University of California, Berkeley
U.C. Berkeley Division of Biostatistics Working Paper Series

Year 2010 *Paper* 263

Permutation-based Pathway Testing using the Super Learner Algorithm

Paul Chaffee* Alan E. Hubbard†

Mark L. van der Laan‡

*Division of Biostatistics, UC Berkeley, chafe66@gmail.com

†Division of Biostatistics, UC Berkeley, hubbard@berkeley.edu

‡Division of Biostatistics, UC Berkeley, laan@berkeley.edu

http://biostats.bepress.com/ucbbiostat/paper263

Permutation-based Pathway Testing using the Super Learner Algorithm

Paul Chaffee, Alan E. Hubbard, and Mark L. van der Laan

Abstract

Many diseases and other important phenotypic outcomes are the result of a combination of factors. For example, expression levels of genes have been used as input to various statistical methods for predicting phenotypic outcomes. One particular popular variety is the so-called gene set enrichment analysis (GSEA). This paper discusses an augmentation to an existing strategy to estimate the significance of an associations between a disease outcome and a predetermined combination of biological factors, based on a specific data adaptive regression method (the "Super Learner," van der Laan et al., 2007). The procedure uses an aggressive search procedure, potentially resulting in final models that imply associations that would not be discovered using non data-adaptive procedures (e.g., multiple linear regression). A test statistic derived from the "fit" of the Super Learner model to the original data is compared to the permutation distribution of the same statistic, the latter being generated by permuting the outcome labels with respect to the covariate vectors. This comparison is the basis for rejection criteria for the null hypothesis of no association between a set of biological factors (e.g., gene expression levels) and binary phenotypic outcomes. We include simulations that compare the statistical power of the test derived from the Super Learner method with that of other methods for two different data generating distributions.

Introduction

Background

Many diseases and other important types of phenotypic outcomes are caused by a number of factors working in concert. A general example of this is the way a set of genes, each of which performs a similar biological function, or which are involved in the same type of biological function, are thought to be the basis for specific diseases. In these cases it may be that no single gene in a particular set of genes is statistically significant between the different outcome groups, yet the set of genes of which it is a member, taken as a whole, is significant. More specifically, certain cancers may be the result of the accumulation of mutations in various genes, or the complex interaction of these mutations rather than mutations in a single gene.

What is desired then is a statistical approach that is capable of uncovering a gene set-wide association with an outcome, even if no particular gene alone in the set is marginally associated. The approach should be capable of detecting interactions and other complicated relationships. This paper is an extension of existing techniques to achieve this goal, the main difference from earlier work being the algorithm used to derive a test statistic for association.

Existing Procedures

Consider a set of observations O consisting of outcomes Y and covariates X. That is, each individual observation O_i consists of $(X_i, Y_i), i = 1, 2, ..., n$, where X_i is a vector of covariates of dimension p. Thus X is an n x p matrix and Y is an n x 1 vector of outcomes. In this paper we focus on binary outcomes Y, but the discussion easily extends to generally discrete or continuous Y. The procedure easily accommodates situations in which p is much greater than n, which is often the case when X consists of gene expression data.

We seek to detect a statistically significant association between factors X and outcomes Y. Our null hypothesis is simply

$$Y \perp X \tag{1}$$

There are already well known methods for detecting such an association, for example, multiple testing and Gene Set Enrichment Analysis (GSEA), which we describe below.

Permutation Test

Our goal is to construct a powerful test derived from a test statistic based on the fit of a model of $E(Y|X)$, and a robust method of assessing the significance of this test statistic. Birkner et al. (2005) describe a method for testing the association of a biological pathway with observed (phenotypic) outcomes using data-adaptive regression (DAR) and a permutation-based null distribution. Here a *pathway* is defined as "a subset of biologically relevant factors grouped by some a priori set of characteristics, e.g., common function." A typical situation that meets this definition is a set (or sets) of genes thought to be associated with a common function. We have in mind a situation in which a specific set of genes or factors has been pre-identified as possibly being associated with the outcome of interest, and the researcher seeks to find statistical significance for this specific set alone.

The method of Birkner et al. was a generalization and extension of earlier methods, proposed separately by Goeman et al. (2004) and implemented in the R programming language as *globaltest()*, and by Ruczinski et al. (2003), in the development of their logic regression algorithm. The approach of Goeman, et al., referred to as "a global test for a group of genes," gives a p-value for each group of genes specified by the user. They model the way Y depends on X according to the framework of the generalized linear model of McCullagh and Nelder (1989), of which logistic regression is a special case. For a pre-specified group of genes, the association is modeled with the gene expression values as main terms:

$$E\left(Y_i|\beta_i, X_i\right) = h^{-1}\left(\alpha + \sum_{j=1}^{p} \beta_j x_{ij}\right) \tag{2}$$

Here x_{ij} is the j^{th} explanatory variable (e.g., a gene expression value) for the i^{th} observation and h is the link function, which for binary outcomes would typically be the logit function. The β_j are p unknown

coefficients to be determined. The null hypothesis of no association between Y and X corresponds to the case $\beta_1 = \beta_2 = ... = \beta_p = 0$. If the number of observations, n, is sufficiently large compared to p, standard regression techniques apply, and the test of no association reduces to a standard likelihood ratio test. However, when p approaches n, it is not valid to rely on the asymptotic distribution of the likelihood ratio statistic for hypothesis testing. The authors deal with these cases by making the assumption that the β_j are from a common distribution with mean 0 and variance τ^2. Under these assumptions, the null hypothesis becomes simply $\tau^2 = 0$, and now this single parameter (rather than p parameters) is a measure of the deviations from zero of the β_j.

The authors propose a score test for $\tau^2 = 0$. The test statistic for this null hypothesis is found by taking the derivative of the log likelihood of $Y|X$ with respect to τ^2 at $\tau^2 = 0$, and dividing it by its standard deviation. Under the null hypothesis, this statistic is asymptotically normally distributed. However, as they note, for small samples, p-values computed from this statistic may be incorrect. Their solution in these cases is to apply the permutation test and compare the test statistic to its permuted null distribution.

A similar idea—based on a very different model for $Y|X$ (logic regression)—was also described by Ruczinski et al. (2003). Logic regression is an algorithm that constructs a model whose terms are Boolean combinations of binary covariates. Note that even if the covariates are, as in our case, gene expression measures, they can easily be converted into binary variables by, e.g., assigning the value 1 to all genes whose expression measure is greater than or equal to some specified value, and 0 to all others. Again, as in our procedure, the authors find the "best scoring model" generated by applying their algorithm to a set of data consisting of gene expression measures. They then compare that score with the scores of the models built from the data with outcomes permuted, which they call "The Null Model Test." This test is exactly analogous to our procedure here.

Multiple Testing

One of the earliest approaches to address the null (1) was to test each factor separately for association with the outcome, and then adjust the type I error rate "accumulated" for the number of such tests performed.

There are various ways of adjusting the type I error rate, but we will refer to such tests generically as "multiple testing" (MT), with an associated Family-wise Error Rate (FWER). For example, suppose one performs K hypothesis tests on a set of data, one test of association between each of the factors and the outcome (this implies K factors), and of the K tests the null hypothesis is rejected R times. Of those R times, let us call the random variable that is the number of incorrect rejections (i.e., the number of times the null is rejected when it is in fact true) R_F. We define FWER as

$$FWER = P(R_F \geq 1| \text{ Global Null})$$

Thus FWER is the probability of one or more false rejections of the K tests, i.e., the probability of at least one occurrence of a type I error. The usual type I rate of 0.05 translates to $FWER \leq 0.05$ for K tests. For our comparisons we used the basic Bonferroni correction for multiple testing (amongst the most conservative of the FWER correction methods), $\alpha^* = 1 - (1 - \alpha)^{1/K} \doteq \alpha/K$ for small α, where α is the FWER achieved for K individual tests each of which rejects at level α^*.

In our simulations we performed a separate t-test for each of the p covariates in the simulated data. Thus for p covariates, p t-tests were performed, and if the t statistic computed for one or more of the tests corresponded to a p-value $\leq 0.05/p$, the null hypothesis of no association between the covariates and outcomes was rejected.

