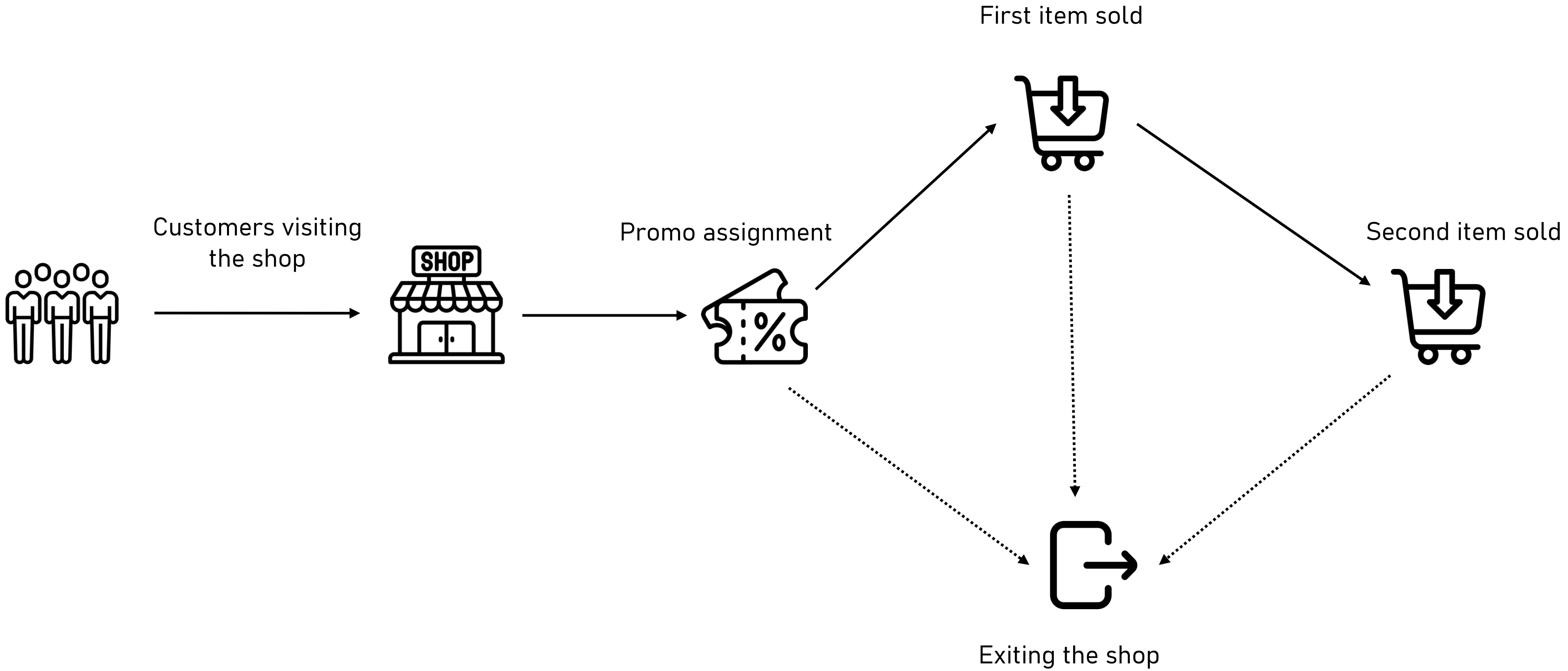
A shop is focusing on selling two products:

1. mountaineering boots
2. mountain sticks

Objectives:

1. Finding the optimal price for both the first and second item.
2. Incentivize the selling of the second item trough proper promo codes assignment.

Scenario

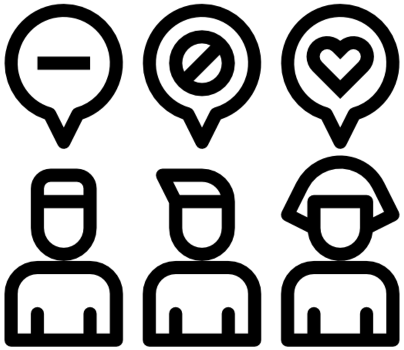


First item sold
Customers visiting the shop
Promo assignment
Second item sold
Exiting the shop

Customers

The customers are divided in four different classes:

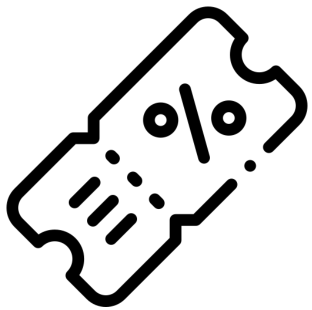
- Junior Professionals (18 – 30 y.o.)
- Junior Amateurs (18 – 30 y.o.)
- Senior Professionals (> 30 y.o.)
- Senior Amateurs (> 30 y.o.)



Promo Codes

In our scenario we have different types of promo codes, each one granting a different level of discount:

- P0 – no discount
- P1 – 15% discount
- P2 – 25% discount
- P3 – 40% discount



Promo Codes allocation

The marketing unit allocates different amounts of promo codes, for each type of discount.

We worked in two different settings:

	P0	P1	P2	P3
Setting 0	0.4	0.2	0.22	0.18
Setting 1	0.2	0.3	0.1	0.4

NOTE: the values express the percentage of the promo codes with respect to the total amount of customers


OFFLINE PROBLEM
and
VARIABLES INTRODUCTION

Mathematical formulation

Objective function:

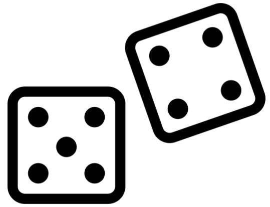
$$\max \sum_{i=1}^{N} \sum_{j=1}^{N} cr_1(p_1, cl_i) \; \{ \, m_1 \; + \; x_{ij} \; cr_2(p_2 \, (1 - promo_j), cl_i) \; m_2(promo_j) \; \}$$

Constraints:

$$\sum_{i=1}^{N} x_{ij} \leq 1 \qquad \forall j = 1, 2, \ldots, N$$

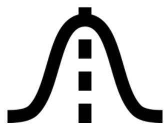
$$\sum_{j=1}^{N} x_{ij} \leq 1 \qquad \forall i = 1, 2, \ldots, N$$

Variables introduction

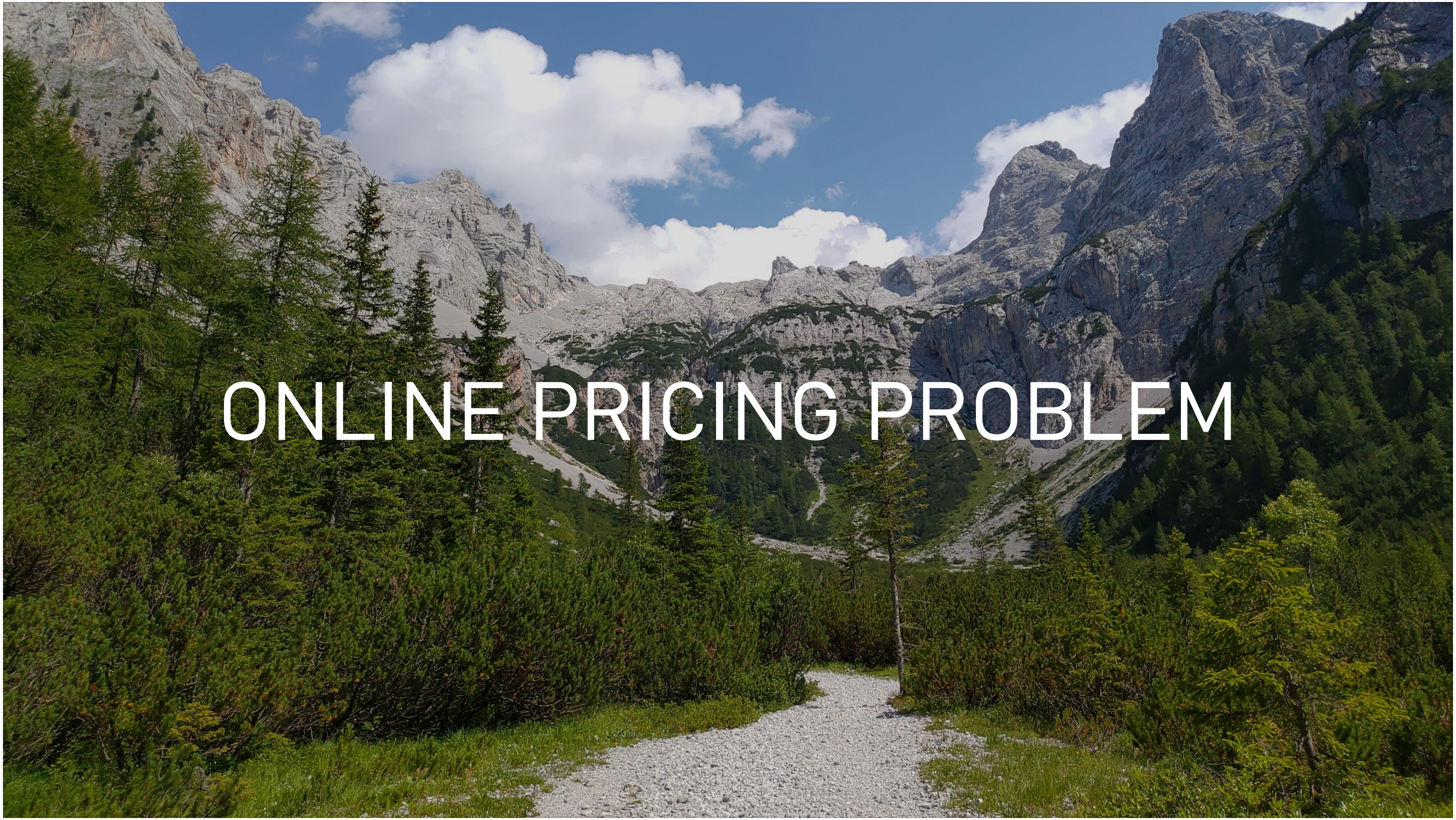


We have identified the following **random variables** that characterize the problem which we are dealing with:

- **Daily number of customers per class**: modelled with truncated Gaussian distributions.



- **Observed rewards**: modelled with Bernoulli distributions with mean in specific values of the conversion rates
  - 0 = customer does not buy the item
  - 1 = customer buys the item


ONLINE PRICING PROBLEM

ASSUMPTIONS:

- The prices of both the items are the same for each different class of customers.
- The conversion rates of both the items are stationary.

OBJECTIVE:

- Find the optimal price for the first item.

