OLA project: Pricing and Advertising

Lorenzo Ferretti	lorenzo2.ferretti@mail.polimi.it	10713719
Nicolò Fontana	nicolo.fontana@mail.polimi.it	10581197
Carlo Sgaravatti	carlo.sgaravatti@mail.polimi.it	10660072
Marco Venere	marco.venere@mail.polimi.it	10865088
Enrico Zardi	enrico.zardi@mail.polimi.it	10659549

Professors:
Castiglioni Matteo,
Gatti Nicola,
Bernasconi de Luca Martino

A.Y. 2022/2023

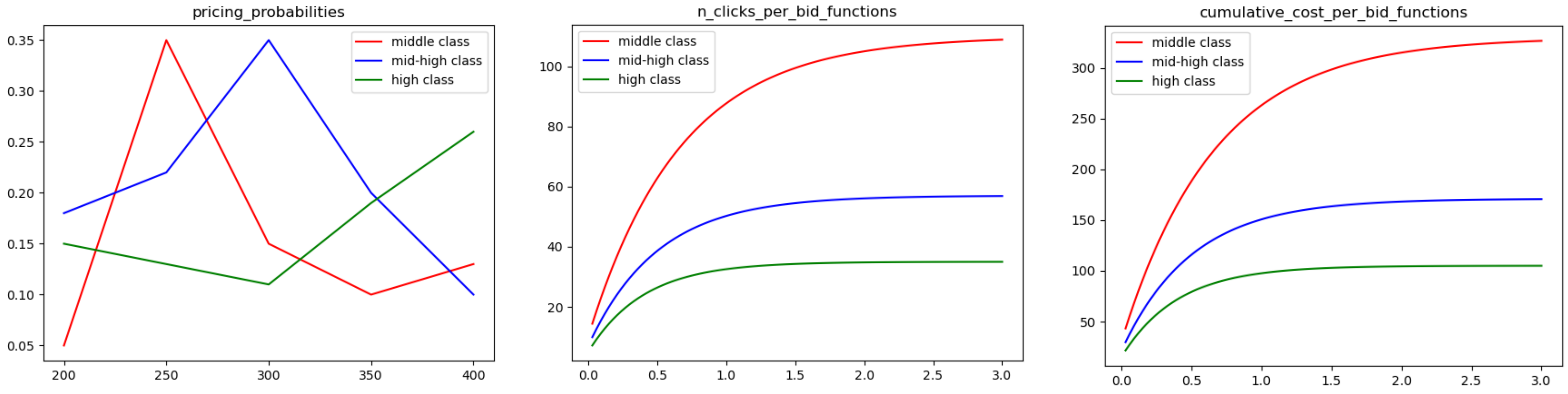
The Problem

- Setting: an **e-commerce** website sells a **product** and can control both the **price** and the **advertising strategy**.
- Users have **two binary features** that can be observed by the advertising platforms: **F1** and **F2**
- Users can belong to **three different classes** according to such features: **C1**, **C2**, and **C3**
- Each class is described by two functions:
  1. the **number of daily clicks** as the bid varies
  2. the **cumulative daily cost of the clicks** as the bid varies
- Each class presents a different **purchase conversion rate**
- The time horizon is **365** rounds long
- The reward is given as $R = CR(p) \cdot n_{clicks}(bid) \cdot (p - c) - c_{cost}(bid)$ where p is the chosen prince, b is the chosen bid, and c the unit cost of the product. Indeed, $p - c$ is the margin.

Step 0 : Motivations and Environment design

- The object sold by the website is a **brand watch**.
- The prices space from 200 € to 400 € with intervals of 50 €.
- Each product has a **unit cost** set to 150$
- The E-commerce can observe two different features
  - F1 : **income level**
  - F2: **lifestyle** (fancy or sober)
- Users in class C1 are characterized by a **medium income level** and they are independent from the second feature.
- Users in class C2 are usually **high-income** and **sober lifestyle**
- Users in class C3 are **high-income** but with **fancy lifestyle**

Step 0 : Environment design – Conversion Rates



Step1 : Learning for pricing - Request

The scenario is the following:

- Consider all the users belonging to class **C1**
- The curves related to the advertising part of the problem are **known**
- The curve related to the pricing problem is **unknown**

The request is to:

- Apply the **UCB1 algorithm** to estimate the conversion rates
- Apply the **TS algorithm** to estimate the conversion rates
- Plot the average value and standard deviation of the **cumulative regret**, **cumulative reward**, **instantaneous regret**, and **instantaneous reward**

Step1 : Learning for pricing - Solution

- The bid b is chosen such that the associated number of clicks and the cumulative cost maximize the reward, given the chosen price, and the margin associated with that price.

- At each round, the two regret minimizers determine the price to choose by maximizing the product between the estimated conversion rate and the margin associated with that price:

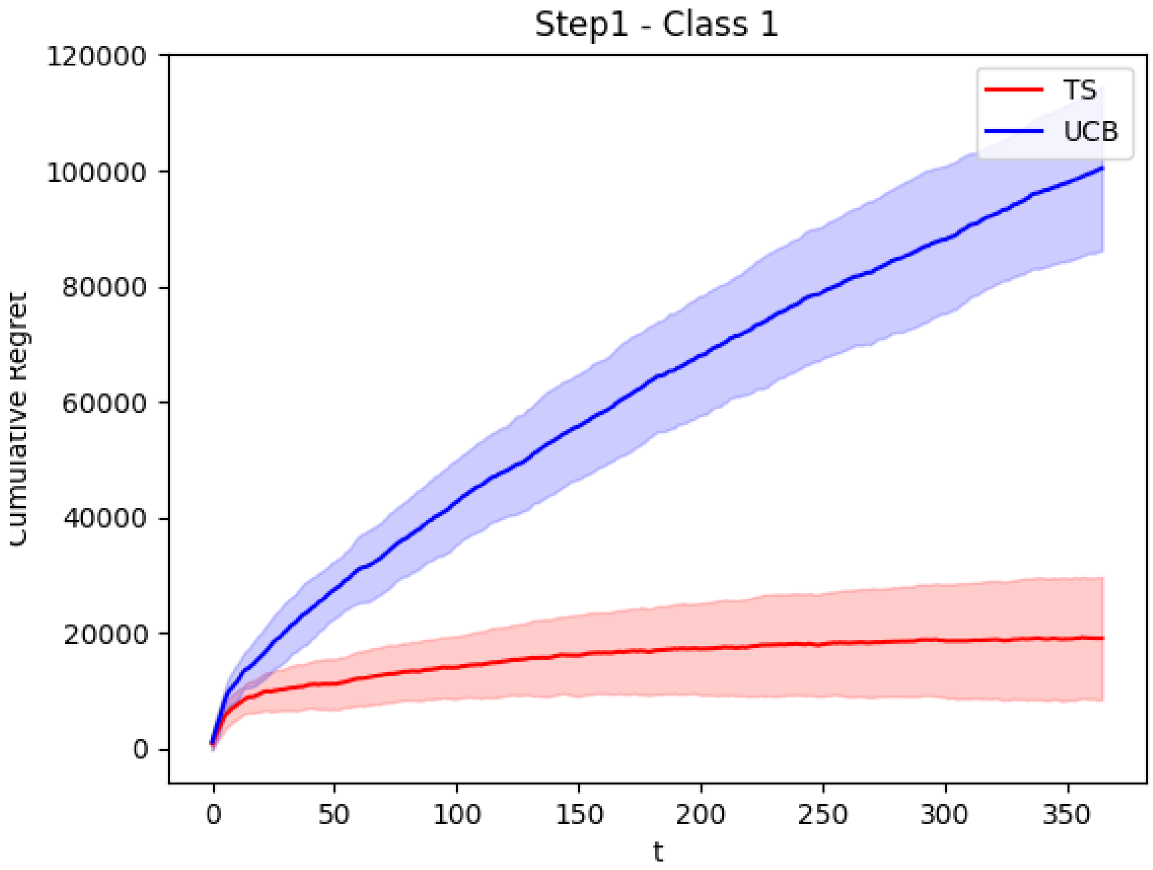
$$p^* \in \arg\max \widetilde{CR}_p \cdot (p - c)$$

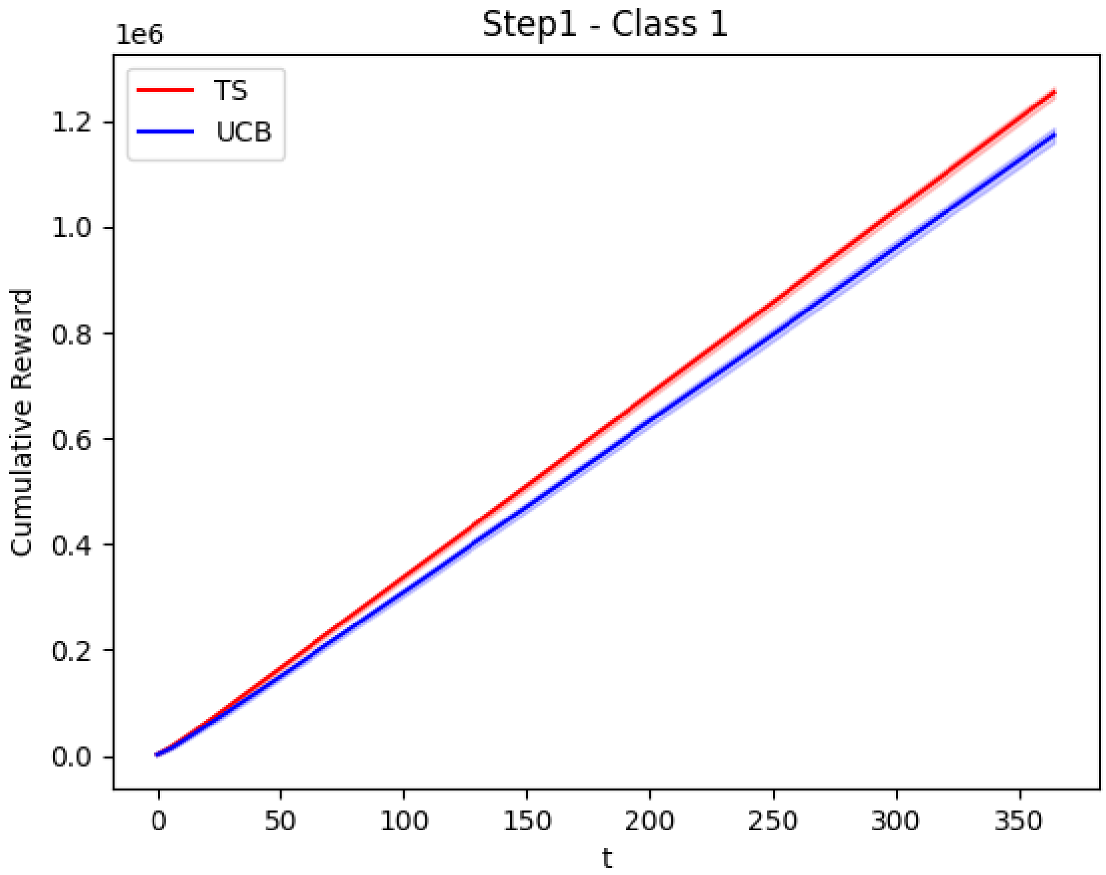
where $\widetilde{CR}_p$ is the estimated conversion rate for the arm associated with price p.

- The update rules for the two algorithms are:
  1. For TS algorithm: $(\alpha_{a_t}, \beta_{a_t}) \leftarrow (\alpha_{a_t}, \beta_{a_t}) + (c_{p,t}, c_{n,t})$
  2. For UCB1 algorithm: $n_{a_t}(t+1) \leftarrow n_{a_t}(t) + c_{p,t} + c_{n,t}$

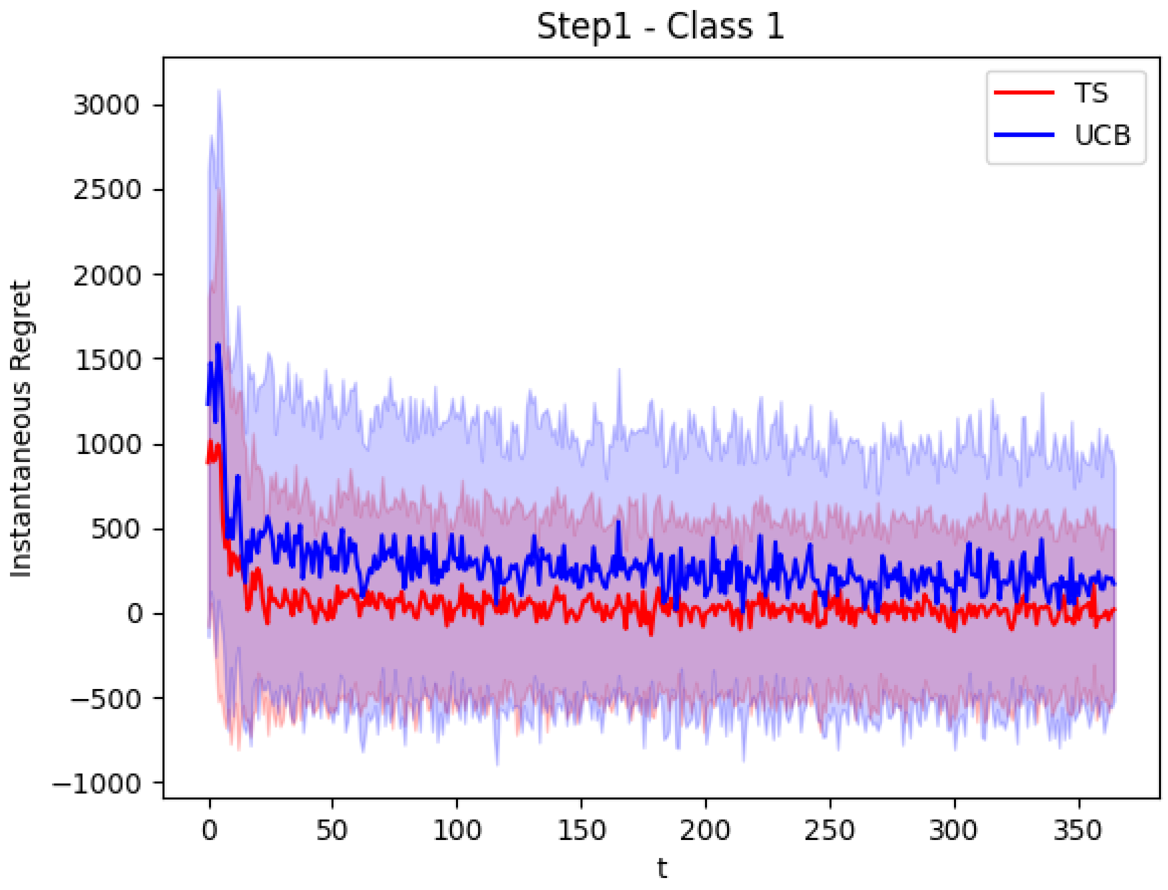
  where $c_{p,t}$ is the number of positive conversions, $c_{n,t}$ is the number of negative conversion, and their sum is the number of clicks of that round

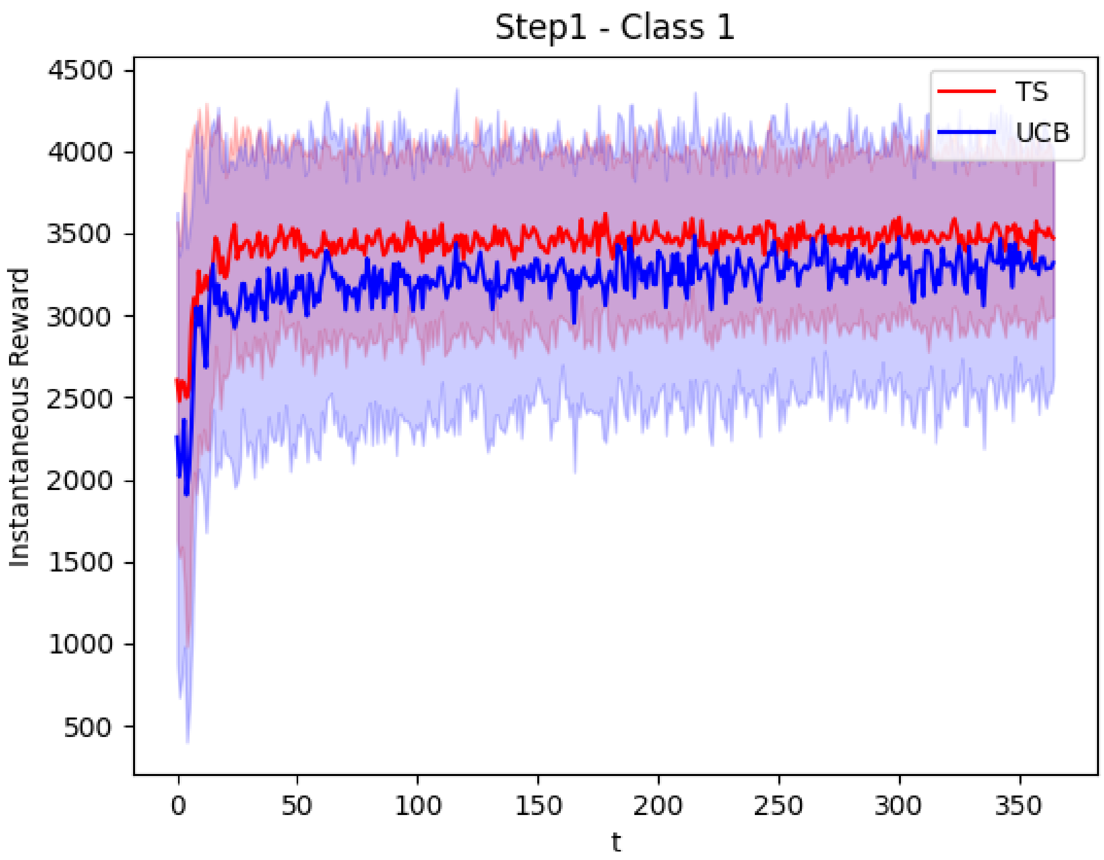
Step1 : Learning for pricing – Results: Cumulative Plots