We also report on the performance of MT based on a permutation test, details of which are presented in the Simulations section below.

GSEA

The original GSEA methodology was not developed to determine the significance of a particular pathway considered by itself, which is our main interest here. Rather, the method was meant to determine the significance of the association of phenotype with a particular pathway or gene set with respect to other gene sets in a genome-wide analysis (Subramanian et al., 2005). However, later versions of GSEA (Jiang and Gentleman, 2007) expanded upon the initial methods and provide

a means for testing the null hypothesis of our analysis, and we include one such method for comparison here. Jiang and Gentleman (2007) consider a variety of tests (t-test, log rank test, etc.) and associated test statistics, as well as various parameters of their empirical distributions (mean, median, etc.) for a particular group of genes. The analysis we ran chose the average p-value of the genes in the group and compared this with the associated permutation distribution. The p-value for each gene in the group was based on a t-test, as in our MT procedure. The overall p-value associated with this version of the GSEA analysis will thus always be greater than or equal to our version of permutation-based MT testing as described above.

Thus we don't expect this method to have high power for the kind of data and hypothesis we're considering here–i.e., the significance of the association of one gene set without regard to the significance of other gene sets in the genome. We nevertheless include it here because of its general relevance in detecting associations between pre-specified sets of genes and phenotypic outcomes.

Method

Our method is to use a much more general machine learning approach to search a large model space to obtain a data-adaptive model for $E(Y|X)$. The larger the space searched, the greater the likelihood of obtaining the correct model, though searching very large model spaces comes at the cost of computing time and estimation variability in small sample sizes. Our procedure relies on the Super Learner algorithm (van der Laan et al., 2007) which itself combines a variety of data-adaptive regression or classification algorithms into one model, effectively drawing on the strength of this library of candidate learners. The Super Learner synthesizes these various candidate learners by weighting the models built by each in a final larger model. The procedure is described in more detail below.

In our procedure, a test statistic, W^*, is generated based on the fit of the model constructed by the Super Learner algorithm to the data. The original outcome vector Y is then permuted with respect to X, and for each permutation, the Super Learner does its best to model the resulting empirical density, from which the test statistic corresponding to that particular permutation is

computed, just as with the original data. The significance of the association of Y with X using this method is determined from comparing the value of W^* with the distribution of permuted test statistics, the latter being the approximate null distribution for W; it is only an approximation because the number of permutations typically done is not exhaustive. In principle, one can generate as many permutations as the size of the sample allows, which is astronomical for sample sizes greater than about 35. Naturally, computing time puts a practical limit on this as well.

As mentioned above, this method is an extension of the method originally set forth in Ruczinski et al. (2003), and expanded upon and generalized by Birkner et al. (2005). In both latter cases a single data-adaptive regression algorithm was fed the data and the test statistic W^* was generated based on the outcome of that algorithm alone. Our method is different in its application of the Super Learner as the data adaptive algorithm. The permutation aspect of the procedure is the same as what has been proposed in these earlier works.

For this discussion, consider data concerning a particular pathway for a set of observations of size n, and p is the number of variables in the pathway. The i^{th} row of X is the set of values these variables take for the i^{th} observation. Suppose also that we have a binary vector of outcomes Y of length n, the i^{th} element of which is the outcome associated with the i^{th} observation.

If we believed the mean of $Y|X$ were accurately described by (2) with h being the logit function, then the pathway test would consist in conducting the procedure à la Goeman et al., as previously described.

The application of the above pathway test is severely limited. Since the model does not even include interaction terms, if the true data generating distribution has weak linear term dependence, but strong associations with multiplicative interaction terms, this test has a poor chance of picking up the association. If one chooses to include interaction terms, the model can become large very rapidly, especially with 10 or more covariates. In any case, the model will certainly be wrong and therefore a data-adaptive model-building method will have a better chance of fitting the data.

The idea in data-adaptive pathway testing is, first, to apply a data-adaptive regression (DAR) algorithm such as Random Forests (Breiman, 2001), or Logic Regression (Ruczinski et al., 2003) to the data in order to estimate a model, and to generate a test statistic, W^*, which is large when the loss (as

measured by, e.g., log likelihood) for the model is low. Candidate test statistics are, for example, RSS (for linear models), the likelihood ratio statistic (logistic models), and pseudo R-squared for binary prediction algorithms, such as Random Forests. Next, an empirical null distribution for this test statistic is generated by randomly permuting Y with respect to X (say Z times). Let $Y^{(z)}$ be the z^{th} permuted outcome vector where $z \in \boldsymbol{Z} = \{1, 2, ..., Z\}$. For each $z \in \boldsymbol{Z}$, the DAR algorithm is run on $(Y^{(z)}, X)$ and the corresponding test statistic W_z is calculated. The pathway test p-value, p_{pw} is then estimated from the proportion of test statistics W_z generated from the permuted cases that are greater than W^*:

$$p_{pw} = \frac{1}{Z} \sum_{z=1}^{Z} I(W_z > W^*) \tag{3}$$

Here I stands for the indicator variable. Simulations done by Birkner et al. (2005) using POLYCLASS (Kooperberg et al., 1997) as the DAR algorithm indicate that tests based on these algorithms and the permutation test can have much greater power than main-terms logistic regression, Bonferroni-adjusted multiple testing and several other algorithms when the true data generating distribution included interaction or non-linear terms. Conversely, when the true model is simple and thus the simpler approaches contain the true model, these procedures based on data-adaptive algorithms still maintain relatively good power.

The null hypothesis for the pathway test (1), as mentioned earlier, is very general. We expect the value of W_z to be low on average for the permuted cases, since they are cases in which, by definition, $Y^{(z)}$ is independent of X. Therefore, models generated for those cases should not be able to consistently predict $Y_i^{(z)}$ from X_i, and such models will tend to generate a small W_z. On the other hand, if the DAR algorithm is able to discover a model that has good success in predicting Y from X in the original, unpermuted data, then W^* is likely to be high relative to the W_z, and p_{pw} is likely to be low. Thus the selection of the particular form of the DAR is the cornerstone of the procedure, and it's worth the effort to obtain the best model-building algorithm one can in order to get the most out of the test. Since the true model is never known, flexibility in terms of modeling a variety of data-generating distributions is crucial. This translates to searching the largest possible set of data-generating distributions.

The Super Learner (van der Laan et al., 2007) is a data-adaptive meta learner that employs multiple sub-learners (the authors refer to them as "candidate learners") and combines them in a final model. This is accomplished as follows. Suppose one had in hand m favored DAR algorithms, say $DAR_1, DAR_2, ..., DAR_m$. Figure 1 is a schematic representation of the process, described below.

Step 0. Run each DAR separately on the entire data set to obtain a model from each.

Step 1. Split the data into V blocks.

Step 2. Run each DAR separately on the training set of each block.

Step 3. For each DAR, use the model it builds on the training set of each block to predict the outcomes for the validation set of that block. Repeating this V times then gives a vector of predicted outcomes, $\widehat{Y}$, for each DAR. This yields an n x m matrix of predicted outcomes, which becomes the new design matrix.

Step 4. Regress these predicted values on the true outcomes to obtain a final "output model" which in this case consists of m covariates, each corresponding to its associated DAR. In essence, we now have a model which consists of weighted values assigned to each DAR, which are retained for the final step. (If one is only interested in uncovering association between X and Y, as in our case, one can simply compute the deviance of this model, which is then equivalent to W^*.)

Step 5. Combine step 0. with the covariates obtained in step 4. The final overall model thus contains the sub models built by each DAR from the entire dataset, each multiplied by its corresponding coefficient determined in step 4.

