Variable Importance Matching for Causal Inference (Supplementary material)

Quinn Lanners[1] **Harsh Parikh**[2] **Alexander Volfovsky**[3] **Cynthia Rudin**[2] **David Page**[1]

[1]Dept. of Biostatistics, Duke University, Durham, NC, USA.
[2]Dept. of Computer Science, Duke University, Durham, NC, USA.
[3]Dept. of Statistical Science, Duke University, Durham, NC, USA.

A PROOFS FOR THEOREMS IN SECTION 5

Theorem 5.1 (Closeness in $\mathbf{X}$ implies closeness in Y). Consider a p-dimensional covariate space where for $t' \in \{0,1\}$, $f^{(t')}(\mathbf{X}_i) = \mathbb{E}[Y_i|\mathbf{X} = \mathbf{X}_i, T = t'] = \mathbf{X}_i\boldsymbol{\beta}^{(t')}$. Construct $\mathcal{M} \in \mathbb{R}^{p\times p}$ where for all $l, r \in \{1, ..., p\}$ $\mathcal{M}_{l,l} = |\beta_l^{(t')}|$ and for $l \neq r$ $\mathcal{M}_{l,r} = 0$. Then, $\forall i, j$, we have that $d_{\mathcal{M}}(\mathbf{X}_i, \mathbf{X}_j) \geq \left|f^{(t')}(\mathbf{X}_i) - f^{(t')}(\mathbf{X}_j)\right|$.

Proof for Theorem 5.1.

$$d_{\mathcal{M}}(\mathbf{X}_i, \mathbf{X}_j) = \sum_{l=1}^{p} \mathcal{M}_{l,l}|X_{i,l} - X_{j,l}| = \sum_{l=1}^{p} |\beta_l^{(t')}||X_{i,l} - X_{j,l}| \geq \left|\sum_{l=1}^{p} \beta_l^{(t')}(X_{i,l} - X_{j,l})\right|$$
$$= \left|f^{(t')}(\mathbf{X}_i) - f^{(t')}(\mathbf{X}_j)\right|.$$

QED

Theorem 5.2 (Optimality of $\mathcal{M}$). Using the setup of Theorem 5.1, let $\text{supp}(\mathbf{X}) = \mathbb{R}^p$. Consider an arbitrary diagonal Mahalanobis distance matrix $\widetilde{\mathcal{M}} \in \mathbb{R}^{p\times p}$ where $\|\widetilde{\mathcal{M}}\|_1 = \|\boldsymbol{\beta}^{(t')}\|_1$ and $\widetilde{\mathcal{M}}_{l,l} > 0$ when $|\beta_l^{(t')}| > 0$. For some $\epsilon \geq 0$ and $\mathbf{X}_1 \in \mathbb{R}^p$, define $S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1) := \{\mathbf{X}_2 : \mathbf{X}_2 \in \mathbb{R}^p, d_{\widetilde{\mathcal{M}}}(\mathbf{X}_1, \mathbf{X}_2) = \epsilon\}$. Then,

$$\sup_{\mathbf{X}_2 \in S_{\mathcal{M},\epsilon}(\mathbf{X}_1)} |f^{(t')}(\mathbf{X}_1) - f^{(t')}(\mathbf{X}_2)| \leq \sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} |f^{(t')}(\mathbf{X}_1) - f^{(t')}(\mathbf{X}_3)|.$$

In what follows, we recall that a diagonal Mahalanobis distance matrix, $\widetilde{\mathcal{M}}$, is:

- diagonal: for all $l, r \in \{1, ..., p\}$, $l \neq r$, $\widetilde{\mathcal{M}}_{l,r} = 0$.
- non-negative entries: for all $l \in \{1, ..., p\}$, $\widetilde{\mathcal{M}}_{l,l} \geq 0$.

To prove this result, we first prove the following two lemmas.

Lemma 1 (Maximum Absolute Difference in Expected Outcomes under $\mathcal{M}$). Consider a p-dimensional covariate space where $\text{supp}(\mathbf{X}) = \mathbb{R}^p$ and for $t' \in \{0,1\}$, $f^{(t')}(\mathbf{X}_i) = \mathbb{E}[Y_i|\mathbf{X} = \mathbf{X}_i, T = t'] = \mathbf{X}_i\boldsymbol{\beta}^{(t')}$. Define $\mathcal{L} := \{l : \left|\beta_l^{(t')}\right| > 0\}$. Construct any diagonal Mahalanobis distance matrix, $\widetilde{\mathcal{M}}$, where $\|\widetilde{\mathcal{M}}\|_1 = \|\boldsymbol{\beta}^{(t')}\|_1$ and $\widetilde{\mathcal{M}}_{l,l} > 0$ when $|\beta_l^{(t')}| > 0$. Then, for some $\epsilon \geq 0$ and $\mathbf{X}_1 \in \mathbb{R}^p$, let $S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)$ be as defined in Theorem 5.2. We can conclude that

$$\sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} |f^{(t')}(\mathbf{X}_1) - f^{(t')}(\mathbf{X}_3)| = \epsilon \max_{l\in\mathcal{L}} \left\{\frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}}\right\}.$$

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

Proof of Lemma 1.

