Evaluating and Improving Targeting Policies with Field Experiments
Using Counterfactual Policy Logging

September 2018

Duncan Simester
MIT

Artem Timoshenko
MIT

Spyros I. Zoumpoulis
INSEAD

The gold standard for evaluating targeting policies is to evaluate them using a randomized experiment. We present an approach to designing and analyzing targeting experiments based on “counterfactual policy logging,” which logs recommended actions from candidate targeting policies. Leveraging counterfactual policy logging offers three important advantages over standard methods. First, the proposed approach can be used to evaluate any policies, including policies designed after the experiment is implemented. If we are interested only in comparing policies, counterfactual policy logging allows the omission of some customer segments, reducing experimentation costs without losing precision in estimation. Second, accounting for the counterfactual policies yields more efficient estimates of the difference in the performance of the policies. Third, counterfactual policy logging offers opportunities to improve targeting policies. We illustrate these advantages using data from an actual field experiment.

This is a revision of an August 2017 version of this paper that was previously titled “Efficiently Evaluating Targeting Policies Using Field Experiments.”
Evaluating and Improving Targeting Policies with Field Experiments
Using Counterfactual Policy Logging

1. Introduction

Targeting policies are used in marketing to match different firm actions to different customers. For example, retailers want to send different promotions to different customers, media owners want to show different digital advertisements to different users, online entertainment platforms recommend different content to different customers, real estate agents want to show different homes, financial advisors want to recommend different products, and car dealers want to propose different prices.

A standard approach to comparing alternative targeting policies is to implement the policies on randomly selected groups of participants. The participants’ responses are then typically used to calculate the aggregate outcome for each policy, and these aggregate outcomes are then compared across policies. For example, Skiera and Nabout (2013) propose a model that targets different search engine keywords with different bids by an advertiser. They test their model using a field experiment in which bids for twenty keywords were submitted using either the current policy or the proposed model. Similarly, Mantrala et al. (2006) proposed a model for setting different prices for different automobile parts. They validated their model using a field experiment in which 200 stores were randomly assigned to the proposed policy, and 300 stores were randomly assigned to the current policy.

In this paper, we propose a program for evaluating, comparing and selecting from a candidate set of targeting policies, using data from a field experiment. The program includes four steps.

Step 1: Implement a validation experiment by randomly assigning customers to actions. Instead of one experimental condition for each candidate policy, there is one experimental condition for each possible action.

Step 2: Segment customers using counterfactual policy logging. Targeting policies represent a mapping from targeting variables (covariates) to recommended actions, and can be thought of as candidate methods for summarizing the information provided by the covariates. The mapping to recommended actions provides a segmentation of customers.

Step 3: Use the segmentation in Step 2 to evaluate policies and compare the performance of any pair of policies. Use OLS or double machine learning to incorporate additional covariates to control for customer heterogeneity.

Step 4: Improve targeting methods using the segmentation of customers in Step 2. Evaluate the performance of the new policy using the experimental data from Step 1, without needing to re-run the experiment in Step 1.

This program addresses the following components of the problem we are studying: experimentation, evaluation and comparison of different policies, and improvement of the candidate policies. It offers important benefits across all three components:

1. Experimentation. A disadvantage of the standard randomization-by-policy experimental design is that it only allows comparison of the policies implemented in the experiment. In contrast, the proposed
randomization-by-action design (Step 1) allows evaluation of any policy, including policies designed after the experiment was implemented.

2. Evaluation and comparison of policies. The segmentation and estimation approach described in Steps 2 and 3 provide a more efficient comparison of two policies than simply calculating the difference in the aggregate performance of each policy. More specifically, when holding the total number of participants fixed, we obtain a more accurate estimate of the performance difference. These efficiency benefits have two sources:

   a. For customers for whom different policies recommend the same action, we know the true difference in the performance of the policies is precisely zero. The proposed approach recognizes this, and avoids introducing error due to random differences in observed performances.

   b. When comparing targeting policies across heterogeneous customers, it is generally more efficient to segment the customers, calculate the difference in performance within each segment, and then aggregate these differences across segments (versus calculating the difference in the aggregate performance of each policy).

Both of these efficiency benefits are enabled by the segmentation we propose in Step 2 of our program. In addition to these two benefits, further efficiency improvements may be obtained by including additional covariates to control for heterogeneity, and using double machine learning to address the overfitting and regularization biases.

3. Improvement of policies. The data from the Step 1 experiment provides new information that can be used to develop and evaluate a new targeting policy. However, training the new policy solely using this new dataset, may mean losing some of the knowledge that was used to create existing policies. The segmentation in Step 2 provides a way to incorporate the knowledge contained in existing policies into the training of the new policy without needing to merge data. We show that this can improve the performance of the new policy, even if the performance of the existing policies is relatively poor. The new improved policy can then be used in practice going forward. This is Step 4 of our program.

In this paper we describe each of the four steps in detail. We also illustrate their implementation and document their benefits using data from a large field experiment.

The program uses “counterfactual policy logging” to segment customers in Step 2, and many of the benefits of the program rely upon this feature. Instead of simply asking which experimental treatment each subject was treated with, counterfactual policy logging asks: which treatment would the subject receive under the alternative policies? “Policy logging” describes the recording of recommended actions from a candidate policy, while the “counterfactual” term recognizes that the recommended action may not be the action that the customer was treated with. Counterfactual policy logging identifies groups of customers for which the candidate policies recommend the same actions, irrespective of which actions were actually implemented in the experiment.

The benefits of segmenting using counterfactual policy logging extend from experimental design, to policy evaluation and policy improvement. When designing the experiment, if we are interested only in the comparison between policies, we can omit targeting segments of customers where the policies would assign the same action. This results in reduced cost to attain the same precision in estimating the
performance difference, or increased precision at the same cost. Furthermore, the segmentation of
customers using counterfactual policy logging enables the elimination of random error when comparing
policies in segments for which the policies recommend the same action (Benefit 2a). It also provides a
convenient way of using the targeting variables to group customers, and so facilitates segment-level
analysis, which helps remove between-segment variance (Benefit 2b). Finally, counterfactual policy
logging summarizes the information contained in existing policies, which provides a convenient way to
incorporate this information when training new policies (Benefit 3).

It is helpful to distinguish the research on evaluating targeting methods from other marketing topics that
have been studied using field experiments. In a typical marketing study, a researcher implements one or
more actions (e.g., promotions) and compares them with a no-action (control) condition. The analysis will
generally include calculation of an average treatment effect, and heterogeneous treatment effects across
different customer segments. Targeting methods extend this analysis by adding an optimization step to
match interventions with different customer segments. Evaluation of targeting policies thus involves
evaluating the optimized mappings from the customer segments to the interventions.

This distinction between evaluating an action and evaluating a targeting policy is important because it
changes the traditional experimental design. Consider, for example, the design of an experiment to
measure the impact of a marketing promotion. The traditional approach would be to randomly assign the
promotion to some customers and not others. This is what we call a “randomized-by-action” design.

In contrast, now consider the design of an experiment to compare two targeting policies. The traditional
approach is to randomly assign the two policies to different groups of customers and compare the
outcomes. This is what we call a “randomized-by-policy” design. Instead, our program proposes a
randomized-by-action design, even though we are comparing targeting policies.

Examples of Recent Studies Comparing Targeting Policies

We will illustrate our proposed program using data from a recent study comparing seven widely-used
targeting methods (Simester, Timoshenko, and Zoumpoulis, 2018). The study compares how robust the
methods are to common data challenges, including covariate shift, concept shift, information loss through
aggregation, and imbalanced data. This study provides an ideal illustration of our proposed program
because it includes both randomized-by-policy and randomized-by-action experimental conditions. The
STZ paper is one of a wave of recent studies investigating and evaluating targeting methods. The recent
acceleration of research on targeting reflects the growth of interest in machine learning methods. Other
examples of recent papers that study targeting of marketing actions, and which represent potential
applications of the proposed program, include Dubé and Misra (2017), Ostrovsky and Schwarz (2011),
and Rafieian and Yoganarasimhan (2018). We will briefly review each of these possible applications.

Dubé and Misra (2017) train a price targeting model and validate this model using an experiment in
which subjects are randomly assigned to policies. Their design includes two uniform benchmark policies
and their proposed policy. A randomized-by-action design could be a viable alternative in this setting, and
would allow developing and evaluating new policies after the experiment. A limitation of this proposal is
that the range of prices a firm can charge is continuous, and so the action space is infinite. One solution is
to discretize this continuous variable. For example, Dubé and Misra (2017) round targeted prices down to
the nearest $9 price ending, yielding 39 possible prices ranging from $119 to $499 in $10 increments.
Although implementing a pricing experiment with as many as 39 treatments may not be feasible, or may
yield relatively few customers assigned to each treatment, we propose options for accommodating large
action spaces in our discussion of scalability in Section 8. Even without changing the experimental design, the proposed estimation approach could improve the efficiency of their comparisons. In particular, for subjects for whom their optimized policy recommends the same price as one of the benchmark policies, we know that the true difference between the proposed policy and the benchmark policy is zero. The proposed estimation approach would set this difference to zero and eliminate variance introduced by random noise.

Ostrovsky and Schwarz (2011) study how to set reserve prices in Internet advertising auctions. They present the results of an experiment in which reserve prices were randomly assigned either to a uniform benchmark price or to a price proposed by the targeting model. Similar to Dubé and Misra (2017), randomization-by-action design would require discretizing prices, but would again enable evaluation of a much wider range of targeting policies. Our proposed estimation approach would also improve the efficiency of the comparisons of the candidate policies.

Rafieian and Yoganarasimhan (2018) is an example of a targeting setting with many potential actions, where the randomization-by-action design is actually implemented. This paper studies targeting of mobile advertising at a large advertisement network. The platform uses a quasi-proportional auction mechanism to allocate advertisement positions, which ensures positive probabilities of displaying advertisements by all bidders. This randomization-by-action design allows evaluation of any targeting policy. Rafieian and Yoganarasimhan (2018) include a range of covariates when comparing their proposed model with the firm’s current model. Counterfactual policy logging has the potential to improve the efficiency of this performance comparison, particularly given the presence of covariates. It also provides a convenient way to use the information in the current policy when training a new targeting policy.

Outline of the Paper

The paper continues in Section 2 with a review of the literature. In Sections 3, 4 and 5 we discuss the first three steps of the proposed program. We illustrate these steps using an example of an actual field experiment conducted to compare candidate targeting policies. The efficiency advantages are reviewed in Section 6, where we provide formal guarantees that efficiency will improve when using Steps 2 and 3 compared to the traditional approach of comparing aggregate outcomes across policies. In Section 7, we illustrate these efficiency advantages using actual data from the example field experiment. In Section 8 we discuss the scalability of the proposed program. Scaling can occur along several dimensions, including the number of actions, number of policies and number of covariates. We discuss the feasibility of scaling along each of these dimensions, and describe the adjustments that may be required. In Section 9 we illustrate how counterfactual policy logging can be used to improve the training of new targeting policies. The paper concludes in Section 10.

2. Literature Review

Our program is composed of multiple features, some of which are established concepts in the computer science literature, and some of which have recently received attention in the marketing literature. Our contribution is to compose these isolated methods into a systematic program that provides a foundation for future research on evaluating and improving targeting of marketing actions. In this section we describe how the proposed program relates to the existing literatures. We highlight how counterfactual policy logging can be used to complement and improve standard methods.
Offline Policy Evaluation in Computer Science

Our work relates to the stream of computer science research on offline policy evaluation and learning, where a policy is a function that maps observed covariates to recommended actions. In particular, we assume the researcher has access to the experimental data from Step 1, but no ability to gather new data (Langford et al., 2008; Strehl et al., 2010; Dudík et al., 2011). This setting is also known more generally as the offline version of the contextual bandits problem. Two related tasks arise in this setting: policy evaluation and policy optimization (or learning). Step 3 in our proposed program focuses on policy evaluation, which can be used to both estimate the expected reward of a given policy, and to compare the performance of multiple policies. Step 4 focuses on policy optimization, which seeks to improve policies.

