Pricing and Advertising

Project for the Data Intelligence Application Class
PoliMi 2019-2020

Mattia Bacarella - 928345
Francesco Corda - 920212
Davide Damato - 920616
Luciano Franchin - 921093
Luca Massini - 945027

August 25, 2020

Objective of the Project

The goal is to model a scenario in which a seller exploits advertising tools to attract more and more users to its website, thus increasing the number of possible buyers. The seller needs to learn simultaneously the conversion rate and the number of users the advertising tools can attract.

Project Structure

- Repository:
  `https://github.com/francescocorda/dia-project`
- Structure:

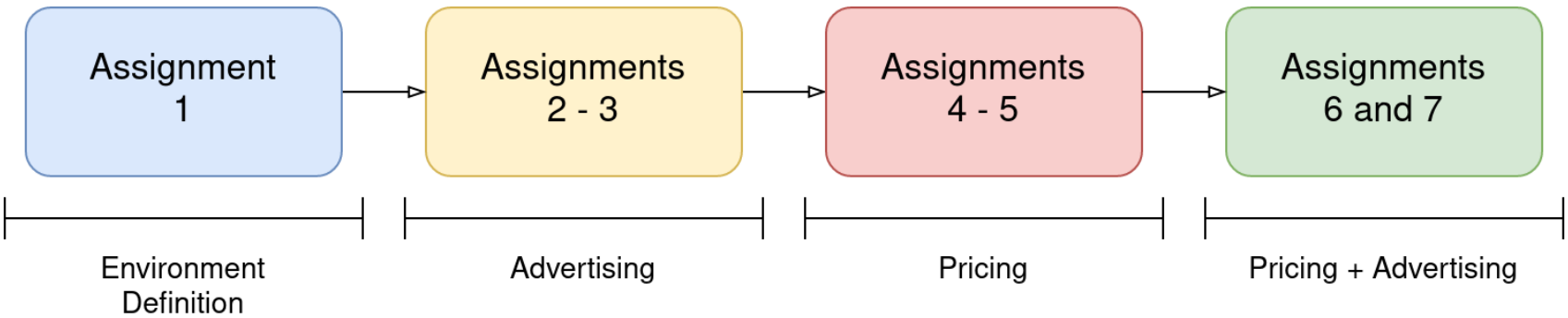
```
dia_project
├── assignment_2
├── assignment_3
├── assignment_4
├── assignment_5
├── assignment_6
├── assignment_7
├── data
└── utils
```

- The project is organized in modules, one for each assignment. To execute the code and reproduce our results, run the file assignment_x.py in each module.

Project Roadmap

After the definition of the working environment, the project can be divided in three main sections:

- a first part dedicated to advertising the product, with the goal of maximizing the traffic (expressed as number of clicks) directed to our web page
- a second part dedicated to the pricing, with the goal of finding the optimal selling point to obtain the highest revenue
- a third part in which the actions are combined, the algorithm learns simultaneously to advertise and sell



Assignment 1

Imagine one product to sell:

- *three classes of users, where, for every user, we can observe the values of two binary features (feel free to choose the features and their domains);*
- *the conversion rate curve of each class of users;*
- *three subcampaigns, each with a different ad, to advertise the product, and each targeting a different class of users;*
- *there are three abrupt phases;*
- *for every abrupt phase and for every subcampaign, the probability distribution over the daily number of clicks for every value of budget allocated to that subcampaign.*

Assignment 1 - The Product

We decided to analyze the pricing and the advertising of a **gym membership**, a product that presents the following properties:

- it's a well-known and established item, used by a vast and diversified population of users
- it's a highly customizable product, that allows for different advertising campaigns and price offers for different classes of users
- it's strongly affected by seasonality during the year

Assignment 1 - Users

The users are characterized with a set of 2 binary features, namely:

- Age = {Under 40, Over 40}
- Fitness Level = {Novice, Experienced}

We decided to focus on the following 3 classes generated by the features:

- (Under 40, Novice)
- (Under 40, Experienced)
- (Over 40, Experienced)

We excluded the class of (Over 40, Novice) users because of the online nature of our advertising campaign: an adult person which is not already interested in fitness is less likely to look on the internet for arguments related to our campaign.

Assignment 1 - (Under 40, Novice) class

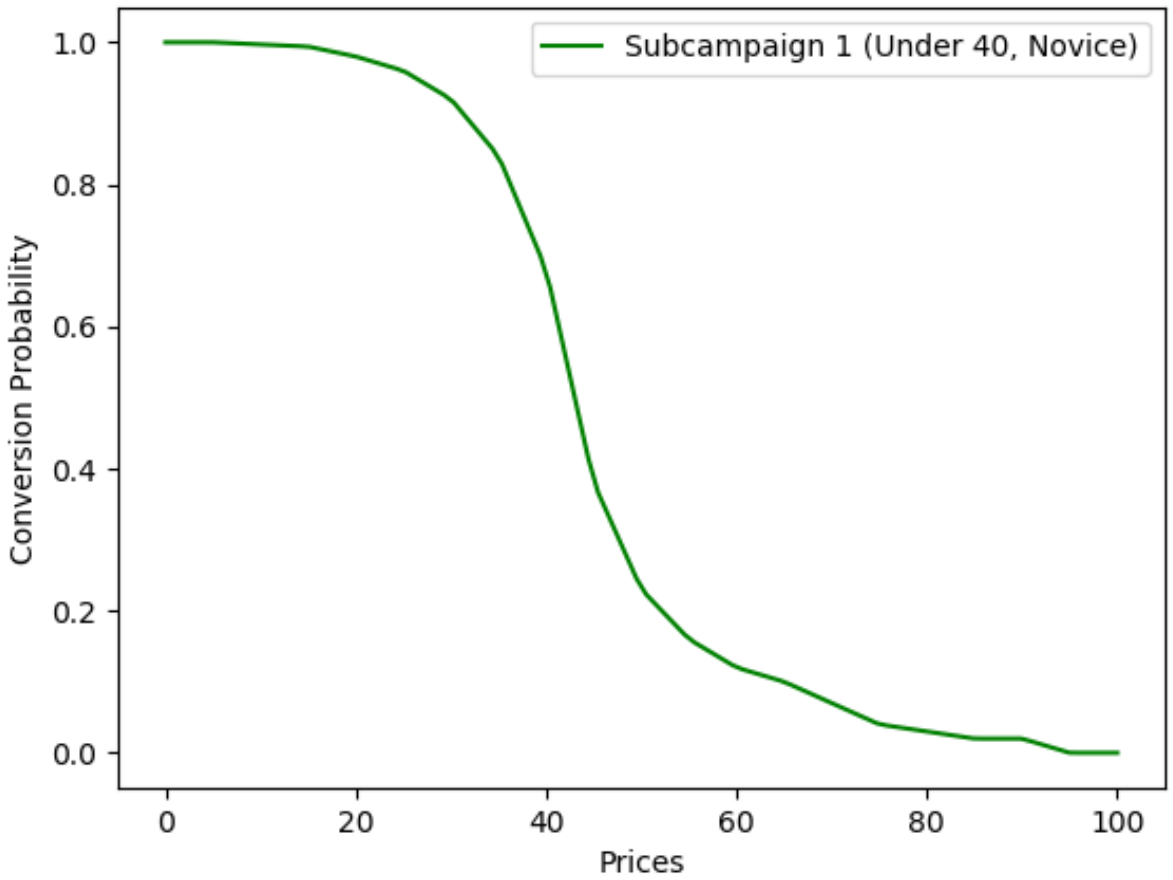


Figure:

