The Price of Personalization: An Application of Contextual Bandits to Mobile Health

A THESIS PRESENTED
BY
ISAAC XIA
TO
THE DEPARTMENTS OF COMPUTER SCIENCE AND MATHEMATICS

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
BACHELOR OF ARTS
IN THE SUBJECTS OF
COMPUTER SCIENCE AND MATHEMATICS

HARVARD UNIVERSITY
CAMBRIDGE, MASSACHUSETTS
MARCH 2018

Thesis advisor: Professor Susan Murphy Isaac Xia

The Price of Personalization: An Application of Contextual Bandits to Mobile Health

Abstract

One goal in healthcare is to be able to accurately personalize treatments, including the ability to maintaining overall efficacy in treatment while minimizing the harm and quantity of mistreated patients. With the recent prevalence of mobile devices, rapid collection of data was made possible to leverage in personalizing treatment of long-term diseases, as in the HeartSteps study, an adaptive mHealth (mobile health) intervention application for cardiovascular maintenance.

We frame the HeartSteps study as a contextual multi-armed bandit (MAB) problem, a reinforcement learning setting in which the agent must choose the optimal treatment action among several based on contextual information.

We investigate and test the use of several different variants of the Thompson Sampling heuristic, a lightweight but effective reinforcement learning algorithm, to solve the Multi-armed Bandit problem as applied to HeartSteps. Experimental bootstrapping results are interpreted and then used to corroborate theoretically backed modifications to Thompson Sampling, guiding the future design of HeartSteps to maximize overall treatment performance while minimizing variance of per-patient performance. Through these evaluations, we examine the price of personalization, or the trade-off between optimizing the overall treatment efficacy versus optimizing the fairness of individual treatment efficacies.

Contents

This thesis is dedicated to my mother Yan, my father Leon, and sister Amanda, to whom I owe both the joy and pain of growing up.

Acknowledgments

I dedicate my gratitude to my advisor, Prof. Susan Murphy. Without her kindness, I would not have had the opportunity to work with this project, and without her vision, I would not have been able to overcome the gaps I could not see.

I am furthermore deeply indebted to my lab advisors, Peng Liao and Kristjan Greenewald. Thank you both for your immense patience for answering my incessant lines of questioning.

Finally, I thank my blinkmates for their endless care. If not for the streams of encouragement, their constant source of happiness, or the numerous late-night but fruitful work sessions accompanied by Eliza, I would not have had the courage to finish this endeavor.

1

Introduction

Over the past decade, the increasing convenience and power of smartphones (e.g. Apple iPhones or Android-powered phones) and mobile wearables (e.g. Fitbit Alta, Apple Watch, Jawbone UP) have induced furious adoption by the public. These devices come packed with data collecting sensors able to detect a wide swath of information, ranging from automatic collection of location data and physical activity data to user check-ins on dietary data or mental state-of-mind. While this data is usually collected for the individuals to track their own lifestyles or for intellectual endeavors of the companies making the devices, there are a large number of life-changing applications[10].

While classical healthcare models have previously been composed of intermittent trips to the doctor's office, sensors on these mobile devices allow for continuous monitoring of health indicators that greatly enhance long-term conditions, including heart disease, arthritis, or diabetes. As such, the healthcare industry has started to use indicators of health, such as physical activity or mental state-of-mind to adopt methods in utilizing patient data for personalizing and enhancing individual

treatments. This use of mobile devices in healthcare is called Mobile Health, or mHealth for short.

One promising mHealth application is to lower heart disease risks by using the frequent contact that individuals have with their mobile devices to encourage physical activity. From a naive but well-intentioned researcher's perspective, it may seem that the optimal course of action is to frequently prompt patients to engage in physical activity; however, from a behavioral standpoint, it is often times not ideal or even feasible for patients to engage in physical activity, such as when a patient discovers it is raining outside, has just returned from a long walk, is occupied with work, or even is in the middle of commuting. Furthermore, inundating patients with suggestions of physical activity when it is not possible may cause disengagement due to overburdening, disinterest, or distrust of the application. Such a mHealth application must be carefully tuned to give well-timed suggestions that "intervene" with a patient's ordinary day.

HeartSteps is an academic mobile application that aims to address the given problem: optimally delivering small but precisely-timed personalized interventions to maintain physical activity. While it is possible to cater to the average user, HeartSteps must factor in both the contexts in which it is delivering interventions, as well as information about individual patients' behavioral responses. This type of mHealth application is known as "Just-In-Time Adaptive Interventions," and not only has the benefit of minimizing the invasiveness of each suggestion (compare several suggestions a day of standing up for a minute each versus a broad suggestion to a broad suggestion each day of walking several miles), but also lends itself to statistical analysis with its plentiful data.

In this thesis, we apply variants of a rapidly-learning reinforcement learning algorithm called the Thompson Sampler to the HeartSteps mobile health problem. We experimentally investigate and evaluate the impact of personalizing individual patient's treatments to the overall efficacy of each variant algorithm using real trial data, and use these results as guidelines to craft the next iteration of HeartSteps.

Exercise should be regarded as tribute to the heart.

Gene Tunny

2

HeartSteps

In the 21st century, cardiovascular disease is the leading cause of death across all regions other than Africa[9]. Fortunately, while treatment is often difficult, McGill et al. [8] estimates that nearly 90% of cardiovascular diseases are preventable and are associated with several key risk factors. One key preventative measure is the incorporation of physical activity, which can be readily addressed via mHealth using JITAI – a novel study investigating the efficacy and optimal methodology of suggesting physical activity is *HeartSteps*.

In the HeartSteps study, multiple small check-ins are scheduled each day at which point one of three actions may be taken by the application: no message may be transmitted, an anti-sedentary message to get up may be transmitted, or a physical activity message to take a walk may be transmitted. Each suggestion was delivered as a mobile notification to the user at the same decision time point during the day and was intended to generate an immediate effect of getting up or walking

around within the subsequent hour.

We introduce the purpose and protocol of HeartSteps v1 as in Smith et al. [11], the first iteration that randomly selected actions and observed behavior, as well as guiding principles for HeartSteps v2, a future iteration of the study that will utilize the learned data and apply reinforcement learning to personalize treatments to patients.

2.1 Randomized Choices: HeartSteps v1

In the first iteration of HeartSteps, actions were randomly selected; no message was sent with probability 60%, and physical activity as well as anti-sedentary messages were sent with probability 20%. This unequal balance was chosen to reduce participation burden on the users. Each of the two suggestions types was furthermore customized based on specific contextual data measured by the user's mobile phone or fitness tracker, such as weather conditions, location information, and recent step-count.

The $D = 5$ decision points throughout the day were scheduled corresponding to the morning commute, mid-day, mid-afternoon, evening commute, and post-dinner times, and are chosen with user input to minimize inconvenient suggestion times.

For the purposes of this project, we only consider the difference between sending no suggestion (encoded as action 0) and sending a physical activity suggestion (encoded as action 1).

2.1.1 HeartSteps Data and JITAI Notation

We adopt the following notation as in Liao et al. [7], summarized in table 2.1. Throughout, we refer to times as a tuple (t, d) of day $1 \leq t \leq T$ and decision time point $1 \leq d \leq D = 5$, and use the notation $(t, d) + k$ as k decision time points after the time point of (t, d), so that

$$(t,d)+1=\begin{cases}(t, d+1);\ 1 \leq d \leq 4\\ (t+1, 1);\ d=5.\end{cases}$$

At every time point (t, d) of day $1 \leq t \leq T$ and decision point $1 \leq d \leq D$, there is a contextual feature vector $S_{(t,d)} \in \mathcal{S}$ with 7 elements, a chosen action $A_{(t,d)} \in \{0, 1\}$ of activity suggestion, and the observed reward $R_{(t,d)+1} \in \mathbb{R}$.

Note that the reward $R_{(t,d)+1}$ has the subsequent time index, but results from observation of taking action $A_{(t,d)}$ in context $S_{(t,d)}$. To summarize a history of the data, we denote the set of variables up until time (t, d) as $\mathcal{H}_{(t,d)} \left\{ S_{(t',d')}, A_{(t',d')}, R_{(t',d')+1} \right\}_{(t',d')<(t,d)}$.

2.2 The Power of Learning: HeartSteps v2

Using the randomized data collected from HeartSteps v1, the next phase of the study is to apply this data to both test variations of reinforcement learning algorithms and provide priors on their initializations in the next iteration, HeartSteps v2. In this section, we discuss overarching goals to guide the design of our algorithm to motivate the design of our algorithm subsequent sections.

Our primary goal is to maximize the reward throughout the time period, which is the overall number of steps. However, due to the complex and individualized nature of human behavior, there is an inherent exploration-exploitation trade-off. We want to ensure that the HeartSteps algorithm does not suffer from a poor initialization and forever select actions based on its poor reward model, hereby invoking exploitation too quickly.

Next, our model should be flexible enough to account for non-stationary reward functions, as users may change their preferences over the course of the study based on latent contextual features.

Finally, we aim to avoid user disengagement from the application, which may lead to disuse or uninstallation. Disengagement can be caused by a variety of reasons relating to overburdening, distrust, or annoyance – if users receive too many suggestions to walk around, receive suggestions to walk outside in inclement weather, or receive suggestions to walk around shortly after completing a brief jog, they are more likely to disengage from future suggestions, prematurely ending or compromising treatment for the user.

Table 2.1: HeartSteps v2 Notation

Term	Name	Description
$\mathcal{S}, S_{n,t,d}$	Context	Set of 7 contextual features
p_1	Baseline features dimension	Full model consists of 7 features with 1 bias term, Small model consists of 2 with 1 bias term
p_2	Interaction features dimension	Full model consists of 3 features with 1 bias term, Small model consists of 2 with 1 bias term
$\mathcal{A}, a_{n,t,d}$	Actions	Binary – 1: active message sent, 0: no active message sent
$\mathcal{R}, R_{n,t,d}$	Reward	Log-transformed step count in 30 minutes following decision point
$f_1 : \mathcal{S} \to \mathbb{R}^{p_1}$	Baseline feature mapping	Maps context to baseline features
$f_2 : \mathcal{S} \to \mathbb{R}^{p_2}$	Interaction feature mapping	Maps context to interaction features, which are multiplied by $\mathcal{A}$
Θ	'True' generative model coefficients	From regression on HSv1 data: $\mathcal{R} \sim [f_1(\mathcal{S}), f_2(\mathcal{A} \cdot \mathcal{S})]^T \Theta$
$\varepsilon, \epsilon_{n,t,d}$	Linear model residuals	Residuals from HSv1 data after regression
N	Number of users	$N = 37$ in HSv1; $N = 2500$ in simulations
T	Number of days in study	$T = 42$
D	Number of decision points per day	$D = 5$
K	Number of cross-validations per test	Set to $K = 3$

My last piece of advice to the degenerate slot player who thinks he can beat the one-armed bandit consists of four little words: It can't be done.

John Scarne

3

Contextual Multi-armed Bandits

There is a fundamental trade-off between the accuracy of a trained model versus the amount of time necessary to train the model in reinforcement learning. Some RL techniques train incredibly precise and accurate models, such as through use of Markov decision processes in Sutton & Barto [12], but require immense amounts of data and time to train. On the other hand, simply associating the reward distributions with actions as in the *multi-armed bandit problem* is quick to learn and can run off of minimal data. However, this does not sufficiently personalize interventions for patients, especially in mHealth. To effectively personalize the treatment, the interventions must take into account contextual factors that have potential predictivity of the reward function; this is the setup of the *contextual bandit* problem, which is solved by the *Thompson Sampling* heuristic.

3.1 Contextual Bandits and Thompson Sampling

The most basic form of an action-reward learning problem is known as the multi-armed bandit problem, where an agent is presented K choices at any time, each with their own reward distribution; an analogy is to place yourself in a row of K slot machines, hence the name multi-armed bandit as a play on one-armed bandits, a nickname for slot machines operated by a single lever.

The more complicated problem applicable in the HeartSteps setting is to assume that the reward distribution of taking any action depends on the contextual state in which the action is taken. For every time step, the algorithm receives a context $S_{(t,d)}$, selects an action $A_{(t,d)}$ to play, and observes reward $R_{(t,d)+1}$. How the algorithm selects the action is known as the *contextual bandit* problem.

The method through which actions are selected can be thought of as being in one of two phases at any given time: exploration of the reward functions as they associate with rewards, and exploitation once it has deemed to have learned the rewards well enough. Two simple heuristics exemplifying each of the exploration and exploitation phases are the random and greedy heuristics. The random algorithm always selects an action at random, while the greedy heuristic always selects the action maximizing the overall reward according to its own model; these can be combined into an algorithm known as ε-greedy, which with probability ε selects a random action and otherwise greedily selects the maximal reward action.

3.1.1 Thompson Sampling

A well-known Bayesian heuristic called Thompson Sampling melds these two phases together[1]. Thompson Sampling assumes that each reward function depends on the state, action, and a latent parameter that must be learned and updated using Bayesian learning; specifically, there are the following variables and distributions:

1. The history $\mathcal{H}_{(t,d)}$ of associated contexts, actions, and states.
2. A distribution of the reward $P\left(R_{(t,d)+1}|\Theta_{(t,d)}, S_{(t,d)}, A_{(t,d)}\right)$ dependent on the associated context and action as well as a parameter $\Theta_{(t,d)}$.

3. A prior on the parameter P_Θ, along with posterior $P_\Theta\left(\theta|\mathcal{H}_{(t,d)}\right) \propto P_\Theta(\theta)P_{\mathcal{H}_{(t,d)}}\left(\mathcal{H}_{(t,d)}|\Theta_{(t,d)}\right)$.

The Thompson Sampling heuristic first samples a value $\hat{\Theta}$ of the parameter Θ according to its posterior at the time, then takes the action $A_{(t,d)}$ that maximizes the expected reward according to the reward distribution $P\left(R_{(t,d)+1}|\hat{\Theta}, A_{(t,d)}, S_{(t,d)}\right)$ based on the sampled $\hat{\Theta}$ and context $S_{(t,d)}$. Upon observing the reward $R_{(t,d)+1}$, the new history $\mathcal{H}_{(t,d)}$ is used to update the posterior of Θ, allowing for a new iteration.

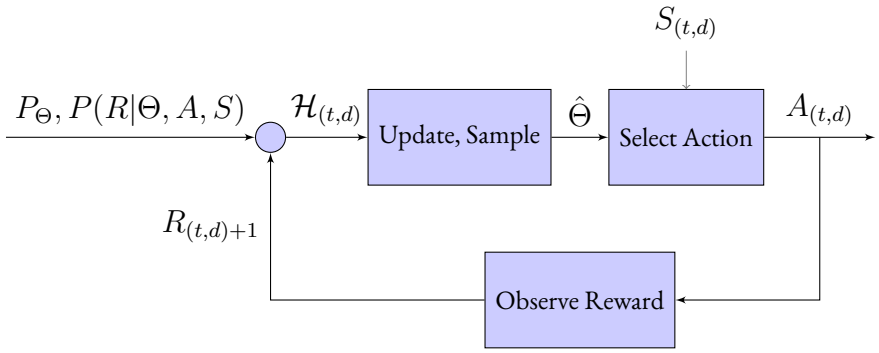


Figure 3.1: Thompson Sampling heuristic for multi-armed contextual bandits

One particular reward distribution is to assume a linear model on the created features, the baseline and the interaction terms. Thus, if $S_{(t,d)}$ is our contextual feature, we create stacked feature vector $F_{(t,d)} = \begin{bmatrix} f_1(S_{(t,d)}) \\ A_{(t,d)} \cdot f_2(S_{(t,d)}) \end{bmatrix}$, where $f_1 : \mathcal{S} \to \mathbb{R}^{p_1}$ linearly maps contextual feature vectors to a baseline feature set, and $f_2 : \mathcal{S} \to \mathbb{R}^{p_2}$ linearly maps contextual feature vectors to an interaction set, which is multiplied by the associated binary action $A_{(t,d)}$.

The reward then is assumed to be normally distributed as $R_{(t,d)+1} \sim \mathcal{N}\left(F_{(t,d)}^T\Theta, \sigma^2\right)$, with some unobserved variable σ^2 that is part of the data generating process.

In this project, we use variants of the Thompson Sampling algorithm. While other more popular heuristics exist, such as the Upper Confidence Bound (UCB) or parametric variants like the LinUCB[5], there are several reasons we stick with Thompson Sampling. First, it is very simple to implement and tune, having minimal internal parameters. Second, because of the non-deterministic action selection of Thompson Sampling, it is less liable to sub-optimal reward due to delayed effects.

These delayed effects are likely prevalent in HeartSteps, for example, occurring if a user is unable to act on an activity suggestion during the hour it is presented but will remember the missed suggestion for the rest of the day. Finally, it is highly competitive with UCB-type algorithms and definitively better than ε-greedy algorithms, as experimentally shown by Chapelle & Li [2].

3.2 Application Survey of Contextual Bandits

Contextual bandits have not been applied in the emerging field of mobile healthcare, but have seen wide applications in other personalization recommendation-based systems, such as in display advertising services or personalization of daily news article selections.

Li et al. [6] explored contextual bandits with generalized linear models and showed success in offline evaluation on personalization of Yahoo! front page advertisement placements, while Chapelle et al. [3] experimentally found Thompson Sampling to have lower regret results than LinUCB or ε-greedy approaches in Yahoo's Right Media Exchange (RMX)'s data, one of the largest exchanges between online publishers and advertisers.

Although these approaches may be similarly applied to JITAI systems, the domain science is quite different. While it is more important to maximize the total reward in advertising or individual rewards in news recommendation systems, mobile health requires careful application of learning algorithms. The potential to do harm in healthcare is much higher than in the above fields, and patients cannot be interchangeably ignored to further the learning algorithm or increase the overall reward, especially in more sensitive sub-disciplines, such as drug testing. As such, while we aim to personalize HeartSteps action suggestions to users, we also must be mindful of the price on unlucky users.

This modeling thing, it's pretty easy, but actually it's also really tough.

Cara Delevigne

4

Models and Algorithms

Several different variants of Thompson Sampling applied to the multi-armed contextual bandit problem were investigated to test feasibility in the HeartSteps mobile application for future iterations. Each variant tested the inclusion or exclusion of a set of features of the bandit algorithm, as detailed in this section.

4.1 Replayer: True Reward Model

We require setting a 'True reward model in order to run and evaluate the Bandit algorithm variants. However, we do not have such a reward model from HeartSteps v1 data, because we only have the observed reward associated with one of the possible actions taken at every time point. As such, we create a replayer as in Li et al. [6], which is an unbiased estimator that we can use for offline evaluation, and furthermore can be used to compare different Bandit variants by keeping the same true genera-

tive model throughout; future work may investigate whether the same findings hold under different generative models.

To generate the 'True' rewards from contexts, we set the 'True' generative as the full set of features, and only feed in contexts from true data. As we feed in real contexts, we can utilize the actual rewards to form estimators of generated rewards for actions that were not selected; this is explored further in sections 5.4 and 5.5.

All contextual features are defined in Table 4.1.

4.2 Bandit Reward Models

Two types of reward models were specified for use in the Bandit algorithms: the *Full model* and the *Small model*. The Full model uses the all of the engineered features, while the Small model uses smaller set of features to train over, to determine efficacy of including additional features in the Bandit's generative model. Features in the Full model were previously selected based on domain science and statistical significance, while Features in the Small model were selected by Backward Stepwise Regression from the Full model, removing features until all were significant in the regression.

In most of the tests, we specify the Bandit's reward model to be the same as the 'True' reward model, sans the residual reward from OLS that we add back in that emulates the misspecified portion of the 'True' reward model. However, in some tests, we emulate further misspecification by removing additional features of the context and using the *Small model* for the Bandit's reward model.

4.3 Bandit Algorithm Features

In this section, we describe features non-standard to Linear Contextual Thomson Sampling that are used in all of our Bandit algorithms.

Table 4.1: Contextual Features in HeartSteps

Feature	Description	Interact	Small	Small Interact
Study Day	Participant's day in study, with gaps in time removed	✓		✓
Work location indicator	Binary indicator for location at user's work or not		✓	
Other location indicator	Binary indicator for location not at user's work nor home	✓		✓
Standard deviation of step count in last 7 days	Computed as standard deviation of total step counts in hour-long interval [decision point time $-$ 30 mins, decision point time $+$ 30 mins] from each of the last 7 days	✓	✓	
Step count in previous 30 minutes	Computed as log(Tracker step count in 30 minutes before decision point$+0.5$)		✓	
Square-root steps yesterday	Computed as $\sqrt{\text{Total tracked steps from previous day}}$		✓	
Temperature	Outdoor temperature at time of decision point			
Reward: Step count in 30 minutes following decision point	Computed as log(Tracker step count in 30 minutes after decision point $+ 0.5$)			

4.3.1 Batch Model Update: Daily Model Updates

While standard bandit models update their internal models after each action-reward observation, we opt to update our models only at the end of the day, batch updating 5 decision time points together. This yields several benefits, the foremost of which is to protect against overfitting – updating less often buffers against the noisy contextual data, as well as giving the model a holistic view into the

user's behavior over an entire day rather than for different hours throughout a day. A secondary benefit is that this allows the HeartSteps mobile application to only contact the cloud for statistical learning only once a day, which can be valuable for conserving battery-life.

We opt to always include this feature in our variants of the Bandit, but in future work it may be valuable to consider varying the amount of time between each update of the Bandit's generative models.

4.3.2 Time Varying Reward Function: Gaussian Process Prior

In the standard multi-armed bandit setting, we must make the assumption that the reward function $r(S_t, A_t)$ does not change over time. However, in the HeartSteps setting, it is not clear that the reward function does not change, as users' preferences for activity change as they progress through the duration of the study, and furthermore there may be additional unobserved contexts. To model this change, we allow for a time-varying reward function by setting our prior to be a Gaussian process, which allows baseline reward to be IID and time invariant under the assumption that the reward coefficients sequence $\Theta_{(t,d)}$ are marginally normal according to the initialized prior parameters[4]. In addition, a secondary benefit of using a non-stationary reward model is that this forces the Bandit algorithm to continue exploration, as the Bandit algorithm is forced to adapt to changing environments; these updates are maintained on line 17 of algorithm 1.

The Gaussian Process Prior has an update rate parameter γ and prior parameters $(\mu_\Theta, \Sigma_\Theta)$, and is a time-varying set of model coefficients of $\{\Theta_{(t,d)}\}_{(t,d)} \leq (T, D)$ over the course of the study.

For a single user, our reward model is

$$R_{(t,d)+1} = \begin{bmatrix} f_1(S_{(t,d)}) \\ A_{(t,d)} \cdot f_2(S_{(t,d)}) \end{bmatrix}^T \Theta_{(t,d)} + \epsilon_{(t,d)} = F_{(t,d)}^T \Theta_{(t,d)} + \epsilon_{(t,d)}, \tag{4.1}$$

$$\epsilon_{(t,d)} \overset{iid}{\sim} \mathcal{N}\left(0, \sigma^2\right). \tag{4.2}$$

Note that we assume iid reward residuals, and that the reward from context/action $S_{(t,d)}, A_{(t,d)}$

at time (t, d) yields a reward $R_{(t,d)+1}$ measured at the 'next' time.

Denote a history of contexts, actions, and rewards at time (t, d) as all information gathered before that time; that is,

$$\mathcal{H}_{(t,d)} := \left\{S_{(t',d')}, A_{(t',d')}, R_{(t',d')+1}\right\}_{(t',d')<(t,d)}. \tag{4.3}$$

Then, our Gaussian Prior updates knowing the history are are defined by

$$\Theta_{(t,d)}|H_{(t,d)} \sim \mathcal{N}\left(\mu_{(t,d)}, \Sigma_{(t,d)}\right); \tag{4.4}$$

$$\mu_{(t,d)} = \gamma\mu_{(t,d)-1} + (1-\gamma)\mu_\Theta, \tag{4.5}$$

$$\Sigma_{(t,d)} = \gamma^2\Sigma_{(t,d)-1} + (1-\gamma^2)\Sigma_\Theta. \tag{4.6}$$

To derive the posterior, we find the joint distribution of $\begin{bmatrix}\Theta_{(t,d)} \\ R_{(t,d)+1}\end{bmatrix} | \left\{\mathcal{H}_{(t,d)}, S_{(t,d)}, A_{(t,d)}\right\}$, which is Gaussian itself.

Recalling our model from 4.1 for the reward $R_{(t,d)+1}$ and that $S_{(t,d)}|\mathcal{H}_{(t,d)} \perp\!\!\!\perp A_{(t,d)}|S_{(t,d)}, \mathcal{H}_{(t,d)}$, we have that

$$\text{Cov}\left(\Theta_{(t,d)}, R_{(t,d)+1}\right) = \text{Cov}\left(F_{(t,d)}^T\Theta_{(t,d)} + \epsilon_{(t,d)};\ \Theta_{(t,d)}\right) = F_{(t,d)}^T\Sigma_{(t,d)},$$

so we obtain the joint distribution:

$$\begin{bmatrix}\Theta_{(t,d)} \\ R_{(t,d)+1}\end{bmatrix} | \left\{\mathcal{H}_{(t,d)}, S_{(t,d)}, A_{(t,d)}\right\} \sim \mathcal{N}\left(\begin{bmatrix}\mu_{(t,d)} \\ F_{(t,d)}^T\mu_{(t,d)}\end{bmatrix}, \begin{bmatrix}\Sigma_{(t,d)} & \Sigma_{(t,d)}F_{(t,d)} \\ F_{(t,d)}^T\Sigma_{(t,d)} & F_{(t,d)}^T\Sigma_{(t,d)}F_{(t,d)} + \sigma^2\end{bmatrix}\right). \tag{4.7}$$

Combining the result giving conditionals within the Gaussian with the Sherman-Morrison inver-

sion formula, we obtain that the posterior on the additional history of $H_{(t,d)+1} = \{H_{(t,d)}, S_{(t,d)}, A_{(t,d)}, R_{(t,d)+1}\}$ is

$$
\begin{aligned}
\Theta_{(t,d)}|H_{(t,d)+1} &\sim \mathcal{N}\left(\hat{\mu}_{(t,d)}, \hat{\Sigma}_{(t,d)}\right), \text{ where :} \\
\hat{\mu}_{(t,d)} &= \mu_{(t,d)} + \left(\Sigma_{(t,d)}F_{(t,d)}\right)\left(F_{(t,d)}^T\Sigma_{(t,d)}F_{(t,d)} + \sigma^2\right)^{-1}\left(R_{(t,d)+1} - F_{(t,d)}^T\mu_{(t,d)}\right) \\
&= \mu_{(t,d)} + \left(\frac{1}{\sigma^2 + F_{(t,d)}^T\Sigma_{(t,d)}F_{(t,d)}}\right)\Sigma_{(t,d)}F_{(t,d)}(R_{(t,d+1)} - F_{(t,d)}^T\mu_{(t,d)}), \\
\hat{\Sigma}_{(t,d)} &= \Sigma_{(t,d)} - \Sigma_{(t,d)}F_{(t,d)}\left(F_{(t,d)}^T\Sigma_{(t,d)}F_{(t,d)}\right)^{-1}F_{(t,d)}^T\Sigma_{(t,d)} \\
&= \Sigma_{(t,d)} - \left(\frac{1}{\sigma^2 + F_{(t,d)}^T\Sigma_{(t,d)}F_{(t,d)}}\right)\Sigma_{(t,d)}F_{(t,d)}F_{(t,d)}^T\Sigma_{(t,d)}.
\end{aligned}
$$

4.4 Thompson Sampling Variants

Now, we describe non-standard features used in the variants of the Thompson Sampling algorithm tested. While each feature address specific problems in HeartSteps, we test how including features impact the performance of the Bandit, and whether it makes sense to exclude for the sake of parsimony.

