Introduction
The code is for our submission 'Rate-Optimal Online Learning for Dynamic Assortment Selection with Positioning'.

File structure:

├─── README.md # The file you are reading now describes the file structure and how to use the code.
├─── requirement.txt # The file contains the environment of python for our codes.

├─── Methods - Folder
(1) Pre_Functions.py - This Python file prepares and contains functions used to define the main policies.
(2) TLR_UCB.py - This Python file contains codes for the proposed TLR-UCB policy.
(3) EI_TLR.py - This Python file contains codes for the proposed EI-TLR policy.
(4) EI_TLR_Explore.py - This Python file contains codes for the exploration and joint estimation part of the proposed EI-TLR policy.
(5) LR_UCB.py - This Python file contains codes for the LR-UCB policy.
(6) A_UCB_V.py - This Python file contains codes for the A-UCB-V policy.
(7) A_UCB.py - This Python file contains codes for the A-UCB policy.
(8) S_ETC.py - This Python file contains codes for the S-ETC policy.
(9) Sub_Gap.py - This Python file contains the function that computes suboptimality gaps for DAP instances.

├─── Simulation - Folder
General structure: This folder contains Python files that produce results through simulations; these results are saved in the "Results" subfolder; these results are then used by "Plot_..." Python files to produce Figures in the main paper and Appendix. In the following, we present the details based on the 'code–results–plot' relationships.

(1) "Ex1_Known.py" implements TLR-UCB, LR-UCB and A-UCB-V policies on Example 1. It produces the results "TLR-UCB_Ex1_reg_mean_1.npy", "LR-UCB_Ex1_reg_mean_1.npy", "A-UCB-V_Ex1_reg_mean_1.npy", saved in the "Results" subfolder, and then used by "Plot_Ex1_Known.py" to produce the left panel of Figure 1 in the main paper.

(2) "Ex2_Known.py" implements TLR-UCB, LR-UCB and A-UCB-V policies on Example 2. It produces the results "TLR-UCB_Ex2_reg_mean_1.npy", "LR-UCB_Ex2_reg_mean_1.npy", "A-UCB-V_Ex2_reg_mean_1.npy", saved in the "Results" subfolder, and then used by "Plot_Ex2_Known.py" to produce the middle panel of Figure 1 in the main paper.

(3) "Ex3_Known.py" implements TLR-UCB, LR-UCB and A-UCB-V policies on Example 3. It produces the results "TLR-UCB_Ex3_reg_mean_1.npy", "LR-UCB_Ex3_reg_mean_1.npy", "A-UCB-V_Ex3_reg_mean_1.npy", saved in the "Results" subfolder, and then used by "Plot_Ex3_Known.py" to produce the right panel of Figure 1 in the main paper.

(4) "Ex1_Unknown.py" implements EI-TLR, A-UCB and S-ETC policies on Example 1. It produces the results "EI-TLR_Ex1_reg_mean_1.npy", "A-UCB_Ex1_reg_mean_1.npy", "S-

ETC_Ex1_reg_mean_1.npy", saved in the "Results" subfolder, and then used by "Plot_Ex1_Unknown.py" to produce the left panel of Figure 2 in the main paper.

(5) "Ex2_Unknown.py" implements EI-TLR, A-UCB and S-ETC policies on Example 2. It produces the results "EI-TLR_Ex2_reg_mean_1.npy", "A-UCB_Ex2_reg_mean_1.npy", "S-ETC_Ex2_reg_mean_1.npy", saved in the "Results" subfolder, and then used by "Plot_Ex2_Unknown.py" to produce the middle panel of Figure 2 in the main paper.

(6) "Ex3_Unknown.py" implements EI-TLR, A-UCB and S-ETC policies on Example 3. It produces the results "EI-TLR_Ex3_reg_mean_1.npy", "A-UCB_Ex3_reg_mean_1.npy", "S-ETC_Ex3_reg_mean_1.npy", saved in the "Results" subfolder, and then used by "Plot_Ex3_Unknown.py" to produce the right panel of Figure 2 in the main paper.

(7) "Ex1_Sensitivity.py" implements TLR-UCB on Example 1, with varying values of the parameter \nu_{0}. It also implements LR-UCB and A-UCB-V on Example 1. It produces the results "TLR-UCB_Ex1_reg_mean_Sen_1.npy", "TLR-UCB_Ex1_reg_mean_Sen_2.npy", "TLR-UCB_Ex1_reg_mean_Sen_3.npy", "LR-UCB_Ex1_reg_mean_Sen_1.npy", "A-UCB-V_Ex1_reg_mean_Sen_1.npy", saved in the "Results" subfolder, and then used by "Plot_Ex1_Sensitivity.py" to produce Figure 1 in the Appendix.