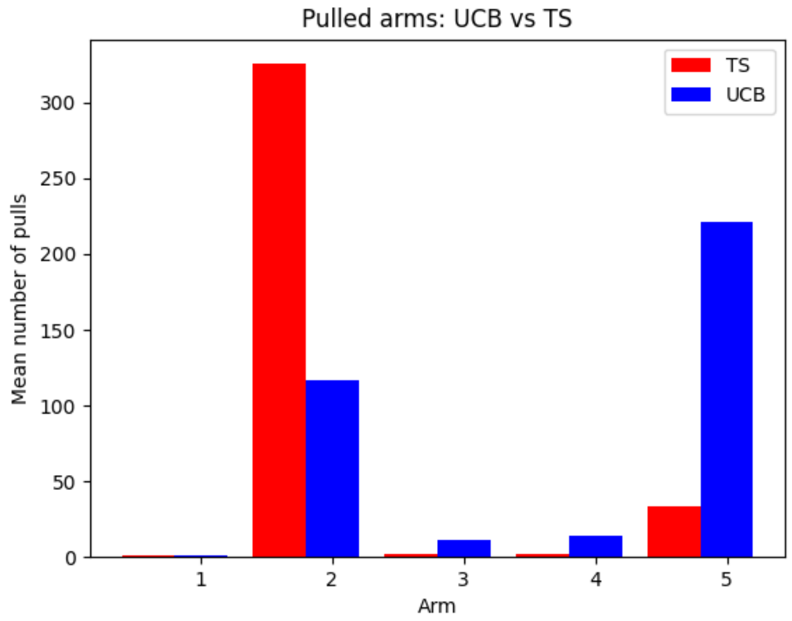


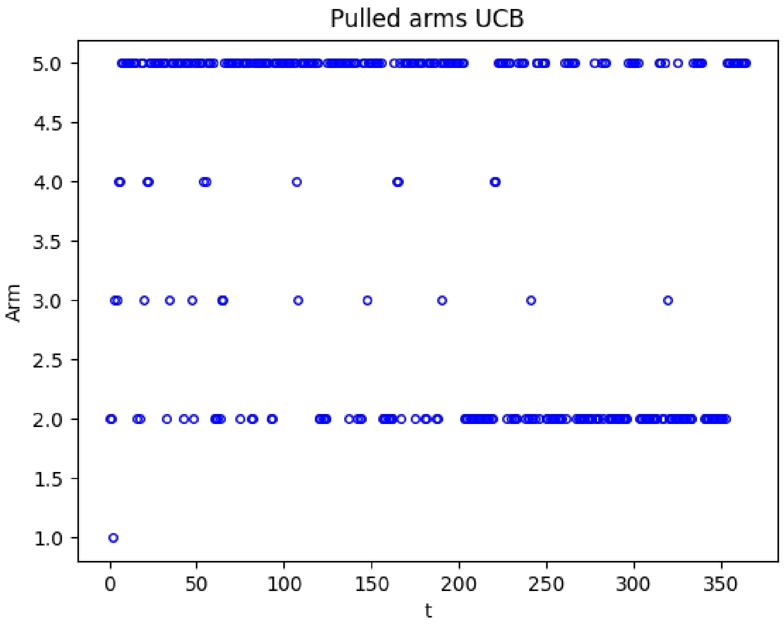
Step1 : Learning for pricing – Results: Instantaneous Plots





Step1 : Learning for pricing - Observations





Step2 : Learning for advertising - Request

The scenario is the following:

- Consider all the users belonging to class **C1**
- The curve related to the pricing problem is **known**
- The curves related to the advertising part of the problem are un**known**

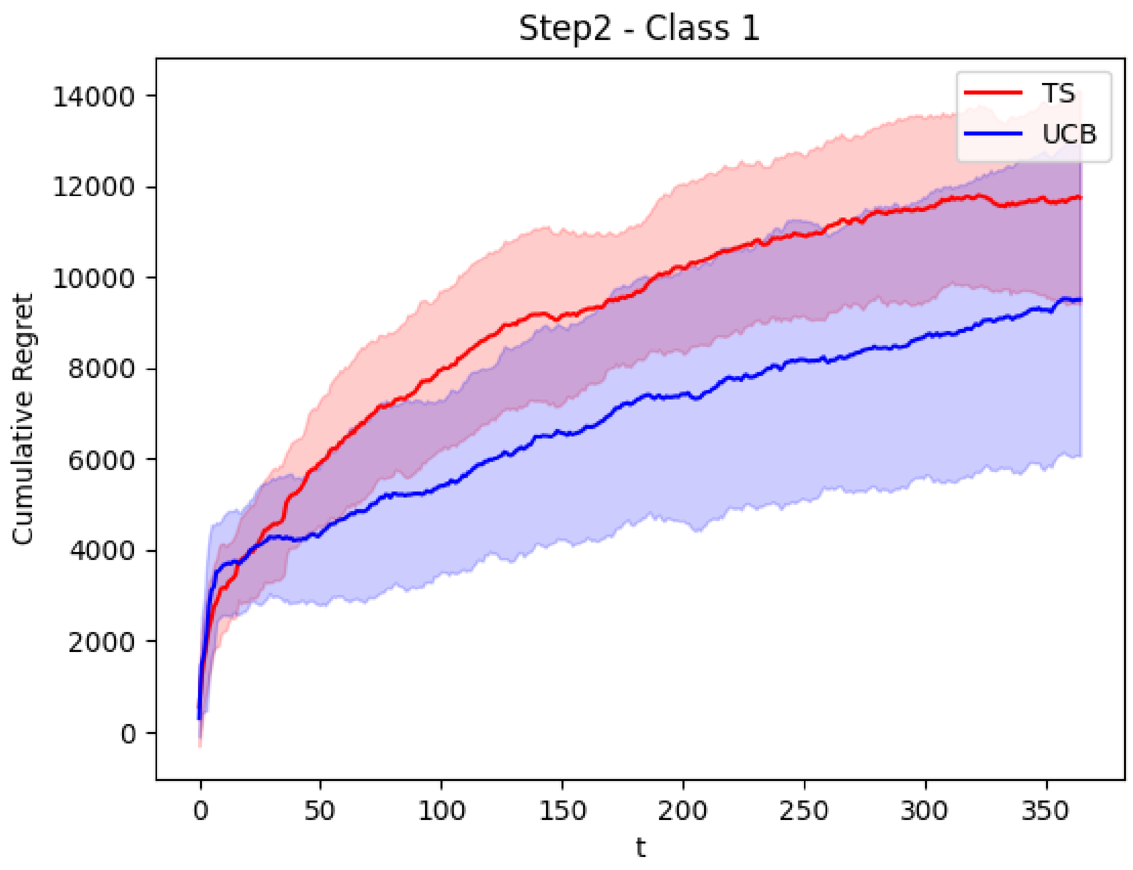
The request is to:

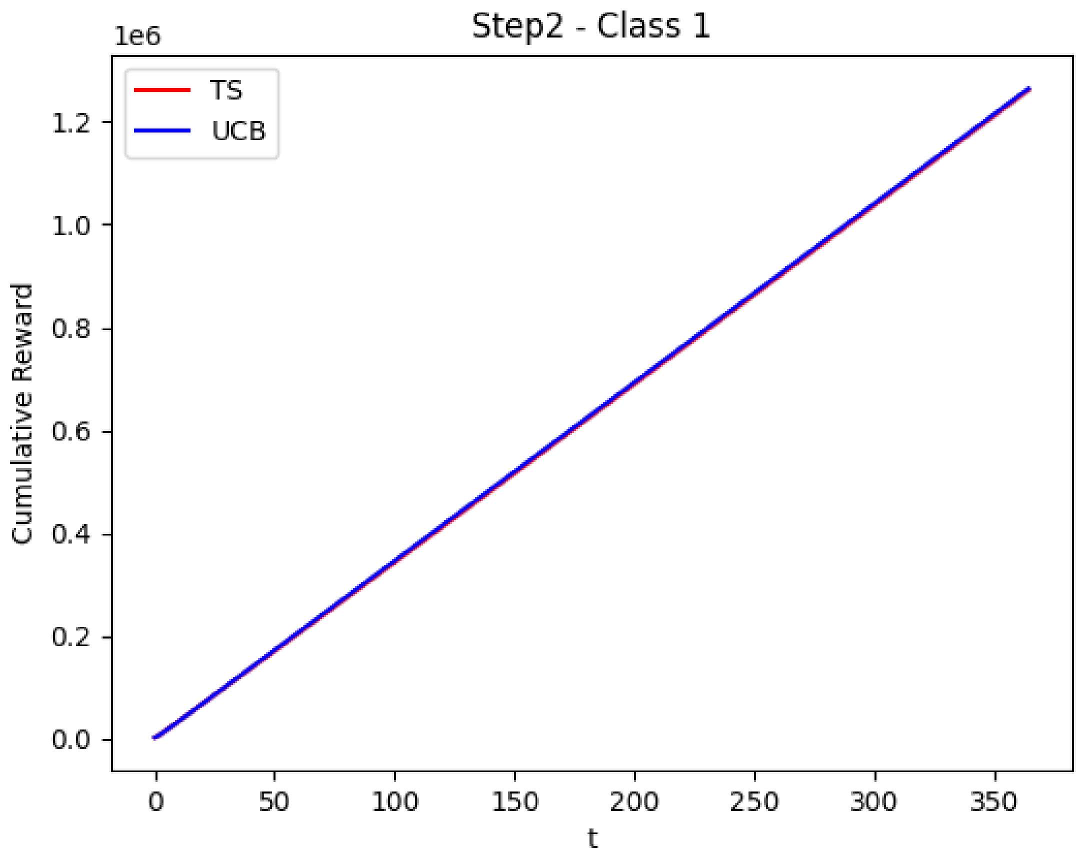
- Apply the **GP-UCB algorithm** to estimate the curves of the advertising problem
- Apply the **GP-TS algorithm** to estimate the curves of the advertising problem
- Plot the average value and standard deviation of the **cumulative regret**, **cumulative reward**, **instantaneous regret**, and **instantaneous reward**.

Step2 : Learning for advertising - Solution

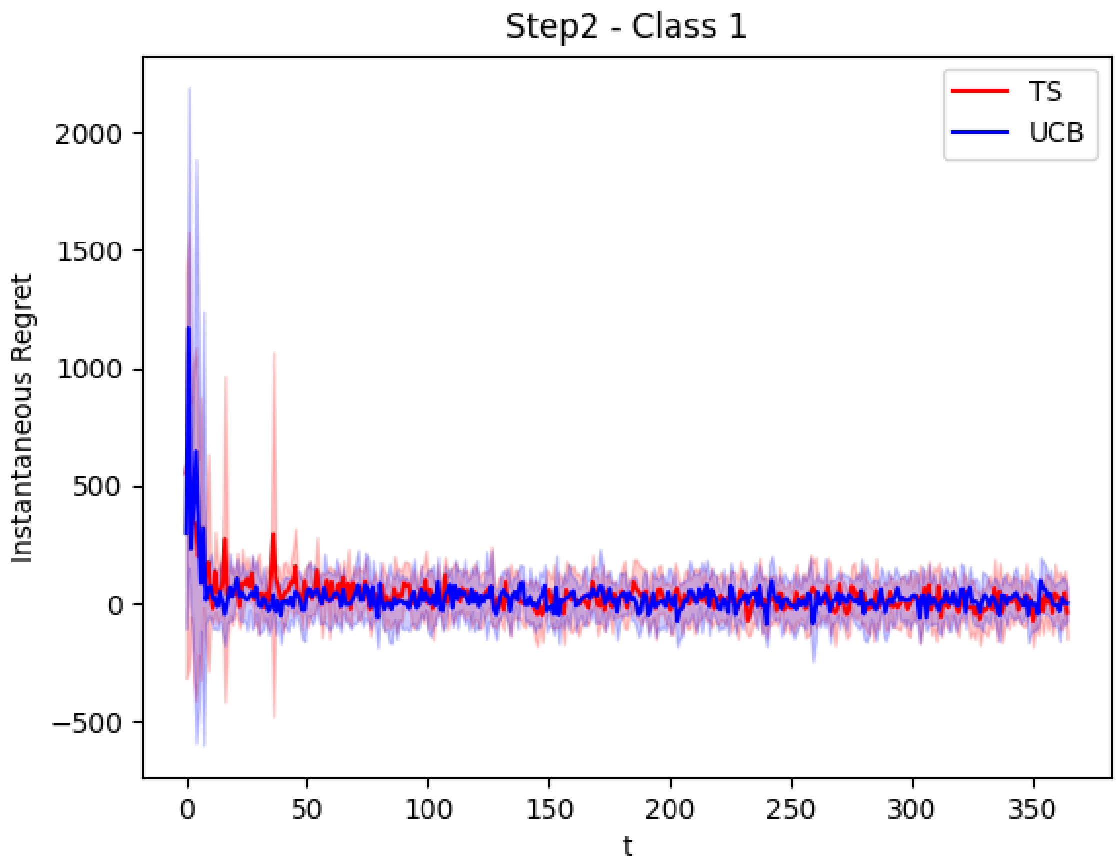
- The **price** is fixed to maximize the **conversion rate multiplied by the margin**
- Both regret minimizers use **Gaussian processes** and store the number of clicks and the cumulative cost w.r.t. the bids.
- We assume a **correlation exists between arms** near each other, thus the two functions must be sufficiently **smooth**.
- Obtain **uncertainty** over the prediction for each regret minimizer.
  - In the UCB algorithm, this measure is used to compute the **confidence bound** around the predicted mean.
  - In the TS algorithm, it is used as the **standard deviation** of the Gaussian distribution with the predicted mean
- Finally, the bid to be chosen maximizes the **reward** with respect to the optimal price (known).

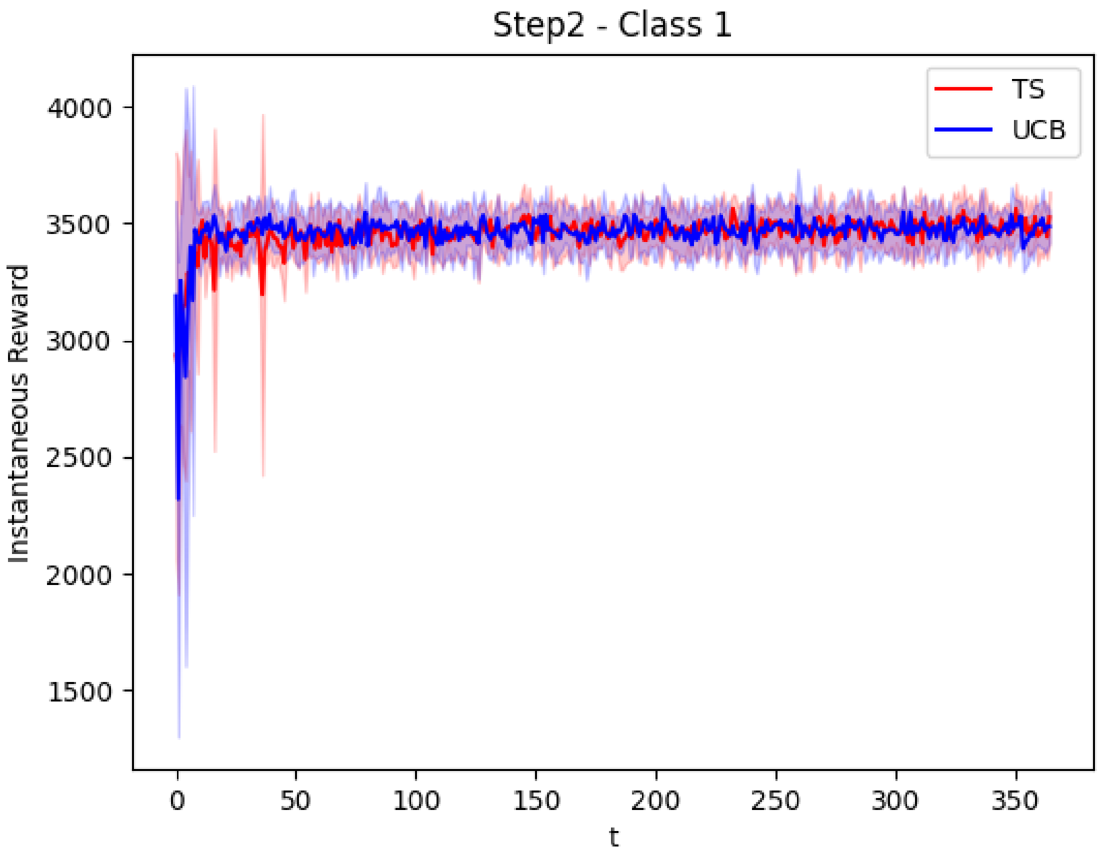
Step2 : Learning for advertising – Results: Cumulative Plots





Step2 : Learning for advertising – Results: Instantaneous Plots





Step3 : Learning for joint pricing and advertising – Request

The scenario is the following:

- Consider all the users belonging to class **C1**
- The curves related to the advertising part of the problem are **unknown**
- The curve related to the pricing problem is **unknown**

The request is to:

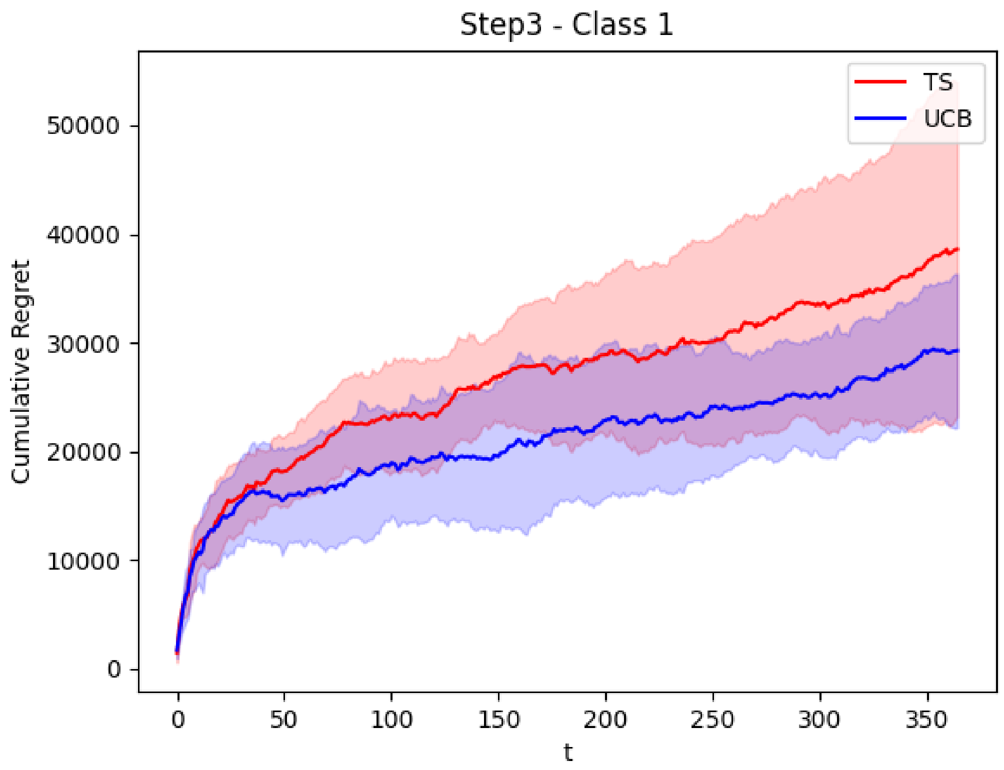
- Apply the **GP-UCB algorithm** to estimate the curves of the advertising problem
- Apply the **GP-TS algorithm** to estimate the curves of the advertising problem
- Plot the average value and standard deviation of the **cumulative regret**, **cumulative reward**, **instantaneous regret**, and **instantaneous reward**

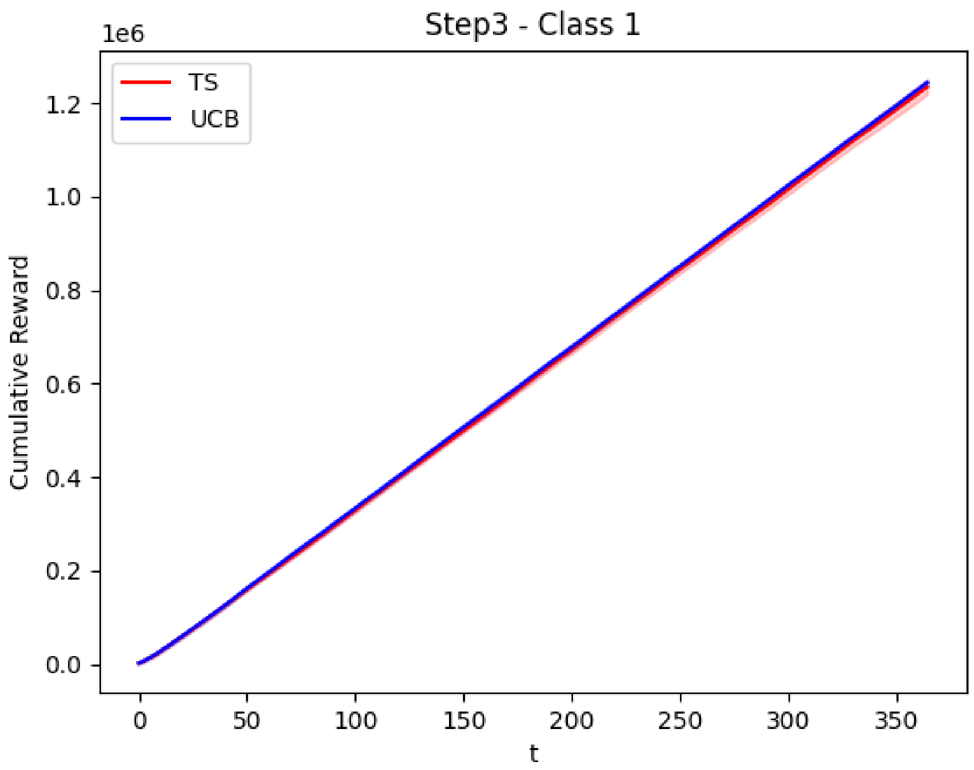
Step3 : Learning for joint pricing and advertising – Solution

