UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

A Simulation Study Comparing the Use of Supervised Machine Learning Variable Selection Methods in the Psychological Sciences

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

In partial fulfillment of the requirement for the degree of

MASTER OF SCIENCE

By

CATHERINE BAIN

Norman, Oklahoma

2022

A Simulation Study Comparing the Use of Supervised Machine Learning Variable Selection Methods in the Psychological Sciences

A THESIS APPROVED FOR THE

DEPARTMENT OF PSYCHOLOGY

BY THE COMMITTEE CONSISTING OF

Dr. Jordan Loeffelman

Dr. Lauren Ethridge

Dr. Dingjing Shi

Acknowledgements

First, I would like to thank Dr. Jordan Loeffelman for her continued support throughout this process. I would also like to thank my committee: Dr. Dingjing Shi and Dr. Lauren Ethridge. I truly appreciate all the feedback and support I was given throughout this process. In addition, thanks to the Ethridge Lab for the use of their data, with special thanks to Jordan Norris for walking me through that data so I may better understand it and for her continued support throughout this entire process. Another thank you to Dr. Cassie Boness for the use of her data.

To my family, who pushed me to apply to graduate school in the first place and continue to support me during my time here. To Dr. Ankur Gupta for always providing a fresh lens through which to view my project. To Dr. Cassidy Krantz for all her continued feedback and reminders to take breaks – whether those be in the Wichita Mountains or just playing a quick game of Bananagrams. To Margaret Cox and Katie Ellis who continually remind me that I am more than my work here in this program.

Finally, I'd like to thank all my current and former colleagues and professors who have encouraged me throughout my entire academic journey so far. Who knows where I would be without them.

Table of Contents

Abstract

When specifying a predictive model for classification, variable selection (or subset selection) is one of the most important steps for researchers to consider. Reducing the necessary number of variables in a prediction model is vital for many reasons, including reducing the burden of data collection and increasing model efficiency and generalizability. The pool of variable selection methods from which to choose is large, and researchers often struggle to identify which method they should use given the specific features of their data set. Yet, there is a scarcity of literature available to guide researchers in their choice; the literature centers on comparing different implementations of a given method rather than comparing different methodologies under vary data features. Through the implementation of a large-scale Monte Carlo simulation and the application to three psychological datasets, we evaluated the prediction error rates, area under the receiver operating curve, number of variables selected, computation times, and true positive rates of five different variable selection methods using R under varying parameterizations (i.e., default vs. grid tuning): the genetic algorithm (*ga*), LASSO (*glmnet*), Elastic Net (*glmnet*), Support Vector Machines (*svmfs*), and random forest (*Boruta*). Performance measures did not converge upon a single best method; as such, researchers should guide their method selection based on what measure of performance they deem most important. Results do, however, indicate that the genetic algorithm is the most widely applicable method, exhibiting minimum error rates in hold-out samples when compared to other variable selection methods. Thus, if little is known of the format of the data by the researcher, choosing to implement the genetic algorithm will provide strong results.

The sheer volume of data available to behavioral scientists has increased exponentially in recent years in many parts thanks to the internet. Through websites such as Amanzon's Mechanical Turk, it is now easier and cheaper for researchers to recruit large, diverse samples of participants (Buhrmester et al., 2011; Crump et al., 2013; Gosling et al., 2004; Ipeirotis et al., 2010). Mechanical Turk has even been shown to be useful in the collection of clinical data (Arditte et al., 2016; Shapiro et al., 2013). In addition, online data repositories such as the University of Michigan's Inter-university Consortium for Political and Social Research, Harvard's Institute for Quantitative Social Science Dataverse Network, and the National Institute of Mental Health's Data Archive provide researchers with opportunities for secondary data analysis on large datasets that may or may not have been able to have been addressed within a traditional lab setting (Yarkoni, 2012).

These large datasets are often referred to as "big data". The umbrella of big data covers not only these big internet-based datasets containing large sample sizes and large numbers of variables but also large national studies looking at many psychological factors at once and integrated data (i.e., pooled or coordinated analysis across several data sets; Curran & Hussong, 2009; Hofer & Piccinin, 2009). Big data is often characterized with several big Vs. Originally there were only 3 Vs of big data: Volume, Variety, and Velocity (Laney, 2001). Volume refers to the amount of data – that is, the number of observations and the number of variables in a dataset. Different types or kinds of variables, such as some numeric and some categorical variables, determine the variety of a dataset. Velocity indicates the speed at which the data is collected. As big data has become more popular, the number of Vs has grown. As of 2016, there were at least 19 unique Vs used to characterize big data (Cartledge, 2016). This is an indication that "big"

may not mean the same thing across all researcher disciplines or even across individual researchers.

Perhaps what is deemed "big" is a matter of perspective. For behavioral scientists, if looking at a multi-site or population study, it is likely big data may be comprised of thousands of observations of hundreds of variables while big data comprised of a clinical or convenience sample of may only contain a few hundred observations of a couple dozen variables. Yet, to computer science researchers, samples of these size are nowhere near what they would deem "big data". As such, it makes sense that behavioral scientists have begun to look to those in computer science, specifically those within the area of machine learning, for techniques to use to analyze these large datasets.

Machine learning is an area of computer science that aims to detect patterns in data (Kodratoff, 2014), firmly established on an underlying foundation of statistical principles referred to as statistical learning theory (SLT). As mentioned previously, big data is often comprised of a large number of variables; it may even be the case that the number of variables outweighs the number of observations. This often poses a problem for researchers when it comes to analysis. For one, many common statistical methods do not handle large numbers of predictors, but even when considering methodologies that do handle many predictors, in practice, researchers should work to minimize the number of predictors required for obtaining outcome predictions to improve efficiency. For example, rather than using a lengthy diagnostic tool, one may prefer to use only a subset of the most important questions (or predictor variables) as a screening tool. To solve this problem, researchers implement dimension reduction, or variable selection, techniques.

If approaching a dimension reduction problem with the desire to account for a given dependent variable, behavioral science researchers will most often employ the use of stepwise regression, adding and removing variables one at a time. Assuming the study has reasonable power, and the assumptions of the chosen model are met, this can be an effective method to test statistical null hypotheses about given predictor variables. However, not all research directly translates to null hypothesis testing; some research requires examining a large pool of potential predictors, developing a model with generalizable predictive power, or both (Chapman et al., 2016). In these kinds of research, it is possible that a researcher may be interested in examining anything related to a given outcome variable rather than determining the significance of a single predictor variable. Typically, this kind of research is driven by a more general hypothesis about types of items may be linked to the outcome variable, often in hopes to create what is called a criterion-keyed scale (Anastasi & Urbina, 1997).

Criterion-keyed scales are meant to predict a particular outcome or criterion, for example a scale created to screen for depression like the Center for Epidemiological Studies – Depression (CES-D) or alcohol use disorder like the Alcohol Use Disorders Identification Test (AUDIT) (de Meneses-Gaya, 2011; Radloff, 1977). When trying to form criterion-keyed scales, stepwise regression is not the most effective methodology to use because it emphasizes statistical significance over predictive power. Simulation studies have revealed that stepwise regression tends to perform poorly in terms of recovering the true structure of the model (Derksen & Keselman, 1992; Kok et al., 2021; Whittingham et al., 2006; Wiegand, 2010), and does not address a common problem researchers face when creating a model: overfitting. Overfitting occurs when a model fits the data it is trained on exceptionally well (sometimes exactly), but does not generalize well to other, unseen data. To combat this problem of overfitting, researchers

have turned to more complex machine learning techniques such as the Least Absolute Shrinkage and Selection Operator (LASSO), Elastic Net, Random Forest, Support Vector Machines (SVM), and the Genetic Algorithm (GA) in hopes that they will yield more promising and more generalizable results.

These machine learning techniques work to combat overfitting through something called the bias-variance tradeoff. Bias refers to the average error between the predicted model value and the true value and can be conceptualized as $bias = avg(f'(x) - f(x))$ while variance refers to the average variability in the model prediction for the given dataset and can be conceptualized as $variance = avg[(f'(x) - avg[f'(x)])^2]$. The bias of the model tells us the capacity of the model to predict the true values while the variance of the model tells us how much the model can adjust to a change in the dataset (or an introduction of new data). With these definitions in place, we can see that a model which is high in bias is overly simplified, underfit, and high in error on both the data it was created on, and new data. In addition, we can see that a model which is high in variance is overly complex, overfit, and has low error on the data it was created on but high error on new data. As a result, good models need to balance bias and variance to combat both under and overfitting. The bias variance tradeoff works to slightly increase bias to reduce variance, or slightly increase variance to reduce bias, that is to say, a compromise between bias and variance must be made to create a well fit model. More traditional variable selection models such as stepwise regression work well fit a model to the a given dataset but will perform poorly with the introduction of new data (high variance, low bias). Machine learning techniques are also known to overfit data, but applying cross validation techniques like 5-fold or 10-fold cross validation can mitigate this issue. Cross validation is a resampling method that uses different portions of the data to test and train a model on different iterations. For example, 5-fold cross

validation will partition the dataset into 5, and then iteratively test the model on different sets of those partitions. On the first iteration, the machine learning technique will use the first four partitions as a training set and the last as a test set, on the second iteration, it will use partitions one, two, three, and five for the training set and the fourth partition as a testing set. This process will then be repeated until all partitions have been used as the test set. The same can be said for 10-fold cross validation, except the data is partitioned into ten instead of five. The techniques work to find the optimum bias and variance that will minimize the average error across all iterations.

Variable selection machine learning techniques can be applied within the context of supervised or unsupervised learning. In supervised learning, the dataset is already labeled, meaning that the data already contains information about the outcome of interest. For example, a labeled dataset containing information about an individual's drinking habits would include a variable diagnosing the individual or not based on the DSM-5. Unsupervised learning works to analyze and cluster unlabeled datasets, (i.e., the dataset would not contain a diagnosis for the individual). The current paper is looking only at supervised learning techniques. The literature classifies all machine learning variable selection techniques into one of three categories: filter methods, wrapper methods, or embedded methods.

Filter Methods

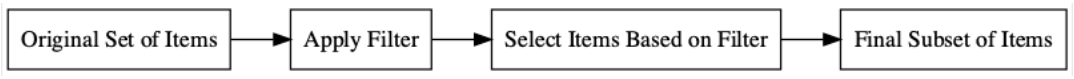

Figure 1. Filter Methods

Filter methods are often used as a preprocessing step but can be used as a stand-alone method. These techniques, when applied to variable selection problems, choose items before building a model to measure the given construct (i.e., items are selected based on a certain characteristic). For example, the filter could select items based on a calculated feature relevance score, the item's correlation with the construct measured, or its level of variance. In cases where significance testing is used to determine item selection (e.g., whether the item has a significant correlation with the outcome variable or not), researchers have recommended controlling for multiple hypothesis tests to prevent inflation of Type I error (Bourgon et al., 2010).

Filter methods are advantageous because of their computational simplicity, their ability to scale well to many measurement items, and their independence from the given classification algorithm (Saeys et al., 2007). There are also disadvantages to filter methods. For one, most proposed filter techniques, such as χ^2, Euclidean distance, or the i-test are univariate. As a result, these techniques may miss an item's interaction with the classifier, an interaction among items, or covariance between items. Some multivariate filter methods have been developed such as the correlation-based feature selection (CFS) statistic, which account for joint statistical properties within a set of items (Hall, 1999); however, these methods lose the advantage of scalability. The biggest disadvantage of filter methods is that they are entirely controlled by the researcher and thus, the guidelines by which to select an item are highly subjective. This paper does not implement any filter methods, but rather includes them here for completeness

Wrapper Methods

Wrapper methods, unlike filter methods, take an item's ability to measure the construct of interest into account when determining whether it should be included in the selection. Each wrapper method operates under a specific algorithmic ideology from machine learning. For

example, forward and backward stepwise regression, which are commonly used as variable selection methods, operate as greedy algorithms, choosing the item that would optimize the selected criteria at each step aiming to provide the best subset solution. Greedy algorithms, however, are prone to reach locally optimal solutions instead of globally optimal solutions. As a result, research has indicated a preference for heuristic-based algorithms (Brusco, 2014; Brusco et al., 2009; Cadima et al., 2004).

In a very simple sense, heuristic methods operate as iterative search procedures. These techniques examine a search space of possible subsets to examine which subset reaches a predefined property. There are $2^k - 2$ possible strict subsets for k items (i.e., a subset differing from the parent set by at least one item), excluding the empty set. Complete enumeration of all subsets would guarantee the optimal solution, but it is computationally expensive and therefore not always feasible. Thus, researchers have turned to the use of wrapper methods which use a systematic approach to determine candidate subsets to assess. One wrapper technique that searches the solution space for candidate item sets effectively and efficiently is the genetic algorithm (GA; Shen et al., 2005; van der Linden, 1998; Yarkoni, 2010).

Genetic Algorithm

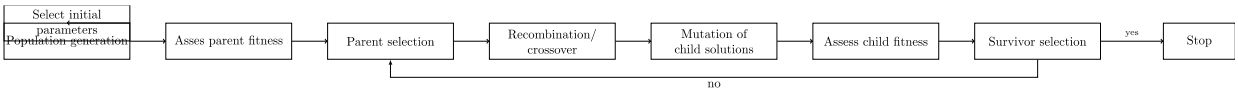

Figure 2. The Genetic Algorithm

Variable selection problems, at their heart, are combinatorial optimization problems. Combinatorial optimization problems work to find the optimal set of items to fulfill a given goal while following a specific set of constraints. The genetic algorithm has been used to solve many types of combinatorial optimization problems including graph coloring (Hindi & Yampolskiy,

2012), multidimensional scaling (Groenen et al., 2000), and clustering (Cowgill et al., 1999) suggesting it would perform well on variable selection problems. The genetic algorithm is an evolutionary algorithm using principles mimicking natural selection to explore solutions for the combinatorial problem (Goldberg, 1989; Holland, 2019). The algorithm begins with a random set of two potential solutions to the problem (i.e., chromosomes) and recursively improves the solution through crossover and mutation until it reaches a subset that optimizes the selected criteria (e.g., a balance between predictive ability and subset size). Within each chromosome, a gene of 1 indicates that the item has been chosen for the subset, while a gene of 0 indicates that the item has not been chosen. Crossover, or recombination of the two parent solutions, can occur at any given point in the selected parent solution set. Figure 2 demonstrates a one-point blocked crossover. In blocked crossover, a random number indicates where to split the initial solution sets (i.e., "parents"). Two solutions (i.e., "children") are then created by obtaining the set of variables before the split for each parent and passing these on to the children (i.e., parent 1 gives their genes before the split to child 1, and parent two gives their genes before the split to child 2). Then, the genes of each parent after the split are given to the opposite child (i.e., parent 1 passes their genes after the split on to child 2, and parent 2 passes their genes after the split on to child 1). It is important to note that although Figure 3 illustrates single-point crossover, crossover could occur in many forms. For example, a random percentage of genes could be selected instead of setting a single point on which to split the genes or genes could be chosen at random instead of in a block. After children are produced through crossover, mutation can be performed in a small percentage of the solutions so as to obtain new information that could not be obtained from the parents.

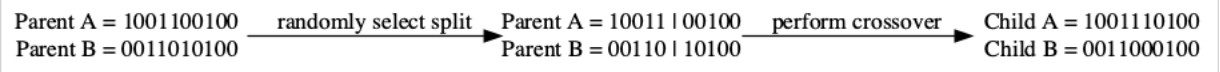

Figure 3. Single point crossover in the GA

The children are then evaluated based on the optimization criteria, and the optimal child is chosen as the survivor to continue the next iteration. For this study, the ARI was used as the optimization criteria. The ARI is a measure of agreeability between the predicted classifications and the true classifications that can be calculated with the following equations:

$$ARI = \frac{RI - Expected\ RI}{\max(RI) - Expected\ RI}$$

$$RI = \frac{\#\ of\ agreeing\ pairs}{\#\ of\ disagreeing\ pairs}$$

The genetic algorithm (GA) is a machine learning approach which has been shown to perform just as well as, if not better than, traditional methodological approaches such as stepwise regression or Principal Component Analysis when constructing a short form (Sahdra et al., 2016; Sandy et al., 2014). Unlike traditional methods, GA is entirely automated, highly sophisticated, and does not require researchers to manually balance various criteria when implementing it for the use of item selection.

Yarkoni performed three variable selection studies validating the use of the GA on personality measures (2010). The first study provided evidence that the GA was a successful method for programmatically generating a 10-item Big-Five inventory from a pre-existing 44-item inventory. Moreover, this study also found its 10-item measure produced results almost as reliable as the 44-item measure (Yarkoni, 2010). The second study aimed to produce a shortened version of the 240-item NEO-PI-R (a widely used personality measure assessing 30 distinct 'facets' of personality). This study produced a 56-item measure capturing most of the variance

from the original 240-item measure (Yarkoni, 2010). The third study aimed to expand the methodology used in the second study to shorten multiple scales at once. This study produced a 181-item measure which reliably captured most of the variance from the original 2019 items (Yarkoni, 2010). These three studies indicated that the GA was a reliable methodology for use on variable selection problems within the field of personality measures. The genetic algorithm has since been applied to other personality measures to derive short forms. Eisenbarth et al. created a short form of the widely used Psychopathic Personality Inventory-Revised (PPI-R) containing only 40-items (as compared to the original 154 items; 2015). This shortened form showed highly convergent correlations and outperformed an alternative measure generated using more typical methods such as Item Response Theory or retaining the top N items for each scale (Eisenbarth et al., 2015).

Schroeders et al. examined the use of the GA in the field of education, developing a short form from an 89-item picture-based vocabulary test (2016). Researchers compared the use of the GA, ant colony optimization (ACO) – another heuristic based technique, and stepwise confirmatory factor analysis (SCOFA), a previously validated method for variable selection, and found that both the GA and ACO performed better than SCOFA, indicating that the GA is a reliable methodology for variable selection problems within the field of education.

The genetic algorithm has also been applied to variable selection problems within the clinical realm of psychology. For example, Sahdra et. al. applied the GA to abbreviate the Multidimensional Experiential Avoidance Questionnaire (MEAQ) which is used within the clinical realm of psychology (2016). This implementation reduced the MEAQ by slightly more than 50%, reducing the original 62-item measure to a 30-item measure while maintaining inter-subscale correlations, factor structure, and factor correlations (Sahdra et al., 2016). Rachmani et

al., were able to reduce a 47-item health literacy questionnaire (HLS-EU-Q47) to a 10-item questionnaire (2019). It is important to note that the HLS-EU-Q47 is multidimensional, and Rachmani et al. were able to maintain multidimensionality with the 10-item scale (2019).

Embedded Methods

Embedded methods are similar to wrapper methods in that they consider how well the candidate item set predicts the given outcome variable. Embedded methods perform item selection while simultaneously estimating the prediction model during the training stage (Guyon & Elisseeff, 2003). As a result, embedded methods rely highly on their specified optimization function. This contrasts wrapper methods where feature selection and model evaluation happen relatively independently, making the model easy to estimate but computationally intensive to obtain. Some embedded methods, such as decisions trees like CART (Brieman et al., 2017), have been used for decades to perform variable selection tasks. Other embedded methods such as Least Absolute Shrinkage and Selection Operator (LASSO) (Tibshirani, 1996) are newer methods to the study of variable selection. The following embedded methods will be evaluated by the current paper and have demonstrated success in variable selection problems: LASSO (Descloux & Sardy, 2021; Fonti, 2017; Kok et al., 2021; T. T. Wu et al., 2009), Elastic Net (Algamal & Lee, 2015b; Liu & Li, 2017; Marafino et al., 2015), Random Forest (Diaz-Uriarte & de Andres, 2005; Genuer et al., 2010; Hapfelmeier & Ulm, 2013), and Support Vector Machines (SVM; Becker et al., 2009; Neumann et al., 2005; Rakotomamonjy, 2003).

Regularization Techniques

Both LASSO and Elastic Net are considered regularization techniques. Regularization is a common method used to combat issues of overfitting found in models estimated with maximum likelihood estimator (like logistic regression). There are three main regularization

techniques: Ridge Regression, LASSO Regression, and Elastic Net Regression. Each of these techniques works to combat overfitting by intentionally adding a small amount of bias into the optimization function such that a generic regularization function takes the form:

$$L^{Reg}(\beta) = L^{mle}(\beta) - \lambda P(\beta)$$

where L^{Reg} is the penalized optimization function, L^{mle} is the standard maximum likelihood estimate (MLE) optimization function, λ is a regularization parameter (i.e., a tuning parameter), and P is a penalty function that varies depending on the type of regularization being applied. The example here is given in terms of logistic regression because this paper focuses on applications with categorical outcome variables, but it is important to note that regularization techniques also work with OLS regression – however, since OLS minimizes the optimization function, the penalty would be added instead of subtracted to the overall function. Determining the magnitude of the penalty is of vital importance because the function must be penalized enough to reduce overfitting but not too much as to render meaningless estimates. Through this controlled introduction of bias, the variance of the estimates is reduced so that the estimates are stable, less susceptible to small changes in the data, and thus more generalizable to unseen data (McNeish, 2015).

Ridge

Although Ridge Regression cannot act as a variable selection technique, it is a natural predecessor for both LASSO and Elastic Net and is thus included for completeness. As mentioned above, regularization techniques each apply a unique penalty to the normal regression function which introduces a small amount of bias to the model. Ridge regression's (Hoerl & Kennard, 1970; McDonald, 2009) penalty is applied to the sum of the squared coefficients such that $P^{ridge}(\beta) = \sum_{i=1}^{p} \beta_i^2$. Therefore, the optimization function for ridge regression takes the form:

$$L^{Ridge}(\beta) = \sum_{i=1}^{P} log\ (P_\beta(Y_i|X_i)\ -\ \lambda \sum_{i=1}^{P} \beta_i^2$$

where $L^{Ridge}(\beta)$ is the optimization function (or the loss function), $\sum_{i=1}^{P} lo\ g\big(P(x_i|\theta)\big)$ is a summation of the MLE for all predictor variables, $\lambda \geq 0$ is the regularization parameter that controls the degree of shrinkage, and β_i is the given regression coefficient. By using the sum of the squared coefficients, the magnitude of the shrinkage is proportional to the magnitude of the coefficient estimate, that is to say, the larger the coefficient estimate, the greater the shrinkage (Hesterberg et al., 2008; McNeish, 2015). As $\lambda \longrightarrow 0$, ridge estimates approach the original MLE coefficient estimate. However, as $\lambda \longrightarrow \infty$, ridge estimates asymptotically approach zero meaning that they will never shrink to zero and thus, ridge regression does not perform de facto variable selection (McNeish, 2015). There is an advantage to ridge regression not performing model selection: the ability to calculate standard errors. Additionally, ridge regression is able to handle collinear predictors (McNeish, 2015). Ridge regression is not implemented in this study as a result of its inability to perform variable selection, and thus, it is not discussed further.