The computer science literature distinguishes between on-policy and off-policy evaluation. In on-policy evaluation the evaluation data is constructed by implementing the policy. In off-policy evaluation the policy is evaluated using data constructed by implementing a different policy (Sutton and Barto, 1998). Randomizing by policy is an example of on-policy evaluation, while randomizing by action is an example of off-policy evaluation.

The advantage of on-policy evaluation is that it is generally easier to evaluate the outcome for an implemented policy than for a policy that has not been implemented. Of course, the limitation with on-policy evaluation is that it is not always possible to implement all policies that the researcher wants to evaluate. Thus off-policy evaluation is critical in real-world applications of reinforcement learning where it is infeasible to estimate a policy by running it. Off-policy evaluation has been applied extensively in domains such as robotics, business, healthcare and public policy (Li et al., 2011; Bottou et al., 2013; Thomas et al., 2015; Murphy et al., 2001; Hirano et al., 2003).

The primary challenge in off-policy learning is that the evaluation data does not reveal the outcomes for actions that were not implemented. Randomization in the evaluation data can solve this challenge. Early research on off-policy evaluation in the machine learning literature (e.g., Langford et al. 2008, Strehl et al. 2010) establishes several conditions under which off-policy evaluation is possible. First, if there is sufficient randomization in the evaluation data or the candidate targeting policies. Second, if the policy used to construct the evaluation data does not depend upon the covariates and each action is chosen sufficiently often. Third, if the evaluation data was constructed using multiple policies, as long as there is enough variation in the policies and the choice of which policy to use for each subject is independent of the covariates or context (including the case of random assignment). Fourth, if the policy used to construct the evaluation data varies over time.

All of these conditions may yield sufficient exogenous variation in the evaluation data to allow off-policy evaluation. Notice that randomizing by policy falls under the third possibility. If there is sufficient variation in a randomized-by-policy design, it may be possible to evaluate any policy. However, as we will illustrate, randomizing by policy will not always provide enough variation for this to be possible. Randomizing by action guarantees that the randomization is sufficient (although precision will depend upon the sample size and the size of the action space).

There are two main approaches to off-policy, offline policy evaluation. The first approach (sometimes called the “direct method”) is regression-based. These methods estimate the reward function from data and use this estimate in place of the actual reward to evaluate the candidate policy (Dudík et al., 2011; Li et al., 2011; Li et al., 2014; Swaminathan and Joachims, 2015). The second approach uses the inverse
propensity score (IPS). IPS methods estimate a model of the action probabilities of the data generating policy, and use importance weighting to correct for the incorrect proportions of actions in the data from the data generating policy, as compared to the candidate policy (Strehl et al. 2010; Dudík et al. 2011; Dudík et al. 2014; Jiang and Li 2016).

The statistics literature includes previous proposals to combine the two approaches. Cassel et al. (1976), Robins et al. (1994) and Robins and Rotnitzky (1995) propose a “doubly robust” technique, which combines estimators, so that the final estimate is accurate as long as at least one of the combined estimators is accurate. In a standard application, a regression-based estimator and an importance sampling-based estimator are combined, producing an estimator that leverages the strengths and overcomes the weaknesses of the two approaches. The doubly robust estimator produces accurate value estimates as long as either a good model of rewards or a good model of the used policy is available.

Recent applications to policy evaluation include Dudík et al. (2011), Dudík et al. (2014), and Jiang and Li (2016). A variation of our implementation for the evaluation and comparison of policies (Step 3 of our program) is based on “double machine learning” (Chernozhukov et al., 2018), which borrows the doubly robust estimator to ensure inference of the treatment effect is sufficiently robust to mistakes in estimating the reward functions. We discuss this method in detail in Section 5, where we propose using double machine learning to include additional covariates to control for customer heterogeneity.

Policy Improvement

Better policy evaluation may facilitate policy optimization. Strehl et al. (2010) show that the policy that maximizes their proposed estimator for policy evaluation competes well with other policies. Dudík et al. (2011) and Dudík et al. (2014) run experiments where the ideas from policy evaluation are applied to policy optimization. Athey and Wager (2017) consider the problem of policy optimization subject to practical constraints, and derive lower bounds for the minimax regret of policy learning under constraints.

While the policy optimization literature has focused on proposing and evaluating new policy learning methods, in Section 9 we demonstrate that logging counterfactual recommendations of existing policies can help to incorporate existing knowledge when training new targeting models.

Causal Inference in Statistics

The evaluation and comparison of targeting policies using randomized experiments is also related to the large literature on causal inference of treatment effects from randomized experiments and observational data. Statisticians have studied the problem over the course of the last one hundred years. The potential outcomes framework, also known as the Rubin Causal Model following work by Rubin (1974, 1975, 1978), provides a conceptual foundation for the problem. Fisher’s exact p-values approach (Fisher 1925, 1935) calculates the probability that the test statistic is as large as or larger than its realized value, under the null hypothesis of no effect. Neyman’s repeated sampling approach (Neyman, 1923, 1990) derives a direct estimate of the treatment effect and also estimates the sampling variance of the estimator. Regression methods estimate the treatment effect as a coefficient in a linear model, and provide a

---

1 Dudík et al. (2011) contrast the two approaches using an example of a survey in which some members of the population did not respond. One approach is to calculate the probability of a response from each sub-population, and project to the overall population using the inverse probability of a response (the inverse propensity score). For example, if only a third of the members in a sub-population responded, we can correct for the non-response by weighting the recorded responses by a factor of three. The alternative is to use covariates from any available source of information and regress to directly predict the survey outcome.
straightforward and familiar way to incorporate covariates. Imbens and Rubin (2015) provide an excellent survey of these approaches.

An extension of this research has considered heterogeneous treatment effects, rather than just average treatment effects. Recent work employs machine learning techniques: tree- and forest-based methods (Athey and Imbens, 2016; Wager and Athey, 2017), deep neural networks (Hartford et al., 2017), and regularized regression, such as elastic net and Lasso (Athey et al., 2018).

A focus of the causal inference literature has been the efficiency advantages that can be obtained through either stratification (segmentation) or inclusion of covariates. Both stratification and including covariates as controls in a regression model are now standard approaches to reduce the variance in a regression estimate of a treatment effect.

The basic idea of stratification is to divide the sampling region into strata (segments), sample within each stratum separately, and then aggregate results from individual strata together to give an overall estimate, which usually has a smaller variance than the estimate without stratification. For example, Deng, Xu, Kohavi and Walker (2013), Imbens and Rubin (2015) and Athey and Imbens (2017), all recommend analyzing randomized experiments by stratum (segment). Our approach incorporates stratification by estimating effects per segment (and then aggregating). These segments summarize the information obtained through counterfactual policy logging (Step 2).

We also recommend including additional covariates in the estimation model to control for customer heterogeneity. These observable customer features might include demographics, past browsing, or past purchasing behavior (see for example Rafieian and Yoganarasimhan, 2018). It is well understood that including covariates can be an effective way to reduce the variance in the regression estimate of a treatment effect. Contrary to conventional wisdom, Freedman (2008) argued that OLS adjustment can lead to worsened asymptotic precision, invalid measures of precision, and small-sample bias. In sufficiently large samples, these problems are either minor or easily fixed (Lin 2013). In particular, OLS adjustment cannot hurt asymptotic precision when a full set of treatment–covariate interactions is included.

However, a full set of interactions may introduce dimensionality and overfitting problems. As a solution, Bloniarz et al. (2016) suggest deriving estimates of treatment effect using Lasso. We employ double machine learning, which also offers a solution to dimensionality and overfitting, and allows the improved use of a very broad set of machine learning methods (including Lasso).

**Randomization by Action and Counterfactual Policy Logging in Marketing**

Use of field experiments in the marketing literature has grown rapidly in recent years (Simester 2017). While this growth includes several recent studies using field experiments to validate new targeting policies, marketing has been slower to recognize the advantages of randomizing by action instead of randomizing by policy. A notable exception is a recent working paper by Hitsch and Misra (2018), whose research coincides with our own research on this topic. Hitsch and Misra (2018) also address the evaluation and comparison of targeting policies (their focus does not extend to policy improvement). They recommend a randomized-by-action experimental design, and recognize that data from such an

---

2 Buja et al. (2014) also review the arguments for conditioning on covariates, and the breakdown of these arguments in the presence of model misspecification. The fundamental issue is that when the covariates are random, they help estimation of the slope coefficients if and only if the model is correctly specified (see also Fogarty 2018).
experiment can be used to evaluate any targeting policy. However, they focus their discussion on the
distinction and comparison between direct and indirect methods of predicting the conditional average
treatment effect, which is a question that lies outside of our scope. Conventional methods estimate the
conditional average treatment effect indirectly, by first training a regression function based on the
difference between the observed and predicted outcome levels, and then providing an indirect prediction
of the effect from the predicted difference in outcome levels between treated and untreated units with
identical features. In contrast, recently developed estimators such as causal forests (Wager and Athey,
2017) and the k-nearest neighbors-based “treatment effect projection,” which Hitch and Misra (2018)
propose, are directly trained to predict the conditional average treatment effect. The main conceptual
difference between the two groups of methods is that direct estimation methods focus directly on the
conditional average treatment effect, without estimating outcome levels. Our evaluation methods in Step
3 of our program (Section 5) train a regression function based on outcome levels, and would thus be
classified as indirect methods by Hitch and Misra (2018).

There is at least one recent example in marketing in which counterfactual policy logging is used to
improve the efficiency of OLS estimates of average treatment effects by identifying customers for which
there is no true difference in performance. Johnson, Lewis and Reiley (2017) recognize that an intent to
treat on Yahoo! may fail if: “many users in the experiment do not see an ad because they either do not
visit Yahoo! at all or do not browse enough pages on Yahoo! during the campaigns”. They use
counterfactual policy logging to identify these customers in both the treatment and control groups.

Before estimating their OLS model, Johnson, Lewis and Reiley (2017) simply remove from the treatment
and control groups any customers for whom an intent to treat fails because their browsing behavior did
not provide a treatment opportunity. The true difference in the treatment and control outcomes for these
customers is zero, and so removal of these customers improves the efficiency of the estimation of
performance differences by eliminating differences due to random error. We also use the insight that the
true difference in performance for some customers is zero. However, comparing targeting policies is a
different problem from estimating the average effect of an advertising treatment on the treated (TOT), and
so we do not omit these customers from the estimation. As we will discuss in Section 5, omitting
segments of customers for whom the true difference in performance is zero would distort estimates of the
average difference in the performance of the candidate policies.

Johnson, Lewis and Nubbemeyer (2017) provide another example that addresses a related digital
advertising problem through counterfactual logging. They also focus on a reason that an intention to treat
a subject may fail. In particular, they recognize that a challenge in measuring the effectiveness of online
advertising is that advertising platforms allocate exposures to customers systematically. As a result, the
customers who see an advertisement are systematically different than customers who do not. These
selection differences mean that we cannot simply compare purchasing behavior of customers who are
exposed to an advertisement with those who are not. One solution is to compare all of the customers that
the platform intends to treat (irrespective of whether they are served advertisements by the platform), with
an equivalent group of customers that the platform does not intend to treat. However, this comparison is
inefficient because the outcomes for customers who will never be treated just add noise. To address this
problem, the authors propose identifying a superset of customers that would qualify to receive the
advertisements and then randomly assign whether the customers are shown the ads or not. This allows
removal of customers who would never be treated from both the treatment and the control group.

Johnson, Lewis and Nubbemeyer (2017) remove customers who the platforms will not treat, and then
calculate the mean performance in the treatment and control groups (together with the difference in these means).

3. **Randomize by Actions (Step 1)**

Recall that the standard experimental design for evaluating targeting policies constructs a separate experimental condition for each candidate policy. Customers are randomly assigned to policies, and so all customers in an experimental condition are targeted with the actions recommended by a single policy. A disadvantage of this standard design is that it generally does not allow the evaluation and comparison of new policies without an additional experiment. Our program proposes an alternative randomized-by-action experimental design, which allows evaluation and comparison of any targeting policies, and so any new policy can be evaluated using the existing data.

