POLITECNICO
MILANO 1863

Data Intelligence Applications: homework

d'Amato Francesco - 10560062 - 945814,
Frantuma Elia - 10567359 - 945729,
Fucci Tiziano - 10524029 - 946638

A.Y. 2020/2021

Table of contents

Chapter 1

Introduction

1.1 Scenario

Consider the scenario in which advertisement is used to attract users on an ecommerce website and the users, after the purchase of the first unit of a consumable item, will buy additional units of the same item in future. The goal is to find the best joint bidding and pricing strategy taking into account future purchases.

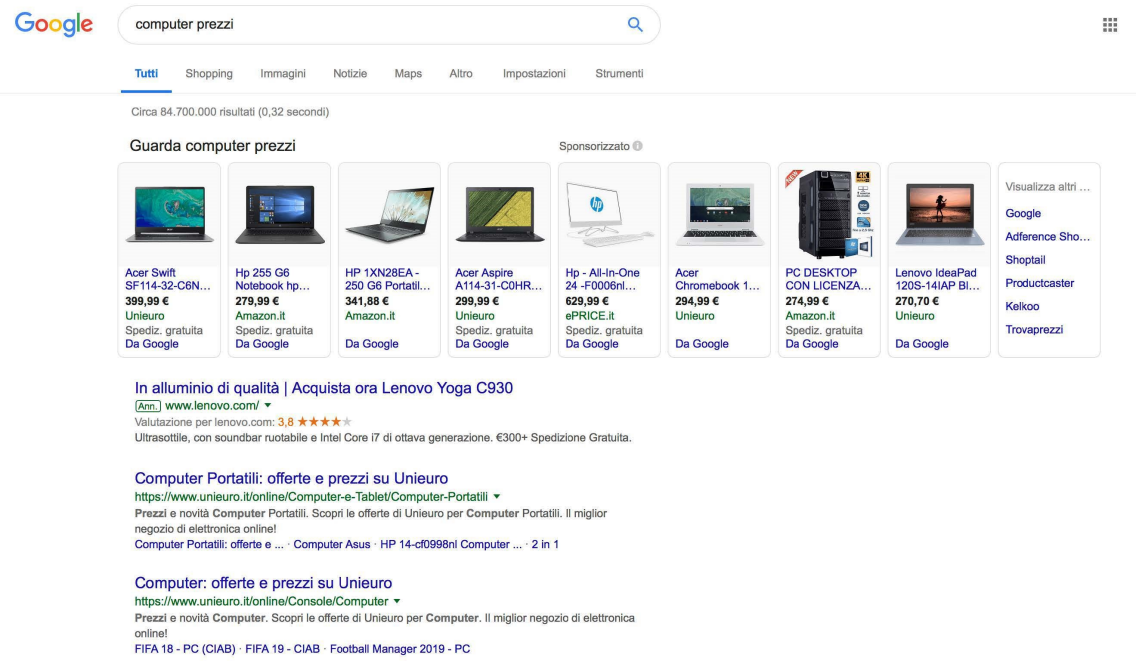


Figure 1.1: Advertising example

1.2 The product

The product we have chosen to simulate this advertising scenario is an energy drink. As we will say later, the first unit of product comes with a "dash button", to encourage the customer to buy it again and simulate the re-buy process. In our setting, it is assumed that the company providing the advertised dash buttons is launching its new consumer experience with the following offer: the button for the specific product is included with the first order of said product, and, for the following month, every restock of the product, when requested by using the dash button, will be guaranteed to be at the same price as the first order.

Figure 1.2: The sold product

Chapter 2

Environment

In this section we give a precise definition of the customer classes and their features, cost functions and distribution probabilities on which the model is based.

2.1 Customers classes

In the environment model we have three customer classes: C1, C2 and C3. They represent customers with different needs, age and tastes and thus different interest in buying the product. The following are the features used to infer the class to which a user belongs.

Feature A: active/sedentary (0|1)

Feature B: old/young (0|1)

Classes	old	young
active	Sportsman	Sportsman
sedentary	Retired man	Programmer

2.1.1 Class 1: the sportsman

The first class is composed by people who play sports or train regularly. They are interested in buying the energy drink to improve their performance.

Their conversion rate function is:

$$\mathcal{C}_1(p) = 1.4e^{-0.14p}$$

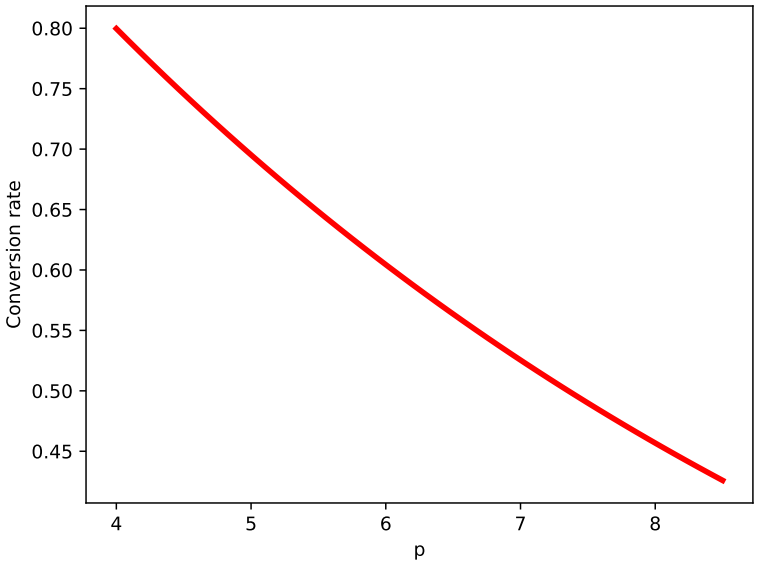


Figure 2.1: Conversion rate function of Class 1.

and the number of purchases following the first one is described by:

$$X \sim \mathcal{P}oisson(\lambda),$$

where:

$$\lambda = \frac{4.0}{\frac{p}{5} + 0.5}$$

2.1.2 Class 2: the retired man

The average members of this class buy the product just to enjoy its taste.

Their conversion rate function is:

$$\mathcal{C}_2(p) = -0.1 + 5e^{-0.4p}$$

Figure 2.2: Conversion rate function of Class 2.

and the number of purchases following the first one is described by:

$$X \sim \mathcal{P}oisson(\lambda),$$

where:

$$\lambda = \frac{2.0}{\frac{p}{5} + 0.5}$$

2.1.3 Class 3: the programmer

Programmers need to stay focused for a long time while at work, so they are interested in buying the product to work better and avoid introducing bugs in the code.