LASSO

LASSO (Tibshirani, 1996) penalizes the loss function by setting a constraint on the absolute value of the sum of the regression coefficients which allows the regression coefficients to shrink. The LASSO penalty can be written as $P^{LASSO}(\beta) = \lambda \sum_{i=1}^{P} |\beta_i|$, thus the LASSO optimization function can be written as a penalized MLE function for a dataset with *P* predictor variables such that:

$$L^{LASSO}(\beta) = \sum_{i=1}^{P} log\ (P_\beta(Y_i|X_i) - \lambda \sum_{i=1}^{P} |\beta_i|$$

where $L^{LASSO}(\beta)$ is the loss function, $\sum_{i=1}^{P} log\ (P(x_i|\theta))$ is a summation of the MLE for all predictor variables, $\lambda \geq 0$ is the regularization parameter that controls the degree of shrinkage,

and β_i is given regression coefficient. Because the penalty is applied to the sum of the absolute values of the regression coefficients, estimates may shrink to exactly zero, therefore deselecting them from the model (Tibshirani, 1996). If multiple coefficients obtain a value of 0, the model is deemed sparse (McNeish, 2015). It is important to note that LASSO applies the λ penalty equally across all regression coefficients resulting in coefficients with larger magnitudes (i.e., coefficients with stronger potential importance) shrinking less (Hesterberg et al., 2008).

Although the use of the absolute value of the regression coefficients when applying the penalty is advantageous in its ability to select (or rather deselect) variables, it also results in a LASSO optimization function that is not differentiable, meaning that to find the signs of the regression coefficients, the algorithm must search through the entire 2^p possible sign combinations to find the correct ones (McNeish, 2015). Therefore, LASSO must employ optimization functions to search through these combinations in a computationally efficient manner.

There are many optimization functions that can be applied to LASSO including Tibshirani's (1996b) quadratic programming algorithm which requires $n2^p$ computations – still a computationally intensive strategy – or an implementation of least angle regression (LARS) which requires np^2 computations (Efron et al., 2004). This paper implements a version of LASSO from the glmnet R package which uses cyclical coordinate descent (Friedman et al., 2010a)

The use of cyclical coordinate descent for LASSO has been proposed numerous times (Daubechies et al., 2004; Shevade & Keerthi, 2003; van der Kooij, 2007) but has gained traction more recently (Friedman et al., 2007, 2010a; Genkin et al., 2012; Krishnapuram et al., 2005; Shevade & Keerthi, 2003; T. T. Wu et al., 2009; T. T. Wu & Lange, 2008). Cyclic coordinate

descent works by iterating through the signs one at a time, optimizing the function of interest (the LASSO function) with respect to one beta sign at a time and then repeating this process until no change is seen in the optimization function. For a more technical understanding, see the above papers. Research indicates that this strategy is remarkably efficient. The implementation of LASSO in glmnet which uses this technique has been shown to outperform all competitors in terms of algorithmic efficiency and speed (Friedman et al., 2010a) and as a result has become the most used implementation of LASSO.

Previous research has indicates that LASSO selects the most relevant variables when building a model (Fonti, 2017), removes irrelevant variables without compromising the integrity of the hypothesized model (Kok et al., 2021) and provides more parsimonious variable selection and more generalizable inferences than stepwise regression, OLS, or MLE estimations (McNeish, 2015) when compared directly. Statisticians are also quick to point out that behavioral science is lagging in its implementation of LASSO (Johnson & Sinharay, 2011; McNeish, 2015) even as software for its implementation, such as the glmnet R package (Friedman et al., 2010a), become increasing available in programs familiar to behavioral science researchers.

Although there is significant evidence that LASSO performs well on variable selection problems, there is also evidence of certain drawbacks. There are three main scenarios where limitations of LASSO have been identified. First, if the number of predictors (p) is larger than the number of observations (n), LASSO will only select up to n predictor variables. Second, if there is a set of predictors that have high pairwise correlations, LASSO will, at random, select one to be representative although it may not be the strongest predictor within the set. Third, if predictors are highly correlated with each other, LASSO does not seem to perform as well as Ridge Regression (Zou & Hastie, 2005). The first two scenarios illustrate that LASSO is not

necessarily the best variable selection technique in all situations. A related regularization technique, Elastic Net, works to overcome these limitations.

Elastic Net

Elastic Net regression (Zou & Hastie, 2005) uses both the ridge penalty and the LASSO penalty, allowing it to leverage the strengths of both methodologies. The Elastic Net loss function takes the form:

$$L^{Elastic\ Net}(\beta) = \sum_{i=1}^{P} log\ (P_{\beta}(Y_i|X_i)\ - \lambda_1 \sum_{i=1}^{P} \beta_i^2\ -\ \lambda_2 \sum_{i=1}^{P} |\beta_i|$$

where $L^{Elastic\ Net}(\beta)$ is the loss function, $\sum_{i=1}^{P} log\ (P(x_i|\theta)$ is a summation of the MLE for all predictor variables, $\lambda_1 \geq 0$ is the regularization parameter that controls the degree of shrinkage implemented by the ridge penalty, $\lambda_2 \geq 0$ is the regularization parameter that controls the degree of shrinkage implemented by the LASSO penalty, and β_i is the given regression coefficient.

Similar to LASSO, Elastic Net performs automated variable selection and continuous estimate shrinkage simultaneously, but unlike LASSO, Elastic Net allows for the selection of multiple variables from a set with strong pairwise correlations. In addition, the inclusion of the Ridge penalty allows for Elastic Net to select all *p* predictor variables when $p \geq n$ and not only *n* as is seen in LASSO. Elastic Net produces a sparse model (meaning a model with few numbers of predictors) with good prediction accuracy, while encouraging a grouping effect which allows for potential superiority over LASSO in certain situations (Zou & Hastie, 2005). Its creators view Elastic Net as a generalization of the LASSO as a valuable tool for model fitting and feature

extraction (Zou & Hastie, 2005); it is then fair to deduce that Elastic Net would also perform well on these types of tasks.

Much of the recent work with Elastic Net and variable selection problems has been outside the field of psychology. For example, (Algamal & Lee, 2015a) examined the performance of Elastic Net on two cancer datasets as it compares to other variable selection techniques including LASSO. They found that Elastic Net, on average, selected fewer variables than LASSO and had higher classification accuracy (Algamal & Lee, 2015a). The same researchers note that in a study examining Elastic Net's abilities to correctly classify cancer patients and select genes, the inclusion of the ridge penalty in the Elastic Net loss function may help to select or omit the highly correlated genes together if their coefficients are close together but prohibits the estimation of the coefficients of these same highly correlated genes to have different magnitudes or different signs (Algamal & Lee, 2015b). Despite this limitation, support was found for the use of Elastic Net in classification problems in that it had high classification accuracy and consistent gene selection (Algamal & Lee, 2015b).

Elastic Net has also been used for feature selection problems, such as text classification (Marafino et al., 2015). Researchers found that applying Elastic Net regularization to classifiers based on clinical notes reduced the number of features selected by more than a thousandfold, making these classifiers more easily interpretable and maintaining performance. In addition, the research showed that Elastic Net selected clinically relevant features that correlated well with current ICU outcomes (Marafino et al., 2015).

Although there is evidence of success in the use of Elastic Net on variable selection problems in fields outside of psychology, very little research within the field has taken advantage of Elastic Net when facing variable selection problems. Overwhelmingly, psychologists still

prefer outdated methods like stepwise regression. At least one psychological assessment journal completely banned studies employing stepwise selection as early as 1995 (Thompson, 1995). This paper hopes to illustrate the power of regularization techniques such as LASSO and Elastic Net when confronting variable selection problems, specifically applied to binary classification problems.

Support Vector Machines

Created by Crotes & Vapnik (1995), the Support Vector Machine (SVM) is one of the most used machine learning techniques for binary classification in many disciplines, but its power has not yet been fully acknowledged in psychological research. In an SVM, the data points are plotted in an p-dimensional space (where p is the number of variables in the given dataset) with the value of each variable being the value of each coordinate. The classification of said data occurs by finding an optimal hyperplane to differentiate the different classes by maximizing the margin between the classes' closest points.

A hyperplane is a simply a generalization of a plane that takes a different form depending on the number of dimensions of the space. In two dimensions, the hyperplane is a line, in three dimensions, a plane, and in more than three dimensions, a hyperplane. Hyperplanes can be considered decision boundaries which classify data points into their respective classes in a multi-dimensional space. Data points falling on one side of the hyperplane are attributed to one class, while points falling on the other side are attributed to a different class. Given a training dataset $(x_1, y_1), \dots, (x_n, y_n), x \in \Re^p, y \in -1,1$ with $x = (x_1, \dots x_p)$- input data and y-class label. The linear SVM separates classes by a linear hyperplane of the form:

$$f(x) = \sum_{i=1}^{p} w_i \, x_i + b$$

where $w = (w_1, \dots, w_p)$ are coefficients of the hyperplane and b denotes the intercept of the hyperplane. The class assignment for a test dataset x_{test} is thus given as $y_{test} = sign[f(x_{test})]$.

The SVM is known as a large margin classifier, and as such, works to maximize the distance between the hyperplane and the closest datapoints. This distance is referred to as the margin. The best or optimal hyperplane that can separate the two classes is thus referred to as the large-margin hyperplane. The margin is calculated as the perpendicular distance from the hyperplane to the closest points. Since the margin is calculated by accounting for specific data points (the closest points), these points are deemed the support vectors since they influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. If we delete or change the support vectors, the position of the optimal hyperplane will also change. There are two main methodologies for determining which hyperplane is optimal: hard margin and soft margin.

In a hard margin SVM, the algorithm finds the optimal hyperplane without tolerating any form of misclassification. Hard margin SVMs are not used in this paper due to their problems with overfitting, and therefore are mentioned solely for completion. A soft margin SVM however, still finds the optimal hyperplane, but it adds a degree of tolerance for misclassification (i.e., a certain number of data points are allowed to be misclassified), allowing for a more generalizable model. Classification of linear data is easy, one simply picks a line, or hyperplane, that divides the two classes well. Where SVM truly shines, however, is in its ability to classify non-linear data.

The original SVM (Crotes & Vapnik, 1995) constructed a linear classifier. Computing the soft-margin SVM classifier minimizes an expression of the form:

$$min_{b,w} \sum [1 - y_i f(x_i)] + \lambda \|w\|_2^2$$

The penalty term $\lambda\|w\|_2^2$ takes the form of the ridge penalty which only shrinks the coefficients but does not set the exactly to zero. The λ parameter here controls for the degree of tolerance for misclassification. If a small enough value is chosen for λ, then this equation will yield the hard-margin classifier.

Since the proposal of the linear SVM, a nonlinear classifier has been developed using a kernel function to artificially project the input vectors (or variables) into a higher dimensional space (often called a feature space), therefore making the data linearly separable (Boser et al., 1992). They referred to this SVM algorithm as the maximum margin training algorithm which has an equation of the form:

$$D(x) = \sum_{k=1}^{p} \beta_k K(X_k, X) + b$$

where the coefficients β_k are the parameters to be adjusted, and the X_k are the training variables. The function $K(X_k, X)$ is a predefined kernel function discussed below (Boser et al., 1992).

Kernel functions can take many forms; the *svm* function in R's e1071 package allows for linear, polynomial, radial basis, and sigmoid kernels. The RBF is one of the most preferred and most used kernel functions when working with data that is non-linearly separable. In concept, it is similar to K-Nearest Neighbor, another popular classification algorithm. Because much of our data in psychology cannot be linearly separated, only the RBF kernel will be discussed further. The RBF computes the similarity or distance between two points X_1 and X_2 and takes the form:

$$K(X_1, X_2) = e^{-\frac{\|X_1 - X_2\|^2}{2\sigma^2}}$$

where σ is the variance of the distribution set by the researcher, $\|X_1 - X_2\|$ is the Euclidean distance between two points X_1 and X_2. The maximum value of the RBF kernel is 1 and occurs

when the distance is 0 and the points are the same, i.e., $X_1 = X_2$. If the points are separated by a very large distance, or are dissimilar, the kernel value is close to 0.

The penalty term for ordinary SVM uses the ridge penalty (L_2):

$$L_2 = \lambda \sum_{i=1}^{p} \|w\|_2^2 = \lambda \sum_{i=1}^{p} \|w\|_2^2$$

The L_2 penalty shrinks the coefficients to control their variance, but it provides no shrinkage of the coefficients to zero, and hence no feature selection is performed. To overcome this restriction, a number of variable selection methods using other penalties have been proposed. The SVM adaptation of the LASSO (L_1) penalty takes the form:

$$L_1 = \lambda \|\mathrm{w}\|_1 = \lambda \sum_{i=1}^{P} |\beta_i|$$

As a result of the singularity of the LASSO penalty function (L_1), the L_1 SVM can automatically select variables by shrinking small coefficients of the hyperplane to exactly zero. Thus, the L_1 SVM is an effective variable selection tool (Becker et al., 2009). A fast implementation of the L_1 SVM is implemented in the *penalizedSVM* package (Fung & Mangasarian, 2004). The L_1 penalty, however, has two limitations, as mentioned previously, in that (1) the number of selected features is bounded by the number of samples, and (2) it tends to select only one feature from a group of correlated features and drops the others.

The SCAD penalty is a non-convex penalty function first proposed by Fan and Li (Fan & Li, 2011). Later, Zhang et al. combined the SVM technique with the SCAD penalty for feature selection (Zhang et al., 2006). The SCAD penalty function for a single coefficient w_i is defined as:

$$p(x) = \begin{cases} \lambda|w_i| & \text{if } |w_i| \leq \lambda \\ -\dfrac{|w_i|^2 - 2a\lambda|w_i| + \lambda^2}{2(a-1)} & \text{if } \lambda < |w_i| \leq a\lambda \\ \dfrac{(a+1)\lambda^2}{2} & \text{if } |w_i| > a\lambda \end{cases}$$

where $w_i,\ i = 1, \ldots, p$ are the coefficients defining the hyperplane and $a > 2$ and $\lambda > 0$ are tuning parameters. Fan and Li (2011) showed that SCAD prediction is not sensitive to the selection of the tuning parameter a. The penalty term for SCAD SVM has the form:

$$p_\lambda(w) = \sum_{i=1}^{p} P_{SCAD(\lambda)}(w_i).$$

For small coefficients $w_i, i = 1, \ldots, p$, SCAD yields the same behavior as L_1. For larger coefficients, however, SCAD applies a constant penalty, reducing estimation bias. In addition, the SCAD penalty holds better theoretical properties than the L_1 penalty (Fan & Li, 2011).

With the success of the L_1 SVM, researchers began looking at other regularization techniques to combine with SVM to create more SVM feature selection methods. Becker et al. (2011) proposed a novel penalty function for SVM classification tasks called Elastic SCAD which implemented a combination of SCAD and ridge penalties, in SVM. In addition, this algorithm adopts an interval search algorithm which, in comparison to fixed grid search, finds rapidly and more precisely a global optimal solution, combatting SVMs sensitivity to tuning parameters (Becker et al., 2011).

Becker et al. (2011) hypothesizes that using the SCAD penalty may be too strict in selecting variables for non-sparse data. Thus, they proposed a modification of the SCAD penalty analogously to Elastic Net which could leverage the advantages of SCAD while avoiding its limitations. Their new combination, termed Elastic SCAD, combined the SCAD and L_2 penalties producing a penalty term of the form:

$$p_\lambda(w) = \sum_{i=1}^{p} p_{SCAD(\lambda_1)}(w_i) + \lambda_2 \|w\|_2^2,$$

where $\lambda_1, \lambda_2 \geq 0$ are tuning parameters. Elastic SCAD was found to outperform the L_1 SVM and SCAD SVM (Becker et al., 2011). Thus, this paper will implement the Elastic SVM in the R *penalizedSVM* package.

Random Forest

Random Forest is a popular machine learning approach for predictive model building (Brieman et al., 2017). Random forests are an ensemble method, creating a collection of decision trees, either classification or regression (Brieman et al., 2017), using binary splits of predictor variables to determine outcomes. Decision trees are favorable for two reasons: (1) they are relatively easy to create, and (2) they offer an intuitive method for predicting outcomes, making them easy to interpret. Despite these benefits, decision trees are low in bias, making them very poor predictors of new data (Bengio et al., 2010) or complex datasets, e.g., large datasets or datasets with complex variable interactions (Speiser et al., 2019). Random Forest directly combats this weakness by creating many classification or regression trees (a forest of trees, if you will) using training datasets which are randomly selected and a random subset of predictor variables for modeling outcomes. Results from each tree are then aggregated to give a predictive model with slightly higher levels bias, allowing the model higher accuracy with new data compared to a single tree model (Speiser et al., 2015). Research indicates that random forests consistently offer among the highest prediction accuracy compared to other models within the realm of classification (Fernández-Delgado et al., 2014).

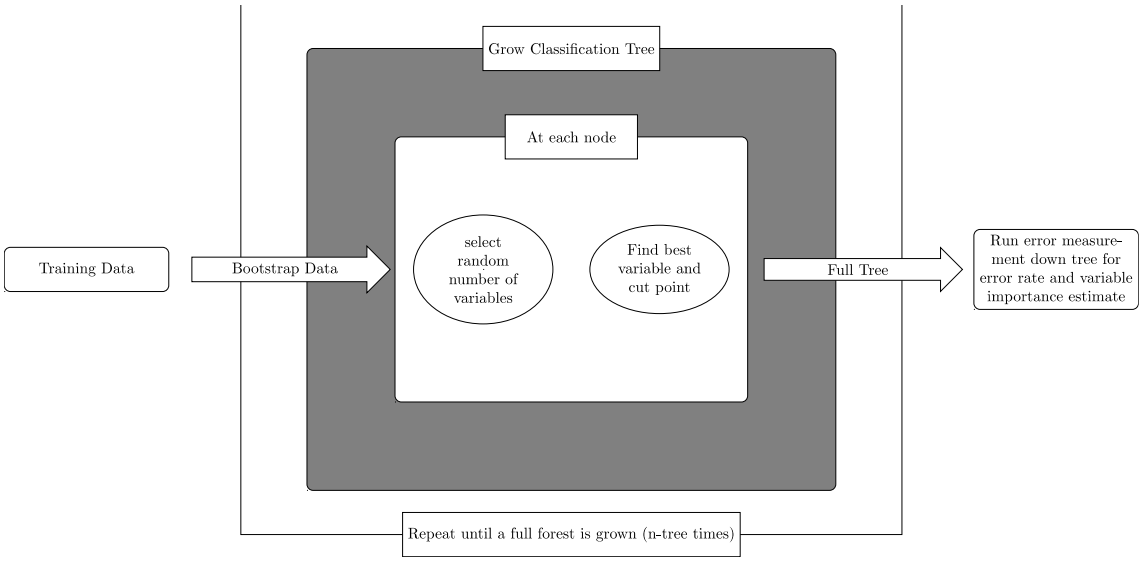

Figure 4. A flowchart for random forest algorithm representing a workflow for classification problems (Ram et al., 2017)

Although it is true that there is little need to fine-tune parameters, researchers note that the most important parameter is mtry, the number of input variables tried at each split, but research indicates that the default value of your chosen function is typically a good choice, even though this value may change across functions (Liaw & Wiener, 2002a). The user also needs to decide how many trees to grow for the forest (n-tree) as well as the minimum size of the terminal nodes (node size). In Figure 4, we see the process of the random forest algorithm representing a workflow for classification problems, but we shall also walk through the steps here:

1) Draw n-tree bootstrap samples from the original data
2) For each of the bootstrap samples, grow an unpruned classification tree with the following modifications:
    a. At each node, rather than choosing the best split among all predictors (as is done when building a single classification tree), randomly sample mtry of the

predictors and select the best split from that subset, then based on the splitting separate this node into two nodes.

   b. Recursively partition the data until the trees reach their largest size (i.e., for each observation, a final node) and the node size reaches its smallest value without which the tree is pruned.

3) Repeat steps 1 and 2 to create a random forest of trees.
4) Predict new data by aggregating the predictions of the n-tree trees (i.e., majority vote for classification)
5) Compute an error rate using new data
   a. this can either be an out-of-bag (OOB) error rate using the data not in the bootstrapped sample, or you can use cross-validation as usual.

In addition, random forest does not require a separate test to calculate an unbiased error estimate of the validation set in the random forest, since it performs this calculation during the execution. However, there is research indicating that minimizing the curve of the OOB prediction error can be misleading, indicating that variable selection should be performed when it may not be necessary or suggested (Svetnik et al., 2000). This research indicates positive performance of cross-validation instead.

There are a wide array of benefits for using random forest for classification problems. Below we have listed seven of the most notable strengths as discussed in (Ram et al., 2017):

1) It is suitable for datasets where there are many more variables than there are observations ($n \ll p$).
2) When most of the predictive variables are noisy (i.e., have a lot of randomness), it has good predictive performance, therefore not requiring pre-selection of predictors.

3) It does not overfit.
4) It can handle a mixture of categorical and continuous predictors and incorporates interactions among predictor variables.
5) There are high quality and free implementations in various statistical software programs such as R and Python.
6) It returns measures of importance for the predictors.
7) There is little need to fine-tune parameters to achieve excellent performance

Although random forest works well with a large number of predictors, in practice, the number of predictors may often need to be minimized for efficiency (i.e., variable selection needs to be performed). There are several methods available for performing variable selection with random forest classification. Many R packages provide random forest variable selection procedures such as *boruta* (Kursa & Rudnicki, 2010), *varSelRF* (Diaz-Uriarte & de Andres, 2005), *VSURF* (Genuer et al., 2010), *caret* (Kuhn, 2008), party (Hothorn et al., 2015), *randomForestSRC* (Ishwaran & Kogalur, 2021), *RRF* (Deng & Runger, 2013), *vita* (Janitza et al., 2018), *AUCRF* (Calle et al., 2011), and *fuzzyForest* (Conn et al., 2015). With the wide array of implementations of random forest available, the literature provides little guidance about which methods are preferable in terms of prediction error rate, parsimony, computation time, and area under the receiver operating curve (AUC) for different types of datasets. While some researchers have investigated which methods to use on simulated data (Cadenas et al., 2013; Degenhardt et al., 2019; Hapfelmeier & Ulm, 2013; Sanchez-Pinto et al., 2018), Speiser et al. has performed the most comprehensive comparison study to date (2019). This study recommends the use of VSURF for datasets with binary outcomes, and datasets with less than fifty predictors, whereas they recommend varSelRF or Boruta for datasets with many predictors because they are more

computationally efficient compared to other methods (Speiser et al., 2019). Based on these recommendations, the current study will implement Boruta because of its computationally efficiency.

The Present Study

Although a large amount of research has been done comparing various implementations of the Genetic Algorithm, LASSO, Elastic Net, SVM, and Random Forest, very little research has been done comparing these methods against each other in the context of variable selection for classification problems with labeled data (i.e. supervised learning). In addition, most research about these methodologies has been performed within the fields of computer science or biology. Very little research has been done looking at applications within psychology. As mentioned previously, the implementation of machine learning techniques for variable selection is relatively new to the field of psychology. In the medical community, researchers believe there is immense promise for improving accuracy of medical diagnoses with machine learning, and thus, it is important for the psychological community, especially the clinical psychology community, to begin to examine these techniques within the world of mental health diagnoses. If accurate predictive models can be built, it is possible that diagnoses would be able to be more individualized and operate less by the "gold standard" we see today. Given this limitation in the research, there is a need for a comprehensive comparison of variable selection procedures for classification problems in order to provide recommendations about which procedures are appropriate for different types of datasets.