Instead of randomly assigning customers to policies, the randomization-by-action design randomly assigns customers to actions. Instead of one experimental condition for each candidate policy, there is one experimental condition for each possible action. The number of experimental conditions matches the number of possible actions (i.e., the size of the action space). In contrast, under the standard design, the number of experimental conditions matches the number of candidate targeting policies.

Because all participants within an experimental condition receive the same action, this is a relatively simple experiment to implement. The simplicity may also help to reduce implementation errors. Moreover, as long as the samples are randomly selected, the sample sizes across the experimental conditions need not be the same.3

Randomizing by action guarantees that, within each segment, each possible action is received by a random sample of customers. This means that for any segment of customers, there is an equivalent sub-segment assigned to each of the actions. As a result, we are always able to compare the performance of policies that recommend different actions for any segment.4

We can illustrate this point with an example. Consider a retailer that wants to target customers with promotional offers a, b, or c. Consider two targeting policies, Policy 1 and Policy 2, which both recommend sending promotion a to male customers. The policies differ in their recommendations for female customers. Policy 1 recommends sending promotion b to the female customers, while policy 2 recommends sending promotion c to female customers. Consider an experimental design that randomly assigns a third of all customers to each promotion (a, b, and c). Because the assignment is random, the three sub-samples of female customers are equivalent. The key idea behind evaluating a policy using randomized-by-action data is that to evaluate a policy on a segment of customers for which the policy

---

3 Our formal comparison of the efficiency of the proposed and traditional approaches in Section 6 relies on an assumption that the experimental design is balanced, so that the proportions assigned to each treatment are the same in each segment. However, the proportions do not need to be uniform for each treatment. We note also that this proof focuses on randomized-by-policy data.

4 It may not be possible to evaluate a new policy accurately using data collected from an already implemented policy if the implemented policy is deterministic. If there is no randomization at all across the different actions in the evaluation data, and the implemented policy depends on covariates (which is natural in the targeting setting), it will generally not be possible to evaluate new untested policies (Langford et al., 2008). One can alternatively think about randomization over actions as a stochastic targeting policy. A stochastic targeting policy maps covariates to distributions over actions. The use of stochastic policies has a tradition in the offline contextual bandits literature (Strehl et al., 2010; Dudík et al., 2011; Dudík et al, 2014; Jiang and Li, 2016).
recommends a specific action, we use the customers within the segment that were randomized to that specific action. In our example, this allows us to directly evaluate the outcomes from the female customers that received promotion $b$ and the female customers that received promotion $c$. If we also use the outcomes from the male customers who received promotion $a$, we can then evaluate the outcome of Policy 1 and Policy 2.

It is important that to evaluate every alternative policy, the experiment randomly assigns every type of customer to every possible action. For example, if the experiment just randomized between promotions $a$ and $b$, we could not evaluate policies that recommend promotion $c$. This never arises when customers are randomly assigned to actions, but it may occur when randomly assigning customers to policies. Consider a segment of customers for which two policies both recommend promotion $a$. If we randomly assign customers to policies, we will not be able to evaluate an alternative policy that recommends sending promotion $b$ or $c$ to that segment.

We can also illustrate this point using an actual study. Simester, Timoshenko, and Zoumpoulis (2018) (hereafter “STZ”) investigate how a retailer should target prospective customers with promotions. They consider three different actions, including two different types of promotions sent by mail and a third no-mail (control) treatment. We will label these actions as actions $a$, $b$, and $c$. They compare seven different optimized targeting policies, which we label Policy 1, ..., Policy 7. In their study, STZ randomly assign approximately four million prospective households to ten experimental conditions. These ten experimental conditions correspond to the seven candidate policies and the three uniform policies corresponding to each of the three actions. If a household was in one of the seven experimental conditions associated with a targeting policy, the household received the action recommended for it by that targeting policy. If the household was in one of the three experimental conditions associated with an action, the household received that action.

The STZ study highlights the advantage of randomizing by action. To evaluate any possible targeting policy we require that for any segment of customers there is a random sample of customers that received action $a$, another random sample that received action $b$, and a third random sample that received action $c$. Randomizing by action ensures that this is easily satisfied. In contrast, randomizing by policy does not guarantee this. It is possible that variation in the recommended actions across the seven targeting policies may mean that randomizing by policy yields enough variation. In particular, we will focus on the segmentation we propose in Step 2 (segmenting by the recommended actions for each policy) and ask: are all three actions recommended by at least one of the seven policies in every segment? For example, if Policy 1 recommends action $a$, Policy 4 recommends action $c$ and Policy 7 recommends action $b$ (within the same segment), then all three actions are represented in that segment. In the STZ study, all three actions are represented by the seven implemented policies for just 30.92% of the approximately four million households. For the remaining 69.08% of households, there is at least one action that is not recommended by any of the seven candidate policies. For these households it is not possible to evaluate and compare policies that recommend the omitted action(s).5

**Limitations of Randomizing by Action**

One potential disadvantage of the randomization-by-action approach is cost. It is sometimes obvious that an action is optimal for only a small segment of the population, and so randomly assigning customers to

---

5 We note that it is, however, possible to evaluate and compare new policies that choose from the set of actions chosen by the implemented policies. Langford et al. (2008) make a similar point.
receive this action may lead to an opportunity cost. For example, if mailing a catalog to customers is profitable for most customers, then deciding to withhold these catalogs from a randomly selected sample of customers will result in foregone profit. This cost can be minimized by under-sampling the actions that are not optimal for most customers.

In a related point, it may be unethical or unacceptable to randomly assign some customers to some conditions. For example, there is an extensive literature studying the impact of interventions designed to reduce poverty. An important research question in this literature is the design of targeting policies so as to ensure that the interventions only target the truly poor households and not the rich (Hanna and Karlan, 2017). Randomly assigning actions to households could result in some of the rich households receiving the interventions, which may be politically unacceptable. Alternatively, in the medical field, it may be unethical to withhold some treatments from some patients. This limitation can be easily addressed by designing the randomization procedures to prevent experimental conditions that are unacceptable or unethical. Although this may prevent evaluation of every possible policy, it allows evaluation of any policy that is acceptable and ethical.

We next describe how to segment customers using the actions recommended by each policy (Step 2).

### 4. Segment Customers Using Counterfactual Policy Logging (Step 2)

A common thread throughout this paper is the segmentation of customers by the actions recommended by each policy. We will illustrate our proposed segmentation using the STZ study. Three of the experimental treatments in STZ households correspond to the randomization by action proposed in Step 1. In particular, 357,653 households received action \( a \), 360,773 households received action \( b \), and 370,936 households received action \( c \). The households in the STZ study were grouped into carrier routes (approximately 400 households), and the randomization was conducted at the carrier route level, so that all of the households in the same carrier route received the same treatment. As a result there is more variation in sample sizes between the treatments than we might otherwise expect.

In Table 4.1 we summarize the segmentation of these households according to the actions recommended by each of two policies (Policy 1 and Policy 2). There are three possible actions and two policies and so this yields nine possible segments. Because the policies dictate an action for each customer, and this action is known, constructing the segmentation of customers by recommended actions is straightforward.

The segmentation facilitates evaluation of policies. For example, the shading in Table 4.1 illustrates how this sample can be used to evaluate Policy 1. To evaluate a policy within a segment, we use the customers within the segment that received the action recommended by that policy. For example, within the segment of customers in which Policy 1 recommended action \( a \) and Policy 2 recommended action \( b \), we evaluate Policy 1 using the 13,038 customers that were randomly assigned to action \( a \).

We can also see how to compare Policy 1 with Policy 2. There are 40,619 customers \((13,038 + 11,495 + 16,086)\) for which Policy 1 recommended \( a \) and Policy 2 recommended \( b \). To compare Policies 1 and 2 for this segment of customers we can compare the outcome for the 13,038 customers that received treatment \( a \) with the 11,495 customers that received treatment \( b \). Because the treatment these customers received was randomly allocated, we can safely compare the outcomes across these two treatment groups without concern for customer differences. An overall comparison of the two policies just requires aggregating these segment-level comparisons across segments.
Table 4.1 An Example of Segmentation by Recommended Action

<table>
<thead>
<tr>
<th>Segment Label</th>
<th>Recommended Action</th>
<th>Sample Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Policy 1</td>
<td>Policy 2</td>
</tr>
<tr>
<td>Segment_aa</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>Segment_ab</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>Segment_ac</td>
<td>a</td>
<td>c</td>
</tr>
<tr>
<td>Segment_ba</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>Segment_bb</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>Segment_bc</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>Segment_ca</td>
<td>c</td>
<td>a</td>
</tr>
<tr>
<td>Segment_cb</td>
<td>c</td>
<td>b</td>
</tr>
<tr>
<td>Segment_cc</td>
<td>c</td>
<td>c</td>
</tr>
</tbody>
</table>

The table reports the sample sizes from an experiment reported by Simester, Timoshenko and Zoumpoulis (2018). It reports the sample size in the three experimental conditions associated with actions a, b and c. The shading identifies the sample that can be used to evaluate Policy 1. The average profits per segment and per experimental condition are reported in the Appendix.

We make several comments. First, Policies 1 and 2 could be any two policies, even arbitrary policies or policies designed after the experiment was implemented. This comparison is agnostic to the nature or source of the policies.

Second, the proposed segmentation using the recommended actions from each policy allows for convenient evaluation of a single policy or comparison of policies. This is because within a segment, each policy is associated with a single action. If instead of segmenting customers by the recommended actions, we segment by a covariate, then evaluation and comparison would no longer be as simple. For example, remember our earlier male – female example. The two targeting policies (Policy 1 and 2) both recommend sending promotion a to male customers. The policies differ in their recommendations for female customers. Policy 1 recommends sending promotion b to the female customers, while Policy 2 recommends sending promotion c to female customers. Imagine that we want to evaluate Policy 1 and, instead of segmenting by gender, we segmented customers by age: young and old. Within the young segment, Policy 1 recommends action a for young male customers and action b for young female customers. As a result, to evaluate Policy 1 within the young segment, we would need to use the response from some young customers who were randomly allocated to receive action a and other young customers who were randomly allocated to receive action b. In contrast, segmenting by recommended actions ensures that, within a segment, each policy is associated with a single action, and we can evaluate the policy using the responses from all customers randomly allocated to receive that action.

Third, because the actions are randomly assigned, every treatment should be received by a random sample of customers in every segment. If a combination of recommended actions on one of the segments arises
infrequently, the sampling plan (in Step 1) may need to be adjusted to ensure that every segment – action combination has sufficient sample to provide reliable estimates.

A limitation of this approach is that segmenting customers by recommended actions may divide the sample into impractically many segments. We discuss this issue in Section 8.

**Blocking versus Post-Stratification**

The causal inference literature distinguishes blocking from post-stratification, and it is helpful to describe how both terms apply to the first two steps of our proposed program. Under both blocking and post-stratification, segments are constructed using pre-randomization covariates. The researcher segments observations, estimates treatment effects within each segment, and then uses a weighted average of these segment estimates to calculate an average treatment effect.\(^6\) The difference between the two approaches is the timing of the segmentation. Under blocking, segmentation is used in the experimental design stage, before assignment of the treatments. The researcher first stratifies (segments) the sample. The researcher then decides on counts per experimental condition for each stratum, and randomly assigns treatments within each segment, respecting the pre-determined counts per experimental condition. Under post-stratification, segmentation occurs after treatments are assigned. This means that in post-stratification, the researcher first decides on counts for each experimental condition across the entire sample (but not per segment), and randomly assigns treatments in the entire sample, respecting these pre-determined counts. The researcher then stratifies. As long as there is heterogeneity in treatment effects across units, and the individual-level treatment effect is correlated with the segmentation variables, then post-stratification will improve efficiency (compared to no stratification).