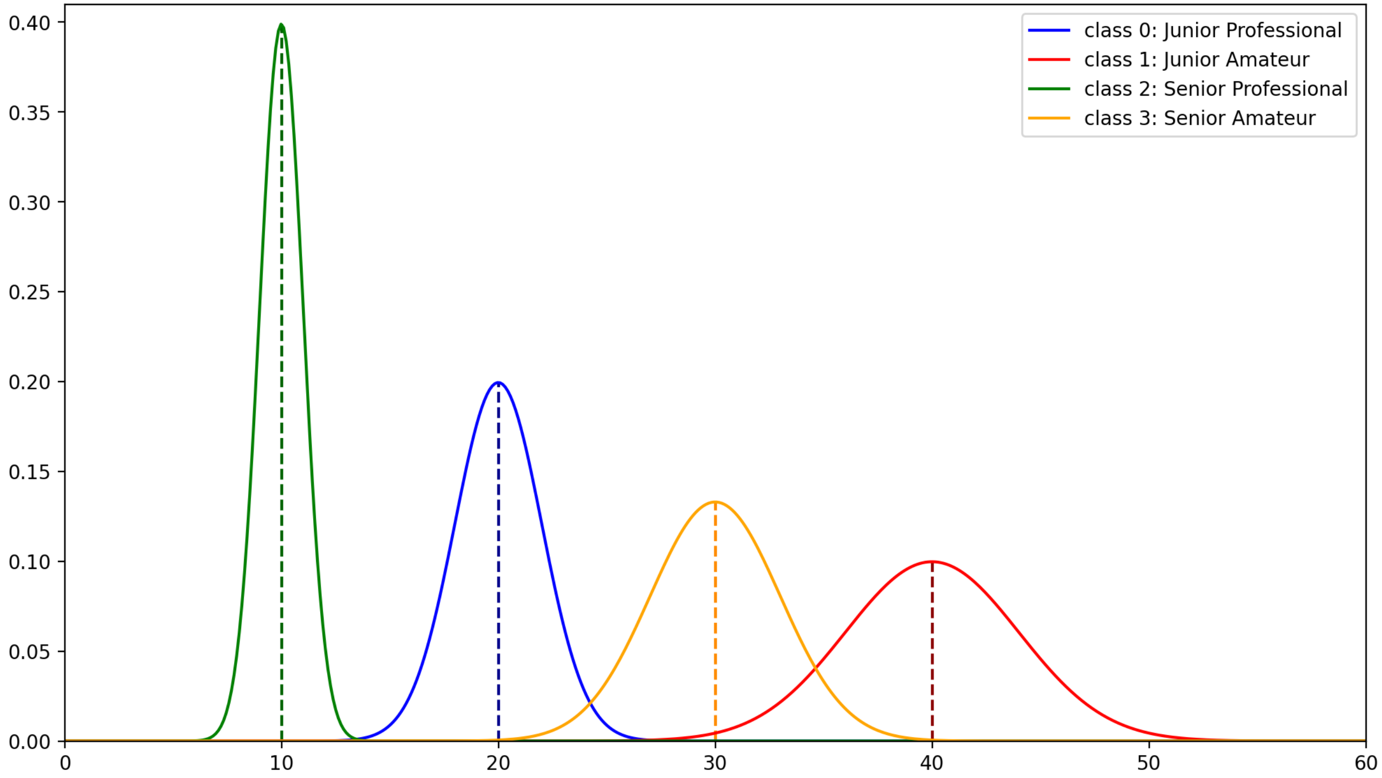
We solved the problem in two different scenarios:

	price 1	price 2	conversion rate 1	conversion rate 2	number of daily customers
step 3	to be learnt	fixed	unknown	known	known
step 4	to be learnt	fixed	unknown	unknown	unknown

Number of daily customers

- In the **first scenario** we **fixed** a daily number of visits by the customers of each class. More in detail:
  - Class 0 : 20 customers
  - Class 1 : 40 customers
  - Class 2 : 10 customers
  - Class 3 : 30 customers

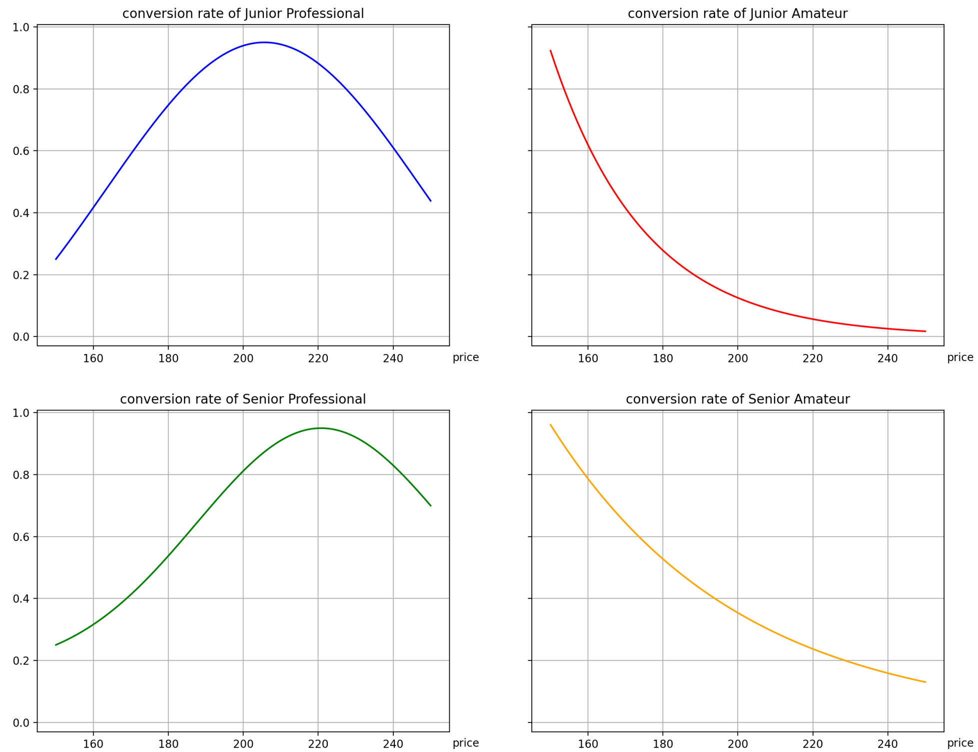
- In the **second scenario**, instead, the previous numbers were used as mean values of 4 different **truncated Gaussian distributions** from which, each day, the number of customers is extracted.



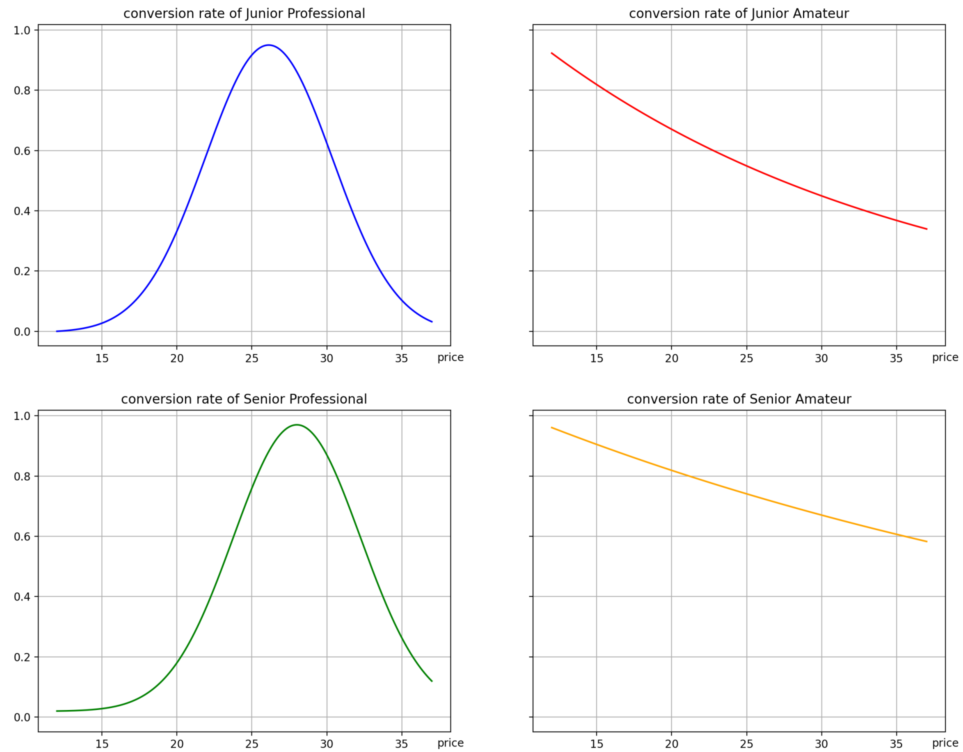
Conversion rates

Each class is associated with a different conversion rate function:

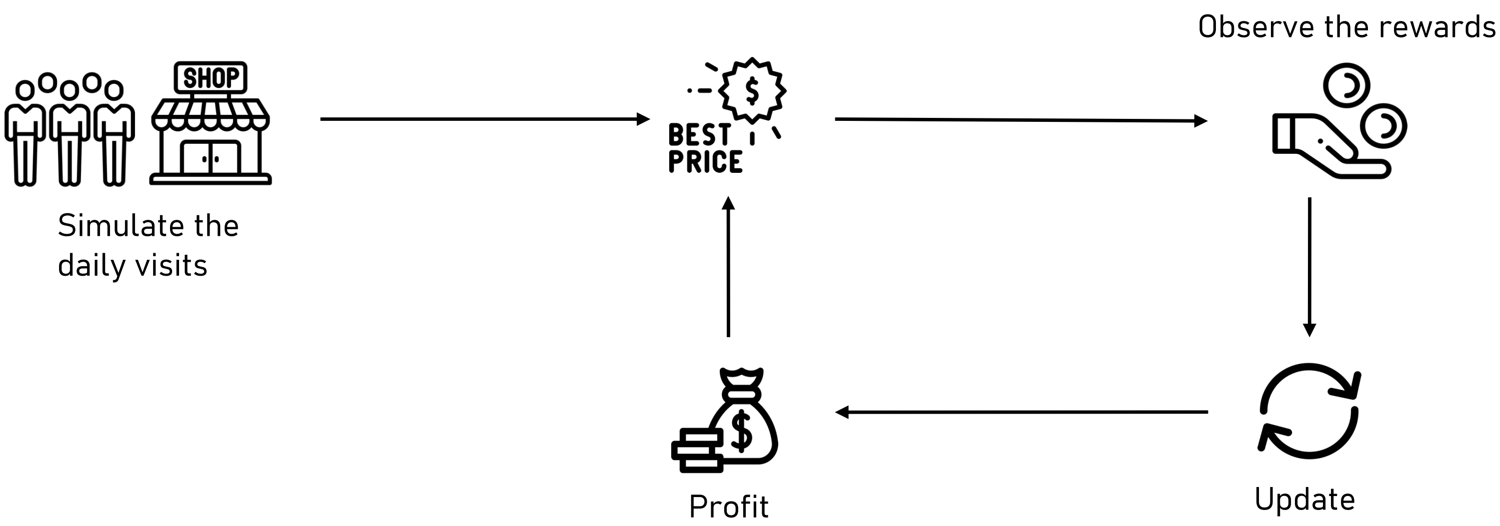
Conversion rates for **item 1**



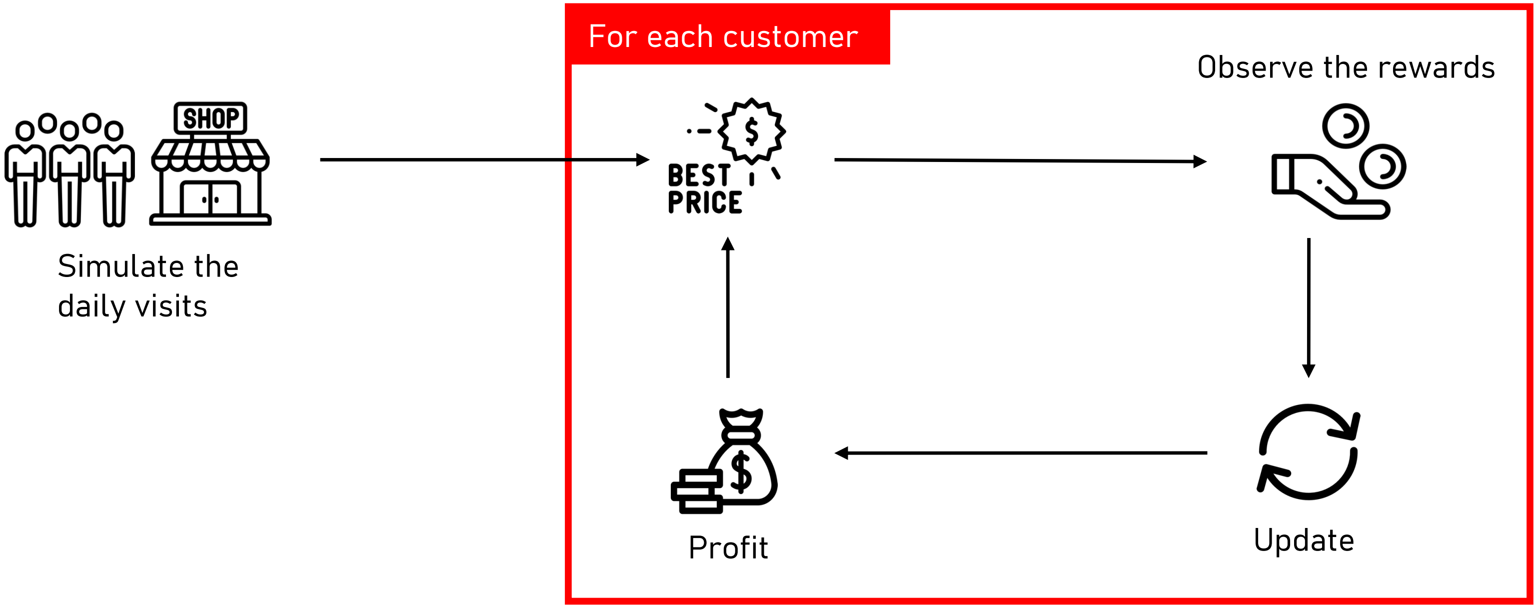
Conversion rates for **item 2**



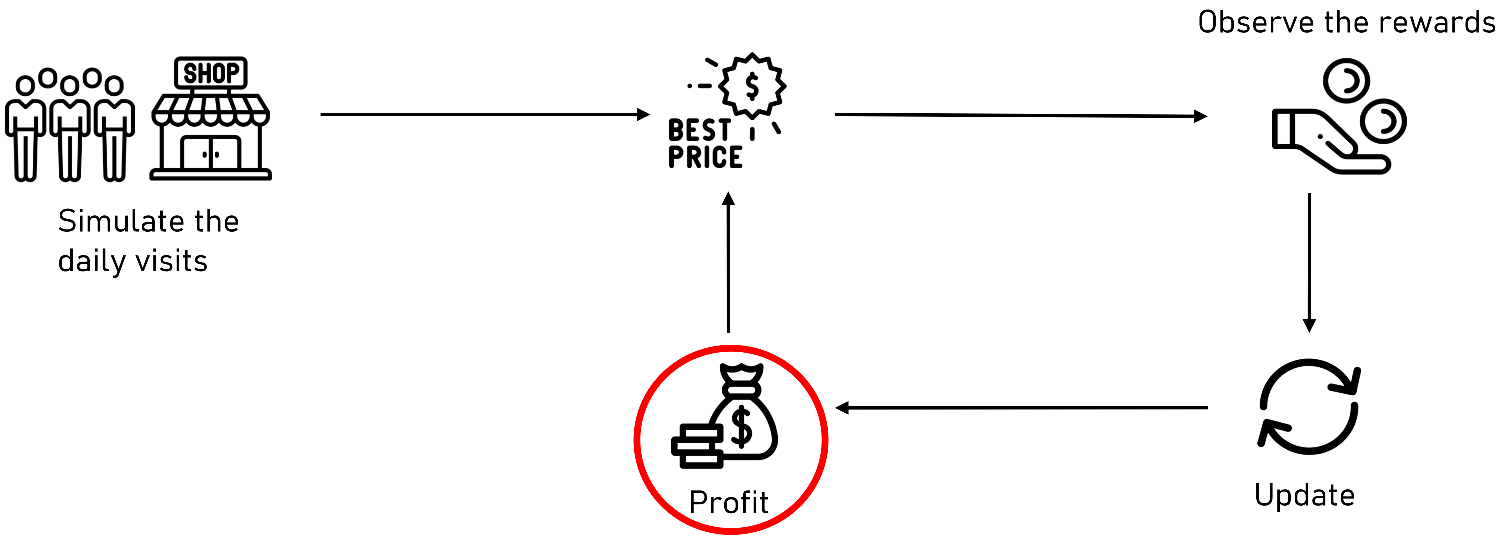
Algorithm core idea



Algorithm core idea

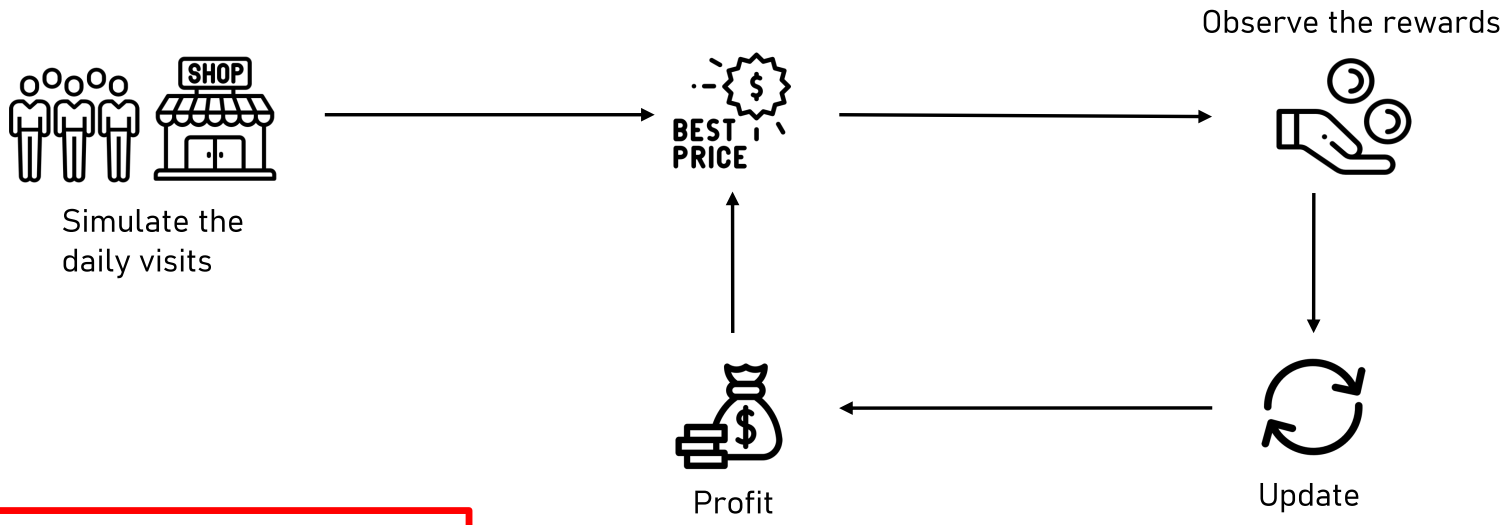


Algorithm core idea



$$\boxed{r_1(c) * (\, m_1 + r_2(c,p) * m_2(p) \,)}$$

Algorithm core idea

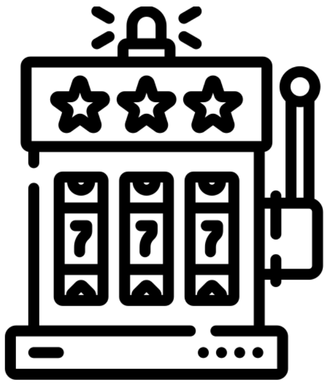


In the **second scenario**, at the end of each day, the Learner performs also the **tuning of the estimates over the number of customers** that will visit the shop the next day

Solution approaches

Being a pricing problem, our work has been focused on the implementation and adaptation of two well-known bandit algorithms:

1. Thompson Sampling
2. UCB1



Thompson Sampling approach

- In the first scenario:
  - Each time, the price to be selected is the one that maximizes the **expected profit**.
  - The conversion rate over the price of the **first item** is estimated **sampling** from a Beta distribution.
  - The Beta distribution is initialized as a Uniform and then updated after observing the rewards from the Environment.

- In the second scenario:
  - **Also** the conversion rate over the price of the **second item** is estimated **sampling** from a Beta distribution.

UCB1 approach

- In both scenarios:
  - At the beginning, all prices are selected once.
  - Then, the price to be selected is the one with the highest **upper confidence bound of the empirical mean** over the daily profits.
  - Each upper confidence bound is computed adding a confidence term to the empirical mean
  - The confidence term is:

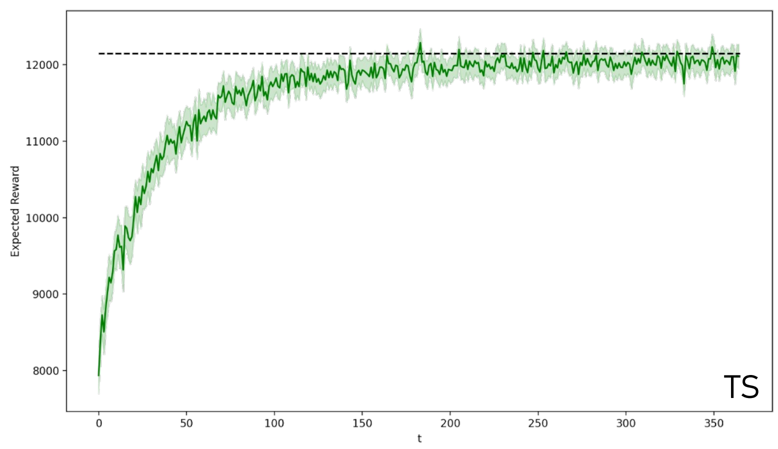
$$\sqrt{\frac{2\ \log(t)}{\max\{1,\ n\_pulled\_arm\}}} * profit$$

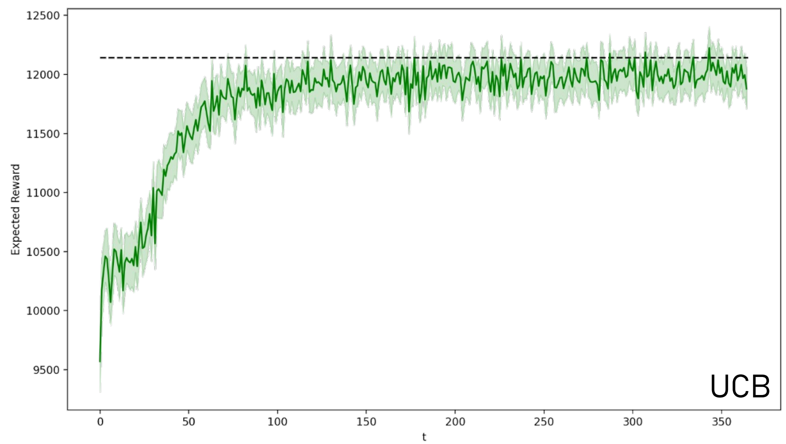
UCB1 approach

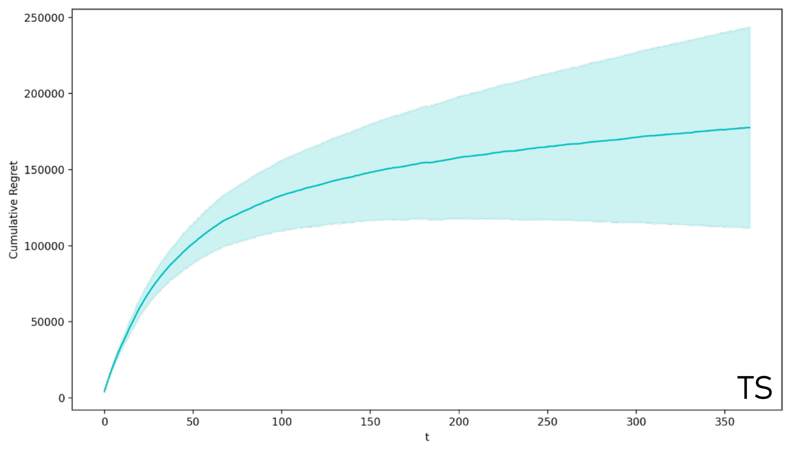
- In both scenarios:
  - At the beginning, all prices are selected once.
  - Then, the price to be selected is the one with the highest **upper confidence bound of the empirical mean** over the daily profits.
  - Each upper confidence bound is computed adding a confidence term to the empirical mean
  - The confidence term is:

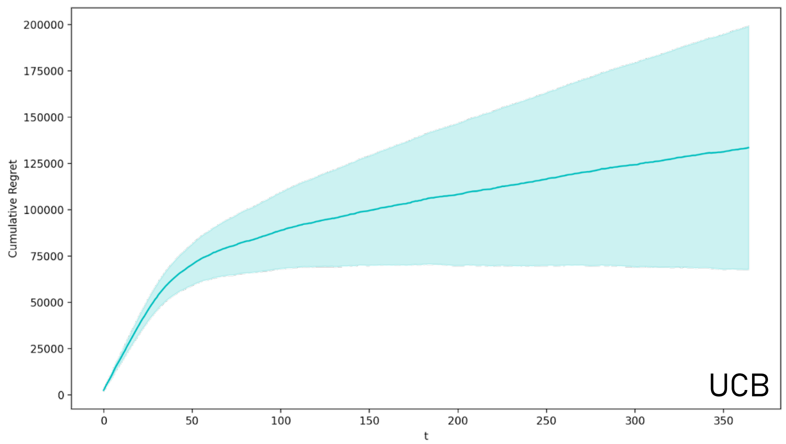
$$\sqrt{\frac{2\ \log(t)}{\max\{1,\ n\_pulled\_arm\}}} * profit$$

Results – step 3

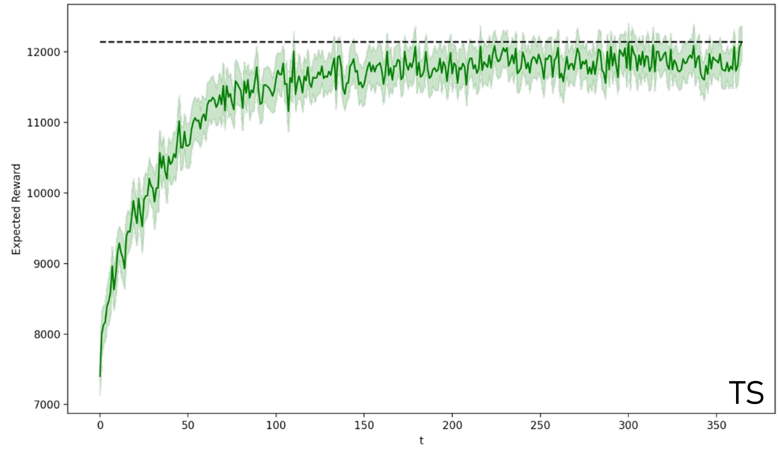


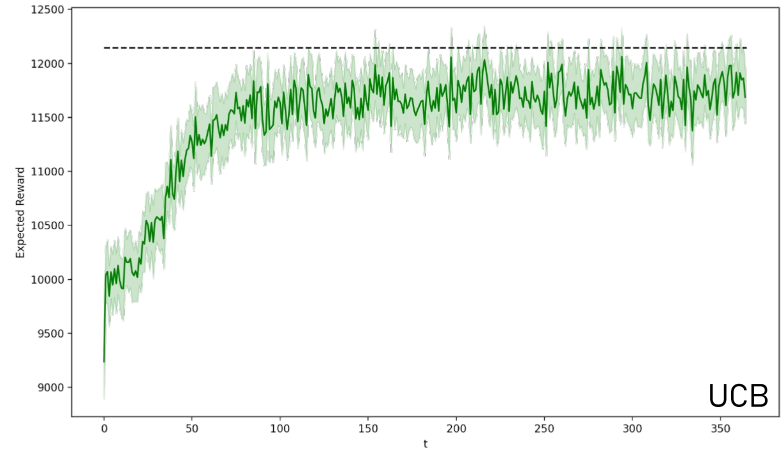


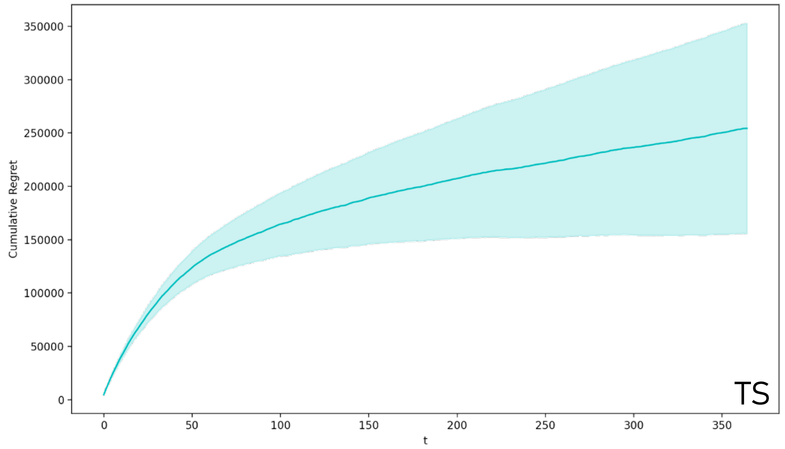


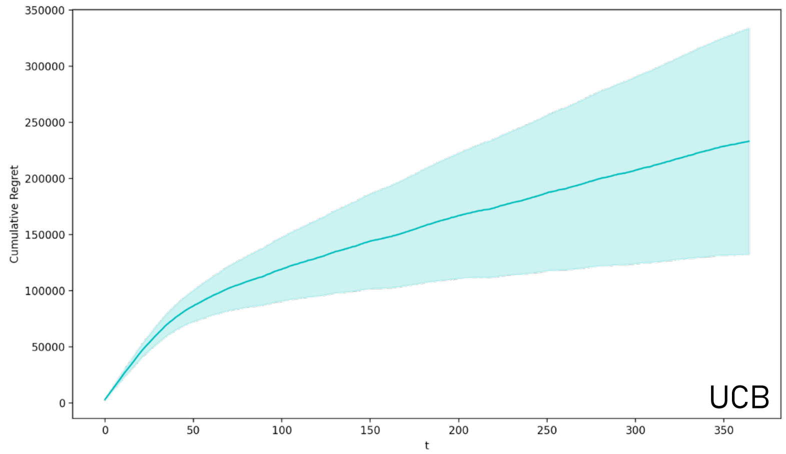


Results – step 4










ONLINE MATCHING PROBLEM

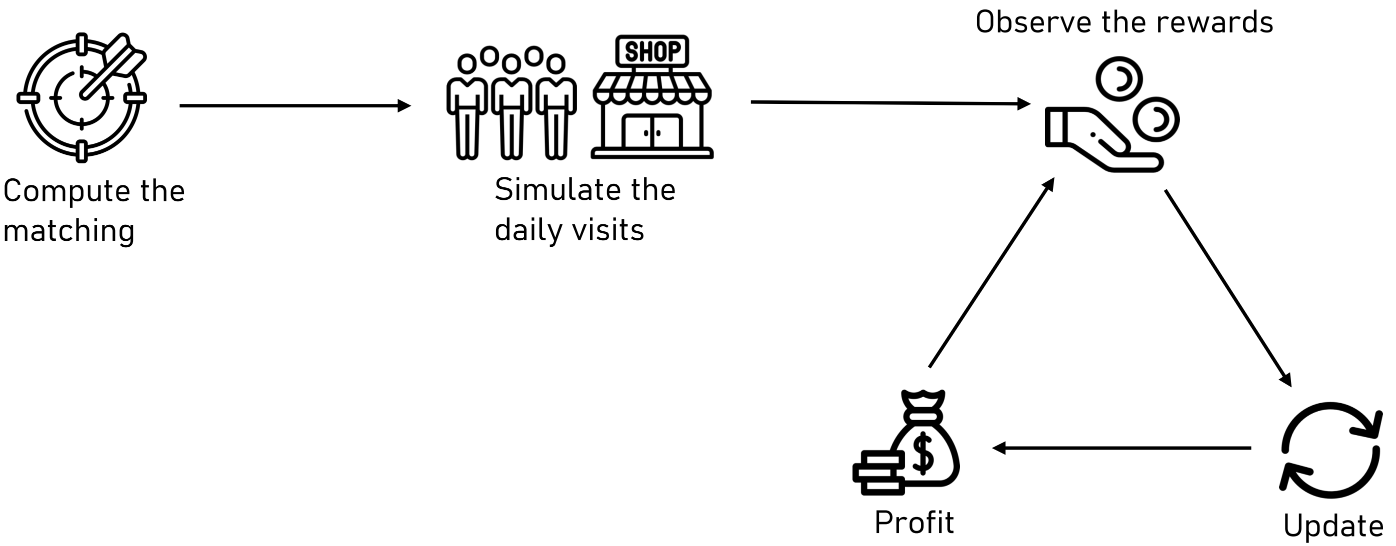
ASSUMPTIONS:

- The prices of both items are fixed.
- The prices of both items are the same for each different class of customers.
- The conversion rates over the prices of both items are stationary.

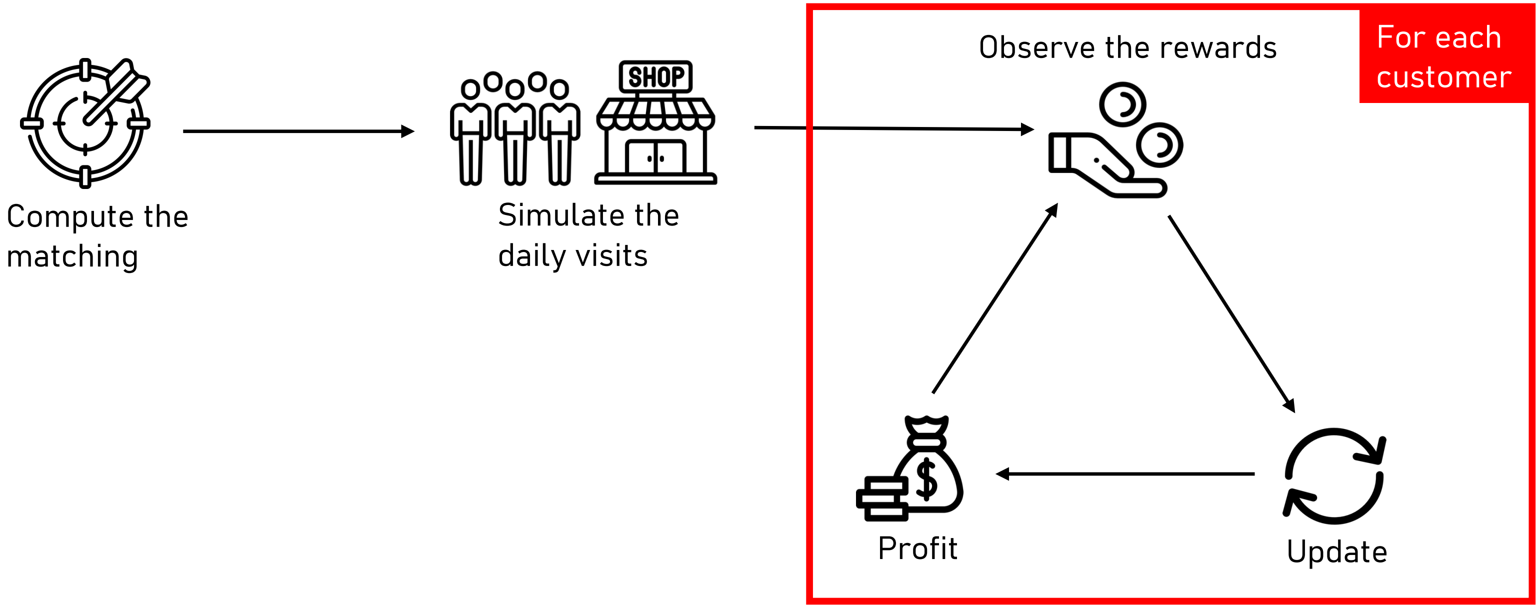
OBJECTIVE:

- Optimize the assignment of promos.