Methods

For each of the datasets described below, variable selection was performed using the methods listed in Table 1. The models built by the methods in the first three rows will be used

solely for comparison, as they do not perform variable selection. All models were built and evaluated using cross-validation. A supervised learning classification model was built with each method using the default values of the available R package, as well as a model with tuned parameters. For each variable selection method and dataset, predictive accuracy was measured in two ways: the prediction error rate (defined as the proportion of incorrect predictions for the test data) and the area under the ROC curve (AUC). In addition, the number of variables used, the computation time, and the true positive rate (defined as the proportion of selected variables that were not noise) were recorded for each model. To obtain AUC estimates, we employed the R package *pROC* (Robin et al., 2011). R version 4.1.2 on a computer with a 2.5 GHz Eight-Core 11th Gen Intel® Core™ i7-11700 processor and 32.0 GB of RAM was used for analysis.

Simulation

Data Generation

Previous simulations examining classification have varied features such as the type of predictor (binary vs. continuous), the number of predictors, the sample size, and the correlation within blocks of variables (Brusco, 2014; Brusco et al., 2009; Jiang et al., 2021). Data for this study will be simulated using similar techniques in R. Altogether, five features will be varied to produce a wide variety of data sets for a robust application of these variable selection techniques.

Distribution of the Predictors. Predictors were generated either be continuous, with an underlying normal distribution ($\mu = 0,\ \sigma = 1$) with an added small amount of bias, or binary, with an underlying Bernoulli distribution with marginal probability $p_i, i = 1,2,3, \dots n$. Research indicates that ordinal variables such as likert-type scales can be treated as continuous variables, the continuous data generated here can generalize to likert-type data as well as true continuous data (Robitzsch, 2020).

Prevalence Rates. The probability of a diagnoses seen in the outcome variable was simulated $p = .20, .25, .30, .40$. This covers a wide range of prevalence rates seen in the DSM-5. The use of different prevalence rates allowed us to assess the performance of the techniques on varying differences in group sizes.

Number of Predictors. Data sets were simulated to be comprised of m = 10, 50, 100 predictor variables. A predictor variable is a variable measuring our outcome of interest. The use of a wide range of number of predictors allowed for comparisons on how the techniques perform with increasing dimensions.

Sample size. The data set was simulated to have n = 50, 200, 500 observations to examine the performance of techniques on varying sample sizes. Some previous literature indicates that a sample size of 200 is adequate to see strong performance (Marsh et al., 1998) while other researchers have indicated a much larger sample size is needed (Dolan, 1994).

Noise. As mentioned previously, predictor variables are measuring our outcome of interest. It is important to include variables that are not measuring this construct (i.e., noise) to evaluate the performance of the techniques on their ability to select only predictor variables. The data set was simulated to have p = 0, m, or $m/2$ noise variables.

After simulating data sets varying features of the data, the columns of the generated raw data will be randomly permuted, saving the index of the true variables, to examine the performance of variable identification from the techniques.

Applications

Alcohol Use Disorder

We have been given access to a pre-existing dataset concerning alcohol use disorder. The techniques studied in this paper were applied to this dataset to evaluate their performance. The

dataset consists of 909 undergraduate students ($\bar{x}_{age} = 18.64$) enrolled in introduction to psychology courses at the University of Missouri. Participants completed the entire survey battery online and course credit was awarded upon completion. Most participants were female (56%) and White (91%).

The dataset consists of 87 AUD symptom indicators for the 11 DSM-5 criteria derived or adapted by Boness et al. (2019) from 9 well-validated diagnostic interviews and self-report scales. Items were rated on a five-point Likert scale (0 = Never/Not in the Past 12 Months, 4 = Yes, 4+ Times). Participants reported on age of first drinking experience, age of first heavy drinking experience, consumption, hangover, blackout, self- and other-identified problem use, treatment use, legal problems, cannabis use frequency, and three composites of functioning: general health, mental health, and physical functioning. For more information on the dataset, see Boness et al. (2019). In addition, a variable is included containing a diagnosis based on the DSM-5 criteria. This variable was used for the pre-determined classification against which to check our models' predicted classifications.

Misophonia

We have been given access to use a pre-existing dataset concerning misophonia. The techniques studied in this paper was applied to this dataset to evaluate their performance. The dataset consists of 343 undergraduate students ($\bar{x}_{age} = 18.96$) recruited using a secure online research participation system at the University of Oklahoma (OU) in Norman, Oklahoma and all data were collected anonymously via Qualtrics™. Those who completed the survey received 1 hour worth of credit for participation to count towards class credit for research experience. All study procedures were approved by the affiliated university's institutional review board (i.e., OU

IRB). Participants were predominately female (69.7%) and white (76.7%) Self-report current diagnoses were also collected.

The dataset contains questions addressing an array of symptoms characteristic of misophonia, referred to as the Spectrum Characteristic Survey (SCS). The SCS was comprised of a demographics section and 6 clinical measures designed to address various aspects of misophonia, related symptoms, and comorbidities. For the purpose of this study, we used the first two clinical measures from this dataset. Each clinical measure will be treated as its own dataset.

Misophonia Questionnaire (MQ). The MQ contains 19 self-report questions indexing misophonia symptoms (M. S. Wu et al., 2014). Participants were asked to rate their sensitivity to auditory triggers on a scale from 0 ("not at all true") to 4 ("always true"). Questions assessing symptom severity were used to split participants into groups reflecting clinically or non-clinically relevant levels of misophonia symptoms. These groups were used as our pre-determined classifications against which to compare our models' predicted classifications.

S-Five (2018). This dataset contains a version of the S-Five published in 2018 (Silia & Chloe, 2018). The S-Five contains 86 total items assessing two aspects of misophonia: triggers and statements regarding behavior associated with misophonic triggers. Participants were asked to rate their typical reaction to 36 trigger items on a 5-point likert scale from 0 ("does not bother me") to 4 ("so unbearable that I need to plan beforehand to avoid it"). Participants were asked to read each of 50 statements and indicate their level of agreement on a 6-point likert scale 0 ("very strongly disagree") to 5 ("very strongly agree"). We used the same groups developed via the *MQ* as our pre-determined classifications with S-Five predictors against which to compare our models' predicted classifications.

Results

Simulation Study

The three methods evaluated which do not perform variable selection were LR, RF, and SVM. These three methods create models using the full set of items and are only included in the analyses for comparison purposes. Specifically, they are included so the predictive power (prediction error rates and AUC) of models created with a subset of variables can be compared to the predictive power of models creating using all variables. GA, LASSO, Elastic Net, Elastic SVM, and Boruta are all methods which do perform variable selection.

Results Across All Simulated Datasets

The distributions of the prediction error rate, computation time, AUC, and proportion of items included within the reduced models were compared in Table 2 and Figures 4 and 5. The mean prediction error rates ranged from 0.080 to 0.357 across the variable selection methods and were normally distributed with the exception of GA which is positively skewed. All variable selection methods produced models with lower prediction error rates than the model built by LR, but the only method with a lower average prediction error rate than RF and SVM was the GA. The top three variable selection methods with the lowest prediction error rates were GA, Elastic SVM, and Boruta. No models had very large prediction error rates, indicating that all methods predicted outcomes fairly well.

Of the variable selection methods, which all took longer than the methods using the full set of variables, LASSO Tuned had the lowest computation time (0.405 s) and Elastic SVM had the highest (37.203 s). These methods also had the lowest and highest variability in computation time respectively. Note the large margin in computation time between Elastic SVM and the next highest computation time from EN Tuned (Table 2).

Values of AUC for models created using the full variable set (LR, RF, SVM) ranged from 0.588 to 0.623. Both GA and Boruta produced models with higher AUCs, 0.884 and 0.627 respectively. LASSO Tuned and EN Tuned produced models with AUCs that were higher than LR, but lower than RF and SVM. LASSO and Elastic Net created models with AUCs of 0.523 and 0.522 respectively, which are lower than the models using the full variable set.

The distributions of the proportions of variables included within the models varied widely across method. LASSO Tuned and EN Tuned had negative skewed distributions, GA's distribution was fairly normal, and all other models had positively skewed distributions. Elastic SVM and GA selected between 50 and 60% of the variables. EN Tuned selected the most variables, with an average proportion of 94%, while its non-tuned partner, Elastic Net, only selected an average of 15% of the variables. LASSO selected the least, a mere 8% on average.

It should be noted that not all methods were able to produce models utilizing at least one variable on all datasets. Table 2 contains the number of models (*N*) created by each of the methods. Although most methods provided models for at least 95% of simulated datasets, GA, Elastic SVM, and EN Tuned were the only methods to create models for all simulated datasets. LASSO and Elastic Net were only able to produce models using at least one variable for about 50% of simulated datasets.

The median AUC was plotted by median computation time, proportion of variables selected, and prediction error rate (Figure 6). Elastic SVM had significantly higher computation times than other methods and yet, did not create models with high AUCs relative to other methods. LASSO Tuned and EN Tuned selected a higher proportion of variables than other methods but had comparable median AUCs to methods that used far fewer variables. GA produced models with high AUC and low prediction error rates using, on average, about 50% of

the original number of variables, in a relatively small amount of time. There are clear clusters of methods in terms of comparing AUC and computation time and AUC and prediction error rates. We see that LASSO, LASSO Tuned, Elastic Net, EN Tuned, Boruta, and Elastic SVM all perform very similarly on all these metrics. One should note, however, that these methods do not cluster quite as tightly when comparing AUC and proportion of variables selected.

Given that some of the methods did not create models containing at least one variable for all simulated datasets, performance of the methods was also investigated using only simulated datasets for which all methods created such models. Table 3 displays the distribution of prediction error rates, AUC, proportion of variables selected by each method, proportion of the variables selected that are not noise variables (TPR), and computation times of the models created on these simulated datasets specifically. Of the total 10,800 simulated datasets, only 4,927 datasets were used in this analysis.

Though many datasets were not used in this analysis, we see the same general relative performance of the methods regarding prediction error rate, AUC, computation time, and proportion of variables selected as we see in Table 2 (i.e., the analysis including all 10,800 simulated datasets). Given this similarity of results, all other analyses were done using all 10,800 simulated datasets.

Comparing methods grouped by characteristics of the datasets

Comparison of methods with binary vs. continuous predictors. These results can be found in Tables 4 and 5 and Figures 7 and 8. Half of our simulated datasets had binary predictors while the other half had continuous predictors. Although all methods using the full measure (e.g., LR, RF, and SVM) produced slightly higher AUCs for datasets containing binary predictors, this pattern did not hold across all variable selection methods. LASSO and Elastic Net produced

models with slightly lower average AUCs with datasets using binary predictors as compared to continuous predictors. GA also produced models with lower AUCs when binary predictors were present, but we see a significant rather than a slight decrease in the AUC. Boruta, LASSO Tuned, and EN Tuned followed the pattern set in our full models, creating models with higher AUCs with binary predictors than continuous predictors. Prediction error rates of models created by all methods except GA seem to be rather unaffected by the predictor type. We do, however, see that the average prediction error rate of GA models using binary predictors is significantly higher than GA models using continuous predictors.

In models built with all methods except GA and LASSO, we see that methods selected fewer variables from datasets with binary predictors than datasets with continuous predictors. The number of variables selected by LASSO does not seem to vary across predictor type, while GA selects slightly more predictors when they are binary than when they are continuous. No significant differences were seen regarding computation time or TPR.

Comparison of methods across prevalence rates. Results for these comparisons can be found in Tables 6 through 10 as well as Figures 9 and 10. No real differences in AUC were found across prevalence in models created by LASSO, LASSO Tuned, Elastic Net, or EN Tuned. We do see a large increase in AUC when comparing Elastic SVM models with a prevalence rate of 0.4 to other Elastic SVM models. The other differences we see regarding Elastic SVM occur purely in the spread of the AUC of the models rather than their average AUC. We see the opposite pattern occurring with GA models in that the AUC of the model decreases as the prevalence rate increases. It appears that this decrease occurs at about the same rate across all conditions. Despite this decrease, it is important to remember that GA outperformed all other

methods regarding AUC. We see slight variation in Boruta with small increases in AUC occurring as the prevalence rate increases.

All methods follow the same pattern regarding prediction error rate. As the prevalence rate increases, so does the prediction error rate. Computation time does not appear to be affected by prevalence rates, except in the case of Elastic SVM where we see a very small direct relationship.

The proportion of variables selected seems to have an indirect relationship with prevalence rate in models built with LASSO, Elastic Net and Boruta. LASSO Tuned, and Elastic SVM, however, seem to select more variables with higher prevalence rates. Models built using the GA and EN Tuned do not seem to differ in the proportion of variables selected across prevalence rates. No differences were seen regarding TPR as compared across prevalence rates.

Comparison of methods across sample sizes. Results from these analyses can be found in Tables 11, 12, and 13 and Figures 11 and 12. All methods except for GA, Boruta, and LASSO Tuned produced models with comparable results across all sample sizes regarding AUCs. GA, however, saw significant decreases in AUCs with increases in sample sizes. We see a slight decrease in average AUCs from a sample size of 50 to 200, and then a large decrease in average AUC with a sample size of 500. It is important to note that we also see a larger spread in AUCs at larger sample sizes, which may account for some of this decrease. With models created using Boruta and LASSO Tuned, we see small increases in AUCs as the sample size increases (the increases in Boruta models were slightly larger than those in LASSO Tuned models).

We see slightly more deviation across sample sizes when we look at prediction error rates. LASSO and Elastic Net produce models with lower prediction error rates as the sample size increases, while their tuned counterparts, LASSO Tuned and EN Tuned (as well as Elastic

SVM) have the highest prediction error rates with sample sizes of 200. GA produces models that have higher prediction error rates as the sample size increases. Boruta models see significant decreases in prediction error rates as the sample size increases.

We see that all methods select a higher proportion of variables as the sample size of the dataset increases. LASSO Tuned sees the largest differences in the proportion of variables across sample sizes, followed by Boruta. Models created with GA seem to be fairly consistent in the numbers of variables they use across sample size.

All methods have higher computation times as the sample size increases, as is to be expected. The only methods that appear to create models sensitive to sample size when examining the TPR are LASSO and Elastic Net.

Comparison of methods across ratios of predictors to observations. Results for these analyses can be found in Tables 14, 15, and 16 and Figures 13 and 14. Analyses were done to examine the performance of methods in $p \gg n$ conditions versus other ratios. Regarding AUC values, differences were only seen in models built using GA and Boruta. GA models had a significantly lower AUC in models where the data were $p \gg n$. We see a significantly larger range of AUC values for $p \gg n$ GA models than other GA models which may accout for this decrease. Boruta models had slightly higher average AUCs in models where the data were $p \gg n$.

We see the same outliers in terms of prediction error rates. Only GA and Boruta models seem to be influenced by the ratio of predictors to observations. GA models had higher prediction error rates when the data were $p \gg n$, while Boruta models had lower prediction error rates with $p \gg n$ datasets.

Computation time appears to be affected by the ratio of predictor to observations in all models. All methods except GA and Elastic SVM took longer to create their models when the data were $p \gg n$. GA appears to vary more widely in the amount of time it takes to create models with $p \gg n$ data, while Elastic SVM creates its models much faster with data of this form.

All methods selected a higher proportion of variables when the data were $p \gg n$. We see the largest difference in models created by LASSO Tuned, followed by Boruta models. The smallest difference occurs in models created by GA. In addition, LASSO Tuned sees significant increases in TPR when the data is $p \gg n$.

Comparing methods grouped by characteristics of the method

The methods used in this comparison study can be grouped into three general categories: regression-based methods, tree-based methods, and metaheuristics. LASSO, LASSO Tuned, Elastic Net, Elastic Net Tuned, and SVM are all regression-based methods. Boruta is a tree-based method, and GA is a metaheuristic. Analyses were run comparing the performance of methods averaging within these categories. Results from these analyses can be found in Table 17 and Figure 15.

We see that the metaheuristic algorithm, GA, results in increased average AUCs and decreased prediction error rates when comparing to other methods, followed by the tree-based method, Boruta, and then all regression-based methods. Note that prediction error rates for tree-based methods are only slightly lower than those of regression-based methods.

In accordance with previous literature, metaheuristic and tree-based methods have larger computation times than regression-based methods. We also see that tree-based methods select the lowest proportion of variables while metaheuristic algorithms select the highest proportion.

Regression based methods, however, have the largest range regarding proportion of variable selected. No real differences are seen regarding TPR when comparing across method type.

Alcohol Use Disorder

Results from the application of each method to the alcohol use disorder dataset can be found in Table 18. Prediction error rates ranged from 0 to 27.7% across all the variable selection methods. The three methods with the lowest prediction error rates were GA, GA tuned, and LASSO tuned with prediction error rates of 0, 2.9%, and 4.7% respectively. The models with the largest prediction error rates were Elastic SVM and ESVM tuned. This indicates that these models did not predict a diagnosis of alcohol use disorder well.

It is interesting to note that GA created models with lower prediction error rates than all base models using the entire set of predictors. This indicates that there are not only unnecessary variables included in the data set, but that these variables are detrimental for prediction.

Most methods selected between 50 and 70 of the original 86 variables. LASSO offered the lowest number of variables (21), followed by bTuned (51), GA tuned (52), and GA (57). Elastic SVM, EN tuned, and ESVM tuned selected the most variables. It is important to note that even with a large selection of variables, Elastic SVM and ESVM tuned created models with the highest prediction error rates. This is surprising given that, out of the models using all 86 variables, SVM created the model with the lowest prediction error rate.

LASSO tuned had the lowest computation time followed by Elastic Net, LASSO, and EN tuned. Methods with the greatest computation times (taking over an hour) were ESVM tuned and bTuned. Elastic SVM took only 130 seconds. GA tuned also had a high computation time, taking around 45 minutes, but GA took only 21 seconds to run. Other methods had comparable computation times ranging between 1 and 7 seconds.

Values of AUC for most of the models ranged from 1 to 0.86 with two outliers of Elastic SVM and ESVM tuned having AUCs of 0.5. This indicates that most of the models offered good fit. GA and GA tuned produced the highest AUCs of 1 and 0.968 respectively.

In terms of which variables were selected by each method, 12 variables were selected by all models (Figure 17), and 18 more variables were selected by all but one method. In addition, there were two variables that were not selected in any of the models, indicating that they are not strong predictors of alcohol use disorder (Figure 17).

Misophonia

Misophonia Questionnaire. Results from the application of each method to the misophonia questionnaire dataset can be found in Table 19. Prediction error rates ranged from 0 to 17.2% across all the variable selection methods. The three methods with the lowest prediction error rates were GA, GA tuned, and Elastic Net with prediction error rates of 0%, 0%, and 12.5% respectively. All other models had a prediction error rate of 14.1%.

Two methods, LASSO tuned and EN Tuned selected all 19 variables to create their models while LASSO selected 0 variables. Other methods selected between 4 and 14 of the original 19 variables. Of the models that contained at least one variable, Elastic SVM selected the fewest number of variables (4), with bTuned, ESVM tuned, and Boruta following close behind with 5, 6, and 7 selected variables respectively. It is important to remind the reader that with a wide range of selected variables, seven of our methods produced models with the same prediction error rate of 14.1%.

LASSO tuned had the lowest computation time followed by Elastic Net, LASSO, and EN tuned. Methods with the greatest computation times (taking at least 4 minutes) were ESVM tuned and bTuned. GA tuned also had a high computation time, taking just over a minute, but

GA took only half a second to run. Other methods had comparable computation times ranging between 1 and 4 seconds.

Values of AUC for most of models ranged from 0.686 to 1 with four outliers of LASSO, Elastic Net, Elastic SVM and ESVM tuned having AUCs of, or near in the case of Elastic Net, 0.500. Note that the model created from LASSO contained zero variables. These AUCs indicates that most of the models offered good fit. GA tuned and GA produced the highest AUCs of 1.

In terms of which variables were selected by each method, one variable was selected by all nine of the ten methods that selected at least one variable, two more by eight methods, one more by seven, and four more by six methods (Figure 18). All variables were selected by at least 3 methods.

S-Five (2018). Results from the application of each method to the S-Five dataset can be found in Table 20. Prediction error rates ranged from 0 to 36.6% across all the variable selection methods. The four methods with the lowest prediction error rates were GA, GA tuned, LASSO and Elastic Net with prediction error rates of 0%, 0%, 11.9%, and 11.9% respectively. The models with the largest prediction error rates were LASSO tuned and EN tuned with predictive error rates of 36.6% and 35.6% respectively. This indicates that these two models did not predict misophonia well.

EN tuned selected the most variables (85). One should note that LASSO did not select a single variable. Other methods selected between 6 and 68 of the original 86 variables. The lowest number of variables in models that selected at least one variable were selected by bTuned (6), followed by Boruta (14), Elastic SVM (41), GA (56) and GA tuned (56).

LASSO tuned had the lowest computation time followed by LASSO and Elastic Net. The method with the greatest computation times (taking over an hour) was ESVM tuned. Taking just

over 20 minutes was bTuned, and the method with the next highest computation time was GA tuned, taking about 11 minutes. GA, however, only took 3 seconds. Elastic SVM took 22 seconds to run. Other methods had comparable computation times ranging between 1 and 2 seconds.

Values of AUC for most models ranged from 0.432 to 0.500 with two outliers of GA and GAtuned having AUCs of 1. This indicates that most of the models did not offer a good fit. Only GA and GA tuned produced perfect AUCs of 1. It is important to note that all models created with all 86 of the original variables had AUCs ranging from 0.456 to 0.5. Thus, in comparison, the models created with variable selection methods did not see significant decreases in AUC from the base models. In fact, the models created by genetic algorithms saw increases in AUCs.

In terms of which variables were selected by each method, three variables were selected by all eight of the ten methods that selected at least one variable, four more by seven methods, and fifteen more by six methods. One variable was selected by only one method, and an additional seven were only selected by two methods (Figure 18).

Discussion

Through a large-scale Monte Carlo simulation and application to three real datasets, this paper provided a comprehensive comparison of supervised machine learning variable selection methods for classification used Gin the psychological sciences. Performance measures did not converge upon a single best method; as such, researchers should guide their method selection based on what measure of performance they deem most important. When prioritizing predictive power over all other considerations, GA should be implemented. When prioritizing both predictive power and number of variables used, my recommendation is for Boruta. If a researcher's priority is computation time, LASSO or Elastic Net is recommended with caution.

First, models built using LASSO and Elastic Net had significantly lower predictive power (low AUC and high prediction error rates) than other methods and second, these methods were able to produce models with at least one variable for the least number of datasets, and thus may not work for your dataset.

It is worth discussing possible explanations for why LASSO and Elastic Net eliminated all variables for so many simulated datasets. The logical interpretation of these results is the assumption that no linear combination of any subset of the regressors may be useful for predicting the outcomes. Given the results of our other methods, however, this does not appear to be a correct interpretation. Rather, it is possible that the selected λ_1 and/or λ_2 obtained during the cross-validation approach were too high. This explanation is possible, given that LASSO Tuned created models for all simulated datasets and EN Tuned created models for all but one simulated dataset. Given the strict boundaries implemented in our grid search, λ_1 and/or λ_2 were not allowed to reach sizes large enough to warrant eliminating all variables from the model.