The proposed segmentation in Step 2 of our program represents an example of post-stratification. The recommended actions from each policy are obtained using pre-treatment data, and these recommended actions serve as the stratification variables. The segmentation by recommended actions occurs after actions are randomly assigned. As we discuss in Section 5, treatment effects are estimated for each stratum (segment), and then aggregated to calculate an average treatment effect. The requirements that treatment effects are heterogeneous and correlated with the stratification variables are clearly met.

Targeting policies recommend different actions to different customers because of the heterogeneity in the response. Segmenting customers by recommended actions groups customers according to their responses, leading to efficient estimation.

As an alternative, the proposed program could employ blocking. To do so requires reversing Steps 1 and 2. Start by logging what action each policy recommends for each customer (counterfactual policy logging) and use these recommended actions to stratify the customers into segments. Then pre-determine the counts for each action in each segment, and randomly assign actions to customers within each segment according to the pre-determined counts. Notice that the stratification occurs before the random assignment.

A key distinction between blocking and post-stratification is that in post-stratification the number of customers allocated to each experimental condition in each stratum becomes a random variable. This additional source of variation makes post-stratification less efficient than blocking. Blocking also offers an additional benefit, which we will discuss in Section 6, where we formally investigate why

---

\(^6\) See Miratrix, Sekhon and Yu (2013) for helpful discussions of both definitions, and Berman and Feit (2018) for a recent application of post-stratification in marketing.
segmentation improves efficiency. Miratrix, Sekhon and Yu (2013) compare the two approaches and show that the difference in efficiency is generally small. We present our program with randomization before stratification, and we emphasize that blocking in the experimentation stage is not required in order to ensure the efficiency benefits of our approach.

We next describe how to use the results of the randomized-by-action field experiment (Step 1) and the segmentation of customers according to the actions recommended by each policy (Step 2) to evaluate and compare the performance of two targeting policies (Step 3).

5. Using OLS and Double Machine Learning to Evaluate Policies and their Difference (Step 3)

Having segmented customers using the recommended actions, we can directly evaluate the performance of any policy, as well as compare the performance of candidate policies. We will again illustrate using the STZ study, and explain how to evaluate and compare Policy 1 and Policy 2 in this example. We start with two preliminary observations about identifying weights to aggregate effects across segments, and describing the construction of estimation samples for evaluating each policy. We begin with a brief note about weights.

Throughout this section we will recommend calculating average treatment effects by first calculating conditional treatment effects within each segment, and then calculating a weighted average across segments. The segments are obtained in Step 2 using the recommended actions from pairs of candidate policies. We assume that the researcher knows what weights to use. Counterfactual policy logging can be useful for constructing these weights. For example, in the STZ study, the firm wanted to use the findings to select a target policy to implement on a larger implementation sample. The researcher can ask what action each policy would recommend for each member of the implementation sample. The intersection of these recommended actions reveals which segment each member of the implementation sample is in, which in turn reveals the weight of each segment. We provide explicit expressions for these weights in the Appendix.

Our second preliminary observation involves the construction of a sample of observations to evaluate each policy. Step 1 of our program produces randomized-by-action data. Our method in Step 3 estimates the performance of policies and comparisons between policies using this randomized-by-action data. Let Policy 1 Dataset contain all of the observations/customers that were randomly assigned to receive the same treatment as the one recommended by Policy 1. This is the set of customers highlighted with shading in Table 4.1. We define the Policy 2 Dataset similarly. Notice that in segments in which both policies recommend the same action, the Policy 1 Dataset and the Policy 2 Dataset will both include the same customers. For example, in Segment_aa, both datasets will contain the 595 customers that were randomly assigned to receive action a.

The simplest approach to evaluate Policy 1 is to calculate the average response across the observations in the Policy 1 Dataset. We can do the same with Policy 2 using the Policy 2 Dataset. The difference in the average response across the two datasets provides an unbiased estimate of the difference in the performance of the two policies (the average treatment effect), for which standard errors can be obtained by bootstrapping. In this section we propose three ways to improve the efficiency of this estimate: (a) segmenting the data using the recommended actions and conducting the analysis separately by segment, (b) introducing additional covariates to control for customer heterogeneity, and (c) using state-of-the-art
methods (double machine learning) to address overfitting and regularization biases. We start with segmentation.

**Conducting the Analysis Separately by Segment**

In the previous section we described how to construct segments using the recommended actions from pairs of policies. Using the STZ example, we can use this segmentation approach to identify nine segments in the *Policy 1 Dataset* and the corresponding nine segments in the *Policy 2 Dataset*. Instead of calculating the overall average response for each policy and then calculating the differences in these overall average responses, it is more efficient to calculate the average response within a segment for each policy. The within segment difference between the two policies in the average responses represents a conditional treatment effect. We can then construct an average treatment effect by calculating a (weighted) average of these within-segment differences.

The efficiency benefits from segmentation occur for two reasons. The first reason is that comparing the difference in outcomes between policies within a segment helps to eliminate between-segment differences. For example, if the outcomes for males are systematically different from the outcomes for females, then calculating an aggregate outcome for each policy will include variation introduced by gender. As a result, when comparing the average outcome between policies, the standard error of the difference will also include variance introduced by gender differences. We eliminate this variation if instead we compare the difference in outcomes within a segment (where the gender remains the same), and then aggregate these within-segment differences. The differencing within segment ensures we only compare males with males, and females with females. Although this is the standard benefit of stratification when analyzing randomized experiments (see, for example, Imbens and Rubin, 2015; Athey and Imbens, 2017), we note that our special choice of segments (using recommended actions) leads naturally to segments of customers that are homogeneous with respect to their treatment effects. This ensures that the variance within segments is smaller than the variance across segments, and thus leads to efficiency benefits in estimation.

The second efficiency advantage begins with the insight that in segments for which two policies recommend the same action, the *true difference* in the performance of the policies is precisely zero (because the policies are identical for this segment). Conducting the analysis at the segment level ensures that the *observed difference* in the performance of the two policies will also be zero in these segments, because the two policies are evaluated using exactly the same observations. This is one of the core ideas of our paper.

In contrast, calculating the difference in the overall average performance of two policies (without segmentation) will generally not recognize that there is no difference in the performance of two policies when they both recommend the same action. This is true both when we introduce additional covariates (see discussion below), but also without additional covariates. We illustrate why using a simple example. Assume that there are two policies and two segments: for males the policies recommend the same action and for females they recommend different actions. There are 50 males who receive the action recommended by the two policies and the average response across these 50 males equals $1. Assume that randomizing by action results in 49 females who receive the recommended action from one of the policies with an average response of $2 and 51 females who receive the recommended action from the other policy, with an average response of $3. The average overall response across the 99 observations associated with the first policy is \((50 \times 1 + 98 \times 3) / 99 = 1.495\). For the second policy, the average overall
response is ($50 + $153) / 101 = $2.010, yielding an average treatment effect of $0.515. In contrast, if we conduct the analysis by segment, then we have zero difference in the male segment and a difference of $1 per observation in the female segment. If we weight males and females equally, conducting the analysis by segment yields an average treatment effect of $0.500. Both are unbiased estimates, but the analysis by segment is more efficient because it recognizes that there is no performance difference between the two policies for males. Without segmentation, random variation in sample sizes associated with each action may mean that the observed difference in the performance of the two policies in segments where the two policies recommend the same actions is not zero. This introduces a source of random error. Of course, we could fix this issue by re-weighting male and female observations, but this would require segmentation.

This benefit of segmentation will be particularly important when comparing two policies that are more similar. The more similar the policies, the larger the segments for which the two policies recommend the same action, and for which the true difference in the performance of the two policies is precisely zero.

Using segmentation to identify customers for whom two policies recommend the same action also has an additional implication, related to experimental design. An experiment provides no information about the difference in the performance of two policies in segments in which the two policies recommend the same action. Therefore, if we are only interested in comparing the relative performance of candidate targeting policies (rather than evaluating the absolute performance of any single policy), we can omit these segments entirely from the experimental design. We can thus save implementation costs with no loss of information about the relative performance of the policies. For example, in our earlier male – female example, we can omit mailing to male customers with no loss of information about relative performance of the two policies. We caution that if we omit male customers, we can no longer calculate the absolute outcome across all customers for any single policy. For this reason, instead of omitting male customers, it may be preferable to simply under-sample male customers.

Notice that the first efficiency advantage from segmentation (reducing variance from between-segment differences) occurs when comparing any two experimental conditions; the experiment need not compare two targeting policies. It also does not require segmentation using recommend actions from targeting policies. In contrast, the second efficiency benefit relies upon the insight that in segments for which two policies recommend the same action, we know the true difference in the performance of the policies is precisely zero. As a result, this second efficiency advantage is specific to the comparison of two targeting policies, and requires that the segmentation be constructed using the recommended actions (Step 2).

For the sake of clarity, for the STZ study we recommended conducting the analysis separately in all nine segments. However, it should now be clear that there is no need to estimate anything in the three segments in which two policies recommend the same action, because the performance difference is always zero in these segments. We will also later consider variants in which we pool across segments when estimating effects in the other six segments (for which the recommended actions are different).

The advantages of segmentation may become even stronger when using additional covariates. We discuss this approach next.

**Introducing Additional Covariates**

Conducting the analysis within segments to reduce between-segment differences is not the only way to remove variance due to customer heterogeneity. A standard alternative approach is to introduce additional covariates. For example, consider the following OLS model:
\[ Y_j = \alpha_{st} + \beta_{st} \text{Covariates}_j + \varepsilon_j. \]  (5.1)

The unit of analysis is a customer \((j)\) in a randomized-by-action experiment. The dependent variable \(Y\) measures the outcome of the experiment for each customer (such as profit). To compare two policies we estimate Equation 5.1 separately for each policy and each segment, except the segments in which the two policies recommend the same actions. For example, to compare Policy 1 and Policy 2, we estimate the model using the Policy 1 Dataset six times, once for each of the segments in which the two policies recommend different actions. We do the same for the Policy 2 Dataset. The subscripts for the coefficients \((s,t)\) recognize that we obtain separate estimates for each segment Segment\(_{st}\), where \(s,t\) index the actions recommended by Policy 1 and Policy 2 respectively.

The Covariates, term could include multiple additional covariates, transformations of additional covariates, or even interactions of additional covariates. If the covariates are standardized (zero mean and unit standard deviation) then coefficient \(\alpha_{st}\) estimates the average performance of Policy 1 in Segment\(_{st}\) when each of the standardized covariates is zero (\(\alpha_{st}\) equals the mean of \(Y_j\) within Segment\(_{st}\)). The vector of coefficients \(\beta_{st}\) estimates the change in the outcome when the (standardized) covariates increase by one unit. \(\varepsilon_j\) is an error term.

Having estimated the \(\alpha_{st}\) coefficients by segment for each policy, we can calculate conditional treatment effects using the difference in these coefficients within each segment. In segments in which the two policies recommend the same action (Segment\(_{aa}\), Segment\(_{bb}\), Segment\(_{cc}\)), we do not need to estimate \(\alpha_{st}\) coefficients from Equation 5.1. Within these segments the coefficients will be identical for the two policies because both policies use the same data in these segments. As a result, our approach correctly recognizes that the difference in performance is exactly zero in these segments, with zero variance.

In contrast, an OLS-based approach that incorporates covariates, but estimates a single model for each policy (without segmentation), would not recognize that the performance difference in these segments is zero. This arises even though the two policies use the same data within these segments. Without segmentation, the covariate coefficients will be different in the Policy 1 model than in the Policy 2 model. The predicted responses from the two models will therefore not be identical in segments in which the two models recommend the same actions.