$$\sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} |f^{(t')}(\mathbf{X}_1) - f^{(t')}(\mathbf{X}_3)| = \sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} \left| \sum_{l \in \mathcal{L}} \beta_l^{(t')} (X_{1,l} - X_{3,l}) \right|.$$

Note that since $\text{supp}(\mathbf{X}) = \mathbb{R}^p$, with probability strictly greater than zero there exists an $\mathbf{X}_1$ and $\mathbf{X}_3$ such that $d_{\widetilde{\mathcal{M}}}(\mathbf{X}_1, \mathbf{X}_3) = \epsilon$ and for all $l \in \mathcal{L}$, $X_{1,l} > X_{3,l}$ when $\beta_l^{(t')} > 0$ and $X_{1,l} < X_{3,l}$ when $\beta_l^{(t')} < 0$. Then,

$$\begin{aligned}
\sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} \left| \sum_{l \in \mathcal{L}} \beta_l^{(t')} (X_{1,l} - X_{3,l}) \right| &= \sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} \left\{ \sum_{l \in \mathcal{L}} \left| \beta_l^{(t')} (X_{1,l} - X_{3,l}) \right| \right\} \\
&= \sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} \left\{ \sum_{l \in \mathcal{L}} \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} \widetilde{\mathcal{M}}_{l,l} |X_{1,l} - X_{3,l}| \right\}.
\end{aligned}$$

Note that $\left\{ \sum_{l \in \mathcal{L}} \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} \widetilde{\mathcal{M}}_{l,l} |X_{1,l} - X_{3,l}| : \mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1) \right\}$ is maximized at $\epsilon \max_{l \in \mathcal{L}} \left\{ \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} \right\}$. It is known that if the maximum value of a set is in the set, the supremum of that set equals the maximum value of that set. Therefore, we conclude that,

$$\sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} \left\{ \sum_{l \in \mathcal{L}} \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} \widetilde{\mathcal{M}}_{l,l} |X_{1,l} - X_{3,l}| \right\} = \epsilon \max_{l \in \mathcal{L}} \left\{ \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} \right\}.$$

QED

Lemma 2 Under the same setup as Lemma 1, $\max_{l \in \mathcal{L}} \left\{ \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} \right\} \geq 1$.

Proof of Lemma 2. First note that $\sum_{l \in \mathcal{L}} \widetilde{\mathcal{M}}_{l,l} \leq \sum_{l=1}^{p} \widetilde{\mathcal{M}}_{l,l} = \sum_{l=1}^{p} |\beta_l^{(t')}| = \sum_{l \in \mathcal{L}} |\beta_l^{(t')}|$. There are two possible cases. In case one, $\forall l \in \mathcal{L}$, $\widetilde{\mathcal{M}}_{l,l} = \mathcal{M}_{l,l} = |\beta_l^{(t')}|$. Then $\max_{l \in \mathcal{L}} \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} = 1$. In case two, there exists $l \in \mathcal{L}$ for which $\widetilde{\mathcal{M}}_{l,l} \neq |\beta_l^{(t')}|$. But then there must exist an $l' \in \mathcal{L}$ for which $\widetilde{\mathcal{M}}_{l',l'} < |\beta_{l'}^{(t')}| \implies \max_{l \in \mathcal{L}} \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} > 1$. QED

Proof of Theorem 5.2. First note that $\mathcal{M}$ is a diagonal Mahalanobis distance matrix, $\|\mathcal{M}\|_1 = \|\boldsymbol{\beta}^{(t')}\|_1$, and $\mathcal{M}_{l,l} > 0$ when $|\beta_l^{(t')}| > 0$. The proof of the theorem then follows directly from Lemma 1 and Lemma 2.

$$\begin{aligned}
\sup_{\mathbf{X}_2 \in S_{\mathcal{M},\epsilon}(\mathbf{X}_1)} |f^{(t')}(\mathbf{X}_1) - f^{(t')}(\mathbf{X}_2)| &= \epsilon \max_{l \in \mathcal{L}} \left\{ \frac{|\beta_l^{(t')}|}{\mathcal{M}_{l,l}} \right\} \\
&= \epsilon \max_{l \in \mathcal{L}} \left\{ \frac{|\beta_l^{(t')}|}{|\beta_l^{(t')}|} \right\} \\
&= \epsilon \\
&\leq \epsilon \max_{l \in \mathcal{L}} \left\{ \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} \right\} \\
&= \sup_{\mathbf{X}_3 \in S_{\widetilde{\mathcal{M}},\epsilon}(\mathbf{X}_1)} |f^{(t')}(\mathbf{X}_1) - f^{(t')}(\mathbf{X}_3)|.
\end{aligned}$$

Where $\epsilon \leq \epsilon \max_{l \in \mathcal{L}} \left\{ \frac{|\beta_l^{(t')}|}{\widetilde{\mathcal{M}}_{l,l}} \right\}$ because of Lemma 2. QED

Theorem 5.3 (Consistency of LCM). For $t' \in \{0, 1\}$, let $f^{(t')}(\mathbf{X}_i) = \mathbb{E}[Y_i | \mathbf{X} = \mathbf{X}_i, T = t']$. Let $f^{(t')}$ be Lipschitz continuous and,

$$\text{supp}\left(f^{(t')}\right) := \left\{ j : \text{importance of } \mathbf{X}_{\cdot,j} \text{ in } f^{(t')} \text{ is } > 0 \right\}.$$

Denote $d_{\mathcal{M}^*}$ as the distance metric learned by LCM in Section 4 and let $\Gamma(\mathcal{M}^*) = \{j : \mathcal{M}^*_{j,j} > 0\}$. LCM is consistent for CATE estimation if $\text{supp}\left(f^{(0)}\right) \bigcup \text{supp}\left(f^{(1)}\right) \subseteq \Gamma(\mathcal{M}^*)$.

