Comparative Analysis of Machine Learning in Generalized PS Estimation
Link to our Video: Click here to view the video

Bella Qian
Biostatistics
HSPH
yqian@hsph.harvard.edu

Bowen Ma
Biostatistics
HSPH
bowenma@hsph.harvard.edu

Min Guo
Biostatistics
HSPH
minguo@hsph.harvard.edu

1 Introduction

Propensity score (PS) analysis is extensively utilized in social science and clinical research. They serve to balance treatment and control groups by adjusting for a range of covariates, thereby minimizing bias in the assessment of treatment effects in scenarios where random assignment is infeasible[1]. All PS methods center around the calculation of PS, which are a conditional probability that a participant is assigned to the treatment group given a set of observed covariates [2]. Numerous models can estimate Propensity Score (PS), with Logistic Regression (LR) being the most common. However, LR's effectiveness is limited by its linear assumption and struggles with high-dimensional datasets [3]. Non-parametric models like General Boosted Models (GBM), Classification and Regression Trees, Neural Networks, and Native Bayesian methods are effective in analyzing large datasets [4–8]. Despite various ML models being explored for PS estimation in numerous studies, applying these models in PS treatment effect analysis poses distinct challenges. While the focus has primarily been on binary treatments, there's a growing interest in comparing multiple interventions [9, 10]. Some studies have expanded methodologies to multinomial treatments, yet no existing paper comprehensively includes all treatment types in a comparative analysis of ML model performance [4, 11–15]. In this project, we aim to bridge this gap by systematically comparing the ML methods used in binary $\mathcal{B}$, multinomial $\mathcal{M}$, and continuous $\mathcal{C}$ treatment settings. We focus on two prevalent ML algorithms for PS analysis, GBM and deep neural nets (DNN), applying them to both real-world and simulated datasets to highlight the distinctions between these methodologies [3, 6, 16–18].

Generalized PS (GPS), intended for $\mathcal{M}$, calculates the likelihood of receiving each treatment level based on covariates. GBM methods are extensively utilized in PS analysis for binary and pairwise comparison in multinomial datasets [9, 4, 13]. Similar to the $\mathcal{B}$ case, pairwise comparisons are restricted to a few treatment groups (typically around 3); they cannot concurrently evaluate treatment effects, and such comparisons are not feasible for $\mathcal{C}$ dataset [9]. DeepNN can also estimate PS when trained properly, given its capacity to predict the probabilities and capture nonlinear effects with no prerequisite of detailed knowledge [17–19]. The benefits of using DNNs are: (1) They ensure high classification accuracy for high-dimensional data; [6, 19]; (2) Their complexity enables the derivation of smooth polynomial functions without needing prior knowledge, although this may complicate model interpretation; [6, 19]; (3) Flexibility in altering the output layer allows DNNs to adapt for various treatment types, employing sigmoid functions for binary, softmax for multinomial classification, and a single output layer for continuous PS estimation [20]. In this project, section 2 describes GBM, DNN method and analysis in $\mathcal{B}$, $\mathcal{M}$, $\mathcal{C}$ dataset. Section 3 shows how our data is simulated. Section 4 is for result and discussion.

2 Method and Analysis

PS estimation involves using treatment assignment as the dependent variable and selected covariates as independent variables. This estimation is then applied in further analyses like matching, weighting, outcome analysis, and treatment effect comparison across different treatment groups [1, 21].