(8) "Ex4_Gap" implements TLR-UCB on varying settings in Example 4. It produces the results "TLR-UCB_Ex4_reg_mean_1.npy", "TLR-UCB_Ex4_reg_mean_2.npy", "TLR-UCB_Ex4_reg_mean_3.npy", "TLR-UCB_Ex4_reg_mean_4.npy", "TLR-UCB_Ex4_reg_mean_5.npy", saved in the "Results" subfolder, and then used by "Plot_Ex4_Gap.py" to produce Figure 5 in the Appendix.

(9) "Ex1_ErrorRate.py" implements the exploration and joint estimation part of EI-TLR on Example 1. It produces the results "Ex1_range_epoch_1.npy", "Ex1_pre_est_err_mean_1.npy", "Ex1_pos_est_err_mean_1.npy", saved in the "Results/ErrorRate" subfolder, and then used by "Plot_Ex1_ErrorRate.py" to produce the left and middle panels of Figure 3 in the main paper.

(10) "Ex2_ErrorRate.py" implements the exploration and joint estimation part of EI-TLR on Example 2. It produces the results "Ex2_range_epoch_1.npy", "Ex2_pre_est_err_mean_1.npy", "Ex2_pos_est_err_mean_1.npy", saved in the "Results/ErrorRate" subfolder, and then used by "Plot_Ex2_ErrorRate.py" to produce the left and middle panels of Figure 2 in the Appendix.

(11) "Ex3_ErrorRate.py" implements the exploration and joint estimation part of EI-TLR on Example 3. It produces the results "Ex3_range_epoch_1.npy", "Ex3_pre_est_err_mean_1.npy", "Ex3_pos_est_err_mean_1.npy", saved in the "Results/ErrorRate" subfolder, and then used by "Plot_Ex3_ErrorRate.py" to produce the left and middle panels of Figure 3 in the Appendix.

(12) "Ex1_RegretRate.py" implements EI-TLR on Example 1 with varying horizon lengths. It produces the results "EI-TLR_Ex1_reg_exp2_mean_1.npy" -- "EI-TLR_Ex1_reg_exp2_mean_20.npy", saved in the "Results/Ex1_RegretRate" subfolder, and then used by "Plot_Ex1_RegretRate.py" to produce the right panel of Figure 3 in the main paper.

(13) "Ex2_RegretRate.py" implements EI-TLR on Example 2 with varying horizon lengths. It produces the results "EI-TLR_Ex2_reg_exp2_mean_1.npy" -- "EI-TLR_Ex2_reg_exp2_mean_20.npy", saved in the "Results/Ex2_RegretRate" subfolder, and then used by "Plot_Ex2_RegretRate.py" to produce the right panel of Figure 2 in the Appendix.

(14) "Ex3_RegretRate.py" implements EI-TLR on Example 3 with varying horizon lengths. It produces the results "EI-TLR_Ex3_reg_exp2_mean_1.npy" -- "EI-TLR_Ex3_reg_exp2_mean_20.npy", saved in the "Results/Ex3_RegretRate" subfolder, and then used by "Plot_Ex3_RegretRate.py" to produce the right panel of Figure 3 in the Appendix.

(15) "Ex123_Subgap.py" calculates and presents sub-optimality gaps for Examples 1 -- 3, thus reproducing results in Table 1 in the Appendix.

(16) "Ex4_Subgap.py" calculates and presents sub-optimality gaps for varying settings in Examples 4, thus reproducing results in Table 2 in the Appendix.

*** Note that in (12) -- (14), to save time, "Ex1_RegretRate.py" (and "Ex2_RegretRate.py", "Ex3_RegretRate.py") recommends to implement EI-TLR with different horizon lengths separately, i.e., in a parallel fashion. Thus the parameter i needs to be manually varied from 0 -- 19.

├── Real_Application - Folder

(1) Real_Data - Subfolder - Contain the raw real data and its description.

(2) Pre_Process - Subfolder

(1) "RealData_Pre.py" preprocess the raw real data and obtain the covariate "Car_x.npy" and response "Car_y.npy".

(2) "L2_Logistic.py" estimates the ground truth preferences and obtain "Car_Pre.npy", which serves as the preference "v" vector in the applications of various policies on the real car data.

(3) "Car_TLR_UCB.py" implements the proposed TLR-UCB policy on the real car data under known position effects. It produces the results "TLR-UCB_reg_mean_1.npy" -- "TLR-UCB_reg_mean_50.npy", saved in the "Results/TLR_UCB" subfolder.

