A Permutation Approach to Assess Confounding in Machine Learning Applications for Digital Health

Elias Chaibub Neto
elias.chaibub.neto@sagebase.org
Sage Bionetworks
Seattle, WA

Abhishek Pratap
Sage Bionetworks
University of Washington
Seattle, WA

Thanneer M Perumal
Meghasyam Tummalacherla
Sage Bionetworks
Seattle, WA

Brian M Bot
Sage Bionetworks
Seattle, WA

Lara Mangravite
Sage Bionetworks
Seattle, WA

Larsson Omberg
Sage Bionetworks
Seattle, WA

ABSTRACT

Machine learning applications are often plagued with confounders that can impact the generalizability of the learners. In clinical settings, demographic characteristics often play the role of confounders. Confounding is especially problematic in remote digital health studies where the participants self-select to enter the study, thereby making it difficult to balance the demographic characteristics of participants. One effective approach to combat confounding is to match samples with respect to the confounding variables in order to improve the balance of the data. This procedure, however, leads to smaller datasets and hence negatively impact the inferences drawn from the learners. Alternatively, confounding adjustment methods that make more efficient use of the data (such as inverse probability weighting) usually rely on modeling assumptions, and it is unclear how robust these methods are to violations of these assumptions. Here, instead of proposing a new method to control for confounding, we develop novel permutation based statistical tools to detect and quantify the influence of observed confounders, and estimate the unconfounded performance of the learner. Our tools can be used to evaluate the effectiveness of existing confounding adjustment methods. We evaluate the statistical properties of our methods in a simulation study, and illustrate their application using real-life data from a Parkinson's disease mobile health study collected in an uncontrolled environment.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning**; • **Mathematics of computing** → **Probability and statistics**; **Probabilistic inference problems**; • **Applied computing** → **Health informatics**;

KEYWORDS

Confounding; machine learning; restricted permutations; permutation tests; digital health

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08.
https://doi.org/10.1145/3292500.3330903

ACM Reference Format:
Elias Chaibub Neto, Abhishek Pratap, Thanneer M Perumal Meghasyam Tummalacherla, Brian M Bot, Lara Mangravite, and Larsson Omberg. 2019. A Permutation Approach to Assess Confounding in Machine Learning Applications for Digital Health. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3292500.3330903

1 INTRODUCTION

Machine learning (ML) algorithms have been increasingly used as diagnostic tools in biomedical research[7, 8, 10]. The widespread availability of smartphones and other health tracking devices generates high volumes of sensor data, and makes machine learning uniquely well posed to impact clinical research using digital health tools. In clinical applications, gender, age, and other demographic characteristics of the study participants often play the role of confounders. Confounding is particularly prevalent in mobile health studies run under uncontrolled conditions outside clinical and laboratory settings, where we have little control over the demographic and clinical characteristics of the cohort of participants that self-select to participate in a study.

In the context of predictive modeling, we define a confounder as a variable that causes spurious associations between the features and response variable. In machine learning applications, the presence of confounding can lead to ambiguous inference and poor generalizability of models. Confounding is usually present when the joint probability distribution of the confounder and response variables is different in the data available to develop the learner (which we from now on denote as the "development dataset") relative to the population where the learner will be applied (denoted as the "target population")[22]. For example, consider a diagnostic application where most cases are old aged while most controls are young, but where age is not associated with disease status in the target population (e.g., the target population is composed of older patients only). If the classifier can more efficiently detect age-related signals than disease-related signals, then it will likely perform poorly when deployed in the target population.

Confounding adjustment is an active area of research in machine learning. The goal is to prevent an algorithm from learning the confounding signal. Since any variable that confounds the feature-response relationship has to be associated with both the features and the response, most of the methods proposed in the literature can

be divided into two approaches: (i) methods that remove the association between the confounder and the response; or (ii) methods that remove the association between the confounder and the features. A standard example of the first approach is to match subjects from the development data in order to obtain a subsample that more closely resembles the target population. This strategy, however, results in a smaller number of participants to train and evaluate the machine learning algorithm, and, in highly unbalanced situations, might lead to the exclusion of most of the participants from the analyses. Alternative methods that make more efficient use of the data include inverse probability weighting approaches[18, 22], which weight the training samples in order to make model training better tailored to the target population. A canonical example of the second approach (i.e., reduce the association between the confounders and the features) is to separately regress each feature on the confounders, and use the residuals as the predictors in the machine learning algorithm. Other approaches that do not fall into categories (i) or (ii) include penalized learners[16] and backdoor adjustment[13].

In this paper, we present statistical methods to detect and quantify the influence of observed confounders, and to estimate the actual (i.e., unconfounded) predictive performance of a learner. We use a large Parkinson's digital health study cohort to illustrate how our methods can be used to evaluate the effectiveness of standard confounding adjustment methods.

2 METHODS

We adopt restricted permutations[9, 22] to isolate the contribution of the confounder from the predictive performance of a learner. The key idea is to shuffle the response data within the levels of a categorical/ordinal confounder (as illustrated in the Figure 1) in order to destroy the direct association between the response and the features while still preserving the indirect association due to the confounder. Algorithm 1 describes the procedure for an arbitrary performance metric, m (such as the area under the receiver operating characteristic curve, AUC, or root mean square error).

Algorithm 1 Restricted Monte Carlo permutation null distribution for performance metric m

1: **Input**: Number of permutations, b. Development data set feature matrix, response vector, and confounder vector, $\boldsymbol{X}$, $\boldsymbol{y}$, $\boldsymbol{c}$. Training and test set indexes, i_{train}, i_{test}
2: Split $\boldsymbol{X}$, $\boldsymbol{y}$ and $\boldsymbol{c}$ into training and test sets
3: **for** $i = 1, 2, \ldots, b$ **do**
4: $\quad \boldsymbol{y}^*_{train} \leftarrow \text{RestrictedShuffle}(\boldsymbol{y}_{train}, \boldsymbol{c}_{train})$[1]
5: $\quad \boldsymbol{y}^*_{test} \leftarrow \text{RestrictedShuffle}(\boldsymbol{y}_{test}, \boldsymbol{c}_{test})$
6: $\quad$ Train a ML algorithm on the $\boldsymbol{X}_{train}$ and $\boldsymbol{y}^*_{train}$ data
7: $\quad$ Evaluate the algorithm on the $\boldsymbol{X}_{test}$ and $\boldsymbol{y}^*_{test}$ data
8: $\quad$ Compute the performance metric, m^*_i, on the shuffled data
9: **end for**
10: **Output**: $m^*_1, m^*_2, \ldots, m^*_b$
11: [1]The pseudo code for the `RestrictedShuffle` function is presented in the Supplement.

Building upon the restricted permutation null distribution, we developed two statistical tools to deal with confounding:

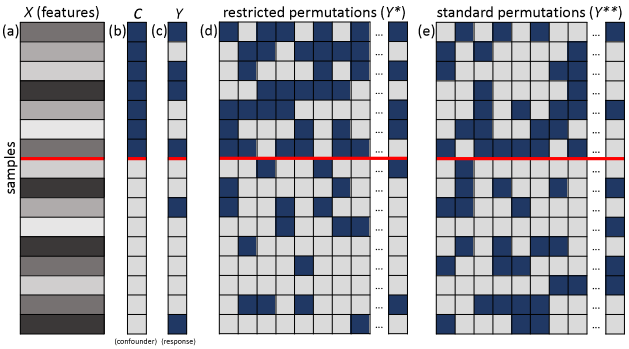

Figure 1: Panels a, b, and c, represent the features (X), confounder (C), and response (Y) data, respectively. In this cartoon example, we have 16 samples, and both C and Y are binary (light and dark cells represent 0 and 1 values, respectively). The confounder vector (panel b) was sorted, and the red line splits the data relative to the levels of C (i.e., the top 7 samples have confounding value 1, while the bottom 9 have confounding value 0). Note that in panel c we have 4 positive response values (dark cells) above the red line, and 2 below it. Panel d illustrates the restricted permutation scheme. Each column shows a distinct permutation. In all permutations, we still have 4 dark cells above the red line and 2 below it. The restricted permutations destroy the association between Y and X, while still preserving the association between Y and C. Panel e illustrates the standard permutation scheme, where we shuffle the response values freely across the entire response vector (now, each column is no longer constrained to have 4 dark cells above the red line and 2 below it). The standard permutations destroy the association between Y and C and between Y and X.