2.1 Generalized Boosted Models

GBM, a machine learning algorithm, boosts a model's predictive accuracy through sequential training of multiple weak models. This iterative approach employs several regression trees to intricately and nonlinearly relate treatment assignment to pretreatment covariates, avoiding data overfitting [4, 22]. Given the benefits of GBM, we applied it in PS estimation to find the optimal number of trees.

```
Algorithm 1: GBM algorithm
GBM Algorithm:
1.Compute the negative gradient as the working response $z_i = -\left[\frac{\partial}{\partial f(x_i)}\Psi(y_i, f(x_i))\right]_{f(x_i)=f(x_i)}$
2. Randomly select $\rho \times N$ cases from the dataset. Fit a regression tree with $M$ terminal nodes,
   $g(x) = E(z|x)$. This tree is fit using only those randomly selected observations.
3. Compute the optimal terminal node predictions, $\rho_1, \dots, \rho_M$, as
   $\rho_m = \arg\min_\rho \sum_{x_i \in S_m} \Psi(y_i, f(x_i) + \rho)$, where $S_m$ is the set of $x$s that define terminal node $m$. Again,
   this step uses only the randomly selected observations.
4. Update $f(x)$ as $\hat{f}(x) \leftarrow \hat{f}(x) + \lambda\rho_m(x)$, where $m(x)$ indicates the index of the terminal node into
   which an observation with features $x$ would fall.
Bootstrapping to calculate the weighted correlation coefficient (stop method for $\mathcal{C}$)
1. Calculate generalized propensity score $\hat{r}(T_i, X_i)$ using boosting with $M$ trees. Then, calculate
   $w_i = \frac{\hat{r}(T_i)}{\hat{r}(T_i, X_i)}$ for $i = 1, \dots, n$.
2. Sample $n$ observations from the original dataset with replacement. Each data point is sampled with
   the inverse probability weight obtained from the first step. Calculate the Pearson coefficient between
   $T$ and $X_j$ on the weighted sample and denote it as $\bar{d}_{ij}$.
3. Repeat Step (2) $k$ times and get $d_{j1}, d_{j2}, \dots, d_{jk}$. Calculate the average correlation coefficient
   between $T$ and $X_j$, denoted as $\bar{d}_j$.
4. Perform a Fisher transformation on $\bar{d}_j$, $z_j = \frac{1}{2}\ln\left(\frac{1+\bar{d}_j}{1-\bar{d}_j}\right)$.
5. Average $|z_j|$ over all the covariates and get the average absolute correlation coefficient (AACC).
```

In each iteration of the GBM process, one tree is added and its PS and weight are determined. The tree fits the residuals and maximizes data log-likelihood, then combined with shrunken predictions to enhance model smoothness and fit. The function subsequently calculates the stopping criteria, and the optimal number of trees is chosen by evaluating where the lowest stopping criteria statistics appear. Based on various advantages introduced in the video presentation, we choose mean Kolmogorov-Smirnov (KS) statistics as the stopping criteria for $\mathcal{B}/\mathcal{M}$.

2.2 Deep Neural Networks

Deep Neural Networks (DNNs) are recognized as a potent and innovative alternative for the estimation of PS. Traditional methods, such as LR and GBM, have been commonly used for $\mathcal{B}$ PS matching. However, due to the inherent limitations of their algorithms, some setbacks cannot be resolved using these traditional methods. Therefore, we introduce a General PropensityNet algorithm (Algorithm 2) that is capable of handling not only $\mathcal{B}$ but also more complex forms of treatments, such as $\mathcal{M}$ or $\mathcal{C}$.

Algorithm 2: General PropensityNet

```
Data: Dataset $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^N$ with features $X$ and outcome $Y$.
1. Estimate PS using a deep neural network, PropensityNet: $\hat{e}(X) \leftarrow \text{PropensityNet}(X_{\text{train}}, Y_{\text{train}})$;
for each training epoch do
    for each training batch $(X_{batch}, Y_{batch})$ do
        Compute predictions $\hat{Y}_{\text{batch}} \leftarrow f(X_{\text{batch}}; \theta)$;
        Calculate loss $\mathcal{L}(\theta) \leftarrow -\frac{1}{|X_{\text{batch}}|} \sum_i Y_{\text{batch},i} \log(\hat{Y}_{\text{batch},i}) + (1 - Y_{\text{batch},i}) \log(1 - \hat{Y}_{\text{batch},i})$ for
          classification Or $\mathcal{L}(\theta) \leftarrow \frac{1}{|X_{\text{batch}}|} \sum_i (Y_{\text{batch},i} - \hat{Y}_{\text{batch},i})^2$ for regression;
        Update parameters $\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$ using Adam optimization;
    end
end
2. Evaluate the model performance on the test set:
 $\text{loss}, \text{evaluation metrics} \leftarrow \text{PropensityNet.evaluate}(X_{\text{test}}, Y_{\text{test}})$;
```