Their conversion rate function is:

$$\mathcal{C}_3(p) = \begin{cases} 0.8e^{-0.5(p-5.5)^2} & 4.0 \leq p < 6 \\ 20e^{-0.557p} & p \geq 6 \end{cases}$$

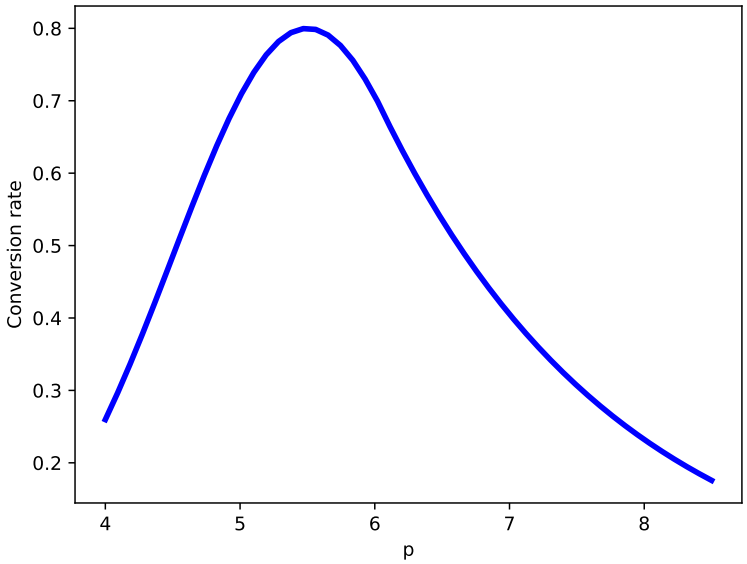


Figure 2.3: Conversion rate function of Class 3.

and the number of purchases following the first one is described by:

$$X \sim \mathcal{P}oisson(\lambda),$$

where:

$$\lambda = \frac{3.0}{\frac{p}{5} + 0.5}$$

2.2 Advertising

2.2.1 Auctions

In our model, we make the hypothesis of a GSP (Generalized Second Price) auction mechanism:

$$p_a = \frac{q_a}{q_{a+1}} v_{a+1} \left(\leq \frac{q_a}{q_a} v_a = v_a \right)$$

However, since we are not required to model the other auctionists, we make the following simplification:

$$p_a = v_a - |X|$$

where $X \sim \mathcal{N}(\mu, \sigma^2)$ having $\mu = \frac{v_a}{10}$, $\sigma^2 = 0.1$, that is assuming that we are generally paying for $\frac{9}{10}$ of our bid. The random variable p_a represents the stochastic cost per click.

2.2.2 Daily clicks of new users

The number of daily clicks of new users of class $\mathcal{C}_i$ is a random variable drawn from a Gaussian distribution:

$$X_i \sim \mathcal{N}(\mu_i,\, \sigma^2|b)$$

If the task requires to find the optimal bidding strategy, we model the influence of the bid on the number of daily clicks:

$$X_i \sim \mathcal{N}(\mu_{0,i} + \mu_{bid,i}(b),\, \sigma^2)$$

where $\mu_{0,i}$ is a fixed value and $\mu_{bid,i}(b)$ is an increment which grows monotonically with the bid.

2.2.3 Conversion rate functions

The conversion rate functions are defined for each class as the probability that the user of that class will buy the product at a given price. In particular, we have:

$$\mathcal{C}_1(p) = 1.4e^{-0.14p}$$

$$\mathcal{C}_2(p) = 0.1 + 6e^{-0.6p}$$

$$\mathcal{C}_3(p) = \begin{cases} 0.8e^{-0.5(p-5.5)^2} & 4.0 \leq p < 6 \\ 20e^{-0.557p} & p \geq 6 \end{cases}$$

$$\mathcal{C}_{agg}(p,b) = \frac{n_1(b)\mathcal{C}_1(p) + n_2(b)\mathcal{C}_2(p) + n_3(b)\mathcal{C}_3(p)}{n_1(b) + n_2(b) + n_3(b)}$$

where n_i is the population of Class i and the coefficients $n_i(b)$ are function of the bid. Since in our model $4.00 \leq p \leq 8.50$, $\mathcal{C}_i(p) \in [0,1]\ \forall i$.

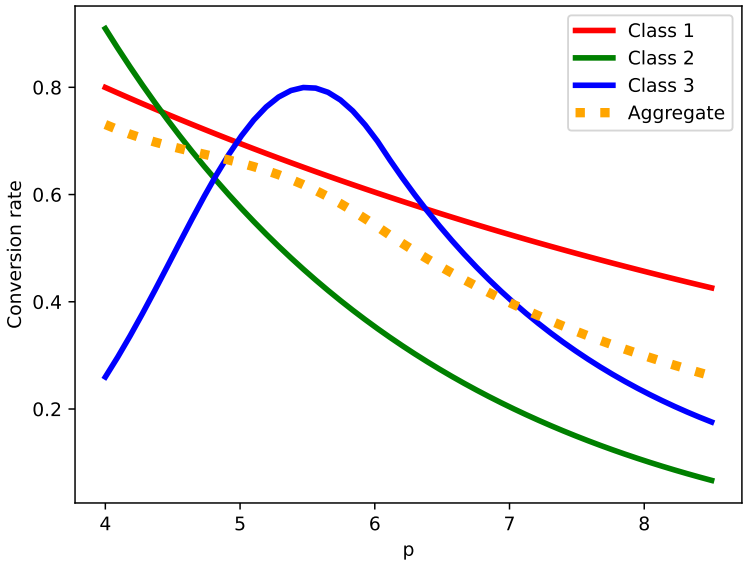


Figure 2.4: Conversion rate function of the aggregate Class, obtained for a fixed bid $b = 2,50€$.

2.3 Re-buy process

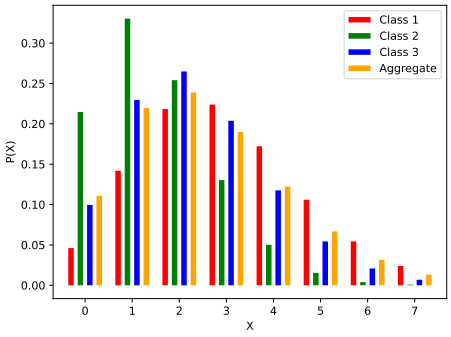
We modeled the re-buy process as the orders the customer makes using a dash button, which comes for free together with the first bought item. We assume that the price of the additional purchases is the same of the first one. The number of purchases (following the first one) that a customer of Class i makes in one month is modeled as a random variable with a Poisson probability distribution:

$$X_i \sim \mathcal{P}oisson(\lambda_i)$$

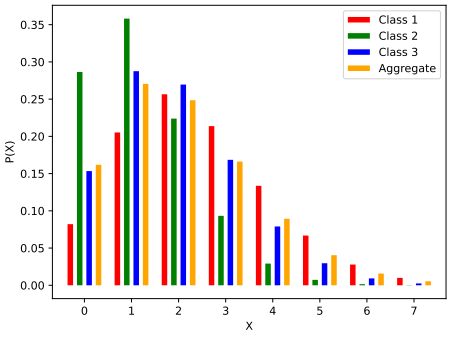
where the parameter λ_i is function of the price: $\lambda_i(p) = \frac{\alpha_i}{\beta_i(p+k_i)}$, with α_i, β_i and k_i normalizing constants. For the aggregate model:

$$\lambda_{agg}(p,b) = \frac{n_1(b)\lambda_1(p) + n_2(b)\lambda_2(p) + n_3(b)\lambda_3(p)}{n_1(b) + n_2(b) + n_3(b)}$$