There is no theoretical upper limit to the number of algorithms the Super Learner can incorporate, though obviously computing time considerations place a practical upper limit on that number. Further, the larger the library of learners, the larger n needs to be for the procedure to be effective. In our simulations, only four algorithms were used (see the simulation section), though at present versions of the Super Learner that contain close to 100 candidate learners have been coded (Polley, 2009). The authors of the

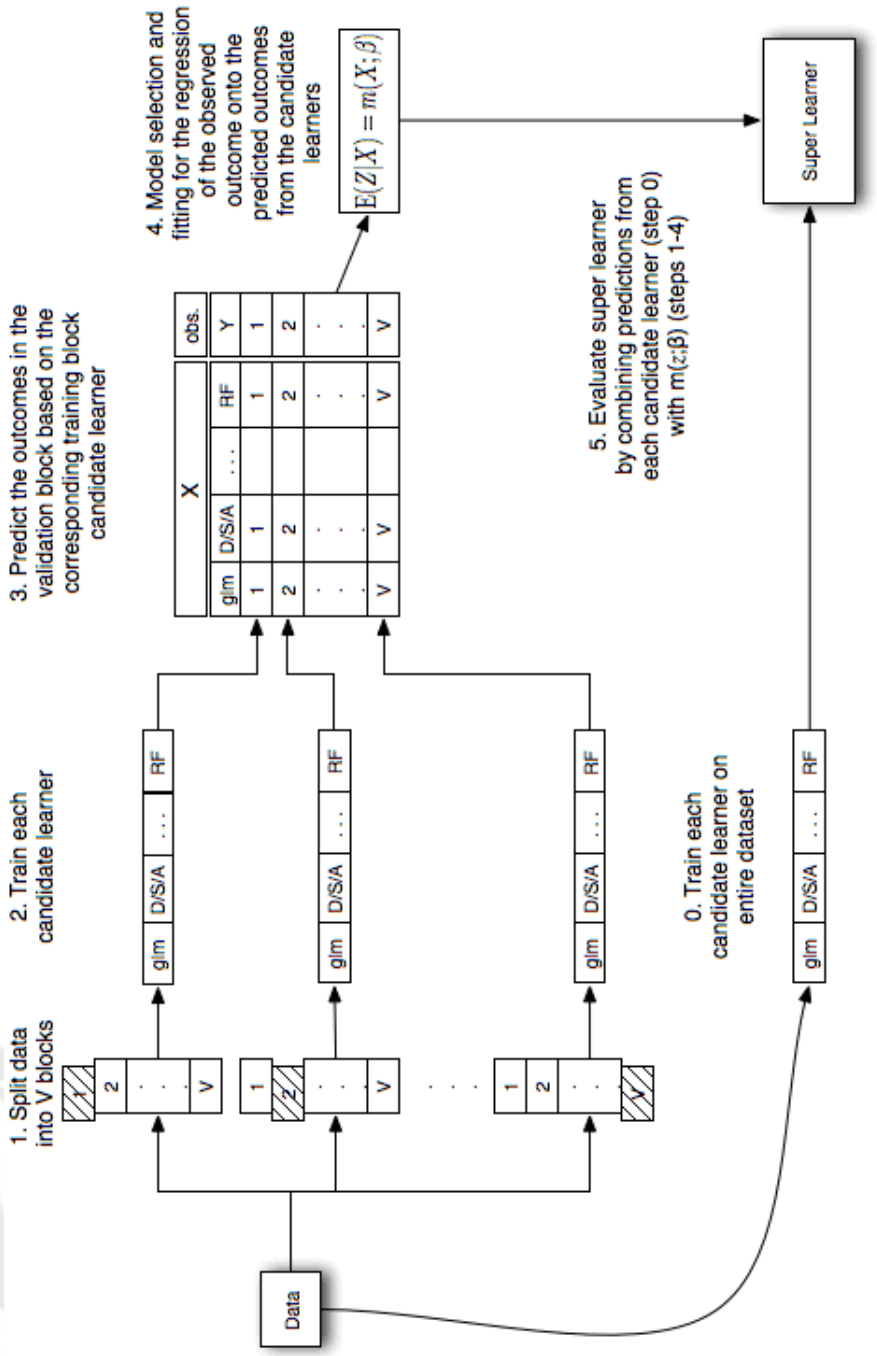

Figure 1: Schematic representation of the Super Learner algorithm (courtesy Polley, 2009). Specific algorithms are denoted in the figure. In the text we refer to the library of learners more generally as "$DAR_1, DAR_2, ..., DAR_m$."

Super Learner have emphasized the prudence of including learners that are disparate in their model-building procedures. For example including only learners that build polynomial base functions would not be as flexible as a library consisting of one such DAR, another that uses CART, another for logic regression, etc.

For our purposes, the form of the Super Learner output model is inconsequential. What matters is the magnitude of the test statistic computed from it using the original data compared to the distribution of the permuted-data test statistics. For this reason, we use only steps 1 - 4 of the Super Learner schema described above. Steps 0 and 5 are required for cases in which prediction of outcomes for a new X_i is desired, which is not relevant for our study. The power of the test is directly related to the expected value of W^* when an association is present.

In our case the outcomes are binary, so it was natural to use the logit link function for the Super Learner output model, though other link functions are also suitable, since prediction was not our aim. Indeed we also ran simulations with an ordinary linear model as the link function for the final Super Learner model; the results were nearly identical to those when the link function was the logit function. As with linear models, logistic regression lends itself well to a simple likelihood ratio test, in which the test statistic is the null deviance minus the actual deviance of the output model. This is equivalent to the likelihood ratio statistic.

van der Laan et al. (2007) prove that the Super Learner performs as well as the 'oracle' candidate learner in terms of expected risk difference between the truth and the selector, "up to a typically second order term." In our case, this translates, asymptotically, to producing a test statistic that approaches what would be produced by the oracle learner. The oracle learner is defined as "the estimator, among the [m] learners considered, which minimizes risk under the true data-generating distribution." Therefore, the Super Learner is arguably the most adaptive and flexible algorithm one can construct, since any algorithm that putatively performs better for a given data-generating distribution can simply be subsumed by the Super Learner, thus increasing the likelihood that the oracle estimator is incorporated within it.

Simulations

Data was simulated in order to compare the Super Learner pathway test to 1) multiple testing based on a t-statistic, 2) results from the GSEA package in R (the specific method for which is described above) 3) a pathway test using a logistic model that included only main terms, and 4) a pathway test using Random Forests as the single DAR algorithm. The comparison was in terms of the number of times the null hypothesis of no association was rejected when in fact an association was present. Simulations were also performed with no $X - Y$ association to establish that the type I error rate was indeed controlled at the desired level (0.05 in our tests).

Since the motivation of the Super Learner Pathway Test is to be able to pick up subtle and/or complex associations between the genetic factors and the outcome, we sought a data-generating distribution that was expected to be difficult for the competing methods to detect. One such distribution contained interaction terms and relatively weak linear main terms. For these simulations the rows of X (i.e., the covariates) were drawn from a multivariate normal with mean 0, and covariance Σ with

$$\Sigma = \begin{pmatrix} a_{11} & \dots & a_{1p} \\ \vdots & \ddots & \vdots \\ a_{p1} & \dots & a_{pp} \end{pmatrix} \text{ where } a_{ij} = 36 \text{ for } i = j \text{ and } a_{ij} = 1.8 \text{ for } i \neq j$$

This corresponds to a constant off-diagonal correlation of 0.05. Outcome values for the ith "observation" were generated in two steps. First a deterministic value for the ith observation, y_i^*, was generated according to

$$y_i^* = (1 + exp\left(-\left[f(X_i)\right]\right))^{-1} \tag{4}$$

where $f(X_i)$ is a function of the covariates. For one set of simulations, we made this function more complex and, we expected, less apt to be discovered by most of the comparison methods. This function was

$$f(X_i) = \frac{1}{3x_{i1}} + (0.25)\frac{x_{i3}^2}{x_{i2}} + (4.2)\frac{x_{i1}}{x_{i3}} \tag{5}$$

For the other set of simulations, we made $f(X_i)$ such that we expected all the methods to be able to detect the association, namely,

$$f(X_i) = x_{i1} + (2.3)x_{i5} + (3.22)x_{i6} + (0.5)x_{i9} \tag{6}$$

In each case x_{ij} is the value of the j^{th} covariate for the i^{th} person. Next, random error was introduced into the values generated above by creating binary random variables y_i according to

$$y_i = \begin{cases} 1 & \text{with prob } y_i^* \\ 0 & \text{with prob } 1 - y_i^*, \end{cases} \tag{7}$$

i.e., $y_i \sim Ber(y_i^*)$. Run time considerations for our simulations forced us to limit n and p to 10 covariates and 150 observations, respectively, though one would like to observe cases in which $p \geq 100$ since pathways can consist of hundreds or thousands of genes.