Proof of Theorem 5.3. First, let us introduce the concept of a smooth distance metric (defined in Parikh et al. [2022]).

Definition A.1 (Smooth Distance Metric)**.** $d : \mathbf{X} \times \mathbf{X} \to \mathbb{R}^+$ is a smooth distance metric if there exists a monotonically increasing bounded function $\delta_d(\cdot)$ with zero intercepts, such that $\forall i, j \in \mathcal{S}$ if $T_i = T_j = t'$ and $d(\mathbf{X}_i, \mathbf{X}_j) \leq a$ then $|\mathbb{E}[Y_i(t')|\mathbf{X}_i] - \mathbb{E}[Y_j(t')|\mathbf{X}_j]| \leq \delta_d(a)$.

Theorem 1 in [Parikh et al., 2022] shows that matching with a smooth distance metric guarantees consistency of CATE estimates.

Recovering the correct support for the potential outcome functions implies that restricting to only variables in the recovered support, the potential outcomes are independent of the covariates: $(Y(1), Y(0)) \perp \mathbf{X} \mid \{\mathbf{X}_{\cdot,j}\}_{j \in \text{supp}(f^{(0)}) \cup \text{supp}(f^{(1)})}$. Also, note that if $\{\mathbf{X}_{i,j}\}_{j \in \text{supp}(f^{(0)}) \cup \text{supp}(f^{(1)})}$ is close to $\{\mathbf{X}_{k,j}\}_{j \in \text{supp}(f^{(0)}) \cup \text{supp}(f^{(1)})}$ then $f^{(0)}(X_i)$ is close to $f^{(0)}(X_k)$ and $f^{(1)}(X_i)$ is close to $f^{(1)}(X_k)$ by the definition of support and the Lipschitz continuity assumption. Thus, if $\text{supp}(f^{(0)}) \cup \text{supp}(f^{(1)}) \subseteq \Gamma(\mathcal{M}^*)$ then $d^*_{\mathcal{M}}$ is a smooth distance metric. This guarantees the consistency of our estimates. QED

Consistency of LASSO. Much work has been done on the consistency of LASSO for feature selection [Zhang et al., 2016]. The ability for LASSO to recover the correct support even in the case of non-linear targets makes it more robust to model misspecification. LASSO is consistent for support recovery if $f(\mathbf{X}_i, t) = \mathbb{E}[Y_i|\mathbf{X} = \mathbf{X}_i, T = t']$ satisfies one of the following conditions:

(i) $f(\mathbf{X}_i, t') = \mathbf{X}_i \boldsymbol{\beta}^{(\boldsymbol{t'})}$

(ii) $f(\mathbf{X}_i, t') = g\left(\mathbf{X}_i \boldsymbol{\beta}^{(\boldsymbol{t'})}\right)$ where $\beta_k^{(t')} \neq 0$ for $k \in \{1, .., r\}$, for some $r \leq p$, and, if $r < p$, $\beta_k^{(t)} = 0$ for $k \in \{r, ..., p\}$, and the following conditions are met:

   (a) $\mathbf{Cov}(\mathbf{X}, \mathbf{X})$ is invertible.

   (b) The eigenvalues of $\Sigma_{r,r} = \mathbf{Cov}(\mathbf{X}_{1:r}, \mathbf{X}_{1:r})$ are such that $0 < c_1 \leq \Lambda(\Sigma_{r,r}) \leq c_2 < \infty$. Where $\Lambda(\Sigma_{r,r})$ are the eigenvalues of $\Sigma_{r,r}$.

   (c) $E[Y(t')]^4 < \infty$

   (d) g is differentiable almost everywhere and for $t \sim \mathcal{N}(0, 1)$, $E(|g(t)|) < \infty$ and $E(|g'(t)|) < \infty$.

   (e) For all i, $E\left[X_i^T X_i \left|g\left(\mathbf{X}_i \boldsymbol{\beta}^{(\boldsymbol{t'})}\right)\right|^2\right] < \infty$.

B METHOD IMPLEMENTATION FOR EXPERIMENTS

In this section we outline how we implemented each method used in our experiments. To calculate CATE estimates for all samples, we employed the same η-fold cross-fitting strategy for each method. In particular, we train models to estimate the $\widehat{Y}_i(t') = f^{(t')}(\mathbf{X}_i)$ for $t' \in \{0, 1\}$ using $S_{n,tr}$ and perform estimation on $S_{n,est}$. The only method that we did not use cross-fitting for was GenMatch, which does not use the outcome to learn it's distance metric and thus does not require a training set. All references to scikit-learn refer the Python machine learning package from Pedregosa et al. [2011].