(i) First, we estimate the "unconfounded" predictive performance of a learner by building a mapping from the restricted permutation null to the standard permutation null (where the standard permutation null distribution is generated by shuffling the labels in the usual unconstrained manner). As fully described in Section 2.1 (see below), for any performance metric that can be expressed as a (generalized) U-statistic[6, 11, 15, 25] (e.g., AUC), or expressed as a simple average (e.g., mean square error, mean absolute error, and classification accuracy), we have that an asymptotic estimate of the unconfounded performance metric is given by,

$$\hat{m}_u = (m_o - a_{\hat{\pi}^*}) \frac{s_{\hat{\pi}^{**}}}{s_{\hat{\pi}^*}} + a_{\hat{\pi}^{**}}, \quad (1)$$

where m_o represents the uncorrected metric value; $a_{\hat{\pi}^*}$ and $s^2_{\hat{\pi}^*}$ represent the sample average and variance of the restricted permutation null; and $a_{\hat{\pi}^{**}}$ and $s^2_{\hat{\pi}^{**}}$ represent the analogous quantities for the standard permutation null.

(ii) Second, by noticing that the location of the restricted permutation null provides a natural measure of the amount of confounding signal learned by the algorithm, we adopt the average of the restricted permutation null as a test statistic,

and develop a statistical test to compare the hypotheses,

$$H_0^c : \text{the ML algorithm has not learned the confounding signal ,} \qquad (2)$$
$$H_1^c : \text{the ML algorithm has learned the confounding signal ,}$$

and detect confounding learning per se. Section 2.2 (see below) describes this statistical test in detail.

2.1 The unconfounded metric estimate

The observed metric m_o captures the contributions of both response and confounder learning. In order to estimate the "unconfounded" value, m_u, we need to determine what value would the observed performance metric have assumed, had the response variable not been associated with the confounder. In other words we need to map a value sampled from a distribution where the response and confounder are associated to a distribution where they are not. To this end, we construct a mapping from the restricted permutation null distribution (where the association between the response and the confounder is preserved) to the standard permutation null (where this association is removed).

Let F_{π^*} and $F_{\pi^{**}}$ represent, respectively, the restricted and standard permutation null distributions, and $\hat{F}_{\hat{\pi}^*}$ and $\hat{F}_{\hat{\pi}^{**}}$ represent the respective Monte Carlo versions of these permutation distributions. An obvious mapping would be to define $m_u = m_o - a_{\hat{\pi}^*} + a_{\hat{\pi}^{**}}$, where $a_{\hat{\pi}^*}$ and $a_{\hat{\pi}^{**}}$ correspond, respectively, to the sample mean of $\hat{F}_{\hat{\pi}^*}$ and $\hat{F}_{\hat{\pi}^{**}}$. This mapping, however, only focus on the means and fails to take into consideration the different spreads of the restricted and standard permutation null distributions. Ideally, we should define a mapping that accounts for the entire probability distributions. Therefore, we define and estimate the unconfounded metric m_u by equating $F_{\pi^{**}}(m_u)$ to $F_{\pi^*}(m_o)$,

$$F_{\pi^{**}}(\hat{m}_u) = F_{\pi^*}(m_o) \Leftrightarrow \hat{m}_u = F^{-1}_{\pi^{**}}\left(F_{\pi^*}(m_o)\right) . \qquad (3)$$

Note that, equating $F_{\pi^*}(m_o)$ to $F_{\pi^{**}}(m_u)$ is equivalent to equating the p-values, as illustrated in Figure 2.

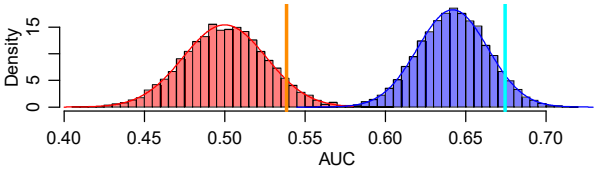

Figure 2: The figure shows an example of the restricted (blue) and standard (red) permutation null distributions for the AUC metric. The cyan line represents the observed AUC value (m_o), while the orange line shows the unconfounded estimate ($\hat{m}_u$). Note that the tail probabilities to the right of the cyan and orange lines are the same (i.e., the p-values are preserved).

In general, $\hat{F}_{\hat{\pi}^*}$ and $\hat{F}_{\hat{\pi}^{**}}$ are unknown distributions. However, because popular performance metrics such as the mean square error, mean absolute error, and the classification accuracy correspond to averages, while metrics such as the AUC correspond to generalized U-statistics[6, 15], we have that the distribution of these statistics can be well approximated by Gaussian distributions when the test set is large enough (due to central limit theorems associated with averages, and to the asymptotic normality of (generalized) U-statistics[11, 25]). Hence, in practice, we will often be able to approximate $\hat{F}_{\hat{\pi}^*}$ and $\hat{F}_{\hat{\pi}^{**}}$ by,

$$\hat{F}_{\hat{\pi}^*} \approx N\left(a_{\hat{\pi}^*}, s^2_{\hat{\pi}^*}\right) , \qquad \hat{F}_{\hat{\pi}^{**}} \approx N\left(a_{\hat{\pi}^{**}}, s^2_{\hat{\pi}^{**}}\right) , \qquad (4)$$

where $s^2_{\hat{\pi}^*}$ and $s^2_{\hat{\pi}^{**}}$ correspond, respectively, to the sample variances of $\hat{F}_{\hat{\pi}^*}$ and $\hat{F}_{\hat{\pi}^{**}}$, and $a_{\hat{\pi}^*}$ and $a_{\hat{\pi}^{**}}$ represent, as before, the respective sample averages. (The blue and red densities on top of the histograms in Figure 2 correspond, respectively, to the normal approximations in (4).) Now, by replacing F_{π^*} and $F_{\pi^{**}}$ in equation (3) by the approximate Gaussian distributions in (4) we have that,

$$\hat{F}_{\hat{\pi}^{**}}(\hat{m}_u) \approx \Phi\left(\frac{\hat{m}_u - a_{\hat{\pi}^{**}}}{s_{\hat{\pi}^{**}}}\right) = \Phi\left(\frac{m_o - a_{\hat{\pi}^*}}{s_{\hat{\pi}^*}}\right) \approx \hat{F}_{\hat{\pi}^*}(m_o) , \qquad (5)$$

where $\Phi(.)$ represents the cumulative distribution function of a standard normal random variable, and we can estimate $\hat{m}_u$ using the estimator presented in eq. (1).

At this point, it is important to mention that we don't view the unconfounded metric estimation as an adjustment method (in the sense that it does not prevent an algorithm from learning the confounding signal in the first place). It simply quantifies the amount of response signal learned by the algorithm, after the algorithm has had a chance to learn both confounding and response signals.

2.2 A statistical test to detect confounding

In the presence of confounding, the restricted permutation null distribution will be shifted away from the baseline random guess value, and this shift can be used to informally infer the presence of confounding. Here, we present a hypothesis test to formally test the hypotheses presented in eq. (2).

We adopt the sample mean of the restricted permutation null,

$$\bar{M}^* = \frac{1}{b}\sum_{i=1}^{b} M_i^* , \qquad (6)$$

as a test statistic, since it represents a natural measure of confounding. Note that under the null hypothesis that an algorithm has not learned the confounding signal, the restricted permutation null will have the same distribution as the standard permutation null. Hence, for large enough test sets we have that $M^* \approx N(a_{\hat{\pi}^{**}}, s^2_{\hat{\pi}^{**}})$, and our test statistic is asymptotically distributed as,

$$\bar{M}^* \approx N\left(a_{\hat{\pi}^{**}}, s^2_{\hat{\pi}^{**}}/b\right) . \qquad (7)$$

Note that the variance of this null distribution depends on the number of permutations (b) used to generate the restricted permutation null, and gets smaller as we increase b. As a consequence, we can easily obtain a statistically significant result by increasing the number of permutations. In order to avoid this artifact, we replace b by the number of test set samples in the computation of the p-value,

$$\text{p-value} = 1 - \Phi\left(\frac{a_{\hat{\pi}^*} - a_{\hat{\pi}^{**}}}{s_{\hat{\pi}^{**}}/\sqrt{n}}\right) . \qquad (8)$$

By doing so, we guarantee that we will only be able to detect small confounding effects when we are truly well powered to do so. In Section 3, we report the results of a simulation study evaluating the empirical performance of the confounding test.