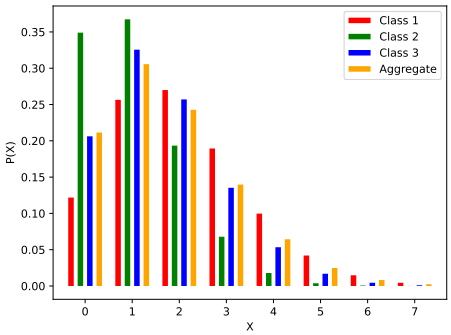
The following diagram shows the behaviour of X_i as the price increases:



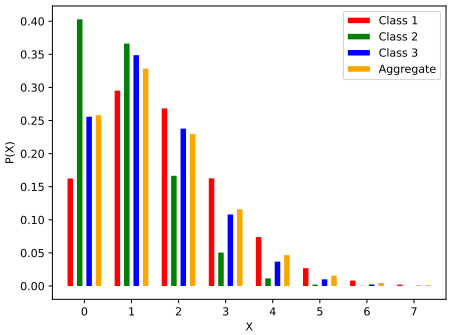
(i) p = 4€



(ii) p = 5.5€



(iii) p = 7€



(iv) p = 8.5€

Figure 2.5: Poisson distributions for the three Classes and the aggregate Class, obtained for different prices and a fixed bid $b = 2,50$€.

Chapter 3

Assignments

3.1 Step 1

3.1.1 Task

Formulate the objective function when assuming that, once a user makes a purchase with a price p, then the ecommerce will propose the same price p to future visits of the same user and this user will surely buy the item. The revenue function must take into account the cost per click, while there is no budget constraint. Provide an algorithm to find the best joint bidding/pricing strategy and describe its complexity in the number of values of the bids and prices available (assume here that the values of the parameters are known). In the following steps, assume that the number of bid values are 10 as well as the number of price values.

3.1.2 Solution

Given a set of bids B and a set of prices P, the objective function is:

$$maximize \;\; \mathcal{R}(p,b) = \sum_{i=1}^{q(b)} [(p - c_{prod})(1 + n_i(p)) - c_i(b)]$$

so we are interested in finding the best pair of price and bid:

$$(p_{opt}, b_{opt}) = \underset{p \in P,\, b \in B}{\operatorname{argmax}} \{\mathcal{R}(p,b)\} = \underset{p \in P,\, b \in B}{\operatorname{argmax}} \left\{ \sum_{i=1}^{q(b)} [(p - c_{prod})(1 + n_i(p)) - c_i(b)] \right\}$$

where:

- $q(b)$ is the number of "first" purchases;
- c_{prod} is the production cost per item;
- $n_i(p)$ is the number of future visits of user i;
- $c_i(b)$ is the cost per click for the single user i;

The algorithm is the iteration over all the values of p in P and b in B to find the maximum return.

$\mathcal{A}$:
```
max ← −∞
for p in P
    for b in B
        if R(p, b) > max
        then max ← R(p, b)
return max
```

and thus its complexity is: $\mathfrak{C}(\mathcal{A}) = \mathcal{O}(|P| \cdot |B|)$

3.2 Step 2

3.2.1 Task

Consider the online learning version of the above optimization problem when the parameters are not known. Identify the random variables, potential delays in the feedback, and choose a model for each of them when a round corresponds to a single day. Consider a time horizon of one year.

3.2.2 Solution

As explained in chapter 2, the random variables are the following (we now consider the aggregate model):

- the daily clicks of new users: $d_t(b_t) \sim \mathcal{N}(\mu_{0,i} + \mu_{bid,i}(b_t),\, \sigma^2)$
- the number of "first" purchases at day t: $q_t(b_t, p_t) = \sum_{i=1}^{d_t(b_t)} X(p_t)$, where $X(p_t) \sim \mathcal{B}e(r(p_t), 1 - r(p_t))$ and $r(p_t)$ is the conversion rate function evaluated at price p_t.
- the cost per click: $c_i(b_t) = b_t - |X(b_t)|$, $X \sim \mathcal{N}(\frac{b_t}{\alpha}, 0.1)$, where α is a normalized constant found by averaging the values for each class.
- the number of future purchases of each user: $n_i(p) \sim \mathcal{P}oisson(\lambda)$

the feedback delay is $T = 30d$ and the objective function is:

$$maximize \quad \sum_{t=1}^{365} \mathcal{R}_t(p,b) = \sum_{t=1}^{365} \sum_{i=1}^{q_t(b,p)} [(p_t - c_{prod})(1 + n_i(p_t)) - c_i(b_t)]$$

where:

- p_t is the price chosen for day t
- b_t is the bid chosen for day t
- $q_t(b,p)$ is the number of "first" purchases at day t;
- c_{prod} is the production cost per item;
- $n_i(p)$ is the number of future visits of user i;
- $c_i(b)$ is the cost per click for the single user i;

3.3 Step 3

3.3.1 Task

Consider the case in which the bid is fixed and learn in online fashion the best pricing strategy when the algorithm does not discriminate among the customers' classes (and therefore the algorithm works with aggregate data). Assume that the number of daily clicks and the daily cost per click are known. Adopt both an upper-confidence bound approach and a Thompson-sampling approach and compare their performance.

3.3.2 Results

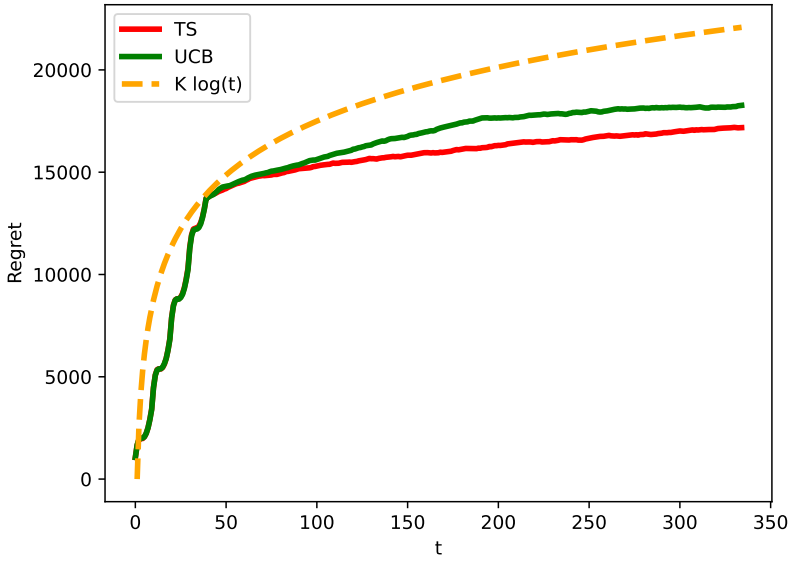


Figure 3.1: Regret plots of both TS and UCB algorithms.