- We adopt a **two-level optimization** approach.
- Since no constraint was given on how to solve the **pricing problem** we decided, for simplicity, to adopt the classic **Thompson Sampling algorithm**: the best-performing algorithm in Step1.
- **Reward**: $R = CR(p) \cdot n_{clicks}(bid) \cdot (p - c) - c_{cost}(bid)$
- The optimization process is as follows:
  - At each round we need to **choose the best arm** to pull for the conversion rate and the best bid to pull.
  - The TS learner selects the **highest draw** coming from the Beta distributions associated with each arm, and we get $\widetilde{CR_p}$, the **estimated conversion rate** (the drawn value) and P_d, the **price** associated with that arm.
  - $b^* = \arg\min_b \widetilde{CR_p} \cdot n_{clicks}(b) \cdot (p - c) - c_{cost}(b)$

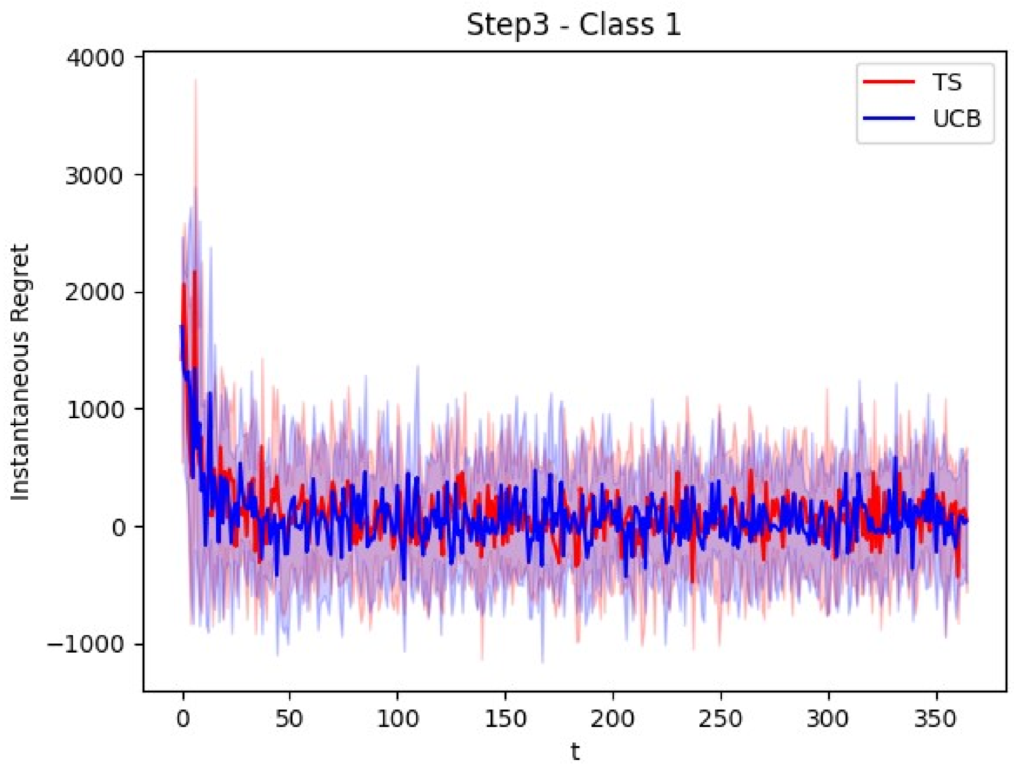
    where $n_{clicks}(b)$ and $c_{cost}(b)$ are estimated by the Gaussian process
  - Then, the selected prices and bids are played and the environment returns the **effective conversion rates**, **number of clicks** and **cumulative cost**, that upgrade a **TS learner** and the **two Gaussian processes**.

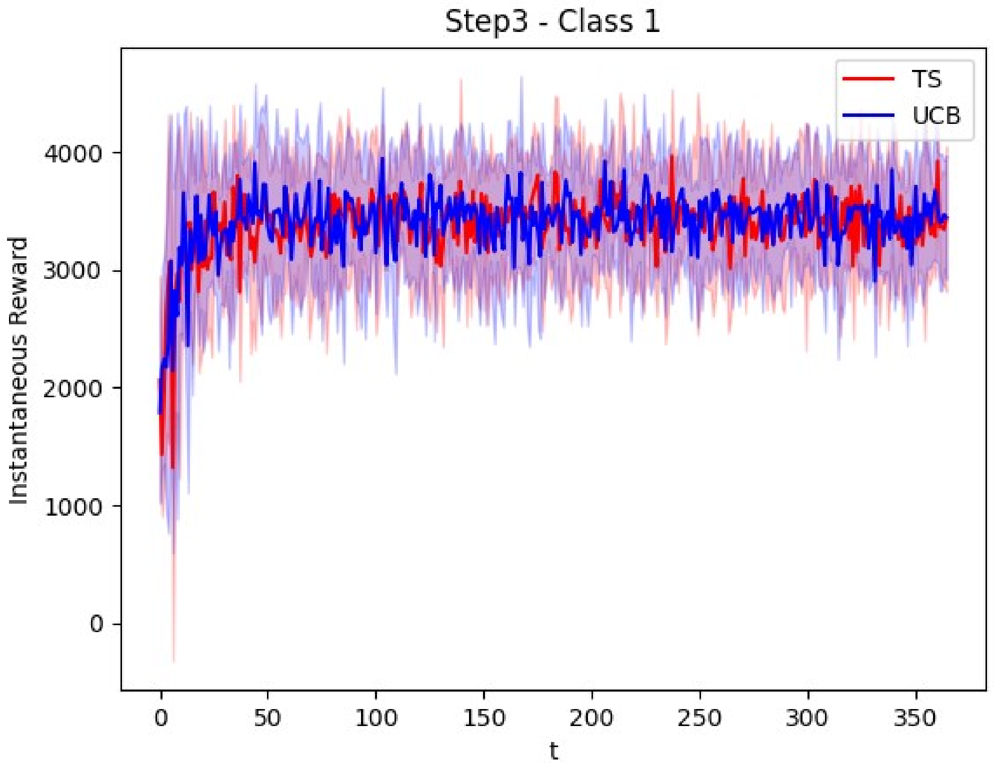
Step3 : Learning for joint pricing and advertising – Results



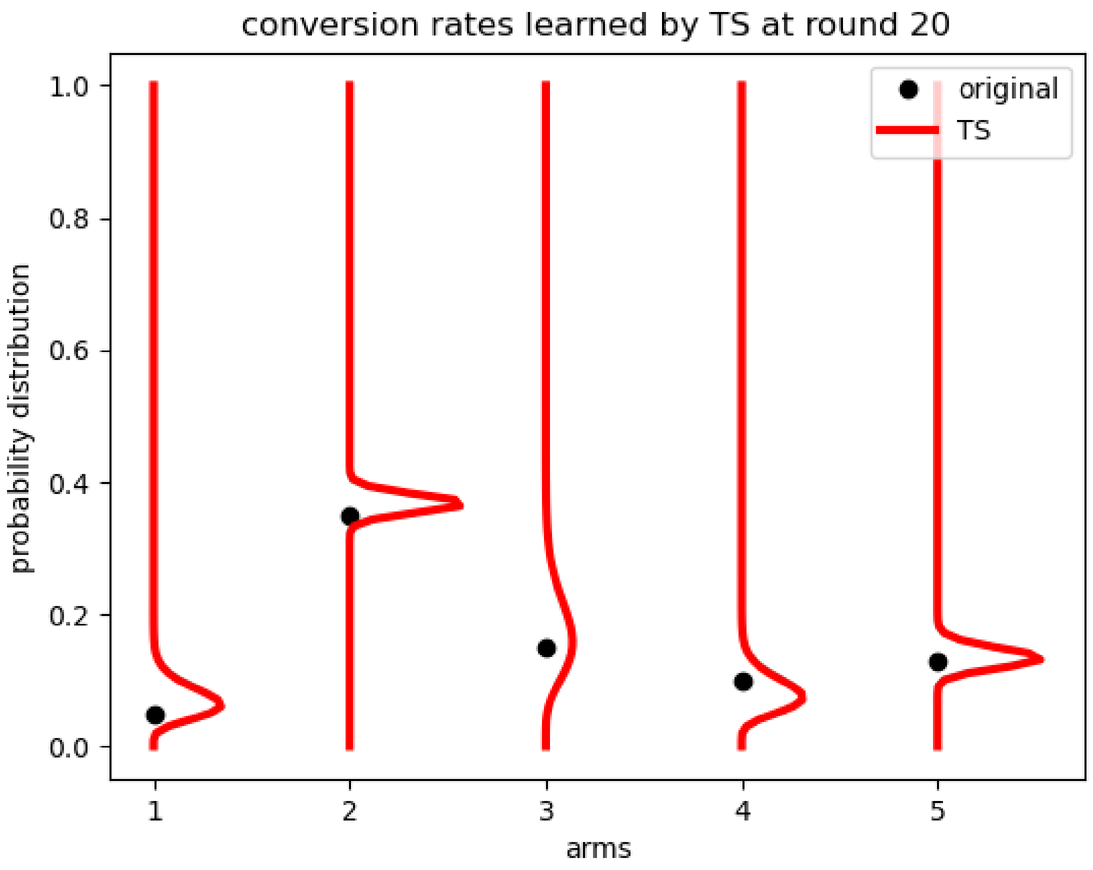


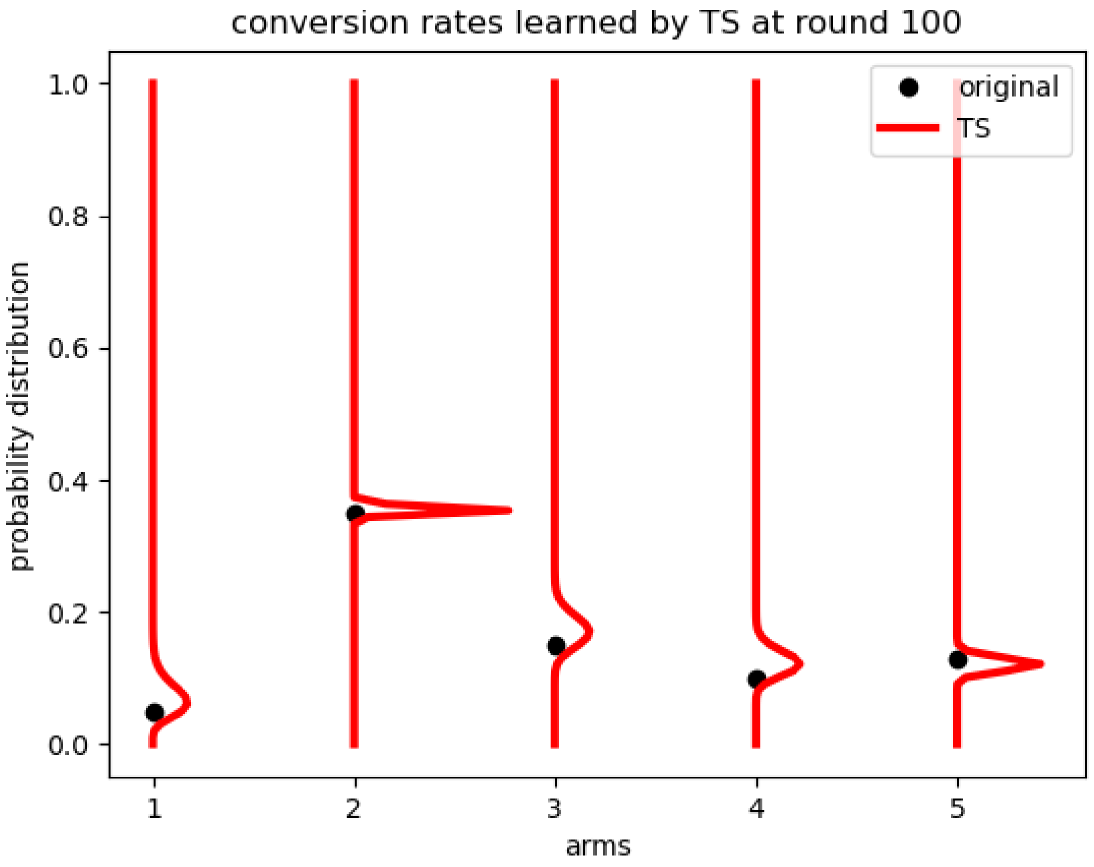
Step3 : Learning for joint pricing and advertising – Results



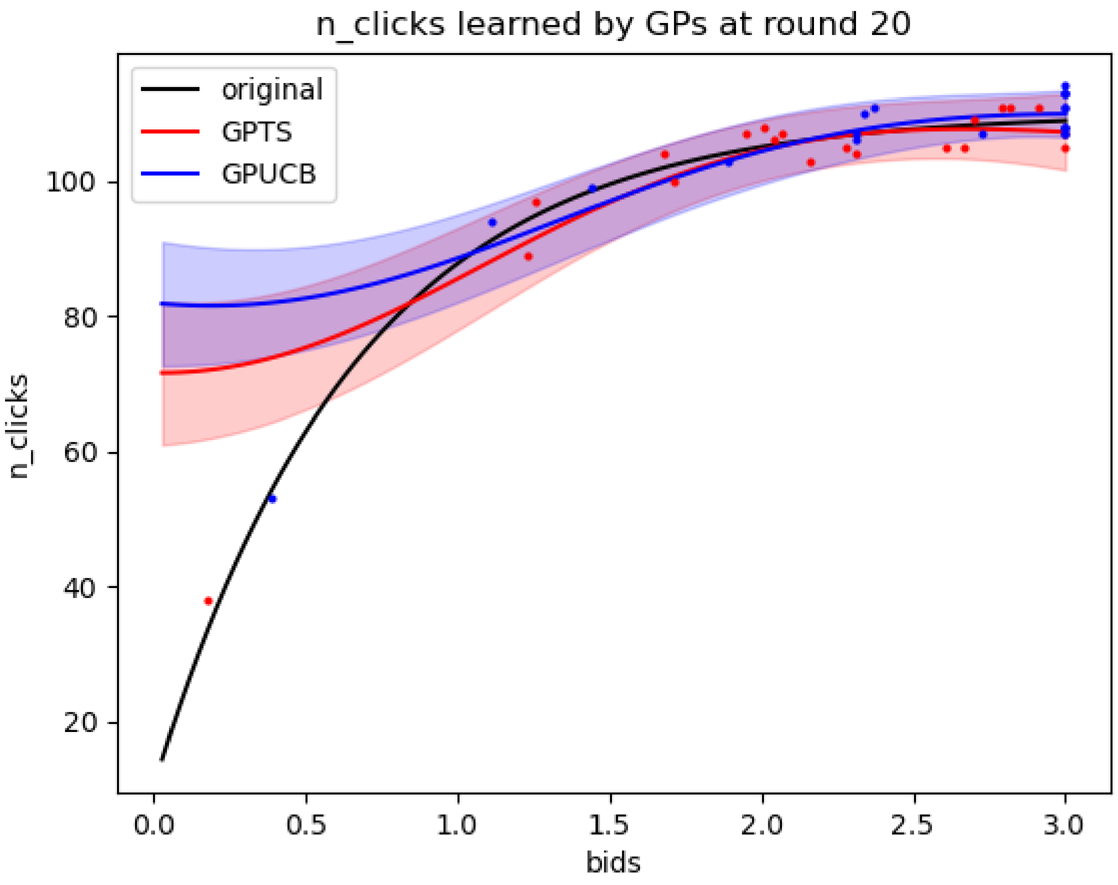


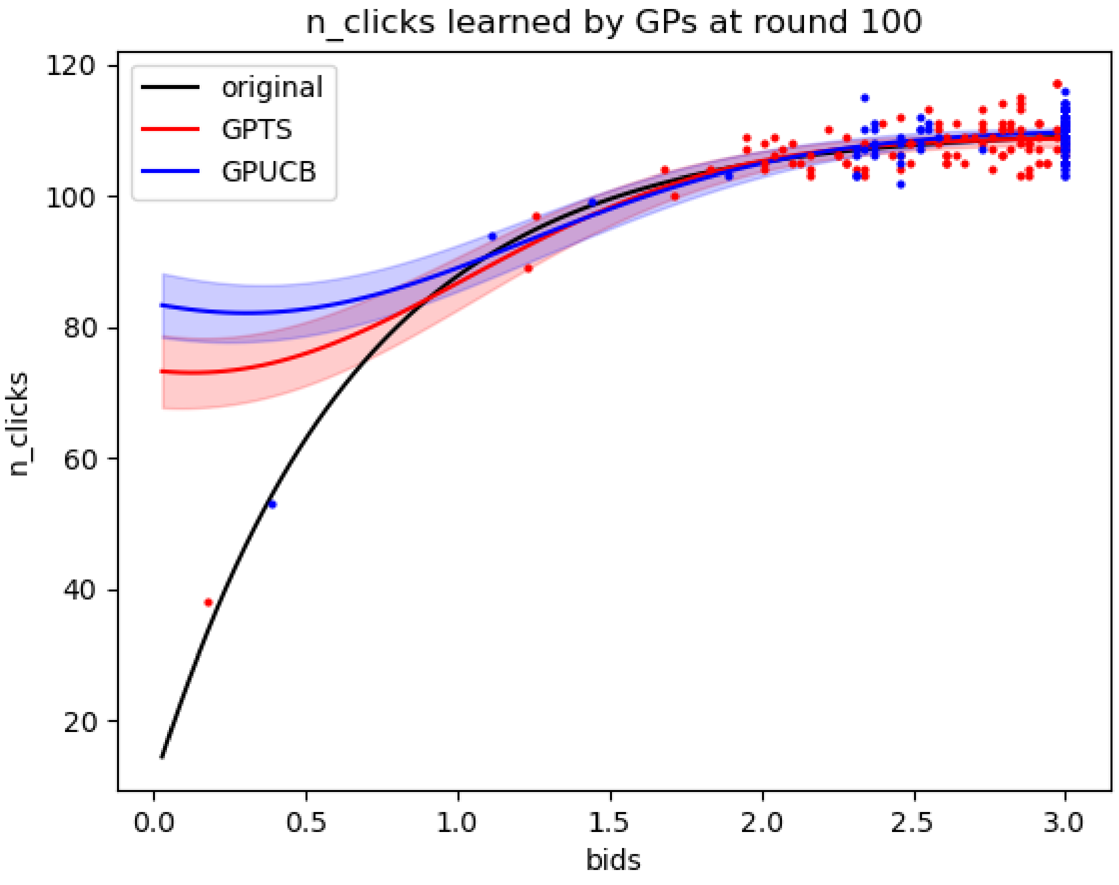
Step3 : Learning for joint pricing and advertising – Results