2.3 Analytical results for the AUC metric

For the AUC metric, additional analytical results are available for approximating the standard permutation null distribution and, in practice, we usually don't need to generate the standard permutation null. Next, we show how these analytical approximations can be used in the computation of the unconfounded AUC estimate and the confounding statistical test.

It has been shown[2] that, when there are no ties in the predicted class probabilities used for the computation of the AUC, the test statistic of the Wilcoxon rank sum test (also known as the Mann-Whitney U test), U, is related to the AUC statistic by, $U = n_n\, n_p(1 - AUC)$, where n_n and n_p represent the number of negative and positive labels in the test set (see Section 2 of reference[19] for details). For large test sets, and under the null hypothesis that the ML algorithm has not learned the response and the confounding signal, this distribution can be approximated[19] by

$$U \;\approx\; N\left(\frac{n_n\, n_p}{2}\;,\; \frac{n_n\, n_p(n_n + n_p + 1)}{12}\right). \tag{9}$$

Now, from the relation $AUC = 1 - U/(n_n\, n_p)$ it follows that,

$$AUC \;\approx\; N\left(\frac{1}{2}\,,\; \frac{n_n + n_p + 1}{12\, n_n\, n_p}\right), \tag{10}$$

so that the standard permutation null distribution, $F_{\pi^{**}}$, can be approximated by the above normal distribution.

Now, by approximating $a_{\hat{\pi}^{**}}$ by 1/2 and $s^2_{\hat{\pi}^{**}}$ by $(n_n + n_p + 1)/(12\, n_n\, n_p)$, we have that the unconfounded AUC estimate is given by,

$$auc_u \;=\; (auc_o - a_{\hat{\pi}^*})\sqrt{\frac{n_n + n_p + 1}{12\, n_n\, n_p\, s^2_{\hat{\pi}^*}}} \;+\; 0.5\,. \tag{11}$$

and the null distribution and p-value for the confounding statistical test can be approximated by,

$$\bar{AUC}^* = b^{-1}\sum_{i=1}^{b} AUC^*_i \;\sim\; N\left(\frac{1}{2}\,,\; \frac{n_n + n_p + 1}{12\, n_n\, n_p\, n}\right), \tag{12}$$

and,

$$p = 1 - \Phi\left(\frac{(a_{\hat{\pi}^*} - 0.5)\sqrt{12\, n_n\, n_p\, n}}{\sqrt{n_n + n_p + 1}}\right), \tag{13}$$

where, as described before, we replaced the number of permutations, b, by the number of samples in the test set, n.

3 SIMULATION EXPERIMENTS

Here, we investigate the statistical power and type I error rates of the confounding statistical test (H_0^c vs H_1^c). We simulated data according to the model in Figure 3, where C represents a binary confounder, Y represents the disease status, X_1, X_2, X_3 represent the features, and θ and β represent, respectively, the confounding and disease effects. In order to generate an association between C and Y (i.e., $C \leftrightarrow Y$) we jointly sample these binary variables from a bivariate Bernoulli distribution[5] (described in Section 7.2 in the Supplement). We performed several simulation experiments based on data generated with:

(i) confounding and disease signal (i.e., under H_1^c and H_1^d);
(ii) confounding but no disease signal (i.e., under H_1^c and H_0^d);
(iii) no confounding or disease signal (i.e., under H_0^c and H_0^d);
(iv) disease but no confounding signal (i.e., under H_0^c and H_1^d).

In each experiment we generated 3,000 data sets. (Details about the simulation parameter choices are provided in Section 7.3 in the Supplement.)

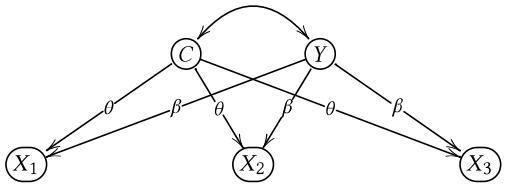

Figure 3: Graphical model representation of the data generation process used to simulate the synthetic datasets employed in the simulation study.

Figures 4a and b report the empirical power curves for data simulated under H_1^c in both the presence of disease signal (i.e., under H_1^d) in panel a, and in the absence of disease signal (i.e., under H_0^d) in panel b. Each panel shows three power curves, corresponding to datasets simulated with increasing amounts of confounding signal, θ. (We estimated empirical power by recording the proportion of times that we rejected the null hypothesis across a grid of nominal significance levels varying from 0 to 1.) As expected, the empirical power to detect confounding increased with the strength of the confounding signal. Figures 4c and d report the distribution of the confounding test p-values for data simulated under the null hypothesis H_0^c in the presence (panel c) and in the absence (panel d) of disease signal. As expected, the distribution is close to the uniform distribution in the $[0, 1]$ interval, showing well controlled type I error rates.

For the sake of completeness, we also checked if our estimator of unconfounded performance was working as expected in the simulations. Figure 5 shows the distributions of the observed AUC scores, AUC_0 (cyan boxplots), and unconfounded estimates, AUC_u (orange boxplots), for each of the four simulation experiments. As expected, in the presence of confounding (panels a and b), the AUC_o values tended to be higher than the AUC_u scores (recall that AUC_o captures the contribution of both the disease and confounder signals, while AUC_u captures only the disease signal). Accordingly, in the absence of confounding (panels c and d), the AUC_o and AUC_u values closely matched each other. Note, as well, that in the absence of disease signal (i.e., under H_0^d) the AUC_u scores tended to be distributed around 0.5 (panels b and d), while the AUC_o scores were still above 0.5 in the presence of confounding signal (panel b), but around 0.5 in the absence of confounding (panel d).

4 REAL DATA ILLUSTRATIONS

A key practical application of our tools is to evaluate if an adjustment method is working as expected. This is important in practice

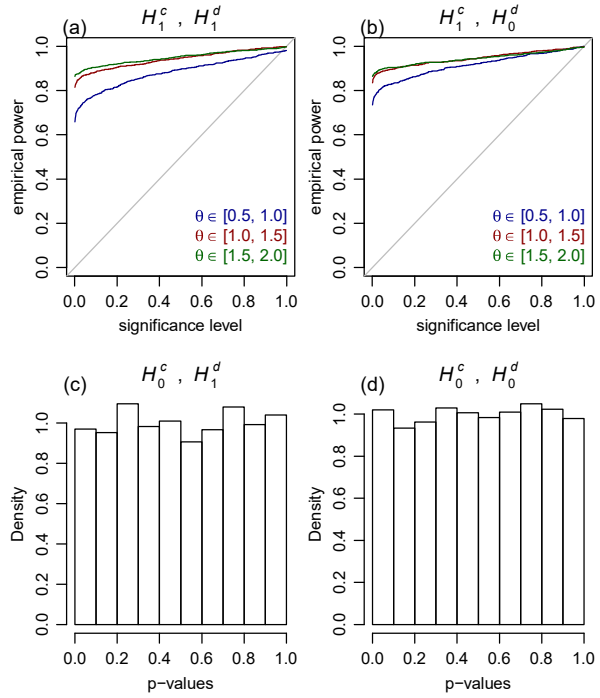

Figure 4: Simulation results. Panels a and b show the empirical power curves for data simulated with confounding (i.e., under H_1^c) in the presence (panel a) and absence (panel b) of disease signal. Panels c and d show the distribution of the p-values for data simulated in the absence of confounding (i.e., under H_0^c) in the presence (panel c) and absence (panel d) of disease signal.

