Data Intelligence Applications
Pricing and Matching

Ivan Cavadini (941927)
Simone Marforio (944320)
Nicolò Molinari (942404)

2020/2021

POLITECNICO
MILANO 1863

Contents

Introduction

Scenario

Consider the scenario in which a shop has a number of 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.

Environment

Imagine two items (referred to as first and second items; for each item we have an infinite number of units) and four customers' classes. The daily number of customers of each class is described by a potentially different (truncated) Gaussian probability distribution. Each class is also associated with a potentially different conversion rate returning the probability that the user will buy the first item at a given price.

Once a buyer has bought the item, she/he can decide to buy the second item that can be or not promoted. There are four different promos P0, P1, P2, P3, each corresponding to a different level of discount. P0 corresponds to no discount. Given the total number of customers, the business unit of the shop decides the number of promos as a fraction of the total number of the daily customers and is fixed (use two different settings in your experiments that you are free to choose). Each customers' class is also associated with a potentially different conversion rate returning the probability that the user will buy the second item at a given price after she/he has bought the first. The promos will affect the conversion rate as they actually reduce the price. Every price available is associated with a margin obtained by the sale that is known beforehand. This holds both for the first and the second item. The conversion rates will change during time according to some phases due to, e.g., seasonality.

Steps

1. Provide a mathematical formulation of the problem in the case in which the daily optimization is performed using the average number of customers per class. Provide an algorithm to find the optimal solution in the offline case in which all the parameters are known. Then, during the day when customers arrive, the shop uses a randomized approach to assure that a fraction of the customers of a given class gets a specified promo according to the optimal solution. For instance, at the optimal solution, a specific fraction of the customers

of the first class gets P0, another fraction P1, and so on. These fractions will be used as probabilities during the day.

2. Consider the online learning version of the above optimization problem, identify the random variables, and choose a model for them when each round corresponds to a single day. Consider a time horizon of one year.

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 first item. Assume that the number of users per class is known as well as the conversion rate associated with the second item. Also assume that the prices are the same for all: the classes (assume the same in the following) and that the conversion rates do not change unless specified differently below. Adopt both an upper-confidence bound approach and a Thompson-sampling approach and compare their performance.

4. Do the same as Step 3 when instead the conversion rate associated with the second item is not known. Also assume that the number of customers per class is not known.

5. Consider the case in which prices are fixed, but the assignment of promos to users need to be optimized by using an assignment algorithm. All the parameters need to be learnt.

6. Consider the general case in which the shop needs to optimize the prices and the assignment of promos to the customers in the case all the parameters need to be learnt.

7. Do the same as Step 6 when the conversion rates are not stationary. Adopt a sliding-window approach.

8. Do the same as Step 6 when the conversion rates are not stationary. Adopt a change-detection test approach.

Context Modeling

We have considered a ski shop that sells racing skis as first item and racing ski helmets as second item. The optimization problem have a time horizon of one year, splitted in three seasons that change the conversion rates of the two items. Customers are splitted into four different categories that define their purchasing behavior (conversion rates), according to the season and the price of the item.

Items	Racing Skis Racing Ski Helmet	Professional racing skis Professional racing skis helmet
Customer categories	Sport addicted Gifter Worried Amateur	Who loves and practices ski frequently Who wants to give away the both items Who pays a lot of attention to the price of the items Who sometimes practices ski
seasons	Spring-Summer Autumn Winter	Buyers are not tempted to spend a lot, ski season is far away Buyers are willing to spend in anticipation of the arrival of the ski season Ski season has begun, those who have not yet bought the equipment have hurried to buy it so as not to waste the season

Assumption

- Seasonality is taken into accout only for the 7^{th}, 8^{th} requests, while for all the other, the seasonality of the products is not considered and the conversion rates remain static. For this requests the default season is the first one, in our context called Spring.
- In our mathematical formulation, for the total reward maximization problem, we consider the production cost of both the items equal to zero.