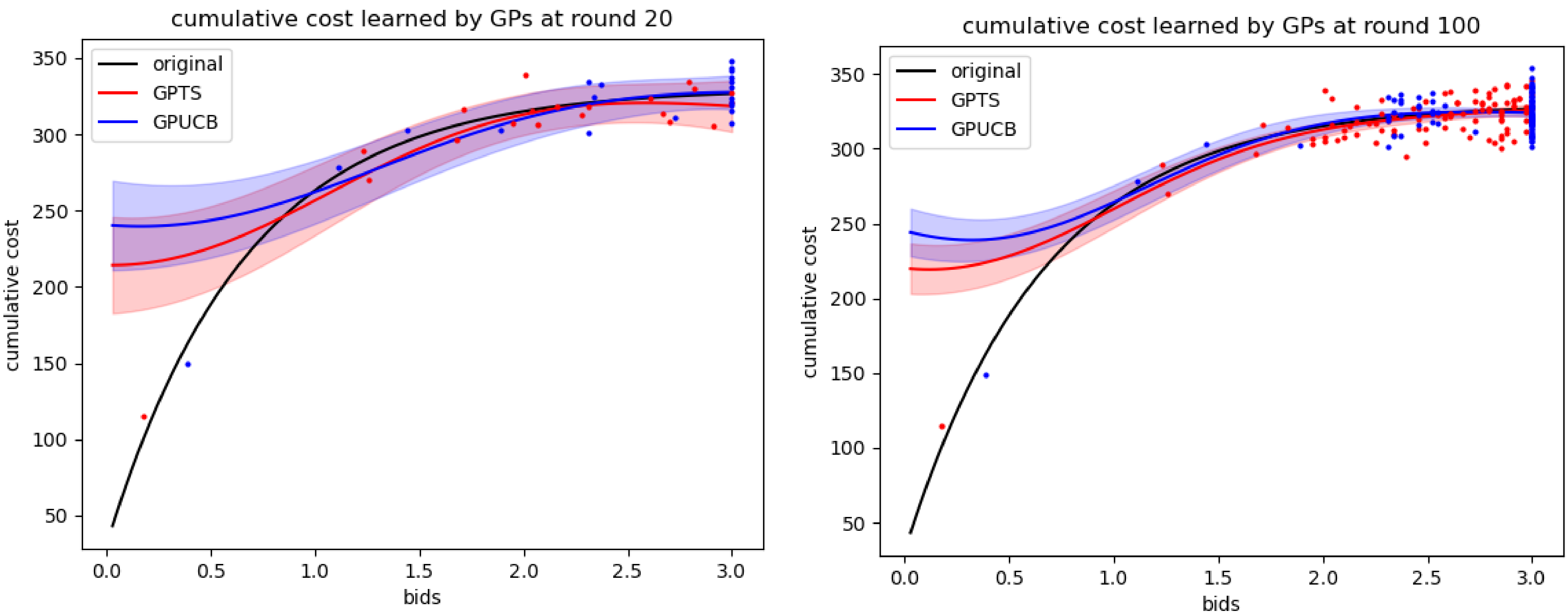


Step3 : Learning for joint pricing and advertising – Results





Step3 : Learning for joint pricing and advertising – Results



Step4 : Contexts and their generation– Request

The setting is the following:

- Consider three classes of users: **C1**, **C2**, and **C3**.
- The curves related to both the pricing and advertising problems are **unknown**

The request is to consider two scenarios:

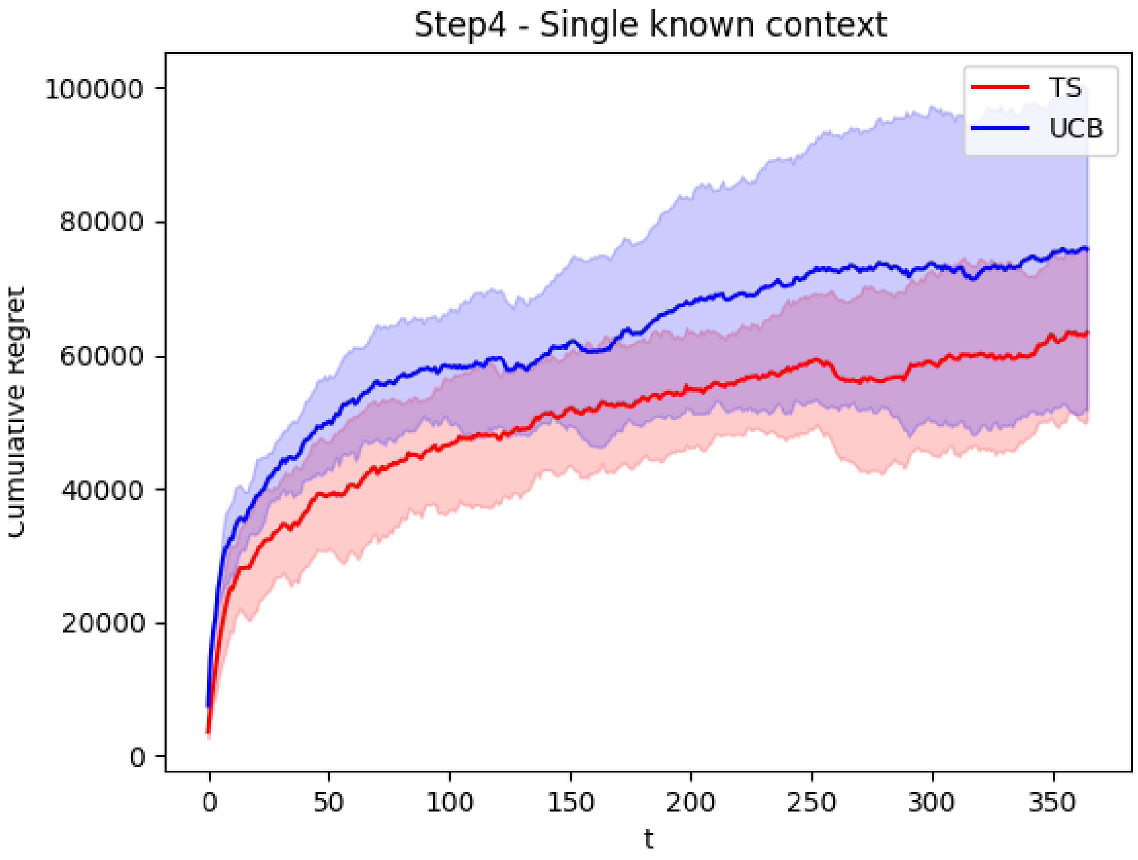
1. With the context structure **known beforehand**, apply the **GP-UCB** and **GP-TS algorithms** to estimate the curves of the **advertising problem**, and report the related plots.
2. With the context structure **not known beforehand**, by observing the features and data, apply the **GP-UCB** and **GP-TS** algorithms when using GPs to model the two advertising curves paired with a context generation algorithm. Apply the **context generation algorithms** every two weeks of the simulation.
   - Run the GP-UCB and GP-TS algorithms **without context generation** and **compare their performance** with the performance of the previous algorithms used for the second scenario

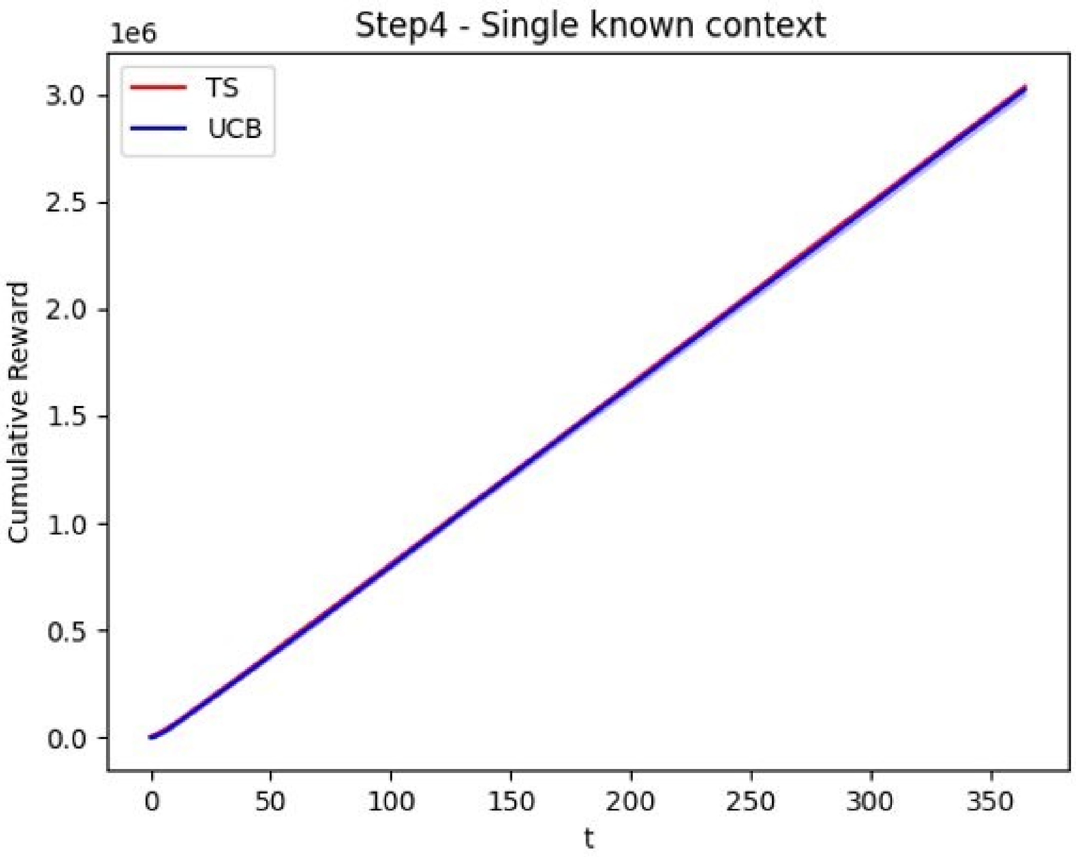
Step4 : Contexts and their generation– Solution Scenario 1

- We assigned a **regret minimizer** to **each class** of users, optimizing each class independently, pulling at each round a price and a bid for each class of users

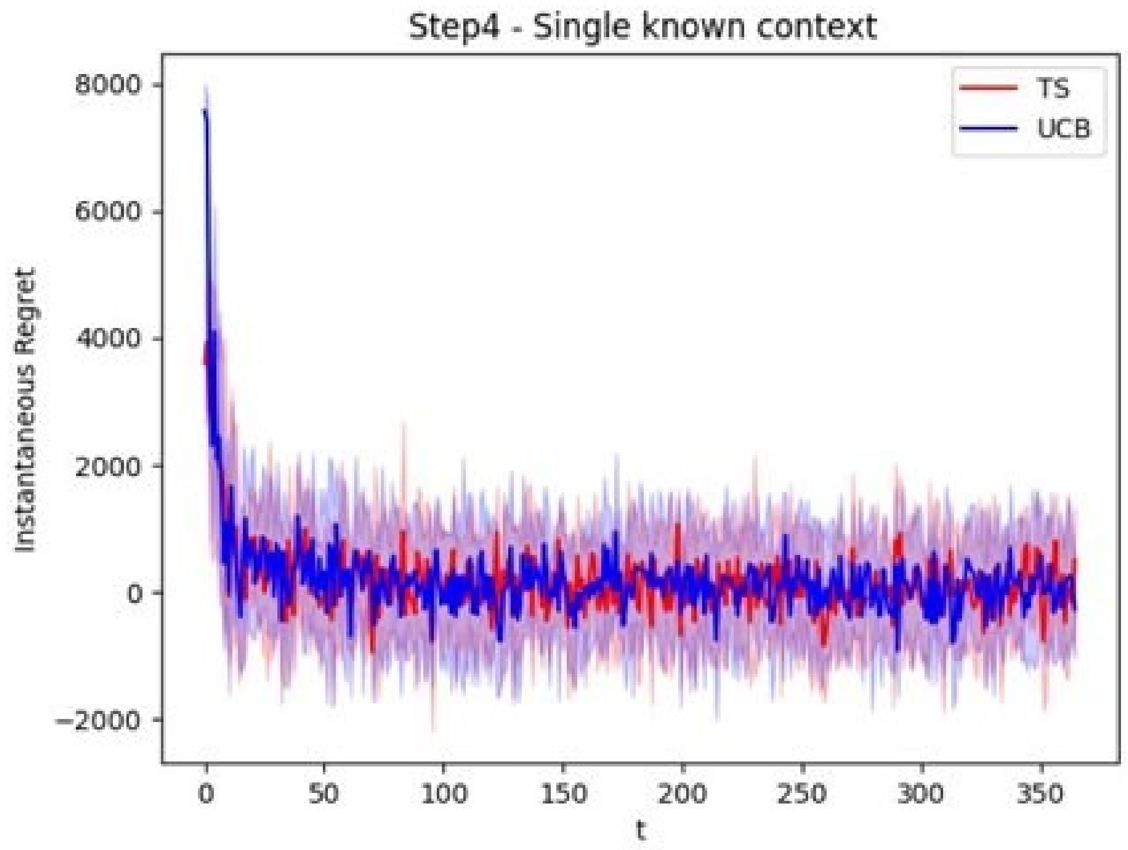
- The regret is then defined as the **sum of the regrets** of the three regret minimizers.

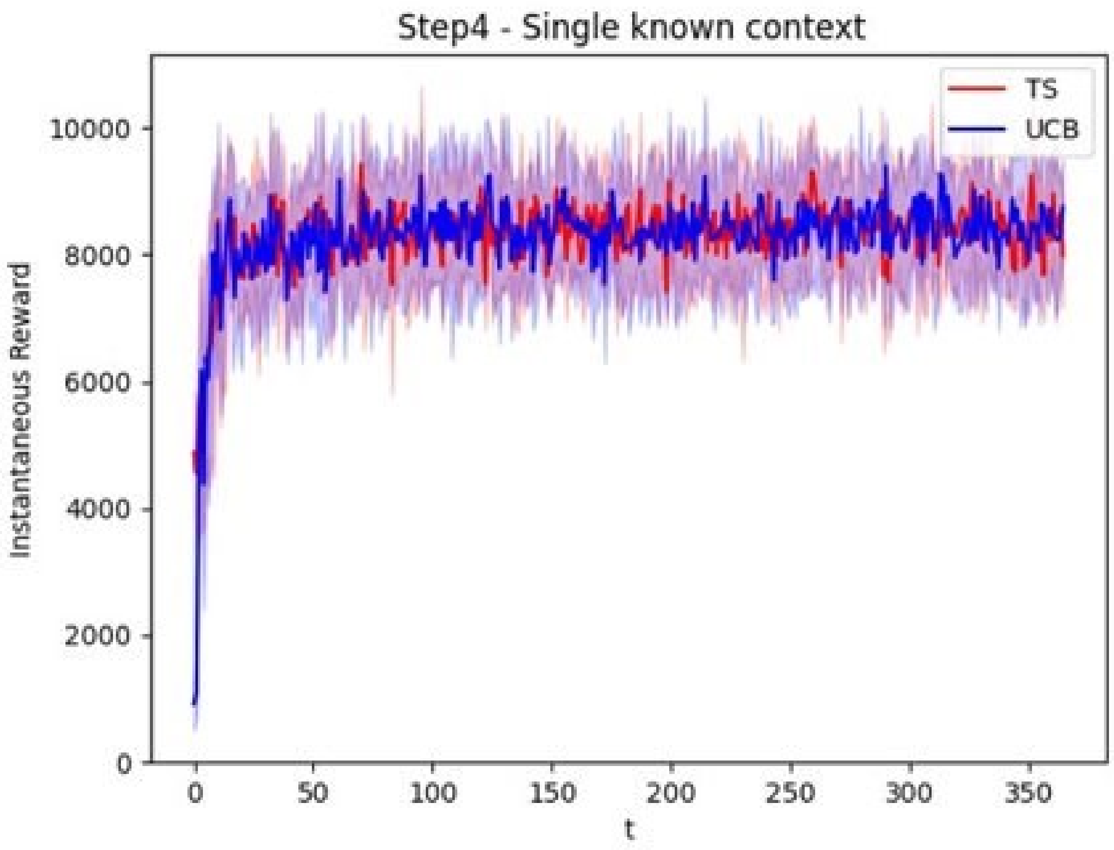
Step4 : Contexts and their generation– Results Scenario 1





Step4 : Contexts and their generation– Results Scenario 1





Step4 : Contexts and their generation– Solution Scenario 2

- We start by working on an **aggregated** model, with one single regret minimizer.
- Every two weeks a **context generation algorithm** decides to build the new context structure from zero, using **all the samples** collected from time t = 0.
- Thus, at each round and for each combination of features, we **save the pulled arms** (price and bid), the **number of clicks**, the **cumulative cost**, and the **number of positive conversions** associated with that combination of features.
- We have chosen to use the **greedy algorithm** since we have binary features. The algorithm builds a feature tree based on the following split condition: $\underline{p}_{c1}\underline{\mu}_{a_{c1}*,c1} + \underline{p}_{c2}\underline{\mu}_{a_{c2}*,c2} > \underline{\mu}_{a_{c0}*,c0}$
  - C_0 is the **current context structure** in a node of the features tree
  - $\{C_1, C_2\}$ is the context structure that will be obtained by **splitting** on a certain feature
  - $\underline{p}_c$ is the **lower bound** of the **probability** of occurrence of a user that belongs to the context c
  - $\underline{\mu}_{a_{c}*,c}$ is the **lower bound** of the **reward** of the optimal arm of context *c*

Step4 : Contexts and their generation– Solution Scenario 2

- In order to obtain the bound on the reward of the optimal arm for a certain context, we reasoned with a **two-step procedure** by computing first the optimal price and then the optimal bid :

  1. The lower bound on the conversion rate for a price p at time t for a context c can be obtained using the **Hoeffding bound** for a **Bernoulli** distribution:

$$CR_p = \frac{\sum_{k=1}^{t} nconv_{k,c}}{\sum_{k=1}^{t} nclicks_{k,c}} - \sqrt{-\frac{\log \delta}{2\sum_{k=1}^{t} nclicks_{k,c}}}$$

The optimal price has the **highest product** beween the price and the lower bound of the conversion rate .

$$p^* \in \arg\max_p CR_p \cdot (p - cost)$$

Step4 : Contexts and their generation– Solution Scenario 2

2. For the bid, we have two parameters to estimate: the **number of clicks** and the **cumulative cost**:

- We have considered the **lower bound** of the number of clicks and the upper bound of the cumulative cost (since the cumulative cost has a minus in the formula of the reward).

- Both are assumed to be Gaussian variables, thus we have obtained them using the bounds of **95% confidence interval** for a **Gaussian** process.