When handling data with $\mathcal{B}$, traditional methods like LR require the manual creation of interaction terms for non-linear relationships, whereas DNN automatically captures complex interactions using non-linear activation functions (e.g., ReLU, Sigmoid). DNNs, with their multiple hidden layers, automatically extract features and learn hierarchical patterns, progressing from simple to more abstract concepts through layers. This automatic feature extraction and pattern learning capability is not present in LR or GBM.

Moreover, DNNs offer flexibility in design, with varying depths and widths (number of layers and nodes per layer) depending on the complexity and volume of the data. However, DNNs can be prone to overfitting. Techniques like dropout and L2 regularization are employed to prevent the loss function from converging to a minimum that fits the training data noise rather than the underlying distribution. The computation in DNNs involves numerous matrix multiplications, backpropagation, and gradient descent over many parameters, leading to time complexity drawbacks. Specifically, the time complexity of a forward pass for a DNN is $O(N \times d \times h)$, where N is the number of samples, d the dimensional input, and $h = \sum \text{size(hidden layers)}$. In comparison, LR has a complexity of $O(N \times d)$ per iteration of gradient descent, and GBM is about $O(T \times N \times \log(N))$ for T trees. GBM's time complexity can exceed that of DNNs when attempting to capture complex patterns with a large number of trees.

To address the time complexity issue in DNNs, an early stopping callback method is applied. The network is considered converged when the validation loss does not improve for 10 iterations. The weights with the least validation loss are recorded and selected for the final model, significantly improving the runtime of PropensityNet.

3 Simulated Data: flight ticket Price-Demand

The $\mathcal{M}/\mathcal{C}$ dataset we used was inspired by Hartford 2017 description of flight ticket price dataset ($\mathcal{B}$ dataset introduced in Video) [23]. Consider an airline using historical data to optimize ticket prices, where the ticket price is the treatment variable and the customer's purchase decision is the outcome. We assumed that there are 7 customer types $s \in \{1, ..., 7\}$ with different levels of emotion. We modeled the holiday effect on sales by letting the customer's price sensitivity vary continuously throughout the year according to a complex non-linear function 1, The time of year t is generated as $t \sim \text{Unif}(1, 12)$. Ticket prices, formulated in function 3, are strategically set by fuel prices and average price sensitivity. We model this abstractly by generating our latent errors e with a parameter ρ to adjust the correlation between price and errors, with sales being produced according to function 2. We equally separated price as low, medium, and high to create $\mathcal{M}$.

$$\psi_t = 2((t-5)^4/600 + \exp\left[-4(t-5)^2\right] + t/10 - 2) \tag{1}$$

$$y = 100 + (10+p) \cdot s \cdot \psi_t - 2p + e \tag{2}$$

$$p = 25 + (z+3) \cdot \psi_t + v \cdot z,\ v \sim \mathcal{N}(0,1), e \sim \mathcal{N}(\rho v, 1-\rho^2) \tag{3}$$

4 Result & Discussion (Supplement to Video)

In assessing model performance, we used a hold-out testing set and a confusion matrix, as detailed in our video. Both GBM and Propensity Net showed comparable results in predicting $\mathcal{B}/\mathcal{M}$. Although predictions of treatment were similar, variations in PS values between models can impact weights and, consequently, model quality in further analysis. GBM is notably resource-intensive, particularly for $\mathcal{M}$ where multiple models compare treatment levels. Conversely, PropensityNet's run-time is not significantly affected by additional treatment levels, as adjustments are confined to the output layer. Table 1 displays the outcomes of applying GBM and DNN. In DNN, a deeper network architecture leads to near-perfect $\mathcal{B}$ PS estimation. For $\mathcal{M}$, the smaller dataset and feature space necessitate a simpler PropensityNet than the $\mathcal{B}$ model, resulting in slightly less precise fits. The continuous DNN model, while more sensitive to outliers due to its use of mean squared error loss $\text{Loss}_{\text{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2$, results in a better fit to the test data, as indicated by its lower root mean square error (RMSE) score.