Formal Model

Variables definition

- i= user category
- j = promotional discount: $\mathrm{P}_0 = 0\%$, $\mathrm{P}_1 = 10\%$, $\mathrm{P}_2 = 20\%$, $\mathrm{P}_3 = 30\%$
- $p1$ = full price of the first item (*Racing skis*)
- $p2$ = full price of second item (*Racing ski helmet*)
- $p2_j$ = price of the *Racing ski helmet* when applied the promo j

- $c1$ = production cost of *Racing skis* = 0
- $c2$ = production cost of *Racing ski helmet* = 0
- $q1_i(p1)$ = conversion rate for user category i, for *Racing skis* sold at the price $p1$
- $q2_i(p2)$ = conversion rate for user category i, for *Racing ski helmet* sold at price the $p2$
- $s_{ji}(p2)$ = discounted price of *Racing ski helmet*, for user category i, according to promo discount j
- d_{ij} = amount of promo j distributed to user category i
- d_{max} = maximum number of promos to be to distributed ($\#\mathrm{P}_1 + \#\mathrm{P}_2 + \#\mathrm{P}_3$)
- $avgCustomer_i$ = average number of customers for category i

Formulation of elaborated variables

- $p1 * q1_i(p1) * avgCustomer_i$ = revenue for the sale of *Racing skis* at price $p1$ to user category i
- $s_{ji}(p2) * q2_i(s_{ji}(p2)) * d_{ij} * avgCustomer_i$ = revenue for the sale of *Racing ski helmet* at the discounted price $p2$, according to the user-promo assignement
- $(p1 * q1_i(p1) - c1 * q1_i(p1)) * avgCustomer_i$ = revenue for the sale of *Racing skis* taking into account the production cost $c1$
- $(q2_i(p2) * (s_{ji}(p2) * q2_i(s_{ji}(p2)) * d_{ij} - q2_i(s_{ji}(p2))) * c2) * avgCustomer_i$ = revenue for the sale of *Racing ski helmet* taking into account the production cost $c2$

Objective Function

We have a maximization problem with the following objective function:

$$\max\left(\sum_{i=0,j=0}^{i=4,j=4}[(p1*q1_i(p1)-c1*q1_i(p1)+q2_i(p2)(s_{ji}(p2)*q2_i(s_{ji}(p2))*d_{ij}-q2_i(s_{ji}(p2))*c2))*avgCustomer_i]\right)$$

s.t: $\forall j > 0 : [\sum_{i=0}^{i=4} d_{ij}] = d_{max}$

We have fixed the full prices of the two items: $p1$, $p2$. We retrieve the discounted prices of $p2$, applying the promos j.

We know: the average number of customers per category i, $avgCustomer_i$, the conversion rate for both products ($q1_i(p1)$, $q2_i(p2)$) and the maximum number of promos to distribute (d_{max}).

As assumption the production costs of the two items is zero ($c1 = 0$, $c2 = 0$).

It is possible to retrieve the total revenue for *Racing skis* as the product between the full price of the first item, the conversion rate for the considered user category and the average number of customers for that category: $(p1 * q1_i(p1) * avgCustomer_i)$.

For the second item the calculation of the reward is the same except for the fact that the product is buyed only if also the first one is purchased (so we multiply also the conversion rate of the first item) and the considered price have to be discounted according to the assigned promotion.

The solution of our optimization problem consists in the distribution of the fraction of promo codes among the user categories.

Offline problem - designed algorithm

In this scenario we have to find the optimal solution in an offline manner (a solution to our maximization problem knowing all the parameters), considering the constraints that the shop uses a randomized approach to assure that a fraction of the customers of a given category gets a specified promotion according to the optimal solution.

We achieve the solution of the problem using a matching approach. To reach the optimal solution we have used an iterative approach: we build a matrix category-promotion containing the mean expected rewards for every couple, calculated as the conversion rate of the *Racing ski helmet* multiplied with its discounted price. The goal is to obtain, for each category-promotion couple the fraction of customers that will receive this discount.

We exploit this matrix to perform the matching between the customer classes and the promo types, in order to maximize the total reward.

