Appendices:
A Comparison of Methods for Treatment Assignment with an Application to Playlist Generation

Carlos Fernández-Loría
Hong Kong University of Science and Technology

Foster Provost
New York University

Jesse Anderton
Spotify, New York, USA

Benjamin Carterette
Spotify, New York, USA

Praveen Chandar
Spotify, New York, USA

Appendix A Proof that Equation 6 is an unbiased estimator of Equation 5

We present here a simplified proof that Equation 6 is an unbiased estimator of Equation 5 (see Li et al. (2010) for a detailed proof):

$$\begin{aligned}\mathbb{E}_{X,Y,T}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}(\hat{a}(X)=T)\frac{Y}{\mathbb{P}(T)}\right] &= \mathbb{E}_{X,Y,T}\left[\mathbf{1}(\hat{a}(X)=T)\frac{Y}{\mathbb{P}(T)}\right] \\ &= \sum_{\forall j\in T}\mathbb{E}_{X,Y}\left[\mathbf{1}(\hat{a}(X)=j)\frac{Y}{\mathbb{P}(T=j)}\Big|T=j\right]\mathbb{P}(T=j) \\ &= \sum_{\forall j\in T}\mathbb{E}_{X,Y}[\mathbf{1}(\hat{a}(X)=j)Y|T=j],\end{aligned}$$

and given the unconfoundedness assumption:

$$\begin{aligned}&= \sum_{\forall j\in T}\mathbb{E}_{X,Y(j)}[\mathbf{1}(\hat{a}(X)=j)Y(j)] \\ &= \mathbb{E}_{X,\{Y(j):j\in T\}}\left[\sum_{\forall j\in T}\mathbf{1}(\hat{a}(X)=j)Y(j)\right] \\ &= \mathbb{E}_{Y(\hat{a}(X))}[Y(\hat{a}(X))]\end{aligned}$$

Q.E.D.

Appendix B Proof that minimizing weighted misclassification rate (Equation 13) is equivalent to minimizing expected regret (Equation 4)

We present here a simplified proof that minimizing Equation 13 is equivalent to minimizing Equation 4 (see Beygelzimer and Langford (2009) for more details):

$$\begin{aligned}\arg\min_{\hat{a}}\ WMR(\hat{a}) &= \arg\min_{\hat{a}}\ \mathbb{E}_{X,Y,T}\left[\mathbf{1}(\hat{a}(X)\neq T)\frac{Y}{\mathbb{P}(T)}\right] \\ &= \arg\min_{\hat{a}}\ \mathbb{E}_{Y,T}\left[\frac{Y}{\mathbb{P}(T)}\right] - \mathbb{E}_{X,Y,T}\left[\mathbf{1}(\hat{a}(X)=T)\frac{Y}{\mathbb{P}(T)}\right] \\ &= \arg\min_{\hat{a}}\ -\mathbb{E}_{X,Y,T}\left[\mathbf{1}(\hat{a}(X)=T)\frac{Y}{\mathbb{P}(T)}\right],\end{aligned}$$

and from the proof in Appendix A, it follows that:

$$\begin{aligned}&= \arg\min_{\hat{a}} -\mathbb{E}_{Y(\hat{a}(X))}[Y(\hat{a}(X))] \\ &= \arg\min_{\hat{a}} \mathbb{E}_{Y(a^*(X)),Y(\hat{a}(X))}[Y(a^*(X)) - Y(\hat{a}(X))] \\ &= \arg\min_{\hat{a}}\ \text{Regret}(\hat{a})\end{aligned}$$

Q.E.D.

Appendix C Data generating process of the simulated example in Section 4.3

Below are the formulations for all variables in the data generation process, including the potential outcome when treated, $Y(1)$; the potential outcome when untreated, $Y(0)$; and the feature used to make intervention decisions, X. The potential outcomes are normally distributed and the feature is uniformly distributed:

$$X \sim \mathcal{U}_{[0,1]} \tag{1}$$

$$Y(1) \sim \mathcal{N}(\mu_1(X), 0.2) \tag{2}$$

$$Y(0) \sim \mathcal{N}(\mu_0(X), 0.2) \tag{3}$$

$$\mu_1(x) = \frac{1}{1+e^{2-5x}} + 0.5 \tag{4}$$

$$\mu_0(x) = \frac{0.1}{x+0.4} + 0.5x^2 + 0.5 \tag{5}$$

The data we generated consists of 50 treated observations and 50 untreated observations.