- **LASSO Coefficient Matching**: We implemented the method described in this paper in Python. We use scikit-learn's `LassoCV` to learn $d_{\mathcal{M}^*}$ and `NearestNeighbors` with `metric='manhattan'` to perform nearest neighbor matching.
- **Linear and Nonparametric Prognostic Score Matching**: We follow the notion of a prognostic score outlined in Hansen [2008]. In particular, we employ a *double* prognostic score matching method were we model both the control and treatment space separately as $\widehat{Y}_i(t') = f^{(t')}(\mathbf{X}_i)$ for $t' \in \{0, 1\}$. For linear PGM we use scikit-learn's `LassoCV` as our prognostic score models and for nonparametric PGM we use `GradientBoostingRegressor` for our prognostic score models. We then match with replacement on $[f^{(0)}(\mathbf{X}_i), f^{(1)}(\mathbf{X}_i]$ using scikit-learn's `NearestNeighbors` with `metric='euclidean'` to perform nearest neighbor matching. We estimated CATEs with the same mean estimator as LCM.
- **MALTS Matching**: We use the method developed in Parikh et al. [2022] that was implemented in Python [Parikh, 2020]. We use the package's `mean` CATE estimator with `smooth_cate=False`.
- **MatchIt**: We use MatchIt's implementation of GenMatch [Ho et al., 2007]. We kept the default setting of `ratio=1`, which set $K = 1$ for matching. But we matched with replacement to be in line with LCM and the other matching methods we compared with.

- **Linear and Nonparametric TLearner**: We use the EconML TLearner implementation from Battocchi et al. [2019]. For Linear TLearner we use scikit-learn's `LassoCV` for our models and for Nonparametric TLearner we use scikit-learn's `GradientBoostingRegressor` for our models.
- **AHB**: We use the method developed in Morucci et al. [2020] that was implemented in R [Lab, 2022]. We use the package's `AHB_fast_match` implementation with the default settings.
- **Bart T-Learner**: We use the dbarts R package from Dorie et al. [2019]. We train a BART model on $S_{n,tr}$ to model $\widehat{Y}_i(t') = f^{(t')}(\mathbf{X}_i)$ for $t' \in \{0, 1\}$. We then estimate CATEs for each $j \in S_{n,est}$ as $f^{(1)}(\mathbf{X}_j) - f^{(0)}(\mathbf{X}_j)$.
- **Linear DoubleML**: We use the `econml.dml.DML` class in the econml Python package from Battocchi et al. [2019]. We fit a model on $S_{n,tr}$ setting `model_y=WeightedLassoCV`, `model_t=LogisticRegressionCV`, and `model_final=LassoCV`. We then estimate CATEs for each $j \in S_{n,est}$ using the `.effect()` method.
- **Causal Forest DoubleML**: We use the `econml.dml.CausalForestDML` class in the econml Python package from Battocchi et al. [2019]. We fit a model on $S_{n,tr}$ setting `model_y=WeightedLassoCV` and `model_t=LogisticRegressionCV`. We then estimate CATEs for each $j \in S_{n,est}$ using the `.effect()` method.
- **Causal Forest**: We use the implementation of causal forest from the grf R package from Battocchi et al. [2019]. We fit a model on $S_{n,tr}$ with the default package settings. We then used the fit model to estimate CATEs for each $j \in S_{n,est}$.

C EXPERIMENTAL DETAILS FOR SECTION 6 AND SECTION 7

In this section, we describe the data generating processes used and provide further details regarding the setup of each experiment conducted in this paper. The source code necessary to reproduce all of the experiments in this paper is located in the GitHub repository: `https://github.com/almost-matching-exactly/variable_imp_matching`.

C.1 DATA GENERATION PROCESSES

Here we outline the data generation processes (DGPs) not fully outlined in the main text.

Sine and Exponential DGPs. *Used in Sections 6.2 and 7.1*. We generate the covariates and treatment assignments for the Sine and Exponential DGPs in a similar manner. For both, we generate data as follows:

$$\begin{aligned} X_{i,1}, \ldots, X_{i,p} &\overset{iid}{\sim} \text{Uniform}(-\alpha, \beta) \\ \epsilon_{i,y} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2), \epsilon_{i,t} &\overset{iid}{\sim} \mathcal{N}(0, 1) \\ T_i = \mathbb{1}\left[\text{expit}\Big(X_{i,1} + X_{i,2} + \epsilon_{i,t}\Big) > 0.5\right] \\ Y_i = T_i Y_i(1) + (1 - T_i) Y_i(0) + \epsilon_{i,y}, \end{aligned}$$

where expit is the logistic sigmoid: $\text{expit}(x) = \frac{1}{1+e^{-x}}$.

For **Sine** we set $\alpha = \beta = \pi$, $\sigma^2 = 0.1$ and calculate the potential outcomes as

$$Y_i(0) = \sin(X_{i,1}),\ Y_i(1) = \sin(X_{i,1}) - \sin(X_{i,2}).$$

For **Exponential** we set $\alpha = \beta = 3$, $\sigma^2 = 1$ and calculate the potential outcomes as

$$Y_i(0) = 2e^{X_{i,1}} - \sum_{j=2}^{3} e^{X_{i,j}},\ Y_i(1) = 2e^{X_{i,1}} - \sum_{j=2}^{3} e^{X_{i,j}} + e^{X_{i,4}}.$$