Assignment 1 - (Under 40, Experienced) class

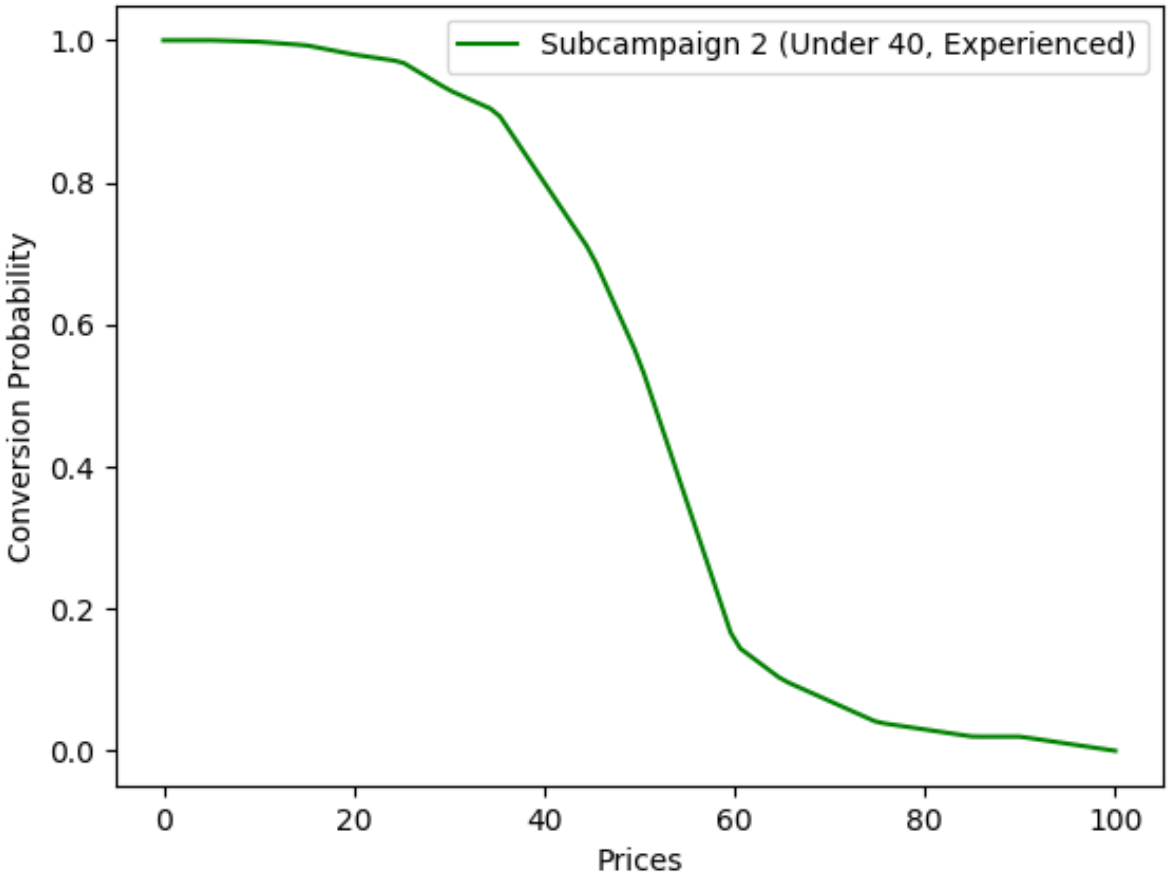


Figure:

Assignment 1 - (Over 40, Experienced) class

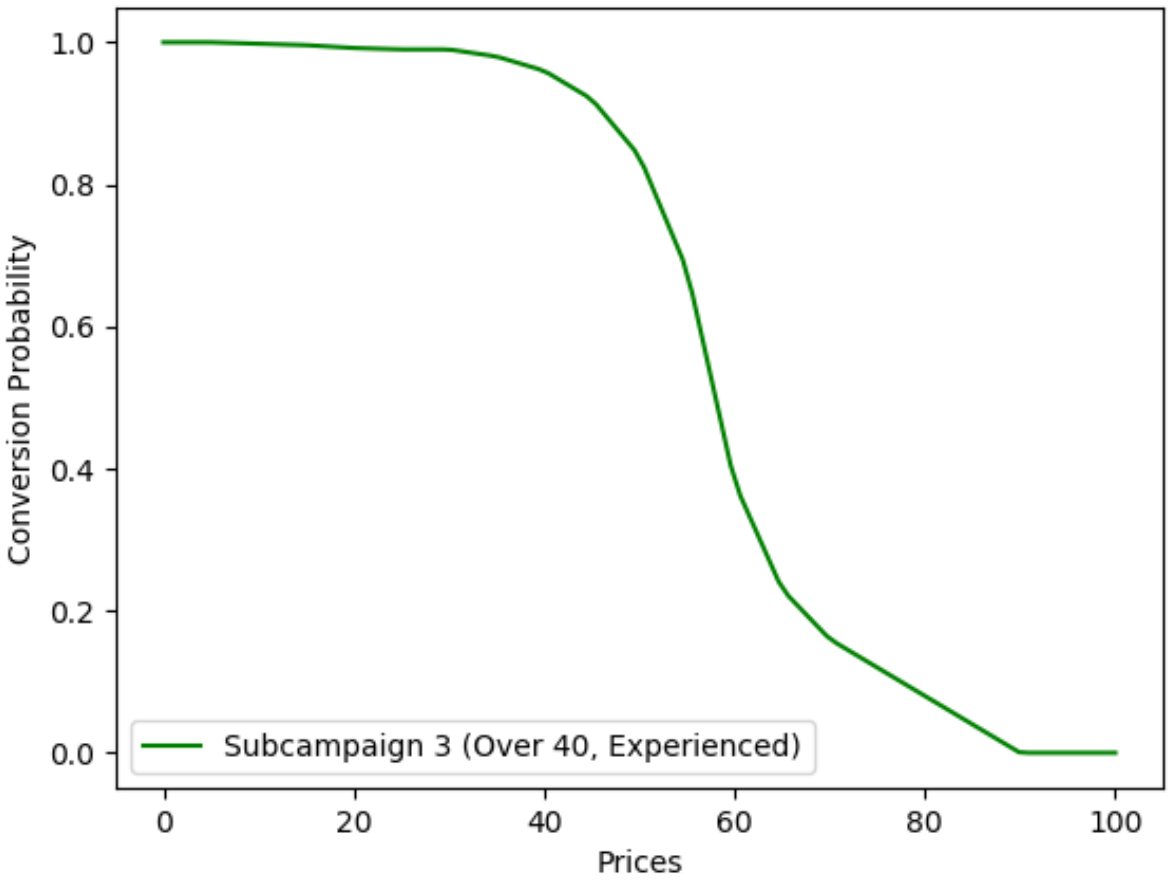


Figure:

Assignment 1 - Subcampaign 1

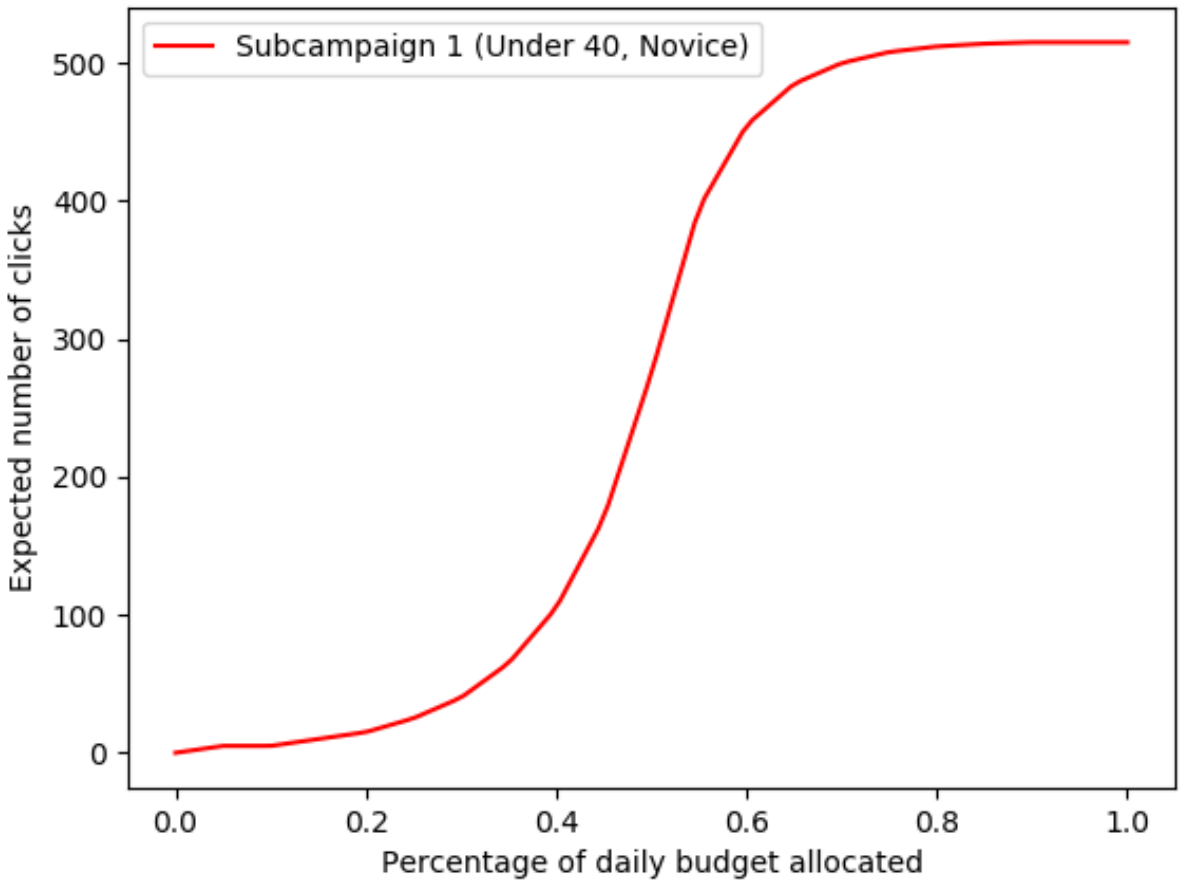


Figure:

Assignment 1 - Subcampaign 2

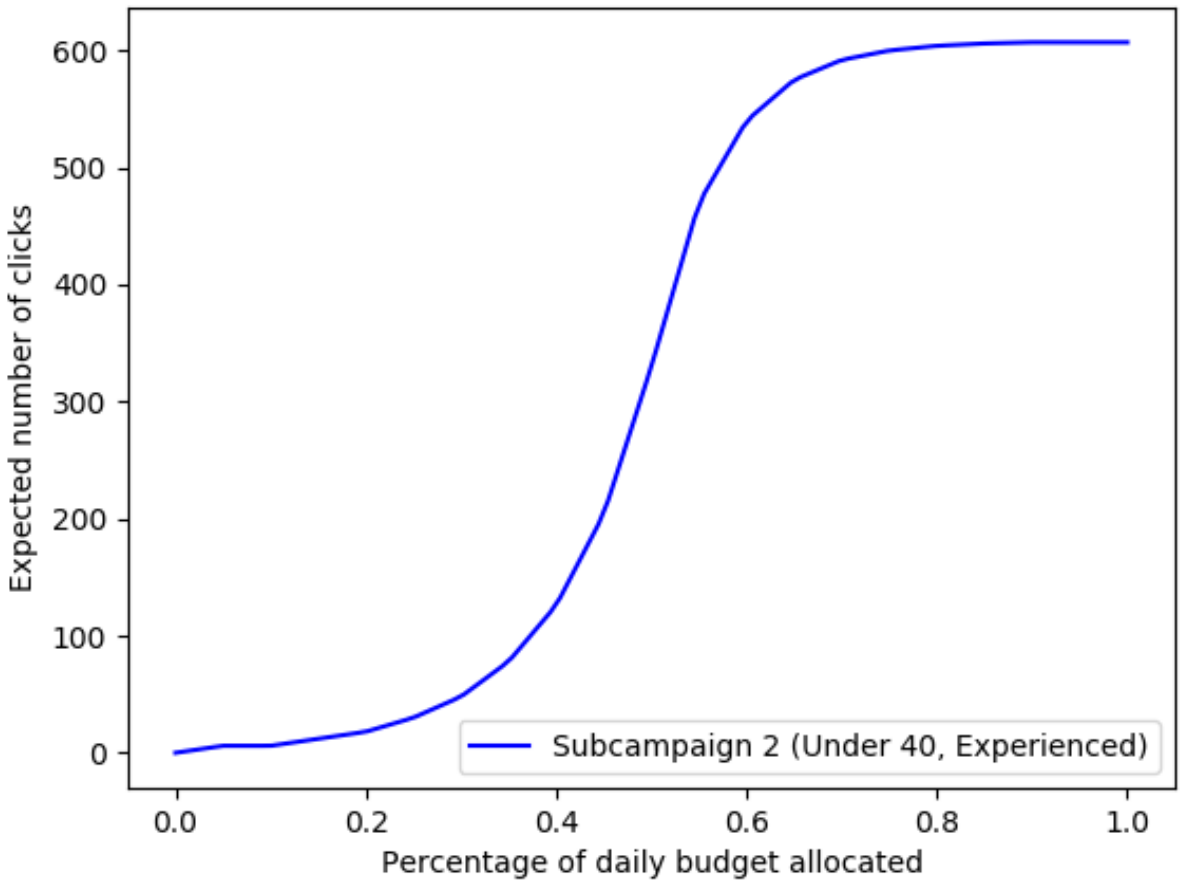


Figure:

Assignment 1 - Subcampaign 3

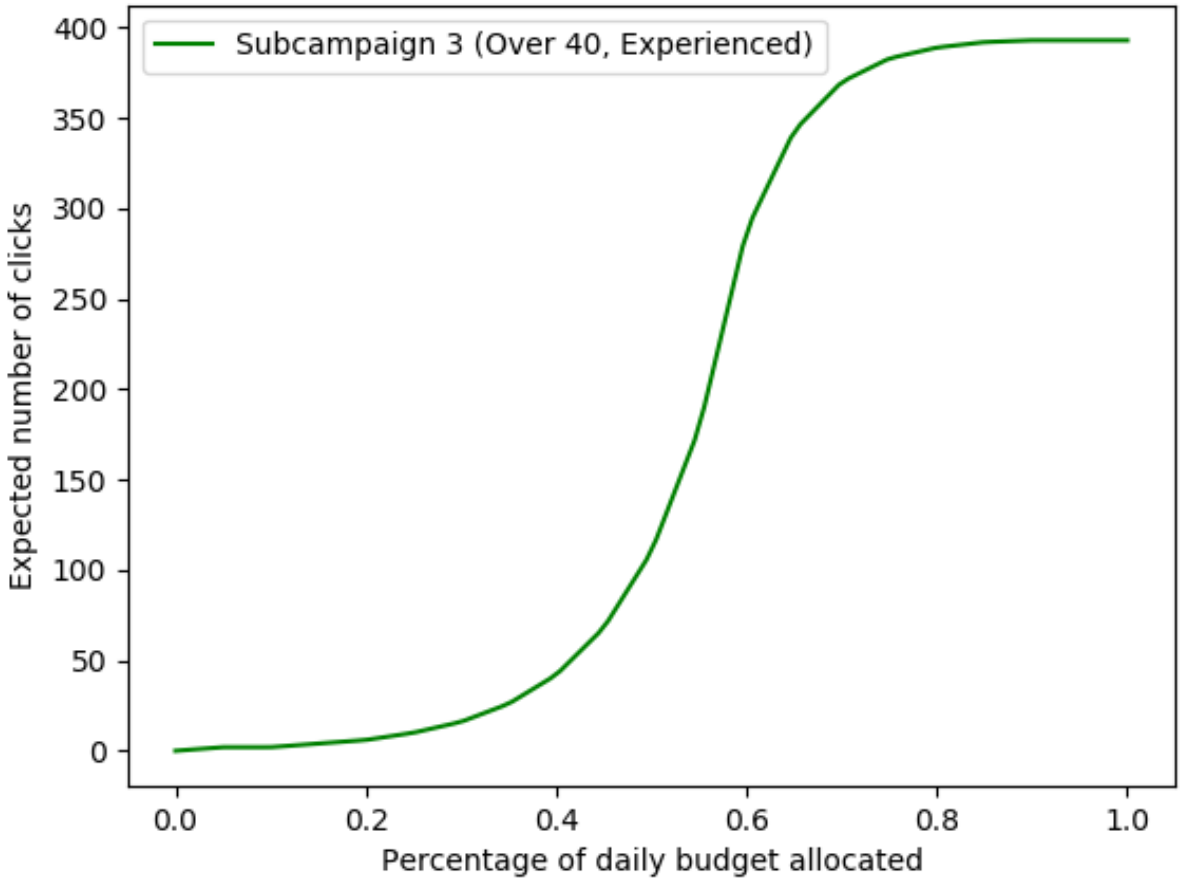


Figure:

Assignment 1 - Abrupt Phases

We identified 3 abrupt phases influenced by the seasonality that characterizes gym subscriptions during a year:

1. [January-April], the "New Year's Resolutions" phase
2. [May-August], the "Swimsuit Season" phase
3. [September-December], the "Back to Normal" phase

Each period has the same length of 4 months and is denoted by almost-stationary behaviors of the users for its duration.