Appendix D Comparison to conventional treatment assignment problems

There are a few ways in which the deployment of content selection systems as treatments is different from conventional treatment assignment problems. First, one system can be a better (or worse) treatment than another system only to the extent that it generates different content (i.e., playlists that have different songs

or rank songs differently). If the content produced for a given user does not vary across systems, then all systems are essentially the same treatment for that user, and the user's engagement cannot be improved by deploying a different variant than the system currently in production.

Therefore, helpful features for treatment assignment are not only the ones that describe user behavior—as in conventional treatment assignment problems—but also the ones that describe how the system's content could be affected (relative to other systems). Going back to our example with new and experienced users, if System B relies on a more complex machine learning system than System A, then we may expect System B to be more susceptible to cold start problems, and as a result user tenure could be a good candidate feature to decide whether to assign one system or the other. Additionally, user behavior could also play a role when deciding what system to assign. For example, if System B generally produces more niche playlists than System A (e.g., the former considers songs in languages other than English whereas the latter does not), we may also want to consider features that describe users that may appreciate such playlists (e.g., the country where users live).

Second, when deploying systems as treatments, features may need to be selected based on the feasibility of actually using them to make system assignments. More specifically, implementation challenges may prevent us from using features that are helpful to determine whether one system is better than another in a particular context. For instance, even though "time active in session" may be a useful feature to decide whether to assign a system that produces playlists based on the last songs the user has played, using this feature effectively would require the ability to make system assignments throughout the user session, which may be a non-trivial technical capability. On the other hand, features that change less frequently (e.g., type of subscription, country of residence) would allow making system assignments even before users log in.

Finally, treatment assignment policies in our setting are meant to be used to make large-scale decisions automatically; millions of playlists are generated by Spotify every day, each in a fraction of a second. This presents a salient contrast to other settings where treatment assignment policies are intended to support and improve human decision-making, such as in medical prescriptions and public policy. In such settings, transparency is critical and the rationale behind each decision must be clearly understood.

Understanding the rationale behind treatment assignments can also be important in our context. For example, deeper understanding of the most important factors for system assignment may help to inform the development of new playlist generation systems (i.e., new 'treatments') that may further improve engagement.

However, our study focuses on how to learn a policy to identify the system that leads to the highest expected number of song streams for each user, not on understanding what are the factors that result in one system being better than another.

Appendix E Extended analysis

In this extended empirical analysis, we consider how (1) the available features and (2) the machine learning procedure can influence the preference of one metalearner over the others.

E.1. The importance of the available features

The usefulness of estimating treatment assignment policies with machine learning depends to a large extent on the features that are available to classify individuals into different treatments. If the features are not informative of preferred treatments, then there is no advantage over using a standard A/B test to determine (and assign) the alternative that works best on average. However, with more features, it also becomes harder to estimate *optimal* treatment assignments (from a regret minimization perspective) because the optimal policy also becomes more complex. Thus, typically, the more features there are, the more training data it takes for the metalearners to converge to the optimal treatment assignment policy.

Figure 1 shows the results of deploying treatment assignment policies that were estimated with an increasing number of features. Features are ordered from the one with the most categorical values to the one with the least categorical values. As before, the lines correspond to each of the three metalearners—O-learner (blue), E-learner (orange) and A-learner (green)—and the areas around the lines represent 95% confidence intervals. As expected, all metalearners perform better with more features because they have more information about the individual to make assignments. Furthermore, the metalearners have a similar performance when there are few features because there is enough training data for them to converge to the same policy. For example, the A-learner performs the same as the O-learner when there are 3 features and the same as the E-learner when there are one or two features. As mentioned, this is expected because machine learning algorithms converge faster to best-in-class models when there are fewer features. Trees are universal approximators, so they lead to the same (optimal) treatment assignment policy given enough training data.

On the other hand, as the number of features increases, the A-learner becomes better than the other metalearners. As discussed in Section 4.4, the A-learner focuses on minimizing prediction errors that negatively affect decision making and ignores all other types of errors. As a result, it does a better job at exploring