- The **lower bound** of cumulative cost is defined as: $NC_b = \frac{1}{t}\sum_{k=1}^{t} nclicks_{k,c} - 1{,}96\frac{\sigma_{nclicks_c}}{\sqrt{t}}$

- The **upper bound** of cumulative cost is defined as : $CO_b = \frac{1}{t}\sum_{k=1}^{t} cumcost_{k,c} + 1{,}96\frac{\sigma_{cumcost_c}}{\sqrt{t}}$

Step4 : Contexts and their generation– Solution Scenario 2

- The optimal bid is defined as the one that **maximises** the total reward per click :

$$b^* \in \arg\max_p \frac{1}{NC_b}\left[CR_{p^*}NC_b(p^* - cost) - CO_b\right]$$

- The lower bound for the optimal reward per click for context c is given by :

$$\underline{\mu}_{a_{c^*},c} = \frac{1}{NC_{b^*}}\left[CR_{p^*}NC_{b^*}(p^* - cost) - CO_{b^*}\right]$$

- In order to compute the lower bound on the probability of the context c_1 (or c_2), we used the **Hoeffding bound**:

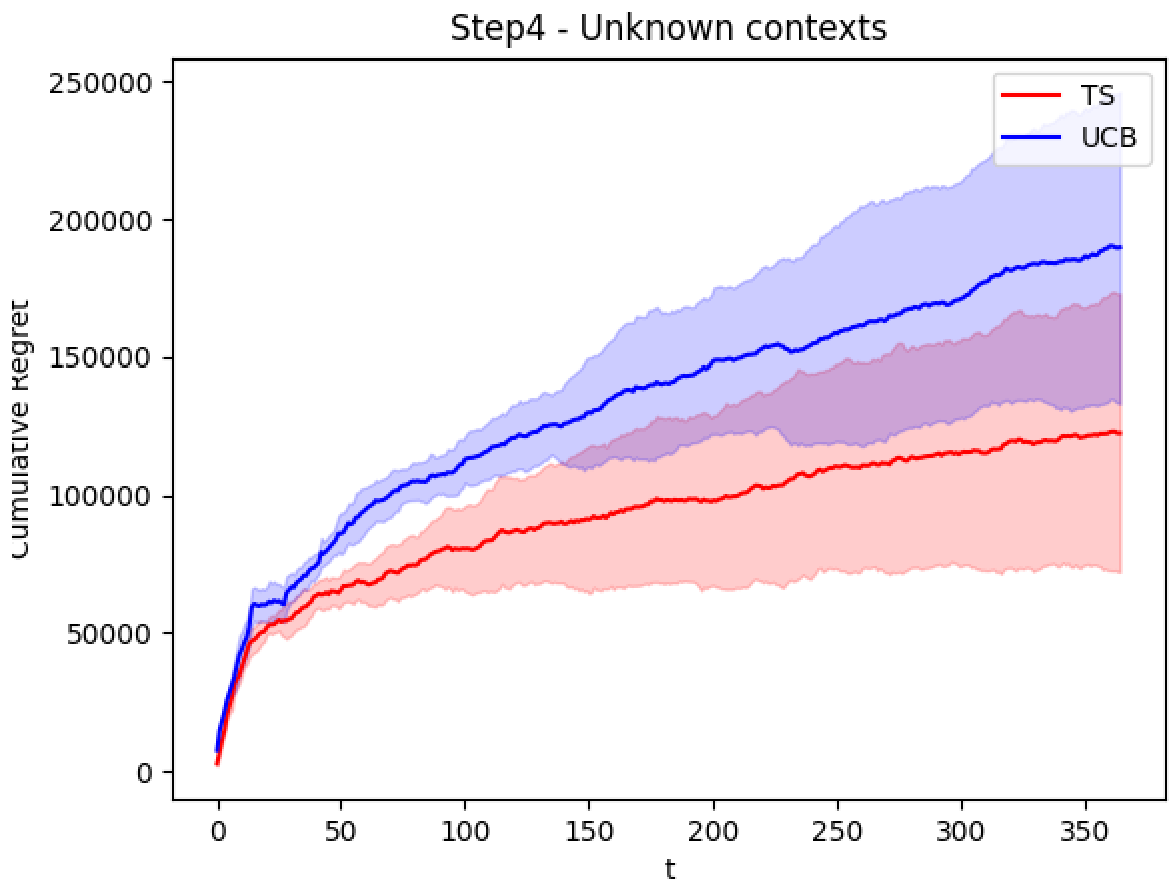
$$\underline{p}_{c_1} = \frac{\sum_{k=1}^{t} nclicks_{k,c_1}}{\sum_{k=1}^{t} nclicks_{k,c_1} + \sum_{k=1}^{t} nclicks_{k,c_2}} - \sqrt{-\frac{\log\delta}{2\sum_{k=1}^{t} nclicks_{k,c_1}}}$$

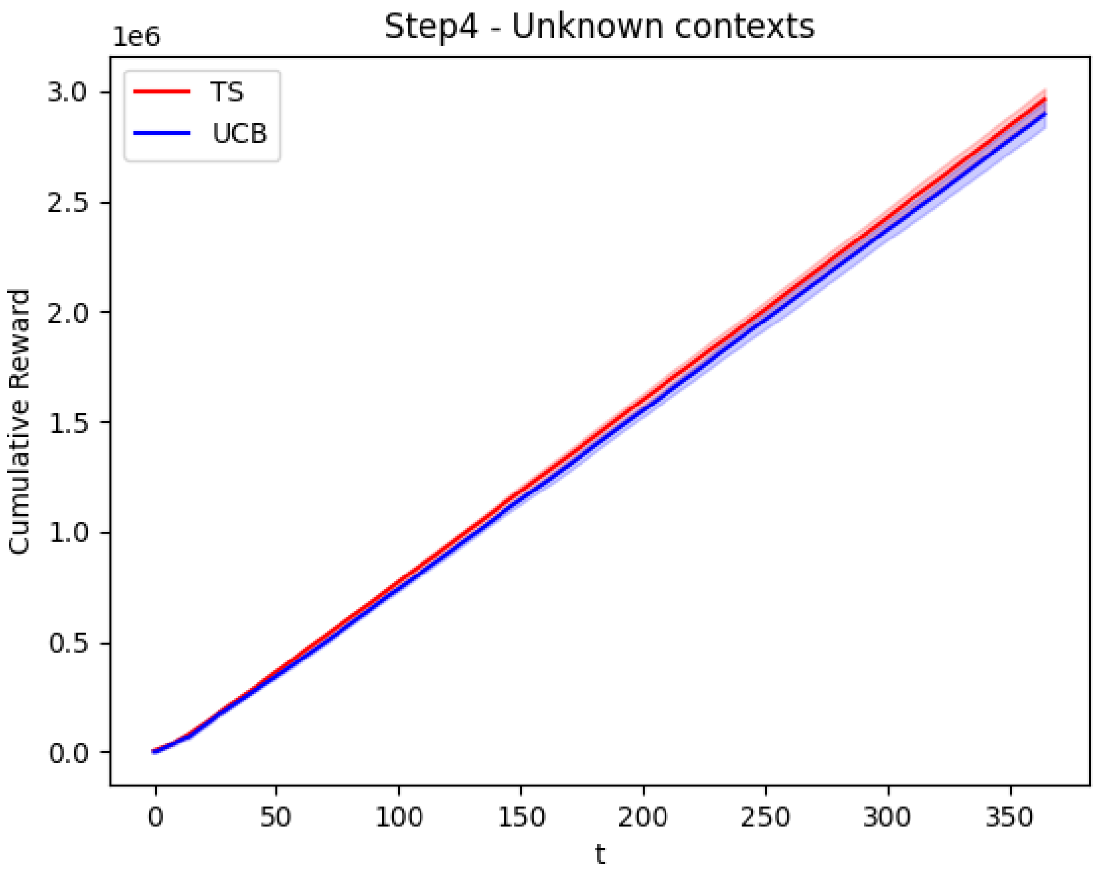
Step4 : Contexts and their generation– Solution Scenario 2

- After the execution of the **greedy algorithm**, we assign to each context a regret minimizer:
  - if the context is new, a new **regret minimizer** is assigned to it
  - otherwise the **previous regret minimizer** will continue to be associated with it.
- If a contest is not present in the new structure its regret minimizer will be **deleted**.
- The new regret minimizers will **not** start from zero; indeed, we initialize them with all the past samples belonging to the associated context, performing a **bulk update** after creating them.