Assignment 1 - Subcampaign 1, Abrupt Phases

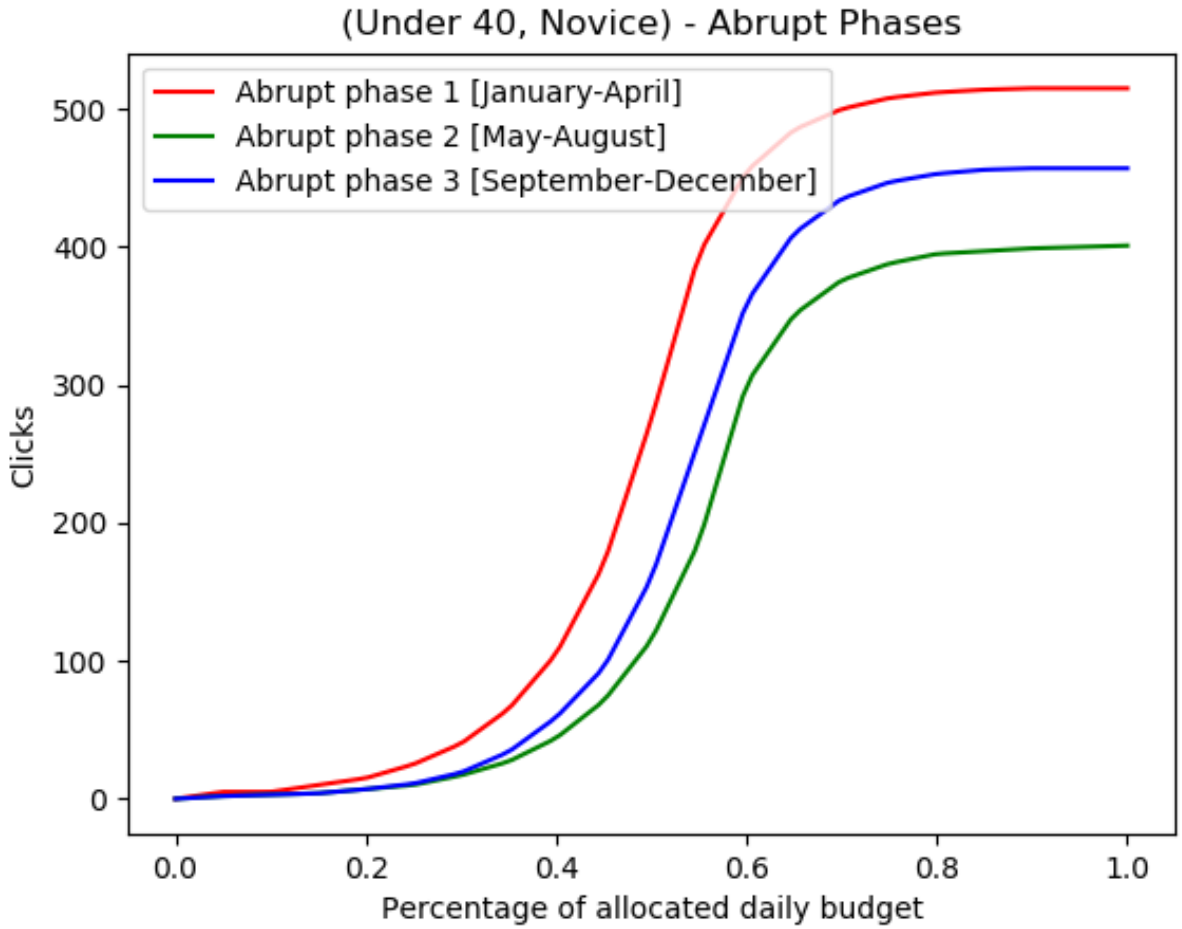


Figure:

Assignment 1 - Subcampaign 2, Abrupt Phases

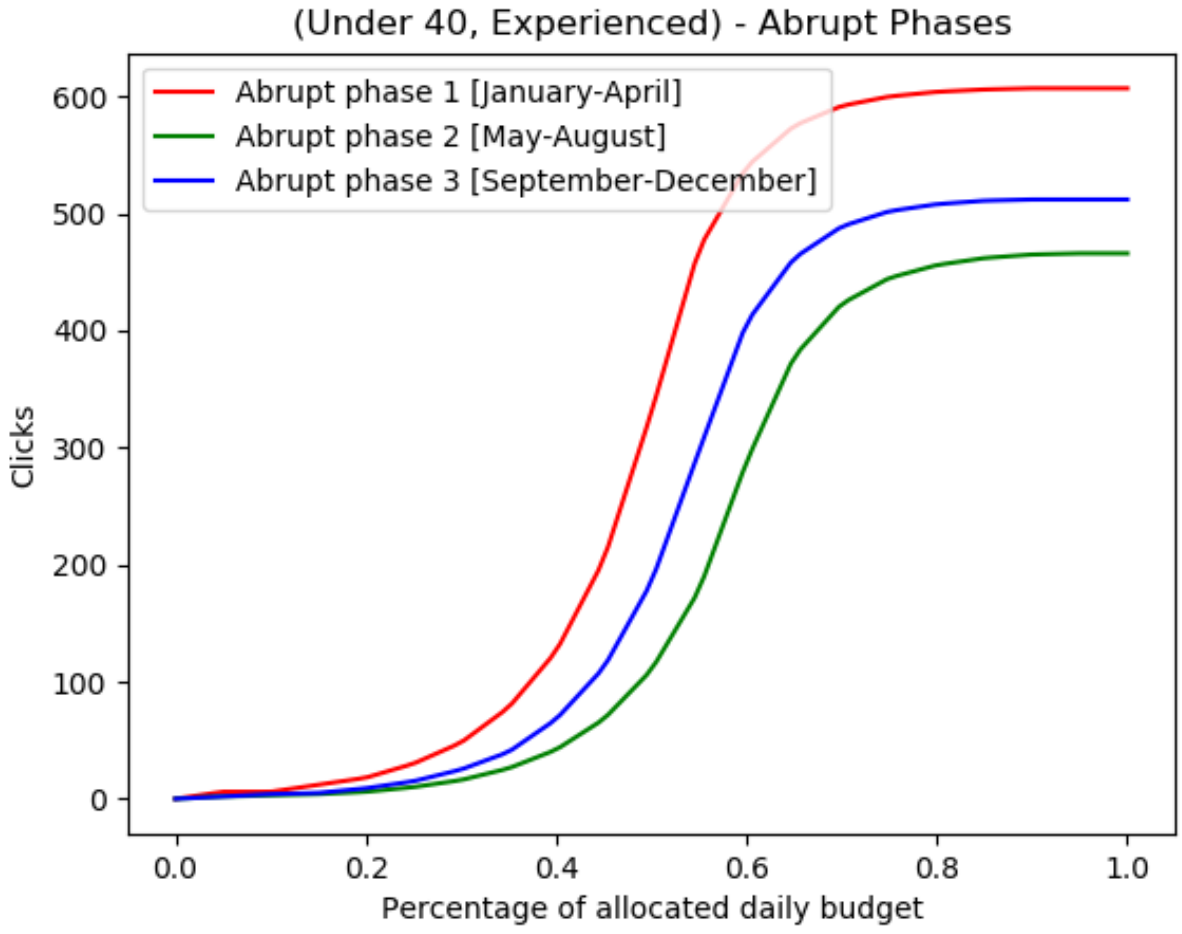


Figure:

Assignment 1 - Subcampaign 3, Abrupt Phases

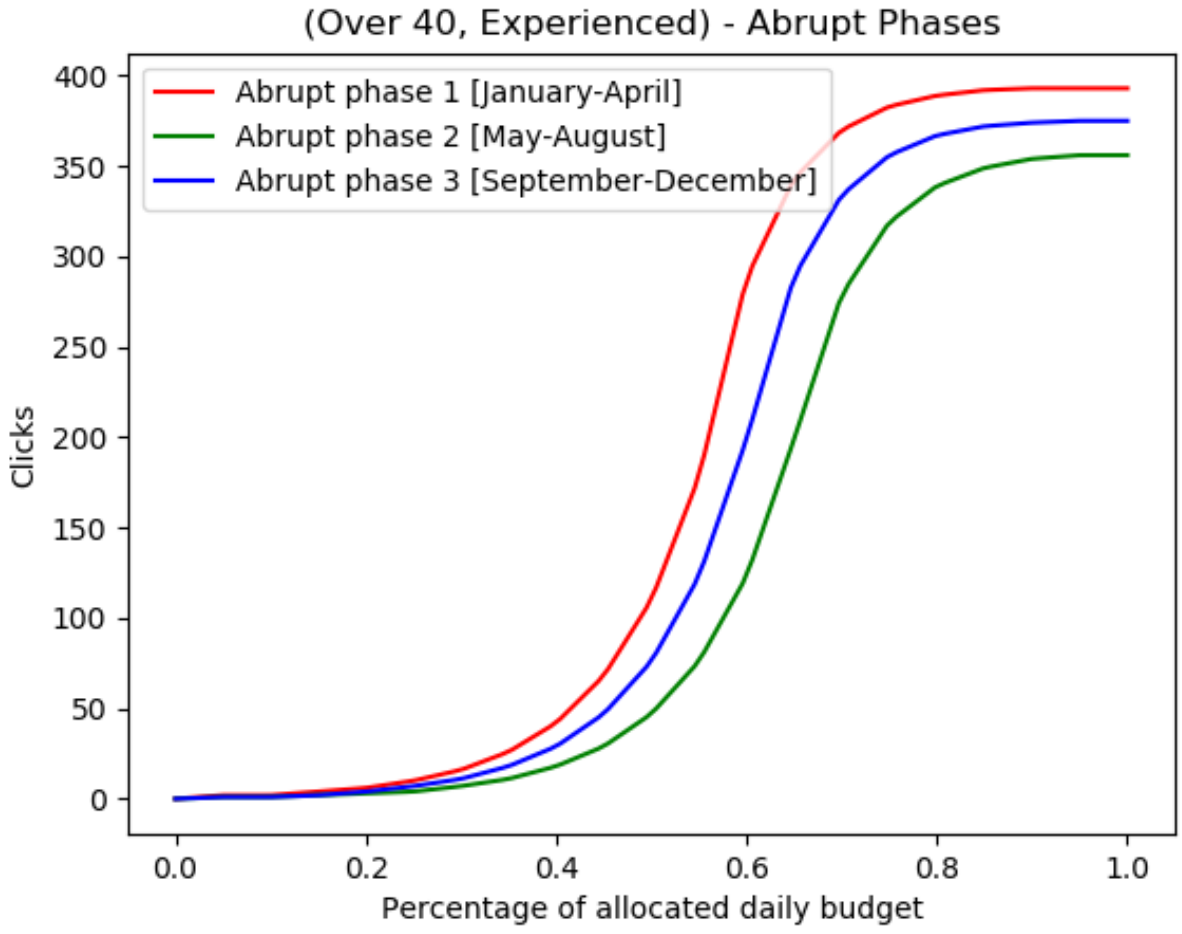


Figure:

Assignment 2

Design a combinatorial bandit algorithm to optimize the budget allocation over the three subcampaigns to maximize the total number of clicks when, for simplicity, there is only one phase. Plot the cumulative regret.