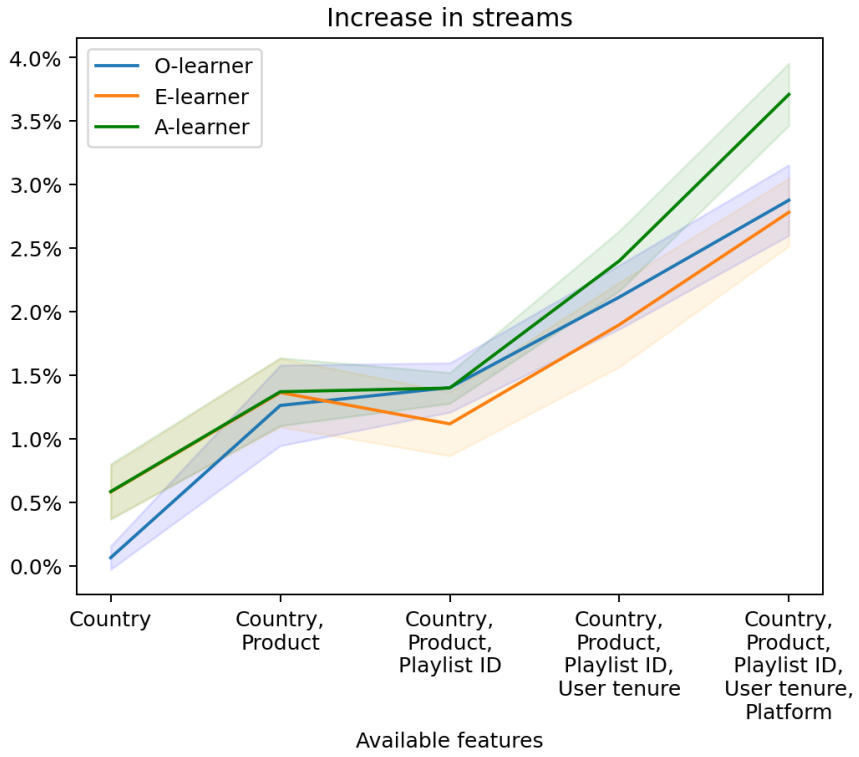

Figure 1 **Treatment assignment performance by number of features. All treatment assignment metalearners improve substantially with more features. The A-learner is more likely to outperform the other metalearners when the number of features is larger.**

the feature space and discriminating individuals based on their preferred treatment assignment (rather than based on causal effects or outcomes).

E.2. The importance of the machine learning algorithm.

In this extended analysis, we show how the choice of the base learner may also affect the comparison of the metalearners. We consider multiple base learners here: the tree-induction algorithms discussed in Section 5.3, random forest, and generalized linear regression. Random forest was implemented using ensembles of the tree-based methods described in Section 5.3. In the case of the generalized linear regression models, for the O-learner, we used a linear regression for each treatment condition. For the E-learner, we used a linear regression on the transformed variable proposed by Athey and Imbens (2016) for each of the 3 non-control systems. Finally, for the A-learner, we used a weighted logistic regression.

Figure 2 shows the performance of the methods we considered. The black lines represent 95% confidence intervals. These confidence intervals describe deviations in the *individual* performance of each method, but

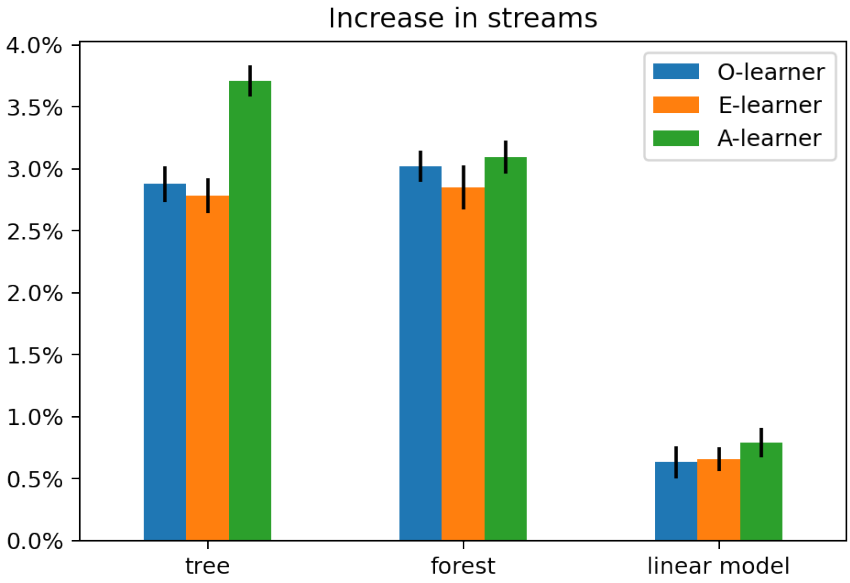

Figure 2 **Treatment assignment performance by type of metalearner and base learner. The A-learner with the tree base learner works best.**

they are not appropriate to compare the metalearners because they do not account for the correlation in the deviations (e.g., when one method works particularly well in a fold, the other methods will also tend to work particularly well). Importantly, this figure shows that the best performing method corresponds to the A-learner in our main analysis (i.e., the one that uses tree induction as a base learner).