4.4.1 Action Centering

A highly salient observation is that in order to choose the optimal action, our Bandit algorithm does not need to predict the entirety of the reward function, but just be able to estimate the interaction portion of the reward function. Specifically, rewriting our reward model as $R_{(t,d)_1} = f_1(S_{(t,d)})^T\Theta_{1:p_1+1} + A_{(t,d)}f_2(S_{(t,d)})^T\Theta_{p_1+1:p_1+p_2} + \varepsilon_{(t,d)}$, we are only concerned with the term where the action $A_{(t,d)}$ can affect the reward, and thus with the sub-parameter $\Theta_{p_1+1:p_1+p_2}$.

One way to isolate this effect is to centering the action as in Greenewald et al. [4]. Here, instead of setting the Bandit to directly use the step-count rewards of $R_{(t,d)+1)}$, we use the differential reward of $R_{(t,d)+1} - R^{(0)}_{(t,d)+1}$, where $R^{(0)}_{(t,d)+1}$ is the reward of taking action 0 at time (t, d). Then, we see

that

$$R_{(t,d)+1} - R^{(0)}_{(t,d)+1} = A_{(t,d)} f_2(S_{(t,d)})^T \Theta_{p_1+1:p_1+p_2} + \varepsilon'_{(t,d)}, \tag{4.8}$$

for sub-Gaussian noise $\varepsilon'_{(t,d)}$ with variance σ^2.

Applying this feature allows the Bandit to learn a less complex reward model, while still preserving the same reward benefits as learning the full reward model.

To apply this feature, we simply use the unbiased estimator $(A_{(t,d)} - \pi_{(t,d)})R_{(t,d)+1}$ for $R_{(t,d)+1} - R^{(0)}_{(t,d)+1}$. This appears on line 16 of algorithm 1; to remove the use of this feature, we change the line to $m_t \leftarrow [f_1(S_t), A_t f_2(S_t)]$.

4.4.2 Feedback Controller

A challenge in mHealth is to avoid user disengagement. To address this concern, the Bandit algorithm alleviates the problem of sending too many or too few suggestions by controlling $\pi_{(t,d)}$, the probability of selecting an active suggestion action $A_{(t,d)} = 1$, based on the recent dosage, which is measured as the count of active suggestion actions given to the user. If there have been too many active suggestion actions in the last few decision points, the Bandit algorithm decreases the probability of selecting an active suggestion relative to its original proposed probability. A secondary benefit is that this better allows for postmortem inference, as the researcher will have access to the result of selecting non-locally optimal actions.

Specifically, our feedback controller consists of parameters (λ, N_c, T_c). In step 9 of Algorithm 1, the feedback controller simply makes it harder to select a probability of an making an active suggestion through the limiting term $\lambda(N_t - N_c)_+$; removing the feedback controller replaces this term with 0.

4.4.3 Probability Clipping

To ensure that the bandit does not suffer from a poor initialization stay in a local maximum, we clip all action probabilities $\pi \in [\pi_{\text{min}}, \pi_{\text{max}}]$ to be within a certain probability range. For the experiment, we set $[\pi_{\text{min}}, \pi_{\text{max}}] = [0.1, 0.8]$; this allows the Bandit algorithm enough slack to sometimes choose locally suboptimal actions to further exploration of the reward function. An additional benefit is that this limits the maximum expected number of activity suggestion actions while allowing the algorithm a reasonable probability range to adapt to large changes in contexts. Without this feature, our algorithm changes in step 10 of algorithm 1, instead becoming $\pi_t \leftarrow \mathbb{P}\left[f_2(S_t)^T X > 0\right]$.

Algorithm 1: HeartSteps Full Bandit Algorithm

Data:

1. Gaussian Process Prior $(\gamma, \mu_\Theta, \Sigma_\Theta)$
2. Reward noise estimate σ^2
3. Feedback Controller parameters (λ, N_c, T_c)
4. Probability Clipping parameters $(\pi_{\min}, \pi_{\max})$
5. Bandit Reward model mappings: Baseline mapping $f_1 : \mathcal{S} \to \mathbb{R}^{p_1}$, Interaction mapping $f_2 : \mathcal{S} \to \mathbb{R}^{p_2}$.

Posterior Parameters:

1. $(\mu_{\text{start}}, \Sigma_{\text{start}})$: Start-of-day Bandit reward model posterior
2. $(\mu_{\text{cur}}, \Sigma_{\text{cur}})$: Bandit reward model posterior, updated throughout decision points

$\mu_{\text{start}}, \Sigma_{\text{start}} \leftarrow \mu_\Theta, \Sigma_\Theta$; // Set start-of-day posterior

for $1 \le t \le T = 90$ do

  $\mu_{\text{cur}}, \Sigma_{\text{cur}} \leftarrow \mu_{\text{start}}, \Sigma_{\text{start}}$; // Set current posterior

  for $1 \le d \le 5$; // Iterate through decision points

  do

    Obtain $S_{(t,d)}$, current context vector;

    Compute $N_{(t,d)}$, the dosage from past T_c decision times;

    $X \sim \mathcal{N}_{p_2}\left(\mu_{\text{cur}}[p_1 : p_1 + p_2], \Sigma_{\text{cur}}[p_1 : p_1 + p_2, p_1 : p_1 + p_2]\right)$; // Randomly sample from Gaussian posterior distribution of interaction term only

    $\pi_{(t,d)} \leftarrow \mathbb{P}\left[f_2(S_t)^T X > \lambda(N_t - N_c)_+\right]$; // Use Feedback Controller to compute unclipped randomization probability

    $\pi_t \leftarrow \min\left(\pi_{\max}, \max(\pi_{(t,d)}, \pi_{\min})\right)$; // Probability Clipping

    Return action $A_{(t,d)} \sim \text{Bern}(\pi_{(t,d)})$, collect reward $R_{(t,d)+1}$; // Bandit action and reward observation

    $\mu_{\text{cur}} \leftarrow (1-\gamma)\mu_\Theta + \gamma\mu_{\text{start}}, \ \Sigma_{\text{cur}} \leftarrow (1-\gamma^2)\Sigma_\Theta + \gamma^2\Sigma_{\text{start}}$; // Update current parameters as posterior of next decision time point

  end

  $\mu_{\text{cur}}, \Sigma_{\text{cur}} \leftarrow \mu_{\text{start}}, \Sigma_{\text{start}}$; // Reset current posterior to start-of-day posterior

  // Perform end-of-day batch update to Bandit Models

  for $1 \le d \le 5$ do

    $F_{(t,d)} \leftarrow \left[f_1(S_{(t,d)}), (A_{(t,d)} - \pi_{(t,d)}) f_2(S_{(t,d)})\right]$ // Full model feature vector, Action-Centering

    // Gaussian Process Update Procedure

$$\hat{\mu}_{\text{cur}} \leftarrow \mu_{\text{cur}} + \left(\frac{R_{(t,d)+1} - F_{(t,d)}^T \mu_{\text{cur}}}{\sigma^2 + F_{(t,d)}^T \Sigma_{\text{cur}} F_{(t,d)}}\right) \Sigma_{\text{cur}} F_{(t,d)};$$

$$\hat{\Sigma}_{\text{cur}} \leftarrow \Sigma_{\text{cur}} - \left(\frac{1}{\sigma^2 + F_{(t,d)}^T \Sigma_{\text{cur}} F_{(t,d)}}\right) \Sigma_{\text{cur}} F_{(t,d)} F_{(t,d)}^T \Sigma_{\text{cur}};$$

$$\mu_{\text{cur}} \leftarrow (1-\gamma)\mu_\Theta + \gamma\hat{\mu}_{\text{cur}};$$

$$\Sigma_{\text{cur}} \leftarrow (1-\gamma^2)\Sigma_\Theta + \gamma^2\hat{\Sigma}_{\text{cur}};$$

  end

  $\mu_{\text{start}}, \Sigma_{\text{start}} \leftarrow \mu_{\text{cur}}, \Sigma_{\text{cur}}$; // Set start of next day posterior as current posterior

end

4.5 Training Bandit Algorithm Tuning Parameters

For all variants of Thompson Sampling algorithms we utilize, we tune parameters following one of two protocols. In the first, we solely minimize the $MUER$, and in the second, we minimize the $MUER$ subject to the standard deviation of $MUER$.

Recall that there were the following parameters when using all features of our full Bandit algorithm.

- Gaussian Prior parameters $(\gamma, \mu_\Theta, \Sigma_\Theta)$
- Feedback controller parameters (λ, N_c, T_c), where N_c is the desired dosage (number of 1 actions) over the past T_c decision times, and λ controls the strength of the feedback controller
- σ^2, an estimate of the reward noise variance
- Probability clipping $(\pi_{\text{min}}, \pi_{\text{max}})$
- Baseline Features $f_1 : \mathcal{S} \to \mathbb{R}^{p_1}$, $f_2 : \mathcal{S} \to \mathbb{R}^{p_2}$. We set these to either the *Full set* or *Small set*

We set some values of parameters for which we have a good sense of prior as the following:

1. Set $\pi_{\text{min}} = 0.1, \pi_{\text{max}} = 0.8$, set by domain science of the minimum engagement and maximum burden that users can handle.
2. Set σ^2 to $\hat{\sigma^2}$, the empirical residual (noise) variance.
3. Depending on variant of Bandit, set f_1, f_2 to either the *Full model*, which are the identity mappings, or the *Small model*, defined in 4.2.
4. Set $\mu_\Theta = \mathbf{0}$ as the 0 vector in Train batches; this is motivated by the standardization of the contexts. In Test batches, set to True Training parameters.

We aim to tune the remaining parameters by cycling through each parameter, conducting a parameter sweep, and continuing for $r_{\text{cycles}} = 4$ cycles. We control the following parameters, where Optimization Param is the exact parameter we optimize, and Tuning Param is the corresponding parameter from the Bandit Model that it affects. We set several seed values according to the Starting Value based on initial random search optimization, and test within the Range.

Table 4.2: Parameters for Optimization

Optimization Param	Tuning Param	Description	Starting Value	Tested Range
`N_c_mult`	N_c	Multiplier on T_c to give N_c	0.5	$[0.05, 0.9]$
`T_c`	T_c		5	$[3, 70]$
`sig2_mult`	σ^2	Multiplier on empirical residual variance to give σ^2	1	$[0.1, 2.5]$
`gamma`	γ		0.9	$[0, 1]$
`lamb`	λ		$1.$	$[0.1, 10]$
`prior_cov_mult`	Σ_Θ	Multiplier on $\mathbb{I}$ to give Σ_Θ	0.5	$[0.1, 3]$

The order of the parameters listed in Table 4.2 are from most to least impactful from initial random search optimization. Thus, we use this ordering to optimize using Algorithm 2, where we set $r_{\text{cycles}} = 4$ to be the number of times we cycle through optimizing the list parameters.

In the second type of optimization, we also consider standard deviation of $MUER$ in our minimization. Specifically, when optimizing a parameter, we first identify the minimum standard deviation of $MUER$, then only consider values of the parameter yielding less than `StdCutoff` $= 1.1$ times the minimum standard deviation of $MUER$. This is shown in algorithm 3.

Algorithm 2: $MUER$ Minimization Optimization

Data: Starting Parameter Values, Parameter Testing Ranges
Result: Optimal parameter values
Set Parameters to Starting Parameter Values;
for $1 \leq n \leq r_{cycles}$ do
    for *Parameter in Parameter List* do
        for *Parameter Value in Parameter Testing Range* do
            Compute Mean $MUER$(Parameter Value);
        end
        Set Parameter to Parameter Value that minimizes Mean $MUER$(Parameter Value);
    end
end
Return final set of Parameters;

Algorithm 3: Standard Deviation Cutoff Optimization

Data: Starting Parameter Values, Parameter Testing Ranges, `StdCutoff`
Result: Optimal parameter values
Set Parameters to Starting Parameter Values;
for $1 \leq n \leq r_{cycles}$ do
    for *Parameter in Parameter List* do
        for *Parameter Value in Parameter Testing Range* do
            Compute Mean, StdDev of $MUER$(Parameter Value);
        end
        Set Parameter to Parameter Value that minimizes Mean $MUER$(Parameter Value), subject to

$$StdDev\left(MUER(\text{Parameter Value})\right) < \texttt{StdCutoff} * \min StdDev\left(MUER(\text{Parameter Value})\right);$$

    end
end
Return final set of Parameters;

All life is an experiment. The more experiments you make the better.

Ralph Waldo Emerson

5

Methods and Experiments

Throughout this project, we work with the HeartSteps v1 Data, hereupon abbreviated HSv1. Contextual features have been created from the measurements through domain science, through which we assume that the Bandit algorithms use linear models for the purposes of the project.

The contextual bandit algorithms used are stochastic, meaning for every user n, day t, and decision point d, the algorithm generates a probability $\pi_{n,t,d}$ of action. Thus, a taken action is generated by

$$A_{n,t,d} \sim \text{Bern}(\pi_{n,t,d}).$$

The main metric of performance on a user we use is Mean User Expected Regret, or $MUER$.

This is computed in equation 5.1 using the reward $R^{(a)}_{n,t,d}$ of either action $a = 0$ or $a = 1$ under our generative model:

$$MUER(\mathcal{S}, \Theta, \varepsilon, n) = \frac{1}{TD} \sum_{\substack{t=1,\dots,T \\ d=1,\dots,D}} \left[\mathbb{E}_{\pi^\star}(R_{n,t,d}) - \mathbb{E}_{\pi_{n,t,d}}(R_{n,t,d}) \right], \tag{5.1}$$

$$R^{(a)}_{n,t,d} = \begin{bmatrix} f_1(S_{n,t,d}) \\ a \cdot f_2(S_{n,t,d}) \end{bmatrix}^T \Theta + \epsilon_{n,t,d}, \tag{5.2}$$

$$\mathbb{E}_{\pi_{n,t,d}}(R_{n,t,d}) := \pi_{n,t,d} R^{(1)}_{n,t,d} + (1 - \pi_{n,t,d}) R^{(0)}_{n,t,d} \tag{5.3}$$

$$\pi^\star_{n,t,d} := \underset{\pi \in [\pi_{\min}, \pi_{\max}]}{\operatorname{argmax}} \pi R^{(1)}_{n,t,d} - (1 - \pi) R^{(0)}_{n,t,d} \tag{5.4}$$

where expression 5.2 is the Reward of action $a \in [0, 1]$ for user n at time/decision point t, d, expression 5.3gives the expected reward given chosen probability $\pi_{n,t,d}$ of an activity suggestion action, and expression 5.4 chooses the optimal clipped probability $\pi \in [\pi_{\min}, \pi_{\max}]$ knowing the rewards.

Note that we add in $\epsilon_{n,t,d}$ to better account for misspecification in our 'true' generative model.

For each simulation of N users, we can now compute the mean and standard deviation of $MUER$ as our main performance metrics.

5.1 Methodology Overview

For simulations, our overall methodology is the following. We first designate a variant of the Bandit algorithm, as well as a replayer to set up our 'true' generative model, using f_1, f_2, to form simulated rewards from the context and given action. Next, we split HSv1 user data into K-fold cross validation sets. For each split, we perform the below process:

1. Use the training and testing data to generate training simulated users and testing simulated users, with $N = 2500$ users in simulation.

2. Use training simulated users to tune parameters of the given Bandit algorithm based on the

mean of $MUER$ computed across all users and decision points, contingent on quality metrics.

3. Run simulated users from test split using optimally tuned parameters, observing quality metrics and impact of each tuning parameter on the mean and standard deviation of all computed $MUER$.

Each part is described in more detail in table 2.1.

5.2 Reward Generative Model

We set different 'true' generative models, where we assume:

$$\mathcal{R} = \begin{bmatrix} f_1(\mathcal{S}) \\ \mathcal{A} \odot f_2(\mathcal{S}) \end{bmatrix}^T \Theta + \varepsilon \tag{5.5}$$

and that f_1, f_2 are feature functions from our set of contextual features $\mathcal{S}$ to baseline and interaction terms, $\odot$ denotes the term-wise product, and each $\varepsilon \sim \mathcal{N}\left(0, \sigma^2\right)$ for σ^2, which is a tuning parameter with estimator $\hat{\sigma^2} = \text{Var}(\varepsilon)$.

We describe these models more in Section 4.1.

5.3 Cross-Validation

We use $K = 3$ fold cross-validation to separate training and testing batches. We do not train the bandit algorithm's parameters on the test batches, as to simulate application of HSv1 data for HSv2.

To do this, we randomly order the $N = 37$ users, then group the randomly ordered users into K groups; the K-th fold cross-validation is performed holding the K-th group out as the test sample, and the remaining groups in as the training sample. We then bootstrap from the given splits to form the requisite $N = 2500$ sample size.

For each user, we have a series across all days T and decision points t per day of Reward, Action, and Context. We will refer to them jointly as $(R, A, S)_{(N,T,t)}$, which are implicitly indexed in order by the user, day, and decision point, creating $N \times T \times t$ data points.

5.4 Residual Formation

Within each test batch and train batch, we conduct ordinary least squares linear regression to residualize additional effects, missing from our model due to possible misspecification, for each user from the baseline and interaction effects. Specifically, we use the model in Equation 5.5.

Recall that f_1, f_2 are the baseline and interaction functions, that are parameters of the generative model.

In this way, we obtain a 'true' Θ for the simulation, as well as a series of ε corresponding with each data point. This series will be used when computing the 'true' generative model reward used in training and testing the Bandit model; note that this allows us to give the actual reward obtained in HSvi when choosing the actual action used by the corresponding context from HSvi, but only getting an OLS estimate when choosing the other action.

5.5 Simulated User Generation

For both the training and test original users' series of $(\mathcal{R}, \mathcal{A}, \mathcal{S})$, we can create train and test batches of $N = 2500$ users each by randomly sampling with replacement from the train and test pools respectively. This value of N was chosen to give each training user roughly 100 runs and each testing user roughly 200, as there are $24 - 25$ users in each training split and $12 - 13$ users in each testing split, as well as to give statistical significance in our stochastic algorithm.

We note finally that in each of the $N = 37$ original HSvi users, we imputed the mean value for missing contextual feature measurements, and set availability to True when the reward and at least one of the contextual features is measured.

5.6 Simulated User Testing and Quality Metrics

Once the tuning parameters have been optimized for the batch, we run the test users on with these parameters, and analyze the quality metrics described below:

1. Time series of cumulative expected regret
2. Time series of $\pi_t(1|S_t)$, the probability of taking action 1
3. Histogram of $|\pi_t(1|S_t) - opt_t(S_t)|$

   We define $opt_t(S_t)$ as the optimal probability in context S_t:

$$opt_t(S_t) = 0.8\ \mathbb{1}\{\text{optimal action is 1 in } S_t\} + 0.2\ \mathbb{1}\{\text{optimal action is 0 in } S_t\}. \tag{5.6}$$

4. Histogram of number of actions taken per day
5. Time series of number of actions taken per day
6. Histogram of number of times per day that the feedback controller was invoked
7. Tim series of number of times per day that the feedback controller was invoked
8. Time series of MSE between 'True' Θ and the Bandit's estimate μ_Θ

6

Results

In this section, we detail the findings of our experiments, where we tested the impact of each of the features on the outcome of the Bandit algorithm, on the same batches. The features were Action Centering, Feedback Controller, Probability Clipping, Full vs Small Contextual feature set, which are abbreviated ac, fc, pc, small, respectively. The name of each variant contains just the abbreviated feature names that are in use, with the naked variant using none of the features called No Features, and the full variant called ac fc pc.

Throughout the experiments, we compute the MUER using the same optimal probabilities of either $\pi_{\text{min}} = 0.1, \pi_{\text{max}} = 0.8$, even if probability clipping was not present or in the simple randomized treatment experiment; this is to allow for comparisons between different models using the same metrics.

Furthermore, using the optimization method using a standard deviation cutoff, as in algorithm 3.

6.1 Randomized Treatment

As a baseline, we ran tests akin to the original HeartSteps v1 study, with purely random choices of action; we set this to be $\pi = 0.6$, but Figure A.1 explores the relationship as π varies between $[\pi_{\text{min}}, \pi_{\text{max}}]$ for each of the same $k = 3$ CV training and testing batches we split into as the other experiments. In general, the $MUER$ mean linearly increased or decreased depending on the set of users that were in the batch, and the $MUER$ standard deviation was convex, with an average minimum close to 0.55.

6.2 Impact of Bandit Features

We investigate the utility of including each feature of the Bandit in this section, starting with a simple survey of the $MUER$ and continuing onto the Quality Metrics, depicted in Appendix A.2.

6.2.1 Overall $MUER$

Below, we summarize statistics for $MUER$, with $N = 2500$ in each batch, for a total of $N = 7500$ in each averaged test. We evaluate the quality of a bandit variant based on two axes: minimizing average $MUER$ and minimizing the total standard deviation of $MUER$. The average $MUER$ measures the overall efficacy of the bandit variant, while the standard deviation of $MUER$ measures the fairness of treatment personalization.

In 6.2, we note that while the mean $MUER$ does not favor more feature-laden bandit variants, we see that they serve well to reduce the variance, especially in the testing set. The overall variance is minimized for variant ac fc pc, which is our full variant including action centering, feedback controlling, and probability clipping, while using the Full set of contextual features in the bandit model. In the diagram, we also present the decomposition of total variance into within-user variance and between-user variance, discussed further in Section 6.2.2.

In figure 6.1, in particular for the full variant in dotted green, we see that the slope of the training cumulative regret starts off shallower than that of testing. However, there is some concavity in the cumulative regret for the testing batches. Although the full variant has worse mean cumulative regret than most other variants at earlier time points, it evens out and is comparable to the others by the end.

One important note for figure 6.1 is that although the cumulative average $MUER$ appears to stop increasing for all variants and batches past decision point 175, it does not. Instead, it is due to the fact there were varying counts of data points for study participants, and past 175, there were few left – the flatness is due to the fact that their increasing cumulative regret is averaged out by the other users' cumulative regrets not increasing.