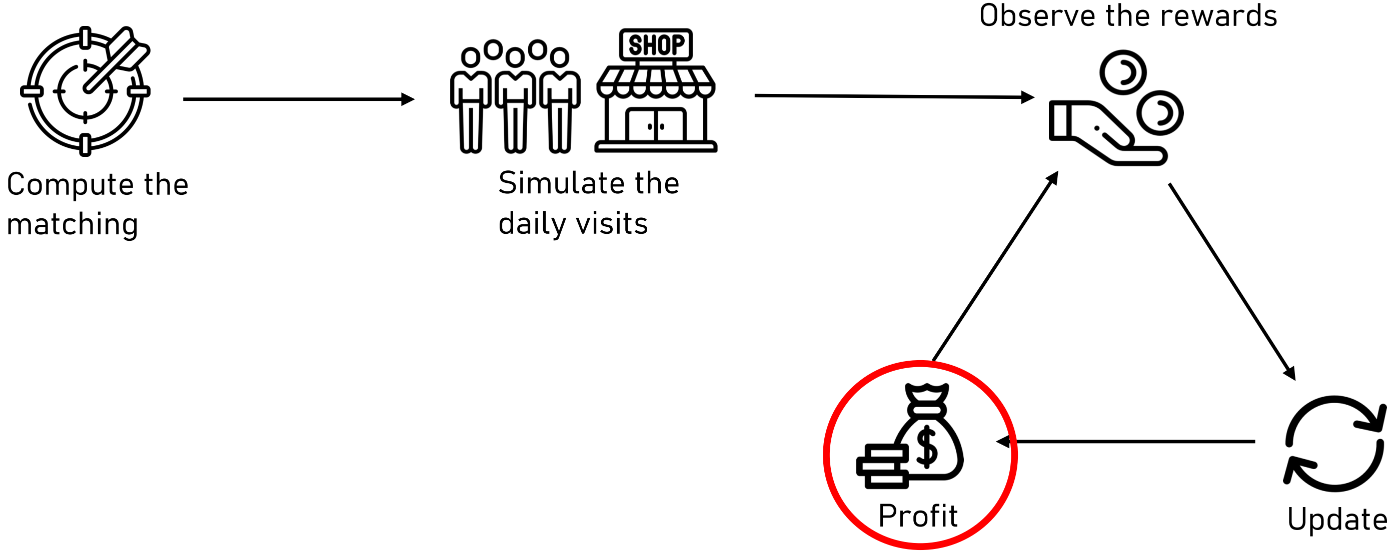
Algorithm core idea



Algorithm core idea

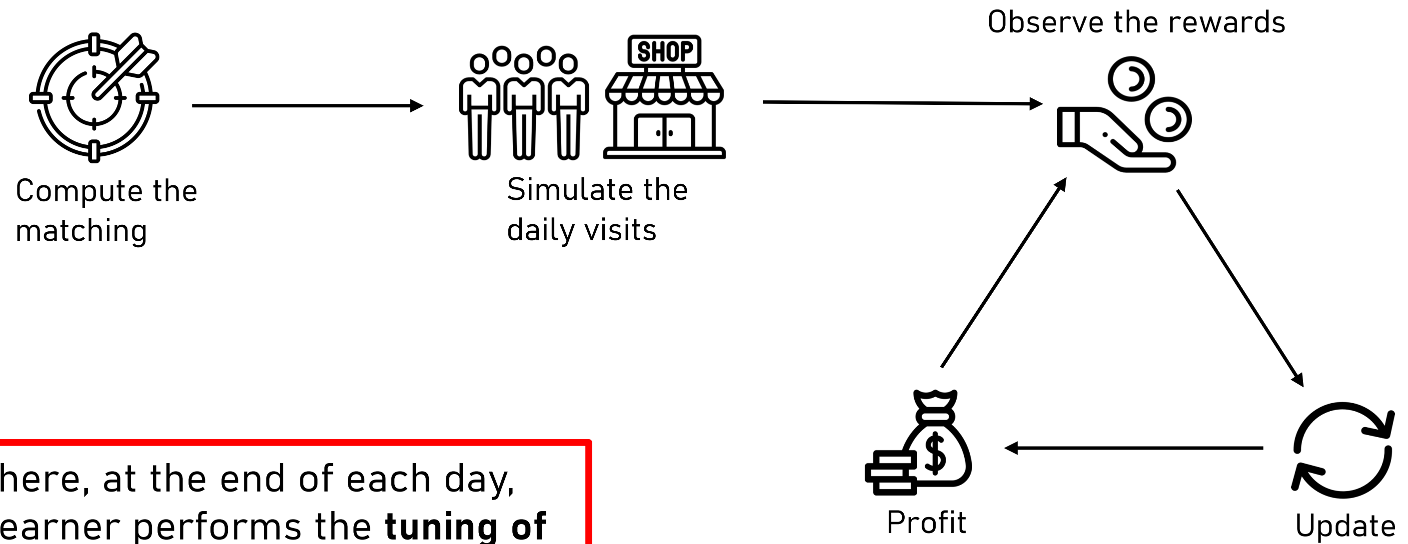


Algorithm core idea



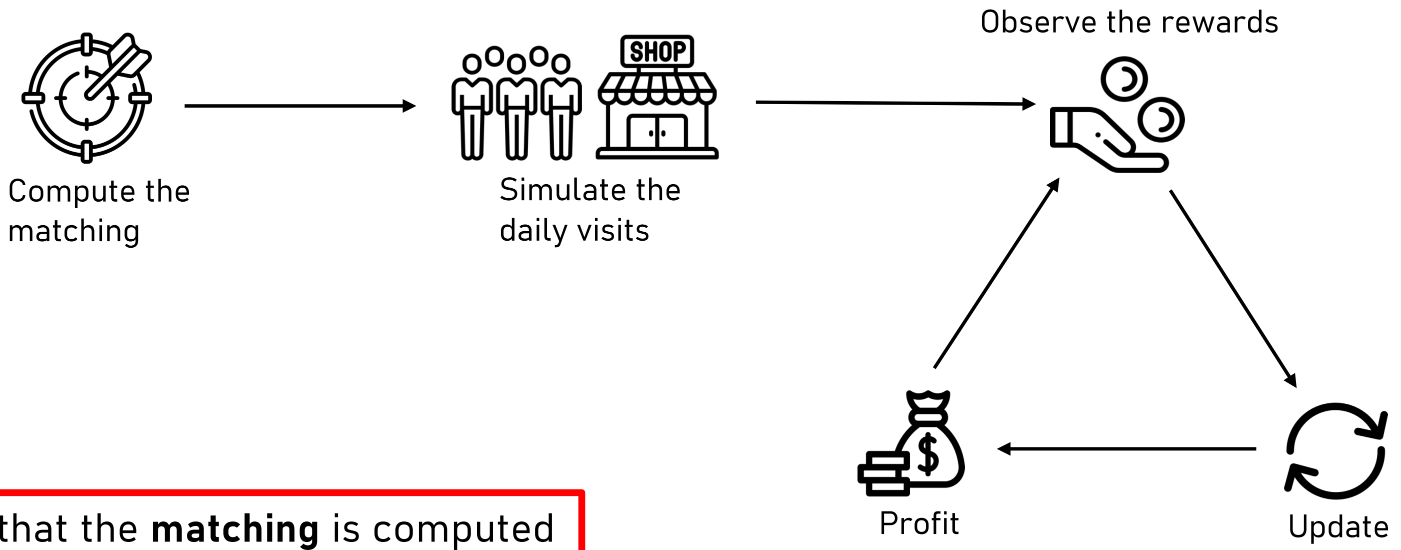
$$\boxed{r_1(c) * (\, m_1 + r_2(c,p) * m_2(p) \,)}$$

Algorithm core idea



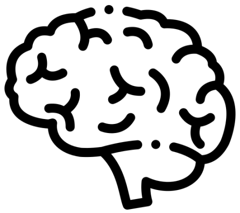
Also here, at the end of each day, the Learner performs the **tuning of the estimates over the number of customers** that will visit the shop the next day

Algorithm core idea



Note that the **matching** is computed at the beginning of each day, when the new expectations about the customers are available

Matching rationale



- The matching problem is associated to a matrix whose dimension is

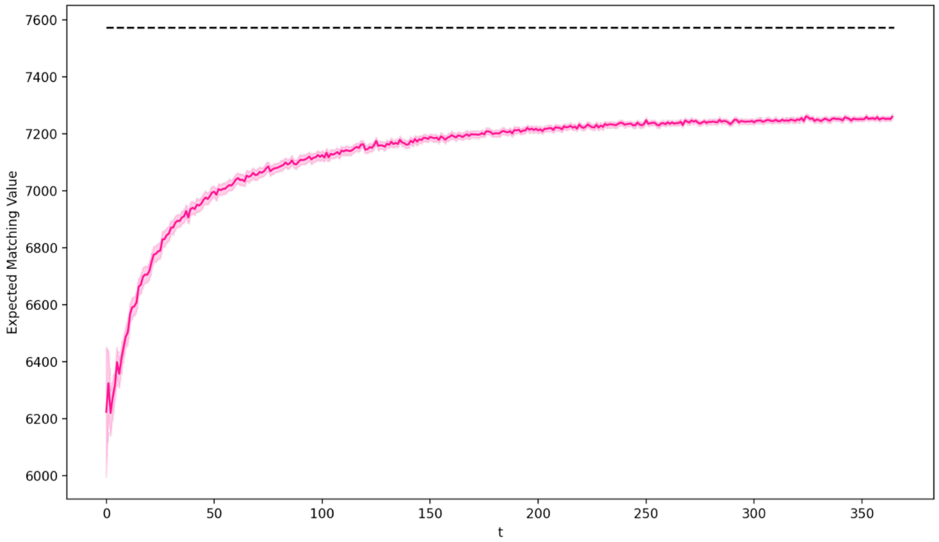
$$number\_of\_customers \times number\_of\_promos$$

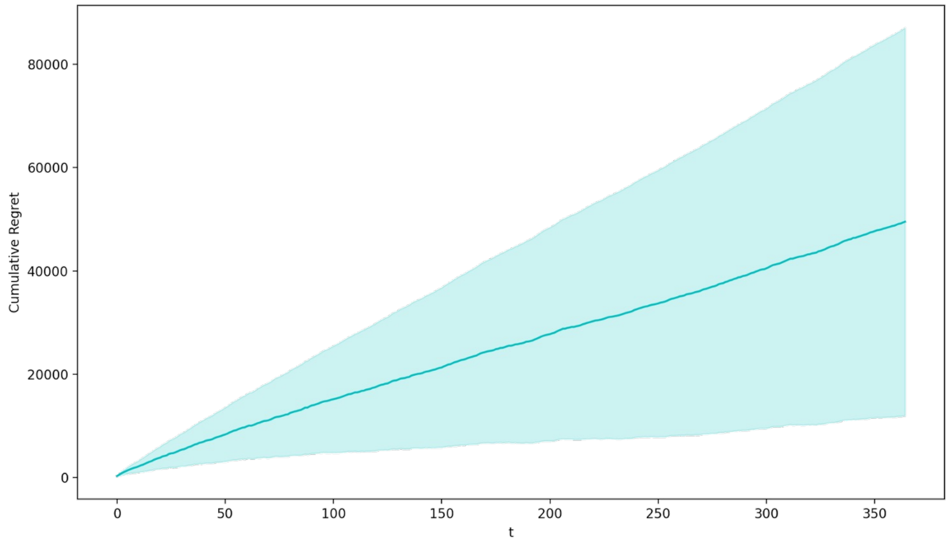
- The elements of the matrix are the **expected profits** related to the considered couple of prices:

$$profit = conversion\_rate\_1 * (margin\_1 + conversion\_rate\_2 * margin\_2)$$

- The conversion rates are estimated sampling from a Beta distribution, in a TS-like approach.