We ran one set of 500 simulated data sets for each of the two different data generating distributions above. Rejection of the null for each simulated data set was determined for each of the competing methods as follows.

i) Multiple Testing

1. For each simulated data set a t-test was performed separately for each "gene." The lowest p-value amongst the 10 was recorded, and if

$$10 * min\left\{p_k : k\right\} \leq \alpha = 0.05, k = 1, 2, ..., 10$$

then MT rejected the null for that data set, where p_k is the p-value associated with the k^{th} gene.

2. We also computed an MT statistic based on a permutation test with $Z = 1500$ permutations. In this case, we compared the largest absolute value of the t-statistic of the p covariates for the original data with the permutation distribution of the corresponding statistic. In other words, let

$$t^* = max\left\{|t_1^*|, |t_2^*|, ..., |t_p^*|\right\}$$

where t_k^* is the t-statistic computed for the k^{th} covariate of the unpermuted data. If $t^{(z)}$ denotes the analogous t-statistic for the z^{th} permutation, then the MT permutation test statistic yields a p-value

$$p_{MT} = \frac{1}{Z}\sum_{z=1}^{Z} I(t^{(z)} > t^*)$$

ii) Permutation Test using Main Terms Logistic Regression
This test involved performing a pathway test using a main terms logistic regression model as the source of the relevant test statistic. Note that this test becomes untenable as the number of covariates increases. If there are 100 genes in a particular pathway of interest, then this test requires the specification of a one hundred-term logistic model. And, of course, as p approaches n, the variance of the coefficients increases dramatically. For each simulated data set, a logistic regression model was fit that included each of the main gene terms, but no interaction terms. That is, coefficients β_j were determined from specifying the following model in the *glm* function of R:

$$E[Y|X] = \left[1 + exp\left(-\left[\alpha + \Sigma_{j=1}^{10}\beta_j X_j\right]\right)\right]^{-1}$$

For each simulation, the likelihood ratio statistic, $LR = 2[logL_1 - logL_0]$ was calculated, where L_1 is the likelihood of the model selected and L_0 is that for the null (intercept only) model. Next, $Z = 1500$ permutations of the Y vector with respect to the covariate matrix were performed. The above logistic model was fit for each $Y^{(z)}$ in place of Y and the corresponding LR_z was calculated. The proportion of these statistics greater than LR^* (the LR for the unpermuted, original data) was computed (as in eq. 3) , which is the p-value for the logistic regression method for that data set. A p-value $\leq \alpha = 0.05$ was again grounds for rejection.

iii) Random Forests
Random Forests (Breiman, 2001) is a classification and regression algorithm that "grows" multiple classification trees instead of just one tree, as in standard classification and regression. It is implemented in the R programming language in the *randomForest* package. Each tree in the forest is grown from a bootstrap sample of the original data. A random subset of the variables (covariates) of size $m << p$ is also selected at each tree node (the default number of variables is $\sqrt{p}$), and the best split of these m variables is used to split the node. Outcome prediction based on a new covariate vector X^* is obtained by putting the X^* down each tree in the forest that was grown from the data. Assume there are K trees in the forest grown from a particular data set. Each tree gives a classification prediction (i.e., a $\hat{y}_k^*$) based on the new X^*, and the predicted class for a given tree counts as that tree's "vote" for the outcome class. The final prediction of the outcome for X^* is the class that received the most votes from the forest.

Upon obtaining a regression model using the Random Forests algorithm (i.e., a forest), we used it to generate the vector of fitted values, $\hat{Y}$, as explained above. There are various test statistics that are appropriate for determining the model fit for binary outcomes. We chose an "Adjusted count R-squared" statistic, which is a type of so-called pseudo R-squared statistic (Hardin and Hilbe, 2007). This particular version of pseudo R-squared is sensitive to the number of correct observations, but penalizes blanket guesses that, e.g., simply predict each outcome to be the most frequent observation. Concern regarding the latter arises when one considers the following type of algorithm. Suppose the proposed rule is simply to determine the most common 1/0 outcome, and then to predict every outcome to be that value. This algorithm always gets at least half of the outcomes correct, but it clearly cannot yield anything informative about the relationship between X and Y since the prediction method has no dependence whatsoever on the explanatory factors, X. The adjusted count statistic adjusts for this scenario, being defined as:

$$R^2 \equiv \frac{n_c - n_f}{n - n_f}$$

Here, n_c is the number of correct predictions, $n_c = \sum_{i=1}^{n} I(y_i = \hat{y}_i)$, where $\hat{y}_i$ is the predicted value for observation i; and n_f is the number of outcomes of the most frequent type, i.e., $n_f = max\{ \sum_{i=1}^{n} I(y_i = 1), \sum_{i=1}^{n} I(y_i = 0)\}$. If $\sum_{i=1}^{n} I(y_i = 1) = \sum_{i=1}^{n} I(y_i = 0)$, then $n_f = n/2$. Note that, assuming $1 < n$ and $n/2 \leq n_f < n$, the range of the R^2 statistic is

$$1 - n < R^2 < 1$$

Though the R^2 statistic can theoretically attain values as low as $1 - n$, applying it to Random Forests in our permutation simulations always produced a range of R^2 values between -1 and 1, and typically well away from even those extremes. This is because the range of values of n_f was always well below its possible extreme value, n. Indeed, as long as $n_f \approx n/2$, the range will be exactly as observed. Moreover, though the R^2 statistic is the ratio of two random variables, our simulations show its null distribution to be very close to Normal.

The p-value associated with the Random Forests pathway test was computed analogously as for the other permutation-based methods but with R^2 as the test statistic.

(iv) Super Learner

Running the Super Learner on the data entailed, first, that a "library" of learners is chosen, and for each learner, a prediction vector is generated according to step 2 of the Super Learner schema explained in the previous section. In our tests we chose the number of folds, V = 2 strictly on the basis of run-time considerations. In these simulations the library consisted of only four candidate learners, 1) L2 penalized logistic regression with stepwise variable selection, described in Park and Hastie (2007) (available in R as package *stepPlr*), 2) Logic Regression, described in Ruczinski et al. (2003) (R package *LogicReg*) 3) Main terms logistic regression, as described in ii) above, and 4) R Part, which is a classification and regression algorithm available as package *rpart* in R (see, e.g., Brieman et al., 1984). For the logic regression algorithm, the multivariate normal covariate values generated according to (4) were converted to binary values. For each covariate, the average value across all observations for that covariate served as the dividing point for re-assigning each observed value a 0 or 1 (0 if less than the mean, 1 otherwise).

After each learner in the library selects a model, the final step is to regress the outcomes Y on the predictions of these learners (recall that steps 0 and 5 of the Super Learner schema are omitted in our procedure). This final regression yields a deviance that we used to generate the Super Learner test statistic, W, which is equivalent to the likelihood ratio statistic. Note that this statistic does not follow a Chi Square distribution because the DAR algorithms in the library are likely to specify different models for different sets of data. The deviance statistic's following a Chi Square distribution in standard model fitting is based on the relevant model's being static, and not dependent on the data, which is contrary to the very nature of DARs. Preliminary simulations suggest however that the statistic may follow a Gamma distribution whose parameters are functions of the number of algorithms used in the Super Learner library (which corresponds to the number of terms in the Super Learner regression model). If it can be established that the null distribution of W is indeed gamma with known parameters, performing permutations to determine the null will of course be unnecessary. However, the empirical observation that the permutation distribution of deviance is gamma-distributed needs to be theoretically verified.

As in all the permutation-based pathway tests mentioned above, a p-value is obtained from comparing the test statistic of the unpermuted data, W^*, with the distribution of permuted test statistic values, W_z, as in (3).

Results

When an association exists, the rejection rate of the tests is their respective power. The first set of simulations was generated according to (4), (5) and (7), and thus represents a case in which we expect most of the comparison methods to have relatively low power. Table 1 lists the number of rejections out of 500 independent data simulations for each method, and the corresponding power.