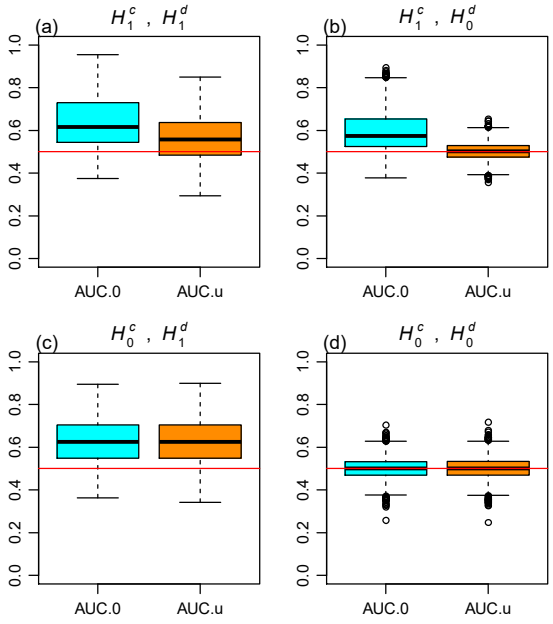

Figure 5: Distributions of the observed AUC scores, AUC_0 (cyan boxplots) and of the unconfounded estimates, AUC_u (orange boxplots), for each of the 4 simulation experiments.

since most of these methods rely on assumptions, and it is generally unclear how robust they are to violations of these assumptions. Here we illustrate the application of our tools to two confounding adjustment methods: sample matching, and approximate inverse probability weighting (IPW) based on the propensity score[24].

Our development data was collected in a digital health study on Parkinsons disease[3, 26] and consists of features generated from 30 second inertial sensor readings captured during walking. We focused on walking, as walking patterns are influenced by age and gender[12] in addition to Parkinson's disease. The development data was split into training and test sets with similar joint distributions for the age, gender, and disease status (Figure 6).

We applied the adjustment methods to both training and test sets, and the analyses were based on a combined gender/discretized age confounder with levels: young male, young female, middle age male, middle age female, senior male, and senior female. Note that while, in theory, we can only perform restricted permutations using categorical/ordinal confounders, in practice we can discretize and evaluate continuous confounders as well. Clearly, if the discretization is too coarse the discretized confounder might not be able to fully capture the association between the confounder and the response, and we might end up underestimating the amount of confounding learned by the algorithm. In practice, one should experiment with distinct discretizations, as illustrated in Figure 7.

Figures 8 and 9 show the results based on logistic regression and random forest classifiers, respectively. In all panels, the blue histograms represent the restricted permutation null distributions generated by Algorithm 1, the red curves represent the normal approximation for the standard permutation null distribution presented in eq. (10), the orange line shows the unconfounded estimate of AUC computed using eq. (11), and the cyan line represents the observed AUC.

For the sake of comparison, panel a in Figures 8 and 9 report the results when no confounding adjustment is performed. Both logistic regression and random forest classifiers are clearly learning confounding signal since the restricted permutation nulls are centered around 0.7, and the confounding test p-values (eq. 13) are highly significant ($p < 10^{-16}$). Hence, the high AUC scores (cyan lines above 0.81) reflect the classifiers' ability to detect both disease and confounding signals, while the unconfounded estimates (orange scores around 0.66) are considerably more modest.

Panel b in Figures 8 and 9 show the results based on a matched subset of participants. The fact that the restricted permutation nulls are centered around 0.5, and closely match the standard permutation null density (red curve), suggests that matching effectively prevented the classifier from learning the confounding signal ($p < 0.51$ and $p < 0.58$, respectively) and that the classifiers are only learning the disease signal. As expected, the observed and unconfounded AUC scores match each other closely in this situation. Finally, note that the much larger spread of the null distributions (in comparison to panel a) is due to the smaller test set available after matching.

Panel c in Figures 8 and 9 report the results for the approximate IPW approach. This method makes use of the entire data set and attempts to prevent confounding learning by weighting the samples according to the inverse of their estimated propensity scores (i.e.,

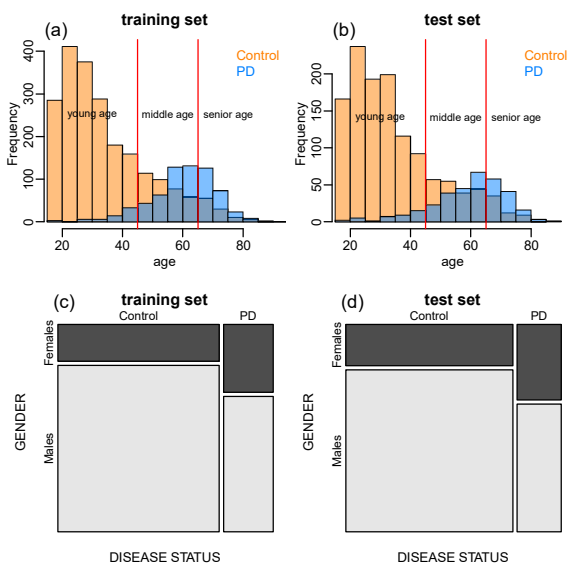

Figure 6: Age and gender associations in the mPower data. Panels a and b show that, for both training and test sets, the age distributions of PD and control participants have reduced overlap, with control participants being usually younger than PD participants leading to a strong association between age and disease status. Panels c and d present mosaic plot of disease status by gender, showing again an association between these variables (with a larger proportion of female participants in the PD group than in the control group). The training and test sets are composed, respectively of 658 and 331 cases and 2,144 and 1,255 controls.

the conditional probability that a participant has the disease given its gender and age). While several approaches have been proposed in the literature for the estimation of propensity scores[14, 20], here, we adopt the most commonly used method based on logistic regression. (A detailed description of the approximate IPW approach is presented in Section 7.4 in the Supplement.) The panels show that the approximate IPW approach managed to reduce the amount of confounding (the blue histograms are closer to 0.5 compared to panel a). However, it didn't remove it completely ($p < 10^{-16}$). This suggests that the estimated inverse probability weights did not generate a well balanced augmented data set. (Figure 10 confirms this is indeed the case.) Most likely, the reason for this suboptimal performance is that propensity score estimation using logistic regression makes the strong assumption that the observed associations between the confounders and disease labels can be well described by the logistic function. This example illustrates how the violation of a parametric modeling assumption can lead to an inefficient confounding adjustment.

5 ACCOUNTING FOR THE CONFOUNDER / RESPONSE ASSOCIATION STRUCTURE IN THE TARGET POPULATION

For the sake of clarity, our illustrations have focused on the case where confounder and response are associated in the development

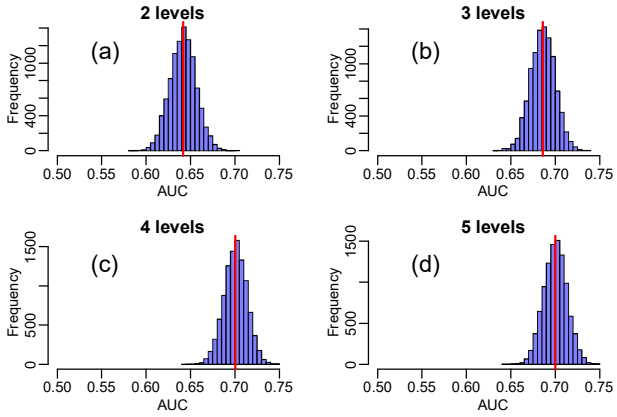

Figure 7: Here, we illustrate how the granularity of the discretization can influence the amount of confounding signal detected by the restricted permutation null. As pointed out in the main text, if the discretization is too coarse the discretized confounder might not be able to fully capture the association between the confounder and the response, and we might underestimate the amount of confounding learned by the algorithm. Here, we experiment with distinct discretizations of the age confounder, namely, categorizing age into 2, 3, 4, and 5 levels. (These level categorizations are given, respectively, by the following age ranges $\{[18, 58], [59, 99]\}$, $\{[18, 44], [45, 65], [66, 99]\}$, $\{[18, 35], [36, 50], [51, 65], [66, 99]\}$, and $\{[18, 30], [31, 45], [46, 60], [61, 75], [76, 99]\}$.) Inspection of the results suggest that the discretization based on 4 levels seems to be enough for this feature set, as increasing the discretization to 5 levels does not shift the restricted null. Clearly, splitting age into 2 levels is not enough since the restricted permutation null is located at much lower AUC values, showing that a fair amount of confounding signal was not captured by this coarse discretization. Splitting age into 3 levels still seem to miss some of the association, as we obtain a slightly stronger confounding signal using 4 levels. (In the illustrations from this paper, however, we continue to use the 3 level categorization, as it already captures enough age signal.)

set but not in the target population. We can, however, still apply our methodology when response and confounder are known to be associated in the target population but have a different joint probability distribution compared to the development data.