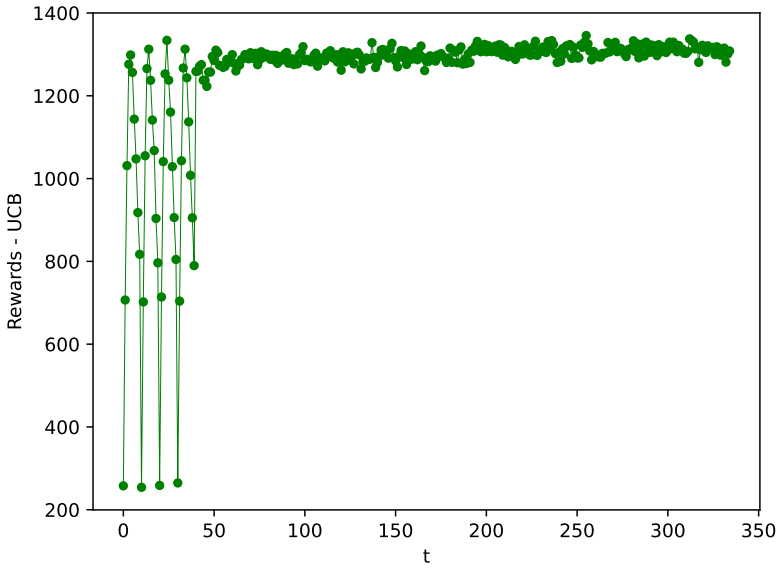


Figure 3.2: Rewards $\mathcal{R}_t$ at each timestep of the UCB algorithm.

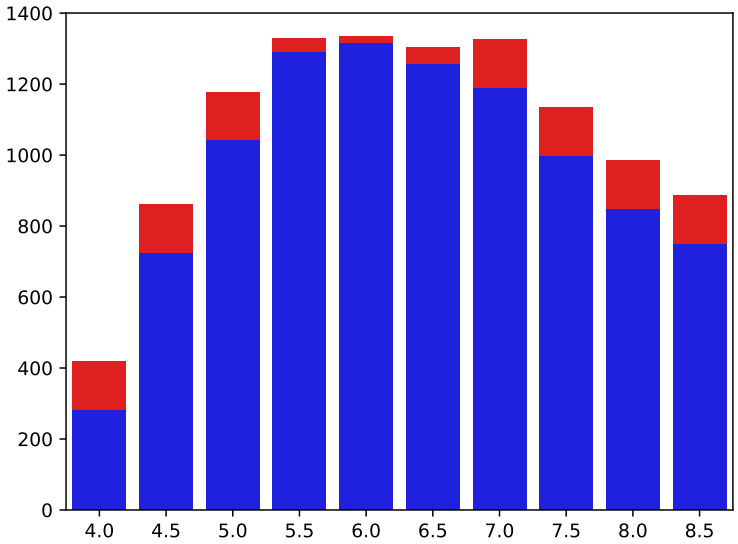


Figure 3.3: Estimated rewards for each arm and relative upper bounds.

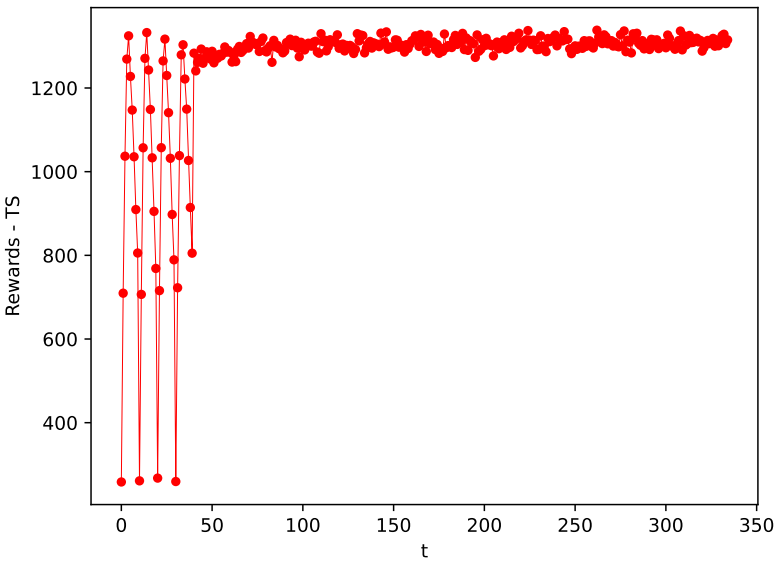


Figure 3.4: Rewards $\mathcal{R}_t$ at each timestep of the TS algorithm.

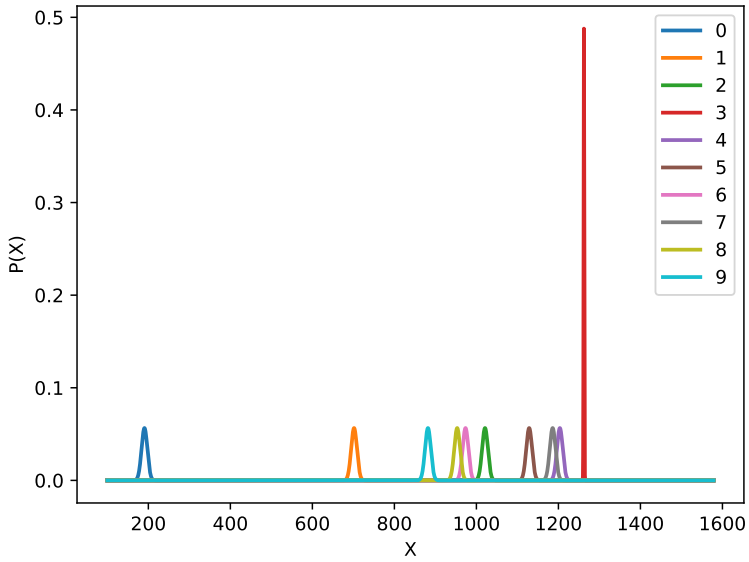


Figure 3.5: Posterior distribution over the arms reward obtained by TS.

3.4 Step 4

3.4.1 Task

Do the same as step 3 when instead a context-generation approach is adopted to identify the classes of customers and adopt a potentially different pricing strategy per class. In doing that, evaluate the performance of the pricing strategies in the different classes only at the optimal solution (e.g., if prices that are not optimal for two customers' classes provide different performance, you do not split the contexts). Let us remark that no discrimination of the customers' classes is performed at the advertising level. From this step on, choose one approach between the upper-confidence bound one and the Thompson-sampling one.