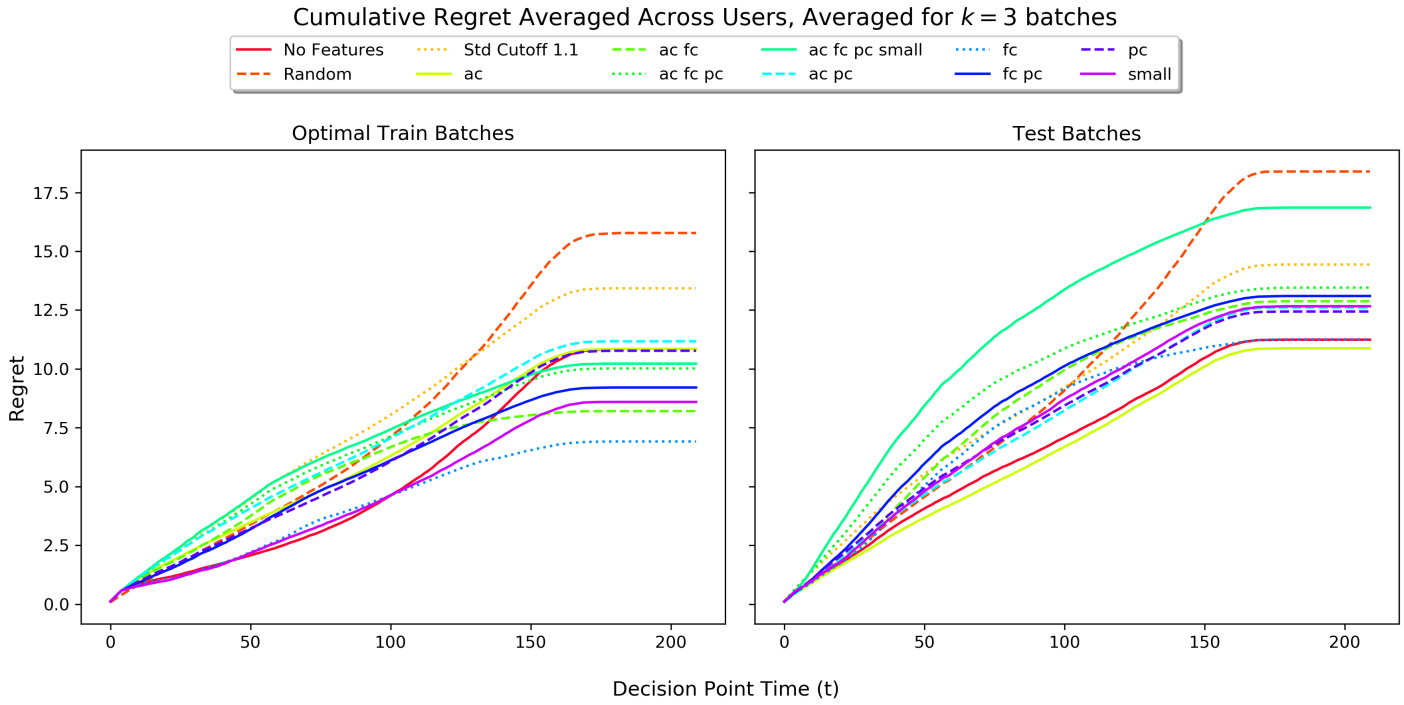


Figure 6.1: Cumulative regrets for all Bandit Variants, lower regret indicates better performance. (Separated Batches in Figure A.2)

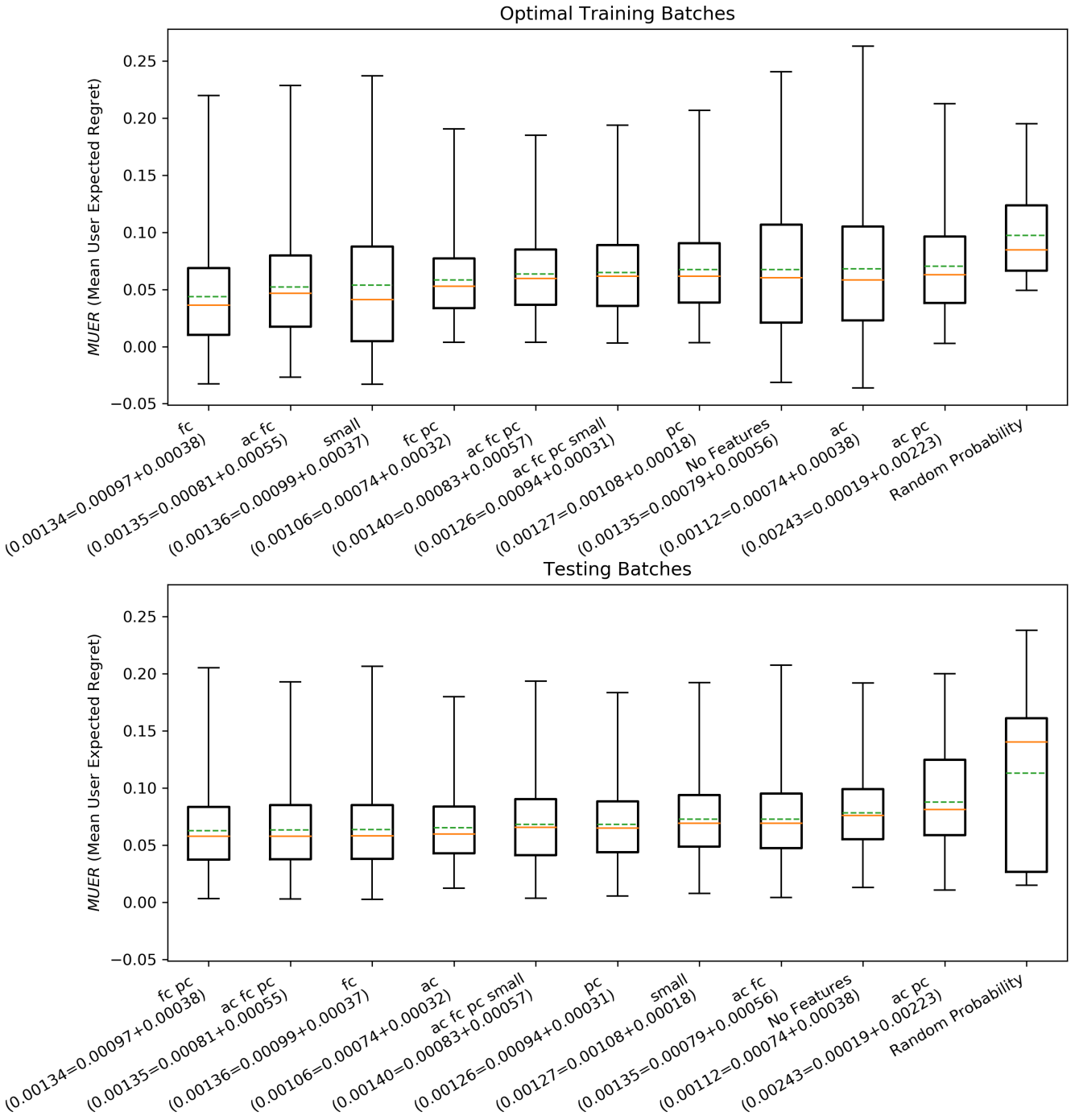


Figure 6.2: Boxplots of MUER for all Bandit Variants, sorted by mean $MUER$. Mean in Green, Median in Orange. Lower means and lower variance indicate better performance. Variance Decomposition shown in label under each Variant name: $\langle \textit{Total Var} \rangle = \langle \textit{Within Var} \rangle + \langle \textit{Between Var} \rangle$

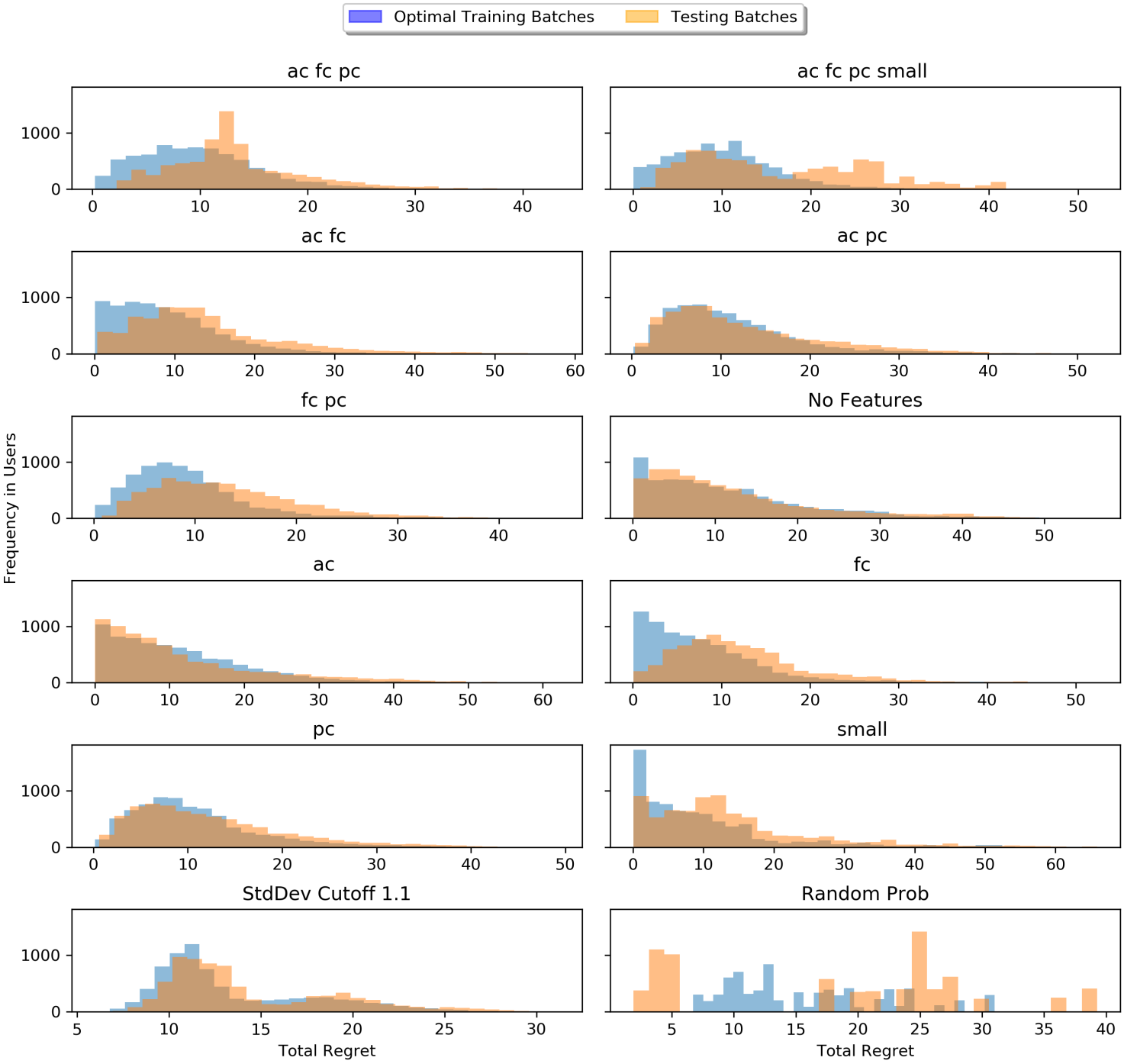


Figure 6.3: Histogram of all users' total regrets for all Bandit Variants, lower regrets indicate better performance.

6.2.2 $MUER$ Variance Decomposition

We study the decompsition of the Total Variance in simulated $MUER$ into Within-User Variance and Betweeen-User Variance. As we bootstrapped to obtain simulation users, we can reindex to match each simulation user with the i-th trial for real user n. Let $Num(n)$ denote the number of times user n was sampled within our set of simulation users.

As we reindex, we denote that simulation i for user n has $MUER$ as $MUER_{n,i}$. Then, the mean $MUER$ for user n is computed by averaging all $Num(n)$ simulation trials, which we denote as $\overline{MUER}_n$. Finally, we can denote the overall weighted mean across all users as $\overline{MUER}$, which is computed by weighting each user's mean MUER by $Num(n)$. Recalling that $N = 37$ is the number of real users from the original HeartSteps v1 study while 2500 is the number of bootstrapped simulation users, we summarize our decompsitions as such:

$$\sigma^2_{total} = \sigma^2_{within} + \sigma^2_{between} \tag{6.1}$$

$$\sigma^2_{total} := \frac{1}{2500} \sum_{n=1}^{N} \sum_{i=1}^{Num(n)} \left(MUER_{n,i} - \overline{MUER}\right)^2 \tag{6.2}$$

$$\sigma^2_{within} := \frac{1}{2500} \sum_{n=1}^{N} \sigma^2_{within,n} \cdot Num(n) \tag{6.3}$$

$$\sigma^2_{within,n} := \frac{1}{Num(n)} \sum_{i=1}^{Num(n)} \left(MUER_{n,i} - \overline{MUER}_n\right)^2 \tag{6.4}$$

$$\sigma^2_{between} := \frac{1}{N} \sum_{n=1}^{N} \left(\overline{MUER}_n - \overline{MUER}\right)^2 \tag{6.5}$$

In table 6.1, we report the simulation results; these are also depicted in the boxplots of figure 6.2.

	Batch	Total var	Within var	Between var
ac	Test	2.661305	1.854323	0.806983
No Features	Test	2.804786	1.856737	0.948049
pc	Test	3.144949	2.359168	0.785780
small	Test	3.162685	2.706956	0.455729
fc pc	Test	3.358141	2.414217	0.943925
ac fc	Test	3.382839	1.980111	1.402729
ac fc pc	Test	3.385391	2.016630	1.368761
fc	Test	3.407053	2.486476	0.920577
ac fc pc small	Test	3.504155	2.074450	1.429705
ac pc	Test	6.071459	0.486644	5.584816
No Features	Train	2.523148	1.848379	0.674769
ac fc	Train	2.592400	1.728600	0.863799
ac fc pc	Train	3.062984	2.010644	1.052340
fc	Train	3.088712	2.424037	0.664676
small	Train	3.097342	2.599052	0.498290
fc pc	Train	3.123756	2.280590	0.843166
ac fc pc small	Train	3.222406	1.889396	1.333010
ac	Train	3.239059	2.276554	0.962505
pc	Train	3.334281	2.360475	0.973806
ac pc	Train	3.719548	0.524511	3.195038

Table 6.1: Variance decomposition into Within-User and Between-User Variance, averaged across $k = 3$ split batches.

6.2.3 Action Centering

The presence of action centering does not greatly affect many quality metrics. The $MUER$ variance slightly increases with the presence of this feature.

One observation is found in A.17, we see that the naked variant and the variant with just action centering differ in the fact that the 95% and 75% percentiles for the Θ MSE is markedly smaller. In particular, we observe that this does not happen with adding just one of any of the other features, suggesting that lone action centering does well to mitigate poor training as compared to the others. However, this benefit is highly dampened when comparing the full variant with and without action centering.

6.2.4 Feedback Controller

The addition of a feedback controller slightly decreased the mean $MUER$ as well as the variance. Looking at histogram 6.3, the full variant without a feedback controller yields a heavier and longer tail versus with a feedback controller, especially for the testing batches.

While the training $MUER$ remained similar when a feedback controller was added to the naked variant, the testing $MUER$ markedly increased, suggesting that this may not be a good feature to use by itself.

6.2.5 Probability Clipping

Without probability clipping, we note that negative regret is easily achievable – this is because we keep the regret function as the same $MUER$ that is computed by choosing the optimal of π_{min} or π_{max}, as to be able to compare different models under the same metrics.

The mean $MUER$ does not change too much and in fact, decreases slightly, but we especially see that the variance and the difference between the 95% and 5% percentiles increase far more over time.

In practice, it does not seem that there are $MUER$ performance-based reasons to remove the

probability clipping, especially with its behavioral and inferential benefits. Nonetheless, future work may investigate varying levels of probability clipping to infer.

6.2.6 Small Contexts

Removing contextual features from the Full set to the Small set did not yield a large change in training batches, and while the performance between the Small and Full sets in testing batches 1 and 2 are similar, there are particular users in testing batch 3 that yielded a marked increase in the overall regret levels.

This result is surprising – we may expect that if a smaller set of contextual features would have similar training performance as a larger set, that it would be less prone to overfitting, and have better testing performance. This suggests that we require the full set of proposed features to maintain regret levels, not due to an overall diminishing of performance, but rather performance on several individuals.

The Small set also slightly pushes the generated probabilities $\pi_{(t,d)}$ downward.

6.3 Impact of Tuning Parameters

In appendix A.3.1, we depict the result of optimizing the full model (ac fc pc) on the first training batch, running through each of the 6 parameters in order of `N_c_mult`, `T_c`, `sig2_mult`, `gamma`, `lamb`, `prior_cov_mult` and cycling through $r_{\text{cycles}} = 4$ times. The following results observationally hold through the other batches.

6.3.1 `N_c_mult`

Over the parameter cycling, `N_c_mult` starts off fairly bumpy, owing to the fact that they only affect quantized values of N_c. While there are strong correspondences between `N_c_mult` and the average and StdDev $MUER$, the correspondences are not consistent. Higher values of N_c correspond with a less prevalent feedback controller, and correspond with lower $MUER$, but also correspond with a convex StdDev $MUER$ that can either be increasing or decreasing.

The largest effect on mean ranges between a 30% and 40% decrease between suboptimal and optimal values, while the effect on StdDev can be as much as a 60% decrease.

6.3.2 `T_c`

Here, we see that T_c is also fairly quantized, and in general does not impact the mean nor standard deviation of the $MUER$ heavily. Most of the time, there is a slight tendency toward decreased mean $MUER$ but increased StdDev $MUER$ as T_c increases; however, towards the end of the StdDev Cutoff Optimization routine, the impact increases on the Mean $MUER$, and the StdDev $MUER$ actually decreases with T_c.

The effects on the mean and StdDev are quite small, both ranging between $1\% - 4\%$ as a decrease.

6.3.3 `sig2_mult`

In general, we see higher values of `sig2_mult` cause to non-linear decreasing the StdDev $MUER$, and values close to 1 are chosen.

The effect on the mean is quite small, around 2%, while the effect on StdDev can range between $25\% - 40\%$.

6.3.4 `gamma`

As we tested values of $\gamma \in [0, 1]$, we see that the bulk of the effect occurs when γ is close to 1; when $\gamma = 1$ precisely, our Gaussian Process Prior becomes a standard Bayesian normal posterior update. We see consistently that higher values of γ decrease the overall mean $MUER$, but also increase the StdDev $MUER$ significantly.

The effect on mean ranges from $10\% - 25\%$, but the effect on StdDev can range from $400\% - 600\%$.

6.3.5 `lamb`

Seeing as λ controls the aggressiveness of the feedback controller once it has been invoked, we expect and observe that tuning its values has a similarly minor effect like with tuning N_c and T_c. Mean and StdDev $MUER$ remain fairly constant, but for values of $\lambda < 0.75$, the effect on the StdDev intensifies, but in a direction dependent on the other parameters. Notably, because the mean is so constant, it seems we can tune this parameter directly in favor of the StdDev $MUER$.

There is virtually no effect on the mean, except for a 1% fluctuation around $\lambda = 0$, while the effect on StdDev can be from $0\% - 5\%$.

6.3.6 `prior_cov_mult`

For the multiplier on $\Sigma_\Theta = \mathbb{I}$, we see that it does not majorly impact the mean nor StdDev $MUER$ – however, lower values of the multiplier consistently decreases the mean $MUER$ while increasing the StdDev $MUER$.

While the effect on the mean is about 1% and fairly constant the effect on StdDev $MUER$ is between $5\% - 10\%$ and always decreasing.

	MUER Min	StdDev Cutoff
N_c_mult	0.777	0.526
T_c	67.000	70.000
sig2_mult	1.111	0.191
gamma	1.000	0.736
lamb	0.349	0.100
prior_cov_mult	0.100	0.519
Train Mean	0.062	0.072
Train StdDev	0.032	0.007
Test Mean	0.080	0.080
Test StdDev	0.009	0.009

Table 6.2: Optimally tuned parameter values and performance comparisons between Optimization Routines

7

Conclusion

In this thesis, we have evaluated the performance of adding various features to variants of the Thompson Sampler, applied to the MAB problem in HeartSteps. These were used to make recommendations guiding the design of the HeartSteps v2 application.

We considered both the average efficacy of treatment in evaluation, as well as the fairness in treatment; these were measured through the mean and standard deviation of $MUER$ (Mean User Expected Regret), where lower means and lower standard deviations are more desirable.

Empirically, we have the following recommendations and findings:

- We should use the full set of contextual features, as using the full set of contextual features does not cause the bandit to overfit, but rather improves the variation between per-user $MUER$ while maintaining mean $MUER$.
- Action centering greatly reduces the complexity of the model that the bandit must correctly

specify to optimally select actions, and is seen to reduce the variance of $MUER$.

- We suggest including the Feedback controller to thin out both the intensity and frequency of patients receiving poor treatments, as well as address user disengagement concerns.
- Probability clipping must be included for the bandit to continue learning as well as not to overwhelm or disengage the user, and we have seen that the variants removing probability clipping do not serve to improve performance.

Future work may include the following suggestions.

1. Tuning the Gaussian Process Prior generally yielded a value γ very close to 1, in which case we have a stationary reward function; this suggests that the reward function is not overly non-stationary, or there are heavy delayed reward effects. It is worth exploring how to better model these effects, and whether the Gaussian Process Prior is masquerading as a solution to a more fundamental challenge.
2. Adjusting the batch update process may be investigated; although currently at the end of every day, less variation in $MUER$ or model overfitting may occur with more infrequent batch updating.
3. Variation in the data generating process model. We assumed the same linear model that generated our rewards in section 4.1, but it is unclear whether this is a valid assumption. It is worth investigating whether our findings hold if these contextual features are included in a GLM or non-linear reward model.

A

Additional Figures

A.1 Summary Figures

A.2 Quality Metric Figures

A.3 Parameter Tuning vs $MUER$ Figures

All parameters were tuned in the given range from table 4.2, with 160 steps in between (160 chosen to leverage available dedicated Odyssey Research cluster computing cores).

A.3.1 $MUER$ Minimization Parameter Optimization

A.3.2 Standard Deviation Cutoff Parameter Optimization

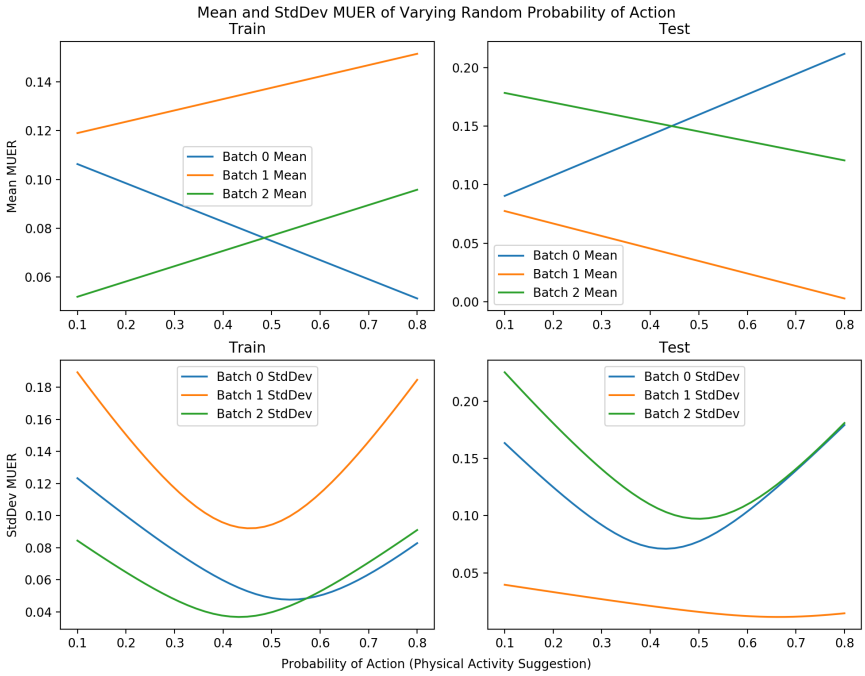


Figure A.1: Overall Regrets for Varying Action Probability $\pi_{(t,d)}$ vs $MUER$

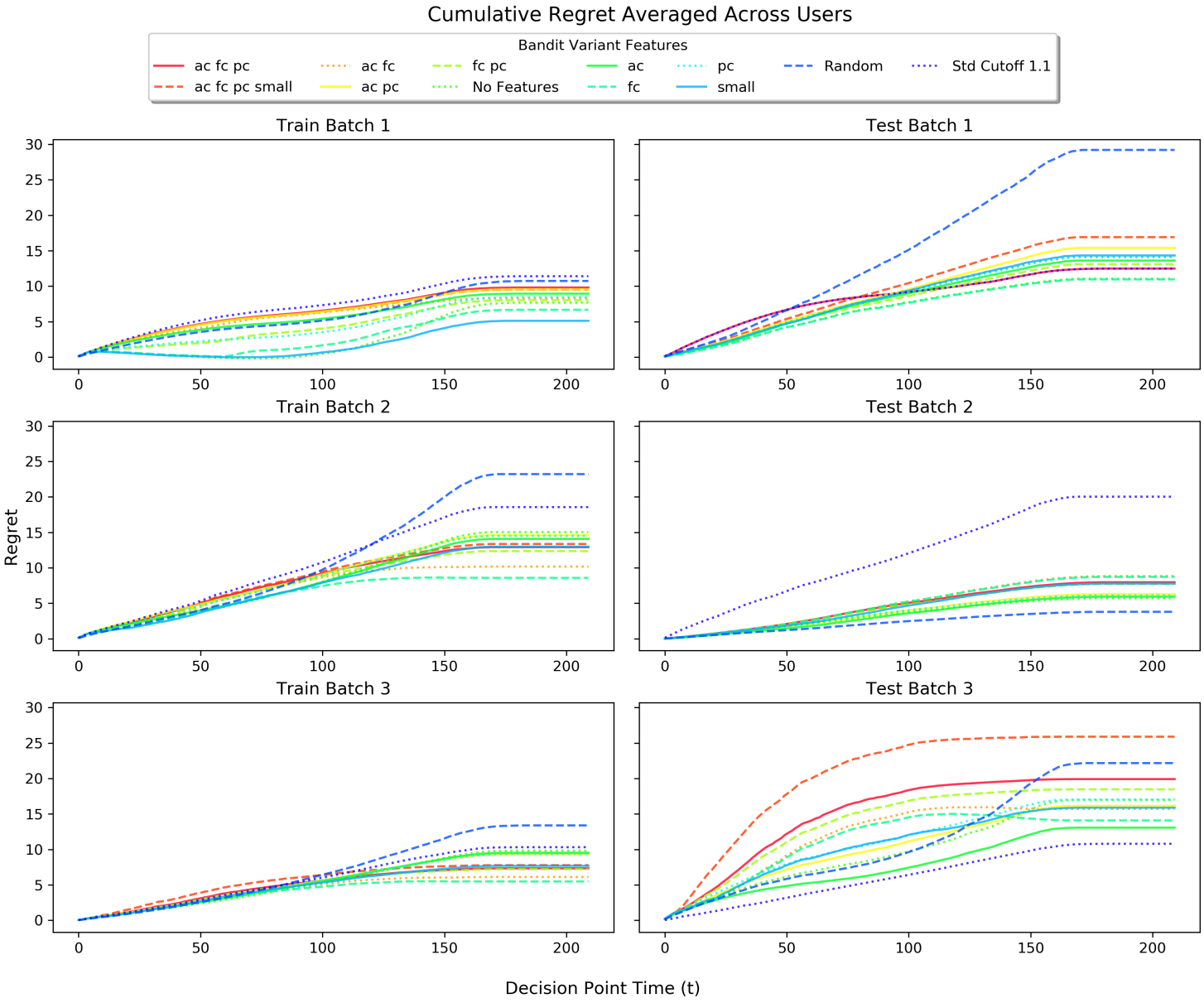


Figure A.2: Cumulative Regrets for all Bandit Variants, Separated by Batch

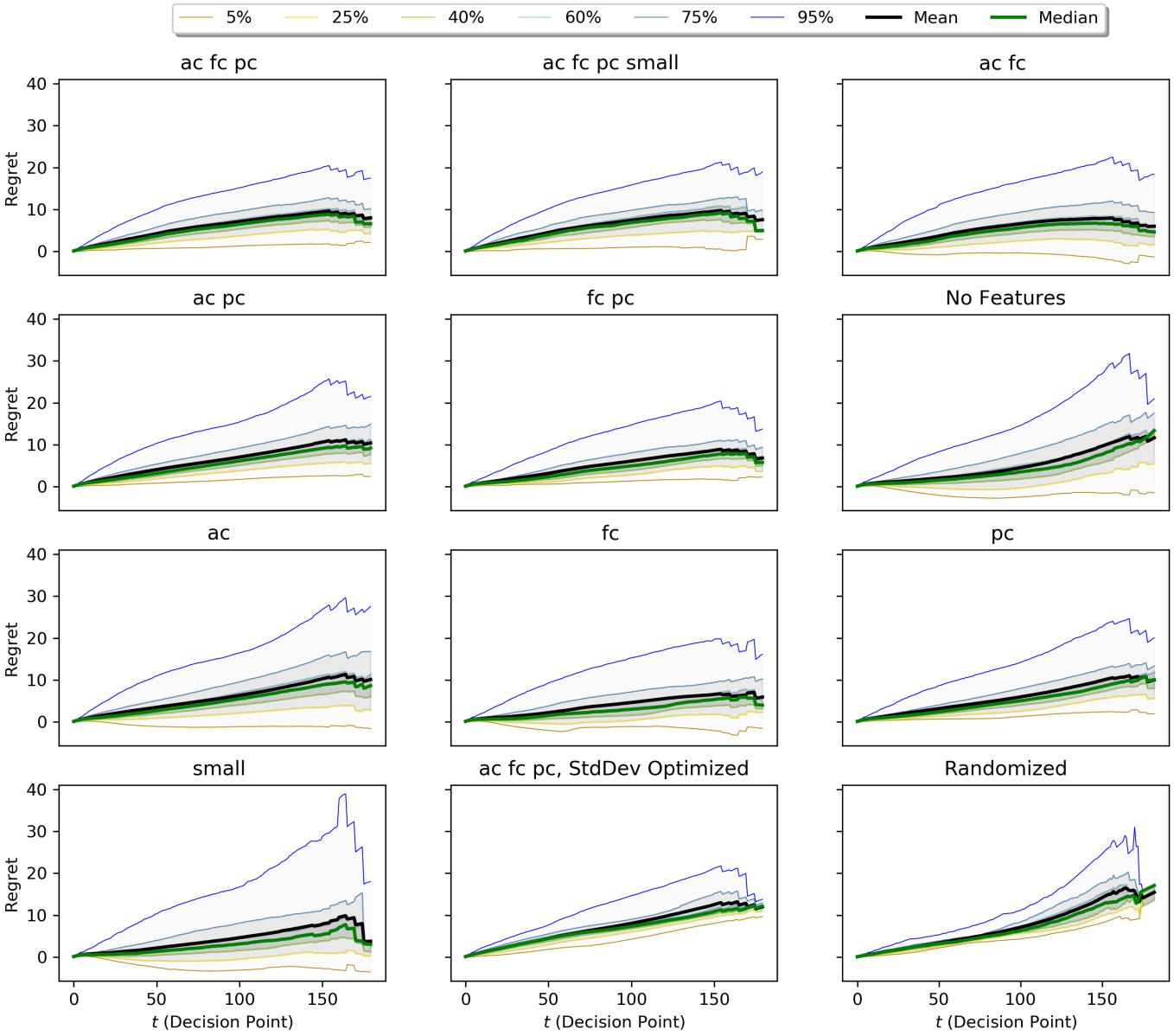


Figure A.3: Cumulative Regret over Decision Point for All Users in Optimal Training Batches