To account for the association structure in the target population, we need to derive a baseline null distribution that preserves this structure, and then use this distribution to replace the standard permutation null in our tools. For concreteness, we present next a synthetic data example describing the approach.

Suppose that it is known, a priori, that a disease affects one third of the population and is two times more common in males than in females in the target population. The mosaic plot in Figure 11a describes the joint distribution of gender and disease status in the target population. Suppose, as well, that we have access to a

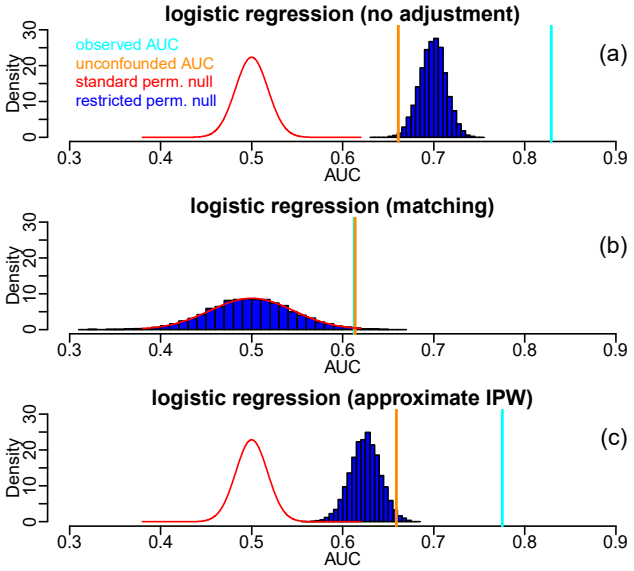

Figure 8: Comparison of confounding adjustment methods using the logistic regression classifier. In all panels, the blue histogram represents the restricted permutation null, the red curve represents the normal approximation for the standard permutation null, and the cyan and orange lines show, respectively, the observed and unconfounded AUC scores.

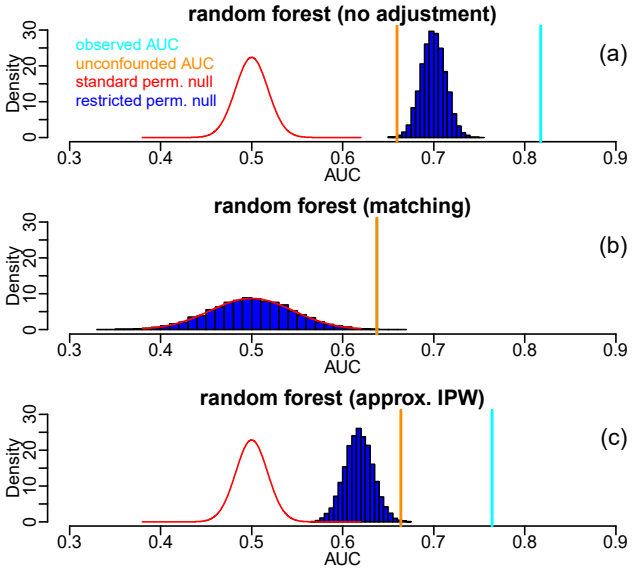

Figure 9: Comparison of confounding adjustment methods using the random forest classifier.

development set containing 10,000 samples, but that, due to self-selection mechanisms, gender and disease status are more strongly associated in the development dataset than in the target population. Figure 11b shows a mosaic plot describing the joint distribution of gender and disease status in the development dataset. (The data was generated as described in Section 7.2 in the Supplement.)

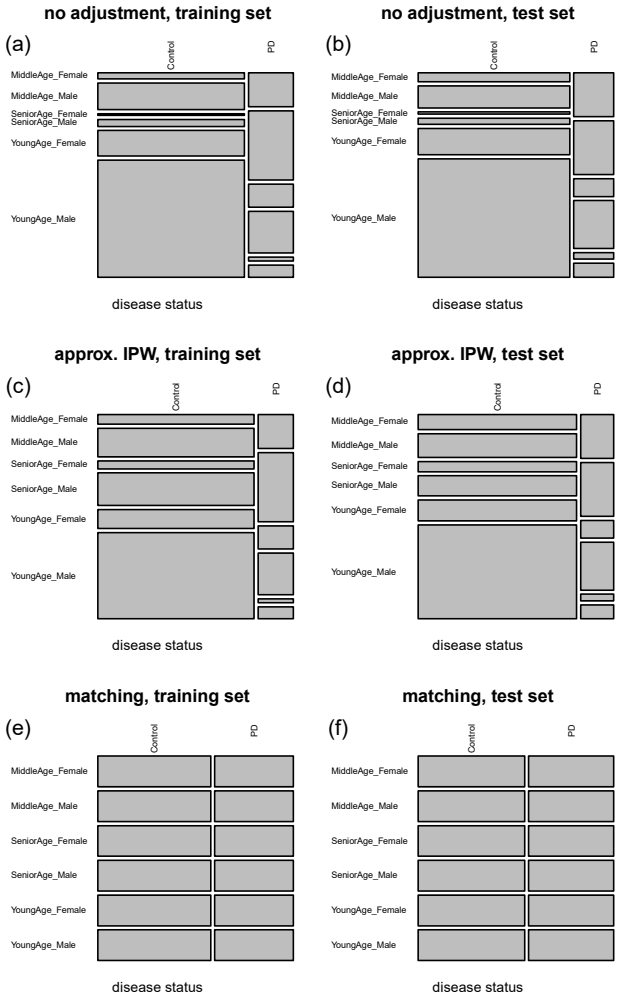

Figure 10: Checking confounder balancing achieved by the approximate IPW and the matching approaches. The panels show mosaic plots for the combined age/gender confounder versus the disease status (age was discretized into young, middle, and senior age categories). For the sake of comparison, panels a to b show the results for the original data. Panels b to c show the respective plots for the augmented training and test sets generated by the approximate IPW method. While the method clearly improved the balance (in comparison to the results in the top panels), it still did not manage to generate truly well balanced training and test sets. Panels e and f show the results for the matching approach. The mosaic plots show a perfect balance for the combined age/gender confounder versus the disease status.

To account for the association structure in the target population, we first sub-sample (from the development population) a training and a test set showing the same joint distribution of gender and disease status as the target population. Figures 11c and d show the mosaic plots for these baseline sets. Next, we apply restricted permutations to these subsets in order to generate the baseline