Results – step 5






FULL ONLINE PROBLEM

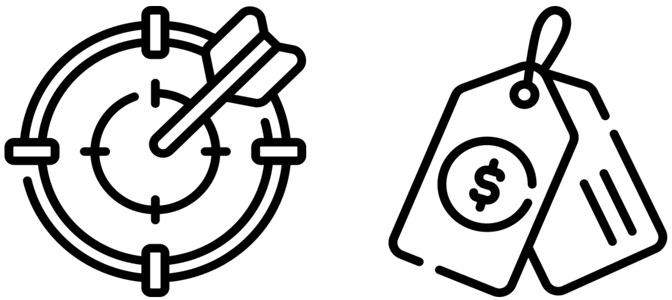
Stationary context

ASSUMPTIONS:

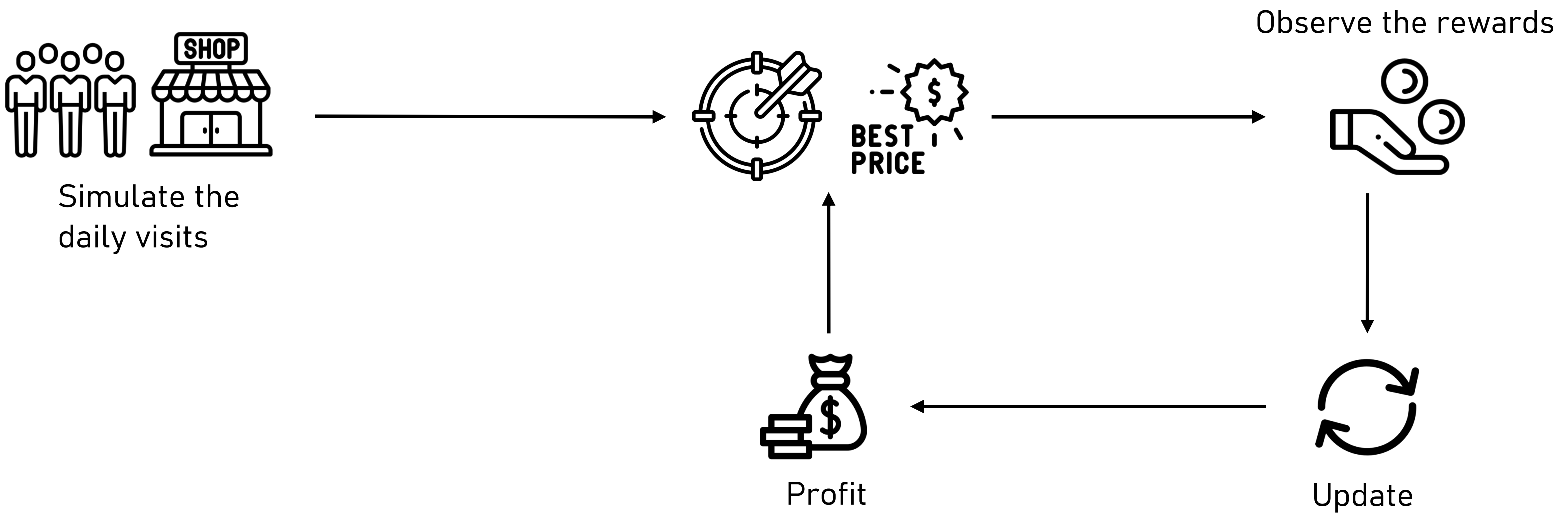
- The prices of both items are the same for each different class of customers.
- The **conversion rates** over the prices of both items are **stationary**.

OBJECTIVES:

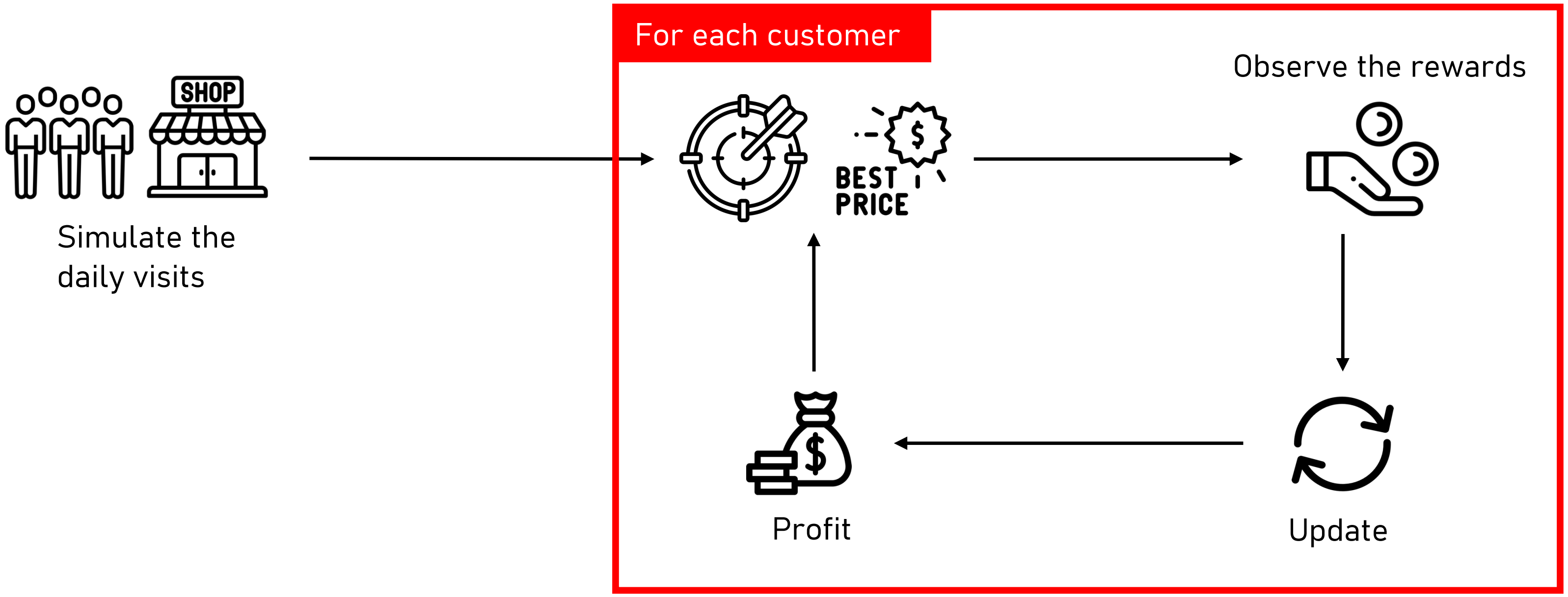
- Find the optimal price for both items.
- Optimize the assignment of promos.