Assignment 2 - Pseudocode

```
initialize click envs
for exp in n_experiments:
  initialize gp-ts learners
  for t in time_horizon:
    draw samples for each subcampaign
    find super-arm by solving knapsack problem
    pull arms given by super-arm
    update gp-ts learners
calculate clairvoyant value
plot rewards and regrets
```

Assignment 2 - Cumulative Regret

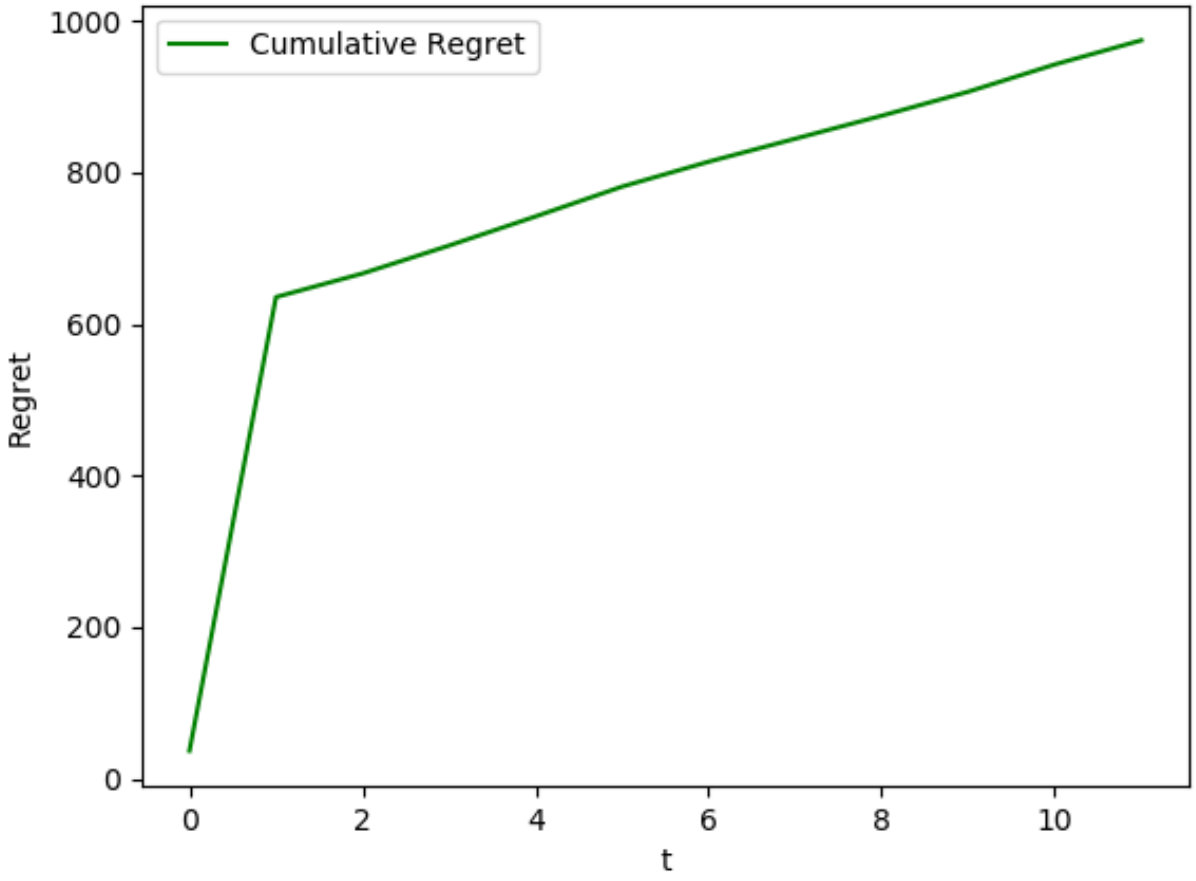


Figure:

Assignment 2 - Regret

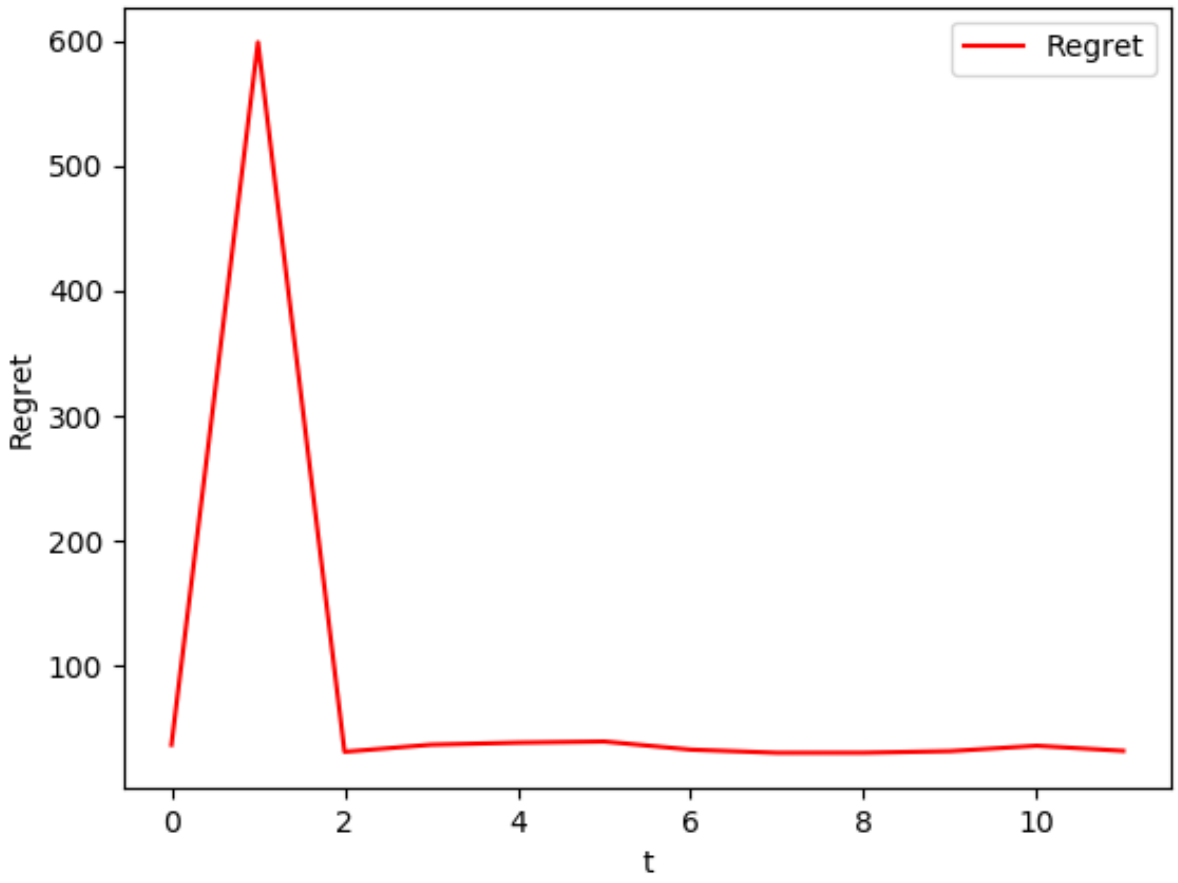


Figure:

Assignment 3

Design a sliding-window combinatorial bandit algorithm for the case, instead, in which there are the three phases aforementioned. Plot the cumulative regret and compare it with the cumulative regret that a non-sliding-window algorithm would obtain.