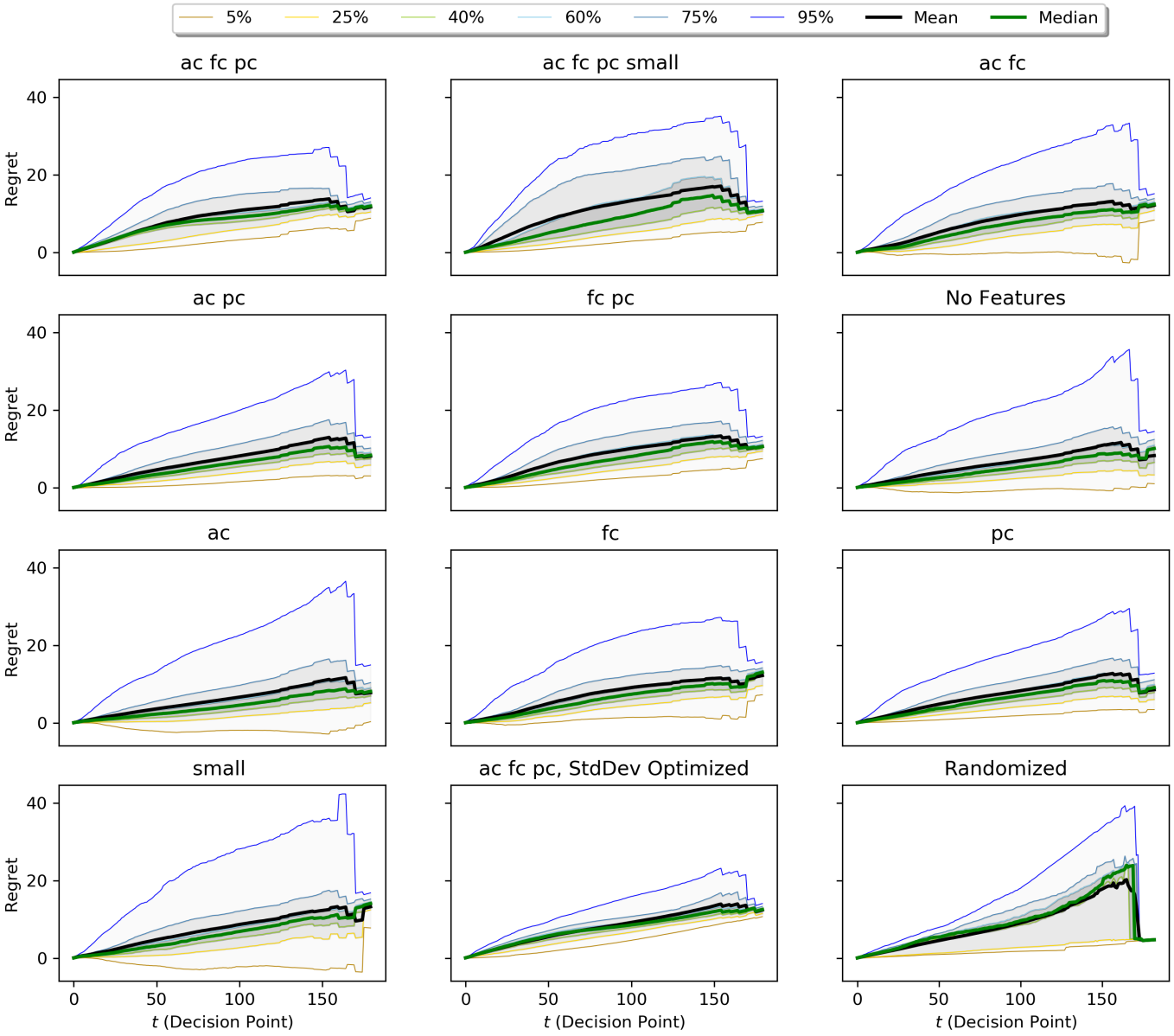


Figure A.4: Cumulative Regret over Decision Point for All Users in Testing Batches

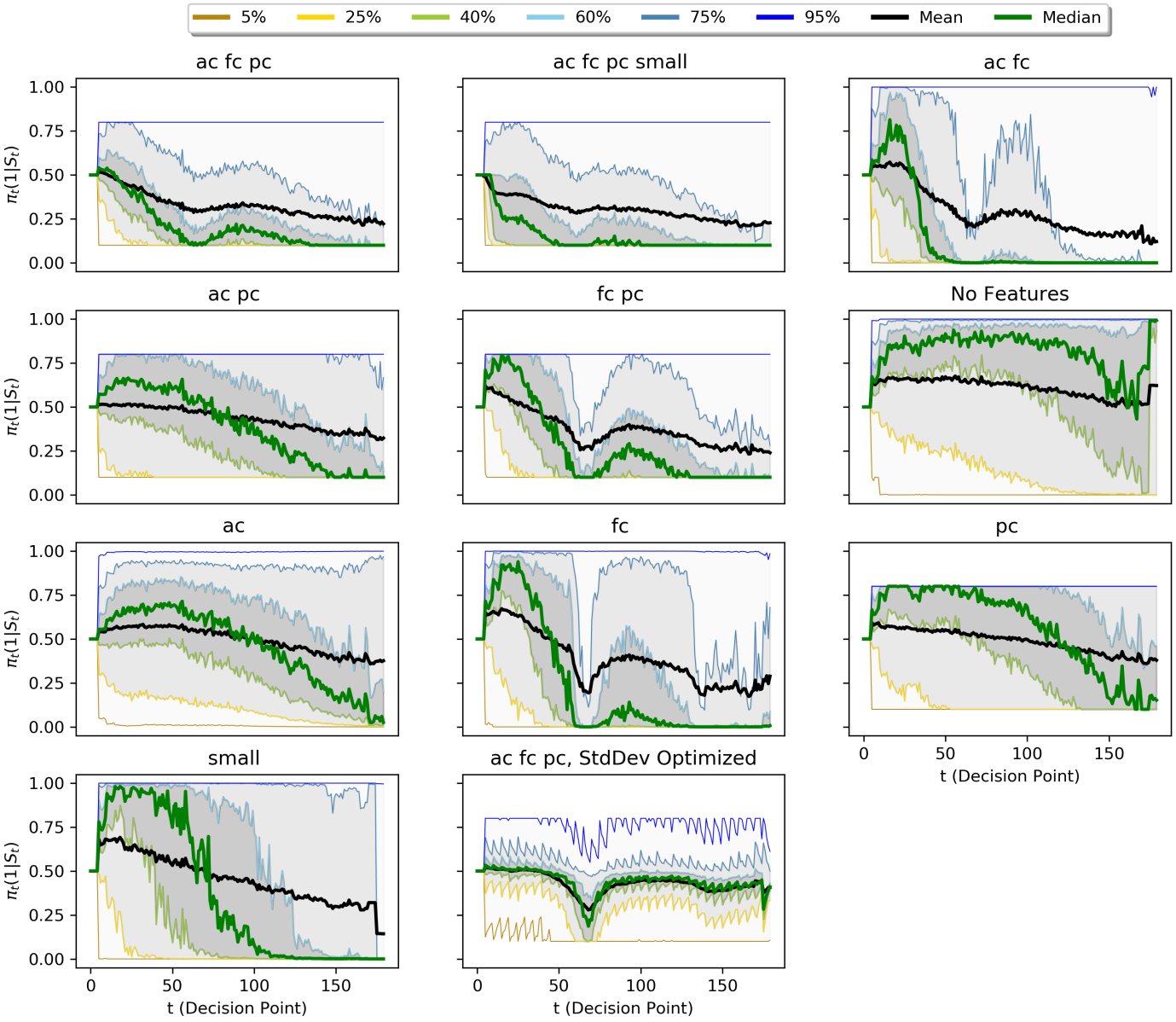


Figure A.5: Action Probability over Users in Optimal Training Batches

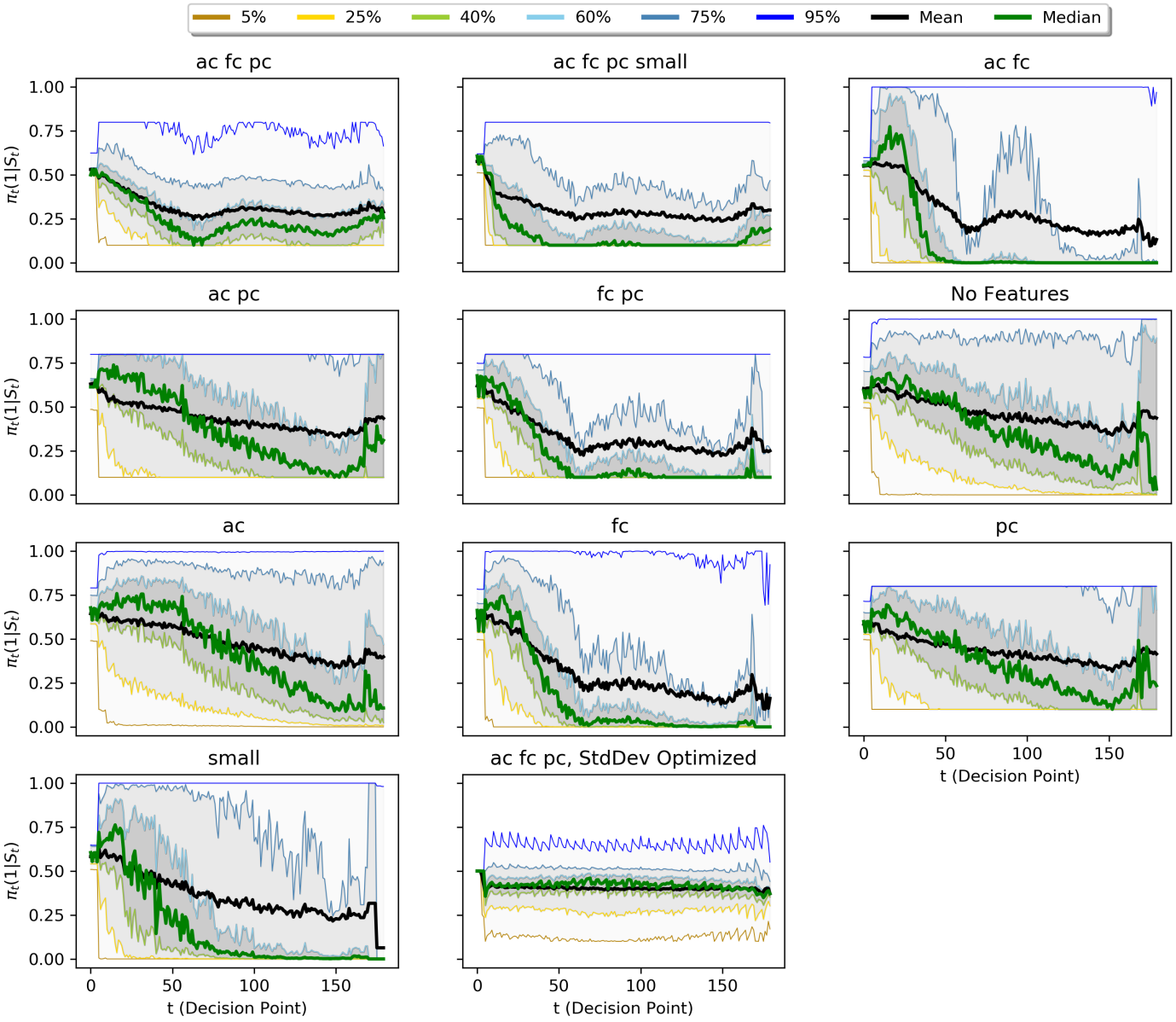


Figure A.6: Action Probability over Users in Testing Batches

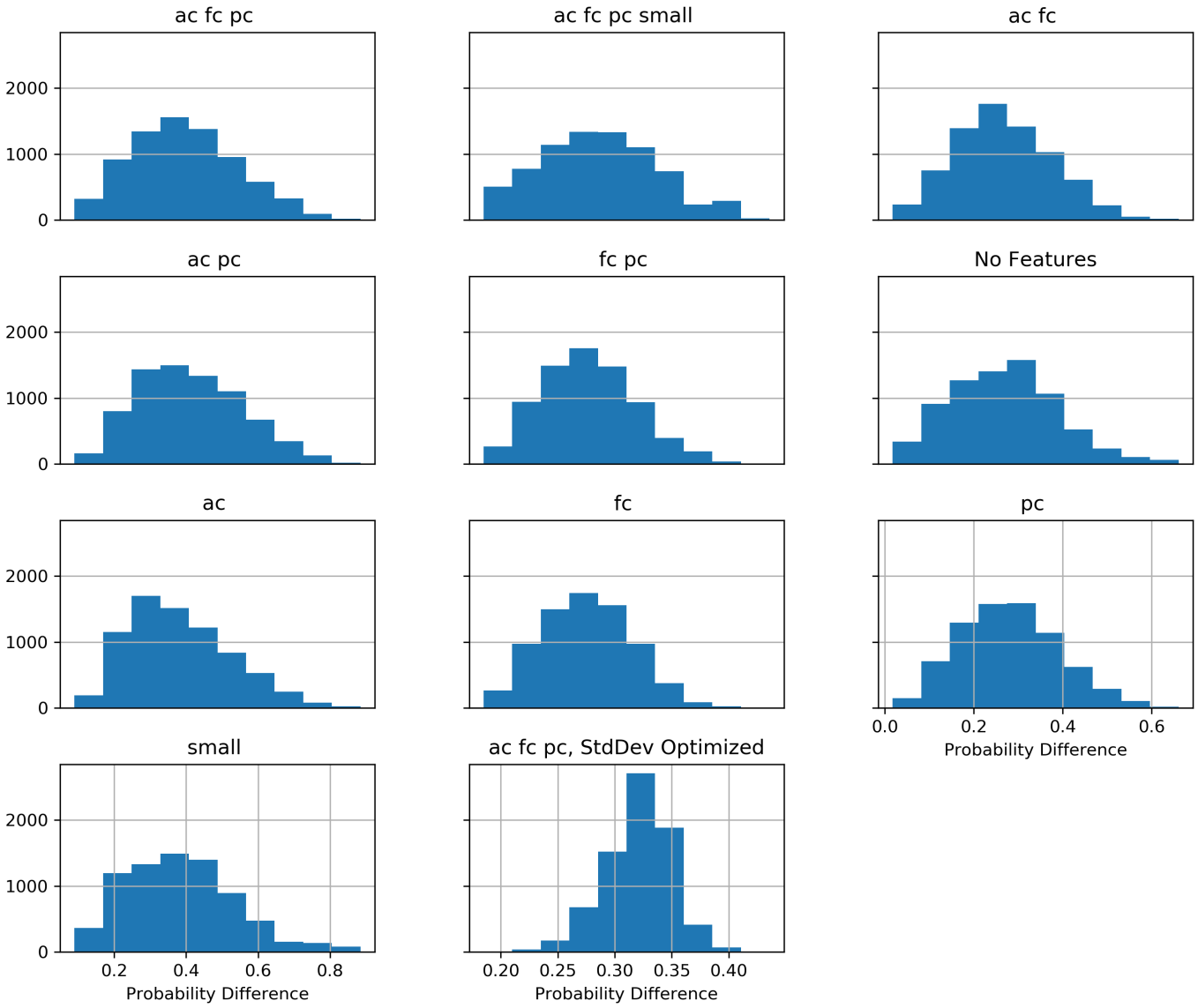


Figure A.7: |Action Probability - Optimal Probability| Averaged over all t for 7500 Users in Optimal Training Batches

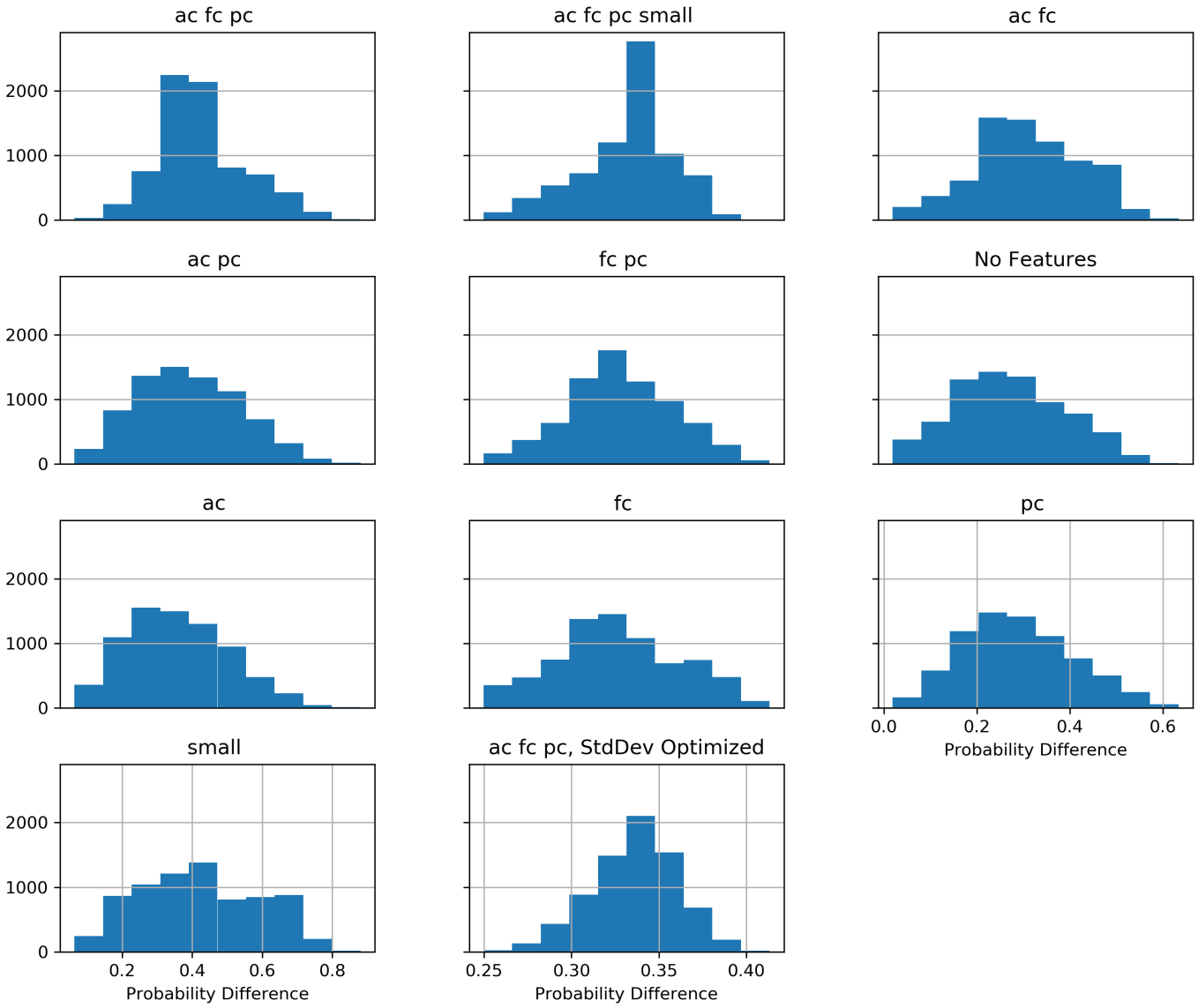


Figure A.8: |Action Probability - Optimal Probability| Averaged over all t for 7500 Users in Testing Batches

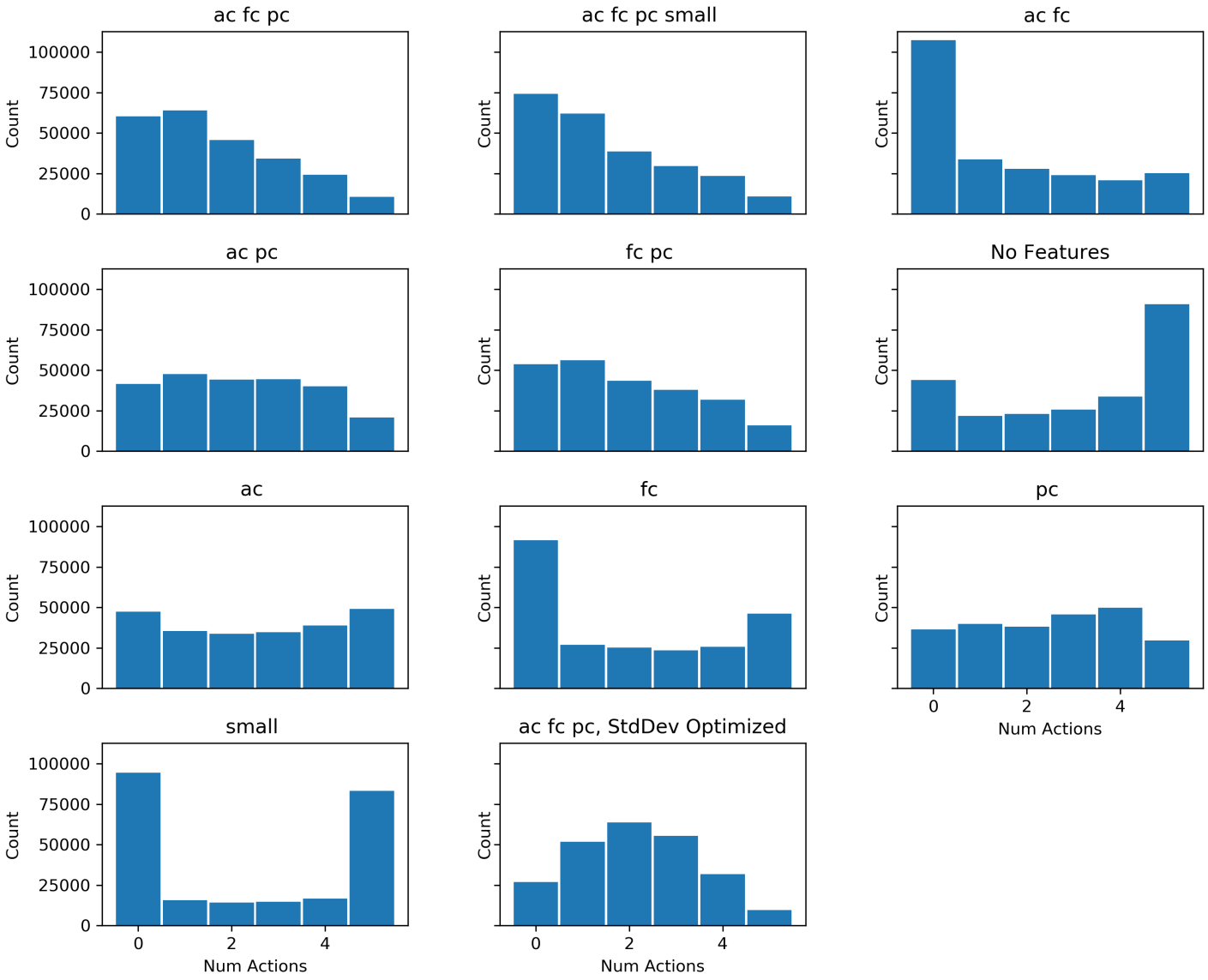


Figure A.9: Number of Actions per Day across all Users in Optimal Training Batches