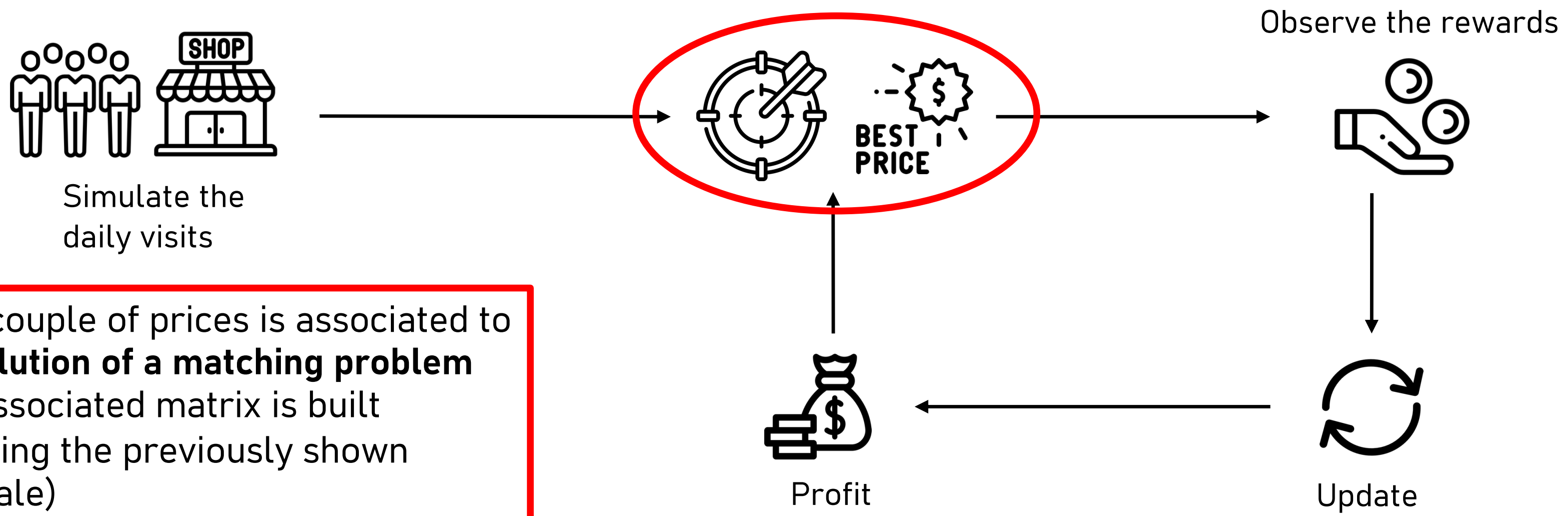
Algorithm core idea



Algorithm core idea

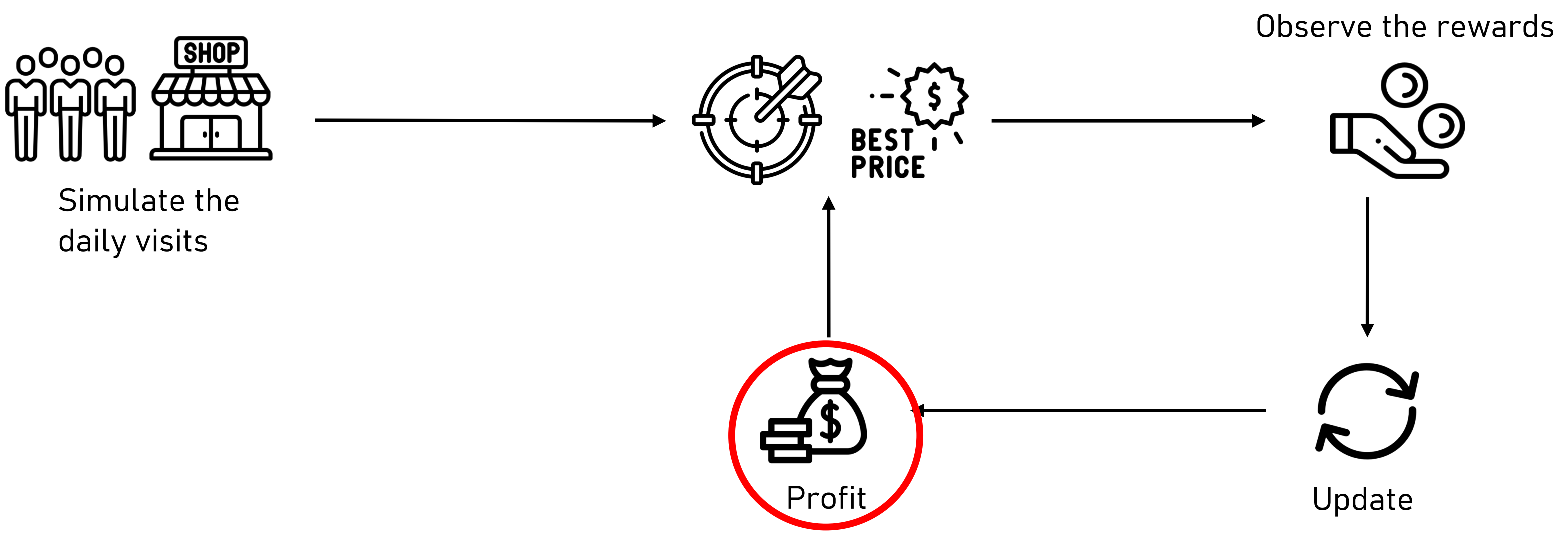


Algorithm core idea



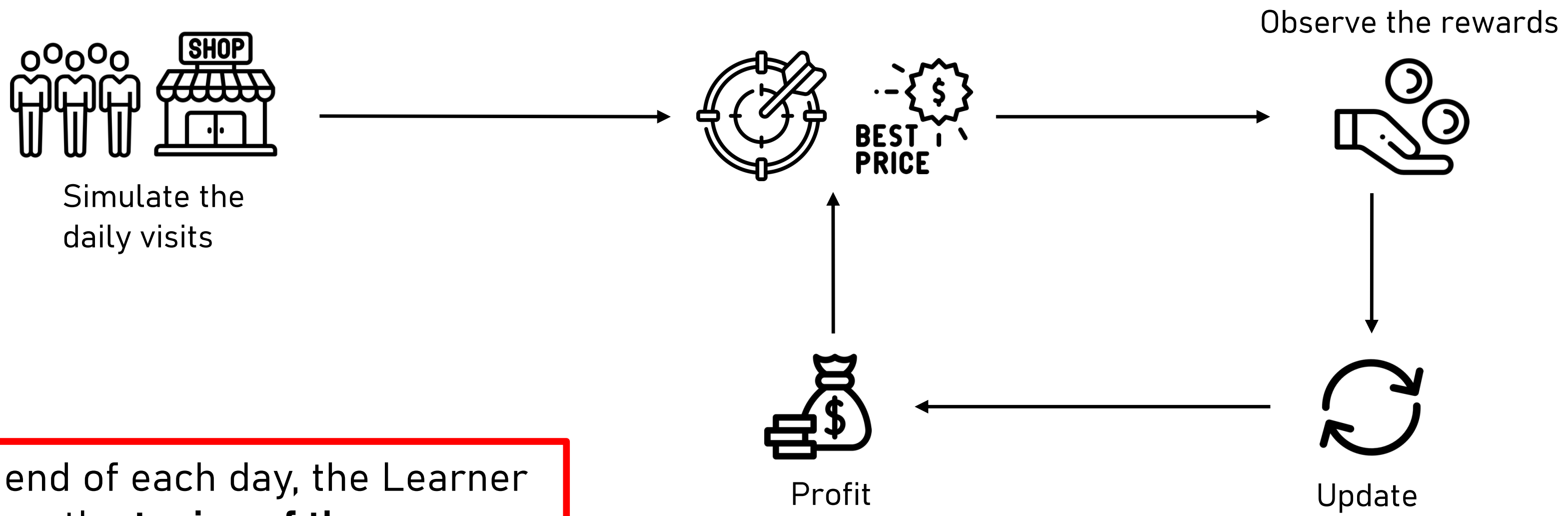
- Each couple of prices is associated to the **solution of a matching problem** (the associated matrix is built following the previously shown rationale)
- The best couple is the one that **maximizes the matching value**
- The assignment of promos is coherent with the matching associated to the best couple of prices

Algorithm core idea



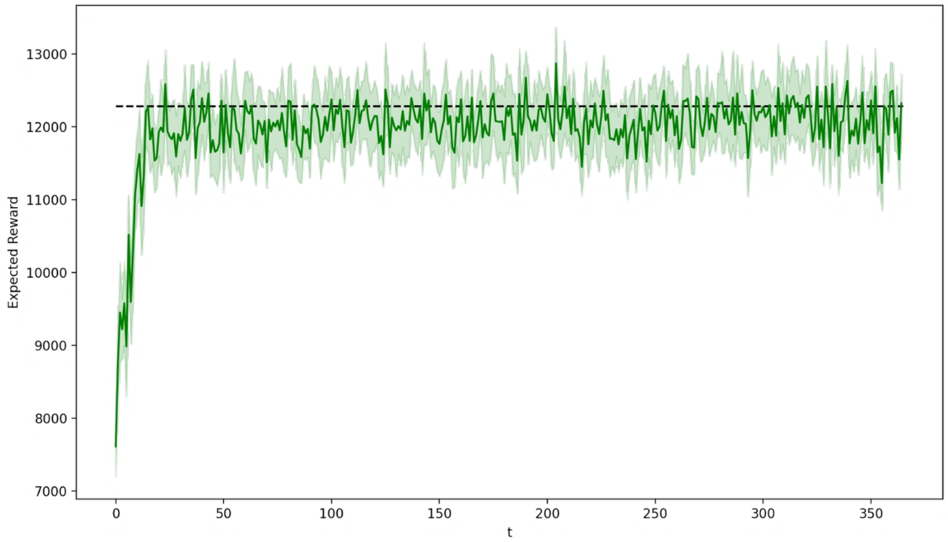
$$\boxed{r_1(c) * (\, m_1 + r_2(c,p) * m_2(p)\,)}$$

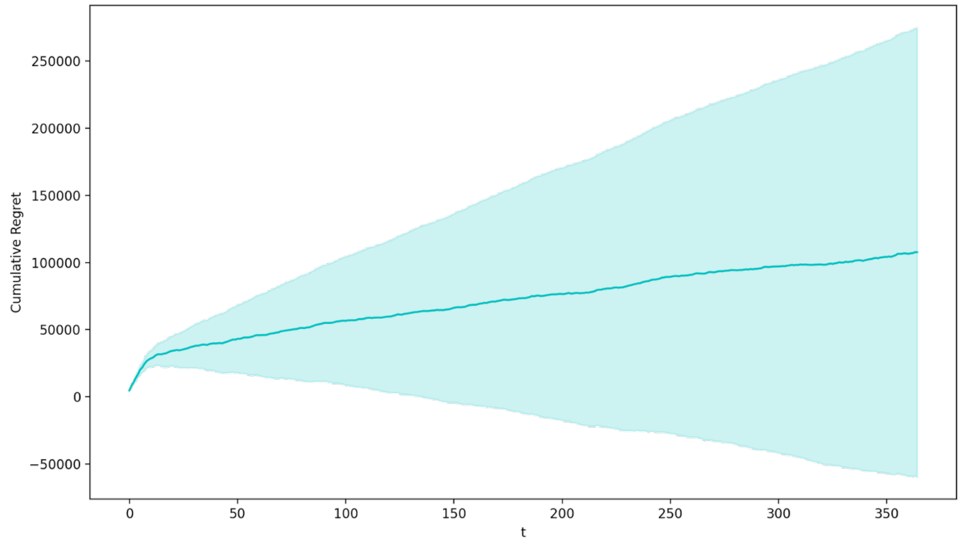
Algorithm core idea



At the end of each day, the Learner performs the **tuning of the estimates over the number of customers** that will visit the shop the next day

Results – step 6





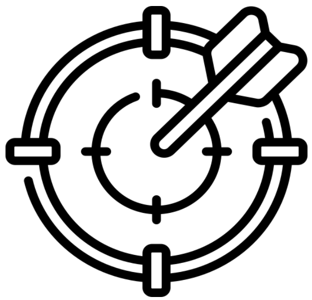
Non-stationary context

ASSUMPTIONS:

- The prices of both items are the same for each different class of customers.
- The **conversion rates** over the prices of both items are **no more stationary** and varies with respect to the seasons of the year.

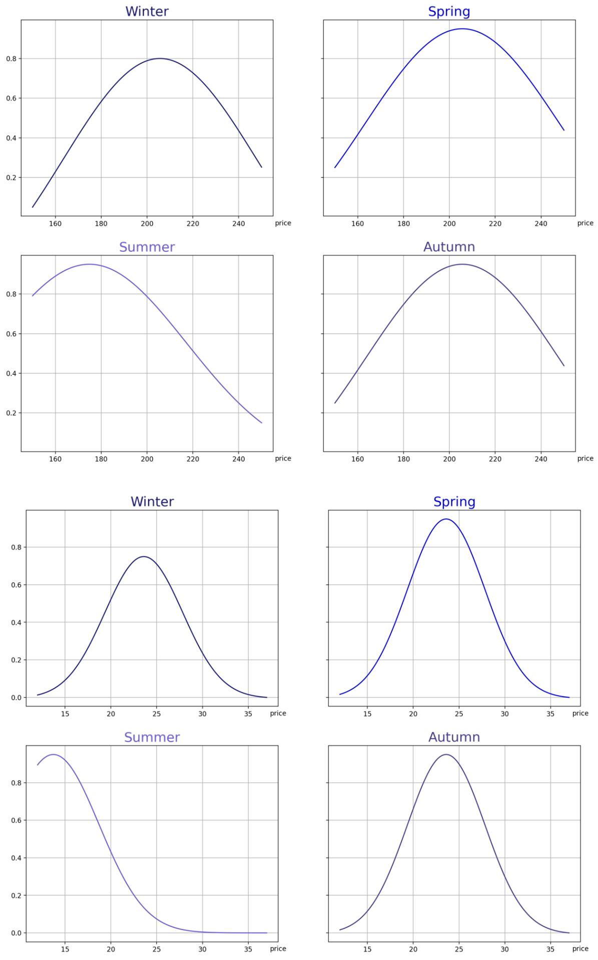
OBJECTIVES:

- Find the optimal price for both items.
- Optimize the assignment of promos.