As an alternative to estimating Equation 5.1 separately for each (policy, segment) pair, and then calculating the difference between the \(\alpha_{st}\) coefficients for each policy, we can instead estimate a single model for each segment. In particular, if we pool the data from the Policy 1 Dataset and Policy 2 Dataset, we can estimate the following model within each segment:

\[ Y_j = \alpha_{st} + \beta_1 \text{Policy 1}_j + \beta_{st} \text{Covariates}_j + \varepsilon_j. \]  (5.2)

In this model, Policy 1 is an indicator variable identifying observations from the Policy 1 Dataset. The coefficient for this variable (\(\beta_1\)) provides a conditional treatment effect within that segment, estimating the increase (decrease) in the response variable \((Y)\) in Policy 1 compared to Policy 2. The difference
between the Equation 5.1 and 5.2 approaches is that the Equation 5.2 approach requires only a single estimate per segment, and it restricts the coefficients on the covariates ($\beta_{st}$) to be identical for the two policies, so that the estimated impact of the covariates on the response is the same for the two policies.

To calculate an average treatment effect across all of the segments, we either calculate a weighted average of the difference between the $\alpha_{st}$ coefficients in each segment (if using Equation 5.1), or a weighted average of the $\beta_1$ coefficients (if using Equation 5.2). When calculating this weighted average we cannot ignore the three segments in which the policies recommend the same actions. We know that the difference in performance in these three segments is zero, and so omitting these segments would result in positive bias in the absolute magnitude of the average treatment effect.

Because the observations are independent across segments, the variance of the average treatment effect can be obtained by aggregating the variances across segments. Efficient estimates of the variance of the performance differences can also be obtained through bootstrapping. If using the model in Equation 5.1, notice that in segments in which the two policies recommend different actions, a generally conservative (i.e., upwardly biased) estimate of the variance of the estimated difference between the policies is the sum of the variance of the $\alpha_{st}$ coefficients for each policy (we further explain the variance estimator in the Appendix). If using the model in Equation 5.2, for segments in which the two policies recommend different actions, the variance is the regression variance of the $\beta_1$ coefficient. In segments in which the two policies recommend the same actions, not only is the performance difference zero, the variance in the performance difference is also zero.

If we use a standard OLS estimator, we can also use standard regression techniques to improve the estimates of the coefficients or their standard errors. For example, recall that the households in the STZ study were grouped into carrier routes, and the treatments were randomized at the carrier route level. If we believe the errors are correlated across households within the same carrier route, we can cluster the standard errors at the carrier route level. We can also use Eicker-Huber-White standard errors (Eicker, 1967; Huber, 1967; White, 1980) to correct for heteroscedasticity.

The third way to improve efficiency draws on recent innovations designed to reduce the limitations of OLS and other estimation methods.

**Double Machine Learning**

The traditional OLS estimation is not only limited in terms of its modeling ability, but also susceptible to overfitting. For these reasons we recommend using the “double machine learning” approach (Chernozhukov et al., 2018). Double machine learning reduces the regularization and overfitting biases, and allows the improved use of a very broad set of machine learning methods.

We now present in some detail a double machine learning formulation for estimating and comparing policies, while a more precise description of the algorithm is included in the Appendix. Following the standard double machine learning setup (Chernozhukov et al., 2018), we model the response measure as a (possibly complex) function of the covariates. We do this in each segment for which the policies recommend different actions, and separately for each of the two policies. In particular, we model the response to Policy 1 in Segment_{st} by:

---

7 A neat description of the standard algorithm for double machine learning, which we adjust for our needs, can be found in Chernozhukov et al. (2017).
\[ Y_j = g_1^{st}(\text{Covariates}_j) + \zeta_j, \]

where the unit of analysis is an experimental participant (j), and errors \( \zeta \) have mean zero, conditioned on the covariates. We model the response to Policy 2 similarly, using function \( g_2^{st}(\text{Covariates}_j) \).

For each segment where the two policies recommend different actions, we split the observations (customers) in \( k \) equal folds. We estimate functions \( g_1^{st} \) and \( g_2^{st} \) on data from \( k-1 \) of the folds, using a machine learning method, such as Lasso or a simple linear regression (OLS). We then calculate the performance of each of the two policies, as well as their difference, on the left-out fold. To estimate the performance of Policy 1, we employ the following doubly robust estimator:

\[
\hat{Y}_1^{st} = \frac{1}{|\text{customers in the leftout fold}|} \sum_j \left[ g_1^{st}(\text{Covariates}_j) + \frac{1(\text{Policy 1 assigns what } j \text{ got})}{\hat{p}(\text{action that } j \text{ got})} (Y_j - g_1^{st}(\text{Covariates}_j)) \right],
\]

(5.3)

where \( \hat{Y}_1^{st} \) denotes our estimate of the performance of Policy 1 in Segment \( st \), with \( s \neq t \). The doubly robust estimator uses IPS (inverse propensity score) weighting, which scales using an estimate \( \hat{p} \) of the probability that unit \( j \) was randomized in the Step 1 data to the action Policy 1 would assign to it. This scaling adjustment corrects for the fact that unit \( j \) is assigned in Step 1 the same action that Policy 1 would assign to it only with some probability, and not certainly. Overall, the second term inside the summation corrects for function \( g_1^{st} \) not perfectly approximating \( Y_j \), and so the term interpolates the approximation error from the observations where the error is known to the complete sample.

We employ an equivalent estimator to produce an estimate for the performance of Policy 2 in Segment \( st \). We then subtract the estimates to produce an estimate of the difference between the two policies.

For each segment where the two policies recommend different actions, we repeat this process across all the \( k \) folds. To derive an average estimate of the difference for each segment, we average the estimates from each of the \( k \) folds. To derive an estimate of the standard error for each segment, we use, for each unit, the deviation between the estimated treatment effect (from the left-out fold) and the difference in the doubly robust estimates for the performance of each policy.

In segments in which the two policies recommend the same action (Segment \( aa \), Segment \( bb \), Segment \( cc \)), we force the difference between the policies to be zero. As a result, our approach correctly recognizes that the difference in performance is exactly zero in these segments, with zero variance.

Having produced estimates of the difference and standard errors in each of the segments, we then aggregate across all segments to produce the average treatment effect and standard error.

There are two key ingredients to this state-of-the-art approach. First, the use of doubly robust scores (Robins and Rotnitzky, 1995) ensures that the inference of the treatment effect is sufficiently insensitive to small mistakes in the estimation of the functions \( g \), which capture how the outcome depends on the covariates.\(^8\) This ensures that high-quality inference for the treatment effect may still be obtained in the

\(^8\) Chernozhukov et al. (2017) explain that doubly robust scores are automatically Neyman orthogonal, which ensures robustness to mistakes in the estimation of the nuisance functions.
Robustness of the inference of the treatment effect to small mistakes in the estimation of functions \( g \) is crucial when the estimators of functions \( g \) are based on machine learning methods. The second key ingredient of the double machine learning approach is \( k \)-fold cross-fitting, which helps eliminate overfitting, which can easily occur in the application of machine learning methods.

**Summary**

The simplest estimate of the difference in performance of two policies is the difference between the average overall response to one policy and the average overall response to the second policy. We propose three ways to improve the efficiency of this estimate. One of the improvements is standard; add covariates to reduce variance due to customer heterogeneity. Another improvement is obtained by using the state-of-the-art double machine learning estimator to reduce regularization and overfitting biases. Although both the double machine learning estimator, and the doubly robust estimator on which it is based, are new to marketing, they are now becoming standard approaches in other fields. The third improvement is novel, and specific to the comparison of targeting methods. It begins with the insight that there is no difference in the performance of two targeting policies among customers for which two policies recommend the same action. We can identify these customers by simply segmenting customers using the recommended actions (Step 2). Setting the performance difference to zero for these customers, and aggregating this (zero) difference with the performance difference for the remaining customers, improves efficiency by removing random error.

In Section 7, we evaluate the efficiency improvements using the STZ data. We implement double machine learning using both an OLS estimator and Lasso. We focus on decomposing the increased efficiency available from (a) segmentation using counterfactual policy logging, (b) controlling for heterogeneity using additional covariates, and (c) controlling for overfitting and regularization biases using double machine learning. Before presenting these empirical results, we first present a formal investigation of the efficiency advantages that segmentation offers.

### 6. Segmentation Improves Efficiency

We have argued that conducting the analysis by segment has two efficiency advantages. First, it recognizes that among customers for which two policies recommend the same action, the true performance difference between the policies is zero. Second, it reduces variance introduced by between-segment differences. We can show analytically that the standard errors of the estimators strictly improve when using segmentation, as compared to the standard approach of calculating the difference in the aggregate performance of policies.

Our argument requires two assumptions. The first assumption is that the experimental design is balanced; in a randomized-by-policy design, the proportion of customers assigned to a policy within a segment is the same as the proportion of customers assigned to that policy in any other segment (and the same is true for all policies we are evaluating). We can ensure that this assumption is satisfied through blocking. Recall that blocking is an experimental design in which customers are first segmented; the researcher then decides on counts per experimental condition for each segment, and randomly assigns experimental

---

9 As we note in the Introduction, Johnson, Lewis and Reiley (2017), and Johnson, Lewis and Nubbemeyer (2017) propose a related method to improve the efficiency of estimating the average treatment effects by removing from both treatment and control conditions customers for whom an intention to treat would fail.
conditions within each segment, respecting the pre-determined counts. Blocking can be used to ensure that the proportion of observations assigned to each condition is identical across each segment. Notice that this is not possible with post-stratification, because stratification occurs after the random assignment.

The second assumption is that for all policies we are evaluating, the observed variance within any segment that received the policy is not larger than the observed variance across all observations (in any segment) in the condition that received that policy. This assumption is aligned with the well-known intended benefit of stratification (e.g., Imbens and Rubin, 2015): we segment in order to achieve balance in the covariates. This means that the units within each segment would be similar with respect to the covariates or some functions of the covariates. As noted above, our special choice of segments (using policies’ recommended actions) leads naturally to segments of customers that are homogeneous. It therefore makes sense to expect that within any segment, the observed variance in outcomes is not larger than in the aggregate.

Theorem 1 in the Appendix formalizes our result that the proposed approach improves efficiency (when both the proposed and the standard approach use randomized-by-policy data). Our proof compares Steps 2 and 3 of our proposed program with the traditional approach of calculating the average aggregate performance of each method, and then calculating the difference between methods in the average performances. In the proof, we assume that both evaluation methods have access to the same randomized-by-policy data, and so the proof does not depend upon Step 1 in our proposed program. Recall that the benefit of using a randomized-by-action design is that it allows off-policy evaluation of any targeting policy. This comes as an additional advantage of our overall program, supplementing the efficiency advantages in policy comparison discussed in this section.

**Limitations of the Proposed Approach for Efficiency**

We have highlighted how segmenting using the actions recommended by candidate policies can lead to more efficient performance comparisons. Theorem 1 in the Appendix establishes that, under our assumptions, the estimated variance under the proposed approach will be smaller than the estimated variance under the standard approach. However, we recognize that there are limitations to this approach when the assumptions of Theorem 1 about balanced experimental design and relatively small within-segment variances are violated. The most prominent limitation is that, if the between-segment variation is small, this approach could theoretically lower precision due to small sample effects.

Imbens and Rubin (2015) and Athey and Imbens (2017) argue that, in expectation, the variance under stratification (i.e., segmentation) cannot be larger than the variance without stratification, despite the small samples. This means that there is no cost to segmentation in terms of the variance itself; nevertheless, there is a cost in terms of estimation of the variance. The variance with segmentation is less than or equal to the variance without segmentation. Assuming unbiased estimators for the variance, the expectation of the estimated variance with segmentation will therefore be less than or equal to the expectation of the estimated variance without segmentation. However, it is the variance of the estimator of the variance that can be larger with segmentation than without segmentation. The reduction of the

---

10 For example, if a segment contains 100 customers and we want to randomly assign 50 customers to a treatment condition and 50 to a control condition, we can create 50 (virtual) tokens and randomly distribute them across the 100 observations. We can then assign the treatment to the 50 customers that receive the tokens. We can repeat this for all segments, ensuring that half the customers are assigned the treatment in each segment.
degrees of freedom due to smaller sample sizes may result in cases where the estimated variance with segmentation is larger than the estimated variance without segmentation.