3.4.2 Results

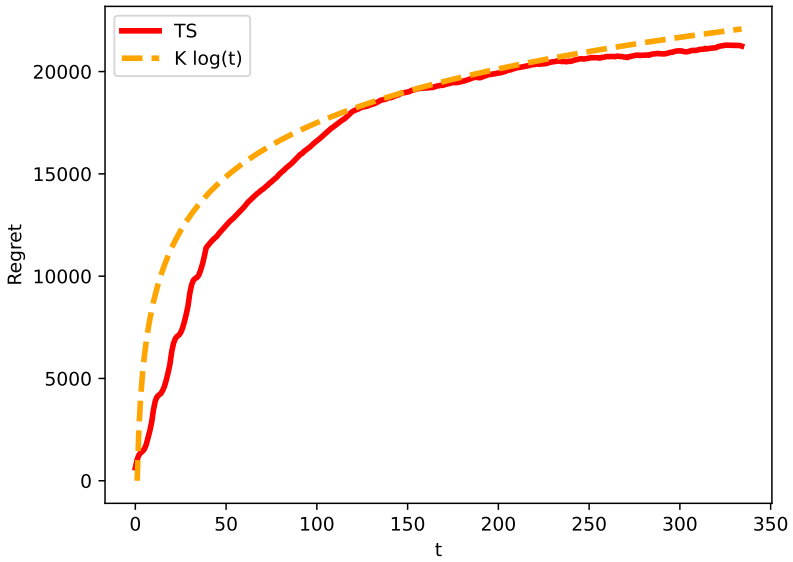


Figure 3.6: Regret plots of the TS algorithm with context generation.

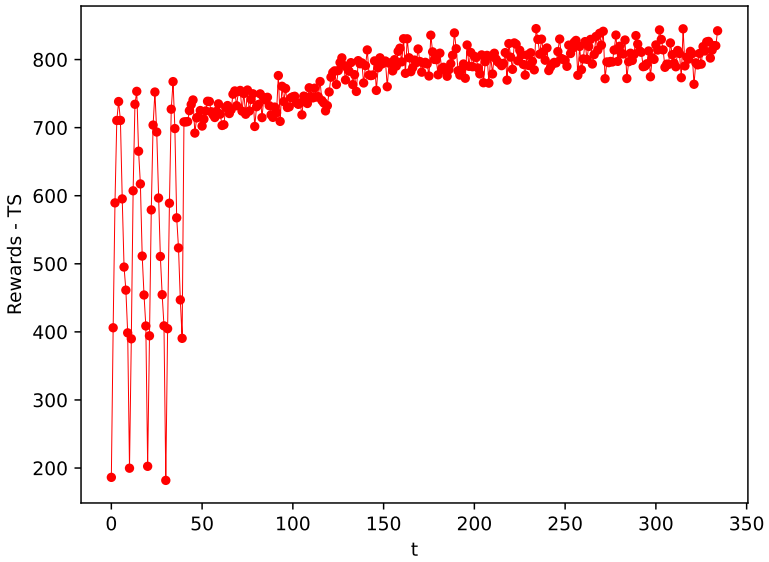


Figure 3.7: Rewards $\mathcal{R}_t$ at each timestep of the TS algorithm with context generation.

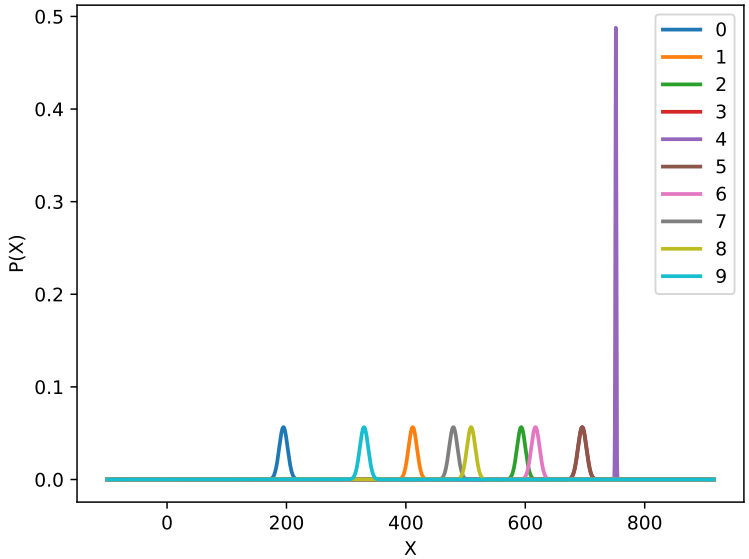
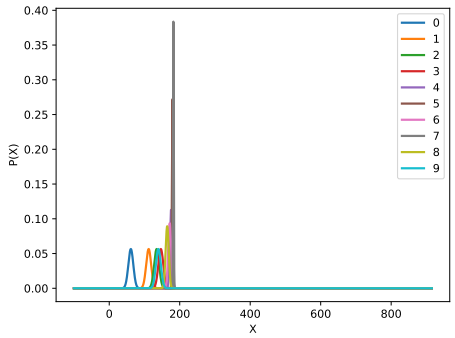
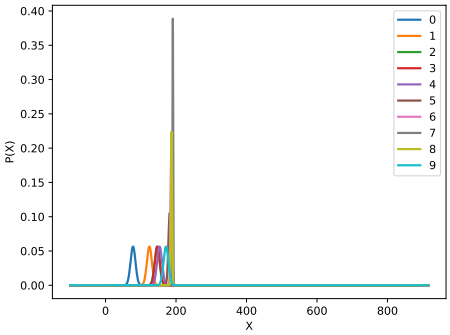


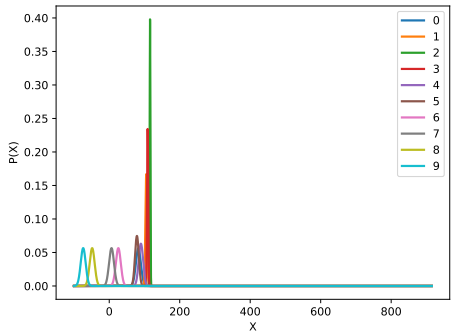
Figure 3.8: Posterior distribution of the rewards of the aggregate model.



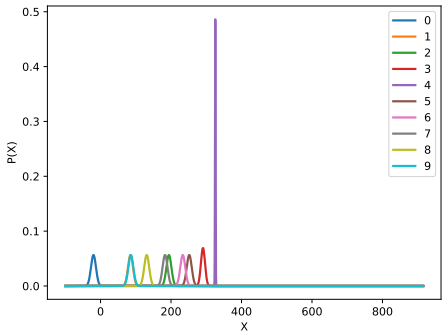
(i) Subclass 00



(ii) Subclass 01



(iii) Subclass 10



(iv) Subclass 11

Figure 3.9: Posterior distribution of the rewards of the single subclasses.

3.5 Step 5

3.5.1 Task

Consider the case in which the prices are fixed and learn in online fashion the best bidding strategy when the algorithm does not discriminate among the customers' classes. Assume that the conversion probability is known. However, we need to guarantee some form of safety to avoid the play of arms that provide a negative revenue with a given probability. This can be done by estimating the probability distribution over the revenue for every arm and making an arm eligible only when the probability to have a negative revenue is not larger than a threshold (e.g., 20%). Apply this safety constraint after 10 days to avoid that the feasible set of arms is empty, while in the first 10 days choose the arm to pull with uniform probability. Do not discriminate over the customers' classes.