Step4 : Contexts and their generation – Results

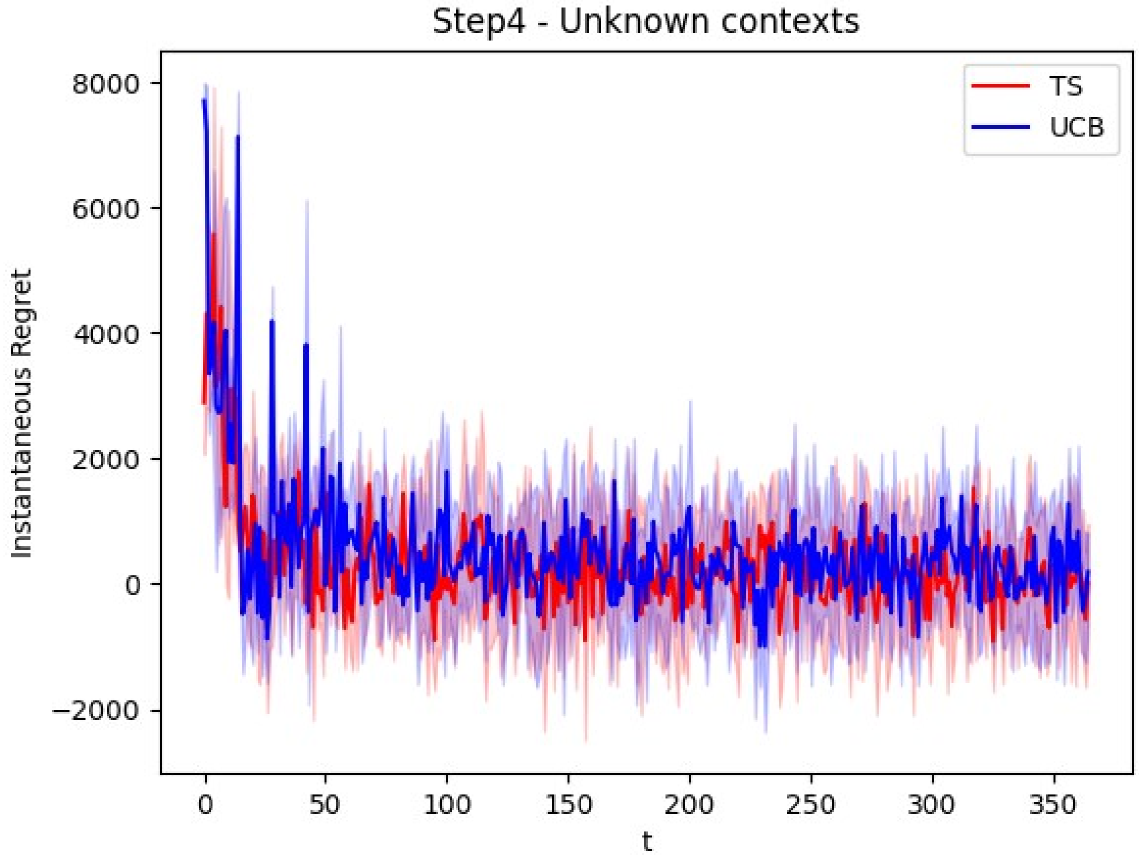
With context generation

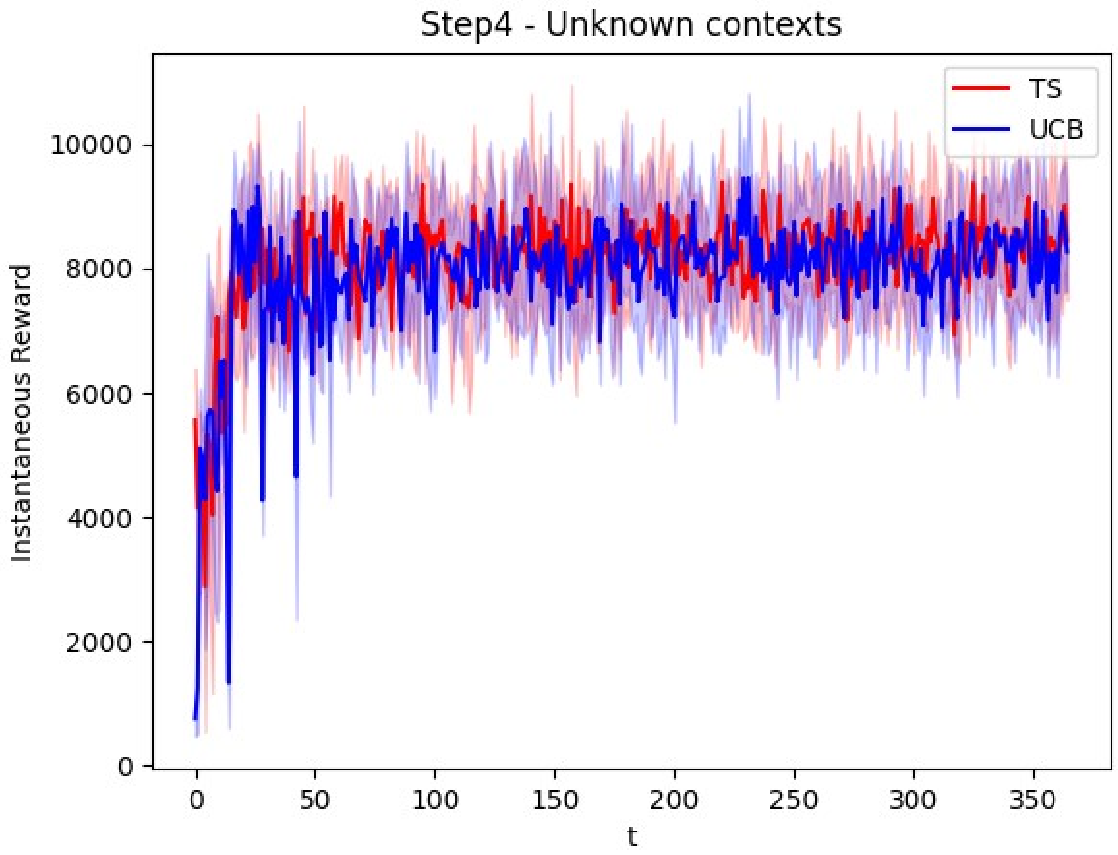




Step4 : Contexts and their generation – Results

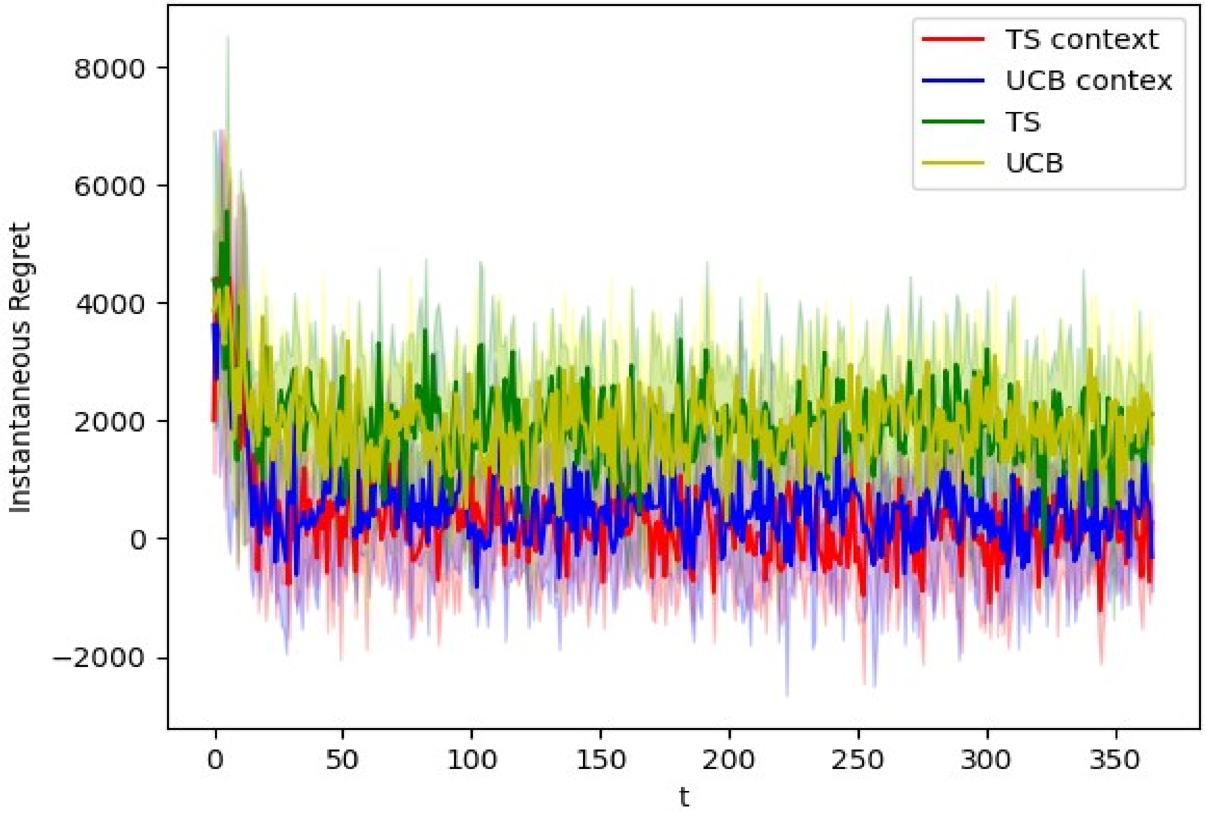
With context generation

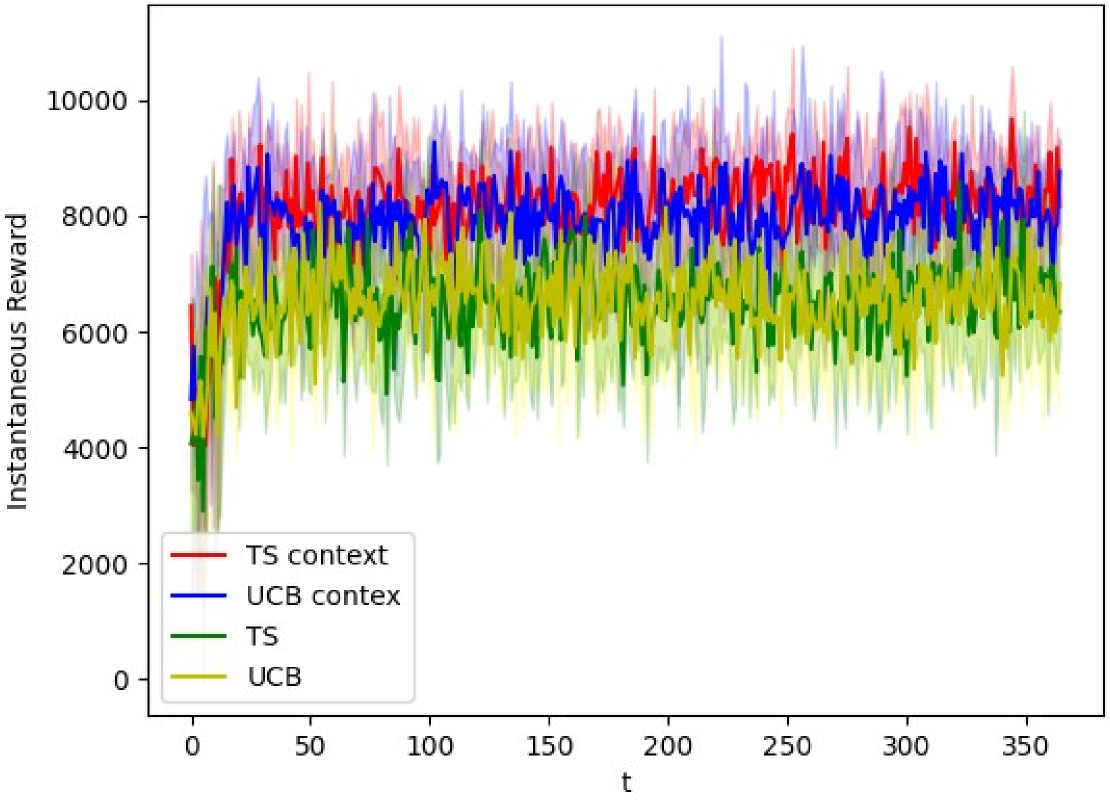




Step4 : Contexts and their generation – Results

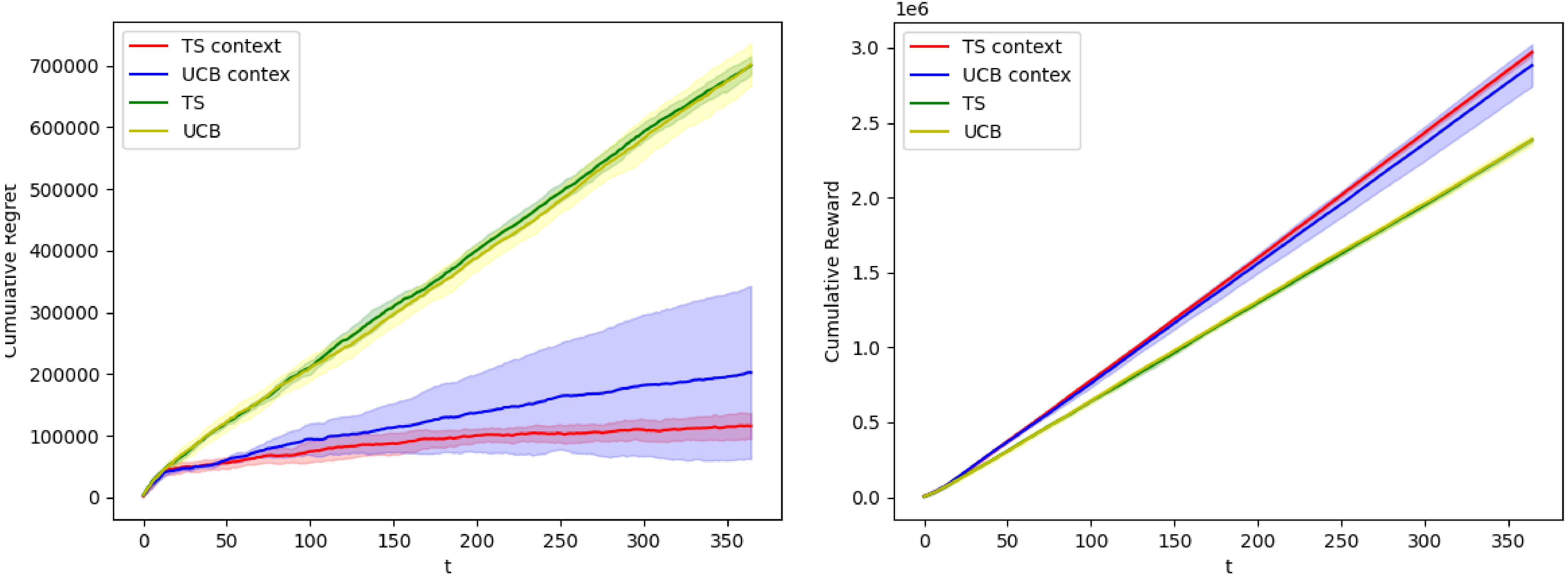
Without context generation





Step4 : Contexts and their generation – Results

Without context generation



Step5 : Dealing with non-stationary environments with two abrupt changes - Request

The setting is the following:

- Consider all users belonging to class **C1**.
- The curves related to the advertising part of the problem are **known**
- The curve related to the pricing problem is **unknown**, **non-stationary** and have **three different phases**.

The request is to:

- Apply the **UCB1 algorithm with sliding windows** to estimate the curve of the princing problem
- Apply the **UCB1 algorithm with change detection** to estimate the curve of the princing problem
- Provide a **sensitivity analysis** for the parameters of the algorithm.
- Plot the average value and standard deviation of the **cumulative regret**, **cumulative reward**, **instantaneous regret**, and **instantaneous reward**

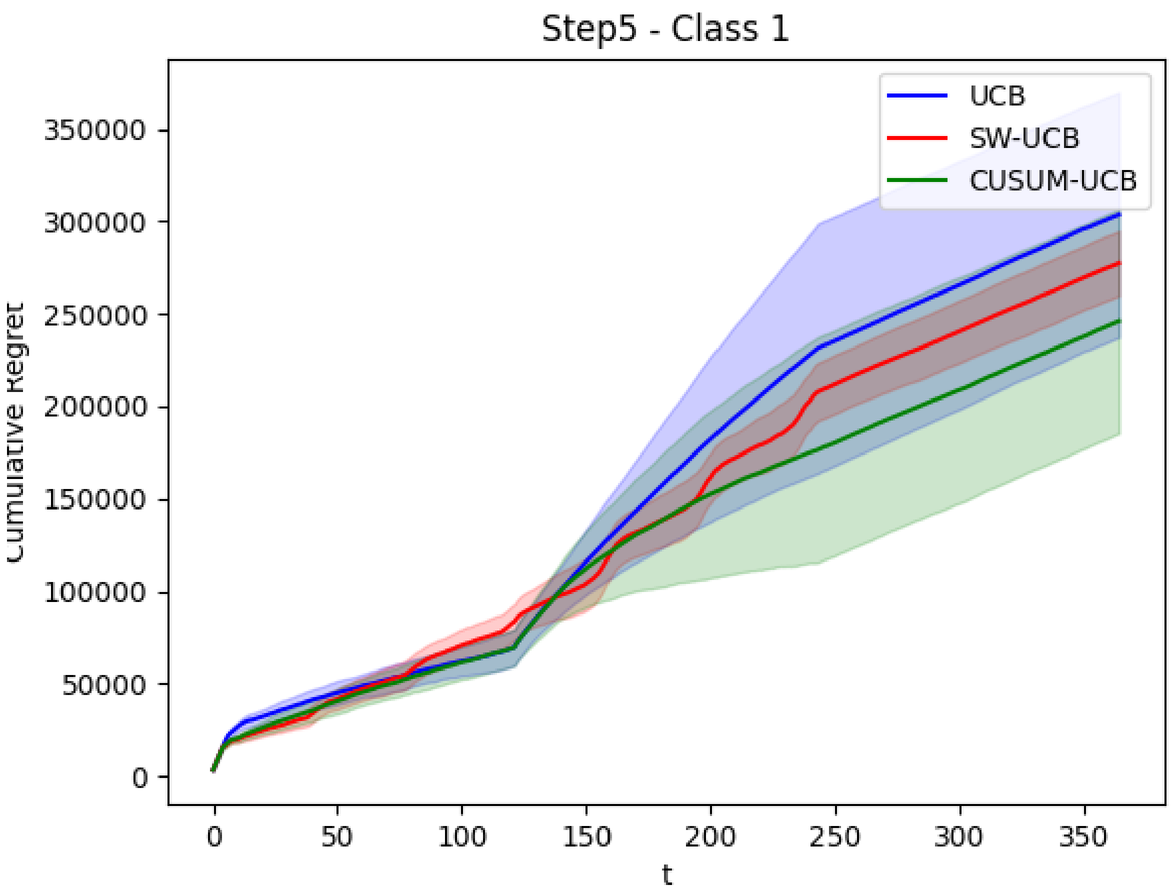
Step5 : Dealing with non-stationary environments with two abrupt changes - Solution

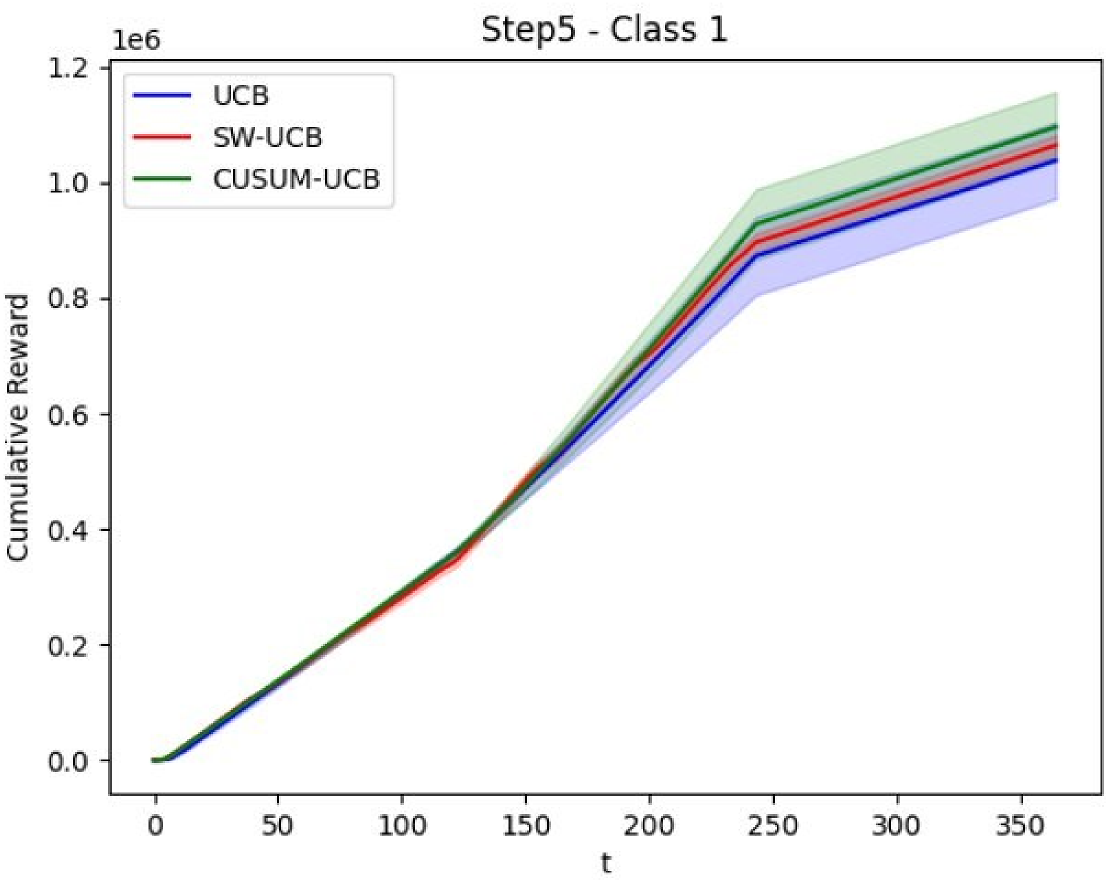
- Since the price is known, we used an optimizer to learn the price giving the **highest** reward computed using the **real values** of the number of clicks and cumulative cost given by the bid related to the chosen price
- We associated each phase with a different time of the year, having a time horizon of 365 days starting from May.
  1. The **first** phase : **regular** purchasing where the preferred price is the **second cheapest.**
  2. The **second** phase: people move to **higher** cost, the preferred price is the **middle** one and every price meets an **increase** in their conversion rate.
  3. Finally, the **third** phase: characterized by **discounted** sales, the prefferd price is the cheapest with a huge peak in its conversion rate. while the others suffer a **reduction** in their conversion rate.

Step5 : Dealing with non-stationary environments with two abrupt changes - Solution

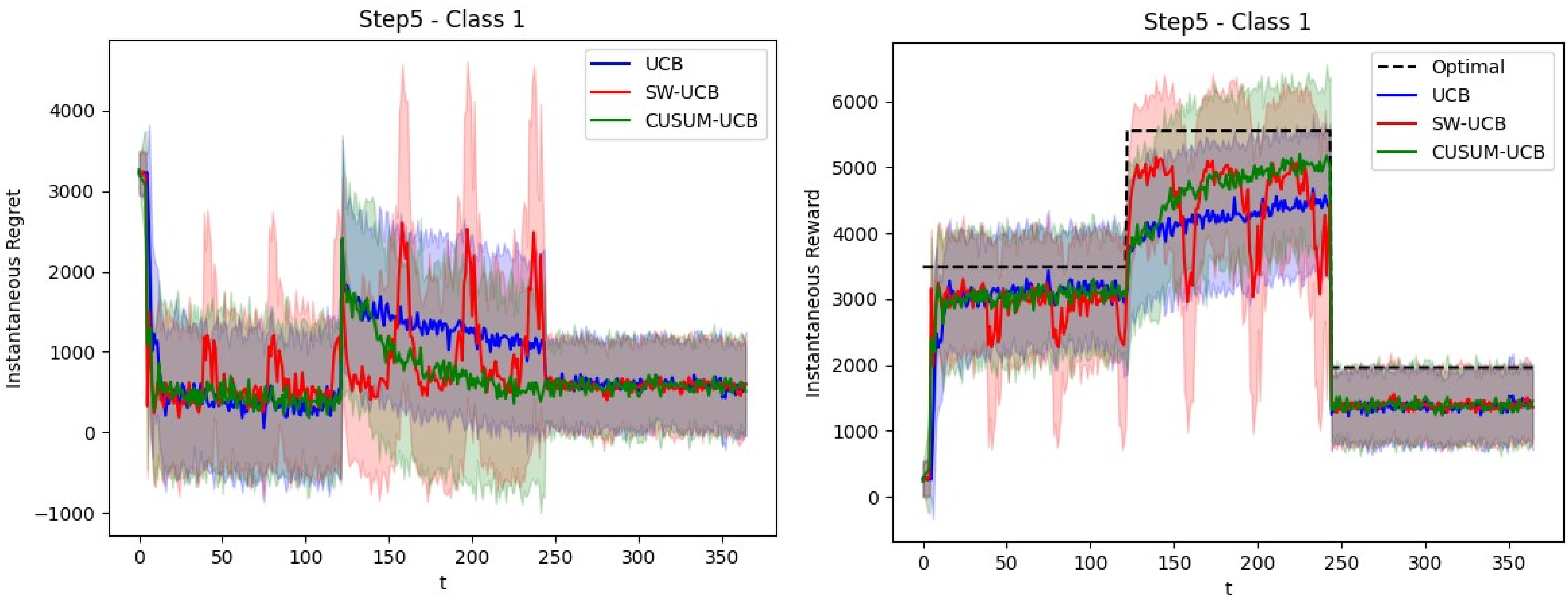
- To choose the parameters of the two variants of UCB1 we relied on theoretical suggestions:
  - For the **sliding window size**, we opted for a value directly proportional to $\sqrt{T}$
  - For the active change detection variant we used **CUSUM:**
    - change detection parameter : $\approx logT$
    - Exploration parameter $\approx \sqrt{\frac{logT}{T}}$
- We did not detach from the **theoretical suggestions**, and we did not perform any kind of tuning because both algorithms are deployed over a **single** case of functions. The **limitation** in the variety of cases analyzed could have been the cause of serious **overfitting**.

Step5 : Dealing with non-stationary environments with two abrupt changes - Results





Step5 : Dealing with non-stationary environments with two abrupt changes - Results



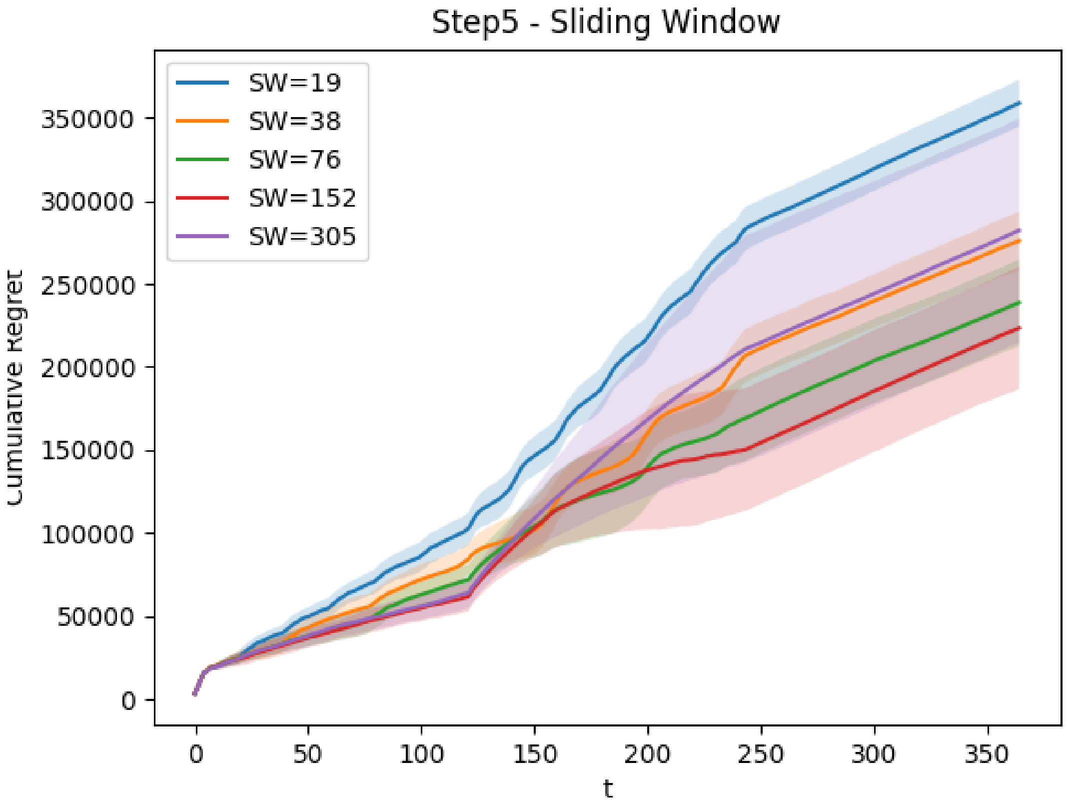
Step5 : Dealing with non-stationary environments with two abrupt changes - Sensitivity analysis

For the **sensitivity analysis**, we considered the following parameters:

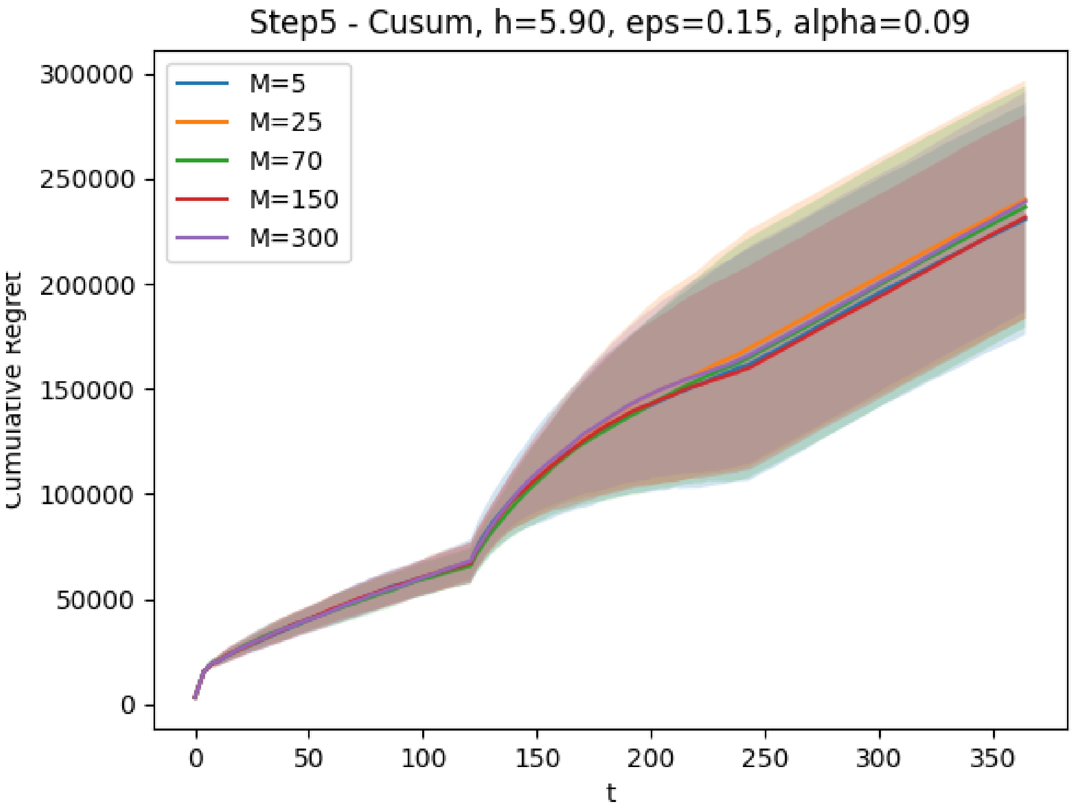
- UCB sliding window parameters tested:
  - **Window Size**
- UCB cusum parameters tested:
  - **M**: Number of samples to be used to compute the empirical mean that will be used as a reference for the current behavior in the actual change detection phase.
  - ***h**:* Threshold to detect any change in the customers' behavior.
  - ***ε**:* Critical level used to adjust the sensitivity of the change.
  - α*:* Amount of exploration of the algorithm

Step5 : Dealing with non-stationary environments with two abrupt changes - Sensitivity analysis

Sliding Window: Window size analysis.