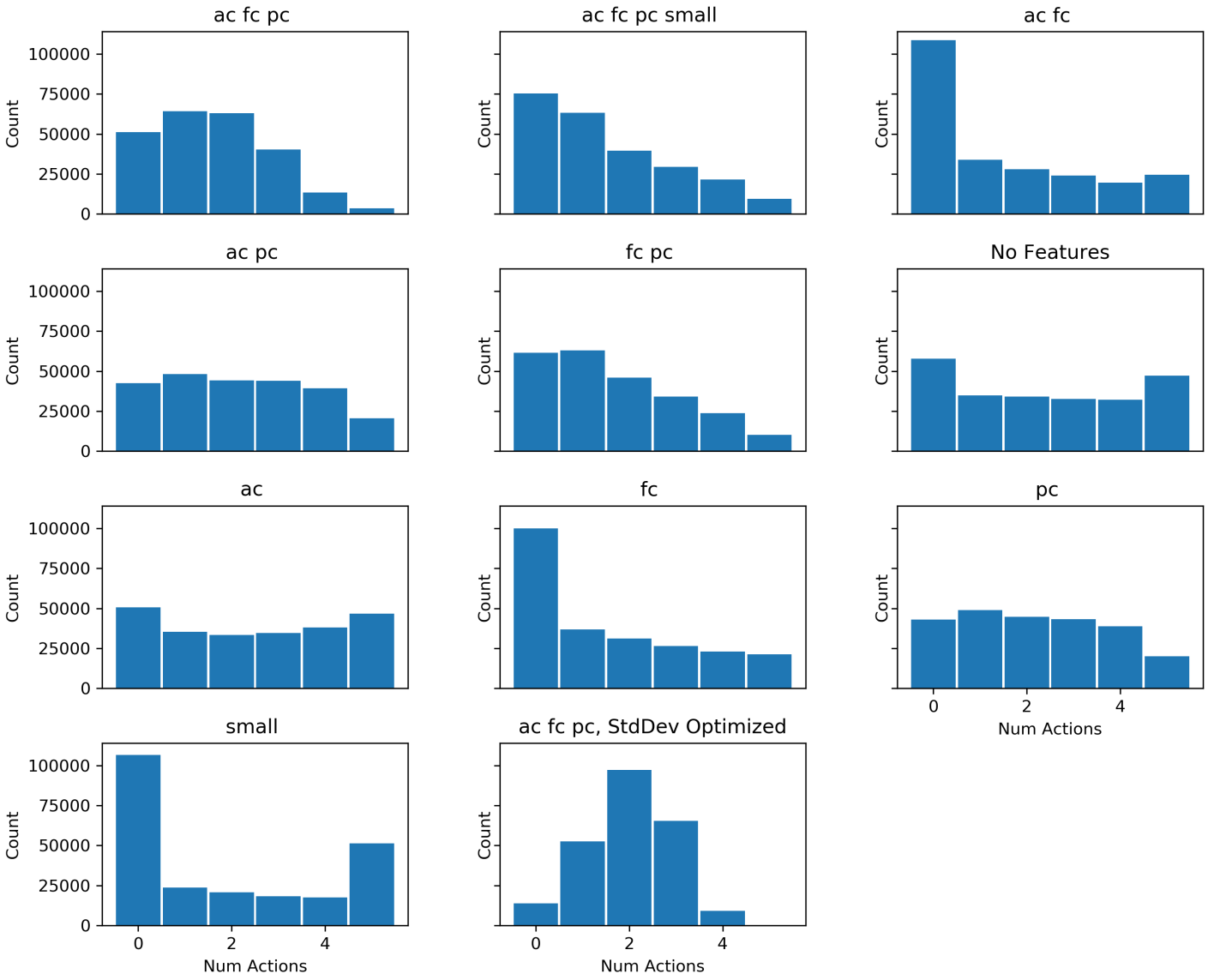


Figure A.10: Number of Actions per Day across all Users in Testing Batches

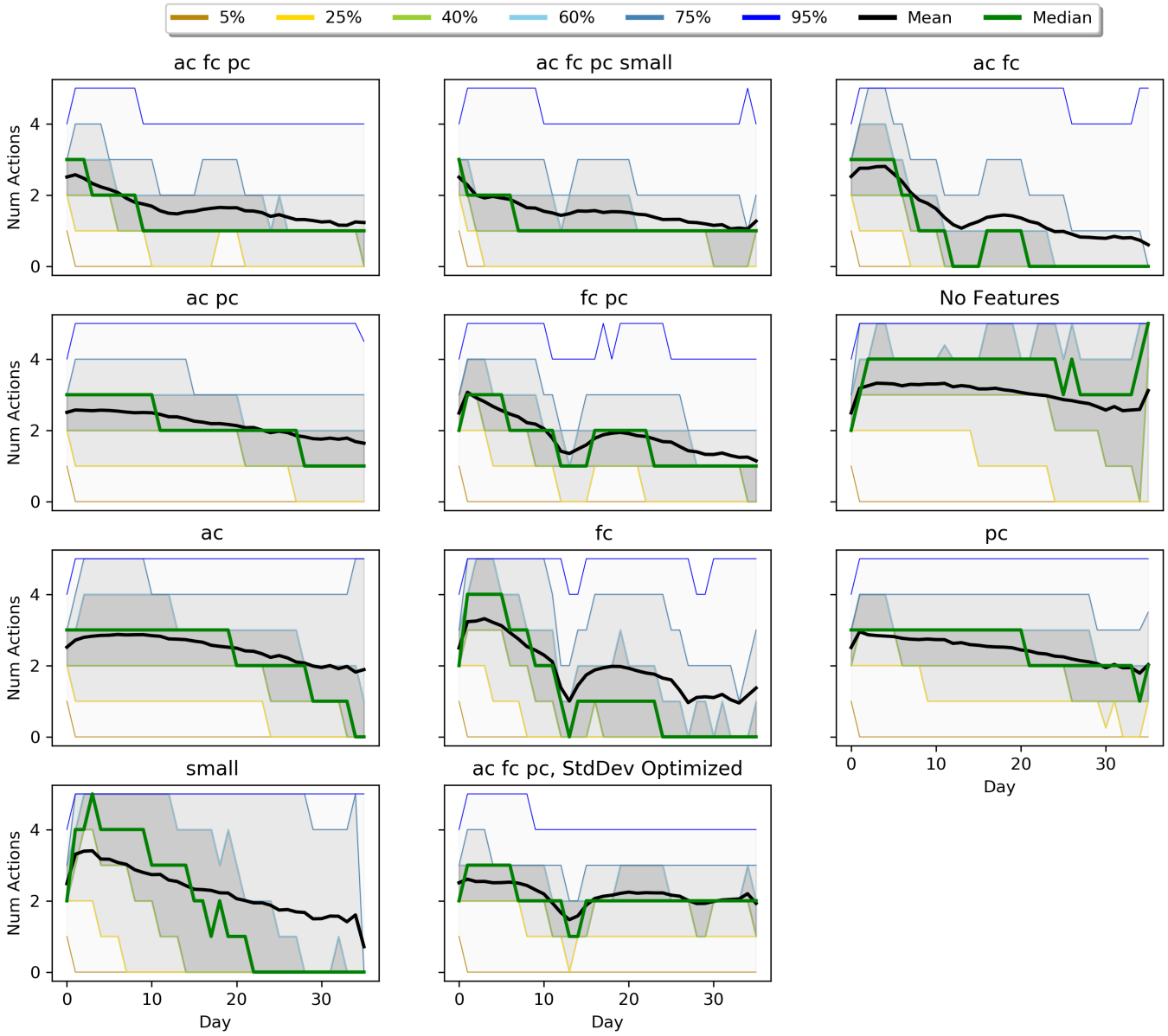


Figure A.11: Number of Actions per Day for Users in Optimal Training Batches

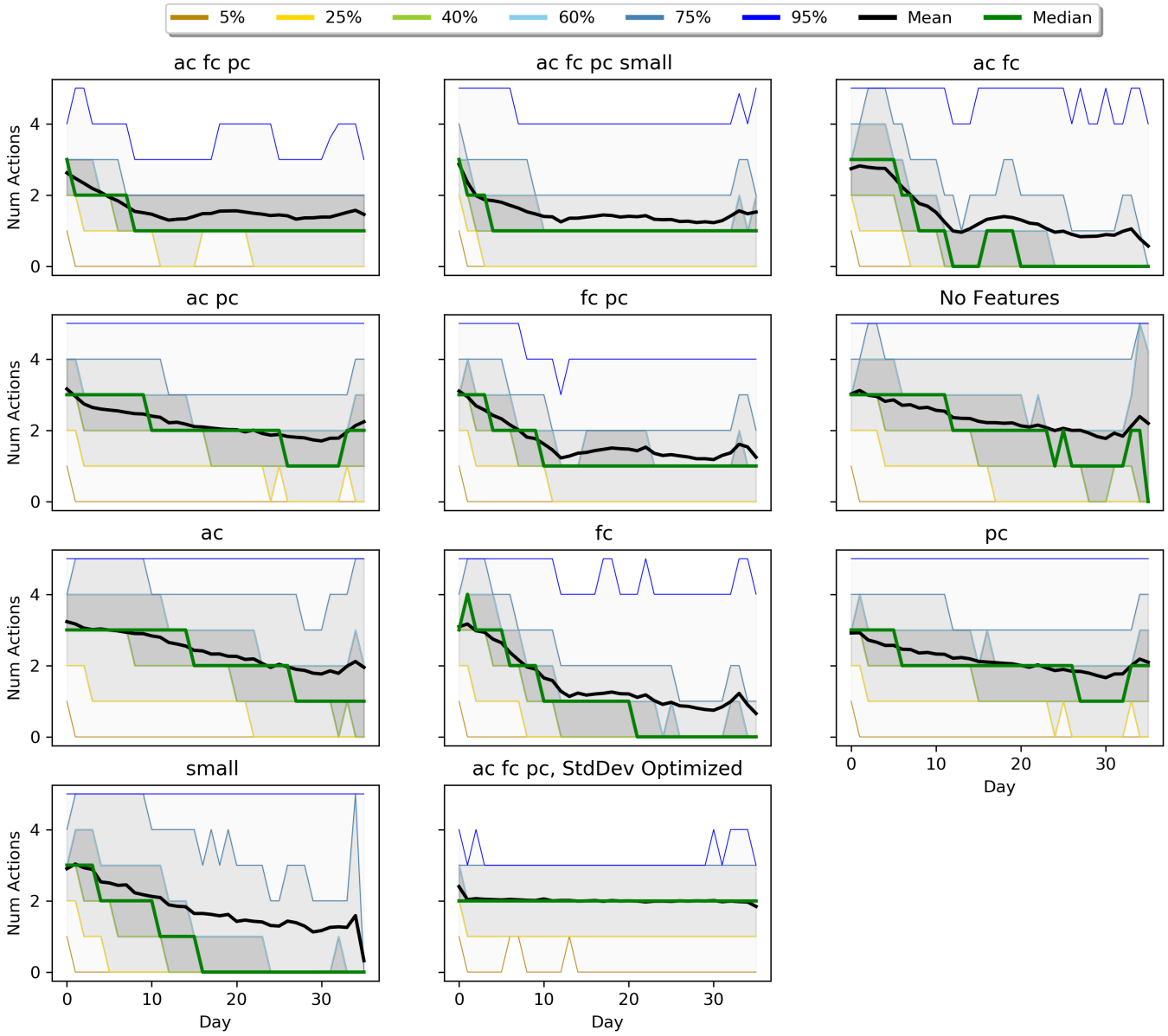


Figure A.12: Number of Actions per Day for Users in Testing Batches

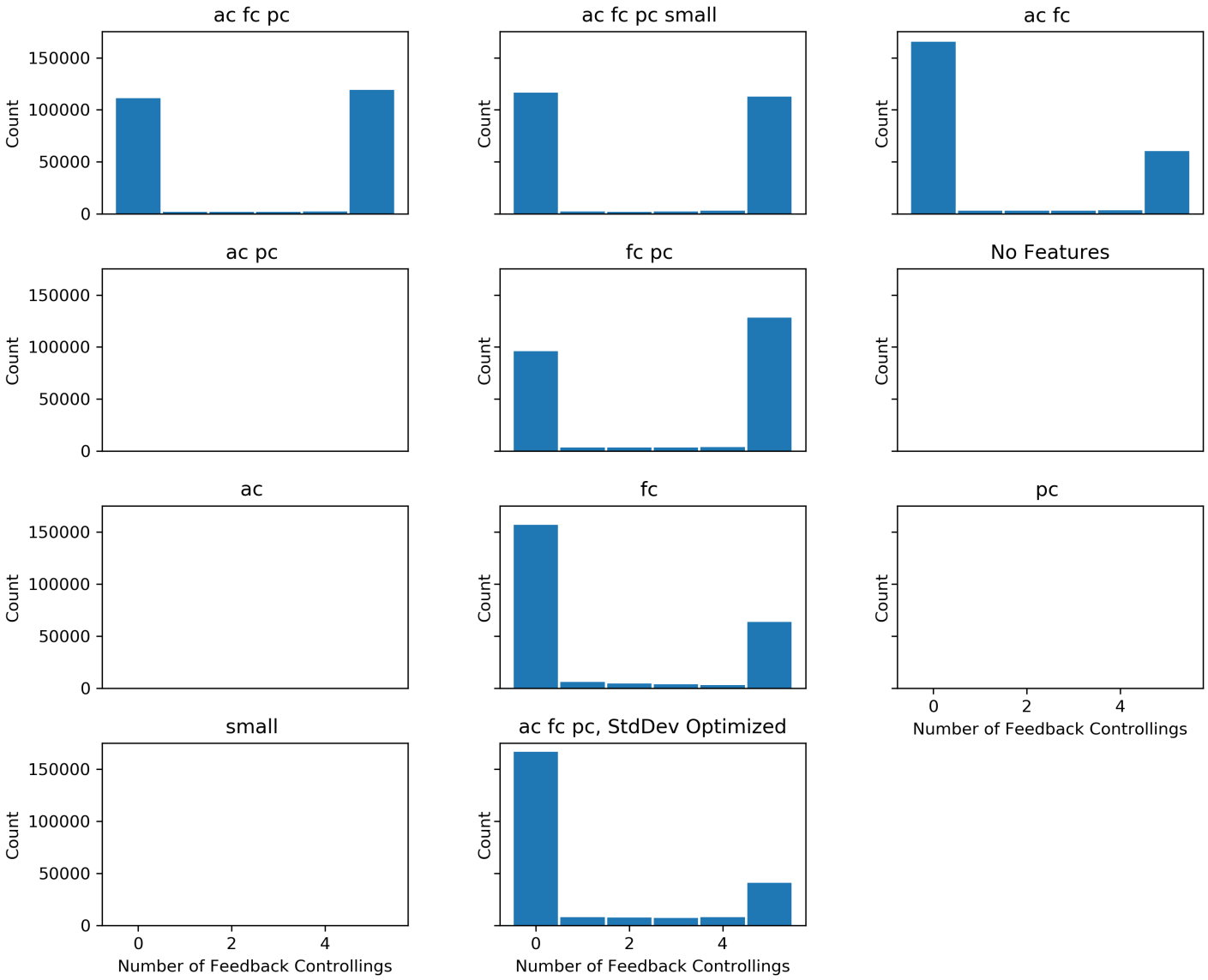


Figure A.13: Number of Feedback Controllings per Day across all Users in Optimal Training Batches

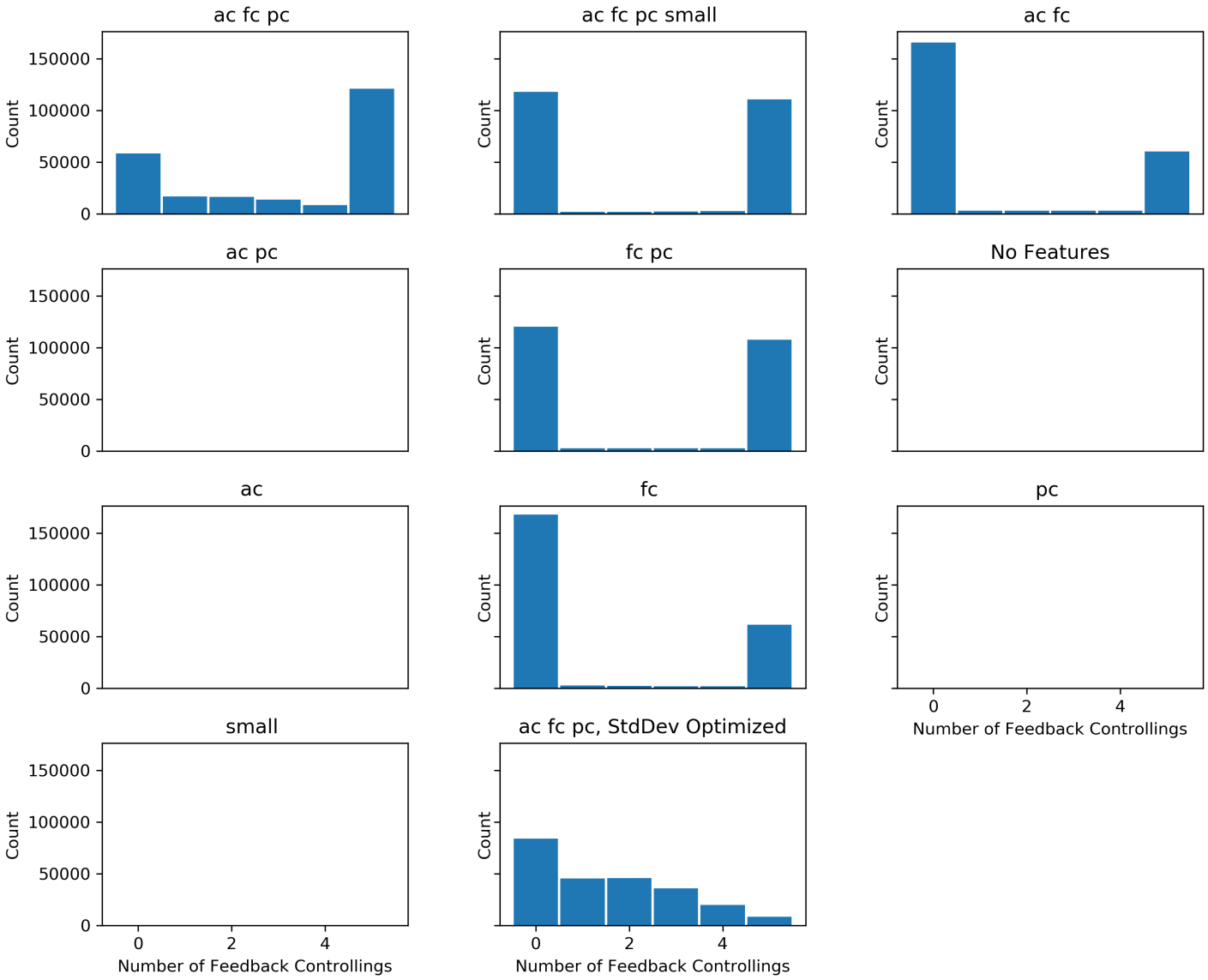


Figure A.14: Number of Feedback Controllings per Day across all Users in Testing Batches

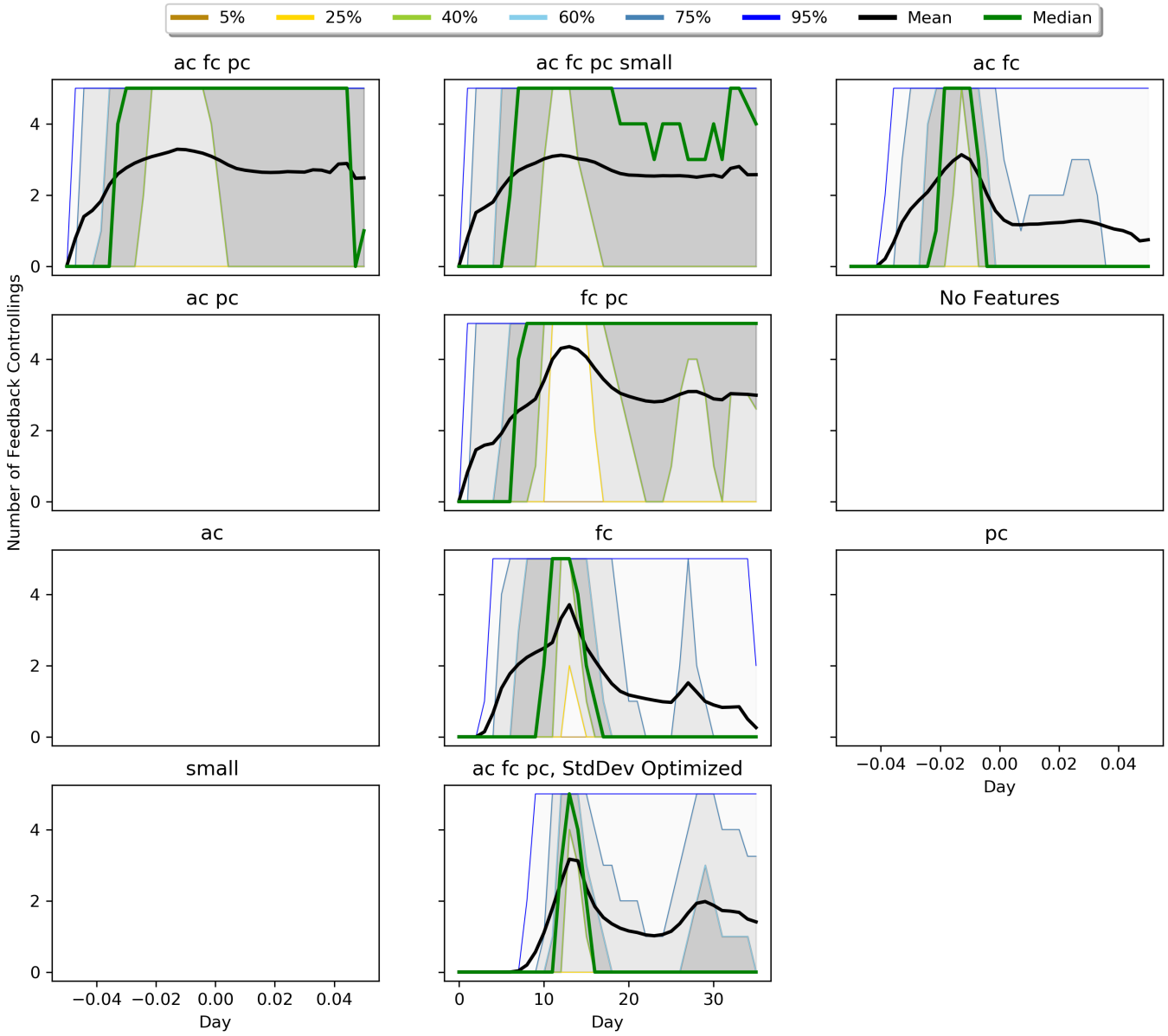


Figure A.15: Number of Feedback Controllings per Day for Users in Optimal Training Batches

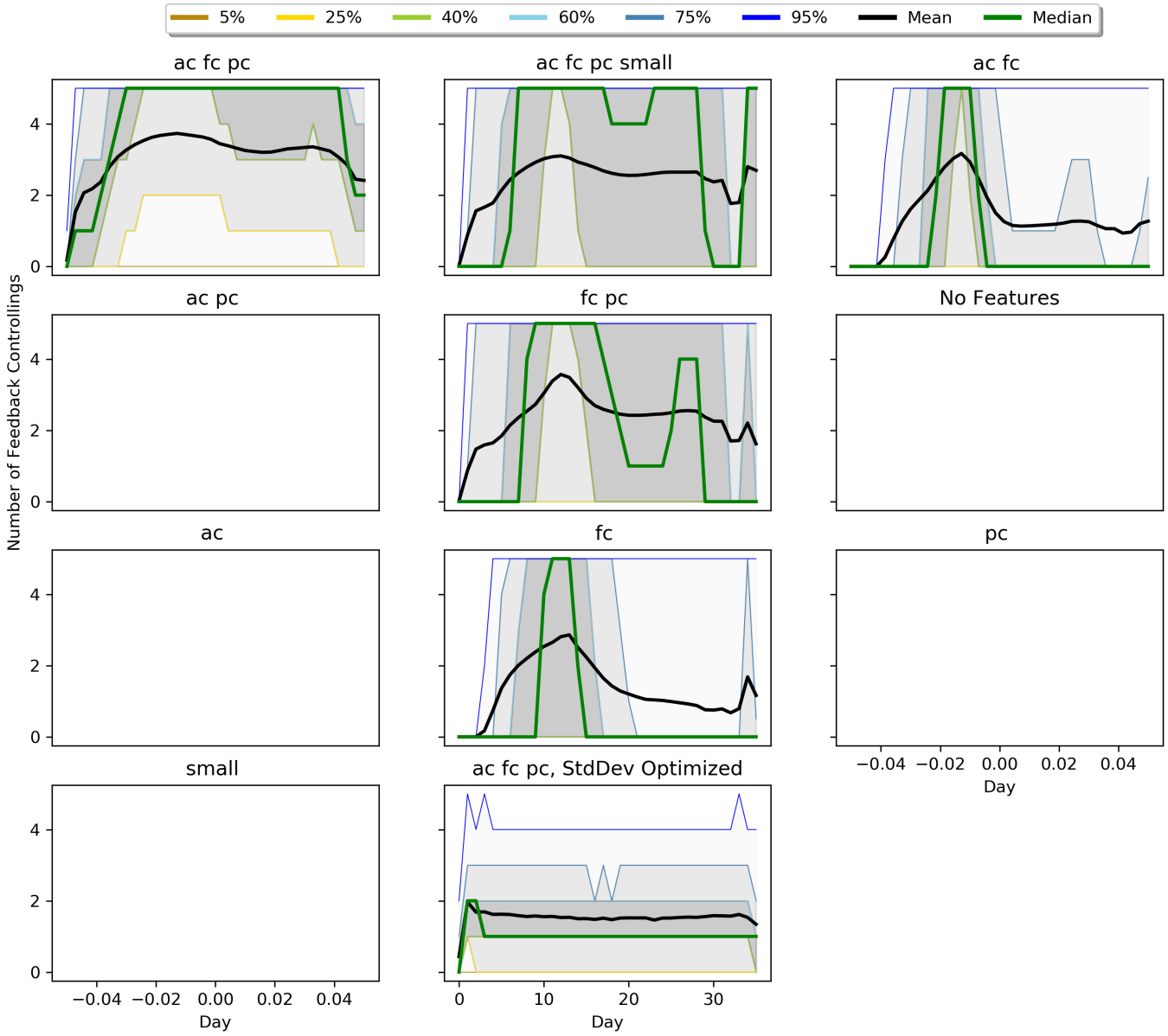


Figure A.16: Number of Feedback Controllings per Day for Users in Testing Batches