Researchers who are most interested in strong predictive power (e.g., high AUC and low prediction error rates), and can afford to implement a less computationally efficient method which selects a higher number of variables, GA is recommended. Researchers most interested in a method that selects the fewest number of variables should consider implementing LASSO or Elastic Net. No significant improvements were seen in predictive power when implementing a grid search to tune LASSO and Elastic Net (LASSO Tuned and EN Tuned, respectively) as compared to the methods tuned using the cross-validation method built into the *glmnet* package in R. If a researcher's priority is predictive power, I recommend use of the built-in cross validation approach because it does not rely on the researcher to create a grid of possible parameters. However, in some cases, tuning with a grid search was seen to be faster, so for those

who are willing to sacrifice an amount of predictive power for more efficient computation times, a grid search should be considered. Researchers aiming to maintain high predictive accuracy with a lower number of variables selected, Boruta should be considered for it has considerable predictive power while significantly decreasing the required number of variables. For these strengths, Boruta does, however, still have high computation times. I advise against the use of Elastic SVM, for it did not appear to perform significantly better on any performance measure than other methods and had the highest average computation time (over 11 times that of Boruta).

The structure of the data should also be considered when selecting the method. Although GA outperforms all methods in terms of predictive power, in data where $p \gg n$, GA did not perform as well as with other ratios of predictors to observations. Boruta, however, performed slightly better in these conditions. In addition, there were differences seen in performance regarding the type of predictor in the dataset. With binary predictors, the possible predictive power (i.e., the AUC and prediction error rates of models built using all variables) was better with binary predictors than continuous predictors, we saw that LASSO, Elastic Net, EN Tuned and GA performed worse on datasets with binary predictors. Boruta, however, seemed to perform better with binary predictors. Thus, if one has a dataset using only binary predictors, Boruta should be considered.

All methods seemed to have higher AUCs and lower prediction error rates when the sample size was larger. All methods appeared to have higher prediction error rates when the prevalence rate was higher (indicating more even group sizes in the data). Prediction error rates appear to be least affected by prevalence rate through use of GA, while Boruta produces models where the AUC is least affected by prevalence rates. In fact, the average AUC is almost identical in models created with Boruta on datasets with prevalence rates of .25, .30, and .40.

Overall, this paper illustrates that metaheuristic approaches perform best on all datasets while tree-based methods perform second best and regression-based methods perform the worst. One should consider that the use of Elastic SVM did decrease average performance metrics in terms of regression-based methods, and without its inclusion, average performance would be slightly better, but they would still not outperform metaheuristics. This is a good indication that the future of variable selection research lies in the development of new and improvement of current metaheuristics. There are a few current obstacles metaheuristics face in terms of their use by behavioral scientists. For one they do have significantly higher computation times than other methods, and thus it can be hard to convince researchers that what they give up in computation time is worth what they gain in model performance – I hope this paper will help to illustrate this point. Two, the use of metaheuristics requires a level of understanding of computer science and machine learning which are two areas many behavioral researchers do not have experience in. With the increased popularity of R, however, this gap in knowledge appears to be decreasing. In addition, we are seeing more and more R packages being introduced that use metaheuristics for variable selection problems. One popular one is the *ShortForm* package by Raborn and Leite (2020). Metaheuristics are still not available in other popular software like SPSS or SAS which are commonly used by behavioral researchers. Thirdly, metaheuristics are often seen as somewhat of a "black box" approach which makes them hard for users to understand. However, with the continued increase in their use and popularity, I have confidence that these obstacles may decrease with time.

Limitations

This study had some limitations worth addressing such as the size of the datasets both regarding the number of observations and number of predictors. To better understand how the

methods assessed in this study perform with data in larger dimensions, subsequent studies should consider expanding the dataset size and number of predictors. The simulation was constrained to these conditions due to computational expense – larger datasets would have taken several hours to complete. Despite these constraints, 108 unique conditions were analyzed, so results can guide researcher's toward implementing the most appropriate variable selection method for their given dataset. Another aspect that should be examined in future research is how these methods perform with missing data, as the current study did not address this question.

There are several avenues of future work stemming from this paper. For example, one could examine the performance of each of these methods with missing data – especially given that Boruta treats missing values as a separate category when splitting its trees. One could also vary the amount of missing data and the methods through which missing data is handled (e.g., leaving it as is, use of multiple imputation, random forest imputation, k-nearest neighbor imputation, etc.). Additionally, one could implement this study using more than two classifications. This still has implementations in terms of diagnosis – for example, alcohol use disorder can be diagnosed at different levels: mild, moderate, or severe. Another future study could repeat analyses of these methods in higher dimensions, perhaps with the inclusion of parallel computing to speed up run time. Lastly, another potential expansion of this research would be through the inclusion of additional tree-based methods (see Speiser et al., 2019 for a list of available random forest variable selection methods in R) and metaheuristic methods such as Tabu search, simulated annealing, or particle swarm algorithms.

The primary contribution of this study is the ability to assess different supervised machine learning variable selection methods used in the psychological sciences. Specifically, this study provided computation times for models, which addressed an important gap often

ignored within the field. In addition, this paper illustrates an area in which machine learning is beneficial to behavioral scientists. I hope that this paper encourages researchers to consider the use of machine learning methods in variable selection problems and how the field behavioral science could benefit from their use.

References

Algamal, Z. Y., & Lee, M. H. (2015a). Applying penalized binary logistic regression with correlation based elastic net for variables selection. *Journal of Modern Applied Statistical Methods*, *14*(1), 168–179. https://doi.org/10.22237/jmasm/1430453640

Algamal, Z. Y., & Lee, M. H. (2015b). Regularized logistic regression with adjusted adaptive elastic net for gene selection in high dimensional cancer classification. *Computers in Biology and Medicine*, *67*, 136–145. https://doi.org/10.1016/j.compbiomed.2015.10.008

Anastasi, A., & Urbina, S. (1997). *Psychological Testing* (7th ed.). Prentice Hall.

Arditte, K. A., Çek, D., Shaw, A. M., & Timpano, K. R. (2016). The importance of assessing clinical phenomena in Mechanical Turk Research. *Psychological Assessment*, *28*, 684–691. https://doi.org/10.1037/pas0000217

Becker, N., Toedt, G., Lichter, P., & Benner, A. (2011). Elastic SCAD as a novel penalization method for SVM classification tasks in high-dimensional data. *BMC Bioinformatics*, *12*. https://doi.org/10.1186/1471-2105-12-138

Becker, N., Werft, W., Toedt, G., Lichter, P., & Benner, A. (2009). Data and text mining penalizedSVM: a R-package for feature selection SVM classification. *BIOINFORMATICS APPLICATIONS NOTE*, *25*(13), 1711–1712. https://doi.org/10.1093/bioinformatics/btp286

Bengio, Y., Delalleau, O., & Simard, C. (2010). Decision trees do not generalize to new variations. *Computational Intelligence*, *26*(4), 449–467. https://doi.org/10.1111/j.1467-8640.2010.00366.x

Boness, C. L., Lane, S. P., & Sher, K. J. (2019). Not all alcohol use disorder criteria are equally severe: Toward severity grading of individual criteria in college drinkers. *Psychology of*

Addictive Behaviors : Journal of the Society of Psychologists in Addictive Behaviors, *33*(1), 35–49. https://doi.org/10.1037/ADB0000443

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A Training Algorithm for Optimal Margin Classifiers. *COLT '92 - Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 144–152.

Bourgon, R., Gentleman, R., & Huber, W. (2010). Independent filtering increases detection power for high-throughput experiments. *Proceedings of the National Academy of Sciences of the United States of America*, *107*(21), 9546–9551. https://doi.org/10.1073/pnas.0914005107

Brieman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (2017). *Classification and Regression Trees*. CRC Press.

Brusco, M. J. (2014). A comparison of simulated annealing algorithms for variable selection in principal component analysis and discriminant analysis. *Computational Statistics & Data Analysis*, *77*, 38–53. https://doi.org/10.1016/j.csda.2014.03.001

Brusco, M. J., Singh, R., & Steinley, D. (2009). Variable neighborhood search heuristics for selecting a subset of variables in principal component analysis. *Psychometrika*, *74*(4), 705–726. https://doi.org/10.1007/s11336-009-9130-3

Buhrmester, M., Kwang, T., & Gosling, S. D. (2011). Amazon's Mechanical Turk: A New Source of Inexpensive, Yet High-Quality, Data? *Https://Doi.Org/10.1177/1745691610393980*, *6*(1), 3–5. https://doi.org/10.1177/1745691610393980

Cadenas, J. M., Garrido, C. M., & MartíNez, R. (2013). Feature subset selection Filter-Wrapper based on low quality data. *Expert Systems with Applications: An International Journal*, *40*(16), 6241–6252. https://doi.org/10.1016/J.ESWA.2013.05.051

Cadima, J., Cerdeira, J. O., & Minhoto, M. (2004). Computational aspects of algorithms for variable selection in the context of principal components. *Computational Statistics and Data Analysis*, *47*(2 SPEC. ISS.), 225–236. https://doi.org/10.1016/j.csda.2003.11.001

Calle, M. L., Urrea, V., Boulesteix, A. L., & Malats, N. (2011). AUC-RF: A New Strategy for Genomic Profiling with Random Forest. *Human Heredity*, *72*, 121–132. https://doi.org/10.1159/000330778

Cartledge, C. (2016). *How Many Vs are there in Big Data?*

Chapman, B. P., Weiss, A., & Duberstein, P. (2016). Statistical Learning Theory for High Dimensional Prediction: Application to Criterion-Keyed Scale Development. *Psychological Methods*, *21*(4), 603–620. https://doi.org/10.1037/met0000088

Conn, D., Ngun, T., Li, G., & Ramirez, C. (2015). Fuzzy Forests: Extending Random Forests for Correlated, High-Dimensional Data. *UCLA EScholarship Research Report*, 1–6.

Cowgill, 1vi C, Harvey, R. J., & Watson, L. T. (1999). A Genetic Algorithm Approach to Cluster Analysis. In *Computers and Mathematics with Applications* (Vol. 37).

Cramer, J. S. (2005). The Origins of Logistic Regression. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.360300

Crotes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, *20*, 273–297. https://doi.org/10.1109/64.163674

Crump, M. J. C., McDonnell, J. v., & Gureckis, T. M. (2013). Evaluating Amazon's Mechanical Turk as a Tool for Experimental Behavioral Research. *PLOS ONE*, *8*(3), e57410. https://doi.org/10.1371/JOURNAL.PONE.0057410

Curran, P. J., & Hussong, A. M. (2009). Integrative Data Analysis: The Simultaneous Analysis of Multiple Data Sets. *Psychological Methods*, *14*(2), 81–100. https://doi.org/10.1037/A0015914

Daubechies, I., Defrise, M., & Mol, C. de. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, *57*(11), 1413–1457. https://doi.org/10.1002/CPA.20042

de Meneses-Gaya, C. (2011). Alcohol Use Disorders Identification Test (AUDIT): An updated systematic review of psychometric properties. *Psychology & Neuroscience*, *2*(1), 83. https://doi.org/10.3922/J.PSNS.2009.1.12

Degenhardt, F., Seifert, S., & Szymczak, S. (2019). Evaluation of variable selection methods for random forests and omics data sets. *Briefings in Bioinformatics*, *20*(2), 492–503. https://doi.org/10.1093/bib/bbx124

Deng, H., & Runger, G. (2013). *Gene Selection With Guided Regularized Random Forest*.

Derksen, S., & Keselman, H. J. (1992). Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. *British Journal of Mathematical and Statistical Psychology*, *45*(2), 265–282. https://doi.org/10.1111/J.2044-8317.1992.TB00992.X

Descloux, P., & Sardy, S. (2021). Model Selection With Lasso-Zero: Adding Straw to the Haystack to Better Find Needles. *Https://Doi.Org/10.1080/10618600.2020.1869026*. https://doi.org/10.1080/10618600.2020.1869026

Diaz-Uriarte, R., & de Andres, S. A. (2005). *Variable selection from random forests: application to gene expression data*. 1–11. http://arxiv.org/abs/q-bio/0503025

Dolan, C. v. (1994). Factor analysis of variables with 2 , 3 , 5 and 7 response categories : A comparison of categorical variable estimators using simulated data. *British Journal of Mathematical and Statistical Psychology*, *47*, 309–326.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, *32*(2), 407–499. https://doi.org/10.1214/009053604000000067

Eisenbarth, H., Lilienfeld, S. O., & Yarkoni, T. (2015a). Using a genetic algorithm to abbreviate the psychopathic personality inventory-revised (PPI-R). *Psychological Assessment*, *27*(1), 194–202. https://doi.org/10.1037/pas0000032

Eisenbarth, H., Lilienfeld, S. O., & Yarkoni, T. (2015b). Using a genetic algorithm to abbreviate the psychopathic personality inventory-revised (PPI-R). *Psychological Assessment*, *27*(1), 194–202. https://doi.org/10.1037/pas0000032

Fan, J., & Li, R. (2011). Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties. *Https://Doi.Org/10.1198/016214501753382273*, *96*(456), 1348–1360. https://doi.org/10.1198/016214501753382273

Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, *15*, 3133–3181. https://doi.org/10.1117/1.JRS.11.015020

Fonti, V. (2017). Feature Selection using LASSO. *VU Amsterdam*, 1–26.

Friedman, J., Hastie, T., Höfling, H., & Tibshirani, R. (2007). Pathwise coordinate optimization. *Https://Doi.Org/10.1214/07-AOAS131*, *1*(2), 302–332. https://doi.org/10.1214/07-AOAS131

Friedman, J., Hastie, T., & Tibshirani, R. (2010a). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, *33*(1), 1–22.

Friedman, J., Hastie, T., & Tibshirani, R. (2010b). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, *33*(1), 1–22. https://www.jstatsoft.org/v33/i01/

Fu, W. J. (n.d.). *Penalized Regressions: The Bridge Versus the Lasso*.

Fung, G. M., & Mangasarian, O. L. (2004). A feature selection Newton method for support vector machine classification. *Computational Optimization and Applications*, *28*(2), 185–202. https://doi.org/10.1023/B:COAP.0000026884.66338.DF

Genkin, A., Lewis, D. D., & Madigan, D. (2012). Large-Scale Bayesian Logistic Regression for Text Categorization. *Http://Dx.Doi.Org/10.1198/004017007000000245*, *49*(3), 291–304. https://doi.org/10.1198/004017007000000245

Genuer, R., Poggi, J.-M., & Tuleau-Malot, C. (2010). Variable selection using random forests. *Pattern Recognition Letters*, *31*(14), 2225–2236. https://doi.org/10.1016/j.patrec.2010.03.014

Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning. *Choice Reviews Online*, *27*(02), 27-0936-27–0936. https://doi.org/10.5860/choice.27-0936

Gosling, S. D., Vazire, S., Srivastava, S., & John, O. P. (2004). Should We Trust Web-Based Studies? A Comparative Analysis of Six Preconceptions About Internet Questionnaires. *American Psychologist*, *59*(2), 93–104. https://doi.org/10.1037/0003-066X.59.2.93

Groenen, P. J. F., Mathar, R., & Trejos, J. (2000). *Global Optimization Methods for Multidimensional Scaling Applied to Mobile Communications* (pp. 459–469). https://doi.org/10.1007/978-3-642-58250-9_37

Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, *3*, 1157–1182.

Hall, M. A. (1999). *Correlation-based Feature Selection for Machine Learning* (Issue April).

Hapfelmeier, A., & Ulm, K. (2013). A new variable selection approach using Random Forests. *Computational Statistics and Data Analysis*, *60*(1), 50–69. https://doi.org/10.1016/j.csda.2012.09.020

Hesterberg, T., Choi, N. H., Meier, L., & Fraley, C. (2008). Least angle and ℓ 1 penalized regression: A review * †. *Statistics Surveys*, *2*, 61–93. https://doi.org/10.1214/08-SS035

Hindi, M. M., & Yampolskiy, R. v. (2012). Genetic Algorithm Applied to the Graph Coloring Problem. *Midwest Artificial Intelligence and Cognitive Science Conference*, 60–66.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, *12*(1), 55–67.

Hofer, S. M., & Piccinin, A. M. (2009). Integrative data analysis through coordination of measurement and analysis protocol across independent longitudinal studies. *Psychological Methods*, *14*(2), 150–164. https://doi.org/10.1037/A0015566

Holland, J. H. (2019). *Adaptation in Natural and Artificial Systems.* The MIT Press.

Hothorn, T., Hornik, K., Strobl, C., & Zeileis, A. (2015). party: A Laboratory for Recursive Partytioning. *R Package Version 0.9-0, URL Http://CRAN. R-Project. Org*, *1994*, 37.

Ipeirotis, P. G., Provost, F., & Wang, J. (2010). *Quality Management on Amazon Mechanical Turk*.

Ishwaran, H., & Kogalur, U. B. (2021). *Package ‘ randomForestSRC .’*

Janitza, S., Celik, E., & Boulesteix, A. L. (2018). A computationally fast variable importance test for random forests for high-dimensional data. *Advances in Data Analysis and Classification*, *12*(4), 885–915. https://doi.org/10.1007/s11634-016-0276-4

Jiang, W., Song, S., Hou, L., & Zhao, H. (2021). A Set of Efficient Methods to Generate High-Dimensional Binary Data With Specified Correlation Structures. *American Statistician*, *75*(3), 310–322. https://doi.org/10.1080/00031305.2020.1816213

Johnson, M., & Sinharay, S. (2011). Remarks from the new editors. *Journal of Educational and Behavioral Statistics*, *36*(1), 3–5. https://doi.org/10.3102/1076998610387267

Kodratoff, Y. (2014). *Introduction to machine learning*. Morgan Kaufmann.

Kok, B. C., Choi, J. S., Oh, H., & Choi, J. Y. (2021). Sparse Extended Redundancy Analysis: Variable Selection via the Exclusive LASSO. *Multivariate Behavioral Research*, *56*(3), 426–446. https://doi.org/10.1080/00273171.2019.1694477

Krishnapuram, B., Carin, L., Figueiredo, M., & Hartemink, A. (2005). Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(6), 957–968. https://doi.org/10.1109/TPAMI.2005.127

Kuhn, M. (2008). Building predictive models in R using the caret package. *Journal of Statistical Software*, *28*(5), 1–26. https://doi.org/10.18637/jss.v028.i05

Kursa, M. B., & Rudnicki, W. R. (2010). Feature selection with the boruta package. *Journal of Statistical Software*, *36*(11), 1–13. https://doi.org/10.18637/jss.v036.i11

Laney, D. (2001). 3D Data Management: Controlling Data Volume, Velocity, and Variety. *Meta Group Research Note*, *6*.

Liaw, A., & Wiener, M. (2002a). Classification and Regression by randomForest. *R News*, *2*(3), 18–22.

Liaw, A., & Wiener, M. (2002b). Classification and Regression by randomForest. *R News*, *2*(3), 18–22. https://CRAN.R-project.org/doc/Rnews/

Liu, W., & Li, Q. (2017). An Efficient Elastic Net with Regression Coefficients Method for Variable Selection of Spectrum Data. *PLOS ONE*, *12*(2). https://doi.org/10.1371/journal.pone.0171122

Marafino, B. J., John Boscardin, W., & Adams Dudley, R. (2015). Efficient and sparse feature selection for biomedical text classification via the elastic net: Application to ICU risk stratification from nursing notes. *Journal of Biomedical Informatics*, *54*, 114–120. https://doi.org/10.1016/j.jbi.2015.02.003

Marsh, H. W., Hau, K. T., Balla, J. R., & Grayson, D. (1998). Is more ever too much? The number of indicators per factor in confirmatory factor analysis. *Multivariate Behavioral Research*, *33*(2), 181–220. https://doi.org/10.1207/s15327906mbr3302_1

McDonald, G. C. (2009). Ridge regression. *Wiley Interdisciplinary Reviews: Computational Statistics*, *1*(1), 93–100. https://doi.org/10.1002/WICS.14

McNeish, D. M. (2015). Using Lasso for Predictor Selection and to Assuage Overfitting: A Method Long Overlooked in Behavioral Sciences. *Multivariate Behavioral Research*, *50*(5), 471–484. https://doi.org/10.1080/00273171.2015.1036965

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., & Leisch, F. (2021). *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), Tu Wein*. https://CRAN.R-project.org/package=e1071

Neumann, J., Schnörr, C., & Steidl, G. (2005). Combined SVM-based feature selection and classification. *Machine Learning*, *61*(1–3), 129–150. https://doi.org/10.1007/s10994-005-1505-9

Rachmani, E., Hsu, C. Y., Nurjanah, N., Chang, P. W., Shidik, G. F., Noersasongko, E., Jumanto, J., Fuad, A., Ningrum, D. N. A., Kurniadi, A., & Lin, M. C. (2019). Developing an Indonesia's health literacy short-form survey questionnaire (HLS-EU-SQ10-IDN) using the feature selection and genetic algorithm. *Computer Methods and Programs in Biomedicine*, *182*, 105047. https://doi.org/10.1016/j.cmpb.2019.105047

Rachmani, E., Hsu, C.-Y., & Nurjanah, N. (2019). Developing an Indonesia's health literacy short-form survey questionnaire (HLS-EU-SQ10-IDN) using the feature selection and genetic algorithm. *Computer Methods and Programs in Biomedicine*, *182*, 105047. https://doi.org/10.1016/j.cmpb.2019.105047

Radloff, L. S. (1977). The CES-D Scale: A Self-Report Depression Scale for Research in the General Population. *Applied Psychological Measurement*, *1*(3), 385–401. https://doi.org/10.1177/014662167700100306

Rakotomamonjy, A. (2003). Variable selection using SVM-based criteria. *Journal of Machine Learning Research*, *3*, 1357–1370.

Ram, M., Najafi, A., & Shakeri, M. T. (2017). Classification and Biomarker Genes Selection for Cancer Gene Expression Data Using Random Forest. *JOURNAL OF PATHOLOGY Iranian Journal of Pathology*, *12*(4), 339–347.

Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.-C., & Müller, M. (2011). pROC: an open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinformatics*, *12*, 77.

Robitzsch, A. (2020). Why Ordinal Variables Can (Almost) Always Be Treated as Continuous Variables: Clarifying Assumptions of Robust Continuous and Ordinal Factor Analysis Estimation Methods. *Frontiers in Education*, *5*. https://doi.org/10.3389/feduc.2020.589965

Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics*, *23*(19), 2507–2517. https://doi.org/10.1093/bioinformatics/btm344

Sahdra, B. K., Ciarrochi, J., Parker, P., & Scrucca, L. (2016). Using Genetic Algorithms in a Large Nationally Representative American Sample to Abbreviate the Multidimensional Experiential Avoidance Questionnaire. *Frontiers in Psychology*, *7*(February), 1–14. https://doi.org/10.3389/fpsyg.2016.00189

Sanchez-Pinto, L. N., Venable, L. R., Fahrenbach, J., & Churpek, M. M. (2018). Comparison of variable selection methods for clinical predictive modeling. *International Journal of Medical Informatics*, *116*(February), 10–17. https://doi.org/10.1016/j.ijmedinf.2018.05.006

Sandy, C. J., Gosling, S. D., & Koelkebeck, T. (2014). Psychometric comparison of automated versus rational methods of scale abbreviation: An illustration using a brief measure of values. *Journal of Individual Differences*, *35*(4), 221–235. https://doi.org/10.1027/1614-0001/a000144

Schroeders, U., Wilhelm, O., & Olaru, G. (2016). Meta-Heuristics in Short Scale Construction: Ant Colony Optimization and Genetic Algorithm. *PLOS ONE*, *11*(11), e0167110. https://doi.org/10.1371/journal.pone.0167110

Scrucca, L. (2013). GA: A package for genetic algorithms in R. *Journal of Statistical Software*, *53*(4), 1–37. https://doi.org/10.18637/jss.v053.i04

Shapiro, D. N., Chandler, J., & Mueller, P. A. (2013). Using Mechanical Turk to Study Clinical Populations: *Https://Doi.Org/10.1177/2167702612469015*, *1*(2), 213–220. https://doi.org/10.1177/2167702612469015

Shen, Q., Jiang, J. H., Tao, J. C., Shen, G. L., & Yu, R. Q. (2005). Modified ant colony optimization algorithm for variable selection in QSAR modeling: QSAR studies of cyclooxygenase inhibitors. *Journal of Chemical Information and Modeling*, *45*(4), 1024–1029. https://doi.org/10.1021/ci049610z

Shevade, S. K., & Keerthi, S. S. (2003). A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, *19*(17), 2246–2253. https://doi.org/10.1093/BIOINFORMATICS/BTG308

Silia, V., & Chloe, H. (2018). *S-Five : a psychometric tool for assessing misophonia*.