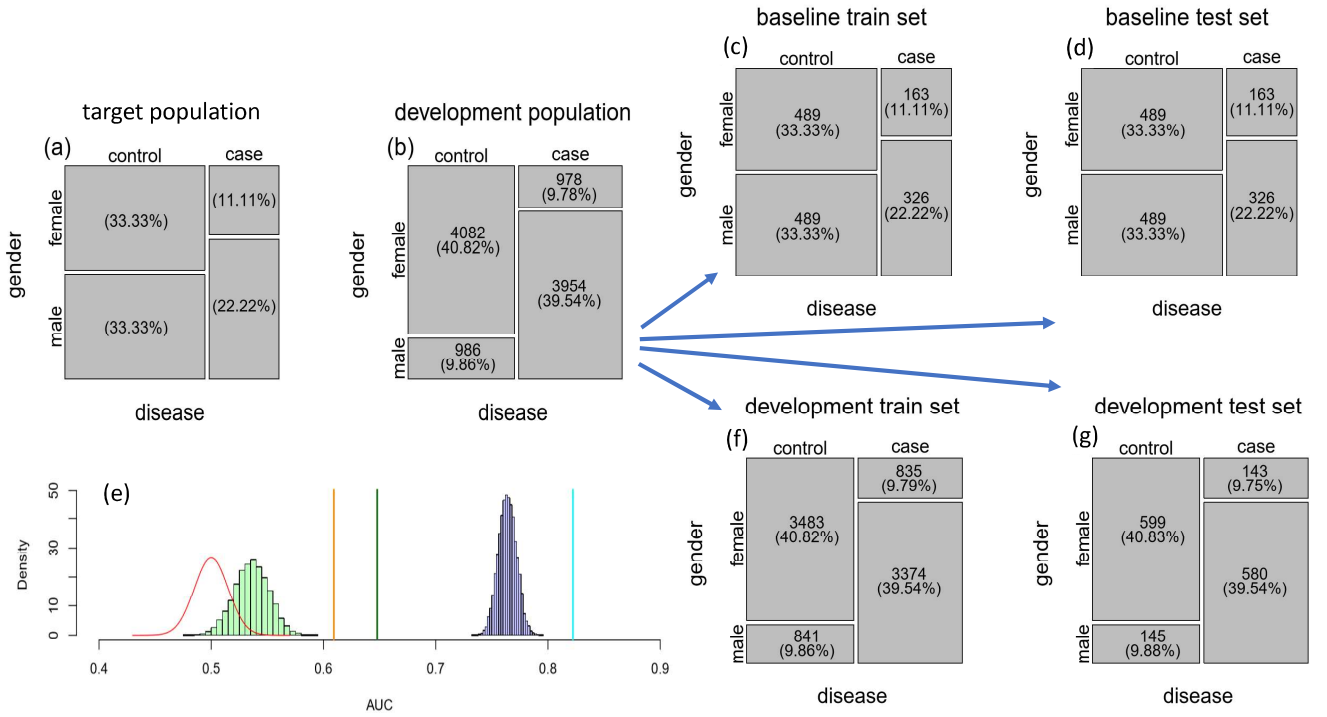

Figure 11: An example on how to account for the confounder/response association structure in the target population. See the text for details.

null distribution (green histogram in Figure 11e), which captures the gender/response association structure of the target population. (Note how this null distribution is shifted away from 0.5, due to the association between gender and disease status.)

Once the association structure in the target distribution has been quantified, the next step is to measure the amount of association in the development data. This is so that we can quantify how much of the observed predictive performance of the classifier is actually due to the stronger association structure in the development data. To this end, we generate a restricted permutation null distribution based on a random split of development dataset into training and test sets that preserve the joint distribution observed in Figure 11b. We call these subsets the "development training set" and the "development test set". Figures 11f and g show the respective mosaic plots. The blue histogram in Figure 11e shows the restricted permutation null derived from the development training and test sets[1].

Now, in order to compute the unconfounded predictive performance of the classifier (relative to the target population) we only need to use the baseline null distribution (green histogram in Figure 11e) in place of the standard permutation null (red density in Figure 11e). For instance, setting a_b and s_b to represent the mean and standard deviation of the baseline null, and letting $a_{\hat{\pi}^*}$ and $s_{\hat{\pi}^*}$ represent, as before, the mean and standard deviation of the restricted permutation null (blue histogram in Figure 11e), we can estimate the unconfounded AUC value as,

$$(AUC_o - a_{\hat{\pi}^*})\frac{s_b}{s_{\hat{\pi}^*}} + a_b \ . \tag{14}$$

The green line in Figure 11e represents the above estimate (while, for the sake of comparison, the orange one shows the estimate with respect to the standard null distribution).

Similarly, we can still test for the presence of confounding (which in this example is measured by the amount to association between the confounder and the response that goes beyond the association present in the target population). To this end, we can use the $N(a_b\,,\, s_b^2/n)$ distribution as an approximate null and compute the p-value for the confounding test as,

$$p \;=\; 1 - \Phi\left(\frac{a_{\hat{\pi}^*} - a_b}{s_b/\sqrt{n}}\right) \ . \tag{15}$$

Note that the estimator in eq. (14) and the p-value in eq. (15) correspond, respectively, to the estimator in eq. (1) and the p-value in eq. (8) with $a_{\hat{\pi}^{**}}$ and $s_{\hat{\pi}^{**}}$ replaced by a_b and s_b.

6 FINAL REMARKS

Digital health enabled diagnostic systems have the potential to provide low cost remote diagnostic tools to underserved communities that lack easy access to medical care. However this opportunity cannot be fully realized without (i) efficient approaches to combat confounding (without which we run the risk of making spurious inferences from the data) and (ii) rigorous methods to evaluate these

[1]Note that, in order to make the restricted null distribution (blue histogram) and baseline null distribution (green histogram) comparable, the development test set should have the same size as the baseline test set. (Recall that the spread of the permutation null distribution decreases with increasing sample sizes.) In Figure 11, both the baseline (panel d) and development test set (panel g) contain 1,467 samples.

adjustment methods. The tools proposed in this paper address the second need.

To the best of our knowledge, the use of restricted permutations in the context of predictive modeling has only been leveraged by[22]. These authors, however, use restricted permutations to test if a ML algorithm has learned the response signal in the presence (or absence) of confounders, but not to detect and quantify confounding learning per se, as proposed in this paper.

While this paper provides a practical approach to assess the influence of potential confounders, a few noteworthy limitations remain. For instance, the approach might become impractical in situations where the feature/response association is confounded by a relatively large number of confounders containing a relatively high number of levels per confounder. Furthermore, because the approach can only investigate the influence of observed confounders, the performance of a ML algorithm might still be biased by unobserved confounders.

All illustrations presented in this paper were based on shallow classifiers. In this setting, the features are fixed and the restricted permutations allow us to evaluate if the ML algorithm was able to learn the confounding signal from the fixed features. Deep models, on the other hand, learn the feature representations from the raw data. As a consequence, if we train a deep model using shuffled labels generated by restricted permutations, the deep model might learn feature representations that can capture the confounding signal[2]. These feature representations, however, do not correspond to the feature representations the deep model would have learned, had the labels not been shuffled. (It is well known that deep models show distinct learning behaviors when trained with totally or partially shuffled labels[1, 27].) Hence, in order to quantify the amount of confounding signal contained in the feature representation learned by a deep model trained under natural conditions, one should first train the model using the original labels, and then use the features learned by the deep model as inputs in a shallow model trained with labels shuffled by restricted permutations. Supplementary Figure S2 presents an illustration using a feature set learned by a deep model as the input variables in a random forest classifier.

The methodology presented in this paper relies on asymptotic approximations and, therefore, requires test sets of reasonable size. We point out, however, that it is still possible to perform a permutation test to check if the ML algorithm has learned the confounding signal in small test set settings (see Section 7.5 in the Supplement for details).

Finally, we point out that while this paper has focused on digital health applications, the proposed tools can be more generally applied to any other areas impacted by confounders.

REFERENCES

[1] Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. 2017. A closer look at memorization in deep networks. *International Conference on Machine Learning (ICML 2017)*.

[2] Donald Bamber. 1975. The area above the ordinal dominance graph and the area below the receiver operating characteristic graph. *Journal of Mathematical Psychology* **12**: 387-415.

[3] Brian M. Bot, Christine Suver, Elias Chaibub Neto, Michael Kellen, Arno Klein, Christopher Bare, Megan Doerr, Abhishek Pratap, John Wilbanks, Ray E. Dorsey, Stephen H. Friend, and Andrew D. Trister. 2016. The mPower study, Parkinson disease mobile data collected using ResearchKit. *Scientific Data* 3:160011 doi:10.1038/sdata.2016.11

[4] Leo Breiman. 2001. Random forests. *Machine Learning*, **45**, 5-32.

[5] Bin Dai, Shilin Ding, and Grace Wahba. 2013. Multivariate Bernoulli distribution. *Bernoulli* **19**: 1465-1483.

[6] Elizabeth R. DeLong, David M. DeLong, and Daniel L. Clarke-Pearson. Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. *Biometrics* **44**: 837-845.

[7] Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. Blau, and Sebastian Thrun. 2017. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **542**: 115-118.

[8] Jeffrey Alan Golden. 2017. Deep learning algorithms for detection of lymph node metastases from breast cancer: helping artificial intelligence be seen. *Jama* **318**: 2184-2186.

[9] Phillip Good. 2000. *Permutation tests: a practical guide to resampling methods for testing hypothesis.* 2nd ed. Springer, New York.

[10] Varun Gulshan, Lily Peng, Marc Coram, Martin C. Stumpe, Derek Wu, Arunachalam Narayanaswamy, Subhashini Venugopalan, Kasumi Widner, Tom Madams, Jorge Cuadros, Ramasamy Kim, Rajiv Raman, Philip C. Nelson, Jessica L. Mega, and Dale R.Webster. 2016. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *Jama* **316**: 2402-2410.

[11] Wassily Hoeffding. 1948. A class of statistics with asymptotically normal distribution. *Annals of Mathematical Statistics* **19**: 293-325.