Our project evaluates two prominent ML algorithms for PS estimation, widely applied in real-world scenarios. $\mathcal{B}$ (treated vs. untreated) remains the most common tool for hypothesis testing. As clinical practices advance, there is a growing interest in $\mathcal{M}$, especially for its role in comparing various treatment regimens. This aspect is vital in Phase IV clinical trials, which focus on evaluating the safety and effectiveness of drugs. $\mathcal{C}$ is of great value for drug dose-response relationships, such as air pollution and respiratory health, dietary intake, and obesity. Our project's limitations are threefold: (1) The scope of machine learning algorithms included was restricted by time constraints; (2) Testing was limited to fewer than five $\mathcal{M}$ classes, owing to the project's scale; (3) Implementation with more complex or real-world $\mathcal{M}/\mathcal{C}$ dataset was not feasible due to funding and time limitations. In the comparison of GBM and DNN, both showed high accuracy in $\mathcal{B}/\mathcal{M}$ contexts, with DNN being easier to implement, particularly for output layer function changes. In $\mathcal{C}$, DNN excelled over GBM in accuracy and efficiency of implementation.

Table 1: Results

Method		Binary	Multinomial	Continuous
GBM	n.trees	3308	[380,496,1795]	12363
	step size	0.015	0.015	0.1
	Stop Method (minimize)	ks.mean	ks.mean	AACC
	Results	0.0320	[0.2620,0.4535,0.2898]	0.9450
DNN	Input Layer	47	4	4
	Hidden Layers	[94, 1175, 293, 94]	[32, 64]	[32, 64, 128, 64]
	Output Layer	(1, Sigmoid)	(3, Softmax)	1
	Dropout	0.3	0.1	0.3
	Loss Function	Binary CrossEntropy	Sparse CrossEntropy	MSE
	Evaluation Metrics	Accuracy+AUC	Accuracy	RMSE+MAE
	Results (Loss|Metrics)	0.0364 | 99.4%	0.2553 | 88.9%	1.9060 | 1.3637

5 Collaboration

Software used including Rstudio Version 4.3.1 and Python 3.11.5. Specifically, GBM model implementation requires Rstudio, and DNN model implementation requires Python and its corresponding dependencies.

RStudio Team (2023). RStudio: Integrated Development for R. RStudio, PBC, Boston, MA URL http://www.rstudio.com/; Python Software Foundation. Python Language Reference, version 3.11.5. Available at http://www.python.org

Guo Min handles the application of GBM in binary and multinomial treatment settings. Bella Qian oversees GBM for continuous treatment and Monte Carlo data simulation. Bowen Ma is in charge of the implementation of DNN across binary, multinomial, and continuous treatments. All authors have collectively contributed to the literature review, presentation, and the writing of the project.

Professor Shen Shen and Tommi Jaakkola helped us substantially to formulate our project topic and layout.

References

[1] Miguel A. Hernán and James M. Robins. *Causal Inference: What If.* Chapman and Hall/CRC, Boca Raton, 2020

[2] Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983

[3] Daniel F. McCaffrey, Greg Ridgeway, and Andrew R. Morral. Propensity score estimation with boosted regression for evaluating causal effects in observational studies. *Psychological Methods*, 9(4):403, 2004

[4] Daniel F McCaffrey, Beth Ann Griffin, Daniel Almirall, Mary Ellen Slaughter, Rajeev Ramchand, and Lane F Burgette. A tutorial on propensity score estimation for multiple treatments using generalized boosted models. *Statistics in medicine*, 32(19):3388–3414, 2013