In practice, because the segments are constructed using the recommended actions, the adjustment in degrees of freedom due to small samples will generally be outweighed by the benefits of aggregating within-segment differences, to remove between-segment variation.

Summary
The segmentation approach (Steps 2 and 3) provides a more efficient method for comparing two policies than the standard approach of simply calculating the difference in the aggregate performance of each policy. More specifically, when holding the total number of participants fixed, we obtain a more accurate estimate of the performance difference. Alternatively, we can obtain the same level of precision using fewer participants. We show theoretically that our proposed method strictly improves the standard error both when evaluating the performance of a single policy and when comparing the performance of two policies.

In the next section we use the data from the STZ study to illustrate the efficiency advantages offered by the proposed approach. We will show empirically that the efficiency improvements can be large; our proposed method reduces the standard error by more than 50%.

7. Using the STZ Study to Illustrate the Efficiency Advantages
In Section 5 we proposed three ways to improve efficiency when comparing the performance of two targeting policies: segmentation, introducing additional covariates, and double machine learning. In this section we use data from the STZ study to illustrate the efficiency improvements contributed by each of these recommendations.

In particular, we compare the performance of Policy 1 and Policy 2 in the STZ study using data from each of the experimental designs. We also re-estimate the findings using different estimation methods, and different segmentation approaches. We start by describing how we construct estimation samples from the two experimental designs, before describing the estimation approaches, and different levels of segmentation.

Data From Two Experimental Designs
Steps 2 and 3 of our proposed program apply equally to a randomized-by-action and a randomized-by-policy design, and the STZ study provides an ideal dataset to illustrate these two applications. Recall that the STZ study implemented both types of designs. In seven of the experimental conditions, customers were assigned to a targeting policy, and received the action that the policy recommended for them (a randomized-by-policy design). In three of the experimental conditions, customers were assigned to an action uniformly (a randomized-by-action design).

To compare Policy 1 and Policy 2 using the randomized-by-action experimental data, we use the experimental conditions associated with the three actions. We first construct the Policy 1 Dataset using the customers that were randomly assigned to receive the same action as the one recommended for them by Policy 1. This is the set of customers highlighted with shading in Table 4.1. We then use the same approach to construct the Policy 2 Dataset. Having constructed these two datasets we pool them to obtain a single estimation dataset for this experimental design.
To compare Policy 1 and Policy 2 using the randomized-by-policy experimental design, we use the experimental conditions associated with Policy 1 and Policy 2. When using data from a randomized-by-policy design, the Policy 1 Dataset is simply the customers in the experimental condition associated with Policy 1 (and similarly for Policy 2). These datasets are again pooled to obtain a single estimation dataset (for this experimental design).

Four Estimation Approaches

We investigate four different estimation approaches:

- **Simple OLS**
  
  Use OLS to estimate: \( Y_j = \alpha + \beta_1 Policy_1 + \epsilon_j \).

- **OLS with Covariates**
  
  Use OLS to estimate: \( Y_j = \alpha + \beta_1 Policy_1 + \beta Covariates_j + \epsilon_j \).

- **Double ML OLS**
  
  Use double ML with OLS to estimate the response to Policy \( i \) as \( Y_j = g_i(Covariates_j) + \zeta_j \).

- **Double ML Lasso**
  
  Use double ML with Lasso to estimate the response to Policy \( i \) as \( Y_j = g_i(Covariates_j) + \zeta_j \).

The Policy 1 variable is a binary indicator flagging observations from the Policy 1 Dataset (recall that the estimation sample is the union of the Policy 1 Dataset and the Policy 2 Dataset). The dependent variable in all of these models measures profits. The covariates include the full set of thirteen covariates that the firm that sponsored the experiment used in its standard targeting model (the variables are listed and defined in the Appendix). These thirteen covariates were also used to train Policy 1 and Policy 2 using a separate dataset. A more complete description of the outcome measure, together with summary statistics for the covariates are provided in the STZ paper.

The simple OLS approach is equivalent to directly calculating sample means for each policy, and then taking the difference. In particular, the coefficient of the Policy 1 variable provides an estimate of the difference in performance between the two policies.11

The OLS with covariates approach uses Equation 5.2 to estimate the difference in performance between the two policies. As we discussed in Section 5, the performance difference (Policy 1 - Policy 2) is measured by the coefficient on the Policy 1 indicator variable (\( \beta_1 \)). The OLS and Lasso double ML approaches use double machine learning, where the response functions are estimated using either OLS or Lasso.

All of the methods produce unbiased estimates of the difference in performance between the two policies. For this reason, our focus is not on the estimates of the performance differences. Instead, we focus on the efficiency of these estimates, which is measured by their standard errors. Comparing simple OLS with OLS with covariates allows us to evaluate the efficiency improvement from introducing additional covariates. Comparing OLS with covariates with the two double machine learning approaches reveals the incremental efficiency improvement offered by using a sophisticated approach to control for overfitting and regularization biases.

11 The standard errors from OLS are different from the standard errors directly calculated from the sample. The reason is that the regression model estimates a combined, pooled estimate of the variance across groups, assuming homoscedasticity. Applying OLS with heteroscedasticity-consistent (Eicker-Huber-White) standard errors essentially corrects for this discrepancy.
Two Segmentation Approaches

Recall that we can use counterfactual policy logging to generate nine segments describing the recommended actions from Policy 1 and Policy 2 (Step 2). We contrast two approaches to using this segmentation:

**No Segmentation**  
Pooling across all nine segments.

**Partial Segmentation**  
Pooling the six segments in which the policies recommend different actions, and setting the performance difference to zero in the three segments in which they recommend the same action.

In the partial segmentation approach we calculate the overall performance difference as a weighted average of the estimated performance difference in the six segments, and the zero performance difference in the other three segments.\(^{12}\) Comparing the standard errors from this approach with the standard errors from the no segmentation approach provides a measure of the efficiency gains from segmentation.

We also considered the full segmentation approach, in which we estimated the models separately in each of the six segments in which the policies recommend different actions, and set the performance difference to zero in the three segments in which they recommend the same action. However, there was too little data in the STZ study in some segments to estimate some of the models. We will later compare the partial segmentation variance estimates with a full segmentation approach that analyzes the performance differences separately in all nine segments.

**Results**

We repeat the analysis at two different levels of aggregation. Although we observe outcomes at the customer level, in the STZ study the randomization was conducted at the carrier route level (groups of approximately 400 geographically-clustered households). Moreover, the covariates are all identified at the carrier route level; the values on the covariates do not differ across customers within the same carrier route.\(^{13}\) For this reason, we repeated the analysis both using an individual customer as the unit of analysis, and aggregating up to the carrier route as the unit of analysis. We report the findings in Table 7.1 at the carrier route level, and report the individual customer level findings in the Appendix. The pattern of findings is consistent using the two sets of analysis.

In Table 7.1 we restrict attention to standard errors, and relegate the estimates of the performance differences between Policy 1 and Policy 2 to the Appendix. The standard errors from OLS are the analytical standard errors, adjusted for heteroscedasticity using the Eicker-Huber-White adjustment. This has little impact on the pattern of findings (we report unadjusted standard errors in the Appendix).

The findings confirm the efficiency benefits of segmentation. For each estimation method we see that segmentation reduces the standard errors by over 50%, for both experimental designs. We conclude that when comparing targeting policies, identifying segments of customers for which there is no true difference in the performance of the two targeting policies can greatly improve efficiency.

\(^{12}\) The weights use the size of the segments when pooling across the three randomized-by-action experimental conditions.

\(^{13}\) This occurs because the STZ study targets prospective customers. If the study targeted existing customers, the covariates would include measures of past purchasing for each customer.
In the absence of segmentation, adding covariates to OLS also improves efficiency; the standard errors reduce from $0.101 to $0.086 (randomized-by-action data) and from $0.107 to $0.090 (randomized-by-policy data). This is consistent with evidence elsewhere in the literature that including additional covariates can improve efficiency by controlling for customer heterogeneity.

Table 7.1 Standard Errors Using Different Estimation Methods, Different Levels of Segmentation, and Different Experimental Designs

<table>
<thead>
<tr>
<th>Data Source</th>
<th>Estimation Method</th>
<th>No Segmentation</th>
<th>Partial Segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Randomized-by-Action</td>
<td>Simple OLS</td>
<td>$0.101</td>
<td>$0.045</td>
</tr>
<tr>
<td></td>
<td>OLS with Covariates</td>
<td>$0.086</td>
<td>$0.039</td>
</tr>
<tr>
<td></td>
<td>OLS Double ML</td>
<td>$0.086</td>
<td>$0.040</td>
</tr>
<tr>
<td></td>
<td>Lasso Double ML</td>
<td>$0.087</td>
<td>$0.039</td>
</tr>
<tr>
<td>Randomized-by-Policy</td>
<td>Simple OLS</td>
<td>$0.107</td>
<td>$0.047</td>
</tr>
<tr>
<td></td>
<td>OLS with Covariates</td>
<td>$0.090</td>
<td>$0.040</td>
</tr>
<tr>
<td></td>
<td>OLS Double ML</td>
<td>$0.090</td>
<td>$0.041</td>
</tr>
<tr>
<td></td>
<td>Lasso Double ML</td>
<td>$0.091</td>
<td>$0.041</td>
</tr>
</tbody>
</table>

Note: The table reports standard errors measuring the difference in the performance of Policy 1 and Policy 2 in the STZ study under different estimation approaches. The data is aggregated to the carrier route level (the unit of analysis is a carrier route). The standard errors are adjusted for heteroscedasticity using the Eicker-Huber-White adjustment. The mean differences under each approach together with unadjusted standard errors are reported in the Appendix. Results at the individual level are also reported in the Appendix. To preserve confidentiality, the profits are multiplied by a common random number.

The efficiency gains from segmentation are much larger than the efficiency gains from introducing additional covariates. When using OLS, segmentation reduces the standard errors by $0.056 and $0.060 in the randomized-by-action and randomized-by-policy datasets (respectively). In contrast, introducing additional covariates (without segmentation) lowers the standard errors by just $0.015 and $0.017. This finding mirrors results in Johnson, Lewis and Reiley (2017). They report that using counterfactual policy logging to exclude customers who are not exposed to an ad improves the precision of their estimate of ad lift by 31%. In contrast, adding covariates improved precision by just 5%.

It is also notable that the efficiency improvement from introducing additional covariates is much smaller when using segmentation. This is because segmentation is itself a way to control for customer heterogeneity. This is particularly true when the candidate policies are trained using the same set of covariates (as in this case). The recommended actions from the candidate policies can be thought of as a summary of the information contained in the covariates about differences in how customers will respond to each action. For this reason, segmenting customers using counterfactual policy logging controls for customer heterogeneity by dividing customers into segments of customers that are expected to respond in
similar ways. Later in this section we will directly compare the efficiency improvements from these two approaches to control for customer heterogeneity (adding covariates or analyzing by segment).

Double machine learning does not yield any incremental efficiency improvement over OLS with covariates. This suggests that overfitting and regularization biases in OLS are not important factors in this data. While this is reassuring, we caution that it does not mean these are never important factors when evaluating targeting policies using experimental data.\(^{14}\)

The randomized-by-policy findings are very similar to the randomized-by-action results. However, there are two comparisons of interest. First, we might expect lower standard errors when comparing two policies (without segmentation) using randomized-by-action data versus using randomized-by-policy data. When two policies recommend the same actions, we evaluate them using (exactly) the same data in the randomized-by-action setting, but we use different data in the randomized-by-policy setting.\(^ {15}\) This introduces an additional source of random noise in the randomized-by-policy data. Comparing the standard errors from simple OLS without segmentation, we do see slightly lower standard errors when using the randomized-by-action data ($0.101$) compared to randomized-by-policy data ($0.107$). This pattern is consistent across all four estimation methods, although the differences are very small.