(4) "Car_LR_UCB.py" implements the LR-UCB policy on the real car data under known position effects. It produces the results "LR-UCB_reg_mean_1.npy" -- "LR-UCB_reg_mean_50.npy", saved in the "Results/LR_UCB" subfolder.

(5) "Car_A_UCB_V.py" implements the A-UCB-V policy on the real car data under known position effects. It produces the results "A-UCB-V_reg_mean_1.npy" -- "A-UCB-V_reg_mean_50.npy", saved in the "Results/A_UCB_V" subfolder.

(6) "Car_EI_TLR.py" implements the proposed EI-TLR policy on the real car data under unknown position effects. It produces the results "EI_TLR_reg_mean_1.npy" -- "EI_TLR_reg_mean_50.npy", saved in the "Results/EI_TLR" subfolder.

(7) "Car_A_UCB.py" implements the A-UCB policy on the real car data under unknown position effects. It produces the results "A_UCB_V_reg_mean_1.npy" -- "A_UCB_V_reg_mean_50.npy", saved in the "Results/A_UCB" subfolder.

(8) "Plot_Car_Known.py" uses the results in the subfolders "Results/TLR_UCB", "Results/LR_UCB", "Results/A_UCB_V" to produce the left panel of Figure 4 in the main paper.

(9) "Plot_Car_Unknown.py" uses the results in the subfolders "Results/EI_TLR", "Results/A_UCB" to produce the right panel of Figure 4 in the main paper.

*** Note that in (3) -- (7), due to the long horizon length and large N and K, to save time, it is recommended to implement each policy in a parallel fashion, i.e., one replication each time. Specifically, the parameter i, which corresponds to the i-th replication and the used random seed, needs to be manually varied from 0 -- 49 to conduct 50 replications.

*** Note that unlike in the Simulation folder, the results are not included due to their large size. Therefore, the "Plot..." Python files cannot be directly applied to reproduce the figures.

Install Python packages

We use Python 3 to run experiments and plot figures. The Python packages required are listed in `requirements.txt`, and you can install them using the bash command `pip3 install -r requirements.txt`. For more details, please refer to the [official Python tutorial](https://packaging.python.org/en/latest/tutorials/installing-packages/) about installing packages.

The code is tested with Python 3.9.6.

All path-related codes (e.g., sys.path.append(), os.chdir()) in these Python files need to be adapted to the local environment.

Guidelines for Reproducing figures and tables in the main paper and Appendix:

Main paper:

(1) Left panel of Figure 1: Run Ex1_Known, and then Plot_Ex1_Known

(2) Middle panel of Figure 1: Run Ex2_Known, and then Plot_Ex2_Known

(3) Right panel of Figure 1: Run Ex3_Known, and then Plot_Ex3_Known

(4) Left panel of Figure 2: Run Ex1_Unknown, and then Plot_Ex1_Unknown

(5) Middle panel of Figure 2: Run Ex2_Unknown, and then Plot_Ex2_Unknown

(6) Right panel of Figure 2: Run Ex3_Unknown, and then Plot_Ex3_Unknown

(7) Left and middle panels of Figure 3: Run Ex1_ErrorRate, and then Plot_Ex1_ErrorRate

(8) Right panel of Figure 3: Run Ex1_RegretRate, and then Plot_Ex1_RegretRate

(9) Left panel of Figure 4: Run Car_TLR_UCB, Car_LR_UCB, Car_A_UCB_V, and then Plot_Car_Known

(10) Right panel of Figure 4: Run Car_EI_TLR, Car_A_UCB, and then Plot_Car_Unknown

Appendix:

(1) Figure 1: Run Ex1_Sensitivity, and then Plot_Ex1_Sensitivity

(2) Left and middle panels of Figure 2: Run Ex2_ErrorRate, and then Plot_Ex2_ErrorRate

(3) Right panel of Figure 2: Run Ex2_RegretRate, and then Plot_Ex2_RegretRate

(4) Left and middle panels of Figure 3: Run Ex3_ErrorRate, and then Plot_Ex3_ErrorRate

(5) Right panel of Figure 3: Run Ex3_RegretRate, and then Plot_Ex3_RegretRate

(6) Figure 4: Exactly Figure 1 in the main paper + left panel of Figure 4 in the main paper

(7) Figure 5: Run Ex4_Gap, and then Plot_Ex4_Gap

(8) Table 1: Run Ex123_Subgap

(9) Table 2: Run Ex4_Subgap