3.5.2 Results

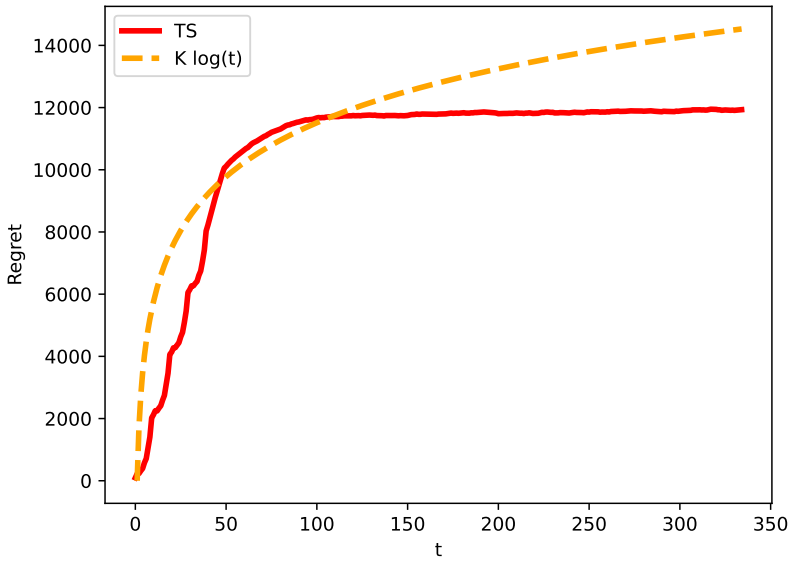


Figure 3.10: Regret plots of the Thompson Sampling algorithm.

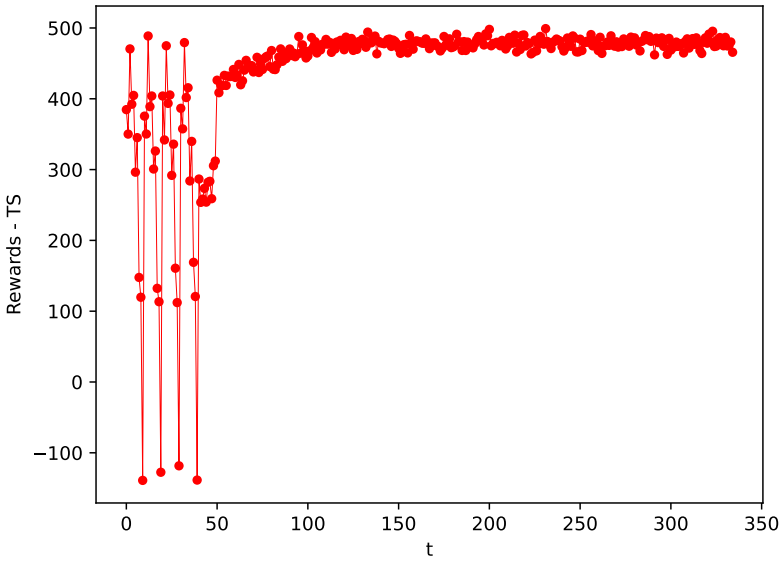


Figure 3.11: Rewards $\mathcal{R}_t$ at each timestep of the TS algorithm with context generation.

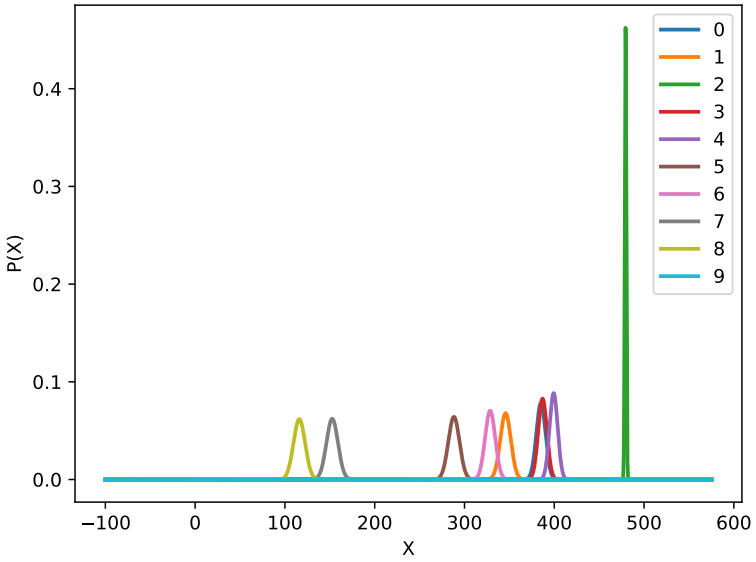


Figure 3.12: Posterior distribution over the arms rewards obtained by TS.

3.6 Step 6

3.6.1 Task

Consider the general case in which one needs to learn the joint pricing and bidding strategy under the safety constraint introduced in step 5. Do not discriminate over the customers' classes both for advertising and pricing.

3.6.2 Results

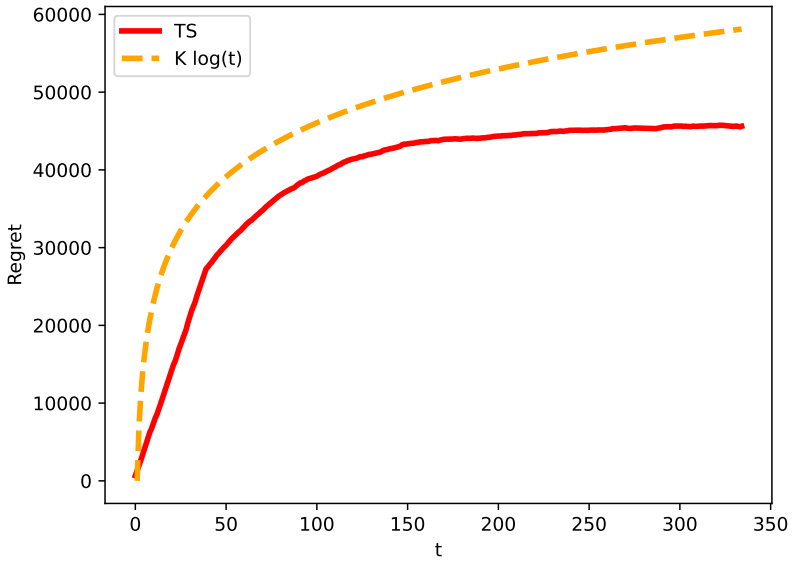


Figure 3.13: Regret plot of the GPTS algorithm.