Method	**Rejections**	**Power**
MT, Bonferroni	158	0.316
MT, permutation test	162	0.324
Logistic Regression, main terms	138	0.276
Random Forests	421	0.842
GSEA	75	0.15
Super Learner	457	0.914

Table 1: Number of rejections and corresponding power of the various methods (500 simulations). Data generation based on functions (4), (5) and (7)

As we expected, the Super Learner pathway test out-performs logistic regression by a wide margin, since the latter model included only main effect terms. See figures 2 and 3 for typical distributions. MT also compares relatively poorly. Random Forests, on the other hand, does a much better job than the latter two methods, though still falls short of the Super Learner. Random Forests could, of course, be incorporated into the library of the Super Learner algorithm, a fact which serves to underscore again the main strength of this method.

It is noteworthy that in 301 of the 500 simulations, W^* was far greater than any W_z. In these cases the p-values computed were exactly 0, which is of course an artifact of our method of determining p-values based on a discrete distribution of 1500 permutations. It would be more accurate to say that the true p-values in these cases were less than $1500^{-1} (\approx 6.7 \cdot 10^{-4})$. Nevertheless, judging from the value of W^* with respect to the distribution W_z in these cases, the p-value would likely be very much less than this for much larger values of Z, probably on the order of $1/Z$. (See figure 2.)

We also simulated data according to (6) and (7), i.e., in which the data

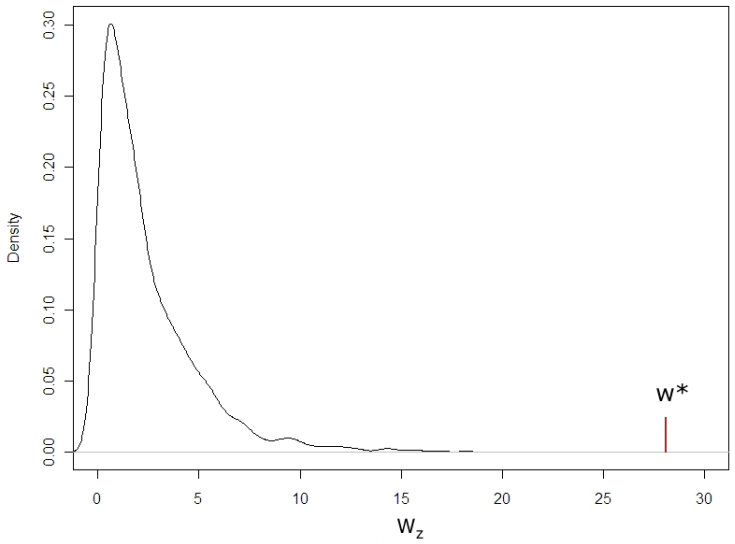

Figure 2: Typical permutation distribution of the Super Learner test statistic and value of W^* when an association is present. Data simulated according to (5) and (7).

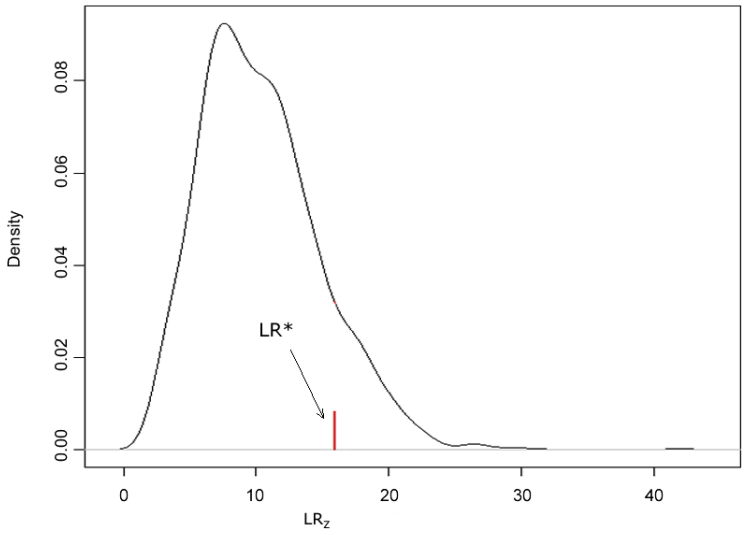

Figure 3: Typical permutation distribution of the deviance for the logistic regression model and value of LR^* when an association is present. Data simulated according to (5) and (7).

Method	Rejections	Power
MT, Bonferroni	500	1.0
MT, permutation test	500	1.0
Logistic Regression, main terms	499	0.998
Random Forests	500	1.0
GSEA	500	1.0
Super Learner	496	0.992

Table 2: Number of rejections and corresponding power of the various methods (500 simulations). Data generation based on functions (4) and (5)

generating distribution involved only linear terms. Table 2 gives the results of these simulations. Under these circumstances we expect the existing tests to perform very well, and perhaps better than the Super Learner method, since the latter is predicted to perform only asymptotically as well as the oracle model, which is in fact included amongst the four learners.

Clearly every method performed well (some flawlessly) for a strong main terms effect data generating function, with the Super Learner lagging just slightly. One issue is that the Super Learner must determine the form of the best model from the data, rather than benefiting from having the correct model pre-specified, though this is true of Random Forests as well. Even though the correct model is included in the Super Learner's library, we still expect it to be slightly out-performed by a pathway test that uses the correct model alone. This is because the Super Learner is expected to perform only asymptotically as well as the oracle estimator. But since in all cases of interest involving real data one never knows the true model, this simulation shows that even when a linear terms logistic model has the unrealistic benefit of being handed data generated according to its specific form, it performs only negligibly better than the Super Learner. This is very strong evidence of the Super Learner's adaptive capacity, and that the price this adaptability pays in its ability to correctly reject the null is minimal. The important point is that the Super Learner performs well under a wide range of possible data generating distributions, but the other methods have limited effectiveness except in simple cases, like main term only effects.

Discussion

Of particular interest in the analysis is the degradation in terms of power of the various methods when the associations involve other than linear main effects.

Note that multiple testing as utilized here is incapable of distinguishing cases in which a single gene in the pre-specified gene set is significant from cases in which many or all genes in a particular gene set are significant. Worse, MT will tend to miss altogether cases in which multiple genes are associated with the outcome but all per-gene effects are small. This is a serious drawback in pathway testing, since we seek a method that is capable of detecting precisely such cases.

The version of GSEA we used suffers from a similar problem, though there are other GSEA tests that do not. Indeed, GSEA was developed, in part, to detect the cases that concern us here, namely, cases in which though no particular gene in a pathway has a particularly strong association with the outcome variable, the pathway taken as a whole is significant. However, all of the latter GSEA methods that we are aware of focus on situations in which more than one gene set is involved in the analysis, and one seeks to find which of these gene sets is significant, or most significant. This is not the scenario of interest in the present study.

Parametric Estimation of the Distribution of W

In the course of examining the null-distribution of W we noticed that the density appeared approximately Gamma (see fig. 2). More convincing evidence for this conjecture was obtained by applying the method of moments to estimate the parameters of a Gamma, and then computing the Kilmogorov-Smirnov goodness-of-fit statistic for the comparison of the resulting fitted Gamma with the null W. For example, the p-value associated with the K-S statistic from the data for figure 2 was 0.43. It's important to note that the K-S goodness of fit critical values are not strictly valid when the parameters of the fitted distribution are derived from the data to which the test is being applied, as we do here. Nevertheless, we think this statistic is a reasonable measure of the goodness of fit, especially when used in conjunction with a measure of fit we've devised (see below), as long as the computed p-value is not used as a strict cut-off criterion.

If it is known that the null distribution of W is truly Gamma, then the number of permutations required to get an acceptably accurate p-value for the original W-statistic (i.e., W^*) reduces to the number required to get a correspondingly accurate estimate of the parameters for a Gamma distribution fitted to the permutation distribution of W. (We refer to such a Gamma distribution as $\widehat{F}_G$.) The hope is that this number of permutations is far less than that needed using the null distribution of W itself, and computing time would thereby be greatly reduced.

We know of no theory that predicts the distribution of W should be Gamma, but we have found that the parameters of $\widehat{F}_G$ are functions of various aspects of the Super Learner, and of the marginal distributions of Y and X. As table 3 shows, the parameters of $\widehat{F}_G$, and the goodness of fit are both sensitive to 1) the number of cross-validation folds employed in the Super Learner 2) the number of learners in the Super Learner library, and 3) the marginal distributions of Y and X. (The parameters are certain to be functions also of the specific learners in the Super Learner library as well, since different algorithms have different levels of success in predicting Y from X.)