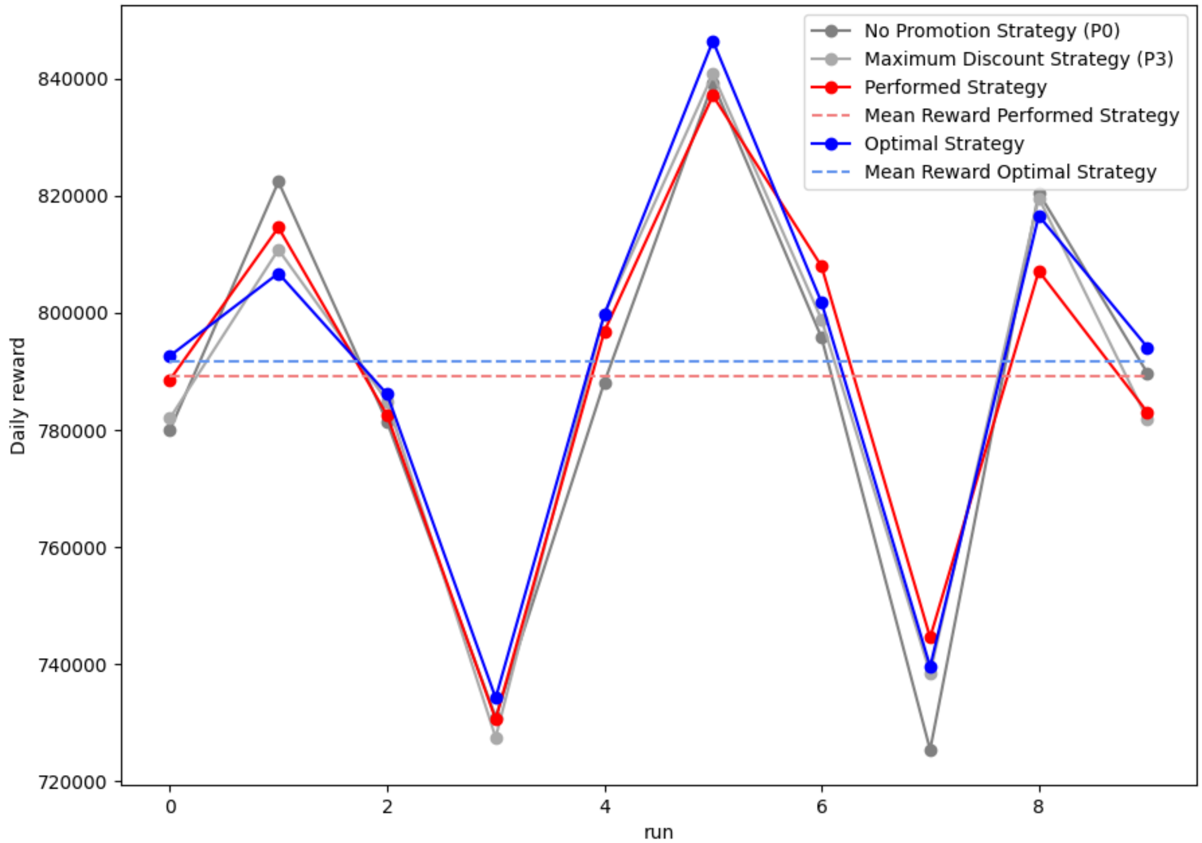
We select the best reward for every class, for four times, retrieving, at every iteration, the four best combination of category-promotion and assigning an infinite weight to the obtained sub-optimal matching.

Every matching is represented by a reward configuration that maximize the total reward. Every iteration is weighted and represent a different goodnesses of the solution, the first is the best, the last is the worst.

Through the sub-optimal matchings, we have retrieved the fractions of different promos to assign to every customer categories, based on the proportional weight of the previous sub-optimal matching.
The proportions retrieved, are normalized category per category.

PESEUDOCODICE

```
Optimal solution: probability distribution of promos per class (rows: class, col: promos)
[[0.12 0.28 0.52 0.08]
 [0.06 0.53 0.15 0.26]
 [0.52 0.07 0.28 0.14]
 [0.17 0.09 0.05 0.69]]
```