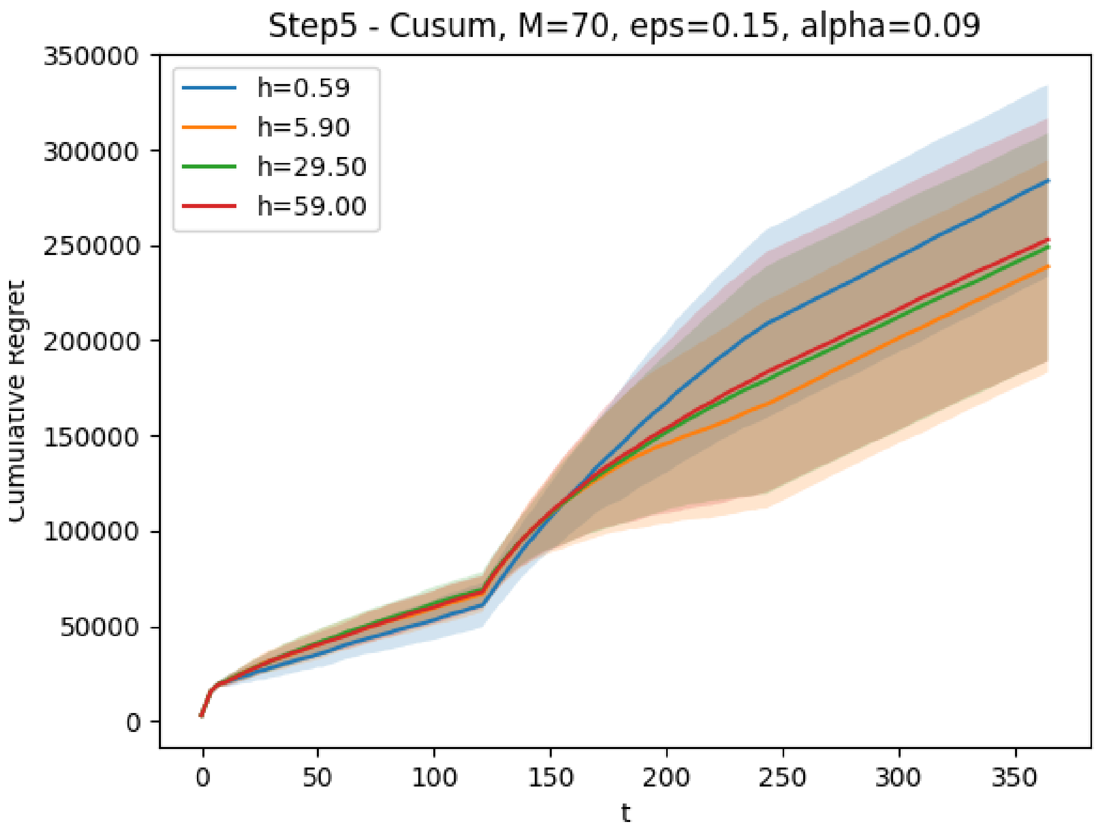


CUSUM: M

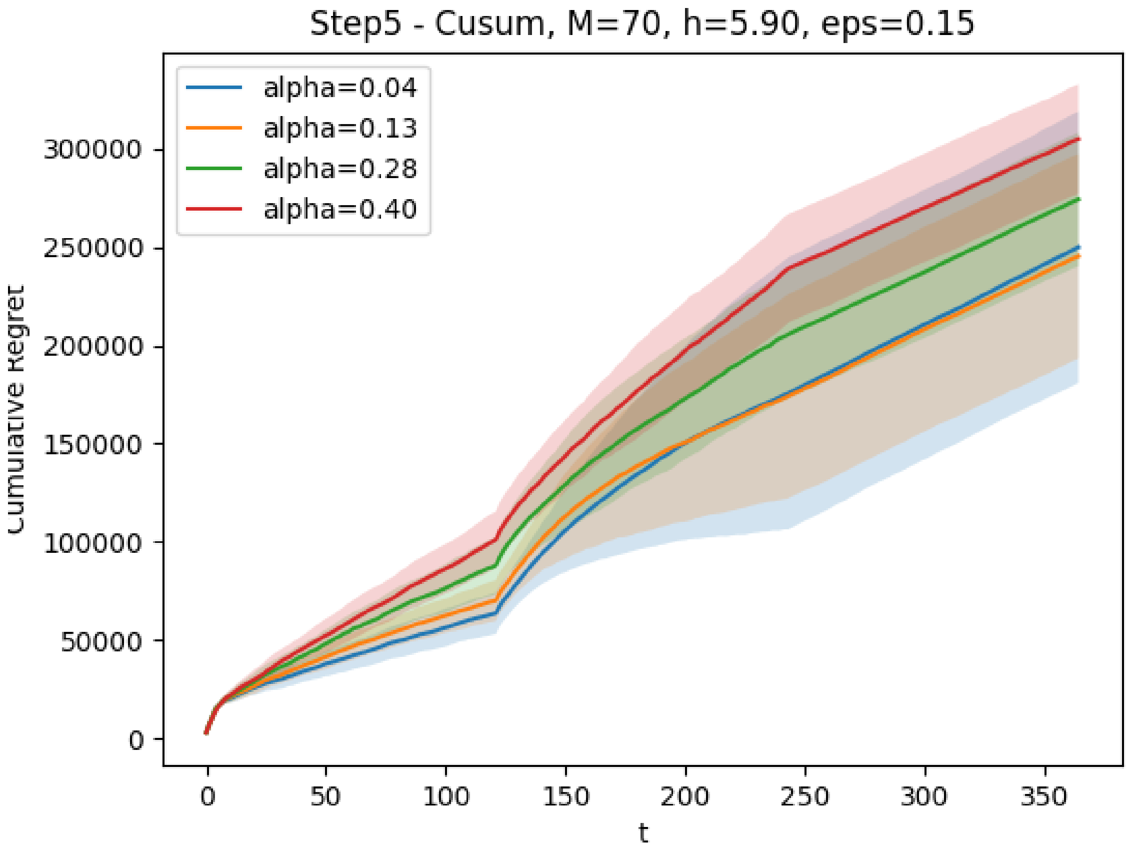


Step5 : Dealing with non-stationary environments with two abrupt changes - Sensitivity analysis

CUSUM: h

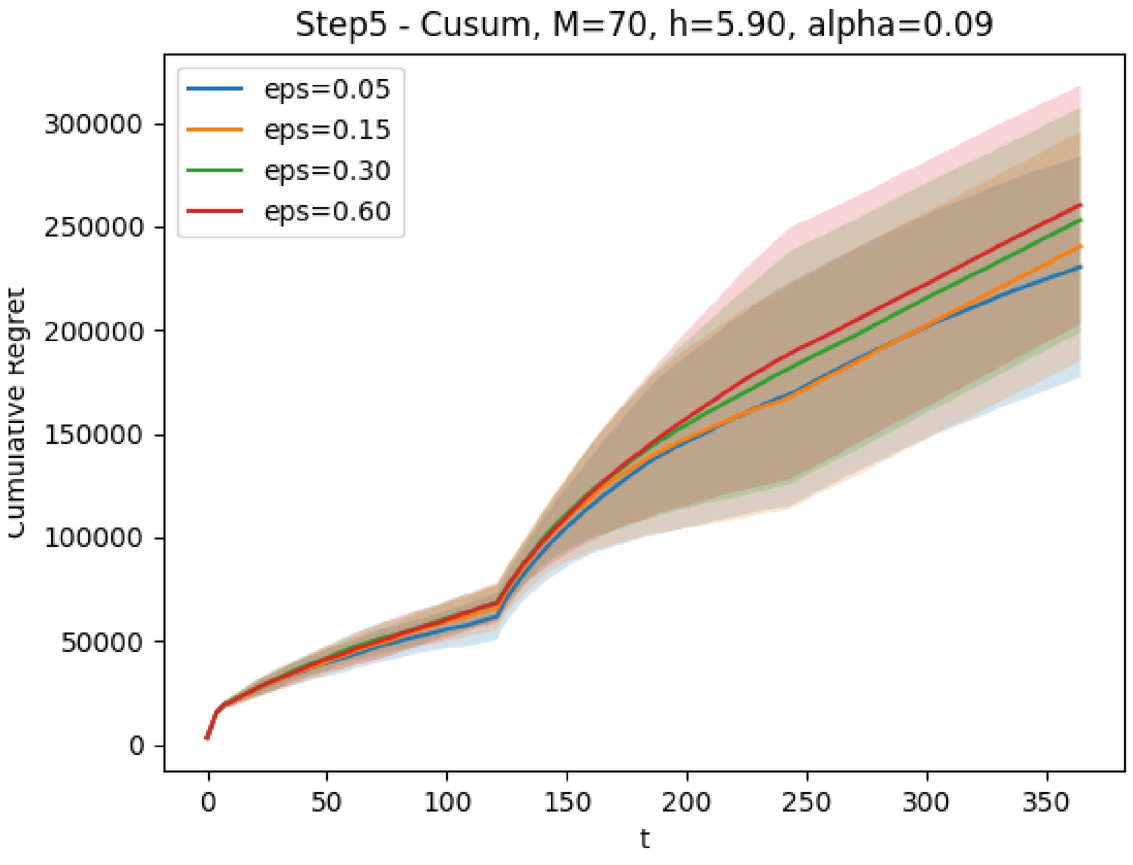


CUSUM: α



Step5 : Dealing with non-stationary environments with two abrupt changes - Sensitivity analysis

CUSUM: ε



Step6 : Dealing with non-stationary environments with many abrupt changes - Request

Two settings:

1. In the first one, the environment is **non-stationary**. It is a simplified version of Step5, with the **bid fixed**.

- Develop and apply the **EXP3 algorithm**, and verify that it performs **worse** than the two non-stationary versions of UCB1.

1. In the second one, the environment has a **higher degree of non-stationarity** with 5 phases that frequently change.

- Apply **EXP3**, **UCB1**, and the **two non-stationary flavors** of UCB1. Verify that EXP3 **outperforms** the non-stationary version of UCB1.

Step6 : Dealing with non-stationary environments with many abrupt changes - Solution

- A scenario with **many abrupt changes** poses **harder obstacles** in the learning process, that is is called an **adversarial bandit**.
- In an adversarial bandit setting we **cannot** expect to learn quantities changing over time. We can only try to play a good arm without the hope to learn anything useful from its reward. The parameters we need to estimate to **maximize** the monetary reward are the **conversion rates** which are unknown and rapidly changing.
- The EXP3 algorithm proposed is an algorithm designed to play in an **adversarial bandit setting**. At each round the arm to be played is selected by a **random draw** from a probability distribution over each arm computed at the previous step. The probability distribution of arm i at round t is computed by: $$p_i(t) = (1 - \gamma)\frac{w_i(t)}{\sum_{j=1}^{k} w_j(t)} + \frac{\gamma}{K}$$

where *K* is the number of arms, $\gamma \in (0,1)$ is a **hyperparameter** to tune, and $w_i(t)$ is the weight associated with arm *i* at round *t* and computed as: $$w_{i_t}(t+1) = w_{i_t}(t) \cdot e^{\frac{\gamma}{K}\hat{x}_{i_t}}$$

where i_t is the arm pulled at time *t* and $\hat{x}_{i_t}$ the expected reward obtained from that arm at time *t*: $\hat{x}_{i_t} = \frac{x_{i_t}}{p_{i_t}}$.

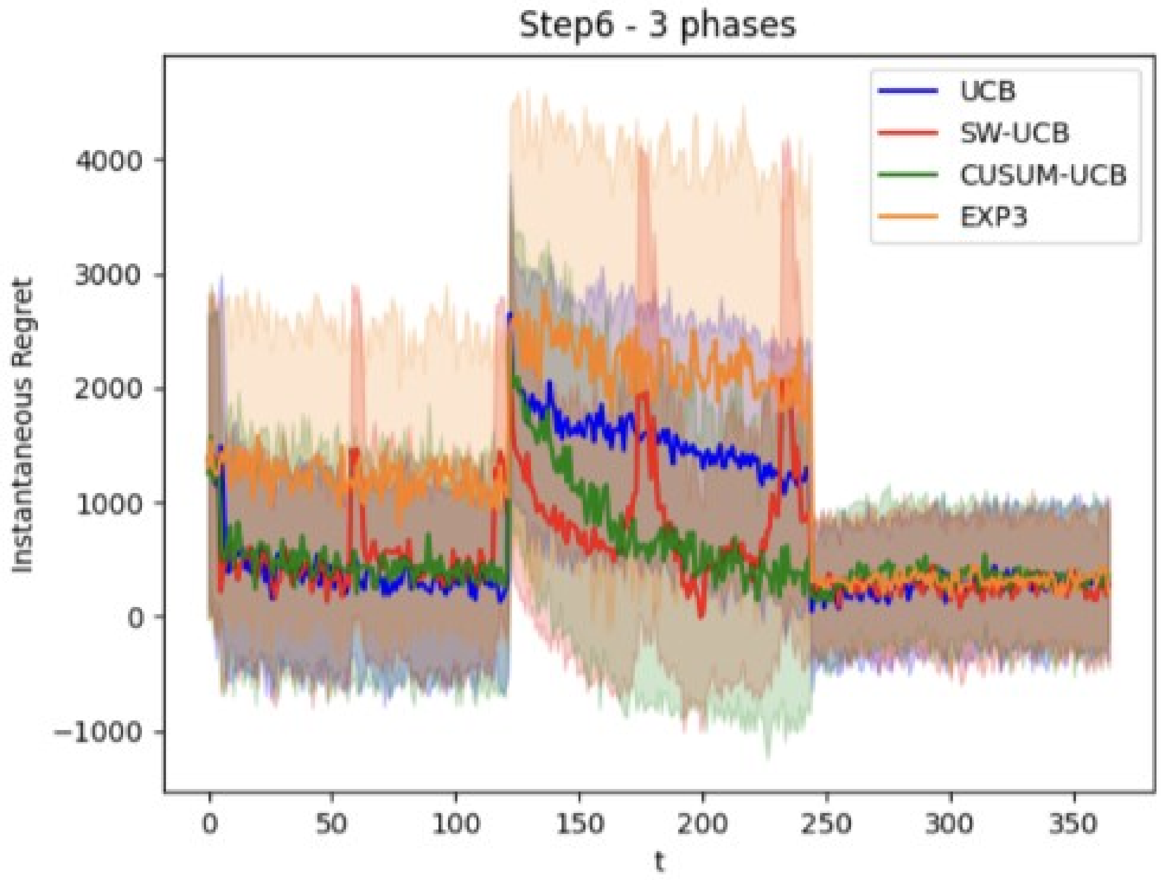
Step6 : Dealing with non-stationary environments with many abrupt changes - Solution

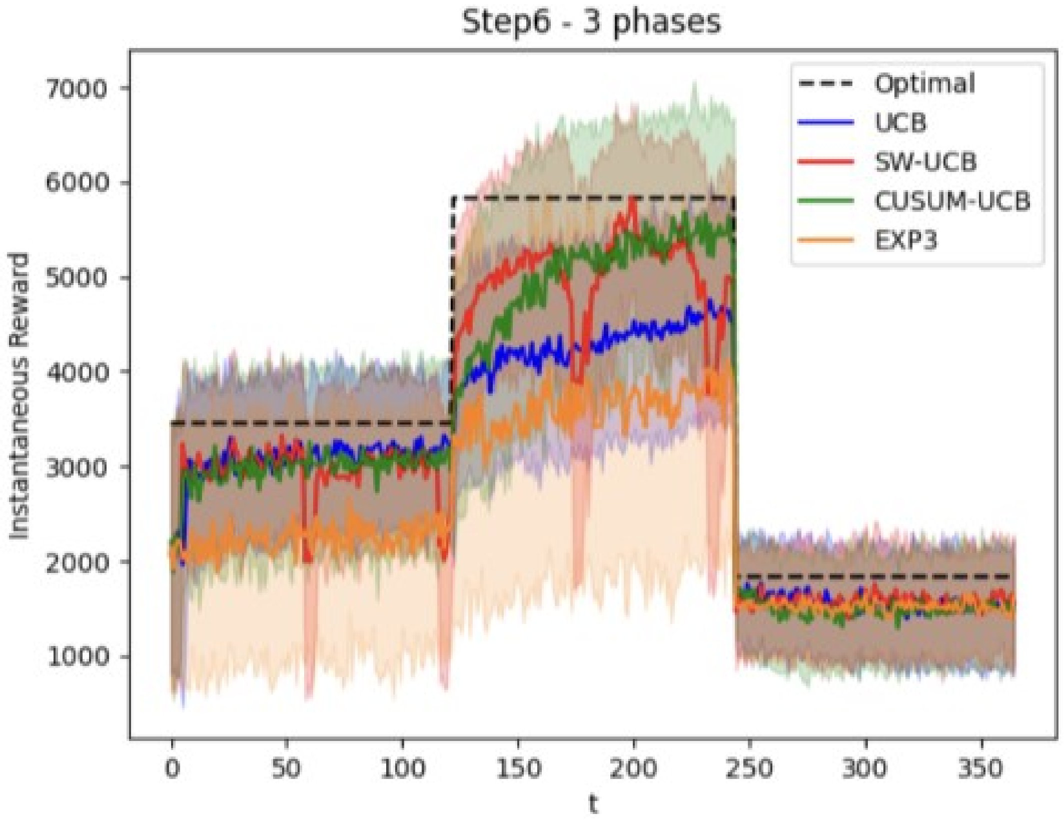
- Weights are **initialized** to 1 for each arms, and rewards are **scaled** to the interval [0, 1].

- γ must be **tuned**. If it is closer to 1, the arms tend to be uniform, while if it is closer to 0 the algorithm gives more probability according to the weights and therefore according to the rewards obtained from the game.