[12] Seung-uk Ko, Magdalena I. Tolea. Jeffrey M. Hausdorff, and Luigi Ferrucci. 2011. Sex-specific differences in gait patterns of healthy older adults: results from the Baltimore longitudinal study of aging. *Journal of Biomechanics* **44**: 1974-1979.

[13] Virgile Landeiro, and Aron Culotta. 2016. Robust text classification in the presence of confounding bias. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence* (AAAI-16), 186-193.

[14] Brian K. Lee, Justin Lessler, and Elizabeth A. Stuart. 2010. Improving propensity score weighting using machine learning. *Statistics in Medicine* 29: 337-346.

[15] Erich L. Lehmann. 1951. Consistency and unbiasedness of certain nonparametric tests. *Annals of Mathematical Statistics* **22**: 165-179.

[16] Limin Li, Barbara Rakitsch, and Karsten Borgwardt. 2011. ccSVM: correcting Support Vector Machines for confounding factors in biological data classification. *Bioinformatics* **27**: 342-348.

[17] Andy Liaw, and Matthew Wiener. 2002. Classification and regression by randomForest. *R News*, **2**, 18-22.

[18] Kristin A. Linn, Bilwaj Gaonkar, Jimit Doshi, Christos Davatzikos, and Russell T. Shinohara. 2016. Addressing confounding in predictive models with an application to neuroimaging. *International Journal of Biostatistics* **12**: 31-44.

[19] Simon J. Mason, and Nicholas E. Graham. 2002. Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: statistical significance and interpretation. *Quarterly Journal of the Royal Meteorological Society* **128**: 2145-2166.

[20] Romain Pirracchio, Maya L. Petersen, and Mark van der Laan. 2014. Improving propensity scores estimators' robustness to model misspecification using super learner. *American Journal of Epidemiology* **181**: 108-119.

[21] R Core Team. 2018. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

[22] Anil Rao, Joao M. Monteiro, Janaina Mourao-Miranda. 2017. Predictive modelling using neuroimaging data in the presence of confounds. *NeuroImage* **150**: 23-49.

[23] Xavier Robin, Natacha Turck, Alexandre Hainard, Natalia Tiberti, Frederique Lisacek, Jean-Charles Sanchez, and Markus Muller. pROC: an open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinformatics*, **12**, 77.

[24] Paul R. Rosenbaum, and Donald B. Rubin. 1983. The central role of propensity score in observational studies of causal effects. *Biometrika* **70**: 41-55.

[25] Robert J. Serfling. 1980. *Approximation theorems of mathematical statistics.* Jowh Wiley & Sons.

[26] Andrew D. Trister, Ray E. Dorsey, and Stephen H. Friend. 2016. Smartphones as new tools in the management and understanding of Parkinson's disease. *npj Parkinson's Disease* 16006.

[27] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. *International Conference on Learning Representations (ICRL 2017).*

[2]Recall that while the restricted permutations destroy the association between the outcome labels and the inputs, the association between the confounder and the inputs is still preserved.

ACKNOWLEDGEMENTS

Work funded by the Robert Wood Johnson Foundation. Data was contributed by users of the mPower app [doi:10.7303/syn4993293].

7 SUPPLEMENT

Algorithm 2 RestrictedShuffle

Input: Response data vector, $\boldsymbol{y}$; confounder data vector, $\boldsymbol{c}$
Get vector with the levels of $\boldsymbol{c}$, $clevel \leftarrow$ Unique($\boldsymbol{c}$)
Count the number of levels, $n_l \leftarrow$ Length($clevels$)
Initialize vector of shuffled labels, $\boldsymbol{y}^* \leftarrow \boldsymbol{y}$
for $j = 1, \ldots, n_l$ **do**
    Get level indexes, $idx \leftarrow$ Which($\boldsymbol{c} = clevel[j]$)
    $\boldsymbol{y}^*[idx] \leftarrow$ StandardShuffle($\boldsymbol{y}[idx]$)
end for
Output: Vector of shuffled labels, $\boldsymbol{y}^*$

Note that the `StandardShuffle` function simply generates an unrestricted permutation of the elements of a vector.

7.1 Model fitting

The real data illustrations were performed using logistic regression (implemented using the `glm` function in R[21]) and random forest classification[4] (implemented using the `randomForest`[17] R package, using the default tuning parameter settings). The simulation studies and synthetic data illustrations were based on the random forest classifier. Classification performance was evaluated using the area under the receiver operating characteristic curve (AUC) metric implemented in the `pROC`[23] R package.

7.2 Synthetic data generation

In the simulation study and synthetic data illustration presented in the main text (Sections 3 and 5, respectively), we generated data from a binary classification task influenced by confounders according to the graphical model presented in Figure 3 in the main text.

In order to generate an association between C and Y (i.e., $C \leftrightarrow Y$) we jointly sampled these binary variables from a bivariate Bernoulli distribution[5], with probability density function given by,

$$p(Y, C) = p_{11}^{y\,c}\, p_{10}^{y(1-c)}\, p_{01}^{(1-y)c}\, p_{00}^{(1-y)(1-c)}, \tag{16}$$

where $p_{ij} = P(Y = i, C = j)$, and $p_{11} + p_{10} + p_{01} + p_{00} = 1$.

Note that the covariance between Y and C is given by[5],

$$Cov(Y, C) = p_{11}\, p_{00} - p_{01}\, p_{10}, \tag{17}$$

and we can tune the strength of the association between Y and C by changing these parameters. Once, we have sampled a $\{y, c\}$ pair from this distribution, we sample the features from a multivariate normal distribution,

$$N_3\big((y\,\beta + c\,\theta)\mathbf{1},\ \boldsymbol{\Sigma}\big), \tag{18}$$

where $\mathbf{1}$ represents the vector of ones, β and θ are the regression coefficients, and $\boldsymbol{\Sigma}$ represents a correlation matrix with ijth element given by $\rho^{|i-j|}$.

7.3 Parameter choices for the simulation studies

We performed four simulation experiments (Section 3 in the main text) based on data generated with: confounding and disease signal (H_1^c and H_1^d); confounding but no disease signal (H_1^c and H_0^d); no confounding or disease signal (H_0^c and H_0^d); and disease but no confounding signal (H_0^c and H_1^d). In each experiment we generated 3,000 data sets. Each data set was generated (according to the model described in Figure 3 in the main text) using a unique combination of simulation parameter values. The simulation parameters included: the sample size, n; the disease effect, β; the confounding effect, θ; the feature's correlation strength, ρ; and the probability $p_{11} = P(Y = 1, C = 1)$. Table S1 presents the ranges of the simulation parameter values employed in each experiment. In each simulation, each of these parameters were independently sampled from a uniform distribution in the respective parameter range.

par.	exp. 1 H_1^c, H_1^d	exp. 2 H_1^c, H_0^d	exp. 3 H_0^c, H_0^d	exp. 4 H_0^c, H_1^d
n	$\{300, \ldots, 500\}$	$\{300, \ldots, 500\}$	$\{300, \ldots, 500\}$	$\{300, \ldots, 500\}$
β	$[0.1,\ 1.0]$	0	0	$[0.1,\ 1.0]$
θ	$[0.5,\ 2]$	$[0.5,\ 2.0]$	0	0
ρ	$[0.2,\ 0.8]$	$[0.2,\ 0.8]$	$[0.2,\ 0.8]$	$[0.2,\ 0.8]$
p_{11}	$[0.05,\ 0.45]$	$[0.05,\ 0.45]$	$[0.05,\ 0.45]$	$[0.05,\ 0.45]$
p_{00}	p_{11}	p_{11}	$0.5 - p_{11}$	$0.5 - p_{11}$
p_{10}	$0.5 - p_{11}$	$0.5 - p_{11}$	p_{11}	p_{11}
p_{01}	$0.5 - p_{11}$	$0.5 - p_{11}$	$0.5 - p_{11}$	$0.5 - p_{11}$

Table S1: Simulation study parameters.