While this shows that the prospect of finding a single Gamma distribution that approximates well F_W for a fixed set of learners in the Super Learner is probably hopeless, the benefit in computing time is still good cause to pursue model fitting.

C-Statistic

In simulating the various distributions of Y and X, we discovered that the K-S statistic was often sensitive to differences between F_W and the corresponding $\widehat{F}_G$ that would not be significant for the question of interest here. The K-S statistic, defined as

$$D_n = \sup_x [F_n(x) - F(x)]$$

for some distribution F is somewhat sensitive to differences close to the center of the distributions being compared, and differences there may not be of consequence if one is interested in, for example, the 95th quantile, as we are here. We therefore devised a GoF statistic that is less sensitive to the differences that K-S is, though it is more sensitive to individual large discrepancies (for example, in the extreme right tail). We call this statistic the $C - statistic$ and compute it as

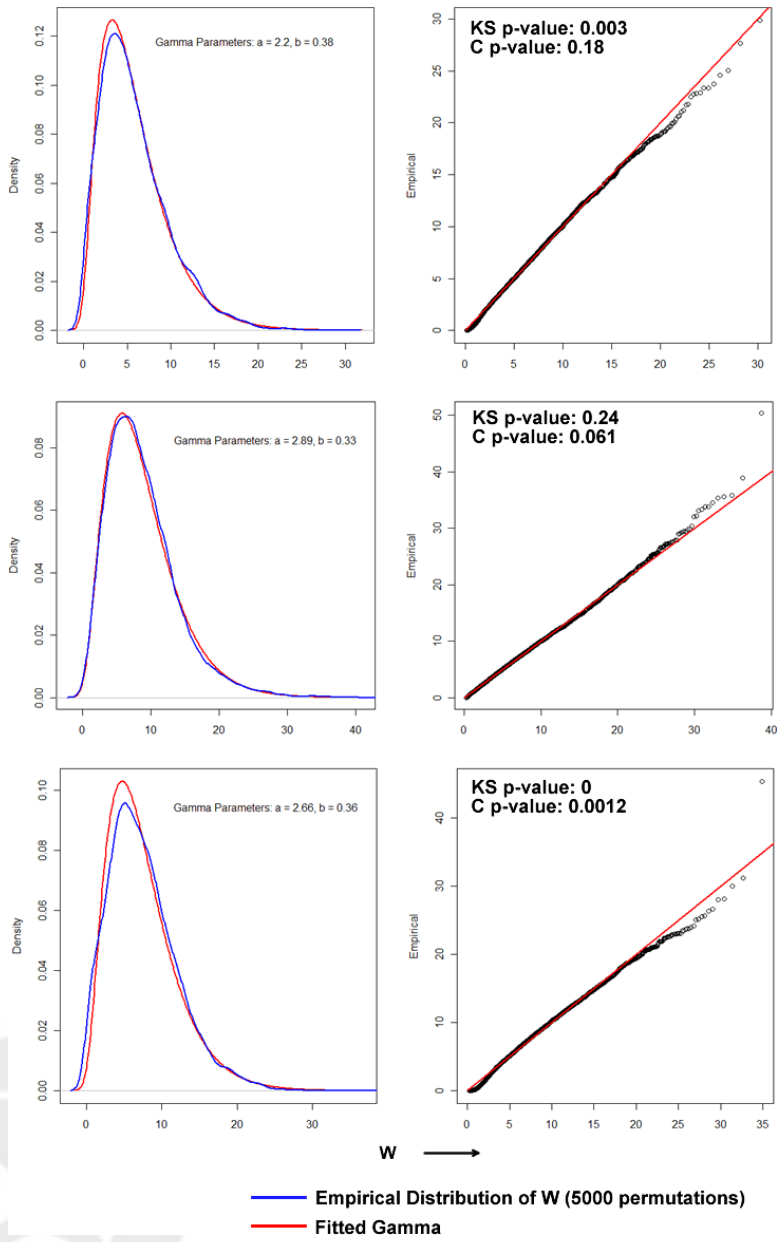

Figure 4: Comparison of densities and Q-Q plots for $\widehat{F}_G$ using method of moments and that of the permutation distribution of W (5,000 permutations) for various numbers of learners in the Super Learner, various numbers of cross-validation folds, and different marginal distributions of X. Three scenarios are represented: *Top* Low K-S p-value but high C-statistic p-value (2 learners, 6 X-validation folds, X$\sim$ Normal with mean 0 and no correlation); *Middle:* High K-S p-value, low C p-value (5 learners, 4 X-validation folds, X$\sim$ Normal with mean 0 and no correlation); *Bottom:* Low K-S and C p-value, (2 learners, 8 X-validation folds, X$\sim$ Normal with mean 0 and high correlation).

$X \sim N(0, \Sigma_1)$						
Folds	**Shape Parameter**		**GoF p-val (K-S)**		**GoF p-val (C)**	
	3 Learners	5 Learners	3 Learners	5 Learners	3 Learners	5 Learners
4	a = 2.1	a = 2.98	0.70	0.33	0.40	0.25
8	a = 3.1	a = 3.4	0.0037	0.11	0.025	.023
$X \sim N(0, \Sigma_2)$						
	3 Learners	5 Learners	3 Learners	5 Learners	3 Learners	5 Learners
4	a = 1.89	a = 2.89	0.13	0.24	0.039	0.069
8	a = 2.23	a = 3.20	0.094	0.42	0.096	0.0178
$X \sim$ Bernoulli						
	4 Learners	5 Learners	4 Learners	5 Learners	4 Learners	5 Learners
10	a = 1.71	a = 2.12	0.017	0.023	0.078	0.052

Table 3: Gamma shape parameter and goodness of fit p-values based on two different statistics. Sample size = 500, number of permutations = 5000, binary outcomes and X generated according to 1) a mean-0 Normal distribution with high correlation between covariates, 2) a mean-0 Normal distribution with low correlation between covariates and 3) a binary version of the covariates where the value for observation j's i^{th} covariate x_{ij}, was computed as $I(x_{ij} \geq x_{i\cdot}/n)$. Low C-statistic p-values generally signify large discrepancies in the extreme order statistics.

$$C^* = \frac{1}{Z}\sum_{i=1}^{Z}\left[W_{(i)} - \widehat{F}_G^{-1}\left(\frac{i}{Z+1}\right)\right]^2 \tag{8}$$

where $W_{(i)}$ is the i^{th} order statistic $(i = 1, 2, ..., Z)$ of the empirical null W distribution and $\widehat{F}_G^{-1}\left(\frac{i}{Z+1}\right)$ is the $i/(Z+1)^{th}$ quantile from a Gamma distribution with parameters fitted from the data. The asterisk here indicates the computation is for the original, unpermuted data. Parameters were fitted both using method of moments (MOM) and maximum likelihood estimation (MLE), and MOM was generally superior based on both goodness of fit measures. C^* is thus the average squared difference between these two statistics. One can also think of it as the average squared vertical deviation from the line of slope 1 in a q-q plot of the W distribution vs that of $\widehat{F}_G$. Like a residual, C^* is not robust to outliers, but is less sensitive to accumulations of small deviations from the slope-1 q-q line than is the K-S statistic. The p-value associated with C^* is obtained by comparing it to a monte carlo distribu-

tion of the same statistic but with $W_{(i)}$ replaced by the corresponding order statistic from a set of Z randomly generated Gamma random variables using the same fitted gamma parameters as $\widehat{F}_G$. That is, the statistic generated above, C^*, is compared to a monte carlo-generated distribution of

$$C = \frac{1}{Z}\sum_{i=1}^{Z}\left[Q_{(i)} - \widehat{F}_G^{-1}\left(\frac{i}{Z+1}\right)\right]^2 \tag{9}$$

where $Q_{(i)}$ is the i^{th} order statistic of a set of n i.i.d. $G(a,b)$ random variables, where a and b are again the Gamma parameters fitted from the data, i.e., the parameters of $\widehat{F}_G$. One then gets values $C_1, C_2, ...C_K$ and the p-value associated with C^* is estimated as

$$\hat{p}_{C^*} = \frac{1}{K}\sum_{k=1}^{K} I\left(C_k > C^*\right)$$

.