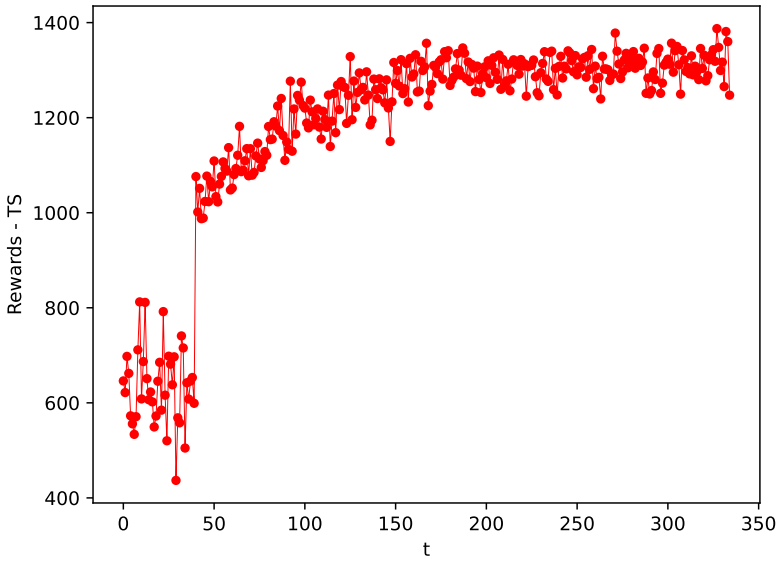


Figure 3.14: Rewards $\mathcal{R}_t$ at each timestep of the GPTS algorithm.

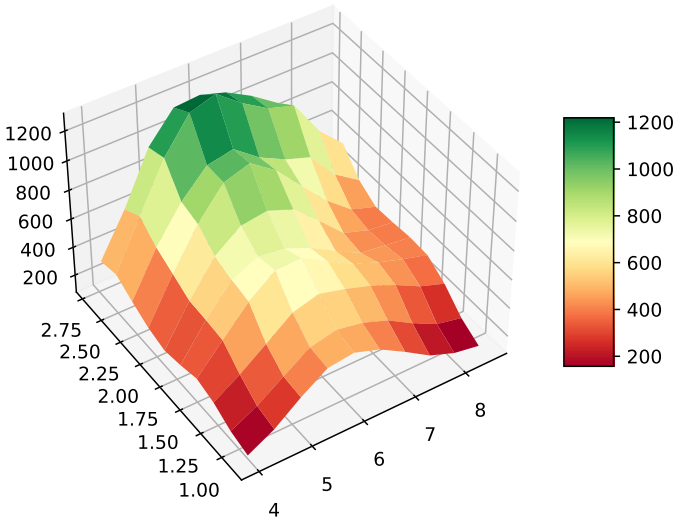


Figure 3.15: Posterior distribution over the arms rewards obtained by GPTS.

3.7 Step 7

3.7.1 Task

Do the same as step 6 when instead discriminating over the customers' classes for pricing. In doing that, adopt the context structure already discovered in step 4.

3.7.2 Results

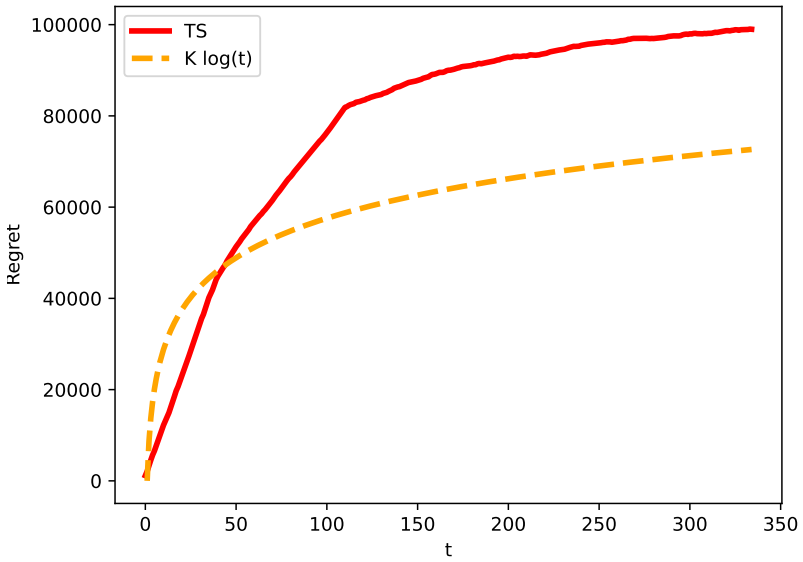


Figure 3.16: Regret plots of the TS algorithm with context generation.

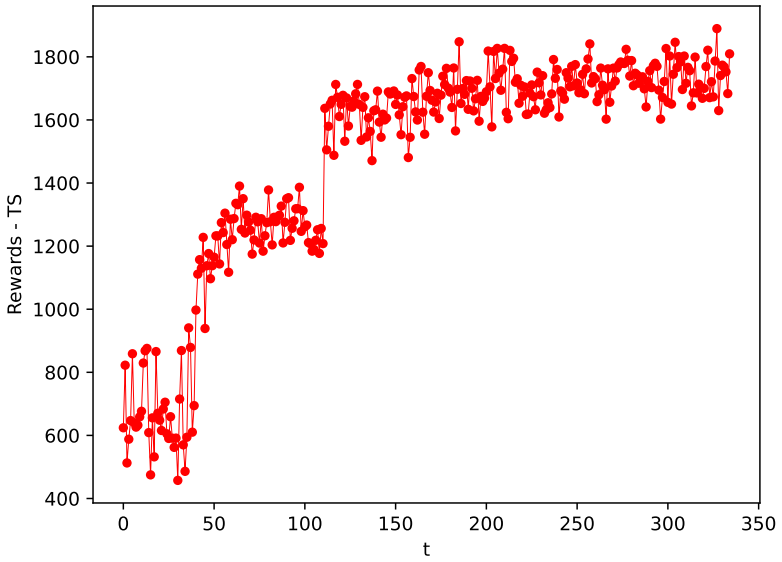


Figure 3.17: Rewards $\mathcal{R}_t$ at each timestep of the TS algorithm with context generation.