Speiser, J. L., Durkalski, V. L., & Lee, W. M. (2015). Random forest classifcation of etiologies for an orphan disease. *Statistics in Medicine*, *34*, 887–899. https://doi.org/10.1002/sim.6351

Speiser, J. L., Miller, M. E., Tooze, J., & Ip, E. (2019). A comparison of random forest variable selection methods for classification prediction modeling. *Expert Systems with Applications*, *134*, 93–101. https://doi.org/10.1016/j.eswa.2019.05.028

Svetnik, V., Liaw, A., & Tong, C. (2000). Variable Selection in Random Forest with Application to Quantitative Structure-Activity Relationship. *Proceedings of the 7th Course on Ensemble Methods for Learning Machines*, *January*, 1–8.

Thompson, B. (1995). Stepwise Regression and Stepwise Discriminat Analysis Need Not Apply Here: A Guidelines Editorial. *Educational and Psychological Measurement*, *55*(4), 525–534.

Tibshirani, R. (1996). Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267–288. https://doi.org/10.1111/J.2517-6161.1996.TB02080.X

van der Kooij, A. J. (2007). *Prediction Accuracy and Stability of Regression with Optimal Scaling Transformations*. https://openaccess.leidenuniv.nl/dspace/handle/1887/12096

van der Linden, W. J. (1998). Optimal Assembly of Psychological and Educational Tests. *Applied Psychological Measurement*, *22*(3), 195–211. https://doi.org/10.1177/01466216980223001

Whittingham, M. J., Stephens, P. A., Bradbury, R. B., & Freckleton, R. P. (2006). Why do we still use stepwise modelling in ecology and behaviour? *Journal of Animal Ecology*, *75*(5), 1182–1189. https://doi.org/10.1111/J.1365-2656.2006.01141.X

Wiegand, R. E. (2010). *Performance of using multiple stepwise algorithms for variable selection ‡ Background and review of SVS algorithms*. https://doi.org/10.1002/sim.3943

Wu, M. S., Lewin, A. B., Murphy, T. K., & Storch, E. A. (2014). Misophonia: Incidence, phenomenology, and clinical correlates in an undergraduate student sample. *Journal of Clinical Psychology*, *70*(10), 994–1007. https://doi.org/10.1002/jclp.22098

Wu, T. T., Chen, Y. F., Hastie, T., Sobel, E., & Lange, K. (2009). Genome-wide association analysis by lasso penalized logistic regression. *Bioinformatics*, *25*(6), 714–721. https://doi.org/10.1093/BIOINFORMATICS/BTP041

Wu, T. T., & Lange, K. (2008). Coordinate descent algorithms for lasso penalized regression. *Annals of Applied Statistics*, *2*(1), 224–244. https://doi.org/10.1214/07-aoas147

Yarkoni, T. (2010). The abbreviation of personality, or how to measure 200 personality scales with 200 items. *Journal of Research in Personality*, *44*(2), 180–198. https://doi.org/10.1016/j.jrp.2010.01.002

Yarkoni, T. (2012). Psychoinformatics: New Horizons at the Interface of the Psychological and Computing Sciences. *Https://Doi.Org/10.1177/0963721412457362*, *21*(6), 391–397. https://doi.org/10.1177/0963721412457362

Zhang, H., Ahn, J., Lin, X., & Park, C. (2006). Gene selection using support vector machines with non-convex penalty. *Bioinformatics (Oxford, England)*, *22*(1), 88–95. https://doi.org/10.1093/BIOINFORMATICS/BTI736

Zou, H., & Hastie, T. (2005). Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society.*, *67*(2), 301–320. https://www.jstor.org/stable/3647580

Figures

Table 1
Summary of variable selection methods

Abbreviation in paper	Publication	R package
LR	(Cramer, 2005)	stats
SVM; SVMtuned	(Meyer et al., 2021)	e1071
RF; RFtuned	(Liaw & Wiener, 2002b)	randomForest
GA; GA tuned	(Scrucca, 2013)	GA
LASSO; LASSO tuned	(Friedman et al., 2010b)	glmnet
Elastic Net; EN tuned	(Friedman et al., 2010b)	glmnet
Elastic SVM; ESVM tuned	(Becker et al., 2011)	penalizedSVM
Boruta; bTuned	(Kursa & Rudnicki, 2010)	Boruta

Table 2

Distribution of prediction error rates, AUC, proportion of variables selected, TPR, and computation times for all methods. Note, the methods are not sorted in any way. For comparison, higher AUC and TPR were considered better while lower prediction error rates, proportion of variables selected, and computation time were considered better. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Prediction Error Rate	AUC	Proportion of Variables Selected	TPR	Computation Time (s)
Function (N)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta (10314)	0.265 (0.109)	0.627 (0.122)	0.337 (0.272)	0.737 (0.204)	3.893 (3.733)
$cv.glmnet_{elasticNet}$ (5488)	0.277 (0.102)	0.522 (0.06)	0.152 (0.221)	0.937 (0.156)	1.34 (2.783)
$glmnet_{elasticNet}$ (10800)	0.305 (0.114)	0.597 (0.114)	0.94 (0.1)	0.74 (0.204)	6.16 (10.344)
svmfs (10800)	0.264 (0.104)	0.572 (0.111)	0.522 (0.403)	0.789 (0.229)	37.203 (53.37)
ga (10800)	0.08 (0.096)	0.884 (0.137)	0.569 (0.1)	0.726 (0.213)	3.548 (4.532)
$cv.glmnet_{LASSO}$ (5462)	0.278 (0.102)	0.523 (0.064)	0.08 (0.139)	0.93 (0.182)	1.791 (4.576)
$glmnet_{LASSO}$ (10791)	0.311 (0.114)	0.596 (0.114)	0.677 (0.312)	0.669 (0.281)	0.405 (1.024)
glm (10800)	0.357 (0.128)	0.588 (0.106)	1 (0)	0.722 (0.208)	0.032 (0.055)
randomForest (10800)	0.258 (0.106)	0.623 (0.121)	1 (0)	0.722 (0.208)	0.225 (0.283)
svm (10800)	0.254 (0.102)	0.603 (0.116)	1 (0)	0.722 (0.208)	0.023 (0.029)

N: Number of models built with at least one variable (total possible, 10800)

SD: Standard Deviation

TPR: Proportion of selected variables that are not noise variables

Table 3

Distribution of prediction error rates, AUC, proportion of variables selected, TPR, and computation times for all methods excluding all datasets for which at least one of the methods did not produce a model utilizing at least one variable (4927 datasets used in this analysis). Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Prediction Error Rate	AUC	Proportion of Variables Selected	TPR	Computation Time (s)
Function	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.243 (0.103)	0.689 (0.103)	0.482 (0.272)	0.753 (0.204)	4.448 (4.209)
$cv.glmnet_{elasticNet}$	0.28 (0.101)	0.547 (0.079)	0.306 (0.231)	0.948 (0.141)	1.282 (2.411)
$glmnet_{elasticNet}$	0.275 (0.106)	0.648 (0.104)	0.935 (0.106)	0.753 (0.204)	5.744 (9.112)
svmfs	0.273 (0.1)	0.579 (0.11)	0.569 (0.415)	0.8 (0.219)	39.258 (54.437)
ga	0.097 (0.097)	0.865 (0.133)	0.572 (0.103)	0.739 (0.215)	3.63 (4.741)
$cv.glmnet_{LASSO}$	0.28 (0.101)	0.549 (0.085)	0.16 (0.16)	0.944 (0.161)	1.684 (4.029)
$glmnet_{LASSO}$	0.279 (0.107)	0.647 (0.102)	0.729 (0.293)	0.705 (0.26)	0.36 (0.918)
glm	0.317 (0.121)	0.626 (0.096)	1 (0)	0.734 (0.21)	0.033 (0.057)
randomForest	0.239 (0.099)	0.691 (0.102)	1 (0)	0.734 (0.21)	0.242 (0.274)
svm	0.236 (0.095)	0.682 (0.103)	1 (0)	0.734 (0.21)	0.024 (0.027)

SD: Standard Deviation

TPR: Proportion of selected variables that are not noise variables

Table 4

Distribution of prediction error rates, AUC, and proportion of variables selected for all methods across type of predictor. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Prediction Error Rate		AUC		Proportion of Variables Selected	
	Mean (SD)		Mean (SD)		Mean (SD)	
Function	Binary	Continuous	Binary	Continuous	Binary	Continuous
Boruta	0.254 (0.104)	0.276 (0.113)	0.637 (0.13)	0.618 (0.114)	0.311 (0.252)	0.363 (0.289)
$cv.glmnet_{elasticNet}$	0.278 (0.103)	0.277 (0.101)	0.52 (0.066)	0.525 (0.054)	0.139 (0.21)	0.165 (0.231)
$glmnet_{elasticNet}$	0.297 (0.113)	0.313 (0.114)	0.608 (0.122)	0.586 (0.105)	0.912 (0.117)	0.968 (0.068)
svmfs	0.259 (0.101)	0.269 (0.106)	0.586 (0.121)	0.559 (0.099)	0.471 (0.394)	0.573 (0.406)
ga	0.091 (0.093)	0.069 (0.099)	0.864 (0.136)	0.903 (0.136)	0.578 (0.101)	0.56 (0.097)
$cv.glmnet_{LASSO}$	0.279 (0.104)	0.277 (0.101)	0.521 (0.07)	0.526 (0.057)	0.088 (0.154)	0.072 (0.121)
$glmnet_{LASSO}$	0.301 (0.112)	0.321 (0.115)	0.607 (0.121)	0.585 (0.104)	0.66 (0.306)	0.694 (0.317)
glm	0.335 (0.121)	0.378 (0.131)	0.597 (0.112)	0.579 (0.098)	1 (0)	1 (0)
randomForest	0.253 (0.106)	0.264 (0.107)	0.632 (0.128)	0.614 (0.114)	1 (0)	1 (0)
svm	0.249 (0.102)	0.259 (0.102)	0.615 (0.126)	0.59 (0.104)	1 (0)	1 (0)

SD: Standard Deviation

Table 5

Distribution of TPR, and computation times for all methods across type of predictor. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	TPR		Computation Time (s)	
	Mean (SD)		Mean (SD)	
Function	Binary	Continuous	Binary	Continuous
Boruta	0.76 (0.194)	0.714 (0.211)	3.528 (3.195)	4.254 (4.168)
$cv.glmnet_{elasticNet}$	0.947 (0.137)	0.926 (0.173)	1.243 (2.545)	1.436 (3)
$glmnet_{elasticNet}$	0.763 (0.194)	0.718 (0.212)	5.627 (10.664)	6.693 (9.987)
svmfs	0.827 (0.218)	0.754 (0.234)	44.479 (61.644)	29.928 (42.326)
ga	0.728 (0.212)	0.724 (0.215)	3.994 (5.447)	3.102 (3.319)
$cv.glmnet_{LASSO}$	0.945 (0.151)	0.915 (0.208)	1.751 (4.416)	1.83 (4.731)
$glmnet_{LASSO}$	0.707 (0.253)	0.631 (0.302)	0.406 (1.152)	0.404 (0.877)
glm	0.722 (0.208)	0.722 (0.208)	0.034 (0.055)	0.031 (0.055)
randomForest	0.722 (0.208)	0.722 (0.208)	0.238 (0.306)	0.212 (0.257)
svm	0.722 (0.208)	0.722 (0.208)	0.021 (0.026)	0.026 (0.031)

SD: Standard Deviation

TPR: Proportion of selected variables that are not noise variables

Table 6

Distribution of prediction error rates for each method across prevalence rates. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Prediction Error Rates			
	Prevalence Rate .20	Prevalence Rate .25	Prevalence Rate .30	Prevalence Rate .40
Function	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.198 (0.079)	0.239 (0.087)	0.282 (0.098)	0.343 (0.113)
$cv.glmnet_{elasticNet}$	0.199 (0.065)	0.243 (0.072)	0.29 (0.077)	0.378 (0.094)
$glmnet_{elasticNet}$	0.24 (0.094)	0.284 (0.099)	0.321 (0.102)	0.376 (0.114)
svmfs	0.199 (0.073)	0.237 (0.081)	0.278 (0.089)	0.342 (0.109)
ga	0.053 (0.068)	0.069 (0.082)	0.085 (0.097)	0.113 (0.121)
$cv.glmnet_{LASSO}$	0.2 (0.065)	0.244 (0.073)	0.29 (0.077)	0.377 (0.096)
$glmnet_{LASSO}$	0.249 (0.095)	0.291 (0.102)	0.326 (0.104)	0.38 (0.112)
glm	0.315 (0.133)	0.34 (0.126)	0.366 (0.122)	0.406 (0.112)
randomForest	0.19 (0.075)	0.23 (0.081)	0.274 (0.092)	0.339 (0.112)
svm	0.187 (0.069)	0.226 (0.077)	0.269 (0.087)	0.336 (0.107)

SD: Standard Deviation

Table 7

Distribution of AUC for each method across prevalence rates. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	AUC			
	Prevalence Rate .20	Prevalence Rate .25	Prevalence Rate .30	Prevalence Rate .40
Function	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.627 (0.135)	0.624 (0.123)	0.626 (0.116)	0.633 (0.113)
$cv.glmnet_{elasticNet}$	0.518 (0.056)	0.517 (0.051)	0.521 (0.055)	0.532 (0.075)
$glmnet_{elasticNet}$	0.602 (0.126)	0.592 (0.113)	0.594 (0.11)	0.599 (0.108)
svmfs	0.549 (0.104)	0.562 (0.109)	0.575 (0.111)	0.603 (0.113)
ga	0.896 (0.137)	0.885 (0.138)	0.881 (0.137)	0.872 (0.137)
$cv.glmnet_{LASSO}$	0.517 (0.055)	0.518 (0.054)	0.524 (0.063)	0.535 (0.079)
$glmnet_{LASSO}$	0.601 (0.124)	0.593 (0.115)	0.593 (0.109)	0.597 (0.106)
glm	0.593 (0.118)	0.586 (0.106)	0.586 (0.102)	0.586 (0.094)
randomForest	0.622 (0.132)	0.619 (0.121)	0.621 (0.116)	0.629 (0.115)
svm	0.59 (0.123)	0.596 (0.115)	0.605 (0.113)	0.621 (0.111)

SD: Standard Deviation

Table 8

Distribution of proportion of variables selected for each method across prevalence rates. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Proportion of Variables Selected			
	Prevalence Rate .20	Prevalence Rate .25	Prevalence Rate .30	Prevalence Rate .40
Function	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.355 (0.272)	0.349 (0.272)	0.339 (0.274)	0.305 (0.268)
$cv.glmnet_{elasticNet}$	0.147 (0.236)	0.148 (0.227)	0.151 (0.217)	0.163 (0.204)
$glmnet_{elasticNet}$	0.939 (0.099)	0.94 (0.097)	0.94 (0.101)	0.941 (0.102)
svmfs	0.446 (0.366)	0.49 (0.392)	0.521 (0.409)	0.632 (0.42)
ga	0.566 (0.1)	0.567 (0.1)	0.57 (0.098)	0.573 (0.1)
$cv.glmnet_{LASSO}$	0.075 (0.144)	0.076 (0.14)	0.08 (0.137)	0.089 (0.136)
$glmnet_{LASSO}$	0.646 (0.319)	0.671 (0.312)	0.687 (0.307)	0.703 (0.307)
glm	1 (0)	1 (0)	1 (0)	1 (0)
randomForest	1 (0)	1 (0)	1 (0)	1 (0)
svm	1 (0)	1 (0)	1 (0)	1 (0)

SD: Standard Deviation

Table 9

Distribution of TPR for each method across prevalence rates. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	TPR			
	Prevalence Rate .20	Prevalence Rate .25	Prevalence Rate .30	Prevalence Rate .40
Function	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.74 (0.205)	0.738 (0.204)	0.738 (0.204)	0.731 (0.203)
$cv.glmnet_{elasticNet}$	0.952 (0.119)	0.947 (0.14)	0.942 (0.148)	0.916 (0.187)
$glmnet_{elasticNet}$	0.741 (0.204)	0.741 (0.204)	0.741 (0.205)	0.739 (0.205)
svmfs	0.781 (0.24)	0.785 (0.235)	0.795 (0.227)	0.796 (0.214)
ga	0.726 (0.214)	0.726 (0.214)	0.725 (0.213)	0.727 (0.212)
$cv.glmnet_{LASSO}$	0.945 (0.147)	0.941 (0.17)	0.936 (0.173)	0.908 (0.211)
$glmnet_{LASSO}$	0.665 (0.285)	0.668 (0.281)	0.672 (0.28)	0.671 (0.278)
glm	0.722 (0.208)	0.722 (0.208)	0.722 (0.208)	0.722 (0.208)
randomForest	0.722 (0.208)	0.722 (0.208)	0.722 (0.208)	0.722 (0.208)
svm	0.722 (0.208)	0.722 (0.208)	0.722 (0.208)	0.722 (0.208)

SD: Standard Deviation
TPR: Proportion of selected variables that are not noise variables

Table 10

Distribution of computation times for each method across prevalence rates. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Computation Time (s)			
	Prevalence Rate .20	Prevalence Rate .25	Prevalence Rate .30	Prevalence Rate .40
Function	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	3.762 (3.671)	3.912 (3.851)	3.981 (3.792)	3.916 (3.609)
$cv.glmnet_{elasticNet}$	1.341 (2.516)	1.322 (2.679)	1.346 (2.862)	1.349 (3.049)
$glmnet_{elasticNet}$	6.536 (10.808)	6.043 (9.296)	6.17 (10.958)	5.892 (10.225)
svmfs	26.348 (40.572)	33.6 (51.212)	41.77 (57.533)	47.095 (59.736)
ga	3.577 (4.437)	3.588 (4.56)	3.566 (4.922)	3.461 (4.178)
$cv.glmnet_{LASSO}$	1.746 (4.065)	1.728 (4.206)	1.766 (4.478)	1.923 (5.43)
$glmnet_{LASSO}$	0.424 (0.991)	0.395 (0.944)	0.403 (1.08)	0.398 (1.074)
glm	0.034 (0.056)	0.033 (0.055)	0.032 (0.055)	0.031 (0.054)
randomForest	0.211 (0.269)	0.221 (0.279)	0.23 (0.287)	0.237 (0.296)
svm	0.022 (0.028)	0.023 (0.028)	0.024 (0.029)	0.025 (0.03)

SD: Standard Deviation

Table 11

Distribution of prediction error rates and AUC for all methods across sample size. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Prediction Error Rate			AUC		
	Sample Size 50	Sample Size 200	Sample Size 500	Sample Size 50	Sample Size 200	Sample Size 500
	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.283 (0.141)	0.265 (0.094)	0.249 (0.086)	0.622 (0.154)	0.621 (0.109)	0.638 (0.1)
$cv.glmnet_{elasticNet}$	0.284 (0.129)	0.276 (0.089)	0.272 (0.081)	0.518 (0.067)	0.522 (0.056)	0.528 (0.056)
$glmnet_{elasticNet}$	0.315 (0.142)	0.311 (0.101)	0.289 (0.091)	0.595 (0.147)	0.592 (0.098)	0.603 (0.088)
svmfs	0.275 (0.133)	0.263 (0.088)	0.254 (0.08)	0.565 (0.123)	0.569 (0.103)	0.584 (0.106)
ga	0.032 (0.066)	0.079 (0.098)	0.129 (0.096)	0.95 (0.103)	0.885 (0.139)	0.816 (0.132)
$cv.glmnet_{LASSO}$	0.285 (0.13)	0.277 (0.089)	0.272 (0.081)	0.52 (0.075)	0.522 (0.058)	0.528 (0.058)
$glmnet_{LASSO}$	0.33 (0.14)	0.314 (0.101)	0.29 (0.091)	0.591 (0.147)	0.592 (0.098)	0.603 (0.087)
glm	0.404 (0.147)	0.365 (0.116)	0.301 (0.093)	0.587 (0.135)	0.577 (0.089)	0.599 (0.084)
randomForest	0.274 (0.136)	0.257 (0.091)	0.244 (0.083)	0.608 (0.149)	0.621 (0.108)	0.639 (0.1)
svm	0.271 (0.131)	0.252 (0.087)	0.24 (0.079)	0.574 (0.129)	0.604 (0.108)	0.63 (0.102)

SD: Standard Deviation

Table 12

Distribution of the proportion of variables selected by each method and TPR for all methods across sample size. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Proportion of Variables Selected			TPR		
	Sample Size 50	Sample Size 200	Sample Size 500	Sample Size 50	Sample Size 200	Sample Size 500
	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.182 (0.189)	0.325 (0.256)	0.487 (0.271)	0.182 (0.189)	0.325 (0.256)	0.487 (0.271)
$cv.glmnet_{elasticNet}$	0.075 (0.153)	0.147 (0.215)	0.235 (0.254)	0.075 (0.153)	0.147 (0.215)	0.235 (0.254)
$glmnet_{elasticNet}$	0.914 (0.112)	0.953 (0.09)	0.953 (0.09)	0.914 (0.112)	0.953 (0.09)	0.953 (0.09)
svmfs	0.487 (0.408)	0.507 (0.402)	0.573 (0.394)	0.487 (0.408)	0.507 (0.402)	0.573 (0.394)
ga	0.555 (0.11)	0.574 (0.098)	0.578 (0.088)	0.555 (0.11)	0.574 (0.098)	0.578 (0.088)
$cv.glmnet_{LASSO}$	0.038 (0.097)	0.075 (0.134)	0.127 (0.164)	0.038 (0.097)	0.075 (0.134)	0.127 (0.164)
$glmnet_{LASSO}$	0.411 (0.312)	0.719 (0.234)	0.9 (0.135)	0.411 (0.312)	0.719 (0.234)	0.9 (0.135)
glm	1 (0)	1 (0)	1 (0)	1 (0)	1 (0)	1 (0)
randomForest	1 (0)	1 (0)	1 (0)	1 (0)	1 (0)	1 (0)
svm	1 (0)	1 (0)	1 (0)	1 (0)	1 (0)	1 (0)