We ran $K = 5000$ sets of draws of size Z from $\widehat{F}_G$ for each estimate of p_{C^*} for each simulation shown in table 3. (Note that even for a fixed value of C^*, $\hat{p}_{C^*}$ is a random variable with SE proportional to $1/\sqrt{K}$.) As mentioned above, the C-statistic is sensitive to outliers, and thus if there are a few of these in the extremes of the tails, but the fit of the Gamma to the null of W is otherwise good, the few outliers can be removed to get a better assessment of the goodness of fit.

Rather than tinkering with outliers, a better method of assessing the goodness of fit of the Gamma near a specific point in the distribution (for example near $F^{-1}(0.95)$) using the C-statistic is to systematically down-weight the differences in order statistics that are far from the point of interest. For example, if one is using $\alpha' = 0.05$ as a GoF cutoff p-value, then one would down-weight differences in the empirical and fitted Gamma distributions that are far from $F^{-1}(0.95)$. The resulting p-value will then be relatively insensitive to all differences sufficiently far from the region of interest, and so this modified statistic now only measures goodness of fit close to this region. We achieve this by multiplying the summands in (8) and (9) by a kernel function. For example, (8) becomes

$$C^* = \frac{1}{Z}\sum_{i=1}^{Z}\left[W_{(i)} - \widehat{F}_G^{-1}\left(\frac{i}{Z+1}\right)\right]^2 * K(i,\alpha)$$

where $K(i, \alpha)$ can be any of a number of common kernel functions, and α is the cutoff p-value of interest. An example of a Gaussian kernel function in this context with scale $(2\alpha)^2$ is

$$K(i, \alpha) = exp\left[-\frac{\left(\frac{i}{Z+1} - (1-\alpha)\right)^2}{(2\alpha)^2}\right]$$

The magnitude of down weighting as $i/(Z+1)$ moves away from the point of interest, $(1-\alpha)$, is adjusted with the scale term, i.e., the kernel bandwidth. The p-value obtained with this technique is somewhat sensitive to the choice of scale: setting the scale term too low will target the cutoff point too narrowly, over-stressing the importance of the subjective choice of cutoff, α. At the other extreme, setting the scale too high will include regions of the distributions being compared that are far from the point of interest, and may thus include irrelevant discrepancies, which of course defeats the purpose of the kernel modification.

With the scale term set to $\alpha = 0.05$, even the extreme right order statistics get a non-zero weight (≈ 0.4), and we thus consider this a somewhat conservative bandwidth. In our simulations, the difference between W_i and $\widehat{F}_G^{-1}\left(\frac{i}{Z+1}\right)$ was typically very small in the region of interest ($\frac{i}{Z+1} \approx 0.95$). The modified version of the C-statistic using a kernel thus gave high goodness of fit p-values for most of the scenarios explored (see Table 4). We found that for all simulations the bandwidth could be adjusted downward such that C^* became significant (at level $\alpha' = 0.05$) for the 0.95 quantile of the permutation distribution. Of course, as with all selection criteria, the bandwidth of the kernel function must be chosen before performing the model fitting procedure.

The utility of the C-statistic is that one might well reject a best-fit Gamma distribution based on the K-S statistic alone when in fact the fit is rather good in the region of interest. We therefore recommend accepting the plausibility of a particular Gamma fit if either statistic is significant, or if the kernel-modified C^* is significant. The K-S and unmodified C-statistic give overall GoF, while the modified C-statistic gives a targeted GoF for near the cutoff region.

Figure 5 shows the type I error rate associated with using a Gamma approximation at various numbers of permutations, compared to that of the raw permutation distribution (F_W) for $Z = 10{,}000$ permutations. The data

$X \sim N(0, \Sigma_1)$				
Folds	**C**		**Modified C**	
	3 Learners	5 Learners	3 Learners	5 Learners
4	0.40	0.25	0.46	0.23
8	0.025	0.023	0.068	0.067
$X \sim N(0, \Sigma_2)$				
	3 Learners	5 Learners	3 Learners	5 Learners
4	0.039	0.069	0.060	0.12
8	0.096	0.0178	0.15	0.02
$X \sim$ Bernoulli				
	4 Learners	5 Learners	4 Learners	5 Learners
10	0.078	0.052	0.19	0.09

Table 4: Goodness of fit p-values from the C-statistic and modified C-statistic for the same simulation scenarios as table 3. The p-value is greater for the modified C for every case but one, indicating that the Gamma fit in the region of interest (0.95 quantile) is better than the overall fit in these cases. Note that for some of the simulations the p-value is significant (assuming a cutoff of $\alpha' = 0.05$) for the modified C but not for the unmodified version.

simulated were from the worst case scenario of table 3 (high correlation between covariates), in the sense that the Gamma fit was the poorest for any of the categories of data we simulated. It thus represents a worst case scenario for a Gamma-fitting procedure. Error rates were computed for the various subsamples of F_W by finding the fraction of the ordered subsample to the right of $F_W^{-1}(0.95)$ where F_W was the permutation (i.e. null) distribution. $F_W^{-1}(0.95)$ was considered the "true" 95th quantile for the given simulated data set (even though F_W, as we use the term here, is, of course, an empirical estimation of the true null distribution of W). In other words, for subsample s, the estimated error rate based on a Gamma fit for that subsample is given by

$$1 - F_W\left(\widehat{F}_{s_G}^{-1}(0.95)\right) \tag{10}$$

where F_s is just the distribution function of subsample s and $\widehat{F}_{s_G}$ is the Gamma distribution whose parameters were estimated from s. Similarly, using the ordered subsample without fitting a Gamma to it, the error rate

is given by $1 - F_W\left(F_S^{-1}(0.95)\right)$, where F_s is just the distribution function of the subsample.

Figure 5 shows the distributions of (10) for each of 5000 sets of subsamples, each of which includes various subsample sizes. Each of the subsamples was cumulative in the following sense. First, a subsample of 50 permutations was drawn from the original 10,000 permutations. Then an additional 50 were drawn from the remaining 9950 permutations to give a subsample of 100, and so on up to a subsample size of 2000. Thus the 5000 sets of subsamples mentioned above consisted of 5000 independent sets of values at each subsample size, gathered in the manner just described.

The figure suggests the Gamma-based error estimates are very slightly biased at larger subsample sizes compared to those based on the ordered subsample itself, but the Gamma-based variances of the estimates are also lower. This is also noticeable in Figure 6, which shows box plots of the distribution of estimates of the 95th quantile for 5000 independent sets of subsamples of the original 10,000 permutations. The variance of $\widehat{F}^{-1}(0.95)$ at each subsample size is clearly lower for the Gamma-fitted estimates than for the raw subsamples themselves.

The lower variance for the Gamma-fitted estimates leads to a lower MSE, and therefore beats the raw permutation estimates at the various subsample sizes. The real question is what an acceptable trade off is between number of permutations and variance in the expected error rate. We have no definitive answer for this question, but it seems a permutation sample of between 500 and 1000 would correspond to an acceptable error rate error.

To summarize, a recommended procedure for using Gamma-fitting to estimate the p-value of W^* is as follows.