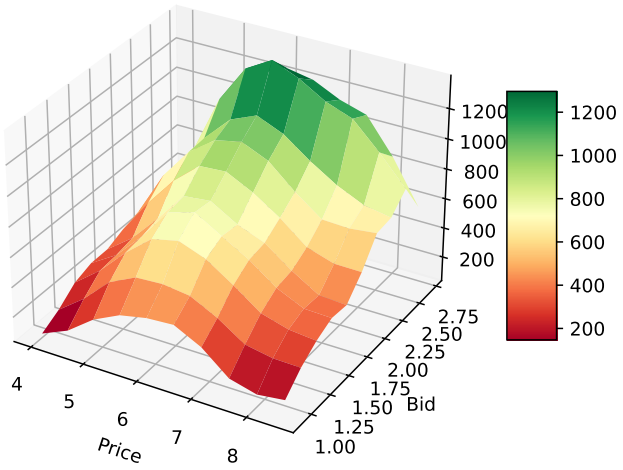
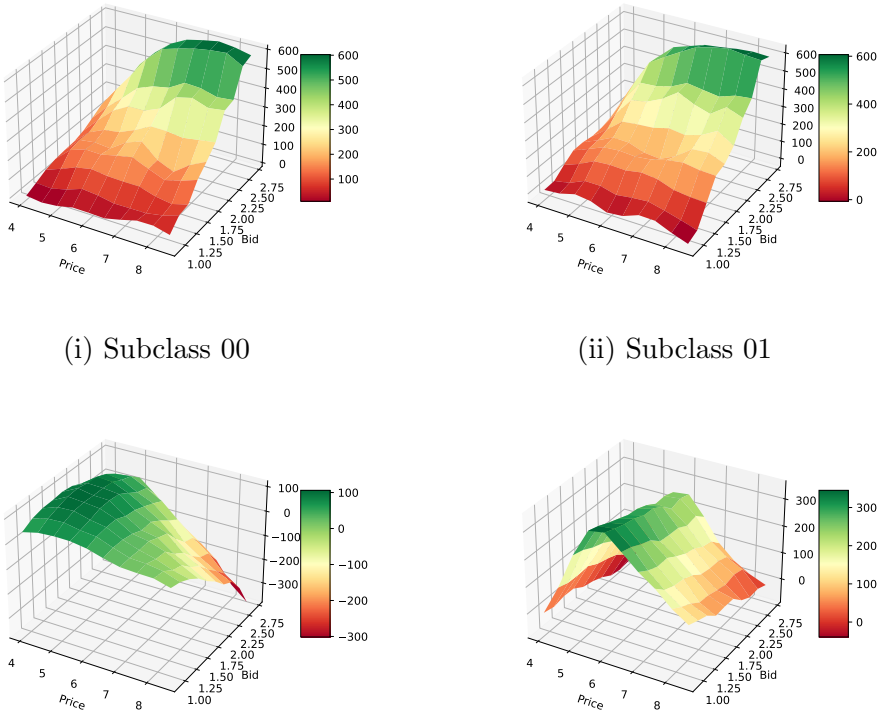


Figure 3.18: Posterior distribution of the rewards of the aggregate model.



(i) Subclass 00

(ii) Subclass 01

(iii) Subclass 10

(iv) Subclass 11

Figure 3.19: Posterior distribution of the rewards of the single subclasses.

3.8 Design choices

3.8.1 Aggregate class

Tasks which don't require to perform discrimination use an aggregate model. The aggregate model is obtained by a weighted average of all the features and probability distributions of the three classes, instead of using a "dummy" fourth class. This is to simulate a realistic situation in which time and computational resources are not employed to perform discrimination and to obtain a measure of the improvement that context generation can bring.

3.8.2 Future purchases

The Poisson distribution has been chosen to model the future visits of the users due to the fact that it is defined over positive integers, it is simple (just

one parameter λ) and suitable to the values that we needed to simulate a realistic process.

3.8.3 Context generation

It can be observed that the context generation algorithm does not provide the optimal splitting at every experiment. This is due to the complexity of the problem and the high variances in the rewards produced by the environment, if compared to the closeness of the arms' means. Here we decided to keep real values, instead of simplifying the model to ensure an optimal splitting for every experiment. With a larger time horizon, we would have a sufficient number of samples to always split correctly.

Chapter 4

Algorithms and other details

4.1 UCB

Step 3 requires to adopt an upper-confidence bound approach. The algorithm we use is a slightly modified version of the UCB1 algorithm working on $|A| = 10$ arms:

i) play the arm $a = t \ \% \ |A|$ for the first $|A| + 30$ timesteps;

ii) at timestep t play arm a_t such that:

$$a_t \leftarrow \underset{a \in A}{\operatorname{argmax}} \left\{ \bar{x} + 80 \sqrt{\frac{2 \log(t)}{n_a(t-1)}} \right\}$$

where $n_a(t-1)$ is the number of times arm a has been pulled before timestep t. The factor 80 is a normalizing constant, since the original expression of the UCB1 algorithm bound comes from the Hoeffding bound, which is made to estimate the expected value of a random variable with support in [0,1]. Since the rewards have much larger values, we need to adjust the respective upper bound on their expected value.

The regret of the UCB algorithm is:

4.2 Thompson Sampling

For steps 3, 4 and 5 we follow the Thompson Sampling approach. Our implementation of the Thompson Sampling algorithm takes into account the Gaussian nature of the reward per each arm, since the number of new customers is a Gaussian random variable, given the bid and the price. For this

reason, the prior distribution is still Gaussian. Except for the prior updates, the procedure is similar to the one used for Bernoulli variables:

i) model the prior distributions:

$$\mathbb{P}(\mu_a = \theta_a) = \mathcal{N}(\theta_a, \sigma_a^2)$$

ii) initialize the posterior distributions:

$$\mathbb{P}(X|\mu_{0,a}, \tau_{0,a}) = \mathcal{N}(X|\mu_{0,a}, 1/\tau_{0,a})$$

iii) at every time t, for every arm a:

$$\tilde{\theta}_a \leftarrow Sample\left(\mathbb{P}(\mu_a = \theta_a)\right)$$

iv) at every time t play arm a_t such that:

$$a_t \leftarrow \underset{a \in A}{\operatorname{argmax}}\{\tilde{\theta}_a\}$$

v) update the Gaussian distribution of arm a_t as:

$$\tau_{0,a} \leftarrow \tau_{0,a} + n_a(t-1)\tau_a$$

$$\mu_{0,a} \leftarrow \frac{\tau_{0,a}\mu_{0,a} + \tau_a \sum_{i=1}^{n} x_i}{\tau_{0,a} + n_a(t)\tau_a}$$

for more details, see chapter 5.

where:

- $\tau_a = 1/\sigma_a^2$ is the precision of the prior distribution;
- $n_a(t)$ is the number of times the arm has been pulled until time t;
- x_i is the reward received at each round the arm was pulled;
- $\mu_{0,a}$ is the estimated mean of arm a;
- $\tau_{0,a}$ is the precision of the output model.

4.3 Gaussian Process Thompson Sampling

For Gaussian Process Thompson Sampling, we use the following Gaussian (RBF) kernel:

```
kernel = C(1e5, 'fixed') * RBF([2*0.15, 5*0.15],
        length_scale_bounds='fixed')
```

The kernel's length scale is a vector in the form $k*[0.2, 0.5]$, to compensate for the difference in scale of the input variables (bids are spaced by 0.2, while prices are spaced by 0.5). The resulting kernel makes each sample have symmetric influence on neighboring arms.

4.4 Context Generation algorithm

In order to pull the best arm for each context, we train 9 learners together, based on the features (XX , 0X, 1X, X0, X1, 00, 01, 10 11). Starting from a time instant $\bar{t}$ and once every time interval t^* the algorithm checks if it is worth to split along one feature and the most promising feature is chosen accordingly. The choice is performed by exploiting a lower confidence bound: $\mu_1 + \mu_2 \geq \mu_0$, where μ is the mean of rewards of the best arm minus its lower bound for that context and the lower bound is computed, starting from the variance of the arms' rewards.

Chapter 5

References

5.1 Links

- GitHub repository of the project: `https://github.com/tizianofucci/DIA2021AdvertisingAndPrincing`
- Application of Gaussian Thompson Sampling: `https://towardsdatascience.com/thompson-sampling-fc28817eacb8`