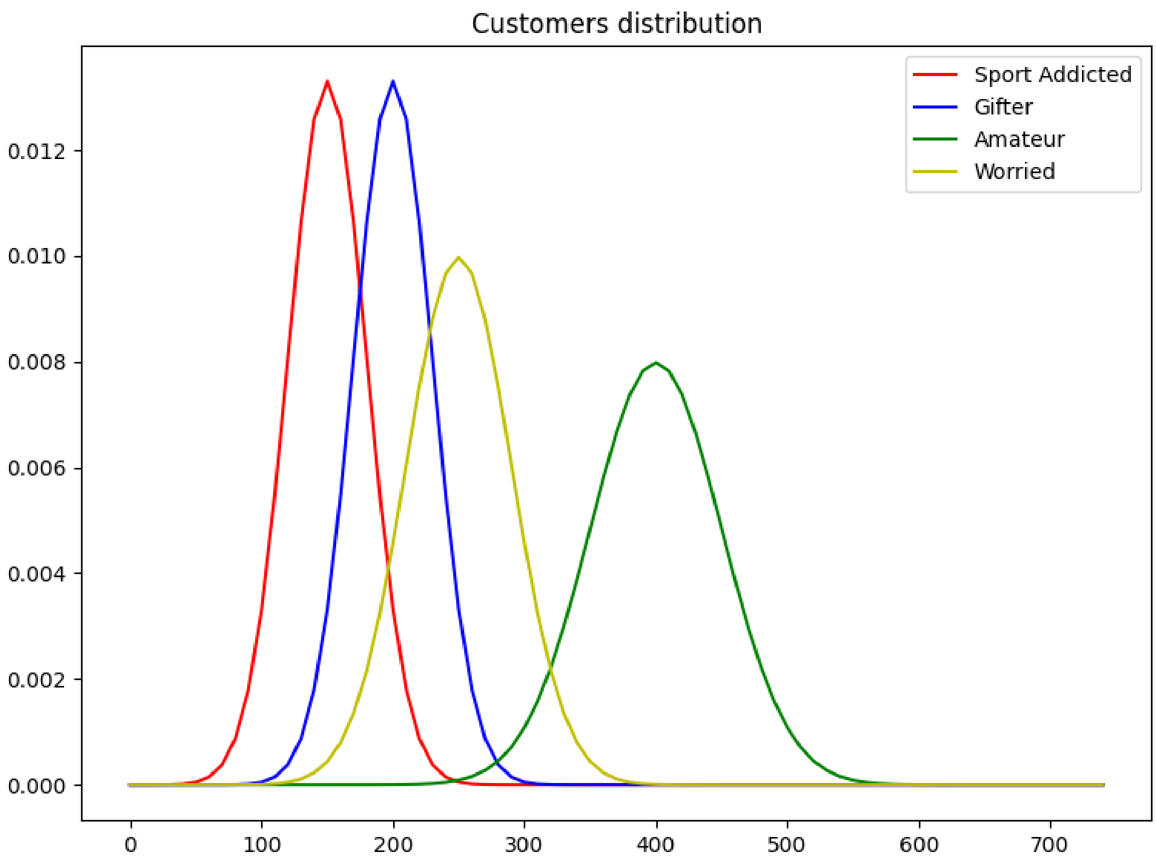
Random Varaibles

For the online problem have identified and modeled the following radom variables

Daily customers distribution

We modeled the daily customers distribution as a **Gaussian Distribution**, with a normalizing factor of 1000 daily customers. Average and variance of the Gaussian distribution are reported in the table below.

Customer Category	Average	Variance
Sport addicted	0.15	0.03
Gifter	0.20	0.03
Worried	0.40	0.05
Amateur	0.25	0.04