The second comparison is also motivated by the same issue. The benefits of segmentation are higher when using randomized-by-policy data than when using randomized-by-action data. This is again because we do not use identical data to evaluate the policies in a randomized-by-policy dataset. This introduces greater potential for random noise, which segmentation removes. If we compare the simple OLS rows we do see evidence that segmentation leads to a slightly larger reduction in standard errors in the randomized-by-policy data. Segmentation lowers the standard errors by $0.056$ in the randomized-by-action data (from $0.101$ to $0.045$), and by $0.060$ in the randomized-by-policy data (from $0.107$ to $0.047$). Although this pattern is consistent across the estimation methods, the difference is again very small. Notice that this point and the previous point both rely on the same insight; in segments in which two policies recommend the same action there is less random variation in randomized-by-action data. The evidence that this results in only small differences in standard errors without segmentation (the previous point), also suggests that it will only have a small impact on the efficiency gains from segmentation.

In our discussion of the findings in Table 7.1 we highlighted that the efficiency gains from segmentation are much larger than the efficiency gains from introducing additional covariates. In our next analysis we use the randomized-by-policy data to decompose the source of the efficiency gains from segmentation.

\(^{14}\) We also note that one of the primary benefits of double machine learning is not relevant when analyzing experimental data. Double machine learning has been proposed as a method for identifying causal relationships from historical data. Establishing causality is generally not an issue with experimental data. For this reason, when introducing the method, Chernozhukov et al. (2018) treat experimental data as a special case.

\(^{15}\) In this analysis we evaluate the outcome for Policy 1 just using the customers assigned to Policy 1 (and similarly for Policy 2). An alternative approach in a randomized-by-policy setting is to pool observations in a segment that receive the recommended actions, even if the observations were (randomly) assigned to one of the other experimental conditions. For example, in the STZ study there are seven conditions associated with the seven optimized policies. We could group customers from all seven experimental conditions into segments using the recommended actions from Policies 1 and 2, and then compare the outcome for any customer that received the action recommended by Policy 1 with the outcome for any customer that received the action recommended by Policy 2. This pooling across experimental conditions would further improve the efficiency of the standard errors calculated using segmentation. This pooling approach requires segmentation. Without segmentation we cannot assign weights to segments, and so pooling would result in a distribution of observations that is not representative.
Decomposing the Efficiency Gains from Segmentation

As we discussed in Section 5, the efficiency benefits from segmentation occur for two reasons. The first reason is that comparing the difference in outcomes between policies within a segment helps to eliminate between-segment differences (Benefit 1). The second efficiency advantage is that in segments for which two policies recommend the same action, the difference in the performance of the policies can be set to zero (Benefit 2). In Table 7.2 we use the randomized-by-policy data to decompose the contribution of these efficiency gains by comparing the standard error estimates under different approaches:

<table>
<thead>
<tr>
<th>Estimation Method</th>
<th>No Segmentation</th>
<th>Partial Segmentation</th>
<th>Full Segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neither Benefit</td>
<td>$0.107</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Only Benefit 1</td>
<td>$0.105</td>
<td>$0.101</td>
<td></td>
</tr>
<tr>
<td>Benefit 1 and Benefit 2</td>
<td>$0.047</td>
<td>$0.043</td>
<td></td>
</tr>
</tbody>
</table>

Note: The table reports standard errors measuring the difference in the performance of Policy 1 and Policy 2 in the STZ study under different estimation approaches. The data is aggregated to the carrier route level (the unit of analysis is a carrier route). The standard errors are adjusted for heteroscedasticity using the Eicker-Huber-White adjustment. To preserve confidentiality, the profits are multiplied by a common random number.

We see that in the STZ data the largest source of efficiency improvement is setting the performance difference to zero in the three segments in which the policies recommend the same action (Benefit 2). For partial segmentation, the standard error estimate reduces by $0.060 from $0.107 to $0.047, of which

---

16 Notice that we cannot calculate the Only Benefit 1 approach using the randomized-by-action data. When using randomized-by-action data, segmentation always results in the performance difference being set to zero when the policies recommend the same actions (because the policies are evaluated using identical data within these segments).
$0.058 (96.7\%)$ comes from Benefit 2. Similarly, under full segmentation, the standard error estimate reduces by $0.064$ from $0.107$ to $0.043$, of which $0.058 (90.6\%)$ comes from Benefit 2.

As we might expect, full segmentation is more effective at controlling for between-segment heterogeneity (Benefit 1) than partial segmentation. However, even with full segmentation, the improvement in efficiency is relatively small.

The findings in Tables 7.1 and 7.2 also offer a comparison of the relative effectiveness of the two approaches for controlling for customer heterogeneity. One approach is to conduct the analysis separately within each segment (analyze by stratum), while the alternative approach is to include covariates. In the SZT data, including covariates is more effective than analyzing by segment. In particular, in Table 7.1 we see that adding covariates (without segmentation) reduces the standard error from $0.107$ to $0.090$. In contrast, in Table 7.2 we see that conducting the analysis separately in each of the nine segments (without adding covariates or setting the difference to zero) only reduces the standard error from $0.107$ to $0.101$.

We caution that this does not mean that including covariates is always more effective than analyzing the data by segment. In particular, the counterfactual policy segments we use merely take into account which action is recommended by each policy. They do not account for the extent to which one action is preferred over another action. Alternative segments that further control for the magnitude of the expected performance differences may result in larger efficiency improvements.

The data from the STZ study also highlights one of the limitations of the randomized-by-action experimental design. We discuss this limitation next.

**Opportunity Cost of Randomizing by Action**

Recall that randomly assigning customers to actions may introduce an opportunity cost if some actions are optimal for only a small number of customers. In the STZ study, action $b$ is more profitable than the other two actions. The seven optimized policies recognize the profitability of action $b$, and so they recommend this action for most of the households. As a result, the average profit across the approximately 2.8 million households in the seven experimental conditions pertaining to the seven policies is significantly higher than the average profit earned from the approximately 1.2 million in the conditions associated with the three actions. Randomization by action on these approximately 1.2 million customers resulted in an opportunity cost to the firm of over $100,000. This cost could be reduced by underweighting actions $a$ and $c$ when randomly allocating customers to actions.

**Summary**

In this section we used data from the STZ study to illustrate the efficiency gains from: (a) segmentation, (b) including additional covariates, and (c) using double machine learning to address overfitting and regularization biases. The findings highlight the power of segmentation using counterfactual policy logging as a simple way to improve efficiency. These efficiency gains come primarily from setting the performance difference to zero in segments where the two policies recommend the same action.

We also see efficiency gains from introducing additional covariates to control for customer heterogeneity. However, the efficiency gains from adding covariates are much smaller than the efficiency gains from segmentation. Moreover, because segmentation is itself a mechanism for controlling for customer heterogeneity, the gains from adding covariates are much smaller when also using segmentation.
We do not see large efficiency gains from double machine learning. We interpret this result as evidence that overfitting and regularization biases in OLS are not important factors in the STZ data, but caution that this does not mean they are never important when evaluating targeting policies using experimental data. Therefore, in general we recommend using double machine learning when evaluating and comparing targeting policies with data from field experiments.

8. The Scalability of the Proposed Approach

We have demonstrated our proposed program by applying it to two policies from the STZ study, which choose from three potential actions. In this section we discuss the feasibility of scaling this comparison, and the adjustments required to do so. Scaling can occur along three dimensions: the size of the action space, the number of candidate policies, and the number of additional covariates. We will discuss each of them in turn.

Increasing the Size of the Action Space

Changing the size of the action space affects all of the following: the design of the experiment in Step 1, the number of segments designed using counterfactual policy logging in Step 2, and the amount of data available to estimate the conditional treatment effects in Step 3. We begin by focusing on the design of the experiment.

Recall that a randomized-by-action experimental design assigns one experimental treatment per action. If policies can choose from a large number of potential actions, then a randomized-by-action experimental design has to divide the experimental sample into a large number of treatments. If the number of possible actions is too large then a randomized-by-action design may become impractical.

This limitation is particularly relevant for dynamic policies that involve a sequence of decisions. Each unique sequence of actions represents a single “action”. For example, Simester et al. (2006) tested their dynamic catalog targeting policy using a sequence of twelve catalog mailing opportunities. With a mail or no mail decision on each mailing opportunity, this yielded an action space with 4,096 possible actions. Similarly, Toubia et al. (2013) tested their dynamic conjoint design algorithm using a sequence of sixteen paired comparison conjoint questions. An action in this setting represents a sequence of sixteen paired comparison conjoint questions, where each question is chosen from a wide range of potential questions. As a result, the action space includes millions of potential question sequences. In this setting, it would be impractical to implement a randomized-by-action design.

Even in a static world, without dynamic policies, the number of available actions may be large. For example, recall the Dubé and Misra (2017) price targeting study that we discussed in the Introduction. After rounding targeted prices down to the nearest $9 price ending, their action space included 39 prices ranging from $119 to $499 in $10 increments. Implementing a pricing experiment with 39 treatments may not be feasible, or it could yield relatively few customers assigned to each treatment.

Data requirements depend not just upon the number of customers assigned to each treatment, they also depend upon the number of segments. In particular, the variance of the segment-level treatment estimates in Equation 5.1 is governed by the number of customers in that segment that receive a specific action. Because the segments are designed using counterfactual policy logging, increasing the size of the action space increases the number of customers assigned to each treatment, and this, in turn, increases the variance of the segment-level treatment estimates.
space unfortunately increases both the number of treatments and the number of segments. As a result, increasing the action space quickly reduces the amount of data available to estimate each segment-level treatment effect. We addressed this limitation by considering a partial segmentation approach in Section 7.

We conclude that the feasibility of the proposed program depends upon the size of the action space. If the action space is too large then it may not be feasible to implement a randomized-by-action experimental design, and/or we may have too few observations to accurately estimate the conditional treatment effects.

Potential solutions include supplementing the experimental data with historical data, particularly where the same actions were also implemented in the past. Interpolation may also allow the removal of some intermediate actions from the experimental design (Step 1). For example, if the experiment includes prices of $119 and $139, the responses might be averaged (or modeled) to estimate the response to a price of $129. Alternatively, researchers may choose to pool some segments in the estimation process (Step 3), particularly those segments that arise infrequently in the implementation data (and so receive little weight when calculating average treatment effects).

An alternative solution is to revert from a randomized-by-action to a randomized-by-policy experimental design. Steps 2-4 of our program apply equally to both randomization approaches. Increasing the number of actions does not affect the feasibility of implementing a randomized-by-policy design (although it may affect the precision of the evaluation under a randomized-by-policy design). Instead, the feasibility of this design depends upon the number of candidate policies. We consider this issue next.

**Increasing the Number of Policies**

A randomized-by-policy experimental design can quickly become impractical as the number of candidate policies increases. In contrast, a randomized-by-action design can be used to evaluate any number of policies. The ability to scale the number of candidate policies is the primary benefit of a randomized-by-action design.

Our proposed estimation approach can also scale with the number of policies. In particular, an increase in the number of policies does not affect our ability to directly estimate the pair-wise performance difference between any two policies. However, there is a subtle issue that can produce unexpected results.

Imagine we want to estimate the differences between three policies: Policy 1, Policy 2, and Policy 3. The subtle issue involves the way the model specification is modified to compare the third policy. We can use Equation 5.1 to estimate the pair-wise difference between Policy 1 and Policy 2. To do so, we estimate this equation separately for Policies 1 and 2 in each of the segments defined by their recommended actions. We can then use an analogous approach to estimate the pair-wise difference between Policy 3 and Policy 2. This allows us to directly estimate both the Policy 1 - Policy 2 performance difference and the Policy 3 - Policy 2 performance difference. Because we have these performance differences, we can also calculate an estimate of the Policy 1 - Policy 3 performance difference.

---

17 With $T$ policies, each choosing from an action space of $d$ actions, there are $d^T$ possible segments. However, some of these segments may be empty. For example, if one policy recommends mailing many more promotions than another policy, there may be no customers in the implementation data for which the first promotion recommends not mailing and the second promotion recommends mailing.