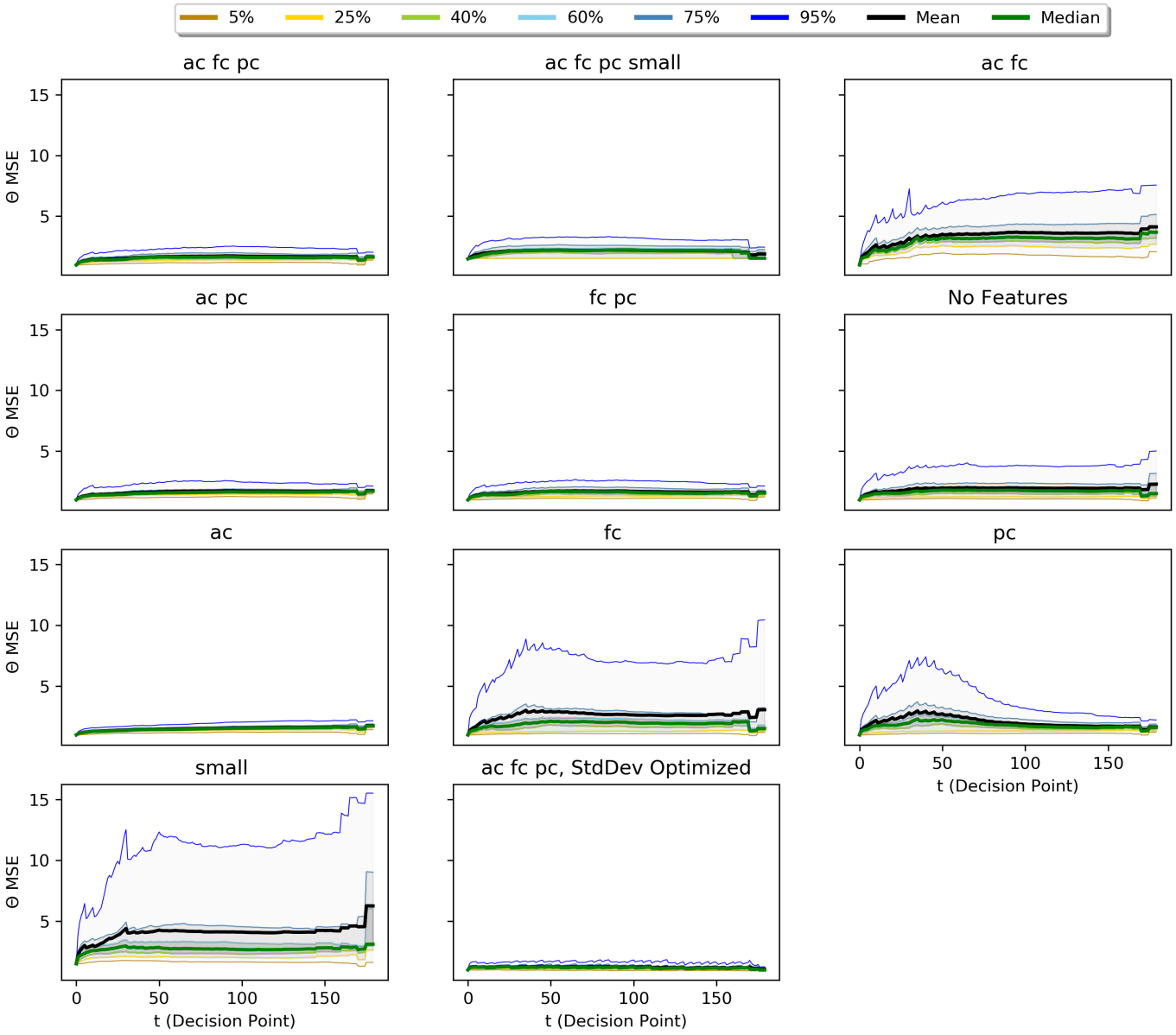


Figure A.17: Bandit Model Θ MSE for All Users in Optimal Training Batches

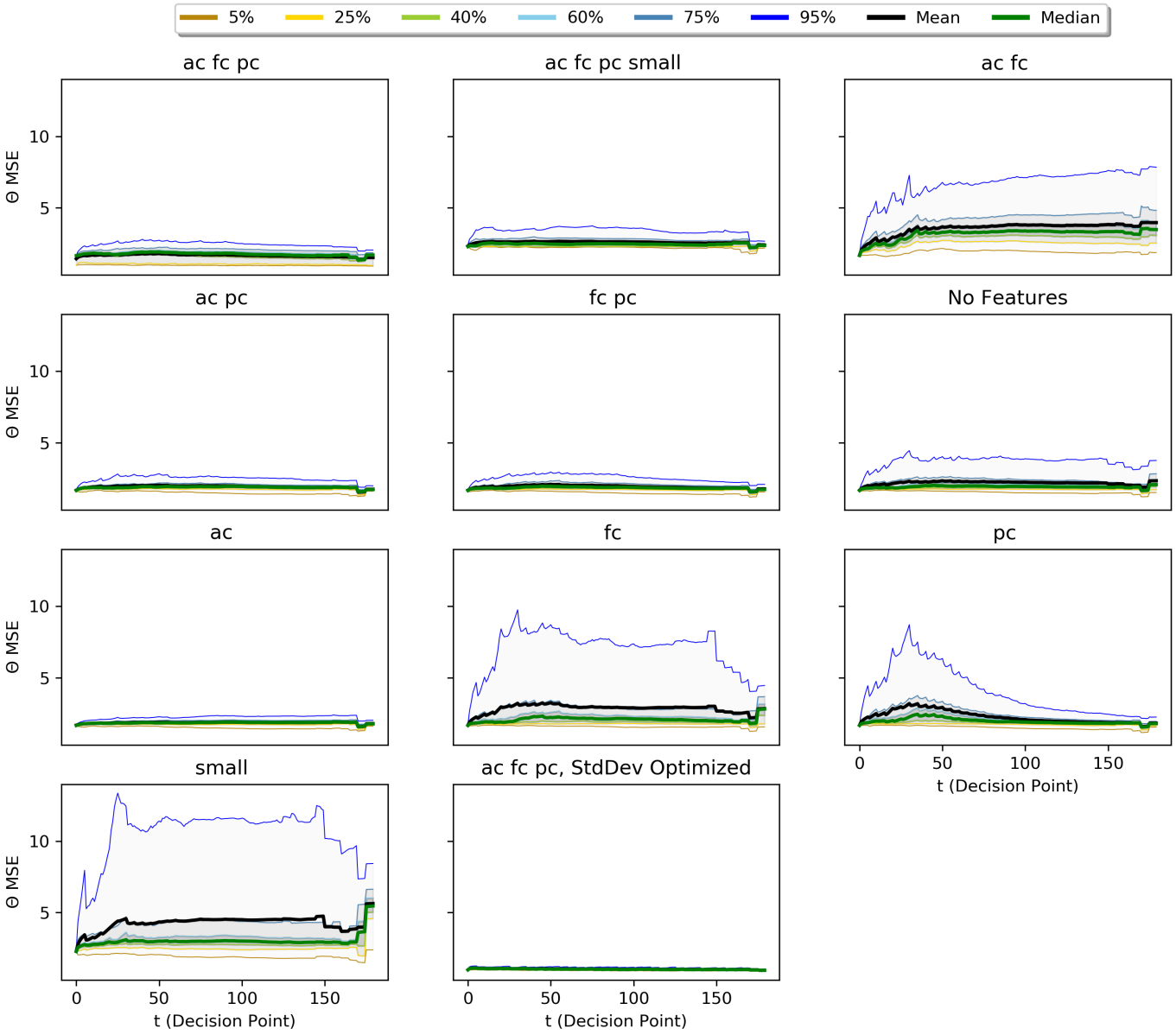


Figure A.18: Bandit Model Θ MSE for All Users in Testing Batches

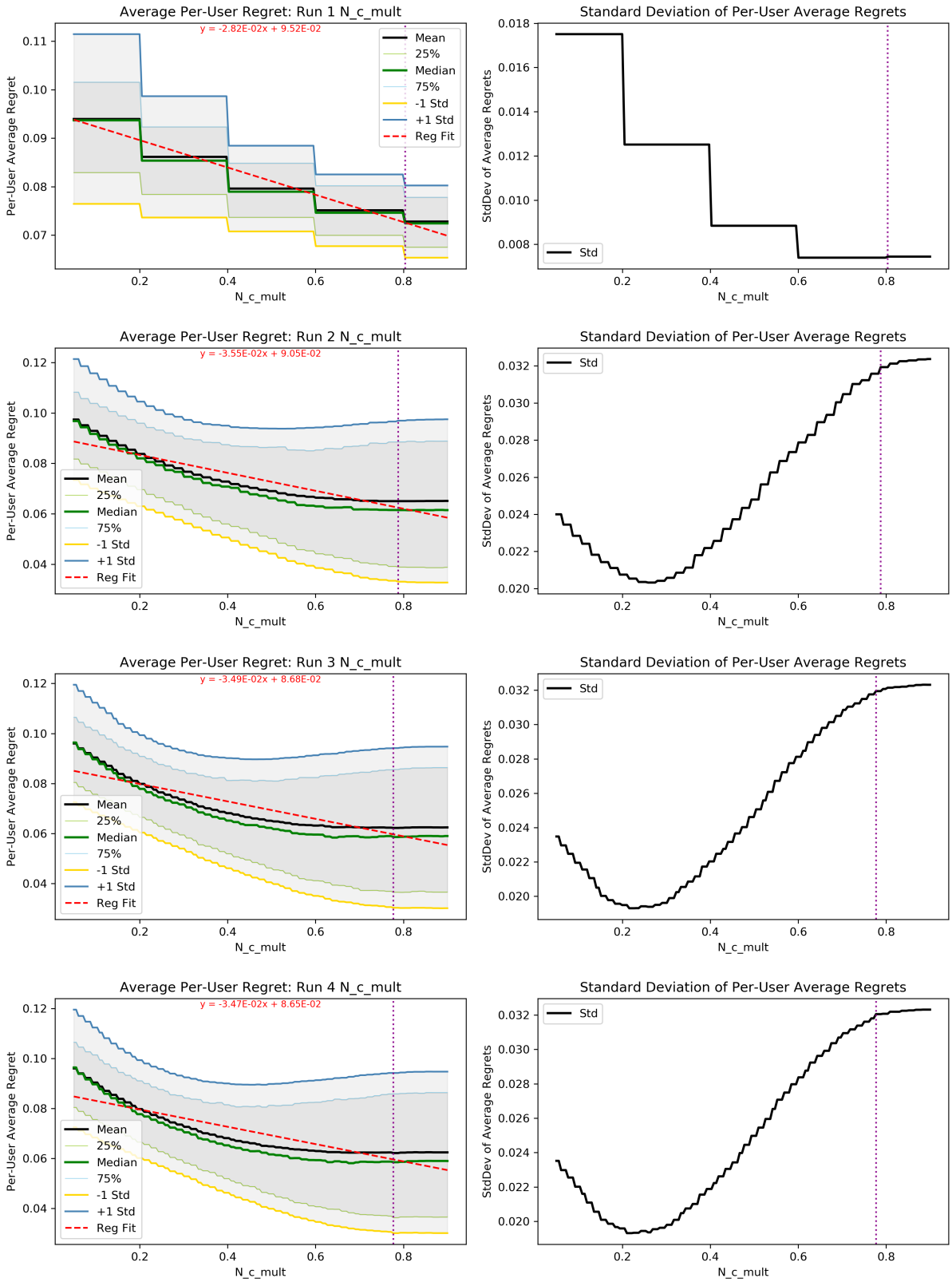


Figure A.19: $MUER$ for Varying `N_c_mult`, $MUER$ Minimization Optimization

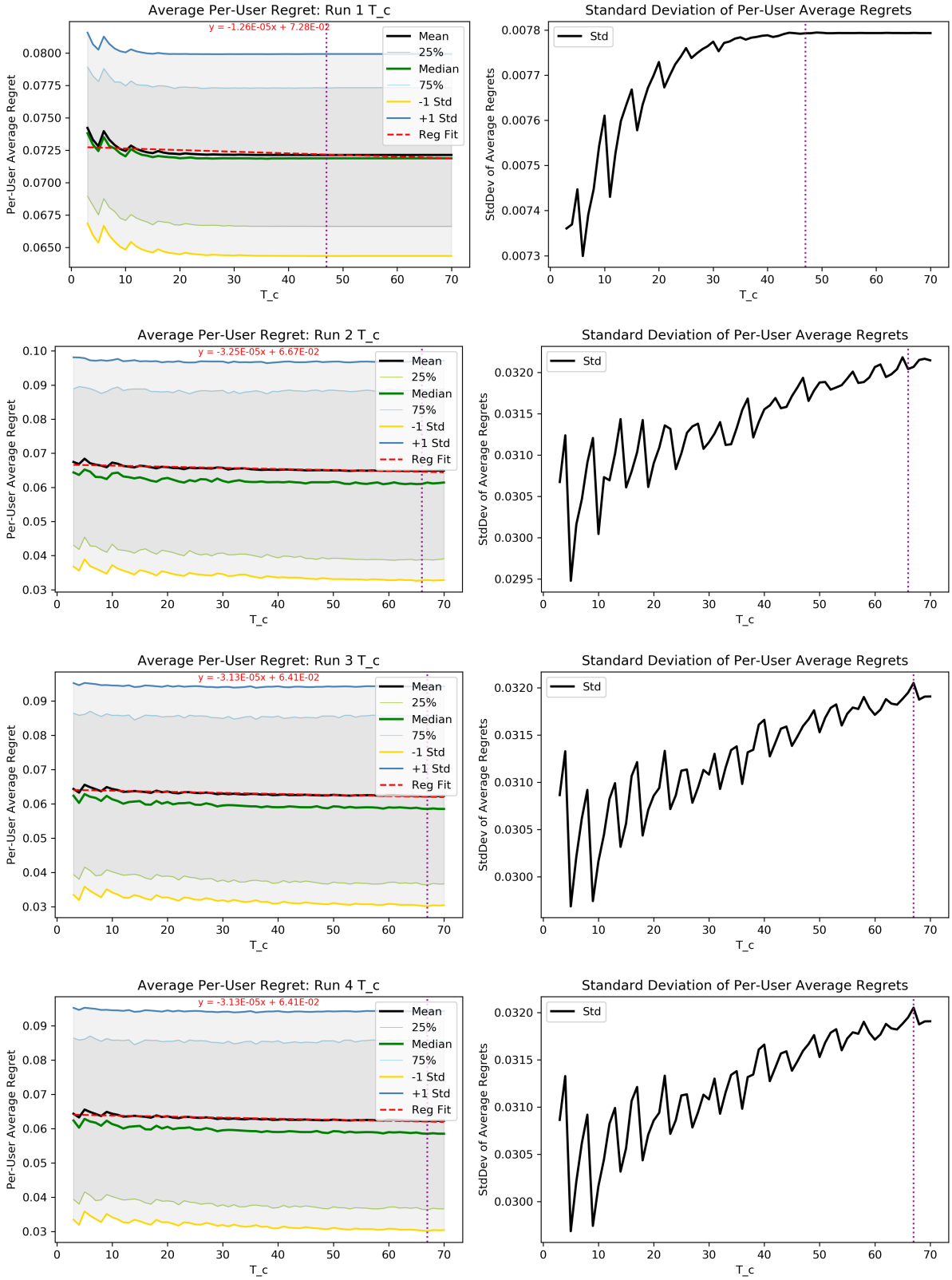


Figure A.20: $MUER$ for Varying T_c, $MUER$ Minimization Optimization

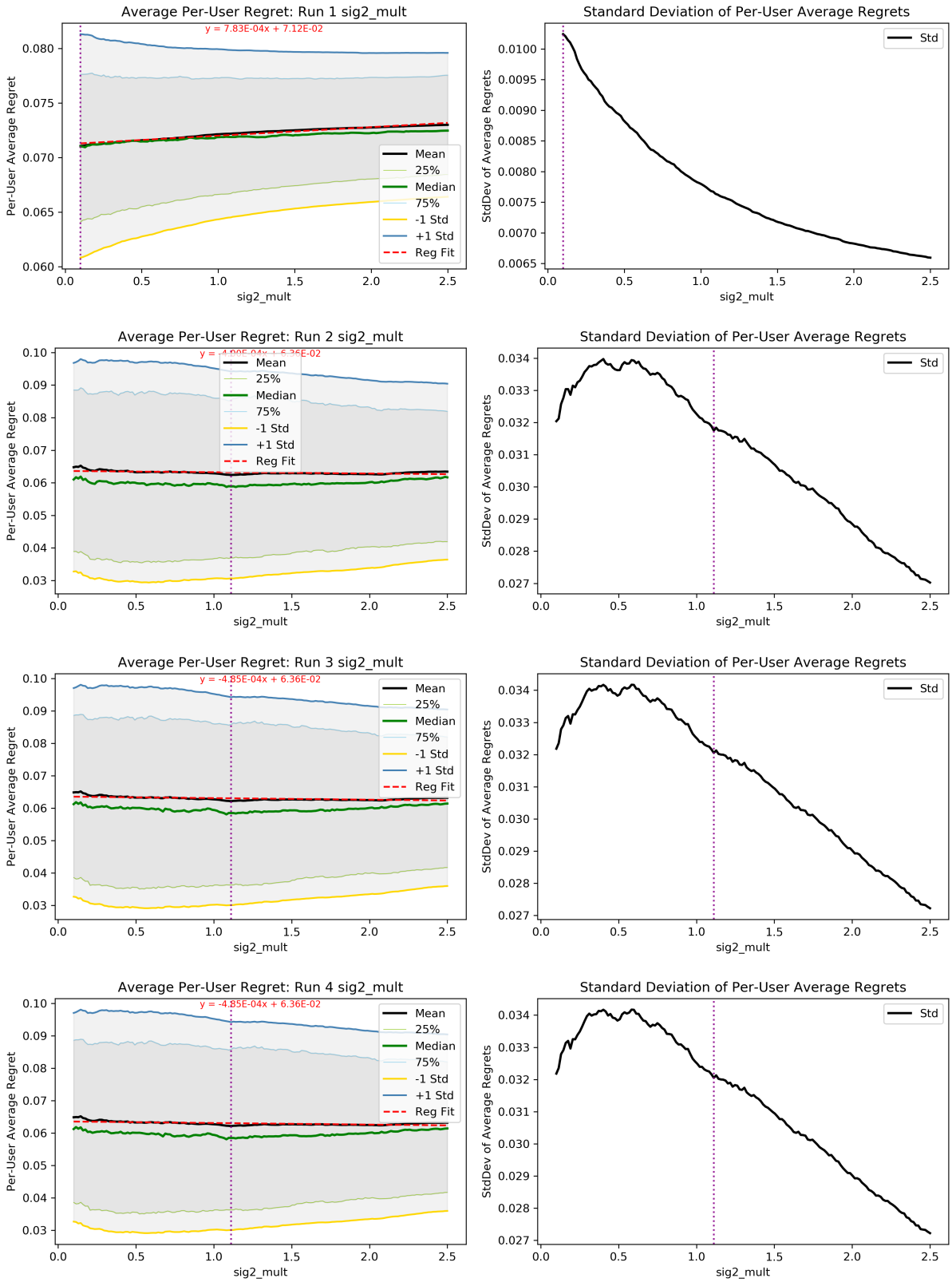


Figure A.21: $MUER$ for Varying `sig2_mult`, $MUER$ Minimization Optimization

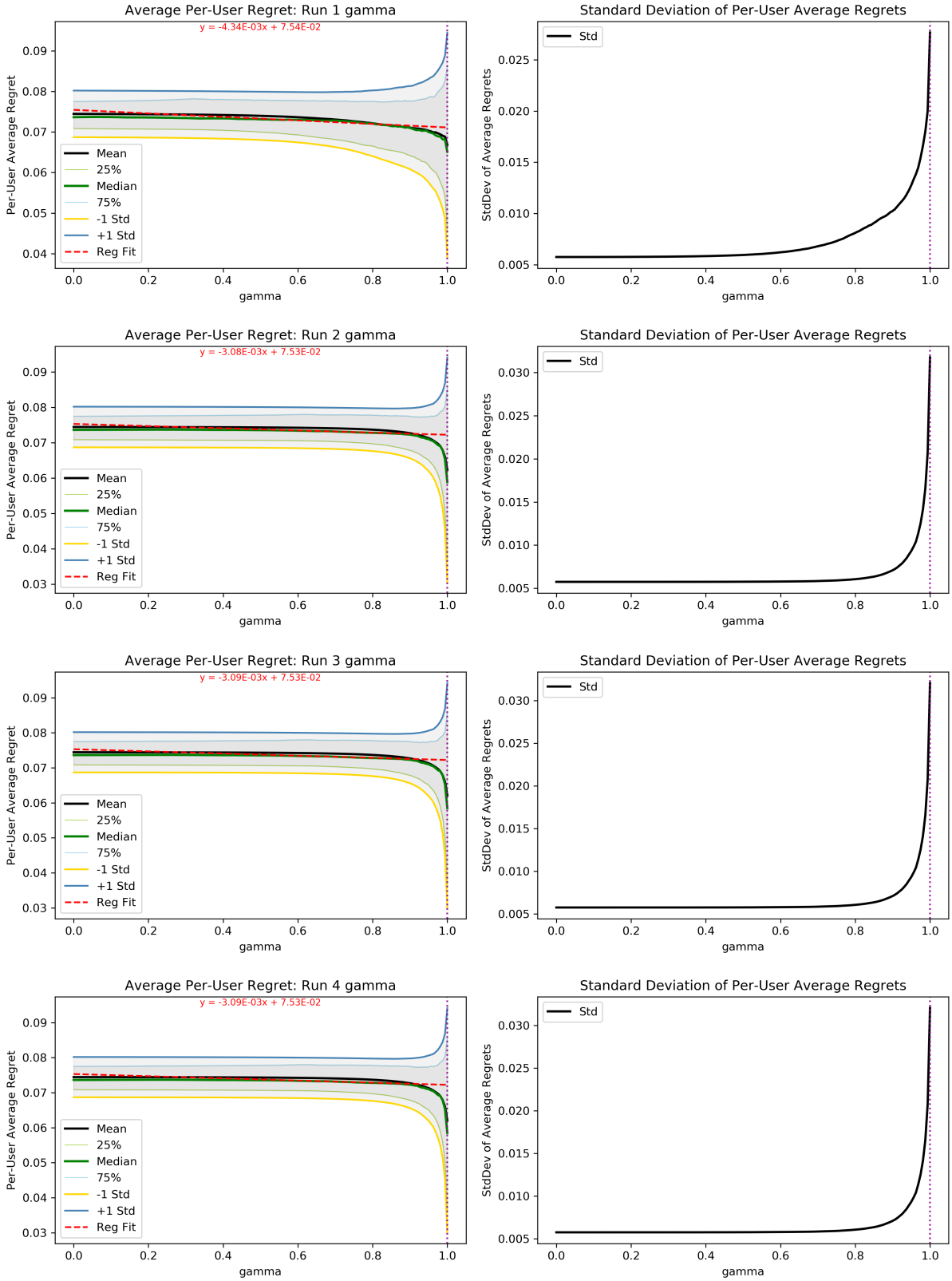


Figure A.22: $MUER$ for Varying `gamma`, $MUER$ Minimization Optimization

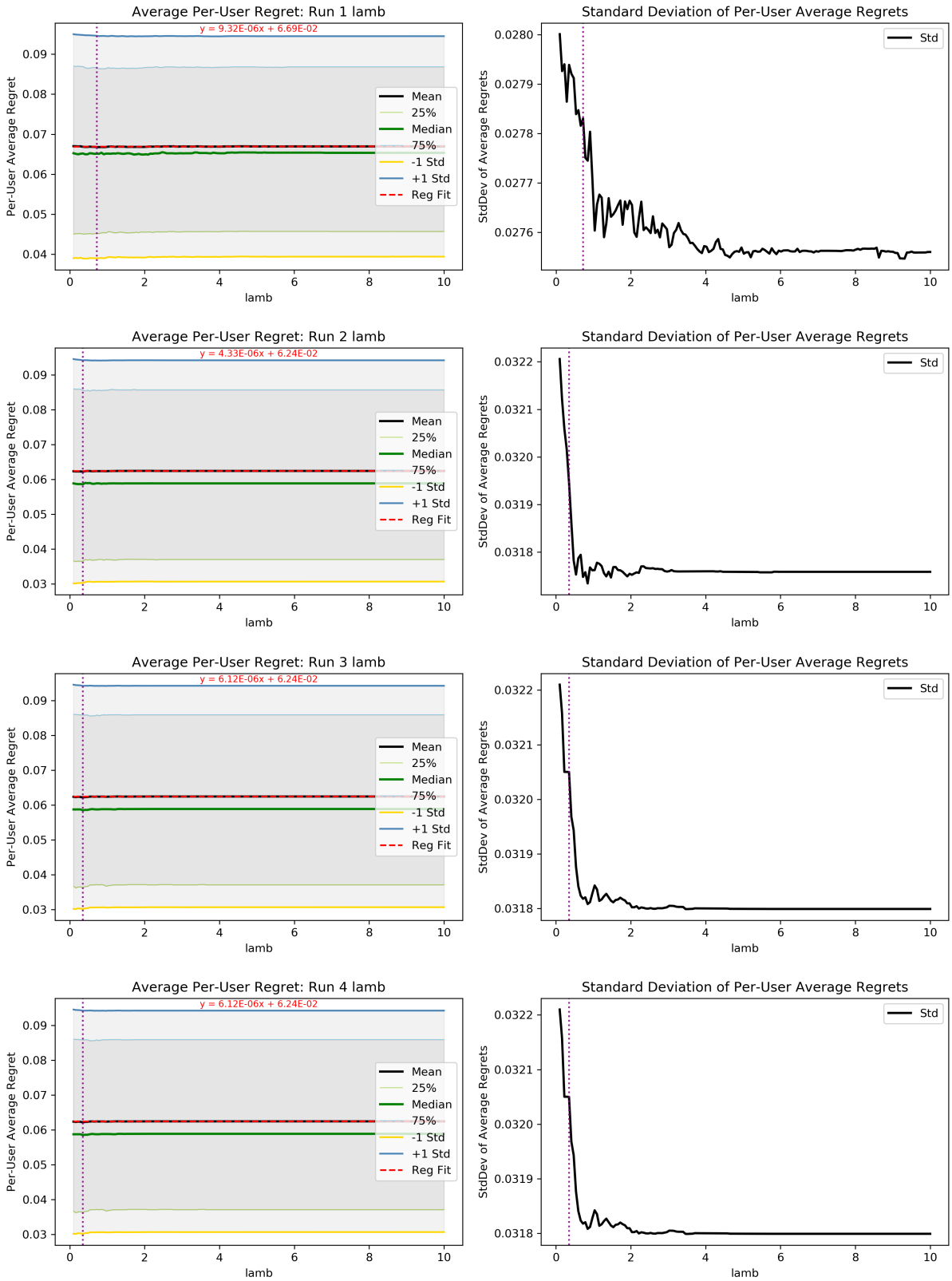


Figure A.23: $MUER$ for Varying `lamb`, $MUER$ Minimization Optimization

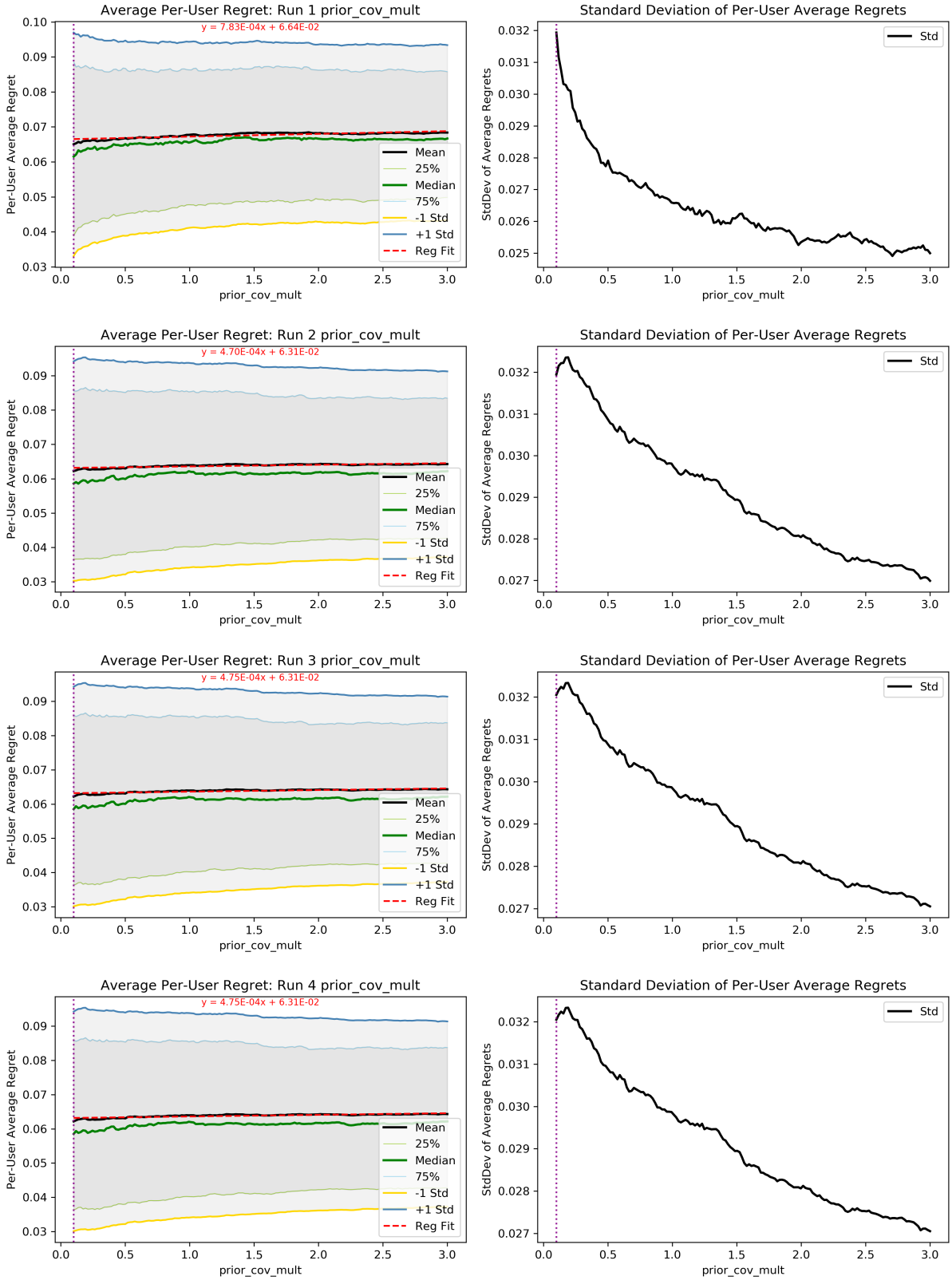


Figure A.24: $MUER$ for Varying `prior_cov_mult`, $MUER$ Minimization Optimization