SD: Standard Deviation

TPR: Proportion of variables selected that are not noise variables

Table 13

Distribution of the computation times for all methods across sample size. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Computation Time (s)		
	Sample Size 50	Sample Size 200	Sample Size 500
	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	1.391 (0.476)	2.674 (1.178)	7.312 (4.445)
$cv.glmnet_{elasticNet}$	0.095 (0.037)	0.674 (0.715)	3.249 (4.134)
$glmnet_{elasticNet}$	2.011 (2.468)	5.328 (4.733)	11.141 (15.806)
svmfs	23.677 (28.982)	28.446 (36.112)	59.487 (75.142)
ga	1.633 (0.994)	3.261 (2.751)	5.75 (6.668)
$cv.glmnet_{LASSO}$	0.094 (0.063)	0.773 (1.045)	4.505 (7.103)
$glmnet_{LASSO}$	0.103 (0.198)	0.326 (0.425)	0.785 (1.638)
glm	0.011 (0.014)	0.028 (0.03)	0.059 (0.082)
randomForest	0.031 (0.023)	0.158 (0.108)	0.485 (0.344)
svm	0.009 (0.012)	0.016 (0.016)	0.046 (0.036)

SD: Standard Deviation

Table 14

Distribution of prediction error rates and AUC for all methods across ratio of observations to number of predictors. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Prediction Error Rate		AUC	
	$p \leq n$	$p \gg n$	$p \leq n$	$p \gg n$
	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.261 (0.101)	0.286 (0.14)	0.629 (0.114)	0.622 (0.155)
$\text{cv.glmnet}_{\text{elasticNet}}$	0.275 (0.095)	0.287 (0.129)	0.524 (0.059)	0.517 (0.065)
$\text{glmnet}_{\text{elasticNet}}$	0.306 (0.107)	0.302 (0.14)	0.596 (0.105)	0.599 (0.149)
svmfs	0.261 (0.095)	0.275 (0.134)	0.572 (0.107)	0.575 (0.13)
ga	0.095 (0.099)	0.013 (0.036)	0.862 (0.14)	0.978 (0.066)
$\text{cv.glmnet}_{\text{LASSO}}$	0.276 (0.095)	0.288 (0.13)	0.524 (0.061)	0.52 (0.075)
$\text{glmnet}_{\text{LASSO}}$	0.308 (0.107)	0.326 (0.139)	0.596 (0.105)	0.593 (0.147)
glm	0.341 (0.118)	0.427 (0.145)	0.588 (0.099)	0.588 (0.131)
randomForest	0.255 (0.099)	0.273 (0.134)	0.626 (0.114)	0.608 (0.149)
svm	0.251 (0.095)	0.27 (0.129)	0.609 (0.112)	0.574 (0.13)

SD: Standard Deviation

Table 15

Distribution of the proportion of variables selected and TPR for all methods across ratio of observations to number of predictors. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Proportion of Variables Selected		TPR	
	$p \leq n$	$p \gg n$	$p \leq n$	$p \gg n$
	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Boruta	0.389 (0.27)	0.097 (0.094)	0.747 (0.205)	0.687 (0.189)
$\text{cv.glmnet}_{\text{elasticNet}}$	0.178 (0.236)	0.037 (0.063)	0.951 (0.134)	0.841 (0.243)
$\text{glmnet}_{\text{elasticNet}}$	0.955 (0.09)	0.875 (0.114)	0.751 (0.206)	0.695 (0.192)
svmfs	0.54 (0.402)	0.443 (0.4)	0.791 (0.226)	0.781 (0.243)
ga	0.584 (0.102)	0.503 (0.047)	0.739 (0.217)	0.667 (0.186)
$\text{cv.glmnet}_{\text{LASSO}}$	0.095 (0.15)	0.015 (0.03)	0.948 (0.15)	0.807 (0.296)
$\text{glmnet}_{\text{LASSO}}$	0.79 (0.222)	0.18 (0.079)	0.715 (0.245)	0.465 (0.334)
glm	1 (0)	1 (0)	0.735 (0.211)	0.667 (0.183)
randomForest	1 (0)	1 (0)	0.735 (0.211)	0.667 (0.183)
svm	1 (0)	1 (0)	0.735 (0.211)	0.667 (0.183)

SD: Standard Deviation

TPR: Proportion of variables selected that are not noise variables

Table 16

Distribution of computation times for all methods across ratio of observations to number of predictors. Note that the last three rows of the table contain results from methods that are not variable selection methods and are only included for relative comparison to a model built using all possible variables.

	Computation Time (s)	
	$p \leq n$	$p \gg n$
	Mean (SD)	Mean (SD)
Boruta	4.403 (3.926)	1.519 (0.487)
cv.glmnet$_{elasticNet}$	1.624 (3.011)	0.087 (0.027)
glmnet$_{elasticNet}$	6.975 (11.201)	2.573 (3.152)
svmfs	37.704 (56.83)	35.002 (34.144)
ga	3.83 (4.962)	2.308 (0.82)
cv.glmnet$_{LASSO}$	2.18 (4.988)	0.077 (0.028)
glmnet$_{LASSO}$	0.466 (1.118)	0.136 (0.254)
glm	0.036 (0.06)	0.016 (0.015)
randomForest	0.266 (0.299)	0.045 (0.021)
svm	0.026 (0.031)	0.013 (0.012)

SD: Standard Deviation

Table 17

Distribution of prediction error rates, AUC, number of predictors, TPR, and computation time selected across method types.

Method Type	Prediction Error Rate	Computation Time (s)	AUC	TPR	Proportion of Variables Selected
	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Metaheuristic	0.08 (0.096)	3.377 (4.373)	0.884 (0.137)	0.726 (0.213)	0.567 (0.1)
Regression-based	0.285 (0.108)	10.438 (29.784)	0.565 (0.1)	0.763 (0.235)	0.453 (0.357)
Tree-based	0.266 (0.111)	3.794 (3.661)	0.627 (0.122)	0.737 (0.204)	0.336 (0.273)

SD: Standard Deviation

TPR: Proportion of variables selected that are not noise variables

Table 18

Prediction error rates, AUC, number of predictors, and computation time selected by each method when applied to the Alcohol Use Disorder dataset.

Function Used	Prediction Error Rate	AUC	Number of Predictors	Computation Time (seconds)	Parameter settings
glm	0.058	0.920	86	0.090	default
svm	0.058	0.932	86	0.060	default
svm[1]	0.036	0.955	86	0.860	cost = 1.75
randomForest	0.076	0.884	86	0.700	default
randomForest[1]	0.076	0.884	86	92.420	mtry = 14, ntree = 500
ga	0	1	57	20.740	default and run = 5
ga[1]	0.029	0.968	52	2559.680	run = 5, popSize = 30, pcrossover = 0.6, pmutation = 0.001
cv.glmnet	0.083	0.863	21	1.720	lambda= 0.026, alpha = 1
glmnet[1]	0.047	0.936	69	0.190	lambda= 0.0001, alpha = 1
cv.glmnet	0.040	0.945	61	1.510	alpha=.05, lambda= 0.0295
glmnet[1]	0.050	0.929	78	3.060	alpha= 0.053 lambda= .0001
svmfs	0.277	0.5	83	130.340	lambda1 = -5.86 , lambda2 = 3.05, bounds = (-10,10)
svmfs[1]	0.277	0.5	77	6725.570	lambda1 = 1e-04, lambda2 = .0001, maxevals = 10, maxIter = 100
Boruta	0.076	0.884	62	7.970	Default, proximity = T
Boruta[1]	0.068	0.893	51	4024.32	maxRuns = 500, mtry = 5, ntree = 100

[1] The parameters for these methods were tuned using a grid search.

Table 19

Prediction error rates, AUC, number of predictors, and computation time selected by each method when applied to the Misophonia Questionnaire.

Method	Prediction Error Rate	AUC	Number of Predictors	Computation Time (seconds)	Parameter settings
glm	0.141	0.732	19	0.020	default
svm	0.141	0.546	19	0.020	default
svm[1]	0.203	0.557	19	0.070	cost = 5
randomForest	0.156	0.677	19	0.040	default
randomForest[1]	0.172	0.668	19	1.840	mtry = 5, ntree = 100
ga	0	1	13	0.700	default and run = 5
ga[1]	0	1	14	22.670	run = 5, popSize = 30, pcrossover = 0.5, pmutation = 0.001
cv.glmnet	0.141	0.500	0	0.109	lambda= 0.175
glmnet[1]	0.141	0.732	19	0.030	lambda= 0.0001
cv.glmnet	0.125	0.556	11	0.070	alpha=.05, lambda= 0.175
glmnet[1]	0.141	0.732	19	0.720	alpha= 0.579, lambda= .0001
svmfs	0.141	0.500	4	4.050	lambda1 = -1.60 , lambda2 = 7.87, bounds = (-10,10)
svmfs[1]	0.141	0.500	6	480.520	lambda1 = 0.05, lambda2 = 0.05, maxevals = 10, maxIter = 500
Boruta	0.141	0.686	7	1.270	Default, proximity = T
Boruta[1]	0.141	0.686	5	291.260	maxRuns = 500, mtry = 9, ntree = 100

[1] The parameters for these methods were tuned using a grid search.

Table 20

Prediction error rates, AUC, number of predictors, and computation time selected by each method when applied to the S-Five.

Method	Prediction Error Rate	AUC	Number of Predictors	Computation Time (seconds)	Parameter settings
glm	0.386	0.456	86	0.040	default
svm	0.119	0.500	86	0.030	default
svm[1]	0.149	0.483	86	0.280	cost = 5
randomForest	0.119	0.500	86	0.230	default
randomForest[1]	0.119	0.500	86	36.060	mtry = 16, ntree = 100
ga	0	1	56	3.030	default and run = 5
ga[1]	0	1	56	661.940	run = 5, popSize = 30, pcrossover = 0.5, pmutation = 0.001
cv.glmnet	0.119	0.500	0	0.780	lambda= 0.076
glmnet[1]	0.366	0.432	68	0.110	lambda= 0.0001
cv.glmnet	0.119	0.500	0	0.750	alpha=.05, lambda= 0.076
glmnet[1]	0.356	0.437	85	1.780	alpha= 0.053 lambda= .0001
svmfs	0.168	0.500	41	22.030	lambda1 = -4.36 , lambda2 = 3.72, bounds = (-10,10
svmfs[1]	0.129	0.494	59	3808.450	lambda1 = 0.01, lambda2 = .3, maxevals = 10, maxIter = 50
Boruta	0.139	0.489	14	2.050	Default, proximity = TRUE
Boruta[1]	0.218	0.444	6	1383.970	maxRuns = 500, mtry = 3, ntree = 100, proximity = TRUE

[1] The parameters for these methods were tuned using a grid search.

Figure 4

This figure displays boxplots of the prediction error rate and AUC for methods.

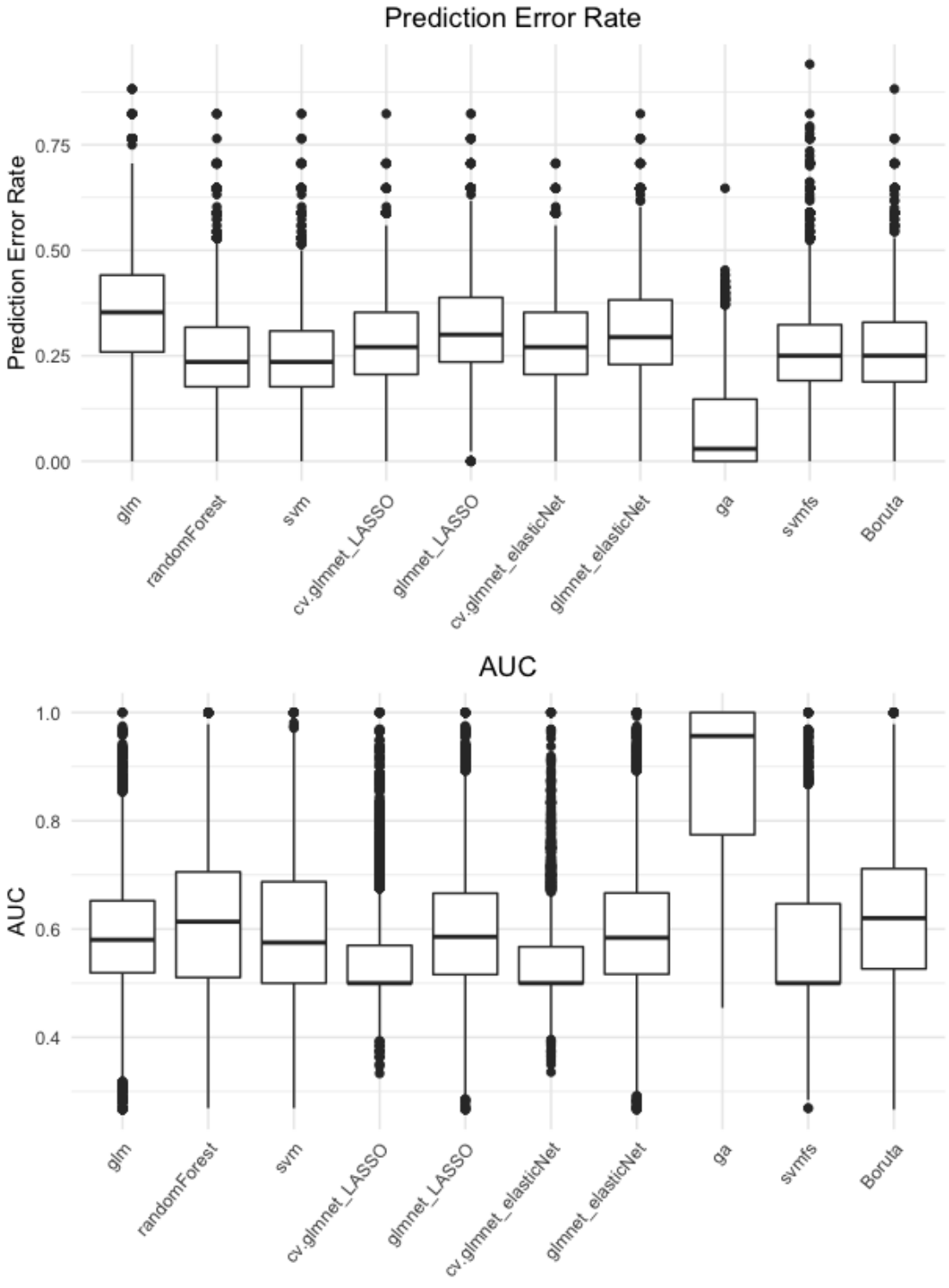

Figure 5

This figure displays boxplots of the computation times, proportion of variables selected and TPR for the methods. The TPR refers to the proportion of variables selected that are not noise variables. Note that the computation time of svmfs ranged from 1 to 439 seconds but the range has been limited to increase legibility of the plot.

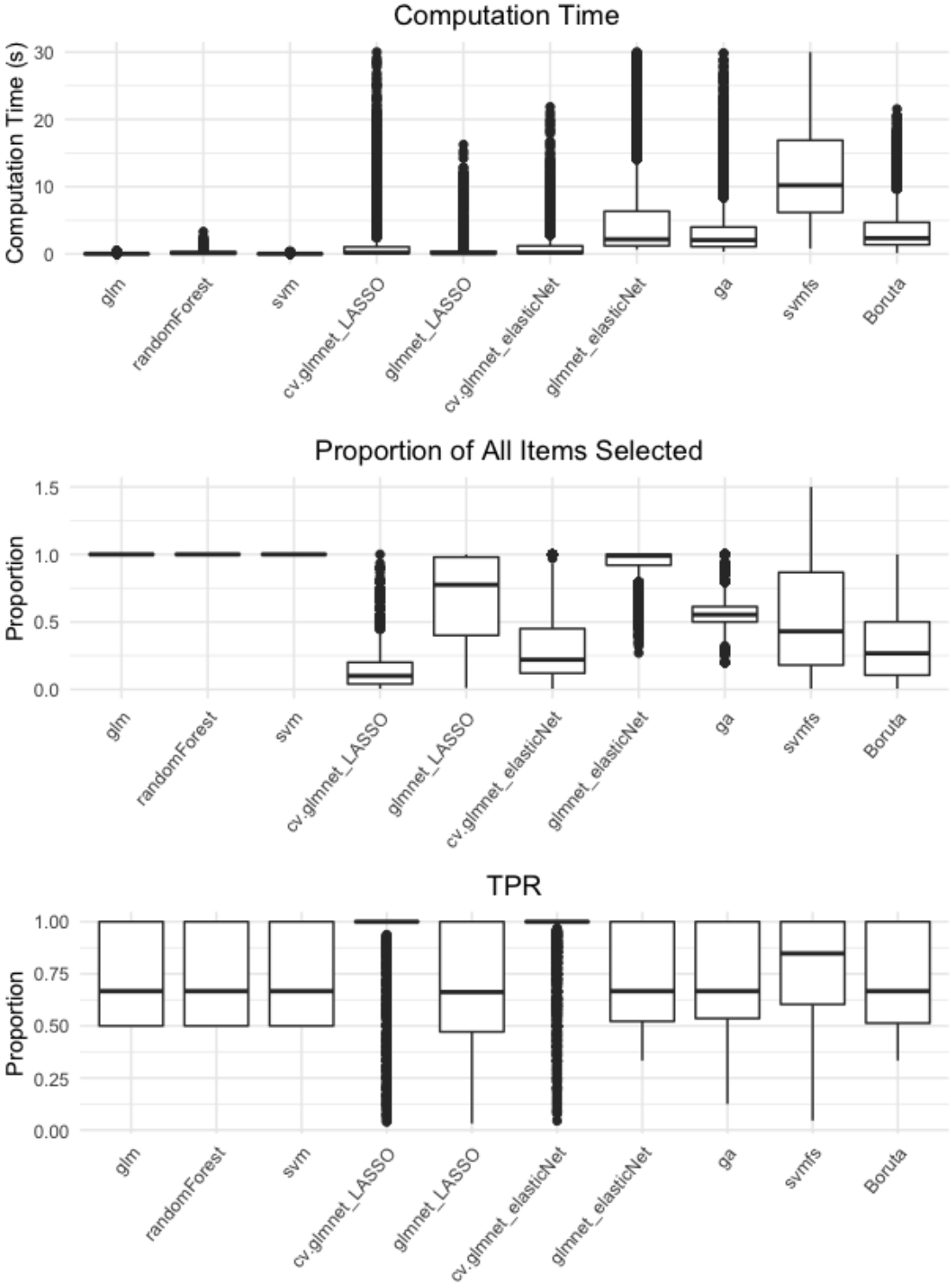

Figure 6

This figure displays plots of the median prediction error rate by median computation time, proportion of variables selected, and AUC.

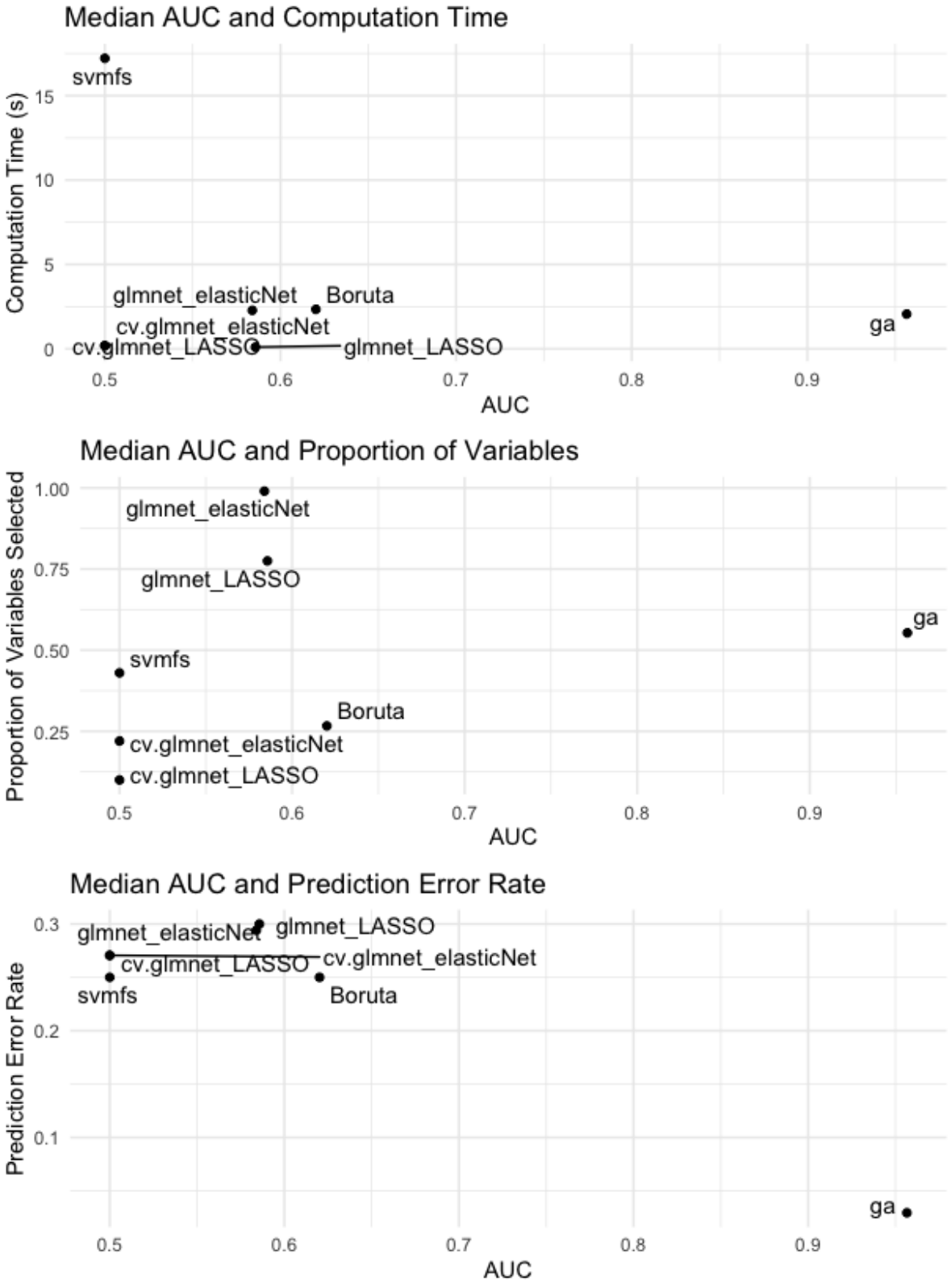

Figure 7

This figure displays boxplots of the prediction error rate and AUC for the methods as compared across predictor type (continuous vs. binary).