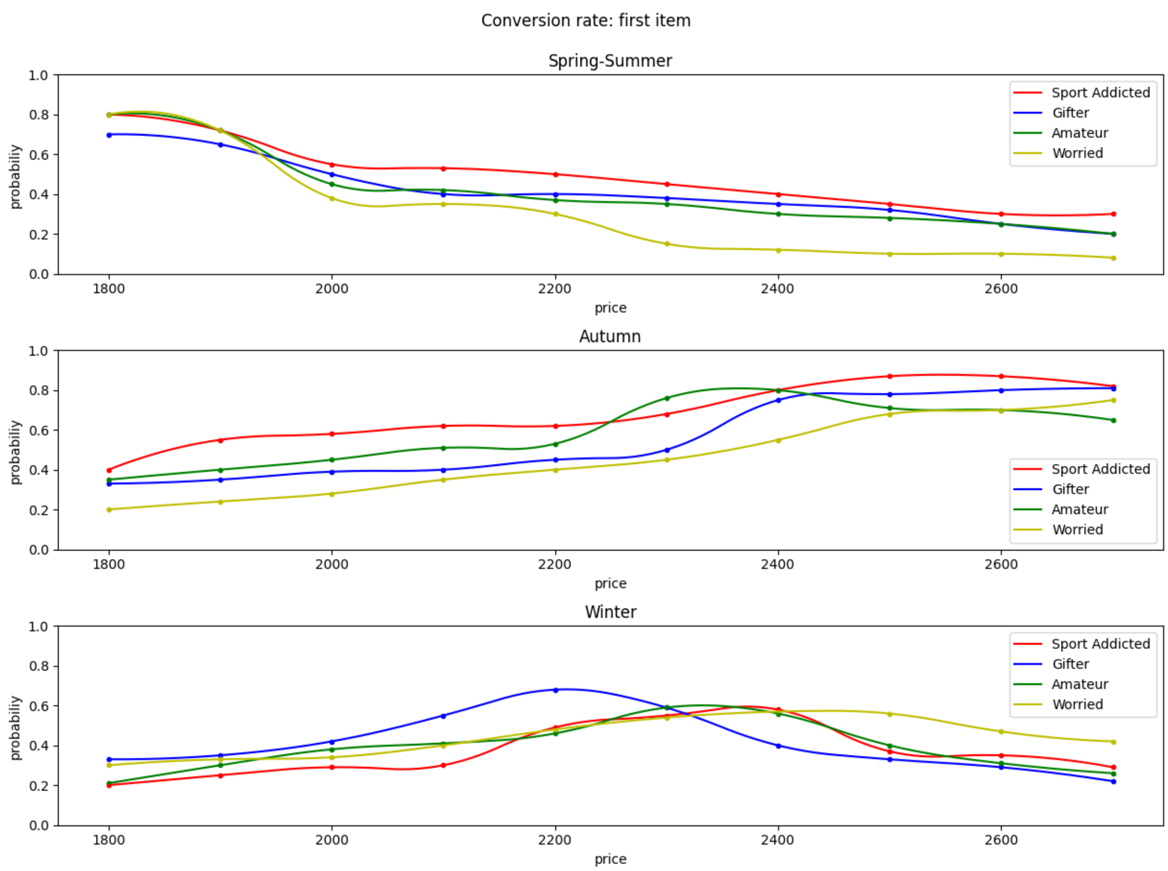


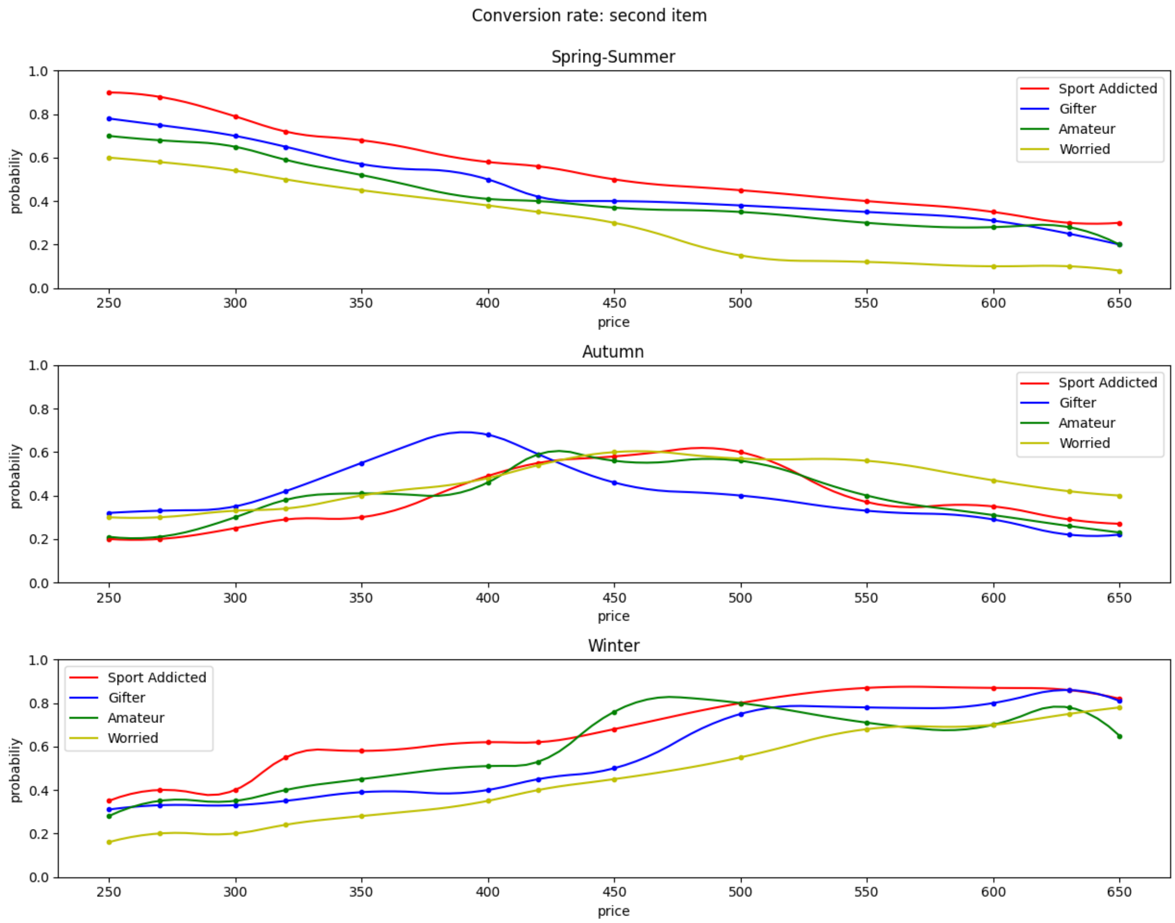
Conversions rates

The odds of buying one item are modelled with a **Bernulli Distribution** different for every user category and item price.

- Probability of buying *Racing skis*: Bernoulli $\tilde{0}$,1
- Probability of buying *Racing ski helmet*: Bernoulli $\tilde{0}$,1

The following graphs are the demand curves of the two items: the first three graphs are the demand curves of the first item, associated to the three seasonality (Spring-Summer, Autumn, Winter), while the last three graphs are the demand curves of the second item with its respective seasonality.





Online Approach

Our general approach for the online problem is to simulate, day by day the shop, generating the customers and emulating their behavior, collecting the results and, according to the considered scenario and constraints, provide an optimal solution that maximise the reward.

Every day we retrieve the daily customer distribution per class using the previously presented random variable that model the daily customer distribution. Randomly we simulate the entry of a new customer, of which we know the category of belonging, in the shop. With an online approach we select the best price to be prosed to the client, in order to maximize the reward. The purchase is simulated with the previously presented random variable with a Bernulli distribution. The second item is proposed to the client only if the first has been purchased. The price at which it

is proposed is retrieved with an online matching approach that try to sugget which is the best discout to apply to the user in oreder to maximize the reward.
This procedure is repeated for all clients during the entire time horizon of 356 days.

Experiment Context

Online pricing for first item

Problem explaination The problem is to learn the optimal price of the first item, especially comparing the adoption of a Thompson-sampling approach and an upper-confidence bound approach, in the following scenario:

- Assignment of promos is fixed
- Price of second item is fixed
- Number of users per class is known
- Conversion rate associated with second item is known
- Prices are the same for all the classe
- Conversion rates do not change

Strategy The problem described is a combinatorial bandit problem,which is a decision making problem in which the decision maker selects one single arm in each round, and observes a realization of the corresponding unknown reward distribution. Each decision is based on past decisions and observed rewards. The objective is to maximize the expected cumulative reward over some time horizon by balancing exploitation and exploration. We can solve it, through an online algorithm, where only for the feasible solutions we will have precise estimations. We implement a script that works on a time horizon of ten days, repeating the experiment ten times. We simulate a random arrival of the customers, providing to them the prices given by the two learners (UCB and TS) for the first item and simulating the purchase of it. In case the customer buys the first item, we retrieve the reward of the second item as the product between the conversion rate of that customer and the respective discounted price We calculate the expected reward and we update the learners with the results.