In order to better control the amount of correlation between the response and the confounder, we sampled the p_{11} values freely in the range $[0.05, 0.45]$, but constrained the parameters p_{10}, p_{01} and p_{00} to specific values. (Recall that $Cov(Y, C) = p_{11}p_{00} - p_{01}p_{10}$.) For instance, in Experiments 1 and 2 we constrained $p_{00} = p_{11}$ and $p_{10} = p_{01} = 0.5 - p_{11}$ so that $Cov(C, Y) = p_{11}p_{11} - (0.5 - p_{11})(0.5 - p_{11}) = p_{11} - 0.25$. Now, because the marginal distributions for Y and C correspond, respectively, to Bernoulli distributions with probability of success equal to $p_{10} + p_{11}$ and $p_{01} + p_{11}$[5], these constraints imply that $Var(Y) = (p_{10} + p_{11})(1 - p_{10} - p_{11}) = (0.5 - p_{11} + p_{11})(1 - 0.5 + p_{11} - p_{11}) = 0.25$ and $Var(C) = 0.25$, so that $Cor(C, Y) = Cov(Y, C)/\sqrt{Var(Y)\,Var(C)} = 4\,p_{11} - 1$. Therefore, it follows that sampling p_{11} from a uniform distribution in $[0.05, 0.45]$ is equivalent to imposing that $Cor(C, Y)$ is uniformly distributed in the range $[-0.8, 0.8]$ in Experiments 1 and 2.

In Experiments 3 and 4, on the other hand, we constrained $p_{10} = p_{11}$ and $p_{00} = p_{01} = 0.5 - p_{11}$ so that $Cov(C, Y) = p_{11}(0.5 - p_{11}) - p_{11}(0.5 - p_{11}) = 0$ and $Cor(C, Y) = 0$.

7.4 The approximate IPW algorithm

Following reference[18] we implemented the approximate IPW adjustment as follows:

(i) For each sample, estimate the propensity score, $\hat{p}_i = P(Y_i = 1 \mid C_i)$.
(ii) For each sample, estimate the inverse probability weight $\hat{t}_i = 1/(\hat{p}_i\, \mathbb{1}\{Y_i = 1\} + (1 - \hat{p}_i)\mathbb{1}\{Y_i = 0\})$ and round it to the nearest integer if $\hat{t}_i > 1$ (otherwise set it to 1).
(iii) Create an augmented dataset of size $\sum_i \hat{t}_i$ by over-sampling each sample $\hat{t}_i$ times.
(iv) Train a classifier using the augmented training set and evaluate its performance in the augmented test set.

In this paper, we adopted logistic regression to estimate the propensity scores (that is, we regressed the binary labels on the confounders using logistic regression and used the estimated probability that a sample belonged to the positive class class ($\hat{p}_i$), as the propensity score.

7.5 A permutation test to detect confounding

In situations where the test set is small, it might not be reasonable to approximate the null distribution of the $\bar{m}^*$ statistic by a normal distribution. In this case, however, it is still possible to test if an algorithm has learned the confounding signal using a permutation test.

To this end, we need to generate a permutation null distribution (for the $\bar{m}^*$ statistic) where the indirect association mediated by the confounder is destroyed. Accordingly, we shuffle the confounder vector in a standard fashion, before computing the $\bar{m}^*$ statistic, as described in Algorithm 3 below.

Algorithm 3 Monte Carlo permutation null distribution to detect confounding

Input: Number of standard permutations, b_s; $\boldsymbol{X}$; $\boldsymbol{y}$; $\boldsymbol{c}$; training and test set indexes, i_{train}, i_{test}
Split $\boldsymbol{X}$, $\boldsymbol{y}$ and $\boldsymbol{c}$ into training and test sets
Set the number of restricted permutations to the test set size, $b_r \leftarrow \text{Length}(i_{test})$
for $i = 1, 2, \ldots, b_s$ **do**
    $\boldsymbol{c}^{**}_{train} \leftarrow \text{StandardShuffle}(\boldsymbol{c}_{train})$
    $\boldsymbol{c}^{**}_{test} \leftarrow \text{StandardShuffle}(\boldsymbol{c}_{test})$
    for $j = 1, 2, \ldots, b_r$ **do**
        $\boldsymbol{y}^{*}_{train} \leftarrow \text{RestrictedShuffle}(\boldsymbol{y}_{train}, \boldsymbol{c}^{**}_{train})$
        $\boldsymbol{y}^{*}_{test} \leftarrow \text{RestrictedShuffle}(\boldsymbol{y}_{test}, \boldsymbol{c}^{**}_{test})$
        Train a ML algorithm on the $\boldsymbol{X}_{train}$ and $\boldsymbol{y}^{*}_{train}$ data
        Evaluate the algorithm on the $\boldsymbol{X}_{test}$ and $\boldsymbol{y}^{*}_{test}$ data
        Compute the perf. metric, m^{*}_{j}, on the shuffled data
    end for
    Compute and store $\bar{m}^{*}_{i} = b_r^{-1} \sum_{j=1}^{b_r} m^{*}_{j}$
end for
Output: $\bar{m}^{*}_{1}, \bar{m}^{*}_{2}, \ldots, \bar{m}^{*}_{b_s}$

Figure S1 compares the permutation and normal approximation null distributions for test set sizes 10 and 100. The blue curve corresponds to a density estimate of the permutation null distribution generated using Algorithm 3. The red curve corresponds to the normal approximation based on eq. (12).

The main drawback of this permutation approach is its computational demands (note that for each of the b_s standard permutations, we need to perform b_r restricted permutations).

7.6 Code availability

All the R[21] code used to generate the results in this paper is available at:
`https://github.com/echaibub/codeForKDD2019`

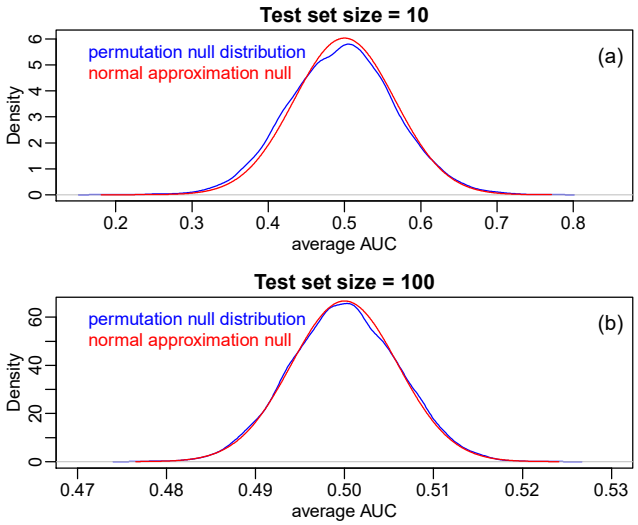

Figure S1: Permutation versus normal approximation null distributions for test set sizes 10 and 100. Results based on 10,000 permutations.

7.7 Supplementary Figures

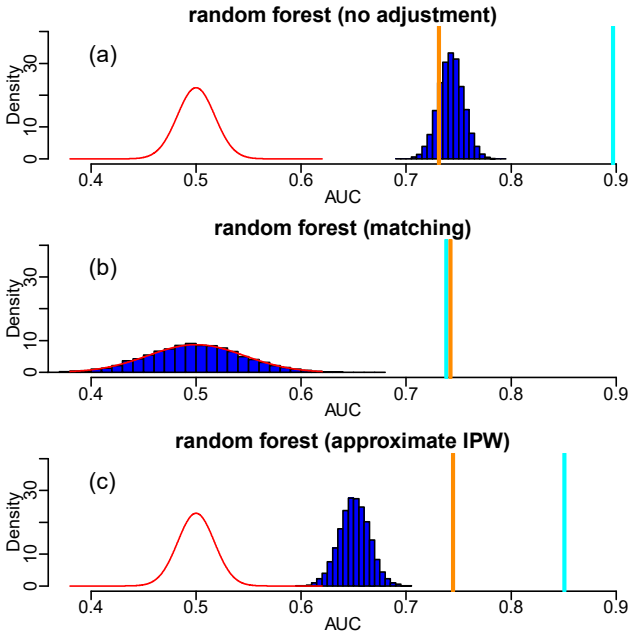

Figure S2: Comparison of confounding adjustment methods using the random forest classifier trained with a set of features generated by a deep learning model. This feature set [`https://www.synapse.org/#!Synapse:syn10949406`] corresponds to the winning submission of sub-challenge 1 of the Digital Biomarker Dream Challenge.