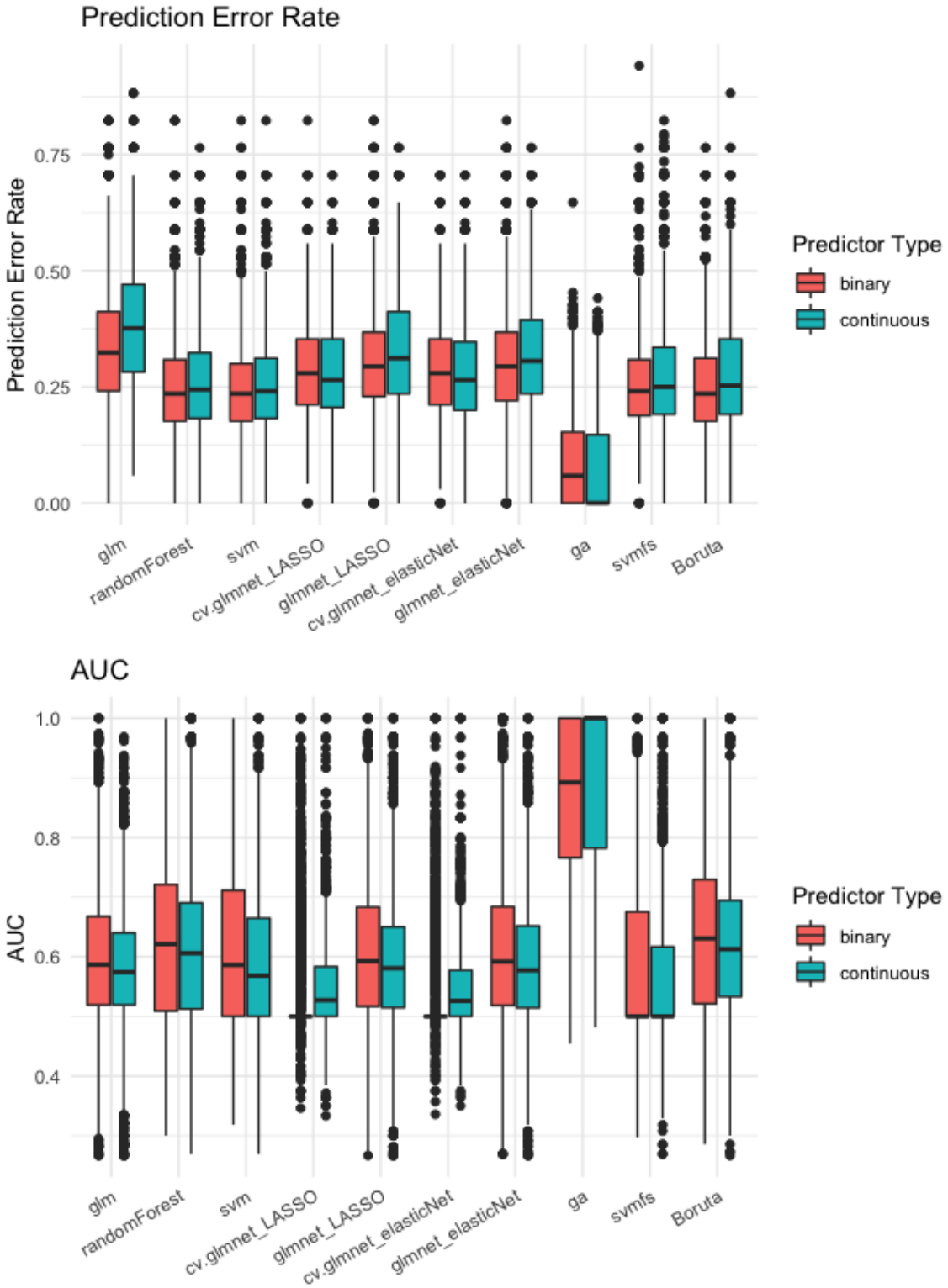

Figure 8

This figure displays boxplots of the computation times, proportion of variables selected, and TPR for the methods as compared across predictor type (continuous vs. binary). Note that TPR refers to the proportion of variables selected that are not noise variables and that the range in computation time has been restricted to enhance visibility.

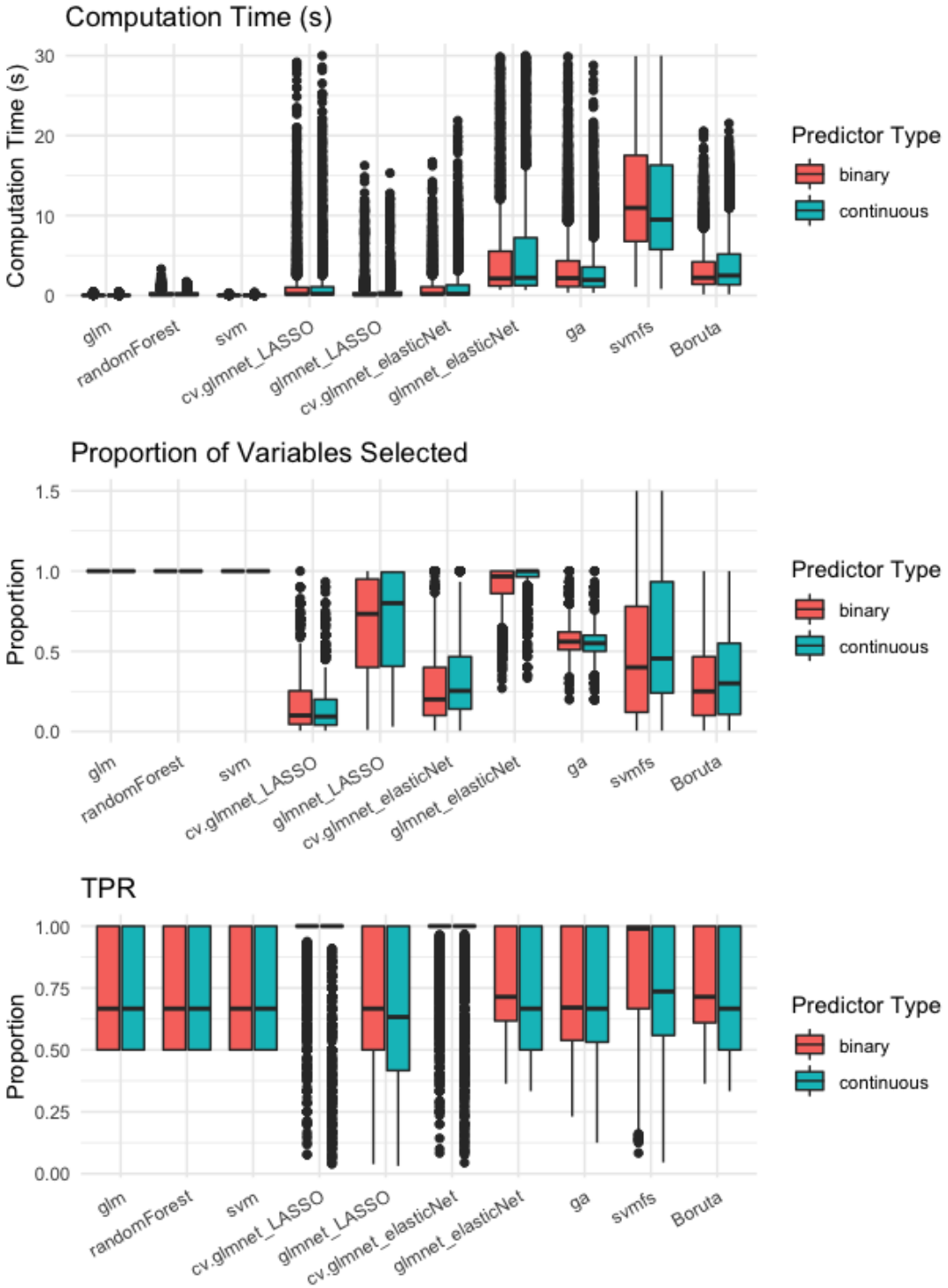

Figure 9

This figure displays boxplots of the prediction error rates and AUC for the methods for classification as compared across prevalence rate.

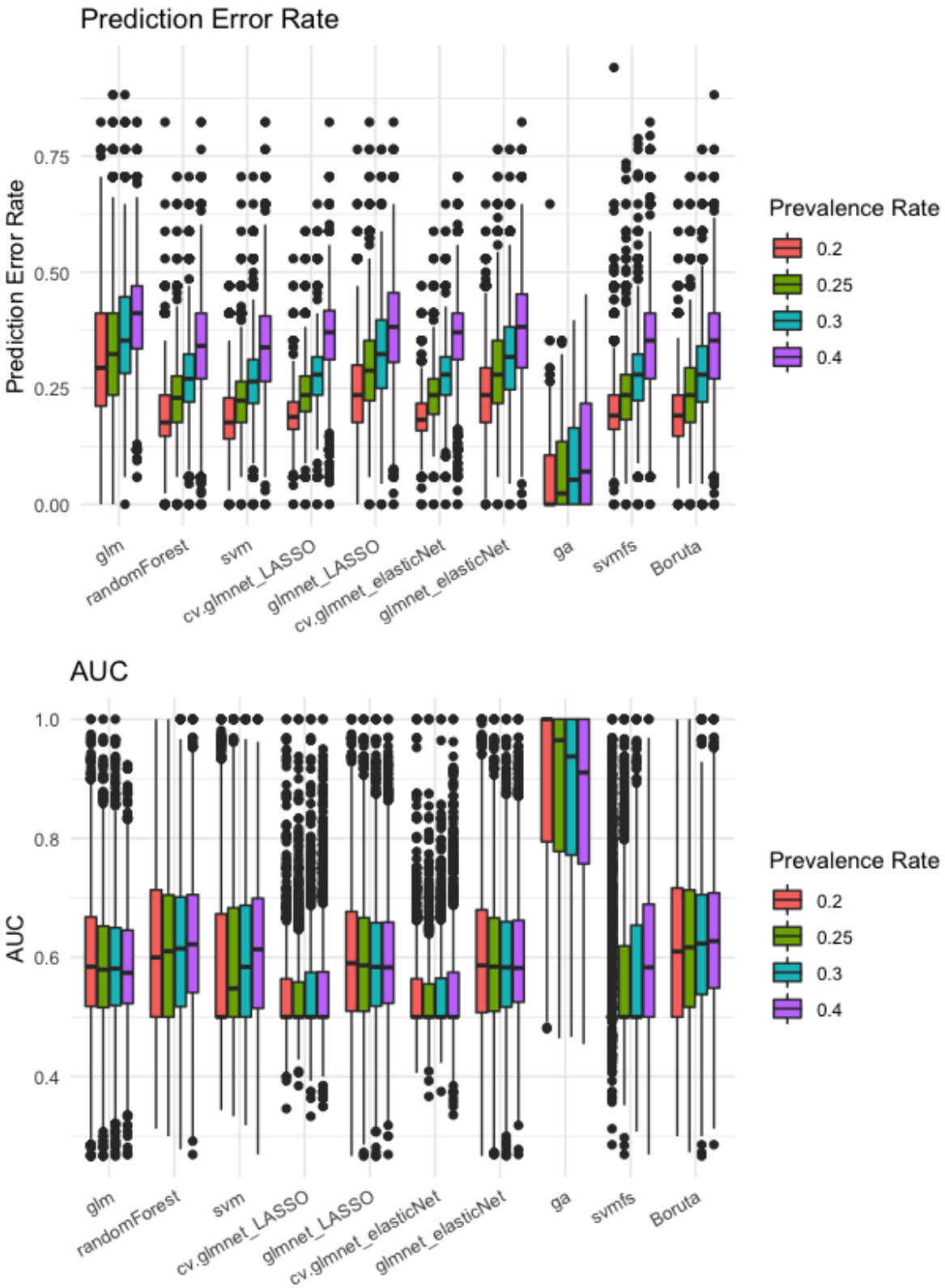

Figure 10

This figure displays boxplots of the computation times, proportion of variables selected, and TPR for the methods as compared across prevalence rates. Note that TPR refers to the proportion of variables selected that are not noise variables and the range in computation time has been restricted to enhance visibility.

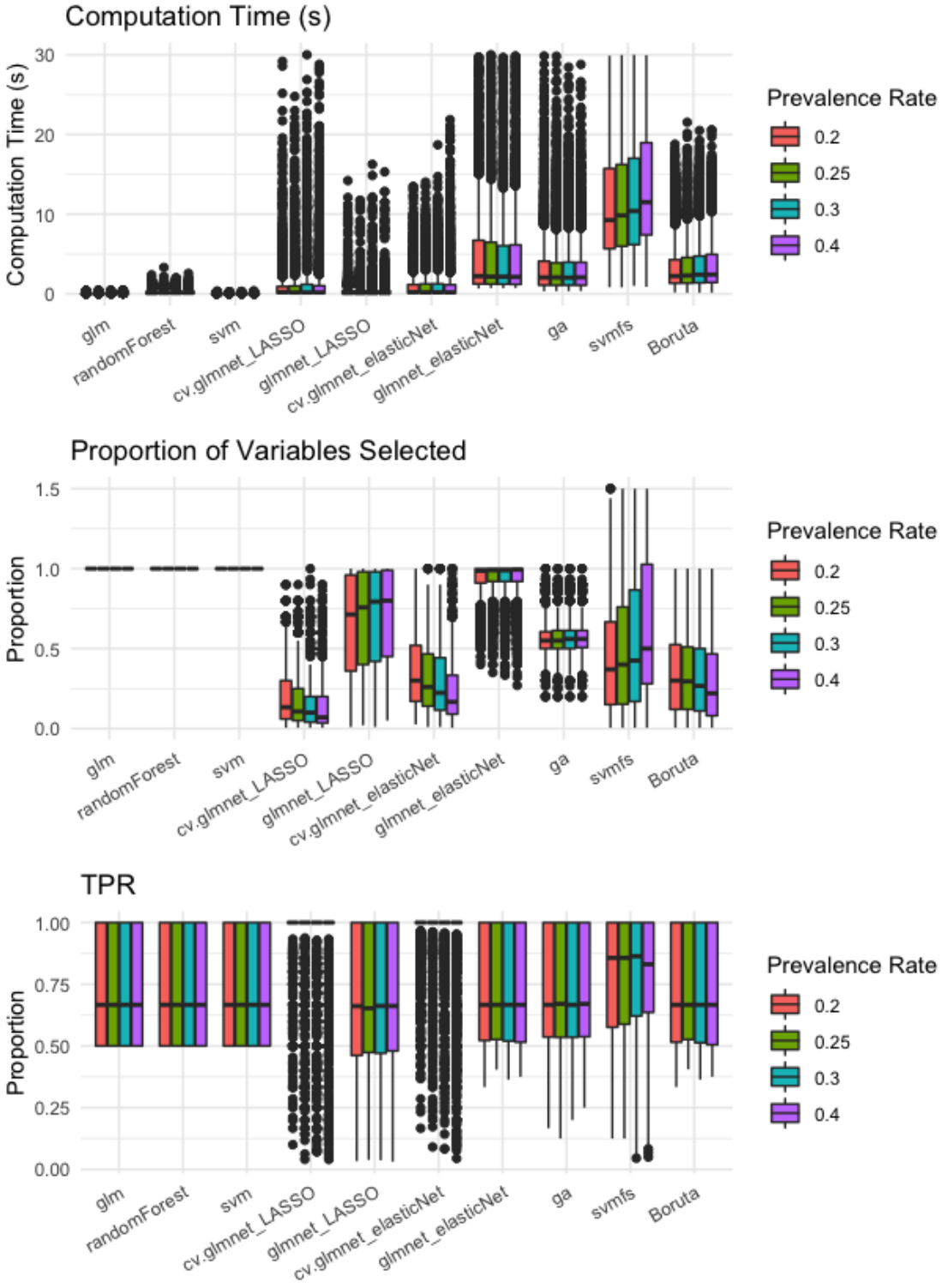

Figure 11

This figure displays boxplots of the prediction error rates and AUC for the methods as compared across sample sizes.

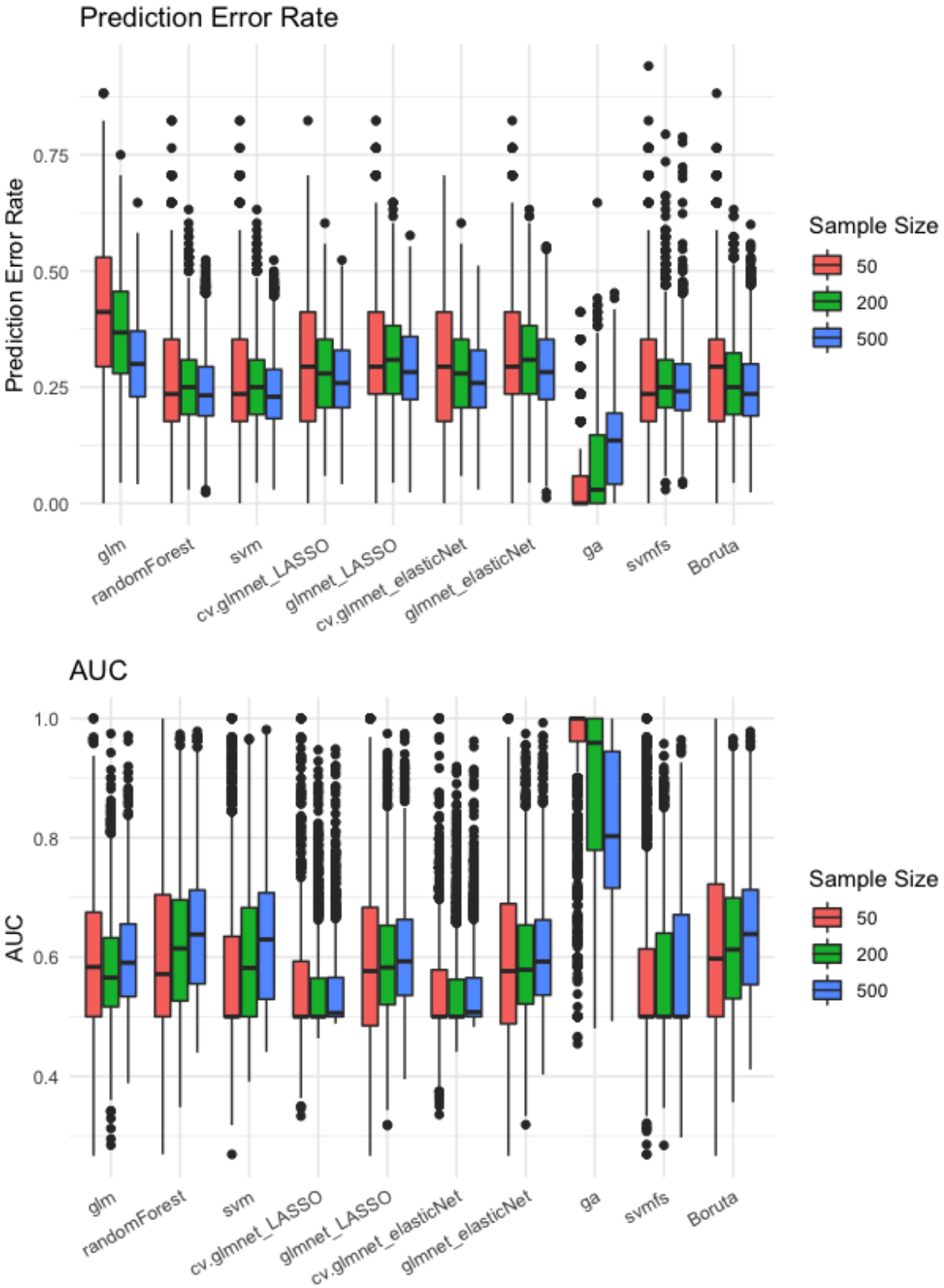

Figure 12

This figure displays boxplots of the computation times, proportion of variables selected, and TPR for the methods as compared across sample sizes. Note that TPR refers to the proportion of variables selected that are not noise variables and the range in computation time has been restricted to enhance visibility.

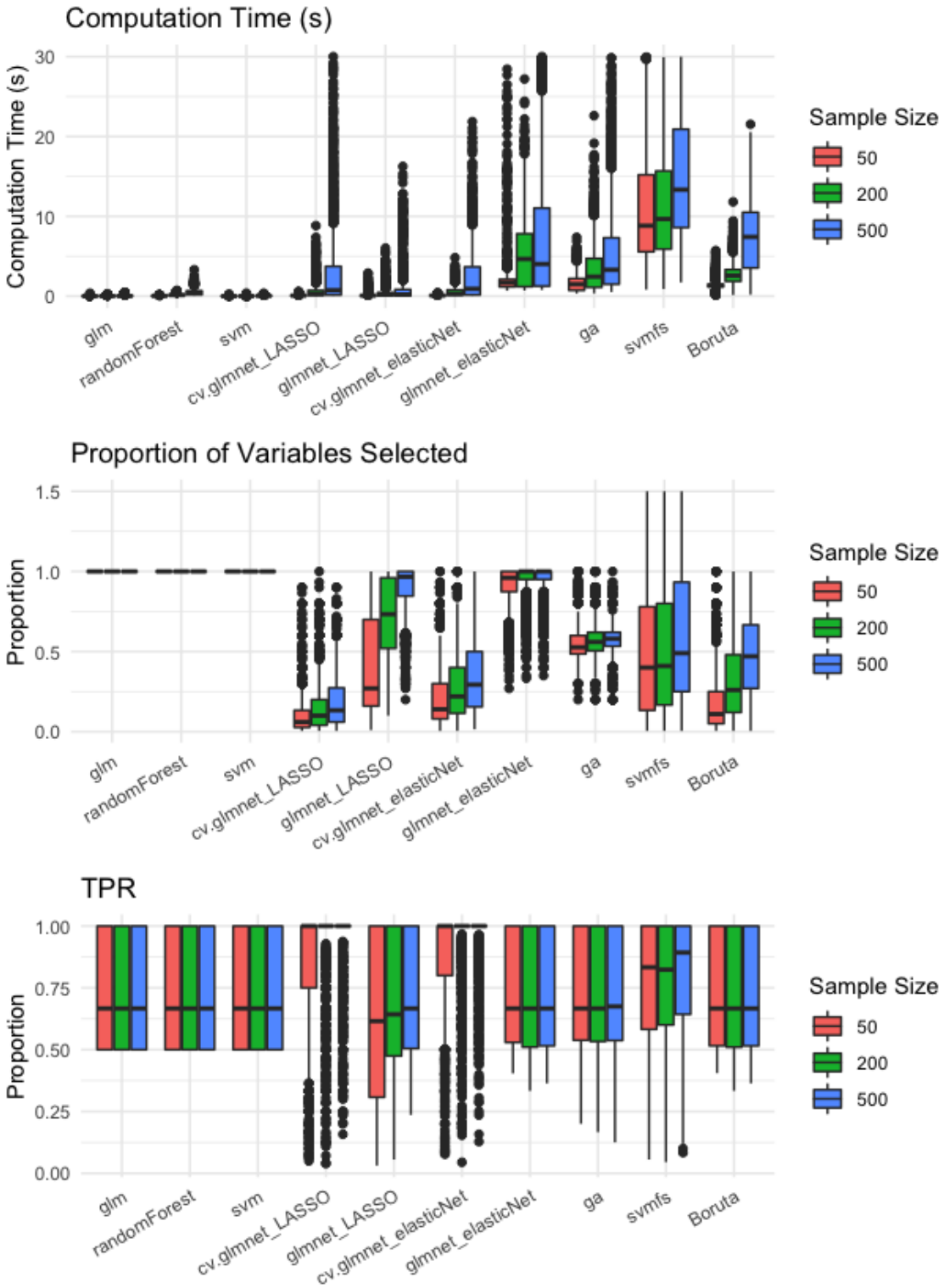

Figure 13

This figure displays boxplots of the prediction error rates and AUC for the methods as compared across data dimensions.