1. Select the data-adaptive algorithms that are to be used in the Super Learner and obtain W^* from the original data.

2. Permute the data as described in the Simulations section, $Z = 500$ to 1000 times.

3. Run the Super Learner on each permutation to obtain W_z and record it.

4. With F_W now in hand, use method of moments (or MLE) to estimate the parameters of $\widehat{F}_G$. Compute K-S and C p-values. If any of the

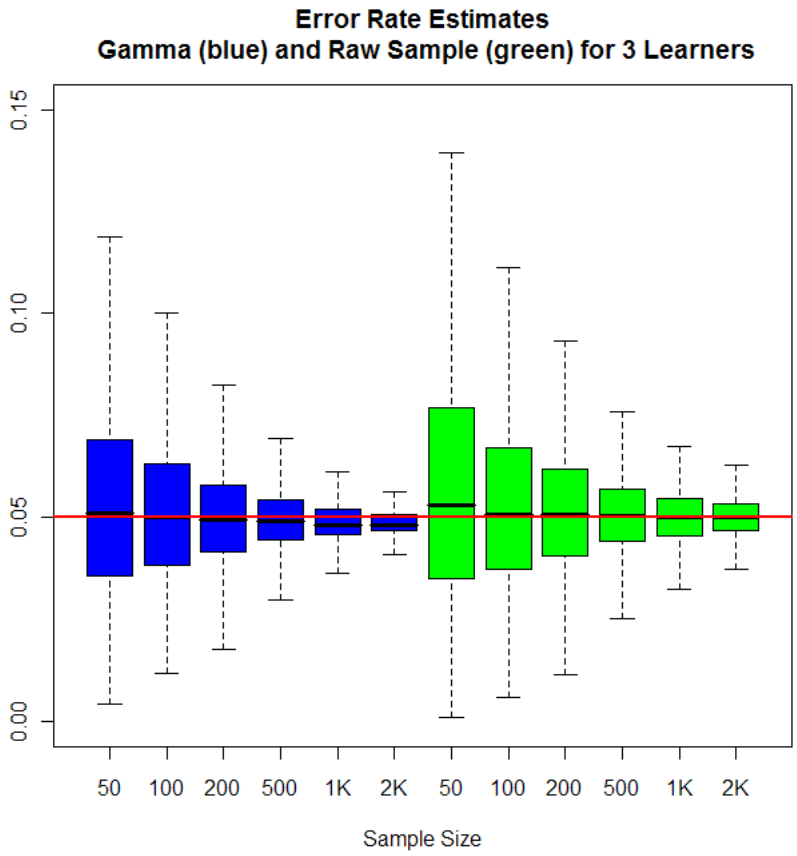

Figure 5: Estimated type I error rates from 5000 sets of subsamples of the original 10,000 permutations. Subsample sizes were 50, 100, 200, 500, 1000 and 2000. The simulated data had high correlation between covariates, and the data here is for 3 learners and 8 cross-validation folds.

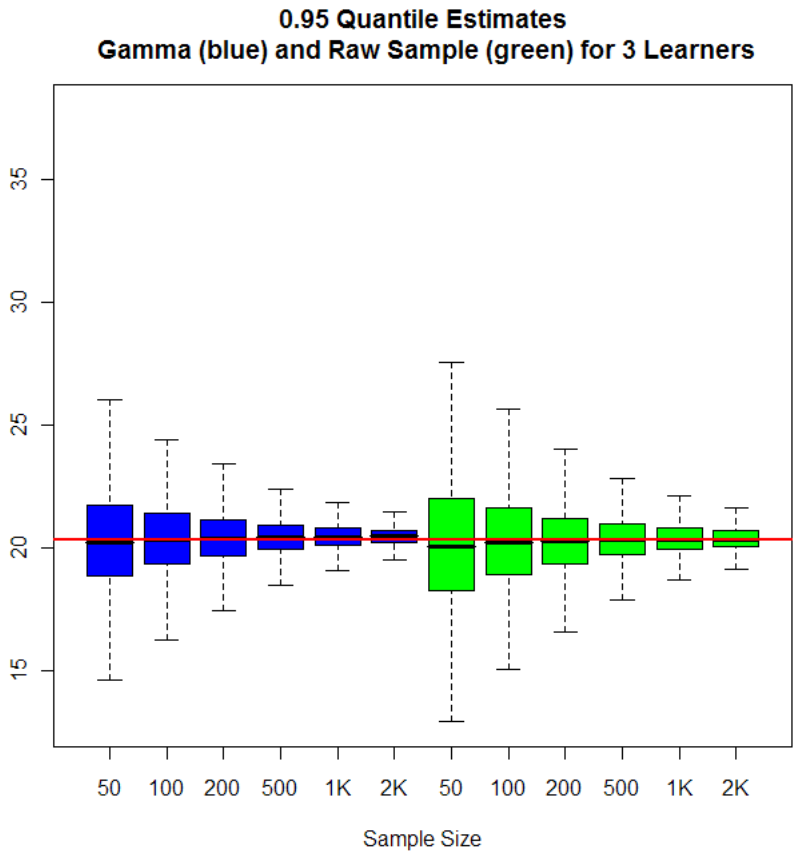

Figure 6: Distribution of 0.95 quantiles from 5000 sets of subsamples of the original 10,000 permutations for a simulation with high correlation between covariates. Subsample sizes were 50, 100, 200, 500, 1000 and 2000. The data here is again for 3 learners and 8 cross-validation folds. The red line represents the "true" 95th quantile obtained from the permutation distribution, F_W.

methods (K-S, C or kernel-modified C) produces a p-value ≥ 0.05 - 0.1, consider the fit good, go to the next step. Otherwise stick with raw permutation distribution with ≥ 3000 permutations, and reject the null if $p_{pw} \leq \alpha$.

5. Reject the null if $W^* > \widehat{F}_G^{-1}(1-\alpha)$.

Note that if many data sets are to be tested for the global null mentioned in the introduction, and all have the same type of distribution of marginal X and marginal Y, then one could perform the procedure above with a very large number of permutations (say, $Z = 10{,}000$) on one data set, and thereby get a much better sense of the goodness of fit of $\widehat{F}_G$, and a better approximation to the true F_G (if the null distribution of W truly is Gamma).

Conclusions

The results show that aggressive data-adaptive regression techniques can be used to generate powerful tests of association when the relationship between covariates and binary outcomes is subtle. Arguably, the most adaptive of all possible data adaptive algorithms for this purpose when the true data generating distribution is unknown is the Super Learner.

We expect the Super Learner algorithm to perform ever better as more algorithms are included in its library. The number of learners can be polynomial in sample size. Thus for sample sizes greater than 100, the number of learners one could potential include is enormous. In the simulations here, many more learners could have been included, but the computation time was too great given that we needed to simulate 500 or more data sets.

In practice, we believe it is reasonable to perform model fitting (specifically, Gamma) to the permutation distribution for a small number of permutations, on the order of 500 - 1000, when the data is of the type assumed in our simulations, and with up to five learners in the Super Learner. If the Gamma fit is good according to the criteria discussed in the previous section, it is reasonable to compare W^* to the fitted Gamma for null hypothesis rejection.

Though we have focused on binary outcomes, the method can easily be extended to categorical and continuous outcomes. One would of course want

to ensure that the Super Learner library includes algorithms that are appropriate for those outcomes.

References

Merrill D. Birkner, Alan E. Hubbard, and Mark J. van der Laan. Data adaptive pathway testing. *U.C. Berkeley Division of Biostatistics Working Paper Series*, Working Paper 197, 2005. http://www.bepress.com/ucbbiostat/paper197.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. Brieman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA, 1984.

J.J. Goeman, S.A. van de Geer, F. de Kort, and H.C. van Houwelingen. A global test for groups of genes: testing association with a clinical outcome. *Bioinformatics*, 20(1):93–99, 2004.

James W. Hardin and Joseph M. Hilbe. *Generalized Linear Models and Extensions*. Stata Press, 2nd edition, 2007.

Z. Jiang and R. Gentleman. Extensions to gene set enrichment analysis. *Bioinformatics*, 23:306–313, 2007.

Charles Kooperberg, Smarajit Bose, and Charles J. Stone. Polychotomous regression. *Journal of The American Statistical Association*, 92:117–127, 1997.

P. McCullagh and J.A. Nelder. *Generalized Linear Models*. Chapman and Hall, Boca Raton, USA, 1989.

Mee Young Park and Trevor Hastie. Penalized logistic regression for detecting gene interactions. *Biostatistics*, 9(1):30–50, 2007.

Eric Polley, 2009. Eric Polley, private communication.

Ingo Ruczinski, Charles Kooperberg, and Michael L. LeBlanc. Logic regression. *Journal of Graphical and Computational Statistics*, 12:475–511, 2003.

A. Subramanian, P. Tamayo, and V.K. Mootha. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.

Mark J. van der Laan, Eric C. Polley, and Alan E. Hubbard. Super learner. *U.C. Berkeley Division of Biostatistics Working Paper Series*, Working paper 222, 2007. http://www.bepress.com/ucbbiostat/paper222.