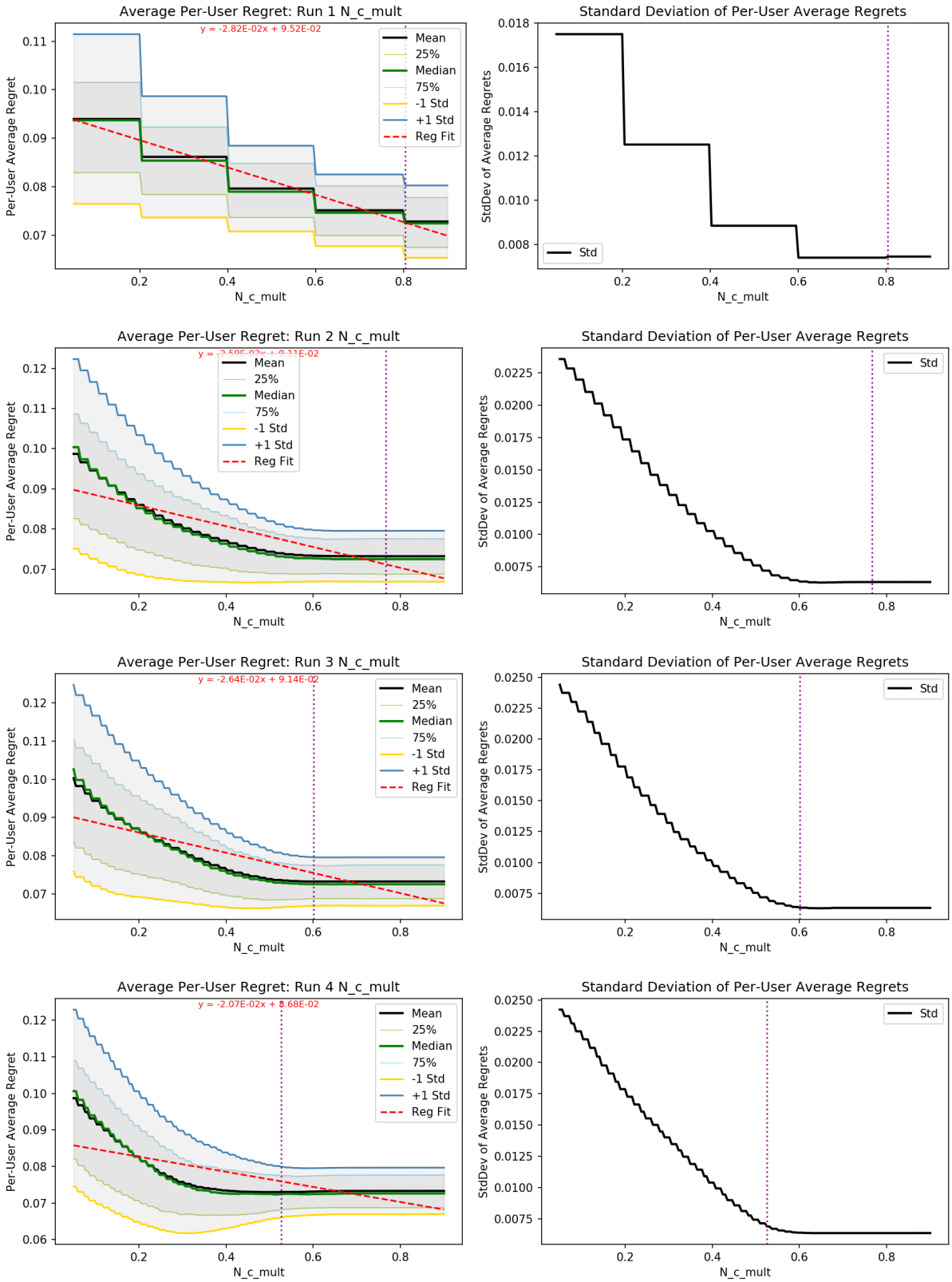


Figure A.25: $MUER$ for Varying `N_c_mult`, StdDev Cutoff Optimization

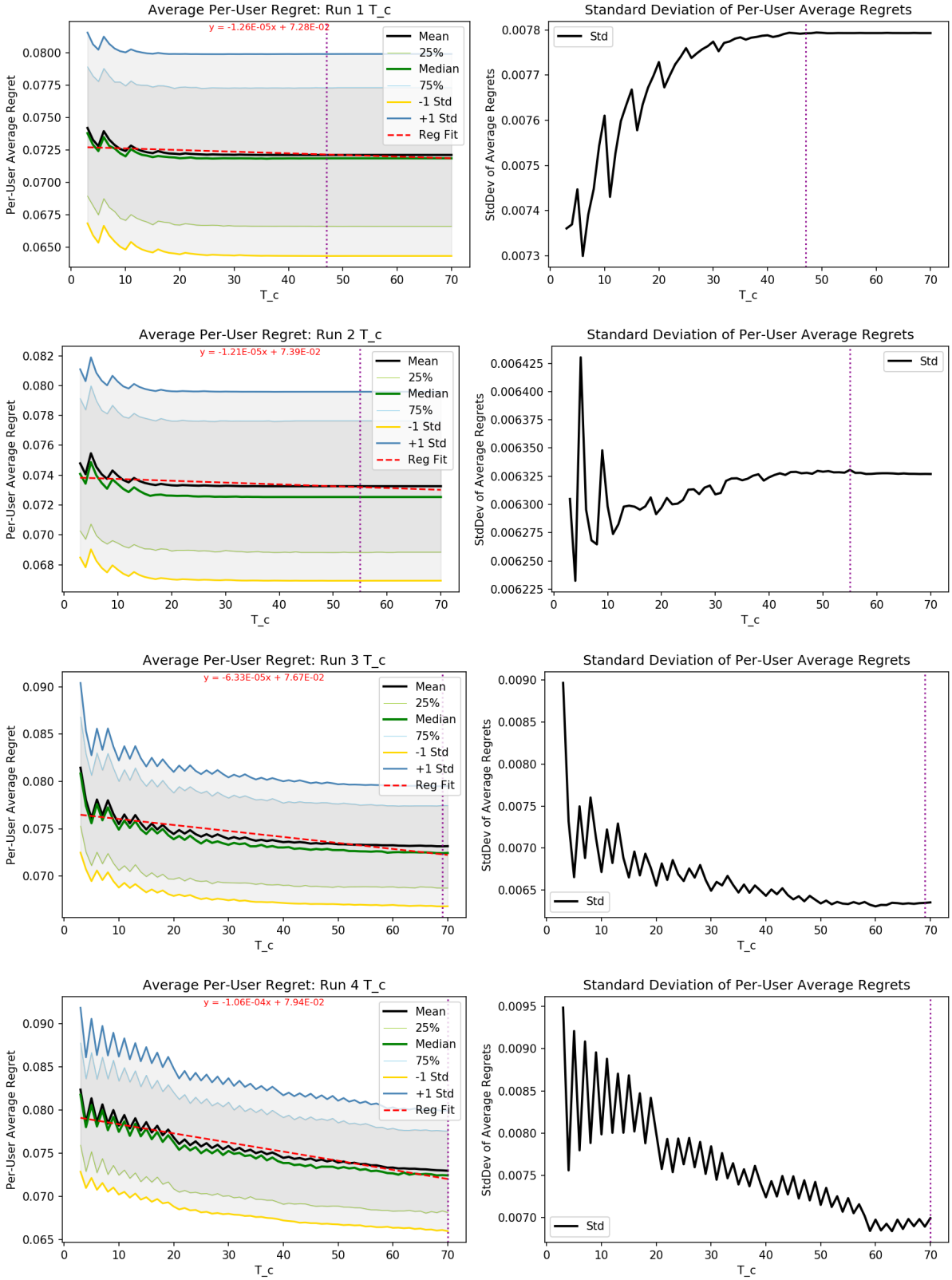


Figure A.26: $MUER$ for Varying T_c, StdDev Cutoff Optimization

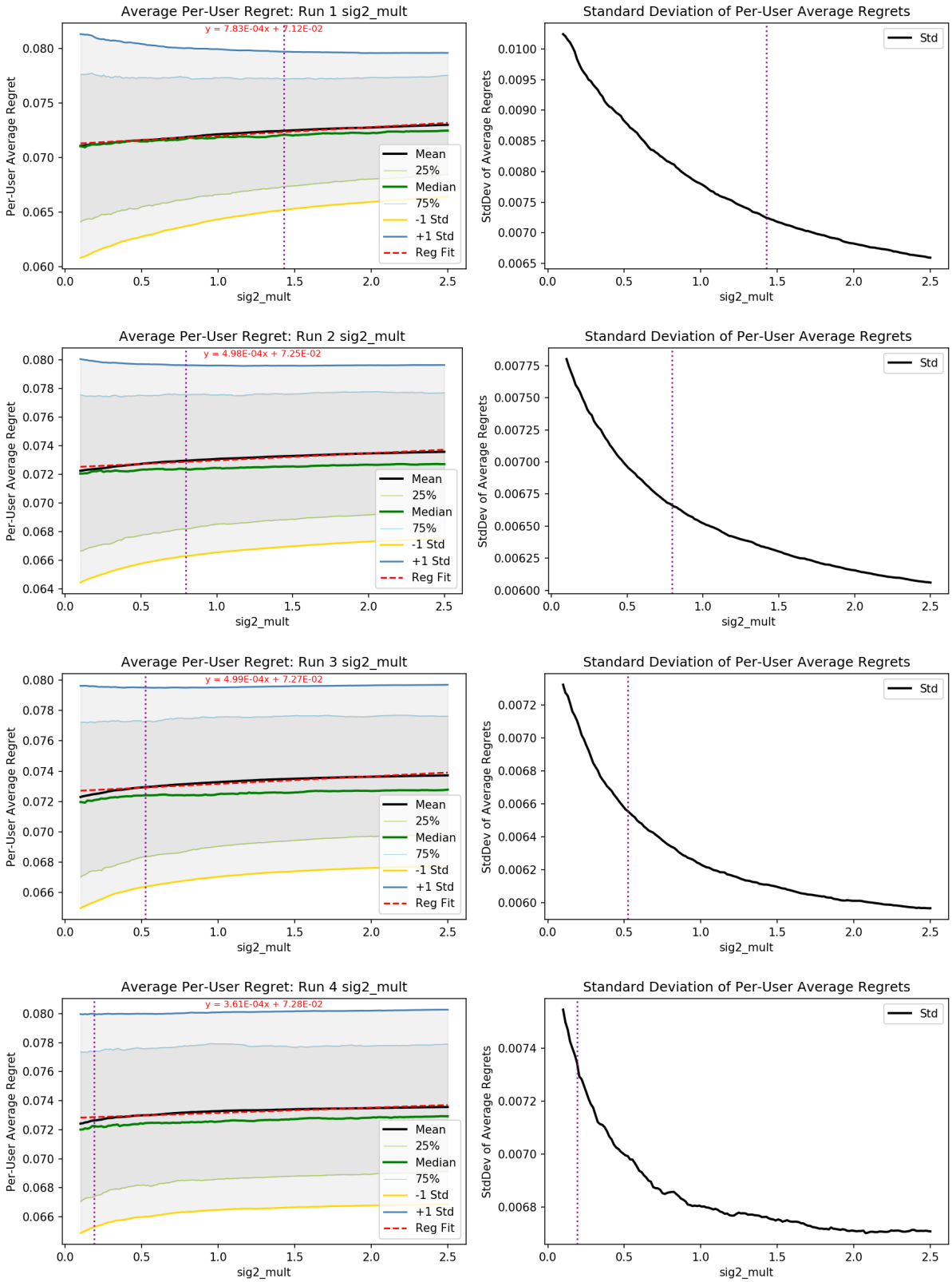


Figure A.27: $MUER$ for Varying `sig2_mult`, StdDev Cutoff Optimization

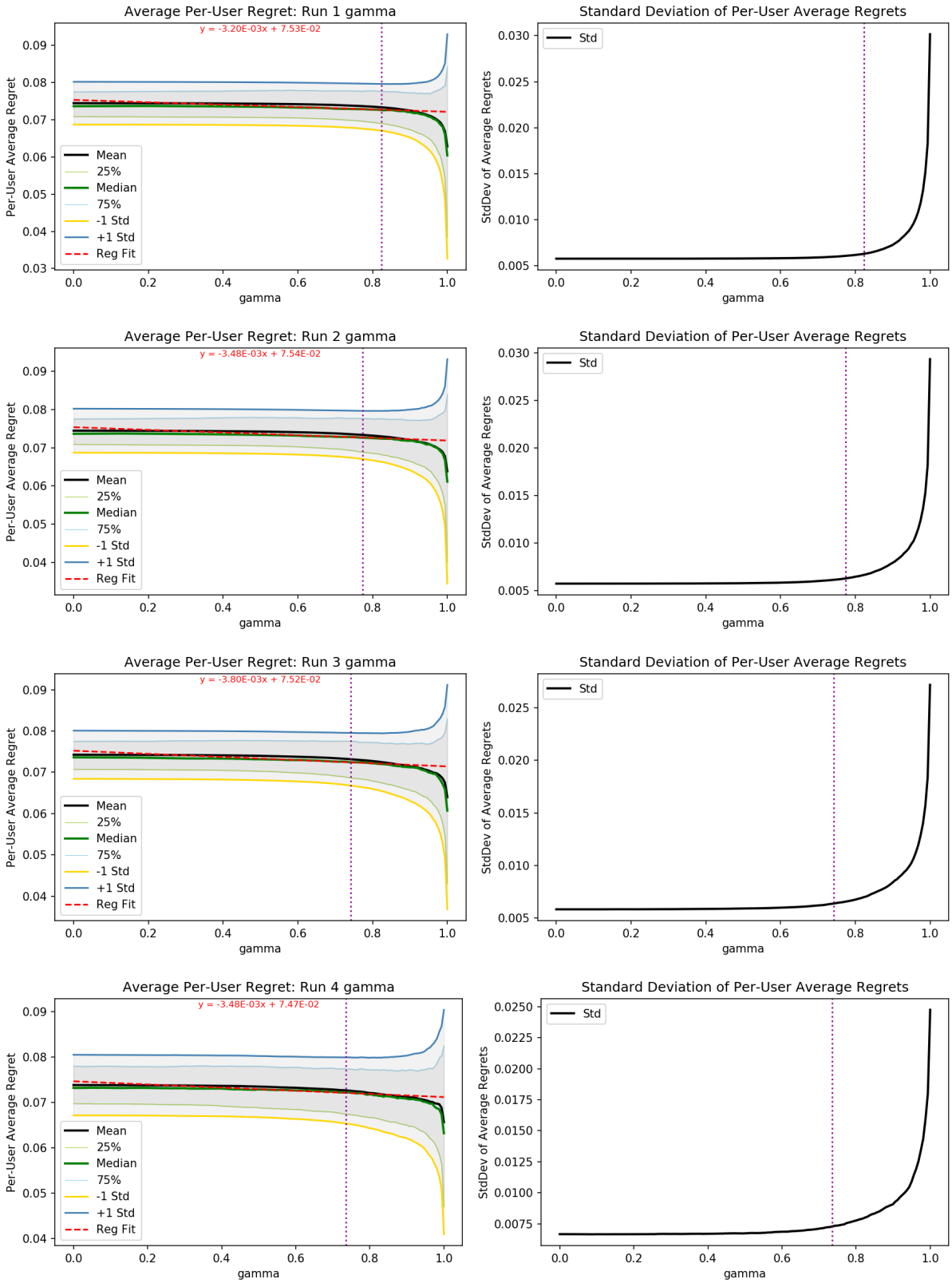


Figure A.28: $MUER$ for Varying `gamma`, StdDev Cutoff Optimization

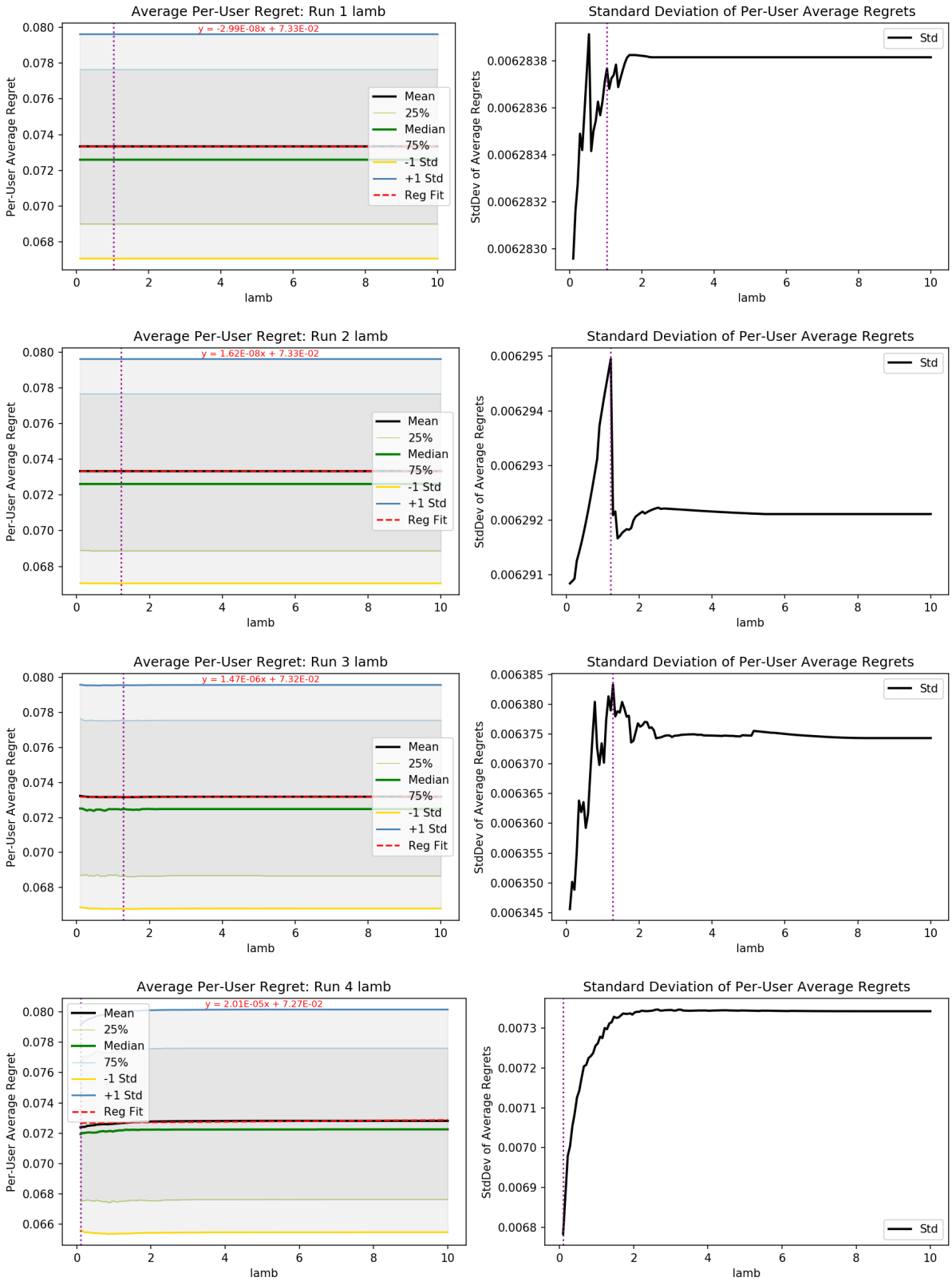


Figure A.29: $MUER$ for Varying `lamb`, StdDev Cutoff Optimization

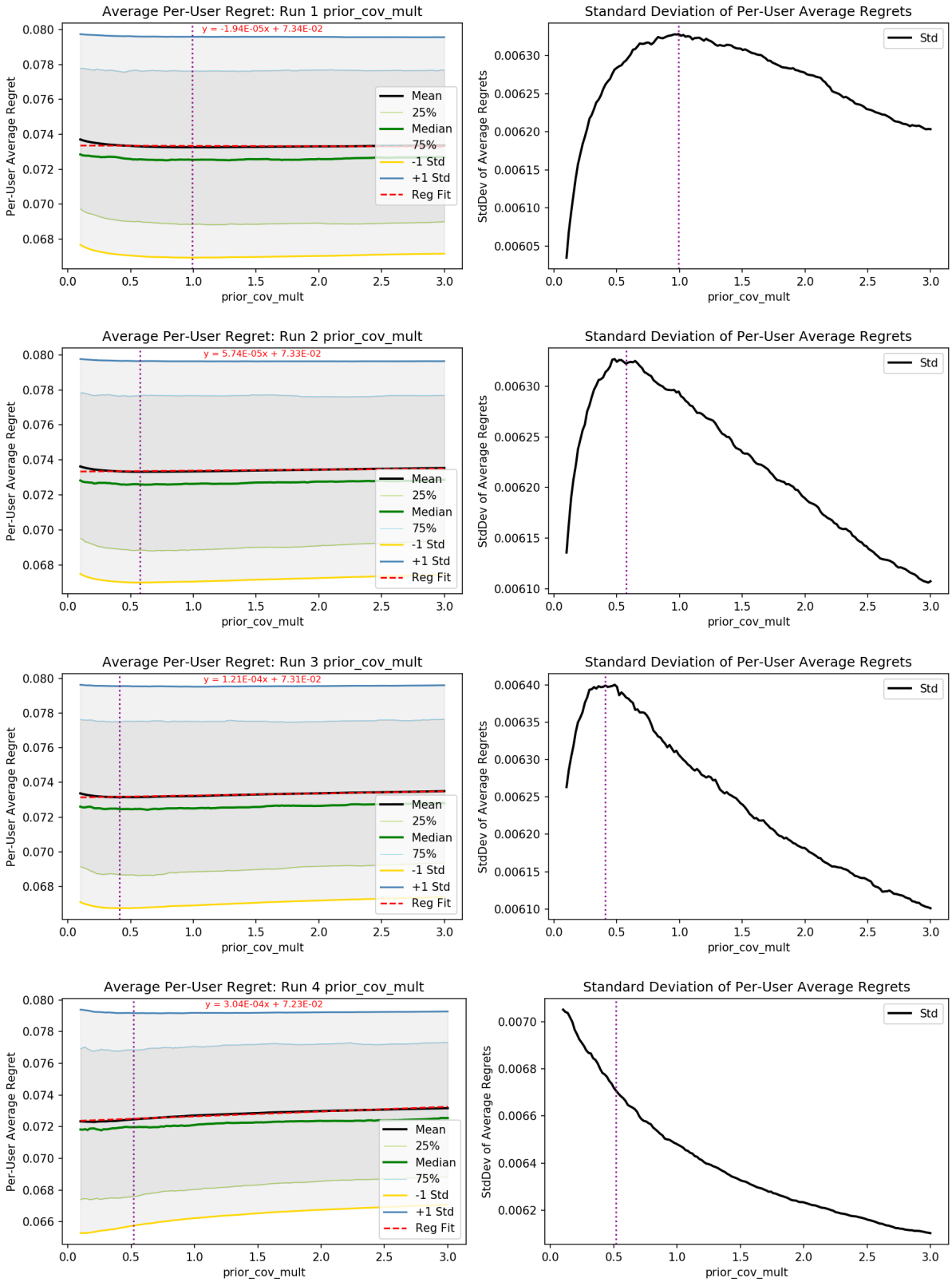


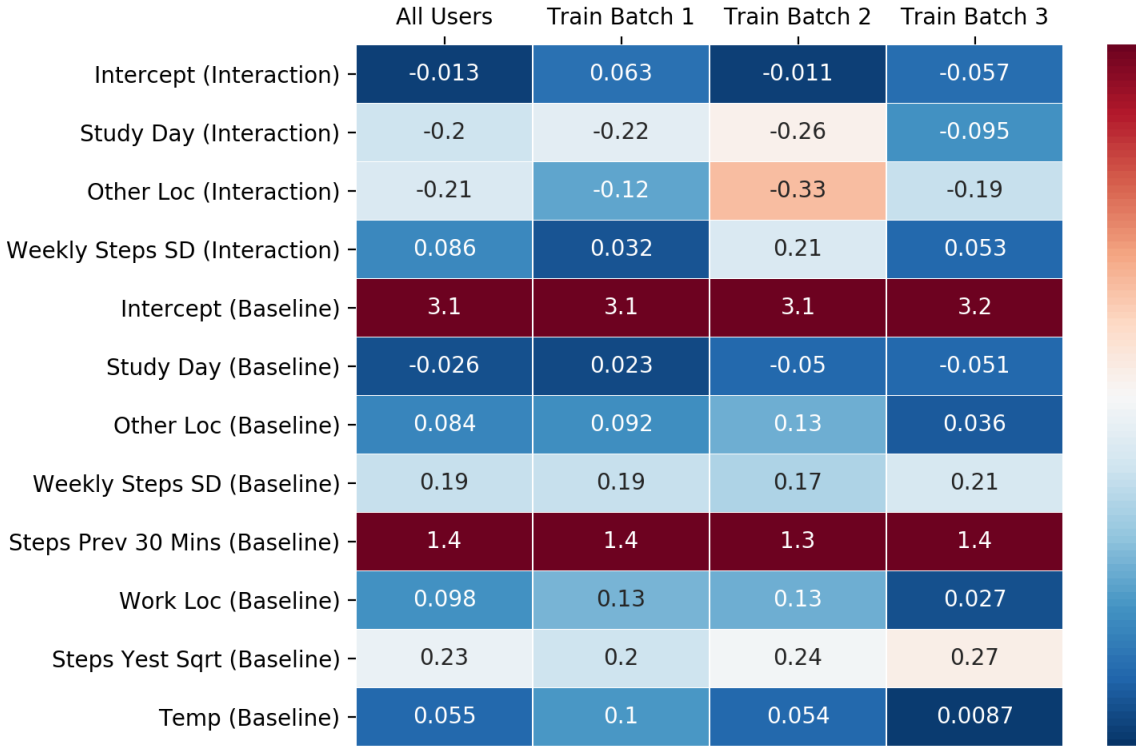
Figure A.30: $MUER$ for Varying `prior_cov_mult`, StdDev Cutoff Optimization

B

OLS Results and Coefficients

B.1 Full Model OLS Results

B.2 Small Model OLS Results



Feature		All Users	Train Batch 1	Train Batch 2	Train Batch 3
Intercept	Interact	-0.012933	0.063370	-0.011041	-0.057476
Study Day	Interact	-0.199678	-0.220172	-0.263731	-0.095406
Other Loc	Interact	-0.212436	-0.115700	-0.327226	-0.191645
Weekly Steps SD	Interact	0.086211	0.031911	0.212389	0.052531
Intercept	Baseline	3.128955	3.097138	3.112697	3.182046
Study Day	Baseline	-0.026399	0.023134	-0.050334	-0.051322
Other Loc	Baseline	0.084443	0.092011	0.128750	0.035890
Weekly Steps SD	Baseline	0.190982	0.192708	0.172558	0.205276
Steps Prev 30 Mins	Baseline	1.364978	1.352779	1.336849	1.400528
Work Loc	Baseline	0.098259	0.131462	0.128784	0.026610
Steps Yest Sqrt	Baseline	0.231959	0.198348	0.243200	0.269906
Temp	Baseline	0.054507	0.101924	0.053583	0.008715

Figure B.1: Full Model Regression $\hat{\Theta}$ Estimates

Dep. Variable:	Reward	R-squared:	0.237
Model:	OLS	Adj. R-squared:	0.236
Method:	Least Squares	F-statistic:	168.1
Date:	Sat, 05 May 2018	Prob (F-statistic):	0.00
Time:	22:47:56	Log-Likelihood:	-14307.
No. Observations:	5961	AIC:	2.864e+04
Df Residuals:	5949	BIC:	2.872e+04
Df Model:	11		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept (Interaction)	-0.0129	0.074	-0.174	0.862	-0.158	0.133
Study Day (Interaction)	-0.1997	0.074	-2.708	0.007	-0.344	-0.055
Other Loc (Interaction)	-0.2124	0.073	-2.925	0.003	-0.355	-0.070
Weekly Steps SD (Interaction)	0.0862	0.082	1.053	0.292	-0.074	0.247
Intercept (Baseline)	3.1290	0.043	73.188	0.000	3.045	3.213
Study Day (Baseline)	-0.0264	0.043	-0.610	0.542	-0.111	0.058
Other Loc (Baseline)	0.0844	0.049	1.712	0.087	-0.012	0.181
Weekly Steps SD (Baseline)	0.1910	0.048	4.009	0.000	0.098	0.284
Steps Prev 30 Mins (Baseline)	1.3650	0.035	38.626	0.000	1.296	1.434
Work Loc (Baseline)	0.0983	0.041	2.392	0.017	0.018	0.179
Steps Yest Sqrt (Baseline)	0.2320	0.036	6.390	0.000	0.161	0.303
Temp (Baseline)	0.0545	0.038	1.428	0.153	-0.020	0.129

Omnibus:	281.851	Durbin-Watson:	1.915
Prob(Omnibus):	0.000	Jarque-Bera (JB):	150.891
Skew:	-0.223	Prob(JB):	1.72e-33
Kurtosis:	2.360	Cond. No.	3.34

Table B.1: Full Model Regression, All Users Together

Dep. Variable:	Reward	R-squared:	0.231
Model:	OLS	Adj. R-squared:	0.229
Method:	Least Squares	F-statistic:	108.6
Date:	Sat, 05 May 2018	Prob (F-statistic):	3.07e-217
Time:	22:47:56	Log-Likelihood:	-9619.4
No. Observations:	3983	AIC:	1.926e+04
Df Residuals:	3971	BIC:	1.934e+04
Df Model:	11		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept (Interaction)	0.0634	0.094	0.673	0.501	-0.121	0.248
Study Day (Interaction)	-0.2202	0.092	-2.381	0.017	-0.401	-0.039
Other Loc (Interaction)	-0.1157	0.089	-1.297	0.195	-0.291	0.059
Weekly Steps SD (Interaction)	0.0319	0.090	0.354	0.723	-0.145	0.209
Intercept (Baseline)	3.0971	0.053	58.455	0.000	2.993	3.201
Study Day (Baseline)	0.0231	0.054	0.431	0.666	-0.082	0.128
Other Loc (Baseline)	0.0920	0.059	1.572	0.116	-0.023	0.207
Weekly Steps SD (Baseline)	0.1927	0.052	3.703	0.000	0.091	0.295
Steps Prev 30 Mins (Baseline)	1.3528	0.044	30.987	0.000	1.267	1.438
Work Loc (Baseline)	0.1315	0.050	2.635	0.008	0.034	0.229
Steps Yest Sqrt (Baseline)	0.1983	0.043	4.627	0.000	0.114	0.282
Temp (Baseline)	0.1019	0.048	2.110	0.035	0.007	0.197

Omnibus:	197.556	Durbin-Watson:	1.882
Prob(Omnibus):	0.000	Jarque-Bera (JB):	97.931
Skew:	-0.196	Prob(JB):	5.43e-22
Kurtosis:	2.339	Cond. No.	3.31

Table B.2: Full Model Regression, Training Batch 1

Dep. Variable:	Reward	R-squared:	0.236
Model:	OLS	Adj. R-squared:	0.234
Method:	Least Squares	F-statistic:	113.5
Date:	Sat, 05 May 2018	Prob (F-statistic):	1.31e-226
Time:	22:47:56	Log-Likelihood:	-9754.8
No. Observations:	4057	AIC:	1.953e+04
Df Residuals:	4045	BIC:	1.961e+04
Df Model:	11		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept (Interaction)	-0.0110	0.089	-0.124	0.902	-0.186	0.164
Study Day (Interaction)	-0.2637	0.089	-2.960	0.003	-0.438	-0.089
Other Loc (Interaction)	-0.3272	0.090	-3.638	0.000	-0.504	-0.151
Weekly Steps SD (Interaction)	0.2124	0.116	1.838	0.066	-0.014	0.439
Intercept (Baseline)	3.1127	0.053	58.396	0.000	3.008	3.217
Study Day (Baseline)	-0.0503	0.053	-0.948	0.343	-0.154	0.054
Other Loc (Baseline)	0.1288	0.065	1.975	0.048	0.001	0.257
Weekly Steps SD (Baseline)	0.1726	0.071	2.433	0.015	0.034	0.312
Steps Prev 30 Mins (Baseline)	1.3368	0.043	31.015	0.000	1.252	1.421
Work Loc (Baseline)	0.1288	0.051	2.517	0.012	0.028	0.229
Steps Yest Sqrt (Baseline)	0.2432	0.046	5.291	0.000	0.153	0.333
Temp (Baseline)	0.0536	0.047	1.140	0.254	-0.039	0.146

Omnibus:	240.090	Durbin-Watson:	1.917
Prob(Omnibus):	0.000	Jarque-Bera (JB):	110.868
Skew:	-0.203	Prob(JB):	8.42e-25
Kurtosis:	2.300	Cond. No.	4.04

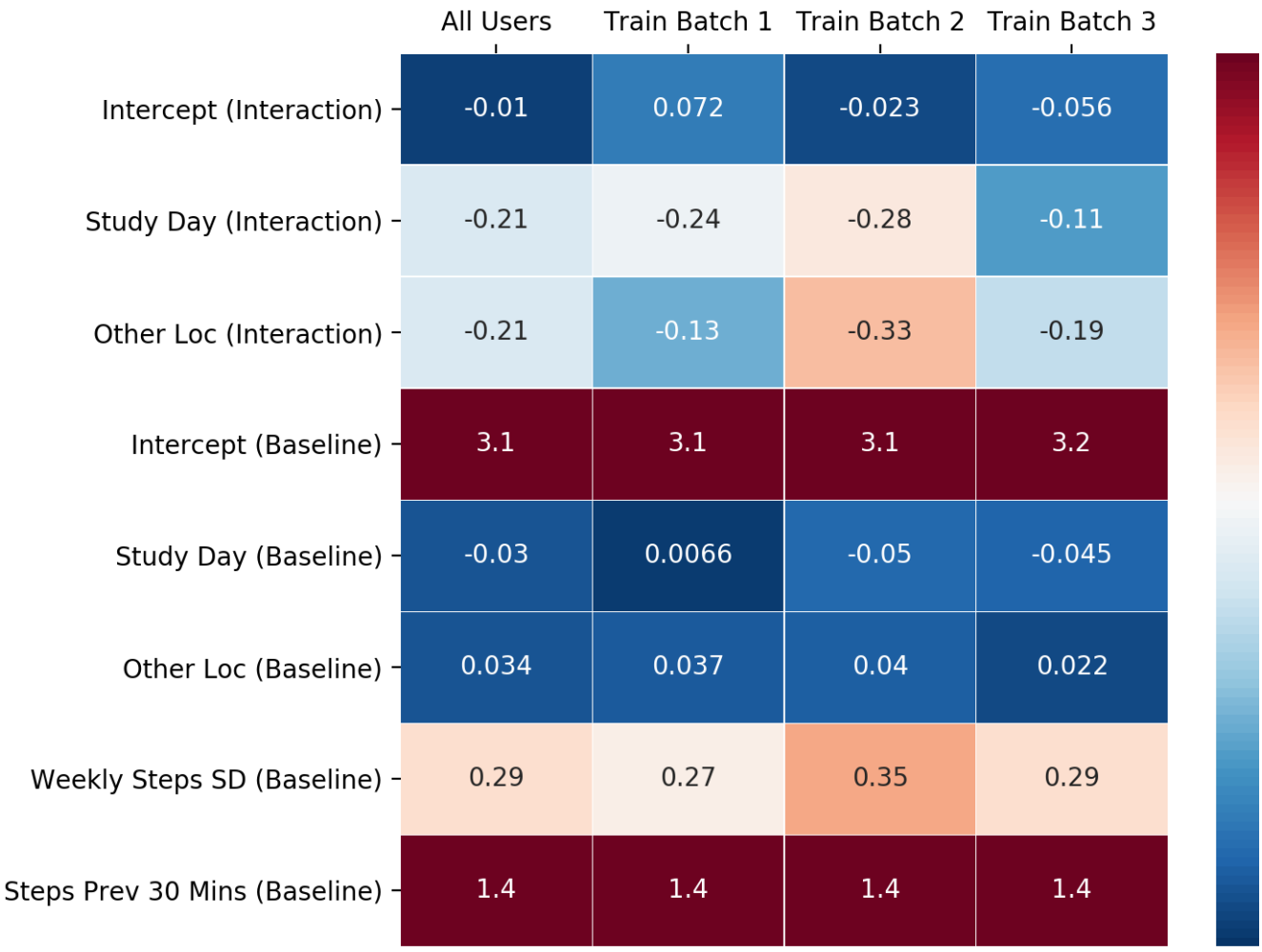
Table B.3: Full Model Regression, Training Batch 2

Dep. Variable:	Reward	R-squared:	0.248
Model:	OLS	Adj. R-squared:	0.246
Method:	Least Squares	F-statistic:	116.1
Date:	Sat, 05 May 2018	Prob (F-statistic):	4.65e-230
Time:	22:47:56	Log-Likelihood:	-9227.7
No. Observations:	3882	AIC:	1.848e+04
Df Residuals:	3870	BIC:	1.855e+04
Df Model:	11		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept (Interaction)	-0.0575	0.091	-0.634	0.526	-0.235	0.120
Study Day (Interaction)	-0.0954	0.090	-1.066	0.287	-0.271	0.080
Other Loc (Interaction)	-0.1916	0.088	-2.170	0.030	-0.365	-0.018
Weekly Steps SD (Interaction)	0.0525	0.101	0.519	0.604	-0.146	0.251
Intercept (Baseline)	3.1820	0.052	60.638	0.000	3.079	3.285
Study Day (Baseline)	-0.0513	0.052	-0.981	0.327	-0.154	0.051
Other Loc (Baseline)	0.0359	0.059	0.610	0.542	-0.080	0.151
Weekly Steps SD (Baseline)	0.2053	0.057	3.605	0.000	0.094	0.317
Steps Prev 30 Mins (Baseline)	1.4005	0.043	32.403	0.000	1.316	1.485
Work Loc (Baseline)	0.0266	0.052	0.514	0.607	-0.075	0.128
Steps Yest Sqrt (Baseline)	0.2699	0.045	5.978	0.000	0.181	0.358
Temp (Baseline)	0.0087	0.047	0.186	0.852	-0.083	0.100

Omnibus:	141.111	Durbin-Watson:	1.962
Prob(Omnibus):	0.000	Jarque-Bera (JB):	95.075
Skew:	-0.267	Prob(JB):	2.26e-21
Kurtosis:	2.450	Cond. No.	3.32

Table B.4: Full Model Regression, Training Batch 3