Assignment 3 - Pseudocode

```
initialize click envs
for exp in n_experiments:
  initialize gp-ts and gp-ts-sw learners
  for t in time_horizon:
    if abprupt_phase is changed:
      reset memory of the learners
    draw samples for each subcampaign
    find super-arm by solving knapsack problem
    pull arms given by super-arm
    update gp-ts and gp-ts-sw learners
calculate clairvoyant value
plot rewards and regrets for the 2 learners
```

Assignment 3 - Cumulative Regret

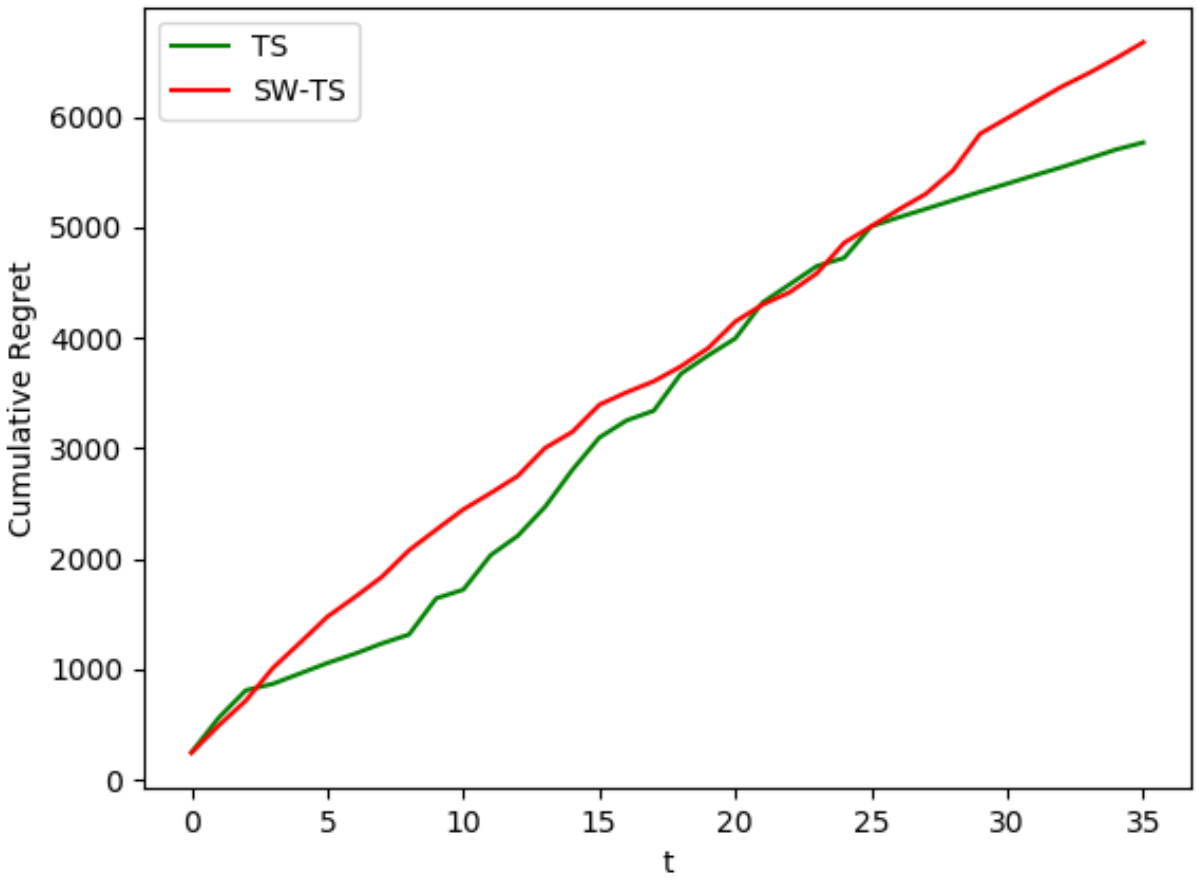


Figure:

Assignment 3 - Rewards

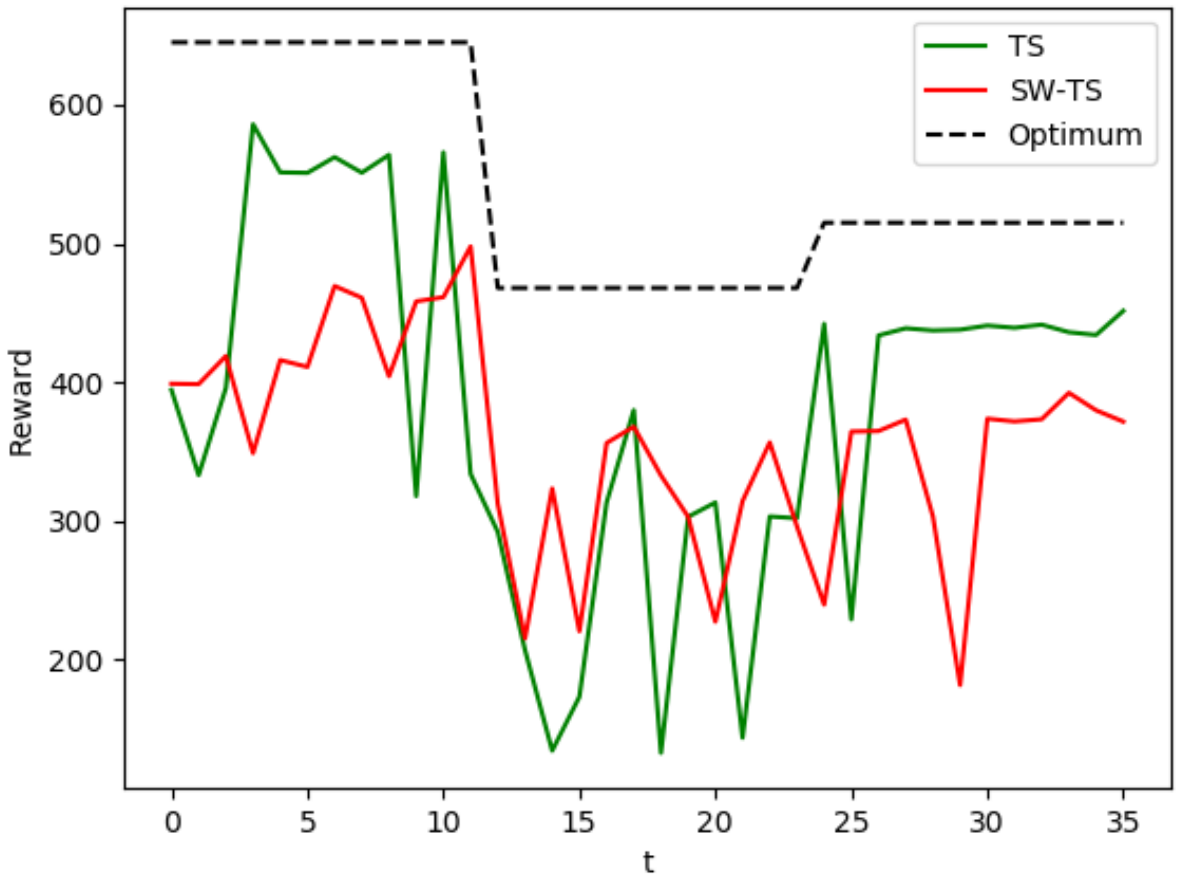


Figure:

Assignment 4

Design a learning algorithm for pricing when the users that will buy the product are those that have clicked on the ads. Assume that the allocation of the budget over the three subcampaigns is fixed and there is only one phase (make this assumption also in the next steps). Plot the cumulative regret.

Assignment 4 - Pseudocode

```
initialize pricing envs
for exp in n_experiments:
  initialize ts and greedy learners
  for t in time_horizon:
    draw samples for each subcampaign (ts learner)
    update ts learner
    draw samples for each subcampaign (greedy learner)
    update greedy learner
calculate clairvoyant value
plot rewards and regrets for the 2 learners
```

Assignment 4 - Regrets

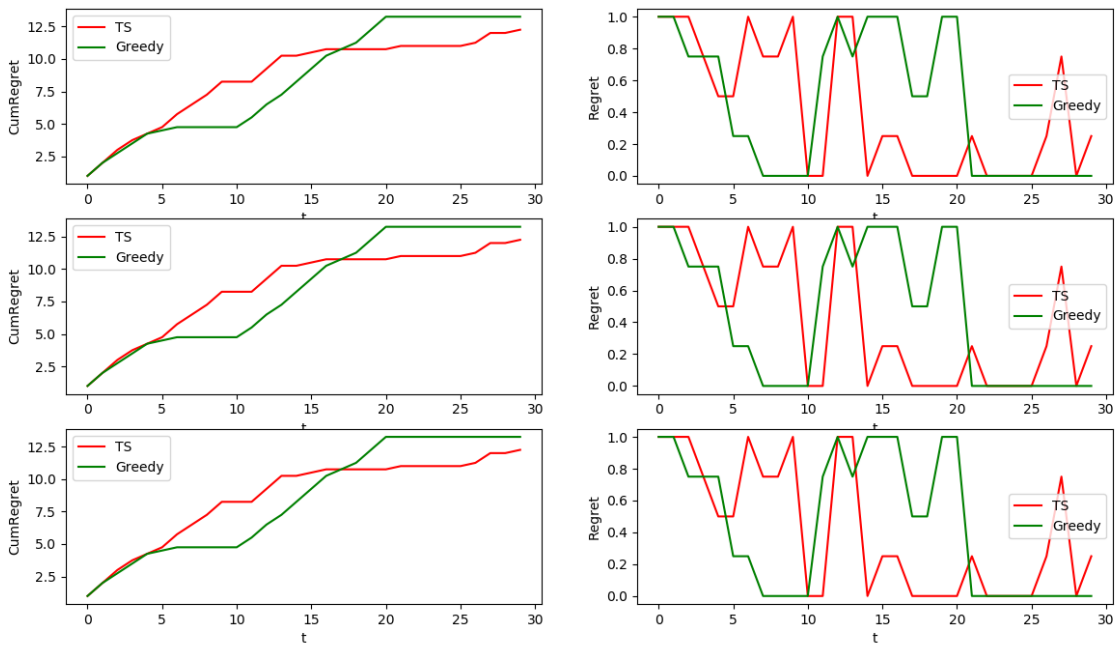


Figure:

Assignment 5

Design and run a context generation algorithm for the pricing when the budget allocated to each single subcampaign is fixed. At the end of every week, use the collected data to generate contexts and then use these contexts for the following week. Plot the cumulative regret as time increases. In the next steps, do not use the generated contexts, but use all the data together.

Assignment 5 - Pseudocode

```
initialize pricing envs
for exp in n_experiments:
  initialize context generator
  for t in time_horizon:
    if week is ended:
      generate new contexts (greedy approach)
    for context in n_contexts:
      run ts algorithm
calculate clairvoyant value
plot rewards and regrets
```

Assignment 5 - Cumulative Regret

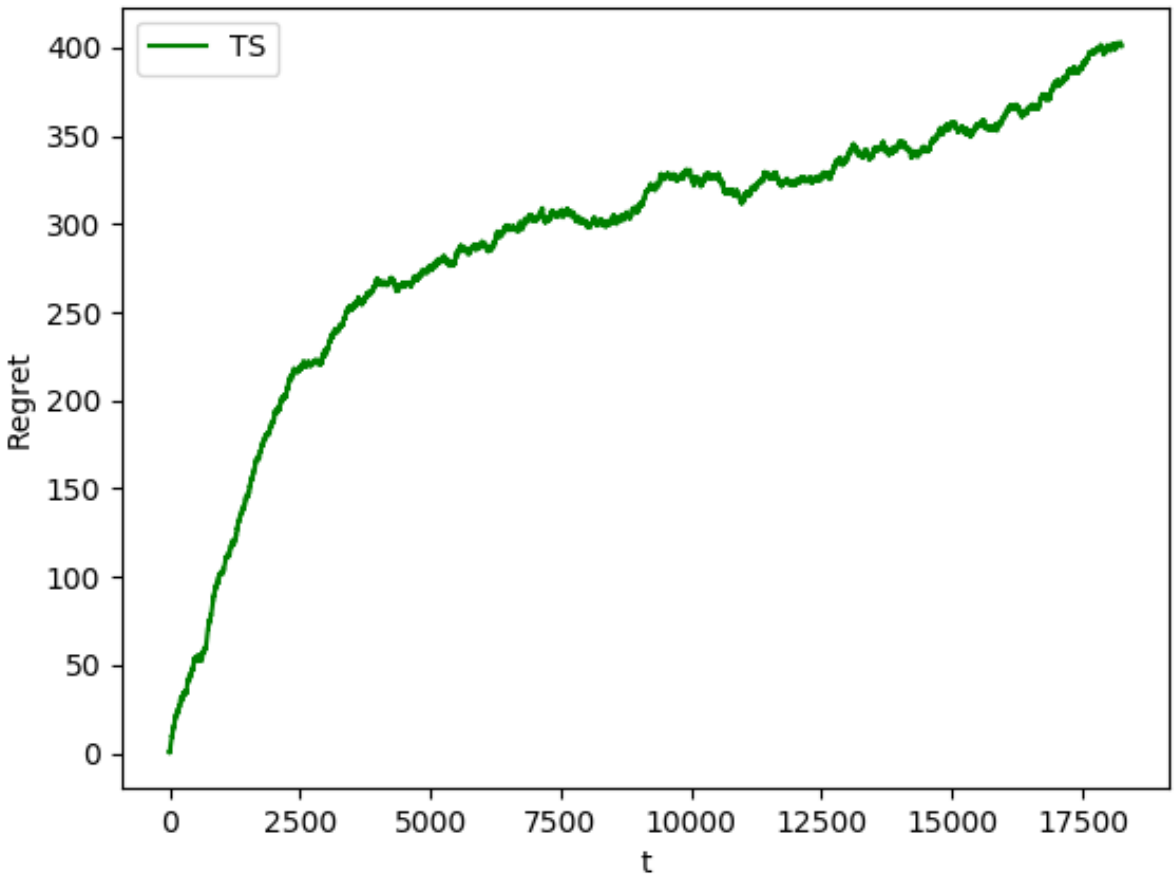


Figure:

Assignment 5 - Rewards

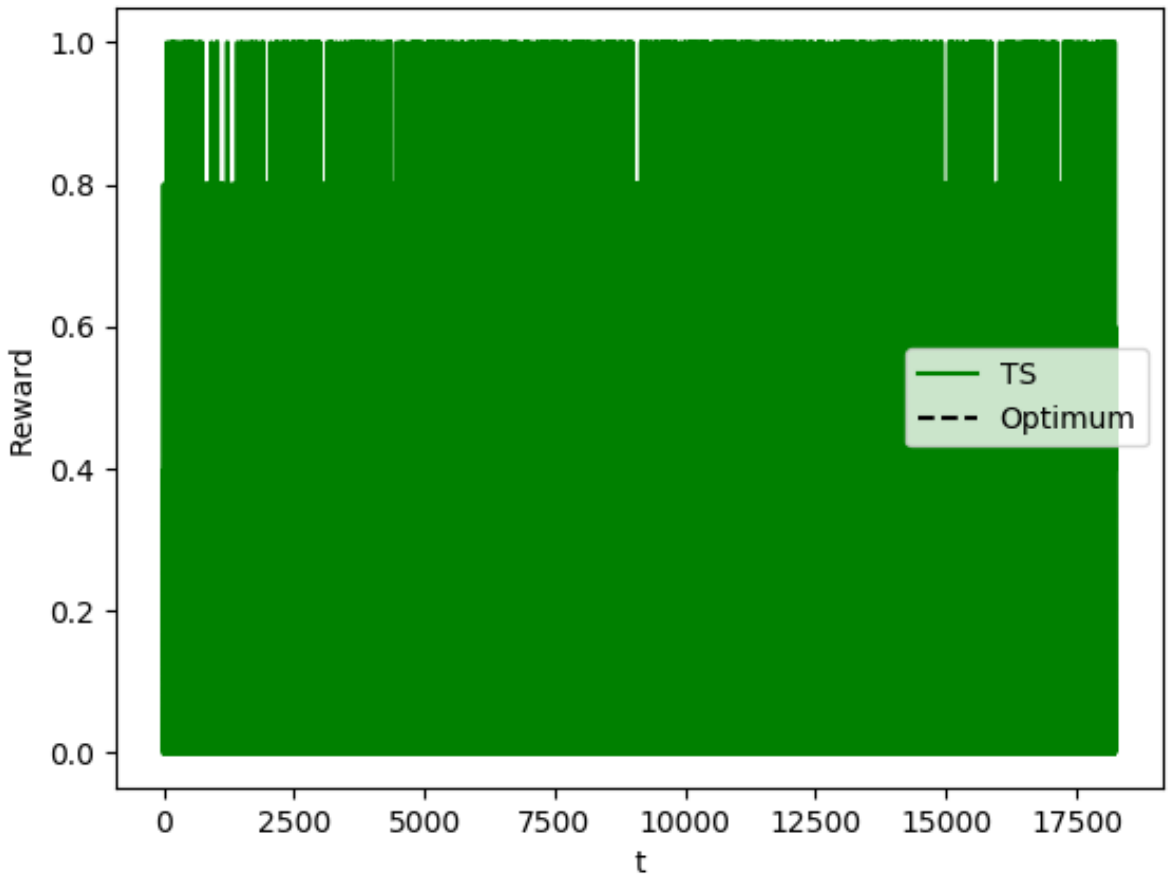


Figure:

Assignment 6

Design an optimization algorithm combining the allocation of budget and the pricing when the seller a priori knows that every subcampaign is associated with a different context and charges a different price for every context. Suggestion: the value per click to use in the knapsack-like problem depends on the pricing, that depends on the number of users of a specific class interested in buying the product. Notice that the two problems, namely, pricing and advertising, can be decomposed since each subcampaign targets a single class of users, thus allowing the computation of the value per click of a campaign only on the basis of the number of clicks generated by that subcampaign. Plot the cumulative regret when the algorithm learns both the conversion rate curves and the performance of the advertising subcampaigns.