Quadratic DGP. *Used in Sections 6.3 and 7.3*. This quadratic data generation process is also described in Parikh et al. [2022]. This DGP includes both linear and quadratic terms. For each sample, let $\mathbf{X}_i$ be a p-dimensional vector where the first $k \leq p$ covariates are relevant and $\kappa \leq k$ is the number of covariates relevant to determining the treatment choice. The DGP is outlined below.

$$X_{i,p} \overset{iid}{\sim} \mathcal{N}(1, 1.5),\ \epsilon_{i,y}\epsilon_{i,t} \overset{iid}{\sim} \mathcal{N}(0,1),\ \ s_1, \ldots, s_{|k|} \overset{iid}{\sim} \text{Uniform}\{-1, 1\}$$
$$\alpha_j | s_j \overset{iid}{\sim} \mathcal{N}(10 s_j, 9),\ \beta_1, \ldots, \beta_{|k|} \overset{iid}{\sim} \mathcal{N}(1, 0.25)$$

$$Y_i(0) = \sum_{j \le k} \alpha_j X_{i,j}$$

$$Y_i(1) = \sum_{j \le k} \alpha_j X_{i,j} + \sum_{j \le k} \beta_j X_{i,j} + \sum_{j \le k} \sum_{j' \le k} X_{i,j} X_{i,j'}$$

$$T_i = \mathbb{1}\left[\text{expit}\Big(\sum_{j \le \kappa} X_{i,j} - \kappa + \epsilon_{i,t} \Big) > 0.5 \right]$$

$$Y_i = T_i Y_i(1) + (1 - T_i) Y_i(0) + \epsilon_{i,y}$$

Where $\text{expit}(x) = \frac{1}{1+e^{-x}}$.

Basic Quadratic DGP. *Used in Section 7.2*. This DGP is a quadratic DGP centered at zero. We generate each sample as shown.

$$X_{i,1}, \ldots, X_{i,10} \overset{iid}{\sim} \mathcal{N}(0, 2.5),\ \epsilon_{i,y} \overset{iid}{\sim} \mathcal{N}(0,1),\ T_i \sim \text{Bernoulli}(0.5)$$
$$Y_i(0) = X_{i,1}^2,\ Y_i(1) = X_{i,1}^2 + 10$$
$$Y_i = T_i Y_i(1) + (1 - T_i) Y_i(0) + \epsilon_{i,y}$$

C.2 EXPERIMENTAL DETAILS

In Table 1 we provide details on the experiments shown in this paper. We include additional notes for selected experiments below:

- Section 6.1: Accuracy and Auditability: We included the school id as a categorical covariate in our dataset. After preprocessing the categorical covariates, we had 6 continuous covariates and 98 binary covariates that we used as input to each model. We used only two splits due to the small occurrence rate of many of the categorical values. We repeated the cross-fitting process 50 times to smooth out treatment effect estimates for each method. All of the results in this section are for the combined 50 iterations.
- Section 6.3: Scalability: The matchit package only performs k:1 matching, so we kept K=1 for GenMatch (which is the default value). Reported runtimes were measured on a Slurm cluster with VMware, where each VM was an Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz. For measuring runtime, we ran each method 20 times on each dataset size. We report the average runtime for each method on each dataset. The variability across the 20 runs was negligible so we ommitted bars showing the standard deviation from the final plot. Each individual runtime measurement was ran on a separate Slurm job that was allocated a single core with 16GB RAM.
- Section 7.3: LCM-Augmented-PGM: For ease of implementation, we did not perform cross-fitting for this experiment. Rather, we just used half of the samples (2500) for training and the other half of the samples (2500) for estimation.

D ADDITIONAL EXPERIMENTAL RESULTS

In this section, we include additional experimental results using LCM. We first discuss further findings from experiments in Section 6 and Section 7. We then show results of additional experiments comparing LCM to non-matching methods and matching methods with equal weights after feature selection.

Section 6.1: Accuracy and Auditability. Figure 1 in this document is an expanded plot of Figure 1(a) in the main text. The supplementary material's Figure 1 includes S3, X1, and all other effect modifiers X2, C1=1, C1=13, and C1=14. As mentioned in the caption of Figure 1(a) in the main text, S3 indicates the self-reported prior achievements of students and X1 indicates school-level average mindset score of the students. X2 is a school-level continuous covariate that measures the school's achievement level and C1 is a categorical covariate for race/ethnicity. We measure closeness in continuous covariates using the same mean absolute difference metric used in Figure 1(a) in the main text. Whereas, we measure

Table 1: Details of Experiments in Sections 6 and 7. The *Additional Information* column indicates if further details for that experiment are included in Section C.2.

Section	**Dataset**	**# Samples**	**# Covariates**	**K**	η	**Additional Notes**
6.1: Accuracy and Auditability	ACIC 2018 Learning Mindset Dataset	10,000	10	10	2	Y
6.2: Nonlinear Outcome	Sine	5000	100	10	10	
	Exponential	5000	100	10	10	
6.3: Scalability	Linear + Quadratic	Varies	Varies	10 (1 for GenMatch - see notes)	2	Y
7.1: Metalearner LCM	Sine	500	10	10	5	
7.2: Feature Importance Matching	Simple Quadratic	500	10	10	5	
7.3: LCM-Augmented-PGM	Linear + Quadratic	5000	20	25 using PGM followed by 5 using LCM	N/A	Y