Figure 3 appropriately shows the difference in performance between metalearners. For example, "from O-learner to A-learner" corresponds to the the difference in performance between the A-learner and the O-learner (i.e., the difference between green bars and blue bars in Figure 2). The black lines represent 95% confidence intervals. The A-learner is the best performing metalearner in all cases, but the improvement is not as large when random forests and linear models are used. Thus, as discussed in Section 6.3, results can be affected by the base learner.

There are multiple points worth discussing here. First, how the base learner affects the comparison of the metalearners can vary from one data set to another. For example, Olaya et al. (2020) compare various instances of the E-learner and the O-learner across multiple data sets using random forest and logistic regression as base learners. They find that none of the evaluated techniques consistently outperforms the other techniques, which implies that choosing among metalearners (and base learners) should be an empirical undertaking. The results we present here just show that the choice of the base learner can affect how the

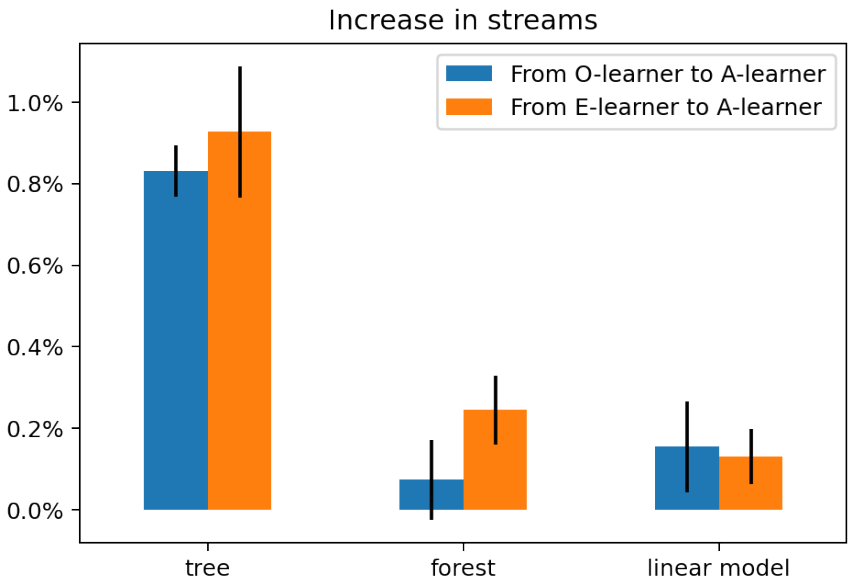

Figure 3 **Increase in streams when the A-learner is used instead of the other meta-learners. The A-learner generally works better, but the improvement is moderated by the base learner.**

metalearners compare to each other; they do not imply that, in general, the difference between metalearners should be smaller or larger for any of the base learners under consideration.

Second, recall that, when the base learner is a consistent estimator, all metalearners estimate the same (optimal) treatment assignment policy with large enough data. This implies that, in general, differences in the estimated policies will be larger with smaller data. Therefore, given that most firms cannot conduct A/B tests as large as the one in this study (770 million observations), we should expect to observe larger differences in performance between metalearners in other practical settings.

Third, one possible explanation for the results in Figure 3 is that tree models suffer from higher variance than random forests and linear models, and as a result, differences between the metalearners will tend to be larger when trees are used as base learners. Nevertheless, the distinction between metalearners can also be important when low-variance models are used (e.g., linear models) because bias also plays an important role in the learning procedure. As discussed in Section 4.4, the A-learner focuses on minimizing bias that negatively affects decision making, whereas the O-learner and the E-learner focus on minimizing bias that negatively affects outcome and causal effect predictions. Therefore, in general, the A-learner will converge to better policies than the other metalearners when biased base learners are used. Elmachtoub and Grigas (2021) provide evidence of this for non-causal decision making.

References

Susan Athey and Guido Imbens. 2016. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences* 113, 27 (2016), 7353–7360.

Alina Beygelzimer and John Langford. 2009. The offset tree for learning with partial labels. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 129–138.

Adam N Elmachtoub and Paul Grigas. 2021. Smart "predict, then optimize". *Management Science* (2021).

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 661–670.

Diego Olaya, Kristof Coussement, and Wouter Verbeke. 2020. A survey and benchmarking study of multi-treatment uplift modeling. *Data Mining and Knowledge Discovery* 34, 2 (2020), 273–308.