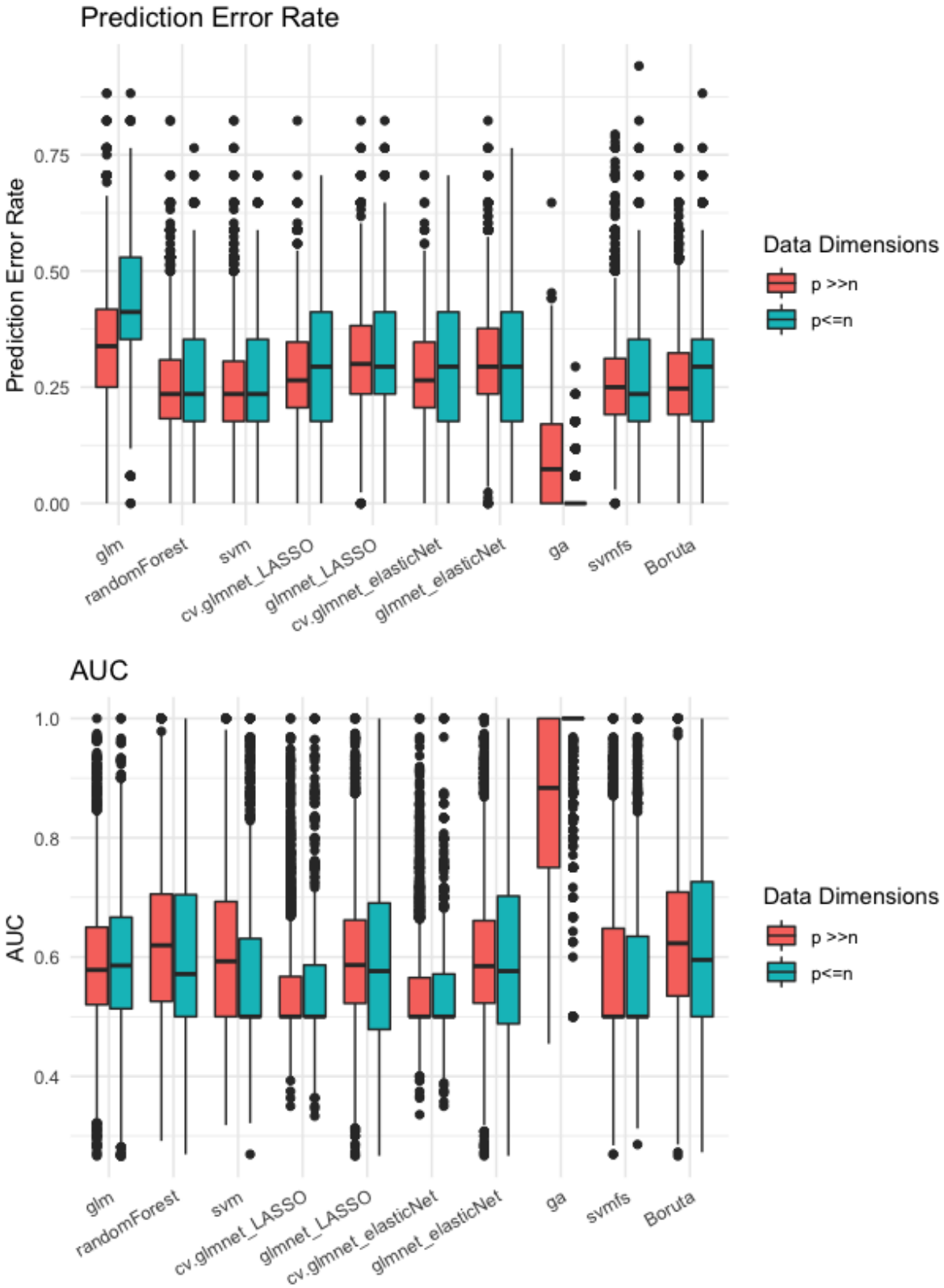

Figure 14

This figure displays boxplots of the computation times, proportion of variables selected, and TPR for the methods as compared across data dimensions. Note that TPR refers to the proportion of variables selected that are not noise variables and the range in computation time has been restricted to enhance visibility.

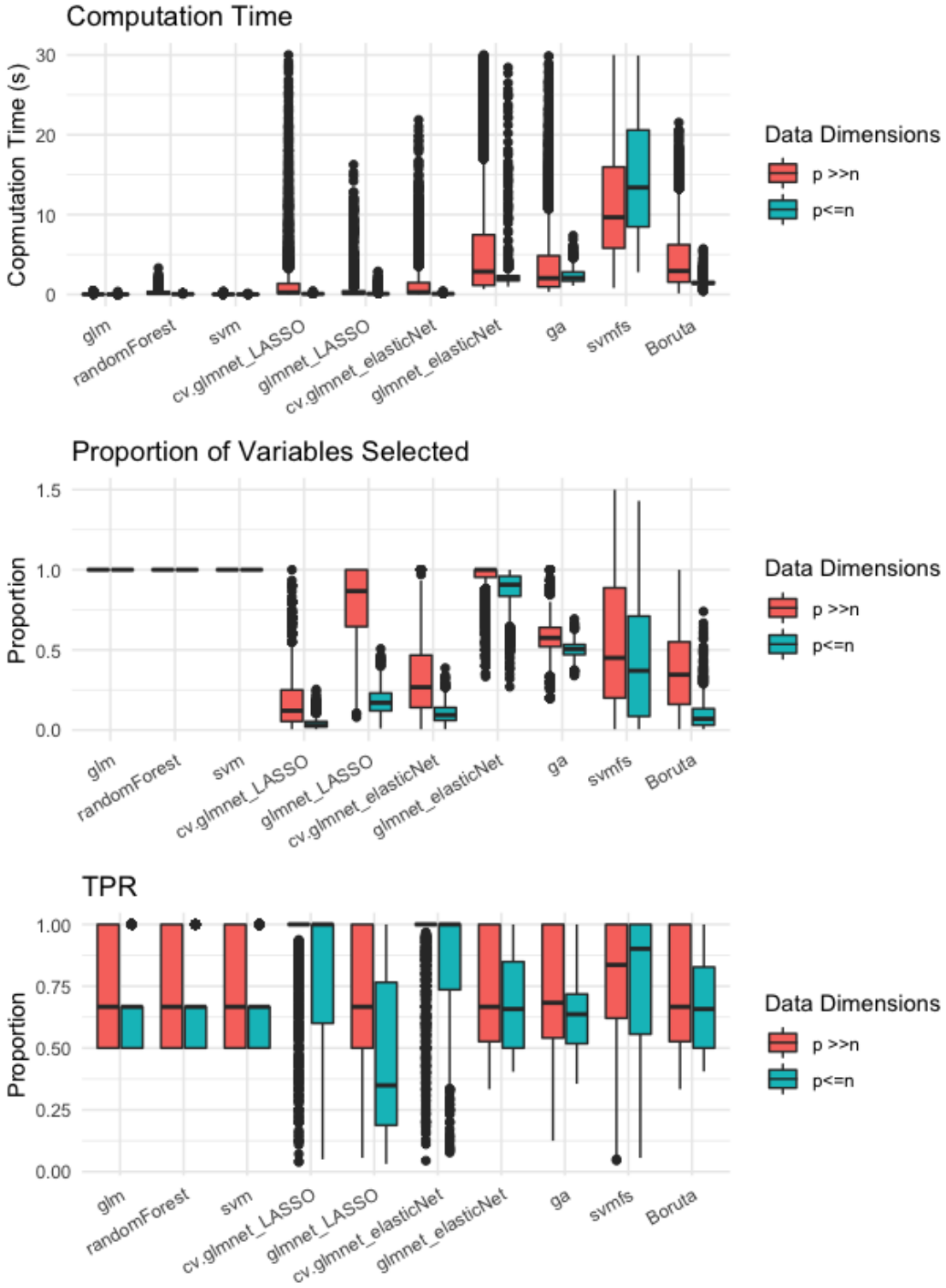

Figure 15

This figure displays boxplots of the prediction error rates and AUC for the methods as compared across type of method.

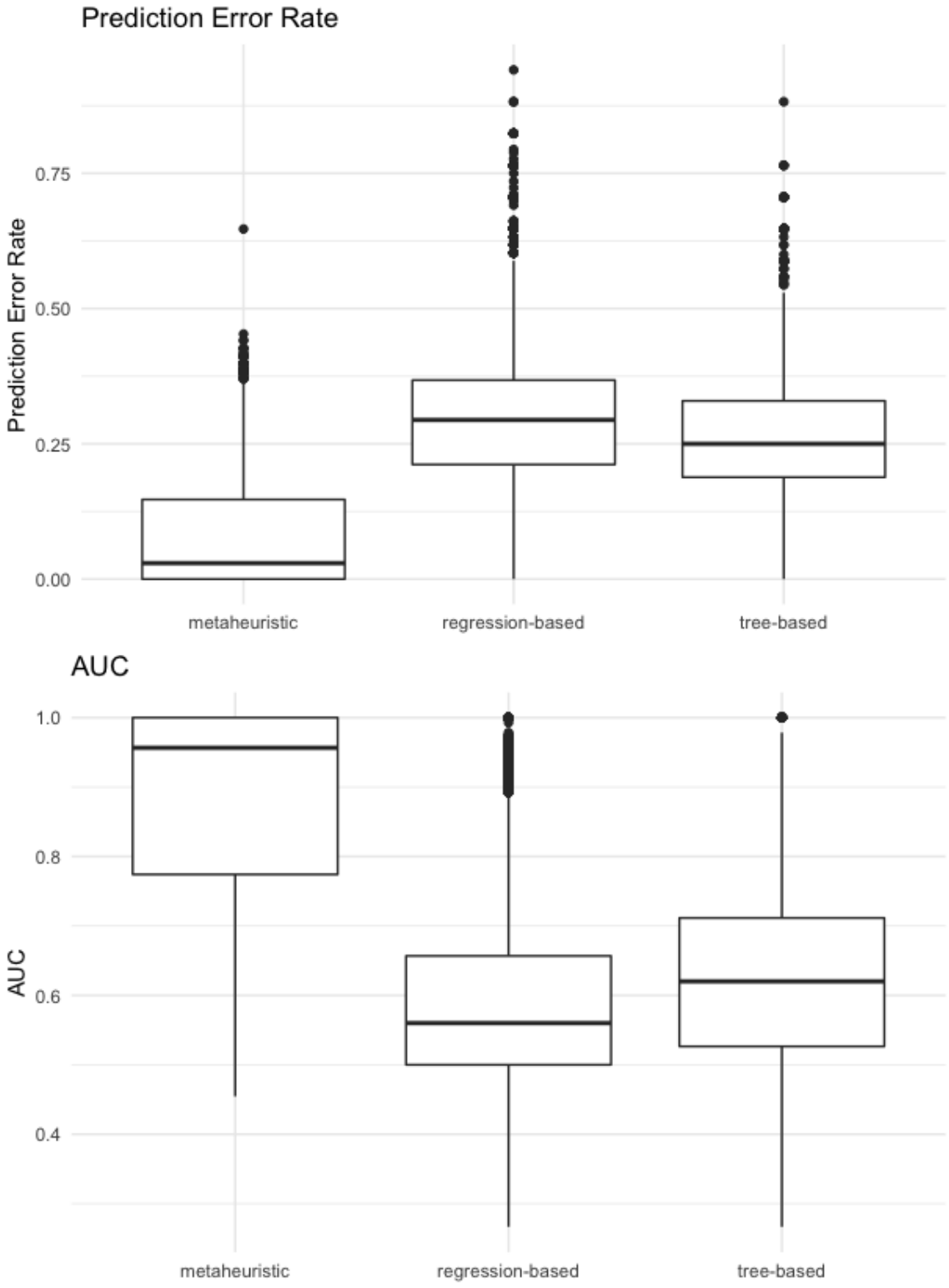

Figure 16

This figure displays boxplots of the computation times, proportion of variables selected, and TPR for the methods as compared across type of method. Note that TPR refers to the proportion of variables selected that are not noise variables and the range in computation time has been restricted to enhance visibility.

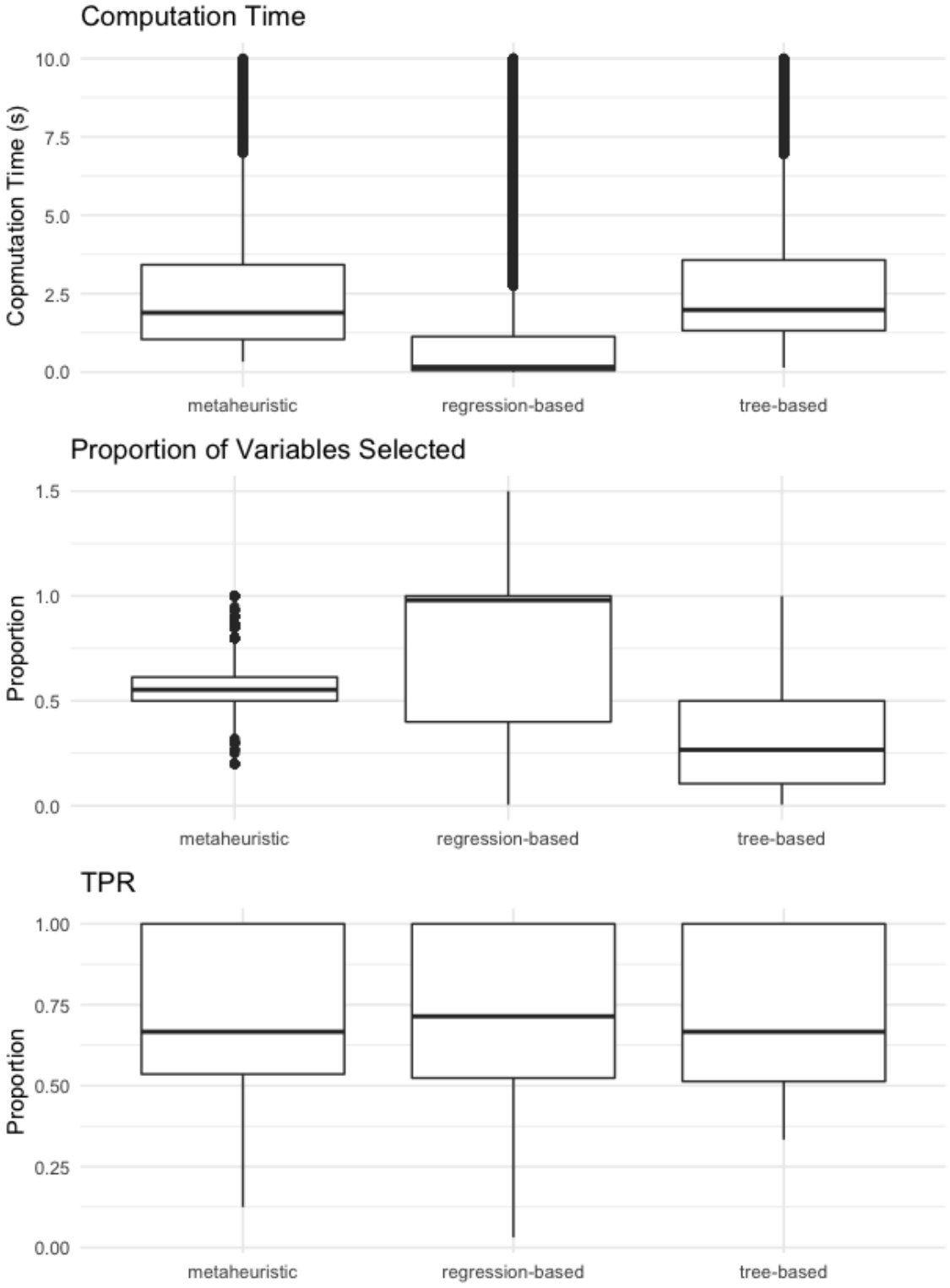

Figure 17

Variables Selected for Alcohol Use Disorder

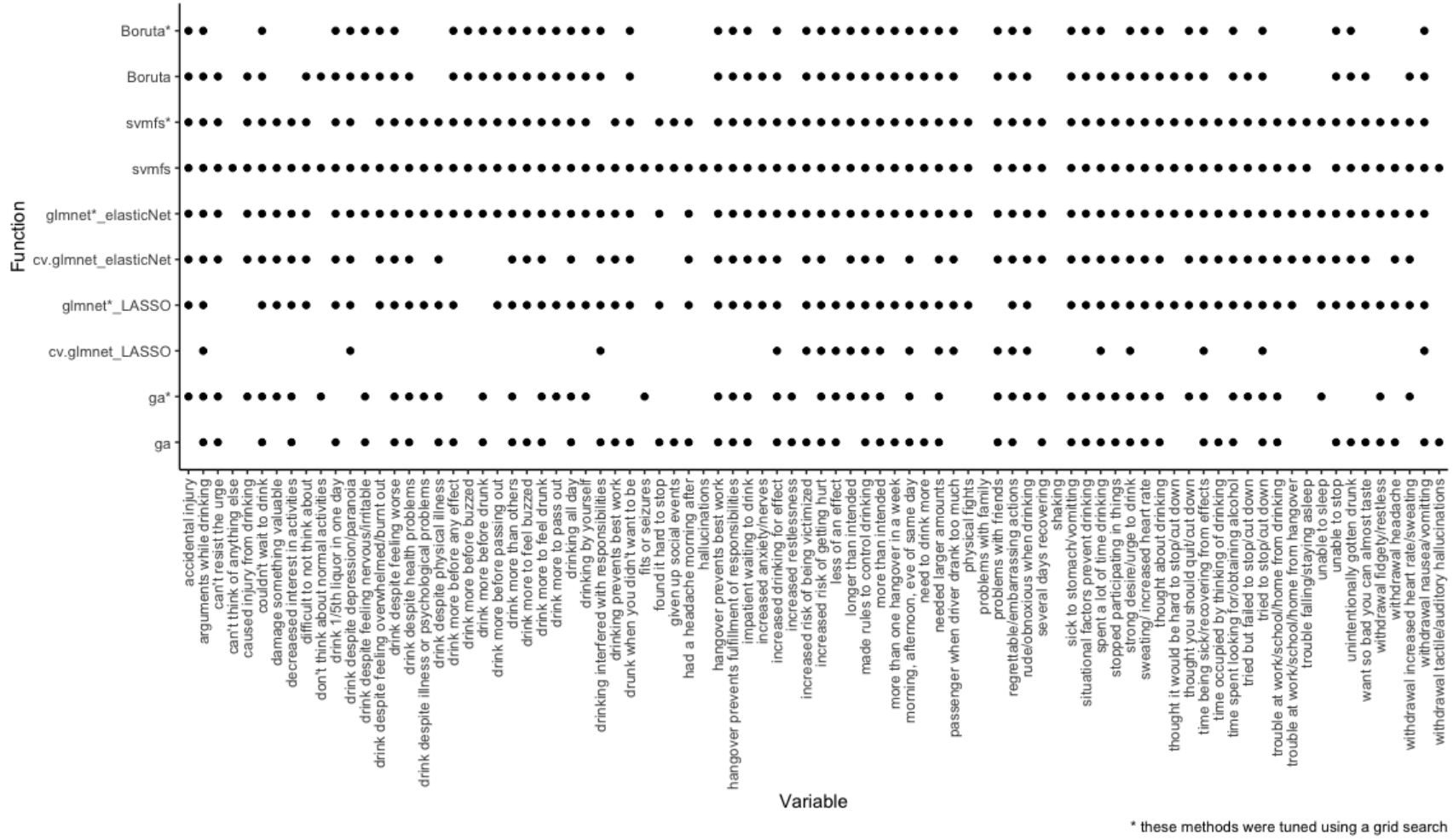

Figure 18

Variables Selected for the Misophonia Questionnaire

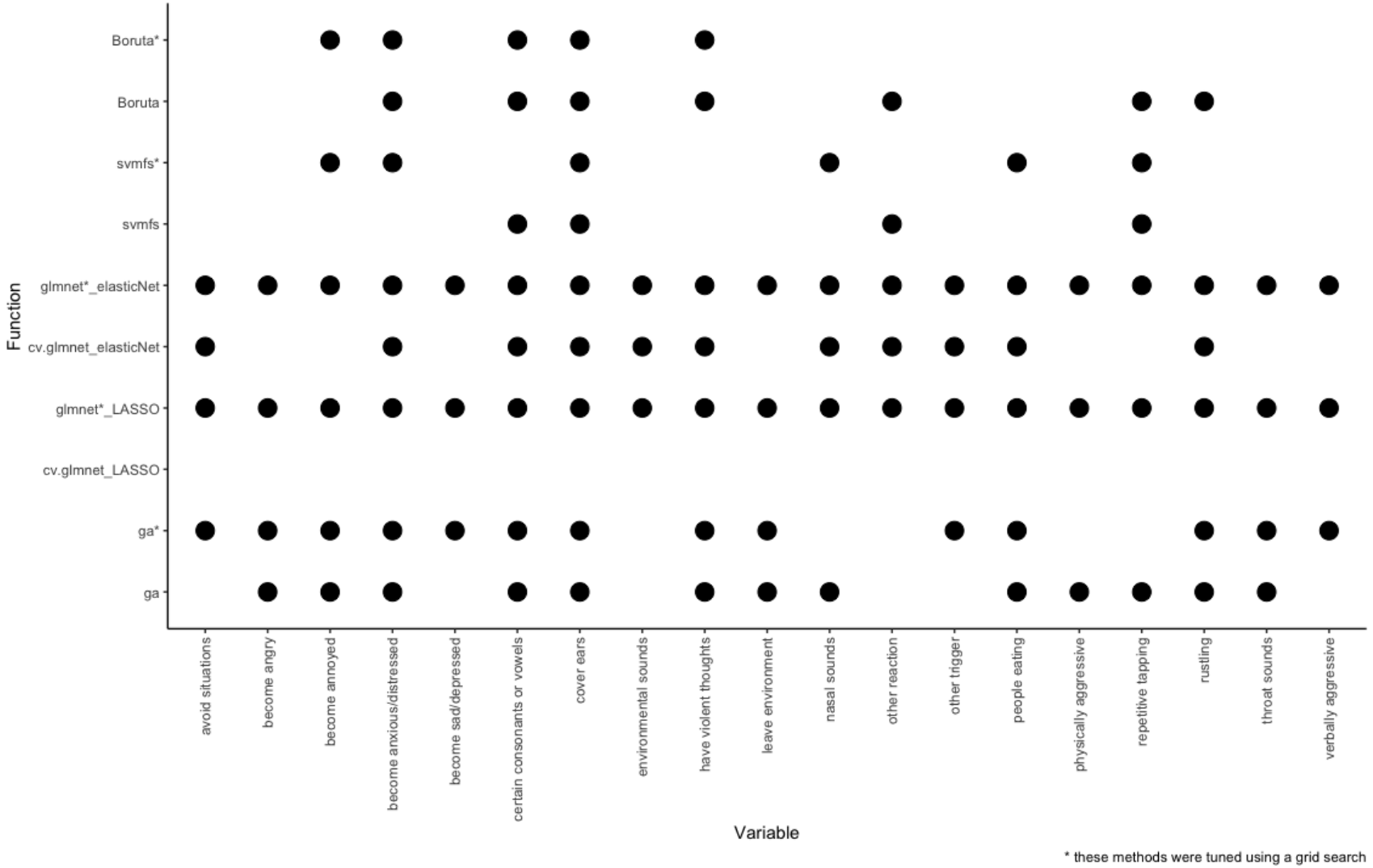

Figure 19

Variables Selected for the S-Five