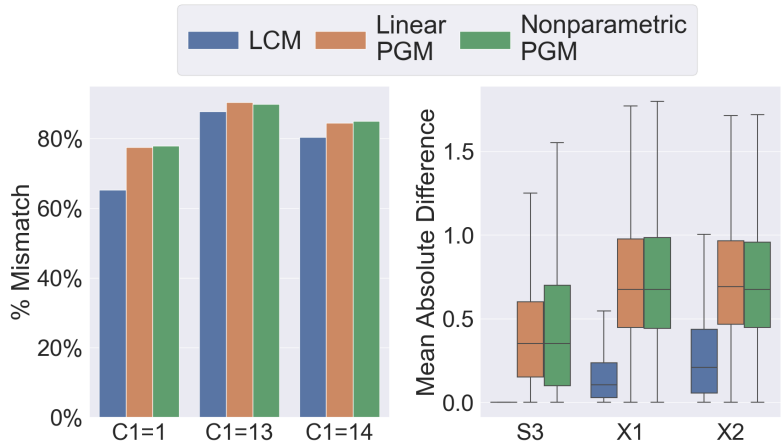

Figure 1: Closeness in important covariates for matched groups produced by LCM, linear PGM, and nonparametric (NP) PGM. Smaller values imply better and tighter matches.

closeness in categorical covariates as the percent of samples in a match group that do not have the same label as the query unit (% Mismatch). LCM matches much more tightly on all of the continuous covariates. For categorical covariates, while LCM matches tighter than PGM methods, it struggles compared to continuous covariates. We theorize this is due to the low occurrence rate of these features. In particular, C1=1 in 9.5%, C1=13 in 1.8% and C1=14 in 6.2% of samples. Therefore, it is difficult to find matches that have the same C1 value and are also similar in all of the other important covariates. LCM sometimes prioritizes matching almost-exactly on other covariates at the expense of these rare categorical covariates.

Carvalho et al. [2019] also states that although XC (Urbanicity) is not an effect modifier it is strongly related to X1 (student's fixed mindsets - summarized at the school level) and X2 (school achievement level) which are true effect modifiers. Because of this, seven of the eight methods that are summarized in Carvalho et al. [2019] identified XC as an effect modifier. Carvalho et al. [2019] further shows that, in this dataset, marginally the true cates for XC=3 are much lower than other values of XC. We show in Figure 2 that LCM also identifies this trend in XC.

For Section 6.1, we did not compare to other almost-matching-exactly methods (i.e. MALTS, AHB, GenMatch) due to the large size of the dataset. The ACIC 2018 Learning Mindset Dataset has 50,000 samples and >100 covariates after encoding the categorical features. Results from Section 6.3 highlight how intractable it would be to run other AME methods on a dataset of this size.

Section 6.2: Nonlinear Outcomes. Figure 3 shows CATE estimation accuracy for the same experiment in Section 6.2 with

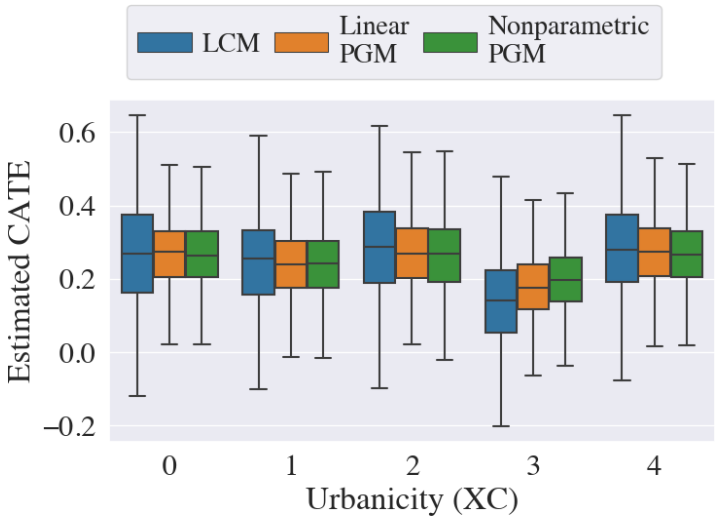

Figure 2: Marginal CATE estimates produced by LCM, Linear PGM, and Nonparametric PGM for the categorical school-level covariate of urbanicty (XC).

the number of covariates increased to 500 for both the **Sine** and **Exponential** datasets. Given that we used 10 splits for this experiment, the training set in each fold had 500 samples. Note that LCM's accuracy does not suffer in this extremely high-dimensional setting where the number of samples equals the number of covariates. These results further highlight the ability of LCM to scale to very high-dimensional data even in the case of nonlinear outcome functions.

Section 7.1: Metalearner LCM. For the Metalearner LCM, here we show the effect of learning unique distance metrics for calculating control vs treated KNNs. We measure the distance between query unit's covariate values and the values of the ten nearest neighbors' of each treatment type. In particular, we calculate the mean absolute difference between a query unit's value and the values of its ten nearest neighbors. As explained in Section 7.1, X1 is a relevant covariate to the outcome under both treatment regimes, whereas X2 is only relevant to the outcome under treatment. X3 is unimportant in both setting and shown as a reference point. Figure 4 shows that while LCM's nearest neighbors are equally close on X0 and X1 in both treatment spaces, Metalearner LCM considers X2 as unimportant when calculating KNNs who are in the control group. This highlights how Metalearner LCM is able to adapt to outcome spaces that are different under different treatment regimes.