[5] Brian K Lee, Justin Lessler, and Elizabeth A Stuart. Improving propensity score weighting using machine learning. *Statistics in medicine*, 29(3):337–346, 2010

[6] Daniel Westreich, Justin Lessler, and Michele Jonsson Funk. Propensity score estimation: neural networks, support vector machines, decision trees (cart), and meta-classifiers as alternatives to logistic regression. *Journal of clinical epidemiology*, 63(8):826–833, 2010

[7] Massimo Cannas and Bruno Arpino. A comparison of machine learning algorithms and covariate balance measures for propensity score matching and weighting. *Biometrical Journal*, 61(4):1049–1072, 2019

[8] Romain Pirracchio, Maya L Petersen, and Mark Van Der Laan. Improving propensity score estimators' robustness to model misspecification using super learner. *American journal of epidemiology*, 181(2): 108–119, 2015

[9] Mohammed Shurrab, Dennis T Ko, Cynthia A Jackevicius, Karen Tu, Allan Middleton, Faith Michael, and Peter C Austin. A review of the use of propensity score methods with multiple treatment groups in the general internal medicine literature. *Pharmacoepidemiology and Drug Safety*, 2023

[10] Yeying Zhu, Donna L Coffman, and Debashis Ghosh. A boosting algorithm for estimating generalized propensity scores with continuous treatments. *Journal of causal inference*, 3(1):25–40, 2015

[11] Kosuke Imai and David A Van Dyk. Causal inference with general treatment regimes: Generalizing the propensity score. *Journal of the American Statistical Association*, 99(467):854–866, 2004

[12] Guido W Imbens. The role of the propensity score in estimating dose-response functions. *Biometrika*, 87(3):706–710, 2000

[13] Lane Burgette, Beth Ann Griffin, and Dan McCaffrey. Propensity scores for multiple treatments: A tutorial for the mnps function in the twang package. *R package. Rand Corporation*, 2017

[14] Marieke Dingena Spreeuwenberg, Anna Bartak, Marcel A Croon, Jacques A Hagenaars, Jan JV Busschbach, Helene Andrea, Jos Twisk, and Theo Stijnen. The multiple propensity score as control for bias in the comparison of more than two treatment arms: an introduction from a case study in mental health. *Medical care*, pages 166–174, 2010

[15] Jeremy A Rassen, Abhi A Shelat, Jessica M Franklin, Robert J Glynn, Daniel H Solomon, and Sebastian Schneeweiss. Matching by propensity score in cohort studies with three treatment groups. *Epidemiology*, pages 401–409, 2013

[16] Yun Ju, Guangyu Sun, Quanhe Chen, Min Zhang, Huixian Zhu, and Mujeeb Ur Rehman. A model combining convolutional neural network and lightgbm algorithm for ultra-short-term wind power forecasting. *IEEE Access*, 7:28309–28318, 2019. doi: 10.1109/ACCESS.2019.2901920

[17] Seungman Kim, Jaehoon Lee, and Kwanhee Jung. Propensity score estimation using neural networks: A comparison of dnn, cnn, and logistic regression. 2023

[18] Vikas Ramachandra. Deep learning for causal inference. *arXiv preprint arXiv:1803.00149*, 2018

[19] Zachary K Collier, Walter L Leite, and Allison Karpyn. Neural networks to estimate generalized propensity scores for continuous treatment doses. *Evaluation review*, page 0193841X21992199, 2021

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016

[21] Vikas Ramachandra. Deep learning for causal inference, 2018

[22] Greg Ridgeway. Generalized boosted models: A guide to the gbm package. *Update*, 1(1):2007, 2007

[23] Jason Hartford, Greg Lewis, Kevin Leyton-Brown, and Matt Taddy. Deep iv: A flexible approach for counterfactual prediction. In *International Conference on Machine Learning*, pages 1414–1423. PMLR, 2017