18 We would estimate Equation 5.1 for each of Policies 3 and 2 in segments defined by the recommended actions of Policies 3 and 2. Recall that we also described how to do this with a single equation in Section 5 (see the discussion of Equation 5.2).
However, these estimates are not invariant to the choice of the pair-wise setup. If we use Equation 5.1 to directly estimate the difference between Policy 1 and Policy 3, the resulting estimate of the Policy 1 - Policy 3 performance difference will almost certainly be different than if we calculate it indirectly from the Policy 1 - Policy 2 and Policy 3 - Policy 2 differences.

This discrepancy arises because in segments where two policies recommend the same action, direct estimation using Equation 5.1 (correctly) forces the estimated difference in the two policies to be zero. However, the indirect comparison of the two policies does not do this. Instead, the estimate of the differences in the performance of the policies is subject to random noise. For example, the indirect comparison of Policy 1 and Policy 3 does not force the estimated difference between Policies 1 and 3 to be zero when they recommend the same action.19 Because the pair-wise setup determines when the performance difference is set to zero (and when it is not), the estimated differences depend upon the choice of the two policies in the pair-wise comparison.

A solution is to propose a regression model that estimates differences between the three policies using the union of the three datasets: Policy 1 Dataset, Policy 2 Dataset, and Policy 3 Dataset. To do so, we construct segments using the joint recommended actions from all three policies. This ensures that the model recognizes that the true difference in performance when two policies recommend the same action is zero. A limitation of this approach is that the size of the dataset used to evaluate each model is fixed (for example, the size of the Policy 1 Dataset does not vary), and so dividing this dataset into more segments means that less data is available to estimate Equation 5.1 in each segment. For this reason, we recommend the following procedure to compare the performance of multiple policies:

1. Use the randomized-by-action data to calculate the performance of all $T$ policies. To calculate the performance of a policy, we can divide the customers into segments according to the recommended action of that policy (without using recommended actions of any other policies). To evaluate that policy within a segment where the policy recommends a specific action, we average the outcomes from the customers within the segment that were randomized to receive that specific action. Alternatively, we can use the doubly robust estimator in Equation 5.3 and double machine learning. To calculate the overall performance of the policy, we then aggregate across all segments using appropriate weights.

2. Identify the policy with the highest estimated performance and treat this as the benchmark policy.

3. Estimate $T-1$ pair-wise comparisons between the remaining policies with this benchmark policy.

This procedure both identifies the policy with the best overall performance, and reveals the extent to which this policy outperforms each of the other policies.

### Increasing the Number of Additional Covariates

Recall that our recommended approach uses “additional” covariates that describe observable customer characteristics, such as demographics, which are included to reduce the variance in the regression estimate of the treatment effect. Remember also that, in our approach, the estimation of the treatment effect is done per segment, and therefore is potentially based upon limited data. This means that a large number of covariates may introduce dimensionality problems. Furthermore, a large number of covariates can result in overfitting.

---

19 The same is true if we compare Policy 1 and Policy 2 indirectly, via their respective direct comparisons with Policy 3.
Fortunately double machine learning addresses both limitations. The double machine learning technique estimates how the response variable depends on covariates using machine learning estimators. While the use of sample-splitting eliminates the overfitting bias of these possibly complex estimators, the use of doubly robust scores serves to reduce their regularization and modeling biases. Overall, double machine learning allows the improved use of a very broad set of machine learning methods in the presence of many covariates.

Our focus in the previous sections has been on policy evaluation. The creation of a new dataset through experimentation in Step 1 suggests that our program may also provide opportunities for policy learning. We address this issue in the next section.

9. Improving the Targeting Policies (Step 4)

In this section we describe the final step in our program (Step 4), which focuses on using the experimental data from Step 1 to train a new policy. Our motivation for this proposed procedure is that firms generally have existing targeting policies. However, the information used to construct the existing policies is often informal, incompletely documented, or out-of-date. As a result, merging this old information with the new experimental data obtained in Step 1 may be impractical. This introduces a problem. Training a model only using the new data means forgoing everything that has already been learned.

For example, in the STZ field data there are seven existing policies (of which we have focused on two). These existing policies were trained using data from a previous field experiment. STZ provide clear evidence of non-stationarity in both the distribution of the targeting variables and in the underlying customer response function. This means that the original training data is now out-of-date. However, the policies created using that data contain at least some knowledge about which customers should receive each promotion. This knowledge offers the potential to improve policies designed using the new data.

Step 4 of our program provides a procedure for incorporating information in existing policies into the training of new policies without merging old and new datasets. We summarize this procedure as follows:

a. For each observation in the dataset from Step 1, determine what action each of the existing policies would recommend. Create binary indicator variables for recommended actions, and add these binary variables to the new dataset. This is an example of counterfactual policy logging.

b. Train the new model using both the original targeting variables and the binary indicators. By including the binary indicators, we include the information contained in the existing policies into the training of the new policy. The training process can decide how much weight (if any) to give to this existing knowledge by evaluating whether these indicator variables improve the goodness of fit in the training data.

We offer several comments on this procedure. The procedure can easily be extended to incorporate multiple existing policies. The information from the multiple existing policies may be incorporated independently, by creating separate indicator variables for each policy. Alternatively, the information can be incorporated jointly, by creating segments using the joint recommendations from more than one policy. Under this second approach, the segments condition the recommended actions from one policy on the recommended actions from other policies. This will result in a larger set of indicator variables, and thus require stronger regularization to accommodate more variables in developing the new targeting policy.
The procedure does not depend upon and is agnostic to the source of the existing policy. Because of this, it can incorporate the information contained in any existing policy. The procedure simply requires that the existing policy is sufficiently exhaustive that for any observation in the new dataset, the existing policy provides a recommended action (including possibly a default recommended action).

The procedure may improve the new method even if the average performance of the existing policy on the new data is relatively bad. An existing method has the potential to improve the new policy as long as it contributes independent information that is accurate. The improvement depends not just upon the accuracy of the information in the existing policy, but also upon the independence of that information. It is even possible that a relatively bad existing policy will provide a larger improvement in the new policy than a relatively good existing policy. This can occur if the new dataset contains a lot of the information in the good existing policy, but little of the information in the bad existing policy.

The procedure also does not depend upon the machine learning method used to train the new policy. Any method that can accommodate the addition of the indicator variables can be used.

To illustrate the implementation and potential benefits of this procedure, we applied it to the experimental data provided by the STZ study. For existing methods we use the same Policy 1 and Policy 2 that we used in the previous sections.

We start by randomly dividing the randomization-by-action Step 1 data into two subsets: training (70% of the data) and validation (the remaining 30% of the data). We then use the training data to train a new policy using Lasso, which we label “standard Lasso”. We then use Equation (5.1) to compare Policy 1 and Policy 2 to Standard Lasso on the validation data. We also include a naïve benchmark, the most profitable uniform policy, which sends the same promotion (action $b$) to every household. We repeat this procedure 1,000 times using different random draws for training and validation from the Step 1 data. In Table 9.1 we use Standard Lasso as a baseline, and report the average profit difference of the three policies with Standard Lasso (where, to preserve confidentiality, we multiply the profits by a common random number).

### Table 9.1 Average Profits From the Existing Policies and a Naïve Benchmark Compared to Standard Lasso

<table>
<thead>
<tr>
<th>Calculation</th>
<th>Average Profit</th>
<th>Standard Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Existing Policy 1</td>
<td>-$0.050</td>
<td>$0.006</td>
</tr>
<tr>
<td>Existing Policy 2</td>
<td>-$0.081</td>
<td>$0.007</td>
</tr>
<tr>
<td>Naïve Benchmark (Uniform Policy)</td>
<td>-$0.107</td>
<td>$0.008</td>
</tr>
</tbody>
</table>

The table compares the average profits and standard errors of three policies to Standard Lasso. Standard Lasso is the policy trained using the new STZ experimental data. Negative values indicate that Standard Lasso is more profitable than the other policies. The profit differences are averaged across 1,000 Monte-Carlo cross-validation iterations. In each iteration, we split data into training and validation sub-samples (70%:30%). We calibrate the targeting methods on the training data and evaluate performance on the validation data. To preserve confidentiality, the profits are multiplied by a common random number.
There are two findings of interest. First, standard Lasso outperforms both of the existing policies. This is not surprising. Standard Lasso is trained using the new data, which we know is equivalent to the validation data (due to the random sampling). In contrast, the existing policies were trained using old data, which, as we discussed, is not equivalent to the new data due to non-stationarity.

Second, Policy 2 is a relatively poor policy; its performance is significantly worse than performance of Policy 1. This provides an opportunity to evaluate how the quality of the existing policy affects its value for training of a new policy. To investigate this question we define six indicator variables. Policy1_a is an indicator variable which equals one if Policy 1 recommends action a (and equals zero otherwise). We similarly define Policy1_b, Policy1_c, Policy2_a, Policy2_b, and Policy2_c.

We then developed three new policies by adding these indicator variables to the new dataset. We label these new policies as: Lasso with Policy 1, Lasso with Policy 2, and Lasso with Both Policies. We used the same random draws of the data to evaluate all (old and new) policies on the validation data. The findings are reported in Table 9.2 (where we again multiply the profits by the same random number). We include the performance of standard Lasso as a benchmark (the results for this policy are also reported in Table 9.1).

The findings confirm that using counterfactual policy logging to incorporate the information from the existing policies improves the performance of the new policy. We also observe that incorporating counterfactual policies of higher quality leads to larger improvements. Including information from Policy 2 does not lead to a significant improvement of the performance in Table 9.2.

We have shown how to use counterfactual policy logging to incorporate knowledge from existing policies when training new policies. The proposed approach does not require merging old and new datasets and can leverage any existing policy, including policies developed using intuition or data that is no longer available.

### Table 9.2 Average Profits From the New Policies Compared to Standard Lasso

<table>
<thead>
<tr>
<th>Calculation</th>
<th>Average Profit</th>
<th>Standard Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lasso with Policy 1</td>
<td>$0.012</td>
<td>$0.004</td>
</tr>
<tr>
<td>Lasso with Policy 2</td>
<td>$0.003</td>
<td>$0.004</td>
</tr>
<tr>
<td>Lasso with Both Policies</td>
<td>$0.006</td>
<td>$0.004</td>
</tr>
</tbody>
</table>

The table compares the average profits and standard errors of three policies, trained using the new STZ experimental data and counterfactual policy logging, to Standard Lasso. Standard Lasso is the policy trained using the new STZ experimental data. Positive values indicate that Standard Lasso is less profitable than the new policies. The profit differences are averaged across 1,000 Monte-Carlo cross-validation iterations. In each iteration, we split data into training (70%) and validation (30%) subsamples. We calibrate the targeting methods on the training data and evaluate performance on the validation data. The profits are multiplied by a common random number.
10. Conclusions

The gold standard for evaluating targeting policies is to evaluate them using an experiment. We have presented an approach to designing and analyzing targeting experiments that offers three important advantages. First, the proposed experimental design allows evaluation (and comparison) of any policies, including policies designed after the experiment is implemented. Second, our approach yields more efficient estimates of the difference in the performance of the policies. Third, the proposed approach offers opportunities to improve targeting policies.

Segmenting customers based on the actions that different policies would have recommended (counterfactual policy logging), is the cornerstone of our proposed program. When designing the experiment, it is due to counterfactual policy logging that we can omit targeting segments of customers where the policies would assign the same action, thus cutting implementation costs. Our estimation approach also relies crucially on counterfactual policy logging to improve the precision of the estimates of the difference between policies. It does so by both recognizing that the true difference in performance is zero in segments for which the policies recommend the same action, and by reducing variation introduced by between-segment differences. Finally, counterfactual policy logging can help improve a new targeting policy, by summarizing information from existing policies when training the new policy, even if the existing policies were developed using intuition or data that is no longer available.

We illustrated the efficiency improvements of our proposed program using data from an actual field experiment. The findings confirm that the benefits of using the proposed approach can be substantial.

References