LCM vs Machine Learning Methods. Previous almost-matching-exactly literature has established that AME methods perform as well as (and often better than) machine learning methods like BART, causal forest, and double machine learning for estimating CATEs [Parikh et al., 2022, Morucci et al., 2020, Wang et al., 2017]. For this reason, this paper focuses on comparing LCM to matching methods and particularly other AME methods. However, here we include an experiment comparing the CATE estimation accuracy of LCM to various machine learning methods on a high-dimensional non-linear dataset.

We use the Quadratic DGP with 25 relevant covariates, 2 of which are relevant to the treatment choice, and 125 irrelevant covariates. We generate 2500 samples and set $\eta = 5$. We run LCM with two configurations. *LCM Mean* is run with $K = 10$ and uses a mean estimator inside the match groups. *LCM Linear* is run with $K = 40$ and uses linear regression as the estimator inside the match groups. We compare to state-of-the-art machine learning methods double machine learning (DML), causal forest, and BART TLearner. Figure 5 shows that LCM Mean performs on par with the machine learning methods on this dataset, further highlighting the accuracy our method. LCM Linear improves upon LCM Mean, showing that we can achieve better accuracy with more sophisticated estimators if we are willing to increase the size of the match groups.

LCM vs Feature Selection. Here we show CATE estimation accuracy of LCM compared to matching equally on the covariates after feature selection. To compare with LCM, we estimate CATEs using feature selection by simply following the same steps as LCM but replacing the $\mathcal{M}^*$ with an $\mathcal{M} \in \mathbb{R}^{p \times p}$ such that $\mathcal{M}_{l,l} = 1$ when $\mathcal{M}^*_{l,l} > 0$ and $\mathcal{M}_{l,l} = 0$ when $\mathcal{M}^*_{l,l} = 0$. We refer to this method as *LASSO FS*. We also compare to an *Oracle* feature selector in which we assume that we know which covariates are important and match equally only on the important covariates.

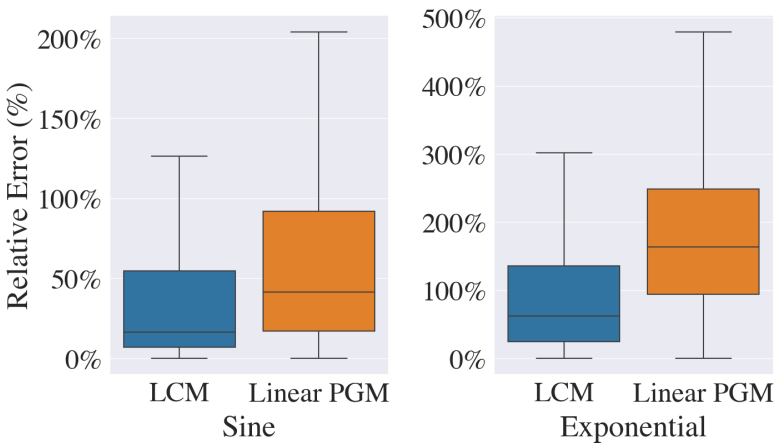

Figure 3: Comparing LCM's and Linear PGM's performances for high-dimensional nonlinear synthetically generated datasets **Sine** and **Exponential**.

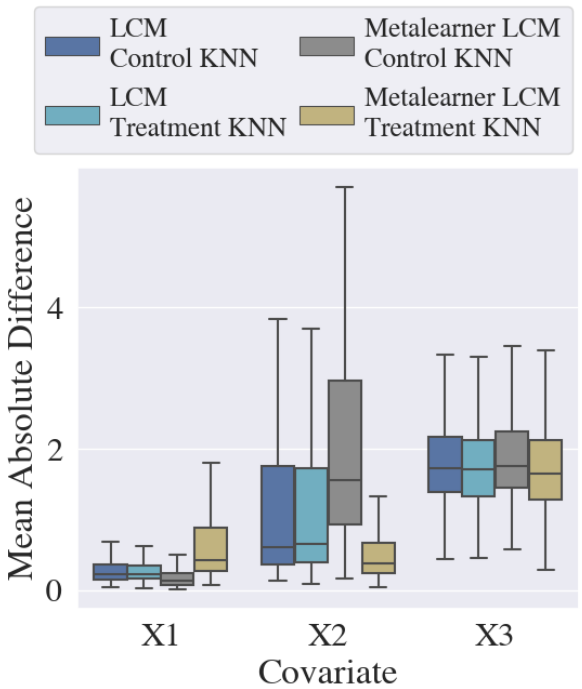

Figure 4: Measure of how tightly the KNN groups are for LCM versus Metalearner LCM under different treatment regimes.