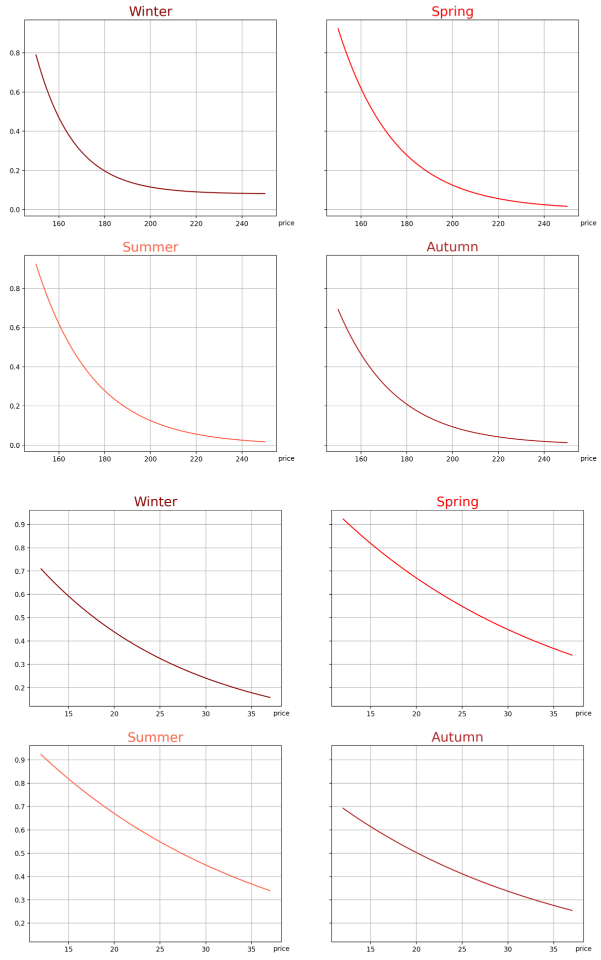


Non-stationary conversion rates - Juniors

Junior Professionals

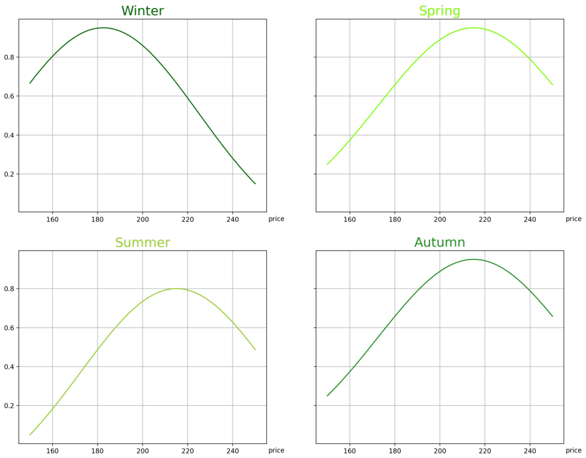


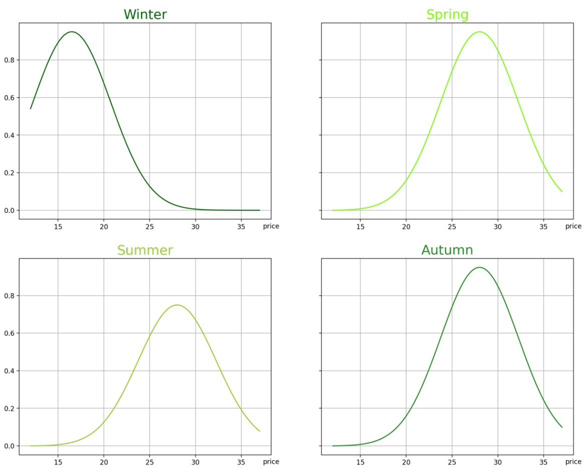
Junior Amateurs



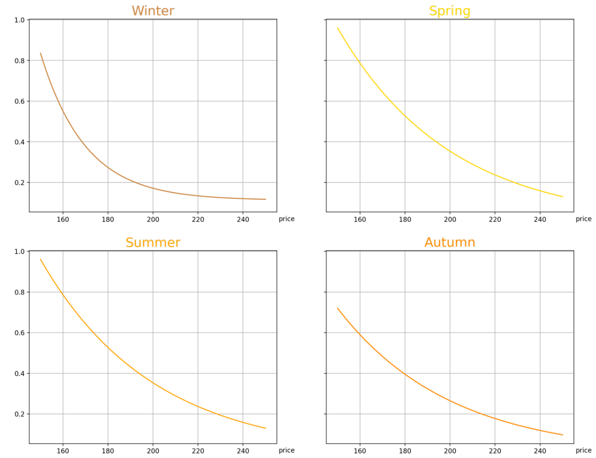
Non-stationary conversion rates - Seniors

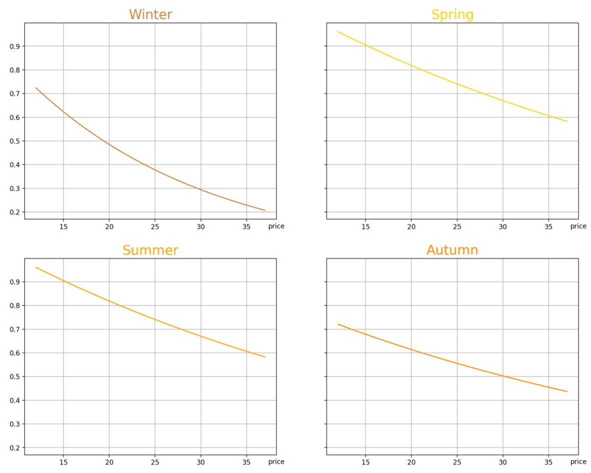
Senior Professionals





Senior Amateurs

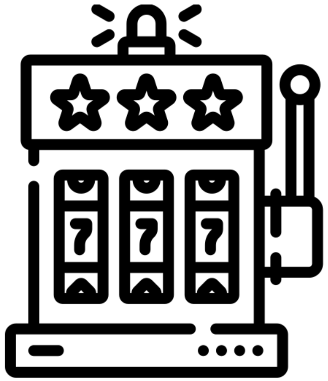




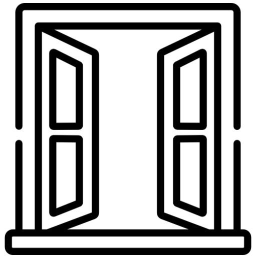
Solution approaches

The algorithm core idea is left unchanged from the stationary case, but, in order to handle the seasonality phenomenon, two different approaches have been followed:

- Sliding-Window Thompson Sampling
- Change-Detection UCB



SW Thompson Sampling approach



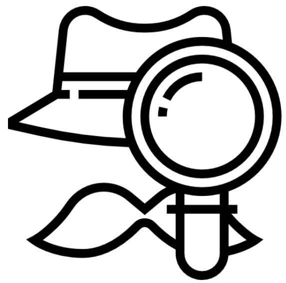
- As usual, the conversion rates are estimated **sampling** from Beta distributions.
- The difference is that each round, a fixed number of the **latest observed rewards** is used to update those distributions. That quantity is:

$$n_{rewards} = window\_size * expected\_daily\_customers$$

$$window\_size = \sqrt{T}$$

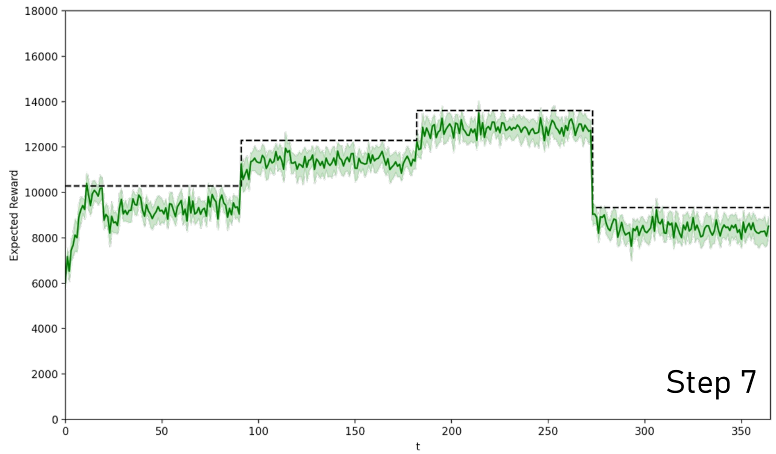
- The matrix used for the matching contains the values of the **expected profits**.

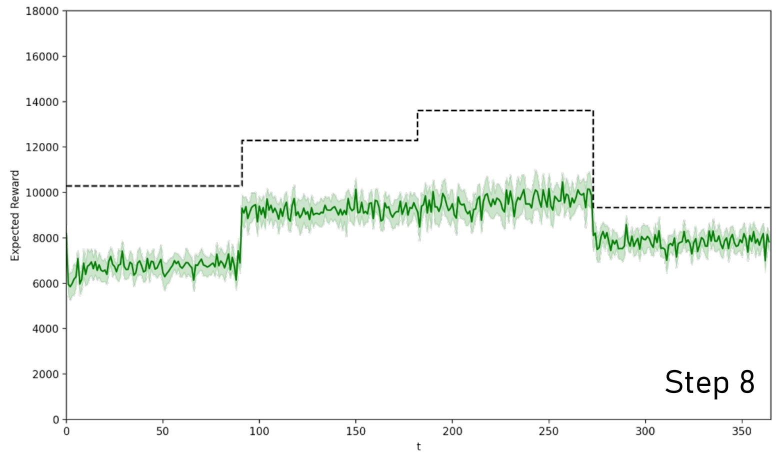
CD UCB approach

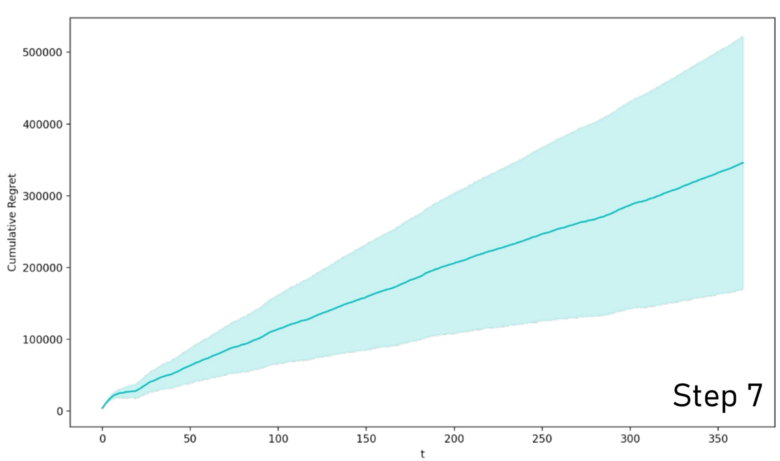


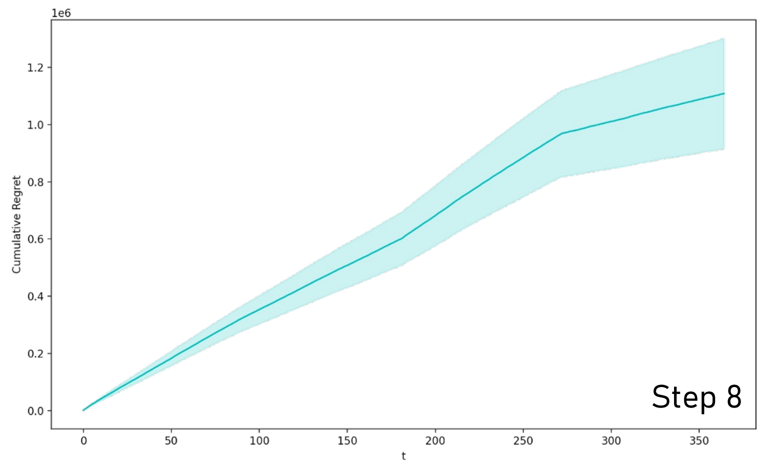
- The elements composing the matrix, associated with the **matching problem**, are no longer computed with the conversion rates sampled by a Beta distribution, but they are computed as **upper confidence bounds** over the *normalized profits.*
- A *normalized profit* is associated with a couple *(class_of_customer, assigned_promo)*, and it is the quantity observed in order to detect abrupt changes in customers behaviours.
- A **detection** is raised when the cumulative sum of the difference between the observed samples and a reference value is larger than a fixed threshold.

Results – step 7 and step 8










Thank you for the kind attention