Assignment 6 - Pseudocode

```
initialize pricing and click envs
for exp in n_experiments:
  initialize ts and gp-ts learners
  for t in time_horizon:
    for s in subcampaign:
      draw a (price, conversion_rate) sample with ts learner
      get return from the pricing env, given the price
      update ts learner
      draw click samples with gp-ts learner
      weight the clicks with price and conversion_rate previously selected by the ts learner
    find super-arm by solving knapsack problem
    get revenues = clicks * price * conversion_rate from the click envs by pulling the super-arm
    update gp-ts learners
calculate clairvoyant value
plot rewards and regrets
```

Assignment 6 - Cumulative Regret

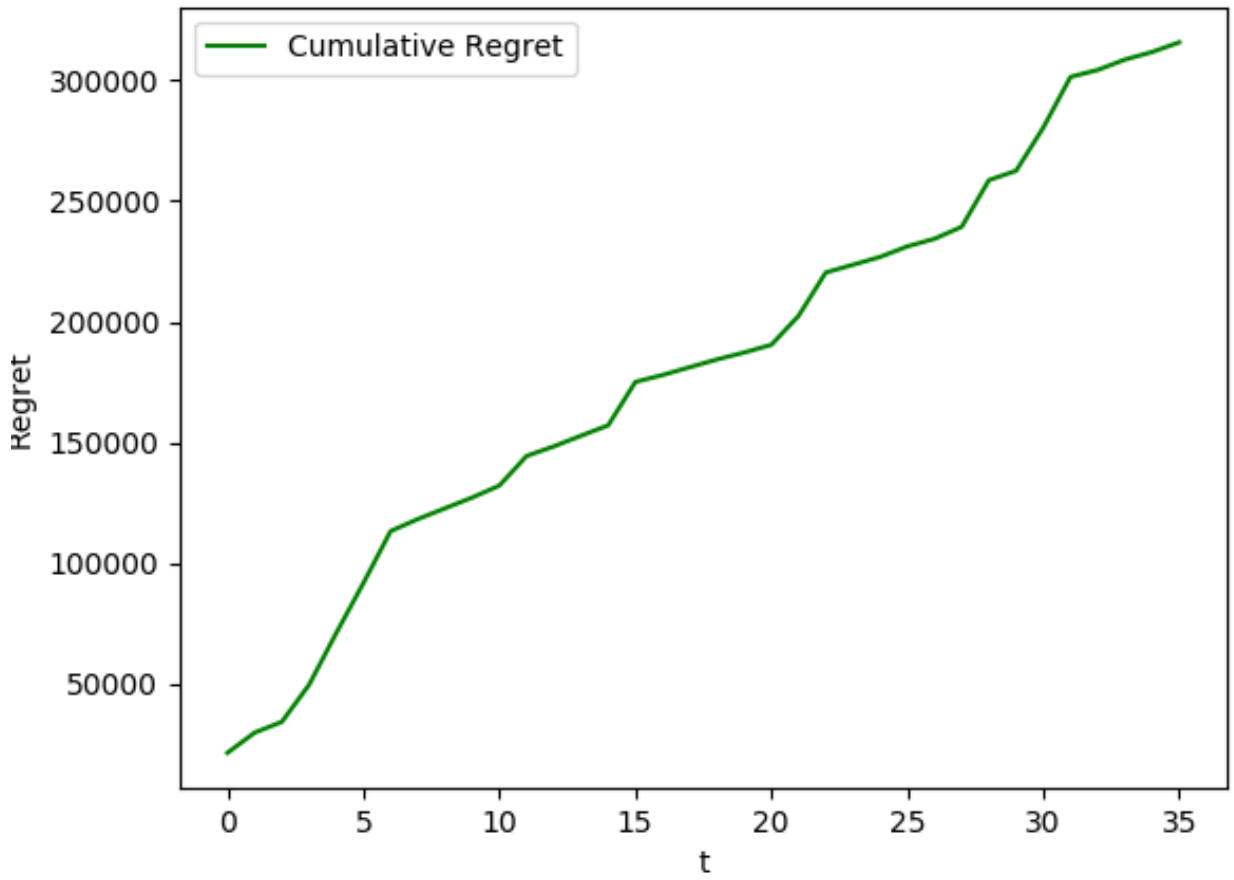


Figure:

Assignment 6 - Regret

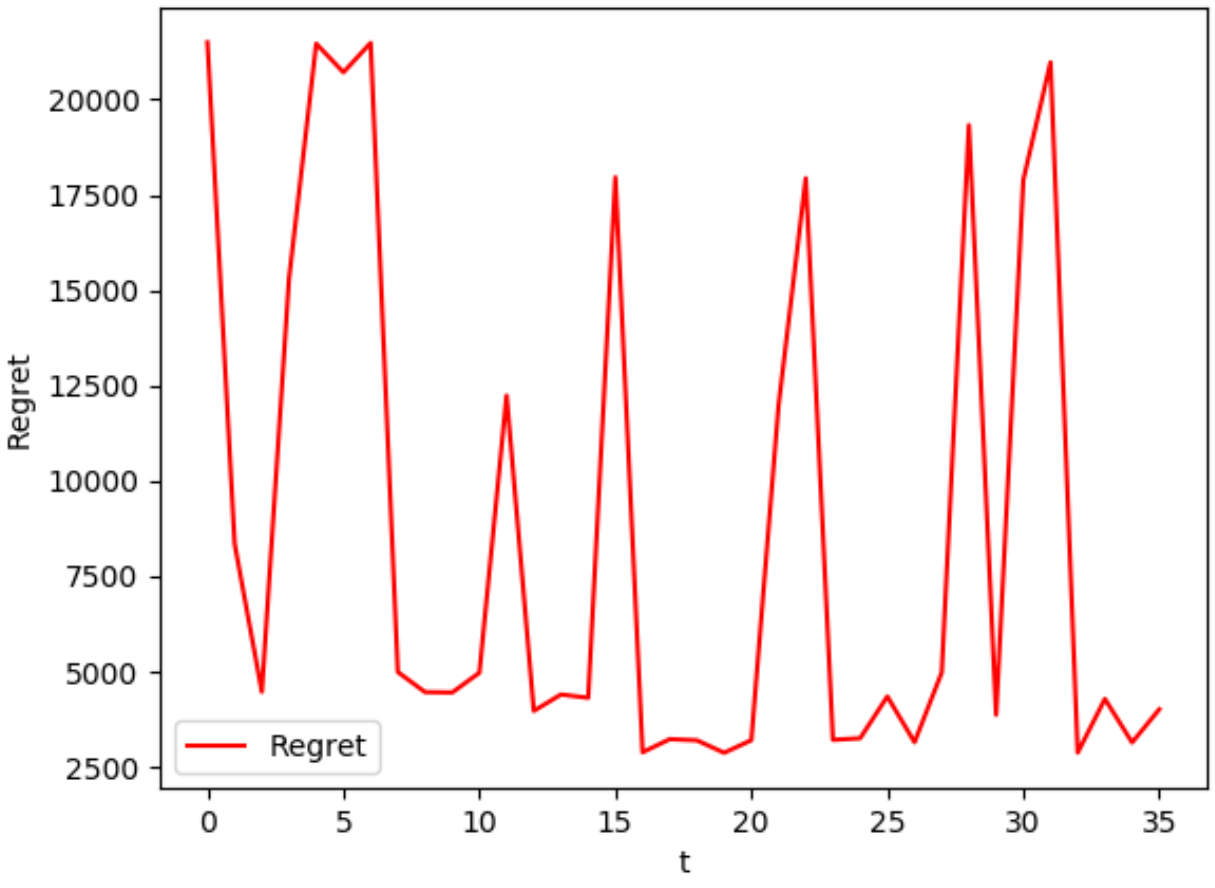


Figure:

Assignment 7

Do the same of Step 6 under the constraint that the seller charges a unique price to all the classes of users. Suggestion: for every possible price, fix this price and repeat the algorithm used in Step 6. Plot the cumulative regret when the algorithm learns both the conversion rate curves and the performance of the advertising subcampaigns.

Assignment 7 - Pseudocode

```
initialize pricing and click envs
for exp in n_experiments:
  for price in n_prices:
    initialize ts and gp-ts learners
    for t in time_horizon:
      for s in subcampaign:
        select current price and conversion_rate
        get return from the pricing env, given the
       price
        update ts learner
        draw click samples with gp-ts learner
        weight the clicks with price and
         conversion_rate previously selected
      find super-arm by solving knapsack problem
      collect revenues = = clicks * price *
     conversion_rate from the click envs by pulling
     the super-arm
      update gp-ts learners
calculate clairvoyant value
plot rewards and regrets
```

Assignment 7 - Regrets

TODO

Assignment 7 - Rewards

TODO