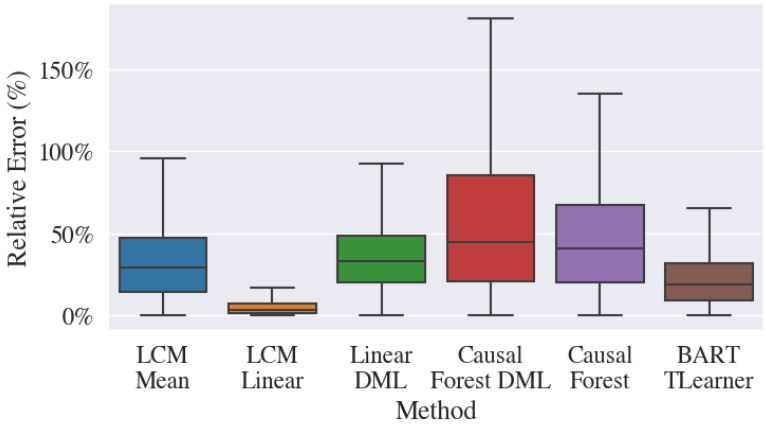

Figure 5: Estimated CATE absolute error relative to the true ATE for LCM Mean, LCM Linear, and state-of-the-art machine learning methods. DML stands for double machine learning.

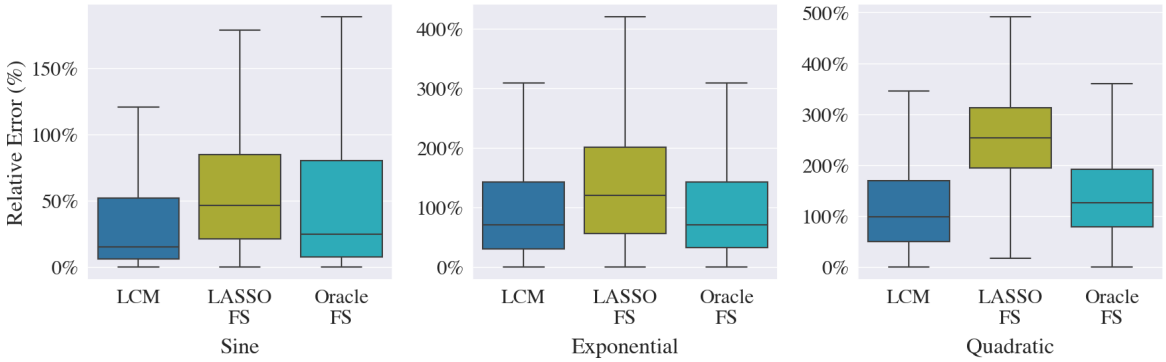

Figure 6: Estimated CATE absolute error relative to the true ATE for LCM and matching equally on covariates after LASSO and Oracle feature selection.

We run our analysis on three of the data generation processes used earlier in this paper. Namely, we run on the **Sine**, **Exponential**, and **Quadratic** DGPs described in Section C.1. We generate 5000 samples and 100 covariates for each DGP and have two important covariates for **Sine**, four important covariates for **Exponential**, and five important covariates for **Quadratic**. All tests set $\eta = 5$ and $K = 10$. Figure 6 shows that LCM outperforms LASSO feature selection and performs on par with an Oracle feature selector. This highlights how using the relative weights of feature importance values in a distance metric, and thus matching tighter on covariates that more heavily contribute to the outcome, ultimately leads to more accurate CATE estimates.

References

Keith Battocchi, Eleanor Dillon, Maggie Hei, Greg Lewis, Paul Oka, Miruna Oprescu, and Vasilis Syrgkanis. EconML: A Python Package for ML-Based Heterogeneous Treatment Effects Estimation. https://github.com/py-why/EconML, 2019. Version 0.14.0.

Carlos Carvalho, Avi Feller, Jared Murray, Spencer Woody, and David Yeager. Assessing treatment effect variation in observational studies: Results from a data challenge, 2019. URL `https://arxiv.org/abs/1907.07592`.

Vincent Dorie, Hugh Chipman, Robert McCulloch, Armon Dadgar, R Core Team, Guido U Draheim, Maarten Bosmans, Christophe Tournayre, Michael Petch, Rafael de Lucena Valle, et al. Package 'dbarts'. 2019.

Ben B. Hansen. The prognostic analogue of the propensity score. *Biometrika*, 95(2):481–488, 2008. ISSN 00063444, 14643510. URL `http://www.jstor.org/stable/20441477`.

Daniel E Ho, Kosuke Imai, Gary King, and Elizabeth A Stuart. Matching as nonparametric preprocessing for reducing model dependence in parametric causal inference. *Political analysis*, 15(3):199–236, 2007.

Almost Matching Exactly Lab. AME-ahb-r-package. `https://github.com/almost-matching-exactly/AHB-R-package`, 2022.

Marco Morucci, Vittorio Orlandi, Sudeepa Roy, Cynthia Rudin, and Alexander Volfovsky. Adaptive hyper-box matching for interpretable individualized treatment effect estimation. In *Conference on Uncertainty in Artificial Intelligence*, pages 1089–1098. PMLR, 2020.

Harsh Parikh. AME-pymalts. `https://github.com/almost-matching-exactly/MALTS`, 2020.

Harsh Parikh, Alexander Volfovsky, and Cynthia Rudin. Malts: Matching after learning to stretch. *Journal of Machine Learning Research*, 23(240), 2022.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Tianyu Wang, Sudeepa Roy, Cynthia Rudin, and Alexander Volfovsky. FLAME: A fast large-scale almost matching exactly approach to causal inference. *arXiv preprint arXiv:1707.06315*, 2017.

Yue Zhang, Soumya Ray, and Weihong Guo. On the consistency of feature selection with lasso for non-linear targets. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, volume 48, page 183–191, 2016.