Upper-Confidence Bound (UCB1) Main idea:

- Every arm is associated with an upper confidence bound
- At every round, the arm with the highest upper confidence bound is chosen
- After having observed the realization of the reward of the arm, the upper confidence bound is updated

Notation:

- t time
- A set of arms
- a arm
- a_t arm played at time t
- a^* optimal arm
- X_a random variable (bernoulli) associated to arm a
- μ_a expected value of random variable X_a
- $x_{a,t}$ realization of rv X_a at time t
- x_a realizations of X_a
- $\bar{x}_a$ empirical mean of x_a
- $n_a(t)$ number of samples of arm a at time t

Pseudocode

1. Play once each arm $a \in A$
2. At every time t play arm a such that:
3. $a_t \leftarrow \arg\max_a \left\{ \left[\bar{x}_a + \sqrt{\frac{2log(t)}{n_a(t-1)}} \right] \times a \right\}$

Thompson Sampling (TS) Main idea:

- For every arm, we have a prior on its expected value
- In the case the arms' rewards are Bernoulli distribution, the priors are Beta distributions
- Notice that, with the opportune parameters, a Beta distribution is a uniform distribution
- For every arm, we draw a sample according to the corresponding Beta
- We choose the arm with the best sample
- We update the Beta distribution of the chosen arm according the observed realization

Notation (in addition to classical UCB):

- $\mathbb{P}(\mu_a = \theta_a)$ prior of the expected value of X_a
- θ_a variable of $\mathbb{P}(\mu_a = \theta_a)$
- $(\alpha_{a_t}, \beta_{a_t})$ parameters of the beta distribution $P(\mu_a = \theta_a)$

Pseudocode

1. At every time t for every arm a:
$\tilde{\theta_a} \leftarrow Sample(\mathbb{P}(\mu_a = \theta_a))$

2. At every time t play arm a_t such that
$a_t \leftarrow \arg\max_a \left\{\tilde{\theta_a} \times a\right\}$

3. Update beta distribution of arm a_t
$(\alpha_{a_t}, \beta_{a_t}) \leftarrow (\alpha_{a_t}, \beta_{a_t}) + (x_{a_t,t}, 1 - x_{a_t,t})$

Results

Considerations

Online pricing for first item with purchase simulation

Problem explaination The goal is the same of the previous problem, but there are few changes in the scenario:

- The conversion rates associated to the second item are not known
- The number of customers per class is not known

Strategy The script that we have implemented is similar to the previous one: it works on a time horizon of ten days, repeating the experiment ten times. We simulate a random arrival of the customers, providing to them the prices given by the two learners (UCB and TS) for the first item and simulating the purchase of it. In case the customer buys the first item, instead of retrieving the reward of the second item as the product between the conversion rate of that customer and the respective discounted price, we simulate the purchase of it. We calculate the expected reward and we update the learners with the results. The algorihm that we have used are the same of the previous one: UCB and TS.

Results

Considerations

Matching problem: promo assignment

Problem explaination The problem requires to optimize the assignment of the problem using an assignment algorithm, in the following scenario:

- The prices are fixed
- All the parameter need to be learnt

Strategy DIRE CHE BANDIT PROBLEM è e RIVEDERE SE QUELLI PRECEDENTI SONO GIUSTI. AGGIUNGERE UN PO' DI TEORIA We simulate the randomly arrival of the customers and the purchase of the first item. In case of purchase of the first item, we retrieve the optimal matching for the user categories from the learner (UCB matching) and we propose the second item to him, at the discounted price based on the matching suggested by the learner. The learner provides us a set of arms, so we cannot update it with the reward given by the single customer. We introduce two matrixes of shape (|User category |, |Type of promo|), every cell represents the matching between a specific customer category and a specific promo.The first matrix contains the sum of the rewards for that matching, the second contains the number of occurences that the matching has been chosen. At the beginning of the simulation, we use the first 1000 samples (customers) to initialize the two matrix, forcing all the possible combinations of matching. In this way we can update the learner, customer by customer, using the average reward for every matching.

UCB Matching

Pseudocode
At t, play a superarm a_t such that:
$a_t \leftarrow \arg\max_{\mathbf{a} \in M} \left\{ \sum_{a \in \mathbf{a}} \bar{x}_{a,t} + \sqrt{\frac{2log(t)}{n_a(t-1)}} \right\}$
where M is the set of matches.

Results

Considerations

Pricing and Matching problem

Problem explaination

Strategy

Results

Considerations

Seasonal Pricing and Matching problem: Sliding Window

Problem explaination

Strategy

Results

Considerations

Seasonal Pricing and Matching problem: Change Detection

Problem explaination

Strategy

Results

Considerations