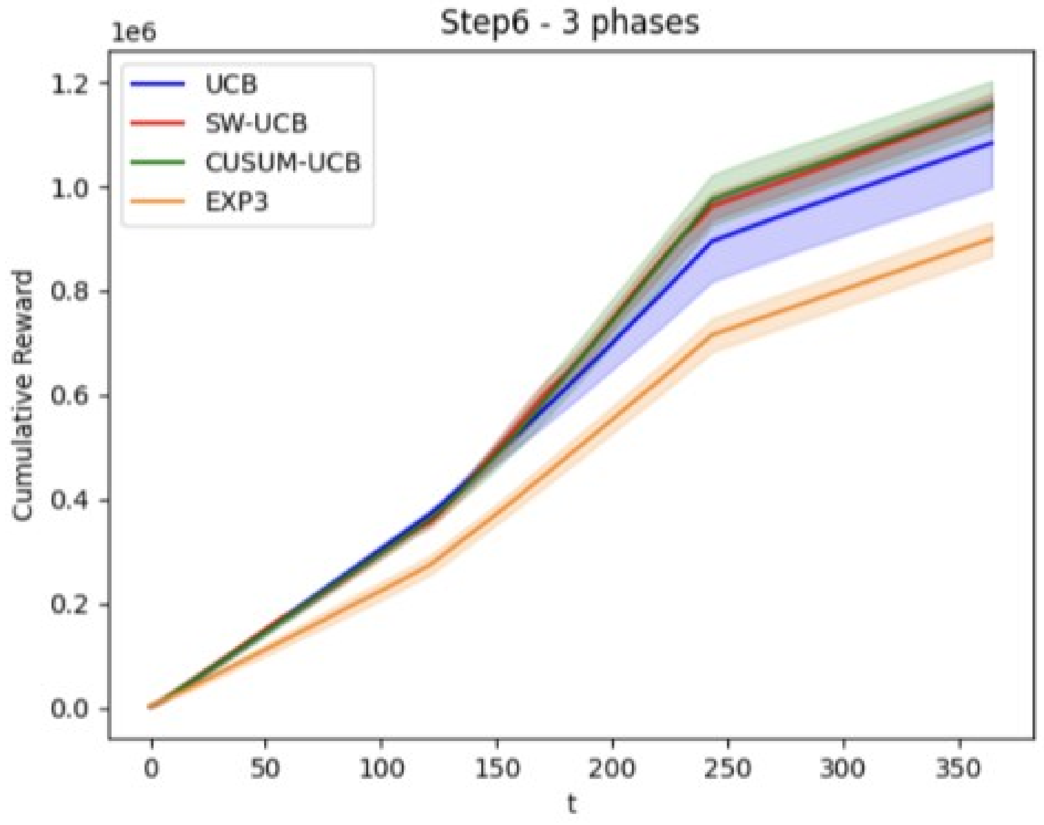
- EXP3 being an adversarial bandit algorithm presents, as said before, **learning inclinations**. In fact when an arm is selected its weight increases if the reward is high and by consequence the probability of drawing it is higher. Compared to classical bandit, such as TS or UCB, EXP3 explores more between all the arms. This behavior is needed to **compensate** for the lack of fixed rewards in time.

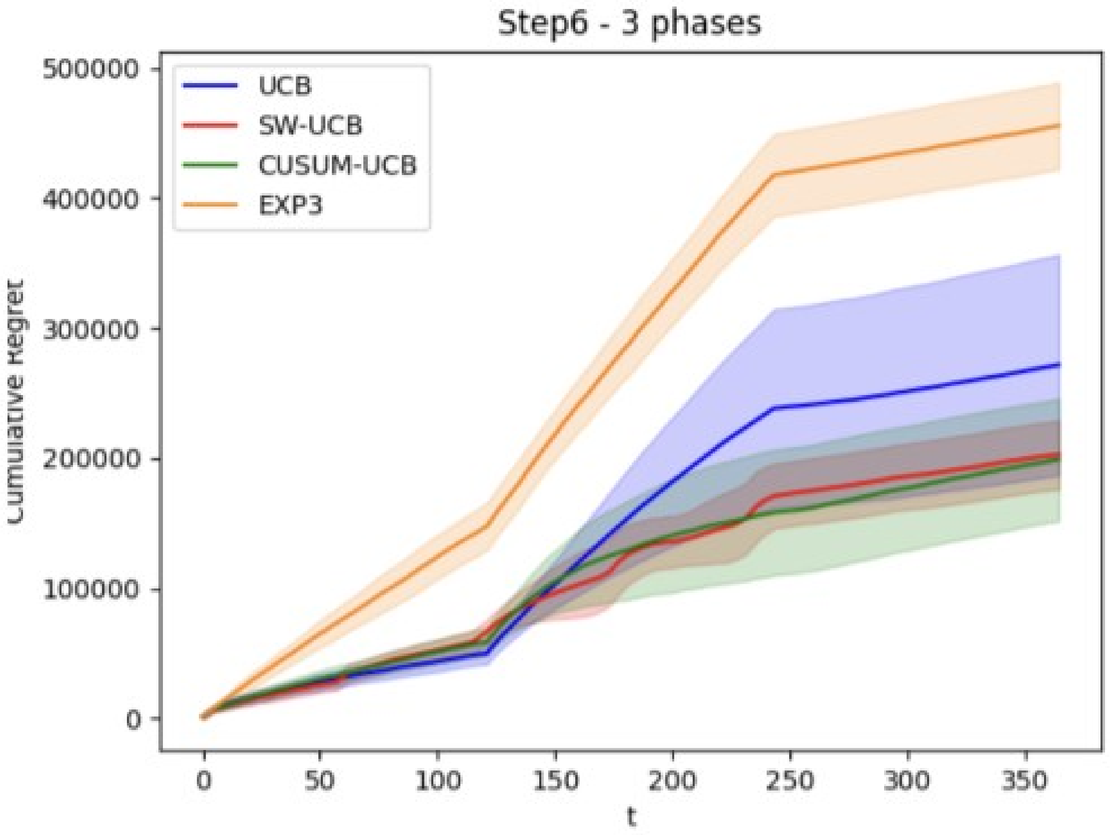
Step6 : Dealing with non-stationary environments with many abrupt changes - Results (Low Frequency)



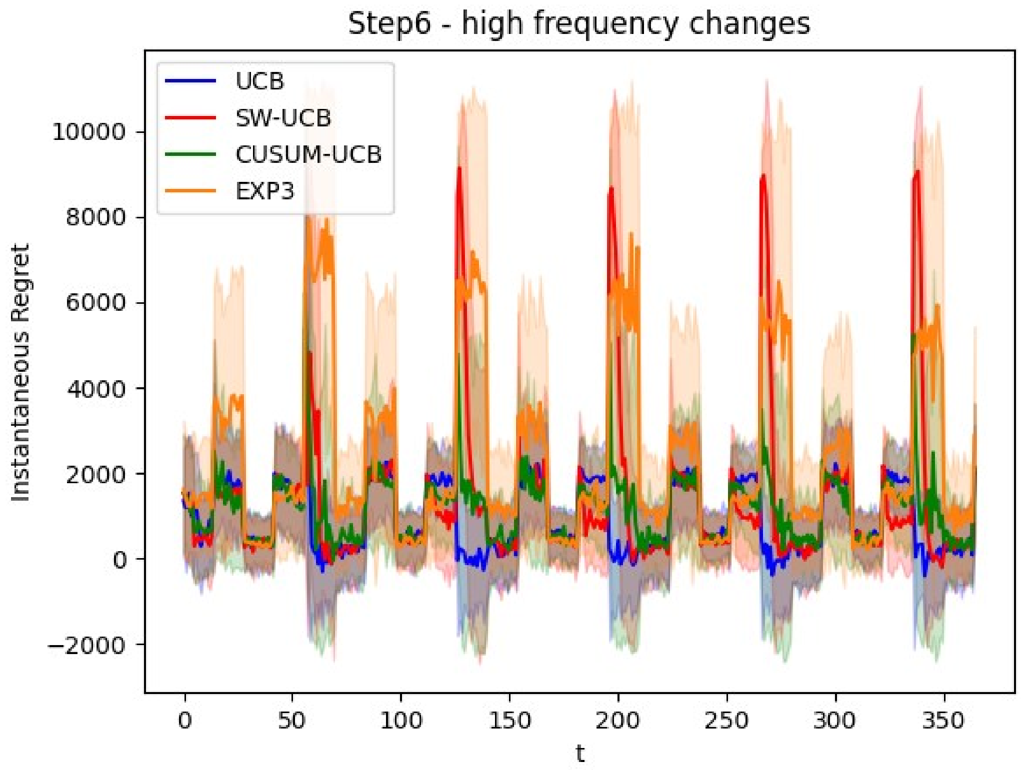


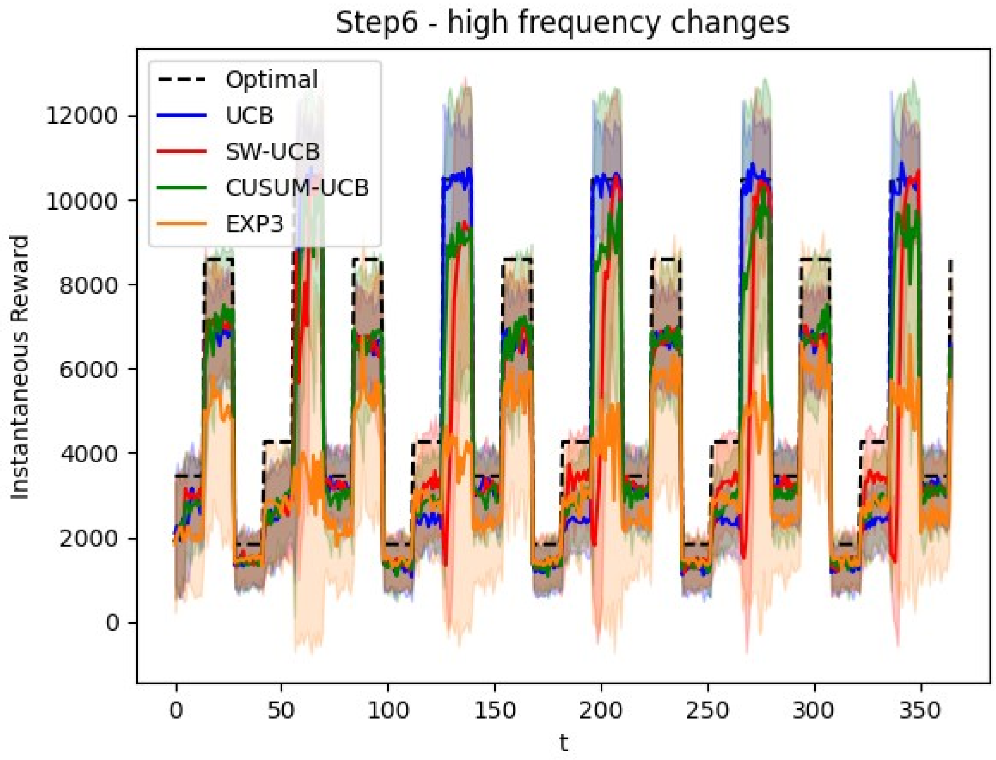
Step6 : Dealing with non-stationary environments with many abrupt changes - Results (Low Frequency)



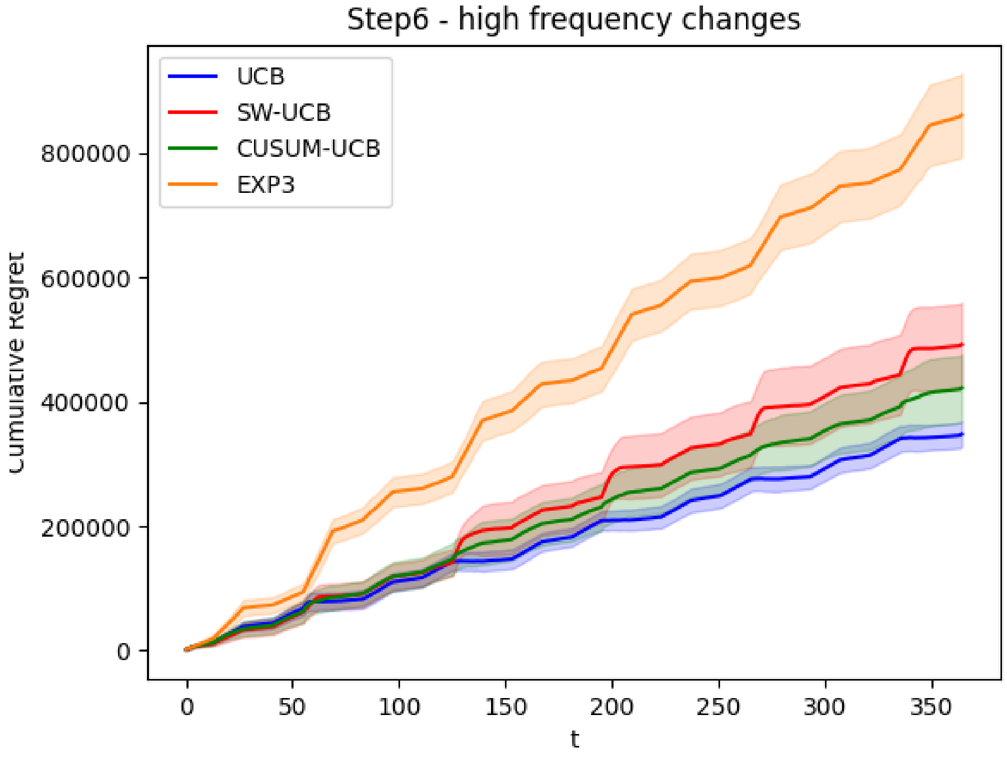


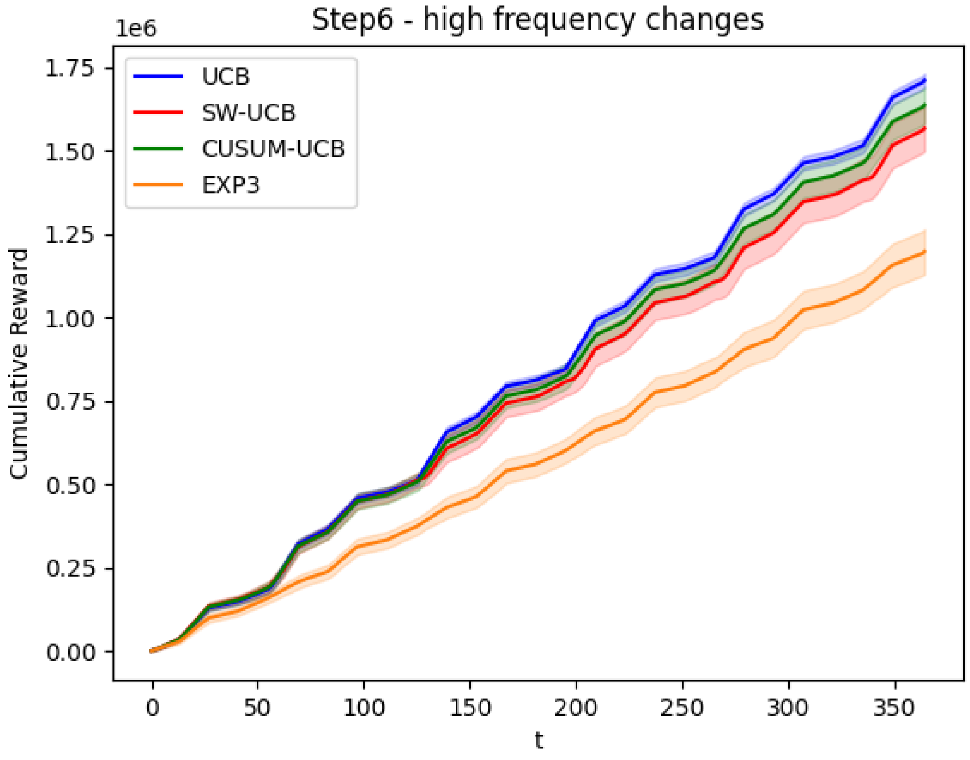
Step6 : Dealing with non-stationary environments with many abrupt changes - Results (High Frequency)



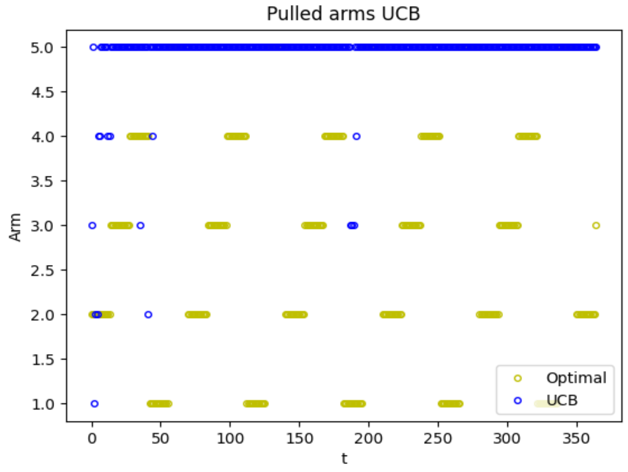


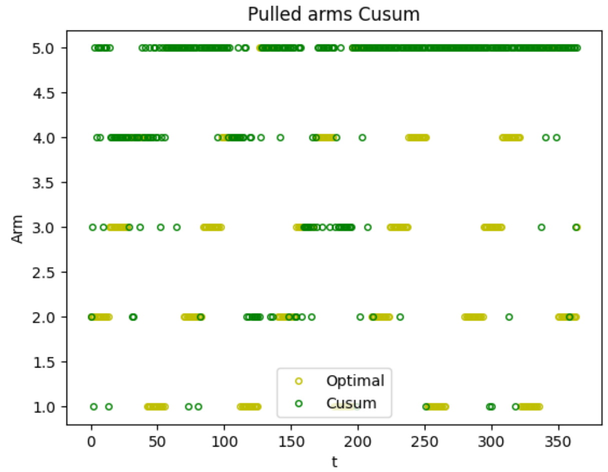
Step6 : Dealing with non-stationary environments with many abrupt changes - Results (High Frequency)

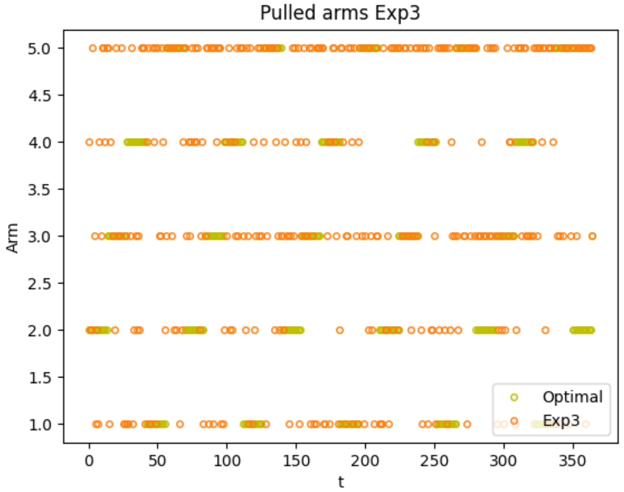


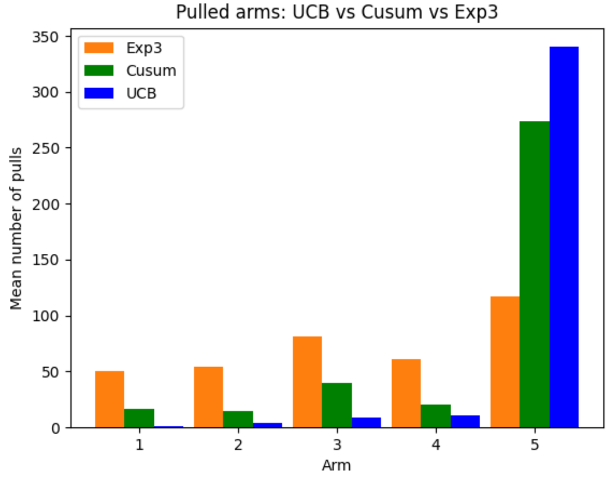


Step6 : Dealing with non-stationary environments with many abrupt changes - Results (High Frequency)









Thanks for your attention!

Lorenzo Ferretti	lorenzo2.ferretti@mail.polimi.it	10713719
Nicolò Fontana	nicolo.fontana@mail.polimi.it	10581197
Carlo Sgaravatti	carlo.sgaravatti@mail.polimi.it	10660072
Marco Venere	marco.venere@mail.polimi.it	10865088
Enrico Zardi	enrico.zardi@mail.polimi.it	10659549

Professors:
Castiglioni Matteo,
Gatti Nicola,
Bernasconi de Luca Martino

A.Y. 2022/2023