Feature		All Users	Train Batch 1	Train Batch 2	Train Batch 3
Intercept	Interact	-0.010436	0.071858	-0.023353	-0.056086
Study Day	Interact	-0.211407	-0.236143	-0.276794	-0.107167
Other Loc	Interact	-0.214480	-0.125237	-0.325202	-0.185378
Intercept	Baseline	3.127301	3.096886	3.125608	3.163705
Study Day	Baseline	-0.030159	0.006554	-0.050069	-0.045362
Other Loc	Baseline	0.033786	0.037067	0.040477	0.022279
Weekly Steps SD	Baseline	0.292857	0.266203	0.347553	0.290910
Steps Prev 30 Mins	Baseline	1.384952	1.370012	1.358397	1.425091

Figure B.2: Small Model Regression $\hat{\Theta}$ Estimates

Dep. Variable:	Reward	R-squared:	0.230
Model:	OLS	Adj. R-squared:	0.230
Method:	Least Squares	F-statistic:	254.6
Date:	Sat, 05 May 2018	Prob (F-statistic):	0.00
Time:	22:46:55	Log-Likelihood:	-14334.
No. Observations:	5961	AIC:	2.868e+04
Df Residuals:	5953	BIC:	2.874e+04
Df Model:	7		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept (Interaction)	-0.0104	0.074	-0.140	0.889	-0.156	0.136
Study Day (Interaction)	-0.2114	0.074	-2.863	0.004	-0.356	-0.067
Other Loc (Interaction)	-0.2145	0.073	-2.943	0.003	-0.357	-0.072
Intercept (Baseline)	3.1273	0.043	72.859	0.000	3.043	3.211
Study Day (Baseline)	-0.0302	0.043	-0.703	0.482	-0.114	0.054
Other Loc (Baseline)	0.0338	0.045	0.757	0.449	-0.054	0.121
Weekly Steps SD (Baseline)	0.2929	0.039	7.588	0.000	0.217	0.369
Steps Prev 30 Mins (Baseline)	1.3850	0.035	39.227	0.000	1.316	1.454

Omnibus:	294.145	Durbin-Watson:	1.896
Prob(Omnibus):	0.000	Jarque-Bera (JB):	144.932
Skew:	-0.192	Prob(JB):	3.38e-32
Kurtosis:	2.340	Cond. No.	2.73

Table B.5: Small Model Regression, All Users Together

Dep. Variable:	Reward	R-squared:	0.225
Model:	OLS	Adj. R-squared:	0.223
Method:	Least Squares	F-statistic:	164.7
Date:	Sat, 05 May 2018	Prob (F-statistic):	2.13e-214
Time:	22:46:55	Log-Likelihood:	-9636.0
No. Observations:	3983	AIC:	1.929e+04
Df Residuals:	3975	BIC:	1.934e+04
Df Model:	7		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept (Interaction)	0.0719	0.094	0.763	0.446	-0.113	0.257
Study Day (Interaction)	-0.2361	0.093	-2.551	0.011	-0.418	-0.055
Other Loc (Interaction)	-0.1252	0.089	-1.400	0.162	-0.301	0.050
Intercept (Baseline)	3.0969	0.053	58.427	0.000	2.993	3.201
Study Day (Baseline)	0.0066	0.053	0.124	0.902	-0.097	0.111
Other Loc (Baseline)	0.0371	0.054	0.685	0.493	-0.069	0.143
Weekly Steps SD (Baseline)	0.2662	0.042	6.293	0.000	0.183	0.349
Steps Prev 30 Mins (Baseline)	1.3700	0.044	31.476	0.000	1.285	1.455

Omnibus:	206.934	Durbin-Watson:	1.867
Prob(Omnibus):	0.000	Jarque-Bera (JB):	95.047
Skew:	-0.167	Prob(JB):	2.30e-21
Kurtosis:	2.321	Cond. No.	2.89

Table B.6: Small Model Regression, Training Batch 1

Dep. Variable:	Reward	R-squared:	0.228
Model:	OLS	Adj. R-squared:	0.227
Method:	Least Squares	F-statistic:	170.7
Date:	Sat, 05 May 2018	Prob (F-statistic):	6.01e-222
Time:	22:46:55	Log-Likelihood:	-9775.8
No. Observations:	4057	AIC:	1.957e+04
Df Residuals:	4049	BIC:	1.962e+04
Df Model:	7		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept (Interaction)	-0.0234	0.090	-0.261	0.794	-0.199	0.152
Study Day (Interaction)	-0.2768	0.089	-3.104	0.002	-0.452	-0.102
Other Loc (Interaction)	-0.3252	0.090	-3.601	0.000	-0.502	-0.148
Intercept (Baseline)	3.1256	0.053	58.908	0.000	3.022	3.230
Study Day (Baseline)	-0.0501	0.053	-0.950	0.342	-0.153	0.053
Other Loc (Baseline)	0.0405	0.056	0.723	0.470	-0.069	0.150
Weekly Steps SD (Baseline)	0.3476	0.056	6.210	0.000	0.238	0.457
Steps Prev 30 Mins (Baseline)	1.3584	0.043	31.485	0.000	1.274	1.443

Omnibus:	248.433	Durbin-Watson:	1.900
Prob(Omnibus):	0.000	Jarque-Bera (JB):	106.236
Skew:	-0.170	Prob(JB):	8.53e-24
Kurtosis:	2.284	Cond. No.	2.74

Table B.7: Small Model Regression, Training Batch 2

Dep. Variable:	Reward	R-squared:	0.241
Model:	OLS	Adj. R-squared:	0.240
Method:	Least Squares	F-statistic:	175.7
Date:	Sat, 05 May 2018	Prob (F-statistic):	1.80e-226
Time:	22:46:55	Log-Likelihood:	-9246.3
No. Observations:	3882	AIC:	1.851e+04
Df Residuals:	3874	BIC:	1.856e+04
Df Model:	7		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept (Interaction)	-0.0561	0.091	-0.618	0.537	-0.234	0.122
Study Day (Interaction)	-0.1072	0.090	-1.196	0.232	-0.283	0.069
Other Loc (Interaction)	-0.1854	0.089	-2.091	0.037	-0.359	-0.012
Intercept (Baseline)	3.1637	0.052	60.819	0.000	3.062	3.266
Study Day (Baseline)	-0.0454	0.052	-0.873	0.383	-0.147	0.056
Other Loc (Baseline)	0.0223	0.054	0.412	0.681	-0.084	0.128
Weekly Steps SD (Baseline)	0.2909	0.047	6.210	0.000	0.199	0.383
Steps Prev 30 Mins (Baseline)	1.4251	0.043	33.022	0.000	1.340	1.510

Omnibus:	146.229	Durbin-Watson:	1 .936
Prob(Omnibus):	0.000	Jarque-Bera (JB):	92.085
Skew:	-0.246	Prob(JB):	1.01e-20
Kurtosis:	2.428	Cond. No.	2.73

Table B.8: Small Model Regression, Training Batch 3

References

[1] Agrawal, S. & Goyal, N. (2012). Thompson sampling for contextual bandits with linear payoffs.

[2] Chapelle, O. & Li, L. (2011). An empirical evaluation of thompson sampling. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 24* (pp. 2249–2257). Curran Associates, Inc.

[3] Chapelle, O., Manavoglu, E., & Rosales, R. (2014). Simple and scalable response prediction for display advertising. *ACM Transactions on Intelligent Systems and Technology*, 5(4), 1–34.

[4] Greenewald, K., Tewari, A., Murphy, S., & Klasnja, P. (2017). Action centered contextual bandits. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 30* (pp. 5977–5985). Curran Associates, Inc.

[5] Li, L., Chu, W., Langford, J., & Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web - WWW '10*: ACM Press.

[6] Li, L., Chu, W., Langford, J., & Wang, X. (2011). Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining - WSDM '11*: ACM Press.

[7] Liao, P., Klasnja, P., Tewari, A., & Murphy, S. A. (2015). Micro-randomized trials in mhealth.

[8] McGill, H. C., McMahan, C. A., & Gidding, S. S. (2008). Preventing heart disease in the 21st century: Implications of the pathobiological determinants of atherosclerosis in youth (PDAY) study. *Circulation*, 117(9), 1216–1227.

[9] Organization, W. H. (2012). *Global Atlas on Cardiovascular Disease Prevention and Control*. World Health Organization.

[10] Ramkumar, P. N., Muschler, G. F., Spindler, K. P., Harris, J. D., McCulloch, P. C., & Mont, M. A. (2017). Open mHealth architecture: A primer for tomorrow's orthopedic surgeon and introduction to its use in lower extremity arthroplasty. *The Journal of Arthroplasty*, 32(4), 1058–1062.

[11] Smith, S. N., Lee, A. J., Hall, K., Seewald, N. J., Boruvka, A., Murphy, S. A., & Klasnja, P. (2017). Design lessons from a micro-randomized pilot study in mobile health. In *Mobile Health* (pp. 59–82). Springer International Publishing.

[12] Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. A Bradford Book.